# SAMV71Q21RT

## Radiation-Tolerant 32-bit Arm® Cortex®-M7 MCU

## Introduction

The SAMV71Q21RT is a radiation-tolerant Arm® Cortex®-M7-based microcontroller providing the best combination of connectivity interfaces along with highest performance levels. The embedded dual CAN-FD interface and Ethernet-AVB controller provide state-of-the-art technology for high bandwidth communication. In addition to one of the most powerful Arm cores delivering 600 DMIPS, the SAMV71Q21RT features a flexible bus and memories architecture coupled with a powerful Floating Point Unit (FPU), thus providing advanced DSP and real-time capabilities to serve the most demanding aerospace applications.

## Features

**Core**

- Arm Cortex-M7 running at up to 300 MHz
- 16 Kbytes of ICache and 16 Kbytes of DCache with Error Code Correction (ECC)
- Single- and double-precision HW Floating Point Unit (FPU)
- Memory Protection Unit (MPU) with 16 zones
- DSP Instructions, Thumb®-2 Instruction Set
- Embedded Trace Module (ETM) with instruction trace stream, including Trace Port Interface Unit (TPIU)

**Memories**

- 2048 Kbytes embedded Flash with unique identifier and user signature for user-defined data
- 384 Kbytes embedded Multi-port SRAM
- Tightly Coupled Memory (TCM) interface with four configurations (disabled, 2 x 32 Kbytes, 2 x 64 Kbytes, 2 x 128 Kbytes)
- 16 Kbytes ROM with embedded Bootloader routines (UART0, USB) and IAP routines
- 16-bit Static Memory Controller (SMC) with support for SRAM, PSRAM, LCD module, NOR and NAND Flash with on-the-fly scrambling
- 16-bit SDRAM Controller (SDRAMC) interfacing up to 256 MB and with on-the-fly scrambling
- Flash Write/Erase Cycles (Ground Level only): 10K Cycles
- Flash Data Retention:
  - 12 years with $T_A$ = 125°C
  - 26 years with $T_A$ = 110°C
  - 62 years with $T_A$ = 95°C

**System**

- Embedded voltage regulator for single-supply operation
- Power-on-Reset (POR), Brown-out Detector (BOD) and Dual Watchdog for safe operation
- Quartz or ceramic resonator oscillators: 3 to 20 MHz main oscillator with failure detection, 12 MHz or 16 MHz needed for USB operations. Optional low-power 32.768 kHz for RTC or device clock
- RTC with Gregorian Calendar mode, waveform generation in low-power modes
- RTC counter calibration circuitry compensates for 32.768 kHz crystal frequency variations
- 32-bit low-power Real-time Timer (RTT)
- High-precision Main RC oscillator with 12 MHz default frequency for device start-up. In-application trimming access for frequency adjustment. 8/12 MHz are factory-trimmed.

© 2021 Microchip Technology Inc. and its subsidiaries

- 32.768 kHz crystal oscillator or Slow RC oscillator as source of Low-Power mode device clock (SLCK)
- One 500 MHz PLL for system clock
- Temperature Sensor
- One dual-port 24-channel central DMA Controller (XDMAC)

**Low-Power Features**

- Low-power Sleep, Wait and Backup modes, with typical power consumption down to 1.1 µA in Backup mode with RTC, RTT and wake-up logic enabled
- Ultra low-power RTC and RTT
- 1 Kbyte of backup RAM (BRAM) with dedicated regulator

**Peripherals**

- One Ethernet MAC (GMAC) 10/100 Mbps in MII mode and RMII mode with dedicated DMA. IEEE®1588 PTP frames and 802.3az Energy-efficiency support. Ethernet AVB support with IEEE802.1AS Timestamping and IEEE802.1Qav credit-based traffic-shaping hardware support.
- 12-bit ITU-R BT. 601/656 Image Sensor Interface (ISI)
- Two master Controller Area Networks (MCAN) with Flexible Data Rate (CAN-FD) with SRAM-based mailboxes, time- and event-triggered transmission
- MediaLB® device with 3-wire mode, up to 1024 x Fs speed, supporting MOST25 and MOST50 networks
- Three USARTs. USART0/1/2 support LIN mode, ISO7816, IrDA®, RS-485, SPI, Manchester and Modem modes; USART1 supports LON mode.
- Five 2-wire UARTs with SleepWalking™ support
- Three Two-Wire Interfaces (TWIHS) ($I^2C$-compatible) with SleepWalking support
- Quad I/O Serial Peripheral Interface (QSPI) interfacing up to 256 MB Flash and with eXecute-In-Place and on-the-fly scrambling
- Two Serial Peripheral Interfaces (SPI)
- One Serial Synchronous Controller (SSC) with I2S and TDM support
- Two Inter-IC Sound Controllers (I2SC)
- One High-speed Multimedia Card Interface (HSMCI) (SDIO/SD Card/e.MMC)
- Four Three-Channel 16-bit Timer/Counters (TC) with Capture, Waveform, Compare and PWM modes, constant on time. Quadrature decoder logic and 2-bit Gray Up/Down Counter for stepper motor
- Two 4-channel 16-bit PWMs with complementary outputs, Dead Time Generator and eight fault inputs per PWM for motor control, two external triggers to manage power factor correction (PFC), DC-DC and lighting control.
- Two Analog Front-End Controllers (AFEC), each supporting up to 12 channels with differential input mode and programmable gain stage, allowing dual sample-and-hold at up to 1.7 Msps. Offset and gain error correction feature.
- One 2-channel 12-bit 1 Msps-per-channel Digital-to-Analog Converter (DAC) with Differential and Over Sampling modes
- One Analog Comparator Controller (ACC) with flexible input selection, selectable input hysteresis

**Cryptography**

- True Random Number Generator (TRNG)
- AES: 256-, 192-, 128-bit Key Algorithm, Compliant with FIPS PUB-197 Specifications
- Integrity Check Monitor (ICM). Supports Secure Hash Algorithm SHA1, SHA224 and SHA256.

**I/O**

- 114 I/O Lines with external interrupt capability (edge- or level-sensitivity), debouncing, glitch filtering and On-die Series Resistor Termination
- Five Parallel Input/Output Controllers (PIO)

**Operating Range**

- Temperature: -55°C to +125°C
- Single Supply Voltage: 3.0V to 3.6V
- Dual Supply Voltage
  - VDDIO: 3.0V to 3.6V

© 2021 Microchip Technology Inc. and its subsidiaries

  – VDDCORE: 1.2V to 1.32V

**Packages**

- CQFP144, 144-lead CQFP, 22 x 22 mm, pitch 0.5 mm
- LQFP144, 144-lead LQFP, 20 x 20 mm, pitch 0.5 mm

**Radiation Performance**

- No Single Event Latch-up Below an LET Threshold of 60 MeV.cm$^2$/mg @125°C
- Total Ionizing Dose of 30 krad(Si) RHA

**ESD**

- HBM 3000V
- CDM 750V

**Mass**

- CQFP144: 6187 mg
- LQFP144: 1365 mg

© 2021 Microchip Technology Inc.
and its subsidiaries

# Table of Contents

Introduction....1

Features....1

1. Configuration Summary....14

2. Ordering Information....16

3. Block Diagram....17

4. Signal Description....18

5. Space Quality Grade....25

6. Package and Pinout....26
   - 6.1. CQFP Package Outline....26
   - 6.2. LQFP Package Outline....27
   - 6.3. Pinout....27

7. Power Considerations....30
   - 7.1. Power Supplies....30
   - 7.2. Power Constraints....30
   - 7.3. Voltage Regulator....31
   - 7.4. Backup SRAM Power Switch....31
   - 7.5. Active Mode....32
   - 7.6. Low-power Modes....32
   - 7.7. Wakeup Sources....36
   - 7.8. Fast Startup....36

8. Input/Output Lines....37
   - 8.1. General-Purpose I/O Lines....37
   - 8.2. System I/O Lines....37
   - 8.3. NRST Pin....38
   - 8.4. ERASE Pin....38

9. Interconnect....40

10. Product Mapping....41

11. Memories....42
   - 11.1. Embedded Memories....42
   - 11.2. External Memories....48

12. Event System....49
   - 12.1. Embedded Characteristics....49
   - 12.2. Real-time Event Mapping....50

13. System Controller....54
   - 13.1. System Controller and Peripherals Mapping....54
   - 13.2. Power-on-Reset, Brownout and Supply Monitor....54

© 2021 Microchip Technology Inc. and its subsidiaries

13.3. Reset Controller ..... 54

14. Peripherals ..... 55
14.1. Peripheral Identifiers ..... 55
14.2. Peripheral Signal Multiplexing on I/O Lines ..... 57

15. ARM Cortex-M7 (ARM) ..... 58
15.1. ARM Cortex-M7 Configuration ..... 58

16. Debug and Test Features ..... 59
16.1. Description ..... 59
16.2. Embedded Characteristics ..... 59
16.3. Associated Documents ..... 59
16.4. Debug and Test Block Diagram ..... 60
16.5. Debug and Test Pin Description ..... 60
16.6. Application Examples ..... 61
16.7. Functional Description ..... 62

17. SAM-BA Boot Program ..... 66
17.1. Description ..... 66
17.2. Embedded Characteristics ..... 66
17.3. Hardware and Software Constraints ..... 66
17.4. Flow Diagram ..... 66
17.5. Device Initialization ..... 67
17.6. SAM-BA Monitor ..... 67

18. Fast Flash Programming Interface (FFPI) ..... 71
18.1. Description ..... 71
18.2. Embedded Characteristics ..... 71
18.3. Parallel Fast Flash Programming ..... 71

19. Bus Matrix (MATRIX) ..... 79
19.1. Description ..... 79
19.2. Embedded Characteristics ..... 79
19.3. Functional Description ..... 81
19.4. Register Summary ..... 85

20. USB Transmitter Macrocell Interface (UTMI) ..... 103
20.1. Description ..... 103
20.2. Embedded Characteristics ..... 103
20.3. Register Summary ..... 104

21. Chip Identifier (CHIPID) ..... 107
21.1. Description ..... 107
21.2. Embedded Characteristics ..... 107
21.3. Register Summary ..... 108

22. Enhanced Embedded Flash Controller (EEFC) ..... 112
22.1. Description ..... 112
22.2. Embedded Characteristics ..... 112

© 2021 Microchip Technology Inc.
and its subsidiaries

22.3. Product Dependencies ..... 112
22.4. Functional Description ..... 112
22.5. Register Summary ..... 129

23. Supply Controller (SUPC) ..... 137
23.1. Description ..... 137
23.2. Embedded Characteristics ..... 137
23.3. Block Diagram ..... 138
23.4. Functional Description ..... 139
23.5. Register Summary ..... 149

24. Watchdog Timer (WDT) ..... 160
24.1. Description ..... 160
24.2. Embedded Characteristics ..... 160
24.3. Block Diagram ..... 160
24.4. Functional Description ..... 161
24.5. Register Summary ..... 163

25. Reinforced Safety Watchdog Timer (RSWDT) ..... 168
25.1. Description ..... 168
25.2. Embedded Characteristics ..... 168
25.3. Block Diagram ..... 169
25.4. Functional Description ..... 169
25.5. Register Summary ..... 171

26. Reset Controller (RSTC) ..... 176
26.1. Description ..... 176
26.2. Embedded Characteristics ..... 176
26.3. Block Diagram ..... 176
26.4. Functional Description ..... 177

27. Real-time Clock (RTC) ..... 187
27.1. Description ..... 187
27.2. Embedded Characteristics ..... 187
27.3. Block Diagram ..... 187
27.4. Product Dependencies ..... 188
27.5. Functional Description ..... 188
27.6. Register Summary ..... 196

28. Real-time Timer (RTT) ..... 214
28.1. Description ..... 214
28.2. Embedded Characteristics ..... 214
28.3. Block Diagram ..... 214
28.4. Functional Description ..... 214
28.5. Register Summary ..... 217

29. General Purpose Backup Registers (GPBR) ..... 223
29.1. Description ..... 223
29.2. Embedded Characteristics ..... 223

© 2021 Microchip Technology Inc.
and its subsidiaries

29.3. Register Summary....224

30. Clock Generator....226
30.1. Description....226
30.2. Embedded Characteristics....226
30.3. Block Diagram....227
30.4. Slow Clock....227
30.5. Main Clock....228
30.6. PLLA Clock....231
30.7. UTMI PLL Clock....232

31. Power Management Controller (PMC)....234
31.1. Description....234
31.2. Embedded Characteristics....234
31.3. Block Diagram....235
31.4. Master Clock Controller....235
31.5. Processor Clock Controller....235
31.6. SysTick External Clock....235
31.7. USB Full-speed Clock Controller....236
31.8. Core and Bus Independent Clocks for Peripherals....236
31.9. Peripheral and Generic Clock Controller....236
31.10. Asynchronous Partial Wakeup....237
31.11. Free-running Processor Clock....239
31.12. Programmable Clock Output Controller....239
31.13. Fast Startup....239
31.14. Startup from Embedded Flash....240
31.15. Main Crystal Oscillator Failure Detection....240
31.16. 32.768 kHz Crystal Oscillator Frequency Monitor....241
31.17. Recommended Programming Sequence....242
31.18. Clock Switching Details....244
31.19. Register Write Protection....246
31.20. Register Summary....248

32. Parallel Input/Output Controller (PIO)....294
32.1. Description....294
32.2. Embedded Characteristics....294
32.3. Block Diagram....295
32.4. Product Dependencies....295
32.5. Functional Description....296
32.6. Register Summary....308

33. External Bus Interface (EBI)....369
33.1. Description....369
33.2. Embedded Characteristics....369
33.3. EBI Block Diagram....370
33.4. I/O Lines Description....370
33.5. Application Example....372

34. SDRAM Controller (SDRAMC)....376

© 2021 Microchip Technology Inc.
and its subsidiaries

34.1. Description ... 376
34.2. Embedded Characteristics ... 376
34.3. Signal Description ... 376
34.4. Software Interface/SDRAM Organization, Address Mapping ... 377
34.5. Product Dependencies ... 378
34.6. Functional Description ... 379
34.7. Register Summary ... 385

35. Static Memory Controller (SMC) ... 401
35.1. Description ... 401
35.2. Embedded Characteristics ... 401
35.3. I/O Lines Description ... 401
35.4. Multiplexed Signals ... 402
35.5. Product Dependencies ... 402
35.6. External Memory Mapping ... 402
35.7. Connection to External Devices ... 403
35.8. Application Example ... 406
35.9. Standard Read and Write Protocols ... 408
35.10. Scrambling/Unscrambling Function ... 415
35.11. Automatic Wait States ... 416
35.12. Data Float Wait States ... 419
35.13. External Wait ... 422
35.14. Slow Clock Mode ... 426
35.15. Asynchronous Page Mode ... 428
35.16. Register Summary ... 431

36. DMA Controller (XDMAC) ... 442
36.1. Description ... 442
36.2. Embedded Characteristics ... 442
36.3. Block Diagram ... 443
36.4. DMA Controller Peripheral Connections ... 443
36.5. Functional Description ... 445
36.6. Linked List Descriptor Operation ... 448
36.7. XDMAC Maintenance Software Operations ... 451
36.8. XDMAC Software Requirements ... 451
36.9. Register Summary ... 453

37. Image Sensor Interface (ISI) ... 515
37.1. Description ... 515
37.2. Embedded Characteristics ... 516
37.3. Block Diagram ... 516
37.4. Product Dependencies ... 516
37.5. Functional Description ... 517
37.6. Register Summary ... 526

38. GMAC - Ethernet MAC ... 560
38.1. Description ... 560
38.2. Embedded Characteristics ... 560
38.3. Block Diagram ... 561

38.4. Signal Interface ..... 561
38.5. Product Dependencies ..... 562
38.6. Functional Description ..... 562
38.7. Programming Interface ..... 589
38.8. Register Summary ..... 593

39. USB High Speed Interface (USBHS) ..... 737
39.1. Description ..... 737
39.2. Embedded Characteristics ..... 737
39.3. Block Diagram ..... 738
39.4. Product Dependencies ..... 739
39.5. Functional Description ..... 739
39.6. Register Summary ..... 761

40. High-Speed Multimedia Card Interface (HSMCI) ..... 912
40.1. Description ..... 912
40.2. Embedded Characteristics ..... 912
40.3. Block Diagram ..... 913
40.4. Application Block Diagram ..... 913
40.5. Pin Name List ..... 914
40.6. Product Dependencies ..... 914
40.7. Bus Topology ..... 914
40.8. High-Speed Multimedia Card Operations ..... 916
40.9. SD/SDIO Card Operation ..... 925
40.10. CE-ATA Operation ..... 925
40.11. HSMCI Boot Operation Mode ..... 926
40.12. HSMCI Transfer Done Timings ..... 927
40.13. Register Write Protection ..... 928
40.14. Register Summary ..... 929

41. Serial Peripheral Interface (SPI) ..... 959
41.1. Description ..... 959
41.2. Embedded Characteristics ..... 959
41.3. Block Diagram ..... 960
41.4. Application Block Diagram ..... 960
41.5. Signal Description ..... 961
41.6. Product Dependencies ..... 961
41.7. Functional Description ..... 961
41.8. Register Summary ..... 974

42. Quad Serial Peripheral Interface (QSPI) ..... 991
42.1. Description ..... 991
42.2. Embedded Characteristics ..... 991
42.3. Block Diagram ..... 992
42.4. Signal Description ..... 992
42.5. Product Dependencies ..... 992
42.6. Functional Description ..... 993
42.7. Register Summary ..... 1009

© 2021 Microchip Technology Inc.
and its subsidiaries

43. Two-wire Interface (TWIHS)....1030
43.1. Description....1030
43.2. Embedded Characteristics....1030
43.3. List of Abbreviations....1031
43.4. Block Diagram....1031
43.5. Product Dependencies....1032
43.6. Functional Description....1032
43.7. Register Summary....1070

44. Synchronous Serial Controller (SSC)....1097
44.1. Description....1097
44.2. Embedded Characteristics....1097
44.3. Block Diagram....1098
44.4. Application Block Diagram....1098
44.5. SSC Application Examples....1098
44.6. Pin Name List....1100
44.7. Product Dependencies....1100
44.8. Functional Description....1101
44.9. Register Summary....1111

45. Inter-IC Sound Controller (I2SC)....1139
45.1. Description....1139
45.2. Embedded Characteristics....1139
45.3. Block Diagram....1140
45.4. I/O Lines Description....1140
45.5. Product Dependencies....1140
45.6. Functional Description....1141
45.7. I2SC Application Examples....1145
45.8. Register Summary....1149

46. Universal Synchronous Asynchronous Receiver Transceiver (USART)....1164
46.1. Description....1164
46.2. Embedded Characteristics....1164
46.3. Block Diagram....1166
46.4. I/O Lines Description....1166
46.5. Product Dependencies....1167
46.6. Functional Description....1167
46.7. Register Summary....1215

47. Universal Asynchronous Receiver Transmitter (UART)....1287
47.1. Description....1287
47.2. Embedded Characteristics....1287
47.3. Block Diagram....1287
47.4. Product Dependencies....1288
47.5. Functional Description....1288
47.6. Register Summary....1297

48. Media Local Bus (MLB)....1311
48.1. Description....1311

48.2. Embedded Characteristics ..... 1312
48.3. Block Diagram ..... 1312
48.4. Signal Description ..... 1313
48.5. Product Dependencies ..... 1313
48.6. Functional Description ..... 1314
48.7. Register Summary ..... 1355

49. Controller Area Network (MCAN) ..... 1389
49.1. Description ..... 1389
49.2. Embedded Characteristics ..... 1389
49.3. Block Diagram ..... 1390
49.4. Product Dependencies ..... 1390
49.5. Functional Description ..... 1391
49.6. Register Summary ..... 1416

50. Timer Counter (TC) ..... 1476
50.1. Description ..... 1476
50.2. Embedded Characteristics ..... 1476
50.3. Block Diagram ..... 1477
50.4. Pin List ..... 1478
50.5. Product Dependencies ..... 1478
50.6. Functional Description ..... 1478
50.7. Register Summary ..... 1499

51. Pulse Width Modulation Controller (PWM) ..... 1531
51.1. Description ..... 1531
51.2. Embedded Characteristics ..... 1531
51.3. Block Diagram ..... 1533
51.4. I/O Lines Description ..... 1533
51.5. Product Dependencies ..... 1534
51.6. Functional Description ..... 1536
51.7. Register Summary ..... 1574

52. Analog Front-End Controller (AFEC) ..... 1638
52.1. Description ..... 1638
52.2. Embedded Characteristics ..... 1638
52.3. Block Diagram ..... 1639
52.4. Signal Description ..... 1639
52.5. Product Dependencies ..... 1640
52.6. Functional Description ..... 1640
52.7. Register Summary ..... 1656

53. Digital-to-Analog Converter Controller (DACC) ..... 1690
53.1. Description ..... 1690
53.2. Embedded Characteristics ..... 1690
53.3. Block Diagram ..... 1691
53.4. Signal Description ..... 1691
53.5. Product Dependencies ..... 1692
53.6. Functional Description ..... 1692

53.7. Register Summary....1698

54. Analog Comparator Controller (ACC)....1714
54.1. Description....1714
54.2. Embedded Characteristics....1714
54.3. Block Diagram....1714
54.4. Signal Description....1715
54.5. Product Dependencies....1715
54.6. Functional Description....1715
54.7. Register Summary....1717

55. Integrity Check Monitor (ICM)....1728
55.1. Description....1728
55.2. Embedded Characteristics....1729
55.3. Block Diagram....1729
55.4. Product Dependencies....1730
55.5. Functional Description....1730
55.6. Register Summary....1743

56. True Random Number Generator (TRNG)....1762
56.1. Description....1762
56.2. Embedded Characteristics....1762
56.3. Block Diagram....1762
56.4. Product Dependencies....1762
56.5. Functional Description....1762
56.6. Register Summary....1764

57. Advanced Encryption Standard (AES)....1771
57.1. Description....1771
57.2. Embedded Characteristics....1771
57.3. Product Dependencies....1771
57.4. Functional Description....1772
57.5. Register Summary....1783

58. Electrical Characteristics....1803
58.1. Absolute Maximum Ratings....1803
58.2. DC Characteristics....1803
58.3. Power Consumption....1808
58.4. Oscillator Characteristics....1812
58.5. PLLA Characteristics....1816
58.6. PLLUSB Characteristics....1816
58.7. USB Transceiver Characteristics....1816
58.8. AFE Characteristics....1817
58.9. Analog Comparator Characteristics....1824
58.10. Temperature Sensor....1825
58.11. 12-bit DAC Characteristics....1826
58.12. Embedded Flash Characteristics....1828
58.13. Timings for STH Conditions....1829

© 2021 Microchip Technology Inc.
and its subsidiaries

59. Schematic Checklist....1848
59.1. Power Supplies....1848
59.2. General Hardware Recommendations....1854
59.3. Boot Program Hardware Constraints....1865
60. Marking....1867
60.1. CQFP144....1867
60.2. LQFP144....1867
61. Mechanical Characteristics....1869
61.1. 144-Lead Plastic Quad Flatpack (2SB) - 20x20x1.4 mm Body [LQFP] Atmel Legacy Global Package Code AEI....1869
61.2. 144-pin CQFP Package....1872
62. Errata....1874
62.1. USB High-Speed (USBHS)....1874
62.2. Other Modules....1874
63. Revision History....1875
63.1. Rev. D - 7/2021....1875
63.2. Rev. C - 8/2019....1875
63.3. Rev. B - 6/2019....1875
63.4. Rev. A - 12/2018....1875
The Microchip Website....1876
Product Change Notification Service....1876
Customer Support....1876
Product Identification System....1877
Microchip Devices Code Protection Feature....1877
Legal Notice....1877
Trademarks....1878
Quality Management System....1878
Worldwide Sales and Service....1879

© 2021 Microchip Technology Inc. and its subsidiaries

# 1. Configuration Summary

**Table 1-1. Configuration Summary**

| Feature | SAM V71Q21RT |
|---|---|
| Flash (Kbytes) | 2048 |
| Multi-port SRAM (Kbytes) | 384 |
| Cache(I/D) (Kbytes) | 16/16 |
| Packages | CQFP144<br>LQFP144 |
| Number of PIOs | 114 |
| External Bus Interface | 16-bit data, 4 chip selects, 24-bit address |
| SDRAM Interface | Yes |
| Media LB Interface | Yes |
| Central DMA | 24 |
| 12-bit ADC | 24 ch. (see **Note 1**) |
| 12-bit DAC | 2 ch. |
| Timer Counter Channels | 12 |
| Timer Counter Channels I/O | 36 |
| USART/UART | 3/5(see **Note 2**) |
| QSPI | Yes |
| SPI0 | Yes |
| SPI1 | Yes |
| USART SPI | 3 |
| TWI | 3 |
| HSMCI | 1 port<br>4 bits |
| CAN | 2 ports |
| GMAC | MII, RMII |
| ISI | 12-bit |
| SSC | Yes |
| I2SC | 2 |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | |
|---|---|
| **Feature** | **SAM V71Q21RT** |
| USB | Not supported |
| Analog Comparator | Yes |
| Embedded Trace Macrocell (ETM) | Yes |

**Notes:**

1. One channel is reserved for internal temperature sensor.
2. LON support on USART1 only.

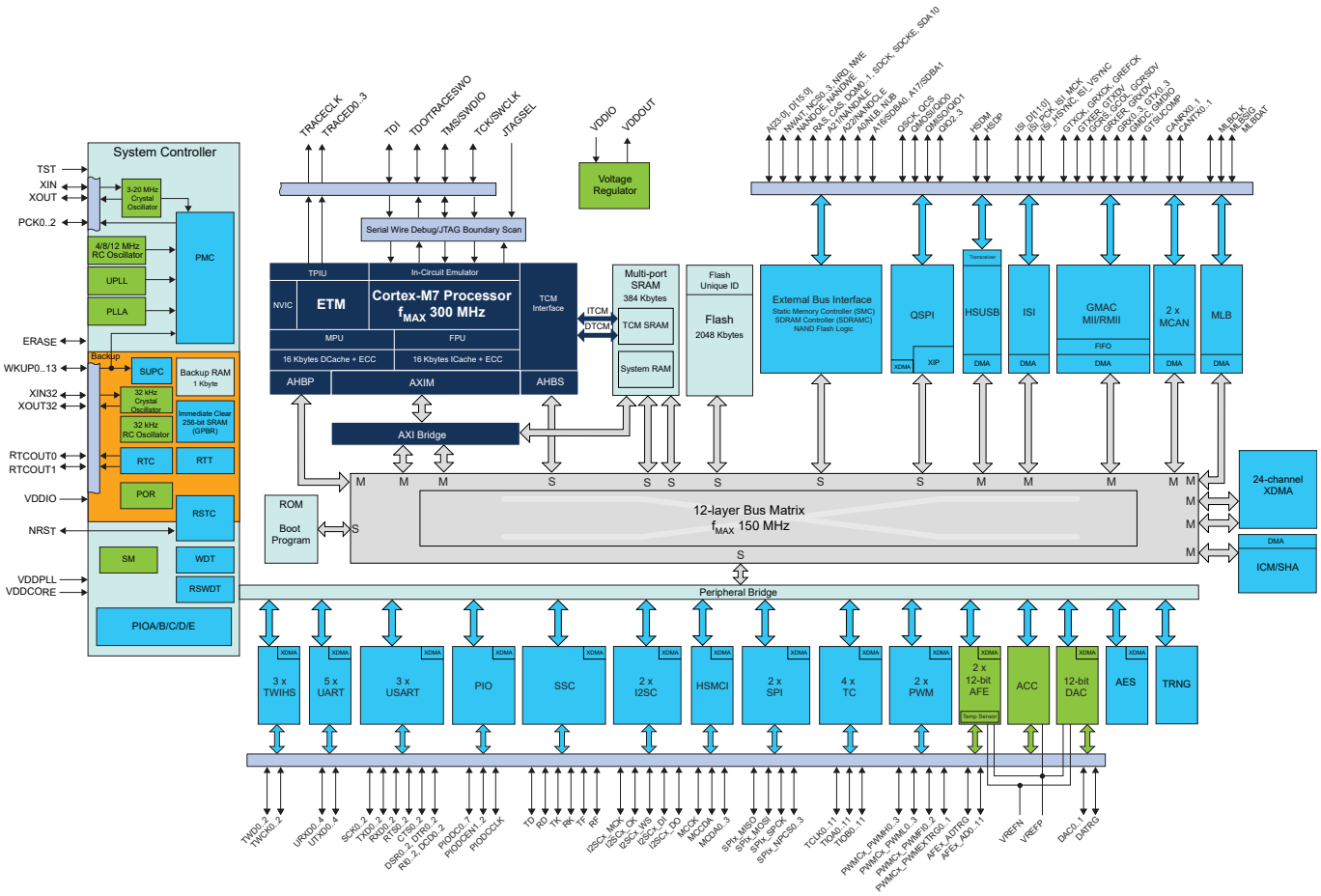
## 2. Ordering Information

| Ordering Code | Speed (MHz) | Power Supply | Package | Flow |
|---|---|---|---|---|
| SAMV71Q21RT-DHB-E | 300 | 3.0–3.6V | CQFP144 | Engineering Samples |
| SAMV71Q21RT-DHB-HC | | | | Hirel Ceramic |
| SAMV71Q21RT-DHB-MQ | | | | QML-Q Equivalent |
| SAMV71Q21RT-DHB-SV | | | | QML-V Equivalent |
| SAMV71Q21RT-H8X-HP | | | LQFP144 | Hirel Plastic |
| SAMV71Q21RT-H8X-SN | | | | QML-N Equivalent |

© 2021 Microchip Technology Inc.
and its subsidiaries

# 3. Block Diagram

**Figure 3-1. SAMV71Q21RT Block Diagram**



# 4. Signal Description

The following table provides details on signal names classified by peripheral.

**Table 4-1. Signal Description List**

| Signal Name | Function | Type | Active Level | Voltage Reference | Comments |
|---|---|---|---|---|---|
| Power Supplies | | | | | |
| VDDIO | Peripherals I/O Lines Power Supply | Power | – | – | – |
| VDDIN | Voltage Regulator Input, AFE, DAC and Analog Comparator Power Supply (see **Note**) | Power | – | – | – |
| VDDOUT | Voltage Regulator Output | Power | – | – | – |
| VDDPLL | PLLA Power Supply | Power | – | – | – |
| VDDPLLUSB | USB PLL and Oscillator Power Supply | Power | – | – | – |
| VDDCORE | Powers the core, the embedded memories and the peripherals | Power | – | – | – |
| GND, GNDPLL, GNDPLLUSB, GNDANA, GNDUTMI | Ground | Ground | – | – | – |
| VDDUTMII | USB Transceiver Power Supply | Power | – | – | – |
| VDDUTMIC | USB Core Power Supply | Power | – | – | – |
| GNDUTMI | USB Ground | Ground | – | – | – |
| Clocks, Oscillators and PLLs | | | | | |
| XIN | Main Oscillator Input | Input | – | VDDIO | – |
| XOUT | Main Oscillator Output | Output | – | | – |
| XIN32 | Slow Clock Oscillator Input | Input | – | | – |
| XOUT32 | Slow Clock Oscillator Output | Output | – | | – |
| PCK0–PCK2 | Programmable Clock Output | Output | – | | – |
| Real Time Clock | | | | | |
| RTCOUT0 | Programmable RTC Waveform Output | Output | – | VDDIO | – |
| RTCOUT1 | Programmable RTC Waveform Output | Output | – | | – |
| Serial Wire Debug/JTAG Boundary Scan | | | | | |

.....continued

| Signal Name | Function | Type | Active Level | Voltage Reference | Comments |
|---|---|---|---|---|---|
| SWCLK/TCK | Serial Wire Clock / Test Clock (Boundary scan mode only) | Input | – | VDDIO | – |
| TDI | Test Data In (Boundary scan mode only) | Input | – | | – |
| TDO/TRACESWO | Test Data Out (Boundary scan mode only) | Output | – | | – |
| SWDIO/TMS | Serial Wire Input/Output / Test Mode Select (Boundary scan mode only) | I/O / Input | – | | – |
| JTAGSEL | JTAG Selection | Input | High | | – |
| Trace Debug Port | | | | | |
| TRACECLK | Trace Clock | Output | – | VDDIO | PCK3 is used for ETM |
| TRACED0–TRACED3 | Trace Data | Output | – | | – |
| Flash Memory | | | | | |
| ERASE | Flash and NVM Configuration Bits Erase Command | Input | High | VDDIO | – |
| Reset/Test | | | | | |
| NRST | Synchronous Microcontroller Reset | I/O | Low | VDDIO | – |
| TST | Test Select | Input | – | | – |
| Universal Asynchronous Receiver Transceiver - UART(x=[0:4]) | | | | | |
| URXDx | UART Receive Data | Input | – | – | PCK4 can be used to generate the baud rate |
| UTXDx | UART Transmit Data | Output | – | – | |
| PIO Controller - PIOA - PIOB - PIOC - PIOD - PIOE | | | | | |
| PA0–PA31 | Parallel IO Controller A | I/O | – | VDDIO | – |
| PB0–PB9, PB12–PB13 | Parallel IO Controller B | I/O | – | | – |
| PC0– PC31 | Parallel IO Controller C | I/O | – | | – |
| PD0–PD31 | Parallel IO Controller D | I/O | – | – | – |
| PE0–PE5 | Parallel IO Controller E | I/O | – | – | – |
| PIO Controller - Parallel Capture Mode | | | | | |

.....continued

| Signal Name | Function | Type | Active Level | Voltage Reference | Comments |
|---|---|---|---|---|---|
| PIODC0–PIODC7 | Parallel Capture Mode Data | Input | – | VDDIO | – |
| PIODCCLK | Parallel Capture Mode Clock | Input | – | | – |
| PIODCEN1–PIODCEN2 | Parallel Capture Mode Enable | Input | – | | – |
| External Bus Interface | | | | | |
| D[15:0] | Data Bus | I/O | – | – | – |
| A[23:0] | Address Bus | Output | – | – | – |
| NWAIT | External Wait Signal | Input | Low | – | – |
| Static Memory Controller - SMC | | | | | |
| NCS0–NCS3 | Chip Select Lines | Output | Low | – | – |
| NRD | Read Signal | Output | Low | – | – |
| NWE | Write Enable | Output | Low | – | – |
| NWR0–NWR1 | Write Signal | Output | Low | – | – |
| NBS0–NBS1 | Byte Mask Signal | Output | Low | – | Used also for SDRAMC |
| NAND Flash Logic | | | | | |
| NANDOE | NAND Flash Output Enable | Output | Low | – | – |
| NANDWE | NAND Flash Write Enable | Output | Low | – | – |
| SDR-SDRAM Controller Logic | | | | | |
| SDCK | SDRAM Clock | Output | – | – | – |
| SDCKE | SDRAM Clock Enable | Output | – | – | – |
| SDCS | SDRAM Controller Chip Select | Output | – | – | – |
| BA0–BA1 | Bank Select | Output | – | – | – |
| SDWE | SDRAM Write Enable | Output | – | – | – |
| RAS–CAS | Row and Column Signal | Output | – | – | – |
| SDA10 | SDRAM Address 10 Line | Output | – | – | – |
| High Speed Multimedia Card Interface - HSMCI | | | | | |
| MCCK | Multimedia Card Clock | O | – | – | – |
| MCCDA | Multimedia Card Slot A Command | I/O | – | – | – |
| MCDA0–MCDA3 | Multimedia Card Slot A Data | I/O | – | – | – |
| Universal Synchronous Asynchronous Receiver Transmitter USART(x=[0:2]) | | | | | |

.....continued

| Signal Name | Function | Type | Active Level | Voltage Reference | Comments |
|---|---|---|---|---|---|
| SCKx | USARTx Serial Clock | I/O | – | – | PCK4 can be used to generate the baud rate |
| TXDx | USARTx Transmit Data | I/O | – | – | |
| RXDx | USARTx Receive Data | Input | – | – | |
| RTSx | USARTx Request To Send | Output | – | – | |
| CTSx | USARTx Clear To Send | Input | – | – | |
| DTRx | USARTx Data Terminal Ready | Output | – | – | |
| DSRx | USARTx Data Set Ready | Input | – | – | |
| DCDx | USARTx Data Carrier Detect | Input | – | – | |
| RIx | USARTx Ring Indicator | Input | – | – | |
| LONCOL1 | LON Collision Detection | Input | – | – | |
| Synchronous Serial Controller - SSC | | | | | |
| TD | SSC Transmit Data | Output | – | – | – |
| RD | SSC Receive Data | Input | – | – | – |
| TK | SSC Transmit Clock | I/O | – | – | – |
| RK | SSC Receive Clock | I/O | – | – | – |
| TF | SSC Transmit Frame Sync | I/O | – | – | – |
| RF | SSC Receive Frame Sync | I/O | – | – | – |
| Inter-IC Sound Controller - I2SC[1..0] | | | | | |
| I2SCx_MCK | Master Clock | Output | – | VDDIO | GCLK[PID] can be used to generate the baud rate |
| I2SCx_CK | Serial Clock | I/O | – | VDDIO | |
| I2SCx_WS | $I^2S$ Word Select | I/O | – | VDDIO | |
| I2SCx_DI | Serial Data Input | Input | – | VDDIO | |
| I2SCx_DO | Serial Data Output | Output | – | VDDIO | |
| Image Sensor Interface - ISI | | | | | |
| ISI_D0–ISI_D11 | Image Sensor Data | Input | – | – | – |
| ISI_MCK | Image sensor Reference clock.<br>No dedicated signal, PCK1 can be used. | Output | – | – | – |
| ISI_HSYNC | Image Sensor Horizontal Synchro | Input | – | – | – |
| ISI_VSYNC | Image Sensor Vertical Synchro | Input | – | – | – |
| ISI_PCK | Image Sensor Data clock | Input | – | – | – |

.....continued

| Signal Name | Function | Type | Active Level | Voltage Reference | Comments |
|---|---|---|---|---|---|
| Timer Counter - TC(x=[0:11]) | | | | | |
| TCLKx | TC Channel x External Clock Input | Input | – | – | PCK6 can be used as an input clock PCK7 can be used as an input clock for TC0 only |
| TIOAx | TC Channel x I/O Line A | I/O | – | – | |
| TIOBx | TC Channel x I/O Line B | I/O | – | – | |
| Pulse Width Modulation Controller- PWMC(x=[0..1]) | | | | | |
| PWMCx_PWMH0–PWMCx_PWMH3 | Waveform Output High for Channel 0–3 | Output | – | – | – |
| PWMCx_PWML0–PWMCx_PWML3 | Waveform Output Low for Channel 0–3 | Output | – | – | Only output in complementary mode when dead time insertion is enabled. |
| PWMCx_PWMFI0–PWMCx_PWMFI2 | Fault Input | Input | – | – | – |
| PWMCx_PWMEXTRG0–PWMCx_PWMEXTRG1 | External Trigger Input | Input | – | – | – |
| Serial Peripheral Interface - SPI(x=[0..1]) | | | | | |
| SPIx_MISO | Master In Slave Out | I/O | – | – | – |
| SPIx_MOSI | Master Out Slave In | I/O | – | – | – |
| SPIx_SPCK | SPI Serial Clock | I/O | – | – | – |
| SPIx_NPCS0 | SPI Peripheral Chip Select 0 | I/O | Low | – | – |
| SPIx_NPCS1–SPIx_NPCS3 | SPI Peripheral Chip Select | Output | Low | – | – |
| Quad IO SPI - QSPI | | | | | |
| QSCK | QSPI Serial Clock | Output | – | – | – |
| QCS | QSPI Chip Select | Output | – | – | – |
| QIO0–QIO3 | QSPI I/O<br>QIO0 is QMOSI Master Out Slave In<br>QIO1 is QMISO Master In Slave Out | I/O | – | – | – |
| Two-Wire Interface - TWIHS (x=0..2) | | | | | |
| TWDx | TWIx Two-wire Serial Data | I/O | – | – | – |
| TWCKx | TWIx Two-wire Serial Clock | I/O | – | – | – |

.....continued

| Signal Name | Function | Type | Active Level | Voltage Reference | Comments |
|---|---|---|---|---|---|
| Analog | | | | | |
| VREFP | ADC, DAC and Analog Comparator Positive Reference | Analog | – | – | – |
| VREFN | ADC, DAC and Analog Comparator Negative Reference Must be connected to GND or GNDANA. | Analog | – | – | – |
| 12-bit Analog Front End - (x=[0..1]) | | | | | |
| AFEx_AD0–AFEx_AD11 | Analog Inputs | Analog, Digital | – | – | – |
| AFEx_ADTRG | ADC Trigger | Input | – | VDDIO | – |
| 12-bit Digital-to-Analog Converter - DAC | | | | | |
| DAC0–DAC1 | Analog Output | Analog, Digital | – | – | – |
| DATRG | DAC Trigger | Input | – | VDDIO | – |
| Fast Flash Programming Interface - FFPI | | | | | |
| PGMEN0–PGMEN1 | Programming Enabling | Input | – | VDDIO | – |
| PGMM0–PGMM3 | Programming Mode | Input | – | VDDIO | – |
| PGMD0–PGMD15 | Programming Data | I/O | – | | – |
| PGMRDY | Programming Ready | Output | High | | – |
| PGMNVALID | Data Direction | Output | Low | | – |
| PGMNOE | Programming Read | Input | Low | | – |
| PGMNCMD | Programming Command | Input | Low | | – |
| USB High Speed - USBHS | | | | | |
| HSDM | USB High Speed Data - | Analog, Digital | – | VDDUTMII | – |
| HSDP | USB High Speed Data + | | – | | – |
| VBG | Bias Voltage Reference for USB | Analog | – | – | – |
| Ethernet MAC 10/100 - GMAC | | | | | |
| GREFCK | Reference Clock | Input | – | – | RMII only |
| GTXCK | Transmit Clock | Input | – | – | MII only |
| GRXCK | Receive Clock | Input | – | – | MII only |
| GTXEN | Transmit Enable | Output | – | – | – |
| GTX0 - GTX3 | Transmit Data | Output | – | – | GTX0–GTX1 only in RMII |
| GTXER | Transmit Coding Error | Output | – | – | MII only |

.....continued

| Signal Name | Function | Type | Active Level | Voltage Reference | Comments |
|---|---|---|---|---|---|
| GRXDV | Receive Data Valid | Input | – | – | MII only |
| GRX0 - GRX3 | Receive Data | Input | – | – | GRX0–GRX1 only in RMII |
| GRXER | Receive Error | Input | – | – | – |
| GCRS | Carrier Sense | Input | – | – | MII only |
| GCOL | Collision Detected | Input | – | – | MII only |
| GMDC | Management Data Clock | Output | – | – | – |
| GMDIO | Management Data Input/ Output | I/O | – | – | – |
| GTSUCOMP | TSU timer comparison valid | Output | – | – | – |
| Controller Area Network - MCAN (x=[0:1]) | | | | | |
| CANRXx | CAN Receive | Input | – | – | CANRX1 is available on PD28 for 100-pin only CANRX1 is available on PC12 for 144-pin only |
| CANTXx | CAN Transmit | Output | – | – | PCK5 can be used for CAN clock PCK6 and PCK7 can be used for CAN timestamping |
| MediaLB - MLB | | | | | |
| MLBCLK | MLB Clock | input | – | – | – |
| MLBSIG | MLB Signal | I/O | – | – | – |
| MLBDAT | MLB Data | I/O | – | – | – |

**Note:** Refer to the "Active Mode" section in the Power Considerations chapter for restrictions on the voltage range of analog cells.

# 5. Space Quality Grade

The hermetic SAMV71Q21RT is manufactured in compliance with MIL class Q or class V requirements (screening testing, qualification testing, and TCI/SCI-specifications) as well as in Hirel Ceramic. The plastic SAMV71Q21RT is qualified in compliance with the AEC-Q100 automotive requirements, with specific additional tests necessary for space applications. Screening and qualification flows are described in Aerospace & Defense AEQA0242 specification, available on Microchip web site.

© 2021 Microchip Technology Inc.
and its subsidiaries

# 6. Package and Pinout

In the tables that follow, the column “Reset State” indicates the reset state of the line with mnemonics.

- “PIO” “/” signal

Indicates whether the PIO Line resets in I/O mode or in peripheral mode. If “PIO” is mentioned, the PIO line is maintained in a static state as soon as the reset is released. As a result, the bit corresponding to the PIO line in the register PIO_PSR (Peripheral Status Register) resets low.

If a signal name is mentioned in the “Reset State” column, the PIO line is assigned to this function and the corresponding bit in PIO_PSR resets high. This is the case of pins controlling memories, in particular the address lines, which require the pin to be driven as soon as the reset is released.

- “I” / ”O”

Indicates whether the signal is input or output state.

- “PU” / “PD”

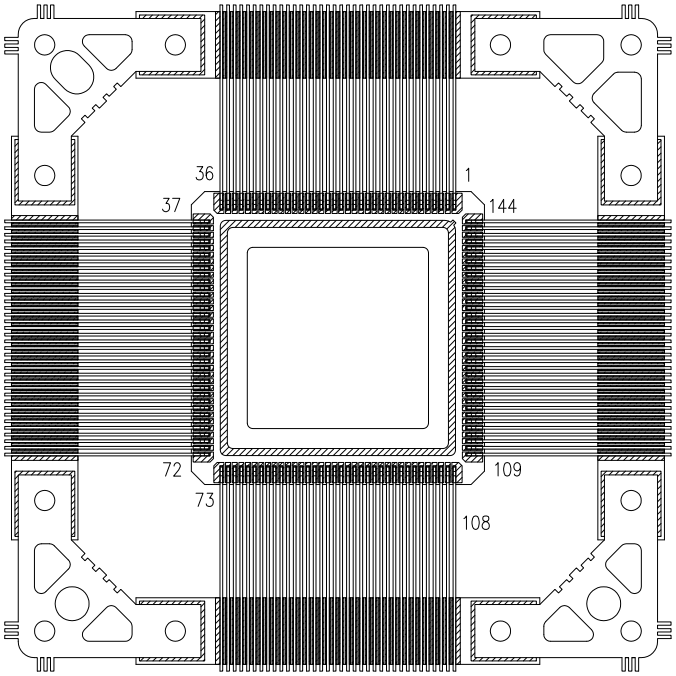
Indicates whether pullup, pulldown or nothing is enabled.

- “ST”

Indicates if Schmitt Trigger is enabled.

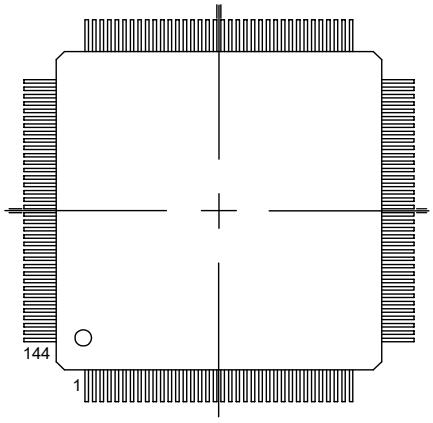
## 6.1 CQFP Package Outline

**Figure 6-1. Orientation of the 144-pin CQFP Package**



## 6.2 LQFP Package Outline

**Figure 6-2. Orientation of the 144-pin LQFP Package**




## 6.3 Pinout

**Table 6-1. CQFP/LQFP Package Pinout**

| | | | Primary | | Alternate | | PIO Peripheral A | | PIO Peripheral B | | PIO Peripheral C | | PIO Peripheral D | | Reset State |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Pin No. | Power Rail | I/O Type | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal, Dir, PU, PD, HiZ, ST |
| 102 | VDDIO | GPIO_AD | PA0 | I/O | WKUP0[1] | I | PWMC0_ PWMH0 | O | TIOA0 | I/O | A17/BA1 | O | I2SC0_MCK | O | PIO, I, PU, ST |
| 99 | VDDIO | GPIO_AD | PA1 | I/O | WKUP1[1] | I | PWMC0_ PWML0 | O | TIOB0 | I/O | A18 | O | I2SC0_CK | I/O | PIO, I, PU, ST |
| 93 | VDDIO | GPIO | PA2 | I/O | WKUP2[1] | I | PWMC0_ PWMH1 | O | – | – | DATRG | I | – | – | PIO, I, PU, ST |
| 91 | VDDIO | GPIO_AD | PA3 | I/O | PIODC0[2] | I | TWD0 | I/O | LONCOL1 | I | PCK2 | O | – | – | PIO, I, PU, ST |
| 77 | VDDIO | GPIO | PA4 | I/O | WKUP3/P IODC1[3] | I | TWCK0 | O | TCLK0 | I | UTXD1 | O | – | – | PIO, I, PU, ST |
| 73 | VDDIO | GPIO_AD | PA5 | I/O | WKUP4/P IODC2[3] | I | PWMC1_ PWML3 | O | ISI_D4 | I | URXD1 | I | – | – | PIO, I, PU, ST |
| 114 | VDDIO | GPIO_AD | PA6 | I/O | – | – | – | – | PCK0 | O | UTXD1 | O | – | – | PIO, I, PU, ST |
| 35 | VDDIO | CLOCK | PA7 | I/O | XIN32[4] | I | – | – | PWMC0_ PWMH3 | O | – | – | – | – | PIO, HiZ |
| 36 | VDDIO | CLOCK | PA8 | I/O | XIN32[4] | O | PWMC1_ PWMH3 | O | AFE0_ADTRG | I | – | – | – | – | PIO, HiZ |
| 75 | VDDIO | GPIO_AD | PA9 | I/O | WKUP6/P IODC3[3] | I | URXD0 | I | ISI_D3 | I | PWMC0_ PWMFI0 | I | – | – | PIO, I, PU, ST |
| 66 | VDDIO | GPIO_AD | PA10 | I/O | PIODC4[2] | I | UTXD0 | O | PWMC0_ PWMEXTRG0 | I | RD | I | – | – | PIO, I, PU, ST |
| 64 | VDDIO | GPIO_AD | PA11 | I/O | WKUP7/P IODC5[3] | I | QCS | O | PWMC0_ PWMH0 | O | PWMC1_ PWML0 | O | – | – | PIO, I, PU, ST |
| 68 | VDDIO | GPIO_AD | PA12 | I/O | PIODC6[2] | I | QIO1 | I/O | PWMC0_ PWMH1 | O | PWMC1_ PWMH0 | O | – | – | PIO, I, PU, ST |
| 42 | VDDIO | GPIO_AD | PA13 | I/O | PIODC7[2] | I | QIO0 | I/O | PWMC0_ PWMH2 | O | PWMC1_ PWML1 | O | – | – | PIO, I, PU, ST |
| 51 | VDDIO | GPIO_CLK | PA14 | I/O | WKUP8/P IODCEN1[3] | I | QSCK | O | PWMC0_ PWMH3 | O | PWMC1_ PWMH1 | O | – | – | PIO, I, PU, ST |
| 49 | VDDIO | GPIO_AD | PA15 | I/O | – | – | D14 | I/O | TIOA1 | I/O | PWMC0_ PWML3 | O | I2SC0_WS | I/O | PIO, I, PU, ST |
| 45 | VDDIO | GPIO_AD | PA16 | I/O | – | – | D15 | I/O | TIOB1 | I/O | PWMC0_ PWML2 | O | I2SC0_DI | I | PIO, I, PU, ST |
| 25 | VDDIO | GPIO_AD | PA17 | I/O | AFE0_AD 6[5] | I | QIO2 | I/O | PCK1 | O | PWMC0_ PWMH3 | O | – | – | PIO, I, PU, ST |
| 24 | VDDIO | GPIO_AD | PA18 | I/O | AFE0_AD 7[5] | I | PWMC1_ PWMEXTRG1 | I | PCK2 | O | A14 | O | – | – | PIO, I, PU, ST |
| 23 | VDDIO | GPIO_AD | PA19 | I/O | AFE0_AD 8/WKUP9[6] | I | – | – | PWMC0_ PWML0 | O | A15 | O | I2SC1_MCK | O | PIO, I, PU, ST |
| 22 | VDDIO | GPIO_AD | PA20 | I/O | AFE0_AD 9/WKUP10[6] | I | – | – | PWMC0_ PWML1 | O | A16/BA0 | O | I2SC1_CK | I/O | PIO, I, PU, ST |
| 32 | VDDIO | GPIO_AD | PA21 | I/O | AFE0_AD 1/ PIODCEN 2[8] | I | RXD1 | I | PCK1 | O | PWMC1_ PWMFI0 | I | – | – | PIO, I, PU, ST |
| 37 | VDDIO | GPIO_AD | PA22 | I/O | PIODCCL K[2] | I | RK | I/O | PWMC0_ PWMEXTRG1 | I | NCS2 | O | – | – | PIO, I, PU, ST |
| 46 | VDDIO | GPIO_AD | PA23 | I/O | – | – | SCK1 | I/O | PWMC0_ PWMH0 | O | A19 | O | PWMC1_ PWML2 | O | PIO, I, PU, ST |
| 56 | VDDIO | GPIO_AD | PA24 | I/O | – | – | RTS1 | O | PWMC0_ PWMH1 | O | A20 | O | ISI_PCK | I | PIO, I, PU, ST |
| 59 | VDDIO | GPIO_AD | PA25 | I/O | – | – | CTS1 | I | PWMC0_ PWMH2 | O | A23 | O | MCCK | O | PIO, I, PU, ST |
| 62 | VDDIO | GPIO | PA26 | I/O | – | – | DCD1 | I | TIOA2 | O | MCDA2 | I/O | PWMC1_ PWMFI1 | I | PIO, I, PU, ST |
| 70 | VDDIO | GPIO_AD | PA27 | I/O | – | – | DTR1 | O | TIOB2 | I/O | MCDA3 | I/O | ISI_D7 | I | PIO, I, PU, ST |
| 112 | VDDIO | GPIO | PA28 | I/O | – | – | DSR1 | I | TCLK1 | I | MCCDA | I/O | PWMC1_ PWMFI2 | I | PIO, I, PU, ST |
| 129 | VDDIO | GPIO | PA29 | I/O | – | – | RI1 | I | TCLK2 | I | – | – | – | – | PIO, I, PU, ST |
| 116 | VDDIO | GPIO | PA30 | I/O | WKUP11[1] | I | PWMC0_ PWML2 | O | PWMC1_ PWMEXTRG0 | I | MCDA0 | I/O | I2SC0_DO | O | PIO, I, PU, ST |
| 118 | VDDIO | GPIO_AD | PA31 | I/O | – | – | SPI0_NPCS1 | I/O | PCK2 | O | MCDA1 | I/O | PWMC1_ PWMH2 | O | PIO, I, PU, ST |
| 21 | VDDIO | GPIO | PB0 | I/O | AFE0_AD10/ RTCOUT0[7] | I | PWMC0_ PWMH0 | O | – | – | RXD0 | I | TF | I/O | PIO, I, PU, ST |
| 20 | VDDIO | GPIO | PB1 | I/O | AFE1_AD 0/ RTCOUT 1[7] | I | PWMC0_ PWMH1 | O | GTSUCOMP | O | TXD0 | I/O | TK | I/O | PIO, I, PU, ST |
| 26 | VDDIO | GPIO | PB2 | I/O | AFE0_AD5[5] | I | CANTX0 | O | – | – | CTS0 | I | SPI0_NPCS0 | I/O | PIO, I, PU, ST |
| 31 | VDDIO | GPIO_AD | PB3 | I/O | AFE0_AD2/WKUP12[6] | I | CANRX0 | I | PCK2 | O | RTS0 | O | ISI_D2 | I | PIO, I, PU, ST |
| 105 | VDDIO | GPIO_MLB | PB4 | I/O | TDI[9] | I | TWD1 | I/O | PWMC0_ PWMH2 | O | MLBCLK | I | TXD1 | I/O | PIO, I, PD, ST |
| 109 | VDDIO | GPIO_MLB | PB5 | I/O | TDO/TRACESWO/ WKUP13[9] | O | TWCK1 | O | PWMC0_ PWML0 | O | MLBDAT | I/O | TD | O | O, PU |
| 79 | VDDIO | GPIO | PB6 | I/O | SWDIO/TMS[9] | I | – | – | – | – | – | – | – | – | PIO, I, ST |

.....continued

| Pin No. | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral A | | PIO Peripheral B | | PIO Peripheral C | | PIO Peripheral D | | Reset State |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal, Dir, PU, PD, HiZ, ST |
| 89 | VDDIO | GPIO | PB7 | I/O | SWCLK/TCK[9] | I | – | – | – | – | – | – | – | – | PIO, I, ST |
| 141 | VDDIO | CLOCK | PB8 | I/O | XOUT[10] | O | – | – | – | – | – | – | – | – | PIO, HiZ |
| 142 | VDDIO | CLOCK | PB9 | I/O | XIN[10] | I | – | – | – | – | – | – | – | – | PIO, HiZ |
| 87 | VDDIO | GPIO | PB12 | I/O | ERASE[9] | I | PWMC0_ PWML1 | O | GTSUCOMP | O | – | – | PCK0 | O | PIO, I, PD, ST |
| 144 | VDDIO | GPIO_AD | PB13 | I/O | DAC0[11] | O | PWMC0_ PWML2 | O | PCK0 | O | SCK0 | I/O | – | – | PIO, I, PU, ST |
| 11 | VDDIO | GPIO_AD | PC0 | I/O | AFE1_AD9[5] | I | D0 | I/O | PWMC0_ PWML0 | O | – | – | – | – | PIO, I, PU, ST |
| 38 | VDDIO | GPIO_AD | PC1 | I/O | – | – | D1 | I/O | PWMC0_ PWML1 | O | – | – | – | – | PIO, I, PU, ST |
| 39 | VDDIO | GPIO_AD | PC2 | I/O | – | – | D2 | I/O | PWMC0_ PWML2 | O | – | – | – | – | PIO, I, PU, ST |
| 40 | VDDIO | GPIO_AD | PC3 | I/O | – | – | D3 | I/O | PWMC0_ PWML3 | O | – | – | – | – | PIO, I, PU, ST |
| 41 | VDDIO | GPIO_AD | PC4 | I/O | – | – | D4 | I/O | – | – | – | – | – | – | PIO, I, PU, ST |
| 58 | VDDIO | GPIO_AD | PC5 | I/O | – | – | D5 | I/O | TIOA6 | I/O | – | – | – | – | PIO, I, PU, ST |
| 54 | VDDIO | GPIO_AD | PC6 | I/O | – | – | D6 | I/O | TIOB6 | I/O | – | – | – | – | PIO, I, PU, ST |
| 48 | VDDIO | GPIO_AD | PC7 | I/O | – | – | D7 | I/O | TCLK6 | I | – | – | – | – | PIO, I, PU, ST |
| 82 | VDDIO | GPIO_AD | PC8 | I/O | – | – | NWR0/NWE | O | TIOA7 | I/O | – | – | – | – | PIO, I, PU, ST |
| 86 | VDDIO | GPIO_AD | PC9 | I/O | – | – | NANDOE | O | TIOB7 | I/O | – | – | – | – | PIO, I, PU, ST |
| 90 | VDDIO | GPIO_AD | PC10 | I/O | – | – | NANDWE | O | TCLK7 | I | – | – | – | – | PIO, I, PU, ST |
| 94 | VDDIO | GPIO_AD | PC11 | I/O | – | – | NRD | O | TIOA8 | I/O | – | – | – | – | PIO, I, PU, ST |
| 17 | VDDIO | GPIO_AD | PC12 | I/O | AFE1_AD3[5] | I | NCS3 | O | TIOB8 | I/O | CANRX1 | I | – | – | PIO, I, PU, ST |
| 19 | VDDIO | GPIO_AD | PC13 | I/O | AFE1_AD1[5] | I | NWAIT | I | PWMC0_ PWMH3 | O | SDA10 | O | – | – | PIO, I, PU, ST |
| 97 | VDDIO | GPIO_AD | PC14 | I/O | – | – | NCS0 | O | TCLK8 | I | CANTX1 | O | – | – | PIO, I, PU, ST |
| 18 | VDDIO | GPIO_AD | PC15 | I/O | AFE1_AD2[5] | I | NCS1/SDCS | O | PWMC0_ PWML3 | O | – | – | – | – | PIO, I, PU, ST |
| 100 | VDDIO | GPIO_AD | PC16 | I/O | – | – | A21/NANDALE | O | – | – | – | – | – | – | PIO, I, PU, ST |
| 103 | VDDIO | GPIO_AD | PC17 | I/O | – | – | A22/NANDCLE | O | – | – | – | – | – | – | PIO, I, PU, ST |
| 111 | VDDIO | GPIO_AD | PC18 | I/O | – | – | A0/NBS0 | O | PWMC0_ PWML1 | O | – | – | – | – | PIO, I, PU, ST |
| 117 | VDDIO | GPIO_AD | PC19 | I/O | – | – | A1 | O | PWMC0_ PWMH2 | O | – | – | – | – | PIO, I, PU, ST |
| 120 | VDDIO | GPIO_AD | PC20 | I/O | – | – | A2 | O | PWMC0_ PWML2 | O | – | – | – | – | PIO, I, PU, ST |
| 122 | VDDIO | GPIO_AD | PC21 | I/O | – | – | A3 | O | PWMC0_ PWMH3 | O | – | – | – | – | PIO, I, PU, ST |
| 124 | VDDIO | GPIO_AD | PC22 | I/O | – | – | A4 | O | PWMC0_ PWML3 | O | – | – | – | – | PIO, I, PU, ST |
| 127 | VDDIO | GPIO_AD | PC23 | I/O | – | – | A5 | O | TIOA3 | I/O | – | – | – | – | PIO, I, PU, ST |
| 130 | VDDIO | GPIO_AD | PC24 | I/O | – | – | A6 | O | TIOB3 | I/O | SPI1_SPCK | O | – | – | PIO, I, PU, ST |
| 133 | VDDIO | GPIO_AD | PC25 | I/O | – | – | A7 | O | TCLK3 | I | SPI1_NPCS0 | I/O | – | – | PIO, I, PU, ST |
| 13 | VDDIO | GPIO_AD | PC26 | I/O | AFE1_AD 7[5] | I | A8 | O | TIOA4 | I/O | SPI1_MISO | I | – | – | PIO, I, PU, ST |
| 12 | VDDIO | GPIO_AD | PC27 | I/O | AFE1_AD8[5] | I | A9 | O | TIOB4 | I/O | SPI1_MOSI | O | – | – | PIO, I, PU, ST |
| 76 | VDDIO | GPIO_AD | PC28 | I/O | – | – | A10 | O | TCLK4 | I | SPI1_NPCS1 | I/O | – | – | PIO, I, PU, ST |
| 16 | VDDIO | GPIO_AD | PC29 | I/O | AFE1_AD4[5] | I | A11 | O | TIOA5 | I/O | SPI1_NPCS2 | O | – | – | PIO, I, PU, ST |
| 15 | VDDIO | GPIO_AD | PC30 | I/O | AFE1_AD5[5] | I | A12 | O | TIOB5 | I/O | SPI1_NPCS3 | O | – | – | PIO, I, PU, ST |
| 14 | VDDIO | GPIO_AD | PC31 | I/O | AFE1_AD6[5] | I | A13 | O | TCLK5 | I | – | – | – | – | PIO, I, PU, ST |
| 1 | VDDIO | GPIO_AD | PD0 | I/O | DAC1[11] | I | GTXCK | I | PWMC1_ PWML0 | O | SPI1_NPCS1 | I/O | DCD0 | I | PIO, I, PU, ST |
| 132 | VDDIO | GPIO | PD1 | I/O | – | – | GTXEN | O | PWMC1_ PWMH0 | O | SPI1_NPCS2 | I/O | DTR0 | O | PIO, I, PU, ST |
| 131 | VDDIO | GPIO | PD2 | I/O | – | – | GTX0 | O | PWMC1_ PWML1 | O | SPI1_NPCS3 | I/O | DSR0 | I | PIO, I, PU, ST |
| 128 | VDDIO | GPIO | PD3 | I/O | – | – | GTX1 | O | PWMC1_ PWMH1 | O | UTXD4 | O | RI0 | I | PIO, I, PU, ST |
| 126 | VDDIO | GPIO_CLK | PD4 | I/O | – | – | GRXDV | I | PWMC1_ PWML2 | O | TRACED0 | O | DCD2 | I | PIO, I, PU, ST |
| 125 | VDDIO | GPIO_CLK | PD5 | I/O | – | – | GRX0 | I | PWMC1_ PWMH2 | O | TRACED1 | O | DTR2 | O | PIO, I, PU, ST |
| 121 | VDDIO | GPIO_CLK | PD6 | I/O | – | – | GRX1 | I | PWMC1_ PWML3 | O | TRACED2 | O | DSR2 | I | PIO, I, PU, ST |
| 119 | VDDIO | GPIO_CLK | PD7 | I/O | – | – | GRXER | I | PWMC1_ PWMH3 | O | TRACED3 | O | RI2 | I | PIO, I, PU, ST |
| 113 | VDDIO | GPIO_CLK | PD8 | I/O | – | – | GMDC | O | PWMC0_ PWMFI1 | I | – | – | TRACECLK | O | PIO, I, PU, ST |
| 110 | VDDIO | GPIO_CLK | PD9 | I/O | – | – | GMDIO | I/O | PWMC0_ PWMFI2 | I | AFE1_ADTRG | I | – | – | PIO, I, PU, ST |
| 101 | VDDIO | GPIO_MLB | PD10 | I/O | – | – | GCRS | I | PWMC0_ PWML0 | O | TD | O | MLBSIG | I/O | PIO, I, PD, ST |
| 98 | VDDIO | GPIO_AD | PD11 | I/O | – | – | GRX2 | I | PWMC0_ PWMH0 | O | GTSUCOMP | O | ISI_D5 | I | PIO, I, PU, ST |
| 92 | VDDIO | GPIO_AD | PD12 | I/O | – | – | GRX3 | I | CANTX1 | O | SPI0_NPCS2 | O | ISI_D6 | I | PIO, I, PU, ST |
| 88 | VDDIO | GPIO_CLK | PD13 | I/O | – | – | GCOL | I | – | – | SDA10 | O | – | – | PIO, I, PU, ST |
| 84 | VDDIO | GPIO_AD | PD14 | I/O | – | – | GRXCK | I | – | – | SDCKE | O | – | – | PIO, I, PU, ST |
| 106 | VDDIO | GPIO_AD | PD15 | I/O | – | – | GTX2 | O | RXD2 | I | NWR1/NBS1 | O | – | – | PIO, I, PU, ST |
| 78 | VDDIO | GPIO_AD | PD16 | I/O | – | – | GTX3 | O | TXD2 | I/O | RAS | O | – | – | PIO, I, PU, ST |
| 74 | VDDIO | GPIO_AD | PD17 | I/O | – | – | GTXER | O | SCK2 | I/O | CAS | O | – | – | PIO, I, PU, ST |
| 69 | VDDIO | GPIO_AD | PD18 | I/O | – | – | NCS1/SDCS | O | RTS2 | O | URXD4 | I | – | – | PIO, I, PU, ST |
| 67 | VDDIO | GPIO_AD | PD19 | I/O | – | – | NCS3 | O | CTS2 | I | UTXD4 | O | – | – | PIO, I, PU, ST |
| 65 | VDDIO | GPIO | PD20 | I/O | – | – | PWMC0_ PWMH0 | O | SPI0_MIS O | I/O | GTSUCOMP | O | – | – | PIO, I, PU, ST |
| 63 | VDDIO | GPIO_AD | PD21 | I/O | – | – | PWMC0_ PWMH1 | O | SPI0_MOSI | I/O | TIOA11 | I/O | ISI_D1 | I | PIO, I, PU, ST |
| 60 | VDDIO | GPIO_AD | PD22 | I/O | – | – | PWMC0_ PWMH2 | O | SPI0_SPCK | O | TIOB11 | I/O | ISI_D0 | I | PIO, I, PU, ST |
| 57 | VDDIO | GPIO_CLK | PD23 | I/O | – | – | PWMC0_ PWMH3 | O | – | – | SDCK | O | – | – | PIO, I, PU, ST |
| 55 | VDDIO | GPIO_AD | PD24 | I/O | – | – | PWMC0_ PWML0 | O | RF | I/O | TCLK11 | I | ISI_HSYNC | I | PIO, I, PU, ST |
| 52 | VDDIO | GPIO_AD | PD25 | I/O | – | – | PWMC0_ PWML1 | O | SPI0_NPCS1 | I/O | URXD2 | I | ISI_VSYNC | I | PIO, I, PU, ST |
| 53 | VDDIO | GPIO | PD26 | I/O | – | – | PWMC0_ PWML2 | O | TD | O | UTXD2 | O | UTXD1 | O | PIO, I, PU, ST |
| 47 | VDDIO | GPIO_AD | PD27 | I/O | – | – | PWMC0_ PWML3 | O | SPI0_NPCS3 | O | TWD2 | O | ISI_D8 | I | PIO, I, PU, ST |
| 71 | VDDIO | GPIO_AD | PD28 | I/O | WKUP5[1] | I | URXD3 | I | – | – | TWCK2 | O | ISI_D9 | I | PIO, I, PU, ST |
| 108 | VDDIO | GPIO_AD | PD29 | I/O | – | – | – | – | – | – | SDWE | O | – | – | PIO, I, PU, ST |
| 34 | VDDIO | GPIO_AD | PD30 | I/O | AFE0_AD0[5] | I | UTXD3 | 0 | – | – | – | – | ISI_D10 | I | PIO, I, PU, ST |

.....continued

| Pin No. | Power Rail | I/O Type | Primary | | Alternate | | PIO Peripheral A | | PIO Peripheral B | | PIO Peripheral C | | PIO Peripheral D | | Reset State |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal | Dir | Signal, Dir, PU, PD, HiZ, ST |
| 2 | VDDIO | GPIO_AD | PD31 | I/O | – | – | QIO3 | I/O | UTXD3 | O | PCK2 | O | ISI_D11 | I | PIO, I, PU, ST |
| 4 | VDDIO | GPIO_AD | PE0 | I/O | AFE1_AD 11[5] | I | D8 | I/O | TIOA9 | I/O | I2SC1_WS | I/O | – | – | PIO, I, PU, ST |
| 6 | VDDIO | GPIO_AD | PE1 | I/O | – | – | D9 | I/O | TIOB9 | I/O | I2SC1_DO | O | – | – | PIO, I, PU, ST |
| 7 | VDDIO | GPIO_AD | PE2 | I/O | – | – | D10 | I/O | TCLK9 | I | I2SC1_DI | I | – | – | PIO, I, PU, ST |
| 10 | VDDIO | GPIO_AD | PE3 | I/O | AFE1_AD10[5] | I | D11 | I/O | TIOA10 | I/O | – | – | – | – | PIO, I, PU, ST |
| 27 | VDDIO | GPIO_AD | PE4 | I/O | AFE0_AD4[5] | I | D12 | I/O | TIOB10 | I/O | – | – | – | – | PIO, I, PU, ST |
| 28 | VDDIO | GPIO_AD | PE5 | I/O | AFE0_AD3[5] | I | D13 | I/O | TCLK10 | I/O | – | – | – | – | PIO, I, PU, ST |
| 3 | VDDOUT | Power | VDDOUT | – | – | – | – | – | – | – | – | – | – | – | – |
| 5 | VDDIN | Power | VDDIN | – | – | – | – | – | – | – | – | – | – | – | – |
| 8 | GND | Reference | VREFN | I | – | – | – | – | – | – | – | – | – | – | – |
| 9 | VDDIO | Reference | VREFP | I | – | – | – | – | – | – | – | – | – | – | – |
| 83 | VDDIO | RST | NRST | I/O | – | – | – | – | – | – | – | – | – | – | I, PU |
| 85 | VDDIO | TEST | TST | I | – | – | – | – | – | – | – | – | – | – | I, PD |
| 30,43, 72, 80, 96 | VDDIO | Power | VDDIO | – | – | – | – | – | – | – | – | – | – | – | – |
| 104 | VDDIO | TEST | JTAGSEL | I | – | – | – | – | – | – | – | – | – | – | I, PD |
| 29,33, 50, 81, 107 | VDDCORE | Power | VDDCORE | – | – | – | – | – | – | – | – | – | – | – | – |
| 123 | VDDPLL | Power | VDDPLL | – | – | – | – | – | – | – | – | – | – | – | – |
| 134 | VDDUTMII | Power | VDDUTMII | – | – | – | – | – | – | – | – | – | – | – | – |
| 136 | VDDUTMII | USBHS | HSDM | I/O | – | – | – | – | – | – | – | – | – | – | – |
| 137 | VDDUTMII | USBHS | HSDP | I/O | – | – | – | – | – | – | – | – | – | – | – |
| 44,61, 95, 115, 135,1 38 | GND | Ground | GND | – | – | – | – | – | – | – | – | – | – | – | – |
| – | GNDANA | Ground | GNDANA | – | – | – | – | – | – | – | – | – | – | – | – |
| – | GNDUTMI | Ground | GNDUTMI | – | – | – | – | – | – | – | – | – | – | – | – |
| – | GNDPLLUSB | Ground | GNDPLLUSB | – | – | – | – | – | – | – | – | – | – | – | – |
| – | GNDPLL | Ground | GNDPLL | – | – | – | – | – | – | – | – | – | – | – | – |
| 139 | VDDUTMIC | Power | VDDUTMIC | – | – | – | – | – | – | – | – | – | – | – | – |
| 140 | – | VBG | VBG | I | – | – | – | – | – | – | – | – | – | – | – |
| 143 | VDDPLLUSB | Power | VDDPLLUSB | – | – | – | – | – | – | – | – | – | – | – | – |

**Notes:**

1. WKUPx can be used if the PIO Controller defines the I/O line as “input”.
2. To select this extra function, refer to the 32.5.14 Parallel Capture Mode section in the Parallel Input/Output Controller (PIO) chapter.
3. PIODCEN1/PIODCx has priority over WKUPx. Refer to the 32.5.14 Parallel Capture Mode section in the PIO chapter.
4. Refer to the 23.4.2 Slow Clock Generator section in the Supply Controller (SUPC) chapter.
5. To select this extra function, refer to the 33.5.2.1 I/O Lines section in the External Bus Interface (EBI) chapter. This selection is independent of the PIO line configuration. PIO lines must be configured according to required settings (PU or PD).
6. Analog input has priority over WKUPx pin. To select the analog input, refer to the 33.5.2.1 I/O Lines section in the EBI chapter. WKUPx can be used if the PIO controller defines the I/O line as “input”.
7. Analog input has priority over RTCOUTx pin. To select the analog input, refer to the 33.5.2.1 I/O Lines section in the EBI chapter. Refer to the Waveform Generation section in the Real-Time Clock (RTC) chapter to select RTCOUTx.
8. Analog input has priority over WKUPx pin. To select the analog input, refer to the 33.5.2.1 I/O Lines section in the EBI chapter. To select PIODCEN2, refer to the 32.5.14 Parallel Capture Mode in the PIO chapter.
9. Refer to the System I/O Configuration Register (19.4.7 CCFG_SYSIO) in the Bus Matrix (MATRIX) chapter.
10. Refer to the 30.5.3 Main Crystal Oscillator section in the Clock Generator chapter. This selection is independent of the PIO line configuration. PIO lines must be configured according to XINxx (I) and XOUTxx (O).
11. DAC0 is selected when DACC_CHER.CH0 is set. DAC1 is selected when DACC_CHER.CH1 is set. Refer to the DACC Channel Enable Register in the Digital-to-Analog Converter Controller (DACC) chapter.

# 7. Power Considerations

## 7.1 Power Supplies

The following table defines the power supply rails of the SAMV71Q21RT and the estimated power consumption at typical voltage.

**Table 7-1. Power Supplies**

| Name | Associated Ground | Powers |
|---|---|---|
| VDDCORE | GND | Core, embedded memories and peripherals. |
| VDDIO | GND | Peripheral I/O lines (Input/Output Buffers), backup part, 1 Kbytes of backup SRAM, 32 kHz crystal oscillator, oscillator pads. For USB operations, VDDIO voltage range must be between 3.0V and 3.6V. |
| VDDIN | GND, GNDANA | Voltage regulator input. Also supplies the ADC, DAC and analog voltage comparator. |
| VDDPLL | GND, GNDPLL | PLLA and the fast RC oscillator. |
| VDDPLLUSB | GND, GNDPLLUSB | UTMI PLL and the 3 to 20 MHz oscillator. |
| VDDUTMII | GNDUTMI | USB transceiver interface. Must be connected to VDDIO. |
| VDDUTMIC | GNDUTMI | USB transceiver core. |

## 7.2 Power Constraints

The following power constraints are applied to SAMV71Q21RT devices. Deviating from these constraints may lead to unpredictable results.

- VDDIN and VDDIO must have the same level
- VDDIN and VDDIO must always be higher than or equal to VDDCORE
- VDDCORE, VDDPLL and VDDUTMIC voltage levels must not vary by more than 0.6V
- For the USB to be operational, VDDUTMII, VDDPLLUSB, VDDIN and VDDIO must be higher than or equal to 3.0V

### 7.2.1 Powerup

VDDIO and VDDIN must rise simultaneously, prior to VDDCORE, VDDPLL and VDDUTMIC rising. This is respected if VDDCORE, VDDPLL and VDDUTMIC are supplied by the embedded voltage regulator.

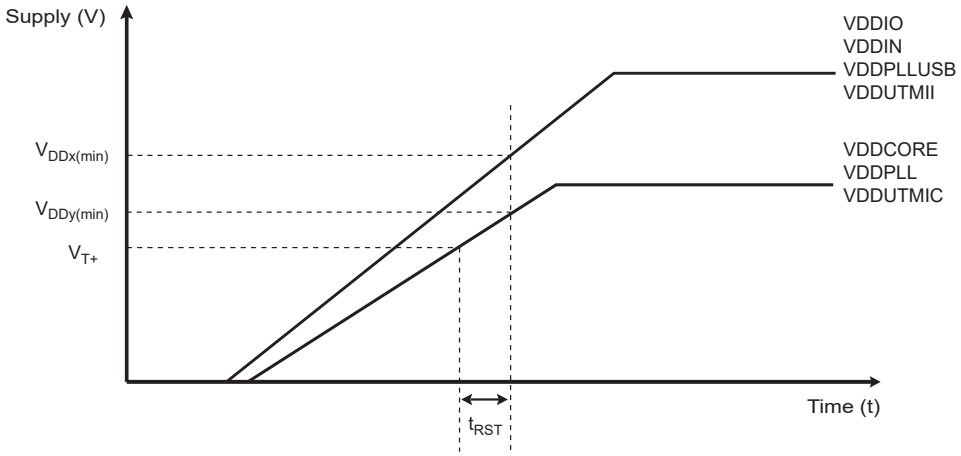
If VDDCORE is powered by an external voltage regulator, VDDIO and VDDIN must reach their minimum operating voltage before VDDCORE has reached $VDDCORE_{min}$. The minimum slope for VDDCORE is defined by:

$$(VDDCORE_{min} - V_{T+min})/(t_{RESmin})$$

If VDDCORE rises at the same time as VDDIO and VDDIN, the minimum and maximum rising slopes of VDDIO and VDDIN must be respected. Refer to the section “DC Characteristics”.

In order to prevent any overcurrent at powerup, it is required that VREFP rises simultaneously with VDDIO and VDDIN.

**Figure 7-1. Powerup Sequence**




**Related Links**

23.4.6 Backup Power Supply Reset

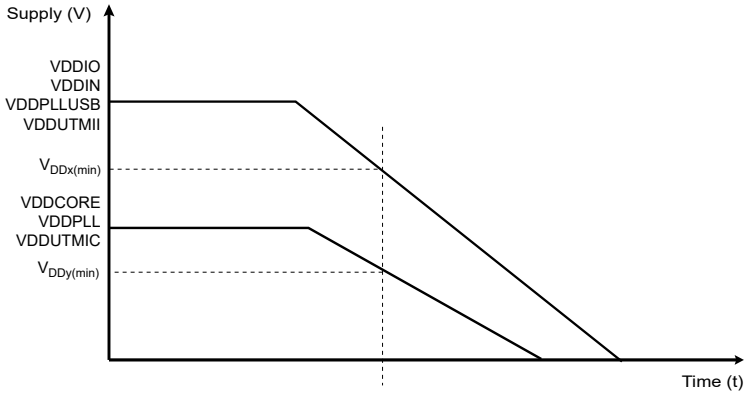
23.4.6.1 Raising the Backup Power Supply

### 7.2.2 Powerdown

If VDDCORE, VDDPLL and VDDUTMIC are not supplied by the embedded voltage regulator, VDDIO, VDDIN, VDDPLLUSB and VDDUTMII should fall simultaneously, prior to VDDCORE, VDDPLL and VDDUTMIC falling. The VDDCORE falling slope must not be faster than 20V/ms.

In order to prevent any overcurrent at powerdown, it is required that VREFP falls simultaneously with VDDIO and VDDIN.

**Figure 7-2. Powerdown Sequence**




## 7.3 Voltage Regulator

The SAMV71Q21RT embeds a voltage regulator that is managed by the Supply Controller.

For adequate input and output power supply decoupling/bypassing, refer to the DC Characteristics in the Electrical Characteristics chapter.

## 7.4 Backup SRAM Power Switch

The SAMV71Q21RT embeds a power switch to supply the 1 Kbyte of backup SRAM. It is activated only when VDDCORE is switched off to ensure retention of the contents of the backup SRAM. When VDDCORE is switched on, the backup SRAM is powered with VDDCORE.

To save the power consumption of the backup SRAM, the user can disable the backup SRAM power switch by clearing the bit SRAMON in the Supply Controller Mode Register (SUPC_MR). By default, after VDDIO rises, the backup SRAM power switch is enabled.

## 7.5 Active Mode

Active mode is the normal running mode with the core clock running from the fast RC oscillator, the main crystal oscillator or the PLLA. The Power Management Controller can be used to adapt the core, bus and peripheral frequencies and to enable and/or disable the peripheral clocks.

## 7.6 Low-power Modes

The SAMV71Q21RT features the following three Low-Power modes:

- Backup mode
- Wait mode
- Sleep mode

### 7.6.1 Backup Mode

The purpose of Backup mode is to achieve the lowest power consumption possible in a system which is performing periodic wake-ups to perform tasks but not requiring fast startup time.

The Supply Controller, zero-power Power-On Reset (POR), RTT, RTC, backup SRAM, backup registers and 32 kHz oscillator (RC or crystal oscillator selected by software in the Supply Controller) are running. The regulator and the core supply are off.

Backup mode is based on the Cortex-M7 Deep-Sleep mode with the voltage regulator disabled.

Wake-up from Backup mode is done through WKUP0–13 pins, the supply monitor (SM), the RTT, or an RTC wake-up event.

Backup mode is entered by using the VROFF bit in the Supply Controller Control Register (SUPC_CR) and the SLEEPDEEP bit in the Cortex-M7 System Control Register set to 1. Refer to information on Power Management in the" Arm Cortex-M7 documentation", which is available for download at www.arm.com.

To enter Backup mode, follow the steps below:

1. Set the SLEEPDEEP bit of the Cortex-M7 processor.
2. Set the VROFF bit of SUPC_CR.

Exit from Backup mode occurs as a result of one of the following enabled wake-up events:

- WKUP0–13 pins (level transition, configurable debouncing)
- Supply Monitor alarm
- RTC alarm
- RTT alarm

### 7.6.2 Wait Mode

The purpose of Wait mode is to achieve very low-power consumption while maintaining the whole device in a powered state for a startup time of less than 10 µs.

In Wait mode, the clocks of the core, peripherals and memories are stopped. However, the core, peripherals and memories power supplies are still powered.

Wait mode is entered when the WAITMODE bit is set in CKGR_MOR and the field FLPM is configured to 00 or 01 in the PMC Fast Startup Mode register (PMC_FSMR).

The Cortex-M is able to handle external events or internal events to wake up the core. This is done by configuring the external lines WKUP0–13 as fast startup wake-up pins (refer to the “Fast Startup” section). RTC or RTT alarms or USB wake-up events can be used to wake up the processor. Resume from Wait mode is also achieved when a debug request occurs and the bit CDBGPWRUPREQ is set in the processor.

© 2021 Microchip Technology Inc. and its subsidiaries

To enter Wait mode, first, select the Main RC oscillator as Main Clock and perform the following steps:

1. Configure the FLPM field in the PMC_FSMR.
2. Set Flash Wait State at 0.
3. Set HCLK = MCK by configuring MDIV to 0 in the PMC Master Clock register (PMC_MCKR).
4. Set the WAITMODE bit in the PMC Clock Generator Main Oscillator register (CKGR_MOR).
5. Wait for MCKRDY = 1 in the PMC Status register (PMC_SR).

**Note:** Internal main clock resynchronization cycles are necessary between writing the MOSCRCEN bit and the entry in Wait mode. Depending on the user application, waiting for the MOSCRCEN bit to be cleared is recommended to ensure that the core will not execute undesired instructions.

### 7.6.3 Sleep Mode

The purpose of Sleep mode is to optimize power consumption of the device versus response time. In this mode, only the core clock is stopped. The peripheral clocks can be enabled. The current consumption in this mode is application-dependent.

This mode is entered using the instruction Wait for Interrupt (WFI).

Processor wakeup is triggered by an interrupt if the WFI instruction of the Cortex-M processor is used.

### 7.6.4 Low-Power Mode Summary Table

The modes detailed above are the main low-power modes. Each part can be set to on or off separately and wake up sources can be individually configured. The following table provides a summary of the configurations of the low-power modes.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Table 7-2. Low-power Mode Configuration Summary**

| Mode | SUPC, 32 kHz Oscillator, RTC, RTT Backup SRAM (BRAM), Backup Registers (GPBR), POR (Backup Area) | Regulator | Core Memory Peripherals | Mode Entry Configuration | Potential Wakeup Sources | Core at Wakeup | PIO State while in Low-Power Mode | PIO State at Wakeup | Wakeup Time (see Note 2) |
|---|---|---|---|---|---|---|---|---|---|
| Backup Mode | ON | OFF | OFF<br>(Not powered) | SUPC_CR.VROFF = 1<br>SLEEPDEEP = 1<br>(see **Note 1)** | WKUP0–13 pins<br>Supply Monitor<br>RTC alarm<br>RTT alarm | Reset | Previous state maintained | PIOA, PIOB, PIOC, PIOD & PIOE inputs with pullups | < 2 ms |
| Wait Mode w/Flash in Deep Power-down Mode | ON | ON | Powered<br>(Not clocked) | PMC_MCKR.MDIV = 0<br>CKGR_MOR.WAITMODE =1<br>SLEEPDEEP = 0<br>PMC_FSMR.LPM = 1<br>PMC_FSMR.FLPM = 1<br>(see **Note 1)** | WKUP0–13 pins<br>RTC<br>RTT<br>USBHS<br>Processor debug (see **Note 6**)<br>GMAC Wake on LAN event<br>Wakeup from CAN (see **Note 7**) | Clocked back (see **Note 3**) | Previous state maintained | Unchanged | < 10 µs |
| Wait Mode w/Flash in Standby Mode | ON | ON | Powered<br>(Not clocked) | PMC_MCKR.MDIV = 0<br>CKGR_MOR.WAITMODE =1<br>SLEEPDEEP = 0<br>PMC_FSMR.LPM = 1<br>PMC_FSMR.FLPM = 0<br>(see **Note 1**) | WKUP0–13 pins<br>RTC<br>RTT<br>USBHS<br>Processor debug (see **Note 6)**<br>GMAC Wake on LAN<br>Wakeup from CAN (see **Note 7**) | Clocked back (see **Note 3**) | Previous state maintained | Unchanged | < 10 µs |

© 2021 Microchip Technology Inc. and its subsidiaries

| .....continued | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Mode** | **SUPC, 32 kHz Oscillator, RTC, RTT Backup SRAM (BRAM), Backup Registers (GPBR), POR (Backup Area)** | **Regulator** | **Core Memory Peripherals** | **Mode Entry Configuration** | **Potential Wakeup Sources** | **Core at Wakeup** | **PIO State while in Low-Power Mode** | **PIO State at Wakeup** | **Wakeup Time (see Note 2)** |
| Sleep Mode | ON | ON | Powered (Not clocked) (see **Note 4**) | WFI<br>SLEEPDEEP = 0<br>PMC_FSMR.LPM = 0<br>(see **Note 1**) | Any enabled Interrupt | Clocked back | Previous state maintained | Unchanged | (see **Note 5**) |

© 2021 Microchip Technology Inc. and its subsidiaries

**Notes:**

1. The bit SLEEPDEEP is in the Cortex-M7 System Control Register.
2. When considering wakeup time, the time required to start the PLL is not taken into account. Once started, the device works with the Main RC oscillator. The user has to add the PLL startup time if it is needed in the system. The wakeup time is defined as the time taken for wakeup until the first instruction is fetched.
3. HCLK = MCK. The user may need to revert back to the previous clock configuration.
4. Depends on MCK frequency.
5. In this mode, the core is supplied and not clocked. Some peripherals can be clocked.
6. Resume from Wait mode if a debug request occurs (CDBGPWRUPREQ is set in the processor).
7. CAN wake-up requires the use of any WKUP0–13 pin.

## 7.7 Wakeup Sources

Wakeup events allow the device to exit Backup mode. When a wakeup event is detected, the Supply Controller performs a sequence which automatically reenables the core power supply and the SRAM power supply, if they are not already enabled.

## 7.8 Fast Startup

The SAMV71Q21RT allows the processor to restart in a few microseconds while the processor is in Wait mode or in Sleep mode. A fast startup can occur upon detection of a low level on any of the following wake-up sources:

- WKUP0 to WKUP13 pins
- Supply Monitor
- RTC alarm
- RTT alarm
- USBHS interrupt line (WAKEUP)
- Processor debug request (CDBGPWRUPREQ)
- GMAC wake on LAN event

**Note:** CAN wake-up requires the use of any WKUP0–13 pin.

The fast restart circuitry is fully asynchronous and provides a fast startup signal to the Power Management Controller. As soon as the fast startup signal is asserted, the PMC automatically restarts the Main RC oscillator, switches the Master clock on this clock and re-enables the processor clock.

# 8. Input/Output Lines

The SAMV71Q21RT features both general purpose I/Os (GPIO) and system I/Os. GPIOs can have alternate functionality due to multiplexing capabilities of the PIO controllers. The same PIO line can be used, whether in I/O mode or by the multiplexed peripherals. System I/Os include pins such as test pins, oscillators, erase or analog inputs.

## 8.1 General-Purpose I/O Lines

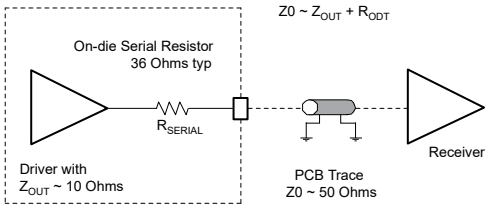
General-purpose (GPIO) lines are managed by PIO Controllers. All I/Os have several input or output modes such as pull-up or pull-down, input Schmitt triggers, multi-drive (open-drain), glitch filters, debouncing or input change interrupt. Programming of these modes is performed independently for each I/O line through the PIO controller user interface. For more details, refer to 32. Parallel Input/Output Controller (PIO).

The input/output buffers of the PIO lines are supplied through VDDIO power supply rail.

The SAMV71Q21RT embeds high-speed pads able to handle the high-speed clocks for HSMCI, SPI and QSPI (MCK/2). Refer to Electrical Characteristics for more details. Typical pull-up and pull-down value is 100 kΩ for all I/Os.

Each I/O line also embeds an $R_{SERIAL}$ (On-die Serial Resistor), as shown in the following figure. It consists of an internal series resistor termination scheme for impedance matching between the driver output (SAMV71Q21RT) and the PCB trace impedance preventing signal reflection. The series resistor helps to reduce I/Os switching current (di/dt), thereby reducing EMI. It also decreases overshoot and undershoot (ringing) due to inductance of interconnect between devices or between boards. Finally, $R_{SERIAL}$ helps diminish signal integrity issues.

**Figure 8-1. On-Die Termination (ODT)**




## 8.2 System I/O Lines

System I/O lines are pins used by oscillators, Test mode, reset, JTAG and other features. The following table lists the SAMV71Q21RT system I/O lines shared with PIO lines.

These pins are software-configurable as general-purpose I/Os or system pins. At startup, the default function of these pins is always used.

**Table 8-1. System I/O Configuration Pin List**

| CCFG_SYSIO Bit Number | Default Function After Reset | Other Function | Constraints for Normal Start | Configuration |
|---|---|---|---|---|
| 12 | ERASE | PB12 | Low Level at startup (see **Note 1**) | In Matrix User Interface Registers (Refer to the 19.4.7 CCFG_SYSIO register) |
| 7 | TCK/SWCLK | PB7 | – | |
| 6 | TMS/SWDIO | PB6 | – | |
| 5 | TDO/TRACESWO | PB5 | – | |
| 4 | TDI | PB4 | – | |

| .....continued | | | | |
|---|---|---|---|---|
| CCFG_SYSIO Bit Number | Default Function After Reset | Other Function | Constraints for Normal Start | Configuration |
| – | PA7 | XIN32 | – | (see **Note 2**) |
| – | PA8 | XOUT32 | – | |
| – | PB9 | XIN | – | (see **Note 3**) |
| – | PB8 | XOUT | – | |

**Notes:**

1. If PB12 is used as PIO input in user applications, a low level must be ensured at startup to prevent Flash erase before the user application sets PB12 into PIO mode.
2. Refer to 23.4.2 Slow Clock Generator.
3. Refer to 30.5.3 Main Crystal Oscillator.

### 8.2.1 Serial Wire Debug Port (SW-DP) Pins

The SW-DP pins SWCLK and SWDIO are commonly provided on a standard 20-pin JTAG connector defined by ARM. For more details about voltage reference and reset state, refer to Table 4-1.

At startup, SW-DP pins are configured in SW-DP mode to allow connection with debugging probe. For more details, refer to 16. Debug and Test Features.

SW-DP pins can be used as standard I/Os to provide users more general input/output pins when the debug port is not needed in the end application. Mode selection between SW-DP mode (System IO mode) and general IO mode is performed through the AHB Matrix Special Function Registers (MATRIX_SFR). Configuration of the pad for pull-up, triggers, debouncing and glitch filters is possible regardless of the mode.

The JTAGSEL pin is used to select the JTAG boundary scan when asserted at a high level. It integrates a permanent pulldown resistor of about 15 kΩ to GND, so that it can be left unconnected for normal operations.

The JTAG Debug Port TDI, TDO, TMS and TCK is inactive. It is provided for Boundary Scan Manufacturing Test purpose only.

### 8.2.2 Embedded Trace Module (ETM) Pins

The Embedded Trace Module (ETM) depends on the Trace Port Interface Unit (TPIU) to export data out of the system.

The TPUI features the following pins:

- TRACECLK is always exported to enable synchronization with the data.
- TRACED0–TRACED3 is the instruction trace stream.

## 8.3 NRST Pin

The NRST pin is bidirectional. It is handled by the on-chip Reset Controller (RSTC) and can be driven low to provide a reset signal to the external components or asserted low externally to reset the microcontroller. It resets the core and the peripherals, with the exception of the Backup area (RTC, RTT, Backup SRAM and Supply Controller). The NRST pin integrates a permanent pullup resistor to VDDIO of about 100 kΩ.

By default, the pin is configured as an input.

## 8.4 ERASE Pin

The ERASE pin is used to reinitialize the Flash content and some of its NVM bits to an erased state (all bits read as logic level 1). The ERASE pin and the ROM code ensure an in-situ reprogrammability of the Flash content without the use of a debug tool. When the security bit is activated, the ERASE pin provides the capability to reprogram the Flash content. The ERASE pin integrates a pull-down resistor of about 100 kΩ to GND, so that it can be left unconnected for normal operations.

The ERASE pin is a system I/O pin that can be used as a standard I/O. At startup, this system I/O pin defaults to the ERASE function. This pin is debounced by SLCK to improve the glitch tolerance. To avoid unexpected erase at power-up due to glitches, a minimum ERASE pin assertion time is required. This time is defined in the section "Embedded Flash Characteristics".

The erase operation cannot be performed when the system is in Wait mode.

If the ERASE pin is used as a standard I/O in Input or Output mode, note the following considerations and behavior:

- I/O Input mode: at startup of the device, the logic level of the pin must be low to prevent unwanted erasing until the user application has reconfigured this system I/O pin to a standard I/O pin.
- I/O Output mode: asserting the pin to low does not erase the Flash

During software application development, faulty software may put the device into a deadlock. This may be due to:

- Programming an incorrect clock switching sequence
- Using this system I/O pin as a standard I/O pin
- Entering Wait mode without any wake-up events programmed

To recover normal behavior, the Flash must be erased by following the steps below:

1. Apply a logic “1” level on the ERASE pin.
2. Apply a logic “0” level on the NRST pin.
3. Maintain the ERASE pin to logic “1” level for at least the minimum assertion time after releasing the NRST pin to logic “1” level.

© 2021 Microchip Technology Inc.
and its subsidiaries

# 9. Interconnect

The system architecture is based on the ARM Cortex-M7 processor connected to the main AHB Bus Matrix, the embedded Flash, the multi-port SRAM and the ROM.

The 32-bit AHBP interface is a single 32-bit wide interface that accesses the peripherals connected on the main Bus Matrix. It is used only for data access. Instruction fetches are never performed on the AHBP interface. The bus, AHBP or AXIM, accessing the peripheral memory area [0x40000000 to 0x60000000] is selected in the AHBP control register.

The 32-bit AHBS interface provides system access to the ITCM, D1TCM, and D0TCM. It is connected on the main Bus Matrix and allows the XDMA to transfer from memory or peripherals to the instruction or data TCMs.

The 64-bit AXIM interface is a single 64-bit wide interface connected through two ports of the AXI Bridge to the main AHB Bus Matrix and to two ports of the multi-port SRAM. The AXIM interface allows:

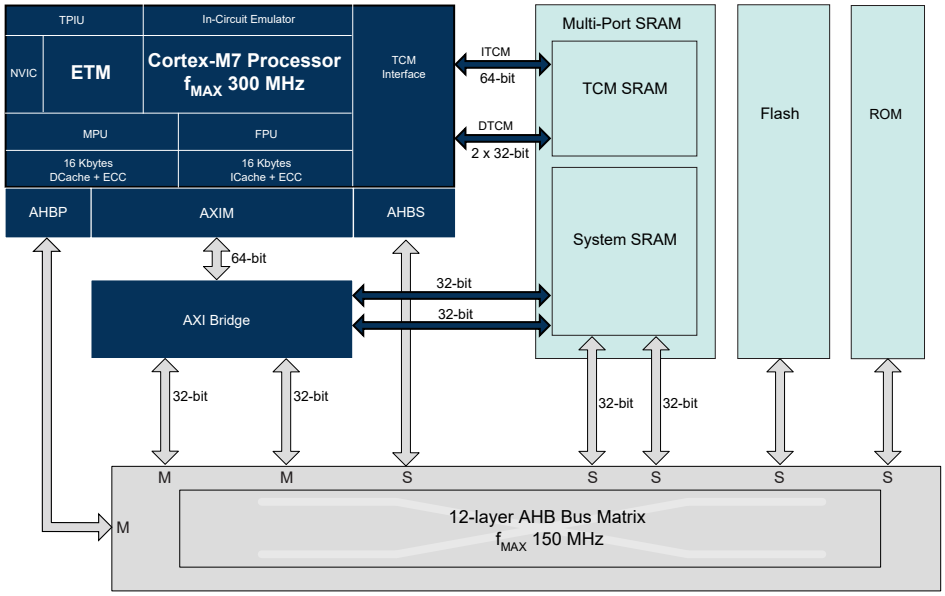
- Instruction fetches
- Data cache linefills and evictions
- Non-cacheable normal-type memory data accesses
- Device and strongly-ordered type data accesses, generally to peripherals

The interleaved multi-port SRAM optimizes the Cortex-M7 accesses to the internal SRAM.

The interconnect of the other masters and slaves is described in 19. Bus Matrix (MATRIX).

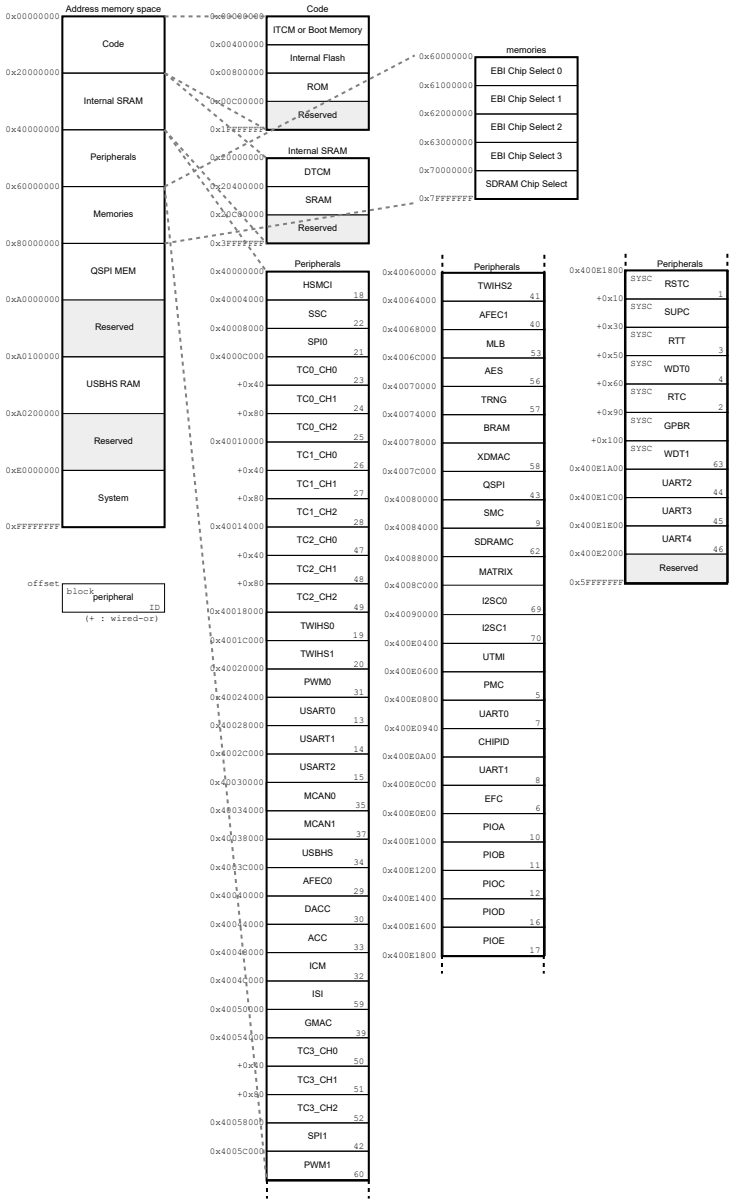
The figure below shows the connections of the different Cortex-M7 ports.

**Figure 9-1. Interconnect Block Diagram**



# 10. Product Mapping

**Figure 10-1. SAMV71Q21RT Product Mapping**



# 11. Memories

## 11.1 Embedded Memories

### 11.1.1 Internal SRAM

The SAMV71Q21RT embeds 384 Kbytes of high-speed SRAM.

The SRAM is accessible over the system Cortex-M bus at address 0x2040 0000.

The SAMV71Q21RT embeds a Multi-Port SRAM with four ports to optimize the bandwidth and latency. The priorities, defined in the Bus Matrix for each SRAM port slave are propagated, for each request, up to the SRAM slaves.

The Bus Matrix supports four priority levels: Normal, Bandwidth-sensitive, Latency-sensitive and Latency-critical in order to increase the overall processor performance while securing the high-priority latency-critical requests from the peripherals.

The SRAM controller manages interleaved addressing of SRAM blocks to minimize access latencies. It uses Bus Matrix priorities to give the priority to the most urgent request. The less urgent request is performed no later than the next cycle.

Two SRAM slave ports are dedicated to the Cortex-M7 while two ports are shared by the AHB masters.

### 11.1.2 Tightly Coupled Memory (TCM) Interface

The SAMV71Q21RT embeds Tightly Coupled Memory (TCM) running at processor speed.

- ITCM is a single 64-bit interface, based at 0x0000 0000 (code region).
- DTCM is composed of dual 32-bit interfaces interleaved, based at 0x2000 0000 (data region).

ICTM and DTCM are enabled/disabled in the ITCMR and DTCMR registers in Arm SCB.

DTCM is enabled at reset by default. ITCM is disabled by default at reset.

There are four TCM configurations controlled by software. When enabled, ITCM is located at 0x0000 0000, overlapping ROM or Flash depending on the general-purpose NVM bit 1 (GPNVM). The configuration is done with GPNVM bits [8:7].

**Table 11-1. TCM Configurations in Kbytes**

| ITCM | DTCM | SRAM for 384K RAM-based | GPNVM Bits [8:7] |
|---|---|---|---|
| 0 | 0 | 384 | 0 |
| 32 | 32 | 320 | 1 |
| 64 | 64 | 256 | 2 |
| 128 | 128 | 128 | 3 |

Accesses made to TCM regions when the relevant TCM is disabled and accesses made to the Code and SRAM region above the TCM size limit are performed on the AHB matrix, i.e., on internal Flash or on ROM depending on remap GPNVM bit.

Accesses made to the SRAM above the size limit will not generate aborts.

The Memory Protection Unit (MPU) can to be used to protect these areas.

### 11.1.3 Internal ROM

The SAMV71Q21RT embeds an Internal ROM for the SAM Boot Assistant (SAM-BA®), In Application Programming functions (IAP) and Fast Flash Programming Interface (FFPI).

At any time, the ROM is mapped at address 0x0080 0000.

The ROM may also be mapped at 0x00000000 depending on GPNVM bit setting and ITCM use.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 11.1.4 Backup SRAM

The SAMV71Q21RT embeds 1 Kbytes of backup SRAM located at 0x4007 4000.

The backup SRAM is accessible in 32-bit words only. Byte or half-word accesses are not supported.

The backup SRAM is supplied by VDDCORE in Normal mode.

In Backup mode, the backup SRAM supply is automatically switched to VDDIO through the backup SRAM power switch when VDDCORE falls. For more details, see the "Backup SRAM Power Switch" section.

### 11.1.5 Flash Memories

The SAMV71Q21RT embeds 2084 Kbytes of internal Flash mapped at address 0x00400000.

The device features a Quad SPI (QSPI) interface, mapped at address 0x80000000, that extends the Flash size by adding an external SPI or QSPI Flash.

When accessed by the Cortex-M7 processor for programming operations, the QSPI and internal Flash address spaces must be defined in the Cortex-M7 memory protection unit (MPU) with the attribute 'Device' or 'Strongly Ordered'. For fetch or read operations, the attribute 'Normal memory' must be set to benefit from the internal cache. Refer to the Arm Cortex-M7 Technical Reference Manual (ARM DDI 0489) available on www.arm.com.

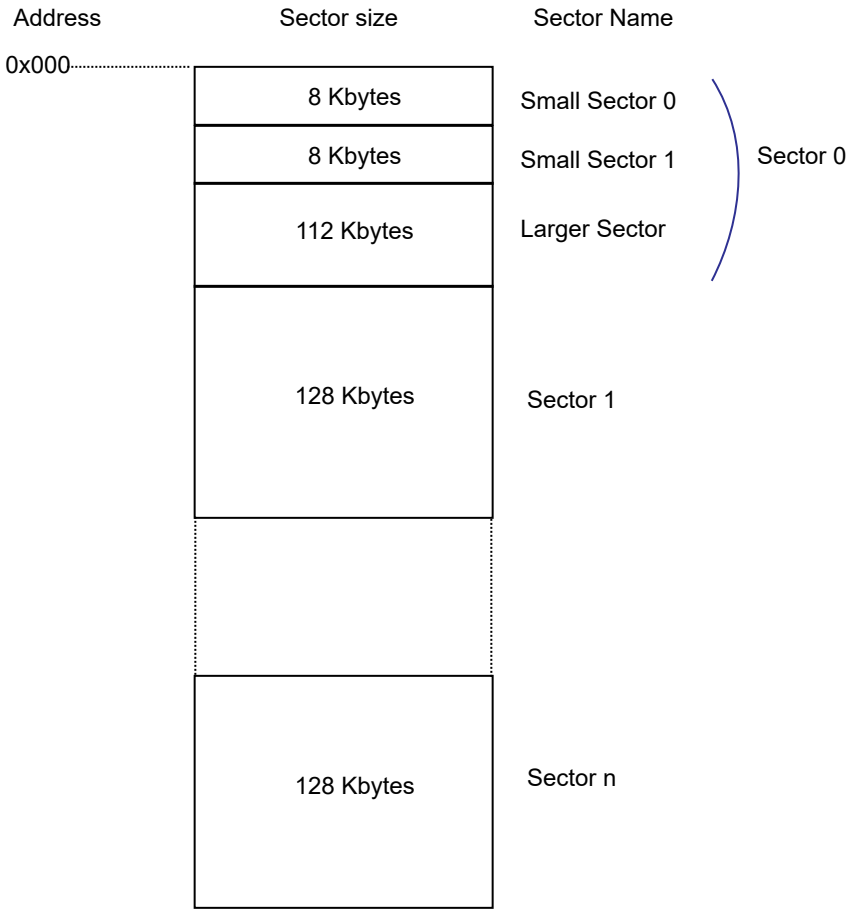
Some precautions must be taken when the accesses are performed by the central DMA. Refer to 22. Enhanced Embedded Flash Controller (EEFC) and 42. Quad Serial Peripheral Interface (QSPI).

#### 11.1.5.1 Embedded Flash Overview

The memory is organized in sectors. Each sector has a size of 128 Kbytes. The first sector is divided into three smaller sectors.

The three smaller sectors are organized in two sectors of 8 Kbytes and one sector of 112 Kbytes. Refer to the figure below.

**Figure 11-1. Global Flash Organization**



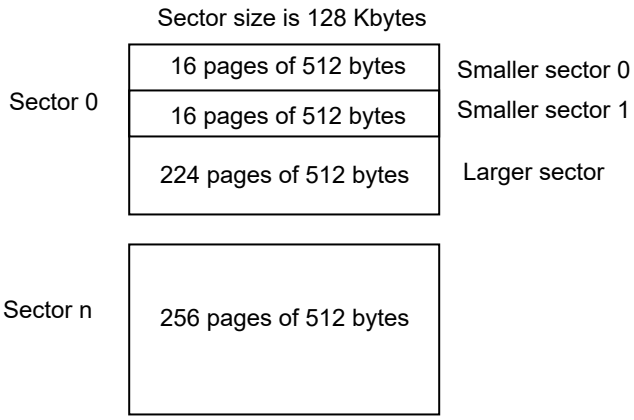

Each sector is organized in pages of 512 bytes.

For sector 0:

- The smaller sector 0 has 16 pages of 512 bytes
- The smaller sector 1 has 16 pages of 512 bytes
- The larger sector has 224 pages of 512 bytes

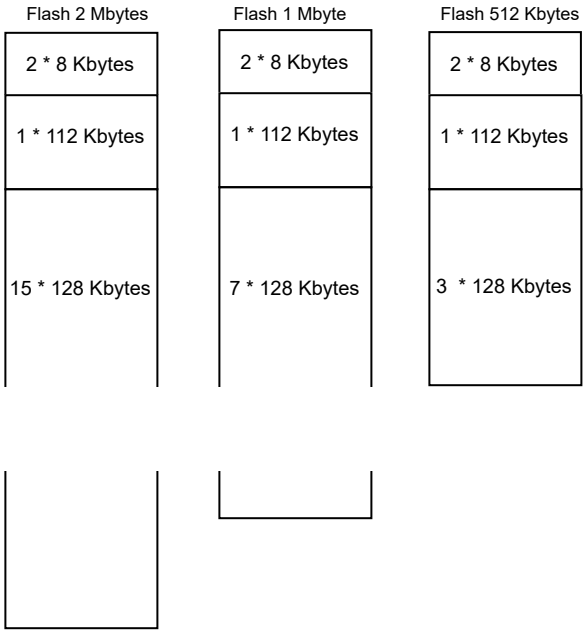
The rest of the array is composed of 128-Kbyte sectors of 256 pages of 512 bytes each. See below.

**Figure 11-2. Flash Sector Organization**




The figure below illustrates the organization of the Flash depending on its size.

**Figure 11-3. Flash Size**




Erasing the memory can be performed:

- Chip Erase
- By block of 8 Kbytes
- By sector of 128 Kbytes
- By 512-byte page
  - Erase memory by page is possible only in an 8 Kbyte sector
  - EWP and EWPL commands can be only used in 8 Kbyte sectors

The memory has one additional reprogrammable page that can be used as page signature by the user. It is accessible through specific modes, for erase, write and read operations. Erase pin assertion will not erase the User Signature page.

#### 11.1.5.2 Enhanced Embedded Flash Controller

Each Enhanced Embedded Flash Controller manages accesses performed by the masters of the system. It enables reading the Flash and writing the write buffer. It also contains a User Interface, mapped on the APB.

The Enhanced Embedded Flash Controller ensures the interface of the Flash block.

It manages the programming, erasing, locking and unlocking sequences of the Flash using a full set of commands.

One of the commands returns the embedded Flash descriptor definition that informs the system about the Flash organization, thus making the software generic.

#### 11.1.5.3 Flash Speed

The user must set the number of wait states depending on the system frequency.

For more details, refer to Embedded Flash Characteristics in the Electrical Characteristics section.

#### 11.1.5.4 Lock Regions

Several lock bits are used to protect write and erase operations on lock regions. A lock region is composed of several consecutive pages, and each lock region has its associated lock bit.

**Table 11-2. Flash Lock Bits**

| Flash Size (Kbytes) | Number of Lock Bits | Lock Region Size |
|---|---|---|
| 2048 | 128 | 16 Kbytes |

| .....continued | | |
|---|---|---|
| **Flash Size (Kbytes)** | **Number of Lock Bits** | **Lock Region Size** |
| 1024 | 64 | 16 Kbytes |
| 512 | 32 | 16 Kbytes |

Asserting the ERASE pin clears the lock bits, thus unlocking the entire Flash.

#### 11.1.5.5 Security Bit Feature

The SAMV71Q21RT features a security bit based on the GPNVM bit 0. When security is enabled, any access to the Flash, SRAM, core registers and internal peripherals, either through the SW-DP, the ETM interface or the Fast Flash Programming Interface, is blocked. This ensures the confidentiality of the code programmed in the Flash.

This security bit can only be enabled through the command “Set General-purpose NVM Bit 0” of the EEFC User Interface. Disabling the security bit can only be achieved by asserting the ERASE pin at 1, and after a full Flash erase is performed. When the security bit is deactivated, all accesses to the Flash, SRAM, Core registers, Internal Peripherals are permitted.

#### 11.1.5.6 Unique Identifier

The device contains a unique identifier of 2 pages of 512 bytes. These 2 pages are read-only and cannot be erased even by the ERASE pin.

The sequence to read the unique identifier area is described in 22.4.3.8 Unique Identifier Area.

The mapping is as follows:

- Bytes [0..15]: 128 bits for unique identifier
- Bytes[16..1023]: Reserved

#### 11.1.5.7 User Signature

Each device contains a user signature of 512 bytes that is available to the user. The user signature can be used to store information such as trimming, keys, etc., that the user does not want to be erased by asserting the ERASE pin or by software ERASE command. Read, write and erase of this area is allowed.

#### 11.1.5.8 Fast Flash Programming Interface (FFPI)

The Fast Flash Programming Interface (FFPI) allows programming the device through a multiplexed fully-handshaked parallel port. It allows gang programming with market-standard industrial programmers.

The FFPI supports read, page program, page erase, full erase, lock, unlock and protect commands.

The FFPI is enabled and the Fast Programming mode is entered when TST and PA3 and PA4 are tied low.

**Table 11-3. FFPI on PIO Controller A (PIOA)**

| I/O Line | System Function |
|---|---|
| PD10 | PGMEN0 |
| PD11 | PGMEN1 |
| PB0 | PGMM0 |
| PB1 | PGMM1 |
| PB2 | PGMM2 |
| PB3 | PGMM3 |
| PA3 | PGMNCMD |
| PA4 | PGMRDY |
| PA5 | PGMNOE |
| PA21 | PGMNVALID |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | |
|---|---|
| **I/O Line** | **System Function** |
| PA7 | PGMD0 |
| PA8 | PGMD1 |
| PA9 | PGMD2 |
| PA10 | PGMD3 |
| PA11 | PGMD4 |
| PA12 | PGMD5 |
| PA13 | PGMD6 |
| PA14 | PGMD7 |
| PD0 | PGMD8 |
| PD1 | PGMD9 |
| PD2 | PGMD10 |
| PD3 | PGMD11 |
| PD4 | PGMD12 |
| PD5 | PGMD13 |
| PD6 | PGMD14 |
| PD7 | PGMD15 |

#### 11.1.5.9 SAM-BA Boot

The SAM-BA Boot is a default boot program which provides an easy way to program in-situ the on-chip Flash memory.

The SAM-BA Boot Assistant supports serial communication via the UART0 and USB[1].

The SAM-BA Boot provides an interface with SAM-BA computer application.

The SAM-BA Boot is in ROM at address 0x0 when the bit GPNVM1 is set to 0.

**Note:**

1. USB is not supported on this device.

#### 11.1.5.10 General-purpose NVM (GPNVM) Bits

All SAMV71Q21RT devices feature nine general-purpose NVM (GPNVM) bits that can be cleared or set, through the “Clear GPNVM Bit” and “Set GPNVM Bit” commands of the EEFC User Interface.

The GPNVM0 bit is the security bit.

The GPNVM1 bit is used to select the Boot mode (Boot always at 0x00) on ROM or Flash.

**Table 11-4. General-purpose Non volatile Memory Bits**

| GPNVM Bit | Function |
|---|---|
| 0 | Security bit |
| 1 | Boot mode selection<br>0: ROM (default)<br>1: Flash |
| 5:2 | Free |
| 6 | Reserved |

| .....continued | |
|---|---|
| **GPNVM Bit** | **Function** |
| 8:7 | TCM configuration<br>00: 0 Kbytes DTCM + 0 Kbytes ITCM (default)<br>01: 32 Kbytes DTCM + 32 Kbytes ITCM<br>10: 64 Kbytes DTCM + 64 Kbytes ITCM<br>11: 128 Kbytes DTCM + 128 Kbytes ITCM<br>**Note:** After programming, reboot must be done. |

### 11.1.6 Boot Strategies

The system always boots at address 0x0. To ensure maximum boot possibilities, the memory layout can be changed using GPNVM bits.

A GPNVM bit is used to boot either on the ROM (default) or from the Flash.

The GPNVM bit can be cleared or set, respectively, through the commands “Clear General-purpose NVM Bit” and “Set General-purpose NVM Bit” of the EEFC User Interface.

Setting the bit GPNVM1 selects boot from the Flash. Clearing it selects boot from the ROM. Asserting ERASE resets the bit GPNVM1 and thus selects boot from ROM.

## 11.2 External Memories

The SAMV71Q21RT features one External Bus Interface to provide an interface to a wide range of external memories and to any parallel peripheral.

# 12. Event System

The events generated by peripherals (source) are designed to be directly routed to peripherals (destination) using these events without processor intervention. The trigger source can be programmed in the destination peripheral.

## 12.1 Embedded Characteristics

- Timers, PWM, I/Os and peripherals generate event triggers which are directly routed to destination peripherals, such as AFEC or DACC to start measurement/conversion without processor intervention.
- UART, USART, QSPI, SPI, TWI, PWM, HSMCI, AES, AFEC, DACC, PIO, TC (Capture mode) also generate event triggers directly connected to the DMA Controller for data transfer without processor intervention.
- Parallel capture logic is directly embedded in the PIO and generates trigger events to the DMA Controller to capture data without processor intervention.
- PWM safety events (faults) are in combinational form and directly routed from event generators (AFEC, ACC, PMC, TC) to the PWM module.
- PWM output comparators (OCx) generate events directly connected to the TC.
- PMC safety event (clock failure detection) can be programmed to switch the MCK on reliable main RC internal clock without processor intervention.

© 2021 Microchip Technology Inc.
and its subsidiaries

## 12.2 Real-time Event Mapping

**Table 12-1. Real-time Event Mapping List**

| Function | Application | Description | Event Source | Event Destination |
|---|---|---|---|---|
| Safety | General-purpose | Automatic switch to reliable main RC oscillator in case of main crystal clock failure (see **Note 1**) | Power Management Controller (PMC) | PMC |
| | General-purpose, motor control, power factor correction (PFC) | Puts the PWM outputs in Safe mode in case of main crystal clock failure (see **Notes 1, 2**) | PMC | Pulse Width Modulation 0 and 1 (PWM0 and PWM1) |
| | Motor control, PFC | Puts the PWM outputs in Safe mode (overcurrent detection, etc.) (see **Notes 2, 3**) | Analog Comparator Controller (ACC) | PWM0 and PWM1 |
| | Motor control, PFC | Puts the PWM outputs in Safe mode (overspeed, overcurrent detection, etc.) (see **Notes 2, 4**) | Analog Front-End Controller (AFEC0) | PWM0 and PWM1 |
| | | | AFEC1 | PWM0 and PWM1 |
| | Motor control | Puts the PWM outputs in Safe mode (overspeed detection through timer quadrature decoder) (see **Notes 2, 6**) | TC0 | PWM0 |
| | | | TC1 | PWM1 |
| | General-purpose, motor control, power factor correction (PFC) | Puts the PWM outputs in Safe mode (general-purpose fault inputs) (see **Note 2**) | PIO PA9, PD8, PD9 | PWM0 |
| | | | PIO PA21, PA26, PA28 | PWM1 |
| Security | General-purpose | Immediate GPBR clear (asynchronous) on tamper detection through WKUP0/1 IO pins (see **Note 5**) | PIO WKUP0/1 | GPBR |

.....continued

| Function | Application | Description | Event Source | Event Destination |
|---|---|---|---|---|
| Measurement trigger | Power factor correction (DC-DC, lighting, etc.) | Duty cycle output waveform correction<br>Trigger source selection in PWM (see **Notes 7, 8**) | ACC | PWM0 |
| | | | PIO PA10, PA22 | PWM0 |
| | | | ACC | PWM1 |
| | | | PIO PA30, PA18 | PWM1 |
| | General-purpose | Trigger source selection in AFEC (see **Note 9**) | PIO AFE0_ADTRG | AFEC0 |
| | | | TC0 TIOA0 | AFEC0 |
| | | | TC0 TIOA1 | AFEC0 |
| | | | TC0 TIOA2 | AFEC0 |
| | | | ACC | AFEC0 |
| | Motor control | ADC-PWM synchronization (see **Notes 12, 14**) Trigger source selection in AFEC (see **Note 9**) | PWM0 Event Line 0 and 1 | AFEC0 |
| | General-purpose | Trigger source selection in AFEC (see **Note 9**) | PIO AFE1_ADTRG | AFEC1 |
| | | | TC1 TIOA3 | AFEC1 |
| | | | TC1 TIOA4 | AFEC1 |
| | | | TC1 TIOA5 | AFEC1 |
| | | | ACC | AFEC1 |
| | Motor control | ADC-PWM synchronization (see **Notes 12, 14**)<br>Trigger source selection in AFEC (see **Note 9**) | PWM1 Event Line 0 and 1 | AFEC1 |
| | General-purpose | Temperature sensor<br>Low-speed measurement (see **Notes 10, 11**) | RTC RTCOUT0 | AFEC0 and AFEC1 |
| Conversion trigger | General-purpose | Trigger source selection in DACC (Digital-to-Analog Converter Controller) (see **Note 13**) | TC0 TIOA0, TIOA1, TIOA2 | DACC |
| | | | PIO DATRG | DACC |
| | | | PWM0 Event Line 0 and 1(14) | DACC |
| | | | PWM1 Event Line 0 and 1(14) | DACC |
| Image capture | Low-cost image sensor | Direct image transfer from sensor to system memory via DMA(15) | PIO PA3/4/5/9/10/11/12/13, PA22, PA14, PA21 | DMA |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Function | Application | Description | Event Source | Event Destination |
|---|---|---|---|---|
| Delay measurement | Motor control | Propagation delay of external components (IOs, power transistor bridge driver, etc.) See **Notes 16, 17**) | PWM0 Comparator Output OC0 | TC0 TIOA0 and TIOB0 |
| | | | PWM0 Comparator Output OC1 | TC0 TIOA1 and TIOB1 |
| | | | PWM0 Comparator Output OC2 | TC0 TIOA2 and TIOB2 |
| | | | PWM1 Comparator Output OC0 | TC1 TIOA3 and TIOB3 |
| | | | PWM1 Comparator Output OC1 | TC1 TIOA4 and TIOB4 |
| | | | PWM1 Comparator Output OC2 | TC1 TIOA5 and TIOB5 |
| | | | PWM0 Comparator Output OC0 | TC2 TIOA6 and TIOB6 |
| | | | PWM0 Comparator Output OC1 | TC2 TIOA7 and TIOB7 |
| | | | PWM0 Comparator Output OC2 | TC2 TIOA8 and TIOB8 |
| | | | PWM1 Comparator Output OC0 | TC3 TIOA9 and TIOB9 |
| | | | PWM1 Comparator Output OC1 | TC3 TIOA10 and TIOB10 |
| Audio clock recovery from Ethernet | Audio | GMAC GTSUCOMP signal adaptation via TC (TC_EMR.TRIGSRCB) in order to drive the clock reference of the external PLL for the audio clock | GMAC GTSUCOMP | TC3 TIOB11 |
| Direct Memory Access | General-purpose | Peripheral trigger event generation to transfer data to/from system memory (see **Note 18**) | USART, UART, TWIHS, SPI, QSPI, AFEC, TC (Capture), SSC, HSMCI, DAC, AES, PWM, PIO, I2SC | XDMA |

**Notes:**

1. Refer to 31.15 Main Crystal Oscillator Failure Detection.
2. Refer to 51.5.4 Fault Inputs and 51.6.2.7 Fault Protection.
3. Refer to 54.6.4 Fault Mode.
4. Refer to 54.5.4 Fault Output.
5. Refer to 23.4.9.2 Low-power Tamper Detection and Anti-Tampering and 29.3.1 SYS_GPBRx.
6. Refer to 50.6.18 Fault Mode.
7. Refer to 51.7.49 PWM_ETRGx.
8. Refer to 51.6.5 PWM External Trigger Mode.
9. Refer to 52.6.6 Conversion Triggers and 52.7.2 AFEC_MR.
10. Refer to Temperature Sensor.
11. Refer to 27.5.8 Waveform Generation.
12. Refer to 51.7.36 PWM_CMPVx and 51.6.4 PWM Event Lines.
13. Refer to 53.7.3 DACC_TRIGR.
14. Refer to 51.6.3 PWM Comparison Units and 51.6.4 PWM Event Lines.
15. Refer to 32.5.14 Parallel Capture Mode.
16. Refer to 51.6.2.2 Comparator.
17. Refer to 50.6.14 Synchronization with PWM.
18. Refer to 36. DMA Controller (XDMAC).

© 2021 Microchip Technology Inc.
and its subsidiaries

# 13. System Controller

The System Controller is a set of peripherals that handles key elements of the system, such as power, resets, clocks, time, interrupts, watchdog, and so on.

## 13.1 System Controller and Peripherals Mapping

Refer to the Product Mapping section.

## 13.2 Power-on-Reset, Brownout and Supply Monitor

The SAMV71Q21RT embeds three features to monitor, warn and/or reset the chip:

- Power-on-Reset (POR) on VDDIO
- POR on VDDCORE
- Brown-out-Detector (BOD) on VDDCORE
- Supply Monitor on VDDIO

### 13.2.1 Power-on-Reset

The Power-on-Reset monitors VDDIO and VDDCORE. It is always activated and monitors voltage at start up but also during power down. If VDDIO or VDDCORE goes below the threshold voltage, the entire chip is Reset. For more information, refer to Electrical Characteristics.

### 13.2.2 Brownout Detector on VDDCORE

The Brownout Detector monitors VDDCORE. It is active by default. It can be deactivated by software through the Supply Controller (SUPC_MR). It is especially recommended to disable it during low-power modes such as wait or sleep modes.

If VDDCORE goes below the threshold voltage, the reset of the core is asserted. For more information, refer to 23. Supply Controller (SUPC) and Electrical Characteristics.

### 13.2.3 Supply Monitor on VDDIO

The Supply Monitor monitors VDDIO. It is not active by default. It can be activated by software and is fully programmable with 16 steps for the threshold (between 1.6V to 3.4V). It is controlled by the Supply Controller (SUPC). A sample mode is possible, which allows the supply monitor power consumption to be divided by a factor of up to 2048. For more information, refer to 23. Supply Controller (SUPC) and Electrical Characteristics.

## 13.3 Reset Controller

The Reset Controller is based on two POR cells, one on VDDIO and one on VDDCORE, and a Supply Monitor on VDDIO.

The Reset Controller returns the source of the last reset to the software. This may be a general reset, a wakeup reset, a software reset, a user reset or a watchdog reset.

The Reset Controller controls the internal resets of the system and the pin input/output. It can shape a reset signal for the external devices, simplifying the connection of a push-button on the NRST pin to implement a manual reset.

The configuration of the Reset Controller is saved as supplied on VDDIO.

© 2021 Microchip Technology Inc.
and its subsidiaries

# 14. Peripherals

## 14.1 Peripheral Identifiers

The following table defines the peripheral identifiers of the SAMV71Q21RT. A peripheral identifier is required for the control of the peripheral interrupt with the Nested Vectored Interrupt Controller and control of the peripheral clock with the Power Management Controller.

**Table 14-1. Peripheral Identifiers**

| Instance ID | Instance Name | NVIC Interrupt | PMC Clock Control | Description |
|---|---|---|---|---|
| 0 | SUPC | X | – | Supply Controller |
| 1 | RSTC | X | – | Reset Controller |
| 2 | RTC | X | – | Real Time Clock |
| 3 | RTT | X | – | Real Time Timer |
| 4 | WDT | X | – | Watchdog Timer |
| 5 | PMC | X | – | Power Management Controller |
| 6 | EFC | X | – | Enhanced Embedded Flash Controller |
| 7 | UART0 | X | X | Universal Asynchronous Receiver/Transmitter |
| 8 | UART1 | X | X | Universal Asynchronous Receiver/Transmitter |
| 9 | SMC | – | X | Static Memory Controller |
| 10 | PIOA | X | X | Parallel I/O Controller A |
| 11 | PIOB | X | X | Parallel I/O Controller B |
| 12 | PIOC | X | X | Parallel I/O Controller C |
| 13 | USART0 | X | X | Universal Synchronous/Asynchronous Receiver/ Transmitter |
| 14 | USART1 | X | X | Universal Synchronous/Asynchronous Receiver/ Transmitter |
| 15 | USART2 | X | X | Universal Synchronous/Asynchronous Receiver/ Transmitter |
| 16 | PIOD | X | X | Parallel I/O Controller D |
| 17 | PIOE | X | X | Parallel I/O Controller E |
| 18 | HSMCI | X | X | Multimedia Card Interface |
| 19 | TWIHS0 | X | X | Two-wire Interface (I2C-compatible) |
| 20 | TWIHS1 | X | X | Two-wire Interface (I2C-compatible) |
| 21 | SPI0 | X | X | Serial Peripheral Interface |
| 22 | SSC | X | X | Synchronous Serial Controller |
| 23 | TC0_CHANNEL0 | X | X | 16-bit Timer Counter 0, Channel 0 |
| 24 | TC0_CHANNEL1 | X | X | 16-bit Timer Counter 0, Channel 1 |
| 25 | TC0_CHANNEL2 | X | X | 16-bit Timer Counter 0, Channel 2 |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Instance ID | Instance Name | NVIC Interrupt | PMC Clock Control | Description |
|---|---|---|---|---|
| 26 | TC1_CHANNEL0 | X | X | 16-bit Timer Counter 1, Channel 0 |
| 27 | TC1_CHANNEL1 | X | X | 16-bit Timer Counter 1, Channel 1 |
| 28 | TC1_CHANNEL2 | X | X | 16-bit Timer Counter 1, Channel 2 |
| 29 | AFEC0 | X | X | Analog Front-End Controller |
| 30 | DACC | X | X | Digital-to-Analog Converter |
| 31 | PWM0 | X | X | Pulse Width Modulation Controller |
| 32 | ICM | X | X | Integrity Check Monitor |
| 33 | ACC | X | X | Analog Comparator Controller |
| 34 | USBHS | X | X | USB Host / Device Controller |
| 35 | MCAN0 | X | X | CAN IRQ Line 0 |
| 36 | MCAN0 | INT1 | – | CAN IRQ Line 1 |
| 37 | MCAN1 | X | X | CAN IRQ Line 0 |
| 38 | MCAN1 | INT1 | – | CAN IRQ Line 1 |
| 39 | GMAC | X | X | Ethernet MAC |
| 40 | AFEC1 | X | X | Analog Front End Controller |
| 41 | TWIHS2 | X | X | Two-wire Interface |
| 42 | SPI1 | X | X | Serial Peripheral Interface |
| 43 | QSPI | X | X | Quad I/O Serial Peripheral Interface |
| 44 | UART2 | X | X | Universal Asynchronous Receiver/Transmitter |
| 45 | UART3 | X | X | Universal Asynchronous Receiver/Transmitter |
| 46 | UART4 | X | X | Universal Asynchronous Receiver/Transmitter |
| 47 | TC2_CHANNEL0 | X | X | 16-bit Timer Counter 2, Channel 0 |
| 48 | TC2_CHANNEL1 | X | X | 16-bit Timer Counter 2, Channel 1 |
| 49 | TC2_CHANNEL2 | X | X | 16-bit Timer Counter 2, Channel 2 |
| 50 | TC3_CHANNEL0 | X | X | 16-bit Timer Counter 3, Channel 0 |
| 51 | TC3_CHANNEL1 | X | X | 16-bit Timer Counter 3, Channel 1 |
| 52 | TC3_CHANNEL2 | X | X | 16-bit Timer Counter 3, Channel 2 |
| 53 | MLB | X | X | MediaLB IRQ 0 |
| 54 | MLB | X | – | MediaLB IRQ 1 |
| 55 | – | X | – | Reserved |
| 56 | AES | X | X | Advanced Encryption Standard |
| 57 | TRNG | X | X | True Random Number Generator |
| 58 | XDMAC | X | X | DMA Controller |
| 59 | ISI | X | X | Image Sensor Interface |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | | | | |
|---|---|---|---|---|
| **Instance ID** | **Instance Name** | **NVIC Interrupt** | **PMC Clock Control** | **Description** |
| 60 | PWM1 | X | X | Pulse Width Modulation Controller |
| 61 | ARM | FPU | – | Arm Floating Point Unit interrupt associated with OFC, UFC, IOC, DZC and IDC bits |
| 62 | SDRAMC | X | – | SDRAM Controller |
| 63 | RSWDT | X | – | Reinforced Safety Watchdog Timer |
| 64 | ARM | CCW | – | Arm Cache ECC Warning |
| 65 | ARM | CCF | – | Arm Cache ECC Fault |
| 66 | GMAC | Q1 | – | GMAC Queue 1 Interrupt signal toggled on a DMA write to the first word of each DMA data buffer associated with queue 1 |
| 67 | GMAC | Q2 | – | GMAC Queue 2 Interrupt signal toggled on a DMA write to the first word of each DMA data buffer associated with queue 2 |
| 68 | ARM | IXC | – | Floating Point Unit Interrupt IXC associated with FPU cumulative exception bit |
| 69 | I2SC0 | X | X | Inter-IC Sound Controller |
| 70 | I2SC1 | X | X | Inter-IC Sound Controller |
| 71 | GMAC | Q3 | – | GMAC Queue 3 Interrupt signal toggled on a DMA write to the first word of each DMA data buffer associated with queue 3 |
| 72 | GMAC | Q4 | – | GMAC Queue 4 Interrupt signal toggled on a DMA write to the first word of each DMA data buffer associated with queue 4 |
| 73 | GMAC | Q5 | – | GMAC Queue 5 Interrupt signal toggled on a DMA write to the first word of each DMA data buffer associated with queue 5 |

## 14.2 Peripheral Signal Multiplexing on I/O Lines

The SAMV71Q21RT features

- Two PIO controllers on 64-pin versions (PIOA and PIOB)
- Three PIO controllers on the 100-pin version (PIOA, PIOB and PIOD)
- Five PIO controllers on the 144-pin version (PIOA, PIOB, PIOC, PIOD and PIOE), that multiplex the I/O lines of the peripheral set.

The SAMV71Q21RT PIO Controllers control up to 32 lines and each line can be assigned to one of four peripheral functions: A, B, C or D.

For more information on multiplexed signals, refer to the Package and Pinout chapter.

# 15. ARM Cortex-M7 (ARM)

Refer to ARM reference documents *Cortex-M7 Processor User Guide* (ARM DUI 0644) and *Cortex-M7 Technical Reference Manual* (ARM DDI 0489), available on www.arm.com.

## 15.1 ARM Cortex-M7 Configuration

The following table provides the configuration for the ARM Cortex-M7 processor in SAMV71Q21RT devices.

**Table 15-1. ARM Cortex-M7 Configuration**

| Features | Configuration |
|---|---|
| **Debug** | |
| Comparator set | Full comparator set: 4 DWT and 8 FPB comparators |
| ETM support | Instruction ETM interface |
| Internal Trace support (ITM) | ITM and DWT trace functionality implemented |
| CTI and WIC | Not embedded |
| **TCM** | |
| ITCM max size | 128 KB |
| DTCM max size | 256 KB |
| **Cache** | |
| Cache size | 16 KB for instruction cache, 16 KB for data cache |
| Number of sets | 256 for instruction cache, 128 for data cache |
| Number of ways | 2 for instruction cache, 4 for data cache |
| Number of words per cache line | 8 words (32 bytes) |
| ECC on Cache | Embedded |
| **NVIC** | |
| IRQ number | 74 |
| IRQ priority levels | 8 |
| **MPU** | |
| Number of regions | 16 |
| **FPU** | |
| FPU precision | Single and double precision |
| **AHB Port** | |
| AHBP addressing size | 512 MB |

© 2021 Microchip Technology Inc.
and its subsidiaries

# 16. Debug and Test Features

## 16.1 Description

The device features a number of complementary debug and test capabilities. The Serial Wire Debug Port (SW-DP) is used for standard debugging functions, such as downloading code and single-stepping through programs. It also embeds a serial wire trace.

## 16.2 Embedded Characteristics

- Debug access to all memory and registers in the system, including Cortex-M register bank, when the core is running, halted, or held in reset.
- Serial Wire Debug Port (SW-DP) debug access
- Flash Patch and Breakpoint (FPB) unit for implementing breakpoints and code patches
- Data Watchpoint and Trace (DWT) unit for implementing watchpoints, data tracing, and system profiling
- Instrumentation Trace Macrocell (ITM) for support of printf style debugging
- 6-pin Embedded Trace Macrocell (ETM) for instruction trace stream, including CoreSight™ Trace Port Interface Unit (TPIU)
- IEEE1149.1 JTAG Boundary scan on All Digital Pins

## 16.3 Associated Documents

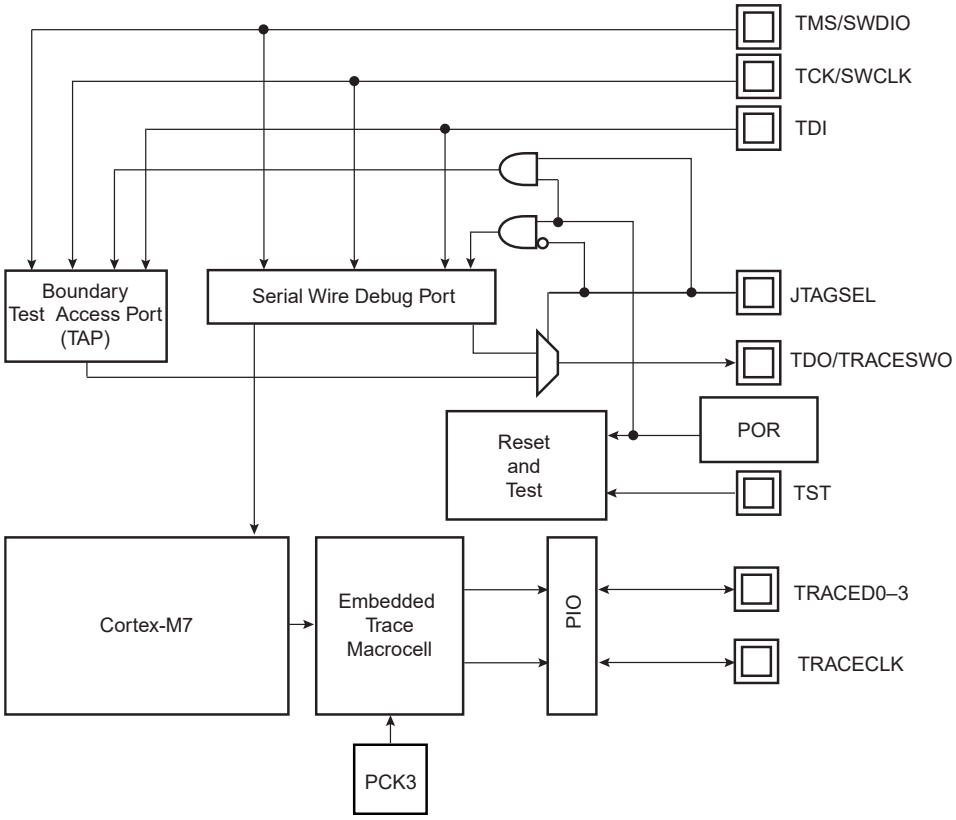
The SAMV71Q21RT implements the standard ARM CoreSight macrocell. For information on CoreSight, the following reference documents are available from the ARM web site (www.arm.com):

- Cortex-M7 User Guide Reference Manual (ARM DUI 0644)
- Cortex-M7 Technical Reference Manual (ARM DDI 0489)
- CoreSight Technology System Design Guide (ARM DGI 0012)
- CoreSight Components Technical Reference Manual (ARM DDI 0314)
- ARM Debug Interface v5 Architecture Specification (Doc. ARM IHI 0031)
- ARMv7-M Architecture Reference Manual (ARM DDI 0403)

© 2021 Microchip Technology Inc.
and its subsidiaries

## 16.4 Debug and Test Block Diagram

**Figure 16-1. Debug and Test Block Diagram**




## 16.5 Debug and Test Pin Description

**Table 16-1. Debug and Test Signal List**

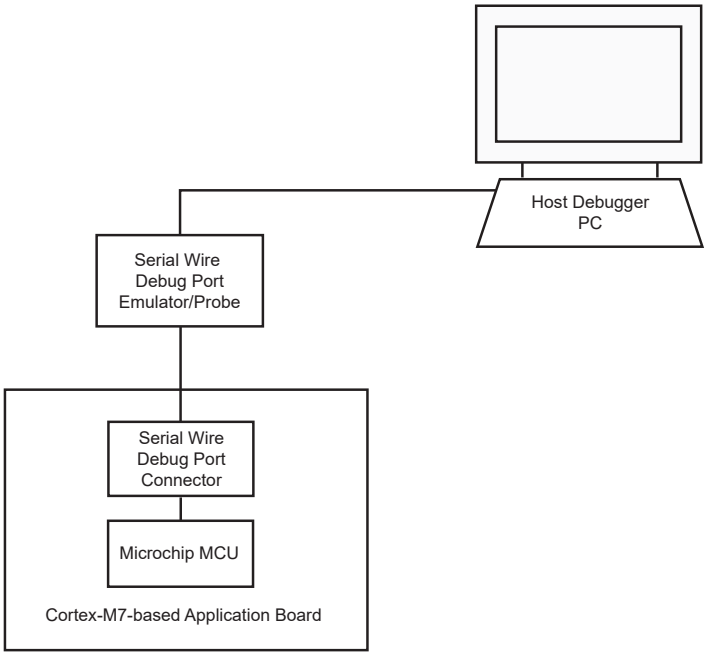
| Signal Name | Function | Type | Active Level |
|---|---|---|---|
| Reset/Test | | | |
| NRST | Microcontroller Reset | Input/Output | Low |
| TST | Test Select | Input | – |
| Serial Wire Debug Port/JTAG Boundary Scan | | | |
| TCK/SWCLK | Test Clock/Serial Wire Clock | Input | – |
| TDI | Test Data In | Input | – |
| TDO/TRACESWO | Test Data Out/Trace Asynchronous Data Out | Output | – |
| TMS/SWDIO | Test Mode Select/Serial Wire Input/Output | Input | – |
| JTAGSEL | JTAG Selection | Input | High |
| Trace Debug Port | | | |
| TRACECLK | Trace Clock | Output | – |
| TRACED0–3 | Trace Data | Output | – |

## 16.6 Application Examples

### 16.6.1 Debug Environment

The figure below shows a complete debug environment example. The SW-DP interface is used for standard debugging functions, such as downloading code and single-stepping through the program and viewing core and peripheral registers.

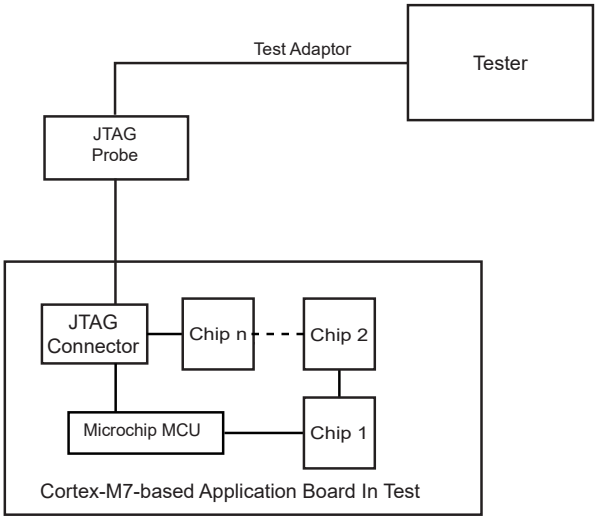
**Figure 16-2. Application Debug Environment Example**




### 16.6.2 Test Environment

The figure below shows a test environment example (JTAG Boundary scan). Test vectors are sent and interpreted by the tester. In this example, the “board in test” is designed using a number of JTAG-compliant devices. These devices can be connected to form a single scan chain.

**Figure 16-3. Application Test Environment Example**



## 16.7 Functional Description

### 16.7.1 Test Pin

The TST pin is used for JTAG Boundary Scan Manufacturing Test or Fast Flash Programming mode. The TST pin integrates a permanent pulldown resistor of about 15 kΩ to GND, so that it can be left unconnected for normal operations. To enable Fast Flash Programming mode, refer to 18. Fast Flash Programming Interface (FFPI).

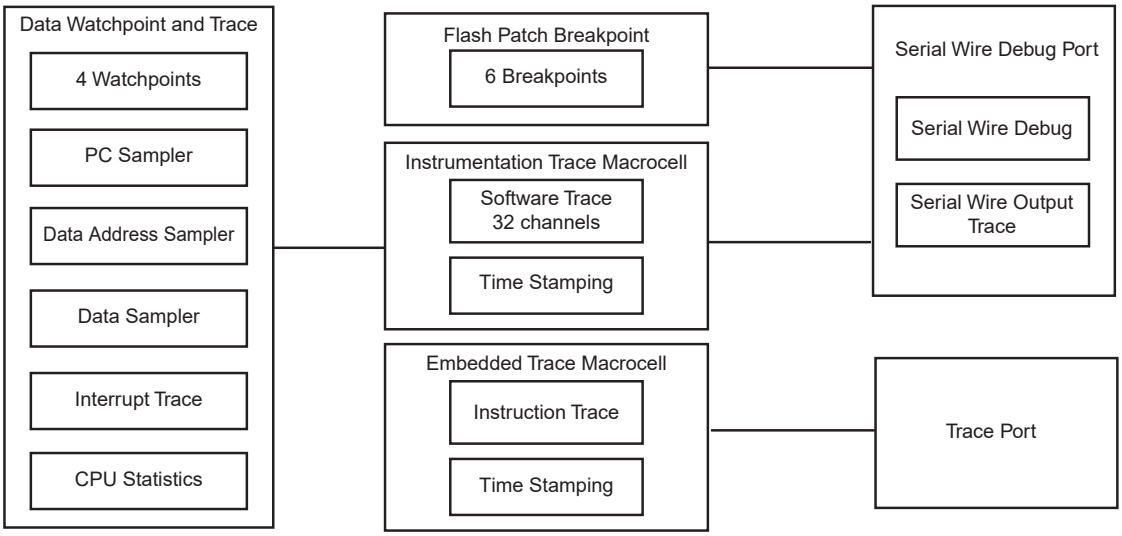
### 16.7.2 Debug Architecture

Figure 16-4 shows the debug architecture used. The Cortex-M7 embeds six functional units for debug:

- Serial Wire Debug Port (SW-DP) debug access
- FPB (Flash Patch Breakpoint)
- DWT (Data Watchpoint and Trace)
- ITM (Instrumentation Trace Macrocell)
- 6-pin Embedded Trace Macrocell (ETM) for instruction trace stream, including CoreSight Trace Port Interface Unit (TPIU)
- IEEE1149.1 JTAG Boundary scan on all digital pins

The debug architecture information that follows is mainly dedicated to developers of SW-DP Emulators/Probes and debugging tool vendors for Cortex-M7-based microcontrollers. For further details on SW-DP, see the Cortex - M7 Technical Reference Manual.

**Figure 16-4. Debug Architecture**




### 16.7.3 Serial Wire Debug Port (SW-DP) Pins

The SW-DP pins SWCLK and SWDIO are commonly provided on a standard 20-pin JTAG connector defined by ARM. For more details on voltage reference and reset state, refer to the "Signal Description" chapter.

At startup, SW-DP pins are configured in SW-DP mode to allow connection with debugging probe.

SW-DP pins can be used as standard I/Os to provide users more general input/output pins when the debug port is not needed in the end application. Mode selection between SW-DP mode (System I/O mode) and general I/O mode is performed through the AHB Matrix Chip Configuration registers (CCFG_SYSIO). Configuration of the pad for pullup, triggers, debouncing and glitch filters is possible regardless of the mode.

The JTAGSEL pin is used to select the JTAG boundary scan when asserted at a high level. It integrates a permanent pulldown resistor of about 15 kΩ to GND, so that it can be left unconnected for normal operations.

The JTAG debug ports TDI, TDO, TMS and TCK are inactive. They are provided for Boundary Scan Manufacturing Test purposes only. By default the SW-DP is active; TDO/TRACESWO can be used for trace.

**Table 16-2. SW-DP Pin List**

| Pin Name | JTAG Boundary Scan | Serial Wire Debug Port |
|---|---|---|
| TMS/SWDIO | TMS | SWDIO |
| TCK/SWCLK | TCK | SWCLK |
| TDI | TDI | – |
| TDO/TRACESWO | TDO | TRACESWO (optional: trace) |

SW-DP is selected when JTAGSEL is low. It is not possible to switch directly between SW-DP and JTAG boundary scan operations. A chip reset must be performed after JTAGSEL is changed.

### 16.7.4 Embedded Trace Module (ETM) Pins

The Embedded Trace Module (ETM) uses the Trace Port Interface Unit (TPIU) to export data out of the system.

The TPUI features the pins:

- TRACECLK–always exported to enable synchronization back with the data. PCK3 is used internally.
- TRACED0–3–the instruction trace stream.

### 16.7.5 Flash Patch Breakpoint (FPB)

The FPB implements hardware breakpoints.

### 16.7.6 Data Watchpoint and Trace (DWT)

The DWT contains four comparators which can be configured to generate:

- PC sampling packets at set intervals
- PC or Data watchpoint packets
- Watchpoint event to halt core

The DWT contains counters for:

- Clock cycle (CYCCNT)
- Folded instructions
- Load Store Unit (LSU) operations
- Sleep cycles
- CPI (all instruction cycles except for the first cycle)
- Interrupt overhead

### 16.7.7 Instrumentation Trace Macrocell (ITM)

The ITM is an application driven trace source that supports `printf` style debugging to trace Operating System (OS) and application events, and emits diagnostic system information. The ITM emits trace information as packets which can be generated by three different sources with several priority levels:

- Software trace: Software can write directly to ITM stimulus registers. This can be done using the `printf` function. For more information, refer to 16.7.5 Flash Patch Breakpoint (FPB).
- Hardware trace: The ITM emits packets generated by the DWT.
- Timestamping: Timestamps are emitted relative to packets. The ITM contains a 21-bit counter to generate the timestamp.

#### 16.7.7.1 How to Configure the ITM

The following example describes how to output trace data in asynchronous trace mode.

Configure the TPIU for asynchronous trace mode. Refer to 16.7.7.3 How to Configure the TPIU.

1. Enable the write accesses into the ITM registers by writing “0xC5ACCE55” into the Lock Access Register (Address: 0xE0000FB0)
2. Write 0x00010015 into the Trace Control register:

- Enable ITM.
- Enable Synchronization packets.
- Enable SWO behavior.
- Fix the ATB ID to 1.

3. Write 0x1 into the Trace Enable register:
   - Enable the Stimulus port 0.
4. Write 0x1 into the Trace Privilege register:
   - Stimulus port 0 only accessed in privileged mode (Clearing a bit in this register will result in the corresponding stimulus port being accessible in user mode.)
5. Write into the Stimulus port 0 register: TPIU (Trace Port Interface Unit)
   The TPIU acts as a bridge between the on-chip trace data and the Instruction Trace Macrocell (ITM).

   The TPIU formats and transmits trace data off-chip at frequencies asynchronous to the core.

#### 16.7.7.2 Asynchronous Mode

The TPIU is configured in asynchronous mode, trace data are output using the single TRACESWO pin. The TRACESWO signal is multiplexed with the TDO signal. As a consequence, asynchronous trace mode is only available when the Serial Wire Debug mode is selected.

Two encoding formats are available for the single pin output:

- Manchester encoded stream. This is the reset value.
- NRZ_based UART byte structure

#### 16.7.7.3 How to Configure the TPIU

This example only concerns the asynchronous trace mode.

Set the TRCENA bit to 1 into the Debug Exception and Monitor Register (0xE000EDFC) to enable the use of trace and debug blocks.

1. Write 0x2 into the Selected Pin Protocol Register.
   - Select the Serial Wire output – NRZ
2. Write 0x100 into the Formatter and Flush Control Register.
3. Set the suitable clock prescaler value into the Async Clock Prescaler Register to scale the baud rate of the asynchronous output (this can be done automatically by the debugging tool).

### 16.7.8 IEEE1149.1 JTAG Boundary Scan

IEEE1149.1 JTAG Boundary Scan allows pin-level access independent of the device packaging technology.

IEEE1149.1 JTAG Boundary Scan is enabled when TST is tied to high, PD0 tied to low, and JTAGSEL tied to high during powerup. These pins must be maintained in their respective states for the duration of the boundary scan operation. The SAMPLE, EXTEST and BYPASS functions are implemented. In Serial Wire Debug mode, the ARM processor responds with a non-JTAG chip ID that identifies the processor. This is not IEEE1149.1 JTAG-compliant.

It is not possible to switch directly between JTAG Boundary Scan and SWJ Debug Port operations. A chip reset must be performed after JTAGSEL is changed.

A Boundary Scan Descriptor Language (BSDL) file to set up the test is provided on www.microchip.com.

#### 16.7.8.1 JTAG Boundary Scan Register

The Boundary Scan Register (BSR) contains a number of bits which correspond to active pins and associated control signals.

Each input/output pin corresponds to a 3-bit register in the BSR. The OUTPUT bit contains data that can be forced on the pad. The INPUT bit facilitates the observability of data applied to the pad. The CONTROL bit selects the direction of the pad.

For more information, refer to BDSL files available on www.microchip.com.

### 16.7.9 ID Code Register

**Access:** Read-only

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| VERSION | | | | PART NUMBER | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PART NUMBER | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| PART NUMBER | | | | MANUFACTURER IDENTITY | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MANUFACTURER IDENTITY | | | | | | | 1 |

- **VERSION[31:28]: Product Version Number**
  Set to 0x0.
- **PART NUMBER[27:12]: Product Part Number**
  Set to 0x0.

| PART NUMBER |
|---|
| 0x5B3D |

- **MANUFACTURER IDENTITY[11:1]: Manufacturer ID**
  Set to 0x01F.
- **Bit[0]: Required by IEEE Std. 1149.1**
  Set to 0x1.

| JTAG ID Code |
|---|
| 0x5B3D_D03F |

# 17. SAM-BA Boot Program

## 17.1 Description

The SAM-BA Boot Program integrates an array of programs permitting download and/or upload into the different memories of the product.

## 17.2 Embedded Characteristics

- Default Boot Program
- Interface with SAM-BA Graphic User Interface
- SAM-BA Boot
  - Supports several communication media
    - Serial Communication on UART0
    - USB device port communication up to 1Mbyte/s
  - USB Requirements
    - Not supported

## 17.3 Hardware and Software Constraints

- SAM-BA Boot uses the first 2048 bytes of the SRAM for variables and stacks. The remaining available bytes can be used for user code.
- USB Requirements:
  - Not supported
- UART0 Requirements:
  - None. If no accurate external clock source is available, the internal 12 MHz RC meets RS-232 standards.

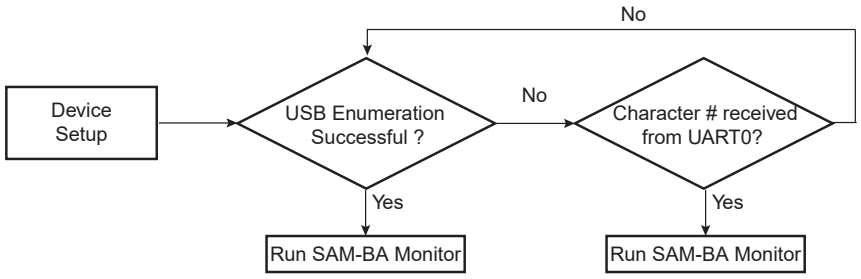
**Table 17-1. Pins Driven during Boot Program Execution**

| Peripheral | Pin | PIO Line |
|---|---|---|
| UART0 | URXD0 | PA9 |
| UART0 | UTXD0 | PA10 |

## 17.4 Flow Diagram

The boot program implements the algorithm below.

**Figure 17-1. Boot Program Algorithm Flow Diagram**




The SAM-BA boot program looks for a source clock, either from the embedded main oscillator with external crystal (main oscillator enabled) or from a supported frequency signal applied to the XIN pin (Main oscillator in bypass mode).

If a clock is supplied by one of the two sources, the boot program checks that the frequency is one of the supported external frequencies. If the frequency is supported, USB activation is allowed. If no clock is supplied, or if a clock is supplied but the frequency is not a supported external frequency, the internal 12 MHz RC oscillator is used as the main clock. In this case, the USB is not activated due to the frequency drift of the 12 MHz RC oscillator.

## 17.5 Device Initialization

Initialization by the boot program follows the steps described below:

Stack setup.

1. Embedded Flash Controller setup.
2. External clock (crystal or external clock on XIN) detection.
3. External crystal or clock with supported frequency supplied.
   a. If yes, USB activation is allowed.
   b. If no, USB activation is not allowed. The internal 12 MHz RC oscillator is used.
4. Master clock switch to main oscillator.
5. C variable initialization.
6. PLLA setup: PLLA is initialized to generate a 48 MHz clock.
7. Watchdog disable.
8. Initialization of UART0 (115200 bauds, 8, N, 1).
9. Initialization of the USB Device Port (only if USB activation is allowed; see Step 4.).
10. Wait for one of the following events:
    a. Check if USB device enumeration has occurred.
    b. Check if characters have been received in UART0.
11. Jump to SAM-BA Monitor (refer to 17.6 SAM-BA Monitor)

## 17.6 SAM-BA Monitor

Once the communication interface is identified, the monitor runs in an infinite loop, waiting for different commands, as shown in the following table.

**Table 17-2. Commands Available through the SAM-BA Boot**

| Command | Action | Arguments | Example |
|---|---|---|---|
| N | Set Normal mode | No argument | N# |
| T | Set Terminal mode | No argument | T# |
| O | Write a byte | Address, Value# | O200001,CA# |
| o | Read a byte | Address,# | o200001,# |
| H | Write a half word | Address, Value# | H200002,CAFE# |
| h | Read a half word | Address,# | h200002,# |
| W | Write a word | Address, Value# | W200000,CAFEDECA# |
| w | Read a word | Address,# | w200000,# |
| S | Send a file | Address,# | S200000,# |
| R | Receive a file | Address, NbOfBytes# | R200000,1234# |
| G | Go | Address# | G200200# |
| V | Display version | No argument | V# |

- Mode commands:
  - Normal mode configures SAM-BA Monitor to send/receive data in binary format
  - Terminal mode configures SAM-BA Monitor to send/receive data in ASCII format
- Write commands: Write a byte (O), a halfword (H) or a word (W) to the target
  - Address: Address in hexadecimal
  - Value: Byte, halfword or word to write in hexadecimal
- Read commands: Read a byte (o), a halfword (h) or a word (w) from the target
  - Address: Address in hexadecimal
  - Output: The byte, halfword or word read in hexadecimal
- Send a file (S): Send a file to a specified address
  - Address: Address in hexadecimal
    **Note:** There is a timeout on this command which is reached when the prompt '>' appears before the end of the command execution.
- Receive a file (R): Receive data into a file from a specified address
  - Address: Address in hexadecimal
  - NbOfBytes: Number of bytes in hexadecimal to receive
- Go (G): Jump to a specified address and execute the code
  - Address: Address to jump in hexadecimal
- Get Version (V): Return the SAM-BA boot version
  **Note:** In Terminal mode, when the requested command is performed, SAM-BA Monitor adds the following prompt sequence to its answer: <LF>+<CR>+'>'.

### 17.6.1 UART0 Serial Port

Communication is performed through the UART0 initialized to 115200 Baud, 8, n, 1.

The Send and Receive File commands use the Xmodem protocol to communicate. Any terminal performing this protocol can be used to send the application file to the target. The size of the binary file to send depends on the SRAM size embedded in the product. In all cases, the size of the binary file must be smaller than the SRAM size because the Xmodem protocol requires some SRAM memory to work. Refer to the "Hardware and Software Constraints" section.

### 17.6.2 Xmodem Protocol

The Xmodem protocol supported is the 128-byte length block. This protocol uses a two-character CRC-16 to guarantee detection of a maximum bit error.

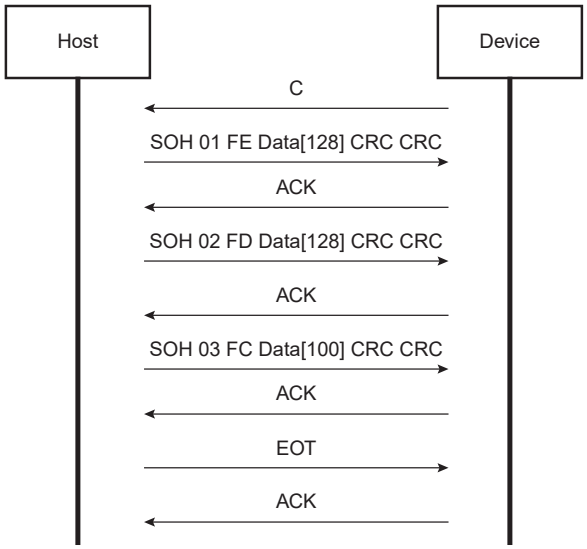
The Xmodem protocol with CRC is accurate if both sender and receiver report successful transmission. Each block of the transfer has the following format:

<SOH><blk #><255-blk #><--128 data bytes--><checksum> in which:

- <SOH> = 01 hex
- <blk #> = binary number, starts at 01, increments by 1, and wraps 0FFH to 00H (not to 01)
- <255-blk #> = 1's complement of the blk#.
- <checksum> = 2 bytes CRC16

The figure below shows a transmission using this protocol.

**Figure 17-2. Xmodem Transfer Example**




### 17.6.3 USB Device Port

The device uses the USB communication device class (CDC) drivers to take advantage of the installed PC RS-232 software to talk over the USB. The CDC class is implemented in all releases of Windows®, beginning with Windows 98SE. The CDC document, available at www.usb.org, describes a way to implement devices such as ISDN modems and virtual COM ports.

The Vendor ID (VID) is the Atmel vendor ID 0x03EB. The product ID (PID) is 0x6124. These references are used by the host operating system to mount the correct driver. On Windows systems, the INF files contain the correspondence between vendor ID and product ID.

For more details on VID/PID for end product/systems, refer to the Vendor ID form available from the USB Implementers Forum found at http://www.usb.org/.

**⚠WARNING** Unauthorized use of assigned or unassigned USB Vendor ID Numbers and associated Product ID Numbers is strictly prohibited.

#### 17.6.3.1 Enumeration Process

The USB protocol is a master/slave protocol. This is the host that starts the enumeration sending requests to the device through the control endpoint. The device handles standard requests as defined in the USB Specification.

**Table 17-3. Handled Standard Requests**

| Request | Definition |
|---|---|
| GET_DESCRIPTOR | Returns the current device configuration value. |
| SET_ADDRESS | Sets the device address for all future device access. |
| SET_CONFIGURATION | Sets the device configuration. |
| GET_CONFIGURATION | Returns the current device configuration value. |
| GET_STATUS | Returns status for the specified recipient. |
| SET_FEATURE | Set or Enable a specific feature. |
| CLEAR_FEATURE | Clear or Disable a specific feature. |

The device also handles some class requests defined in the CDC class.

**Table 17-4. Handled Class Requests**

| Request | Definition |
|---|---|
| SET_LINE_CODING | Configures DTE rate, stop bits, parity and number of character bits. |
| GET_LINE_CODING | Requests current DTE rate, stop bits, parity and number of character bits. |
| SET_CONTROL_LINE_STATE | RS-232 signal used to tell the DCE device the DTE device is now present. |

Unhandled requests are STALLed.

#### 17.6.3.2 Communication Endpoints

There are two communication endpoints. Endpoint 0 is used for the enumeration process. Endpoint 1 is a 64-byte Bulk OUT endpoint. Endpoint 2 is a 64-byte Bulk IN endpoint. SAM-BA Boot commands are sent by the host through endpoint 1. If required, the message is split by the host into several data payloads by the host driver.

If the command requires a response, the host can send IN transactions to pick up the response.

### 17.6.4 In Application Programming (IAP) Feature

The IAP feature is a function located in ROM that can be called by any software application.

When called, this function sends the desired FLASH command to the EEFC and waits for the Flash to be ready (looping while the FRDY bit is not set in the MC_FSR register).

Since this function is executed from ROM, this allows Flash programming (such as sector write) to be done by code running in Flash.

The IAP function entry point is retrieved by reading the NMI vector in ROM (0x00800008).

This function takes two arguments as parameters:

- the index of the Flash bank to be programmed: 0 for EEFC0, 1 for EEFC1. For devices with only one bank, this parameter has no effect and can be either 0 or 1, only EEFC0 will be accessed.
- the command to be sent to the EEFC Command register.

This function returns the value of the EEFC_FSR register.

An example of IAP software code follows:

```
// Example: How to write data in page 200 of the flash memory using ROM IAP function

flash_page_num = 200

flash_cmd = 0

flash_status = 0

eefc_index = 0 (0 for EEFC0, 1 for EEFC1)

// Initialize the function pointer (retrieve function address from NMI vector)*/

iap_function_address = 0x00800008

// Fill the Flash page buffer at address 200 with the data to be written

for i=0, i < page_size, i++ do

flash_sector_200_address[i] = your_data[i]

// Prepare the command to be sent to the EEFC Command register: key, page number and
write command

flash_cmd = (0x5A << 24) | (flash_page_num << 8) | flash_write_command;

// Call the IAP function with the right parameters and retrieve the status in
flash_status after completion

flash_status = iap_function (eefc_index, flash_cmd);
```

# 18. Fast Flash Programming Interface (FFPI)

## 18.1 Description

The Fast Flash Programming Interface (FFPI) provides parallel high-volume programming using a standard gang programmer. The parallel interface is fully handshaked and the device is considered to be a standard EEPROM. Additionally, the parallel protocol offers an optimized access to all the embedded Flash functionalities.

Although the Fast Flash Programming mode is a dedicated mode for high volume programming, this mode is not designed for in-situ programming.

## 18.2 Embedded Characteristics

- Programming Mode for High-volume Flash Programming Using Gang Programmer
  - Offers Read and Write Access to the Flash Memory Plane
  - Enables Control of Lock Bits and General-purpose NVM Bits
  - Enables Security Bit Activation
  - Disabled Once Security Bit is Set
- Parallel Fast Flash Programming Interface
  - Provides a 16-bit Parallel Interface to Program the Embedded Flash
  - Full Handshake Protocol

## 18.3 Parallel Fast Flash Programming

### 18.3.1 Device Configuration

In Fast Flash Programming mode, the device is in a specific test mode. Only a certain set of pins is significant. The rest of the PIOs are used as inputs with a pullup. The crystal oscillator is in Bypass mode. Other pins must be left unconnected.

**Figure 18-1. 16-bit Parallel Programming Interface**

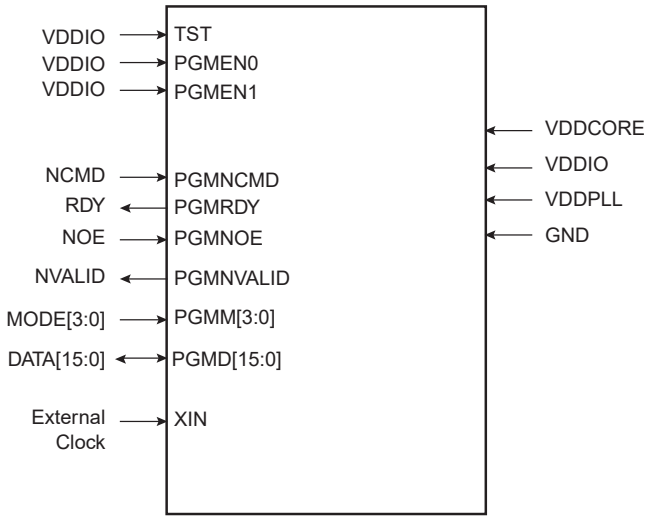



**Table 18-1. Signal Description List**

| Signal Name | Function | Type | Active Level | Comments |
|---|---|---|---|---|
| Power | | | | |

| .....continued | | | | |
|---|---|---|---|---|
| **Signal Name** | **Function** | **Type** | **Active Level** | **Comments** |
| VDDIO | I/O Lines Power Supply | Power | – | – |
| VDDCORE | Core Power Supply | Power | – | – |
| VDDPLL | PLL Power Supply | Power | – | – |
| GND | Ground | Ground | – | – |
| Clocks | | | | |
| XIN | Main Clock Input | Input | – | – |
| Test | | | | |
| TST | Test Mode Select | Input | High | Must be connected to VDDIO |
| PGMEN0 | Test Mode Select | Input | Low | Must be connected to GND |
| PGMEN1 | Test Mode Select | Input | High | Must be connected to VDDIO |
| PIO | | | | |
| PGMNCMD | Valid command available | Input | Low | Pulled-up input at reset |
| PGMRDY | 0: Device is busy<br>1: Device is ready for a new command | Output | High | Pulled-up input at reset |
| PGMNOE | Output Enable (active high) | Input | Low | Pulled-up input at reset |
| PGMNVALID | 0: DATA[15:0] is in input mode<br>1: DATA[15:0] is in output mode | Output | Low | Pulled-up input at reset |
| PGMM[3:0] | Specifies DATA type (see Table 18-2) | Input | – | Pulled-up input at reset |
| PGMD[15:0] | Bidirectional data bus | Input/Output | – | Pulled-up input at reset |

### 18.3.2 Signal Names

Depending on the MODE settings, DATA is latched in different internal registers.

**Table 18-2. Mode Coding**

| MODE[3:0] | Symbol | Data |
|---|---|---|
| 0000 | CMDE | Command Register |
| 0001 | ADDR0 | Address Register LSBs |
| 0010 | ADDR1 | – |
| 0011 | ADDR2 | – |
| 0100 | ADDR3 | Address Register MSBs |
| 0101 | DATA | Data Register |
| Default | IDLE | No register |

When MODE is equal to CMDE, then a new command (strobed on DATA[15:0] signals) is stored in the command register.

**Table 18-3. Command Bit Coding**

| DATA[15:0] | Symbol | Command Executed |
|---|---|---|
| 0x0011 | READ | Read Flash |

| .....continued | | |
|---|---|---|
| **DATA[15:0]** | **Symbol** | **Command Executed** |
| 0x0012 | WP | Write Page Flash |
| 0x0022 | WPL | Write Page and Lock Flash |
| 0x0032 | EWP | Erase Page and Write Page |
| 0x0042 | EWPL | Erase Page and Write Page then Lock |
| 0x0013 | EA | Erase All |
| 0x0014 | SLB | Set Lock Bit |
| 0x0024 | CLB | Clear Lock Bit |
| 0x0015 | GLB | Get Lock Bit |
| 0x0034 | SGPB | Set General Purpose NVM bit |
| 0x0044 | CGPB | Clear General Purpose NVM bit |
| 0x0025 | GGPB | Get General Purpose NVM bit |
| 0x0054 | SSE | Set Security Bit |
| 0x0035 | GSE | Get Security Bit |
| 0x001F | WRAM | Write Memory |
| 0x001E | GVE | Get Version |

### 18.3.3 Entering Parallel Programming Mode

The following algorithm puts the device in Parallel Programming mode:

1. Apply the supplies as described in table Signal Description List.
2. If an external clock is available, apply it to XIN within the VDDCORE POR reset time-out period, as defined in the section “Electrical Characteristics”.
3. Wait for the end of this reset period.
4. Start a read or write handshaking.

### 18.3.4 Programmer Handshaking

A handshake is defined for read and write operations. When the device is ready to start a new operation (RDY signal set), the programmer starts the handshake by clearing the NCMD signal. The handshaking is completed once the NCMD signal is high and RDY is high.

#### 18.3.4.1 Write Handshaking

For details on the write handshaking sequence, refer to the following figure and table.

**Figure 18-2. Parallel Programming Timing, Write Sequence**

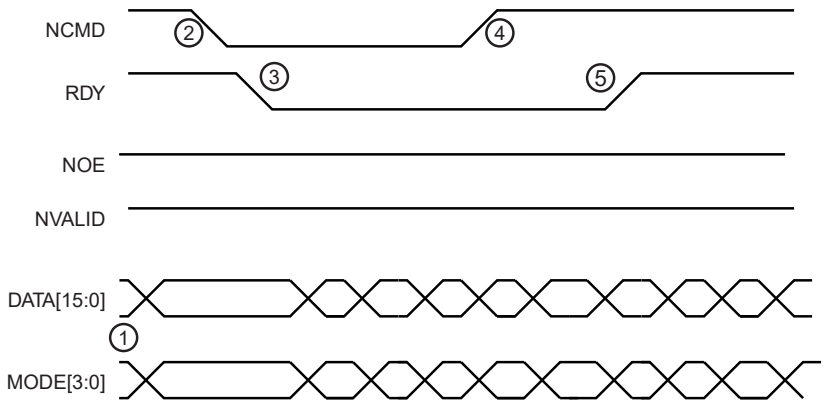



**Table 18-4. Write Handshake**

| Step | Programmer Action | Device Action | Data I/O |
|---|---|---|---|
| 1 | Sets MODE and DATA signals | Waits for NCMD low | Input |
| 2 | Clears NCMD signal | Latches MODE and DATA | Input |
| 3 | Waits for RDY low | Clears RDY signal | Input |
| 4 | Releases MODE and DATA signals | Executes command and polls NCMD high | Input |
| 5 | Sets NCMD signal | Executes command and polls NCMD high | Input |
| 6 | Waits for RDY high | Sets RDY | Input |

### 18.3.4.2 Read Handshaking

For details on the read handshaking sequence, refer to the following figure and table.

**Figure 18-3. Parallel Programming Timing, Read Sequence**

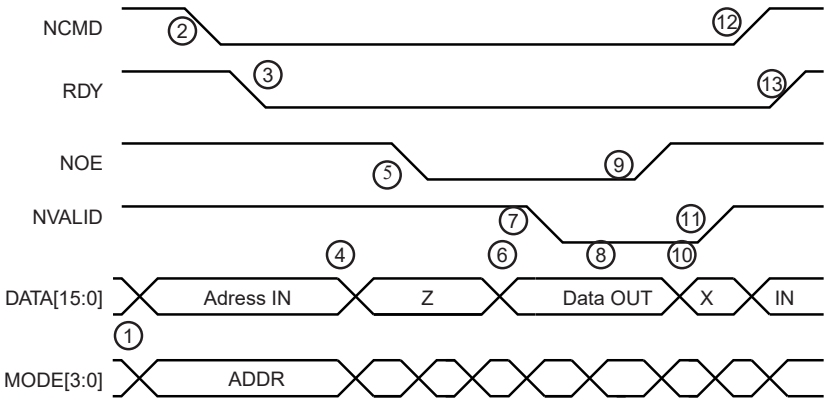



**Table 18-5. Read Handshake**

| Step | Programmer Action | Device Action | DATA I/O |
|---|---|---|---|
| 1 | Sets MODE and DATA signals | Waits for NCMD low | Input |
| 2 | Clears NCMD signal | Latch MODE and DATA | Input |
| 3 | Waits for RDY low | Clears RDY signal | Input |

| .....continued | | | |
|---|---|---|---|
| **Step** | **Programmer Action** | **Device Action** | **DATA I/O** |
| 4 | Sets DATA signal in tristate | Waits for NOE Low | Input |
| 5 | Clears NOE signal | – | Tristate |
| 6 | Waits for NVALID low | Sets DATA bus in output mode and outputs the flash contents. | Output |
| 7 | – | Clears NVALID signal | Output |
| 8 | Reads value on DATA Bus | Waits for NOE high | Output |
| 9 | Sets NOE signal | – | Output |
| 10 | Waits for NVALID high | Sets DATA bus in input mode | X |
| 11 | Sets DATA in output mode | Sets NVALID signal | Input |
| 12 | Sets NCMD signal | Waits for NCMD high | Input |
| 13 | Waits for RDY high | Sets RDY signal | Input |

### 18.3.5 Device Operations

Several commands on the Flash memory are available. These commands are summarized in table Command Bit Coding. Each command is driven by the programmer through the parallel interface running several read/write handshaking sequences.

When a new command is executed, the previous one is automatically achieved. Thus, chaining a read command after a write automatically flushes the load buffer in the Flash.

#### 18.3.5.1 Flash Read Command

This command is used to read the contents of the Flash memory. The read command can start at any valid address in the memory plane and is optimized for consecutive reads. Read handshaking can be chained; an internal address buffer is automatically increased.

**Table 18-6. Read Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | READ |
| 2 | Write handshaking | ADDR0 | Memory Address LSB |
| 3 | Write handshaking | ADDR1 | Memory Address |
| 4 | Read handshaking | DATA | *Memory Address++ |
| 5 | Read handshaking | DATA | *Memory Address++ |
| ... | ... | ... | ... |
| n | Write handshaking | ADDR0 | Memory Address LSB |
| n+1 | Write handshaking | ADDR1 | Memory Address |
| n+2 | Read handshaking | DATA | *Memory Address++ |
| n+3 | Read handshaking | DATA | *Memory Address++ |
| ... | ... | ... | ... |

#### 18.3.5.2 Flash Write Command

This command is used to write the Flash contents.

The Flash memory plane is organized into several pages. Data to be written are stored in a load buffer that corresponds to a Flash memory page. The load buffer is automatically flushed to the Flash:

- before access to any page other than the current one
- when a new command is validated (MODE = CMDE)

The Write Page command (WP) is optimized for consecutive writes. Write handshaking can be chained; an internal address buffer is automatically increased.

**Table 18-7. Write Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | WP or WPL or EWP or EWPL |
| 2 | Write handshaking | ADDR0 | Memory Address LSB |
| 3 | Write handshaking | ADDR1 | Memory Address |
| 4 | Write handshaking | DATA | *Memory Address++ |
| 5 | Write handshaking | DATA | *Memory Address++ |
| ... | ... | ... | ... |
| n | Write handshaking | ADDR0 | Memory Address LSB |
| n+1 | Write handshaking | ADDR1 | Memory Address |
| n+2 | Write handshaking | DATA | *Memory Address++ |
| n+3 | Write handshaking | DATA | *Memory Address++ |
| ... | ... | ... | ... |

The Flash command Write Page and Lock (WPL) is equivalent to the Flash Write Command. However, the lock bit is automatically set at the end of the Flash write operation. As a lock region is composed of several pages, the programmer writes to the first pages of the lock region using Flash write commands and writes to the last page of the lock region using a Flash write and lock command.

The Flash command Erase Page and Write (EWP) is equivalent to the Flash Write Command. However, before programming the load buffer, the page is erased.

The Flash command Erase Page and Write the Lock (EWPL) combines EWP and WPL commands.

#### 18.3.5.3 Flash Full Erase Command

This command is used to erase the Flash memory planes.

All lock regions must be unlocked before the Full Erase command by using the CLB command. Otherwise, the erase command is aborted and no page is erased.

**Table 18-8. Full Erase Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | EA |
| 2 | Write handshaking | DATA | 0 |

#### 18.3.5.4 Flash Lock Commands

Lock bits can be set using WPL or EWPL commands. They can also be set by using the Set Lock command (SLB). With this command, several lock bits can be activated. A Bit Mask is provided as argument to the command. When bit 0 of the bit mask is set, then the first lock bit is activated.

In the same way, the Clear Lock command (CLB) is used to clear lock bits.

**Table 18-9. Set and Clear Lock Bit Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | SLB or CLB |

| .....continued | | | |
|---|---|---|---|
| **Step** | **Handshake Sequence** | **MODE[3:0]** | **DATA[15:0]** |
| 2 | Write handshaking | DATA | Bit Mask |

Lock bits can be read using Get Lock Bit command (GLB). The $n^{th}$ lock bit is active when the bit n of the bit mask is set.

**Table 18-10. Get Lock Bit Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | GLB |
| 2 | Read handshaking | DATA | Lock Bit Mask Status<br>0 = Lock bit is cleared<br>1 = Lock bit is set |

#### 18.3.5.5 Flash General-purpose NVM Commands

General-purpose NVM bits (GP NVM bits) can be set using the Set GPNVM command (SGPB). This command also activates GP NVM bits. A bit mask is provided as argument to the command. When bit 0 of the bit mask is set, then the first GP NVM bit is activated.

In the same way, the Clear GPNVM command (CGPB) is used to clear general-purpose NVM bits. The general-purpose NVM bit is deactivated when the corresponding bit in the pattern value is set to 1.

**Table 18-11. Set/Clear GP NVM Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | SGPB or CGPB |
| 2 | Write handshaking | DATA | GP NVM bit pattern value |

General-purpose NVM bits can be read using the Get GPNVM Bit command (GGPB). The $n^{th}$ GP NVM bit is active when bit n of the bit mask is set.

**Table 18-12. Get GP NVM Bit Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | GGPB |
| 2 | Read handshaking | DATA | GP NVM Bit Mask Status<br>0 = GP NVM bit is cleared<br>1 = GP NVM bit is set |

#### 18.3.5.6 Flash Security Bit Command

A security bit can be set using the Set Security Bit command (SSE). Once the security bit is active, the Fast Flash programming is disabled. No other command can be run. An event on the Erase signal can erase the security bit once the contents of the Flash have been erased.

**Table 18-13. Set Security Bit Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | SSE |
| 2 | Write handshaking | DATA | 0 |

Once the security bit is set, it is not possible to access FFPI. The only way to erase the security bit is to erase the Flash.

To erase the Flash, perform the following steps:

1. Power off the chip.
2. Power on the chip with TST = 0.
3. Assert the ERASE signal for at least the ERASE pin assertion time as defined in the section “Electrical Characteristics”.
4. Power off the chip.

Return to FFPI mode to check that the Flash is erased.

#### 18.3.5.7 Memory Write Command

This command is used to perform a write access to any memory location.

The Memory Write command (WRAM) is optimized for consecutive writes. Write handshaking can be chained; an internal address buffer is automatically increased.

**Table 18-14. Write Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | WRAM |
| 2 | Write handshaking | ADDR0 | Memory Address LSB |
| 3 | Write handshaking | ADDR1 | Memory Address |
| 4 | Write handshaking | DATA | *Memory Address++ |
| 5 | Write handshaking | DATA | *Memory Address++ |
| ... | ... | ... | ... |
| n | Write handshaking | ADDR0 | Memory Address LSB |
| n+1 | Write handshaking | ADDR1 | Memory Address |
| n+2 | Write handshaking | DATA | *Memory Address++ |
| n+3 | Write handshaking | DATA | *Memory Address++ |
| ... | ... | ... | ... |

#### 18.3.5.8 Get Version Command

The Get Version (GVE) command retrieves the version of the FFPI interface.

**Table 18-15. Get Version Command**

| Step | Handshake Sequence | MODE[3:0] | DATA[15:0] |
|---|---|---|---|
| 1 | Write handshaking | CMDE | GVE |
| 2 | Read handshaking | DATA | Version |

# 19. Bus Matrix (MATRIX)

## 19.1 Description

The Bus Matrix (MATRIX) implements a multi-layer AHB, based on the AHB-Lite protocol, that enables parallel access paths between multiple AHB masters and slaves in a system, thus increasing the overall bandwidth. The MATRIX interconnects 13 AHB masters to 9 AHB slaves. The normal latency to connect a master to a slave is one cycle. The exception is the default master of the accessed slave which is connected directly (zero cycle latency).

The MATRIX user interface is compliant with ARM Advanced Peripheral Bus.

## 19.2 Embedded Characteristics

- 13 Masters
- 9 Slaves
- One Decoder for Each Master
- Several Possible Boot Memories for Each Master before Remap
- One Remap Function for Each Master
- Support for Long Bursts of 32, 64, 128 and up to the 256-beat Word Burst AHB Limit
- Enhanced Programmable Mixed Arbitration for Each Slave
  - Round-Robin
  - Fixed Priority
- Programmable Default Master for Each Slave
  - No Default Master
  - Last Accessed Default Master
  - Fixed Default Master
- Deterministic Maximum Access Latency for Masters
- Zero or One Cycle Arbitration Latency for the First Access of a Burst
- Bus Lock Forwarding to Slaves
- Master Number Forwarding to Slaves
- Configurable Automatic Clock-off Mode for Power Reduction
- One Special Function Register for Each Slave (not dedicated)
- Register Write Protection

### 19.2.1 Matrix Masters

The MATRIX manages the masters listed in he following table. Each master can perform an access to an available slave concurrently with other masters. lists the available masters.

Each master has its own specifically-defined decoder. To simplify addressing, all the masters have the same decodings.

**Table 19-1. Bus Matrix Masters**

| Master Index | Name |
|---|---|
| 0 | Cortex-M7 |
| 1 | Cortex-M7 |
| 2 | Cortex-M7 Peripheral Port |
| 3 | Integrated Check Monitor |
| 4, 5 | XDMAC |

| .....continued | |
|---|---|
| **Master Index** | **Name** |
| 6 | ISI DMA |
| 7 | Media LB |
| 8 | USB DMA |
| 9 | Ethernet MAC DMA |
| 10 | CAN0 DMA |
| 11 | CAN1 DMA |
| 12 | Cortex-M7 |

**Note:** Master 12 (Cortex-M7) is only on revision B.

### 19.2.2 Matrix Slaves

The MATRIX manages the slaves listed in the following table. Each slave has its own arbiter, providing a different arbitration per slave.

**Table 19-2. Bus Matrix Slaves**

| Slave Index | Name |
|---|---|
| 0 | Internal SRAM |
| 1 | Internal SRAM |
| 2 | Internal ROM |
| 3 | Internal Flash |
| 4 | USB High Speed Dual Port RAM (DPR) |
| 5 | External Bus Interface |
| 6 | QSPI |
| 7 | Peripheral Bridge |
| 8 | AHB Slave |

### 19.2.3 Master to Slave Access

The following table provides valid paths for master to slave accesses. The paths shown as “-” are forbidden or not wired.

**Table 19-3. Master to Slave Access**

| Masters | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Slaves | | Cortex-M7 | Cortex-M7 | Cortex-M7 Peripheral Port | ICM | Central DMA IF0 | Central DMA IF1 | ISI DMA | MediaLB DMA | USB DMA | GMAC DMA | CAN0 DMA | CAN1 DMA | Cortex-M7 |
| 0 | Internal SRAM | – | – | – | X | X | – | – | – | – | – | – | – | – |
| 1 | Internal SRAM | – | – | – | – | – | X | X | X | X | X | X | X | – |
| 2 | Internal ROM | X | – | – | – | – | – | – | – | – | – | – | – | – |
| 3 | Internal Flash | X | – | – | X | – | X | – | – | X | X | – | – | – |

.....continued

| Masters | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | USB HS Dual Port RAM | – | X | – | – | – | – | – | – | – | – | – | – | – |
| 5 | External Bus Interface | – | X | – | X | X | X | X | X | X | X | X | X | – |
| 6 | QSPI | – | – | – | X | – | X | – | – | X | X | – | – | X |
| 7 | Peripheral Bridge | – | X | X | – | – | X | – | – | – | – | – | – | – |
| 8 | Cortex-M7 AHB Slave (AHBS) (see **Note**) | – | – | – | X | X | – | X | X | X | X | X | X | – |

**Note:** For the connection of the Cortex-M7 processor to the SRAM, refer to the sections “Interconnect” and “Memories”, sub-section “Embedded Memories”.

**Related Links**

11.1 Embedded Memories

## 19.3 Functional Description

### 19.3.1 Memory Mapping

The MATRIX provides one decoder for every AHB master interface. The decoder offers each AHB master several memory mappings. Each memory area may be assigned to several slaves. Thus booting at the same address while using different AHB slaves (i.e., external RAM, internal ROM or internal Flash, etc.) is possible.

The MATRIX user interface provides the Master Remap Control Register (MATRIX_MRCR) that performs remap action for every master independently.

### 19.3.2 Special Bus Granting Mechanism

The MATRIX provides some speculative bus granting techniques in order to anticipate access requests from masters. This technique reduces latency at the first access of a burst, or for a single transfer, as long as the slave is free from any other master access. Bus granting sets a different default master for every slave.

At the end of the current access, if no other request is pending, the slave remains connected to its associated default master. A slave can be associated with three kinds of default masters:

- No default master
- Last access master
- Fixed default master

To change from one type of default master to another, the MATRIX user interface provides the Slave Configuration registers, one for every slave, that set a default master for each slave. The Slave Configuration register contains the fields DEFMSTR_TYPE and FIXED_DEFMSTR. The 2-bit DEFMSTR_TYPE field selects the default master type (no default, last access master, fixed default master), whereas the 4-bit FIXED_DEFMSTR field selects a fixed default master provided that DEFMSTR_TYPE is set to fixed default master. Please refer to the "Bus Matrix Slave Configuration Registers" section.

#### 19.3.2.1 No Default Master

After the end of the current access, if no other request is pending, the slave is disconnected from all masters.

This configuration incurs one latency clock cycle for the first access of a burst after bus Idle. Arbitration without default master may be used for masters that perform significant bursts or several transfers with no Idle in between, or if the slave bus bandwidth is widely used by one or more masters.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput whatever the number of requesting masters.

#### 19.3.2.2 Last Access Master

After the end of the current access, if no other request is pending, the slave remains connected to the last master that performed an access request.

This allows the MATRIX to remove the one latency cycle for the last master that accessed the slave. Other non privileged masters still get one latency clock cycle if they want to access the same slave. This technique is useful for masters that mainly perform single accesses or short bursts with some Idle cycles in between.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput whatever is the number of requesting masters.

#### 19.3.2.3 Fixed Default Master

At the end of the current access, if no other request is pending, the slave connects to its fixed default master. Unlike the last access master, the fixed default master does not change unless the user modifies it by software (FIXED_DEFMSTR field of the related MATRIX_SCFG).

This allows the MATRIX arbiters to remove the one latency clock cycle for the fixed default master of the slave. All requests attempted by the fixed default master do not cause any arbitration latency, whereas other non-privileged masters will get one latency cycle. This technique is useful for a master that mainly performs single accesses or short bursts with Idle cycles in between.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput, regardless of the number of requesting masters.

### 19.3.3 Arbitration

The MATRIX provides an arbitration technique that reduces latency when conflicting cases occur; for example. when two or more masters try to access the same slave at the same time. One arbiter per AHB slave is provided, so that each slave is arbitrated differently.

The MATRIX provides the user with two arbitration types for each slave:

1. Round-robin Arbitration (default)
2. Fixed Priority Arbitration

Each algorithm may be complemented by selecting a default master configuration for each slave.

When re-arbitration is required, specific conditions apply. Refer to the "Arbitration Rules" section.

#### 19.3.3.1 Arbitration Rules

Each arbiter has the ability to arbitrate between requests from two or more masters. To avoid burst breaking and to provide maximum throughput for slave interfaces, arbitration should take place during the following cycles:

- Idle cycles: When a slave is not connected to any master or is connected to a master which is not currently accessing it
- Single cycles: When a slave is performing a single access
- End of Burst cycles: When the current cycle is the last cycle of a burst transfer. For a defined length burst, predicted end of burst matches the size of the transfer but is managed differently for undefined length burst. Refer to the "Undefined Length Burst Arbitration" section.
- Slot cycle limit: When the slot cycle counter has reached the limit value indicating that the current master access is too long and must be broken. Refer to the "Slot Cycle Limit Arbitration" section.

##### 19.3.3.1.1 Undefined Length Burst Arbitration

In order to prevent slave handling during undefined length bursts, the user can trigger the re-arbitration before the end of the incremental bursts. The re-arbitration period can be selected from the following Undefined Length Burst Type (ULBT) possibilities:

1. Unlimited: no predetermined end of burst is generated. This value enables 1-Kbyte burst lengths.
2. 1-beat bursts: predetermined end of burst is generated at each single transfer during the INCR transfer.
3. 4-beat bursts: predetermined end of burst is generated at the end of each 4-beat boundary during INCR transfer.
4. 8-beat bursts: predetermined end of burst is generated at the end of each 8-beat boundary during INCR transfer.

© 2021 Microchip Technology Inc. and its subsidiaries

5. 16-beat bursts: predetermined end of burst is generated at the end of each 16-beat boundary during INCR transfer.
6. 32-beat bursts: predetermined end of burst is generated at the end of each 32-beat boundary during INCR transfer.
7. 64-beat bursts: predetermined end of burst is generated at the end of each 64-beat boundary during INCR transfer.
8. 128-beat bursts: predetermined end of burst is generated at the end of each 128-beat boundary during INCR transfer.

The use of undefined length16-beat bursts, or less, is discouraged since this decreases the overall bus bandwidth due to arbitration and slave latencies at each first access of a burst.

If the master does not permanently and continuously request the same slave or has an intrinsically limited average throughput, the ULBT should be left at its default unlimited value, knowing that the AHB specification natively limits all word bursts to 256 beats and double-word bursts to 128 beats because of its 1-Kbyte address boundaries.

Unless duly needed, the ULBT should be left at its default value of 0 for power saving.

This selection is made through the ULBT field of the Master Configuration Registers (MATRIX_MCFG).

#### 19.3.3.1.2 Slot Cycle Limit Arbitration

The MATRIX contains specific logic to break long accesses, such as very long bursts on a very slow slave (e.g., an external low speed memory). At each arbitration time, a counter is loaded with the value previously written in the SLOT_CYCLE field of the related Slave Configuration Register (MATRIX_SCFG) and decreased at each clock cycle. When the counter elapses, the arbiter has the ability to rearbitrate at the end of the current AHB bus access cycle.

Unless a master has a very tight access latency constraint, which could lead to data overflow or underflow due to a badly undersized internal FIFO with respect to its throughput, the Slot Cycle Limit should be disabled (SLOT_CYCLE = 0) or set to its default maximum value in order not to inefficiently break long bursts performed by some bus masters.

In most cases, this feature is not needed and should be disabled for power saving.

**⚠WARNING** This feature does not prevent a slave from locking its access indefinitely.

### 19.3.3.2 Arbitration Priority Scheme

The MATRIX arbitration scheme is organized in priority pools.

Round-robin priority is used in the highest and lowest priority pools, whereas fixed level priority is used between priority pools and in the intermediate priority pools.

For each slave, each master is assigned to one of the slave priority pools through the priority registers for slaves (MxPR fields of MATRIX_PRAS and MATRIX_PRBS). When evaluating master requests, this programmed priority level always takes precedence.

After reset, all the masters except those of the Cortex-M7 belong to the lowest priority pool (MxPR = 0) and are therefore granted bus access in a true round-robin order.

The highest priority pool must be specifically reserved for masters requiring very low access latency. If more than one master belongs to this pool, they will be granted bus access in a biased round-robin manner which allows tight and deterministic maximum access latency from AHB bus requests. In the worst case, any currently occurring high-priority master request will be granted after the current bus master access has ended and other high priority pool master requests, if any, have been granted once each.

The lowest priority pool shares the remaining bus bandwidth between AHB Masters.

Intermediate priority pools allow fine priority tuning. Typically, a moderately latency-critical master or a bandwidth-only critical master will use such a priority level. The higher the priority level (MxPR value), the higher the master priority.

All combinations of MxPR values are allowed for all masters and slaves. For example, some masters might be assigned the highest priority pool (round-robin), and remaining masters the lowest priority pool (round-robin), with no master for intermediate fix priority levels.

If more than one master requests the slave bus, regardless of the respective masters priorities, no master will be granted the slave bus for two consecutive runs. A master can only get back-to-back grants so long as it is the only requesting master.

#### 19.3.3.2.1 Fixed Priority Arbitration

The fixed priority arbitration algorithm is the first and only arbitration algorithm applied between masters from distinct priority pools. It is also used in priority pools other than the highest and lowest priority pools (intermediate priority pools).

Fixed priority arbitration is used by the MATRIX arbiters to dispatch the requests from different masters to the same slave by using the fixed priority defined by the user. If requests from two or more masters are active at the same time, the master with the highest priority number is serviced first. If requests from two or more masters with the same priority are active at the same time, the master with the highest number is serviced first.

For each slave, the priority of each master is defined in the MxPR field in the Priority Registers, MATRIX_PRAS and MATRIX_PRBS.

#### 19.3.3.2.2 Round-Robin Arbitration

Round-robin arbitration is only used in the highest and lowest priority pools. It allows the MATRIX arbiters to properly dispatch requests from different masters to the same slave. If two or more master requests are active at the same time in the priority pool, they are serviced in a round-robin increasing master number order.

### 19.3.4 System I/O Configuration

The System I/O Configuration register (CCFG_SYSIO) configures I/O lines in System I/O mode (such as JTAG, ERASE, USB, etc.) or as general purpose I/O lines. Enabling or disabling the corresponding I/O lines in peripheral mode or in PIO mode (PIO_PER or PIO_PDR registers) in the PIO controller as no effect. However, the direction (input or output), pull-up, pull-down and other mode control is still managed by the PIO controller.

### 19.3.5 SMC NAND Flash Chip Select Configuration

The SMC Nand Flash Chip Select Configuration Register (CCFG_SMCNFCS) manages the chip select signal (NCSx) and its assignment to NAND Flash.

Each NCSx may or may not be individually assigned to NAND Flash. When the NCSx is assigned to NAND Flash, the signals NANDOE and NANDWE are used for the NCSx signals selected.

### 19.3.6 Configuration of Automatic Clock-off Mode

To reduce power consumption, MATRIX, Bridge and EFC automatic clock gating can be enabled by writing a ‘1’ to bits MATCKG, BRIDCKG and EFCCKG, respectively, in the Dynamic Clock Gating register (CCFG_DYNCKG).

### 19.3.7 Register Write Protection

To prevent any single software error from corrupting MATRIX behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the Write Protection Mode Register (MATRIX_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the Write Protection Status Register (MATRIX_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS flag is reset by writing the Bus Matrix Write Protect Mode Register (MATRIX_WPMR) with the appropriate access key WPKEY.

The following registers can be write-protected:

- Bus Matrix Master Configuration Registers
- Bus Matrix Slave Configuration Registers
- Bus Matrix Priority Registers A For Slaves
- Bus Matrix Priority Registers B For Slaves
- Bus Matrix Master Remap Control Register

## 19.4 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | MATRIX_MCFG0 | 7:0 | | | | | | ULBT[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| ... | | | | | | | | | | |
| 0x30 | MATRIX_MCFG12 | 7:0 | | | | | | ULBT[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | MATRIX_SCFG0 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x44 | MATRIX_SCFG1 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x48 | MATRIX_SCFG2 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x4C | MATRIX_SCFG3 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x50 | MATRIX_SCFG4 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x54 | MATRIX_SCFG5 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x58 | MATRIX_SCFG6 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x5C | MATRIX_SCFG7 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x60 | MATRIX_SCFG8 | 7:0 | SLOT_CYCLE[6:0] | | | | | | | |
| | | 15:8 | | | | | | | SLOT_CYCLE[8:7] | |
| | | 23:16 | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x64 ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | MATRIX_PRAS0 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x84 | MATRIX_PRBS0 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x88 | MATRIX_PRAS1 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0x8C | MATRIX_PRBS1 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x90 | MATRIX_PRAS2 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0x94 | MATRIX_PRBS2 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x98 | MATRIX_PRAS3 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0x9C | MATRIX_PRBS3 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0xA0 | MATRIX_PRAS4 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xA4 | MATRIX_PRBS4 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0xA8 | MATRIX_PRAS5 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xAC | MATRIX_PRBS5 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0xB0 | MATRIX_PRAS6 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xB4 | MATRIX_PRBS6 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0xB8 | MATRIX_PRAS7 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xBC | MATRIX_PRBS7 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0xC0 | MATRIX_PRAS8 | 7:0 | | | M1PR[1:0] | | | | M0PR[1:0] | |
| | | 15:8 | | | M3PR[1:0] | | | | M2PR[1:0] | |
| | | 23:16 | | | M5PR[1:0] | | | | M4PR[1:0] | |
| | | 31:24 | | | M7PR[1:0] | | | | M6PR[1:0] | |
| 0xC4 | MATRIX_PRBS8 | 7:0 | | | M9PR[1:0] | | | | M8PR[1:0] | |
| | | 15:8 | | | M11PR[1:0] | | | | M10PR[1:0] | |
| | | 23:16 | | | | | | | M12PR[1:0] | |
| | | 31:24 | | | | | | | | |
| 0xC8 ... 0xFF | Reserved | | | | | | | | | |
| 0x0100 | MATRIX_MRCR | 7:0 | RCB7 | RCB6 | RCB5 | RCB4 | RCB3 | RCB2 | RCB1 | RCB0 |
| | | 15:8 | | | | RCB12 | RCB11 | RCB10 | RCB9 | RCB8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0104 ... 0x010F | Reserved | | | | | | | | | |
| 0x0110 | CCFG_CAN0 | 7:0 | Reserved[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | Reserved[8] |
| | | 23:16 | CAN0DMABA[7:0] | | | | | | | |
| | | 31:24 | CAN0DMABA[15:8] | | | | | | | |
| 0x0114 | CCFG_SYSIO | 7:0 | SYSIO7 | SYSIO6 | SYSIO5 | SYSIO4 | | | | |
| | | 15:8 | | | | SYSIO12 | | | | |
| | | 23:16 | CAN1DMABA[7:0] | | | | | | | |
| | | 31:24 | CAN1DMABA[15:8] | | | | | | | |
| 0x0118 | CCFG_PCCR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | I2SC1CC | I2SC0CC | TC0CC | | | | |
| | | 31:24 | | | | | | | | |
| 0x011C | CCFG_DYNCKG | 7:0 | | | | | | EFCCKG | BRIDCKG | MATCKG |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0120 ... 0x0123 | Reserved | | | | | | | | | |
| 0x0124 | CCFG_SMCNFCS | 7:0 | | | | SDRAMEN | SMC_NFCS3 | SMC_NFCS2 | SMC_NFCS1 | SMC_NFCS0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0128 ... 0x01E3 | Reserved | | | | | | | | | |
| 0x01E4 | MATRIX_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0x01E8 | MATRIX_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 19.4.1 Bus Matrix Master Configuration Registers

**Name:** MATRIX_MCFGx
**Offset:** 0x00 + x*0x04 [x=0..12]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | ULBT[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – ULBT[2:0]** Undefined Length Burst Type

| Value | Name | Description |
|---|---|---|
| 0 | UNLTD_LENGTH | Unlimited Length Burst—No predicted end of burst is generated, therefore INCR bursts coming from this master can only be broken if the Slave Slot Cycle Limit is reached. If the Slot Cycle Limit is not reached, the burst is normally completed by the master, at the latest, on the next AHB 1-Kbyte address boundary, allowing up to 256-beat word bursts or 128-beat double-word bursts.<br>This value should not be used in the very particular case of a master capable of performing back-to-back undefined length bursts on a single slave, since this could indefinitely freeze the slave arbitration and thus prevent another master from accessing this slave. |
| 1 | SINGLE_ACCESS | Single Access—The undefined length burst is treated as a succession of single accesses, allowing re-arbitration at each beat of the INCR burst or bursts sequence. |
| 2 | 4BEAT_BURST | 4-beat Burst—The undefined length burst or bursts sequence is split into 4-beat bursts or less, allowing re-arbitration every 4 beats. |
| 3 | 8BEAT_BURST | 8-beat Burst—The undefined length burst or bursts sequence is split into 8-beat bursts or less, allowing re-arbitration every 8 beats. |
| 4 | 16BEAT_BURST | 16-beat Burst—The undefined length burst or bursts sequence is split into 16-beat bursts or less, allowing re-arbitration every 16 beats. |
| 5 | 32BEAT_BURST | 32-beat Burst —The undefined length burst or bursts sequence is split into 32-beat bursts or less, allowing re-arbitration every 32 beats. |
| 6 | 64BEAT_BURST | 64-beat Burst—The undefined length burst or bursts sequence is split into 64-beat bursts or less, allowing re-arbitration every 64 beats. |

| Value | Name | Description |
|---|---|---|
| 7 | 128BEAT_BURST | 128-beat Burst—The undefined length burst or bursts sequence is split into 128-beat bursts or less, allowing re-arbitration every 128 beats.<br>**Note:** Unless duly needed, the ULBT should be left at its default 0 value for power saving. |

### 19.4.2 Bus Matrix Slave Configuration Registers

**Name:** MATRIX_SCFGx
**Offset:** 0x40 + x*0x04 [x=0..8]
**Reset:** 0x000001FF
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | FIXED_DEFMSTR[3:0] | | | | DEFMSTR_TYPE[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SLOT_CYCLE[8:7] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SLOT_CYCLE[6:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |

**Bits 21:18 – FIXED_DEFMSTR[3:0]** Fixed Default Master
Number of the Default Master for this slave. Only used if DEFMSTR_TYPE is 2. Specifying the number of a master which is not connected to the selected slave is equivalent to setting DEFMSTR_TYPE to 0.

**Bits 17:16 – DEFMSTR_TYPE[1:0]** Default Master Type

| Value | Name | Description |
|---|---|---|
| 0 | NONE | No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters.<br>This results in a one clock cycle latency for the first access of a burst transfer or for a single access. |
| 1 | LAST | Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it.<br>This results in not having one clock cycle latency when the last master tries to access the slave again. |
| 2 | FIXED | Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field.<br>This results in not having one clock cycle latency when the fixed master tries to access the slave again. |

**Bits 9:1 – SLOT_CYCLE[8:0]** Maximum Bus Grant Duration for Masters
When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.
If SLOT_CYCLE = 0, the slot cycle limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.
This limit must not be too small. Unreasonably small values break every burst and the MATRIX arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.
In most cases, this feature is not needed and should be disabled for power saving.
See “Slot Cycle Limit Arbitration” for details.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 19.4.3 Bus Matrix Priority Registers A For Slaves

**Name:** MATRIX_PRASx
**Offset:** 0x80 + x*0x08 [x=0..8]
**Reset:** 0x00000222
**Property:** Read/Write

This register can only be written if the WPE bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | M7PR[1:0] | | | | M6PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | M5PR[1:0] | | | | M4PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | M3PR[1:0] | | | | M2PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | M1PR[1:0] | | | | M0PR[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 1 | 0 | | | 1 | 0 |

**Bits 0:1, 4:5, 8:9, 12:13, 16:17, 20:21, 24:25, 28:29 – MxPR** Master x Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See “Arbitration Priority Scheme” for details.

### 19.4.4 Bus Matrix Priority Registers B For Slaves

**Name:** MATRIX_PRBSx
**Offset:** 0x84 + x*0x08 [x=0..8]
**Reset:** 0x00000222
**Property:** Read/Write

This register can only be written if the WPE bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 1 | 0 | | | 1 | 0 |

**Bits 0:1, 4:5, 8:9, 12:13, 16:17 – MxPR** Master 8 Priority
Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.
All the masters programmed with the same MxPR value for the slave make up a priority pool.
Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.
Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).
See "Arbitration Priority Scheme" for details.

### 19.4.5 Bus Matrix Master Remap Control Register

**Name:** MATRIX_MRCR
**Offset:** 0x0100
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | RCB12 | RCB11 | RCB10 | RCB9 | RCB8 |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RCB7 | RCB6 | RCB5 | RCB4 | RCB3 | RCB2 | RCB1 | RCB0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 – RCBx** Remap Command Bit for Master x

| Value | Description |
|---|---|
| 0 | Disables remapped address decoding for the selected Master. |
| 1 | Enables remapped address decoding for the selected Master. |

### 19.4.6 CAN0 Configuration Register

**Name:** CCFG_CAN0
**Offset:** 0x0110
**Reset:** 0x2040019D
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CAN0DMABA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CAN0DMABA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Reserved[8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | Reserved[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |

**Bits 31:16 – CAN0DMABA[15:0]** CAN0 DMA Base Address
Gives the 16-bit MSB of the CAN0 DMA base address. The 16-bit LSB must be programmed into CAN0 user interface.
Default address is 0x20400000.

**Bits 8:0 – Reserved[8:0]** Do not change the reset value

### 19.4.7 System I/O and CAN1 Configuration Register

**Name:** CCFG_SYSIO
**Offset:** 0x0114
**Reset:** 0x20400000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CAN1DMABA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CAN1DMABA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | SYSIO12 | | | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SYSIO7 | SYSIO6 | SYSIO5 | SYSIO4 | | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

**Bits 31:16 – CAN1DMABA[15:0]** CAN1 DMA Base Address
Give the 16-bit MSB of the CAN1 DMA base address. The 16-bit LSB must be programmed into CAN1 User interface.
Default address is 0x20400000.

**Bit 12 – SYSIO12** PB12 or ERASE Assignment

| Value | Description |
|---|---|
| 0 | ERASE function selected. |
| 1 | PB12 function selected. |

**Bit 7 – SYSIO7** PB7 or TCK/SWCLK Assignment

| Value | Description |
|---|---|
| 0 | TCK/SWCLK function selected. |
| 1 | PB7 function selected. |

**Bit 6 – SYSIO6** PB6 or TMS/SWDIO Assignment

| Value | Description |
|---|---|
| 0 | TMS/SWDIO function selected. |
| 1 | PB6 function selected. |

**Bit 5 – SYSIO5** PB5 or TDO/TRACESWO Assignment

| Value | Description |
|---|---|
| 0 | TDO/TRACESWO function selected. |
| 1 | PB5 function selected. |

**Bit 4 – SYSIO4** PB4 or TDI Assignment

| Value | Description |
|---|---|
| 0 | TDI function selected. |
| 1 | PB4 function selected. |

### 19.4.8 Peripheral Clock Configuration Register

**Name:** CCFG_PCCR
**Offset:** 0x0118
**Reset:** 0x00022224
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | I2SC1CC | I2SC0CC | TC0CC | | | | |
| Access | | R/W | R/W | R/W | | | | |
| Reset | | 0 | 0 | 0 | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 22 – I2SC1CC** I2SC1 Clock Configuration

| Value | Description |
|---|---|
| 0 | Peripheral clock of I2SC1 is used. |
| 1 | GCLK is used. |

**Bit 21 – I2SC0CC** I2SC0 Clock Configuration

| Value | Description |
|---|---|
| 0 | Peripheral clock of I2SC0 is used. |
| 1 | GCLK is used. |

**Bit 20 – TC0CC** TC0 Clock Configuration

| Value | Description |
|---|---|
| 0 | PCK6 is used (default). |
| 1 | PCK7 is used. |

### 19.4.9 Dynamic Clock Gating Register

**Name:** CCFG_DYNCKG
**Offset:** 0x011C
**Reset:** 0
**Property:** Read/Write

**Note:** Clearing this register optimizes the power consumption of the system bus circuitry.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | EFCCKG | BRIDCKG | MATCKG |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – EFCCKG** EFC Dynamic Clock Gating Enable

| Value | Description |
|---|---|
| 0 | EFC dynamic clock gating enabled. The Embedded Flash Controller circuitry is driven by the clock only when an access to the Flash memory is being performed. Power consumption is optimized. |
| 1 | EFC dynamic clock gating disabled. The Embedded Flash Controller is always driven by the clock in Active mode. |

**Bit 1 – BRIDCKG** Bridge Dynamic Clock Gating Enable

| Value | Description |
|---|---|
| 0 | Bridge dynamic clock gating enabled. The peripheral bridge circuitry is driven by the clock only when a transfer to/from any peripheral located on the APB bus is being performed. Power consumption is optimized. |
| 1 | Bridge dynamic clock gating disabled. The peripheral bridge circuitry is always driven by the clock in Active mode. |

**Bit 0 – MATCKG** MATRIX Dynamic Clock Gating

| Value | Description |
|---|---|
| 0 | MATRIX dynamic clock gating enabled. The MATRIX circuitry is driven by the clock only when a transfer to a peripheral is being performed. Power consumption is optimized. |
| 1 | MATRIX dynamic clock gating disabled. The MATRIX circuitry is always driven by the clock in Active mode. |

### 19.4.10 SMC NAND Flash Chip Select Configuration Register

**Name:** CCFG_SMCNFCS
**Offset:** 0x0124
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | SDRAMEN | SMC_NFCS3 | SMC_NFCS2 | SMC_NFCS1 | SMC_NFCS0 |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bit 4 – SDRAMEN** SDRAM Enable

**WARNING** This bit must not be used if SMC_NFCS1 is set.

WARNING: This must not be used if SMC_NFCS1 is set.

| Value | Description |
|---|---|
| 0 | NCS1 is not assigned to SDRAM. |
| 1 | NCS1 is assigned to SDRAM. |

**Bit 3 – SMC_NFCS3** SMC NAND Flash Chip Select 3 Assignment

| Value | Description |
|---|---|
| 0 | NCS3 is not assigned to a NAND Flash (NANDOE and NANWE not used for NCS3). |
| 1 | NCS3 is assigned to a NAND Flash (NANDOE and NANWE used for NCS3). |

**Bit 2 – SMC_NFCS2** SMC NAND Flash Chip Select 2 Assignment

| Value | Description |
|---|---|
| 0 | NCS2 is not assigned to a NAND Flash (NANDOE and NANWE not used for NCS2). |
| 1 | NCS2 is assigned to a NAND Flash (NANDOE and NANWE used for NCS2). |

**Bit 1 – SMC_NFCS1** SMC NAND Flash Chip Select 1 Assignment

**WARNING** This bit must not be used if SDRAMEN is set.

| Value | Description |
|---|---|
| 0 | NCS1 is not assigned to a NAND Flash (NANDOE and NANWE not used for NCS1). |
| 1 | NCS1 is assigned to a NAND Flash (NANDOE and NANWE used for NCS1). |

**Bit 0 – SMC_NFCS0** SMC NAND Flash Chip Select 0 Assignment

| Value | Description |
|---|---|
| 0 | NCS0 is not assigned to a NAND Flash (NANDOE and NANWE not used for NCS0). |
| 1 | NCS0 is assigned to a NAND Flash (NANDOE and NANWE used for NCS0). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 19.4.11 Write Protection Mode Register

**Name:** MATRIX_WPMR
**Offset:** 0x01E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x4D4154 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

Refer to the "Register Write Protection" section for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x4D4154 (“MAT” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x4D4154 (“MAT” in ASCII). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 19.4.12 Write Protection Status Register

**Name:** MATRIX_WPSR
**Offset:** 0x01E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last write of the MATRIX_WPMR. |
| 1 | A write protection violation has occurred since the last write of the MATRIX_WPMR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 20. USB Transmitter Macrocell Interface (UTMI)



**Important:** This module is not supported. Refer to the section Errata for more information.

## 20.1 Description

The USB Transmitter Macrocell Interface (UTMI) registers manage specific aspects of the integrated USB transmitter macrocell functionality not controlled in USB sections.

## 20.2 Embedded Characteristics

- 32-bit UTMI Registers Control Product-specific Behavior

© 2021 Microchip Technology Inc.
and its subsidiaries

## 20.3 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | UTMI_OHCIICR | 7:0 | | | APPSTART | ARIE | | | | RESx |
| | | 15:8 | | | | | | | | |
| | | 23:16 | UDPPUDIS | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | UTMI_CKTRIM | 7:0 | | | | | | | FREQ[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 20.3.1 OHCI Interrupt Configuration Register

**Name:** UTMI_OHCIICR
**Offset:** 0x10
**Reset:** 0x0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UDPPUDIS | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | APPSTART | ARIE | | | | RESx |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | | | 0 |

**Bit 23 – UDPPUDIS** USB Device Pull-up Disable

| Value | Description |
|---|---|
| 0 | USB device pull-up connection is enabled. |
| 1 | USB device pull-up connection is disabled. |

**Bit 5 – APPSTART** Reserved

| Value | Description |
|---|---|
| 0 | Must write 0. |

**Bit 4 – ARIE** OHCI Asynchronous Resume Interrupt Enable

| Value | Description |
|---|---|
| 0 | Interrupt disabled. |
| 1 | Interrupt enabled. |

**Bit 0 – RESx** USB PORTx Reset

| Value | Description |
|---|---|
| 0 | Resets USB port. |
| 1 | Usable USB port. |

### 20.3.2 UTMI Clock Trimming Register

**Name:** UTMI_CKTRIM
**Offset:** 0x30
**Reset:** 0x00010000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | FREQ[1:0] | |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – FREQ[1:0]** UTMI Reference Clock Frequency

| Value | Name | Description |
|---|---|---|
| 0 | XTAL12 | 12 MHz reference clock |
| 1 | XTAL16 | 16 MHz reference clock |

# 21. Chip Identifier (CHIPID)

## 21.1 Description

Chip Identifier (CHIPID) registers are used to recognize the device and its revision. These registers provide the sizes and types of the on-chip memories, as well as the set of embedded peripherals.

Two CHIPID registers are embedded: Chip ID register (CHIPID_CIDR) and Chip ID Extension register (CHIPID_EXID). Both registers contain a hard-wired value that is read-only.

The CHIPID_CIDR register contains the following fields:

- VERSION: Identifies the revision of the silicon
- EPROC: Indicates the embedded ARM processor
- NVPTYP and NVPSIZ: Identify the type of embedded non-volatile memory and the size
- SRAMSIZ: Indicates the size of the embedded SRAM
- ARCH: Identifies the set of embedded peripherals
- EXT: Shows the use of the extension identifier register

The CHIPID_EXID register is device-dependent and reads ‘0’ if CHIPID_CIDR.EXT = 0.

## 21.2 Embedded Characteristics

- Chip ID Registers
  - Identification of the Device Revision, Sizes of the Embedded Memories, Set of Peripherals, Embedded Processor

© 2021 Microchip Technology Inc.
and its subsidiaries

## 21.3 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CHIPID_CIDR | 7:0 | EPROC[2:0] | | | VERSION[4:0] | | | | |
| | | 15:8 | NVPSIZ2[3:0] | | | | NVPSIZ[3:0] | | | |
| | | 23:16 | ARCH[3:0] | | | | SRAMSIZ[3:0] | | | |
| | | 31:24 | EXT | NVPTYP[2:0] | | | ARCH[7:4] | | | |
| 0x04 | CHIPID_EXID | 7:0 | EXID[7:0] | | | | | | | |
| | | 15:8 | EXID[15:8] | | | | | | | |
| | | 23:16 | EXID[23:16] | | | | | | | |
| | | 31:24 | EXID[31:24] | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 21.3.1 Chip ID Register

**Name:** CHIPID_CIDR
**Offset:** 0x0
**Reset:** 0xA1220E01
**Property:** Read-only

Values not listed for bitfields must be considered as “reserved”

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | EXT | NVPTYP[2:0] | | | ARCH[7:4] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ARCH[3:0] | | | | SRAMSIZ[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NVPSIZ2[3:0] | | | | NVPSIZ[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EPROC[2:0] | | | VERSION[4:0] | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bit 31 – EXT** Extension Flag

| Value | Description |
|---|---|
| 0 | Chip ID has a single register definition without extension. |
| 1 | An extended Chip ID exists. |

**Bits 30:28 – NVPTYP[2:0]** Non-volatile Program Memory Type

| Value | Name | Description |
|---|---|---|
| 2 | FLASH | Embedded Flash Memory |

**Bits 27:20 – ARCH[7:0]** Architecture Identifier

| Value | Name | Description |
|---|---|---|
| 18 | SAMV71 | SAMV71 |

**Bits 19:16 – SRAMSIZ[3:0]** Internal SRAM Size

| Value | Name | Description |
|---|---|---|
| 2 | 384K | 384 Kbytes |

**Bits 15:12 – NVPSIZ2[3:0]** Second Non-volatile Program Memory Size

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |

**Bits 11:8 – NVPSIZ[3:0]** Non-volatile Program Memory Size

| Value | Name | Description |
|---|---|---|
| 14 | 2048K | 2048 Kbytes |

**Bits 7:5 – EPROC[2:0]** Embedded Processor

| Value | Name | Description |
|---|---|---|
| 0 | SAM x7 | Cortex-M7 |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bits 4:0 – VERSION[4:0]** Version of the Device
Current version of the device.

### 21.3.2 Chip ID Extension Register

**Name:** CHIPID_EXID
**Offset:** 0x4
**Reset:** 0x00000002
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | EXID[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EXID[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXID[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EXID[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**Bits 31:0 – EXID[31:0]** Chip ID Extension
This field is cleared if CHIPID_CIDR.EXT = 0.

# 22. Enhanced Embedded Flash Controller (EEFC)

## 22.1 Description

The Enhanced Embedded Flash Controller (EEFC) provides the interface of the Flash block with the 32-bit internal bus.

Its 128-bit wide memory interface increases performance. It also manages the programming, erasing, locking and unlocking sequences of the Flash using a full set of commands. One of the commands returns the embedded Flash descriptor definition that informs the system about the Flash organization, thus making the software generic.

## 22.2 Embedded Characteristics

- Increases Performance in Thumb-2 Mode with 128-bit-wide Memory Interface up to 150 MHz
- Code Loop Optimization
- 128 Lock Bits, Each Protecting a Lock Region
- 9 General-purpose GPNVM Bits
- One-by-one Lock Bit Programming
- Commands Protected by a Keyword
- Erase the Entire Flash
- Erase by Sector
- Erase by Page
- Provides Unique Identifier
- Provides 512-byte User Signature Area
- Supports Erasing before Programming
- Locking and Unlocking Operations
- ECC Single and Multiple Error Flags Report
- Supports Read of the Calibration Bits
- Register Write Protection

## 22.3 Product Dependencies

### 22.3.1 Power Management

The Enhanced Embedded Flash Controller (EEFC) is continuously clocked. The Power Management Controller has no effect on its behavior.

### 22.3.2 Interrupt Sources

The EEFC interrupt line is connected to the interrupt controller. Using the EEFC interrupt requires the interrupt controller to be programmed first. The EEFC interrupt is generated only if the value of EEFC_FMR.FRDY is ‘1’.

## 22.4 Functional Description

### 22.4.1 Embedded Flash Organization

The embedded Flash interfaces directly with the internal bus. The embedded Flash is composed of:

- One memory plane organized in several pages of the same size for the code
- A separate 2 x 512-byte memory area which includes the unique chip identifier
- A separate 512-byte memory area for the user signature

© 2021 Microchip Technology Inc.
and its subsidiaries

- Two 128-bit read buffers used for code read optimization
- One 128-bit read buffer used for data read optimization
- One write buffer that manages page programming. The write buffer size is equal to the page size. This buffer is write-only and accessible all along the 1 Mbyte address space, so that each word can be written to its final address.
- Several lock bits used to protect write/erase operation on several pages (lock region). A lock bit is associated with a lock region composed of several pages in the memory plane.
- Several bits that may be set and cleared through the EEFC interface, called general-purpose non-volatile memory bits (GPNVM bits)

The embedded Flash size, the page size, the organization of lock regions and the definition of GPNVM bits are specific to the device. The EEFC returns a descriptor of the Flash controller after a ‘Get Flash Descriptor’ command has been issued by the application (see the “Get Flash Descriptor Command” section).

**Figure 22-1. Flash Memory Areas**

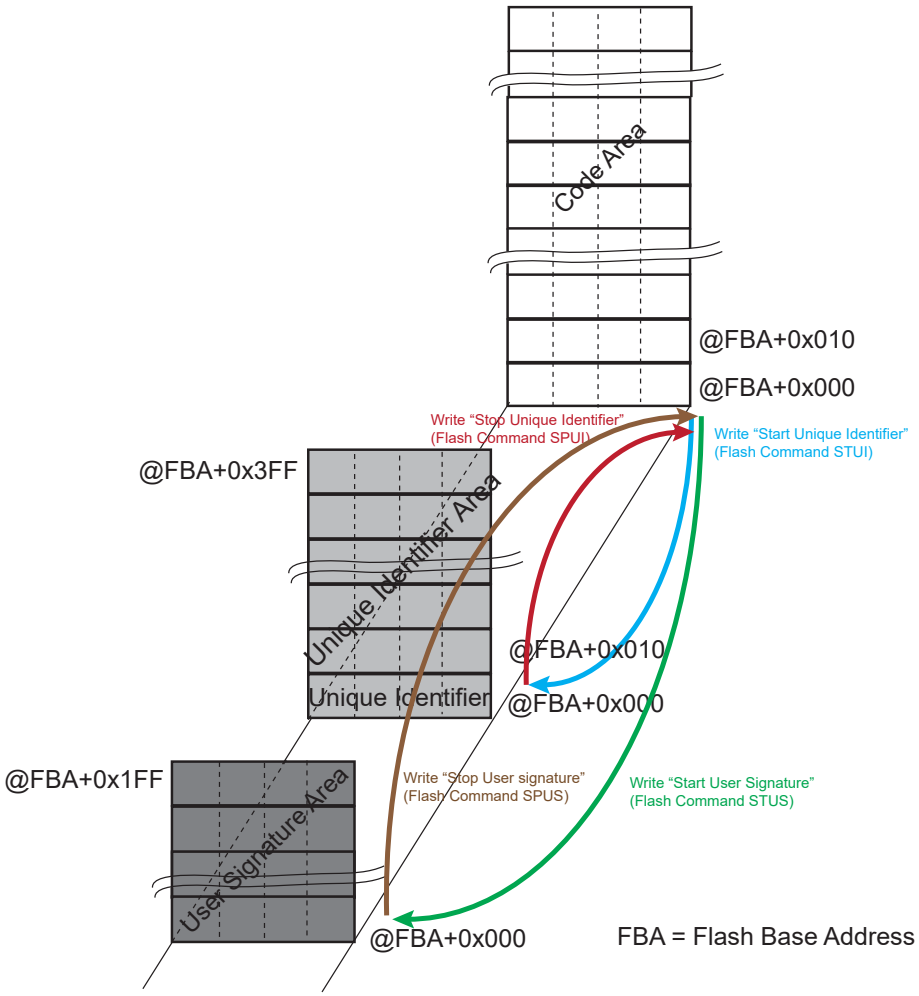
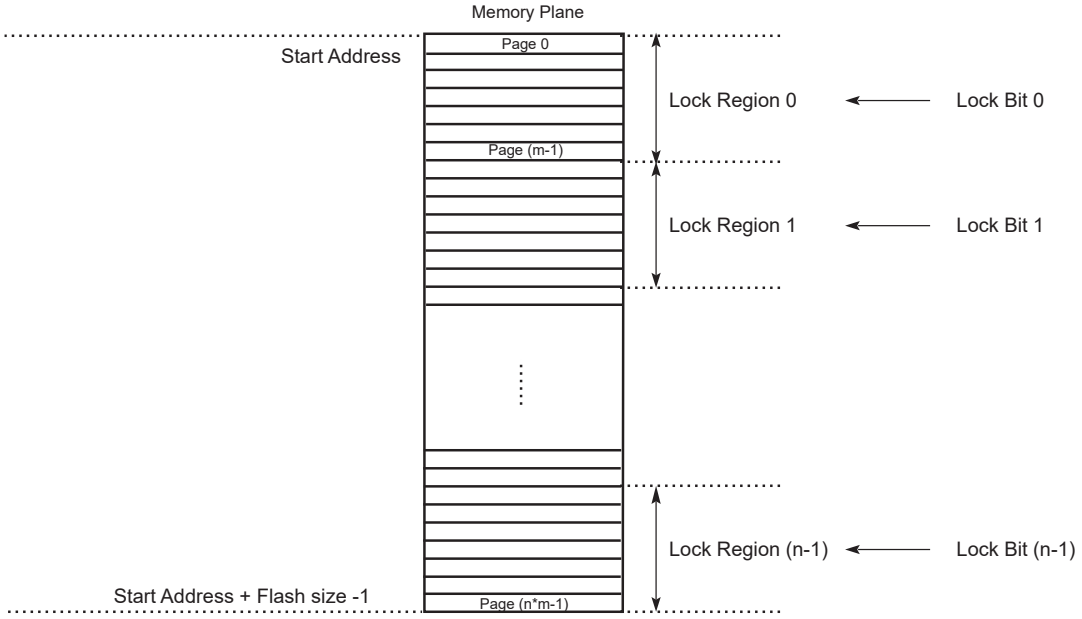


**Figure 22-2. Organization of Embedded Flash for Code**




### 22.4.2 Read Operations

An optimized controller manages embedded Flash reads, thus increasing performance when the processor is running in Thumb-2 mode by means of the 128-bit-wide memory interface.

The Flash memory is accessible through 8-, 16- and 32-bit reads.

As the Flash block size is smaller than the address space reserved for the internal memory area, the embedded Flash wraps around the address space and appears to be repeated within it.

The read operations can be performed with or without wait states. Wait states must be programmed in the field FWS in the Flash Mode register (EEFC_FMR). Defining FWS as 0 enables the single-cycle access of the embedded Flash. For more details, refer to the section “Electrical Characteristics” of this datasheet.

#### 22.4.2.1 Code Read Optimization

Code read optimization is enabled if the bit EEFC_FMR.SCOD is cleared.

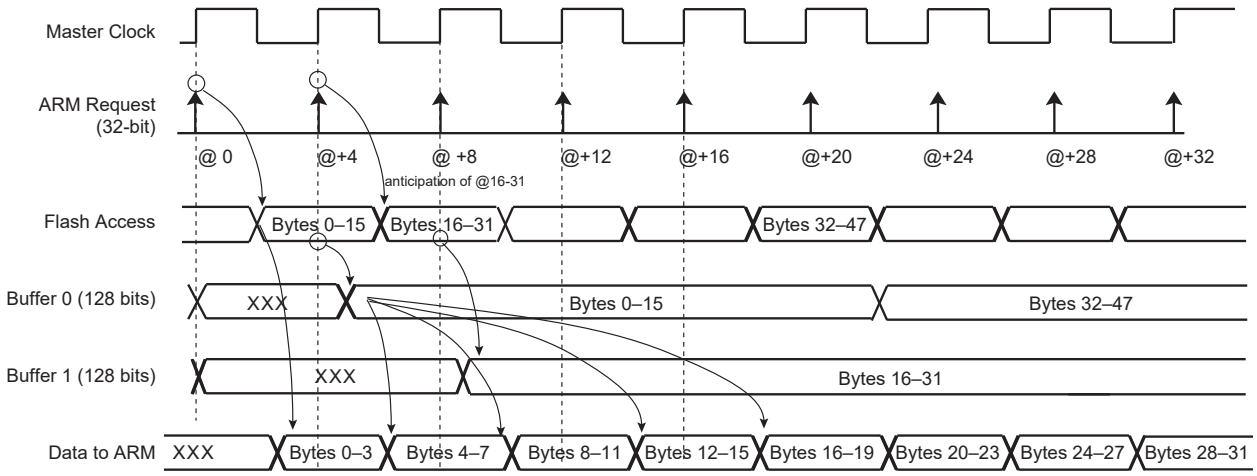
A system of 2 x 128-bit buffers is added in order to optimize sequential code fetch.

**Note:** Immediate consecutive code read accesses are not mandatory to benefit from this optimization.

The sequential code read optimization is enabled by default. If the bit EEFC_FMR.SCOD is set, these buffers are disabled and the sequential code read is no longer optimized.

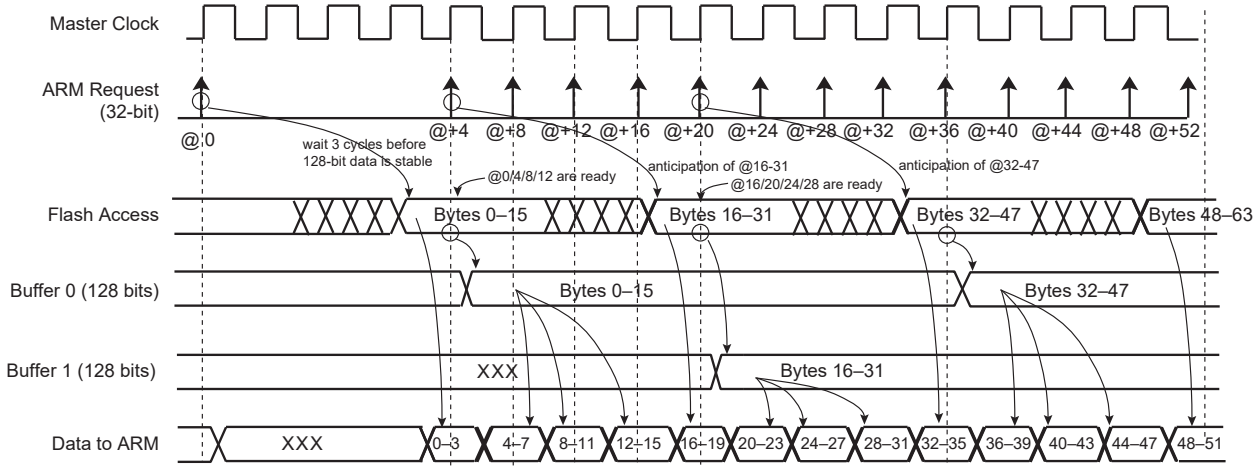
Another system of 2 x 128-bit buffers is added in order to optimize loop code fetch. Refer to the “Code Loop Optimization” section for more details.

**Figure 22-3. Code Read Optimization for FWS = 0**




**Note:** When FWS is equal to '0', all the accesses are performed in a single-cycle access.

**Figure 22-4. Code Read Optimization for FWS = 3**




**Note:** When FWS is between 1 and 3, in case of sequential reads, the first access takes (FWS + 1) cycles. The following accesses take only one cycle.

### 22.4.2.2 Code Loop Optimization

Code loop optimization is enabled when the bit EEFC_FMR.CLOE is set.

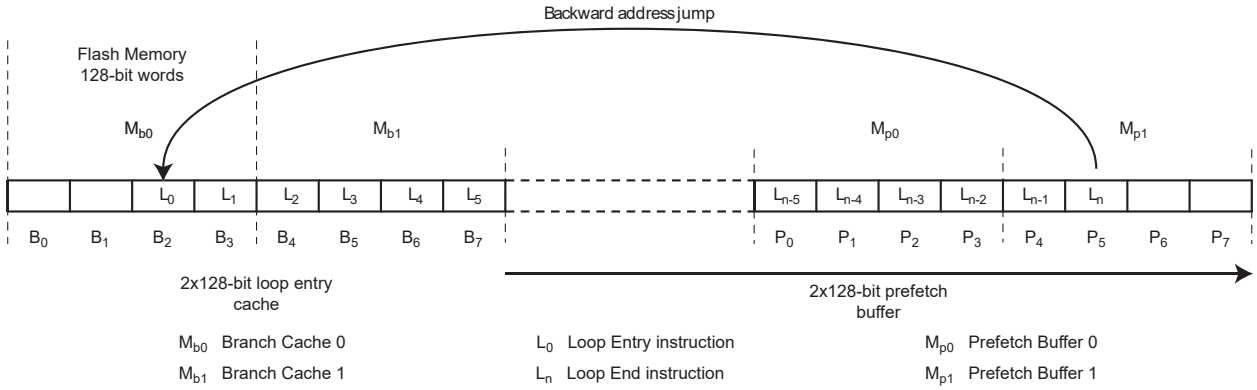
When a backward jump is inserted in the code, the pipeline of the sequential optimization is broken and becomes inefficient. In this case, the loop code read optimization takes over from the sequential code read optimization to prevent the insertion of wait states. The loop code read optimization is enabled by default. In EEFC_FMR, if the bit CLOE is reset to 0 or the bit SCOD is set, these buffers are disabled and the loop code read is not optimized.

When code loop optimization is enabled, if inner loop body instructions $L_0$ to $L_n$ are positioned from the 128-bit Flash memory cell $M_{b0}$ to the memory cell $M_{p1}$, after recognition of a first backward branch, the first two Flash memory cells $M_{b0}$ and $M_{b1}$ targeted by this branch are cached for fast access from the processor at the next loop iteration.

Then by combining the sequential prefetch (described in the "Code Read Optimization" section) through the loop body with the fast read access to the loop entry cache, the entire loop can be iterated with no wait state.

The following figure illustrates code loop optimization.

**Figure 22-5. Code Loop Optimization**

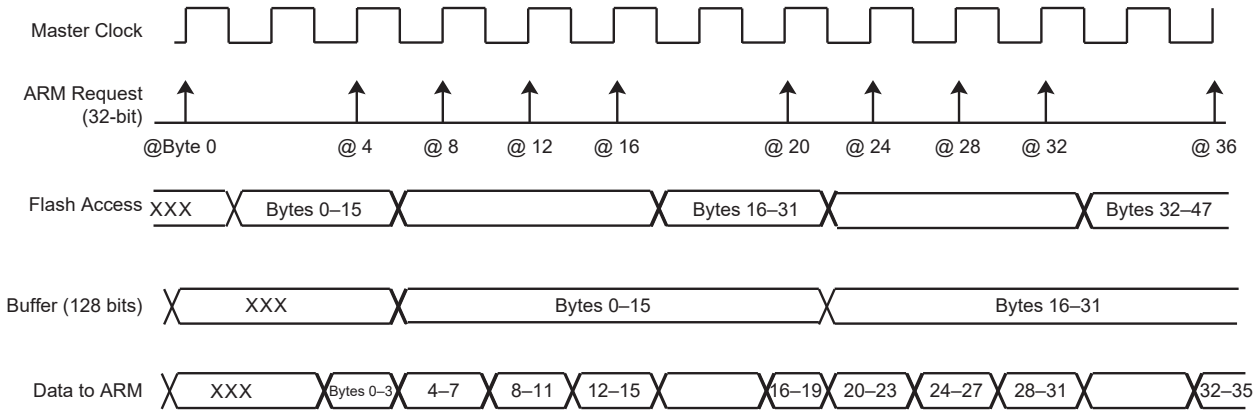



### 22.4.2.3 Data Read Optimization

The organization of the Flash in 128 bits is associated with two 128-bit prefetch buffers and one 128-bit data read buffer, thus providing maximum system performance. This buffer is added in order to store the requested data plus all the data contained in the 128-bit aligned data. This speeds up sequential data reads if, for example, FWS is equal to 1 (see Figure 22-6). The data read optimization is enabled by default. If the bit EEFC_FMR.SCOD is set, this buffer is disabled and the data read is no longer optimized.

Note: No consecutive data read accesses are mandatory to benefit from this optimization.

**Figure 22-6. Data Read Optimization for FWS = 1**




## 22.4.3 Flash Commands

The EEFC offers a set of commands to manage programming the Flash memory, locking and unlocking lock regions, consecutive programming, locking and full Flash erasing, etc.

The commands are listed in the following table.

**Table 22-1. Set of Commands**

| Command | Value | Mnemonic |
|---|---|---|
| Get Flash Descriptor | 0x00 | GETD |
| Write Page | 0x01 | WP |
| Write Page and Lock | 0x02 | WPL |
| Erase Page and Write Page | 0x03 | EWP |
| Erase Page and Write Page and then Lock | 0x04 | EWPL |
| Erase All | 0x05 | EA |

.....continued

| Command | Value | Mnemonic |
|---|---|---|
| Erase Pages | 0x07 | EPA |
| Set Lock Bit | 0x08 | SLB |
| Clear Lock Bit | 0x09 | CLB |
| Get Lock Bit | 0x0A | GLB |
| Set GPNVM Bit | 0x0B | SGPB |
| Clear GPNVM Bit | 0x0C | CGPB |
| Get GPNVM Bit | 0x0D | GGPB |
| Start Read Unique Identifier | 0x0E | STUI |
| Stop Read Unique Identifier | 0x0F | SPUI |
| Get CALIB Bit | 0x10 | GCALB |
| Erase Sector | 0x11 | ES |
| Write User Signature | 0x12 | WUS |
| Erase User Signature | 0x13 | EUS |
| Start Read User Signature | 0x14 | STUS |
| Stop Read User Signature | 0x15 | SPUS |

To execute one of these commands, select the required command using the FCMD field in the Flash Command register (EEFC_FCR). As soon as EEFC_FCR is written, the FRDY flag and the FVALUE field in the Flash Result register (EEFC_FRR) are automatically cleared. Once the current command has completed, the FRDY flag is automatically set. If an interrupt has been enabled by setting the bit EEFC_FMR.FRDY, the corresponding interrupt line of the interrupt controller is activated. (Note that this is true for all commands except for the STUI command. The FRDY flag is not set when the STUI command has completed.)

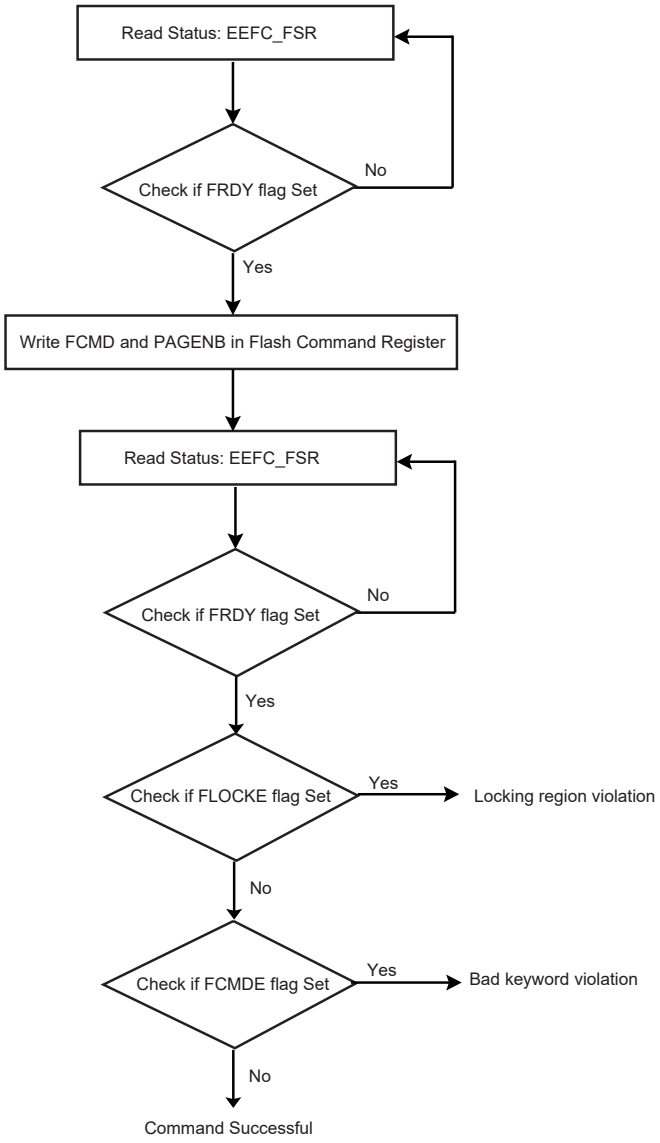
All the commands are protected by the same keyword, which must be written in the eight highest bits of EEFC_FCR.

Writing EEFC_FCR with data that does not contain the correct key and/or with an invalid command has no effect on the whole memory plane, but the FCMDE flag is set in the Flash Status register (EEFC_FSR). This flag is automatically cleared by a read access to EEFC_FSR.

When the current command writes or erases a page in a locked region, the command has no effect on the whole memory plane, but the FLOCKE flag is set in EEFC_FSR. This flag is automatically cleared by a read access to EEFC_FSR.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 22-7. Command State Chart**




### 22.4.3.1 Get Flash Descriptor Command

This command provides the system with information on the Flash organization. The system can take full advantage of this information. For instance, a device could be replaced by one with more Flash capacity, and so the software is able to adapt itself to the new configuration.

To get the embedded Flash descriptor, the application writes the GETD command in EEFC_FCR. The first word of the descriptor can be read by the software application in EEFC_FRR as soon as the FRDY flag in EEFC_FSR rises. The next reads of EEFC_FRR provide the following word of the descriptor. If extra read operations to EEFC_FRR are done after the last word of the descriptor has been returned, the EEFC_FRR value is 0 until the next valid command.

**Table 22-2. Flash Descriptor Definition**

| Symbol | Word Index | Description |
|---|---|---|
| FL_ID | 0 | Flash interface description |
| FL_SIZE | 1 | Flash size in bytes |
| FL_PAGE_SIZE | 2 | Page size in bytes |

| .....continued | | |
|---|---|---|
| **Symbol** | **Word Index** | **Description** |
| FL_NB_PLANE | 3 | Number of planes |
| FL_PLANE[0] | 4 | Number of bytes in the plane |
| FL_NB_LOCK | 4 + FL_NB_PLANE | Number of lock bits. A bit is associated with a lock region. A lock bit is used to prevent write or erase operations in the lock region. |
| FL_LOCK[0] | 4 + FL_NB_PLANE + 1 | Number of bytes in the first lock region |

#### 22.4.3.2 Write Commands

DMA write accesses must be 32-bit aligned. If a single byte has to be written in a 32-bit word, the rest of the word must be written with ones.

Several commands are used to program the Flash.

Only ‘0’ values can be programmed using Flash technology; ‘1’ is the erased value. In order to program words in a page, the page must first be erased. Commands are available to erase the entire Flash or a given number of pages. With the EWP and EWPL commands, a page erase is done automatically before a page programming.

After programming, the page (the entire lock region) can be locked to prevent miscellaneous write or erase sequences. The lock bit can be automatically set after page programming using WPL or EWPL commands.

Data to be programmed in the Flash must be written in an internal latch buffer before writing the programming command in EEFC_FCR. Data can be written at their final destination address, as the latch buffer is mapped into the Flash memory address space and wraps around within this Flash address space.

Byte and half-word AHB accesses to the latch buffer are not allowed. Only 32-bit word accesses are supported.

32-bit words must be written continuously, in either ascending or descending order. Writing the latch buffer in a random order is not permitted. This prevents mapping a C-code structure to the latch buffer and accessing the data of the structure in any order. It is instead recommended to fill in a C-code structure in SRAM and copy it in the latch buffer in a continuous order.

Write operations in the latch buffer are performed with the number of wait states programmed for reading the Flash.

The latch buffer is automatically re-initialized, that is, written with logical ‘1’, after execution of each programming command.

The programming sequence is as follows:

1. Write the data to be programmed in the latch buffer.
2. Write the programming command in EEFC_FCR. This automatically clears the EEFC_FSR.FRDY bit.
3. When Flash programming is completed, the EEFC_FSR.FRDY bit rises. If an interrupt has been enabled by setting the EEFC_FMR.FRDY bit, the interrupt line of the EEFC is activated.

Three errors can be detected in EEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Lock Error: The page to be programmed belongs to a locked region. A command must be run previously to unlock the corresponding region.
- Flash Error: When programming is completed, the WriteVerify test of the Flash memory has failed.

Only one page can be programmed at a time. It is possible to program all the bits of a page (full page programming) or only some of the bits of the page (partial page programming).

Depending on the number of bits to be programmed within the page, the EEFC adapts the write operations required to program the Flash.

When a ‘Write Page’ (WP) command is issued, the EEFC starts the programming sequence and all the bits written at ‘0’ in the latch buffer are cleared in the Flash memory array.

During programming, that is, until EEFC_FSR.FDRY rises, access to the Flash is not allowed.

© 2021 Microchip Technology Inc.
and its subsidiaries

#### 22.4.3.2.1 Full Page Programming

To program a full page, all the bits of the page must be erased before writing the latch buffer and issuing the WP command. The latch buffer must be written in ascending order, starting from the first address of the page. See Figure 22-8.

#### 22.4.3.2.2 Partial Page Programming

To program only part of a page using the WP command, the following constraints must be respected:

- Data to be programmed must be contained in integer multiples of 128-bit address-aligned words.
- 128-bit words can be programmed only if all the corresponding bits in the Flash array are erased (at logical value ‘1’).

See 22.4.3.2.4 Programming Bytes.

#### 22.4.3.2.3 Optimized Partial Page Programming

The EEFC automatically detects the number of 128-bit words to be programmed. If only one 128-bit aligned word is to be programmed in the Flash array, the process is optimized to reduce the time needed for programming.

If several 128-bit words are to be programmed, a standard page programming operation is performed.

See Figure 22-10.

#### 22.4.3.2.4 Programming Bytes

Individual bytes can be programmed using the Partial Page Programming mode.

In this case, an area of 128 bits must be reserved for each byte.

Refer to Figure 22-11

**Figure 22-8. Full Page Programming**

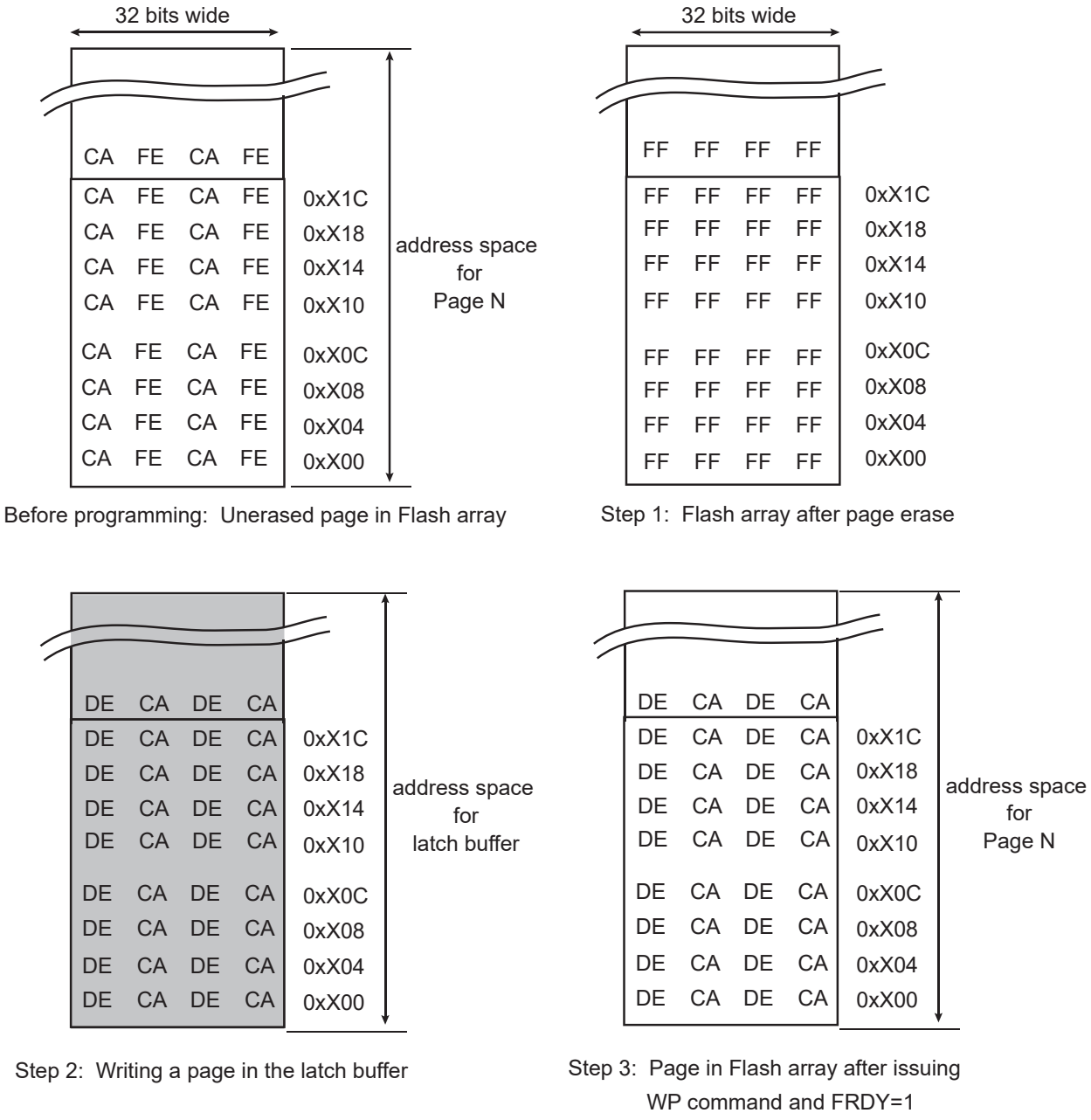
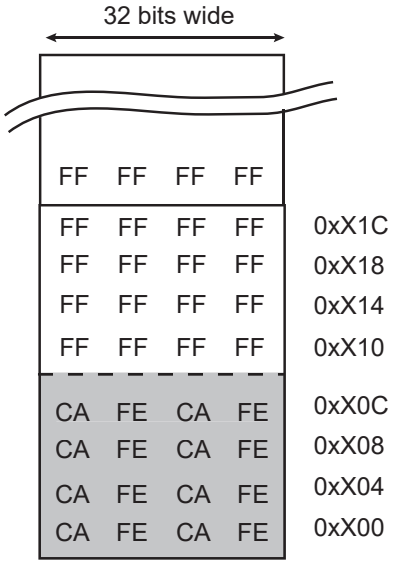


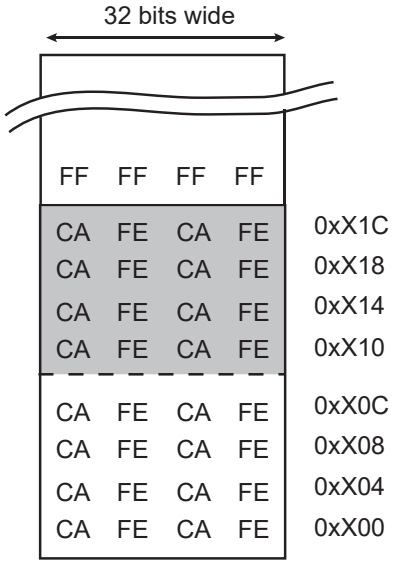
**Figure 22-9. Partial Page Programming**




Step 1: Flash array after page erase

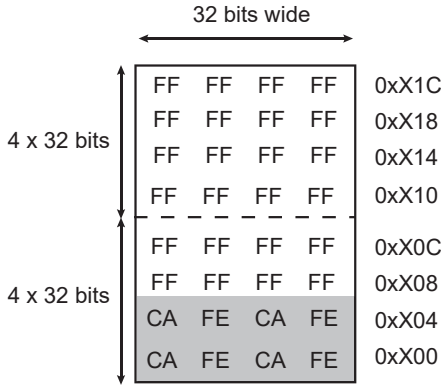



Step 2: Flash array after programming
128-bit at address 0xX00 (write latch buffer + WP)

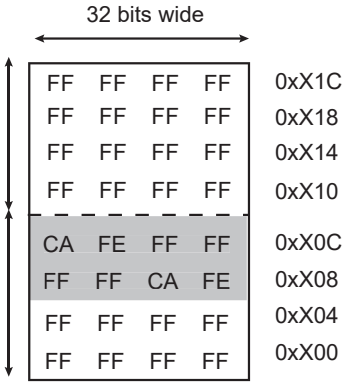



Step 3: Flash array after programming
a second 128-bit data at address 0xX10
(write latch buffer + WP)

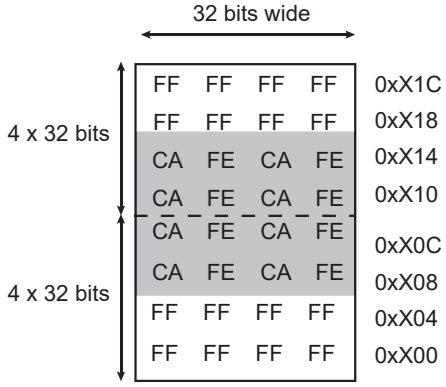
**Figure 22-10. Optimized Partial Page Programming**



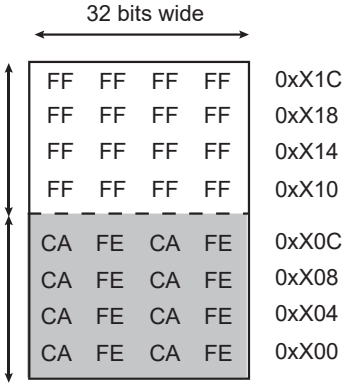

Case 1: 2 x 32 bits modified, not crossing 128-bit boundary
User programs WP, Flash Controller sends Write Word
=> Only 1 word programmed => programming period reduced




Case 2: 2 x 32 bits modified, not crossing 128-bit boundary
User programs WP, Flash Controller sends Write Word
=> Only 1 word programmed => programming period reduced

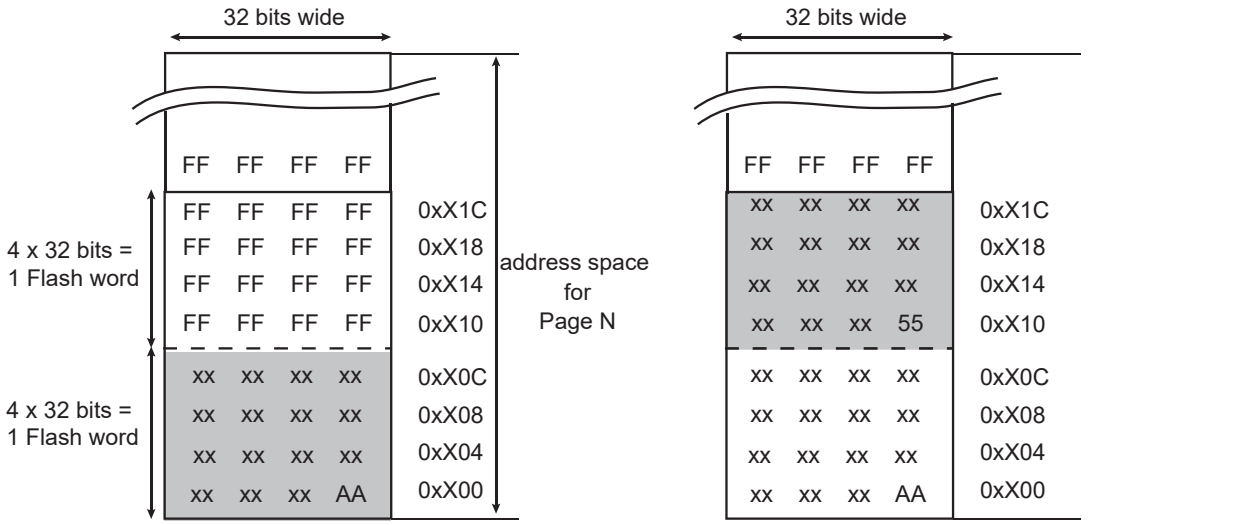



Case 3: 4 x 32 bits modified across 128-bit boundary
User programs WP, Flash Controller sends WP
=> Whole page programmed




Case 4: 4 x 32 bits modified, not crossing 128-bit boundary
User programs WP, Flash Controller sends Write Word
=> Only 1 word programmed => programming period reduced

**Figure 22-11. Programming Bytes in the Flash**




Step 1: Flash array after programming first byte (0xAA)
128-bit used at address 0xX00 (write latch buffer + WP)

Step 2: Flash array after programming second byte (0x55)
128-bit used at address 0xX10 (write latch buffer + WP)

Note: The byte location shown here is for example only, it can be any byte location within a 64-bit word

### 22.4.3.3 Erase Commands

Erase commands are allowed only on unlocked regions. Depending on the Flash memory, several commands can be used to erase the Flash:

- Erase All Memory (EA): All memory is erased. The processor must not fetch code from the Flash memory.
- Erase Pages (EPA): 4, 8, 16, or 32 pages are erased in the Flash sector selected. The first page to be erased is specified in the FARG[15:2] field of the EEFC_FCR. The first page number must be a multiple of 8, 16, or 32 depending on the number of pages to erase simultaneously.
- Erase Sector (ES): A full memory sector is erased. Sector size depends on the Flash memory. EEFC_FCR.FARG must be set with a page number that is in the sector to be erased.

**Note:** If one sub-sector is locked within the first sector, the Erase Sector (ES) command cannot be processed on non-locked sub-sectors of the first sector. All the lock bits of the first sector must be cleared prior to issuing an ES command on the first sector. After the ES command has been issued, the first sector lock bits must be reverted to the state before clearing them.

If the processor is fetching code from the Flash memory while the EPA or ES command is being executed, the processor accesses are stalled until the EPA command is completed. To avoid stalling the processor, the code can be run out of internal SRAM.

The following are the erase sequence:

1. Erase starts immediately one of the erase commands and the FARG field are written in EEFC_FCR. For the EPA command, the two lowest bits of the FARG field define the number of pages to be erased (FARG[1:0]), see table below.

**Table 22-3. EEFC_FCR.FARG Field for EPA Command**

| FARG[1:0] | Number of pages to be erased with EPA command |
|---|---|
| 0 | 4 pages (only valid for small 8-KB sectors) |
| 1 | 8 pages (only valid for small 8-KB sectors) |
| 2 | 16 pages |

| .....continued | |
|---|---|
| **FARG[1:0]** | **Number of pages to be erased with EPA command** |
| 3 | 32 pages (not valid for small 8-KB sectors) |

2. When erasing is completed, the EEFC_FSR.FRDY bit rises. If an interrupt has been enabled by setting the EEFC_FMR.FRDY bit, the interrupt line of the interrupt controller is activated.

Three errors can be detected in EEFC_FSR after an erasing sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Lock Error: At least one page to be erased belongs to a locked region. The erase command has been refused, no page has been erased. A command must be run previously to unlock the corresponding region.
- Flash Error: At the end of the erase period, the EraseVerify test of the Flash memory has failed.

#### 22.4.3.4 Lock Bit Protection

Lock bits are associated with several pages in the embedded Flash memory plane. This defines lock regions in the embedded Flash memory plane. They prevent writing/erasing protected pages.

The lock sequence is the following:

1. Execute the ‘Set Lock Bit’ command by writing EEFC_FCR.FCMD with the SLB command and EEFC_FCR.FARG with a page number to be protected.
2. When the locking completes, the bit EEFC_FSR.FRDY rises. If an interrupt has been enabled by setting the bit EEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.
3. The result of the SLB command can be checked running a ‘Get Lock Bit’ (GLB) command.
   Note: The value of the FARG argument passed together with SLB command must not exceed the higher lock bit index available in the product.

Two errors can be detected in EEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Flash Error: At the end of the programming, the EraseVerify or WriteVerify test of the Flash memory has failed.

It is possible to clear lock bits previously set. After the lock bits are cleared, the locked region can be erased or programmed. The unlock sequence is the following:

1. Execute the ‘Clear Lock Bit’ command by writing EEFC_FCR.FCMD with the CLB command and EEFC_FCR.FARG with a page number to be unprotected.
2. When the unlock completes, the bit EEFC_FSR.FRDY rises. If an interrupt has been enabled by setting the bit EEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.
   Note: The value of the FARG argument passed together with CLB command must not exceed the higher lock bit index available in the product.

Two errors can be detected in EEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Flash Error: At the end of the programming, the EraseVerify or WriteVerify test of the Flash memory has failed.

The status of lock bits can be returned by the EEFC. The ‘Get Lock Bit’ sequence is the following:

1. Execute the ‘Get Lock Bit’ command by writing EEFC_FCR.FCMD with the GLB command. Field EEFC_FCR.FARG is meaningless.
2. Lock bits can be read by the software application in EEFC_FRR. The first word read corresponds to the 32 first lock bits, next reads providing the next 32 lock bits as long as it is meaningful. Extra reads to EEFC_FRR return 0.

For example, if the third bit of the first word read in EEFC_FRR is set, the third lock region is locked.

Two errors can be detected in EEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Flash Error: At the end of the programming, the EraseVerify or WriteVerify test of the Flash memory has failed.

  Note: Access to the Flash in read is permitted when a ‘Set Lock Bit’, ‘Clear Lock Bit’ or ‘Get Lock Bit’ command is executed.

#### 22.4.3.5 GPNVM Bit

GPNVM bits do not interfere with the embedded Flash memory plane. For more details, refer to the section "Memories".

The ‘Set GPNVM Bit’ sequence is the following:

1. Execute the ‘Set GPNVM Bit’ command by writing EEFC_FCR.FCMD with the SGPB command and EEFC_FCR.FARG with the number of GPNVM bits to be set.
2. When the GPNVM bit is set, the bit EEFC_FSR.FRDY rises. If an interrupt was enabled by setting the bit EEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.
3. The result of the SGPB command can be checked by running a ‘Get GPNVM Bit’ (GGPB) command.

**Note:** The value of the FARG argument passed together with SGPB command must not exceed the higher GPNVM index available in the product. Flash data content is not altered if FARG exceeds the limit. Command Error is detected only if FARG is greater than 8.

Two errors can be detected in EEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Flash Error: At the end of the programming, the EraseVerify or WriteVerify test of the Flash memory has failed.

It is possible to clear GPNVM bits previously set. The ‘Clear GPNVM Bit’ sequence is the following:

1. Execute the ‘Clear GPNVM Bit’ command by writing EEFC_FCR.FCMD with the CGPB command and EEFC_FCR.FARG with the number of GPNVM bits to be cleared.
2. When the clear completes, the bit EEFC_FSR.FRDY rises. If an interrupt has been enabled by setting the bit EEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.

**Note:** The value of the FARG argument passed together with CGPB command must not exceed the higher GPNVM index available in the product. Flash data content is not altered if FARG exceeds the limit. Command Error is detected only if FARG is greater than 8.

Two errors can be detected in EEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Flash Error: At the end of the programming, the EraseVerify or WriteVerify test of the Flash memory has failed.

The status of GPNVM bits can be returned by the EEFC. The sequence is the following:

1. Execute the ‘Get GPNVM Bit’ command by writing EEFC_FCR.FCMD with the GGPB command. Field EEFC_FCR.FARG is meaningless.
2. GPNVM bits can be read by the software application in EEFC_FRR. The first word read corresponds to the 32 first GPNVM bits, following reads provide the next 32 GPNVM bits as long as it is meaningful. Extra reads to EEFC_FRR return 0.

For example, if the third bit of the first word read in EEFC_FRR is set, the third GPNVM bit is active.

One error can be detected in EEFC_FSR after a programming sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.

**Note:** Access to the Flash in read is permitted when a ‘Set GPNVM Bit’, ‘Clear GPNVM Bit’ or ‘Get GPNVM Bit’ command is executed.

**Related Links**

11. Memories

#### 22.4.3.6 Calibration Bit

Calibration bits do not interfere with the embedded Flash memory plane.

The calibration bits cannot be modified.

The status of calibration bits are returned by the EEFC. The sequence is as follows:

1. Execute the ‘Get CALIB Bit’ command by writing EEFC_FCR.FCMD with the GCALB command. Field EEFC_FCR.FARG is meaningless.

© 2021 Microchip Technology Inc. and its subsidiaries

2. Calibration bits can be read by the software application in EEFC_FRR. The first word read corresponds to the first 32 calibration bits. The following reads provide the next 32 calibration bits as long as it is meaningful. Extra reads to EEFC_FRR return 0.

The 8/12 MHz internal RC oscillator is calibrated in production. This calibration can be read through the GCALB command. Table 22-4 shows the bit implementation.

The RC calibration for the 4 MHz is set to '1000000'.

**Table 22-4. Calibration Bit Indexes**

| Description | EEFC_FRR Bits |
|---|---|
| 8 MHz RC calibration output | [28–22] |
| 12 MHz RC calibration output | [38–32] |

#### 22.4.3.7 Security Bit Protection

When the security bit is enabled, the Embedded Trace Macrocell (ETM) is disabled and access to the Flash through the SWD interface or through the Fast Flash Programming interface is forbidden. This ensures the confidentiality of the code programmed in the Flash.

The security bit is GPNVM0.

Disabling the security bit can only be achieved by asserting the ERASE signal at '1', and after a full Flash erase is performed. When the security bit is deactivated, all accesses to the Flash are permitted.

#### 22.4.3.8 Unique Identifier Area

Each device is programmed with a 128-bit unique identifier area .

See Figure 22-1.

The sequence to read the unique identifier area is the following:

1. Execute the 'Start Read Unique Identifier' command by writing EEFC_FCR.FCMD with the STUI command. Field EEFC_FCR.FARG is meaningless.
2. Wait until the bit EEFC_FSR.FRDY falls to read the unique identifier area. The unique identifier field is located in the first 128 bits of the Flash memory mapping. The 'Start Read Unique Identifier' command reuses some addresses of the memory plane for code, but the unique identifier area is physically different from the memory plane for code.
3. To stop reading the unique identifier area, execute the 'Stop Read Unique Identifier' command by writing EEFC_FCR.FCMD with the SPUI command. Field EEFC_FCR.FARG is meaningless.
4. When the SPUI command has been executed, the bit EEFC_FSR.FRDY rises. If an interrupt was enabled by setting the bit EEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.

**Note:** During the sequence, the software cannot be fetched from the Flash.

#### 22.4.3.9 User Signature Area

Each product contains a user signature area of 512 bytes. It can be used for storage. Read, write and erase of this area is allowed.

See Figure 22-1.

The sequence to read the user signature area is the following:

1. Execute the 'Start Read User Signature' command by writing EEFC_FCR.FCMD with the STUS command. Field EEFC_FCR.FARG is meaningless.
2. Wait until the bit EEFC_FSR.FRDY falls to read the user signature area. The user signature area is located in the first 512 bytes of the Flash memory mapping. The 'Start Read User Signature' command reuses some addresses of the memory plane but the user signature area is physically different from the memory plane
3. To stop reading the user signature area, execute the 'Stop Read User Signature' command by writing EEFC_FCR.FCMD with the SPUS command. Field EEFC_FCR.FARG is meaningless.
4. When the SPUI command has been executed, the bit EEFC_FSR.FRDY rises. If an interrupt was enabled by setting the bit EEFC_FMR.FRDY, the interrupt line of the interrupt controller is activated.

© 2021 Microchip Technology Inc. and its subsidiaries

**Note:** During the sequence, the software cannot be fetched from the Flash or from the second plane in case of dual plane.

One error can be detected in EEFC_FSR after this sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.

The sequence to write the user signature area is the following:

1. Write the full page, at any page address, within the internal memory area address space.
2. Execute the 'Write User Signature' command by writing EEFC_FCR.FCMD with the WUS command. Field EEFC_FCR.FARG is meaningless.
3. When programming is completed, the bit EEFC_FSR.FRDY rises. If an interrupt has been enabled by setting the bit EEFC_FMR.FRDY, the corresponding interrupt line of the interrupt controller is activated.

Two errors can be detected in EEFC_FSR after this sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Flash Error: At the end of the programming, the WriteVerify test of the Flash memory has failed.

The sequence to erase the user signature area is the following:

1. Execute the 'Erase User Signature' command by writing EEFC_FCR.FCMD with the EUS command. Field EEFC_FCR.FARG is meaningless.
2. When programming is completed, the bit EEFC_FSR.FRDY rises. If an interrupt has been enabled by setting the bit EEFC_FMR.FRDY, the corresponding interrupt line of the interrupt controller is activated.

Two errors can be detected in EEFC_FSR after this sequence:

- Command Error: A bad keyword has been written in EEFC_FCR.
- Flash Error: At the end of the programming, the EraseVerify test of the Flash memory has failed.

#### 22.4.3.10 ECC Errors and Corrections

The Flash embeds an ECC module able to correct one unique error and able to detect two errors. The errors are detected while a read access is performed into memory array and stored in EEFC_FSR (see "EEFC Flash Status Register"). The error report is kept until EEFC_FSR is read.

There is one flag for a unique error on lower half part of the Flash word (64 LSB) and one flag for the upper half part (MSB). The multiple errors are reported in the same way.

Due to the anticipation technique to improve bandwidth throughput on instruction fetch, a reported error can be located in the next sequential Flash word compared to the location of the instruction being executed, which is located in the previously fetched Flash word.

If a software routine processes the error detection independently from the main software routine, the entire Flash located software must be rewritten because there is no storage of the error location.

If only a software routine is running to program and check pages by reading EEFC_FSR, the situation differs from the previous case. Performing a check for ECC unique errors just after page programming completion involves a read of the newly programmed page. This read sequence is viewed as data accesses and is not optimized by the Flash controller. Thus, in case of unique error, only the current page must be reprogrammed.

### 22.4.4 Register Write Protection

To prevent any single software error from corrupting EEFC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the "EEFC Write Protection Mode Register" (EEFC_WPMR).

The following register can be write-protected:

- "EEFC Flash Mode Register"

## 22.5 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | EEFC_FMR | 7:0 | | | | | | | | FRDY |
| | | 15:8 | | | | | FWS[3:0] | | | |
| | | 23:16 | | | | | | | | SCOD |
| | | 31:24 | | | | | | CLOE | | |
| 0x04 | EEFC_FCR | 7:0 | FCMD[7:0] | | | | | | | |
| | | 15:8 | FARG[7:0] | | | | | | | |
| | | 23:16 | FARG[15:8] | | | | | | | |
| | | 31:24 | FKEY[7:0] | | | | | | | |
| 0x08 | EEFC_FSR | 7:0 | | | | | FLERR | FLOCKE | FCMDE | FRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | MECCEMSB | UECCEMSB | MECCELSB | UECCELSB |
| | | 31:24 | | | | | | | | |
| 0x0C | EEFC_FRR | 7:0 | FVALUE[7:0] | | | | | | | |
| | | 15:8 | FVALUE[15:8] | | | | | | | |
| | | 23:16 | FVALUE[23:16] | | | | | | | |
| | | 31:24 | FVALUE[31:24] | | | | | | | |
| 0x10 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | EEFC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 22.5.1 EEFC Flash Mode Register

**Name:** EEFC_FMR
**Offset:** 0x00
**Reset:** 0x04000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the "EEFC Write Protection Mode Register" .

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CLOE | | |
| Access | | | | | | R/W | | |
| Reset | | | | | | 1 | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SCOD |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FWS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | FRDY |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 26 – CLOE** Code Loop Optimization Enable
No Flash read should be done during change of this field.

| Value | Description |
|---|---|
| 0 | The opcode loop optimization is disabled. |
| 1 | The opcode loop optimization is enabled. |

**Bit 16 – SCOD** Sequential Code Optimization Disable
No Flash read should be done during change of this field.

| Value | Description |
|---|---|
| 0 | The sequential code optimization is enabled. |
| 1 | The sequential code optimization is disabled. |

**Bits 11:8 – FWS[3:0]** Flash Wait State
This field defines the number of wait states for read and write operations:
FWS = Number of cycles for Read/Write operations - 1

**Bit 0 – FRDY** Flash Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | Flash ready does not generate an interrupt. |
| 1 | Flash ready (to accept a new command) generates an interrupt. |

### 22.5.2 EEFC Flash Command Register

**Name:** EEFC_FCR
**Offset:** 0x04
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FARG[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FARG[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FCMD[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:24 – FKEY[7:0]** Flash Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x5A | PASSWD | The 0x5A value enables the command defined by the bits of the register. If the field is written with a different value, the write is not performed and no action is started. |

**Bits 23:8 – FARG[15:0]** Flash Command Argument

| | | |
|---|---|---|
| GETD, GLB, GGPB, STUI, SPUI, GCALB, WUS, EUS, STUS, SPUS, EA | Commands requiring no argument, including Erase all command | FARG is meaningless, must be written with 0 |
| ES | Erase sector command | FARG must be written with any page number within the sector to be erased |
| EPA | Erase pages command | FARG[1:0] defines the number of pages to be erased<br>The start page must be written in FARG[15:2].<br>FARG[1:0] = 0: Four pages to be erased. FARG[15:2] = Page_Number / 4<br>FARG[1:0] = 1: Eight pages to be erased. FARG[15:3] = Page_Number / 8, FARG[2]=0<br>FARG[1:0] = 2: Sixteen pages to be erased. FARG[15:4] = Page_Number / 16, FARG[3:2]=0<br>FARG[1:0] = 3: Thirty-two pages to be erased. FARG[15:5] = Page_Number / 32, FARG[4:2]=0<br>Refer to “EEFC_FCR.FARG Field for EPA Command”. |

| | | |
|---|---|---|
| WP, WPL, EWP, EWPL | Programming commands | FARG must be written with the page number to be programmed |
| SLB, CLB | Lock bit commands | FARG defines the page number to be locked or unlocked |
| SGPB, CGPB | GPNVM commands | FARG defines the GPNVM number to be programmed |

**Bits 7:0 – FCMD[7:0]** Flash Command

| Value | Name | Description |
|---|---|---|
| 0x00 | GETD | Get Flash descriptor |
| 0x01 | WP | Write page |
| 0x02 | WPL | Write page and lock |
| 0x03 | EWP | Erase page and write page |
| 0x04 | EWPL | Erase page and write page then lock |
| 0x05 | EA | Erase all |
| 0x07 | EPA | Erase pages |
| 0x08 | SLB | Set lock bit |
| 0x09 | CLB | Clear lock bit |
| 0x0A | GLB | Get lock bit |
| 0x0B | SGPB | Set GPNVM bit |
| 0x0C | CGPB | Clear GPNVM bit |
| 0x0D | GGPB | Get GPNVM bit |
| 0x0E | STUI | Start read unique identifier |
| 0x0F | SPUI | Stop read unique identifier |
| 0x10 | GCALB | Get CALIB bit |
| 0x11 | ES | Erase sector |
| 0x12 | WUS | Write user signature |
| 0x13 | EUS | Erase user signature |
| 0x14 | STUS | Start read user signature |
| 0x15 | SPUS | Stop read user signature |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 22.5.3 EEFC Flash Status Register

**Name:** EEFC_FSR
**Offset:** 0x08
**Reset:** 0x00000001
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MECCEMSB | UECCEMSB | MECCELSB | UECCELSB |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FLERR | FLOCKE | FCMDE | FRDY |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 1 |

**Bit 19 – MECCEMSB** Multiple ECC Error on MSB Part of the Memory Flash Data Bus (cleared on read)

| Value | Description |
|---|---|
| 0 | No multiple error detected on 64 MSB part of the Flash memory data bus since the last read of EEFC_FSR. |
| 1 | Multiple errors detected and NOT corrected on 64 MSB part of the Flash memory data bus since the last read of EEFC_FSR. |

**Bit 18 – UECCEMSB** Unique ECC Error on MSB Part of the Memory Flash Data Bus (cleared on read)

| Value | Description |
|---|---|
| 0 | No unique error detected on 64 MSB data bus of the Flash memory since the last read of EEFC_FSR. |
| 1 | One unique error detected but corrected on 64 MSB data bus of the Flash memory since the last read of EEFC_FSR. |

**Bit 17 – MECCELSB** Multiple ECC Error on LSB Part of the Memory Flash Data Bus (cleared on read)

| Value | Description |
|---|---|
| 0 | No multiple error detected on 64 LSB part of the Flash memory data bus since the last read of EEFC_FSR. |
| 1 | Multiple errors detected and NOT corrected on 64 LSB part of the Flash memory data bus since the last read of EEFC_FSR. |

**Bit 16 – UECCELSB** Unique ECC Error on LSB Part of the Memory Flash Data Bus (cleared on read)

| Value | Description |
|---|---|
| 0 | No unique error detected on 64 LSB data bus of the Flash memory since the last read of EEFC_FSR. |
| 1 | One unique error detected but corrected on 64 LSB data bus of the Flash memory since the last read of EEFC_FSR. |

**Bit 3 – FLERR** Flash Error Status (cleared when a programming operation starts)

| Value | Description |
|---|---|
| 0 | No Flash memory error occurred at the end of programming (EraseVerify or WriteVerify test has passed). |
| 1 | A Flash memory error occurred at the end of programming (EraseVerify or WriteVerify test has failed). |

**Bit 2 – FLOCKE** Flash Lock Error Status (cleared on read)
This flag is automatically cleared when EEFC_FSR is read or EEFC_FCR is written.

| Value | Description |
|---|---|
| 0 | No programming/erase of at least one locked region has happened since the last read of EEFC_FSR. |
| 1 | Programming/erase of at least one locked region has happened since the last read of EEFC_FSR. |

**Bit 1 – FCMDE** Flash Command Error Status (cleared on read or by writing EEFC_FCR)

| Value | Description |
|---|---|
| 0 | No invalid commands and no bad keywords were written in EEFC_FCR. |
| 1 | An invalid command and/or a bad keyword was/were written in EEFC_FCR. |

**Bit 0 – FRDY** Flash Ready Status (cleared when Flash is busy)
When set, this flag triggers an interrupt if the FRDY flag is set in EEFC_FMR.
This flag is automatically cleared when the EEFC is busy.

| Value | Description |
|---|---|
| 0 | The EEFC is busy. |
| 1 | The EEFC is ready to start a new command. |

### 22.5.4 EEFC Flash Result Register

**Name:** EEFC_FRR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FVALUE[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FVALUE[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FVALUE[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FVALUE[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – FVALUE[31:0]** Flash Result Value
The result of a Flash command is returned in this register. If the size of the result is greater than 32 bits, the next resulting value is accessible at the next register read.

### 22.5.5 EEFC Write Protection Mode Register

**Name:** EEFC_WPMR
**Offset:** 0xE4
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key
See "Register Write Protection" for the list of registers that can be protected.

| Value | Name | Description |
|---|---|---|
| 0x454643 | PASSWD | Writing any other value in this field aborts the write operation.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See "Register Write Protection" for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x454643 (EFC in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x454643 (EFC in ASCII). |

© 2021 Microchip Technology Inc.
and its subsidiaries

# 23. Supply Controller (SUPC)

## 23.1 Description

The Supply Controller (SUPC) controls the supply voltages of the system and manages the Backup mode. In this mode, current consumption is reduced to a few microamps for backup power retention. Exit from this mode is possible on multiple wakeup sources. The SUPC also generates the slow clock by selecting either the slow RC oscillator or the 32.768 kHz crystal oscillator.

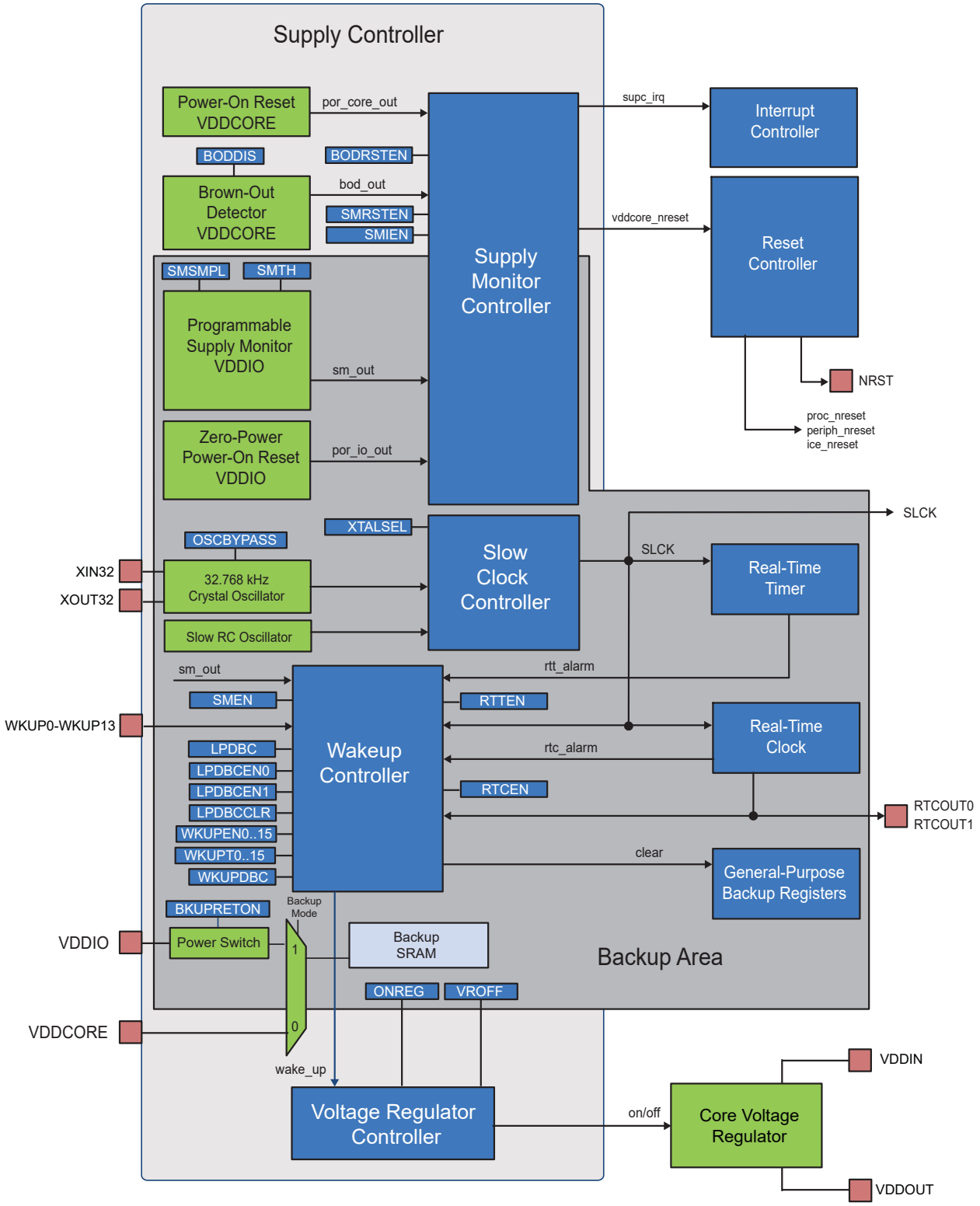
## 23.2 Embedded Characteristics

- Management of the Core Power Supply VDDCORE and Backup Mode via the Embedded Voltage Regulator
- Supply Monitor Detection on VDDIO or a Brownout Detection on VDDCORE Triggers a Core Reset
- Generates the Slow Clock SLCK by selecting either the 22-42 kHz Slow RC Oscillator or the 32.768 kHz Crystal Oscillator
- Backup SRAM
- Low-power Tamper Detection on Two Inputs
- Anti-tampering by Immediate Clear of the General-purpose Backup Registers
- Support of Multiple Wakeup Sources for Exit from Backup Mode
  - 14 Wakeup Inputs with Programmable Debouncing
  - Real-Time Clock Alarm
  - Real-Time Timer Alarm
  - Supply Monitor Detection on VDDIO, with Programmable Scan Period and Voltage Threshold

© 2021 Microchip Technology Inc.
and its subsidiaries

## 23.3 Block Diagram

**Figure 23-1. Supply Controller Block Diagram**



## 23.4 Functional Description

### 23.4.1 Overview

The device is divided into two power supply areas:

- VDDIO power supply: includes the Supply Controller, part of the Reset Controller, the slow clock switch, the general-purpose backup registers, the supply monitor and the clock which includes the Real-time Timer and the Real-time Clock.
- Core power supply: includes part of the Reset Controller, the Brownout Detector, the processor, the SRAM memory, the Flash memory and the peripherals.

The Supply Controller (SUPC) controls the supply voltage of the core power supply. The SUPC intervenes when the VDDIO power supply rises (when the system is starting) or when Backup mode is entered.

The SUPC also integrates the slow clock generator, which is based on a 32.768 kHz crystal oscillator, and a slow RC oscillator. The slow clock defaults to the slow RC oscillator, but the software can enable the 32.768 kHz crystal oscillator and select it as the slow clock source.

The SUPC and the VDDIO power supply have a reset circuitry based on a zero-power power-on reset cell. The zero-power power-on reset allows the SUPC to start correctly as soon as the VDDIO voltage becomes valid.

At startup of the system, once the backup voltage VDDIO is valid and the slow RC oscillator is stabilized, the SUPC starts up the core by sequentially enabling the internal voltage regulator. The SUPC waits until the core voltage VDDCORE is valid, then releases the reset signal of the core vddcore_nreset signal.

Once the system has started, the user can program a supply monitor and/or a brownout detector. If the supply monitor detects a voltage level on VDDIO that is too low, the SUPC asserts the reset signal of the core vddcore_nreset signal until VDDIO is valid. Likewise, if the brownout detector detects a core voltage level VDDCORE that is too low, the SUPC asserts the reset signal vddcore_nreset until VDDCORE is valid.

When Backup mode is entered, the SUPC sequentially asserts the reset signal of the core power supply vddcore_nreset and disables the voltage regulator, in order to supply only the VDDIO power supply. Current consumption is reduced to a few microamps for the backup part retention. Exit from this mode is possible on multiple wakeup sources including an event on WKUP pins, or a clock alarm. To exit this mode, the SUPC operates in the same way as system startup.

### 23.4.2 Slow Clock Generator

The SUPC embeds a slow clock generator that is supplied with the VDDIO power supply. As soon as the VDDIO is supplied, both the 32.768 kHz crystal oscillator and the slow RC oscillator are powered up, but only the slow RC oscillator is enabled. When the slow RC oscillator is selected as the slow clock source, the slow clock stabilizes more quickly than when the 32.768 kHz crystal oscillator is selected.

The user can select the 32.768 kHz crystal oscillator to be the source of the slow clock, as it provides a more accurate frequency than the slow RC oscillator. The 32.768 kHz crystal oscillator is selected by setting the XTALSEL bit in the SUPC Control register (SUPC_CR). The following sequence must be used to switch from the slow RC oscillator to the 32.768 kHz crystal oscillator:

1. The PIO lines multiplexed with XIN32 and XOUT32 are configured to be driven by the oscillator.
2. The 32.768 kHz crystal oscillator is enabled.
3. A number of slow RC oscillator clock periods is counted to cover the startup time of the 32.768 kHz crystal oscillator. Refer to the section “Electrical Characteristics” for information on the 32.768 kHz crystal oscillator startup time.
4. The slow clock is switched to the output of the 32.768 kHz crystal oscillator.
5. The slow RC oscillator is disabled to save power.

The switching time may vary depending on the slow RC oscillator clock frequency range. The switch of the slow clock source is glitch-free. The OSCSEL bit of the SUPC Status register (SUPC_SR) indicates when the switch sequence is finished.

Reverting to the slow RC oscillator as a slow clock source is only possible by shutting down the VDDIO power supply.

If the user does not need the 32.768 kHz crystal oscillator, the XIN32 and XOUT32 pins should be left unconnected.

The user can also set the 32.768 kHz crystal oscillator in Bypass mode instead of connecting a crystal. In this case, the user has to provide the external clock signal on XIN32. The input characteristics of the XIN32 pin are given in the section “Electrical Characteristics”. To enter Bypass mode, the OSCBYPASS bit in the Mode register (SUPC_MR) must be set before setting XTALSEL.

### 23.4.3 Core Voltage Regulator Control/Backup Low-power Mode

The SUPC controls the embedded voltage regulator.

The voltage regulator automatically adapts its quiescent current depending on the required load current. Refer to the section “Electrical Characteristics”.

The user can switch off the voltage regulator, and thus put the device in Backup mode, by writing a ‘1’ to SUPC_CR.VROFF.

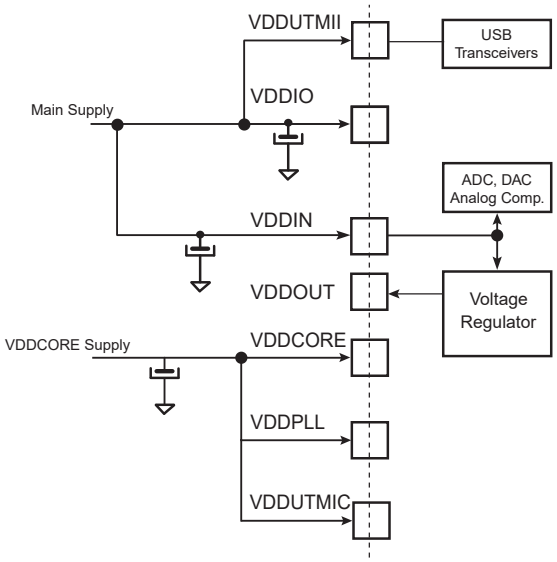
This asserts the vddcore_nreset signal after the write resynchronization time, which lasts two slow clock cycles (worst case). Once the vddcore_nreset signal is asserted, the processor and the peripherals are stopped one slow clock cycle before the core power supply shuts off.

When the internal voltage regulator is not used and VDDCORE is supplied by an external supply, the voltage regulator can be disabled by writing a ‘0’ to SUPC_MR.ONREG.

### 23.4.4 Using Backup Batteries/Backup Supply

When backup batteries or, more generally, a separate backup supply is used, only VDDIO is present in Backup mode. No other external supply is applied.

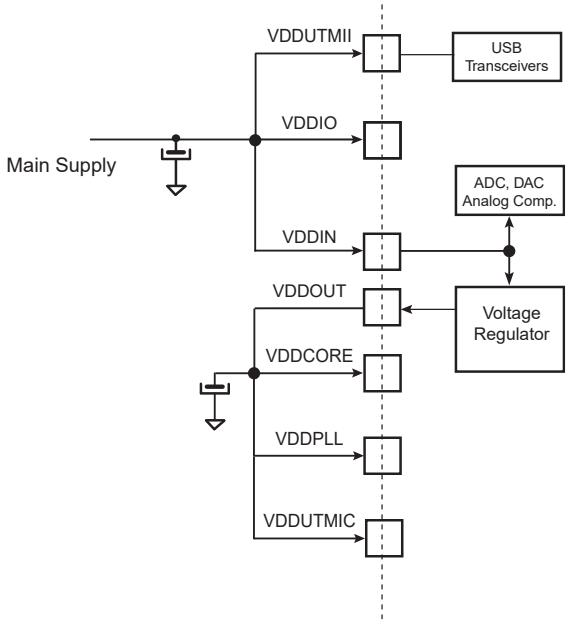
**Figure 23-2. Separate Backup Supply Powering Scheme**




**Note:** Restrictions
With main supply < 3.0V, USB is not usable.
With main supply < 2.7V, MediaLB is not usable.
With main supply < 2.0V, ADC, DAC and Analog comparator are not usable.
With main supply and VDDIN > 3V, all peripherals are usable.

When no separate backup supply for VDDIO is used, since the external voltage applied on VDDIO is kept, all of the I/O configurations (i.e., WKUP pin configuration) are maintained in Backup mode. When not using backup batteries, VDDIORDY is set so the user does not need to program it.

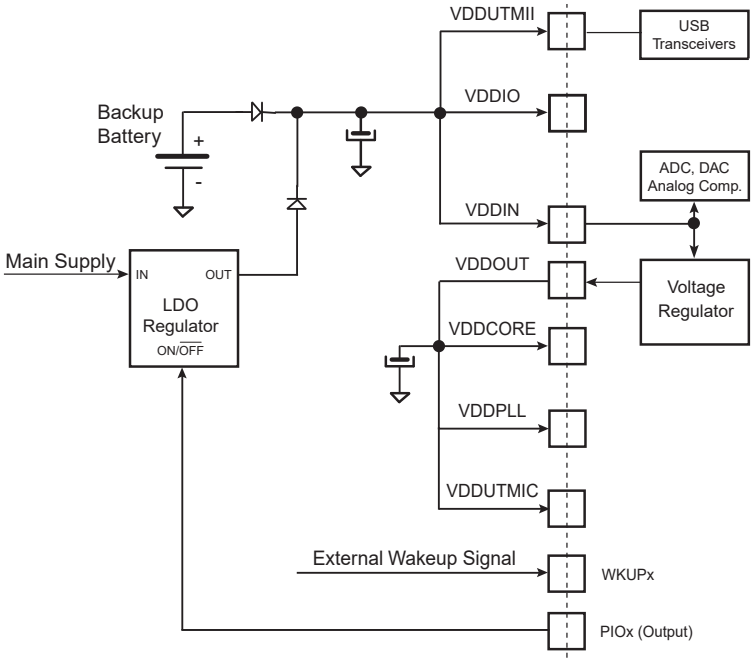
**Figure 23-3. No Separate Backup Supply Powering Scheme**




**Note:** Restrictions
with main supply < 2.0 V, USB and ADC/DAC and analog comparator are not usable.
With main supply > 2.0V and < 3V, USB is not usable.
With main supply < 2.7V, MediaLB is not usable.
With main supply > 3V, all peripherals are usable.

The following figure illustrates an example of the powering scheme when using a backup battery. Since the PIO state is preserved when in Backup mode, any free PIO line can be used to switch off the external regulator by driving the PIO line at low level (PIO is input, pull-up enabled after backup reset). System wakeup can be performed using a wakeup pin (WKUPx). See the "Wakeup Sources" section for further details.

**Figure 23-4. Battery Backup**




Note: The two diodes provide a “switchover circuit” between the backup battery and the main supply when the system is put in Backup mode.

### 23.4.5 Supply Monitor

The SUPC embeds a supply monitor located in the VDDIO power supply and which monitors VDDIO power supply.

The supply monitor can be used to prevent the processor from falling into an unpredictable state if the main power supply drops below a certain level.

The threshold of the supply monitor is programmable in the SMTH field of the Supply Monitor Mode register (SUPC_SMMR). Refer to the section “Electrical Characteristics”.

The supply monitor can also be enabled during one slow clock period on every one of either 32, 256 or 2048 slow clock periods, depending on the user selection. This is configured in the SUPC_SMMR.SMSMPL.

Enabling the supply monitor for such reduced times divides the typical supply monitor power consumption by factors of 2, 16 and 128, respectively, if continuous monitoring of the VDDIO power supply is not required.

A supply monitor detection generates either a reset of the core power supply or a wakeup of the core power supply. Generating a core reset when a supply monitor detection occurs is enabled by setting SUPC_SMMR.SMRSTEN.

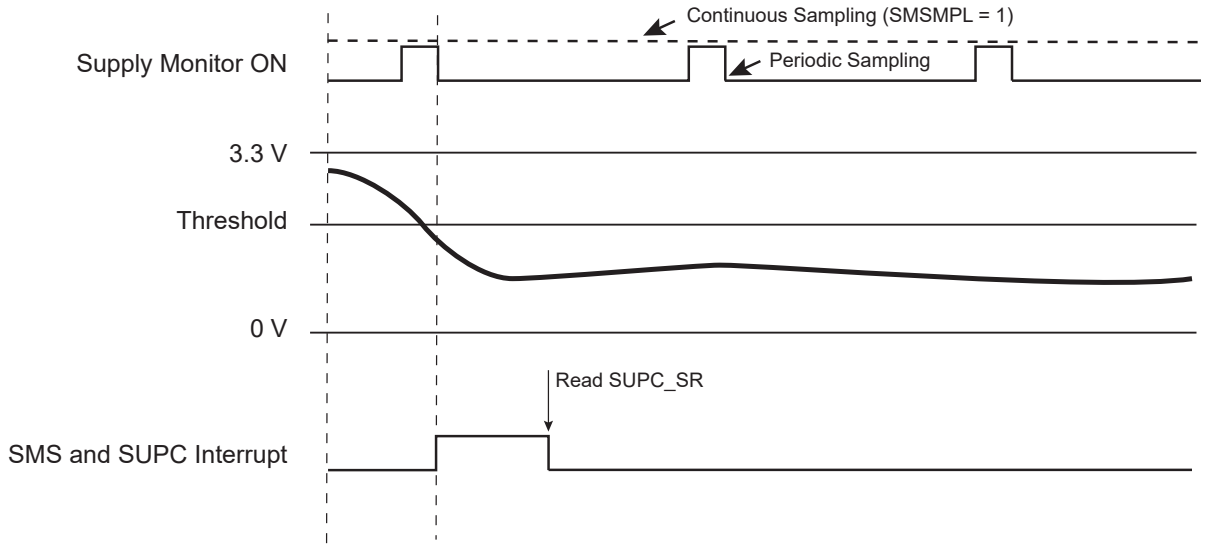
Waking up the core power supply when a supply monitor detection occurs can be enabled by setting the SMEN bit in the Wakeup Mode register (SUPC_WUMR).

The SUPC provides two status bits in the SUPC_SR for the supply monitor that determine whether the last wakeup was due to the supply monitor:

- SUPC_SR.SMOS provides real-time information, updated at each measurement cycle or updated at each slow clock cycle, if the measurement is continuous.
- SUPC_SR.SMS provides saved information and shows a supply monitor detection has occurred since the last read of SUPC_SR.

The SMS flag generates an interrupt if SUPC_SMMR.SMIEN is set.

**Figure 23-5. Supply Monitor Status Bit and Associated Interrupt**



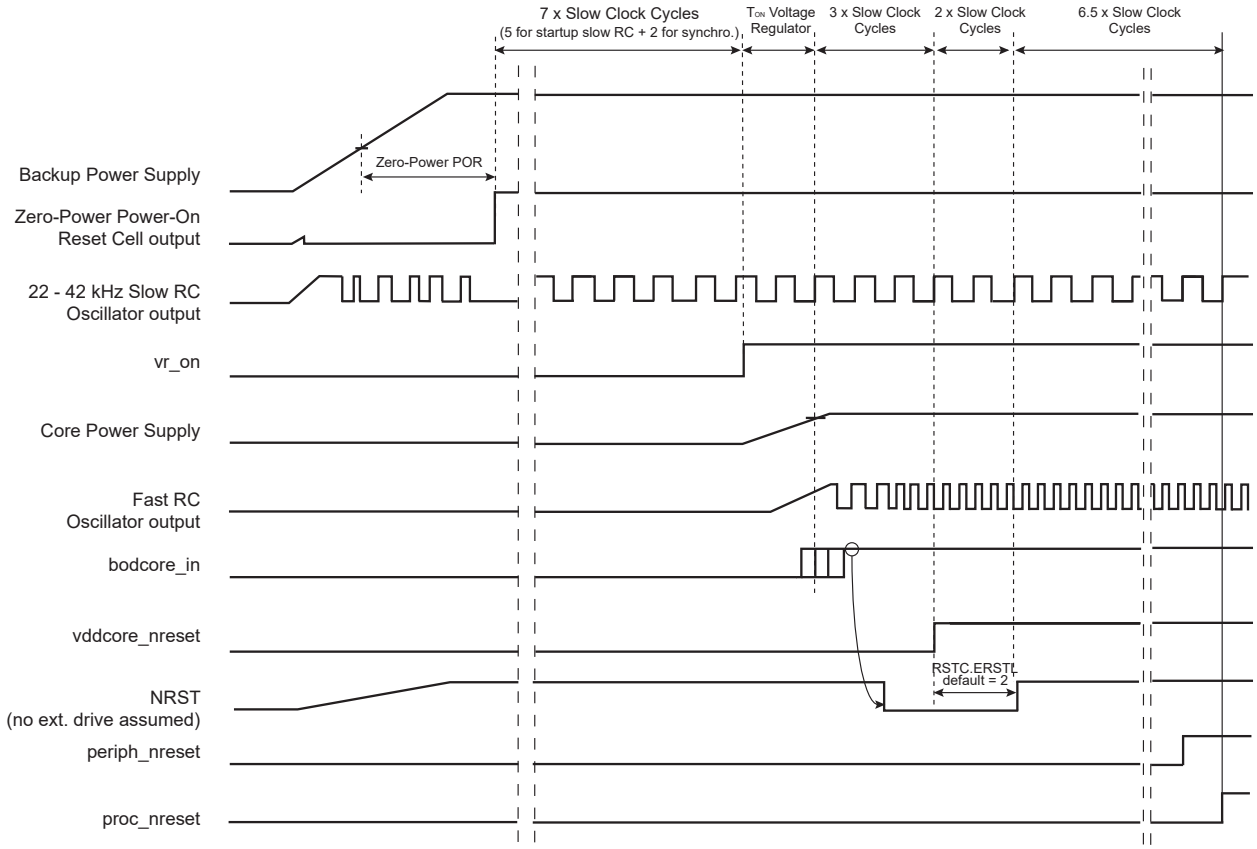

### 23.4.6 Backup Power Supply Reset

#### 23.4.6.1 Raising the Backup Power Supply

When the backup voltage VDDIO rises, the slow RC oscillator is powered up and the zero-power power-on reset cell maintains its output low as long as VDDIO has not reached its target voltage. During this period, the SUPC is reset. When the VDDIO voltage becomes valid and the zero-power power-on reset signal is released, a counter is started for five slow clock cycles. This is the time required for the slow RC oscillator to stabilize.

After this time, the voltage regulator is enabled. The core power supply rises and the brownout detector provides the bodcore_in signal as soon as the core voltage VDDCORE is valid. This results in releasing the vddcore_nreset signal to the Reset Controller after the bodcore_in signal has been confirmed as being valid for at least one slow clock cycle.

**Figure 23-6. Raising the VDDIO Power Supply**




Note: After “proc_nreset” rising, the core starts fetching instructions from Flash.

### 23.4.7 Core Reset

The Supply Controller manages the vddcore_nreset signal to the Reset Controller, as described in the "Backup Power Supply Reset" section. The vddcore_nreset signal is normally asserted before shutting down the core power supply and released as soon as the core power supply is correctly regulated.

There are two additional sources which can be programmed to activate vddcore_nreset:

- a supply monitor detection
- a brownout detection

#### 23.4.7.1 Supply Monitor Reset

The supply monitor is capable of generating a reset of the system. This is enabled by setting SUPC_SMMR.SMRSTEN.

If SUPC_SMMR.SMRSTEN is set and if a supply monitor detection occurs, the vddcore_nreset signal is immediately activated for a minimum of one slow clock cycle.

#### 23.4.7.2 Brownout Detector Reset

The brownout detector provides the bodcore_in signal to the SUPC. This signal indicates that the voltage regulation is operating as programmed. If this signal is lost for longer than 1 slow clock period while the voltage regulator is enabled, the SUPC asserts vddcore_nreset if SUPC_MR.BODRSTEN is written to ‘1’.

If SUPC_MR.BODRSTEN is set and the voltage regulation is lost (output voltage of the regulator too low), the vddcore_nreset signal is asserted for a minimum of one slow clock cycle and then released if bodcore_in has been reactivated. SUPC_SR.BODRSTS indicates the source of the last reset.

Until bodcore_in is deactivated, the vddcore_nreset signal remains active.

### 23.4.8 Controlling the SRAM Power Supply

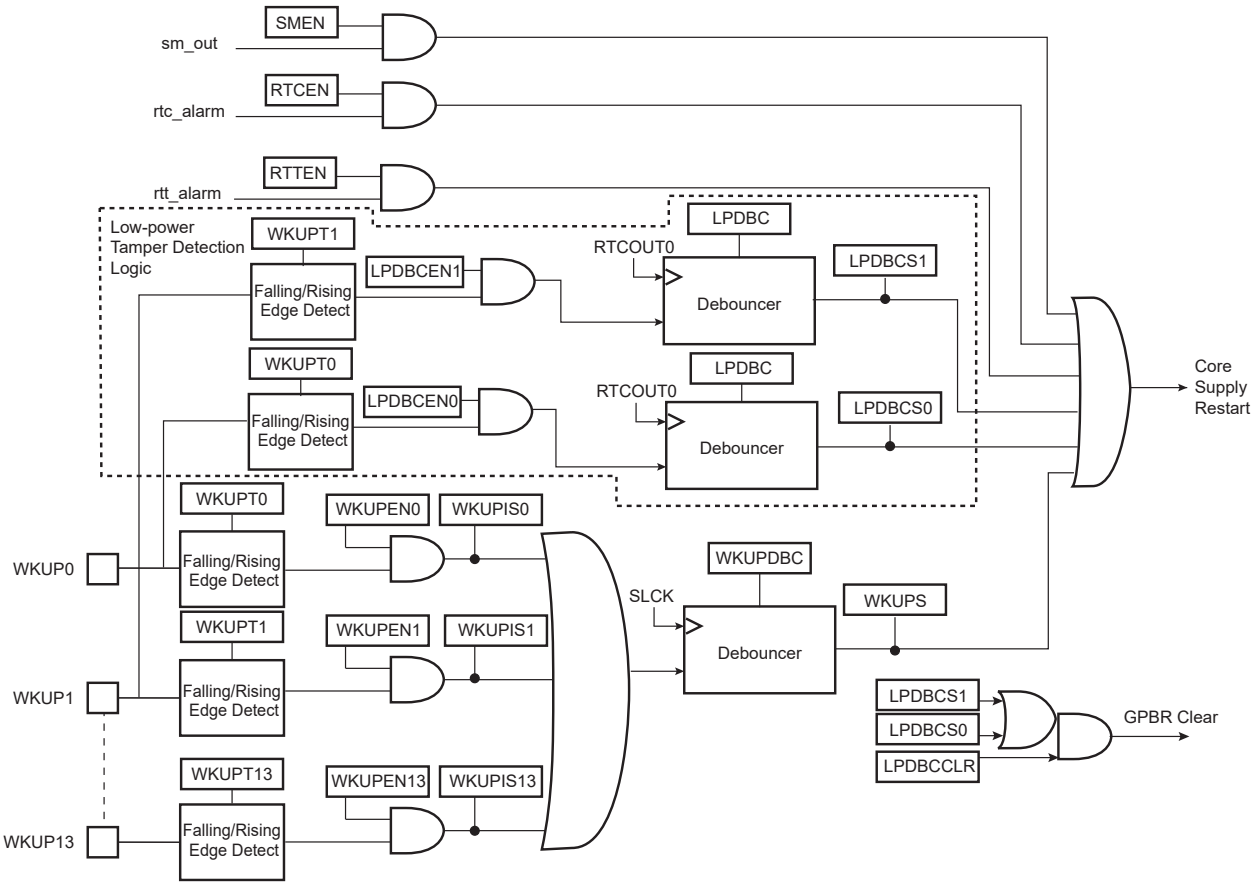
The SUPC can be used to switch on or off the power supply of the backup SRAM by opening or closing the SRAM power switch. This power switch is controlled by SUPC_MR.BKUPRETON. However, the battery backup SRAM is automatically switched on when the core power supply is enabled, as the processor requires the SRAM as data memory space.

- If SUPC_MR.BKUPRETON is written to ‘1’, there is no immediate effect, but the SRAM will be left powered when the SUPC enters Backup mode, thus retaining its content.
- If SUPC_MR.BKUPRETON is written to ‘0’, there is no immediate effect, but the SRAM will be switched off when the SUPC enters Backup mode. The SRAM is automatically switched on when Backup mode is exited.

### 23.4.9 Wakeup Sources

The wakeup events allow the device to exit Backup mode. When a wakeup event is detected, the SUPC performs a sequence that automatically reenables the core power supply.

**Figure 23-7. Wakeup Sources**




#### 23.4.9.1 Wakeup Inputs

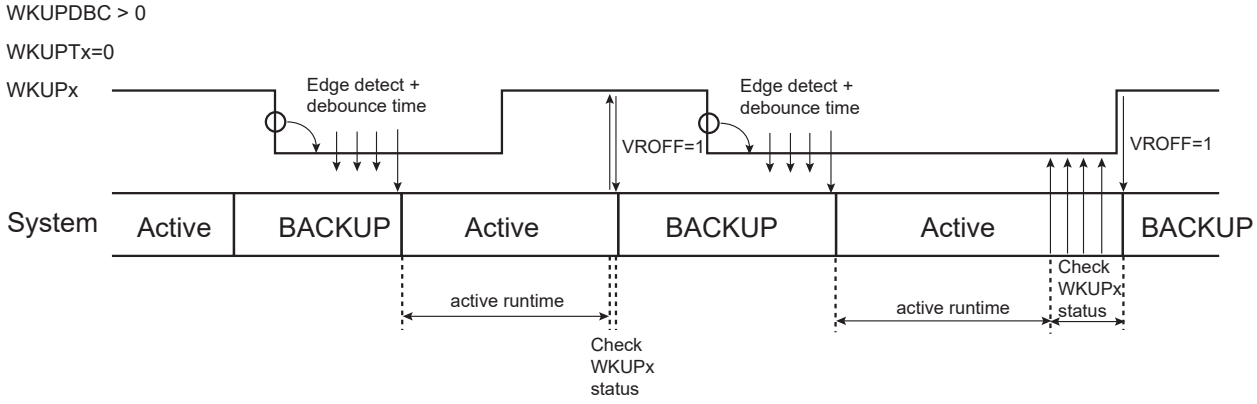
The wakeup inputs, WKUPx, can be programmed to perform a wakeup of the core power supply. Each input can be enabled by writing a ‘1’ to the corresponding bit, WKUPENx, in the Wakeup Inputs register (SUPC_WUIR). The wakeup level can be selected with the corresponding polarity bit, WKUPTx, also located in SUPC_WUIR.

The resulting signals are wired-ORed to trigger a debounce counter, which is programmed with SUPC_WUMR.WKUPDBC. This field selects a debouncing period of 3, 32, 512, 4,096 or 32,768 slow clock cycles. The duration of these periods corresponds, respectively, to about 100 µs, about 1 ms, about 16 ms, about 128 ms and about 1 second (for a typical slow clock frequency of 32 kHz). Programming SUPC_WUMR.WKUPDBC to 0 selects an immediate wakeup, i.e., an enabled WKUP pin must be active according to its polarity during a minimum of one slow clock period to wake up the core power supply.

If an enabled WKUP pin is asserted for a duration longer than the debouncing period, a wakeup of the core power supply is started and the signals, WKUP0 to WKUPx as shown in "Wakeup Sources", are latched in SUPC_SR. This allows the user to identify the source of the wakeup. However, if a new wakeup condition occurs, the primary information is lost. No new wakeup can be detected since the primary wakeup condition has disappeared.

Before instructing the system to enter Backup mode, if the field SUPC_WUMR.WKUPDBC > 0, it must be checked that none of the WKUPx pins that are enabled for a wakeup (exit from Backup mode) holds an active polarity. This is checked by reading the pin status in the PIO Controller. If SUPC_WUIR.WKUPENx=1 and the pin WKUPx holds an active polarity, the system must not be instructed to enter Backup mode.

**Figure 23-8. Entering and Exiting Backup Mode with a WKUP Pin**




### 23.4.9.2 Low-power Tamper Detection and Anti-Tampering

Low-power debouncer inputs (WKUP0, WKUP1) can be used for tamper detection. If the tamper sensor is biased through a resistor and constantly driven by the power supply, this leads to power consumption as long as the tamper detection switch is in its active state. To prevent power consumption when the switch is in active state, the tamper sensor circuitry must be intermittently powered, and thus a specific waveform must be applied to the sensor circuitry.

The waveform is generated using RTCOUTx in all modes including Backup mode. Refer to the section "Real-Time Clock (RTC)" for waveform generation.

Separate debouncers are embedded, one for WKUP0 input, one for WKUP1 input.

The WKUP0 and/or WKUP1 inputs perform a system wakeup upon tamper detection. This is enabled by setting SUPC_WUMR.LPDBCEN0/1.

WKUP0 and/or WKUP1 inputs can also be used when VDDCORE is powered to detect a tamper.

When SUPC_WUMR.LPDBCENx is written to '1', WKUPx pins must not be configured to act as a debouncing source for the WKUPDBC counter (WKUPENx must be cleared in SUPC_WUIR).

Low-power tamper detection or debounce requires RTC output (RTCOUTx) to be configured to generate a duty cycle programmable pulse (i.e., OUT0 = 0x7 in RTC_MR) in order to create the sampling points of both debouncers. The sampling point is the falling edge of the RTCOUTx waveform.

The following figure shows an example of an application where two tamper switches are used. RTCOUTx powers the external pull-up used by the tamper sensor circuitry.

**Figure 23-9. Low-power Debouncer (Push-to-Make Switch, Pull-up Resistors)**

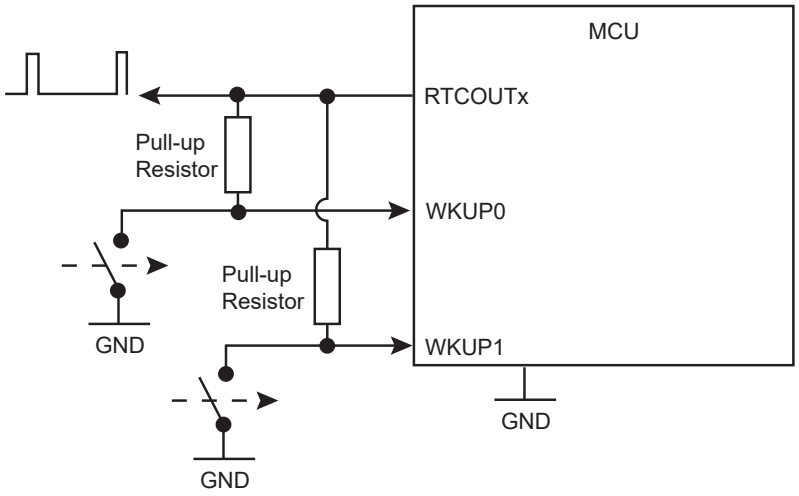
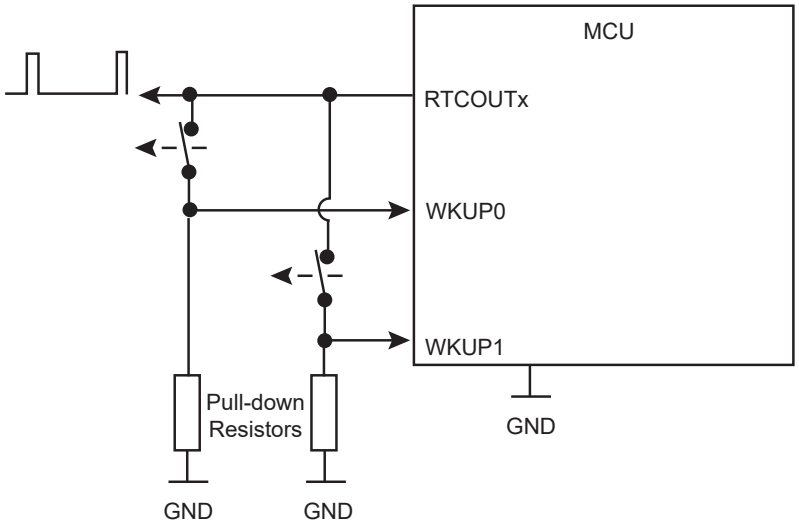



**Figure 23-10. Low-power Debouncer (Push-to-Break Switch, Pull-down Resistors)**




The debouncing period duration is configurable. The period is set for all debouncers (i.e., the duration cannot be adjusted for each debouncer). The number of successive identical samples to wake up the system can be configured from 2 up to 8 in SUPC_WUMR.LPDBC. The period of time between two samples can be configured by programming RTC_MR.TPERIOD. Power parameters can be adjusted by modifying the period of time in RTC_MR.THIGH.

The wakeup polarity of the inputs can be independently configured by writing SUPC_WUMR.WKUPT0 and/ or SUPC_WUMR.WKUPT1.

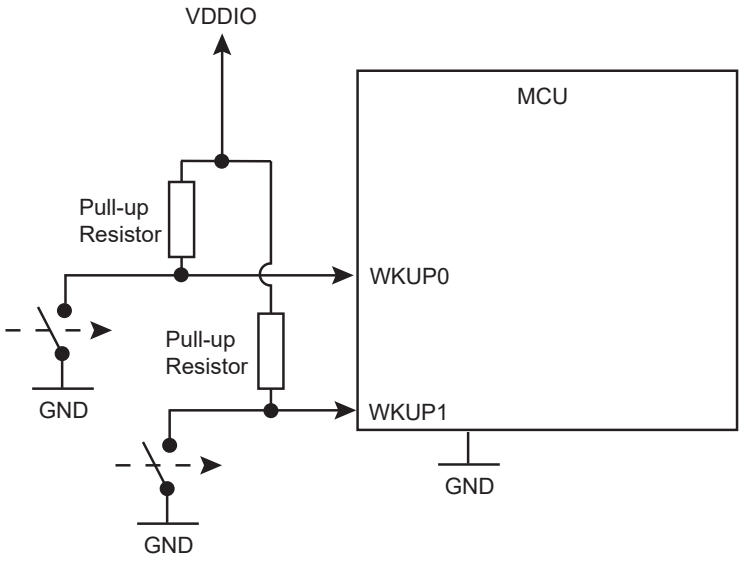
In order to determine which wakeup/tamper pin triggers the system wakeup, a status flag is associated for each low-power debouncer. These flags are read in SUPC_SR.

A debounce event (tamper detection) can perform an immediate clear (0 delay) on the first half the general-purpose backup registers (GPBR). SUPC_WUMR.LPDBCCLR bit must be set.

Note that it is not mandatory to use the RTCOUTx pin when using the WKUP0/WKUP1 pins as tampering inputs in any mode. Using the RTCOUTx pin provides a “sampling mode” to further reduce the power consumption of the tamper detection circuitry. If RTCOUTx is not used, the RTC must be configured to create an internal sampling point for the debouncer logic. The period of time between two samples can be configured by programming RTC_MR.TPERIOD.

The following figure illustrates the use of WKUPx without the RTCOUTx pin.

**Figure 23-11. Using WKUP Pins Without RTCOUTx Pins**




**Related Links**

27. Real-time Clock (RTC)

### 23.4.9.3 Clock Alarms

The RTC and the RTT alarms can generate a wakeup of the core power supply. This can be enabled by setting, respectively, SUPC_WUMR.RTCEN and SUPC_WUMR.RTTEN.

The Supply Controller does not provide any status as the information is available in the user interface of either the Real-Time Timer or the Real-Time Clock.

### 23.4.9.4 Supply Monitor Detection

The supply monitor can generate a wakeup of the core power supply. See "Supply Monitor".

## 23.4.10 Register Write Protection

To prevent any single software error from corrupting SYSC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the "System Controller Write Protection Mode Register" (SYSC_WPMR).

The following registers can be write-protected:

- RSTC Mode Register(1)
- RTT Mode Register(2)
- RTT Alarm Register(2)
- RTC Control Register(3)
- RTC Mode Register(3)
- RTC Time Alarm Register(3)
- RTC Calendar Alarm Register(3)
- General Purpose Backup Registers(4)
- Supply Controller Control Register
- Supply Controller Supply Monitor Mode Register
- Supply Controller Mode Register
- Supply Controller Wakeup Mode Register
- Supply Controller Wakeup Inputs Register

**Notes:**

1. See the section "Reset Controller (RSTC)".
2. See the section "Real Time Timer (RTT)".
3. See the section "Real Time Clock (RTC)".
4. See the section "General Purpose Backup Registers (GPBR)".

### 23.4.11 Register Bits in Backup Domain (VDDIO)

The following configuration registers, or certain bits of the registers, are physically located in the product backup domain:

- RSTC Mode Register (all bits)(1)
- RTT Mode Register (all bits)(2)
- RTT Alarm Register (all bits)(2)
- RTC Control Register (all bits)(3)
- RTC Mode Register (all bits)(3)
- RTC Time Alarm Register (all bits)(3)
- RTC Calendar Alarm Register (all bits)(3)
- General Purpose Backup Registers (all bits)(4)
- Supply Controller Control Register (see register description for details)
- Supply Controller Supply Monitor Mode Register (all bits)
- Supply Controller Mode Register (see register description for details)
- Supply Controller Wakeup Mode Register (all bits)
- Supply Controller Wakeup Inputs Register (all bits)
- Supply Controller Status Register (all bits)

**Notes:**

1. See the section "Reset Controller (RSTC)".
2. See the section "Real Time Timer (RTT)".
3. See the section "Real Time Clock (RTC)".
4. See the section "General Purpose Backup Registers (GPBR)".

© 2021 Microchip Technology Inc.
and its subsidiaries

## 23.5 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SUPC_CR | 7:0 | | | | | XTALSEL | VROFF | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |
| 0x04 | SUPC_SMMR | 7:0 | | | | | SMTH[3:0] | | | |
| | | 15:8 | | | SMIEN | SMRSTEN | | SMSMPL[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | SUPC_MR | 7:0 | | | | | | | | |
| | | 15:8 | | ONREG | BODDIS | BODRSTEN | | | | |
| | | 23:16 | | | | OSCBYPASS | | | BKUPRETON | |
| | | 31:24 | KEY[7:0] | | | | | | | |
| 0x0C | SUPC_WUMR | 7:0 | LPDBCCLR | LPDBCEN1 | LPDBCEN0 | | RTCEN | RTTEN | SMEN | |
| | | 15:8 | | WKUPDBC[2:0] | | | | | | |
| | | 23:16 | | | | | | LPDBC[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x10 | SUPC_WUIR | 7:0 | WKUPEN[7:0] | | | | | | | |
| | | 15:8 | | | WKUPEN[13:8] | | | | | |
| | | 23:16 | WKUPT[7:0] | | | | | | | |
| | | 31:24 | | | WKUPT[13:8] | | | | | |
| 0x14 | SUPC_SR | 7:0 | OSCSEL | SMOS | SMS | SMRSTS | BODRSTS | SMWS | WKUPS | |
| | | 15:8 | | LPDBCS1 | LPDBCS0 | | | | | |
| | | 23:16 | WKUPIS[7:0] | | | | | | | |
| | | 31:24 | | | WKUPIS[13:8] | | | | | |
| 0x18 ... 0xD3 | Reserved | | | | | | | | | |
| 0xD4 | SYSC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 23.5.1 Supply Controller Control Register

**Name:** SUPC_CR
**Offset:** 0x00
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | XTALSEL | VROFF | | |
| Access | | | | | W | W | | |
| Reset | | | | | | | | |

**Bits 31:24 – KEY[7:0]** Password

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 3 – XTALSEL** Crystal Oscillator Select
Note: This bit is located in the VDDIO domain.

| Value | Description |
|---|---|
| 0 | (NO_EFFECT): No effect. |
| 1 | (CRYSTAL_SEL): If KEY is correct, XTALSEL switches the slow clock on the 32.768 kHz crystal oscillator output. |

**Bit 2 – VROFF** Voltage Regulator Off
Note: This bit is located in the VDDIO domain.

| Value | Description |
|---|---|
| 0 | (NO_EFFECT): No effect. |
| 1 | (STOP_VREG): If KEY is correct, VROFF asserts the vddcore_nreset and stops the voltage regulator. |

### 23.5.2 Supply Controller Supply Monitor Mode Register

**Name:** SUPC_SMMR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register is located in the VDDIO domain.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | SMIEN | SMRSTEN | | SMSMPL[2:0] | | |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SMTH[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 13 – SMIEN** Supply Monitor Interrupt Enable

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): The SUPC interrupt signal is not affected when a supply monitor detection occurs. |
| 1 | (ENABLE): The SUPC interrupt signal is asserted when a supply monitor detection occurs. |

**Bit 12 – SMRSTEN** Supply Monitor Reset Enable

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): The core reset signal vddcore_nreset is not affected when a supply monitor detection occurs. |
| 1 | (ENABLE): The core reset signal, vddcore_nreset is asserted when a supply monitor detection occurs. |

**Bits 10:8 – SMSMPL[2:0]** Supply Monitor Sampling Period

| Value | Name | Description |
|---|---|---|
| 0x0 | SMD | Supply Monitor disabled |
| 0x1 | CSM | Continuous Supply Monitor |
| 0x2 | 32SLCK | Supply Monitor enabled one SLCK period every 32 SLCK periods |
| 0x3 | 256SLCK | Supply Monitor enabled one SLCK period every 256 SLCK periods |
| 0x4 | 2048SLCK | Supply Monitor enabled one SLCK period every 2,048 SLCK periods |

**Bits 3:0 – SMTH[3:0]** Supply Monitor Threshold
Selects the threshold voltage of the supply monitor. Refer to the section “Electrical Characteristics” for voltage values.

### 23.5.3 Supply Controller Mode Register

**Name:** SUPC_MR
**Offset:** 0x08
**Reset:** 0x00005A00
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | OSCBYPASS | | | BKUPRETON | |
| Access | | | | R/W | | | R/W | |
| Reset | | | | 0 | | | 0 | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ONREG | BODDIS | BODRSTEN | | | | |
| Access | | R/W | R/W | R/W | | | | |
| Reset | | 1 | 0 | 1 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:24 – KEY[7:0]** Password Key

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 20 – OSCBYPASS** Oscillator Bypass
**Note:** This bit is located in the VDDIO domain.

| Value | Description |
|---|---|
| 0 | (NO_EFFECT): No effect. Clock selection depends on the value of SUPC_CR.XTALSEL. |
| 1 | (BYPASS): The 32.768 kHz crystal oscillator is bypassed if SUPC_CR.XTALSEL is set. OSCBYPASS must be set prior to setting XTALSEL. |

**Bit 17 – BKUPRETON** SRAM On In Backup Mode

| Value | Description |
|---|---|
| 0 | SRAM (Backup) switched off in Backup mode. |
| 1 | SRAM (Backup) switched on in Backup mode.<br>Note: This bit is located in the VDDIO domain. |

**Bit 14 – ONREG** Voltage Regulator Enable
**Note:** This bit is located in the VDDIO domain.

| Value | Description |
|---|---|
| 0 | (ONREG_UNUSED): Internal voltage regulator is not used (external power supply is used). |
| 1 | (ONREG_USED): Internal voltage regulator is used. |

**Bit 13 – BODDIS** Brownout Detector Disable
**Note:** This bit is located in the VDDIO domain.

| Value | Description |
|---|---|
| 0 | (ENABLE): The core brownout detector is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | (DISABLE): The core brownout detector is disabled. |

**Bit 12 – BODRSTEN** Brownout Detector Reset Enable
**Note:** This bit is located in the VDDIO domain.

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): The core reset signal vddcore_nreset is not affected when a brownout detection occurs. |
| 1 | (ENABLE): The core reset signal, vddcore_nreset is asserted when a brownout detection occurs. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 23.5.4 Supply Controller Wakeup Mode Register

**Name:** SUPC_WUMR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read/Write

This register is located in the VDDIO domain.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | R/W | R/W | R/W | | | | |
| Reset | | 0 | 0 | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LPDBCCLR | LPDBCEN1 | LPDBCEN0 | | RTCEN | RTTEN | SMEN | |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | | 0 | 0 | 0 | |

**Bits 18:16 – LPDBC[2:0]** Low-power Debouncer Period

| Value | Name | Description |
|---|---|---|
| 0 | DISABLE | Disables the low-power debouncers. |
| 1 | 2_RTCOUT | WKUP0/1 in active state for at least 2 RTCOUTx clock periods |
| 2 | 3_RTCOUT | WKUP0/1 in active state for at least 3 RTCOUTx clock periods |
| 3 | 4_RTCOUT | WKUP0/1 in active state for at least 4 RTCOUTx clock periods |
| 4 | 5_RTCOUT | WKUP0/1 in active state for at least 5 RTCOUTx clock periods |
| 5 | 6_RTCOUT | WKUP0/1 in active state for at least 6 RTCOUTx clock periods |
| 6 | 7_RTCOUT | WKUP0/1 in active state for at least 7 RTCOUTx clock periods |
| 7 | 8_RTCOUT | WKUP0/1 in active state for at least 8 RTCOUTx clock periods |

**Bits 14:12 – WKUPDBC[2:0]** Wakeup Inputs Debouncer Period

| Value | Name | Description |
|---|---|---|
| 0 | IMMEDIATE | Immediate, no debouncing, detected active at least on one Slow Clock edge. |
| 1 | 3_SLCK | WKUPx shall be in its active state for at least 3 SLCK periods |
| 2 | 32_SLCK | WKUPx shall be in its active state for at least 32 SLCK periods |
| 3 | 512_SLCK | WKUPx shall be in its active state for at least 512 SLCK periods |
| 4 | 4096_SLCK | WKUPx shall be in its active state for at least 4,096 SLCK periods |
| 5 | 32768_SLCK | WKUPx shall be in its active state for at least 32,768 SLCK periods |

**Bit 7 – LPDBCCLR** Low-power Debouncer Clear

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): A low-power debounce event does not create an immediate clear on the first half of GPBR registers. |
| 1 | (ENABLE): A low-power debounce event on WKUP0 or WKUP1 generates an immediate clear on the first half of GPBR registers. |

**Bit 6 – LPDBCEN1** Low-power Debouncer Enable WKUP1

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): The WKUP1 input pin is not connected to the low-power debouncer. |
| 1 | (ENABLE): The WKUP1 input pin is connected to the low-power debouncer and forces a system wakeup. |

**Bit 5 – LPDBCEN0** Low-power Debouncer Enable WKUP0

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): The WKUP0 input pin is not connected to the low-power debouncer. |
| 1 | (ENABLE): The WKUP0 input pin is connected to the low-power debouncer and forces a system wakeup. |

**Bit 3 – RTCEN** Real-time Clock Wakeup Enable

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): The RTC alarm signal has no wakeup effect. |
| 1 | (ENABLE): The RTC alarm signal forces the wakeup of the core power supply. |

**Bit 2 – RTTEN** Real-time Timer Wakeup Enable

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): The RTT alarm signal has no wakeup effect. |
| 1 | (ENABLE): The RTT alarm signal forces the wakeup of the core power supply. |

**Bit 1 – SMEN** Supply Monitor Wakeup Enable

| Value | Description |
|---|---|
| 0 | (NOT_ENABLE): The supply monitor detection has no wakeup effect. |
| 1 | (ENABLE): The supply monitor detection forces the wakeup of the core power supply. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 23.5.5 Supply Controller Wakeup Inputs Register

**Name:** SUPC_WUIR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This register is located in the VDDIO domain. This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | WKUPT[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WKUPT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | WKUPEN[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WKUPEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 29:16 – WKUPT[13:0]** Wakeup Input Type ('x' = 0-13)

| Value | Description |
|---|---|
| 0 | (LOW): A falling edge followed by a low level for a period defined by WKUPDBC on the corresponding wakeup input forces the wakeup of the core power supply. |
| 1 | (HIGH): A rising edge followed by a high level for a period defined by WKUPDBC on the corresponding wakeup input forces the wakeup of the core power supply. |

**Bits 13:0 – WKUPEN[13:0]** Wakeup Input Enablex ('x' = 0-13)

| Value | Description |
|---|---|
| 0 | (DISABLE): The corresponding wakeup input has no wakeup effect. |
| 1 | (ENABLE): The corresponding wakeup input is enabled for a wakeup of the core power supply. |

### 23.5.6 Supply Controller Status Register

**Name:** SUPC_SR
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read-only

**Note:** Because of the asynchronism between the Slow Clock (SLCK) and the System Clock (MCK), the status register flag reset is taken into account only 2 slow clock cycles after the read of the SUPC_SR.

This register is located in the VDDIO domain.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | WKUPIS[13:8] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WKUPIS[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | LPDBCS1 | LPDBCS0 | | | | | |
| Access | | R | R | | | | | |
| Reset | | 0 | 0 | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OSCSEL | SMOS | SMS | SMRSTS | BODRSTS | SMWS | WKUPS | |
| Access | R | R | R | R | R | R | R | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bits 29:16 – WKUPIS[13:0]** WKUPx ('x' = 0-13) Input Status (cleared on read)

| Value | Description |
|---|---|
| 0 | (DIS): The corresponding wakeup input is disabled, or was inactive at the time the debouncer triggered a wakeup event. |
| 1 | (EN): The corresponding wakeup input was active at the time the debouncer triggered a wakeup event since the last read of SUPC_SR. |

**Bit 14 – LPDBCS1** Low-power Debouncer Wakeup Status on WKUP1 (cleared on read)

| Value | Description |
|---|---|
| 0 | (NO): No wakeup due to the assertion of the WKUP1 pin has occurred since the last read of SUPC_SR. |
| 1 | (PRESENT): At least one wakeup due to the assertion of the WKUP1 pin has occurred since the last read of SUPC_SR. |

**Bit 13 – LPDBCS0** Low-power Debouncer Wakeup Status on WKUP0 (cleared on read)

| Value | Description |
|---|---|
| 0 | (NO): No wakeup due to the assertion of the WKUP0 pin has occurred since the last read of SUPC_SR. |
| 1 | (PRESENT): At least one wakeup due to the assertion of the WKUP0 pin has occurred since the last read of SUPC_SR. |

**Bit 7 – OSCSEL** 32-kHz Oscillator Selection Status

| Value | Description |
|---|---|
| 0 | (RC): The slow clock, SLCK, is generated by the slow RC oscillator. |
| 1 | (CRYST): The slow clock, SLCK, is generated by the 32.768 kHz crystal oscillator. |

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 6 – SMOS** Supply Monitor Output Status

| Value | Description |
|---|---|
| 0 | (HIGH): The supply monitor detected VDDIO higher than its threshold at its last measurement. |
| 1 | (LOW): The supply monitor detected VDDIO lower than its threshold at its last measurement. |

**Bit 5 – SMS** Supply Monitor Status (cleared on read)

| Value | Description |
|---|---|
| 0 | (NO): No supply monitor detection since the last read of SUPC_SR. |
| 1 | (PRESENT): At least one supply monitor detection since the last read of SUPC_SR. |

**Bit 4 – SMRSTS** Supply Monitor Reset Status (cleared on read)

| Value | Description |
|---|---|
| 0 | (NO): No supply monitor detection has generated a core reset since the last read of the SUPC_SR. |
| 1 | (PRESENT): At least one supply monitor detection has generated a core reset since the last read of the SUPC_SR. |

**Bit 3 – BODRSTS** Brownout Detector Reset Status (cleared on read)
When the voltage remains below the defined threshold, there is no rising edge event at the output of the brownout detection cell. The rising edge event occurs only when there is a voltage transition below the threshold.

| Value | Description |
|---|---|
| 0 | (NO): No core brownout rising edge event has been detected since the last read of the SUPC_SR. |
| 1 | (PRESENT): At least one brownout output rising edge event has been detected since the last read of the SUPC_SR. |

**Bit 2 – SMWS** Supply Monitor Detection Wakeup Status (cleared on read)

| Value | Description |
|---|---|
| 0 | (NO): No wakeup due to a supply monitor detection has occurred since the last read of SUPC_SR. |
| 1 | (PRESENT): At least one wakeup due to a supply monitor detection has occurred since the last read of SUPC_SR. |

**Bit 1 – WKUPS** WKUP Wakeup Status (cleared on read)

| Value | Description |
|---|---|
| 0 | (NO): No wakeup due to the assertion of the WKUP pins has occurred since the last read of SUPC_SR. |
| 1 | (PRESENT): At least one wakeup due to the assertion of the WKUP pins has occurred since the last read of SUPC_SR. |

### 23.5.7 System Controller Write Protection Mode Register

**Name:** SYSC_WPMR
**Offset:** 0xD4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R?W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key.

| Value | Name | Description |
|---|---|---|
| 0x525443 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See "Register Write Protection" for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x525443 (“RTC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x525443 (“RTC” in ASCII). |

© 2021 Microchip Technology Inc.
and its subsidiaries

# 24. Watchdog Timer (WDT)

## 24.1 Description

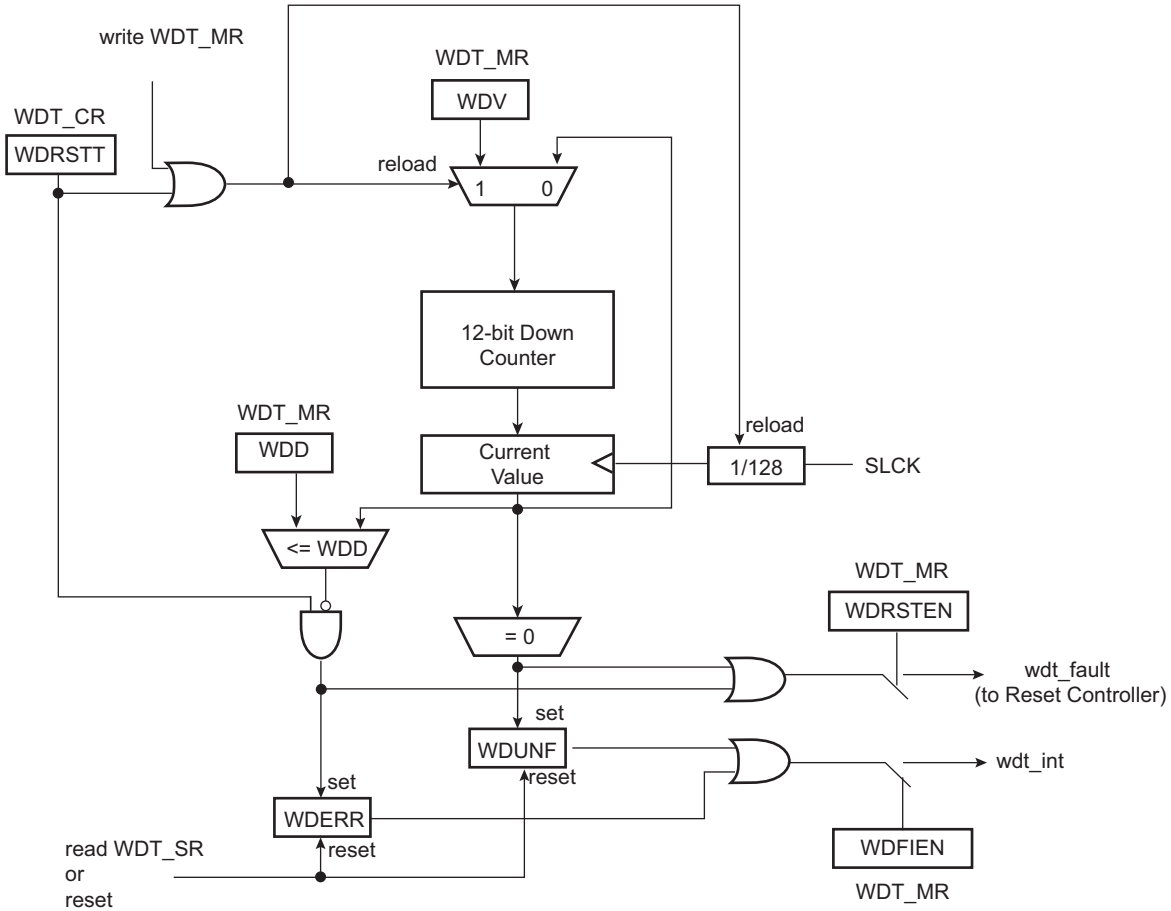
The Watchdog Timer (WDT) is used to prevent system lock-up if the software becomes trapped in a deadlock. It features a 12-bit down counter that allows a watchdog period of up to 16 seconds (slow clock around 32 kHz). It can generate a general reset or a processor reset only. In addition, it can be stopped while the processor is in Debug mode or Sleep mode (Idle mode).

## 24.2 Embedded Characteristics

- 12-bit Key-protected Programmable Counter
- Watchdog Clock is Independent from Processor Clock
- Provides Reset or Interrupt Signals to the System
- Counter May Be Stopped while the Processor is in Debug State or in Idle Mode

## 24.3 Block Diagram

**Figure 24-1. Watchdog Timer Block Diagram**



## 24.4 Functional Description

The Watchdog Timer is used to prevent system lock-up if the software becomes trapped in a deadlock. It is supplied with VDDCORE. It restarts with initial values on processor reset.

The watchdog is built around a 12-bit down counter, which is loaded with the value defined in the field WDV of the Mode Register (WDT_MR). The Watchdog Timer uses the slow clock divided by 128 to establish the maximum watchdog period to be 16 seconds (with a typical slow clock of 32.768 kHz).

After a processor reset, the value of WDV is 0xFFF, corresponding to the maximum value of the counter with the external reset generation enabled (field WDRSTEN at 1 after a backup reset). This means that a default watchdog is running at reset, i.e., at power-up. The user can either disable the WDT by setting bit WDT_MR.WDDIS or reprogram the WDT to meet the maximum watchdog period the application requires.

When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

If the watchdog is restarted by writing into the Control Register (WDT_CR), WDT_MR must not be programmed during a period of time of three slow clock periods following the WDT_CR write access. In any case, programming a new value in WDT_MR automatically initiates a restart instruction.

WDT_MR can be written only once. Only a processor reset resets it. Writing WDT_MR reloads the timer with the newly programmed mode parameters.

In normal operation, the user reloads the watchdog at regular intervals before the timer underflow occurs, by setting bit WDT_CR.WDRSTT. The watchdog counter is then immediately reloaded from WDT_MR and restarted, and the slow clock 128 divider is reset and restarted. WDT_CR is write-protected. As a result, writing WDT_CR without the correct hard-coded key has no effect. If an underflow does occur, the “wdt_fault” signal to the Reset Controller is asserted if bit WDT_MR.WDRSTEN is set. Moreover, the bit WDUNF is set in the Status Register (WDT_SR).

The reload of the watchdog must occur while the watchdog counter is within a window between 0 and WDD. WDD is defined in WDT_MR.

Any attempt to restart the watchdog while the watchdog counter is between WDV and WDD results in a watchdog error, even if the watchdog is disabled. The bit WDT_SR.WDERR is updated and the “wdt_fault” signal to the Reset Controller is asserted.

Note that this feature can be disabled by programming a WDD value greater than or equal to the WDV value. In such a configuration, restarting the Watchdog Timer is permitted in the whole range [0; WDV] and does not generate an error. This is the default configuration on reset (the WDD and WDV values are equal).

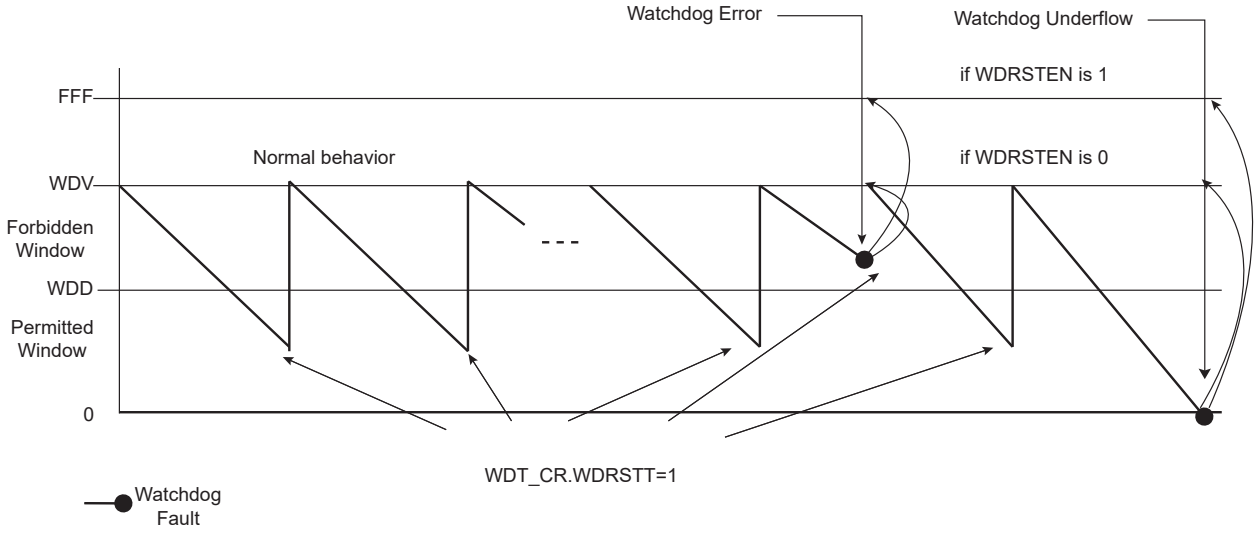
The status bits WDUNF (Watchdog Underflow) and WDERR (Watchdog Error) trigger an interrupt, provided the bit WDT_MR.WDFIEN is set. The signal “wdt_fault” to the Reset Controller causes a watchdog reset if the WDRSTEN bit is set as already explained in the Reset Controller documentation. In this case, the processor and the Watchdog Timer are reset, and the WDERR and WDUNF flags are reset.

If a reset is generated or if WDT_SR is read, the status bits are reset, the interrupt is cleared, and the “wdt_fault” signal to the reset controller is deasserted.

Writing WDT_MR reloads and restarts the down counter.

While the processor is in debug state or in Sleep mode, the counter may be stopped depending on the value programmed for the bits WDIDLEHLT and WDDBGHLT in WDT_MR.

**Figure 24-2. Watchdog Behavior**



## 24.5 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | WDT_CR | 7:0 | | | | | | | | WDRSTT |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |
| 0x04 | WDT_MR | 7:0 | WDV[7:0] | | | | | | | |
| | | 15:8 | WDDIS | | WDRSTEN | WDFIEN | WDV[11:8] | | | |
| | | 23:16 | WDD[7:0] | | | | | | | |
| | | 31:24 | | | WDIDLEHLT | WDDBGHLT | WDD[11:8] | | | |
| 0x08 | WDT_SR | 7:0 | | | | | | | WDERR | WDUNF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 24.5.1 Watchdog Timer Control Register

**Name:** WDT_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

The WDT_CR register values must not be modified within three slow clock periods following a restart of the watchdog performed by a write access in WDT_CR. Any modification will cause the watchdog to trigger an end of period earlier than expected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WDRSTT |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bits 31:24 – KEY[7:0]** Password

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 0 – WDRSTT** Watchdog Restart

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Restarts the watchdog if KEY is written to 0xA5. |

### 24.5.2 Watchdog Timer Mode Register

**Name:** WDT_MR
**Offset:** 0x04
**Reset:** 0x3FFF2FFF
**Property:** Read/Write Once

The first write access prevents any further modification of the value of this register. Read accesses remain possible.

The WDT_MR register values must not be modified within three slow clock periods following a restart of the watchdog performed by a write access in WDT_CR. Any modification will cause the watchdog to trigger an end of period earlier than expected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | WDIDLEHLT | WDDBGHLT | WDD[11:8] | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WDD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WDDIS | | WDRSTEN | WDFIEN | WDV[11:8] | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 1 | 0 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WDV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 29 – WDIDLEHLT** Watchdog Idle Halt

| Value | Description |
|---|---|
| 0 | The watchdog runs when the system is in idle state. |
| 1 | The watchdog stops when the system is in idle state. |

**Bit 28 – WDDBGHLT** Watchdog Debug Halt

| Value | Description |
|---|---|
| 0 | The watchdog runs when the processor is in debug state. |
| 1 | The watchdog stops when the processor is in debug state. |

**Bits 27:16 – WDD[11:0]** Watchdog Delta Value
Defines the permitted range for reloading the Watchdog Timer.
If the Watchdog Timer value is less than or equal to WDD, setting bit WDT_CR.WDRSTT restarts the timer.
If the Watchdog Timer value is greater than WDD, setting bit WDT_CR.WDRSTT causes a watchdog error.

**Bit 15 – WDDIS** Watchdog Disable
When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

| Value | Description |
|---|---|
| 0 | Enables the Watchdog Timer. |
| 1 | Disables the Watchdog Timer. |

**Bit 13 – WDRSTEN** Watchdog Reset Enable

| Value | Description |
|---|---|
| 0 | A watchdog fault (underflow or error) has no effect on the resets. |

| Value | Description |
|---|---|
| 1 | A watchdog fault (underflow or error) triggers a watchdog reset. |

**Bit 12 – WDFIEN** Watchdog Fault Interrupt Enable

| Value | Description |
|---|---|
| 0 | A watchdog fault (underflow or error) has no effect on interrupt. |
| 1 | A watchdog fault (underflow or error) asserts interrupt. |

**Bits 11:0 – WDV[11:0]** Watchdog Counter Value
Defines the value loaded in the 12-bit watchdog counter.

### 24.5.3 Watchdog Timer Status Register

**Name:** WDT_SR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | WDERR | WDUNF |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – WDERR** Watchdog Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No watchdog error occurred since the last read of WDT_SR. |
| 1 | At least one watchdog error occurred since the last read of WDT_SR. |

**Bit 0 – WDUNF** Watchdog Underflow (cleared on read)

| Value | Description |
|---|---|
| 0 | No watchdog underflow occurred since the last read of WDT_SR. |
| 1 | At least one watchdog underflow occurred since the last read of WDT_SR. |

# 25. Reinforced Safety Watchdog Timer (RSWDT)

## 25.1 Description

The Reinforced Safety Watchdog Timer (RSWDT) works in parallel with the Watchdog Timer (WDT) to reinforce safe watchdog operations.

The RSWDT can be used to reinforce the safety level provided by the WDT in order to prevent system lock-up if the software becomes trapped in a deadlock. The RSWDT works in a fully operable mode, independent of the WDT. Its clock source is automatically selected from either the Slow RC oscillator clock, or from the Main RC oscillator divided clock to get an equivalent Slow RC oscillator clock. If the WDT clock source (for example, the 32 kHz crystal oscillator) fails, the system lock-up is no longer monitored by the WDT because the RSWDT performs the monitoring. Thus, there is no lack of safety regardless of the external operating conditions. The RSWDT shares the same features as the WDT (i.e., a 12-bit down counter that allows a watchdog period of up to 16 seconds with slow clock at 32.768 kHz). It can generate a general reset or a processor reset only. In addition, it can be stopped while the processor is in Debug mode or Idle mode.

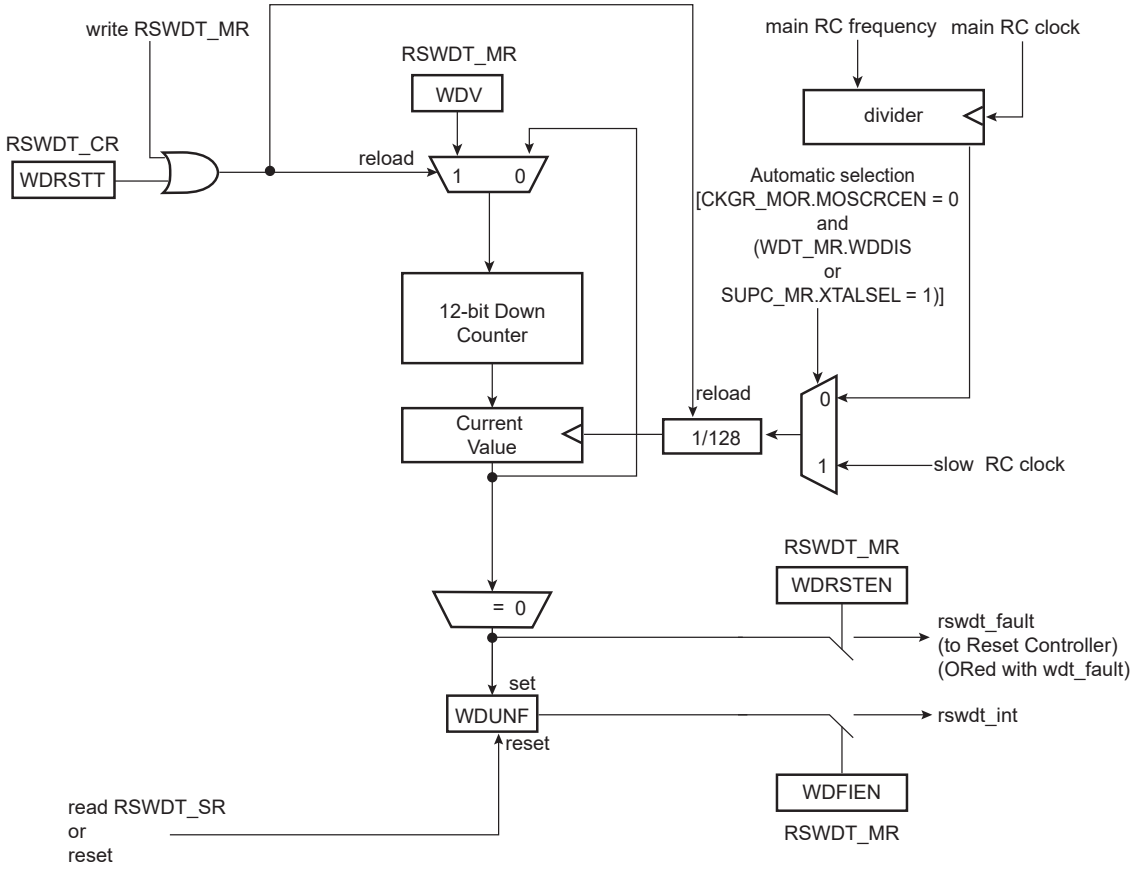
## 25.2 Embedded Characteristics

- Automatically Selected Reliable RSWDT Clock Source (independent of WDT clock source)
- 12-bit Key-protected Programmable Counter
- Provides Reset or Interrupt Signals to the System
- Counter may be Stopped While Processor is in Debug State or Idle Mode

© 2021 Microchip Technology Inc.
and its subsidiaries

## 25.3 Block Diagram

**Figure 25-1. Reinforced Safety Watchdog Timer Block Diagram**




## 25.4 Functional Description

The RSWDT is supplied by VDDCORE. The RSWDT is initialized with default values on processor reset or on a power-on sequence and is disabled (its default mode) under such conditions.

The RSWDT must not be enabled if the WDT is disabled.

The Main RC oscillator divided clock is selected if the Main RC oscillator is already enabled by the application (CKGR_MOR.MOSCRCEN = 1) or if the WDT is driven by the Slow RC oscillator.

The RSWDT is built around a 12-bit down counter, which is loaded with a slow clock value other than that of the slow clock in the WDT, defined in the WDV (Watchdog Counter Value) field of the Mode Register (RSWDT_MR). The RSWDT uses the slow clock divided by 128 to establish the maximum watchdog period to be 16 seconds (with a typical slow clock of 32.768 kHz).

After a processor reset, the value of RSWDT_MR.WDV is 0xFFF, corresponding to the maximum value of the counter with the external reset generation enabled (RSWDT_MR.WDRSTEN = 1 after a backup reset). This means that a default watchdog is running at reset, i.e., at power-up.

If the watchdog is restarted by writing into the Control Register (RSWDT_CR), the RSWDT_MR must not be programmed during a period of time of three slow clock periods following the RSWDT_CR write access. Programming a new value in the RSWDT_MR automatically initiates a restart instruction.

RSWDT_MR can be written only once. Only a processor reset resets it. Writing RSWDT_MR reloads the timer with the newly programmed mode parameters.

In normal operation, the user reloads the watchdog at regular intervals before the timer underflow occurs, by setting bit RSWDT_CR.WDRSTT. The watchdog counter is then immediately reloaded from the RSWDT_MR and restarted, and the slow clock 128 divider is reset and restarted. The RSWDT_CR is write-protected. As a result, writing RSWDT_CR without the correct hard-coded key has no effect. If an underflow does occur, the “wdt_fault” signal to the Reset Controller is asserted if RSWDT_MR.WDRSTEN is set. Moreover, WDUNF (Watchdog Underflow) is set in the Status Register (RSWDT_SR).

The status bits WDUNF and WDERR trigger an interrupt, provided the WDFIEN bit is set in the RSWDT_MR. The signal “wdt_fault” to the Reset Controller causes a Watchdog reset if the WDRSTEN bit. For details, refer to the section “Reset Controller (RSTC)”. In this case, the processor and the RSWDT are reset, and the WDUNF and WDERR flags are reset.

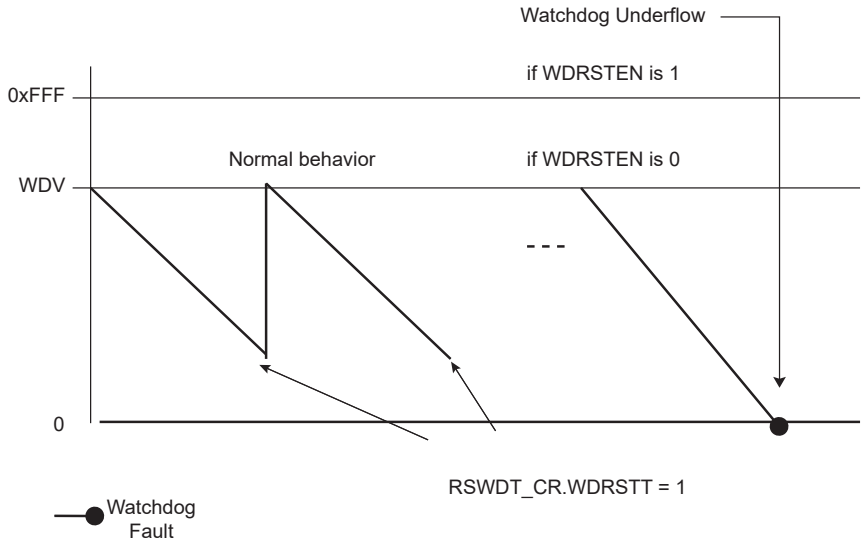
If a reset is generated, or if RSWDT_SR is read, the status bits are reset, the interrupt is cleared, and the “wdt_fault” signal to the reset controller is deasserted

Writing RSWDT_MR reloads and restarts the down counter.

The RSWDT is disabled after any power-on sequence.

While the processor is in Debug state or in Idle mode, the counter may be stopped depending on the value programmed for the WDIDLEHLT and WDDBGHLT bits in the RSWDT_MR.

**Figure 25-2. Watchdog Behavior**




**Related Links**

26. Reset Controller (RSTC)

## 25.5 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | RSWDT_CR | 7:0 | | | | | | | | WDRSTT |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |
| 0x04 | RSWDT_MR | 7:0 | WDV[7:0] | | | | | | | |
| | | 15:8 | WDDIS | | WDRSTEN | WDFIEN | WDV[11:8] | | | |
| | | 23:16 | ALLONES[7:0] | | | | | | | |
| | | 31:24 | | | WDIDLEHLT | WDDBGHLT | ALLONES[11:8] | | | |
| 0x08 | RSWDT_SR | 7:0 | | | | | | | | WDUNF |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 25.5.1 Reinforced Safety Watchdog Timer Control Register

**Name:** RSWDT_CR
**Offset:** 0x00
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WDRSTT |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:24 – KEY[7:0]** Password

| Value | Name | Description |
|---|---|---|
| 0xC4 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 0 – WDRSTT** Watchdog Restart

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Restarts the watchdog. |

### 25.5.2 Reinforced Safety Watchdog Timer Mode Register

**Name:** RSWDT_MR
**Offset:** 0x04
**Reset:** 0x3FFFAFFF
**Property:** Read/Write Once

**Note:** The first write access prevents any further modification of the value of this register; read accesses remain possible. The WDV value must not be modified within three slow clock periods following a restart of the watchdog performed by means of a write access in the RSWDT_CR, else the watchdog may trigger an end of period earlier than expected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | WDIDLEHLT | WDDBGHLT | ALLONES[11:8] | | | |
| Access | | | | | | | | |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ALLONES[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WDDIS | | WDRSTEN | WDFIEN | WDV[11:8] | | | |
| Access | | | | | | | | |
| Reset | 1 | | 1 | 0 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WDV[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 29 – WDIDLEHLT** Watchdog Idle Halt

| Value | Description |
|---|---|
| 0 | The RSWDT runs when the system is in idle mode. |
| 1 | The RSWDT stops when the system is in idle state. |

**Bit 28 – WDDBGHLT** Watchdog Debug Halt

| Value | Description |
|---|---|
| 0 | The RSWDT runs when the processor is in debug state. |
| 1 | The RSWDT stops when the processor is in debug state. |

**Bits 27:16 – ALLONES[11:0]** Must Always Be Written with 0xFFF

**Bit 15 – WDDIS** Watchdog Disable

| Value | Description |
|---|---|
| 0 | Enables the RSWDT. |
| 1 | Disables the RSWDT. |

**Bit 13 – WDRSTEN** Watchdog Reset Enable

| Value | Description |
|---|---|
| 0 | A Watchdog fault (underflow or error) has no effect on the resets. |
| 1 | A Watchdog fault (underflow or error) triggers a watchdog reset. |

**Bit 12 – WDFIEN** Watchdog Fault Interrupt Enable

| Value | Description |
|---|---|
| 0 | A Watchdog fault (underflow or error) has no effect on interrupt. |

| Value | Description |
|---|---|
| 1 | A Watchdog fault (underflow or error) asserts interrupt. |

**Bits 11:0 – WDV[11:0]** Watchdog Counter Value
Defines the value loaded in the 12-bit watchdog counter.

### 25.5.3 Reinforced Safety Watchdog Timer Status Register

**Name:** RSWDT_SR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WDUNF |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bit 0 – WDUNF** Watchdog Underflow

| Value | Description |
|---|---|
| 0 | No watchdog underflow occurred since the last read of RSWDT_SR. |
| 1 | At least one watchdog underflow occurred since the last read of RSWDT_SR. |

# 26. Reset Controller (RSTC)

## 26.1 Description

The Reset Controller (RSTC), driven by Power-On Reset (POR) cells, software, external reset pin and peripheral events, handles all the resets of the system without any external components. It reports which reset occurred last.

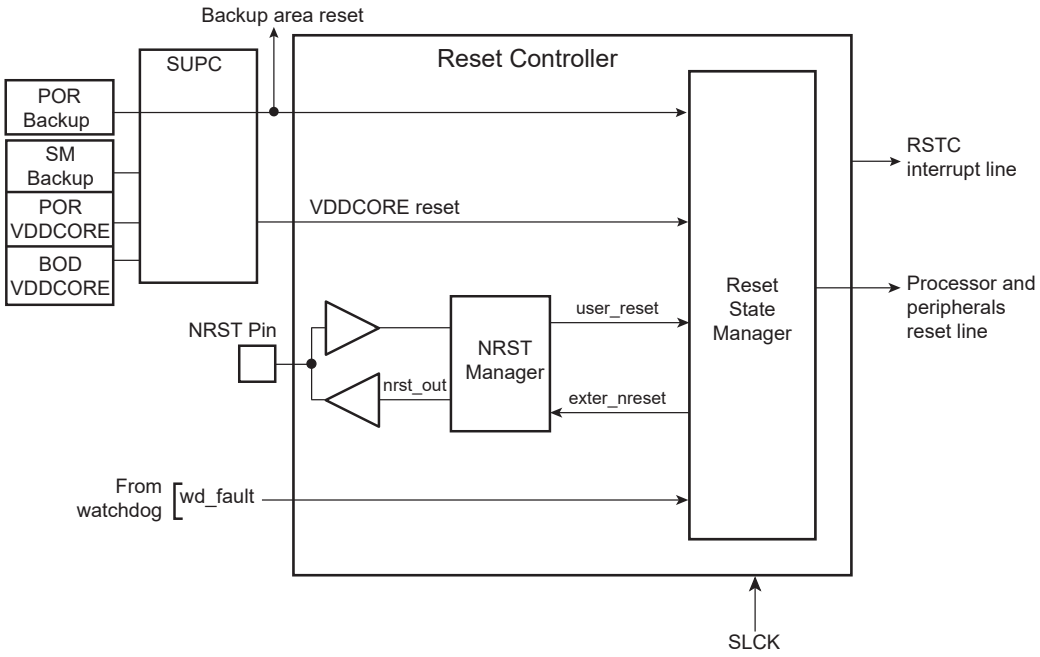
The RSTC also drives simultaneously the external reset and the peripheral and processor resets.

## 26.2 Embedded Characteristics

- Driven by embedded POR, software, external reset pin and peripheral events
- Management of all system resets, including:
  - External devices through the NRST pin
  - Processor
  - Peripheral set
- Reset source status:
  - Status of the last reset
  - Either VDDCORE and VDDIO POR, Software Reset, User Reset, Watchdog Reset
- External reset signal control and shaping

## 26.3 Block Diagram

**Figure 26-1. Reset Controller Block Diagram**



## 26.4 Functional Description

### 26.4.1 Overview

The RSTC is made up of an NRST manager and a reset state manager. It runs at SLCK frequency and generates the following reset signals:

- proc_nreset: Processor reset line (also resets the Watchdog Timer)
- periph_nreset: Affects the whole set of embedded peripherals
- nrst_out: Drives the NRST pin

**Note:** proc_nreset and periph_nreset are driven in the same way.

These reset signals are asserted by the RSTC, either on events generated by peripherals, events on the NRST pin, or on a software action. The reset state manager controls the generation of reset signals and provides a signal to the NRST manager when an assertion of the NRST pin is required.

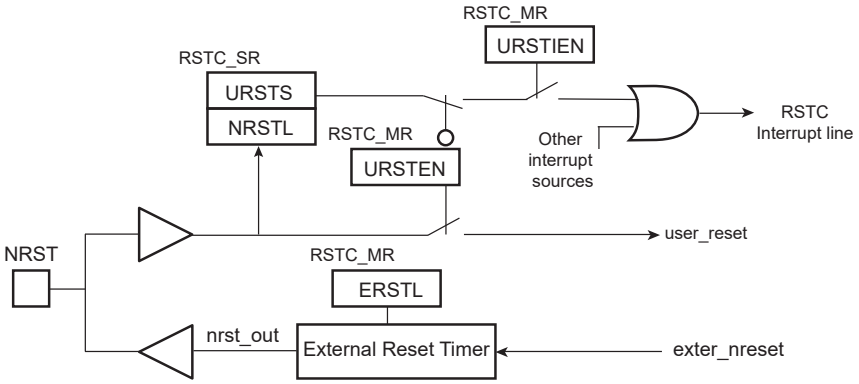
The NRST manager shapes the NRST assertion during a programmable time, thus controlling external device resets.

The RSTC Mode register (RSTC_MR), used to configure the RSTC, is powered with VDDIO, so that its configuration is saved as long as VDDIO is on.

### 26.4.2 NRST Manager

The NRST manager samples the NRST input pin and drives this pin low when required by the reset state manager. The figure below shows the block diagram of the NRST manager.

**Figure 26-2. NRST Manager**




#### 26.4.2.1 NRST Signal or Interrupt

The NRST manager samples the NRST pin at SLCK speed. When the NRST line is low for more than three clock cycles, a User Reset is reported to the reset state manager. The NRST pin must be asserted for at least 1 SLCK clock cycle to ensure execution of a user reset.

However, the NRST manager can be programmed to not trigger a reset when an assertion of NRST occurs. Writing a ‘0’ to RSTC_MR.URSTEN disables the User Reset trigger.

The level of the pin NRST can be read at any time in the bit NRSTL in the RSTC Status Register (RSTC_SR). As soon as the NRST pin is asserted, RSTC_SR. URSTS is written to ‘1’. This bit is cleared only when the RSTC_SR is read.

The RSTC can also be programmed to generate an interrupt instead of generating a reset. To do so, RSTC_MR.URSTIEN must be set.

#### 26.4.2.2 NRST External Reset Control

The reset state manager asserts the signal exter_nreset to assert the NRST pin. When this occurs, the “nrst_out” signal is driven low by the NRST manager for a time programmed by RSTC_MR.ERSTL. This assertion duration, named External Reset Length, lasts $2^{(ERSTL+1)}$ SLCK cycles. This gives the approximate duration of an assertion between 60 µs and 2 seconds. Note that ERSTL at ‘0’ defines a two-cycle duration for the NRST pulse.

This feature allows the RSTC to shape the NRST pin level, and thus to guarantee that the NRST line is driven low for a time compliant with potential external devices connected on the system reset.

RSTC_MR is backed up, making it possible to use the value of ERSTL to shape the system powerup reset for devices requiring a longer startup time than that of the MCU.

### 26.4.3 Reset States

The reset state manager handles the different reset sources and generates the internal reset signals. It reports the reset status in RSTTYP of the Status Register (RSTC_SR). The update of RSTC_SR.RSTTYP is performed when the processor reset is released.

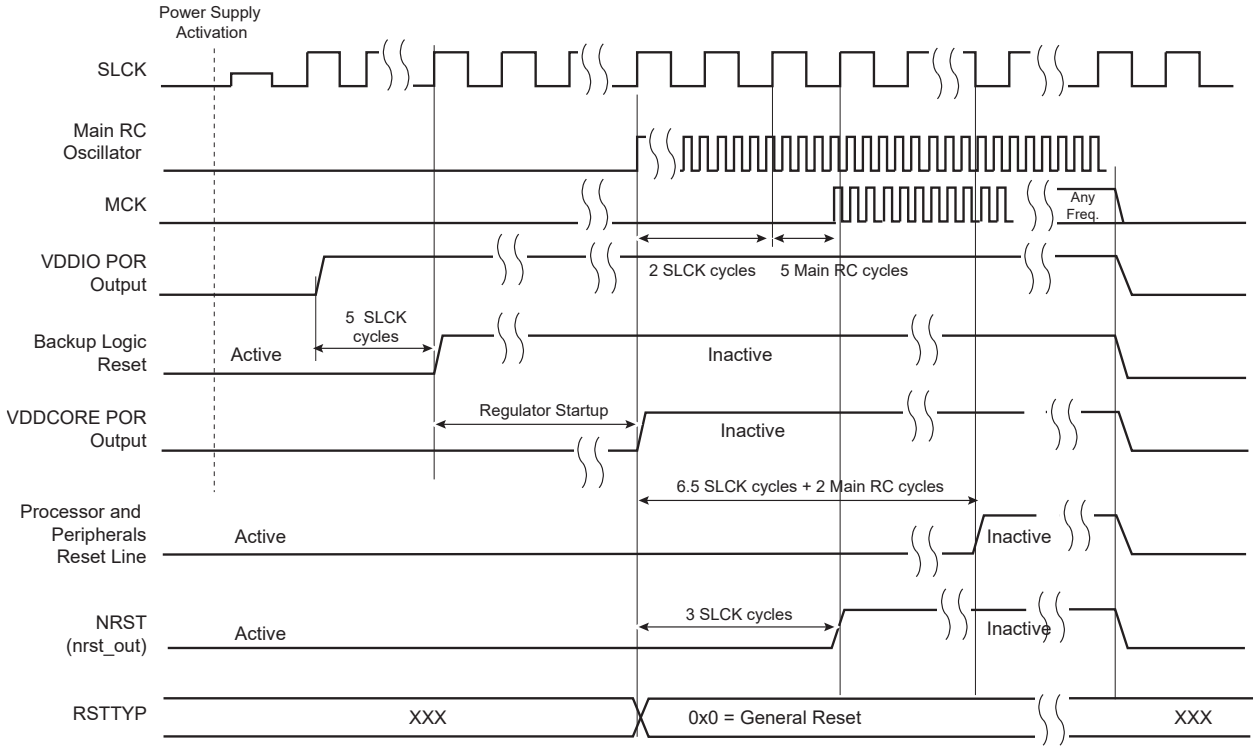
#### 26.4.3.1 General Reset

A general reset occurs when a VDDIO POR is detected, a brown out or a voltage regulation loss is detected by the Supply Controller. The vddcore_nreset signal is asserted by the Supply Controller when a general reset occurs.

All the reset signals are released and RSTC_SR.RSTTYP reports a general reset. As the RSTC_MR is written to ‘0’, the NRST line rises two cycles after the vddcore_nreset, as ERSTL defaults at value 0x0.

The figure below ilustrates how the general reset affects the reset signals.

**Figure 26-3. General Reset Timing Diagram**




#### 26.4.3.2 Backup Reset

A backup reset occurs when the chip exits from Backup mode. While exiting Backup mode, the vddcore_nreset signal is asserted by the Supply Controller.

Field RSTC_SR.RSTTYP is updated to report a backup reset.

#### 26.4.3.3 Watchdog Reset

The watchdog reset is entered when a watchdog fault occurs. This reset lasts three SLCK cycles.

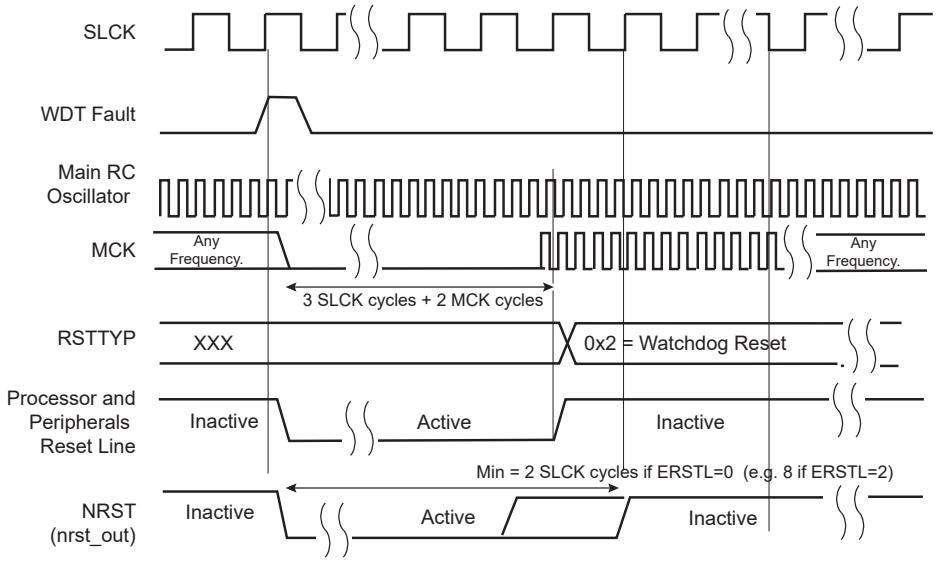
When in watchdog reset, the processor reset and the peripheral reset are asserted. The NRST line is also asserted, depending on the value of RSTC_MR.ERSTL. However, the resulting low level on NRST does not result in a user reset state.

The Watchdog Timer is reset by the proc_nreset signal. As the watchdog fault always causes a processor reset if WDT_MR.WDRSTEN is written to ‘1’, the Watchdog Timer is always reset after a watchdog reset, and the Watchdog is enabled by default and with a period set to a maximum.

When WDT_MR.WDRSTEN is written to ‘0’, the watchdog fault has no impact on the RSTC.

After a watchdog overflow occurs, the report on the RSTC_SR.RSTTYP may differ (either WDT_RST or USER_RST) depending on the external components driving the NRST pin. For example, if the NRST line is driven through a resistor and a capacitor (NRST pin debouncer), the reported value is USER_RST if the low to high transition is greater than one SLCK cycle.

**Figure 26-4. Watchdog Reset Timing Diagram**




### 26.4.3.4 Software Reset

The RSTC offers commands to assert the different reset signals. These commands are performed by writing the Control register (RSTC_CR) with the following bits at ‘1’:

- RSTC_CR.PROCRST: Writing a ‘1’ to PROCRST resets the processor and all the embedded peripherals, including the memory system and, in particular, the Remap Command.
- RSTC_CR.EXTRST: Writing a ‘1’ to EXTRST asserts low the NRST pin during a time defined by the field RSTC_MR.ERSTL.

The software reset is entered if at least one of these bits is written to ‘1’ by the software. All these commands can be performed independently or simultaneously. The software reset lasts three SLCK cycles.

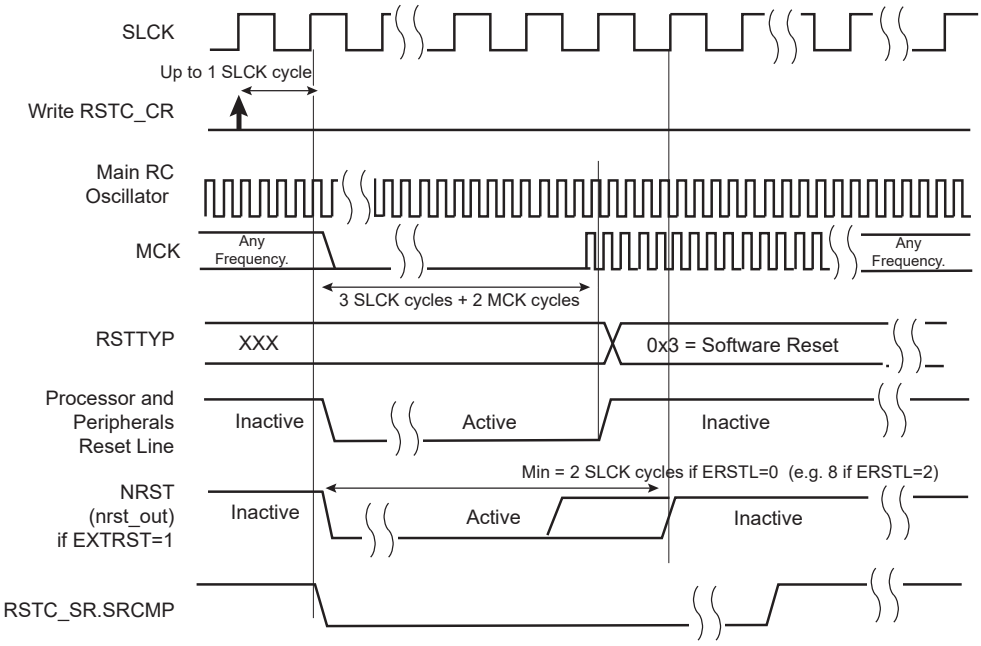
The internal reset signals are asserted as soon as the register write is performed. This is detected on the Master Clock (MCK). They are released when the software reset has ended, i.e., synchronously to SLCK.

If EXTRST is written to ‘1’, the nrst_out signal is asserted depending on the configuration of RSTC_MR.ERSTL. However, the resulting falling edge on NRST does not lead to a user reset.

If and only if the RSTC_CR.PROCRST is written to ‘1’, the RSTC reports the software status in field RSTC_SR.RSTTYP. Other software resets are not reported in RSTTYP.

As soon as a software operation is detected, RSTC_SR.SRCMP is written to ‘1’. SRCMP is cleared at the end of the software reset. No other software reset can be performed while SRCMP is written to ‘1’, and writing any value in the RSTC_CR has no effect.

**Figure 26-5. Software Reset Timing Diagram**




### 26.4.3.5 User Reset

A user reset is generated when a low level is detected on the NRST pin and RSTC_MR.URSTEN is at ‘1’. The NRST input signal is resynchronized with SLCK to ensure proper behavior of the system. Thus, the NRST pin must be asserted for at least 1 SLCK clock cycle to ensure execution of a user reset.

The user reset is triggered 2 SLCK cycles after a low level is detected on NRST. The processor reset and the peripheral reset are asserted.

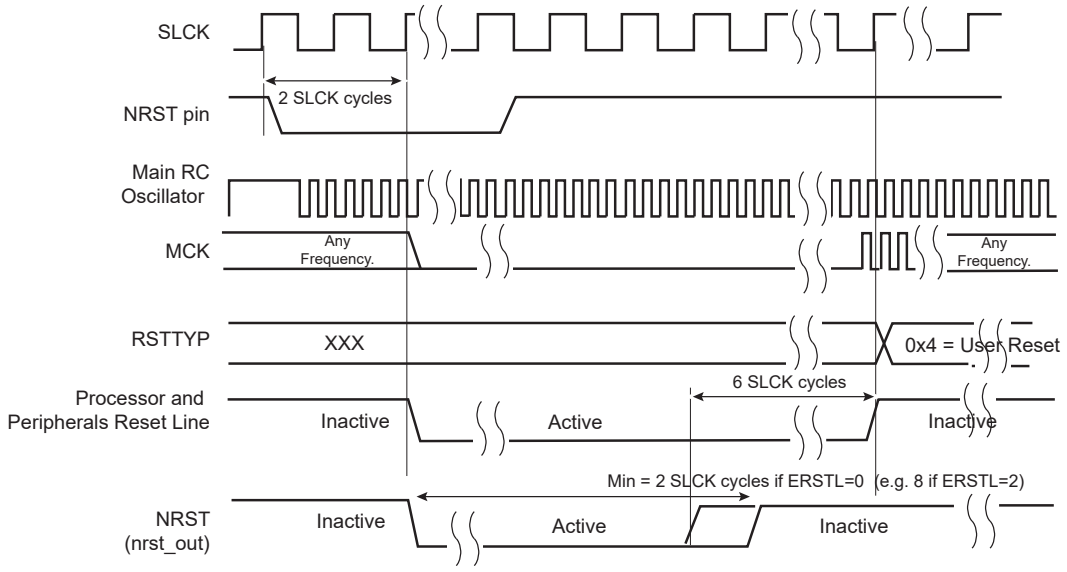
The user reset ends when NRST rises, after a two-cycle resynchronization time and a three-cycle processor startup. The processor clock is reenabled as soon as NRST is confirmed high.

When the processor reset signal is released, RSTC_SR.RSTTYP is loaded with the value ‘4’, indicating a user reset.

The NRST manager guarantees that the NRST line is asserted for External Reset Length SLCK cycles, as configured in RSTC_MR.ERSTL. However, if NRST does not rise after External Reset Length because it is driven low externally, the internal reset lines remain asserted until NRST actually rises.

© 2021 Microchip Technology Inc. and its subsidiaries

**Figure 26-6. User Reset Timing Diagram**




### 26.4.4 Reset State Priorities

The reset state manager manages the priorities among the different reset sources. The resets are listed in order of priority as follows:

1. General reset
2. Backup reset
3. Watchdog reset
4. Software reset
5. User reset

Specific cases are listed below:

- When in user reset:
  - A watchdog event is impossible because the Watchdog Timer is being reset by the proc_nreset signal.
  - A software reset is impossible, since the processor reset is being activated.
- When in software reset:
  - A watchdog event has priority over the current state.
  - The NRST has no effect.
- When in watchdog reset:
  - The processor reset is active and so a software reset cannot be programmed.
  - A user reset cannot be entered.

### 26.4.5 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | RSTC_CR | 7:0 | | | | | EXTRST | | | PROCRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |
| 0x04 | RSTC_SR | 7:0 | | | | | | | | URSTS |
| | | 15:8 | | | | | | RSTTYP[2:0] | | |
| | | 23:16 | | | | | | | SRCMP | NRSTL |
| | | 31:24 | | | | | | | | |
| 0x08 | RSTC_MR | 7:0 | | | | URSTIEN | | | | URSTEN |
| | | 15:8 | | | | | ERSTL[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | KEY[7:0] | | | | | | | |

### 26.4.5.1 RSTC Control Register

**Name:** RSTC_CR
**Offset:** 0x00
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EXTRST | | | PROCRST |
| Access | | | | | W | | | W |
| Reset | | | | | – | | | – |

**Bits 31:24 – KEY[7:0]** System Reset Key

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 3 – EXTRST** External Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If KEY = 0xA5, asserts the NRST pin. |

**Bit 0 – PROCRST** Processor Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If KEY = 0xA5, resets the processor and all the embedded peripherals. |

### 26.4.5.2 RSTC Status Register

**Name:** RSTC_SR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read-only

The register reset value assumes that a general reset has been performed; it is subject to change if other types of reset are generated.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SRCMP | NRSTL |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTTYP[2:0] | | |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URSTS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 17 – SRCMP** Software Reset Command in Progress
When set, this bit indicates that a software reset command is in progress and that no further software reset should be performed until the end of the current one. This bit is automatically cleared at the end of the current software reset.

| Value | Description |
|---|---|
| 0 | No software command is being performed by the RSTC. The RSTC is ready for a software command. |
| 1 | A software reset command is being performed by the RSTC. The RSTC is busy. |

**Bit 16 – NRSTL** NRST Pin Level
Registers the NRST pin level sampled on each MCK rising edge.

**Bits 10:8 – RSTTYP[2:0]** Reset Type
This field reports the cause of the last processor reset. Reading this RSTC_SR does not reset this field.

| Value | Name | Description |
|---|---|---|
| 0 | GENERAL_RST | First powerup reset |
| 1 | BACKUP_RST | Return from Backup mode |
| 2 | WDT_RST | Watchdog fault occurred |
| 3 | SOFT_RST | Processor reset required by the software |
| 4 | USER_RST | NRST pin detected low |
| 5 | – | Reserved |
| 6 | – | Reserved |
| 7 | – | Reserved |

**Bit 0 – URSTS** User Reset Status
A high-to-low transition of the NRST pin sets the URSTS. This transition is also detected on the MCK rising edge. If the user reset is disabled (URSTEN = 0 in RSTC_MR) and if the interrupt is enabled by RSTC_MR.URSTIEN, URSTS triggers an interrupt. Reading the RSTC_SR resets URSTS and clears the interrupt.

| Value | Description |
|---|---|
| 0 | No high-to-low edge on NRST happened since the last read of RSTC_SR. |
| 1 | At least one high-to-low transition of NRST has been detected since the last read of RSTC_SR. |

### 26.4.5.3 RSTC Mode Register

**Name:** RSTC_MR
**Offset:** 0x08
**Reset:** 0x00000001
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ERSTL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | URSTIEN | | | | URSTEN |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 1 |

**Bits 31:24 – KEY[7:0]** Write Access Password

| Value | Name | Description |
|---|---|---|
| 0xA5 | PASSWD | Writing any other value in this field aborts the write operation. Always reads as 0. |

**Bits 11:8 – ERSTL[3:0]** External Reset Length
This field defines the external reset length. The external reset is asserted during a time of $2^{(ERSTL+1)}$ SLCK cycles. This allows assertion duration to be programmed between 60 µs and 2 seconds. Note that synchronization cycles must also be considered when calculating the actual reset length as previously described.

**Bit 4 – URSTIEN** User Reset Interrupt Enable

| Value | Description |
|---|---|
| 0 | RSTC_SR.USRTS at ‘1’ has no effect on the RSTC interrupt line. |
| 1 | RSTC_SR.USRTS at ‘1’ asserts the RSTC interrupt line if URSTEN = 0. |

**Bit 0 – URSTEN** User Reset Enable

| Value | Description |
|---|---|
| 0 | The detection of a low level on the NRST pin does not generate a user reset. |
| 1 | The detection of a low level on the NRST pin triggers a user reset. |

# 27. Real-time Clock (RTC)

## 27.1 Description

The Real-time Clock (RTC) peripheral is designed for very low power consumption. For optimal functionality, the RTC requires an accurate external 32.768 kHz clock, which can be provided by a crystal oscillator.

It combines a complete time-of-day clock with alarm and a Gregorian or Persian calendar, complemented by a programmable periodic interrupt. The alarm and calendar registers are accessed by a 32-bit data bus.

The time and calendar values are coded in binary-coded decimal (BCD) format. The time format can be 24-hour mode or 12-hour mode with an AM/PM indicator.

Updating time and calendar fields and configuring the alarm fields are performed by a parallel capture on the 32-bit data bus. An entry control is performed to avoid loading registers with incompatible BCD format data or with an incompatible date according to the current month/year/century.

A clock divider calibration circuitry can be used to compensate for crystal oscillator frequency variations.

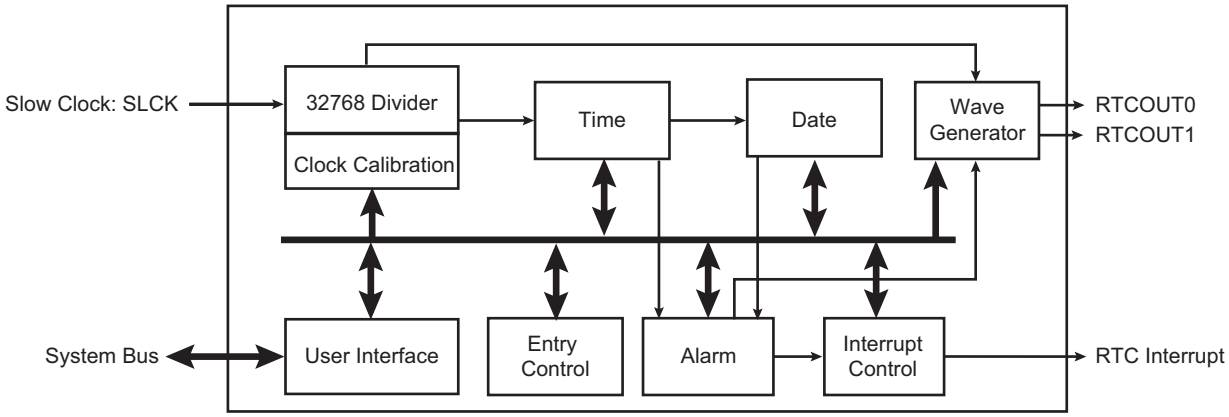
An RTC output can be programmed to generate several waveforms, including a prescaled clock derived from 32.768 kHz.

## 27.2 Embedded Characteristics

- Full Asynchronous Design for Ultra Low Power Consumption
- Gregorian and Persian Modes Supported
- Programmable Periodic Interrupt
- Safety/security Features:
  - Valid Time and Date Programming Check
  - On-The-Fly Time and Date Validity Check
- Counters Calibration Circuitry to Compensate for Crystal Oscillator Variations
- Waveform Generation
- Register Write Protection

## 27.3 Block Diagram

**Figure 27-1. Real-time Clock Block Diagram**



## 27.4 Product Dependencies

### 27.4.1 Power Management

The Real-time Clock is continuously clocked at 32.768 kHz. The Power Management Controller has no effect on RTC behavior.

### 27.4.2 Interrupt

Within the System Controller, the RTC interrupt is OR-wired with all the other module interrupts.

Only one System Controller interrupt line is connected on one of the internal sources of the interrupt controller.

RTC interrupt requires the interrupt controller to be programmed first.

When a System Controller interrupt occurs, the service routine must first determine the cause of the interrupt. This is done by reading each status register of the System Controller peripherals successively.

## 27.5 Functional Description

The RTC provides a full binary-coded decimal (BCD) clock that includes century (19/20), year (with leap years), month, date, day, hours, minutes and seconds reported in RTC Time Register (RTC_TIMR).

The valid year range is up to 2099 in Gregorian mode (or 1300 to 1499 in Persian mode).

The RTC can operate in 24-hour mode or in 12-hour mode with an AM/PM indicator.

Corrections for leap years are included (all years divisible by 4 being leap years except 1900). This is correct up to the year 2099.

The RTC can generate configurable waveforms on RTCOUT0/1 outputs.

### 27.5.1 Reference Clock

The reference clock is the Slow Clock (SLCK) which can be driven internally or by an external 32.768 kHz crystal.

During low-power modes of the processor, the oscillator runs and power consumption is critical. The crystal selection must consider the current consumption for power saving and the frequency drift due to temperature effect on the circuit for time accuracy.

### 27.5.2 Timing

The RTC is updated in real time at one-second intervals in Normal mode for the counters of seconds, at one-minute intervals for the counter of minutes and so on.

Due to the asynchronous operation of the RTC with respect to the rest of the chip, to be certain that the value read in the RTC registers (century, year, month, date, day, hours, minutes, seconds) are valid and stable, it is necessary to read these registers twice. If the data is the same both times, then it is valid. Therefore, a minimum of two and a maximum of three accesses are required.

### 27.5.3 Alarm

The RTC has five programmable fields: month, date, hours, minutes and seconds.

Each of these fields can be enabled or disabled to match the alarm condition:

- If all the fields are enabled, an alarm flag is generated (the corresponding flag is asserted and an interrupt generated if enabled) at a given month, date, hour/minute/second.
- If only the “seconds” field is enabled, then an alarm is generated every minute.

Depending on the combination of fields enabled, a large number of possibilities are available to the user ranging from minutes to 365/366 days.

Hour, minute and second matching alarms (SECEN, MINEN, HOUREN) can be enabled independently of SEC, MIN, HOUR fields.

© 2021 Microchip Technology Inc. and its subsidiaries

**Note:** To change one of the SEC, MIN, HOUR, DATE, MONTH fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_TIMALR or RTC_CALALR. The first access clears the enable corresponding to the field to change (SECEN, MINEN, HOUREN, DATEEN, MTHEN). If the field is already cleared, this access is not required. The second access performs the change of the value (SEC, MIN, HOUR, DATE, MONTH). The third access is required to re-enable the field by writing 1 in SECEN, MINEN, HOUREn, DATEEN, MTHEN fields.

### 27.5.4 Error Checking when Programming

Verification on user interface data is performed when accessing the century, year, month, date, day, hours, minutes, seconds and alarms. A check is performed on illegal BCD entries such as illegal date of the month with regard to the year and century configured.

If one of the time fields is not correct, the data is not loaded into the register/counter and a flag is set in the validity register. The user can not reset this flag. It is reset as soon as an acceptable value is programmed. This avoids any further side effects in the hardware. The same procedure is followed for the alarm.

The following checks are performed:

1. Century (check if it is in range 19–20 or 13–14 in Persian mode)
2. Year (BCD entry check)
3. Date (check range 01–31)
4. Month (check if it is in BCD range 01–12, check validity regarding “date”)
5. Day (check range 1–7)
6. Hour (BCD checks: in 24-hour mode, check range 00–23 and check that AM/PM flag is not set if RTC is set in 24-hour mode; in 12-hour mode check range 01–12)
7. Minute (check BCD and range 00–59)
8. Second (check BCD and range 00–59)

**Note:** If the 12-hour mode is selected by means of the RTC Mode Register (RTC_MR), a 12-hour value can be programmed and the returned value on RTC_TIMR will be the corresponding 24-hour value. The entry control checks the value of the AM/PM indicator (bit 22 of RTC_TIMR) to determine the range to be checked.

### 27.5.5 RTC Internal Free Running Counter Error Checking

To improve the reliability and security of the RTC, a permanent check is performed on the internal free running counters to report non-BCD or invalid date/time values.

An error is reported by TDERR bit in the status register (RTC_SR) if an incorrect value has been detected. The flag can be cleared by setting the TDERRCLR bit in the Status Clear Command Register (RTC_SCCR).

The TDERR error flag will be set again if the source of the error has not been cleared before clearing the TDERR flag. The clearing of the source of such error can be done by reprogramming a correct value on RTC_CALR and/or RTC_TIMR.

The RTC internal free running counters may automatically clear the source of TDERR due to their roll-over (i.e., every 10 seconds for SECONDS[3:0] field in RTC_TIMR). In this case the TDERR is held high until a clear command is asserted by TDERRCLR bit in RTC_SCCR.

### 27.5.6 Updating Time/Calendar

#### 27.5.6.1 Description

The update of the time/calendar must be synchronized on a second periodic event by either polling the RTC_SR.SEC status bit or by enabling the SECEN interrupt in the RTC_IER register.

Once the second event occurs, the user must stop the RTC by setting the corresponding field in the Control Register (RTC_CR). Bit UPDTIM must be set to update time fields (hour, minute, second) and bit UPDCAL must be set to update calendar fields (century, year, month, date, day).

The ACKUPD bit must then be read to 1 by either polling the RTC_SR or by enabling the ACKUPD interrupt in the RTC_IER. Once ACKUPD is read to 1, it is mandatory to clear this flag by writing the corresponding bit in the RTC_SCCR, after which the user can write to the Time Register, the Calendar Register, or both.

Once the update is finished, the user must write UPDTIM and/or UPDCAL to 0 in the RTC_CR.

© 2021 Microchip Technology Inc. and its subsidiaries

The timing sequence of the time/calendar update is described in the figure below.

When entering the programming mode of the calendar fields, the time fields remain enabled and both the time and the calendar fields are stopped. This is due to the location of the calendar logical circuity (downstream for low-power considerations). It is highly recommended to prepare all the fields to be updated before entering programming mode. In successive update operations, the user must wait for at least one second after resetting the UPDTIM/UPDCAL bit in the RTC_CR before setting these bits again. This is done by waiting for the SEC flag in the RTC_SR before setting the UPDTIM/UPDCAL bit. After resetting UPDTIM/UPDCAL, the SEC flag must also be cleared.

**Figure 27-2. Time/Calendar Update Timing Diagram**

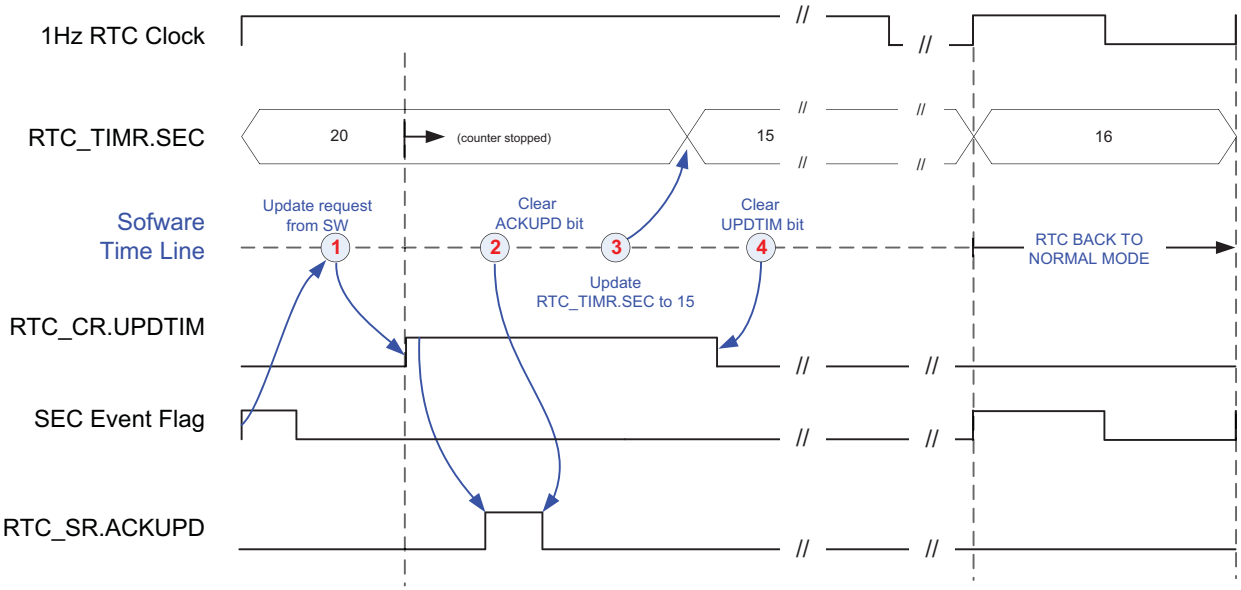
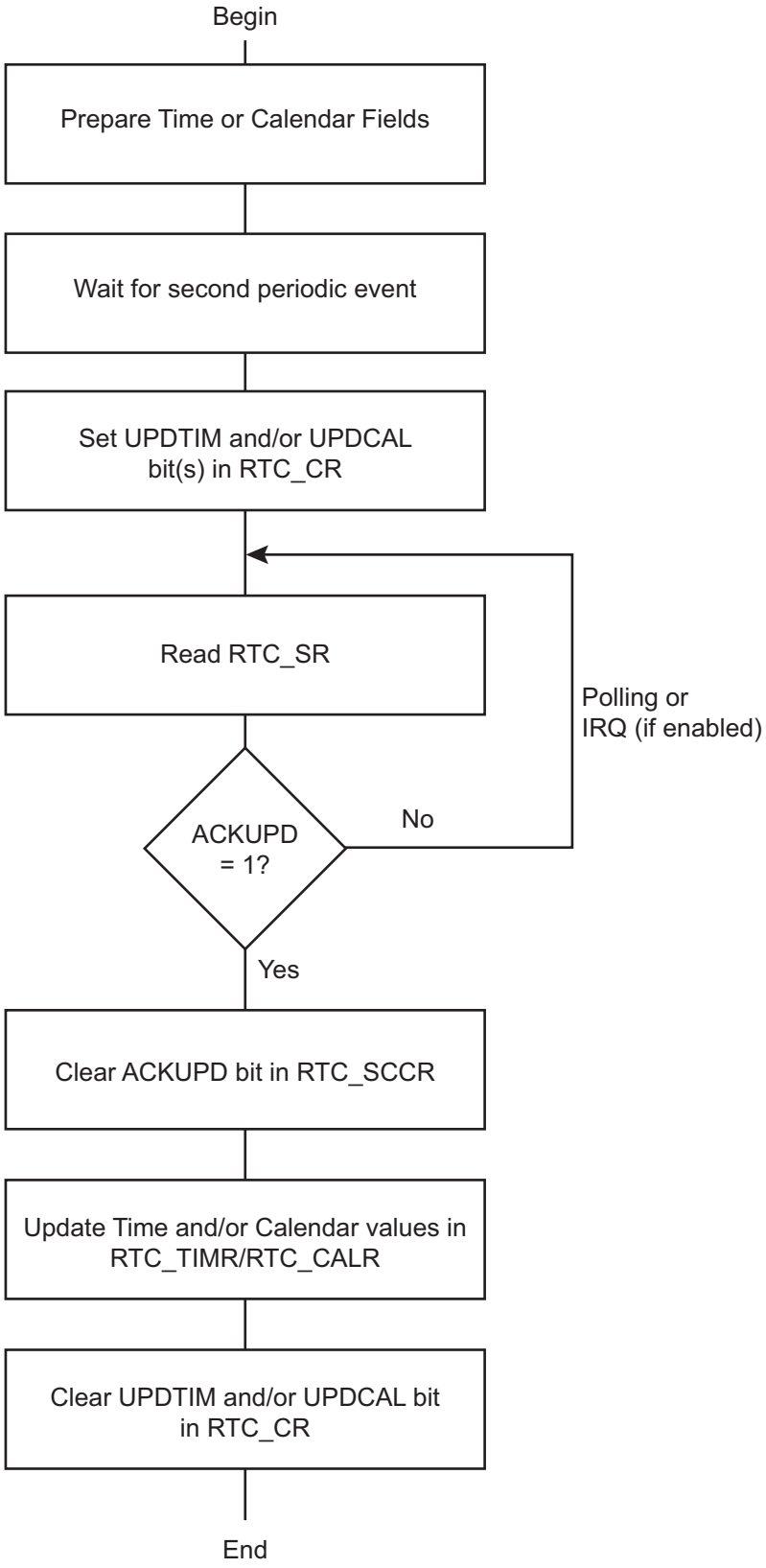


**Figure 27-3. Gregorian and Persian Modes Update Sequence**



### 27.5.7 RTC Accurate Clock Calibration

The crystal oscillator that drives the RTC may not be as accurate as expected mainly due to temperature variation. The RTC is equipped with circuitry able to correct slow clock crystal drift.

To compensate for possible temperature variations over time, this accurate clock calibration circuitry can be programmed on-the-fly and also programmed during application manufacturing, in order to correct the crystal frequency accuracy at room temperature (20–25°C). The typical clock drift range at room temperature is ±20 ppm.

In the device operating temperature range, the 32.768 kHz crystal oscillator clock inaccuracy can be up to -200 ppm.

The RTC clock calibration circuitry allows positive or negative correction in a range of 1.5 ppm to 1950 ppm.

The calibration circuitry is fully digital. Thus, the configured correction is independent of temperature, voltage, process, etc., and no additional measurement is required to check that the correction is effective.

If the correction value configured in the calibration circuitry results from an accurate crystal frequency measure, the remaining accuracy is bounded by the values listed below:

- Below 1 ppm, for an initial crystal drift between 1.5 ppm up to 20 ppm, and from 30 ppm to 90 ppm
- Below 2 ppm, for an initial crystal drift between 20 ppm up to 30 ppm, and from 90 ppm to 130 ppm
- Below 5 ppm, for an initial crystal drift between 130 ppm up to 200 ppm

The calibration circuitry does not modify the 32.768 kHz crystal oscillator clock frequency but it acts by slightly modifying the 1 Hz clock period from time to time. The correction event occurs every 1 + [(20 - (19 x HIGHPPM)) x CORRECTION] seconds. When the period is modified, depending on the sign of the correction, the 1 Hz clock period increases or reduces by around 4 ms. Depending on the CORRECTION, NEGPPM and HIGHPPM values configured in RTC_MR, the period interval between two correction events differs.

**Figure 27-4. Calibration Circuitry**

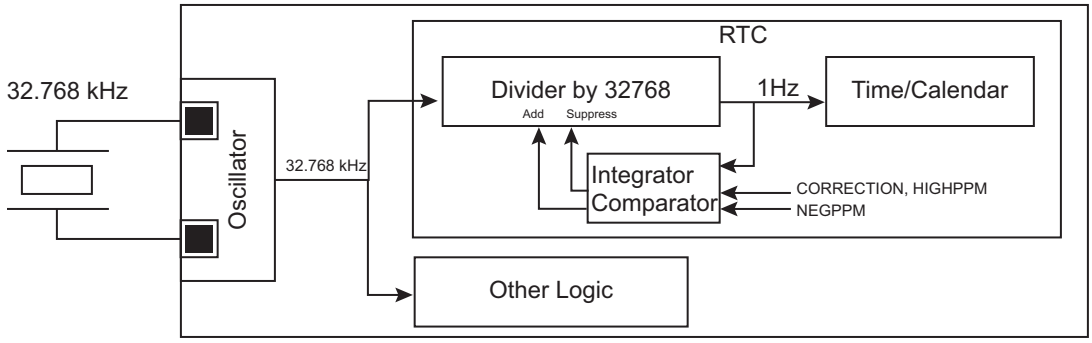
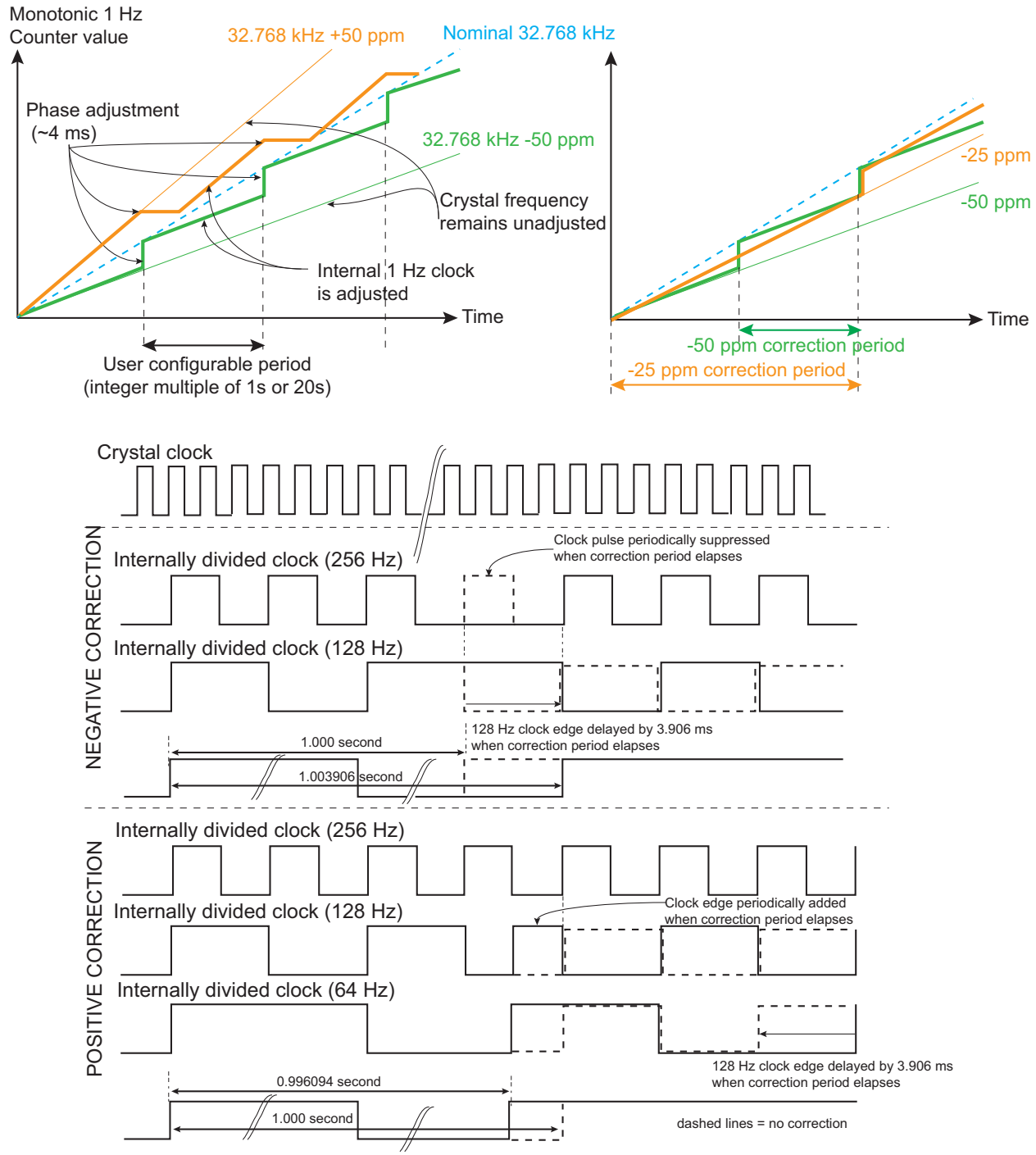


**Figure 27-5. Calibration Circuitry Waveforms**




The inaccuracy of a crystal oscillator at typical room temperature (±20 ppm at 20–25 °C) can be compensated if a reference clock/signal is used to measure such inaccuracy. This kind of calibration operation can be set up during the final product manufacturing by means of measurement equipment embedding such a reference clock. The correction of value must be programmed into the (RTC_MR), and this value is kept as long as the circuitry is powered (backup area). Removing the backup power supply cancels this calibration. This room temperature calibration can be further processed by means of the networking capability of the target application.

To ease the comparison of the inherent crystal accuracy with the reference clock/signal during manufacturing, an internal prescaled 32.768 kHz clock derivative signal can be assigned to drive RTC output. To accommodate the measure, several clock frequencies can be selected among 1 Hz, 32 Hz, 64 Hz, 512 Hz.

The clock calibration correction drives the internal RTC counters but can also be observed in the RTC output when one of the following three frequencies 1 Hz, 32 Hz or 64 Hz is configured. The correction is not visible in the RTC output if 512 Hz frequency is configured.

**Note:** This adjustment does not consider the temperature variation.

The frequency drift (up to -200 ppm) due to temperature variation can be compensated using a reference time if the application can access such a reference. If a reference time cannot be used, a temperature sensor can be placed close to the crystal oscillator in order to get the operating temperature of the crystal oscillator. Once obtained, the temperature may be converted using a lookup table (describing the accuracy/temperature curve of the crystal oscillator used) and RTC_MR configured accordingly. The calibration can be performed on-the-fly. This adjustment method is not based on a measurement of the crystal frequency/drift and therefore can be improved by means of the networking capability of the target application.

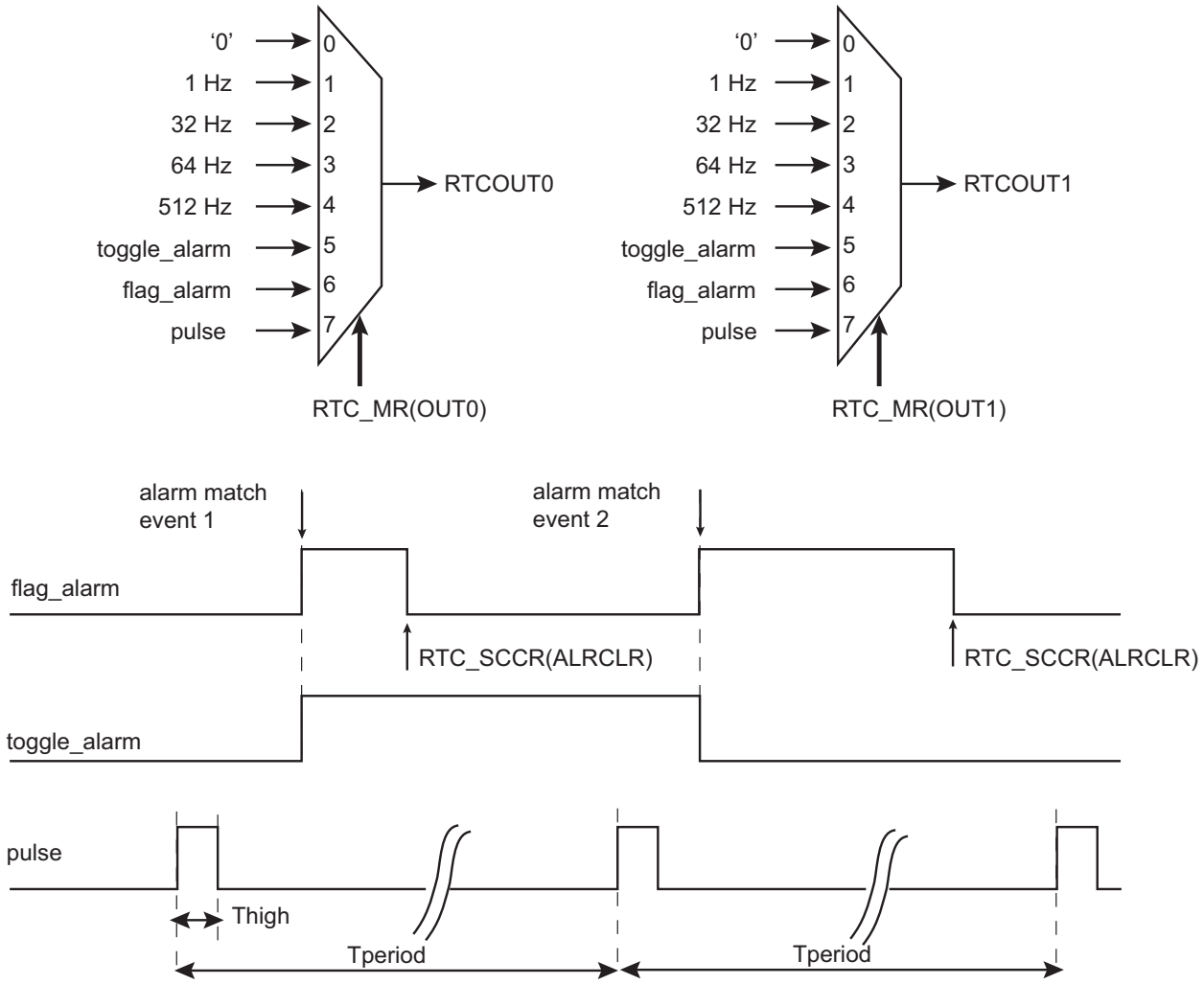
If no crystal frequency adjustment has been done during manufacturing, it is still possible to do it. In the case where a reference time of the day can be obtained through LAN/WAN network, it is possible to calculate the drift of the application crystal oscillator by comparing the values read on RTC Time Register (RTC_TIMR) and programming the HIGHPPM and CORRECTION fields on RTC_MR according to the difference measured between the reference time and those of RTC_TIMR.

### 27.5.8 Waveform Generation

Waveforms can be generated in order to take advantage of the RTC inherent prescalers while the RTC is the only powered circuitry (Low-power mode of operation, Backup mode) or in any active mode. Entering Backup or Low-power operating modes does not affect the waveform generation outputs.

The outputs RTCOUT0 and RTCOUT1 can be configured to provide several types of waveforms. The figure below illustrates the different signals available to generate RTCOUT0 and RTCOUT1.

**Figure 27-6. Waveform Generation**



## 27.6 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | RTC_CR | 7:0 | | | | | | | UPDCAL | UPDTIM |
| | | 15:8 | | | | | | | TIMEVSEL[1:0] | |
| | | 23:16 | | | | | | | CALEVSEL[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x04 | RTC_MR | 7:0 | | | | NEGPPM | | | PERSIAN | HRMOD |
| | | 15:8 | HIGHPPM | CORRECTION[6:0] | | | | | | |
| | | 23:16 | | OUT1[2:0] | | | | OUT0[2:0] | | |
| | | 31:24 | | | TPERIOD[1:0] | | | THIGH[2:0] | | |
| 0x08 | RTC_TIMR | 7:0 | | SEC[6:0] | | | | | | |
| | | 15:8 | | MIN[6:0] | | | | | | |
| | | 23:16 | | AMPM | HOUR[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | RTC_CALR | 7:0 | | CENT[6:0] | | | | | | |
| | | 15:8 | YEAR[7:0] | | | | | | | |
| | | 23:16 | DAY[2:0] | | | MONTH[4:0] | | | | |
| | | 31:24 | | | DATE[5:0] | | | | | |
| 0x10 | RTC_TIMALR | 7:0 | SECEN | SEC[6:0] | | | | | | |
| | | 15:8 | MINEN | MIN[6:0] | | | | | | |
| | | 23:16 | HOUREN | AMPM | HOUR[5:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | RTC_CALALR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | MTHEN | | | MONTH[4:0] | | | | |
| | | 31:24 | DATEEN | | DATE[5:0] | | | | | |
| 0x18 | RTC_SR | 7:0 | | | TDERR | CALEV | TIMEV | SEC | ALARM | ACKUPD |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | RTC_SCCR | 7:0 | | | TDERRCLR | CALCLR | TIMCLR | SECCLR | ALRCLR | ACKCLR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | RTC_IER | 7:0 | | | TDERREN | CALEN | TIMEN | SECEN | ALREN | ACKEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | RTC_IDR | 7:0 | | | TDERRDIS | CALDIS | TIMDIS | SECDIS | ALRDIS | ACKDIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | RTC_IMR | 7:0 | | | TDERR | CAL | TIM | SEC | ALR | ACK |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | RTC_VER | 7:0 | | | | | NVCALALR | NVTIMALR | NVCAL | NVTIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 27.6.1 RTC Control Register

**Name:** RTC_CR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CALEVSEL[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | TIMEVSEL[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | UPDCAL | UPDTIM |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 17:16 – CALEVSEL[1:0]** Calendar Event Selection
The event that generates the flag CALEV in RTC_SR depends on the value of CALEVSEL

| Value | Name | Description |
|---|---|---|
| 0 | WEEK | Week change (every Monday at time 00:00:00) |
| 1 | MONTH | Month change (every 01 of each month at time 00:00:00) |
| 2 | YEAR | Year change (every January 1 at time 00:00:00) |
| 3 | YEAR | Reserved |

**Bits 9:8 – TIMEVSEL[1:0]** Time Event Selection
The event that generates the flag TIMEV in RTC_SR depends on the value of TIMEVSEL.

| Value | Name | Description |
|---|---|---|
| 0 | MINUTE | Minute change |
| 1 | HOUR | Hour change |
| 2 | MIDNIGHT | Every day at midnight |
| 3 | NOON | Every day at noon |

**Bit 1 – UPDCAL** Update Request Calendar Register
Calendar counting consists of day, date, month, year and century counters. Calendar counters can be programmed once this bit is set and acknowledged by the bit ACKUPD of the RTC_SR.

| Value | Description |
|---|---|
| 0 | No effect or, if UPDCAL has been previously written to 1, stops the update procedure. |
| 1 | Stops the RTC calendar counting. |

**Bit 0 – UPDTIM** Update Request Time Register
Time counting consists of second, minute and hour counters. Time counters can be programmed once this bit is set and acknowledged by the bit ACKUPD of the RTC_SR.

| Value | Description |
|---|---|
| 0 | No effect or, if UPDTIM has been previously written to 1, stops the update procedure. |
| 1 | Stops the RTC time counting. |

### 27.6.2 RTC Mode Register

**Name:** RTC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TPERIOD[1:0] | | | THIGH[2:0] | | |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | OUT1[2:0] | | | | OUT0[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HIGHPPM | CORRECTION[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NEGPPM | | | PERSIAN | HRMOD |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

**Bits 29:28 – TPERIOD[1:0]** Period of the Output Pulse

| Value | Name | Description |
|---|---|---|
| 0 | P_1S | 1 second |
| 1 | P_500MS | 500 ms |
| 2 | P_250MS | 250 ms |
| 3 | P_125MS | 125 ms |

**Bits 26:24 – THIGH[2:0]** High Duration of the Output Pulse

| Value | Name | Description |
|---|---|---|
| 0 | H_31MS | 31.2 ms |
| 1 | H_16MS | 15.6 ms |
| 2 | H_4MS | 3.91 ms |
| 3 | H_976US | 976 µs |
| 4 | H_488US | 488 µs |
| 5 | H_122US | 122 µs |
| 6 | H_30US | 30.5 µs |
| 7 | H_15US | 15.2 µs |

**Bits 22:20 – OUT1[2:0]** RTCOUT1 Output Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | NO_WAVE | No waveform, stuck at ‘0’ |
| 1 | FREQ1HZ | 1 Hz square wave |
| 2 | FREQ32HZ | 32 Hz square wave |
| 3 | FREQ64HZ | 64 Hz square wave |
| 4 | FREQ512HZ | 512 Hz square wave |
| 5 | ALARM_TOGGLE | Output toggles when alarm flag rises |

| Value | Name | Description |
|---|---|---|
| 6 | ALARM_FLAG | Output is a copy of the alarm flag |
| 7 | PROG_PULSE | Duty cycle programmable pulse |

**Bits 18:16 – OUT0[2:0]** RTCOUT0 Output Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | NO_WAVE | No waveform, stuck at '0' |
| 1 | FREQ1HZ | 1 Hz square wave |
| 2 | FREQ32HZ | 32 Hz square wave |
| 3 | FREQ64HZ | 64 Hz square wave |
| 4 | FREQ512HZ | 512 Hz square wave |
| 5 | ALARM_TOGGLE | Output toggles when alarm flag rises |
| 6 | ALARM_FLAG | Output is a copy of the alarm flag |
| 7 | PROG_PULSE | Duty cycle programmable pulse |

**Bit 15 – HIGHPPM** HIGH PPM Correction
If the absolute value of the correction to be applied is lower than 30 ppm, it is recommended to clear HIGHPPM. HIGHPPM set to 1 is recommended for 30 ppm correction and above.
**Formula:**
If HIGHPPM = 0, then the clock frequency correction range is from 1.5 ppm up to 98 ppm. The RTC accuracy is less than 1 ppm for a range correction from 1.5 ppm up to 30 ppm.
The correction field must be programmed according to the required correction in ppm; the formula is as follows:

$$\text{CORRECTION} = \frac{3906}{20 \times \text{ppm}} - 1$$

The value obtained must be rounded to the nearest integer prior to being programmed into CORRECTION field.
If HIGHPPM = 1, then the clock frequency correction range is from 30.5 ppm up to 1950 ppm. The RTC accuracy is less than 1 ppm for a range correction from 30.5 ppm up to 90 ppm.
The correction field must be programmed according to the required correction in ppm; the formula is as follows:

$$\text{CORRECTION} = \frac{3906}{\text{ppm}} - 1$$

The value obtained must be rounded to the nearest integer prior to be programmed into CORRECTION field.
If NEGPPM is set to 1, the ppm correction is negative (used to correct crystals that are faster than the nominal 32.768 kHz).

| Value | Description |
|---|---|
| 0 | Lower range ppm correction with accurate correction. |
| 1 | Higher range ppm correction with accurate correction. |

**Bits 14:8 – CORRECTION[6:0]** Slow Clock Correction

| Value | Description |
|---|---|
| 0 | No correction |
| 1-127 | The slow clock will be corrected according to the formula given in HIGHPPM description. |

**Bit 4 – NEGPPM** Negative PPM Correction
See CORRECTION and HIGHPPM field descriptions.
NEGPPM must be cleared to correct a crystal slower than 32.768 kHz.

| Value | Description |
|---|---|
| 0 | Positive correction (the divider will be slightly higher than 32768). |
| 1 | Negative correction (the divider will be slightly lower than 32768). |

**Bit 1 – PERSIAN** PERSIAN Calendar

| Value | Description |
|---|---|
| 0 | Gregorian calendar. |
| 1 | Persian calendar. |

**Bit 0 – HRMOD** 12-/24-hour Mode

| Value | Description |
|---|---|
| 0 | 24-hour mode is selected. |

| Value | Description |
|---|---|
| 1 | 12-hour mode is selected. |

### 27.6.3 RTC Time Register

**Name:** RTC_TIMR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | AMPM | HOUR[5:0] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | MIN[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SEC[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 22 – AMPM** Ante Meridiem Post Meridiem Indicator
This bit is the AM/PM indicator in 12-hour mode.

| Value | Description |
|---|---|
| 0 | AM. |
| 1 | PM. |

**Bits 21:16 – HOUR[5:0]** Current Hour
The range that can be set is 1–12 (BCD) in 12-hour mode or 0–23 (BCD) in 24-hour mode.

**Bits 14:8 – MIN[6:0]** Current Minute
The range that can be set is 0–59 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

**Bits 6:0 – SEC[6:0]** Current Second
The range that can be set is 0–59 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 27.6.4 RTC Calendar Register

**Name:** RTC_CALR
**Offset:** 0x0C
**Reset:** 0x01E11320
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | DATE[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DAY[2:0] | | | MONTH[4:0] | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | YEAR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CENT[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:24 – DATE[5:0]** Current Day in Current Month
The range that can be set is 01–31 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

**Bits 23:21 – DAY[2:0]** Current Day in Current Week
The range that can be set is 1–7 (BCD).
The coding of the number (which number represents which day) is user-defined as it has no effect on the date counter.

**Bits 20:16 – MONTH[4:0]** Current Month
The range that can be set is 01–12 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

**Bits 15:8 – YEAR[7:0]** Current Year
The range that can be set is 00–99 (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

**Bits 6:0 – CENT[6:0]** Current Century
The range that can be set is 19–20 (Gregorian) or 13–14 (Persian) (BCD).
The lowest four bits encode the units. The higher bits encode the tens.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 27.6.5 RTC Time Alarm Register

**Name:** RTC_TIMALR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

To change one of the SEC, MIN, HOUR fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_TIMALR. The first access clears the enable corresponding to the field to change (SECEN, MINEN, HOUREN). If the field is already cleared, this access is not required. The second access performs the change of the value (SEC, MIN, HOUR). The third access is required to re-enable the field by writing 1 in SECEN, MINEN, HOUREN fields.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HOUREN | AMPM | HOUR[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MINEN | MIN[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SECEN | SEC[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 23 – HOUREN** Hour Alarm Enable

| Value | Description |
|---|---|
| 0 | The hour-matching alarm is disabled. |
| 1 | The hour-matching alarm is enabled. |

**Bit 22 – AMPM** AM/PM Indicator
This field is the alarm field corresponding to the BCD-coded hour counter.

**Bits 21:16 – HOUR[5:0]** Hour Alarm
This field is the alarm field corresponding to the BCD-coded hour counter.

**Bit 15 – MINEN** Minute Alarm Enable

| Value | Description |
|---|---|
| 0 | The minute-matching alarm is disabled. |
| 1 | The minute-matching alarm is enabled. |

**Bits 14:8 – MIN[6:0]** Minute Alarm
This field is the alarm field corresponding to the BCD-coded minute counter.

**Bit 7 – SECEN** Second Alarm Enable

| Value | Description |
|---|---|
| 0 | The second-matching alarm is disabled. |
| 1 | The second-matching alarm is enabled. |

**Bits 6:0 – SEC[6:0]** Second Alarm
This field is the alarm field corresponding to the BCD-coded second counter.

### 27.6.6 RTC Calendar Alarm Register

**Name:** RTC_CALALR
**Offset:** 0x14
**Reset:** 0x01010000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

To change one of the DATE, MONTH fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_CALALR. The first access clears the enable corresponding to the field to change (DATEEN, MTHEN). If the field is already cleared, this access is not required. The second access performs the change of the value (DATE, MONTH). The third access is required to re-enable the field by writing 1 in DATEEN, MTHEN fields.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATEEN | | DATE[5:0] | | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MTHEN | | | MONTH[4:0] | | | | |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – DATEEN** Date Alarm Enable

| Value | Description |
|---|---|
| 0 | The date-matching alarm is disabled. |
| 1 | The date-matching alarm is enabled. |

**Bits 29:24 – DATE[5:0]** Date Alarm
This field is the alarm field corresponding to the BCD-coded date counter.

**Bit 23 – MTHEN** Month Alarm Enable

| Value | Description |
|---|---|
| 0 | The month-matching alarm is disabled. |
| 1 | The month-matching alarm is enabled. |

**Bits 20:16 – MONTH[4:0]** Month Alarm
This field is the alarm field corresponding to the BCD-coded month counter.

### 27.6.7 RTC Status Register

**Name:** RTC_SR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERR | CALEV | TIMEV | SEC | ALARM | ACKUPD |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – TDERR** Time and/or Date Free Running Error

| Value | Name | Description |
|---|---|---|
| 0 | CORRECT | The internal free running counters are carrying valid values since the last read of the Status Register (RTC_SR). |
| 1 | ERR_TIMEDATE | The internal free running counters have been corrupted (invalid date or time, non-BCD values) since the last read and/or they are still invalid. |

**Bit 4 – CALEV** Calendar Event

The calendar event is selected in the CALEVSEL field in the Control Register (RTC_CR) and can be any one of the following events: week change, month change and year change.

| Value | Name | Description |
|---|---|---|
| 0 | NO_CALEVENT | No calendar event has occurred since the last clear. |
| 1 | CALEVENT | At least one calendar event has occurred since the last clear. |

**Bit 3 – TIMEV** Time Event

The time event is selected in the TIMEVSEL field in the Control Register (RTC_CR) and can be any one of the following events: minute change, hour change, noon, midnight (day change).

| Value | Name | Description |
|---|---|---|
| 0 | NO_TIMEVENT | No time event has occurred since the last clear. |
| 1 | TIMEVENT | At least one time event has occurred since the last clear. |

**Bit 2 – SEC** Second Event

| Value | Name | Description |
|---|---|---|
| 0 | NO_SECEVENT | No second event has occurred since the last clear. |
| 1 | SECEVENT | At least one second event has occurred since the last clear. |

**Bit 1 – ALARM** Alarm Flag

| Value | Name | Description |
|---|---|---|
| 0 | NO_ALARMEVENT | No alarm matching condition occurred. |

| Value | Name | Description |
|---|---|---|
| 1 | ALARMEVENT | An alarm matching condition has occurred. |

**Bit 0 – ACKUPD** Acknowledge for Update

| Value | Name | Description |
|---|---|---|
| 0 | FREERUN | Time and calendar registers cannot be updated. |
| 1 | UPDATE | Time and calendar registers can be updated. |

### 27.6.8 RTC Status Clear Command Register

**Name:** RTC_SCCR
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERRCLR | CALCLR | TIMCLR | SECCLR | ALRCLR | ACKCLR |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

**Bit 5 – TDERRCLR** Time and/or Date Free Running Error Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears corresponding status flag in the Status Register (RTC_SR). |

**Bit 4 – CALCLR** Calendar Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears corresponding status flag in the Status Register (RTC_SR). |

**Bit 3 – TIMCLR** Time Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears corresponding status flag in the Status Register (RTC_SR). |

**Bit 2 – SECCLR** Second Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears corresponding status flag in the Status Register (RTC_SR). |

**Bit 1 – ALRCLR** Alarm Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears corresponding status flag in the Status Register (RTC_SR). |

**Bit 0 – ACKCLR** Acknowledge Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears corresponding status flag in the Status Register (RTC_SR). |

### 27.6.9 RTC Interrupt Enable Register

**Name:** RTC_IER
**Offset:** 0x20
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERREN | CALEN | TIMEN | SECEN | ALREN | ACKEN |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

**Bit 5 – TDERREN** Time and/or Date Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The time and date error interrupt is enabled. |

**Bit 4 – CALEN** Calendar Event Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The selected calendar event interrupt is enabled. |

**Bit 3 – TIMEN** Time Event Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The selected time event interrupt is enabled. |

**Bit 2 – SECEN** Second Event Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The second periodic interrupt is enabled. |

**Bit 1 – ALREN** Alarm Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The alarm interrupt is enabled. |

**Bit 0 – ACKEN** Acknowledge Update Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The acknowledge for update interrupt is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 27.6.10 RTC Interrupt Disable Register

**Name:** RTC_IDR
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERRDIS | CALDIS | TIMDIS | SECDIS | ALRDIS | ACKDIS |
| Access | | | W | W | W | W | W | W |
| Reset | | | – | – | – | – | – | – |

**Bit 5 – TDERRDIS** Time and/or Date Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The time and date error interrupt is disabled. |

**Bit 4 – CALDIS** Calendar Event Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The selected calendar event interrupt is disabled. |

**Bit 3 – TIMDIS** Time Event Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The selected time event interrupt is disabled. |

**Bit 2 – SECDIS** Second Event Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The second periodic interrupt is disabled. |

**Bit 1 – ALRDIS** Alarm Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The alarm interrupt is disabled. |

**Bit 0 – ACKDIS** Acknowledge Update Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The acknowledge for update interrupt is disabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 27.6.11 RTC Interrupt Mask Register

**Name:** RTC_IMR
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TDERR | CAL | TIM | SEC | ALR | ACK |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – TDERR** Time and/or Date Error Mask

| Value | Description |
|---|---|
| 0 | The time and/or date error event is disabled. |
| 1 | The time and/or date error event is enabled. |

**Bit 4 – CAL** Calendar Event Interrupt Mask

| Value | Description |
|---|---|
| 0 | The selected calendar event interrupt is disabled. |
| 1 | The selected calendar event interrupt is enabled. |

**Bit 3 – TIM** Time Event Interrupt Mask

| Value | Description |
|---|---|
| 0 | The selected time event interrupt is disabled. |
| 1 | The selected time event interrupt is enabled. |

**Bit 2 – SEC** Second Event Interrupt Mask

| Value | Description |
|---|---|
| 0 | The second periodic interrupt is disabled. |
| 1 | The second periodic interrupt is enabled. |

**Bit 1 – ALR** Alarm Interrupt Mask

| Value | Description |
|---|---|
| 0 | The alarm interrupt is disabled. |
| 1 | The alarm interrupt is enabled. |

**Bit 0 – ACK** Acknowledge Update Interrupt Mask

| Value | Description |
|---|---|
| 0 | The acknowledge for update interrupt is disabled. |
| 1 | The acknowledge for update interrupt is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 27.6.12 RTC Valid Entry Register

**Name:** RTC_VER
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | NVCALALR | NVTIMALR | NVCAL | NVTIM |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – NVCALALR** Non-valid Calendar Alarm

| Value | Description |
|---|---|
| 0 | No invalid data has been detected in RTC_CALALR (Calendar Alarm Register). |
| 1 | RTC_CALALR has contained invalid data since it was last programmed. |

**Bit 2 – NVTIMALR** Non-valid Time Alarm

| Value | Description |
|---|---|
| 0 | No invalid data has been detected in RTC_TIMALR (Time Alarm Register). |
| 1 | RTC_TIMALR has contained invalid data since it was last programmed. |

**Bit 1 – NVCAL** Non-valid Calendar

| Value | Description |
|---|---|
| 0 | No invalid data has been detected in RTC_CALR (Calendar Register). |
| 1 | RTC_CALR has contained invalid data since it was last programmed. |

**Bit 0 – NVTIM** Non-valid Time

| Value | Description |
|---|---|
| 0 | No invalid data has been detected in RTC_TIMR (Time Register). |
| 1 | RTC_TIMR has contained invalid data since it was last programmed. |

© 2021 Microchip Technology Inc.
and its subsidiaries

# 28. Real-time Timer (RTT)

## 28.1 Description

The Real-Time Timer (RTT) is built around a 32-bit counter used to count roll-over events of the programmable 16-bit prescaler driven from the 32-kHz slow clock source. It generates a periodic interrupt and/or triggers an alarm on a programmed value.

The RTT can also be configured to be driven by the 1Hz RTC signal, thus taking advantage of a calibrated 1Hz clock.

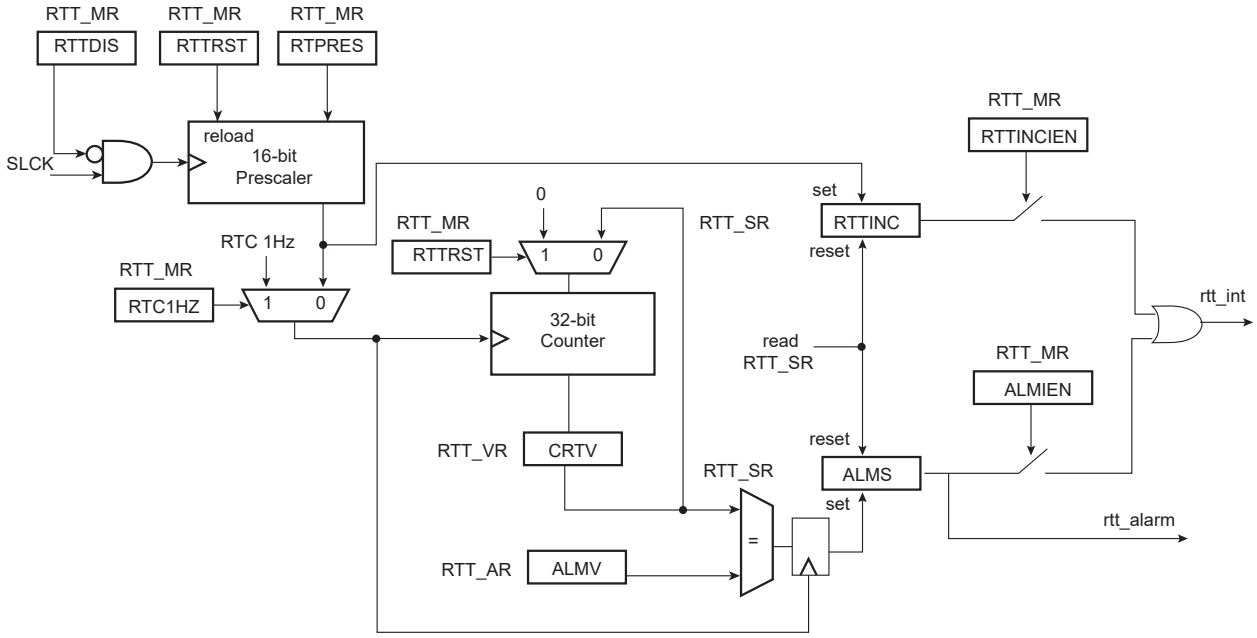
The slow clock source can be fully disabled to reduce power consumption when only an elapsed seconds count is required.

## 28.2 Embedded Characteristics

- 32-bit Free-running Counter on prescaled slow clock or RTC calibrated 1Hz clock
- 16-bit Configurable Prescaler
- Interrupt on Alarm or Counter Increment

## 28.3 Block Diagram

**Figure 28-1. Real-time Timer Block Diagram**




## 28.4 Functional Description

The programmable 16-bit prescaler value can be configured through the RTPRES field in the RTT Mode register (RTT_MR).

Configuring the RTPRES field value to 0x8000 (default value) corresponds to feeding the real-time counter with a 1Hz signal (if the slow clock is 32.768 kHz). The 32-bit counter can count up to $2^{32}$ seconds, corresponding to more than 136 years, then roll over to 0. Bit RTTINC in the RTT Status Register (RTT_SR) is set each time there is a prescaler roll-over.

The real-time 32-bit counter can also be supplied by the 1Hz RTC clock. This mode is interesting when the RTC 1Hz is calibrated (CORRECTION field ≠ 0 in RTC_MR) in order to guaranty the synchronism between RTC and RTT counters.

Setting the RTC1HZ bit in the RTT_MR drives the 32-bit RTT counter from the 1Hz RTC clock. In this mode, the RTPRES field has no effect on the 32-bit counter.

The prescaler roll-over generates an increment of the real-time timer counter if RTC1HZ = 0. Otherwise, if RTC1HZ = 1, the RTT counter is incremented every second. The RTTINC bit is set independently from the 32-bit counter increment.

The RTT can also be used as a free-running timer with a lower time-base. The best accuracy is achieved by writing RTPRES to 3 in RTT_MR.

Programming RTPRES to 1 or 2 is forbidden.

If the RTT is configured to trigger an interrupt, the interrupt occurs two slow clock cycles after reading the RTT_SR. To prevent several executions of the interrupt handler, the interrupt must be disabled in the interrupt handler and re-enabled when the RTT_SR is cleared.

The CRTV field can be read at any time in the RTT Value register (RTT_VR). As this value can be updated asynchronously with the Master Clock, the CRTV field must be read twice at the same value to read a correct value.

The current value of the counter is compared with the value written in the RTT Alarm register (RTT_AR). If the counter value matches the alarm, the ALMS bit in the RTT_SR is set. The RTT_AR is set to its maximum value (0xFFFFFFFF) after a reset.

The ALMS flag is always a source of the RTT alarm signal that may be used to exit the system from low power modes (see the Real-time Timer Block Diagram above).

The alarm interrupt must be disabled (ALMIEN must be cleared in RTT_MR) when writing a new ALMV value in the RTT_AR.

The RTTINC bit can be used to start a periodic interrupt, the period being one second when the RTPRES field value = 0x8000 and the slow clock = 32.768 kHz.

The RTTINCIEN bit must be cleared prior to writing a new RTPRES value in the RTT_MR.

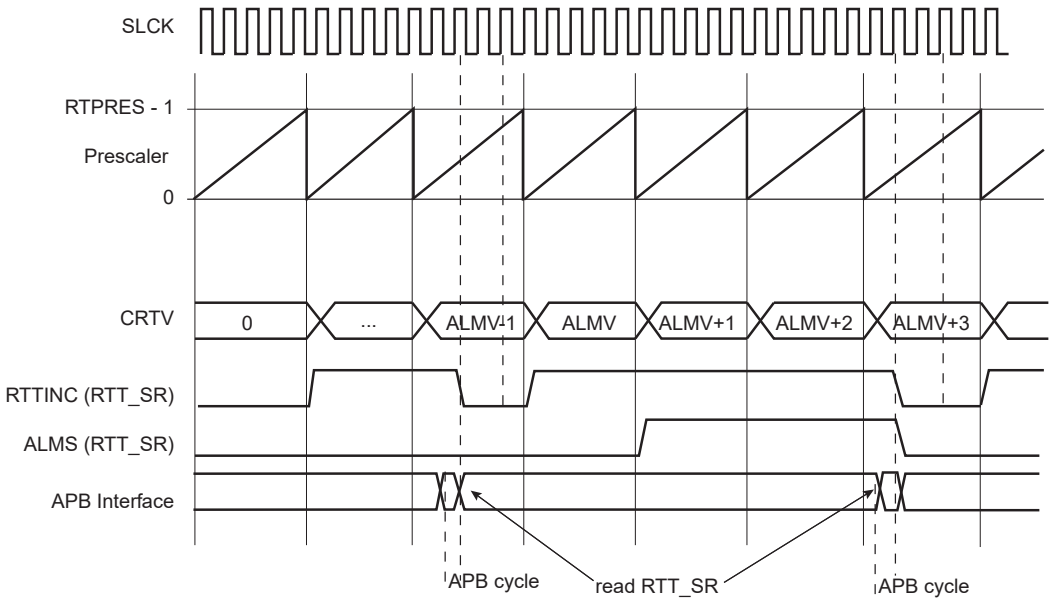
Reading the RTT_SR automatically clears the RTTINC and ALMS bits.

Writing the RTTRST bit in the RTT_MR immediately reloads and restarts the clock divider with the new programmed value. This also resets the 32-bit counter.

When not used, the RTT can be disabled in order to suppress dynamic power consumption in this module. This can be achieved by setting the RTTDIS bit in the RTT_MR.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 28-2. RTT Counting**



## 28.5 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | RTT_MR | 7:0 | RTPRES[7:0] | | | | | | | |
| | | 15:8 | RTPRES[15:8] | | | | | | | |
| | | 23:16 | | | | RTTDIS | | RTTRST | RTTINCIEN | ALMIEN |
| | | 31:24 | | | | | | | | RTC1HZ |
| 0x04 | RTT_AR | 7:0 | ALMV[7:0] | | | | | | | |
| | | 15:8 | ALMV[15:8] | | | | | | | |
| | | 23:16 | ALMV[23:16] | | | | | | | |
| | | 31:24 | ALMV[31:24] | | | | | | | |
| 0x08 | RTT_VR | 7:0 | CRTV[7:0] | | | | | | | |
| | | 15:8 | CRTV[15:8] | | | | | | | |
| | | 23:16 | CRTV[23:16] | | | | | | | |
| | | 31:24 | CRTV[31:24] | | | | | | | |
| 0x0C | RTT_SR | 7:0 | | | | | | | RTTINC | ALMS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 28.5.1 Real-time Timer Mode Register

**Name:** RTT_MR
**Offset:** 0x00
**Reset:** 0x00008000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RTC1HZ |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | RTTDIS | | RTTRST | RTTINCIEN | ALMIEN |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RTPRES[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RTPRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – RTC1HZ** Real-Time Clock 1Hz Clock Selection

| Value | Description |
|---|---|
| 0 | The RTT 32-bit counter is driven by the 16-bit prescaler roll-over events. |
| 1 | The RTT 32-bit counter is driven by the 1Hz RTC clock. |

**Bit 20 – RTTDIS** Real-time Timer Disable

| Value | Description |
|---|---|
| 0 | The RTT is enabled. |
| 1 | The RTT is disabled (no dynamic power consumption). |

**Bit 18 – RTTRST** Real-time Timer Restart

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Reloads and restarts the clock divider with the new programmed value. This also resets the 32-bit counter. |

**Bit 17 – RTTINCIEN** Real-time Timer Increment Interrupt Enable

| Value | Description |
|---|---|
| 0 | The bit RTTINC in RTT_SR has no effect on interrupt. |
| 1 | The bit RTTINC in RTT_SR asserts interrupt. |

**Bit 16 – ALMIEN** Alarm Interrupt Enable

| Value | Description |
|---|---|
| 0 | The bit ALMS in RTT_SR has no effect on interrupt. |
| 1 | The bit ALMS in RTT_SR asserts interrupt. |

**Bits 15:0 – RTPRES[15:0]** Real-time Timer Prescaler Value
Defines the number of SLCK periods required to increment the RTT. The RTTINCIEN bit must be cleared prior to writing a new RTPRES value.

RTPRES is defined as follows:

- RTPRES = 0: The prescaler period is equal to $2^{16}$ * SLCK periods.
- RTPRES = 1 or 2: forbidden.
- RTPRES ≠ 0,1 or 2: The prescaler period is equal to RTPRES * SLCK periods.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 28.5.2 Real-time Timer Alarm Register

**Name:** RTT_AR
**Offset:** 0x04
**Reset:** 0xFFFFFFFF
**Property:** Read/Write

The alarm interrupt must be disabled (ALMIEN must be cleared in RTT_MR) when writing a new ALMV value.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ALMV[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ALMV[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ALMV[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ALMV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 31:0 – ALMV[31:0]** Alarm Value
When the CRTV value in RTT_VR equals the ALMV field, the ALMS flag is set in RTT_SR. As soon as the ALMS flag rises, the CRTV value equals ALMV+1 (refer to the figure *RTT Counting* above).

### 28.5.3 Real-time Timer Value Register

**Name:** RTT_VR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CRTV[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CRTV[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CRTV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRTV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CRTV[31:0]** Current Real-time Value
Returns the current value of the RTT.
As CRTV can be updated asynchronously, it must be read twice at the same value.

### 28.5.4 Real-time Timer Status Register

**Name:** RTT_SR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RTTINC | ALMS |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – RTTINC** Prescaler Roll-over Status (cleared on read)

| Value | Description |
|---|---|
| 0 | No prescaler roll-over occurred since the last read of the RTT_SR. |
| 1 | Prescaler roll-over occurred since the last read of the RTT_SR. |

**Bit 0 – ALMS** Real-time Alarm Status (cleared on read)

| Value | Description |
|---|---|
| 0 | The Real-time Alarm has not occurred since the last read of RTT_SR. |
| 1 | The Real-time Alarm occurred since the last read of RTT_SR. |

# 29. General Purpose Backup Registers (GPBR)

## 29.1 Description

The System Controller embeds 128 bits of General Purpose Backup registers organized as 8 32-bit registers.

It is possible to generate an immediate clear of the content of General Purpose Backup registers 0 to 3 (first half) if a Low-power Debounce event is detected on one of the wakeup pins, WKUP0 or WKUP1. The content of the other General Purpose Backup registers (second half) remains unchanged.

The Supply Controller module must be programmed accordingly. In the register SUPC_WUMR in the Supply Controller module, LPDBCCLR, LPDBCEN0 and/or LPDBCEN1 bit must be configured to 1 and LPDBC must be other than 0.

If a Tamper event has been detected, it is not possible to write to the General Purpose Backup registers while the LPDBCS0 or LPDBCS1 flags are not cleared in the Supply Controller Status Register (SUPC_SR).

## 29.2 Embedded Characteristics

- 128 bits of General Purpose Backup Registers
- Immediate Clear on Tamper Event

© 2021 Microchip Technology Inc.
and its subsidiaries

## 29.3 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SYS_GPBRx | 7:0 | GPBR_VALUE[7:0] | | | | | | | |
| | | 15:8 | GPBR_VALUE[15:8] | | | | | | | |
| | | 23:16 | GPBR_VALUE[23:16] | | | | | | | |
| | | 31:24 | GPBR_VALUE[31:24] | | | | | | | |

### 29.3.1 General Purpose Backup Register x

**Name:** SYS_GPBRx
**Offset:** 0x00
**Reset:** 0
**Property:** R/W

These registers are reset at first power-up and on each loss of VDDIO.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | GPBR_VALUE[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GPBR_VALUE[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GPBR_VALUE[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GPBR_VALUE[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – GPBR_VALUE[31:0]** Value of GPBR x
If a Tamper event has been detected, it is not possible to write GPBR_VALUE as long as the LPDBCS0 or LPDBCS1 flag has not been cleared in the Supply Controller Status Register (SUPC_SR).

# 30. Clock Generator

## 30.1 Description

The Clock Generator user interface is embedded within the Power Management Controller and is described in Power Management Controller (PMC) User Interface. However, the Clock Generator registers are named CKGR_.

## 30.2 Embedded Characteristics

The Clock Generator is comprised of the following:

- A low-power 32.768 kHz crystal oscillator with Bypass mode
- A low-power Slow RC oscillator (32 kHz typical)
- A 3 to 20 MHz Main crystal oscillator with Bypass mode
- A Main RC oscillator. Three output frequencies can be selected: 4/8/12 MHz. By default 12 MHz is selected. 8 MHz and 12 MHz are factory-trimmed.
- A 480 MHz UTMI PLL, providing a clock for the USB high-speed controller
- A 160 to 500 MHz programmable PLL (input from 8 to 32 MHz)

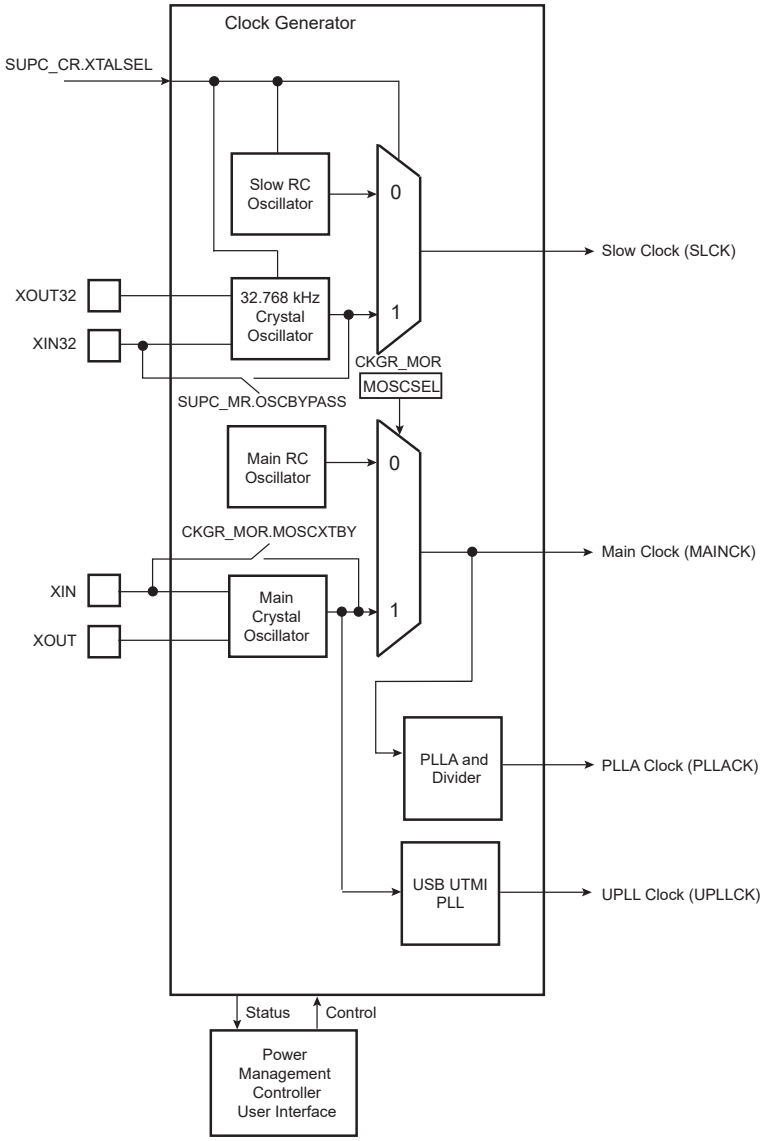
It provides the following clocks:

- SLCK — Slow clock. The only permanent clock within the system
- MAINCK — output of the Main clock oscillator selection: either the Main crystal oscillator or Main RC oscillator
- PLLACK — output of the divider and 160 to 500 MHz programmable PLL (PLLA)
- UPLLCK — output of the 480 MHz UTMI PLL (UPLL)

© 2021 Microchip Technology Inc.
and its subsidiaries

## 30.3 Block Diagram

**Figure 30-1. Clock Generator Block Diagram**




## 30.4 Slow Clock

The Supply Controller embeds a slow clock generator that is supplied with the VDDIO power supply. As soon as VDDIO is supplied, both the 32.768 kHz crystal oscillator and the Slow RC oscillator are powered, but only the Slow RC oscillator is enabled. This allows the Slow clock (SLCK) to be valid in a short time (about 100 µs).

SLCK is generated either by the 32.768 kHz crystal oscillator or by the Slow RC oscillator.

To select the clock source, the selection is made via the XTALSEL bit in the Supply Controller Control Register (SUPC_CR).

### 30.4.1 Slow RC Oscillator (32 kHz typical)

By default, the Slow RC oscillator is enabled and selected as a source of SLCK.

Compared to the 32.768 kHz crystal oscillator, this oscillator offers a faster startup time and is less exposed to the external environment, as it is fully integrated. However, its output frequency is subject to larger variations with supply voltage, temperature and manufacturing process. Therefore, the user must take these variations into account when this oscillator is used as a time base (startup counter, frequency monitor, etc.). Refer to the section “Electrical Characteristics”.

This oscillator is disabled by clearing the SUPC_CR.XTALSEL.

### 30.4.2 32.768 kHz Crystal Oscillator

By default, the 32.768 kHz oscillator is disabled. To use this oscillator, the XIN32 and XOUT32 pins must be connected to a 32.768 kHz crystal or to a ceramic resonator. Refer to the section “Electrical Characteristics” for appropriate loading capacitors selection on XIN32 and XOUT32.

Note that the user is not obliged to use the 32.768 kHz crystal oscillator and can use the Slow RC oscillator instead. Using the 32.768 kHz crystal oscillator provides a more accurate frequency than the Slow RC oscillator.

To select the 32.768 kHz crystal oscillator as the source of SLCK, the bit SUPC_CR.XTALSEL must be set. This results in a sequence which first configures the PIO lines multiplexed with XIN32 and XOUT32 to be driven by the crystal oscillator, then enables the 32.768 kHz crystal oscillator and then disables the Slow RC oscillator to save power. The switch of SLCK source is glitch-free.

Reverting to the Slow RC oscillator is only possible by shutting down the VDDIO power supply. If the user does not need the 32.768 kHz crystal oscillator, the XIN32 and XOUT32 pins can be left unconnected since by default the XIN32 and XOUT32 system I/O pins are in PIO input mode with pullup after reset.

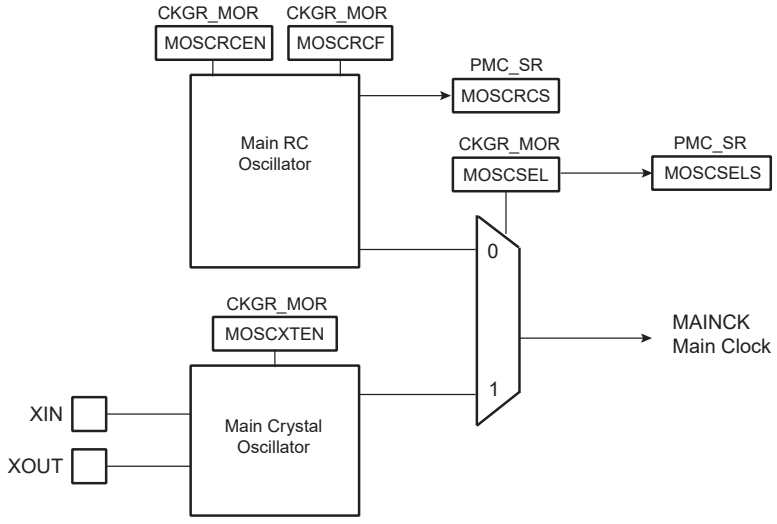
The user can also set the 32.768 kHz crystal oscillator in Bypass mode instead of connecting a crystal. In this case, the user must provide the external clock signal on XIN32. For input characteristics of the XIN32 pin, refer to the section “Electrical Characteristics”. To enter Bypass mode, the OSCBYPASS bit of the Supply Controller Mode register (SUPC_MR) must be set prior to setting SUPC_CR.XTALSEL.

## 30.5 Main Clock

The Main clock (MAINCK) has two sources:

- A Main RC oscillator (4/8/12 MHz) with a fast startup time and that is selected by default to start the system
- A Main crystal oscillator with Bypass mode

**Figure 30-2. Main Clock (MAINCK) Block Diagram**




### 30.5.1 Main RC Oscillator

After reset, the Main RC oscillator is enabled with the 12 MHz frequency selected. This oscillator is selected as the source of MAINCK. MAINCK is the default clock selected to start the system.

Only the 8/12 MHz RC oscillator frequencies are calibrated in production. Refer to the section “Electrical Characteristics”.

The software can disable or enable the Main RC oscillator with the MOSCRCEN bit in the Clock Generator Main Oscillator Register (CKGR_MOR).

The output frequency of the Main RC oscillator can be selected among 4, 8 or 12 MHz. Selection is done by configuring the field MOSCRCF in CKGR_MOR. When changing the frequency selection, the MOSCRCS bit in the Power Management Controller Status Register (PMC_SR) is automatically cleared and MAINCK is stopped until the oscillator is stabilized. Once the oscillator is stabilized, MAINCK restarts and PMC_SR.MOSCRCS is set. Note that enabling the Main RC oscillator (MOSCRCEN = 1) and changing its frequency (MOSCRCF) at the same time is not allowed.

This oscillator must be enabled first and its frequency changed in a second step.

When disabling the Main RC oscillator by clearing the CKGR_MOR.MOSCRCEN bit, the PMC_SR.MOSCRCS bit is automatically cleared, indicating that the oscillator is OFF.

Setting the MOSCRCS bit in the Power Management Controller Interrupt Enable Register (PMC_IER) triggers an interrupt to the processor.

### 30.5.2 Main RC Oscillator Frequency Adjustment

The 8 MHz and 12 MHz frequencies are factory-centered to the typical values by using Flash calibration bits (refer to the “Electrical Characteristics” chapter).

The Flash calibration bits setting the Main RC oscillator frequency to 8 MHz and 12 MHz vary from device to device. To get a starting point when changing the CAL8 or CAL12 fields, it is recommended to first read their corresponding Flash calibration bits in the Flash Controller.

The user can adjust the value of the Main RC oscillator frequency by modifying the trimming values done in production on 8 MHz and 12 MHz. This may be used to compensate frequency drifts due to temperature or voltage. The values stored in the Flash cannot be erased by a Flash erase command or by the ERASE signal. Values written by the user application in the Oscillator Calibration Register (PMC_OCR) are reset after each power-up or peripheral reset.

By default, SEL4/SEL8/SEL12 are cleared, so the Main RC oscillator is driven with the factory-programmed Flash calibration bits which are programmed during chip production.

In order to calibrate the oscillator lower frequency, SEL4 must be set to ‘1’ and a valid frequency value must be configured in CAL4. Likewise, SEL8/12 must be set to ‘1’ and a trim value must be configured in CAL8/12 in order to adjust the other frequencies of the oscillator.

It is possible to adjust the oscillator frequency while operating from this oscillator. For example, when running on lowest frequency, it is possible to change the CAL4 value if SEL4 is set in PMC_OCR.

At any time, the user can measure the main RC oscillator output frequency by means of the Main Frequency Counter (refer to "Main Frequency Counter"). Once the frequency measurement is done, the main RC oscillator calibration field (CALx) can be adjusted accordingly to correct this oscillator output frequency.

### 30.5.3 Main Crystal Oscillator

After reset, the Main crystal oscillator is disabled and is not selected as the source of MAINCK.

As the source of MAINCK, the Main crystal oscillator provides a very precise frequency. The software enables or disables this oscillator in order to reduce power consumption via CKGR_MOR.MOSCXTEN.

When disabling this oscillator by clearing the CKGR_MOR.MOSCXTEN, PMC_SR.MOSCXTS is automatically cleared, indicating the oscillator is off.

When enabling this oscillator, the user must initiate the startup time counter. The startup time depends on the characteristics of the external device connected to this oscillator.

When CKGR_MOR.MOSCXTEN and CKGR_MOR.MOSCXTST are written to enable this oscillator, the PIO lines multiplexed with XIN and XOUT are driven by the Main crystal oscillator. PMC_SR.MOSCXTS is cleared and the counter starts counting down on SLCK divided by 8 from the CKGR_MOR.MOSCXTST value. Since the CKGR_MOR.MOSCXTST value is coded with 8 bits, the startup time can be programmed up to 65536 SLCKperiods, corresponding to about 62 ms when running at 32.768 kHz.

© 2021 Microchip Technology Inc. and its subsidiaries

When the startup time counter reaches ‘0’, PMC_SR.MOSCXTS is set, indicating that the oscillator is stabilized. Setting the MOSCXTS bit in the Interrupt Mask Register (PMC_IMR) can trigger an interrupt to the processor.

### 30.5.4 Main Clock Source Selection

The source of MAINCK can be selected from the following:

- The Main RC oscillator
- The Main crystal oscillator
- An external clock signal provided on the XIN input (Bypass mode of the Main crystal oscillator)

The advantage of the Main RC oscillator is its fast startup time. By default, this oscillator is selected to start the system and it must be selected prior to entering Wait mode.

The advantage of the Main crystal oscillator is its high level of accuracy.

The selection of the oscillator is made with bit CKGR_MOR.MOSCSEL. The switchover of the MAINCK source is glitch-free, so there is no need to run MCK out of SLCK, PLLACK or UPLLCK in order to change the selection. PMC_SR.MOSCSELS indicates when the switch sequence is done.

Setting PMC_IMR.MOSCSELS triggers an interrupt to the processor.

**MAINCK Switching Sequence**

When switching the Main Clock MAINCK source from the Main Crystal oscillator to the Main RC oscillator it is mandatory to follow the below steps:

- Start the Main RC oscillator and keep MAINCK on the Main Crystal Oscillator (this step is optional at startup as it is the default configuration)
- Switch MAINCK to the Main RC oscillator and keep the Main Crystal Oscillator on
- Switch off the Main Crystal Oscillator is a third separate step

### 30.5.5 Bypassing the Main Crystal Oscillator

Prior to bypassing the Main crystal oscillator, the external clock frequency provided on the XIN pin must be stable and within the values specified in the XIN Clock characteristics in the section “Electrical Characteristics”.

The sequence is as follows:

1. Ensure that an external clock is connected on XIN.
2. Enable the bypass by setting CKGR_MOR.MOSCXTBY.
3. Disable the Main crystal oscillator by clearing CKGR_MOR.MOSCXTEN.

### 30.5.6 Main Frequency Counter

The Main frequency counter measures the Main RC oscillator and the Main crystal oscillator against the SLCK and is managed by CKGR_MCFR.

During the measurement period, the Main frequency counter increments at the speed of the clock defined by the bit CKGR_MCFR.CCSS.

A measurement is started in the following cases:

- When CKGR_MCFR.RCMEAS is written to ‘1’.
- When the Main RC oscillator is selected as the source of MAINCK and when this oscillator is stable (i.e., when the MOSCRCS bit is set)
- When the Main crystal oscillator is selected as the source of MAINCK and when this oscillator is stable (i.e., when the MOSCXTS bit is set)
- When MAINCK source selection is modified

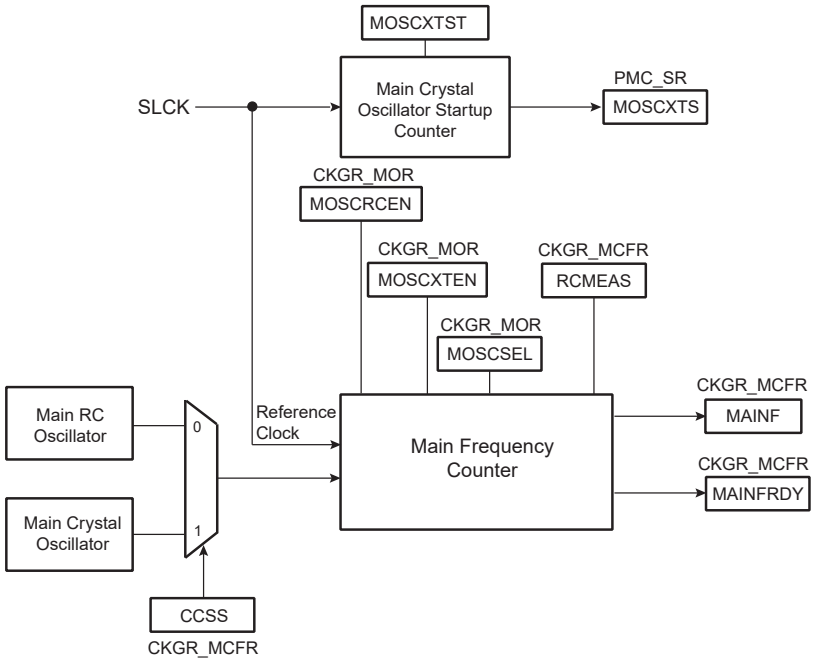
The measurement period ends at the 16th falling edge of SLCK, the MAINFRDY bit in CKGR_MCFR is set and the counter stops counting. Its value can be read in the MAINF field of CKGR_MCFR and gives the number of clock cycles during 16 periods of SLCK, so that the frequency of the Main RC oscillator or Main crystal oscillator can be determined.

If switching the source of MAINCK to the Main crystal oscillator from the Main RC oscillator, follow the programming sequence below to ensure that the oscillator is present and that its frequency is valid:

1. Enable the Main crystal oscillator by setting CKGR_MOR.MOSCXTEN. Configure the CKGR_MOR. MOSCXTST field with the Main crystal oscillator startup time as defined in the section “Electrical Characteristics”.
2. Wait for PMC_SR.MOSCXTS flag to rise, indicating the end of a startup period of the Main crystal oscillator.
3. Select the Main crystal oscillator as the source clock of the Main frequency counter by setting CKGR_MCFR.CCSS.
4. Initiate a frequency measurement by setting CKGR_MCFR.RCMEAS.
5. Read CKGR_MCFR.MAINFRDY until its value equals 1.
6. Read CKGR_MCFR.MAINF and compute the value of the Main crystal frequency.

If the MAINF value is valid, software can switch MAINCK to the Main crystal oscillator. Refer to "Main Clock Source Selection".

**Figure 30-3. Main Frequency Counter Block Diagram**

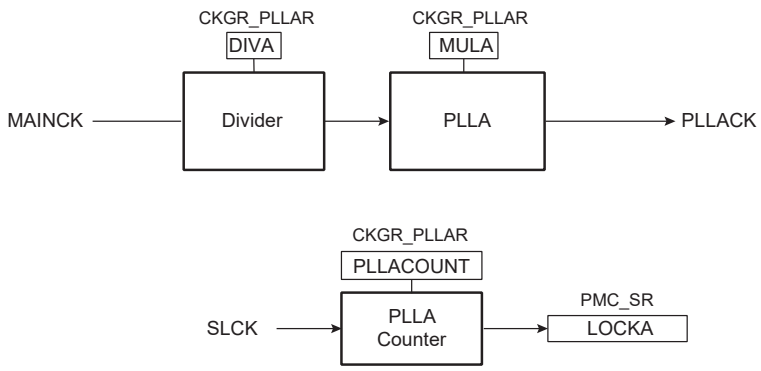



## 30.6 PLLA Clock

The PLLA clock (PLLACK) is generated from MAINCK by the PLLA and a predivider. This combination allows a wide range of frequencies to be selected on either MCK, HCLK or the PCKx outputs.

The following figure shows the block diagram of the dividers and PLLA blocks.

**Figure 30-4. Divider and PLLA Block Diagram**



### 30.6.1 Divider and Phase Lock Loop Programming

The divider can be set between 1 and 255 in steps of 1. When a divider field (DIV) is cleared, the output of the corresponding divider and the PLL output is a continuous signal at level 0. On reset, each DIV field is cleared, thus the corresponding PLL input clock is stuck at ‘0’.

The PLL (PLLA) allows multiplication of the divider’s outputs. The PLL clock signal has a frequency that depends on the respective source signal frequency and on the parameters DIV (DIVA) and MUL (MULA). The factor applied to the source signal frequency is (MUL + 1)/DIV. When MUL is written to ‘0’ or DIV = 0, the PLL is disabled and its power consumption is saved. Note that there is a delay of two SLCK clock cycles between the disable command and the real disable of the PLL. Re-enabling the PLL can be performed by writing a value higher than ‘0’ in the MUL field and DIV higher than ‘0’.

Whenever the PLL is re-enabled or one of its parameters is changed, the LOCK (LOCKA) bit in PMC_SR is automatically cleared. The values written in the PLLCOUNT field (PLLACOUNT) in CKGR_PLLR (CKGR_PLLAR) are loaded in the PLL counter. The PLL counter then decrements at the speed of SLCK until it reaches ‘0’. At this time, PMC_SR.LOCK is set and can trigger an interrupt to the processor. The user has to load the number of SLCK cycles required to cover the PLL transient time into the PLLCOUNT field.

To avoid programming the PLL with a multiplication factor that is too high, the user can saturate the multiplication factor value sent to the PLL by setting the PLLA_MMAX field in the PLL Maximum Multiplier Value Register (PMC_PMMR).

It is forbidden to change the MAINCK characteristics (oscillator selection, frequency adjustment of the Main RC oscillator) when:

- MAINCK is selected as the PLLA clock source, and
- MCK is sourced from PLLA.

To change the MAINCK characteristics, the user must:

1. Switch the MCK source to MAINCK by writing a ‘1’ to PMC_MCKR.CSS.
2. Change the Main RC oscillator frequency (MOSCRCF) or oscillator selection (MOSCSEL) in CKGR_MOR.
3. Wait for MOSCRCS (if frequency changes) or MOSCSELS (if oscillator selection changes) in PMC_SR.
4. Disable and then enable the PLL.
5. Wait for the LOCK flag in PMC_SR.
6. Switch back MCK to the PLLA by writing the appropriate value to PMC_MCKR.CSS.

## 30.7 UTMI PLL Clock

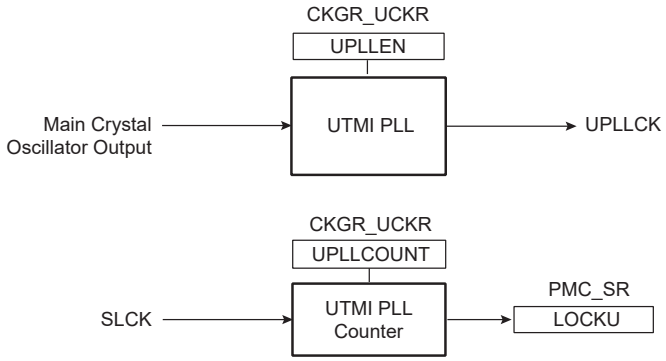
The source of the UTMI PLL (UPLL) is the Main Crystal oscillator. The UPLL provides the UTMI PLL Clock (UPLLCK) and UPLLCKDIV clock signals.

The UPLL has two possible multiplying factors: x40 and x30. To generate UPLLCK at 480 MHz (typical USB case), this leads to two possible crystal oscillator frequencies: 12 or 16 MHz. The crystal oscillator frequency (12 or 16 MHz) must be programmed in UTMI_CKTRIM.FREQ prior to enabling the UPLL.

When the UPLL is enabled by writing a ‘1’ to bit UPLLEN in the UTMI Clock Register (CKGR_UCKR), the LOCKU bit in PMC_SR is automatically cleared. The values written in the PLLCOUNT field in CKGR_UCKR are loaded in the UTMI PLL counter. The UTMI PLL counter then decrements at the speed of SLCK divided by 8 until it reaches ‘0’. At this time, the LOCKU bit is set in PMC_SR and can trigger an interrupt to the processor. The user has to load the number of SLCK cycles required to cover the UTMI PLL transient time into the PLLCOUNT field.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 30-5. UTMI PLL Block Diagram**



# 31. Power Management Controller (PMC)

## 31.1 Description

The Power Management Controller (PMC) optimizes power consumption by controlling all system and user peripheral clocks. The PMC enables/disables the clock inputs to many of the peripherals and the Cortex-M7 processor.

The Supply Controller selects either the Slow RC oscillator or the 32.768 kHz crystal oscillator as the source of SLCK. The unused oscillator is disabled automatically so that power consumption is optimized.

By default, at startup, the chip runs out of MCK using the Main RC oscillator running at 12 MHz.

## 31.2 Embedded Characteristics

The Power Management Controller provides the following clocks:

- Master Clock (MCK), programmable from a few hundred Hz to the maximum operating frequency of the device. It is available to the modules running permanently, such as the Enhanced Embedded Flash Controller
- Processor Clock (HCLK), automatically switched off when entering the processor in Sleep mode
- Free-running processor Clock (FCLK)
- The Cortex-M7 SysTick external clock
- USB Clock (USB_48M), required by the USB peripheral
- Peripheral Clocks with independent ON/OFF control, provided to the peripherals
- Programmable Clock Outputs (PCKx), selected from the clock generator outputs to drive the device PCK pins
- Clock sources independent of MCK and HCLK, provided by internal PCKx for USART, UART, TC, Embedded Trace Macrocell (ETM) and CAN Clocks
- Generic Clock (GCLK) with controllable division and ON/OFF control, independent of MCK and HCLK. Provided to selected peripherals.

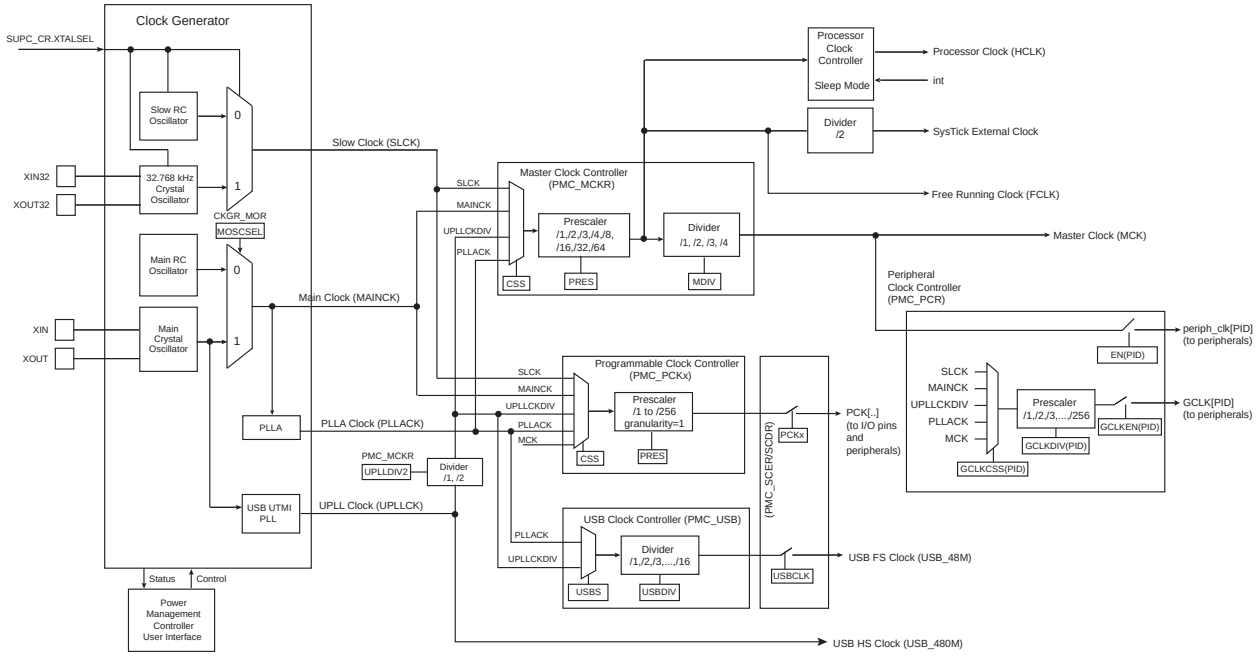
The Power Management Controller also provides the following features on clocks:

- A Main crystal oscillator failure detector
- A 32.768 kHz crystal oscillator frequency monitor
- A frequency counter on Main crystal oscillator or Main RC oscillator
- An on-the-fly adjustable Main RC oscillator frequency

© 2021 Microchip Technology Inc.
and its subsidiaries

## 31.3 Block Diagram

**Figure 31-1. General Clock Distribution Block Diagram**




## 31.4 Master Clock Controller

The Master Clock Controller provides the Master Clock (MCK) with the selection and division of the clock generator's output signals. MCK is the source clock of the peripheral clocks.

The clock to be selected between SLCK, MAINCK, PLLACK and UPLLCKDIV is configured in PMC_MCKR.CSS. The prescaler supports the 1, 2, 3, 4, 8, 16, 32, 64 division factors and is configured using PMC_MCKR.PRES.

Each time PMC_MCKR is configured to define a new MCK, the MCKRDY bit is cleared in PMC_SR. It reads ‘0’ until MCK is established. Then, the MCKRDY bit is set and can trigger an interrupt to the processor. This feature is useful when switching from a high-speed clock to a lower one to inform the software when the change is completed.

**Note:** Users cannot modify MDIV and CSS at the same access. Each field must be modified separately with a wait for the MCKRDY flag between the first field modification and the second field modification.

## 31.5 Processor Clock Controller

The PMC features a Processor Clock (HCLK) Controller that implements the processor Sleep mode. HCLK can be disabled by executing the WFI (WaitForInterrupt) or the WFE (WaitForEvent) processor instruction while the LPM bit is at ‘0’ in the PMC Fast Startup Mode register (PMC_FSMR).

HCLK is enabled after a reset and is automatically re-enabled by any enabled interrupt. The processor Sleep mode is entered by disabling HCLK, which is automatically re-enabled by any enabled fast or normal interrupt, or by the reset of the product.

When processor Sleep mode is entered, the current instruction is finished before the clock is stopped, but this does not prevent data transfers from other masters of the system bus.

## 31.6 SysTick External Clock

When the processor selects the SysTick external clock, the calibration value is fixed to 150000. This allows the generation of a time base of 1 ms with the SysTick clock at the maximum frequency on MCK divided by 2.

Refer to the section “ARM Cortex-M7 Processor” for details on selecting the SysTick external clock.

**Related Links**

15. ARM Cortex-M7 (ARM)

## 31.7 USB Full-speed Clock Controller

The user can select the PLLA or the UPLL output as the USB FS clock (USB_48M) by writing a ‘1’ to the USBS bit in the USB Clock Register (PMC_USB). The user then must program the corresponding PLL to generate an appropriate frequency depending on the USBDIV bit in PMC_USB.

When PMC_SR.LOCKA and PMC_SR.LOCKU are set to ‘1’, the PLLA and UPLL are stable. Then, USB_48M can be enabled by setting the USBCLK bit in the System Clock Enable register (PMC_SCER). To save power on this peripheral when not used, the user can set the USBCLK bit in the System Clock Disable register (PMC_SCDR). The USBCLK bit in the System Clock Status register (PMC_SCSR) gives the status of this clock. The USB port requires both the USB clock signal and the peripheral clock. The USB peripheral clock is controlled by means of the Master Clock Controller.

## 31.8 Core and Bus Independent Clocks for Peripherals

The following table lists the peripherals that require a PCKx clock to operate while the core, bus and peripheral clock frequencies are modified, thus providing communications at a bit rate which is independent for the core/bus/peripheral clock. This mode of operation is possible by using the internally generated independent clock sources.

Internal clocks can be independently selected between SLCK, MAINCK, any available PLL clock, and MCK by configuring PMC_PCKx.CSS. The independent clock sources can be also divided by configuring PMC_PCKx.PRES.

Each internal clock signal (PCKx) can be enabled and disabled by writing a ‘1’ to the corresponding PMC_SCER.PCKx and PMC_SCDR.PCKx, respectively. The status of the internal clocks are given in PMC_SCSR.PCKx.

The status flag PMC_SR.PCKRDYx indicates that the programmable internal clock has been programmed in the Programmable clock registers.

The independent clock source must also be selected in each peripheral in the Clock Assignments table to operate communications, timings, etc without influencing the frequency of the core/bus/peripherals (except frequency limitations listed in each peripheral).

**Table 31-1. Clock Assignments**

| Clock Name | Peripheral |
|---|---|
| PCK3 | ETM |
| PCK4 | UARTx/USARTx |
| PCK5 | MCANx |
| PCK6 | TCx |
| PCK7 | TC0 |

**Note:** USB, GMAC and MLB do not require PCKx to operate independently of core and bus peripherals.

## 31.9 Peripheral and Generic Clock Controller

The PMC controls the clocks of the embedded peripherals by means of the Peripheral Control register (PMC_PCR). With this register, the user can enable and disable the different clocks used by the peripherals:

- Peripheral clocks (periph_clk[PID]), routed to every peripheral and derived from the master clock (MCK), and
- Generic clocks (GCLK[PID]), routed to I2SC0 and I2SC1. These clocks are independent of the core and bus clocks (HCLK, MCK and periph_clk[PID]). They are generated by selection and division of the following sources:

© 2021 Microchip Technology Inc. and its subsidiaries

SLCK, MAINCK, UPLLCKDIV, PLLACK and MCK. Refer to the description of each peripheral for the limitation to be applied to GCLK[PID] compared to periph_clk[PID].

To configure a peripheral's clocks, PMC_PCR.CMD must be written to '1' and PMC_PCR.PID must be written with the index of the corresponding peripheral. All other configuration fields must be correctly set.

To read the current clock configuration of a peripheral, PMC_PCR.CMD must be written to '0' and PMC_PCR.PID must be written with the index of the corresponding peripheral regardless of the values of other fields. This write does not modify the configuration of the peripheral. The PMC_PCR can then be read to know the configuration status of the corresponding PID.

The user can also enable and disable these clocks by configuring the Peripheral Clock Enable (PMC_PCERx) and Peripheral Clock Disable (PMC_PCDRx) registers. The status of the peripheral clock activity can be read in the Peripheral Clock Status registers (PMC_PCSRx).

When a peripheral or a generic clock is disabled, it is immediately stopped. These clocks are disabled after a reset.

To stop a peripheral clock, it is recommended that the system software wait until the peripheral has executed its last programmed operation before disabling the clock. This is to avoid data corruption or erroneous behavior of the system.

The bit number in PMC_PCERx, PMC_PCDRx, and PMC_PCSRx is the Peripheral Identifier defined at the product level. The bit number corresponds to the interrupt source number assigned to the peripheral.

## 31.10 Asynchronous Partial Wakeup

### 31.10.1 Description

The asynchronous partial wakeup wakes up a peripheral in a fully asynchronous way when activity is detected on the communication line. The asynchronous partial wakeup function automatically manages the peripheral clock. It reduces overall power consumption of the system by clocking peripherals only when needed.

Asynchronous partial wakeup can be enabled in Wait mode (SleepWalking), or in Active mode.

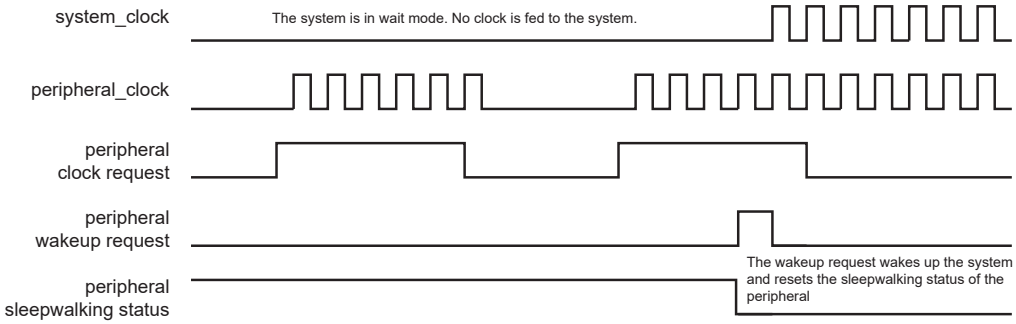
Only the following peripherals can be configured with asynchronous partial wakeup: UARTx and TWIHSx.

The peripheral selected for asynchronous partial wakeup must first be configured so that its clock is enabled. To do so, write a '1' to the appropriate PIDx bit in PMC_PCER registers.

### 31.10.2 Asynchronous Partial Wakeup in Wait Mode (SleepWalking)

When the system is in Wait mode, all clocks of the system except SLCK are stopped. When an asynchronous clock request from a peripheral occurs, the PMC partially wakes up the system to feed the clock only to this peripheral. The rest of the system is not fed with the clock, thus optimizing power consumption. Finally, depending on user-configurable conditions, the peripheral either wakes up the whole system if these conditions are met or stops the peripheral clock until the next clock request. If a wakeup request occurs, SleepWalking is automatically disabled until the user instructs the PMC to enable SleepWalking. This is done by writing a '1' to PIDx in the PMC SleepWalking Enable register (PMC_SLPWK_ER).

**Figure 31-2. SleepWalking Waveforms**



#### 31.10.2.1 Configuration Procedure

Before configuring SleepWalking for a peripheral, check that the PIDx bit in PMC_PCSR is set. This ensures that the peripheral clock is enabled.

The steps to enable SleepWalking for a peripheral are the following:

1. Check that the corresponding PIDx bit in the PMC SleepWalking Activity Status register (PMC_SLPWK_ASR) is set to ‘0’. This ensures that the peripheral has no activity in progress.
2. Enable SleepWalking for the peripheral by writing a ‘1’ to the corresponding PIDx bit in the PMC_SLPWK_ER.
3. Check that the corresponding PIDx bit in PMC_SLPWK_ASR is set to ‘0’. This ensures that no activity has started during the enable phase.
4. In the PMC_SLPWK_ASR, if the corresponding PIDx bit is set, SleepWalking must be immediately disabled by writing a ‘1’ to the PIDx bit in the PMC SleepWalking Disable register (PMC_SLPWK_DR). Wait for the end of peripheral activity before reinitializing the procedure.
   If the corresponding PIDx bit is set to ‘0’, then the peripheral clock is disabled and the system can then be placed in Wait mode.

Before entering Wait mode, check that the AIP bit in the PMC SleepWalking Activity In Progress Register (PMC_SLPWK_AIPR) is cleared. This ensures that none of the peripherals is currently active.

**Note:** When SleepWalking for a peripheral is enabled and the core is running (system not in Wait mode), the peripheral must not be accessed before a wakeup of the peripheral is performed.

### 31.10.3 Asynchronous Partial Wakeup in Active Mode

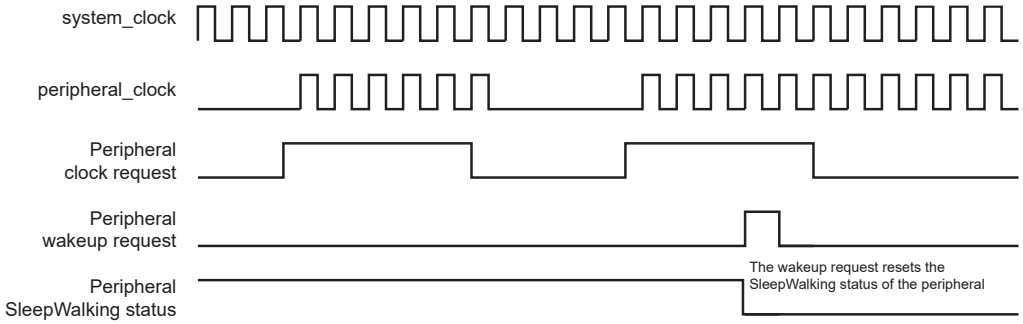
When the system is in Active mode, peripherals enabled for asynchronous partial wakeup have their respective clocks stopped until the peripherals request a clock. When a peripheral requests the clock, the PMC provides the clock without processor intervention.

The triggering of the peripheral clock request depends on conditions which can be configured for each peripheral. If these conditions are met, the peripheral asserts a request to the PMC. The PMC disables the Asynchronous Partial Wakeup mode of the peripheral and provides the clock to the peripheral until the user instructs the PMC to re-enable partial wakeup on the peripheral. This is done by setting PMC_SLPWK_ER.PIDx.

If the conditions are not met, the peripheral clears the clock request and the PMC stops the peripheral clock until the clock request is reasserted by the peripheral.

**Note:** Configuring Asynchronous Partial Wake-up mode requires the same registers as Sleep-Walking mode.

**Figure 31-3. Asynchronous Partial Wake-up in Active Mode**




#### 31.10.3.1 Configuration Procedure

Before configuring the asynchronous partial wakeup function of a peripheral, check that the PIDx bit in PMC_PCSR is set. This ensures that the peripheral clock is enabled.

The steps to enable the asynchronous partial wakeup function of a peripheral are the following:

1. Check that the corresponding PIDx bit in the PMC SleepWalking Activity Status register (PMC_SLPWK_ASR) is set to ‘0’. This ensures that the peripheral has no activity in progress.
2. Enable the asynchronous partial wakeup function of the peripheral by writing a ‘1’ to the corresponding PIDx bit in the PMC_SLPWK_ER.

3. Check that the corresponding PIDx bit in PMC_SLPWK_ASR is set to ‘0’. This ensures that no activity has started during the enable phase.

If an activity has started during the enable phase, the asynchronous partial wakeup function must be immediately disabled by writing a ‘1’ to the PIDx bit in the PMC SleepWalking Disable register (PMC_SLPWK_DR). Wait for the end of peripheral activity before reinitializing the procedure.

## 31.11 Free-running Processor Clock

The free-running Processor clock (FCLK) used for sampling interrupts and clocking debug blocks ensures that interrupts can be sampled, and sleep events can be traced, while the processor is sleeping.

## 31.12 Programmable Clock Output Controller

The PMC controls three signals to be output on the external pins PCKx. Each signal can be independently programmed via the Programmable Clock registers (PMC_PCKx).

PCKx can be independently selected between SLCK, MAINCK, PLLACK, UPLLCKDIV and MCK by configuring PMC_PCKx.CSS. Each output signal can also be divided by 1 to 256 by configuring PMC_PCKx.PRES.

Each output signal can be enabled and disabled by writing a ‘1’ to the corresponding bits PMC_SCER.PCKx and PMC_SCDR.PCKx, respectively. The status of the active programmable output clocks is given in PMC_SCSR.PCKx.

The status flag PMC_SR.PCKRDYx indicates that PCKx is actually what has been programmed in registers PMC_PCKx.

As the Programmable Clock Controller does not manage with glitch prevention when switching clocks, it is strongly recommended to disable PCKx before any configuration change and to re-enable it after the change is performed.

## 31.13 Fast Startup

At exit from Wait mode, the device allows the processor to restart in several microseconds only if the C-code function that manages the Wait mode entry and exit is linked to and executed from on-chip SRAM.

The fast startup time cannot be achieved if the first instruction after an exit is located in the embedded Flash.

If fast startup is not required, or if the first instruction after exit from Wait mode is located in embedded Flash, see "Startup from Embedded Flash".

To instruct the device to enter Wait mode, refer to section “Power Considerations”.

A fast startup occurs upon the detection of a programmed level on one of the 14 wakeup inputs (WKUP) or upon an active alarm from the RTC, RTT and USB Controller. The polarity of each of the 14 wakeup inputs is programmable in the PMC Fast Startup Polarity Register (PMC_FSPR).

**⚠WARNING** The duration of the WKUPx pins active level must be greater than four MAINCK cycles.

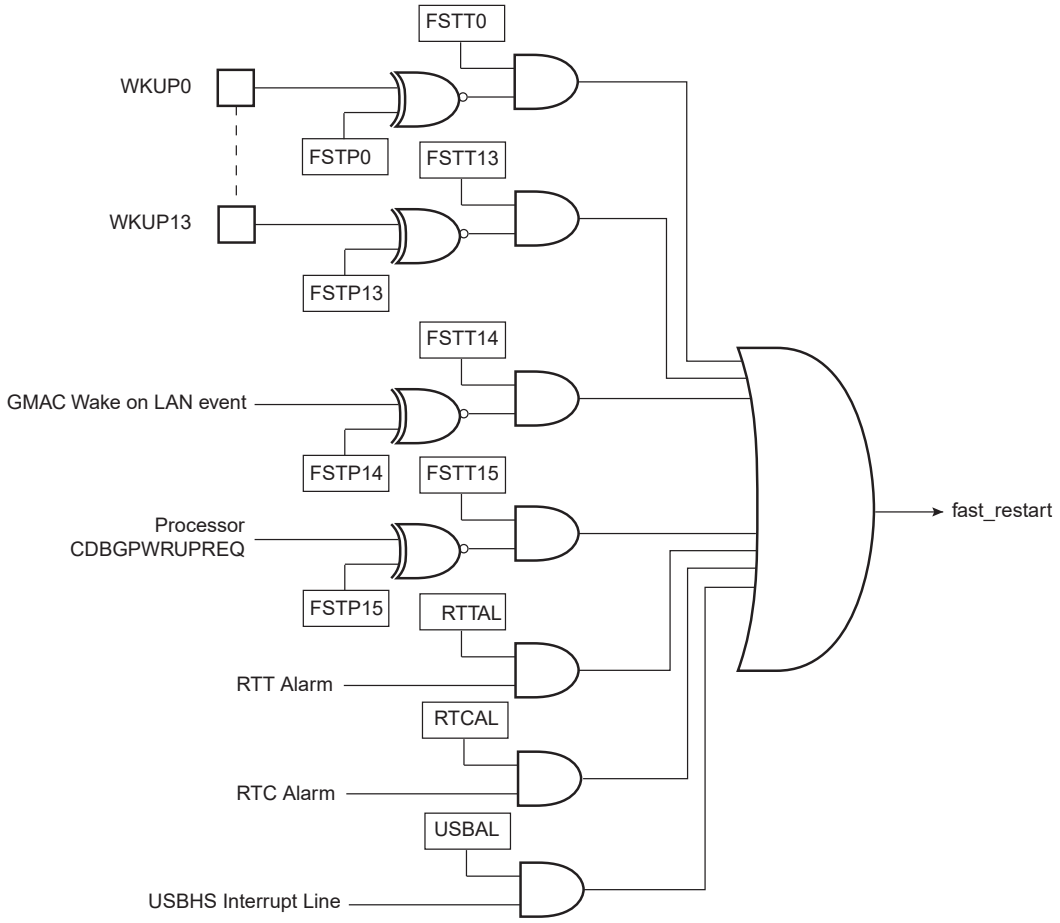
The fast startup circuitry, as shown in the following figure, is fully asynchronous and provides a fast startup signal to the PMC. As soon as the fast startup signal is asserted, the Main RC oscillator restarts automatically.

When entering Wait mode, the embedded Flash can be placed in one of the low-power modes (Deep-powerdown or Standby mode) with PMC_FSMR.FLPM. FLPM can be configured at any time and its value will be applied to the next Wait mode period.

The power consumption reduction is optimal when PMC_FSMR.FLPM is configured to ‘1’ (Deep-powerdown mode). If the field is configured to ‘0’ (Standby mode), the power consumption is slightly higher than in Deep-powerdown mode.

When PMC_FSMR.FLPM is configured to ‘2’, the Wait mode Flash power consumption is equivalent to that of the Active mode when there is no read access on the Flash.

**Figure 31-4. Fast Startup Circuitry**




Each wakeup input pin and alarm can be enabled to generate a fast startup event by setting the corresponding bit in PMC_FSMR.

The user interface does not provide any status for fast startup. The status can be read in the PIO Controller and the status registers of the RTC, RTTand USB Controller.

**Related Links**

7. Power Considerations

## 31.14 Startup from Embedded Flash

The inherent startup time of the embedded Flash cannot provide a fast startup of the system.

If system fast startup time is not required, the first instruction after a Wait mode exit can be located in the embedded Flash. Under these conditions, prior to entering Wait mode, the Flash controller must be programmed to perform access in 0 wait-state (refer to the embedded Flash controller section).

The procedure and conditions to enter Wait mode and the circuitry to exit Wait mode are strictly the same as fast startup (see "Fast Startup").

**Related Links**

22. Enhanced Embedded Flash Controller (EEFC)

## 31.15 Main Crystal Oscillator Failure Detection

The Main crystal oscillator failure detector monitors the Main crystal oscillator against the Slow RC oscillator and provides an automatic switchover of the MAINCK source to the Main RC oscillator in case of failure detection.

The failure detector can be enabled or disabled by configuring the CKGR_MOR.CFDEN, and it can also be disabled in either of the following cases:

- After a VDDCORE reset
- When the Main crystal oscillator is disabled (MOSCXTEN = 0)

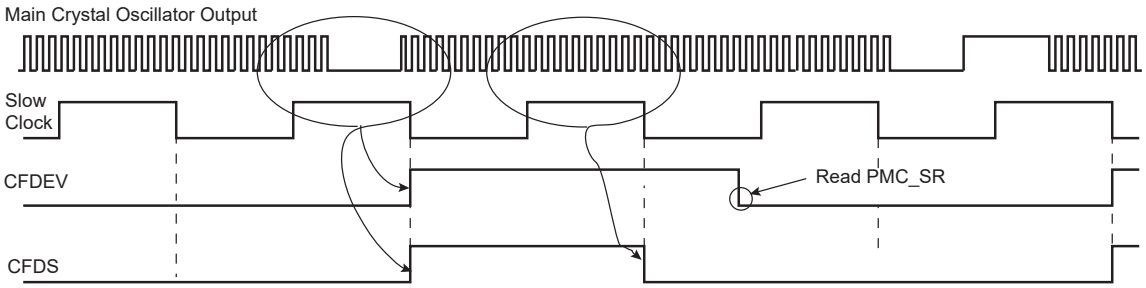
A failure is detected by means of a counter incrementing on the Main crystal oscillator output and detection logic is triggered by the Slow RC oscillator which is automatically enabled when CFDEN = 1.

The counter is cleared when the Slow RC oscillator clock signal is low and enabled when the signal is high. Thus, the failure detection time is one Slow RC oscillator period. If, during the high level period of the Slow RC oscillator clock signal, less than eight Main crystal oscillator clock periods have been counted, then a failure is reported. Note that when enabling the failure detector, up to two cycles of the Slow RC oscillator are needed to detect a failure of the Main crystal oscillator.

If a failure of Main crystal oscillator is detected, PMC_SR.CFDEV and PMC_SR.FOS both indicate a failure event. PMC_SR.CFDEV is cleared on read of PMC_SR, and PMC_SR.FOS is cleared by writing a ‘1’ to the FOCLR bit in the PMC Fault Output Clear Register (PMC_FOCR).

Only PMC_SR.CFDEV can generate an interrupt if the corresponding interrupt source is enabled in PMC_IER. The current status of the clock failure detection can be read at any time from PMC_SR.CFDS.

**Figure 31-5. Clock Failure Detection Example**




Note: Ratio of clock periods is for illustration purposes only.

If the Main crystal oscillator is selected as the source clock of MAINCK (CKGR_MOR.MOSCSEL = 1), and if the MCK source is PLLACK or UPLLCKDIV (CSS = 2 or 3), a clock failure detection automatically forces MAINCK to be the source clock for MCK. Then, regardless of the PMC configuration, a clock failure detection automatically forces the Main RC oscillator to be the source clock for MAINCK. If the Main RC oscillator is disabled when a clock failure detection occurs, it is automatically re-enabled by the clock failure detection mechanism.

Two Slow RC oscillator clock cycles are necessary to detect and switch from the Main crystal oscillator to the Main RC oscillator if the source of MCK is MAINCK, or three Slow RC oscillator clock cycles if the source of MCK is PLLACK or UPLLCKDIV.

A clock failure detection activates a fault output that is connected to the Pulse Width Modulator (PWM) Controller. With this connection, the PWM controller is able to force its outputs and to protect the driven device, if a clock failure is detected.

## 31.16 32.768 kHz Crystal Oscillator Frequency Monitor

The frequency of the 32.768 kHz crystal oscillator can be monitored by means of logic driven by the Main RC oscillator known as a reliable clock source. This function is enabled by configuring the XT32KFME bit of CKGR_MOR. Prior to enabling this frequency monitor, the 32.768 kHz crystal oscillator must be started and its startup time be elapsed. Refer to details on the Slow clock generator in the section “Supply Controller (SUPC)”.

An error flag (XT32KERR in PMC_SR) is asserted when the 32.768 kHz crystal oscillator frequency is out of the ±10% nominal frequency value (i.e., 32.768 kHz). The error flag can be cleared only if the frequency monitor is disabled.

When the Main RC oscillator frequency is set to 4 MHz, the accuracy of the measurement is ±40% as this frequency is not trimmed during production. Therefore, ±10% accuracy is obtained only if the Main RC oscillator frequency is configured for 8 or 12 MHz.

The monitored clock frequency is declared invalid if at least 4 consecutive clock period measurement results are over the nominal period ±10%. Note that modifying the trimming values of the Main RC oscillator (PMC_OCR) may impact the monitor accuracy and lead to inappropriate failure detection.

Due to the possible frequency variation of the Main RC oscillator acting as reference clock for the monitor logic, any 32.768 kHz crystal frequency deviation over ±10% of the nominal frequency is systematically reported as an error by means of PMC_SR.XT32KERR. Between -1% and -10% and +1% and +10%, the error is not systematically reported.

Thus only a crystal running at 32.768 kHz frequency ensures that the error flag will not be asserted. The permitted drift of the crystal is 10000 ppm (1%), which allows any standard crystal to be used.

If the Main RC oscillator frequency range needs to be changed while the frequency monitor is operating, the monitoring must be stopped prior to change the Main RC oscillator frequency. Then it can be re-enabled as soon as PMC_SR.MOSCRCS is set.

The error flag can be defined as an interrupt source of the PMC by setting PMC_IER.XT32KERR. This flag is also routed to the RSTC and may generate a reset of the device.

**Related Links**

23. Supply Controller (SUPC)

## 31.17 Recommended Programming Sequence

Follow the steps below to program the PMC:

1. If the Main crystal oscillator is not required, the PLL and divider can be directly configured (Step 6.) else this oscillator must be started (Step 2.).
2. Enable the Main crystal oscillator by setting CKGR_MOR.MOSCXTEN. The user can define a startup time. This can be done by configuring the appropriate value in CKGR_MOR.MOSCXTST. Once this register has been correctly configured, the user must wait for PMC_SR.MOSCXTS to be set. This can be done either by polling PMC_SR.MOSCXTS, or by waiting for the interrupt line to be raised if the associated interrupt source (MOSCXTS) has been enabled in PMC_IER.
3. Switch MAINCK to the Main crystal oscillator by setting CKGR_MOR.MOSCSEL.
4. Wait for PMC_SR.MOSCSELS to be set to ensure the switch is complete.
5. Check MAINCK frequency:
   This frequency can be measured via CKGR_MCFR.

   Read CKGR_MCFR until the MAINFRDY field is set, after which the user can read CKGR_MCFR.MAINF by performing an additional read. This provides the number of Main clock cycles that have been counted during a period of 16 SLCK cycles.

   If MAINF = 0, switch MAINCK to the Main RC Oscillator by clearing CKGR_MOR.MOSCSEL. If MAINF ≠ 0, proceed to Step 6.
6. Set PLLA and Divider (if not required, proceed to Step 7.):
   All parameters needed to configure PLLA and the divider are located in CKGR_PLLAR.

   CKGR_PLLAR.DIVA is used to control the divider. This parameter can be programmed between 0 and 127. Divider output is divider input divided by DIVA parameter. By default, DIVA field is cleared which means that the divider and PLLA are turned off.

   CKGR_PLLAR.MULA is the PLLA multiplier factor. This parameter can be programmed between 0 and 62. If MULA is cleared, PLLA will be turned off, otherwise the PLLA output frequency is PLLA input frequency multiplied by (MULA + 1).

   CKGR_PLLAR.PLLACOUNT specifies the number of SLCK cycles before PMC_SR.LOCKA is set after CKGR_PLLAR has been written.

   Once CKGR_PLLAR has been written, the user must wait for PMC_SR.LOCKA to be set. This can be done either by polling PMC_SR.LOCKA or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKA) has been enabled in PMC_IER. All fields in CKGR_PLLAR can be programmed in a single write operation. If MULA or DIVA is modified, the LOCKA bit goes low to indicate that PLLA is not yet ready. When PLLA is locked, LOCKA is set again. The user must wait for the LOCKA bit to be set before using the PLLA output clock.

© 2021 Microchip Technology Inc. and its subsidiaries

7. Select MCK and HCLK:
   MCK and HCLK are configurable via PMC_MCKR.

   CSS is used to select the clock source of MCK and HCLK. By default, the selected clock source is MAINCK.

   PRES is used to define the HCLK and MCK prescaler.s The user can choose between different values (1, 2, 3, 4, 8, 16, 32, 64). Prescaler output is the selected clock source frequency divided by the PRES value.

   MDIV is used to define the MCK divider. It is possible to choose between different values (0, 1, 2, 3). MCK output is the HCLK frequency divided by 1, 2, 3 or 4, depending on the value programmed in MDIV.

   By default, MDIV is cleared, which indicates that the HCLK is equal to MCK.

   Once the PMC_MCKR has been written, the user must wait for PMC_SR.MCKRDY to be set. This can be done either by polling PMC_SR.MCKRDY or by waiting for the interrupt line to be raised if the associated interrupt source (MCKRDY) has been enabled in PMC_IER. PMC_MCKR must not be programmed in a single write operation. The programming sequence for PMC_MCKR is as follows:

   If a new value for PMC_MCKR.CSS corresponds to any of the available PLL clocks:

   a. Program PMC_MCKR.PRES.

   b. Wait for PMC_SR.MCKRDY to be set.

   c. Program PMC_MCKR.MDIV.

   d. Wait for PMC_SR.MCKRDY to be set.

   e. Program PMC_MCKR.CSS.

   f. Wait for PMC_SR.MCKRDY to be set.

   If a new value for PMC_MCKR.CSS corresponds to MAINCK or SLCK:

   a. Program PMC_MCKR.CSS.

   b. Wait for PMC_SR.MCKRDY to be set.

   c. Program PMC_MCKR.PRES.

   d. Wait for PMC_SR.MCKRDY to be set.

   If CSS, MDIV or PRES are modified at any stage, the MCKRDY bit goes low to indicate that MCK and HCLK are not yet ready. The user must wait for MCKRDY bit to be set again before using MCK and HCLK.

   **Note:** If PLLA clock was selected as MCK and the user decides to modify it by writing a new value into CKGR_PLLAR, the MCKRDY flag will go low while PLLA is unlocked. Once PLLA is locked again, LOCKA goes high and MCKRDY is set.
   While PLLA is unlocked, MCK selection is automatically changed to SLCK for PLLA. For further information, see "Clock Switching Waveforms".

   MCK is MAINCK divided by 2.
8. Select the Programmable clocks (PCKx):
   PCKx are controlled via registers PMC_SCER, PMC_SCDR and PMC_SCSR.

   PCKx can be enabled and/or disabled via PMC_SCER and PMC_SCDR. Three PCKx can be used. PMC_SCSR indicates which PCKx is enabled. By default all PCKx are disabled.

   PMC_PCKx registers are used to configure PCKx.

   PMC_PCKx.CSS is used to select the PCKx divider source. Several clock options are available:

   – MAINCK
   – SLCK
   – MCK
   – PLLACK
   – UPLLCKDIV
     SLCK is the default clock source.

PMC_PCKx.PRES is used to control the PCKx prescaler. It is possible to choose between different values (1 to 256). PCKx output is prescaler input divided by PRES. By default, the PRES value is cleared which means that PCKx is equal to Slow clock.

Once PMC_PCKx has been configured, the corresponding PCKx must be enabled and the user must wait for PMC_SR.PCKRDYx to be set. This can be done either by polling PMC_SR.PCKRDYx or by waiting for the interrupt line to be raised if the associated interrupt source (PCKRDYx) has been enabled in PMC_IER. All parameters in PMC_PCKx can be programmed in a single write operation.

If the PMC_PCKx.CSS and PMC_PCKx.PRES parameters are to be modified, the corresponding PCKx must be disabled first. The parameters can then be modified. Once this has been done, the user must re-enable PCKx and wait for the PCKRDYx bit to be set.

9. Enable the peripheral clocks
   Once all of the previous steps have been completed, the peripheral clocks can be enabled and/or disabled via registers PMC_PCERx and PMC_PCDRx.

## 31.18 Clock Switching Details

### 31.18.1 Master Clock Switching Timings

The following two tables, Clock Switching Timings (Worst Case) and Clock Switching Timings Between Two PLLs (Worst Case) give the worst case timings required for MCK to switch from one selected clock to another one. This is in the event that the prescaler is deactivated. When the prescaler is activated, an additional time of 64 clock cycles of the newly selected clock has to be added.

**Table 31-2. Clock Switching Timings (Worst Case)**

| From | MAINCK | SLCK | PLL Clock |
|---|---|---|---|
| To | | | |
| MAINCK | – | 4 x SLCK +<br>2.5 x MAINCK | 3 x PLL Clock +<br>4 x SLCK +<br>1 x MAINCK |
| SLCK | 0.5 x MAINCK +<br>4.5 x SLCK | – | 3 x PLL Clock +<br>5 x SLCK |
| PLL Clock | 0.5 x MAINCK +<br>4 x SLCK +<br>PLLCOUNT x SLCK +<br>2.5 x PLL Clock | 2.5 x PLL Clock +<br>5 x SLCK +<br>PLLCOUNT x SLCK | See the following table. |

**Notes:**

1. PLL designates any available PLL of the Clock Generator.
2. PLLCOUNT designates either PLLACOUNT or UPLLCOUNT.

**Table 31-3. Clock Switching Timings Between Two PLLs (Worst Case)**

| | From | PLLACK | UPLL Clock |
|---|---|---|---|
| To | | | |
| PLLACK | | – | 3 x PLLACK +<br>4 x SLCK +<br>1.5 x PLLACK |
| UPLLCKDIV | | 3 x UPLLCKDIV +<br>4 x SLCK +<br>1.5 x UPLLCKDIV | – |

### 31.18.2 Clock Switching Waveforms

**Figure 31-6. Switch Master Clock (MCK) from Slow Clock to PLLx Clock**

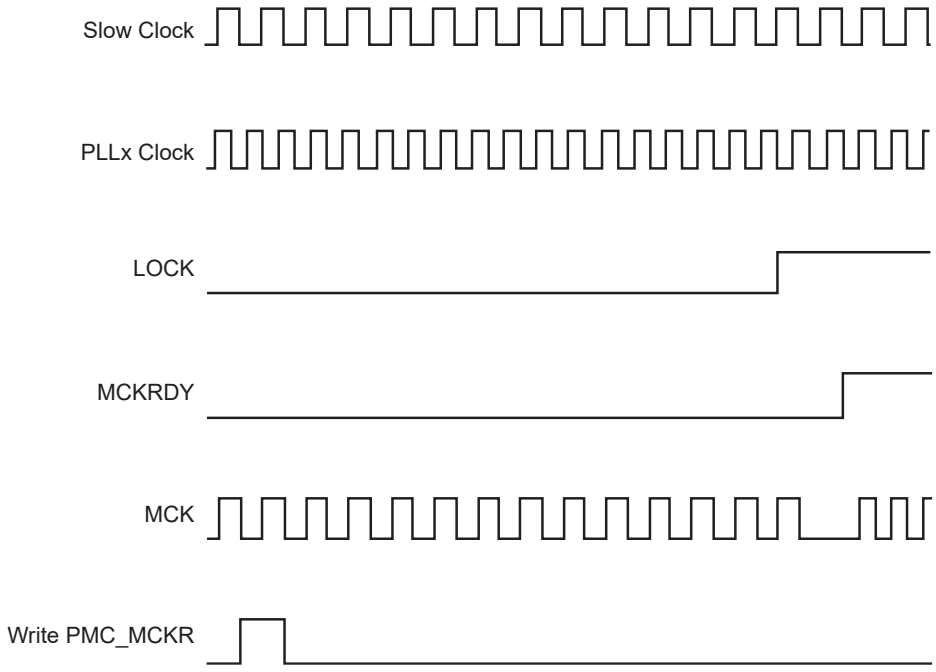



**Figure 31-7. Switch Master Clock (MCK) from Main Clock (MAINCK) to Slow Clock**

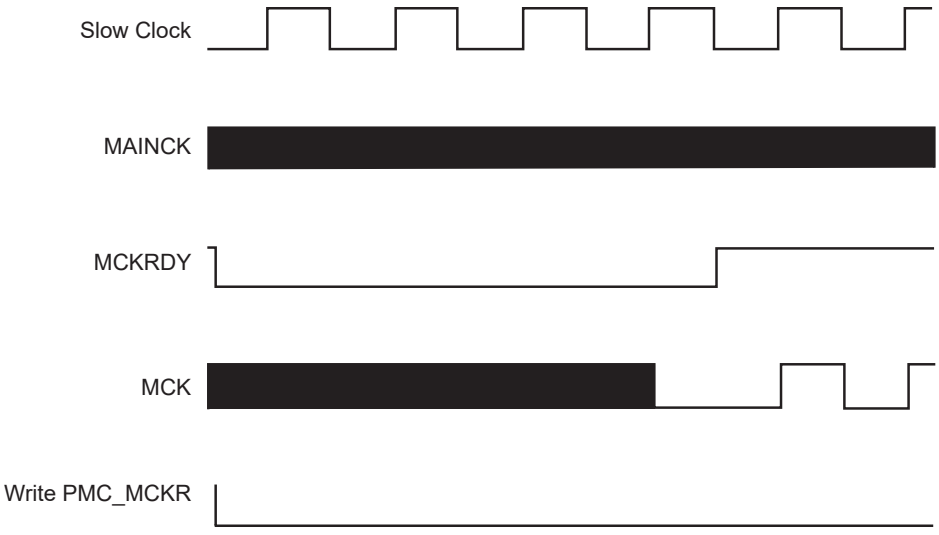


**Figure 31-8. Change PLLA Programming**

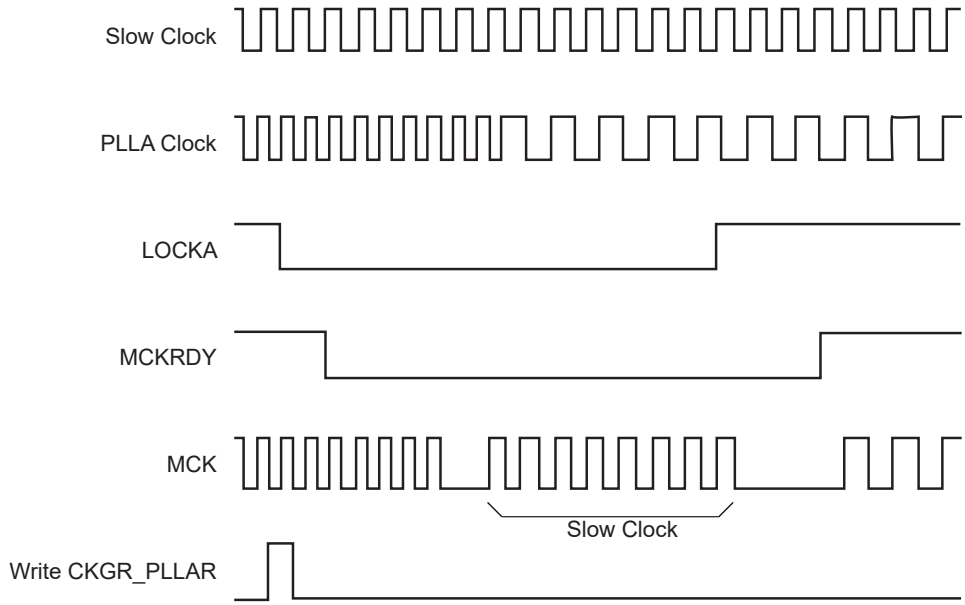
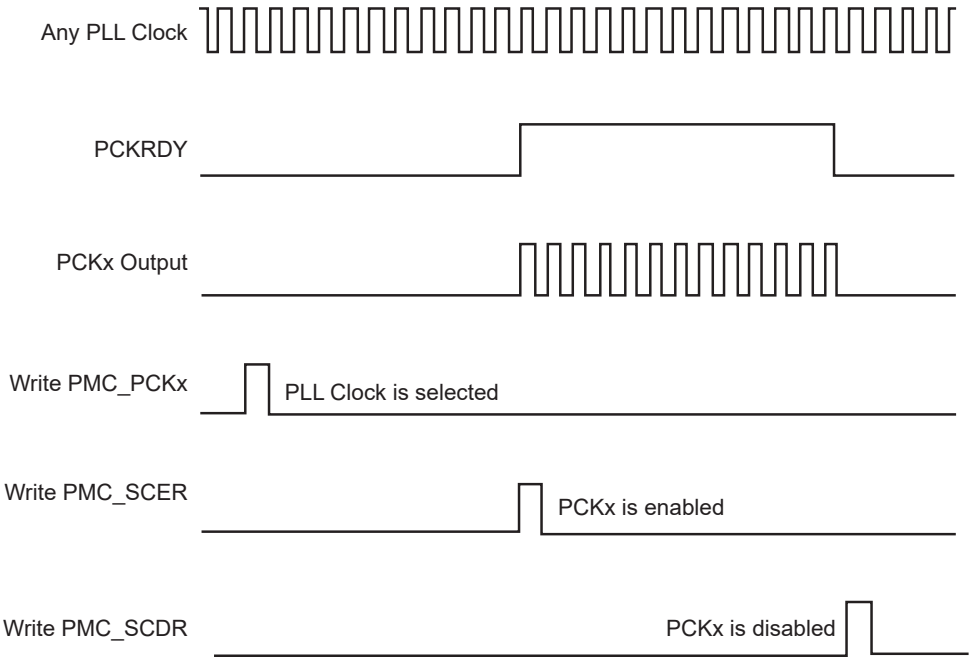



**Figure 31-9. Programmable Clock Output Programming**




## 31.19 Register Write Protection

To prevent any single software error from corrupting PMC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the PMC Write Protection Mode Register (PMC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the PMC Write Protection Status Register (PMC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the PMC_WPSR.

The following registers are write-protected when the WPEN bit is set in PMC_WPMR:

- PMC System Clock Disable Register

- PMC Peripheral Clock Enable Register 0
- PMC Peripheral Clock Disable Register 0
- PMC Clock Generator Main Oscillator Register
- PMC Clock Generator Main Clock Frequency Register
- PMC Clock Generator PLLA Register
- PMC UTMI Clock Configuration Register
- PMC Master Clock Register
- PMC USB Clock Register
- PMC Programmable Clock Register
- PMC Fast Startup Mode Register
- PMC Fast Startup Polarity Register
- PMC Peripheral Clock Enable Register 1
- PMC Peripheral Clock Disable Register 1
- PMC Oscillator Calibration Register
- PMC SleepWalking Enable Register 0
- PMC SleepWalking Disable Register 0
- PLL Maximum Multiplier Value Register
- PMC SleepWalking Enable Register 1
- PMC SleepWalking Disable Register 1

© 2021 Microchip Technology Inc.
and its subsidiaries

## 31.20 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | PMC_SCER | 7:0 | | | USBCLK | | | | | |
| | | 15:8 | PCK7 | PCK6 | PCK5 | PCK4 | PCK3 | PCK2 | PCK1 | PCK0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | PMC_SCDR | 7:0 | | | USBCLK | | | | | |
| | | 15:8 | PCK7 | PCK6 | PCK5 | PCK4 | PCK3 | PCK2 | PCK1 | PCK0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | PMC_SCSR | 7:0 | | | USBCLK | | | | | HCLKS |
| | | 15:8 | PCK7 | PCK6 | PCK5 | PCK4 | PCK3 | PCK2 | PCK1 | PCK0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | PMC_PCER0 | 7:0 | PID7 | | | | | | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x14 | PMC_PCDR0 | 7:0 | PID7 | | | | | | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x18 | PMC_PCSR0 | 7:0 | PID7 | | | | | | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x1C | CKGR_UCKR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | UPLLCOUNT[3:0] | | | | | | | UPLLEN |
| | | 31:24 | | | | | | | | |
| 0x20 | CKGR_MOR | 7:0 | | MOSCRCF[2:0] | | | MOSCRCEN | WAITMODE | MOSCXTBY | MOSCXTEN |
| | | 15:8 | MOSCXTST[7:0] | | | | | | | |
| | | 23:16 | KEY[7:0] | | | | | | | |
| | | 31:24 | | | | | | XT32KFME | CFDEN | MOSCSEL |
| 0x24 | CKGR_MCFR | 7:0 | MAINF[7:0] | | | | | | | |
| | | 15:8 | MAINF[15:8] | | | | | | | |
| | | 23:16 | | | | RCMEAS | | | | MAINFRDY |
| | | 31:24 | | | | | | | | CCSS |
| 0x28 | CKGR_PLLAR | 7:0 | DIVA[7:0] | | | | | | | |
| | | 15:8 | | | PLLACOUNT[5:0] | | | | | |
| | | 23:16 | MULA[7:0] | | | | | | | |
| | | 31:24 | | | ONE | | | MULA[10:8] | | |
| 0x2C ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | PMC_MCKR | 7:0 | | PRES[2:0] | | | | | CSS[1:0] | |
| | | 15:8 | | | UPLLDIV2 | | | | MDIV[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 ... 0x37 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x38 | PMC_USB | 7:0 | | | | | | | | USBS |
| | | 15:8 | | | | | USBDIV[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | PMC_PCKx0 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | PMC_PCKx1 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | PMC_PCKx2 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | PMC_PCKx3 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x50 | PMC_PCKx4 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x54 | PMC_PCKx5 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x58 | PMC_PCKx6 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x5C | PMC_PCKx7 | 7:0 | PRES[3:0] | | | | | CSS[2:0] | | |
| | | 15:8 | | | | | PRES[7:4] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x60 | PMC_IER | 7:0 | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| | | 15:8 | PCKRDY7 | PCKRDY6 | PCKRDY5 | PCKRDY4 | PCKRDY3 | PCKRDY2 | PCKRDY1 | PCKRDY0 |
| | | 23:16 | | | XT32KERR | | | CFDEV | MOSCRCS | MOSCSELS |
| | | 31:24 | | | | | | | | |
| 0x64 | PMC_IDR | 7:0 | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| | | 15:8 | PCKRDY7 | PCKRDY6 | PCKRDY5 | PCKRDY4 | PCKRDY3 | PCKRDY2 | PCKRDY1 | PCKRDY0 |
| | | 23:16 | | | XT32KERR | | | CFDEV | MOSCRCS | MOSCSELS |
| | | 31:24 | | | | | | | | |
| 0x68 | PMC_SR | 7:0 | OSCSELS | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| | | 15:8 | PCKRDY7 | PCKRDY6 | PCKRDY5 | PCKRDY4 | PCKRDY3 | PCKRDY2 | PCKRDY1 | PCKRDY0 |
| | | 23:16 | | | XT32KERR | FOS | CFDS | CFDEV | MOSCRCS | MOSCSELS |
| | | 31:24 | | | | | | | | |
| 0x6C | PMC_IMR | 7:0 | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| | | 15:8 | PCKRDY7 | PCKRDY6 | PCKRDY5 | PCKRDY4 | PCKRDY3 | PCKRDY2 | PCKRDY1 | PCKRDY0 |
| | | 23:16 | | | XT32KERR | | | CFDEV | MOSCRCS | MOSCSELS |
| | | 31:24 | | | | | | | | |
| 0x70 | PMC_FSMR | 7:0 | FSTT7 | FSTT6 | FSTT5 | FSTT4 | FSTT3 | FSTT2 | FSTT1 | FSTT0 |
| | | 15:8 | FSTT15 | FSTT14 | FSTT13 | FSTT12 | FSTT11 | FSTT10 | FSTT9 | FSTT8 |
| | | 23:16 | FFLPM | FLPM[1:0] | | LPM | | USBAL | RTCAL | RTTAL |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x74 | PMC_FSPR | 7:0 | FSTP7 | FSTP6 | FSTP5 | FSTP4 | FSTP3 | FSTP2 | FSTP1 | FSTP0 |
| | | 15:8 | FSTP15 | FSTP14 | FSTP13 | FSTP12 | FSTP11 | FSTP10 | FSTP9 | FSTP8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | PMC_FOCR | 7:0 | | | | | | | | FOCLR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x7C ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | PMC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | PMC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xEC ... 0xFF | Reserved | | | | | | | | | |
| 0x0100 | PMC_PCER1 | 7:0 | PID[0] | | PID | | PID[3:0] | | | |
| | | 15:8 | PID[8:1] | | | | | | | |
| | | 23:16 | PID[16:9] | | | | | | | |
| | | 31:24 | | PID | | PID[21:17] | | | | |
| 0x0104 | PMC_PCDR1 | 7:0 | PID[0] | | PID | | PID[3:0] | | | |
| | | 15:8 | PID[8:1] | | | | | | | |
| | | 23:16 | PID[16:9] | | | | | | | |
| | | 31:24 | | PID | | PID[21:17] | | | | |
| 0x0108 | PMC_PCSR1 | 7:0 | PID0 | | | | | | | |
| | | 15:8 | PID8 | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | PID1 |
| | | 23:16 | PID16 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 |
| | | 31:24 | PID24 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 |
| 0x010C | PMC_PCR | 7:0 | | PID[6:0] | | | | | | |
| | | 15:8 | | | | CMD | | GCLKCSS[2:0] | | |
| | | 23:16 | GCLKDIV[3:0] | | | | | | | |
| | | 31:24 | | | GCLKEN | EN | GCLKDIV[7:4] | | | |
| 0x0110 | PMC_OCR | 7:0 | SEL4 | CAL4[6:0] | | | | | | |
| | | 15:8 | SEL8 | CAL8[6:0] | | | | | | |
| | | 23:16 | SEL12 | CAL12[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0114 | PMC_SLPWK_ER0 | 7:0 | PID7 | | | | | | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x0118 | PMC_SLPWK_DR0 | 7:0 | PID7 | | | | | | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x011C | PMC_SLPWK_SR0 | 7:0 | PID7 | | | | | | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| 0x0120 | PMC_SLPWK_ASR0 | 7:0 | PID7 | | | | | | | |
| | | 15:8 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| | | 23:16 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| | | 31:24 | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0124 ... 0x012F | Reserved | | | | | | | | | |
| 0x0130 | PMC_PMMR | 7:0 | PLLA_MMAX[7:0] | | | | | | | |
| | | 15:8 | | | | | | PLLA_MMAX[10:8] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0134 | PMC_SLPWK_ER1 | 7:0 | PID[0] | | PID | | PID[3:0] | | | |
| | | 15:8 | PID[8:1] | | | | | | | |
| | | 23:16 | PID[16:9] | | | | | | | |
| | | 31:24 | | PID | | PID[21:17] | | | | |
| 0x0138 | PMC_SLPWK_DR1 | 7:0 | PID[0] | | PID | | PID[3:0] | | | |
| | | 15:8 | PID[8:1] | | | | | | | |
| | | 23:16 | PID[16:9] | | | | | | | |
| | | 31:24 | | PID | | PID[21:17] | | | | |
| 0x013C | PMC_SLPWK_SR1 | 7:0 | PID[0] | | PID | | PID[3:0] | | | |
| | | 15:8 | PID[8:1] | | | | | | | |
| | | 23:16 | PID[16:9] | | | | | | | |
| | | 31:24 | | PID | | PID[21:17] | | | | |
| 0x0140 | PMC_SLPWK_ASR1 | 7:0 | PID[0] | | PID | | PID[3:0] | | | |
| | | 15:8 | PID[8:1] | | | | | | | |
| | | 23:16 | PID[16:9] | | | | | | | |
| | | 31:24 | | PID | | PID[21:17] | | | | |
| 0x0144 | PMC_SLPWK_AIPR | 7:0 | | | | | | | | AIP |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 31.20.1 PMC System Clock Enable Register

**Name:** PMC_SCER
**Offset:** 0x0000
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PCK7 | PCK6 | PCK5 | PCK4 | PCK3 | PCK2 | PCK1 | PCK0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | USBCLK | | | | | |
| Access | | | W | | | | | |
| Reset | | | – | | | | | |

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – PCK** Programmable Clock x Output Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding Programmable Clock output. |

**Bit 5 – USBCLK** Enable USB FS Clock

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables USB FS clock. |

### 31.20.2 PMC System Clock Disable Register

**Name:** PMC_SCDR
**Offset:** 0x0004
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PCK7 | PCK6 | PCK5 | PCK4 | PCK3 | PCK2 | PCK1 | PCK0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | USBCLK | | | | | |
| Access | | | W | | | | | |
| Reset | | | – | | | | | |

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – PCK** Programmable Clock x Output Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding Programmable Clock output. |

**Bit 5 – USBCLK** Disable USB FS Clock

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables USB FS clock. |

### 31.20.3 PMC System Clock Status Register

**Name:** PMC_SCSR
**Offset:** 0x0008
**Reset:** 0x00000001
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PCK7 | PCK6 | PCK5 | PCK4 | PCK3 | PCK2 | PCK1 | PCK0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | USBCLK | | | | | HCLKS |
| Access | | | R | | | | | R |
| Reset | | | 0 | | | | | 1 |

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – PCK** Programmable Clock x Output Status

| Value | Description |
|---|---|
| 0 | The corresponding Programmable Clock output is disabled. |
| 1 | The corresponding Programmable Clock output is enabled. |

**Bit 5 – USBCLK** USB FS Clock Status

| Value | Description |
|---|---|
| 0 | The USB FS clock is disabled. |
| 1 | The USB FS clock is enabled. |

**Bit 0 – HCLKS** HCLK Status

| Value | Description |
|---|---|
| 0 | HCLK is disabled. |
| 1 | HCLK is enabled. |

### 31.20.4 PMC Peripheral Clock Enable Register 0

**Name:** PMC_PCER0
**Offset:** 0x0010
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

PIDx refers to identifiers defined in the section “Peripheral Identifiers”. Other peripherals can be enabled in PMC_PCER1 (see "PMC Peripheral Clock Enable Register 1").

Programming the control bits of the Peripheral ID that are not implemented has no effect on the behavior of the PMC.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | | | | | | | |
| Access | W | | | | | | | |
| Reset | – | | | | | | | |

**Bits 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding peripheral clock. |

### 31.20.5 PMC Peripheral Clock Disable Register 0

**Name:** PMC_PCDR0
**Offset:** 0x0014
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | | | | | | | |
| Access | W | | | | | | | |
| Reset | – | | | | | | | |

**Bits 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Disable
PIDx refers to identifiers defined in the section “Peripheral Identifiers”. Other peripherals can be disabled in PMC_PCDR1 (see "PMC Peripheral Clock Disable Register 1").

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding peripheral clock. |

### 31.20.6 PMC Peripheral Clock Status Register 0

**Name:** PMC_PCSR0
**Offset:** 0x0018
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

**Bits 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Status
PIDx refers to identifiers defined in the section “Peripheral Identifiers”. Other peripherals status can be read in PMC_PCSR1 (see "PMC Peripheral Clock Status Register 1").

| Value | Description |
|---|---|
| 0 | The corresponding peripheral clock is disabled. |
| 1 | The corresponding peripheral clock is enabled. |

### 31.20.7 PMC UTMI Clock Configuration Register

**Name:** CKGR_UCKR
**Offset:** 0x001C
**Reset:** 0x10200800
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UPLLCOUNT[3:0] | | | | | | | UPLLEN |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 1 | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 23:20 – UPLLCOUNT[3:0]** UTMI PLL Startup Time
Specifies the number of SLCK cycles multiplied by 8 for the UTMI PLL startup time.

**Bit 16 – UPLLEN** UTMI PLL Enable
When UPLLEN is set, the LOCKU flag is set once the UTMI PLL startup time is achieved.

| Value | Description |
|---|---|
| 0 | The UTMI PLL is disabled. |
| 1 | The UTMI PLL is enabled. |

### 31.20.8 PMC Clock Generator Main Oscillator Register

**Name:** CKGR_MOR
**Offset:** 0x0020
**Reset:** 0x00000008
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | XT32KFME | CFDEN | MOSCSEL |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MOSCXTST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | MOSCRCF[2:0] | | | MOSCRCEN | WAITMODE | MOSCXTBY | MOSCXTEN |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

**Bit 26 – XT32KFME** 32.768 kHz Crystal Oscillator Frequency Monitoring Enable

| Value | Description |
|---|---|
| 0 | The 32.768 kHz crystal oscillator frequency monitoring is disabled. |
| 1 | The 32.768 kHz crystal oscillator frequency monitoring is enabled. |

**Bit 25 – CFDEN** Clock Failure Detector Enable

| Value | Description |
|---|---|
| 0 | The clock failure detector is disabled. |
| 1 | The clock failure detector is enabled. |

**Bit 24 – MOSCSEL** Main Clock Oscillator Selection

| Value | Description |
|---|---|
| 0 | The Main RC oscillator is selected. |
| 1 | The Main crystal oscillator is selected. |

**Bits 23:16 – KEY[7:0]** Write Access Password

| Value | Name | Description |
|---|---|---|
| 0x37 | PASSWD | Writing any other value in this field aborts the write operation.<br>Always reads as 0. |

**Bits 15:8 – MOSCXTST[7:0]** Main Crystal Oscillator Startup Time
Specifies the number of SLCK cycles multiplied by 8 for the main crystal oscillator startup time.

**Bits 6:4 – MOSCRCF[2:0]** Main RC Oscillator Frequency Selection
At startup, the Main RC oscillator frequency is 12 MHz.
MOSCRCF must be changed only if MOSCRCS is set in the PMC_SR. Therefore, MOSCRCF and MOSCRCEN cannot be changed at the same time.

© 2021 Microchip Technology Inc. and its subsidiaries

| Value | Name | Description |
|---|---|---|
| 0 | 4_MHz | The RC oscillator frequency is at 4 MHz |
| 1 | 8_MHz | The RC oscillator frequency is at 8 MHz |
| 2 | 12_MHz | The RC oscillator frequency is at 12 MHz |

**Bit 3 – MOSCRCEN** Main RC Oscillator Enable
When MOSCRCEN is set, the MOSCRCS flag is set once the Main RC oscillator startup time is achieved.

| Value | Description |
|---|---|
| 0 | The Main RC oscillator is disabled. |
| 1 | The Main RC oscillator is enabled. |

**Bit 2 – WAITMODE** Wait Mode Command (write-only)

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Puts the device in Wait mode. |

**Bit 1 – MOSCXTBY** Main Crystal Oscillator Bypass
When MOSCXTBY is set, the MOSCXTS flag in PMC_SR is automatically set.
Clearing MOSCXTEN and MOSCXTBY bits clears the MOSCXTS flag.
When the crystal oscillator bypass is disabled (MOSCXTBY = 0), the MOSCXTS flag must be read at ‘0’ in PMC_SR before enabling the crystal oscillator (MOSCXTEN = 1).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Main crystal oscillator is bypassed. MOSCXTEN must be cleared. An external clock must be connected on XIN. |

**Bit 0 – MOSCXTEN** Main Crystal Oscillator Enable
A crystal must be connected between XIN and XOUT.
When MOSCXTEN is set, the MOSCXTS flag is set once the Main crystal oscillator startup time is achieved.

| Value | Description |
|---|---|
| 0 | The Main crystal oscillator is disabled. |
| 1 | The Main crystal oscillator is enabled. MOSCXTBY must be cleared. |

### 31.20.9 PMC Clock Generator Main Clock Frequency Register

**Name:** CKGR_MCFR
**Offset:** 0x0024
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CCSS |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | RCMEAS | | | | MAINFRDY |
| Access | | | | R/W | | | | R/W |
| Reset | | | | 0 | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MAINF[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MAINF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – CCSS** Counter Clock Source Selection

| Value | Description |
|---|---|
| 0 | The measured clock of the MAINF counter is the Main RC oscillator. |
| 1 | The measured clock of the MAINF counter is the Main crystal oscillator. |

**Bit 20 – RCMEAS** RC Oscillator Frequency Measure (write-only)
The measurement is performed on the main frequency (i.e., not limited to the Main RC oscillator only). If the source of MAINCK is the Main crystal oscillator, the restart of measurement may not be required because of the stability of crystal oscillators.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Restarts measuring of the frequency of MAINCK. MAINF carries the new frequency as soon as a low-to-high transition occurs on the MAINFRDY flag. |

**Bit 16 – MAINFRDY** Main Clock Frequency Measure Ready
To ensure that a correct value is read on the MAINF field, the MAINFRDY flag must be read at '1' then another read access must be performed on the register to get a stable value on the MAINF field.

| Value | Description |
|---|---|
| 0 | MAINF value is not valid or the measured oscillator is disabled or a measure has just been started by means of RCMEAS. |
| 1 | The measured oscillator has been enabled previously and MAINF value is available. |

**Bits 15:0 – MAINF[15:0]** Main Clock Frequency
Gives the number of cycles of the clock selected by the bit CCSS within 16 SLCK periods. To calculate the frequency of the measured clock:
$f_{SELCLK} = (MAINF \times f_{SLCK})/16$
where frequency is in MHz.

### 31.20.10 PMC Clock Generator PLLA Register

**Name:** CKGR_PLLAR
**Offset:** 0x0028
**Reset:** 0x00003F00
**Property:** Read/Write

Possible limitations on PLLA input frequencies and multiplier factors should be checked before using the PMC.

**WARNING** Bit 29 must always be set to ‘1’ when programming the CKGR_PLLAR.

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ONE | | | MULA[10:8] | | |
| Access | | | R/W | | | R/W | R/W | R/W |
| Reset | | | 0 | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MULA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PLLACOUNT[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIVA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – ONE** Must Be Set to 1
Bit 29 must always be set to ‘1’ when programming the CKGR_PLLAR.

**Bits 26:16 – MULA[10:0]** PLLA Multiplier
1 up to 62 = PLLCK frequency is the PLLA input frequency multiplied by MULA + 1.
Unlisted values are forbidden.

| Value | Description |
|---|---|
| 0 | The PLLA is disabled (PLLA also disabled if DIVA = 0). |

**Bits 13:8 – PLLACOUNT[5:0]** PLLA Counter
Specifies the number of SLCK cycles before the LOCKA bit is set in PMC_SR after CKGR_PLLAR is written.

**Bits 7:0 – DIVA[7:0]** PLLA Front End Divider

| Value | Name | Description |
|---|---|---|
| 0 | 0 | PLLA is disabled. |
| 1 | BYPASS | Divider is bypassed (divide by 1) and PLLA is enabled. |
| 2-255 | – | Divider output is the selected clock divided by DIVA. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 31.20.11 PMC Master Clock Register

**Name:** PMC_MCKR
**Offset:** 0x0030
**Reset:** 0x00000001
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | UPLLDIV2 | | | | MDIV[1:0] | |
| Access | | | R/W | | | | R/W | R/W |
| Reset | | | 0 | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PRES[2:0] | | | | | CSS[1:0] | |
| Access | | R/W | R/W | R/W | | | R/W | R/W |
| Reset | | 0 | 0 | 0 | | | 0 | 1 |

**Bit 13 – UPLLDIV2** UPLL Divider by 2

| Value | Description |
|---|---|
| 0 | UPLLCK frequency is divided by 1. |
| 1 | UPLLCK frequency is divided by 2. |

**Bits 9:8 – MDIV[1:0]** Master Clock Division

| Value | Name | Description |
|---|---|---|
| 0 | EQ_PCK | MCK is FCLK divided by 1. |
| 1 | PCK_DIV2 | MCK is FCLK divided by 2. |
| 2 | PCK_DIV4 | MCK is FCLK divided by 4. |
| 3 | PCK_DIV3 | MCK is FCLK divided by 3. |

**Bits 6:4 – PRES[2:0]** Processor Clock Prescaler

| Value | Name | Description |
|---|---|---|
| 0 | CLK_1 | Selected clock |
| 1 | CLK_2 | Selected clock divided by 2 |
| 2 | CLK_4 | Selected clock divided by 4 |
| 3 | CLK_8 | Selected clock divided by 8 |
| 4 | CLK_16 | Selected clock divided by 16 |
| 5 | CLK_32 | Selected clock divided by 32 |
| 6 | CLK_64 | Selected clock divided by 64 |
| 7 | CLK_3 | Selected clock divided by 3 |

**Bits 1:0 – CSS[1:0]** Master Clock Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | SLOW_CLK | SLCK is selected |
| 1 | MAIN_CLK | MAINCK is selected |

| Value | Name | Description |
|---|---|---|
| 2 | PLLA_CLK | PLLACK is selected |
| 3 | UPLL_CLK | UPPLLCKDIV is selected |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 31.20.12 PMC USB Clock Register

**Name:** PMC_USB
**Offset:** 0x0038
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | USBDIV[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | USBS |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 11:8 – USBDIV[3:0]** Divider for USB_48M
USB_48M is input clock divided by USBDIV+1.

**Bit 0 – USBS** USB Input Clock Selection

| Value | Description |
|---|---|
| 0 | USB_48M input is PLLA. |
| 1 | USB_48M input is UPLL. |

### 31.20.13 PMC Programmable Clock Register

**Name:** PMC_PCKx
**Offset:** 0x40 + x*0x04 [x=0..7]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRES[7:4] | | | |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PRES[3:0] | | | | | CSS[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 11:4 – PRES[7:0]** Programmable Clock Prescaler

| Value | Description |
|---|---|
| 0-255 | Selected clock is divided by PRES+1. |

**Bits 2:0 – CSS[2:0]** Programmable Clock Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | SLOW_CLK | SLCK is selected |
| 1 | MAIN_CLK | MAINCK is selected |
| 2 | PLLA_CLK | PLLACK is selected |
| 3 | UPLL_CLK | UPLLCKDIV is selected |
| 4 | MCK | MCK is selected |

### 31.20.14 PMC Interrupt Enable Register

**Name:** PMC_IER
**Offset:** 0x0060
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | XT32KERR | | | CFDEV | MOSCRCS | MOSCSELS |
| Access | | | W | | | W | W | W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PCKRDY7 | PCKRDY6 | PCKRDY5 | PCKRDY4 | PCKRDY3 | PCKRDY2 | PCKRDY1 | PCKRDY0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| Access | | W | | | W | | W | |
| Reset | | | | | | | | |

**Bit 21 – XT32KERR** 32.768 kHz Crystal Oscillator Error Interrupt Enable

**Bit 18 – CFDEV** Clock Failure Detector Event Interrupt Enable

**Bit 17 – MOSCRCS** Main RC Oscillator Status Interrupt Enable

**Bit 16 – MOSCSELS** Main Clock Source Oscillator Selection Status Interrupt Enable

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – PCKRDY** Programmable Clock Ready x Interrupt Enable

**Bit 6 – LOCKU** UTMI PLL Lock Interrupt Enable

**Bit 3 – MCKRDY** Master Clock Ready Interrupt Enable

**Bit 1 – LOCKA** PLLA Lock Interrupt Enable

**Bit 0 – MOSCXTS** Main Crystal Oscillator Status Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 31.20.15 PMC Interrupt Disable Register

**Name:** PMC_IDR
**Offset:** 0x0064
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | XT32KERR | | | CFDEV | MOSCRCS | MOSCSELS |
| Access | | | W | | | W | W | W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PCKRDY7 | PCKRDY6 | PCKRDY5 | PCKRDY4 | PCKRDY3 | PCKRDY2 | PCKRDY1 | PCKRDY0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| Access | | W | | | W | | W | W |
| Reset | | | | | | | | |

**Bit 21 – XT32KERR** 32.768 kHz Crystal Oscillator Error Interrupt Disable

**Bit 18 – CFDEV** Clock Failure Detector Event Interrupt Disable

**Bit 17 – MOSCRCS** Main RC Status Interrupt Disable

**Bit 16 – MOSCSELS** Main Clock Source Oscillator Selection Status Interrupt Disable

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – PCKRDY** Programmable Clock Ready x Interrupt Disable

**Bit 6 – LOCKU** UTMI PLL Lock Interrupt Disable

**Bit 3 – MCKRDY** Master Clock Ready Interrupt Disable

**Bit 1 – LOCKA** PLLA Lock Interrupt Disable

**Bit 0 – MOSCXTS** Main Crystal Oscillator Status Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 31.20.16 PMC Status Register

**Name:** PMC_SR
**Offset:** 0x0068
**Reset:** 0x00030008
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | XT32KERR | FOS | CFDS | CFDEV | MOSCRCS | MOSCSELS |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PCKRDY7 | PCKRDY6 | PCKRDY5 | PCKRDY4 | PCKRDY3 | PCKRDY2 | PCKRDY1 | PCKRDY0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OSCSELS | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| Access | R | R | | | R | | R | |
| Reset | 0 | 0 | | | 1 | | 0 | 0 |

**Bit 21 – XT32KERR** Slow Crystal Oscillator Error

| Value | Description |
|---|---|
| 0 | The frequency of the 32.768 kHz crystal oscillator is correct (32.768 kHz ±1%) or the monitoring is disabled. |
| 1 | The frequency of the 32.768 kHz crystal oscillator is incorrect or has been incorrect for an elapsed period of time since the monitoring has been enabled. |

**Bit 20 – FOS** Clock Failure Detector Fault Output Status

| Value | Description |
|---|---|
| 0 | The fault output of the clock failure detector is inactive. |
| 1 | The fault output of the clock failure detector is active. This status is cleared by writing a ‘1’ to FOCLR in PMC_FOCR. |

**Bit 19 – CFDS** Clock Failure Detector Status

| Value | Description |
|---|---|
| 0 | A clock failure of the Main crystal oscillator clock is not detected. |
| 1 | A clock failure of the Main crystal oscillator clock is detected. |

**Bit 18 – CFDEV** Clock Failure Detector Event

| Value | Description |
|---|---|
| 0 | No clock failure detection of the Main crystal oscillator clock has occurred since the last read of PMC_SR. |
| 1 | At least one clock failure detection of the Main crystal oscillator clock has occurred since the last read of PMC_SR. |

**Bit 17 – MOSCRCS** Main RC Oscillator Status

| Value | Description |
|---|---|
| 0 | Main RC oscillator is not stabilized. |
| 1 | Main RC oscillator is stabilized. |

**Bit 16 – MOSCSELS** Main Clock Source Oscillator Selection Status

| Value | Description |
|---|---|
| 0 | Selection is in progress. |
| 1 | Selection is done. |

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – PCKRDY** Programmable Clock Ready Status

| Value | Description |
|---|---|
| 0 | Programmable Clock x is not ready. |
| 1 | Programmable Clock x is ready. |

**Bit 7 – OSCSELS** Slow Clock Source Oscillator Selection

| Value | Description |
|---|---|
| 0 | Slow RC oscillator is selected. |
| 1 | 32.768 kHz crystal oscillator is selected. |

**Bit 6 – LOCKU** UTMI PLL Lock Status

| Value | Description |
|---|---|
| 0 | UTMI PLL is not locked |
| 1 | UTMI PLL is locked. |

**Bit 3 – MCKRDY** Master Clock Status

| Value | Description |
|---|---|
| 0 | Master Clock is not ready. |
| 1 | Master Clock is ready. |

**Bit 1 – LOCKA** PLLA Lock Status

| Value | Description |
|---|---|
| 0 | PLLA is not locked |
| 1 | PLLA is locked. |

**Bit 0 – MOSCXTS** Main Crystal Oscillator Status

| Value | Description |
|---|---|
| 0 | Main crystal oscillator is not stabilized. |
| 1 | Main crystal oscillator is stabilized. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 31.20.17 PMC Interrupt Mask Register

**Name:** PMC_IMR
**Offset:** 0x006C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | XT32KERR | | | CFDEV | MOSCRCS | MOSCSELS |
| Access | | | R | | | R | R | R |
| Reset | | | 0 | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PCKRDY7 | PCKRDY6 | PCKRDY5 | PCKRDY4 | PCKRDY3 | PCKRDY2 | PCKRDY1 | PCKRDY0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | LOCKU | | | MCKRDY | | LOCKA | MOSCXTS |
| Access | | R | | | R | | R | R |
| Reset | | 0 | | | 0 | | 0 | 0 |

**Bit 21 – XT32KERR** 32.768 kHz Crystal Oscillator Error Interrupt Mask

**Bit 18 – CFDEV** Clock Failure Detector Event Interrupt Mask

**Bit 17 – MOSCRCS** Main RC Status Interrupt Mask

**Bit 16 – MOSCSELS** Main Clock Source Oscillator Selection Status Interrupt Mask

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – PCKRDY** Programmable Clock Ready x Interrupt Mask

**Bit 6 – LOCKU** UTMI PLL Lock Interrupt Mask

**Bit 3 – MCKRDY** Master Clock Ready Interrupt Mask

**Bit 1 – LOCKA** PLLA Lock Interrupt Mask

**Bit 0 – MOSCXTS** Main Crystal Oscillator Status Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 31.20.18 PMC Fast Startup Mode Register

**Name:** PMC_FSMR
**Offset:** 0x0070
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FFLPM | FLPM[1:0] | | LPM | | USBAL | RTCAL | RTTAL |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FSTT15 | FSTT14 | FSTT13 | FSTT12 | FSTT11 | FSTT10 | FSTT9 | FSTT8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FSTT7 | FSTT6 | FSTT5 | FSTT4 | FSTT3 | FSTT2 | FSTT1 | FSTT0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 23 – FFLPM** Force Flash Low-power Mode

| Value | Description |
|---|---|
| 0 | The Flash Low-power mode, defined in the FLPM field, is automatically applied when in Wait mode and released when going back to Active mode. |
| 1 | The Flash Low-power mode is user defined by the FLPM field and immediately applied. |

**Bits 22:21 – FLPM[1:0]** Flash Low-power Mode

| Value | Name | Description |
|---|---|---|
| 0 | FLASH_STANDBY | Flash is in Standby Mode when system enters Wait Mode |
| 1 | FLASH_DEEP_POWERDOWN | Flash is in Deep-powerdown mode when system enters Wait Mode |
| 2 | FLASH_IDLE | Idle mode |

**Bit 20 – LPM** Low-power Mode

| Value | Description |
|---|---|
| 0 | The WaitForInterrupt (WFI) or the WaitForEvent (WFE) instruction of the processor makes the processor enter Sleep mode. |
| 1 | The WaitForEvent (WFE) instruction of the processor makes the system enter Wait mode. |

**Bit 18 – USBAL** USB Alarm Enable

| Value | Description |
|---|---|
| 0 | The USB alarm has no effect on the PMC. |
| 1 | The USB alarm enables a fast restart signal to the PMC. |

**Bit 17 – RTCAL** RTC Alarm Enable

| Value | Description |
|---|---|
| 0 | The RTC alarm has no effect on the PMC. |
| 1 | The RTC alarm enables a fast restart signal to the PMC. |

**Bit 16 – RTTAL** RTT Alarm Enable

| Value | Description |
|---|---|
| 0 | The RTT alarm has no effect on the PMC. |
| 1 | The RTT alarm enables a fast restart signal to the PMC. |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – FSTT** Fast Startup Input Enable

| Value | Description |
|---|---|
| 0 | The corresponding wake-up input has no effect on the PMC. |
| 1 | The corresponding wake-up input enables a fast restart signal to the PMC. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 31.20.19 PMC Fast Startup Polarity Register

**Name:** PMC_FSPR
**Offset:** 0x0074
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FSTP15 | FSTP14 | FSTP13 | FSTP12 | FSTP11 | FSTP10 | FSTP9 | FSTP8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FSTP7 | FSTP6 | FSTP5 | FSTP4 | FSTP3 | FSTP2 | FSTP1 | FSTP0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 – FSTP** Fast Startup Input Polarity x bits
Defines the active polarity of the corresponding wake-up input. If the corresponding wake-up input is enabled and at the FSTP level, it enables a fast restart signal.

### 31.20.20 PMC Fault Output Clear Register

**Name:** PMC_FOCR
**Offset:** 0x0078
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | FOCLR |

**Bit 0 – FOCLR** Fault Output Clear
Clears the clock failure detector fault output.

### 31.20.21 PMC Write Protection Mode Register

**Name:** PMC_WPMR
**Offset:** 0x00E4
**Reset:** 0x000000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x504D43 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

See "Register Write Protection" for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x504D43 (“PMC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x504D43 (“PMC” in ASCII). |

### 31.20.22 PMC Write Protection Status Register

**Name:** PMC_WPSR
**Offset:** 0x00E8
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the PMC_WPSR. |
| 1 | A write protection violation has occurred since the last read of the PMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

### 31.20.23 PMC Peripheral Clock Enable Register 1

**Name:** PMC_PCER1
**Offset:** 0x0100
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID62 | | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PID39 | | PID37 | | PID35 | PID34 | PID33 | PID32 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0:3,5,7:28,30 – PID** Peripheral Clock x Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding peripheral clock. |

### 31.20.24 PMC Peripheral Clock Disable Register 1

**Name:** PMC_PCDR1
**Offset:** 0x104
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PCM Write Protection Mode Register

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID62 | | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PID39 | | PID37 | | PID35 | PID34 | PID33 | PID32 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0:3,5,7:28,30 – PID** Peripheral Clock x Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding peripheral clock. |

**Note:** “PIDx” refers to identifiers as defined in the section “Peripheral Identifiers”.

### 31.20.25 PMC Peripheral Clock Status Register 1

**Name:** PMC_PCSR1
**Offset:** 0x0108
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID24 | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID16 | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID8 | PID7 | PID6 | PID5 | PID4 | PID3 | PID2 | PID1 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID0 | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | | | | | | |

**Bits 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral Clock x Status

| Value | Description |
|---|---|
| 0 | The corresponding peripheral clock is disabled. |
| 1 | The corresponding peripheral clock is enabled.<br>Note: “PIDx” refers to identifiers as defined in the section “Peripheral Identifiers”. |

### 31.20.26 PMC Peripheral Control Register

**Name:** PMC_PCR
**Offset:** 0x010C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | GCLKEN | EN | GCLKDIV[7:4] | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GCLKDIV[3:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | CMD | | GCLKCSS[2:0] | | |
| Access | | | | R/W | | R/W | R/W | R/W |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PID[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – GCLKEN** Generic Clock Enable

| Value | Description |
|---|---|
| 0 | The selected generic clock is disabled. |
| 1 | The selected generic clock is enabled. |

**Bit 28 – EN** Enable

| Value | Description |
|---|---|
| 0 | Selected Peripheral clock is disabled. |
| 1 | Selected Peripheral clock is enabled. |

**Bits 27:20 – GCLKDIV[7:0]** Generic Clock Division Ratio
Generic clock is the selected clock period divided by GCLKDIV + 1.
GCLKDIV must not be changed while the peripheral selects GCLKx (e.g., bit rate, etc.).

**Bit 12 – CMD** Command

| Value | Description |
|---|---|
| 0 | Read mode. |
| 1 | Write mode. |

**Bits 10:8 – GCLKCSS[2:0]** Generic Clock Source Selection

| Value | Name | Description |
|---|---|---|
| 0 | SLOW_CLK | SLCK is selected |
| 1 | MAIN_CLK | MAINCK is selected |
| 2 | PLLA_CLK | PLLACK is selected |
| 3 | UPLL_CLK | UPLLCK is selected |
| 4 | MCK_CLK | MCK is selected |

**Bits 6:0 – PID[6:0]** Peripheral ID
Peripheral ID selection from PID2 to PID127.
“PID2 to PID127” refers to identifiers as defined in section “Peripheral Identifiers”.

### 31.20.27 PMC Oscillator Calibration Register

**Name:** PMC_OCR
**Offset:** 0x0110
**Reset:** 0x00404040
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SEL12 | CAL12[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SEL8 | CAL8[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SEL4 | CAL4[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 23 – SEL12** Selection of Main RC Oscillator Calibration Bits for 12 MHz

| Value | Description |
|---|---|
| 0 | Factory-determined value stored in Flash memory. |
| 1 | Value written by user in CAL12 field of this register. |

**Bits 22:16 – CAL12[6:0]** Main RC Oscillator Calibration Bits for 12 MHz
Calibration bits applied to the RC Oscillator when SEL12 is set.

**Bit 15 – SEL8** Selection of Main RC Oscillator Calibration Bits for 8 MHz

| Value | Description |
|---|---|
| 0 | Factory-determined value stored in Flash memory. |
| 1 | Value written by user in CAL8 field of this register. |

**Bits 14:8 – CAL8[6:0]** Main RC Oscillator Calibration Bits for 8 MHz
Calibration bits applied to the RC Oscillator when SEL8 is set.

**Bit 7 – SEL4** Selection of Main RC Oscillator Calibration Bits for 4 MHz

| Value | Description |
|---|---|
| 0 | Default value stored in Flash memory. |
| 1 | Value written by user in CAL4 field of this register. |

**Bits 6:0 – CAL4[6:0]** Main RC Oscillator Calibration Bits for 4 MHz
Calibration bits applied to the RC Oscillator when SEL4 is set.

### 31.20.28 PMC SleepWalking Enable Register 0

**Name:** PMC_SLPWK_ER0
**Offset:** 0x0114
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral x SleepWalking Enable
Not all PIDs can be configured with asynchronous partial wake-up.
Only the following PID can be configured with asynchronous partial wake-up: UARTx and TWIHSx.
The clock of the peripheral must be enabled before using its asynchronous partial wake-up (SleepWalking) function (its associated PIDx field in PMC Peripheral Clock Status Register 0 or PMC Peripheral Clock Status Register 1 is set to ‘1’).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The asynchronous partial wakeup (SleepWalking) function of the corresponding peripheral is enabled.<br>**Note:** “PIDx” refers to identifiers as defined in the section “Peripheral Identifiers” |

### 31.20.29 PMC SleepWalking Enable Register 1

**Name:** PMC_SLPWK_ER1
**Offset:** 0x0134
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID62 | | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PID39 | | PID37 | | PID35 | PID34 | PID33 | PID32 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0:3,5,7:28,30 – PID** Peripheral Clock x Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding peripheral clock. |

### 31.20.30 PMC SleepWalking Disable Register 0

**Name:** PMC_SLPWK_DR0
**Offset:** 0x0118
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral x SleepWalking Disable
Not all PIDs can be configured with asynchronous partial wake-up.
Only the following PIDs can be configured with asynchronous partial wake-up: UARTx and TWIHSx.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The asynchronous partial wakeup (SleepWalking) function of the corresponding peripheral is disabled.<br>Note: “PIDx” refers to identifiers as defined in the section “Peripheral Identifiers”. |

### 31.20.31 PMC SleepWalking Disable Register 1

**Name:** PMC_SLPWK_DR1
**Offset:** 0x0138
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID62 | | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PID39 | | PID37 | | PID35 | PID34 | PID33 | PID32 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0:3,5,7:28,30 – PID** Peripheral Clock x Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding peripheral clock. |

### 31.20.32 PMC SleepWalking Status Register 0

**Name:** PMC_SLPWK_SR0
**Offset:** 0x011C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | | | | | | |

**Bits 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral x SleepWalking Status
Not all PIDs can be configured with asynchronous partial wake-up.
Only the following PIDs can be configured with asynchronous partial wake-up: UARTx and TWIHSx.

| Value | Description |
|---|---|
| 0 | The asynchronous partial wake-up (SleepWalking) function of the peripheral is currently disabled or the peripheral enabled for asynchronous partial wake-up (SleepWalking) cleared the PIDn bit upon detection of a wake-up condition. |
| 1 | The asynchronous partial wake-up (SleepWalking) function of the peripheral is currently enabled.<br>**Note:** “PIDx” refers to identifiers as defined in the section “Peripheral Identifiers”. |

### 31.20.33 PMC SleepWalking Status Register 1

**Name:** PMC_SLPWK_SR1
**Offset:** 0x013C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID62 | | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PID39 | | PID37 | | PID35 | PID34 | PID33 | PID32 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0:3,5,7:28,30 – PID** Peripheral Clock x Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding peripheral clock. |

### 31.20.34 PMC SleepWalking Activity Status Register 0

**Name:** PMC_SLPWK_ASR0
**Offset:** 0x0120
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PID31 | PID30 | PID29 | PID28 | PID27 | PID26 | PID25 | PID24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PID23 | PID22 | PID21 | PID20 | PID19 | PID18 | PID17 | PID16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PID15 | PID14 | PID13 | PID12 | PID11 | PID10 | PID9 | PID8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PID7 | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | | | | | | |

**Bits 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – PIDx** Peripheral x Activity Status
Only the following PIDs can be configured with asynchronous partial wake-up: UARTx and TWIHSx.
All other PIDs are always read at ‘0’.

| Value | Description |
|---|---|
| 0 | The peripheral x is not currently active. The asynchronous partial wake-up (SleepWalking) function can be activated. |
| 1 | The peripheral x is currently active. The asynchronous partial wake-up (SleepWalking) function must not be activated.<br>**Note:** “PIDx” refers to identifiers as defined in the section “Peripheral Identifiers”. |

### 31.20.35 PLL Maximum Multiplier Value Register

**Name:** PMC_PMMR
**Offset:** 0x0130
**Reset:** 0x000007FF
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | PLLA_MMAX[10:8] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PLLA_MMAX[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 10:0 – PLLA_MMAX[10:0]** PLLA Maximum Allowed Multiplier Value
Defines the maximum value of multiplication factor that can be sent to PLLA. Any value of the MULA field (see PMC Clock Generator PLLA Register) above PLLA_MMAX is saturated to PLLA_MMAX. PLLA_MMAX write operation is cancelled in the following cases:

- The value of MULA is currently saturated by PLLA_MMAX.
- The user is trying to write a value of PLLA_MMAX that is smaller than the current value of MULA.

### 31.20.36 PMC SleepWalking Activity Status Register 1

**Name:** PMC_SLPWK_ASR1
**Offset:** 0x0140
**Reset:** 0x00000000
**Property:** Read-only

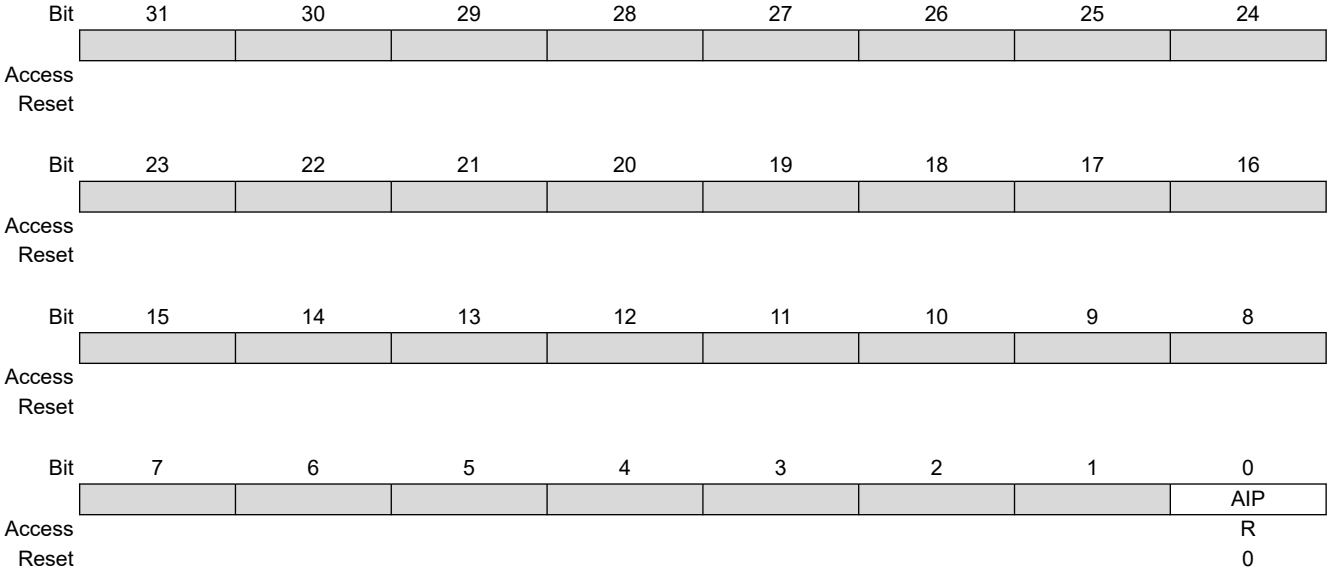
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PID62 | | PID60 | PID59 | PID58 | PID57 | PID56 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | | | PID53 | PID52 | PID51 | PID50 | PID49 | PID48 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PID47 | PID46 | PID45 | PID44 | PID43 | PID42 | PID41 | PID40 |
| Access | | | | | | | | |
| Reset | | | | | | | | |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PID39 | | PID37 | | PID35 | PID34 | PID33 | PID32 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0:3,5,7:28,30 – PID** Peripheral Clock x Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding peripheral clock. |

### 31.20.37 PMC SleepWalking Activity In Progress Register

**Name:** PMC_SLPWK_AIPR
**Offset:** 0x0144
**Reset:** 0x400E0744
**Property:** Read-only




**Bit 0 – AIP** Activity In Progress
Only the following PIDs can be configured with asynchronous partial wakeup: UARTx and TWIHSx.

| Value | Description |
|---|---|
| 0 | There is no activity on peripherals. The asynchronous partial wakeup (SleepWalking) function can be activated on one or more peripherals. The device can enter Wait mode. |
| 1 | One or more peripherals are currently active. The device must not enter Wait mode if the asynchronous partial wakeup is enabled for one of the following PIDs: UARTx and TWIHSx. |

# 32. Parallel Input/Output Controller (PIO)

## 32.1 Description

The Parallel Input/Output Controller (PIO) manages up to 32 fully programmable input/output lines. Each I/O line may be dedicated as a general-purpose I/O or be assigned to a function of an embedded peripheral. This ensures effective optimization of the pins of the product.

Each I/O line is associated with a bit number in all of the 32-bit registers of the 32-bit wide user interface.

Each I/O line of the PIO Controller features the following:

- An input change interrupt enabling level change detection on any I/O line
- Additional Interrupt modes enabling rising edge, falling edge, low-level or high-level detection on any I/O line
- A glitch filter providing rejection of glitches lower than one-half of peripheral clock cycle
- A debouncing filter providing rejection of unwanted pulses from key or push button operations
- Multi-drive capability similar to an open drain I/O line
- Control of the I/O line pullup and pulldown
- Input visibility and output control

The PIO Controller also features a synchronous output providing up to 32 bits of data output in a single write operation.

An 8-bit Parallel Capture mode is also available which can be used to interface a CMOS digital image sensor, an ADC, a DSP synchronous port in Synchronous mode, etc.

## 32.2 Embedded Characteristics

- Up to 32 Programmable I/O Lines
- Fully Programmable through Set/Clear Registers
- Multiplexing of Four Peripheral Functions per I/O Line
- For each I/O Line (Whether Assigned to a Peripheral or Used as General Purpose I/O)
  - Input Change Interrupt
  - Programmable Glitch Filter
  - Programmable Debouncing Filter
  - Multi-drive Option Enables Driving in Open Drain
  - Programmable Pullup on Each I/O Line
  - Pin Data Status Register, Supplies Visibility of the Level on the Pin at Any Time
  - Additional Interrupt Modes on a Programmable Event: Rising Edge, Falling Edge, Low-Level or High-Level
  - Lock of the Configuration by the Connected Peripheral
- Synchronous Output, Provides Set and Clear of Several I/O Lines in a Single Write
- Register Write Protection
- Programmable Schmitt Trigger Inputs
- Programmable I/O Drive
- Parallel Capture Mode
  - Can Be Used to Interface a CMOS Digital Image Sensor, an ADC, etc.
  - One Clock, 8-bit Parallel Data and Two Data Enable on I/O Lines
  - Data Can be Sampled Every Other Time (For Chrominance Sampling Only)
  - Supports Connection of One DMA Controller Channel Which Offers Buffer Reception Without Processor Intervention

© 2021 Microchip Technology Inc.
and its subsidiaries

## 32.3 Block Diagram

**Figure 32-1. Block Diagram**

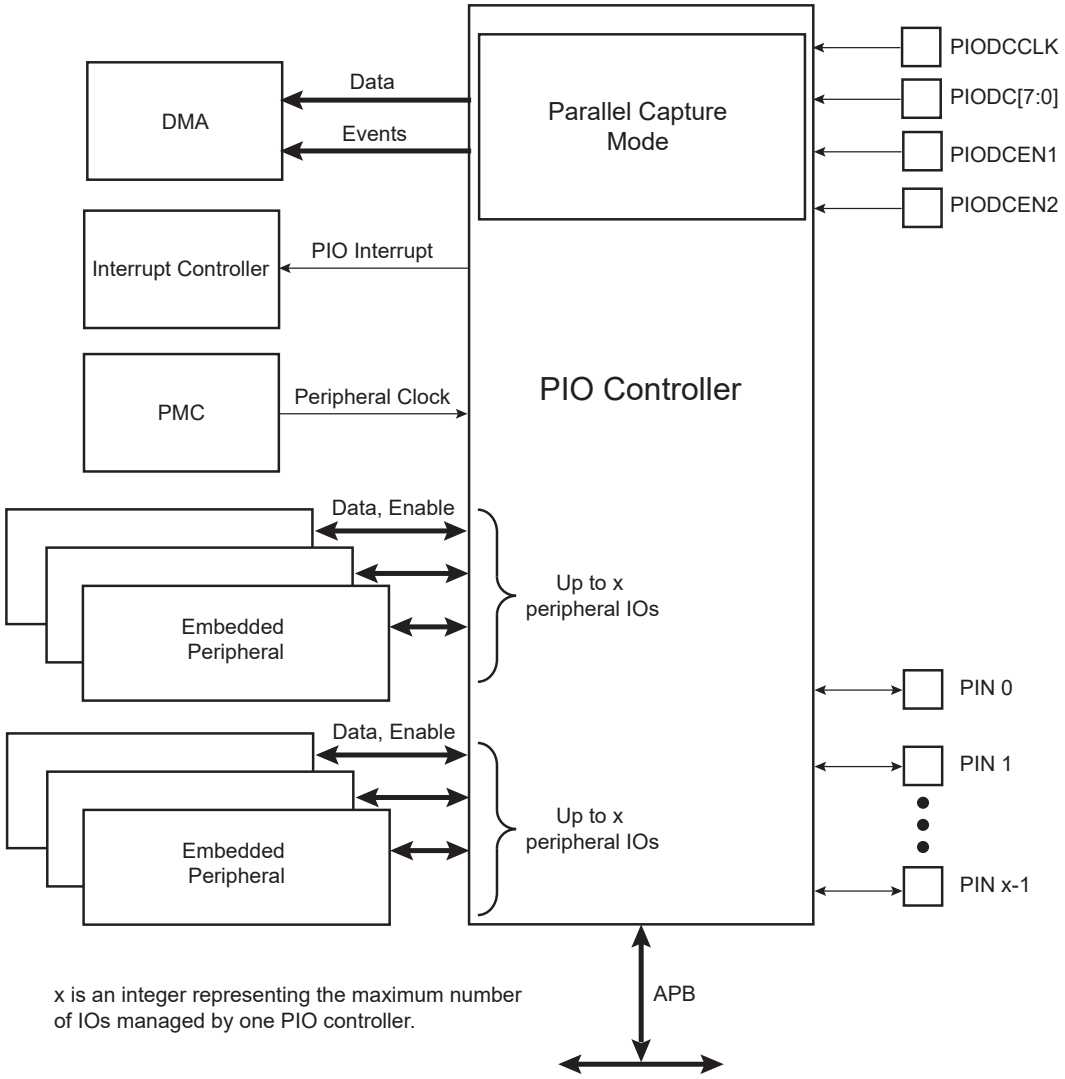



**Table 32-1. Signal Description**

| Signal Name | Signal Description | Signal Type |
|---|---|---|
| PIODCCLK | Parallel Capture Mode Clock | Input |
| PIODC[7:0] | Parallel Capture Mode Data | Input |
| PIODCEN1 | Parallel Capture Mode Data Enable 1 | Input |
| PIODCEN2 | Parallel Capture Mode Data Enable 2 | Input |

## 32.4 Product Dependencies

### 32.4.1 Pin Multiplexing

Each pin is configurable, depending on the product, as either a general-purpose I/O line only, or as an I/O line multiplexed with one or two peripheral I/Os. As the multiplexing is hardware defined and thus product-dependent,

the hardware designer and programmer must carefully determine the configuration of the PIO Controllers required by their application. When an I/O line is general-purpose only, i.e., not multiplexed with any peripheral I/O, programming of the PIO Controller regarding the assignment to a peripheral has no effect and only the PIO Controller can control how the pin is driven by the product.

### 32.4.2 External Interrupt Lines

When the WKUPx input pins must be used as external interrupt lines, the PIO Controller must be configured to disable the peripheral control on these IOs, and the corresponding IO lines must be set to Input mode.

### 32.4.3 Power Management

The Power Management Controller controls the peripheral clock in order to save power. Writing any of the registers of the user interface does not require the peripheral clock to be enabled. This means that the configuration of the I/O lines does not require the peripheral clock to be enabled.

However, when the clock is disabled, not all of the features of the PIO Controller are available, including glitch filtering. Note that the input change interrupt, the interrupt modes on a programmable event and the read of the pin level require the clock to be validated.

After a hardware reset, the peripheral clock is disabled by default.

The user must configure the Power Management Controller before any access to the input line information.

### 32.4.4 Interrupt Sources

For interrupt handling, the PIO Controllers are considered as user peripherals. This means that the PIO Controller interrupt lines are connected among the interrupt sources. Refer to the PIO Controller peripheral identifier in the Peripheral Identifiers table to identify the interrupt sources dedicated to the PIO Controllers. Using the PIO Controller requires the Interrupt Controller to be programmed first.

The PIO Controller interrupt can be generated only if the peripheral clock is enabled.

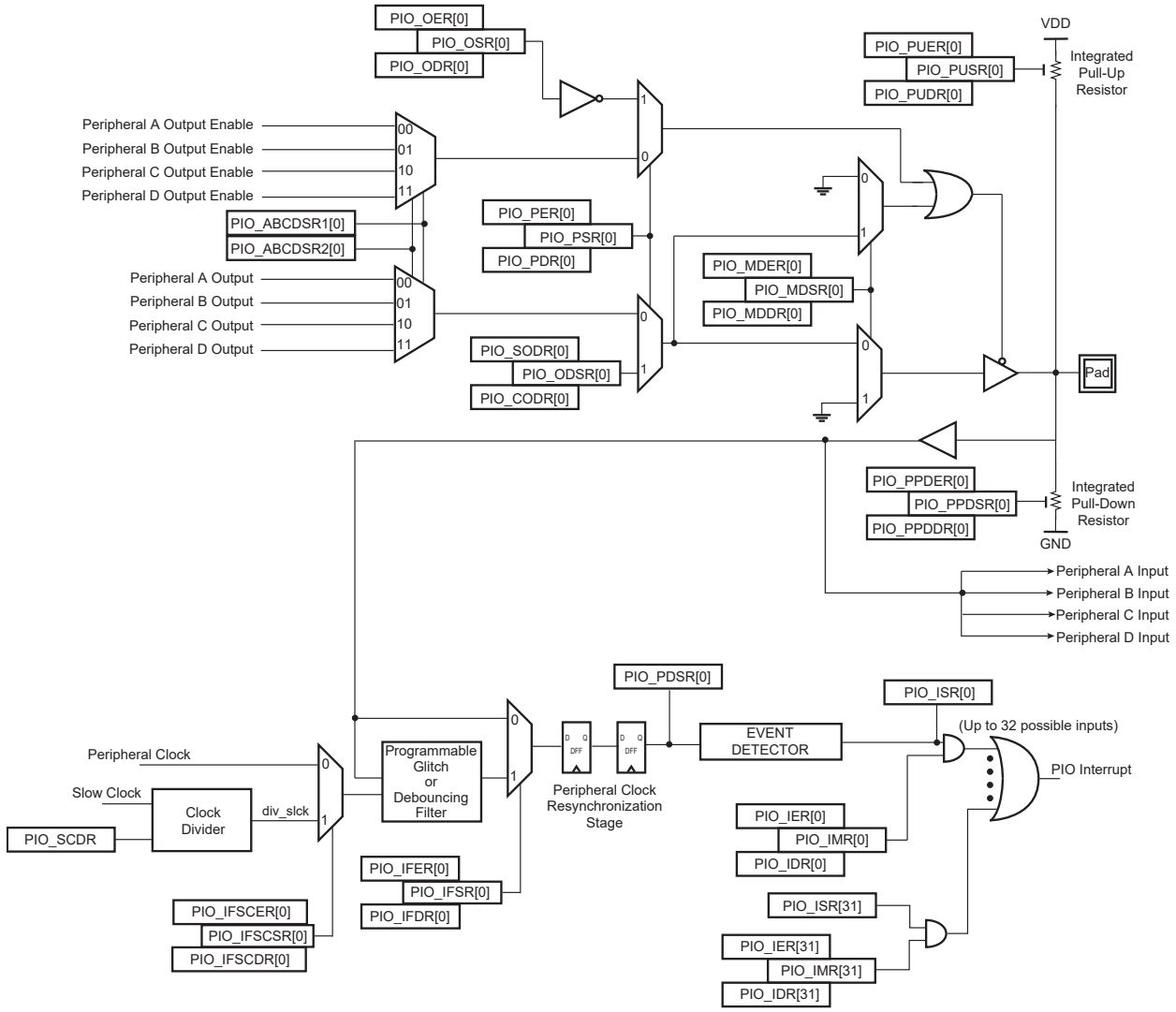
**Related Links**

14.1 Peripheral Identifiers

## 32.5 Functional Description

The PIO Controller features up to 32 fully-programmable I/O lines. Most of the control logic associated to each I/O is represented in the following figure. In this description each signal shown represents one of up to 32 possible indexes.

**Figure 32-2. I/O Line Control Logic**




### 32.5.1 Pullup and Pulldown Resistor Control

Each I/O line is designed with an embedded pullup resistor and an embedded pulldown resistor. The pullup resistor can be enabled or disabled by writing to the Pull-Up Enable Register (PIO_PUER) or Pull-Up Disable Register (PIO_PUDR), respectively. Writing to these registers results in setting or clearing the corresponding bit in the Pull-Up Status Register (PIO_PUSR). Reading a one in PIO_PUSR means the pullup is disabled and reading a zero means the pullup is enabled. The pulldown resistor can be enabled or disabled by writing the Pull-Down Enable Register (PIO_PPDER) or the Pull-Down Disable Register (PIO_PPDDR), respectively. Writing in these registers results in setting or clearing the corresponding bit in the Pull-Down Status Register (PIO_PPDSR). Reading a one in PIO_PPDSR means the pullup is disabled and reading a zero means the pulldown is enabled.

Enabling the pulldown resistor while the pullup resistor is still enabled is not possible. In this case, the write of PIO_PPDER for the relevant I/O line is discarded. Likewise, enabling the pullup resistor while the pulldown resistor is still enabled is not possible. In this case, the write of PIO_PUER for the relevant I/O line is discarded.

Control of the pullup resistor is possible regardless of the configuration of the I/O line.

After reset, depending on the I/O, pullup or pulldown can be set.

### 32.5.2 I/O Line or Peripheral Function Selection

When a pin is multiplexed with one or two peripheral functions, the selection is controlled with the Enable Register (PIO_PER) and the Disable Register (PIO_PDR). The Status Register (PIO_PSR) is the result of the set and clear registers and indicates whether the pin is controlled by the corresponding peripheral or by the PIO Controller. A value of zero indicates that the pin is controlled by the corresponding on-chip peripheral selected in the Peripheral ABCD Select registers (PIO_ABCDSR1 and PIO_ABCDSR2). A value of one indicates the pin is controlled by the PIO Controller.

If a pin is used as a general-purpose I/O line (not multiplexed with an on-chip peripheral), PIO_PER and PIO_PDR have no effect and PIO_PSR returns a one for the corresponding bit.

After reset, the I/O lines are controlled by the PIO Controller, i.e., PIO_PSR resets at one. However, in some events, it is important that PIO lines are controlled by the peripheral (as in the case of memory chip select lines that must be driven inactive after reset, or for address lines that must be driven low for booting out of an external memory). Thus, the reset value of PIO_PSR is defined at the product level and depends on the multiplexing of the device.

### 32.5.3 Peripheral A or B or C or D Selection

Enter a short description of your topic here (optional).

The PIO Controller provides multiplexing of up to four peripheral functions on a single pin. The selection is performed by writing PIO_ABCDSR1 and PIO_ABCDSR2.

For each pin:

- The corresponding bit at level zero in PIO_ABCDSR1 and the corresponding bit at level zero in PIO_ABCDSR2 means peripheral A is selected.
- The corresponding bit at level one in PIO_ABCDSR1 and the corresponding bit at level zero in PIO_ABCDSR2 means peripheral B is selected.
- The corresponding bit at level zero in PIO_ABCDSR1 and the corresponding bit at level one in PIO_ABCDSR2 means peripheral C is selected.
- The corresponding bit at level one in PIO_ABCDSR1 and the corresponding bit at level one in PIO_ABCDSR2 means peripheral D is selected.

Note that multiplexing of peripheral lines A, B, C and D only affects the output line. The peripheral input lines are always connected to the pin input (see I/O Line Control Logic ).

Writing in PIO_ABCDSR1 and PIO_ABCDSR2 manages the multiplexing regardless of the configuration of the pin. However, assignment of a pin to a peripheral function requires a write in PIO_ABCDSR1 and PIO_ABCDSR2 in addition to a write in PIO_PDR.

After reset, PIO_ABCDSR1 and PIO_ABCDSR2 are zero, thus indicating that all the PIO lines are configured on peripheral A. However, peripheral A generally does not drive the pin as the PIO Controller resets in I/O Line mode.

If the software selects a peripheral A, B, C or D which does not exist for a pin, no alternate functions are enabled for this pin and the selection is taken into account. The PIO Controller does not carry out checks to prevent selection of a peripheral which does not exist.

### 32.5.4 Output Control

When the I/O line is assigned to a peripheral function, i.e., the corresponding bit in PIO_PSR is at zero, the drive of the I/O line is controlled by the peripheral. Peripheral A or B or C or D depending on the value in PIO_ABCDSR1 and PIO_ABCDSR2 determines whether the pin is driven or not.

When the I/O line is controlled by the PIO Controller, the pin can be configured to be driven. This is done by writing the Output Enable Register (PIO_OER) and Output Disable Register (PIO_ODR). The results of these write operations are detected in the Output Status Register (PIO_OSR). When a bit in this register is at zero, the corresponding I/O line is used as an input only. When the bit is at one, the corresponding I/O line is driven by the PIO Controller.

The level driven on an I/O line can be determined by writing in the Set Output Data Register (PIO_SODR) and the Clear Output Data Register (PIO_CODR). These write operations, respectively, set and clear the Output Data Status Register (PIO_ODSR), which represents the data driven on the I/O lines. Writing in PIO_OER and PIO_ODR manages PIO_OSR whether the pin is configured to be controlled by the PIO Controller or assigned to a peripheral function. This enables configuration of the I/O line prior to setting it to be managed by the PIO Controller.

© 2021 Microchip Technology Inc. and its subsidiaries

Similarly, writing in PIO_SODR and PIO_CODR affects PIO_ODSR. This is important as it defines the first level driven on the I/O line.

### 32.5.5 Synchronous Data Output

Clearing one or more PIO line(s) and setting another one or more PIO line(s) synchronously cannot be done by using PIO_SODR and PIO_CODR. It requires two successive write operations into two different registers. To overcome this, the PIO Controller offers a direct control of PIO outputs by single write access to PIO_ODSR. Only bits unmasked by the Output Write Status Register (PIO_OWSR) are written. The mask bits in PIO_OWSR are set by writing to the Output Write Enable Register (PIO_OWER) and cleared by writing to the Output Write Disable Register (PIO_OWDR).

After reset, the synchronous data output is disabled on all the I/O lines as PIO_OWSR resets at 0x0.

### 32.5.6 Multi-Drive Control (Open Drain)

Each I/O can be independently programmed in open drain by using the multi-drive feature. This feature permits several drivers to be connected on the I/O line which is driven low only by each device. An external pullup resistor (or enabling of the internal one) is generally required to guarantee a high level on the line.

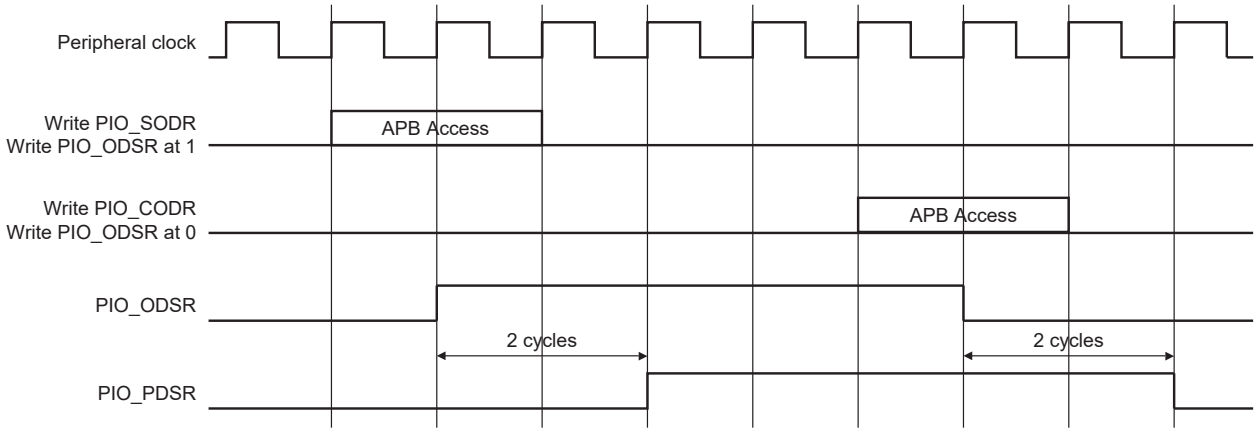
The multi-drive feature is controlled by the Multi-driver Enable Register (PIO_MDER) and the Multi-driver Disable Register (PIO_MDDR). The multi-drive can be selected whether the I/O line is controlled by the PIO Controller or assigned to a peripheral function. The Multi-driver Status Register (PIO_MDSR) indicates the pins that are configured to support external drivers.

After reset, the multi-drive feature is disabled on all pins, i.e., PIO_MDSR resets at value 0x0.

### 32.5.7 Output Line Timings

The following figure shows how the outputs are driven either by writing PIO_SODR or PIO_CODR, or by directly writing PIO_ODSR. This last case is valid only if the corresponding bit in PIO_OWSR is set. The Output Line Timings figure also shows when the feedback in the Pin Data Status Register (PIO_PDSR) is available.

**Figure 32-3. Output Line Timings**




### 32.5.8 Inputs

The level on each I/O line can be read through PIO_PDSR. This register indicates the level of the I/O lines regardless of their configuration, whether uniquely as an input, or driven by the PIO Controller, or driven by a peripheral.

Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSR reads the levels present on the I/O line at the time the clock was disabled.

### 32.5.9 Input Glitch and Debouncing Filters

Optional input glitch and debouncing filters are independently programmable on each I/O line.

The glitch filter can filter a glitch with a duration of less than 1/2 peripheral clock and the debouncing filter can filter a pulse of less than 1/2 period of a programmable divided slow clock.

© 2021 Microchip Technology Inc.
and its subsidiaries

The selection between glitch filtering or debounce filtering is done by writing in the PIO Input Filter Slow Clock Disable Register (PIO_IFSCDR) and the PIO Input Filter Slow Clock Enable Register (PIO_IFSCER). Writing PIO_IFSCDR and PIO_IFSCER, respectively, clears and sets bits in the Input Filter Slow Clock Status Register (PIO_IFSCSR).

The current selection status can be checked by reading the PIO_IFSCSR.

- If PIO_IFSCSR[i] = 0: The glitch filter can filter a glitch with a duration of less than 1/2 peripheral clock period.
- If PIO_IFSCSR[i] = 1: The debouncing filter can filter a pulse with a duration of less than 1/2 programmable divided slow clock period.

For the debouncing filter, the period of the divided slow clock is defined by writing in the DIV field of the Slow Clock Divider Debouncing Register (PIO_SCDR):

$t_{div_slck} = ((DIV + 1) \times 2) \times t_{slck}$

When the glitch or debouncing filter is enabled, a glitch or pulse with a duration of less than 1/2 selected clock cycle (selected clock represents peripheral clock or divided slow clock depending on PIO_IFSCDR and PIO_IFSCER programming) is automatically rejected, while a pulse with a duration of one selected clock (peripheral clock or divided slow clock) cycle or more is accepted. For pulse durations between 1/2 selected clock cycle and one selected clock cycle, the pulse may or may not be taken into account, depending on the precise timing of its occurrence. Thus for a pulse to be visible, it must exceed one selected clock cycle, whereas for a glitch to be reliably filtered out, its duration must not exceed 1/2 selected clock cycle.

The filters also introduce some latencies, illustrated in the following two figures.

The glitch filters are controlled by the Input Filter Enable Register (PIO_IFER), the Input Filter Disable Register (PIO_IFDR) and the Input Filter Status Register (PIO_IFSR). Writing PIO_IFER and PIO_IFDR respectively sets and clears bits in PIO_IFSR. This last register enables the glitch filter on the I/O lines.

When the glitch and/or debouncing filter is enabled, it does not modify the behavior of the inputs on the peripherals. It acts only on the value read in PIO_PDSR and on the input change interrupt detection. The glitch and debouncing filters require that the peripheral clock is enabled.

**Figure 32-4. Input Glitch Filter Timing**

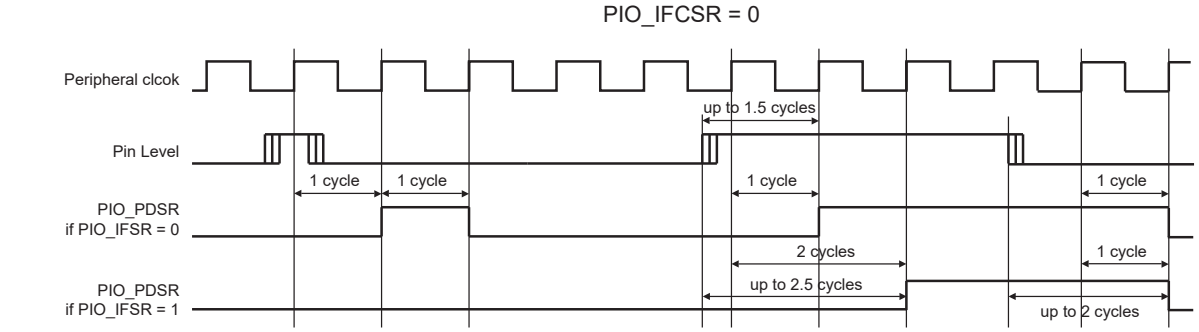
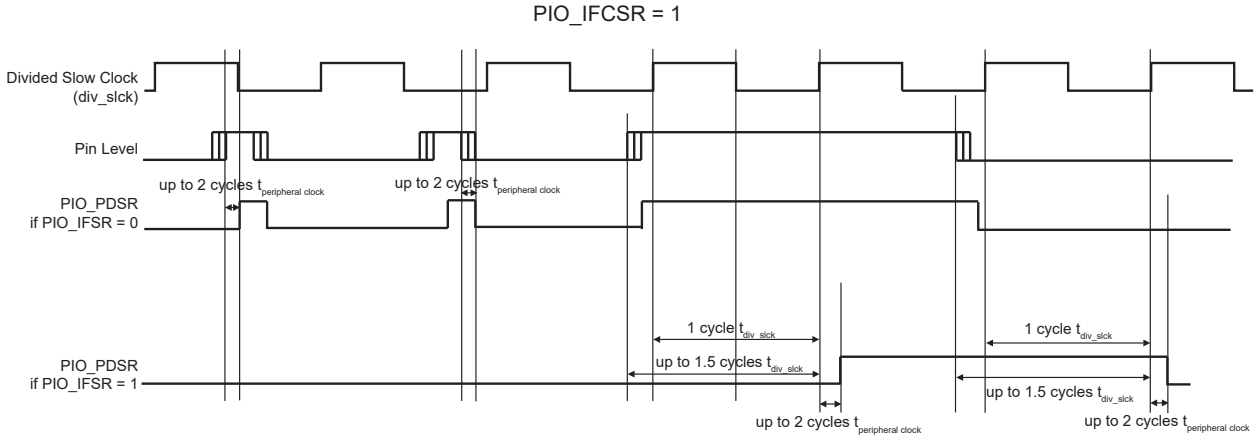



**Figure 32-5. Input Debouncing Filter Timing**



### 32.5.10 Input Edge/Level Interrupt

The PIO Controller can be programmed to generate an interrupt when it detects an edge or a level on an I/O line. The Input Edge/Level interrupt is controlled by writing the Interrupt Enable Register (PIO_IER) and the Interrupt Disable Register (PIO_IDR), which enable and disable the input change interrupt respectively by setting and clearing the corresponding bit in the Interrupt Mask Register (PIO_IMR). As input change detection is possible only by comparing two successive samplings of the input of the I/O line, the peripheral clock must be enabled. The Input Change interrupt is available regardless of the configuration of the I/O line, i.e., configured as an input only, controlled by the PIO Controller or assigned to a peripheral function.

By default, the interrupt can be generated at any time an edge is detected on the input.

Some additional interrupt modes can be enabled/disabled by writing in the Additional Interrupt Modes Enable Register (PIO_AIMER) and Additional Interrupt Modes Disable Register (PIO_AIMDR). The current state of this selection can be read through the Additional Interrupt Modes Mask Register (PIO_AIMMR).

These additional modes are:

- Rising edge detection
- Falling edge detection
- Low-level detection
- High-level detection

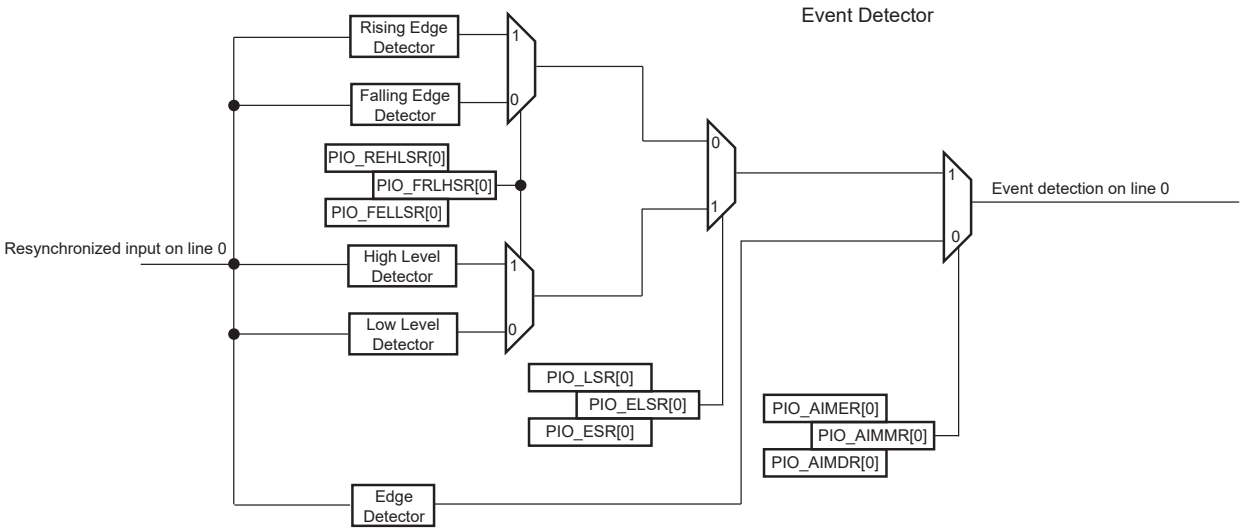
In order to select an additional interrupt mode:

- The type of event detection (edge or level) must be selected by writing in the Edge Select Register (PIO_ESR) and Level Select Register (PIO_LSR) which select, respectively, the edge and level detection. The current status of this selection is accessible through the Edge/Level Status Register (PIO_ELSR).
- The polarity of the event detection (rising/falling edge or high/low-level) must be selected by writing in the Falling Edge/Low-Level Select Register (PIO_FELLSR) and Rising Edge/High-Level Select Register (PIO_REHLSR) which allow to select falling or rising edge (if edge is selected in PIO_ELSR) edge or high- or low-level detection (if level is selected in PIO_ELSR). The current status of this selection is accessible through the Fall/Rise - Low/High Status Register (PIO_FRLHSR).

When an input edge or level is detected on an I/O line, the corresponding bit in the Interrupt Status Register (PIO_ISR) is set. If the corresponding bit in PIO_IMR is set, the PIO Controller interrupt line is asserted.The interrupt signals of the 32 channels are ORed-wired together to generate a single interrupt signal to the interrupt controller.

When the software reads PIO_ISR, all the interrupts are automatically cleared. This signifies that all the interrupts that are pending when PIO_ISR is read must be handled. When an Interrupt is enabled on a “level”, the interrupt is generated as long as the interrupt source is not cleared, even if some read accesses in PIO_ISR are performed.

**Figure 32-6. Event Detector on Input Lines (Figure Represents Line 0)**




Example of interrupt generation on following lines:

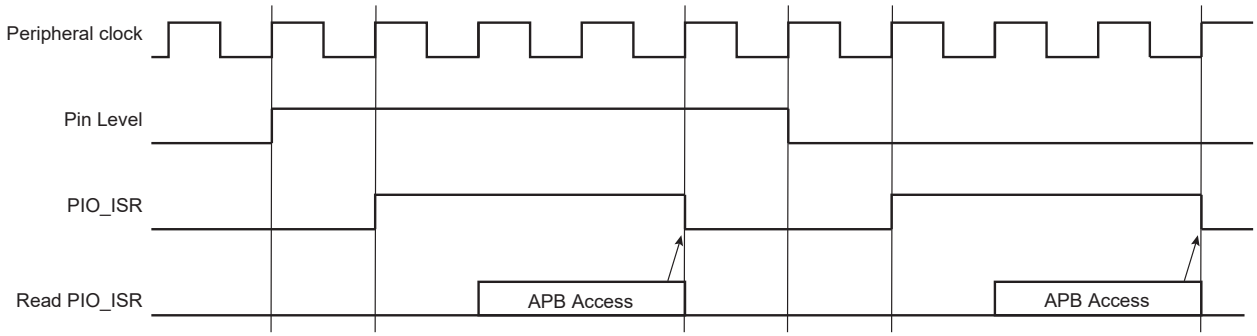
- Rising edge on PIO line 0
- Falling edge on PIO line 1
- Rising edge on PIO line 2
- Low-level on PIO line 3
- High-level on PIO line 4
- High-level on PIO line 5
- Falling edge on PIO line 6
- Rising edge on PIO line 7
- Any edge on the other lines

The following table provides the required configuration for this example.

**Table 32-2. Configuration for Example Interrupt Generation**

| Configuration | Description |
|---|---|
| Interrupt Mode | All the interrupt sources are enabled by writing 32'hFFFF_FFFF in PIO_IER.<br>Then the additional Interrupt mode is enabled for lines 0 to 7 by writing 32'h0000_00FF in PIO_AIMER. |
| Edge or Level Detection | Lines 3, 4 and 5 are configured in level detection by writing 32'h0000_0038 in PIO_LSR.<br>The other lines are configured in edge detection by default, if they have not been previously configured. Otherwise, lines 0, 1, 2, 6 and 7 must be configured in edge detection by writing 32'h0000_00C7 in PIO_ESR. |
| Falling/Rising Edge or Low/High-Level Detection | Lines 0, 2, 4, 5 and 7 are configured in rising edge or high-level detection by writing 32'h0000_00B5 in PIO_REHLSR.<br>The other lines are configured in falling edge or low-level detection by default if they have not been previously configured. Otherwise, lines 1, 3 and 6 must be configured in falling edge/low-level detection by writing 32'h0000_004A in PIO_FELLSR. |

**Figure 32-7. Input Change Interrupt Timings When No Additional Interrupt Modes**




### 32.5.11 I/O Lines Lock

When an I/O line is controlled by a peripheral (particularly the Pulse Width Modulation Controller PWM), it can become locked by the action of this peripheral via an input of the PIO Controller. When an I/O line is locked, the write of the corresponding bit in PIO_PER, PIO_PDR, PIO_MDER, PIO_MDDR, PIO_PUDR, PIO_PUER, PIO_ABCDSR1 and PIO_ABCDSR2 is discarded in order to lock its configuration. The user can know at any time which I/O line is locked by reading the PIO Lock Status Register (PIO_LOCKSR). Once an I/O line is locked, the only way to unlock it is to apply a hardware reset to the PIO Controller.

### 32.5.12 Programmable I/O Drive

It is possible to configure the I/O drive for pads PA0-31, PB0-9, PB12-13, PC0-31, PD0-31 and PE0-5. Refer to the section "Electrical Characteristics".

### 32.5.13 Programmable Schmitt Trigger

It is possible to configure each input for the Schmitt trigger. By default the Schmitt trigger is active. Disabling the Schmitt trigger is requested when using the QTouch® Library.

### 32.5.14 Parallel Capture Mode

#### 32.5.14.1 Overview

The PIO Controller integrates an interface able to read data from a CMOS digital image sensor, a high-speed parallel ADC, a DSP synchronous port in Synchronous mode, etc. For better understanding and to ease reading, the following description uses an example with a CMOS digital image sensor.

#### 32.5.14.2 Functional Description

The CMOS digital image sensor provides a sensor clock, an 8-bit data synchronous with the sensor clock and two data enables which are also synchronous with the sensor clock.

**Figure 32-8. PIO Controller Connection with CMOS Digital Image Sensor**

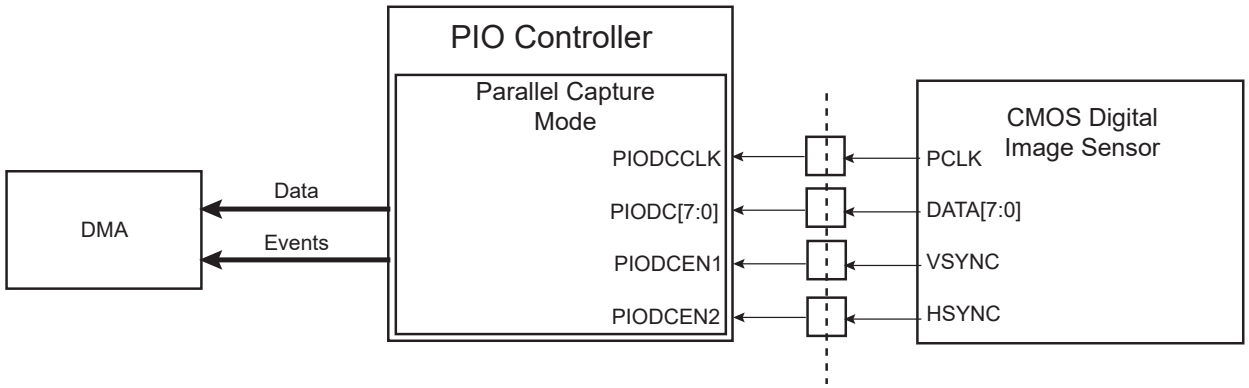
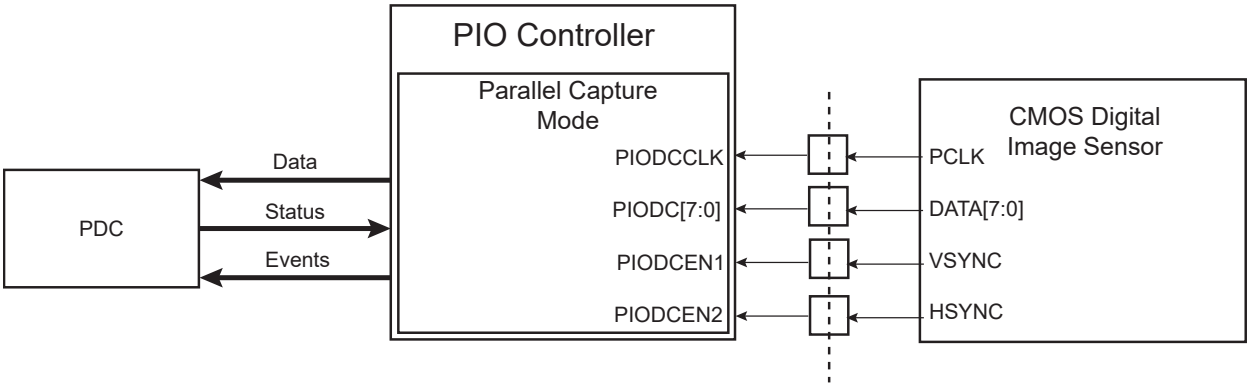



**Figure 32-9.**




As soon as the Parallel Capture mode is enabled by writing a one to the PCEN bit in PIO_PCMR, the I/O lines connected to the sensor clock (PIODCCLK), the sensor data (PIODC[7:0]) and the sensor data enable signals (PIODCEN1 and PIODCEN2) are configured automatically as inputs. To know which I/O lines are associated with the sensor clock, the sensor data and the sensor data enable signals, refer to the I/O multiplexing table(s) in the section “Package and Pinout”.

Once enabled, the Parallel Capture mode samples the data at rising edge of the sensor clock and resynchronizes it with the peripheral clock domain.

The size of the data which can be read in PIO_PCRHR can be programmed using the DSIZE field in PIO_PCMR. If this data size is larger than 8 bits, then the Parallel Capture mode samples several sensor data to form a concatenated data of size defined by DSIZE. Then this data is stored in PIO_PCRHR and the flag DRDY is set to one in PIO_PCISR.

The Parallel Capture mode can be associated with a reception channel of the DMA Controller. This performs reception transfer from Parallel Capture mode to a memory buffer without any intervention from the CPU.

The Parallel Capture mode can take into account the sensor data enable signals or not. If the bit ALWYS is set to zero in PIO_PCMR, the Parallel Capture mode samples the sensor data at the rising edge of the sensor clock only if both data enable signals are active (at one). If the bit ALWYS is set to one, the Parallel Capture mode samples the sensor data at the rising edge of the sensor clock whichever the data enable signals are.

The Parallel Capture mode can sample the sensor data only one time out of two. This is particularly useful when the user wants only to sample the luminance Y of a CMOS digital image sensor which outputs a YUV422 data stream. If the HALFS bit is set to zero in PIO_PCMR, the Parallel Capture mode samples the sensor data in the conditions described above. If the HALFS bit is set to one in PIO_PCMR, the Parallel Capture mode samples the sensor data in the conditions described above, but only one time out of two. Depending on the FRSTS bit in PIO_PCMR, the sensor can either sample the even or odd sensor data. If sensor data are numbered in the order that they are received with an index from zero to n, if FRSTS equals zero then only data with an even index are sampled. If FRSTS equals one, then only data with an odd index are sampled. If data is ready in PIO_PCRHR and it is not read before a new data is stored in PIO_PCRHR, then an overrun error occurs. The previous data is lost and the OVRE flag in PIO_PCISR is set to one. This flag is automatically reset when PIO_PCISR is read (reset after read).

The flags DRDY and OVRE can be a source of the PIO interrupt.

**Figure 32-10. Parallel Capture Mode Waveforms (DSIZE = 2, ALWYS = 0, HALFS = 0)**

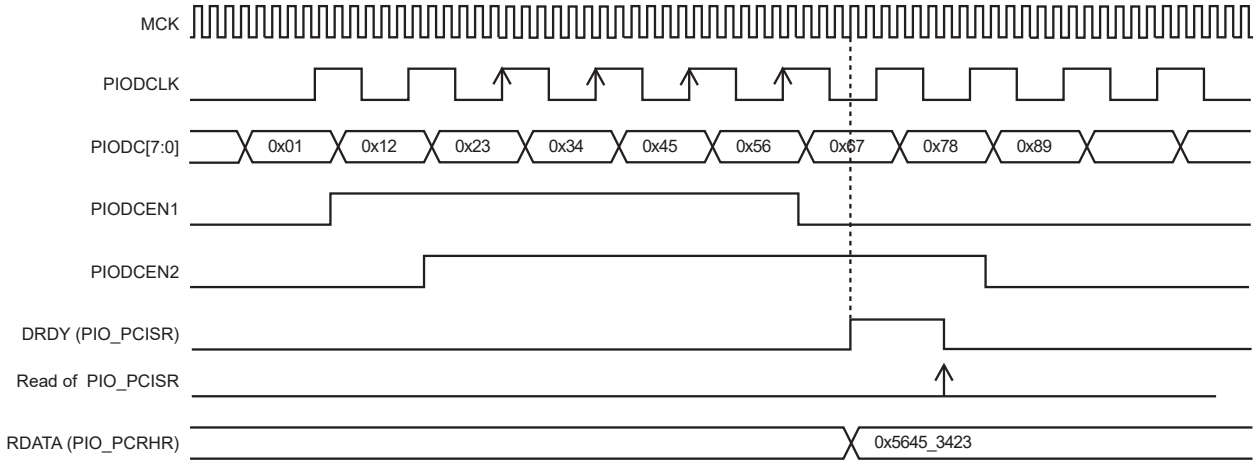



**Figure 32-11. Parallel Capture Mode Waveforms (DSIZE = 2, ALWYS = 1, HALFS = 0)**

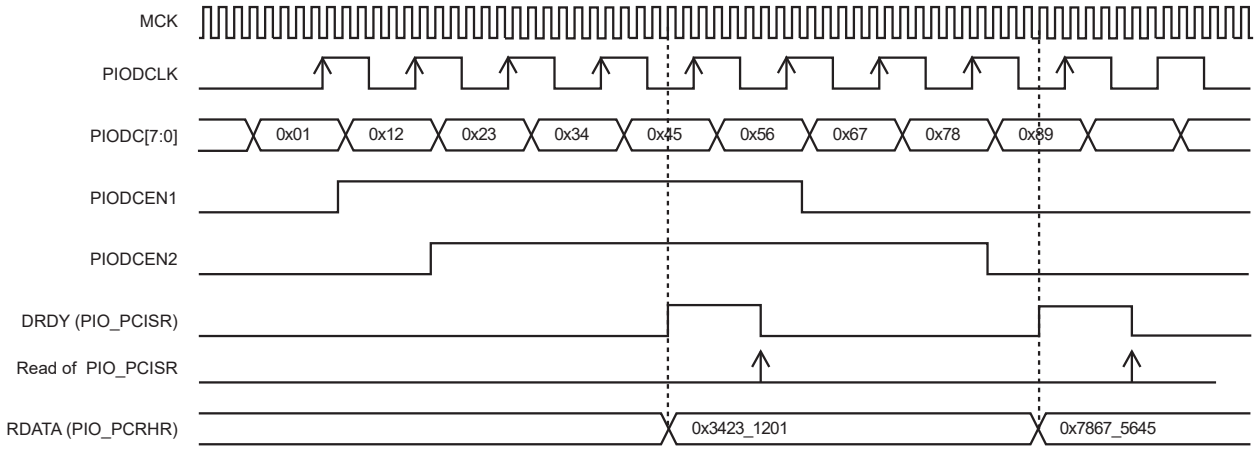


**Figure 32-12. Parallel Capture Mode Waveforms (DSIZE = 2, ALWYS = 0, HALFS = 1, FRSTS = 0)**

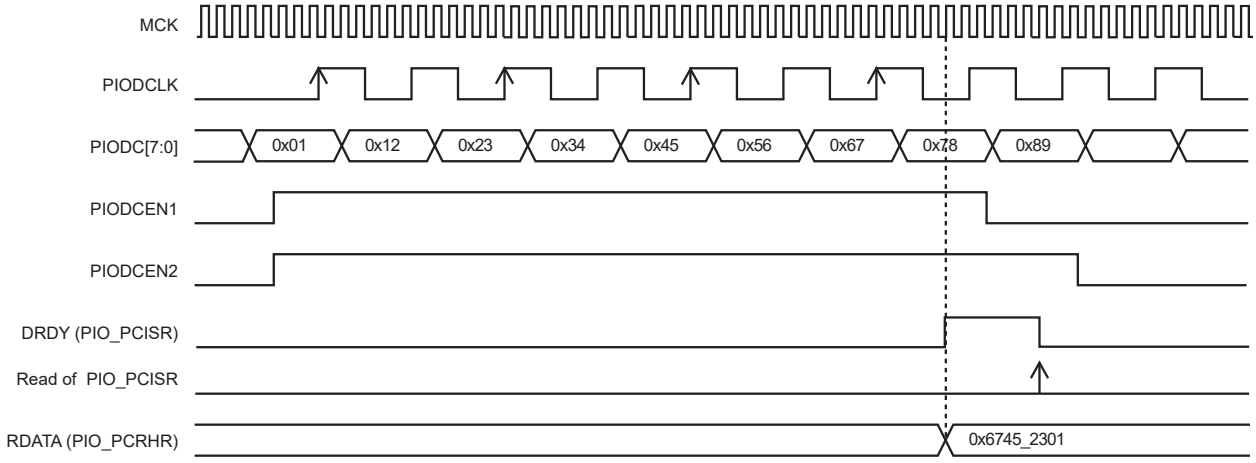
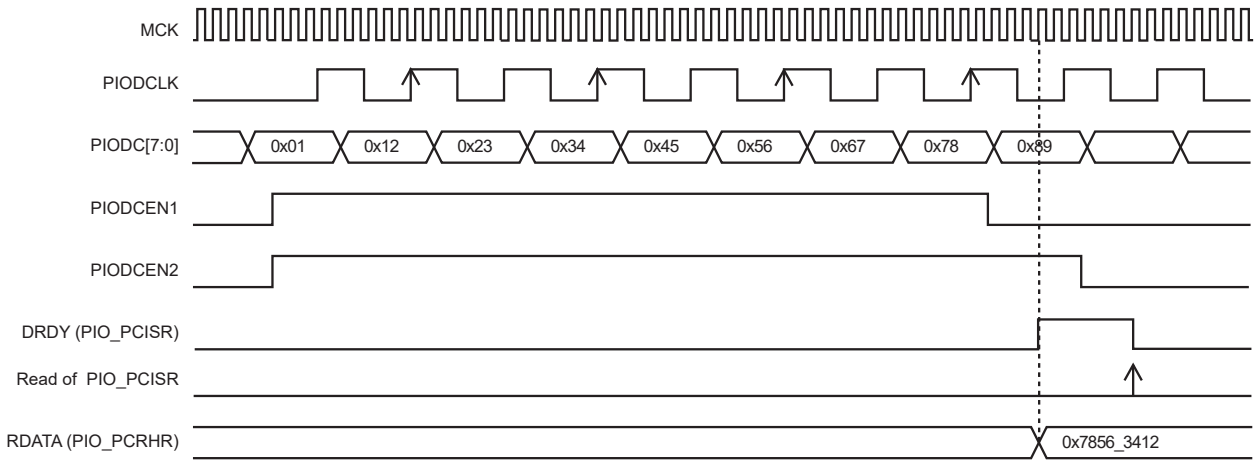



**Figure 32-13. Parallel Capture Mode Waveforms (DSIZE = 2, ALWYS = 0, HALFS = 1, FRSTS = 1)**




#### 32.5.14.3 Restrictions

- Configuration fields DSIZE, ALWYS, HALFS and FRSTS in PIO_PCMR can be changed ONLY if the Parallel Capture mode is disabled at this time (PCEN = 0 in PIO_PCMR).
- The frequency of peripheral clock must be strictly superior to two times the frequency of the clock of the device which generates the parallel data.

#### 32.5.14.4 Programming Sequence

##### 32.5.14.4.1 Without DMA

1. Write PIO_PCIDR and PIO_PCIER in order to configure the Parallel Capture mode interrupt mask.
2. Write PIO_PCMR to set the fields DSIZE, ALWYS, HALFS and FRSTS in order to configure the Parallel Capture mode WITHOUT enabling the Parallel Capture mode.
3. Write PIO_PCMR to set the PCEN bit to one in order to enable the Parallel Capture mode WITHOUT changing the previous configuration.
4. Wait for a data ready by polling the DRDY flag in PIO_PCISR or by waiting for the corresponding interrupt.
5. Check OVRE flag in PIO_PCISR.
6. Read the data in PIO_PCRHR.
7. If new data are expected, go to step 4.
8. Write PIO_PCMR to set the PCEN bit to zero in order to disable the Parallel Capture mode WITHOUT changing the previous configuration.

#### 32.5.14.4.2 With DMA

1. Write PIO_PCIDR and PIO_PCIER in order to configure the Parallel Capture mode interrupt mask.
2. Configure DMA transfer in DMA registers.
3. Write PIO_PCMR to set the fields DSIZE, ALWYS, HALFS and FRSTS in order to configure the Parallel Capture mode WITHOUT enabling the Parallel Capture mode.
4. Write PIO_PCMR to set PCEN bit to one in order to enable the Parallel Capture mode WITHOUT changing the previous configuration.
5. Wait for the DMA status flag to indicate that the buffer transfer is complete.
6. Check OVRE flag in PIO_PCISR.
7. If a new buffer transfer is expected, go to step 5.
8. Write PIO_PCMR to set the PCEN bit to zero in order to disable the Parallel Capture mode WITHOUT changing the previous configuration.

### 32.5.15 I/O Lines Programming Example

The programming example shown in the following table is used to obtain the following configuration:

- 4-bit output port on I/O lines 0 to 3 (should be written in a single write operation), open-drain, with pullup resistor
- Four output signals on I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pullup resistor, no pulldown resistor
- Four input signals on I/O lines 8 to 11 (to read push-button states for example), with pullup resistors, glitch filters and input change interrupts
- Four input signals on I/O line 12 to 15 to read an external device status (polled, thus no input change interrupt), no pullup resistor, no glitch filter
- I/O lines 16 to 19 assigned to peripheral A functions with pullup resistor
- I/O lines 20 to 23 assigned to peripheral B functions with pulldown resistor
- I/O lines 24 to 27 assigned to peripheral C with input change interrupt, no pullup resistor and no pulldown resistor
- I/O lines 28 to 31 assigned to peripheral D, no pullup resistor and no pulldown resistor

**Table 32-3. Programming Example**

| Register | Value to be Written |
|---|---|
| PIO_PER | 0x0000_FFFF |
| PIO_PDR | 0xFFFF_0000 |
| PIO_OER | 0x0000_00FF |
| PIO_ODR | 0xFFFF_FF00 |
| PIO_IFER | 0x0000_0F00 |
| PIO_IFDR | 0xFFFF_F0FF |
| PIO_SODR | 0x0000_0000 |
| PIO_CODR | 0x0FFF_FFFF |
| PIO_IER | 0x0F00_0F00 |
| PIO_IDR | 0xF0FF_F0FF |
| PIO_MDER | 0x0000_000F |
| PIO_MDDR | 0xFFFF_FFF0 |
| PIO_PUDR | 0xFFF0_00F0 |
| PIO_PUER | 0x000F_FF0F |
| PIO_PPDDR | 0xFF0F_FFFF |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | |
|---|---|
| **Register** | **Value to be Written** |
| PIO_PPDER | 0x00F0_0000 |
| PIO_ABCDSR1 | 0xF0F0_0000 |
| PIO_ABCDSR2 | 0xFF00_0000 |
| PIO_OWER | 0x0000_000F |
| PIO_OWDR | 0x0FFF_FFF0 |

### 32.5.16 Register Write Protection

To prevent any single software error from corrupting PIO behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the PIO Write Protection Mode Register (PIO_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the PIO Write Protection Status Register (PIO_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the PIO_WPSR.

The following registers can be write-protected:

- PIO Enable Register
- PIO Disable Register
- PIO Output Enable Register
- PIO Output Disable Register
- PIO Input Filter Enable Register
- PIO Input Filter Disable Register
- PIO Multi-driver Enable Register
- PIO Multi-driver Disable Register
- PIO Pull-Up Disable Register
- PIO Pull-Up Enable Register
- PIO Peripheral ABCD Select Register 1
- PIO Peripheral ABCD Select Register 2
- PIO Output Write Enable Register
- PIO Output Write Disable Register
- PIO Pad Pull-Down Disable Register
- PIO Pad Pull-Down Enable Register
- PIO Parallel Capture Mode Register

© 2021 Microchip Technology Inc.
and its subsidiaries

## 32.6 Register Summary

Each I/O line controlled by the PIO Controller is associated with a bit in each of the PIO Controller User Interface registers. Each register is 32 bits wide. If a parallel I/O line is not defined, writing to the corresponding bits has no effect. Undefined bits read zero. If the I/O line is not multiplexed with any peripheral, the I/O line is controlled by the PIO Controller and PIO_PSR returns one systematically.

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | PIO_PER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x04 | PIO_PDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x08 | PIO_PSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | PIO_OER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x14 | PIO_ODR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x18 | PIO_OSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x1C ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | PIO_IFER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x24 | PIO_IFDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x28 | PIO_IFSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x2C ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | PIO_SODR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x34 | PIO_CODR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x38 | PIO_ODSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x3C | PIO_PDSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x40 | PIO_IER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x44 | PIO_IDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x48 | PIO_IMR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x4C | PIO_ISR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x50 | PIO_MDER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x54 | PIO_MDDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x58 | PIO_MDSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x5C ... 0x5F | Reserved | | | | | | | | | |
| 0x60 | PIO_PUDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x64 | PIO_PUER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x68 | PIO_PUSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x6C ... 0x6F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x70 | PIO_ABCDSR1 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x74 | PIO_ABCDSR2 | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x78 ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | PIO_IFSCDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x84 | PIO_IFSCER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x88 | PIO_IFSCSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x8C | PIO_SCDR | 7:0 | DIV[7:0] | | | | | | | |
| | | 15:8 | | | DIV[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x90 | PIO_PPDDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x94 | PIO_PPDER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x98 | PIO_PPDSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0x9C ... 0x9F | Reserved | | | | | | | | | |
| 0xA0 | PIO_OWER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xA4 | PIO_OWDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xA8 | PIO_OWSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xAC ... 0xAF | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xB0 | PIO_AIMER | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xB4 | PIO_AIMDR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xB8 | PIO_AIMMR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xBC ... 0xBF | Reserved | | | | | | | | | |
| 0xC0 | PIO_ESR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xC4 | PIO_LSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xC8 | PIO_ELSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xCC ... 0xCF | Reserved | | | | | | | | | |
| 0xD0 | PIO_FELLSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xD4 | PIO_REHLSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xD8 | PIO_FRLHSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xDC ... 0xDF | Reserved | | | | | | | | | |
| 0xE0 | PIO_LOCKSR | 7:0 | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| | | 15:8 | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| | | 23:16 | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| | | 31:24 | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| 0xE4 | PIO_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | PIO_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xEC ... 0xFF | Reserved | | | | | | | | | |
| 0x0100 | PIO_SCHMITT | 7:0 | SCHMITT7 | SCHMITT6 | SCHMITT5 | SCHMITT4 | SCHMITT3 | SCHMITT2 | SCHMITT1 | SCHMITT0 |
| | | 15:8 | SCHMITT15 | SCHMITT14 | SCHMITT13 | SCHMITT12 | SCHMITT11 | SCHMITT10 | SCHMITT9 | SCHMITT8 |
| | | 23:16 | SCHMITT23 | SCHMITT22 | SCHMITT21 | SCHMITT20 | SCHMITT19 | SCHMITT18 | SCHMITT17 | SCHMITT16 |
| | | 31:24 | SCHMITT31 | SCHMITT30 | SCHMITT29 | SCHMITT28 | SCHMITT27 | SCHMITT26 | SCHMITT25 | SCHMITT24 |
| 0x0104 ... 0x0117 | Reserved | | | | | | | | | |
| 0x0118 | PIO_DRIVER1 | 7:0 | LINE7 | LINE6 | LINE5 | LINE4 | LINE3 | LINE2 | LINE1 | LINE0 |
| | | 15:8 | LINE15 | LINE14 | LINE13 | LINE12 | LINE11 | LINE10 | LINE9 | LINE8 |
| | | 23:16 | LINE23 | LINE22 | LINE21 | LINE20 | LINE19 | LINE18 | LINE17 | LINE16 |
| | | 31:24 | LINE31 | LINE30 | LINE29 | LINE28 | LINE27 | LINE26 | LINE25 | LINE24 |
| 0x011C ... 0x014F | Reserved | | | | | | | | | |
| 0x0150 | PIO_PCMR | 7:0 | | | DSIZE[1:0] | | | | | PCEN |
| | | 15:8 | | | | | FRSTS | HALFS | ALWYS | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0154 | PIO_PCIER | 7:0 | | | | | RXBUFF | ENDRX | OVRE | DRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | PIO_PCIDR | 7:0 | | | | | RXBUFF | ENDRX | OVRE | DRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x015C | PIO_PCIMR | 7:0 | | | | | RXBUFF | ENDRX | OVRE | DRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0160 | PIO_PCISR | 7:0 | | | | | | | OVRE | DRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0164 | PIO_PCRHR | 7:0 | RDATA[7:0] | | | | | | | |
| | | 15:8 | RDATA[15:8] | | | | | | | |
| | | 23:16 | RDATA[23:16] | | | | | | | |
| | | 31:24 | RDATA[31:24] | | | | | | | |

### 32.6.1 PIO Enable Register

**Name:** PIO_PER
**Offset:** 0x0000
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the PIO to control the corresponding pin (disables peripheral control of the pin). |

### 32.6.2 PIO Disable Register

**Name:** PIO_PDR
**Offset:** 0x0004
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the PIO from controlling the corresponding pin (enables peripheral control of the pin). |

### 32.6.3 PIO Status Register

**Name:** PIO_PSR
**Offset:** 0x0008
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Status

| Value | Description |
|---|---|
| 0 | PIO is inactive on the corresponding I/O line (peripheral is active). |
| 1 | PIO is active on the corresponding I/O line (peripheral is inactive). |

### 32.6.4 PIO Output Enable Register

**Name:** PIO_OER
**Offset:** 0x0010
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Output Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the output on the I/O line. |

### 32.6.5 PIO Output Disable Register

**Name:** PIO_ODR
**Offset:** 0x0014
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Output Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the output on the I/O line. |

### 32.6.6 PIO Output Status Register

**Name:** PIO_OSR
**Offset:** 0x0018
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Output Status

| Value | Description |
|---|---|
| 0 | The I/O line is a pure input. |
| 1 | The I/O line is enabled in output. |

### 32.6.7 PIO Input Filter Enable Register

**Name:** PIO_IFER
**Offset:** 0x0020
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Input Filter Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the input glitch filter on the I/O line. |

### 32.6.8 PIO Input Filter Disable Register

**Name:** PIO_IFDR
**Offset:** 0x0024
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Input Filter Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the input glitch filter on the I/O line. |

### 32.6.9 PIO Input Filter Status Register

**Name:** PIO_IFSR
**Offset:** 0x0028
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Input Filter Status

| Value | Description |
|---|---|
| 0 | The input glitch filter is disabled on the I/O line. |
| 1 | The input glitch filter is enabled on the I/O line. |

### 32.6.10 PIO Set Output Data Register

**Name:** PIO_SODR
**Offset:** 0x0030
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Set Output Data

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Sets the data to be driven on the I/O line. |

### 32.6.11 PIO Clear Output Data Register

**Name:** PIO_CODR
**Offset:** 0x0034
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Clear Output Data

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the data to be driven on the I/O line. |

### 32.6.12 PIO Output Data Status Register

**Name:** PIO_ODSR
**Offset:** 0x0038
**Property:** Read-only
or Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Output Data Status

| Value | Description |
|---|---|
| 0 | The data to be driven on the I/O line is 0. |
| 1 | The data to be driven on the I/O line is 1. |

### 32.6.13 PIO Pin Data Status Register

**Name:** PIO_PDSR
**Offset:** 0x003C
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Output Data Status

| Value | Description |
|---|---|
| 0 | The I/O line is at level 0. |
| 1 | The I/O line is at level 1. |

### 32.6.14 PIO Interrupt Enable Register

**Name:** PIO_IER
**Offset:** 0x0040
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Input Change Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the input change interrupt on the I/O line. |

### 32.6.15 PIO Interrupt Disable Register

**Name:** PIO_IDR
**Offset:** 0x0044
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Input Change Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the input change interrupt on the I/O line. |

### 32.6.16 PIO Interrupt Mask Register

**Name:** PIO_IMR
**Offset:** 0x0048
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Input Change Interrupt Mask

| Value | Description |
|---|---|
| 0 | Input change interrupt is disabled on the I/O line. |
| 1 | Input change interrupt is enabled on the I/O line. |

### 32.6.17 PIO Interrupt Status Register

**Name:** PIO_ISR
**Offset:** 0x004C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Input Change Interrupt Status

| Value | Description |
|---|---|
| 0 | No input change has been detected on the I/O line since PIO_ISR was last read or since reset. |
| 1 | At least one input change has been detected on the I/O line since PIO_ISR was last read or since reset. |

### 32.6.18 PIO Multi-driver Enable Register

**Name:** PIO_MDER
**Offset:** 0x0050
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Multi-drive Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables multi-drive on the I/O line. |

### 32.6.19 PIO Multi-driver Disable Register

**Name:** PIO_MDDR
**Offset:** 0x0054
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Multi-drive Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables multi-drive on the I/O line. |

### 32.6.20 PIO Multi-driver Status Register

**Name:** PIO_MDSR
**Offset:** 0x0058
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Multi-drive Status

| Value | Description |
|---|---|
| 0 | The multi-drive is disabled on the I/O line. The pin is driven at high- and low-level. |
| 1 | The multi-drive is enabled on the I/O line. The pin is driven at low-level only. |

### 32.6.21 PIO Pull-Up Disable Register

**Name:** PIO_PUDR
**Offset:** 0x0060
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Pull-Up Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the pullup resistor on the I/O line. |

### 32.6.22 PIO Pull-Up Enable Register

**Name:** PIO_PUER
**Offset:** 0x0064
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Pull-Up Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the pullup resistor on the I/O line. |

### 32.6.23 PIO Pull-Up Status Register

**Name:** PIO_PUSR
**Offset:** 0x0068
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Pull-Up Status

| Value | Description |
|---|---|
| 0 | Pullup resistor is enabled on the I/O line. |
| 1 | Pullup resistor is disabled on the I/O line. |

### 32.6.24 PIO Peripheral ABCD Select Register 1

**Name:** PIO_ABCDSR1
**Offset:** 0x0070
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Peripheral Select
*If the same bit is set to '0' in PIO_ABCDSR2:*
0: Assigns the I/O line to the Peripheral A function.
1: Assigns the I/O line to the Peripheral B function.
*If the same bit is set to '1' in PIO_ABCDSR2:*
0: Assigns the I/O line to the Peripheral C function.
1: Assigns the I/O line to the Peripheral D function.

### 32.6.25 PIO Peripheral ABCD Select Register 2

**Name:** PIO_ABCDSR2
**Offset:** 0x0074
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Peripheral Select
*If the same bit is set to '0' in PIO_ABCDSR1:*
0: Assigns the I/O line to the Peripheral A function.
1: Assigns the I/O line to the Peripheral C function.
*If the same bit is set to '1' in PIO_ABCDSR1:*
0: Assigns the I/O line to the Peripheral B function.
1: Assigns the I/O line to the Peripheral D function.

### 32.6.26 PIO Input Filter Slow Clock Disable Register

**Name:** PIO_IFSCDR
**Offset:** 0x0080
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Peripheral Clock Glitch Filtering Select

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The glitch filter is able to filter glitches with a duration < $t_{peripheral\ clock}/2$. |

### 32.6.27 PIO Input Filter Slow Clock Enable Register

**Name:** PIO_IFSCER
**Offset:** 0x0084
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Slow Clock Debouncing Filtering Select

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The debouncing filter is able to filter pulses with a duration < $t_{div_slck}/2$. |

### 32.6.28 PIO Input Filter Slow Clock Status Register

**Name:** PIO_IFSCSR
**Offset:** 0x0088
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Glitch or Debouncing Filter Selection Status

| Value | Description |
|---|---|
| 0 | The glitch filter is able to filter glitches with a duration < $t_{peripheral\ clock}/2$. |
| 1 | The debouncing filter is able to filter pulses with a duration < $t_{div_slck}/2$. |

### 32.6.29 PIO Slow Clock Divider Debouncing Register

**Name:** PIO_SCDR
**Offset:** 0x008C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | DIV[13:8] | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | DIV[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 13:0 – DIV[13:0]** Slow Clock Divider Selection for Debouncing
$t_{div_slck} = ((DIV + 1) \times 2) \times t_{slck}$

### 32.6.30 PIO Pad Pull-Down Disable Register

**Name:** PIO_PPDDR
**Offset:** 0x0090
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Pull-Down Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the pull-down resistor on the I/O line. |

### 32.6.31 PIO Pad Pull-Down Enable Register

**Name:** PIO_PPDER
**Offset:** 0x0094
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Pull-Down Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the pull-down resistor on the I/O line. |

### 32.6.32 PIO Pad Pull-Down Status Register

**Name:** PIO_PPDSR
**Offset:** 0x0098
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Pull-Down Status

| Value | Description |
|---|---|
| 0 | Pull-down resistor is enabled on the I/O line. |
| 1 | Pull-down resistor is disabled on the I/O line. |

### 32.6.33 PIO Output Write Enable Register

**Name:** PIO_OWER
**Offset:** 0x00A0
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Output Write Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables writing PIO_ODSR for the I/O line. |

### 32.6.34 PIO Output Write Disable Register

**Name:** PIO_OWDR
**Offset:** 0x00A4
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Output Write Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables writing PIO_ODSR for the I/O line. |

### 32.6.35 PIO Output Write Status Register

**Name:** PIO_OWSR
**Offset:** 0x00A8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Output Write Status

| Value | Description |
|---|---|
| 0 | Writing PIO_ODSR does not affect the I/O line. |
| 1 | Writing PIO_ODSR affects the I/O line. |

### 32.6.36 PIO Additional Interrupt Modes Enable Register

**Name:** PIO_AIMER
**Offset:** 0x00B0
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Additional Interrupt Modes Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The interrupt source is the event described in PIO_ELSR and PIO_FRLHSR. |

### 32.6.37 PIO Additional Interrupt Modes Disable Register

**Name:** PIO_AIMDR
**Offset:** 0x00B4
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Additional Interrupt Modes Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Interrupt mode is set to the default Interrupt mode (Both-edge Detection). |

### 32.6.38 PIO Additional Interrupt Modes Mask Register

**Name:** PIO_AIMMR
**Offset:** 0x00B8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO I/O Line Index
Selects the I/O event type triggering an interrupt.

| Value | Description |
|---|---|
| 0 | The interrupt source is a both-edge detection event. |
| 1 | The interrupt source is described by the registers PIO_ELSR and PIO_FRLHSR. |

### 32.6.39 PIO Edge Select Register

**Name:** PIO_ESR
**Offset:** 0x00C0
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Edge Interrupt Selection

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The interrupt source is an edge-detection event. |

### 32.6.40 PIO Level Select Register

**Name:** PIO_LSR
**Offset:** 0x00C4
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Level Interrupt Selection

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The interrupt source is a level-detection event. |

### 32.6.41 PIO Edge/Level Status Register

**Name:** PIO_ELSR
**Offset:** 0x00C8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Edge/Level Interrupt Source Selection

| Value | Description |
|---|---|
| 0 | The interrupt source is an edge-detection event. |
| 1 | The interrupt source is a level-detection event. |

### 32.6.42 PIO Falling Edge/Low-Level Select Register

**Name:** PIO_FELLSR
**Offset:** 0x00D0
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Falling Edge/Low-Level Interrupt Selection

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The interrupt source is set to a falling edge detection or low-level detection event, depending on PIO_ELSR. |

### 32.6.43 PIO Rising Edge/High-Level Select Register

**Name:** PIO_REHLSR
**Offset:** 0x00D4
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Rising Edge/High-Level Interrupt Selection

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The interrupt source is set to a rising edge detection or high-level detection event, depending on PIO_ELSR. |

### 32.6.44 PIO Fall/Rise - Low/High Status Register

**Name:** PIO_FRLHSR
**Offset:** 0x00D8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Edge/Level Interrupt Source Selection

| Value | Description |
|---|---|
| 0 | The interrupt source is a falling edge detection (if PIO_ELSR = 0) or low-level detection event (if PIO_ELSR = 1). |
| 1 | The interrupt source is a rising edge detection (if PIO_ELSR = 0) or high-level detection event (if PIO_ELSR = 1). |

### 32.6.45 PIO Lock Status Register

**Name:** PIO_LOCKSR
**Offset:** 0x00E0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P31 | P30 | P29 | P28 | P27 | P26 | P25 | P24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P23 | P22 | P21 | P20 | P19 | P18 | P17 | P16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P15 | P14 | P13 | P12 | P11 | P10 | P9 | P8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P7 | P6 | P5 | P4 | P3 | P2 | P1 | P0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – P**
PIO Lock Status

| Value | Description |
|---|---|
| 0 | The I/O line is not locked. |
| 1 | The I/O line is locked. |

### 32.6.46 PIO Write Protection Mode Register

**Name:** PIO_WPMR
**Offset:** 0x00E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x50494F | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
Refer to "Register Write Protection" for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x50494F ("PIO" in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x50494F ("PIO" in ASCII). |

### 32.6.47 PIO Write Protection Status Register

**Name:** PIO_WPSR
**Offset:** 0x00E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the PIO_WPSR. |
| 1 | A write protection violation has occurred since the last read of the PIO_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

### 32.6.48 PIO Schmitt Trigger Register

**Name:** PIO_SCHMITT
**Offset:** 0x0100
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SCHMITT31 | SCHMITT30 | SCHMITT29 | SCHMITT28 | SCHMITT27 | SCHMITT26 | SCHMITT25 | SCHMITT24 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SCHMITT23 | SCHMITT22 | SCHMITT21 | SCHMITT20 | SCHMITT19 | SCHMITT18 | SCHMITT17 | SCHMITT16 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCHMITT15 | SCHMITT14 | SCHMITT13 | SCHMITT12 | SCHMITT11 | SCHMITT10 | SCHMITT9 | SCHMITT8 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCHMITT7 | SCHMITT6 | SCHMITT5 | SCHMITT4 | SCHMITT3 | SCHMITT2 | SCHMITT1 | SCHMITT0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – SCHMITT** PIO Schmitt Trigger Control

| Value | Description |
|---|---|
| 0 | Schmitt trigger is enabled. |
| 1 | Schmitt trigger is disabled. |

### 32.6.49 PIO I/O Drive Register 1

**Name:** PIO_DRIVER1
**Offset:** 0x0118
**Property:** Read/Write

Register Reset value: 0x000000000xAAAAAAAA

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINE31 | LINE30 | LINE29 | LINE28 | LINE27 | LINE26 | LINE25 | LINE24 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LINE23 | LINE22 | LINE21 | LINE20 | LINE19 | LINE18 | LINE17 | LINE16 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINE15 | LINE14 | LINE13 | LINE12 | LINE11 | LINE10 | LINE9 | LINE8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LINE7 | LINE6 | LINE5 | LINE4 | LINE3 | LINE2 | LINE1 | LINE0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – LINE** Drive of PIO Line

| Value | Name | Description |
|---|---|---|
| 0 | LOW_DRIVE | Lowest drive |
| 1 | HIGH_DRIVE | Highest drive |

### 32.6.50 PIO Parallel Capture Mode Register

**Name:** PIO_PCMR
**Offset:** 0x0150
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FRSTS | HALFS | ALWYS | |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | DSIZE[1:0] | | | | | PCEN |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | | | 0 |

**Bit 11 – FRSTS** Parallel Capture Mode First Sample
This bit is useful only if the HALFS bit is set to 1. If data are numbered in the order that they are received with an index from 0 to n:

| Value | Description |
|---|---|
| 0 | Only data with an even index are sampled. |
| 1 | Only data with an odd index are sampled. |

**Bit 10 – HALFS** Parallel Capture Mode Half Sampling
Independently from the ALWYS bit:

| Value | Description |
|---|---|
| 0 | The Parallel Capture mode samples all the data. |
| 1 | The Parallel Capture mode samples the data only every other time. |

**Bit 9 – ALWYS** Parallel Capture Mode Always Sampling

| Value | Description |
|---|---|
| 0 | The Parallel Capture mode samples the data when both data enables are active. |
| 1 | The Parallel Capture mode samples the data whatever the data enables are. |

**Bits 5:4 – DSIZE[1:0]** Parallel Capture Mode Data Size

| Value | Name | Description |
|---|---|---|
| 0 | BYTE | The reception data in the PIO_PCRHR is a byte (8-bit) |
| 1 | HALF-WORD | The reception data in the PIO_PCRHR is a half-word (16-bit) |
| 2 | WORD | The reception data in the PIO_PCRHR is a word (32-bit) |
| 3 | Reserved | Reserved |

**Bit 0 – PCEN** Parallel Capture Mode Enable

| Value | Description |
|---|---|
| 0 | The Parallel Capture mode is disabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | The Parallel Capture mode is enabled. |

### 32.6.51 PIO Parallel Capture Interrupt Enable Register

**Name:** PIO_PCIER
**Offset:** 0x0154
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | RXBUFF | ENDRX | OVRE | DRDY |
| Reset | | | | | | | | |

**Bit 3 – RXBUFF** Reception Buffer Full Interrupt Enable

**Bit 2 – ENDRX** End of Reception Transfer Interrupt Enable

**Bit 1 – OVRE** Parallel Capture Mode Overrun Error Interrupt Enable

**Bit 0 – DRDY** Parallel Capture Mode Data Ready Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 32.6.52 PIO Parallel Capture Interrupt Disable Register

**Name:** PIO_PCIDR
**Offset:** 0x0158
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | RXBUFF | ENDRX | OVRE | DRDY |
| Reset | | | | | | | | |

**Bit 3 – RXBUFF** Reception Buffer Full Interrupt Disable

**Bit 2 – ENDRX** End of Reception Transfer Interrupt Disable

**Bit 1 – OVRE** Parallel Capture Mode Overrun Error Interrupt Disable

**Bit 0 – DRDY** Parallel Capture Mode Data Ready Interrupt Disable

### 32.6.53 PIO Parallel Capture Interrupt Mask Register

**Name:** PIO_PCIMR
**Offset:** 0x015C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: Corresponding interrupt is not enabled.

1: Corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXBUFF | ENDRX | OVRE | DRDY |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – RXBUFF** Reception Buffer Full Interrupt Mask

**Bit 2 – ENDRX** End of Reception Transfer Interrupt Mask

**Bit 1 – OVRE** Parallel Capture Mode Overrun Error Interrupt Mask

**Bit 0 – DRDY** Parallel Capture Mode Data Ready Interrupt Mask

### 32.6.54 PIO Parallel Capture Interrupt Status Register

**Name:** PIO_PCISR
**Offset:** 0x0160
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OVRE | DRDY |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – OVRE** Parallel Capture Mode Overrun Error
The OVRE flag is automatically reset when this register is read or when the Parallel Capture mode is disabled.

| Value | Description |
|---|---|
| 0 | No overrun error occurred since the last read of this register. |
| 1 | At least one overrun error occurred since the last read of this register. |

**Bit 0 – DRDY** Parallel Capture Mode Data Ready
The DRDY flag is automatically reset when PIO_PCRHR is read or when the Parallel Capture mode is disabled.

| Value | Description |
|---|---|
| 0 | No new data is ready to be read since the last read of PIO_PCRHR. |
| 1 | A new data is ready to be read since the last read of PIO_PCRHR. |

### 32.6.55 PIO Parallel Capture Reception Holding Register

**Name:** PIO_PCRHR
**Offset:** 0x0164
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RDATA[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RDATA[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RDATA[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDATA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RDATA[31:0]** Parallel Capture Mode Reception Data
If DSIZE = 0 in PIO_PCMR, only the 8 LSBs of RDATA are useful.
If DSIZE = 1 in PIO_PCMR, only the 16 LSBs of RDATA are useful.

# 33. External Bus Interface (EBI)

## 33.1 Description

The External Bus Interface (EBI) is designed to ensure the successful data transfer between several external devices and the embedded Memory Controller of an ARM-based device.

The Static Memory and SDRAM Controllers are all featured external Memory Controllers on the EBI. These external Memory Controllers are capable of handling several types of external memory and peripheral devices, such as SRAM, PROM, EPROM, EEPROM, Flash and SDR-SDRAM. The EBI operates with a 3.3V power supply (VDDIO).

The EBI also supports the NAND Flash protocols via integrated circuitry that greatly reduces the requirements for external components. Furthermore, the EBI handles data transfers with up to six external devices, each assigned to six address spaces defined by the embedded Memory Controller. Data transfers are performed through a 16-bit or 32-bit data bus, an address bus of up to 24 bits, up to four chip select lines (NCS[3:0]) and several control pins that are generally multiplexed between the different external Memory Controllers.

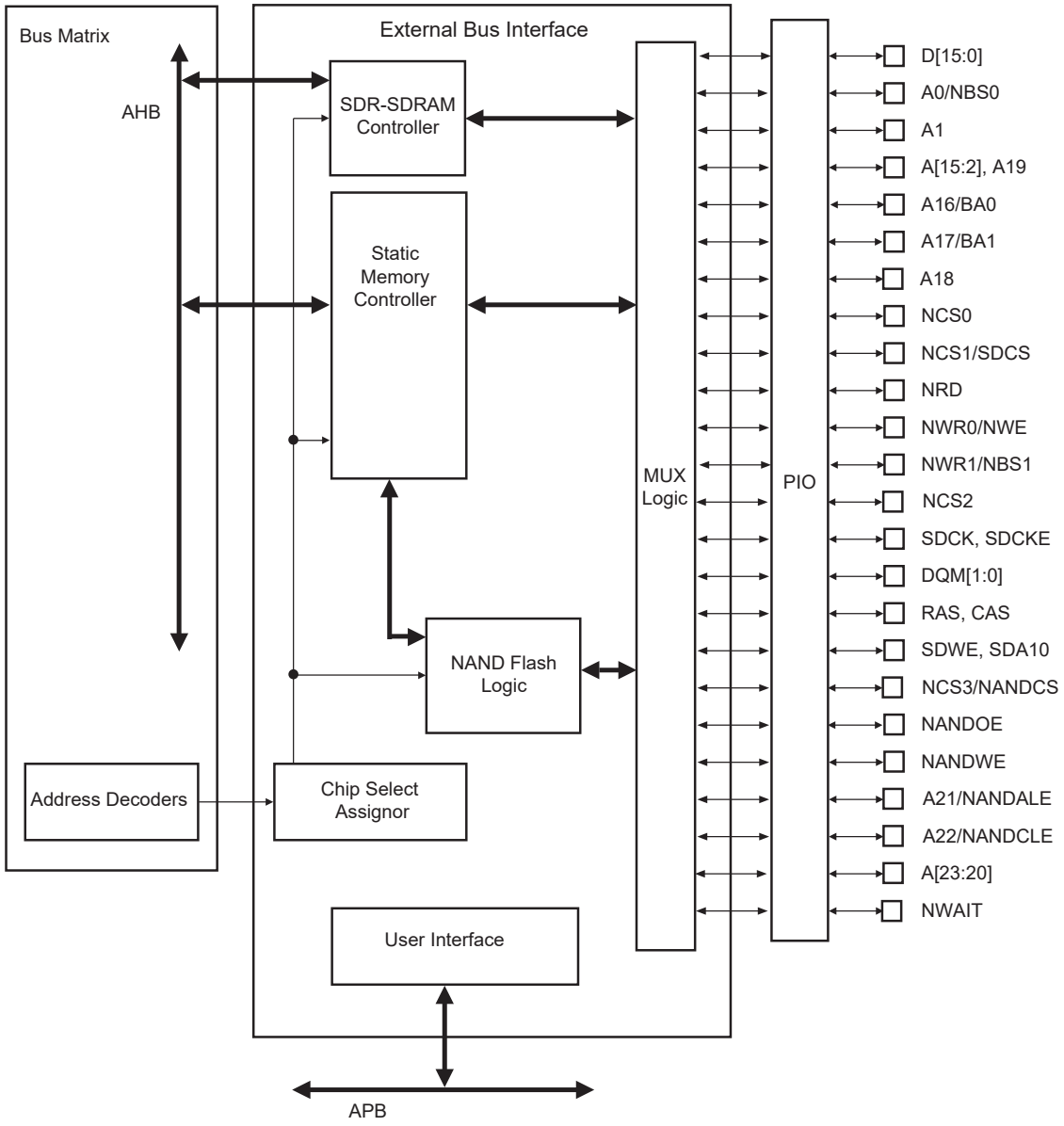
## 33.2 Embedded Characteristics

- Integrates two External Memory Controllers
  - Static Memory Controller
  - SDR-SDRAM Controller
- Integrates NAND Flash Logic
- Up to 24-bit Address Bus (up to 16 Mbytes linear per chip select)
- Up to four Chip Selects, Configurable Assignment
  - Static Memory Controller on NCS0, NCS1, NCS2, NCS3
  - SDR-SDRAM Controller (SDCS) or Static Memory Controller on NCS1
  - NAND Flash support on NCS0, NCS1, NSCS2 and NCS3

© 2021 Microchip Technology Inc.
and its subsidiaries

## 33.3 EBI Block Diagram

**Figure 33-1. Organization of the External Bus Interface**




## 33.4 I/O Lines Description

**Table 33-1. EBI I/O Lines Description**

| Name | Function | Type | Active Level |
|---|---|---|---|
| EBI | | | |
| D0–D15 | Data Bus | I/O | |
| A0–A23 | Address Bus | Output | |
| NWAIT | External Wait Signal | Input | Low |

| .....continued | | | |
|---|---|---|---|
| **Name** | **Function** | **Type** | **Active Level** |
| SMC | | | |
| NCS0–EBI_NCS3 | Chip Select Lines | Output | Low |
| NWR0–NWR1 | Write Signals | Output | Low |
| NRD | Read Signal | Output | Low |
| NWE | Write Enable | Output | Low |
| NBS0–NBS1 | Byte Mask Signals | Output | Low |
| EBI for NAND Flash Support | | | |
| NANDCS | NAND Flash Chip Select Line | Output | Low |
| NANDOE | NAND Flash Output Enable | Output | Low |
| NANDWE | NAND Flash Write Enable | Output | Low |
| SDRAM Controller | | | |
| SDCK (see **Note**) | SDR-SDRAM Clock | Output | |
| SDCKE | SDR-SDRAM Clock Enable | Output | High |
| SDCS | SDR-SDRAM Controller Chip Select Line | Output | Low |
| BA0–1 | Bank Select | Output | |
| SDWE | SDR-SDRAM Write Enable | Output | Low |
| RAS - CAS | Row and Column Signal | Output | Low |
| SDA10 | SDRAM Address 10 Line | Output | |

**Note:** SDCK is the MCK clock for EBI, SDRAM Controller and SMC interfaces.

The connection of some signals through the MUX logic is not direct and depends on the Memory Controller in use at the moment.

The following table details the connections between the two Memory Controllers and the EBI pins.

**Table 33-2. EBI Pins and Memory Controllers I/O Lines Connections**

| EBIx Pins | SDRAM I/O Lines | SMC I/O Lines |
|---|---|---|
| NWR1/NBS1 | NBS1 | NWR1 |
| A0/NBS0 | NBS0 | SMC_A0 |
| A1 | Not Supported | SMC_A1 |
| A[11:2] | SDRAMC_A[9:0] | SMC_A[11:2] |
| SDA10 | SDRAMC_A10 | Not Supported |
| A12 | Not Supported | SMC_A12 |
| A[15:13] | SDRAMC_A[13:11] | SMC_A[15:13] |
| A[25:16] | Not Supported | SMC_A[25:16] |
| D[15:0] | D[15:0] | D[15:0] |

© 2021 Microchip Technology Inc.
and its subsidiaries

## 33.5 Application Example

### 33.5.1 Hardware Interface

The following table details the connections to be applied between the EBI pins and the external devices for each Memory Controller.

**Table 33-3. EBI Pins and External Static Device Connections**

| Signals: EBI_ | Pins of the Interfaced Device | | |
|---|---|---|---|
| | 8-bit Static Device | 2 x 8-bit Static Devices | 16-bit Static Device |
| Controller | SMC | | |
| D0–D7 | D0–D7 | D0–D7 | D0–D7 |
| D8–D15 | – | D8–D15 | D8–D15 |
| A0/NBS0 | A0 | – | NLB |
| A1 | A1 | A0 | A0 |
| A2–A23 | A[2:23] | A[1:22] | A[1:22] |
| NCS0 | CS | CS | CS |
| NCS1/DDRSDCS | CS | CS | CS |
| NCS2 | CS | CS | CS |
| NCS3/NANDCS | CS | CS | CS |
| NRD | OE | OE | OE |
| NWR0/NWE | WE | WE (see **Note**) | WE |
| NWR1/NBS1 | – | WE (see **Note**) | NUB |

**Note:** NWR1 enables upper byte writes. NWR0 enables lower byte writes.

**Table 33-4. EBI Pins and External Device Connections**

| Signals: EBI_ | Power supply | Pins of the Interfaced Device | |
|---|---|---|---|
| | | SDR/LPSDR | NAND Flash |
| Controller | | SDRAMC | NFC |
| D0–D15 | VDDIO | D0–D15 | D0–D15 |
| A0/NBS0 | VDDIO | DQM0 | – |
| A1 | VDDIO | – | – |
| A2–A10 | VDDIO | A[0:8] | – |
| A11 | VDDIO | A9 | – |
| SDA10 | VDDIO | A10 | – |
| A12 | VDDIO | – | – |
| A13–A14 | VDDIO | A[11:12] | – |
| A15 | VDDIO | A13 | – |
| A16/BA0 | VDDIO | BA0 | – |

© 2021 Microchip Technology Inc. and its subsidiaries

| .....continued | | | |
|---|---|---|---|
| Signals: EBI_ | Power supply | Pins of the Interfaced Device | |
| | | SDR/LPSDR | NAND Flash |
| Controller | | SDRAMC | NFC |
| A17/BA1 | VDDIO | BA1 | – |
| A18 | VDDIO | – | – |
| A19 | VDDIO | – | – |
| A20 | VDDIO | – | – |
| A21/NANDALE | VDDIO | – | ALE |
| A22/NANDCLE | VDDIO | – | CLE |
| A23 | VDDIO | – | – |
| NCS0 | VDDIO | – | – |
| NCS1/SDCS | VDDIO | SDCS | – |
| NCS2 | VDDIO | – | – |
| NCS3/NANDCS | VDDIO | – | CE |
| NANDOE | VDDIO | – | OE |
| NANDWE | VDDIO | – | WE |
| NRD | VDDIO | – | – |
| NWR0/NWE | VDDIO | – | – |
| NWR1/NBS1 | VDDIO | DQM1 | – |
| SDCK | VDDIO | CK | – |
| SDCKE | VDDIO | CKE | – |
| RAS | VDDIO | RAS | – |
| CAS | VDDIO | CAS | – |
| SDWE | VDDIO | WE | – |
| Pxx | VDDIO | – | CE |
| Pxx | VDDIO | – | RDY |

### 33.5.2 Product Dependencies

#### 33.5.2.1 I/O Lines

The pins used for interfacing the External Bus Interface may be multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the External Bus Interface pins to their peripheral function. If I/O lines of the External Bus Interface are not used by the application, they can be used for other purposes by the PIO Controller.

### 33.5.3 Functional Description

The EBI transfers data between the internal AHB Bus (handled by the Bus Matrix) and the external memories or peripheral devices. It controls the waveforms and the parameters of the external address, data and control buses and is composed of the following elements:

- Static Memory Controller (SMC)
- SDR-SDRAM Controller (SDRC)
- A chip select assignment feature that assigns an AHB address space to the external devices

- A multiplex controller circuit that shares the pins between the different Memory Controllers
- Programmable NAND Flash support logic

#### 33.5.3.1 Bus Multiplexing

The EBI offers a complete set of control signals that share the 16-bit data lines, the address lines of up to 24 bits and the control signals through a multiplex logic operating in function of the memory area requests.

Multiplexing is specifically organized in order to guarantee the maintenance of the address and output control lines at a stable state while no external access is being performed. Multiplexing is also designed to respect the data float times defined in the Memory Controllers. Furthermore, refresh cycles of the SDR-SDRAM are executed independently by the SDR Controller without delaying the other external Memory Controller accesses.

#### 33.5.3.2 Static Memory Controller

For information on the Static Memory Controller, refer to 35. Static Memory Controller (SMC)

#### 33.5.3.3 SDRAM Controller

For information on the SDR Controller, refer to 34. SDRAM Controller (SDRAMC).

#### 33.5.3.4 NAND Flash Support

External Bus Interfaces integrate circuitry that interfaces to NAND Flash devices.

To ensure that the processor preserves transaction order and thus the correct NAND Flash behavior, the NAND Flash address space is to be declared in the Memory Protection Unit (MPU) as “Device” or “Strongly-ordered” memory. Refer to the ARM Cortex-M7 Technical Reference Manual (ARM DDI 0489) available on www.arm.com.

**External Bus Interface**

The NAND Flash Chip Select (NANDCS) is driven by the Static Memory Controller on the NCS0, NCS1, NCS2 or NCS3 address space depending on value of SMC_SMCSx bits. For example, programming the SMC_NFC3 field in the CCFG_SMCNFCS Register in the Chip Configuration User Interface to the appropriate value enables the NAND Flash logic. For details on this register, refer to 19. Bus Matrix (MATRIX). Access to an external NAND Flash device is then made by accessing the address space reserved to NCS3 (i.e., between 0x6300 0000 and 0x6FFF FFFF).

The NAND Flash logic drives the read and write command signals of the SMC on the NANDOE and NANDWE signals when the required SMC_NFCSx signal is active. NANDOE and NANDWE are invalidated as soon as the transfer address fails to lie in the selected NCSx address space. For details on these waveforms, refer to 35. Static Memory Controller (SMC).

**NAND Flash Signals**

The address latch enable and command latch enable signals on the NAND Flash device are driven by address bits A22 and A21 of the EBI address bus. The command, address or data words on the data bus of the NAND Flash device are distinguished by using their address within the NCSx address space. The chip enable (CE) signal of the device and the ready/busy (R/B) signals are connected to PIO lines. The CE signal then remains asserted even when NCSx is not selected, preventing the device from returning to standby mode.

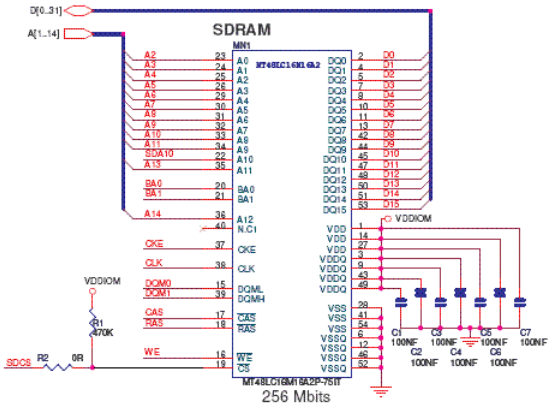
### 33.5.4 Implementation Examples

The following hardware configurations are given for illustration only. The user should refer to the memory manufacturer web site to check current device availability.

© 2021 Microchip Technology Inc. and its subsidiaries

### 33.5.4.1 16-bit SDRAM on NCS1

**Figure 33-2. Hardware Configuration**




Software Configuration

The following configuration has to be performed:

- Enable the SDRAM support by setting the bit SDRAMEN field in the CCFG_SMCNFCS Register in the Bus Matrix.
- Initialize the SDRAM Controller depending on the SDRAM device and system bus frequency.

The Data Bus Width is to be programmed to 16 bits.

The SDRAM initialization sequence is described in 34.5.1 SDRAM Device Initialization.

# 34. SDRAM Controller (SDRAMC)

## 34.1 Description

The SDRAM Controller (SDRAMC) extends the memory capabilities of a chip by providing the interface to external 16-bit DRAM devices. The page size supports ranges from 2048 to 8192 and the number of columns from 256 to 2048. It supports byte (8-bit), half-word (16-bit) and word (32-bit) accesses.

The SDRAMC supports a read or write burst length of one location. It keeps track of the active row in each bank, thus maximizing SDRAM performance, for example, the application may be placed in one bank and data in the other banks. For optimized performance, it is advisable to avoid accessing different rows in the same bank.

The SDRAMC supports a CAS latency of 2 or 3 and optimizes the read access depending on the frequency.

The available different modes, such as Self-refresh, Powerdown and Deep Powerdown modes, minimizes the power consumption on the SDRAM device.

## 34.2 Embedded Characteristics

- Numerous Configurations Supported
  - 2K, 4K, 8K row address memory parts
  - SDRAM with two or four internal banks
  - SDRAM with 16-bit data path
- Programming Facilities
  - Word, half-word, byte access
  - Automatic Page break when memory boundary has been reached
  - Multibank ping-pong access
  - Timing parameters specified by software
  - Automatic refresh operation, refresh rate is programmable
  - Automatic update of DS, TCR and PASR parameters (mobile SDRAM devices)
- Energy-Saving Capabilities
  - Self-refresh, Powerdown and Deep Power modes Supported
  - Supports mobile SDRAM devices
- Error Detection
  - Refresh error interrupt
- SDRAM Power-up Initialization by Software
- CAS Latency of 2, 3 Supported
- Auto Precharge Command Not Used
- Zero Wait State Scrambling/Unscrambling Function with User Key

## 34.3 Signal Description

**Table 34-1. Signal Description**

| Name | Description | Type | Active Level |
|---|---|---|---|
| SDCK | SDRAM Clock | Output | – |
| SDCKE | SDRAM Clock Enable | Output | High |
| SDCS | SDRAMC Chip Select | Output | Low |
| BA[1:0] | Bank Select Signals | Output | – |

| .....continued | | | |
|---|---|---|---|
| **Name** | **Description** | **Type** | **Active Level** |
| RAS | Row Signal | Output | Low |
| CAS | Column Signal | Output | Low |
| SDWE | SDRAM Write Enable | Output | Low |
| NBS[1:0] | Data Mask Enable Signals | Output | Low |
| SDRAMC_A[12:0] | Address Bus | Output | – |
| D[15:0] | Data Bus | I/O | – |

## 34.4 Software Interface/SDRAM Organization, Address Mapping

The SDRAM address space is organized into banks, rows, and columns. The SDRAMC allows mapping different memory types according to the values set in the Configuration register (SDRAMC_CR).

The SDRAMC makes the SDRAM device access protocol transparent to the user. The following tables illustrate the SDRAM device memory mapping seen by the user in correlation with the device structure. Various configurations are illustrated.

### 34.4.1 SDRAM Address Mapping for 16-bit Memory Data Bus Width

**Table 34-2. SDRAM Configuration Mapping: 2K Rows, 256/512/1024/2048 Columns**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | | Bk[1:0] | | Row[10:0] | | | | | | | | | | | Column[7:0] | | | | | | | | M0 |
| | | | | | Bk[1:0] | | Row[10:0] | | | | | | | | | | | | Column[8:0] | | | | | | | | M0 |
| | | | | Bk[1:0] | | Row[10:0] | | | | | | | | | | | Column[9:0] | | | | | | | | | | M0 |
| | | | Bk[1:0] | | Row[10:0] | | | | | | | | | | | Column[10:0] | | | | | | | | | | | M0 |

**Note:** M0 is the byte address inside a 16-bit half-word and Bk[1] = BA1, Bk[0] = BA0.

**Table 34-3. SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048 Columns**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | Column[7:0] | | | | | | | | | | M0 |
| | | | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | | Column[8:0] | | | | | | | | | M0 |
| | | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M0 |
| | | Bk[1:0] | | Row[11:0] | | | | | | | | | | | | Column[10:0] | | | | | | | | | | | M0 |

**Note:** M0 is the byte address inside a 16-bit half-word and Bk[1] = BA1, Bk[0] = BA0.

**Table 34-4. SDRAM Configuration Mapping: 8K Rows, 256/512/1024/2048 Columns**

| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| | | | | Bk[1:0] | | Row[12:0] | | | | | | | | | | | | | Column[7:0] | | | | | | | | | M0 |
| | | | Bk[1:0] | | Row[12:0] | | | | | | | | | | | | | Column[8:0] | | | | | | | | | M0 |

| .....continued | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CPU Address Line | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | Bk[1:0] | | Row[12:0] | | | | | | | | | | | | | Column[9:0] | | | | | | | | | | M0 |
| | Bk[1:0] | | | Row[12:0] | | | | | | | | | | | | Column[10:0] | | | | | | | | | | | M0 |

**Note:** M0 is the byte address inside a 16-bit half-word and Bk[1] = BA1, Bk[0] = BA0.

## 34.5 Product Dependencies

### 34.5.1 SDRAM Device Initialization

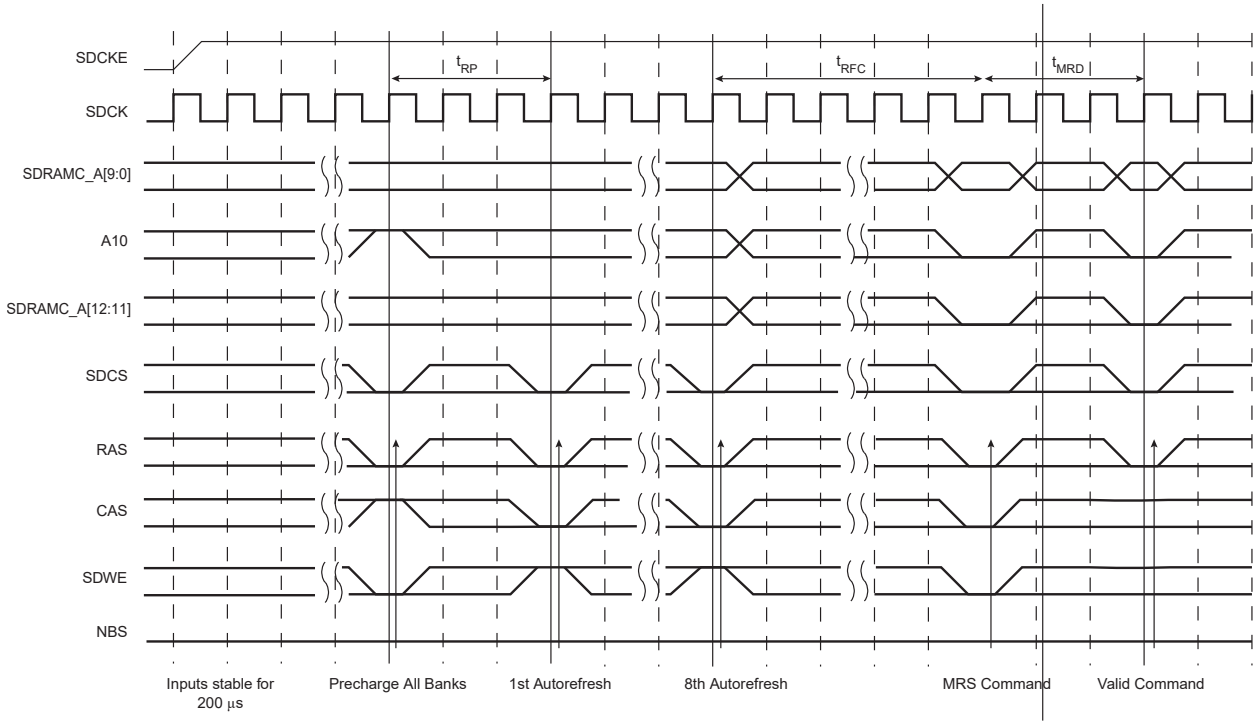
The initialization sequence is generated by software. The sequence to initialize SDRAM devices is the following:

1. Set the SDRAM features in the SDRAMC_CR: asynchronous timings (TRC, TRAS, etc.), number of columns, number of rows, CAS latency and data bus width. Set UNAL bit in SDRAMC_CFR1.
2. For mobile SDRAM, configure temperature-compensated self-refresh (TCSR), drive strength (DS) and partial array self-refresh (PASR) in the Low Power register (SDRAMC_LPR).
3. Select the SDRAM memory device type in the Memory Device register (SDRAMC_MDR).
4. A pause of at least 200 µs must be observed before a signal toggle.
5. A NOP command is issued to the SDRAM devices. The application must write a 1 to the MODE field in the Mode register (SDRAMC_MR) (see **Note**). Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM address.
6. An All Banks Precharge command is issued to the SDRAM. The application must write a 2 to the MODE field in the SDRAMC_MR. Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM address.
7. Eight autorefresh (CBR) cycles are provided. The application must set the MODE field to 4 in the SDRAMC_MR. Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any SDRAM location eight times.
8. A Mode Register set (MRS) cycle is issued to program the parameters of the SDRAM, in particular CAS latency and burst length. The application must write a 3 to the MODE field in the SDRAMC_MR. Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the SDRAM. The write address must be chosen so that BA[1:0] are set to 0. For example, with a 16-bit 128 MB SDRAM (12 rows, 9 columns, 4 banks) bank address, the SDRAM write access should be done at the address 0x70000000.
9. For mobile SDRAM initialization, an Extended Mode Register set (EMRS) cycle is issued to program the SDRAM parameters (TCSR, PASR, DS). The application must set the MODE field to 5 in the SDRAMC_MR. Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the SDRAM. The write address must be chosen so that BA[1] or BA[0] are set to 1. For example, with a 16-bit 128 MB SDRAM (12 rows, 9 columns, 4 banks) bank address, the SDRAM write access should be done at address 0x70800000 or 0x70400000.
10. The application must go into Normal mode. Configure MODE to 0 in the SDRAMC_MR. Read the SDRAMC_MR and add a memory barrier assembler instruction just after the read. Perform a write access at any location in the SDRAM.
11. Write the refresh rate into the COUNT field in the Refresh Timer register (SDRAMC_TR). (Refresh rate = delay between refresh cycles). The SDRAM device requires a refresh every 15.625 µs or 7.81 µs. With a 100 MHz frequency, the Refresh Timer register must be set with the value 1562 (15.625 µs x 100 MHz) or 781 (7.81 µs x 100 MHz).

After initialization, the SDRAM devices are fully functional.

**Note:** The instructions stated in Step 5 of the initialization process must be respected to make sure the subsequent commands issued by the SDRAMC are taken into account.

**Figure 34-1. SDRAM Device Initialization Sequence**




### 34.5.2 I/O Lines

The pins used for interfacing the SDRAMC may be multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the SDRAMC pins to their peripheral function. If I/O lines of the SDRAMC are not used by the application, they can be used for other purposes by the PIO Controller.

### 34.5.3 Power Management

The SDRAMC may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SDRAMC clock.

The SDRAM clock on pin SDCK is output as soon as the first access to the SDRAM is made during the initialization phase. To stop the SDRAM clock signal, the SDRAMC_LPR must be programmed with the self-refresh command.

### 34.5.4 Interrupt Sources

The SDRAMC interrupt (Refresh Error notification) is connected to the memory controller. This interrupt may be ORed with other system peripheral interrupt lines and is finally provided as the system interrupt source (Source 1) to the interrupt controller.

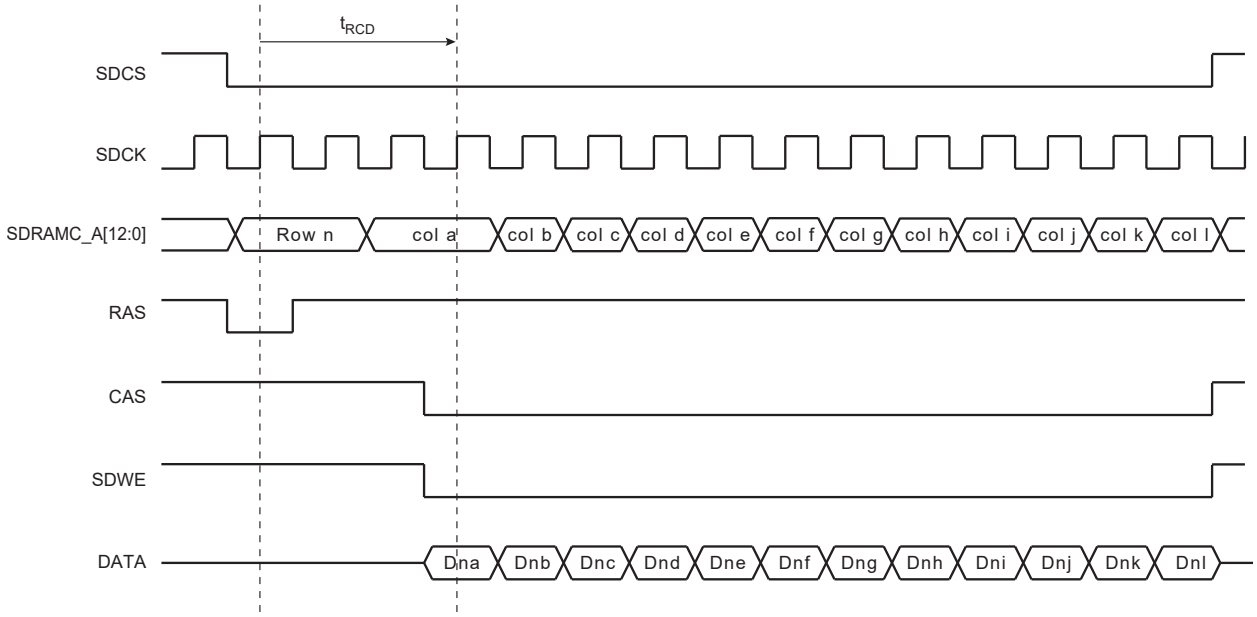
Using the SDRAMC interrupt requires the interrupt controller to be programmed first.

## 34.6 Functional Description

### 34.6.1 SDRAM Controller Write Cycle

The SDRAMC allows burst access or single access. In both cases, the SDRAMC keeps track of the active row in each bank, thus maximizing performance. To initiate a burst access, the SDRAMC uses the transfer type signal provided by the master requesting the access. If the next access is a sequential write access, writing to the SDRAM device is carried out. If the next access is a write-sequential access, but the current access is to a boundary page, or if the next access is in another row, then the SDRAMC generates a precharge command, activates the new row and initiates a write command. To comply with SDRAM timing parameters, additional clock cycles are inserted between precharge and active commands ($t_{RP}$), and between active and write commands ($t_{RCD}$). For definition of these timing parameters, refer to the SDRAMC Configuration Register. Refer to the following figure.

**Figure 34-2. Write Burst SDRAM Access**



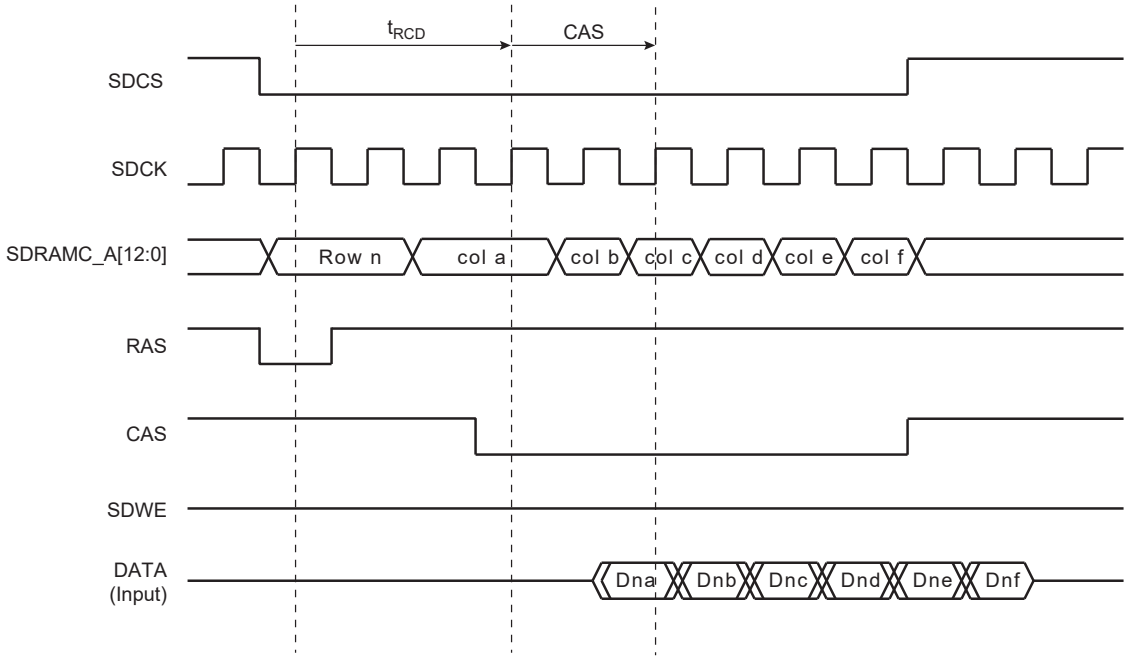

### 34.6.2 SDRAM Controller Read Cycle

The SDRAMC allows burst access, incremental burst of unspecified length or single access. In all cases, the SDRAMC keeps track of the active row in each bank, thus maximizing performance of the SDRAM. If row and bank addresses do not match the previous row/bank address, then the SDRAMC automatically generates a precharge command, activates the new row and starts the read command. To comply with the SDRAM timing parameters, additional clock cycles on SDCK are inserted between precharge and active commands ($t_{RP}$), and between active and read commands ($t_{RCD}$). These two parameters are set in the SDRAMC_CR. After a read command, additional wait states are generated to comply with the CAS latency ( 2 or 3 clock delays specified in the SDRAMC_CR).

For a single access or an incremented burst of unspecified length, the SDRAMC anticipates the next access. While the last value of the column is returned by the SDRAMC on the bus, the SDRAMC anticipates the read to the next column and thus anticipates the CAS latency. This reduces the effect of the CAS latency on the internal bus.

For burst access of specified length (4, 8, 16 words), access is not anticipated. This case leads to the best performance. If the burst is broken (border, Busy mode, etc.), the next access is handled as an incrementing burst of unspecified length.

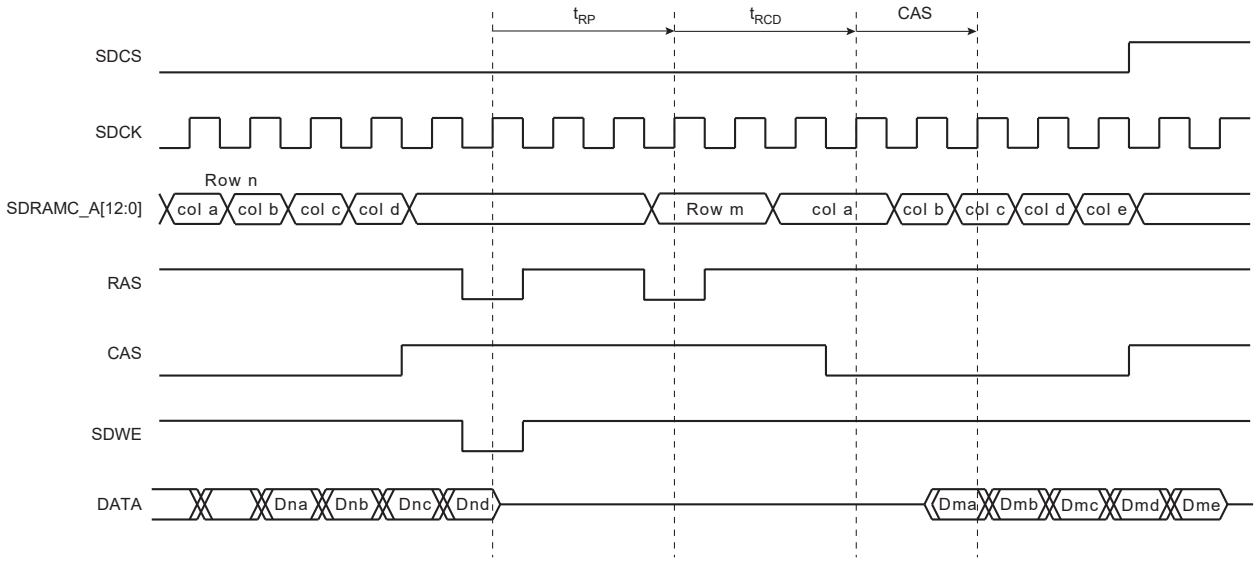
**Figure 34-3. Read Burst SDRAM Access**




### 34.6.3 Border Management

When the memory row boundary has been reached, an automatic page break is inserted. In this case, the SDRAMC generates a precharge command, activates the new row and initiates a read or write command. To comply with SDRAM timing parameters, an additional clock cycle is inserted between the precharge and the active command ($t_{RP}$) and between the active and the read command ($t_{RCD}$). Refer to the following figure.

**Figure 34-4. Read Burst with Boundary Row Access**



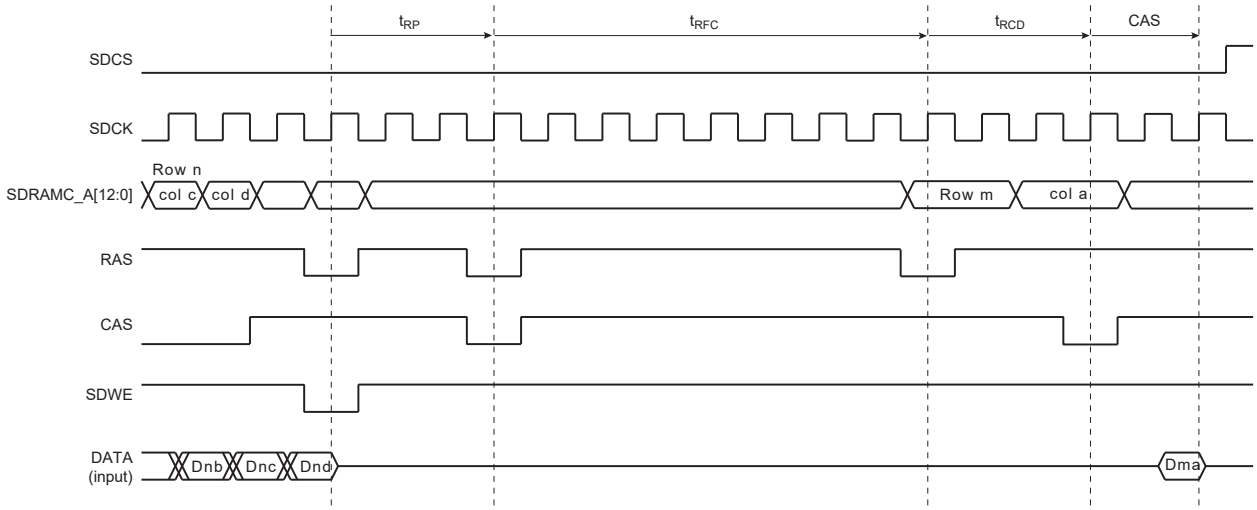

### 34.6.4 SDRAM Controller Refresh Cycles

An autorefresh command is used to refresh the SDRAM device. Refresh addresses are generated internally by the SDRAM device and incremented after each autorefresh automatically. The SDRAMC generates these autorefresh commands periodically. An internal timer is loaded with the value in SDRAMC_TR that indicates the number of clock cycles between refresh cycles.

A refresh error interrupt is generated when the previous autorefresh command did not perform. It is acknowledged by reading the Interrupt Status register (SDRAMC_ISR).

When the SDRAMC initiates a refresh of the SDRAM device, internal memory accesses are not delayed. However, if the processor tries to access the SDRAM, the slave indicates that the device is busy and the master is held by a wait signal. Refer to the following figure.

**Figure 34-5. Refresh Cycle Followed by a Read Access**




### 34.6.5 Power Management

Three low-power modes are available:

- Self-refresh mode: The SDRAM executes its own Autorefresh cycle without control of the SDRAMC. Current drained by the SDRAM is very low.
- Powerdown mode: Autorefresh cycles are controlled by the SDRAMC. Between autorefresh cycles, the SDRAM is in powerdown. Current drained in Powerdown mode is higher than in Self-refresh Mode.
- Deep Powerdown mode (only available with Mobile SDRAM): The SDRAM contents are lost, but the SDRAM does not drain any current.

The SDRAMC activates one low-power mode as soon as the SDRAM device is not selected. It is possible to delay the entry in Self-refresh and Powerdown modes after the last access by programming a timeout value in the SDRAMC_LPR.

#### 34.6.5.1 Self-refresh Mode

This mode is selected by configuring SDRAMC_LPR.LPCB to 1. In Self-refresh mode, the SDRAM device retains data without external clocking and provides its own internal clocking, thus performing its own autorefresh cycles. All the inputs to the SDRAM device become “don’t care” except SDCKE, which remains low. As soon as the SDRAM device is selected, the SDRAMC provides a sequence of commands and exits Self-refresh mode.

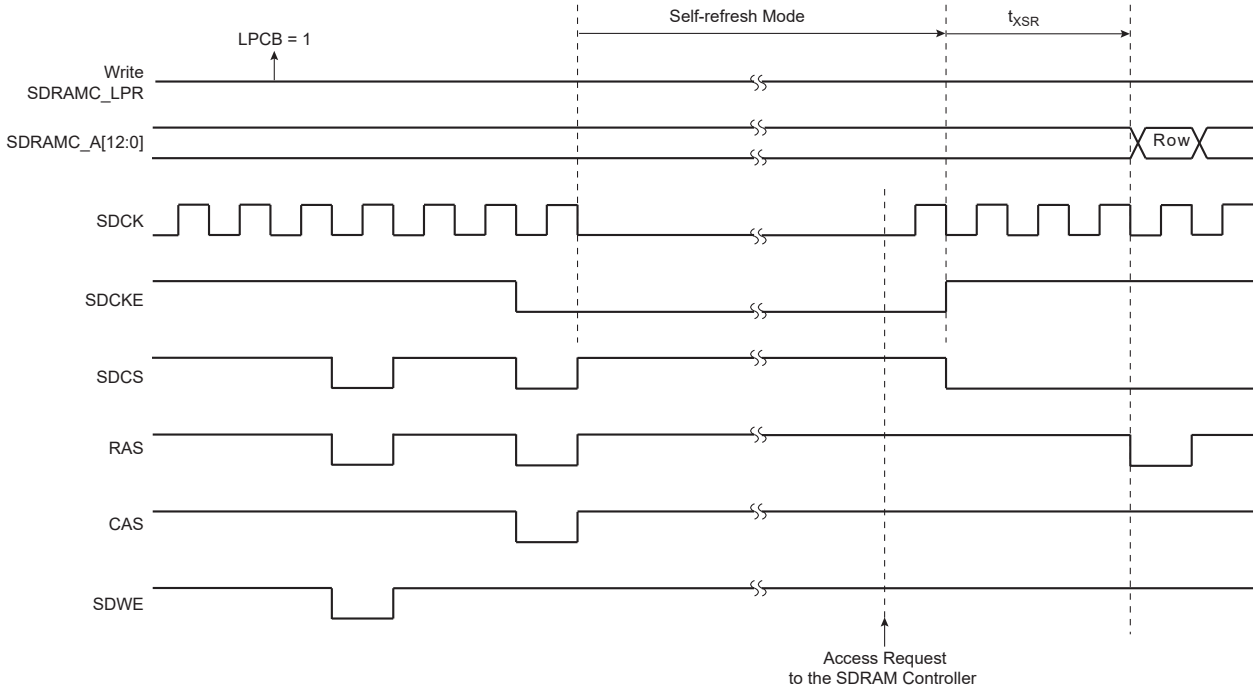
Some low-power SDRAMs (e.g., mobile SDRAM) can refresh only one-quarter or a half quarter or all banks of the SDRAM array. This feature reduces the self-refresh current. To configure this feature, Temperature Compensated Self-Refresh (TCSR), Partial Array Self-Refresh (PASR) and Drive Strength (DS) must be set in the SDRAMC_LPR and transmitted to the low-power SDRAM during initialization.

After initialization, as soon as the PASR/DS/TCSR fields are modified and Self-refresh mode is activated, the Extended Mode register is accessed automatically and the PASR/DS/TCSR bits are updated before entry into Self-refresh mode. This feature is not supported when SDRAMC shares an external bus with another controller.

The SDRAM device must remain in Self-refresh mode for a minimum period of $t_{RAS}$ and may remain in Self-refresh mode for an indefinite period. Refer to the following figure.

**Note:** Some SDRAM providers impose some cycles of burst autorefresh immediately before self-refresh entry and immediately after self-refresh exit. For example, a SDRAM with 4096 rows will impose 4096 cycles of burst autorefresh. This constraint is not supported.

**Figure 34-6. Self-refresh Mode Behavior**

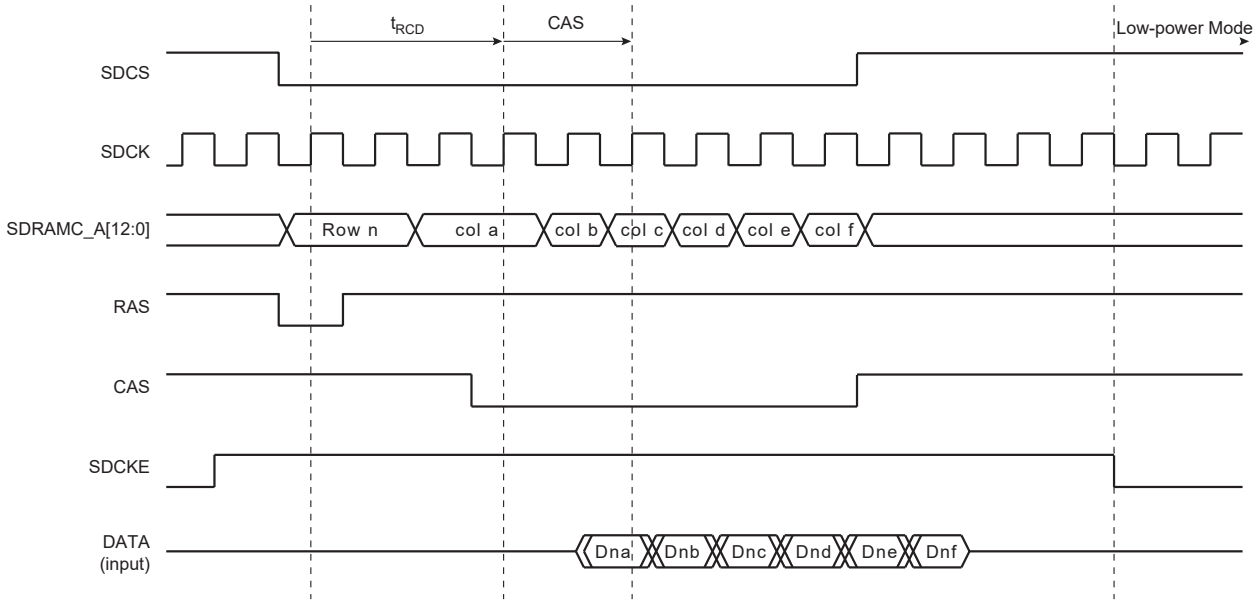



### 34.6.5.2 Low-power Mode

This mode is selected by configuring SDRAMC_LPR.LPCB to 2. Power consumption is greater than in Self-refresh mode. All the input and output buffers of the SDRAM device are deactivated except SDCKE, which remains low. In contrast to Self-refresh mode, the SDRAM device cannot remain in Low-power mode longer than the refresh period (64 ms for a whole device refresh operation). As no autorefresh operations are performed by the SDRAM itself, the SDRAMC carries out the refresh operation. The exit procedure is faster than in Self-refresh mode.

Refer to the following figure.

**Figure 34-7. Low-power Mode Behavior**



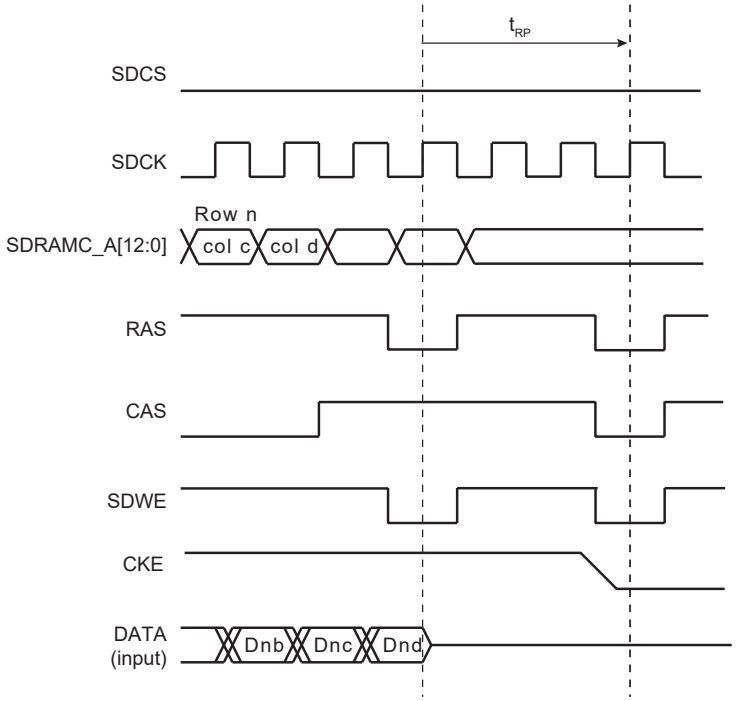
#### 34.6.5.3 Deep Powerdown Mode

This mode is selected by configuring SDRAMC_LPR.LPCB to 3. When this mode is activated, all internal voltage generators inside the SDRAM are stopped and all data is lost.

When this mode is enabled, the application must not access the SDRAM until a new initialization sequence is done (see "SDRAM Device Initialization").

Refer to the following figure.

**Figure 34-8. Deep Powerdown Mode Behavior**




### 34.6.6 Scrambling/Unscrambling Function

The external data bus can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either microcontroller or memory device.

The scrambling and unscrambling are performed on-the-fly without additional wait states.

The scrambling/unscrambling function can be enabled or disabled by configuring the SDR_SE bit in the OCMS register (SDRAMC_OCMS). This bit cannot be reconfigured as long as the external memory device is powered.

The scrambling method depends on two user-configurable key registers, SDRAMC_OCMS_KEY1 and SDRAMC_OCMS_KEY2 plus a random value depending on device processing characteristics. These key registers are only accessible in Write mode.

The scrambling user key or the seed for key generation must be securely stored in a reliable nonvolatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

When multiple chip selects are handled, it is possible to configure the scrambling function per chip select using the OCMS field in the SDRAMC_OCMS registers.

© 2021 Microchip Technology Inc.
and its subsidiaries

## 34.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SDRAMC_MR | 7:0 | | | | | | MODE[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | SDRAMC_TR | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | | | | | COUNT[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | SDRAMC_CR | 7:0 | DBW | CAS[1:0] | | NB | NR[1:0] | | NC[1:0] | |
| | | 15:8 | TRC_TRFC[3:0] | | | | TWR[3:0] | | | |
| | | 23:16 | TRCD[3:0] | | | | TRP[3:0] | | | |
| | | 31:24 | TXSR[3:0] | | | | TRAS[3:0] | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | SDRAMC_LPR | 7:0 | | PASR[2:0] | | | | | LPCB[1:0] | |
| | | 15:8 | | | TIMEOUT[1:0] | | DS[1:0] | | TCSR[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | SDRAMC_IER | 7:0 | | | | | | | | RES |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | SDRAMC_IDR | 7:0 | | | | | | | | RES |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | SDRAMC_IMR | 7:0 | | | | | | | | RES |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | SDRAMC_ISR | 7:0 | | | | | | | | RES |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | SDRAMC_MDR | 7:0 | | | | | | | MD[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | SDRAMC_CFR1 | 7:0 | | | | | TMRD[3:0] | | | |
| | | 15:8 | | | | | | | | UNAL |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | SDRAMC_OCMS | 7:0 | | | | | | | | SDR_SE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | SDRAMC_OCMS_KEY1 | 7:0 | KEY1[7:0] | | | | | | | |
| | | 15:8 | KEY1[15:8] | | | | | | | |
| | | 23:16 | KEY1[23:16] | | | | | | | |
| | | 31:24 | KEY1[31:24] | | | | | | | |
| 0x34 | SDRAMC_OCMS_KEY2 | 7:0 | KEY2[7:0] | | | | | | | |
| | | 15:8 | KEY2[15:8] | | | | | | | |
| | | 23:16 | KEY2[23:16] | | | | | | | |
| | | 31:24 | KEY2[31:24] | | | | | | | |

### 34.7.1 SDRAMC Mode Register

**Name:** SDRAMC_MR
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | MODE[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – MODE[2:0]** SDRAMC Command Mode
This field defines the command issued by the SDRAMC when the SDRAM device is accessed.

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode. Any access to the SDRAM is decoded normally. To activate this mode, the command must be followed by a write to the SDRAM. |
| 1 | NOP | The SDRAMC issues a NOP command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM. |
| 2 | ALLBANKS_PRECHARGE | The SDRAMC issues an “All Banks Precharge” command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM. |
| 3 | LOAD_MODEREG | The SDRAMC issues a “Load Mode Register” command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM. |
| 4 | AUTO_REFRESH | The SDRAMC issues an “Autorefresh” Command when the SDRAM device is accessed regardless of the cycle. Previously, an “All Banks Precharge” command must be issued. To activate this mode, the command must be followed by a write to the SDRAM. |
| 5 | EXT_LOAD_MODEREG | The SDRAMC issues an “Extended Load Mode Register” command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the “Extended Load Mode Register” command must be followed by a write to the SDRAM. The write in the SDRAM must be done in the appropriate bank; most low-power SDRAM devices use the bank 1. |
| 6 | DEEP_POWERDOWN | Deep Powerdown mode. Enters Deep Powerdown mode. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 34.7.2 SDRAMC Refresh Timer Register

**Name:** SDRAMC_TR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | COUNT[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – COUNT[11:0]** SDRAMC Refresh Timer Count
This 12-bit field is loaded into a timer that generates the refresh pulse. Each time the refresh pulse is generated, a refresh burst is initiated. The SDRAM device requires a refresh every 15.625 µs or 7.81 µs. With a 100 MHz frequency, the Refresh Timer Counter Register must be set with the value 1562 (15.625 µs x 100 MHz) or 781 (7.81 µs x 100 MHz).
To refresh the SDRAM device, this 12-bit field must be written. If this condition is not satisfied, no refresh command is issued and no refresh of the SDRAM device is carried out.

### 34.7.3 SDRAMC Configuration Register

**Name:** SDRAMC_CR
**Offset:** 0x08
**Reset:** 0x852372C0
**Property:** Read/Write

**WARNING** Bit 7 (DBW) must always be set when programming the SDRAMC_CR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TXSR[3:0] | | | | TRAS[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TRCD[3:0] | | | | TRP[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TRC_TRFC[3:0] | | | | TWR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DBW | CAS[1:0] | | NB | NR[1:0] | | NC[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:28 – TXSR[3:0]** Exit Self-Refresh to Active Delay
Reset value is eight cycles.
This field defines the delay between SCKE set high and an Activate Command in number of cycles. Number of cycles is between 0 and 15.

**Bits 27:24 – TRAS[3:0]** Active to Precharge Delay
Reset value is five cycles.
This field defines the delay between an Activate Command and a Precharge Command in number of cycles. Number of cycles is between 0 and 15.

**Bits 23:20 – TRCD[3:0]** Row to Column Delay
Reset value is two cycles.
This field defines the delay between an Activate Command and a Read/Write Command in number of cycles. Number of cycles is between 0 and 15.

**Bits 19:16 – TRP[3:0]** Row Precharge Delay
Reset value is three cycles.
This field defines the delay between a Precharge Command and another Command in number of cycles. Number of cycles is between 0 and 15.

**Bits 15:12 – TRC_TRFC[3:0]** Row Cycle Delay and Row Refresh Cycle
Reset value is seven cycles.
This field defines two timings:

- the delay ($t_{RFC}$) between two Refresh commands and between a Refresh command and an Activate command
- the delay ($t_{RC}$) between two Active commands in number of cycles.

The number of cycles is between 0 and 15. The end user must program max {$t_{RC}$, $t_{RFC}$}.

**Bits 11:8 – TWR[3:0]** Write Recovery Delay
Reset value is two cycles.
This field defines the Write Recovery Time in number of cycles. Number of cycles is between 0 and 15.

**Bit 7 – DBW** Data Bus Width
Reset value is 16 bits.
This bit defines the Data Bus Width, which is 16 bits. It must be set to 1.

| Value | Description |
|---|---|
| 0 | Data bus width is 32 bits. |
| 1 | Data bus width is 16 bits. |

**Bits 6:5 – CAS[1:0]** CAS Latency
Reset value is two cycles. In the SDRAMC, only a CAS latency of two and three cycles is managed.

| Value | Name | Description |
|---|---|---|
| 0 | Reserved | – |
| 1 | Reserved | – |
| 1 | LATENCY1 | 1 cycle latency |
| 2 | LATENCY2 | 2 cycle latency |
| 3 | LATENCY3 | 3 cycle latency |

**Bit 4 – NB** Number of Banks
Reset value is two banks.

| Value | Name | Description |
|---|---|---|
| 0 | BANK2 | 2 banks |
| 1 | BANK4 | 4 banks |

**Bits 3:2 – NR[1:0]** Number of Row Bits
Reset value is 11 row bits.

| Value | Name | Description |
|---|---|---|
| 0 | ROW11 | 11 bits to define the row number, up to 2048 rows |
| 1 | ROW12 | 12 bits to define the row number, up to 4096 rows |
| 2 | ROW13 | 13 bits to define the row number, up to 8192 rows |
| 3 | Reserved | |

**Bits 1:0 – NC[1:0]** Number of Column Bits
Reset value is 8 column bits.

| Value | Name | Description |
|---|---|---|
| 0 | COL8 | 8 bits to define the column number, up to 256 columns. |
| 1 | COL9 | 9 bits to define the column number, up to 512 columns. |
| 2 | COL10 | 10 bits to define the column number, up to 1024 columns. |
| 3 | COL11 | 11 bits to define the column number, up to 2048 columns. |

### 34.7.4 SDRAMC Low-Power Register

**Name:** SDRAMC_LPR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TIMEOUT[1:0] | | DS[1:0] | | TCSR[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PASR[2:0] | | | | | LPCB[1:0] | |
| Access | | R/W | R/W | R/W | | | R/W | R/W |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

**Bits 13:12 – TIMEOUT[1:0]** Time to Define When Low-power Mode Is Enabled

| Value | Name | Description |
|---|---|---|
| 0 | LP_LAST_XFER | The SDRAMC activates the SDRAM Low-power mode immediately after the end of the last transfer. |
| 1 | LP_LAST_XFER_64 | The SDRAMC activates the SDRAM Low-power mode 64 clock cycles after the end of the last transfer. |
| 2 | LP_LAST_XFER_128 | The SDRAMC activates the SDRAM Low-power mode 128 clock cycles after the end of the last transfer. |
| 3 | Reserved | |

**Bits 11:10 – DS[1:0]** Drive Strength (only for low-power SDRAM)
DS is transmitted to the SDRAM during initialization to select the SDRAM strength of data output. This parameter must be set according to the SDRAM device specification.
After initialization, as soon as the DS field is modified and Self-refresh mode is activated, the Extended Mode Register is accessed automatically and DS bits are updated before entry in Self-refresh mode. This feature is not supported when SDRAMC shares an external bus with another controller.

**Bits 9:8 – TCSR[1:0]** Temperature Compensated Self-Refresh (only for low-power SDRAM)
TCSR is transmitted to the SDRAM during initialization to set the refresh interval during Self-refresh mode depending on the temperature of the low-power SDRAM. This parameter must be set according to the SDRAM device specification.
After initialization, as soon as the TCSR field is modified and Self-refresh mode is activated, the Extended Mode Register is accessed automatically and TCSR bits are updated before entry in Self-refresh mode. This feature is not supported when SDRAMC shares an external bus with another controller.

**Bits 6:4 – PASR[2:0]** Partial Array Self-refresh (only for low-power SDRAM)
PASR is transmitted to the SDRAM during initialization to specify whether only one quarter, one half or all banks of the SDRAM array are enabled. Disabled banks are not refreshed in Self-refresh mode. This parameter must be set according to the SDRAM device specification.

© 2021 Microchip Technology Inc. and its subsidiaries

After initialization, as soon as the PASR field is modified and Self-refresh mode is activated, the Extended Mode Register is accessed automatically and PASR bits are updated before entry in Self-refresh mode. This feature is not supported when SDRAMC shares an external bus with another controller.

**Bits 1:0 – LPCB[1:0]** Low-power Configuration Bits

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | The low-power feature is inhibited: no Powerdown, Self-refresh or Deep Powerdown command is issued to the SDRAM device. |
| 1 | SELF_REFRESH | The SDRAMC issues a Self-refresh command to the SDRAM device, the SDCK clock is deactivated and the SDCKE signal is set low. The SDRAM device leaves the Self-refresh mode when accessed and enters it after the access. |
| 2 | POWER_DOWN | The SDRAMC issues a Powerdown Command to the SDRAM device after each access, the SDCKE signal is set to low. The SDRAM device leaves the Powerdown mode when accessed and enters it after the access. |
| 3 | DEEP_POWER_DOWN | The SDRAMC issues a Deep Powerdown command to the SDRAM device. This mode is unique to low-power SDRAM. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 34.7.5 SDRAMC Interrupt Enable Register

**Name:** SDRAMC_IER
**Offset:** 0x14
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | RES |

**Bit 0 – RES** Refresh Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the refresh error interrupt. |

### 34.7.6 SDRAMC Interrupt Disable Register

**Name:** SDRAMC_IDR
**Offset:** 0x18
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | W |
| Reset | | | | | | | | RES |

**Bit 0 – RES** Refresh Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the refresh error interrupt. |

### 34.7.7 SDRAMC Interrupt Mask Register

**Name:** SDRAMC_IMR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RES |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – RES** Refresh Error Interrupt Mask

| Value | Description |
|---|---|
| 0 | The refresh error interrupt is disabled. |
| 1 | The refresh error interrupt is enabled. |

### 34.7.8 SDRAMC Interrupt Status Register

**Name:** SDRAMC_ISR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | RES |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – RES** Refresh Error Status (cleared on read)

| Value | Description |
|---|---|
| 0 | No refresh error has been detected since the register was last read. |
| 1 | A refresh error has been detected since the register was last read. |

### 34.7.9 SDRAMC Memory Device Register

**Name:** SDRAMC_MDR
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | MD[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 1:0 – MD[1:0]** Memory Device Type

| Value | Name | Description |
|---|---|---|
| 0 | SDRAM | SDRAM |
| 1 | LPSDRAM | Low-power SDRAM |
| 2 | – | Reserved |
| 3 | – | Reserved |

### 34.7.10 SDRAMC Configuration Register 1

**Name:** SDRAMC_CFR1
**Offset:** 0x28
**Reset:** 0x00000002
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | UNAL |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TMRD[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 1 | 0 |

**Bit 8 – UNAL** Support Unaligned Access
This mode is enabled with masters which have an AXI interface.

| Value | Name | Description |
|---|---|---|
| 0 | UNSUPPORTED | Unaligned access is not supported. |
| 1 | SUPPORTED | Unaligned access is supported. |

**Bits 3:0 – TMRD[3:0]** Load Mode Register Command to Active or Refresh Command
Reset value is 2 cycles.
This field defines the delay between a “Load Mode Register” command and an active or refresh command in number of cycles. Number of cycles is between 0 and 15.

### 34.7.11 SDRAMC OCMS Register

**Name:** SDRAMC_OCMS
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SDR_SE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – SDR_SE** SDRAM Memory Controller Scrambling Enable

| Value | Description |
|---|---|
| 0 | Disables off-chip scrambling for SDR-SDRAM access. |
| 1 | Enables off-chip scrambling for SDR-SDRAM access. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 34.7.12 SDRAMC OCMS KEY1 Register

**Name:** SDRAMC_OCMS_KEY1
**Offset:** 0x30
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | | | | | | | | |

**Bits 31:0 – KEY1[31:0]** Off-chip Memory Scrambling (OCMS) Key Part 1
When off-chip memory scrambling is enabled, the data scrambling depends on KEY1 and KEY2 values.

### 34.7.13 SDRAMC OCMS KEY2 Register

**Name:** SDRAMC_OCMS_KEY2
**Offset:** 0x34
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – KEY2[31:0]** Off-chip Memory Scrambling (OCMS) Key Part 2
When off-chip memory scrambling is enabled, the data scrambling depends on KEY1 and KEY2 values.

# 35. Static Memory Controller (SMC)

## 35.1 Description

The External Bus Interface (EBI) is designed to ensure the successful data transfer between several external devices and the ARM-based microcontroller. The Static Memory Controller (SMC) is part of the EBI.

The SMC handles several types of external memory and peripheral devices, such as SRAM, PSRAM, PROM, EPROM, EEPROM, LCD Module, NOR Flash and NAND Flash.

The SMC generates the signals that control the access to the external memory devices or peripheral devices. It has 4 chip selects, a 24-bit address bus, and a configurable 8 or 16-bit data bus. Separate read and write control signals allow for direct memory and peripheral interfacing. Read and write signal waveforms are fully adjustable.

The SMC can manage wait requests from external devices to extend the current access. The SMC is provided with an automatic Slow clock mode. In Slow clock mode, it switches from user-programmed waveforms to slow-rate specific waveforms on read and write signals. The SMC supports asynchronous burst read in Page mode access for page sizes up to 32 bytes.

The external data bus can be scrambled/unscrambled by means of user keys.

## 35.2 Embedded Characteristics

- Four Chip Selects Available
- 16-Mbyte Address Space per Chip Select
- 8-bit or 16-bit Data Bus
- Zero Wait State Scrambling/Unscrambling Function with User Key
- Word, Halfword, Byte Transfers
- Byte Write or Byte Select Lines
- Programmable Setup, Pulse And Hold Time for Read Signals per Chip Select
- Programmable Setup, Pulse And Hold Time for Write Signals per Chip Select
- Programmable Data Float Time per Chip Select
- External Wait Request
- Automatic Switch to Slow Clock Mode
- Asynchronous Read in Page Mode Supported: Page Size Ranges from 4 to 32 Bytes
- Register Write Protection

## 35.3 I/O Lines Description

**Table 35-1. I/O Line Description**

| Name | Description | Type | Active Level |
|---|---|---|---|
| NCS[3:0] | Static Memory Controller Chip Select Lines | Output | Low |
| NRD | Read Signal | Output | Low |
| NWR0/NWE | Write 0/Write Enable Signal | Output | Low |
| NWR1/NBS1 | Write 1/Byte 1 Select Signal | Output | Low |
| A0/NBS0 | Address Bit 0/Byte 0 Select Signal | Output | Low |
| A[23:1] | Address Bus | Output | – |
| D[15:0] | Data Bus | I/O | – |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | | | |
|---|---|---|---|
| **Name** | **Description** | **Type** | **Active Level** |
| NWAIT | External Wait Signal | Input | Low |
| NANDCS | NAND Flash Chip Select Line | Output | Low |
| NANDOE | NAND Flash Output Enable | Output | Low |
| NANDWE | NAND Flash Write Enable | Output | Low |
| NANDALE | NAND Flash Address Latch Enable | Output | – |
| NANDCLE | NAND Flash Command Latch Enable | Output | – |

## 35.4 Multiplexed Signals

**Table 35-2. Static Memory Controller (SMC) Multiplexed Signals**

| Multiplexed Signals | | Related Function |
|---|---|---|
| NWR0 | NWE | Byte-write or Byte-select access.<br>See "Byte Write Access" and "Byte Select Access" |
| A0 | NBS0 | 8-bit or 16-bit data bus. See "Data Bus Width" |
| NWR1 | NBS1 | Byte-write or Byte-select access. See "Byte Write Access" and "Byte Select Access" |
| A22 | NANDCLE | NAND Flash Command Latch Enable |
| A21 | NANDALE | NAND Flash Address Latch Enable |

## 35.5 Product Dependencies

### 35.5.1 I/O Lines

The pins used for interfacing the SMC are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the SMC pins to their peripheral function. If I/O lines of the SMC are not used by the application, they can be used for other purposes by the PIO Controller.

### 35.5.2 Power Management

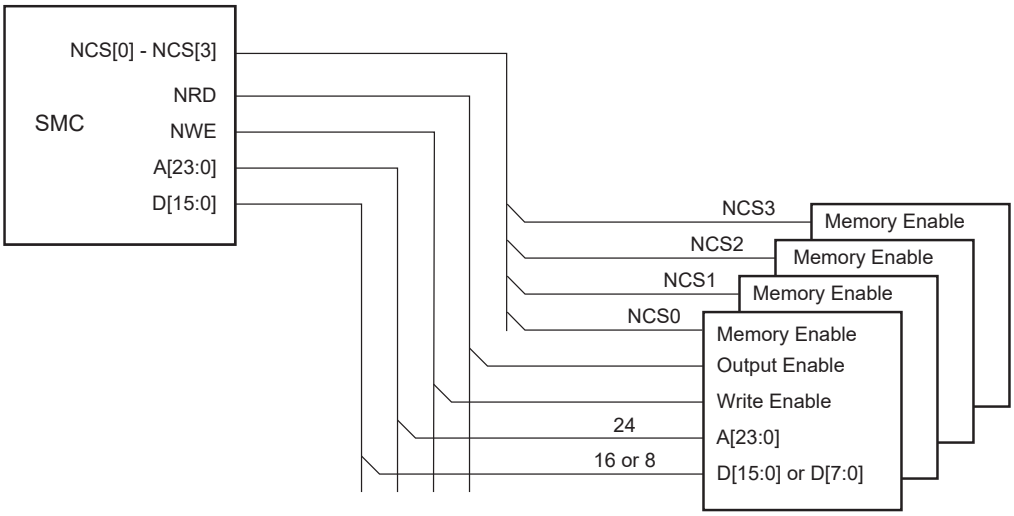
The SMC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SMC clock.

## 35.6 External Memory Mapping

The SMC provides up to 24 address lines, A[23:0]. This allows each chip select line to address up to 16 Mbytes of memory.

If the physical memory device connected on one chip select is smaller than 16 Mbytes, it wraps around and appears to be repeated within this space. The SMC correctly handles any valid access to the memory device within the page (see the following figure).

**Figure 35-1. Memory Connections for Four External Devices**




## 35.7 Connection to External Devices

### 35.7.1 Data Bus Width

A data bus width of 8 or 16 bits can be selected for each chip select. This option is controlled by the bit DBW in the Mode register (SMC_MODE) for the corresponding chip select.

Figure 35-2 shows how to connect a 512-Kbyte × 8-bit memory on NCS2. Figure 35-3 shows how to connect a 512-Kbyte × 16-bit memory on NCS2.

**Figure 35-2. Memory Connection for an 8-bit Data Bus**

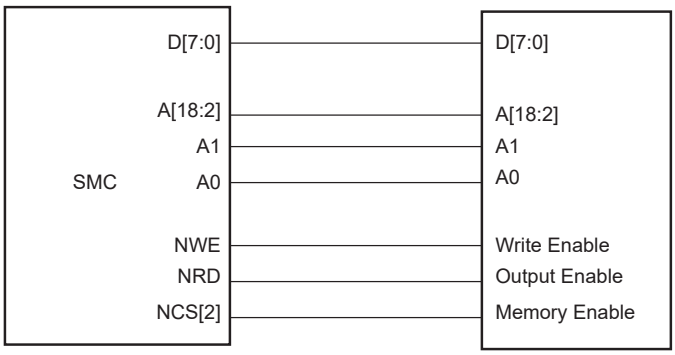
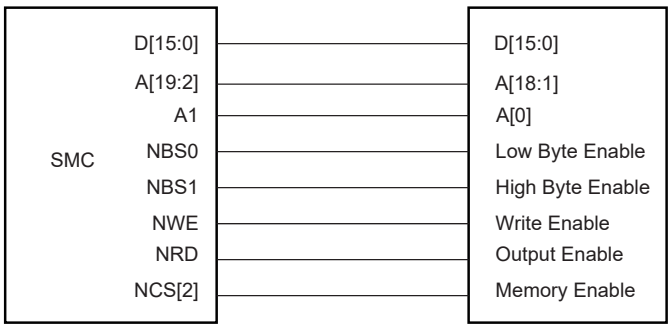



**Figure 35-3. Memory Connection for a 16-bit Data Bus**



### 35.7.2 Byte Write or Byte Select Access

Each chip select with a 16-bit data bus can operate with one of two different types of write access: byte write or byte select. This is controlled by the BAT field of the SMC_MODE register for the corresponding chip select.

#### 35.7.2.1 Byte Write Access

Byte write access is used to connect 2 × 8-bit devices as a 16-bit memory, and supports one write signal per byte of the data bus and a single read signal.

Note that the SMC does not allow boot in Byte write access mode.

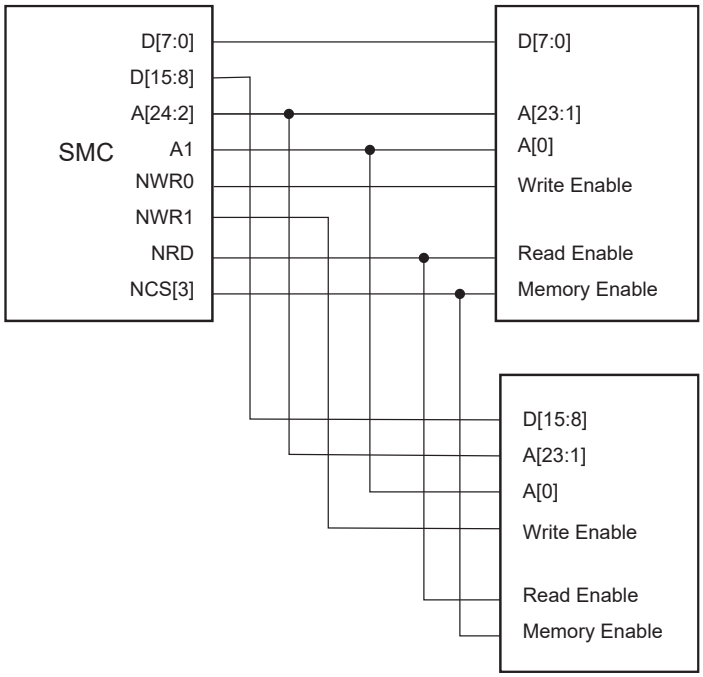
For 16-bit devices, the SMC provides NWR0 and NWR1 write signals for respectively Byte0 (lower byte) and Byte1 (upper byte) of a 16-bit bus. One single read signal (NRD) is provided.

#### 35.7.2.2 Byte Select Access

Byte select access is used to connect one 16-bit device. In this mode, read/write operations can be enabled/disabled at byte level. One byte-select line per byte of the data bus is provided. One NRD and one NWE signal control read and write.

For 16-bit devices, the SMC provides NBS0 and NBS1 selection signals for respectively Byte0 (lower byte) and Byte1 (upper byte) of a 16-bit bus.

**Figure 35-4. Connection of 2 × 8-bit Devices on a 16-bit Bus: Byte Write Option**




#### 35.7.2.3 Signal Multiplexing

Depending on the byte access type (BAT), only the byte write signals or the byte select signals are used. To save I/Os at the external bus interface, control signals at the SMC interface are multiplexed. The following table shows signal multiplexing depending on the data bus width and the byte access type.

For 16-bit devices, bit A0 of address is unused. When the Byte Select option is selected, NWR1 is unused. When the Byte Write option is selected, NBS0 is unused.

**Table 35-3. SMC Multiplexed Signal Translation**

| Device Type | Signal Name | | |
|---|---|---|---|
| | 16-bit Bus | | 8-bit Bus |
| | 1 x 16-bit | 2 x 8-bit | 1 x 8-bit |
| Byte Access Type (BAT) | Byte Select | Byte Write | – |
| NBS0_A0 | NBS0 | – | A0 |
| NWE_NWR0 | NWE | NWR0 | NWE |
| NBS1_NWR1 | NBS1 | NWR1 | – |
| A1 | A1 | A1 | A1 |

### 35.7.3 NAND Flash Support

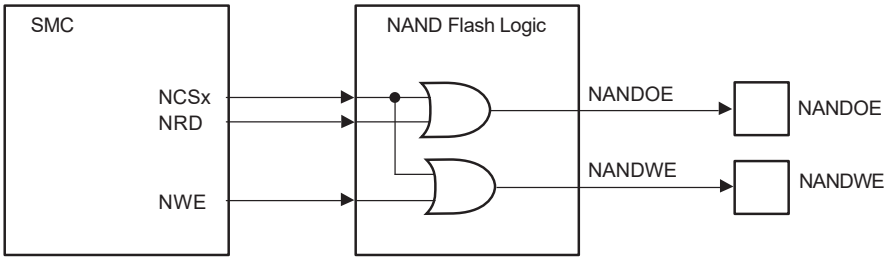
The SMC integrates circuitry that interfaces to NAND Flash devices.

The NAND Flash logic is driven by the SMC. Configuration is done via the SMC_NFCSx field in the CCFG_SMCNFCS register in the Bus Matrix. For details on this register, refer to the section “Bus Matrix (MATRIX)” of this datasheet. The external NAND Flash device is accessed via the address space reserved for the chip select programmed.

The user can connect up to four NAND Flash devices with separate chip selects.

The NAND Flash logic drives the read and write command signals of the SMC on the NANDOE and NANDWE signals when the NCSx programmed is active. NANDOE and NANDWE are disabled as soon as the transfer address fails to lie in the NCSx programmed address space.

**Figure 35-5. NAND Flash Signal Multiplexing on SMC Pins**




**Note:** 1. NCSx is active when CCFG_SMCNFCS.SMC_NFCSx=1.

**Note:** 2. When the NAND Flash logic is activated, (SMC_NFCSx=1), the NWE pin can be used only in Peripheral mode (NWE function). If the NWE function is not used for other external memories (SRAM, LCD), it must be configured in one of the following modes:
PIO input with pull-up enabled (default state after reset)
and PIO output set at level 1.

The address latch enable and command latch enable signals on the NAND Flash device are driven by address bits A22 and A21of the address bus. Any bit of the address bus can also be used for this purpose. The command, address or data words on the data bus of the NAND Flash device use their own addresses within the NCSx address space (configured in the register CCFG_SMCNFCS in the Bus Matrixe). The chip enable (CE) signal of the device and the ready/busy (R/B) signals are connected to PIO lines. The CE signal then remains asserted even when NAND Flash chip select is not selected, preventing the device from returning to Standby mode. The NANDCS output signal should be used in accordance with the external NAND Flash device type.

Two types of CE behavior exist depending on the NAND Flash device:

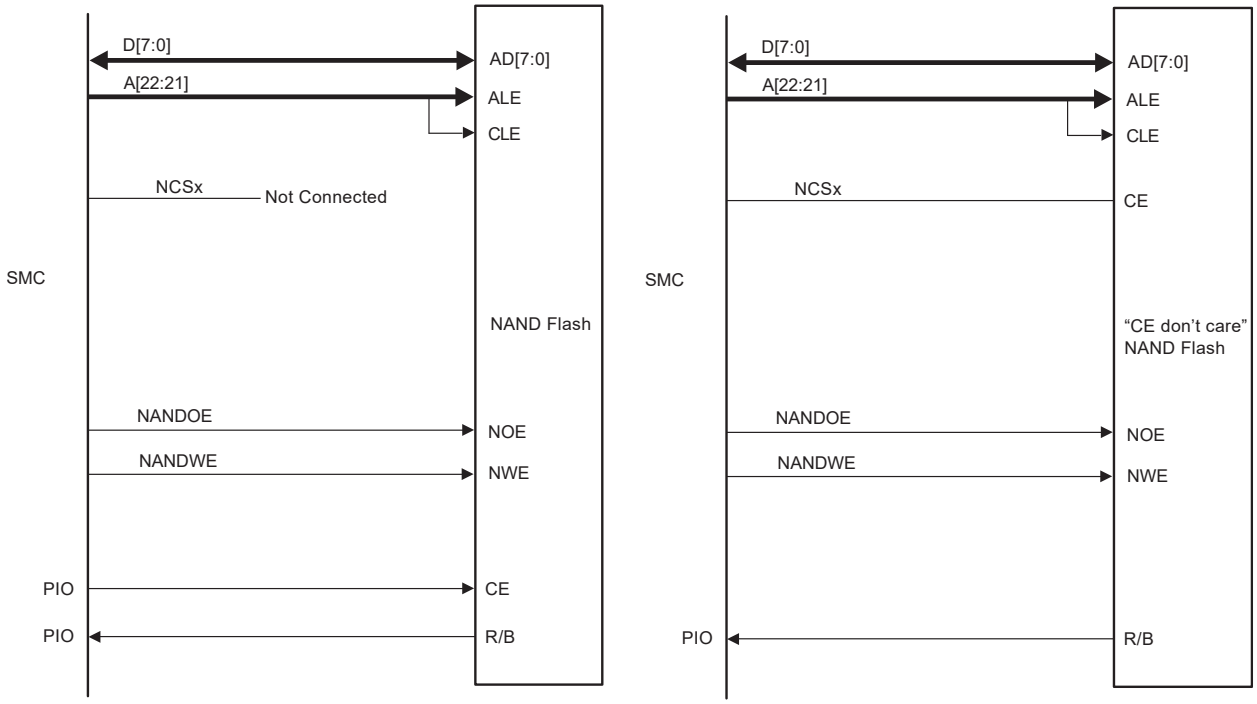
- Standard NAND Flash devices require that the CE pin remains asserted low continuously during the read busy period to prevent the device from returning to Standby mode. Since the SMC asserts the NCSx signal high, it is necessary to connect the CE pin of the NAND Flash device to a GPIO line, in order to hold it low during the busy period preceding data read out.

© 2021 Microchip Technology Inc.
and its subsidiaries

- This restriction has been removed for “CE don’t care” NAND Flash devices. The NCSx signal can be directly connected to the CE pin of the NAND Flash device.

The following figure illustrates both topologies: Standard and “CE don’t care” NAND Flash.

**Figure 35-6. Standard and “CE don’t care” NAND Flash Application Examples**




**Related Links**

19. Bus Matrix (MATRIX)

## 35.8 Application Example

### 35.8.1 Implementation Examples

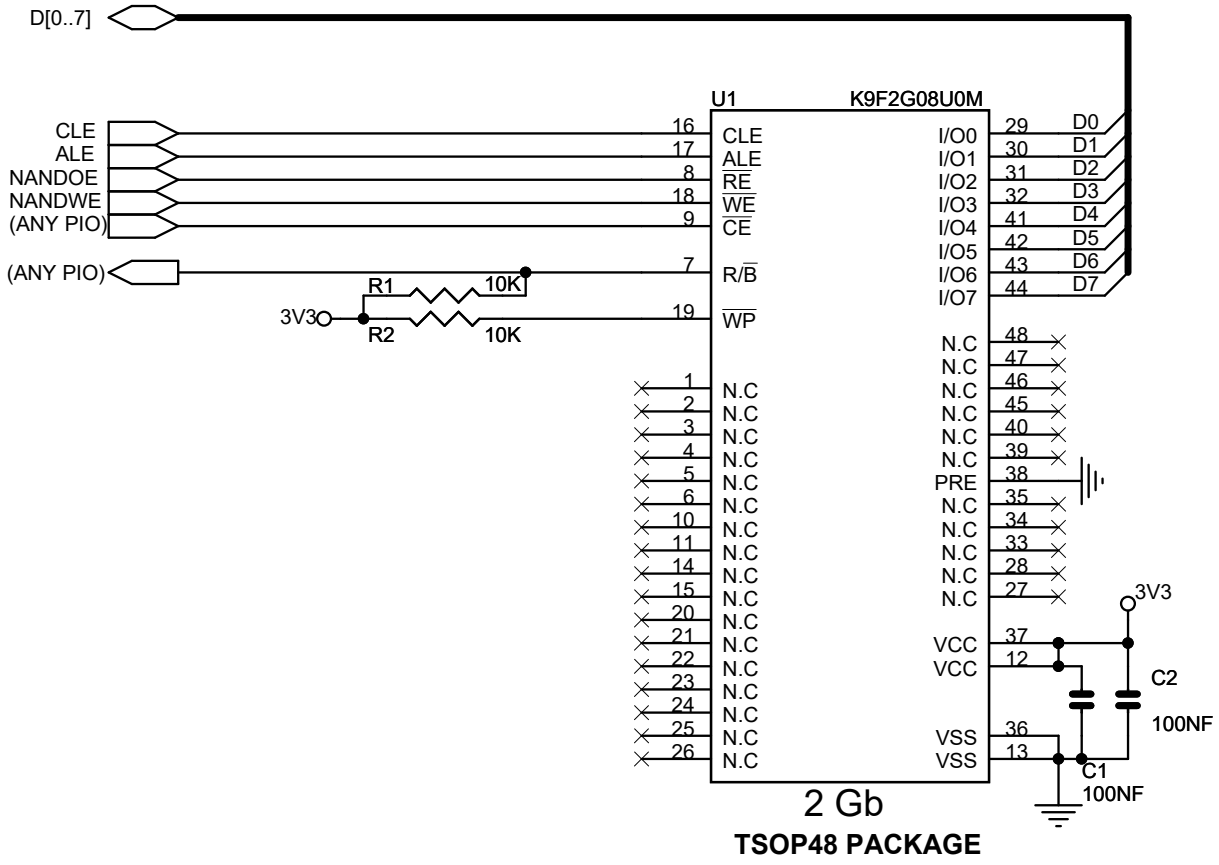
Hardware configurations are given for illustration only. The user should refer to the manufacturer web site to check for memory device availability.

For hardware implementation examples, refer to the evaluation kit schematics for this microcontroller, which show examples of a connection to an LCD module and NAND Flash.

#### 35.8.1.1 8-bit NAND Flash

**Hardware Configuration**

**Figure 35-7. 8-bit NAND Flash**




Software Configuration
Perform the following configuration:

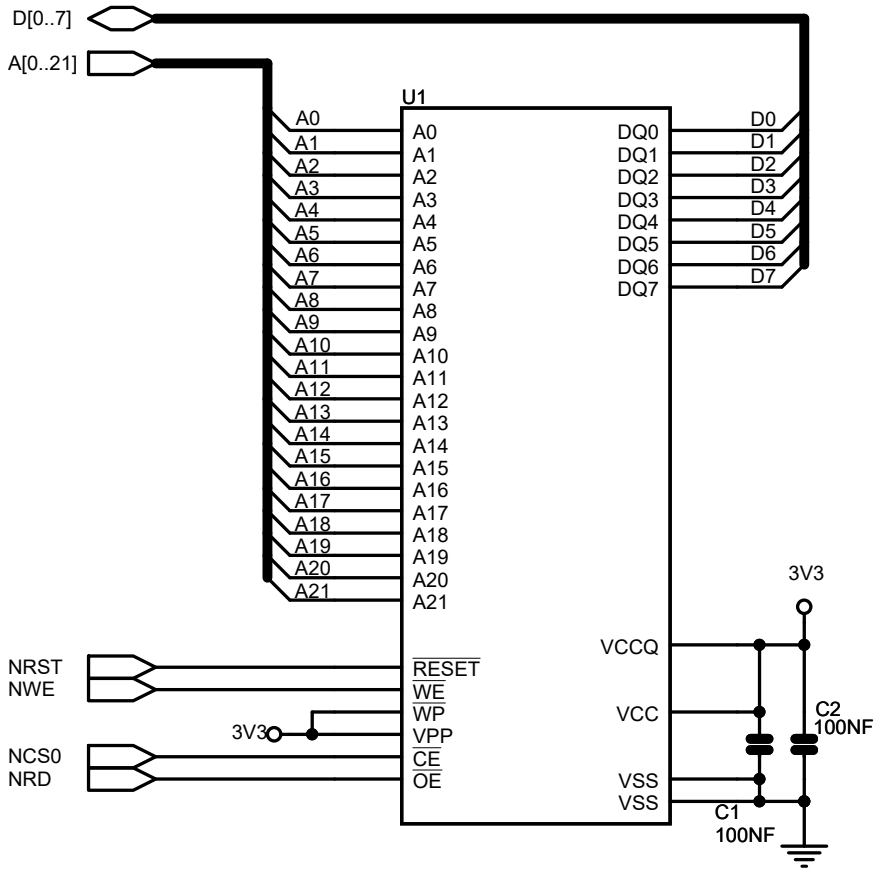
1. Select the chip select used to drive the NAND Flash by setting the bit CCFG_SMCNFCS.SMC_NFCSx.
2. Reserve A21 / A22 for ALE / CLE functions. Address and Command Latches are controlled by setting the address bits A21 and A22, respectively, during accesses.
3. NANDOE and NANDWE signals are multiplexed with PIO lines. Thus, the dedicated PIOs must be programmed in Peripheral mode in the PIO controller.
4. Configure a PIO line as an input to manage the Ready/Busy signal.
5. Configure SMC CS3 Setup, Pulse, Cycle and Mode according to NAND Flash timings, the data bus width and the system bus frequency.

In this example, the NAND Flash is not addressed as a “CE don’t care”. To address it as a “CE don’t care”, connect NCS3 (if SMC_NFCS3 is set) to the NAND Flash CE.

#### 35.8.1.2 NOR Flash

**Hardware Configuration**

**Figure 35-8. NOR Flash**




Software Configuration
Configure the SMC CS0 Setup, Pulse, Cycle, and Mode, depending on Flash timings and system bus frequency.

## 35.9 Standard Read and Write Protocols

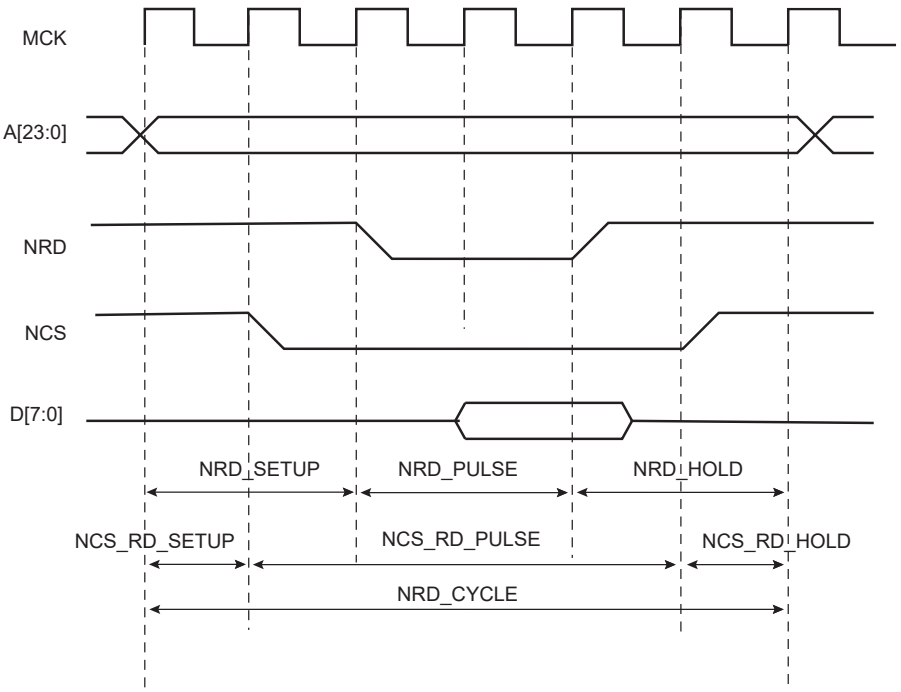
In the following sections, the byte access type is not considered. Byte select lines (NBS0 to NBS1) always have the same timing as the A address bus. NWE represents either the NWE signal in byte select access type or one of the byte write lines (NWR0 to NWR1) in byte write access type. NWR0 to NWR1 have the same timings and protocol as NWE. If D[15:8] are used, they have the same timing as D[7:0]. In the same way, NCS represents one of the NCS[0..3] chip select lines.

### 35.9.1 Read Waveforms

The read cycle is shown in the following figure.

The read cycle starts with the address setting on the memory address bus.

**Figure 35-9. Standard Read Cycle**




#### 35.9.1.1 NRD Waveform

The NRD signal is characterized by a setup timing, a pulse width and a hold timing.

- nrd_setup— NRD setup time is defined as the setup of address before the NRD falling edge;
- nrd_pulse—NRD pulse length is the time between NRD falling edge and NRD rising edge;
- nrd_hold—NRD hold time is defined as the hold time of address after the NRD rising edge.

#### 35.9.1.2 NCS Waveform

The NCS signal can be divided into a setup time, pulse length and hold time:

- ncs_rd_setup—NCS setup time is defined as the setup time of address before the NCS falling edge.
- ncs_rd_pulse—NCS pulse length is the time between NCS falling edge and NCS rising edge;
- ncs_rd_hold—NCS hold time is defined as the hold time of address after the NCS rising edge.

#### 35.9.1.3 Read Cycle

The NRD_CYCLE time is defined as the total duration of the read cycle, i.e., from the time where address is set on the address bus to the point where address may change. The total read cycle time is defined as:

NRD_CYCLE = NRD_SETUP + NRD_PULSE + NRD_HOLD,

as well as

NRD_CYCLE = NCS_RD_SETUP + NCS_RD_PULSE + NCS_RD_HOLD

All NRD and NCS timings are defined separately for each chip select as an integer number of Master Clock cycles. The NRD_CYCLE field is common to both the NRD and NCS signals, thus the timing period is of the same duration.

NRD_CYCLE, NRD_SETUP, and NRD_PULSE implicitly define the NRD_HOLD value as:

NRD_HOLD = NRD_CYCLE - NRD SETUP - NRD PULSE

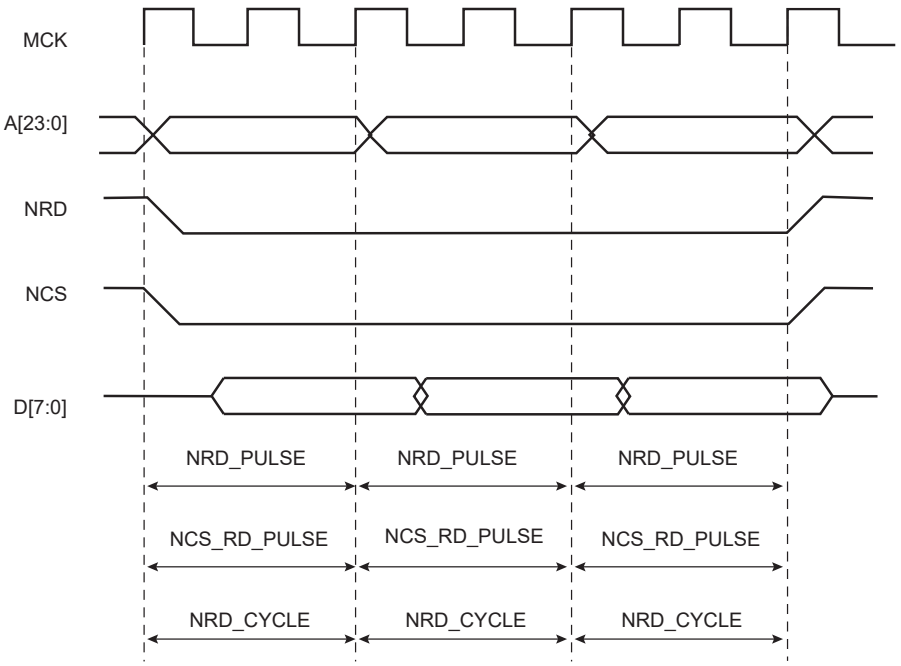
NRD_CYCLE, NCS_RD_SETUP, and NCS_RD_PULSE implicitly define the NCS_RD_HOLD value as:

NCS_RD_HOLD = NRD_CYCLE - NCS_RD_SETUP - NCS_RD_PULSE

#### 35.9.1.4 Null Delay Setup and Hold

If null setup and hold parameters are programmed for NRD and/or NCS, NRD and NCS remain active continuously in case of consecutive read cycles in the same memory (see the following figure).

**Figure 35-10. No Setup, No Hold on NRD and NCS Read Signals**




#### 35.9.1.5 Null Pulse

Programming a null pulse is not permitted. The pulse must be at least set to 1. A null value leads to unpredictable behavior.

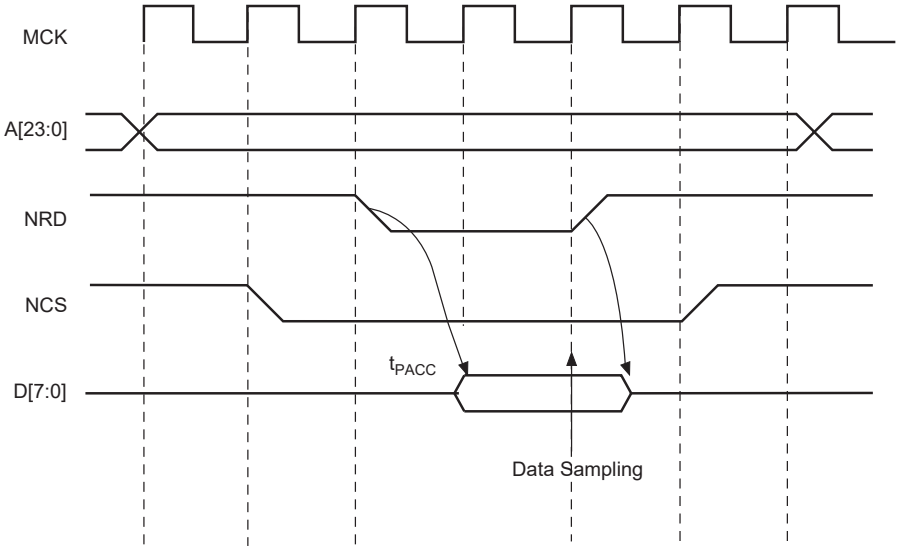
### 35.9.2 Read Mode

As NCS and NRD waveforms are defined independently of one other, the SMC needs to know when the read data is available on the data bus. The SMC does not compare NCS and NRD timings to know which signal rises first. The READ_MODE bit in the SMC_MODE register of the corresponding chip select indicates which signal of NRD and NCS controls the read operation.

#### 35.9.2.1 Read is Controlled by NRD (SMC_MODE.READ_MODE = 1):

The following figure shows the waveforms of a read operation of a typical asynchronous RAM. The read data is available $t_{PACC}$ after the falling edge of NRD, and turns to ‘Z’ after the rising edge of NRD. In this case, SMC_MODE.READ_MODE must be set to 1 (read is controlled by NRD), to indicate that data is available with the rising edge of NRD. The SMC samples the read data internally on the rising edge of Master Clock that generates the rising edge of NRD, whatever the programmed waveform of NCS may be.

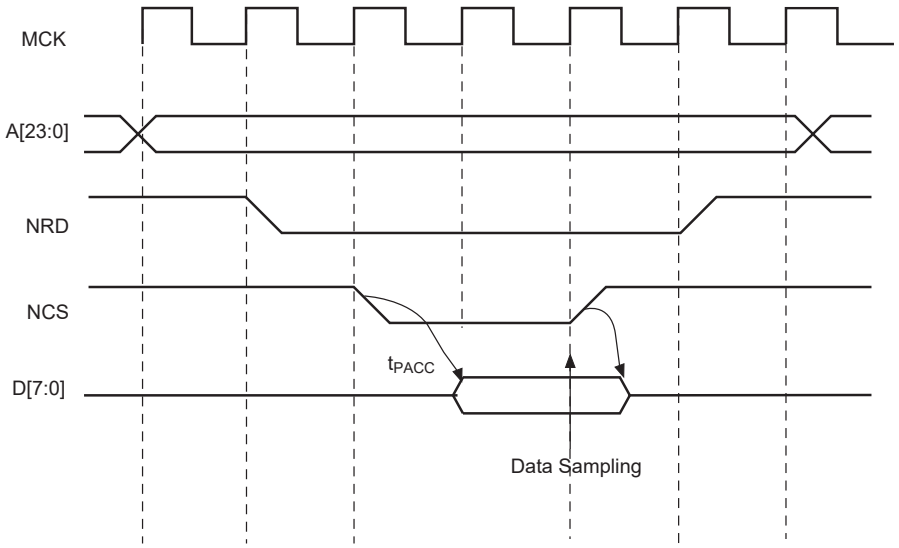
**Figure 35-11. SMC_MODE.READ_MODE = 1: Data is sampled by SMC before the rising edge of NRD**




#### 35.9.2.2 Read is Controlled by NCS (SMC_MODE.READ_MODE = 0)

The following figure shows the typical read cycle of an LCD module. The read data is valid $t_{PACC}$ after the falling edge of the NCS signal and remains valid until the rising edge of NCS. Data must be sampled when NCS is raised. In this case, the SMC_MODE.READ_MODE must be set to 0 (read is controlled by NCS): the SMC internally samples the data on the rising edge of Master Clock that generates the rising edge of NCS, whatever the programmed waveform of NRD may be.

**Figure 35-12. SMC_MODE.READ_MODE = 0: Data is Sampled by SMC Before the Rising Edge of NCS**




### 35.9.3 Write Waveforms

The write protocol is similar to the read protocol. It is depicted in Figure 35-13. The write cycle starts with the address setting on the memory address bus.

#### 35.9.3.1 NWE Waveforms

The NWE signal is characterized by a setup timing, a pulse width and a hold timing.

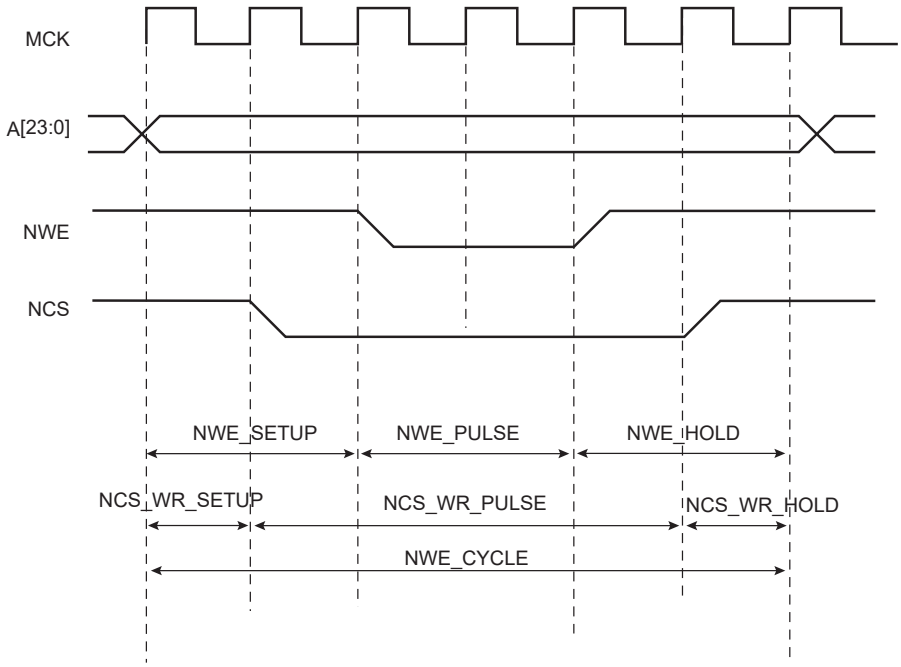
- NWE_SETUP—the NWE setup time is defined as the setup of address and data before the NWE falling edge;
- NWE_PULSE—the NWE pulse length is the time between NWE falling edge and NWE rising edge;
- NWE_HOLD—the NWE hold time is defined as the hold time of address and data after the NWE rising edge.

### 35.9.3.2 NCS Waveforms

The NCS signal waveforms in write operation are not the same that those applied in read operations, but are separately defined:

- ncs_wr_setup—the NCS setup time is defined as the setup time of address before the NCS falling edge.
- ncs_wr_pulse—the NCS pulse length is the time between NCS falling edge and NCS rising edge;
- ncs_wr_hold—the NCS hold time is defined as the hold time of address after the NCS rising edge.

**Figure 35-13. Write Cycle**




### 35.9.3.3 Write Cycle

The write_cycle time is defined as the total duration of the write cycle; that is, from the time where address is set on the address bus to the point where address may change. The total write cycle time is defined as:

NWE_CYCLE = NWE_SETUP + NWE_PULSE + NWE_HOLD,

as well as

NWE_CYCLE = NCS_WR_SETUP + NCS_WR_PULSE + NCS_WR_HOLD

All NWE and NCS (write) timings are defined separately for each chip select as an integer number of Master Clock cycles. The NWE_CYCLE field is common to both the NWE and NCS signals, thus the timing period is of the same duration.

NWE_CYCLE, NWE_SETUP, and NWE_PULSE implicitly define the NWE_HOLD value as:

NWE_HOLD = NWE_CYCLE - NWE_SETUP - NWE_PULSE

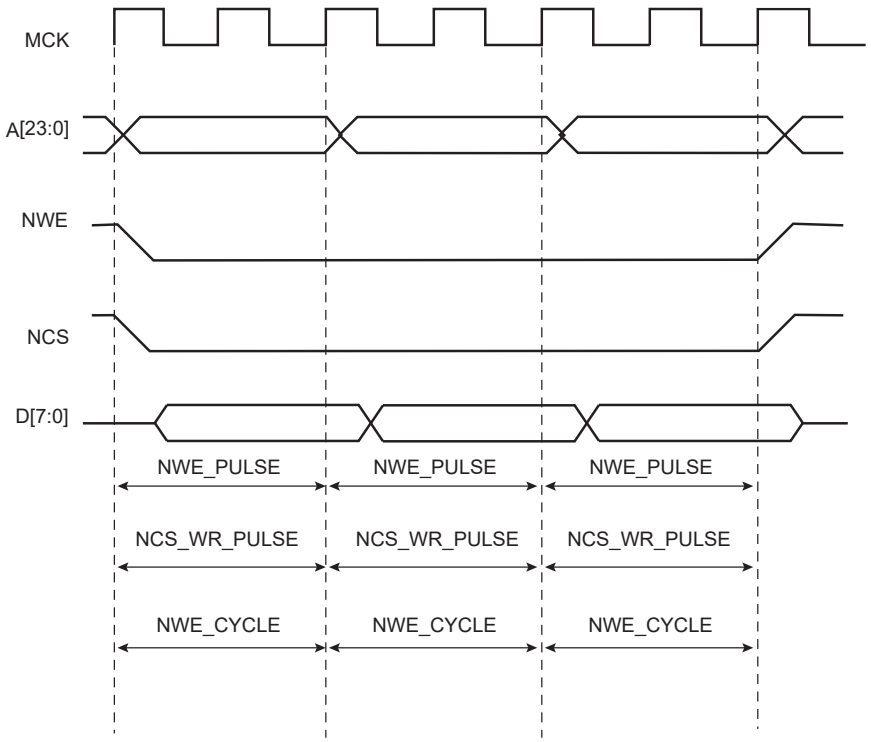
NWE_CYCLE, NCS_WR_SETUP, and NCS_WR_PULSE implicitly define the NCS_WR_HOLD value as:

NCS_WR_HOLD = NWE_CYCLE - NCS_WR_SETUP - NCS_WR_PULSE

### 35.9.3.4 Null Delay Setup and Hold

If null setup parameters are programmed for NWE and/or NCS, NWE and/or NCS remain active continuously in case of consecutive write cycles in the same memory (see the following figure). However, for devices that perform write operations on the rising edge of NWE or NCS, such as SRAM, either a setup or a hold must be programmed.

**Figure 35-14. Null Setup and Hold Values of NCS and NWE in Write Cycle**




#### 35.9.3.5 Null Pulse

Programming null pulse is not permitted. Pulse must be at least set to 1. A null value leads to unpredictable behavior.

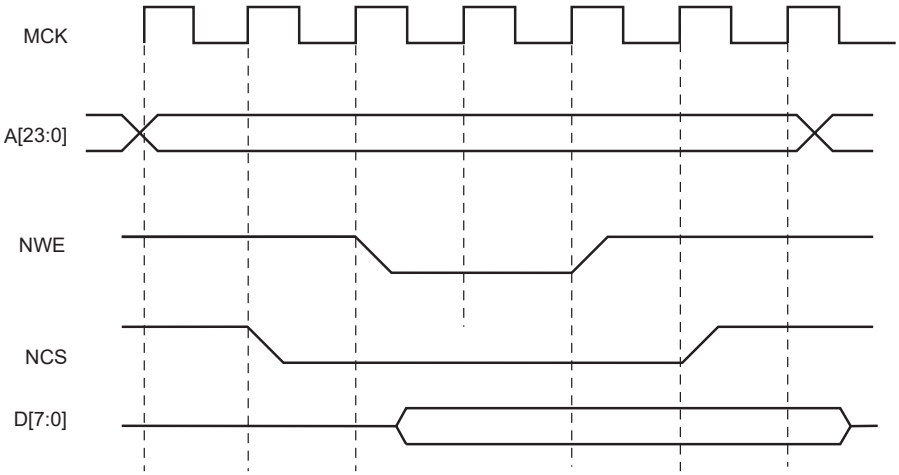
### 35.9.4 Write Mode

The bit WRITE_MODE in the SMC_MODE register of the corresponding chip select indicates which signal controls the write operation.

#### 35.9.4.1 Write is Controlled by NWE (SMC.MODE.WRITE_MODE = 1):

The following figure shows the waveforms of a write operation with SMC_MODE.WRITE_MODE set . The data is put on the bus during the pulse and hold steps of the NWE signal. The internal data buffers are switched to Output mode after the NWE_SETUP time, and until the end of the write cycle, regardless of the programmed waveform on NCS.

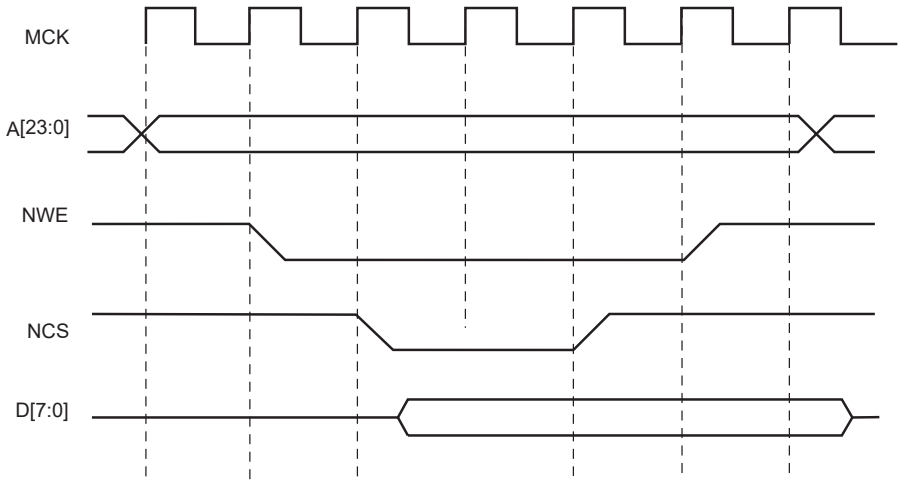
**Figure 35-15. SMC_MODE.WRITE_MODE = 1. Write Operation is Controlled by NWE**



#### 35.9.4.2 Write is Controlled by NCS (SMC.MODE.WRITE_MODE = 0)

The following figure shows the waveforms of a write operation with SMC_MODE.WRITE_MODE cleared. The data is put on the bus during the pulse and hold steps of the NCS signal. The internal data buffers are switched to Output mode after the NCS_WR_SETUP time, and until the end of the write cycle, regardless of the programmed waveform on NWE.

**Figure 35-16. WRITE_MODE = 0. Write Operation is Controlled by NCS**




### 35.9.5 Register Write Protection

To prevent any single software error that may corrupt SMC behavior, the registers listed below can be write-protected by setting the WPEN bit in the SMC Write Protection Mode register (SMC_WPMR).

If a write access in a write-protected register is detected, the WPVS flag in the SMC Write Protection Status register (SMC_WPSR) is set and the field WPVSRC indicates in which register the write access has been attempted.

The WPVS flag is automatically cleared after reading the SSMC_WPSR.

The following registers can be write-protected:

- "SMC Setup Register"
- "SMC Pulse Register"
- "SMC Cycle Register"
- "SMC Mode Register"
- "SMC Off-chip Memory Scrambling Register"

### 35.9.6 Coding Timing Parameters

All timing parameters are defined for one chip select and are grouped together in one register according to their type.

The SMC_SETUP register groups the definition of all setup parameters:

- NRD_SETUP
- NCS_RD_SETUP
- NWE_SETUP
- NCS_WR_SETUP

The SMC_PULSE register groups the definition of all pulse parameters:

- NRD_PULSE
- NCS_RD_PULSE
- NWE_PULSE
- NCS_WR_PULSE

The SMC_CYCLE register groups the definition of all cycle parameters:

- NRD_CYCLE

- NWE_CYCLE

The following table shows how the timing parameters are coded and their permitted range.

**Table 35-4. Coding and Range of Timing Parameters**

| Coded Value | Number of Bits | Effective Value | Permitted Range | |
|---|---|---|---|---|
| | | | Coded Value | Effective Value |
| setup [5:0] | 6 | 128 × setup[5] + setup[4:0] | 0 ≤ 31 | 0 ≤ 128+31 |
| pulse [6:0] | 7 | 256 × pulse[6] + pulse[5:0] | 0 ≤ 63 | 0 ≤ 256+63 |
| cycle [8:0] | 9 | 256 × cycle[8:7] + cycle[6:0] | 0 ≤ 127 | 0 ≤ 256+127<br>0 ≤ 512+127<br>0 ≤ 768+127 |

### 35.9.7 Reset Values of Timing Parameters

The following table provides the default value of timing parameters at reset.

**Table 35-5. Reset Values of Timing Parameters**

| Parameter | Reset Value | Definition |
|---|---|---|
| SMC_SETUP | 0x01010101 | All setup timings are set to 1. |
| SMC_PULSE | 0x01010101 | All pulse timings are set to 1. |
| SMC_CYCLE | 0x00030003 | The read and write operations continue for 3 Master Clock cycles and provide one hold cycle. |
| WRITE_MODE | 1 | Write is controlled with NWE. |
| READ_MODE | 1 | Read is controlled with NRD. |

### 35.9.8 Usage Restriction

The SMC does not check the validity of the user-programmed parameters. If the sum of SETUP and PULSE parameters is larger than the corresponding CYCLE parameter, this leads to unpredictable behavior of the SMC.

- For read operations:
  Null but positive setup and hold of address and NRD and/or NCS can not be guaranteed at the memory interface because of the propagation delay of theses signals through external logic and pads. If positive setup and hold values must be verified, then it is strictly recommended to program non-null values so as to cover possible skews between address, NCS and NRD signals.
- For write operations:
  If a null hold value is programmed on NWE, the SMC can guarantee a positive hold of address and NCS signal after the rising edge of NWE. This is true for SMC_MODE.WRITE_MODE = 1 only. See "Early Read Wait State".
- For read and write operations:
  A null value for pulse parameters is forbidden and may lead to unpredictable behavior.

  In read and write cycles, the setup and hold time parameters are defined in reference to the address bus. For external devices that require setup and hold time between NCS and NRD signals (read), or between NCS and NWE signals (write), these setup and hold times must be converted into setup and hold times in reference to the address bus.

## 35.10 Scrambling/Unscrambling Function

The external data bus can be scrambled to protect intellectual property data located in off-chip memories by means of data analysis at the package pin level of either the microcontroller or the memory device.

The scrambling and unscrambling are performed on-the-fly without additional wait states.

The scrambling/unscrambling function can be enabled or disabled by configuring the CSxSE bits in the SMC Off-Chip Memory Scrambling Register (SMC_OCMS).

When multiple chip selects are handled, the scrambling function per chip select is configurable using the CSxSE bits in the SMC_OCMS register.

The scrambling method depends on two user-configurable key registers, SMC_KEY1 and SMC_KEY2 plus a random value depending on device processing characteristics. These key registers cannot be read. They can be written once after a system reset.

The scrambling user key or the seed for key generation must be securely stored in a reliable non-volatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

## 35.11 Automatic Wait States

Under certain circumstances, the SMC automatically inserts idle cycles between accesses to avoid bus contention or operation conflict.

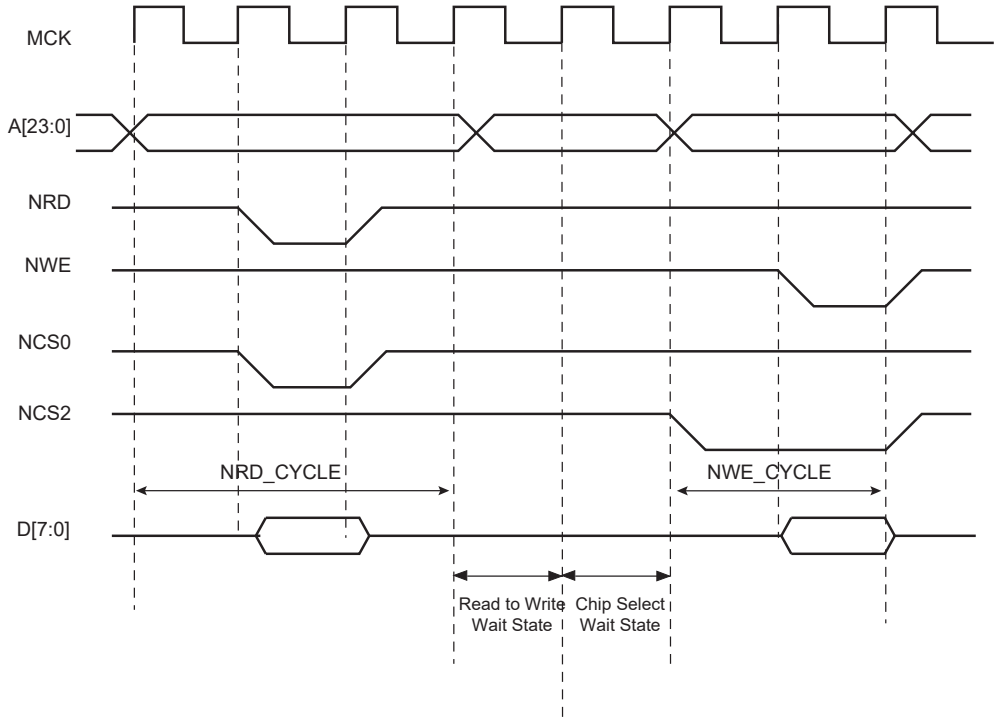
### 35.11.1 Chip Select Wait States

The SMC always inserts an idle cycle between two transfers on separate Chip Selects. This idle cycle ensures that there is no bus contention between the deactivation of one device and the activation of the next one.

During Chip Select Wait state, all control lines are turned inactive: NWR, NCS[0..3], NRD lines are all set to 1.

The following figure illustrates a Chip Select Wait state between access on Chip Select 0 and Chip Select 2.

**Figure 35-17. Chip Select Wait State between a Read Access on NCS0 and a Write Access on NCS2**




### 35.11.2 Early Read Wait State

In some cases, the SMC inserts a wait state cycle between a write access and a read access to allow time for the write cycle to end before the subsequent read cycle begins. This wait state is not generated in addition to a chip select wait state. The early read cycle thus only occurs between a write and read access to the same memory device (same chip select).

An early read wait state is automatically inserted if at least one of the following conditions is valid:

- if the write controlling signal has no hold time and the read controlling signal has no setup time (Figure 35-18).
- in NCS Write controlled mode (SMC_MODE.WRITE_MODE = 0), if there is no hold timing on the NCS signal and the NCS_RD_SETUP parameter is set to 0, regardless of the Read mode (Figure 35-19). The write operation must end with a NCS rising edge. Without an Early Read Wait State, the write operation could not complete properly.
- in NWE controlled mode (SMC_MODE.WRITE_MODE = 1) and if there is no hold timing (NWE_HOLD = 0), the feedback of the write control signal is used to control address, data, and chip select lines. If the external write control signal is not inactivated as expected due to load capacitances, an Early Read Wait State is inserted and address, data and control signals are maintained one more cycle. See Figure 35-20.

**Figure 35-18. Early Read Wait State: Write with No Hold Followed by Read with No Setup**

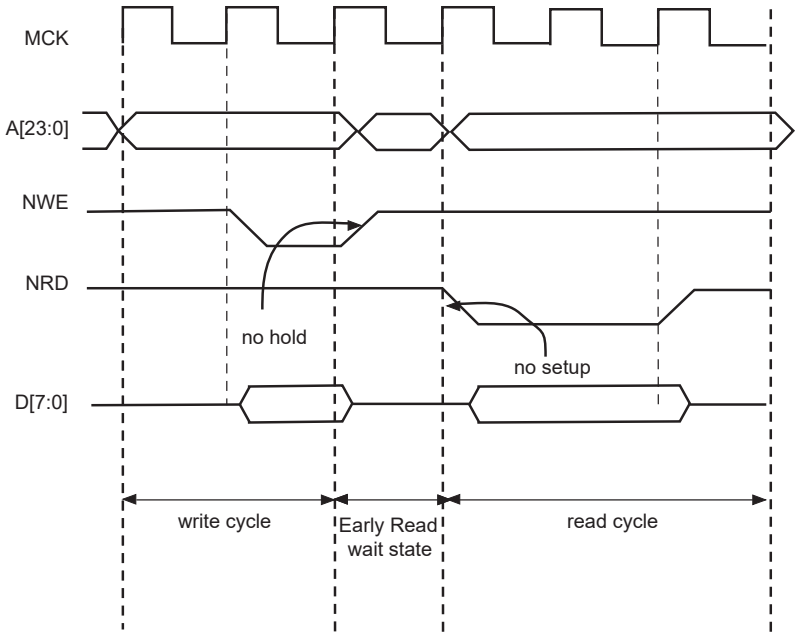



**Figure 35-19. Early Read Wait State: NCS-controlled write with no hold followed by a read with no NCS setup**

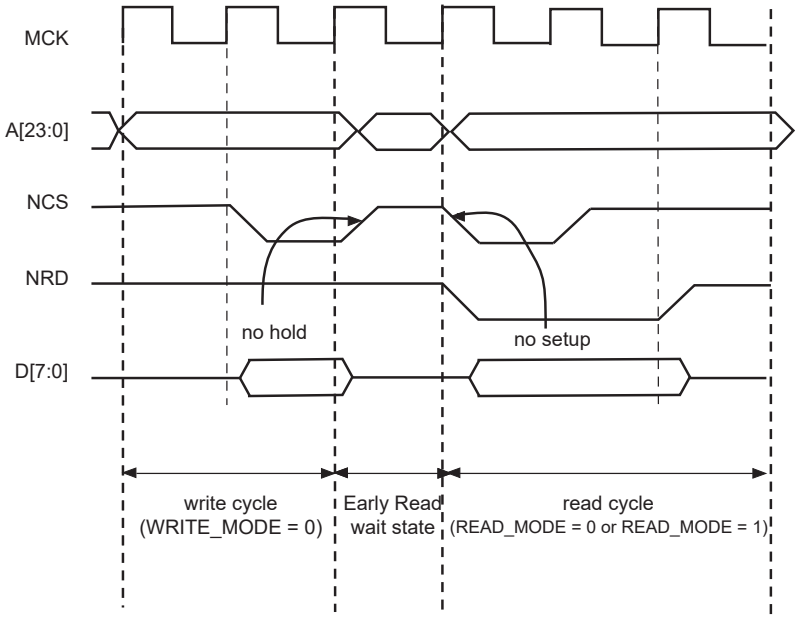
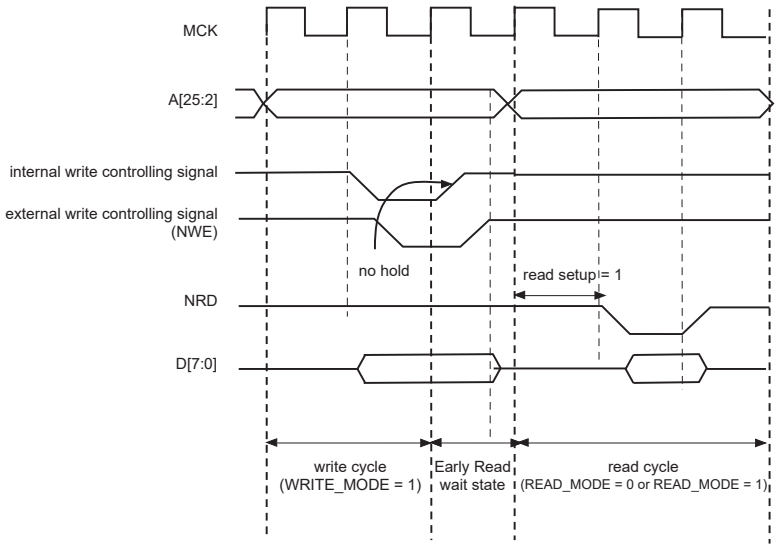


**Figure 35-20. Early Read Wait State: NWE-controlled write with no hold followed by a read with one set-up cycle**




### 35.11.3 Reload User Configuration Wait State

The user may change any of the configuration parameters by writing the SMC user interface.

When detecting that a new user configuration has been written in the user interface, the SMC inserts a wait state before starting the next access. This “reload user configuration wait state” is used by the SMC to load the new set of parameters to apply to next accesses.

The reload configuration wait state is not applied in addition to the chip select wait state. If accesses before and after re-programming the user interface are made to different devices (chip selects), then one single chip select wait state is applied.

On the other hand, if accesses before and after writing the user interface are made to the same device, a reload configuration wait state is inserted, even if the change does not concern the current chip select.

#### 35.11.3.1 User Procedure

To insert a reload configuration wait state, the SMC detects a write access to any SMC_MODE register of the user interface. If the user only modifies timing registers (SMC_SETUP, SMC_PULSE, SMC_CYCLE registers) in the user interface, he must validate the modification by writing the SMC_MODE, even if no change was made on the mode parameters.

The user must not change the configuration parameters of an SMC chip select (Setup, Pulse, Cycle, Mode) if accesses are performed on this CS during the modification. Any change of the chip select parameters, while fetching the code from a memory connected on this CS, may lead to unpredictable behavior. The instructions used to modify the parameters of an SMC chip select can be executed from the internal RAM or from a memory connected to another CS.

#### 35.11.3.2 Slow Clock Mode Transition

A reload configuration wait state is also inserted when the Slow Clock mode is entered or exited, after the end of the current transfer (see ”Slow Clock Mode”).

### 35.11.4 Read to Write Wait State

Due to an internal mechanism, a wait cycle is always inserted between consecutive read and write SMC accesses.

This wait cycle is referred to as a read to write wait state in this document.

This wait cycle is applied in addition to chip select and reload user configuration wait states when they are to be inserted. See Figure 12-1.

## 35.12 Data Float Wait States

Some memory devices are slow to release the external bus. For such devices, it is necessary to add wait states (data float wait states) after a read access:

- before starting a read access to a different external memory
- before starting a write access to the same device or to a different external one.

The data float output time ($t_{DF}$) for each external memory device is programmed in the SMC_MODE.TDF_CYCLES field for the corresponding chip select. The value of SMC_MODE.TDF_CYCLES indicates the number of data float wait cycles (between 0 and 15) before the external device releases the bus, and represents the time allowed for the data output to go to high impedance after the memory is disabled.

Data float wait states do not delay internal memory accesses. Hence, a single access to an external memory with long $t_{DF}$ will not slow down the execution of a program from internal memory.

The data float wait states management depends on SMC_MODE.READ_MODE and the SMC_MODE.TDF_MODE fields for the corresponding chip select.

### 35.12.1 SMC_MODE.READ_MODE

Setting SMC_MODE.READ_MODE to 1 indicates to the SMC that the NRD signal is responsible for turning off the tri-state buffers of the external memory device. The Data Float Period then begins after the rising edge of the NRD signal and lasts SMC_MODE.TDF_CYCLES MCK cycles.

When the read operation is controlled by the NCS signal (SMC_MODE.READ_MODE = 0), the TDF field gives the number of MCK cycles during which the data bus remains busy after the rising edge of NCS.

Figure 35-21 illustrates the Data Float Period in NRD-controlled mode (SMC_MODE.READ_MODE =1), assuming a data float period of 2 cycles (SMC_MODE.TDF_CYCLES = 2). Figure 35-22 shows the read operation when controlled by NCS (SMC_MODE.READ_MODE = 0) and SMC_MODE.TDF_CYCLES = 3.

**Figure 35-21. TDF Period in NRD Controlled Read Access (TDF = 2)**

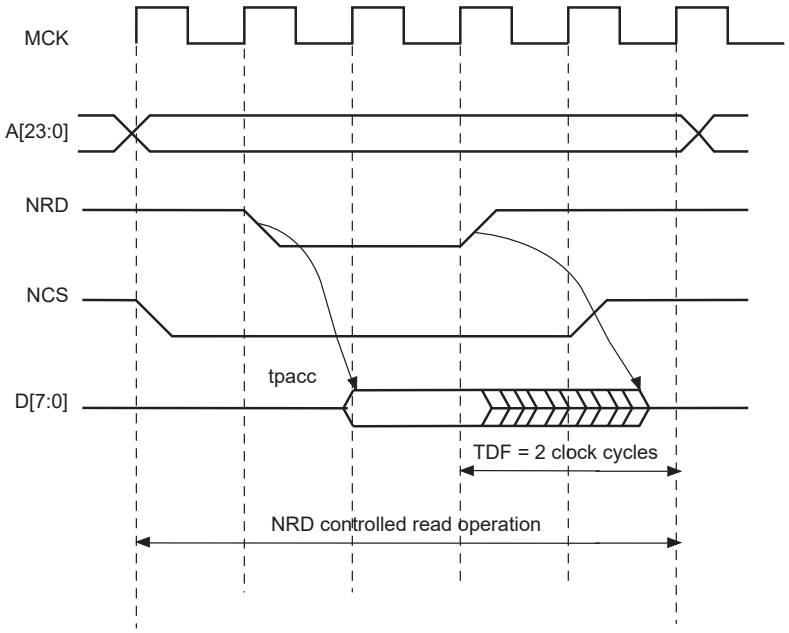
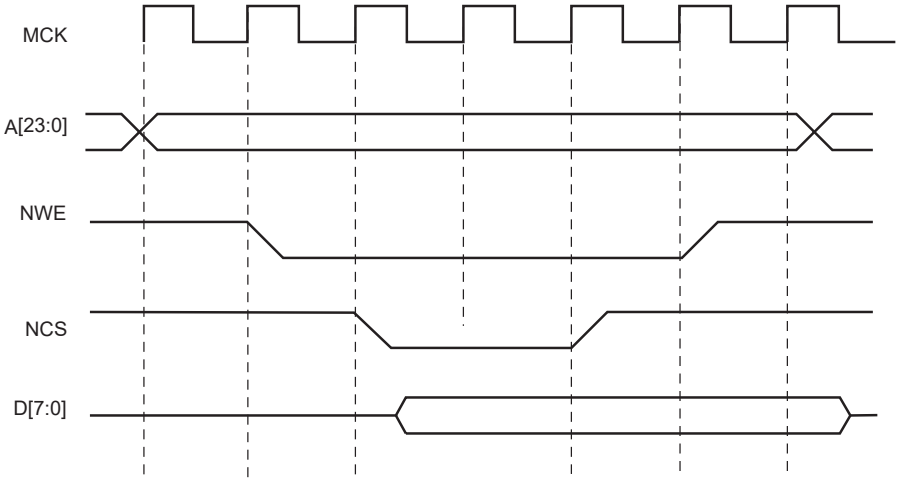


**Figure 35-22. TDF Period in NCS Controlled Read Operation (TDF = 3)**




### 35.12.2 TDF Optimization Enabled (SMC_MODE.TDF_MODE = 1)

When SMC_MODE.TDF_MODE is set to 1 (TDF optimization is enabled), the SMC takes advantage of the setup period of the next access to optimize the number of wait states cycle to insert.

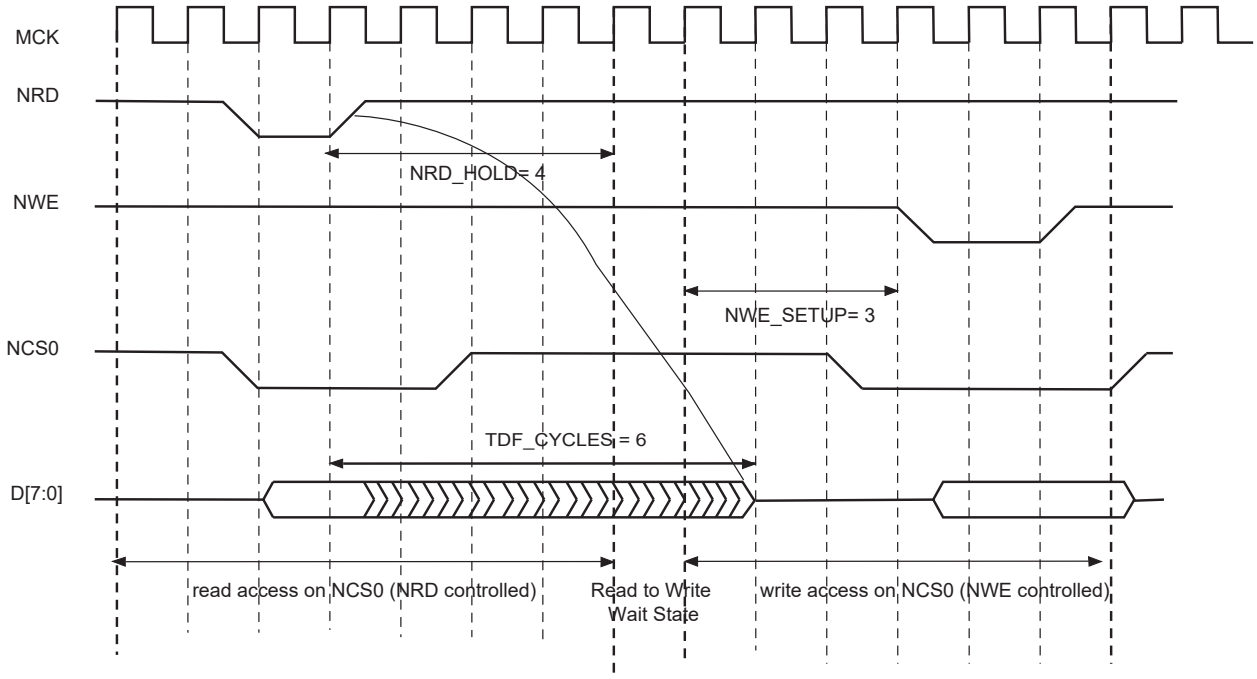
The following figure shows a read access controlled by NRD, followed by a write access controlled by NWE, on Chip Select 0. Chip Select 0 has been programmed with:

nrd_hold = 4; SMC_MODE.read_mode = 1 (NRD controlled)

nwe_setup = 3; SMC_MODE.write_mode = 1 (NWE controlled)

SMC_MODE.TDF_CYCLES = 6; SMC_MODE.TDF_MODE = 1 (optimization enabled).

**Figure 35-23. TDF Optimization: No TDF wait states are inserted if the TDF period is over when the next access begins**



### 35.12.3 TDF Optimization Disabled (SMC_MODE.TDF_MODE = 0)

When optimization is disabled, TDF Wait states are inserted at the end of the read transfer, so that the data float period is ended when the second access begins. If the hold period of the read1 controlling signal overlaps the data float period, no additional TDF Wait states will be inserted.

Figure 35-24, Figure 35-25 and Figure 35-26 illustrate the cases:

- read access followed by a read access on another Chip Select,
- read access followed by a write access on another Chip Select,
- read access followed by a write access on the same Chip Select,

with no TDF optimization.

**Figure 35-24. TDF Optimization Disabled (TDF Mode = 0): TDF wait states between 2 read accesses on different chip selects**

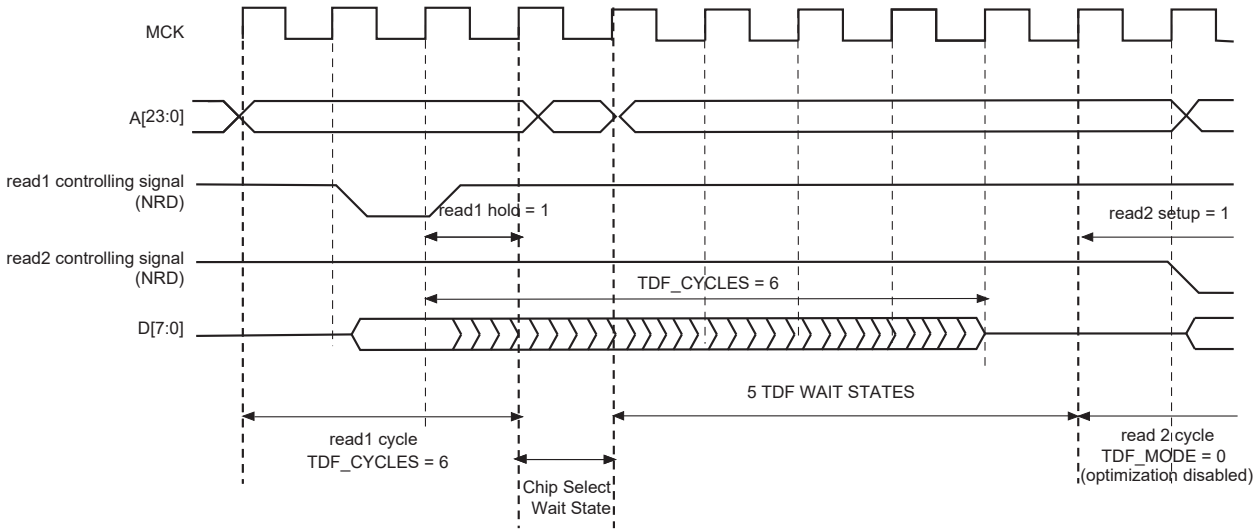



**Figure 35-25. TDF Mode = 0: TDF wait states between a read and a write access on different chip selects**

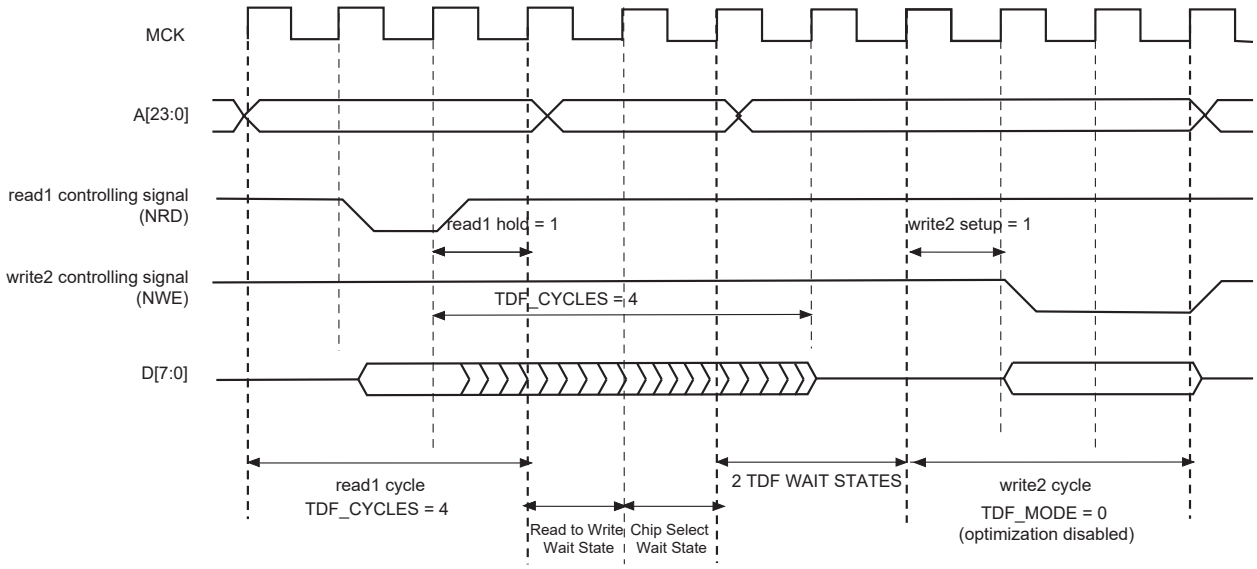
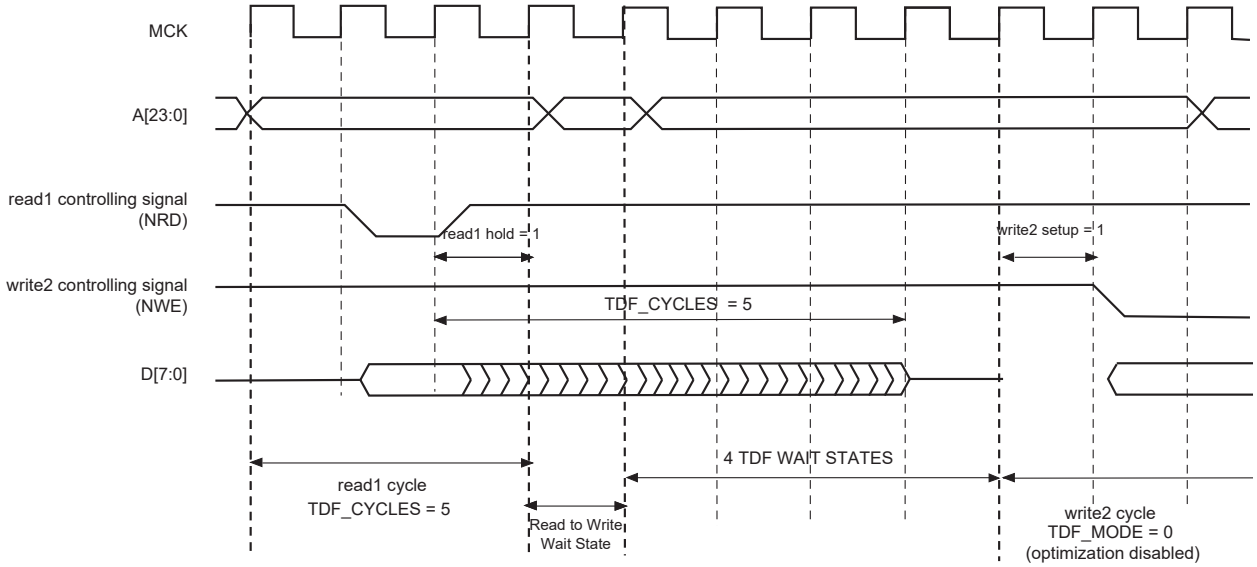


**Figure 35-26. TDF Mode = 0: TDF wait states between read and write accesses on the same chip select**




## 35.13 External Wait

Any access can be extended by an external device using the NWAIT input signal of the SMC. The SMC_MODE.EXNW_MODE field on the corresponding chip select must be set either to “10” (Frozen mode) or “11” (Ready mode). When SMC_MODE.EXNW_MODE is set to “00” (disabled), the NWAIT signal is simply ignored on the corresponding chip select. The NWAIT signal delays the read or write operation in regards to the read or write controlling signal, depending on the Read and Write modes of the corresponding chip select.

### 35.13.1 Restriction

When SMC_MODE.EXNW_MODE is enabled, it is mandatory to program at least one hold cycle for the read/write controlling signal. For that reason, the NWAIT signal cannot be used in Page mode (35.15 Asynchronous Page Mode), or in Slow clock mode (”Slow Clock Mode”).

The NWAIT signal is assumed to be a response of the external device to the read/write request of the SMC. Then NWAIT is examined by the SMC only in the pulse state of the read or write controlling signal. The assertion of the NWAIT signal outside the expected period has no impact on SMC behavior.

### 35.13.2 Frozen Mode

When the external device asserts the NWAIT signal (active low), and after internal synchronization of this signal, the SMC state is frozen, i.e., SMC internal counters are frozen, and all control signals remain unchanged. When the resynchronized NWAIT signal is deasserted, the SMC completes the access, resuming the access from the point where it was stopped. See Figure 35-27. This mode must be selected when the external device uses the NWAIT signal to delay the access and to freeze the SMC.

The assertion of the NWAIT signal outside the expected period is ignored as illustrated in Figure 35-28.

**Figure 35-27. Write Access with NWAIT Assertion in Frozen Mode (SMC_MODE.EXNW_MODE = 10)**

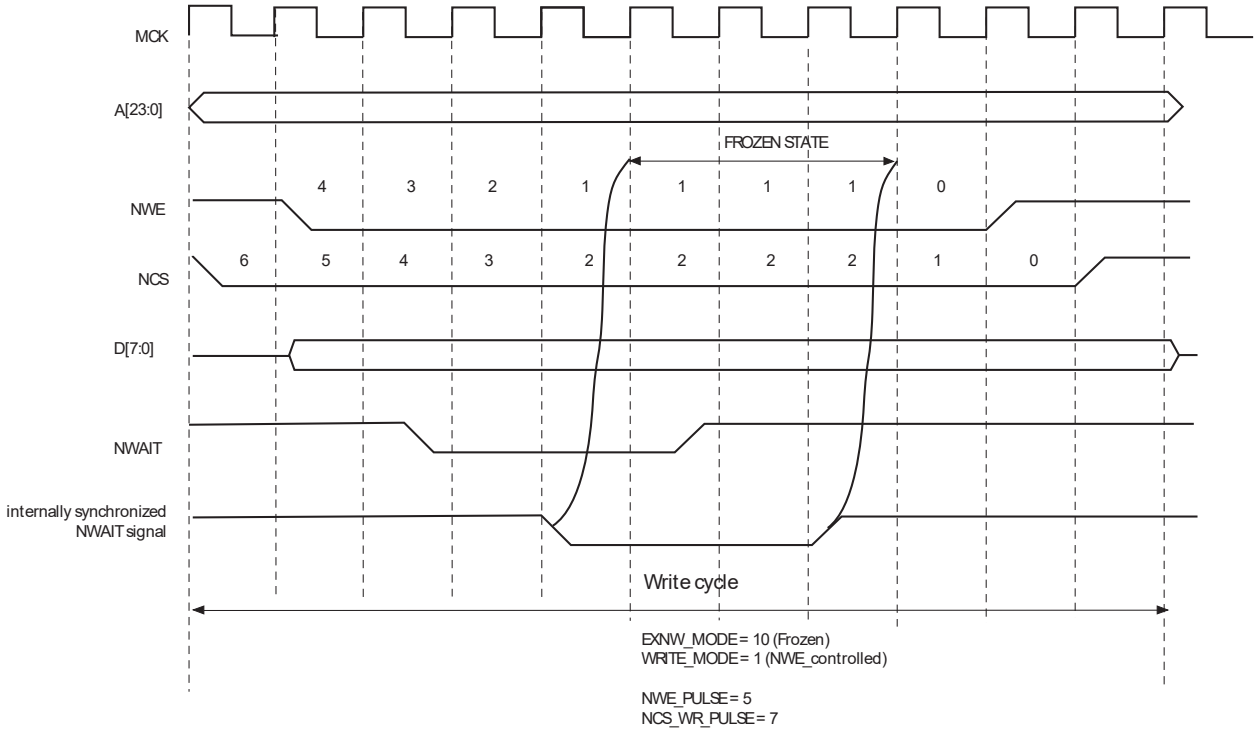
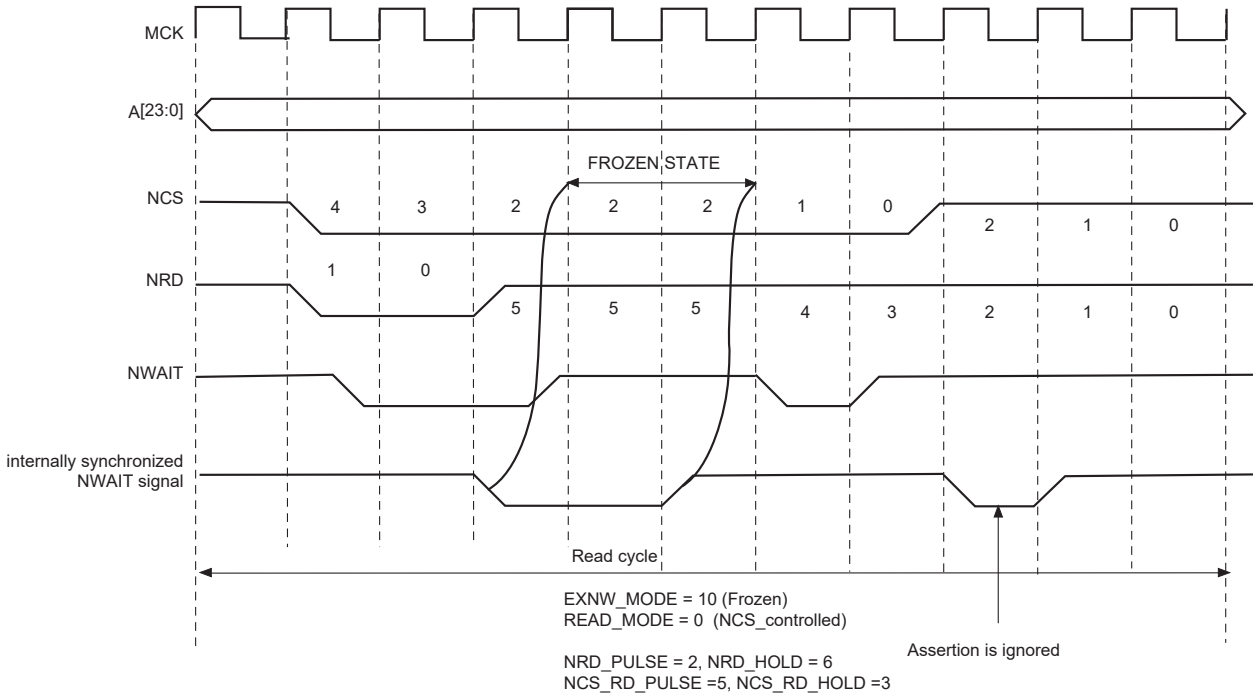



**Figure 35-28. Read Access with NWAIT Assertion in Frozen Mode (SMC_MODE.EXNW_MODE = 10)**




### 35.13.3 Ready Mode

In Ready mode (SMC_MODE.EXNW_MODE = 11), the SMC behaves differently. Normally, the SMC begins the access by down counting the setup and pulse counters of the read/write controlling signal. In the last cycle of the pulse phase, the resynchronized NWAIT signal is examined.

If asserted, the SMC suspends the access as shown in Figure 35-29 and Figure 35-30. After deassertion, the access is completed: the hold step of the access is performed.

This mode must be selected when the external device uses deassertion of the NWAIT signal to indicate its ability to complete the read or write operation.

If the NWAIT signal is deasserted before the end of the pulse, or asserted after the end of the pulse of the controlling read/write signal, it has no impact on the access length as shown in Figure 35-30.

**Figure 35-29. NWAIT Assertion in Write Access: Ready Mode (SMC_MODE.EXNW_MODE = 11)**

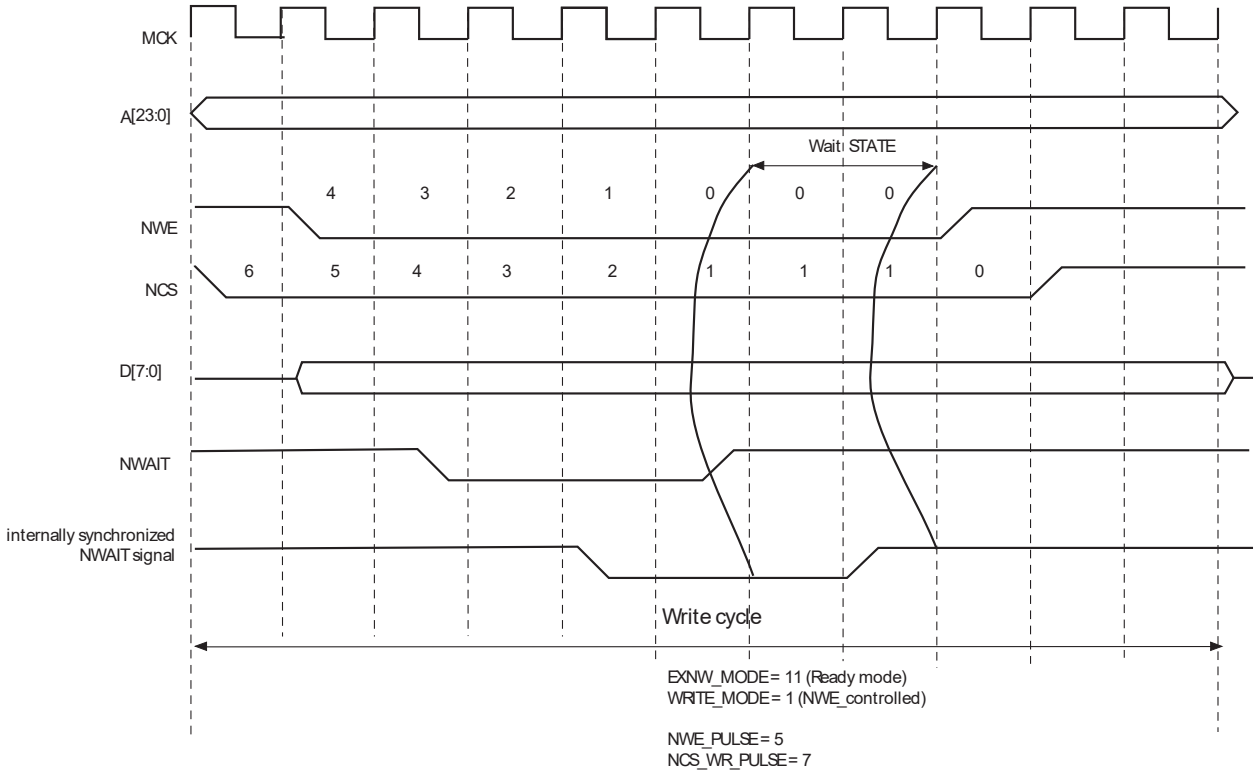
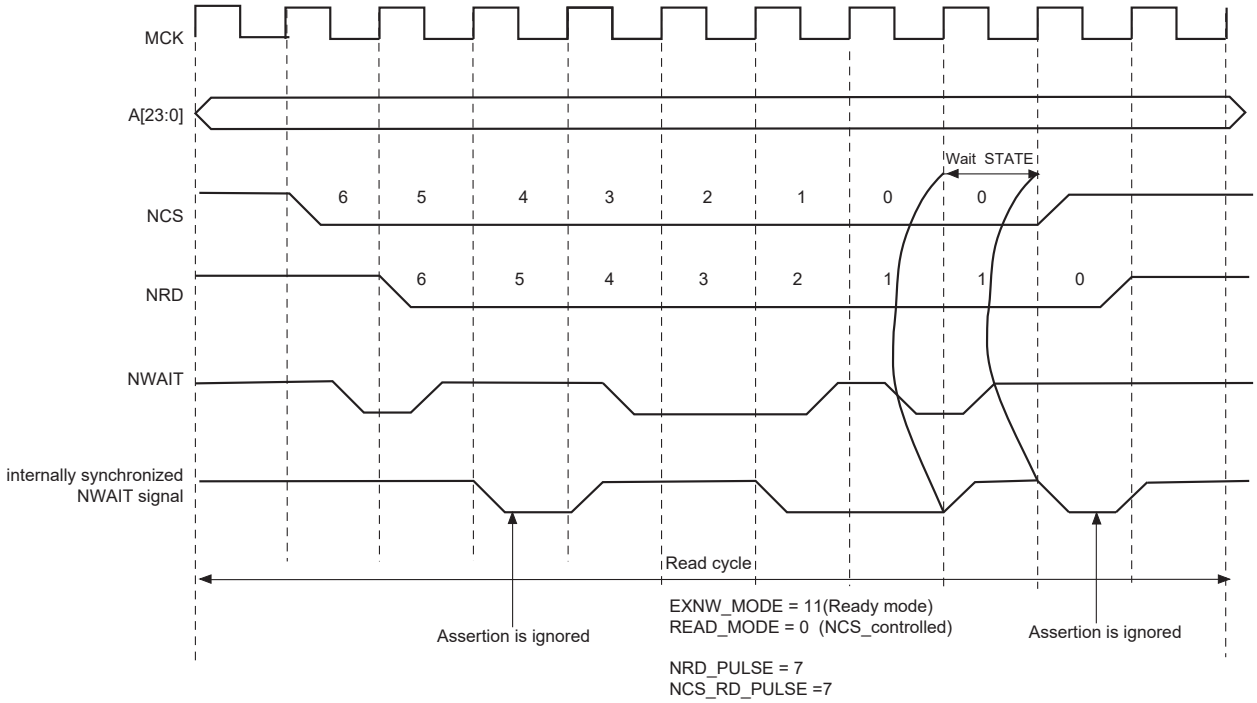


**Figure 35-30. NWAIT Assertion in Read Access: Ready Mode (SMC_MODE.EXNW_MODE = 11)**



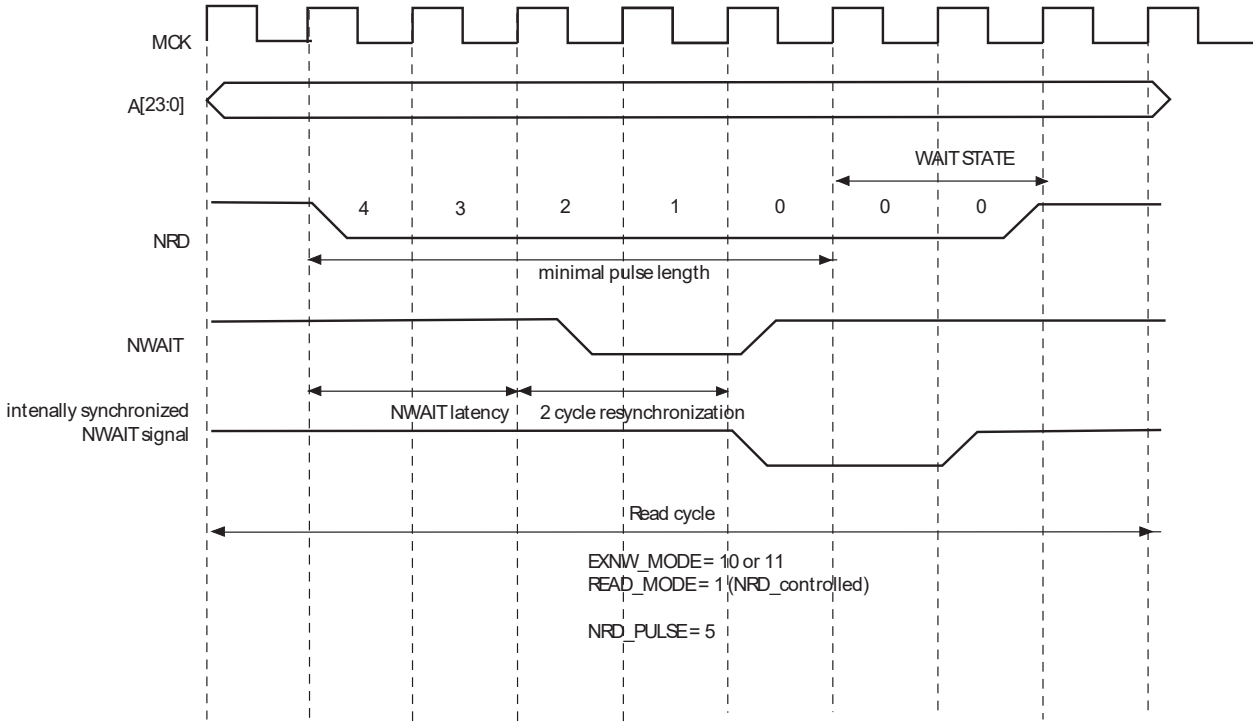

### 35.13.4 NWAIT Latency and Read/Write Timings

There may be a latency between the assertion of the read/write controlling signal and the assertion of the NWAIT signal by the device. The programmed pulse length of the read/write controlling signal must be at least equal to this latency plus the 2 cycles of resynchronization + one cycle. Otherwise, the SMC may enter the hold state of the access without detecting the NWAIT signal assertion. This is true in Frozen mode as well as in Ready mode. This is illustrated in the following figure.

When SMC_MODE.EXNW_MODE is enabled (ready or frozen), the user must program a pulse length of the read and write controlling signal of at least:

Minimal pulse length = NWAIT latency + 2 resynchronization cycles + 1 cycle

**Figure 35-31. NWAIT Latency**




## 35.14 Slow Clock Mode

The SMC is able to automatically apply a set of “Slow clock mode” read/write waveforms when an internal signal driven by the Power Management Controller is asserted because MCK has been turned to a very slow clock rate (typically 32kHz clock rate). In this mode, the user-programmed waveforms are ignored and the Slow clock mode waveforms are applied. This mode is provided so as to avoid reprogramming the User Interface with appropriate waveforms at a very slow clock rate. When activated, the Slow clock mode is active on all chip selects.

### 35.14.1 Slow Clock Mode Waveforms

Figure 35-32 illustrates the read and write operations in Slow Clock mode. They are valid on all Chip Selects. Table 35-6 indicates the value of read and write parameters in Slow Clock mode.

**Figure 35-32. Read/Write Cycles in Slow Clock Mode**

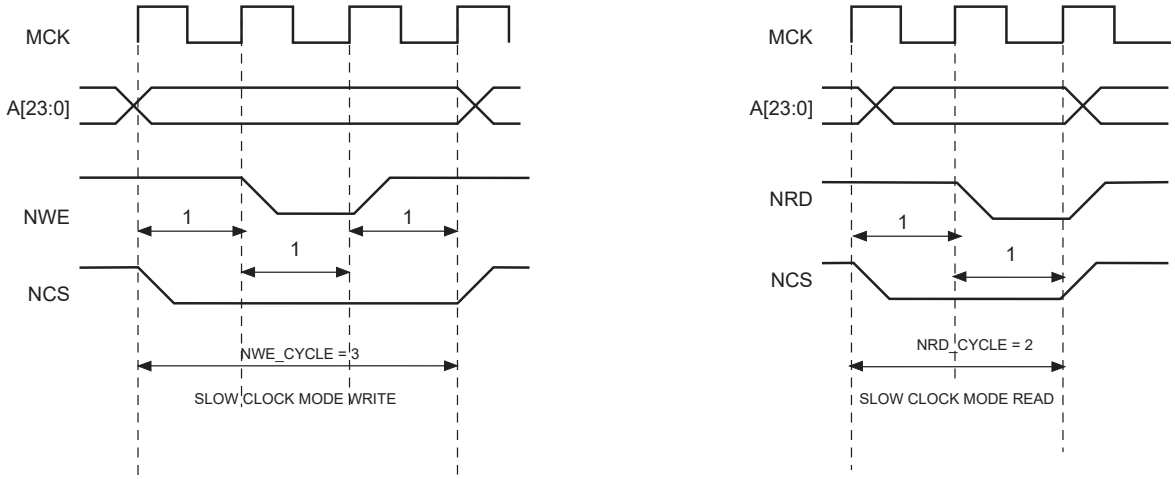


**Table 35-6. Read and Write Timing Parameters in Slow Clock Mode**

| Read Parameters | Duration (cycles) | Write Parameters | Duration (cycles) |
|---|---|---|---|
| NRD_SETUP | 1 | NWE_SETUP | 1 |
| NRD_PULSE | 1 | NWE_PULSE | 1 |
| NCS_RD_SETUP | 0 | NCS_WR_SETUP | 0 |
| NCS_RD_PULSE | 2 | NCS_WR_PULSE | 3 |
| NRD_CYCLE | 2 | NWE_CYCLE | 3 |

### 35.14.2 Switching from (to) Slow Clock Mode to (from) Normal Mode

When switching from Slow clock mode to Normal mode, the current Slow clock mode transfer is completed at a high clock rate, with the set of Slow clock mode parameters (see Figure 35-33). The external device may not be fast enough to support such timings.

Figure 35-34 illustrates the recommended procedure to switch from one mode to the other.

**Figure 35-33. Clock Rate Transition Occurs while the SMC is Performing a Write Operation**

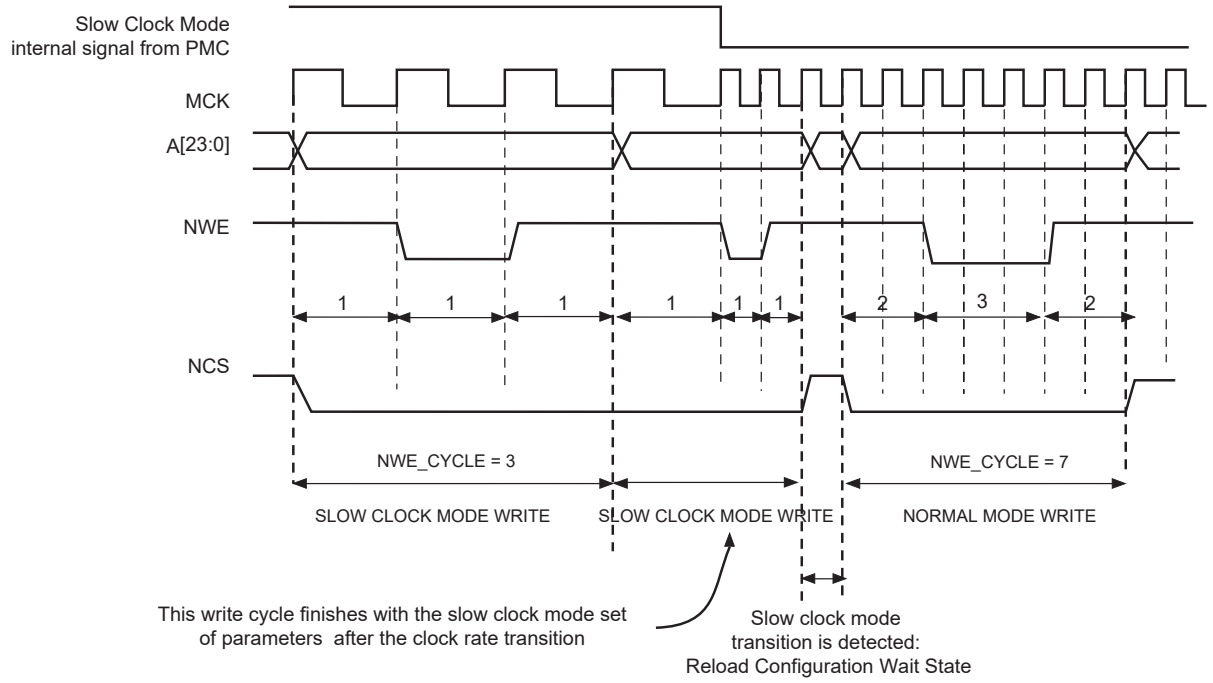
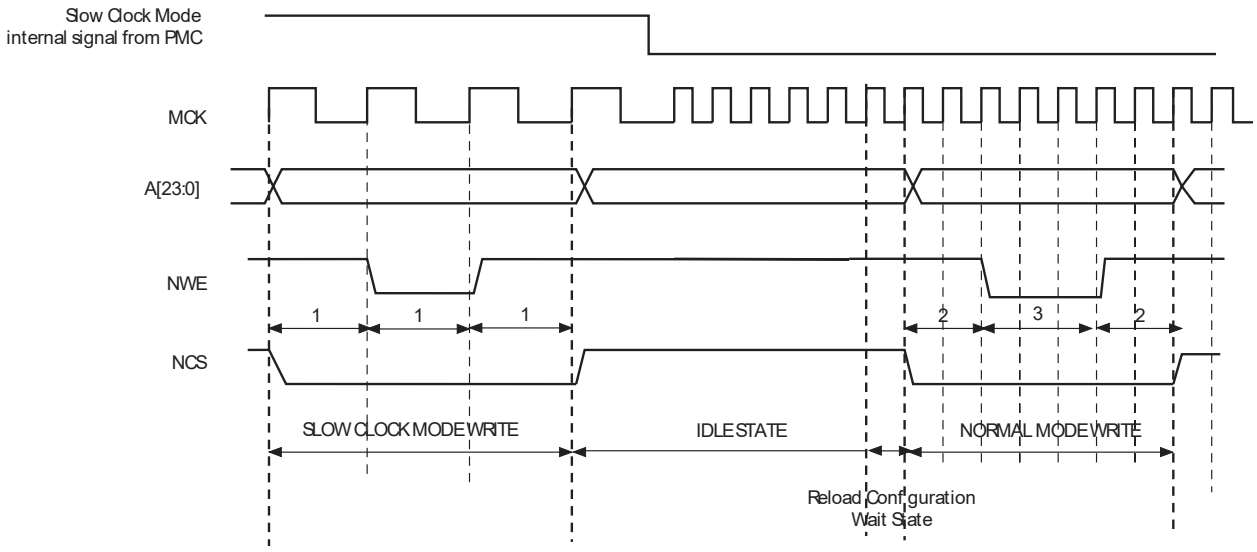


**Figure 35-34. Recommended Procedure to Switch from Slow Clock Mode to Normal Mode or from Normal Mode to Slow Clock Mode**




## 35.15 Asynchronous Page Mode

The SMC supports asynchronous burst reads in Page mode, provided that the Page mode is enabled (SMC_MODE.PMEN =1). The page size must be configured in the SMC_MODE register (PS field) to 4, 8, 16 or 32 bytes.

The page defines a set of consecutive bytes into memory. A 4-byte page (resp. 8-, 16-, 32-byte page) is always aligned to 4-byte boundaries (resp. 8-, 16-, 32-byte boundaries) of memory. The MSB of data address defines the address of the page in memory, the LSB of address define the address of the data in the page as detailed in the following table.

With Page mode memory devices, the first access to one page ($t_{pa}$) takes longer than the subsequent accesses to the page ($t_{sa}$) as shown in Page Mode Read Protocol. When in Page mode, the SMC enables the user to define different read timings for the first access within one page, and next accesses within the page.

**Table 35-7. Page Address and Data Address within a Page**

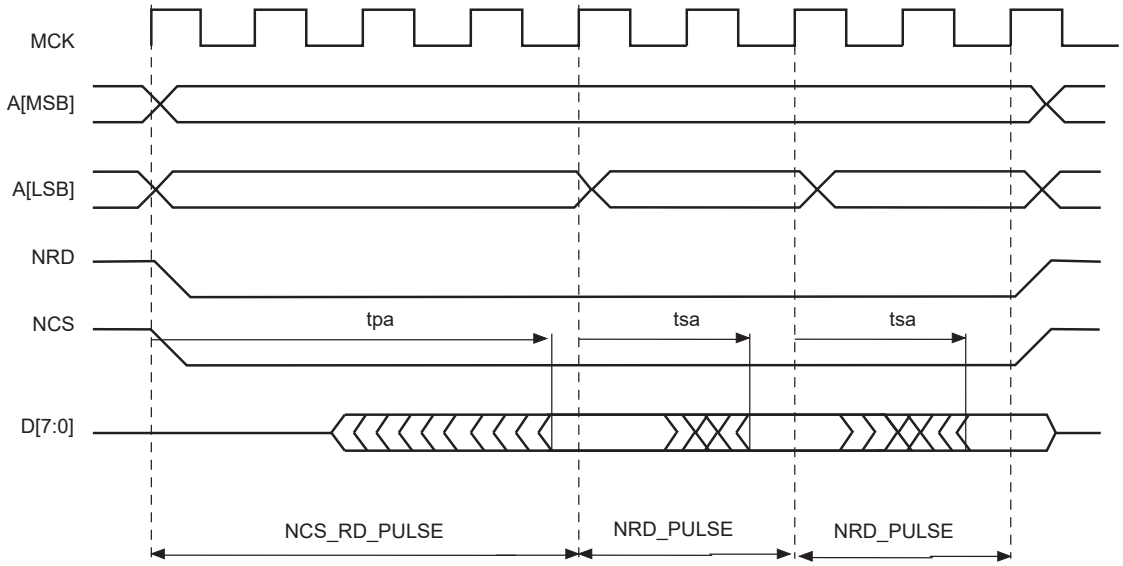
| Page Size | Page Address (see Note) | Data Address in the Page |
|---|---|---|
| 4 bytes | A[23:2] | A[1:0] |
| 8 bytes | A[23:3] | A[2:0] |
| 16 bytes | A[23:4] | A[3:0] |
| 32 bytes | A[23:5] | A[4:0] |

**Note:** “A” denotes the address bus of the memory device.

### 35.15.1 Protocol and Timings in Page Mode

The following figure shows the NRD and NCS timings in Page mode access.

**Figure 35-35. Page Mode Read Protocol (Address MSB and LSB are defined in Table 35-7)**




The NRD and NCS signals are held low during all read transfers, whatever the programmed values of the setup and hold timings in the User Interface may be. Moreover, the NRD and NCS timings are identical. The pulse length of the first access to the page is defined with the NCS_RD_PULSE field of the SMC_PULSE register. The pulse length of subsequent accesses within the page are defined using the NRD_PULSE parameter.

In Page mode, the programming of the read timings is described in the following table:

**Table 35-8. Programming of Read Timings in Page Mode**

| Parameter | Value | Definition |
|---|---|---|
| READ_MODE | 'x' | No impact. |
| NCS_RD_SETUP | 'x' | No impact. |
| NCS_RD_PULSE | $t_{pa}$ | Access time of first access to the page. |
| NRD_SETUP | 'x' | No impact. |
| NRD_PULSE | $t_{sa}$ | Access time of subsequent accesses in the page. |
| NRD_CYCLE | 'x' | No impact. |

The SMC does not check the coherency of timings. It will always apply the NCS_RD_PULSE timings as page access timing ($t_{pa}$) and the NRD_PULSE for accesses to the page ($t_{sa}$), even if the programmed value for $t_{pa}$ is shorter than the programmed value for $t_{sa}$.

### 35.15.2 Page Mode Restriction

The Page mode is not compatible with the use of the NWAIT signal. Using the Page mode and the NWAIT signal may lead to unpredictable behavior.

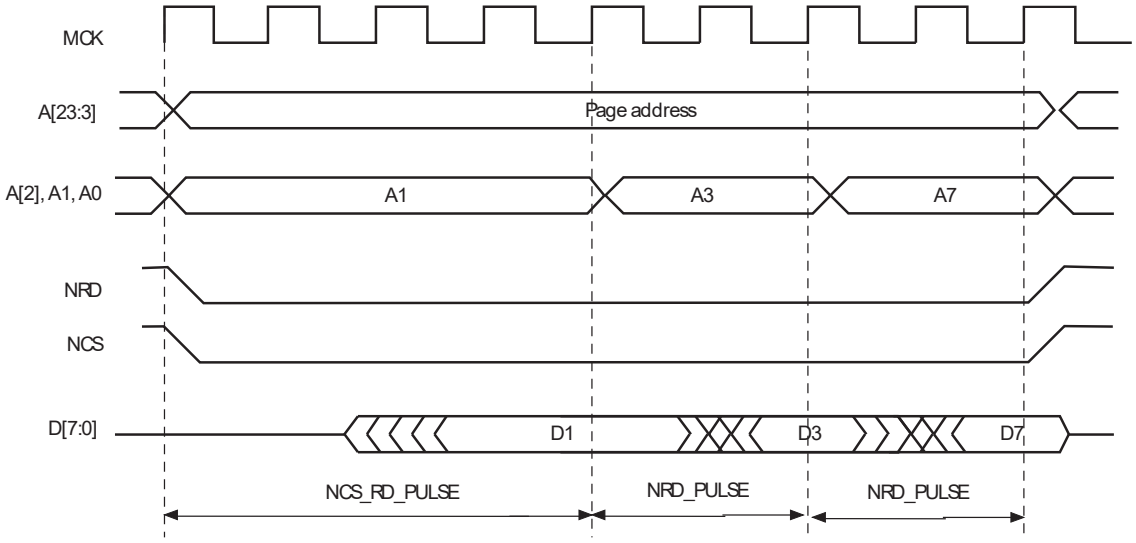
### 35.15.3 Sequential and Non-sequential Accesses

If the chip select and the MSB of addresses as defined in Table 35-7 are identical, then the current access lies in the same page as the previous one, and no page break occurs.

Using this information, all data within the same page, sequential or not sequential, are accessed with a minimum access time ($t_{sa}$). The following figure illustrates access to an 8-bit memory device in Page mode, with 8-byte pages. Access to D1 causes a page access with a long access time ($t_{pa}$). Accesses to D3 and D7, though they are not sequential accesses, only require a short access time ($t_{sa}$).

If the MSB of addresses are different, the SMC performs the access of a new page. In the same way, if the chip select is different from the previous access, a page break occurs. If two sequential accesses are made to the Page mode memory, but separated by an other internal or external peripheral access, a page break occurs on the second access because the chip select of the device was deasserted between both accesses.

**Figure 35-36. Access to Non-Sequential Data within the Same Page**



## 35.16 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SMC_SETUP[0..3] | 7:0 | | | NWE_SETUP[5:0] | | | | | |
| | | 15:8 | | | NCS_WR_SETUP[5:0] | | | | | |
| | | 23:16 | | | NRD_SETUP[5:0] | | | | | |
| | | 31:24 | | | NCS_RD_SETUP[5:0] | | | | | |
| 0x00 | SMC_PULSE[0..3] | 7:0 | | NWE_PULSE[6:0] | | | | | | |
| | | 15:8 | | NCS_WR_PULSE[6:0] | | | | | | |
| | | 23:16 | | NRD_PULSE[6:0] | | | | | | |
| | | 31:24 | | NCS_RD_PULSE[6:0] | | | | | | |
| 0x00 | SMC_CYCLE[0..3] | 7:0 | NWE_CYCLE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | NWE_CYCLE[8] |
| | | 23:16 | NRD_CYCLE[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | NRD_CYCLE[8] |
| 0x00 | SMC_MODE[0..3] | 7:0 | | | EXNW_MODE[1:0] | | | | WRITE_MODE | READ_MODE |
| | | 15:8 | | | | DBW | | | | BAT |
| | | 23:16 | | | | TDF_MODE | TDF_CYCLES[3:0] | | | |
| | | 31:24 | | | PS[1:0] | | | | | PMEN |
| 0x04 ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | SMC_OCMS | 7:0 | | | | | | | | SMSE |
| | | 15:8 | | | | | CS3SE | CS2SE | CS1SE | CS0SE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 | SMC_KEY1 | 7:0 | KEY1[7:0] | | | | | | | |
| | | 15:8 | KEY1[15:8] | | | | | | | |
| | | 23:16 | KEY1[23:16] | | | | | | | |
| | | 31:24 | KEY1[31:24] | | | | | | | |
| 0x88 | SMC_KEY2 | 7:0 | KEY2[7:0] | | | | | | | |
| | | 15:8 | KEY2[15:8] | | | | | | | |
| | | 23:16 | KEY2[23:16] | | | | | | | |
| | | 31:24 | KEY2[31:24] | | | | | | | |
| 0x8C ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | SMC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | SMC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 35.16.1 Static Memory Controller (SMC) User Interface

The SMC is programmed using the registers listed in the following table. For each Chip Select, a set of four registers is used to program the parameters of the external device connected on it. In the Register Summary, “CS_number” denotes the Chip Select number. 16 bytes (0x10) are required per Chip Select.

### 35.16.1.1 SMC Setup Register

**Name:** SMC_SETUP[0..3]
**Offset:** 0x00
**Reset:** 0
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the "SMC Write Protection Mode Register" .

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | NCS_RD_SETUP[5:0] | | | | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | NRD_SETUP[5:0] | | | | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NCS_WR_SETUP[5:0] | | | | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | NWE_SETUP[5:0] | | | | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:24 – NCS_RD_SETUP[5:0]** NCS Setup Length in READ Access
In read access, the NCS signal setup length is defined as:
NCS setup length = (128* NCS_RD_SETUP[5] + NCS_RD_SETUP[4:0]) clock cycles

**Bits 21:16 – NRD_SETUP[5:0]** NRD Setup Length
The NRD signal setup length is defined in clock cycles as:
NRD setup length = (128* NRD_SETUP[5] + NRD_SETUP[4:0]) clock cycles

**Bits 13:8 – NCS_WR_SETUP[5:0]** NCS Setup Length in WRITE Access
In write access, the NCS signal setup length is defined as:
NCS setup length = (128* NCS_WR_SETUP[5] + NCS_WR_SETUP[4:0]) clock cycles

**Bits 5:0 – NWE_SETUP[5:0]** NWE Setup Length
The NWE signal setup length is defined as:
NWE setup length = (128* NWE_SETUP[5] + NWE_SETUP[4:0]) clock cycles

### 35.16.1.2 SMC Pulse Register

**Name:** SMC_PULSE[0..3]
**Offset:** 0x00
**Reset:** 0
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the "SMC Write Protection Mode Register" .

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | NCS_RD_PULSE[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | NRD_PULSE[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | NCS_WR_PULSE[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | NWE_PULSE[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:24 – NCS_RD_PULSE[6:0]** NCS Pulse Length in READ Access
In standard read access, the NCS signal pulse length is defined as:
NCS pulse length = (256* NCS_RD_PULSE[6] + NCS_RD_PULSE[5:0]) clock cycles
The NCS pulse length must be at least 1 clock cycle.
In Page mode read access, the NCS_RD_PULSE parameter defines the duration of the first access to one page.

**Bits 22:16 – NRD_PULSE[6:0]** NRD Pulse Length
In standard read access, the NRD signal pulse length is defined in clock cycles as:
NRD pulse length = (256* NRD_PULSE[6] + NRD_PULSE[5:0]) clock cycles
The NRD pulse length must be at least 1 clock cycle.
In Page mode read access, the NRD_PULSE parameter defines the duration of the subsequent accesses in the page.

**Bits 14:8 – NCS_WR_PULSE[6:0]** NCS Pulse Length in WRITE Access
In write access, the NCS signal pulse length is defined as:
NCS pulse length = (256* NCS_WR_PULSE[6] + NCS_WR_PULSE[5:0]) clock cycles
The NCS pulse length must be at least 1 clock cycle.

**Bits 6:0 – NWE_PULSE[6:0]** NWE Pulse Length
The NWE signal pulse length is defined as:
NWE pulse length = (256* NWE_PULSE[6] + NWE_PULSE[5:0]) clock cycles
The NWE pulse length must be at least 1 clock cycle.

### 35.16.1.3 SMC Cycle Register

**Name:** SMC_CYCLE[0..3]
**Offset:** 0x00
**Reset:** 0
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the "SMC Write Protection Mode Register" .

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | NRD_CYCLE[8] |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NRD_CYCLE[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | NWE_CYCLE[8] |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NWE_CYCLE[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 24:16 – NRD_CYCLE[8:0]** Total Read Cycle Length
The total read cycle length is the total duration in clock cycles of the read cycle. It is equal to the sum of the setup, pulse and hold steps of the NRD and NCS signals. It is defined as:
Read cycle length = (NRD_CYCLE[8:7]*256 + NRD_CYCLE[6:0]) clock cycles

**Bits 8:0 – NWE_CYCLE[8:0]** Total Write Cycle Length
The total write cycle length is the total duration in clock cycles of the write cycle. It is equal to the sum of the setup, pulse and hold steps of the NWE and NCS signals. It is defined as:
Write cycle length = (NWE_CYCLE[8:7]*256 + NWE_CYCLE[6:0]) clock cycles

© 2021 Microchip Technology Inc.
and its subsidiaries

### 35.16.1.4 SMC Mode Register

**Name:** SMC_MODE[0..3]
**Offset:** 0x00
**Reset:** 0
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the "SMC Write Protection Mode Register" .

The user must confirm the SMC configuration by writing any one of the SMC_MODE registers.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | PS[1:0] | | | | | PMEN |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | TDF_MODE | TDF_CYCLES[3:0] | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | DBW | | | | BAT |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EXNW_MODE[1:0] | | | | WRITE_MODE | READ_MODE |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 29:28 – PS[1:0]** Page Size
If page mode is enabled, this field indicates the size of the page in bytes.

| Value | Name | Description |
|---|---|---|
| 0 | 4_BYTE | 4-byte page |
| 1 | 8_BYTE | 8-byte page |
| 2 | 16_BYTE | 16-byte page |
| 3 | 32_BYTE | 32-byte page |

**Bit 24 – PMEN** Page Mode Enabled

| Value | Description |
|---|---|
| 0 | Standard read is applied. |
| 1 | Asynchronous burst read in page mode is applied on the corresponding chip select. |

**Bit 20 – TDF_MODE** TDF Optimization

| Value | Description |
|---|---|
| 0 | TDF optimization disabled–the number of TDF wait states is inserted before the next access begins. |
| 1 | TDF optimization enabled–the number of TDF wait states is optimized using the setup period of the next read/write access. |

**Bits 19:16 – TDF_CYCLES[3:0]** Data Float Time
This field gives the integer number of clock cycles required by the external device to release the data after the rising edge of the read controlling signal. The SMC always provide one full cycle of bus turnaround after the TDF_CYCLES period. The external bus cannot be used by another chip select during TDF_CYCLES + 1 cycles. From 0 up to 15 TDF_CYCLES can be set.

**Bit 12 – DBW** Data Bus Width

| Value | Name | Description |
|---|---|---|
| 0 | 8_BIT | 8-bit Data Bus |
| 1 | 16_BIT | 16-bit Data Bus |

**Bit 8 – BAT** Byte Access Type
This field is used only if DBW defines a 16-bit data bus.

| Value | Name | Description |
|---|---|---|
| 0 | BYTE_SELECT | Byte select access type:<br>- Write operation is controlled using NCS, NWE, NBS0, NBS1.<br>- Read operation is controlled using NCS, NRD, NBS0, NBS1. |
| 1 | BYTE_WRITE | Byte write access type:<br>- Write operation is controlled using NCS, NWR0, NWR1.<br>- Read operation is controlled using NCS and NRD. |

**Bits 5:4 – EXNW_MODE[1:0]** NWAIT Mode
The NWAIT signal is used to extend the current read or write signal. It is only taken into account during the pulse phase of the read and write controlling signal. When the use of NWAIT is enabled, at least one cycle hold duration must be programmed for the read and write controlling signal.

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | Disabled–The NWAIT input signal is ignored on the corresponding chip select. |
| 1 | Reserved | |
| 2 | FROZEN | Frozen Mode–If asserted, the NWAIT signal freezes the current read or write cycle. After deassertion, the read/write cycle is resumed from the point where it was stopped. |
| 3 | READY | Ready Mode–The NWAIT signal indicates the availability of the external device at the end of the pulse of the controlling read or write signal, to complete the access. If high, the access normally completes. If low, the access is extended until NWAIT returns high. |

**Bit 1 – WRITE_MODE** Write Mode

| Value | Description |
|---|---|
| 0 | The write operation is controlled by the NCS signal.<br>– If TDF optimization is enabled (TDF_MODE =1), TDF wait states will be inserted after the setup of NCS. |
| 1 | The write operation is controlled by the NWE signal.<br>– If TDF optimization is enabled (TDF_MODE =1), TDF wait states will be inserted after the setup of NWE. |

**Bit 0 – READ_MODE** Read Mode

| Value | Description |
|---|---|
| 0 | The read operation is controlled by the NCS signal.<br>– If TDF cycles are programmed, the external bus is marked busy after the rising edge of NCS.<br>– If TDF optimization is enabled (TDF_MODE =1), TDF wait states are inserted after the setup of NCS. |
| 1 | The read operation is controlled by the NRD signal.<br>– If TDF cycles are programmed, the external bus is marked busy after the rising edge of NRD.<br>– If TDF optimization is enabled (TDF_MODE =1), TDF wait states are inserted after the setup of NRD. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 35.16.1.5 SMC Off-Chip Memory Scrambling Register

**Name:** SMC_OCMS
**Offset:** 0x80
**Reset:** 0x00000000
**Property:** Read/Write

**Note:** This register can only be written if the WPEN bit is cleared in the SMC Write Protection Mode Register (35.16.1.8 SMC_WPMR).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CS3SE | CS2SE | CS1SE | CS0SE |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SMSE |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 8, 9, 10, 11 – CSSE** Chip Select x Scrambling Enable

| Value | Description |
|---|---|
| 0 | Disable scrambling for CSx. |
| 1 | Enable scrambling for CSx. |

**Bit 0 – SMSE** Static Memory Controller Scrambling Enable

| Value | Description |
|---|---|
| 0 | Disable scrambling for SMC access. |
| 1 | Enable scrambling for SMC access. |

### 35.16.1.6 SMC Off-Chip Memory Scrambling Key1 Register

**Name:** SMC_KEY1
**Offset:** 0x84
**Reset:** 0x00000000
**Property:** Write-once

**Note:**

1. 'Write-once' access indicates that the first write access after a system reset prevents any further modification of the value of this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY1[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – KEY1[31:0]** Off-Chip Memory Scrambling (OCMS) Key Part 1
When off-chip memory scrambling is enabled, KEY1 and KEY2 values determine data scrambling.

### 35.16.1.7 SMC Off-Chip Memory Scrambling Key2 Register

**Name:** SMC_KEY2
**Offset:** 0x88
**Reset:** 0x00000000
**Property:** Write-once

**Note:** 'Write-once' access indicates that the first write access after a system reset prevents any further modification of the value of this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEY2[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – KEY2[31:0]** Off-Chip Memory Scrambling (OCMS) Key Part 2
When off-chip memory scrambling is enabled, KEY1 and KEY2 values determine data scrambling.

### 35.16.1.8 SMC Write Protection Mode Register

**Name:** SMC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x534D43 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protect Enable

See "Register Write Protection" for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x534D43 (“SMC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x534D43 (“SMC” in ASCII). |

### 35.16.1.9 SMC Write Protection Status Register

**Name:** SMC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the SMC_WPSR register. |
| 1 | A write protection violation has occurred since the last read of the SMC_WPSR register. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 36. DMA Controller (XDMAC)

## 36.1 Description

The DMA Controller (XDMAC) is a AHB-protocol central direct memory access controller. It performs peripheral data transfer and memory move operations over one or two bus ports through the unidirectional communication channel. Each channel is fully programmable and provides both peripheral or memory-to-memory transfers. The channel features are configurable at implementation.

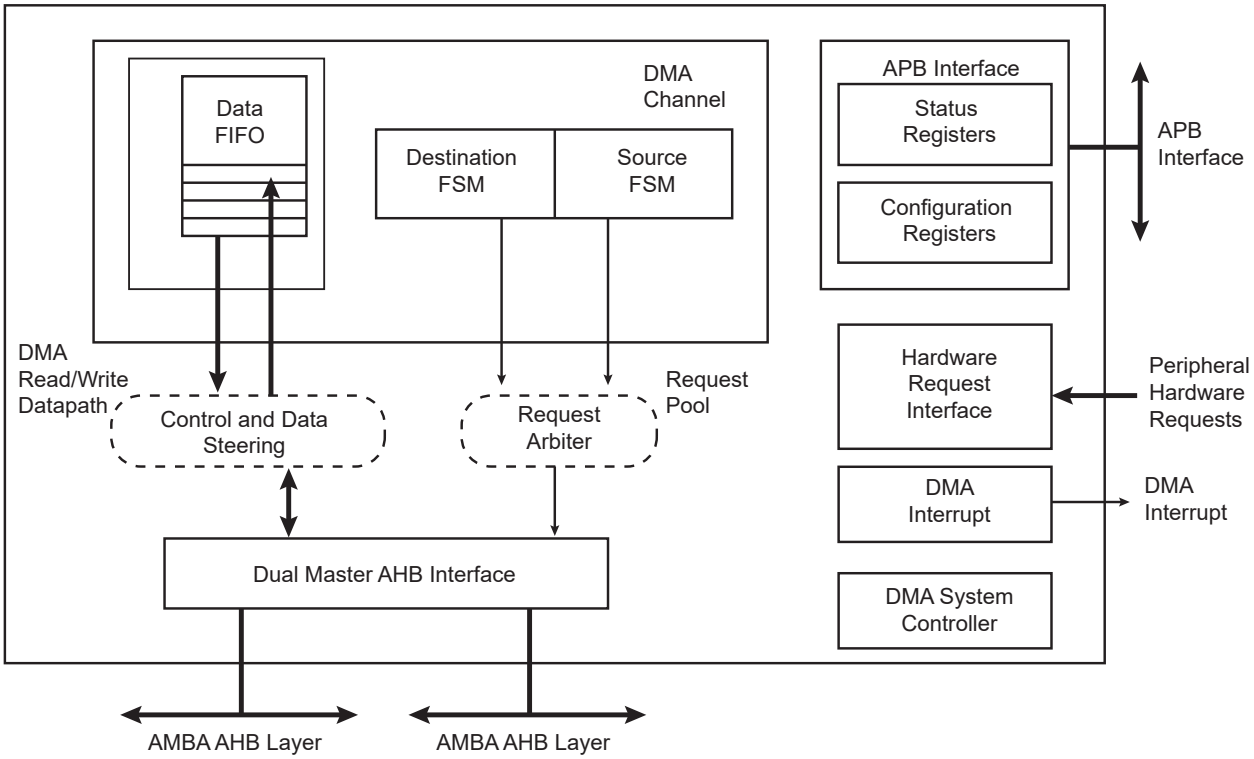
## 36.2 Embedded Characteristics

- 2 AHB Master Interfaces
- 24 DMA Channels
- 43 Hardware Requests
- 3.1 Kbytes Embedded FIFO
- Supports Peripheral-to-Memory, Memory-to-Peripheral, or Memory-to-Memory Transfer Operations
- Peripheral DMA Operation Runs on Bytes (8-bit), Half-Word (16-bit) and Word (32-bit)
- Memory DMA Operation Runs on Bytes (8 bit), Half-Word (16-bit) and Word (32 -bit)
- Supports Hardware and Software Initiated Transfers
- Supports Linked List Operations
- Supports Incrementing or Fixed Addressing Mode
- Supports Programmable Independent Data Striding for Source and Destination
- Supports Programmable Independent Microblock Striding for Source and Destination
- Configurable Priority Group and Arbitration Policy
- Programmable AHB Burst Length
- Configuration Interface Accessible through APB Interface
- XDMAC Architecture Includes Multiport FIFO
- Supports Multiple View Channel Descriptor
- Automatic Flush of Channel Trailing Bytes
- Automatic Coarse-Grain and Fine-Grain Clock Gating
- Hardware Acceleration of Memset Pattern

© 2021 Microchip Technology Inc.
and its subsidiaries

## 36.3 Block Diagram

**Figure 36-1. DMA Controller (XDMAC) Block Diagram**




## 36.4 DMA Controller Peripheral Connections

**Table 36-1. Peripheral Hardware Requests**

| Peripheral Name | Transfer Type | HW Interface Number (XDMAC_CC.PERID) |
|---|---|---|
| HSMCI | Transmit/Receive | 0 |
| SPI0 | Transmit | 1 |
| SPI0 | Receive | 2 |
| SPI1 | Transmit | 3 |
| SPI1 | Receive | 4 |
| QSPI | Transmit | 5 |
| QSPI | Receive | 6 |
| USART0 | Transmit | 7 |
| USART0 | Receive | 8 |
| USART1 | Transmit | 9 |
| USART1 | Receive | 10 |
| USART2 | Transmit | 11 |
| USART2 | Receive | 12 |

.....continued

| Peripheral Name | Transfer Type | HW Interface Number (XDMAC_CC.PERID) |
|---|---|---|
| PWM0 | Transmit | 13 |
| TWIHS0 | Transmit | 14 |
| TWIHS0 | Receive | 15 |
| TWIHS1 | Transmit | 16 |
| TWIHS1 | Receive | 17 |
| TWIHS2 | Transmit | 18 |
| TWIHS2 | Receive | 19 |
| UART0 | Transmit | 20 |
| UART0 | Receive | 21 |
| UART1 | Transmit | 22 |
| UART1 | Receive | 23 |
| UART2 | Transmit | 24 |
| UART2 | Receive | 25 |
| UART3 | Transmit | 26 |
| UART3 | Receive | 27 |
| UART4 | Transmit | 28 |
| UART4 | Receive | 29 |
| DACC | Transmit | 30 |
| SSC | Transmit | 32 |
| SSC | Receive | 33 |
| PIOA | Receive | 34 |
| AFEC0 | Receive | 35 |
| AFEC1 | Receive | 36 |
| AES | Transmit | 37 |
| AES | Receive | 38 |
| PWM1 | Transmit | 39 |
| TC0 | Receive | 40 |
| TC3 | Receive | 41 |
| TC6 | Receive | 42 |
| TC9 | Receive | 43 |
| I2SC0 | Transmit Left | 44 |
| I2SC0 | Receive Left | 45 |
| I2SC1 | Transmit Left | 46 |
| I2SC1 | Receive Left | 47 |
| I2SC0 | Transmit Right | 48 |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | | |
|---|---|---|
| **Peripheral Name** | **Transfer Type** | **HW Interface Number (XDMAC_CC.PERID)** |
| I2SC0 | Receive Right | 49 |
| I2SC1 | Transmit Right | 50 |
| I2SC1 | Receive Right | 51 |

## 36.5 Functional Description

### 36.5.1 Basic Definitions

**Source Peripheral**: Slave device, memory mapped on the interconnection network, from where the XDMAC reads data. The source peripheral teams up with a destination peripheral to form a channel. A data read operation is scheduled when the peripheral transfer request is asserted.

**Destination Peripheral**: Slave device, memory mapped on the interconnection network, to which the XDMAC writes. A write data operation is scheduled when the peripheral transfer request is asserted.

**Channel**: The data movement between source and destination creates a logical channel.

**Transfer Type**: The transfer is hardware-synchronized when it is paced by the peripheral hardware request, otherwise the transfer is self-triggered (memory to memory transfer).

### 36.5.2 Transfer Hierarchy Diagram

**XDMAC Master Transfer**: The Master Transfer is composed of a linked list of blocks. The channel address, control and configuration registers can be modified at the inter block boundary. The descriptor structure modifies the channel registers conditionally. Interrupts can be generated on a per block basis or when the end of linked list event occurs.

**XDMAC Block**: An XDMAC block is composed of a programmable number of microblocks. The channel configuration registers remain unchanged at the inter microblock boundary. The source and destination addresses are conditionally updated with a programmable signed number.

**XDMAC Microblock**: The microblock is composed of a programmable number of data. The channel configuration registers remain unchanged at the data boundary. The data address may be fixed (a FIFO location, a peripheral transmit or receive register), incrementing (a memory-mapped area) by a programmable signed number.

**XDMAC Burst and Incomplete Burst**: In order to improve the overall performance when accessing dynamic external memory, burst access is mandatory. Each data of the microblock is considered as a part of a memory burst. The programmable burst value indicates the largest memory burst allowed on a per channel basis. When the microblock length is not an integral multiple of the burst size, an incomplete burst is performed to read or write the last trailing bytes.

**XDMAC Chunk and Incomplete Chunk**: When a peripheral synchronized transfer is activated, the microblock splits into a number of data chunks. The chunk size is programmable. The larger the chunk is, the better the performance is. When the transfer size is not a multiple of the chunk size, the last chunk may be incomplete.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 36-2. XDMAC Memory Transfer Hierarchy**

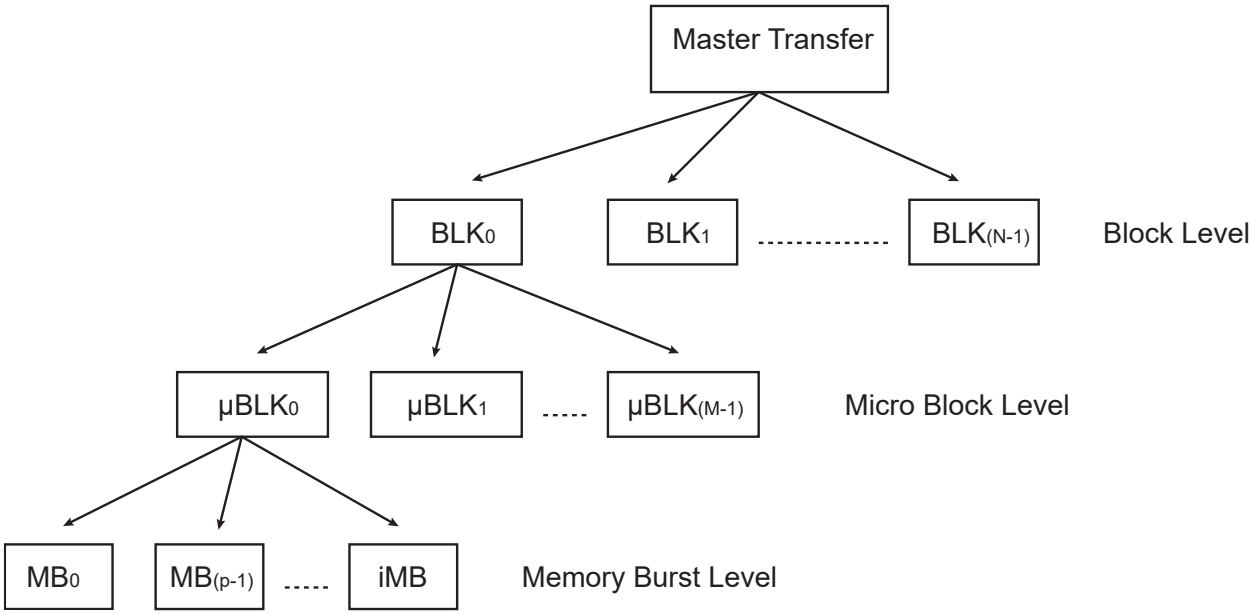
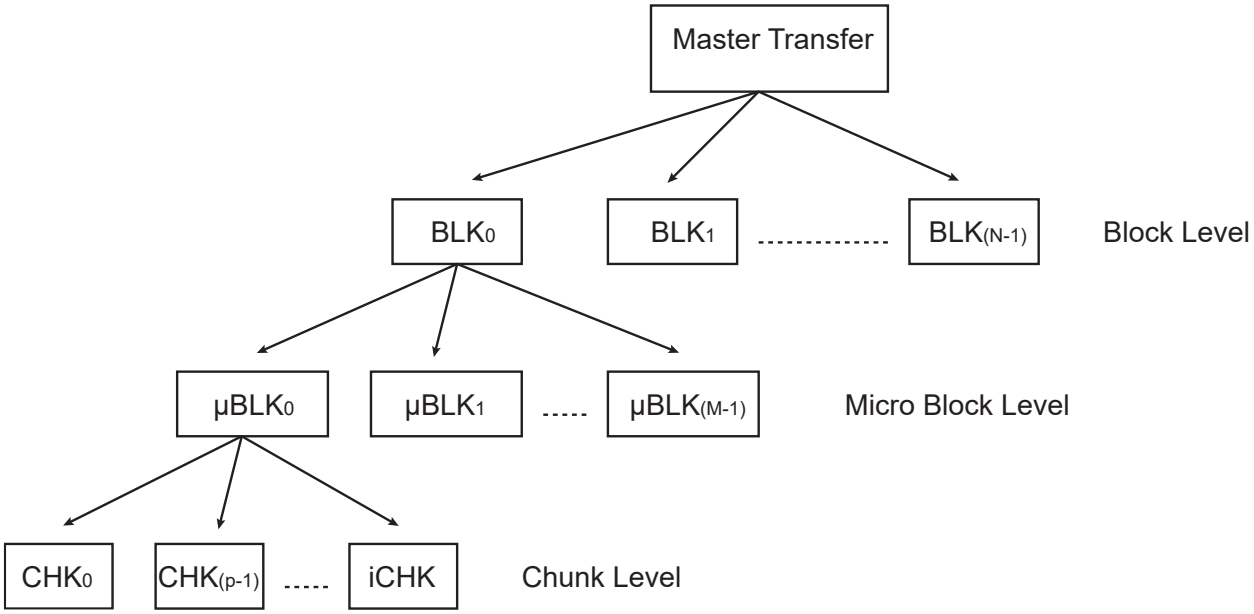



**Figure 36-3. XDAMC Peripheral Transfer Hierarchy**




### 36.5.3 Peripheral Synchronized Transfer

A peripheral hardware request interface is used to control the pace of the chunk transfer. When a peripheral is ready to transmit or receive a chunk of data, it asserts its request line and the DMA Controller transfers a data to or from the memory to the peripheral.

#### 36.5.3.1 Software Triggered Synchronized Transfer

The Peripheral hardware request can be software controlled using the SWREQ field of the XDMAC Global Channel Software Request Register (XDMAC_GSWR). The peripheral synchronized transfer is paced using a processor write access in the XDMAC_GSWR. Each bit of that register triggers a transfer request. The XDMAC Global Channel Software Request Status Register (XDMAC_GSWS) indicates the status of the request; when set, the request is still pending.

### 36.5.4 XDMAC Transfer Software Operation

#### 36.5.4.1 Single Block Transfer With Single Microblock

1. Read the XDMAC Global Channel Status Register (XDMAC_GS) to select a free channel.
2. Clear the pending Interrupt Status bit(s) by reading the selected XDMAC Channel x Interrupt Status Register (XDMAC_CISx).
3. Write the XDMAC Channel x Source Address Register (XDMAC_CSAx) for channel x.
4. Write the XDMAC Channel x Destination Address Register (XDMAC_CDAx) for channel x.
5. Program field UBLEN in the XDMAC Channel x Microblock Control Register (XDMAC_CUBCx) with the number of data.
6. Program the XDMAC Channel x Configuration Register (XDMAC_CCx):
   6.1. Clear XDMAC_CCx.TYPE for a memory-to-memory transfer, otherwise set this bit.
   6.2. Configure XDMAC_CCx.MBSIZE to the memory burst size used.
   6.3. Configure XDMAC_CCx.SAM and DAM to Memory Addressing mode.
   6.4. Configure XDMAC_CCx.DSYNC to select the peripheral transfer direction.
   6.5. Configure XDMAC_CCx.CSIZE to configure the channel chunk size (only relevant for peripheral synchronized transfer).
   6.6. Configure XDMAC_CCx.DWIDTH to configure the transfer data width.
   6.7. Configure XDMAC_CCx.SIF, XDMAC_CCx.DIF to configure the master interface used to read data and write data, respectively.
   6.8. Configure XDMAC_CCx.PERID to select the active hardware request line (only relevant for a peripheral synchronized transfer).
   6.9. Set XDMAC_CCx.SWREQ to use a software request (only relevant for a peripheral synchronized transfer).
7. Clear the following five registers:
   – XDMAC Channel x Next Descriptor Control Register (XDMAC_CNDCx)
   – XDMAC Channel x Block Control Register (XDMAC_CBCx)
   – XDMAC Channel x Data Stride Memory Set Pattern Register (XDMAC_CDS_MSPx)
   – XDMAC Channel x Source Microblock Stride Register (XDMAC_CSUSx)
   – XDMAC Channel x Destination Microblock Stride Register (XDMAC_CDUSx)
   This indicates that the linked list is disabled, there is only one block and striding is disabled.
8. Enable the Microblock interrupt by writing a ‘1’ to bit BIE in the XDMAC Channel x Interrupt Enable Register (XDMAC_CIEx). Enable the Channel x Interrupt Enable bit by writing a ‘1’ to bit IEx in the XDMAC Global Interrupt Enable Register (XDMAC_GIE).
9. Enable channel x by writing a ‘1’ to bit ENx in the XDMAC Global Channel Enable Register (XDMAC_GE). XDMAC_GS.STx (XDMAC Channel x Status bit) is set by hardware.
10. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

#### 36.5.4.2 Single Block Transfer With Multiple Microblock

1. Read the XDMAC_GS register to choose a free channel.
2. Clear the pending Interrupt Status bit by reading the chosen XDMAC_CISx register.
3. Write the XDMAC_CSAx register for channel x.
4. Write the XDMAC_CDAx register for channel x.
5. Program XDMAC_CUBCx.UBLEN with the number of data.
6. Program XDMAC_CCx register (see “Single Block Transfer With Single Microblock”).
7. Program XDMAC_CBCx.BLEN with the number of microblocks of data.
8. Clear the following registers:
   – XDMAC_CNDCx
   – XDMAC_CDS_MSPx
   – XDMAC_CSUSx XDMAC_CDUSx

© 2021 Microchip Technology Inc. and its subsidiaries

This indicates that the linked list is disabled and striding is disabled.

9. Enable the Block interrupt by writing a ‘1’ to XDMAC_CIEx.BIE, enable the Channel x Interrupt Enable bit by writing a ‘1’ to XDMAC_GIEx.IEx.
10. Enable channel x by writing a ‘1’ to the XDMAC_GE.ENx. XDMAC_GS.STx is set by hardware.
11. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

#### 36.5.4.3 Master Transfer

1. Read the XDMAC_GS register to choose a free channel.
2. Clear the pending Interrupt Status bit by reading the chosen XDMAC_CISx register.
3. Build a linked list of transfer descriptors in memory. The descriptor view is programmable on a per descriptor basis. The linked list items structure must be word aligned. MBR_UBC.NDE must be configured to 0 in the last descriptor to terminate the list.
4. Configure field NDA in the XDMAC Channel x Next Descriptor Address Register (XDMAC_CNDAx) with the first descriptor address and bit XDMAC_CNDAx.NDAIF with the master interface identifier.
5. Configure the XDMAC_CNDCx register:
   5.1. Set XDMAC_CNDCx.NDE to enable the descriptor fetch.
   5.2. Set XDMAC_CNDCx.NDSUP to update the source address at the descriptor fetch time, otherwise clear this bit.
   5.3. Set XDMAC_CNDCx.NDDUP to update the destination address at the descriptor fetch time, otherwise clear this bit.
   5.4. Configure XDMAC_CNDCx.NDVIEW to define the length of the first descriptor.
6. Enable the End of Linked List interrupt by writing a ‘1’ to XDMAC_CIEx.LIE.
7. Enable channel x by writing a ‘1’ to XDMAC_GE.ENx. XDMAC_GS.STx is set by hardware.
8. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

#### 36.5.4.4 Disabling A Channel Before Transfer Completion

Under normal operation, the software enables a channel by writing a ‘1’ to XDMAC_GE.ENx, then the hardware disables a channel on transfer completion by clearing bit XDMAC_GS.STx. To disable a channel, write a ‘1’ to bit XDMAC_GD.DIx and poll the XDMAC_GS register.

## 36.6 Linked List Descriptor Operation

### 36.6.1 Linked List Descriptor View

#### 36.6.1.1 Channel Next Descriptor View 0–3 Structures

**Table 36-2. Channel Next Descriptor View 0–3 Structures**

| Channel Next Descriptor | Offset | Structure member | Name |
|---|---|---|---|
| View 0 Structure | DSCR_ADDR+0x00 | Next Descriptor Address Member | MBR_NDA |
| | DSCR_ADDR+0x04 | Microblock Control Member | MBR_UBC |
| | DSCR_ADDR+0x08 | Transfer Address Member | MBR_TA |
| View 1 Structure | DSCR_ADDR+0x00 | Next Descriptor Address Member | MBR_NDA |
| | DSCR_ADDR+0x04 | Microblock Control Member | MBR_UBC |
| | DSCR_ADDR+0x08 | Source Address Member | MBR_SA |
| | DSCR_ADDR+0x0C | Destination Address Member | MBR_DA |

| .....continued | | | |
|---|---|---|---|
| **Channel Next Descriptor** | **Offset** | **Structure member** | **Name** |
| View 2 Structure | DSCR_ADDR+0x00 | Next Descriptor Address Member | MBR_NDA |
| | DSCR_ADDR+0x04 | Microblock Control Member | MBR_UBC |
| | DSCR_ADDR+0x08 | Source Address Member | MBR_SA |
| | DSCR_ADDR+0x0C | Destination Address Member | MBR_DA |
| | DSCR_ADDR+0x10 | Configuration Register | MBR_CFG |
| View 3 Structure | DSCR_ADDR+0x00 | Next Descriptor Address Member | MBR_NDA |
| | DSCR_ADDR+0x04 | Microblock Control Member | MBR_UBC |
| | DSCR_ADDR+0x08 | Source Address Member | MBR_SA |
| | DSCR_ADDR+0x0C | Destination Address Member | MBR_DA |
| | DSCR_ADDR+0x10 | Configuration Member | MBR_CFG |
| | DSCR_ADDR+0x14 | Block Control Member | MBR_BC |
| | DSCR_ADDR+0x18 | Data Stride Member | MBR_DS |
| | DSCR_ADDR+0x1C | Source Microblock Stride Member | MBR_SUS |
| | DSCR_ADDR+0x20 | Destination Microblock Stride Member | MBR_DUS |

### 36.6.2 Descriptor Structure Members Description

### 36.6.2.1 Descriptor Structure Microblock Control Member

**Name:** MBR_UBC
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | NVIEW[1:0] | | NDEN | NSEN | NDE |
| Access | | | | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | | | | | | | | |

**Bits 28:27 – NVIEW[1:0]** Next Descriptor View

| Value | Name | Description |
|---|---|---|
| 0 | NDV0 | Next Descriptor View 0 |
| 1 | NDV1 | Next Descriptor View 1 |
| 2 | NDV2 | Next Descriptor View 2 |
| 3 | NDV3 | Next Descriptor View 3 |

**Bit 26 – NDEN** Next Descriptor Destination Update

| Value | Description |
|---|---|
| 0 | Destination parameters remain unchanged. |
| 1 | Destination parameters are updated when the descriptor is retrieved. |

**Bit 25 – NSEN** Next Descriptor Source Update

| Value | Description |
|---|---|
| 0 | Source parameters remain unchanged. |
| 1 | Source parameters are updated when the descriptor is retrieved. |

**Bit 24 – NDE** Next Descriptor Enable

| Value | Description |
|---|---|
| 0 | Descriptor fetch is disabled. |
| 1 | Descriptor fetch is enabled. |

**Bits 23:0 – UBLEN[23:0]** Microblock Length
This field indicates the number of data in the microblock. The microblock contains UBLEN data.

## 36.7 XDMAC Maintenance Software Operations

### 36.7.1 Disabling a Channel

A disable channel request occurs when a write operation is performed in the XDMAC_GD register. If the channel is source peripheral synchronized (bit XDMAC_CCx.TYPE is set and bit XDMAC_CCx.DSYNC is cleared), then pending bytes (bytes located in the FIFO) are written to memory and bit XDMAC_CISx.DIS is set. If the channel is not source peripheral synchronized, the current channel transaction (read or write) is terminated and XDMAC_CISx.DIS is set. XDMAC_GS.STx is cleared by hardware when the current transfer is completed. The channel is no longer active and can be reused.

### 36.7.2 Suspending a Channel

A disable channel request occurs when a write operation is performed in the XDMAC_GD register. If the channel is source peripheral synchronized (bit XDMAC_CCx.TYPE is set and bit XDMAC_CCx.DSYNC is cleared), then pending bytes (bytes located in the FIFO) are written to memory and bit XDMAC_CISx.DIS is set. If the channel is not source peripheral synchronized, the current channel transaction (read or write) is terminated and XDMAC_CISx.DIS is set. XDMAC_GS.STx is cleared by hardware when the current transfer is completed. The channel is no longer active and can be reused.

### 36.7.3 Flushing a Channel

A FIFO flush command is issued by writing to the XDMAC_SWF register. The content of the FIFO is written to memory. XDMAC_CISx.FIS (End of Flush Interrupt Status bit) is set when the last byte is successfully transferred to memory. The channel is not disabled. The flush operation is not blocking, meaning that read operation can be scheduled during the flush write operation. The flush operation is only relevant for peripheral to memory transfer where pending peripheral bytes are buffered into the channel FIFO.

### 36.7.4 Maintenance Operation Priority

#### 36.7.4.1 Disable Operation Priority

- When a disable request occurs on a suspended channel, the XDMAC_GWS.WSx (Channel x Write Suspend bit) is cleared. If the transfer is source peripheral synchronized, the pending bytes are drained to memory. The bit XDMAC_CISx.DIS is set.
- When a disable request follows a flush request, if the flush last transaction is not yet scheduled, the flush request is discarded and the disable procedure is applied. Bit XDMAC_CISx.FIS is not set. Bit XDMAC_CISx.DIS is set when the disable request is completed. If the flush request transaction is already scheduled, the XDMAC_CISx.FIS is set. XDMAC_CISx.DIS is also set when the disable request is completed.

#### 36.7.4.2 Flush Operation Priority

- When a flush request occurs on a suspended channel, if there are pending bytes in the FIFO, they are written out to memory, XDMAC_CISx.FIS is set. If the FIFO is empty, XDMAC_CISx.FIS is also set.
- If the flush operation is performed after a disable request, the flush command is ignored. XDMAC_CISx.FIS is not set.

#### 36.7.4.3 Suspend Operation Priority

If the suspend operation is performed after a disable request, the write suspend operation is ignored.

## 36.8 XDMAC Software Requirements

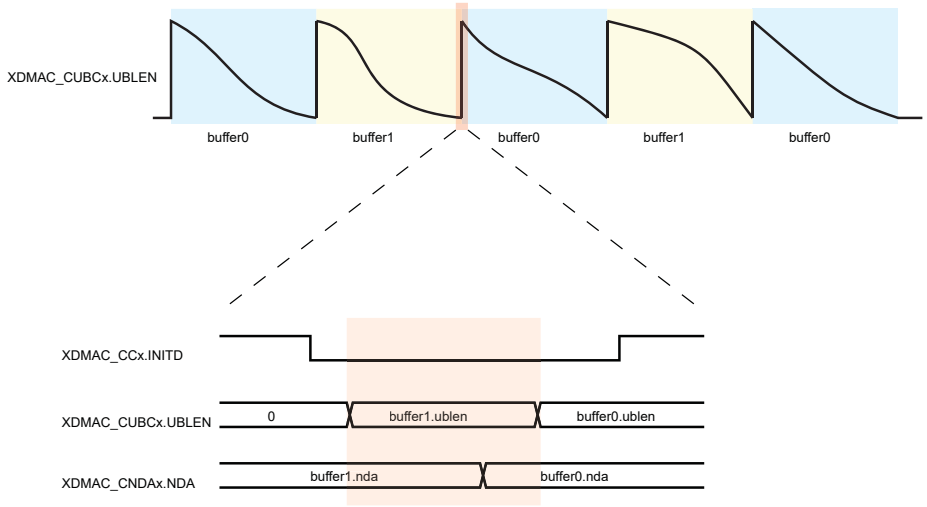
- Write operations to channel registers are not be performed in an active channel after the channel is enabled. If any channel parameters must be reprogrammed, this can only be done after disabling the XDMAC channel.
- XDMAC_CSAx and XDMAC_CDAx channel registers are to be programmed with a byte, half-word or word aligned address depending on the Channel x Data Width field (DWIDTH) of the XDMAC Channel x Configuration Register.

- When XDMAC_CC.INITD is set to 0, XDMAC_CUBC.UBLEN and XDMAC_CNDA.NDA field values are unreliable when the descriptor is being updated. The following procedure applies to get the buffer descriptor identifier and the residual bytes:

```
Read XDMAC_CNDAx.NDA(nda0)
Read XDMAC_CCx.INITD(initd0)
Read XDMAC_CCx.INITD(initd0)
Read XDMAC_CUBCx.UBLEN(ublen)
Read XDMAC_CCx.INITD(initd1)
Read XDMA_CNDAx.NDA(nda1)
If (nda0 == nda1 && initd0 == 1 && initd1 == 1).
Then the ublen is correct, the buffer id is nda.
Else retry
```

See the figure below.

**Figure 36-4. INITD Timing Diagram**



## 36.9 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | XDMAC_GTYPE | 7:0 | FIFO_SZ[2:0] | | | | NB_CH[4:0] | | | |
| | | 15:8 | FIFO_SZ[10:3] | | | | | | | |
| | | 23:16 | | NB_REQ[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | XDMAC_GCFG | 7:0 | | | | | CGDISIF | CGDISFIFO | CGDISPIPE | CGDISREG |
| | | 15:8 | | | | | | | | BXKBEN |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | XDMAC_GWAC | 7:0 | PW1[3:0] | | | | PW0[3:0] | | | |
| | | 15:8 | PW3[3:0] | | | | PW2[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | XDMAC_GIE | 7:0 | IE7 | IE6 | IE5 | IE4 | IE3 | IE2 | IE1 | IE0 |
| | | 15:8 | IE15 | IE14 | IE13 | IE12 | IE11 | IE10 | IE9 | IE8 |
| | | 23:16 | IE23 | IE22 | IE21 | IE20 | IE19 | IE18 | IE17 | IE16 |
| | | 31:24 | | | | | | | | |
| 0x10 | XDMAC_GID | 7:0 | ID7 | ID6 | ID5 | ID4 | ID3 | ID2 | ID1 | ID0 |
| | | 15:8 | ID15 | ID14 | ID13 | ID12 | ID11 | ID10 | ID9 | ID8 |
| | | 23:16 | ID23 | ID22 | ID21 | ID20 | ID19 | ID18 | ID17 | ID16 |
| | | 31:24 | | | | | | | | |
| 0x14 | XDMAC_GIM | 7:0 | IM7 | IM6 | IM5 | IM4 | IM3 | IM2 | IM1 | IM0 |
| | | 15:8 | IM15 | IM14 | IM13 | IM12 | IM11 | IM10 | IM9 | IM8 |
| | | 23:16 | IM23 | IM22 | IM21 | IM20 | IM19 | IM18 | IM17 | IM16 |
| | | 31:24 | | | | | | | | |
| 0x18 | XDMAC_GIS | 7:0 | IS7 | IS6 | IS5 | IS4 | IS3 | IS2 | IS1 | IS0 |
| | | 15:8 | IS15 | IS14 | IS13 | IS12 | IS11 | IS10 | IS9 | IS8 |
| | | 23:16 | IS23 | IS22 | IS21 | IS20 | IS19 | IS18 | IS17 | IS16 |
| | | 31:24 | | | | | | | | |
| 0x1C | XDMAC_GE | 7:0 | EN7 | EN6 | EN5 | EN4 | EN3 | EN2 | EN1 | EN0 |
| | | 15:8 | EN15 | EN14 | EN13 | EN12 | EN11 | EN10 | EN9 | EN8 |
| | | 23:16 | EN23 | EN22 | EN21 | EN20 | EN19 | EN18 | EN17 | EN16 |
| | | 31:24 | | | | | | | | |
| 0x20 | XDMAC_GD | 7:0 | DI7 | DI6 | DI5 | DI4 | DI3 | DI2 | DI1 | DI0 |
| | | 15:8 | DI15 | DI14 | DI13 | DI12 | DI11 | DI10 | DI9 | DI8 |
| | | 23:16 | DI23 | DI22 | DI21 | DI20 | DI19 | DI18 | DI17 | DI16 |
| | | 31:24 | | | | | | | | |
| 0x24 | XDMAC_GS | 7:0 | ST7 | ST6 | ST5 | ST4 | ST3 | ST2 | ST1 | ST0 |
| | | 15:8 | ST15 | ST14 | ST13 | ST12 | ST11 | ST10 | ST9 | ST8 |
| | | 23:16 | ST23 | ST22 | ST21 | ST20 | ST19 | ST18 | ST17 | ST16 |
| | | 31:24 | | | | | | | | |
| 0x28 | XDMAC_GRS | 7:0 | RS7 | RS6 | RS5 | RS4 | RS3 | RS2 | RS1 | RS0 |
| | | 15:8 | RS15 | RS14 | RS13 | RS12 | RS11 | RS10 | RS9 | RS8 |
| | | 23:16 | RS23 | RS22 | RS21 | RS20 | RS19 | RS18 | RS17 | RS16 |
| | | 31:24 | | | | | | | | |
| 0x2C | XDMAC_GWS | 7:0 | WS7 | WS6 | WS5 | WS4 | WS3 | WS2 | WS1 | WS0 |
| | | 15:8 | WS15 | WS14 | WS13 | WS12 | WS11 | WS10 | WS9 | WS8 |
| | | 23:16 | WS23 | WS22 | WS21 | WS20 | WS19 | WS18 | WS17 | WS16 |
| | | 31:24 | | | | | | | | |
| 0x30 | XDMAC_GRWS | 7:0 | RWS7 | RWS6 | RWS5 | RWS4 | RWS3 | RWS2 | RWS1 | RWS0 |
| | | 15:8 | RWS15 | RWS14 | RWS13 | RWS12 | RWS11 | RWS10 | RWS9 | RWS8 |
| | | 23:16 | RWS23 | RWS22 | RWS21 | RWS20 | RWS19 | RWS18 | RWS17 | RWS16 |
| | | 31:24 | | | | | | | | |
| 0x34 | XDMAC_GRWR | 7:0 | RWR7 | RWR6 | RWR5 | RWR4 | RWR3 | RWR2 | RWR1 | RWR0 |
| | | 15:8 | RWR15 | RWR14 | RWR13 | RWR12 | RWR11 | RWR10 | RWR9 | RWR8 |
| | | 23:16 | RWR23 | RWR22 | RWR21 | RWR20 | RWR19 | RWR18 | RWR17 | RWR16 |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x38 | XDMAC_GSWR | 7:0 | SWREQ7 | SWREQ6 | SWREQ5 | SWREQ4 | SWREQ3 | SWREQ2 | SWREQ1 | SWREQ0 |
| | | 15:8 | SWREQ15 | SWREQ14 | SWREQ13 | SWREQ12 | SWREQ11 | SWREQ10 | SWREQ9 | SWREQ8 |
| | | 23:16 | SWREQ23 | SWREQ22 | SWREQ21 | SWREQ20 | SWREQ19 | SWREQ18 | SWREQ17 | SWREQ16 |
| | | 31:24 | | | | | | | | |
| 0x3C | XDMAC_GSWS | 7:0 | SWRS7 | SWRS6 | SWRS5 | SWRS4 | SWRS3 | SWRS2 | SWRS1 | SWRS0 |
| | | 15:8 | SWRS15 | SWRS14 | SWRS13 | SWRS12 | SWRS11 | SWRS10 | SWRS9 | SWRS8 |
| | | 23:16 | SWRS23 | SWRS22 | SWRS21 | SWRS20 | SWRS19 | SWRS18 | SWRS17 | SWRS16 |
| | | 31:24 | | | | | | | | |
| 0x40 | XDMAC_GSWF | 7:0 | SWF7 | SWF6 | SWF5 | SWF4 | SWF3 | SWF2 | SWF1 | SWF0 |
| | | 15:8 | SWF15 | SWF14 | SWF13 | SWF12 | SWF11 | SWF10 | SWF9 | SWF8 |
| | | 23:16 | SWF23 | SWF22 | SWF21 | SWF20 | SWF19 | SWF18 | SWF17 | SWF16 |
| | | 31:24 | | | | | | | | |
| 0x44 ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | XDMAC_CIE0 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x54 | XDMAC_CID0 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x58 | XDMAC_CIM0 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x5C | XDMAC_CIS0 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x60 | XDMAC_CSA0 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x64 | XDMAC_CDA0 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x68 | XDMAC_CNDA0 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x6C | XDMAC_CNDC0 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x70 | XDMAC_CUBC0 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 | XDMAC_CBC0 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | XDMAC_CC0 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x7C | XDMAC_CDS_MSP0 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x80 | XDMAC_CSUS0 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 | XDMAC_CDUS0 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x88 ... 0x8F | Reserved | | | | | | | | | |
| 0x90 | XDMAC_CIE1 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x94 | XDMAC_CID1 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | XDMAC_CIM1 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x9C | XDMAC_CIS1 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA0 | XDMAC_CSA1 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0xA4 | XDMAC_CDA1 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0xA8 | XDMAC_CNDA1 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0xAC | XDMAC_CNDC1 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB0 | XDMAC_CUBC1 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB4 | XDMAC_CBC1 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB8 | XDMAC_CC1 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xBC | XDMAC_CDS_MSP1 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0xC0 | XDMAC_CSUS1 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC4 | XDMAC_CDUS1 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC8 ... 0xCF | Reserved | | | | | | | | | |
| 0xD0 | XDMAC_CIE2 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD4 | XDMAC_CID2 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD8 | XDMAC_CIM2 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xDC | XDMAC_CIS2 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xE0 | XDMAC_CSA2 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0xE4 | XDMAC_CDA2 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0xE8 | XDMAC_CNDA2 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0xEC | XDMAC_CNDC2 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF0 | XDMAC_CUBC2 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF4 | XDMAC_CBC2 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF8 | XDMAC_CC2 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xFC | XDMAC_CDS_MSP2 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0100 | XDMAC_CSUS2 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0104 | XDMAC_CDUS2 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0108 ... 0x010F | Reserved | | | | | | | | | |
| 0x0110 | XDMAC_CIE3 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0114 | XDMAC_CID3 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0118 | XDMAC_CIM3 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x011C | XDMAC_CIS3 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0120 | XDMAC_CSA3 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0124 | XDMAC_CDA3 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0128 | XDMAC_CNDA3 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x012C | XDMAC_CNDC3 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0130 | XDMAC_CUBC3 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0134 | XDMAC_CBC3 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0138 | XDMAC_CC3 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x013C | XDMAC_CDS_MSP3 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0140 | XDMAC_CSUS3 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0144 | XDMAC_CDUS3 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0148 ... 0x014F | Reserved | | | | | | | | | |
| 0x0150 | XDMAC_CIE4 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0154 | XDMAC_CID4 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | XDMAC_CIM4 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x015C | XDMAC_CIS4 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0160 | XDMAC_CSA4 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0164 | XDMAC_CDA4 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0168 | XDMAC_CNDA4 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x016C | XDMAC_CNDC4 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0170 | XDMAC_CUBC4 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0174 | XDMAC_CBC4 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0178 | XDMAC_CC4 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x017C | XDMAC_CDS_MSP4 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0180 | XDMAC_CSUS4 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0184 | XDMAC_CDUS4 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0188 ... 0x018F | Reserved | | | | | | | | | |
| 0x0190 | XDMAC_CIE5 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0194 | XDMAC_CID5 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0198 | XDMAC_CIM5 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x019C | XDMAC_CIS5 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A0 | XDMAC_CSA5 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x01A4 | XDMAC_CDA5 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x01A8 | XDMAC_CNDA5 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x01AC | XDMAC_CNDC5 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B0 | XDMAC_CUBC5 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B4 | XDMAC_CBC5 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B8 | XDMAC_CC5 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01BC | XDMAC_CDS_MSP5 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x01C0 | XDMAC_CSUS5 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01C4 | XDMAC_CDUS5 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01C8 ... 0x01CF | Reserved | | | | | | | | | |
| 0x01D0 | XDMAC_CIE6 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01D4 | XDMAC_CID6 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01D8 | XDMAC_CIM6 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01DC | XDMAC_CIS6 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01E0 | XDMAC_CSA6 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x01E4 | XDMAC_CDA6 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x01E8 | XDMAC_CNDA6 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x01EC | XDMAC_CNDC6 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01F0 | XDMAC_CUBC6 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01F4 | XDMAC_CBC6 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01F8 | XDMAC_CC6 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01FC | XDMAC_CDS_MSP6 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0200 | XDMAC_CSUS6 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0204 | XDMAC_CDUS6 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0208 ... 0x020F | Reserved | | | | | | | | | |
| 0x0210 | XDMAC_CIE7 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0214 | XDMAC_CID7 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0218 | XDMAC_CIM7 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x021C | XDMAC_CIS7 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0220 | XDMAC_CSA7 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0224 | XDMAC_CDA7 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0228 | XDMAC_CNDA7 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x022C | XDMAC_CNDC7 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0230 | XDMAC_CUBC7 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0234 | XDMAC_CBC7 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0238 | XDMAC_CC7 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x023C | XDMAC_CDS_MSP7 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0240 | XDMAC_CSUS7 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0244 | XDMAC_CDUS7 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0248 ... 0x024F | Reserved | | | | | | | | | |
| 0x0250 | XDMAC_CIE8 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0254 | XDMAC_CID8 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0258 | XDMAC_CIM8 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x025C | XDMAC_CIS8 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0260 | XDMAC_CSA8 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0264 | XDMAC_CDA8 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0268 | XDMAC_CNDA8 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x026C | XDMAC_CNDC8 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0270 | XDMAC_CUBC8 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0274 | XDMAC_CBC8 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0278 | XDMAC_CC8 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x027C | XDMAC_CDS_MSP8 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0280 | XDMAC_CSUS8 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0284 | XDMAC_CDUS8 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0288 ... 0x028F | Reserved | | | | | | | | | |
| 0x0290 | XDMAC_CIE9 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0294 | XDMAC_CID9 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0298 | XDMAC_CIM9 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x029C | XDMAC_CIS9 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02A0 | XDMAC_CSA9 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x02A4 | XDMAC_CDA9 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x02A8 | XDMAC_CNDA9 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x02AC | XDMAC_CNDC9 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B0 | XDMAC_CUBC9 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B4 | XDMAC_CBC9 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02B8 | XDMAC_CC9 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02BC | XDMAC_CDS_MSP9 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x02C0 | XDMAC_CSUS9 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02C4 | XDMAC_CDUS9 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02C8 ... 0x02CF | Reserved | | | | | | | | | |
| 0x02D0 | XDMAC_CIE10 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02D4 | XDMAC_CID10 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02D8 | XDMAC_CIM10 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02DC | XDMAC_CIS10 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02E0 | XDMAC_CSA10 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x02E4 | XDMAC_CDA10 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x02E8 | XDMAC_CNDA10 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x02EC | XDMAC_CNDC10 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F0 | XDMAC_CUBC10 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F4 | XDMAC_CBC10 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x02F8 | XDMAC_CC10 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x02FC | XDMAC_CDS_MSP10 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0300 | XDMAC_CSUS10 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0304 | XDMAC_CDUS10 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0308 ... 0x030F | Reserved | | | | | | | | | |
| 0x0310 | XDMAC_CIE11 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0314 | XDMAC_CID11 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0318 | XDMAC_CIM11 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x031C | XDMAC_CIS11 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0320 | XDMAC_CSA11 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0324 | XDMAC_CDA11 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0328 | XDMAC_CNDA11 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x032C | XDMAC_CNDC11 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0330 | XDMAC_CUBC11 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0334 | XDMAC_CBC11 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0338 | XDMAC_CC11 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x033C | XDMAC_CDS_MSP11 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0340 | XDMAC_CSUS11 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0344 | XDMAC_CDUS11 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0348 ... 0x034F | Reserved | | | | | | | | | |
| 0x0350 | XDMAC_CIE12 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0354 | XDMAC_CID12 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0358 | XDMAC_CIM12 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x035C | XDMAC_CIS12 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0360 | XDMAC_CSA12 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0364 | XDMAC_CDA12 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0368 | XDMAC_CNDA12 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x036C | XDMAC_CNDC12 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0370 | XDMAC_CUBC12 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0374 | XDMAC_CBC12 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0378 | XDMAC_CC12 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x037C | XDMAC_CDS_MSP12 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0380 | XDMAC_CSUS12 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0384 | XDMAC_CDUS12 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0388 ... 0x038F | Reserved | | | | | | | | | |
| 0x0390 | XDMAC_CIE13 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0394 | XDMAC_CID13 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0398 | XDMAC_CIM13 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x039C | XDMAC_CIS13 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03A0 | XDMAC_CSA13 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x03A4 | XDMAC_CDA13 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x03A8 | XDMAC_CNDA13 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x03AC | XDMAC_CNDC13 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B0 | XDMAC_CUBC13 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B4 | XDMAC_CBC13 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03B8 | XDMAC_CC13 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x03BC | XDMAC_CDS_MSP13 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x03C0 | XDMAC_CSUS13 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C4 | XDMAC_CDUS13 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C8 ... 0x03CF | Reserved | | | | | | | | | |
| 0x03D0 | XDMAC_CIE14 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03D4 | XDMAC_CID14 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03D8 | XDMAC_CIM14 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03DC | XDMAC_CIS14 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03E0 | XDMAC_CSA14 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x03E4 | XDMAC_CDA14 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x03E8 | XDMAC_CNDA14 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x03EC | XDMAC_CNDC14 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03F0 | XDMAC_CUBC14 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03F4 | XDMAC_CBC14 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03F8 | XDMAC_CC14 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x03FC | XDMAC_CDS_MSP14 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0400 | XDMAC_CSUS14 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0404 | XDMAC_CDUS14 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0408 ... 0x040F | Reserved | | | | | | | | | |
| 0x0410 | XDMAC_CIE15 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0414 | XDMAC_CID15 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0418 | XDMAC_CIM15 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x041C | XDMAC_CIS15 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0420 | XDMAC_CSA15 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0424 | XDMAC_CDA15 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0428 | XDMAC_CNDA15 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x042C | XDMAC_CNDC15 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0430 | XDMAC_CUBC15 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0434 | XDMAC_CBC15 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0438 | XDMAC_CC15 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x043C | XDMAC_CDS_MSP15 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0440 | XDMAC_CSUS15 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0444 | XDMAC_CDUS15 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0448 ... 0x044F | Reserved | | | | | | | | | |
| 0x0450 | XDMAC_CIE16 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0454 | XDMAC_CID16 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0458 | XDMAC_CIM16 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x045C | XDMAC_CIS16 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0460 | XDMAC_CSA16 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0464 | XDMAC_CDA16 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0468 | XDMAC_CNDA16 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x046C | XDMAC_CNDC16 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0470 | XDMAC_CUBC16 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0474 | XDMAC_CBC16 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0478 | XDMAC_CC16 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x047C | XDMAC_CDS_MSP16 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0480 | XDMAC_CSUS16 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0484 | XDMAC_CDUS16 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0488 ... 0x048F | Reserved | | | | | | | | | |
| 0x0490 | XDMAC_CIE17 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0494 | XDMAC_CID17 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0498 | XDMAC_CIM17 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x049C | XDMAC_CIS17 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04A0 | XDMAC_CSA17 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x04A4 | XDMAC_CDA17 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x04A8 | XDMAC_CNDA17 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x04AC | XDMAC_CNDC17 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04B0 | XDMAC_CUBC17 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04B4 | XDMAC_CBC17 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04B8 | XDMAC_CC17 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x04BC | XDMAC_CDS_MSP17 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x04C0 | XDMAC_CSUS17 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04C4 | XDMAC_CDUS17 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04C8 ... 0x04CF | Reserved | | | | | | | | | |
| 0x04D0 | XDMAC_CIE18 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04D4 | XDMAC_CID18 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04D8 | XDMAC_CIM18 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04DC | XDMAC_CIS18 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04E0 | XDMAC_CSA18 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x04E4 | XDMAC_CDA18 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x04E8 | XDMAC_CNDA18 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x04EC | XDMAC_CNDC18 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04F0 | XDMAC_CUBC18 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04F4 | XDMAC_CBC18 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04F8 | XDMAC_CC18 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x04FC | XDMAC_CDS_MSP18 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0500 | XDMAC_CSUS18 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0504 | XDMAC_CDUS18 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0508 ... 0x050F | Reserved | | | | | | | | | |
| 0x0510 | XDMAC_CIE19 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0514 | XDMAC_CID19 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0518 | XDMAC_CIM19 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x051C | XDMAC_CIS19 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0520 | XDMAC_CSA19 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0524 | XDMAC_CDA19 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0528 | XDMAC_CNDA19 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x052C | XDMAC_CNDC19 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0530 | XDMAC_CUBC19 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0534 | XDMAC_CBC19 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0538 | XDMAC_CC19 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x053C | XDMAC_CDS_MSP19 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0540 | XDMAC_CSUS19 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0544 | XDMAC_CDUS19 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0548 ... 0x054F | Reserved | | | | | | | | | |
| 0x0550 | XDMAC_CIE20 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0554 | XDMAC_CID20 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0558 | XDMAC_CIM20 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x055C | XDMAC_CIS20 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0560 | XDMAC_CSA20 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0564 | XDMAC_CDA20 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0568 | XDMAC_CNDA20 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x056C | XDMAC_CNDC20 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0570 | XDMAC_CUBC20 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0574 | XDMAC_CBC20 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0578 | XDMAC_CC20 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x057C | XDMAC_CDS_MSP20 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0580 | XDMAC_CSUS20 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0584 | XDMAC_CDUS20 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0588 ... 0x058F | Reserved | | | | | | | | | |
| 0x0590 | XDMAC_CIE21 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0594 | XDMAC_CID21 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0598 | XDMAC_CIM21 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x059C | XDMAC_CIS21 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05A0 | XDMAC_CSA21 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x05A4 | XDMAC_CDA21 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x05A8 | XDMAC_CNDA21 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x05AC | XDMAC_CNDC21 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05B0 | XDMAC_CUBC21 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05B4 | XDMAC_CBC21 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05B8 | XDMAC_CC21 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x05BC | XDMAC_CDS_MSP21 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x05C0 | XDMAC_CSUS21 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05C4 | XDMAC_CDUS21 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05C8 ... 0x05CF | Reserved | | | | | | | | | |
| 0x05D0 | XDMAC_CIE22 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05D4 | XDMAC_CID22 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05D8 | XDMAC_CIM22 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05DC | XDMAC_CIS22 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05E0 | XDMAC_CSA22 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x05E4 | XDMAC_CDA22 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x05E8 | XDMAC_CNDA22 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x05EC | XDMAC_CNDC22 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05F0 | XDMAC_CUBC22 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05F4 | XDMAC_CBC22 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05F8 | XDMAC_CC22 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x05FC | XDMAC_CDS_MSP22 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0600 | XDMAC_CSUS22 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0604 | XDMAC_CDUS22 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0608 ... 0x060F | Reserved | | | | | | | | | |
| 0x0610 | XDMAC_CIE23 | 7:0 | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0614 | XDMAC_CID23 | 7:0 | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0618 | XDMAC_CIM23 | 7:0 | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x061C | XDMAC_CIS23 | 7:0 | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0620 | XDMAC_CSA23 | 7:0 | SA[7:0] | | | | | | | |
| | | 15:8 | SA[15:8] | | | | | | | |
| | | 23:16 | SA[23:16] | | | | | | | |
| | | 31:24 | SA[31:24] | | | | | | | |
| 0x0624 | XDMAC_CDA23 | 7:0 | DA[7:0] | | | | | | | |
| | | 15:8 | DA[15:8] | | | | | | | |
| | | 23:16 | DA[23:16] | | | | | | | |
| | | 31:24 | DA[31:24] | | | | | | | |
| 0x0628 | XDMAC_CNDA23 | 7:0 | NDA[5:0] | | | | | | | NDAIF |
| | | 15:8 | NDA[13:6] | | | | | | | |
| | | 23:16 | NDA[21:14] | | | | | | | |
| | | 31:24 | NDA[29:22] | | | | | | | |
| 0x062C | XDMAC_CNDC23 | 7:0 | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0630 | XDMAC_CUBC23 | 7:0 | UBLEN[7:0] | | | | | | | |
| | | 15:8 | UBLEN[15:8] | | | | | | | |
| | | 23:16 | UBLEN[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0634 | XDMAC_CBC23 | 7:0 | BLEN[7:0] | | | | | | | |
| | | 15:8 | | | | | BLEN[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0638 | XDMAC_CC23 | 7:0 | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| | | 15:8 | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| | | 23:16 | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| | | 31:24 | | PERID[6:0] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x063C | XDMAC_CDS_MSP23 | 7:0 | SDS_MSP[7:0] | | | | | | | |
| | | 15:8 | SDS_MSP[15:8] | | | | | | | |
| | | 23:16 | DDS_MSP[7:0] | | | | | | | |
| | | 31:24 | DDS_MSP[15:8] | | | | | | | |
| 0x0640 | XDMAC_CSUS23 | 7:0 | SUBS[7:0] | | | | | | | |
| | | 15:8 | SUBS[15:8] | | | | | | | |
| | | 23:16 | SUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0644 | XDMAC_CDUS23 | 7:0 | DUBS[7:0] | | | | | | | |
| | | 15:8 | DUBS[15:8] | | | | | | | |
| | | 23:16 | DUBS[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 36.9.1 XDMAC Global Type Register

**Name:** XDMAC_GTYPE
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | NB_REQ[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FIFO_SZ[10:3] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FIFO_SZ[2:0] | | | NB_CH[4:0] | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 22:16 – NB_REQ[6:0]** Number of Peripheral Requests Minus One

**Bits 15:5 – FIFO_SZ[10:0]** Number of Bytes

**Bits 4:0 – NB_CH[4:0]** Number of Channels Minus One

### 36.9.2 XDMAC Global Configuration Register

**Name:** XDMAC_GCFG
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | BXKBEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CGDISIF | CGDISFIFO | CGDISPIPE | CGDISREG |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 8 – BXKBEN** Boundary X Kilobyte Enable

| Value | Description |
|---|---|
| 0 | The 1 Kbyte boundary is used. |
| 1 | The controller does not meet the AHB specification. |

**Bit 3 – CGDISIF** Bus Interface Clock Gating Disable

| Value | Description |
|---|---|
| 0 | The automatic clock gating is enabled for the system bus interface. |
| 1 | The automatic clock gating is disabled for the system bus interface. |

**Bit 2 – CGDISFIFO** FIFO Clock Gating Disable

| Value | Description |
|---|---|
| 0 | The automatic clock gating is enabled for the main FIFO. |
| 1 | The automatic clock gating is disabled for the main FIFO. |

**Bit 1 – CGDISPIPE** Pipeline Clock Gating Disable

| Value | Description |
|---|---|
| 0 | The automatic clock gating is enabled for the main pipeline. |
| 1 | The automatic clock gating is disabled for the main pipeline. |

**Bit 0 – CGDISREG** Configuration Registers Clock Gating Disable

| Value | Description |
|---|---|
| 0 | The automatic clock gating is enabled for the configuration registers. |
| 1 | The automatic clock gating is disabled for the configuration registers. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 36.9.3 XDMAC Global Weighted Arbiter Configuration Register

**Name:** XDMAC_GWAC
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PW3[3:0] | | | | PW2[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PW1[3:0] | | | | PW0[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:12 – PW3[3:0]** Pool Weight 3
This field indicates the weight of pool 3 in the arbitration scheme of the DMA scheduler.

**Bits 11:8 – PW2[3:0]** Pool Weight 2
This field indicates the weight of pool 2 in the arbitration scheme of the DMA scheduler.

**Bits 7:4 – PW1[3:0]** Pool Weight 1
This field indicates the weight of pool 1 in the arbitration scheme of the DMA scheduler.

**Bits 3:0 – PW0[3:0]** Pool Weight 0
This field indicates the weight of pool 0 in the arbitration scheme of the DMA scheduler.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 36.9.4 XDMAC Global Interrupt Enable Register

**Name:** XDMAC_GIE
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IE23 | IE22 | IE21 | IE20 | IE19 | IE18 | IE17 | IE16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IE15 | IE14 | IE13 | IE12 | IE11 | IE10 | IE9 | IE8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IE7 | IE6 | IE5 | IE4 | IE3 | IE2 | IE1 | IE0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – IE** XDMAC Channel x Interrupt Enable

| Value | Description |
|---|---|
| 0 | This bit has no effect. The Channel x Interrupt Mask bit (XDMAC_GIM.IMx) is not modified. |
| 1 | The corresponding mask bit is set. The XDMAC Channel x Interrupt Status register (XDMAC_GIS) can generate an interrupt. |

### 36.9.5 XDMAC Global Interrupt Disable Register

**Name:** XDMAC_GID
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ID23 | ID22 | ID21 | ID20 | ID19 | ID18 | ID17 | ID16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ID15 | ID14 | ID13 | ID12 | ID11 | ID10 | ID9 | ID8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ID7 | ID6 | ID5 | ID4 | ID3 | ID2 | ID1 | ID0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – ID** XDMAC Channel x Interrupt Disable

| Value | Description |
|---|---|
| 0 | This bit has no effect. The Channel x Interrupt Mask bit (XDMAC_GIM.IMx) is not modified. |
| 1 | The corresponding mask bit is reset. The Channel x Interrupt Status register interrupt (XDMAC_GIS) is masked. |

### 36.9.6 XDMAC Global Interrupt Mask Register

**Name:** XDMAC_GIM
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IM23 | IM22 | IM21 | IM20 | IM19 | IM18 | IM17 | IM16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IM15 | IM14 | IM13 | IM12 | IM11 | IM10 | IM9 | IM8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IM7 | IM6 | IM5 | IM4 | IM3 | IM2 | IM1 | IM0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – IM** XDMAC Channel x Interrupt Mask

| Value | Description |
|---|---|
| 0 | This bit indicates that the channel x interrupt source is masked. The interrupt line is not raised. |
| 1 | This bit indicates that the channel x interrupt source is unmasked. |

### 36.9.7 XDMAC Global Interrupt Status Register

**Name:** XDMAC_GIS
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IS23 | IS22 | IS21 | IS20 | IS19 | IS18 | IS17 | IS16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IS15 | IS14 | IS13 | IS12 | IS11 | IS10 | IS9 | IS8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IS7 | IS6 | IS5 | IS4 | IS3 | IS2 | IS1 | IS0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – IS** XDMAC Channel x Interrupt Status

| Value | Description |
|---|---|
| 0 | This bit indicates that either the interrupt source is masked at the channel level or no interrupt is pending for channel x. |
| 1 | This bit indicates that an interrupt is pending for the channel x. |

### 36.9.8 XDMAC Global Channel Enable Register

**Name:** XDMAC_GE
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EN23 | EN22 | EN21 | EN20 | EN19 | EN18 | EN17 | EN16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EN15 | EN14 | EN13 | EN12 | EN11 | EN10 | EN9 | EN8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EN7 | EN6 | EN5 | EN4 | EN3 | EN2 | EN1 | EN0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – EN** XDMAC Channel x Enable

| Value | Description |
|---|---|
| 0 | This bit has no effect. |
| 1 | Enables channel n. This operation is permitted if the Channel x Status bit (XDMAC_GS.STx) was read as '0'. |

### 36.9.9 XDMAC Global Channel Disable Register

**Name:** XDMAC_GD
**Offset:** 0x20
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DI23 | DI22 | DI21 | DI20 | DI19 | DI18 | DI17 | DI16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DI15 | DI14 | DI13 | DI12 | DI11 | DI10 | DI9 | DI8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DI7 | DI6 | DI5 | DI4 | DI3 | DI2 | DI1 | DI0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – DI** XDMAC Channel x Disable

| Value | Description |
|---|---|
| 0 | This bit has no effect. |
| 1 | Disables channel x. |

### 36.9.10 XDMAC Global Channel Status Register

**Name:** XDMAC_GS
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ST23 | ST22 | ST21 | ST20 | ST19 | ST18 | ST17 | ST16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ST15 | ST14 | ST13 | ST12 | ST11 | ST10 | ST9 | ST8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ST7 | ST6 | ST5 | ST4 | ST3 | ST2 | ST1 | ST0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – ST** XDMAC Channel x Status

| Value | Description |
|---|---|
| 0 | This bit indicates that the channel x is disabled. |
| 1 | This bit indicates that the channel x is enabled. If a channel disable request is issued, this bit remains asserted until pending transaction is completed. |

### 36.9.11 XDMAC Global Channel Read Suspend Register

**Name:** XDMAC_GRS
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RS23 | RS22 | RS21 | RS20 | RS19 | RS18 | RS17 | RS16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RS15 | RS14 | RS13 | RS12 | RS11 | RS10 | RS9 | RS8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RS7 | RS6 | RS5 | RS4 | RS3 | RS2 | RS1 | RS0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – RSx** XDMAC Channel x Read Suspend

| Value | Description |
|---|---|
| 0 | The read channel is not suspended. |
| 1 | The source requests for channel n are no longer serviced by the system scheduler. |

### 36.9.12 XDMAC Global Channel Write Suspend Register

**Name:** XDMAC_GWS
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WS23 | WS22 | WS21 | WS20 | WS19 | WS18 | WS17 | WS16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WS15 | WS14 | WS13 | WS12 | WS11 | WS10 | WS9 | WS8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WS7 | WS6 | WS5 | WS4 | WS3 | WS2 | WS1 | WS0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – WSx** XDMAC Channel x Write Suspend

| Value | Description |
|---|---|
| 0 | The write channel is not suspended. |
| 1 | Destination requests are no longer routed to the scheduler. |

### 36.9.13 XDMAC Global Channel Read Write Suspend Register

**Name:** XDMAC_GRWS
**Offset:** 0x30
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RWS23 | RWS22 | RWS21 | RWS20 | RWS19 | RWS18 | RWS17 | RWS16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWS15 | RWS14 | RWS13 | RWS12 | RWS11 | RWS10 | RWS9 | RWS8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RWS7 | RWS6 | RWS5 | RWS4 | RWS3 | RWS2 | RWS1 | RWS0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – RWSx** XDMAC Channel x Read Write Suspend

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Read and write requests are suspended. |

### 36.9.14 XDMAC Global Channel Read Write Resume Register

**Name:** XDMAC_GRWR
**Offset:** 0x34
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RWR23 | RWR22 | RWR21 | RWR20 | RWR19 | RWR18 | RWR17 | RWR16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWR15 | RWR14 | RWR13 | RWR12 | RWR11 | RWR10 | RWR9 | RWR8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RWR7 | RWR6 | RWR5 | RWR4 | RWR3 | RWR2 | RWR1 | RWR0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – RWRx** XDMAC Channel x Read Write Resume

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Read and write requests are serviced. |

### 36.9.15 XDMAC Global Channel Software Request Register

**Name:** XDMAC_GSWR
**Offset:** 0x38
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SWREQ23 | SWREQ22 | SWREQ21 | SWREQ20 | SWREQ19 | SWREQ18 | SWREQ17 | SWREQ16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWREQ15 | SWREQ14 | SWREQ13 | SWREQ12 | SWREQ11 | SWREQ10 | SWREQ9 | SWREQ8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWREQ7 | SWREQ6 | SWREQ5 | SWREQ4 | SWREQ3 | SWREQ2 | SWREQ1 | SWREQ0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – SWREQ** XDMAC Channel x Software Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Requests a DMA transfer for channel x. |

### 36.9.16 XDMAC Global Channel Software Request Status Register

**Name:** XDMAC_GSWS
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SWRS23 | SWRS22 | SWRS21 | SWRS20 | SWRS19 | SWRS18 | SWRS17 | SWRS16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWRS15 | SWRS14 | SWRS13 | SWRS12 | SWRS11 | SWRS10 | SWRS9 | SWRS8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRS7 | SWRS6 | SWRS5 | SWRS4 | SWRS3 | SWRS2 | SWRS1 | SWRS0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – SWRS** XDMAC Channel x Software Request Status

| Value | Description |
|---|---|
| 0 | Channel x source request is serviced. |
| 1 | Channel x source request is pending. |

### 36.9.17 XDMAC Global Channel Software Flush Request Register

**Name:** XDMAC_GSWF
**Offset:** 0x40
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SWF23 | SWF22 | SWF21 | SWF20 | SWF19 | SWF18 | SWF17 | SWF16 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWF15 | SWF14 | SWF13 | SWF12 | SWF11 | SWF10 | SWF9 | SWF8 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWF7 | SWF6 | SWF5 | SWF4 | SWF3 | SWF2 | SWF1 | SWF0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 – SWFx** XDMAC Channel x Software Flush Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Requests a DMA transfer flush for channel x. This bit is only relevant when the transfer is source peripheral synchronized. |

### 36.9.18 XDMAC Channel x Interrupt Enable Register [x=0..23]

**Name:** XDMAC_CIE
**Offset:** 0x50 + n*0x40 [n=0..23]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ROIE | WBIE | RBIE | FIE | DIE | LIE | BIE |
| Access | | W | W | W | W | W | W | W |
| Reset | | – | – | – | – | – | – | – |

**Bit 6 – ROIE** Request Overflow Error Interrupt Enable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables request overflow error interrupt. |

**Bit 5 – WBIE** Write Bus Error Interrupt Enable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables write bus error interrupt. |

**Bit 4 – RBIE** Read Bus Error Interrupt Enable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables read bus error interrupt. |

**Bit 3 – FIE** End of Flush Interrupt Enable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables end of flush interrupt. |

**Bit 2 – DIE** End of Disable Interrupt Enable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables end of disable interrupt. |

**Bit 1 – LIE** End of Linked List Interrupt Enable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables end of linked list interrupt. |

**Bit 0 – BIE** End of Block Interrupt Enable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables end of block interrupt. |

### 36.9.19 XDMAC Channel x Interrupt Disable Register [x = 0..23]

**Name:** XDMAC_CID
**Offset:** 0x54 + n*0x40 [n=0..23]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ROID | WBEID | RBEID | FID | DID | LID | BID |
| Access | | W | W | W | W | W | W | W |
| Reset | | – | – | – | – | – | – | – |

**Bit 6 – ROID** Request Overflow Error Interrupt Disable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables request overflow error interrupt. |

**Bit 5 – WBEID** Write Bus Error Interrupt Disable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables bus error interrupt. |

**Bit 4 – RBEID** Read Bus Error Interrupt Disable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables bus error interrupt. |

**Bit 3 – FID** End of Flush Interrupt Disable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables end of flush interrupt. |

**Bit 2 – DID** End of Disable Interrupt Disable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables end of disable interrupt. |

**Bit 1 – LID** End of Linked List Interrupt Disable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables end of linked list interrupt. |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 0 – BID** End of Block Interrupt Disable Bit

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables end of block interrupt. |

### 36.9.20 XDMAC Channel x Interrupt Mask Register [x = 0..23]

**Name:** XDMAC_CIM
**Offset:** 0x58 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ROIM | WBEIM | RBEIM | FIM | DIM | LIM | BIM |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – ROIM** Request Overflow Error Interrupt Mask Bit

| Value | Description |
|---|---|
| 0 | Request overflow interrupt is masked. |
| 1 | Request overflow interrupt is activated. |

**Bit 5 – WBEIM** Write Bus Error Interrupt Mask Bit

| Value | Description |
|---|---|
| 0 | Bus error interrupt is masked. |
| 1 | Bus error interrupt is activated. |

**Bit 4 – RBEIM** Read Bus Error Interrupt Mask Bit

| Value | Description |
|---|---|
| 0 | Bus error interrupt is masked. |
| 1 | Bus error interrupt is activated. |

**Bit 3 – FIM** End of Flush Interrupt Mask Bit

| Value | Description |
|---|---|
| 0 | End of flush interrupt is masked. |
| 1 | End of flush interrupt is activated. |

**Bit 2 – DIM** End of Disable Interrupt Mask Bit

| Value | Description |
|---|---|
| 0 | End of disable interrupt is masked. |
| 1 | End of disable interrupt is activated. |

**Bit 1 – LIM** End of Linked List Interrupt Mask Bit

| Value | Description |
|---|---|
| 0 | End of linked list interrupt is masked. |
| 1 | End of linked list interrupt is activated. |

**Bit 0 – BIM** End of Block Interrupt Mask Bit

| Value | Description |
|---|---|
| 0 | Block interrupt is masked. |
| 1 | Block interrupt is activated. |

### 36.9.21 XDMAC Channel x Interrupt Status Register [x = 0..23]

**Name:** XDMAC_CIS
**Offset:** 0x5C + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | ROIS | WBEIS | RBEIS | FIS | DIS | LIS | BIS |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – ROIS** Request Overflow Error Interrupt Status Bit

| Value | Description |
|---|---|
| 0 | Overflow condition has not occurred. |
| 1 | Overflow condition has occurred at least once. (This information is only relevant for peripheral synchronized transfers.) |

**Bit 5 – WBEIS** Write Bus Error Interrupt Status Bit

| Value | Description |
|---|---|
| 0 | Write bus error condition has not occurred. |
| 1 | At least one bus error has been detected in a write access since the last read of the Status register. |

**Bit 4 – RBEIS** Read Bus Error Interrupt Status Bit

| Value | Description |
|---|---|
| 0 | Read bus error condition has not occurred. |
| 1 | At least one bus error has been detected in a read access since the last read of the Status register. |

**Bit 3 – FIS** End of Flush Interrupt Status Bit

| Value | Description |
|---|---|
| 0 | End of flush condition has not occurred. |
| 1 | End of flush condition has occurred since the last read of the Status register. |

**Bit 2 – DIS** End of Disable Interrupt Status Bit

| Value | Description |
|---|---|
| 0 | End of disable condition has not occurred. |
| 1 | End of disable condition has occurred since the last read of the Status register. |

**Bit 1 – LIS** End of Linked List Interrupt Status Bit

| Value | Description |
|---|---|
| 0 | End of linked list condition has not occurred. |

| Value | Description |
|---|---|
| 1 | End of linked list condition has occurred since the last read of the Status register. |

**Bit 0 – BIS** End of Block Interrupt Status Bit

| Value | Description |
|---|---|
| 0 | End of block interrupt has not occurred. |
| 1 | End of block interrupt has occurred since the last read of the Status register. |

### 36.9.22 XDMAC Channel x Source Address Register [x = 0..23]

**Name:** XDMAC_CSA
**Offset:** 0x60 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – SA[31:0]** Channel x Source Address
Program this register with the source address of the DMA transfer.
A configuration error is generated when this address is not aligned with the transfer data size.

### 36.9.23 XDMAC Channel x Destination Address Register [x = 0..23]

**Name:** XDMAC_CDA
**Offset:** 0x64 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DA[31:0]** Channel x Destination Address
Program this register with the destination address of the DMA transfer.
A configuration error is generated when this address is not aligned with the transfer data size.

### 36.9.24 XDMAC Channel x Next Descriptor Address Register [x = 0..23]

**Name:** XDMAC_CNDA
**Offset:** 0x68 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NDA[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NDA[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NDA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NDA[5:0] | | | | | | | NDAIF |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

**Bits 31:2 – NDA[29:0]** Channel x Next Descriptor Address
The 30-bit width of the NDA field represents the next descriptor address range 31:2. The descriptor is word-aligned and the two least significant register bits 1:0 are ignored.

**Bit 0 – NDAIF** Channel x Next Descriptor Interface

| Value | Description |
|---|---|
| 0 | The channel descriptor is retrieved through system interface 0. |
| 1 | The channel descriptor is retrieved through system interface 1. |

### 36.9.25 XDMAC Channel x Next Descriptor Control Register [x = 0..23]

**Name:** XDMAC_CNDC
**Offset:** 0x6C + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | NDVIEW[1:0] | | NDDUP | NDSUP | NDE |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:3 – NDVIEW[1:0]** Channel x Next Descriptor View

| Value | Name | Description |
|---|---|---|
| 0 | NDV0 | Next Descriptor View 0 |
| 1 | NDV1 | Next Descriptor View 1 |
| 2 | NDV2 | Next Descriptor View 2 |
| 3 | NDV3 | Next Descriptor View 3 |

**Bit 2 – NDDUP** Channel x Next Descriptor Destination Update
0 (DST_PARAMS_UNCHANGED): Destination parameters remain unchanged.
1 (DST_PARAMS_UPDATED): Destination parameters are updated when the descriptor is retrieved.

**Bit 1 – NDSUP** Channel x Next Descriptor Source Update
0 (SRC_PARAMS_UNCHANGED): Source parameters remain unchanged.
1 (SRC_PARAMS_UPDATED): Source parameters are updated when the descriptor is retrieved.

**Bit 0 – NDE** Channel x Next Descriptor Enable
0 (DSCR_FETCH_DIS): Descriptor fetch is disabled.
1 (DSCR_FETCH_EN): Descriptor fetch is enabled.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 36.9.26 XDMAC Channel x Microblock Control Register [x = 0..23]

**Name:** XDMAC_CUBC
**Offset:** 0x70 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UBLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – UBLEN[23:0]** Channel x Microblock Length
This field indicates the number of data in the microblock. The microblock contains UBLEN data.

### 36.9.27 XDMAC Channel x Block Control Register [x = 0..23]

**Name:** XDMAC_CBC
**Offset:** 0x74 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BLEN[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – BLEN[11:0]** Channel x Block Length
The length of the block is (BLEN+1) microblocks.

### 36.9.28 XDMAC Channel x Configuration Register [x = 0..23]

**Name:** XDMAC_CC
**Offset:** 0x78 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PERID[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WRIP | RDIP | INITD | | DAM[1:0] | | SAM[1:0] | |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | DIF | SIF | DWIDTH[1:0] | | CSIZE[2:0] | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MEMSET | SWREQ | | DSYNC | | MBSIZE[1:0] | | TYPE |
| Access | R/W | R/W | | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | | 0 | | 0 | 0 | 0 |

**Bits 30:24 – PERID[6:0]** Channel x Peripheral Hardware Request Line Identifier
This field contains the peripheral hardware request line identifier. PERID refers to identifiers defined in "DMA Controller Peripheral Connections".

**Bit 23 – WRIP** Write in Progress (this bit is read-only)
0 (DONE): No active write transaction on the bus.
1 (IN_PROGRESS): A write transaction is in progress.

**Bit 22 – RDIP** Read in Progress (this bit is read-only)
0 (DONE): No active read transaction on the bus.
1 (IN_PROGRESS): A read transaction is in progress.

**Bit 21 – INITD** Channel Initialization Done (this bit is read-only)
0 (IN_PROGRESS): Channel initialization is in progress.
1 (TERMINATED): Channel initialization is completed.
Note: When set to 0, XDMAC_CUBC.UBLEN and XDMAC_CNDA.NDA field values are unreliable each time a descriptor is being updated. See 36.8 XDMAC Software Requirements.

**Bits 19:18 – DAM[1:0]** Channel x Destination Addressing Mode

| Value | Name | Description |
|---|---|---|
| 0 | FIXED_AM | The address remains unchanged. |
| 1 | INCREMENTED_AM | The addressing mode is incremented (the increment size is set to the data size). |
| 2 | UBS_AM | The microblock stride is added at the microblock boundary. |
| 3 | UBS_DS_AM | The microblock stride is added at the microblock boundary; the data stride is added at the data boundary. |

**Bits 17:16 – SAM[1:0]** Channel x Source Addressing Mode

| Value | Name | Description |
|---|---|---|
| 0 | FIXED_AM | The address remains unchanged. |

| Value | Name | Description |
|---|---|---|
| 1 | INCREMENTED_AM | The addressing mode is incremented (the increment size is set to the data size). |
| 2 | UBS_AM | The microblock stride is added at the microblock boundary. |
| 3 | UBS_DS_AM | The microblock stride is added at the microblock boundary, the data stride is added at the data boundary. |

**Bit 14 – DIF** Channel x Destination Interface Identifier
0 (AHB_IF0): The data is written through system bus interface 0.
1 (AHB_IF1): The data is written though system bus interface 1.

**Bit 13 – SIF** Channel x Source Interface Identifier
0 (AHB_IF0): The data is read through system bus interface 0.
1 (AHB_IF1): The data is read through system bus interface 1.

**Bits 12:11 – DWIDTH[1:0]** Channel x Data Width

| Value | Name | Description |
|---|---|---|
| 0 | BYTE | The data size is set to 8 bits |
| 1 | HALFWORD | The data size is set to 16 bits |
| 2 | WORD | The data size is set to 32 bits |

**Bits 10:8 – CSIZE[2:0]** Channel x Chunk Size

| Value | Name | Description |
|---|---|---|
| 0 | CHK_1 | 1 data transferred |
| 1 | CHK_2 | 2 data transferred |
| 2 | CHK_4 | 4 data transferred |
| 3 | CHK_8 | 8 data transferred |
| 4 | CHK_16 | 16 data transferred |

**Bit 7 – MEMSET** Channel x Fill Block of Memory
0 (NORMAL_MODE): Memset is not activated.
1 (HW_MODE): Sets the block of memory pointed by DA field to the specified value. This operation is performed on 8-, 16- or 32-bit basis.

**Bit 6 – SWREQ** Channel x Software Request Trigger
0 (HWR_CONNECTED): Hardware request line is connected to the peripheral request line.
1 (SWR_CONNECTED): Software request is connected to the peripheral request line.

**Bit 4 – DSYNC** Channel x Synchronization
0 (PER2MEM): Peripheral-to-memory transfer.
1 (MEM2PER): Memory-to-peripheral transfer.

**Bits 2:1 – MBSIZE[1:0]** Channel x Memory Burst Size

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE | The memory burst size is set to one. |
| 1 | FOUR | The memory burst size is set to four. |
| 2 | EIGHT | The memory burst size is set to eight. |
| 3 | SIXTEEN | The memory burst size is set to sixteen. |

**Bit 0 – TYPE** Channel x Transfer Type
0 (MEM_TRAN): Self-triggered mode (memory-to-memory transfer).
1 (PER_TRAN): Synchronized mode (peripheral-to-memory or memory-to-peripheral transfer).

© 2021 Microchip Technology Inc.
and its subsidiaries

### 36.9.29 XDMAC Channel x Data Stride Memory Set Pattern Register [x = 0..23]

**Name:** XDMAC_CDS_MSP
**Offset:** 0x7C + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DDS_MSP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DDS_MSP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SDS_MSP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SDS_MSP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – DDS_MSP[15:0]** Channel x Destination Data Stride or Memory Set Pattern
When XDMAC_CCx.MEMSET = 0, this field indicates the destination data stride.
When XDMAC_CCx.MEMSET = 1, this field indicates the memory set pattern.

**Bits 15:0 – SDS_MSP[15:0]** Channel x Source Data stride or Memory Set Pattern
When XDMAC_CCx.MEMSET = 0, this field indicates the source data stride.
When XDMAC_CCx.MEMSET = 1, this field indicates the memory set pattern.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 36.9.30 XDMAC Channel x Source Microblock Stride Register [x = 0..23]

**Name:** XDMAC_CSUS
**Offset:** 0x80 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SUBS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SUBS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SUBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – SUBS[23:0]** Channel x Source Microblock Stride
Two’s complement microblock stride for channel x.

### 36.9.31 XDMAC Channel x Destination Microblock Stride Register [x = 0..23]

**Name:** XDMAC_CDUS
**Offset:** 0x84 + n*0x40 [n=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DUBS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DUBS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DUBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – DUBS[23:0]** Channel x Destination Microblock Stride
Two’s complement microblock stride for channel x.

# 37. Image Sensor Interface (ISI)

## 37.1 Description

The Image Sensor Interface (ISI) connects a CMOS-type image sensor to the processor and provides image capture in various formats. The ISI performs data conversion, if necessary, before the storage in memory through DMA.

The ISI supports color CMOS image sensor and grayscale image sensors with a reduced set of functionalities.

In Grayscale mode, the data stream is stored in memory without any processing and so is not compatible with the LCD controller.

Internal FIFOs on the preview and codec paths are used to store the incoming data. The RGB output on the preview path is compatible with the LCD controller. This module outputs the data in RGB format (LCD compatible) and has scaling capabilities to make it compliant to the LCD display resolution (see the table RGB Format in Default Mode, RGB_CFG = 00, No Swap).

Several input formats such as preprocessed RGB or YCbCr are supported through the data bus interface.

The ISI supports two synchronization modes:

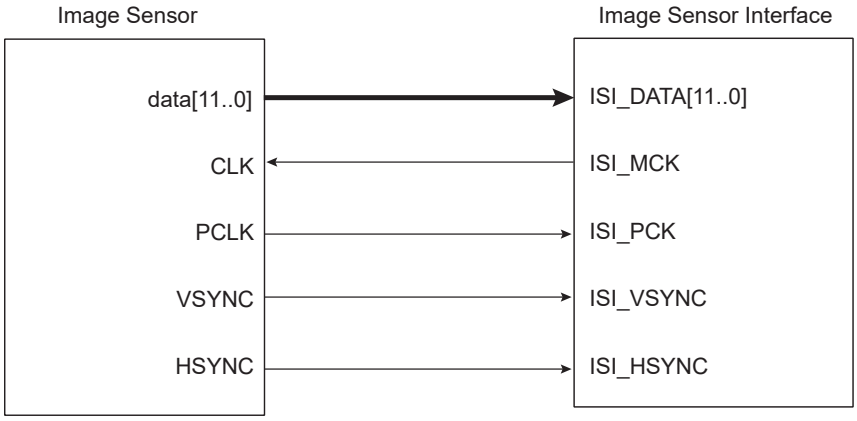
- Hardware with ISI_VSYNC and ISI_HSYNC signals
- International Telecommunication Union Recommendation ITU-R BT.656-4 Start-of-Active-Video (SAV) and End-of-Active-Video (EAV) synchronization sequence

Using EAV/SAV for synchronization reduces the pin count (ISI_VSYNC, ISI_HSYNC not used). The polarity of the synchronization pulse is programmable to comply with the sensor signals.

**Table 37-1. I/O Description**

| Signal | Direction | Description |
|---|---|---|
| ISI_VSYNC | In | Vertical Synchronization |
| ISI_HSYNC | In | Horizontal Synchronization |
| ISI_DATA[11..0] | In | Sensor Pixel Data |
| ISI_MCK | Out | Master Clock provided to the Image Sensor. Refer to “Clocks”. |
| ISI_PCK | In | Pixel Clock provided by the Image Sensor |

**Figure 37-1. ISI Connection Example**

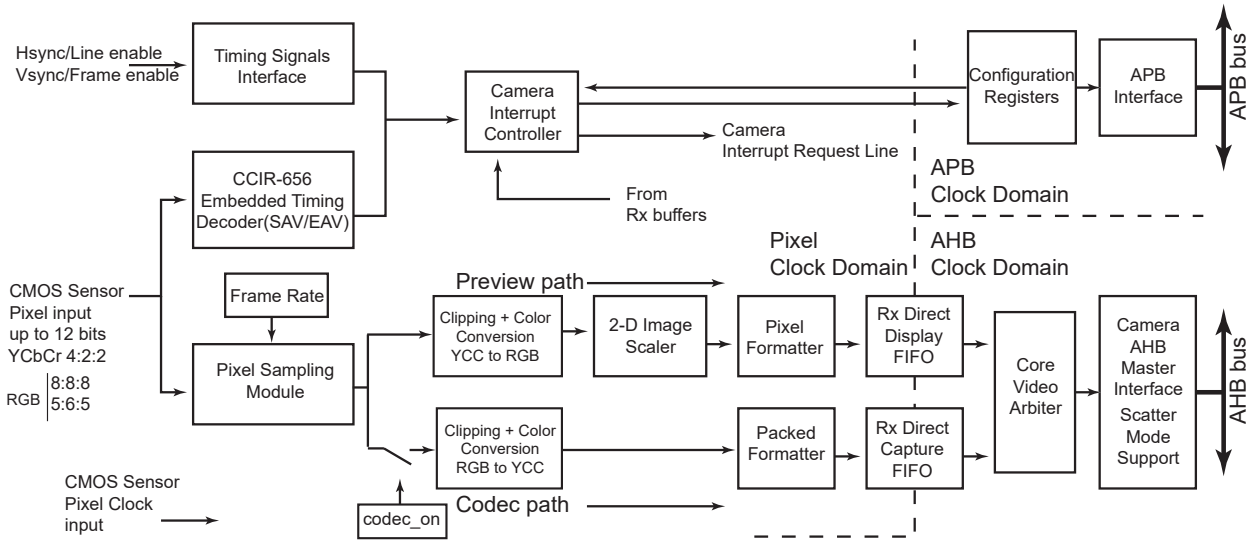


## 37.2 Embedded Characteristics

- ITU-R BT. 601/656 8-bit Mode External Interface Support
- Supports up to 12-bit Grayscale CMOS Sensors
- Support for ITU-R BT.656-4 SAV and EAV Synchronization
- Vertical and Horizontal Resolutions up to 2048 × 2048
- Preview Path up to 640 × 480 in RGB Mode
- Codec Path up to 2048 × 2048
- 16-byte FIFO on Codec Path
- 16-byte FIFO on Preview Path
- Support for Packed Data Formatting for YCbCr 4:2:2 Formats
- Preview Scaler to Generate Smaller Size image
- Programmable Frame Capture Rate
- VGA, QVGA, CIF, QCIF Formats Supported for LCD Preview
- Custom Formats with Horizontal and Vertical Preview Size as Multiples of 16 Also Supported for LCD Preview

## 37.3 Block Diagram

**Figure 37-2. ISI Block Diagram**




## 37.4 Product Dependencies

### 37.4.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the ISI pins to their peripheral functions.

### 37.4.2 Power Management

The ISI can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the ISI clock.

### 37.4.3 Interrupt Sources

The ISI interface has an interrupt line connected to the interrupt controller. Handling the ISI interrupt requires programming the interrupt controller before configuring the ISI.

## 37.5 Functional Description

The Image Sensor Interface (ISI) supports direct connection to the ITU-R BT. 601/656 8-bit mode compliant sensors and up to 12-bit grayscale sensors. It receives the image data stream from the image sensor on the 12-bit data bus.

This module receives up to 12 bits for data, the horizontal and vertical synchronizations and the pixel clock. The reduced pin count alternative for synchronization is supported for sensors that embed SAV (start of active video) and EAV (end of active video) delimiters in the data stream.

The Image Sensor Interface interrupt line is connected to the Advanced Interrupt Controller and can trigger an interrupt at the beginning of each frame and at the end of a DMA frame transfer. If the SAV/EAV synchronization is used, an interrupt can be triggered on each delimiter event.

For 8-bit color sensors, the data stream received can be in several possible formats: YCbCr 4:2:2, RGB 8:8:8, RGB 5:6:5 and may be processed before the storage in memory. When the preview DMA channel is configured and enabled, the preview path is activated and an ‘RGB frame’ is moved to memory. The preview path frame rate is configured with the FRATE field of the ISI_CFG1 register. When the codec DMA channel is configured and enabled, the codec path is activated and a ‘YCbCr 4:2:2 frame’ is captured as soon as the ISI_CDC bit of the ISI Control Register (ISI_CR) is set.

When the FULL bit of the ISI_CFG1 register is set, both preview DMA channel and codec DMA channel can operate simultaneously. When a zero is written to the FULL bit of the ISI_CFG1 register, a hardware scheduler checks the FRATE field. If its value is zero, a preview frame is skipped and a codec frame is moved to memory instead. If its value is other than zero, at least one free frame slot is available. The scheduler postpones the codec frame to that free available frame slot.

The data stream may be sent on both preview path and codec path if the value of bit ISI_CDC in the ISI_CR is one. To optimize the bandwidth, the codec path should be enabled only when a capture is required.

In Grayscale mode, the input data stream is stored in memory without any processing. The 12-bit data, which represent the grayscale level for the pixel, is stored in memory one or two pixels per word, depending on the GS_MODE bit in the ISI_CFG2 register. The codec datapath is not available when grayscale image is selected.

A frame rate counter allows users to capture all frames or 1 out of every 2 to 8 frames.

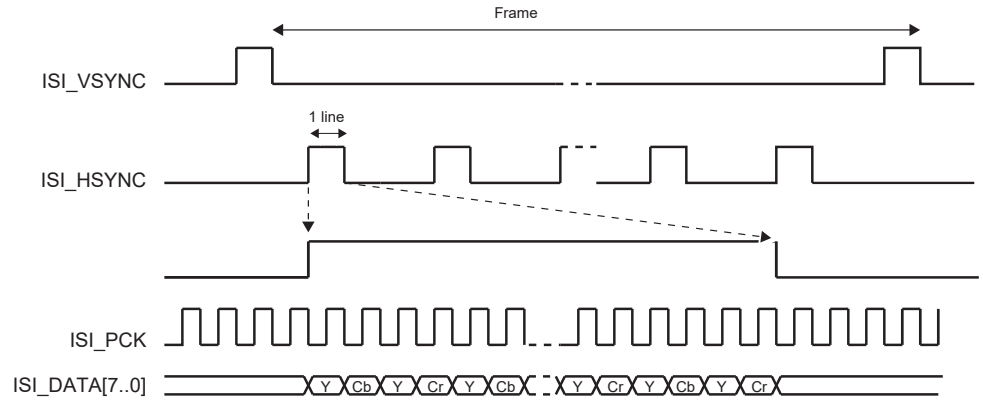
### 37.5.1 Data Timing

#### 37.5.1.1 VSYNC/HSYNC Data Timing

In the VSYNC/HSYNC synchronization, the valid data is captured with the active edge of the pixel clock (ISI_PCK), after SFD lines of vertical blanking and SLD pixel clock periods delay programmed in the ISI_CR.

The data timing using horizontal and vertical synchronization are shown in the following figure.

**Figure 37-3. HSYNC and VSYNC Synchronization**



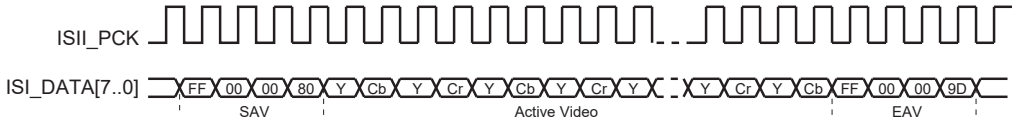

#### 37.5.1.2 SAV/EAV Data Timing

The ITU-RBT.656-4 standard defines the functional timing for an 8-bit wide interface.

There are two timing reference signals, one at the beginning of each video data block SAV (0xFF000080) and one at the end of each video data block EAV (0xFF00009D). Only data sent between EAV and SAV is captured. Horizontal blanking and vertical blanking are ignored. Use of the SAV and EAV synchronization eliminates the ISI_VSYNC and ISI_HSYNC signals from the interface, thereby reducing the pin count. In order to retrieve both frame and line synchronization properly, at least one line of vertical blanking is mandatory.

The data timing using EAV/SAV sequence synchronization are shown in the following figure.

**Figure 37-4. SAV and EAV Sequence Synchronization**




### 37.5.2 Data Ordering

The RGB color space format is required for viewing images on a display screen preview, and the YCbCr color space format is required for encoding.

All the sensors do not output the YCbCr or RGB components in the same order. The ISI allows the user to program the same component order as the sensor, reducing software treatments to restore the right format.

**Table 37-2. Data Ordering in YCbCr Mode**

| Mode | Byte 0 | Byte 1 | Byte 2 | Byte 3 |
|---|---|---|---|---|
| Default | Cb(i) | Y(i) | Cr(i) | Y(i+1) |
| Mode 1 | Cr(i) | Y(i) | Cb(i) | Y(i+1) |
| Mode 2 | Y(i) | Cb(i) | Y(i+1) | Cr(i) |
| Mode 3 | Y(i) | Cr(i) | Y(i+1) | Cb(i) |

**Table 37-3. RGB Format in Default Mode, RGB_CFG = 00, No Swap**

| Mode | Byte | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|---|---|
| RGB 8:8:8 | Byte 0 | R7(i) | R6(i) | R5(i) | R4(i) | R3(i) | R2(i) | R1(i) | R0(i) |
| | Byte 1 | G7(i) | G6(i) | G5(i) | G4(i) | G3(i) | G2(i) | G1(i) | G0(i) |
| | Byte 2 | B7(i) | B6(i) | B5(i) | B4(i) | B3(i) | B2(i) | B1(i) | B0(i) |
| | Byte 3 | R7(i+1) | R6(i+1) | R5(i+1) | R4(i+1) | R3(i+1) | R2(i+1) | R1(i+1) | R0(i+1) |
| RGB 5:6:5 | Byte 0 | R4(i) | R3(i) | R2(i) | R1(i) | R0(i) | G5(i) | G4(i) | G3(i) |
| | Byte 1 | G2(i) | G1(i) | G0(i) | B4(i) | B3(i) | B2(i) | B1(i) | B0(i) |
| | Byte 2 | R4(i+1) | R3(i+1) | R2(i+1) | R1(i+1) | R0(i+1) | G5(i+1) | G4(i+1) | G3(i+1) |
| | Byte 3 | G2(i+1) | G1(i+1) | G0(i+1) | B4(i+1) | B3(i+1) | B2(i+1) | B1(i+1) | B0(i+1) |

**Table 37-4. RGB Format, RGB_CFG = 10 (Mode 2), No Swap**

| Mode | Byte | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|---|---|
| RGB 5:6:5 | Byte 0 | G2(i) | G1(i) | G0(i) | R4(i) | R3(i) | R2(i) | R1(i) | R0(i) |
| | Byte 1 | B4(i) | B3(i) | B2(i) | B1(i) | B0(i) | G5(i) | G4(i) | G3(i) |
| | Byte 2 | G2(i+1) | G1(i+1) | G0(i+1) | R4(i+1) | R3(i+1) | R2(i+1) | R1(i+1) | R0(i+1) |
| | Byte 3 | B4(i+1) | B3(i+1) | B2(i+1) | B1(i+1) | B0(i+1) | G5(i+1) | G4(i+1) | G3(i+1) |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Table 37-5. RGB Format in Default Mode, RGB_CFG = 00, Swap Activated**

| Mode | Byte | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|---|---|---|---|---|---|---|---|---|---|
| RGB 8:8:8 | Byte 0 | R0(i) | R1(i) | R2(i) | R3(i) | R4(i) | R5(i) | R6(i) | R7(i) |
| | Byte 1 | G0(i) | G1(i) | G2(i) | G3(i) | G4(i) | G5(i) | G6(i) | G7(i) |
| | Byte 2 | B0(i) | B1(i) | B2(i) | B3(i) | B4(i) | B5(i) | B6(i) | B7(i) |
| | Byte 3 | R0(i+1) | R1(i+1) | R2(i+1) | R3(i+1) | R4(i+1) | R5(i+1) | R6(i+1) | R7(i+1) |
| RGB 5:6:5 | Byte 0 | G3(i) | G4(i) | G5(i) | R0(i) | R1(i) | R2(i) | R3(i) | R4(i) |
| | Byte 1 | B0(i) | B1(i) | B2(i) | B3(i) | B4(i) | G0(i) | G1(i) | G2(i) |
| | Byte 2 | G3(i+1) | G4(i+1) | G5(i+1) | R0(i+1) | R1(i+1) | R2(i+1) | R3(i+1) | R4(i+1) |
| | Byte 3 | B0(i+1) | B1(i+1) | B2(i+1) | B3(i+1) | B4(i+1) | G0(i+1) | G1(i+1) | G2(i+1) |

The RGB 5:6:5 input format is processed to be displayed as RGB 5:6:5 format, compliant with the 16-bit mode of the LCD controller.

### 37.5.3 Clocks

The sensor master clock (ISI_MCK) can be generated either by the Advanced Power Management Controller (APMC) through a Programmable Clock output or by an external oscillator connected to the sensor.

None of the sensors embed a power management controller, so providing the clock by the APMC is a simple and efficient way to control power consumption of the system.

Care must be taken when programming the system clock. The ISI has two clock domains, the sensor master clock and the pixel clock provided by sensor. The two clock domains are not synchronized, but the sensor master clock must be faster than the pixel clock.

### 37.5.4 Preview Path

#### 37.5.4.1 Scaling, Decimation (Subsampling)

This module resizes captured 8-bit color sensor images to fit the LCD display format. The resize module performs only downscaling. The same ratio is applied for both horizontal and vertical resize, then a fractional decimation algorithm is applied.

The decimation factor is a multiple of 1/16; values 0 to 15 are forbidden.

**Table 37-6. Decimation Factor**

| Decimation Value | 0–15 | 16 | 17 | 18 | 19 | ... | 124 | 125 | 126 | 127 |
|---|---|---|---|---|---|---|---|---|---|---|
| Decimation Factor | — | 1 | 1.063 | 1.125 | 1.188 | ... | 7.750 | 7.813 | 7.875 | 7.938 |

**Table 37-7. Decimation and Scaler Offset Values**

| OUTPUT | INPUT | 352 × 288 | 640 × 480 | 800 × 600 | 1280 × 1024 | 1600 × 1200 | 2048 × 1536 |
|---|---|---|---|---|---|---|---|
| VGA<br>640 × 480 | F | — | 16 | 20 | 32 | 40 | 51 |
| QVGA<br>320 × 240 | F | 16 | 32 | 40 | 64 | 80 | 102 |
| CIF<br>352 × 288 | F | 16 | 26 | 33 | 56 | 66 | 85 |
| QCIF<br>176 × 144 | F | 32 | 53 | 66 | 113 | 133 | 170 |

© 2021 Microchip Technology Inc.
and its subsidiaries

Example:

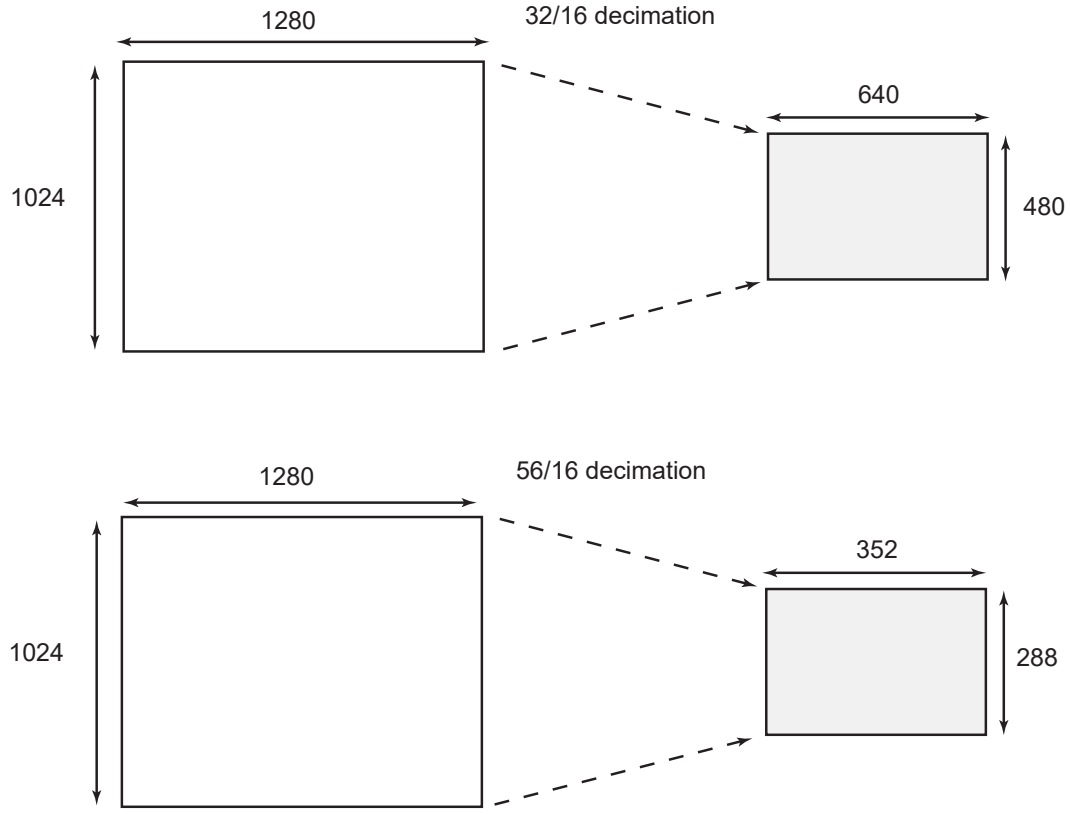
Input 1280 × 1024 Output = 640 × 480

Hratio = 1280/640 = 2

Vratio = 1024/480 = 2.1333

The decimation factor is 2 so 32/16.

**Figure 37-5. Resize Examples**




### 37.5.4.2 Color Space Conversion

This module converts YCrCb or YUV pixels to RGB color space. Clipping is performed to ensure that the samples value do not exceed the allowable range. The conversion matrix is defined below and is fully programmable:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} C_0 & 0 & C_1 \\ C_0 & -C_2 & -C_3 \\ C_0 & C_4 & 0 \end{bmatrix} \times \begin{bmatrix} Y - Y_{\text{off}} \\ C_b - C_{\text{boff}} \\ C_r - C_{\text{roff}} \end{bmatrix}$$

Example of programmable value to convert YCrCb to RGB:

$$\begin{cases} R = 1.164 \cdot (Y - 16) + 1.596 \cdot (C_r - 128) \\ G = 1.164 \cdot (Y - 16) - 0.813 \cdot (C_r - 128) - 0.392 \cdot (C_b - 128) \\ B = 1.164 \cdot (Y - 16) + 2.107 \cdot (C_b - 128) \end{cases}$$

An example of programmable value to convert from YUV to RGB:

$$\begin{cases} R = Y + 1.596 \cdot V \\ G = Y - 0.394 \cdot U - 0.436 \cdot V \\ B = Y + 2.032 \cdot U \end{cases}$$

### 37.5.4.3 Memory Interface

#### 37.5.4.3.1 RGB Mode

The preview datapath contains a data formatter that converts 8:8:8 pixel to RGB 5:6:5 format compliant with the 16-bit format of the LCD controller. In general, when converting from a color channel with more bits to one with fewer bits, the formatter module discards the lower-order bits.

For example, converting from RGB 8:8:8 to RGB 5:6:5, the formatter module discards the three LSBs from the red and blue channels, and two LSBs from the green channel.

#### 37.5.4.3.2 12-bit Grayscale Mode

ISI_DATA[11:0] is the physical interface to the ISI. These bits are sampled and written to memory.

When 12-bit Grayscale mode is enabled, two memory formats are supported:

ISI_CFG2.GS_MODE = 0: two pixels per word

ISI_CFG2.GS_MODE = 1: one pixel per word

The following tables illustrate the memory mapping for the two formats.

**Table 37-8. Grayscale Memory Mapping Configuration for 12-bit Data (ISI_CFG2.GS_MODE = 0: two pixels per word)**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Pixel 0 [11:4] | | | | | | | |
| | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Pixel 0 [3:0] | | | | – | – | – | – |
| | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Pixel 1 [11:4] | | | | | | | |
| | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 1 [3:0] | | | | – | – | – | – |

If ISI_CFG1.GRAYLE = 0, the pixels map as follows:

If ISI_CFG1.GRAYLE=1, the pixels map as follows:

**Table 37-9. Grayscale Memory Mapping Configuration for 12-bit Data (ISI_CFG2.GS_MODE = 0: two pixels per word)**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Pixel 1 [11:4] | | | | | | | |
| | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Pixel 1 [3:0] | | | | – | – | – | – |
| | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Pixel 0 [11:4] | | | | | | | |
| | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 0 [3:0] | | | | – | – | – | – |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Table 37-10. Grayscale Memory Mapping Configuration for 12-bit Data (ISI_CFG2.GS_MODE = 1: one pixel per word)**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Pixel 0 [11:4] | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Pixel 0 [3:0] | | | | – | – | – | – |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| – | – | – | – | – | – | – | – |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| – | – | – | – | – | – | – | – |

#### 37.5.4.3.3 8-bit Grayscale Mode

For 8-bit Grayscale mode, ISI_DATA[7:0] on the 12-bit data bus is the physical interface to the ISI. These bits are sampled and written to memory.

To enable 8-bit Grayscale mode, configure ISI_CFG2 as follows:

- Clear ISI_CFG2.GRAYSCALE.
- Clear ISI_CFG2.RGB_SWAP.
- Clear ISI_CFG2.COL_SPACE.
- Configure the field ISI_CFG2.YCC_SWAP to value 0.
- Configure the field ISI_CFG2.IM_VSIZE with the vertical resolution of the image minus 1.
- Configure the field ISI_CFG2.IM_HSIZE with the horizontal resolution of the image divided by 2. The horizontal resolution must be a multiple of 2.

The codec datapath is used to capture the 8-bit grayscale image. Use the following configuration:

- Set ISI_DMA_C_CTRL.C_FETCH.
- Configure ISI_DMA_C_DSCR.C_DSCR with the descriptor address.
- Write a one to the bit ISI_DMA_CHER.C_CH_EN.

**Table 37-11. Memory Mapping for 8-bit Grayscale Mode**

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| Pixel 3 | | | | | | | |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Pixel 2 | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Pixel 1 | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Pixel 0 | | | | | | | |

### 37.5.4.4 FIFO and DMA Features

Both preview and codec datapaths contain FIFOs. These asynchronous buffers are used to safely transfer formatted pixels from the pixel clock domain to the AHB clock domain. A video arbiter is used to manage FIFO thresholds and triggers a relevant DMA request through the AHB master interface. Thus, depending on the FIFO state, a specified length burst is asserted. Regarding AHB master interface, it supports Scatter DMA mode through linked

list operation. This mode of operation improves flexibility of image buffer location and allows the user to allocate two or more frame buffers. The destination frame buffers are defined by a series of Frame Buffer Descriptors (FBD). Each FBD controls the transfer of one entire frame and then optionally loads a further FBD to switch the DMA operation at another frame buffer address. The FBD is defined by a series of three words. The first word defines the current frame buffer address (named DMA_X_ADDR register), the second defines control information (named DMA_X_CTRL register) and the third defines the next descriptor address (named DMA_X_DSCR). DMA Transfer mode with linked list support is available for both codec and preview datapaths. The data to be transferred described by an FBD requires several burst accesses. In the following example, the use of two ping-pong frame buffers is described.

Example:

The first FBD, stored at address 0x00030000, defines the location of the first frame buffer. This address is programmed in the ISI user interface DMA_P_DSCR. To enable the descriptor fetch operation, the value 0x00000001 must be written to the DMA_P_CTRL register. LLI_0 and LLI_1 are the two descriptors of the linked list.

Destination address: frame buffer ID0 0x02A000 (LLI_0.DMA_P_ADDR)

Transfer 0 Control Information, fetch and writeback: 0x00000003 (LLI_0.DMA_P_CTRL)

Next FBD address: 0x00030010 (LLI_0.DMA_P_DSCR)

The second FBD, stored at address 0x00030010, defines the location of the second frame buffer.

Destination address: frame buffer ID1 0x0003A000 (LLI_1.DMA_P_ADDR)

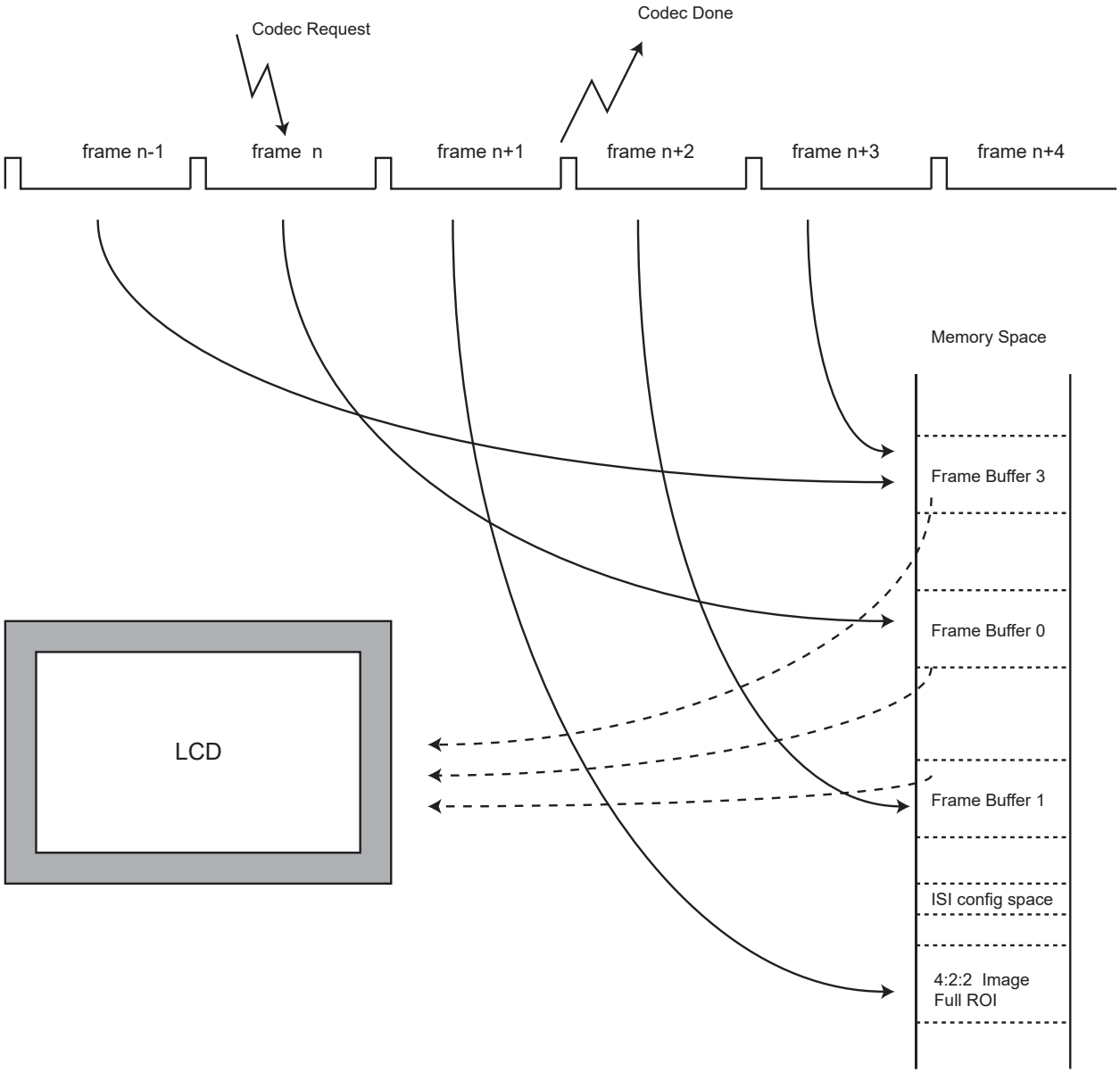
Transfer 1 Control information fetch and writeback: 0x00000003 (LLI_1.DMA_P_CTRL)

The third FBD address: 0x00030000, wrapping to first FBD (LLI_1.DMA_P_DSCR)

Using this technique, several frame buffers can be configured through the linked list. The following figure illustrates a typical three-frame buffer application. Frame n is mapped to frame buffer 0, frame n+1 is mapped to frame buffer 1, frame n+2 is mapped to frame buffer 2 and further frames wrap. A codec request occurs, and the full-size 4:2:2 encoded frame is stored in a dedicated memory space.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 37-6. Three Frame Buffers Application and Memory Mapping**




### 37.5.5 Codec Path

#### 37.5.5.1 Color Space Conversion

Depending on user selection, this module can be bypassed so that input YCrCb stream is directly connected to the format converter module. If the RGB input stream is selected, this module converts RGB to YCrCb color space with the formulas given below:

$$\begin{bmatrix} Y \\ C_r \\ C_b \end{bmatrix} = \begin{bmatrix} C_0 & C_1 & C_2 \\ C_3 & -C_4 & -C_5 \\ -C_6 & -C_7 & C_8 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} Y_{\text{off}} \\ \text{Cr}_{\text{off}} \\ \text{Cb}_{\text{off}} \end{bmatrix}$$

An example of coefficients is given below:

$$\begin{cases} Y = 0.257 \cdot R + 0.504 \cdot G + 0.098 \cdot B + 16 \\ C_r = 0.439 \cdot R - 0.368 \cdot G - 0.071 \cdot B + 128 \\ C_b = -0.148 \cdot R - 0.291 \cdot G + 0.439 \cdot B + 128 \end{cases}$$

#### 37.5.5.2 Memory Interface

Dedicated FIFOs are used to support packed memory mapping. YCrCb pixel components are sent in a single 32-bit word in a contiguous space (packed). Data is stored in the order of natural scan lines. Planar mode is not supported.

#### 37.5.5.3 DMA Features

Like preview datapath, codec datapath DMA mode uses linked list operation.

© 2021 Microchip Technology Inc.
and its subsidiaries

## 37.6 Register Summary

**Note:** Several parts of the ISI controller use the pixel clock provided by the image sensor (ISI_PCK). Thus the user must first program the image sensor to provide this clock (ISI_PCK) before programming the Image Sensor Controller.

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ISI_CFG1 | 7:0 | CRC_SYNC | EMB_SYNC | GRAYLE | PIXCLK_POL | VSYNC_POL | HSYNC_POL | | |
| | | 15:8 | | THMASK[1:0] | | FULL | DISCR | FRATE[2:0] | | |
| | | 23:16 | SLD[7:0] | | | | | | | |
| | | 31:24 | SFD[7:0] | | | | | | | |
| 0x04 | ISI_CFG2 | 7:0 | IM_VSIZE[7:0] | | | | | | | |
| | | 15:8 | COL_SPACE | RGB_SWAP | GRAYSCALE | RGB_MODE | GS_MODE | IM_VSIZE[10:8] | | |
| | | 23:16 | IM_HSIZE[7:0] | | | | | | | |
| | | 31:24 | RGB_CFG[1:0] | | YCC_SWAP[1:0] | | | IM_HSIZE[10:8] | | |
| 0x08 | ISI_PSIZE | 7:0 | PREV_VSIZE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | PREV_VSIZE[9:8] | |
| | | 23:16 | PREV_HSIZE[7:0] | | | | | | | |
| | | 31:24 | | | | | | | PREV_HSIZE[9:8] | |
| 0x0C | ISI_PDECF | 7:0 | DEC_FACTOR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | ISI_Y2R_SET0 | 7:0 | C0[7:0] | | | | | | | |
| | | 15:8 | C1[7:0] | | | | | | | |
| | | 23:16 | C2[7:0] | | | | | | | |
| | | 31:24 | C3[7:0] | | | | | | | |
| 0x14 | ISI_Y2R_SET1 | 7:0 | C4[7:0] | | | | | | | |
| | | 15:8 | | Cboff | Croff | Yoff | | | | C4[8] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | ISI_R2Y_SET0 | 7:0 | | C0[6:0] | | | | | | |
| | | 15:8 | | C1[6:0] | | | | | | |
| | | 23:16 | | C2[6:0] | | | | | | |
| | | 31:24 | | | | | | | | Roff |
| 0x1C | ISI_R2Y_SET1 | 7:0 | | C3[6:0] | | | | | | |
| | | 15:8 | | C4[6:0] | | | | | | |
| | | 23:16 | | C5[6:0] | | | | | | |
| | | 31:24 | | | | | | | | Goff |
| 0x20 | ISI_R2Y_SET2 | 7:0 | | C6[6:0] | | | | | | |
| | | 15:8 | | C7[6:0] | | | | | | |
| | | 23:16 | | C8[6:0] | | | | | | |
| | | 31:24 | | | | | | | | Boff |
| 0x24 | ISI_CR | 7:0 | | | | | | ISI_SRST | ISI_DIS | ISI_EN |
| | | 15:8 | | | | | | | | ISI_CDC |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | ISI_SR | 7:0 | | | | | | SRST | DIS_DONE | ENABLE |
| | | 15:8 | | | | | | VSYNC | | CDC_PND |
| | | 23:16 | | | | | SIP | | CXFR_DONE | PXFR_DONE |
| | | 31:24 | | | | | FR_OVR | CRC_ERR | C_OVR | P_OVR |
| 0x2C | ISI_IER | 7:0 | | | | | | SRST | DIS_DONE | |
| | | 15:8 | | | | | | VSYNC | | |
| | | 23:16 | | | | | | | CXFR_DONE | PXFR_DONE |
| | | 31:24 | | | | | FR_OVR | CRC_ERR | C_OVR | P_OVR |
| 0x30 | ISI_IDR | 7:0 | | | | | | SRST | DIS_DONE | |
| | | 15:8 | | | | | | VSYNC | | |
| | | 23:16 | | | | | | | CXFR_DONE | PXFR_DONE |
| | | 31:24 | | | | | FR_OVR | CRC_ERR | C_OVR | P_OVR |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x34 | ISI_IMR | 7:0 | | | | | | SRST | DIS_DONE | |
| | | 15:8 | | | | | | VSYNC | | |
| | | 23:16 | | | | | | | CXFR_DONE | PXFR_DONE |
| | | 31:24 | | | | | FR_OVR | CRC_ERR | C_OVR | P_OVR |
| 0x38 | ISI_DMA_CHER | 7:0 | | | | | | | C_CH_EN | P_CH_EN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | ISI_DMA_CHDR | 7:0 | | | | | | | C_CH_DIS | P_CH_DIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | ISI_DMA_CHSR | 7:0 | | | | | | | C_CH_S | P_CH_S |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | ISI_DMA_P_ADDR | 7:0 | P_ADDR[5:0] | | | | | | | |
| | | 15:8 | P_ADDR[13:6] | | | | | | | |
| | | 23:16 | P_ADDR[21:14] | | | | | | | |
| | | 31:24 | P_ADDR[29:22] | | | | | | | |
| 0x48 | ISI_DMA_P_CTRL | 7:0 | | | | | P_DONE | P_IEN | P_WB | P_FETCH |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | ISI_DMA_P_DSCR | 7:0 | P_DSCR[5:0] | | | | | | | |
| | | 15:8 | P_DSCR[13:6] | | | | | | | |
| | | 23:16 | P_DSCR[21:14] | | | | | | | |
| | | 31:24 | P_DSCR[29:22] | | | | | | | |
| 0x50 | ISI_DMA_C_ADDR | 7:0 | C_ADDR[5:0] | | | | | | | |
| | | 15:8 | C_ADDR[13:6] | | | | | | | |
| | | 23:16 | C_ADDR[21:14] | | | | | | | |
| | | 31:24 | C_ADDR[29:22] | | | | | | | |
| 0x54 | ISI_DMA_C_CTRL | 7:0 | | | | | C_DONE | C_IEN | C_WB | C_FETCH |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x58 | ISI_DMA_C_DSCR | 7:0 | C_DSCR[5:0] | | | | | | | |
| | | 15:8 | C_DSCR[13:6] | | | | | | | |
| | | 23:16 | C_DSCR[21:14] | | | | | | | |
| | | 31:24 | C_DSCR[29:22] | | | | | | | |
| 0x5C ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | ISI_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | ISI_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 37.6.1 ISI Configuration 1 Register

**Name:** ISI_CFG1
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SFD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SLD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | THMASK[1:0] | | FULL | DISCR | FRATE[2:0] | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CRC_SYNC | EMB_SYNC | GRAYLE | PIXCLK_POL | VSYNC_POL | HSYNC_POL | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:24 – SFD[7:0]** Start of Frame Delay
SFD lines are skipped at the beginning of the frame.

**Bits 23:16 – SLD[7:0]** Start of Line Delay
SLD pixel clock periods to wait before the beginning of a line.

**Bits 14:13 – THMASK[1:0]** Threshold Mask

| Value | Name | Description |
|---|---|---|
| 0 | BEATS_4 | Only 4 beats AHB burst allowed |
| 1 | BEATS_8 | Only 4 and 8 beats AHB burst allowed |
| 2 | BEATS_16 | 4, 8 and 16 beats AHB burst allowed |

**Bit 12 – FULL** Full Mode is Allowed

| Value | Description |
|---|---|
| 0 | The codec frame is transferred to memory when an available frame slot is detected. |
| 1 | Both preview and codec DMA channels are operating simultaneously. |

**Bit 11 – DISCR** Disable Codec Request

| Value | Description |
|---|---|
| 0 | Codec datapath DMA interface requires a request to restart. |
| 1 | Codec datapath DMA automatically restarts. |

**Bits 10:8 – FRATE[2:0]** Frame Rate [0..7]

| Value | Description |
|---|---|
| 0 | All the frames are captured, else one frame every FRATE + 1 is captured. |

**Bit 7 – CRC_SYNC** Embedded Synchronization Correction

| Value | Description |
|---|---|
| 0 | No CRC correction is performed on embedded synchronization. |
| 1 | CRC correction is performed. If the correction is not possible, the current frame is discarded and the CRC_ERR bit is set in the ISI_SR. |

**Bit 6 – EMB_SYNC** Embedded Synchronization

| Value | Description |
|---|---|
| 0 | Synchronization by HSYNC, VSYNC. |
| 1 | Synchronization by embedded synchronization sequence SAV/EAV. |

**Bit 5 – GRAYLE** Grayscale Little Endian
Refer to Table 37-8 and Table 37-9 for details.

| Value | Description |
|---|---|
| 0 | The two pixels are represented in big-endian format within a 32-bit register. |
| 1 | The two pixels are represented in little-endian format within a 32-bit register. |

**Bit 4 – PIXCLK_POL** Pixel Clock Polarity

| Value | Description |
|---|---|
| 0 | Data is sampled on rising edge of pixel clock. |
| 1 | Data is sampled on falling edge of pixel clock. |

**Bit 3 – VSYNC_POL** Vertical Synchronization Polarity

| Value | Description |
|---|---|
| 0 | VSYNC active high. |
| 1 | VSYNC active low. |

**Bit 2 – HSYNC_POL** Horizontal Synchronization Polarity

| Value | Description |
|---|---|
| 0 | HSYNC active high. |
| 1 | HSYNC active low. |

### 37.6.2 ISI Configuration 2 Register

**Name:** ISI_CFG2
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RGB_CFG[1:0] | | YCC_SWAP[1:0] | | | IM_HSIZE[10:8] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IM_HSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COL_SPACE | RGB_SWAP | GRAYSCALE | RGB_MODE | GS_MODE | IM_VSIZE[10:8] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IM_VSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:30 – RGB_CFG[1:0]** RGB Pixel Mapping Configuration
Defines RGB pattern when RGB_MODE is set to 1.
If RGB_MODE is set to RGB 8:8:8, then RGB_CFG = 0 implies RGB color sequence, else it implies BGR color sequence.

| Value | Name | Description |
|---|---|---|
| 0 | DEFAULT | Byte 0 R/G(MSB)<br>Byte 1 G(LSB)/B<br>Byte 2 R/G(MSB)<br>Byte 3 G(LSB)/B |
| 1 | MODE1 | Byte 0 B/G(MSB)<br>Byte 1 G(LSB)/R<br>Byte 2 B/G(MSB)<br>Byte 3 G(LSB)/R |
| 2 | MODE2 | Byte 0 G(LSB)/R<br>Byte 1 B/G(MSB)<br>Byte 2 G(LSB)/R<br>Byte 3 B/G(MSB) |
| 3 | MODE3 | Byte 0 G(LSB)/B<br>Byte 1 R/G(MSB)<br>Byte 2 G(LSB)/B<br>Byte 3 R/G(MSB) |

**Bits 29:28 – YCC_SWAP[1:0]** YCrCb Format Swap Mode
Defines the YCC image data.

| Value | Name | Description |
|---|---|---|
| 0 | DEFAULT | Byte 0 Cb(i)<br>Byte 1 Y(i)<br>Byte 2 Cr(i)<br>Byte 3 Y(i+1) |
| 1 | MODE1 | Byte 0 Cr(i)<br>Byte 1 Y(i)<br>Byte 2 Cb(i)<br>Byte 3 Y(i+1) |
| 2 | MODE2 | Byte 0 Y(i)<br>Byte 1 Cb(i)<br>Byte 2 Y(i+1)<br>Byte 3 Cr(i) |
| 3 | MODE3 | Byte 0 Y(i)<br>Byte 1 Cr(i)<br>Byte 2 Y(i+1)<br>Byte 3 Cb(i) |

**Bits 26:16 – IM_HSIZE[10:0]** Horizontal Size of the Image Sensor [0..2047]
If 8-bit Grayscale mode is enabled, IM_HSIZE = (Horizontal size/2) - 1.
Else IM_HSIZE = Horizontal size - 1.

**Bit 15 – COL_SPACE** Color Space for the Image Data

| Value | Description |
|---|---|
| 0 | YCbCr. |
| 1 | RGB. |

**Bit 14 – RGB_SWAP** RGB Format Swap Mode
The RGB_SWAP has no effect when Grayscale mode is enabled.

| Value | Description |
|---|---|
| 0 | D7 → R7. |
| 1 | D0 → R7. |

**Bit 13 – GRAYSCALE** Grayscale Mode Format Enable

| Value | Description |
|---|---|
| 0 | Grayscale mode is disabled. |
| 1 | Input image is assumed to be grayscale-coded. |

**Bit 12 – RGB_MODE** RGB Input Mode

| Value | Description |
|---|---|
| 0 | RGB 8:8:8 24 bits. |
| 1 | RGB 5:6:5 16 bits. |

**Bit 11 – GS_MODE** Grayscale Pixel Format Mode

| Value | Description |
|---|---|
| 0 | 2 pixels per word. |
| 1 | 1 pixel per word. |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bits 10:0 – IM_VSIZE[10:0]** Vertical Size of the Image Sensor [0..2047]
IM_VSIZE = Vertical size - 1

### 37.6.3 ISI Preview Size Register

**Name:** ISI_PSIZE
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PREV_HSIZE[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PREV_HSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PREV_VSIZE[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PREV_VSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25:16 – PREV_HSIZE[9:0]** Horizontal Size for the Preview Path
PREV_HSIZE = Horizontal Preview size - 1 (640 max only in RGB mode).

**Bits 9:0 – PREV_VSIZE[9:0]** Vertical Size for the Preview Path
PREV_VSIZE = Vertical Preview size - 1 (480 max only in RGB mode).

### 37.6.4 ISI Preview Decimation Factor Register

**Name:** ISI_PDECF
**Offset:** 0x0C
**Reset:** 0x00000010
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DEC_FACTOR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

**Bits 7:0 – DEC_FACTOR[7:0]** Decimation Factor
DEC_FACTOR is 8-bit width, range is from 16 to 255. Values from 0 to 16 do not perform any decimation.

### 37.6.5 ISI Color Space Conversion YCrCb to RGB Set 0 Register

**Name:** ISI_Y2R_SET0
**Offset:** 0x10
**Reset:** 0x6832CC95
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | C3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | C2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | C1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |

**Bits 31:24 – C3[7:0]** Color Space Conversion Matrix Coefficient C3
C3 element default step is 1/128, ranges from 0 to 1.9921875.

**Bits 23:16 – C2[7:0]** Color Space Conversion Matrix Coefficient C2
C2 element default step is 1/128, ranges from 0 to 1.9921875.

**Bits 15:8 – C1[7:0]** Color Space Conversion Matrix Coefficient C1
C1 element default step is 1/128, ranges from 0 to 1.9921875.

**Bits 7:0 – C0[7:0]** Color Space Conversion Matrix Coefficient C0
C0 element default step is 1/128, ranges from 0 to 1.9921875.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 37.6.6 ISI Color Space Conversion YCrCb to RGB Set 1 Register

**Name:** ISI_Y2R_SET1
**Offset:** 0x14
**Reset:** 0x00007102
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | Cboff | Croff | Yoff | | | | C4[8] |
| Access | | R/W | R/W | R/W | | | | R/W |
| Reset | | 1 | 1 | 1 | | | | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C4[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**Bit 14 – Cboff** Color Space Conversion Blue Chrominance Default Offset

| Value | Description |
|---|---|
| 0 | No offset. |
| 1 | Offset = 16. |

**Bit 13 – Croff** Color Space Conversion Red Chrominance Default Offset

| Value | Description |
|---|---|
| 0 | No offset. |
| 1 | Offset = 16. |

**Bit 12 – Yoff** Color Space Conversion Luminance Default Offset

| Value | Description |
|---|---|
| 0 | No offset. |
| 1 | Offset = 128. |

**Bits 8:0 – C4[8:0]** Color Space Conversion Matrix Coefficient C4
C4 element default step is 1/128, ranges from 0 to 3.9921875.

### 37.6.7 ISI Color Space Conversion RGB to YCrCb Set 0 Register

**Name:** ISI_R2Y_SET0
**Offset:** 0x18
**Reset:** 0x01324145
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Roff |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | C2[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 1 | 1 | 0 | 0 | 1 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | C1[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | C0[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 0 | 0 | 0 | 1 | 0 | 1 |

**Bit 24 – Roff** Color Space Conversion Red Component Offset

| Value | Description |
|---|---|
| 0 | No offset |
| 1 | Offset = 16 |

**Bits 22:16 – C2[6:0]** Color Space Conversion Matrix Coefficient C2
C2 element default step is 1/512, from 0 to 0.2480468875.

**Bits 14:8 – C1[6:0]** Color Space Conversion Matrix Coefficient C1
C1 element default step is 1/128, from 0 to 0.9921875.

**Bits 6:0 – C0[6:0]** Color Space Conversion Matrix Coefficient C0
C0 element default step is 1/256, from 0 to 0.49609375.

### 37.6.8 ISI Color Space Conversion RGB to YCrCb Set 1 Register

**Name:** ISI_R2Y_SET1
**Offset:** 0x1C
**Reset:** 0x01245E38
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Goff |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | C5[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | C4[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 0 | 1 | 1 | 1 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | C3[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 1 | 1 | 1 | 0 | 0 | 0 |

**Bit 24 – Goff** Color Space Conversion Green Component Offset

| Value | Description |
|---|---|
| 0 | No offset. |
| 1 | Offset = 128. |

**Bits 22:16 – C5[6:0]** Color Space Conversion Matrix Coefficient C5
C1 element default step is 1/512, ranges from 0 to 0.2480468875.

**Bits 14:8 – C4[6:0]** Color Space Conversion Matrix Coefficient C4
C1 element default step is 1/256, ranges from 0 to 0.49609375.

**Bits 6:0 – C3[6:0]** Color Space Conversion Matrix Coefficient C3
C0 element default step is 1/128, ranges from 0 to 0.9921875.

### 37.6.9 ISI Color Space Conversion RGB to YCrCb Set 2 Register

**Name:** ISI_R2Y_SET2
**Offset:** 0x20
**Reset:** 0x01384A4B
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | Boff |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | C8[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 1 | 1 | 1 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | C7[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 0 | 0 | 1 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | C6[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 1 | 0 | 0 | 1 | 0 | 1 | 1 |

**Bit 24 – Boff** Color Space Conversion Blue Component Offset

| Value | Description |
|---|---|
| 0 | No offset. |
| 1 | Offset = 128. |

**Bits 22:16 – C8[6:0]** Color Space Conversion Matrix Coefficient C8
C8 element default step is 1/128, ranges from 0 to 0.9921875.

**Bits 14:8 – C7[6:0]** Color Space Conversion Matrix Coefficient C7
C7 element default step is 1/256, ranges from 0 to 0.49609375.

**Bits 6:0 – C6[6:0]** Color Space Conversion Matrix Coefficient C6
C6 element default step is 1/512, ranges from 0 to 0.2480468875.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 37.6.10 ISI Control Register

**Name:** ISI_CR
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | ISI_CDC |
| Reset | | | | | | | | W |
| | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | ISI_SRST | ISI_DIS | ISI_EN |
| Reset | | | | | | W | W | W |
| | | | | | | – | – | – |

**Bit 8 – ISI_CDC** ISI Codec Request
Write a one to this bit to enable the codec datapath and capture a full resolution frame. A new request cannot be taken into account while CDC_PND bit is active in the ISI_SR.

**Bit 2 – ISI_SRST** ISI Software Reset Request
Write a one to this bit to request a software reset of the module. Software must poll the SRST bit in the ISI_SR to verify that the software request command has terminated.

**Bit 1 – ISI_DIS** ISI Module Disable Request
Write a one to this bit to disable the module. If both ISI_EN and ISI_DIS are asserted at the same time, the disable request is not taken into account. Software must poll the DIS_DONE bit in the ISI_SR to verify that the command has successfully completed.

**Bit 0 – ISI_EN** ISI Module Enable Request
Write a one to this bit to enable the module. Software must poll the ENABLE bit in the ISI_SR to verify that the command has successfully completed.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 37.6.11 ISI Status Register

**Name:** ISI_SR
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FR_OVR | CRC_ERR | C_OVR | P_OVR |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SIP | | CXFR_DONE | PXFR_DONE |
| Access | | | | | R | | R | R |
| Reset | | | | | 0 | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | VSYNC | | CDC_PND |
| Access | | | | | | R | | R |
| Reset | | | | | | 0 | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SRST | DIS_DONE | ENABLE |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 27 – FR_OVR** Frame Rate Overrun (cleared on read)

| Value | Description |
|---|---|
| 0 | No frame overrun |
| 1 | Frame overrun. The current frame is being skipped because a vsync signal has been detected while flushing FIFOs since the last read of ISI_SR. |

**Bit 26 – CRC_ERR** CRC Synchronization Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No CRC error in the embedded synchronization frame (SAV/EAV) |
| 1 | Embedded Synchronization Correction is enabled (CRC_SYNC bit is set) in the ISI_CR and an error has been detected and not corrected since the last read of ISI_SR. The frame is discarded and the ISI waits for a new one. |

**Bit 25 – C_OVR** Codec Datapath Overflow (cleared on read)

| Value | Description |
|---|---|
| 0 | No overflow |
| 1 | An overrun condition has occurred in input FIFO on the codec path. The overrun happens when the FIFO is full and an attempt is made to write a new sample to the FIFO since the last read of ISI_SR. |

**Bit 24 – P_OVR** Preview Datapath Overflow (cleared on read)

| Value | Description |
|---|---|
| 0 | No overflow |
| 1 | An overrun condition has occurred in input FIFO on the preview path. The overrun happens when the FIFO is full and an attempt is made to write a new sample to the FIFO since the last read of ISI_SR. |

**Bit 19 – SIP** Synchronization in Progress
When the status of the preview or codec DMA channel is modified, a minimum amount of time is required to perform the clock domain synchronization.

| Value | Description |
|---|---|
| 0 | The clock domain synchronization process is terminated. |
| 1 | This bit is set when the clock domain synchronization operation occurs. No modification of the channel status is allowed when this bit is set, to guarantee data integrity. |

**Bit 17 – CXFR_DONE** Codec DMA Transfer has Terminated (cleared on read)

| Value | Description |
|---|---|
| 0 | Codec transfer done not detected. |
| 1 | Codec transfer done detected. When set, this bit indicates that the data transfer on the codec channel has completed since the last read of ISI_SR. |

**Bit 16 – PXFR_DONE** Preview DMA Transfer has Terminated (cleared on read)

| Value | Description |
|---|---|
| 0 | Preview transfer done not detected. |
| 1 | Preview transfer done detected. When set, this bit indicates that the data transfer on the preview channel has completed since the last read of ISI_SR. |

**Bit 10 – VSYNC** Vertical Synchronization (cleared on read)

| Value | Description |
|---|---|
| 0 | Indicates that the vertical synchronization has not been detected since the last read of the ISI_SR. |
| 1 | Indicates that a vertical synchronization has been detected since the last read of the ISI_SR. |

**Bit 8 – CDC_PND** Pending Codec Request

| Value | Description |
|---|---|
| 0 | Indicates that no codec request is pending |
| 1 | Indicates that the request has been taken into account but cannot be serviced within the current frame. The operation is postponed to the next frame. |

**Bit 2 – SRST** Module Software Reset Request has Terminated (cleared on read)

| Value | Description |
|---|---|
| 0 | Indicates that the request is not completed (if a request was issued). |
| 1 | Software reset request has completed. This flag is reset after a read operation. |

**Bit 1 – DIS_DONE** Module Disable Request has Terminated (cleared on read)

| Value | Description |
|---|---|
| 0 | Indicates that the request is not completed (if a request was issued). |
| 1 | Disable request has completed. This flag is reset after a read operation. |

**Bit 0 – ENABLE** Module Enable

| Value | Description |
|---|---|
| 0 | Module is disabled. |
| 1 | Module is enabled. |

### 37.6.12 ISI Interrupt Enable Register

**Name:** ISI_IER
**Offset:** 0x2C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FR_OVR | CRC_ERR | C_OVR | P_OVR |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CXFR_DONE | PXFR_DONE |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | VSYNC | | |
| Access | | | | | | W | | |
| Reset | | | | | | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SRST | DIS_DONE | |
| Access | | | | | | W | W | |
| Reset | | | | | | – | – | |

**Bit 27 – FR_OVR** Frame Rate Overflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

**Bit 26 – CRC_ERR** Embedded Synchronization CRC Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

**Bit 25 – C_OVR** Codec Datapath Overflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

**Bit 24 – P_OVR** Preview Datapath Overflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

**Bit 17 – CXFR_DONE** Codec DMA Transfer Done Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

**Bit 16 – PXFR_DONE** Preview DMA Transfer Done Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Enables the corresponding interrupt. |

**Bit 10 – VSYNC** Vertical Synchronization Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

**Bit 2 – SRST** Software Reset Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

**Bit 1 – DIS_DONE** Disable Done Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 37.6.13 ISI Interrupt Disable Register

**Name:** ISI_IDR
**Offset:** 0x30
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FR_OVR | CRC_ERR | C_OVR | P_OVR |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CXFR_DONE | PXFR_DONE |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | VSYNC | | |
| Access | | | | | | W | | |
| Reset | | | | | | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SRST | DIS_DONE | |
| Access | | | | | | W | W | |
| Reset | | | | | | – | – | |

**Bit 27 – FR_OVR** Frame Rate Overflow Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

**Bit 26 – CRC_ERR** Embedded Synchronization CRC Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

**Bit 25 – C_OVR** Codec Datapath Overflow Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

**Bit 24 – P_OVR** Preview Datapath Overflow Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

**Bit 17 – CXFR_DONE** Codec DMA Transfer Done Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

**Bit 16 – PXFR_DONE** Preview DMA Transfer Done Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Disables the corresponding interrupt. |

**Bit 10 – VSYNC** Vertical Synchronization Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

**Bit 2 – SRST** Software Reset Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

**Bit 1 – DIS_DONE** Disable Done Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 37.6.14 ISI Interrupt Mask Register

**Name:** ISI_IMR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FR_OVR | CRC_ERR | C_OVR | P_OVR |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CXFR_DONE | PXFR_DONE |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | VSYNC | | |
| Access | | | | | | R | | |
| Reset | | | | | | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SRST | DIS_DONE | |
| Access | | | | | | R | R | |
| Reset | | | | | | 0 | 0 | |

**Bit 27 – FR_OVR** Frame Rate Overrun

| Value | Description |
|---|---|
| 0 | The Frame Rate Overrun interrupt is disabled. |
| 1 | The Frame Rate Overrun is enabled. |

**Bit 26 – CRC_ERR** CRC Synchronization Error

| Value | Description |
|---|---|
| 0 | The CRC Synchronization Error interrupt is disabled. |
| 1 | The CRC Synchronization Error interrupt is enabled. |

**Bit 25 – C_OVR** Codec FIFO Overflow

| Value | Description |
|---|---|
| 0 | The Codec FIFO Overflow interrupt is disabled. |
| 1 | The Codec FIFO Overflow interrupt is enabled. |

**Bit 24 – P_OVR** Preview FIFO Overflow

| Value | Description |
|---|---|
| 0 | The Preview FIFO Overflow interrupt is disabled. |
| 1 | The Preview FIFO Overflow interrupt is enabled. |

**Bit 17 – CXFR_DONE** Codec DMA Transfer Completed

| Value | Description |
|---|---|
| 0 | The Codec DMA Transfer Completed interrupt is disabled. |
| 1 | The Codec DMA Transfer Completed interrupt is enabled. |

**Bit 16 – PXFR_DONE** Preview DMA Transfer Completed

| Value | Description |
|---|---|
| 0 | The Preview DMA Transfer Completed interrupt is disabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | The Preview DMA Transfer Completed interrupt is enabled. |

**Bit 10 – VSYNC** Vertical Synchronization

| Value | Description |
|---|---|
| 0 | The Vertical Synchronization interrupt is disabled. |
| 1 | The Vertical Synchronization interrupt is enabled. |

**Bit 2 – SRST** Software Reset Completed

| Value | Description |
|---|---|
| 0 | The Software Reset Completed interrupt is disabled. |
| 1 | The Software Reset Completed interrupt is enabled. |

**Bit 1 – DIS_DONE** Module Disable Operation Completed

| Value | Description |
|---|---|
| 0 | The Module Disable Operation Completed interrupt is disabled. |
| 1 | The Module Disable Operation Completed interrupt is enabled. |

### 37.6.15 DMA Channel Enable Register

**Name:** ISI_DMA_CHER
**Offset:** 0x38
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | C_CH_EN | P_CH_EN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 1 – C_CH_EN** Codec Channel Enable
Write a one to this bit to enable the codec DMA channel.

**Bit 0 – P_CH_EN** Preview Channel Enable
Write a one to this bit to enable the preview DMA channel.

### 37.6.16 DMA Channel Disable Register

**Name:** ISI_DMA_CHDR
**Offset:** 0x3C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | C_CH_DIS | P_CH_DIS |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 1 – C_CH_DIS** Codec Channel Disable Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the channel. Poll C_CH_S in DMA_CHSR to verify that the codec channel status has been successfully modified. |

**Bit 0 – P_CH_DIS** Preview Channel Disable Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the channel. Poll P_CH_S in DMA_CHSR to verify that the preview channel status has been successfully modified. |

### 37.6.17 DMA Channel Status Register

**Name:** ISI_DMA_CHSR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | C_CH_S | P_CH_S |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – C_CH_S** Code DMA Channel Status

| Value | Description |
|---|---|
| 0 | Indicates that the Codec DMA channel is disabled. |
| 1 | Indicates that the Codec DMA channel is enabled. |

**Bit 0 – P_CH_S** Preview DMA Channel Status

| Value | Description |
|---|---|
| 0 | Indicates that the Preview DMA channel is disabled. |
| 1 | Indicates that the Preview DMA channel is enabled. |

### 37.6.18 DMA Preview Base Address Register

**Name:** ISI_DMA_P_ADDR
**Offset:** 0x44
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P_ADDR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P_ADDR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P_ADDR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P_ADDR[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – P_ADDR[29:0]** Preview Image Base Address
This address is word-aligned.

### 37.6.19 DMA Preview Control Register

**Name:** ISI_DMA_P_CTRL
**Offset:** 0x48
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | P_DONE | P_IEN | P_WB | P_FETCH |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – P_DONE** Preview Transfer Done
This bit is only updated in the memory.

| Value | Description |
|---|---|
| 0 | The transfer related to this descriptor has not been performed. |
| 1 | The transfer related to this descriptor has completed. This bit is updated in memory at the end of the transfer, when writeback operation is enabled. |

**Bit 2 – P_IEN** Transfer Done Flag Control

| Value | Description |
|---|---|
| 0 | Preview transfer done flag generation is enabled. |
| 1 | Preview transfer done flag generation is disabled. |

**Bit 1 – P_WB** Descriptor Writeback Control Bit

| Value | Description |
|---|---|
| 0 | Preview channel writeback operation is disabled. |
| 1 | Preview channel writeback operation is enabled. |

**Bit 0 – P_FETCH** Descriptor Fetch Control Bit

| Value | Description |
|---|---|
| 0 | Preview channel fetch operation is disabled. |
| 1 | Preview channel fetch operation is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 37.6.20 DMA Preview Descriptor Address Register

**Name:** ISI_DMA_P_DSCR
**Offset:** 0x4C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | P_DSCR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | P_DSCR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | P_DSCR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | P_DSCR[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – P_DSCR[29:0]** Preview Descriptor Base Address
This address is word-aligned.

### 37.6.21 DMA Codec Base Address Register

**Name:** ISI_DMA_C_ADDR
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | C_ADDR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | C_ADDR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | C_ADDR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C_ADDR[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – C_ADDR[29:0]** Codec Image Base Address
This address is word-aligned.

### 37.6.22 DMA Codec Control Register

**Name:** ISI_DMA_C_CTRL
**Offset:** 0x54
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | C_DONE | C_IEN | C_WB | C_FETCH |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – C_DONE** Codec Transfer Done
This bit is only updated in the memory.

| Value | Description |
|---|---|
| 0 | The transfer related to this descriptor has not been performed. |
| 1 | The transfer related to this descriptor has completed. This bit is updated in memory at the end of the transfer when writeback operation is enabled. |

**Bit 2 – C_IEN** Transfer Done Flag Control

| Value | Description |
|---|---|
| 0 | Codec transfer done flag generation is enabled. |
| 1 | Codec transfer done flag generation is disabled. |

**Bit 1 – C_WB** Descriptor Writeback Control Bit

| Value | Description |
|---|---|
| 0 | Codec channel writeback operation is disabled. |
| 1 | Codec channel writeback operation is enabled. |

**Bit 0 – C_FETCH** Descriptor Fetch Control Bit

| Value | Description |
|---|---|
| 0 | Codec channel fetch operation is disabled. |
| 1 | Codec channel fetch operation is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 37.6.23 DMA Codec Descriptor Address Register

**Name:** ISI_DMA_C_DSCR
**Offset:** 0x58
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | C_DSCR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | C_DSCR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | C_DSCR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | C_DSCR[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – C_DSCR[29:0]** Codec Descriptor Base Address
This address is word-aligned.

### 37.6.24 ISI Write Protection Mode Register

**Name:** ISI_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key Password

| Value | Name | Description |
|---|---|---|
| 0x495349 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x495349 (“ISI” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x495349 (“ISI” in ASCII). |

### 37.6.25 ISI Write Protection Status Register

**Name:** ISI_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source

| Value | Name |
|---|---|
| 0 | No Write Protection Violation occurred since the last read of this register (ISI_WPSR). |
| 1 | Write access in ISI_CFG1 while Write Protection was enabled (since the last read). |
| 2 | Write access in ISI_CFG2 while Write Protection was enabled (since the last read). |
| 3 | Write access in ISI_PSIZE while Write Protection was enabled (since the last read). |
| 4 | Write access in ISI_PDECF while Write Protection was enabled (since the last read). |
| 5 | Write access in ISI_Y2R_SET0 while Write Protection was enabled (since the last read). |
| 6 | Write access in ISI_Y2R_SET1 while Write Protection was enabled (since the last read). |
| 7 | Write access in ISI_R2Y_SET0 while Write Protection was enabled (since the last read). |
| 8 | Write access in ISI_R2Y_SET1 while Write Protection was enabled (since the last read). |
| 9 | Write access in ISI_R2Y_SET2 while Write Protection was enabled (since the last read). |

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Name |
|---|---|
| 0 | No write protection violation occurred since the last read of ISI_WPSR. |
| 1 | A write protection violation has occurred since the last read of the ISI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 38. GMAC - Ethernet MAC

The description and registers of this peripheral are using the 'GMAC' designation although the device does not support Gigabit Ethernet functionality.

## 38.1 Description

The Ethernet Media Access Controller (GMAC) module implements a 10/100 Mbps Ethernet MAC, compatible with the IEEE 802.3 standard. The GMAC can operate in either half or full duplex mode at all supported speeds.

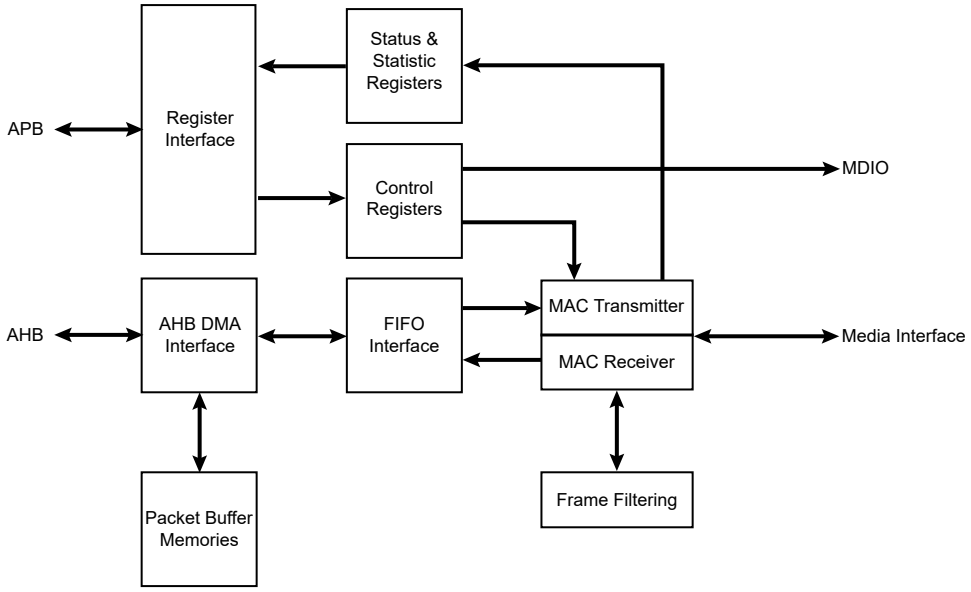
## 38.2 Embedded Characteristics

- Compatible with IEEE Standard 802.3
- 10, 100 Mbps operation
- Full and half duplex operation at all supported speeds of operation
- Statistics Counter Registers for RMON/MIB
- MII/RMII interface to the physical layer
- Integrated physical coding
- Direct memory access (DMA) interface to external memory
- Support for 6 priority queues in DMA
- 8-KByte transmit RAM and 4-KByte receive RAM (refer to Table 38-4 for queue-specific sizes
- Programmable burst length and endianism for DMA
- Interrupt generation to signal receive and transmit completion, errors or other events
- Automatic pad and cyclic redundancy check (CRC) generation on transmitted frames
- Automatic discard of frames received with errors
- Receive and transmit IP, TCP and UDP checksum offload. Both IPv4 and IPv6 packet types supported
- Address checking logic for four specific 48-bit addresses, four type IDs, promiscuous mode, hash matching of unicast and multicast destination addresses and Wake-on-LAN
- Management Data Input/Output (MDIO) interface for physical layer management
- Support for jumbo frames up to 10240 Bytes
- Full duplex flow control with recognition of incoming pause frames and hardware generation of transmitted pause frames
- Half duplex flow control by forcing collisions on incoming frames
- Support for 802.1Q VLAN tagging with recognition of incoming VLAN and priority tagged frames
- Support for 802.1Qbb priority-based flow control
- Programmable Inter Packet Gap (IPG) Stretch
- Recognition of IEEE 1588 PTP frames
- IEEE 1588 time stamp unit (TSU)
- Support for 802.1AS timing and synchronization
- Supports 802.1Qav traffic shaping on two highest priority queues
- Support for 802.3az Energy Efficient Ethernet

© 2021 Microchip Technology Inc.
and its subsidiaries

## 38.3 Block Diagram

**Figure 38-1. Block Diagram**




## 38.4 Signal Interface

The GMAC includes the following signal interfaces:

- MII, RMII to an external PHY
- MDIO interface for external PHY management
- Slave APB interface for accessing GMAC registers
- Master AHB interface for memory access
- GTSUCOMP signal for TSU timer count value comparison

**Table 38-1. GMAC Connections in Different Modes**

| Signal Name | Function | MII | RMII |
|---|---|---|---|
| GTXCK[1] | Transmit Clock or Reference Clock | TXCK | REFCK |
| GTXEN | Transmit Enable | TXEN | TXEN |
| GTX[3..0] | Transmit Data | TXD[3:0] | TXD[1:0] |
| GTXER | Transmit Coding Error | TXER | Not Used |
| GRXCK | Receive Clock | RXCK | Not Used |
| GRXDV | Receive Data Valid | RXDV | CRSDV |
| GRX[3..0] | Receive Data | RXD[3:0] | RXD[1:0] |
| GRXER | Receive Error | RXER | RXER |
| GCRS | Carrier Sense and Data Valid | CRS | Not Used |
| GCOL | Collision Detect | COL | Not Used |
| GMDC | Management Data Clock | MDC | MDC |
| GMDIO | Management Data Input/Output | MDIO | MDIO |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Note:**

1. Input only. GTXCK must be provided with a 25 MHz / 50 MHz external clock signal from the Ethernet PHY for MII / RMII interfaces, respectively.

## 38.5 Product Dependencies

### 38.5.1 I/O Lines

The pins used for interfacing the GMAC may be multiplexed with PIO lines. The programmer must first program the PIO Controller to assign the pins to their peripheral function. If I/O lines of the GMAC are not used by the application, they can be used for other purposes by the PIO Controller.

### 38.5.2 Power Management

The GMAC is not continuously clocked. The user must first enable the GMAC clock in the Power Management Controller before using it.

### 38.5.3 Interrupt Sources

The GMAC interrupt line is connected to one of the internal sources of the interrupt controller. Using the GMAC interrupt requires prior programming of the interrupt controller.

The GMAC features 6 interrupt sources. Refer to the table "Peripheral Identifiers" in the section "Peripherals" for the interrupt numbers for GMAC priority queues.

**Related Links**

14.1 Peripheral Identifiers

## 38.6 Functional Description

### 38.6.1 Media Access Controller

The Transmit Block of the Media Access Controller (MAC) takes data from FIFO, adds preamble, checks and adds padding and frame check sequence (FCS). Both half duplex and full duplex Ethernet modes of operation are supported.

When operating in half duplex mode, the MAC Transmit Block generates data according to the Carrier Sense Multiple Access with Collision Detect (CSMA/CD) protocol. The start of transmission is deferred if Carrier Sense (CRS) is active. If Collision (COL) is detected during transmission, a jam sequence is asserted and the transmission is retried after a random back off. The CRS and COL signals have no effect in full duplex mode.

The Receive Block of the MAC checks for valid preamble, FCS, alignment and length, and presents received frames to the MAC address checking block and FIFO. Software can configure the GMAC to receive jumbo frames of up to 10240 Bytes. It can optionally strip CRC (Cyclic Redundancy Check) from the received frame before transferring it to FIFO.

The Address Checker recognizes four specific 48-bit addresses, can recognize four different types of ID values, and contains a 64-bit Hash register for matching multicast and unicast addresses as required. It can recognize the broadcast address *all-'1'* (0xFFFFFFFFFFFF) and copy all frames. The MAC can also reject all frames that are not VLAN tagged, and recognize Wake on LAN events.

The MAC Receive Block supports offloading of IP, TCP and UDP checksum calculations (both IPv4 and IPv6 packet types supported), and can automatically discard bad checksum frames.

### 38.6.2 IEEE 1588 Time Stamp Unit

The IEEE 1588 time stamp unit (TSU) is implemented as a 94-bit timer.

- The 48 upper bits [93:46] of the timer count seconds and are accessible in the GMAC 1588 Timer Seconds High Register” (GMAC_TSH) and GMAC 1588 Timer Seconds Low Register (GMAC_TSL).
- The 30 lower bits [45:16] of the timer count nanoseconds and are accessible in the GMAC 1588 Timer Nanoseconds Register (GMAC_TN).

© 2021 Microchip Technology Inc. and its subsidiaries

- The lowest 16 bits [15:0] of the timer count sub-nanoseconds.

The 46 lower bits roll over when they have counted to 1s. The timer increments by a programmable period (to approximately 15.2fs resolution) with each MCK period and can also be adjusted in 1ns resolution (incremented or decremented) through APB register accesses.

### 38.6.3 AHB Direct Memory Access Interface

The GMAC DMA controller is connected to the MAC FIFO interface and provides a scatter-gather type capability for packet data storage.

The DMA implements packet buffering where dual-port memories are used to buffer multiple frames.

#### 38.6.3.1 Packet Buffer DMA

- Easier to guarantee maximum line rate due to the ability to store multiple frames in the packet buffer, where the number of frames is limited by the amount of packet buffer memory and Ethernet frame size
- Full store and forward, or partial store and forward programmable options (partial store will cater for shorter latency requirements)
- Support for Transmit TCP/IP checksum offload
- Support for priority queuing
- When a collision on the line occurs during transmission, the packet will be automatically replayed directly from the packet buffer memory rather than having to re-fetch through the AHB (full store and forward ONLY)
- Received erroneous packets are automatically dropped before any of the packet is presented to the AHB (full store and forward ONLY), thus reducing AHB activity
- Supports manual RX packet flush capabilities
- Optional RX packet flush when there is lack of AHB resource

#### 38.6.3.2 Partial Store and Forward Using Packet Buffer DMA

The DMA uses SRAM-based packet buffers, and can be programmed into a low latency mode, known as Partial Store and Forward. This mode allows for a reduced latency as the full packet is not buffered before forwarding.
**Note:** This option is only available when the device is configured for full duplex operation.

This feature is enabled via the programmable TX and RX Partial Store and Forward registers (GMAC_TPSF and GMAC_RPSF). When the transmit Partial Store and Forward mode is activated, the transmitter will only begin to forward the packet to the MAC when there is enough packet data stored in the packet buffer. Likewise, when the receive Partial Store and Forward mode is activated, the receiver will only begin to forward the packet to the AHB when enough packet data is stored in the packet buffer. The amount of packet data required to activate the forwarding process is programmable via watermark registers. These registers are located at the same address as the partial store and forward enable bits.
**Note:** The minimum operational value for the TX partial store and forward watermark is 20. There is no operational limit for the RX partial store and forward watermark.

Enabling Partial Store and Forward is a useful means to reduce latency, but there are performance implications. The GMAC DMA uses separate transmit and receive lists of buffer descriptors, with each descriptor describing a buffer area in memory. This allows Ethernet packets to be broken up and scattered around the AHB memory space.

#### 38.6.3.3 Receive AHB Buffers

Received frames, optionally including FCS, are written to receive AHB buffers stored in memory. The receive buffer depth is programmable in the range of 64 Bytes to 16 KBytes through the DMA Configuration register (GMAC_DCFGR), with the default being 128 Bytes.

The start location for each receive AHB buffer is stored in memory in a list of receive buffer descriptors at an address location pointed to by the receive buffer queue pointer. The base address for the receive buffer queue pointer is configured in software using the Receive Buffer Queue Base Address register (GMAC_RBQB).

Each list entry consists of two words. The first is the address of the receive AHB buffer and the second the receive status.

If the length of a receive frame exceeds the AHB buffer length, the status word for the used buffer is written with zeroes except for the “Start of Frame” bit, which is always set for the first buffer in a frame.

Bit zero of the address field is written to 1 to show that the buffer has been used. The receive buffer manager then reads the location of the next receive AHB buffer and fills that with the next part of the received frame data. AHB

buffers are filled until the frame is complete and the final buffer descriptor status word contains the complete frame status. See the following table for details of the receive buffer descriptor list.

**Table 38-2. Receive Buffer Descriptor Entry**

| Bit | Function |
|---|---|
| Word 0 | |
| 31:2 | Address of beginning of buffer |
| 1 | Wrap—marks last descriptor in receive buffer descriptor list. |
| 0 | Ownership—needs to be zero for the GMAC to write data to the receive buffer. The GMAC sets this to one once it has successfully written a frame to memory.<br>Software has to clear this bit before the buffer can be used again. |
| Word 1 | |
| 31 | Global all ones broadcast address detected |
| 30 | Multicast hash match |
| 29 | Unicast hash match |
| 28 | – |
| 27 | Specific Address Register match found, bit 25 and bit 26 indicate which Specific Address Register causes the match. |
| 26:25 | Specific Address Register match. Encoded as follows:<br>00: Specific Address Register 1 match<br>01: Specific Address Register 2 match<br>10: Specific Address Register 3 match<br>11: Specific Address Register 4 match<br>If more than one specific address is matched only one is indicated with priority 4 down to 1. |
| 24 | This bit has a different meaning depending on whether RX checksum offloading is enabled.<br>With RX checksum offloading disabled: (bit 24 clear in Network Configuration Register)<br>Type ID register match found, bit 22 and bit 23 indicate which type ID register causes the match.<br>With RX checksum offloading enabled: (bit 24 set in Network Configuration Register)<br>0: The frame was not SNAP encoded and/or had a VLAN tag with the Canonical Format Indicator (CFI) bit set.<br>1: The frame was SNAP encoded and had either no VLAN tag or a VLAN tag with the CFI bit not set. |

.....continued

| Bit | Function |
|---|---|
| 23:22 | This bit has a different meaning depending on whether RX checksum offloading is enabled.<br>With RX checksum offloading disabled: (bit 24 clear in Network Configuration)<br>Type ID register match. Encoded as follows:<br>00: Type ID register 1 match<br>01: Type ID register 2 match<br>10: Type ID register 3 match<br>11: Type ID register 4 match<br>If more than one Type ID is matched only one is indicated with priority 4 down to 1.<br>With RX checksum offloading enabled: (bit 24 set in Network Configuration Register)<br>00: Neither the IP header checksum nor the TCP/UDP checksum was checked.<br>01: The IP header checksum was checked and was correct. Neither the TCP nor UDP checksum was checked.<br>10: Both the IP header and TCP checksum were checked and were correct.<br>11: Both the IP header and UDP checksum were checked and were correct. |
| 21 | VLAN tag detected—type ID of 0x8100. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a type ID of 0x8100 |
| 20 | Priority tag detected—type ID of 0x8100 and null VLAN identifier. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a type ID of 0x8100 and a null VLAN identifier. |
| 19:17 | VLAN priority—only valid if bit 21 is set. |
| 16 | Canonical format indicator (CFI) bit (only valid if bit 21 is set). |
| 15 | End of frame—when set the buffer contains the end of a frame. If end of frame is not set, then the only valid status bit is start of frame (bit 14). |
| 14 | Start of frame—when set the buffer contains the start of a frame. If both bits 15 and 14 are set, the buffer contains a whole frame. |
| 13 | This bit has a different meaning depending on whether jumbo frames and ignore FCS modes are enabled. If neither mode is enabled this bit will be zero.<br>With jumbo frame mode enabled: (bit 3 set in Network Configuration Register) Additional bit for length of frame (bit[13]), that is concatenated with bits[12:0]<br>With ignore FCS mode enabled and jumbo frames disabled: (bit 26 set in Network Configuration Register and bit 3 clear in Network Configuration Register) This indicates per frame FCS status as follows:<br>0: Frame had good FCS<br>1: Frame had bad FCS, but was copied to memory as ignore FCS enabled. |
| 12:0 | These bits represent the length of the received frame which may or may not include FCS depending on whether FCS discard mode is enabled.<br>With FCS discard mode disabled: (bit 17 clear in Network Configuration Register)<br>Least significant 12 bits for length of frame including FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit[13] of the descriptor above.<br>With FCS discard mode enabled: (bit 17 set in Network Configuration Register)<br>Least significant 12 bits for length of frame excluding FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit[13] of the descriptor above. |

© 2021 Microchip Technology Inc.
and its subsidiaries

Each receive AHB buffer start location is a word address. The start of the first AHB buffer in a frame can be offset by up to three Bytes, depending on the value written to bits 14 and 15 of the Network Configuration register (GMAC_NCFGR). If the start location of the AHB buffer is offset, the available length of the first AHB buffer is reduced by the corresponding number of Bytes.

To receive frames, the AHB buffer descriptors must be initialized by writing an appropriate address to bits 31:2 in the first word of each list entry. Bit 0 must be written with zero. Bit 1 is the wrap bit and indicates the last entry in the buffer descriptor list.

The start location of the receive buffer descriptor list must be written with the receive buffer queue base address before reception is enabled (receive enable in the Network Control register GMAC_NCR). Once reception is enabled, any writes to the Receive Buffer Queue Base Address register (GMAC_RBQB) are ignored. When read, it will return the current pointer position in the descriptor list, though this is only valid and stable when receive is disabled.

If the filter block indicates that a frame should be copied to memory, the receive data DMA operation starts writing data into the receive buffer. If an error occurs, the buffer is recovered.

An internal counter within the GMAC represents the receive buffer queue pointer and it is not visible through the CPU interface. The receive buffer queue pointer increments by two words after each buffer has been used. It re-initializes to the receive buffer queue base address if any descriptor has its wrap bit set.

As receive AHB buffers are used, the receive AHB buffer manager sets bit zero of the first word of the descriptor to logic one indicating the AHB buffer has been used.

Software should search through the “used” bits in the AHB buffer descriptors to find out how many frames have been received, checking the start of frame and end of frame bits.

When the DMA is configured in the packet buffer Partial Store And Forward mode, received frames are written out to the AHB buffers as soon as enough frame data exists in the packet buffer. For both cases, this may mean several full AHB buffers are used before some error conditions can be detected. If a receive error is detected the receive buffer currently being written will be recovered. Previous buffers will not be recovered. As an example, when receiving frames with cyclic redundancy check (CRC) errors or excessive length, it is possible that a frame fragment might be stored in a sequence of AHB receive buffers. Software can detect this by looking for start of frame bit set in a buffer following a buffer with no end of frame bit set.

To function properly, a 10/100 Ethernet system should have no excessive length frames or frames greater than 128 Bytes with CRC errors. Collision fragments will be less than 128 Bytes long, therefore it will be a rare occurrence to find a frame fragment in a receive AHB buffer, when using the default value of 128 Bytes for the receive buffers size.

When in packet buffer full store and forward mode, only good received frames are written out of the DMA, so no fragments will exist in the AHB buffers due to MAC receiver errors. There is still the possibility of fragments due to DMA errors, for example used bit read on the second buffer of a multi-buffer frame.

If bit zero of the receive buffer descriptor is already set when the receive buffer manager reads the location of the receive AHB buffer, the buffer has been already used and cannot be used again until software has processed the frame and cleared bit zero. In this case, the “buffer not available” bit in the receive status register is set and an interrupt triggered. The receive resource error statistics register is also incremented.

When the DMA is configured in the packet buffer full store and forward mode, the user can optionally select whether received frames should be automatically discarded when no AHB buffer resource is available. This feature is selected via the DMA Discard Receive Packets bit in the DMA Configuration register (GMAC_DCFGR.DDRP). By default, the received frames are not automatically discarded. If this feature is off, then received packets will remain to be stored in the SRAM-based packet buffer until AHB buffer resource next becomes available. This may lead to an eventual packet buffer overflow if packets continue to be received when bit zero (used bit) of the receive buffer descriptor remains set.
**Note:** After a used bit has been read, the receive buffer manager will re-read the location of the receive buffer descriptor every time a new packet is received. When the DMA is not configured in the packet buffer full store and forward mode and a used bit is read, the frame currently being received will be automatically discarded.

When the DMA is configured in the packet buffer full store and forward mode, a receive overrun condition occurs when the receive SRAM-based packet buffer is full, or because HRESP was not OK. In all other modes, a receive overrun condition occurs when either the AHB bus was not granted quickly enough, or because HRESP was not OK, or because a new frame has been detected by the receive block, but the status update or write back for the previous frame has not yet finished. For a receive overrun condition, the receive overrun interrupt is asserted and the buffer currently being written is recovered. The next frame that is received whose address is recognized reuses the buffer.

© 2021 Microchip Technology Inc.
and its subsidiaries

In any packet buffer mode, writing a '1' to the Flush Next Package bit in the NCR register (GMAC_NCR.FNP) will force a packet from the external SRAM-based receive packet buffer to be flushed. This feature is only acted upon when the RX DMA is not currently writing packet data out to AHB, i.e., it is in an IDLE state. If the RX DMA is active, GMAC_NCR.FNP=1 is ignored.

#### 38.6.3.4 Transmit AHB Buffers

Frames to transmit are stored in one or more transmit AHB buffers. Transmit frames can be between 1 and 16384 Bytes long, so it is possible to transmit frames longer than the maximum length specified in the IEEE 802.3 standard. It should be noted that zero length AHB buffers are allowed and that the maximum number of buffers permitted for each transmit frame is 128.

The start location for each transmit AHB buffer is stored in memory in a list of transmit buffer descriptors at a location pointed to by the transmit buffer queue pointer. The base address for this queue pointer is set in software using the Transmit Buffer Queue Base Address register. Each list entry consists of two words. The first is the Byte address of the transmit buffer and the second containing the transmit control and status. For the packet buffer DMA, the start location for each AHB buffer is a Byte address, the bottom bits of the address being used to offset the start of the data from the data-word boundary (i.e., bits 2,1 and 0 are used to offset the address for 64-bit data paths).

Frames can be transmitted with or without automatic Cyclic Redundancy Checksum (CRC) generation. If CRC is automatically generated, pad will also be automatically generated to take frames to a minimum length of 64 Bytes. When CRC is not automatically generated (as defined in word 1 of the transmit buffer descriptor), the frame is assumed to be at least 64 Bytes long and pad is not generated.

An entry in the transmit buffer descriptor list is described in this table:

**Table 38-3. Transmit Buffer Descriptor Entry**

| Bit | Function |
|---|---|
| Word 0 | |
| 31:0 | Byte address of buffer |
| Word 1 | |
| 31 | Used—must be zero for the GMAC to read data to the transmit buffer. The GMAC sets this to one for the first buffer of a frame once it has been successfully transmitted. Software must clear this bit before the buffer can be used again. |
| 30 | Wrap—marks last descriptor in transmit buffer descriptor list. This can be set for any buffer within the frame. |
| 29 | Retry limit exceeded, transmit error detected |
| 28 | Reserved. |
| 27 | Transmit frame corruption due to AHB error—set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted).<br>Also set if single frame is too large for configured packet buffer memory size. |
| 26 | Late collision, transmit error detected. |
| 25:23 | Reserved |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | |
|---|---|
| **Bit** | **Function** |
| 22:20 | Transmit IP/TCP/UDP checksum generation offload errors:<br>000: No Error.<br>001: The Packet was identified as a VLAN type, but the header was not fully complete, or had an error in it.<br>010: The Packet was identified as a SNAP type, but the header was not fully complete, or had an error in it.<br>011: The Packet was not of an IP type, or the IP packet was invalidly short, or the IP was not of type IPv4/IPv6.<br>100: The Packet was not identified as VLAN, SNAP or IP.<br>101: Non supported packet fragmentation occurred. For IPv4 packets, the IP checksum was generated and inserted.<br>110: Packet type detected was not TCP or UDP. TCP/UDP checksum was therefore not generated. For IPv4 packets, the IP checksum was generated and inserted.<br>111: A premature end of packet was detected and the TCP/UDP checksum could not be generated. |
| 19:17 | Reserved |
| 16 | No CRC to be appended by MAC. When set, this implies that the data in the buffers already contains a valid CRC, hence no CRC or padding is to be appended to the current frame by the MAC.<br>This control bit must be set for the first buffer in a frame and will be ignored for the subsequent buffers of a frame.<br>Note that this bit must be clear when using the transmit IP/TCP/UDP checksum generation offload, otherwise checksum generation and substitution will not occur. |
| 15 | Last buffer, when set this bit will indicate the last buffer in the current frame has been reached. |
| 14 | Reserved |
| 13:0 | Length of buffer |

To transmit frames, the buffer descriptors must be initialized by writing an appropriate Byte address to bits [31:0] of the first word of each descriptor list entry.

The second word of the transmit buffer descriptor is initialized with control information that indicates the length of the frame, whether or not the MAC is to append CRC and whether the buffer is the last buffer in the frame.

After transmission the status bits are written back to the second word of the first buffer along with the used bit. Bit 31 is the used bit which must be zero when the control word is read if transmission is to take place. It is written to '1' once the frame has been transmitted. Bits[29:20] indicate various transmit error conditions. Bit 30 is the wrap bit which can be set for any buffer within a frame. If no wrap bit is encountered the queue pointer continues to increment.

The Transmit Buffer Queue Base Address register can only be updated while transmission is disabled or halted; otherwise any attempted write will be ignored. When transmission is halted the transmit buffer queue pointer will maintain its value. Therefore when transmission is restarted the next descriptor read from the queue will be from immediately after the last successfully transmitted frame. As long as transmit is disabled by writing a '0' to the Transmit Enable bit in the Network Control register (GMAC_NCR.TXEN), the transmit buffer queue pointer resets to point to the address indicated by the Transmit Buffer Queue Base Address register (GMAC_TBQB).
**Note:** Disabling receive does not have the same effect on the receive buffer queue pointer.

Once the transmit queue is initialized, transmit is activated by writing a '1' to the Start Transmission bit of the Network Control register (GMAC_NCR.TSTART). Transmit is halted when a buffer descriptor with its used bit set is read, a transmit error occurs, or by writing to the Transmit Halt bit of the Network Control register (GMAC_NCR.THALT). Transmission is suspended if a pause frame is received while the Transmit Pause Frame bit is '1' in the Network Configuration register (GMAC_NCR.TXPF). Rewriting the Start bit (GMAC_NCR.TSTART) while transmission is active is allowed. This is implemented by the Transmit Go variable which is readable in the Transmit Status register (GMAC_TSR.TXGO). The TXGO variable is reset when:

- Transmit is disabled.

- A buffer descriptor with its ownership bit set is read.
- Bit 10, THALT, of the Network Control register is written.
- There is a transmit error such as too many retries or a transmit underrun.

To set TXGO, write a '1' to GMAC_NCR.TSTART. Transmit halt does not take effect until any ongoing transmit finishes.

If the DMA is configured for packet buffer Partial Store and Forward mode and a collision occurs during transmission of a multi-buffer frame, transmission will automatically restart from the first buffer of the frame. For packet buffer mode, the entire contents of the frame are read into the transmit packet buffer memory, so the retry attempt will be replayed directly from the packet buffer memory rather than having to re-fetch through the AHB.

If a used bit is read midway through transmission of a multi-buffer frame, this is treated as a transmit error. Transmission stops, GTXER is asserted and the FCS will be bad.

If transmission stops due to a transmit error or a used bit being read, transmission restarts from the first buffer descriptor of the frame being transmitted when the transmit start bit is rewritten.

#### 38.6.3.5 DMA Bursting on the AHB

The DMA will always use SINGLE, or INCR type AHB accesses for buffer management operations. When performing data transfers, the AHB burst length is selected by the Fixed Burst Length for DMA Data Operations bit field in the DMA Configuration register (GMAC_DCFGR.FBLDO) so that either SINGLEor fixed length incrementing bursts (INCR4, INCR8 or INCR16) are used where possible:

When there is enough space and enough data to be transferred, the programmed fixed length bursts will be used. If there is not enough data or space available, for example when at the beginning or the end of a buffer, SINGLE type accesses are used. Also SINGLE type accesses are used at 1024 Byte boundaries, so that the 1 KByte boundaries are not burst over as per AHB requirements.

The DMA will not terminate a fixed length burst early, unless an error condition occurs on the AHB or if receive or transmit are disabled in the Network Control register (GMAC_NCR).

#### 38.6.3.6 DMA Packet Buffer

The DMA uses packet buffers for both transmit and receive paths. This mode allows multiple packets to be buffered in both transmit and receive directions. This allows the DMA to withstand far greater access latencies on the AHB and make more efficient use of the AHB bandwidth. There are two modes of operation—Full Store and Forward and Partial Store and Forward.

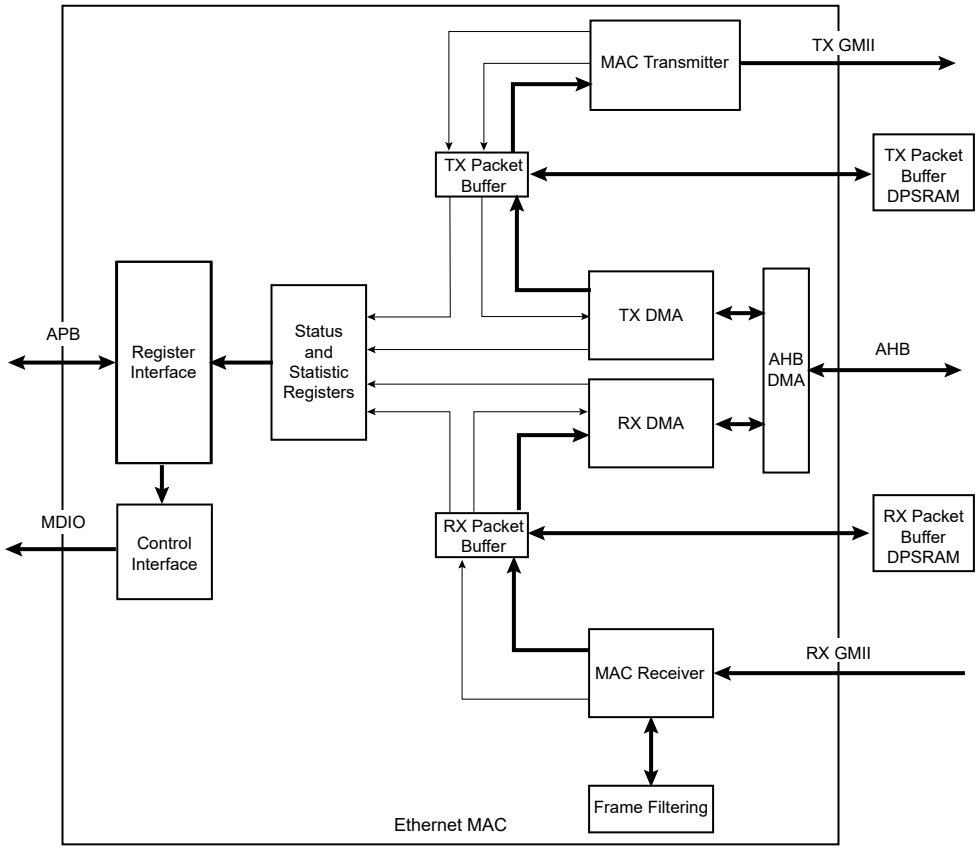
As described above, the DMA can be programmed into a low latency mode, known as Partial Store and Forward. For further details of this mode, see the related Links.

When the DMA is in full store and forward mode, full packets are buffered which provides the possibility to:

- Discard packets with error on the receive path before they are partially written out of the DMA, thus saving AHB bus bandwidth and driver processing overhead,
- Retry collided transmit frames from the buffer, thus saving AHB bus bandwidth,
- Implement transmit IP/TCP/UDP checksum generation offload.

With the packet buffers included, the structure of the GMAC data paths is shown in this image:

**Figure 38-2. Data Paths with Packet Buffers Included**




### 38.6.3.7 Transmit Packet Buffer

The transmitter packet buffer will continue attempting to fetch frame data from the AHB system memory until the packet buffer itself is full, at which point it will attempt to maintain its full level.

To accommodate the status and statistics associated with each frame, three words per packet (or two if the GMAC is configured in 64-bit data path mode) are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet. Storing the status in the DPRAM is required in order to decouple the DMA interface of the buffer from the MAC interface, to update the MAC status/ statistics and to generate interrupts in the order in which the packets that they represent were fetched from the AHB memory.

If any errors occur on the AHB while reading the transmit frame, the fetching of packet data from AHB memory is halted. The MAC transmitter will continue to fetch packet data, thereby emptying the packet buffer and allowing any good (non-erroneous) frames to be transmitted successfully. Once these have been fully transmitted, the status/ statistics for the erroneous frame will be updated and software will be informed via an interrupt that an AHB error occurred. This way, the error is reported in the correct packet order.

The transmit packet buffer will only attempt to read more frame data from the AHB when space is available in the packet buffer memory. If space is not available it must wait until the a packet fetched by the MAC completes transmission and is subsequently removed from the packet buffer memory.
**Note:** If full store and forward mode is active and if a single frame is fetched that is too large for the packet buffer memory, the frame is flushed and the DMA halted with an error status. This is because a complete frame must be written into the packet buffer before transmission can begin, and therefore the minimum packet buffer memory size should be chosen to satisfy the maximum frame to be transmitted in the application.

In full store and forward mode, once the complete transmit frame is written into the packet buffer memory, a trigger is sent across to the MAC transmitter, which will then begin reading the frame from the packet buffer memory. Since the whole frame is present and stable in the packet buffer memory an underflow of the transmitter is not possible. The frame is kept in the packet buffer until notification is received from the MAC that the frame data has either been successfully transmitted or can no longer be retransmitted (too many retries in half duplex mode). When this

notification is received the frame is flushed from memory to make room for a new frame to be fetched from AHB system memory.

In Partial Store and Forward mode, a trigger is sent across to the MAC transmitter as soon as sufficient packet data is available, which will then begin fetching the frame from the packet buffer memory. If, after this point, the MAC transmitter is able to fetch data from the packet buffer faster than the AHB DMA can fill it, an underflow of the transmitter is possible. In this case, the transmission is terminated early, and the packet buffer is completely flushed. Transmission can only be restarted by writing a '1' to the Transmit Start bit in the Network Control register (GMAC_NCR.TSTART).

In half duplex mode, the frame is kept in the packet buffer until notification is received from the MAC that the frame data has either been successfully transmitted or can no longer be retransmitted (too many retries in half duplex mode). When this notification is received the frame is flushed from memory to make room for a new frame to be fetched from AHB system memory.

In full duplex mode, the frame is removed from the packet buffer on the fly.

Other than underflow, the only MAC related errors that can occur are due to collisions during half duplex transmissions. When a collision occurs the frame still exists in the packet buffer memory so can be retried directly from there. After sixteen failed transmit attempts, the frame will be flushed from the packet buffer.

#### 38.6.3.8 Receive Packet Buffer

The receive packet buffer stores frames from the MAC receiver along with their status and statistics. Frames with errors are flushed from the packet buffer memory, while good frames are pushed onto the DMA AHB interface.

The receiver packet buffer monitors the FIFO write interface from the MAC receiver and translates the FIFO pushes into packet buffer writes. At the end of the received frame the status and statistics are buffered so that the information can be used when the frame is read out. When programmed in full store and forward mode and the frame has an error, the frame data is immediately flushed from the packet buffer memory allowing subsequent frames to utilize the freed up space. The status and statistics for bad frames are still used to update the GMAC registers.

To accommodate the status and statistics associated with each frame, three words per packet (or two if configured in 64-bit datapath mode) are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet.

The receiver packet buffer will also detect a full condition so that an overflow condition can be detected. If this occurs, subsequent packets are dropped and an RX overflow interrupt is raised.

For full store and forward, the DMA only begins packet fetches once the status and statistics for a frame are available. If the frame has a bad status due to a frame error, the status and statistics are passed on to the GMAC registers. If the frame has a good status, the information is used to read the frame from the packet buffer memory and burst onto the AHB using the DMA buffer management protocol. Once the last frame data has been transferred to the packet buffer, the status and statistics are updated to the GMAC registers.

If Partial Store and Forward mode is active, the DMA will begin fetching the packet data before the status is available. As soon as the status becomes available, the DMA will fetch this information as soon as possible before continuing to fetch the remainder of the frame. Once the last frame data has been transferred to the packet buffer, the status and statistics are updated to the GMAC registers.

#### 38.6.3.9 Priority Queuing in the DMA

The DMA by default uses a single transmit and receive queue. This means the list of transmit/receive buffer descriptors point to data buffers associated with a single transmit/receive data stream. The GMAC can select up to 6 priority queues. Each queue has an independent list of buffer descriptors pointing to separate data streams.

The table below gives the DPRAM size associated with each queue.

**Table 38-4. Queue Size**

| Queue Number | Queue Size |
|---|---|
| 5 (highest priority) | 1 KB |
| 4 | 2 KB |
| 3 | 2 KB |

| .....continued | |
|---|---|
| **Queue Number** | **Queue Size** |
| 2 | 512 bytes |
| 1 | 512 bytes |
| 0 (lowest priority) | 2 KB |

In the transmit direction, higher priority queues are always serviced before lower priority queues, with Q0 as lowest priority and Q5 as highest priority. This strict priority scheme requires the user to ensure that high priority traffic is constrained so that lower priority traffic will have required bandwidth. The GMAC DMA will determine the next queue to service by initiating a sequence of buffer descriptor reads interrogating the ownership bits of each. The buffer descriptor corresponding to the highest priority queue is read first.

As an example, if the ownership bit of this descriptor is set, the DMA will progress by reading the 2nd highest priority queue's descriptor. If that ownership bit read of this lower priority queue is set as well, the DMA will read the 3rd highest priority queue's descriptor. If all the descriptors return an ownership bit set, a resource error has occurred, so an interrupt is generated and transmission is automatically halted. Transmission can only be restarted by writing a '1' to the Transmission Start bit in the Network Control register (GMAC_NCR.TSTART). The GMAC DMA will need to identify the highest available queue to transmit from when the TSTART bit is written and the TX is in a halted state, or when the last word of any packet has been fetched from external AHB memory.

The GMAC transmit DMA maximizes the effectiveness of priority queuing by ensuring that high priority traffic be transmitted as early as possible after being fetched from AHB. High priority traffic fetched from AHB will be pushed to the MAC layer, depending on traffic shaping being enabled and the associated credit value for that queue, before any lower priority traffic that may pre-exist in the transmit SRAM-based packet buffer. This is achieved by separating the transmit SRAM-based packet buffer into regions, one region per queue. The size of each region determines the amount of SRAM space allocated per queue.

For each queue, there is an associated Transmit Buffer Queue Base Address register (GMAC_TBQB). For the lowest priority queue (or the only queue when only one queue is selected), the Transmit Buffer Queue Base Address is located at address 0x1C. For all other queues, the Transmit Buffer Queue Base Address registers are located at sequential addresses starting at address 0x440.

In the receive direction each packet is written to AHB data buffers in the order that it is received. For each queue, there is an independent set of receive AHB buffers for each queue. There is therefore a separate Receive Buffer Queue Base Address register for each queue (GMAC_RBQBAx). For the lowest priority queue (or the only queue when only one queue is selected), the Receive Buffer Queue Base Address is located at address 0x18. For all other queues, the Receive Buffer Queue Base Address registers are located at sequential addresses starting at address 0x480. Every received packet will pass through a programmable screening algorithm which will allocate a particular queue to that frame. The user interface to the screeners is through two types of programmable registers:

- Screening Type 1 registers: The module features 4 Screening Type 1 registers. Screening Type 1 registers hold values to match against specific IP and UDP fields of the received frames. The fields matched against are DS (Differentiated Services field of IPv4 frames), TC (Traffic class field of IPv6 frames) and/or the UDP destination port.
- Screening Type 2 registers: The module features 8 Screening Type 2 registers GMAC_ST2RPQ. Screening Type 2 registers operate independently of Screening Type 1 registers and offer additional match capabilities. Screening Type 2 allows a screen to be configured that is the combination of all or any of the following comparisons:
  - An enable bit VLAN priority, VLANE. A VLAN priority match will be performed if the VLAN priority enable is set. The extracted priority field in the VLAN header is compared against VLANP in the GMAC_ST2RPQ itself.
  - An enable bit EtherType, ETHE. The EtherType field I2ETH inside the GMAC_ST2RPQ maps to one of 4 EtherType match registers, GMAC_ST2ER. The extracted EtherType is compared against GMAC_ST2ER designated by this EtherType field.
  - An enable bit Compare A, COMPAE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.
  - An enable bit Compare B, COMPBE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.

© 2021 Microchip Technology Inc. and its subsidiaries

– An enable bit Compare C, COMPCE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.

Each screener type has an enable bit, a match pattern and a queue number. If a received frame matches on an enabled screening register, then the frame will be tagged with the queue value in the associated screening register, and forwarded onto the DMA and subsequently into the external memory associated with that queue. If two screeners are matched then the one which resides at the lowest register address will take priority so care must be taken on the selection of the screener location.

When the priority queuing feature is enabled, the number of interrupt outputs from the GMAC core is increased to match the number of supported queues. The number of Interrupt Status registers is increased by the same number. Only DMA related events are reported using the individual interrupt outputs, as the GMAC can relate these events to specific queues. All other events generated within the GMAC are reported in the interrupt associated with the lowest priority queue. For the lowest priority queue (or the only queue when only 1 queue is selected), the Interrupt Status register is located at address 0x24. For all other queues, the Interrupt Status register is located at sequential addresses starting at address 0x400.

**Note:** The address matching is the first level of filtering. If there is a match, the screeners are the next level of filtering for routing the data to the appropriate queue. See MAC Filtering Block for more details.

The additional screening done by the functions Compare A, B, and C each have an enable bit and compare register field. COMPA, COMPB and COMPC in GMAC_ST2RPQ are pointers to a configured offset (OFFSVAL), value (COMPVAL), and mask (MASKVAL). If enabled, the compare is true if the data at the offset into the frame, ANDed with MASKVAL, is equal to the value of COMPVAL ANDed with MASKVAL. A 16-bit word comparison is done. The byte at the offset number of bytes from the index start is compared to bits 7:0 of the configured COMPVAL and MASKVAL. The byte at the offset number of bytes + 1 from the index start is compared to bits 15:8 of the configured COMPVAL and MASKVAL.

The offset value in bytes, OFFSVAL, ranges from 0 to 127 bytes from either the start of the frame, the byte after the EtherType field, the byte after the IP header (IPv4 or IPv6) or the byte after the TCP/UDP header. Note the logic to decode the IP header or the TCP/UDP header is reused from the TCP/UDP/IP checksum offload logic and therefore has the same restrictions on use (the main limitation is that IP fragmentation is not supported). Refer to the Checksum Offload for IP, TCP and UDP section of this documentation for further details.

Compare A, B, and C use a common set of 24 GMAC_ST2CW0/1 registers, thus all COMPA, COMPB and COMPC fields in the registers GMAC_ST2RPQ point to a single pool of 24 GMAC_ST2CW0/1 registers.

Note that Compare A, B and C together allow matching against an arbitrary 48 bits of data and so can be used to match against a MAC address.

All enabled comparisons are ANDed together to form the overall type 2 screening match.

**Related Links**

38.6.6 Checksum Offload for IP, TCP and UDP

### 38.6.4 MAC Transmit Block

The MAC transmitter can operate in either half duplex or full duplex mode and transmits frames in accordance with the Ethernet IEEE 802.3 standard. In half duplex mode, the CSMA/CD protocol of the IEEE 802.3 specification is followed.

A small input buffer receives data through the FIFO interface which will extract data in 32-bit form. All subsequent processing prior to the final output is performed in bytes.

Transmit data can be output using the MII interface.

Frame assembly starts by adding preamble and the start frame delimiter. Data is taken from the transmit FIFO interface a word at a time.

If necessary, padding is added to take the frame length to 60 bytes. CRC is calculated using an order 32-bit polynomial. This is inverted and appended to the end of the frame taking the frame length to a minimum of 64 bytes. If the no CRC bit is set in the second word of the last buffer descriptor of a transmit frame, neither pad nor CRC are appended. The no CRC bit can also be set through the FIFO interface.

In full duplex mode (at all data rates), frames are transmitted immediately. Back to back frames are transmitted at least 96 bit times apart to guarantee the interframe gap.

© 2021 Microchip Technology Inc. and its subsidiaries

In half duplex mode, the transmitter checks carrier sense. If asserted, the transmitter waits for the signal to become inactive, and then starts transmission after the interframe gap of 96 bit times. If the collision signal is asserted during transmission, the transmitter will transmit a jam sequence of 32 bits taken from the data register and then retry transmission after the back off time has elapsed. If the collision occurs during either the preamble or Start Frame Delimiter (SFD), then these fields will be completed prior to generation of the jam sequence.

The back off time is based on an XOR of the 10 least significant bits of the data coming from the transmit FIFO interface and a 10-bit pseudo random number generator. The number of bits used depends on the number of collisions seen. After the first collision 1 bit is used, then the second 2 bits and so on up to the maximum of 10 bits. All 10 bits are used above ten collisions. An error will be indicated and no further attempts will be made if 16 consecutive attempts cause collision. This operation is compliant with the description in Clause 4.2.3.2.5 of the IEEE 802.3 standard which refers to the truncated binary exponential back off algorithm.

In 10/100 mode, both collisions and late collisions are treated identically, and back off and retry will be performed up to 16 times. This condition is reported in the transmit buffer descriptor word 1 (late collision, bit 26) and also in the Transmit Status register (late collision, bit 7). An interrupt can also be generated (if enabled) when this exception occurs, and bit 5 in the Interrupt Status register will be set.

In all modes of operation, if the transmit DMA underruns, a bad CRC is automatically appended using the same mechanism as jam insertion and the GTXER signal is asserted. For a properly configured system this should never happen and also it is impossible if configured to use the DMA with packet buffers, as the complete frame is buffered in local packet buffer memory.

By setting when bit 28 is set in the Network Configuration register, the Inter Packet Gap (IPG) may be stretched beyond 96 bits depending on the length of the previously transmitted frame and the value written to the IPG Stretch register (GMAC_IPGS). The least significant 8 bits of the IPG Stretch register multiply the previous frame length (including preamble). The next significant 8 bits (+1 so as not to get a divide by zero) divide the frame length to generate the IPG. IPG stretch only works in full duplex mode and when bit 28 is set in the Network Configuration register. The IPG Stretch register cannot be used to shrink the IPG below 96 bits.

If the back pressure bit is set in the Network Control register, or if the HDFC configuration bit is set in the GMAC_UR register (10M or 100M half duplex mode), the transmit block transmits 64 bits of data, which can consist of 16 nibbles of 1011 or in bit rate mode 64 1s, whenever it sees an incoming frame to force a collision. This provides a way of implementing flow control in half duplex mode.

### 38.6.5 MAC Receive Block

All processing within the MAC receive block is implemented using a 16-bit data path. The MAC receive block checks for valid preamble, FCS, alignment and length, presents received frames to the FIFO interface and stores the frame destination address for use by the address checking block.

If, during the frame reception, the frame is found to be too long, a bad frame indication is sent to the FIFO interface. The receiver logic ceases to send data to memory as soon as this condition occurs.

At end of frame reception the receive block indicates to the DMA block whether the frame is good or bad. The DMA block will recover the current receive buffer if the frame was bad.

Ethernet frames are normally stored in DMA memory complete with the FCS. Setting the FCS remove bit in the network configuration (bit 17) causes frames to be stored without their corresponding FCS. The reported frame length field is reduced by four bytes to reflect this operation.

The receive block signals to the register block to increment the alignment, CRC (FCS), short frame, long frame, jabber or receive symbol errors when any of these exception conditions occur.

If bit 26 is set in the network configuration, CRC errors will be ignored and CRC errored frames will not be discarded, though the Frame Check Sequence Errors statistic register will still be incremented. Additionally, if not enabled for jumbo frames mode, then bit[13] of the receiver descriptor word 1 will be updated to indicate the FCS validity for the particular frame. This is useful for applications such as EtherCAT whereby individual frames with FCS errors must be identified.

Received frames can be checked for length field error by setting the length field error frame discard bit of the Network Configuration register (bit-16). When this bit is set, the receiver compares a frame's measured length with the length field (bytes 13 and 14) extracted from the frame. The frame is discarded if the measured length is shorter. This checking procedure is for received frames between 64 bytes and 1518 bytes in length.

© 2021 Microchip Technology Inc.
and its subsidiaries

Each discarded frame is counted in the 10-bit length field error statistics register. Frames where the length field is greater than or equal to 0x0600 hex will not be checked.

### 38.6.6 Checksum Offload for IP, TCP and UDP

The GMAC can be programmed to perform IP, TCP and UDP checksum offloading in both receive and transmit directions, which is enabled by setting bit 24 in the Network Configuration register for receive and bit 11 in the DMA Configuration register for transmit.

IPv4 packets contain a 16-bit checksum field, which is the 16-bit 1's complement of the 1's complement sum of all 16-bit words in the header. TCP and UDP packets contain a 16-bit checksum field, which is the 16-bit 1's complement of the 1's complement sum of all 16-bit words in the header, the data and a conceptual IP pseudo header.

To calculate these checksums in software requires each byte of the packet to be processed. For TCP and UDP this can use a large amount of processing power. Offloading the checksum calculation to hardware can result in significant performance improvements.

For IP, TCP or UDP checksum offload to be useful, the operating system containing the protocol stack must be aware that this offload is available so that it can make use of the fact that the hardware can either generate or verify the checksum.

#### 38.6.6.1 Receiver Checksum Offload

When receive checksum offloading is enabled in the GMAC Network Configuration Register (NCFGR.RXCOEN), the IPv4 header checksum is checked as per RFC 791, where the packet meets the following criteria:

- If present, the VLAN header must be four octets long and the CFI bit must not be set.
- Encapsulation must be RFC 894 Ethernet Type Encoding or RFC 1042 SNAP Encoding.
- IPv4 packet
- IP header is of a valid length

The GMAC also checks the TCP checksum as per RFC 793, or the UDP checksum as per RFC 768, if the following criteria are met:

- IPv4 or IPv6 packet
- Good IP header checksum (if IPv4)
- No IP fragmentation
- TCP or UDP packet

When an IP, TCP or UDP frame is received, the receive buffer descriptor gives an indication if the GMAC was able to verify the checksums. There is also an indication if the frame had SNAP encapsulation. These indication bits will replace the type ID match indication bits when the receive checksum offload is enabled. For details of these indication bits refer to "Receive Buffer Descriptor Entry".

If any of the checksums are verified as incorrect by the GMAC, the packet is discarded and the appropriate statistics counter incremented.

#### 38.6.6.2 Transmitter Checksum Offload

The transmitter checksum offload is only available if the full store and forward mode is enabled. This is because the complete frame to be transmitted must be read into the packet buffer memory before the checksum can be calculated and written back into the headers at the beginning of the frame.

Transmitter checksum offload is enabled by setting bit [11] in the DMA Configuration register. When enabled, it will monitor the frame as it is written into the transmitter packet buffer memory to automatically detect the protocol of the frame. Protocol support is identical to the receiver checksum offload.

For transmit checksum generation and substitution to occur, the protocol of the frame must be recognized and the frame must be provided without the FCS field, by making sure that bit [16] of the transmit descriptor word 1 is clear. If the frame data already had the FCS field, this would be corrupted by the substitution of the new checksum fields.

If these conditions are met, the transmit checksum offload engine will calculate the IP, TCP and UDP checksums as appropriate. Once the full packet is completely written into packet buffer memory, the checksums will be valid and the relevant DPRAM locations will be updated for the new checksum fields as per standard IP/TCP and UDP packet structures.

© 2021 Microchip Technology Inc. and its subsidiaries

If the transmitter checksum engine is prevented from generating the relevant checksums, bits [22:20] of the transmitter DMA writeback status will be updated to identify the reason for the error. Note that the frame will still be transmitted but without the checksum substitution, as typically the reason that the substitution did not occur was that the protocol was not recognized.

### 38.6.7 MAC Filtering Block

The filter block determines which frames should be written to the FIFO interface and on to the DMA.

Whether a frame is passed depends on what is enabled in the Network Configuration register, the state of the external matching pins, the contents of the specific address, type and Hash registers and the frame's destination address and type field.

If bit 25 of the Network Configuration register is not set, a frame will not be copied to memory if the GMAC is transmitting in half duplex mode at the time a destination address is received.

Ethernet frames are transmitted a byte at a time, least significant bit first. The first six bytes (48 bits) of an Ethernet frame make up the destination address. The first bit of the destination address, which is the LSB of the first byte of the frame, is the group or individual bit. This is one for multicast addresses and zero for unicast. The all ones address is the broadcast address and a special case of multicast.

The GMAC supports recognition of four specific addresses. Each specific address requires two registers, Specific Address register Bottom and Specific Address register Top. Specific Address register Bottom stores the first four bytes of the destination address and Specific Address register Top contains the last two bytes. The addresses stored can be specific, group, local or universal.

The destination address of received frames is compared against the data stored in the Specific Address registers once they have been activated. The addresses are deactivated at reset or when their corresponding Specific Address register Bottom is written. They are activated when Specific Address register Top is written. If a receive frame address matches an active address, the frame is written to the FIFO interface and on to DMA memory.

Frames may be filtered using the type ID field for matching. Four type ID registers exist in the register address space and each can be enabled for matching by writing a one to the MSB (bit 31) of the respective register. When a frame is received, the matching is implemented as an OR function of the various types of match.

The contents of each type ID register (when enabled) are compared against the length/type ID of the frame being received (e.g., bytes 13 and 14 in non-VLAN and non-SNAP encapsulated frames) and copied to memory if a match is found. The encoded type ID match bits (Word 0, Bit 22 and Bit 23) in the receive buffer descriptor status are set indicating which type ID register generated the match, if the receive checksum offload is disabled.

The reset state of the type ID registers is zero, hence each is initially disabled.

The following example illustrates the use of the address and type ID match registers for a MAC address of 21:43:65:87:A9:CB:

| | |
|---|---|
| Preamble | 55 |
| SFD | D5 |
| DA (Octet 0 - LSB) | 21 |
| DA (Octet 1) | 43 |
| DA (Octet 2) | 65 |
| DA (Octet 3) | 87 |
| DA (Octet 4) | A9 |
| DA (Octet 5 - MSB) | CB |
| SA (LSB) | 00 (see **Note**) |
| SA | 00(see **Note**) |
| SA | 00(see **Note**) |
| SA | 00(see **Note**) |

| | |
|---|---|
| SA | 00(see **Note**) |
| SA (MSB) | 00(see **Note**) |
| Type ID (MSB) | 43 |
| Type ID (LSB) | 21 |

**Note:** Contains the address of the transmitting device.

The previous sequence shows the beginning of an Ethernet frame. Byte order of transmission is from top to bottom, as shown. For a successful match to specific address 1, the following address matching registers must be set up:

Specific Address 1 Bottom register (GMAC_SAB1) (Address 0x088) 0x87654321

Specific Address 1 Top register (GMAC_SAT1) (Address 0x08C) 0x0000CBA9

For a successful match to the type ID, the following Type ID Match 1 register must be set up:

Type ID Match 1 register (GMAC_TIDM1) (Address 0x0A8) 0x80004321

### 38.6.8 Broadcast Address

Frames with the broadcast address of 0xFFFFFFFFFFFF are stored to memory only if the 'no broadcast' bit in the Network Configuration register is set to zero.

### 38.6.9 Hash Addressing

The hash address register is 64 bits long and takes up two locations in the memory map. The least significant bits are stored in Hash Register Bottom and the most significant bits in Hash Register Top.

The unicast hash enable and the multicast hash enable bits in the Network Configuration register enable the reception of hash matched frames. The destination address is reduced to a 6-bit index into the 64-bit Hash register using the following hash function: The hash function is an XOR of every sixth bit of the destination address.

hash_index[05] = da[05] ^ da[11] ^ da[17] ^ da[23] ^ da[29] ^ da[35] ^ da[41] ^ da[47]

hash_index[04] = da[04] ^ da[10] ^ da[16] ^ da[22] ^ da[28] ^ da[34] ^ da[40] ^ da[46]

hash_index[03] = da[03] ^ da[09] ^ da[15] ^ da[21] ^ da[27] ^ da[33] ^ da[39] ^ da[45]

hash_index[02] = da[02] ^ da[08] ^ da[14] ^ da[20] ^ da[26] ^ da[32] ^ da[38] ^ da[44]

hash_index[01] = da[01] ^ da[07] ^ da[13] ^ da[19] ^ da[25] ^ da[31] ^ da[37] ^ da[43]

hash_index[00] = da[00] ^ da[06] ^ da[12] ^ da[18] ^ da[24] ^ da[30] ^ da[36] ^ da[42]

da[0] represents the least significant bit of the first byte received, that is, the multicast/unicast indicator, and da[47] represents the most significant bit of the last byte received.

If the hash index points to a bit that is set in the Hash register then the frame will be matched according to whether the frame is multicast or unicast.

A multicast match will be signaled if the multicast hash enable bit is set, da[0] is logic 1 and the hash index points to a bit set in the Hash register.

A unicast match will be signaled if the unicast hash enable bit is set, da[0] is logic 0 and the hash index points to a bit set in the Hash register.

To receive all multicast frames, the Hash register should be set with all ones and the multicast hash enable bit should be set in the Network Configuration register.

### 38.6.10 Copy all Frames (Promiscuous Mode)

If the Copy All Frames bit is set in the Network Configuration register then all frames (except those that are too long, too short, have FCS errors or have GRXER asserted during reception) will be copied to memory. Frames with FCS errors will be copied if bit 26 is set in the Network Configuration register.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.6.11 Disable Copy of Pause Frames

Pause frames can be prevented from being written to memory by setting the disable copying of pause frames control bit 23 in the Network Configuration register. When set, pause frames are not copied to memory regardless of the Copy All Frames bit, whether a hash match is found, a type ID match is identified or if a destination address match is found.

### 38.6.12 VLAN Support

The following table describes an Ethernet encoded 802.1Q VLAN tag.

**Table 38-5. 802.1Q VLAN Tag**

| TPID (Tag Protocol Identifier) 16 bits | TCI (Tag Control Information) 16 bits |
|---|---|
| 0x8100 | First 3 bits priority, then CFI bit, last 12 bits VID |

The VLAN tag is inserted at the 13th byte of the frame adding an extra four bytes to the frame. To support these extra four bytes, the GMAC can accept frame lengths up to 1536 bytes by setting bit 8 in the Network Configuration register.

If the VID (VLAN identifier) is null (0x000) this indicates a priority-tagged frame.

The following bits in the receive buffer descriptor status word give information about VLAN tagged frames:-

- Bit 21 set if receive frame is VLAN tagged (i.e., type ID of 0x8100).
- Bit 20 set if receive frame is priority tagged (i.e., type ID of 0x8100 and null VID). (If bit 20 is set, bit 21 will be set also.)
- Bit 19, 18 and 17 set to priority if bit 21 is set.
- Bit 16 set to CFI if bit 21 is set.

The GMAC can be configured to reject all frames except VLAN tagged frames by setting the discard non-VLAN frames bit in the Network Configuration register.

### 38.6.13 Wake on LAN Support

The receive block supports Wake on LAN by detecting the following events on incoming receive frames:

- Magic packet
- Address Resolution Protocol (ARP) request to the device IP address
- Specific address 1 filter match
- Multicast hash filter match

These events can be individually enabled through bits [19:16] of the Wake on LAN register. Also, for Wake on LAN detection to occur, receive enable must be set in the Network Control register, however a receive buffer does not have to be available.

In case of an ARP request, specific address 1 or multicast filter events will occur even if the frame is errored. For magic packet events, the frame must be correctly formed and error free.

A magic packet event is detected if all of the following are true:

- Magic packet events are enabled through bit 16 of the Wake on LAN register
- The frame's destination address matches specific address 1
- The frame is correctly formed with no errors
- The frame contains at least 6 bytes of 0xFF for synchronization
- There are 16 repetitions of the contents of Specific Address 1 register immediately following the synchronization

An ARP request event is detected if all of the following are true:

- ARP request events are enabled through bit 17 of the Wake on LAN register
- Broadcasts are allowed by bit 5 in the Network Configuration register
- The frame has a broadcast destination address (bytes 1 to 6)
- The frame has a type ID field of 0x0806 (bytes 13 and 14)

© 2021 Microchip Technology Inc.
and its subsidiaries

- The frame has an ARP operation field of 0x0001 (bytes 21 and 22)
- The least significant 16 bits of the frame's ARP target protocol address (bytes 41 and 42) match the value programmed in bits[15:0] of the Wake on LAN register

The decoding of the ARP fields adjusts automatically if a VLAN tag is detected within the frame. The reserved value of 0x0000 for the Wake on LAN target address value will not cause an ARP request event, even if matched by the frame.

A specific address 1 filter match event will occur if all of the following are true:

- Specific address 1 events are enabled through bit 18 of the Wake on LAN register
- The frame's destination address matches the value programmed in the Specific Address 1 registers

A multicast filter match event will occur if all of the following are true:

- Multicast hash events are enabled through bit 19 of the Wake on LAN register
- Multicast hash filtering is enabled through bit 6 of the Network Configuration register
- The frame destination address matches against the multicast hash filter
- The frame destination address is not a broadcast

### 38.6.14 IEEE 1588 Support

IEEE 1588 is a standard for precision time synchronization in local area networks. It works with the exchange of special Precision Time Protocol (PTP) frames. The PTP messages can be transported over IEEE 802.3/Ethernet, over Internet Protocol Version 4 or over Internet Protocol Version 6 as described in the annex of IEEE P1588.D2.1.

The GMAC indicates the message time-stamp point (asserted on the start packet delimiter and de-asserted at end of frame) for all frames and the passage of PTP event frames (asserted when a PTP event frame is detected and de-asserted at end of frame).

IEEE 802.1AS is a subset of IEEE 1588. One difference is that IEEE 802.1AS uses the Ethernet multicast address 0180C200000E for sync frame recognition whereas IEEE 1588 does not. GMAC is designed to recognize sync frames with both IEEE 802.1AS and IEEE 1588 addresses and so can support both 1588 and 802.1AS frame recognition simultaneously.

Synchronization between master and slave clocks is a two stage process.

First, the offset between the master and slave clocks is corrected by the master sending a sync frame to the slave with a follow up frame containing the exact time the sync frame was sent. Hardware assist modules at the master and slave side detect exactly when the sync frame was sent by the master and received by the slave. The slave then corrects its clock to match the master clock.

Second, the transmission delay between the master and slave is corrected. The slave sends a delay request frame to the master which sends a delay response frame in reply. Hardware assist modules at the master and slave side detect exactly when the delay request frame was sent by the slave and received by the master. The slave will now have enough information to adjust its clock to account for delay. For example, if the slave was assuming zero delay, the actual delay will be half the difference between the transmit and receive time of the delay request frame (assuming equal transmit and receive times) because the slave clock will be lagging the master clock by the delay time already.

The time-stamp is taken when the message time-stamp point passes the clock time-stamp point. This can generate an interrupt if enabled (GMAC_IER). However, MAC Filtering configuration is needed to actually ‘copy’ the message to memory. For Ethernet, the message time-stamp point is the SFD and the clock time-stamp point is the MII interface. (The IEEE 1588 specification refers to sync and delay_req messages as event messages as these require time-stamping. These events are captured in the registers GMAC_EFTx and GMAC_EFRx, respectively. Follow up, delay response and management messages do not require time-stamping and are referred to as general messages.)

1588 version 2 defines two additional PTP event messages. These are the peer delay request (Pdelay_Req) and peer delay response (Pdelay_Resp) messages. These events are captured in the registers GMAC_PEFTx and GMAC_PEFRx, respectively. These messages are used to calculate the delay on a link. Nodes at both ends of a link send both types of frames (regardless of whether they contain a master or slave clock). The Pdelay_Resp message contains the time at which a Pdelay_Req was received and is itself an event message. The time at which a Pdelay_Resp message is received is returned in a Pdelay_Resp_Follow_Up message.

© 2021 Microchip Technology Inc.
and its subsidiaries

1588 version 2 introduces transparent clocks of which there are two kinds, peer-to-peer (P2P) and end-to-end (E2E). Transparent clocks measure the transit time of event messages through a bridge and amend a correction field within the message to allow for the transit time. P2P transparent clocks additionally correct for the delay in the receive path of the link using the information gathered from the peer delay frames. With P2P transparent clocks delay_req messages are not used to measure link delay. This simplifies the protocol and makes larger systems more stable.

The GMAC recognizes four different encapsulations for PTP event messages:

1. 1588 version 1 (UDP/IPv4 multicast)
2. 1588 version 2 (UDP/IPv4 multicast)
3. 1588 version 2 (UDP/IPv6 multicast)
4. 1588 version 2 (Ethernet multicast)

**Table 38-6. Example of Sync Frame in 1588 Version 1 Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | — |
| SA (Octets 6–11) | — |
| Type (Octets 12–13) | 0800 |
| IP stuff (Octets 14–22) | — |
| UDP (Octet 23) | 11 |
| IP stuff (Octets 24–29) | — |
| IP DA (Octets 30–32) | E00001 |
| IP DA (Octet 33) | 81 or 82 or 83 or 84 |
| Source IP port (Octets 34–35) | — |
| Dest IP port (Octets 36–37) | 013F |
| Other stuff (Octets 38–42) | — |
| Version PTP (Octet 43) | 01 |
| Other stuff (Octets 44–73) | — |
| Control (Octet 74) | 00 |
| Other stuff (Octets 75–168) | — |

**Table 38-7. Example of Delay Request Frame in 1588 Version 1 Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | — |
| SA (Octets 6–11) | — |
| Type (Octets 12–13) | 0800 |
| IP stuff (Octets 14–22) | — |
| UDP (Octet 23) | 11 |
| IP stuff (Octets 24–29) | — |
| IP DA (Octets 30–32) | E00001 |
| IP DA (Octet 33) | 81 or 82 or 83 or 84 |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | |
|---|---|
| **Frame Segment** | **Value** |
| Source IP port (Octets 34–35) | — |
| Dest IP port (Octets 36–37) | 013F |
| Other stuff (Octets 38–42) | — |
| Version PTP (Octet 43) | 01 |
| Other stuff (Octets 44–73) | — |
| Control (Octet 74) | 01 |
| Other stuff (Octets 75–168) | — |

For 1588 version 1 messages, sync and delay request frames are indicated by the GMAC if the frame type field indicates TCP/IP, UDP protocol is indicated, the destination IP address is 224.0.1.129/130/131 or 132, the destination UDP port is 319 and the control field is correct.

The control field is 0x00 for sync frames and 0x01 for delay request frames.

For 1588 version 2 messages, the type of frame is determined by looking at the message type field in the first byte of the PTP frame. Whether a frame is version 1 or version 2 can be determined by looking at the version PTP field in the second byte of both version 1 and version 2 PTP frames.

In version 2 messages sync frames have a message type value of 0x0, delay_req have 0x1, Pdelay_Req have 0x2 and Pdelay_Resp have 0x3.

**Table 38-8. Example of Sync Frame in 1588 Version 2 (UDP/IPv4) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | — |
| SA (Octets 6–11) | — |
| Type (Octets 12–13) | 0800 |
| IP stuff (Octets 14–22) | — |
| UDP (Octet 23) | 11 |
| IP stuff (Octets 24–29) | — |
| IP DA (Octets 30–33) | E0000181 |
| Source IP port (Octets 34–35) | — |
| Dest IP port (Octets 36–37) | 013F |
| Other stuff (Octets 38–41) | — |
| Message type (Octet 42) | 00 |
| Version PTP (Octet 43) | 02 |

**Table 38-9. Example of Pdelay_Req Frame in 1588 Version 2 (UDP/IPv4) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | — |
| SA (Octets 6–11) | — |

| .....continued | |
|---|---|
| **Frame Segment** | **Value** |
| Type (Octets 12–13) | 0800 |
| IP stuff (Octets 14–22) | — |
| UDP (Octet 23) | 11 |
| IP stuff (Octets 24–29) | — |
| IP DA (Octets 30–33) | E000006B |
| Source IP port (Octets 34–35) | — |
| Dest IP port (Octets 36–37) | 013F |
| Other stuff (Octets 38–41) | — |
| Message type (Octet 42) | 02 |
| Version PTP (Octet 43) | 02 |

**Table 38-10. Example of Sync Frame in 1588 Version 2 (UDP/IPv6) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | — |
| SA (Octets 6–11) | — |
| Type (Octets 12–13) | 86dd |
| IP stuff (Octets 14–19) | — |
| UDP (Octet 20) | 11 |
| IP stuff (Octets 21–37) | — |
| IP DA (Octets 38–53) | FF0X00000000018 |
| Source IP port (Octets 54–55) | — |
| Dest IP port (Octets 56–57) | 013F |
| Other stuff (Octets 58–61) | — |
| Message type (Octet 62) | 00 |
| Other stuff (Octets 63–93) | — |
| Version PTP (Octet 94) | 02 |

**Table 38-11. Example of Pdelay_Resp Frame in 1588 Version 2 (UDP/IPv6) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | — |
| SA (Octets 6–11) | — |
| Type (Octets 12–13) | 86dd |
| IP stuff (Octets 14–19) | — |
| UDP (Octet 20) | 11 |

| .....continued | |
|---|---|
| **Frame Segment** | **Value** |
| IP stuff (Octets 21–37) | — |
| IP DA (Octets 38–53) | FF0200000000006B |
| Source IP port (Octets 54–55) | — |
| Dest IP port (Octets 56–57) | 013F |
| Other stuff (Octets 58–61) | — |
| Message type (Octet 62) | 03 |
| Other stuff (Octets 63–93) | — |
| Version PTP (Octet 94) | 02 |

For the multicast address 011B19000000 sync and delay request frames are recognized depending on the message type field, 00 for sync and 01 for delay request.

**Table 38-12. Example of Sync Frame in 1588 Version 2 (Ethernet Multicast) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | 011B19000000 |
| SA (Octets 6–11) | — |
| Type (Octets 12–13) | 88F7 |
| Message type (Octet 14) | 00 |
| Version PTP (Octet 15) | 02 |

Pdelay request frames need a special multicast address so they can pass through ports blocked by the spanning tree protocol. For the multicast address 0180C200000E sync, Pdelay_Req and Pdelay_Resp frames are recognized depending on the message type field, 00 for sync, 02 for pdelay request and 03 for pdelay response.

**Table 38-13. Example of Pdelay_Req Frame in 1588 Version 2 (Ethernet Multicast) Format**

| Frame Segment | Value |
|---|---|
| Preamble/SFD | 55555555555555D5 |
| DA (Octets 0–5) | 0180C200000E |
| SA (Octets 6–11) | — |
| Type (Octets 12–13) | 88F7 |
| Message type (Octet 14) | 00 |
| Version PTP (Octet 15) | 02 |

### 38.6.15 Time Stamp Unit

**Overview**

The TSU consists of a timer and registers to capture the time at which PTP event frames cross the message timestamp point. An interrupt is issued when a capture register is updated.

The 1588 time stamp unit (TSU) is implemented as a 94-bit timer.

- The 48 upper bits [93:46] of the timer count seconds and are accessible in the GMAC 1588 Timer Seconds High Register" (GMAC_TSH) and GMAC 1588 Timer Seconds Low Register (GMAC_TSL).
- The 30 lower bits [45:16] of the timer count nanoseconds and are accessible in the GMAC 1588 Timer Nanoseconds Register (GMAC_TN).
- The lowest 16 bits [15:0] of the timer count sub-nanoseconds.

The 46 lower bits roll over when they have counted to 1s. An interrupt is generated when the seconds increment. The timer increments by a programmable period (to approximately 15.2fs resolution) with each MCK period. The timer value can be read, written and adjusted with 1ns resolution (incremented or decremented) through the APB interface.

**Timer Adjustment**

The amount by which the timer increments each clock cycle is controlled by the Timer Increment register (GMAC_TI). Bits [7:0] are the default increment value in nanoseconds. Additional 16 bits of sub-nanosecond resolution are available using the Timer Increment Sub-Nanoseconds register (GMAC_TISUBN). If the rest of the register is written with zero, the timer increments by the value in [7:0], plus the value of the GMAC_TISUBN for each clock cycle.

The GMAC_TISUBN allows a resolution of approximately 15fs.

Bits [15:8] of the increment register are the alternative increment value in nanoseconds, and bits [23:16] are the number of increments after which the alternative increment value is used. If [23:16] are zero the alternative increment value will never be used.

> Taking the example of 10.2MHz, there are 102 cycles every 10µs or 51 cycles every 5µs. So a timer with a 10.2MHz clock source is constructed by incrementing by 98ns for fifty cycles and then incrementing by 100ns (98ns × 50 + 100ns = 5000ns). This is programmed by writing the value 0x00326462 to the Timer Increment register (GMAC_TI).

> In a second example, a 49.8 MHz clock source requires 20ns for 248 cycles, followed by an increment of 40ns (20ns × 248 + 40ns = 5000ns). This is programmed by writing the value 0x00F82814 to the GMAC_TI register.

The Number of Increments bit field in the GMAC_TI register is 8 bit in size, so frequencies up to 50MHz are supported with 200kHz resolution.

Without the alternative increment field the period of the clock would be limited to an integer number of nanoseconds, resulting in supported clock frequencies of 8, 10, 20, 25, 40, 50, 100, 125, 200 and 250 MHz.

There are eight additional 80-bit registers that capture the time at which PTP event frames are transmitted and received. An interrupt is issued when these registers are updated. The TSU timer count value can be compared to a programmable comparison value. For the comparison, the 48 bits of the seconds value and the upper 22 bits of the nanoseconds value are used. A signal (GTSUCOMP) is output from the core to indicate when the TSU timer count value is equal to the comparison value stored in the TSU timer comparison value registers (GMAC_NSC, GMAC_SCL, and GMAC_SCH). The GTSUCOMP signal can be routed to the Timer peripheral to automatically toggle pin TIOA11/PD21. This can be used as the reference clock for an external PLL to regenerate the audio clock in Ethernet AVB.An interrupt can also be generated (if enabled) when the TSU timer count value and comparison value are equal, mapped to bit 29 of the interrupt status register.

### 38.6.16 MAC 802.3 Pause Frame Support

**Note:** Refer to the Clause 31, and Annex 31A and 31B of the IEEE standard 802.3 for a full description of MAC 802.3 pause operation.

The following table shows the start of a MAC 802.3 pause frame.

**Table 38-14. Start of an 802.3 Pause Frame**

| Address | | Type (MAC Control Frame) | Pause | |
|---|---|---|---|---|
| Destination | Source | | Opcode | Time |
| 0x0180C2000001 | 6 bytes | 0x8808 | 0x0001 | 2 bytes |

The GMAC supports both hardware controlled pause of the transmitter, upon reception of a pause frame, and hardware generated pause frame transmission.

#### 38.6.16.1 802.3 Pause Frame Reception

Bit 13 of the Network Configuration register is the pause enable control for reception. If this bit is set, transmission will pause if a non zero pause quantum frame is received.

If a valid pause frame is received then the Pause Time register is updated with the new frame's pause time, regardless of whether a previous pause frame is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled (bit 12 and bit 13 of the Interrupt Mask register). Pause frames received with non zero quantum are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quantum are indicated on bit 13 of the Interrupt Status register.

Once the Pause Time register is loaded and the frame currently being transmitted has been sent, no new frames are transmitted until the pause time reaches zero. The loading of a new pause time, and hence the pausing of transmission, only occurs when the GMAC is configured for full duplex operation. If the GMAC is configured for half duplex there will be no transmission pause, but the pause frame received interrupt will still be triggered. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address register 1 or if it matches the reserved address of 0x0180C2000001. It must also have the MAC control frame type ID of 0x8808 and have the pause opcode of 0x0001.

Pause frames that have frame check sequence (FCS) or other errors will be treated as invalid and will be discarded. 802.3 Pause frames that are received after Priority-based Flow Control (PFC) has been negotiated will also be discarded. Valid pause frames received will increment the pause frames received statistic register.

The pause time register decrements every 512 bit times once transmission has stopped. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Pause Time register to decrement every GTXCK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Pause Time register decrements to zero (assuming it has been enabled by bit 13 in the Interrupt Mask register). This interrupt is also set when a zero quantum pause frame is received.

#### 38.6.16.2 802.3 Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit pause frame bits of the Network Control register. If either bit 11 or bit 12 of the Network Control register is written with logic 1, an 802.3 pause frame will be transmitted, providing full duplex is selected in the Network Configuration register and the transmit block is enabled in the Network Control register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise the following:

- A destination address of 01-80-C2-00-00-01
- A source address taken from Specific Address register 1
- A type ID of 88-08 (MAC control frame)
- A pause opcode of 00-01
- A pause quantum register
- Fill of 00 to take the frame to minimum frame length
- Valid FCS

The pause quantum used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 11 is written with a '1', the pause quantum will be taken from the Transmit Pause Quantum register. The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as default.
- If bit 12 is written with a '1', the pause quantum will be zero.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only statistics register that will be incremented will be the Pause Frames Transmitted register.

Pause frames can also be transmitted by the MAC using normal frame transmission methods.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.6.17 MAC PFC Priority-based Pause Frame Support

**Note:** Refer to the 802.1Qbb standard for a full description of priority-based pause operation.

The following table shows the start of a Priority-based Flow Control (PFC) pause frame.

**Table 38-15. Start of a PFC Pause Frame**

| Address | | Type (Mac Control Frame) | Pause Opcode | Priority Enable Vector | Pause Time |
|---|---|---|---|---|---|
| Destination | Source | | | | |
| 0x0180C2000001 | 6 bytes | 0x8808 | 0x1001 | 2 bytes | 8 × 2 bytes |

The GMAC supports PFC priority-based pause transmission and reception. Before PFC pause frames can be received, bit 16 of the Network Control register must be set.

#### 38.6.17.1 PFC Pause Frame Reception

The ability to receive and decode priority-based pause frames is enabled by setting bit 16 of the Network Control register. When this bit is set, the GMAC will match either classic 802.3 pause frames or PFC priority-based pause frames. Once a priority-based pause frame has been received and matched, then from that moment on the GMAC will only match on priority-based pause frames (this is an 802.1Qbb requirement, known as PFC negotiation). Once priority-based pause has been negotiated, any received 802.3x format pause frames will not be acted upon.

If a valid priority-based pause frame is received then the GMAC will decode the frame and determine which, if any, of the eight priorities require to be paused. Up to eight Pause Time registers are then updated with the eight pause times extracted from the frame regardless of whether a previous pause operation is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled (bit 12 and bit 13 of the Interrupt Mask register). Pause frames received with non zero quantum are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quantum are indicated on bit 13 of the Interrupt Status register. The loading of a new pause time only occurs when the GMAC is configured for full duplex operation. If the GMAC is configured for half duplex, the pause time counters will not be loaded, but the pause frame received interrupt will still be triggered. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address register 1 or if it matches the reserved address of 0x0180C2000001. It must also have the MAC control frame type ID of 0x8808 and have the pause opcode of 0x0101.

Pause frames that have frame check sequence (FCS) or other errors will be treated as invalid and will be discarded. Valid pause frames received will increment the Pause Frames Received Statistic register.

The Pause Time registers decrement every 512 bit times immediately following the PFC frame reception. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Pause Time register to decrement every GRXCK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Pause Time register decrements to zero (assuming it has been enabled by bit 13 in the Interrupt Mask register). This interrupt is also set when a zero quantum pause frame is received.

#### 38.6.17.2 PFC Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit priority-based pause frame bit of the Network Control register. If bit 17 of the Network Control register is written with logic 1, a PFC pause frame will be transmitted providing full duplex is selected in the Network Configuration register and the transmit block is enabled in the Network Control register. When bit 17 of the Network Control register is set, the fields of the priority-based pause frame will be built using the values stored in the Transmit PFC Pause register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise the following:

- A destination address of 01-80-C2-00-00-01
- A source address taken from Specific Address register 1
- A type ID of 88-08 (MAC control frame)
- A pause opcode of 01-01

© 2021 Microchip Technology Inc.
and its subsidiaries

- A priority enable vector taken from Transmit PFC Pause register
- 8 pause quantum registers
- Fill of 00 to take the frame to minimum frame length
- Valid FCS

The pause quantum registers used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 17 of the Network Control register is written with a one, then the priority enable vector of the priority-based pause frame will be set equal to the value stored in the Transmit PFC Pause register [7:0]. For each entry equal to zero in the Transmit PFC Pause register [15:8], the pause quantum field of the pause frame associated with that entry will be taken from the transmit pause quantum register. For each entry equal to one in the Transmit PFC Pause register [15:8], the pause quantum associated with that entry will be zero.
- The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as default.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only statistics register that will be incremented will be the Pause Frames Transmitted register.

PFC Pause frames can also be transmitted by the MAC using normal frame transmission methods.

### 38.6.18 Energy Efficient Ethernet Support

**Features**

- Energy Efficient Ethernet according to IEEE 802.3az
- A system's transmit path can enter a low power mode if there is nothing to transmit.
- A PHY can detect whether its link partner's transmit path is in low power mode, and configure its own receive path to enter low power mode.
- Link remains up during lower power mode and no frames are dropped.
- Asymmetric, one direction can be in low power mode while the other is transmitting normally.
- LPI (Low Power Idle) signaling is used to control entry and exit to and from low power modes.
  **Note:** LPI signaling can only take place if both sides have indicated support for it through auto-negotiation.

**Operation**

- Low power control is done at the MII (reconciliation sublayer).
- As an architectural convenience in writing the 802.3az it is assumed that transmission is deferred by asserting carrier sense - in practice it will not be done this way. This system will know when it has nothing to transmit and only enter low power mode when it is not transmitting.
- LPI should not be requested unless the link has been up for at least one second.
- LPI is signaled on the MII transmit path by asserting 0x01 on txd with tx_en low and tx_er high.
- A PHY on seeing LPI requested on the MII will send the sleep signal before going quiet. After going quiet it will periodically emit refresh signals.
- The sleep, quiet and refresh periods are defined in 802.3az, Table 78-2.
- LPI mode ends by transmitting normal idle for the wake time. There is a default time for this but it can be adjusted in software using the Link Layer Discovery Protocol (LLDP) described in 802.3az, Clause 79.
- LPI is indicated at the receive side when sleep and refresh signaling has been detected.

### 38.6.19 802.1Qav Support - Credit-based Shaping

A credit-based shaping algorithm is available on the two highest priority queues and is defined in the standard 802.1Qav: Forwarding and Queuing Enhancements for Time-Sensitive Streams. This allows traffic on these queues to be limited and to allow other queues to transmit.

Traffic shaping is enabled via the CBS (Credit Based Shaping) Control register. This enables a counter which stores the amount of transmit 'credit', measured in bytes that a particular queue has. A queue may only transmit if it has non-negative credit. If a queue has data to send, but is held off from doing as another queue is transmitting, then credit will accumulate in the credit counter at the rate defined in the IdleSlope register (GMAC_CBSISQx) for that queue.

**portTransmitRate** is the transmission rate, in bits per second, that the underlying MAC service that supports transmission through the Port provides. The value of this parameter is determined by the operation of the MAC.

IdleSlope is the rate of change of increasing credit when waiting to transmit and must be less than the value of the portTransmitRate.

**IdleSlope** is the rate of change of credit when waiting to transmit and must be less than the value of the portTransmitRate.

The max value of IdleSlope (or sendSlope) is (portTransmitRate / bits_per_MII_Clock).

In case of 100 Mbps, maximum IdleSlope = (100 Mbps / 4) = 0x17D7840.

When this queue is transmitting the credit counter is decremented at the rate of sendSlope which is defined as (portTransmitRate - IdleSlope). A queue can accumulate negative credit when transmitting which will hold off any other transfers from that queue until credit returns to a non-negative value. No transfers are halted when a queue's credit becomes negative; it will accumulate negative credit until the transfer completes.

The highest priority queue always has priority regardless of which queue has the most credit.

### 38.6.20 LPI Operation in the EMAC

It is best to use firmware to control LPI. LPI operation happens at the system level. Firmware gives maximum control and flexibility of operation. LPI operation is straightforward and firmware should be capable of responding within the required timeframes.

Autonegotiation:

1. Indicate EEE capability using next page autonegotiation.

For the transmit path:

1. If the link has been up for 1 second and there is nothing being transmitted, write to the TXLPIEN bit in the Network Control register.
2. Wake up by clearing the TXLPIEN bit in the Network Control register.

For the receive path:

1. Enable RXLPISBC bit in GMAC_IER. The bit RXLPIS is set in Network Status Register triggering an interrupt.
2. Wait for an interrupt to indicate that LPI has been received.
3. Disable relevant parts of the receive path if desired.
4. The RXLPIS bit in Network Status Register gets cleared to indicate that regular idle has been received. This triggers an interrupt.
5. Re-enable the receive path.

### 38.6.21 PHY Interface

Different PHY interfaces are supported by the Ethernet MAC:

- MII
- RMII

The MII interface is provided for 10/100 operation and uses txd[3:0] and rxd[3:0]. The RMII interface is provided for 10/100 operation and uses txd[1:0] and rxd[1:0].

### 38.6.22 10/100 Operation

The 10/100 Mbps speed bit in the Network Configuration register is used to select between 10 Mbps and 100 Mbps.

### 38.6.23 Jumbo Frames

The jumbo frames enable bit in the Network Configuration register allows the GMAC, in its default configuration, to receive jumbo frames up to 10240 bytes in size. This operation does not form part of the IEEE 802.3 specification and is normally disabled. When jumbo frames are enabled, frames received with a frame size greater than 10240 bytes are discarded.

© 2021 Microchip Technology Inc. and its subsidiaries

## 38.7 Programming Interface

### 38.7.1 Initialization

#### 38.7.1.1 Configuration

Initialization of the GMAC configuration (e.g., loop back mode, frequency ratios) must be done while the transmit and receive circuits are disabled. See the description of the Network Control register and Network Configuration register earlier in this document.

To change loop back mode, the following sequence of operations must be followed:

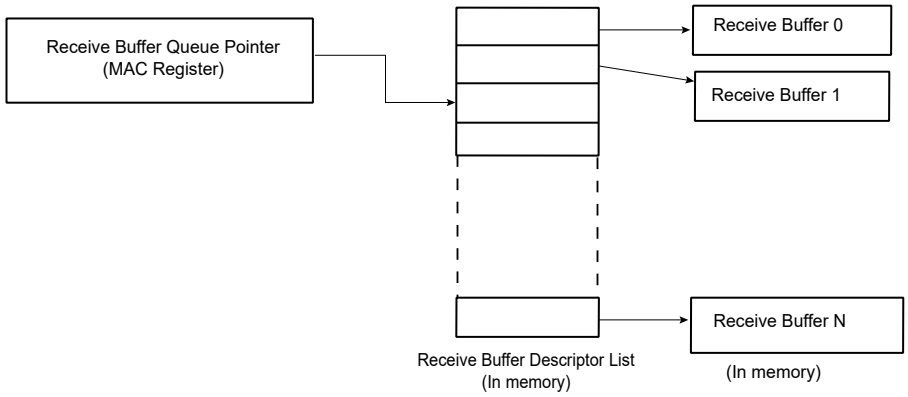
1. Write to Network Control register to disable transmit and receive circuits.
2. Write to Network Control register to change loop back mode.
3. Write to Network Control register to re-enable transmit or receive circuits.
   Note: These writes to the Network Control register cannot be combined in any way.

#### 38.7.1.2 Receive Buffer List

Receive data is written to areas of data (i.e., buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (receive buffer queue) is a sequence of descriptor entries as defined in the table Receive Buffer Descriptor Entry.

The Receive Buffer Queue Pointer register points to this data structure.

**Figure 38-3. Receive Buffer List**




To create the list of buffers:

1. Allocate a number (N) of buffers of X bytes in system memory, where X is the DMA buffer length programmed in the DMA Configuration register.
2. Allocate an area 8N bytes for the receive buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GMAC, i.e., bit 0 of word 0 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 1 in word 0 set to 1).
4. Write address of receive buffer descriptor list and control information to GMAC register receive buffer queue pointer
5. The receive circuits can then be enabled by writing to the address recognition registers and the Network Control register.

**Note:** The queue pointers must be initialized and point to USED descriptors for all queues including those not intended for use.

#### 38.7.1.3 Transmit Buffer List

Transmit data is read from areas of data (the buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (Transmit Buffer Queue) is a sequence of descriptor entries as defined in the table Transmit Buffer Descriptor Entry.

The Transmit Buffer Queue Pointer register points to this data structure.

To create this list of buffers:

1. Allocate a number (N) of buffers of between 1 and 2047 bytes of data to be transmitted in system memory. Up to 128 buffers per frame are allowed.
2. Allocate an area 8N bytes for the transmit buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GMAC, i.e., bit 31 of word 1 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 30 in word 1 set to 1).
4. Write address of transmit buffer descriptor list and control information to GMAC register transmit buffer queue pointer.
5. The transmit circuits can then be enabled by writing to the Network Control register.

**Note:** The queue pointers must be initialized and point to USED descriptors for all queues including those not intended for use.

#### 38.7.1.4 Address Matching

The GMAC Hash register pair and the four Specific Address register pairs must be written with the required values. Each register pair comprises of a bottom register and top register, with the bottom register being written first. The address matching is disabled for a particular register pair after the bottom register has been written and re-enabled when the top register is written. Each register pair may be written at any time, regardless of whether the receive circuits are enabled or disabled.

As an example, to set Specific Address register 1 to recognize destination address 21:43:65:87:A9:CB, the following values are written to Specific Address register 1 bottom and Specific Address register 1 top:

- Specific Address register 1 bottom bits 31:0 (0x98): 0x8765_4321.
- Specific Address register 1 top bits 31:0 (0x9C): 0x0000_CBA9.
  **Note:** The address matching is the first level of filtering. If there is a match, the screeners are the next level of filtering for routing the data to the appropriate queue. See Priority Queueing in the DMA for more details.

#### 38.7.1.5 PHY Maintenance

The PHY Maintenance register is implemented as a shift register. Writing to the register starts a shift operation which is signalled as complete when bit two is set in the Network Status register (about 2000 MCK cycles later when bits 18:16 are set to 010 in the Network Configuration register). An interrupt is generated as this bit is set.

During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each Management Data Clock (MDC) cycle. This causes the transmission of a PHY management frame on MDIO. See section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation will return the current contents of the shift register. At the end of the management operation the bits will have shifted back to their original locations. For a read operation the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The Management Data Clock (MDC) should not toggle faster than 2.5 MHz (minimum period of 400 ns), as defined by the IEEE 802.3 standard. MDC is generated by dividing down MCK. Three bits in the Network Configuration register determine by how much MCK should be divided to produce MDC.

#### 38.7.1.6 Interrupts

There are 18 interrupt conditions that are detected within the GMAC. The conditions are ORed to make multiple interrupts. Depending on the overall system design this may be passed through a further level of interrupt collection (interrupt controller). On receipt of the interrupt signal, the CPU enters the interrupt handler. Refer to the device interrupt controller documentation to identify that it is the GMAC that is generating the interrupt. To ascertain which interrupt, read the Interrupt Status register. Note that in the default configuration this register will clear itself after being read, though this may be configured to be write-one-to-clear if desired.

At reset all interrupts are disabled. To enable an interrupt, write to Interrupt Enable register with the pertinent interrupt bit set to 1. To disable an interrupt, write to Interrupt Disable register with the pertinent interrupt bit set to 1. To check whether an interrupt is enabled or disabled, read Interrupt Mask register. If the bit is set to 1, the interrupt is disabled.

#### 38.7.1.7 Transmitting Frames

The procedure to set up a frame for transmission is the following:

1. Enable transmit in the Network Control register.
2. Allocate an area of system memory for transmit data. This does not have to be contiguous, varying byte lengths can be used if they conclude on byte borders.

© 2021 Microchip Technology Inc. and its subsidiaries

3. Set-up the transmit buffer list by writing buffer addresses to word zero of the transmit buffer descriptor entries and control and length to word one.
4. Write data for transmission into the buffers pointed to by the descriptors.
5. Write the address of the first buffer descriptor to transmit buffer descriptor queue pointer.
6. Enable appropriate interrupts.
7. Write to the transmit start bit (TSTART) in the Network Control register.

#### 38.7.1.8 Receiving Frames

When a frame is received and the receive circuits are enabled, the GMAC checks the address and, in the following cases, the frame is written to system memory:

- If it matches one of the four Specific Address registers.
- If it matches one of the four type ID registers.
- If it matches the hash address function.
- If it is a broadcast address (0xFFFFFFFFFFFF) and broadcasts are allowed.
- If the GMAC is configured to “copy all frames”.

The register receive buffer queue pointer points to the next entry in the receive buffer descriptor list and the GMAC uses this as the address in system memory to write the frame to.

Once the frame has been completely and successfully received and written to system memory, the GMAC then updates the receive buffer descriptor entry (see Receive Buffer Descriptor Entry) with the reason for the address match and marks the area as being owned by software. Once this is complete, a receive complete interrupt is set. Software is then responsible for copying the data to the application area and releasing the buffer (by writing the ownership bit back to 0).

If the GMAC is unable to write the data at a rate to match the incoming frame, then a receive overrun interrupt is set. If there is no receive buffer available, i.e., the next buffer is still owned by software, a receive buffer not available interrupt is set. If the frame is not successfully received, a statistics register is incremented and the frame is discarded without informing software.

### 38.7.2 Statistics Registers

Statistics registers are described in the User Interface beginning with GMAC Octets Transmitted Low Register and ending with GMAC UDP Checksum Errors Register.

The statistics register block begins at 0x100 and runs to 0x1B0, and comprises the registers listed below.

| | |
|---|---|
| Octets Transmitted Low Register | Broadcast Frames Received Register |
| Octets Transmitted High Register | Multicast Frames Received Register |
| Frames Transmitted Register | Pause Frames Received Register |
| Broadcast Frames Transmitted Register | 64 Byte Frames Received Register |
| Multicast Frames Transmitted Register | 65 to 127 Byte Frames Received Register |
| Pause Frames Transmitted Register | 128 to 255 Byte Frames Received Register |
| 64 Byte Frames Transmitted Register | 256 to 511 Byte Frames Received Register |
| 65 to 127 Byte Frames Transmitted Register | 512 to 1023 Byte Frames Received Register |
| 128 to 255 Byte Frames Transmitted Register | 1024 to 1518 Byte Frames Received Register |
| 256 to 511 Byte Frames Transmitted Register | 1519 to Maximum Byte Frames Received Register |
| 512 to 1023 Byte Frames Transmitted Register | Undersize Frames Received Register |
| 1024 to 1518 Byte Frames Transmitted Register | Oversize Frames Received Register |
| Greater Than 1518 Byte Frames Transmitted Register | Jabbers Received Register |
| Transmit Underruns Register | Frame Check Sequence Errors Register |

© 2021 Microchip Technology Inc.
and its subsidiaries

| | |
|---|---|
| Single Collision Frames Register | Length Field Frame Errors Register |
| Multiple Collision Frames Register | Receive Symbol Errors Register |
| Excessive Collisions Register | Alignment Errors Register |
| Late Collisions Register | Receive Resource Errors Register |
| Deferred Transmission Frames Register | Receive Overrun Register |
| Carrier Sense Errors Register | IP Header Checksum Errors Register |
| Octets Received Low Register | TCP Checksum Errors Register |
| Octets Received High Register | UDP Checksum Errors Register |
| Frames Received Register | |

These registers reset to zero on a read and stick at all ones when they count to their maximum value. They should be read frequently enough to prevent loss of data.

The receive statistics registers are only incremented when the receive enable bit (RXEN) is set in the Network Control register.

Once a statistics register has been read, it is automatically cleared. When reading the Octets Transmitted and Octets Received registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

© 2021 Microchip Technology Inc.
and its subsidiaries

## 38.8 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | GMAC_NCR | 7:0 | WESTAT | INCSTAT | CLRSTAT | MPE | TXEN | RXEN | LBL | |
| | | 15:8 | SRTSM | | | TXZQPF | TXPF | THALT | TSTART | BP |
| | | 23:16 | | | | | | FNP | TXPBPF | ENPBPR |
| | | 31:24 | | | | | | | | |
| 0x04 | GMAC_NCFGR | 7:0 | UNIHEN | MTIHEN | NBC | CAF | JFRAME | DNVLAN | FD | SPD |
| | | 15:8 | RXBUFO[1:0] | | PEN | RTY | | | | MAXFS |
| | | 23:16 | DCPF | DBW[1:0] | | CLK[2:0] | | | RFCS | LFERD |
| | | 31:24 | | IRXER | RXBP | IPGSEN | | IRXFCS | EFRHD | RXCOEN |
| 0x08 | GMAC_NSR | 7:0 | | | | | | IDLE | MDIO | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | GMAC_UR | 7:0 | | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | GMAC_DCFGR | 7:0 | ESPA | ESMA | | FBLDO[4:0] | | | | |
| | | 15:8 | | | | | TXCOEN | TXPBMS | RXBMS[1:0] | |
| | | 23:16 | DRBS[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | DDRP |
| 0x14 | GMAC_TSR | 7:0 | | | TXCOMP | TFC | TXGO | RLE | COL | UBR |
| | | 15:8 | | | | | | | | HRESP |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | GMAC_RBQB | 7:0 | ADDR[5:0] | | | | | | | |
| | | 15:8 | ADDR[13:6] | | | | | | | |
| | | 23:16 | ADDR[21:14] | | | | | | | |
| | | 31:24 | ADDR[29:22] | | | | | | | |
| 0x1C | GMAC_TBQB | 7:0 | ADDR[5:0] | | | | | | | |
| | | 15:8 | ADDR[13:6] | | | | | | | |
| | | 23:16 | ADDR[21:14] | | | | | | | |
| | | 31:24 | ADDR[29:22] | | | | | | | |
| 0x20 | GMAC_RSR | 7:0 | | | | | HNO | RXOVR | REC | BNA |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | GMAC_ISR | 7:0 | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| | | 15:8 | | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| | | 23:16 | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| | | 31:24 | | | TSUTIMCMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| 0x28 | GMAC_IER | 7:0 | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| | | 15:8 | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| | | 23:16 | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| | | 31:24 | | | TSUTIMCMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| 0x2C | GMAC_IDR | 7:0 | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| | | 15:8 | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| | | 23:16 | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| | | 31:24 | | | TSUTIMCMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| 0x30 | GMAC_IMR | 7:0 | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| | | 15:8 | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| | | 23:16 | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| | | 31:24 | | | TSUTIMCMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| 0x34 | GMAC_MAN | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | PHYA[0] | REGA[4:0] | | | | | WTN[1:0] | |
| | | 31:24 | WZO | CLTTO | OP[1:0] | | PHYA[4:1] | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x38 | GMAC_RPQ | 7:0 | RPQ[7:0] | | | | | | | |
| | | 15:8 | RPQ[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | GMAC_TPQ | 7:0 | TPQ[7:0] | | | | | | | |
| | | 15:8 | TPQ[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | GMAC_TPSF | 7:0 | TPB1ADR[7:0] | | | | | | | |
| | | 15:8 | | | | | TPB1ADR[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENTXP | | | | | | | |
| 0x44 | GMAC_RPSF | 7:0 | RPB1ADR[7:0] | | | | | | | |
| | | 15:8 | | | | | RPB1ADR[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENRXP | | | | | | | |
| 0x48 | GMAC_RJFML | 7:0 | FML[7:0] | | | | | | | |
| | | 15:8 | | | FML[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | GMAC_HRB | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x84 | GMAC_HRT | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x88 | GMAC_SAB1 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x8C | GMAC_SAT1 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x90 | GMAC_SAB2 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x94 | GMAC_SAT2 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | GMAC_SAB3 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x9C | GMAC_SAT3 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA0 | GMAC_SAB4 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xA4 | GMAC_SAT4 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA8 | GMAC_TIDM1 | 7:0 | TID[7:0] | | | | | | | |
| | | 15:8 | TID[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENIDn | | | | | | | |
| 0xAC | GMAC_TIDM2 | 7:0 | TID[7:0] | | | | | | | |
| | | 15:8 | TID[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENIDn | | | | | | | |
| 0xB0 | GMAC_TIDM3 | 7:0 | TID[7:0] | | | | | | | |
| | | 15:8 | TID[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENIDn | | | | | | | |
| 0xB4 | GMAC_TIDM4 | 7:0 | TID[7:0] | | | | | | | |
| | | 15:8 | TID[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ENIDn | | | | | | | |
| 0xB8 | GMAC_WOL | 7:0 | IP[7:0] | | | | | | | |
| | | 15:8 | IP[15:8] | | | | | | | |
| | | 23:16 | | | | | MTI | SA1 | ARP | MAG |
| | | 31:24 | | | | | | | | |
| 0xBC | GMAC_IPGS | 7:0 | FL[7:0] | | | | | | | |
| | | 15:8 | FL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC0 | GMAC_SVLAN | 7:0 | VLAN_TYPE[7:0] | | | | | | | |
| | | 15:8 | VLAN_TYPE[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | ESVLAN | | | | | | | |
| 0xC4 | GMAC_TPFCP | 7:0 | PEV[7:0] | | | | | | | |
| | | 15:8 | PQ[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC8 | GMAC_SAMB1 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0xCC | GMAC_SAMT1 | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD0 ... 0xDB | Reserved | | | | | | | | | |
| 0xDC | GMAC_NSC | 7:0 | NANOSEC[7:0] | | | | | | | |
| | | 15:8 | NANOSEC[15:8] | | | | | | | |
| | | 23:16 | | | NANOSEC[21:16] | | | | | |
| | | 31:24 | | | | | | | | |
| 0xE0 | GMAC_SCL | 7:0 | SEC[7:0] | | | | | | | |
| | | 15:8 | SEC[15:8] | | | | | | | |
| | | 23:16 | SEC[23:16] | | | | | | | |
| | | 31:24 | SEC[31:24] | | | | | | | |
| 0xE4 | GMAC_SCH | 7:0 | SEC[7:0] | | | | | | | |
| | | 15:8 | SEC[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE8 | GMAC_EFTSH | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xEC | GMAC_EFRSH | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF0 | GMAC_PEFTSH | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF4 | GMAC_PEFRSH | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xF8 ... 0xFF | Reserved | | | | | | | | | |
| 0x0100 | GMAC_OTLO | 7:0 | TXO[7:0] | | | | | | | |
| | | 15:8 | TXO[15:8] | | | | | | | |
| | | 23:16 | TXO[23:16] | | | | | | | |
| | | 31:24 | TXO[31:24] | | | | | | | |
| 0x0104 | GMAC_OTHI | 7:0 | TXO[7:0] | | | | | | | |
| | | 15:8 | TXO[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0108 | GMAC_FT | 7:0 | FTX[7:0] | | | | | | | |
| | | 15:8 | FTX[15:8] | | | | | | | |
| | | 23:16 | FTX[23:16] | | | | | | | |
| | | 31:24 | FTX[31:24] | | | | | | | |
| 0x010C | GMAC_BCFT | 7:0 | BFTX[7:0] | | | | | | | |
| | | 15:8 | BFTX[15:8] | | | | | | | |
| | | 23:16 | BFTX[23:16] | | | | | | | |
| | | 31:24 | BFTX[31:24] | | | | | | | |
| 0x0110 | GMAC_MFT | 7:0 | MFTX[7:0] | | | | | | | |
| | | 15:8 | MFTX[15:8] | | | | | | | |
| | | 23:16 | MFTX[23:16] | | | | | | | |
| | | 31:24 | MFTX[31:24] | | | | | | | |
| 0x0114 | GMAC_PFT | 7:0 | PFTX[7:0] | | | | | | | |
| | | 15:8 | PFTX[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0118 | GMAC_BFT64 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x011C | GMAC_TBFT127 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0120 | GMAC_TBFT255 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0124 | GMAC_TBFT511 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0128 | GMAC_TBFT1023 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x012C | GMAC_TBFT1518 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0130 | GMAC_GTBFT1518 | 7:0 | NFTX[7:0] | | | | | | | |
| | | 15:8 | NFTX[15:8] | | | | | | | |
| | | 23:16 | NFTX[23:16] | | | | | | | |
| | | 31:24 | NFTX[31:24] | | | | | | | |
| 0x0134 | GMAC_TUR | 7:0 | TXUNR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | TXUNR[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0138 | GMAC_SCF | 7:0 | SCOL[7:0] | | | | | | | |
| | | 15:8 | SCOL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | SCOL[17:16] | |
| | | 31:24 | | | | | | | | |
| 0x013C | GMAC_MCF | 7:0 | MCOL[7:0] | | | | | | | |
| | | 15:8 | MCOL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | MCOL[17:16] | |
| | | 31:24 | | | | | | | | |
| 0x0140 | GMAC_EC | 7:0 | XCOL[7:0] | | | | | | | |
| | | 15:8 | | | | | | | XCOL[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0144 | GMAC_LC | 7:0 | LCOL[7:0] | | | | | | | |
| | | 15:8 | | | | | | | LCOL[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0148 | GMAC_DTF | 7:0 | DEFT[7:0] | | | | | | | |
| | | 15:8 | DEFT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | DEFT[17:16] | |
| | | 31:24 | | | | | | | | |
| 0x014C | GMAC_CSE | 7:0 | CSR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | CSR[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0150 | GMAC_ORLO | 7:0 | RXO[7:0] | | | | | | | |
| | | 15:8 | RXO[15:8] | | | | | | | |
| | | 23:16 | RXO[23:16] | | | | | | | |
| | | 31:24 | RXO[31:24] | | | | | | | |
| 0x0154 | GMAC_ORHI | 7:0 | RXO[7:0] | | | | | | | |
| | | 15:8 | RXO[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0158 | GMAC_FR | 7:0 | FRX[7:0] | | | | | | | |
| | | 15:8 | FRX[15:8] | | | | | | | |
| | | 23:16 | FRX[23:16] | | | | | | | |
| | | 31:24 | FRX[31:24] | | | | | | | |
| 0x015C | GMAC_BCFR | 7:0 | BFRX[7:0] | | | | | | | |
| | | 15:8 | BFRX[15:8] | | | | | | | |
| | | 23:16 | BFRX[23:16] | | | | | | | |
| | | 31:24 | BFRX[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0160 | GMAC_MFR | 7:0 | MFRX[7:0] | | | | | | | |
| | | 15:8 | MFRX[15:8] | | | | | | | |
| | | 23:16 | MFRX[23:16] | | | | | | | |
| | | 31:24 | MFRX[31:24] | | | | | | | |
| 0x0164 | GMAC_PFR | 7:0 | PFRX[7:0] | | | | | | | |
| | | 15:8 | PFRX[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0168 | GMAC_BFR64 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x016C | GMAC_TBFR127 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0170 | GMAC_TBFR255 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0174 | GMAC_TBFR511 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0178 | GMAC_TBFR1023 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x017C | GMAC_TBFR1518 | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0180 | GMAC_TMXBFR | 7:0 | NFRX[7:0] | | | | | | | |
| | | 15:8 | NFRX[15:8] | | | | | | | |
| | | 23:16 | NFRX[23:16] | | | | | | | |
| | | 31:24 | NFRX[31:24] | | | | | | | |
| 0x0184 | GMAC_UFR | 7:0 | UFRX[7:0] | | | | | | | |
| | | 15:8 | | | | | | | UFRX[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0188 | GMAC_OFR | 7:0 | OFRX[7:0] | | | | | | | |
| | | 15:8 | | | | | | | OFRX[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x018C | GMAC_JR | 7:0 | JRX[7:0] | | | | | | | |
| | | 15:8 | | | | | | | JRX[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0190 | GMAC_FCSE | 7:0 | FCKR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | FCKR[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0194 | GMAC_LFFE | 7:0 | LFER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | LFER[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0198 | GMAC_RSE | 7:0 | RXSE[7:0] | | | | | | | |
| | | 15:8 | | | | | | | RXSE[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x019C | GMAC_AE | 7:0 | AER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | AER[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A0 | GMAC_RRE | 7:0 | RXRER[7:0] | | | | | | | |
| | | 15:8 | RXRER[15:8] | | | | | | | |
| | | 23:16 | | | | | | | RXRER[17:16] | |
| | | 31:24 | | | | | | | | |
| 0x01A4 | GMAC_ROE | 7:0 | RXOVR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | RXOVR[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A8 | GMAC_IHCE | 7:0 | HCKER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01AC | GMAC_TCE | 7:0 | TCKER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B0 | GMAC_UCE | 7:0 | UCKER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B4 ... 0x01BB | Reserved | | | | | | | | | |
| 0x01BC | GMAC_TISUBN | 7:0 | LSBTIR[7:0] | | | | | | | |
| | | 15:8 | LSBTIR[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01C0 | GMAC_TSH | 7:0 | TCS[7:0] | | | | | | | |
| | | 15:8 | TCS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01C4 ... 0x01CF | Reserved | | | | | | | | | |
| 0x01D0 | GMAC_TSL | 7:0 | TCS[7:0] | | | | | | | |
| | | 15:8 | TCS[15:8] | | | | | | | |
| | | 23:16 | TCS[23:16] | | | | | | | |
| | | 31:24 | TCS[31:24] | | | | | | | |
| 0x01D4 | GMAC_TN | 7:0 | TNS[7:0] | | | | | | | |
| | | 15:8 | TNS[15:8] | | | | | | | |
| | | 23:16 | TNS[23:16] | | | | | | | |
| | | 31:24 | | | TNS[29:24] | | | | | |
| 0x01D8 | GMAC_TA | 7:0 | ITDT[7:0] | | | | | | | |
| | | 15:8 | ITDT[15:8] | | | | | | | |
| | | 23:16 | ITDT[23:16] | | | | | | | |
| | | 31:24 | ADJ | | ITDT[29:24] | | | | | |
| 0x01DC | GMAC_TI | 7:0 | CNS[7:0] | | | | | | | |
| | | 15:8 | ACNS[7:0] | | | | | | | |
| | | 23:16 | NIT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01E0 | GMAC_EFTSL | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | RUD[31:24] | | | | | | | |
| 0x01E4 | GMAC_EFTN | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | | | RUD[29:24] | | | | | |
| 0x01E8 | GMAC_EFRSL | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | RUD[31:24] | | | | | | | |
| 0x01EC | GMAC_EFRN | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | | | RUD[29:24] | | | | | |
| 0x01F0 | GMAC_PEFTSL | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | RUD[31:24] | | | | | | | |
| 0x01F4 | GMAC_PEFTN | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | | | RUD[29:24] | | | | | |
| 0x01F8 | GMAC_PEFRSL | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | RUD[31:24] | | | | | | | |
| 0x01FC | GMAC_PEFRN | 7:0 | RUD[7:0] | | | | | | | |
| | | 15:8 | RUD[15:8] | | | | | | | |
| | | 23:16 | RUD[23:16] | | | | | | | |
| | | 31:24 | | | RUD[29:24] | | | | | |
| 0x0200 ... 0x026F | Reserved | | | | | | | | | |
| 0x0270 | GMAC_RXLPI | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | COUNT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0274 | GMAC_RXLPITIME | 7:0 | LPITIME[7:0] | | | | | | | |
| | | 15:8 | LPITIME[15:8] | | | | | | | |
| | | 23:16 | LPITIME[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0278 | GMAC_TXLPI | 7:0 | COUNT[7:0] | | | | | | | |
| | | 15:8 | COUNT[15:8] | | | | | | | |
| | | 23:16 | COUNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x027C | GMAC_TXLPITIME | 7:0 | LPITIME[7:0] | | | | | | | |
| | | 15:8 | LPITIME[15:8] | | | | | | | |
| | | 23:16 | LPITIME[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0280 ... 0x03FF | Reserved | | | | | | | | | |
| 0x0400 | GMAC_ISRPQ1 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0404 | GMAC_ISRPQ2 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0408 | GMAC_ISRPQ3 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x040C | GMAC_ISRPQ4 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0410 | GMAC_ISRPQ5 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0414 ... 0x043F | Reserved | | | | | | | | | |
| 0x0440 | GMAC_TBQBAPQ1 | 7:0 | TXBQBA[5:0] | | | | | | | |
| | | 15:8 | TXBQBA[13:6] | | | | | | | |
| | | 23:16 | TXBQBA[21:14] | | | | | | | |
| | | 31:24 | TXBQBA[29:22] | | | | | | | |
| 0x0444 | GMAC_TBQBAPQ2 | 7:0 | TXBQBA[5:0] | | | | | | | |
| | | 15:8 | TXBQBA[13:6] | | | | | | | |
| | | 23:16 | TXBQBA[21:14] | | | | | | | |
| | | 31:24 | TXBQBA[29:22] | | | | | | | |
| 0x0448 | GMAC_TBQBAPQ3 | 7:0 | TXBQBA[5:0] | | | | | | | |
| | | 15:8 | TXBQBA[13:6] | | | | | | | |
| | | 23:16 | TXBQBA[21:14] | | | | | | | |
| | | 31:24 | TXBQBA[29:22] | | | | | | | |
| 0x044C | GMAC_TBQBAPQ4 | 7:0 | TXBQBA[5:0] | | | | | | | |
| | | 15:8 | TXBQBA[13:6] | | | | | | | |
| | | 23:16 | TXBQBA[21:14] | | | | | | | |
| | | 31:24 | TXBQBA[29:22] | | | | | | | |
| 0x0450 | GMAC_TBQBAPQ5 | 7:0 | TXBQBA[5:0] | | | | | | | |
| | | 15:8 | TXBQBA[13:6] | | | | | | | |
| | | 23:16 | TXBQBA[21:14] | | | | | | | |
| | | 31:24 | TXBQBA[29:22] | | | | | | | |
| 0x0454 ... 0x047F | Reserved | | | | | | | | | |
| 0x0480 | GMAC_RBQBAPQ1 | 7:0 | RXBQBA[5:0] | | | | | | | |
| | | 15:8 | RXBQBA[13:6] | | | | | | | |
| | | 23:16 | RXBQBA[21:14] | | | | | | | |
| | | 31:24 | RXBQBA[29:22] | | | | | | | |
| 0x0484 | GMAC_RBQBAPQ2 | 7:0 | RXBQBA[5:0] | | | | | | | |
| | | 15:8 | RXBQBA[13:6] | | | | | | | |
| | | 23:16 | RXBQBA[21:14] | | | | | | | |
| | | 31:24 | RXBQBA[29:22] | | | | | | | |
| 0x0488 | GMAC_RBQBAPQ3 | 7:0 | RXBQBA[5:0] | | | | | | | |
| | | 15:8 | RXBQBA[13:6] | | | | | | | |
| | | 23:16 | RXBQBA[21:14] | | | | | | | |
| | | 31:24 | RXBQBA[29:22] | | | | | | | |
| 0x048C | GMAC_RBQBAPQ4 | 7:0 | RXBQBA[5:0] | | | | | | | |
| | | 15:8 | RXBQBA[13:6] | | | | | | | |
| | | 23:16 | RXBQBA[21:14] | | | | | | | |
| | | 31:24 | RXBQBA[29:22] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0490 | GMAC_RBQBAPQ5 | 7:0 | RXBQBA[5:0] | | | | | | | |
| | | 15:8 | RXBQBA[13:6] | | | | | | | |
| | | 23:16 | RXBQBA[21:14] | | | | | | | |
| | | 31:24 | RXBQBA[29:22] | | | | | | | |
| 0x0494 ... 0x049F | Reserved | | | | | | | | | |
| 0x04A0 | GMAC_RBSRPQ1 | 7:0 | RBS[7:0] | | | | | | | |
| | | 15:8 | RBS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04A4 | GMAC_RBSRPQ2 | 7:0 | RBS[7:0] | | | | | | | |
| | | 15:8 | RBS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04A8 | GMAC_RBSRPQ3 | 7:0 | RBS[7:0] | | | | | | | |
| | | 15:8 | RBS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04AC | GMAC_RBSRPQ4 | 7:0 | RBS[7:0] | | | | | | | |
| | | 15:8 | RBS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04B0 | GMAC_RBSRPQ5 | 7:0 | RBS[7:0] | | | | | | | |
| | | 15:8 | RBS[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04B4 ... 0x04BB | Reserved | | | | | | | | | |
| 0x04BC | GMAC_CBSCR | 7:0 | | | | | | | QAE | QBE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04C0 | GMAC_CBSISQA | 7:0 | IS[7:0] | | | | | | | |
| | | 15:8 | IS[15:8] | | | | | | | |
| | | 23:16 | IS[23:16] | | | | | | | |
| | | 31:24 | IS[31:24] | | | | | | | |
| 0x04C4 | GMAC_CBSISQB | 7:0 | IS[7:0] | | | | | | | |
| | | 15:8 | IS[15:8] | | | | | | | |
| | | 23:16 | IS[23:16] | | | | | | | |
| | | 31:24 | IS[31:24] | | | | | | | |
| 0x04C8 ... 0x04FF | Reserved | | | | | | | | | |
| 0x0500 | GMAC_ST1RPQ0 | 7:0 | DSTCM[3:0] | | | | | QNB[2:0] | | |
| | | 15:8 | UDPM[3:0] | | | | | DSTCM[7:4] | | |
| | | 23:16 | UDPM[11:4] | | | | | | | |
| | | 31:24 | | | UDPE | DSTCE | UDPM[15:12] | | | |
| 0x0504 | GMAC_ST1RPQ1 | 7:0 | DSTCM[3:0] | | | | | QNB[2:0] | | |
| | | 15:8 | UDPM[3:0] | | | | | DSTCM[7:4] | | |
| | | 23:16 | UDPM[11:4] | | | | | | | |
| | | 31:24 | | | UDPE | DSTCE | UDPM[15:12] | | | |
| 0x0508 | GMAC_ST1RPQ2 | 7:0 | DSTCM[3:0] | | | | | QNB[2:0] | | |
| | | 15:8 | UDPM[3:0] | | | | | DSTCM[7:4] | | |
| | | 23:16 | UDPM[11:4] | | | | | | | |
| | | 31:24 | | | UDPE | DSTCE | UDPM[15:12] | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x050C | GMAC_ST1RPQ3 | 7:0 | DSTCM[3:0] | | | | | QNB[2:0] | | |
| | | 15:8 | UDPM[3:0] | | | | DSTCM[7:4] | | | |
| | | 23:16 | UDPM[11:4] | | | | | | | |
| | | 31:24 | | | UDPE | DSTCE | UDPM[15:12] | | | |
| 0x0510 ... 0x053F | Reserved | | | | | | | | | |
| 0x0540 | GMAC_ST2RPQ0 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | | I2ETH[2:0] | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | | | COMPC[4:0] | | | COMPBE |
| 0x0544 | GMAC_ST2RPQ1 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | | I2ETH[2:0] | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | | | COMPC[4:0] | | | COMPBE |
| 0x0548 | GMAC_ST2RPQ2 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | | I2ETH[2:0] | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | | | COMPC[4:0] | | | COMPBE |
| 0x054C | GMAC_ST2RPQ3 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | | I2ETH[2:0] | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | | | COMPC[4:0] | | | COMPBE |
| 0x0550 | GMAC_ST2RPQ4 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | | I2ETH[2:0] | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | | | COMPC[4:0] | | | COMPBE |
| 0x0554 | GMAC_ST2RPQ5 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | | I2ETH[2:0] | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | | | COMPC[4:0] | | | COMPBE |
| 0x0558 | GMAC_ST2RPQ6 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | | I2ETH[2:0] | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | | | COMPC[4:0] | | | COMPBE |
| 0x055C | GMAC_ST2RPQ7 | 7:0 | | VLANP[2:0] | | | | QNB[2:0] | | |
| | | 15:8 | COMPA[2:0] | | | ETHE | | I2ETH[2:0] | | VLANE |
| | | 23:16 | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| | | 31:24 | | COMPCE | | | COMPC[4:0] | | | COMPBE |
| 0x0560 ... 0x05FF | Reserved | | | | | | | | | |
| 0x0600 | GMAC_IERPQ1 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0604 | GMAC_IERPQ2 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0608 | GMAC_IERPQ3 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x060C | GMAC_IERPQ4 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0610 | GMAC_IERPQ5 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0614 ... 0x061F | Reserved | | | | | | | | | |
| 0x0620 | GMAC_IDRPQ1 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0624 | GMAC_IDRPQ2 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0628 | GMAC_IDRPQ3 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x062C | GMAC_IDRPQ4 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0630 | GMAC_IDRPQ5 | 7:0 | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0634 ... 0x063F | Reserved | | | | | | | | | |
| 0x0640 | GMAC_IMRPQ1 | 7:0 | TCOMP | AHB | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0644 | GMAC_IMRPQ2 | 7:0 | TCOMP | AHB | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0648 | GMAC_IMRPQ3 | 7:0 | TCOMP | AHB | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x064C | GMAC_IMRPQ4 | 7:0 | TCOMP | AHB | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0650 | GMAC_IMRPQ5 | 7:0 | TCOMP | AHB | RLEX | | | RXUBR | RCOMP | |
| | | 15:8 | | | | | HRESP | ROVR | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0654 ... 0x06DF | Reserved | | | | | | | | | |
| 0x06E0 | GMAC_ST2ER0 | 7:0 | COMPVAL[7:0] | | | | | | | |
| | | 15:8 | COMPVAL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x06E4 | GMAC_ST2ER1 | 7:0 | COMPVAL[7:0] | | | | | | | |
| | | 15:8 | COMPVAL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06E8 | GMAC_ST2ER2 | 7:0 | COMPVAL[7:0] | | | | | | | |
| | | 15:8 | COMPVAL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06EC | GMAC_ST2ER3 | 7:0 | COMPVAL[7:0] | | | | | | | |
| | | 15:8 | COMPVAL[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06F0 ... 0x06FF | Reserved | | | | | | | | | |
| 0x0700 | GMAC_ST2CW00 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0704 | GMAC_ST2CW10 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0708 | GMAC_ST2CW01 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x070C | GMAC_ST2CW11 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0710 | GMAC_ST2CW02 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0714 | GMAC_ST2CW12 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0718 | GMAC_ST2CW03 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x071C | GMAC_ST2CW13 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0720 | GMAC_ST2CW04 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0724 | GMAC_ST2CW14 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0728 | GMAC_ST2CW05 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x072C | GMAC_ST2CW15 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0730 | GMAC_ST2CW06 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0734 | GMAC_ST2CW16 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0738 | GMAC_ST2CW07 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x073C | GMAC_ST2CW17 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0740 | GMAC_ST2CW08 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0744 | GMAC_ST2CW18 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0748 | GMAC_ST2CW09 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x074C | GMAC_ST2CW19 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0750 | GMAC_ST2CW010 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0754 | GMAC_ST2CW110 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0758 | GMAC_ST2CW011 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x075C | GMAC_ST2CW111 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0760 | GMAC_ST2CW012 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0764 | GMAC_ST2CW112 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0768 | GMAC_ST2CW013 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x076C | GMAC_ST2CW113 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0770 | GMAC_ST2CW014 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0774 | GMAC_ST2CW114 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0778 | GMAC_ST2CW015 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x077C | GMAC_ST2CW115 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0780 | GMAC_ST2CW016 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0784 | GMAC_ST2CW116 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0788 | GMAC_ST2CW017 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x078C | GMAC_ST2CW117 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0790 | GMAC_ST2CW018 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x0794 | GMAC_ST2CW118 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0798 | GMAC_ST2CW019 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x079C | GMAC_ST2CW119 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07A0 | GMAC_ST2CW020 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x07A4 | GMAC_ST2CW120 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07A8 | GMAC_ST2CW021 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x07AC | GMAC_ST2CW121 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07B0 | GMAC_ST2CW022 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x07B4 | GMAC_ST2CW122 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x07B8 | GMAC_ST2CW023 | 7:0 | MASKVAL[7:0] | | | | | | | |
| | | 15:8 | MASKVAL[15:8] | | | | | | | |
| | | 23:16 | COMPVAL[7:0] | | | | | | | |
| | | 31:24 | COMPVAL[15:8] | | | | | | | |
| 0x07BC | GMAC_ST2CW123 | 7:0 | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| | | 15:8 | | | | | | | | OFFSSTRT[1] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 38.8.1 GMAC Network Control Register

**Name:** GMAC_NCR
**Offset:** 0x000
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | FNP | TXPBPF | ENPBPR |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SRTSM | | | TXZQPF | TXPF | THALT | TSTART | BP |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WESTAT | INCSTAT | CLRSTAT | MPE | TXEN | RXEN | LBL | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bit 18 – FNP** Flush Next Packet
Writing a '1' to this bit will flush the next packet from the external RX DPRAM. Flushing the next packet will only take effect if the DMA is not currently writing a packet already stored in the DPRAM to memory.

**Bit 17 – TXPBPF** Transmit PFC Priority-based Pause Frame
Takes the values stored in the Transmit PFC Pause Register.

**Bit 16 – ENPBPR** Enable PFC Priority-based Pause Reception
Writing a '1' to this bit enables PFC Priority Based Pause Reception capabilities, enabling PFC negotiation and recognition of priority-based pause frames.

| Value | Description |
|---|---|
| 0 | Normal operation |
| 1 | PFC Priority-based Pause frames are recognized. |

**Bit 15 – SRTSM** Store Receive Time Stamp to Memory
Writing a '1' to this bit causes the CRC of every received frame to be replaced with the value of the nanoseconds field of the 1588 timer that was captured as the receive frame passed the message time stamp point.
Note that bit RFCS in register GMAC_NCFGR may not be set to 1 when the timer should be captured.

| Value | Description |
|---|---|
| 0 | Normal operation |
| 1 | All received frames' CRC is replaced with a time stamp. |

**Bit 12 – TXZQPF** Transmit Zero Quantum Pause Frame
Writing a '1' to this bit causes a pause frame with zero quantum to be transmitted.
Writing a '0' to this bit has no effect.

**Bit 11 – TXPF** Transmit Pause Frame
Writing one to this bit causes a pause frame to be transmitted.
Writing a '0' to this bit has no effect.

**Bit 10 – THALT** Transmit Halt
Writing a '1' to this bit halts transmission as soon as any ongoing frame transmission ends.
Writing a '0' to this bit has no effect.

**Bit 9 – TSTART** Start Transmission
Writing a '1' to this bit starts transmission.
Writing a '0' to this bit has no effect.

**Bit 8 – BP** Back Pressure
In 10M or 100M half duplex mode, writing a '1' to this bit forces collisions on all received frames. Ignored in gigabit half duplex mode.

| Value | Description |
|---|---|
| 0 | Frame collisions are not forced. |
| 1 | Frame collisions are forced in 10M and 100M half duplex mode. |

**Bit 7 – WESTAT** Write Enable for Statistics Registers
Writing a '1' to this bit makes the statistics registers writable for functional test purposes.

| Value | Description |
|---|---|
| 0 | Statistics Registers are write-protected. |
| 1 | Statistics Registers are write-enabled. |

**Bit 6 – INCSTAT** Increment Statistics Registers
Writing a '1' to this bit increments all Statistics Registers by one for test purposes.
Writing a '0' to this bit has no effect.
This bit will always read '0'.

**Bit 5 – CLRSTAT** Clear Statistics Registers
Writing a '1' to this bit clears the Statistics Registers.
Writing a '0' to this bit has no effect.
This bit will always read '0'.

**Bit 4 – MPE** Management Port Enable
Writing a '1' to this bit enables the Management Port.
Writing a '0' to this bit disables the Management Port, and forces MDIO to high impedance state and MDC to low impedance.

| Value | Description |
|---|---|
| 0 | Management Port is disabled. |
| 1 | Management Port is enabled. |

**Bit 3 – TXEN** Transmit Enable
Writing a '1' to this bit enables the GMAC transmitter to send data.
Writing a '0' to this bit stops transmission immediately, the transmit pipeline and control registers is cleared, and the Transmit Queue Pointer Register will be set to point to the start of the transmit descriptor list.

| Value | Description |
|---|---|
| 0 | Transmit is disabled. |
| 1 | Transmit is enabled. |

**Bit 2 – RXEN** Receive Enable
Writing a '1' to this bit enables the GMAC to receive data.
Writing a '0' to this bit stops frame reception immediately, and the receive pipeline is cleared. The Receive Queue Pointer Register is not affected.

| Value | Description |
|---|---|
| 0 | Receive is disabled. |
| 1 | Receive is enabled. |

**Bit 1 – LBL** Loop Back Local
Writing '1' to this bit connects GTX to GRX, GTXEN to GRXDV, and forces full duplex mode.

© 2021 Microchip Technology Inc.
and its subsidiaries

GRXCK and GTXCK may malfunction as the GMAC is switched into and out of internal loop back. It is important that receive and transmit circuits have already been disabled when making the switch into and out of internal loop back.

| Value | Description |
|---|---|
| 0 | Loop back local is disabled. |
| 1 | Loop back local is enabled. |

### 38.8.2 GMAC Network Configuration Register

**Name:** GMAC_NCFGR
**Offset:** 0x004
**Reset:** 0x00080000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | IRXER | RXBP | IPGSEN | | IRXFCS | EFRHD | RXCOEN |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DCPF | DBW[1:0] | | CLK[2:0] | | | RFCS | LFERD |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXBUFO[1:0] | | PEN | RTY | | | | MAXFS |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNIHEN | MTIHEN | NBC | CAF | JFRAME | DNVLAN | FD | SPD |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – IRXER** Ignore IPG GRXER
When this bit is written to '1', the Receive Error signal (GRXER) has no effect on the GMAC operation when Receive Data Valid signal (GRXDV) is low.

**Bit 29 – RXBP** Receive Bad Preamble
When written to '1', frames with non-standard preamble are not rejected.

**Bit 28 – IPGSEN** IP Stretch Enable
Writing a '1' to this bit allows the transmit IPG to increase above 96 bit times, depending on the previous frame length using the IPG Stretch Register.

**Bit 26 – IRXFCS** Ignore RX FCS
For normal operation this bit must be written to zero.
When this bit is written to '1', frames with FCS/CRC errors will not be rejected. FCS error statistics will still be collected for frames with bad FCS, and FCS status will be recorded in the DMA descriptor of the frame.

**Bit 25 – EFRHD** Enable Frames Received in half-duplex
Writing a '1' to this bit enables frames to be received in half-duplex mode while transmitting.

**Bit 24 – RXCOEN** Receive Checksum Offload Enable
Writing a '1' to this bit enables the receive checksum engine, and frames with bad IP, TCP or UDP checksums are discarded.

**Bit 23 – DCPF** Disable Copy of Pause Frames
Writing a '1' to this bit prevents valid pause frames from being copied to memory. Pause frames are not copied regardless of the state of the Copy All Frames (CAF) bit, whether a hash match is found or whether a type ID match is identified.
If a destination address match is found, the pause frame will be copied to memory. Note that valid pause frames received will still increment pause statistics and pause the transmission of frames, as required.

© 2021 Microchip Technology Inc. and its subsidiaries

**Bits 22:21 – DBW[1:0]** Data Bus Width
Should always be written to '0'.

| Value | Name | Description |
|---|---|---|
| 0 | DBW32 | 32-bit data bus width |
| 1 | DBW64 | 64-bit data bus width |

**Bits 20:18 – CLK[2:0]** MDC Clock Division
These bits must be set according to MCK speed, and determine the number MCK will be divided by to generate Management Data Clock (MDC). For conformance with the 802.3 specification, MDC must not exceed 2.5MHz.
**Note:** MDC is only active during MDIO read and write operations.

| Value | Name | Description |
|---|---|---|
| 0 | MCK_8 | MCK divided by 8 (MCK up to 20MHz) |
| 1 | MCK_16 | MCK divided by 16 (MCK up to 40MHz) |
| 2 | MCK_32 | MCK divided by 32 (MCK up to 80MHz) |
| 3 | MCK_48 | MCK divided by 48 (MCK up to 120MHz) |
| 4 | MCK_64 | MCK divided by 64 (MCK up to 160MHz) |
| 5 | MCK_96 | MCK divided by 96 (MCK up to 240MHz) |

**Bit 17 – RFCS** Remove FCS
Writing this bit to '1' will cause received frames to be written to memory without their frame check sequence (last 4 bytes). The indicated frame length will be reduced by four bytes in this mode.

**Bit 16 – LFERD** Length Field Error Frame Discard
Writing a '1' to this bit discards frames with a measured length shorter than the extracted length field (as indicated by bytes 13 and 14 in a non-VLAN tagged frame). This only applies to frames with a length field less than 0x0600.

**Bits 15:14 – RXBUFO[1:0]** Receive Buffer Offset
These bits determine the number of bytes by which the received data is offset from the start of the receive buffer.

**Bit 13 – PEN** Pause Enable
When written to '1', transmission will pause if a non-zero 802.3 classic pause frame is received and PFC has not been negotiated.

**Bit 12 – RTY** Retry Test
This bit must be written to '0' for normal operation.
When writing a '1' to this bit, the back-off between collisions will always be one slot time. This setting helps testing the too many retries condition. This setting is also useful for pause frame tests by reducing the pause counter's decrement time from "512 bit times" to "every GRXCK cycle".

**Bit 8 – MAXFS** 1536 Maximum Frame Size
Writing a '1' to this bit increases the maximum accepted frame size to 1536 bytes in length. When written to '0', any frame above 1518 bytes in length is rejected.

**Bit 7 – UNIHEN** Unicast Hash Enable
When writing a '1' to this bit, unicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash Register.
Writing a '0' to this bit disables unicast hashing.

**Bit 6 – MTIHEN** Multicast Hash Enable
When writing a '1' to this bit, multicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash Register.
Writing a '0' to this bit disables multicast hashing.

**Bit 5 – NBC** No Broadcast
Writing a '1' to this bit will reject frames addressed to the broadcast address 0xFFFFFFFFFFFF (all '1').
Writing a '0' to this bit allows broadcasting to 0xFFFFFFFFFFFF.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 4 – CAF** Copy All Frames
When writing a '1' to this bit, all valid frames will be accepted.

**Bit 3 – JFRAME** Jumbo Frame Size
Writing a '1' to this bit enables jumbo frames of up to 10240 bytes to be accepted. The default length is 10240 bytes.

**Bit 2 – DNVLAN** Discard Non-VLAN Frames
Writing a '1' to this bit allows only VLAN-tagged frames to pass to the address matching logic.
Writing a '0' to this bit allows both VLAN_tagged and untagged frames to pass to the address matching logic.

**Bit 1 – FD** Full Duplex
Writing a '1' enables full duplex operation, so the transmit block ignores the state of collision and carrier sense and allows receive while transmitting.
Writing a '0' disables full duplex operation.

**Bit 0 – SPD** Speed
Writing a '1' selects 100Mbps operation.
Writing a '0' to this bit selects 10Mbps operation.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.3 GMAC Network Status Register

**Name:** GMAC_NSR
**Offset:** 0x008
**Reset:** 0x000001X0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | IDLE | MDIO | |
| Access | | | | | | R | R | |
| Reset | | | | | | 0 | 0 | |

**Bit 2 – IDLE** PHY Management Logic Idle
The PHY management logic is idle (i.e., has completed).

**Bit 1 – MDIO** MDIO Input Status
Returns status of the MDIO pin.

### 38.8.4 GMAC User Register

**Name:** GMAC_UR
**Offset:** 0x00C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 –** Reduced MII Mode

| Value | Description |
|---|---|
| 0 | RMII mode is selected |
| 1 | MII mode is selected |

### 38.8.5 GMAC DMA Configuration Register

**Name:** GMAC_DCFGR
**Offset:** 0x010
**Reset:** 0x00020004
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DDRP |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DRBS[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TXCOEN | TXPBMS | RXBMS[1:0] | |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ESPA | ESMA | | FBLDO[4:0] | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | | 0 | 0 | 1 | 0 | 0 |

**Bit 24 – DDRP** DMA Discard Receive Packets
A write to this bit is ignored if the DMA is not configured in the packet buffer full store and forward mode.

| Value | Description |
|---|---|
| 0 | Received packets are stored in the SRAM based packet buffer until next AHB buffer resource becomes available. |
| 1 | Receive packets from the receiver packet buffer memory are automatically discarded when no AHB resource is available. |

**Bits 23:16 – DRBS[7:0]** DMA Receive Buffer Size
These bits defined by these bits determines the size of buffer to use in main AHB system memory when writing received data.
The value is defined in multiples of 64 bytes. For example:

- 0x02: 128 bytes
- 0x18: 1536 bytes (1 × max length frame/buffer)
- 0xA0: 10240 bytes (1 × 10K jumbo frame/buffer)

**WARNING** Do not write 0x00 to this bit field.

**Bit 11 – TXCOEN** Transmitter Checksum Generation Offload Enable
Transmitter IP, TCP and UDP checksum generation offload enable.

| Value | Description |
|---|---|
| 0 | Frame data is unaffected. |
| 1 | The transmitter checksum generation engine calculates and substitutes checksums for transmit frames. |

**Bit 10 – TXPBMS** Transmitter Packet Buffer Memory Size Select
When written to zero, the amount of memory used for the transmit packet buffer is reduced by 50%. This reduces the amount of memory used by the GMAC.
It is important to write this bit to '1' if the full configured physical memory is available. The value in parentheses represents the size that would result for the default maximum configured memory size of 4KBytes.

| Value | Description |
|---|---|
| 0 | Top address bits not used. (2KByte used.) |
| 1 | Full configured addressable space (4KBytes) used. |

**Bits 9:8 – RXBMS[1:0]** Receiver Packet Buffer Memory Size Select
The default receive packet buffer size is FULL=4 Kbytes. The table below shows how to configure this memory to FULL, HALF, QUARTER or EIGHTH of the default size.

| Value | Name | Description |
|---|---|---|
| 0 | EIGHTH | 4/8 Kbyte Memory Size |
| 1 | QUARTER | 4/4 Kbytes Memory Size |
| 2 | HALF | 4/2 Kbytes Memory Size |
| 3 | FULL | 4 Kbytes Memory Size |

**Bit 7 – ESPA** Endian Swap Mode Enable for Packet Data Accesses

| Value | Description |
|---|---|
| 0 | Little endian mode for AHB transfers selected. |
| 1 | Big endian mode for AHB transfers selected. |

**Bit 6 – ESMA** Endian Swap Mode Enable for Management Descriptor Accesses

| Value | Description |
|---|---|
| 0 | Little endian mode for AHB transfers selected. |
| 1 | Big endian mode for AHB transfers selected. |

**Bits 4:0 – FBLDO[4:0]** Fixed Burst Length for DMA Data Operations
Selects the burst length to attempt to use on the AHB when transferring frame data. Not used for DMA management operations and only used where space and data size allow. Otherwise SINGLE type AHB transfers are used.
One-hot priority encoding enforced automatically on register writes as follows. ‘x’ represents don’t care.

| Value | Name | Description |
|---|---|---|
| 0 | - | Reserved |
| 1 | SINGLE | 00001: Always use SINGLE AHB bursts |
| 2 | - | Reserved |
| 4 | INCR4 | 001xx: Attempt to use INCR4 AHB bursts (Default) |
| 8 | INCR8 | 01xxx: Attempt to use INCR8 AHB bursts |
| 16 | INCR16 | 1xxxx: Attempt to use INCR16 AHB bursts |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.6 GMAC Transmit Status Register

**Name:** GMAC_TSR
**Offset:** 0x014
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | HRESP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TXCOMP | TFC | TXGO | RLE | COL | UBR |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – HRESP** HRESP Not OK
Set when the DMA block sees HRESP not OK.
This bit is cleared by writing a '1' to it.

**Bit 5 – TXCOMP** Transmit Complete
Set when a frame has been transmitted.
This bit is cleared by writing a '1' to it.

**Bit 4 – TFC** Transmit Frame Corruption Due to AHB Error
This bit is set when an error occurs during reading transmit frame from the AHB. Error causes include HRESP errors and buffers exhausted mid frame. (If the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted).
In DMA packet buffer mode, this bit is also set if a single frame is too large for the configured packet buffer memory size.
This bit is cleared by writing a '1' to it.

**Bit 3 – TXGO** Transmit Go
This bit is '1' when transmit is active. When using the DMA interface this bit represents the TXGO variable as specified in the transmit buffer description.

**Bit 2 – RLE** Retry Limit Exceeded
This bit is cleared by writing a '1' to it.

**Bit 1 – COL** Collision Occurred
When operating in 10/100Mbps mode, this bit is set by the assertion of either a collision or a late collision.
This bit is cleared by writing a '1' to it.

**Bit 0 – UBR** Used Bit Read
This bit is set when a transmit buffer descriptor is read with its used bit set.

This bit is cleared by writing a '1' to it.

### 38.8.7 GMAC Receive Buffer Queue Base Address Register

**Name:** GMAC_RBQB
**Offset:** 0x018
**Reset:** 0x00000000
**Property:** Read/Write

This register holds the start address of the receive buffer queue (receive buffers descriptor list). The receive buffer queue base address must be initialized before receive is enabled through bit 2 of the Network Control Register. Once reception is enabled, any write to the Receive Buffer Queue Base Address Register is ignored. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the “used” bits.

In terms of AMBA AHB operation, the descriptors are read from memory using a single 32-bit AHB access. The descriptors should be aligned at 32-bit boundaries and the descriptors are written to using two individual non sequential accesses.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – ADDR[29:0]** Receive Buffer Queue Base Address
Written with the address of the start of the receive queue.

### 38.8.8 GMAC Transmit Buffer Queue Base Address Register

**Name:** GMAC_TBQB
**Offset:** 0x01C
**Reset:** 0x00000000
**Property:** -

This register holds the start address of the transmit buffer queue (transmit buffers descriptor list). The Transmit Buffer Queue Base Address Register must be initialized before transmit is started through bit 9 of the Network Control Register. Once transmission has started, any write to the Transmit Buffer Queue Base Address Register is illegal and therefore ignored.

Note that due to clock boundary synchronization, it takes a maximum of four MCK cycles from the writing of the transmit start bit before the transmitter is active. Writing to the Transmit Buffer Queue Base Address Register during this time may produce unpredictable results.

Reading this register returns the location of the descriptor currently being accessed. Since the DMA handles two frames at once, this may not necessarily be pointing to the current frame being transmitted.

In terms of AMBA AHB operation, the descriptors are written to memory using a single 32-bit AHB access. The descriptors should be aligned at 32-bit boundaries and the descriptors are read from memory using two individual non sequential accesses.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – ADDR[29:0]** Transmit Buffer Queue Base Address
Written with the address of the start of the transmit queue.

### 38.8.9 GMAC Receive Status Register

**Name:** GMAC_RSR
**Offset:** 0x020
**Reset:** 0x00000000
**Property:** -

This register, when read, provides receive status details. Once read, individual bits may be cleared by writing a '1' to them. It is not possible to set a bit to '1' by writing to this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HNO | RXOVR | REC | BNA |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – HNO** HRESP Not OK
This bit is set when the DMA block sees HRESP not OK.
This bit is cleared by writing a '1' to it.

**Bit 2 – RXOVR** Receive Overrun
This bit is set if the receive status was not taken at the end of the frame. The buffer will be recovered if an overrun occurs.
This bit is cleared by writing a '1' to it.

**Bit 1 – REC** Frame Received
This bit is set to when one or more frames have been received and placed in memory.
This bit is cleared by writing a '1' to it.

**Bit 0 – BNA** Buffer Not Available
When this bit is set, an attempt was made to get a new buffer and the pointer indicated that it was owned by the processor. The DMA will re-read the pointer each time an end of frame is received until a valid pointer is found. This bit is set following each descriptor read attempt that fails, even if consecutive pointers are unsuccessful and software has in the mean time cleared the status flag.
This bit is cleared by writing a '1' to it.

### 38.8.10 GMAC Interrupt Status Register

**Name:** GMAC_ISR
**Offset:** 0x024
**Reset:** 0x00000000
**Property:** Read-only

This register indicates the source of the interrupt. An interrupt source must be enabled in the mask register first so the corresponding bits of this register will be set and the GMAC interrupt signal will be asserted in the system.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TSUTIMCMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| Access | R | R | R | R | R | R | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| Access | | R | R | R | R | R | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – TSUTIMCMP** TSU Timer Comparison
Indicates when TSU timer count value is equal to programmed value.
Cleared on read.

**Bit 28 – WOL** Wake On LAN
WOL interrupt. Indicates a WOL message has been received.

**Bit 27 – RXLPISBC** Receive LPI indication Status Bit Change
Receive LPI indication status bit change.
Cleared on read.

**Bit 26 – SRI** TSU Seconds Register Increment
Indicates the register has incremented.
Cleared on read.

**Bit 25 – PDRSFT** PDelay Response Frame Transmitted
Indicates a PTP pdelay_resp frame has been transmitted.
Cleared on read.

**Bit 24 – PDRQFT** PDelay Request Frame Transmitted
Indicates a PTP pdelay_req frame has been transmitted.
Cleared on read.

**Bit 23 – PDRSFR** PDelay Response Frame Received
Indicates a PTP pdelay_resp frame has been received.
Cleared on read.

**Bit 22 – PDRQFR** PDelay Request Frame Received
Indicates a PTP pdelay_req frame has been received.
Cleared on read.

**Bit 21 – SFT** PTP Sync Frame Transmitted
Indicates a PTP sync frame has been transmitted.
Cleared on read.

**Bit 20 – DRQFT** PTP Delay Request Frame Transmitted
Indicates a PTP delay_req frame has been transmitted.
Cleared on read.

**Bit 19 – SFR** PTP Sync Frame Received
Indicates a PTP sync frame has been received.
Cleared on read.

**Bit 18 – DRQFR** PTP Delay Request Frame Received
Indicates a PTP delay_req frame has been received.
Cleared on read.

**Bit 14 – PFTR** Pause Frame Transmitted
Indicates a pause frame has been successfully transmitted after being initiated from the Network Control Register.
Cleared on read.

**Bit 13 – PTZ** Pause Time Zero
Set when either the Pause Time Register at address 0x38 decrements to zero, or when a valid pause frame is received with a zero pause quantum field.
Cleared on read.

**Bit 12 – PFNZ** Pause Frame with Non-zero Pause Quantum Received
Indicates a valid pause has been received that has a non-zero pause quantum field.
Cleared on read.

**Bit 11 – HRESP** HRESP Not OK
Set when the DMA block sees HRESP not OK.
Cleared on read.

**Bit 10 – ROVR** Receive Overrun
Set when the receive overrun status bit is set.
Cleared on read.

**Bit 7 – TCOMP** Transmit Complete
Set when a frame has been transmitted.
Cleared on read.

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error
Transmit frame corruption due to AHB error. Set if an error occurs during reading a transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame.

**Bit 5 – RLEX** Retry Limit Exceeded
Retry Limit Exceeded Transmit error.
Cleared on read.

**Bit 4 – TUR** Transmit Underrun
This interrupt is set if the transmitter was forced to terminate an ongoing frame transmission due to further data being unavailable.

© 2021 Microchip Technology Inc.
and its subsidiaries

This interrupt is also set if a transmitter status write back has not completed when another status write back is attempted.
This interrupt is also set when the transmit DMA has written the SOP data into the FIFO and either the AHB bus was not granted in time for further data, or because an AHB not OK response was returned, or because the used bit was read.

**Bit 3 – TXUBR** TX Used Bit Read
Set when a transmit buffer descriptor is read with its used bit set.
Cleared on read.

**Bit 2 – RXUBR** RX Used Bit Read
Set when a receive buffer descriptor is read with its used bit set.
Cleared on read.

**Bit 1 – RCOMP** Receive Complete
A frame has been stored in memory.
Cleared on read.

**Bit 0 – MFS** Management Frame Sent
The PHY Maintenance Register has completed its operation.
Cleared on read.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.11 GMAC Interrupt Enable Register

**Name:** GMAC_IER
**Offset:** 0x028
**Reset:** –
**Property:** Write-only

This register is write-only and will always return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TSUTIMCMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| Access | | | W | W | R | W | W | W |
| Reset | | | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – TSUTIMCMP** TSU Timer Comparison

**Bit 28 – WOL** Wake On LAN

**Bit 27 – RXLPISBC** Receive LPI indication Status Bit Change
Receive LPI indication status bit change.
Cleared on read.

**Bit 26 – SRI** TSU Seconds Register Increment

**Bit 25 – PDRSFT** PDelay Response Frame Transmitted

**Bit 24 – PDRQFT** PDelay Request Frame Transmitted

**Bit 23 – PDRSFR** PDelay Response Frame Received

**Bit 22 – PDRQFR** PDelay Request Frame Received

**Bit 21 – SFT** PTP Sync Frame Transmitted

**Bit 20 – DRQFT** PTP Delay Request Frame Transmitted

**Bit 19 – SFR** PTP Sync Frame Received

**Bit 18 – DRQFR** PTP Delay Request Frame Received

**Bit 15 – EXINT** External Interrupt

**Bit 14 – PFTR** Pause Frame Transmitted

**Bit 13 – PTZ** Pause Time Zero

**Bit 12 – PFNZ** Pause Frame with Non-zero Pause Quantum Received

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 4 – TUR** Transmit Underrun

**Bit 3 – TXUBR** TX Used Bit Read

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

**Bit 0 – MFS** Management Frame Sent

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.12 GMAC Interrupt Disable Register

**Name:** GMAC_IDR
**Offset:** 0x02C
**Reset:** –
**Property:** Write-only

This register is write-only and will always return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TSUTIMCMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| Access | | | W | W | R | W | W | W |
| Reset | | | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – TSUTIMCMP** TSU Timer Comparison

**Bit 28 – WOL** Wake On LAN

**Bit 27 – RXLPISBC** Receive LPI indication Status Bit Change
Receive LPI indication status bit change.
Cleared on read.

**Bit 26 – SRI** TSU Seconds Register Increment

**Bit 25 – PDRSFT** PDelay Response Frame Transmitted

**Bit 24 – PDRQFT** PDelay Request Frame Transmitted

**Bit 23 – PDRSFR** PDelay Response Frame Received

**Bit 22 – PDRQFR** PDelay Request Frame Received

**Bit 21 – SFT** PTP Sync Frame Transmitted

**Bit 20 – DRQFT** PTP Delay Request Frame Transmitted

**Bit 19 – SFR** PTP Sync Frame Received

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 18 – DRQFR** PTP Delay Request Frame Received

**Bit 15 – EXINT** External Interrupt

**Bit 14 – PFTR** Pause Frame Transmitted

**Bit 13 – PTZ** Pause Time Zero

**Bit 12 – PFNZ** Pause Frame with Non-zero Pause Quantum Received

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 4 – TUR** Transmit Underrun

**Bit 3 – TXUBR** TX Used Bit Read

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

**Bit 0 – MFS** Management Frame Sent

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.13 GMAC Interrupt Mask Register

**Name:** GMAC_IMR
**Offset:** 0x030
**Reset:** 0x07FFFFFF
**Property:** Read/Write

This register is a read-only register indicating which interrupts are masked. All bits are set at Reset and can be reset individually by writing to the Interrupt Enable Register (GMAC_IER), or set individually by writing to the Interrupt Disable Register (GMAC_IDR).

For test purposes there is a write-only function to this register that allows the bits in the Interrupt Status Register to be set or cleared, regardless of the state of the mask register. A write to this register directly affects the state of the corresponding bit in the Interrupt Status Register, causing an interrupt to be generated if a 1 is written.

The following values are valid for all listed bit names of this register when read:

0: The corresponding interrupt is enabled.

1: The corresponding interrupt is not enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TSUTIMCMP | WOL | RXLPISBC | SRI | PDRSFT | PDRQFT |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PDRSFR | PDRQFR | SFT | DRQFT | SFR | DRQFR | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EXINT | PFTR | PTZ | PFNZ | HRESP | ROVR | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | TUR | TXUBR | RXUBR | RCOMP | MFS |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 29 – TSUTIMCMP** TSU Timer Comparison

**Bit 28 – WOL** Wake On LAN

**Bit 27 – RXLPISBC** Receive LPI indication Status Bit Change
Receive LPI indication status bit change.
Cleared on read.

**Bit 26 – SRI** TSU Seconds Register Increment

**Bit 25 – PDRSFT** PDelay Response Frame Transmitted

**Bit 24 – PDRQFT** PDelay Request Frame Transmitted

**Bit 23 – PDRSFR** PDelay Response Frame Received

**Bit 22 – PDRQFR** PDelay Request Frame Received

**Bit 21 – SFT** PTP Sync Frame Transmitted

**Bit 20 – DRQFT** PTP Delay Request Frame Transmitted

**Bit 19 – SFR** PTP Sync Frame Received

**Bit 18 – DRQFR** PTP Delay Request Frame Received

**Bit 15 – EXINT** External Interrupt

**Bit 14 – PFTR** Pause Frame Transmitted

**Bit 13 – PTZ** Pause Time Zero

**Bit 12 – PFNZ** Pause Frame with Non-zero Pause Quantum Received

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded

**Bit 4 – TUR** Transmit Underrun

**Bit 3 – TXUBR** TX Used Bit Read

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

**Bit 0 – MFS** Management Frame Sent

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.14 GMAC PHY Maintenance Register

**Name:** GMAC_MAN
**Offset:** 0x034
**Reset:** 0x00000000
**Property:** Read/Write

This register is a shift register. Writing to it starts a shift operation which is signaled completed when bit 2 is set in the Network Status Register (GMAC_NSR). It takes about 2000 MCK cycles to complete, when MDC is set for MCK divide by 32 in the Network Configuration Register. An interrupt is generated upon completion.

During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each MDC cycle. This causes transmission of a PHY management frame on MDIO. Refer also to section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation returns the current contents of the shift register. At the end of management operation, the bits will have shifted back to their original locations. For a read operation, the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The MDIO interface can read IEEE 802.3 clause 45 PHYs, as well as clause 22 PHYs. To read clause 45 PHYs, bit 30 should be written with a '0' rather than a '1'. To write clause 45 PHYs, bits 31:28 should be written as 0x1:

| PHY | Access | Bit Value | | | |
|---|---|---|---|---|---|
| | | WZO | CLTTO | OP[1] | OP[0] |
| Clause 22 | Read | 0 | 1 | 1 | 0 |
| | Write | 0 | 1 | 0 | 1 |
| Clause 45 | Read | 0 | 0 | 1 | 1 |
| | Write | 0 | 0 | 0 | 1 |
| | Read + Address | 0 | 0 | 1 | 0 |

For a description of MDC generation, see also the 'GMAC Network Configuration Register' (GMAC_NCR) description.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WZO | CLTTO | OP[1:0] | | PHYA[4:1] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PHYA[0] | REGA[4:0] | | | | | WTN[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – WZO** Write ZERO
Must be written to '0'.

| Value | Description |
|---|---|
| 0 | Mandatory |
| 1 | Reserved |

**Bit 30 – CLTTO** Clause 22 Operation

| Value | Description |
|---|---|
| 0 | Clause 45 operation |
| 1 | Clause 22 operation |

**Bits 29:28 – OP[1:0]** Operation

| Value | Description |
|---|---|
| 01 | Write |
| 10 | Read |
| Other | Reseved |

**Bits 27:23 – PHYA[4:0]** PHY Address

**Bits 22:18 – REGA[4:0]** Register Address
Specifies the register in the PHY to access.

**Bits 17:16 – WTN[1:0]** Write Ten
Must be written to '10'.

| Value | Description |
|---|---|
| 10 | Mandatory |
| Other | Reserved |

**Bits 15:0 – DATA[15:0]** PHY Data
For a write operation, this field is written with the data to be written to the PHY.
After a read operation, this field contains the data read from the PHY.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.15 GMAC Receive Pause Quantum Register

**Name:** GMAC_RPQ
**Offset:** 0x038
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RPQ[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RPQ[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RPQ[15:0]** Received Pause Quantum
Stores the current value of the Receive Pause Quantum Register which is decremented every 512 bit times.

### 38.8.16 GMAC Transmit Pause Quantum Register

**Name:** GMAC_TPQ
**Offset:** 0x03C
**Reset:** 0x0000FFFF
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TPQ[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TPQ[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 15:0 – TPQ[15:0]** Transmit Pause Quantum
Written with the pause quantum value for pause frame transmission.

### 38.8.17 GMAC TX Partial Store and Forward Register

**Name:** GMAC_TPSF
**Offset:** 0x040
**Reset:** 0x00000FFF
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENTXP | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TPB1ADR[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TPB1ADR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 31 – ENTXP** Enable TX Partial Store and Forward Operation

**Bits 11:0 – TPB1ADR[11:0]** Transmit Partial Store and Forward Address Watermark value.

### 38.8.18 GMAC RX Partial Store and Forward Register

**Name:** GMAC_RPSF
**Offset:** 0x044
**Reset:** 0x00000FFF
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENRXP | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RPB1ADR[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RPB1ADR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bit 31 – ENRXP** Enable RX Partial Store and Forward Operation

**Bits 11:0 – RPB1ADR[11:0]** Receive Partial Store and Forward Address Watermark value. Reset = 1.

### 38.8.19 GMAC RX Jumbo Frame Max Length Register

**Name:** GMAC_RJFML
**Offset:** 0x048
**Reset:** 0x00003FFF
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | FML[13:8] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FML[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 13:0 – FML[13:0]** Frame Max Length
Rx jumbo frame maximum length.

### 38.8.20 GMAC Hash Register Bottom

**Name:** GMAC_HRB
**Offset:** 0x080
**Reset:** 0x00000000
**Property:** Read/Write

The unicast hash enable (UNIHEN) and the multicast hash enable (MITIHEN) bits in the Network Configuration Register (GMAC_NCFGR) enable the reception of hash matched frames.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Hash Address
The first 32 bits of the Hash Address Register.

### 38.8.21 GMAC Hash Register Top

**Name:** GMAC_HRT
**Offset:** 0x084
**Reset:** 0x00000000
**Property:** Read/Write

The Unicast Hash Enable (UNIHEN) and the Multicast Hash Enable (MITIHEN) bits in the Network Configuration Register (GMAC_NCFGR) enable the reception of hash matched frames.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Hash Address
Bits 63 to 32 of the Hash Address Register.

### 38.8.22 GMAC Specific Address n Bottom Register

**Name:** GMAC_SABx
**Offset:** 0x88 + (x-1)*0x08 [x=1..4]
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Specific Address n
Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

### 38.8.23 GMAC Specific Address n Top Register

**Name:** GMAC_SATx
**Offset:** 0x8C + (x-1)*0x08 [x=1..4]
**Reset:** 0x00000000
**Property:** Read/Write

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – ADDR[15:0]** Specific Address n
The most significant bits of the destination address, that is, bits 47:32.

### 38.8.24 GMAC Type ID Match n Register

**Name:** GMAC_TIDMx
**Offset:** 0xA8 + (x-1)*0x04 [x=1..4]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ENIDn | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TID[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TID[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ENIDn** Enable Copying of TID Matched Frames

| Value | Description |
|---|---|
| 0 | TID n is not part of the comparison match. |
| 1 | TID n is processed for the comparison match. |

**Bits 15:0 – TID[15:0]** Type ID Match n
For use in comparisons with received frames type ID/length frames.

### 38.8.25 GMAC Wake on LAN Register

**Name:** GMAC_WOL
**Offset:** 0x0B8
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MTI | SA1 | ARP | MAG |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 19 – MTI** Multicast Hash Event Enable

| Value | Description |
|---|---|
| 0 | Wake on LAN multicast hash Event disabled |
| 1 | Wake on LAN multicast hash Event enabled |

**Bit 18 – SA1** Specific Address Register 1 Event Enable

| Value | Description |
|---|---|
| 0 | Wake on Specific Address Register 1 Event disabled |
| 1 | Wake on Specific Address Register 1 Event enabled |

**Bit 17 – ARP** ARP Request Event Enable

| Value | Description |
|---|---|
| 0 | Wake on LAN ARP request Event disabled |
| 1 | Wake on LAN ARP request Event enabled |

**Bit 16 – MAG** Magic Packet Event Enable

| Value | Description |
|---|---|
| 0 | Wake on LAN magic packet Event disabled |
| 1 | Wake on LAN magic packet Event enabled |

**Bits 15:0 – IP[15:0]** ARP Request IP Address
Wake on LAN ARP request IP address. Written to define the 16 least significant bits of the target IP address that is matched to generate a Wake on LAN event.

| Value | Description |
|---|---|
| 0x0000 | No Event generated, even if matched by the received frame. |
| 0x0001-0xFFFF | Wake on LAN Event generated for matching LSB of the target IP address. |

### 38.8.26 GMAC IPG Stretch Register

**Name:** GMAC_IPGS
**Offset:** 0x0BC
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – FL[15:0]** Frame Length
Bits FL[7:0] are multiplied with the previously transmitted frame length (including preamble), and divided by FL[15:8]+1 (adding 1 to prevent division by zero). $\mathrm{RESULT} = \frac{\mathrm{FL}[7:0]}{\mathrm{F}[15+8]+1}$
If RESULT > 96 and the IP Stretch Enable bit in the Network Configuration Register (GMAC_NCFGR.IPGSEN) is written to '1', RESULT is used for the transmit inter-packet-gap.

### 38.8.27 GMAC Stacked VLAN Register

**Name:** GMAC_SVLAN
**Offset:** 0x0C0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ESVLAN | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | VLAN_TYPE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VLAN_TYPE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ESVLAN** Enable Stacked VLAN Processing Mode
0: Disable the stacked VLAN processing mode
1: Enable the stacked VLAN processing mode

| Value | Description |
|---|---|
| 0 | Stacked VLAN Processing disabled |
| 1 | Stacked VLAN Processing enabled |

**Bits 15:0 – VLAN_TYPE[15:0]** User Defined VLAN_TYPE Field
When Stacked VLAN is enabled (ESVLAN=1), the first VLAN tag in a received frame will only be accepted if the VLAN type field is equal to this user defined VLAN_TYPE, OR equal to the standard VLAN type (0x8100).
**Note:** The second VLAN tag of a Stacked VLAN packet will only be matched correctly if its VLAN_TYPE field equals 0x8100.

### 38.8.28 GMAC Transmit PFC Pause Register

**Name:** GMAC_TPFCP
**Offset:** 0x0C4
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PQ[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PEV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:8 – PQ[7:0]** Pause Quantum
When the Remove FCS bit in the GMAC Network Configuration register (GMAC_NCFGR.RFCS) is written to '1', and one or more bits in this bit field are written to '0', the associated PFC pause frame's pause quantum field value is taken from the Transmit Pause Quantum register (GMAC_TPQ).
For each entry equal to '1' in this bit field, the pause quantum associated with that entry will be zero.

**Bits 7:0 – PEV[7:0]** Priority Enable Vector
When the Remove FCS bit in the GMAC Network Configuration register (GMAC_NCFGR.RFCS) is written to '1', the priority enable vector of the PFC priority-based pause frame is set to the value stored in this bit field.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.29 GMAC Specific Address 1 Mask Bottom

**Name:** GMAC_SAMB1
**Offset:** 0x0C8
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Specific Address 1 Mask
Setting a bit to '1' masks the corresponding bit in the Specific Address 1 Bottom register (GMAC_SAB1).

### 38.8.30 GMAC Specific Address Mask 1 Top

**Name:** GMAC_SAMT1
**Offset:** 0x0CC
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – ADDR[15:0]** Specific Address 1 Mask
Setting a bit to '1' masks the corresponding bit in the Specific Address 1 register GMAC_SAT1.

### 38.8.31 GMAC 1588 Timer Nanosecond Comparison Register

**Name:** GMAC_NSC
**Offset:** 0x0DC
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | NANOSEC[21:16] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NANOSEC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NANOSEC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 21:0 – NANOSEC[21:0]** 1588 Timer Nanosecond Comparison Value
Value is compared to the bits [45:24] of the TSU timer count value (upper 22 bits of nanosecond value).

### 38.8.32 GMAC 1588 Timer Second Comparison Low Register

**Name:** GMAC_SCL
**Offset:** 0x0E0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SEC[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SEC[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SEC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SEC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – SEC[31:0]** 1588 Timer Second Comparison Value
Value is compared to seconds value bits [31:0] of the TSU timer count value.

### 38.8.33 GMAC 1588 Timer Second Comparison High Register

**Name:** GMAC_SCH
**Offset:** 0x0E4
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SEC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SEC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – SEC[15:0]** 1588 Timer Second Comparison Value
Value is compared to the top 16 bits (most significant 16 bits [47:32] of seconds value) of the TSU timer count value.

### 38.8.34 GMAC PTP Event Frame Transmitted Seconds High Register

**Name:** GMAC_EFTSH
**Offset:** 0x0E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RUD[15:0]** Register Update
The register is updated with the value that the IEEE 1588 timer seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.35 GMAC PTP Event Frame Received Seconds High Register

**Name:** GMAC_EFRSH
**Offset:** 0x0EC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RUD[15:0]** Register Update
The register is updated with the value that the IEEE 1588 timer seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.36 GMAC PTP Peer Event Frame Transmitted Seconds High Register

**Name:** GMAC_PEFTSH
**Offset:** 0x0F0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RUD[15:0]** Register Update
The register is updated with the value that the IEEE 1588 timer seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.37 GMAC PTP Peer Event Frame Received Seconds High Register

**Name:** GMAC_PEFRSH
**Offset:** 0x0F4
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RUD[15:0]** Register Update
The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.38 GMAC Octets Transmitted Low Register

**Name:** GMAC_OTLO
**Offset:** 0x100
**Reset:** 0x00000000
**Property:** -

When reading the Octets Transmitted and Octets Received Registers, bits [31:0] should be read prior to bits [47:32] to ensure reliable operation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TXO[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXO[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXO[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXO[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – TXO[31:0]** Transmitted Octets
Transmitted octets in valid frames of any type without errors, bits [31:0]. This counter is 48-bits, and is read through two registers. This count does not include octets from automatically generated pause frames.

### 38.8.39 GMAC Octets Transmitted High Register

**Name:** GMAC_OTHI
**Offset:** 0x104
**Reset:** 0x00000000
**Property:** -

When reading the Octets Transmitted and Octets Received Registers, bits [31:0] should be read prior to bits [47:32] to ensure reliable operation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXO[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXO[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TXO[15:0]** Transmitted Octets
Transmitted octets in valid frames of any type without errors, bits [47:32]. This counter is 48-bits, and is read through two registers. This count does not include octets from automatically generated pause frames.

### 38.8.40 GMAC Frames Transmitted

**Name:** GMAC_FT
**Offset:** 0x108
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – FTX[31:0]** Frames Transmitted without Error
Frames transmitted without error. This register counts the number of frames successfully transmitted, i.e., no underrun and not too many retries. Excludes pause frames.

### 38.8.41 GMAC Broadcast Frames Transmitted Register

**Name:** GMAC_BCFT
**Offset:** 0x10C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BFTX[31:0]** Broadcast Frames Transmitted without Error
This register counts the number of broadcast frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

### 38.8.42 GMAC Multicast Frames Transmitted Register

**Name:** GMAC_MFT
**Offset:** 0x110
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MFTX[31:0]** Multicast Frames Transmitted without Error
This register counts the number of multicast frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

### 38.8.43 GMAC Pause Frames Transmitted Register

**Name:** GMAC_PFT
**Offset:** 0x114
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – PFTX[15:0]** Pause Frames Transmitted Register
This register counts the number of pause frames transmitted. Only pause frames triggered by the register interface or through the external pause pins are counted as pause frames. Pause frames received through the FIFO interface are counted in the frames transmitted counter.

### 38.8.44 GMAC 64 Byte Frames Transmitted Register

**Name:** GMAC_BFT64
**Offset:** 0x118
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 64 Byte Frames Transmitted without Error
This register counts the number of 64 byte frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

### 38.8.45 GMAC 65 to 127 Byte Frames Transmitted Register

**Name:** GMAC_TBFT127
**Offset:** 0x11C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 65 to 127 Byte Frames Transmitted without Error
This register counts the number of 65 to 127 byte frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

### 38.8.46 GMAC 128 to 255 Byte Frames Transmitted Register

**Name:** GMAC_TBFT255
**Offset:** 0x120
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 128 to 255 Byte Frames Transmitted without Error
This register counts the number of 128 to 255 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

### 38.8.47 GMAC 256 to 511 Byte Frames Transmitted Register

**Name:** GMAC_TBFT511
**Offset:** 0x124
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 256 to 511 Byte Frames Transmitted without Error
This register counts the number of 256 to 511 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

### 38.8.48 GMAC 512 to 1023 Byte Frames Transmitted Register

**Name:** GMAC_TBFT1023
**Offset:** 0x128
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 512 to 1023 Byte Frames Transmitted without Error
This register counts the number of 512 to 1023 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

### 38.8.49 GMAC 1024 to 1518 Byte Frames Transmitted Register

**Name:** GMAC_TBFT1518
**Offset:** 0x12C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** 1024 to 1518 Byte Frames Transmitted without Error
This register counts the number of 1024 to 1518 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

### 38.8.50 GMAC Greater Than 1518 Byte Frames Transmitted Register

**Name:** GMAC_GTBFT1518
**Offset:** 0x130
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFTX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFTX[31:0]** Greater than 1518 Byte Frames Transmitted without Error
This register counts the number of 1518 or above byte frames successfully transmitted without error i.e., no underrun and not too many retries.

### 38.8.51 GMAC Transmit Underruns Register

**Name:** GMAC_TUR
**Offset:** 0x134
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | TXUNR[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXUNR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – TXUNR[9:0]** Transmit Underruns
This register counts the number of frames not transmitted due to a transmit underrun. If this register is incremented then no other statistics register is incremented.

### 38.8.52 GMAC Single Collision Frames Register

**Name:** GMAC_SCF
**Offset:** 0x138
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SCOL[17:16] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCOL[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SCOL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17:0 – SCOL[17:0]** Single Collision
This register counts the number of frames experiencing a single collision before being successfully transmitted i.e., no underrun.

### 38.8.53 GMAC Multiple Collision Frames Register

**Name:** GMAC_MCF
**Offset:** 0x13C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | MCOL[17:16] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MCOL[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MCOL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17:0 – MCOL[17:0]** Multiple Collision
This register counts the number of frames experiencing between two and fifteen collisions prior to being successfully transmitted, i.e., no underrun and not too many retries.

### 38.8.54 GMAC Excessive Collisions Register

**Name:** GMAC_EC
**Offset:** 0x140
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | XCOL[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | XCOL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – XCOL[9:0]** Excessive Collisions
This register counts the number of frames that failed to be transmitted because they experienced 16 collisions.

### 38.8.55 GMAC Late Collisions Register

**Name:** GMAC_LC
**Offset:** 0x144
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | LCOL[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LCOL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – LCOL[9:0]** Late Collisions
This register counts the number of late collisions occurring after the slot time (512 bits) has expired. In 10/100 mode, late collisions are counted twice i.e., both as a collision and a late collision.

### 38.8.56 GMAC Deferred Transmission Frames Register

**Name:** GMAC_DTF
**Offset:** 0x148
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DEFT[17:16] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DEFT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DEFT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17:0 – DEFT[17:0]** Deferred Transmission
This register counts the number of frames experiencing deferral due to carrier sense being active on their first attempt at transmission. Frames involved in any collision are not counted nor are frames that experienced a transmit underrun.

### 38.8.57 GMAC Carrier Sense Errors Register

**Name:** GMAC_CSE
**Offset:** 0x14C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CSR[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – CSR[9:0]** Carrier Sense Error
This register counts the number of frames transmitted with carrier sense was not seen during transmission or where carrier sense was de-asserted after being asserted in a transmit frame without collision (no underrun). Only incremented in half duplex mode. The only effect of a carrier sense error is to increment this register. The behavior of the other statistics registers is unaffected by the detection of a carrier sense error.

### 38.8.58 GMAC Octets Received Low Register

**Name:** GMAC_ORLO
**Offset:** 0x150
**Reset:** 0x00000000
**Property:** -

When reading the Octets Transmitted and Octets Received Registers, bits [31:0] should be read prior to bits [47:32] to ensure reliable operation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXO[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RXO[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXO[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXO[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RXO[31:0]** Received Octets
Received octets in frame without errors [31:0]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.59 GMAC Octets Received High Register

**Name:** GMAC_ORHI
**Offset:** 0x154
**Reset:** 0x00000000
**Property:** -

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXO[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXO[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RXO[15:0]** Received Octets
Received octets in frame without errors [47:32]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.60 GMAC Frames Received Register

**Name:** GMAC_FR
**Offset:** 0x158
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – FRX[31:0]** Frames Received without Error
This bit field counts the number of frames successfully received, excluding pause frames. It is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.61 GMAC Broadcast Frames Received Register

**Name:** GMAC_BCFR
**Offset:** 0x15C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BFRX[31:0]** Broadcast Frames Received without Error
Broadcast frames received without error. This bit field counts the number of broadcast frames successfully received. This excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.62 GMAC Multicast Frames Received Register

**Name:** GMAC_MFR
**Offset:** 0x160
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MFRX[31:0]** Multicast Frames Received without Error
This register counts the number of multicast frames successfully received without error, excluding pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.63 GMAC Pause Frames Received Register

**Name:** GMAC_PFR
**Offset:** 0x164
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – PFRX[15:0]** Pause Frames Received Register
This register counts the number of pause frames received without error.

### 38.8.64 GMAC 64 Byte Frames Received Register

**Name:** GMAC_BFR64
**Offset:** 0x168
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 64 Byte Frames Received without Error
This bit field counts the number of 64 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.65 GMAC 65 to 127 Byte Frames Received Register

**Name:** GMAC_TBFR127
**Offset:** 0x16C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 65 to 127 Byte Frames Received without Error
This bit field counts the number of 65 to 127 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.66 GMAC 128 to 255 Byte Frames Received Register

**Name:** GMAC_TBFR255
**Offset:** 0x170
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 128 to 255 Byte Frames Received without Error
This bit field counts the number of 128 to 255 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.67 GMAC 256 to 511 Byte Frames Received Register

**Name:** GMAC_TBFR511
**Offset:** 0x174
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 256 to 511 Byte Frames Received without Error
This bit fields counts the number of 256 to 511 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.68 GMAC 512 to 1023 Byte Frames Received Register

**Name:** GMAC_TBFR1023
**Offset:** 0x178
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 512 to 1023 Byte Frames Received without Error
This bit field counts the number of 512 to 1023 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

### 38.8.69 GMAC 1024 to 1518 Byte Frames Received Register

**Name:** GMAC_TBFR1518
**Offset:** 0x17C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 1024 to 1518 Byte Frames Received without Error
This bit field counts the number of 1024 to 1518 byte frames successfully received without error, i.e., no underrun and not too many retries.

### 38.8.70 GMAC 1519 to Maximum Byte Frames Received Register

**Name:** GMAC_TMXBFR
**Offset:** 0x180
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NFRX[31:0]** 1519 to Maximum Byte Frames Received without Error
This bit field counts the number of 1519 Byte or above frames successfully received without error. Maximum frame size is determined by the Maximum Frame Size bit (MAXFS, 1536 Bytes) or Jumbo Frame Size bit (JFRAME, 10240 Bytes) in the Network Configuration Register (GMAC_NCFGR). Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.71 GMAC Undersized Frames Received Register

**Name:** GMAC_UFR
**Offset:** 0x184
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | UFRX[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – UFRX[9:0]** Undersize Frames Received
This bit field counts the number of frames received less than 64 bytes in length (10/100 mode, full duplex) that do not have either a CRC error or an alignment error.

### 38.8.72 GMAC Oversized Frames Received Register

**Name:** GMAC_OFR
**Offset:** 0x188
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | OFRX[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OFRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – OFRX[9:0]** Oversized Frames Received
This pit field counts the number of frames received exceeding 1518 Bytes in length (1536 Bytes if GMAC_NCFGR.MAXFS is written to '1') but do not have either a CRC error, an alignment error, nor a receive symbol error.

### 38.8.73 GMAC Jabbers Received Register

**Name:** GMAC_JR
**Offset:** 0x18C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | JRX[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | JRX[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – JRX[9:0]** Jabbers Received
This bit field counts the number of frames received exceeding 1518 Bytes in length (1536 Bytes if GMAC_NCFGR.MAXFS is written to '1') and have either a CRC error, an alignment error or a receive symbol error.

### 38.8.74 GMAC Frame Check Sequence Errors Register

**Name:** GMAC_FCSE
**Offset:** 0x190
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | FCKR[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FCKR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – FCKR[9:0]** Frame Check Sequence Errors
The register counts frames that are an integral number of bytes, have bad CRC and are between 64 and 1518 bytes in length (1536 Bytes if GMAC_NCFGR.MAXFS is written to '1'). This register is also incremented if a symbol error is detected and the frame is of valid length and has an integral number of bytes.
This register is incremented for a frame with bad FCS, regardless of whether it is copied to memory due to ignore FCS mode (enabled by writing GMAC_NCFGR.IRXFCS=1).

### 38.8.75 GMAC Length Field Frame Errors Register

**Name:** GMAC_LFFE
**Offset:** 0x194
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | LFER[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LFER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – LFER[9:0]** Length Field Frame Errors
This bit field counts the number of frames received that have a measured length shorter than that extracted from the length field (Bytes 13 and 14). This condition is only counted if the value of the length field is less than 0x0600, the frame is not of excessive length and checking is enabled by writing a '1' to the Length Field Error Frame Discard bit in the Network Configuration Register (GMAC_NCFGR.LFERD).

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.76 GMAC Receive Symbol Errors Register

**Name:** GMAC_RSE
**Offset:** 0x198
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXSE[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXSE[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – RXSE[9:0]** Receive Symbol Errors
This bit field counts the number of frames that had GRXER asserted during reception. For 10/100 mode symbol errors are counted regardless of frame length checks. Receive symbol errors will also be counted as an FCS or alignment error if the frame is between 64 and 1518 Bytes (1536 Bytes if GMAC_NCFGR.MAXFS=1). If the frame is larger it will be recorded as a jabber error.

### 38.8.77 GMAC Alignment Errors Register

**Name:** GMAC_AE
**Offset:** 0x19C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | AER[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – AER[9:0]** Alignment Errors
This bit field counts the frames that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of Bytes and are between 64 and 1518 Bytes in length (1536 if GMAC_NCFGR.MAXFS=1). This register is also incremented if a symbol error is detected and the frame is of valid length and does not have an integral number of bytes.

### 38.8.78 GMAC Receive Resource Errors Register

**Name:** GMAC_RRE
**Offset:** 0x1A0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXRER[17:16] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXRER[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXRER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 17:0 – RXRER[17:0]** Receive Resource Errors
This bit field counts frames that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of Bytes and are between 64 and 1518 Bytes in length (1536 if GMAC_NCFGR.MAXFS=1). This bit field is also incremented if a symbol error is detected and the frame is of valid length and does not have an integral number of Bytes.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.79 GMAC Receive Overruns Register

**Name:** GMAC_ROE
**Offset:** 0x1A4
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXOVR[9:8] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXOVR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – RXOVR[9:0]** Receive Overruns
This bit field counts the number of frames that are address recognized but were not copied to memory due to a receive overrun.

### 38.8.80 GMAC IP Header Checksum Errors Register

**Name:** GMAC_IHCE
**Offset:** 0x1A8
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HCKER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – HCKER[7:0]** IP Header Checksum Errors
This register counts the number of frames discarded due to an incorrect IP header checksum, but are between 64 and 1518 Bytes (1536 Bytes if GMAC_NCFGR.MAXFS=1) and do not have a CRC error, an alignment error, nor a symbol error.

### 38.8.81 GMAC TCP Checksum Errors Register

**Name:** GMAC_TCE
**Offset:** 0x1AC
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCKER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TCKER[7:0]** TCP Checksum Errors
This register counts the number of frames discarded due to an incorrect TCP checksum, but are between 64 and 1518 Bytes (1536 Bytes if GMAC_NCFGR.MAXFS=1) and do not have a CRC error, an alignment error, nor a symbol error.

### 38.8.82 GMAC UDP Checksum Errors Register

**Name:** GMAC_UCE
**Offset:** 0x1B0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UCKER[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – UCKER[7:0]** UDP Checksum Errors
This register counts the number of frames discarded due to an incorrect UDP checksum, but are between 64 and 1518 Bytes (1536 Bytes if GMAC_NCFGR.MAXFS=1) and do not have a CRC error, an alignment error, nor a symbol error.

### 38.8.83 GMAC 1588 Timer Increment Sub-nanoseconds Register

**Name:** GMAC_TISUBN
**Offset:** 0x1BC
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LSBTIR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LSBTIR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – LSBTIR[15:0]** Lower Significant Bits of Timer Increment Register
Lower significant bits of Timer Increment Register [15:0], giving a 24-bit timer_increment counter. These bits are the sub-ns value which the 1588 timer will be incremented each clock cycle. Bit n = $2^{(n-16)}$ ns giving a resolution of approximately 15.2E$^{-15}$ sec.

### 38.8.84 GMAC 1588 Timer Seconds High Register

**Name:** GMAC_TSH
**Offset:** 0x1C0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TCS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TCS[15:0]** Timer Count in Seconds
This register is writable. It increments by 1 when the IEEE 1588 nanoseconds counter counts to one second. It may also be incremented when the Timer Adjust Register is written.

### 38.8.85 GMAC 1588 Timer Seconds Low Register

**Name:** GMAC_TSL
**Offset:** 0x1D0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TCS[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TCS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TCS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – TCS[31:0]** Timer Count in Seconds
This register is writable. It increments by 1 when the IEEE 1588 nanoseconds counter counts to one second. It may also be incremented when the Timer Adjust Register is written.

### 38.8.86 GMAC 1588 Timer Nanoseconds Register

**Name:** GMAC_TN
**Offset:** 0x1D4
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | TNS[29:24] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TNS[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TNS[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TNS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – TNS[29:0]** Timer Count in Nanoseconds
This register is writable. It can also be adjusted by writes to the IEEE 1588 Timer Adjust Register. It increments by the value of the IEEE 1588 Timer Increment Register each clock cycle.

### 38.8.87 GMAC 1588 Timer Adjust Register

**Name:** GMAC_TA
**Offset:** 0x1D8
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADJ | | ITDT[29:24] | | | | | |
| Access | W | | W | W | W | W | W | W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ITDT[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ITDT[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ITDT[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ADJ** Adjust 1588 Timer
Write as '1' to subtract from the 1588 timer. Write as '0' to add to it.

**Bits 29:0 – ITDT[29:0]** Increment/Decrement
The number of nanoseconds to increment or decrement the IEEE 1588 Timer Nanoseconds Register. If necessary, the IEEE 1588 Seconds Register will be incremented or decremented.

### 38.8.88 GMAC IEEE 1588 Timer Increment Register

**Name:** GMAC_TI
**Offset:** 0x1DC
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NIT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ACNS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CNS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – NIT[7:0]** Number of Increments
The number of increments after which the alternative increment is used.

**Bits 15:8 – ACNS[7:0]** Alternative Count Nanoseconds
Alternative count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

**Bits 7:0 – CNS[7:0]** Count Nanoseconds
A count of nanoseconds by which the IEEE 1588 Timer Nanoseconds Register will be incremented each clock cycle.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.89 GMAC PTP Event Frame Transmitted Seconds Low Register

**Name:** GMAC_EFTSL
**Offset:** 0x1E0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RUD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RUD[31:0]** Register Update
The register is updated with the value that the IEEE 1588 Timer Seconds Register holds when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.90 GMAC PTP Event Frame Transmitted Nanoseconds Register

**Name:** GMAC_EFTN
**Offset:** 0x1E4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RUD[29:24] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – RUD[29:0]** Register Update
The register is updated with the value that the IEEE 1588 Timer Nanoseconds Register holds when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the bit field is updated.

### 38.8.91 GMAC PTP Event Frame Received Seconds Low Register

**Name:** GMAC_EFRSL
**Offset:** 0x1E8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RUD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RUD[31:0]** Register Update
The register is updated with the value that the IEEE 1588 Timer Seconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.92 GMAC PTP Event Frame Received Nanoseconds Register

**Name:** GMAC_EFRN
**Offset:** 0x1EC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RUD[29:24] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – RUD[29:0]** Register Update
The register is updated with the value that the IEEE 1588 Timer Nanoseconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.93 GMAC PTP Peer Event Frame Transmitted Seconds Low Register

**Name:** GMAC_PEFTSL
**Offset:** 0x1F0
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RUD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RUD[31:0]** Register Update
The register is updated with the value that the IEEE 1588 Timer Seconds Register holds when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.94 GMAC PTP Peer Event Frame Transmitted Nanoseconds Register

**Name:** GMAC_PEFTN
**Offset:** 0x1F4
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RUD[29:24] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – RUD[29:0]** Register Update
The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.95 GMAC PTP Peer Event Frame Received Seconds Low Register

**Name:** GMAC_PEFRSL
**Offset:** 0x1F8
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RUD[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RUD[31:0]** Register Update
The register is updated with the value that the IEEE 1588 Timer Seconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.96 GMAC PTP Peer Event Frame Received Nanoseconds Register

**Name:** GMAC_PEFRN
**Offset:** 0x1FC
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | RUD[29:24] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RUD[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RUD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:0 – RUD[29:0]** Register Update
The register is updated with the value that the IEEE 1588 Timer Nanoseconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

### 38.8.97 GMAC Received LPI Transitions

**Name:** GMAC_RXLPI
**Offset:** 0x270
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – COUNT[15:0]** Count of Received LPI Transitions
A count of the number of times there is a transition from receiving normal idle to receiving low power idle. Cleared on read.

### 38.8.98 GMAC Received LPI Time

**Name:** GMAC_RXLPITIME
**Offset:** 0x274
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – LPITIME[23:0]** Time in LPI
This field increments once every 16 MCK cycles when the bit RXLPIS (LPI Indication (bit 7)) is set in the GMAC_NSR.
Cleared on read.

### 38.8.99 GMAC Transmit LPI Transitions

**Name:** GMAC_TXLPI
**Offset:** 0x278
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COUNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – COUNT[23:0]** Count of LIP Transitions
A count of the number of times the bit TXLPIEN (Enable LPI Transmission (bit 19)) goes from low to high in the GMAC_NCR.
Cleared on read.

### 38.8.100 GMAC Transmit LPI Time

**Name:** GMAC_TXLPITIME
**Offset:** 0x27C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LPITIME[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – LPITIME[23:0]** Time in LPI
This field increments once every 16 MCK cycles when the bit TXLPIEN (Enable LPI Transmission (bit 19)) is set in GMAC_NCR.
Cleared on read.

### 38.8.101 GMAC Interrupt Status Register Priority Queue x

**Name:** GMAC_ISRPQx
**Offset:** 0x0400 + (x-1)*0x04 [x=1..5]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HRESP | ROVR | | |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | 0 | 0 | |

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error
Transmit frame corruption due to AHB error—set if an error occurs whilst midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame.

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.102 GMAC Transmit Buffer Queue Base Address Register Priority Queue x

**Name:** GMAC_TBQBAPQx
**Offset:** 0x0440 + (x-1)*0x04 [x=1..5]
**Reset:** 0x00000000
**Property:** Read/Write

These registers hold the start address of the transmit buffer queues (transmit buffers descriptor lists) for the additional queues and must be initialized to the address of valid descriptors, even if the priority queues are not used.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TXBQBA[29:22] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TXBQBA[21:14] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TXBQBA[13:6] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXBQBA[5:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – TXBQBA[29:0]** Transmit Buffer Queue Base Address
Contains the address of the start of the transmit queue.

### 38.8.103 GMAC Receive Buffer Queue Base Address Register Priority Queue x

**Name:** GMAC_RBQBAPQx
**Offset:** 0x0480 + (x-1)*0x04 [x=1..5]
**Reset:** 0x00000000
**Property:** Read/Write

These registers hold the start address of the receive buffer queues (receive buffers descriptor lists) for the additional queues used when priority queues are employed.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXBQBA[29:22] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RXBQBA[21:14] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXBQBA[13:6] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXBQBA[5:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 31:2 – RXBQBA[29:0]** Receive Buffer Queue Base Address
Holds the address of the start of the receive queue.

### 38.8.104 GMAC Receive Buffer Size Register Priority Queue x

**Name:** GMAC_RBSRPQx
**Offset:** 0x04A0 + (x-1)*0x04 [x=1..5]
**Reset:** 0x00000002
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RBS[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RBS[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

**Bits 15:0 – RBS[15:0]** Receive Buffer Size
DMA receive buffer size in AHB system memory. The value defined by these bits determines the size of buffer to use in main AHB system memory when writing received data.
The value is defined in multiples of 64 Bytes such that a value of 0x01 corresponds to buffers of 64 Bytes, 0x02 corresponds to 128 Bytes etc.
Examples:

- 0x18: 1536 Bytes (1 × max length frame/buffer)
- 0xA0: 10240 Bytes (1 × 10K jumbo frame/buffer)

**Note:** This value should never be written as zero.

### 38.8.105 GMAC Credit-Based Shaping Control Register

**Name:** GMAC_CBSCR
**Offset:** 0x4BC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | QAE | QBE |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – QAE** Queue A CBS Enable

| Value | Description |
|---|---|
| 0 | Credit-based shaping on the second highest priority queue (queue A) is disabled. |
| 1 | Credit-based shaping on the second highest priority queue (queue A) is enabled. |

**Bit 0 – QBE** Queue B CBS Enable

| Value | Description |
|---|---|
| 0 | Credit-based shaping on the highest priority queue (queue B) is disabled. |
| 1 | Credit-based shaping on the highest priority queue (queue B) is enabled. |

### 38.8.106 GMAC Credit-Based Shaping IdleSlope Register for Queue A

**Name:** GMAC_CBSISQA
**Offset:** 0x4C0
**Reset:** 0x00000000
**Property:** Read/Write

Credit-based shaping must be disabled in the GMAC_CBSCR before updating this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IS[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IS[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IS[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IS[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – IS[31:0]** IdleSlope
IdleSlope value for queue A in Bytes per second.
The IdleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must not exceed the port transmit rate which is dependent on the speed of operation, e.g., 100 Mb/second = 32'h017D7840. If 50% of bandwidth was to be allocated to a particular queue in 100 Mb/second mode, then the IdleSlope value for that queue would be calculated as 32'h017D7840 / 2.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.107 GMAC Credit-Based Shaping IdleSlope Register for Queue B

**Name:** GMAC_CBSISQB
**Offset:** 0x4C4
**Reset:** 0x00000000
**Property:** Read/Write

Credit-based shaping must be disabled in the GMAC_CBSCR before updating this register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IS[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IS[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IS[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IS[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – IS[31:0]** IdleSlope
IdleSlope value for queue B in bytes/second.
The IdleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must not exceed the port transmit rate which is dependent on the speed of operation, e.g., 100 Mb/second = 32'h017D7840. If 50% of bandwidth was to be allocated to a particular queue in 100 Mb/sec mode, then the IdleSlope value for that queue would be calculated as 32'h017D7840 / 2

### 38.8.108 GMAC Screening Type 1 Register x Priority Queue

**Name:** GMAC_ST1RPQx
**Offset:** 0x0500 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

Screening type 1 registers are used to allocate up to 6 priority queues to received frames based on certain IP or UDP fields of incoming frames.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | UDPE | DSTCE | UDPM[15:12] | | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | UDPM[11:4] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UDPM[3:0] | | | | DSTCM[7:4] | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DSTCM[3:0] | | | | | QNB[2:0] | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 29 – UDPE** UDP Port Match Enable
When this bit is written to '1', the UDP Destination Port of the received UDP frame is matched against the value stored in the bit field UDPM.

**Bit 28 – DSTCE** Differentiated Services or Traffic Class Match Enable
When this bit is written to '1', the DS (differentiated services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers are matched against the value stored in bit field DSTCM.

**Bits 27:12 – UDPM[15:0]** UDP Port Match
When UDP port match enable is set (UDPME=1), the UDP Destination Port of the received UDP frame is matched against this bit field.

**Bits 11:4 – DSTCM[7:0]** Differentiated Services or Traffic Class Match
When DS/TC match enable is set (DSTCE), the DS (differentiated services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers are matched against this bit field.

**Bits 2:0 – QNB[2:0]** Queue Number
If a match is successful, then the queue value programmed in this bit field is allocated to the frame.

### 38.8.109 GMAC Screening Type 2 Register x Priority Queue

**Name:** GMAC_ST2RPQx
**Offset:** 0x0540 + x*0x04 [x=0..7]
**Reset:** 0x00000000
**Property:** Read/Write

Screening type 2 registers are used to allocate up to 6 priority queues to received frames based on the VLAN priority field of received Ethernet frames.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | COMPCE | COMPC[4:0] | | | | | COMPBE |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COMPB[4:0] | | | | | COMPAE | COMPA[4:3] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COMPA[2:0] | | | ETHE | I2ETH[2:0] | | | VLANE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | VLANP[2:0] | | | | QNB[2:0] | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 30 – COMPCE** Compare C Enable

| Value | Description |
|---|---|
| 0 | Compare C is disabled. |
| 1 | Comparison via the register designated by index COMPC is enabled. |

**Bits 29:25 – COMPC[4:0]** Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPC is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPCE=1, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

**Bit 24 – COMPBE** Compare B Enable

| Value | Description |
|---|---|
| 0 | Compare B is disabled. |
| 1 | Comparison via the register designated by index COMPB is enabled. |

**Bits 23:19 – COMPB[4:0]** Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPB is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPBE=1, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

**Bit 18 – COMPAE** Compare A Enable

| Value | Description |
|---|---|
| 0 | Compare A is disabled. |
| 1 | Comparison via the register designated by index COMPA is enabled. |

**Bits 17:13 – COMPA[4:0]** Index of Screening Type 2 Compare Word 0/Word 1 register x
COMPA is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPAE=1, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

**Bit 12 – ETHE** EtherType Enable

| Value | Description |
|---|---|
| 0 | EtherType match is disabled |
| 1 | EtherType match with bits [15:0] of the register designated by the value in I2ETH is enabled |

**Bits 11:9 – I2ETH[2:0]** Index of Screening Type 2 EtherType register x
When EtherType is enabled (ETHE=1), the EtherType field (last EtherType in the header if the frame is VLAN-tagged) is compared with bits [15:0] in the register designated by the value of this bit field.

**Bit 8 – VLANE** VLAN Enable

| Value | Description |
|---|---|
| 0 | VLAN match disabled |
| 1 | VLAN match is enabled |

**Bits 6:4 – VLANP[2:0]** VLAN Priority
When VLAN match is enabled (VLANE=1), the VLAN Priority field of the received frame is matched against the value of this bit field.

**Bits 2:0 – QNB[2:0]** Queue Number
If a match is successful, then the queue value programmed in QNB is allocated to the frame.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.110 GMAC Interrupt Enable Register Priority Queue x

**Name:** GMAC_IERPQx
**Offset:** 0x0600 + (x-1)*0x04 [x=1..5]
**Reset:** –
**Property:** Write-only

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | HRESP | ROVR | | |
| Access | | | | | W | W | | |
| Reset | | | | | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TCOMP | TFC | RLEX | | | RXUBR | RCOMP | |
| Access | W | W | W | | | W | W | |
| Reset | – | – | – | | | – | – | |

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

© 2021 Microchip Technology Inc.
and its subsidiaries

### 38.8.111 GMAC Interrupt Disable Register Priority Queue x

**Name:** GMAC_IDRPQx
**Offset:** 0x0620 + (x-1)*0x04 [x=1..5]
**Reset:** –
**Property:** Write-only

The following values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | W | W | | |
| Reset | | | | | – | – | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | W | W | W | | | W | W | |
| Reset | – | – | – | | | – | – | |

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – TFC** Transmit Frame Corruption Due to AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

### 38.8.112 GMAC Interrupt Mask Register Priority Queue x

**Name:** GMAC_IMRPQx
**Offset:** 0x0640 + (x-1)*0x04 [x=1..5]
**Reset:** 0x00000000
**Property:** Read/Write

A read of this register returns the value of the receive complete interrupt mask.

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Register, causing an interrupt to be generated if a '1' is written.

The following values are valid for all listed bit names of this register:

0: Corresponding interrupt is enabled.

1: Corresponding interrupt is disabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | HRESP | ROVR | | |
| Reset | | | | | 0 | 0 | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | TCOMP | AHB | RLEX | | | RXUBR | RCOMP | |
| Reset | 0 | 0 | 0 | | | 0 | 0 | |

**Bit 11 – HRESP** HRESP Not OK

**Bit 10 – ROVR** Receive Overrun

**Bit 7 – TCOMP** Transmit Complete

**Bit 6 – AHB** AHB Error

**Bit 5 – RLEX** Retry Limit Exceeded or Late Collision

**Bit 2 – RXUBR** RX Used Bit Read

**Bit 1 – RCOMP** Receive Complete

### 38.8.113 GMAC Screening Type 2 EtherType Register x

**Name:** GMAC_ST2ERx
**Offset:** 0x06E0 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | COMPVAL[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | COMPVAL[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – COMPVAL[15:0]** EtherType Compare Value
When the bit GMAC_ST2RPQ.ETHE is written to '1', the EtherType (last EtherType in the header if the frame is VLAN tagged) is compared with bits [15:0] in the register designated by GMAC_ST2RPQ.I2ETH.

### 38.8.114 GMAC Screening Type 2 Compare Word 0 Register x

**Name:** GMAC_ST2CW0x
**Offset:** 0x0700 + x*0x08 [x=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | COMPVAL[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | COMPVAL[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MASKVAL[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MASKVAL[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – COMPVAL[15:0]** Compare Value
The byte stored in bits [23:16] is compared against the first byte of the 2 bytes extracted from the frame.
The byte stored in bits [31:24] is compared against the second byte of the 2 bytes extracted from the frame.

**Bits 15:0 – MASKVAL[15:0]** Mask Value
The value of MASKVAL ANDed with the 2 bytes extracted from the frame is compared to the value of MASKVAL ANDed with the value of COMPVAL.

### 38.8.115 GMAC Screening Type 2 Compare Word 1 Register x

**Name:** GMAC_ST2CW1x
**Offset:** 0x0704 + x*0x08 [x=0..23]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | OFFSSTRT[1] |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | OFFSSTRT[0] | OFFSVAL[6:0] | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:7 – OFFSSTRT[1:0]** Ethernet Frame Offset Start

| Value | Name | Description |
|---|---|---|
| 0 | FRAMESTART | Offset from the start of the frame |
| 1 | ETHERTYPE | Offset from the byte after the EtherType field |
| 2 | IP | Offset from the byte after the IP header field |
| 3 | TCP_UDP | Offset from the byte after the TCP/UDP header field |

**Bits 6:0 – OFFSVAL[6:0]** Offset Value in Bytes
The value of OFFSVAL ranges from 0 to 127 bytes, and is counted from either the start of the frame, the byte after the EtherType field (last EtherType in the header if the frame is VLAN tagged), the byte after the IP header (IPv4 or IPv6) or the byte after the TCP/UDP header.

© 2021 Microchip Technology Inc.
and its subsidiaries

# 39. USB High Speed Interface (USBHS)

**Important:** This module is not supported. Refer to the section Errata for more information.

## 39.1 Description

The USB High-Speed Interface (USBHS) complies with the Universal Serial Bus (USB) 2.0 specification in all speeds.

Each pipe/endpoint can be configured in one of several USB transfer types. It can be associated with one, two or three banks of a DPRAM used to store the current data payload. If two or three banks are used, then one DPRAM bank is read or written by the CPU or the DMA, while the other is read or written by the USBHS core. This feature is mandatory for isochronous pipes/endpoints.

The following table describes the hardware configuration of the USB MCU device.

**Table 39-1. Description of USB Pipes/Endpoints**

| Pipe/Endpoint | Mnemonic | Max. Number Banks | DMA | High Band Width | Max. Pipe/ Endpoint Size | Type |
|---|---|---|---|---|---|---|
| 0 | PEP_0 | 1 | N | N | 64 | Control |
| 1 | PEP_1 | 3 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |
| 2 | PEP_2 | 3 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |
| 3 | PEP_3 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |
| 4 | PEP_4 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |
| 5 | PEP_5 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |
| 6 | PEP_6 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |
| 7 | PEP_7 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |
| 8 | PEP_8 | 2 | N | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |
| 9 | PEP_9 | 2 | N | Y | 1024 | Isochronous/Bulk/Interrupt/ Control |

## 39.2 Embedded Characteristics

- Compatible with the USB 2.0 Specification
- Supports High-Speed (480 Mbps), Full-Speed (12 Mbps) and Low-Speed (1.5 Mbps) Communication
- 10 Pipes/Endpoints

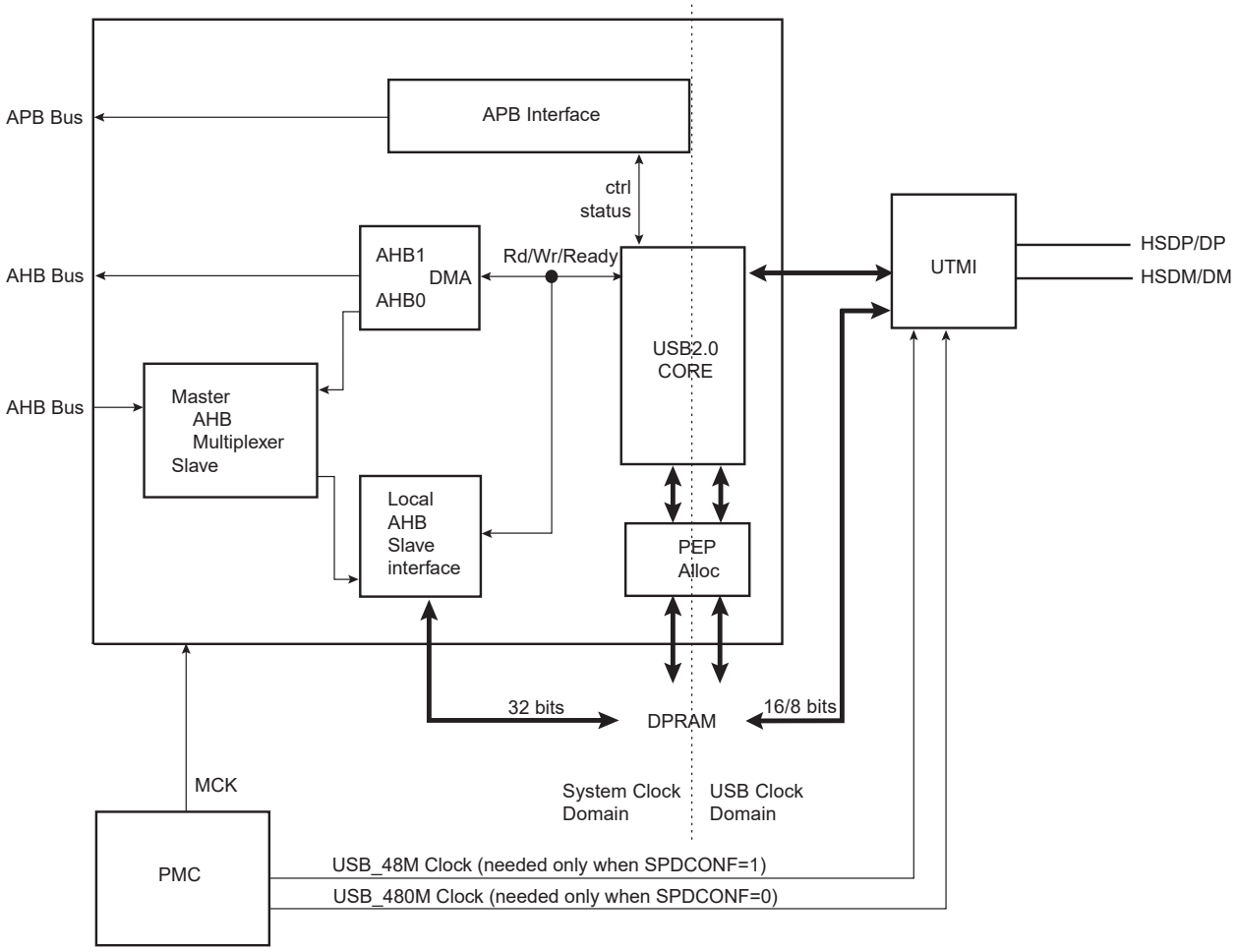
- 4096 bytes of Embedded Dual-Port RAM (DPRAM) for Pipes/Endpoints
- Up to 3 Memory Banks per Pipe/Endpoint (not for Control Pipe/Endpoint)
- Flexible Pipe/Endpoint Configuration and Management with Dedicated DMA Channels
- On-chip UTMI Transceiver including Pull-ups/Pull-downs

## 39.3 Block Diagram

The USBHS provides a hardware device to interface a USB link to a data flow stored in a dual-port RAM (DPRAM).

In normal operation (SPDCONF = 0), the UTMI transceiver requires the UTMI PLL (480 MHz). In case of full-speed or low-speed only, for a lower consumption (SPDCONF = 1), the UTMI transceiver only requires 48 MHz.

**Figure 39-1. USBHS Block Diagram**




### 39.3.1 Signal Description

**Table 39-2. Signal Description**

| Name | Description | Type |
|---|---|---|
| HSDM/DM | HS/FS Differential Data Line - | Input/Output |
| HSDP/DP | HS/FS Differential Data Line + | Input/Output |

## 39.4 Product Dependencies

### 39.4.1 I/O Lines

A regular PIO line must be used to control VBUS. This is configured in the I/O Controller.

### 39.4.2 Clocks

The clock for the USBHS bus interface is generated by the Power Management Controller. This clock can be enabled or disabled in the Power Management Controller. It is recommended to disable the USBHS before disabling the clock, to avoid freezing the USBHS in an undefined state.

Before enabling the USB clock in the Power Management Controller, the USBHS must be enabled (by writing a one to the USBHS_CTRL.USBE bit and a zero to the USBHS_CTRL.FRZCLK bit).

The USBHS can work in two modes:

- Normal mode (SPDCONF = 0) where High speed, Full speed and Low speed are available.
- Low-power mode (SPDCONF = 1) where Full speed and Low speed are available.

To ensure successful startup, follow the sequences below:

- In Normal mode:

1. Enable the USBHS peripheral clock. This is done via the register PMC_PCER.
2. Enable the USBHS (UIMOD, USBE = 1, FRZCLK = 0).
3. Enable the UPLL 480 MHz.
4. Wait for the UPLL 480 MHz to be considered as locked by the PMC.

- In Low-power mode:

1. As USB_48M must be set to 48 MHz (refer to the section “Power Management Controller (PMC)”), select either the PLLA or the UPLL (previously set to ON), and program the PMC_USB register (source selection and divider).
2. Enable the USBHS peripheral clock (PMC_PCER).
3. Put the USBHS in Low-power mode (SPDCONF = 1).
4. Enable the USBHS (UIMOD, USBE = 1, FRZCLK = 0).
5. Enable the USBCK bit (PMC_SCER).

**Related Links**

31. Power Management Controller (PMC)

### 39.4.3 Interrupt Sources

The USBHS interrupt request line is connected to the interrupt controller. Using the USBHS interrupt requires the interrupt controller to be programmed first.

### 39.4.4 USB Pipe/Endpoint x FIFO Data Register (USBFIFOxDATA)

The application has access to each pipe/endpoint FIFO through its reserved 32 KB address space. The application can access a 64-KB buffer linearly or fixedly as the DPRAM address increment is fully handled by hardware. Byte, half-word and word accesses are supported. Data should be accessed in a big-endian way.

Disabling the USBHS (by writing a zero to the USBHS_CTRL.USBE bit) does not reset the DPRAM.

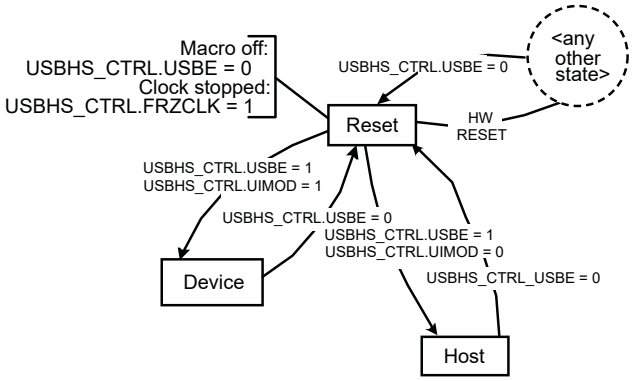
## 39.5 Functional Description

### 39.5.1 USB General Operation

#### 39.5.1.1 Power-On and Reset

The following figure describes the USBHS general states.

**Figure 39-2. General States**




After a hardware reset, the USBHS is in Reset state. In this state:

- The USBHS is disabled. The USBHS Enable bit in the General Control register (USBHS_CTRL.USBE) is zero.
- The USBHS clock is stopped in order to minimize power consumption. The Freeze USB Clock bit (USBHS_CTRL.FRZCLK) is set.
- The UTMI is in Suspend mode.
- The internal states and registers of the Device and Host modes are reset.
- The DPRAM is not cleared and is accessible.

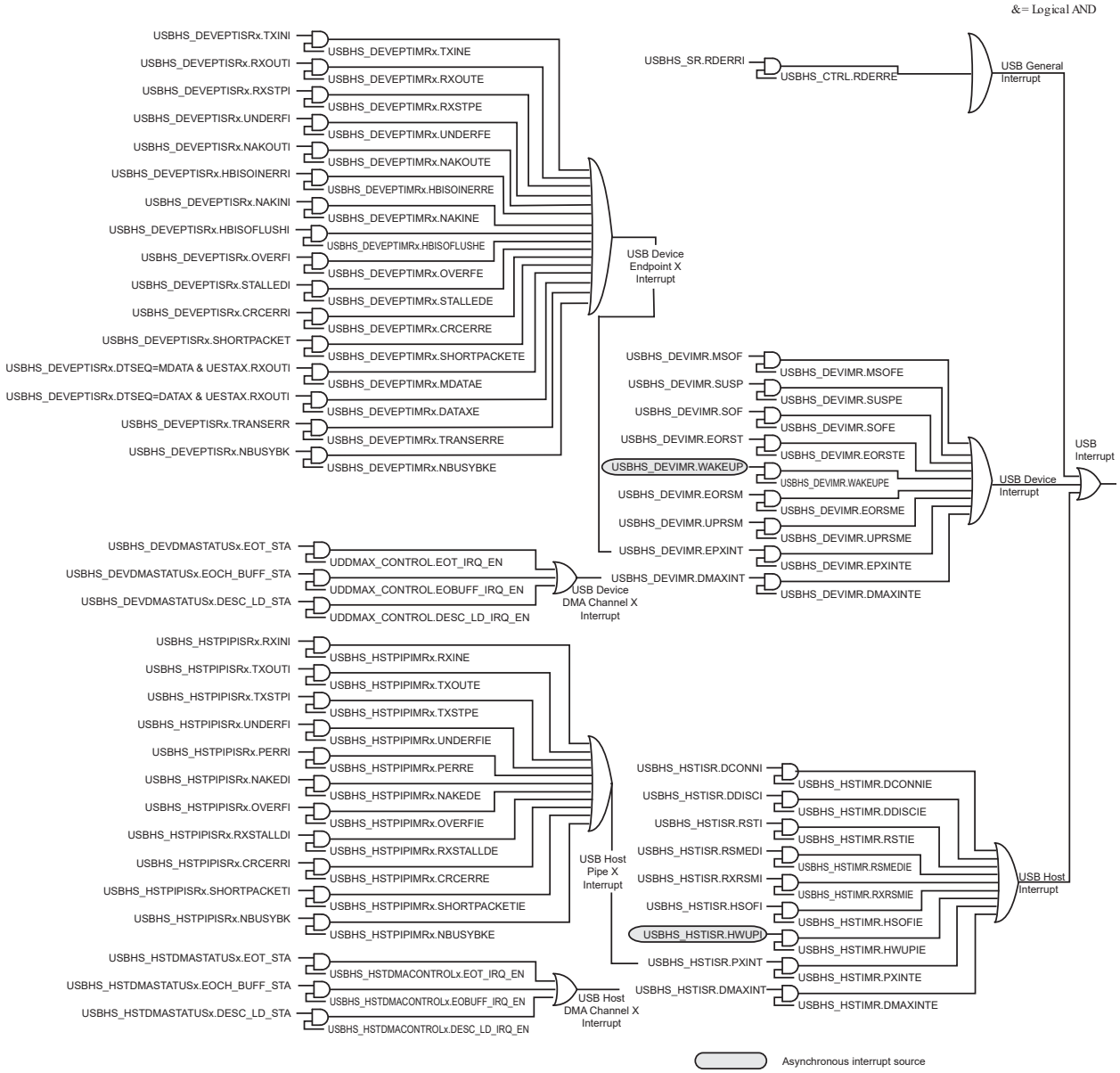
After writing a one to USBHS_CTRL.USBE, the USBHS enters the Device or the Host mode in idle state.

The USBHS can be disabled at any time by writing a zero to USBHS_CTRL.USBE. This acts as a hardware reset, except that the USBHS_CTRL.FRZCLK, USBHS_CTRL.UIMOD and USBHS_DEVCTRL.LS bits are not reset.

#### 39.5.1.2 Interrupts

One interrupt vector is assigned to the USB interface. The following figure shows the structure of the USB interrupt system.

**Figure 39-3. Interrupt System**




See Interrupts in the Device Operation section and Interrupts in the Host Operation section for further details about device and host interrupts.

There are two kinds of general interrupts: processing, i.e., their generation is part of the normal processing, and exception, i.e., errors (not related to CPU exceptions).

### 39.5.1.3 MCU Power Modes

USB Suspend Mode

In Peripheral mode, the Suspend Interrupt bit in the Device Global Interrupt Status register (USBHS_DEVISR.SUSP) indicates that the USB line is in Suspend mode. In this case, the transceiver is automatically set in Suspend mode to reduce consumption.

Clock Frozen

The USBHS can be frozen when the USB line is in the Suspend mode, by writing a one to the USBHS_CTRL.FRZCLK bit, which reduces power consumption.

In this case, it is still possible to access the following:

- USBHS_CTRL.FRZCLK, USBHS_CTRL.USBE and USBHS_DEVCTRL.LS bits

Moreover, when USBHS_CTRL.FRZCLK = 1, only the asynchronous interrupt sources can trigger the USB interrupt:

- Wakeup Interrupt (USBHS_DEVISR.WAKEUP)
- Host Wakeup Interrupt (USBHS_HSTISR.HWUPI)

#### 39.5.1.4 Speed Control

Device Mode

When the USB interface is in Device mode, the speed selection (Full-speed or High-speed) is performed automatically by the USBHS during the USB reset according to the host speed capability. At the end of the USB reset, the USBHS enables or disables high-speed terminations and pull-up.

It is possible to set the USBHS to Full-speed or Low-speed mode via USBHS_DEVCTRL.LS and USBHS_DEVCTRL.SPDCONF.

Host Mode

When the USB interface is in Host mode, internal pull-down resistors are connected on both D+ and D- and the interface detects the speed of the connected device, which is reflected by the Speed Status (USBHS_SR.SPEED) field.

#### 39.5.1.5 DPRAM Management

Pipes and endpoints can only be allocated in ascending order, from pipe/endpoint 0 to the last pipe/endpoint to be allocated. The user should therefore configure them in the same order.

The allocation of a pipe/endpoint x starts when the Endpoint Memory Allocate bit in the Endpoint x Configuration register (USBHS_DEVEPTCFGx.ALLOC) is written to one. Then, the hardware allocates a memory area in the DPRAM and inserts it between the x - 1 and x+ 1 pipes/endpoints. The x+ 1 pipe/endpoint memory window slides up and its data is lost. Note that the following pipe/endpoint memory windows (from x+ 2) do not slide.

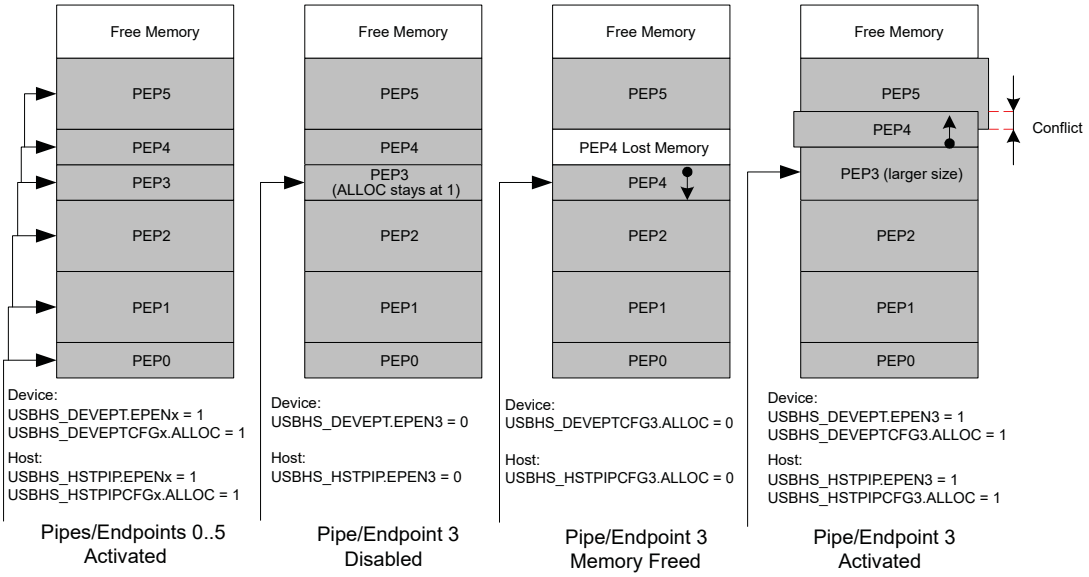
Disabling a pipe, by writing a zero to the Pipe x Enable bit in the Host Pipe register (USBHS_HSTPIP.PENx), or disabling an endpoint, by writing a zero to the Endpoint x Enable bit in the Device Endpoint register (USBHS_DEVEPT.EPENx), does not reset the USBHS_DEVEPTCFGx.ALLOC bit or the Pipe/Endpoint configuration:

- Pipe Configuration
  - Pipe Banks (USBHS_HSTPIPCFGx.PBK)
  - Pipe Size (USBHS_HSTPIPCFGx.PSIZE)
  - Pipe Token (USBHS_HSTPIPCFGx.PTOKEN)
  - Pipe Type (USBHS_HSTPIPCFGx.PTYPE)
  - Pipe Endpoint Number (USBHS_HSTPIPCFGx.PEPNUM)
  - Pipe Interrupt Request Frequency (USBHS_HSTPIPCFGx.INTFRQ)
- Endpoint Configuration
  - Endpoint Banks (USBHS_DEVEPTCFGx.EPBK)
  - Endpoint Size (USBHS_DEVEPTCFGx. EPSIZE)
  - Endpoint Direction (USBHS_DEVEPTCFGx.EPDIR)
  - Endpoint Type (USBHS_DEVEPTCFGx.EPTYPE)

To free endpoint memory, the user must write a zero to the USBHS_DEVEPTCFGx.ALLOC bit. The x+ 1 pipe/endpoint memory window then slides down and its data is lost. Note that the following pipe/endpoint memory windows (from x + 2) do not slide.

The following figure illustrates the allocation and reorganization of the DPRAM in a typical example.

**Figure 39-4. Allocation and Reorganization of the DPRAM**




1. Pipes/endpoints 0 to 5 are enabled, configured and allocated in ascending order. Each pipe/endpoint then owns a memory area in the DPRAM.
2. Pipe/endpoint 3 is disabled, but its memory is kept allocated by the controller.
3. In order to free its memory, its USBHS_DEVEPTCFGx.ALLOC bit is written to zero. The pipe/endpoint 4 memory window slides down, but pipe/endpoint 5 does not move.
4. If the user chooses to reconfigure pipe/endpoint 3 with a larger size, the controller allocates a memory area after the pipe/endpoint 2 memory area and automatically slides up the pipe/endpoint 4 memory window. Pipe/endpoint 5 does not move and a memory conflict appears as the memory windows of pipes/endpoints 4 and 5 overlap. The data of these pipes/endpoints is potentially lost.
   **Note:** 1. The data of pipe or endpoint 0 cannot be lost (except if it is de-allocated) as the memory allocation and de-allocation may affect only higher pipes/endpoints.

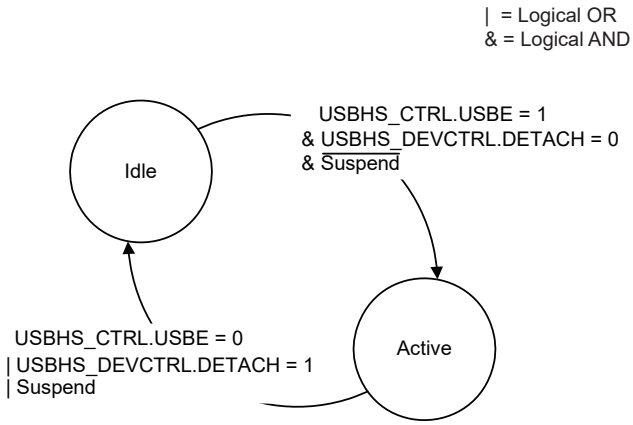
   **Note:** 2. Deactivating then reactivating the same pipe/endpoint with the same configuration only modifies temporarily the controller DPRAM pointer and size for this pipe/endpoint. Nothing changes in the DPRAM. Higher endpoints seem not to have been moved and their data is preserved as long as nothing has been written or received into them while changing the allocation state of the first pipe/endpoint.

   **Note:** 3. When the user writes a one to the USBHS_DEVEPTCFGx.ALLOC bit, the Configuration OK Status bit (USBHS_DEVEPTISRx.CFGOK) is set only if the configured size and number of banks are correct as compared to the endpoint maximum allowed values and to the maximum FIFO size (i.e., the DPRAM size). The USBHS_DEVEPTISRx.CFGOK value does not consider memory allocation conflicts.

#### 39.5.1.6 Pad Suspend

Figure 39-5 shows the pad behavior.

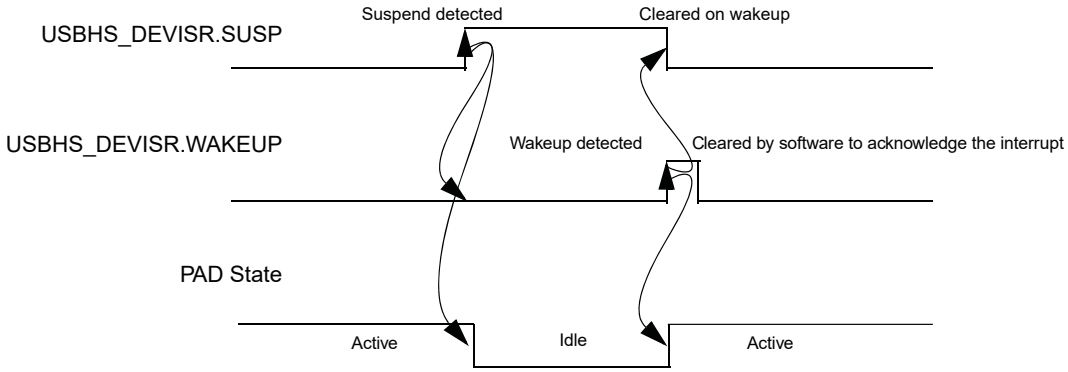
**Figure 39-5. Pad Behavior**




- In Idle state, the pad is put in Low-power mode, i.e., the differential receiver of the USB pad is off, and internal pull-downs with a strong value (15 K) are set in HSDP/D and HSDM/DM to avoid floating lines.
- In Active state, the pad is working.

Figure 39-6 illustrates the pad events leading to a PAD state change.

**Figure 39-6. Pad Events**




The USBHS_DEVISR.SUSP bit is set and the Wakeup Interrupt (USBHS_DEVISR.WAKEUP) bit is cleared when a USB “Suspend” state has been detected on the USB bus. This event automatically puts the USB pad in Idle state. The detection of a non-idle event sets USBHS_DEVISR.WAKEUP, clears USBHS_DEVISR.SUSP and wakes up the USB pad.

The pad goes to the Idle state if the USBHS is disabled or if the USBHS_DEVCTRL.DETACH bit = 1. It returns to the Active state when USBHS_CTRL.USBE = 1 and USBHS_DEVCTRL.DETACH = 0.

## 39.5.2 USB Device Operation

### 39.5.2.1 Introduction

In Device mode, the USBHS supports high-, full- and low-speed data transfers.

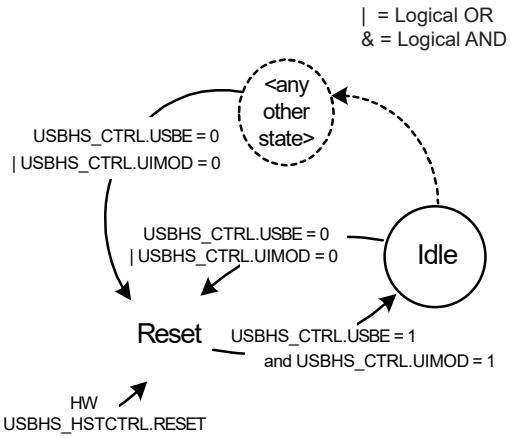
In addition to the default control endpoint, 10 endpoints are provided, which can be configured with an isochronous, bulk or interrupt type, as described in Table 39-1.

As the Device mode starts in Idle state, the pad consumption is reduced to the minimum.

### 39.5.2.2 Power-On and Reset

The following figure describes the USBHS Device mode main states.

**Figure 39-7. Device Mode Main States**




After a hardware reset, the USBHS Device mode is in Reset state. In this state:

- the USBHS clock is stopped to minimize power consumption (USBHS_CTRL.FRZCLK = 1),
- the internal registers of the Device mode are reset,
- the endpoint banks are de-allocated,
- neither D+ nor D- is pulled up (USBHS_DEVCTRL.DETACH = 1).

D+ or D- is pulled up according to the selected speed as soon as the USBHS_DEVCTRL.DETACH bit is written to zero. See "Device Mode" for further details.

When the USBHS is enabled (USBHS_CTRL.USBE = 1) in Device mode (USBHS_CTRL.UIMOD = 1), its Device mode state enters Idle state with minimal power consumption. This does not require the USB clock to be activated.

The USBHS Device mode can be disabled and reset at any time by disabling the USBHS (by writing a zero to USBHS_CTRL.USBE) or when the Host mode is enabled (USBHS_CTRL.UIMOD = 0).

### 39.5.2.3 USB Reset

The USB bus reset is managed by hardware. It is initiated by a connected host.

When a USB reset is detected on the USB line, the following operations are performed by the controller:

- All endpoints are disabled, except the default control endpoint.
- The default control endpoint is reset (see 39.5.2.4 Endpoint Reset for more details).
- The data toggle sequence of the default control endpoint is cleared.
- At the end of the reset process, the End of Reset (USBHS_DEVISR.EORST) bit is set.
- During a reset, the USBHS automatically switches to High-speed mode if the host is High-speed-capable (the reset is called High-speed reset). The user should observe the USBHS_SR.SPEED field to know the speed running at the end of the reset (USBHS_DEVISR.EORST = 1).

### 39.5.2.4 Endpoint Reset

An endpoint can be reset at any time by writing a one to the Endpoint x Reset bit USBHS_DEVEPT.EPRSTx. This is recommended before using an endpoint upon hardware reset or when a USB bus reset has been received. This resets:

- the internal state machine of the endpoint,
- the receive and transmit bank FIFO counters,
- all registers of this endpoint (USBHS_DEVEPTCFGx, USBHS_DEVEPTISRx, the Endpoint x Control (USBHS_DEVEPTIMRx) register), except its configuration (USBHS_DEVEPTCFGx.ALLOC, USBHS_DEVEPTCFGx.EPBK, USBHS_DEVEPTCFGx.EPSIZE, USBHS_DEVEPTCFGx.EPDIR, USBHS_DEVEPTCFGx.EPTYPE) and the Data Toggle Sequence (USBHS_DEVEPTISRx.DTSEQ) field. Note: The interrupt sources located in USBHS_DEVEPTISRx are not cleared when a USB bus reset has been received.

The endpoint configuration remains active and the endpoint is still enabled.

The endpoint reset may be associated with a clear of the data toggle sequence as an answer to the CLEAR_FEATURE USB request. This can be achieved by writing a one to the Reset Data Toggle Set bit (RSTDTS) in the Device Endpoint x Control Set register (this sets the Reset Data Toggle bit USBHS_DEVEPTIMRx.RSTDT).

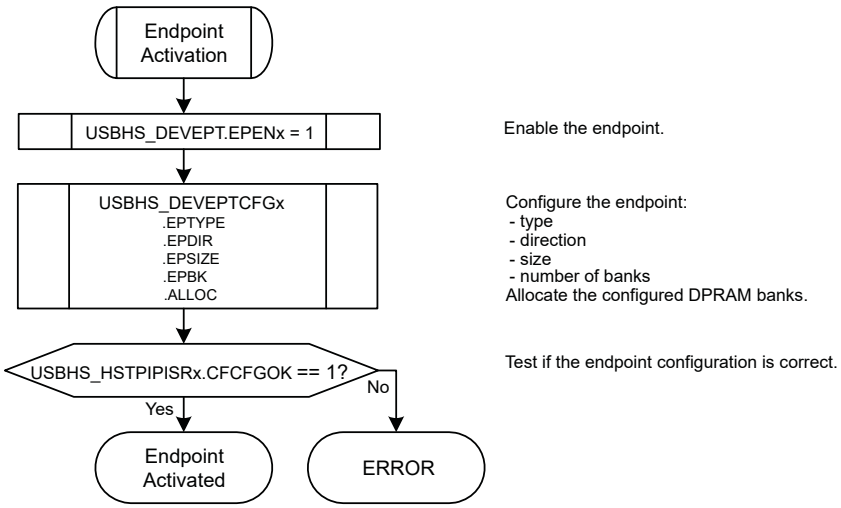
In the end, the user has to write a zero to the USBHS_DEVEPT.EPRSTx bit to complete the reset operation and to start using the FIFO.

#### 39.5.2.5 Endpoint Activation

The endpoint is maintained inactive and reset (see "Endpoint Reset" for more information) as long as it is disabled (USBHS_DEVEPT.EPENx = 0). USBHS_DEVEPTISRx.DTSEQ is also reset.

The algorithm represented in the following figure must be followed to activate an endpoint.

**Figure 39-8. Endpoint Activation Algorithm**




As long as the endpoint is not correctly configured (USBHS_HSTPIPISRx.CFGOK = 0), the controller does not acknowledge the packets sent by the host to this endpoint.

The USBHS_HSTPIPISRx.CFGOK bit is set provided that the configured size and number of banks are correct as compared to the endpoint maximal allowed values (see the Description of USB Pipes/Endpoints table) and to the maximal FIFO size (i.e., the DPRAM size).

See "DPRAM Management" for additional information.

#### 39.5.2.6 Address Setup

The USB device address is set up according to the USB protocol.

- After all kinds of resets, the USB device address is 0.
- The host starts a SETUP transaction with a SET_ADDRESS (addr) request.
- The user writes this address to the USB Address (USBHS_DEVCTRL.UADD) field, and writes a zero to the Address Enable (USBHS_DEVCTRL.ADDEN) bit, so the actual address is still 0.
- The user sends a zero-length IN packet from the control endpoint.
- The user enables the recorded USB device address by writing a one to USBHS_DEVCTRL.ADDEN.

Once the USB device address is configured, the controller filters the packets to accept only those targeting the address stored in USBHS_DEVCTRL.UADD.

USBHS_DEVCTRL.UADD and USBHS_DEVCTRL.ADDEN must not be written all at once.

USBHS_DEVCTRL.UADD and USBHS_DEVCTRL.ADDEN are cleared:

- on a hardware reset,
- when the USBHS is disabled (USBHS_CTRL.USBE = 0),
- when a USB reset is detected.

When USBHS_DEVCTRL.UADD or USBHS_DEVCTRL.ADDEN is cleared, the default device address 0 is used.

#### 39.5.2.7 Suspend and Wakeup

When an idle USB bus state has been detected for 3 ms, the controller sets the Suspend (USBHS_DEVISR.SUSP) interrupt bit. The user may then write a one to the USBHS_CTRL.FRZCLK bit to reduce power consumption.

To recover from the Suspend mode, the user should wait for the Wakeup (USBHS_DEVISR.WAKEUP) interrupt bit, which is set when a non-idle event is detected, then write a zero to USBHS_CTRL.FRZCLK.

As the USBHS_DEVISR.WAKEUP interrupt bit is set when a non-idle event is detected, it can occur whether the controller is in the Suspend mode or not. The USBHS_DEVISR.SUSP and USBHS_DEVISR.WAKEUP interrupts are thus independent, except that one bit is cleared when the other is set.

#### 39.5.2.8 Detach

The reset value of the USBHS_DEVCTRL.DETACH bit is one.

It is possible to initiate a device re-enumeration by simply writing a one, and then a zero, to USBHS_DEVCTRL.DETACH.

USBHS_DEVCTRL.DETACH acts on the pull-up connections of the D+ and D- pads. See “Device Mode” for further details.

#### 39.5.2.9 Remote Wakeup

The Remote Wakeup request (also known as Upstream Resume) is the only one the device may send without a host invitation, assuming a host command allowing the device to send such a request was previously issued. The sequence is the following:

1. The USBHS must have detected a “Suspend” state on the bus, i.e., the Remote Wakeup request can only be sent after a USBHS_DEVISR.SUSP interrupt has been set.
2. The user writes a one to the Remote Wakeup (USBHS_DEVCTRL.RMWKUP) bit to send an upstream resume to the host for a remote wakeup. This will automatically be done by the controller after 5 ms of inactivity on the USB bus.
3. When the controller sends the upstream resume, the Upstream Resume (USBHS_DEVISR.UPRSM) interrupt is set and USBHS_DEVISR.SUSP is cleared.
4. USBHS_DEVCTRL.RMWKUP is cleared at the end of the upstream resume.
5. When the controller detects a valid “End of Resume” signal from the host, the End of Resume (USBHS_DEVISR.EORSM) interrupt is set.

#### 39.5.2.10 STALL Request

For each endpoint, the STALL management is performed using:

- the STALL Request (USBHS_DEVEPTIMRx.STALLRQ) bit to initiate a STALL request,
- the STALLed Interrupt (USBHS_DEVEPTISRx.STALLEDI) bit, which is set when a STALL handshake has been sent.

To answer the next request with a STALL handshake, USBHS_DEVEPTIMRx.STALLRQ has to be set by writing a one to the STALL Request Set (USBHS_DEVEPTIERx.STALLRQS) bit. All following requests are discarded (USBHS_DEVEPTISRx.RXOUTI, etc. is not be set) and handshaked with a STALL until the USBHS_DEVEPTIMRx.STALLRQ bit is cleared, which is done when a new SETUP packet is received (for control endpoints) or when the STALL Request Clear (USBHS_DEVEPTIMRx.STALLRQC) bit is written to one.

Each time a STALL handshake is sent, the USBHS_DEVEPTISRx.STALLEDI bit is set by the USBHS and the PEP_x interrupt is set.

Special Considerations for Control Endpoints

If a SETUP packet is received into a control endpoint for which a STALL is requested, the Received SETUP Interrupt (USBHS_DEVEPTISRx.RXSTPI) bit is set and USBHS_DEVEPTIMRx.STALLRQ and USBHS_DEVEPTISRx.STALLEDI are cleared. The SETUP has to be ACKed.

This simplifies the enumeration process management. If a command is not supported or contains an error, the user requests a STALL and can return to the main task, waiting for the next SETUP request.

STALL Handshake and Retry Mechanism

The retry mechanism has priority over the STALL handshake. A STALL handshake is sent if the USBHS_DEVEPTIMRx.STALLRQ bit is set and if no retry is required.

© 2021 Microchip Technology Inc. and its subsidiaries

### 39.5.2.11 Management of Control Endpoints

Overview

A SETUP request is always ACKed. When a new SETUP packet is received, the USBHS_DEVEPTISRx.RXSTPI is set; the Received OUT Data Interrupt (USBHS_DEVEPTISRx.RXOUTI) bit is not.

The FIFO Control (USBHS_DEVEPTIMRx.FIFOCON) bit and the Read/Write Allowed (USBHS_DEVEPTISRx.RWALL) bit are irrelevant for control endpoints. The user never uses them on these endpoints. When read, their values are always zero.

Control endpoints are managed using:

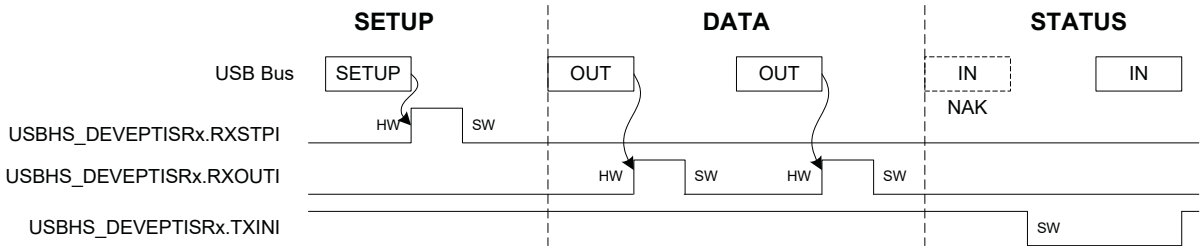
- the USBHS_DEVEPTISRx.RXSTPI bit, which is set when a new SETUP packet is received and which is cleared by firmware to acknowledge the packet and to free the bank;
- the USBHS_DEVEPTISRx.RXOUTI bit, which is set when a new OUT packet is received and which is cleared by firmware to acknowledge the packet and to free the bank;
- the Transmitted IN Data Interrupt (USBHS_DEVEPTISRx.TXINI) bit, which is set when the current bank is ready to accept a new IN packet and which is cleared by firmware to send the packet.

Control Write

Figure 39-9 shows a control write transaction. During the status stage, the controller does not necessarily send a NAK on the first IN token:

- if the user knows the exact number of descriptor bytes that must be read, it can then anticipate the status stage and send a zero-length packet after the next IN token, or
- it can read the bytes and wait for the NAKed IN Interrupt (USBHS_DEVEPTISRx.NAKINI), which acknowledges that all the bytes have been sent by the host and that the transaction is now in the status stage.

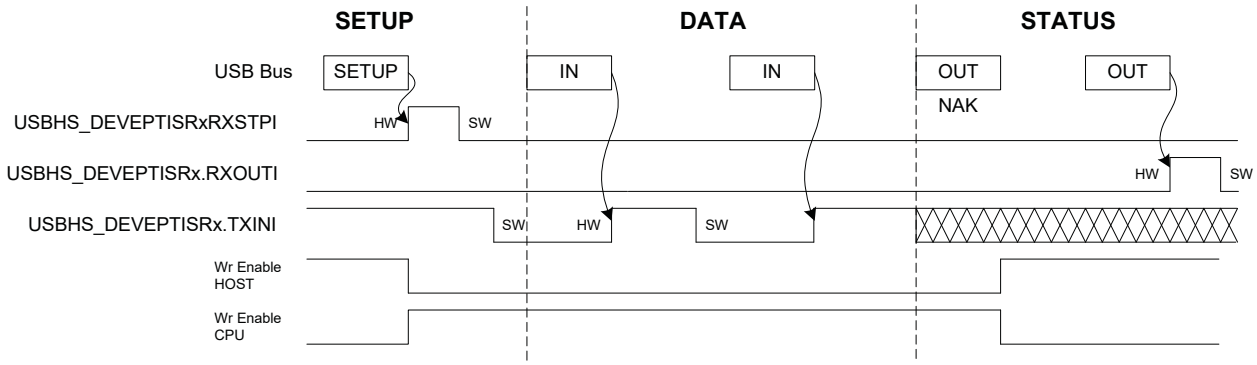
**Figure 39-9. Control Write**




Control Read

Figure 39-10 shows a control read transaction. The USBHS has to manage the simultaneous write requests from the CPU and the USB host.

**Figure 39-10. Control Read**




A NAK handshake is always generated on the first status stage command.

When the controller detects the status stage, all data written by the CPU is lost and clearing USBHS_DEVEPTISRx.TXINI has no effect.

The user checks if the transmission or the reception is complete.

The OUT retry is always ACKed. This reception sets USBHS_DEVEPTISRx.RXOUTI and USBHS_DEVEPTISRx.TXINI. Handle this with the following software algorithm:

set TXINI

wait for RXOUTI OR TXINI

if RXOUTI, then clear bit and return

if TXINI, then continue

Once the OUT status stage has been received, the USBHS waits for a SETUP request. The SETUP request has priority over any other request and has to be ACKed. This means that any other bit should be cleared and the FIFO reset when a SETUP is received.

The user has to consider that the byte counter is reset when a zero-length OUT packet is received.

#### 39.5.2.12 Management of IN Endpoints

Overview

IN packets are sent by the USB device controller upon IN requests from the host. All data which acknowledges or not the bank can be written when it is full.

The endpoint must be configured first.

The USBHS_DEVEPTISRx.TXINI bit is set at the same time as USBHS_DEVEPTIMRx.FIFOCON when the current bank is free. This triggers a PEP_x interrupt if the Transmitted IN Data Interrupt Enable (USBHS_DEVEPTIMRx.TXINE) bit is one.

USBHS_DEVEPTISRx.TXINI is cleared by software (by writing a one to the Transmitted IN Data Interrupt Clear bit (USBHS_DEVEPTIDRx.TXINIC) to acknowledge the interrupt, which has no effect on the endpoint FIFO.

The user then writes into the FIFO and writes a one to the FIFO Control Clear (USBHS_DEVEPTIDRx.FIFOCONC) bit to clear the USBHS_DEVEPTIMRx.FIFOCON bit. This allows the USBHS to send the data. If the IN endpoint is composed of multiple banks, this also switches to the next bank. The USBHS_DEVEPTISRx.TXINI and USBHS_DEVEPTIMRx.FIFOCON bits are updated in accordance with the status of the next bank.

USBHS_DEVEPTISRx.TXINI is always cleared before clearing USBHS_DEVEPTIMRx.FIFOCON.

The USBHS_DEVEPTISRx.RWALL bit is set when the current bank is not full, i.e., when the software can write further data into the FIFO.

**Figure 39-11. Example of an IN Endpoint with one Data Bank**

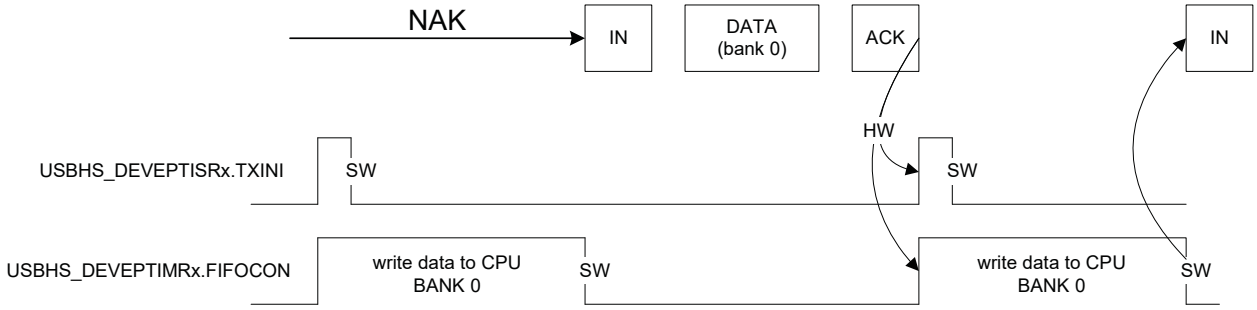
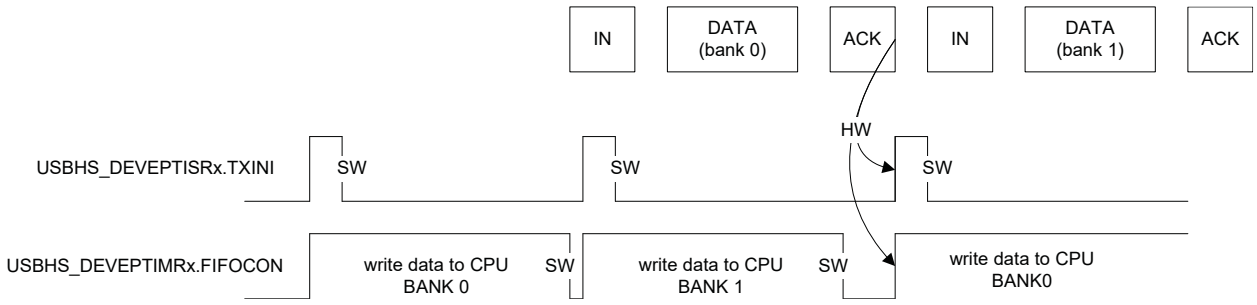



**Figure 39-12. Example of an IN Endpoint with two Data Banks**



Detailed Description

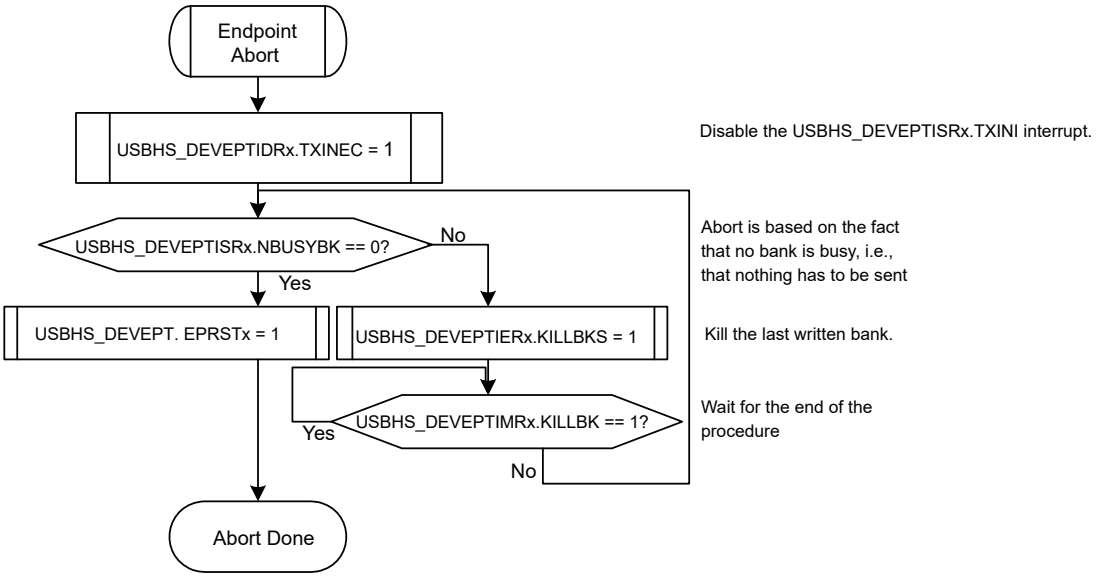
The data is written as follows:

- When the bank is empty, USBHS_DEVEPTISRx.TXINI and USBHS_DEVEPTIMRx.FIFOCON are set, which triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.TXINE = 1.
- The user acknowledges the interrupt by clearing USBHS_DEVEPTISRx.TXINI.
- The user writes the data into the current bank by using the USB Pipe/Endpoint nFIFO Data (USBFIFOnDATA) register, until all the data frame is written or the bank is full (in which case USBHS_DEVEPTISRx.RWALL is cleared and the Byte Count (USBHS_DEVEPTISRx.BYCT) field reaches the endpoint size).
- The user allows the controller to send the bank and switches to the next bank (if any) by clearing USBHS_DEVEPTIMRx.FIFOCON.

If the endpoint uses several banks, the current one can be written while the previous one is being read by the host. Then, when the user clears USBHS_DEVEPTIMRx.FIFOCON, the following bank may already be free and USBHS_DEVEPTISRx.TXINI is set immediately.

An “Abort” stage can be produced when a zero-length OUT packet is received during an IN stage of a control or isochronous IN transaction. The Kill IN Bank (USBHS_DEVEPTIMRx.KILLBK) bit is used to kill the last written bank. The best way to manage this abort is to apply the algorithm represented in the following figure.

**Figure 39-13. Abort Algorithm**




### 39.5.2.13 Management of OUT Endpoints

Overview

OUT packets are sent by the host. All data which acknowledges or not the bank can be read when it is empty.

The endpoint must be configured first.

The USBHS_DEVEPTISRx.RXOUTI bit is set at the same time as USBHS_DEVEPTIMRx.FIFOCON when the current bank is full. This triggers a PEP_x interrupt if the Received OUT Data Interrupt Enable (USBHS_DEVEPTIMRx.RXOUTE) bit is one.

USBHS_DEVEPTISRx.RXOUTI is cleared by software (by writing a one to the Received OUT Data Interrupt Clear (USBHS_DEVEPTICRx.RXOUTIC) bit to acknowledge the interrupt, which has no effect on the endpoint FIFO.

The user then reads from the FIFO and clears the USBHS_DEVEPTIMRx.FIFOCON bit to free the bank. If the OUT endpoint is composed of multiple banks, this also switches to the next bank. The USBHS_DEVEPTISRx.RXOUTI and USBHS_DEVEPTIMRx.FIFOCON bits are updated in accordance with the status of the next bank.

USBHS_DEVEPTISRx.RXOUTI is always cleared before clearing USBHS_DEVEPTIMRx.FIFOCON.

The USBHS_DEVEPTISRx.RWALL bit is set when the current bank is not empty, i.e., when the software can read further data from the FIFO.

**Figure 39-14. Example of an OUT Endpoint with one Data Bank**

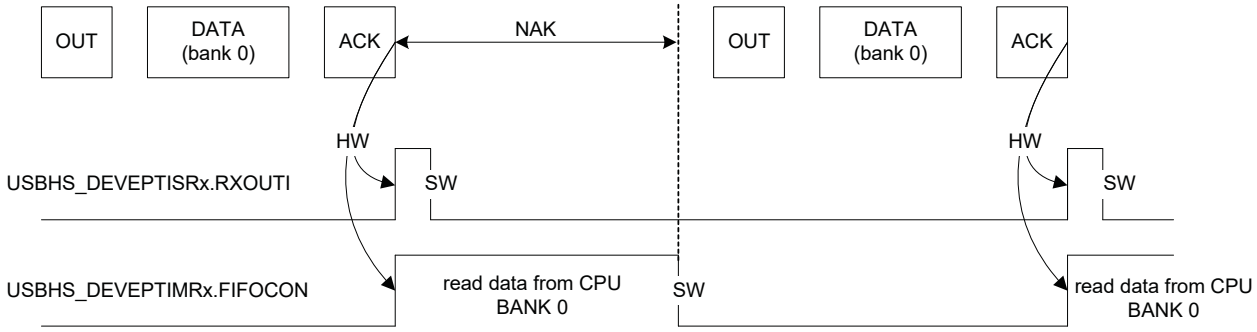
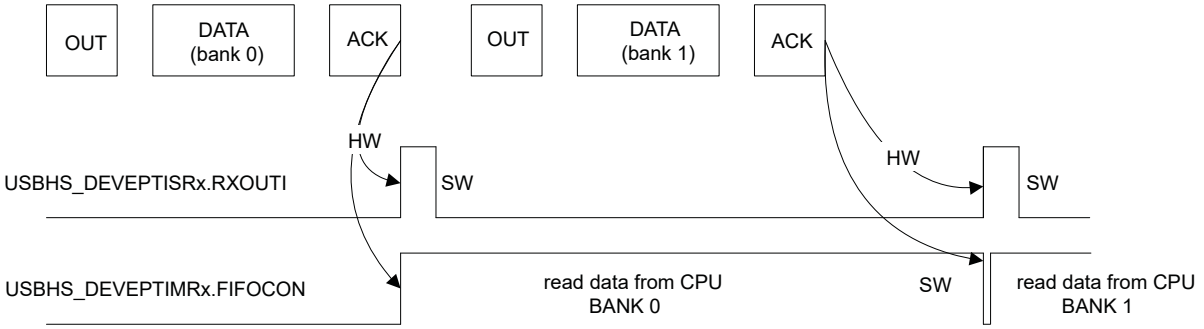



**Figure 39-15. Example of an OUT Endpoint with two Data Banks**




Detailed Description

The data is read as follows:

- When the bank is full, USBHS_DEVEPTISRx.RXOUTI and USBHS_DEVEPTIMRx.FIFOCON are set, which triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.RXOUTE = 1.
- The user acknowledges the interrupt by writing a one to USBHS_DEVEPTICRx.RXOUTIC in order to clear USBHS_DEVEPTISRx.RXOUTI.
- The user can read the byte count of the current bank from USBHS_DEVEPTISRx.BYCT to know how many bytes to read, rather than polling USBHS_DEVEPTISRx.RWALL.
- The user reads the data from the current bank by using the USBFIFOnDATA register, until all the expected data frame is read or the bank is empty (in which case USBHS_DEVEPTISRx.RWALL is cleared and USBHS_DEVEPTISRx.BYCT reaches zero).
- The user frees the bank and switches to the next bank (if any) by clearing USBHS_DEVEPTIMRx.FIFOCON.

If the endpoint uses several banks, the current one can be read while the following one is being written by the host. Then, when the user clears USBHS_DEVEPTIMRx.FIFOCON, the following bank can already be read and USBHS_DEVEPTISRx.RXOUTI is set immediately.

In High-speed mode, the PING and NYET protocols are handled by the USBHS.

- For a single bank, a NYET handshake is always sent to the host (on Bulk-out transaction) to indicate that the current packet is acknowledged but there is no room for the next one.
- For a double bank, the USBHS responds to the OUT/DATA transaction with an ACK handshake when the endpoint accepted the data successfully and has room for another data payload (the second bank is free).

#### 39.5.2.14 Underflow

This error only exists for isochronous IN/OUT endpoints. It sets the Underflow Interrupt (USBHS_DEVEPTISRx.UNDERFI) bit, which triggers a PEP_x interrupt if the Underflow Interrupt Enable (USBHS_DEVEPTIMRx.UNDERFE) bit is one.

- An underflow can occur during the IN stage if the host attempts to read from an empty bank. A zero-length packet is then automatically sent by the USBHS.

- An underflow cannot occur during the OUT stage on a CPU action, since the user may only read if the bank is not empty (USBHS_DEVEPTISRx.RXOUTI = 1 or USBHS_DEVEPTISRx.RWALL = 1).
- An underflow can also occur during the OUT stage if the host sends a packet while the bank is already full. Typically, the CPU is not fast enough. The packet is lost.
- An underflow cannot occur during the IN stage on a CPU action, since the user may only write if the bank is not full (USBHS_DEVEPTISRx.TXINI = 1or USBHS_DEVEPTISRx.RWALL = 1).

#### 39.5.2.15 Overflow

This error exists for all endpoint types. It sets the Overflow interrupt (USBHS_DEVEPTISRx.OVERFI) bit, which triggers a PEP_x interrupt if the Overflow Interrupt Enable (USBHS_DEVEPTIMRx.OVERFE) bit is one.

- An overflow can occur during the OUT stage if the host attempts to write into a bank which is too small for the packet. The packet is acknowledged and the USBHS_DEVEPTISRx.RXOUTI bit is set as if no overflow had occurred. The bank is filled with all the first bytes of the packet that fit in.
- An overflow cannot occur during the IN stage on a CPU action, since the user may only write if the bank is not full (USBHS_DEVEPTISRx.TXINI = 1 or USBHS_DEVEPTISRx.RWALL = 1).

#### 39.5.2.16 HB IsoIn Error

This error only exists for high-bandwidth isochronous IN endpoints.

At the end of the microframe, if at least one packet has been sent to the host and fewer banks than expected have been validated (by clearing the USBHS_DEVEPTIMRx.USBHS_DEVEPTIMRx.FIFOCON) for this microframe, it sets the USBHS_DEVEPTISRx.HBISOINERRORI bit, which triggers a PEP_x interrupt if the High Bandwidth Isochronous IN Error Interrupt Enable (HBISOINERRORE) bit is one.

For example, if the Number of Transactions per MicroFrame for Isochronous Endpoint (NBTRANS) field in USBHS_DEVEPTCFGx is three (three transactions per microframe), only two banks are filled by the CPU (three expected) for the current microframe. Then, the HBISOINERRI interrupt is generated at the end of the microframe. Note that an UNDERFI interrupt is also generated (with an automatic zero-length-packet), except in the case of a missing IN token.

#### 39.5.2.17 HB IsoFlush

This error only exists for high-bandwidth isochronous IN endpoints.

At the end of the microframe, if at least one packet has been sent to the host and there is a missing IN token during this microframe, the bank(s) destined to this microframe is/are flushed out to ensure a good data synchronization between the host and the device.

For example, if NBTRANS is three (three transactions per microframe) and if only the first IN token (among three) is well received by the USBHS, the last two banks are discarded.

#### 39.5.2.18 CRC Error

This error only exists for isochronous OUT endpoints. It sets the CRC Error Interrupt (USBHS_DEVEPTISRx.CRCERRI) bit, which triggers a PEP_x interrupt if the CRC Error Interrupt Enable (USBHS_DEVEPTIMRx.CRCERRE) bit is one.

A CRC error can occur during the OUT stage if the USBHS detects a corrupted received packet. The OUT packet is stored in the bank as if no CRC error had occurred (USBHS_DEVEPTISRx.RXOUTI is set).

#### 39.5.2.19 Interrupts

See the structure of the USB device interrupt system in Figure 39-3.

There are two kinds of device interrupts: processing, i.e., their generation is part of the normal processing, and exception, i.e., errors (not related to CPU exceptions).

Global Interrupts

The processing device global interrupts are:

- Suspend (USBHS_DEVISR.SUSP)
- Start of Frame (USBHS_DEVISR.SOF) interrupt with no frame number CRC error - the Frame Number CRC Error (USBHS_DEVFNUM.FNCERR) bit is zero.
- Micro Start of Frame (USBHS_DEVISR.MSOF) with no CRC error

- End of Reset (USBHS_DEVISR.EORST)
- Wakeup (USBHS_DEVISR.WAKEUP)
- End of Resume (USBHS_DEVISR.EORSM)
- Upstream Resume (USBHS_DEVISR.UPRSM)
- Endpoint x (USBHS_DEVISR.PEP_x)
- DMA Channel x (USBHS_DEVISR.DMA_x)

The exception device global interrupts are:

- Start of Frame (USBHS_DEVISR.SOF) with a frame number CRC error (USBHS_DEVFNUM.FNCERR = 1)
- Micro Start of Frame (USBHS_DEVFNUM.FNCERR.MSOF) with a CRC error
  Endpoint Interrupts

The processing device endpoint interrupts are:

- Transmitted IN Data (USBHS_DEVEPTISRx.TXINI)
- Received OUT Data (USBHS_DEVEPTISRx.RXOUTI)
- Received SETUP (USBHS_DEVEPTISRx.RXSTPI)
- Short Packet (USBHS_DEVEPTISRx.SHORTPACKET)
- Number of Busy Banks (USBHS_DEVEPTISRx.NBUSYBK)
- Received OUT Isochronous Multiple Data (DTSEQ = MDATA & USBHS_DEVEPTISRx.RXOUTI)
- Received OUT Isochronous DataX (DTSEQ = DATAX & USBHS_DEVEPTISRx.RXOUTI)

The exception device endpoint interrupts are:

- Underflow (USBHS_DEVEPTISRx.UNDERFI)
- NAKed OUT (USBHS_DEVEPTISRx.NAKOUTI)
- High-Bandwidth Isochronous IN Error (USBHS_DEVEPTISRx.HBISOINERRI)
- NAKed IN (USBHS_DEVEPTISRx.NAKINI)
- High-Bandwidth Isochronous IN Flush error (USBHS_DEVEPTISRx.HBISOFLUSHI)
- Overflow (USBHS_DEVEPTISRx.OVERFI)
- STALLed (USBHS_DEVEPTISRx.STALLEDI)
- CRC Error (USBHS_DEVEPTISRx.CRCERRI)
- Transaction Error (USBHS_DEVEPTISRx.ERRORTRANS)
  DMA Interrupts

The processing device DMA interrupts are:

- End of USB Transfer Status (USBHS_DEVDMASTATUSx.END_TR_ST)
- End of Channel Buffer Status (USBHS_DEVDMASTATUSx.END_BF_ST)
- Descriptor Loaded Status (USBHS_DEVDMASTATUSx.DESC_LDST)

There is no exception device DMA interrupt.

#### 39.5.2.20 Test Modes

When written to one, the USBHS_DEVCTRL.TSTPCKT bit switches the USB device controller to a “Test-packet” mode:

The transceiver repeatedly transmits the packet stored in the current bank. USBHS_DEVCTRL.TSTPCKT must be written to zero to exit the Test-packet mode. The endpoint is reset by software after a Test-packet mode.

This enables the testing of rise and falling times, eye patterns, jitter, and any other dynamic waveform specifications.

The flow control used to send the packets is as follows:

- USBHS_DEVCTRL.TSTPCKT = 1;
- Store data in an endpoint bank
- Write a zero to the USBHS_DEVEPTIDRx.FIFOCON bit

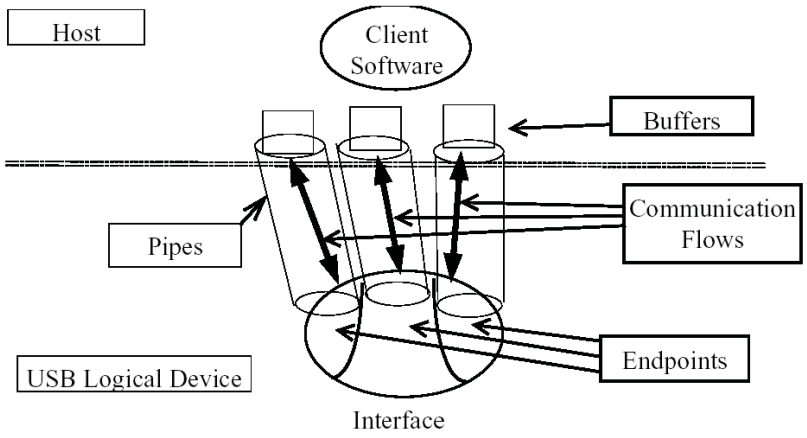
To stop the Test-packet mode, write a zero to the USBHS_DEVCTRL.TSTPCKT bit.

### 39.5.3 USB Host Operation

#### 39.5.3.1 Description of Pipes

For the USBHS in Host mode, the term “pipe” is used instead of “endpoint” (used in Device mode). A host pipe corresponds to a device endpoint, as described in Figure 39-16 (from the USB Specification).

**Figure 39-16. USB Communication Flow**

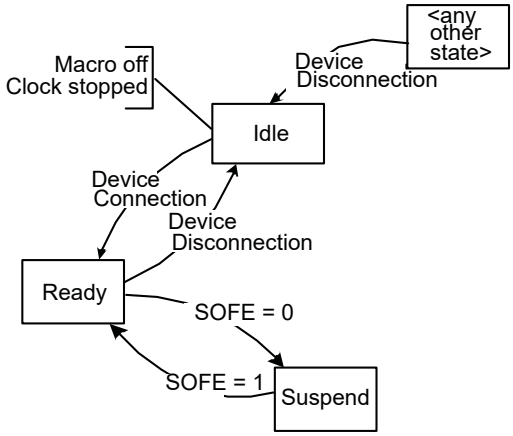



In Host mode, the USBHS associates a pipe to a device endpoint, considering the device configuration descriptors.

#### 39.5.3.2 Power-On and Reset

The following figure describes the USBHS Host mode main states.

**Figure 39-17. Host Mode Main States**




After a hardware reset, the USBHS Host mode is in the Reset state.

When the USBHS is enabled (USBHS_CTRL.USBE = 1) in Host mode (USBHS_CTRL.UIMOD = 0), it goes to the Idle state. In this state, the controller waits for a device connection with a minimal power consumption. The USB pad should be in the Idle state. Once a device is connected, the USBHS enters the Ready state, which does not require the USB clock to be activated.

The controller enters the Suspend state when the USB bus is in a “Suspend” state, i.e., when the Host mode does not generate the “Start of Frame (SOF)”. In this state, the USB consumption is minimal. The Host mode exits the Suspend state when starting to generate the SOF over the USB line.

#### 39.5.3.3 Device Detection

A device is detected by the USBHS Host mode when D+ or D- is no longer tied low, i.e., when the device D+ or D- pull-up resistor is connected. The bit USBHS_SFR.VBUSRQS must be set to ‘1’ to enable this detection.

**Note:** The VBUS supply is not managed by the USBHS interface. It must be generated on-board.

The device disconnection is detected by the host controller when both D+ and D- are pulled down.

#### 39.5.3.4 USB Reset

The USBHS sends a USB bus reset when the user writes a one to the Send USB Reset bit in the Host General Control register (USBHS_HSTCTRL.RESET). The USB Reset Sent Interrupt bit in the Host Global Interrupt Status register (USBHS_HSTISR.RSTI) is set when the USB reset has been sent. In this case, all pipes are disabled and de-allocated.

If the bus was previously in a “Suspend” state (the Start of Frame Generation Enable (USBHS_HSTCTRL.SOFE) bit is zero), the USBHS automatically switches to the “Resume” state, the Host Wakeup Interrupt (USBHS_HSTISR.HWUPI) bit is set and the USBHS_HSTCTRL.SOFE bit is set in order to generate SOFs or micro SOFs immediately after the USB reset.

At the end of the reset, the user should check the USBHS_SR.SPEED field to know the speed running according to the peripheral capability (LS.FS/HS).

#### 39.5.3.5 Pipe Reset

A pipe can be reset at any time by writing a one to the Pipe x Reset (USBHS_HSTPIP.PRSTx) bit. This is recommended before using a pipe upon hardware reset or when a USB bus reset has been sent. This resets:

- the internal state machine of the pipe,
- the receive and transmit bank FIFO counters,
- all the registers of the pipe (USBHS_HSTPIPCFGx, USBHS_HSTPIPISRx, USBHS_HSTPIPIMRx), except its configuration (USBHS_HSTPIPCFGx.ALLOC, USBHS_HSTPIPCFGx.PBK, USBHS_HSTPIPCFGx.PSIZE, USBHS_HSTPIPCFGx.PTOKEN, USBHS_HSTPIPCFGx.PTYPE, USBHS_HSTPIPCFGx.PEPNUM, USBHS_HSTPIPCFGx.INTFRQ) and its Data Toggle Sequence field (USBHS_HSTPIPISRx.DTSEQ).

The pipe configuration remains active and the pipe is still enabled.

The pipe reset may be associated with a clear of the data toggle sequence. This can be achieved by setting the Reset Data Toggle bit in the Pipe x Control register (USBHS_HSTPIPIMRx.RSTDT) (by writing a one to the Reset Data Toggle Set bit in the Pipe x Control Set register (USBHS_HSTPIPIERx.RSTDTS)).

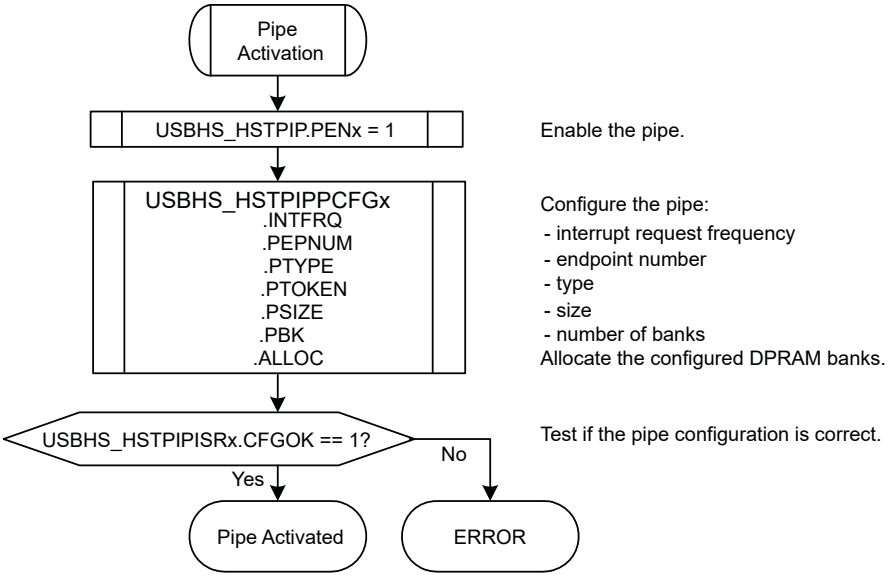
In the end, the user has to write a zero to the USBHS_HSTPIP.PRSTx bit to complete the reset operation and to start using the FIFO.

#### 39.5.3.6 Pipe Activation

The pipe is maintained inactive and reset (see "Pipe Reset" for more details) as long as it is disabled (USBHS_HSTPIP.PENx = 0). The Data Toggle Sequence field (USBHS_HSTPIPISRx.DTSEQ) is also reset.

The algorithm represented in the following figure must be followed to activate a pipe.

**Figure 39-18. Pipe Activation Algorithm**




As long as the pipe is not correctly configured (USBHS_HSTPIPISRx.CFGOK = 0), the controller cannot send packets to the device through this pipe.

The USBHS_HSTPIPISRx.CFGOK bit is only set if the configured size and number of banks are correct as compared to their maximal allowed values for the pipe (see the Description of USB Pipes/Endpoints table) and to the maximal FIFO size (i.e., the DPRAM size).

See "DPRAM Management" for additional information.

Once the pipe is correctly configured (USBHS_HSTPIPISRx.CFGOK = 1), only the USBHS_HSTPIPCFGx.PTOKEN and USBHS_HSTPIPCFGx.INTFRQ fields can be written by software. USBHS_HSTPIPCFGx.INTFRQ is meaningless for non-interrupt pipes.

When starting an enumeration, the user gets the device descriptor by sending a GET_DESCRIPTOR USB request. This descriptor contains the maximal packet size of the device default control endpoint (bMaxPacketSize0) and the user reconfigures the size of the default control pipe with this size parameter.

#### 39.5.3.7 Address Setup

Once the device has answered the first host requests with the default device address 0, the host assigns a new address to the device. The host controller has to send a USB reset to the device and to send a SET_ADDRESS (addr) SETUP request with the new address to be used by the device. Once this SETUP transaction is over, the user writes the new address into the USB Host Address for Pipe x field in the USB Host Device Address register (HSTADDR.HSTADDRPx). All the following requests on all pipes are then performed using this new address.

When the host controller sends a USB reset, the HSTADDRPx field is reset by hardware and the following host requests are performed using the default device address 0.

#### 39.5.3.8 Remote Wakeup

The controller Host mode enters the Suspend state when the USBHS_HSTCTRL.SOFE bit is written to zero. No more “Start of Frame” is sent on the USB bus and the USB device enters the Suspend state 3 ms later.

The device awakes the host by sending an Upstream Resume (Remote Wakeup feature). When the host controller detects a non-idle state on the USB bus, it sets the Host Wakeup interrupt (USBHS_HSTISR.HWUPI) bit. If the non-idle bus state corresponds to an Upstream Resume (K state), the Upstream Resume Received Interrupt (USBHS_HSTISR.RXRSMI) bit is set. The user has to generate a Downstream Resume within 1 ms and for at least 20 ms by writing a one to the Send USB Resume (USBHS_HSTCTRL.RESUME) bit. It is mandatory to write a one to USBHS_HSTCTRL.SOFE before writing a one to USBHS_HSTCTRL.RESUME to enter the Ready state, otherwise USBHS_HSTCTRL.RESUME has no effect.

#### 39.5.3.9 Management of Control Pipes

A control transaction is composed of three stages:

- SETUP
- Data (IN or OUT)
- Status (OUT or IN)

The user has to change the pipe token according to each stage.

For the control pipe only, each token is assigned a specific initial data toggle sequence:

- SETUP: Data0
- IN: Data1
- OUT: Data1

#### 39.5.3.10 Management of IN Pipes

IN packets are sent by the USB device controller upon IN requests from the host. All data which acknowledges or not the bank can be read when it is empty.

The pipe must be configured first.

When the host requires data from the device, the user has to first select the IN Request mode with the IN Request Mode bit in the Pipe x IN Request register (USBHS_HSTPIPINRQx.INMODE):

- When USBHS_HSTPIPINRQx.INMODE = 0, the USBHS performs (INRQ + 1) IN requests before freezing the pipe.
- When USBHS_HSTPIPINRQx.INMODE = 1, the USBHS performs IN requests endlessly when the pipe is not frozen by the user.

© 2021 Microchip Technology Inc. and its subsidiaries

The generation of IN requests starts when the pipe is unfrozen (the Pipe Freeze (USBHS_HSTPIPIMRx.PFREEZE) field in USBHS_HSTPIPIMRx is zero).

The Received IN Data Interrupt (USBHS_HSTPIPISRx.RXINI) bit is set at the same time as the FIFO Control (USBHS_HSTPIPIMRx.FIFOCON) bit when the current bank is full. This triggers a PEP_x interrupt if the Received IN Data Interrupt Enable (USBHS_HSTPIPIMRx.RXINE) bit is one.

USBHS_HSTPIPISRx.RXINI is cleared by software (by writing a one to the Received IN Data Interrupt Clear bit in the Host Pipe x Clear register (USBHS_HSTPIPIDRx.RXINIC)) to acknowledge the interrupt, which has no effect on the pipe FIFO.

The user then reads from the FIFO and clears the USBHS_HSTPIPIMRx.FIFOCON bit (by writing a one to the FIFO Control Clear (USBHS_HSTPIPIDRx.FIFOCONC) bit) to free the bank. If the IN pipe is composed of multiple banks, this also switches to the next bank. The USBHS_HSTPIPISRx.RXINI and USBHS_HSTPIPIMRx.FIFOCON bits are updated in accordance with the status of the next bank.

USBHS_HSTPIPISRx.RXINI is always cleared before clearing USBHS_HSTPIPIMRx.FIFOCON.

The Read/Write Allowed (USBHS_HSTPIPISRx.RWALL) bit is set when the current bank is not empty, i.e., when the software can read further data from the FIFO.

**Figure 39-19. Example of an IN Pipe with one Data Bank**

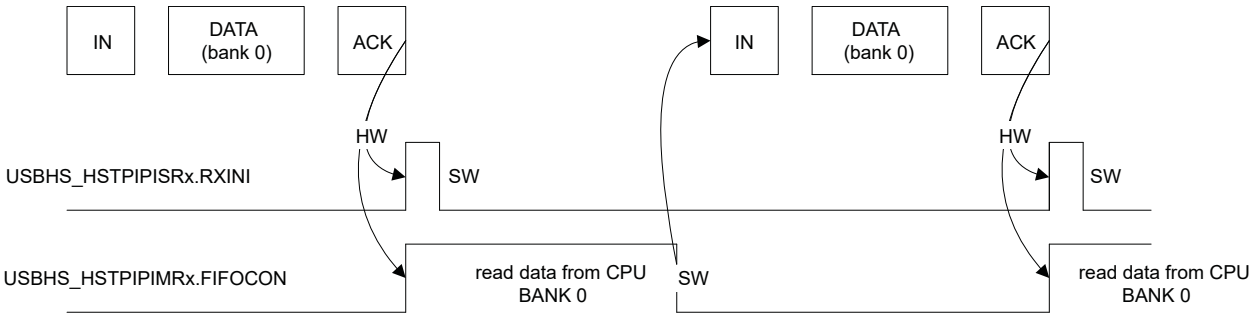
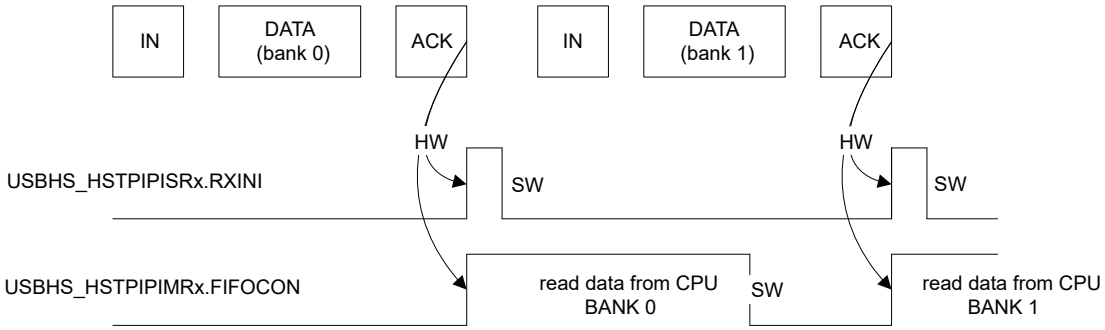



**Figure 39-20. Example of an IN Pipe with two Data Banks**




### 39.5.3.11 Management of OUT Pipes

OUT packets are sent by the host. All data which acknowledges or not the bank can be written when it is full.

The pipe must be configured and unfrozen first.

The Transmitted OUT Data Interrupt (USBHS_HSTPIPISRx.TXOUTI) bit is set at the same time as USBHS_HSTPIPIMRx.FIFOCON when the current bank is free. This triggers a PEP_x interrupt if the Transmitted OUT Data Interrupt Enable (USBHS_HSTPIPIMRx.TXOUTE) bit is one.

USBHS_HSTPIPISRx.TXOUTI is cleared by software (by writing a one to the Transmitted OUT Data Interrupt Clear (USBHS_HSTPIPIDRx.TXOUTIC) bit to acknowledge the interrupt, which has no effect on the pipe FIFO.

The user then writes into the FIFO and clears the USBHS_HSTPIPIDRx.FIFOCON bit to allow the USBHS to send the data. If the OUT pipe is composed of multiple banks, this also switches to the next bank. The USBHS_HSTPIPISRx.TXOUTI and USBHS_HSTPIPIMRx.FIFOCON bits are updated in accordance with the status of the next bank.

USBHS_HSTPIPISRx.TXOUTI is always cleared before clearing USBHS_HSTPIPIMRx.FIFOCON.

The USBHS_HSTPIPISRx.RWALL bit is set when the current bank is not full, i.e., when the software can write further data into the FIFO.

**Notes:**

1. If the user decides to switch to the Suspend state (by writing a zero to the USBHS_HSTCTRL.SOFE bit) while a bank is ready to be sent, the USBHS automatically exits this state and the bank is sent.
2. In High-speed operating mode, the host controller automatically manages the PING protocol to maximize the USB bandwidth. The user can tune the PING protocol by handling the Ping Enable (PINGEN) bit and the bInterval Parameter for the Bulk-Out/Ping Transaction (BINTERVAL) field in USBHS_HSTPIPCFGx. See the Host Pipe x Configuration Register for additional information.

**Figure 39-21. Example of an OUT Pipe with one Data Bank**

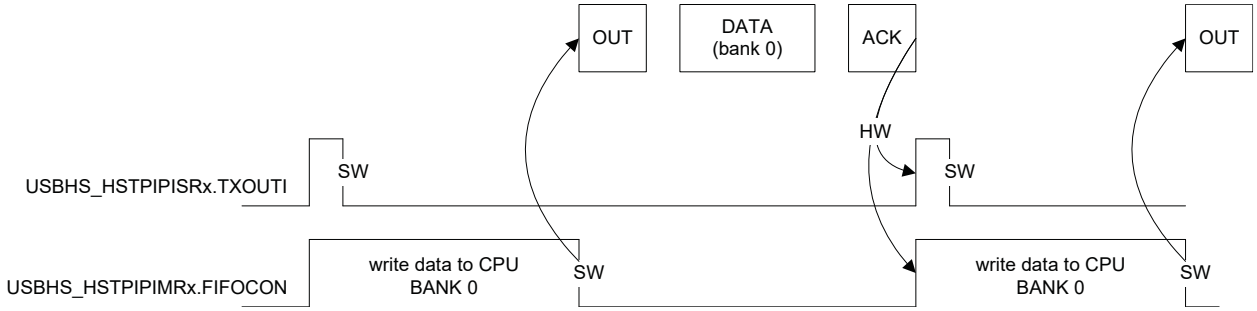



**Figure 39-22. Example of an OUT Pipe with two Data Banks and no Bank Switching Delay**

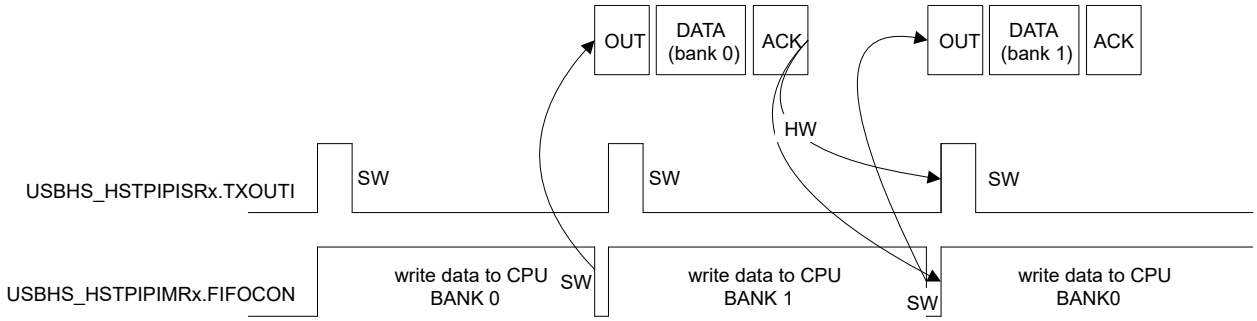
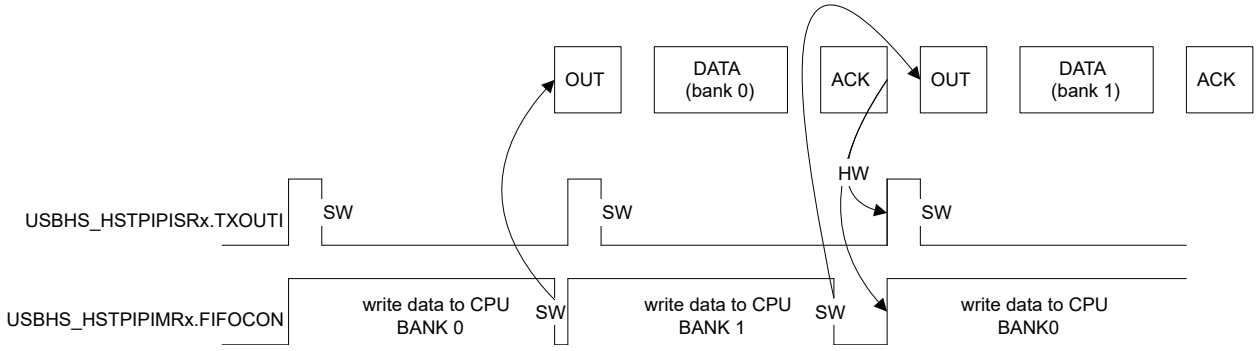



**Figure 39-23. Example of an OUT Pipe with two Data Banks and a Bank Switching Delay**




### 39.5.3.12 CRC Error

This error exists only for isochronous IN pipes. It sets the CRC Error Interrupt (USBHS_HSTPIPISRx.CRCERRI) bit, which triggers a PEP_x interrupt if then the CRC Error Interrupt Enable (USBHS_HSTPIPIMRx.CRCERRE) bit is one.

A CRC error can occur during IN stage if the USBHS detects a corrupted received packet. The IN packet is stored in the bank as if no CRC error had occurred (USBHS_HSTPIPISRx.RXINI is set).

### 39.5.3.13 Interrupts

See the structure of the USB host interrupt system on Figure 39-3.

There are two kinds of host interrupts: processing, i.e., their generation is part of the normal processing, and exception, i.e., errors (not related to CPU exceptions).

Global Interrupts

The processing host global interrupts are:

- Device Connection (USBHS_HSTISR.DCONNI)
- Device Disconnection (USBHS_HSTISR.DDISCI)
- USB Reset Sent (USBHS_HSTISR.RSTI)
- Downstream Resume Sent (USBHS_HSTISR.RSMEDI)
- Upstream Resume Received (USBHS_HSTISR.RXRSMI)
- Host Start of Frame (USBHS_HSTISR.HSOFI)
- Host Wakeup (USBHS_HSTISR.HWUPI)
- Pipe x (USBHS_HSTISR.PEP_x)
- DMA Channel x (USBHS_HSTISR.DMAxINT)

There is no exception host global interrupt.

Pipe Interrupts

The processing host pipe interrupts are:

- Received IN Data (USBHS_HSTPIPISRx.RXINI)
- Transmitted OUT Data (USBHS_HSTPIPISRx.TXOUTI)
- Transmitted SETUP (USBHS_HSTPIPISRx.TXSTPI)
- Short Packet (USBHS_HSTPIPISRx.SHORTPACKETI)
- Number of Busy Banks (USBHS_HSTPIPISRx.NBUSYBK)

The exception host pipe interrupts are:

- Underflow (USBHS_HSTPIPISRx.UNDERFI)
- Pipe Error (USBHS_HSTPIPISRx.PERRI)
- NAKed (USBHS_HSTPIPISRx.NAKEDI)
- Overflow (USBHS_HSTPIPISRx.OVERFI)
- Received STALLed (USBHS_HSTPIPISRx.RXSTALLDI)
- CRC Error (USBHS_HSTPIPISRx.CRCERRI)
  DMA Interrupts

The processing host DMA interrupts are:

- The End of USB Transfer Status (USBHS_HSTDMASTATUSx.END_TR_ST)
- The End of Channel Buffer Status (USBHS_HSTDMASTATUSx.END_BF_ST)
- The Descriptor Loaded Status (USBHS_HSTDMASTATUSx.DESC_LDST)

There is no exception host DMA interrupt.

### 39.5.4 USB DMA Operation

USB packets of any length may be transferred when required by the USBHS. These transfers always feature sequential addressing. Such characteristics mean that in case of high USBHS throughput, both AHB ports benefit from “incrementing burst of unspecified length” since the average access latency of AHB slaves can then be reduced.

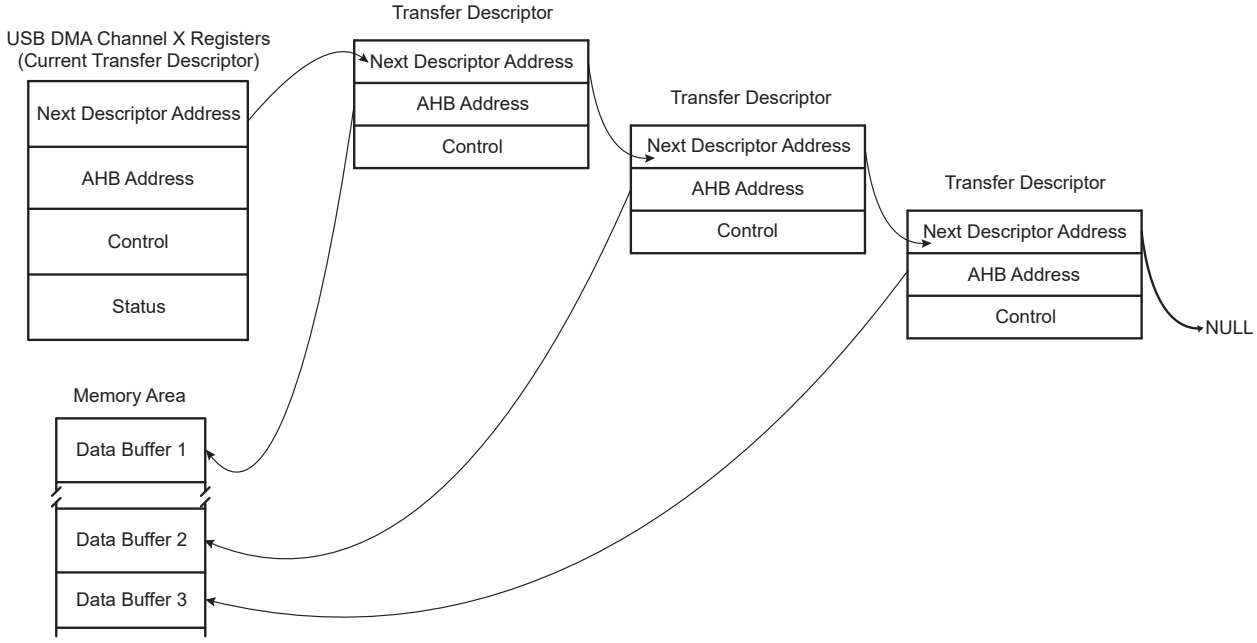
The DMA uses word “incrementing burst of unspecified length” of up to 256 beats for both data transfers and channel descriptor loading. A burst may last on the AHB busses for the duration of a whole USB packet transfer, unless otherwise broken by the AHB arbitration or the AHB 1-Kbyte boundary crossing.

Packet data AHB bursts may be locked on a DMA buffer basis for drastic overall AHB bus bandwidth performance boost with paged memories. This prevents large AHB bursts from being broken in case of conflict with other AHB bus masters, thus avoiding access latencies due to memory row changes. This means up to 128 words single cycle unbroken AHB bursts for bulk pipes/endpoints and 256 words single cycle unbroken bursts for isochronous pipes/endpoints. This maximal burst length is then controlled by the lowest programmed USB

Pipe/Endpoint Size (USBHS_HSTPIPCFGx.PSIZE / USBHS_DEVEPTCFGx.EPSIZE) and the Buffer Byte Length (USBHS_HSTDMACONTROLx.BUFF_LENGTH / USBHS_DEVDMACONTROLx.BUFF_LENGTH) fields.

The USBHS average throughput can reach nearly 480 Mbps. Its average access latency decreases as burst length increases due to the zero wait-state side effect of unchanged pipe/endpoint. Word access allows reducing the AHB bandwidth required for the USB by four, as compared to native byte access. If at least 0 wait-state word burst capability is also provided by the other DMA AHB bus slaves, each DMA AHB bus needs less than 60% bandwidth allocation for full USB bandwidth usage at 33 MHz, and less than 30% at 66 MHz.

**Figure 39-24. Example of a DMA Chained List**




### 39.5.5 USB DMA Channel Transfer Descriptor

The DMA channel transfer descriptor is loaded from the memory. The following structures apply:

Offset 0:

- The address must be aligned: 0xXXXX0
- Next Descriptor Address Register: USBHS_xxxDMANXTDSCx

Offset 4:

- The address must be aligned: 0xXXXX4
- DMA Channelx Address Register: USBHS_xxxDMAADDRESSx

Offset 8:

- The address must be aligned: 0xXXXX8
- DMA Channelx Control Register: USBHS_xxxDMACONTROLx

To use the DMA channel transfer descriptor, fill the structures with the correct values (as described in the following pages), then write directly in USBHS_xxxDMANXTDSCx the address of the descriptor to be used first.

Then write 1 in the USBHS_xxxDMACONTROLx.LDNXT_DSC bit (load next channel transfer descriptor). The descriptor is automatically loaded upon pipe x / endpoint x request for packet transfer.

## 39.6 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | USBHS_DEVCTRL | 7:0 | ADDEN | UADD[6:0] | | | | | | |
| | | 15:8 | TSTPCKT | TSTK | TSTJ | LS | SPDCONF[1:0] | | RMWKUP | DETACH |
| | | 23:16 | | | | | | | | OPMODE2 |
| | | 31:24 | | | | | | | | |
| 0x04 | USBHS_DEVISR | 7:0 | | UPRSM | EORSM | WAKEUP | EORST | SOF | MSOF | SUSP |
| | | 15:8 | PEP_3 | PEP_2 | PEP_1 | PEP_0 | | | | |
| | | 23:16 | | | PEP_9 | PEP_8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x08 | USBHS_DEVICR | 7:0 | | UPRSMC | EORSMC | WAKEUPC | EORSTC | SOFC | MSOFC | SUSPC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | USBHS_DEVIFR | 7:0 | | UPRSMS | EORSMS | WAKEUPS | EORSTS | SOFS | MSOFS | SUSPS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x10 | USBHS_DEVIMR | 7:0 | | UPRSME | EORSME | WAKEUPE | EORSTE | SOFE | MSOFE | SUSPE |
| | | 15:8 | PEP_3 | PEP_2 | PEP_1 | PEP_0 | | | | |
| | | 23:16 | | | PEP_9 | PEP_8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x14 | USBHS_DEVIDR | 7:0 | | UPRSMEC | EORSMEC | WAKEUPEC | EORSTEC | SOFEC | MSOFEC | SUSPEC |
| | | 15:8 | PEP_3 | PEP_2 | PEP_1 | PEP_0 | | | | |
| | | 23:16 | | | PEP_9 | PEP_8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x18 | USBHS_DEVIER | 7:0 | | UPRSMES | EORSMES | WAKEUPES | EORSTES | SOFES | MSOFES | SUSPES |
| | | 15:8 | PEP_3 | PEP_2 | PEP_1 | PEP_0 | | | | |
| | | 23:16 | | | PEP_9 | PEP_8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x1C | USBHS_DEVEPT | 7:0 | EPEN7 | EPEN6 | EPEN5 | EPEN4 | EPEN3 | EPEN2 | EPEN1 | EPEN0 |
| | | 15:8 | | | | | | | EPEN9 | EPEN8 |
| | | 23:16 | EPRST7 | EPRST6 | EPRST5 | EPRST4 | EPRST3 | EPRST2 | EPRST1 | EPRST0 |
| | | 31:24 | | | | | | | EPRST9 | EPRST8 |
| 0x20 | USBHS_DEVFNUM | 7:0 | FNUM[4:0] | | | | | MFNUM[2:0] | | |
| | | 15:8 | FNCERR | | FNUM[10:5] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 ... 0xFF | Reserved | | | | | | | | | |
| 0x0100 | USBHS_DEVEPTCFG0 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0104 | USBHS_DEVEPTCFG1 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0108 | USBHS_DEVEPTCFG2 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x010C | USBHS_DEVEPTCFG3 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0110 | USBHS_DEVEPTCFG4 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0114 | USBHS_DEVEPTCFG5 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0118 | USBHS_DEVEPTCFG6 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x011C | USBHS_DEVEPTCFG7 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0120 | USBHS_DEVEPTCFG8 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0124 | USBHS_DEVEPTCFG9 | 7:0 | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| | | 15:8 | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0128 ... 0x012F | Reserved | | | | | | | | | |
| 0x0130 | USBHS_DEVEPTISR0 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0130 | USBHS_DEVEPTISR0 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0134 | USBHS_DEVEPTISR1 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0134 | USBHS_DEVEPTISR1 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0138 | USBHS_DEVEPTISR2 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0138 | USBHS_DEVEPTISR2 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x013C | USBHS_DEVEPTISR3 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x013C | USBHS_DEVEPTISR3 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0140 | USBHS_DEVEPTISR4 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0140 | USBHS_DEVEPTISR4 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0144 | USBHS_DEVEPTISR5 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0144 | USBHS_DEVEPTISR5 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0148 | USBHS_DEVEPTISR6 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0148 | USBHS_DEVEPTISR6 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x014C | USBHS_DEVEPTISR7 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x014C | USBHS_DEVEPTISR7 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0150 | USBHS_DEVEPTISR8 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0150 | USBHS_DEVEPTISR8 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0154 | USBHS_DEVEPTISR9 | 7:0 | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0154 | USBHS_DEVEPTISR9 (ISOENPT) | 7:0 | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| | | 23:16 | BYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | BYCT[10:4] | | | | | | |
| 0x0158 ... 0x015F | Reserved | | | | | | | | | |
| 0x0160 | USBHS_DEVEPTICR0 | 7:0 | SHORTPACKETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0160 | USBHS_DEVEPTICR0 (ISOENPT) | 7:0 | SHORTPACKETC | CRCERRIC | OVERFIC | HBISOFLUSHIC | HBISOINERRIC | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0164 | USBHS_DEVEPTICR1 | 7:0 | SHORTPACKETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0164 | USBHS_DEVEPTICR1 (ISOENPT) | 7:0 | SHORTPACKETC | CRCERRIC | OVERFIC | HBISOFLUSHIC | HBISOINERRIC | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0168 | USBHS_DEVEPTICR2 | 7:0 | SHORTPACKETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0168 | USBHS_DEVEPTIC R2 (ISOENPT) | 7:0 | SHORTPACK ETC | CRCERRIC | OVERFIC | HBISOFLUSH IC | HBISOINERRI C | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x016C | USBHS_DEVEPTIC R3 | 7:0 | SHORTPACK ETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x016C | USBHS_DEVEPTIC R3 (ISOENPT) | 7:0 | SHORTPACK ETC | CRCERRIC | OVERFIC | HBISOFLUSH IC | HBISOINERRI C | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0170 | USBHS_DEVEPTIC R4 | 7:0 | SHORTPACK ETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0170 | USBHS_DEVEPTIC R4 (ISOENPT) | 7:0 | SHORTPACK ETC | CRCERRIC | OVERFIC | HBISOFLUSH IC | HBISOINERRI C | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0174 | USBHS_DEVEPTIC R5 | 7:0 | SHORTPACK ETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0174 | USBHS_DEVEPTIC R5 (ISOENPT) | 7:0 | SHORTPACK ETC | CRCERRIC | OVERFIC | HBISOFLUSH IC | HBISOINERRI C | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0178 | USBHS_DEVEPTIC R6 | 7:0 | SHORTPACK ETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0178 | USBHS_DEVEPTIC R6 (ISOENPT) | 7:0 | SHORTPACK ETC | CRCERRIC | OVERFIC | HBISOFLUSH IC | HBISOINERRI C | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x017C | USBHS_DEVEPTIC R7 | 7:0 | SHORTPACK ETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x017C | USBHS_DEVEPTIC R7 (ISOENPT) | 7:0 | SHORTPACK ETC | CRCERRIC | OVERFIC | HBISOFLUSH IC | HBISOINERRI C | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0180 | USBHS_DEVEPTICR8 | 7:0 | SHORTPACKETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0180 | USBHS_DEVEPTICR8 (ISOENPT) | 7:0 | SHORTPACKETC | CRCERRIC | OVERFIC | HBISOFLUSHIC | HBISOINERRIC | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0184 | USBHS_DEVEPTICR9 | 7:0 | SHORTPACKETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0184 | USBHS_DEVEPTICR9 (ISOENPT) | 7:0 | SHORTPACKETC | CRCERRIC | OVERFIC | HBISOFLUSHIC | HBISOINERRIC | UNDERFIC | RXOUTIC | TXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0188 ... 0x018F | Reserved | | | | | | | | | |
| 0x0190 | USBHS_DEVEPTIFR0 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0190 | USBHS_DEVEPTIFR0 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0194 | USBHS_DEVEPTIFR1 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0194 | USBHS_DEVEPTIFR1 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0198 | USBHS_DEVEPTIFR2 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0198 | USBHS_DEVEPTIFR2 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x019C | USBHS_DEVEPTIFR3 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x019C | USBHS_DEVEPTIFR3 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A0 | USBHS_DEVEPTIFR4 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A0 | USBHS_DEVEPTIFR4 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A4 | USBHS_DEVEPTIFR5 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A4 | USBHS_DEVEPTIFR5 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A8 | USBHS_DEVEPTIFR6 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01A8 | USBHS_DEVEPTIFR6 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01AC | USBHS_DEVEPTIFR7 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01AC | USBHS_DEVEPTIFR7 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B0 | USBHS_DEVEPTIFR8 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B0 | USBHS_DEVEPTIFR8 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01B4 | USBHS_DEVEPTIFR9 | 7:0 | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B4 | USBHS_DEVEPTIFR9 (ISOENPT) | 7:0 | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x01B8 ... 0x01BF | Reserved | | | | | | | | | |
| 0x01C0 | USBHS_DEVEPTIMR0 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01C0 | USBHS_DEVEPTIMR0 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01C4 | USBHS_DEVEPTIMR1 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01C4 | USBHS_DEVEPTIMR1 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01C8 | USBHS_DEVEPTIMR2 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01C8 | USBHS_DEVEPTIMR2 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01CC | USBHS_DEVEPTIMR3 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01CC | USBHS_DEVEPTIMR3 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01D0 | USBHS_DEVEPTIMR4 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01D0 | USBHS_DEVEPTIMR4 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01D4 | USBHS_DEVEPTIMR5 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01D4 | USBHS_DEVEPTIMR5 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01D8 | USBHS_DEVEPTIMR6 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01D8 | USBHS_DEVEPTIMR6 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01DC | USBHS_DEVEPTIMR7 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01DC | USBHS_DEVEPTIMR7 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01E0 | USBHS_DEVEPTIMR8 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01E0 | USBHS_DEVEPTIMR8 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01E4 | USBHS_DEVEPTIMR9 | 7:0 | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| | | 23:16 | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01E4 | USBHS_DEVEPTIMR9 (ISOENPT) | 7:0 | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| | | 15:8 | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| | | 23:16 | | | | | | RSTDT | | EPDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x01E8 ... 0x01EF | Reserved | | | | | | | | | |
| 0x01F0 | USBHS_DEVEPTIER0 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x01F0 | USBHS_DEVEPTIER0 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x01F4 | USBHS_DEVEPTIER1 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x01F4 | USBHS_DEVEPTIER1 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x01F8 | USBHS_DEVEPTIER2 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x01F8 | USBHS_DEVEPTIER2 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x01FC | USBHS_DEVEPTIER3 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01FC | USBHS_DEVEPTIER3 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0200 | USBHS_DEVEPTIER4 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0200 | USBHS_DEVEPTIER4 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0204 | USBHS_DEVEPTIER5 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0204 | USBHS_DEVEPTIER5 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0208 | USBHS_DEVEPTIER6 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0208 | USBHS_DEVEPTIER6 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x020C | USBHS_DEVEPTIER7 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x020C | USBHS_DEVEPTIER7 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0210 | USBHS_DEVEPTIER8 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0210 | USBHS_DEVEPTIER8 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0214 | USBHS_DEVEPTIER9 | 7:0 | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| | | 23:16 | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0214 | USBHS_DEVEPTIER9 (ISOENPT) | 7:0 | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| | | 15:8 | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| | | 23:16 | | | | | | RSTDTS | | EPDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0218 ... 0x021F | Reserved | | | | | | | | | |
| 0x0220 | USBHS_DEVEPTIDR0 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0220 | USBHS_DEVEPTIDR0 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0224 | USBHS_DEVEPTIDR1 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0224 | USBHS_DEVEPTIDR1 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0228 | USBHS_DEVEPTIDR2 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0228 | USBHS_DEVEPTIDR2 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x022C | USBHS_DEVEPTIDR3 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x022C | USBHS_DEVEPTIDR3 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0230 | USBHS_DEVEPTIDR4 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0230 | USBHS_DEVEPTIDR4 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0234 | USBHS_DEVEPTIDR5 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0234 | USBHS_DEVEPTIDR5 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0238 | USBHS_DEVEPTIDR6 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0238 | USBHS_DEVEPTIDR6 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x023C | USBHS_DEVEPTIDR7 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x023C | USBHS_DEVEPTIDR7 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0240 | USBHS_DEVEPTIDR8 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0240 | USBHS_DEVEPTIDR8 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0244 | USBHS_DEVEPTIDR9 | 7:0 | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0244 | USBHS_DEVEPTIDR9 (ISOENPT) | 7:0 | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| | | 23:16 | | | | | | | | EPDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0248 ... 0x02FF | Reserved | | | | | | | | | |
| 0x0300 | USBHS_DEVDMANXTDSC1 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0304 | USBHS_DEVDMAADDRESS1 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0308 | USBHS_DEVDMACONTROL1 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x030C | USBHS_DEVDMASTATUS1 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0310 | USBHS_DEVDMANXTDSC2 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0314 | USBHS_DEVDMAADDRESS2 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0318 | USBHS_DEVDMACONTROL2 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x031C | USBHS_DEVDMASTATUS2 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0320 | USBHS_DEVDMANXTDSC3 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0324 | USBHS_DEVDMAADDRESS3 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0328 | USBHS_DEVDMACONTROL3 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x032C | USBHS_DEVDMASTATUS3 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0330 | USBHS_DEVDMANXTDSC4 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0334 | USBHS_DEVDMAADDRESS4 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0338 | USBHS_DEVDMACONTROL4 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x033C | USBHS_DEVDMASTATUS4 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0340 | USBHS_DEVDMANXTDSC5 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0344 | USBHS_DEVDMAADDRESS5 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0348 | USBHS_DEVDMACONTROL5 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x034C | USBHS_DEVDMASTATUS5 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0350 | USBHS_DEVDMANXTDSC6 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0354 | USBHS_DEVDMAADDRESS6 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0358 | USBHS_DEVDMACONTROL6 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x035C | USBHS_DEVDMASTATUS6 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0360 | USBHS_DEVDMANXTDSC7 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0364 | USBHS_DEVDMAADDRESS7 | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0368 | USBHS_DEVDMACONTROL7 | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x036C | USBHS_DEVDMASTATUS7 | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0370 ... 0x03FF | Reserved | | | | | | | | | |
| 0x0400 | USBHS_HSTCTRL | 7:0 | | | | | | | | |
| | | 15:8 | | | SPDCONF[1:0] | | | RESUME | RESET | SOFE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0404 | USBHS_HSTISR | 7:0 | | HWUPI | HSOFI | RXRSMI | RSMEDI | RSTI | DDISCI | DCONNI |
| | | 15:8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 | PEP_3 | PEP_2 | PEP_1 | PEP_0 |
| | | 23:16 | | | | | | | PEP_9 | PEP_8 |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0408 | USBHS_HSTICR | 7:0 | | HWUPIC | HSOFIC | RXRSMIC | RSMEDIC | RSTIC | DDISCIC | DCONNIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x040C | USBHS_HSTIFR | 7:0 | | HWUPIS | HSOFIS | RXRSMIS | RSMEDIS | RSTIS | DDISCIS | DCONNIS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x0410 | USBHS_HSTIMR | 7:0 | | HWUPIE | HSOFIE | RXRSMIE | RSMEDIE | RSTIE | DDISCIE | DCONNIE |
| | | 15:8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 | PEP_3 | PEP_2 | PEP_1 | PEP_0 |
| | | 23:16 | | | | | | | PEP_9 | PEP_8 |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x0414 | USBHS_HSTIDR | 7:0 | | HWUPIEC | HSOFIEC | RXRSMIEC | RSMEDIEC | RSTIEC | DDISCIEC | DCONNIEC |
| | | 15:8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 | PEP_3 | PEP_2 | PEP_1 | PEP_0 |
| | | 23:16 | | | | | | | PEP_9 | PEP_8 |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x0418 | USBHS_HSTIER | 7:0 | | HWUPIES | HSOFIES | RXRSMIES | RSMEDIES | RSTIES | DDISCIES | DCONNIES |
| | | 15:8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 | PEP_3 | PEP_2 | PEP_1 | PEP_0 |
| | | 23:16 | | | | | | | PEP_9 | PEP_8 |
| | | 31:24 | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| 0x041C | USBHS_HSTPIP | 7:0 | PEN7 | PEN6 | PEN5 | PEN4 | PEN3 | PEN2 | PEN1 | PEN0 |
| | | 15:8 | | | | | | | | PEN8 |
| | | 23:16 | PRST7 | PRST6 | PRST5 | PRST4 | PRST3 | PRST2 | PRST1 | PRST0 |
| | | 31:24 | | | | | | | | PRST8 |
| 0x0420 | USBHS_HSTFNUM | 7:0 | FNUM[4:0] | | | | | MFNUM[2:0] | | |
| | | 15:8 | | | FNUM[10:5] | | | | | |
| | | 23:16 | FLENHIGH[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0424 | USBHS_HSTADDR1 | 7:0 | HSTADDRP0[6:0] | | | | | | | |
| | | 15:8 | HSTADDRP1[6:0] | | | | | | | |
| | | 23:16 | HSTADDRP2[6:0] | | | | | | | |
| | | 31:24 | HSTADDRP3[6:0] | | | | | | | |
| 0x0428 | USBHS_HSTADDR2 | 7:0 | HSTADDRP4[6:0] | | | | | | | |
| | | 15:8 | HSTADDRP5[6:0] | | | | | | | |
| | | 23:16 | HSTADDRP6[6:0] | | | | | | | |
| | | 31:24 | HSTADDRP7[6:0] | | | | | | | |
| 0x042C | USBHS_HSTADDR3 | 7:0 | HSTADDRP8[6:0] | | | | | | | |
| | | 15:8 | HSTADDRP9[6:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0430 ... 0x04FF | Reserved | | | | | | | | | |
| 0x0500 | USBHS_HSTPIPCFG0 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x0500 | USBHS_HSTPIPCFG0 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |
| 0x0504 | USBHS_HSTPIPCFG1 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x0504 | USBHS_HSTPIPCFG1 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0508 | USBHS_HSTPIPCFG2 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x0508 | USBHS_HSTPIPCFG2 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |
| 0x050C | USBHS_HSTPIPCFG3 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x050C | USBHS_HSTPIPCFG3 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |
| 0x0510 | USBHS_HSTPIPCFG4 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x0510 | USBHS_HSTPIPCFG4 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |
| 0x0514 | USBHS_HSTPIPCFG5 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x0514 | USBHS_HSTPIPCFG5 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |
| 0x0518 | USBHS_HSTPIPCFG6 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x0518 | USBHS_HSTPIPCFG6 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |
| 0x051C | USBHS_HSTPIPCFG7 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x051C | USBHS_HSTPIPCFG7 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |
| 0x0520 | USBHS_HSTPIPCFG8 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x0520 | USBHS_HSTPIPCFG8 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0524 | USBHS_HSTPIPCFG9 | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | | PEPNUM[3:0] | | | |
| | | 31:24 | INTFRQ[7:0] | | | | | | | |
| 0x0524 | USBHS_HSTPIPCFG9 (HSBOHSCP) | 7:0 | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| | | 15:8 | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| | | 23:16 | | | | PINGEN | PEPNUM[3:0] | | | |
| | | 31:24 | BINTERVAL[7:0] | | | | | | | |
| 0x0528 ... 0x052F | Reserved | | | | | | | | | |
| 0x0530 | USBHS_HSTPIPISR0 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0530 | USBHS_HSTPIPISR0 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0530 | USBHS_HSTPIPISR0 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0534 | USBHS_HSTPIPISR1 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0534 | USBHS_HSTPIPISR1 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0534 | USBHS_HSTPIPISR1 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0538 | USBHS_HSTPIPISR2 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0538 | USBHS_HSTPIPISR2 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0538 | USBHS_HSTPIPISR2 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x053C | USBHS_HSTPIPISR3 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x053C | USBHS_HSTPIPISR3 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x053C | USBHS_HSTPIPISR3 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0540 | USBHS_HSTPIPISR4 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0540 | USBHS_HSTPIPISR4 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0540 | USBHS_HSTPIPISR4 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0544 | USBHS_HSTPIPISR5 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0544 | USBHS_HSTPIPISR5 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0544 | USBHS_HSTPIPISR5 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0548 | USBHS_HSTPIPISR6 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0548 | USBHS_HSTPIPISR6 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0548 | USBHS_HSTPIPISR6 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x054C | USBHS_HSTPIPISR7 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x054C | USBHS_HSTPIPISR7 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x054C | USBHS_HSTPIPISR7 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0550 | USBHS_HSTPIPISR8 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0550 | USBHS_HSTPIPISR8 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0550 | USBHS_HSTPIPISR8 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0554 | USBHS_HSTPIPISR9 | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0554 | USBHS_HSTPIPISR9 (INTPIPES) | 7:0 | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0554 | USBHS_HSTPIPISR9 (ISOPIPES) | 7:0 | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| | | 15:8 | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| | | 23:16 | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| | | 31:24 | | PBYCT[10:4] | | | | | | |
| 0x0558 ... 0x055F | Reserved | | | | | | | | | |
| 0x0560 | USBHS_HSTPIPICR0 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0560 | USBHS_HSTPIPICR0 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0560 | USBHS_HSTPIPICR0 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0564 | USBHS_HSTPIPICR1 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0564 | USBHS_HSTPIPICR1 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0564 | USBHS_HSTPIPICR1 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0568 | USBHS_HSTPIPICR2 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0568 | USBHS_HSTPIPICR2 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0568 | USBHS_HSTPIPICR2 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x056C | USBHS_HSTPIPICR3 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x056C | USBHS_HSTPIPICR3 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x056C | USBHS_HSTPIPICR3 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0570 | USBHS_HSTPIPICR4 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0570 | USBHS_HSTPIPICR4 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0570 | USBHS_HSTPIPICR4 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0574 | USBHS_HSTPIPICR5 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0574 | USBHS_HSTPIPICR5 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0574 | USBHS_HSTPIPICR5 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0578 | USBHS_HSTPIPICR6 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0578 | USBHS_HSTPIPICR6 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0578 | USBHS_HSTPIPICR6 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x057C | USBHS_HSTPIPICR7 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x057C | USBHS_HSTPIPICR7 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x057C | USBHS_HSTPIPICR7 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0580 | USBHS_HSTPIPICR8 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0580 | USBHS_HSTPIPICR8 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0580 | USBHS_HSTPIPICR8 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0584 | USBHS_HSTPIPICR9 | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0584 | USBHS_HSTPIPICR9 (INTPIPES) | 7:0 | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0584 | USBHS_HSTPIPICR9 (ISOPIPES) | 7:0 | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0588 ... 0x058F | Reserved | | | | | | | | | |
| 0x0590 | USBHS_HSTPIPIFRx | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | TXSTPIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0590 | USBHS_HSTPIPIFR0 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0590 | USBHS_HSTPIPIFR0 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0594 | USBHS_HSTPIPIFR1 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0594 | USBHS_HSTPIPIFR1 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0598 | USBHS_HSTPIPIFR2 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0598 | USBHS_HSTPIPIFR2 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x059C | USBHS_HSTPIPIFR3 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x059C | USBHS_HSTPIPIFR3 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05A0 | USBHS_HSTPIPIFR4 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05A0 | USBHS_HSTPIPIFR4 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05A4 | USBHS_HSTPIPIFR5 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05A4 | USBHS_HSTPIPIFR5 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05A8 | USBHS_HSTPIPIFR6 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05A8 | USBHS_HSTPIPIFR6 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x05AC | USBHS_HSTPIPIFR7 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05AC | USBHS_HSTPIPIFR7 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05B0 | USBHS_HSTPIPIFR8 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05B0 | USBHS_HSTPIPIFR8 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05B4 | USBHS_HSTPIPIFR9 (INTPIPES) | 7:0 | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05B4 | USBHS_HSTPIPIFR9 (ISOPIPES) | 7:0 | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| | | 15:8 | | | | NBUSYBKS | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x05B8 ... 0x05BF | Reserved | | | | | | | | | |
| 0x05C0 | USBHS_HSTPIPIMR0 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05C0 | USBHS_HSTPIPIMR0 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05C0 | USBHS_HSTPIPIMR0 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05C4 | USBHS_HSTPIPIMR1 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05C4 | USBHS_HSTPIPIMR1 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x05C4 | USBHS_HSTPIPIMR1 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05C8 | USBHS_HSTPIPIMR2 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05C8 | USBHS_HSTPIPIMR2 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05C8 | USBHS_HSTPIPIMR2 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05CC | USBHS_HSTPIPIMR3 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05CC | USBHS_HSTPIPIMR3 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05CC | USBHS_HSTPIPIMR3 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05D0 | USBHS_HSTPIPIMR4 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05D0 | USBHS_HSTPIPIMR4 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05D0 | USBHS_HSTPIPIMR4 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05D4 | USBHS_HSTPIPIMR5 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x05D4 | USBHS_HSTPIPIMR5 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05D4 | USBHS_HSTPIPIMR5 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05D8 | USBHS_HSTPIPIMR6 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05D8 | USBHS_HSTPIPIMR6 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05D8 | USBHS_HSTPIPIMR6 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05DC | USBHS_HSTPIPIMR7 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05DC | USBHS_HSTPIPIMR7 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05DC | USBHS_HSTPIPIMR7 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05E0 | USBHS_HSTPIPIMR8 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05E0 | USBHS_HSTPIPIMR8 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05E0 | USBHS_HSTPIPIMR8 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x05E4 | USBHS_HSTPIPIMR9 | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05E4 | USBHS_HSTPIPIMR9 (INTPIPES) | 7:0 | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05E4 | USBHS_HSTPIPIMR9 (ISOPIPES) | 7:0 | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| | | 15:8 | | FIFOCON | | NBUSYBKE | | | | |
| | | 23:16 | | | | | | RSTDT | PFREEZE | PDISHDMA |
| | | 31:24 | | | | | | | | |
| 0x05E8 ... 0x05EF | Reserved | | | | | | | | | |
| 0x05F0 | USBHS_HSTPIPIER0 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05F0 | USBHS_HSTPIPIER0 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05F0 | USBHS_HSTPIPIER0 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05F4 | USBHS_HSTPIPIER1 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05F4 | USBHS_HSTPIPIER1 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05F4 | USBHS_HSTPIPIER1 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05F8 | USBHS_HSTPIPIER2 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05F8 | USBHS_HSTPIPIER2 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x05F8 | USBHS_HSTPIPIER2 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05FC | USBHS_HSTPIPIER3 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05FC | USBHS_HSTPIPIER3 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x05FC | USBHS_HSTPIPIER3 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0600 | USBHS_HSTPIPIER4 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0600 | USBHS_HSTPIPIER4 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0600 | USBHS_HSTPIPIER4 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0604 | USBHS_HSTPIPIER5 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0604 | USBHS_HSTPIPIER5 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0604 | USBHS_HSTPIPIER5 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0608 | USBHS_HSTPIPIER6 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0608 | USBHS_HSTPIPIER6 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0608 | USBHS_HSTPIPIER6 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x060C | USBHS_HSTPIPIER7 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x060C | USBHS_HSTPIPIER7 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x060C | USBHS_HSTPIPIER7 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0610 | USBHS_HSTPIPIER8 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0610 | USBHS_HSTPIPIER8 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0610 | USBHS_HSTPIPIER8 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0614 | USBHS_HSTPIPIER9 | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0614 | USBHS_HSTPIPIER9 (INTPIPES) | 7:0 | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0614 | USBHS_HSTPIPIER9 (ISOPIPES) | 7:0 | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| | | 15:8 | | | | NBUSYBKES | | | | |
| | | 23:16 | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| | | 31:24 | | | | | | | | |
| 0x0618 ... 0x061F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0620 | USBHS_HSTPIPIDR0 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0620 | USBHS_HSTPIPIDR0 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0620 | USBHS_HSTPIPIDR0 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0624 | USBHS_HSTPIPIDR1 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0624 | USBHS_HSTPIPIDR1 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0624 | USBHS_HSTPIPIDR1 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0628 | USBHS_HSTPIPIDR2 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0628 | USBHS_HSTPIPIDR2 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0628 | USBHS_HSTPIPIDR2 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x062C | USBHS_HSTPIPIDR3 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x062C | USBHS_HSTPIPIDR3 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x062C | USBHS_HSTPIPIDR3 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0630 | USBHS_HSTPIPIDR4 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0630 | USBHS_HSTPIPIDR4 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0630 | USBHS_HSTPIPIDR4 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0634 | USBHS_HSTPIPIDR5 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0634 | USBHS_HSTPIPIDR5 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0634 | USBHS_HSTPIPIDR5 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0638 | USBHS_HSTPIPIDR6 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0638 | USBHS_HSTPIPIDR6 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0638 | USBHS_HSTPIPIDR6 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x063C | USBHS_HSTPIPIDR7 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x063C | USBHS_HSTPIPIDR7 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x063C | USBHS_HSTPIPIDR7 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0640 | USBHS_HSTPIPIDR8 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0640 | USBHS_HSTPIPIDR8 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0640 | USBHS_HSTPIPIDR8 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0644 | USBHS_HSTPIPIDR9 | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0644 | USBHS_HSTPIPIDR9 (INTPIPES) | 7:0 | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0644 | USBHS_HSTPIPIDR9 (ISOPIPES) | 7:0 | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| | | 15:8 | | FIFOCONC | | NBUSYBKEC | | | | |
| | | 23:16 | | | | | | | PFREEZEC | PDISHDMAC |
| | | 31:24 | | | | | | | | |
| 0x0648 ... 0x064F | Reserved | | | | | | | | | |
| 0x0650 | USBHS_HSTPIPINRQ0 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0654 | USBHS_HSTPIPINRQ1 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0658 | USBHS_HSTPIPINRQ2 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x065C | USBHS_HSTPIPINRQ3 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0660 | USBHS_HSTPIPINRQ4 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0664 | USBHS_HSTPIPINRQ5 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0668 | USBHS_HSTPIPINRQ6 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x066C | USBHS_HSTPIPINRQ7 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0670 | USBHS_HSTPIPINRQ8 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0674 | USBHS_HSTPIPINRQ9 | 7:0 | INRQ[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | INMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0678 ... 0x067F | Reserved | | | | | | | | | |
| 0x0680 | USBHS_HSTPIPERR0 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0684 | USBHS_HSTPIPERR1 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0688 | USBHS_HSTPIPERR2 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x068C | USBHS_HSTPIPERR3 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0690 | USBHS_HSTPIPERR4 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0694 | USBHS_HSTPIPERR5 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0698 | USBHS_HSTPIPERR6 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x069C | USBHS_HSTPIPERR7 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x06A0 | USBHS_HSTPIPERR8 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06A4 | USBHS_HSTPIPERR9 | 7:0 | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x06A8 ... 0x06FF | Reserved | | | | | | | | | |
| 0x0700 | USBHS_HSTDMANXTDSC1 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0704 | USBHS_HSTDMAADDRESSx | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0708 | USBHS_HSTDMACONTROLx | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x070C | USBHS_HSTDMASTATUSx | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0710 | USBHS_HSTDMANXTDSC2 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0714 | USBHS_HSTDMAADDRESSx | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0718 | USBHS_HSTDMACONTROLx | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x071C | USBHS_HSTDMASTATUSx | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0720 | USBHS_HSTDMANXTDSC3 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0724 | USBHS_HSTDMAADDRESSx | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0728 | USBHS_HSTDMACONTROLx | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x072C | USBHS_HSTDMASTATUSx | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0730 | USBHS_HSTDMAN XTDSC4 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0734 | USBHS_HSTDMAA DDRESSx | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0738 | USBHS_HSTDMAC ONTROLx | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x073C | USBHS_HSTDMAS TATUSx | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0740 | USBHS_HSTDMAN XTDSC5 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0744 | USBHS_HSTDMAA DDRESSx | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0748 | USBHS_HSTDMAC ONTROLx | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x074C | USBHS_HSTDMAS TATUSx | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0750 | USBHS_HSTDMAN XTDSC6 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0754 | USBHS_HSTDMAA DDRESSx | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |
| 0x0758 | USBHS_HSTDMAC ONTROLx | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x075C | USBHS_HSTDMAS TATUSx | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0760 | USBHS_HSTDMAN XTDSC7 | 7:0 | NXT_DSC_ADD[7:0] | | | | | | | |
| | | 15:8 | NXT_DSC_ADD[15:8] | | | | | | | |
| | | 23:16 | NXT_DSC_ADD[23:16] | | | | | | | |
| | | 31:24 | NXT_DSC_ADD[31:24] | | | | | | | |
| 0x0764 | USBHS_HSTDMAA DDRESSx | 7:0 | BUFF_ADD[7:0] | | | | | | | |
| | | 15:8 | BUFF_ADD[15:8] | | | | | | | |
| | | 23:16 | BUFF_ADD[23:16] | | | | | | | |
| | | 31:24 | BUFF_ADD[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0768 | USBHS_HSTDMACONTROLx | 7:0 | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_LENGTH[7:0] | | | | | | | |
| | | 31:24 | BUFF_LENGTH[15:8] | | | | | | | |
| 0x076C | USBHS_HSTDMASTATUSx | 7:0 | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| | | 15:8 | | | | | | | | |
| | | 23:16 | BUFF_COUNT[7:0] | | | | | | | |
| | | 31:24 | BUFF_COUNT[15:8] | | | | | | | |
| 0x0770 ... 0x07FF | Reserved | | | | | | | | | |
| 0x0800 | USBHS_CTRL | 7:0 | | | | RDERRE | | | | |
| | | 15:8 | USBE | FRZCLK | | | | | | VBUSHWC |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | UIMOD | UID |
| 0x0804 | USBHS_SR | 7:0 | | | | RDERRI | | | | |
| | | 15:8 | | CLKUSABLE | SPEED[1:0] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0808 | USBHS_SCR | 7:0 | | | | RDERRIC | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x080C | USBHS_SFR | 7:0 | | | | RDERRIS | | | | |
| | | 15:8 | | | | | | | VBUSRQS | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 39.6.1 General Control Register

**Name:** USBHS_CTRL
**Offset:** 0x0800
**Reset:** 0x03004000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | UIMOD | UID |
| Access | | | | | | | | |
| Reset | | | | | | | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | USBE | FRZCLK | | | | | | VBUSHWC |
| Access | | | | | | | | |
| Reset | 0 | 1 | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | RDERRE | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | |

**Bit 25 – UIMOD** USBHS Mode
0 (HOST): The module is in USB Host mode.
1 (DEVICE): The module is in USB Device mode.
This bit can be written even if USBE = 0 or FRZCLK = 1. Disabling the USBHS (by writing a zero to the USBE bit) does not reset this bit.

**Bit 24 – UID** UID Pin Enable
Must be set to ‘0’.

**Bit 15 – USBE** USBHS Enable
Writing a zero to this bit resets the USBHS, disables the USB transceiver, and disables the USBHS clock inputs. Unless explicitly stated, all registers then become read-only and are reset.
This bit can be written even if FRZCLK = 1

| Value | Description |
|---|---|
| 0 | The USBHS is disabled. |
| 1 | The USBHS is enabled. |

**Bit 14 – FRZCLK** Freeze USB Clock
This bit can be written even if USBE = 0. Disabling the USBHS (by writing a zero to the USBE bit) does not reset this bit, but it freezes the clock inputs whatever its value.

| Value | Description |
|---|---|
| 0 | The clock inputs are enabled. |
| 1 | The clock inputs are disabled (the resume detection is still active). This reduces the power consumption. Unless explicitly stated, all registers then become read-only. |

**Bit 8 – VBUSHWC** VBUS Hardware Control
Must be set to ‘1’.

| Value | Description |
|---|---|
| 0 | The hardware control over the VBOF output pin is enabled. The USBHS resets the VBOF output pin when a VBUS problem occurs. |
| 1 | The hardware control over the VBOF output pin is disabled. |
| 0 | The hardware control over the PIO line is enabled. The USBHS resets the PIO output pin when a VBUS problem occurs. |
| 1 | The hardware control over the PIO line is disabled. |

**Bit 4 – RDERRE** Remote Device Connection Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | The Remote Device Connection Error Interrupt (USBHS_SR.RDERRI) is disabled. |
| 1 | The Remote Device Connection Error Interrupt (USBHS_SR.RDERRI) is enabled. |

### 39.6.2 General Status Register

**Name:** USBHS_SR
**Offset:** 0x0804
**Reset:** 0x00000400
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | CLKUSABLE | SPEED[1:0] | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | RDERRI | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | |

**Bit 14 – CLKUSABLE** UTMI Clock Usable

| Value | Description |
|---|---|
| 0 | Cleared when the UTMI 30 MHz is not usable. |
| 1 | Set when the UTMI 30 MHz is usable. |

**Bits 13:12 – SPEED[1:0]** Speed Status (Device mode only)
This field is set according to the controller speed mode.

| Value | Name | Description |
|---|---|---|
| 0 | FULL_SPEED | Full-Speed mode |
| 1 | HIGH_SPEED | High-Speed mode |
| 2 | LOW_SPEED | Low-Speed mode |
| 3 | Reserved | |

**Bit 4 – RDERRI** Remote Device Connection Error Interrupt (Host mode only)

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_SCR.RDERRIC = 1. |
| 1 | Set when an error occurs during the remote device connection. This triggers a USB interrupt if USBHS_CTRL.RDERRE = 1. |

### 39.6.3 General Status Clear Register

**Name:** USBHS_SCR
**Offset:** 0x0808
**Property:** Write-only

This register always reads as zero.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | RDERRIC | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 4 – RDERRIC** Remote Device Connection Error Interrupt Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the RDERRI bit in USBHS_SR. |

### 39.6.4 General Status Set Register

**Name:** USBHS_SFR
**Offset:** 0x080C
**Property:** Write-only

This register always reads as zero.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | VBUSRQS | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | RDERRIS | | | | |
| Reset | | | | | | | | |

**Bit 9 – VBUSRQS** VBUS Request Set
Must be set to ‘1’.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Sets the VBUSRQ bit in USBHS_SR. |

**Bit 4 – RDERRIS** Remote Device Connection Error Interrupt Set

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Sets the RDERRI bit in USBHS_SR, which may be useful for test or debug purposes. |

### 39.6.5 Device General Control Register

**Name:** USBHS_DEVCTRL
**Offset:** 0x0000
**Reset:** 0x00000100
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | OPMODE2 |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TSTPCKT | TSTK | TSTJ | LS | SPDCONF[1:0] | | RMWKUP | DETACH |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDEN | UADD[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 16 – OPMODE2** Specific Operational mode

| Value | Description |
|---|---|
| 0 | The UTMI transceiver is in Normal operating mode. |
| 1 | The UTMI transceiver is in the “Disable bit stuffing and NRZI encoding” operational mode for test purposes. |

**Bit 15 – TSTPCKT** Test packet mode

| Value | Description |
|---|---|
| 0 | The UTMI transceiver is in Normal operating mode. |
| 1 | The UTMI transceiver generates test packets for test purposes. |

**Bit 14 – TSTK** Test mode K

| Value | Description |
|---|---|
| 0 | The UTMI transceiver is in Normal operating mode. |
| 1 | The UTMI transceiver generates high-speed K state for test purposes. |

**Bit 13 – TSTJ** Test mode J

| Value | Description |
|---|---|
| 0 | The UTMI transceiver is in Normal operating mode. |
| 1 | The UTMI transceiver generates high-speed J state for test purposes. |

**Bit 12 – LS** Low-Speed Mode Force
This bit can be written even if USBHS_CTRL.USBE = 0 or USBHS_CTRL.FRZCLK = 1. Disabling the USBHS (by writing a zero to the USBHS_CTRL.USBE bit) does not reset this bit.

| Value | Description |
|---|---|
| 0 | The Full-speed mode is active. |
| 1 | The Low-speed mode is active. |

**Bits 11:10 – SPDCONF[1:0]** Mode Configuration
This field contains the peripheral speed:

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | The peripheral starts in Full-speed mode and performs a high-speed reset to switch to High-speed mode if the host is high-speed-capable. |
| 1 | LOW_POWER | For a better consumption, if high speed is not needed. |
| 2 | HIGH_SPEED | Forced high speed. |
| 3 | FORCED_FS | The peripheral remains in Full-speed mode whatever the host speed capability. |

**Bit 9 – RMWKUP** Remote Wakeup
This bit is cleared when the USBHS receives a USB reset or once the upstream resume has been sent.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Sends an upstream resume to the host for a remote wakeup. |

**Bit 8 – DETACH** Detach

| Value | Description |
|---|---|
| 0 | Reconnects the device. |
| 1 | Physically detaches the device (disconnects the internal pull-up resistor from D+ and D-). |

**Bit 7 – ADDEN** Address Enable
This bit is cleared when a USB reset is received.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Activates the UADD field (USB address). |

**Bits 6:0 – UADD[6:0]** USB Address
This field contains the device address.
This field is cleared when a USB reset is received.

### 39.6.6 Device Global Interrupt Status Register

**Name:** USBHS_DEVISR
**Offset:** 0x0004
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | PEP_9 | PEP_8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEP_3 | PEP_2 | PEP_1 | PEP_0 | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | UPRSM | EORSM | WAKEUP | EORST | SOF | MSOF | SUSP |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVDMASTATUSx interrupt source is cleared. |
| 1 | Set when an interrupt is triggered by the DMA channel x. This triggers a USB interrupt if DMA_x = 1. |

**Bits 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 – PEP_** Endpoint x Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the interrupt source is serviced. |
| 1 | Set when an interrupt is triggered by endpoint x (USBHS_DEVEPTISRx, USBHS_DEVEPTIMRx). This triggers a USB interrupt if USBHS_DEVIMR.PEP_x = 1. |

**Bit 6 – UPRSM** Upstream Resume Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVICR.UPRSMC bit is written to one to acknowledge the interrupt (USB clock inputs must be enabled before). |
| 1 | Set when the USBHS sends a resume signal called “Upstream Resume”. This triggers a USB interrupt if USBHS_DEVIMR.UPRSME = 1. |

**Bit 5 – EORSM** End of Resume Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVICR.EORSMC bit is written to one to acknowledge the interrupt. |
| 1 | Set when the USBHS detects a valid “End of Resume” signal initiated by the host. This triggers a USB interrupt if USBHS_DEVIMR.EORSME = 1. |

**Bit 4 – WAKEUP** Wakeup Interrupt
This interrupt is generated even if the clock is frozen by the USBHS_CTRL.FRZCLK bit.

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVICR.WAKEUPC bit is written to one to acknowledge the interrupt (USB clock inputs must be enabled before), or when the Suspend (SUSP) interrupt bit is set. |

| Value | Description |
|---|---|
| 1 | Set when the USBHS is reactivated by a filtered non-idle signal from the lines (not by an upstream resume). This triggers an interrupt if USBHS_DEVIMR.WAKEUPE = 1. |

**Bit 3 – EORST** End of Reset Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVICR.EORSTC bit is written to one to acknowledge the interrupt. |
| 1 | Set when a USB “End of Reset” has been detected. This triggers a USB interrupt if USBHS_DEVIMR.EORSTE = 1. |

**Bit 2 – SOF** Start of Frame Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVICR.SOFC bit is written to one to acknowledge the interrupt. |
| 1 | Set when a USB “Start of Frame” PID (SOF) has been detected (every 1 ms). This triggers a USB interrupt if SOFE = 1. The FNUM field is updated. In High-speed mode, the MFNUM field is cleared. |

**Bit 1 – MSOF** Micro Start of Frame Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVICR.MSOFC bit is written to one to acknowledge the interrupt. |
| 1 | Set in High-speed mode when a USB “Micro Start of Frame” PID (SOF) has been detected (every 125 µs). This triggers a USB interrupt if MSOFE = 1. The MFNUM field is updated. The FNUM field is unchanged. |

**Bit 0 – SUSP** Suspend Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVICR.SUSPC bit is written to one to acknowledge the interrupt, or when the Wakeup (WAKEUP) interrupt bit is set. |
| 1 | Set when a USB “Suspend” idle bus state has been detected for 3 frame periods (J state for 3 ms). This triggers a USB interrupt if USBHS_DEVIMR.SUSPE = 1. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.7 Device Global Interrupt Clear Register

**Name:** USBHS_DEVICR
**Offset:** 0x0008
**Property:** Write-only

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_DEVISR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | UPRSMC | EORSMC | WAKEUPC | EORSTC | SOFC | MSOFC | SUSPC |
| Reset | | | | | | | | |

**Bit 6 – UPRSMC** Upstream Resume Interrupt Clear

**Bit 5 – EORSMC** End of Resume Interrupt Clear

**Bit 4 – WAKEUPC** Wakeup Interrupt Clear

**Bit 3 – EORSTC** End of Reset Interrupt Clear

**Bit 2 – SOFC** Start of Frame Interrupt Clear

**Bit 1 – MSOFC** Micro Start of Frame Interrupt Clear

**Bit 0 – SUSPC** Suspend Interrupt Clear

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.8 Device Global Interrupt Set Register

**Name:** USBHS_DEVIFR
**Offset:** 0x000C
**Property:** Write-only

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_DEVISR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | UPRSMS | EORSMS | WAKEUPS | EORSTS | SOFS | MSOFS | SUSPS |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt Set

**Bit 6 – UPRSMS** Upstream Resume Interrupt Set

**Bit 5 – EORSMS** End of Resume Interrupt Set

**Bit 4 – WAKEUPS** Wakeup Interrupt Set

**Bit 3 – EORSTS** End of Reset Interrupt Set

**Bit 2 – SOFS** Start of Frame Interrupt Set

**Bit 1 – MSOFS** Micro Start of Frame Interrupt Set

**Bit 0 – SUSPS** Suspend Interrupt Set

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.9 Device Global Interrupt Mask Register

**Name:** USBHS_DEVIMR
**Offset:** 0x0010
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | PEP_9 | PEP_8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEP_3 | PEP_2 | PEP_1 | PEP_0 | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | UPRSME | EORSME | WAKEUPE | EORSTE | SOFE | MSOFE | SUSPE |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt Mask

**Bits 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 – PEP_** Endpoint x Interrupt Mask

**Bit 6 – UPRSME** Upstream Resume Interrupt Mask

**Bit 5 – EORSME** End of Resume Interrupt Mask

**Bit 4 – WAKEUPE** Wakeup Interrupt Mask

**Bit 3 – EORSTE** End of Reset Interrupt Mask

**Bit 2 – SOFE** Start of Frame Interrupt Mask

**Bit 1 – MSOFE** Micro Start of Frame Interrupt Mask

**Bit 0 – SUSPE** Suspend Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.10 Device Global Interrupt Disable Register

**Name:** USBHS_DEVIDR
**Offset:** 0x0014
**Property:** Write-only

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_DEVIMR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | PEP_9 | PEP_8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEP_3 | PEP_2 | PEP_1 | PEP_0 | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | UPRSMEC | EORSMEC | WAKEUPEC | EORSTEC | SOFEC | MSOFEC | SUSPEC |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt Disable

**Bits 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 – PEP_** Endpoint x Interrupt Disable

**Bit 6 – UPRSMEC** Upstream Resume Interrupt Disable

**Bit 5 – EORSMEC** End of Resume Interrupt Disable

**Bit 4 – WAKEUPEC** Wakeup Interrupt Disable

**Bit 3 – EORSTEC** End of Reset Interrupt Disable

**Bit 2 – SOFEC** Start of Frame Interrupt Disable

**Bit 1 – MSOFEC** Micro Start of Frame Interrupt Disable

**Bit 0 – SUSPEC** Suspend Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.11 Device Global Interrupt Enable Register

**Name:** USBHS_DEVIER
**Offset:** 0x0018
**Property:** Write-only

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_DEVIMR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | PEP_9 | PEP_8 | PEP_7 | PEP_6 | PEP_5 | PEP_4 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEP_3 | PEP_2 | PEP_1 | PEP_0 | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | UPRSMES | EORSMES | WAKEUPES | EORSTES | SOFES | MSOFES | SUSPES |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt Enable

**Bits 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 – PEP_** Endpoint x Interrupt Enable

**Bit 6 – UPRSMES** Upstream Resume Interrupt Enable

**Bit 5 – EORSMES** End of Resume Interrupt Enable

**Bit 4 – WAKEUPES** Wakeup Interrupt Enable

**Bit 3 – EORSTES** End of Reset Interrupt Enable

**Bit 2 – SOFES** Start of Frame Interrupt Enable

**Bit 1 – MSOFES** Micro Start of Frame Interrupt Enable

**Bit 0 – SUSPES** Suspend Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.12 Device Endpoint Register

**Name:** USBHS_DEVEPT
**Offset:** 0x001C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EPRST9 | EPRST8 |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EPRST7 | EPRST6 | EPRST5 | EPRST4 | EPRST3 | EPRST2 | EPRST1 | EPRST0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EPEN9 | EPEN8 |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EPEN7 | EPEN6 | EPEN5 | EPEN4 | EPEN3 | EPEN2 | EPEN1 | EPEN0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23, 24, 25 – EPRST** Endpoint x Reset
The whole endpoint mechanism (FIFO counter, reception, transmission, etc.) is reset apart from the Data Toggle Sequence field (USBHS_DEVEPTISRx.DTSEQ), which can be cleared by setting the USBHS_DEVEPTIMRx.RSTDT bit (by writing a one to the USBHS_DEVEPTIERx.RSTDTS bit).
The endpoint configuration remains active and the endpoint is still enabled.
This bit is cleared upon receiving a USB reset.

| Value | Description |
|---|---|
| 0 | Completes the reset operation and starts using the FIFO. |
| 1 | Resets the endpoint x FIFO prior to any other operation, upon hardware reset or when a USB bus reset has been received. This resets the endpoint x registers (USBHS_DEVEPTCFGx, USBHS_DEVEPTISRx, USBHS_DEVEPTIMRx) but not the endpoint configuration (USBHS_DEVEPTCFGx.ALLOC, USBHS_DEVEPTCFGx.EPBK, USBHS_DEVEPTCFGx.EPSIZE, USBHS_DEVEPTCFGx.EPDIR, USBHS_DEVEPTCFGx.EPTYPE). |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 – EPEN** Endpoint x Enable

| Value | Description |
|---|---|
| 0 | Endpoint x is disabled, forcing the endpoint x state to inactive (no answer to USB requests) and resetting the endpoint x registers (USBHS_DEVEPTCFGx, USBHS_DEVEPTISRx, USBHS_DEVEPTIMRx) but not the endpoint configuration (USBHS_DEVEPTCFGx.ALLOC, USBHS_DEVEPTCFGx.EPBK, USBHS_DEVEPTCFGx.EPSIZE, USBHS_DEVEPTCFGx.EPDIR, USBHS_DEVEPTCFGx.EPTYPE). |
| 1 | Endpoint x is enabled. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 39.6.13 Device Frame Number Register

**Name:** USBHS_DEVFNUM
**Offset:** 0x0020
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FNCERR | | FNUM[10:5] | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FNUM[4:0] | | | | | MFNUM[2:0] | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – FNCERR** Frame Number CRC Error

| Value | Description |
|---|---|
| 0 | Cleared upon receiving a USB reset. |
| 1 | Set when a corrupted frame number (or microframe number) is received. This bit and the SOF (or MSOF) interrupt bit are updated at the same time. |

**Bits 13:3 – FNUM[10:0]** Frame Number
This field contains the 11-bit frame number information. It is provided in the last received SOF packet.
This field is cleared upon receiving a USB reset.
FNUM is updated even if a corrupted SOF is received.

**Bits 2:0 – MFNUM[2:0]** Micro Frame Number
This field contains the 3-bit micro frame number information. It is provided in the last received MSOF packet.
This field is cleared at the beginning of each start of frame (SOF interrupt) or upon receiving a USB reset.
MFNUM is updated even if a corrupted MSOF is received.

### 39.6.14 Device Endpoint x Configuration Register

**Name:** USBHS_DEVEPTCFGx
**Offset:** 0x0100 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | NBTRANS[1:0] | | EPTYPE[1:0] | | | AUTOSW | EPDIR |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | EPSIZE[2:0] | | | EPBK[1:0] | | ALLOC | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bits 14:13 – NBTRANS[1:0]** Number of transactions per microframe for isochronous endpoint
This field should be written with the number of transactions per microframe to perform high-bandwidth isochronous transfer.
It can be written only for endpoints that have this capability (see USBHS_FEATURES.ENHBISOx bit). Otherwise, this field is 0.
This field is irrelevant for non-isochronous endpoints.

| Value | Name | Description |
|---|---|---|
| 0 | 0_TRANS | Reserved to endpoint that does not have the high-bandwidth isochronous capability. |
| 1 | 1_TRANS | Default value: one transaction per microframe. |
| 2 | 2_TRANS | Two transactions per microframe. This endpoint should be configured as double-bank. |
| 3 | 3_TRANS | Three transactions per microframe. This endpoint should be configured as triple-bank. |

**Bits 12:11 – EPTYPE[1:0]** Endpoint Type
This field should be written to select the endpoint type:
This field is cleared upon receiving a USB reset.

| Value | Name | Description |
|---|---|---|
| 0 | CTRL | Control |
| 1 | ISO | Isochronous |
| 2 | BLK | Bulk |
| 3 | INTRPT | Interrupt |

**Bit 9 – AUTOSW** Automatic Switch
This bit is cleared upon receiving a USB reset.

| Value | Description |
|---|---|
| 0 | The automatic bank switching is disabled. |
| 1 | The automatic bank switching is enabled. |

**Bit 8 – EPDIR** Endpoint Direction
This bit is cleared upon receiving a USB reset.

0 (OUT): The endpoint direction is OUT.
1 (IN): The endpoint direction is IN (nor for control endpoints).

**Bits 6:4 – EPSIZE[2:0]** Endpoint Size
This field should be written to select the size of each endpoint bank:
This field is cleared upon receiving a USB reset (except for endpoint 0).

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8 bytes |
| 1 | 16_BYTE | 16 bytes |
| 2 | 32_BYTE | 32 bytes |
| 3 | 64_BYTE | 64 bytes |
| 4 | 128_BYTE | 128 bytes |
| 5 | 256_BYTE | 256 bytes |
| 6 | 512_BYTE | 512 bytes |
| 7 | 1024_BYTE | 1024 bytes |

**Bits 3:2 – EPBK[1:0]** Endpoint Banks
This field should be written to select the number of banks for the endpoint:
For control endpoints, a single-bank endpoint (0b00) should be selected.
This field is cleared upon receiving a USB reset (except for endpoint 0).

| Value | Name | Description |
|---|---|---|
| 0 | 1_BANK | Single-bank endpoint |
| 1 | 2_BANK | Double-bank endpoint |
| 2 | 3_BANK | Triple-bank endpoint |
| 3 | Reserved | |

**Bit 1 – ALLOC** Endpoint Memory Allocate
This bit is cleared upon receiving a USB reset (except for endpoint 0).

| Value | Description |
|---|---|
| 0 | Frees the endpoint memory. |
| 1 | Allocates the endpoint memory. The user should check the USBHS_DEVEPTISRx.CFGOK bit to know whether the allocation of this endpoint is correct. |

### 39.6.15 Device Endpoint Interrupt Status Register (Control, Bulk, Interrupt Endpoints)

**Name:** USBHS_DEVEPTISRx
**Offset:** 0x0130 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x0, 0x2, or 0x3 in the "Device Endpoint x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | BYCT[10:4] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BYCT[3:0] | | | | | CFGOK | CTRLDIR | RWALL |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKET | STALLEDI | OVERFI | NAKINI | NAKOUTI | RXSTPI | RXOUTI | TXINI |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:20 – BYCT[10:0]** Byte Count
This field is set with the byte count of the FIFO.
For IN endpoints, the field is incremented after each byte written by the software into the endpoint and decremented after each byte sent to the host.
For OUT endpoints, the field is incremented after each byte received from the host and decremented after each byte read by the software from the endpoint.
This field may be updated one clock cycle after the RWALL bit changes, so the user should not poll this field as an interrupt bit.

**Bit 18 – CFGOK** Configuration OK Status
This bit is updated when USBHS_DEVEPTCFGx.ALLOC = 1.
This bit is set if the endpoint x number of banks (USBHS_DEVEPTCFGx.EPBK) and size (USBHS_DEVEPTCFGx.EPSIZE) are correct compared to the maximal allowed number of banks and size for this endpoint and to the maximal FIFO size (i.e., the DPRAM size).
If this bit is cleared, the user should rewrite correct values to the USBHS_DEVEPTCFGx.EPBK and USBHS_DEVEPTCFGx.EPSIZE fields.

**Bit 17 – CTRLDIR** Control Direction

| Value | Description |
|---|---|
| 0 | Cleared after a SETUP packet to indicate that the following packet is an OUT packet. |
| 1 | Set after a SETUP packet to indicate that the following packet is an IN packet. |

**Bit 16 – RWALL** Read/Write Allowed
This bit is set for IN endpoints when the current bank is not full, i.e., the user can write further data into the FIFO.
This bit is set for OUT endpoints when the current bank is not empty, i.e., the user can read further data from the FIFO.
This bit is never set if USBHS_DEVEPTIMRx.STALLRQ = 1 or in case of error.

This bit is cleared otherwise.
This bit should not be used for control endpoints.

**Bits 15:14 – CURRBK[1:0]** Current Bank
This bit is set for non-control endpoints, to indicate the current bank:
This field may be updated one clock cycle after the RWALL bit changes, so the user should not poll this field as an interrupt bit.

| Value | Name | Description |
|---|---|---|
| 0 | BANK0 | Current bank is bank0 |
| 1 | BANK1 | Current bank is bank1 |
| 2 | BANK2 | Current bank is bank2 |
| 3 | Reserved | |

**Bits 13:12 – NBUSYBK[1:0]** Number of Busy Banks
This field is set to indicate the number of busy banks:
For IN endpoints, it indicates the number of banks filled by the user and ready for IN transfer. When all banks are free, this triggers a PEP_x interrupt if NBUSYBKE = 1.
For OUT endpoints, it indicates the number of banks filled by OUT transactions from the host. When all banks are busy, this triggers a PEP_x interrupt if NBUSYBKE = 1.
When the USBHS_DEVEPTIMRx.FIFOCON bit is cleared (by writing a one to the USBHS_DEVEPTIMRx.FIFOCONC bit) to validate a new bank, this field is updated two or three clock cycles later to calculate the address of the next bank.
A PEP_x interrupt is triggered if:

| Value | Name | Description |
|---|---|---|
| 0 | 0_BUSY | 0 busy bank (all banks free) |
| 1 | 1_BUSY | 1 busy bank |
| 2 | 2_BUSY | 2 busy banks |
| 3 | 3_BUSY | 3 busy banks<br>• for IN endpoint, USBHS_DEVEPTIMRx.NBUSYBKE = 1 and all the banks are free;<br>• for OUT endpoint, USBHS_DEVEPTIMRx.NBUSYBKE = 1 and all the banks are busy. |

**Bits 9:8 – DTSEQ[1:0]** Data Toggle Sequence
This field is set to indicate the PID of the current bank:
For IN transfers, it indicates the data toggle sequence that should be used for the next packet to be sent. This is not relative to the current bank.
For OUT transfers, this value indicates the last data toggle sequence received on the current bank.
By default, DTSEQ is 0b01, as if the last data toggle sequence was Data1, so the next sent or expected data toggle sequence should be Data0.

| Value | Name | Description |
|---|---|---|
| 0 | DATA0 | Data0 toggle sequence |
| 1 | DATA1 | Data1 toggle sequence |
| 2 | DATA2 | Reserved for high-bandwidth isochronous endpoint |
| 3 | MDATA | Reserved for high-bandwidth isochronous endpoint |

**Bit 7 – SHORTPACKET** Short Packet Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when SHORTPACKETC = 1. This acknowledges the interrupt. |
| 1 | Set for non-control OUT endpoints, when a short packet has been received. This triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.SHORTPACKETE = 1. |

**Bit 6 – STALLEDI** STALLed Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when STALLEDIC = 1. This acknowledges the interrupt. |
| 1 | Set to signal that a STALL handshake has been sent. To do that, the software has to set the STALLRQ bit (by writing a one to the STALLRQS bit). This triggers a PEP_x interrupt if STALLEDE = 1. |

**Bit 5 – OVERFI** Overflow Interrupt
For all endpoint types, an overflow can occur during the OUT stage if the host attempts to write into a bank that is too small for the packet. The packet is acknowledged and the USBHS_DEVEPTISRx.RXOUTI bit is set as if no overflow had occurred. The bank is filled with all the first bytes of the packet that fit in.

| Value | Description |
|---|---|
| 0 | Cleared when the OVERFIC bit is written to one. This acknowledges the interrupt. |
| 1 | Set when an overflow error occurs. This triggers a PEP_x interrupt if OVERFE = 1. |

**Bit 4 – NAKINI** NAKed IN Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when NAKINIC = 1. This acknowledges the interrupt. |
| 1 | Set when a NAK handshake has been sent in response to an IN request from the host. This triggers a PEP_x interrupt if NAKINE = 1. |

**Bit 3 – NAKOUTI** NAKed OUT Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when NAKOUTIC = 1. This acknowledges the interrupt. |
| 1 | Set when a NAK handshake has been sent in response to an OUT request from the host. This triggers a PEP_x interrupt if NAKOUTE = 1. |

**Bit 2 – RXSTPI** Received SETUP Interrupt
This bit is set, for control endpoints, to signal that the current bank contains a new valid SETUP packet. This triggers a PEP_x interrupt if RXSTPE = 1.
It is cleared by writing a one to the RXSTPIC bit. This acknowledges the interrupt and frees the bank.
This bit is inactive (cleared) for bulk and interrupt IN/OUT endpoints.

**Bit 1 – RXOUTI** Received OUT Data Interrupt
For control endpoints:
0: Cleared by writing a one to the RXOUTIC bit. This acknowledges the interrupt and frees the bank.
1: Set when the current bank contains a bulk OUT packet (data or status stage). This triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.RXOUTE = 1.
For bulk and interrupt OUT endpoints:
0: Cleared by writing a one to the RXOUTIC bit. This acknowledges the interrupt, which has no effect on the endpoint FIFO. USBHS_DEVEPTISRx.RXOUTI shall always be cleared before clearing USBHS_DEVEPTIMRx.FIFOCON.
1: Set at the same time as USBHS_DEVEPTIMRx.FIFOCON when the current bank is full. This triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.RXOUTE = 1.
The user reads from the FIFO and clears the USBHS_DEVEPTIMRx.FIFOCON bit to free the bank. If the OUT endpoint is composed of multiple banks, this also switches to the next bank. The USBHS_DEVEPTISRx.RXOUTI and USBHS_DEVEPTIMRx.FIFOCON bits are set/cleared in accordance with the status of the next bank.
This bit is inactive (cleared) for bulk and interrupt IN endpoints.

**Bit 0 – TXINI** Transmitted IN Data Interrupt
For control endpoints:
0: Cleared when TXINIC = 1. This acknowledges the interrupt and sends the packet.
1: Set when the current bank is ready to accept a new IN packet. This triggers a PEP_x interrupt if TXINE = 1.
For bulk and interrupt IN endpoints:
0: Cleared when TXINIC = 1. This acknowledges the interrupt, which has no effect on the endpoint FIFO. USBHS_DEVEPTISRx.TXINI shall always be cleared before clearing USBHS_DEVEPTIMRx.FIFOCON.
1: Set at the same time as USBHS_DEVEPTIMRx.FIFOCON when the current bank is free. This triggers a PEP_x interrupt if TXINE = 1.
The user writes into the FIFO and clears the USBHS_DEVEPTIMRx.FIFOCON bit to allow the USBHS to send the data. If the IN endpoint is composed of multiple banks, this also switches to the next bank. The USBHS_DEVEPTISRx.TXINI and USBHS_DEVEPTIMRx.FIFOCON bits are set/cleared in accordance with the status of the next bank.
This bit is inactive (cleared) for bulk and interrupt OUT endpoints.

© 2021 Microchip Technology Inc. and its subsidiaries

### 39.6.16 Device Endpoint Interrupt Status Register (Isochronous Endpoints)

**Name:** USBHS_DEVEPTISRx (ISOENPT)
**Offset:** 0x0130 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x1 in the "Device Endpoint x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | BYCT[10:4] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BYCT[3:0] | | | | | CFGOK | | RWALL |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CURRBK[1:0] | | NBUSYBK[1:0] | | | ERRORTRANS | DTSEQ[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKET | CRCERRI | OVERFI | HBISOFLUSHI | HBISOINERRI | UNDERFI | RXOUTI | TXINI |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:20 – BYCT[10:0]** Byte Count
This field is set with the byte count of the FIFO.
For IN endpoints, the field is incremented after each byte written by the software into the endpoint and decremented after each byte sent to the host.
For OUT endpoints, the field is incremented after each byte received from the host and decremented after each byte read by the software from the endpoint.
This field may be updated one clock cycle after the RWALL bit changes, so the user should not poll this field as an interrupt bit.

**Bit 18 – CFGOK** Configuration OK Status
This bit is updated when USBHS_DEVEPTCFGx.ALLOC = 1.
This bit is set if the endpoint x number of banks (USBHS_DEVEPTCFGx.EPBK) and size (USBHS_DEVEPTCFGx.EPSIZE) are correct compared to the maximal allowed number of banks and size for this endpoint and to the maximal FIFO size (i.e., the DPRAM size).
If this bit is cleared, the user should rewrite correct values to the USBHS_DEVEPTCFGx.EPBK and USBHS_DEVEPTCFGx.EPSIZE fields.

**Bit 16 – RWALL** Read/Write Allowed
This bit is set for IN endpoints when the current bank is not full, i.e., the user can write further data into the FIFO.
This bit is set for OUT endpoints when the current bank is not empty, i.e., the user can read further data from the FIFO.
This bit is never set in case of error.
This bit is cleared otherwise.

**Bits 15:14 – CURRBK[1:0]** Current Bank
This field is used to indicate the current bank. It may be updated one clock cycle after the RWALL bit changes, so the user should not poll this field as an interrupt bit.

| Value | Name | Description |
|---|---|---|
| 0 | BANK0 | Current bank is bank0 |
| 1 | BANK1 | Current bank is bank1 |
| 2 | BANK2 | Current bank is bank2 |
| 3 | Reserved | |

**Bits 13:12 – NBUSYBK[1:0]** Number of Busy Banks
This field is set to indicate the number of busy banks:
For IN endpoints, it indicates the number of banks filled by the user and ready for IN transfer. When all banks are free, this triggers a PEP_x interrupt if NBUSYBKE = 1.
For OUT endpoints, it indicates the number of banks filled by OUT transactions from the host. When all banks are busy, this triggers a PEP_x interrupt if NBUSYBKE = 1.
When the USBHS_DEVEPTIMRx.FIFOCON bit is cleared (by writing a one to the USBHS_DEVEPTIMRx.FIFOCONC bit) to validate a new bank, this field is updated two or three clock cycles later to calculate the address of the next bank.
A PEP_x interrupt is triggered if:

| Value | Name | Description |
|---|---|---|
| 0 | 0_BUSY | 0 busy bank (all banks free) |
| 1 | 1_BUSY | 1 busy bank |
| 2 | 2_BUSY | 2 busy banks |
| 3 | 3_BUSY | 3 busy banks<br>• For IN endpoint, USBHS_DEVEPTIMRx.NBUSYBKE = 1 and all the banks are free.<br>• For OUT endpoint, USBHS_DEVEPTIMRx.NBUSYBKE = 1 and all the banks are busy. |

**Bit 10 – ERRORTRANS** High-bandwidth Isochronous OUT Endpoint Transaction Error Interrupt
This bit is set when a transaction error occurs during the current microframe (the data toggle sequencing is not compliant with the USB 2.0 standard). This triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.ERRORTRANSE = 1.
This bit is set as long as the current bank (CURRBK) belongs to the bad n-transactions (n = 1, 2 or 3) transferred during the microframe. It is cleared by software by clearing (at least once) the USBHS_DEVEPTIMRx.FIFOCON bit to switch to the bank that belongs to the next n-transactions (next microframe).

**Bits 9:8 – DTSEQ[1:0]** Data Toggle Sequence
This field is set to indicate the PID of the current bank:
For IN transfers, it indicates the data toggle sequence that should be used for the next packet to be sent. This is not relative to the current bank.
For OUT transfers, this value indicates the last data toggle sequence received on the current bank.
By default, DTSEQ is 0b01, as if the last data toggle sequence was Data1, so the next sent or expected data toggle sequence should be Data0.
For high-bandwidth isochronous endpoint, a PEP_x interrupt is triggered if:

| Value | Name | Description |
|---|---|---|
| 0 | DATA0 | Data0 toggle sequence |
| 1 | DATA1 | Data1 toggle sequence |
| 2 | DATA2 | Data2 toggle sequence (for high-bandwidth isochronous endpoint) |
| 3 | MDATA | MData toggle sequence (for high-bandwidth isochronous endpoint)<br>• USBHS_DEVEPTIMRx.MDATAE = 1 and a MData packet has been received (DTSEQ = MData and USBHS_DEVEPTISRx.RXOUTI = 1).<br>• USBHS_DEVEPTISRx.DATAXE = 1 and a Data0/1/2 packet has been received (DTSEQ = Data0/1/2 and USBHS_DEVEPTISRx.RXOUTI = 1). |

**Bit 7 – SHORTPACKET** Short Packet Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when SHORTPACKETC = 1. This acknowledges the interrupt. |
| 1 | Set for non-control OUT endpoints, when a short packet has been received. This triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.SHORTPACKETE = 1. |

**Bit 6 – CRCERRI** CRC Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when CRCERRIC = 1. This acknowledges the interrupt. |
| 1 | Set to signal that a CRC error has been detected in an isochronous OUT endpoint. The OUT packet is stored in the bank as if no CRC error had occurred. This triggers a PEP_x interrupt if CRCERRE = 1. |

**Bit 5 – OVERFI** Overflow Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when OVERFIC = 1. This acknowledges the interrupt. |
| 1 | Set when an overflow error occurs. This triggers a PEP_x interrupt if OVERFE = 1. For all endpoint types, an overflow can occur during OUT stage if the host attempts to write into a bank that is too small for the packet. The packet is acknowledged and the USBHS_DEVEPTISRx.RXOUTI bit is set as if no overflow had occurred. The bank is filled with all the first bytes of the packet that fit in. |

**Bit 4 – HBISOFLUSHI** High Bandwidth Isochronous IN Flush Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the HBISOFLUSHIC bit is written to one. This acknowledges the interrupt. |
| 1 | Set for High-bandwidth isochronous IN endpoint (with NBTRANS = 2 or 3) at the end of the microframe, if less than N transactions have been completed by the USBHS without underflow error. This may occur in case of a missing IN token. In this case, the banks are flushed out to ensure the data synchronization between the host and the device. This triggers a PEP_x interrupt if HBISOFLUSHE = 1. |

**Bit 3 – HBISOINERRI** High Bandwidth Isochronous IN Underflow Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the HBISOINERRIC bit is written to one. This acknowledges the interrupt. |
| 1 | Set for High-bandwidth isochronous IN endpoint (with NBTRANS = 2 or 3) at the end of the microframe, if less than N banks were written by the CPU within this microframe. This triggers a PEP_x interrupt if HBISOINERRE = 1. |

**Bit 2 – UNDERFI** Underflow Interrupt
This bit is set, for isochronous IN/OUT endpoints, when an underflow error occurs. This triggers a PEP_x interrupt if UNDERFE = 1.
An underflow can occur during IN stage if the host attempts to read from an empty bank. A zero-length packet is then automatically sent by the USBHS.
An underflow can also occur during OUT stage if the host sends a packet while the bank is already full. Typically, the CPU is not fast enough. The packet is lost.
It is cleared by writing a one to the UNDERFIC bit. This acknowledges the interrupt.

**Bit 1 – RXOUTI** Received OUT Data Interrupt
For control endpoints:
0: Cleared by writing a one to the RXOUTIC bit. This acknowledges the interrupt and frees the bank.
1: Set when the current bank contains a bulk OUT packet (data or status stage). This triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.RXOUTE = 1.
For OUT endpoints:
0: Cleared by writing a one to the RXOUTIC bit. This acknowledges the interrupt, which has no effect on the endpoint FIFO. USBHS_DEVEPTISRx.RXOUTI shall always be cleared before clearing USBHS_DEVEPTIMRx.FIFOCON.
1: Set at the same time as USBHS_DEVEPTIMRx.FIFOCON when the current bank is full. This triggers a PEP_x interrupt if USBHS_DEVEPTIMRx.RXOUTE = 1.
The user reads from the FIFO and clears the USBHS_DEVEPTIMRx.FIFOCON bit to free the bank. If the OUT endpoint is composed of multiple banks, this also switches to the next bank. The USBHS_DEVEPTISRx.RXOUTI and USBHS_DEVEPTIMRx.FIFOCON bits are set/cleared in accordance with the status of the next bank.
This bit is inactive (cleared) for IN endpoints.

**Bit 0 – TXINI** Transmitted IN Data Interrupt
For control endpoints:
0: Cleared when TXINIC = 1. This acknowledges the interrupt and sends the packet.
1: Set when the current bank is ready to accept a new IN packet. This triggers a PEP_x interrupt if TXINE = 1.

For IN endpoints:
0: Cleared when TXINIC = 1. This acknowledges the interrupt, which has no effect on the endpoint FIFO. USBHS_DEVEPTISRx.TXINI shall always be cleared before clearing USBHS_DEVEPTIMRx.FIFOCON.
1: Set at the same time as USBHS_DEVEPTIMRx.FIFOCON when the current bank is free. This triggers a PEP_x interrupt if TXINE = 1.
The user writes into the FIFO and clears the USBHS_DEVEPTIMRx.FIFOCON bit to allow the USBHS to send the data. If the IN endpoint is composed of multiple banks, this also switches to the next bank. The USBHS_DEVEPTISRx.TXINI and USBHS_DEVEPTIMRx.FIFOCON bits are set/cleared in accordance with the status of the next bank.
This bit is inactive (cleared) for OUT endpoints.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.17 Device Endpoint Interrupt Clear Register (Control, Bulk, Interrupt Endpoints)

**Name:** USBHS_DEVEPTICRx
**Offset:** 0x0160 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x0, 0x2, or 0x3 in the "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Status Register (Control, Bulk, Interrupt Endpoints)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_DEVEPTISRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETC | STALLEDIC | OVERFIC | NAKINIC | NAKOUTIC | RXSTPIC | RXOUTIC | TXINIC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – SHORTPACKETC** Short Packet Interrupt Clear

**Bit 6 – STALLEDIC** STALLed Interrupt Clear

**Bit 5 – OVERFIC** Overflow Interrupt Clear

**Bit 4 – NAKINIC** NAKed IN Interrupt Clear

**Bit 3 – NAKOUTIC** NAKed OUT Interrupt Clear

**Bit 2 – RXSTPIC** Received SETUP Interrupt Clear

**Bit 1 – RXOUTIC** Received OUT Data Interrupt Clear

**Bit 0 – TXINIC** Transmitted IN Data Interrupt Clear

### 39.6.18 Device Endpoint Interrupt Clear Register (Isochronous Endpoints)

**Name:** USBHS_DEVEPTICRx (ISOENPT)
**Offset:** 0x0160 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x1 in "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Status Register (Isochronous Endpoints)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_DEVEPTISRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETC | CRCERRIC | OVERFIC | HBISOFLUSHIC | HBISOINERRIC | UNDERFIC | RXOUTIC | TXINIC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – SHORTPACKETC** Short Packet Interrupt Clear

**Bit 6 – CRCERRIC** CRC Error Interrupt Clear

**Bit 5 – OVERFIC** Overflow Interrupt Clear

**Bit 4 – HBISOFLUSHIC** High Bandwidth Isochronous IN Flush Interrupt Clear

**Bit 3 – HBISOINERRIC** High Bandwidth Isochronous IN Underflow Error Interrupt Clear

**Bit 2 – UNDERFIC** Underflow Interrupt Clear

**Bit 1 – RXOUTIC** Received OUT Data Interrupt Clear

**Bit 0 – TXINIC** Transmitted IN Data Interrupt Clear

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.19 Device Endpoint Interrupt Set Register (Control, Bulk, Interrupt Endpoints)

**Name:** USBHS_DEVEPTIFRx
**Offset:** 0x0190 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x0, 0x2, or 0x3 in "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Status Register (Control, Bulk, Interrupt Endpoints)".This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_DEVEPTISRx, which may be useful for test or debug purposes.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETS | STALLEDIS | OVERFIS | NAKINIS | NAKOUTIS | RXSTPIS | RXOUTIS | TXINIS |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 12 – NBUSYBKS** Number of Busy Banks Interrupt Set

**Bit 7 – SHORTPACKETS** Short Packet Interrupt Set

**Bit 6 – STALLEDIS** STALLed Interrupt Set

**Bit 5 – OVERFIS** Overflow Interrupt Set

**Bit 4 – NAKINIS** NAKed IN Interrupt Set

**Bit 3 – NAKOUTIS** NAKed OUT Interrupt Set

**Bit 2 – RXSTPIS** Received SETUP Interrupt Set

**Bit 1 – RXOUTIS** Received OUT Data Interrupt Set

**Bit 0 – TXINIS** Transmitted IN Data Interrupt Set

### 39.6.20 Device Endpoint Interrupt Set Register (Isochronous Endpoints)

**Name:** USBHS_DEVEPTIFRx (ISOENPT)
**Offset:** 0x0190 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x1 in "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Status Register (Isochronous Endpoints)".

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_DEVEPTISRx, which may be useful for test or debug purposes.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | NBUSYBKS | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETS | CRCERRIS | OVERFIS | HBISOFLUSHIS | HBISOINERRIS | UNDERFIS | RXOUTIS | TXINIS |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 12 – NBUSYBKS** Number of Busy Banks Interrupt Set

**Bit 7 – SHORTPACKETS** Short Packet Interrupt Set

**Bit 6 – CRCERRIS** CRC Error Interrupt Set

**Bit 5 – OVERFIS** Overflow Interrupt Set

**Bit 4 – HBISOFLUSHIS** High Bandwidth Isochronous IN Flush Interrupt Set

**Bit 3 – HBISOINERRIS** High Bandwidth Isochronous IN Underflow Error Interrupt Set

**Bit 2 – UNDERFIS** Underflow Interrupt Set

**Bit 1 – RXOUTIS** Received OUT Data Interrupt Set

**Bit 0 – TXINIS** Transmitted IN Data Interrupt Set

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.21 Device Endpoint Interrupt Mask Register (Control, Bulk, Interrupt Endpoints)

**Name:** USBHS_DEVEPTIMRx
**Offset:** 0x01C0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x0, 0x2, or 0x3 in "Device Endpoint x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | STALLRQ | RSTDT | NYETDIS | EPDISHDMA |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCON | KILLBK | NBUSYBKE | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETE | STALLEDE | OVERFE | NAKINE | NAKOUTE | RXSTPE | RXOUTE | TXINE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 19 – STALLRQ** STALL Request

| Value | Description |
|---|---|
| 0 | Cleared when a new SETUP packet is received or when USBHS_DEVEPTIDRx.STALLRQC = 0. |
| 1 | Set when USBHS_DEVEPTIERx.STALLRQS = 1. This requests to send a STALL handshake to the host. |

**Bit 18 – RSTDT** Reset Data Toggle
This bit is set when USBHS_DEVEPTIERx.RSTDTS = 1. This clears the data toggle sequence, i.e., sets to Data0 the data toggle sequence of the next sent (IN endpoints) or received (OUT endpoints) packet.
This bit is cleared instantaneously.
The user does not have to wait for this bit to be cleared.

**Bit 17 – NYETDIS** NYET Token Disable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.NYETDISC = 1. This enables the USBHS to handle the high-speed handshake following the USB 2.0 standard. |
| 1 | Set when USBHS_DEVEPTIERx.NYETDISS = 1. This sends a ACK handshake instead of a NYET handshake in High-speed mode. |

**Bit 16 – EPDISHDMA** Endpoint Interrupts Disable HDMA Request
This bit is set when USBHS_DEVEPTIERx.EPDISHDMAS = 1. This pauses the on-going DMA channel x transfer on any Endpoint x interrupt (PEP_x), whatever the state of the Endpoint x Interrupt Enable bit (PEP_x).
The user then has to acknowledge or to disable the interrupt source (e.g. USBHS_DEVEPTISRx.RXOUTI) or to clear the EPDISHDMA bit (by writing a one to the USBHS_DEVEPTIDRx.EPDISHDMAC bit) in order to complete the DMA transfer.

© 2021 Microchip Technology Inc. and its subsidiaries

In Ping-pong mode, if the interrupt is associated to a new system-bank packet (e.g. Bank1) and the current DMA transfer is running on the previous packet (Bank0), then the previous-packet DMA transfer completes normally, but the new-packet DMA transfer does not start (not requested).
If the interrupt is not associated to a new system-bank packet (USBHS_DEVEPTISRx.NAKINI, NAKOUTI, etc.), then the request cancellation may occur at any time and may immediately pause the current DMA transfer.
This may be used for example to identify erroneous packets, to prevent them from being transferred into a buffer, to complete a DMA transfer by software after reception of a short packet, etc.

**Bit 14 – FIFOCON** FIFO Control
For control endpoints:
The FIFOCON and RWALL bits are irrelevant. Therefore, the software never uses them on these endpoints. When read, their value is always 0.
For IN endpoints:
0: Cleared (by writing a one to the USBHS_DEVEPTIDRx.FIFOCONC bit) to send the FIFO data and to switch to the next bank.
1: Set when the current bank is free, at the same time as USBHS_DEVEPTISRx.TXINI.
For OUT endpoints:
0: Cleared (by writing a one to the USBHS_DEVEPTIDRx.FIFOCONC bit) to free the current bank and to switch to the next bank.
1: Set when the current bank is full, at the same time as USBHS_DEVEPTISRx.RXOUTI.

**Bit 13 – KILLBK** Kill IN Bank
This bit is set when the USBHS_DEVEPTIERx.KILLBKS bit is written to one. This kills the last written bank.
This bit is cleared when the bank is killed.

**⚠ CAUTION** The bank is really cleared when the “kill packet” procedure is accepted by the USBHS core. This bit is automatically cleared after the end of the procedure.

The bank is really killed: USBHS_DEVEPTISRx.NBUSYBK is decremented.
The bank is not cleared but sent (IN transfer): USBHS_DEVEPTISRx.NBUSYBK is decremented.
The bank is not cleared because it was empty.
The user should wait for this bit to be cleared before trying to kill another packet.
This kill request is refused if at the same time an IN token is coming and the last bank is the current one being sent on the USB line. If at least two banks are ready to be sent, there is no problem to kill a packet even if an IN token is coming. Indeed, in this case, the current bank is sent (IN transfer) while the last bank is killed.

**Bit 12 – NBUSYBKE** Number of Busy Banks Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.NBUSYBKEC = 0. This disables the Number of Busy Banks interrupt (USBHS_DEVEPTISRx.NBUSYBK). |
| 1 | Set when the USBHS_DEVEPTIERx.NBUSYBKES = 1. This enables the Number of Busy Banks interrupt (USBHS_DEVEPTISRx.NBUSYBK). |

**Bit 7 – SHORTPACKETE** Short Packet Interrupt
If this bit is set for non-control IN endpoints, a short packet transmission is guaranteed upon ending a DMA transfer, thus signaling an end of isochronous frame or a bulk or interrupt end of transfer, provided that the End of DMA Buffer Output Enable (END_B_EN) bit and the Automatic Switch (AUTOSW) = 1.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.SHORTPACKETEC = 1. This disables the Short Packet interrupt (USBHS_DEVEPTISRx.SHORTPACKET). |
| 1 | Set when USBHS_DEVEPTIERx.SHORTPACKETES = 1. This enables the Short Packet interrupt (USBHS_DEVEPTISRx.SHORTPACKET). |

**Bit 6 – STALLEDE** STALLed Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.STALLEDEC = 1. This disables the STALLed interrupt (USBHS_DEVEPTISRx.STALLEDI). |
| 1 | Set when USBHS_DEVEPTIERx.STALLEDES = 1. This enables the STALLed interrupt (USBHS_DEVEPTISRx.STALLEDI). |

**Bit 5 – OVERFE** Overflow Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.OVERFEC = 1. This disables the Overflow interrupt (USBHS_DEVEPTISRx.OVERFI). |
| 1 | Set when USBHS_DEVEPTIERx.OVERFES = 1. This enables the Overflow interrupt (USBHS_DEVEPTISRx.OVERFI). |

**Bit 4 – NAKINE** NAKed IN Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.NAKINEC = 1. This disables the NAKed IN interrupt (USBHS_DEVEPTISRx.NAKINI). |
| 1 | Set when USBHS_DEVEPTIERx.NAKINES = 1. This enables the NAKed IN interrupt (USBHS_DEVEPTISRx.NAKINI). |

**Bit 3 – NAKOUTE** NAKed OUT Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.NAKOUTEC = 1. This disables the NAKed OUT interrupt (USBHS_DEVEPTISRx.NAKOUTI). |
| 1 | Set when USBHS_DEVEPTIERx.NAKOUTES = 1. This enables the NAKed OUT interrupt (USBHS_DEVEPTISRx.NAKOUTI). |

**Bit 2 – RXSTPE** Received SETUP Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIERx.RXSTPEC = 1. This disables the Received SETUP interrupt (USBHS_DEVEPTISRx.RXSTPI). |
| 1 | Set when USBHS_DEVEPTIERx.RXSTPES = 1. This enables the Received SETUP interrupt (USBHS_DEVEPTISRx.RXSTPI). |

**Bit 1 – RXOUTE** Received OUT Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.RXOUTEC = 1. This disables the Received OUT Data interrupt (USBHS_DEVEPTISRx.RXOUTI). |
| 1 | Set when USBHS_DEVEPTIERx.RXOUTES = 1. This enables the Received OUT Data interrupt (USBHS_DEVEPTISRx.RXOUTI). |

**Bit 0 – TXINE** Transmitted IN Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.TXINEC = 1. This disables the Transmitted IN Data interrupt (USBHS_DEVEPTISRx.TXINI). |
| 1 | Set when USBHS_DEVEPTIERx.TXINES = 1. This enables the Transmitted IN Data interrupt (USBHS_DEVEPTISRx.TXINI). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.22 Device Endpoint Interrupt Mask Register (Isochronous Endpoints)

**Name:** USBHS_DEVEPTIMRx (ISOENPT)
**Offset:** 0x01C0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x1 in "Device Endpoint x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTDT | | EPDISHDMA |
| Access | | | | | | | | |
| Reset | | | | | | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCON | KILLBK | NBUSYBKE | | ERRORTRANSE | DATAXE | MDATAE |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETE | CRCERRE | OVERFE | HBISOFLUSHE | HBISOINERRE | UNDERFE | RXOUTE | TXINE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – RSTDT** Reset Data Toggle
This bit is set when USBHS_DEVEPTIERx.RSTDTS = 1. This clears the data toggle sequence, i.e., sets to Data0 the data toggle sequence of the next sent (IN endpoints) or received (OUT endpoints) packet.
This bit is cleared instantaneously.
The user does not have to wait for this bit to be cleared.

**Bit 16 – EPDISHDMA** Endpoint Interrupts Disable HDMA Request
This bit is set when USBHS_DEVEPTIERx.EPDISHDMAS = 1. This pauses the on-going DMA channel x transfer on any Endpoint x interrupt (PEP_x), whatever the state of the Endpoint x Interrupt Enable bit (PEP_x).
The user then has to acknowledge or to disable the interrupt source (e.g. USBHS_DEVEPTISRx.RXOUTI) or to clear the EPDISHDMA bit (by writing a one to the USBHS_DEVEPTIDRx.EPDISHDMAC bit) in order to complete the DMA transfer.
In Ping-pong mode, if the interrupt is associated to a new system-bank packet (e.g. Bank1) and the current DMA transfer is running on the previous packet (Bank0), then the previous-packet DMA transfer completes normally, but the new-packet DMA transfer does not start (not requested).
If the interrupt is not associated to a new system-bank packet (USBHS_DEVEPTISRx.NAKINI, NAKOUTI, etc.), then the request cancellation may occur at any time and may immediately pause the current DMA transfer.
This may be used for example to identify erroneous packets, to prevent them from being transferred into a buffer, to complete a DMA transfer by software after reception of a short packet, etc.

**Bit 14 – FIFOCON** FIFO Control
For control endpoints:
The FIFOCON and RWALL bits are irrelevant. Therefore, the software never uses them on these endpoints. When read, their value is always 0.
For IN endpoints:

0: Cleared (by writing a one to the USBHS_DEVEPTIDRx.FIFOCONC bit) to send the FIFO data and to switch to the next bank.
1: Set when the current bank is free, at the same time as USBHS_DEVEPTISRx.TXINI.
For OUT endpoints:
0: Cleared (by writing a one to the USBHS_DEVEPTIDRx.FIFOCONC bit) to free the current bank and to switch to the next bank.
1: Set when the current bank is full, at the same time as USBHS_DEVEPTISRx.RXOUTI.

**Bit 13 – KILLBK** Kill IN Bank

⚠ CAUTION The bank is really cleared when the “kill packet” procedure is accepted by the USBHS core. This bit is automatically cleared after the end of the procedure.

The bank is really killed: USBHS_DEVEPTISRx.NBUSYBK is decremented.
The bank is not cleared but sent (IN transfer): USBHS_DEVEPTISRx.NBUSYBK is decremented.
The bank is not cleared because it was empty.
The user should wait for this bit to be cleared before trying to kill another packet.
This kill request is refused if at the same time an IN token is coming and the last bank is the current one being sent on the USB line. If at least two banks are ready to be sent, there is no problem to kill a packet even if an IN token is coming. Indeed, in this case, the current bank is sent (IN transfer) while the last bank is killed.

| Value | Description |
|---|---|
| 0 | Cleared when the bank is killed. |
| 1 | Set when USBHS_DEVEPTIERx.KILLBKS = 1. This kills the last written bank. |

**Bit 12 – NBUSYBKE** Number of Busy Banks Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.NBUSYBKEC = 0. This disables the Number of Busy Banks interrupt (USBHS_DEVEPTISRx.NBUSYBK). |
| 1 | Set when USBHS_DEVEPTIERx.NBUSYBKES = 1. This enables the Number of Busy Banks interrupt (USBHS_DEVEPTISRx.NBUSYBK). |

**Bit 10 – ERRORTRANSE** Transaction Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.ERRORTRANSEC = 1. This disables the transaction error interrupt (USBHS_DEVEPTISRx.ERRORTRANS). |
| 1 | Set when USBHS_DEVEPTIERx.ERRORTRANSES = 1. This enables the transaction error interrupt (USBHS_DEVEPTISRx.ERRORTRANS). |

**Bit 9 – DATAXE** DataX Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.DATAXEC = 1. This disables the DATAX interrupt. |
| 1 | Set when the USBHS_DEVEPTIERx.DATAXES = 1. This enables the DATAX interrupt (see DTSEQ bits). |

**Bit 8 – MDATAE** MData Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.MDATAEC = 1. This disables the Multiple DATA interrupt. |
| 1 | Set when the USBHS_DEVEPTIERx.MDATAES = 1. This enables the Multiple DATA interrupt (see DTSEQ bits). |

**Bit 7 – SHORTPACKETE** Short Packet Interrupt
If this bit is set for non-control IN endpoints, a short packet transmission is guaranteed upon ending a DMA transfer, thus signaling an end of isochronous frame or a bulk or interrupt end of transfer, provided that the End of DMA Buffer Output Enable (END_B_EN) bit and the Automatic Switch (AUTOSW) bit = 1.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.SHORTPACKETEC = 1. This disables the Short Packet interrupt (USBHS_DEVEPTISRx.SHORTPACKET). |
| 1 | Set when USBHS_DEVEPTIERx.SHORTPACKETES = 1. This enables the Short Packet interrupt (USBHS_DEVEPTISRx.SHORTPACKET). |

**Bit 6 – CRCERRE** CRC Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.CRCERREC = 1. This disables the CRC Error interrupt (USBHS_DEVEPTISRx.CRCERRI). |
| 1 | Set when USBHS_DEVEPTIERx.CRCERRES = 1. This enables the CRC Error interrupt (USBHS_DEVEPTISRx.CRCERRI). |

**Bit 5 – OVERFE** Overflow Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.OVERFEC = 1. This disables the Overflow interrupt (USBHS_DEVEPTISRx.OVERFI). |
| 1 | Set when USBHS_DEVEPTIERx.OVERFES = 1. This enables the Overflow interrupt (USBHS_DEVEPTISRx.OVERFI). |

**Bit 4 – HBISOFLUSHE** High Bandwidth Isochronous IN Flush Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVEPTIDRx.HBISOFLUSHEC bit disables the HBISOFLUSHI interrupt. |
| 1 | Set when USBHS_DEVEPTIERx.HBISOFLUSHES = 1. This enables the HBISOFLUSHI interrupt. |

**Bit 3 – HBISOINERRE** High Bandwidth Isochronous IN Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_DEVEPTIDRx.HBISOINERREC bit disables the HBISOINERRI interrupt. |
| 1 | Set when USBHS_DEVEPTIERx.HBISOINERRES = 1. This enables the HBISOINERRI interrupt. |

**Bit 2 – UNDERFE** Underflow Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.UNDERFEC = 1. This disables the Underflow interrupt (USBHS_DEVEPTISRx.UNDERFI). |
| 1 | Set when USBHS_DEVEPTIERx.UNDERFES = 1. This enables the Underflow interrupt (USBHS_DEVEPTISRx.UNDERFI). |

**Bit 1 – RXOUTE** Received OUT Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.RXOUTEC = 1. This disables the Received OUT Data interrupt (USBHS_DEVEPTISRx.RXOUTI). |
| 1 | Set when USBHS_DEVEPTIERx.RXOUTES = 1. This enables the Received OUT Data interrupt (USBHS_DEVEPTISRx.RXOUTI). |

**Bit 0 – TXINE** Transmitted IN Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_DEVEPTIDRx.TXINEC = 1. This disables the Transmitted IN Data interrupt (USBHS_DEVEPTISRx.TXINI). |
| 1 | Set when USBHS_DEVEPTIERx.TXINES = 1. This enables the Transmitted IN Data interrupt (USBHS_DEVEPTISRx.TXINI). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.23 Device Endpoint Interrupt Disable Register (Control, Bulk, Interrupt Endpoints)

**Name:** USBHS_DEVEPTIDRx
**Offset:** 0x0220 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x0, 0x2, or 0x3 in "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Mask Register (Control, Bulk, Interrupt Endpoints)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_DEVEPTIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | STALLRQC | | NYETDISC | EPDISHDMAC |
| Access | | | | | | | | |
| Reset | | | | | 0 | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCONC | | NBUSYBKEC | | | | |
| Access | | | | | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETEC | STALLEDEC | OVERFEC | NAKINEC | NAKOUTEC | RXSTPEC | RXOUTEC | TXINEC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 19 – STALLRQC** STALL Request Clear

**Bit 17 – NYETDISC** NYET Token Disable Clear

**Bit 16 – EPDISHDMAC** Endpoint Interrupts Disable HDMA Request Clear

**Bit 14 – FIFOCONC** FIFO Control Clear

**Bit 12 – NBUSYBKEC** Number of Busy Banks Interrupt Clear

**Bit 7 – SHORTPACKETEC** Shortpacket Interrupt Clear

**Bit 6 – STALLEDEC** STALLed Interrupt Clear

**Bit 5 – OVERFEC** Overflow Interrupt Clear

**Bit 4 – NAKINEC** NAKed IN Interrupt Clear

**Bit 3 – NAKOUTEC** NAKed OUT Interrupt Clear

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 2 – RXSTPEC** Received SETUP Interrupt Clear

**Bit 1 – RXOUTEC** Received OUT Data Interrupt Clear

**Bit 0 – TXINEC** Transmitted IN Interrupt Clear

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.24 Device Endpoint Interrupt Disable Register (Isochronous Endpoints)

**Name:** USBHS_DEVEPTIDRx (ISOENPT)
**Offset:** 0x0220 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x1 in "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Mask Register (Isochronous Endpoints)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_DEVEPTIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | EPDISHDMAC |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCONC | | NBUSYBKEC | | ERRORTRANSEC | DATAXEC | MDATEC |
| Access | | | | | | | | |
| Reset | | 0 | | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETEC | CRCERREC | OVERFEC | HBISOFLUSHEC | HBISOINERREC | UNDERFEC | RXOUTEC | TXINEC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 16 – EPDISHDMAC** Endpoint Interrupts Disable HDMA Request Clear

**Bit 14 – FIFOCONC** FIFO Control Clear

**Bit 12 – NBUSYBKEC** Number of Busy Banks Interrupt Clear

**Bit 10 – ERRORTRANSEC** Transaction Error Interrupt Clear

**Bit 9 – DATAXEC** DataX Interrupt Clear

**Bit 8 – MDATEC** MData Interrupt Clear

**Bit 7 – SHORTPACKETEC** Shortpacket Interrupt Clear

**Bit 6 – CRCERREC** CRC Error Interrupt Clear

**Bit 5 – OVERFEC** Overflow Interrupt Clear

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 4 – HBISOFLUSHEC** High Bandwidth Isochronous IN Flush Interrupt Clear

**Bit 3 – HBISOINERREC** High Bandwidth Isochronous IN Error Interrupt Clear

**Bit 2 – UNDERFEC** Underflow Interrupt Clear

**Bit 1 – RXOUTEC** Received OUT Data Interrupt Clear

**Bit 0 – TXINEC** Transmitted IN Interrupt Clear

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.25 Device Endpoint Interrupt Enable Register (Control, Bulk, Interrupt Endpoints)

**Name:** USBHS_DEVEPTIERx
**Offset:** 0x01F0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x0, 0x2, or 0x3 in "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Mask Register (Control, Bulk, Interrupt Endpoints)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_DEVEPTIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | STALLRQS | RSTDTS | NYETDISS | EPDISHDMAS |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCONS | KILLBKS | NBUSYBKES | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETES | STALLEDES | OVERFES | NAKINES | NAKOUTES | RXSTPES | RXOUTES | TXINES |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 19 – STALLRQS** STALL Request Enable

**Bit 18 – RSTDTS** Reset Data Toggle Enable

**Bit 17 – NYETDISS** NYET Token Disable Enable

**Bit 16 – EPDISHDMAS** Endpoint Interrupts Disable HDMA Request Enable

**Bit 14 – FIFOCONS** FIFO Control

**Bit 13 – KILLBKS** Kill IN Bank

**Bit 12 – NBUSYBKES** Number of Busy Banks Interrupt Enable

**Bit 7 – SHORTPACKETES** Short Packet Interrupt Enable

**Bit 6 – STALLEDES** STALLed Interrupt Enable

**Bit 5 – OVERFES** Overflow Interrupt Enable

**Bit 4 – NAKINES** NAKed IN Interrupt Enable

**Bit 3 – NAKOUTES** NAKed OUT Interrupt Enable

**Bit 2 – RXSTPES** Received SETUP Interrupt Enable

**Bit 1 – RXOUTES** Received OUT Data Interrupt Enable

**Bit 0 – TXINES** Transmitted IN Data Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.26 Device Endpoint Interrupt Enable Register (Isochronous Endpoints)

**Name:** USBHS_DEVEPTIERx (ISOENPT)
**Offset:** 0x01F0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if EPTYPE = 0x1 in "Device Endpoint x Configuration Register".

For additional information, see "Device Endpoint x Mask Register (Isochronous Endpoints)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_DEVEPTIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTDTS | | EPDISHDMAS |
| Access | | | | | | | | |
| Reset | | | | | | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCONS | KILLBKS | NBUSYBKES | | ERRORTRANSES | DATAXES | MDATAES |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETES | CRCERRES | OVERFES | HBISOFLUSHES | HBISOINERRES | UNDERFES | RXOUTES | TXINES |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – RSTDTS** Reset Data Toggle Enable

**Bit 16 – EPDISHDMAS** Endpoint Interrupts Disable HDMA Request Enable

**Bit 14 – FIFOCONS** FIFO Control

**Bit 13 – KILLBKS** Kill IN Bank

**Bit 12 – NBUSYBKES** Number of Busy Banks Interrupt Enable

**Bit 10 – ERRORTRANSES** Transaction Error Interrupt Enable

**Bit 9 – DATAXES** DataX Interrupt Enable

**Bit 8 – MDATAES** MData Interrupt Enable

**Bit 7 – SHORTPACKETES** Short Packet Interrupt Enable

**Bit 6 – CRCERRES** CRC Error Interrupt Enable

**Bit 5 – OVERFES** Overflow Interrupt Enable

**Bit 4 – HBISOFLUSHES** High Bandwidth Isochronous IN Flush Interrupt Enable

**Bit 3 – HBISOINERRES** High Bandwidth Isochronous IN Error Interrupt Enable

**Bit 2 – UNDERFES** Underflow Interrupt Enable

**Bit 1 – RXOUTES** Received OUT Data Interrupt Enable

**Bit 0 – TXINES** Transmitted IN Data Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.27 Device DMA Channel x Next Descriptor Address Register

**Name:** USBHS_DEVDMANXTDSCx
**Offset:** 0x0300 + (x-1)*0x10 [x=1..7]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NXT_DSC_ADD[31:0]** Next Descriptor Address
This field points to the next channel descriptor to be processed. This channel descriptor must be aligned, so bits 0 to 3 of the address must be equal to zero.

### 39.6.28 Device DMA Channel x Address Register

**Name:** USBHS_DEVDMAADDRESSx
**Offset:** 0x0304 + (x-1)*0x10 [x=1..7]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BUFF_ADD[31:0]** Buffer Address
This field determines the AHB bus starting address of a DMA channel transfer.
Channel start and end addresses may be aligned on any byte boundary.
The firmware can write this field only when the USBHS_DEVDMASTATUS.CHANN_ENB bit is clear.
This field is updated at the end of the address phase of the current access to the AHB bus. It is incremented by the access byte width. The access width is 4 bytes (or less) at packet start or end, if the start or end address is not aligned on a word boundary.
The packet start address is either the channel start address or the next channel address to be accessed in the channel buffer. The packet end address is either the channel end address or the latest channel address accessed in the channel buffer.
The channel start address is written by software or loaded from the descriptor. The channel end address is either determined by the end of buffer or the USB device, or by the USB end of transfer if the USBHS_DEVDMACONTROLx.END_TR_EN bit is set.

© 2021 Microchip Technology Inc. and its subsidiaries

### 39.6.29 Device DMA Channel x Control Register

**Name:** USBHS_DEVDMACONTROLx
**Offset:** 0x0308 + (x-1)*0x10 [x=1..7]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_LENGTH[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_LENGTH[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – BUFF_LENGTH[15:0]** Buffer Byte Length (Write-only)
This field determines the number of bytes to be transferred until end of buffer. The maximum channel transfer size (32 KBytes) is reached when this field is 0 (default value). If the transfer size is unknown, this field should be set to 0, but the transfer end may occur earlier under USB device control.
When this field is written, the USBHS_DEVDMASTATUSx.BUFF_COUNT field is updated with the write value.
**Note:** 1. Bits [31:2] are only writable when issuing a channel Control Command other than “Stop Now”.

**Note:** 2. For reliability, it is recommended to wait for both the USBHS_DEVDMASTATUSx.CHAN_ACT and the USBHS_DEVDMASTATUSx.CHAN_ENB flags to be at 0, thus ensuring the channel has been stopped before issuing a command other than “Stop Now”.

**Bit 7 – BURST_LCK** Burst Lock Enable

| Value | Description |
|---|---|
| 0 | The DMA never locks bus access. |
| 1 | USB packets AHB data bursts are locked for maximum optimization of the bus bandwidth usage and maximization of fly-by AHB burst duration. |

**Bit 6 – DESC_LD_IT** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | USBHS_DEVDMASTATUSx.DESC_LDST rising does not trigger any interrupt. |
| 1 | An interrupt is generated when a descriptor has been loaded from the bus. |

**Bit 5 – END_BUFFIT** End of Buffer Interrupt Enable

| Value | Description |
|---|---|
| 0 | USBHS_DEVDMA_STATUSx.END_BF_ST rising does not trigger any interrupt. |
| 1 | An interrupt is generated when USBHS_HSTDMASTATUSx.BUFF_COUNT reaches zero. |

**Bit 4 – END_TR_IT** End of Transfer Interrupt Enable
Use when the receive size is unknown.

| Value | Description |
|---|---|
| 0 | USBHS device-initiated buffer transfer completion does not trigger any interrupt at USBHS_DEVDMASTATUSx.END_TR_ST rising. |
| 1 | An interrupt is sent after the buffer transfer is complete, if the USBHS device has ended the buffer transfer. |

**Bit 3 – END_B_EN** End of Buffer Enable Control
This is mainly for short packet IN validations initiated by the DMA reaching end of buffer, but can be used for OUT packet truncation (discarding of unwanted packet data) at the end of DMA buffer.

| Value | Description |
|---|---|
| 0 | DMA Buffer End has no impact on USB packet transfer. |
| 1 | The endpoint can validate the packet (according to the values programmed in the USBHS_DEVEPTCFGx.AUTOSW and USBHS_DEVEPTIERx.SHORTPACKETES fields) at DMA Buffer End, i.e., when USBHS_DEVDMASTATUS.BUFF_COUNT reaches 0. |

**Bit 2 – END_TR_EN** End of Transfer Enable Control (OUT transfers only)
When set, a BULK or INTERRUPT short packet or the last packet of an ISOCHRONOUS (micro) frame (DATAX) closes the current buffer and the USBHS_DEVDMASTATUSx.END_TR_ST flag is raised.
This is intended for a USBHS non-prenegotiated end of transfer (BULK or INTERRUPT) or ISOCHRONOUS microframe data buffer closure.

| Value | Description |
|---|---|
| 0 | The USB end of transfer is ignored. |
| 1 | The USBHS device can put an end to the current buffer transfer. |

**Bit 1 – LDNXT_DSC** Load Next Channel Transfer Descriptor Enable Command
If the CHANN_ENB bit is cleared, the next descriptor is immediately loaded upon transfer request.
DMA Channel Control Command Summary:

| Value LDNXT_DSC | Value CHANN_ENB | Name | Description |
|---|---|---|---|
| 0 | 0 | STOP_NOW | Stop now |
| 0 | 1 | RUN_AND_STOP | Run and stop at end of buffer |
| 1 | 0 | LOAD_NEXT_DESC | Load next descriptor now |
| 1 | 1 | RUN_AND_LINK | Run and link at end of buffer |

| Value | Description |
|---|---|
| 0 | No channel register is loaded after the end of the channel transfer. |
| 1 | The channel controller loads the next descriptor after the end of the current transfer, i.e., when the USBHS_DEVDMASTATUS.CHANN_ENB bit is reset. |

**Bit 0 – CHANN_ENB** Channel Enable Command

| Value | Description |
|---|---|
| 0 | The DMA channel is disabled at end of transfer and no transfer occurs upon request. This bit is also cleared by hardware when the channel source bus is disabled at end of buffer.<br>If the LDNXT_DSC bit has been cleared by descriptor loading, the firmware must set the corresponding CHANN_ENB bit to start the described transfer, if needed.<br>If the LDNXT_DSC bit is cleared, the channel is frozen and the channel registers may then be read and/or written reliably as soon as both USBHS_DEVDMASTATUS.CHANN_ENB and CHANN_ACT flags read as 0.<br>If a channel request is currently serviced when this bit is cleared, the DMA FIFO buffer is drained until it is empty, then the USBHS_DEVDMASTATUS.CHANN_ENB bit is cleared.<br>If the LDNXT_DSC bit is set at or after this bit clearing, then the currently loaded descriptor is skipped (no data transfer occurs) and the next descriptor is immediately loaded. |
| 1 | The USBHS_DEVDMASTATUS.CHANN_ENB bit is set, thus enabling the DMA channel data transfer. Then, any pending request starts the transfer. This may be used to start or resume any requested transfer. |

### 39.6.30 Device DMA Channel x Status Register

**Name:** USBHS_DEVDMASTATUSx
**Offset:** 0x030C + (x-1)*0x10 [x=1..7]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_COUNT[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_COUNT[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

**Bits 31:16 – BUFF_COUNT[15:0]** Buffer Byte Count
This field determines the current number of bytes still to be transferred for this buffer.
This field is decremented from the AHB source bus access byte width at the end of this bus address phase.
The access byte width is 4 by default, or less, at DMA start or end, if the start or end address is not aligned on a word boundary.
At the end of buffer, the DMA accesses the USBHS device only for the number of bytes needed to complete it.
**Note:** For OUT endpoints, if the receive buffer byte length (BUFF_LENGTH) has been defaulted to zero because the USB transfer length is unknown, the actual buffer byte length received is 0x10000-BUFF_COUNT.

**Bit 6 – DESC_LDST** Descriptor Loaded Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when a descriptor has been loaded from the system bus. |

**Bit 5 – END_BF_ST** End of Channel Buffer Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when the BUFF_COUNT count-down reaches zero. |

**Bit 4 – END_TR_ST** End of Channel Transfer Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when the last packet transfer is complete, if the USBHS device has ended the transfer. |

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 1 – CHANN_ACT** Channel Active Status
When a packet transfer is ended, this bit is automatically reset.
When a packet transfer cannot be completed due to an END_BF_ST, this flag stays set during the next channel descriptor load (if any) and potentially until completion of a USBHS packet transfer, if allowed by the new descriptor.

| Value | Description |
|---|---|
| 0 | The DMA channel is no longer trying to source the packet data. |
| 1 | The DMA channel is currently trying to source packet data, i.e., selected as the highest-priority requesting channel. |

**Bit 0 – CHANN_ENB** Channel Enable Status
When any transfer is ended either due to an elapsed byte count or to completion of a USBHS device-initiated transfer, this bit is automatically reset.
This bit is normally set or cleared by writing into the USBHS_DEVDMACONTROLx.CHANN_ENB bit field either by software or descriptor loading.
If a channel request is currently serviced when the USBHS_DEVDMACONTROLx.CHANN_ENB bit is cleared, the DMA FIFO buffer is drained until it is empty, then this status bit is cleared.

| Value | Description |
|---|---|
| 0 | If cleared, the DMA channel no longer transfers data, and may load the next descriptor if the USBHS_DEVDMACONTROLx.LDNXT_DSC bit is set. |
| 1 | If set, the DMA channel is currently enabled and transfers data upon request. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.31 Host General Control Register

**Name:** USBHS_HSTCTRL
**Offset:** 0x0400
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | SPDCONF[1:0] | | | RESUME | RESET | SOFE |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 13:12 – SPDCONF[1:0]** Mode Configuration
This field contains the host speed capability:.

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | The host starts in Full-speed mode and performs a high-speed reset to switch to High-speed mode if the downstream peripheral is high-speed capable. |
| 1 | LOW_POWER | For a better consumption, if high speed is not needed. |
| 2 | HIGH_SPEED | Forced high speed. |
| 3 | FORCED_FS | The host remains in Full-speed mode whatever the peripheral speed capability. |

**Bit 10 – RESUME** Send USB Resume
This bit is cleared when the USB Resume has been sent or when a USB reset is requested.
This bit should be written to one only when the start of frame generation is enabled (SOFE = 1).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Generates a USB Resume on the USB bus. |

**Bit 9 – RESET** Send USB Reset
This bit is cleared when the USB Reset has been sent.
It may be useful to write a zero to this bit when a device disconnection is detected (USBHS_HSTISR.DDISCI = 1) whereas a USB Reset is being sent.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Generates a USB Reset on the USB bus. |

**Bit 8 – SOFE** Start of Frame Generation Enable
This bit is set when a USB reset is requested or an upstream resume interrupt is detected (USBHS_HSTISR.TXRSMI).

| Value | Description |
|---|---|
| 0 | Disables the SOF generation and leaves the USB bus in idle state. |

| Value | Description |
|---|---|
| 1 | Generates SOF on the USB bus in Full- or High-speed mode and sends “keep alive” signals in Low-speed mode. |

### 39.6.32 Host Global Interrupt Status Register

**Name:** USBHS_HSTISR
**Offset:** 0x0404
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PEP_9 | PEP_8 |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEP_7 | PEP_6 | PEP_5 | PEP_4 | PEP_3 | PEP_2 | PEP_1 | PEP_0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | HWUPI | HSOFI | RXRSMI | RSMEDI | RSTI | DDISCI | DCONNI |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the USBHS_HSTDMASTATUSx interrupt source is cleared. |
| 1 | Set when an interrupt is triggered by the DMA channel x. This triggers a USB interrupt if the corresponding bit in USBHS_HSTIMR = 1. |

**Bits 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 – PEP_** Pipe x Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the interrupt source is served. |
| 1 | Set when an interrupt is triggered by pipe x (USBHS_HSTPIPISRx). This triggers a USB interrupt if the corresponding bit in USBHS_HSTIMR = 1. |

**Bit 6 – HWUPI** Host Wakeup Interrupt
This bit is set when the host controller is in Suspend mode (SOFE = 0) and an upstream resume from the peripheral is detected.
This bit is set when the host controller is in Suspend mode (SOFE = 0) and a peripheral disconnection is detected.
This interrupt is generated even if the clock is frozen by the USBHS_CTRL.FRZCLK bit.

**Bit 5 – HSOFI** Host Start of Frame Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTICR.HSOFIC = 1. |
| 1 | Set when a SOF is issued by the host controller. This triggers a USB interrupt when HSOFE = 1. When using the host controller in Low-speed mode, this bit is also set when a keep-alive is sent. |

**Bit 4 – RXRSMI** Upstream Resume Received Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTICR.RXRSMIC = 1. |
| 1 | Set when an Upstream Resume has been received from the device. |

**Bit 3 – RSMEDI** Downstream Resume Sent Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTICR.RSMEDIC = 1. |
| 1 | Set when a Downstream Resume has been sent to the device. |

**Bit 2 – RSTI** USB Reset Sent Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTICR.RSTIC = 1. |
| 1 | Set when a USB Reset has been sent to the device. |

**Bit 1 – DDISCI** Device Disconnection Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTICR.DDISCIC = 1. |
| 1 | Set when the device has been removed from the USB bus. |

**Bit 0 – DCONNI** Device Connection Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTICR.DCONNIC = 1. |
| 1 | Set when a new device has been connected to the USB bus. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.33 Host Global Interrupt Clear Register

**Name:** USBHS_HSTICR
**Offset:** 0x0408
**Property:** Write-only

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_HSTISR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | HWUPIC | HSOFIC | RXRSMIC | RSMEDIC | RSTIC | DDISCIC | DCONNIC |
| Reset | | | | | | | | |

**Bit 6 – HWUPIC** Host Wakeup Interrupt Clear

**Bit 5 – HSOFIC** Host Start of Frame Interrupt Clear

**Bit 4 – RXRSMIC** Upstream Resume Received Interrupt Clear

**Bit 3 – RSMEDIC** Downstream Resume Sent Interrupt Clear

**Bit 2 – RSTIC** USB Reset Sent Interrupt Clear

**Bit 1 – DDISCIC** Device Disconnection Interrupt Clear

**Bit 0 – DCONNIC** Device Connection Interrupt Clear

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.34 Host Global Interrupt Set Register

**Name:** USBHS_HSTIFR
**Offset:** 0x040C
**Property:** Write-only

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_HSTISR, which may be useful for test or debug purposes.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | HWUPIS | HSOFIS | RXRSMIS | RSMEDIS | RSTIS | DDISCIS | DCONNIS |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt Set

**Bit 6 – HWUPIS** Host Wakeup Interrupt Set

**Bit 5 – HSOFIS** Host Start of Frame Interrupt Set

**Bit 4 – RXRSMIS** Upstream Resume Received Interrupt Set

**Bit 3 – RSMEDIS** Downstream Resume Sent Interrupt Set

**Bit 2 – RSTIS** USB Reset Sent Interrupt Set

**Bit 1 – DDISCIS** Device Disconnection Interrupt Set

**Bit 0 – DCONNIS** Device Connection Interrupt Set

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.35 Host Global Interrupt Mask Register

**Name:** USBHS_HSTIMR
**Offset:** 0x0410
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PEP_9 | PEP_8 |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEP_7 | PEP_6 | PEP_5 | PEP_4 | PEP_3 | PEP_2 | PEP_1 | PEP_0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | HWUPIE | HSOFIE | RXRSMIE | RSMEDIE | RSTIE | DDISCIE | DCONNIE |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when the corresponding bit in USBHS_HSTIDR = 1. This disables the DMA Channel x Interrupt (USBHS_HSTISR.DMA_x). |
| 1 | Set when the corresponding bit in USBHS_HSTIER = 1. This enables the DMA Channel x Interrupt (USBHS_HSTISR.DMA_x). |

**Bits 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 – PEP_** Pipe x Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when PEP_x = 1. This disables the Pipe x Interrupt (PEP_x). |
| 1 | Set when the corresponding bit in USBHS_HSTIER = 1. This enables the Pipe x Interrupt (USBHS_HSTISR.PEP_x). |

**Bit 6 – HWUPIE** Host Wakeup Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTIDR.HWUPIEC = 1. This disables the Host Wakeup Interrupt (USBHS_HSTISR.HWUPI). |
| 1 | Set when USBHS_HSTIER.HWUPIES = 1. This enables the Host Wakeup Interrupt (USBHS_HSTISR.HWUPI). |

**Bit 5 – HSOFIE** Host Start of Frame Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTIDR.HSOFIEC = 1. This disables the Host Start of Frame interrupt (USBHS_HSTISR.HSOFI). |
| 1 | Set when USBHS_HSTIER.HSOFIES= 1. This enables the Host Start of Frame interrupt (USBHS_HSTISR.HSOFI). |

**Bit 4 – RXRSMIE** Upstream Resume Received Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTIDR.RXRSMIEC= 1. This disables the Downstream Resume interrupt (USBHS_HSTISR.RXRSMI). |
| 1 | Set when USBHS_HSTIER.RXRSMIES = 1. This enables the Upstream Resume Received interrupt (USBHS_HSTISR.RXRSMI). |

**Bit 3 – RSMEDIE** Downstream Resume Sent Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTIDR.RSMEDIEC = 1. This disables the Downstream Resume interrupt (USBHS_HSTISR.RSMEDI). |
| 1 | Set when USBHS_HSTIER.RSMEDIES = 1. This enables the Downstream Resume interrupt (USBHS_HSTISR.RSMEDI). |

**Bit 2 – RSTIE** USB Reset Sent Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTIDR.RSTIEC = 1. This disables the USB Reset Sent interrupt (USBHS_HSTISR.RSTI). |
| 1 | Set when USBHS_HSTIER.RSTIES = 1. This enables the USB Reset Sent interrupt (USBHS_HSTISR.RSTI). |

**Bit 1 – DDISCIE** Device Disconnection Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTIDR.DDISCIEC = 1. This disables the Device Disconnection interrupt (USBHS_HSTISR.DDISCI). |
| 1 | Set when USBHS_HSTIER.DDISCIES = 1. This enables the Device Disconnection interrupt (USBHS_HSTISR.DDISCI). |

**Bit 0 – DCONNIE** Device Connection Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTIDR.DCONNIEC = 1. This disables the Device Connection interrupt (USBHS_HSTISR.DCONNI). |
| 1 | Set when USBHS_HSTIER.DCONNIES = 1. This enables the Device Connection interrupt (USBHS_HSTISR.DCONNI). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.36 Host Global Interrupt Disable Register

**Name:** USBHS_HSTIDR
**Offset:** 0x0414
**Property:** Write-only

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_HSTIMR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PEP_9 | PEP_8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEP_7 | PEP_6 | PEP_5 | PEP_4 | PEP_3 | PEP_2 | PEP_1 | PEP_0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | HWUPIEC | HSOFIEC | RXRSMIEC | RSMEDIEC | RSTIEC | DDISCIEC | DCONNIEC |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt Disable

**Bits 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 – PEP_** Pipe x Interrupt Disable

**Bit 6 – HWUPIEC** Host Wakeup Interrupt Disable

**Bit 5 – HSOFIEC** Host Start of Frame Interrupt Disable

**Bit 4 – RXRSMIEC** Upstream Resume Received Interrupt Disable

**Bit 3 – RSMEDIEC** Downstream Resume Sent Interrupt Disable

**Bit 2 – RSTIEC** USB Reset Sent Interrupt Disable

**Bit 1 – DDISCIEC** Device Disconnection Interrupt Disable

**Bit 0 – DCONNIEC** Device Connection Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.37 Host Global Interrupt Enable Register

**Name:** USBHS_HSTIER
**Offset:** 0x0418
**Property:** Write-only

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_HSTISR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMA_6 | DMA_5 | DMA_4 | DMA_3 | DMA_2 | DMA_1 | DMA_0 | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PEP_9 | PEP_8 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PEP_7 | PEP_6 | PEP_5 | PEP_4 | PEP_3 | PEP_2 | PEP_1 | PEP_0 |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | HWUPIES | HSOFIES | RXRSMIES | RSMEDIES | RSTIES | DDISCIES | DCONNIES |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 25, 26, 27, 28, 29, 30, 31 – DMA_** DMA Channel x Interrupt Enable

**Bits 8, 9, 10, 11, 12, 13, 14, 15, 16, 17 – PEP_** Pipe x Interrupt Enable

**Bit 6 – HWUPIES** Host Wakeup Interrupt Enable

**Bit 5 – HSOFIES** Host Start of Frame Interrupt Enable

**Bit 4 – RXRSMIES** Upstream Resume Received Interrupt Enable

**Bit 3 – RSMEDIES** Downstream Resume Sent Interrupt Enable

**Bit 2 – RSTIES** USB Reset Sent Interrupt Enable

**Bit 1 – DDISCIES** Device Disconnection Interrupt Enable

**Bit 0 – DCONNIES** Device Connection Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.38 Host Frame Number Register

**Name:** USBHS_HSTFNUM
**Offset:** 0x0420
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FLENHIGH[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | FNUM[10:5] | | | | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FNUM[4:0] | | | | | MFNUM[2:0] | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – FLENHIGH[7:0]** Frame Length
In High-speed mode, this field contains the 8 high-order bits of the 16-bit internal frame counter (at 30 MHz, the counter length is 3750 to ensure a SOF generation every 125 µs).

**Bits 13:3 – FNUM[10:0]** Frame Number
This field contains the current SOF number.
This field can be written. In this case, the MFNUM field is reset to zero.

**Bits 2:0 – MFNUM[2:0]** Micro Frame Number
This field contains the current microframe number (can vary from 0 to 7), updated every 125 µs.
When operating in Full-speed mode, this field is tied to zero.

### 39.6.39 Host Address 1 Register

**Name:** USBHS_HSTADDR1
**Offset:** 0x0424
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | HSTADDRP3[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | HSTADDRP2[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | HSTADDRP1[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | HSTADDRP0[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:24 – HSTADDRP3[6:0]** USB Host Address
This field contains the address of the Pipe3 of the USB device.
This field is cleared when a USB reset is requested.

**Bits 22:16 – HSTADDRP2[6:0]** USB Host Address
This field contains the address of the Pipe2 of the USB device.
This field is cleared when a USB reset is requested.

**Bits 14:8 – HSTADDRP1[6:0]** USB Host Address
This field contains the address of the Pipe1 of the USB device.
This field is cleared when a USB reset is requested.

**Bits 6:0 – HSTADDRP0[6:0]** USB Host Address
This field contains the address of the Pipe0 of the USB device.
This field is cleared when a USB reset is requested.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.40 Host Address 2 Register

**Name:** USBHS_HSTADDR2
**Offset:** 0x0428
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | HSTADDRP7[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | HSTADDRP6[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | HSTADDRP5[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | HSTADDRP4[6:0] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:24 – HSTADDRP7[6:0]** USB Host Address
This field contains the address of the Pipe7 of the USB device.
This field is cleared when a USB reset is requested.

**Bits 22:16 – HSTADDRP6[6:0]** USB Host Address
This field contains the address of the Pipe6 of the USB device.
This field is cleared when a USB reset is requested.

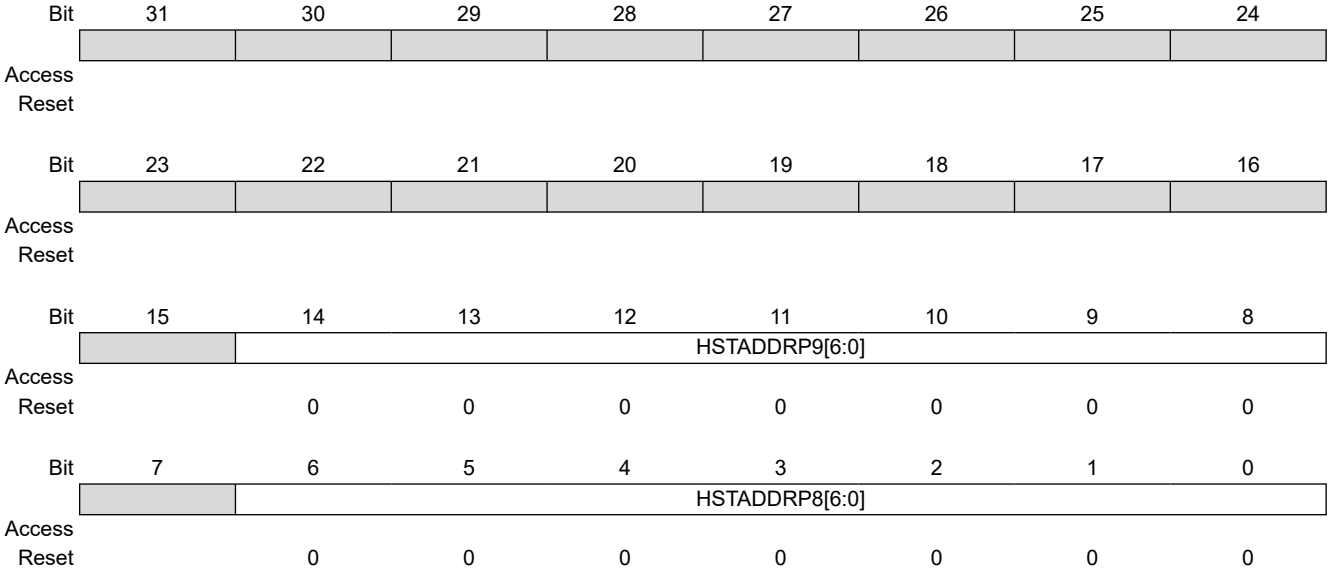
**Bits 14:8 – HSTADDRP5[6:0]** USB Host Address
This field contains the address of the Pipe5 of the USB device.
This field is cleared when a USB reset is requested.

**Bits 6:0 – HSTADDRP4[6:0]** USB Host Address
This field contains the address of the Pipe4 of the USB device.
This field is cleared when a USB reset is requested.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.41 Host Address 3 Register

**Name:** USBHS_HSTADDR3
**Offset:** 0x042C
**Reset:** 0x00000000
**Property:** Read/Write




**Bits 14:8 – HSTADDRP9[6:0]** USB Host Address
This field contains the address of the Pipe9 of the USB device.
This field is cleared when a USB reset is requested.

**Bits 6:0 – HSTADDRP8[6:0]** USB Host Address
This field contains the address of the Pipe8 of the USB device.
This field is cleared when a USB reset is requested.

### 39.6.42 Host Pipe Register

**Name:** USBHS_HSTPIP
**Offset:** 0x0041C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PRST8 |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PRST7 | PRST6 | PRST5 | PRST4 | PRST3 | PRST2 | PRST1 | PRST0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | PEN8 |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PEN7 | PEN6 | PEN5 | PEN4 | PEN3 | PEN2 | PEN1 | PEN0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23, 24 – PRST** Pipe x Reset

| Value | Description |
|---|---|
| 0 | Completes the reset operation and allows to start using the FIFO. |
| 1 | Resets the Pipe x FIFO. This resets the pipe x registers (USBHS_HSTPIPCFGx, USBHS_HSTPIPISRx, USBHS_HSTPIPIMRx), but not the pipe configuration (ALLOC, PBK, PSIZE, PTOKEN, PTYPE, PEPNUM, INTFRQ). The whole pipe mechanism (FIFO counter, reception, transmission, etc.) is reset, apart from the Data Toggle management. The pipe configuration remains active and the pipe is still enabled. |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8 – PEN** Pipe x Enable

| Value | Description |
|---|---|
| 0 | Disables Pipe x, which forces the Pipe x state to inactive and resets the pipe x registers (USBHS_HSTPIPCFGx, USBHS_HSTPIPISRx, USBHS_HSTPIPIMRx), but not the pipe configuration (USBHS_HSTPIPCFGx.ALLOC, USBHS_HSTPIPCFGx.PBK, USBHS_HSTPIPCFGx.PSIZE). |
| 1 | Enables Pipe x. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.43 Host Pipe x Configuration Register

**Name:** USBHS_HSTPIPCFGx
**Offset:** 0x0500 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

For High-speed Bulk-out Pipe, see "Host Pipe x Configuration Register (High-speed Bulk-out or High-speed Control Pipe)".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | INTFRQ[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PEPNUM[3:0] | | | |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bits 31:24 – INTFRQ[7:0]** Pipe Interrupt Request Frequency
This field contains the maximum value in milliseconds of the polling period for an Interrupt Pipe.
This value has no effect for a non-Interrupt Pipe.
This field is cleared upon sending a USB reset.

**Bits 19:16 – PEPNUM[3:0]** Pipe Endpoint Number
This field contains the number of the endpoint targeted by the pipe. This value is from 0 to 10.
This field is cleared upon sending a USB reset.

**Bits 13:12 – PTYPE[1:0]** Pipe Type
This field contains the pipe type.
This field is cleared upon sending a USB reset.

| Value | Name | Description |
|---|---|---|
| 0 | CTRL | Control |
| 1 | ISO | Isochronous |
| 2 | BLK | Bulk |
| 3 | INTRPT | Interrupt |

**Bit 10 – AUTOSW** Automatic Switch
This bit is cleared upon sending a USB reset.

| Value | Description |
|---|---|
| 0 | The automatic bank switching is disabled. |
| 1 | The automatic bank switching is enabled. |

**Bits 9:8 – PTOKEN[1:0]** Pipe Token
This field contains the pipe token.

| Value | Name | Description |
|---|---|---|
| 0 | SETUP | SETUP |
| 1 | IN | IN |
| 2 | OUT | OUT |
| 3 | Reserved | |

**Bits 6:4 – PSIZE[2:0]** Pipe Size
This field contains the size of each pipe bank.
This field is cleared upon sending a USB reset.

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8 bytes |
| 1 | 16_BYTE | 16 bytes |
| 2 | 32_BYTE | 32 bytes |
| 3 | 64_BYTE | 64 bytes |
| 4 | 128_BYTE | 128 bytes |
| 5 | 256_BYTE | 256 bytes |
| 6 | 512_BYTE | 512 bytes |
| 7 | 1024_BYTE | 1024 bytes |

**Bits 3:2 – PBK[1:0]** Pipe Banks
This field contains the number of banks for the pipe.
For control pipes, a single-bank pipe (0b00) should be selected.
This field is cleared upon sending a USB reset.

| Value | Name | Description |
|---|---|---|
| 0 | 1_BANK | Single-bank pipe |
| 1 | 2_BANK | Double-bank pipe |
| 2 | 3_BANK | Triple-bank pipe |
| 3 | Reserved | |

**Bit 1 – ALLOC** Pipe Memory Allocate
This bit is cleared when a USB Reset is requested.
Refer to "DPRAM Management" for more details.

| Value | Description |
|---|---|
| 0 | Frees the pipe memory. |
| 1 | Allocates the pipe memory. |

### 39.6.44 Host Pipe x Configuration Register (High-speed Bulk-out or High-speed Control Pipe)

**Name:** USBHS_HSTPIPCFGx (HSBOHSCP)
**Offset:** 0x0500 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This configuration is relevant only if PTYPE = 0x0 or 0x2 in "Host Pipe x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BINTERVAL[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | PINGEN | PEPNUM[3:0] | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PTYPE[1:0] | | | AUTOSW | PTOKEN[1:0] | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | PSIZE[2:0] | | | PBK[1:0] | | ALLOC | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | |

**Bits 31:24 – BINTERVAL[7:0]** bInterval Parameter for the Bulk-Out/Ping Transaction
This field contains the Ping/Bulk-out period.
• If BINTERVAL > 0 and PINGEN = 1, one PING token is sent every bInterval microframe until it is ACKed by the peripheral.
• If BINTERVAL = 0 and PINGEN = 1, multiple consecutive PING tokens are sent in the same microframe until they are ACKed.
• If BINTERVAL > 0 and PINGEN = 0, one OUT token is sent every bInterval microframe until it is ACKed by the peripheral.
• If BINTERVAL = 0 and PINGEN = 0, multiple consecutive OUT tokens are sent in the same microframe until they are ACKed.
This value must be in the range from 0 to 255.

**Bit 20 – PINGEN** Ping Enable
This bit is relevant for High-speed Bulk-out transaction only (including the control data stage and the control status stage).
This bit is cleared upon sending a USB reset.

| Value | Description |
|---|---|
| 0 | Disables the ping protocol. |
| 1 | Enables the ping mechanism according to the USB 2.0 Standard. |

**Bits 19:16 – PEPNUM[3:0]** Pipe Endpoint Number
This field contains the number of the endpoint targeted by the pipe. This value is from 0 to 10.
This field is cleared upon sending a USB reset.

**Bits 13:12 – PTYPE[1:0]** Pipe Type
This field contains the pipe type.
This field is cleared upon sending a USB reset.

| Value | Name | Description |
|---|---|---|
| 0 | CTRL | Control |
| 1 | Reserved | |
| 2 | BLK | Bulk |
| 3 | Reserved | |

**Bit 10 – AUTOSW** Automatic Switch
This bit is cleared upon sending a USB reset.

| Value | Description |
|---|---|
| 0 | The automatic bank switching is disabled. |
| 1 | The automatic bank switching is enabled. |

**Bits 9:8 – PTOKEN[1:0]** Pipe Token
This field contains the pipe token.

| Value | Name | Description |
|---|---|---|
| 0 | SETUP | SETUP |
| 1 | IN | IN |
| 2 | OUT | OUT |
| 3 | Reserved | |

**Bits 6:4 – PSIZE[2:0]** Pipe Size
This field contains the size of each pipe bank.
This field is cleared upon sending a USB reset.

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8 bytes |
| 1 | 16_BYTE | 16 bytes |
| 2 | 32_BYTE | 32 bytes |
| 3 | 64_BYTE | 64 bytes |
| 4 | 128_BYTE | 128 bytes |
| 5 | 256_BYTE | 256 bytes |
| 6 | 512_BYTE | 512 bytes |
| 7 | 1024_BYTE | 1024 bytes |

**Bits 3:2 – PBK[1:0]** Pipe Banks
This field contains the number of banks for the pipe.
For control pipes, a single-bank pipe (0b00) should be selected.
This field is cleared upon sending a USB reset.

| Value | Name | Description |
|---|---|---|
| 0 | 1_BANK | Single-bank pipe |
| 1 | 2_BANK | Double-bank pipe |
| 2 | 3_BANK | Triple-bank pipe |
| 3 | Reserved | |

**Bit 1 – ALLOC** Pipe Memory Allocate
This bit is cleared when a USB Reset is requested.
Refer to "DPRAM Management" for more details.

| Value | Description |
|---|---|
| 0 | Frees the pipe memory. |
| 1 | Allocates the pipe memory. |

### 39.6.45 Host Pipe x Status Register (Control, Bulk Pipes)

**Name:** USBHS_HSTPIPISRx
**Offset:** 0x0530 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x0 or 0x2 in "Host Pipe x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PBYCT[10:4] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | TXSTPI | TXOUTI | RXINI |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:20 – PBYCT[10:0]** Pipe Byte Count
This field contains the byte count of the FIFO.
For an OUT pipe, the field is incremented after each byte written by the user into the pipe and decremented after each byte sent to the peripheral.
For an IN pipe, the field is incremented after each byte received from the peripheral and decremented after each byte read by the user from the pipe.
This field may be updated 1 clock cycle after the RWALL bit changes, so the user should not poll this field as an interrupt bit.

**Bit 18 – CFGOK** Configuration OK Status
This bit is set/cleared when the USBHS_HSTPIPCFGx.ALLOC bit is set.
This bit is set if the pipe x number of banks (USBHS_HSTPIPCFGx.PBK) and size (USBHS_HSTPIPCFGx.PSIZE) are correct compared to the maximal allowed number of banks and size for this pipe and to the maximal FIFO size (i.e., the DPRAM size).
If this bit is cleared, the user should rewrite correct values for the PBK and PSIZE fields in the USBHS_HSTPIPCFGx register.

**Bit 16 – RWALL** Read/Write Allowed
For an OUT pipe, this bit is set when the current bank is not full, i.e., the software can write further data into the FIFO.
For an IN pipe, this bit is set when the current bank is not empty, i.e., the software can read further data from the FIFO.
This bit is cleared otherwise.
This bit is also cleared when the RXSTALLDI or the PERRI bit = 1.

**Bits 15:14 – CURRBK[1:0]** Current Bank
For non-control pipe, this field indicates the number of the current bank.
This field may be updated 1 clock cycle after the RWALL bit changes, so the user should not poll it as an interrupt bit.

| Value | Name | Description |
|---|---|---|
| 0 | BANK0 | Current bank is bank0 |
| 1 | BANK1 | Current bank is bank1 |
| 2 | BANK2 | Current bank is bank2 |
| 3 | Reserved | |

**Bits 13:12 – NBUSYBK[1:0]** Number of Busy Banks
This field indicates the number of busy banks.
For an OUT pipe, this field indicates the number of busy banks, filled by the user, ready for OUT transfer. When all banks are busy, this triggers a PEP_x interrupt if USBHS_HSTPIPIMRx.NBUSYBKE = 1.
For an IN pipe, this field indicates the number of busy banks filled by IN transaction from the Device. When all banks are free, this triggers a PEP_x interrupt if USBHS_HSTPIPIMRx.NBUSYBKE = 1.

| Value | Name | Description |
|---|---|---|
| 0 | 0_BUSY | 0 busy bank (all banks free) |
| 1 | 1_BUSY | 1 busy bank |
| 2 | 2_BUSY | 2 busy banks |
| 3 | 3_BUSY | 3 busy banks |

**Bits 9:8 – DTSEQ[1:0]** Data Toggle Sequence
This field indicates the data PID of the current bank.
For an OUT pipe, this field indicates the data toggle of the next packet that is to be sent.
For an IN pipe, this field indicates the data toggle of the received packet stored in the current bank.

| Value | Name | Description |
|---|---|---|
| 0 | DATA0 | Data0 toggle sequence |
| 1 | DATA1 | Data1 toggle sequence |
| 2 | Reserved | |
| 3 | Reserved | |

**Bit 7 – SHORTPACKETI** Short Packet Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.SHORTPACKETIC = 1. |
| 1 | Set when a short packet is received by the host controller (packet length inferior to the PSIZE programmed field). |

**Bit 6 – RXSTALLDI** Received STALLed Interrupt
This bit is set when a STALL handshake has been received on the current bank of the pipe. The pipe is automatically frozen. This triggers an interrupt if USBHS_HSTPIPIMR.RXSTALLE = 1.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.RXSTALLDIC = 1. |

**Bit 5 – OVERFI** Overflow Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.OVERFIC = 1. |
| 1 | Set when the current pipe has received more data than the maximum length of the current pipe. An interrupt is triggered if USBHS_HSTPIPIMR.OVERFIE = 1. |

**Bit 4 – NAKEDI** NAKed Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.NAKEDIC = 1. |
| 1 | Set when a NAK has been received on the current bank of the pipe. This triggers an interrupt if USBHS_HSTPIPIMR.NAKEDE = 1. |

**Bit 3 – PERRI** Pipe Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the error source bit is cleared. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | Set when an error occurs on the current bank of the pipe. This triggers an interrupt if the USBHS_HSTPIPIMR.PERRE bit is set. Refer to the USBHS_HSTPIPERRx register to determine the source of the error. |

**Bit 2 – TXSTPI** Transmitted SETUP Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.TXSTPIC = 1. |
| 1 | Set, for control pipes, when the current SETUP bank is free and can be filled. This triggers an interrupt if USBHS_HSTPIPIMR.TXSTPE = 1. |

**Bit 1 – TXOUTI** Transmitted OUT Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.TXOUTIC = 1. |
| 1 | Set when the current OUT bank is free and can be filled. This triggers an interrupt if USBHS_HSTPIPIMR.TXOUTE = 1. |

**Bit 0 – RXINI** Received IN Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.RXINIC = 1. |
| 1 | Set when a new USB message is stored in the current bank of the pipe. This triggers an interrupt if USBHS_HSTPIPIMR.RXINE = 1. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.46 Host Pipe x Status Register (Interrupt Pipes)

**Name:** USBHS_HSTPIPISRx (INTPIPES)
**Offset:** 0x0530 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x3 in "Host Pipe x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PBYCT[10:4] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETI | RXSTALLDI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:20 – PBYCT[10:0]** Pipe Byte Count
This field contains the byte count of the FIFO.
For an OUT pipe, the field is incremented after each byte written by the user into the pipe and decremented after each byte sent to the peripheral.
For an IN pipe, the field is incremented after each byte received from the peripheral and decremented after each byte read by the user from the pipe.
This field may be updated 1 clock cycle after the RWALL bit changes, so the user should not poll this field as an interrupt bit.

**Bit 18 – CFGOK** Configuration OK Status
This bit is set/cleared when the USBHS_HSTPIPCFGx.ALLOC bit is set.
This bit is set if the pipe x number of banks (USBHS_HSTPIPCFGx.PBK) and size (USBHS_HSTPIPCFGx.PSIZE) are correct compared to the maximal allowed number of banks and size for this pipe, and to the maximal FIFO size (i.e., the DPRAM size).
If this bit is cleared, the user should rewrite correct values for the PBK and PSIZE fields in the USBHS_HSTPIPCFGx register.

**Bit 16 – RWALL** Read/Write Allowed
For an OUT pipe, this bit is set when the current bank is not full, i.e., the software can write further data into the FIFO.
For an IN pipe, this bit is set when the current bank is not empty, i.e., the software can read further data from the FIFO.
This bit is cleared otherwise.
This bit is also cleared when RXSTALLDI or PERRI = 1.

**Bits 15:14 – CURRBK[1:0]** Current Bank
For a non-control pipe, this field indicates the number of the current bank.
This field may be updated 1 clock cycle after the RWALL bit changes, so the user should not poll it as an interrupt bit.

| Value | Name | Description |
|---|---|---|
| 0 | BANK0 | Current bank is bank0 |
| 1 | BANK1 | Current bank is bank1 |
| 2 | BANK2 | Current bank is bank2 |
| 3 | Reserved | |

**Bits 13:12 – NBUSYBK[1:0]** Number of Busy Banks
This field indicates the number of busy banks.
For an OUT pipe, this field indicates the number of busy banks, filled by the user, ready for an OUT transfer. When all banks are busy, this triggers a PEP_x interrupt if USBHS_HSTPIPIMRx.NBUSYBKE = 1.
For an IN pipe, this field indicates the number of busy banks filled by IN transaction from the device. When all banks are free, this triggers a PEP_x interrupt if USBHS_HSTPIPIMRx.NBUSYBKE = 1.

| Value | Name | Description |
|---|---|---|
| 0 | 0_BUSY | 0 busy bank (all banks free) |
| 1 | 1_BUSY | 1 busy bank |
| 2 | 2_BUSY | 2 busy banks |
| 3 | 3_BUSY | 3 busy banks |

**Bits 9:8 – DTSEQ[1:0]** Data Toggle Sequence
This field indicates the data PID of the current bank.
For an OUT pipe, this field indicates the data toggle of the next packet that is to be sent.
For an IN pipe, this field indicates the data toggle of the received packet stored in the current bank.

| Value | Name | Description |
|---|---|---|
| 0 | DATA0 | Data0 toggle sequence |
| 1 | DATA1 | Data1 toggle sequence |
| 2 | Reserved | |
| 3 | Reserved | |

**Bit 7 – SHORTPACKETI** Short Packet Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.SHORTPACKETIC = 1. |
| 1 | Set when a short packet is received by the host controller (packet length inferior to the PSIZE programmed field). |

**Bit 6 – RXSTALLDI** Received STALLed Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.RXSTALLDIC = 1. |
| 1 | Set when a STALL handshake has been received on the current bank of the pipe. The pipe is automatically frozen. This triggers an interrupt if USBHS_HSTPIPIMR.RXSTALLE = 1. |

**Bit 5 – OVERFI** Overflow Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.OVERFIC = 1. |
| 1 | Set when the current pipe has received more data than the maximum length of the current pipe. An interrupt is triggered if the USBHS_HSTPIPIMR.OVERFIE bit = 1. |

**Bit 4 – NAKEDI** NAKed Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.NAKEDIC = 1. |
| 1 | Set when a NAK has been received on the current bank of the pipe. This triggers an interrupt if the USBHS_HSTPIPIMR.NAKEDE bit = 1. |

**Bit 3 – PERRI** Pipe Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the error source bit is cleared. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | Set when an error occurs on the current bank of the pipe. This triggers an interrupt if the USBHS_HSTPIPIMR.PERRE bit is set. Refer to the USBHS_HSTPIPERRx register to determine the source of the error. |

**Bit 2 – UNDERFI** Underflow Interrupt
This bit is set, for an isochronous and interrupt IN/OUT pipe, when an error flow occurs. This triggers an interrupt if UNDERFIE = 1.
This bit is set, for an isochronous or interrupt OUT pipe, when a transaction underflow occurs in the current pipe (the pipe cannot send the OUT data packet in time because the current bank is not ready). A zero-length-packet (ZLP) is sent instead.
This bit is set, for an isochronous or interrupt IN pipe, when a transaction flow error occurs in the current pipe, i.e, the current bank of the pipe is not free while a new IN USB packet is received. This packet is not stored in the bank. For an interrupt pipe, the overflowed packet is ACKed to comply with the USB standard.
This bit is cleared when USBHS_HSTPIPICR.UNDERFIEC = 1.

**Bit 1 – TXOUTI** Transmitted OUT Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.TXOUTIC = 1. |
| 1 | Set when the current OUT bank is free and can be filled. This triggers an interrupt if USBHS_HSTPIPIMR.TXOUTE = 1. |

**Bit 0 – RXINI** Received IN Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.RXINIC = 1. |
| 1 | Set when a new USB message is stored in the current bank of the pipe. This triggers an interrupt if the USBHS_HSTPIPIMR.RXINE bit = 1. |

### 39.6.47 Host Pipe x Status Register (Isochronous Pipes)

**Name:** USBHS_HSTPIPISRx (ISOPIPES)
**Offset:** 0x0530 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x1 in "Host Pipe x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | PBYCT[10:4] | | | | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PBYCT[3:0] | | | | | CFGOK | | RWALL |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CURRBK[1:0] | | NBUSYBK[1:0] | | | | DTSEQ[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETI | CRCERRI | OVERFI | NAKEDI | PERRI | UNDERFI | TXOUTI | RXINI |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 30:20 – PBYCT[10:0]** Pipe Byte Count
This field contains the byte count of the FIFO.
For an OUT pipe, the field is incremented after each byte written by the user into the pipe and decremented after each byte sent to the peripheral.
For an IN pipe, the field is incremented after each byte received from the peripheral and decremented after each byte read by the user from the pipe.
This field may be updated 1 clock cycle after the RWALL bit changes, so the user should not poll this field as an interrupt bit.

**Bit 18 – CFGOK** Configuration OK Status
This bit is set/cleared when the USBHS_HSTPIPCFGx.ALLOC bit is set.
This bit is set if the pipe x number of banks (USBHS_HSTPIPCFGx.PBK) and size (USBHS_HSTPIPCFGx.PSIZE) are correct compared to the maximal allowed number of banks and size for this pipe and to the maximal FIFO size (i.e., the DPRAM size).
If this bit is cleared, the user should rewrite correct values for the PBK and PSIZE fields in the USBHS_HSTPIPCFGx register.

**Bit 16 – RWALL** Read/Write Allowed
For an OUT pipe, this bit is set when the current bank is not full, i.e., the software can write further data into the FIFO.
For an IN pipe, this bit is set when the current bank is not empty, i.e., the software can read further data from the FIFO.
This bit is cleared otherwise.
This bit is also cleared when the RXSTALLDI or the PERRI bit = 1.

**Bits 15:14 – CURRBK[1:0]** Current Bank
For a non-control pipe, this field indicates the number of the current bank.
This field may be updated 1 clock cycle after the RWALL bit changes, so the user should not poll it as an interrupt bit.

© 2021 Microchip Technology Inc. and its subsidiaries

| Value | Name | Description |
|---|---|---|
| 0 | BANK0 | Current bank is bank0 |
| 1 | BANK1 | Current bank is bank1 |
| 2 | BANK2 | Current bank is bank2 |
| 3 | Reserved | |

**Bits 13:12 – NBUSYBK[1:0]** Number of Busy Banks
This field indicates the number of busy banks.
For an OUT pipe, this field indicates the number of busy banks, filled by the user, ready for an OUT transfer. When all banks are busy, this triggers a PEP_x interrupt if USBHS_HSTPIPIMRx.NBUSYBKE = 1.
For an IN pipe, this field indicates the number of busy banks filled by IN transaction from the device. When all banks are free, this triggers a PEP_x interrupt if USBHS_HSTPIPIMRx.NBUSYBKE = 1.

| Value | Name | Description |
|---|---|---|
| 0 | 0_BUSY | 0 busy bank (all banks free) |
| 1 | 1_BUSY | 1 busy bank |
| 2 | 2_BUSY | 2 busy banks |
| 3 | 3_BUSY | 3 busy banks |

**Bits 9:8 – DTSEQ[1:0]** Data Toggle Sequence
This field indicates the data PID of the current bank.
For an OUT pipe, this field indicates the data toggle of the next packet that is to be sent.
For an IN pipe, this field indicates the data toggle of the received packet stored in the current bank.

| Value | Name | Description |
|---|---|---|
| 0 | DATA0 | Data0 toggle sequence |
| 1 | DATA1 | Data1 toggle sequence |
| 2 | Reserved | |
| 3 | Reserved | |

**Bit 7 – SHORTPACKETI** Short Packet Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.SHORTPACKETIC = 1. |
| 1 | Set when a short packet is received by the host controller (packet length inferior to the PSIZE programmed field). |

**Bit 6 – CRCERRI** CRC Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.CRCERRIC = 1. |
| 1 | Set when a CRC error occurs on the current bank of the pipe. This triggers an interrupt if the USBHS_HSTPIPIMR.TXSTPE bit = 1. |

**Bit 5 – OVERFI** Overflow Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.OVERFIC = 1. |
| 1 | Set when the current pipe has received more data than the maximum length of the current pipe. An interrupt is triggered if the USBHS_HSTPIPIMR.OVERFIE bit = 1. |

**Bit 4 – NAKEDI** NAKed Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.NAKEDIC = 1. |
| 1 | Set when a NAK has been received on the current bank of the pipe. This triggers an interrupt if the USBHS_HSTPIPIMR.NAKEDE bit = 1. |

**Bit 3 – PERRI** Pipe Error Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when the error source bit is cleared. |

| Value | Description |
|---|---|
| 1 | Set when an error occurs on the current bank of the pipe. This triggers an interrupt if the USBHS_HSTPIPIMR.PERRE bit is set. Refer to the USBHS_HSTPIPERRx register to determine the source of the error. |

**Bit 2 – UNDERFI** Underflow Interrupt
This bit is set, for an isochronous and interrupt IN/OUT pipe, when an error flow occurs. This triggers an interrupt if the UNDERFIE bit = 1.
This bit is set, for an isochronous or interrupt OUT pipe, when a transaction underflow occurs in the current pipe (the pipe cannot send the OUT data packet in time because the current bank is not ready). A zero-length-packet (ZLP) is sent instead.
This bit is set, for an isochronous or interrupt IN pipe, when a transaction flow error occurs in the current pipe, i.e, the current bank of the pipe is not free while a new IN USB packet is received. This packet is not stored in the bank. For an interrupt pipe, the overflowed packet is ACKed to comply with the USB standard.
This bit is cleared when USBHS_HSTPIPICR.UNDERFIEC = 1.

**Bit 1 – TXOUTI** Transmitted OUT Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.TXOUTIC = 1. |
| 1 | Set when the current OUT bank is free and can be filled. This triggers an interrupt if USBHS_HSTPIPIMR.TXOUTE = 1. |

**Bit 0 – RXINI** Received IN Data Interrupt

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPICR.RXINIC = 1. |
| 1 | Set when a new USB message is stored in the current bank of the pipe. This triggers an interrupt if USBHS_HSTPIPIMR.RXINE = 1. |

### 39.6.48 Host Pipe x Clear Register (Control, Bulk Pipes)

**Name:** USBHS_HSTPIPICRx
**Offset:** 0x0560 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x0 or 0x2 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Status Register (Control, Bulk Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_HSTPIPISRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | TXSTPIC | TXOUTIC | RXINIC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 7 – SHORTPACKETIC** Short Packet Interrupt Clear

**Bit 6 – RXSTALLDIC** Received STALLed Interrupt Clear

**Bit 5 – OVERFIC** Overflow Interrupt Clear

**Bit 4 – NAKEDIC** NAKed Interrupt Clear

**Bit 2 – TXSTPIC** Transmitted SETUP Interrupt Clear

**Bit 1 – TXOUTIC** Transmitted OUT Data Interrupt Clear

**Bit 0 – RXINIC** Received IN Data Interrupt Clear

### 39.6.49 Host Pipe x Clear Register (Interrupt Pipes)

**Name:** USBHS_HSTPIPICRx (INTPIPES)
**Offset:** 0x0560 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x3 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Status Register (Interrupt Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_HSTPIPISRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIC | RXSTALLDIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 7 – SHORTPACKETIC** Short Packet Interrupt Clear

**Bit 6 – RXSTALLDIC** Received STALLed Interrupt Clear

**Bit 5 – OVERFIC** Overflow Interrupt Clear

**Bit 4 – NAKEDIC** NAKed Interrupt Clear

**Bit 2 – UNDERFIC** Underflow Interrupt Clear

**Bit 1 – TXOUTIC** Transmitted OUT Data Interrupt Clear

**Bit 0 – RXINIC** Received IN Data Interrupt Clear

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.50 Host Pipe x Clear Register (Isochronous Pipes)

**Name:** USBHS_HSTPIPICRx (ISOPIPES)
**Offset:** 0x0560 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x1 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Status Register (Isochronous Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_HSTPIPISRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIC | CRCERRIC | OVERFIC | NAKEDIC | | UNDERFIC | TXOUTIC | RXINIC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 7 – SHORTPACKETIC** Short Packet Interrupt Clear

**Bit 6 – CRCERRIC** CRC Error Interrupt Clear

**Bit 5 – OVERFIC** Overflow Interrupt Clear

**Bit 4 – NAKEDIC** NAKed Interrupt Clear

**Bit 2 – UNDERFIC** Underflow Interrupt Clear

**Bit 1 – TXOUTIC** Transmitted OUT Data Interrupt Clear

**Bit 0 – RXINIC** Received IN Data Interrupt Clear

### 39.6.51 Host Pipe x Set Register (Control, Bulk Pipes)

**Name:** USBHS_HSTPIPIFRx
**Offset:** 0x0590
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x0 or 0x2 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Status Register (Control, Bulk Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_HSTPIPISRx, which may be useful for test or debug purposes.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | NBUSYBKS | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | TXSTPIS | TXOUTIS | RXINIS |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 12 – NBUSYBKS** Number of Busy Banks Set

**Bit 7 – SHORTPACKETIS** Short Packet Interrupt Set

**Bit 6 – RXSTALLDIS** Received STALLed Interrupt Set

**Bit 5 – OVERFIS** Overflow Interrupt Set

**Bit 4 – NAKEDIS** NAKed Interrupt Set

**Bit 3 – PERRIS** Pipe Error Interrupt Set

**Bit 2 – TXSTPIS** Transmitted SETUP Interrupt Set

**Bit 1 – TXOUTIS** Transmitted OUT Data Interrupt Set

**Bit 0 – RXINIS** Received IN Data Interrupt Set

### 39.6.52 Host Pipe x Set Register (Interrupt Pipes)

**Name:** USBHS_HSTPIPIFRx (INTPIPES)
**Offset:** 0x0590 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x3 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Status Register (Interrupt Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_HSTPIPISRx, which may be useful for test or debug purposes.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | NBUSYBKS | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIS | RXSTALLDIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 12 – NBUSYBKS** Number of Busy Banks Set

**Bit 7 – SHORTPACKETIS** Short Packet Interrupt Set

**Bit 6 – RXSTALLDIS** Received STALLed Interrupt Set

**Bit 5 – OVERFIS** Overflow Interrupt Set

**Bit 4 – NAKEDIS** NAKed Interrupt Set

**Bit 3 – PERRIS** Pipe Error Interrupt Set

**Bit 2 – UNDERFIS** Underflow Interrupt Set

**Bit 1 – TXOUTIS** Transmitted OUT Data Interrupt Set

**Bit 0 – RXINIS** Received IN Data Interrupt Set

© 2021 Microchip Technology Inc. and its subsidiaries

### 39.6.53 Host Pipe x Set Register (Isochronous Pipes)

**Name:** USBHS_HSTPIPIFRx (ISOPIPES)
**Offset:** 0x0590 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x1 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Status Register (Isochronous Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_HSTPIPISRx, which may be useful for test or debug purposes.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | NBUSYBKS | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIS | CRCERRIS | OVERFIS | NAKEDIS | PERRIS | UNDERFIS | TXOUTIS | RXINIS |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 12 – NBUSYBKS** Number of Busy Banks Set

**Bit 7 – SHORTPACKETIS** Short Packet Interrupt Set

**Bit 6 – CRCERRIS** CRC Error Interrupt Set

**Bit 5 – OVERFIS** Overflow Interrupt Set

**Bit 4 – NAKEDIS** NAKed Interrupt Set

**Bit 3 – PERRIS** Pipe Error Interrupt Set

**Bit 2 – UNDERFIS** Underflow Interrupt Set

**Bit 1 – TXOUTIS** Transmitted OUT Data Interrupt Set

**Bit 0 – RXINIS** Received IN Data Interrupt Set

© 2021 Microchip Technology Inc. and its subsidiaries

### 39.6.54 Host Pipe x Mask Register (Control, Bulk Pipes)

**Name:** USBHS_HSTPIPIMRx
**Offset:** 0x05C0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x0 or 0x2 in "Host Pipe x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTDT | PFREEZE | PDISHDMA |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCON | | NBUSYBKE | | | | |
| Access | | | | | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | TXSTPE | TXOUTE | RXINE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – RSTDT** Reset Data Toggle

| Value | Description |
|---|---|
| 0 | No reset of the Data Toggle is ongoing. |
| 0 | Set when USBHS_HSTPIPIER.RSTDTS = 1. This resets the Data Toggle to its initial value for the current pipe. |

**Bit 17 – PFREEZE** Pipe Freeze
This freezes the pipe request generation.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.PFREEZEC = 1. This enables the pipe request generation. |
| 1 | Set when one of the following conditions is met:<br>• USBHS_HSTPIPIER.PFREEZES=<br>• The pipe is not configured.<br>• A STALL handshake has been received on the pipe.<br>• An error has occurred on the pipe (USBHS_HSTPIPISR.PERRI = 1).<br>• (INRQ+1) In requests have been processed.<br>• A Pipe Reset (USBHS_HSTPIP.PRSTx rising) has occurred.<br>• A Pipe Enable (USBHS_HSTPIP.PEN rising) has occurred. |

**Bit 16 – PDISHDMA** Pipe Interrupts Disable HDMA Request Enable
See the USBHS_DEVEPTIMR.EPDISHDMA bit description.

**Bit 14 – FIFOCON** FIFO Control
For OUT and SETUP pipes:

0: Cleared when USBHS_HSTPIPIDR.FIFOCONC = 1. This sends the FIFO data and switches the bank.
1: Set when the current bank is free, at the same time as USBHS_HSTPIPISR.TXOUTI or TXSTPI.
For an IN pipe:
0: Cleared when USBHS_HSTPIPIDR.FIFOCONC = 1. This frees the current bank and switches to the next bank.
1: Set when a new IN message is stored in the current bank, at the same time as USBHS_HSTPIPISR.RXINI.

**Bit 12 – NBUSYBKE** Number of Busy Banks Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.NBUSYBKEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NBUSYBKE). |
| 1 | Set when USBHS_HSTPIPIER.NBUSYBKES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NBUSYBKE). |

**Bit 7 – SHORTPACKETIE** Short Packet Interrupt Enable
If this bit is set for non-control OUT pipes, a short packet transmission is guaranteed upon ending a DMA transfer, thus signaling an end of transfer, provided that End of DMA Buffer Output Enable (USBHS_HSTDMACONTROL.END_B_EN) and Automatic Switch (USBHS_HSTPIPCFG.AUTOSW) = 1.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.SHORTPACKETEC = 1. This disables the Transmitted IN Data IT (USBHS_HSTPIPIMR.SHORTPACKETE). |
| 1 | Set when USBHS_HSTPIPIER.SHORTPACKETIES = 1. This enables the Transmitted IN Data IT (USBHS_HSTPIPIMR.SHORTPACKETIE). |

**Bit 6 – RXSTALLDE** Received STALLed Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.RXSTALLDEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXSTALLDE). |
| 1 | Set when USBHS_HSTPIPIER.RXSTALLDES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXSTALLDE). |

**Bit 5 – OVERFIE** Overflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.OVERFIEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.OVERFIE). |
| 1 | Set when USBHS_HSTPIPIER.OVERFIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.OVERFIE). |

**Bit 4 – NAKEDE** NAKed Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.NAKEDEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NAKEDE). |
| 1 | Set when USBHS_HSTPIPIER.NAKEDES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NAKEDE). |

**Bit 3 – PERRE** Pipe Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.PERREC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.PERRE). |
| 1 | Set when USBHS_HSTPIPIER.PERRES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.PERRE). |

**Bit 2 – TXSTPE** Transmitted SETUP Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.TXSTPEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXSTPE). |
| 1 | Set when USBHS_HSTPIPIER.TXSTPES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXSTPE). |

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 1 – TXOUTE** Transmitted OUT Data Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.TXOUTEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXOUTE). |
| 1 | Set when USBHS_HSTPIPIER.TXOUTES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXOUTE). |

**Bit 0 – RXINE** Received IN Data Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.RXINEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXINE). |
| 1 | Set when USBHS_HSTPIPIER.RXINES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXINE). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.55 Host Pipe x Mask Register (Interrupt Pipes)

**Name:** USBHS_HSTPIPIMRx (INTPIPES)
**Offset:** 0x05C0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x3 in "Host Pipe x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTDT | PFREEZE | PDISHDMA |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCON | | NBUSYBKE | | | | |
| Access | | | | | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIE | RXSTALLDE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – RSTDT** Reset Data Toggle

| Value | Description |
|---|---|
| 0 | 0: No reset of the Data Toggle is ongoing. |
| 1 | Set when USBHS_HSTPIPIER.RSTDTS = 1. This resets the Data Toggle to its initial value for the current pipe. |

**Bit 17 – PFREEZE** Pipe Freeze
This freezes the pipe request generation.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.PFREEZEC = 1. This enables the pipe request generation. |
| 1 | Set when one of the following conditions is met:<br>• USBHS_HSTPIPIER.PFREEZES = 1<br>• The pipe is not configured.<br>• A STALL handshake has been received on the pipe.<br>• An error has occurred on the pipe (USBHS_HSTPIPISR.PERRI = 1).<br>• (INRQ+1) in requests have been processed.<br>• A Pipe Reset (USBHS_HSTPIP.PRSTx rising) has occurred.<br>• A Pipe Enable (USBHS_HSTPIP.PEN rising) has occurred. |

**Bit 16 – PDISHDMA** Pipe Interrupts Disable HDMA Request Enable
See the USBHS_DEVEPTIMR.EPDISHDMA bit description.

**Bit 14 – FIFOCON** FIFO Control
For OUT and SETUP pipes:

0: Cleared when USBHS_HSTPIPIDR.FIFOCONC = 1. This sends the FIFO data and switches the bank.
1: Set when the current bank is free, at the same time as USBHS_HSTPIPISR.TXOUTI or TXSTPI.
For IN pipes:
0: Cleared when USBHS_HSTPIPIDR.FIFOCONC = 1. This frees the current bank and switches to the next bank.
1: Set when a new IN message is stored in the current bank, at the same time as USBHS_HSTPIPISR.RXINI.

**Bit 12 – NBUSYBKE** Number of Busy Banks Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.NBUSYBKEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NBUSYBKE). |
| 1 | Set when USBHS_HSTPIPIER.NBUSYBKES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NBUSYBKE). |

**Bit 7 – SHORTPACKETIE** Short Packet Interrupt Enable
If this bit is set for non-control OUT pipes, a short packet transmission is guaranteed upon ending a DMA transfer, thus signaling an end of transfer, provided that the End of DMA Buffer Output Enable (USBHS_HSTDMACONTROL.END_B_EN) bit and the Automatic Switch (USBHS_HSTPIPCFG.AUTOSW) bit = 1.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.SHORTPACKETEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.SHORTPACKETE). |
| 1 | Set when USBHS_HSTPIPIER.SHORTPACKETIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.SHORTPACKETIE). |

**Bit 6 – RXSTALLDE** Received STALLed Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.RXSTALLDEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXSTALLDE). |
| 1 | Set when USBHS_HSTPIPIER.RXSTALLDES= 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXSTALLDE). |

**Bit 5 – OVERFIE** Overflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.OVERFIEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.OVERFIE). |
| 1 | Set when USBHS_HSTPIPIER.OVERFIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.OVERFIE). |

**Bit 4 – NAKEDE** NAKed Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.NAKEDEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NAKEDE). |
| 1 | Set when USBHS_HSTPIPIER.NAKEDES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NAKEDE). |

**Bit 3 – PERRE** Pipe Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.PERREC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.PERRE). |
| 1 | Set when USBHS_HSTPIPIER.PERRES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.PERRE). |

**Bit 2 – UNDERFIE** Underflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.UNDERFIEC= 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.UNDERFIE). |
| 1 | Set when USBHS_HSTPIPIER.UNDERFIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.UNDERFIE). |

**Bit 1 – TXOUTE** Transmitted OUT Data Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.TXOUTEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXOUTE). |
| 1 | Set when USBHS_HSTPIPIER.TXOUTES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXOUTE). |

**Bit 0 – RXINE** Received IN Data Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.RXINEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXINE). |
| 1 | Set when USBHS_HSTPIPIER.RXINES= 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXINE). |

### 39.6.56 Host Pipe x Mask Register (Isochronous Pipes)

**Name:** USBHS_HSTPIPIMRx (ISOPIPES)
**Offset:** 0x05C0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x1 in "Host Pipe x Configuration Register".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTDT | PFREEZE | PDISHDMA |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCON | | NBUSYBKE | | | | |
| Access | | | | | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIE | CRCERRE | OVERFIE | NAKEDE | PERRE | UNDERFIE | TXOUTE | RXINE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – RSTDT** Reset Data Toggle

| Value | Description |
|---|---|
| 0 | No reset of the Data Toggle is ongoing. |
| 1 | Set when USBHS_HSTPIPIER.RSTDTS = 1. This resets the Data Toggle to its initial value for the current pipe. |

**Bit 17 – PFREEZE** Pipe Freeze
This freezes the pipe request generation.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.PFREEZEC = 1. This enables the pipe request generation. |
| 1 | Set when one of the following conditions is met:<br>• USBHS_HSTPIPIER.PFREEZES = 1.<br>• The pipe is not configured.<br>• A STALL handshake has been received on the pipe.<br>• An error has occurred on the pipe (USBHS_HSTPIPISR.PERRI = 1).<br>• (INRQ+1) In requests have been processed.<br>• A Pipe Reset (USBHS_HSTPIP.PRSTx rising) has occurred.<br>• A Pipe Enable (USBHS_HSTPIP.PEN rising) has occurred. |

**Bit 16 – PDISHDMA** Pipe Interrupts Disable HDMA Request Enable
See the USBHS_DEVEPTIMR.EPDISHDMA bit description.

**Bit 14 – FIFOCON** FIFO Control
For OUT and SETUP pipes:

0: Cleared when USBHS_HSTPIPIDR.FIFOCONC = 1. This sends the FIFO data and switches the bank.
1: Set when the current bank is free, at the same time as USBHS_HSTPIPISR.TXOUTI or TXSTPI.
For IN pipes:
0: Cleared when USBHS_HSTPIPIDR.FIFOCONC = 1. This frees the current bank and switches to the next bank.
1: Set when a new IN message is stored in the current bank, at the same time as USBHS_HSTPIPISR.RXINI.

**Bit 12 – NBUSYBKE** Number of Busy Banks Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.NBUSYBKEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NBUSYBKE). |
| 1 | Set when USBHS_HSTPIPIER.NBUSYBKES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NBUSYBKE). |

**Bit 7 – SHORTPACKETIE** Short Packet Interrupt Enable
If this bit is set for non-control OUT pipes, a short packet transmission is guaranteed upon ending a DMA transfer, thus signaling an end of transfer, provided that the End of DMA Buffer Output Enable (USBHS_HSTDMACONTROL.END_B_EN) bit and the Automatic Switch (USBHS_HSTPIPCFG.AUTOSW) bit = 1.

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.SHORTPACKETEC = 1. This disables the Transmitted interrupt Data IT (USBHS_HSTPIPIMR.SHORTPACKETE). |
| 1 | Set when USBHS_HSTPIPIER.SHORTPACKETIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.SHORTPACKETIE). |

**Bit 6 – CRCERRE** CRC Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.CRCERREC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.CRCERRE). |
| 1 | Set when USBHS_HSTPIPIER.CRCERRES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.CRCERRE). |

**Bit 5 – OVERFIE** Overflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.OVERFIEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.OVERFIE). |
| 1 | Set when USBHS_HSTPIPIER.OVERFIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.OVERFIE). |

**Bit 4 – NAKEDE** NAKed Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.NAKEDEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NAKEDE). |
| 1 | Set when USBHS_HSTPIPIER.NAKEDES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.NAKEDE). |

**Bit 3 – PERRE** Pipe Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.PERREC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.PERRE). |
| 1 | Set when USBHS_HSTPIPIER.PERRES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.PERRE). |

**Bit 2 – UNDERFIE** Underflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.UNDERFIEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.UNDERFIE). |
| 1 | Set when USBHS_HSTPIPIER.UNDERFIES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.UNDERFIE). |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 1 – TXOUTE** Transmitted OUT Data Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.TXOUTEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXOUTE). |
| 1 | Set when USBHS_HSTPIPIER.TXOUTES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.TXOUTE). |

**Bit 0 – RXINE** Received IN Data Interrupt Enable

| Value | Description |
|---|---|
| 0 | Cleared when USBHS_HSTPIPIDR.RXINEC = 1. This disables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXINE). |
| 1 | Set when USBHS_HSTPIPIER.RXINES = 1. This enables the Transmitted IN Data interrupt (USBHS_HSTPIPIMR.RXINE). |

### 39.6.57 Host Pipe x Disable Register (Control, Bulk Pipes)

**Name:** USBHS_HSTPIPIDRx
**Offset:** 0x0620 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x0 or 0x2 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Mask Register (Control, Bulk Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_HSTPIPIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PFREEZEC | PDISHDMAC |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCONC | | NBUSYBKEC | | | | |
| Access | | | | | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC | TXOUTEC | RXINEC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – PFREEZEC** Pipe Freeze Disable

**Bit 16 – PDISHDMAC** Pipe Interrupts Disable HDMA Request Disable

**Bit 14 – FIFOCONC** FIFO Control Disable

**Bit 12 – NBUSYBKEC** Number of Busy Banks Disable

**Bit 7 – SHORTPACKETIEC** Short Packet Interrupt Disable

**Bit 6 – RXSTALLDEC** Received STALLed Interrupt Disable

**Bit 5 – OVERFIEC** Overflow Interrupt Disable

**Bit 4 – NAKEDEC** NAKed Interrupt Disable

**Bit 3 – PERREC** Pipe Error Interrupt Disable

**Bit 2 – TXSTPEC** Transmitted SETUP Interrupt Disable

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 1 – TXOUTEC** Transmitted OUT Data Interrupt Disable

**Bit 0 – RXINEC** Received IN Data Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.58 Host Pipe x Disable Register (Interrupt Pipes)

**Name:** USBHS_HSTPIPIDRx (INTPIPES)
**Offset:** 0x0620 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x3 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Mask Register (Interrupt Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_HSTPIPIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PFREEZEC | PDISHDMAC |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCONC | | NBUSYBKEC | | | | |
| Access | | | | | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIEC | RXSTALLDEC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – PFREEZEC** Pipe Freeze Disable

**Bit 16 – PDISHDMAC** Pipe Interrupts Disable HDMA Request Disable

**Bit 14 – FIFOCONC** FIFO Control Disable

**Bit 12 – NBUSYBKEC** Number of Busy Banks Disable

**Bit 7 – SHORTPACKETIEC** Short Packet Interrupt Disable

**Bit 6 – RXSTALLDEC** Received STALLed Interrupt Disable

**Bit 5 – OVERFIEC** Overflow Interrupt Disable

**Bit 4 – NAKEDEC** NAKed Interrupt Disable

**Bit 3 – PERREC** Pipe Error Interrupt Disable

**Bit 2 – UNDERFIEC** Underflow Interrupt Disable

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 1 – TXOUTEC** Transmitted OUT Data Interrupt Disable

**Bit 0 – RXINEC** Received IN Data Interrupt Disable

### 39.6.59 Host Pipe x Disable Register (Isochronous Pipes)

**Name:** USBHS_HSTPIPIDRx (ISOPIPES)
**Offset:** 0x0620 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x1 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Mask Register (Isochronous Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Clears the corresponding bit in USBHS_HSTPIPIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | PFREEZEC | PDISHDMAC |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FIFOCONC | | NBUSYBKEC | | | | |
| Access | | | | | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIEC | CRCERREC | OVERFIEC | NAKEDEC | PERREC | UNDERFIEC | TXOUTEC | RXINEC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – PFREEZEC** Pipe Freeze Disable

**Bit 16 – PDISHDMAC** Pipe Interrupts Disable HDMA Request Disable

**Bit 14 – FIFOCONC** FIFO Control Disable

**Bit 12 – NBUSYBKEC** Number of Busy Banks Disable

**Bit 7 – SHORTPACKETIEC** Short Packet Interrupt Disable

**Bit 6 – CRCERREC** CRC Error Interrupt Disable

**Bit 5 – OVERFIEC** Overflow Interrupt Disable

**Bit 4 – NAKEDEC** NAKed Interrupt Disable

**Bit 3 – PERREC** Pipe Error Interrupt Disable

**Bit 2 – UNDERFIEC** Underflow Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 1 – TXOUTEC** Transmitted OUT Data Interrupt Disable

**Bit 0 – RXINEC** Received IN Data Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.60 Host Pipe x Enable Register (Control, Bulk Pipes)

**Name:** USBHS_HSTPIPIERx
**Offset:** 0x05F0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x0 or 0x2 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Mask Register (Control, Bulk Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_HSTPIPIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | NBUSYBKES | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | TXSTPES | TXOUTES | RXINES |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – RSTDTS** Reset Data Toggle Enable

**Bit 17 – PFREEZES** Pipe Freeze Enable

**Bit 16 – PDISHDMAS** Pipe Interrupts Disable HDMA Request Enable

**Bit 12 – NBUSYBKES** Number of Busy Banks Enable

**Bit 7 – SHORTPACKETIES** Short Packet Interrupt Enable

**Bit 6 – RXSTALLDES** Received STALLed Interrupt Enable

**Bit 5 – OVERFIES** Overflow Interrupt Enable

**Bit 4 – NAKEDES** NAKed Interrupt Enable

**Bit 3 – PERRES** Pipe Error Interrupt Enable

**Bit 2 – TXSTPES** Transmitted SETUP Interrupt Enable

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 1 – TXOUTES** Transmitted OUT Data Interrupt Enable

**Bit 0 – RXINES** Received IN Data Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.61 Host Pipe x Enable Register (Interrupt Pipes)

**Name:** USBHS_HSTPIPIERx (INTPIPES)
**Offset:** 0x05F0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x3 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Mask Register (Interrupt Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_HSTPIPIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | NBUSYBKES | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIES | RXSTALLDES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – RSTDTS** Reset Data Toggle Enable

**Bit 17 – PFREEZES** Pipe Freeze Enable

**Bit 16 – PDISHDMAS** Pipe Interrupts Disable HDMA Request Enable

**Bit 12 – NBUSYBKES** Number of Busy Banks Enable

**Bit 7 – SHORTPACKETIES** Short Packet Interrupt Enable

**Bit 6 – RXSTALLDES** Received STALLed Interrupt Enable

**Bit 5 – OVERFIES** Overflow Interrupt Enable

**Bit 4 – NAKEDES** NAKed Interrupt Enable

**Bit 3 – PERRES** Pipe Error Interrupt Enable

**Bit 2 – UNDERFIES** Underflow Interrupt Enable

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 1 – TXOUTES** Transmitted OUT Data Interrupt Enable

**Bit 0 – RXINES** Received IN Data Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.62 Host Pipe x Enable Register (Isochronous Pipes)

**Name:** USBHS_HSTPIPIERx (ISOPIPES)
**Offset:** 0x05F0 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

This register view is relevant only if PTYPE = 0x1 in "Host Pipe x Configuration Register".

For additional information, see "Host Pipe x Mask Register (Isochronous Pipes)".

This register always reads as zero.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Sets the corresponding bit in USBHS_HSTPIPIMRx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RSTDTS | PFREEZES | PDISHDMAS |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | NBUSYBKES | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SHORTPACKETIES | CRCERRES | OVERFIES | NAKEDES | PERRES | UNDERFIES | TXOUTES | RXINES |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – RSTDTS** Reset Data Toggle Enable

**Bit 17 – PFREEZES** Pipe Freeze Enable

**Bit 16 – PDISHDMAS** Pipe Interrupts Disable HDMA Request Enable

**Bit 12 – NBUSYBKES** Number of Busy Banks Enable

**Bit 7 – SHORTPACKETIES** Short Packet Interrupt Enable

**Bit 6 – CRCERRES** CRC Error Interrupt Enable

**Bit 5 – OVERFIES** Overflow Interrupt Enable

**Bit 4 – NAKEDES** NAKed Interrupt Enable

**Bit 3 – PERRES** Pipe Error Interrupt Enable

**Bit 2 – UNDERFIES** Underflow Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 1 – TXOUTES** Transmitted OUT Data Interrupt Enable

**Bit 0 – RXINES** Received IN Data Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.63 Host Pipe x IN Request Register

**Name:** USBHS_HSTPIPINRQx
**Offset:** 0x0650 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | INMODE |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | INRQ[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 8 – INMODE** IN Request Mode

| Value | Description |
|---|---|
| 0 | Performs a pre-defined number of IN requests. This number is the INRQ field. |
| 1 | Enables the USBHS to perform infinite IN requests when the pipe is not frozen. |

**Bits 7:0 – INRQ[7:0]** IN Request Number before Freeze
This field contains the number of IN transactions before the USBHS freezes the pipe. The USBHS performs (INRQ+1) IN requests before freezing the pipe. This counter is automatically decreased by 1 each time an IN request has been successfully performed.
This register has no effect when INMODE = 1.

### 39.6.64 Host Pipe x Error Register

**Name:** USBHS_HSTPIPERRx
**Offset:** 0x0680 + x*0x04 [x=0..9]
**Reset:** 0
**Property:** Read/Write

Writing a zero in a bit/field in this register clears the bit/field. Writing a one has no effect.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | COUNTER[1:0] | | CRC16 | TIMEOUT | PID | DATAPID | DATATGL |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:5 – COUNTER[1:0]** Error Counter
This field is incremented each time an error occurs (CRC16, TIMEOUT, PID, DATAPID or DATATGL).
This field is cleared when receiving a USB packet free of error.
When this field reaches 3 (i.e., 3 consecutive errors), this pipe is automatically frozen (USBHS_HSTPIPIMRx.PFREEZE is set).

**Bit 4 – CRC16** CRC16 Error

| Value | Description |
|---|---|
| 0 | No CRC16 error occurred since last clear of this bit. |
| 1 | This bit is automatically set when a CRC16 error has been detected. |

**Bit 3 – TIMEOUT** Time-Out Error

| Value | Description |
|---|---|
| 0 | No Time-Out error occurred since last clear of this bit. |
| 1 | This bit is automatically set when a Time-Out error has been detected. |

**Bit 2 – PID** PID Error

| Value | Description |
|---|---|
| 0 | No PID error occurred since last clear of this bit. |
| 1 | This bit is automatically set when a PID error has been detected. |

**Bit 1 – DATAPID** Data PID Error

| Value | Description |
|---|---|
| 0 | No Data PID error occurred since last clear of this bit. |
| 1 | This bit is automatically set when a Data PID error has been detected. |

**Bit 0 – DATATGL** Data Toggle Error

© 2021 Microchip Technology Inc. and its subsidiaries

| Value | Description |
|---|---|
| 0 | No Data Toggle error occurred since last clear of this bit. |
| 1 | This bit is automatically set when a Data Toggle error has been detected. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.65 Host DMA Channel x Next Descriptor Address Register

**Name:** USBHS_HSTDMANXTDSCx
**Offset:** 0x0700 + (x-1)*0x10 [x=1..7]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NXT_DSC_ADD[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – NXT_DSC_ADD[31:0]** Next Descriptor Address
This field points to the next channel descriptor to be processed. This channel descriptor must be aligned, so bits 0 to 3 of the address must be equal to zero.

### 39.6.66 Host DMA Channel x Address Register

**Name:** USBHS_HSTDMAADDRESSx
**Offset:** 0x0704 + x*0x10 [x=0..6]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_ADD[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – BUFF_ADD[31:0]** Buffer Address
This field determines the AHB bus starting address of a DMA channel transfer.
Channel start and end addresses may be aligned on any byte boundary.
The firmware can write this field only when the USBHS_HSTDMASTATUS.CHANN_ENB bit is cleared.
This field is updated at the end of the address phase of the current access to the AHB bus. It is incremented by the access byte width. The access width is 4 bytes (or less) at packet start or end, if the start or end address is not aligned on a word boundary.
The packet start address is either the channel start address or the next channel address to be accessed in the channel buffer.
The packet end address is either the channel end address or the latest channel address accessed in the channel buffer.
The channel start address is written by software or loaded from the descriptor. The channel end address is either determined by the end of buffer or the USB device, or by the USB end of transfer if the USBHS_HSTDMACONTROLx.END_TR_EN bit is set.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 39.6.67 Host DMA Channel x Control Register

**Name:** USBHS_HSTDMACONTROLx
**Offset:** 0x0708 + x*0x10 [x=0..6]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_LENGTH[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_LENGTH[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BURST_LCK | DESC_LD_IT | END_BUFFIT | END_TR_IT | END_B_EN | END_TR_EN | LDNXT_DSC | CHANN_ENB |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – BUFF_LENGTH[15:0]** Buffer Byte Length (Write-only)
This field determines the number of bytes to be transferred until end of buffer. The maximum channel transfer size (32 KBytes) is reached when this field is 0 (default value). If the transfer size is unknown, this field should be set to 0, but the transfer end may occur earlier under USB device control.
When this field is written, the USBHS_HSTDMASTATUSx.BUFF_COUNT field is updated with the write value.
Notes: 1. Bits [31:2] are only writable when issuing a channel Control Command other than “Stop Now”.
2. For reliability, it is highly recommended to wait for both the USBHS_HSTDMASTATUSx.CHAN_ACT and the CHAN_ENB flags to be at 0, thus ensuring the channel has been stopped before issuing a command other than “Stop Now”.

**Bit 7 – BURST_LCK** Burst Lock Enable

| Value | Description |
|---|---|
| 0 | The DMA never locks the bus access. |
| 1 | USB packets AHB data bursts are locked for maximum optimization of the bus bandwidth usage and maximization of fly-by AHB burst duration. |

**Bit 6 – DESC_LD_IT** Descriptor Loaded Interrupt Enable

| Value | Description |
|---|---|
| 0 | USBHS_HSTDMASTATUSx.DESC_LDST rising does not trigger any interrupt. |
| 1 | An interrupt is generated when a descriptor has been loaded from the bus. |

**Bit 5 – END_BUFFIT** End of Buffer Interrupt Enable

| Value | Description |
|---|---|
| 0 | USBHS_HSTDMASTATUSx.END_BF_ST rising does not trigger any interrupt. |
| 1 | An interrupt is generated when USBHS_HSTDMASTATUSx.BUFF_COUNT reaches zero. |

**Bit 4 – END_TR_IT** End of Transfer Interrupt Enable
Use when the receive size is unknown.

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 0 | Completion of a USBHS device-initiated buffer transfer does not trigger any interrupt at USBHS_HSTDMASTATUSx.END_TR_ST rising. |
| 1 | An interrupt is sent after the buffer transfer is complete, if the USBHS device has ended the buffer transfer. |

**Bit 3 – END_B_EN** End of Buffer Enable Control
This is mainly for short packet OUT validations initiated by the DMA reaching the end of buffer, but could be used for IN packet truncation (discarding of unwanted packet data) at the end of DMA buffer.

| Value | Description |
|---|---|
| 0 | DMA Buffer End has no impact on USB packet transfer. |
| 1 | The pipe can validate the packet (according to the values programmed in the USBHS_HSTPIPCFGx.AUTOSW and USBHS_HSTPIPIMRx.SHORTPACKETIE fields) at DMA Buffer End, i.e., when USBHS_HSTDMASTATUS.BUFF_COUNT reaches 0. |

**Bit 2 – END_TR_EN** End of Transfer Enable Control (OUT transfers only)
When set, a BULK or INTERRUPT short packet closes the current buffer and the USBHS_HSTDMASTATUSx.END_TR_ST flag is raised.
This is intended for a USBHS non-prenegotiated USB transfer size.

| Value | Description |
|---|---|
| 0 | USB end of transfer is ignored. |
| 1 | The USBHS device can put an end to the current buffer transfer. |

**Bit 1 – LDNXT_DSC** Load Next Channel Transfer Descriptor Enable Command
If the CHANN_ENB bit is cleared, the next descriptor is loaded immediately upon transfer request.
DMA Channel Control Command Summary:

| Value LDNXT_DSC | Value CHANN_ENB | Name | Description |
|---|---|---|---|
| 0 | 0 | STOP_NOW | Stop now |
| 0 | 1 | RUN_AND_STOP | Run and stop at end of buffer |
| 1 | 0 | LOAD_NEXT_DESC | Load next descriptor now |
| 1 | 1 | RUN_AND_LINK | Run and link at end of buffer |

| Value | Description |
|---|---|
| 0 | No channel register is loaded after the end of the channel transfer. |
| 1 | The channel controller loads the next descriptor after the end of the current transfer, i.e., when the USBHS_HSTDMASTATUS.CHANN_ENB bit is reset. |

**Bit 0 – CHANN_ENB** Channel Enable Command
If the LDNXT_DSC bit has been cleared by descriptor loading, the firmware has to set the corresponding CHANN_ENB bit to start the described transfer, if needed.
If the LDNXT_DSC bit is cleared, the channel is frozen and the channel registers may then be read and/or written reliably as soon as both the USBHS_HSTDMASTATUS.CHANN_ENB and the CHANN_ACT flags read as 0.
If a channel request is currently serviced when this bit is cleared, the DMA FIFO buffer is drained until it is empty, then the USBHS_HSTDMASTATUS.CHANN_ENB bit is cleared.
If the LDNXT_DSC bit is set or after it has been cleared, the currently loaded descriptor is skipped (no data transfer occurs) and the next descriptor is immediately loaded.

| Value | Description |
|---|---|
| 0 | The DMA channel is disabled and no transfer occurs upon request. This bit is also cleared by hardware when the channel source bus is disabled at the end of the buffer. |
| 1 | The USBHS_HSTDMASTATUS.CHANN_ENB bit is set, enabling DMA channel data transfer. Then, any pending request starts the transfer. This may be used to start or resume any requested transfer. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 39.6.68 Host DMA Channel x Status Register

**Name:** USBHS_HSTDMASTATUSx
**Offset:** 0x070C + x*0x10 [x=0..6]
**Reset:** 0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_COUNT[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BUFF_COUNT[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | DESC_LDST | END_BF_ST | END_TR_ST | | | CHANN_ACT | CHANN_ENB |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

**Bits 31:16 – BUFF_COUNT[15:0]** Buffer Byte Count
This field determines the current number of bytes still to be transferred for this buffer.
This field is decremented from the AHB source bus access byte width at the end of this bus address phase.
The access byte width is 4 by default, or less, at DMA start or end, if the start or end address is not aligned on a word boundary.
At the end of buffer, the DMA accesses the USBHS device only for the number of bytes needed to complete it.
Note: For IN pipes, if the receive buffer byte length (USBHS_HSTDMACONTROL.BUFF_LENGTH) has been defaulted to zero because the USB transfer length is unknown, the actual buffer byte length received is 0x10000-BUFF_COUNT.

**Bit 6 – DESC_LDST** Descriptor Loaded Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when a descriptor has been loaded from the system bus. |

**Bit 5 – END_BF_ST** End of Channel Buffer Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when the BUFF_COUNT count-down reaches zero. |

**Bit 4 – END_TR_ST** End of Channel Transfer Status
Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

| Value | Description |
|---|---|
| 0 | Cleared automatically when read by software. |
| 1 | Set by hardware when the last packet transfer is complete, if the USBHS device has ended the transfer. |

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 1 – CHANN_ACT** Channel Active Status
When a packet transfer is ended, this bit is automatically reset.
When a packet transfer cannot be completed due to an END_BF_ST, this flag stays set during the next channel descriptor load (if any) and potentially until completion of a USBHS packet transfer, if allowed by the new descriptor.

| Value | Description |
|---|---|
| 0 | The DMA channel is no longer trying to source the packet data. |
| 1 | The DMA channel is currently trying to source packet data, i.e., selected as the highest-priority requesting channel. |

**Bit 0 – CHANN_ENB** Channel Enable Status
When any transfer is ended either due to an elapsed byte count or to completion of a USBHS device-initiated transfer, this bit is automatically reset.
This bit is normally set or cleared by writing into the USBHS_HSTDMACONTROLx.CHANN_ENB bit field either by software or descriptor loading.
If a channel request is currently serviced when the USBHS_HSTDMACONTROLx.CHANN_ENB bit is cleared, the DMA FIFO buffer is drained until it is empty, then this status bit is cleared.

| Value | Description |
|---|---|
| 0 | If cleared, the DMA channel no longer transfers data, and may load the next descriptor if the USBHS_HSTDMACONTROLx.LDNXT_DSC bit is set. |
| 1 | If set, the DMA channel is currently enabled and transfers data upon request. |

# 40. High-Speed Multimedia Card Interface (HSMCI)

## 40.1 Description

The High Speed Multimedia Card Interface (HSMCI) supports the MultiMedia Card (MMC) Specification V4.3, the SD Memory Card Specification V2.0, the SDIO V2.0 specification and CE-ATA V1.1.

The HSMCI includes a command register, response registers, data registers, timeout counters and error detection logic that automatically handle the transmission of commands and, when required, the reception of the associated responses and data with a limited processor overhead.

The HSMCI operates at a rate of up to Master Clock divided by 2 and supports the interfacing of 1 slot(s). Each slot may be used to interface with a High Speed MultiMedia Card bus (up to 30 Cards) or with an SD Memory Card. A bit field in the SD Card Register performs this selection.

The SD Memory Card communication is based on a 9-pin interface (clock, command, four data and three power lines) and the High Speed MultiMedia Card on a 7-pin interface (clock, command, one data, three power lines and one reserved for future use).

The SD Memory Card interface also supports High Speed MultiMedia Card operations. The main differences between SD and High Speed MultiMedia Cards are the initialization process and the bus topology.

HSMCI fully supports CE-ATA Revision 1.1, built on the MMC System Specification v4.0. The module includes dedicated hardware to issue the command completion signal and capture the host command completion signal disable.

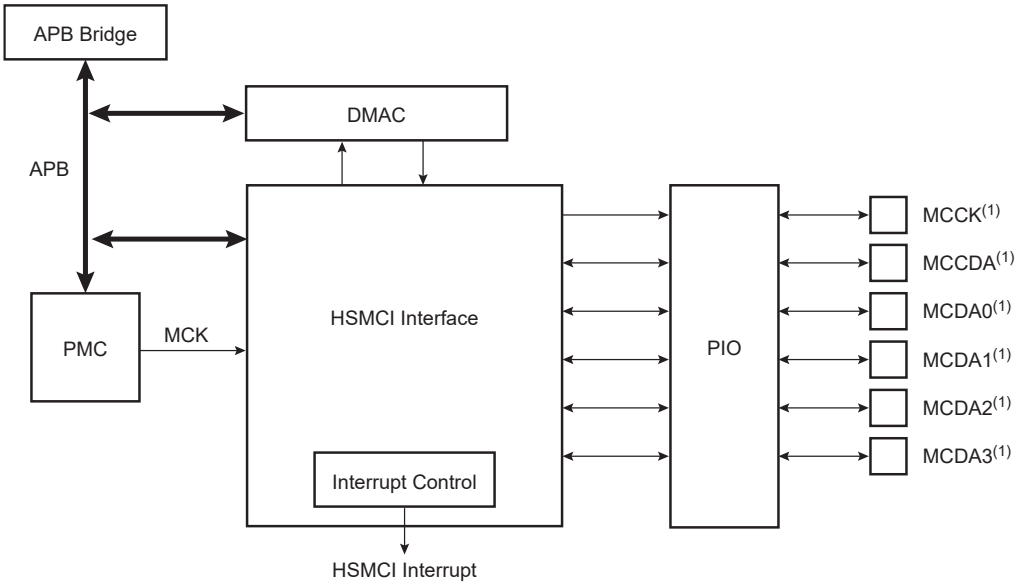
## 40.2 Embedded Characteristics

- Compatible with MultiMedia Card Specification Version 4.3
- Compatible with SD Memory Card Specification Version 2.0
- Compatible with SDIO Specification Version 2.0
- Compatible with CE-ATA Specification 1.1
- Cards Clock Rate Up to Master Clock Divided by 2
- Boot Operation Mode Support
- High Speed Mode Support
- Embedded Power Management to Slow Down Clock Rate When Not Used
- Supports 1 Multiplexed Slot(s)
  - Each Slot for either a High Speed MultiMedia Card Bus (Up to 30 Cards) or an SD Memory Card
- Support for Stream, Block and Multi-block Data Read and Write
- – Minimizes Processor Intervention for Large Buffer Transfers
- Built in FIFO (from 16 to 256 bytes) with Large Memory Aperture Supporting Incremental Access
- Support for CE-ATA Completion Signal Disable Command
- Protection Against Unexpected Modification On-the-Fly of the Configuration Registers

© 2021 Microchip Technology Inc.
and its subsidiaries

## 40.3 Block Diagram

**Figure 40-1. Block Diagram (4-bit configuration)**

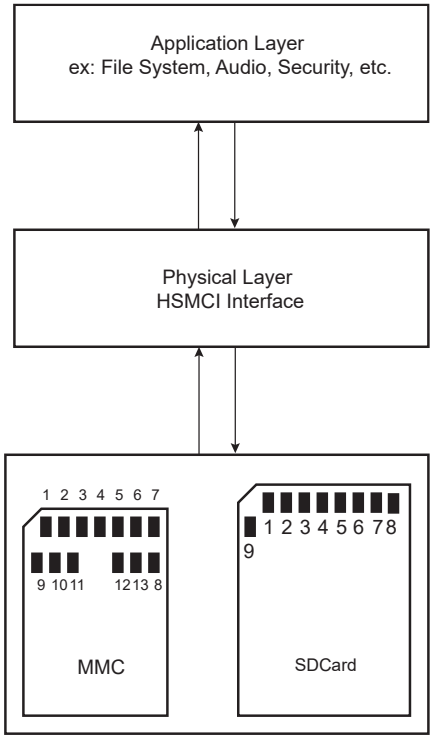



**Note:**

1. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy.

## 40.4 Application Block Diagram

**Figure 40-2. Application Block Diagram**



## 40.5 Pin Name List

**Table 40-1. I/O Lines Description for 4-bit Configuration**

| Pin Name[1] | Pin Description | Type[2] | Comments |
|---|---|---|---|
| MCCDA | Command/response | I/O/PP/OD | CMD of an MMC or SDCard/SDIO |
| MCCK | Clock | O | CLK of an MMC or SD Card/SDIO |
| MCDA0–MCDA3 | Data 0..3 of Slot A | I/O/PP | DAT[0..3] of an MMC<br>DAT[0..3] of an SD Card/SDIO |

Note: 1. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy.

Note: 2. I: Input, O: Output, PP: Push/Pull, OD: Open Drain.

## 40.6 Product Dependencies

### 40.6.1 I/O Lines

The pins used for interfacing the High Speed MultiMedia Cards or SD Cards are multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the peripheral functions to HSMCI pins.

### 40.6.2 Power Management

The HSMCI is clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the HSMCI clock.

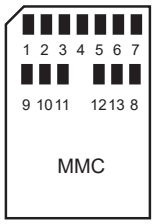
### 40.6.3 Interrupt Sources

The HSMCI has an interrupt line connected to the interrupt controller.

Handling the HSMCI interrupt requires programming the interrupt controller before configuring the HSMCI.

## 40.7 Bus Topology

**Figure 40-3. High Speed MultiMedia Memory Card Bus Topology**




The High Speed MultiMedia Card communication is based on a 13-pin serial bus interface. It has three communication lines and four supply lines.

**Table 40-2. Bus Topology**

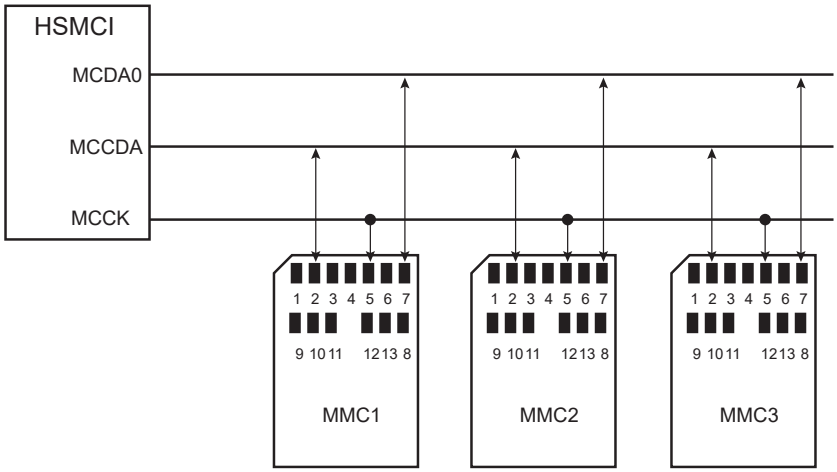
| Pin Number | Name | Type[1] | Description | HSMCI Pin Name[2] (Slot z) |
|---|---|---|---|---|
| 1 | DAT[3] | I/O/PP | Data | MCDz3 |
| 2 | CMD | I/O/PP/OD | Command/response | MCCDz |
| 3 | VSS1 | S | Supply voltage ground | VSS |

| .....continued | | | | |
|---|---|---|---|---|
| **Pin Number** | **Name** | **Type(1)** | **Description** | **HSMCI Pin Name(2) (Slot z)** |
| 4 | VDD | S | Supply voltage | VDD |
| 5 | CLK | O | Clock | MCCK |
| 6 | VSS2 | S | Supply voltage ground | VSS |
| 7 | DAT[0] | I/O/PP | Data 0 | MCDz0 |
| 8 | DAT[1] | I/O/PP | Data 1 | MCDz1 |
| 9 | DAT[2] | I/O/PP | Data 2 | MCDz2 |

**Notes:**

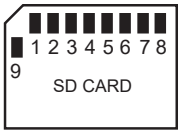
1. I: Input, O: Output, PP: Push/Pull, OD: Open Drain, S: Supply
2. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy.

**Figure 40-4. MMC Bus Connections (One Slot)**




**Note:** When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA MCDAy to HSMCIx_DAy.

**Figure 40-5. SD Memory Card Bus Topology**




The SD Memory Card bus includes the signals listed in the table below.

**Table 40-3. SD Memory Card Bus Signals**

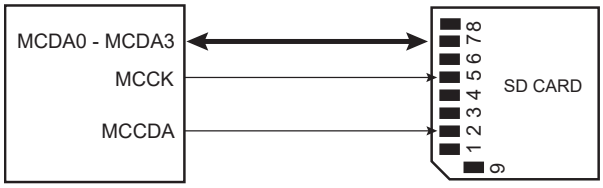
| Pin Number | Name | Type(1) | Description | HSMCI Pin Name(2) (Slot z) |
|---|---|---|---|---|
| 1 | CD/DAT[3] | I/O/PP | Card detect/ Data line Bit 3 | MCDz3 |
| 2 | CMD | PP | Command/response | MCCDz |
| 3 | VSS1 | S | Supply voltage ground | VSS |
| 4 | VDD | S | Supply voltage | VDD |
| 5 | CLK | O | Clock | MCCK |

| .....continued | | | | |
|---|---|---|---|---|
| **Pin Number** | **Name** | **Type(1)** | **Description** | **HSMCI Pin Name(2) (Slot z)** |
| 6 | VSS2 | S | Supply voltage ground | VSS |
| 7 | DAT[0] | I/O/PP | Data line Bit 0 | MCDz0 |
| 8 | DAT[1] | I/O/PP | Data line Bit 1 or Interrupt | MCDz1 |
| 9 | DAT[2] | I/O/PP | Data line Bit 2 | MCDz2 |

**Notes:**

1. I: input, O: output, PP: Push Pull, OD: Open Drain.
2. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy.

**Figure 40-6. SD Card Bus Connections with One Slot**




**Note:** When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA MCDAy to HSMCIx_DAy.

When the HSMCI is configured to operate with SD memory cards, the width of the data bus can be selected in the HSMCI_SDCR. Clearing the SDCBUS bit in this register means that the width is one bit; setting it means that the width is four bits. In the case of High Speed MultiMedia cards, only the data line 0 is used. The other data lines can be used as independent PIOs.

## 40.8 High-Speed Multimedia Card Operations

After a power-on reset, the cards are initialized by a special message-based High-Speed Multimedia Card bus protocol. Each message is represented by one of the following tokens:

- Command—A command is a token that starts an operation. A command is sent from the host either to a single card (addressed command) or to all connected cards (broadcast command). A command is transferred serially on the CMD line.
- Response—A response is a token which is sent from an addressed card or (synchronously) from all connected cards to the host as an answer to a previously received command. A response is transferred serially on the CMD line.
- Data—Data can be transferred from the card to the host or vice versa. Data is transferred via the data line.

Card addressing is implemented using a session address assigned during the initialization phase by the bus controller to all currently connected cards. Their unique CID number identifies individual cards.

The structure of commands, responses and data blocks is described in the High-Speed Multimedia Card System Specification. See Table 40-4 for additional information.

High-Speed Multimedia Card bus data transfers are composed of these tokens.

There are different types of operations. Addressed operations always contain a command and a response token. In addition, some operations have a data token; the others transfer their information directly within the command or response structure. In this case, no data token is present in an operation. The bits on the DAT and the CMD lines are transferred synchronous to the clock HSMCI clock.

Two types of data transfer commands are defined:

- Sequential commands—These commands initiate a continuous data stream. They are terminated only when a stop command follows on the CMD line. This mode reduces the command overhead to an absolute minimum.

- Block-oriented commands—These commands send a data block succeeded by CRC bits.

Both read and write operations allow either single or multiple block transmission. A multiple block transmission is terminated when a stop command follows on the CMD line similarly to the sequential read or when a multiple block transmission has a predefined block count (see “Data Transfer Operation”).

The HSMCI provides a set of registers to perform the entire range of High-Speed Multimedia Card operations.

### 40.8.1 Command - Response Operation

After reset, the HSMCI is disabled and becomes valid after setting the MCIEN bit in the HSMCI_CR.

The PWSEN bit saves power by dividing the HSMCI clock by $2^{PWSDIV}$ + 1 when the bus is inactive.

The two bits, RDPROOF and WRPROOF in the HSMCI Mode Register (HSMCI_MR) allow stopping the HSMCI clock during read or write access if the internal FIFO is full. This will guarantee data integrity, not bandwidth.

All the timings for High Speed MultiMedia Card are defined in the High Speed MultiMedia Card System Specification.

The two bus modes (open drain and push/pull) needed to process all the operations are defined in the HSMCI Command Register (HSMCI_CMDR). The HSMCI_CMDR allows a command to be carried out.

For example, to perform an ALL_SEND_CID command:

| | Host Command | | | | | $N_{ID}$ Cycles | | | Response | | | High Impedance State | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CMD | S | T | Content | CRC | E | Z | ****** | Z | S | T | CID Content | Z | Z | Z |

The command ALL_SEND_CID and the fields and values for the HSMCI_CMDR are described in the following two tables.

**Table 40-4. ALL_SEND_CID Command Description**

| CMD Index | Type | Argument | Response | Abbreviation | Command Description |
|---|---|---|---|---|---|
| CMD2 | bcr(1) | [31:0] stuff bits | R2 | ALL_SEND_CID | Asks all cards to send their CID numbers on the CMD line |

Note: 1. bcr means broadcast command with response.

**Table 40-5. Fields and Values for HSMCI_CMDR**

| Field | Value |
|---|---|
| CMDNB (command number) | 2 (CMD2) |
| RSPTYP (response type) | 2 (R2: 136 bits response) |
| SPCMD (special command) | 0 (not a special command) |
| OPCMD (open drain command) | 1 |
| MAXLAT (max latency for command to response) | 0 (NID cycles ==> 5 cycles) |
| TRCMD (transfer command) | 0 (No transfer) |
| TRDIR (transfer direction) | X (available only in transfer command) |
| TRTYP (transfer type) | X (available only in transfer command) |
| IOSPCMD (SDIO special command) | 0 (not a special command) |

The HSMCI_ARGR contains the argument field of the command.

To send a command, the user must perform the following steps:

1. Fill the argument register (HSMCI_ARGR) with the command argument.
2. Set the command register (HSMCI_CMDR).

The command is sent immediately after writing the command register.

© 2021 Microchip Technology Inc. and its subsidiaries

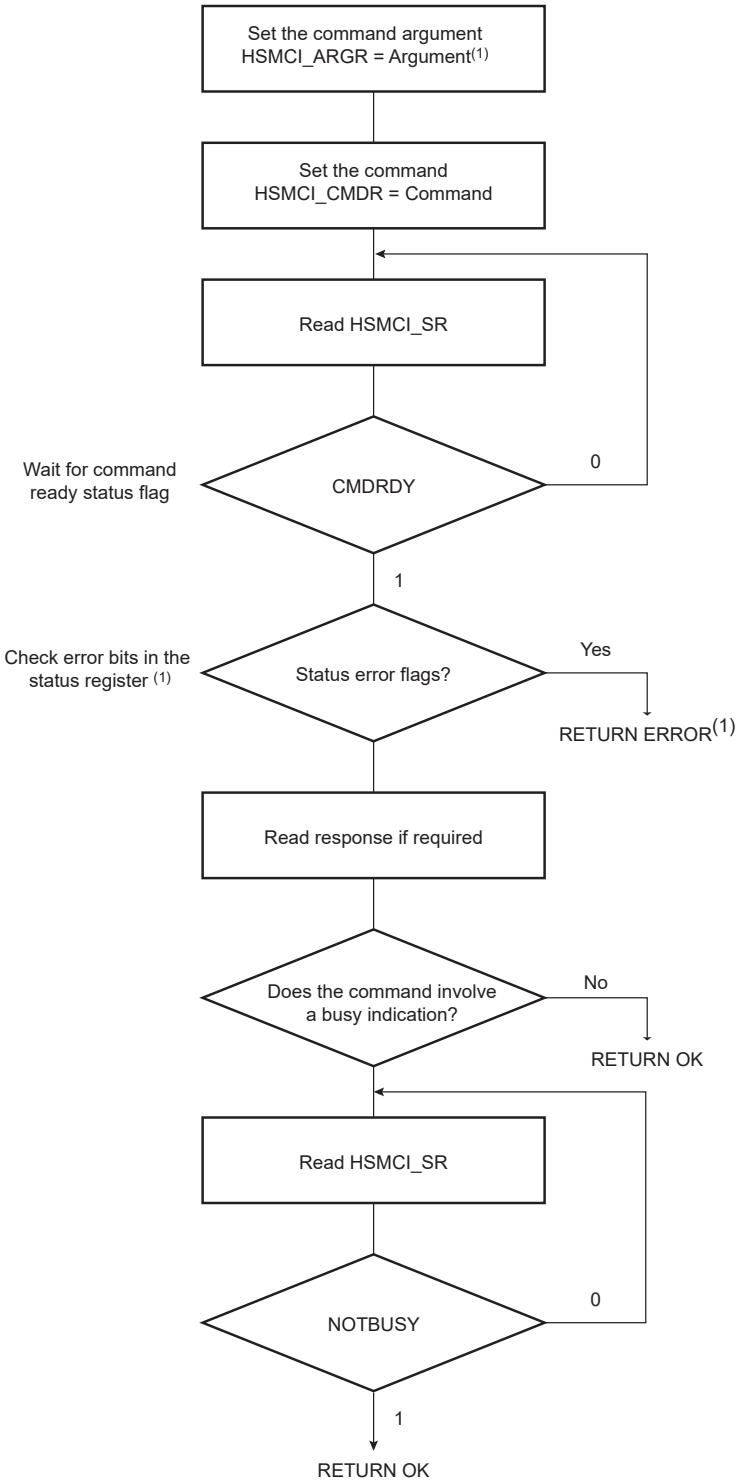
While the card maintains a busy indication (at the end of a STOP_TRANSMISSION command CMD12, for example), a new command shall not be sent. The NOTBUSY flag in the Status Register (HSMCI_SR) is asserted when the card releases the busy indication.

If the command requires a response, it can be read in the HSMCI Response Register (HSMCI_RSPR). The response size can be from 48 bits up to 136 bits depending on the command. The HSMCI embeds an error detection to prevent any corrupted data during the transfer.

The following flowchart shows how to send a command to the card and read the response if needed. In this example, the status register bits are polled but setting the appropriate bits in the HSMCI Interrupt Enable Register (HSMCI_IER) allows using an interrupt method.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 40-7. Command/Response Functional Flow Diagram**




Note: If the command is SEND_OP_COND, the CRC error flag is always present (refer to R3 response in the High Speed MultiMedia Card specification) .

### 40.8.2 Data Transfer Operation

The High Speed MultiMedia Card allows several read/write operations (single block, multiple blocks, stream, etc.). These kinds of transfer can be selected setting the Transfer Type (TRTYP) field in the HSMCI Command Register (HSMCI_CMDR).

In all cases, the block length (BLKLEN field) must be defined either in the HSMCI Mode Register (HSMCI_MR) or in the HSMCI Block Register (HSMCI_BLKR). This field determines the size of the data block.

Consequent to MMC Specification 3.1, two types of multiple block read (or write) transactions are defined (the host can use either one at any time):

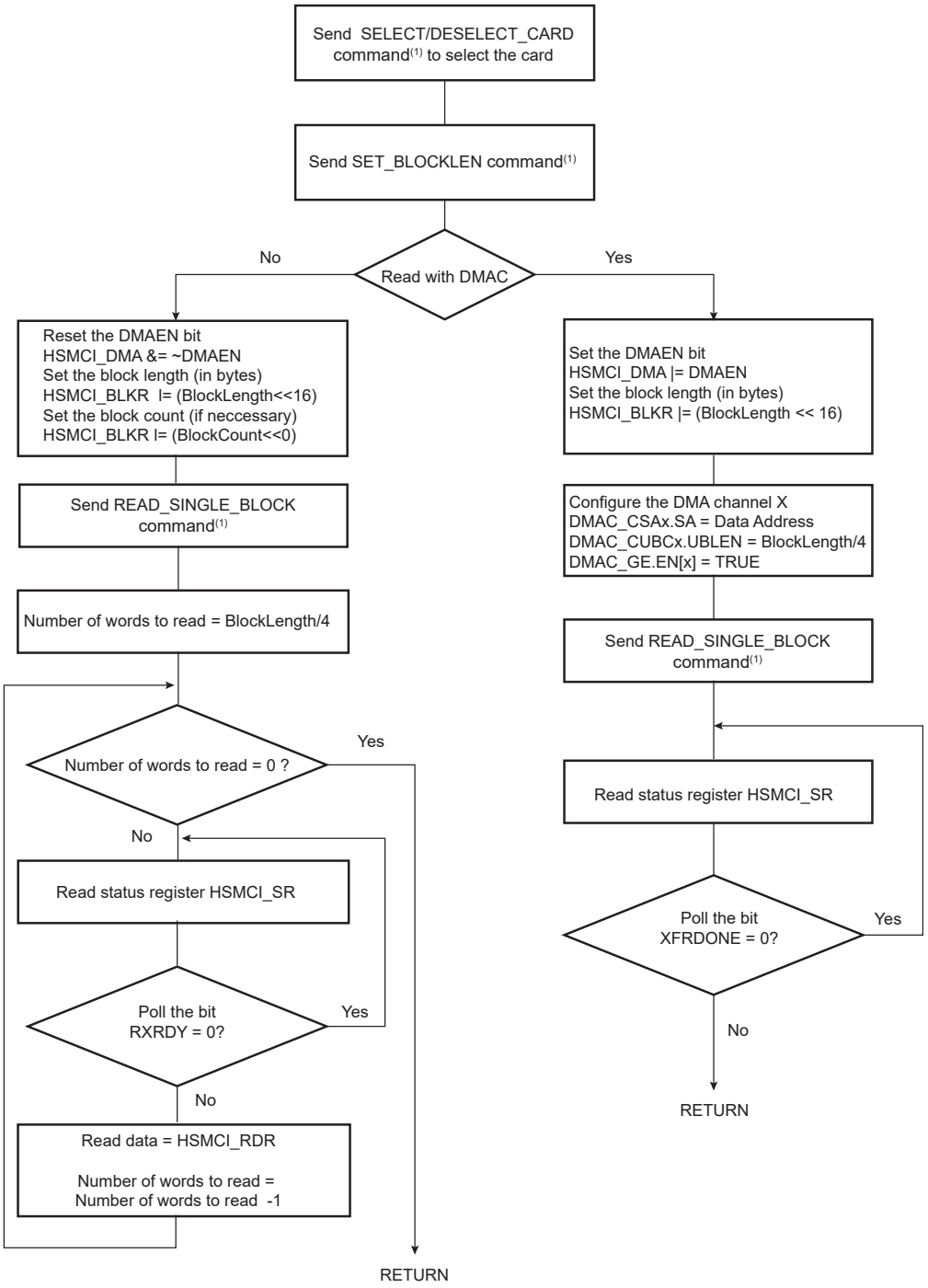
- Open-ended/Infinite Multiple block read (or write):
  The number of blocks for the read (or write) multiple block operation is not defined. The card will continuously transfer (or program) data blocks until a stop transmission command is received.
- Multiple block read (or write) with predefined block count (since version 3.1 and higher):
  The card will transfer (or program) the requested number of data blocks and terminate the transaction. The stop command is not required at the end of this type of multiple block read (or write), unless terminated with an error. In order to start a multiple block read (or write) with predefined block count, the host must correctly program the HSMCI Block Register (HSMCI_BLKR). Otherwise the card will start an open-ended multiple block read. The BCNT field of the HSMCI_BLKR defines the number of blocks to transfer (from 1 to 65535 blocks). Programming the value 0 in the BCNT field corresponds to an infinite block transfer.

### 40.8.3 Read Operation

The following flowchart shows how to read a single block with or without use of DMAC facilities. In this example, a polling method is used to wait for the end of read. Similarly, the user can configure the HSMCI Interrupt Enable Register (HSMCI_IER) to trigger an interrupt at the end of read.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 40-8. Read Functional Flow Diagram**




**Note 1:** It is assumed that this command has been correctly sent (see the Command/Response Functional Flow Diagram).

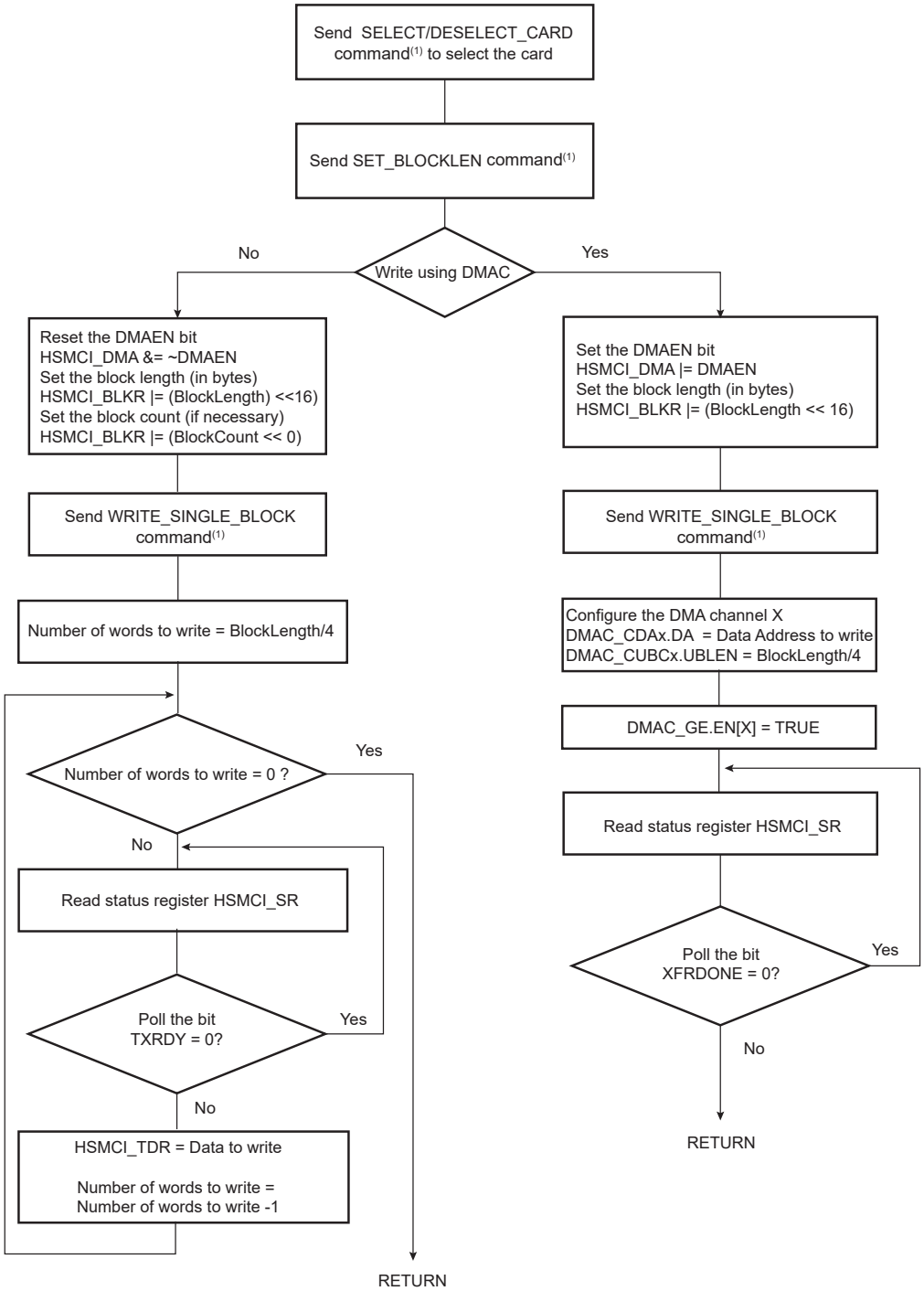
### 40.8.4 Write Operation

In write operation, the HSMCI Mode Register (HSMCI_MR) is used to define the padding value when writing non-multiple block size. If the bit PADV is 0, then 0x00 value is used when padding data, otherwise 0xFF is used.

If set, the bit DMAEN in the HSMCI DMA Condiguration Register (HSMCI_DMA) enables DMA transfer.

The flowchart, Write Functional Flow Diagram, shows how to write a single block with or without use of DMA facilities. Polling or interrupt method can be used to wait for the end of write according to the contents of the HSMCI Interrupt Mask Register (HSMCI_IMR).

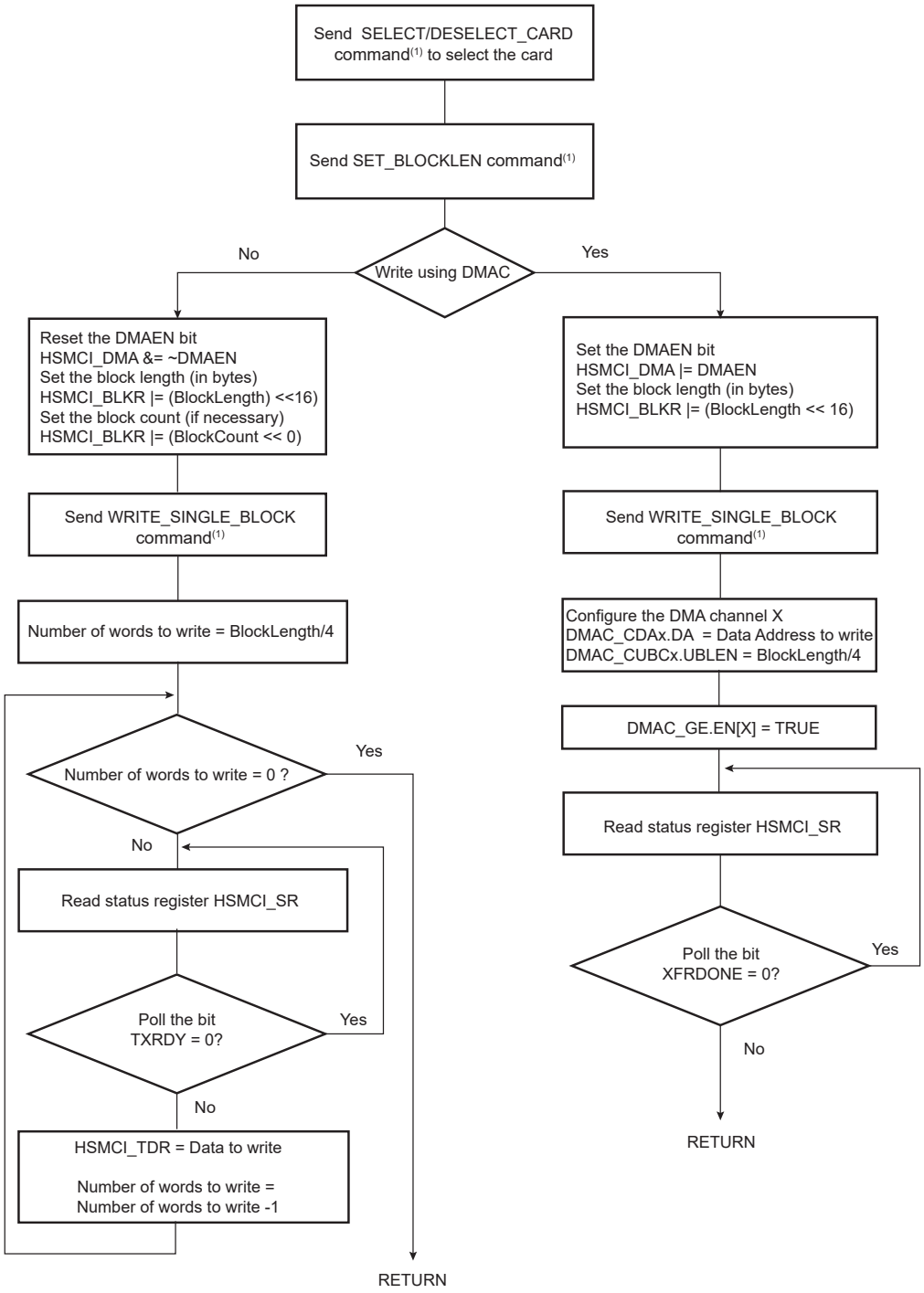
**Figure 40-9. Write Functional Flow Diagram**




Note: 1. It is assumed that this command has been correctly sent (see Command/Response Functional Flow Diagram).

The flowchart in Read and Write Multiple Block shows how to manage read multiple block and write multiple block transfers with the DMA Controller. Polling or interrupt method can be used to wait for the end of write according to the contents of the HSMCI_IMR.

**Figure 40-10. Read and Write Multiple Block**




Notes: 1. It is assumed that this command has been correctly sent (see Command/Response Functional Flow Diagram).

2. Handle errors reported in HSMCI_SR.

### 40.8.5 WRITE_SINGLE_BLOCK/WRITE_MULTIPLE_BLOCK Operation using DMA Controller

1. Wait until the current command execution has successfully terminated.
   a. Check that CMDRDY and NOTBUSY fields are asserted in HSMCI_SR
2. Program the block length in the card. This value defines the value block_length.

3. Program the block length in the HSMCI Configuration Register with block_length value.
4. Configure the fields of the HSMCI_MR as follows:
   a. Program FBYTE to one when the transfer is not multiple of 4, zero otherwise.
5. Issue a WRITE_SINGLE_BLOCK command writing HSMCI_ARGR then HSMCI_CMDR.
6. Program the DMA Controller.
   a. Read the Channel Status Register to choose an available (disabled) channel.

   b. Clear any pending interrupts on the channel from the previous DMAC transfer by reading the DMAC_CISx register.

   c. Program the channel registers.

   d. The DMAC_CSAx register for Channel x must be set to the location of the source data.

   e. The DMAC_CDAx register for Channel x must be set with the starting address of the HSMCI_FIFO address.

   f. Configure the fields of DMAC_CCx of Channel x as follows:

   – DWIDTH is set to WORD when the transfer is multiple of 4, otherwise it is set to BYTE

   – CSIZE must be set according to the value of HSMCI_DMA.CHKSIZE.

   g. Configure the fields of DMAC_CUBCx for Channel x as follows:

   – UBLEN is programmed with block_length/4 when the transfer length is multiple of 4, block_length otherwise.

   h. Enable Channel x, writing one to DMAC_GE.EN[x]. The DMAC is ready and waiting for request.
7. Wait for XFRDONE in the HSMCI_SR.

### 40.8.6 READ_SINGLE_BLOCK/READ_MULTIPLE_BLOCK Operation using DMA Controller

1. Wait until the current command execution has successfully completed.
   a. Check that CMDRDY and NOTBUSY are asserted in HSMCI_SR.
2. Program the block length in the card. This value defines the value block_length.
3. Program the block length in the HSMCI Configuration Register with block_length value.
4. Set RDPROOF bit in HSMCI_MR to avoid overflow.
5. Configure the fields of the HSMCI_MR as follows:
   a. Program FBYTE to one when the transfer is not multiple of 4, zero otherwise.
6. Issue a READ_SINGLE_BLOCK/WRITE_MULTIPLE_BLOCK command.
7. Program the DMA controller.
   a. Read the Channel Status Register to choose an available (disabled) channel.

   b. Clear any pending interrupts on the channel from the previous DMA transfer by reading the DMAC_CISx register.

   c. Program the channel registers.

   d. The DMAC_CSAx register for Channel x must be set with the starting address of the HSMCI_FIFO address.

   e. The DMAC_CDAx register for Channel x must be word aligned.

   f. Configure the fields of DMAC_CCx for Channel x as follows:

   – DWIDTH is set to WORD when the length is a multiple of 4, otherwise it is set to BYTE.

   – CSIZE must be set according to the value of HSMCI_DMA.CHKSIZE.

   g. Configure the fields of the DMAC_CUBCx register of Channel x as follows:

   – UBLEN is programmed with block_length/4 when the transfer length is multiple of 4, block_length otherwise.

   h. Enable Channel x, writing one to DMAC_GE.EN[x]. The DMAC is ready and waiting for request.
8. Wait for XFRDONE in the HSMCI_SR.

© 2021 Microchip Technology Inc.
and its subsidiaries

## 40.9 SD/SDIO Card Operation

The High Speed MultiMedia Card Interface allows processing of SD Memory (Secure Digital Memory Card) and SDIO (SD Input Output) Card commands.

SD/SDIO cards are based on the MultiMedia Card (MMC) format, but are physically slightly thicker and feature higher data transfer rates, a lock switch on the side to prevent accidental overwriting and security features. The physical form factor, pin assignment and data transfer protocol are forward-compatible with the High Speed MultiMedia Card with some additions. SD slots can actually be used for more than flash memory cards. Devices that support SDIO can use small devices designed for the SD form factor, such as GPS receivers, Wi-Fi or Bluetooth adapters, modems, barcode readers, IrDA adapters, FM radio tuners, RFID readers, digital cameras and more.

SD/SDIO is covered by numerous patents and trademarks, and licensing is only available through the Secure Digital Card Association.

The SD/SDIO Card communication is based on a 9-pin interface (Clock, Command, 4 x Data and 3 x Power lines). The communication protocol is defined as a part of this specification. The main difference between the SD/SDIO Card and the High Speed MultiMedia Card is the initialization process.

The SD/SDIO Card Register (HSMCI_SDCR) allows selection of the Card Slot and the data bus width.

The SD/SDIO Card bus allows dynamic configuration of the number of data lines. After power up, by default, the SD/SDIO Card uses only DAT0 for data transfer. After initialization, the host can change the bus width (number of active data lines).

### 40.9.1 SDIO Data Transfer Type

SDIO cards may transfer data in either a multi-byte (1 to 512 bytes) or an optional block format (1 to 511 blocks), while the SD memory cards are fixed in the block transfer mode. The TRTYP field in the HSMCI Command Register (HSMCI_CMDR) allows to choose between SDIO Byte or SDIO Block transfer.

The number of bytes/blocks to transfer is set through the BCNT field in the HSMCI Block Register (HSMCI_BLKR). In SDIO Block mode, the field BLKLEN must be set to the data block size while this field is not used in SDIO Byte mode.

An SDIO Card can have multiple I/O or combined I/O and memory (called Combo Card). Within a multi-function SDIO or a Combo card, there are multiple devices (I/O and memory) that share access to the SD bus. In order to allow the sharing of access to the host among multiple devices, SDIO and combo cards can implement the optional concept of suspend/resume (Refer to the SDIO Specification for more details). To send a suspend or a resume command, the host must set the SDIO Special Command field (IOSPCMD) in the HSMCI Command Register.

### 40.9.2 SDIO Interrupts

Each function within an SDIO or Combo card may implement interrupts (Refer to the SDIO Specification for more details). In order to allow the SDIO card to interrupt the host, an interrupt function is added to a pin on the DAT[1] line to signal the card's interrupt to the host. An SDIO interrupt on each slot can be enabled through the HSMCI Interrupt Enable Register. The SDIO interrupt is sampled regardless of the currently selected slot.

## 40.10 CE-ATA Operation

CE-ATA maps the streamlined ATA command set onto the MMC interface. The ATA task file is mapped onto MMC register space.

CE-ATA utilizes five MMC commands:

- GO_IDLE_STATE (CMD0): used for hard reset.
- STOP_TRANSMISSION (CMD12): causes the ATA command currently executing to be aborted.
- FAST_IO (CMD39): Used for single register access to the ATA taskfile registers, 8-bit access only.
- RW_MULTIPLE_REGISTERS (CMD60): used to issue an ATA command or to access the control/status registers.
- RW_MULTIPLE_BLOCK (CMD61): used to transfer data for an ATA command.

CE-ATA utilizes the same MMC command sequences for initialization as traditional MMC devices.

© 2021 Microchip Technology Inc. and its subsidiaries

### 40.10.1 Executing an ATA Polling Command

1. Issue READ_DMA_EXT with RW_MULTIPLE_REGISTER (CMD60) for 8 KB of DATA.
2. Read the ATA status register until DRQ is set.
3. Issue RW_MULTIPLE_BLOCK (CMD61) to transfer DATA.
4. Read the ATA status register until DRQ && BSY are configured to 0.

### 40.10.2 Executing an ATA Interrupt Command

1. Issue READ_DMA_EXT with RW_MULTIPLE_REGISTER (CMD60) for 8 KB of DATA with nIEN field set to zero to enable the command completion signal in the device.
2. Issue RW_MULTIPLE_BLOCK (CMD61) to transfer DATA.
3. Wait for Completion Signal Received Interrupt.

### 40.10.3 Aborting an ATA Command

If the host needs to abort an ATA command prior to the completion signal it must send a special command to avoid potential collision on the command line. The SPCMD field of the HSMCI_CMDR must be set to 3 to issue the CE-ATA completion Signal Disable Command.

### 40.10.4 CE-ATA Error Recovery

Several methods of ATA command failure may occur, including:

- No response to an MMC command, such as RW_MULTIPLE_REGISTER (CMD60).
- CRC is invalid for an MMC command or response.
- CRC16 is invalid for an MMC data packet.
- ATA Status register reflects an error by setting the ERR bit to one.
- The command completion signal does not arrive within a host specified time out period.

Error conditions are expected to happen infrequently. Thus, a robust error recovery mechanism may be used for each error event. The recommended error recovery procedure after a timeout is:

- Issue the command completion signal disable if nIEN was cleared to zero and the RW_MULTIPLE_BLOCK (CMD61) response has been received.
- Issue STOP_TRANSMISSION (CMD12) and successfully receive the R1 response.
- Issue a software reset to the CE-ATA device using FAST_IO (CMD39).

If STOP_TRANMISSION (CMD12) is successful, then the device is again ready for ATA commands. However, if the error recovery procedure does not work as expected or there is another timeout, the next step is to issue GO_IDLE_STATE (CMD0) to the device. GO_IDLE_STATE (CMD0) is a hard reset to the device and completely resets all device states.

Note that after issuing GO_IDLE_STATE (CMD0), all device initialization needs to be completed again. If the CE-ATA device completes all MMC commands correctly but fails the ATA command with the ERR bit set in the ATA Status register, no error recovery action is required. The ATA command itself failed implying that the device could not complete the action requested, however, there was no communication or protocol failure. After the device signals an error by setting the ERR bit to one in the ATA Status register, the host may attempt to retry the command.

## 40.11 HSMCI Boot Operation Mode

In boot operation mode, the processor can read boot data from the slave (MMC device) by keeping the CMD line low after power-on before issuing CMD1. The data can be read from either the boot area or user area, depending on register setting.

### 40.11.1 Boot Procedure, Processor Mode

1. Configure the HSMCI data bus width programming SDCBUS Field in the HSMCI_SDCR. The BOOT_BUS_WIDTH field located in the device Extended CSD register must be set accordingly.
2. Set the byte count to 512 bytes and the block count to the desired number of blocks, writing BLKLEN and BCNT fields of the HSMCI_BLKR.

3. Issue the Boot Operation Request command by writing to the HSMCI_CMDR with SPCMD field set to BOOTREQ, TRDIR set to READ and TRCMD set to “start data transfer”.
4. The BOOT_ACK field located in the HSMCI_CMDR must be set to one, if the BOOT_ACK field of the MMC device located in the Extended CSD register is set to one.
5. Host processor can copy boot data sequentially as soon as the RXRDY flag is asserted.
6. When Data transfer is completed, host processor shall terminate the boot stream by writing the HSMCI_CMDR with SPCMD field set to BOOTEND.

### 40.11.2 Boot Procedure DMA Mode

1. Configure the HSMCI data bus width by programming SDCBUS Field in the HSMCI_SDCR. The BOOT_BUS_WIDTH field in the device Extended CSD register must be set accordingly.
2. Set the byte count to 512 bytes and the block count to the desired number of blocks by writing BLKLEN and BCNT fields of the HSMCI_BLKR.
3. Enable DMA transfer in the HSMCI_DMA register.
4. Configure DMA controller, program the total amount of data to be transferred and enable the relevant channel.
5. Issue the Boot Operation Request command by writing to the HSMCI_CMDR with SPCND set to BOOTREQ, TRDIR set to READ and TRCMD set to “start data transfer”.
6. DMA controller copies the boot partition to the memory.
7. When DMA transfer is completed, host processor shall terminate the boot stream by writing the HSMCI_CMDR with SPCMD field set to BOOTEND.

## 40.12 HSMCI Transfer Done Timings

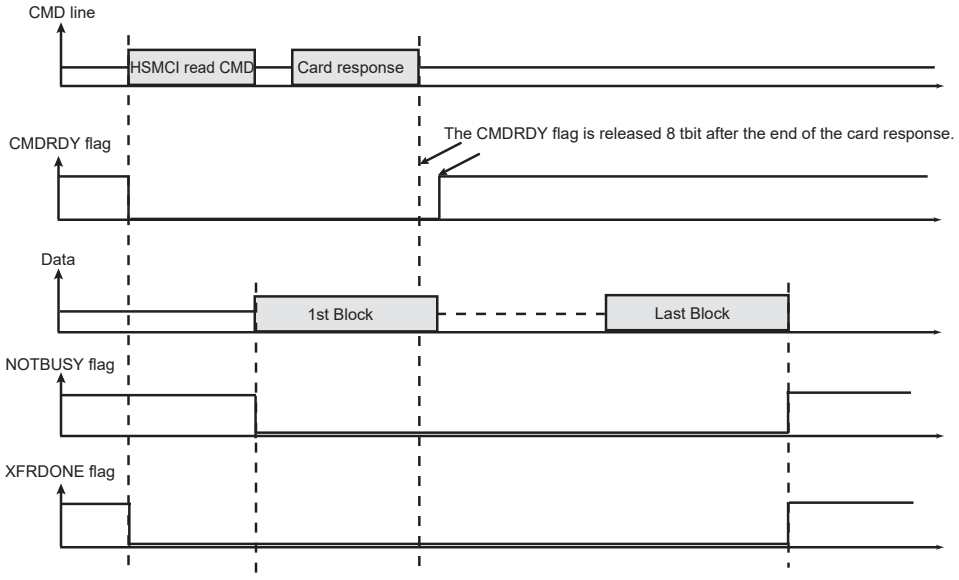
### 40.12.1 Definition

The XFRDONE flag in the HSMCI_SR indicates exactly when the read or write sequence is finished.

### 40.12.2 Read Access

During a read access, the XFRDONE flag behaves as shown in the following figure.

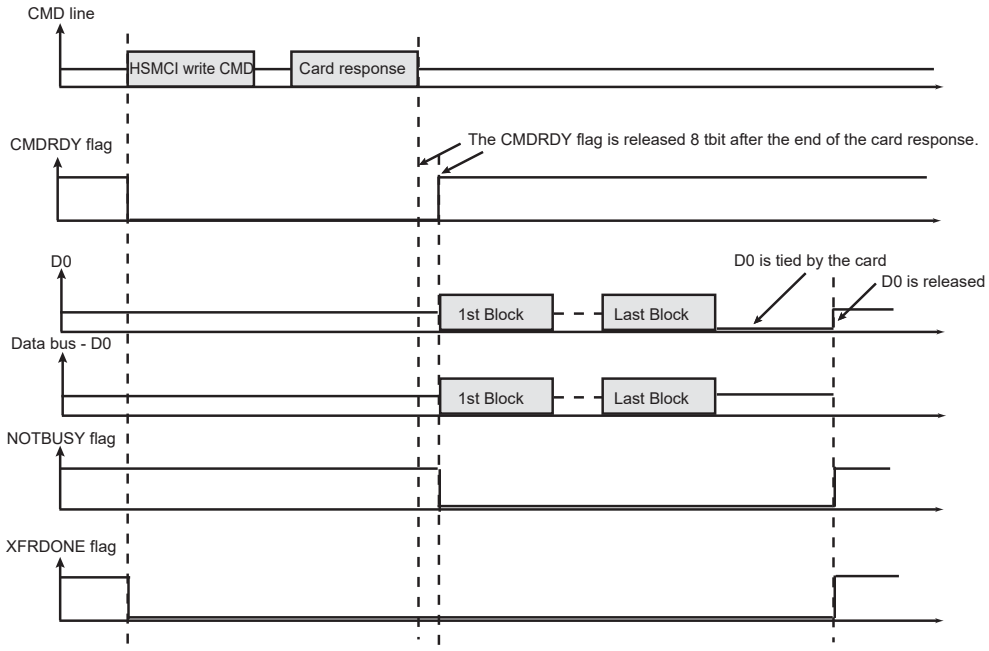
**Figure 40-11. XFRDONE During a Read Access**




### 40.12.3 Write Access

During a write access, the XFRDONE flag behaves as shown in the following figure.

**Figure 40-12. XFRDONE During a Write Access**




## 40.13 Register Write Protection

To prevent any single software error from corrupting HSMCI behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the HSMCI Write Protection Mode Register (HSMCI_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the HSMCI Write Protection Status Register (HSMCI_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the HSMCI_WPSR.

The following registers can be protected:

- HSMCI Mode Register
- HSMCI Data Timeout Register
- HSMCI SDCard/SDIO Register
- HSMCI Completion Signal Timeout Register
- HSMCI DMA Configuration Register
- HSMCI Configuration Register

## 40.14 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | HSMCI_CR | 7:0 | SWRST | | | | PWSDIS | PWSEN | MCIDIS | MCIEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x00 | HSMCI_FIFOx [x=0..255] | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x04 | HSMCI_MR | 7:0 | CLKDIV[7:0] | | | | | | | |
| | | 15:8 | | PADV | FBYTE | WRPROOF | RDPROOF | PWSDIV[2:0] | | |
| | | 23:16 | | | | | | | | CLKODD |
| | | 31:24 | | | | | | | | |
| 0x08 | HSMCI_DTOR | 7:0 | | DTOMUL[2:0] | | | DTOCYC[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | HSMCI_SDCR | 7:0 | SDCBUS[1:0] | | | | | | SDCSEL[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | HSMCI_ARGR | 7:0 | ARG[7:0] | | | | | | | |
| | | 15:8 | ARG[15:8] | | | | | | | |
| | | 23:16 | ARG[23:16] | | | | | | | |
| | | 31:24 | ARG[31:24] | | | | | | | |
| 0x14 | HSMCI_CMDR | 7:0 | RSPTYP[1:0] | | CMDNB[5:0] | | | | | |
| | | 15:8 | | | | MAXLAT | OPDCMD | SPCMD[2:0] | | |
| | | 23:16 | | | TRTYP[2:0] | | | TRDIR | TRCMD[1:0] | |
| | | 31:24 | | | | | BOOT_ACK | ATACS | IOSPCMD[1:0] | |
| 0x18 | HSMCI_BLKR | 7:0 | BCNT[7:0] | | | | | | | |
| | | 15:8 | BCNT[15:8] | | | | | | | |
| | | 23:16 | BLKLEN[7:0] | | | | | | | |
| | | 31:24 | BLKLEN[15:8] | | | | | | | |
| 0x1C | HSMCI_CSTOR | 7:0 | | CSTOMUL[2:0] | | | CSTOCYC[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | HSMCI_RSPR | 7:0 | RSP[7:0] | | | | | | | |
| | | 15:8 | RSP[15:8] | | | | | | | |
| | | 23:16 | RSP[23:16] | | | | | | | |
| | | 31:24 | RSP[31:24] | | | | | | | |
| 0x24 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | HSMCI_RDR | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x34 | HSMCI_TDR | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0x38 ... 0x3F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x40 | HSMCI_SR | 7:0 | | | NOTBUSY | DTIP | BLKE | TXRDY | RXRDY | CMDRDY |
| | | 15:8 | | | CSRCV | SDIOWAIT | | | | SDIOIRQA |
| | | 23:16 | CSTOE | DTOE | DCRCE | RTOE | RENDE | RCRCE | RDIRE | RINDE |
| | | 31:24 | UNRE | OVRE | ACKRCVE | ACKRCV | XFRDONE | FIFOEMPTY | | BLKOVRE |
| 0x44 | HSMCI_IER | 7:0 | | | NOTBUSY | DTIP | BLKE | TXRDY | RXRDY | CMDRDY |
| | | 15:8 | | | CSRCV | SDIOWAIT | | | | SDIOIRQA |
| | | 23:16 | CSTOE | DTOE | DCRCE | RTOE | RENDE | RCRCE | RDIRE | RINDE |
| | | 31:24 | UNRE | OVRE | ACKRCVE | ACKRCV | XFRDONE | FIFOEMPTY | | BLKOVRE |
| 0x48 | HSMCI_IDR | 7:0 | | | NOTBUSY | DTIP | BLKE | TXRDY | RXRDY | CMDRDY |
| | | 15:8 | | | CSRCV | SDIOWAIT | | | | SDIOIRQA |
| | | 23:16 | CSTOE | DTOE | DCRCE | RTOE | RENDE | RCRCE | RDIRE | RINDE |
| | | 31:24 | UNRE | OVRE | ACKRCVE | ACKRCV | XFRDONE | FIFOEMPTY | | BLKOVRE |
| 0x4C | HSMCI_IMR | 7:0 | | | NOTBUSY | DTIP | BLKE | TXRDY | RXRDY | CMDRDY |
| | | 15:8 | | | CSRCV | SDIOWAIT | | | | SDIOIRQA |
| | | 23:16 | CSTOE | DTOE | DCRCE | RTOE | RENDE | RCRCE | RDIRE | RINDE |
| | | 31:24 | UNRE | OVRE | ACKRCVE | ACKRCV | XFRDONE | FIFOEMPTY | | BLKOVRE |
| 0x50 | HSMCI_DMA | 7:0 | | CHKSIZE[2:0] | | | | | | |
| | | 15:8 | | | | | | | | DMAEN |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x54 | HSMCI_CFG | 7:0 | | | | FERRCTRL | | | | FIFOMODE |
| | | 15:8 | | | | LSYNC | | | | HSMODE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x58 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | HSMCI_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | HSMCI_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 40.14.1 HSMCI Control Register

**Name:** HSMCI_CR
**Offset:** 0x00
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | SWRST | | | | PWSDIS | PWSEN | MCIDIS | MCIEN |

**Bit 7 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the HSMCI. A software triggered hardware reset of the HSMCI is performed. |

**Bit 3 – PWSDIS** Power Save Mode Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Power Saving Mode. |

**Bit 2 – PWSEN** Power Save Mode Enable

**WARNING** Before enabling this mode, the user must set a value different from 0 in the PWSDIV field of the HSMCI_MR.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Power Saving Mode if PWSDIS is 0. |

**Bit 1 – MCIDIS** Multi-Media Interface Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Multi-Media Interface. |

**Bit 0 – MCIEN** Multi-Media Interface Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Multi-Media Interface if MCDIS is 0. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 40.14.2 HSMCI Mode Register

**Name:** HSMCI_MR
**Offset:** 0x04
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the HSMCI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | CLKODD |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | PADV | FBYTE | WRPROOF | RDPROOF | PWSDIV[2:0] | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | CLKDIV[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 16 – CLKODD** Clock divider is odd
This bit is the least significant bit of the clock divider and indicates the clock divider parity.

**Bit 14 – PADV** Padding Value
PADV may be only in manual transfer.

| Value | Description |
|---|---|
| 0 | 0x00 value is used when padding data in write transfer. |
| 1 | 0xFF value is used when padding data in write transfer. |

**Bit 13 – FBYTE** Force Byte Transfer
Enabling Force Byte Transfer allow byte transfers, so that transfer of blocks with a size different from modulo 4 can be supported.

**WARNING** BLKLEN value depends on FBYTE.

| Value | Description |
|---|---|
| 0 | Disables Force Byte Transfer. |
| 1 | Enables Force Byte Transfer. |

**Bit 12 – WRPROOF** Write Proof Enable
Enabling Write Proof allows to stop the HSMCI Clock during write access if the internal FIFO is full. This will guarantee data integrity, not bandwidth.

| Value | Description |
|---|---|
| 0 | Disables Write Proof. |
| 1 | Enables Write Proof. |

**Bit 11 – RDPROOF** Read Proof Enable
Enabling Read Proof allows to stop the HSMCI Clock during read access if the internal FIFO is full. This will guarantee data integrity, not bandwidth.

| Value | Description |
|---|---|
| 0 | Disables Read Proof. |
| 1 | Enables Read Proof. |

**Bits 10:8 – PWSDIV[2:0]** Power Saving Divider
High Speed MultiMedia Card Interface clock is divided by $2^{(PWSDIV)}$ + 1 when entering Power Saving Mode.

**WARNING** This value must be different from 0 before enabling the Power Save Mode in the HSMCI_CR (PWSEN bit).

**Bits 7:0 – CLKDIV[7:0]** Clock Divider
High Speed MultiMedia Card Interface clock (MCCK or HSMCI_CK) is Master Clock (MCK) divided by 2 × CLKDIV + CLKODD + 2.

### 40.14.3 HSMCI Data Timeout Register

**Name:** HSMCI_DTOR
**Offset:** 0x08
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the HSMCI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | DTOMUL[2:0] | | | DTOCYC[3:0] | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:4 – DTOMUL[2:0]** Data Timeout Multiplier
If the data time-out set by DTOCYC and DTOMUL has been exceeded, the Data Time-out Error flag (DTOE) in the HSMCI Status Register (HSMCI_SR) rises.

| Value | Name | Description |
|---|---|---|
| 0 | 1 | DTOCYC |
| 1 | 16 | DTOCYC x 16 |
| 2 | 128 | DTOCYC x 128 |
| 3 | 256 | DTOCYC x 256 |
| 4 | 1024 | DTOCYC x 1024 |
| 5 | 4096 | DTOCYC x 4096 |
| 6 | 65536 | DTOCYC x 65536 |
| 7 | 1048576 | DTOCYC x 1048576 |

**Bits 3:0 – DTOCYC[3:0]** Data Timeout Cycle Number
This field determines the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. It equals (DTOCYC x Multiplier).

© 2021 Microchip Technology Inc.
and its subsidiaries

### 40.14.4 HSMCI SDCard/SDIO Register

**Name:** HSMCI_SDCR
**Offset:** 0x0C
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the HSMCI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SDCBUS[1:0] | | | | | | SDCSEL[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | | | | | 0 | 0 |

**Bits 7:6 – SDCBUS[1:0]** SDCard/SDIO Bus Width

| Value | Name | Description |
|---|---|---|
| 0 | 1 | 1 bit |
| 1 | Reserved | |
| 2 | 4 | 4 bits |
| 3 | 8 | 8 bits |

**Bits 1:0 – SDCSEL[1:0]** SDCard/SDIO Slot

| Value | Name | Description |
|---|---|---|
| 0 | SLOTA | Slot A is selected. |
| 1 | SLOTB | Reserved |
| 2 | SLOTC | Reserved |
| 3 | SLOTD | Reserved |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 40.14.5 HSMCI Argument Register

**Name:** HSMCI_ARGR
**Offset:** 0x10
**Reset:** 0x0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ARG[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ARG[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ARG[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ARG[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ARG[31:0]** Command Argument

### 40.14.6 HSMCI Command Register

**Name:** HSMCI_CMDR
**Offset:** 0x14
**Property:** Write-only

This register is write-protected while CMDRDY is 0 in HSMCI_SR. If an Interrupt command is sent, this register is only writable by an interrupt response (field SPCMD). This means that the current command execution cannot be interrupted or modified.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | BOOT_ACK | ATACS | IOSPCMD[1:0] | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TRTYP[2:0] | | | TRDIR | TRCMD[1:0] | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | MAXLAT | OPDCMD | SPCMD[2:0] | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RSPTYP[1:0] | | CMDNB[5:0] | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 27 – BOOT_ACK** Boot Operation Acknowledge
The master can choose to receive the boot acknowledge from the slave when a Boot Request command is issued. When set to one this field indicates that a Boot acknowledge is expected within a programmable amount of time defined with DTOMUL and DTOCYC fields located in the HSMCI_DTOR. If the acknowledge pattern is not received then an acknowledge timeout error is raised. If the acknowledge pattern is corrupted then an acknowledge pattern error is set.

**Bit 26 – ATACS** ATA with Command Completion Signal
0 (NORMAL): Normal operation mode.
1 (COMPLETION): This bit indicates that a completion signal is expected within a programmed amount of time (HSMCI_CSTOR).

**Bits 25:24 – IOSPCMD[1:0]** SDIO Special Command

| Value | Name | Description |
|---|---|---|
| 0 | STD | Not an SDIO Special Command |
| 1 | SUSPEND | SDIO Suspend Command |
| 2 | RESUME | SDIO Resume Command |

**Bits 21:19 – TRTYP[2:0]** Transfer Type

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE | MMC/SD Card Single Block |
| 1 | MULTIPLE | MMC/SD Card Multiple Block |
| 2 | STREAM | MMC Stream |
| 4 | BYTE | SDIO Byte |
| 5 | BLOCK | SDIO Block |

**Bit 18 – TRDIR** Transfer Direction
0 (WRITE): Write.
1 (READ): Read.

**Bits 17:16 – TRCMD[1:0]** Transfer Command

| Value | Name | Description |
|---|---|---|
| 0 | NO_DATA | No data transfer |
| 1 | START_DATA | Start data transfer |
| 2 | STOP_DATA | Stop data transfer |
| 3 | Reserved | Reserved |

**Bit 12 – MAXLAT** Max Latency for Command to Response
0 (5): 5-cycle max latency.
1 (64): 64-cycle max latency.

**Bit 11 – OPDCMD** Open Drain Command
0 (PUSHPULL): Push pull command.
1 (OPENDRAIN): Open drain command.

**Bits 10:8 – SPCMD[2:0]** Special Command

| Value | Name | Description |
|---|---|---|
| 0 | STD | Not a special CMD. |
| 1 | INIT | Initialization CMD:<br>74 clock cycles for initialization sequence. |
| 2 | SYNC | Synchronized CMD:<br>Wait for the end of the current data block transfer before sending the pending command. |
| 3 | CE_ATA | CE-ATA Completion Signal disable Command.<br>The host cancels the ability for the device to return a command completion signal on the command line. |
| 4 | IT_CMD | Interrupt command:<br>Corresponds to the Interrupt Mode (CMD40). |
| 5 | IT_RESP | Interrupt response:<br>Corresponds to the Interrupt Mode (CMD40). |
| 6 | BOR | Boot Operation Request.<br>Start a boot operation mode, the host processor can read boot data from the MMC device directly. |
| 7 | EBO | End Boot Operation.<br>This command allows the host processor to terminate the boot operation mode. |

**Bits 7:6 – RSPTYP[1:0]** Response Type

| Value | Name | Description |
|---|---|---|
| 0 | NORESP | No response |
| 1 | 48_BIT | 48-bit response |
| 2 | 136_BIT | 136-bit response |
| 3 | R1B | R1b response type |

**Bits 5:0 – CMDNB[5:0]** Command Number
This is the command index.

### 40.14.7 HSMCI Block Register

**Name:** HSMCI_BLKR
**Offset:** 0x18
**Reset:** 0x0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BLKLEN[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BLKLEN[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BCNT[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BCNT[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – BLKLEN[15:0]** Data Block Length
This field determines the size of the data block.
Bits 16 and 17 must be configured to 0 if FBYTE is disabled.
Note: In SDIO Byte mode, BLKLEN field is not used.

**Bits 15:0 – BCNT[15:0]** MMC/SDIO Block Count - SDIO Byte Count
This field determines the number of data byte(s) or block(s) to transfer.
The transfer data type and the authorized values for BCNT field are determined by the TRTYP field in the HSMCI Command Register (HSMCI_CMDR).
When TRTYP = 1 (MMC/SDCARD Multiple Block), BCNT can be programmed from 1 to 65535, 0 corresponds to an infinite block transfer.
When TRTYP = 4 (SDIO Byte), BCNT can be programmed from 1 to 511, 0 corresponds to 512-byte transfer. Values in range 512 to 65536 are forbidden.
When TRTYP = 5 (SDIO Block), BCNT can be programmed from 1 to 511, 0 corresponds to an infinite block transfer. Values in range 512 to 65536 are forbidden.

**WARNING** In SDIO Byte and Block modes (TRTYP = 4 or 5), writing the 7 last bits of BCNT field with a value which differs from 0 is forbidden and may lead to unpredictable results.

### 40.14.8 HSMCI Completion Signal Timeout Register

**Name:** HSMCI_CSTOR
**Offset:** 0x1C
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the HSMCI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | CSTOMUL[2:0] | | | CSTOCYC[3:0] | | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 6:4 – CSTOMUL[2:0]** Completion Signal Timeout Multiplier
This field determines the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. Its value is calculated by (CSTOCYC x Multiplier).
These fields determine the maximum number of Master Clock cycles that the HSMCI waits between the end of the data transfer and the assertion of the completion signal. The data transfer comprises data phase and the optional busy phase. If a non-DATA ATA command is issued, the HSMCI starts waiting immediately after the end of the response until the completion signal.
Multiplier is defined by CSTOMUL as shown in the following table:
If the data time-out set by CSTOCYC and CSTOMUL has been exceeded, the Completion Signal Time-out Error flag (CSTOE) in the HSMCI Status Register (HSMCI_SR) rises.

| Value | Name | Description |
|---|---|---|
| 0 | 1 | CSTOCYC x 1 |
| 1 | 16 | CSTOCYC x 16 |
| 2 | 128 | CSTOCYC x 128 |
| 3 | 256 | CSTOCYC x 256 |
| 4 | 1024 | CSTOCYC x 1024 |
| 5 | 4096 | CSTOCYC x 4096 |
| 6 | 65536 | CSTOCYC x 65536 |
| 7 | 1048576 | CSTOCYC x 1048576 |

**Bits 3:0 – CSTOCYC[3:0]** Completion Signal Timeout Cycle Number
This field determines the maximum number of Master Clock cycles that the HSMCI waits between two data block transfers. Its value is calculated by (CSTOCYC x Multiplier).

### 40.14.9 HSMCI Response Register

**Name:** HSMCI_RSPR
**Offset:** 0x20
**Reset:** 0x0
**Property:** Read-only

Note: 1. The response register can be read by N accesses at the same HSMCI_RSPR or at consecutive addresses (0x20 to 0x2C).
N depends on the size of the response.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RSP[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSP[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RSP[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RSP[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RSP[31:0]** Response

### 40.14.10 HSMCI Receive Data Register

**Name:** HSMCI_RDR
**Offset:** 0x30
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Data to Read

### 40.14.11 HSMCI Transmit Data Register

**Name:** HSMCI_TDR
**Offset:** 0x34
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 31:0 – DATA[31:0]** Data to Write

### 40.14.12 HSMCI Status Register

**Name:** HSMCI_SR
**Offset:** 0x40
**Reset:** 0xC0E5
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | ACKRCVE | ACKRCV | XFRDONE | FIFOEMPTY | | BLKOVRE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSTOE | DTOE | DCRCE | RTOE | RENDE | RCRCE | RDIRE | RINDE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CSRCV | SDIOWAIT | | | | SDIOIRQA |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | NOTBUSY | DTIP | BLKE | TXRDY | RXRDY | CMDRDY |
| Access | | | | | | | | |
| Reset | | | 1 | 0 | 0 | 1 | 0 | 1 |

**Bit 31 – UNRE** Underrun (if FERRCTRL = 1, cleared by writing in HSMCI_CMDR or cleared on read if FERRCTRL = 0)
If FERRCTRL = 1 in HSMCI_CFG, OVRE is cleared on read.
If FERRCTRL = 0 in HSMCI_CFG, OVRE is cleared by writing HSMCI_CMDR.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | At least one 8-bit data has been sent without valid information (not written). |

**Bit 30 – OVRE** Overrun (if FERRCTRL = 1, cleared by writing in HSMCI_CMDR or cleared on read if FERRCTRL = 0)
If FERRCTRL = 1 in HSMCI_CFG, OVRE is cleared on read.
If FERRCTRL = 0 in HSMCI_CFG, OVRE is cleared by writing HSMCI_CMDR.

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | At least one 8-bit received data has been lost (not read). |

**Bit 29 – ACKRCVE** Boot Operation Acknowledge Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No boot operation error since the last read of HSMCI_SR |
| 1 | Corrupted Boot Acknowledge signal received since the last read of HSMCI_SR. |

**Bit 28 – ACKRCV** Boot Operation Acknowledge Received (cleared on read)

| Value | Description |
|---|---|
| 0 | No Boot acknowledge received since the last read of the HSMCI_SR. |
| 1 | A Boot acknowledge signal has been received since the last read of HSMCI_SR. |

**Bit 27 – XFRDONE** Transfer Done flag

| Value | Description |
|---|---|
| 0 | A transfer is in progress. |

| Value | Description |
|---|---|
| 1 | Command Register is ready to operate and the data bus is in the idle state. |

**Bit 26 – FIFOEMPTY** FIFO empty flag

| Value | Description |
|---|---|
| 0 | FIFO contains at least one byte. |
| 1 | FIFO is empty. |

**Bit 24 – BLKOVRE** DMA Block Overrun Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | A new block of data is received and the DMA controller has not started to move the current pending block, a block overrun is raised. |

**Bit 23 – CSTOE** Completion Signal Time-out Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | The completion signal time-out set by CSTOCYC and CSTOMUL in HSMCI_CSTOR has been exceeded. |

**Bit 22 – DTOE** Data Time-out Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | The data time-out set by DTOCYC and DTOMUL in HSMCI_DTOR has been exceeded. |

**Bit 21 – DCRCE** Data CRC Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | A CRC16 error has been detected in the last data block. |

**Bit 20 – RTOE** Response Time-out Error (cleared by writing in HSMCI_CMDR)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | The response time-out set by MAXLAT in the HSMCI_CMDR has been exceeded. |

**Bit 19 – RENDE** Response End Bit Error (cleared by writing in HSMCI_CMDR)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | The end bit of the response has not been detected. |

**Bit 18 – RCRCE** Response CRC Error (cleared by writing in HSMCI_CMDR)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | A CRC7 error has been detected in the response. |

**Bit 17 – RDIRE** Response Direction Error (cleared by writing in HSMCI_CMDR)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | The direction bit from card to host in the response has not been detected. |

**Bit 16 – RINDE** Response Index Error (cleared by writing in HSMCI_CMDR)

| Value | Description |
|---|---|
| 0 | No error. |
| 1 | A mismatch is detected between the command index sent and the response index received. |

**Bit 13 – CSRCV** CE-ATA Completion Signal Received (cleared on read)

| Value | Description |
|---|---|
| 0 | No completion signal received since last status read operation. |
| 1 | The device has issued a command completion signal on the command line. |

**Bit 12 – SDIOWAIT** SDIO Read Wait Operation Status

| Value | Description |
|---|---|
| 0 | Normal Bus operation. |
| 1 | The data bus has entered IO wait state. |

**Bit 8 – SDIOIRQA** SDIO Interrupt for Slot A (cleared on read)

| Value | Description |
|---|---|
| 0 | No interrupt detected on SDIO Slot A. |
| 1 | An SDIO Interrupt on Slot A occurred. |

**Bit 5 – NOTBUSY** HSMCI Not Busy

A block write operation uses a simple busy signalling of the write operation duration on the data (DAT0) line: during a data transfer block, if the card does not have a free data receive buffer, the card indicates this condition by pulling down the data line (DAT0) to LOW. The card stops pulling down the data line as soon as at least one receive buffer for the defined data transfer block length becomes free.
Refer to the MMC or SD Specification for more details concerning the busy behavior.
For all the read operations, the NOTBUSY flag is cleared at the end of the host command.
For the Infinite Read Multiple Blocks, the NOTBUSY flag is set at the end of the STOP_TRANSMISSION host command (CMD12).
For the Single Block Reads, the NOTBUSY flag is set at the end of the data read block.
For the Multiple Block Reads with predefined block count, the NOTBUSY flag is set at the end of the last received data block.
The NOTBUSY flag allows to deal with these different states.

| Value | Description |
|---|---|
| 0 | The HSMCI is not ready for new data transfer. Cleared at the end of the card response. |
| 1 | The HSMCI is ready for new data transfer. Set when the busy state on the data line has ended. This corresponds to a free internal data receive buffer of the card. |

**Bit 4 – DTIP** Data Transfer in Progress (cleared at the end of CRC16 calculation)

| Value | Description |
|---|---|
| 0 | No data transfer in progress. |
| 1 | The current data transfer is still in progress, including CRC16 calculation. |

**Bit 3 – BLKE** Data Block Ended (cleared on read)

This flag must be used only for Write Operations.
Refer to the MMC or SD Specification for more details concerning the CRC Status.

| Value | Description |
|---|---|
| 0 | A data block transfer is not yet finished. |
| 1 | A data block transfer has ended, including the CRC16 Status transmission. The flag is set for each transmitted CRC Status. |

**Bit 2 – TXRDY** Transmit Ready (cleared by writing in HSMCI_TDR)

| Value | Description |
|---|---|
| 0 | The last data written in HSMCI_TDR has not yet been transferred in the Shift Register. |
| 1 | The last data written in HSMCI_TDR has been transferred in the Shift Register. |

**Bit 1 – RXRDY** Receiver Ready (cleared by reading HSMCI_RDR)

| Value | Description |
|---|---|
| 0 | Data has not yet been received since the last read of HSMCI_RDR. |
| 1 | Data has been received since the last read of HSMCI_RDR. |

**Bit 0 – CMDRDY** Command Ready (cleared by writing in HSMCI_CMDR)

| Value | Description |
|---|---|
| 0 | A command is in progress. |
| 1 | The last command has been sent. |

### 40.14.13 HSMCI Interrupt Enable Register

**Name:** HSMCI_IER
**Offset:** 0x44
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | ACKRCVE | ACKRCV | XFRDONE | FIFOEMPTY | | BLKOVRE |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSTOE | DTOE | DCRCE | RTOE | RENDE | RCRCE | RDIRE | RINDE |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CSRCV | SDIOWAIT | | | | SDIOIRQA |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | NOTBUSY | DTIP | BLKE | TXRDY | RXRDY | CMDRDY |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – UNRE** Underrun Interrupt Enable

**Bit 30 – OVRE** Overrun Interrupt Enable

**Bit 29 – ACKRCVE** Boot Acknowledge Error Interrupt Enable

**Bit 28 – ACKRCV** Boot Acknowledge Interrupt Enable

**Bit 27 – XFRDONE** Transfer Done Interrupt enable

**Bit 26 – FIFOEMPTY** FIFO empty Interrupt enable

**Bit 24 – BLKOVRE** DMA Block Overrun Error Interrupt Enable

**Bit 23 – CSTOE** Completion Signal Timeout Error Interrupt Enable

**Bit 22 – DTOE** Data Time-out Error Interrupt Enable

**Bit 21 – DCRCE** Data CRC Error Interrupt Enable

**Bit 20 – RTOE** Response Time-out Error Interrupt Enable

**Bit 19 – RENDE** Response End Bit Error Interrupt Enable

**Bit 18 – RCRCE** Response CRC Error Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 17 – RDIRE** Response Direction Error Interrupt Enable

**Bit 16 – RINDE** Response Index Error Interrupt Enable

**Bit 13 – CSRCV** Completion Signal Received Interrupt Enable

**Bit 12 – SDIOWAIT** SDIO Read Wait Operation Status Interrupt Enable

**Bit 8 – SDIOIRQA** SDIO Interrupt for Slot A Interrupt Enable

**Bit 5 – NOTBUSY** Data Not Busy Interrupt Enable

**Bit 4 – DTIP** Data Transfer in Progress Interrupt Enable

**Bit 3 – BLKE** Data Block Ended Interrupt Enable

**Bit 2 – TXRDY** Transmit Ready Interrupt Enable

**Bit 1 – RXRDY** Receiver Ready Interrupt Enable

**Bit 0 – CMDRDY** Command Ready Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 40.14.14 HSMCI Interrupt Disable Register

**Name:** HSMCI_IDR
**Offset:** 0x48
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | ACKRCVE | ACKRCV | XFRDONE | FIFOEMPTY | | BLKOVRE |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSTOE | DTOE | DCRCE | RTOE | RENDE | RCRCE | RDIRE | RINDE |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CSRCV | SDIOWAIT | | | | SDIOIRQA |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | NOTBUSY | DTIP | BLKE | TXRDY | RXRDY | CMDRDY |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – UNRE** Underrun Interrupt Disable

**Bit 30 – OVRE** Overrun Interrupt Disable

**Bit 29 – ACKRCVE** Boot Acknowledge Error Interrupt Disable

**Bit 28 – ACKRCV** Boot Acknowledge Interrupt Disable

**Bit 27 – XFRDONE** Transfer Done Interrupt Disable

**Bit 26 – FIFOEMPTY** FIFO empty Interrupt Disable

**Bit 24 – BLKOVRE** DMA Block Overrun Error Interrupt Disable

**Bit 23 – CSTOE** Completion Signal Time out Error Interrupt Disable

**Bit 22 – DTOE** Data Time-out Error Interrupt Disable

**Bit 21 – DCRCE** Data CRC Error Interrupt Disable

**Bit 20 – RTOE** Response Time-out Error Interrupt Disable

**Bit 19 – RENDE** Response End Bit Error Interrupt Disable

**Bit 18 – RCRCE** Response CRC Error Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 17 – RDIRE** Response Direction Error Interrupt Disable

**Bit 16 – RINDE** Response Index Error Interrupt Disable

**Bit 13 – CSRCV** Completion Signal received interrupt Disable

**Bit 12 – SDIOWAIT** SDIO Read Wait Operation Status Interrupt Disable

**Bit 8 – SDIOIRQA** SDIO Interrupt for Slot A Interrupt Disable

**Bit 5 – NOTBUSY** Data Not Busy Interrupt Disable

**Bit 4 – DTIP** Data Transfer in Progress Interrupt Disable

**Bit 3 – BLKE** Data Block Ended Interrupt Disable

**Bit 2 – TXRDY** Transmit Ready Interrupt Disable

**Bit 1 – RXRDY** Receiver Ready Interrupt Disable

**Bit 0 – CMDRDY** Command Ready Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 40.14.15 HSMCI Interrupt Mask Register

**Name:** HSMCI_IMR
**Offset:** 0x4C
**Reset:** 0x0
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | ACKRCVE | ACKRCV | XFRDONE | FIFOEMPTY | | BLKOVRE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSTOE | DTOE | DCRCE | RTOE | RENDE | RCRCE | RDIRE | RINDE |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CSRCV | SDIOWAIT | | | | SDIOIRQA |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | NOTBUSY | DTIP | BLKE | TXRDY | RXRDY | CMDRDY |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – UNRE** Underrun Interrupt Mask

**Bit 30 – OVRE** Overrun Interrupt Mask

**Bit 29 – ACKRCVE** Boot Operation Acknowledge Error Interrupt Mask

**Bit 28 – ACKRCV** Boot Operation Acknowledge Received Interrupt Mask

**Bit 27 – XFRDONE** Transfer Done Interrupt Mask

**Bit 26 – FIFOEMPTY** FIFO Empty Interrupt Mask

**Bit 24 – BLKOVRE** DMA Block Overrun Error Interrupt Mask

**Bit 23 – CSTOE** Completion Signal Time-out Error Interrupt Mask

**Bit 22 – DTOE** Data Time-out Error Interrupt Mask

**Bit 21 – DCRCE** Data CRC Error Interrupt Mask

**Bit 20 – RTOE** Response Time-out Error Interrupt Mask

**Bit 19 – RENDE** Response End Bit Error Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 18 – RCRCE** Response CRC Error Interrupt Mask

**Bit 17 – RDIRE** Response Direction Error Interrupt Mask

**Bit 16 – RINDE** Response Index Error Interrupt Mask

**Bit 13 – CSRCV** Completion Signal Received Interrupt Mask

**Bit 12 – SDIOWAIT** SDIO Read Wait Operation Status Interrupt Mask

**Bit 8 – SDIOIRQA** SDIO Interrupt for Slot A Interrupt Mask

**Bit 5 – NOTBUSY** Data Not Busy Interrupt Mask

**Bit 4 – DTIP** Data Transfer in Progress Interrupt Mask

**Bit 3 – BLKE** Data Block Ended Interrupt Mask

**Bit 2 – TXRDY** Transmit Ready Interrupt Mask

**Bit 1 – RXRDY** Receiver Ready Interrupt Mask

**Bit 0 – CMDRDY** Command Ready Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 40.14.16 HSMCI DMA Configuration Register

**Name:** HSMCI_DMA
**Offset:** 0x50
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the HSMCI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | DMAEN |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | CHKSIZE[2:0] | | | | | | |
| Reset | | 0 | 0 | 0 | | | | |

**Bit 8 – DMAEN** DMA Hardware Handshaking Enable

| Value | Description |
|---|---|
| 0 | DMA interface is disabled. |
| 1 | DMA Interface is enabled.<br>Note: To avoid unpredictable behavior, DMA hardware handshaking must be disabled when CPU transfers are performed. |

**Bits 324:4 – CHKSIZE[320:0]** DMA Channel Read and Write Chunk Size
The CHKSIZE field indicates the number of data available when the DMA chunk transfer request is asserted.

| Value | Name | Description |
|---|---|---|
| 0 | 1 | 1 data available |
| 1 | 2 | 2 data available |
| 2 | 4 | 4 data available |
| 3 | 8 | 8 data available |
| 4 | 16 | 16 data available |

### 40.14.17 HSMCI Configuration Register

**Name:** HSMCI_CFG
**Offset:** 0x54
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the HSMCI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | LSYNC | | | | HSMODE |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FERRCTRL | | | | FIFOMODE |
| Access | | | | | | | | |
| Reset | | | | 0 | | | | 0 |

**Bit 12 – LSYNC** Synchronize on the last block

| Value | Description |
|---|---|
| 0 | The pending command is sent at the end of the current data block. |
| 1 | The pending command is sent at the end of the block transfer when the transfer length is not infinite (block count shall be different from zero). |

**Bit 8 – HSMODE** High Speed Mode

| Value | Description |
|---|---|
| 0 | Default bus timing mode. |
| 1 | If set to one, the host controller outputs command line and data lines on the rising edge of the card clock. The Host driver shall check the high speed support in the card registers. |

**Bit 4 – FERRCTRL** Flow Error flag reset control mode

| Value | Description |
|---|---|
| 0 | When an underflow/overflow condition flag is set, a new Write/Read command is needed to reset the flag. |
| 1 | When an underflow/overflow condition flag is set, a read status resets the flag. |

**Bit 0 – FIFOMODE** HSMCI Internal FIFO control mode
When the block length is greater than or equal to 3/4 of the HSMCI internal FIFO size, then the write transfer starts as soon as half the FIFO is filled. When the block length is greater than or equal to half the internal FIFO size, then the write transfer starts as soon as one quarter of the FIFO is filled. In other cases, the transfer starts as soon as the total amount of data is written in the internal FIFO.

| Value | Description |
|---|---|
| 0 | A write transfer starts when a sufficient amount of data is written into the FIFO. |
| 1 | A write transfer starts as soon as one data is written into the FIFO. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 40.14.18 HSMCI Write Protection Mode Register

**Name:** HSMCI_WPMR
**Offset:** 0xE4
**Reset:** 0x0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protect Key

| Value | Name | Description |
|---|---|---|
| 0x4D4349 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protect Enable
See "Register Write Protection" for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the Write Protection if WPKEY corresponds to 0x4D4349 ("MCI" in ASCII). |
| 1 | Enables the Write Protection if WPKEY corresponds to 0x4D4349 ("MCI" in ASCII). |

### 40.14.19 HSMCI Write Protection Status Register

**Name:** HSMCI_WPSR
**Offset:** 0xE8
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the HSMCI_WPSR. |
| 1 | A write protection violation has occurred since the last read of the HSMCI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

### 40.14.20 HSMCI FIFOx Memory Aperture

**Name:** HSMCI_FIFOx [x=0..255]
**Offset:** 0x00
**Reset:** 0
**Property:** R/W

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** Data to Read or Data to Write

# 41. Serial Peripheral Interface (SPI)

## 41.1 Description

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master”' which controls the data flow, while the other devices act as “slaves" which have data shifted into and out by the master. Different CPUs can take turn being masters (multiple master protocol, contrary to single master protocol where one CPU is always the master while all of the others are always slaves). One master can simultaneously shift data into multiple slaves. However, only one slave can drive its output to write data back to the master at any given time.

A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI)—This data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO)—This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK)—This control line is driven by the master and regulates the flow of the data bits. The master can transmit data at a variety of baud rates; there is one SPCK pulse for each bit that is transmitted.
- Slave Select (NSS)—This control line allows slaves to be turned on and off by hardware.

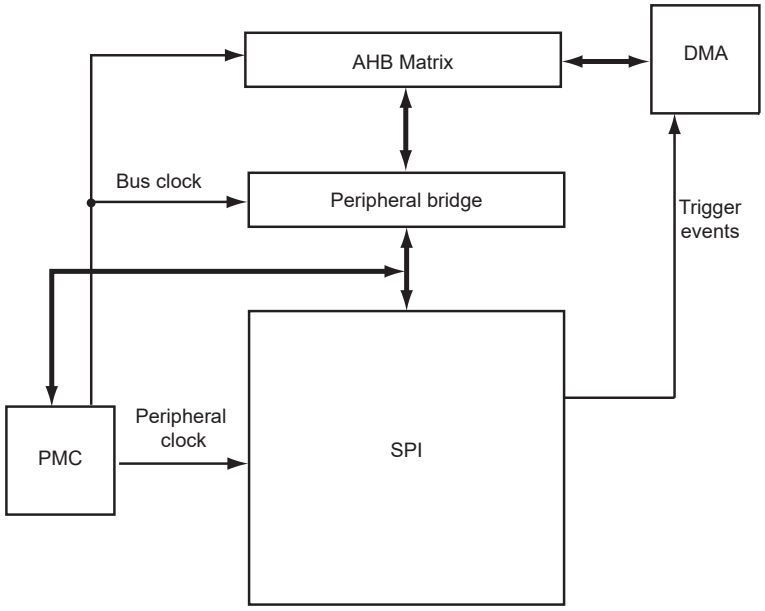
## 41.2 Embedded Characteristics

- Master or Slave Serial Peripheral Bus Interface
  - 8-bit to 16-bit programmable data length per chip select
  - Programmable phase and polarity per chip select
  - Programmable transfer delay between consecutive transfers and delay before SPI clock per chip select
  - Programmable delay between chip selects
  - Selectable mode fault detection
- Master Mode can Drive SPCK up to Peripheral Clock
- Master Mode Bit Rate can be Independent of the Processor/Peripheral Clock
- Slave Mode Operates on SPCK, Asynchronously with Core and Bus Clock
- Four Chip Selects with External Decoder Support Allow Communication with up to 15 Peripherals
- Communication with Serial External Devices Supported
  - Serial memories, such as DataFlash and 3-wire EEPROMs
  - Serial peripherals, such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - External coprocessors
- Connection to DMA Channel Capabilities, Optimizing Data Transfers
  - One channel for the receiver
  - One channel for the transmitter
- Register Write Protection

© 2021 Microchip Technology Inc.
and its subsidiaries

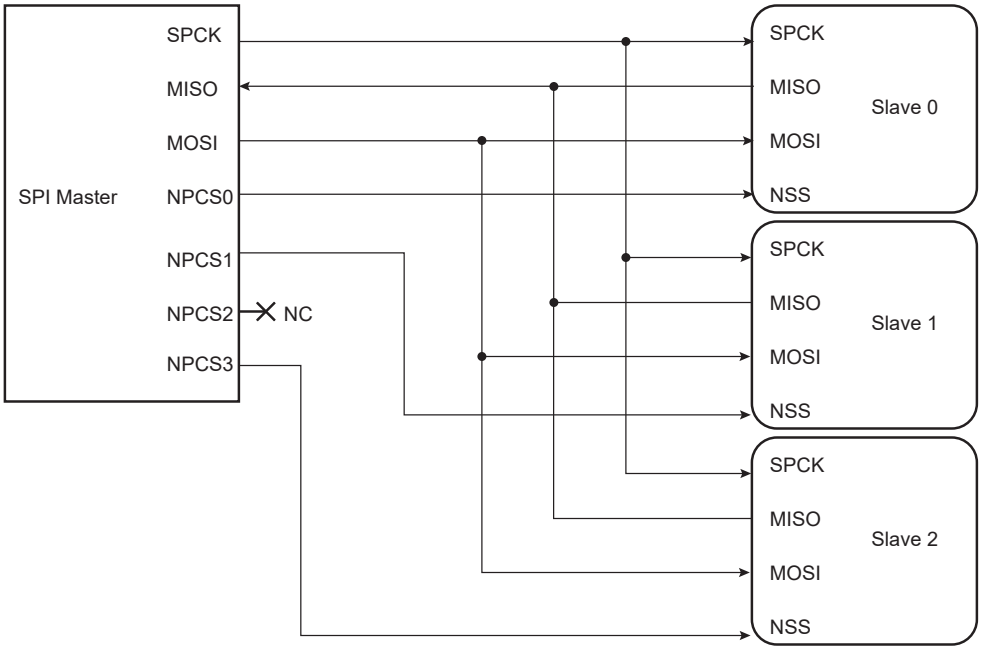
## 41.3 Block Diagram

**Figure 41-1. Block Diagram**




## 41.4 Application Block Diagram

**Figure 41-2. Application Block Diagram: Single Master/Multiple Slave Implementation**



## 41.5 Signal Description

**Table 41-1. Signal Description**

| Pin Name | Pin Description | Type | |
|---|---|---|---|
| | | Master | Slave |
| MISO | Master In Slave Out | Input | Output |
| MOSI | Master Out Slave In | Output | Input |
| SPCK | Serial Clock | Output | Input |
| NPCS1–NPCS3 | Peripheral Chip Selects | Output | Unused |
| NPCS0/NSS | Peripheral Chip Select/Slave Select | Output | Input |

## 41.6 Product Dependencies

### 41.6.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the SPI pins to their peripheral functions.

### 41.6.2 Power Management

The SPI can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SPI clock.

### 41.6.3 Interrupt

The SPI interface has an interrupt line connected to the interrupt controller. Handling the SPI interrupt requires programming the interrupt controller before configuring the SPI.

### 41.6.4 Direct Memory Access Controller (DMAC)

The SPI interface can be used in conjunction with the DMAC in order to reduce processor overhead. For a full description of the DMAC, refer to the relevant section.

## 41.7 Functional Description

### 41.7.1 Modes of Operation

The SPI operates in Master mode or in Slave mode.

- The SPI operates in Master mode by setting the MSTR bit in the SPI Mode Register (SPI_MR):
  - Pins NPCS0 to NPCS3 are all configured as outputs
  - The SPCK pin is driven
  - The MISO line is wired on the receiver input
  - The MOSI line is driven as an output by the transmitter.
- The SPI operates in Slave mode if the MSTR bit in SPI_MR is written to ‘0’:
  - The MISO line is driven by the transmitter output
  - The MOSI line is wired on the receiver input
  - The SPCK pin is driven by the transmitter to synchronize the receiver.
  - The NPCS0 pin becomes an input, and is used as a slave select signal (NSS)
  - The NPCS1 to NPCS3 pins are not driven and can be used for other purposes.

The data transfers are identically programmable for both modes of operation. The baud rate generator is activated only in Master mode.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 41.7.2 Data Transfer

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the SPI Chip Select registers (SPI_CSRx). The clock phase is programmed with the NCPHA bit. These two parameters determine the edges of the clock signal on which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Consequently, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are connected and require different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

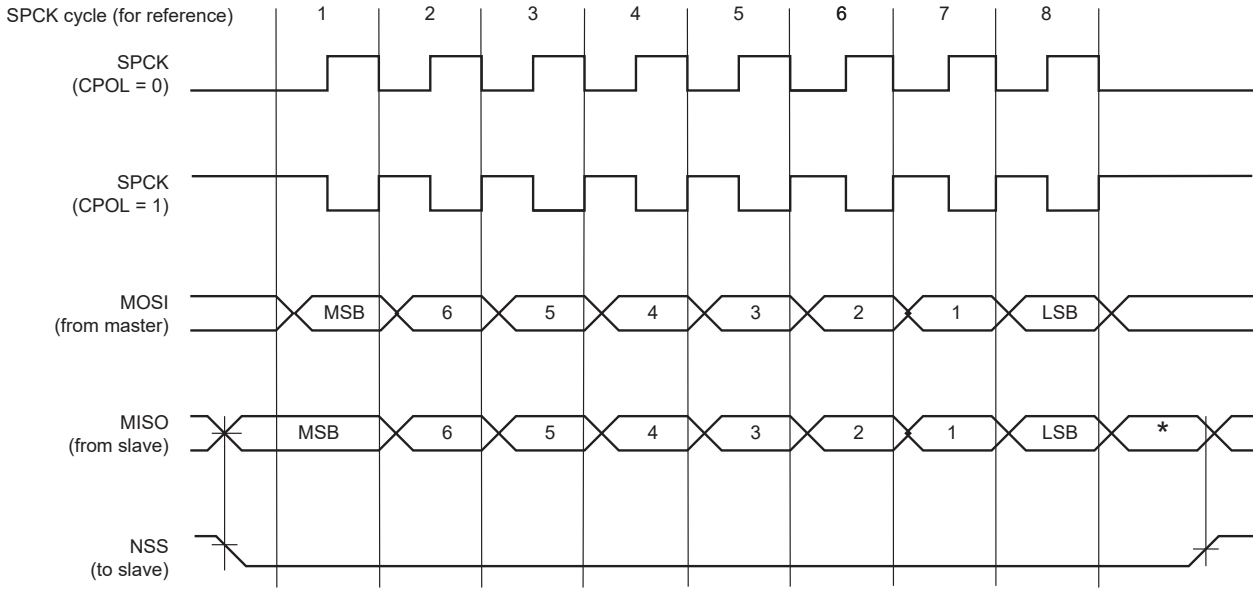
The table below shows the four modes and corresponding parameter settings.

**Table 41-2. SPI Bus Protocol Modes**

| SPI Mode | CPOL | NCPHA | Shift SPCK Edge | Capture SPCK Edge | SPCK Inactive Level |
|---|---|---|---|---|---|
| 0 | 0 | 1 | Falling | Rising | Low |
| 1 | 0 | 0 | Rising | Falling | Low |
| 2 | 1 | 1 | Rising | Falling | High |
| 3 | 1 | 0 | Falling | Rising | High |

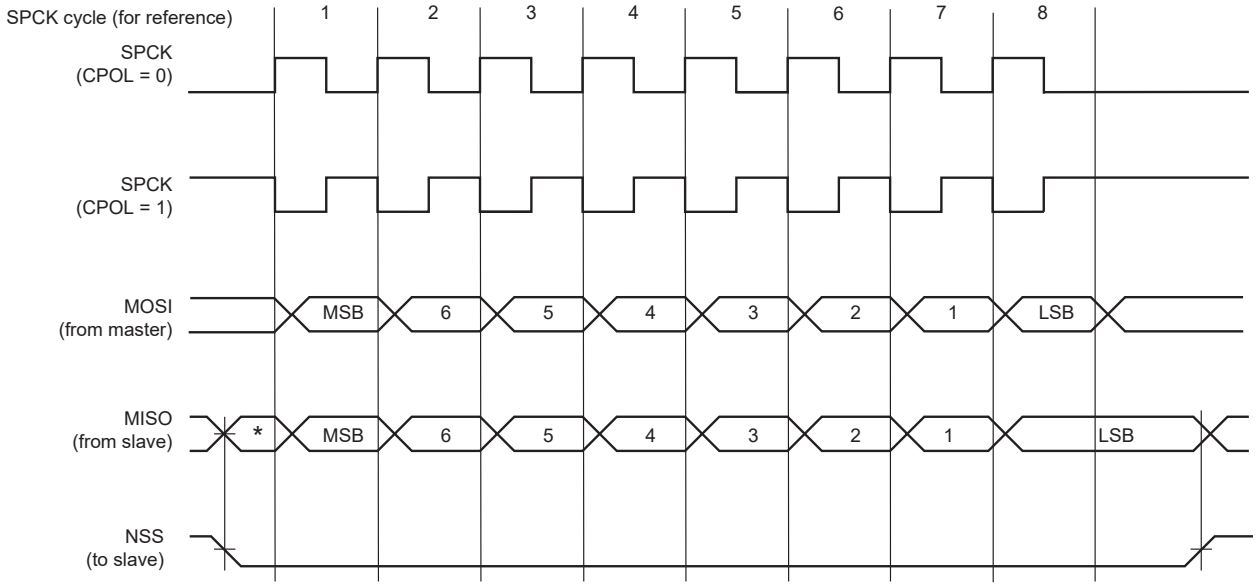
The following figures show examples of data transfers.

**Figure 41-3. SPI Transfer Format (NCPHA = 1, 8 bits per transfer)**




* Not defined.

**Figure 41-4. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)**




* Not defined.

### 41.7.3 Master Mode Operations

When configured in Master mode, the SPI operates on the clock generated by the internal programmable baud rate generator. It fully controls the data transfers to and from the slave(s) connected to the SPI bus. The SPI drives the chip select line to the slave and the serial clock signal (SPCK).

The SPI features two holding registers, the Transmit Data Register (SPI_TDR) and the Receive Data Register (SPI_RDR), and a single shift register. The holding registers maintain the data flow at a constant rate.

After enabling the SPI, a data transfer starts when the processor writes to SPI_TDR. The written data is immediately transferred into the internal shift register and the transfer on the SPI bus starts. While the data in the shift register is shifted on the MOSI line, the MISO line is sampled and shifted into the shift register. Data cannot be loaded in SPI_RDR without transmitting data. If there is no data to transmit, dummy data can be used (SPI_TDR filled with ones). If SPI_MR.WDRBT is set, transmission can occur only if SPI_RDR has been read. If Receiving mode is not required, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the SPI Status register (SPI_SR) can be discarded.

Before writing SPI_TDR, SPI_MR.PCS must be set in order to select a slave.

If new data is written in SPI_TDR during the transfer, it is kept in SPI_TDR until the current transfer is completed. Then, the received data is transferred from the shift register to SPI_RDR, the data in SPI_TDR is loaded in the shift register and a new transfer starts.

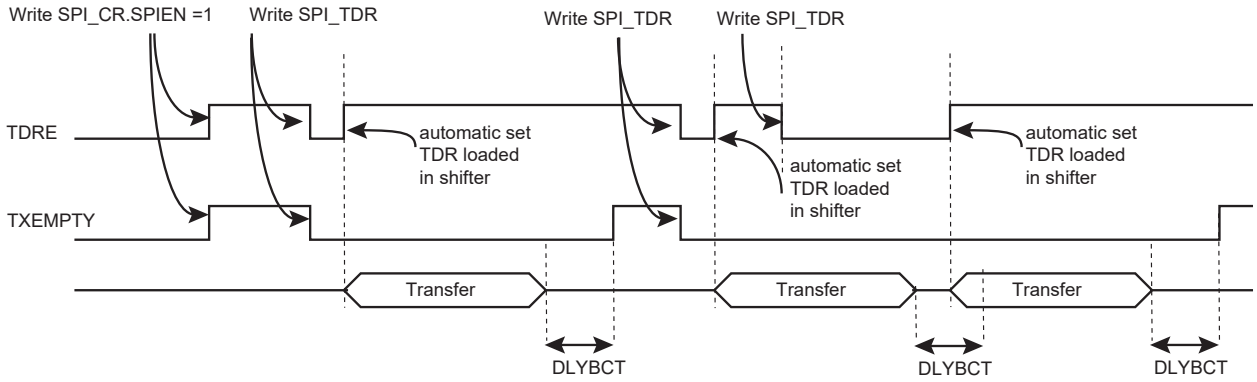
As soon as SPI_TDR is written, the Transmit Data Register Empty (TDRE) flag in SPI_SR is cleared. When the data written in SPI_TDR is loaded into the shift register, TDRE in SPI_SR is set. The TDRE flag is used to trigger the Transmit DMA channel.

See the figure below.

The end of transfer is indicated by the TXEMPTY flag in SPI_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

**Note:** When the SPI is enabled, the TDRE and TXEMPTY flags are set.

**Figure 41-5. TDRE and TXEMPTY Flag Behavior**



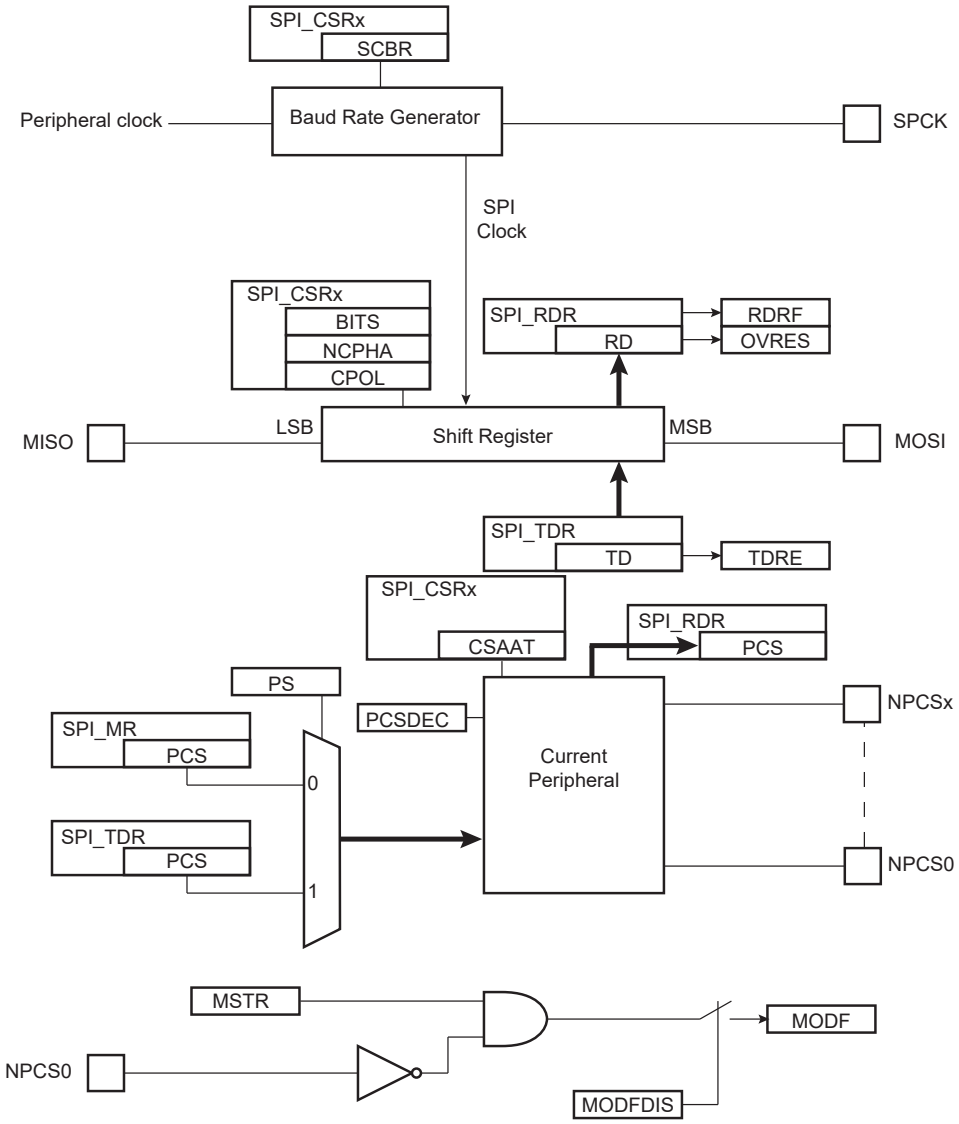

The transfer of received data from the internal shift register to SPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in SPI_SR. When the received data is read, SPI_SR.RDRF is cleared.

If SPI_RDR has not been read before new data is received, the Overrun Error (OVRES) flag in SPI_SR is set. As long as this flag is set, data is loaded in SPI_RDR. The user has to read SPI_SR to clear OVRES.

The following figures show, respectively, a block diagram of the SPI when operating in Master mode and a flow chart describing how transfers are handled.

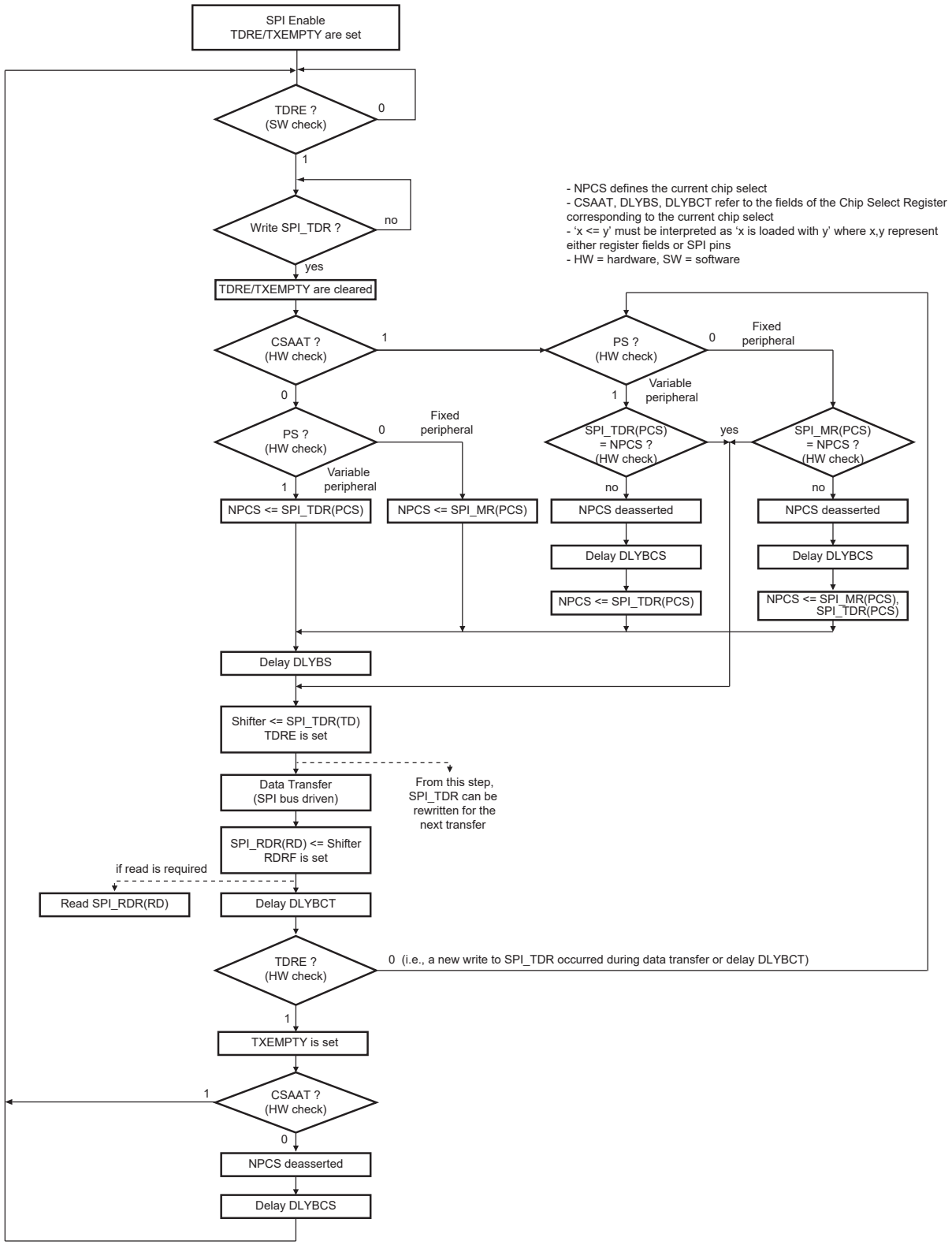
### 41.7.3.1 Master Mode Block Diagram

**Figure 41-6. Master Mode Block Diagram**



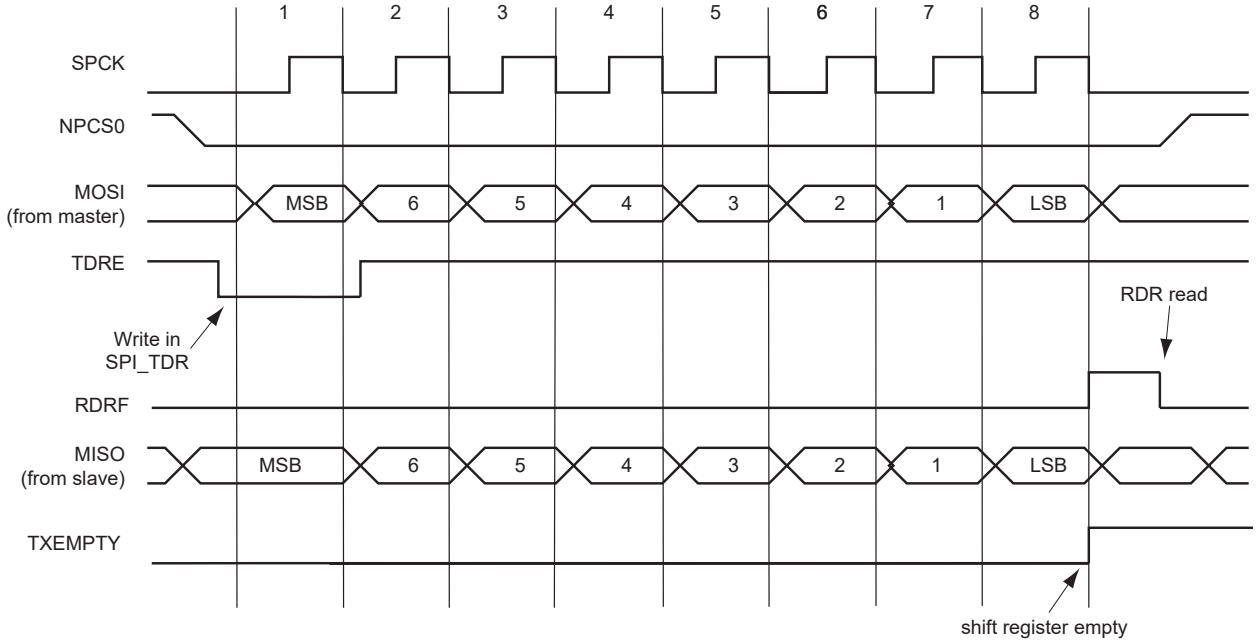
### 41.7.3.2 Master Mode Flow Diagram

**Figure 41-7. Master Mode Flow Diagram**



The figure below shows the behavior of Transmit Data Register Empty (TDRE), Receive Data Register (RDRF) and Transmission Register Empty (TXEMPTY) status flags within SPI_SR during an 8-bit data transfer in Fixed mode without the DMA involved.

**Figure 41-8. Status Register Flags Behavior**




### 41.7.3.3 Clock Generation

The SPI Baud rate clock is generated by dividing the peripheral clock by a value between 1 and 255.

If SPI_CSRx.SCBR is programmed to 1, the operating baud rate is peripheral clock (refer to the section “Electrical Characteristics” for the SPCK maximum frequency). Triggering a transfer while SPI_CSRx.SCBR is at 0 can lead to unpredictable results.

At reset, SPI_CSRx.SCBR=0 and the user has to program it to a valid value before performing the first transfer.

The divisor can be defined independently for each chip select, as it has to be programmed in SPI_CSRx.SCBR. This allows the SPI to automatically adapt the baud rate for each interfaced peripheral without reprogramming.

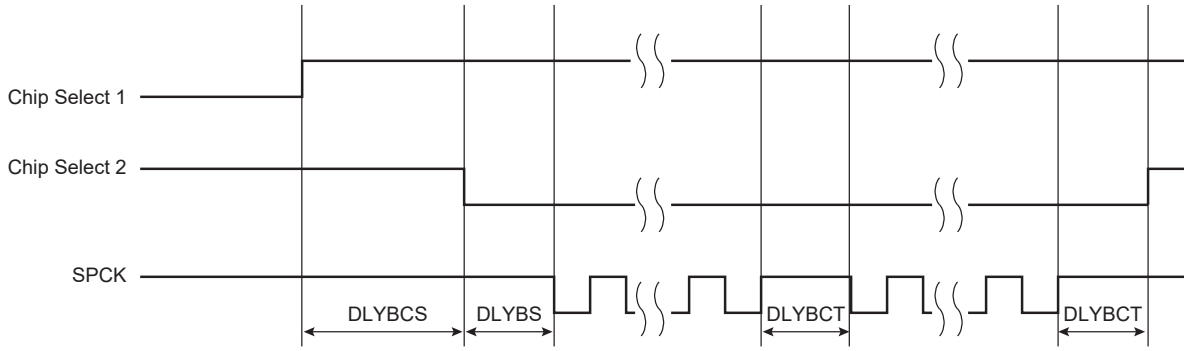
### 41.7.3.4 Transfer Delays

The following figure shows a chip select transfer change and consecutive transfers on the same chip select. Three delays can be programmed to modify the transfer waveforms:

- Delay between the chip selects—programmable only once for all chip selects by writing field SPI_MR.DLYBCS. The SPI slave device deactivation delay is managed through DLYBCS. If there is only one SPI slave device connected to the master, DLYBCS does not need to be configured. If several slave devices are connected to a master, DLYBCS must be configured depending on the highest deactivation delay. Refer to details on the SPI slave device in the section “Electrical Characteristics”.
- Delay before SPCK—independently programmable for each chip select by writing SPI_CSRx.DLYBS. The SPI slave device activation delay is managed through DLYBS. Refer to details on the SPI slave device in the section “Electrical Characteristics” to define DLYBS.
- Delay between consecutive transfers—independently programmable for each chip select by writing SPI_CSRx.DLYBCT. The time required by the SPI slave device to process received data is managed through DLYBCT. This time depends on the SPI slave system activity.

These delays allow the SPI to be adapted to the interfaced peripherals and their speed and bus release time.

**Figure 41-9. Programmable Delays**




### 41.7.3.5 Peripheral Selection

The serial peripherals are selected through the assertion of the NPCS0 to NPCS3 signals. By default, all NPCS signals are high before and after each transfer.

- Fixed Peripheral Select Mode: SPI exchanges data with only one peripheral.
  Fixed Peripheral Select mode is enabled by clearing SPI_MR.PS. In this case, the current peripheral is defined by SPI_MR.PCS. SPI_TDR.PCS has no effect.
- Variable Peripheral Select Mode: Data can be exchanged with more than one peripheral without having to reprogram SPI_MR.PCS.
  Variable Peripheral Select mode is enabled by setting SPI_MR.PS. SPI_TDR.PCS is used to select the current peripheral. This means that the peripheral selection can be defined for each new data. The value must be written in a single access to SPI_TDR in the following format:
  [xxxxxxx(7-bit) + LASTXFER(1-bit)(1)+ xxxx(4-bit) + PCS (4-bit) + TD (8- to 16-bit data)]

  with LASTXFER at 0 or 1 depending on the CSAAT bit, and PCS equal to the chip select to assert, as defined in section SPI Transmit Data Register.

  **Note:**
  1. Optional

  For details on CSAAT, LASTXFER and CSNAAT, see section Peripheral Deselection with another DMA.

  If LASTXFER is used, the command must be issued after writing the last character. Instead of LASTXFER, the user can use the SPIDIS command. After the end of the DMA transfer, it is necessary to wait for the TXEMPTY flag and then write SPIDIS into the SPI Control Register (SPI_CR). This does not change the configuration register values). The NPCS is disabled after the last character transfer. Then, another DMA transfer can be started if SPI_CR.SPIEN has previously been written.

### 41.7.3.6 SPI Direct Access Memory Controller (DMAC)

In both Fixed and Variable modes, the Direct Memory Access Controller (DMAC) can be used to reduce processor overhead.

The fixed peripheral selection allows buffer transfers with a single peripheral. Using the DMAC is an optimal means, as the size of the data transfer between the memory and the SPI is either 8 bits or 16 bits. However, if the peripheral selection is modified, SPI_MR must be reprogrammed.

The variable peripheral selection allows buffer transfers with multiple peripherals without reprogramming SPI_MR. Data written in SPI_TDR is 32 bits wide and defines the real data to be transmitted and the destination peripheral. Using the DMAC in this mode requires 32-bit wide buffers, with the data in the LSBs and the PCS and LASTXFER fields in the MSBs. However, the SPI still controls the number of bits (8 to 16) to be transferred through MISO and MOSI lines with the chip select configuration registers. This is not the optimal means in terms of memory size for the buffers, but it provides a very effective means to exchange data with several peripherals without any intervention of the processor.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 41.7.3.7 Peripheral Chip Select Decoding

The user can program the SPI to operate with up to 15 slave peripherals by decoding the four chip select lines, NPCS0 to NPCS3 with an external decoder/demultiplexer (see figure below). This can be enabled by setting SPI_MR.PCSDEC.

When operating without decoding, the SPI makes sure that in any case only one chip select line is activated, i.e., one NPCS line driven low at a time. If two bits are defined low in a PCS field, only the lowest numbered chip select is driven low.

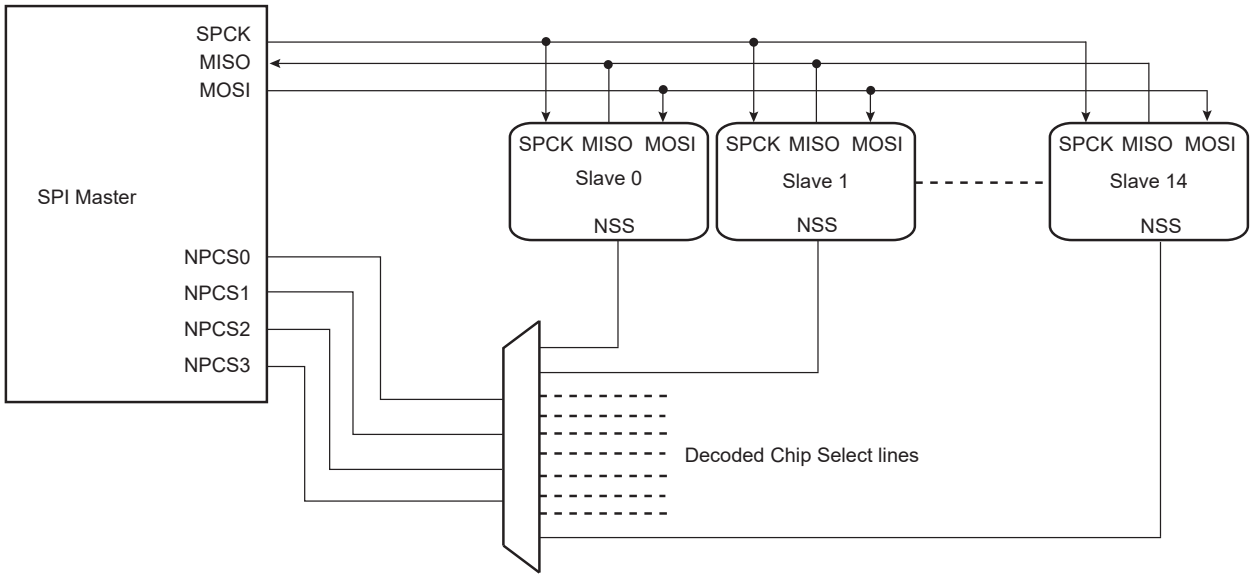
When operating with decoding, the SPI directly outputs the value defined by the PCS field on the NPCS lines of either SPI_MR or SPI_TDR (depending on PS).

As the SPI sets a default value of 0xF on the chip select lines (i.e., all chip select lines at 1) when not processing any transfer, only 15 peripherals can be decoded.

The SPI has four chip select registers (SPI_CSR0...SPI_CSR3). As a result, when external decoding is activated, each NPCS chip select defines the characteristics of up to four peripherals. As an example, SPI_CRS0 defines the characteristics of the externally decoded peripherals 0 to 3, corresponding to the PCS values 0x0 to 0x3. Consequently, the user has to make sure to connect compatible peripherals on the decoded chip select lines 0 to 3, 4 to 7, 8 to 11 and 12 to 14. The following figure shows this type of implementation.

If SPI_CSRx.CSAAT bit is used, with or without the DMAC, the Mode Fault detection for NPCS0 line must be disabled. This is not needed for all other chip select lines since Mode Fault detection is only on NPCS0.

**Figure 41-10. Chip Select Decoding Application Block Diagram: Single Master/Multiple Slave Implementation**




### 41.7.3.8 Peripheral Deselection without DMA

During a transfer of more than one unit of data on a chip select without the DMA, SPI_TDR is loaded by the processor, the TDRE flag rises as soon as the content of SPI_TDR is transferred into the internal shift register. When this flag is detected high, SPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the chip select is not deasserted between the two transfers. But depending on the application software handling the SPI status register flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload SPI_TDR in time to keep the chip select active (low). A null DLYBCT value (delay between consecutive transfers) in SPI_CSR, gives even less time for the processor to reload SPI_TDR. With some SPI slave peripherals, if the chip select line must remain active (low) during a full set of transfers, communication errors can occur.

To facilitate interfacing with such devices, the chip select registers [SPI_CSR0...SPI_CSR3] can be programmed with the Chip Select Active After Transfer (CSAAT) bit at 1. This allows the chip select lines to remain in their current state (low = active) until a transfer to another chip select is required. Even if SPI_TDR is not reloaded, the chip

select remains active. To deassert the chip select line at the end of the transfer, the Last Transfer (LASTXFER) bit in SPI_CR must be set after writing the last data to transmit into SPI_TDR.

#### 41.7.3.9 Peripheral Deselection with DMA

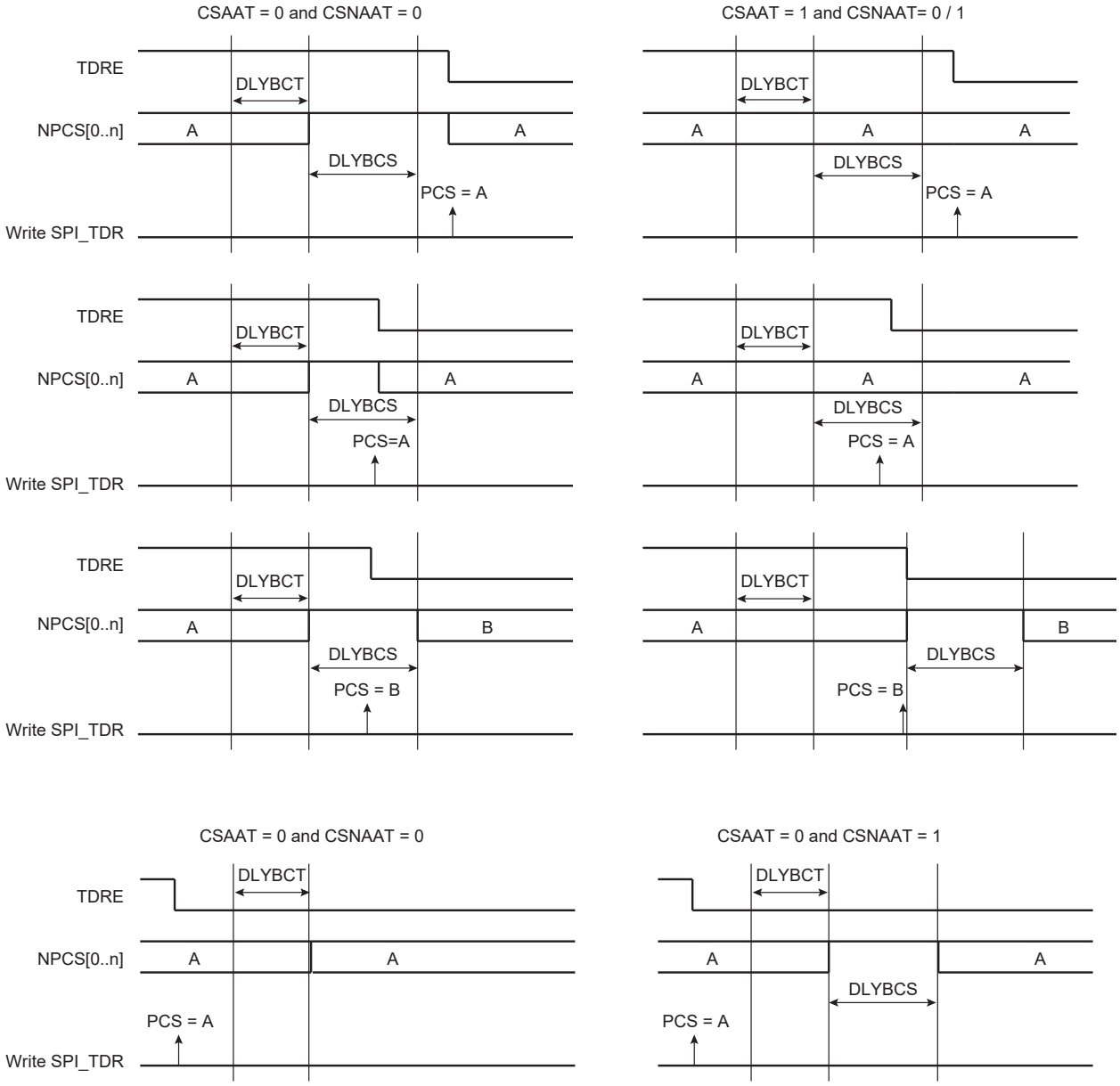
DMA provides faster reloads of SPI_TDR compared to software. However, depending on the system activity, it is not guaranteed that SPI_TDR is written with the next data before the end of the current transfer. Consequently, data can be lost by the deassertion of the NPCS line for SPI slave peripherals requiring the chip select line to remain active between two transfers. The only way to guarantee a safe transfer in this case is the use of the CSAAT and LASTXFER bits.

When the CSAAT bit is configured to 0, the NPCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a chip select, the TDRE flag rises as soon as the content of SPI_TDR is transferred into the internal shift register. When this flag is detected, SPI_TDR can be reloaded. If this reload occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the chip select is not deasserted between the two transfers. This can lead to difficulties to interface with some serial peripherals requiring the chip select to be deasserted after each transfer. To facilitate interfacing with such devices, SPI_CSR can be programmed with the Chip Select Not Active After Transfer (CSNAAT) bit at 1. This allows the chip select lines to be deasserted systematically during a time “DLYBCS” (the value of the CSNAAT bit is processed only if the CSAAT bit is configured to 0 for the same chip select).

The following figure shows different peripheral deselection cases and the effect of the CSAAT and CSNAAT bits.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 41-11. Peripheral Deselection**




### 41.7.3.10 Mode Fault Detection

The SPI has the capability to operate in multimaster environment. Consequently, the NPCS0/NSS line must be monitored. If one of the masters on the SPI bus is currently transmitting, the NPCS0/NSS line is low and the SPI must not transmit any data. A mode fault is detected when the SPI is programmed in Master mode and a low level is driven by an external master on the NPCS0/NSS signal. In multimaster environment, NPCS0, MOSI, MISO and SPCK pins must be configured in open drain (through the PIO controller). When a mode fault is detected, SPI_SR.MODF bit is set until SPI_SR is read and the SPI is automatically disabled until it is reenabled by setting SPI_CR.SPIEN bit.

By default, the mode fault detection is enabled. The user can disable it by setting SPI_MR.MODFDIS bit.

## 41.7.4 SPI Slave Mode

When operating in Slave mode, the SPI processes data bits on the clock provided on the SPI clock pin (SPCK).

The SPI waits until NSS goes active before receiving the serial clock from an external master. When NSS falls, the clock is validated and the data is loaded in SPI_RDR depending on the configuration of SPI_CSR0.BITS. These bits

are processed following a phase and a polarity defined respectively by the NCPHA and CPOL bits in SPI_CSR0. Note that the fields BITS, CPOL and NCPHA of the other chip select registers (SPI_CSR1...SPI_CSR3) have no effect when the SPI is programmed in Slave mode.

The bits are shifted out on the MISO line and sampled on the MOSI line.

**Note:** For more information on SPI_CSRx.BITS, see the note in section SPI Chip Select Register.

When all bits are processed, the received data is transferred in SPI_RDR and the RDRF bit rises. If SPI_RDR has not been read before new data is received, the Overrun Error Status (OVRES) bit in SPI_SR is set. As long as this flag is set, data is loaded in SPI_RDR. The user must read SPI_SR to clear the OVRES bit.

When a transfer starts, the data shifted out is the data present in the internal shift register. If no data has been written in SPI_TDR, the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the internal shift register resets to 0.

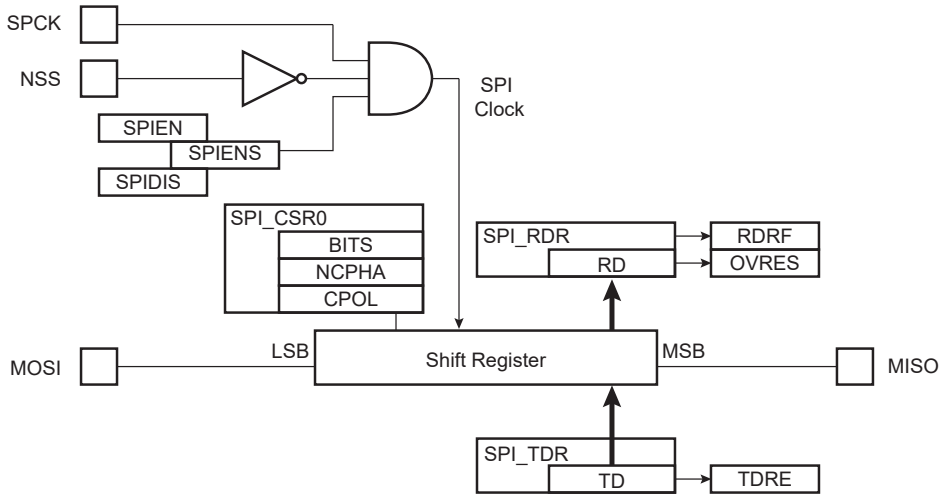
When a first data is written in SPI_TDR, it is transferred immediately in the internal shift register and the TDRE flag rises. If new data is written, it remains in SPI_TDR until a transfer occurs, i.e., NSS falls and there is a valid clock on the SPCK pin. When the transfer occurs, the last data written in SPI_TDR is transferred in the internal shift register and the TDRE flag rises. This enables frequent updates of critical variables with single transfers.

Then, new data is loaded in the internal shift register from SPI_TDR. If no character is ready to be transmitted, i.e., no character has been written in SPI_TDR since the last load from SPI_TDR to the internal shift register, SPI_TDR is retransmitted. In this case the Underrun Error Status Flag (UNDES) is set in SPI_SR.

In Slave mode, if the NSS line rises and the received character length does not match the configuration defined in SPI_CSR0.BITS the flag SFERR is set in SPI_SR.

The following figure shows a block diagram of the SPI when operating in Slave mode.

**Figure 41-12. Slave Mode Functional Block Diagram**




### 41.7.5 Register Write Protection

To prevent any single software error from corrupting SPI behavior, certain registers in the address space can be write-protected in the SPI Write Protection Mode Register (SPI_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the SPI Write Protection Status Register (SPI_WPSR) is set and the WPVSRC field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading SPI_WPSR.

The following registers are write-protected when WPEN is set in SPI_WPMR:

- SPI Mode Register
- SPI Chip Select Register

The following register is write-protected when WPCREN is set in SPI_WPMR:

- SPI Control Register

The following registers are write-protected when WPITEN is set in SPI_WPMR:

- SPI Interrupt Enable Register
- SPI Interrupt Disable Register

## 41.8 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SPI_CR | 7:0 | SWRST | | | | | | SPIDIS | SPIEN |
| | | 15:8 | | | | REQCLR | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | LASTXFER |
| 0x04 | SPI_MR | 7:0 | LLB | | WDRBT | MODFDIS | | PCSDEC | PS | MSTR |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | PCS[3:0] | | | |
| | | 31:24 | DLYBCS[7:0] | | | | | | | |
| 0x08 | SPI_RDR | 7:0 | RD[7:0] | | | | | | | |
| | | 15:8 | RD[15:8] | | | | | | | |
| | | 23:16 | | | | | PCS[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | SPI_TDR | 7:0 | TD[7:0] | | | | | | | |
| | | 15:8 | TD[15:8] | | | | | | | |
| | | 23:16 | | | | | PCS[3:0] | | | |
| | | 31:24 | | | | | | | | LASTXFER |
| 0x10 | SPI_SR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | SFERR | | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | SPIENS |
| | | 31:24 | | | | | | | | |
| 0x14 | SPI_IER | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | SPI_IDR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | SPI_IMR | 7:0 | | | | | OVRES | MODF | TDRE | RDRF |
| | | 15:8 | | | | | | UNDES | TXEMPTY | NSSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | SPI_CSR0 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x34 | SPI_CSR1 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x38 | SPI_CSR2 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x3C | SPI_CSR3 | 7:0 | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | DLYBCT[7:0] | | | | | | | |
| 0x40 ... 0xE3 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE4 | SPI_WPMR | 7:0 | | | | | | WPCREN | WPITEN | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | SPI_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 41.8.1 SPI Control Register

**Name:** SPI_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

This register can only be written if the WPCREN bit is cleared in the SPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | LASTXFER |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | REQCLR | | | | |
| Access | | | | W | | | | |
| Reset | | | | – | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | | | | | SPIDIS | SPIEN |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

**Bit 24 – LASTXFER** Last Transfer
Refer to section Peripheral Selection for more details.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The current NPCS is deasserted after the character written in TD has been transferred. When SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed. |

**Bit 12 – REQCLR** Request to Clear the Comparison Trigger
0: No effect.
1: Restarts the comparison trigger to enable SPI_RDR loading.

**Bit 7 – SWRST** SPI Software Reset
The SPI is in Slave mode after software reset.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Reset the SPI. A software-triggered hardware reset of the SPI interface is performed. |

**Bit 1 – SPIDIS** SPI Disable
All pins are set in Input mode after completion of the transmission in progress, if any.
If a transfer is in progress when SPIDIS is set, the SPI completes the transmission of the shifter register and does not start any new transfer, even if SPI_THR is loaded.
If both SPIEN and SPIDIS are equal to one when SPI_CR is written, the SPI is disabled.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the SPI. |

**Bit 0 – SPIEN** SPI Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the SPI to transfer and receive data. |

### 41.8.2 SPI Mode Register

**Name:** SPI_MR
**Offset:** 0x04
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in theSPI Write Protection Mode Register .

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DLYBCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PCS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LLB | | WDRBT | MODFDIS | | PCSDEC | PS | MSTR |
| Access | R/W | | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | | 0 | 0 | 0 |

**Bits 31:24 – DLYBCS[7:0]** Delay Between Chip Selects
This field defines the delay between the inactivation and the activation of NPCS. The DLYBCS time guarantees nonoverlapping chip selects and solves bus contentions in case of peripherals having long data float times.
If DLYBCS is lower than 6, six peripheral clock periods are inserted by default.
Otherwise, the following equation determines the delay:

$$\text{Delay Between Chip Selects} = \frac{\text{DLYBCS}}{\text{f}_{\text{peripheral clock}}}$$

**Bits 19:16 – PCS[3:0]** Peripheral Chip Select
This field is only used if fixed peripheral select is active (PS = 0).
If SPI_MR.PCSDEC = 0:
PCS = xxx0 NPCS[3:0] = 1110
PCS = xx01 NPCS[3:0] = 1101
PCS = x011 NPCS[3:0] = 1011
PCS = 0111 NPCS[3:0] = 0111
PCS = 1111 forbidden (no peripheral is selected)
(x = don't care)
If SPI_MR.PCSDEC = 1:
NPCS[3:0] output signals = PCS.

**Bit 7 – LLB** Local Loopback Enable
LLB controls the local loopback on the data shift register for testing in Master mode only (MISO is internally connected on MOSI).

| Value | Description |
|---|---|
| 0 | Local loopback path disabled. |
| 1 | Local loopback path enabled. |

**Bit 5 – WDRBT** Wait Data Read Before Transfer

| Value | Description |
|---|---|
| 0 | No Effect. In Master mode, a transfer can be initiated regardless of SPI_RDR state. |
| 1 | In Master mode, a transfer can start only if SPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception. |

**Bit 4 – MODFDIS** Mode Fault Detection

| Value | Description |
|---|---|
| 0 | Mode fault detection enabled |
| 1 | Mode fault detection disabled |

**Bit 2 – PCSDEC** Chip Select Decode

When PCSDEC = 1, up to 15 chip select signals can be generated with the four NPCS lines using an external 4-bit to 16-bit decoder. The chip select registers define the characteristics of the 15 chip selects, with the following rules:
SPI_CSR0 defines peripheral chip select signals 0 to 3.
SPI_CSR1 defines peripheral chip select signals 4 to 7.
SPI_CSR2 defines peripheral chip select signals 8 to 11.
SPI_CSR3 defines peripheral chip select signals 12 to 14.

| Value | Description |
|---|---|
| 0 | The chip select lines are directly connected to a peripheral device. |
| 1 | The four NPCS chip select lines are connected to a 4-bit to 16-bit decoder. |

**Bit 1 – PS** Peripheral Select

| Value | Description |
|---|---|
| 0 | Fixed Peripheral Select |
| 1 | Variable Peripheral Select |

**Bit 0 – MSTR** Master/Slave Mode

| Value | Description |
|---|---|
| 0 | SPI is in Slave mode |
| 1 | SPI is in Master mode |

### 41.8.3 SPI Receive Data Register

**Name:** SPI_RDR
**Offset:** 0x08
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PCS[3:0] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 19:16 – PCS[3:0]** Peripheral Chip Select
In Master mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits are read as zero.
When using Variable Peripheral Select mode (PS = 1 in SPI_MR), it is mandatory to set SPI_MR.WDRBT bit if the PCS field must be processed in SPI_RDR.

**Bits 15:0 – RD[15:0]** Receive Data
Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

### 41.8.4 SPI Transmit Data Register

**Name:** SPI_TDR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | LASTXFER |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PCS[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 24 – LASTXFER** Last Transfer
This field is only used if variable peripheral select is active (SPI_MR.PS = 1).

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | The current NPCS is deasserted after the transfer of the character written in TD. When SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed. |

**Bits 19:16 – PCS[3:0]** Peripheral Chip Select
This field is only used if variable peripheral select is active (SPI_MR.PS = 1).
If SPI_MR.PCSDEC = 0:
PCS = xxx0 NPCS[3:0] = 1110
PCS = xx01 NPCS[3:0] = 1101
PCS = x011 NPCS[3:0] = 1011
PCS = 0111 NPCS[3:0] = 0111
PCS = 1111 forbidden (no peripheral is selected)
(x = don’t care)
If SPI_MR.PCSDEC = 1:
NPCS[3:0] output signals = PCS.

**Bits 15:0 – TD[15:0]** Transmit Data
Data to be transmitted by the SPI interface is stored in this register. Information to be transmitted must be written to this register in a right-justified format.

### 41.8.5 SPI Status Register

**Name:** SPI_SR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SPIENS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | SFERR | | UNDES | TXEMPTY | NSSR |
| Access | | | | R | | R | R | R |
| Reset | | | | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 16 – SPIENS** SPI Enable Status

| Value | Description |
|---|---|
| 0 | SPI is disabled. |
| 1 | SPI is enabled. |

**Bit 12 – SFERR** Slave Frame Error (cleared on read)

| Value | Description |
|---|---|
| 0 | There is no frame error detected for a slave access since the last read of SPI_SR. |
| 1 | In Slave mode, the Chip Select raised while the character defined in SPI_CSR0.BITS was not complete. |

**Bit 10 – UNDES** Underrun Error Status (Slave mode only) (cleared on read)

| Value | Description |
|---|---|
| 0 | No underrun has been detected since the last read of SPI_SR. |
| 1 | A transfer starts whereas no data has been loaded in SPI_TDR. |

**Bit 9 – TXEMPTY** Transmission Registers Empty (cleared by writing SPI_TDR)

| Value | Description |
|---|---|
| 0 | As soon as data is written in SPI_TDR. |
| 1 | SPI_TDR and internal shift register are empty. If a transfer delay has been defined, TXEMPTY is set after the end of this delay. |

**Bit 8 – NSSR** NSS Rising (cleared on read)

| Value | Description |
|---|---|
| 0 | No rising edge detected on NSS pin since the last read of SPI_SR. |
| 1 | A rising edge occurred on NSS pin since the last read of SPI_SR. |

**Bit 3 – OVRES** Overrun Error Status (cleared on read)
An overrun occurs when SPI_RDR is loaded at least twice from the internal shift register since the last read of SPI_RDR.

| Value | Description |
|---|---|
| 0 | No overrun has been detected since the last read of SPI_SR. |
| 1 | An overrun has occurred since the last read of SPI_SR. |

**Bit 2 – MODF** Mode Fault Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No mode fault has been detected since the last read of SPI_SR. |
| 1 | A mode fault occurred since the last read of SPI_SR. |

**Bit 1 – TDRE** Transmit Data Register Empty (cleared by writing SPI_TDR)
0: Data has been written to SPI_TDR and not yet transferred to the internal shift register.
1: The last data written in SPI_TDR has been transferred to the internal shift register.
TDRE is cleared when the SPI is disabled or at reset. Enabling the SPI sets the TDRE flag.

**Bit 0 – RDRF** Receive Data Register Full (cleared by reading SPI_RDR)
0: No data has been received since the last read of SPI_RDR.
1: Data has been received and the received data has been transferred from the internal shift register to SPI_RDR since the last read of SPI_RDR.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 41.8.6 SPI Interrupt Enable Register

**Name:** SPI_IER
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

This register can only be written if the WPITEN bit is cleared in the SPI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNDES | TXEMPTY | NSSR |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 10 – UNDES** Underrun Error Interrupt Enable

**Bit 9 – TXEMPTY** Transmission Registers Empty Enable

**Bit 8 – NSSR** NSS Rising Interrupt Enable

**Bit 3 – OVRES** Overrun Error Interrupt Enable

**Bit 2 – MODF** Mode Fault Error Interrupt Enable

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Enable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 41.8.7 SPI Interrupt Disable Register

**Name:** SPI_IDR
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

This register can only be written if the WPITEN bit is cleared in the SPI Write Protection Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNDES | TXEMPTY | NSSR |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 10 – UNDES** Underrun Error Interrupt Disable

**Bit 9 – TXEMPTY** Transmission Registers Empty Disable

**Bit 8 – NSSR** NSS Rising Interrupt Disable

**Bit 3 – OVRES** Overrun Error Interrupt Disable

**Bit 2 – MODF** Mode Fault Error Interrupt Disable

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Disable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 41.8.8 SPI Interrupt Mask Register

**Name:** SPI_IMR
**Offset:** 0x1C
**Reset:** 0x0
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNDES | TXEMPTY | NSSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | MODF | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 10 – UNDES** Underrun Error Interrupt Mask

**Bit 9 – TXEMPTY** Transmission Registers Empty Mask

**Bit 8 – NSSR** NSS Rising Interrupt Mask

**Bit 3 – OVRES** Overrun Error Interrupt Mask

**Bit 2 – MODF** Mode Fault Error Interrupt Mask

**Bit 1 – TDRE** SPI Transmit Data Register Empty Interrupt Mask

**Bit 0 – RDRF** Receive Data Register Full Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 41.8.9 SPI Chip Select Register

**Name:** SPI_CSRx
**Offset:** 0x30 + x*0x04 [x=0..3]
**Reset:** 0
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

SPI_CSRx must be written even if the user wants to use the default reset values. The BITS field is not updated with the translated value unless the register is written.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DLYBCT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DLYBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCBR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BITS[3:0] | | | | CSAAT | CSNAAT | NCPHA | CPOL |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DLYBCT[7:0]** Delay Between Consecutive Transfers
This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.
When DLYBCT = 0, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.
Otherwise, the following equation determines the delay:
DLYBCT = Delay Between Consecutive Transfers × $f_{peripheral\ clock}$ / 32

**Bits 23:16 – DLYBS[7:0]** Delay Before SPCK
This field defines the delay from NPCS falling edge (activation) to the first valid SPCK transition.
When DLYBS = 0, the delay is half the SPCK clock period.
Otherwise, the following equation determines the delay:
DLYBS = Delay Before SPCK × $f_{peripheral\ clock}$

**Bits 15:8 – SCBR[7:0]** Serial Clock Bit Rate
In Master mode, the SPI Interface uses a modulus counter to derive the SPCK bit rate from the peripheral clock. The bit rate is selected by writing a value from1 to 255 in the SCBR field. The following equation determines the SPCK bit rate:
SCBR = $f_{peripheral\ clock}$ / SPCK Bit Rate
Programming the SCBR field to 0 is forbidden. Triggering a transfer while SCBR is at 0 can lead to unpredictable results.
At reset, SCBR is 0 and the user has to program it at a valid value before performing the first transfer.
**Note:** If one of the SCBR fields in SPI_CSRx is set to 1, the other SCBR fields in SPI_CSRx must be set to 1 as well, if they are used to process transfers. If they are not used to transfer data, they can be set at any value.

**Bits 7:4 – BITS[3:0]** Bits Per Transfer
(See Note under the register table in SPI Chip Select Register.)
The BITS field determines the number of data bits transferred. Reserved values should not be used.

| Value | Name | Description |
|---|---|---|
| 0 | 8_BIT | 8 bits for transfer |
| 1 | 9_BIT | 9 bits for transfer |
| 2 | 10_BIT | 10 bits for transfer |
| 3 | 11_BIT | 11 bits for transfer |
| 4 | 12_BIT | 12 bits for transfer |
| 5 | 13_BIT | 13 bits for transfer |
| 6 | 14_BIT | 14 bits for transfer |
| 7 | 15_BIT | 15 bits for transfer |
| 8 | 16_BIT | 16 bits for transfer |
| 9 | – | Reserved |
| 10 | – | Reserved |
| 11 | – | Reserved |
| 12 | – | Reserved |
| 13 | – | Reserved |
| 14 | – | Reserved |
| 15 | – | Reserved |

**Bit 3 – CSAAT** Chip Select Active After Transfer

| Value | Description |
|---|---|
| 0 | The Peripheral Chip Select Line rises as soon as the last transfer is achieved. |
| 1 | The Peripheral Chip Select Line does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select. |

**Bit 2 – CSNAAT** Chip Select Not Active After Transfer (ignored if CSAAT = 1)

| Value | Description |
|---|---|
| 0 | The Peripheral Chip Select Line does not rise between two transfers if SPI_TDR is reloaded before the end of the first transfer and if the two transfers occur on the same chip select. |
| 1 | The Peripheral Chip Select Line rises systematically after each transfer performed on the same slave. It remains inactive after the end of transfer for a minimal duration of:<br>$\frac{\mathrm{DLYBCS}}{\mathrm{f_{peripheral\ clock}}}$ (If field DLYBCS is lower than 6, a minimum of six periods is introduced.) |

**Bit 1 – NCPHA** Clock Phase
NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | Data is changed on the leading edge of SPCK and captured on the following edge of SPCK. |
| 1 | Data is captured on the leading edge of SPCK and changed on the following edge of SPCK. |

**Bit 0 – CPOL** Clock Polarity
CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | The inactive state value of SPCK is logic level zero. |
| 1 | The inactive state value of SPCK is logic level one. |

### 41.8.10 SPI Write Protection Mode Register

**Name:** SPI_WPMR
**Offset:** 0xE4
**Reset:** 0x0
**Property:** Read/Write

See section Register Write Protection for the list of registers that can be write-protected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | WPCREN | WPITEN | WPEN |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x535049 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 2 – WPCREN** Write Protection Control Register Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection on the Control register if WPKEY corresponds to 0x535049. |
| 1 | Enables the write protection on the Control register if WPKEY corresponds to 0x535049. |

**Bit 1 – WPITEN** Write Protection Interrupt Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection on Interrupt registers if WPKEY corresponds to 0x535049. |
| 1 | Enables the write protection on Interrupt registers if WPKEY corresponds to 0x535049. |

**Bit 0 – WPEN** Write Protection Enable

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII) |
| 1 | Enables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII) |

© 2021 Microchip Technology Inc. and its subsidiaries

### 41.8.11 SPI Write Protection Status Register

**Name:** SPI_WPSR
**Offset:** 0xE8
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 15:8 – WPVSRC[7:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of SPI_WPSR. |
| 1 | A write protection violation has occurred since the last read of SPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 42. Quad Serial Peripheral Interface (QSPI)

## 42.1 Description

The Quad Serial Peripheral Interface (QSPI) is a synchronous serial data link that provides communication with external devices in Master mode.

The QSPI can be used in SPI mode to interface to serial peripherals such as ADCs, DACs, LCD controllers, CAN controllers and sensors, or in Serial Memory mode to interface to serial Flash memories.

The QSPI allows the system to execute code directly from a serial Flash memory (XIP) without code shadowing to RAM. The serial Flash memory mapping is seen in the system as other memories such as ROM, SRAM, DRAM, embedded Flash memory, etc.

With the support of the Quad SPI protocol, the QSPI allows the system to use high-performance serial Flash memories which are small and inexpensive, in place of larger and more expensive parallel Flash memories.

**Note:** Stacked devices with a rollover in the memory address space at each die boundary are not supported.

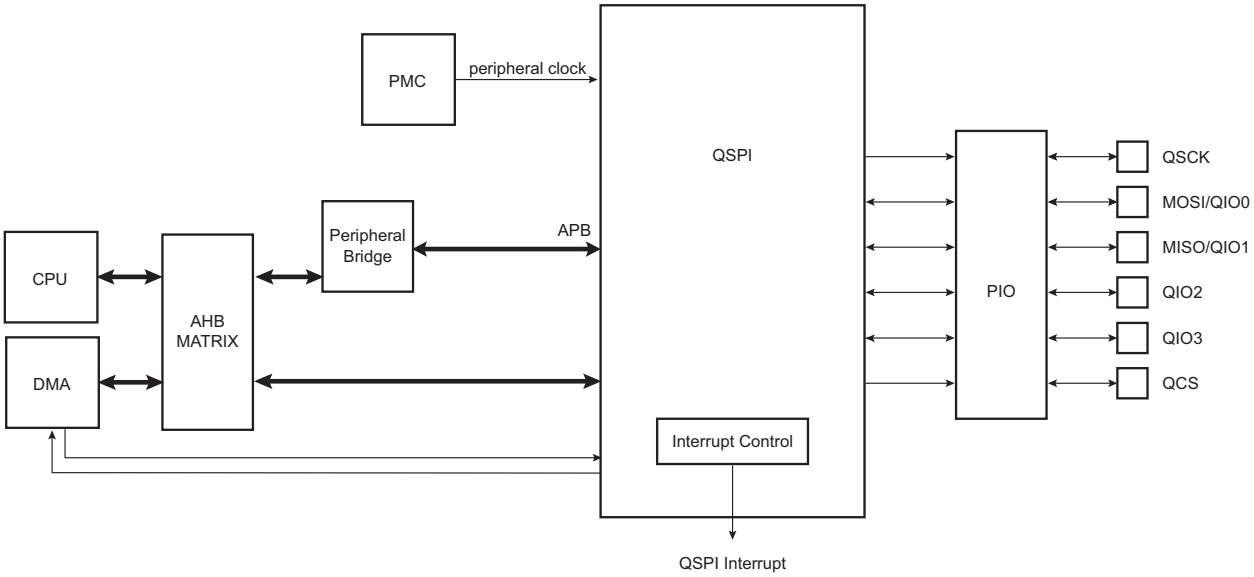
## 42.2 Embedded Characteristics

- Master SPI Interface
  - Programmable clock phase and clock polarity
  - Programmable transfer delays between consecutive transfers, between clock and data, between deactivation and activation of chip select
- SPI Mode
  - Interface to serial peripherals such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - 8-bit/16-bit programmable data length
- Serial Memory Mode
  - Interface to serial Flash memories operating in Single-bit SPI, Dual SPI and Quad SPI
  - Interface to serial Flash Memories operating in Single Data Rate Mode
  - Supports “Execute In Place” (XIP)— code execution by the system directly from a serial Flash memory
  - Flexible instruction register for compatibility with all serial Flash memories
  - 32-bit address mode (default is 24-bit address) to support serial Flash memories larger than 128 Mbits
  - Continuous read mode
  - Scrambling/unscrambling “On-The-Fly”
- Connection to DMA Channel Capabilities Optimizes Data Transfers
  - One channel for the receiver, one channel for the transmitter
- Register Write Protection

© 2021 Microchip Technology Inc.
and its subsidiaries

## 42.3 Block Diagram

**Figure 42-1. Block Diagram**




## 42.4 Signal Description

**Table 42-1. Signal Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| QSCK | Serial Clock | Output |
| MOSI (QIO0) (1)(2) | Data Output (Data Input Output 0) | Output (Input/Output) |
| MISO (QIO1) (1)(2) | Data Input (Data Input Output 1) | Input (Input/Output) |
| QIO2 (3) | Data Input Output 2 | Input/Output |
| QIO3 (3) | Data Input Output 3 | Input/Output |
| QCS | Peripheral Chip Select | Output |

**Notes:**

1. MOSI and MISO are used for single-bit SPI operation.
2. QIO0–QIO1 are used for Dual SPI operation.
3. QIO0–QIO3 are used for Quad SPI operation.

## 42.5 Product Dependencies

### 42.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the QSPI pins to their peripheral functions.

### 42.5.2 Power Management

The QSPI may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the QSPI clock.

### 42.5.3 Interrupt Sources

The QSPI has an interrupt line connected to the Interrupt Controller. Handling the QSPI interrupt requires programming the interrupt controller before configuring the QSPI.

### 42.5.4 Direct Memory Access Controller (DMA)

The QSPI can be used in conjunction with the Direct Memory Access Controller (DMA) in order to reduce processor overhead. For a full description of the DMA, refer to the section “DMA Controller (XDMAC)”.

**Note:** DMA write accesses must be 32-bit aligned. If a single byte is to be written in a 32-bit word, the rest of the word must be filled with ones.

## 42.6 Functional Description

### 42.6.1 Serial Clock Baud Rate

The QSPI baud rate clock is generated by dividing the peripheral clock by a value between 1 and 256.

### 42.6.2 Serial Clock Phase and Polarity

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the QSPI Serial Clock register (QSPI_SCR). The CPHA bit in the QSPI_SCR programs the clock phase. These two parameters determine the edges of the clock signal on which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, the interfaced slave must use the same parameter values to communicate.

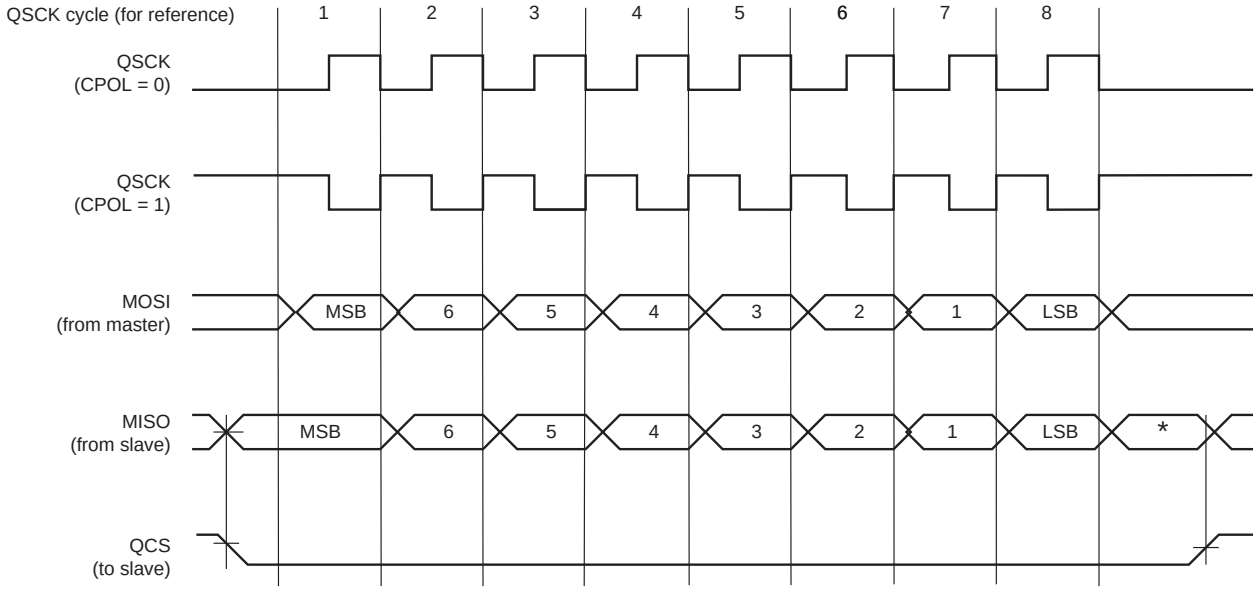
The table below shows the four modes and corresponding parameter settings.

**Table 42-2. QSPI Bus Clock Modes**

| QSPI Clock Mode | QSPI_SCR.CPOL | QSPI_SCR.CPHA | Shift QSCK Edge | Capture QSCK Edge | QSCK Inactive Level |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Falling | Rising | Low |
| 1 | 0 | 1 | Rising | Falling | Low |
| 2 | 1 | 0 | Rising | Falling | High |
| 3 | 1 | 1 | Falling | Rising | High |

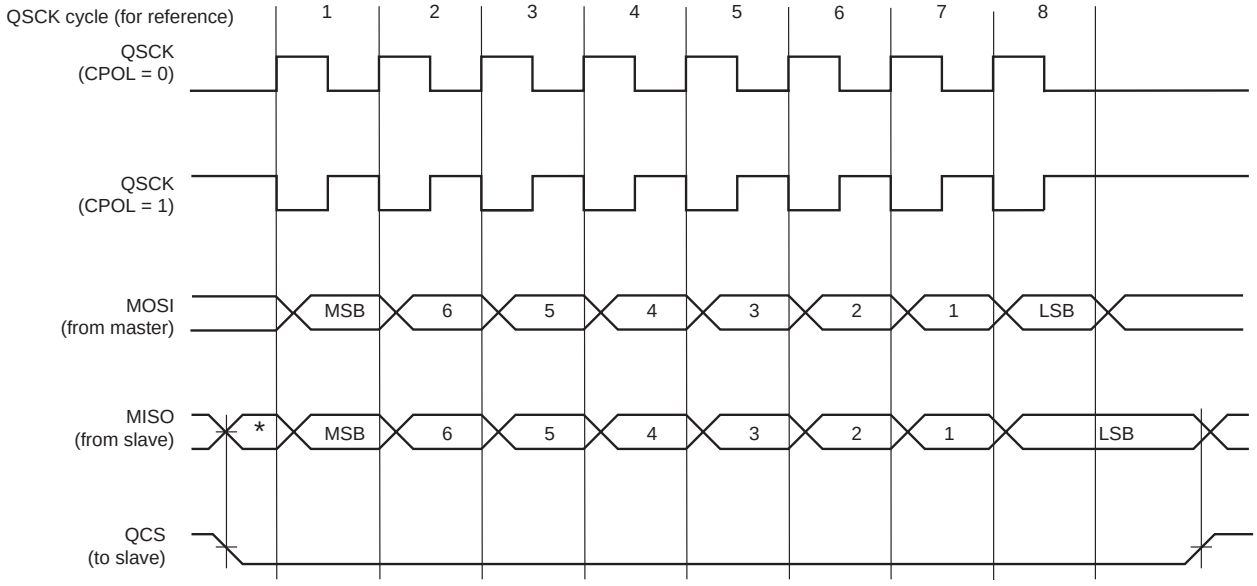
The following figures show examples of data transfers.

**Figure 42-2. QSPI Transfer Format (QSPI_SCR.CPHA = 0, 8 bits per transfer)**




* Not defined, but normally MSB of previous character received.

**Figure 42-3. QSPI Transfer Format (QSPI_SCR.CPHA = 1, 8 bits per transfer)**




* Not defined but normally LSB of previous character transmitted.

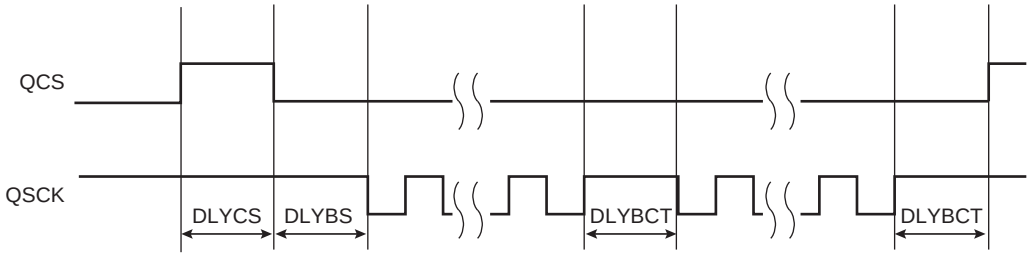
### 42.6.3 Transfer Delays

The figure below shows several consecutive transfers while the chip select is active. Three delays can be programmed to modify the transfer waveforms:

- The delay between the deactivation and the activation of QCS, programmed by writing QSPI_MR.DLYCS. Allows to adjust the minimum time of QCS at high level.
- The delay before QSCK, programmed by writing QSPI_SR.DLYBS. Allows the start of QSCK to be delayed after the chip select has been asserted.
- The delay between consecutive transfers, programmed by writing QSPI_MR.DLYBCT. Allows insertion of a delay between two consecutive transfers. In Serial Memory mode, this delay is not programmable and DLYBCT is ignored. In this mode, DLYBCT must be written to ‘0’.

These delays allow the QSPI to be adapted to the interfaced peripherals and their speed and bus release time.

**Figure 42-4. Programmable Delays**




### 42.6.4 QSPI SPI Mode

In SPI mode, the QSPI acts as a standard SPI Master.

To activate this mode, QSPI_MR.SMM must be written to ‘0’ in QSPI_MR.

#### 42.6.4.1 SPI Mode Operations

The QSPI in standard SPI mode operates on the clock generated by the internal programmable baud rate generator. It fully controls the data transfers to and from the slave connected to the SPI bus. The QSPI drives the chip select line to the slave (QCS) and the serial clock signal (QSCK).

The QSPI features two holding registers, the Transmit Data register (QSPI_TDR) and the Receive Data register (QSPI_RDR), and a single internal shift register. The holding registers maintain the data flow at a constant rate.

After enabling the QSPI, a data transfer begins when the processor writes to the QSPI_TDR. The written data is immediately transferred to the internal shift register and transfer on the SPI bus starts. While the data in the internal shift register is shifted on the MOSI line, the MISO line is sampled and shifted to the internal shift register. Receiving data cannot occur without transmitting data. If receiving mode is not needed, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the Status register (QSPI_SR) can be discarded.

If new data is written in QSPI_TDR during the transfer, it is retained there until the current transfer is completed. Then, the received data is transferred from the internal shift register to the QSPI_RDR, the data in QSPI_TDR is loaded in the internal shift register and a new transfer starts.

The transfer of a data written in QSPI_TDR in the internal shift register is indicated by the Transmit Data Register Empty (TDRE) bit in the QSPI_SR. When new data is written in QSPI_TDR, this bit is cleared. QSPI_SR.TDRE is used to trigger the Transmit DMA channel.

The end of transfer is indicated by the TXEMPTY flag in the QSPI_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, QSPI_SR.TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

The transfer of received data from the internal shift register in QSPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in the QSPI_SR. When the received data is read, QSPI_SR.RDRF bit is cleared.

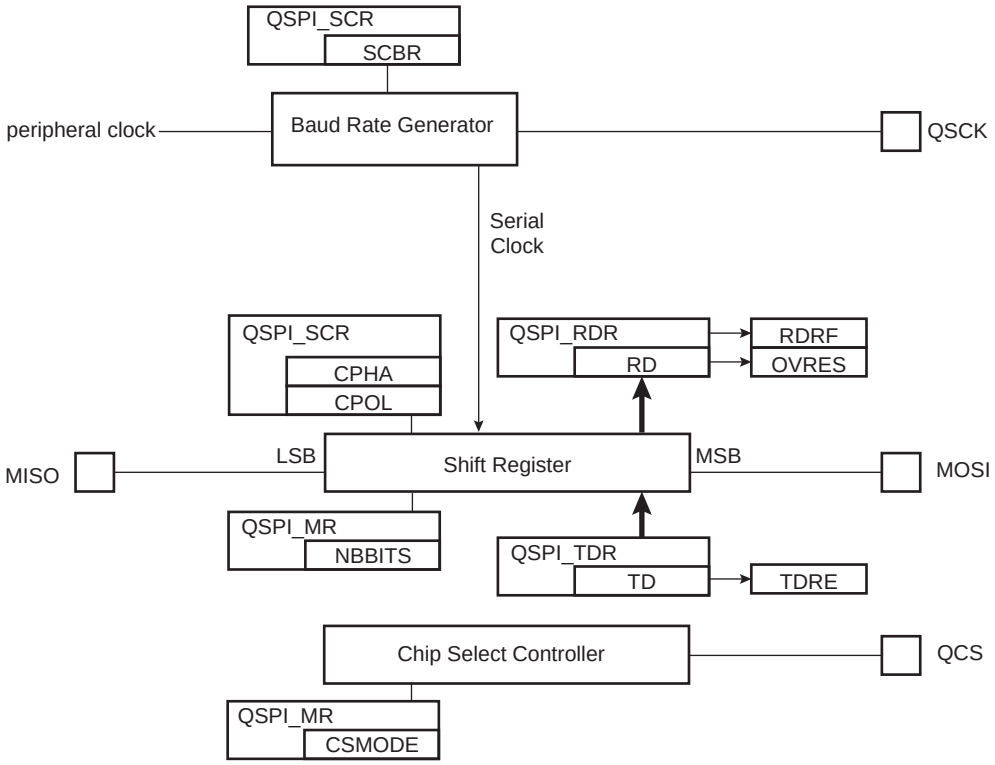
If the QSPI_RDR has not been read before new data is received, the Overrun Error Status (OVRES) bit in QSPI_SR is set. As long as this flag is set, data is loaded in QSPI_RDR. The user must read the QSPI_SR to clear the OVRES bit.

The following figures show, respectively, a block diagram of the SPI when operating in Master mode, and a flow chart describing how transfers are handled.

© 2021 Microchip Technology Inc.
and its subsidiaries

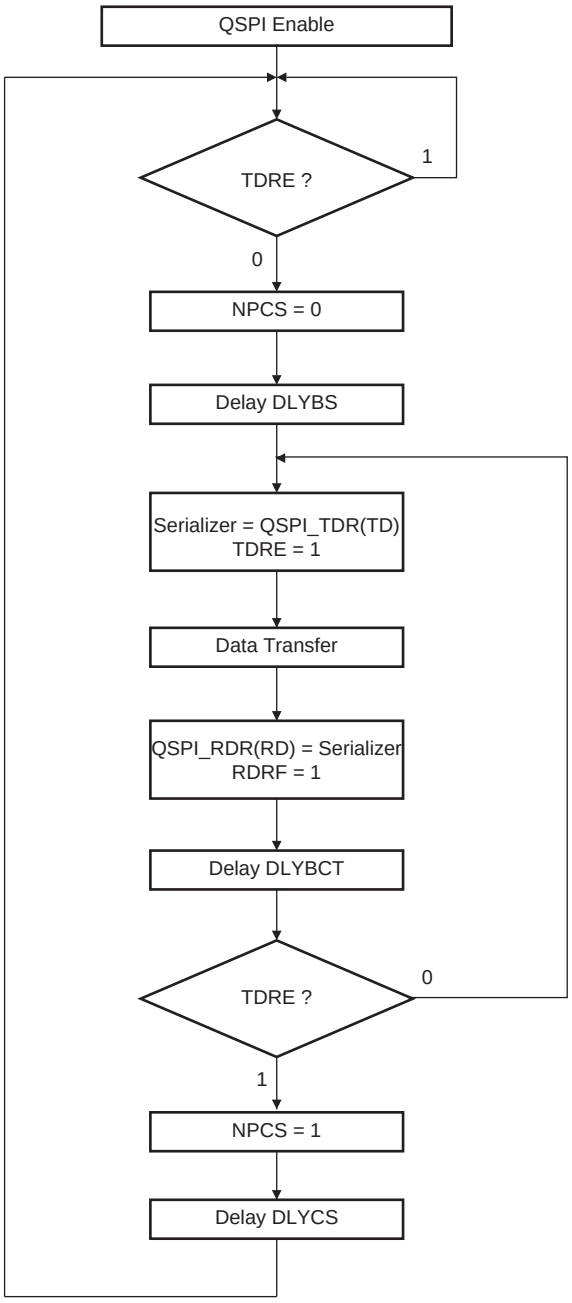
### 42.6.4.2 SPI Mode Block Diagram

**Figure 42-5. SPI Mode Block Diagram**



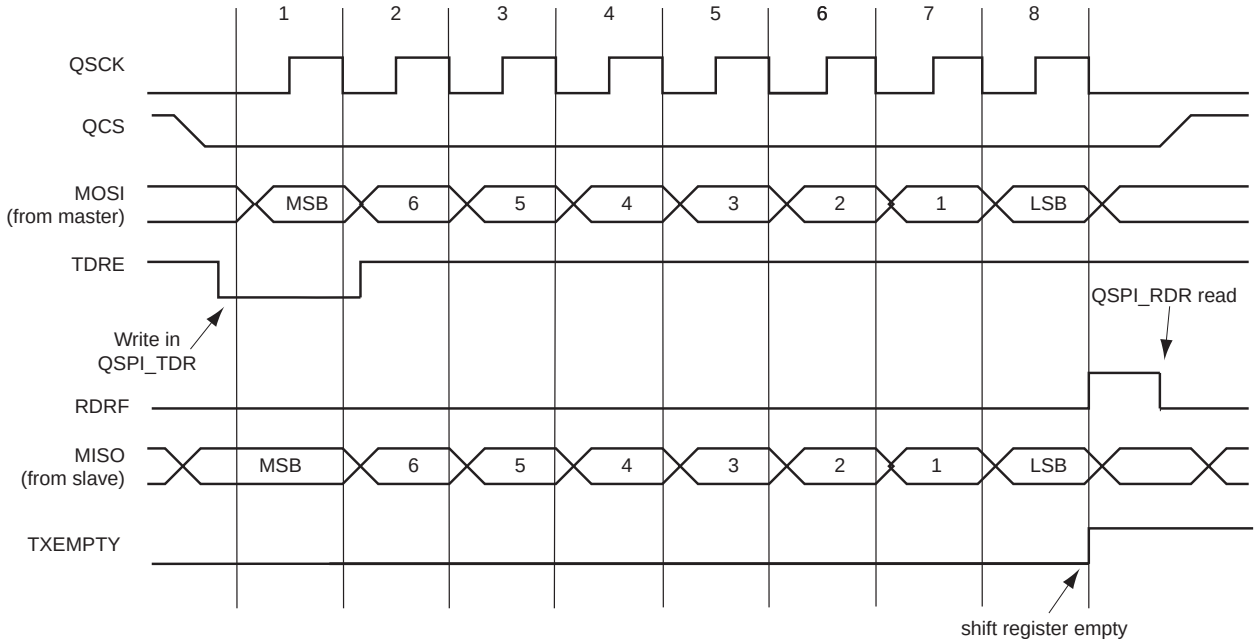
### 42.6.4.3 SPI Mode Flow Diagram

**Figure 42-6. SPI Mode Flow Diagram**




The figure below shows Transmit Data Register Empty (TDRE), Receive Data Register Full (RDRF) and Transmission Register Empty (TXEMPTY) status flags behavior within the QSPI_SR during an 8-bit data transfer in Fixed mode, without DMA.

**Figure 42-7. Status Register Flags Behavior**




### 42.6.4.4 Peripheral Deselection without DMA

During a transfer of more than one data on a Chip Select without the DMA, the QSPI_TDR is loaded by the processor and the flag TDRE rises as soon as the content of the QSPI_TDR is transferred into the internal shift register. When this flag is detected high, the QSPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer, the Chip Select is not deasserted between the two transfers. Depending on the application software handling the QSPI_SR flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload the QSPI_TDR in time to keep the chip select active (low). A null Delay Between Consecutive Transfer (DLYBCT) value in the QSPI_MR gives even less time for the processor to reload the QSPI_TDR. With some SPI slave peripherals, requiring the chip select line to remain active (low) during a full set of transfers may lead to communication errors.

To facilitate interfacing with such devices, QSPI_MR.CSMODE may be configured to ‘1’. This allows the chip select lines to remain in their current state (low = active) until the end of transfer is indicated by the Last Transfer (LASTXFER) bit in the Control register (QSPI_CR). Even if the QSPI_TDR is not reloaded, the chip select remains active. To have the chip select line rise at the end of the last data transfer, QSPI_CR.LASTXFER must be written to ‘1’ at the same time or after writing the last data to transmit into the QSPI_TDR.

### 42.6.4.5 Peripheral Deselection with DMA

When the DMA Controller is used, the Chip Select line remains low during the transfer since the TDRE flag is managed by the DMA itself. Reloading the QSPI_TDR by the DMA is done as soon as the TDRE flag is set. In this case, writing QSPI_MR.CSMODE to ‘1’ may not be needed. However, when other DMA channels connected to other peripherals are also in use, the QSPI DMA could be delayed by another DMA with a higher priority on the bus. Having DMA buffers in slower memories like Flash memory or SDRAM compared to fast internal SRAM, may lengthen the reload time of the QSPI_TDR by the DMA as well. This means that the QSPI_TDR might not be reloaded in time to keep the chip select line low. In this case, the chip select line may toggle between data transfer and according to some SPI Slave devices, the communication might get lost. It may be necessary to configure QSPI_MR.CSMODE to ‘1’.

When QSPI_MR.CSMODE is configured to ‘0’, the QCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a Chip Select, the flag TDRE rises as soon as the content of the QSPI_TDR is transferred into the internal shifter. When this flag is detected, the QSPI_TDR can be reloaded. If this reload occurs before the end of the current transfer, the Chip Select is not deasserted between the two transfers. This might lead to difficulties for interfacing with some serial peripherals requiring the chip select to be deasserted after each transfer. To facilitate interfacing with such devices, the QSPI_MR may be configured with QSPI_MR.CSMODE at ‘2’.

### 42.6.5 QSPI Serial Memory Mode

In Serial Memory mode, the QSPI acts as a serial Flash memory controller. The QSPI can be used to read data from the serial Flash memory allowing the CPU to execute code from it (XIP execute in place). The QSPI can also be used to control the serial Flash memory (Program, Erase, Lock, etc.) by sending specific commands. In this mode, the QSPI is compatible with single-bit SPI, Dual SPI and Quad SPI protocols.

To activate this mode, QSPI_MR.SMM must be written to '1'.

In Serial Memory mode, data is transferred only by writing or reading the QSPI memory space (0x80000000).

#### 42.6.5.1 Instruction Frame

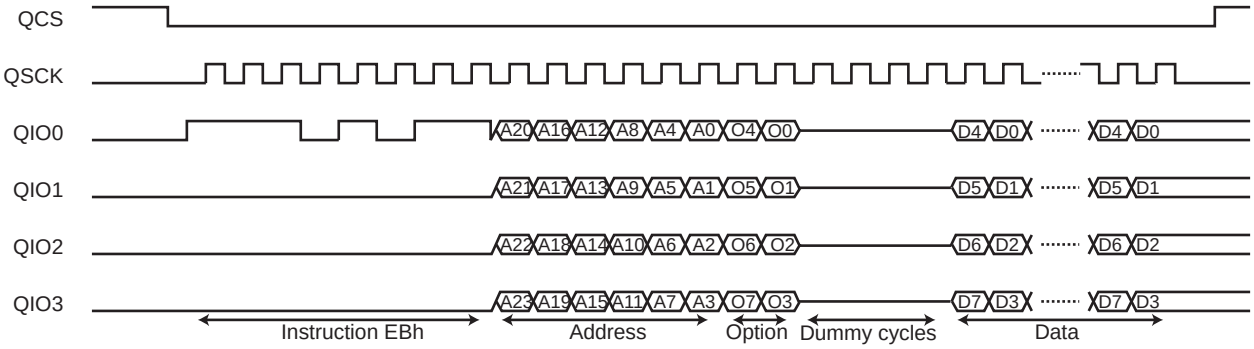
In order to control serial Flash memories, the QSPI is able to send instructions via the SPI bus (ex: READ, PROGRAM, ERASE, LOCK, etc.). Because the instruction set implemented in serial Flash memories is memory vendor-dependent, the QSPI includes a complete Instruction Frame register (QSPI_IFR), which makes it very flexible and compatible with all serial Flash memories.

An instruction frame includes:

- An instruction code (size: 8 bits). The instruction is optional in some cases (see section Continuous Read mode).
- An address (size: 24 bits or 32 bits). The address is optional but is required by instructions such as READ, PROGRAM, ERASE, LOCK. By default the address is 24 bits long, but it can be 32 bits long to support serial Flash memories larger than 128 Mbits (16 Mbytes).
- An option code (size: 1/2/4/8 bits). The option code is not required, but it is useful to activate the XIP mode or the Continuous Read mode (see section Continuous Read mode) for READ instructions, in some serial Flash memory devices. These modes improve the data read latency.
- Dummy cycles. Dummy cycles are optional but required by some READ instructions.
- Data bytes are optional. Data bytes are present for data transfer instructions such as READ or PROGRAM.

The instruction code, the address/option and the data can be sent with Single-bit SPI, Dual SPI or Quad SPI protocols.

**Figure 42-8. Instruction Frame**




#### 42.6.5.2 Instruction Frame Transmission

To send an instruction frame, the user must first configure the address to send by writing the field ADDR in the Instruction Address register (QSPI_IAR). This step is required if the instruction frame includes an address and no data. When data is present, the address of the instruction is defined by the address of the data accesses in the QSPI memory space, not by QSPI_IAR.

If the instruction frame includes the instruction code and/or the option code, the user must configure the instruction code and/or the option code to send by writing the fields INST and OPT in the Instruction Code register (QSPI_ICR).

Then, the user must write QSPI_IFR to configure the instruction frame depending on which instruction must be sent. If the instruction frame does not include data, writing in this register triggers the send of the instruction frame in the QSPI. If the instruction frame includes data, the send of the instruction frame is triggered by the first data access in the QSPI memory space.

The instruction frame is configured by the following bits and fields of QSPI_IFR:

- WIDTH field—used to configure which data lanes are used to send the instruction code, the address, the option code and to transfer the data. It is possible to use two unidirectional data lanes (MISO-MOSI Single-bit SPI), two bidirectional data lanes (QIO0-QIO1 Dual SPI) or four bidirectional data lanes (QIO0–QIO3 Quad SPI).
- INSTEN bit—used to enable the send of an instruction code.
- ADDREN bit—used to enable the send of an address after the instruction code.
- OPTEN bit—used to enable the send of an option code after the address.
- DATAEN bit—used to enable the transfer of data (READ or PROGRAM instruction).
- OPTL field—used to configure the option code length. The value written in OPTL must be consistent with the value written in the field WIDTH. For example: OPTL = 0 (1-bit option code) is not consistent with WIDTH = 6 (option code sent with QuadSPI protocol, thus the minimum length of the option code is 4 bits).
- ADDRL bit—used to configure the address length.
- TFRTYP field—used to define which type of data transfer must be performed.
- NBDUM field—used to configure the number of dummy cycles when reading data from the serial Flash memory. Between the address/option and the data, with some instructions, dummy cycles are inserted by the serial Flash memory.

Refer to 42.6.5.2 Instruction Frame Transmission.

If data transfer is enabled, the user can access the serial memory by reading or writing the QSPI memory space:

- To read in the serial memory, but not a memory data, for example a JEDEC-ID or the QSPI_SR, QSPI_IFR.TFRTYP must be written to ‘0’.
- To read in the serial memory, and particularly a memory data, TFRTYP must be written to ‘1’.
- To write in the serial memory, but not a memory data, for example writing the configuration or the QSPI_SR, TFRTYP must be written to ‘2’.
- If the user wants to write in the serial memory in particular to program a memory data, TFRTYP must be written to ‘3’ .

If QSPI_IFR.TFRTYP has a value other than ‘1’, the address sent in the instruction frame is the address of the first system bus accesses. The addresses of the next accesses are not used by the QSPI. At each system bus access, an SPI transfer is performed with the same size. For example, a halfword system bus access leads to a 16-bit SPI transfer, and a byte system bus access leads to an 8-bit SPI transfer.

If TFRTYP = 1, the address of the first instruction frame is the one of the first read access in the QSPI memory space. Each time the read accesses become nonsequential (addresses are not consecutive), a new instruction frame is sent with the last system bus access address. In this way, the system can read data at a random location in the serial memory. The size of the SPI transfers may differ from the size of the system bus read accesses.

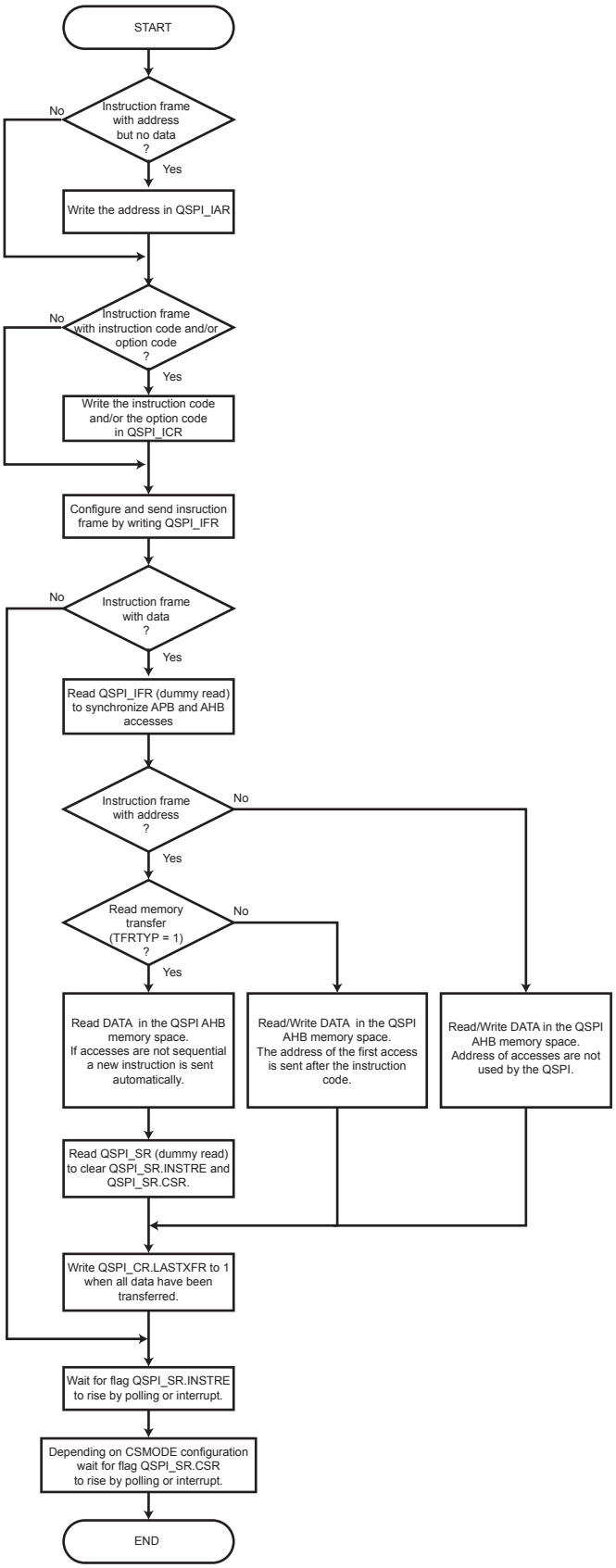
When data transfer is not enabled, the end of the instruction frame is indicated when QSPI_SR.INSTRE rises. (The QSPI_SR.CSR flag indicates when chip select rises. A delay between these flags may exist in case of high clock division or a high DLYBCT value).

When data transfer is enabled, the user must indicate when the data transfer is completed in the QSPI memory space by setting QSPI_CR.LASTXFR. The end of the instruction frame is indicated when QSPI_SR.INSTRE rises.

The following figure illustrates instruction transmission management.

© 2021 Microchip Technology Inc. and its subsidiaries

**Figure 42-9. Instruction Transmission Flow Diagram**



### 42.6.5.3 Read Memory Transfer

The user can access the data of the serial memory by sending an instruction with QSPI_IFR.DATAEN = 1 and QSPI_IFR.TFRTYP = 1.

In this mode, the QSPI is able to read data at random address into the serial Flash memory, allowing the CPU to execute code directly from it (XIP execute-in-place).

In order to fetch data, the user must first configure the instruction frame by writing the QSPI_IFR. Then data can be read at any address in the QSPI address space mapping. The address of the system bus read accesses match the address of the data inside the serial Flash memory.

When Fetch mode is enabled, several instruction frames can be sent before writing QSPI_CR.LASTXFR. Each time the system bus read accesses become nonsequential (addresses are not consecutive), a new instruction frame is sent with the corresponding address.

### 42.6.5.4 Continuous Read Mode

The QSPI is compatible with the Continuous Read mode which is implemented in some serial Flash memories.

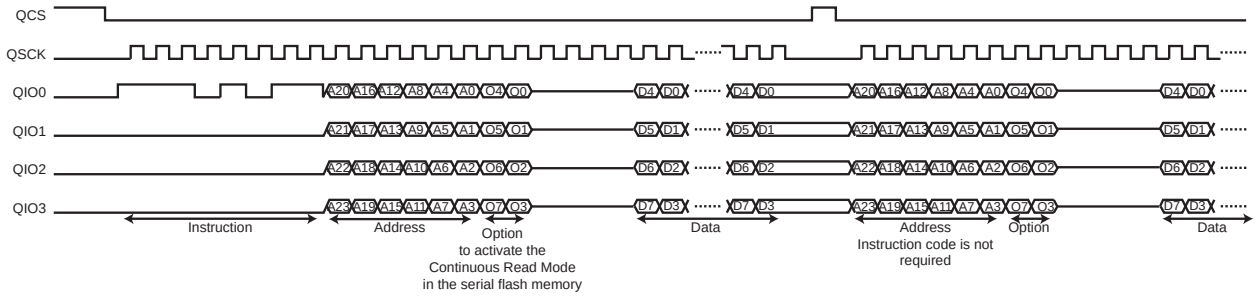
In Continuous Read mode, the instruction overhead is reduced by excluding the instruction code from the instruction frame. When the Continuous Read mode is activated in a serial Flash memory by a specific option code, the instruction code is stored in the memory. For the next instruction frames, the instruction code is not required as the memory uses the stored one.

In the QSPI, Continuous Read mode is used when reading data from the memory (QSPI_IFR.TFRTYP = 1). The addresses of the system bus read accesses are often nonsequential and this leads to many instruction frames that have the same instruction code. By disabling the send of the instruction code, the Continuous Read mode reduces the access time of the data.

To be functional, this mode must be enabled in both the QSPI and the serial Flash memory. The Continuous Read mode is enabled in the QSPI by writing CRM to ‘1’ in the QSPI_IFR (TFRTYP must equal 1). The Continuous Read mode is enabled in the serial Flash memory by sending a specific option code.

**⚠ CAUTION** If the Continuous Read mode is not supported by the serial Flash memory or disabled, CRM bit must not be written to ‘1’, otherwise data read out of the serial Flash memory is unpredictable.

**Figure 42-10. Continuous Read Mode**




### 42.6.5.5 Instruction Frame Transmission Examples

All waveforms in the following examples describe SPI transfers in SPI Clock mode 0 (QSPI_SCR.CPOL = 0 and QSPI_SCR.CPHA = 0; see section Serial Clock Phase and Polarity).

All system bus accesses described below refer to the system bus address phase. System bus wait cycles and system bus data phases are not shown.

Example 1:

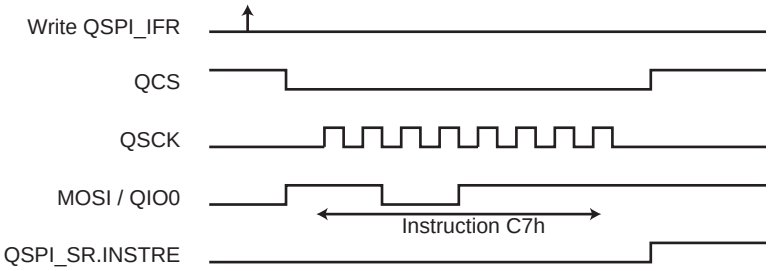
Instruction in Single-bit SPI, without address, without option, without data.

Command: CHIP ERASE (C7h).

- Write 0x0000_00C7 in QSPI_ICR.
- Write 0x0000_0010 in QSPI_IFR.

- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-11. Instruction Transmission Waveform 1**



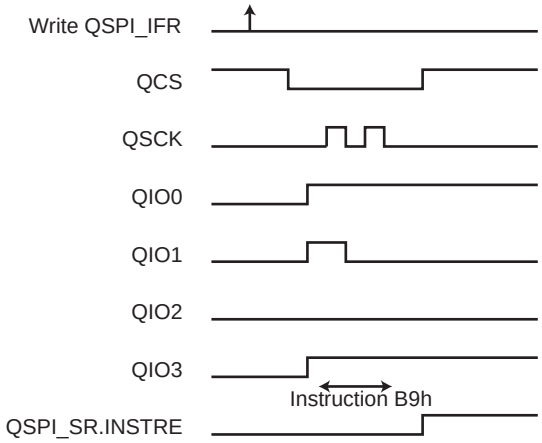

Example 2:

Instruction in Quad SPI, without address, without option, without data.

Command: POWER DOWN (B9h)

- Write 0x0000_00B9 in QSPI_ICR.
- Write 0x0000_0016 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-12. Instruction Transmission Waveform 2**




Example 3:

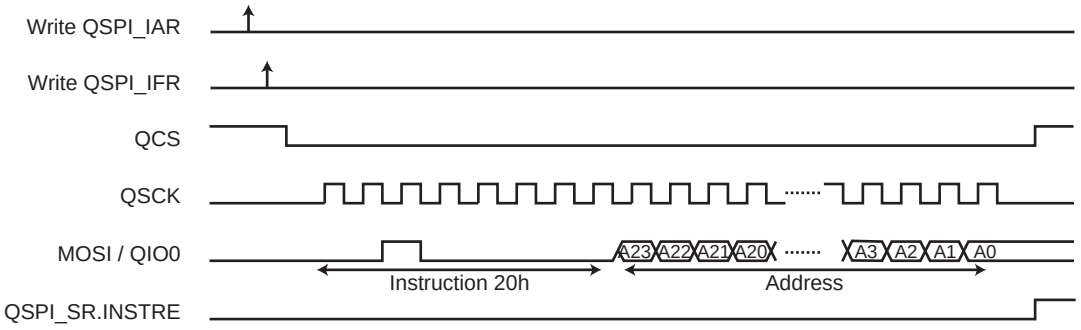
Instruction in Single-bit SPI, with address in Single-bit SPI, without option, without data.

Command: BLOCK ERASE (20h)

- Write the address (of the block to erase) in QSPI_AR.
- Write 0x0000_0020 in QSPI_ICR.
- Write 0x0000_0030 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 42-13. Instruction Transmission Waveform 3**



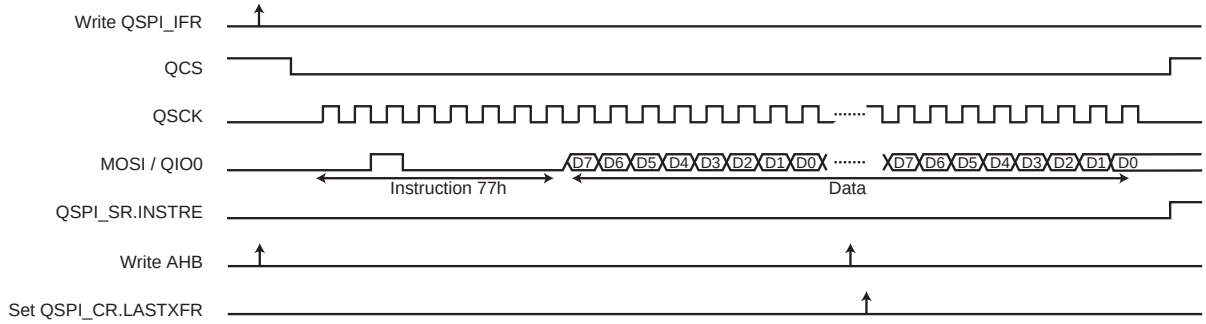

Example 4:

Instruction in Single-bit SPI, without address, without option, with data write in Single-bit SPI.

Command: SET BURST (77h)

- Write 0x0000_0077 in QSPI_ICR.
- Write 0x0000_2090 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Write data in the system bus memory space (0x80000000).
  The address of system bus write accesses is not used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-14. Instruction Transmission Waveform 4**



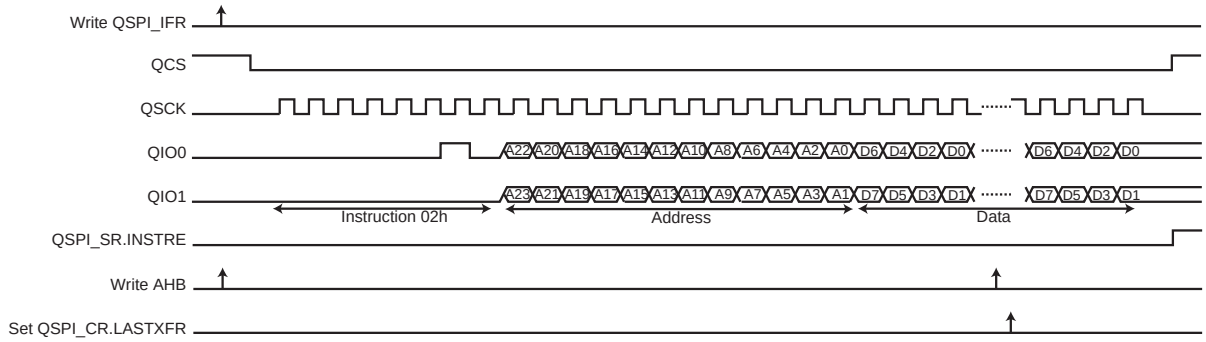

Example 5:

Instruction in Single-bit SPI, with address in Dual SPI, without option, with data write in Dual SPI.

Command: BYTE/PAGE PROGRAM (02h)

- Write 0x0000_0002 in QSPI_ICR.
- Write 0x0000_30B3 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Write data in the QSPI system bus memory space (0x80000000).
  The address of the first system bus write access is sent in the instruction frame.
  The address of the next system bus write accesses is not used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-15. Instruction Transmission Waveform 5**



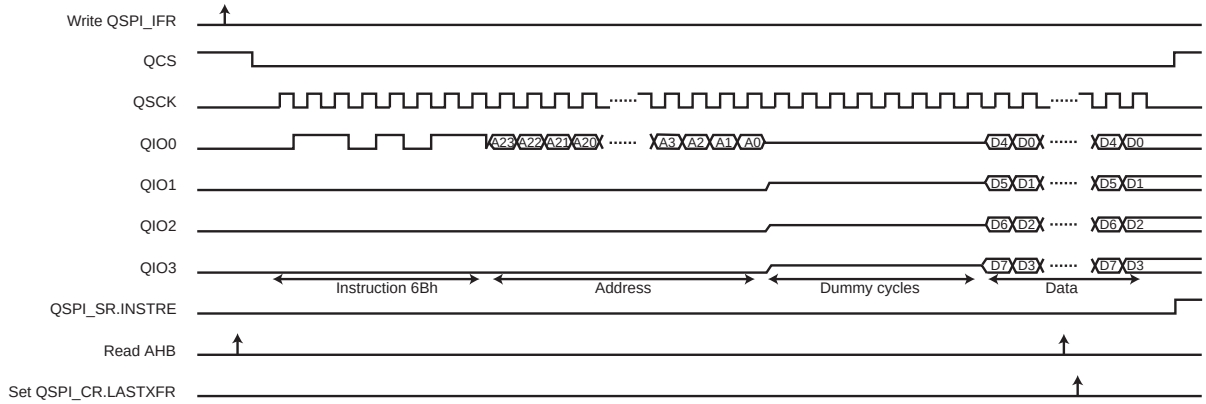

Example 6:

Instruction in Single-bit SPI, with address in Single-bit SPI, without option, with data read in Quad SPI, with eight dummy cycles.

Command: QUAD_OUTPUT READ ARRAY (6Bh)

- Write 0x0000_006B in QSPI_ICR.
- Write 0x0008_10B2 in QSPI_IFR.
- Read QSPI_IR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x80000000).
  The address of the first system bus read access is sent in the instruction frame.
  The address of the next system bus read accesses is not used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-16. Instruction Transmission Waveform 6**



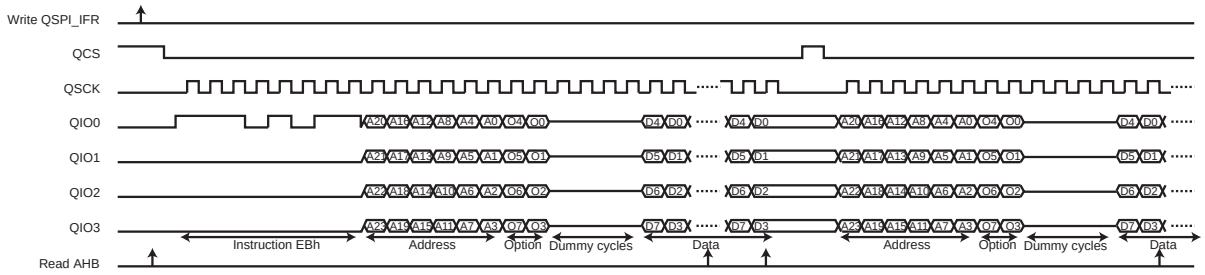

Example 7:

Instruction in Single-bit SPI, with address and option in Quad SPI, with data read in Quad SPI, with four dummy cycles, with fetch and continuous read.

Command: FAST READ QUAD I/O (EBh) - 8-BIT OPTION (0x30h)

- Write 0x0030_00EB in QSPI_ICR.
- Write 0x0004_33F4 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x80000000).
  Fetch is enabled, the address of the system bus read accesses is always used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-17. Instruction Transmission Waveform 7**



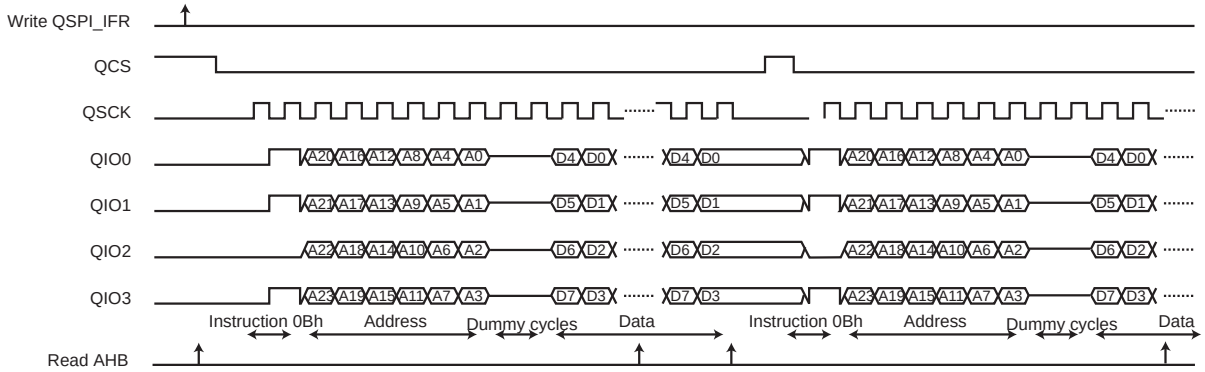

Example 8:

Instruction in Quad SPI, with address in Quad SPI, without option, with data read in Quad SPI, with two dummy cycles, with fetch.

Command: HIGH-SPEED READ (0Bh)

- Write 0x0000_000B in QSPI_ICR.
- Write 0x0002_20B6 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x80000000).
  Fetch is enabled, the address of the system bus read accesses is always used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-18. Instruction Transmission Waveform 8**



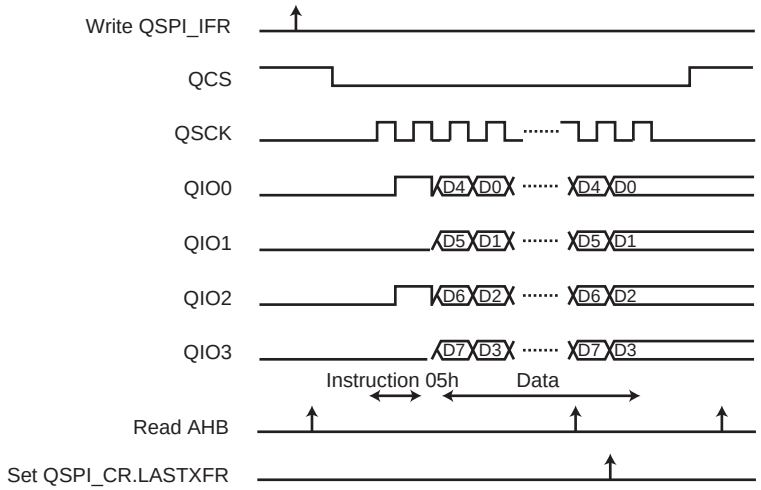

Example 9:

Instruction in Quad SPI, without address, without option, with data read in Quad SPI, without dummy cycles, without fetch.

Command: HIGH-SPEED READ (05h)

- Write 0x0000_0005 in QSPI_ICR.
- Write 0x0000_0096 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x80000000).
  Fetch is disabled.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-19. Instruction Transmission Waveform 9**



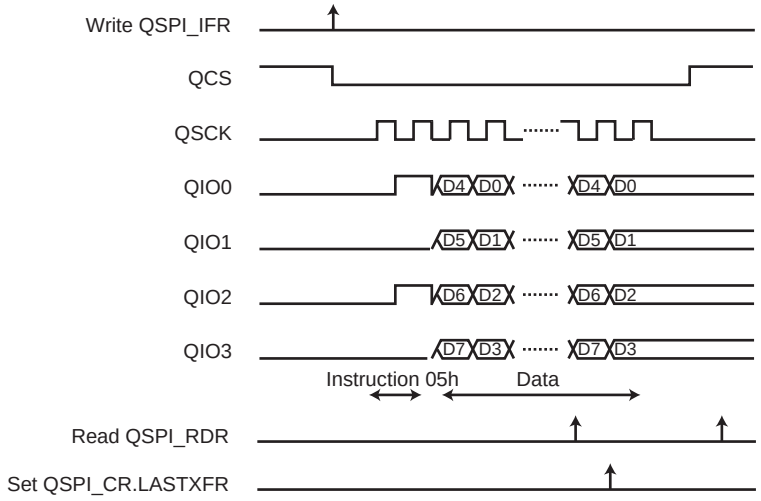

Example 10:

Instruction in Quad SPI, without address, without option, with data read in Quad SPI, without dummy cycles, without fetch, read launched through APB interface.

Command: HIGH-SPEED READ (05h)

- Set SMRM to ‘1’ in QSPI_MR
- Write 0x0000_0005 in QSPI_ICR.
- Write 0x0100_0096 in QSPI_IFR (will start the transfer).
- Wait flag RDRF and Read data in the QSPI_RDR register
  Fetch is disabled.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

**Figure 42-20. Instruction Transmission Waveform 10**




### 42.6.6 Scrambling/Unscrambling Function

The scrambling/unscrambling function cannot be performed on devices other than memories. Data is scrambled when written to memory and unscrambled when data is read.

The external data lines can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either the microcontroller or the QSPI slave device (e.g., memory).

The scrambling/unscrambling function can be enabled by writing a ‘1’ to the SCREN bit in the QSPI Scrambling Mode Register (QSPI_SMR).

The scrambling and unscrambling are performed on-the-fly without impacting the throughput.

The scrambling method depends on the user-configurable user scrambling key (field USRK) in the QSPI Scrambling Key Register (QSPI_SKR). QSPI_SKR is only accessible in Write mode.

If QSPI_SMR.RVDIS is written to ‘0’, the scrambling/unscrambling algorithm includes the user scrambling key plus a random value depending on device processing characteristics. Data scrambled by a given microcontroller cannot be unscrambled by another.

If QSPI_SMR.RVDIS is written to ‘1’, the scrambling/unscrambling algorithm includes only the user scrambling key. No random value is part of the key.

The user scrambling key or the seed for key generation must be securely stored in a reliable nonvolatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

### 42.6.7 Register Write Protection

To prevent any single software error from corrupting QSPI behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the QSPI Write Protection Mode Register (QSPI_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the QSPI Write Protection Status Register (QSPI_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the QSPI_WPSR.

The following registers can be write-protected when WPEN is set in QSPI_WPMR:

- QSPI Mode Register
- QSPI Serial Clock Register
- QSPI Scrambling Mode Register
- QSPI Scrambling Key Register

## 42.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | QSPI_CR | 7:0 | SWRST | | | | | | QSPIDIS | QSPIEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | LASTXFER |
| 0x04 | QSPI_MR | 7:0 | | | CSMODE[1:0] | | | WDRBT | LLB | SMM |
| | | 15:8 | | | | | NBBITS[3:0] | | | |
| | | 23:16 | DLYBCT[7:0] | | | | | | | |
| | | 31:24 | DLYCS[7:0] | | | | | | | |
| 0x08 | QSPI_RDR | 7:0 | RD[7:0] | | | | | | | |
| | | 15:8 | RD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | QSPI_TDR | 7:0 | TD[7:0] | | | | | | | |
| | | 15:8 | TD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | QSPI_SR | 7:0 | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| | | 15:8 | | | | | | INSTRE | CSS | CSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | QSPIENS |
| 0x14 | QSPI_IER | 7:0 | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| | | 15:8 | | | | | | INSTRE | CSS | CSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | QSPI_IDR | 7:0 | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| | | 15:8 | | | | | | INSTRE | CSS | CSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | QSPI_IMR | 7:0 | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| | | 15:8 | | | | | | INSTRE | CSS | CSR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | QSPI_SCR | 7:0 | | | | | | | CPHA | CPOL |
| | | 15:8 | SCBR[7:0] | | | | | | | |
| | | 23:16 | DLYBS[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | QSPI_IAR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | ADDR[15:8] | | | | | | | |
| | | 23:16 | ADDR[23:16] | | | | | | | |
| | | 31:24 | ADDR[31:24] | | | | | | | |
| 0x34 | QSPI_ICR | 7:0 | INST[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | OPT[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | QSPI_IFR | 7:0 | DATAEN | OPTEN | ADDREN | INSTEN | | WIDTH[2:0] | | |
| | | 15:8 | | CRM | TFRTYP[1:0] | | | ADDRL | OPTL[1:0] | |
| | | 23:16 | | | | NBDUM[4:0] | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C ... 0x3F | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x40 | QSPI_SMR | 7:0 | | | | | | | RVDIS | SCREN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | QSPI_SKR | 7:0 | USRK[7:0] | | | | | | | |
| | | 15:8 | USRK[15:8] | | | | | | | |
| | | 23:16 | USRK[23:16] | | | | | | | |
| | | 31:24 | USRK[31:24] | | | | | | | |
| 0x48 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | QSPI_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | QSPI_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 42.7.1 QSPI Control Register

**Name:** QSPI_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | LASTXFER |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | | | | | QSPIDIS | QSPIEN |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

**Bit 24 – LASTXFER** Last Transfer

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The chip select is deasserted after the character written in QSPI_TDR.TD has been transferred. |

**Bit 7 – SWRST** QSPI Software Reset
DMA channels are not affected by software reset.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Reset the QSPI. A software-triggered hardware reset of the QSPI interface is performed. |

**Bit 1 – QSPIDIS** QSPI Disable
As soon as QSPIDIS is set, the QSPI finishes its transfer.
All pins are set in Input mode and no data is received or transmitted.
If a transfer is in progress, the transfer is finished before the QSPI is disabled.
If both QSPIEN and QSPIDIS are equal to one when QSPI_CR is written, the QSPI is disabled.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the QSPI. |

**Bit 0 – QSPIEN** QSPI Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the QSPI to transfer and receive data. |

### 42.7.2 QSPI Mode Register

**Name:** QSPI_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DLYCS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DLYBCT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | NBBITS[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | CSMODE[1:0] | | | WDRBT | LLB | SMM |
| Access | | | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | 0 | 0 | | 0 | 0 | 0 |

**Bits 31:24 – DLYCS[7:0]** Minimum Inactive QCS Delay
This field defines the minimum delay between the deactivation and the activation of QCS. The DLYCS time guarantees the slave minimum deselect time.
If DLYCS written to '0', one peripheral clock period is inserted by default.
Otherwise, the following equation determines the delay:
DLYCS = Minimum inactive × $f_{peripheral\ clock}$

**Bits 23:16 – DLYBCT[7:0]** Delay Between Consecutive Transfers
This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.
When DLYBCT is written to '0', no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers. In Serial Memory mode (SMM = 1), DLYBCT must be written to '0' and no delay is inserted.
Otherwise, the following equation determines the delay:
DLYBCT = (Delay Between Consecutive Transfers × $f_{peripheral\ clock}$) / 32

**Bits 11:8 – NBBITS[3:0]** Number Of Bits Per Transfer

| Value | Name | Description |
|---|---|---|
| 0 | 8_BIT | 8 bits for transfer |
| 8 | 16_BIT | 16 bits for transfer |

**Bits 5:4 – CSMODE[1:0]** Chip Select Mode
The CSMODE field determines how the chip select is deasserted
**Note:** This field is forced to LASTXFER when SMM is written to '1'.

| Value | Name | Description |
|---|---|---|
| 0 | NOT_RELOADED | The chip select is deasserted if QSPI_TDR.TD has not been reloaded before the end of the current transfer. |

| Value | Name | Description |
|---|---|---|
| 1 | LASTXFER | The chip select is deasserted when the bit LASTXFER is written to ‘1’ and the character written in QSPI_TDR.TD has been transferred. |
| 2 | SYSTEMATICALLY | The chip select is deasserted systematically after each transfer. |

**Bit 2 – WDRBT** Wait Data Read Before Transfer
0 (DISABLED): No effect. In SPI mode, a transfer can be initiated whatever the state of the QSPI_RDR is.
1 (ENABLED): In SPI mode, a transfer can start only if the QSPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception.

**Bit 1 – LLB** Local Loopback Enable
0 (DISABLED): Local loopback path disabled.
1 (ENABLED): Local loopback path enabled.
LLB controls the local loopback on the data serializer for testing in SPI mode only. (MISO is internally connected on MOSI).

**Bit 0 – SMM** Serial Memory Mode
0 (SPI): The QSPI is in SPI mode.
1 (MEMORY): The QSPI is in Serial Memory mode.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 42.7.3 QSPI Receive Data Register

**Name:** QSPI_RDR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RD[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RD[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RD[15:0]** Receive Data
Data received by the QSPI is stored in this register right-justified. Unused bits read zero.

### 42.7.4 QSPI Transmit Data Register

**Name:** QSPI_TDR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 15:0 – TD[15:0]** Transmit Data
Data to be transmitted by the QSPI is stored in this register. Information to be transmitted must be written to the Transmit Data register in a right-justified format.

### 42.7.5 QSPI Status Register

**Name:** QSPI_SR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | QSPIENS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 24 – QSPIENS** QSPI Enable Status

| Value | Description |
|---|---|
| 0 | QSPI is disabled. |
| 1 | QSPI is enabled. |

**Bit 10 – INSTRE** Instruction End Status (cleared on read)

| Value | Description |
|---|---|
| 0 | No instruction end has been detected since the last read of QSPI_SR. |
| 1 | At least one instruction end has been detected since the last read of QSPI_SR. |

**Bit 9 – CSS** Chip Select Status

| Value | Description |
|---|---|
| 0 | The chip select is asserted. |
| 1 | The chip select is not asserted. |

**Bit 8 – CSR** Chip Select Rise (cleared on read)

| Value | Description |
|---|---|
| 0 | No chip select rise has been detected since the last read of QSPI_SR. |
| 1 | At least one chip select rise has been detected since the last read of QSPI_SR. |

**Bit 3 – OVRES** Overrun Error Status (cleared on read)
An overrun occurs when QSPI_RDR is loaded at least twice from the serializer since the last read of the QSPI_RDR.

| Value | Description |
|---|---|
| 0 | No overrun has been detected since the last read of QSPI_SR. |
| 1 | At least one overrun error has occurred since the last read of QSPI_SR. |

**Bit 2 – TXEMPTY** Transmission Registers Empty (cleared by writing QSPI_TDR)

| Value | Description |
|---|---|
| 0 | As soon as data is written in QSPI_TDR. |

| Value | Description |
|---|---|
| 1 | QSPI_TDR and the internal shifter are empty. If a transfer delay has been defined, TXEMPTY is set after the completion of such delay. |

**Bit 1 – TDRE** Transmit Data Register Empty (cleared by writing QSPI_TDR)
TDRE equals zero when the QSPI is disabled or at reset. The QSPI enable command sets this bit to one.

| Value | Description |
|---|---|
| 0 | Data has been written to QSPI_TDR and not yet transferred to the serializer. |
| 1 | The last data written in the QSPI_TDR has been transferred to the serializer. |

**Bit 0 – RDRF** Receive Data Register Full (cleared by reading QSPI_RDR)

| Value | Description |
|---|---|
| 0 | No data has been received since the last read of QSPI_RDR. |
| 1 | Data has been received and the received data has been transferred from the serializer to QSPI_RDR since the last read of QSPI_RDR. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 42.7.6 QSPI Interrupt Enable Register

**Name:** QSPI_IER
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 10 – INSTRE** Instruction End Interrupt Enable

**Bit 9 – CSS** Chip Select Status Interrupt Enable

**Bit 8 – CSR** Chip Select Rise Interrupt Enable

**Bit 3 – OVRES** Overrun Error Interrupt Enable

**Bit 2 – TXEMPTY** Transmission Registers Empty Enable

**Bit 1 – TDRE** Transmit Data Register Empty Interrupt Enable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Enable

### 42.7.7 QSPI Interrupt Disable Register

**Name:** QSPI_IDR
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 10 – INSTRE** Instruction End Interrupt Disable

**Bit 9 – CSS** Chip Select Status Interrupt Disable

**Bit 8 – CSR** Chip Select Rise Interrupt Disable

**Bit 3 – OVRES** Overrun Error Interrupt Disable

**Bit 2 – TXEMPTY** Transmission Registers Empty Disable

**Bit 1 – TDRE** Transmit Data Register Empty Interrupt Disable

**Bit 0 – RDRF** Receive Data Register Full Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 42.7.8 QSPI Interrupt Mask Register

**Name:** QSPI_IMR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | INSTRE | CSS | CSR |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRES | TXEMPTY | TDRE | RDRF |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 10 – INSTRE** Instruction End Interrupt Mask

**Bit 9 – CSS** Chip Select Status Interrupt Mask

**Bit 8 – CSR** Chip Select Rise Interrupt Mask

**Bit 3 – OVRES** Overrun Error Interrupt Mask

**Bit 2 – TXEMPTY** Transmission Registers Empty Mask

**Bit 1 – TDRE** Transmit Data Register Empty Interrupt Mask

**Bit 0 – RDRF** Receive Data Register Full Interrupt Mask

### 42.7.9 QSPI Serial Clock Register

**Name:** QSPI_SCR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DLYBS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SCBR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CPHA | CPOL |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 23:16 – DLYBS[7:0]** Delay Before QSCK
This field defines the delay from QCS valid to the first valid QSCK transition.
When DLYBS equals zero, the QCS valid to QSCK transition is 1/2 the QSCK clock period.
Otherwise, the following equation determines the delay:
DLYBS = Delay Before QSCK × $f_{peripheral\ clock}$

**Bits 15:8 – SCBR[7:0]** Serial Clock Baud Rate
The QSPI uses a modulus counter to derive the QSCK baud rate from the peripheral clock. The baud rate is selected by writing a value from 0 to 255 in the SCBR field. The following equation determines the QSCK baud rate:
SCBR = ($f_{peripheral\ clock}$ / QSCK Baudrate) - 1

**Bit 1 – CPHA** Clock Phase
CPHA determines which edge of QSCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | Data is captured on the leading edge of QSCK and changed on the following edge of QSCK. |
| 1 | Data is changed on the leading edge of QSCK and captured on the following edge of QSCK. |

**Bit 0 – CPOL** Clock Polarity
CPOL is used to determine the inactive state value of the serial clock (QSCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | The inactive state value of QSCK is logic level zero. |
| 1 | The inactive state value of QSCK is logic level one. |

### 42.7.10 QSPI Instruction Address Register

**Name:** QSPI_IAR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ADDR[31:0]** Address
Address to send to the serial Flash memory in the instruction frame.

### 42.7.11 QSPI Instruction Code Register

**Name:** QSPI_ICR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | OPT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | INST[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – OPT[7:0]** Option Code
Option code to send to the serial Flash memory.

**Bits 7:0 – INST[7:0]** Instruction Code
Instruction code to send to the serial Flash memory.

### 42.7.12 QSPI Instruction Frame Register

**Name:** QSPI_IFR
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | NBDUM[4:0] | | | | |
| Access | | | | | | | | |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | CRM | TFRTYP[1:0] | | | ADDRL | OPTL[1:0] | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATAEN | OPTEN | ADDREN | INSTEN | | WIDTH[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 20:16 – NBDUM[4:0]** Number Of Dummy Cycles
The NBDUM field defines the number of dummy cycles required by the serial Flash memory before data transfer.

**Bit 14 – CRM** Continuous Read Mode
0 (DISABLED): Continuous Read mode is disabled.
1 (ENABLED): Continuous Read mode is enabled.

**Bits 13:12 – TFRTYP[1:0]** Data Transfer Type

| Value | Name | Description |
|---|---|---|
| 0 | TRSFR_READ | Read transfer from the serial memory.<br>Scrambling is not performed.<br>Read at random location (fetch) in the serial Flash memory is not possible. |
| 1 | TRSFR_READ_MEMORY | Read data transfer from the serial memory.<br>If enabled, scrambling is performed.<br>Read at random location (fetch) in the serial Flash memory is possible. |
| 2 | TRSFR_WRITE | Write transfer into the serial memory.<br>Scrambling is not performed. |
| 3 | TRSFR_WRITE_MEMORY | Write data transfer into the serial memory.<br>If enabled, scrambling is performed. |

**Bit 10 – ADDRL** Address Length
The ADDRL bit determines the length of the address.
0 (24_BIT): The address is 24 bits long.
1 (32_BIT): The address is 32 bits long.

**Bits 9:8 – OPTL[1:0]** Option Code Length

The OPTL field determines the length of the option code. The value written in OPTL must be consistent with the value written in the field WIDTH. For example, OPTL = 0 (1-bit option code) is not consistent with WIDTH = 6 (option code sent with QuadSPI protocol, thus the minimum length of the option code is 4 bits).

| Value | Name | Description |
|---|---|---|
| 0 | OPTION_1BIT | The option code is 1 bit long. |
| 1 | OPTION_2BIT | The option code is 2 bits long. |
| 2 | OPTION_4BIT | The option code is 4 bits long. |
| 3 | OPTION_8BIT | The option code is 8 bits long. |

**Bit 7 – DATAEN** Data Enable

| Value | Description |
|---|---|
| 0 | No data is sent/received to/from the serial Flash memory. |
| 1 | Data is sent/received to/from the serial Flash memory. |

**Bit 6 – OPTEN** Option Enable

| Value | Description |
|---|---|
| 0 | The option is not sent to the serial Flash memory. |
| 1 | The option is sent to the serial Flash memory. |

**Bit 5 – ADDREN** Address Enable

| Value | Description |
|---|---|
| 0 | The transfer address is not sent to the serial Flash memory. |
| 1 | The transfer address is sent to the serial Flash memory. |

**Bit 4 – INSTEN** Instruction Enable

| Value | Description |
|---|---|
| 0 | The instruction is not sent to the serial Flash memory. |
| 1 | The instruction is sent to the serial Flash memory. |

**Bits 2:0 – WIDTH[2:0]** Width of Instruction Code, Address, Option Code and Data

| Value | Name | Description |
|---|---|---|
| 0 | SINGLE_BIT_SPI | Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Single-bit SPI |
| 1 | DUAL_OUTPUT | Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Dual SPI |
| 2 | QUAD_OUTPUT | Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Quad SPI |
| 3 | DUAL_IO | Instruction: Single-bit SPI / Address-Option: Dual SPI / Data: Dual SPI |
| 4 | QUAD_IO | Instruction: Single-bit SPI / Address-Option: Quad SPI / Data: Quad SPI |
| 5 | DUAL_CMD | Instruction: Dual SPI / Address-Option: Dual SPI / Data: Dual SPI |
| 6 | QUAD_CMD | Instruction: Quad SPI / Address-Option: Quad SPI / Data: Quad SPI |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 42.7.13 QSPI Scrambling Mode Register

**Name:** QSPI_SMR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RVDIS | SCREN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – RVDIS** Scrambling/Unscrambling Random Value Disable

| Value | Description |
|---|---|
| 0 | The scrambling/unscrambling algorithm includes the user scrambling key plus a random value that may differ between devices. |
| 1 | The scrambling/unscrambling algorithm includes only the user scrambling key. |

**Bit 0 – SCREN** Scrambling/Unscrambling Enable
0 (DISABLED): The scrambling/unscrambling is disabled.
1 (ENABLED): The scrambling/unscrambling is enabled.

### 42.7.14 QSPI Scrambling Key Register

**Name:** QSPI_SKR
**Offset:** 0x44
**Reset:** –
**Property:** Write-only

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | USRK[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | USRK[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | USRK[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | USRK[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – USRK[31:0]** User Scrambling Key

### 42.7.15 QSPI Write Protection Mode Register

**Name:** QSPI_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x515350 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See section Register Write Protection for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x515350 (QSP in ASCII) |
| 1 | Enables the write protection if WPKEY corresponds to 0x515350 (QSP in ASCII) |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 42.7.16 QSPI Write Protection Status Register

**Name:** QSPI_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 15:8 – WPVSRC[7:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the QSPI_WPSR. |
| 1 | A write protection violation has occurred since the last read of the QSPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 43. Two-wire Interface (TWIHS)

## 43.1 Description

The Two-wire Interface (TWIHS) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 400 kbit/s in Fast mode and up to 3.4 Mbit/s in High-speed slave mode only, based on a byte-oriented transfer format. It can be used with any Two-wire Interface bus Serial EEPROM and I²C-compatible devices, such as a Real-Time Clock (RTC), Dot Matrix/Graphic LCD Controller and temperature sensor. The TWIHS is programmable as a master or a slave with sequential or single-byte access. Multiple master capability is supported.

A configurable baud rate generator permits the output data rate to be adapted to a wide range of core clock frequencies.

The table below lists the compatibility level of the Two-wire Interface in Master mode and a full $I^2C$ compatible device.

**Table 43-1. TWI Compatibility with $I^2C$ Standard**

| $I^2C$ Standard | TWI |
|---|---|
| Standard Mode Speed (100 kHz) | Supported |
| Fast Mode Speed (400 kHz) | Supported |
| High-speed Mode (Slave only, 3.4 MHz) | Supported |
| 7- or 10-bit(1) Slave Addressing | Supported |
| START Byte(2) | Not Supported |
| Repeated Start (Sr) Condition | Supported |
| ACK and NACK Management | Supported |
| Input Filtering | Supported |
| Slope Control | Not Supported |
| Clock Stretching | Supported |
| Multi Master Capability | Supported |

**Note:**

1. 10-bit support in Master mode only.
2. START + b000000001 + Ack + Sr

## 43.2 Embedded Characteristics

- 3 TWIHSs
- Compatible with Two-wire Interface Serial Memory and I²C Compatible Devices(1)
- One, Two or Three Bytes for Slave Address
- Sequential Read/Write Operations
- Master and Multimaster Operation (Standard and Fast Modes Only)
- Slave Mode Operation (Standard, Fast and High-Speed Modes)
- Bit Rate: Up to 400 Kbit/s in Fast Mode and 3.4 Mbit/s in High-Speed Mode (Slave Mode Only)
- General Call Supported in Slave Mode
- SleepWalking (Asynchronous and Partial Wakeup)
- SMBus Support

© 2021 Microchip Technology Inc. and its subsidiaries

- Connection to DMA Controller (DMA) Channel Capabilities Optimizes Data Transfers
- Register Write Protection

**Note:**
See TWI Compatibility with I2C Standard for details on compatibility with I²C Standard.

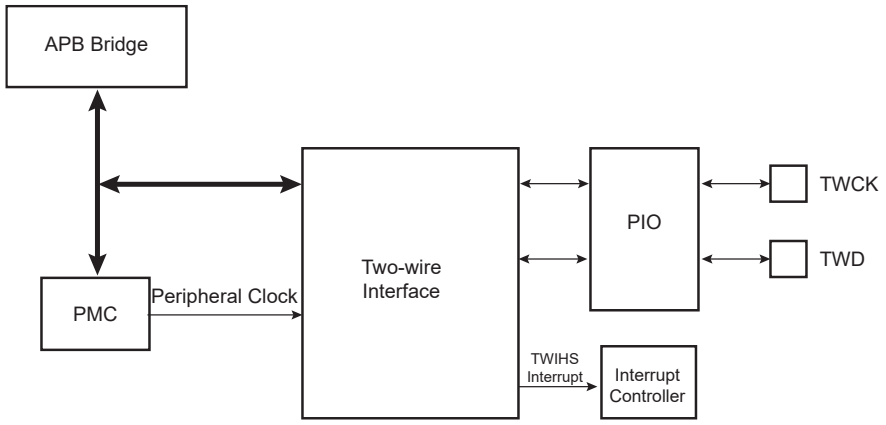
## 43.3 List of Abbreviations

**Table 43-2. Abbreviations**

| Abbreviation | Description |
|---|---|
| TWI | Two-wire Interface |
| A | Acknowledge |
| NA | Non Acknowledge |
| P | Stop |
| S | Start |
| Sr | Repeated Start |
| SADR | Slave Address |
| ADR | Any address except SADR |
| R | Read |
| W | Write |

## 43.4 Block Diagram

**Figure 43-1. Block Diagram**




### 43.4.1 I/O Lines Description

**Table 43-3. I/O Lines Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| TWD | Two-wire Serial Data | Input/Output |
| TWCK | Two-wire Serial Clock | Input/Output |

## 43.5 Product Dependencies

### 43.5.1 I/O Lines

Both TWD and TWCK are bidirectional lines, connected to a positive supply voltage via a current source or pullup resistor. When the bus is free, both lines are high. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function.

TWD and TWCK pins may be multiplexed with PIO lines. To enable the TWIHS, the user must program the PIO Controller to dedicate TWD and TWCK as peripheral lines. When High-speed Slave mode is enabled, the analog pad filter must be enabled.

The user must not program TWD and TWCK as open-drain. This is already done by the hardware.

### 43.5.2 Power Management

Enable the peripheral clock.

The TWIHS may be clocked through the Power Management Controller (PMC), thus the user must first configure the PMC to enable the TWIHS clock.

### 43.5.3 Interrupt Sources

The TWIHS has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TWIHS.

## 43.6 Functional Description

### 43.6.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data is transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited, shown in Transfer Format.

Each transfer begins with a START condition and terminates with a STOP condition, as shown in Figure 43-2.

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines the STOP condition.

**Figure 43-2. START and STOP Conditions**

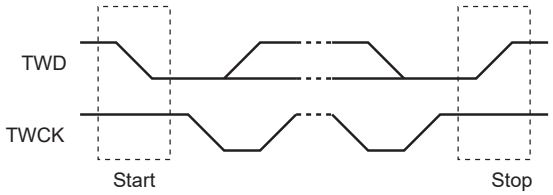
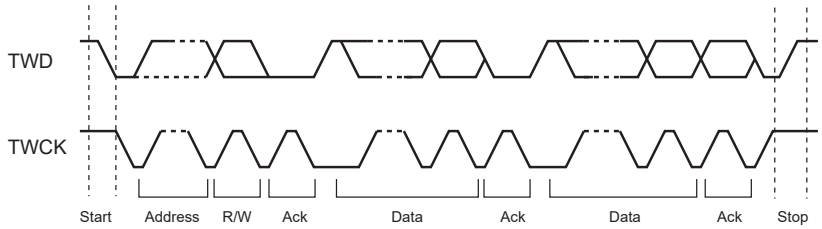



**Figure 43-3. Transfer Format**




### 43.6.2 Modes of Operation

The TWIHS has different modes of operation:

- Master Transmitter mode (Standard and Fast modes only)

- Master Receiver mode (Standard and Fast modes only)
- Multimaster Transmitter mode (Standard and Fast modes only)
- Multimaster Receiver mode (Standard and Fast modes only)
- Slave Transmitter mode (Standard, Fast and High-speed modes)
- Slave Receiver mode (Standard, Fast and High-speed modes)

These modes are described in the following sections.

### 43.6.3 Master Mode

#### 43.6.3.1 Definition

The master is the device that starts a transfer, generates a clock and stops it. This operating mode is not available if High-speed mode is selected.

#### 43.6.3.2 Programming Master Mode

The following registers must be programmed before entering Master mode:

1. TWIHS_MMR.DADR (+ IADRSZ + IADR if a 10-bit device is addressed): The device address is used to access slave devices in Read or Write mode.
2. TWIHS_CWGR.CKDIV + CHDIV + CLDIV: Clock Waveform register
3. TWIHS_CR.SVDIS: Disables the Slave mode
4. TWIHS_CR.MSEN: Enables the Master mode

**Note:** If the TWIHS is already in Master mode, the device address (DADR) can be configured without disabling the Master mode.

#### 43.6.3.3 Master Transmitter Mode

This operating mode is not available if High-speed mode is selected.

After the master initiates a START condition when writing into the Transmit Holding register (TWIHS_THR), it sends a 7-bit slave address, configured in the Master Mode register (DADR in TWIHS_MMR), to notify the slave device. The bit following the slave address indicates the transfer direction, 0 in this case (MREAD = 0 in TWIHS_MMR).

The TWIHS transfers require the slave to acknowledge each received byte. During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. If the slave does not acknowledge the byte, then the Not Acknowledge flag (NACK) is set in the TWIHS Status Register (TWIHS_SR) of the master and a STOP condition is sent. The NACK flag must be cleared by reading TWIHS_SR before the next write into TWIHS_THR. As with the other status bits, an interrupt can be generated if enabled in the Interrupt Enable register (TWIHS_IER). If the slave acknowledges the byte, the data written in the TWIHS_THR is then shifted in the internal shifter and transferred. When an acknowledge is detected, the TXRDY bit is set until a new write in the TWIHS_THR.

TXRDY is used as Transmit Ready for the DMA transmit channel.

While no new data is written in the TWIHS_THR, the serial clock line is tied low. When new data is written in the TWIHS_THR, the SCL is released and the data is sent. Setting the STOP bit in TWIHS_CR generates a STOP condition.

After a master write transfer, the serial clock line is stretched (tied low) as long as no new data is written in the TWIHS_THR or until a STOP command is performed.

To clear the TXRDY flag, first set the bit TWIHS_CR.MSDIS, then set the bit TWIHS_CR.MSEN.

See the figures below.

**Figure 43-4. Master Write with One Data Byte**

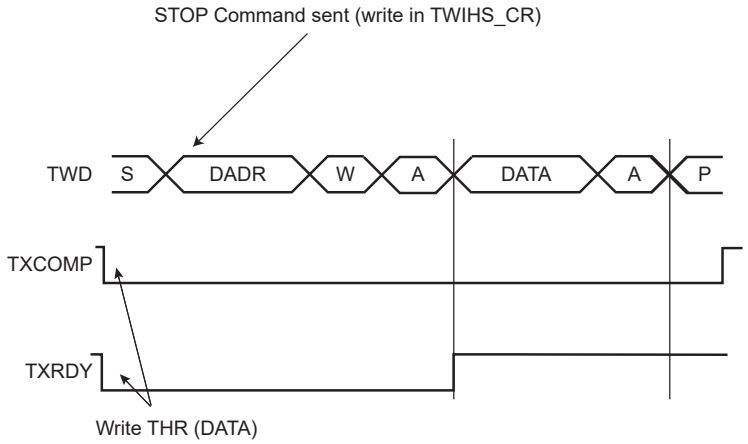



**Figure 43-5. Master Write with Multiple Data Bytes**

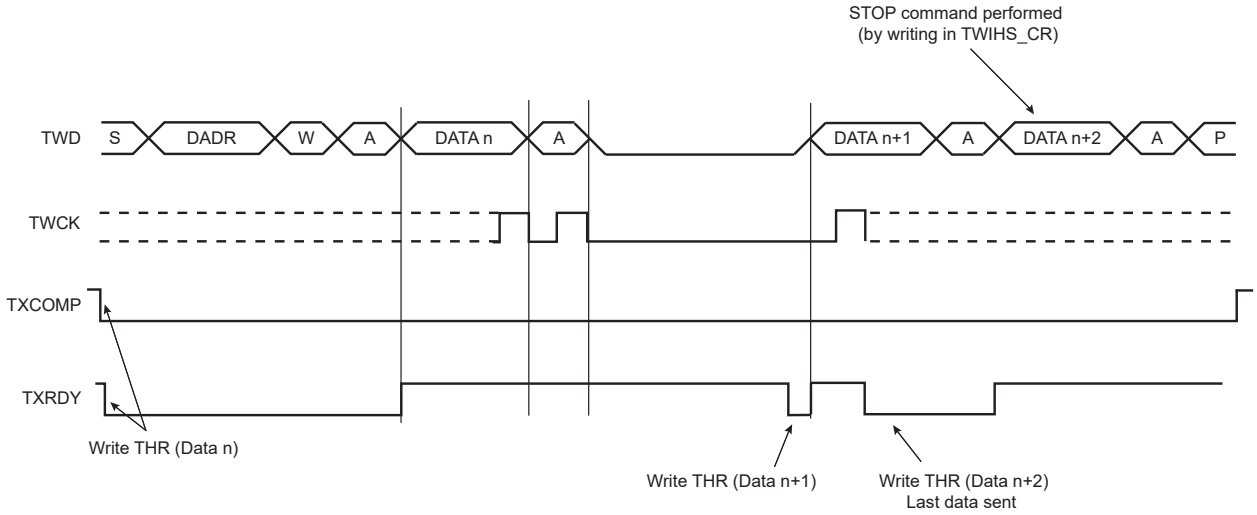
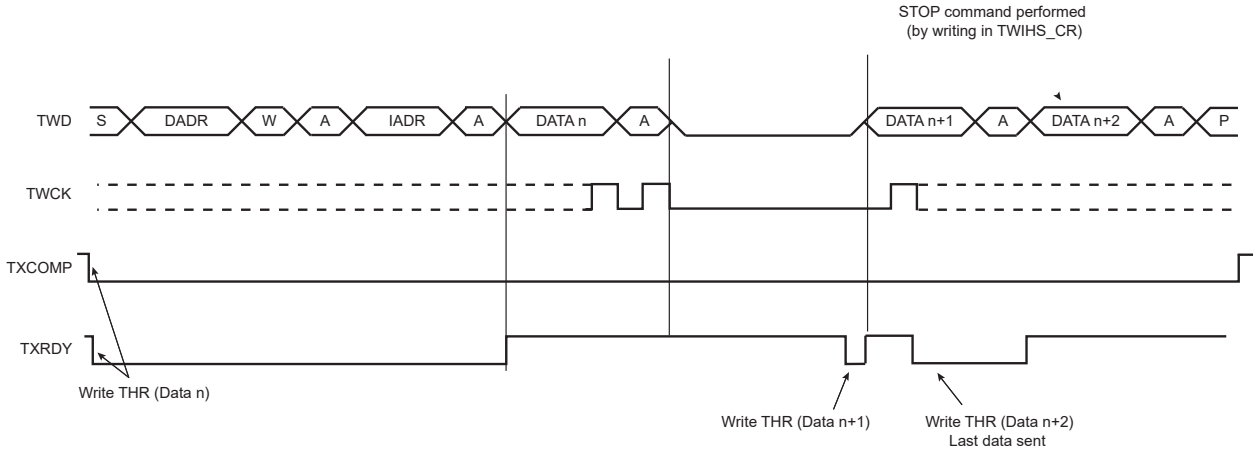



**Figure 43-6. Master Write with One-Byte Internal Address and Multiple Data Bytes**




### 43.6.3.4 Master Receiver Mode

Master Receiver mode is not available if High-speed mode is selected.

The read sequence begins by setting the START bit. After the START condition has been sent, the master sends a 7-bit slave address to notify the slave device. The bit following the slave address indicates the transfer direction, 1 in this case (MREAD = 1 in TWIHS_MMR). During the acknowledge clock pulse (9th pulse), the master releases the

data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse and sets TWIHS_SR.NACK if the slave does not acknowledge the byte.

If an acknowledge is received, the master is then ready to receive data from the slave. After data has been received, the master sends an acknowledge condition to notify the slave that the data has been received except for the last data (see Master Read with One Data Byte). When TWIHS_SR.RXRDY is set, a character has been received in the Receive Holding register (TWIHS_RHR). The RXRDY bit is reset when reading the TWIHS_RHR.

When a single data byte read is performed, with or without internal address (IADR), the START and STOP bits must be set at the same time. See Master Read with One Data Byte. When a multiple data byte read is performed, with or without internal address (IADR), the STOP bit must be set after the next-to-last data received (same condition applies for START bit to generate a REPEATED START). See Master Read with Multiple Data Bytes. For internal address usage, see Internal Address.

If TWIHS_RHR is full (RXRDY high) and the master is receiving data, the serial clock line is tied low before receiving the last bit of the data and until the TWIHS_RHR is read. Once the TWIHS_RHR is read, the master stops stretching the serial clock line and ends the data reception. See Master Read Clock Stretching with Multiple Data Bytes.

**⚠WARNING** When receiving multiple bytes in Master Read mode, if the next-to-last access is not read (the RXRDY flag remains high), the last access is not completed until TWIHS_RHR is read. The last access stops on the next-to-last bit (clock stretching). When the TWIHS_RHR is read, there is only half a bit period to send the STOP (or START) command, else another read access might occur (spurious access).

A possible workaround is to set the STOP (or START) bit before reading the TWIHS_RHR on the next-to-last access (within IT handler).

**Figure 43-7. Master Read with One Data Byte**

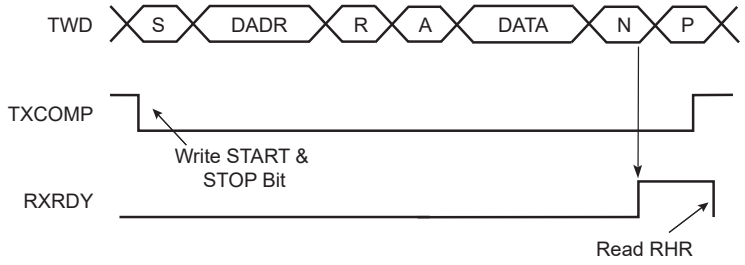



**Figure 43-8. Master Read with Multiple Data Bytes**

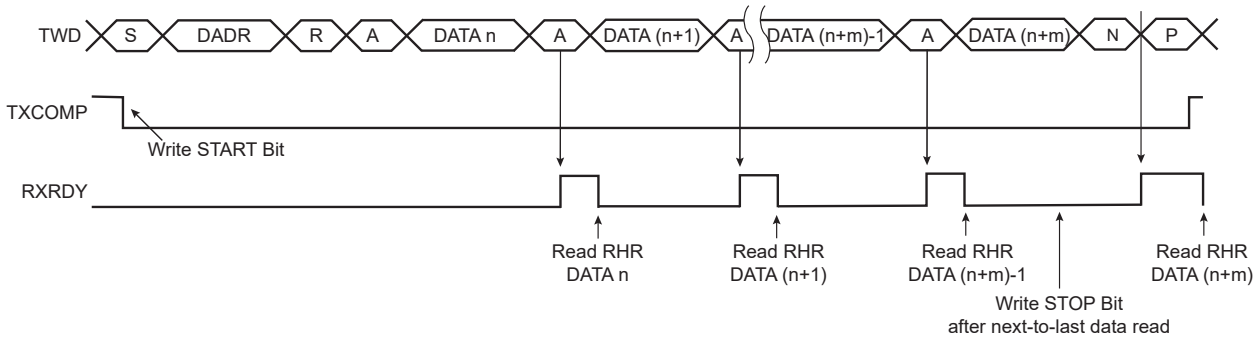
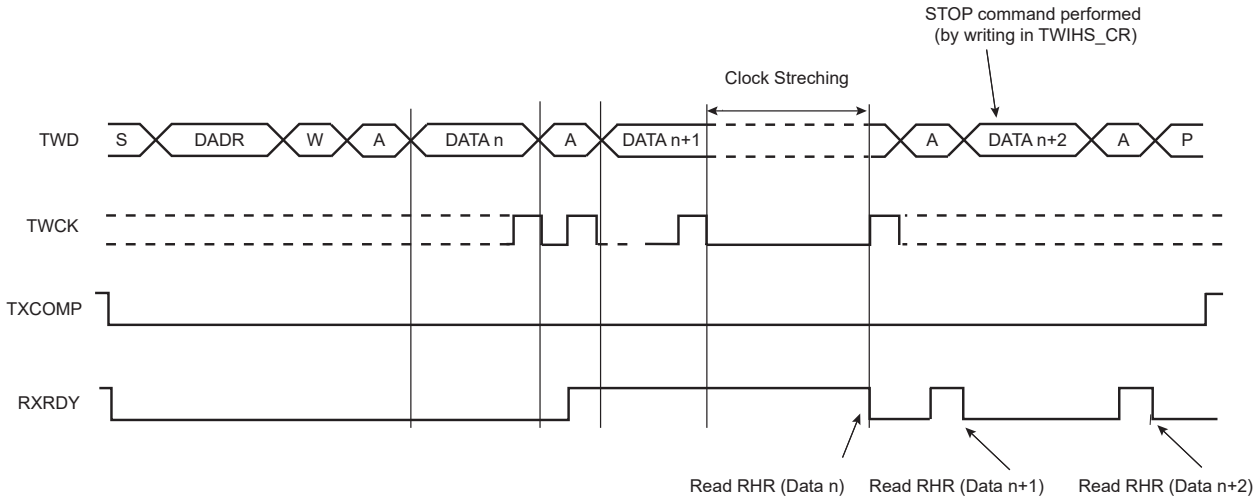


**Figure 43-9. Master Read Clock Stretching with Multiple Data Bytes**




RXRDY is used as receive ready for the DMA receive channel.

### 43.6.3.5 Internal Address

The TWIHS can perform transfers with 7-bit slave address devices and with 10-bit slave address devices.

#### 43.6.3.5.1 7-bit Slave Addressing

When addressing 7-bit slave devices, the internal address bytes are used to perform random address (read or write) accesses to reach one or more data bytes, e.g. within a memory page location in a serial memory. When performing read operations with an internal address, the TWIHS performs a write operation to set the internal address into the slave device, and then switch to Master Receiver mode. Note that the second START condition (after sending the IADR) is sometimes called “repeated start” (Sr) in $I^2C$ fully-compatible devices. See Master Read with One-, Two- or Three-Byte Internal Address and One Data Byte.

See Master Write with One-, Two- or Three-Byte Internal Address and One Data Byte and Internal Address Usage for the master write operation with internal address.

The three internal address bytes are configurable through TWIHS_MMR.

If the slave device supports only a 7-bit address, i.e., no internal address, IADRSZ must be set to 0.

The table below shows the abbreviations used in the figures below.

**Table 43-4. Abbreviations**

| Abbreviation | Definition |
|---|---|
| S | Start |
| Sr | Repeated Start |
| P | Stop |
| W | Write |
| R | Read |
| A | Acknowledge |
| NA | Not Acknowledge |
| DADR | Device Address |
| IADR | Internal Address |

**Figure 43-10. Master Write with One-, Two- or Three-Byte Internal Address and One Data Byte**

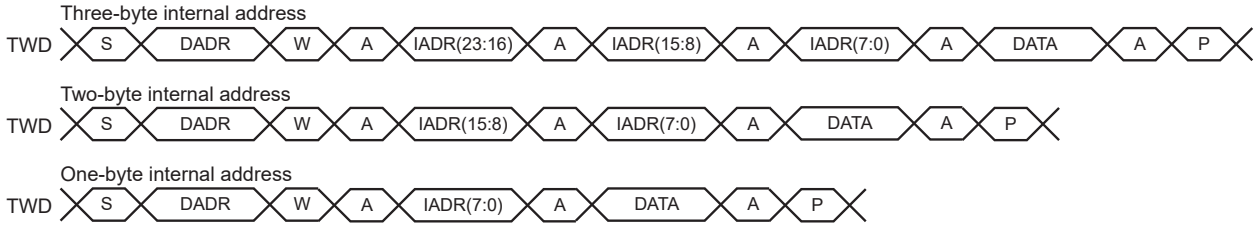
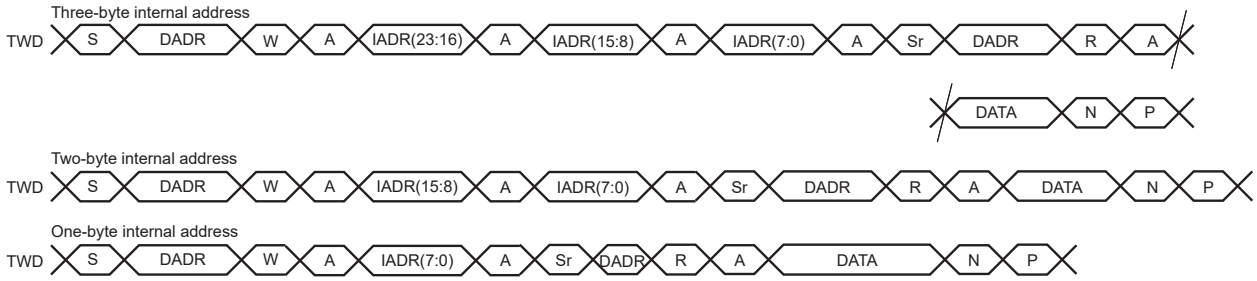



**Figure 43-11. Master Read with One-, Two- or Three-Byte Internal Address and One Data Byte**




#### 43.6.3.5.2 10-bit Slave Addressing

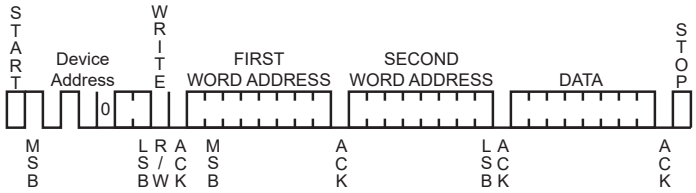
For a slave address higher than seven bits, configure the address size (IADRSZ) and set the other slave address bits in the Internal Address register (TWIHS_IADR). The two remaining internal address bytes, IADR[15:8] and IADR[23:16], can be used the same way as in 7-bit slave addressing.

Example: Address a 10-bit device (10-bit device address is b1 b2 b3 b4 b5 b6 b7 b8 b9 b10)

1. Program IADRSZ = 1,
2. Program DADR with 1 1 1 1 0 b1 b2 (b1 is the MSB of the 10-bit address, b2, etc.)
3. Program TWIHS_IADR with b3 b4 b5 b6 b7 b8 b9 b10 (b10 is the LSB of the 10-bit address)

The figure below shows a byte write to a memory device. This demonstrates the use of internal addresses to access the device.

**Figure 43-12. Internal Address Usage**




### 43.6.3.6 Repeated Start

In addition to Internal Address mode, REPEATED START (Sr) can be generated manually by writing the START bit at the end of a transfer instead of the STOP bit. In such case, the parameters of the next transfer (direction, SADR, etc.) need to be set before writing the START bit at the end of the previous transfer.

See Read/Write Flowcharts for detailed flowcharts.

Note that generating a REPEATED START after a single data read is not supported.

### 43.6.3.7 Bus Clear Command

The TWIHS can perform a Bus Clear command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Start the transfer by setting TWIHS_CR.CLEAR.

### 43.6.3.8 Using the DMA Controller (DMAC) in Master Mode

The use of the DMA significantly reduces the CPU load.

To ensure correct implementation, follow the programming sequences below:

#### 43.6.3.8.1 Data Transmit with the DMA in Master Mode

The DMA transfer size must be defined with the buffer size minus 1. The remaining character must be managed without DMA to ensure that the exact number of bytes are transmitted regardless of system bus latency conditions during the end of the buffer transfer period.

1. Initialize the DMA (channels, memory pointers, size - 1, etc.);
2. Configure the Master mode (DADR, CKDIV, MREAD = 0, etc.) or Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.
5. Disable the DMA.
6. Wait for the TXRDY flag in TWIHS_SR.
7. Set TWIHS_CR.STOP.
8. Write the last character in TWIHS_THR.
9. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

#### 43.6.3.8.2 Data Receive with the DMA in Master Mode

The DMA transfer size must be defined with the buffer size minus 2. The two remaining characters must be managed without DMA to ensure that the exact number of bytes are received regardless of system bus latency conditions encountered during the end of buffer transfer period.

1. Initialize the DMA (channels, memory pointers, size - 2, etc.);
2. Configure the Master mode (DADR, CKDIV, MREAD = 1, etc.) or Slave mode.
3. Enable the DMA.
4. (Master Only) Write TWIHS_CR.START to start the transfer.
5. Wait for the DMA status flag indicating that the buffer transfer is complete.
6. Disable the DMA.
7. Wait for the RXRDY flag in the TWIHS_SR.
8. Set TWIHS_CR.STOP.
9. Read the penultimate character in TWIHS_RHR.
10. Wait for the RXRDY flag in the TWIHS_SR.
11. Read the last character in TWIHS_RHR.
12. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

### 43.6.3.9 SMBus Mode

SMBus mode is enabled when a one is written to TWIHS_CR.SMBEN. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into TWIHS_SMBTR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A set of addresses has been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring TWIHS_CR.

#### 43.6.3.9.1 Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing a one to TWIHS_CR.PECEN enables automatic PEC handling in the current transfer. Transfers with and without PEC can be intermixed in the same system, since some slaves may not support PEC. The PEC LFSR is always updated on every bit transmitted or received, so that PEC handling on combined transfers is correct.

In Master Transmitter mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave compares it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave returns an ACK to the master. If the PEC values differ, data was corrupted, and the slave returns a NACK value. Some slaves may not be able to check the received PEC in time to return a NACK if an error occurred. In this case, the slave should always return an ACK after the PEC byte, and another method must be used to verify that the transmission was received correctly.

© 2021 Microchip Technology Inc. and its subsidiaries

In Master Receiver mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master compares it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and TWIHS_SR.PECERR is set. In Master Receiver mode, the PEC byte is always followed by a NACK transmitted by the master, since it is the last byte in the transfer.

In combined transfers, the PECRQ bit should only be set in the last of the combined transfers.

Consider the following transfer:

S, ADR+W, COMMAND_BYTE, ACK, SR, ADR+R, DATA_BYTE, ACK, PEC_BYTE, NACK, P

See Read/Write Flowcharts for detailed flowcharts.

#### 43.6.3.9.2 Timeouts

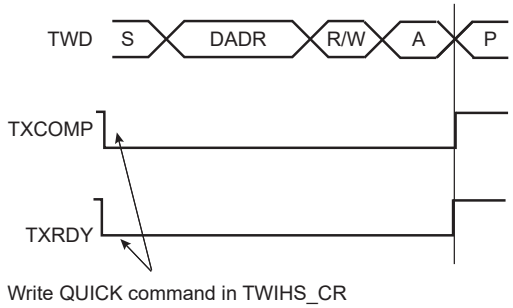
The TLOWS and TLOWM fields in TWIHS_SMBTR configure the SMBus timeout values. If a timeout occurs, the master transmits a STOP condition and leaves the bus. The TOUT bit is also set in TWIHS_SR.

### 43.6.3.10 SMBus Quick Command (Master Mode Only)

The TWIHS can perform a quick command by following these steps:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Write TWIHS_MMR.MREAD at the value of the one-bit command to be sent.
3. Start the transfer by setting TWIHS_CR.QUICK.

**Figure 43-13. SMBus Quick Command**




### 43.6.3.11 Read/Write Flowcharts

The flowcharts give examples for read and write operations. A polling or interrupt method can be used to check the status bits. The interrupt method requires that TWIHS_IER be configured first.

**Figure 43-14. TWIHS Write Operation with Single Data Byte without Internal Address**

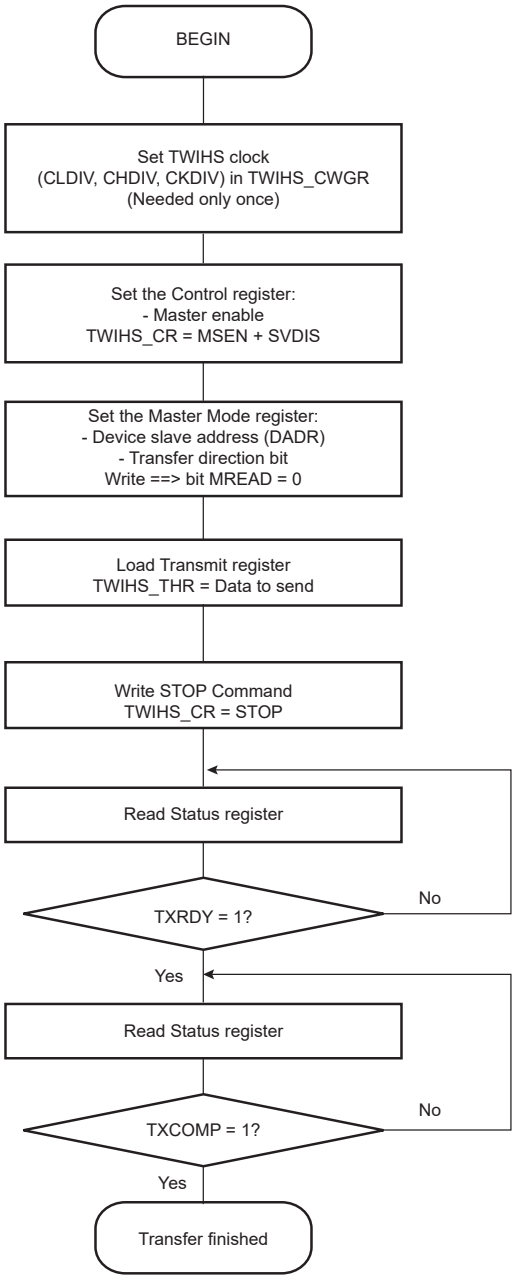


**Figure 43-15. TWIHS Write Operation with Single Data Byte and Internal Address**

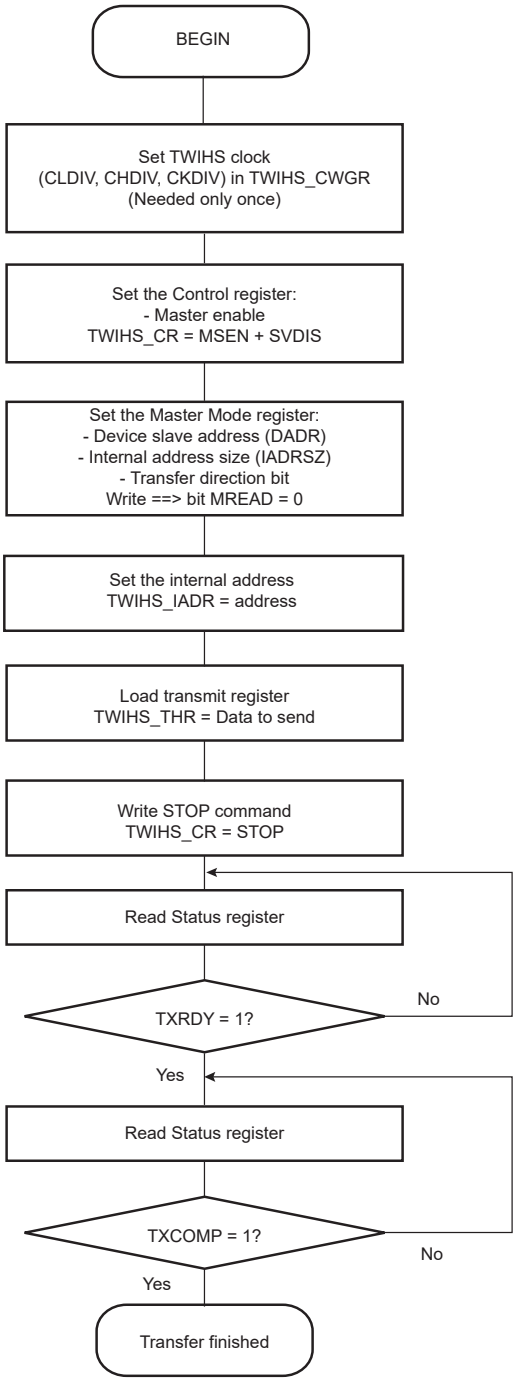


**Figure 43-16. TWIHS Write Operation with Multiple Data Bytes with or without Internal Address**

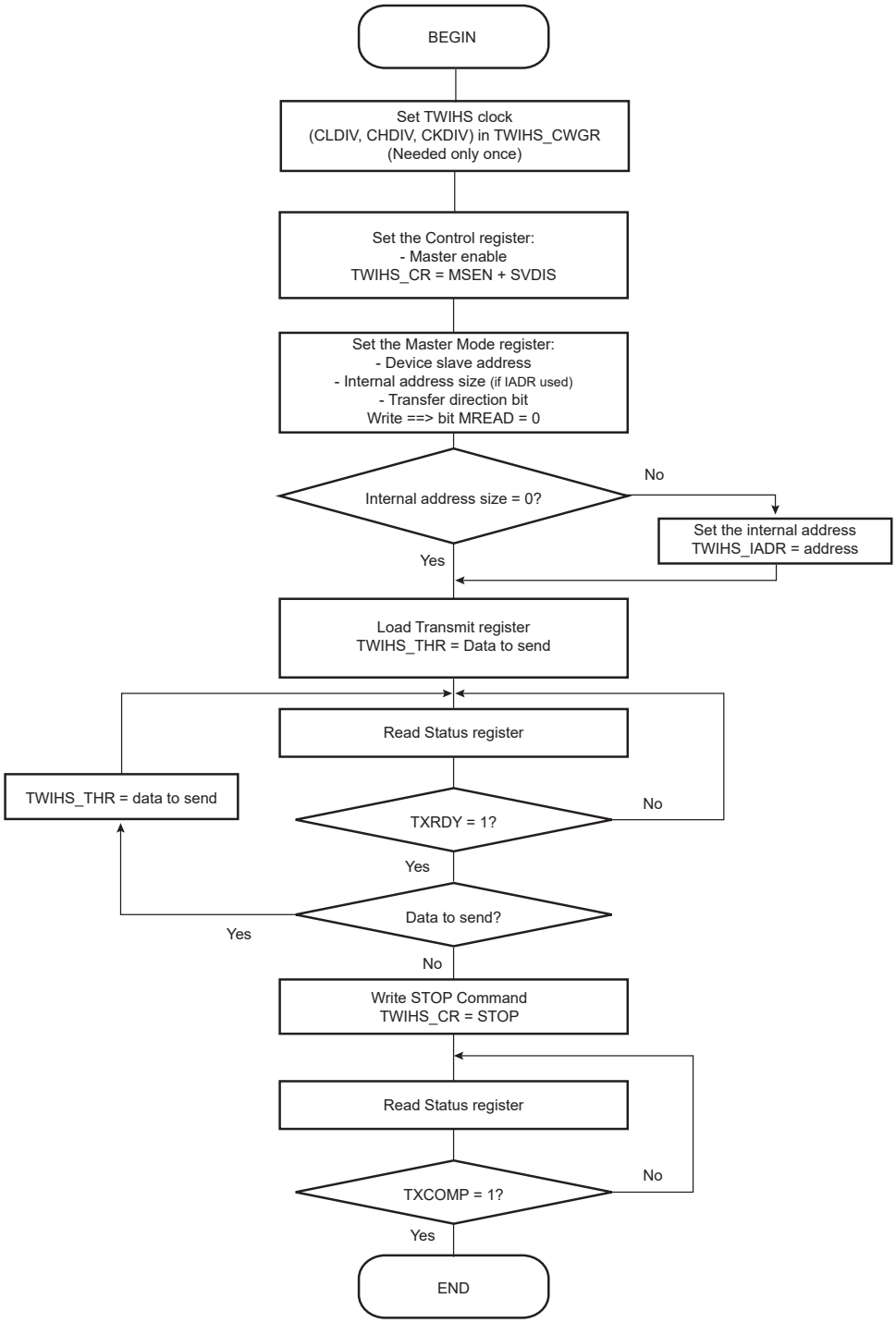


**Figure 43-17. SMBus Write Operation with Multiple Data Bytes with or without Internal Address and PEC Sending**

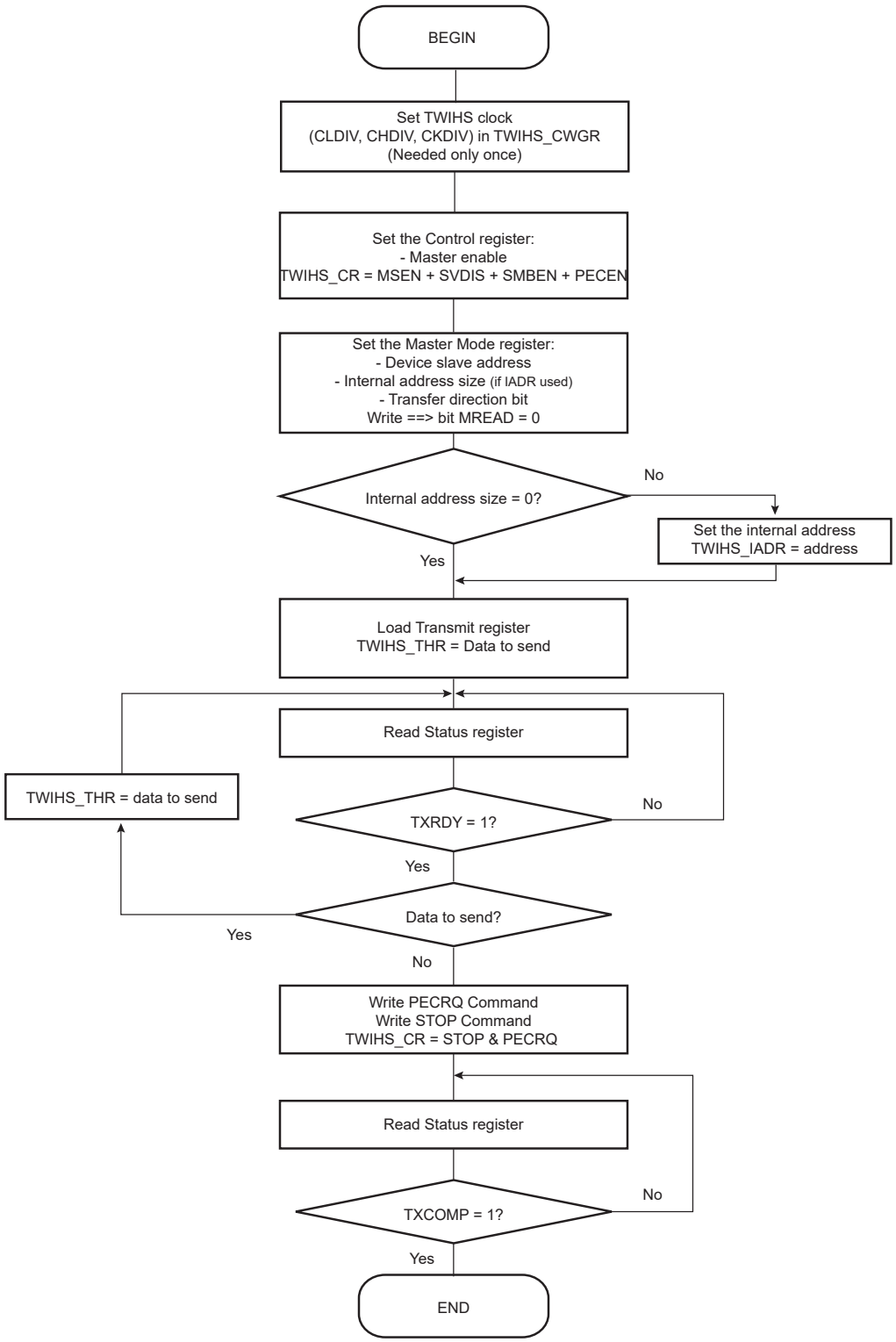


**Figure 43-18. SMBus Write Operation with Multiple Data Bytes with PEC and Alternative Command Mode**

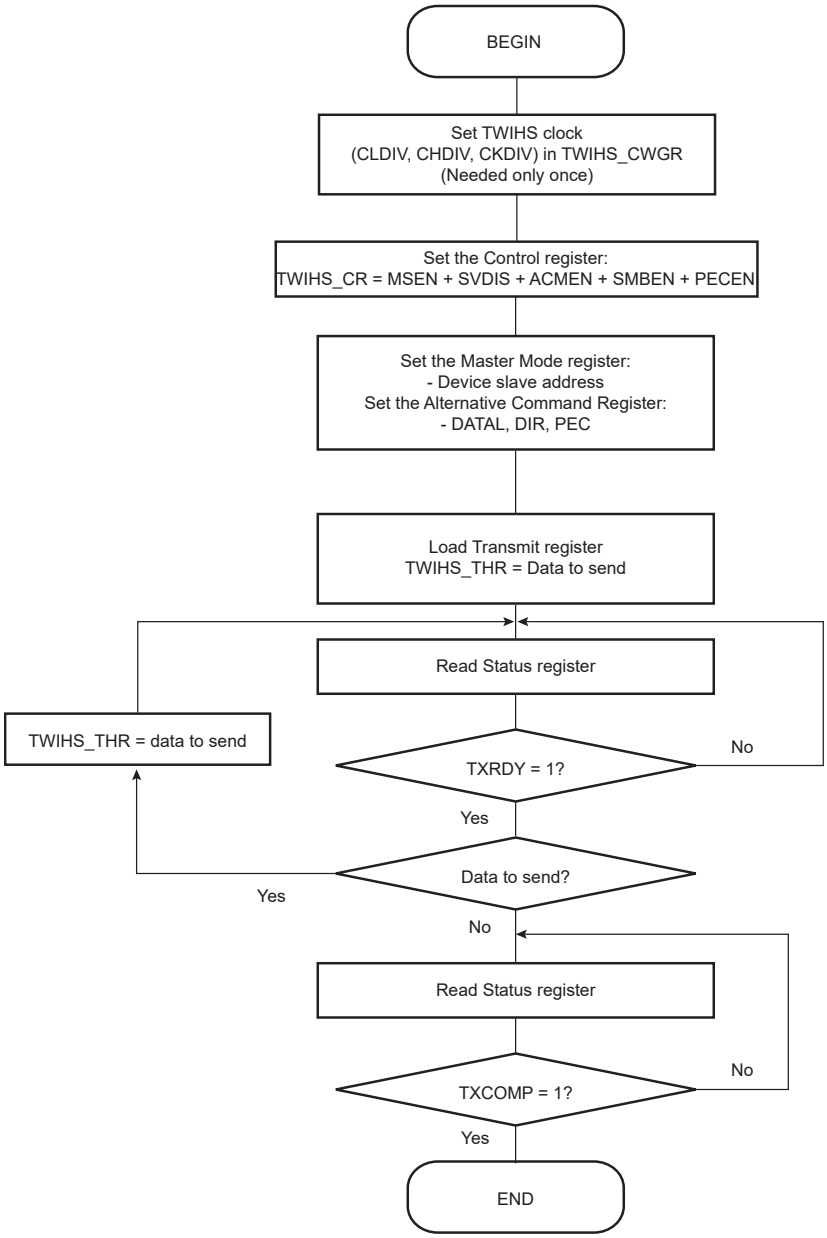


**Figure 43-19. TWIHS Write Operation with Multiple Data Bytes and Read Operation with Multiple Data Bytes (Sr)**

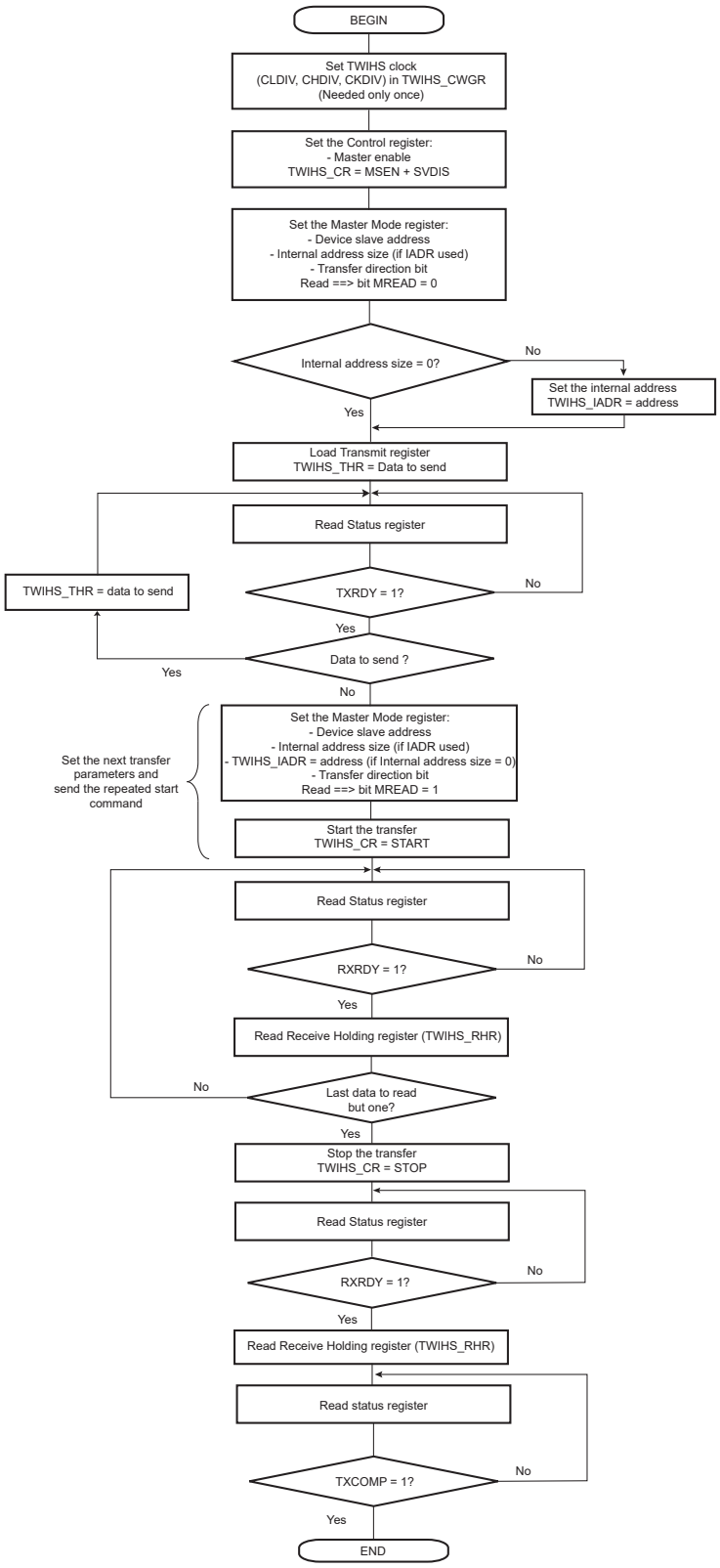


**Figure 43-20. TWIHS Write Operation with Multiple Data Bytes + Read Operation and Alternative Command Mode + PEC**

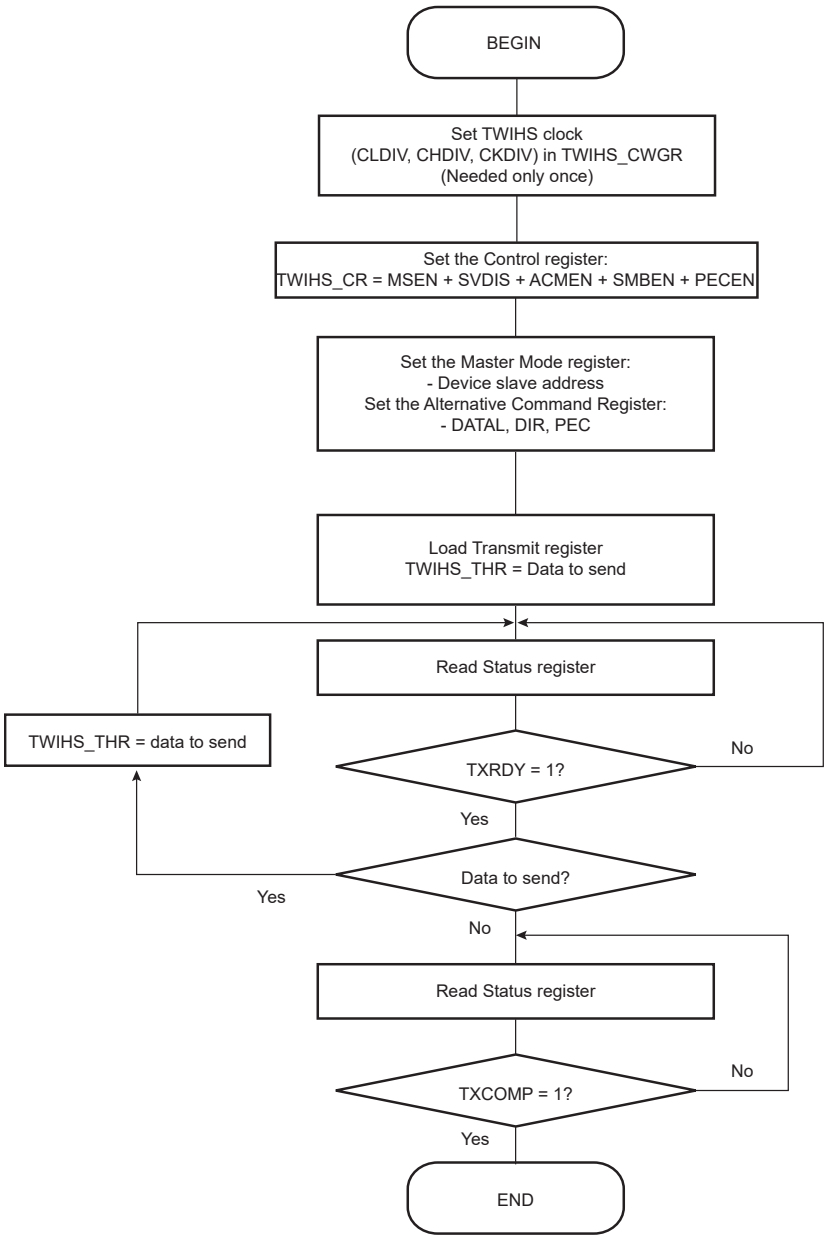


**Figure 43-21. TWIHS Read Operation with Single Data Byte without Internal Address**

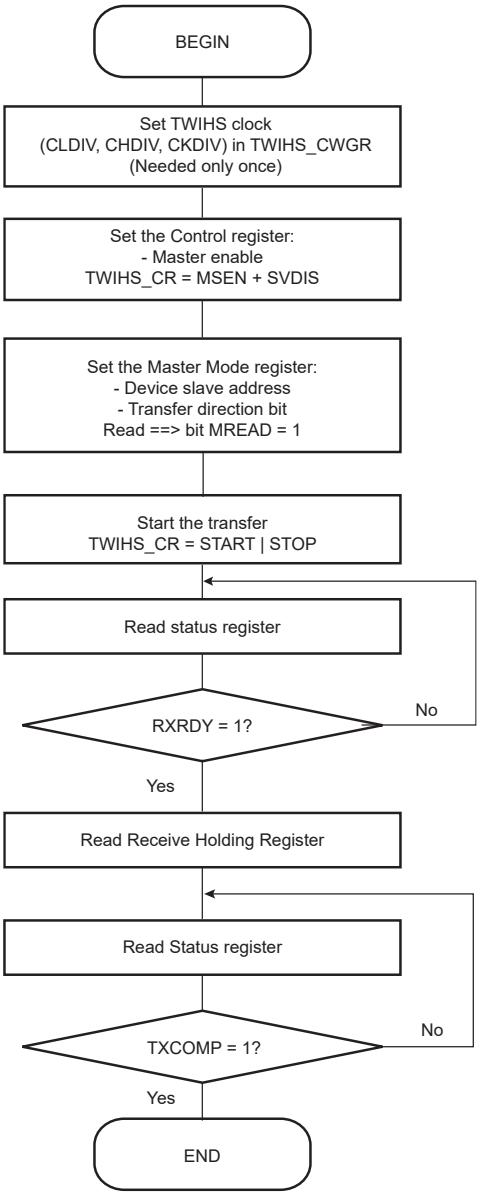


**Figure 43-22. TWIHS Read Operation with Single Data Byte and Internal Address**

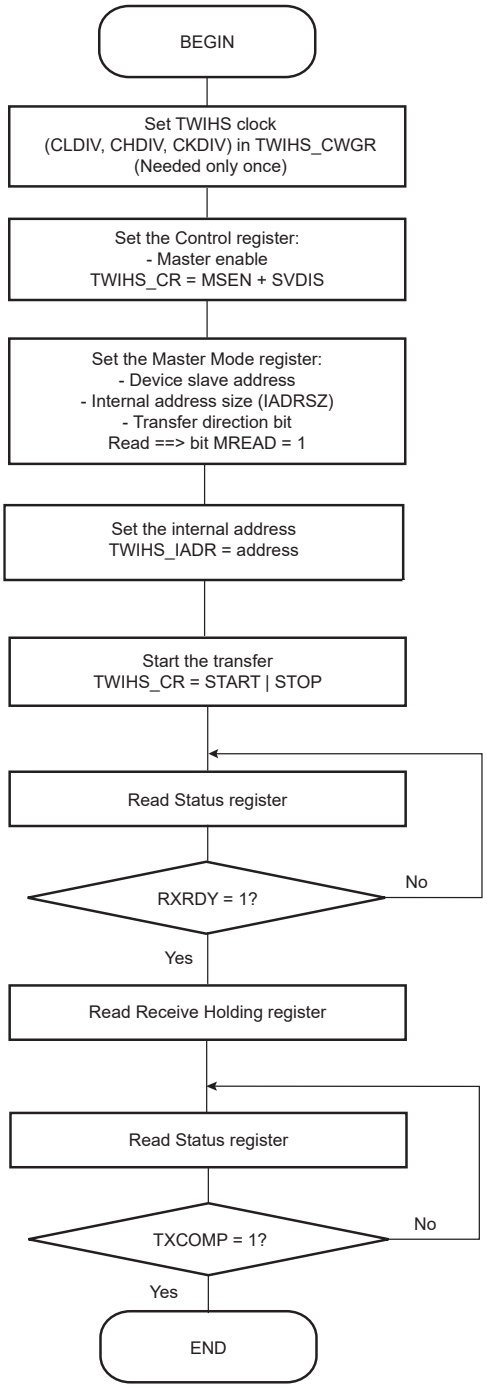


**Figure 43-23. TWIHS Read Operation with Multiple Data Bytes with or without Internal Address**

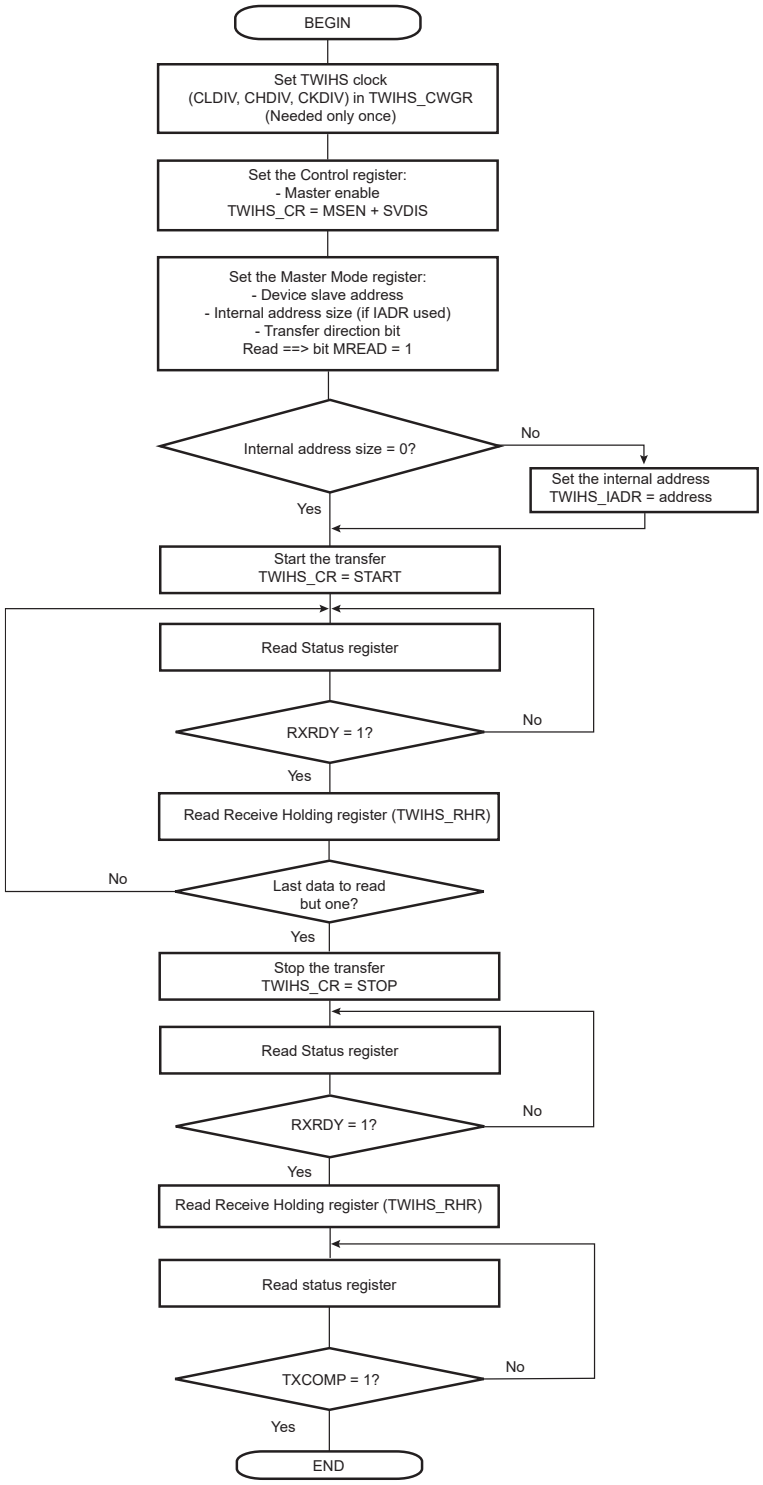


**Figure 43-24. TWIHS Read Operation with Multiple Data Bytes with or without Internal Address with PEC**

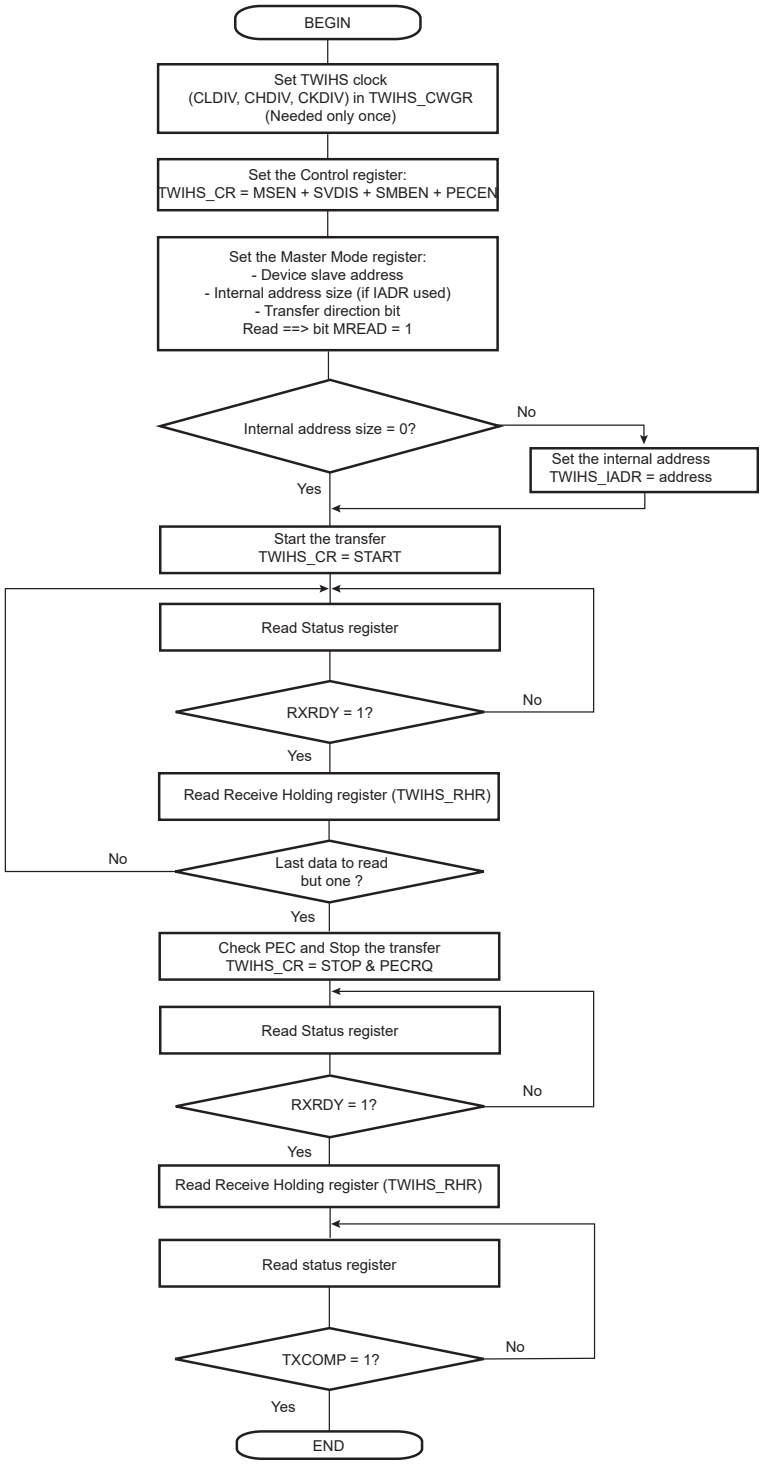


**Figure 43-25. TWIHS Read Operation with Multiple Data Bytes with Alternative Command Mode with PEC**

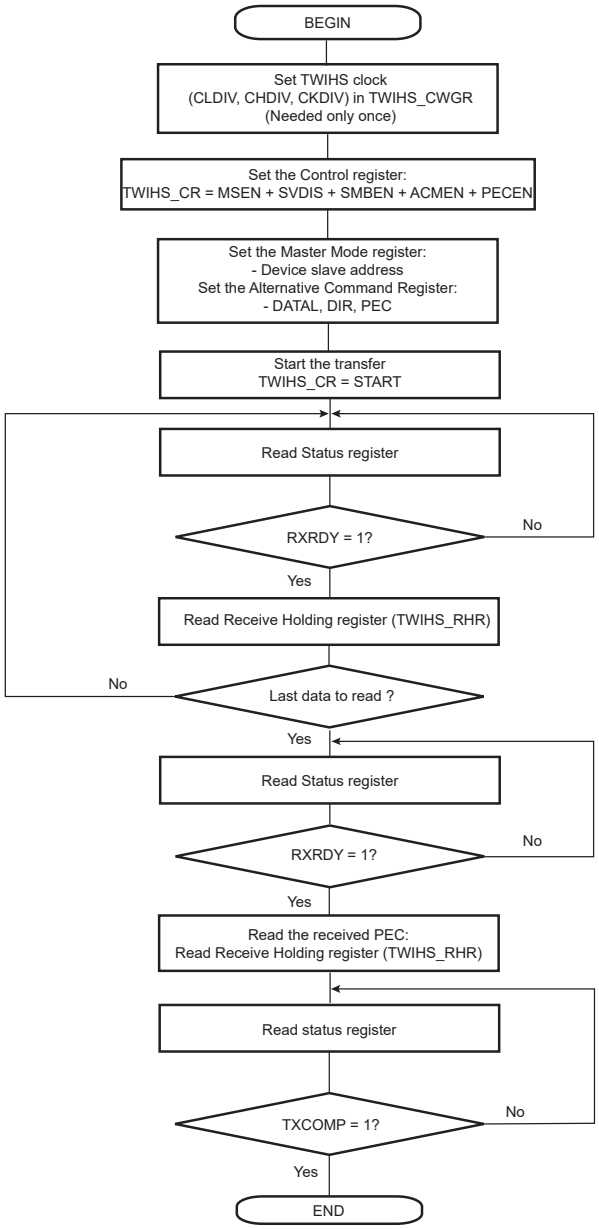


**Figure 43-26. TWIHS Read Operation with Multiple Data Bytes + Write Operation with Multiple Data Bytes (Sr)**

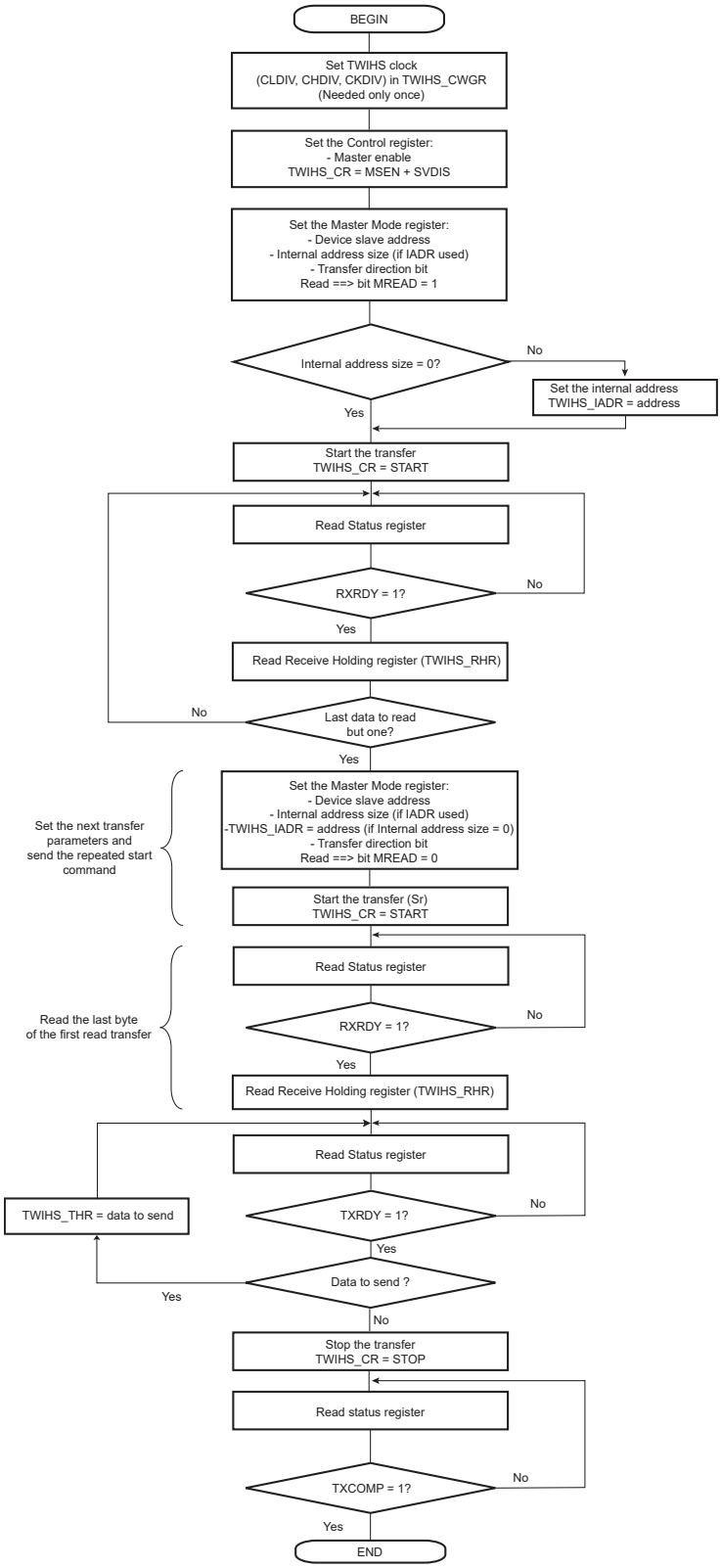
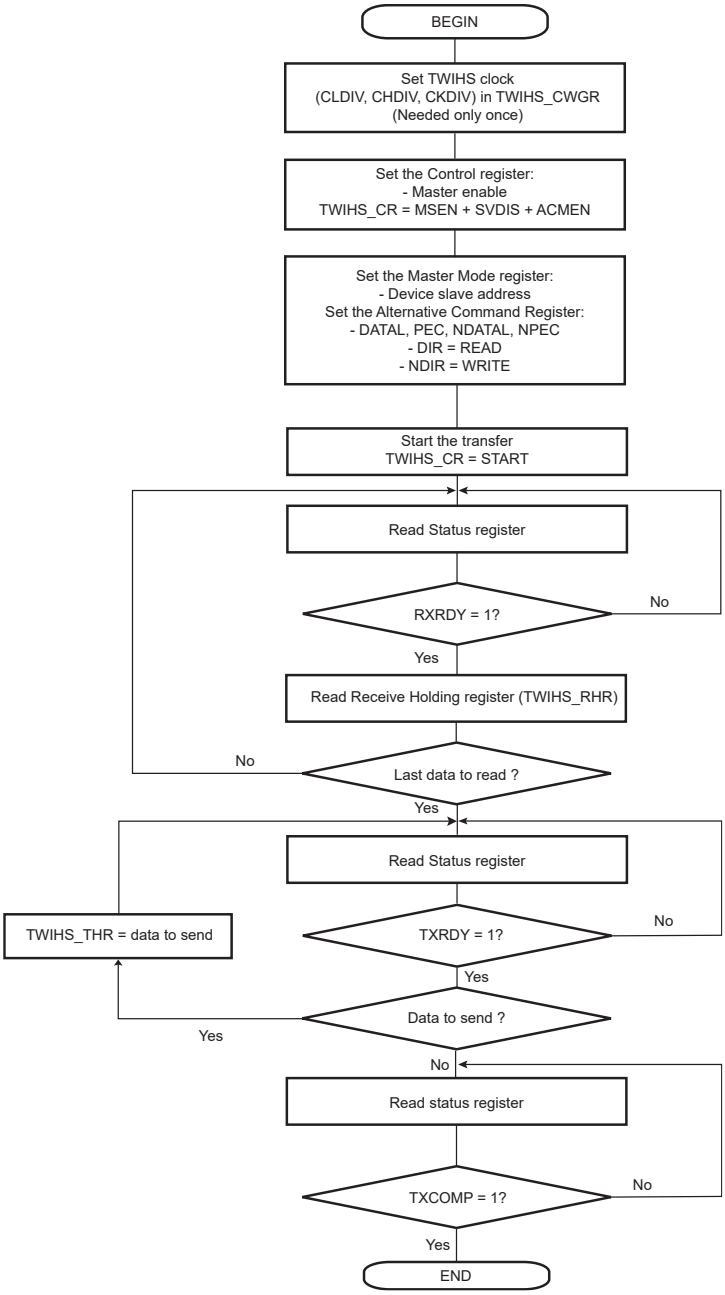


**Figure 43-27. TWIHS Read Operation with Multiple Data Bytes + Write with Alternative Command Mode with PEC**




## 43.6.4 Multimaster Mode

### 43.6.4.1 Definition

In Multimaster mode, more than one master may handle the bus at the same time without data corruption by using arbitration.

Arbitration starts as soon as two or more masters place information on the bus at the same time, and stops (arbitration is lost) for the master that intends to send a logical one while the other master sends a logical zero.

As soon as arbitration is lost by a master, it stops sending data and listens to the bus in order to detect a stop. When the stop is detected, the master that has lost arbitration may put its data on the bus by respecting arbitration.

Arbitration is illustrated in Arbitration Cases.

### 43.6.4.2 Different Multimaster Modes

Two Multimaster modes may be distinguished:

1. The TWIHS is considered as a master only and is never addressed.
2. The TWIHS may be either a master or a slave and may be addressed.

**Note:** Arbitration in supported in both Multimaster modes.

#### 43.6.4.2.1 TWIHS as Master Only

In this mode, the TWIHS is considered as a master only (MSEN is always at one) and must be driven like a master with the ARBLST (Arbitration Lost) flag in addition.

If arbitration is lost (ARBLST = 1), the user must reinitiate the data transfer.

If starting a transfer (ex.: DADR + START + W + Write in THR) and if the bus is busy, the TWIHS automatically waits for a STOP condition on the bus to initiate the transfer (see User Sends Data While the Bus is Busy).

**Note:** The state of the bus (busy or free) is not indicated in the user interface.

#### 43.6.4.2.2 TWIHS as Master or Slave

The automatic reversal from master to slave is not supported in case of a lost arbitration.

Then, in the case where TWIHS may be either a master or a slave, the user must manage the pseudo Multimaster mode described in the steps below:

1. Program the TWIHS in Slave mode (SADR + MSDIS + SVEN) and perform a slave access (if TWIHS is addressed).
2. If the TWIHS has to be set in Master mode, wait until TXCOMP flag is at 1.
3. Program the Master mode (DADR + SVDIS + MSEN) and start the transfer (ex: START + Write in THR).
4. As soon as the Master mode is enabled, the TWIHS scans the bus in order to detect if it is busy or free. When the bus is considered free, the TWIHS initiates the transfer.
5. As soon as the transfer is initiated and until a STOP condition is sent, the arbitration becomes relevant and the user must monitor the ARBLST flag.
6. If the arbitration is lost (ARBLST is set to 1), the user must program the TWIHS in Slave mode in case the master that won the arbitration needs to access the TWIHS.
7. If the TWIHS has to be set in Slave mode, wait until the TXCOMP flag is at 1 and then program the Slave mode.

**Note:** If the arbitration is lost and the TWIHS is addressed, the TWIHS does not acknowledge, even if it is programmed in Slave mode as soon as ARBLST is set to 1. Then the master must repeat SADR.

**Figure 43-28. User Sends Data While the Bus is Busy**

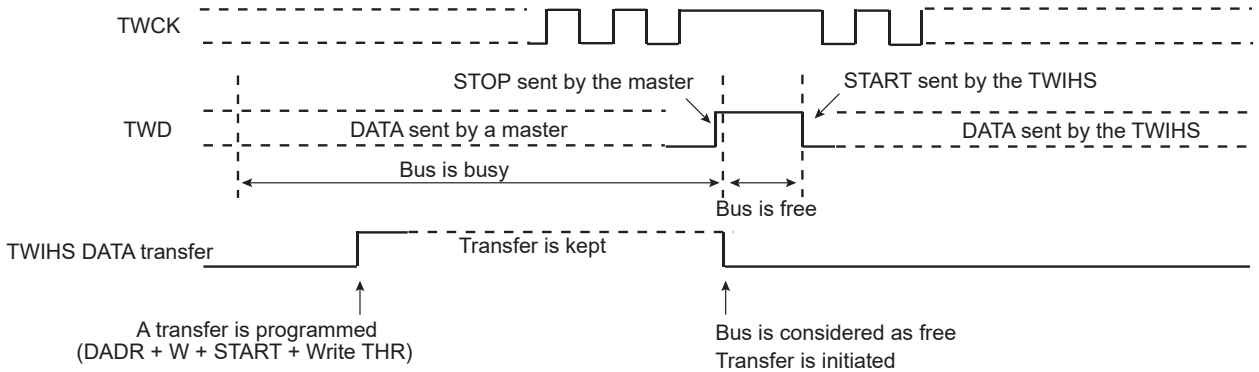
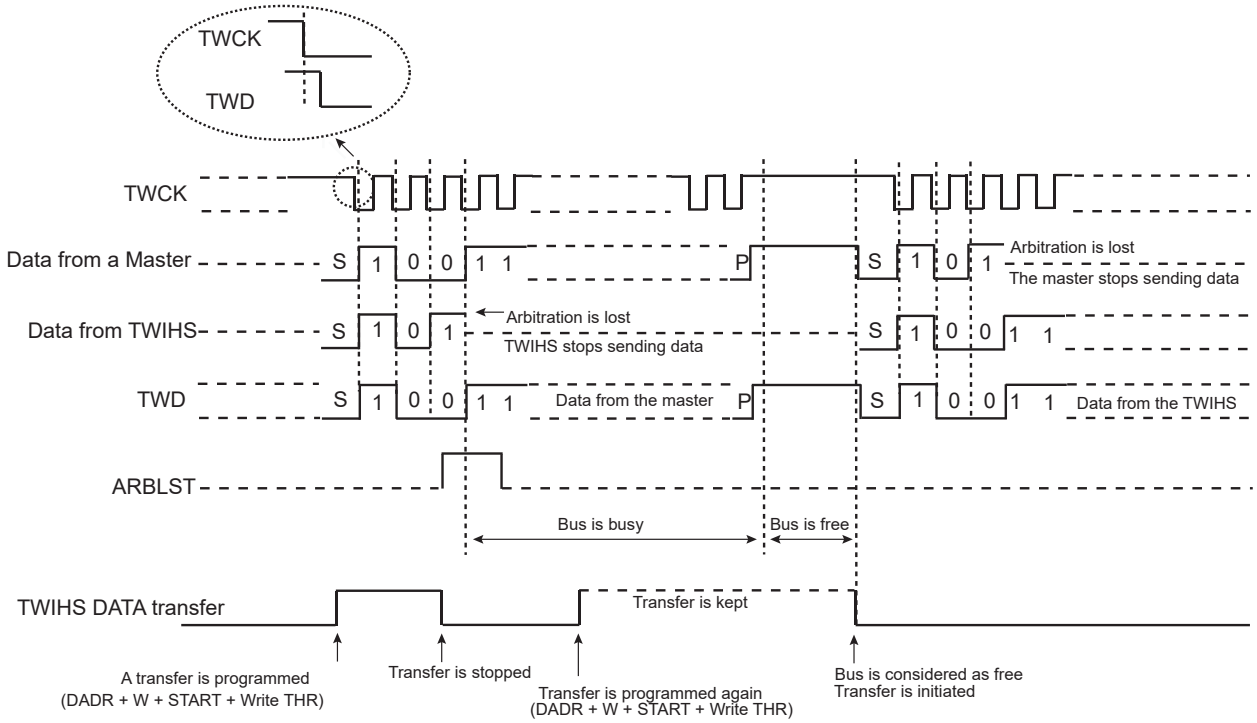
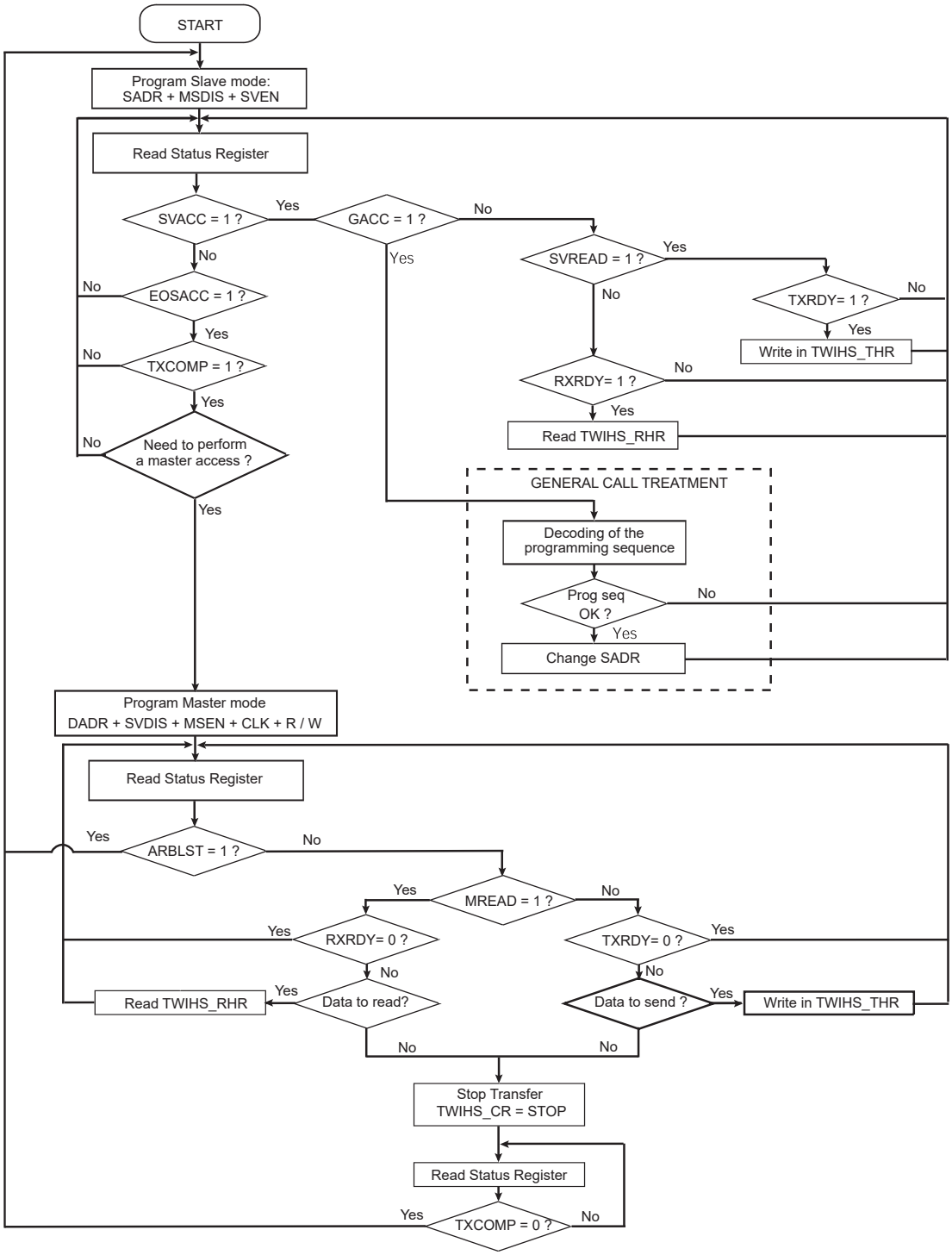


**Figure 43-29. Arbitration Cases**




The flowchart below gives an example of read and write operations in Multimaster mode.

**Figure 43-30. Multimaster Flowchart**




## 43.6.5 Slave Mode

### 43.6.5.1 Definition

Slave mode is defined as a mode where the device receives the clock and the address from another device called the master.

In this mode, the device never initiates and never completes the transmission (START, REPEATED_START and STOP conditions are always provided by the master).

#### 43.6.5.2 Programming Slave Mode

The following fields must be programmed before entering Slave mode:

1. TWIHS_SMR.SADR: The slave device address is used in order to be accessed by master devices in Read or Write mode.
2. (Optional) TWIHS_SMR.MASK can be set to mask some SADR address bits and thus allow multiple address matching.
3. TWIHS_CR.MSDIS: Disables the Master mode.
4. TWIHS_CR.SVEN: Enables the Slave mode.

As the device receives the clock, values written in TWIHS_CWGR are ignored.

#### 43.6.5.3 Receiving Data

After a START or REPEATED START condition is detected, and if the address sent by the master matches the slave address programmed in the SADR (Slave Address) field, the SVACC (Slave Access) flag is set and SVREAD (Slave Read) indicates the direction of the transfer.

SVACC remains high until a STOP condition or a REPEATED START is detected. When such a condition is detected, the EOSACC (End Of Slave Access) flag is set.

##### 43.6.5.3.1 Read Sequence

In the case of a read sequence (SVREAD is high), the TWIHS transfers data written in the TWIHS_THR until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the read sequence TXCOMP (Transmission Complete) flag is set and SVACC reset.

As soon as data is written in the TWIHS_THR, TXRDY (Transmit Holding Register Ready) flag is reset, and it is set when the internal shifter is empty and the sent data acknowledged or not. If the data is not acknowledged, the NACK flag is set.

Note that a STOP or a REPEATED START always follows a NACK.

To clear the TXRDY flag, first set TWIHS_CR.SVDIS, then set TWIHS_CR.SVEN.

See Read Access Ordered by a Master.

##### 43.6.5.3.2 Write Sequence

In the case of a write sequence (SVREAD is low), the RXRDY (Receive Holding Register Ready) flag is set as soon as a character has been received in TWIHS_RHR. RXRDY is reset when reading TWIHS_RHR.

The TWIHS continues receiving data until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the write sequence, the TXCOMP flag is set and SVACC is reset.

See Write Access Ordered by a Master.

##### 43.6.5.3.3 Clock Stretching Sequence

If TWIHS_THR or TWIHS_RHR is not written/read in time, the TWIHS performs a clock stretching.

Clock stretching information is given by the SCLWS (Clock Wait State) bit.

See Clock Stretching in Read Mode and Clock Stretching in Write Mode.

**Note:** Clock stretching can be disabled by configuring the SCLWSDIS bit in TWIHS_SMR. In that case, the UNRE and OVRE flags indicate an underrun (when TWIHS_THR is not filled on time) or an overrun (when TWIHS_RHR is not read on time).

##### 43.6.5.3.4 General Call

In the case where a GENERAL CALL is performed, the GACC (General Call Access) flag is set.

After GACC is set, the user must interpret the meaning of the GENERAL CALL and decode the new address programming sequence.

See Master Performs a General Call.

© 2021 Microchip Technology Inc. and its subsidiaries

### 43.6.5.4 Data Transfer

#### 43.6.5.4.1 Read Operation

The Read mode is defined as a data requirement from the master.

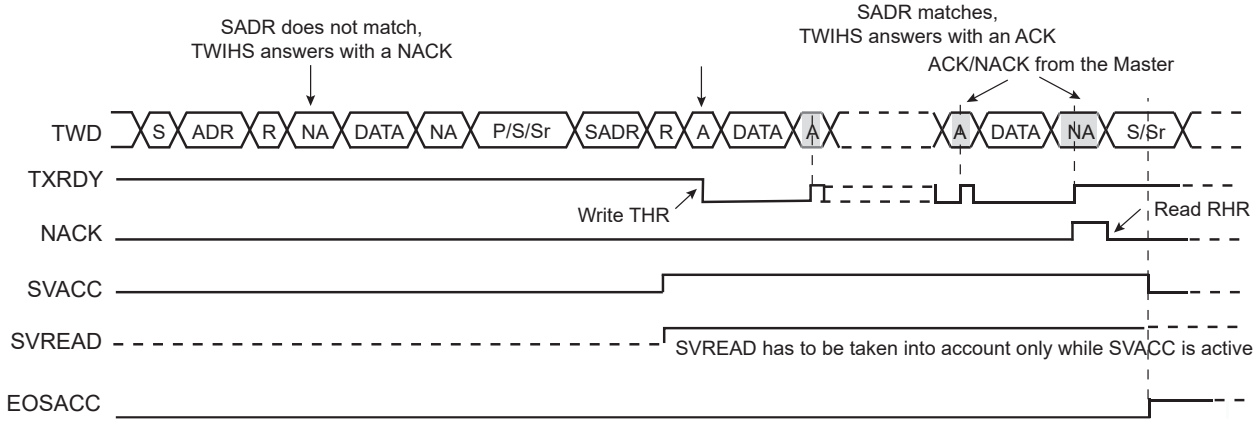
After a START or a REPEATED START condition is detected, the decoding of the address starts. If the slave address (SADR) is decoded, SVACC is set and SVREAD indicates the direction of the transfer.

Until a STOP or REPEATED START condition is detected, the TWIHS continues sending data loaded in TWIHS_THR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

The figure below describes the read operation.

**Figure 43-31. Read Access Ordered by a Master**




**Notes:**

1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. TXRDY is reset when data has been transmitted from TWIHS_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.

#### 43.6.5.4.2 Write Operation

The Write mode is defined as a data transmission from the master.

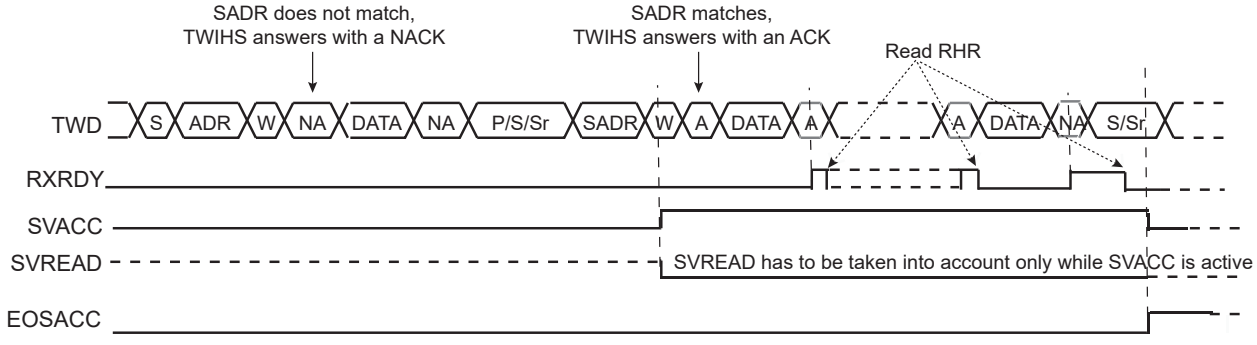
After a START or a REPEATED START, the decoding of the address starts. If the slave address is decoded, SVACC is set and SVREAD indicates the direction of the transfer (SVREAD is low in this case).

Until a STOP or REPEATED START condition is detected, the TWIHS stores the received data in TWIHS_RHR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

The figure below describes the write operation.

**Figure 43-32. Write Access Ordered by a Master**



**Notes:**

1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. RXRDY is set when data has been transmitted from the internal shifter to TWIHS_RHR and reset when this data is read.

#### 43.6.5.4.3 General Call

The general call is performed in order to change the address of the slave.

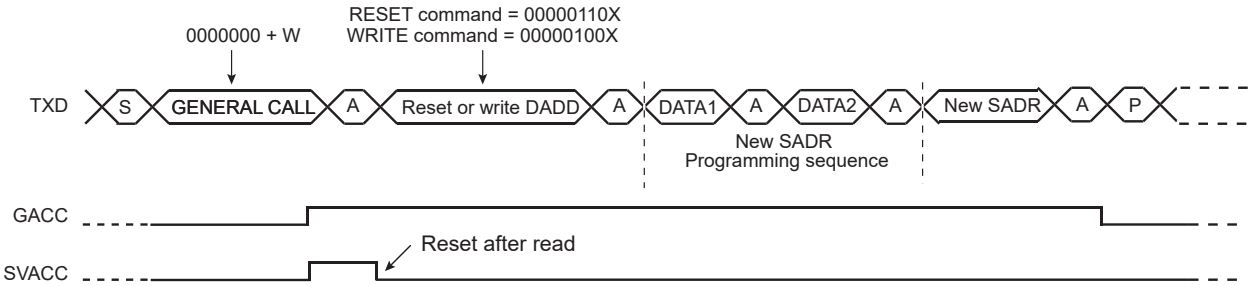
If a GENERAL CALL is detected, GACC is set.

After the detection of general call, decode the commands that follow.

In case of a WRITE command, decode the programming sequence and program a new SADR if the programming sequence matches.

The figure below describes the general call access.

**Figure 43-33. Master Performs a General Call**




**Note:** This method enables the user to create a personal programming sequence by choosing the programming bytes and their number. The programming sequence has to be provided to the master.

#### 43.6.5.4.4 Clock Stretching

In both Read and Write modes, it may occur that TWIHS_THR/TWIHS_RHR buffer is not filled/emptied before the transmission/reception of a new character. In this case, to avoid sending/receiving undesired data, a clock stretching mechanism is implemented.

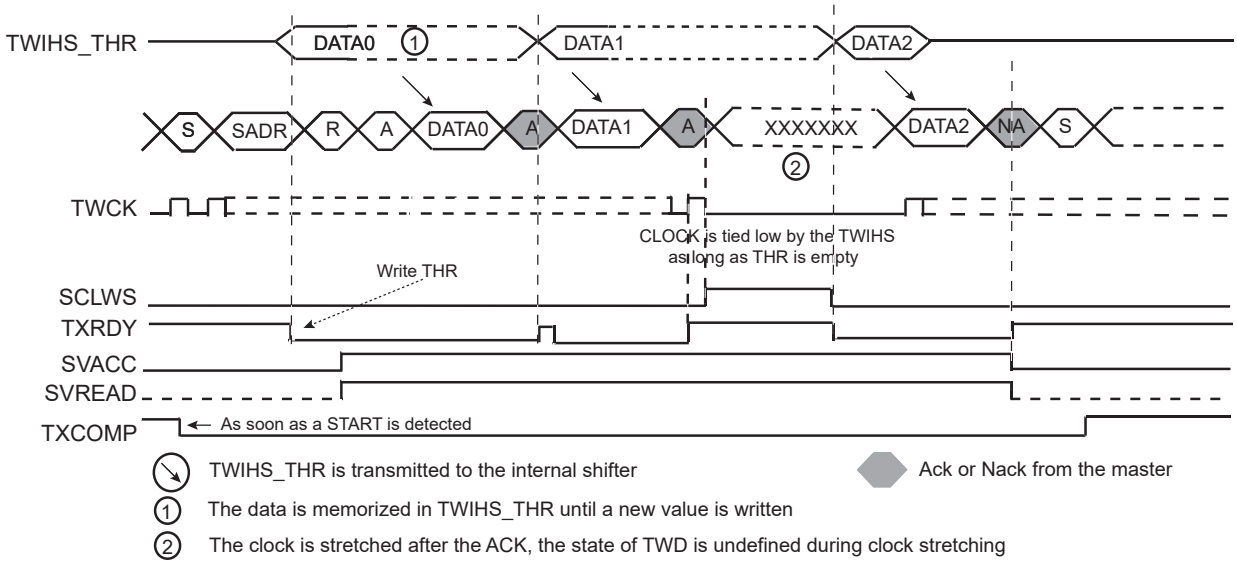
**Note:** Clock stretching can be disabled by setting TWIHS_SMR.SCLWSDIS. In that case the UNRE and OVRE flags indicate an underrun (when TWIHS_THR is not filled on time) or an overrun (when TWIHS_RHR is not read on time).

**Clock Stretching in Read Mode**

The clock is tied low if the internal shifter is empty and if a STOP or REPEATED START condition was not detected. It is tied low until the internal shifter is loaded.

The figure below describes the clock stretching in Read mode.

**Figure 43-34. Clock Stretching in Read Mode**




**Notes:**

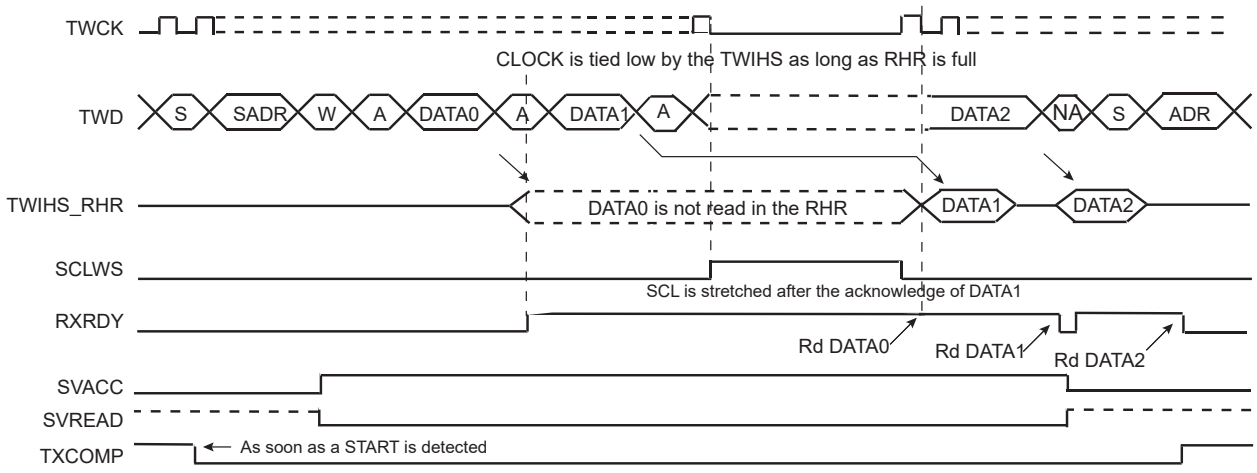
1. TXRDY is reset when data has been written in TWIHS_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.
2. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
3. SCLWS is automatically set when the clock stretching mechanism is started.

**Clock Stretching in Write Mode**

The clock is tied low if the internal shifter and TWIHS_RHR is full. If a STOP or REPEATED_START condition was not detected, it is tied low until TWIHS_RHR is read.

The figure below describes the clock stretching in Write mode.

**Figure 43-35. Clock Stretching in Write Mode**




**Notes:**

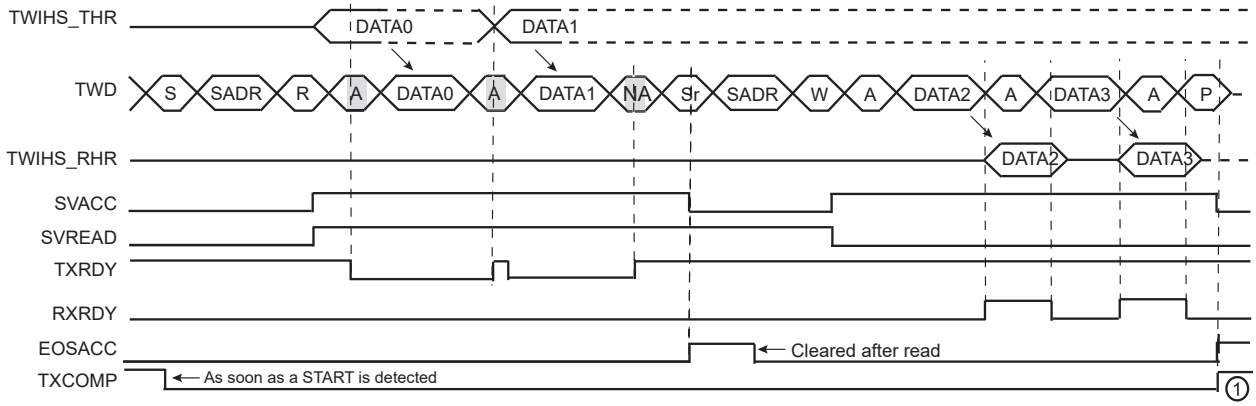
1. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
2. SCLWS is automatically set when the clock stretching mechanism is started and automatically reset when the mechanism is finished.

#### 43.6.5.4.5 Reversal after a Repeated Start

**Reversal of Read to Write**
The master initiates the communication by a read command and finishes it by a write command.

The figure below describes the REPEATED START and the reversal from Read mode to Write mode.

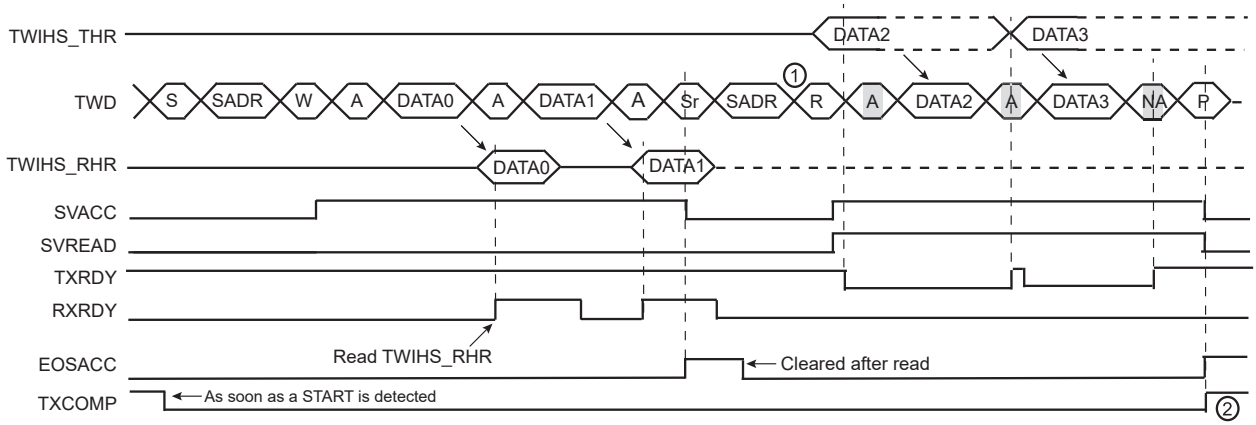
**Figure 43-36. Repeated Start and Reversal from Read Mode to Write Mode**




**Note:** TXCOMP is only set at the end of the transmission. This is because after the REPEATED START, SADR is detected again.

**Reversal of Write to Read**
The master initiates the communication by a write command and finishes it by a read command. The figure below describes the REPEATED START and the reversal from Write mode to Read mode.

**Figure 43-37. Repeated Start and Reversal from Write Mode to Read Mode**




**Notes:**

1. In this case, if TWIHS_THR has not been written at the end of the read command, the clock is automatically stretched before the ACK.
2. TXCOMP is only set at the end of the transmission. This is because after the REPEATED START, SADR is detected again.

### 43.6.5.5 Using the DMA Controller (DMAC) in Slave Mode

The use of the DMAC significantly reduces the CPU load.

#### 43.6.5.5.1 Data Transmit with the DMA in Slave Mode

The following procedure shows an example to transmit data with DMA.

1. Initialize the transmit DMA (memory pointers, transfer size, etc).
2. Configure the Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.

5. Disable the DMA.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

#### 43.6.5.5.2 Data Receive with the DMA in Slave Mode

The following procedure shows an example to transmit data with DMA where the number of characters to receive is known.

1. Initialize the DMA (channels, memory pointers, size, etc.).
2. Configure the Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.
5. Disable the DMA.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

### 43.6.5.6 SMBus Mode

SMBus mode is enabled when a one is written to TWIHS_CR.SMBEN. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into the TWIHS_SMBTR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A set of addresses have been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring the TWIHS_CR.

#### 43.6.5.6.1 Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing a one to TWIHS_CR.PECEN will send/check the PEC field in the current transfer. The PEC generator is always updated on every bit transmitted or received, so that PEC handling on the following linked transfers is correct.

In Slave Receiver mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave compares it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave returns an ACK to the master. If the PEC values differ, data was corrupted, and the slave returns a NACK value. TWIHS_SR.PECERR is set automatically if a PEC error occurred.

In Slave Transmitter mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master compares it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the master must take appropriate action.

See Slave Read Write Flowcharts for detailed flowcharts.

#### 43.6.5.6.2 Timeouts

The TWIHS SMBus Timing Register (TWIHS_SMBTR) configures the SMBus timeout values. If a timeout occurs, the slave leaves the bus. The TOUT bit is also set in TWIHS_SR.

### 43.6.5.7 High-Speed Slave Mode

High-speed mode is enabled when a one is written to TWIHS_CR.HSEN. Furthermore, the analog pad filter must be enabled, a one must be written to TWIHS_FILTR.PADFEN and the FILT bit must be cleared. TWIHS High-speed mode operation is similar to TWIHS operation with the following exceptions:

1. A master code is received first at normal speed before entering High-speed mode period.
2. When TWIHS High-speed mode is active, clock stretching is only allowed after acknowledge (ACK), not-acknowledge (NACK), START (S) or REPEATED START (Sr) (as consequence OVF may happen).

TWIHS High-speed mode allows transfers of up to 3.4 Mbit/s.

The TWIHS slave in High-speed mode requires that slave clock stretching is disabled (SCLWSDIS bit at ‘1’). The peripheral clock must run at a minimum of 11 MHz (assuming the system has no latency).

© 2021 Microchip Technology Inc. and its subsidiaries

**Note:** When slave clock stretching is disabled, the TWIHS_RHR must always be read before receiving the next data (MASTER write frame). It is strongly recommended to use either the polling method on the RXRDY flag in TWIHS_SR, or the DMA. If the receive is managed by an interrupt, the TWIHS interrupt priority must be set to the right level and its latency minimized to avoid receive overrun.

**Note:** When slave clock stretching is disabled, the TWIHS_THR must be filled with the first data to send before the beginning of the frame (MASTER read frame). It is strongly recommended to use either the polling method on the TXRDY flag in TWIHS_SR, or the DMA. If the transmit is managed by an interrupt, the TWIHS interrupt priority must be set to the right level and its latency minimized to avoid transmit underrun.

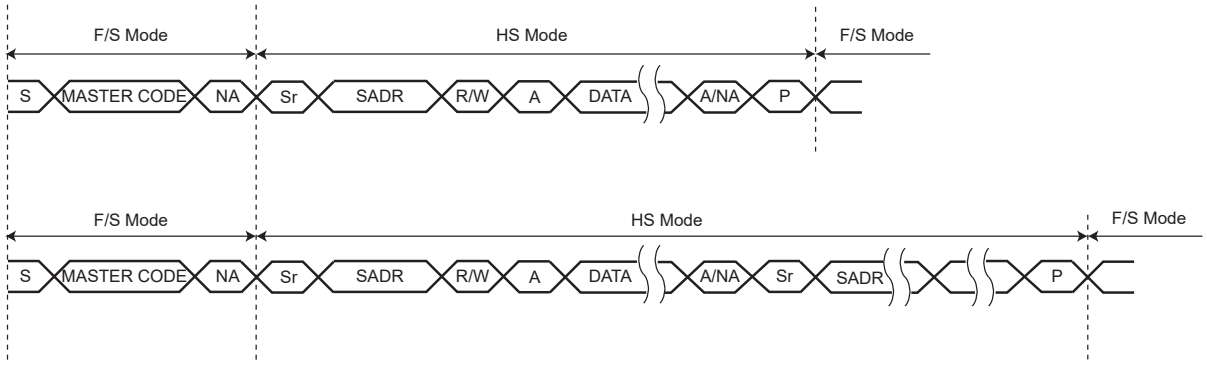
#### 43.6.5.7.1 Read/Write Operation

A TWIHS high-speed frame always begins with the following sequence:

1. START condition (S)
2. Master Code (0000 1XXX)
3. Not-acknowledge (NACK)

When the TWIHS is programmed in Slave mode and TWIHS High-speed mode is activated, master code matching is activated and internal timings are set to match the TWIHS High-speed mode requirements.

**Figure 43-38. High-Speed Mode Read/Write**




#### 43.6.5.7.2 Usage

TWIHS High-speed mode usage is the same as the standard TWIHS (See Read/Write Flowcharts).

### 43.6.5.8 Asynchronous Partial Wakeup (SleepWalking)

The TWIHS includes an asynchronous start condition detector. It is capable of waking the device up from a Sleep mode upon an address match (and optionally an additional data match), including Sleep modes where the TWIHS peripheral clock is stopped.

After detecting the START condition on the bus, the TWIHS stretches TWCK until the TWIHS peripheral clock has started. The time required for starting the TWIHS depends on which Sleep mode the device is in. After the TWIHS peripheral clock has started, the TWIHS releases its TWCK stretching and receives one byte of data (slave address) on the bus. At this time, only a limited part of the device, including the TWIHS module, receives a clock, thus saving power. If the address phase causes a TWIHS address match (and, optionally, if the first data byte causes data match as well), the entire device is woken up and normal TWIHS address matching actions are performed. Normal TWIHS transfer then follows. If the TWIHS is not addressed (or if the optional data match fails), the TWIHS peripheral clock is automatically stopped and the device returns to its original Sleep mode.

The TWIHS has the capability to match on more than one address. The SADR1EN, SADR2EN and SADR3EN bits in TWIHS_SMR enable address matching on additional addresses which can be configured through SADR1, SADR2 and SADR3 fields in the TWIHS_SWMR. The SleepWalking matching process can be extended to the first received data byte if TWIHS_SMR.DATAMEN is set and, in this case, a complete matching includes address matching and first received data matching. TWIHS_SWMR.DATAM configures the data to match on the first received byte.

When the system is in Active mode and the TWIHS enters Asynchronous Partial Wakeup mode, the flag SVACC must be programmed as the unique source of the TWIHS interrupt and the data match comparison must be disabled.

When the system exits Wait mode as the result of a matching condition, the SVACC flag is used to determine if the TWIHS is the source of exit.

**Figure 43-39. Address Match Only (Data Matching Disabled)**

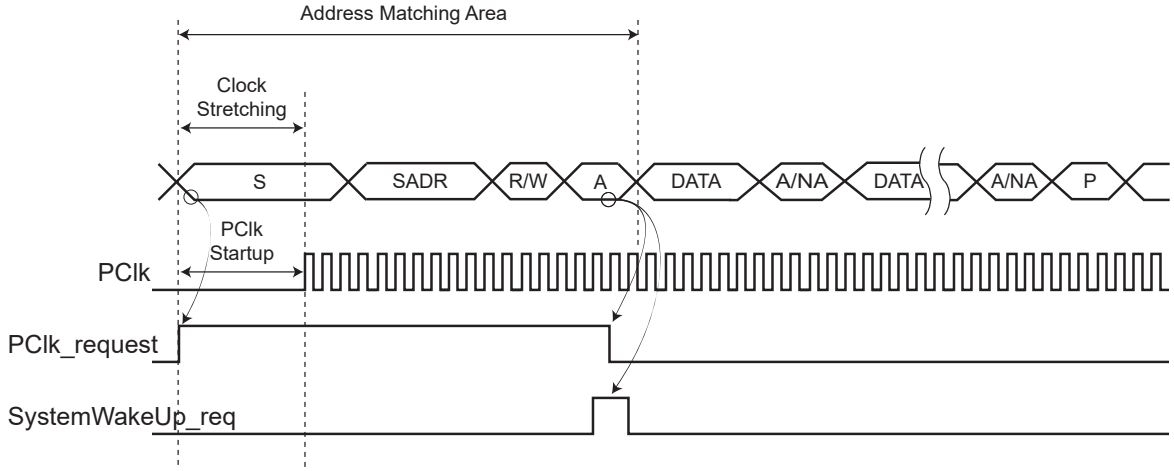



**Figure 43-40. No Address Match (Data Matching Disabled)**

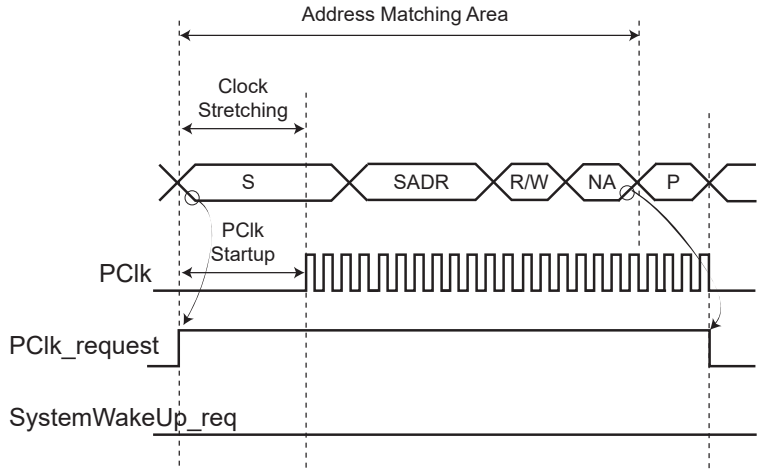



**Figure 43-41. Address Match and Data Match (Data Matching Enabled)**

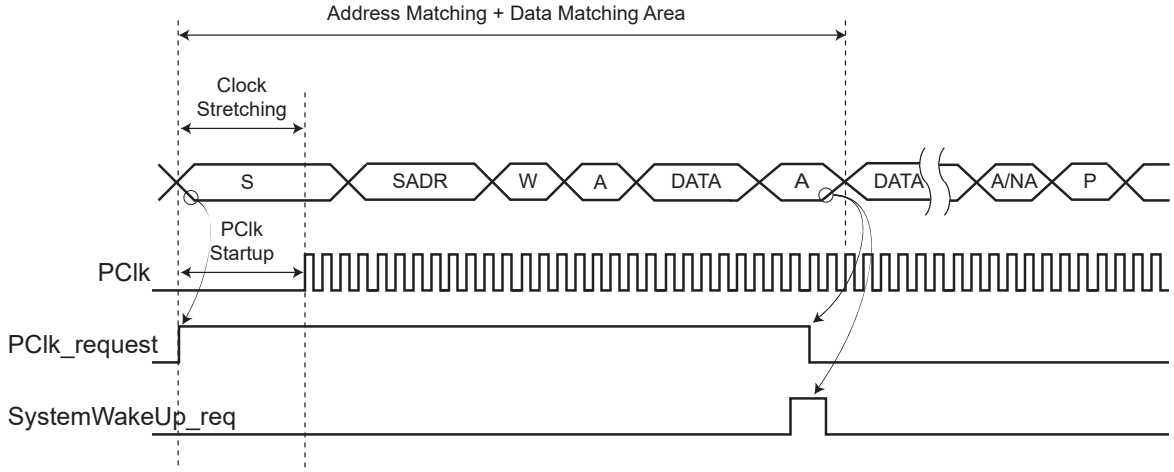
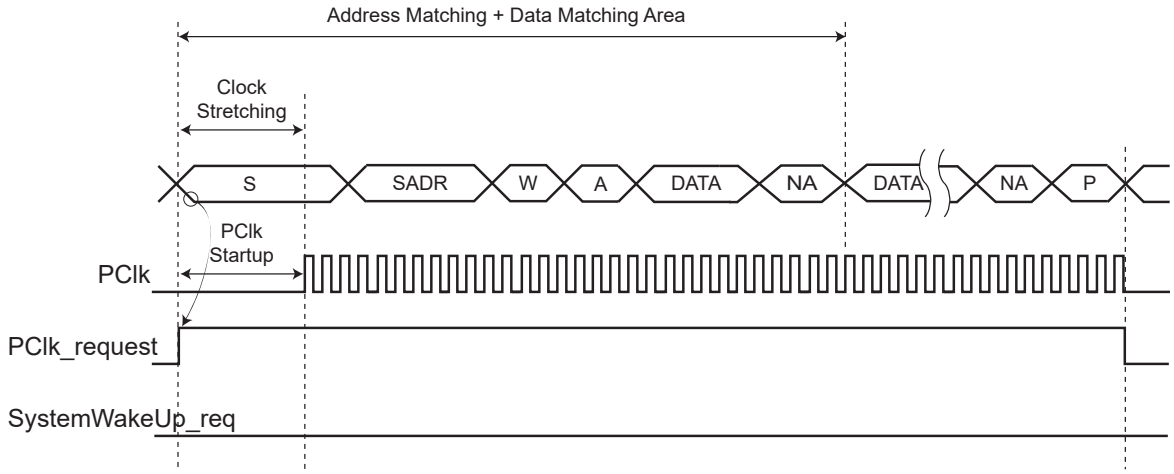


**Figure 43-42. Address Match and No Data Match (Data Matching Enabled)**




### 43.6.5.9 Slave Read Write Flowcharts

The flowchart below illustrates an example of read and write operations in Slave mode. A polling or interrupt method can be used to check the status bits. The interrupt method requires that TWIHS_IER be configured first.

**Figure 43-43. Read Write Flowchart in Slave Mode**

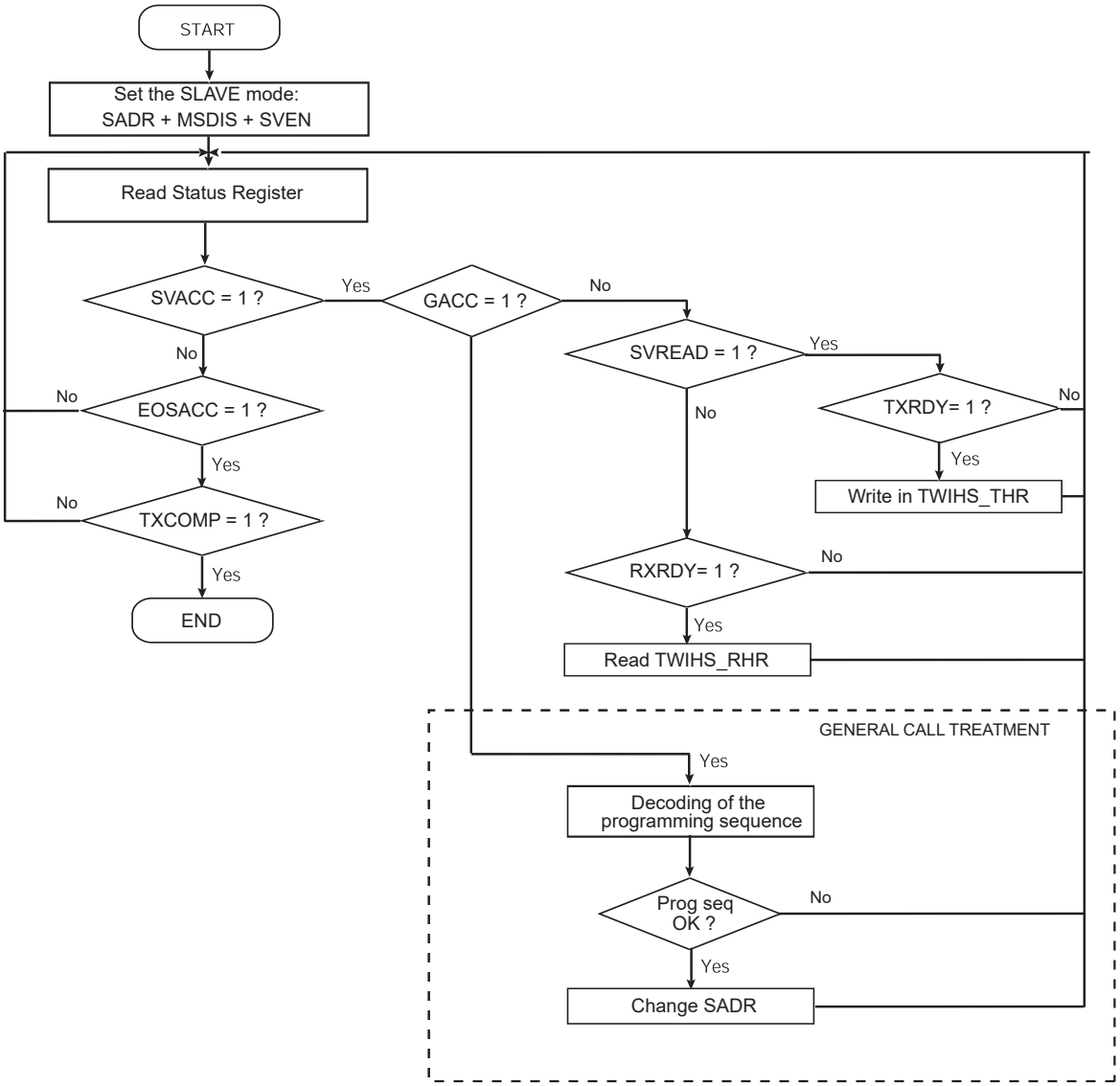


**Figure 43-44. Read Write Flowchart in Slave Mode with SMBus PEC**

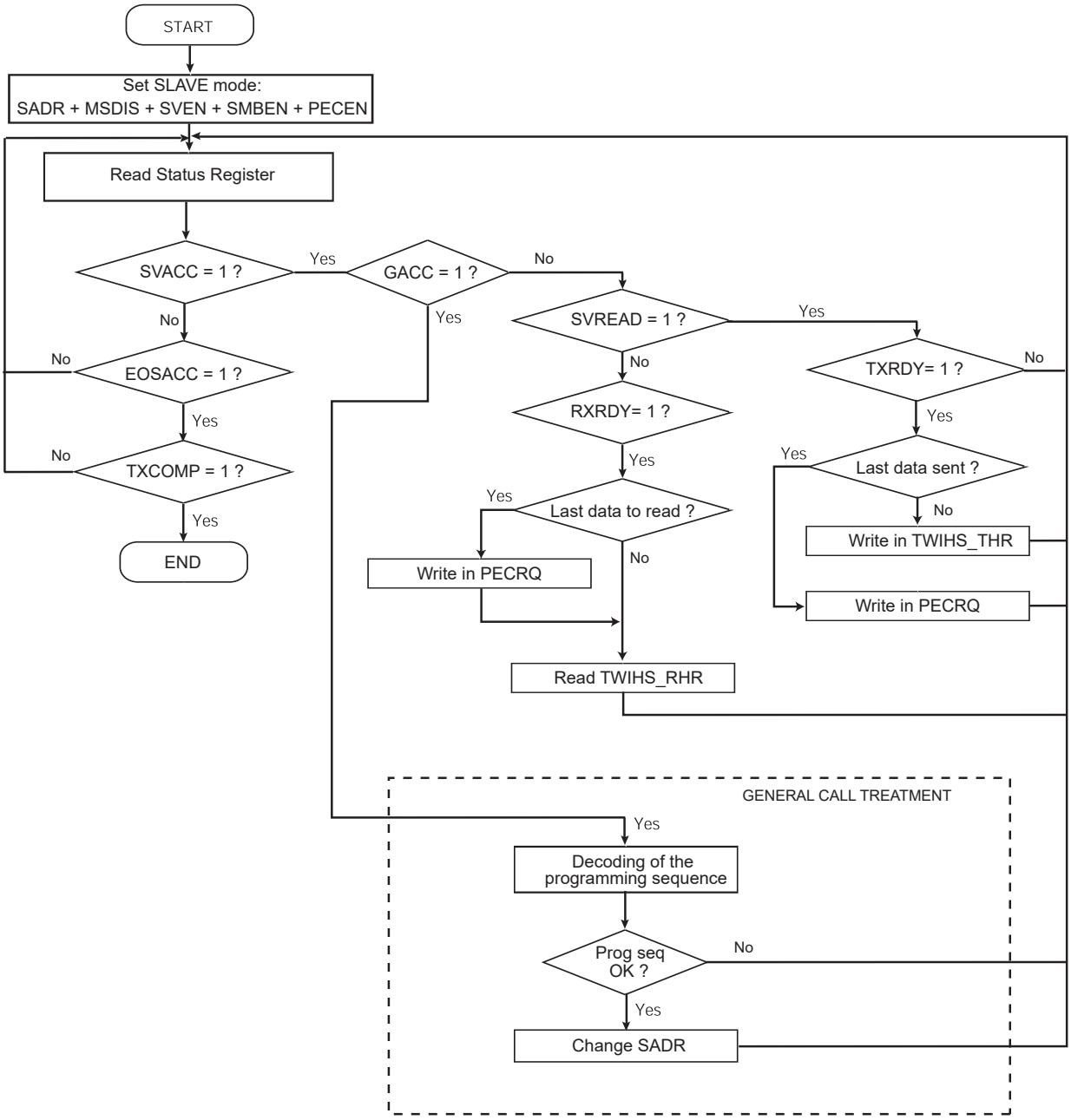
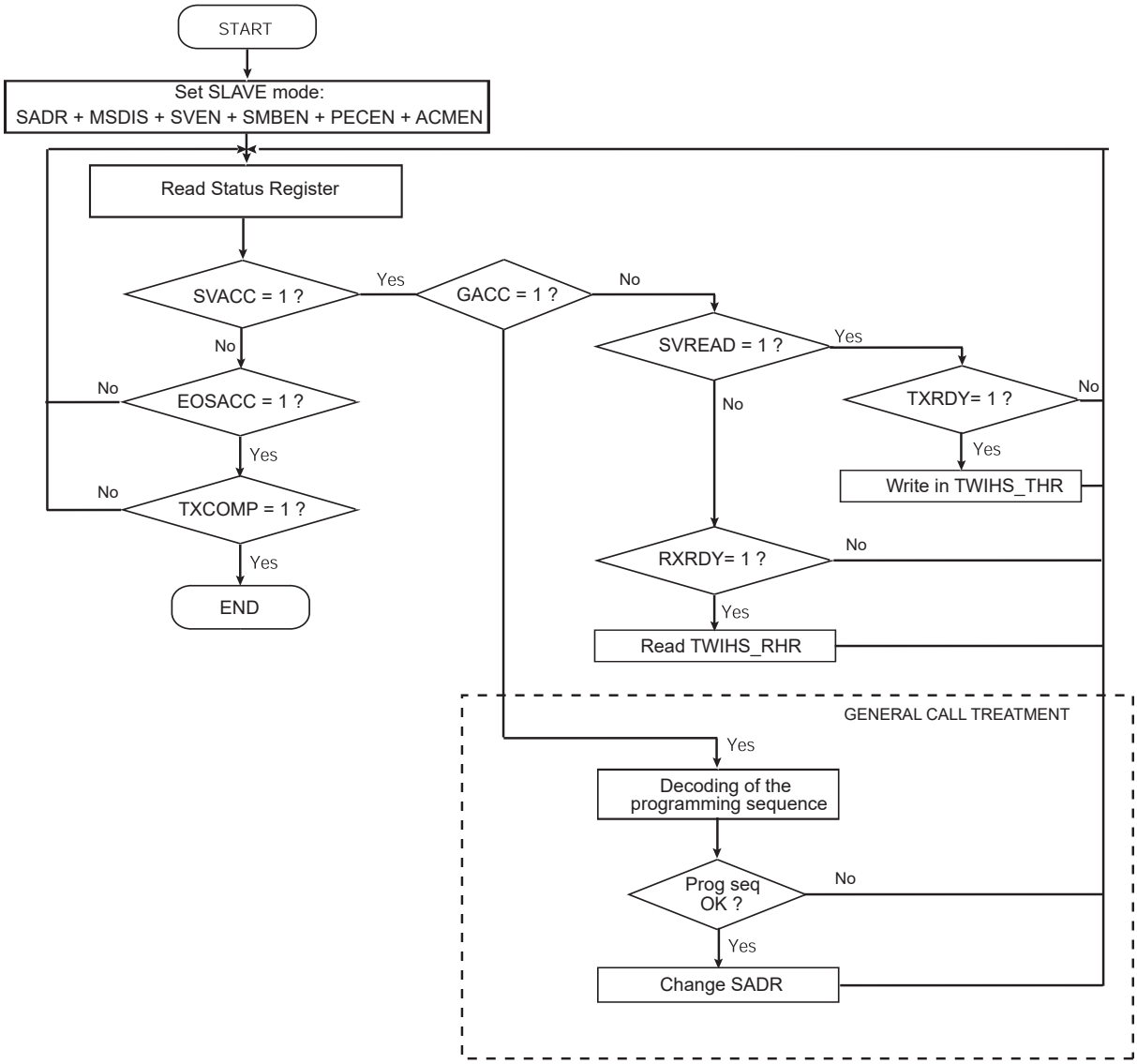


**Figure 43-45. Read Write Flowchart in Slave Mode with SMBus PEC and Alternative Command Mode**




### 43.6.6 TWIHS Comparison Function on Received Character

The comparison function differs if asynchronous partial wakeup (SleepWalking) is enabled or not.

If asynchronous partial wakeup is disabled (see the section “Power Management Controller (PMC)”), the TWIHS can extend the address matching on up to three slave addresses. The SADR1EN, SADR2EN and SADR3EN bits in TWIHS_SMR enable address matching on additional addresses which can be configured through SADR1, SADR2 and SADR3 fields in the TWIHS_SWMR. The DATAMEN bit in the TWIHS_SMR has no effect.

The SVACC bit is set when there is a comparison match with the received slave address.

### 43.6.7 Register Write Protection

To prevent any single software error from corrupting TWIHS behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the TWIHS Write Protection Mode Register (TWIHS_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the TWIHS Write Protection Status Register (TWIHS_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading TWIHS_WPSR.

The following registers can be write-protected:

- TWIHS Clock Waveform Generator Register

© 2021 Microchip Technology Inc.
and its subsidiaries

## 43.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | TWIHS_CR | 7:0 | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| | | 15:8 | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| | | 23:16 | | | | | | | ACMDIS | ACMEN |
| | | 31:24 | | | FIFODIS | FIFOEN | | LOCKCLR | | THRCLR |
| 0x04 | TWIHS_MMR | 7:0 | | | | | | | | |
| | | 15:8 | | | | MREAD | | | IADRSZ[1:0] | |
| | | 23:16 | | DADR[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | TWIHS_SMR | 7:0 | | SCLWSDIS | | | SMHH | SMDA | | NACKEN |
| | | 15:8 | | MASK[6:0] | | | | | | |
| | | 23:16 | | SADR[6:0] | | | | | | |
| | | 31:24 | DATAMEN | SADR3EN | SADR2EN | SADR1EN | | | | |
| 0x0C | TWIHS_IADR | 7:0 | IADR[7:0] | | | | | | | |
| | | 15:8 | IADR[15:8] | | | | | | | |
| | | 23:16 | IADR[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | TWIHS_CWGR | 7:0 | CLDIV[7:0] | | | | | | | |
| | | 15:8 | CHDIV[7:0] | | | | | | | |
| | | 23:16 | | | | | | CKDIV[2:0] | | |
| | | 31:24 | | | HOLD[5:0] | | | | | |
| 0x14 ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | TWIHS_SR | 7:0 | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCLWS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | SDA | SCL |
| 0x24 | TWIHS_IER | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |
| 0x28 | TWIHS_IDR | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |
| 0x2C | TWIHS_IMR | 7:0 | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| | | 15:8 | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| | | 23:16 | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| | | 31:24 | | | | | | | | |
| 0x30 | TWIHS_RHR | 7:0 | RXDATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | TWIHS_THR | 7:0 | TXDATA[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | TWIHS_SMBTR | 7:0 | | | | | PRESC[3:0] | | | |
| | | 15:8 | TLOWS[7:0] | | | | | | | |
| | | 23:16 | TLOWM[7:0] | | | | | | | |
| | | 31:24 | THMAX[7:0] | | | | | | | |
| 0x3C ... 0x43 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x44 | TWIHS_FILTR | 7:0 | | | | | | PADFCFG | PADFEN | FILT |
| | | 15:8 | | | | | | THRES[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 ... 0x4B | Reserved | | | | | | | | | |
| 0x4C | TWIHS_SWMR | 7:0 | | SADR1[6:0] | | | | | | |
| | | 15:8 | | SADR2[6:0] | | | | | | |
| | | 23:16 | | SADR3[6:0] | | | | | | |
| | | 31:24 | DATAM[7:0] | | | | | | | |
| 0x50 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | TWIHS_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | TWIHS_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | WPVSRC[23:16] | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.1 TWIHS Control Register

**Name:** TWIHS_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | FIFODIS | FIFOEN | | LOCKCLR | | THRCLR |
| Access | | | W | W | | W | | W |
| Reset | | | – | – | | – | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ACMDIS | ACMEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CLEAR | PECRQ | PECDIS | PECEN | SMBDIS | SMBEN | HSDIS | HSEN |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | QUICK | SVDIS | SVEN | MSDIS | MSEN | STOP | START |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 29 – FIFODIS** FIFO Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit and Receive FIFOs. |

**Bit 28 – FIFOEN** FIFO Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit and Receive FIFOs. |

**Bit 26 – LOCKCLR** Lock Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the TWIHS FSM lock. |

**Bit 24 – THRCLR** Transmit Holding Register Clear

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clears the Transmit Holding Register and sets TXRDY, TXCOMP flags. |

**Bit 17 – ACMDIS** Alternative Command Mode Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Alternative Command mode disabled. |

**Bit 16 – ACMEN** Alternative Command Mode Enable

| Value | Description |
|---|---|
| 0 | No effect. |

| Value | Description |
|---|---|
| 1 | Alternative Command mode enabled. |

**Bit 15 – CLEAR** Bus CLEAR Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, sends a bus clear command. |

**Bit 14 – PECRQ** PEC Request

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A PEC check or transmission is requested. |

**Bit 13 – PECDIS** Packet Error Checking Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check disabled. |

**Bit 12 – PECEN** Packet Error Checking Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus PEC (CRC) generation and check enabled. |

**Bit 11 – SMBDIS** SMBus Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | SMBus mode disabled. |

**Bit 10 – SMBEN** SMBus Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If SMBDIS = 0, SMBus mode enabled. |

**Bit 9 – HSDIS** TWIHS High-Speed Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode disabled. |

**Bit 8 – HSEN** TWIHS High-Speed Mode Enabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | High-speed mode enabled. |

**Bit 7 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Equivalent to a system reset. |

**Bit 6 – QUICK** SMBus Quick Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | If Master mode is enabled, a SMBus Quick Command is sent. |

**Bit 5 – SVDIS** TWIHS Slave Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | The Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in case of read operation. In write operation, the character being transferred must be completely received before disabling. |

**Bit 4 – SVEN** TWIHS Slave Mode Enabled
Switching from Master to Slave mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Slave mode (SVDIS must be written to 0). |

**Bit 3 – MSDIS** TWIHS Master Mode Disabled

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in case of write operation. In read operation, the character being transferred must be completely received before disabling. |

**Bit 2 – MSEN** TWIHS Master Mode Enabled
Switching from Slave to Master mode is only permitted when TXCOMP = 1.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Master mode (MSDIS must be written to 0). |

**Bit 1 – STOP** Send a STOP Condition

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | STOP condition is sent just after completing the current byte transmission in Master Read mode.<br>• In single data byte master read, both START and STOP must be set.<br>• In multiple data bytes master read, the STOP must be set after the last data received but one.<br>• In Master Read mode, if a NACK bit is received, the STOP is automatically performed.<br>• In master data write operation, a STOP condition will be sent after the transmission of the current data is finished. |

**Bit 0 – START** Send a START Condition
This action is necessary when the TWIHS peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (TWIHS_THR).

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A frame beginning with a START bit is transmitted according to the features defined in the TWIHS Master Mode Register (TWIHS_MMR). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.2 TWIHS Master Mode Register

**Name:** TWIHS_MMR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | DADR[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | MREAD | | | IADRSZ[1:0] | |
| Access | | | | R/W | | | R/W | R/W |
| Reset | | | | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bits 22:16 – DADR[6:0]** Device Address
The device address is used to access slave devices in Read or Write mode. These bits are only used in Master mode.

**Bit 12 – MREAD** Master Read Direction

| Value | Description |
|---|---|
| 0 | Master write direction. |
| 1 | Master read direction. |

**Bits 9:8 – IADRSZ[1:0]** Internal Device Address Size

| Value | Name | Description |
|---|---|---|
| 0 | NONE | No internal device address |
| 1 | 1_BYTE | One-byte internal device address |
| 2 | 2_BYTE | Two-byte internal device address |
| 3 | 3_BYTE | Three-byte internal device address |

### 43.7.3 TWIHS Slave Mode Register

**Name:** TWIHS_SMR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATAMEN | SADR3EN | SADR2EN | SADR1EN | | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SADR[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | MASK[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SCLWSDIS | | | SMHH | SMDA | | NACKEN |
| Access | | R/W | | | R/W | R/W | | R/W |
| Reset | | 0 | | | 0 | 0 | | 0 |

**Bit 31 – DATAMEN** Data Matching Enable

| Value | Description |
|---|---|
| 0 | Data matching on first received data is disabled. |
| 1 | Data matching on first received data is enabled. |

**Bit 30 – SADR3EN** Slave Address 3 Enable

| Value | Description |
|---|---|
| 0 | Slave address 3 matching is disabled. |
| 1 | Slave address 3 matching is enabled. |

**Bit 29 – SADR2EN** Slave Address 2 Enable

| Value | Description |
|---|---|
| 0 | Slave address 2 matching is disabled. |
| 1 | Slave address 2 matching is enabled. |

**Bit 28 – SADR1EN** Slave Address 1 Enable

| Value | Description |
|---|---|
| 0 | Slave address 1 matching is disabled. |
| 1 | Slave address 1 matching is enabled. |

**Bits 22:16 – SADR[6:0]** Slave Address
The slave device address is used in Slave mode in order to be accessed by master devices in Read or Write mode. SADR must be programmed before enabling the Slave mode or after a general call. Writes at other times have no effect.

**Bits 14:8 – MASK[6:0]** Slave Address Mask
A mask can be applied on the slave device address in Slave mode in order to allow multiple address answer. For each bit of the MASK field set to 1, the corresponding SADR bit is masked.
If the MASK field value is 0, no mask is applied to the SADR field.

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 6 – SCLWSDIS** Clock Wait State Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Clock stretching disabled in Slave mode, OVRE and UNRE indicate an overrun/underrun. |

**Bit 3 – SMHH** SMBus Host Header

| Value | Description |
|---|---|
| 0 | Acknowledge of the SMBus host header disabled. |
| 1 | Acknowledge of the SMBus host header enabled. |

**Bit 2 – SMDA** SMBus Default Address

| Value | Description |
|---|---|
| 0 | Acknowledge of the SMBus default address disabled. |
| 1 | Acknowledge of the SMBus default address enabled. |

**Bit 0 – NACKEN** Slave Receiver Data Phase NACK enable

| Value | Description |
|---|---|
| 0 | Normal value to be returned in the ACK cycle of the data phase in Slave Receiver mode. |
| 1 | NACK value to be returned in the ACK cycle of the data phase in Slave Receiver mode. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.4 TWIHS Internal Address Register

**Name:** TWIHS_IADR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IADR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IADR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IADR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – IADR[23:0]** Internal Address
0, 1, 2 or 3 bytes depending on IADRSZ.

### 43.7.5 TWIHS Clock Waveform Generator Register

**Name:** TWIHS_CWGR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

TWIHS_CWGR is used in Master mode only.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | HOLD[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CKDIV[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLDIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:24 – HOLD[5:0]** TWD Hold Time Versus TWCK Falling
If High-speed mode is selected TWD is internally modified on the TWCK falling edge to meet the I2C specified maximum hold time, else if High-speed mode is not configured TWD is kept unchanged after TWCK falling edge for a period of (HOLD + 3) × $t_{peripheral\ clock}$.

**Bits 18:16 – CKDIV[2:0]** Clock Divider
The CKDIV is used to increase both SCL high and low periods.

**Bits 15:8 – CHDIV[7:0]** Clock High Divider
The SCL high period is defined as follows:
$t_{high} = ((CHDIV \times 2^{CKDIV}) + 3) \times t_{peripheral\ clock}$

**Bits 7:0 – CLDIV[7:0]** Clock Low Divider
The SCL low period is defined as follows:
$t_{low} = ((CLDIV \times 2^{CKDIV}) + 3) \times t_{peripheral\ clock}$

### 43.7.6 TWIHS Status Register

**Name:** TWIHS_SR
**Offset:** 0x20
**Reset:** 0x03000009
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SDA | SCL |
| Access | | | | | | | R | R |
| Reset | | | | | | | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | R | R | R | R | | R |
| Reset | | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCLWS | ARBLST | NACK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | SVREAD | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |

**Bit 25 – SDA** SDA Line Value

| Value | Description |
|---|---|
| 0 | SDA line sampled value is ‘0’. |
| 1 | SDA line sampled value is ‘1’. |

**Bit 24 – SCL** SCL Line Value

| Value | Description |
|---|---|
| 0 | SCL line sampled value is ‘0’. |
| 1 | SCL line sampled value is ‘1.’ |

**Bit 21 – SMBHHM** SMBus Host Header Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Host Header Address received since the last read of TWIHS_SR. |
| 1 | An SMBus Host Header Address was received since the last read of TWIHS_SR. |

**Bit 20 – SMBDAM** SMBus Default Address Match (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus Default Address received since the last read of TWIHS_SR. |
| 1 | An SMBus Default Address was received since the last read of TWIHS_SR. |

**Bit 19 – PECERR** PEC Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus PEC error occurred since the last read of TWIHS_SR. |
| 1 | An SMBus PEC error occurred since the last read of TWIHS_SR. |

**Bit 18 – TOUT** Timeout Error (cleared on read)

| Value | Description |
|---|---|
| 0 | No SMBus timeout occurred since the last read of TWIHS_SR. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | An SMBus timeout occurred since the last read of TWIHS_SR. |

**Bit 16 – MCACK** Master Code Acknowledge (cleared on read)
MACK used in Slave mode:

| Value | Description |
|---|---|
| 0 | No Master Code has been received since the last read of TWIHS_SR. |
| 1 | A Master Code has been received since the last read of TWIHS_SR. |

**Bit 11 – EOSACC** End Of Slave Access (cleared on read)
This bit is used in Slave mode only.
EOSACC behavior can be seen in Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | A slave access is being performing. |
| 1 | The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset. |

**Bit 10 – SCLWS** Clock Wait State
This bit is used in Slave mode only.
SCLWS behavior can be seen in the figures, Clock Stretching in Read Mode and Clock Stretching in Write Mode.

| Value | Description |
|---|---|
| 0 | The clock is not stretched. |
| 1 | The clock is stretched. TWIHS_THR / TWIHS_RHR buffer is not filled / emptied before the transmission / reception of a new character. |

**Bit 9 – ARBLST** Arbitration Lost (cleared on read)
This bit is used in Master mode only.

| Value | Description |
|---|---|
| 0 | Arbitration won. |
| 1 | Arbitration lost. Another master of the TWIHS bus has won the multimaster arbitration. TXCOMP is set at the same time. |

**Bit 8 – NACK** Not Acknowledged (cleared on read)

- NACK used in Master mode:

0: Each data byte has been correctly received by the far-end side TWIHS slave component.
1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.

- NACK used in Slave Read mode:

0: Each data byte has been correctly received by the master.
1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set, the user must not fill TWIHS_THR even if TXRDY is set, because it means that the master stops the data transfer or re-initiate it.
**Note:** In Slave Write mode, all data are acknowledged by the TWIHS.

**Bit 7 – UNRE** Underrun Error (cleared on read)
This bit is used only if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | TWIHS_THR has been filled on time. |
| 1 | TWIHS_THR has not been filled on time. |

**Bit 6 – OVRE** Overrun Error (cleared on read)
This bit is used only if clock stretching is disabled.

| Value | Description |
|---|---|
| 0 | TWIHS_RHR has not been loaded while RXRDY was set. |
| 1 | TWIHS_RHR has been loaded while RXRDY was set. Reset by read in TWIHS_SR when TXCOMP is set. |

**Bit 5 – GACC** General Call Access (cleared on read)
This bit is used in Slave mode only.
GACC behavior can be seen in Master Performs a General Call.

| Value | Description |
|---|---|
| 0 | No general call has been detected. |
| 1 | A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes. |

**Bit 4 – SVACC** Slave Access
This bit is used in Slave mode only.
SVACC behavior can be seen in Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | TWIHS is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected. |
| 1 | Indicates that the address decoding sequence has matched (A master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected. |

**Bit 3 – SVREAD** Slave Read
This bit is used in Slave mode only. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.
SVREAD behavior can be seen in Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | Indicates that a write access is performed by a master. |
| 1 | Indicates that a read access is performed by a master. |

**Bit 2 – TXRDY** Transmit Holding Register Ready (cleared by writing TWIHS_THR)

- TXRDY used in Master mode:

0: The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into TWIHS_THR.
1: As soon as a data byte is transferred from TWIHS_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWIHS).
TXRDY behavior in Master mode can be seen in Master Write with One Data Byte, Master Write with Multiple Data Bytes and Master Write with One-Byte Internal Address and Multiple Data Bytes.

- TXRDY used in Slave mode:

0: As soon as data is written in the TWIHS_THR, until this data has been transmitted and acknowledged (ACK or NACK).
1: Indicates that the TWIHS_THR is empty and that data has been transmitted and acknowledged.
If TXRDY is high and if a NACK has been detected, the transmission is stopped. Thus when TRDY = NACK = 1, the user must not fill TWIHS_THR to avoid losing it.
TXRDY behavior in Slave mode can be seen in Read Access Ordered by a Master, Clock Stretching in Read Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

**Bit 1 – RXRDY** Receive Holding Register Ready (cleared by reading TWIHS_RHR)
RXRDY behavior in Master mode can be seen in Master Read with One Data Byte, Master Read with Multiple Data Bytes and Master Read Clock Stretching with Multiple Data Bytes.
RXRDY behavior in Slave mode can be seen in Write Access Ordered by a Master, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

| Value | Description |
|---|---|
| 0 | No character has been received since the last TWIHS_RHR read operation. |
| 1 | A byte has been received in the TWIHS_RHR since the last read. |

**Bit 0 – TXCOMP** Transmission Completed (cleared by writing TWIHS_THR)

- TXCOMP used in Master mode:

0: During the length of the current frame.
1: When both holding register and internal shifter are empty and STOP condition has been sent.
TXCOMP behavior in Master mode can be seen in Master Write with One-Byte Internal Address and Multiple Data Bytes and in Master Read with Multiple Data Bytes.

- TXCOMP used in Slave mode:

0: As soon as a START is detected.
1: After a STOP or a REPEATED START + an address different from SADR is detected.
TXCOMP behavior in Slave mode can be seen in Clock Stretching in Read Mode, Clock Stretching in Write Mode, Repeated Start and Reversal from Read Mode to Write Mode and Repeated Start and Reversal from Write Mode to Read Mode.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.7 TWIHS SMBus Timing Register

**Name:** TWIHS_SMBTR
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | THMAX[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TLOWM[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TLOWS[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRESC[3:0] | | | |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 31:24 – THMAX[7:0]** Clock High Maximum Cycles
Clock cycles in clock high maximum count. Prescaled by PRESC. Used for bus free detection. Used to time THIGH:MAX.

**Bits 23:16 – TLOWM[7:0]** Master Clock Stretch Maximum Cycles

| Value | Description |
|---|---|
| 0 | TLOW:MEXT timeout check disabled. |
| 1-255 | Clock cycles in master maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:MEXT. |

**Bits 15:8 – TLOWS[7:0]** Slave Clock Stretch Maximum Cycles

| Value | Description |
|---|---|
| 0 | TLOW:SEXT timeout check disabled. |
| 1-255 | Clock cycles in slave maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:SEXT. |

**Bits 3:0 – PRESC[3:0]** SMBus Clock Prescaler
Used to specify how to prescale the TLOWS, TLOWM and THMAX counters in SMBTR. Counters are prescaled according to the following formula:

$$f_{\text{Prescaled}} = \frac{f_{\text{peripheral clock}}}{2^{(\text{PRESC}+1)}}$$

### 43.7.8 TWIHS Filter Register

**Name:** TWIHS_FILTR
**Offset:** 0x44
**Reset:** 0x00000000
**Property:** Read/Write

TWIHS digital input filtering follows a majority decision based on three samples from SDA/SCL lines at peripheral clock frequency.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | THRES[2:0] | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | PADFCFG | PADFEN | FILT |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 10:8 – THRES[2:0]** Digital Filter Threshold

| Value | Description |
|---|---|
| 0 | No filtering applied on TWIHS inputs. |
| 1–7 | Maximum pulse width of spikes to be suppressed by the input filter, defined in peripheral clock cycles. |

**Bit 2 – PADFCFG** PAD Filter Config
See the electrical characteristics section for filter configuration details.

**Bit 1 – PADFEN** PAD Filter Enable

| Value | Description |
|---|---|
| 0 | PAD analog filter is disabled. |
| 1 | PAD analog filter is enabled. (The analog filter must be enabled if High-speed mode is enabled.) |

**Bit 0 – FILT** RX Digital Filter

| Value | Description |
|---|---|
| 0 | No filtering applied on TWIHS inputs. |
| 1 | TWIHS input filtering is active (only in Standard and Fast modes) |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.9 TWIHS Interrupt Enable Register

**Name:** TWIHS_IER
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | W | W | W | W | | W |
| Reset | | | – | – | – | – | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | W | W | W | W | | W | W | W |
| Reset | – | – | – | – | | – | – | – |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Enable

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Enable

**Bit 19 – PECERR** PEC Error Interrupt Enable

**Bit 18 – TOUT** Timeout Error Interrupt Enable

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Enable

**Bit 11 – EOSACC** End Of Slave Access Interrupt Enable

**Bit 10 – SCL_WS** Clock Wait State Interrupt Enable

**Bit 9 – ARBLST** Arbitration Lost Interrupt Enable

**Bit 8 – NACK** Not Acknowledge Interrupt Enable

**Bit 7 – UNRE** Underrun Error Interrupt Enable

**Bit 6 – OVRE** Overrun Error Interrupt Enable

**Bit 5 – GACC** General Call Access Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 4 – SVACC** Slave Access Interrupt Enable

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Enable

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Enable

**Bit 0 – TXCOMP** Transmission Completed Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.10 TWIHS Interrupt Disable Register

**Name:** TWIHS_IDR
**Offset:** 0x28
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | W | W | W | W | | W |
| Reset | | | – | – | – | – | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | W | W | W | W | | W | W | W |
| Reset | – | – | – | – | | – | – | – |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Disable

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Disable

**Bit 19 – PECERR** PEC Error Interrupt Disable

**Bit 18 – TOUT** Timeout Error Interrupt Disable

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Disable

**Bit 11 – EOSACC** End Of Slave Access Interrupt Disable

**Bit 10 – SCL_WS** Clock Wait State Interrupt Disable

**Bit 9 – ARBLST** Arbitration Lost Interrupt Disable

**Bit 8 – NACK** Not Acknowledge Interrupt Disable

**Bit 7 – UNRE** Underrun Error Interrupt Disable

**Bit 6 – OVRE** Overrun Error Interrupt Disable

**Bit 5 – GACC** General Call Access Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 4 – SVACC** Slave Access Interrupt Disable

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Disable

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Disable

**Bit 0 – TXCOMP** Transmission Completed Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.11 TWIHS Interrupt Mask Register

**Name:** TWIHS_IMR
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | SMBHHM | SMBDAM | PECERR | TOUT | | MCACK |
| Access | | | R | R | R | R | | R |
| Reset | | | 0 | 0 | 0 | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOSACC | SCL_WS | ARBLST | NACK |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UNRE | OVRE | GACC | SVACC | | TXRDY | RXRDY | TXCOMP |
| Access | R | R | R | R | | R | R | R |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 21 – SMBHHM** SMBus Host Header Address Match Interrupt Mask

**Bit 20 – SMBDAM** SMBus Default Address Match Interrupt Mask

**Bit 19 – PECERR** PEC Error Interrupt Mask

**Bit 18 – TOUT** Timeout Error Interrupt Mask

**Bit 16 – MCACK** Master Code Acknowledge Interrupt Mask

**Bit 11 – EOSACC** End Of Slave Access Interrupt Mask

**Bit 10 – SCL_WS** Clock Wait State Interrupt Mask

**Bit 9 – ARBLST** Arbitration Lost Interrupt Mask

**Bit 8 – NACK** Not Acknowledge Interrupt Mask

**Bit 7 – UNRE** Underrun Error Interrupt Mask

**Bit 6 – OVRE** Overrun Error Interrupt Mask

**Bit 5 – GACC** General Call Access Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 4 – SVACC** Slave Access Interrupt Mask

**Bit 2 – TXRDY** Transmit Holding Register Ready Interrupt Mask

**Bit 1 – RXRDY** Receive Holding Register Ready Interrupt Mask

**Bit 0 – TXCOMP** Transmission Completed Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.12 TWIHS Receive Holding Register

**Name:** TWIHS_RHR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXDATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RXDATA[7:0]** Master or Slave Receive Holding Data

### 43.7.13 TWIHS SleepWalking Matching Register

**Name:** TWIHS_SWMR
**Offset:** 0x4C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATAM[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SADR3[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | SADR2[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SADR1[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – DATAM[7:0]** Data Match
The TWIHS module extends the SleepWalking matching process to the first received data, comparing it with DATAM if DATAMEN bit is enabled.

**Bits 22:16 – SADR3[6:0]** Slave Address 3
Slave address 3. The TWIHS module matches on this additional address if SADR3EN bit is enabled.

**Bits 14:8 – SADR2[6:0]** Slave Address 2
Slave address 2. The TWIHS module matches on this additional address if SADR2EN bit is enabled.

**Bits 6:0 – SADR1[6:0]** Slave Address 1
Slave address 1. The TWIHS module matches on this additional address if SADR1EN bit is enabled.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 43.7.14 TWIHS Transmit Holding Register

**Name:** TWIHS_THR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TXDATA[7:0]** Master or Slave Transmit Holding Data

### 43.7.15 TWIHS Write Protection Mode Register

**Name:** TWIHS_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x545749 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

See Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII). |

© 2021 Microchip Technology Inc. and its subsidiaries

### 43.7.16 TWIHS Write Protection Status Register

**Name:** TWIHS_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPVSRC[23:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the TWIHS_WPSR. |
| 1 | A write protection violation has occurred since the last read of the TWIHS_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 44. Synchronous Serial Controller (SSC)

## 44.1 Description

The Synchronous Serial Controller (SSC) provides a synchronous communication link with external devices. It supports many serial synchronous communication protocols generally used in audio and telecom applications such as I2S, Short Frame Sync, Long Frame Sync, etc.

The SSC contains an independent receiver and transmitter and a common clock divider. The receiver and the transmitter each interface with three signals: the TD/RD signal for data, the TK/RK signal for the clock and the TF/RF signal for the Frame Sync. The transfers can be programmed to start automatically or on different events detected on the Frame Sync signal.

The SSC high-level of programmability and its use of DMA enable a continuous high bit rate data transfer without processor intervention.

Featuring connection to the DMA, the SSC enables interfacing with low processor overhead to:

- Codecs in Master or Slave mode
- DAC through dedicated serial interface, particularly I2S
- Magnetic card reader

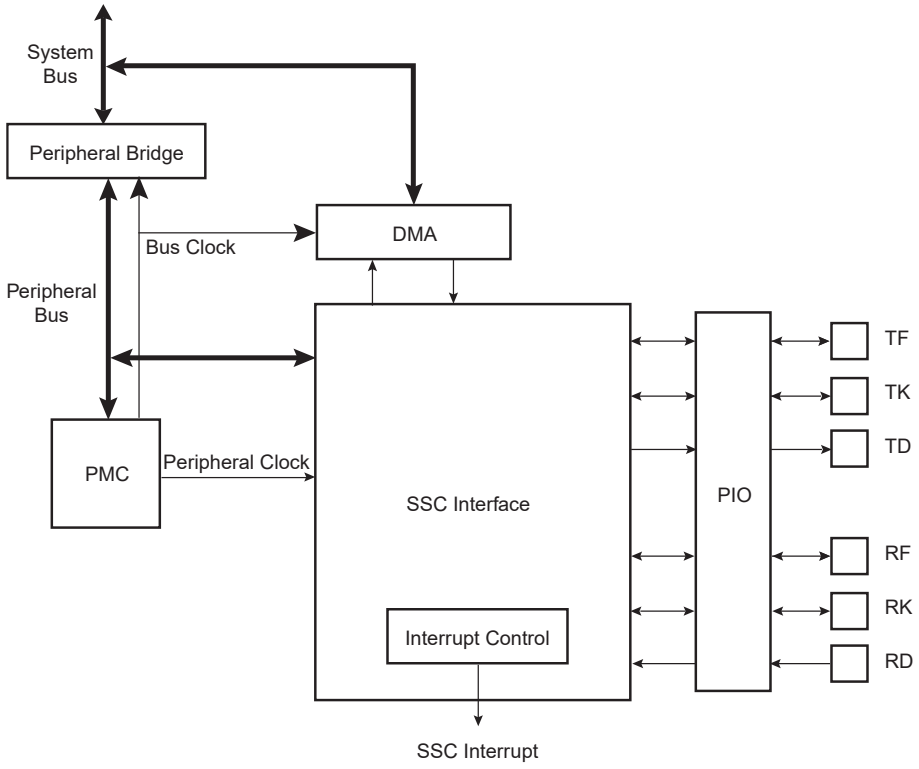
## 44.2 Embedded Characteristics

- Provides Serial Synchronous Communication Links Used in Audio and Telecom Applications
- Contains an Independent Receiver and Transmitter and a Common Clock Divider
- Interfaced with the DMA Controller (DMAC) to Reduce Processor Overhead
- Offers a Configurable Frame Sync and Data Length
- Receiver and Transmitter Can be Programmed to Start Automatically or on Detection of Different Events on the Frame Sync Signal
- Receiver and Transmitter Include a Data Signal, a Clock Signal and a Frame Sync Signal

© 2021 Microchip Technology Inc.
and its subsidiaries

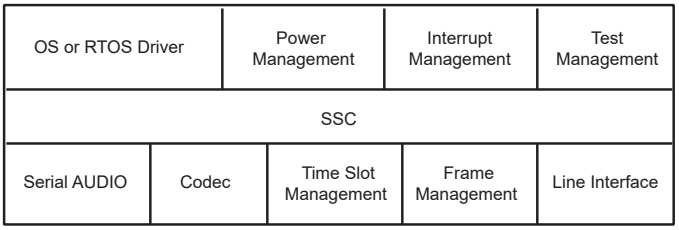
## 44.3 Block Diagram

**Figure 44-1. Block Diagram**




## 44.4 Application Block Diagram

**Figure 44-2. Application Block Diagram**




## 44.5 SSC Application Examples

The SSC can support several serial communication modes used in audio or high speed serial links. Some standard applications are shown in the following figures. All serial link applications supported by the SSC are not listed here.

**Figure 44-3. Audio Application Block Diagram**

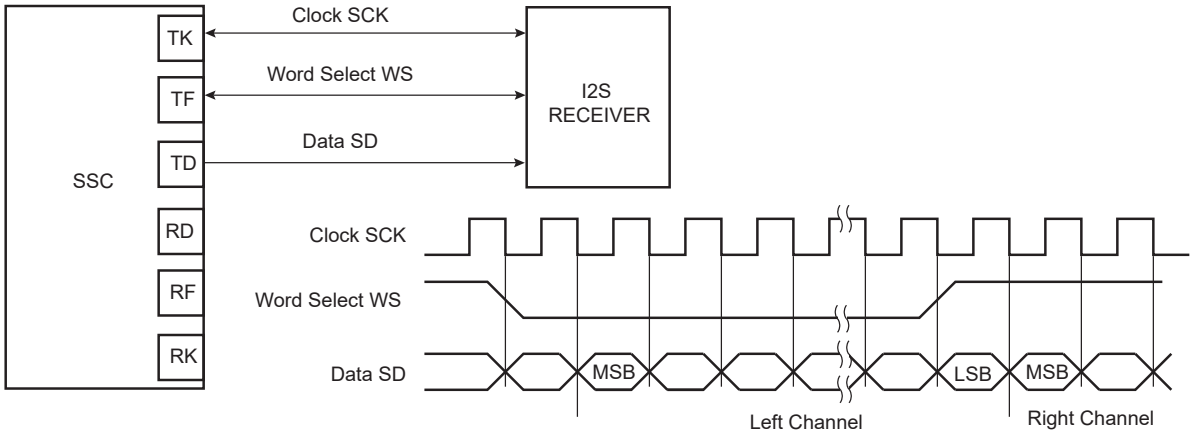



**Figure 44-4. Codec Application Block Diagram**

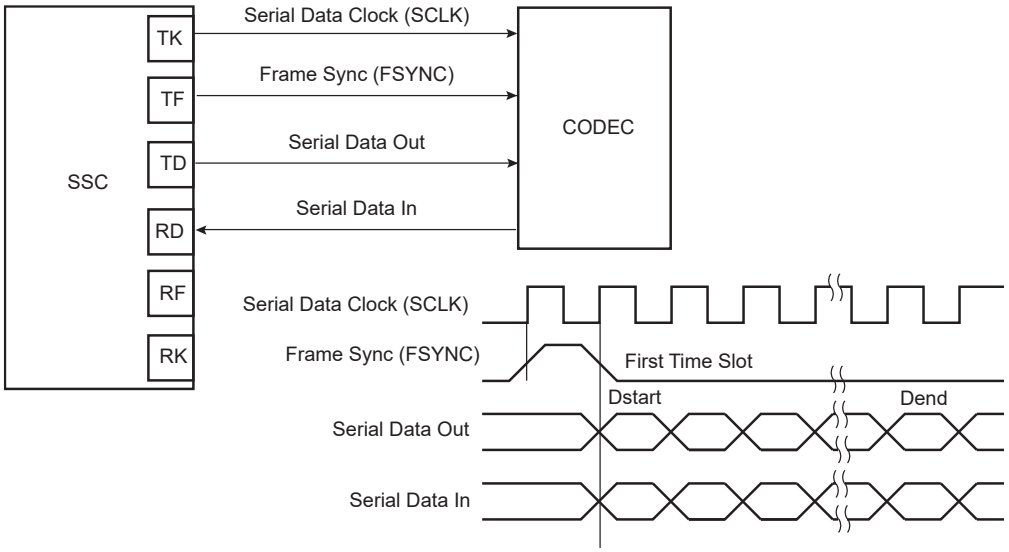
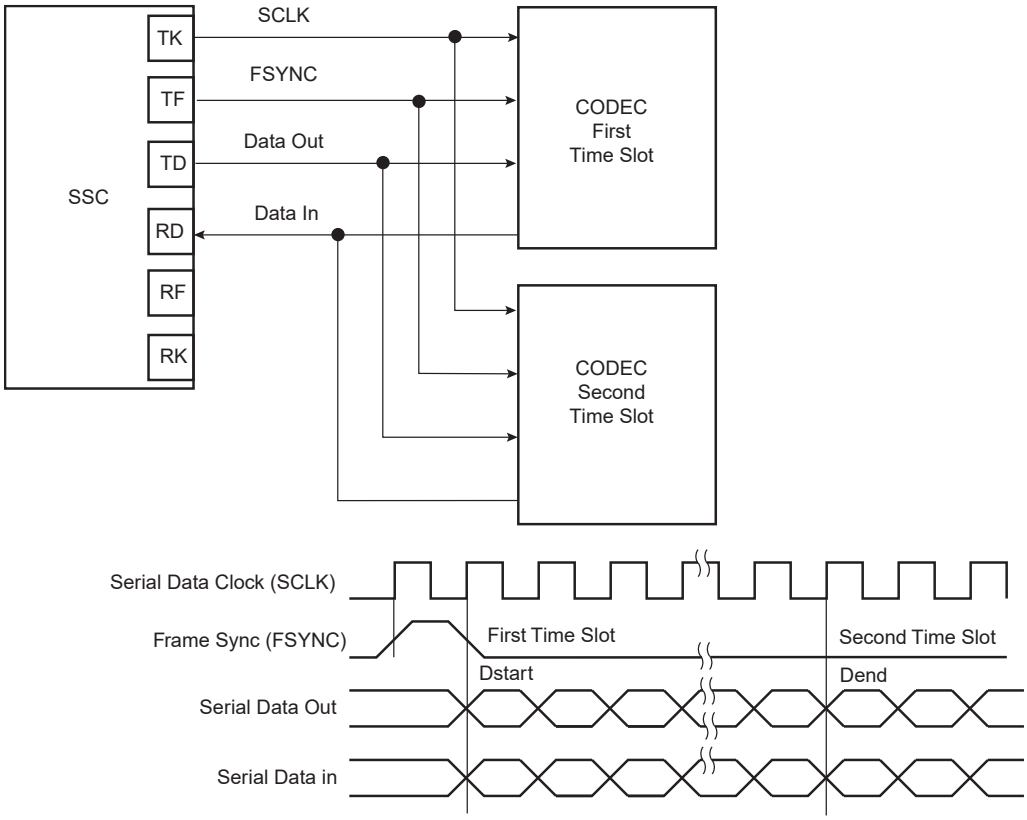


**Figure 44-5. Time Slot Application Block Diagram**




## 44.6 Pin Name List

**Table 44-1. I/O Lines Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| RF | Receive Frame Synchronization | Input/Output |
| RK | Receive Clock | Input/Output |
| RD | Receive Data | Input |
| TF | Transmit Frame Synchronization | Input/Output |
| TK | Transmit Clock | Input/Output |
| TD | Transmit Data | Output |

## 44.7 Product Dependencies

### 44.7.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines.

Before using the SSC receiver, the PIO controller must be configured to dedicate the SSC receiver I/O lines to the SSC Peripheral mode.

Before using the SSC transmitter, the PIO controller must be configured to dedicate the SSC transmitter I/O lines to the SSC Peripheral mode.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 44.7.2 Power Management

The SSC is not continuously clocked. The SSC interface may be clocked through the Power Management Controller (PMC), therefore the programmer must first configure the PMC to enable the SSC clock.

### 44.7.3 Interrupt

The SSC interface has an interrupt line connected to the interrupt controller. Handling interrupts requires programming the interrupt controller before configuring the SSC.

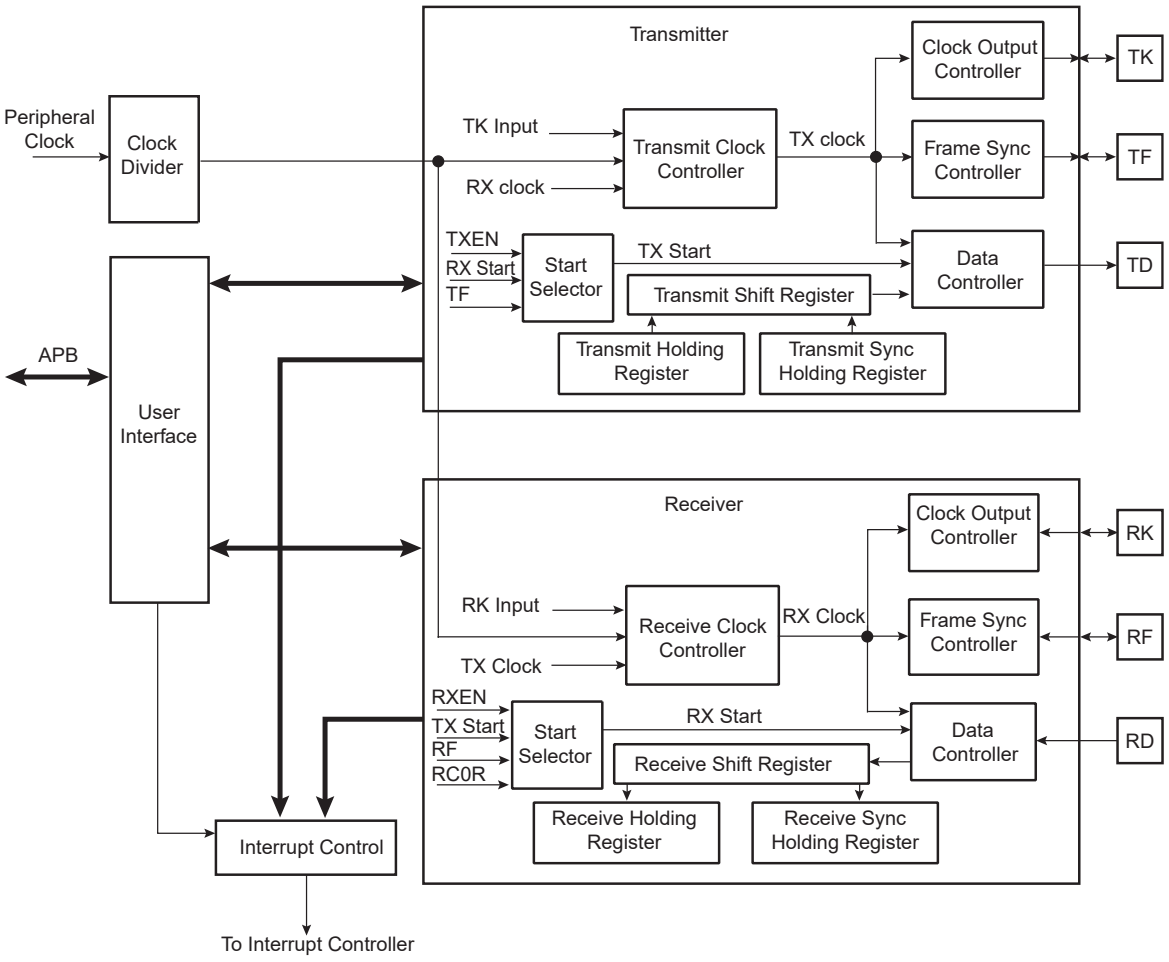
All SSC interrupts can be enabled/disabled configuring the SSC Interrupt Mask Register. Each pending and unmasked SSC interrupt asserts the SSC interrupt line. The SSC interrupt service routine can get the interrupt origin by reading the SSC Interrupt Status Register.

## 44.8 Functional Description

This section contains the functional description of the following: SSC Functional Block, Clock Management, Data Format, Start, Transmit, Receive and Frame Synchronization.

The receiver and transmitter operate separately. However, they can work synchronously by programming the receiver to use the transmit clock and/or to start a data transfer when transmission starts. Alternatively, this can be done by programming the transmitter to use the receive clock and/or to start a data transfer when reception starts. The transmitter and the receiver can be programmed to operate with the clock signals provided on either the TK or RK pins. This allows the SSC to support many Slave mode data transfers. The maximum clock speed allowed on the TK and RK pins is the peripheral clock divided by 2.

**Figure 44-6. SSC Functional Block Diagram**



### 44.8.1 Clock Management

The transmit clock can be generated by:

- an external clock received on the TK I/O pad
- the receive clock
- the internal clock divider

The receive clock can be generated by:

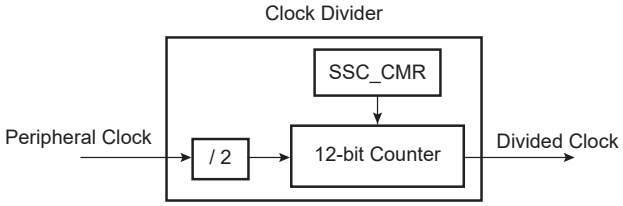
- an external clock received on the RK I/O pad
- the transmit clock
- the internal clock divider

Furthermore, the transmitter block can generate an external clock on the TK I/O pad, and the receive block can generate an external clock on the RK I/O pad.

This allows the SSC to support many Master and Slave mode data transfers.

#### 44.8.1.1 Clock Divider

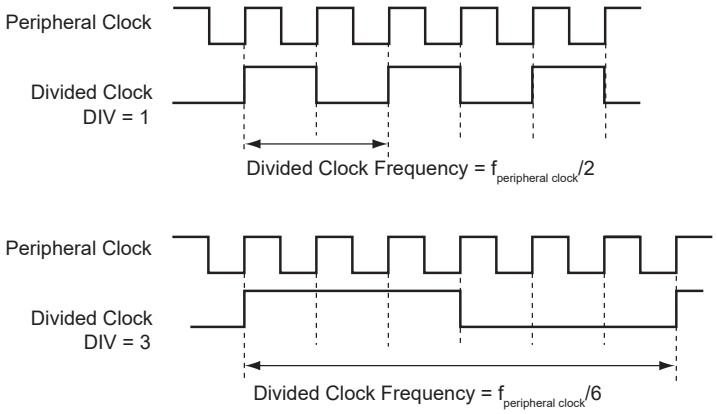
**Figure 44-7. Divided Clock Block Diagram**




The peripheral clock divider is determined by the 12-bit field DIV counter and comparator (so its maximal value is 4095) in the Clock Mode Register (SSC_CMR), allowing a peripheral clock division by up to 8190. The Divided Clock is provided to both the receiver and the transmitter. When this field is programmed to 0, the Clock Divider is not used and remains inactive.

When DIV is set to a value equal to or greater than 1, the Divided Clock has a frequency of peripheral clock divided by 2 times DIV. Each level of the Divided Clock has a duration of the peripheral clock multiplied by DIV. This ensures a 50% duty cycle for the Divided Clock regardless of whether the DIV value is even or odd.

**Figure 44-8. Divided Clock Generation**



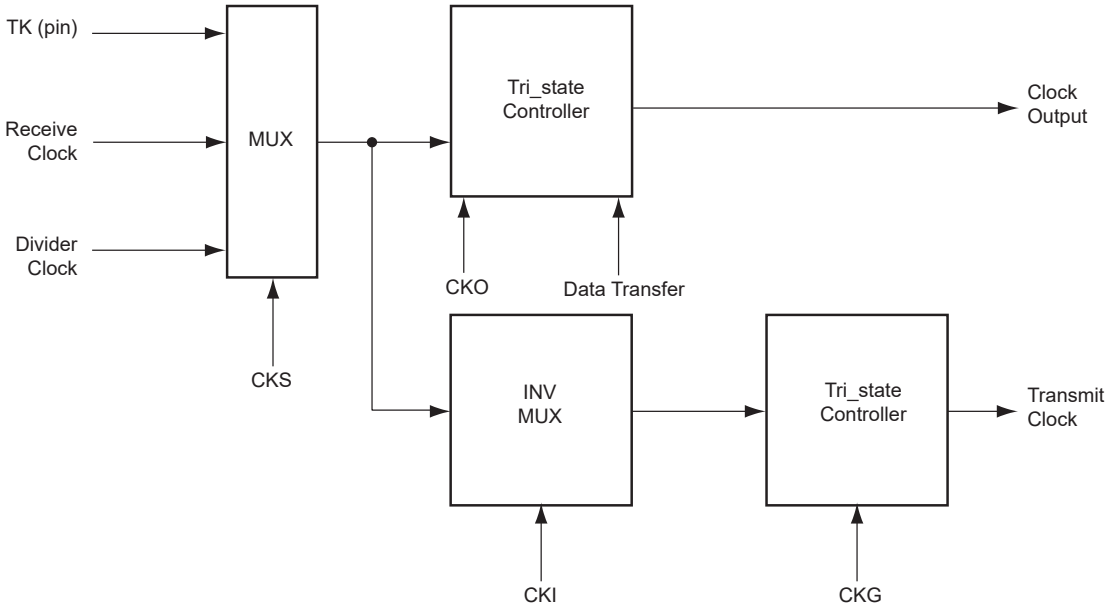

#### 44.8.1.2 Transmit Clock Management

The transmit clock is generated from the receive clock or the divider clock or an external clock scanned on the TK I/O pad. The transmit clock is selected by the CKS field in the Transmit Clock Mode Register (SSC_TCMR). Transmit Clock can be inverted independently by the CKI bits in the SSC_TCMR.

The transmitter can also drive the TK I/O pad continuously or be limited to the current data transfer. The clock output is configured by the SSC_TCMR. The Transmit Clock Inversion (CKI) bits have no effect on the clock outputs. Programming the SSC_TCMR to select TK pin (CKS field) and at the same time Continuous Transmit Clock (CKO field) can lead to unpredictable results.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 44-9. Transmit Clock Management**

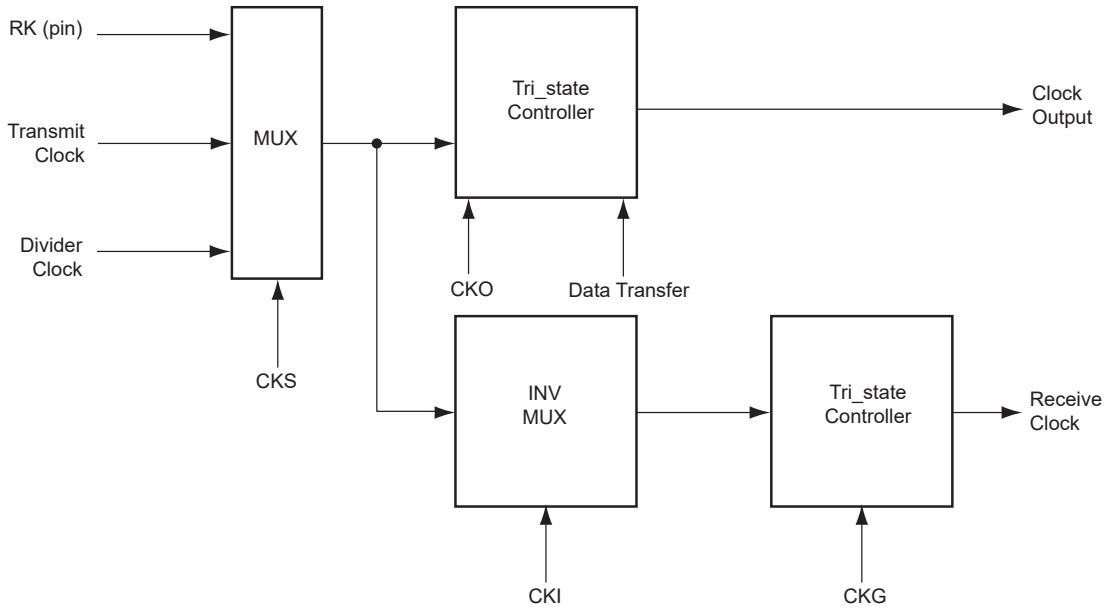



### 44.8.1.3 Receive Clock Management

The receive clock is generated from the transmit clock or the divider clock or an external clock scanned on the RK I/O pad. The Receive Clock is selected by the CKS field in SSC_RCMR (Receive Clock Mode Register). Receive Clocks can be inverted independently by the CKI bits in SSC_RCMR.

The receiver can also drive the RK I/O pad continuously or be limited to the current data transfer. The clock output is configured by the SSC_RCMR. The Receive Clock Inversion (CKI) bits have no effect on the clock outputs. Programming the SSC_RCMR to select RK pin (CKS field) and at the same time Continuous Receive Clock (CKO field) can lead to unpredictable results.

**Figure 44-10. Receive Clock Management**




### 44.8.1.4 Serial Clock Ratio Considerations

The transmitter and the receiver can be programmed to operate with the clock signals provided on either the TK or RK pins. This allows the SSC to support many Slave mode data transfers. In this case, the maximum clock speed allowed on the RK pin is:

- Peripheral clock divided by 2 if Receive Frame Synchronization is input
- Peripheral clock divided by 3 if Receive Frame Synchronization is output

In addition, the maximum clock speed allowed on the TK pin is:

- Peripheral clock divided by 6 if Transmit Frame Synchronization is input
- Peripheral clock divided by 2 if Transmit Frame Synchronization is output

These are only theoretical speed limits for first order calculations. Exact speed limits on TK and RK are provided in the "Electrical Characteristics" chapter.

### 44.8.2 Transmit Operations

A transmit frame is triggered by a start event and can be followed by synchronization data before data transmission.

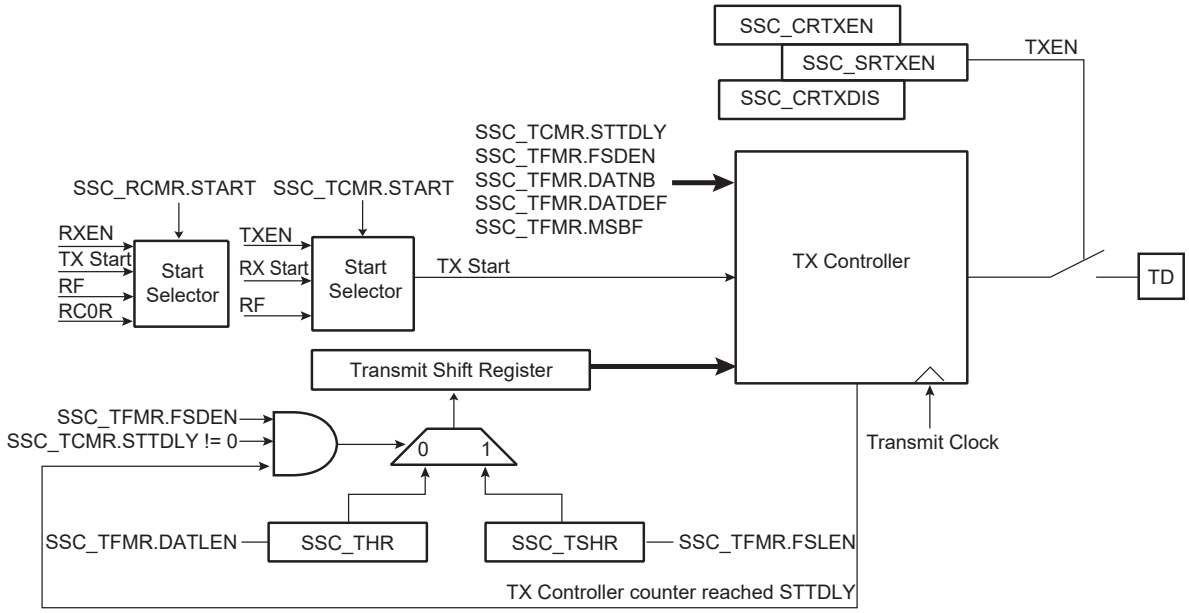
The start event is configured by setting the SSC_TCMR. See Start.

The frame synchronization is configured setting the Transmit Frame Mode Register (SSC_TFMR). See Frame Synchronization.

To transmit data, the transmitter uses a shift register clocked by the transmit clock signal and the start mode selected in the SSC_TCMR. Data is written by the application to the SSC_THR then transferred to the shift register according to the data format selected.

When both the SSC_THR and the transmit shift register are empty, the status flag TXEMPTY is set in the SSC_SR. When the Transmit Holding register is transferred in the transmit shift register, the status flag TXRDY is set in the SSC_SR and additional data can be loaded in the holding register.

**Figure 44-11. Transmit Block Diagram**




### 44.8.3 Receive Operations

A receive frame is triggered by a start event and can be followed by synchronization data before data transmission.

The start event is configured setting the Receive Clock Mode Register (SSC_RCMR). See Start.

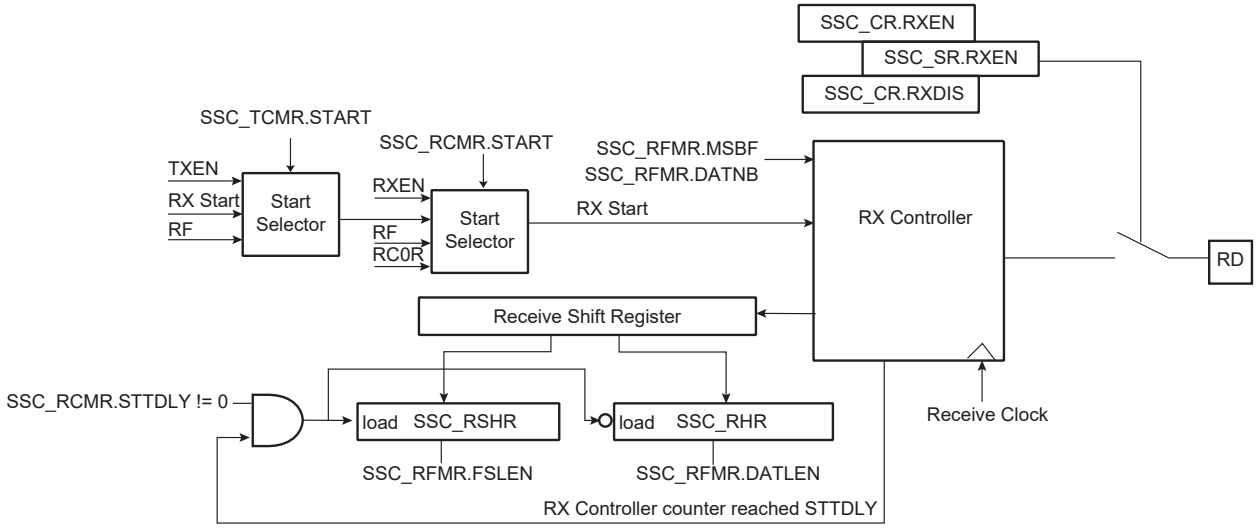
The frame synchronization is configured by setting the Receive Frame Mode Register (SSC_RFMR). See Frame Synchronization.

The receiver uses a shift register clocked by the receive clock signal and the start mode selected in the SSC_RCMR. The data is transferred from the shift register depending on the data format selected.

When the receiver shift register is full, the SSC transfers this data in the holding register, the status flag RXRDY is set in the SSC_SR and the data can be read in the receiver holding register. If another transfer occurs before read of the

Receive Holding Register (SSC_RHR), the status flag OVERUN is set in the SSC_SR and the receiver shift register is transferred in the SSC_RHR.

**Figure 44-12. Receive Block Diagram**




### 44.8.4 Start

The transmitter and receiver can both be programmed to start their operations when an event occurs, respectively in the Transmit Start Selection (START) field of SSC_TCMR and in the Receive Start Selection (START) field of SSC_RCMR.

Under the following conditions the start event is independently programmable:

- Continuous. In this case, the transmission starts as soon as a word is written in SSC_THR and the reception starts as soon as the receiver is enabled.
- Synchronously with the transmitter/receiver
- On detection of a falling/rising edge on TF/RF
- On detection of a low level/high level on TF/RF
- On detection of a level change or an edge on TF/RF

A start can be programmed in the same manner on either side of the Transmit/Receive Clock Register (SSC_RCMR/SSC_TCMR). Thus, the start could be on TF (Transmit) or RF (Receive).

Moreover, the receiver can start when data is detected in the bit stream with the Compare Functions.

Detection on TF/RF input/output is done by the field FSOS of the Transmit/Receive Frame Mode Register (SSC_TFMR/SSC_RFMR).

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 44-13. Transmit Start Mode**

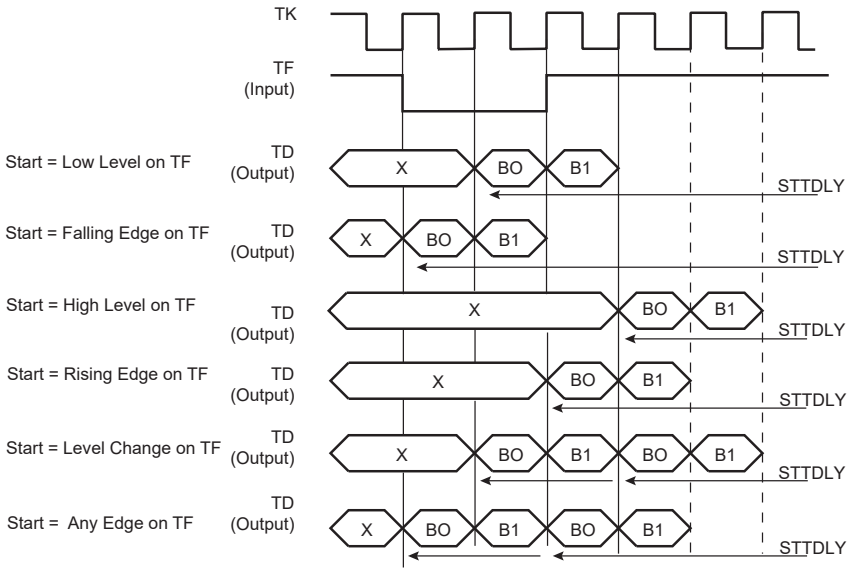
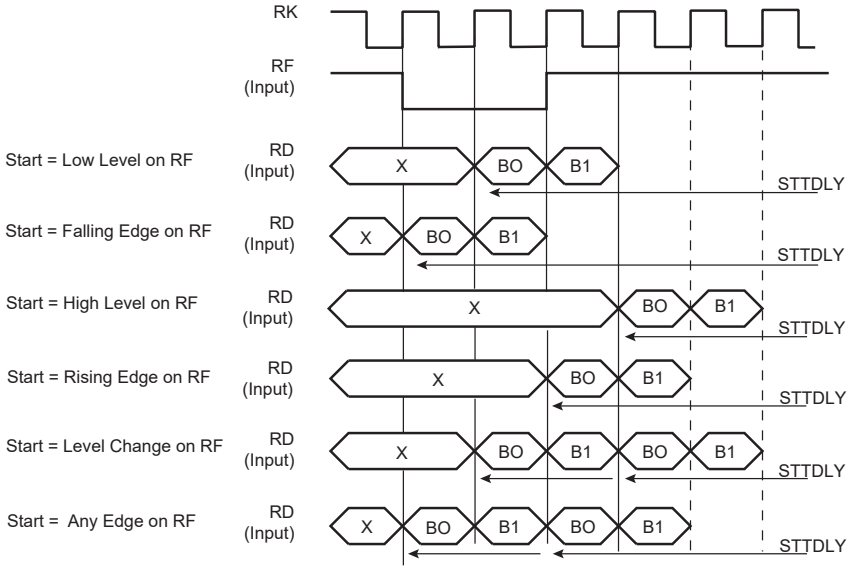



**Figure 44-14. Receive Pulse/Edge Start Modes**




### 44.8.5 Frame Synchronization

The Transmit and Receive Frame Sync pins, TF and RF, can be programmed to generate different kinds of Frame Sync signals. The Frame Sync Output Selection (FSOS) field in the Receive Frame Mode Register (SSC_RFMR) and in the Transmit Frame Mode Register (SSC_TFMR) are used to select the required waveform.

- Programmable low or high levels during data transfer are supported.
- Programmable high levels before the start of data transfers or toggling are also supported.

If a pulse waveform is selected, the Frame Sync Length (FSLEN) field in SSC_RFMR and SSC_TFMR programs the length of the pulse, from 1 bit time up to 256 bit times.

The periodicity of the Receive and Transmit Frame Sync pulse output can be programmed through the Period Divider Selection (PERIOD) field in SSC_RCMR and SSC_TCMR.

#### 44.8.5.1 Frame Sync Data

Frame Sync Data transmits or receives a specific tag during the Frame Sync signal.

During the Frame Sync signal, the receiver can sample the RD line and store the data in the Receive Sync Holding Register and the transmitter can transfer Transmit Sync Holding Register in the shift register. The data length to be sampled/shifted out during the Frame Sync signal is programmed by the FSLEN field in SSC_RFMR/SSC_TFMR and has a maximum value of 256.

Concerning the Receive Frame Sync Data operation, if the Frame Sync Length is equal to or lower than the delay between the start event and the current data reception, the data sampling operation is performed in the Receive Sync Holding Register through the receive shift register.

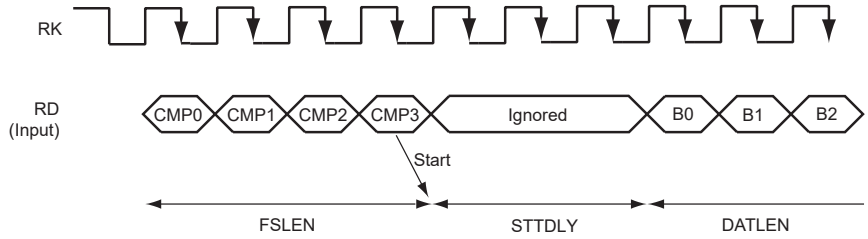
The Transmit Frame Sync Operation is performed by the transmitter only if the bit Frame Sync Data Enable (FSDEN) in SSC_TFMR is set. If the Frame Sync length is equal to or lower than the delay between the start event and the current data transmission, the normal transmission has priority and the data contained in the Transmit Sync Holding Register is transferred in the Transmit Register, then shifted out.

#### 44.8.5.2 Frame Sync Edge Detection

The Frame Sync Edge detection is programmed by the FSEDGE field in SSC_RFMR/SSC_TFMR. This sets the corresponding flags RXSYN/TXSYN in the SSC Status Register (SSC_SR) on Frame Sync Edge detection (signals RF/TF).

### 44.8.6 Receive Compare Modes

**Figure 44-15. Receive Compare Modes**




#### 44.8.6.1 Compare Functions

The length of the comparison patterns (Compare 0, Compare 1) and thus the number of bits they are compared to is defined by FSLEN, but with a maximum value of 256 bits. Comparison is always done by comparing the last bits received with the comparison pattern. Compare 0 can be one start event of the receiver. In this case, the receiver compares at each new sample the last bits received at the Compare 0 pattern contained in the Compare 0 Register (SSC_RC0R). When this start event is selected, the user can program the receiver to start a new data transfer either by writing a new Compare 0, or by receiving continuously until Compare 1 occurs. This selection is done with the STOP bit in the SSC_RCMR.

### 44.8.7 Data Format

The data framing format of both the transmitter and the receiver are programmable through the Transmitter Frame Mode Register (SSC_TFMR) and the Receive Frame Mode Register (SSC_RFMR). In either case, the user can independently select the following parameters:

- Event that starts the data transfer (START)
- Delay in number of bit periods between the start event and the first data bit (STTDLY)
- Length of the data (DATLEN)
- Number of data to be transferred for each start event (DATNB)
- Length of synchronization transferred for each start event (FSLEN)
- Bit sense: most or least significant bit first (MSBF)

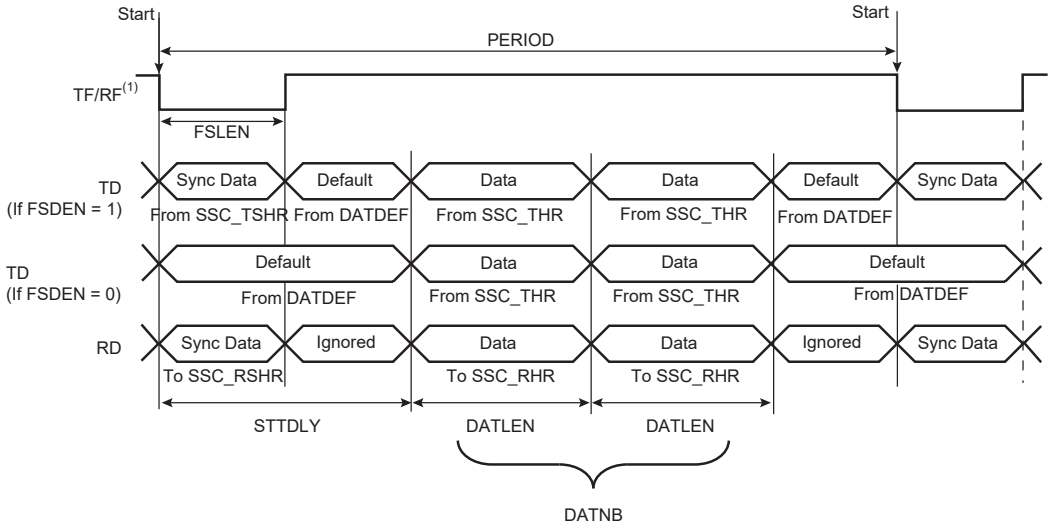
Additionally, the transmitter can be used to transfer synchronization and select the level driven on the TD pin while not in data transfer operation. This is done respectively by the Frame Sync Data Enable (FSDEN) and by the Data Default Value (DATDEF) bits in SSC_TFMR.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Table 44-2. Data Frame Registers**

| Transmitter | Receiver | Field | Length | Comment |
|---|---|---|---|---|
| SSC_TFMR | SSC_RFMR | DATLEN | Up to 32 | Size of word |
| SSC_TFMR | SSC_RFMR | DATNB | Up to 16 | Number of words transmitted in frame |
| SSC_TFMR | SSC_RFMR | MSBF | – | Most significant bit first |
| SSC_TFMR | SSC_RFMR | FSLEN | Up to 256 | Size of Synchro data register |
| SSC_TFMR | – | DATDEF | 0 or 1 | Data default value ended |
| SSC_TFMR | – | FSDEN | – | Enable send SSC_TSHR |
| SSC_TCMR | SSC_RCMR | PERIOD | Up to 512 | Frame size |
| SSC_TCMR | SSC_RCMR | STTDLY | Up to 255 | Size of transmit start delay |

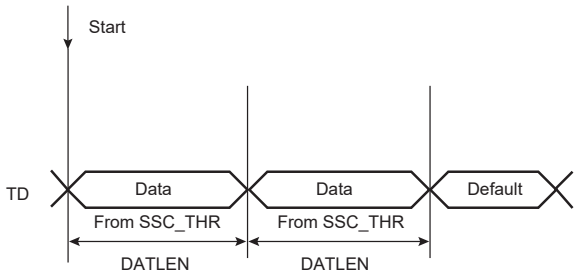
**Figure 44-16. Transmit and Receive Frame Format in Edge/Pulse Start Modes**




Note: 1. Example of input on falling edge of TF/RF.

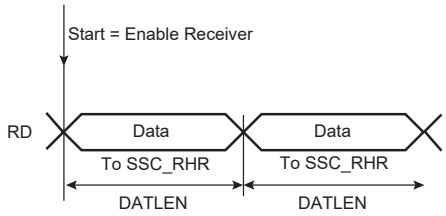
In the example illustrated in Transmit Frame Format in Continuous Mode (STTDLY = 0), the SSC_THR is loaded twice. The FSDEN value has no effect on the transmission. SyncData cannot be output in Continuous mode.

**Figure 44-17. Transmit Frame Format in Continuous Mode (STTDLY = 0)**




Start: 1. TXEMPTY set to 1
2. Write into the SSC_THR

**Figure 44-18. Receive Frame Format in Continuous Mode (STTDLY = 0)**




### 44.8.8 Loop Mode

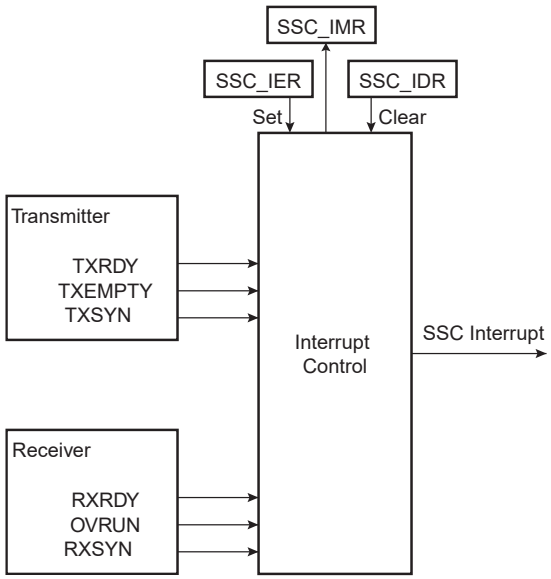
The receiver can be programmed to receive transmissions from the transmitter. This is done by setting the Loop Mode (LOOP) bit in the SSC_RFMR. In this case, RD is connected to TD, RF is connected to TF and RK is connected to TK.

### 44.8.9 Interrupt

Most bits in the SSC_SR have a corresponding bit in interrupt management registers.

The SSC can be programmed to generate an interrupt when it detects an event. The interrupt is controlled by writing the Interrupt Enable Register (SSC_IER) and Interrupt Disable Register (SSC_IDR). These registers enable and disable, respectively, the corresponding interrupt by setting and clearing the corresponding bit in the Interrupt Mask Register (SSC_IMR), which controls the generation of interrupts by asserting the SSC interrupt line connected to the interrupt controller.

**Figure 44-19. Interrupt Block Diagram**




### 44.8.10 Register Write Protection

To prevent any single software error from corrupting SSC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the SSC Write Protection Mode Register (SSC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the SSC Write Protection Status Register (SSC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the SSC_WPSR.

The following registers can be write-protected:

- SSC Clock Mode Register
- SSC Receive Clock Mode Register
- SSC Receive Frame Mode Register

- SSC Transmit Clock Mode Register
- SSC Transmit Frame Mode Register
- SSC Receive Compare 0 Register
- SSC Receive Compare 1 Register

## 44.9 Register Summary

**Note:** Offsets 0x100–0x128 are reserved for PDC registers.

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | SSC_CR | 7:0 | | | | | | | RXDIS | RXEN |
| | | 15:8 | SWRST | | | | | | TXDIS | TXEN |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | SSC_CMR | 7:0 | DIV[7:0] | | | | | | | |
| | | 15:8 | | | | | DIV[11:8] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | SSC_RCMR | 7:0 | CKG[1:0] | | CKI | CKO[2:0] | | | CKS[1:0] | |
| | | 15:8 | | | | STOP | START[3:0] | | | |
| | | 23:16 | STTDLY[7:0] | | | | | | | |
| | | 31:24 | PERIOD[7:0] | | | | | | | |
| 0x14 | SSC_RFMR | 7:0 | MSBF | | LOOP | DATLEN[4:0] | | | | |
| | | 15:8 | | | | | DATNB[3:0] | | | |
| | | 23:16 | | FSOS[2:0] | | | FSLEN[3:0] | | | |
| | | 31:24 | FSLEN_EXT[3:0] | | | | | | | FSEDGE |
| 0x18 | SSC_TCMR | 7:0 | CKG[1:0] | | CKI | CKO[2:0] | | | CKS[1:0] | |
| | | 15:8 | | | | | START[3:0] | | | |
| | | 23:16 | STTDLY[7:0] | | | | | | | |
| | | 31:24 | PERIOD[7:0] | | | | | | | |
| 0x1C | SSC_TFMR | 7:0 | MSBF | | DATDEF | DATLEN[4:0] | | | | |
| | | 15:8 | | | | | DATNB[3:0] | | | |
| | | 23:16 | FSDEN | FSOS[2:0] | | | FSLEN[3:0] | | | |
| | | 31:24 | FSLEN_EXT[3:0] | | | | | | | FSEDGE |
| 0x20 | SSC_RHR | 7:0 | RDAT[7:0] | | | | | | | |
| | | 15:8 | RDAT[15:8] | | | | | | | |
| | | 23:16 | RDAT[23:16] | | | | | | | |
| | | 31:24 | RDAT[31:24] | | | | | | | |
| 0x24 | SSC_THR | 7:0 | TDAT[7:0] | | | | | | | |
| | | 15:8 | TDAT[15:8] | | | | | | | |
| | | 23:16 | TDAT[23:16] | | | | | | | |
| | | 31:24 | TDAT[31:24] | | | | | | | |
| 0x28 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | SSC_RSHR | 7:0 | RSDAT[7:0] | | | | | | | |
| | | 15:8 | RSDAT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | SSC_TSHR | 7:0 | TSDAT[7:0] | | | | | | | |
| | | 15:8 | TSDAT[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x38 | SSC_RC0R | 7:0 | CP0[7:0] | | | | | | | |
| | | 15:8 | CP0[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x3C | SSC_RC1R | 7:0 | CP1[7:0] | | | | | | | |
| | | 15:8 | CP1[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x40 | SSC_SR | 7:0 | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| | | 15:8 | | | | | RXSYN | TXSYN | CP1 | CP0 |
| | | 23:16 | | | | | | | RXEN | TXEN |
| | | 31:24 | | | | | | | | |
| 0x44 | SSC_IER | 7:0 | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| | | 15:8 | | | | | RXSYN | TXSYN | CP1 | CP0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | SSC_IDR | 7:0 | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| | | 15:8 | | | | | RXSYN | TXSYN | CP1 | CP0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | SSC_IMR | 7:0 | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| | | 15:8 | | | | | RXSYN | TXSYN | CP1 | CP0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x50 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | SSC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | SSC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 44.9.1 SSC Control Register

**Name:** SSC_CR
**Offset:** 0x0
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | | | | | TXDIS | TXEN |
| Access | W | | | | | | W | W |
| Reset | – | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXDIS | RXEN |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 15 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Performs a software reset. Has priority on any other bit in SSC_CR. |

**Bit 9 – TXDIS** Transmit Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables Transmit. If a character is currently being transmitted, disables at end of current character transmission. |

**Bit 8 – TXEN** Transmit Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables Transmit if TXDIS is not set. |

**Bit 1 – RXDIS** Receive Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables Receive. If a character is currently being received, disables at end of current character reception. |

**Bit 0 – RXEN** Receive Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables Receive if RXDIS is not set. |

### 44.9.2 SSC Clock Mode Register

**Name:** SSC_CMR
**Offset:** 0x4
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DIV[11:8] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 11:0 – DIV[11:0]** Clock Divider

| Value | Description |
|---|---|
| 0 | The Clock Divider is not active. |
| Any other value | The divided clock equals the peripheral clock divided by 2 times DIV.<br>The maximum bit rate is $f_{peripheral\ clock}/2$. The minimum bit rate is $f_{peripheral\ clock}/2 \times 4095 = f_{peripheral\ clock}/8190$. |

### 44.9.3 SSC Receive Clock Mode Register

**Name:** SSC_RCMR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PERIOD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | STTDLY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | STOP | START[3:0] | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CKG[1:0] | | CKI | CKO[2:0] | | | CKS[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – PERIOD[7:0]** Receive Period Divider Selection
This field selects the divider to apply to the selected Receive Clock in order to generate a new Frame Sync signal. If 0, no PERIOD signal is generated. If not 0, a PERIOD signal is generated each 2 x (PERIOD + 1) Receive Clock.

**Bits 23:16 – STTDLY[7:0]** Receive Start Delay
If STTDLY is not 0, a delay of STTDLY clock cycles is inserted between the start event and the current start of reception. When the receiver is programmed to start synchronously with the transmitter, the delay is also applied.
**Note:**
STTDLY must be configured in relation to the receive synchronization data to be stored in SSC_RSHR.

**Bit 12 – STOP** Receive Stop Selection

| Value | Description |
|---|---|
| 0 | After completion of a data transfer when starting with a Compare 0, the receiver stops the data transfer and waits for a new compare 0. |
| 1 | After starting a receive with a Compare 0, the receiver operates in a continuous mode until a Compare 1 is detected. |

**Bits 11:8 – START[3:0]** Receive Start Selection

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | Continuous, as soon as the receiver is enabled, and immediately after the end of transfer of the previous data. |
| 1 | TRANSMIT | Transmit start |
| 2 | RF_LOW | Detection of a low level on RF signal |
| 3 | RF_HIGH | Detection of a high level on RF signal |
| 4 | RF_FALLING | Detection of a falling edge on RF signal |
| 5 | RF_RISING | Detection of a rising edge on RF signal |
| 6 | RF_LEVEL | Detection of any level change on RF signal |
| 7 | RF_EDGE | Detection of any edge on RF signal |

© 2021 Microchip Technology Inc. and its subsidiaries

| Value | Name | Description |
|---|---|---|
| 8 | CMP_0 | Compare 0 |

**Bits 7:6 – CKG[1:0]** Receive Clock Gating Selection

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | None |
| 1 | EN_RF_LOW | Receive Clock enabled only if RF Low |
| 2 | EN_RF_HIGH | Receive Clock enabled only if RF High |

**Bit 5 – CKI** Receive Clock Inversion
CKI affects only the Receive Clock and not the output clock signal.

| Value | Description |
|---|---|
| 0 | The data inputs (Data and Frame Sync signals) are sampled on Receive Clock falling edge. The Frame Sync signal output is shifted out on Receive Clock rising edge. |
| 1 | The data inputs (Data and Frame Sync signals) are sampled on Receive Clock rising edge. The Frame Sync signal output is shifted out on Receive Clock falling edge. |

**Bits 4:2 – CKO[2:0]** Receive Clock Output Mode Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None, RK pin is an input |
| 1 | CONTINUOUS | Continuous Receive Clock, RK pin is an output |
| 2 | TRANSFER | Receive Clock only during data transfers, RK pin is an output |

**Bits 1:0 – CKS[1:0]** Receive Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | MCK | Divided Clock |
| 1 | TK | TK Clock signal |
| 2 | RK | RK pin |

### 44.9.4 SSC Receive Frame Mode Register

**Name:** SSC_RFMR
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FSLEN_EXT[3:0] | | | | | | | FSEDGE |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | FSOS[2:0] | | | FSLEN[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DATNB[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSBF | | LOOP | DATLEN[4:0] | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:28 – FSLEN_EXT[3:0]** FSLEN Field Extension
Extends FSLEN field. For details, see FSLEN: Receive Frame Sync Length.

**Bit 24 – FSEDGE** Frame Sync Edge Detection
Determines which edge on Frame Sync will generate the interrupt RXSYN in the SSC Status Register.

| Value | Name | Description |
|---|---|---|
| 0 | POSITIVE | Positive Edge Detection |
| 1 | NEGATIVE | Negative Edge Detection |

**Bits 22:20 – FSOS[2:0]** Receive Frame Sync Output Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None, RF pin is an input |
| 1 | NEGATIVE | Negative Pulse, RF pin is an output |
| 2 | POSITIVE | Positive Pulse, RF pin is an output |
| 3 | LOW | Driven Low during data transfer, RF pin is an output |
| 4 | HIGH | Driven High during data transfer, RF pin is an output |
| 5 | TOGGLING | Toggling at each start of data transfer, RF pin is an output |

**Bits 19:16 – FSLEN[3:0]** Receive Frame Sync Length
This field defines the number of bits sampled and stored in the Receive Sync Data Register. When this mode is selected by the START field in the Receive Clock Mode Register, it also determines the length of the sampled data to be compared to the Compare 0 or Compare 1 register.
This field is used with FSLEN_EXT to determine the pulse length of the Receive Frame Sync signal.
Pulse length is equal to FSLEN + (FSLEN_EXT × 16) + 1 Receive Clock periods.

**Bits 11:8 – DATNB[3:0]** Data Number per Frame
This field defines the number of data words to be received after each transfer start, which is equal to (DATNB + 1).

**Bit 7 – MSBF** Most Significant Bit First

| Value | Description |
|---|---|
| 0 | The lowest significant bit of the data register is sampled first in the bit stream. |
| 1 | The most significant bit of the data register is sampled first in the bit stream. |

**Bit 5 – LOOP** Loop Mode

| Value | Description |
|---|---|
| 0 | Normal operating mode. |
| 1 | RD is driven by TD, RF is driven by TF and TK drives RK. |

**Bits 4:0 – DATLEN[4:0]** Data Length

| Value | Description |
|---|---|
| 0 | Forbidden value (1-bit data length not supported). |
| Any other value | The bit stream contains DATLEN + 1 data bits. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 44.9.5 SSC Transmit Clock Mode Register

**Name:** SSC_TCMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | PERIOD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | STTDLY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | START[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CKG[1:0] | | CKI | CKO[2:0] | | | CKS[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – PERIOD[7:0]** Transmit Period Divider Selection
This field selects the divider to apply to the selected Transmit Clock to generate a new Frame Sync signal. If 0, no period signal is generated. If not 0, a period signal is generated at each 2 × (PERIOD + 1) Transmit Clock.

**Bits 23:16 – STTDLY[7:0]** Transmit Start Delay
If STTDLY is not 0, a delay of STTDLY clock cycles is inserted between the start event and the current start of transmission of data. When the transmitter is programmed to start synchronously with the receiver, the delay is also applied.
**Note:**
If STTDLY is too short with respect to transmit synchronization data (SSC_TSHR), SSC_THR.TDAT is transmitted instead of the end of SSC_TSHR.

**Bits 11:8 – START[3:0]** Transmit Start Selection

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | Continuous, as soon as a word is written in the SSC_THR (if Transmit is enabled), and immediately after the end of transfer of the previous data |
| 1 | RECEIVE | Receive start |
| 2 | TF_LOW | Detection of a low level on TF signal |
| 3 | TF_HIGH | Detection of a high level on TF signal |
| 4 | TF_FALLING | Detection of a falling edge on TF signal |
| 5 | TF_RISING | Detection of a rising edge on TF signal |
| 6 | TF_LEVEL | Detection of any level change on TF signal |
| 7 | TF_EDGE | Detection of any edge on TF signal |

**Bits 7:6 – CKG[1:0]** Transmit Clock Gating Selection

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | None |
| 1 | EN_TF_LOW | Transmit Clock enabled only if TF Low |

| Value | Name | Description |
|---|---|---|
| 2 | EN_TF_HIGH | Transmit Clock enabled only if TF High |

**Bit 5 – CKI** Transmit Clock Inversion
CKI affects only the Transmit Clock and not the Output Clock signal.

| Value | Description |
|---|---|
| 0 | The data outputs (Data and Frame Sync signals) are shifted out on Transmit Clock falling edge. The Frame Sync signal input is sampled on Transmit Clock rising edge. |
| 1 | The data outputs (Data and Frame Sync signals) are shifted out on Transmit Clock rising edge. The Frame Sync signal input is sampled on Transmit Clock falling edge. |

**Bits 4:2 – CKO[2:0]** Transmit Clock Output Mode Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None, TK pin is an input |
| 1 | CONTINUOUS | Continuous Transmit Clock, TK pin is an output |
| 2 | TRANSFER | Transmit Clock only during data transfers, TK pin is an output |

**Bits 1:0 – CKS[1:0]** Transmit Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | MCK | Divided Clock |
| 1 | RK | RK Clock signal |
| 2 | TK | TK pin |

### 44.9.6 SSC Transmit Frame Mode Register

**Name:** SSC_TFMR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FSLEN_EXT[3:0] | | | | | | | FSEDGE |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FSDEN | FSOS[2:0] | | | FSLEN[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DATNB[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSBF | | DATDEF | DATLEN[4:0] | | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:28 – FSLEN_EXT[3:0]** FSLEN Field Extension
Extends FSLEN field. For details, seee FSLEN bit description below.

**Bit 24 – FSEDGE** Frame Sync Edge Detection
Determines which edge on frame synchronization will generate the interrupt TXSYN (Status Register).

| Value | Name | Description |
|---|---|---|
| 0 | POSITIVE | Positive Edge Detection |
| 1 | NEGATIVE | Negative Edge Detection |

**Bit 23 – FSDEN** Frame Sync Data Enable

| Value | Description |
|---|---|
| 0 | The TD line is driven with the default value during the Transmit Frame Sync signal. |
| 1 | SSC_TSHR value is shifted out during the transmission of the Transmit Frame Sync signal. |

**Bits 22:20 – FSOS[2:0]** Transmit Frame Sync Output Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None, TF pin is an input |
| 1 | NEGATIVE | Negative Pulse, TF pin is an output |
| 2 | POSITIVE | Positive Pulse, TF pin is an output |
| 3 | LOW | Driven Low during data transfer |
| 4 | HIGH | Driven High during data transfer |
| 5 | TOGGLING | Toggling at each start of data transfer |

**Bits 19:16 – FSLEN[3:0]** Transmit Frame Sync Length
This field defines the length of the Transmit Frame Sync signal and the number of bits shifted out from SSC_TSHR if FSDEN is 1.
This field is used with FSLEN_EXT to determine the pulse length of the Transmit Frame Sync signal.
Pulse length is equal to FSLEN + (FSLEN_EXT × 16) + 1 Transmit Clock period.

**Bits 11:8 – DATNB[3:0]** Data Number per Frame
This field defines the number of data words to be transferred after each transfer start, which is equal to (DATNB + 1).

**Bit 7 – MSBF** Most Significant Bit First

| Value | Description |
|---|---|
| 0 | The lowest significant bit of the data register is shifted out first in the bit stream. |
| 1 | The most significant bit of the data register is shifted out first in the bit stream. |

**Bit 5 – DATDEF** Data Default Value
This bit defines the level driven on the TD pin while out of transmission. Note that if the pin is defined as multi-drive by the PIO Controller, the pin is enabled only if the SCC TD output is 1.

**Bits 4:0 – DATLEN[4:0]** Data Length

| Value | Description |
|---|---|
| 0 | Forbidden value (1-bit data length not supported). |
| Any other value | The bit stream contains DATLEN + 1 data bits. |

### 44.9.7 SSC Receive Holding Register

**Name:** SSC_RHR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RDAT[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RDAT[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RDAT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDAT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RDAT[31:0]** Receive Data
Right aligned regardless of the number of data bits defined by DATLEN in SSC_RFMR.

### 44.9.8 SSC Transmit Holding Register

**Name:** SSC_THR
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TDAT[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TDAT[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TDAT[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TDAT[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – TDAT[31:0]** Transmit Data
Right aligned regardless of the number of data bits defined by DATLEN in SSC_TFMR.

### 44.9.9 SSC Receive Synchronization Holding Register

**Name:** SSC_RSHR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RSDAT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RSDAT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – RSDAT[15:0]** Receive Synchronization Data

### 44.9.10 SSC Transmit Synchronization Holding Register

**Name:** SSC_TSHR
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TSDAT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TSDAT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TSDAT[15:0]** Transmit Synchronization Data

### 44.9.11 SSC Receive Compare 0 Register

**Name:** SSC_RC0R
**Offset:** 0x38
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CP0[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CP0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – CP0[15:0]** Receive Compare Data 0

### 44.9.12 SSC Receive Compare 1 Register

**Name:** SSC_RC1R
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CP1[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CP1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – CP1[15:0]** Receive Compare Data 1

### 44.9.13 SSC Status Register

**Name:** SSC_SR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXEN | TXEN |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXSYN | TXSYN | CP1 | CP0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 17 – RXEN** Receive Enable

| Value | Description |
|---|---|
| 0 | Receive is disabled. |
| 1 | Receive is enabled. |

**Bit 16 – TXEN** Transmit Enable

| Value | Description |
|---|---|
| 0 | Transmit is disabled. |
| 1 | Transmit is enabled. |

**Bit 11 – RXSYN** Receive Sync

| Value | Description |
|---|---|
| 0 | An Rx Sync has not occurred since the last read of the Status Register. |
| 1 | An Rx Sync has occurred since the last read of the Status Register. |

**Bit 10 – TXSYN** Transmit Sync

| Value | Description |
|---|---|
| 0 | A Tx Sync has not occurred since the last read of the Status Register. |
| 1 | A Tx Sync has occurred since the last read of the Status Register. |

**Bit 9 – CP1** Compare 1

| Value | Description |
|---|---|
| 0 | A compare 1 has not occurred since the last read of the Status Register. |
| 1 | A compare 1 has occurred since the last read of the Status Register. |

**Bit 8 – CP0** Compare 0

| Value | Description |
|---|---|
| 0 | A compare 0 has not occurred since the last read of the Status Register. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | A compare 0 has occurred since the last read of the Status Register. |

**Bit 5 – OVRUN** Receive Overrun

| Value | Description |
|---|---|
| 0 | No data has been loaded in SSC_RHR while previous data has not been read since the last read of the Status Register. |
| 1 | Data has been loaded in SSC_RHR while previous data has not yet been read since the last read of the Status Register. |

**Bit 4 – RXRDY** Receive Ready

| Value | Description |
|---|---|
| 0 | SSC_RHR is empty. |
| 1 | Data has been received and loaded in SSC_RHR. |

**Bit 1 – TXEMPTY** Transmit Empty

| Value | Description |
|---|---|
| 0 | Data remains in SSC_THR or is currently transmitted from TSR. |
| 1 | Last data written in SSC_THR has been loaded in TSR and last data loaded in TSR has been transmitted. |

**Bit 0 – TXRDY** Transmit Ready

| Value | Description |
|---|---|
| 0 | Data has been loaded in SSC_THR and is waiting to be loaded in the transmit shift register (TSR). |
| 1 | SSC_THR is empty. |

### 44.9.14 SSC Interrupt Enable Register

**Name:** SSC_IER
**Offset:** 0x44
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXSYN | TXSYN | CP1 | CP0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| Access | | | W | W | | | W | W |
| Reset | | | – | – | | | – | – |

**Bit 11 – RXSYN** Rx Sync Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Rx Sync Interrupt. |

**Bit 10 – TXSYN** Tx Sync Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Tx Sync Interrupt. |

**Bit 9 – CP1** Compare 1 Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Compare 1 Interrupt. |

**Bit 8 – CP0** Compare 0 Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Compare 0 Interrupt. |

**Bit 5 – OVRUN** Receive Overrun Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Receive Overrun Interrupt. |

**Bit 4 – RXRDY** Receive Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | Enables the Receive Ready Interrupt. |

**Bit 1 – TXEMPTY** Transmit Empty Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit Empty Interrupt. |

**Bit 0 – TXRDY** Transmit Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the Transmit Ready Interrupt. |

### 44.9.15 SSC Interrupt Disable Register

**Name:** SSC_IDR
**Offset:** 0x48
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXSYN | TXSYN | CP1 | CP0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| Access | | | W | W | | | W | W |
| Reset | | | – | – | | | – | – |

**Bit 11 – RXSYN** Rx Sync Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Rx Sync Interrupt. |

**Bit 10 – TXSYN** Tx Sync Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Tx Sync Interrupt. |

**Bit 9 – CP1** Compare 1 Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Compare 1 Interrupt. |

**Bit 8 – CP0** Compare 0 Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Compare 0 Interrupt. |

**Bit 5 – OVRUN** Receive Overrun Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Receive Overrun Interrupt. |

**Bit 4 – RXRDY** Receive Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | Disables the Receive Ready Interrupt. |

**Bit 1 – TXEMPTY** Transmit Empty Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit Empty Interrupt. |

**Bit 0 – TXRDY** Transmit Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the Transmit Ready Interrupt. |

### 44.9.16 SSC Interrupt Mask Register

**Name:** SSC_IMR
**Offset:** 0x4C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RXSYN | TXSYN | CP1 | CP0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRUN | RXRDY | | | TXEMPTY | TXRDY |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 11 – RXSYN** Rx Sync Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Rx Sync Interrupt is disabled. |
| 1 | The Rx Sync Interrupt is enabled. |

**Bit 10 – TXSYN** Tx Sync Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Tx Sync Interrupt is disabled. |
| 1 | The Tx Sync Interrupt is enabled. |

**Bit 9 – CP1** Compare 1 Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Compare 1 Interrupt is disabled. |
| 1 | The Compare 1 Interrupt is enabled. |

**Bit 8 – CP0** Compare 0 Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Compare 0 Interrupt is disabled. |
| 1 | The Compare 0 Interrupt is enabled. |

**Bit 5 – OVRUN** Receive Overrun Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Receive Overrun Interrupt is disabled. |
| 1 | The Receive Overrun Interrupt is enabled. |

**Bit 4 – RXRDY** Receive Ready Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Receive Ready Interrupt is disabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | The Receive Ready Interrupt is enabled. |

**Bit 1 – TXEMPTY** Transmit Empty Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Transmit Empty Interrupt is disabled. |
| 1 | The Transmit Empty Interrupt is enabled. |

**Bit 0 – TXRDY** Transmit Ready Interrupt Mask

| Value | Description |
|---|---|
| 0 | The Transmit Ready Interrupt is disabled. |
| 1 | The Transmit Ready Interrupt is enabled. |

### 44.9.17 SSC Write Protection Mode Register

**Name:** SSC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x535343 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

See Register Write Protection for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII). |

### 44.9.18 SSC Write Protection Status Register

**Name:** SSC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protect Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the SSC_WPSR. |
| 1 | A write protection violation has occurred since the last read of the SSC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 45. Inter-IC Sound Controller (I2SC)

## 45.1 Description

The Inter-IC Sound Controller (I2SC) provides a 5-wire, bidirectional, synchronous, digital audio link to external audio devices: I2SC_DI, I2SC_DO, I2SC_WS, I2SC_CK, and I2SC_MCK pins.

The I2SC is compliant with the Inter-IC Sound ($I^2S$) bus specification.

The I2SC consists of a receiver, a transmitter and a common clock generator that can be enabled separately to provide Master, Slave or Controller modes with receiver and/or transmitter active.

DMA Controller channels, separate for the receiver and for the transmitter, allow a continuous high bit rate data transfer without processor intervention to the following:

- Audio CODECs in Master, Slave, or Controller mode
- Stereo DAC or ADC through a dedicated $I^2S$ serial interface

The I2SC can use either a single DMA Controller channel for both audio channels or one DMA Controller channel per audio channel.

The 8- and 16-bit compact stereo format reduces the required DMA Controller bandwidth by transferring the left and right samples within the same data word.

In Master mode, the I2SC can produce a 32 $f_s$ to 1024 $f_s$ master clock that provides an over-sampling clock to an external audio codec or digital signal processor (DSP).

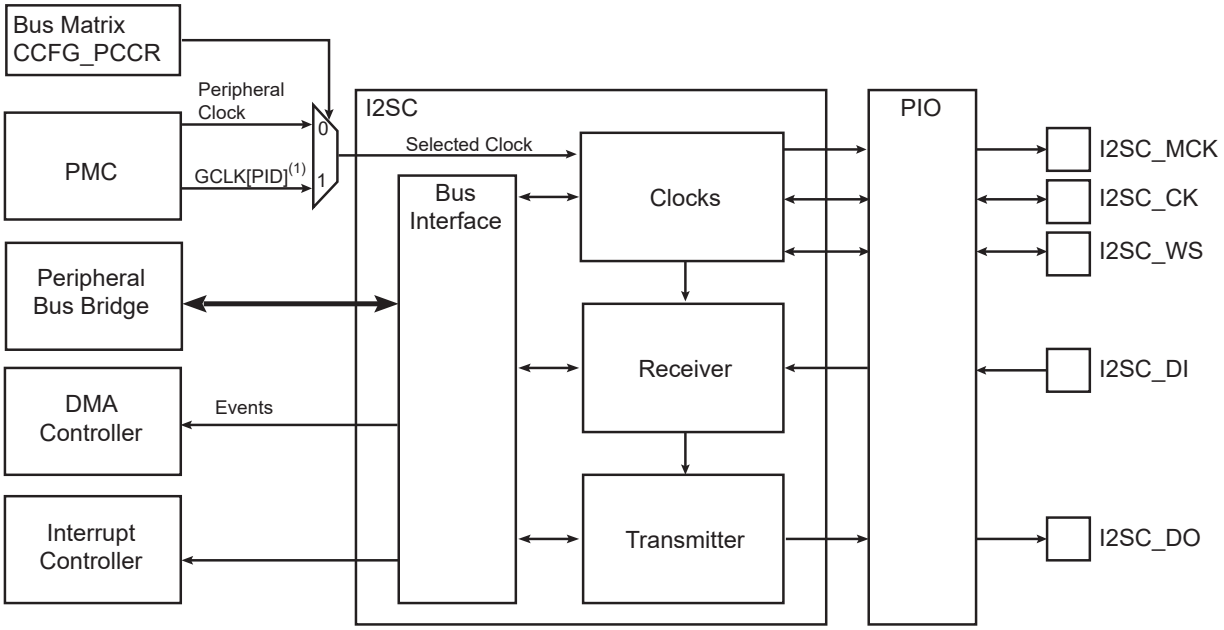
## 45.2 Embedded Characteristics

- Compliant with Inter-IC Sound ($I^2S$) Bus Specification
- Master, Slave, and Controller Modes
  - Slave: Data Received/Transmitted
  - Master: Data Received/Transmitted And Clocks Generated
  - Controller: Clocks Generated
- Individual Enable and Disable of Receiver, Transmitter and Clocks
- Configurable Clock Generator Common to Receiver and Transmitter
  - Suitable for a Wide Range of Sample Frequencies ($f_s$), Including 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, and 192 kHz
  - 32 $f_s$ to 1024 $f_s$ Master Clock Generated for External Oversampling Data Converters
- Support for Multiple Data Formats
  - 32-, 24-, 20-, 18-, 16-, and 8-bit Mono or Stereo Format
  - 16- and 8-bit Compact Stereo Format, with Left and Right Samples Packed in the Same Word to Reduce Data Transfers
- DMA Controller Interfaces the Receiver and Transmitter to Reduce Processor Overhead
  - One DMA Controller Channel for Both Audio Channels, or
  - One DMA Controller Channel Per Audio Channel
- Smart Holding Registers Management to Avoid Audio Channels Mix After Overrun or Underrun

© 2021 Microchip Technology Inc.
and its subsidiaries

## 45.3 Block Diagram

**Figure 45-1. I2SC Block Diagram**




(1) For the value of 'PID', refer to I2SCx in the table "Peripheral Identifiers".

**Related Links**

14.1 Peripheral Identifiers

## 45.4 I/O Lines Description

**Table 45-1. I/O Lines Description**

| Pin Name | Pin Description | Type |
|---|---|---|
| I2SC_MCK | Master Clock | Output |
| I2SC_CK | Serial Clock | Input/Output |
| I2SC_WS | $I^2S$ Word Select | Input/Output |
| I2SC_DI | Serial Data Input | Input |
| I2SC_DO | Serial Data Output | Output |

## 45.5 Product Dependencies

To use the I2SC, other parts of the system must be configured correctly, as described below.

### 45.5.1 I/O Lines

The I2SC pins may be multiplexed with I/O Controller lines. The user must first program the PIO Controller to assign the required I2SC pins to their peripheral function. If the I2SC I/O lines are not used by the application, they can be used for other purposes by the PIO Controller. The user must enable the I2SC inputs and outputs that are used.

### 45.5.2 Power Management

If the CPU enters a Sleep mode that disables clocks used by the I2SC, the I2SC stops functioning and resumes operation after the system wakes up from Sleep mode.

### 45.5.3 Clocks

The clock for the I2SC bus interface is generated by the Power Management Controller (PMC). I2SC must be disabled before disabling the clock to avoid freezing the I2SC in an undefined state.

### 45.5.4 DMA Controller

The I2SC interfaces to the DMA Controller. Using the I2SC DMA functionality requires the DMA Controller to be programmed first.

### 45.5.5 Interrupt Sources

The I2SC interrupt line is connected to the Interrupt Controller. Using the I2SC interrupt requires the Interrupt Controller to be programmed first.

## 45.6 Functional Description

### 45.6.1 Initialization

The I2SC features a receiver, a transmitter and a clock generator for Master and Controller modes. Receiver and transmitter share the same serial clock and word select.

Before enabling the I2SC, the selected configuration must be written to the I2SC Mode Register (I2SC_MR) and to the Peripheral Clock Configuration Register (CCFG_PCCR) described in the section “Bus Matrix (MATRIX)”.

If the I2SC_MR.IMCKMODE bit is set, the I2SC_MR.IMCKFS field must be configured as described in section “Serial Clock and Word Select Generation”.

Once the I2SC_MR has been written, the I2SC clock generator, receiver, and transmitter can be enabled by writing a ’1’ to the CKEN, RXEN, and TXEN bits in the Control Register (I2SC_CR). The clock generator can be enabled alone in Controller mode to output clocks to the I2SC_MCK, I2SC_CK, and I2SC_WS pins. The clock generator must also be enabled if the receiver or the transmitter is enabled.

The clock generator, receiver, and transmitter can be disabled independently by writing a ’1’ to I2SC_CR.CXDIS, I2SC_CR.RXDIS and/or I2SC_CR.TXDIS, respectively. Once requested to stop, they stop only when the transmission of the pending frame transmission is completed.

### 45.6.2 Basic Operation

The receiver can be operated by reading the Receiver Holding Register (I2SC_RHR), whenever the Receive Ready (RXRDY) bit in the Status Register (I2SC_SR) is set. Successive values read from RHR correspond to the samples from the left and right audio channels for the successive frames.

The transmitter can be operated by writing to the Transmitter Holding Register (I2SC_THR), whenever the Transmit Ready (TXRDY) bit in the I2SC_SR is set. Successive values written to THR correspond to the samples from the left and right audio channels for the successive frames.

The RXRDY and TXRDY bits can be polled by reading the I2SC_SR.

The I2SC processor load can be reduced by enabling interrupt-driven operation. The RXRDY and/or TXRDY interrupt requests can be enabled by writing a ’1’ to the corresponding bit in the Interrupt Enable Register (I2SC_IER). The interrupt service routine associated to the I2SC interrupt request is executed whenever the Receive Ready or the Transmit Ready status bit is set.

### 45.6.3 Master, Controller and Slave Modes

In Master and Controller modes, the I2SC provides the master clock, the serial clock and the word select. I2SC_MCK, I2SC_CK, and I2SC_WS pins are outputs.

In Controller mode, the I2SC receiver and transmitter are disabled. Only the clocks are enabled and used by an external receiver and/or transmitter.

In Slave mode, the I2SC receives the serial clock and the word select from an external master. I2SC_CK and I2SC_WS pins are inputs.

The mode is selected by writing the MODE field in the I2SC_MR. Since the MODE field changes the direction of the I2SC_WS and I2SC_SCK pins, the I2SC_MR must be written when the I2SC is stopped.

© 2021 Microchip Technology Inc.
and its subsidiaries

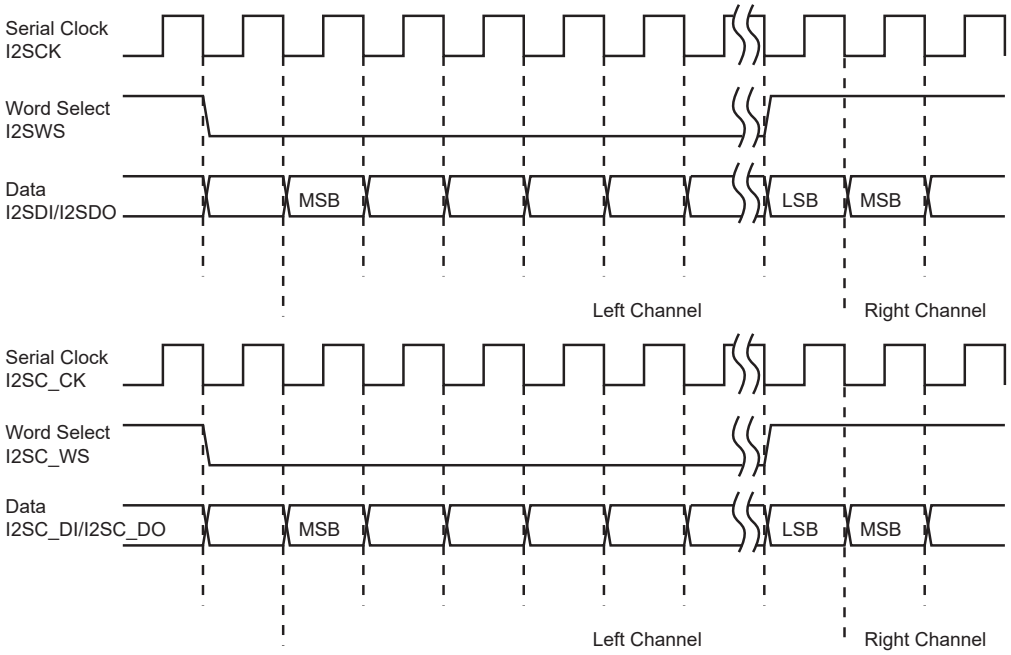
**Related Links**

19. Bus Matrix (MATRIX)

### 45.6.4 I²S Reception and Transmission Sequence

As specified in the I²S protocol, data bits are left-justified in the word select time slot, with the MSB transmitted first, starting one clock period after the transition on the word select line.

**Figure 45-2. I²S Reception and Transmission Sequence**




Data bits are sent on the falling edge of the serial clock and sampled on the rising edge of the serial clock. The word select line indicates the channel in transmission, a low level for the left channel and a high level for the right channel.

The length of transmitted words can be chosen among 8, 16, 18, 20, 24, and 32 bits by writing the I2SC_MR.DATALENGTH field.

If the time slot allows for more data bits than written in the I2SC_MR.DATALENGTH field, zeroes are appended to the transmitted data word or extra received bits are discarded.

### 45.6.5 Serial Clock and Word Select Generation

The generation of clocks in the I2SC is described in figure ”Mono”.

In Slave mode, the serial clock and word select clock are driven by an external master. I2SC_CK and I2SC_WS pins are inputs.

In Master mode, the user can configure the master clock, serial clock, and word select clock through the I2SC_MR. I2SC_MCK, I2SC_CK, and I2SC_WS pins are outputs and MCK is used to derive the I2SC clocks.

In Master mode, if the peripheral clock frequency is higher than 96 MHz, the GCLK[PID] from PMC must be selected as I2SC input clock by writing a ‘1’ in the I2SCxCC bit of the CCFG_PCCR register. Refer to the section “Bus Matrix (MATRIX)” for more details.

Audio codecs connected to the I2SC pins may require a master clock (I2SC_MCK) signal with a frequency multiple of the audio sample frequency ($f_s$), such as 256$f_s$. When the I2SC is in Master mode, writing a ’1’ to I2SC_MR.IMCKMODE outputs MCK as master clock to the I2SC_MCK pin, and divides MCK to create the internal bit clock, output on the I2SC_CK pin. The clock division factor is defined by writing to I2SC_MR.IMCKFS and I2SC_MR.DATALENGTH, as described in the I2SC_MR.IMCKFS field description.

The master clock (I2SC_MCK) frequency is (2×16 × (IMCKFS + 1)) / (IMCKDIV + 1) times the sample frequency ($f_s$), i.e., I2SC_WS frequency.

Example: If the sampling rate is 44.1 kHz with an I2S master clock (I2SC_MCK) ratio of 256, the core frequency must be an integer multiple of 11.2896 MHz. Assuming an integer multiple of 4, the IMCKDIV field must be configured to 4; the field IMCKFS must then be set to 31.

The serial clock (I2SC_CK) frequency is 2 × Slot Length times the sample frequency ($f_s$), where Slot Length is defined in the following table.

**Table 45-2. Slot Length**

| I2SC_MR.DATALENGTH | Word Length | Slot Length |
|---|---|---|
| 0 | 32 bits | 32 |
| 1 | 24 bits | 32 if I2SC_MR.IWS = 0<br>24 if I2SC_MR.IWS = 1 |
| 2 | 20 bits | |
| 3 | 18 bits | |
| 4 | 16 bits | 16 |
| 5 | 16 bits compact stereo | |
| 6 | 8 bits | 8 |
| 7 | 8 bits compact stereo | |

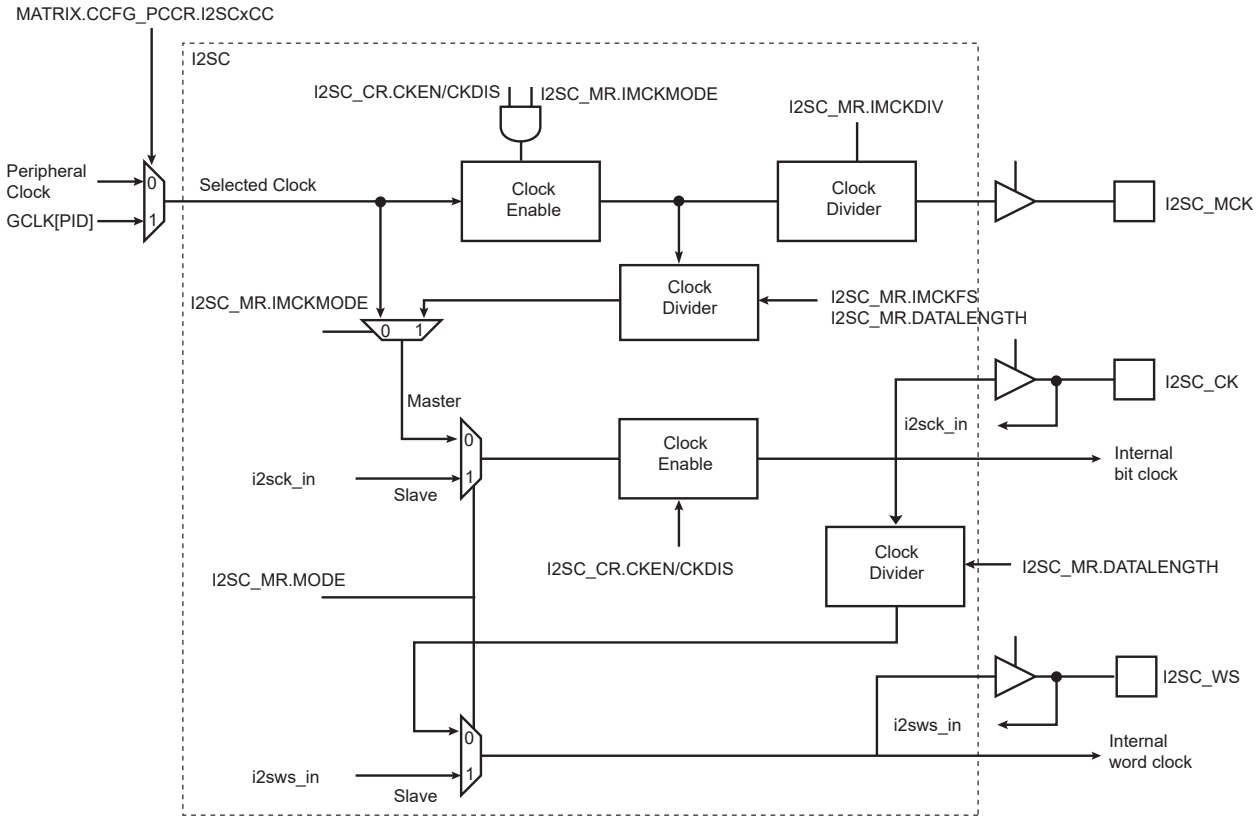
**⚠WARNING** I2SC_MR.IMCKMODE must be written to ’1’ if the master clock frequency is strictly higher than the serial clock.

If a master clock output is not required, the MCK clock is used as I2SC_CK by clearing I2SC_MR.IMCKMODE. Alternatively, if the frequency of the MCK clock used is a multiple of the required I2SC_CK frequency, the I2SC_MCK to I2SC_CK divider can be used with the ratio defined by writing the I2SC_MR.IMCKFS field.

The I2SC_WS pin is used as word select as described in section “I2S Reception and Transmission Sequence”.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 45-3. I2SC Clock Generation**




### 45.6.6 Mono

When the Transmit Mono bit (TXMONO) in I2SC_MR is set, data written to the left channel is duplicated to the right output channel.

When the Receive Mono bit (RXMONO) in I2SC_MR is set, data received from the left channel is duplicated to the right channel.

### 45.6.7 Holding Registers

The I2SC user interface includes a Receive Holding Register (I2SC_RHR) and a Transmit Holding Register (I2SC_THR). These registers are used to access audio samples for both audio channels.

When a new data word is available in I2SC_RHR, the Receive Ready bit (RXRDY) in I2SC_SR is set. Reading I2SC_RHR clears this bit.

A receive overrun condition occurs if a new data word becomes available before the previous data word has been read from I2SC_RHR. In this case, the Receive Overrun bit in I2SC_SR and bit i of the RXORCH field in I2SC_SR are set, where i is the current receive channel number.

When I2SC_THR is empty, the Transmit Ready bit (TXRDY) in I2SC_SR is set. Writing to I2SC_THR clears this bit.

A transmit underrun condition occurs if a new data word needs to be transmitted before it has been written to I2SC_THR. In this case, the Transmit Underrun (TXUR) bit and bit i of the TXORCH field in I2SC_SR are set, where i is the current transmit channel number. If the TXSAME bit in I2SC_MR is ’0’, then a zero data word is transmitted in case of underrun. If I2SC_MR.TXSAME is ’1’, then the previous data word for the current transmit channel number is transmitted.

Data words are right-justified in I2SC_RHR and I2SC_THR. For the 16-bit compact stereo data format, the left sample uses bits 15:0 and the right sample uses bits 31:16 of the same data word. For the 8-bit compact stereo data format, the left sample uses bits 7:0 and the right sample uses bits 15:8 of the same data word.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.6.8 DMA Controller Operation

All receiver audio channels can be assigned to a single DMA Controller channel or individual audio channels can be assigned to one DMA Controller channel per audio channel. The same channel assignment choice applies to the transmitter audio channels.

Channel assignment is selected by writing to the I2SC_MR.RXDMA and I2SC_MR.TXDMA bits. If a single DMA Controller channel is selected, all data samples use I2SC receiver or transmitter DMA Controller channel 0.

The DMA Controller reads from the I2SC_RHR and writes to the I2SC_THR for both audio channels successively.

The DMA Controller transfers may use 32-bit word, 16-bit halfword, or 8-bit byte depending on the value of the I2SC_MR.DATALENGTH field.

### 45.6.9 Loopback Mode

For debug purposes, the I2SC can be configured to loop back the transmitter to the Receiver. Writing a ’1’ to the I2SC_MR.LOOP bit internally connects I2SC_DO to I2SC_DI, so that the transmitted data is also received. Writing a ’0’ to I2SC_MR.LOOP restores the normal behavior with independent Receiver and Transmitter. As for other changes to the Receiver or Transmitter configuration, the I2SC Receiver and Transmitter must be disabled before writing to I2SC_MR to update I2SC_MR.LOOP.

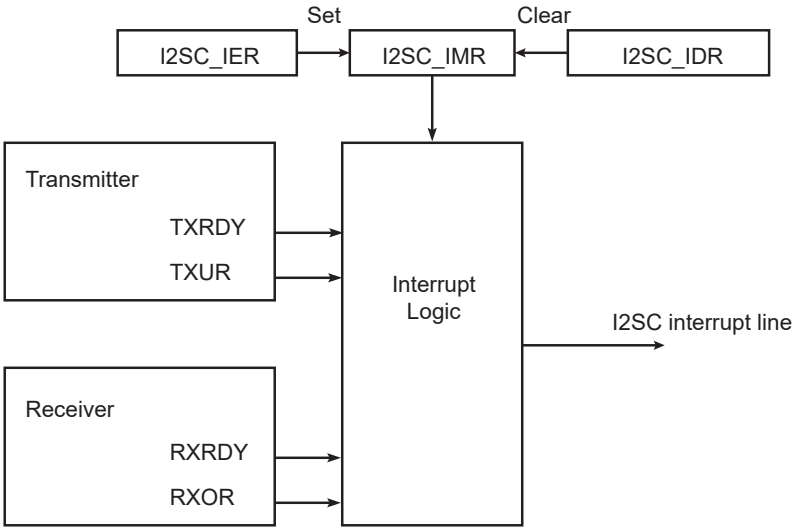
### 45.6.10 Interrupts

An I2SC interrupt request can be triggered whenever one or several of the following bits are set in I2SC_SR: Receive Ready (RXRDY), Receive Overrun (RXOR), Transmit Ready (TXRDY) or Transmit Underrun (TXUR).

The interrupt request is generated if the corresponding bit in the Interrupt Mask Register (I2SC_IMR) is set. Bits in I2SC_IMR are set by writing a ’1’ to the corresponding bit in I2SC_IER and cleared by writing a ’1’ to the corresponding bit in the Interrupt Disable Register (I2SC_IDR). The interrupt request remains active until the corresponding bit in I2SC_SR is cleared by writing a ’1’ to the corresponding bit in the Status Clear Register (I2SC_SCR).

For debug purposes, interrupt requests can be simulated by writing a ’1’ to the corresponding bit in the Status Set Register (I2SC_SSR).

**Figure 45-4. Interrupt Block Diagram**




## 45.7 I2SC Application Examples

The I2SC supports several serial communication modes used in audio or high-speed serial links. Examples of standard applications are shown in the following figures. All serial link applications supported by the I2SC are not listed here.

**Figure 45-5. Slave Transmitter I2SC Application Example**

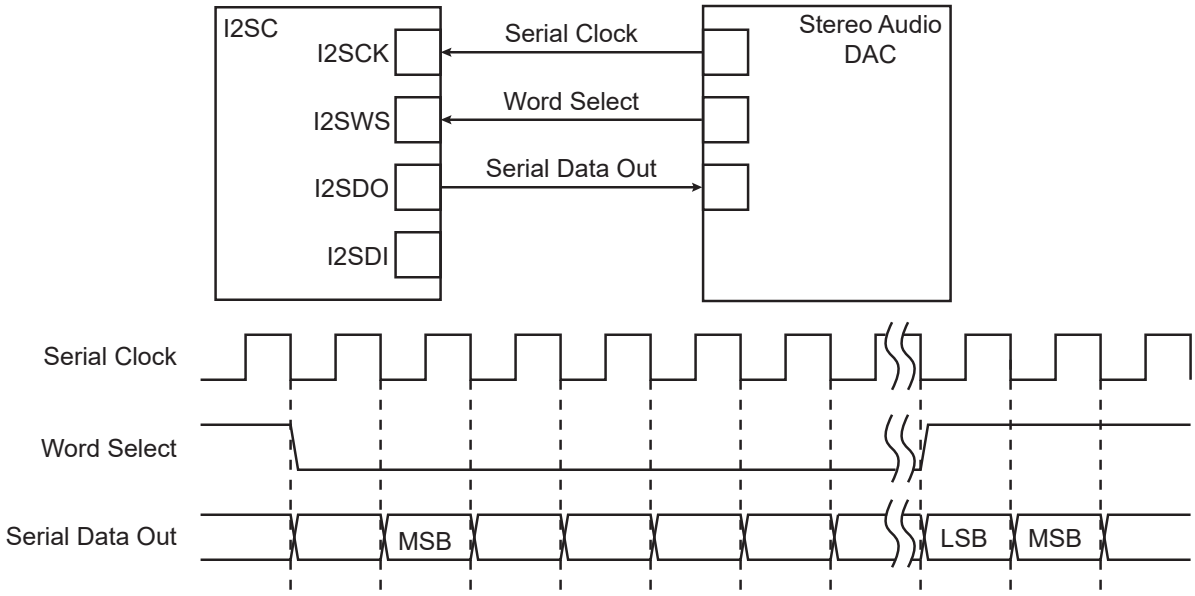
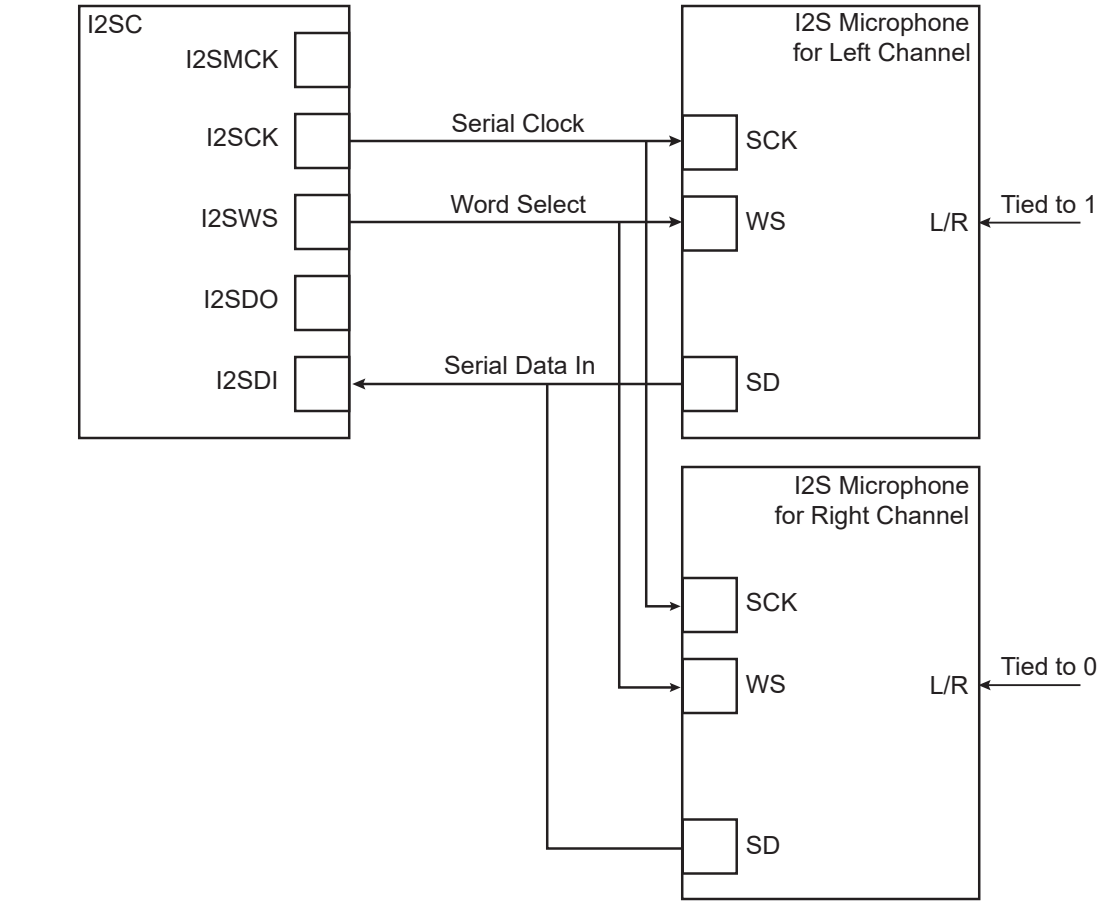


**Figure 45-6. Dual Microphone Application Block Diagram**



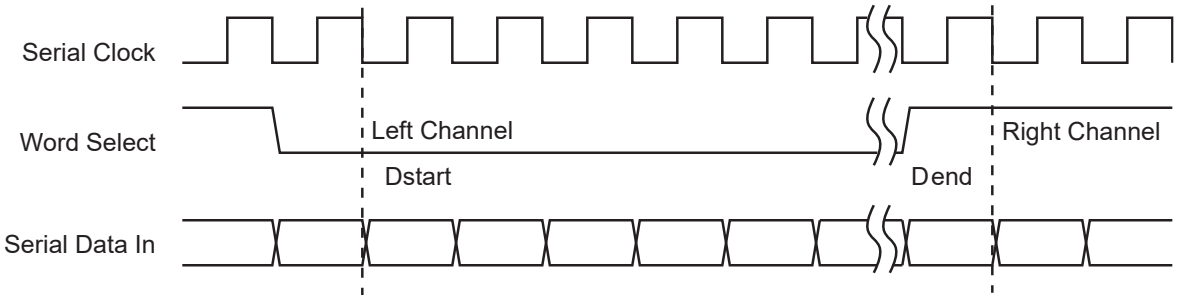
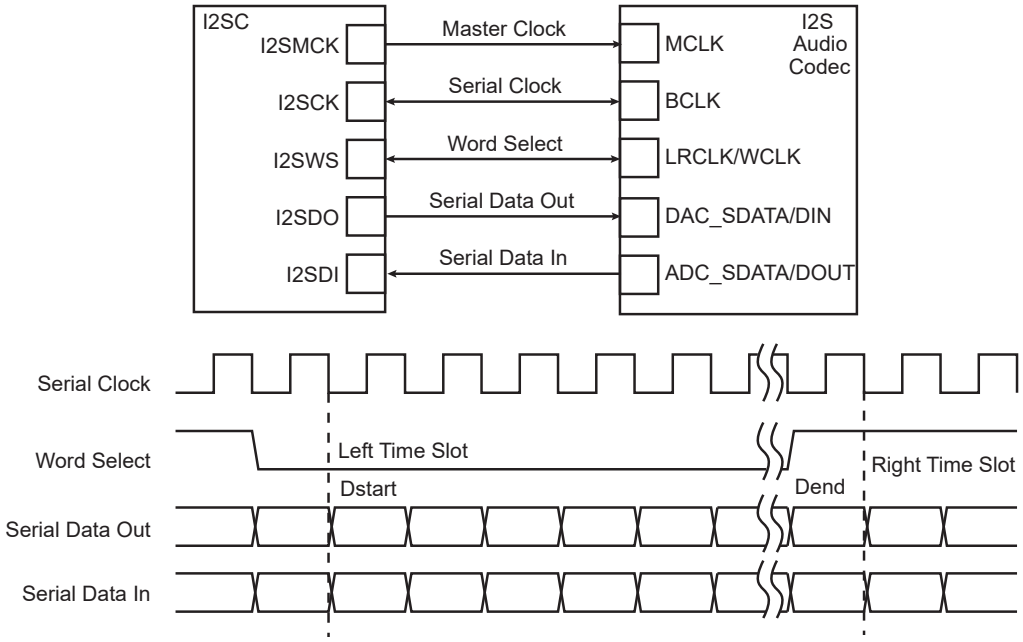



**Figure 45-7. Codec Application Block Diagram**



## 45.8 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | I2SC_CR | 7:0 | SWRST | | TXDIS | TXEN | CKDIS | CKEN | RXDIS | RXEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | I2SC_MR | 7:0 | | | | DATALENGTH[2:0] | | | | MODE |
| | | 15:8 | | TXSAME | TXDMA | TXMONO | | RXLOOP | RXDMA | RXMONO |
| | | 23:16 | | | IMCKDIV[5:0] | | | | | |
| | | 31:24 | IWS | IMCKMODE | IMCKFS[5:0] | | | | | |
| 0x08 | I2SC_SR | 7:0 | | TXUR | TXRDY | TXEN | | RXOR | RXRDY | RXEN |
| | | 15:8 | | | | | | | RXORCH[1:0] | |
| | | 23:16 | | | TXURCH[1:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | I2SC_SCR | 7:0 | | TXUR | | | | RXOR | | |
| | | 15:8 | | | | | | | RXORCH[1:0] | |
| | | 23:16 | | | TXURCH[1:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | I2SC_SSR | 7:0 | | TXUR | | | | RXOR | | |
| | | 15:8 | | | | | | | RXORCH[1:0] | |
| | | 23:16 | | | TXURCH[1:0] | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | I2SC_IER | 7:0 | | TXUR | TXRDY | | | RXOR | RXRDY | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | I2SC_IDR | 7:0 | | TXUR | TXRDY | | | RXOR | RXRDY | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | I2SC_IMR | 7:0 | | TXUR | TXRDY | | | RXOR | RXRDY | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | I2SC_RHR | 7:0 | RHR[7:0] | | | | | | | |
| | | 15:8 | RHR[15:8] | | | | | | | |
| | | 23:16 | RHR[23:16] | | | | | | | |
| | | 31:24 | RHR[31:24] | | | | | | | |
| 0x24 | I2SC_THR | 7:0 | THR[7:0] | | | | | | | |
| | | 15:8 | THR[15:8] | | | | | | | |
| | | 23:16 | THR[23:16] | | | | | | | |
| | | 31:24 | THR[31:24] | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.8.1 I2SC Control Register

**Name:** I2SC_CR
**Offset:** 0x00
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SWRST | | TXDIS | TXEN | CKDIS | CKEN | RXDIS | RXEN |
| Access | W | | W | W | W | W | W | W |
| Reset | – | | – | – | – | – | – | – |

**Bit 7 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit resets all the registers in the I2SC. The I2SC is disabled after the reset. |

**Bit 5 – TXDIS** Transmitter Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit disables the I2SC transmitter. Bit I2SC_SR.TXEN is cleared when the Transmitter is stopped. |

**Bit 4 – TXEN** Transmitter Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit enables the I2SC transmitter, if TXDIS is not one. Bit I2SC_SR.TXEN is set when the Transmitter is started. |

**Bit 3 – CKDIS** Clocks Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a zone to this bit disables the I2SC clock generation. |

**Bit 2 – CKEN** Clocks Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit enables the I2SC clocks generation, if CKDIS is not one. |

**Bit 1 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | Writing a ’1’ to this bit disables the I2SC receiver. Bit I2SC_SR.RXEN is cleared when the receiver is stopped. |

**Bit 0 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit enables the I2SC receiver, if RXDIS is not one. Bit I2SC_SR.RXEN is set when the receiver is activated. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.8.2 I2SC Mode Register

**Name:** I2SC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

The I2SC_MR must be written when the I2SC is stopped. The proper sequence is to write to I2SC_MR, then write to I2SC_CR to enable the I2SC or to disable the I2SC before writing a new value to I2SC_MR.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IWS | IMCKMODE | IMCKFS[5:0] | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | IMCKDIV[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | TXSAME | TXDMA | TXMONO | | RXLOOP | RXDMA | RXMONO |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | DATALENGTH[2:0] | | | | MODE |
| Access | | | | R/W | R/W | R/W | | R/W |
| Reset | | | | 0 | 0 | 0 | | 0 |

**Bit 31 – IWS** I2SC_WS Slot Width
Refer to table Slot Length (I2S format).

| Value | Description |
|---|---|
| 0 | I2SC_WS slot is 32 bits wide for DATALENGTH = 18/20/24 bits. |
| 1 | I2SC_WS slot is 24 bits wide for DATALENGTH = 18/20/24 bits. |

**Bit 30 – IMCKMODE** Master Clock Mode
If I2SC_MCK frequency is the same as I2SC_CK, IMCKMODE must be cleared. Refer to section Serial Clock and Word Select Generation and table Slot Length.

| Value | Description |
|---|---|
| 0 | No master clock generated (Selected Clock drives I2SC_CK output). |
| 1 | Master clock generated (internally generated clock is used as I2SC_MCK output). |

**Bits 29:24 – IMCKFS[5:0]** Master Clock to $f_s$ Ratio
Master clock frequency is [2 x 16 × (IMCKFS + 1)] / (IMCKDIV + 1) times the sample rate, i.e., I2SC_WS frequency.

| Value | Name | Description |
|---|---|---|
| 0 | M2SF32 | Sample frequency ratio set to 32 |
| 1 | M2SF64 | Sample frequency ratio set to 64 |
| 2 | M2SF96 | Sample frequency ratio set to 96 |
| 3 | M2SF128 | Sample frequency ratio set to 128 |
| 5 | M2SF192 | Sample frequency ratio set to 192 |
| 7 | M2SF256 | Sample frequency ratio set to 256 |
| 11 | M2SF384 | Sample frequency ratio set to 384 |
| 15 | M2SF512 | Sample frequency ratio set to 512 |
| 23 | M2SF768 | Sample frequency ratio set to 768 |
| 31 | M2SF1024 | Sample frequency ratio set to 1024 |

© 2021 Microchip Technology Inc. and its subsidiaries

| Value | Name | Description |
|---|---|---|
| 47 | M2SF1536 | Sample frequency ratio set to 1536 |
| 63 | M2SF2048 | Sample frequency ratio set to 2048 |

**Bits 21:16 – IMCKDIV[5:0]** Selected Clock to I2SC Master Clock Ratio
I2SC_MCK Master clock output frequency is Selected Clock divided by (IMCKDIV + 1). Refer to the IMCKFS field description.
**Notes:**

1. This field is write-only. Always read as '0'.
2. Do not write a '0' to this field.

**Bit 14 – TXSAME** Transmit Data when Underrun

| Value | Description |
|---|---|
| 0 | Zero sample transmitted when underrun. |
| 1 | Previous sample transmitted when underrun |

**Bit 13 – TXDMA** Single or Multiple DMA Controller Channels for TransmitterDMA Controller Channels for Transmitter

| Value | Description |
|---|---|
| 0 | The transmitter uses only one DMA Controller channel for all audio channels. |
| 1 | The transmitter uses one DMA Controller channel per audio channel. |

**Bit 12 – TXMONO** Transmit Mono

| Value | Description |
|---|---|
| 0 | Stereo |
| 1 | Mono, with left audio samples duplicated to right audio channel by the I2SC. |

**Bit 10 – RXLOOP** Loopback Test Mode

| Value | Description |
|---|---|
| 0 | Normal mode |
| 1 | I2SC_DO output of I2SC is internally connected to I2SC_DI input. |

**Bit 9 – RXDMA** Single or Multiple DMA Controller Channels for Receiver

| Value | Description |
|---|---|
| 0 | The receiver uses only one DMA Controller channel for all audio channels. |
| 1 | The receiver uses one DMA Controller channel per audio channel. |

**Bit 8 – RXMONO** Receive Mono

| Value | Description |
|---|---|
| 0 | Stereo |
| 1 | Mono, with left audio samples duplicated to right audio channel by the I2SC. |

**Bits 4:2 – DATALENGTH[2:0]** Data Word Length

| Value | Name | Description |
|---|---|---|
| 0 | 32_BITS | Data length is set to 32 bits. |
| 1 | 24_BITS | Data length is set to 24 bits. |
| 2 | 20_BITS | Data length is set to 20 bits. |
| 3 | 18_BITS | Data length is set to 18 bits. |
| 4 | 16_BITS | Data length is set to 16 bits. |
| 5 | 16_BITS_COMPACT | Data length is set to 16-bit compact stereo. Left sample in bits 15:0 and right sample in bits 31:16 of same word. |
| 6 | 8_BITS | Data length is set to 8 bits. |
| 7 | 8_BITS_COMPACT | Data length is set to 8-bit compact stereo. Left sample in bits 7:0 and right sample in bits 15:8 of the same word. |

**Bit 0 – MODE** Inter-IC Sound Controller Mode

| Value | Name | Description |
|---|---|---|
| 0 | SLAVE | I2SC_CK and I2SC_WS pin inputs used as bit clock and word select/frame synchronization. |
| 1 | MASTER | Bit clock and word select/frame synchronization generated by I2SC from MCK and output to I2SC_CK and I2SC_WS pins. Peripheral clock or GCLK is output as master clock on I2SC_MCK if I2SC_MR.IMCKMODE is set. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.8.3 I2SC Status Register

**Name:** I2SC_SR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURCH[1:0] | | | | | |
| Access | | | R | R | | | | |
| Reset | | | 0 | 0 | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXORCH[1:0] | |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | TXRDY | TXEN | | RXOR | RXRDY | RXEN |
| Access | | R | R | R | | R | R | R |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 21:20 – TXURCH[1:0]** Transmit Underrun Channel

| Value | Description |
|---|---|
| 0 | This field is cleared when I2SC_SCR.TXUR is written to ’1’. |
| 1 | Bit i of this field is set when a transmit underrun error occurred in channel i (i = 0 for first channel of the frame). |

**Bits 9:8 – RXORCH[1:0]** Receive Overrun Channel
This field is cleared when I2SC_SCR.RXOR is written to ’1’.
Bit i of this field is set when a receive overrun error occurred in channel i (i = 0 for first channel of the frame).

**Bit 6 – TXUR** Transmit Underrun

| Value | Description |
|---|---|
| 0 | This bit is cleared when the corresponding bit in I2SC_SCR is written to ’1’. |
| 1 | This bit is set when an underrun error occurs on I2SC_THR or when the corresponding bit in I2SC_SSR is written to ’1’. |

**Bit 5 – TXRDY** Transmit Ready

| Value | Description |
|---|---|
| 0 | This bit is cleared when data is written to I2SC_THR. |
| 1 | This bit is set when I2SC_THR is empty and can be written with new data to be transmitted. |

**Bit 4 – TXEN** Transmitter Enabled

| Value | Description |
|---|---|
| 0 | This bit is cleared when the transmitter is disabled, following a I2SC_CR.TXDIS or I2SC_CR.SWRST request. |
| 1 | This bit is set when the transmitter is enabled, following a I2SC_CR.TXEN request. |

**Bit 2 – RXOR** Receive Overrun

| Value | Description |
|---|---|
| 0 | This bit is cleared when the corresponding bit in I2SC_SCR is written to ’1’. |
| 1 | This bit is set when an overrun error occurs on I2SC_RHR or when the corresponding bit in I2SC_SSR is written to ’1’. |

**Bit 1 – RXRDY** Receive Ready

| Value | Description |
|---|---|
| 0 | This bit is cleared when I2SC_RHR is read. |
| 1 | This bit is set when received data is present in I2SC_RHR. |

**Bit 0 – RXEN** Receiver Enabled

| Value | Description |
|---|---|
| 0 | This bit is cleared when the receiver is disabled, following a RXDIS or SWRST request in I2SC_CR. |
| 1 | This bit is set when the receiver is enabled, following a RXEN request in I2SC_CR. |

### 45.8.4 I2SC Status Clear Register

**Name:** I2SC_SCR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURCH[1:0] | | | | | |
| Access | | | W | W | | | | |
| Reset | | | – | – | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXORCH[1:0] | |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | | | | RXOR | | |
| Access | | W | | | | W | | |
| Reset | | – | | | | – | | |

**Bits 21:20 – TXURCH[1:0]** Transmit Underrun Per Channel Status Clear
Writing a ’0’ has no effect.
Writing a ’1’ to any bit in this field clears the corresponding bit in the I2SC_SR and the corresponding interrupt request.

**Bits 9:8 – RXORCH[1:0]** Receive Overrun Per Channel Status Clear
Writing a ’0’ has no effect.
Writing a ’1’ to any bit in this field clears the corresponding bit in the I2SC_SR and the corresponding interrupt request.

**Bit 6 – TXUR** Transmit Underrun Status Clear
Writing a ’0’ to this bit has no effect.
Writing a ’1’ to this bit clears the status bit.

**Bit 2 – RXOR** Receive Overrun Status Clear
Writing a ’0’ to this bit has no effect.
Writing a ’1’ to this bit clears the status bit.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.8.5 I2SC Status Set Register

**Name:** I2SC_SSR
**Offset:** 0x10
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TXURCH[1:0] | | | | | |
| Access | | | W | W | | | | |
| Reset | | | – | – | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXORCH[1:0] | |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | | | | RXOR | | |
| Access | | W | | | | W | | |
| Reset | | – | | | | – | | |

**Bits 21:20 – TXURCH[1:0]** Transmit Underrun Per Channel Status Set
Writing a ’0’ has no effect.
Writing a ’1’ to any bit in this field sets the corresponding bit in I2SC_SR and the corresponding interrupt request.

**Bits 9:8 – RXORCH[1:0]** Receive Overrun Per Channel Status Set
Writing a ’0’ has no effect.
Writing a ’1’ to any bit in this field sets the corresponding bit in I2SC_SR and the corresponding interrupt request.

**Bit 6 – TXUR** Transmit Underrun Status Set
Writing a ’0’ to this bit has no effect.
Writing a ’1’ to this bit sets the status bit.

**Bit 2 – RXOR** Receive Overrun Status Set
Writing a ’0’ to this bit has no effect.
Writing a ’1’ to this bit sets the status bit.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.8.6 I2SC Interrupt Enable Register

**Name:** I2SC_IER
**Offset:** 0x14
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | TXRDY | | | RXOR | RXRDY | |
| Access | | W | W | | | W | W | |
| Reset | | – | – | | | – | – | |

**Bit 6 – TXUR** Transmit Underflow Interrupt Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR. |

**Bit 5 – TXRDY** Transmit Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit as no effect. |
| 1 | Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR. |

**Bit 2 – RXOR** Receiver Overrun Interrupt Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR. |

**Bit 1 – RXRDY** Receiver Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.8.7 I2SC Interrupt Disable Register

**Name:** I2SC_IDR
**Offset:** 0x18
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | TXRDY | | | RXOR | RXRDY | |
| Access | | W | W | | | W | W | |
| Reset | | – | – | | | – | – | |

**Bit 6 – TXUR** Transmit Underflow Interrupt Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR. |

**Bit 5 – TXRDY** Transmit Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR. |

**Bit 2 – RXOR** Receiver Overrun Interrupt Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR. |

**Bit 1 – RXRDY** Receiver Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | Writing a ’0’ to this bit has no effect. |
| 1 | Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.8.8 I2SC Interrupt Mask Register

**Name:** I2SC_IMR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXUR | TXRDY | | | RXOR | RXRDY | |
| Access | | R | R | | | R | R | |
| Reset | | 0 | 0 | | | 0 | 0 | |

**Bit 6 – TXUR** Transmit Underflow Interrupt Disable

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’. |
| 1 | The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’. |

**Bit 5 – TXRDY** Transmit Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’. |
| 1 | The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’. |

**Bit 2 – RXOR** Receiver Overrun Interrupt Disable

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’. |
| 1 | The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’. |

**Bit 1 – RXRDY** Receiver Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’. |
| 1 | The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 45.8.9 I2SC Receiver Holding Register

**Name:** I2SC_RHR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RHR[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RHR[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RHR[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RHR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RHR[31:0]** Receiver Holding Register
This field is set by hardware to the last received data word. If I2SC_MR.DATALENGTH specifies fewer than 32 bits, data is right-justified in the RHR field.

### 45.8.10 I2SC Transmitter Holding Register

**Name:** I2SC_THR
**Offset:** 0x24
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | THR[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | THR[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | THR[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | THR[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 31:0 – THR[31:0]** Transmitter Holding Register
Next data word to be transmitted after the current word if TXRDY is not set. If I2SC_MR.DATALENGTH specifies fewer than 32 bits, data is right-justified in the THR field.

# 46. Universal Synchronous Asynchronous Receiver Transceiver (USART)

## 46.1 Description

The Universal Synchronous Asynchronous Receiver Transceiver (USART) provides one full duplex universal synchronous asynchronous serial link. Data frame format is widely programmable (data length, parity, number of stop bits) to support a maximum of standards. The receiver implements parity error, framing error and overrun error detection. The receiver timeout enables handling variable-length frames and the transmitter timeguard facilitates communications with slow remote devices. Multidrop communications are also supported through address bit handling in reception and transmission.

The USART features three test modes: Remote Loopback, Local Loopback and Automatic Echo.

The USART supports specific operating modes providing interfaces on RS485, LIN, LON, and SPI buses, with ISO7816 T = 0 or T = 1 smart card slots, infrared transceivers and connection to modem ports. The hardware handshaking feature enables an out-of-band flow control by automatic management of the pins RTS and CTS.

The USART supports the connection to the DMA Controller, which enables data transfers to the transmitter and from the receiver. The DMAC provides chained buffer management without any intervention of the processor.

## 46.2 Embedded Characteristics

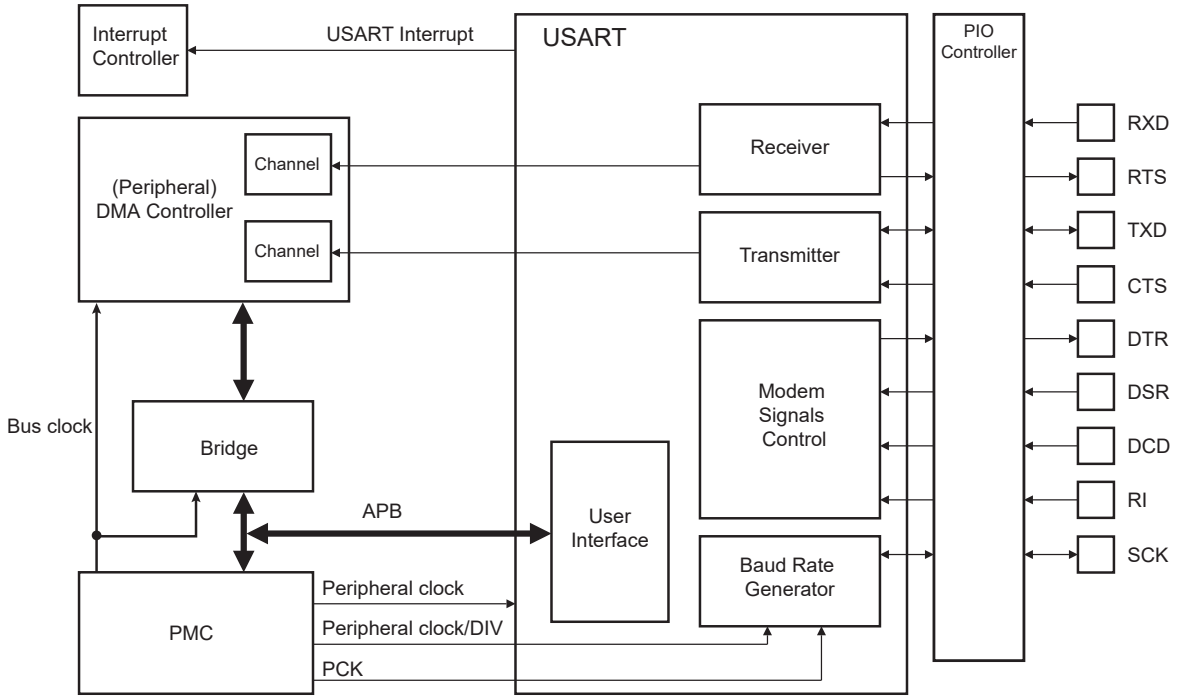
- Programmable Baud Rate Generator
- 5- to 9-bit Full-duplex Synchronous or Asynchronous Serial Communications
  - 1, 1.5 or 2 Stop Bits in Asynchronous Mode or 1 or 2 Stop Bits in Synchronous Mode
  - Parity Generation and Error Detection
  - Framing Error Detection, Overrun Error Detection
  - Digital Filter on Receive Line
  - MSB- or LSB-first
  - Optional Break Generation and Detection
  - By 8 or by 16 Oversampling Receiver Frequency
  - Optional Hardware Handshaking RTS-CTS
  - Optional Modem Signal Management DTR-DSR-DCD-RI
  - Receiver Timeout and Transmitter Timeguard
  - Optional Multidrop Mode with Address Generation and Detection
- RS485 with Driver Control Signal
- ISO7816, T = 0 or T = 1 Protocols for Interfacing with Smart Cards
  - NACK Handling, Error Counter with Repetition and Iteration Limit
- IrDA Modulation and Demodulation
  - Communication at up to 115.2 kbit/s
- SPI Mode
  - Master or Slave
  - Serial Clock Programmable Phase and Polarity
  - SPI Serial Clock (SCK) Frequency up to $f_{peripheral\ clock}/6$
- LIN Mode
  - Compliant with LIN 1.3 and LIN 2.0 SPECIFICATIONS
  - Master or Slave
  - Processing of Frames with Up to 256 Data Bytes
  - Response Data Length can be Configurable or Defined Automatically by the Identifier
  - Self-synchronization in Slave Node Configuration
  - Automatic Processing and Verification of the “Synch Break” and the “Synch Field”

© 2021 Microchip Technology Inc.
and its subsidiaries

  - "Synch Break" Detection Even When Partially Superimposed with a Data Byte
  - Automatic Identifier Parity Calculation/Sending and Verification
  - Parity Sending and Verification Can be Disabled
  - Automatic Checksum Calculation/sending and Verification
  - Checksum Sending and Verification Can be Disabled
  - Support Both "Classic" and "Enhanced" Checksum Types
  - Full LIN Error Checking and Reporting
  - Frame Slot Mode: Master Allocates Slots to the Scheduled Frames Automatically
  - Generation of the Wakeup Signal
- LON Mode
  - Compliant with CEA-709 Specification
  - Full-layer 2 Implementation
  - Differential Manchester Encoding/Decoding (CDP)
  - Preamble Generation Including Bit- and Byte-sync Fields
  - LON Timings Handling (beta1, beta2, IDT, etc.)
  - CRC Generation and Checking
  - Automated Random Number Generation
  - Backlog Calculation and Update
  - Collision Detection Support
  - Supports Both comm_type=1 and comm_type=2 Modes
  - Clock Drift Tolerance Up to 16%
  - Optimal for Node-to-Node Communication (no embedded digital line filter)
- Test Modes
  - Remote Loopback, Local Loopback, Automatic Echo
- Supports Connection of:
  - Two DMA Controller Channels (DMAC)
- Offers Buffer Transfer without Processor Intervention
- Register Write Protection

## 46.3 Block Diagram

**Figure 46-1. USART Block Diagram**




## 46.4 I/O Lines Description

**Table 46-1. I/O Line Description**

| Name | Description | Type | Active Level |
|---|---|---|---|
| SCK | Serial Clock | I/O | — |
| TXD | Transmit Serial Data<br>or Master Out Slave In (MOSI) in SPI Master mode<br>or Master In Slave Out (MISO) in SPI Slave mode | I/O | — |
| RXD | Receive Serial Data<br>or Master In Slave Out (MISO) in SPI Master mode<br>or Master Out Slave In (MOSI) in SPI Slave mode | Input | — |
| RI | Ring Indicator | Input | Low |
| DSR | Data Set Ready | Input | Low |
| DCD | Data Carrier Detect | Input | Low |
| DTR | Data Terminal Ready | Output | Low |
| LCOL | LON Collision Detection | Input | Low |
| CTS | Clear to Send<br>or Slave Select (NSS) in SPI Slave mode | Input | Low |
| RTS | Request to Send<br>or Slave Select (NSS) in SPI Master mode | Output | Low |

## 46.5 Product Dependencies

### 46.5.1 I/O Lines

The pins used for interfacing the USART may be multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the desired USART pins to their peripheral function. If I/O lines of the USART are not used by the application, they can be used for other purposes by the PIO Controller.

All the pins of the modems may or may not be implemented on the USART. On USARTs not equipped with the corresponding pin, the associated control bits and statuses have no effect on the behavior of the USART.

### 46.5.2 Power Management

The USART is not continuously clocked. The programmer must first enable the USART clock in the Power Management Controller (PMC) before using the USART. However, if the application does not require USART operations, the USART clock can be stopped when not needed and be restarted later. In this case, the USART will resume its operations where it left off.

### 46.5.3 Interrupt Sources

The USART interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the USART interrupt requires the Interrupt Controller to be programmed first.

## 46.6 Functional Description

### 46.6.1 Baud Rate Generator

The baud rate generator provides the bit period clock, also named the baud rate clock, to both the receiver and the transmitter.

The baud rate generator clock source is selected by configuring the USCLKS field in the USART Mode register (US_MR) to one of the following:

- The peripheral clock
- A division of the peripheral clock, where the divider is product-dependent, but generally set to 8
- A processor/peripheral independent clock source fully programmable provided by PMC (PCK)
- The external clock, available on the SCK pin

The baud rate generator is based upon a 16-bit divider, which is configured using the CD field of the Baud Rate Generator register (US_BRGR). If CD is configured to ‘0’, the baud rate generator does not generate any clocks. If CD is configured to ‘1’, the divider is bypassed and becomes inactive.

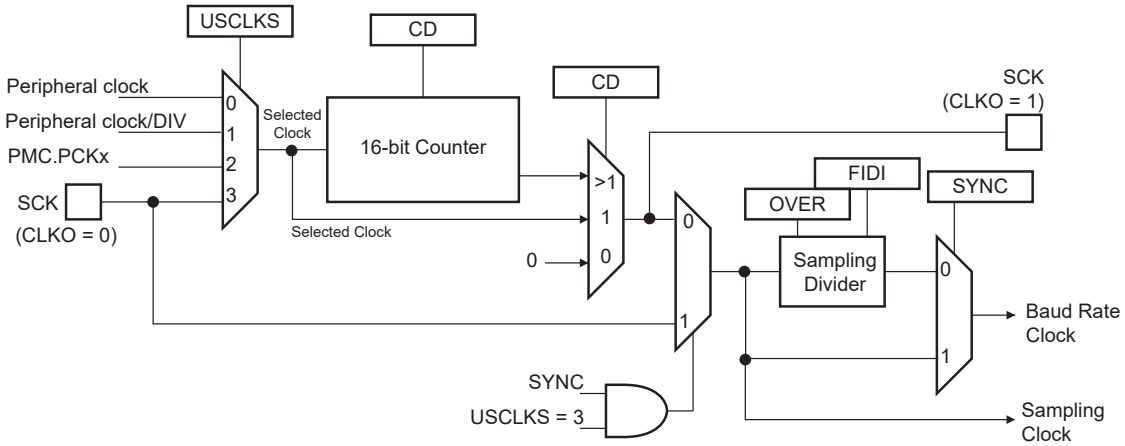
If the external SCK clock is selected, the duration of the low and high levels of the signal provided on the SCK pin must be longer than a peripheral clock period. The frequency of the signal provided on SCK must be at least 3 times lower than the frequency provided on the peripheral clock in USART mode (field USART_MODE differs from 0xE or 0xF), or 6 times lower in SPI mode (field USART_MODE equals 0xE or 0xF).

If PMC PCK is selected, the baud rate is independent of the processor/peripheral clock and thus processor/peripheral clock frequency can be changed without affecting the USART transfer. The PMC PCKx frequency must always be three times lower than the peripheral clock frequency.

If PMC PCK is selected (USCLKS = 2) and the SCK pin is driven (CLKO = 1), the value of US_BRGR.CD must be greater than 1.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 46-2. Baud Rate Generator**




### 46.6.1.1 Baud Rate in Asynchronous Mode

If the USART is programmed to operate in Asynchronous mode, the selected clock is first divided by the value of US_BRGR.CD. The resulting clock is provided to the receiver as a sampling clock and then divided by 16 or 8, depending on the value of US_MR.OVER.

If OVER is set to ‘1’, the receiver sampling is eight times higher than the baud rate clock. If OVER is set to ‘0’, the sampling is performed at 16 times the baud rate clock.

The baud rate is calculated as per the following formula:

$$\text{Baud Rate} = \frac{\text{Selected Clock}}{(8(2 - \text{OVER})\text{CD})}$$

This gives a maximum baud rate of peripheral clock divided by 8, assuming that the peripheral clock is the highest possible clock and that the OVER is written to ‘1’.

#### 46.6.1.1.1 Baud Rate Calculation Example

The following table shows calculations of CD to obtain a baud rate at 38,400 bit/s for different source clock frequencies. This table also shows the actual resulting baud rate and the error.

**Table 46-2. Baud Rate Example (OVER = 0)**

| Source Clock (MHz) | Expected Baud Rate (bit/s) | Calculation Result | CD | Actual Baud Rate (bit/s) | Error |
|---|---|---|---|---|---|
| 3,686,400 | 38,400 | 6.00 | 6 | 38,400.00 | 0.00% |
| 4,915,200 | 38,400 | 8.00 | 8 | 38,400.00 | 0.00% |
| 5,000,000 | 38,400 | 8.14 | 8 | 39,062.50 | 1.70% |
| 7,372,800 | 38,400 | 12.00 | 12 | 38,400.00 | 0.00% |
| 8,000,000 | 38,400 | 13.02 | 13 | 38,461.54 | 0.16% |
| 12,000,000 | 38,400 | 19.53 | 20 | 37,500.00 | 2.40% |
| 12,288,000 | 38,400 | 20.00 | 20 | 38,400.00 | 0.00% |
| 14,318,180 | 38,400 | 23.30 | 23 | 38,908.10 | 1.31% |
| 14,745,600 | 38,400 | 24.00 | 24 | 38,400.00 | 0.00% |
| 18,432,000 | 38,400 | 30.00 | 30 | 38,400.00 | 0.00% |
| 24,000,000 | 38,400 | 39.06 | 39 | 38,461.54 | 0.16% |
| 24,576,000 | 38,400 | 40.00 | 40 | 38,400.00 | 0.00% |

.....continued

| Source Clock (MHz) | Expected Baud Rate (bit/s) | Calculation Result | CD | Actual Baud Rate (bit/s) | Error |
|---|---|---|---|---|---|
| 25,000,000 | 38,400 | 40.69 | 40 | 38,109.76 | 0.76% |
| 32,000,000 | 38,400 | 52.08 | 52 | 38,461.54 | 0.16% |
| 32,768,000 | 38,400 | 53.33 | 53 | 38,641.51 | 0.63% |
| 33,000,000 | 38,400 | 53.71 | 54 | 38,194.44 | 0.54% |
| 40,000,000 | 38,400 | 65.10 | 65 | 38,461.54 | 0.16% |
| 50,000,000 | 38,400 | 81.38 | 81 | 38,580.25 | 0.47% |
| 60,000,000 | 38,400 | 97.66 | 98 | 38,265.31 | 0.35% |
| 70,000,000 | 38,400 | 113.93 | 114 | 38,377.19 | 0.06% |

In this example, the baud rate is calculated with the following formula:

$$\text{Baud Rate} = \text{Selected Clock}/\text{CD} \times 16$$

The baud rate error is calculated with the following formula. It is not recommended to work with an error higher than 5%.

$$\text{Error} = 1 - \left(\frac{\text{Expected Baud Rate}}{\text{Actual Baud Rate}}\right)$$

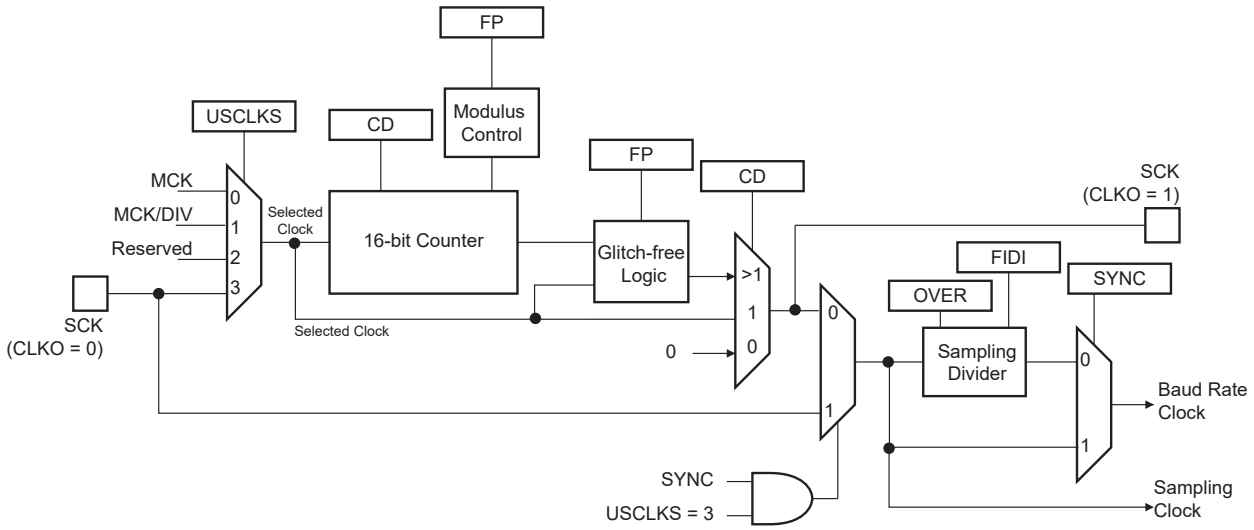
### 46.6.1.2 Fractional Baud Rate in Asynchronous Mode

The baud rate generator is subject to the following limitation: the output frequency changes only by integer multiples of the reference frequency. An approach to this problem is to integrate a fractional N clock generator that has a high resolution. The generator architecture is modified to obtain baud rate changes by a fraction of the reference source clock. This fractional part is programmed using US_BRGR.FP. If FP is not 0, the fractional part is activated. The resolution is one-eighth of the clock divider. The fractional baud rate is calculated using the following formula:

$$\text{Baud Rate} = \frac{\text{Selected Clock}}{\left(8(2 - \text{OVER})\left(\text{CD} + \frac{\text{FP}}{8}\right)\right)}$$

The modified architecture is presented in the following figure.

**Figure 46-3. Fractional Baud Rate Generator**



**WARNING** When the value of US_BRGR.FP is greater than '0', the SCK (oversampling clock) generates non-constant duty cycles. The SCK high duration is increased by “selected clock” period from time to time. The duty cycle depends on the value of USART_BRGR.CD.

#### 46.6.1.3 Baud Rate in Synchronous Mode or SPI Mode

If the USART is programmed to operate in Synchronous mode, the selected clock is divided by the value of US_BRGR.CD.

$$\text{Baud Rate} = \frac{\text{Selected Clock}}{\text{CD}}$$

In Synchronous mode, if the external clock is selected (USCLKS = 3), the clock is provided directly by the signal on the USART SCK pin. No division is active. The value written in US_BRGR has no effect. The external clock frequency must be at least 3 times lower than the system clock. In Master mode, Synchronous mode (USCLKS = 0 or 1, CLKO set to 1), the receive part limits the SCK maximum frequency to Selected Clock/3 in USART mode, or Selected Clock/6 in SPI mode.

When either the external clock SCK or the internal clock divided (peripheral clock/DIV) is selected, the value of CD must be even if the user has to ensure a 50:50 mark/space ratio on the SCK pin. When the peripheral clock is selected, the baud rate generator ensures a 50:50 duty cycle on the SCK pin, even if the value of CD is odd.

#### 46.6.1.4 Baud Rate in ISO 7816 Mode

The ISO7816 specification defines the bit rate with the following formula:

$$B = \frac{\text{Di}}{\text{Fi}} \times f$$

where:

- B is the bit rate
- Di is the bit-rate adjustment factor
- Fi is the clock frequency division factor
- f is the ISO7816 clock frequency (Hz)

Di is a binary value encoded on a 4-bit field, named DI, as represented in Table 46-3.

**Table 46-3. Binary and Decimal Values for Di**

| DI field | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 1000 | 1001 |
|---|---|---|---|---|---|---|---|---|
| Di (decimal) | 1 | 2 | 4 | 8 | 16 | 32 | 12 | 20 |

Fi is a binary value encoded on a 4-bit field, named FI, as represented in Table 46-4.

**Table 46-4. Binary and Decimal Values for Fi**

| FI field | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 1001 | 1010 | 1011 | 1100 | 1101 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Fi (decimal) | 372 | 372 | 558 | 744 | 1116 | 1488 | 1860 | 512 | 768 | 1024 | 1536 | 2048 |

Table 46-5 shows the resulting Fi/Di ratio, which is the ratio between the ISO7816 clock and the baud rate clock.

**Table 46-5. Possible Values for the Fi/Di Ratio**

| Fi/Di | 372 | 558 | 744 | 1116 | 1488 | 1806 | 512 | 768 | 1024 | 1536 | 2048 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 372 | 558 | 744 | 1116 | 1488 | 1860 | 512 | 768 | 1024 | 1536 | 2048 |
| 2 | 186 | 279 | 372 | 558 | 744 | 930 | 256 | 384 | 512 | 768 | 1024 |
| 4 | 93 | 139.5 | 186 | 279 | 372 | 465 | 128 | 192 | 256 | 384 | 512 |
| 8 | 46.5 | 69.75 | 93 | 139.5 | 186 | 232.5 | 64 | 96 | 128 | 192 | 256 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 16 | 23.25 | 34.87 | 46.5 | 69.75 | 93 | 116.2 | 32 | 48 | 64 | 96 | 128 |
| 32 | 11.62 | 17.43 | 23.25 | 34.87 | 46.5 | 58.13 | 16 | 24 | 32 | 48 | 64 |
| 12 | 31 | 46.5 | 62 | 93 | 124 | 155 | 42.66 | 64 | 85.33 | 128 | 170.6 |
| 20 | 18.6 | 27.9 | 37.2 | 55.8 | 74.4 | 93 | 25.6 | 38.4 | 51.2 | 76.8 | 102.4 |

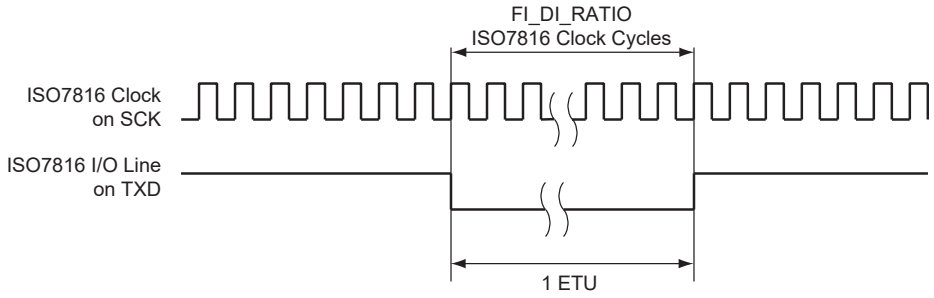
If the USART is configured in ISO7816 mode, the clock selected by US_MR.USCLKS is first divided by the value programmed in US_BRGR.CD. The resulting clock can be provided to the SCK pin to feed the smart card clock inputs. This means that the US_MR.CLKO bit can be written to ‘1’.

This clock is then divided by the value programmed in the FI_DI_RATIO field in the FI DI Ratio register (US_FIDI). This is performed by the Sampling Divider, which performs a division by up to 2047 in ISO7816 mode. The noninteger values of the Fi/Di ratio are not supported and the user must program FI_DI_RATIO to a value as close as possible to the expected value.

FI_DI_RATIO resets to the value 0x174 (372 in decimal) and is the most common divider between the ISO7816 clock and the bit rate (Fi = 372, Di = 1).

The following figure shows the relation between the Elementary Time Unit, corresponding to a bit time, and the ISO 7816 clock.

**Figure 46-4. Elementary Time Unit (ETU)**




### 46.6.2 Receiver and Transmitter Control

After reset, the receiver is disabled. The user must enable the receiver by setting the RXEN bit in the Control register (US_CR). However, the receiver registers can be programmed before the receiver clock is enabled.

After reset, the transmitter is disabled. The user must enable it by writing a ‘1’ to US_CR.TXEN. However, the transmitter registers can be programmed before being enabled.

The receiver and the transmitter can be enabled together or independently.

At any time, the software can perform a reset on the receiver or the transmitter of the USART by writing a ‘1’ to the corresponding bit US_CR.RSTRX and US_CR.RSTTX respectively. The software resets clear the status flag and reset internal state machines but the user interface configuration registers hold the value configured prior to software reset. Regardless of what the receiver or the transmitter is performing, the communication is immediately stopped.

The user can also independently disable the receiver or the transmitter by writing a ‘1’ to US_CR.RXDIS and US_CR.TXDIS, respectively. If the receiver is disabled during a character reception, the USART waits until the end of reception of the current character, then the reception is stopped. If the transmitter is disabled while it is operating, the USART waits the end of transmission of both the current character and character being stored in the Transmit Holding register (US_THR). If a timeguard is programmed, it is handled normally.

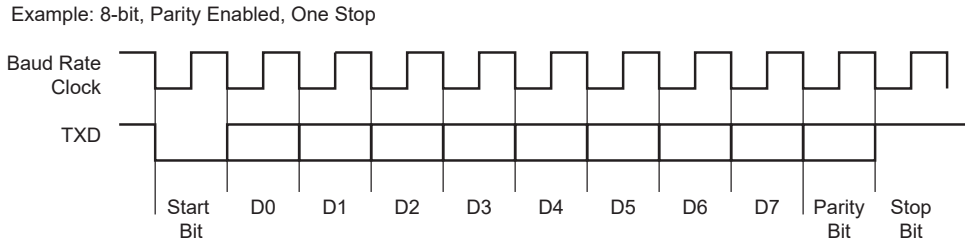
### 46.6.3 Synchronous and Asynchronous Modes

#### 46.6.3.1 Transmitter Operations

The transmitter performs the same in both Synchronous and Asynchronous operating modes (SYNC = 0 or SYNC = 1). One start bit, up to 9 data bits, one optional parity bit and up to two stop bits are successively shifted out on the TXD pin at each falling edge of the programmed serial clock.

The number of data bits is configured in the US_MR.CHRL and the US_MR.MODE9. Nine bits are selected by writing a ‘1’ to US_MR.MODE9 regardless of the CHRL field. The parity is selected by US_MR.PAR. Even, odd, space, marked or none parity bit can be configured. US_MR.MSBF configures which data bit is sent first. If written to ‘1’, the most significant bit is sent first. If written to ‘0’, the less significant bit is sent first. The number of stop bits is selected by US_MR.NBSTOP. The 1.5 stop bit is supported in Asynchronous mode only.

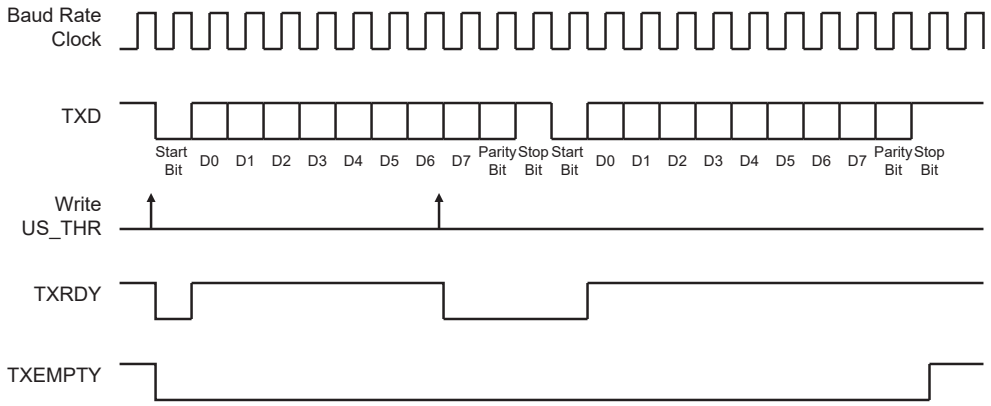
**Figure 46-5. Character Transmit**




The characters are sent by writing in US_THR. The transmitter reports two status bits in the Channel Status register (US_CSR): TXRDY (Transmitter Ready), which indicates that US_THR is empty, and TXEMPTY, which indicates that all the characters written in US_THR have been processed. When the current character processing is completed, the last character written in US_THR is transferred into the Shift register of the transmitter and US_THR becomes empty, thus TXRDY rises.

Both TXRDY and TXEMPTY are low when the transmitter is disabled. Writing a character in US_THR while TXRDY is low has no effect and the written character is lost.

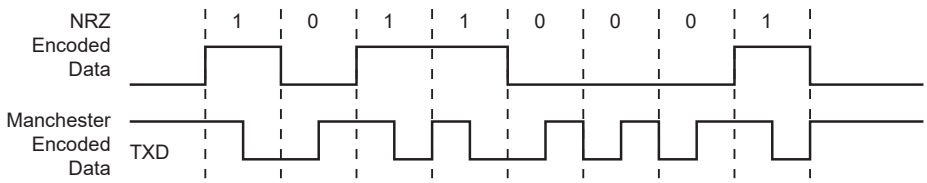
**Figure 46-6. Transmitter Status**




### 46.6.3.2 Manchester Encoder

When the Manchester encoder is in use, characters transmitted through the USART are encoded based on biphase Manchester II format. To enable this mode, write a ‘1’ to USART_MR.MAN. Depending on polarity configuration, a logic level (zero or one), is transmitted as a coded signal one-to-zero or zero-to-one. Thus, a transition always occurs at the midpoint of each bit time. It consumes more bandwidth than the original NRZ signal (2x) but the receiver has more error control since the expected input must show a change at the center of a bit cell. An example of Manchester encoded sequence is: the byte 0xB1 or 10110001 encodes to 10 01 10 10 01 01 01 10, assuming the default polarity of the encoder. Figure 46-7 illustrates this coding scheme.

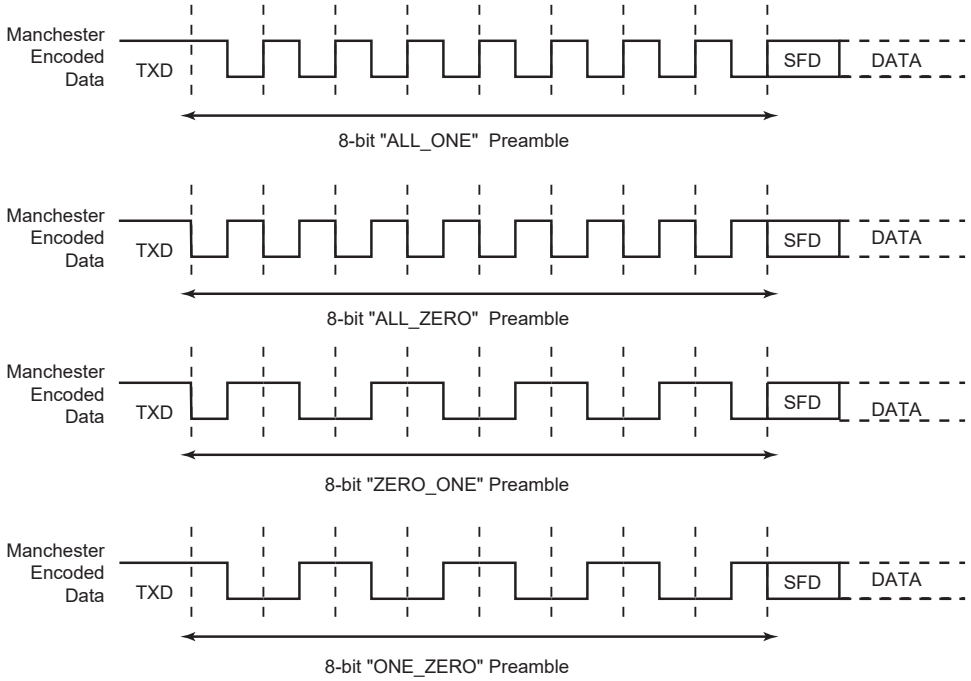
**Figure 46-7. NRZ to Manchester Encoding**




The Manchester encoded character can also be encapsulated by adding both a configurable preamble and a start frame delimiter pattern. Depending on the configuration, the preamble is a training sequence, composed of a predefined pattern with a programmable length from 1 to 15 bit times. If the preamble length is set to '0', the

preamble waveform is not generated prior to any character. The preamble pattern is chosen among the following sequences: ALL_ONE, ALL_ZERO, ONE_ZERO or ZERO_ONE by configuring US_MAN.TX_PP. US_MAN.TX_PL is used to configure the preamble length. Figure 46-8 illustrates and defines the valid patterns. To improve flexibility, the encoding scheme can be configured using US_MAN.TX_MPOL. If TX_MPOL is set to '0' (default), a logic zero is encoded with a zero-to-one transition and a logic one is encoded with a one-to-zero transition. If TX_MPOL is set to '1', a logic one is encoded with a one-to-zero transition and a logic zero is encoded with a zero-to-one transition.

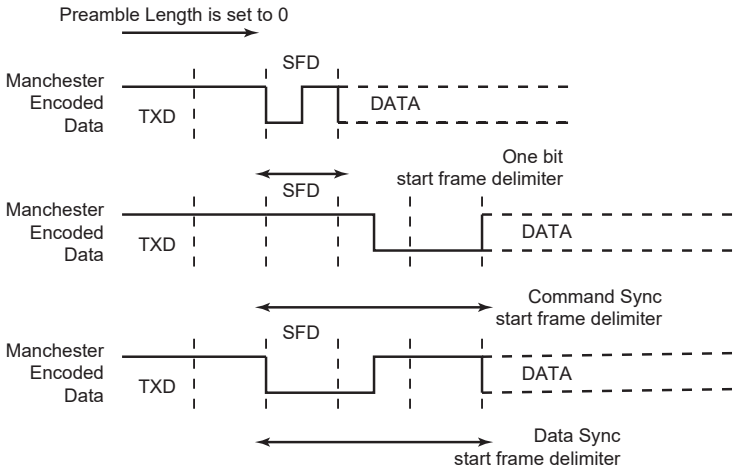
**Figure 46-8. Preamble Patterns, Default Polarity Assumed**




A start frame delimiter is configured using US_MR.ONEBIT. It consists of a user-defined pattern that indicates the beginning of a valid data. Figure 46-9 illustrates these patterns. If the start frame delimiter, also known as the start bit, is one bit, (ONEBIT = 1), a logic zero is Manchester encoded and indicates that a new character is being sent serially on the line. If the start frame delimiter is a synchronization pattern also referred to as sync (ONEBIT = 0), a sequence of three bit times is sent serially on the line to indicate the start of a new character. The sync waveform is in itself an invalid Manchester waveform as the transition occurs at the middle of the second bit time. Two distinct sync patterns are used: the command sync and the data sync. The command sync has a logic one level for one and a half bit times, then a transition to logic zero for the second one and a half bit times. If US_MR.MODSYNC is written to '1', the next character is a command. If it is written to '0', the next character is a data. When direct memory access is used, MODSYNC can be immediately updated with a modified character located in memory. To enable this mode, US_MR.VAR_SYNC must be written to '1'. In this case, MODSYNC is bypassed and the sync configuration is held in US_THR.TXSYNH. The USART character format is modified and includes sync information.

© 2021 Microchip Technology Inc.
and its subsidiaries

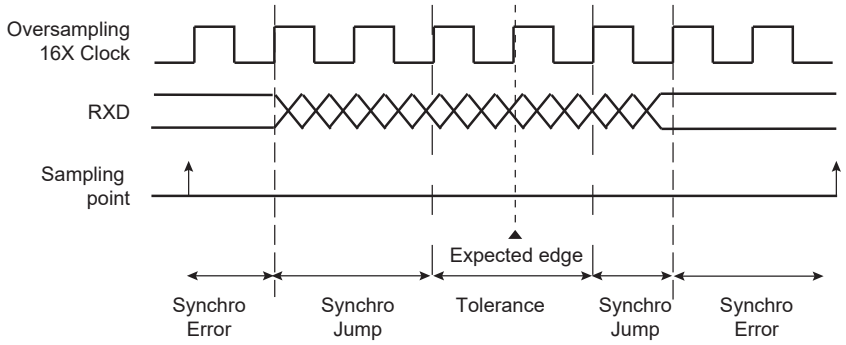
**Figure 46-9. Start Frame Delimiter**




#### 46.6.3.2.1 Drift Compensation

Drift compensation is available only in 16X Oversampling mode. A hardware recovery system allows a larger clock drift. To enable the hardware system, USART_MAN.DRIFT must be written to '1'. If the RXD edge is one 16X clock cycle from the expected edge, this is considered as normal jitter and no corrective action is taken. If the RXD event is between 4 and 2 clock cycles before the expected edge, then the current period is shortened by one clock cycle. If the RXD event is between 2 and 3 clock cycles after the expected edge, then the current period is lengthened by one clock cycle. These intervals are considered to be drift and so corrective actions are automatically taken.

**Figure 46-10. Bit Resynchronization**




### 46.6.3.3 Asynchronous Receiver

If the USART is programmed in Asynchronous operating mode (SYNC = 0), the receiver oversamples the RXD input line. The oversampling is either 16 or 8 times the baud rate clock, depending on the value of US_MR.OVER.

The receiver samples the RXD line. If the line is sampled during one-half of a bit time to 0, a start bit is detected and data, parity and stop bits are successively sampled on the bit rate clock.

If the oversampling is 16 (OVER = 0), a start is detected at the eighth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 16 oversampling clock cycles. If the oversampling is 8 (OVER = 1), a start bit is detected at the fourth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 8 oversampling clock cycles.

The number of data bits, first bit sent and Parity mode are selected by the same fields and bits as the transmitter, i.e., respectively CHRL, MODE9, MSBF and PAR. For the synchronization mechanism only, the number of stop bits has no effect on the receiver as it considers only one stop bit, regardless of the field NBSTOP, so that resynchronization between the receiver and the transmitter can occur. Moreover, as soon as the stop bit is sampled, the receiver starts looking for a new start bit so that resynchronization can also be accomplished when the transmitter is operating with one stop bit.

Figure 46-11 and Figure 46-12 illustrate start detection and character reception when USART operates in Asynchronous mode.

**Figure 46-11. Asynchronous Start Detection**

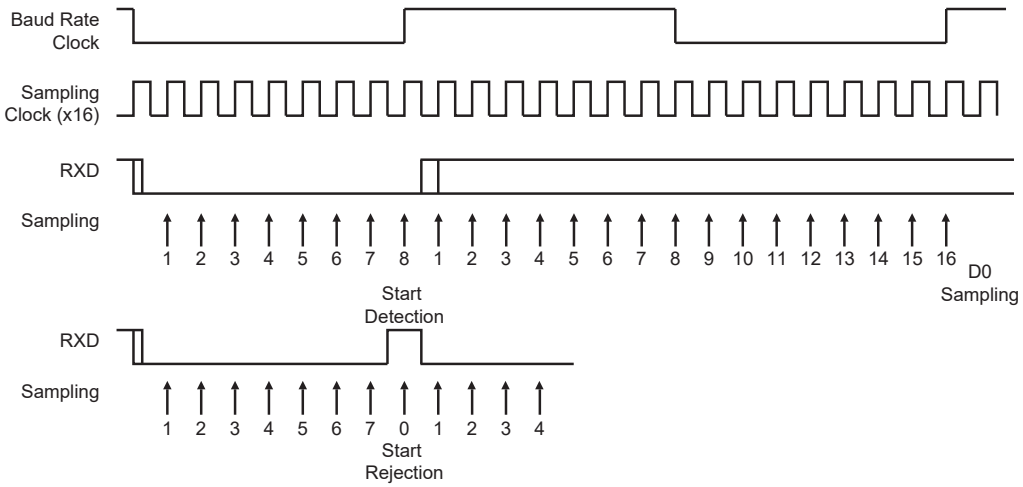
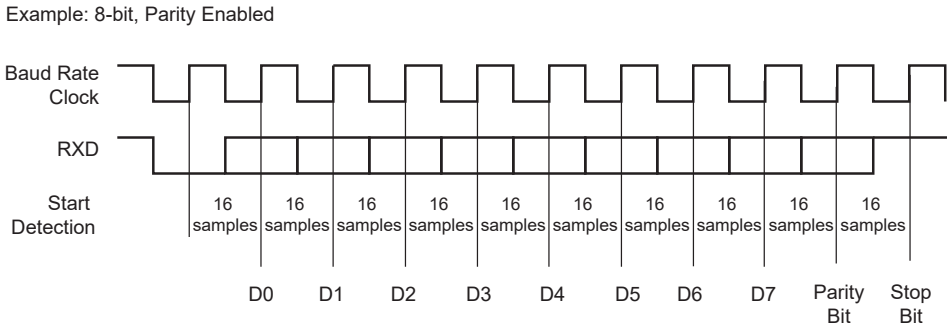



**Figure 46-12. Asynchronous Character Reception**




### 46.6.3.4 Manchester Decoder

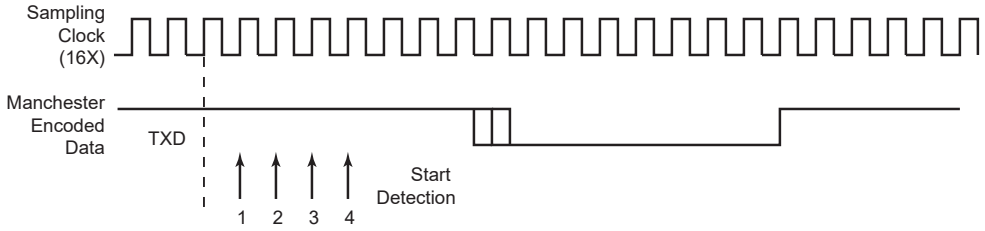
When US_MR.MAN is ‘1’, the Manchester decoder is enabled. The decoder performs both preamble and start frame delimiter detection. One input line is dedicated to Manchester encoded input data.

An optional preamble sequence can be defined, and its length is user-defined and totally independent of the emitter side. The length of the preamble sequence is configured using US_MAN.RX_PL. If RX_PL is ‘0’, no preamble is detected and the function is disabled. The polarity of the input stream is configured with US_MAN.RX_MPOL. Depending on the desired application, the preamble pattern matching is to be defined via the US_MAN. See Figure 46-8 for available preamble patterns.

Unlike preamble, the start frame delimiter is shared between Manchester Encoder and Decoder. If US_MR.ONEBIT is written to ‘1’, only a zero-encoded Manchester can be detected as a valid start frame delimiter. If US_MR.ONEBIT is written to ‘0’, only a sync pattern is detected as a valid start frame delimiter. Decoder operates by detecting transition on incoming stream. If RXD is sampled during one quarter of a bit time to zero, a start bit is detected. See Figure 46-13. The sample pulse rejection mechanism applies.

The US_MAN.RXIDLEV informs the USART of the receiver line idle state value (receiver line inactive). The user must define RXIDLEV to ensure reliable synchronization. By default, RXIDLEV is set to ‘1’ (receiver line is at level 1 when there is no activity).

**Figure 46-13. Asynchronous Start Bit Detection**



The receiver is activated and starts preamble and frame delimiter detection, sampling the data at one quarter and then three quarters. If a valid preamble pattern or start frame delimiter is detected, the receiver continues decoding with the same synchronization. If the stream does not match a valid pattern or a valid start frame delimiter, the receiver resynchronizes on the next valid edge.The minimum time threshold to estimate the bit value is three quarters of a bit time.

If a valid preamble (if used) followed with a valid start frame delimiter is detected, the incoming stream is decoded into NRZ data and passed to the USART for processing. Figure 46-14 illustrates Manchester pattern mismatch. When incoming data stream is passed to the USART, the receiver is also able to detect Manchester code violation. A code violation is a lack of transition in the middle of a bit cell. In this case, the US_CSR.MANERR flag is raised. It is cleared by writing a '1' to US_CR.RSTSTA. See Figure 46-15 for an example of Manchester error detection during data phase.

**Figure 46-14. Preamble Pattern Mismatch**

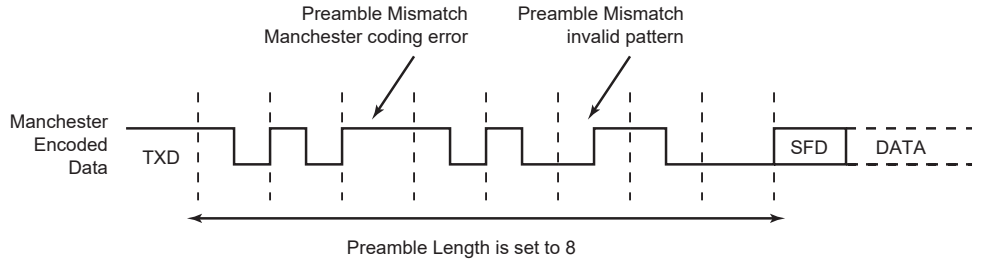
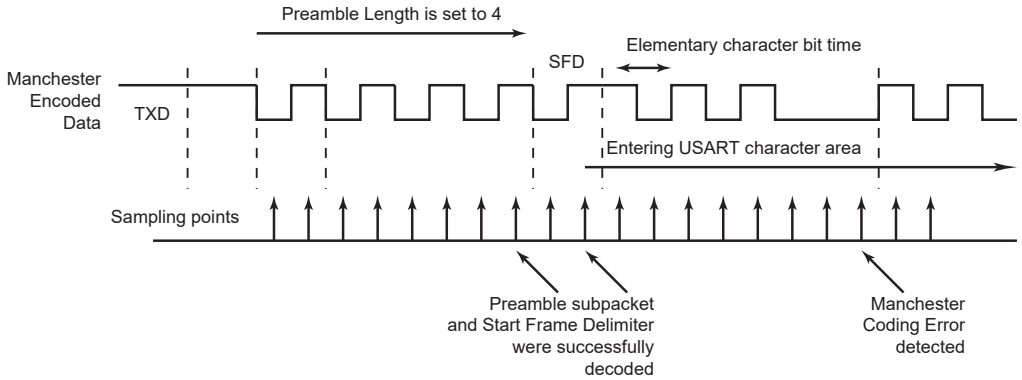



**Figure 46-15. Manchester Error Flag**




When the start frame delimiter is a sync pattern (US_MR.ONEBIT = 0), both command and data delimiter are supported. If a valid sync is detected, the received character is written in RXCHR in the Receive Holding register (US_RHR) and RXSYNH is updated. RXSYNH is set to '1' when the received character is a command, and to '0' if the received character is a data. This alleviates and simplifies the direct memory access as the character contains its own sync field in the same register.

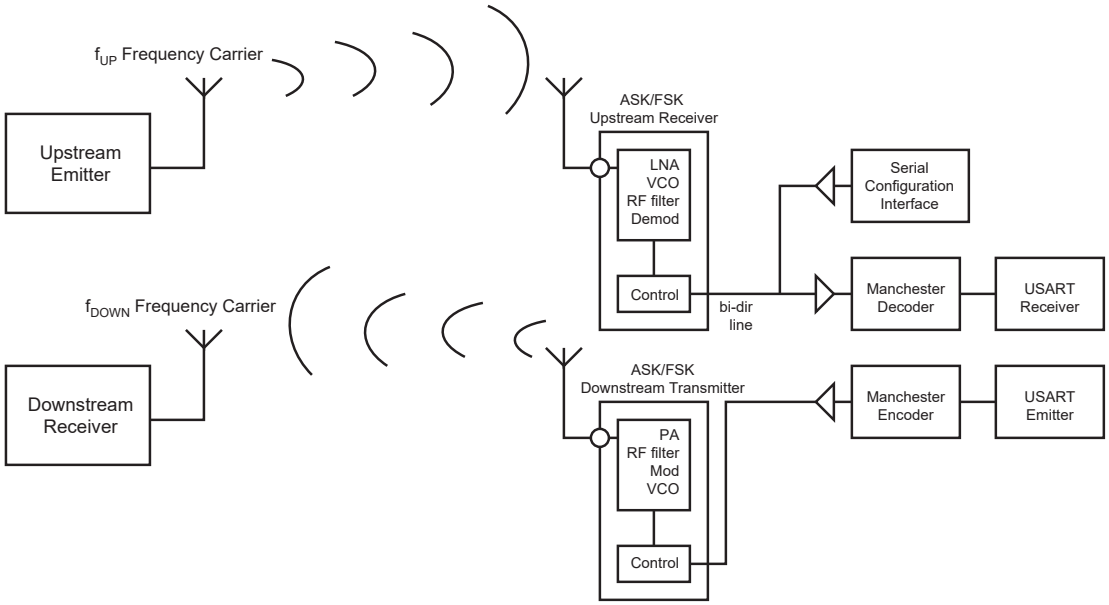
As the decoder is setup to be used in Unipolar mode, the first bit of the frame has to be a zero-to-one transition.

### 46.6.3.5 Radio Interface: Manchester Encoded USART Application

This section describes low data rate RF transmission systems and their integration with a Manchester encoded USART. These systems are based on transmitter and receiver ICs that support ASK and FSK modulation schemes.

The goal is to perform full duplex radio transmission of characters using two different frequency carriers. See the configuration in Figure 46-16.

**Figure 46-16. Manchester Encoded Characters RF Transmission**




The USART peripheral is configured as a Manchester encoder/decoder. Looking at the downstream communication channel, Manchester encoded characters are serially sent to the RF emitter. This may also include a user defined preamble and a start frame delimiter. Mostly, preamble is used in the RF receiver to distinguish between a valid data from a transmitter and signals due to noise. The Manchester stream is then modulated. See Figure 46-17 for an example of ASK modulation scheme. When a logic one is sent to the ASK modulator, the power amplifier, referred to as PA, is enabled and transmits an RF signal at downstream frequency. When a logic zero is transmitted, the RF signal is turned off. If the FSK modulator is activated, two different frequencies are used to transmit data. When a logic one is sent, the modulator outputs an RF signal at frequency F0 and switches to F1 if the data sent is a zero. See Figure 46-18.

From the receiver side, another carrier frequency is used. The RF receiver performs a bit check operation examining demodulated data stream. If a valid pattern is detected, the receiver switches to Receiving mode. The demodulated stream is sent to the Manchester decoder. Because of bit checking inside RF IC, the data transferred to the microcontroller is reduced by a user-defined number of bits. The Manchester preamble length is to be defined in accordance with the RF IC configuration.

**Figure 46-17. ASK Modulator Output**

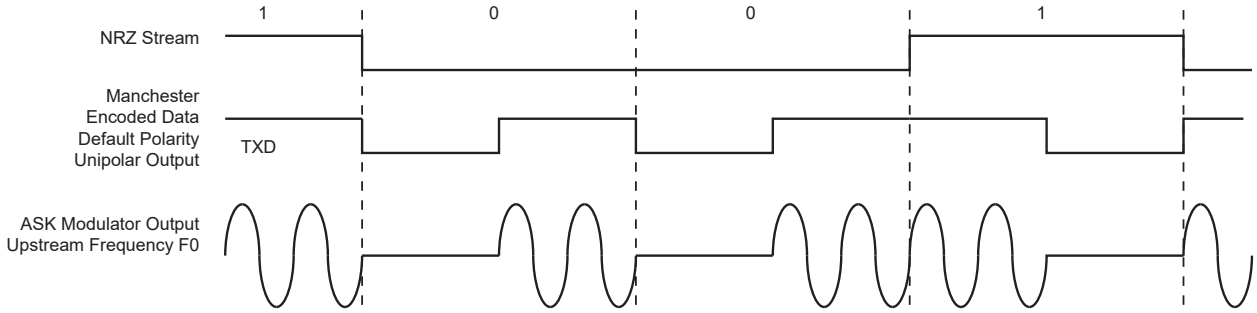
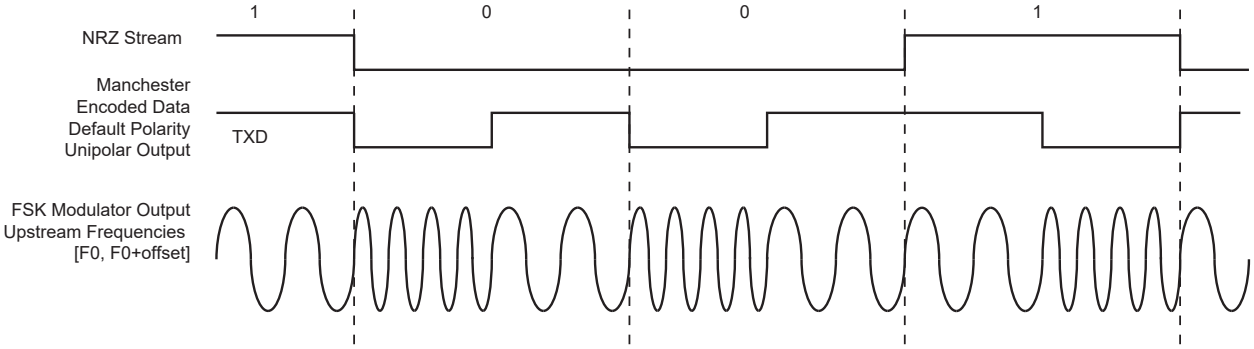


**Figure 46-18. FSK Modulator Output**




### 46.6.3.6 Synchronous Receiver

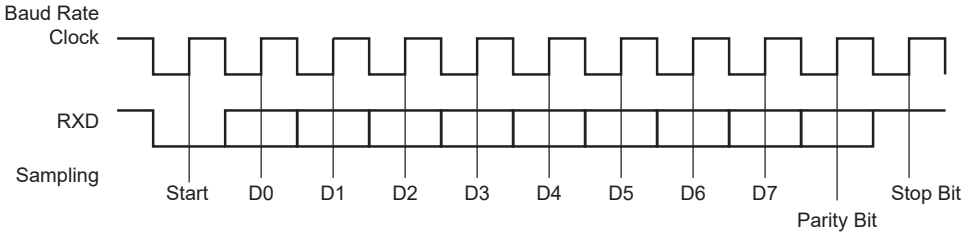
In Synchronous mode (US_MR.SYNC = 1), the receiver samples the RXD signal on each rising edge of the baud rate clock. If a low level is detected, it is considered as a start. All data bits, the parity bit and the stop bits are sampled and the receiver waits for the next start bit. Synchronous mode operations provide a high-speed transfer capability.

Configuration fields and bits are the same as in Asynchronous mode.

The following figure illustrates a character reception in Synchronous mode.

**Figure 46-19. Synchronous Mode Character Reception**

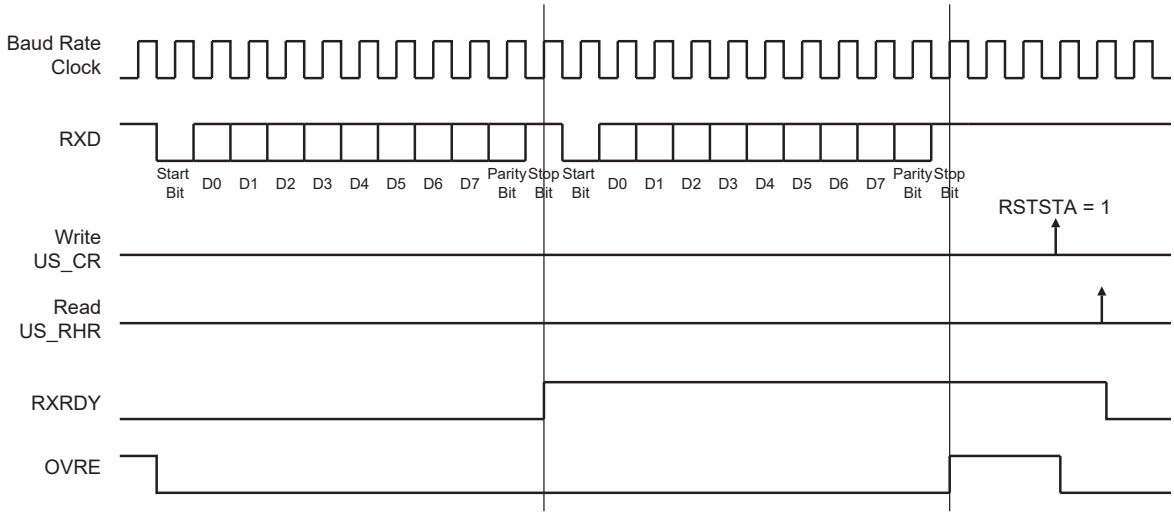
Example: 8-bit, Parity Enabled 1 Stop




### 46.6.3.7 Receiver Operations

When a character reception is completed, it is transferred to the Receive Holding register (US_RHR) and US_CSR.RXRDY rises. If a character is completed while RXRDY is set, the OVRE (Overrun Error) bit is set. The last character is transferred into US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a ‘1’ to US_CR.RSTSTA.

**Figure 46-20. Receiver Status**



### 46.6.3.8 Parity

The USART supports five Parity modes. The PAR field also enables Multidrop mode, see "Multidrop Mode". Even and odd parity bit generation and error detection are supported. The configuration is done in US_MR.PAR.

If even parity is selected, the parity generator of the transmitter drives the parity bit to 0 if a number of 1s in the character data bit is even, and to 1 if the number of 1s is odd. Accordingly, the receiver parity checker counts the number of received 1s and reports a parity error if the sampled parity bit does not correspond. If odd parity is selected, the parity generator of the transmitter drives the parity bit to 1 if a number of 1s in the character data bit is even, and to 0 if the number of 1s is odd. Accordingly, the receiver parity checker counts the number of received 1s and reports a parity error if the sampled parity bit does not correspond. If the mark parity is used, the parity generator of the transmitter drives the parity bit to 1 for all characters. The receiver parity checker reports an error if the parity bit is sampled to 0. If the space parity is used, the parity generator of the transmitter drives the parity bit to 0 for all characters. The receiver parity checker reports an error if the parity bit is sampled to 1. If parity is disabled, the transmitter does not generate any parity bit and the receiver does not report any parity error.

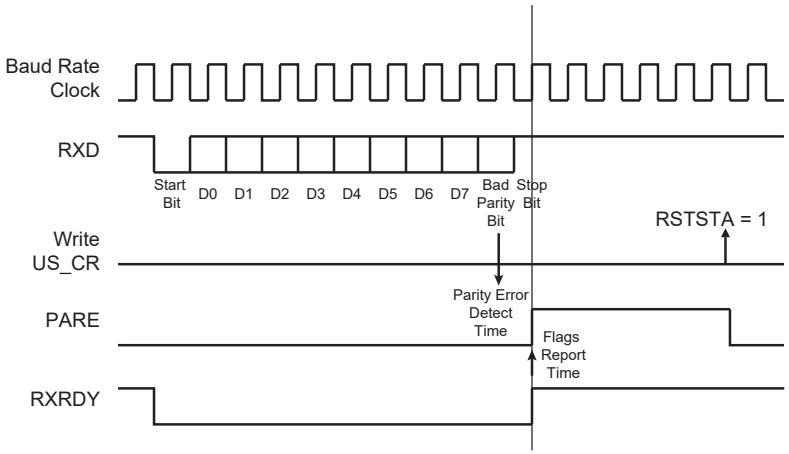
The following table shows an example of the parity bit for the character 0x41 (character ASCII "A") depending on the configuration of the USART. Because there are two bits set to 1 in the character value, the parity bit is set to '1' when the parity is odd, or configured to '0' when the parity is even.

**Table 46-6. Parity Bit Examples**

| Character | Hexadecimal | Binary | Parity Bit | Parity Mode |
|---|---|---|---|---|
| A | 0x41 | 0100 0001 | 1 | Odd |
| A | 0x41 | 0100 0001 | 0 | Even |
| A | 0x41 | 0100 0001 | 1 | Mark |
| A | 0x41 | 0100 0001 | 0 | Space |
| A | 0x41 | 0100 0001 | None | None |

When the receiver detects a parity error, it sets US_CSR.PARE (Parity Error). PARE can be cleared by writing a '1' to the RSTSTA bit the US_CR. The following figure illustrates the parity bit status setting and clearing.

**Figure 46-21. Parity Error**




### 46.6.3.9 Multidrop Mode

If the value 0x6 or 0x07 is written to US_MR.PAR, the USART runs in Multidrop mode. This mode differentiates the data characters and the address characters. Data is transmitted with the parity bit at 0 and addresses are transmitted with the parity bit at 1.

If the USART is configured in Multidrop mode, the receiver sets PARE when the parity bit is high and the transmitter is able to send a character with the parity bit high when a '1' is written to US_CR.SENTA.

To handle parity error, PARE is cleared when a '1' is written to US_CR.RSTSTA.

The transmitter sends an address byte (parity bit set) when US_CR.SENDA = 1. In this case, the next byte written to US_THR is transmitted as an address. Any character written in the US_THR without having written SENDA is transmitted normally with the parity at 0.

### 46.6.3.10 Transmitter Timeguard

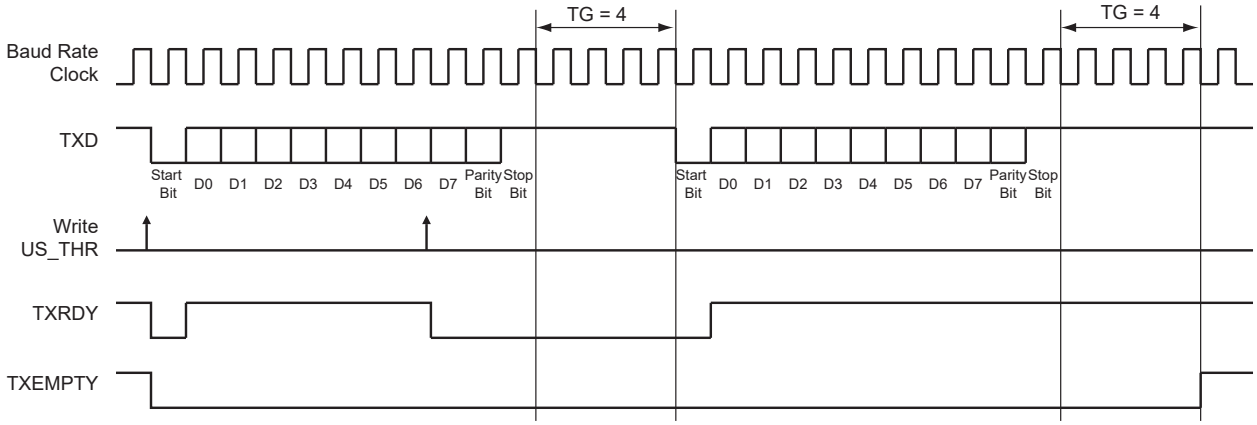
The timeguard feature enables the USART interface with slow remote devices.

The timeguard function enables the transmitter to insert an idle state on the TXD line between two characters. This idle state acts as a long stop bit.

The duration of the idle state is programmed in the TG field of the Transmitter Timeguard register (US_TTGR). When this field is written to '0', no timeguard is generated. Otherwise, the transmitter holds a high level on TXD after each transmitted byte during the number of bit periods programmed in TG in addition to the number of stop bits.

As illustrated in the following figure, the behavior of TXRDY and TXEMPTY status bits is modified by the programming of a timeguard. TXRDY rises only when the start bit of the next character is sent, and thus remains at 0 during the timeguard transmission if a character has been written in US_THR. TXEMPTY remains low until the timeguard transmission is completed as the timeguard is part of the current character being transmitted.

**Figure 46-22. Timeguard Operations**




The following table indicates the maximum length of a timeguard period that the transmitter can handle depending on the baud rate.

**Table 46-7. Maximum Timeguard Length Depending on Baud Rate**

| Baud Rate (bit/s) | Bit Time (µs) | Timeguard (ms) |
|---|---|---|
| 1,200 | 833 | 212.50 |
| 9,600 | 104 | 26.56 |
| 14,400 | 69.4 | 17.71 |
| 19,200 | 52.1 | 13.28 |
| 28,800 | 34.7 | 8.85 |
| 38,400 | 26 | 6.63 |
| 56,000 | 17.9 | 4.55 |
| 57,600 | 17.4 | 4.43 |
| 115,200 | 8.7 | 2.21 |

### 46.6.3.11 Receiver Timeout

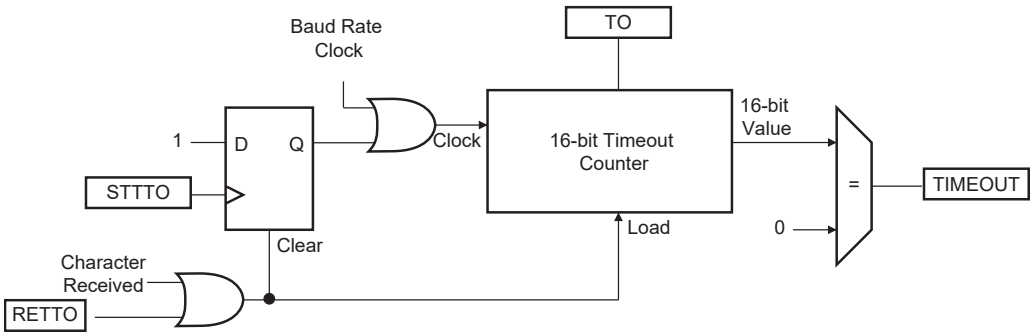
The Receiver Timeout provides support in handling variable-length frames. This feature detects an idle condition on the RXD line. When a timeout is detected, US_CSR.TIMEOUT rises and can generate an interrupt, thus indicating to the driver an end of frame.

The timeout delay period (during which the receiver waits for a new character) is programmed in the TO field of the Receiver Timeout register (US_RTOR). If TO is written to '0', the Receiver Timeout is disabled and no timeout is detected. US_CSR.TIMEOUT remains at '0'. Otherwise, the receiver loads a 16-bit counter with the value programmed in US_RTOR.TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, TIMEOUT rises. Then, the user can either:

- Stop the counter clock until a new character is received. This is performed by writing a '1' to US_CR.STTTO. In this case, the idle state on RXD before a new character is received will not provide a timeout. This prevents having to handle an interrupt before a character is received and allows waiting for the next idle state on RXD after a frame is received.
- Obtain an interrupt while no character is received. This is performed by writing a '1' to the RETTO (Reload and Start Timeout) bit in the US_CR. In this case, the counter starts counting down immediately from the value TO. This generates a periodic interrupt so that a user timeout can be handled, for example when no key is pressed on a keyboard.

The following figure shows the block diagram of the Receiver Timeout feature.

**Figure 46-23. Receiver Timeout Block Diagram**




The following table provides the maximum timeout period for some standard baud rates.

**Table 46-8. Maximum Timeout Period**

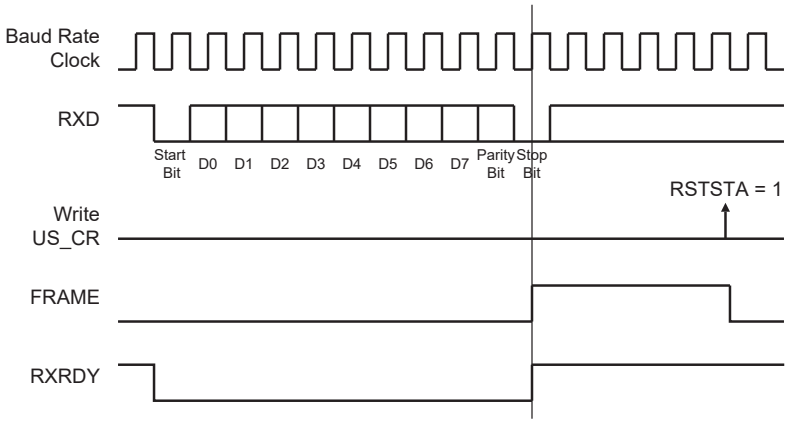
| Baud Rate (bit/s) | Bit Time (µs) | Timeout (ms) |
|---|---|---|
| 600 | 1,667 | 109,225 |
| 1,200 | 833 | 54,613 |
| 2,400 | 417 | 27,306 |
| 4,800 | 208 | 13,653 |
| 9,600 | 104 | 6,827 |
| 14,400 | 69 | 4,551 |
| 19,200 | 52 | 3,413 |
| 28,800 | 35 | 2,276 |
| 38,400 | 26 | 1,704 |
| 56,000 | 18 | 1,170 |
| 57,600 | 17 | 1,138 |
| 200,000 | 5 | 328 |

### 46.6.3.12 Framing Error

The receiver is capable of detecting framing errors. A framing error happens when the stop bit of a received character is detected at level 0. This can occur if the receiver and the transmitter are fully desynchronized.

A framing error is reported in US_CSR.FRAME. FRAME is asserted in the middle of the stop bit as soon as the framing error is detected. It is cleared by writing a '1' to US_CR.RSTSTA.

**Figure 46-24. Framing Error Status**




#### 46.6.3.13 Transmit Break

The user can request the transmitter to generate a break condition on the TXD line. A break condition drives the TXD line low during at least one complete character. It appears the same as a 0x00 character sent with the parity and the stop bits at 0. However, the transmitter holds the TXD line at least during one character until the user requests the break condition to be removed.

A break is transmitted by writing a ‘1’ to US_CR.STTBRK. This can be performed at any time, either while the transmitter is empty (no character in either the Shift register or in US_THR) or when a character is being transmitted. If a break is requested while a character is being shifted out, the character is first completed before the TXD line is held low.

Once STTBRK command is requested, further STTBRK commands are ignored until the end of the break is completed.

The break condition is removed by writing a ‘1’ to US_CR.STPBRK. If the STPBRK is requested before the end of the minimum break duration (one character, including start, data, parity and stop bits), the transmitter ensures that the break condition completes.

The transmitter considers the break as though it is a character, i.e., the STTBRK and STPBRK commands are processed only if US_CSR. TXRDY = 1 and the start of the break condition clears the TXRDY and TXEMPTY bits as if a character is processed.

Writing US_CR with both STTBRK and STPBRK bits to ‘1’ can lead to an unpredictable result. All STPBRK commands requested without a previous STTBRK command are ignored. A byte written into US_THR while a break is pending, but not started, is ignored.

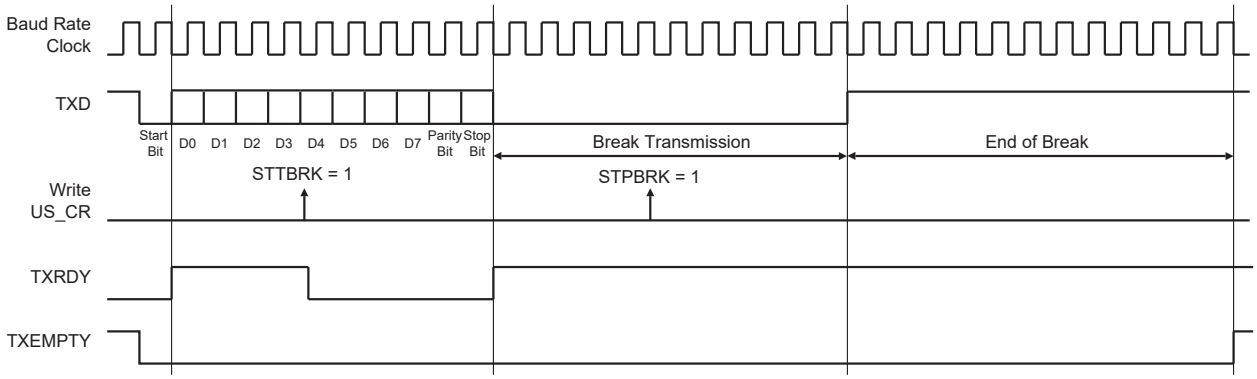
After the break condition, the transmitter returns the TXD line to 1 for a minimum of 12 bit times. Thus, the transmitter ensures that the remote receiver detects correctly the end of break and the start of the next character. If the timeguard is programmed with a value higher than 12, the TXD line is held high for the timeguard period.

After holding the TXD line for this period, the transmitter resumes normal operations.

The following figure illustrates the effect of both the Start Break (STTBRK) and Stop Break (STPBRK) commands on the TXD line.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 46-25. Break Transmission**




### 46.6.3.14 Receive Break

The receiver detects a break condition when all data, parity and stop bits are low. This corresponds to detecting a framing error with data to 0x00, but FRAME remains low.

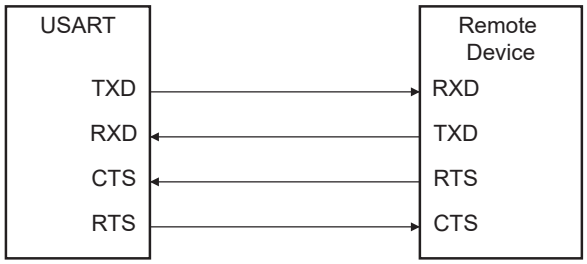
When the low stop bit is detected, the receiver asserts US_CSR.RXBRK. This bit may be cleared by writing a ‘1’ to US_CR.RSTSTA.

An end of receive break is detected by a high level for at least 2/16 of a bit period in Asynchronous operating mode or one sample at high level in Synchronous operating mode. The end of break detection also asserts US_CSR.RXBRK bit.

### 46.6.3.15 Hardware Handshaking

The USART features a hardware handshaking out-of-band flow control. The RTS and CTS pins are used to connect with the remote device, as shown in the following figure.

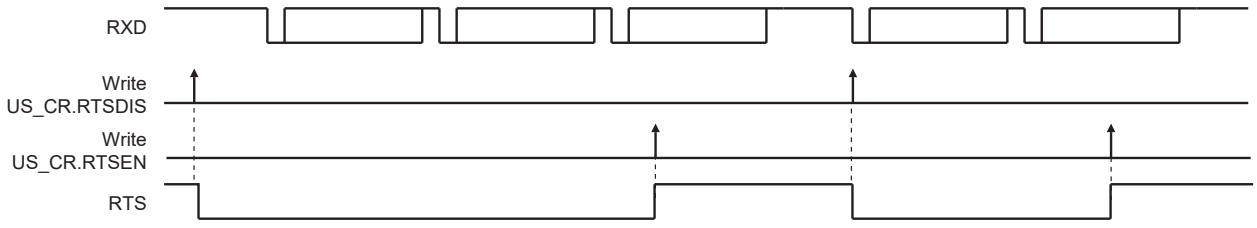
**Figure 46-26. Connection with a Remote Device for Hardware Handshaking**




Setting the USART to operate with hardware handshaking is performed by writing the value 0x2 to US_MR.USART_MODE.

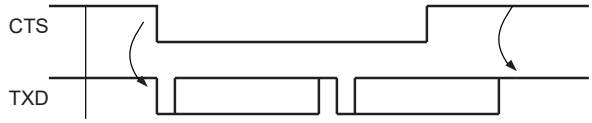
When hardware handshaking is enabled, the USART displays similar behavior as in standard Synchronous or Asynchronous modes, with the difference that the receiver drives the RTS pin and the level on the CTS pin modifies the behavior of the transmitter, as shown in the following figures. The transmitter can handle hardware handshaking in any case.

**Figure 46-27. RTS Line Software Control when US_MR.USART_MODE = 2**




The following figure shows how the transmitter operates if hardware handshaking is enabled. The CTS pin disables the transmitter. If a character is being processed, the transmitter is disabled only after the completion of the current character and transmission of the next character occurs as soon as the pin CTS falls.

**Figure 46-28. Transmitter Behavior when Operating with Hardware Handshaking**




### 46.6.4 ISO7816 Mode

The USART features an ISO7816-compatible operating mode. This mode permits interfacing with smart cards and Security Access Modules (SAM) communicating through an ISO7816 link. Both T = 0 and T = 1 protocols defined by the ISO7816 specification are supported.

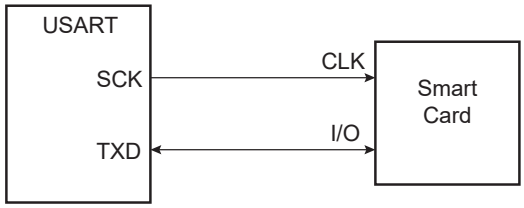
Setting the USART in ISO7816 mode is performed by writing US_MR.USART_MODE to the value 0x4 for protocol T = 0 and to the value 0x6 for protocol T = 1.

#### 46.6.4.1 Overview

The ISO7816 is a half duplex communication on only one bidirectional line. The baud rate is determined by a division of the clock provided to the remote device (see 46.6.1 Baud Rate Generator).

The USART connects to a smart card as shown in the figure below. The TXD line becomes bidirectional and the baud rate generator feeds the ISO7816 clock on the SCK pin. As the TXD pin becomes bidirectional, its output remains driven by the output of the transmitter but only when the transmitter is active while its input is directed to the input of the receiver. The USART is considered as the master of the communication as it generates the clock.

**Figure 46-29. Connection of a Smart Card to the USART**




When operating in ISO7816, either in T = 0 or T = 1 modes, the character format is fixed. The configuration is 8 data bits, even parity and 1 or 2 stop bits, regardless of the values programmed in the Mode register fields CHRL, MODE9, PAR and CHMODE. US_MR.MSBF can be used to transmit LSB or MSB first. US_MR.PAR can be used to transmit in Normal or Inverse mode. Refer to 46.7.3 US_MR and 46.7.3 US_MR.

The USART cannot operate concurrently in both Receiver and Transmitter modes as the communication is unidirectional at a time. It has to be configured according to the required mode by enabling or disabling either the receiver or the transmitter as desired. Enabling both the receiver and the transmitter at the same time in ISO7816 mode may lead to unpredictable results.

The ISO7816 specification defines an inverse transmission format. Data bits of the character must be transmitted on the I/O line at their negative value.

#### 46.6.4.2 Protocol T = 0

In T = 0 protocol, a character is made up of one start bit, eight data bits, one parity bit and one guard time, which lasts two bit times. The transmitter shifts out the bits and does not drive the I/O line during the guard time.

If no parity error is detected, the I/O line remains at 1 during the guard time and the transmitter can continue with the transmission of the next character, as shown in Figure 46-30.

If a parity error is detected by the receiver, it drives the I/O line to 0 during the guard time, as shown in Figure 46-31. This error bit, NACK, for Non Acknowledge. In this case, the character lasts one additional bit time, as the guard time does not change and is added to the error bit time, which lasts one bit time.

When the USART is the receiver and it detects an error, it does not load the erroneous character in US_RHR. It sets US_SR.PARE so that the software can handle the error.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 46-30. T = 0 Protocol without Parity Error**

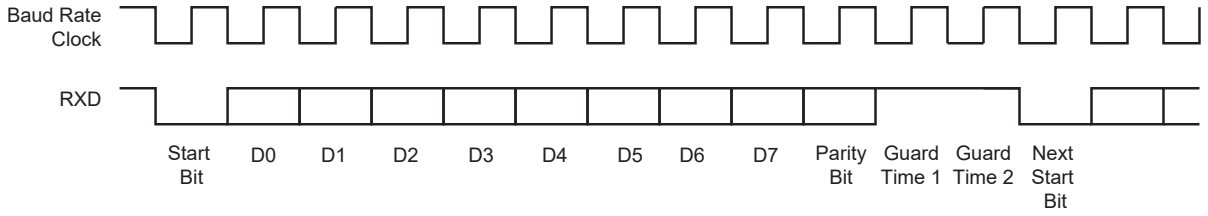
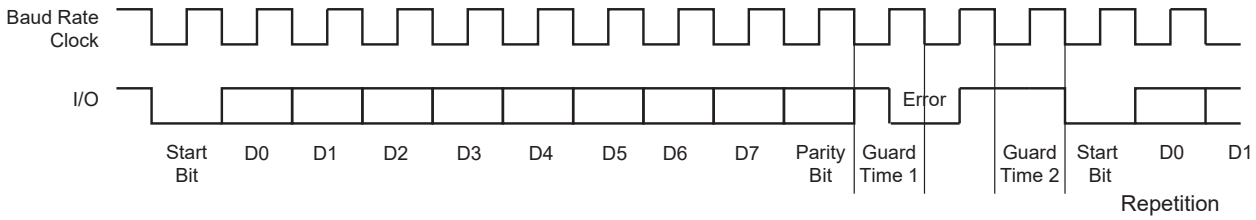



**Figure 46-31. T = 0 Protocol with Parity Error**




#### 46.6.4.2.1 Receive Error Counter

The USART receiver also records the total number of errors. This can be read in the Number of Errors (US_NER) register. The NB_ERRORS field can record up to 255 errors. Reading US_NER automatically clears the NB_ERRORS field.

#### 46.6.4.2.2 Receive NACK Inhibit

The USART can be configured to inhibit an error. This is done by writing a ‘1’ to US_MR.INACK. In this case, no error signal is driven on the I/O line even if a parity bit is detected.

Moreover, if INACK = 1, the erroneous received character is stored in the Receive Holding register as if no error occurred, and the RXRDY bit rises.

#### 46.6.4.2.3 Transmit Character Repetition

When the USART is transmitting a character and gets a NACK, it can automatically repeat the character before moving on to the next one. Repetition is enabled by writing US_MR.MAX_ITERATION to a value greater than 0. Each character can be transmitted up to eight times; the first transmission plus seven repetitions.

If MAX_ITERATION does not equal zero, the USART repeats the character as many times as the value loaded in MAX_ITERATION.

When the USART repetition number reaches MAX_ITERATION and the last repeated character is not acknowledged, the US_CSR.ITER is set. If the repetition of the character is acknowledged by the receiver, the repetitions are stopped and the iteration counter is cleared.

US_CSR.ITER can be cleared by writing a ‘1’ to US_CR.RSTIT.

#### 46.6.4.2.4 Disable Successive Receive NACK

The receiver can limit the number of successive NACKs sent back to the remote transmitter. This is programmed by setting US_MR.DSNACK. The maximum number of NACKs transmitted is configured in US_MR.MAX_ITERATION. As soon as MAX_ITERATION is reached, no error signal is driven on the I/O line and US_CSR.ITER is set.

### 46.6.4.3 Protocol T = 1

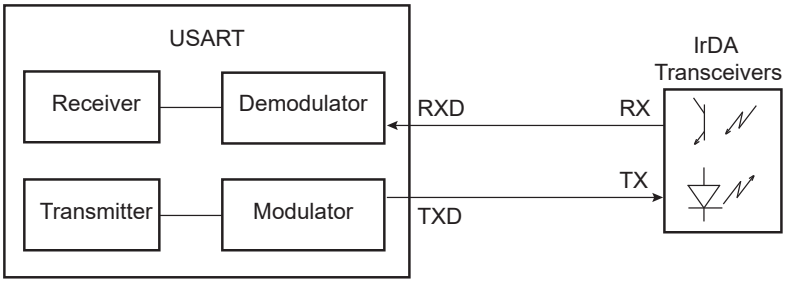
When operating in ISO7816 protocol T = 1, the transmission is similar to an asynchronous format with only one stop bit. The parity is generated when transmitting and checked when receiving. Parity error detection sets US_CSR.PARE.

## 46.6.5 IrDA Mode

The USART features an IrDA mode supplying half-duplex point-to-point wireless communication. It embeds the modulator and demodulator which allows a glueless connection to the infrared transceivers, as shown in the following figure. The modulator and demodulator are compliant with the IrDA specification version 1.1 and support data transfer speeds ranging from 2.4 kbit/s to 115.2 kbit/s.

The IrDA mode is enabled by writing the value 0x8 to US_MR.USART_MODE. The IrDA Filter register (US_IF) is used to configure the demodulator filter. The USART transmitter and receiver operate in a normal Asynchronous mode and all parameters are accessible. Note that the modulator and the demodulator are activated.

**Figure 46-32. Connection to IrDA Transceivers**




The receiver and the transmitter must be enabled or disabled depending on the direction of the transmission to be managed.

To receive IrDA signals, the following needs to be done:

- Disable TX and Enable RX
- Configure the TXD pin as PIO and set it as an output to 0 (to avoid LED emission). Disable the internal pull-up (better for power consumption).
- Receive data

### 46.6.5.1 IrDA Modulation

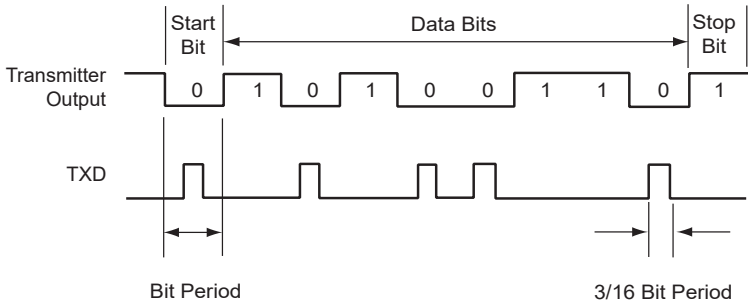
For baud rates up to and including 115.2 kbit/s, the RZI modulation scheme is used. “0” is represented by a light pulse of 3/16th of a bit time. Some examples of signal pulse duration are shown in the following table.

**Table 46-9. IrDA Pulse Duration**

| Baud Rate | Pulse Duration (3/16) |
|---|---|
| 2.4 kbit/s | 78.13 µs |
| 9.6 kbit/s | 19.53 µs |
| 19.2 kbit/s | 9.77 µs |
| 38.4 kbit/s | 4.88 µs |
| 57.6 kbit/s | 3.26 µs |
| 115.2 kbit/s | 1.63 µs |

The following figure shows an example of character transmission.

**Figure 46-33. IrDA Modulation**




### 46.6.5.2 IrDA Baud Rate

The following table provides examples of CD values, baud rate error, and pulse duration. Note that the requirement on the maximum acceptable error of ±1.87% must be met.

**Table 46-10. IrDA Baud Rate Error**

| Peripheral Clock | Baud Rate (bit/s) | CD | Baud Rate Error | Pulse Time (µs) |
|---|---|---|---|---|
| 3,686,400 | 115,200 | 2 | 0.00% | 1.63 |
| 20,000,000 | 115,200 | 11 | 1.38% | 1.63 |
| 32,768,000 | 115,200 | 18 | 1.25% | 1.63 |
| 40,000,000 | 115,200 | 22 | 1.38% | 1.63 |
| 3,686,400 | 57,600 | 4 | 0.00% | 3.26 |
| 20,000,000 | 57,600 | 22 | 1.38% | 3.26 |
| 32,768,000 | 57,600 | 36 | 1.25% | 3.26 |
| 40,000,000 | 57,600 | 43 | 0.93% | 3.26 |
| 3,686,400 | 38,400 | 6 | 0.00% | 4.88 |
| 20,000,000 | 38,400 | 33 | 1.38% | 4.88 |
| 32,768,000 | 38,400 | 53 | 0.63% | 4.88 |
| 40,000,000 | 38,400 | 65 | 0.16% | 4.88 |
| 3,686,400 | 19,200 | 12 | 0.00% | 9.77 |
| 20,000,000 | 19,200 | 65 | 0.16% | 9.77 |
| 32,768,000 | 19,200 | 107 | 0.31% | 9.77 |
| 40,000,000 | 19,200 | 130 | 0.16% | 9.77 |
| 3,686,400 | 9,600 | 24 | 0.00% | 19.53 |
| 20,000,000 | 9,600 | 130 | 0.16% | 19.53 |
| 32,768,000 | 9,600 | 213 | 0.16% | 19.53 |
| 40,000,000 | 9,600 | 260 | 0.16% | 19.53 |
| 3,686,400 | 2,400 | 96 | 0.00% | 78.13 |
| 20,000,000 | 2,400 | 521 | 0.03% | 78.13 |
| 32,768,000 | 2,400 | 853 | 0.04% | 78.13 |

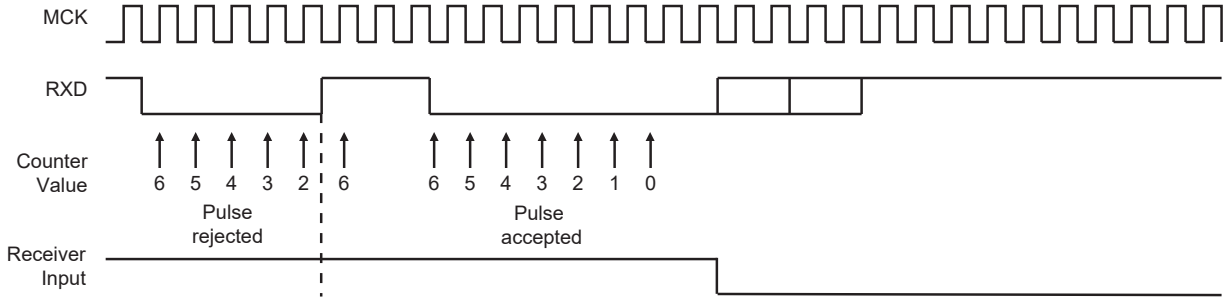
#### 46.6.5.3 IrDA Demodulator

The demodulator is based on the IrDA Receive filter comprised of an 8-bit down counter which is loaded with the value programmed in US_IF. When a falling edge is detected on the RXD pin, the Filter Counter starts counting down at the peripheral clock speed. If a rising edge is detected on the RXD pin, the counter stops and is reloaded with US_IF. If no rising edge is detected when the counter reaches 0, the input of the receiver is driven low during one bit time.

The following figure illustrates the operations of the IrDA demodulator.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 46-34. IrDA Demodulator Operations**




The programmed value in the US_IF register must always meet the following criterion:

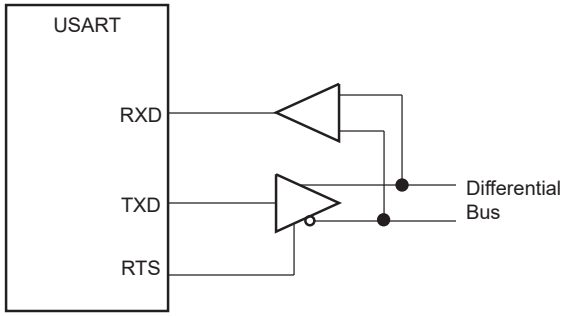
$t_{peripheral\ clock} \times (IRDA_FILTER + 3) < 1.41\ \mu s$

As the IrDA mode uses the same logic as the ISO7816, note that the FI_DI_RATIO field in US_FIDI must be set to a value higher than 0 in order to ensure IrDA communications operate correctly.

### 46.6.6 RS485 Mode

The USART features the RS485 mode to enable line driver control. While operating in RS485 mode, the USART behaves as though in Asynchronous or Synchronous mode and configuration of all the parameters is possible. The difference is that the RTS pin is driven high when the transmitter is operating. The behavior of the RTS pin is controlled by the TXEMPTY bit. A typical connection of the USART to an RS485 bus is shown in Figure 46-35.

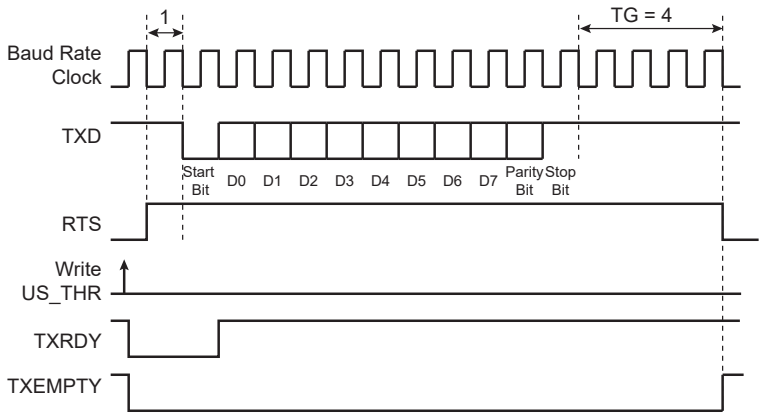
**Figure 46-35. Typical Connection to a RS485 Bus**




RS485 mode is enabled by writing the value 0x1 to the US_MR.USART_MODE.

The RTS pin is at a level inverse to the TXEMPTY bit. Significantly, the RTS pin remains high when a timeguard is programmed so that the line can remain driven after the last character completion. Figure 46-36 gives an example of the RTS waveform during a character transmission when the timeguard is enabled.

**Figure 46-36. Example of RTS Drive with Timeguard**



### 46.6.7 Modem Mode

The USART features the Modem mode, which enables control of the signals DTR (Data Terminal Ready), DSR (Data Set Ready), RTS (Request to Send), CTS (Clear to Send), DCD (Data Carrier Detect), and RI (Ring Indicator). While operating in Modem mode, the USART behaves as a DTE (Data Terminal Equipment) as it drives DTR and RTS and can detect level change on DSR, DCD, CTS, and RI.

Modem mode is enabled by writing the value 0x3 to US_MR.USART_MODE. While operating in Modem mode, the USART behaves as though in Asynchronous mode and all the parameter configurations are available.

The following table provides the correspondence of the USART signals with modem connection standards.

**Table 46-11. Circuit References**

| USART Pin | V24 | CCITT | Direction |
|---|---|---|---|
| TXD | 2 | 103 | From terminal to modem |
| RTS | 4 | 105 | From terminal to modem |
| DTR | 20 | 108.2 | From terminal to modem |
| RXD | 3 | 104 | From modem to terminal |
| CTS | 5 | 106 | From terminal to modem |
| DSR | 6 | 107 | From terminal to modem |
| DCD | 8 | 109 | From terminal to modem |
| RI | 22 | 125 | From terminal to modem |

The control of the DTR output pin is performed by writing a ‘1’ to US_CR.DTRDIS and US_CR.DTREN. The disable command forces the corresponding pin to its inactive level, i.e., high. The enable command forces the corresponding pin to its active level, i.e., low. The RTS output pin is automatically controlled in this mode.

The level changes are detected on the RI, DSR, DCD and CTS pins. If an input change is detected, the bits RIIC, DSRIC, DCDIC and CTSIC in US_CSR are set and can trigger an interrupt. The status is automatically cleared when US_CSR is read. Furthermore, the CTS automatically disables the transmitter when it is detected at its inactive state. If a character is being transmitted when the CTS rises, the character transmission is completed before the transmitter is actually disabled.

### 46.6.8 SPI Mode

The Serial Peripheral Interface (SPI) mode is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master” which controls the data flow, while the other devices act as “slaves" which have data shifted into and out by the master. Different CPUs can take turns being masters and one master may simultaneously shift data into multiple slaves. (Multiple master protocol is the opposite of single master protocol, where one CPU is always the master while all of the others are always slaves.) However, only one slave may drive its output to write data back to the master at any given time.

A slave device is selected when its NSS signal is asserted by the master. The USART in SPI Master mode can address only one SPI slave because it can generate only one NSS signal.

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI): This data line supplies the output data from the master shifted into the input of the slave.
- Master In Slave Out (MISO): This data line supplies the output data from a slave to the input of the master.
- Serial Clock (SCK): This control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of baud rates. The SCK line cycles once for each bit that is transmitted.
- Slave Select (NSS): This control line allows the master to select or deselect the slave.

© 2021 Microchip Technology Inc.
and its subsidiaries

#### 46.6.8.1 Modes of Operation

The USART can operate in SPI Master mode or in SPI Slave mode.

SPI Master mode is enabled by writing 0xE to US_MR.USART_MODE. In this case, the SPI lines must be connected as described below:

- The MOSI line is driven by the output pin TXD
- The MISO line drives the input pin RXD
- The SCK line is driven by the output pin SCK
- The NSS line is driven by the output pin RTS

SPI Slave mode is enabled by writing to 0xF US_MR.USART_MODE. In this case, the SPI lines must be connected as described below:

- The MOSI line drives the input pin RXD
- The MISO line is driven by the output pin TXD
- The SCK line drives the input pin SCK
- The NSS line drives the input pin CTS

In order to avoid unpredictable behavior, any change of the SPI mode must be followed by a software reset of the transmitter and of the receiver (except the initial configuration after a hardware reset). (See Receiver and Transmitter Control).

#### 46.6.8.2 Baud Rate

In SPI mode, the baud rate generator operates in the same way as in USART Synchronous mode. See "Baud Rate in Synchronous Mode or SPI Mode". However, there are some restrictions:

In SPI Master mode:

- The external clock SCK must not be selected (USCLKS ≠ 0x3), and US_MR.CLKO must be written to '1', in order to generate correctly the serial clock on the SCK pin.
- To obtain correct behavior of the receiver and the transmitter, the value programmed in US_BRGR.CD must be greater than or equal to 6.
- If the divided peripheral clock is selected, the value programmed in CD must be even to ensure a 50:50 mark/space ratio on the SCK pin. This value can be odd if the peripheral clock is selected.

In SPI Slave mode:

- The external clock (SCK) selection is forced regardless of the value of the US_MR.USCLKS. Likewise, the value written in US_BRGR has no effect, because the clock is provided directly by the signal on the USART SCK pin.
- To obtain correct behavior of the receiver and the transmitter, the external clock (SCK) frequency must be at least 6 times lower than the system clock.

#### 46.6.8.3 Data Transfer

Up to nine data bits are successively shifted out on the TXD pin at each rising or falling edge (depending on CPOL and CPHA) of the programmed serial clock. There is no Start bit, no Parity bit and no Stop bit.

The number of data bits is selected using US_MR.CHRL and US_MR.MODE9. The nine bits are selected by setting the MODE9 bit regardless of the CHRL field. The MSB data bit is always sent first in SPI mode (Master or Slave).

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed using US_MR.CPOL. The clock phase is programmed using US_MR.CPHA. These two parameters determine the edges of the clock signal upon which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are used and fixed in different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

**Table 46-12. SPI Bus Protocol Mode**

| SPI Bus Protocol Mode | CPOL | CPHA |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |

| .....continued | | |
|---|---|---|
| **SPI Bus Protocol Mode** | **CPOL** | **CPHA** |
| 2 | 1 | 1 |
| 3 | 1 | 0 |

**Figure 46-37. SPI Transfer Format (CPHA = 1, 8 bits per transfer)**

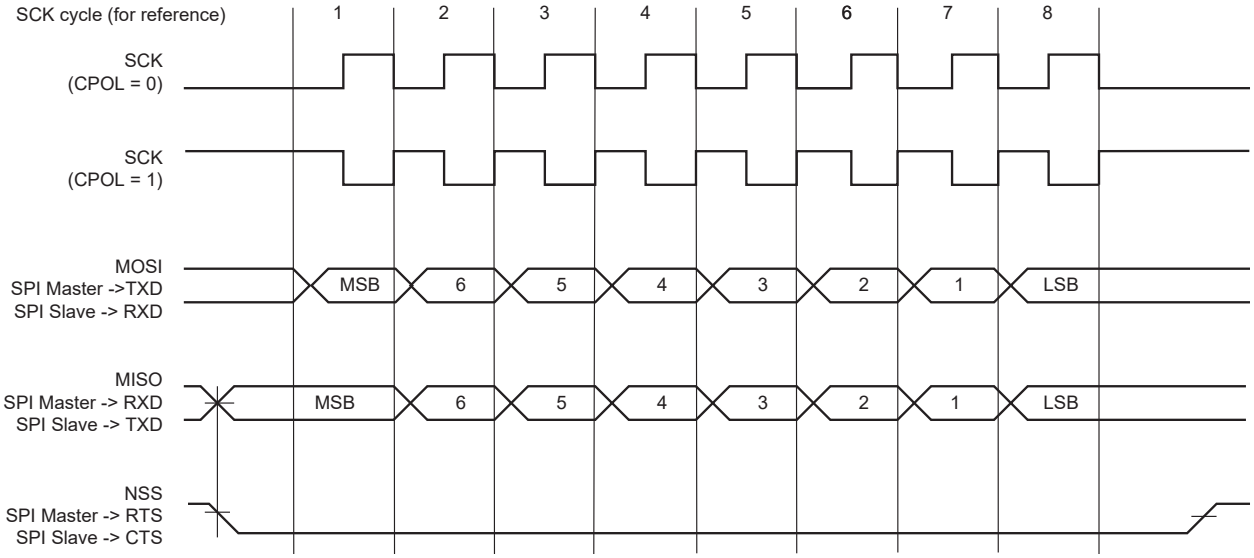
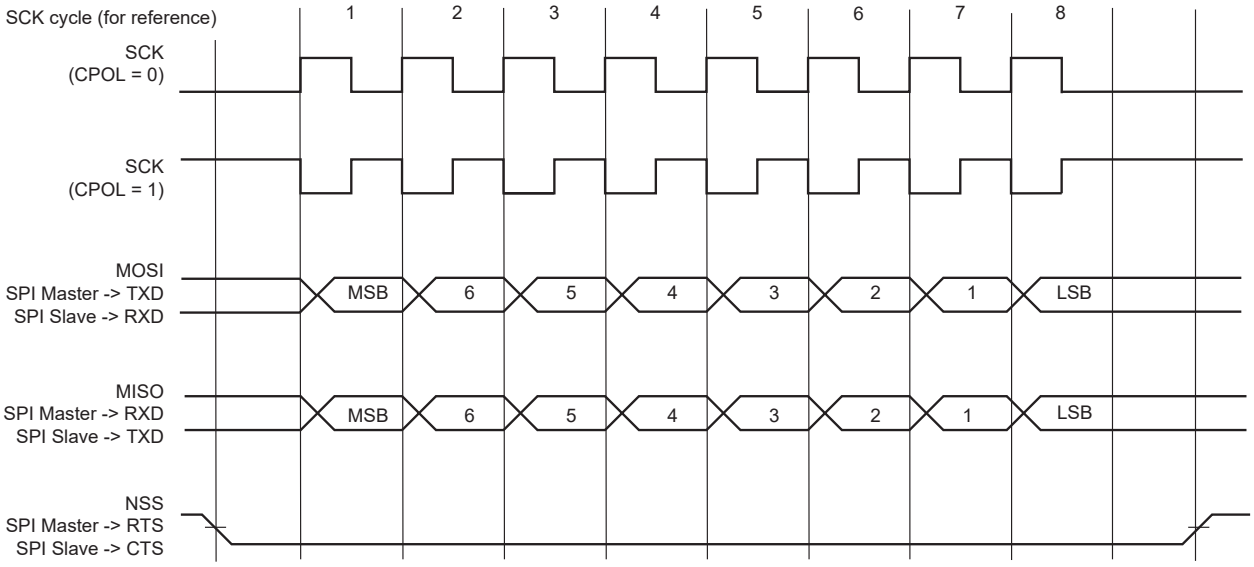



**Figure 46-38. SPI Transfer Format (CPHA = 0, 8 bits per transfer)**




### 46.6.8.4 Receiver and Transmitter Control

See “Receiver and Transmitter Control”.

### 46.6.8.5 Character Transmission

The characters are sent by writing in the US_THR. An additional condition for transmitting a character can be added when the USART is configured in SPI Master mode. In the USART_MR (SPI_MODE), the value of WRDBT can prevent any character transmission (even if US_THR has been written) while the receiver side is not ready (character not read). When WRDBT equals ‘0’, the character is transmitted whatever the receiver status. If WRDBT is set to ‘1’, the transmitter waits for US_RHR to be read before transmitting the character (RXRDY flag cleared), thus preventing any overflow (character loss) on the receiver side.

The chip select line is deasserted for a period equivalent to three bits between the transmission of two data.

The transmitter reports two status bits in US_CSR: TXRDY (Transmitter Ready), which indicates that US_THR is empty and TXEMPTY, which indicates that all the characters written in US_THR have been processed. When the current character processing is completed, the last character written in US_THR is transferred into the Shift register of the transmitter and US_THR becomes empty, thus TXRDY rises.

Both TXRDY and TXEMPTY bits are low when the transmitter is disabled. Writing a character in US_THR while TXRDY is low has no effect and the written character is lost.

If the USART is in SPI Slave mode and if a character must be sent while the US_THR is empty, the UNRE (Underrun Error) bit is set. The TXD transmission line stays at high level during all this time. The UNRE bit is cleared by writing a 1 to the RSTSTA (Reset Status) bit in US_CR.

In SPI Master mode, the slave select line (NSS) is asserted at low level one $t_{bit}$ ($t_{bit}$ being the nominal time required to transmit a bit) before the transmission of the MSB bit and released at high level one $t_{bit}$ after the transmission of the LSB bit. So, the slave select line (NSS) is always released between each character transmission and a minimum delay of three $t_{bit}$ always inserted. However, in order to address slave devices supporting the CSAAT mode (Chip Select Active After Transfer), the slave select line (NSS) can be forced at low level by writing a 1 to the RCS bit in the US_CR. The slave select line (NSS) can be released at high level only by writing a ‘1’ to US_CR.FCS (for example, when all data have been transferred to the slave device).

In SPI Slave mode, the transmitter does not require a falling edge of the slave select line (NSS) to initiate a character transmission but only a low level. However, this low level must be present on the slave select line (NSS) at least one $t_{bit}$ before the first serial clock cycle corresponding to the MSB bit.

#### 46.6.8.6 Character Reception

When a character reception is completed, it is transferred to US_RHR and US_CSR.RXRDY rises. If a character is completed while RXRDY is set, the OVRE (Overrun Error) bit is set. The last character is transferred into US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a ‘1’ to US_CR.RSTSTA.

To ensure correct behavior of the receiver in SPI Slave mode, the master device sending the frame must ensure a minimum delay of one $t_{bit}$ between each character transmission. The receiver does not require a falling edge of the slave select line (NSS) to initiate a character reception but only a low level. However, this low level must be present on the slave select line (NSS) at least one $t_{bit}$ before the first serial clock cycle corresponding to the MSB bit.

#### 46.6.8.7 Receiver Timeout

Because the receiver baud rate clock is active only during data transfers in SPI mode, a receiver timeout is impossible in this mode, whatever the value is in US_RTOR.TO.

### 46.6.9 LIN Mode

The LIN mode provides master node and slave node connectivity on a LIN bus.

The LIN (Local Interconnect Network) is a serial communication protocol which efficiently supports the control of mechatronic nodes in distributed automotive applications.

The main properties of the LIN bus are:

- Single master/multiple slaves concept
- Low-cost silicon implementation based on common UART/SCI interface hardware, an equivalent in software, or as a pure state machine.
- Self synchronization without quartz or ceramic resonator in the slave nodes
- Deterministic signal transmission
- Low cost single-wire implementation
- Speed up to 20 kbit/s

LIN provides cost efficient bus communication where the bandwidth and versatility of CAN are not required.

The LIN mode enables processing LIN frames with a minimum of action from the microprocessor.

#### 46.6.9.1 Modes of Operation

The USART can act either as a LIN master node or as a LIN slave node.

The node configuration is chosen by setting USART_MR.USART_MODE:

- LIN master node (USART_MODE = 0xA)
- LIN slave node (USART_MODE = 0xB)

In order to avoid unpredictable behavior, any change of the LIN node configuration must be followed by a software reset of the transmitter and of the receiver (except the initial node configuration after a hardware reset). (See "Receiver and Transmitter Control".)

#### 46.6.9.2 Baud Rate Configuration

See "Baud Rate in Asynchronous Mode"

- LIN master node: The baud rate is configured in US_BRGR.
- LIN slave node: The initial baud rate is configured in US_BRGR. This configuration is automatically copied in the LIN Baud Rate register (US_LINBRR) when writing US_BRGR. After the synchronization procedure, the baud rate is updated in US_LINBRR.

#### 46.6.9.3 Receiver and Transmitter Control

See "Receiver and Transmitter Control"

#### 46.6.9.4 Character Transmission

See "Transmitter Operations".

#### 46.6.9.5 Character Reception

See "Receiver Operations".

#### 46.6.9.6 Header Transmission (Master Node Configuration)

All the LIN frames start with a header which is sent by the master node and consists of a Synch Break Field, Synch Field and Identifier Field.

So in master node configuration, the frame handling starts with the sending of the header.

The header is transmitted as soon as the identifier is written in the LIN Identifier register (US_LINIR). At this moment the flag TXRDY falls.

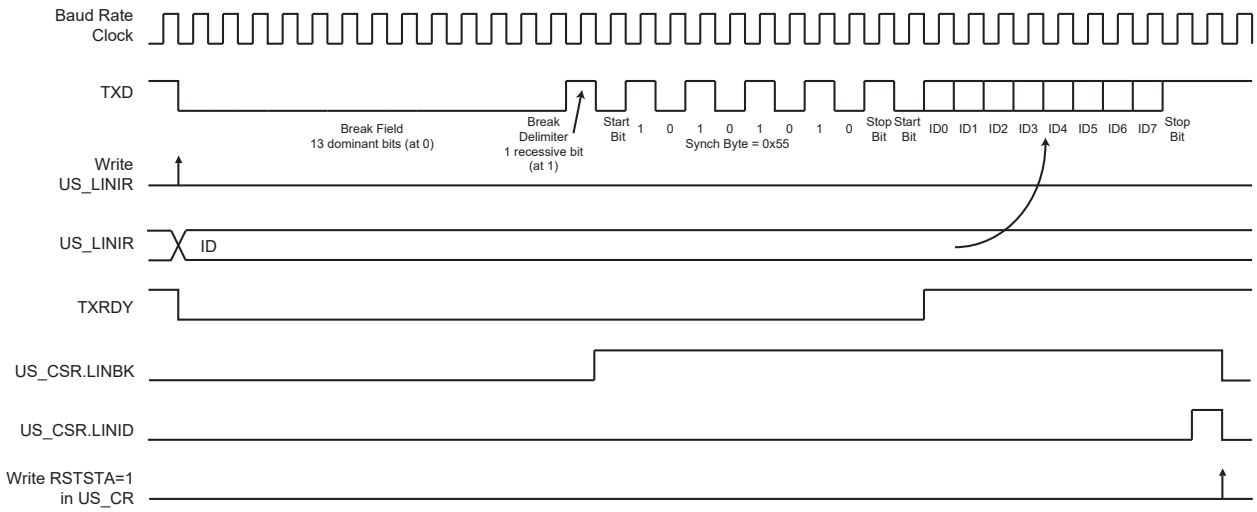
The Break Field, the Synch Field and the Identifier Field are sent automatically one after the other.

The Break Field consists of 13 dominant bits and 1 recessive bit, the Synch Field is the character 0x55 and the Identifier corresponds to the character written in the LIN Identifier register (US_LINIR). The Identifier parity bits can be automatically computed and sent (see "Identifier Parity").

The flag TXRDY rises when the identifier character is transferred into the Shift register of the transmitter.

As soon as the Synch Break Field is transmitted, US_CSR.LINBK is set to '1'. Likewise, as soon as the Identifier Field is sent, US_CSR.LINID is set to '1'. These flags are reset by writing a '1' to US_CR.RSTSTA.

**Figure 46-39. Header Transmission**



### 46.6.9.7 Header Reception (Slave Node Configuration)

All the LIN frames start with a header which is sent by the master node and consists of a Synch Break Field, Synch Field and Identifier Field.

In slave node configuration, the frame handling starts with the reception of the header.

The USART uses a break detection threshold of 11 nominal bit times at the actual baud rate. At any time, if 11 consecutive recessive bits are detected on the bus, the USART detects a Break Field. As long as a Break Field has not been detected, the USART stays idle and the received data are not taken in account.

When a Break Field has been detected, US_CSR.LINBK is set to '1' and the USART expects the Synch Field character to be 0x55. This field is used to update the actual baud rate in order to stay synchronized (see "Slave Node Synchronization"). If the received Synch character is not 0x55, an Inconsistent Synch Field error is generated (see "LIN Errors").

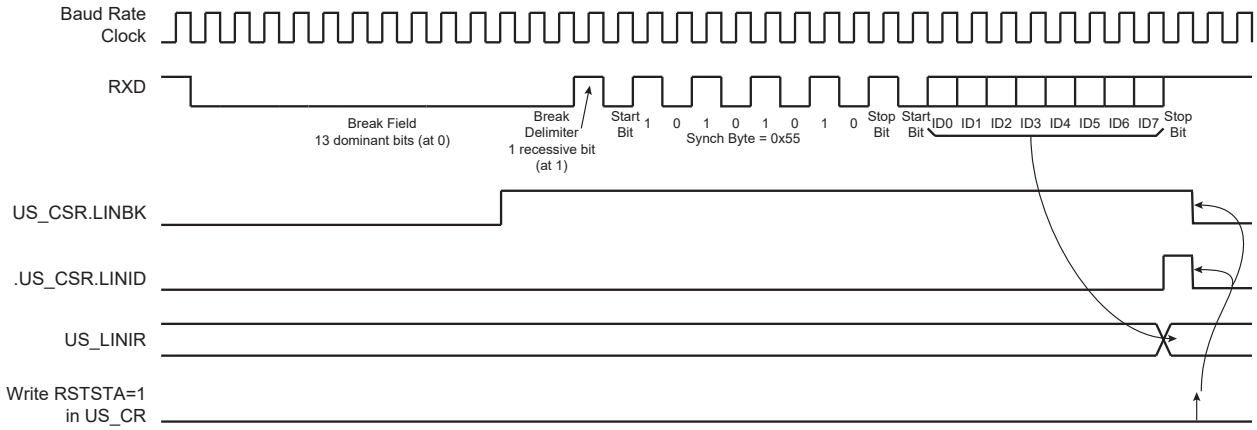
After receiving the Synch Field, the USART expects to receive the Identifier Field.

When the Identifier Field has been received, US_CSR.LINID is set to '1'. At this moment, US_LINIR.IDCHR is updated with the received character. The Identifier parity bits can be automatically computed and checked (see "Identifier Parity").

If the Header is not entirely received within the time given by the maximum length of the header $t_{Header_Maximum}$, the error flag US_CSR.LINHTE is set to '1'.

The flag bits LINID, LINBK and LINHTE are reset by writing a '1' to US_CR.RSTSTA.

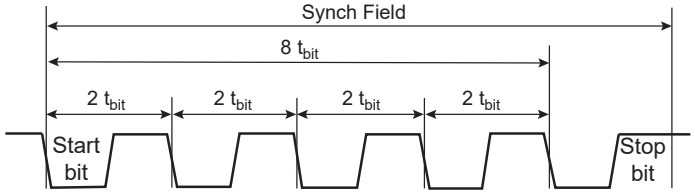
**Figure 46-40. Header Reception**




### 46.6.9.8 Slave Node Synchronization

The synchronization is done only in slave node configuration. The procedure is based on time measurement between falling edges of the Synch Field. The falling edges are available in distances of 2, 4, 6 and 8 bit times.

**Figure 46-41. Synch Field**




The time measurement is made by a 19-bit counter clocked by the sampling clock (see "Baud Rate Generator").

When the start bit of the Synch Field is detected, the counter is reset. Then during the next eight $t_{bit}$ of the Synch Field, the counter is incremented. At the end of these eight $t_{bit}$, the counter is stopped. At this moment, the 16 most significant bits of the counter (value divided by 8) give the new clock divider (LINCD) and the three least significant bits of this value (the remainder) give the new fractional part (LINFP).

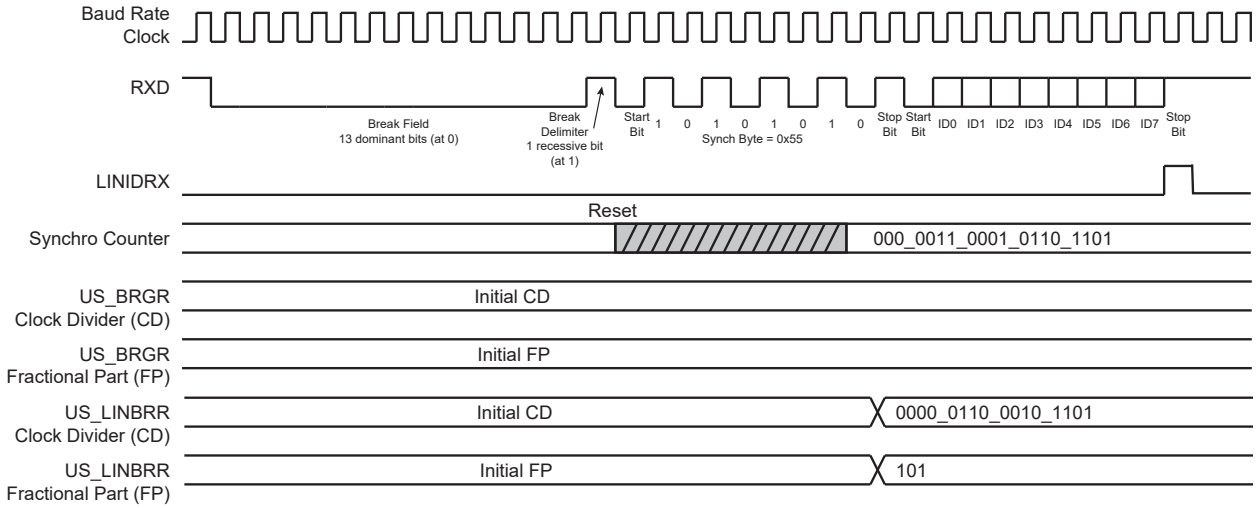
Once the Synch Field has been entirely received, the clock divider (LINCD) and the fractional part (LINFP) are updated in the LIN Baud Rate register (US_LINBRR) with the computed values, if the Synchronization is not disabled by the SYNCDIS bit in the LIN Mode register (US_LINMR).

After reception of the Synch Field:

- If it appears that the computed baud rate deviation compared to the initial baud rate is superior to the maximum tolerance FTol_Unsynch (±15%), then the clock divider (LINCD) and the fractional part (LINFP) are not updated, and the error flag US_CSR.LINSTE is set to ‘1’.
- If it appears that the sampled Synch character is not equal to 0x55, then the clock divider (LINCD) and the fractional part (LINFP) are not updated, and the error flag US_CSR.LINISFE is set to ‘1’.

Flags LINSTE and LINISFE are reset by writing US_CR.RSTSTA to ‘1’.

**Figure 46-42. Slave Node Synchronization**




The accuracy of the synchronization depends on several parameters:

- Nominal clock frequency ($f_{Nom}$) (the theoretical slave node clock frequency)
- Baud Rate
- Oversampling (OVER = 0 => 16X or OVER = 1 => 8X)

The following formula is used to compute the deviation of the slave bit rate relative to the master bit rate after synchronization ($f_{SLAVE}$ is the real slave node clock frequency):

$$\text{Baud rate deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{OVER}) + \beta] \times \text{Baud rate}}{8 \times f_{SLAVE}}\right)\%$$

$$\text{Baud rate deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{OVER}) + \beta] \times \text{Baud rate}}{8 \times \left(\frac{f_{TOL_UNSYNCH}}{100}\right) \times f_{Nom}}\right)\%$$

$-0.5 \le \alpha \le +0.5 \quad -1 < \beta < +1$

$f_{TOL_UNSYNCH}$ is the deviation of the real slave node clock from the nominal clock frequency. The LIN Standard imposes that it must not exceed ±15%. The LIN Standard imposes also that for communication between two nodes, their bit rate must not differ by more than ±2%. This means that the baud rate deviation must not exceed ±1%.

It follows from that, a minimum value for the nominal clock frequency:

$$f_{Nom}(\text{min}) = \left(100 \times \frac{[0.5 \times 8 \times (2 - \text{OVER}) + 1] \times \text{Baud rate}}{8 \times \left(\frac{-15}{100} + 1\right) \times 1\%}\right)\text{Hz}$$

Examples:

- Baud rate = 20 kbit/s, OVER = 0 (Oversampling 16X) => $f_{Nom}$(min) = 2.64 MHz

- Baud rate = 20 kbit/s, OVER = 1 (Oversampling 8X) => $f_{Nom}$(min) = 1.47 MHz
- Baud rate = 1 kbit/s, OVER = 0 (Oversampling 16X) => $f_{Nom}$(min) = 132 kHz
- Baud rate = 1 kbit/s, OVER = 1 (Oversampling 8X) => $f_{Nom}$(min) = 74 kHz

#### 46.6.9.9 Identifier Parity

A protected identifier consists of two subfields: the identifier and the identifier parity. Bits 0 to 5 are assigned to the identifier and bits 6 and 7 are assigned to the parity.

The USART interface can generate/check these parity bits, but this feature can also be disabled. The user can choose between two modes using US_LINMR.PARDIS:

- PARDIS = 0:
  - During header transmission, the parity bits are computed and sent with the six least significant bits of US_LINIR.IDCHR. The bits 6 and 7 of this register are discarded.
  - During header reception, the parity bits of the identifier are checked. If the parity bits are wrong, an Identifier Parity error occurs (see Parity). Only the six least significant bits of the IDCHR field are updated with the received Identifier. The bits 6 and 7 are stuck to 0.
- PARDIS = 1:
  - During header transmission, all the bits of US_LINIR.IDCHR are sent on the bus.
  - During header reception, all the bits of IDCHR are updated with the received Identifier.

#### 46.6.9.10 Node Action

Depending on the identifier, the node is affected – or not – by the LIN response. Consequently, after sending or receiving the identifier, the USART must be configured. There are three possible configurations:

- PUBLISH: the node sends the response.
- SUBSCRIBE: the node receives the response.
- IGNORE: the node is not concerned by the response, it does not send and does not receive the response.

This configuration is made by the field Node Action (NACT) in the US_LINMR (see USART LIN Mode Register).

Example: a LIN cluster that contains a master and two slaves:

- Data transfer from the master to the slave1 and to the slave2:

NACT(master)=PUBLISH

NACT(slave1)=SUBSCRIBE

NACT(slave2)=SUBSCRIBE

- Data transfer from the master to the slave1 only:

NACT(master)=PUBLISH

NACT(slave1)=SUBSCRIBE

NACT(slave2)=IGNORE

- Data transfer from the slave1 to the master:

NACT(master)=SUBSCRIBE

NACT(slave1)=PUBLISH

NACT(slave2)=IGNORE

- Data transfer from the slave1 to the slave2:

NACT(master)=IGNORE

NACT(slave1)=PUBLISH

NACT(slave2)=SUBSCRIBE

- Data transfer from the slave2 to the master and to the slave1:

NACT(master)=SUBSCRIBE

NACT(slave1)=SUBSCRIBE

© 2021 Microchip Technology Inc. and its subsidiaries

NACT(slave2)=PUBLISH

### 46.6.9.11 Response Data Length

The LIN response data length is the number of data fields (bytes) of the response excluding the checksum.

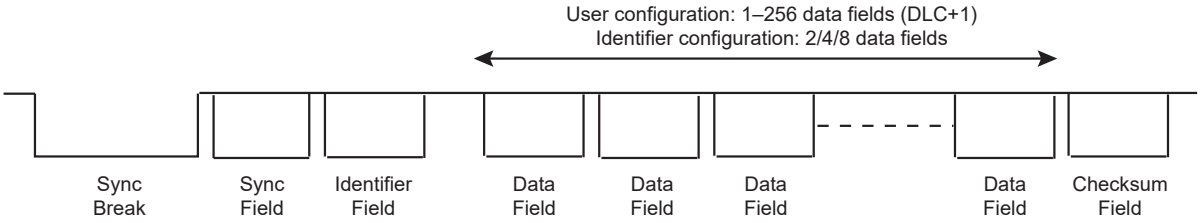
The response data length can either be configured by the user or be defined automatically by bits 4 and 5 of the Identifier (compatibility to LIN Specification 1.1). The user can choose between these two modes using the US_LINMR.DLM:

- DLM = 0: The response data length is configured by the user via US_LINMR.DLC. The response data length is equal to (DLC + 1) bytes. DLC can be programmed from 0 to 255, so the response can contain from 1 data byte up to 256 data bytes.
- DLM = 1: The response data length is defined by the Identifier (US_LINIR.IDCHR) according to the table below. The US_LINMR.DLC is discarded. The response can contain 2 or 4 or 8 data bytes.

**Table 46-13. Response Data Length if DLM = 1**

| IDCHR[5] | IDCHR[4] | Response Data Length [Bytes] |
|---|---|---|
| 0 | 0 | 2 |
| 0 | 1 | 2 |
| 1 | 0 | 4 |
| 1 | 1 | 8 |

**Figure 46-43. Response Data Length**




### 46.6.9.12 Checksum

The last field of a frame is the checksum. The checksum contains the inverted 8-bit sum with carry, over all data bytes or all data bytes and the protected identifier. Checksum calculation over the data bytes only is called classic checksum and it is used for communication with LIN 1.3 slaves. Checksum calculation over the data bytes and the protected identifier byte is called enhanced checksum and it is used for communication with LIN 2.0 slaves.

The USART can be configured to:

- Send/Check an Enhanced checksum automatically (CHKDIS = 0 & CHKTYP = 0)
- Send/Check a Classic checksum automatically (CHKDIS = 0 & CHKTYP = 1)
- Not send/check a checksum (CHKDIS = 1)

This configuration is made by the Checksum Type (CHKTYP) and Checksum Disable (CHKDIS) fields of US_LINMR.

If the checksum feature is disabled, the user can send it manually all the same, by considering the checksum as a normal data byte and by adding 1 to the response data length (see Response Data Length).

### 46.6.9.13 Frame Slot Mode

This mode is useful only for master nodes. It complies with the following rule: each frame slot should be longer than or equal to $t_{Frame_Maximum}$.

If the Frame Slot mode is enabled (FSDIS = 0) and a frame transfer has been completed, the TXRDY flag is set again only after $t_{Frame_Maximum}$ delay, from the start of frame. So the master node cannot send a new header if the frame slot duration of the previous frame is inferior to $t_{Frame_Maximum}$.

If the Frame Slot mode is disabled (FSDIS = 1) and a frame transfer has been completed, the TXRDY flag is set again immediately.

The $t_{Frame_Maximum}$ is calculated as below:

If the Checksum is sent (CHKDIS = 0):

$t_{Header_Nominal} = 34 \times t_{bit}$

$t_{Response_Nominal} = 10 \times (NData + 1) \times t_{bit}$

$t_{Frame_Maximum} = 1.4 \times (t_{Header_Nominal} + t_{Response_Nominal} + 1)$(1)

$t_{Frame_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1 + 1) + 1) \times t_{bit}$

$t_{Frame_Maximum} = (77 + 14 \times DLC) \times t_{bit}$

If the Checksum is not sent (CHKDIS = 1):

$t_{Header_Nominal} = 34 \times t_{bit}$

$t_{Response_Nominal} = 10 \times NData \times t_{bit}$

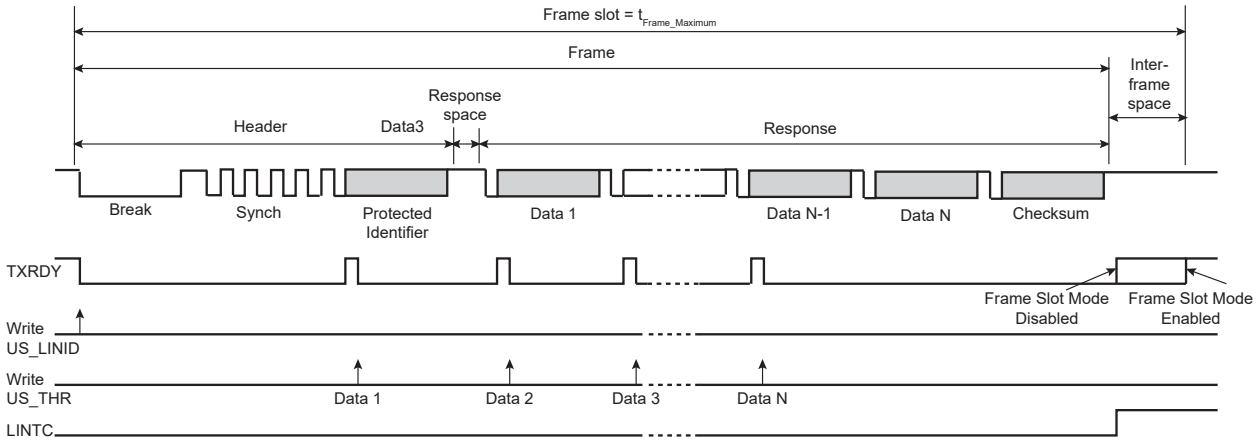
$t_{Frame_Maximum} = 1.4 \times (t_{Header_Nominal} + t_{Response_Nominal} + 1)$(1)

$t_{Frame_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1) + 1) \times t_{bit}$

$t_{Frame_Maximum} = (63 + 14 \times DLC) \times t_{bit}$

Note: 1. The term “+1” leads to an integer result for $t_{Frame_Maximum}$ (LIN Specification 1.3).

**Figure 46-44. Frame Slot Mode**




### 46.6.9.14 LIN Errors

#### 46.6.9.14.1 Bit Error

This error is generated in master of slave node configuration, when the USART is transmitting and if the transmitted value on the Tx line is different from the value sampled on the Rx line. If a bit error is detected, the transmission is aborted at the next byte border.

This error is reported by flag US_CSR.LINBE.

#### 46.6.9.14.2 Inconsistent Synch Field Error

This error is generated in slave node configuration, if the Synch Field character received is other than 0x55.

This error is reported by flag US_CSR.LINISFE.

#### 46.6.9.14.3 Identifier Parity Error

This error is generated in slave node configuration, if the parity of the identifier is wrong. This error can be generated only if the parity feature is enabled (PARDIS = 0).

This error is reported by flag US_CSR.LINIPE.

#### 46.6.9.14.4 Checksum Error

This error is generated in master of slave node configuration, if the received checksum is wrong. This flag can be set to 1 only if the checksum feature is enabled (CHKDIS = 0).

This error is reported by flag US_CSR.LINCE.

#### 46.6.9.14.5 Slave Not Responding Error

This error is generated in master of slave node configuration, when the USART expects a response from another node (NACT = SUBSCRIBE) but no valid message appears on the bus within the time given by the maximum length of the message frame, $t_{Frame_Maximum}$ (see Frame Slot Mode). This error is disabled if the USART does not expect any message (NACT = PUBLISH or NACT = IGNORE).

This error is reported by flag US_CSR.LINSNRE.

#### 46.6.9.14.6 Synch Tolerance Error

This error is generated in slave node configuration if, after the clock synchronization procedure, it appears that the computed baudrate deviation compared to the initial baudrate is superior to the maximum tolerance FTol_Unsynch (±15%).

This error is reported by flag US_CSR.LINSTE.

#### 46.6.9.14.7 Header Timeout Error

This error is generated in slave node configuration, if the Header is not entirely received within the time given by the maximum length of the Header, $t_{Header_Maximum}$.

This error is reported by flag US_CSR.LINHTE.

### 46.6.9.15 LIN Frame Handling

#### 46.6.9.15.1 Master Node Configuration

- Write TXEN and RXEN in US_CR to enable both the transmitter and the receiver.
- Write USART_MODE in US_MR to select the LIN mode and the master node configuration.
- Write CD and FP in US_BRGR to configure the baud rate.
- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM, FSDIS and DLC in US_LINMR to configure the frame transfer.
- Check that TXRDY in US_CSR is set to 1.
- Write IDCHR in US_LINIR to send the header.

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the USART sends the response
  - Wait until TXRDY in US_CSR rises.
  - Write TCHR in US_THR to send a byte.
  - If all the data have not been written, redo the two previous steps.
  - Wait until LINTC in US_CSR rises.
  - Check the LIN errors.
- Case 2: NACT = SUBSCRIBE, the USART receives the response
  - Wait until RXRDY in US_CSR rises.
  - Read RCHR in US_RHR.
  - If all the data have not been read, redo the two previous steps.
  - Wait until LINTC in US_CSR rises.
  - Check the LIN errors.
- Case 3: NACT = IGNORE, the USART is not concerned by the response.
  - Wait until LINTC in US_CSR rises.
  - Check the LIN errors.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 46-45. Master Node Configuration, NACT = PUBLISH**

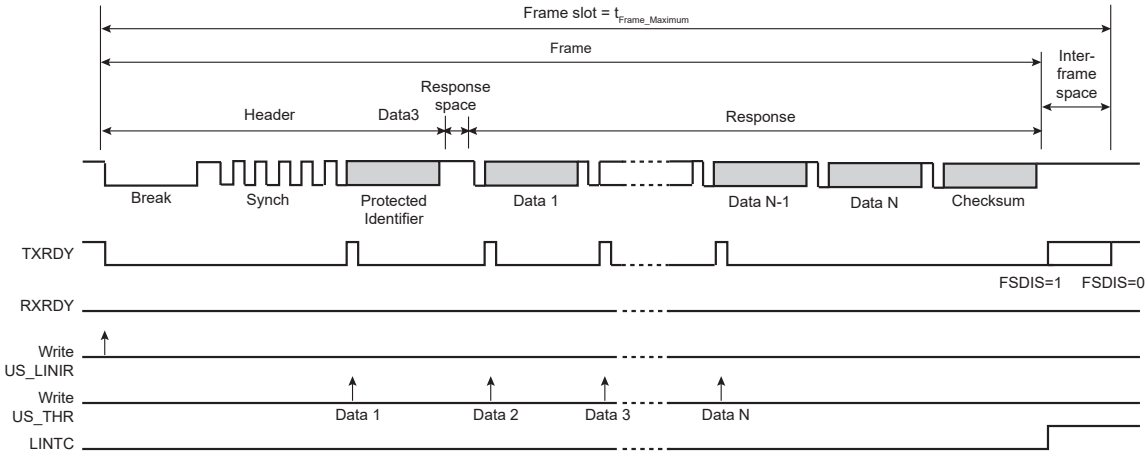



**Figure 46-46. Master Node Configuration, NACT = SUBSCRIBE**

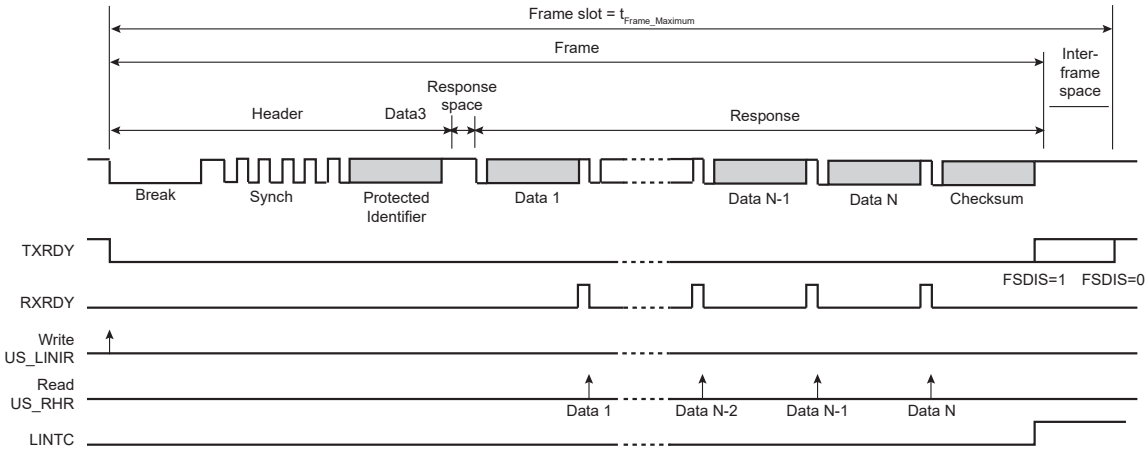
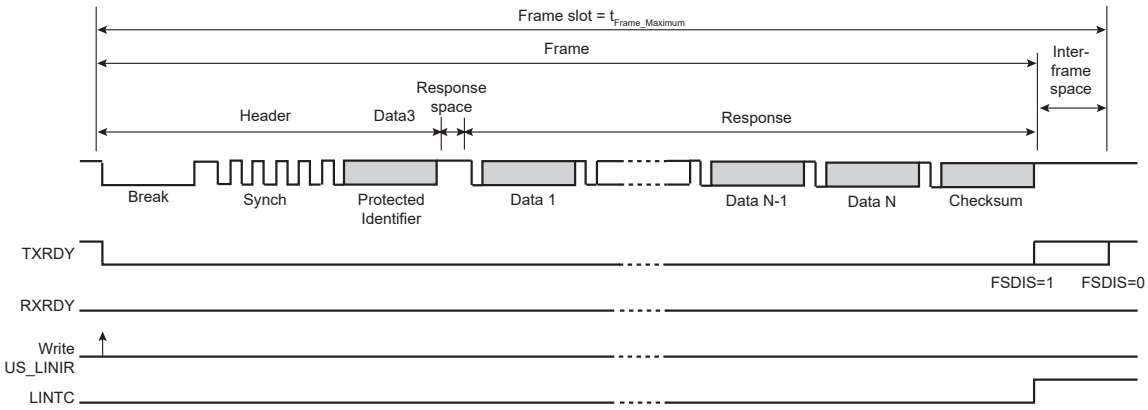



**Figure 46-47. Master Node Configuration, NACT = IGNORE**




#### 46.6.9.15.2 Slave Node Configuration

- Write TXEN and RXEN in US_CR to enable both the transmitter and the receiver.
- Write USART_MODE in US_MR to select the LIN mode and the slave node configuration.
- Write CD and FP in US_BRGR to configure the baud rate.
- Wait until LINID in US_CSR rises.
- Check LINISFE and LINPE errors.
- Read IDCHR in US_RHR.

- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM and DLC in US_LINMR to configure the frame transfer.

IMPORTANT: If the NACT configuration for this frame is PUBLISH, the US_LINMR must be written with NACT = PUBLISH even if this field is already correctly configured, in order to set the TXREADY flag and the corresponding write transfer request.

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the LIN controller sends the response
  - Wait until TXRDY in US_CSR rises.
  - Write TCHR in US_THR to send a byte.
  - If all the data have not been written, redo the two previous steps.
  - Wait until LINTC in US_CSR rises.
  - Check the LIN errors.
- Case 2: NACT = SUBSCRIBE, the USART receives the response
  - Wait until RXRDY in US_CSR rises.
  - Read RCHR in US_RHR.
  - If all the data have not been read, redo the two previous steps.
  - Wait until LINTC in US_CSR rises.
  - Check the LIN errors.
- Case 3: NACT = IGNORE, the USART is not concerned by the response
  - Wait until LINTC in US_CSR rises.
  - Check the LIN errors.

**Figure 46-48. Slave Node Configuration, NACT = PUBLISH**

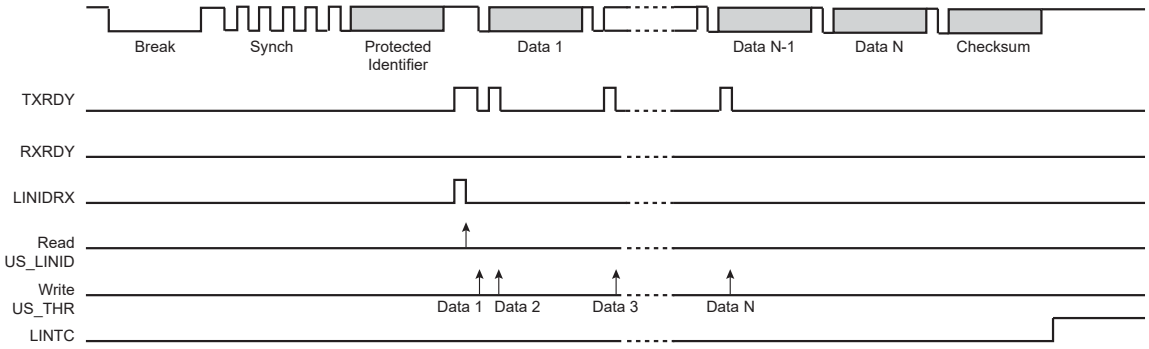



**Figure 46-49. Slave Node Configuration, NACT = SUBSCRIBE**

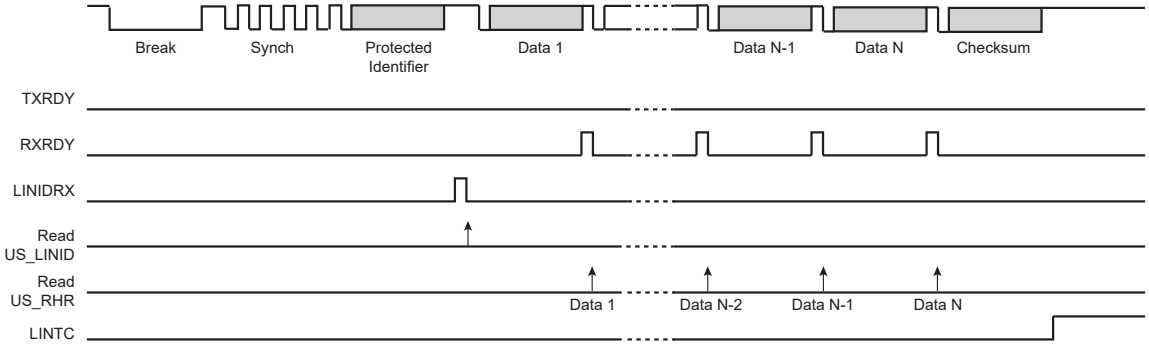
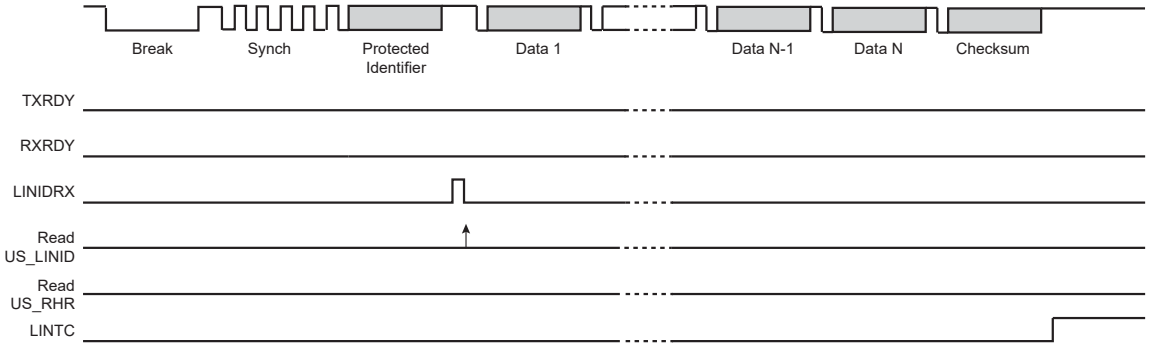


**Figure 46-50. Slave Node Configuration, NACT = IGNORE**




### 46.6.9.16 LIN Frame Handling with the DMAC

The USART can be used in association with the DMAC in order to transfer data directly into/from the on- and off-chip memories without any processor intervention.

The DMAC uses the trigger flags, TXRDY and RXRDY, to write or read into the USART. The DMAC always writes in the Transmit Holding register (US_THR) and it always reads in the Receive Holding register (US_RHR). The size of the data written or read by the DMAC in the USART is always a byte.

#### 46.6.9.16.1 Master Node Configuration

The user can choose between two DMAC modes by the PDCM bit in the US_LINMR:

- PDCM = 1: the LIN configuration is stored in the WRITE buffer and it is written by the DMAC in the Transmit Holding register US_THR (instead of the LIN Mode register US_LINMR). Because the DMAC transfer size is limited to a byte, the transfer is split into two accesses. During the first access the bits, NACT, PARDIS, CHKDIS, CHKTYP, DLM and FSDIS are written. During the second access the 8-bit DLC field is written.
- PDCM = 0: the LIN configuration is not stored in the WRITE buffer and it must be written by the user in US_LINMR.

The WRITE buffer also contains the Identifier and the DATA, if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the DATA if the USART receives the response (NACT = SUBSCRIBE).

**Figure 46-51. Master Node with DMAC (PDCM = 1)**

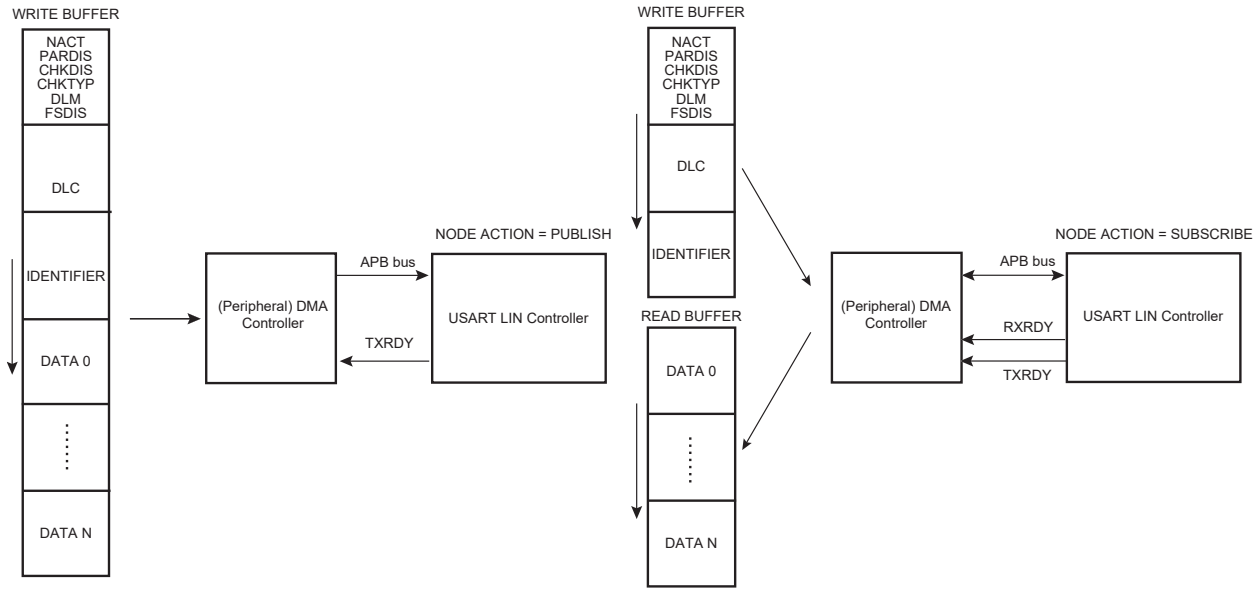
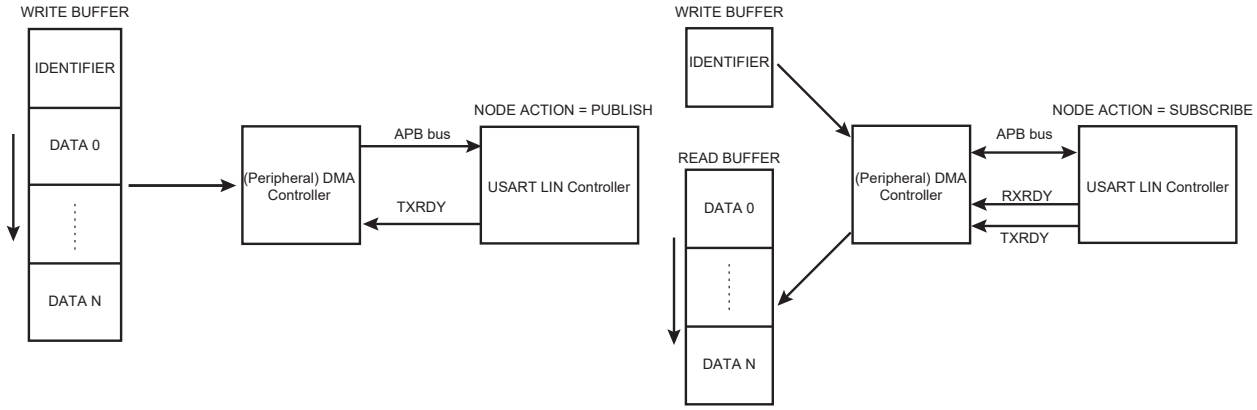


**Figure 46-52. Master Node with DMAC (PDCM = 0)**



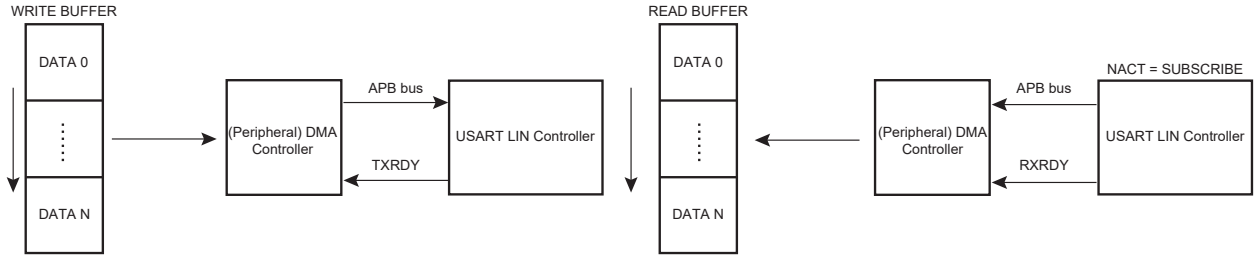

### 46.6.9.16.2 Slave Node Configuration

In this configuration, the DMAC transfers only the DATA. The Identifier must be read by the user in the LIN Identifier register (US_LINIR). The LIN mode must be written by the user in US_LINMR.

The WRITE buffer contains the DATA if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the DATA if the USART receives the response (NACT = SUBSCRIBE).

**Figure 46-53. Slave Node with DMAC**




### 46.6.9.17 Wakeup Request

Any node in a sleeping LIN cluster may request a wakeup.

In the LIN 2.0 specification, the wakeup request is issued by forcing the bus to the dominant state from 250 µs to 5 ms. For this, it is necessary to send the character 0xF0 in order to impose five successive dominant bits. Whatever the baud rate is, this character complies with the specified timings.

- Baud rate min = 1 kbit/s -> $t_{bit}$ = 1 ms -> 5 $t_{bit}$ = 5 ms
- Baud rate max = 20 kbit/s -> $t_{bit}$ = 50 µs -> 5 $t_{bit}$ = 250 µs

In the LIN 1.3 specification, the wakeup request should be generated with the character 0x80 in order to impose eight successive dominant bits.

The user can choose by the WKUPTYP bit in US_LINMR either to send a LIN 2.0 wakeup request (WKUPTYP = 0) or to send a LIN 1.3 wakeup request (WKUPTYP = 1).

A wakeup request is transmitted by writing a ‘1’ to US_CR.LINWKUP. Once the transfer is completed, US_SR.LINTC flag is asserted. It is cleared by writing a ‘1’ to US_CR.RSTSTA.

### 46.6.9.18 Bus Idle Timeout

If the LIN bus is inactive for a certain duration, the slave nodes shall automatically enter in Sleep mode. In the LIN 2.0 specification, this timeout is fixed at 4 seconds. In the LIN 1.3 specification, it is fixed at 25,000 $t_{bit}$.

In slave Node configuration, the receiver timeout detects an idle condition on the RXD line. When a timeout is detected, US_CSR.TIMEOUT rises and can generate an interrupt, thus indicating to the driver to go into Sleep mode.

The timeout delay period (during which the receiver waits for a new character) is programmed in US_RTOR.TO. If a ‘0’ is written to TO, the Receiver Timeout is disabled and no timeout is detected. US_CSR.TIMEOUT remains at ‘0’. Otherwise, the receiver loads a 17-bit counter with the value programmed in TO. This counter is decremented at

each bit period and reloaded each time a new character is received. If the counter reaches 0, US_CSR.TIMEOUT rises.

If US_CR.STTTO is written to ‘1’, the counter clock is stopped until a first character is received.

If US_CR.RETTO is written to ‘1’, the counter starts counting down immediately from the value TO.

**Table 46-14. Receiver Timeout Programming**

| LIN Specification | Baud Rate | Timeout period | US_RTOR.TO |
|---|---|---|---|
| 2.0 | 1,000 bit/s | 4 s | 4,000 |
| | 2,400 bit/s | | 9,600 |
| | 9,600 bit/s | | 38,400 |
| | 19,200 bit/s | | 76,800 |
| | 20,000 bit/s | | 80,000 |
| 1.3 | – | 25,000 $t_{bit}$ | 25,000 |

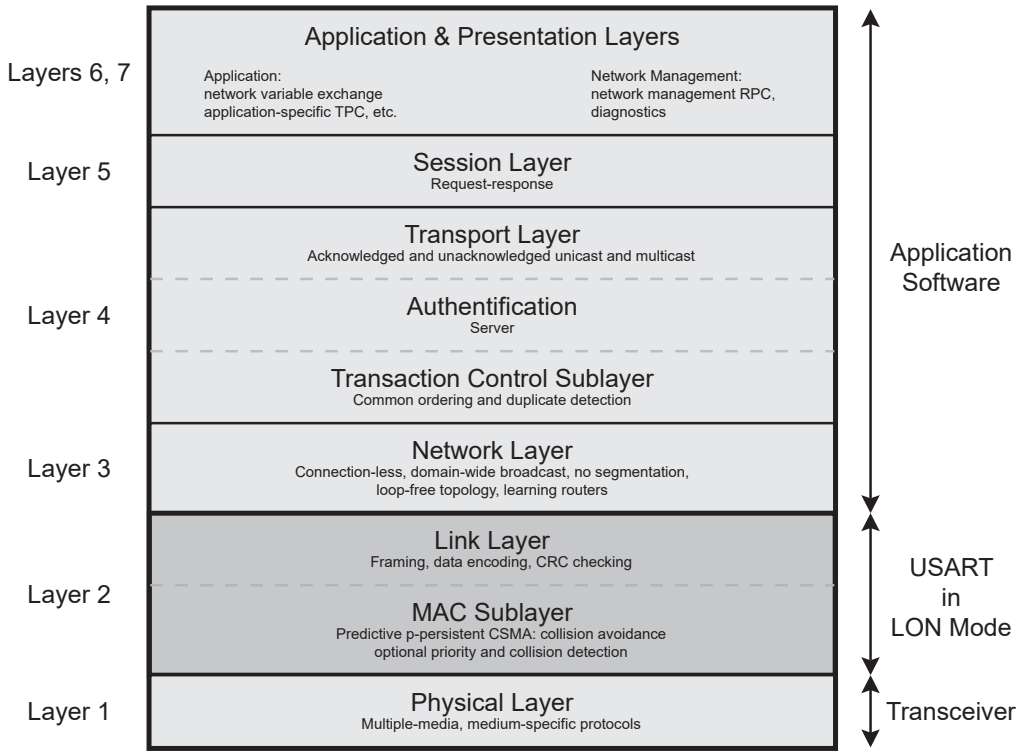
### 46.6.10 LON Mode

The LON mode provides connectivity to the local operating network (LON).

The LON standard covers all seven layers of the OSI (Open Systems Interconnect) reference model from the physical interfaces such as wired, power line, RF, and IP to the application layer and all layers in between. It was designed from the bottom up as a controls communication platform.

The LON mode enables the transmission and reception of Physical Protocol Data Unit (PPDU) frames with minimum intervention from the microprocessor.

**Figure 46-54. LON Protocol Layering**




The USART configured in LON mode is a full-layer 2 implementation including standard timings handling, framing (transmit and receive PPDU frames), backlog estimation and other features. At the frame encoding/decoding level, differential Manchester encoding is used (also known as CDP). When configured in LON mode, there is no embedded digital line filter, thus the optimal usage is node-to-node communication.

#### 46.6.10.1 Mode of Operation

To configure the USART to act as a LON node, the value 0x9 must be written to US_MR.USART_MODE.

To avoid unpredictable behavior, any change of the LON node configuration must be preceded by a software reset of the transmitter and the receiver (except the initial node configuration after a hardware reset) and followed by a transmitter/receiver enable. See Section 7.10.2.

#### 46.6.10.2 Receiver and Transmitter Control

See “Receiver and Transmitter Control”.

#### 46.6.10.3 Character Transmission

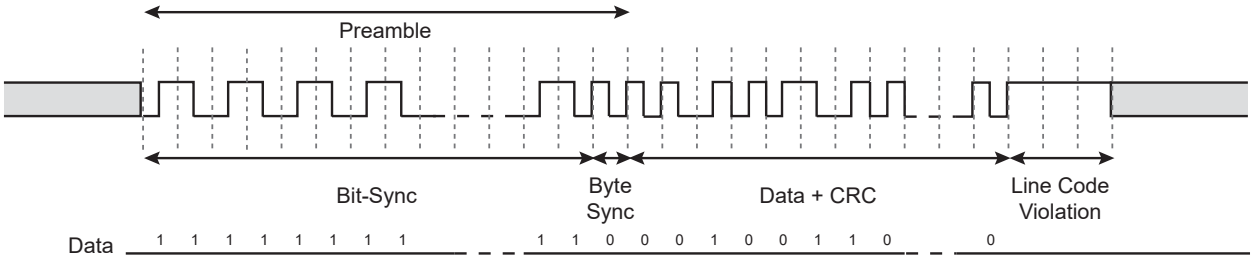
A LON frame is made up of a preamble, a data field (up to 256 bytes) and a 16-bit CRC field. The preamble and CRC fields are automatically generated and the LON node starts the transmission algorithm upon US_LONL2HDR register write. See “Sending A Frame”.

#### 46.6.10.4 Character Reception

When receiving a LON frame, the Receive Holding register (US_RHR) is updated upon completed character reception and the RXRDY bit in the Status register rises. If a character is completed while the RXRDY bit is set, the OVRE (Overrun Error) bit is set. The LON preamble field is only used for synchronization, therefore only the Data and CRC fields are transmitted to the Receive Holding register (US_RHR). See “Receiving A Frame”.

#### 46.6.10.5 LON Frame

**Figure 46-55. LON Framing**




##### 46.6.10.5.1 Encoding / Decoding

The USART configured in LON mode encodes transmitted data and decodes received data using differential Manchester encoding. In differential Manchester encoding, a ‘1’ bit is indicated by making the first half of the signal equal the last half of the previous bit's signal (no transition at the start of the bit-time). A ‘0’ bit is indicated by making the first half of the signal opposite to the last half of the previous bit's signal (a zero bit is indicated by a transition at the beginning of the bit-time). As is the case with normal Manchester encoding, missing transition at the middle of bit-time represents a Manchester code violation.

US_MAN.RXIDLEV informs the USART of the receiver line idle state value (receiver line inactive) thus ensuring higher reliability of preamble synchronization. By default, RXIDLEV is set to ‘1’ (receiver line is at level 1 when there is no activity).

Differential Manchester encoding is polarity insensitive.

**Figure 46-56. LON PPDU**



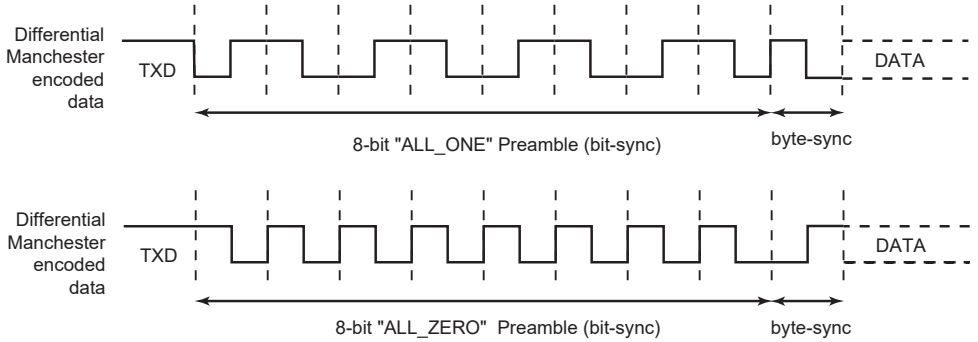

##### 46.6.10.5.2 Preamble Transmission

Each LON frame begins with a preamble of variable length which consists of a bit-sync field and a byte-sync field. The LONPL field of the USART LON Preamble register (US_LONPR) defines the preamble length. Note that preamble length of ‘0’ is not allowed.

The LON implementation allows two different preamble patterns ALL_ONE and ALL_ZERO which can be configured via US_MAN.TX_PL. The following figure illustrates and defines the valid patterns.

Other preamble patterns are not supported.

**Figure 46-57. Preamble Patterns**




#### 46.6.10.5.3 Preamble Reception

LON received frames begin with a preamble of variable length. The receiving algorithm does not check the preamble length, although a minimum of length of 4 bits is required for the receiving algorithm to consider the received preamble as valid.

As is the case with LON preamble transmission, two preamble patterns (ALL_ONE and ALL_ZERO) are allowed and can be configured through US_MAN.RX_PL. Figure 46-57 illustrates and defines the valid patterns.

Other preamble patterns are not supported.

#### 46.6.10.5.4 Header Transmission

Each LON frame, after sending the preamble, starts with the frame header also called L2HDR according to the CEA-709 specification. This header consist of the priority bit, the alternative path bit and the backlog increment. It is the first data to be sent.

In LON mode the transmitting algorithm starts when the US_LONL2HDR register is written (it is the first data to send).

#### 46.6.10.5.5 Header Reception

Each LON frame, after receiving the preamble, receives the frame header also called L2HDR according to the CEA-709 specification. This header consists of the priority bit, the alternative path bit, and the backlog increment.

The frame header is the first received data and the RXRDY bit rises as soon as the frame header as been received and stored in the Receive Holding register (US_RHR).

#### 46.6.10.5.6 Data

Data are sent/received serially after the preamble transmission/reception. Data can be either sent/received MSB first or LSB first depending on US_MR.MSBF.

#### 46.6.10.5.7 CRC

The two last bytes of LON frames are dedicated to CRC.

When transmitting, the CRC of the frame is automatically generated and sent when expected.

When receiving frames the CRC is automatically checked and a LCRCE flag is set in US_CSR if the calculated CRC do not match the received one. Note that the two received CRC bytes are seen as two additional data from the user point of view.

#### 46.6.10.5.8 End Of Frame

The USART configured in LON mode terminates the frame with a 3 $t_{bit}$ long Manchester code violation. After sending the last CRC bit it maintains the data transitionless during three bit periods.

### 46.6.10.6 LON Operating Modes

#### 46.6.10.6.1 Transmitting/Receiving Modules

According to the LON node configuration and LON network state, the transmitting module will be activated if a transmission request has been made and access to the LON bus granted. It returns to idle state once the transmission ends.

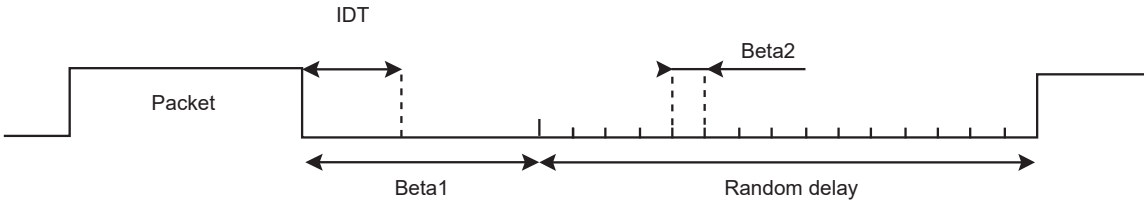
According to the LON node configuration and LON network state, the receiving module will be activated if a valid preamble is detected and the transmitting module is not activated.

#### 46.6.10.6.2 comm_type

In the CEA-709 standard, two communication configurations are defined and configurable through the comm_type variable. The comm_type variable value can be set in the USART LON Mode register (US_LONMR) through the COMMT bit. The selection of the comm_type determines the MAC behavior in the following ways:

- comm_type=1:
  - An indeterminate time is defined during the Beta 1 period in which all transitions on the channel are ignored, as shown in Figure 46-58.
  - The MAC sublayer ignores collisions occurring during the first 25% of the transmitted preamble. It optionally (according to US_LONMR.CDTAIL) ignores collisions reported following the transmission of the CRC but prior to the end of transmission.
  - If a collision is detected during preamble transmission, the MAC sublayer can terminate the packet if so configured according to US_LONMR.TCOL. Collisions detected after the preamble has been sent do not terminate transmission.
- comm_type=2:
  - No indeterminate time is defined at the MAC sublayer.
  - The MAC sublayer shall always terminate the packet upon notification of a collision.

**Figure 46-58. LON Indeterminate Time**




#### 46.6.10.6.3 Collision Detection

As an option of the CEA-709 standard, collision detection is supported through an active low Collision Detect (CD) input from the transceiver.

The Collision Detection source can be either external (See "I/O Lines Description") or internal. The collision detection source selection is defined through US_LONMR.LCDS.

The Collision Detection feature can be activated through US_LONMR.COLDET. If the collision detection feature is enabled and CD signal goes low for at least half $t_{bit}$ period then a collision is detected and reported as defined in "comm_type".

#### 46.6.10.6.4 Collision Detection Mode.

As defined in "comm_type", if comm_type=1 the LON node can be either configured to not terminate transmission upon collision notification during preamble transmission or terminate transmission.

US_LONMR.TCOL determines whether to terminate transmission or not upon collision notification during preamble transmission.

#### 46.6.10.6.5 Collision Detection After CRC

As defined in "comm_type" on page 64, if comm_type=1 the LON node can be either be configured to ignore collision after the CRC has been sent but prior to the end of the frame.

US_LONMR.CDTAIL determines whether such collision notifications must be considered or not.

#### 46.6.10.6.6 Random Number Generation

The Predictive p-persistent CSMA algorithm defined in the CEA-709.1 Standard is based on a random number generation.

This random number is automatically generated by an internal algorithm.

In addition, a USART IC DIFF register (US_ICDIFF) is available to avoid that two same chips with the same software generate the same random number after reset. The value of this register is used by the internal algorithm to generate the random number. Therefore, putting a different value here for each chip ensures that the random number generated after a reset at the same time, will not be the same. It is recommended to put the chip ID code here.

© 2021 Microchip Technology Inc. and its subsidiaries

#### 46.6.10.7 LON Node Backlog Estimation

As defined in the CEA-709 standard, the LON node maintains its own backlog estimation. The node backlog estimation is initially set to 1, will always be greater than 1 and will never exceed 63. If the node backlog estimation exceeds the maximum backlog value, the backlog value is set to 63 and a backlog overflow error flag is set (LBLOVFE flag).

The node backlog estimation is incremented each time a frame is sent or received successfully. The increment to the backlog is encoded into the link layer header, and represents the number of messages that the packet shall cause to be generated upon reception.

The backlog decrements under one of the following conditions:

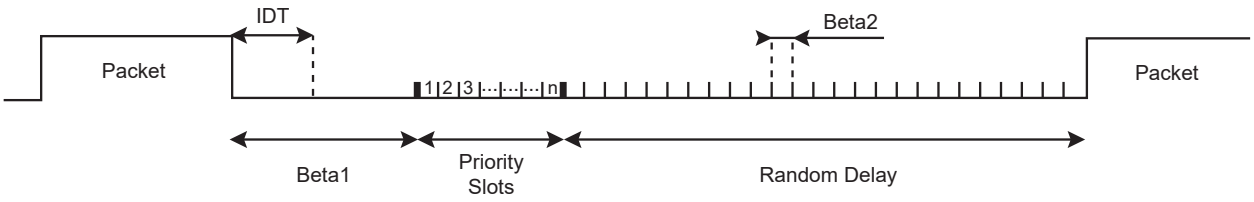
- On waiting to transmit: If Wbase randomizing slots go by without channel activity.
- On receive: If a packet is received with a backlog increment of ‘0’.
- On transmit: If a packet is transmitted with a backlog increment of ‘0’.
- On idle: If a packet cycle time expires without channel activity.

##### 46.6.10.7.1 Optional Collision Detection Feature And Backlog Estimation

Each time a frame is transmitted and a collision occurred, the backlog is incremented by 1. In this case, the backlog increment encoded in the link layer is ignored.

#### 46.6.10.8 LON Timings

**Figure 46-59. LON Timings**




##### 46.6.10.8.1 Beta2

A node wishing to transmit generates a random delay T. This delay is an integer number of randomizing slots of duration Beta2.

The beta2 length (in $t_{bit}$) is configurable through US_FIDI. Note that a length of ‘0’ is not allowed.

##### 46.6.10.8.2 Beta1 Tx/Rx

Beta1 is the period immediately following the end of a packet cycle (see Figure 46-59). A node attempting to transmit monitors the state of the channel, and if it detects no transmission during the Beta1 period, it determines the channel to be idle.

The Beta1 value is different depending on the previous packet type (received packet or transmitted packet).

Beta1Rx and Beta1Tx length can be configured respectively through the USART LON Beta1 Rx register (US_LONB1RX) and the USART LON Beta1 Tx register (US_LONB1TX). Note that a length of ‘0’ is not allowed.

##### 46.6.10.8.3 Pcycle Timer

The packet cycle timer is reset to its initial value whenever the backlog is changed. It is started (begins counting down at its current value) whenever the MAC layer becomes idle. An idle MAC layer is defined as:

- Not receiving
- Not transmitting
- Not waiting to transmit
- Not timing Beta1
- Not waiting for priority slots, and not waiting for the first Wbase randomizing window to complete

On transition from idle to either transmit or receive, the packet cycle timer is halted.

The pcycle timer value can be configured in US_TTGR. Note that ‘0’ value is not allowed.

##### 46.6.10.8.4 Wbase

The wbase timer represents the base windows size. Its duration, derived from Beta2, equals 16 Beta2 slots.

© 2021 Microchip Technology Inc.
and its subsidiaries

#### 46.6.10.8.5 Priority Slots

On a channel by channel basis, the protocol supports optional priority. Priority slots, if any, follow immediately after the Beta1 period that follows the transmission of a packet (see Figure 46-59). The number of priority slots per channel ranges from 0 to 127.

The number of priority slots in the LON network configuration is defined through the PSNB field of the USART LON Priority register (US_LONPRIO). And the priority slot affected to the LON node, if any, is defined through US_LONPRIO.NPS.

#### 46.6.10.8.6 Indeterminate Time

See "comm_type".

Like Beta1, the IDT value is different depending on what was the previous frame (transmitted or received frame).

IDTRx and IDTTx can be configured respectively through the USART LON IDT Rx register (US_LONIDTRX) and the USART LON IDT Tx register (US_LONIDTTX).

#### 46.6.10.8.7 End of Frame Condition

The USART configured in LON mode terminates the frame with a 3 $t_{bit}$ long Manchester code violation. After sending the last CRC bit, it maintains the data transitionless during three bit periods.

While receiving data the USART configured in LON mode will detect an end of frame condition after a $t_{eof}$ transitionless Manchester code violation. US_LONMR.EOFS can configure $t_{eof}$.

### 46.6.10.9 LON Errors

All these flags can be read in the Channel Status register (LON_MODE) (US_CSR) and will generate interrupts if configured in the Interrupt Enable register (LON_MODE) (US_IER ).

These flags can be reset through US_CR.RSTSTA.

#### 46.6.10.9.1 Underrun Error

If the USART is in LON mode and if a character is sent while the Transmit Holding register (US_THR) is empty, the UNRE bit flag is set.

#### 46.6.10.9.2 Collision Detection

The LCOL flag is set whenever a valid collision has been detected and the LON node is configured to report it (see "Collision Detection").

#### 46.6.10.9.3 LON Frame Early Termination

The LFET flag is set whenever a LON frame has been terminated early due to collision detection.

#### 46.6.10.9.4 Reception Error

The LCRCE flag is set if the received frame has an erroneous CRC and the flag LSFE is set if the received frame is too short (LON frames must be at least 8 bytes long).

These flags can be read in US_CSR.

#### 46.6.10.9.5 Backlog Overflow

The LBLOVFE flag is set if the LON node backlog estimation goes over 63 which is the maximum backlog value.

### 46.6.10.10 Drift Compensation

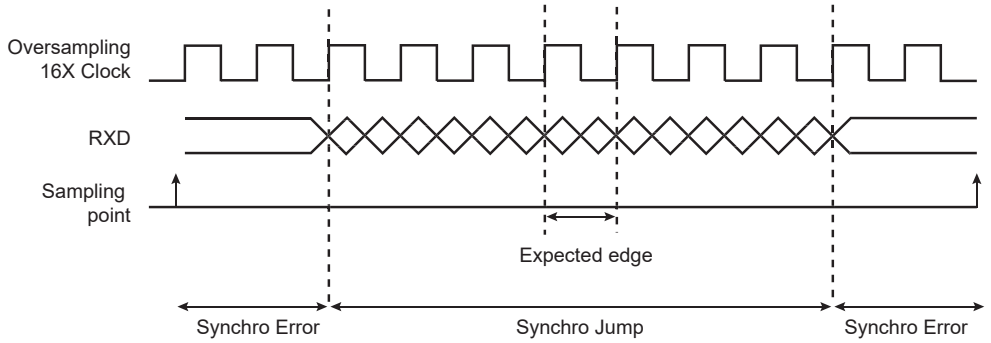
While receiving a frame, the baud rate used by the sender may not be exactly the one expected. In this case, the hardware drift compensation algorithm recovers up to 16% clock drift (expected baud rate ±16% will be supported).

Drift compensation is available only in 16X Oversampling mode. To enable the hardware system, US_MAN.DRIFT must be set. If the RXD edge is between one and three 16X clock cycles far from the expected edge, then the period is shortened or lengthened accordingly, to center the RXD edge.

The drift compensation hardware feature allows up to 16% clock drift to be handled, provided the system clock is fast enough compared to the selected baud rate.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 46-60. Bit Resynchronization**

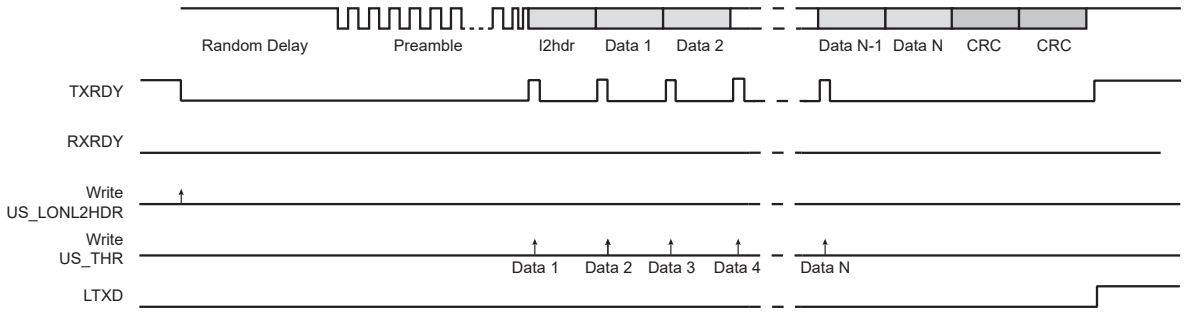



### 46.6.10.11 LON Frame Handling

#### 46.6.10.11.1 Sending A Frame

1. Write TXEN and RXEN in US_CR to enable both the transmitter and the receiver.
2. Write USART_MODE in US_MR to select the LON mode configuration.
3. Write CD and FP in US_BRGR to configure the baud rate.
4. Write COMMT, COLDET, TCOL, CDTAIL, RDMNBM and DMAM in US_LONMR to configure the LON operating mode.
5. Write BETA2, BETA1TX, BETA1RX, PCYCLE, PSNB, NPS, IDTTX and ITDRX respectively in US_FIDI, US_LONB1TX, US_LONB1RX, US_TTGR, US_LONPRIO, US_LONIDTTX and US_LONIDTRX to set the LON network configuration.
6. Write TX_PL in US_MAN to select the preamble pattern to use.
7. Write LONPL and LONDL in US_LONPR and US_LONDL to set the frame transfer.
8. Check that TXRDY in US_CSR is set to 1.
9. Write US_LONL2HDR register to send the header.
10. Wait until TXRDY in US_CSR rises.
11. Write TCHR in US_THR to send a byte.
12. If all the data have not been written, redo the two previous steps.
13. Wait until LTXD in US_CSR rises.
14. Check the LON errors.

**Figure 46-61. Tx Frame**



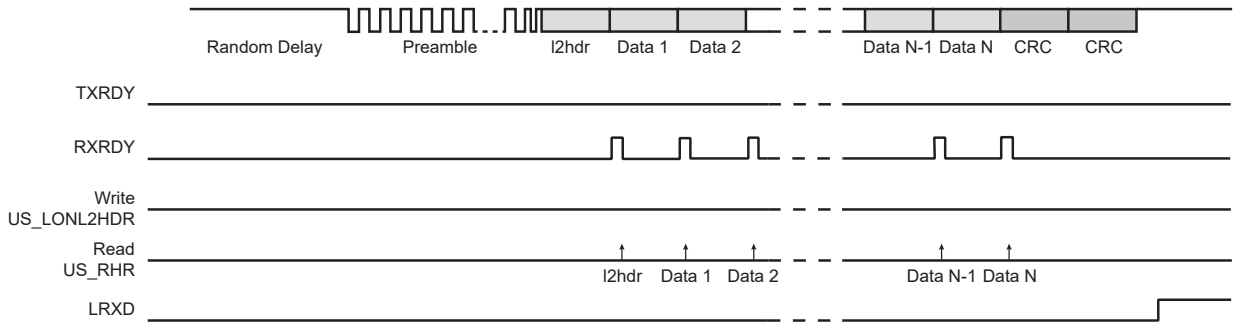

#### 46.6.10.11.2 Receiving A Frame

1. Write TXEN and RXEN in US_CR to enable both the transmitter and the receiver.
2. Write USART_MODE in US_MR to select the LON mode configuration.
3. Write CD and FP in US_BRGR to configure the baud rate.
4. Write COMMT, COLDET, TCOL, CDTAIL, RDMNBM and DMAM in US_LONMR to configure the LON operating mode.
5. Write BETA2, BETA1TX, BETA1RX, PCYCLE, PSNB, NPS, IDTTX and ITDRX respectively in US_FIDI, US_LONB1TX, US_LONB1RX, US_TTGR, US_LONPRIO, US_LONIDTTX and US_LONIDTRX to set the LON network configuration.

© 2021 Microchip Technology Inc.
and its subsidiaries

6. Write RXIDLEV and RX_PL in US_MAN to indicate the receiver line value and select the preamble pattern to use.
7. Wait until RXRDY in US_CSR rises.
8. Read RCHR in US_RHR.
9. If all the data and the two CRC bytes have not been read, redo the two previous steps.
10. Wait until LRXD in US_CSR rises.
11. Check the LON errors.
12.

**Figure 46-62. Rx Frame**




### 46.6.10.12 LON Frame Handling with the Peripheral DMA Controller

The USART can be used in association with the DMA Controller in order to transfer data directly into/from the on- and off-chip memories without any processor intervention.

The DMA uses the trigger flags, TXRDY and RXRDY, to write or read into the USART. The DMA always writes in US_THR and it always reads in US_RHR. The size of the data written or read by the DMA in the USART is always a byte.

#### 46.6.10.12.1 Configuration

The DMA mode is configured in USLONMR.DMAM:

- DMAM = 1: The LON frame data length (DATAL) is stored in the WRITE buffer and it is written by the DMA in US_THR (instead of the LON Data Length register US_LONDL).
- DMAM = 0: The LON frame data length (DATAL) is not stored in the WRITE buffer and it must be written by the user in US_LONDL.

In both DMA modes L2HDR is considered as a data and its value must be stored in the WRITE buffer as the first data to write.

**Figure 46-63. DMAM = 1**

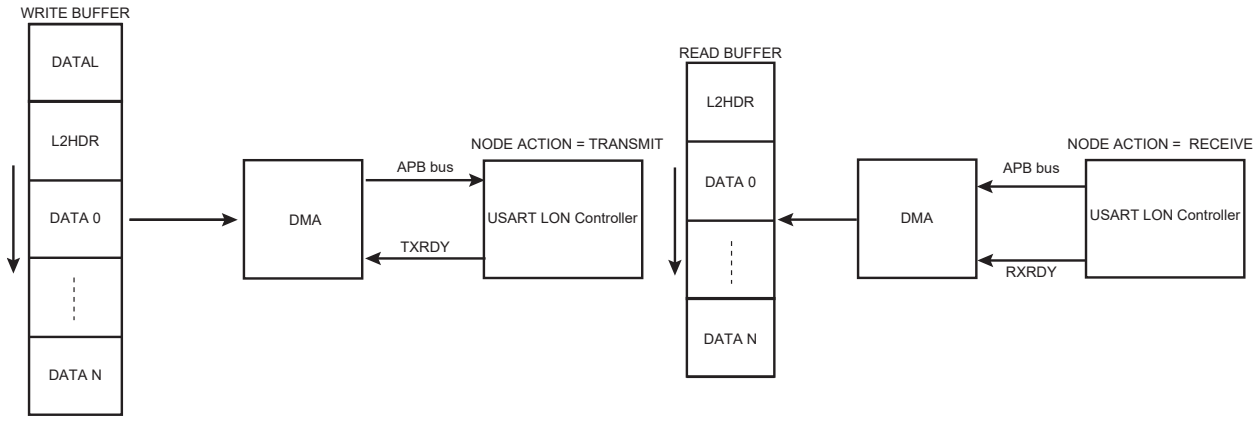
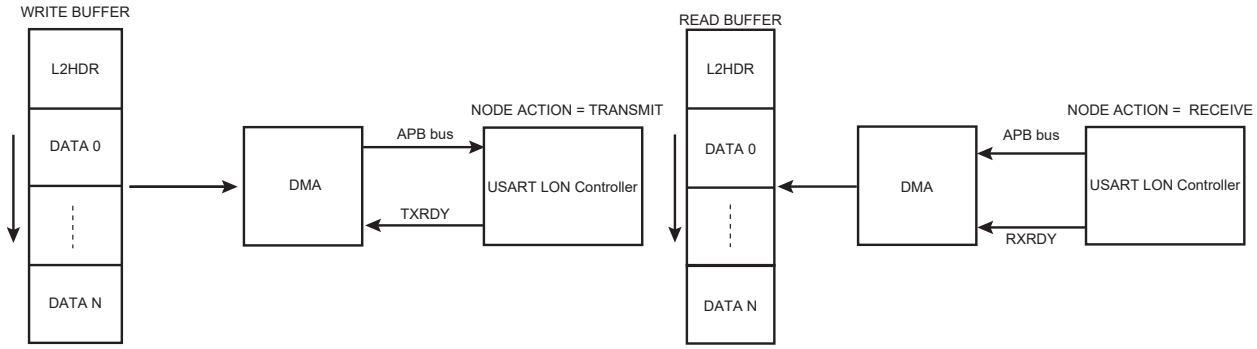


**Figure 46-64. DMAM = 0**



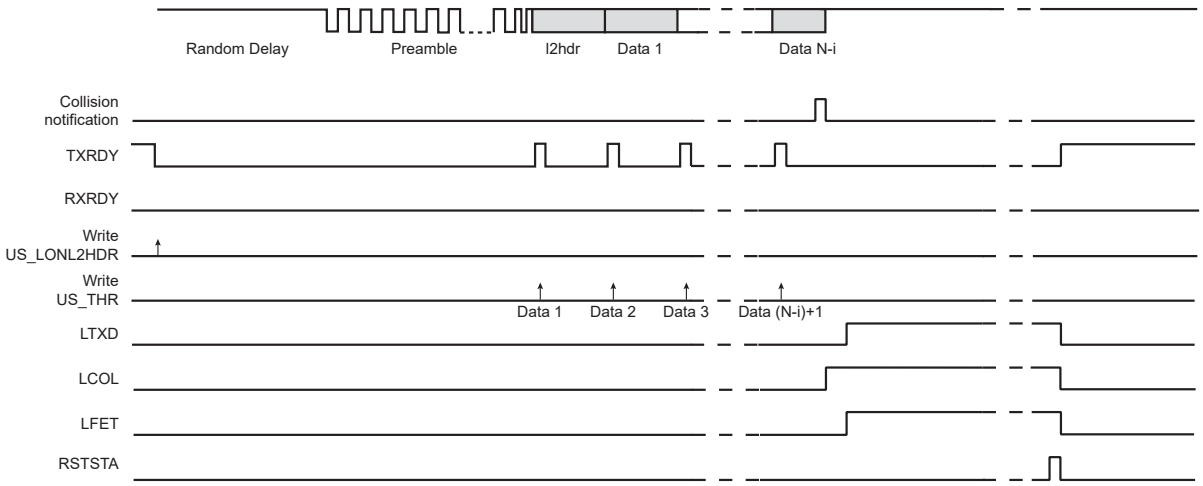

#### 46.6.10.12.2 DMA and Collision Detection

As explained in "comm_type", depending on LON configuration the transmission may be terminated early upon collision notification which means that the DMA transfer may be stopped before its end.

In case of early end of transmission due to collision detection the USART in LON mode acts as follows:

- Send the end of frame trigger.
- Hold down TXRDY avoiding thus any additional DMA transfer.
- Set LTXD, LCOL and LFET flags in US_CSR.
- Wait that the application reconfigure the DMA.
- Wait until LCOL and LFET flags are cleared through US_CR. RSTSTA (it releases the TXRDY signal).

**Figure 46-65. DMA, Collision and Early Frame Termination**



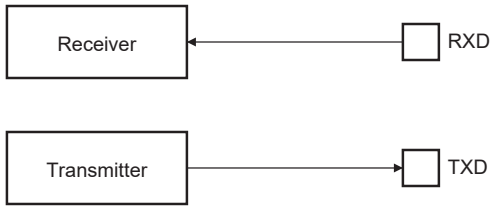

## 46.6.11 Test Modes

The USART can be programmed to operate in three different test modes. The internal loopback capability allows on-board diagnostics. In Loopback mode, the USART interface pins are disconnected or not and reconfigured for loopback internally or externally.

### 46.6.11.1 Normal Mode

Normal mode connects the RXD pin on the receiver input and the transmitter output on the TXD pin.

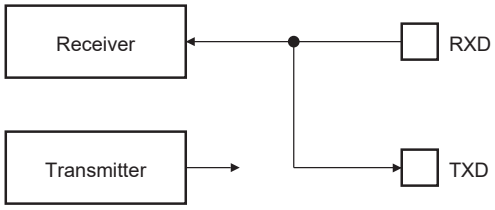
**Figure 46-66. Normal Mode Configuration**




### 46.6.11.2 Automatic Echo Mode

Automatic Echo mode allows bit-by-bit retransmission. When a bit is received on the RXD pin, it is sent to the TXD pin, as shown in the following figure. Programming the transmitter has no effect on the TXD pin. The RXD pin is still connected to the receiver input, thus the receiver remains active.

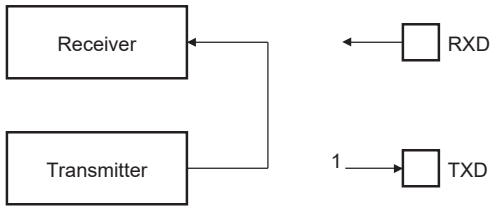
**Figure 46-67. Automatic Echo Mode Configuration**




### 46.6.11.3 Local Loopback Mode

Local Loopback mode connects the output of the transmitter directly to the input of the receiver, as shown in the following figure. The TXD and RXD pins are not used. The RXD pin has no effect on the receiver and the TXD pin is continuously driven high, as in idle state.

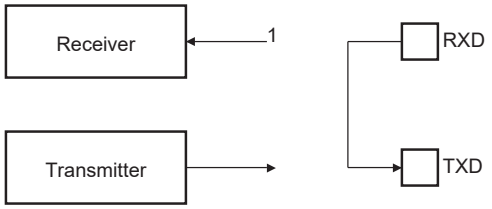
**Figure 46-68. Local Loopback Mode Configuration**




### 46.6.11.4 Remote Loopback Mode

Remote Loopback mode directly connects the RXD pin to the TXD pin, as shown in the following figure. The transmitter and the receiver are disabled and have no effect. This mode allows bit-by-bit retransmission.

**Figure 46-69. Remote Loopback Mode Configuration**




## 46.6.12 Register Write Protection

To prevent any single software error from corrupting USART behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the USART Write Protection Mode Register (US_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the USART Write Protection Status Register (US_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the US_WPSR.

The following registers can be write-protected:

- USART Mode Register
- USART Baud Rate Generator Register
- USART Receiver Timeout Register
- USART Transmitter Timeguard Register
- USART Manchester Configuration Register
- USART LON Mode Register
- USART LON Beta1 Tx Register
- USART LON Beta1 Rx Register
- USART LON Priority Register
- USART LON IDT Tx Register
- USART LON IDT Rx Register
- USART IC DIFF Register

© 2021 Microchip Technology Inc.
and its subsidiaries

## 46.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | US_CR | 7:0 | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| | | 15:8 | RETTO | RSTNACK | RSTIT | SENDA | STTTO | STPBRK | STTBRK | RSTSTA |
| | | 23:16 | | | LINWKUP | LINABT | RTSDIS | RTSEN | DTRDIS | DTREN |
| | | 31:24 | | | | | | | | |
| 0x00 | US_CR (SPI_MODE) | 7:0 | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| | | 15:8 | | | | | | | | RSTSTA |
| | | 23:16 | | | | | RCS | FCS | | |
| | | 31:24 | | | | | | | | |
| 0x04 | US_MR | 7:0 | CHRL[1:0] | | USCLKS[1:0] | | USART_MODE[3:0] | | | |
| | | 15:8 | CHMODE[1:0] | | NBSTOP[1:0] | | PAR[2:0] | | | SYNC |
| | | 23:16 | INVDATA | VAR_SYNC | DSNACK | INACK | OVER | CLKO | MODE9 | MSBF |
| | | 31:24 | ONEBIT | MODSYNC | MAN | FILTER | | MAX_ITERATION[2:0] | | |
| 0x04 | US_MR (SPI_MODE) | 7:0 | CHRL[1:0] | | USCLKS[1:0] | | USART_MODE[3:0] | | | |
| | | 15:8 | | | | | | | | CPHA |
| | | 23:16 | | | | WRDBT | | CLKO | | CPOL |
| | | 31:24 | | | | | | | | |
| 0x08 | US_IER | 7:0 | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| | | 15:8 | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | MANE | CTSIC | DCDIC | DSRIC | RIIC |
| | | 31:24 | | | | | | | | |
| 0x08 | US_IER (SPI_MODE) | 7:0 | | | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | | | | NSSE | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | US_IER (LIN_MODE) | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| 0x08 | US_IER (LON_MODE) | 7:0 | LCRCE | LSFE | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | LBLOVFE | LRXD | LFET | LCOL | LTXD |
| 0x0C | US_IDR | 7:0 | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| | | 15:8 | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | CTSIC | DCDIC | DSRIC | RIIC |
| | | 31:24 | | | | | | | | MANE |
| 0x0C | US_IDR (SPI_MODE) | 7:0 | | | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | | | | NSSE | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | US_IDR (LIN_MODE) | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| 0x0C | US_IDR (LON_MODE) | 7:0 | LCRCE | LSFE | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | LBLOVFE | LRXD | LFET | LCOL | LTXD |
| 0x10 | US_IMR | 7:0 | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| | | 15:8 | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | CTSIC | DCDIC | DSRIC | RIIC |
| | | 31:24 | | | | | | | | MANE |
| 0x10 | US_IMR (SPI_MODE) | 7:0 | | | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | | | | NSSE | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x10 | US_IMR (LIN_MODE) | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | | | | | | | | |
| | | 31:24 | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| 0x10 | US_IMR (LON_MODE) | 7:0 | LCRCE | LSFE | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | LBLOVFE | LRXD | LFET | LCOL | LTXD |
| 0x14 | US_CSR | 7:0 | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| | | 15:8 | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| | | 23:16 | CTS | DCD | DSR | RI | CTSIC | DCDIC | DSRIC | RIIC |
| | | 31:24 | | | | | | | | MANERR |
| 0x14 | US_CSR (SPI_MODE) | 7:0 | | | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | NSS | | | | NSSE | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | US_CSR (LIN_MODE) | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| | | 23:16 | LINBLS | | | | | | | |
| | | 31:24 | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| 0x14 | US_CSR (LON_MODE) | 7:0 | LCRCE | LSFE | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | | | | | | UNRE | TXEMPTY | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | LBLOVFE | LRXD | LFET | LCOL | LTXD |
| 0x18 | US_RHR | 7:0 | RXCHR[7:0] | | | | | | | |
| | | 15:8 | RXSYNH | | | | | | | RXCHR[8] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | US_THR | 7:0 | TXCHR[7:0] | | | | | | | |
| | | 15:8 | TXSYNH | | | | | | | TXCHR[8] |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | US_BRGR | 7:0 | CD[7:0] | | | | | | | |
| | | 15:8 | CD[15:8] | | | | | | | |
| | | 23:16 | | | | | | FP[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x24 | US_RTOR | 7:0 | TO[7:0] | | | | | | | |
| | | 15:8 | TO[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | TO[16] |
| | | 31:24 | | | | | | | | |
| 0x28 | US_TTGR | 7:0 | TG[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | US_TTGR (LON_MODE) | 7:0 | PCYCLE[7:0] | | | | | | | |
| | | 15:8 | PCYCLE[15:8] | | | | | | | |
| | | 23:16 | PCYCLE[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | US_FIDI | 7:0 | FI_DI_RATIO[7:0] | | | | | | | |
| | | 15:8 | FI_DI_RATIO[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 | US_FIDI (LON_MODE) | 7:0 | BETA2[7:0] | | | | | | | |
| | | 15:8 | BETA2[15:8] | | | | | | | |
| | | 23:16 | BETA2[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x44 | US_NER | 7:0 | NB_ERRORS[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 ... 0x4B | Reserved | | | | | | | | | |
| 0x4C | US_IF | 7:0 | IRDA_FILTER[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x50 | US_MAN | 7:0 | | | | | TX_PL[3:0] | | | |
| | | 15:8 | | | | TX_MPOL | | | TX_PP[1:0] | |
| | | 23:16 | | | | | RX_PL[3:0] | | | |
| | | 31:24 | RXIDLEV | DRIFT | ONE | RX_MPOL | | | RX_PP[1:0] | |
| 0x54 | US_LINMR | 7:0 | WKUPTYP | FSDIS | DLM | CHKTYP | CHKDIS | PARDIS | NACT[1:0] | |
| | | 15:8 | DLC[7:0] | | | | | | | |
| | | 23:16 | | | | | | | SYNCDIS | PDCM |
| | | 31:24 | | | | | | | | |
| 0x58 | US_LINIR | 7:0 | IDCHR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x5C | US_LINBRR | 7:0 | LINCD[7:0] | | | | | | | |
| | | 15:8 | LINCD[15:8] | | | | | | | |
| | | 23:16 | | | | | | LINFP[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x60 | US_LONMR | 7:0 | | | LCDS | DMAM | CDTAIL | TCOL | COLDET | COMMT |
| | | 15:8 | | | | | | | | |
| | | 23:16 | EOFS[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x64 | US_LONPR | 7:0 | LONPL[7:0] | | | | | | | |
| | | 15:8 | | | LONPL[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x68 | US_LONDL | 7:0 | LONDL[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x6C | US_LONL2HDR | 7:0 | PB | ALTP | BLI[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x70 | US_LONBL | 7:0 | | | LONBL[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 | US_LONB1TX | 7:0 | BETA1TX[7:0] | | | | | | | |
| | | 15:8 | BETA1TX[15:8] | | | | | | | |
| | | 23:16 | BETA1TX[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x78 | US_LONB1RX | 7:0 | BETA1RX[7:0] | | | | | | | |
| | | 15:8 | BETA1RX[15:8] | | | | | | | |
| | | 23:16 | BETA1RX[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x7C | US_LONPRIO | 7:0 | | PSNB[6:0] | | | | | | |
| | | 15:8 | | NPS[6:0] | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x80 | US_IDTTX | 7:0 | IDTTX[7:0] | | | | | | | |
| | | 15:8 | IDTTX[15:8] | | | | | | | |
| | | 23:16 | IDTTX[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 | US_IDTRX | 7:0 | IDTRX[7:0] | | | | | | | |
| | | 15:8 | IDTRX[15:8] | | | | | | | |
| | | 23:16 | IDTRX[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x88 | US_ICDIFF | 7:0 | | | | | ICDIFF[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x8C ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | US_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | US_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

### 46.7.1 USART Control Register

**Name:** US_CR
**Offset:** 0x0000
**Property:** Write-only

For SPI control, see "USART Control Register (SPI_MODE)".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | LINWKUP | LINABT | RTSDIS | RTSEN | DTRDIS | DTREN |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | RETTO | RSTNACK | RSTIT | SENDA | STTTO | STPBRK | STTBRK | RSTSTA |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |

**Bit 21 – LINWKUP** Send LIN Wakeup Signal

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Sends a wakeup signal on the LIN bus. |

**Bit 20 – LINABT** Abort LIN Transmission

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Abort the current LIN transmission. |

**Bit 19 – RTSDIS** Request to Send Pin Control

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Drives RTS pin to 0 if US_MR.USART_MODE field = 2, else drives RTS pin to 1 if US_MR.USART_MODE field = 0. |

**Bit 18 – RTSEN** Request to Send Pin Control

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Drives RTS pin to 1 if US_MR.USART_MODE field = 2, else drives RTS pin to 0 if US_MR.USART_MODE field = 0. |

**Bit 17 – DTRDIS** Data Terminal Ready Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Drives the pin DTR to 1. |

**Bit 16 – DTREN** Data Terminal Ready Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Drives the pin DTR to 0. |

**Bit 15 – RETTO** Start Timeout Immediately

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Immediately restarts timeout period. |

**Bit 14 – RSTNACK** Reset Non Acknowledge

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets NACK in US_CSR. |

**Bit 13 – RSTIT** Reset Iterations

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets ITER in US_CSR. No effect if the ISO7816 is not enabled. |

**Bit 12 – SENDA** Send Address

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | In Multidrop mode only, the next character written to the US_THR is sent with the address bit set. |

**Bit 11 – STTTO** Clear TIMEOUT Flag and Start Timeout After Next Character Received

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Starts waiting for a character before enabling the timeout counter. Immediately disables a timeout period in progress. Resets the status bit TIMEOUT in US_CSR. |

**Bit 10 – STPBRK** Stop Break

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Stops transmission of the break after a minimum of one character length and transmits a high level during 12-bit periods. No effect if no break is being transmitted. |

**Bit 9 – STTBRK** Start Break

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Starts transmission of a break after the characters present in US_THR and the Transmit Shift Register have been transmitted. No effect if a break is already being transmitted. |

**Bit 8 – RSTSTA** Reset Status Bits

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the status bits PARE, FRAME, OVRE, MANERR, LINBE, LINISFE, LINIPE, LINCE, LINSNRE, LINSTE, LINHTE, LINID, LINTC, LINBK and RXBRK in US_CSR. |

**Bit 7 – TXDIS** Transmitter Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the transmitter. |

**Bit 6 – TXEN** Transmitter Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the transmitter if TXDIS is 0. |

**Bit 5 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the receiver. |

**Bit 4 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the receiver, if RXDIS is 0. |

**Bit 3 – RSTTX** Reset Transmitter

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the transmitter. |

**Bit 2 – RSTRX** Reset Receiver

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the receiver. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.2 USART Control Register (SPI_MODE)

**Name:** US_CR (SPI_MODE)
**Offset:** 0x0000
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | RCS | FCS | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | RSTSTA |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |

**Bit 19 – RCS** Release SPI Chip Select
Applicable if USART operates in SPI Master mode (USART_MODE = 0xE):

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Releases the Slave Select Line NSS (RTS pin). |

**Bit 18 – FCS** Force SPI Chip Select
Applicable if USART operates in SPI Master mode (USART_MODE = 0xE):

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Forces the Slave Select Line NSS (RTS pin) to 0, even if USART is not transmitting, in order to address SPI slave devices supporting the CSAAT mode (Chip Select Active After Transfer). |

**Bit 8 – RSTSTA** Reset Status Bits

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the status bits OVRE, UNRE in US_CSR. |

**Bit 7 – TXDIS** Transmitter Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the transmitter. |

**Bit 6 – TXEN** Transmitter Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the transmitter if TXDIS is 0. |

**Bit 5 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the receiver. |

**Bit 4 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the receiver, if RXDIS is 0. |

**Bit 3 – RSTTX** Reset Transmitter

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the transmitter. |

**Bit 2 – RSTRX** Reset Receiver

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the receiver. |

### 46.7.3 USART Mode Register

**Name:** US_MR
**Offset:** 0x0004
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

For SPI configuration, see "USART Mode Register (SPI_MODE)".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ONEBIT | MODSYNC | MAN | FILTER | | MAX_ITERATION[2:0] | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | INVDATA | VAR_SYNC | DSNACK | INACK | OVER | CLKO | MODE9 | MSBF |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHMODE[1:0] | | NBSTOP[1:0] | | PAR[2:0] | | | SYNC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHRL[1:0] | | USCLKS[1:0] | | USART_MODE[3:0] | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – ONEBIT** Start Frame Delimiter Selector

| Value | Description |
|---|---|
| 0 | Start frame delimiter is COMMAND or DATA SYNC. |
| 1 | Start frame delimiter is one bit. |

**Bit 30 – MODSYNC** Manchester Synchronization Mode

| Value | Description |
|---|---|
| 0 | The Manchester start bit is a 0 to 1 transition |
| 1 | The Manchester start bit is a 1 to 0 transition. |

**Bit 29 – MAN** Manchester Encoder/Decoder Enable

| Value | Description |
|---|---|
| 0 | Manchester encoder/decoder are disabled. |
| 1 | Manchester encoder/decoder are enabled. |

**Bit 28 – FILTER** Receive Line Filter

| Value | Description |
|---|---|
| 0 | The USART does not filter the receive line. |
| 1 | The USART filters the receive line using a three-sample filter (1/16-bit clock) (2 over 3 majority). |

**Bits 26:24 – MAX_ITERATION[2:0]** Maximum Number of Automatic Iteration

| Value | Description |
|---|---|
| 0-7 | Defines the maximum number of iterations in ISO7816 mode, protocol T = 0. |

**Bit 23 – INVDATA** Inverted Data

| Value | Description |
|---|---|
| 0 | The data field transmitted on TXD line is the same as the one written in US_THR or the content read in US_RHR is the same as RXD line. Normal mode of operation. |
| 1 | The data field transmitted on TXD line is inverted (voltage polarity only) compared to the value written on US_THR or the content read in US_RHR is inverted compared to what is received on RXD line (or ISO7816 IO line). Inverted mode of operation, useful for contactless card application. To be used with configuration bit MSBF. |

**Bit 22 – VAR_SYNC** Variable Synchronization of Command/Data Sync Start Frame Delimiter

| Value | Description |
|---|---|
| 0 | User defined configuration of command or data sync field depending on MODSYNC value. |
| 1 | The sync field is updated when a character is written into US_THR. |

**Bit 21 – DSNACK** Disable Successive NACK

| Value | Description |
|---|---|
| 0 | NACK is sent on the ISO line as soon as a parity error occurs in the received character (unless INACK is set). |
| 1 | Successive parity errors are counted up to the value specified in the MAX_ITERATION field. These parity errors generate a NACK on the ISO line. As soon as this value is reached, no additional NACK is sent on the ISO line. The flag ITER is asserted.<br>Note: MAX_ITERATION field must be set to 0 if DSNACK is cleared. |

**Bit 20 – INACK** Inhibit Non Acknowledge

| Value | Description |
|---|---|
| 0 | The NACK is generated. |
| 1 | The NACK is not generated. |

**Bit 19 – OVER** Oversampling Mode

| Value | Description |
|---|---|
| 0 | 16X Oversampling |
| 1 | 8X Oversampling |

**Bit 18 – CLKO** Clock Output Select

| Value | Description |
|---|---|
| 0 | The USART does not drive the SCK pin. |
| 1 | The USART drives the SCK pin if USCLKS does not select the external clock SCK. |

**Bit 17 – MODE9** 9-bit Character Length

| Value | Description |
|---|---|
| 0 | CHRL defines character length. |
| 1 | 9-bit character length. |

**Bit 16 – MSBF** Bit Order

| Value | Description |
|---|---|
| 0 | Least significant bit is sent/received first. |
| 1 | Most significant bit is sent/received first. |

**Bits 15:14 – CHMODE[1:0]** Channel Mode

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode |
| 1 | AUTOMATIC | Automatic Echo. Receiver input is connected to the TXD pin. |
| 2 | LOCAL_LOOPBACK | Local Loopback. Transmitter output is connected to the Receiver Input. |
| 3 | REMOTE_LOOPBACK | Remote Loopback. RXD pin is internally connected to the TXD pin. |

**Bits 13:12 – NBSTOP[1:0]** Number of Stop Bits

| Value | Name | Description |
|---|---|---|
| 0 | 1_BIT | 1 stop bit |
| 1 | 1_5_BIT | 1.5 stop bit (SYNC = 0) or reserved (SYNC = 1) |
| 2 | 2_BIT | 2 stop bits |

**Bits 11:9 – PAR[2:0]** Parity Type

| Value | Name | Description |
|---|---|---|
| 0 | EVEN | Even parity |
| 1 | ODD | Odd parity |
| 2 | SPACE | Parity forced to 0 (Space) |
| 3 | MARK | Parity forced to 1 (Mark) |
| 4 | NO | No parity |
| 6 | MULTIDROP | Multidrop mode |

**Bit 8 – SYNC** Synchronous Mode Select

| Value | Description |
|---|---|
| 0 | USART operates in Asynchronous mode. |
| 1 | USART operates in Synchronous mode. |

**Bits 7:6 – CHRL[1:0]** Character Length

| Value | Name | Description |
|---|---|---|
| 0 | 5_BIT | Character length is 5 bits |
| 1 | 6_BIT | Character length is 6 bits |
| 2 | 7_BIT | Character length is 7 bits |
| 3 | 8_BIT | Character length is 8 bits |

**Bits 5:4 – USCLKS[1:0]** Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | MCK | Peripheral clock is selected |
| 1 | DIV | Peripheral clock divided (DIV=DIV=8) is selected |
| 2 | PCK | PMC programmable clock (PCK) is selected. If the SCK pin is driven (CLKO = 1), the CD field must be greater than 1. |
| 2 | — | Reserved |
| 3 | SCK | Serial clock (SCK) is selected |

**Bits 3:0 – USART_MODE[3:0]** USART Mode of Operation

| Value | Name | Description |
|---|---|---|
| 0x0 | NORMAL | Normal mode |
| 0x1 | RS485 | RS485 |
| 0x2 | HW_HANDSHAKING | Hardware Handshaking |
| 0x3 | MODEM | Modem |
| 0x4 | IS07816_T_0 | IS07816 Protocol: T = 0 |
| 0x6 | IS07816_T_1 | IS07816 Protocol: T = 1 |
| 0x8 | IRDA | IrDA |
| 0x9 | LON | LON |
| 0xA | LIN_MASTER | LIN Master mode |
| 0xB | LIN_SLAVE | LIN Slave mode |
| 0xE | SPI_MASTER | SPI Master mode (CLKO must be written to 1 and USCLKS = 0, 1 or 2) |
| 0xF | SPI_SLAVE | SPI Slave mode |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.4 USART Mode Register (SPI_MODE)

**Name:** US_MR (SPI_MODE)
**Offset:** 0x0004
**Reset:** 0x0
**Property:** Read/Write

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

– Applicable if USART operates in SPI mode (USART_MODE = 0xE or 0xF):

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | WRDBT | | CLKO | | CPOL |
| Reset | | | | 0 | | 0 | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | CPHA |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | CHRL[1:0] | | USCLKS[1:0] | | USART_MODE[3:0] | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 20 – WRDBT** Wait Read Data Before Transfer

| Value | Description |
|---|---|
| 0 | The character transmission starts as soon as a character is written into US_THR (assuming TXRDY was set). |
| 1 | The character transmission starts when a character is written and only if RXRDY flag is cleared (Receive Holding Register has been read). |

**Bit 18 – CLKO** Clock Output Select

| Value | Description |
|---|---|
| 0 | The USART does not drive the SCK pin. |
| 1 | The USART drives the SCK pin if USCLKS does not select the external clock SCK. |

**Bit 16 – CPOL** SPI Clock Polarity

Applicable if USART operates in SPI mode (Slave or Master, USART_MODE = 0xE or 0xF):
CPOL is used to determine the inactive state value of the serial clock (SCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | The inactive state value of SPCK is logic level zero. |
| 1 | The inactive state value of SPCK is logic level one. |

**Bit 8 – CPHA** SPI Clock Phase

CPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

| Value | Description |
|---|---|
| 0 | Data is changed on the leading edge of SPCK and captured on the following edge of SPCK. |

| Value | Description |
|---|---|
| 1 | Data is captured on the leading edge of SPCK and changed on the following edge of SPCK. |

**Bits 7:6 – CHRL[1:0]** Character Length

| Value | Name | Description |
|---|---|---|
| 3 | 8_BIT | Character length is 8 bits |

**Bits 5:4 – USCLKS[1:0]** Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | MCK | Peripheral clock is selected |
| 1 | DIV | Peripheral clock divided (DIV=DIV=8) is selected |
| 3 | SCK | Serial Clock (SCK) is selected |

**Bits 3:0 – USART_MODE[3:0]** USART Mode of Operation

| Value | Name | Description |
|---|---|---|
| 0xE | SPI_MASTER | SPI master |
| 0xF | SPI_SLAVE | SPI Slave |

### 46.7.5 USART Interrupt Enable Register

**Name:** US_IER
**Offset:** 0x0008
**Property:** Write-only

For SPI specific configuration, see “USART Interrupt Enable Register (SPI_MODE)”.

For LIN specific configuration, see “USART Interrupt Enable Register (LIN_MODE)”.

For LON specific configuration, see “USART Interrupt Enable Register (LON_MODE)”.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | MANE | CTSIC | DCDIC | DSRIC | RIIC |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| Reset | | | | | | | | |

**Bit 20 – MANE** Manchester Error Interrupt Enable

**Bit 19 – CTSIC** Clear to Send Input Change Interrupt Enable

**Bit 18 – DCDIC** Data Carrier Detect Input Change Interrupt Enable

**Bit 17 – DSRIC** Data Set Ready Input Change Enable

**Bit 16 – RIIC** Ring Indicator Input Change Enable

**Bit 13 – NACK** Non Acknowledge Interrupt Enable

**Bit 10 – ITER** Max number of Repetitions Reached Interrupt Enable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Enable

**Bit 8 – TIMEOUT** Timeout Interrupt Enable

**Bit 7 – PARE** Parity Error Interrupt Enable

**Bit 6 – FRAME** Framing Error Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 2 – RXBRK** Receiver Break Interrupt Enable

**Bit 1 – TXRDY** TXRDY Interrupt Enable

**Bit 0 – RXRDY** RXRDY Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.6 USART Interrupt Enable Register (SPI_MODE)

**Name:** US_IER (SPI_MODE)
**Offset:** 0x0008
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | NSSE | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | UNRE | TXEMPTY | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | OVRE | | | | TXRDY | RXRDY |

**Bit 19 – NSSE** NSS Line (Driving CTS Pin) Rising or Falling Edge Event Interrupt Enable

**Bit 10 – UNRE** SPI Underrun Error Interrupt Enable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Enable

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 1 – TXRDY** TXRDY Interrupt Enable

**Bit 0 – RXRDY** RXRDY Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.7 USART Interrupt Enable Register (LIN_MODE)

**Name:** US_IER (LIN_MODE)
**Offset:** 0x0008
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – LINHTE** LIN Header Timeout Error Interrupt Enable

**Bit 30 – LINSTE** LIN Synch Tolerance Error Interrupt Enable

**Bit 29 – LINSNRE** LIN Slave Not Responding Error Interrupt Enable

**Bit 28 – LINCE** LIN Checksum Error Interrupt Enable

**Bit 27 – LINIPE** LIN Identifier Parity Interrupt Enable

**Bit 26 – LINISFE** LIN Inconsistent Synch Field Error Interrupt Enable

**Bit 25 – LINBE** LIN Bus Error Interrupt Enable

**Bit 15 – LINTC** LIN Transfer Completed Interrupt Enable

**Bit 14 – LINID** LIN Identifier Sent or LIN Identifier Received Interrupt Enable

**Bit 13 – LINBK** LIN Break Sent or LIN Break Received Interrupt Enable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Enable

**Bit 8 – TIMEOUT** Timeout Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 7 – PARE** Parity Error Interrupt Enable

**Bit 6 – FRAME** Framing Error Interrupt Enable

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 1 – TXRDY** TXRDY Interrupt Enable

**Bit 0 – RXRDY** RXRDY Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.8 USART Interrupt Enable Register (LON_MODE)

**Name:** US_IER (LON_MODE)
**Offset:** 0x0008
**Property:** Write-only

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | LBLOVFE | LRXD | LFET | LCOL | LTXD |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LCRCE | LSFE | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 28 – LBLOVFE** LON Backlog Overflow Error Interrupt Enable

**Bit 27 – LRXD** LON Reception Done Interrupt Enable

**Bit 26 – LFET** LON Frame Early Termination Interrupt Enable

**Bit 25 – LCOL** LON Collision Interrupt Enable

**Bit 24 – LTXD** LON Transmission Done Interrupt Enable

**Bit 10 – UNRE** Underrun Error Interrupt Enable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Enable

**Bit 7 – LCRCE** LON CRC Error Interrupt Enable

**Bit 6 – LSFE** LON Short Frame Error Interrupt Enable

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 1 – TXRDY** TXRDY Interrupt Enable

**Bit 0 – RXRDY** RXRDY Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.9 USART Interrupt Disable Register

**Name:** US_IDR
**Offset:** 0x000C
**Property:** Write-only

For SPI specific configuration, see “USART Interrupt Disable Register (SPI_MODE)”.

For LIN specific configuration, see “USART Interrupt Disable Register (LIN_MODE)”.

For LON specific configuration, see “USART Interrupt Disable Register (LON_MODE)”.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | MANE |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CTSIC | DCDIC | DSRIC | RIIC |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 24 – MANE** Manchester Error Interrupt Disable

**Bit 19 – CTSIC** Clear to Send Input Change Interrupt Disable

**Bit 18 – DCDIC** Data Carrier Detect Input Change Interrupt Disable

**Bit 17 – DSRIC** Data Set Ready Input Change Disable

**Bit 16 – RIIC** Ring Indicator Input Change Disable

**Bit 13 – NACK** Non Acknowledge Interrupt Disable

**Bit 10 – ITER** Max Number of Repetitions Reached Interrupt Disable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Disable

**Bit 8 – TIMEOUT** Timeout Interrupt Disable

**Bit 7 – PARE** Parity Error Interrupt Disable

**Bit 6 – FRAME** Framing Error Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 5 – OVRE** Overrun Error Interrupt Enable

**Bit 2 – RXBRK** Receiver Break Interrupt Disable

**Bit 1 – TXRDY** TXRDY Interrupt Disable

**Bit 0 – RXRDY** RXRDY Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.10 USART Interrupt Disable Register (SPI_MODE)

**Name:** US_IDR (SPI_MODE)
**Offset:** 0x000C
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | NSSE | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | UNRE | TXEMPTY | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | OVRE | | | | TXRDY | RXRDY |

**Bit 19 – NSSE** NSS Line (Driving CTS Pin) Rising or Falling Edge Event Interrupt Disable

**Bit 10 – UNRE** SPI Underrun Error Interrupt Disable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Disable

**Bit 5 – OVRE** Overrun Error Interrupt Disable

**Bit 1 – TXRDY** TXRDY Interrupt Disable

**Bit 0 – RXRDY** RXRDY Interrupt Disable

### 46.7.11 USART Interrupt Disable Register (LIN_MODE)

**Name:** US_IDR (LIN_MODE)
**Offset:** 0x000C
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

Interrupt Disable

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 31 – LINHTE** LIN Header Timeout Error Interrupt Disable

**Bit 30 – LINSTE** LIN Synch Tolerance Error Interrupt Disable

**Bit 29 – LINSNRE** LIN Slave Not Responding Error Interrupt Disable

**Bit 28 – LINCE** LIN Checksum Error Interrupt Disable

**Bit 27 – LINIPE** LIN Identifier Parity Interrupt Disable

**Bit 26 – LINISFE** LIN Inconsistent Synch Field Error Interrupt Disable

**Bit 25 – LINBE** LIN Bus Error Interrupt Disable

**Bit 15 – LINTC** LIN Transfer Completed Interrupt Disable

**Bit 14 – LINID** LIN Identifier Sent or LIN Identifier Received Interrupt Disable

**Bit 13 – LINBK** LIN Break Sent or LIN Break Received

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 8 – TIMEOUT** Timeout Interrupt Disable

**Bit 7 – PARE** Parity Error Interrupt Disable

**Bit 6 – FRAME** Framing Error Interrupt Disable

**Bit 5 – OVRE** Overrun Error Interrupt Disable

**Bit 1 – TXRDY** TXRDY Interrupt Disable

**Bit 0 – RXRDY** RXRDY Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.12 USART Interrupt Disable Register (LON_MODE)

**Name:** US_IDR (LON_MODE)
**Offset:** 0x000C
**Property:** Write-only

This configuration is relevant only if USART_MODE = 0x9 in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | LBLOVFE | LRXD | LFET | LCOL | LTXD |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LCRCE | LSFE | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 28 – LBLOVFE** LON Backlog Overflow Error Interrupt Disable

**Bit 27 – LRXD** LON Reception Done Interrupt Disable

**Bit 26 – LFET** LON Frame Early Termination Interrupt Disable

**Bit 25 – LCOL** LON Collision Interrupt Disable

**Bit 24 – LTXD** LON Transmission Done Interrupt Disable

**Bit 10 – UNRE** Underrun Error Interrupt Disable

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Disable

**Bit 7 – LCRCE** LON CRC Error Interrupt Disable

**Bit 6 – LSFE** LON Short Frame Error Interrupt Disable

**Bit 5 – OVRE** Overrun Error Interrupt Disable

**Bit 1 – TXRDY** TXRDY Interrupt Disable

**Bit 0 – RXRDY** RXRDY Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.13 USART Interrupt Mask Register

**Name:** US_IMR
**Offset:** 0x0010
**Reset:** 0x0
**Property:** Read-only

For SPI specific configuration, see “USART Interrupt Mask Register (SPI_MODE)”.

For LIN specific configuration, see “USART Interrupt Mask Register (LIN_MODE)”.

For LON specific configuration, see “USART Interrupt Mask Register (LON_MODE)”.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | MANE |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CTSIC | DCDIC | DSRIC | RIIC |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| Access | | | | | | | | |
| Reset | | | 0 | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | 0 | 0 | 0 |

**Bit 24 – MANE** Manchester Error Interrupt Mask

**Bit 19 – CTSIC** Clear to Send Input Change Interrupt Mask

**Bit 18 – DCDIC** Data Carrier Detect Input Change Interrupt Mask

**Bit 17 – DSRIC** Data Set Ready Input Change Mask

**Bit 16 – RIIC** Ring Indicator Input Change Mask

**Bit 13 – NACK** Non Acknowledge Interrupt Mask

**Bit 10 – ITER** Max Number of Repetitions Reached Interrupt Mask

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Mask

**Bit 8 – TIMEOUT** Timeout Interrupt Mask

**Bit 7 – PARE** Parity Error Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 6 – FRAME** Framing Error Interrupt Mask

**Bit 5 – OVRE** Overrun Error Interrupt Mask

**Bit 2 – RXBRK** Receiver Break Interrupt Mask

**Bit 1 – TXRDY** TXRDY Interrupt Mask

**Bit 0 – RXRDY** RXRDY Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.14 USART Interrupt Mask Register (SPI_MODE)

**Name:** US_IMR (SPI_MODE)
**Offset:** 0x0010
**Reset:** 0x0
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | 0 | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | 0 | | | | 0 | 0 |

**Bit 19 – NSSE** NSS Line (Driving CTS Pin) Rising or Falling Edge Event Interrupt Mask

**Bit 10 – UNRE** SPI Underrun Error Interrupt Mask

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Mask

**Bit 5 – OVRE** Overrun Error Interrupt Mask

**Bit 1 – TXRDY** TXRDY Interrupt Mask

**Bit 0 – RXRDY** RXRDY Interrupt Mask

### 46.7.15 USART Interrupt Mask Register (LIN_MODE)

**Name:** US_IMR (LIN_MODE)
**Offset:** 0x0010
**Reset:** 0x0
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

Interrupt Mask

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 31 – LINHTE** LIN Header Timeout Error Interrupt Mask

**Bit 30 – LINSTE** LIN Synch Tolerance Error Interrupt Mask

**Bit 29 – LINSNRE** LIN Slave Not Responding Error Interrupt Mask

**Bit 28 – LINCE** LIN Checksum Error Interrupt Mask

**Bit 27 – LINIPE** LIN Identifier Parity Interrupt Mask

**Bit 26 – LINISFE** LIN Inconsistent Synch Field Error Interrupt Mask

**Bit 25 – LINBE** LIN Bus Error Interrupt Mask

**Bit 15 – LINTC** LIN Transfer Completed Interrupt Mask

**Bit 14 – LINID** LIN Identifier Sent or LIN Identifier Received Interrupt Mask

**Bit 13 – LINBK** LIN Break Sent or LIN Break Received

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 8 – TIMEOUT** Timeout Interrupt Mask

**Bit 7 – PARE** Parity Error Interrupt Mask

**Bit 6 – FRAME** Framing Error Interrupt Mask

**Bit 5 – OVRE** Overrun Error Interrupt Mask

**Bit 1 – TXRDY** TXRDY Interrupt Mask

**Bit 0 – RXRDY** RXRDY Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.16 USART Interrupt Mask Register (LON_MODE)

**Name:** US_IMR (LON_MODE)
**Offset:** 0x0010
**Reset:** 0x0
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0x9 in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | LBLOVFE | LRXD | LFET | LCOL | LTXD |
| Access | | | | | | | | |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LCRCE | LSFE | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 28 – LBLOVFE** LON Backlog Overflow Error Interrupt Mask

**Bit 27 – LRXD** LON Reception Done Interrupt Mask

**Bit 26 – LFET** LON Frame Early Termination Interrupt Mask

**Bit 25 – LCOL** LON Collision Interrupt Mask

**Bit 24 – LTXD** LON Transmission Done Interrupt Mask

**Bit 10 – UNRE** Underrun Error Interrupt Mask

**Bit 9 – TXEMPTY** TXEMPTY Interrupt Mask

**Bit 7 – LCRCE** LON CRC Error Interrupt Mask

**Bit 6 – LSFE** LON Short Frame Error Interrupt Mask

**Bit 5 – OVRE** Overrun Error Interrupt Mask

**Bit 1 – TXRDY** TXRDY Interrupt Mask

**Bit 0 – RXRDY** RXRDY Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.17 USART Channel Status Register

**Name:** US_CSR
**Offset:** 0x0014
**Reset:** 0x0
**Property:** Read-only

For SPI specific configuration, see "USART Channel Status Register (SPI_MODE)".

For LIN specific configuration, see "USART Channel Status Register (LIN_MODE)".

For LON specific configuration, see "USART Channel Status Register (LON_MODE)".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | MANERR |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CTS | DCD | DSR | RI | CTSIC | DCDIC | DSRIC | RIIC |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | NACK | | | ITER | TXEMPTY | TIMEOUT |
| Access | | | | | | | | |
| Reset | | | 0 | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | RXBRK | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | 0 | 0 | 0 |

**Bit 24 – MANERR** Manchester Error (cleared by writing a one to the bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No Manchester error has been detected since the last RSTSTA. |
| 1 | At least one Manchester error has been detected since the last RSTSTA. |

**Bit 23 – CTS** Image of CTS Input

| Value | Description |
|---|---|
| 0 | CTS input is driven low. |
| 1 | CTS input is driven high. |

**Bit 22 – DCD** Image of DCD Input

| Value | Description |
|---|---|
| 0 | DCD input is driven low. |
| 1 | DCD input is driven high. |

**Bit 21 – DSR** Image of DSR Input

| Value | Description |
|---|---|
| 0 | DSR input is driven low. |
| 1 | DSR input is driven high. |

**Bit 20 – RI** Image of RI Input

| Value | Description |
|---|---|
| 0 | RI input is driven low. |
| 1 | RI input is driven high. |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 19 – CTSIC** Clear to Send Input Change Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | No input change has been detected on the CTS pin since the last read of US_CSR. |
| 1 | At least one input change has been detected on the CTS pin since the last read of US_CSR. |

**Bit 18 – DCDIC** Data Carrier Detect Input Change Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | No input change has been detected on the DCD pin since the last read of US_CSR. |
| 1 | At least one input change has been detected on the DCD pin since the last read of US_CSR. |

**Bit 17 – DSRIC** Data Set Ready Input Change Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | No input change has been detected on the DSR pin since the last read of US_CSR. |
| 1 | At least one input change has been detected on the DSR pin since the last read of US_CSR. |

**Bit 16 – RIIC** Ring Indicator Input Change Flag (cleared on read)

| Value | Description |
|---|---|
| 0 | No input change has been detected on the RI pin since the last read of US_CSR. |
| 1 | At least one input change has been detected on the RI pin since the last read of US_CSR. |

**Bit 13 – NACK** Non Acknowledge Interrupt (cleared by writing a one to bit US_CR.RSTNACK)

| Value | Description |
|---|---|
| 0 | Non acknowledge has not been detected since the last RSTNACK. |
| 1 | At least one non acknowledge has been detected since the last RSTNACK. |

**Bit 10 – ITER** Max Number of Repetitions Reached (cleared by writing a one to bit US_CR.RSTIT)

| Value | Description |
|---|---|
| 0 | Maximum number of repetitions has not been reached since the last RSTIT. |
| 1 | Maximum number of repetitions has been reached since the last RSTIT. |

**Bit 9 – TXEMPTY** Transmitter Empty (cleared by writing US_THR)

| Value | Description |
|---|---|
| 0 | There are characters in either US_THR or the Transmit Shift Register, or the transmitter is disabled. |
| 1 | There are no characters in US_THR, nor in the Transmit Shift Register. |

**Bit 8 – TIMEOUT** Receiver Timeout (cleared by writing a one to bit US_CR.STTTO)

| Value | Description |
|---|---|
| 0 | There has not been a timeout since the last Start Timeout command (STTTO in US_CR) or the Timeout Register is 0. |
| 1 | There has been a timeout since the last Start Timeout command (STTTO in US_CR). |

**Bit 7 – PARE** Parity Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No parity error has been detected since the last RSTSTA. |
| 1 | At least one parity error has been detected since the last RSTSTA. |

**Bit 6 – FRAME** Framing Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No stop bit has been detected low since the last RSTSTA. |
| 1 | At least one stop bit has been detected low since the last RSTSTA. |

**Bit 5 – OVRE** Overrun Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA. |
| 1 | At least one overrun error has occurred since the last RSTSTA. |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 2 – RXBRK** Break Received/End of Break (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No break received or end of break detected since the last RSTSTA. |
| 1 | Break received or end of break detected since the last RSTSTA. |

**Bit 1 – TXRDY** Transmitter Ready (cleared by writing US_THR)

| Value | Description |
|---|---|
| 0 | A character is in the US_THR waiting to be transferred to the Transmit Shift Register, or an STTBRK command has been requested, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1. |
| 1 | There is no character in the US_THR. |

**Bit 0 – RXRDY** Receiver Ready (cleared by reading US_RHR)

| Value | Description |
|---|---|
| 0 | No complete character has been received since the last read of US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled. |
| 1 | At least one complete character has been received and US_RHR has not yet been read. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.18 USART Channel Status Register (SPI_MODE)

**Name:** US_CSR (SPI_MODE)
**Offset:** 0x0014
**Reset:** 0x0
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NSS | | | | NSSE | | | |
| Access | | | | | | | | |
| Reset | 0 | | | | 0 | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | | | 0 | | | | 0 | 0 |

**Bit 23 – NSS** Image of NSS Line

| Value | Description |
|---|---|
| 0 | NSS line is driven low (if NSSE = 1, falling edge occurred on NSS line). |
| 1 | NSS line is driven high (if NSSE = 1, rising edge occurred on NSS line). |

**Bit 19 – NSSE** NSS Line (Driving CTS Pin) Rising or Falling Edge Event (cleared on read)

| Value | Description |
|---|---|
| 0 | No NSS line event has been detected since the last read of US_CSR. |
| 1 | A rising or falling edge event has been detected on NSS line since the last read of US_CSR. |

**Bit 10 – UNRE** Underrun Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No SPI underrun error has occurred since the last RSTSTA. |
| 1 | At least one SPI underrun error has occurred since the last RSTSTA. |

**Bit 9 – TXEMPTY** Transmitter Empty (cleared by writing US_THR)

| Value | Description |
|---|---|
| 0 | There are characters in either US_THR or the Transmit Shift Register, or the transmitter is disabled. |
| 1 | There are no characters in US_THR, nor in the Transmit Shift Register. |

**Bit 5 – OVRE** Overrun Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA. |
| 1 | At least one overrun error has occurred since the last RSTSTA. |

**Bit 1 – TXRDY** Transmitter Ready (cleared by writing US_THR)

| Value | Description |
|---|---|
| 0 | A character is in the US_THR waiting to be transferred to the Transmit Shift Register or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1. |
| 1 | There is no character in the US_THR. |

**Bit 0 – RXRDY** Receiver Ready (cleared by reading US_RHR)

| Value | Description |
|---|---|
| 0 | No complete character has been received since the last read of US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled. |
| 1 | At least one complete character has been received and US_RHR has not yet been read. |

### 46.7.19 USART Channel Status Register (LIN_MODE)

**Name:** US_CSR (LIN_MODE)
**Offset:** 0x0014
**Reset:** 0x0
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | LINHTE | LINSTE | LINSNRE | LINCE | LINIPE | LINISFE | LINBE | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LINBLS | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LINTC | LINID | LINBK | | | | TXEMPTY | TIMEOUT |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 31 – LINHTE** LIN Header Timeout Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No LIN header timeout error has been detected since the last RSTSTA. |
| 1 | A LIN header timeout error has been detected since the last RSTSTA. |

**Bit 30 – LINSTE** LIN Synch Tolerance Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No LIN synch tolerance error has been detected since the last RSTSTA. |
| 1 | A LIN synch tolerance error has been detected since the last RSTSTA. |

**Bit 29 – LINSNRE** LIN Slave Not Responding Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No LIN slave not responding error has been detected since the last RSTSTA. |
| 1 | A LIN slave not responding error has been detected since the last RSTSTA. |

**Bit 28 – LINCE** LIN Checksum Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No LIN checksum error has been detected since the last RSTSTA. |
| 1 | A LIN checksum error has been detected since the last RSTSTA. |

**Bit 27 – LINIPE** LIN Identifier Parity Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No LIN identifier parity error has been detected since the last RSTSTA. |
| 1 | A LIN identifier parity error has been detected since the last RSTSTA. |

**Bit 26 – LINISFE** LIN Inconsistent Synch Field Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No LIN inconsistent synch field error has been detected since the last RSTSTA |

| Value | Description |
|---|---|
| 1 | The USART is configured as a slave node and a LIN Inconsistent synch field error has been detected since the last RSTSTA. |

**Bit 25 – LINBE** LIN Bit Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No bit error has been detected since the last RSTSTA. |
| 1 | A bit error has been detected since the last RSTSTA. |

**Bit 23 – LINBLS** LIN Bus Line Status

| Value | Description |
|---|---|
| 0 | LIN bus line is set to 0. |
| 1 | LIN bus line is set to 1. |

**Bit 15 – LINTC** LIN Transfer Completed (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | The USART is idle or a LIN transfer is ongoing. |
| 1 | A LIN transfer has been completed since the last RSTSTA. |

**Bit 14 – LINID** LIN Identifier Sent or LIN Identifier Received (cleared by writing a one to bit US_CR.RSTSTA)
If USART operates in LIN Master mode (USART_MODE = 0xA):
If USART operates in LIN Slave mode (USART_MODE = 0xB):

| Value | Description |
|---|---|
| 0 | No LIN identifier has been sent since the last RSTSTA. |
| 1 | At least one LIN identifier has been sent since the last RSTSTA. |
| 0 | No LIN identifier has been received since the last RSTSTA. |
| 1 | At least one LIN identifier has been received since the last RSTSTA |

**Bit 13 – LINBK** LIN Break Sent or LIN Break Received (cleared by writing a one to bit US_CR.RSTSTA)
Applicable if USART operates in LIN master mode (USART_MODE = 0xA):
If USART operates in LIN Slave mode (USART_MODE = 0xB):

| Value | Description |
|---|---|
| 0 | No LIN break has been sent since the last RSTSTA. |
| 1 | At least one LIN break has been sent since the last RSTSTA |
| 0 | No LIN break has received sent since the last RSTSTA. |
| 1 | At least one LIN break has been received since the last RSTSTA. |

**Bit 9 – TXEMPTY** Transmitter Empty (cleared by writing US_THR)

| Value | Description |
|---|---|
| 0 | There are characters in either US_THR or the Transmit Shift Register, or the transmitter is disabled. |
| 1 | There are no characters in US_THR, nor in the Transmit Shift Register. |

**Bit 8 – TIMEOUT** Receiver Timeout (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | There has not been a timeout since the last start timeout command (STTTO in US_CR) or the Timeout Register is 0. |
| 1 | There has been a timeout since the last start timeout command (STTTO in US_CR). |

**Bit 7 – PARE** Parity Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No parity error has been detected since the last RSTSTA. |
| 1 | At least one parity error has been detected since the last RSTSTA. |

**Bit 6 – FRAME** Framing Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No stop bit has been detected low since the last RSTSTA. |
| 1 | At least one stop bit has been detected low since the last RSTSTA. |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 5 – OVRE** Overrun Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA. |
| 1 | At least one overrun error has occurred since the last RSTSTA. |

**Bit 1 – TXRDY** Transmitter Ready (cleared by writing US_THR)

| Value | Description |
|---|---|
| 0 | A character is in the US_THR waiting to be transferred to the Transmit Shift Register or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1. |
| 1 | There is no character in the US_THR. |

**Bit 0 – RXRDY** Receiver Ready (cleared by reading US_THR)

| Value | Description |
|---|---|
| 0 | No complete character has been received since the last read of US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled. |
| 1 | At least one complete character has been received and US_RHR has not yet been read. |

### 46.7.20 USART Channel Status Register (LON_MODE)

**Name:** US_CSR (LON_MODE)
**Offset:** 0x0014
**Reset:** 0x0
**Property:** Read-only

This configuration is relevant only if USART_MODE = 0x9 in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | LBLOVFE | LRXD | LFET | LCOL | LTXD |
| Access | | | | | | | | |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | UNRE | TXEMPTY | |
| Access | | | | | | | | |
| Reset | | | | | | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LCRCE | LSFE | OVRE | | | | TXRDY | RXRDY |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 28 – LBLOVFE** LON Backlog Overflow Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No backlog overflow error occurred since the last RSTSTA. |
| 1 | At least one backlog error overflow occurred since the last RSTSTA. |

**Bit 27 – LRXD** LON Reception End Flag (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | Reception on going or no reception occurred since the last RSTSTA. |
| 1 | At least one reception has been performed since the last RSTSTA. |

**Bit 26 – LFET** LON Frame Early Termination (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No frame has been terminated early due to collision detection since the last RSTSTA. |
| 1 | At least one transmission has been terminated due to collision detection since the last RSTSTA. (This stops the DMA until reset with RSTSTA bit). |

**Bit 25 – LCOL** LON Collision Detected Flag (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No collision occurred while transmitting since the last RSTSTA. |
| 1 | At least one collision occurred while transmitting since the last RSTSTA. |

**Bit 24 – LTXD** LON Transmission End Flag (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | Transmission on going or no transmission occurred since the last RSTSTA. |
| 1 | At least one transmission has been performed since the last RSTSTA. |

**Bit 10 – UNRE** Underrun Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No LON underrun error has occurred since the last RSTSTA. |
| 1 | At least one LON underrun error has occurred since the last RSTSTA. |

**Bit 9 – TXEMPTY** Transmitter Empty (cleared by writing US_THR)

| Value | Description |
|---|---|
| 0 | There are characters in either US_THR or the Transmit Shift Register, or the transmitter is disabled. |
| 1 | There are no characters in US_THR, nor in the Transmit Shift Register. |

**Bit 7 – LCRCE** LON CRC Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No CRC error has been detected since the last RSTSTA. |
| 1 | At least one CRC error has been detected since the last RSTSTA. |

**Bit 6 – LSFE** LON Short Frame Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No short frame received since the last RSTSTA. |
| 1 | At least one short frame received since the last RSTSTA. |

**Bit 5 – OVRE** Overrun Error (cleared by writing a one to bit US_CR.RSTSTA)

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA. |
| 1 | At least one overrun error has occurred since the last RSTSTA. |

**Bit 1 – TXRDY** Transmitter Ready (cleared by writing US_THR)

| Value | Description |
|---|---|
| 0 | A character is in the US_THR waiting to be transferred to the Transmit Shift Register or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1. |
| 1 | There is no character in the US_THR. |

**Bit 0 – RXRDY** Receiver Ready (cleared by reading US_RHR)

| Value | Description |
|---|---|
| 0 | No complete character has been received since the last read of US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled. |
| 1 | At least one complete character has been received and US_RHR has not yet been read. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.21 USART Receive Holding Register

**Name:** US_RHR
**Offset:** 0x0018
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RXSYNH | | | | | | | RXCHR[8] |
| Access | | | | | | | | |
| Reset | 0 | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – RXSYNH** Received Sync

| Value | Description |
|---|---|
| 0 | Last character received is a data. |
| 1 | Last character received is a command. |

**Bits 8:0 – RXCHR[8:0]** Received Character
Last character received if RXRDY is set.

### 46.7.22 USART Transmit Holding Register

**Name:** US_THR
**Offset:** 0x001C
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | TXSYNH | | | | | | | TXCHR[8] |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | TXCHR[7:0] | | | | | | | |
| Reset | | | | | | | | |

**Bit 15 – TXSYNH** Sync Field to be Transmitted

| Value | Description |
|---|---|
| 0 | The next character sent is encoded as a data. Start frame delimiter is DATA SYNC. |
| 1 | The next character sent is encoded as a command. Start frame delimiter is COMMAND SYNC. |

**Bits 8:0 – TXCHR[8:0]** Character to be Transmitted
Next character to be transmitted after the current character if TXRDY is not set.

### 46.7.23 USART Baud Rate Generator Register

**Name:** US_BRGR
**Offset:** 0x0020
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | FP[2:0] | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | CD[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | CD[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 18:16 – FP[2:0]** Fractional Part

**WARNING** When the value of field FP is greater than 0, the SCK (oversampling clock) generates nonconstant duty cycles. The SCK high duration is increased by “selected clock” period from time to time. The duty cycle depends on the value of the CD field.

| Value | Description |
|---|---|
| 0 | Fractional divider is disabled. |
| 1–7 | Baud rate resolution, defined by FP × 1/8. |

**Bits 15:0 – CD[15:0]** Clock Divider

| CD | USART_MODE ≠ ISO7816 | | | USART_MODE = ISO7816 |
|---|---|---|---|---|
| | SYNC = 0 | | SYNC = 1 or USART_MODE = SPI (Master or Slave) | |
| | OVER = 0 | OVER = 1 | | |
| 0 | Baud Rate Clock Disabled | | | |
| 1 to 65535 | CD = Selected Clock / (16 × Baud Rate) | CD = Selected Clock / (8 × Baud Rate) | CD = Selected Clock / Baud Rate | CD = Selected Clock / (FI_DI_RATIO × Baud Rate) |

### 46.7.24 USART Receiver Timeout Register

**Name:** US_RTOR
**Offset:** 0x0024
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TO[16] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TO[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TO[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 16:0 – TO[16:0]** Timeout Value

| Value | Description |
|---|---|
| 0 | The receiver timeout is disabled. |
| 1-65535 | The receiver timeout is enabled and TO is Timeout Delay / Bit Period. |
| 1-131071 | The receiver timeout is enabled and TO is Timeout Delay / Bit Period. |

### 46.7.25 USART Transmitter Timeguard Register

**Name:** US_TTGR
**Offset:** 0x0028
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

For LON specific configuration, see "USART Transmitter Timeguard Register (LON_MODE)".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | TG[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – TG[7:0]** Timeguard Value

| Value | Description |
|---|---|
| 0 | The transmitter timeguard is disabled. |
| 1-255 | The transmitter timeguard is enabled and TG is Timeguard Delay / Bit Period. |

### 46.7.26 USART Transmitter Timeguard Register (LON_MODE)

**Name:** US_TTGR (LON_MODE)
**Offset:** 0x0028
**Reset:** 0x0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PCYCLE[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PCYCLE[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PCYCLE[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – PCYCLE[23:0]** LON PCYCLE Length

| Value | Description |
|---|---|
| 1-16777215 | LON PCYCLE length in $t_{bit}$. |

### 46.7.27 USART FI DI RATIO Register

**Name:** US_FIDI
**Offset:** 0x0040
**Reset:** 0x174
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

For LON specific configuration, see "USART Transmitter Timeguard Register (LON_MODE)".

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FI_DI_RATIO[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FI_DI_RATIO[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |

**Bits 15:0 – FI_DI_RATIO[15:0]** FI Over DI Ratio Value

| Value | Description |
|---|---|
| 0 | If ISO7816 mode is selected, the baud rate generator generates no signal. |
| 1-2 | Do not use. |
| 3-2047 | If ISO7816 mode is selected, the baud rate is the clock provided on SCK divided by FI_DI_RATIO. |

### 46.7.28 USART FI DI RATIO Register (LON_MODE)

**Name:** US_FIDI (LON_MODE)
**Offset:** 0x0040
**Reset:** 0x174
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BETA2[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BETA2[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BETA2[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |

**Bits 23:0 – BETA2[23:0]** LON BETA2 Length

| Value | Description |
|---|---|
| 1–16777215 | LON BETA2 length in $t_{bit}$. |

### 46.7.29 USART Number of Errors Register

**Name:** US_NER
**Offset:** 0x0044
**Reset:** 0x0
**Property:** Read-only

This register is relevant only if USART_MODE = 0x4 or 0x6 in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NB_ERRORS[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – NB_ERRORS[7:0]** Number of Errors
Total number of errors that occurred during an ISO7816 transfer. This register automatically clears when read.

### 46.7.30 USART IrDA Filter Register

**Name:** US_IF
**Offset:** 0x004C
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x8 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IRDA_FILTER[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – IRDA_FILTER[7:0]** IrDA Filter
The IRDA_FILTER value must be defined to meet the following criteria:
$t_{peripheral\ clock} \times (IRDA_FILTER + 3) < 1.41\ \mu s$

### 46.7.31 USART Manchester Configuration Register

**Name:** US_MAN
**Offset:** 0x0050
**Reset:** 0xB30011004
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RXIDLEV | DRIFT | ONE | RX_MPOL | | | RX_PP[1:0] | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 1 | 1 | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | RX_PL[3:0] | | | |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | TX_MPOL | | | TX_PP[1:0] | |
| Access | | | | | | | | |
| Reset | | | | 1 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TX_PL[3:0] | | | |
| Access | | | | | | | | |
| Reset | | | | | 0 | 1 | 0 | 0 |

**Bit 31 – RXIDLEV** Receiver Idle Value

| Value | Description |
|---|---|
| 0 | Receiver line idle value is 0. |
| 1 | Receiver line idle value is 1. |

**Bit 30 – DRIFT** Drift Compensation

| Value | Description |
|---|---|
| 0 | The USART cannot recover from an important clock drift |
| 1 | The USART can recover from clock drift. The 16X clock mode must be enabled. |

**Bit 29 – ONE** Must Be Set to 1
Bit 29 must always be set to 1 when programming the US_MAN register.

**Bit 28 – RX_MPOL** Receiver Manchester Polarity

| Value | Description |
|---|---|
| 0 | Logic zero is coded as a zero-to-one transition, Logic one is coded as a one-to-zero transition. |
| 1 | Logic zero is coded as a one-to-zero transition, Logic one is coded as a zero-to-one transition. |

**Bits 25:24 – RX_PP[1:0]** Receiver Preamble Pattern detected
The following values assume that RX_MPOL field is not set:

| Value | Name | Description |
|---|---|---|
| 00 | ALL_ONE | The preamble is composed of ‘1’s |
| 01 | ALL_ZERO | The preamble is composed of ‘0’s |
| 10 | ZERO_ONE | The preamble is composed of ‘01’s |
| 11 | ONE_ZERO | The preamble is composed of ‘10’s |

**Bits 19:16 – RX_PL[3:0]** Receiver Preamble Length

| Value | Description |
|---|---|
| 0 | The receiver preamble pattern detection is disabled |
| 1–15 | The detected preamble length is RX_PL × Bit Period |

**Bit 12 – TX_MPOL** Transmitter Manchester Polarity

| Value | Description |
|---|---|
| 0 | Logic zero is coded as a zero-to-one transition, Logic one is coded as a one-to-zero transition. |
| 1 | Logic zero is coded as a one-to-zero transition, Logic one is coded as a zero-to-one transition. |

**Bits 9:8 – TX_PP[1:0]** Transmitter Preamble Pattern

The following values assume that TX_MPOL field is not set:

| Value | Name | Description |
|---|---|---|
| 0 | ALL_ONE | The preamble is composed of ‘1’s |
| 1 | ALL_ZERO | The preamble is composed of ‘0’s |
| 2 | ZERO_ONE | The preamble is composed of ‘01’s |
| 3 | ONE_ZERO | The preamble is composed of ‘10’s |

**Bits 3:0 – TX_PL[3:0]** Transmitter Preamble Length

| Value | Description |
|---|---|
| 0 | The transmitter preamble pattern generation is disabled |
| 1–15 | The preamble length is TX_PL × Bit Period |

### 46.7.32 USART LIN Mode Register

**Name:** US_LINMR
**Offset:** 0x0054
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 17 – SYNCDIS** Synchronization Disable

| Value | Description |
|---|---|
| 0 | The synchronization procedure is performed in LIN slave node configuration. |
| 1 | The synchronization procedure is not performed in LIN slave node configuration. |

**Bit 16 – PDCM** DMAC Mode

| Value | Description |
|---|---|
| 0 | The LIN mode register US_LINMR is not written by the DMAC. |
| 1 | The LIN mode register US_LINMR (excepting that flag) is written by the DMAC. |

**Bits 15:8 – DLC[7:0]** Data Length Control

| Value | Description |
|---|---|
| 0-255 | Defines the response data length if DLM = 0,in that case the response data length is equal to DLC+1 bytes. |

**Bit 7 – WKUPTYP** Wakeup Signal Type

| Value | Description |
|---|---|
| 0 | Setting the bit LINWKUP in US_CR sends a LIN 2.0 wakeup signal. |
| 1 | Setting the bit LINWKUP in US_CR sends a LIN 1.3 wakeup signal. |

**Bit 6 – FSDIS** Frame Slot Mode Disable

| Value | Description |
|---|---|
| 0 | The Frame Slot mode is enabled. |
| 1 | The Frame Slot mode is disabled. |

**Bit 5 – DLM** Data Length Mode

| Value | Description |
|---|---|
| 0 | The response data length is defined by field DLC of this register. |

| Value | Description |
|---|---|
| 1 | The response data length is defined by bits 5 and 6 of the identifier (IDCHR in US_LINIR). |

**Bit 4 – CHKTYP** Checksum Type

| Value | Description |
|---|---|
| 0 | LIN 2.0 “enhanced” checksum |
| 1 | LIN 1.3 “classic” checksum |

**Bit 3 – CHKDIS** Checksum Disable

| Value | Description |
|---|---|
| 0 | In master node configuration, the checksum is computed and sent automatically. In slave node configuration, the checksum is checked automatically. |
| 1 | Whatever the node configuration is, the checksum is not computed/sent and it is not checked. |

**Bit 2 – PARDIS** Parity Disable

| Value | Description |
|---|---|
| 0 | In master node configuration, the identifier parity is computed and sent automatically. In master node and slave node configuration, the parity is checked automatically. |
| 1 | Whatever the node configuration is, the Identifier parity is not computed/sent and it is not checked. |

**Bits 1:0 – NACT[1:0]** LIN Node Action

Values which are not listed in the table must be considered as “reserved”.

| Value | Name | Description |
|---|---|---|
| 00 | PUBLISH | The USART transmits the response. |
| 01 | SUBSCRIBE | The USART receives the response. |
| 10 | IGNORE | The USART does not transmit and does not receive the response. |

### 46.7.33 USART LIN Identifier Register

**Name:** US_LINIR
**Offset:** 0x0058
**Reset:** 0x0
**Property:** Read/Write(1)

This register is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | IDCHR[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – IDCHR[7:0]** Identifier Character
If USART_MODE = 0xA (master node configuration):
IDCHR is Read/Write and its value is the identifier character to be transmitted.
If USART_MODE = 0xB (slave node configuration):
IDCHR is Read-only and its value is the last identifier character that has been received.

### 46.7.34 USART LIN Baud Rate Register

**Name:** US_LINBRR
**Offset:** 0x005C
**Reset:** 0x0
**Property:** Read-only

This register is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

Returns the baud rate value after the synchronization process completion.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | LINFP[2:0] | | |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | LINCD[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | LINCD[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 18:16 – LINFP[2:0]** Fractional Part after Synchronization

**Bits 15:0 – LINCD[15:0]** Clock Divider after Synchronization

### 46.7.35 USART LON Mode Register

**Name:** US_LONMR
**Offset:** 0x0060
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | EOFS[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | LCDS | DMAM | CDTAIL | TCOL | COLDET | COMMT |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – EOFS[7:0]** End of Frame Condition Size

| Value | Description |
|---|---|
| 0-255 | Define the minimum transitionless time for the IP to detect a LON end of frame condition. $t_{eof}$ = (EOFS + 1) × $t_{clock}$ × 8 × (2 - OVER) |

**Bit 5 – LCDS** LON Collision Detection Source

| Value | Description |
|---|---|
| 0 | LON collision detection source is external. |
| 1 | LON collision detection source is internal. |

**Bit 4 – DMAM** LON DMA Mode

| Value | Description |
|---|---|
| 0 | The LON data length register US_LONDL is not written by the DMA. |
| 1 | The LON data length register US_LONDL is written by the DMA. |

**Bit 3 – CDTAIL** LON Collision Detection on Frame Tail

| Value | Description |
|---|---|
| 0 | Detect collisions after CRC has been sent but prior end of transmission in LON comm_type = 1 mode. |
| 1 | Ignore collisions after CRC has been sent but prior end of transmission in LON comm_type = 1 mode. |

**Bit 2 – TCOL** Terminate Frame upon Collision Notification

| Value | Description |
|---|---|
| 0 | Do not terminate the frame in LON comm_type = 1 mode upon collision detection. |
| 1 | Terminate the frame in LON comm_type = 1 mode upon collision detection if possible. |

**Bit 1 – COLDET** LON Collision Detection Feature

| Value | Description |
|---|---|
| 0 | LON collision detection feature disabled. |
| 1 | LON collision detection feature enabled. |

**Bit 0 – COMMT** LON comm_type Parameter Value

| Value | Description |
|---|---|
| 0 | LON comm_type = 1 mode. |
| 1 | LON comm_type = 2 mode. |

### 46.7.36 USART LON Preamble Register

**Name:** US_LONPR
**Offset:** 0x0064
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | LONPL[13:8] | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | LONPL[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 13:0 – LONPL[13:0]** LON Preamble Length

| Value | Description |
|---|---|
| 1-16383 | LON preamble length in $t_{bit}$ (without byte-sync). |

### 46.7.37 USART LON Data Length Register

**Name:** US_LONDL
**Offset:** 0x0068
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | LONDL[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – LONDL[7:0]** LON Data Length

| Value | Description |
|---|---|
| 0-255 | LON data length is LONDL+1 bytes. |

### 46.7.38 USART LON L2HDR Register

**Name:** US_LONL2HDR
**Offset:** 0x006C
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PB | ALTP | BLI[5:0] | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – PB** LON Priority Bit

| Value | Description |
|---|---|
| 0 | LON priority bit reset. |
| 1 | LON priority bit set. |

**Bit 6 – ALTP** LON Alternate Path Bit

| Value | Description |
|---|---|
| 0 | LON alternate path bit reset. |
| 1 | LON alternate path bit set. |

**Bits 5:0 – BLI[5:0]** LON Backlog Increment

| Value | Description |
|---|---|
| 0-63 | LON backlog increment to be generated as a result of delivering the LON frame. |

### 46.7.39 USART LON Backlog Register

**Name:** US_LONBL
**Offset:** 0x0070
**Reset:** 0x0
**Property:** Read

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | LONBL[5:0] | | | | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – LONBL[5:0]** LON Node Backlog Value

| Value | Description |
|---|---|
| 1-63 | LON node backlog value. |

### 46.7.40 USART LON Beta1 Tx Register

**Name:** US_LONB1TX
**Offset:** 0x0074
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BETA1TX[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BETA1TX[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BETA1TX[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – BETA1TX[23:0]** LON Beta1 Length after Transmission

| Value | Description |
|---|---|
| 1–16777215 | LON beta1 length after transmission in $t_{bit}$. |

### 46.7.41 USART LON Beta1 Rx Register

**Name:** US_LONB1RX
**Offset:** 0x0078
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | BETA1RX[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BETA1RX[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BETA1RX[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – BETA1RX[23:0]** LON Beta1 Length after Reception

| Value | Description |
|---|---|
| 1-16777215 | LON beta1 length after reception in $t_{bit}$. |

### 46.7.42 USART LON Priority Register

**Name:** US_LONPRIO
**Offset:** 0x007C
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | NPS[6:0] | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | PSNB[6:0] | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 14:8 – NPS[6:0]** LON Node Priority Slot

| Value | Description |
|---|---|
| 0-127 | Node priority slot. |

**Bits 6:0 – PSNB[6:0]** LON Priority Slot Number

| Value | Description |
|---|---|
| 0-127 | Number of priority slots in the LON network configuration. |

### 46.7.43 USART LON IDT Tx Register

**Name:** US_IDTTX
**Offset:** 0x0080
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IDTTX[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IDTTX[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDTTX[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – IDTTX[23:0]** LON Indeterminate Time after Transmission (comm_type = 1 mode only)

| Value | Description |
|---|---|
| 0–16777215 | LON indeterminate time after transmission in $t_{bit}$. |

### 46.7.44 USART LON IDT Rx Register

**Name:** US_IDTRX
**Offset:** 0x0084
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IDTRX[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IDTRX[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDTRX[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – IDTRX[23:0]** LON Indeterminate Time after Reception (comm_type = 1 mode only)

| Value | Description |
|---|---|
| 0–16777215 | LON indeterminate time after reception in $t_{bit}$. |

### 46.7.45 USART IC DIFF Register

**Name:** US_ICDIFF
**Offset:** 0x0088
**Reset:** 0x0
**Property:** Read/Write

This register is relevant only if USART_MODE = 0x9 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | ICDIFF[3:0] | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – ICDIFF[3:0]** IC Differentiator Number

### 46.7.46 USART Write Protection Mode Register

**Name:** US_WPMR
**Offset:** 0x00E4
**Reset:** 0x0
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | WPKEY[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | WPKEY[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | WPEN |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x555341 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Section 7.12 “Register Write Protection” for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x555341 (“USA” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x555341 (“USA” in ASCII). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 46.7.47 USART Write Protection Status Register

**Name:** US_WPSR
**Offset:** 0x00E8
**Reset:** 0x0
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the US_WPSR. |
| 1 | A write protection violation has occurred since the last read of the US_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 47. Universal Asynchronous Receiver Transmitter (UART)

## 47.1 Description

The Universal Asynchronous Receiver Transmitter (UART) features a two-pin UART that can be used for communication and trace purposes and offers an ideal medium for in-situ programming solutions.

Moreover, the association with a DMA controller permits packet handling for these tasks with processor time reduced to a minimum.

## 47.2 Embedded Characteristics

- Two-pin UART
  - Independent Receiver and Transmitter with a Common Programmable Baud Rate Generator
  - Baud Rate can be Driven by Processor-Independent Source Clock
  - Even, Odd, Mark or Space Parity Generation
  - Parity, Framing and Overrun Error Detection
  - Automatic Echo, Local Loopback and Remote Loopback Channel Modes
  - Digital Filter on Receive Line
  - Interrupt Generation
  - Support for Two DMA Channels with Connection to Receiver and Transmitter
  - Supports Asynchronous Partial Wake-up on Receive Line Activity (SleepWalking)
  - Comparison Function on Received Character
  - Register Write Protection

## 47.3 Block Diagram

**Figure 47-1. UART Block Diagram**

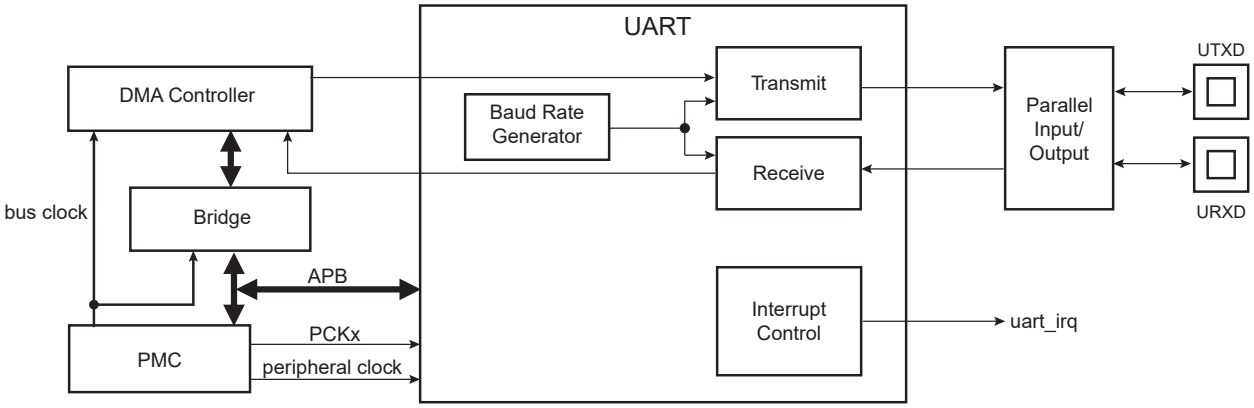



**Table 47-1. UART Pin Description**

| Pin Name | Description | Type |
|---|---|---|
| URXD | UART Receive Data | Input |
| UTXD | UART Trasnmit Data | Output |

## 47.4 Product Dependencies

### 47.4.1 I/O Lines

The UART pins are multiplexed with PIO lines. The user must first configure the corresponding PIO Controller to enable I/O line operations of the UART.

### 47.4.2 Power Management

The UART clock can be controlled through the Power Management Controller (PMC). In this case, the user must first configure the PMC to enable the UART clock. Usually, the peripheral identifier used for this purpose is 1.

In SleepWalking mode (asynchronous partial wake-up), the PMC must be configured to enable SleepWalking for the UART in the Sleepwalking Enable Register (PMC_SLPWK_ER). Depending on the instructions (requests) provided by the UART to the PMC, the system clock may or may not be automatically provided to the UART.

### 47.4.3 Interrupt Sources

The UART interrupt line is connected to one of the interrupt sources of the Interrupt Controller. Interrupt handling requires programming of the Interrupt Controller before configuring the UART.

## 47.5 Functional Description

The UART operates in Asynchronous mode only and supports only 8-bit character handling (with parity). It has no clock pin.

The UART is made up of a receiver and a transmitter that operate independently, and a common baud rate generator. Receiver timeout and transmitter time guard are not implemented. However, all the implemented features are compatible with those of a standard USART.

### 47.5.1 Baud Rate Generator

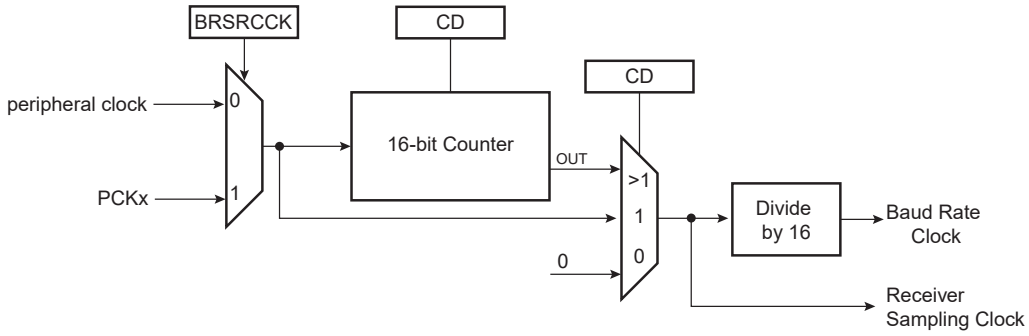
The baud rate generator provides the bit period clock named baud rate clock to both the receiver and the transmitter.

The baud rate clock is the peripheral clock divided by 16 times the clock divisor (CD) value written in the Baud Rate Generator register (UART_BRGR). If UART_BRGR is set to 0, the baud rate clock is disabled and the UART remains inactive. The maximum allowable baud rate is peripheral clock or PMC PCK (PCK) divided by 16. The minimum allowable baud rate is peripheral clock or PCK divided by (16 x 65536).The clock source driving the baud rate generator (peripheral clock or PCK) can be selected by writing the bit BRSRCCK in UART_MR.

If PCK is selected, the baud rate is independent of the processor/bus clock. Thus the processor clock can be changed while UART is enabled. The processor clock frequency changes must be performed only by programming the field PRES in PMC_MCKR (see "Power Management Controller (PMC)"). Other methods to modify the processor/bus clock frequency (PLL multiplier, etc.) are forbidden when UART is enabled.

The peripheral clock frequency must be at least three times higher than PCK.

**Figure 47-2. Baud Rate Generator**



### 47.5.2 Receiver

#### 47.5.2.1 Receiver Reset, Enable and Disable

After device reset, the UART receiver is disabled and must be enabled before being used. The receiver can be enabled by writing the Control Register (UART_CR) with the bit RXEN at 1. At this command, the receiver starts looking for a start bit.

The programmer can disable the receiver by writing UART_CR with the bit RXDIS at 1. If the receiver is waiting for a start bit, it is immediately stopped. However, if the receiver has already detected a start bit and is receiving the data, it waits for the stop bit before actually stopping its operation.

The receiver can be put in reset state by writing UART_CR with the bit RSTRX at 1. In this case, the receiver immediately stops its current operations and is disabled, whatever its current state. If RSTRX is applied when data is being processed, this data is lost.

#### 47.5.2.2 Start Detection and Data Sampling

The UART only supports asynchronous operations, and this affects only its receiver. The UART receiver detects the start of a received character by sampling the URXD signal until it detects a valid start bit. A low level (space) on URXD is interpreted as a valid start bit if it is detected for more than seven cycles of the sampling clock, which is 16 times the baud rate. Hence, a space that is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the URXD at the theoretical midpoint of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the bit sampling point is eight cycles (0.5-bit period) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after detecting the falling edge of the start bit.

Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

**Figure 47-3. Start Bit Detection**

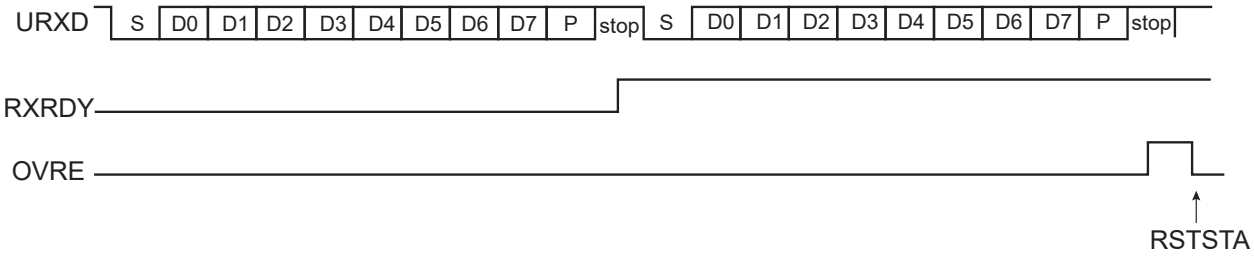
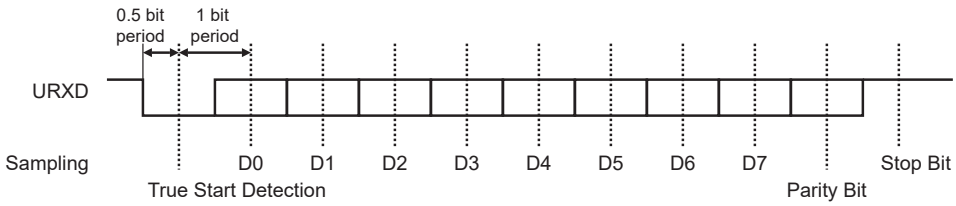



**Figure 47-4. Character Reception**

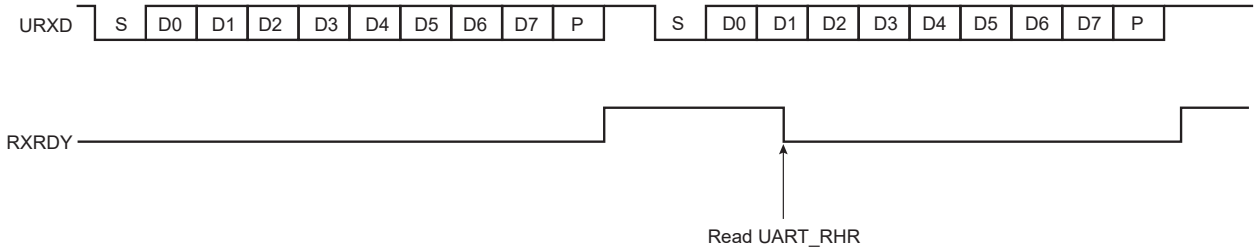
Example: 8-bit, parity enabled 1 stop




#### 47.5.2.3 Receiver Ready

When a complete character is received, it is transferred to the Receive Holding Register (UART_RHR) and the RXRDY status bit in the Status Register (UART_SR) is set. The bit RXRDY is automatically cleared when UART_RHR is read.

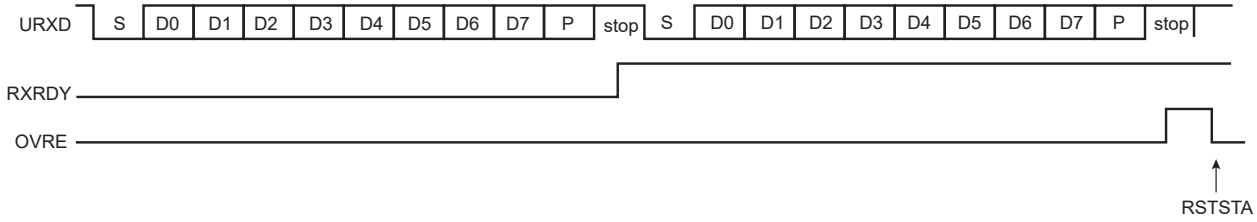
**Figure 47-5. Receiver Ready**




### 47.5.2.4 Receiver Overrun

The OVRE status bit in UART_SR is set if UART_RHR has not been read by the software (or the DMA Controller) since the last transfer, the RXRDY bit is still set and a new character is received. OVRE is cleared when the software writes a 1 to the bit RSTSTA (Reset Status) in UART_CR.

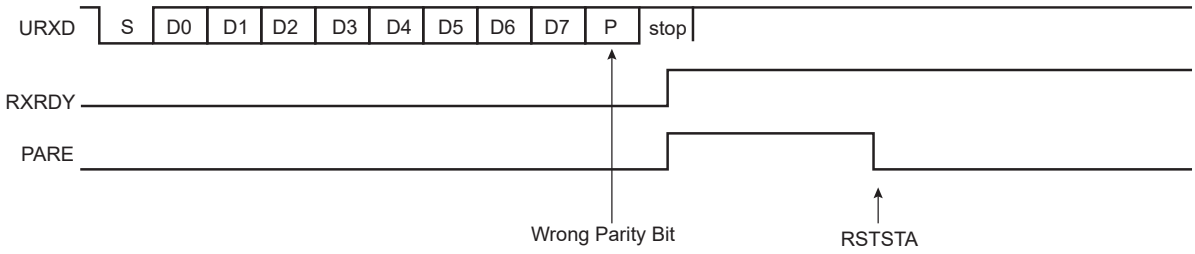
**Figure 47-6. Receiver Overrun**




### 47.5.2.5 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR in the Mode Register (UART_MR). It then compares the result with the received parity bit. If different, the parity error bit PARE in UART_SR is set at the same time RXRDY is set. The parity bit is cleared when UART_CR is written with the bit RSTSTA (Reset Status) at 1. If a new character is received before the reset status command is written, the PARE bit remains at 1.

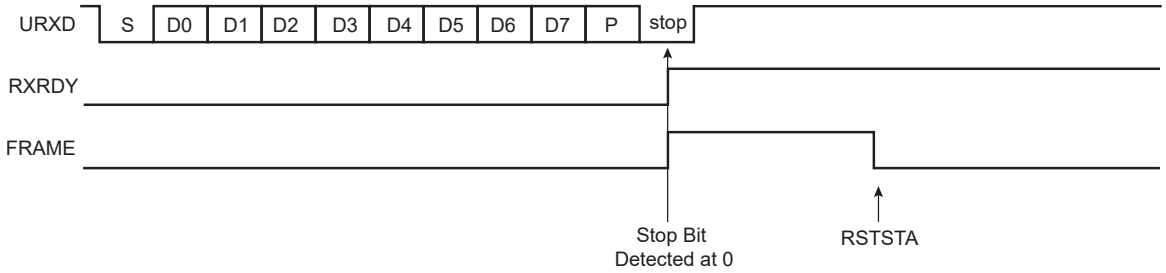
**Figure 47-7. Parity Error**




### 47.5.2.6 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the FRAME (Framing Error) bit in UART_SR is set at the same time the RXRDY bit is set. The FRAME bit remains high until the Control Register (UART_CR) is written with the bit RSTSTA at 1.

**Figure 47-8. Receiver Framing Error**



#### 47.5.2.7 Receiver Digital Filter

The UART embeds a digital filter on the receive line. It is disabled by default and can be enabled by writing a logical 1 in the FILTER bit of UART_MR. When enabled, the receive line is sampled using the 16x bit clock and a three-sample filter (majority 2 over 3) determines the value of the line.

### 47.5.3 Transmitter

#### 47.5.3.1 Transmitter Reset, Enable and Disable

After device reset, the UART transmitter is disabled and must be enabled before being used. The transmitter is enabled by writing UART_CR with the bit TXEN at 1. From this command, the transmitter waits for a character to be written in the Transmit Holding Register (UART_THR) before actually starting the transmission.

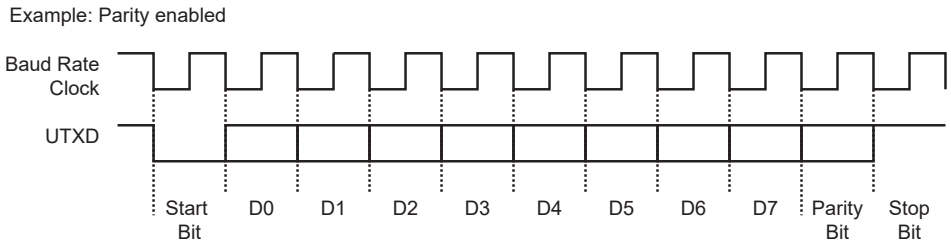
The programmer can disable the transmitter by writing UART_CR with the bit TXDIS at 1. If the transmitter is not operating, it is immediately stopped. However, if a character is being processed into the internal shift register and/or a character has been written in the UART_THR, the characters are completed before the transmitter is actually stopped.

The programmer can also put the transmitter in its reset state by writing the UART_CR with the bit RSTTX at 1. This immediately stops the transmitter, whether or not it is processing characters.

#### 47.5.3.2 Transmit Format

The UART transmitter drives the pin UTXD at the baud rate clock speed. The line is driven depending on the format defined in UART_MR and the data stored in the internal shift register. One start bit at level 0, then the 8 data bits, from the lowest to the highest bit, one optional parity bit and one stop bit at 1 are consecutively shifted out as shown in the following figure. The field PARE in UART_MR defines whether or not a parity bit is shifted out. When a parity bit is enabled, it can be selected between an odd parity, an even parity, or a fixed space or mark bit.

**Figure 47-9. Character Transmission**



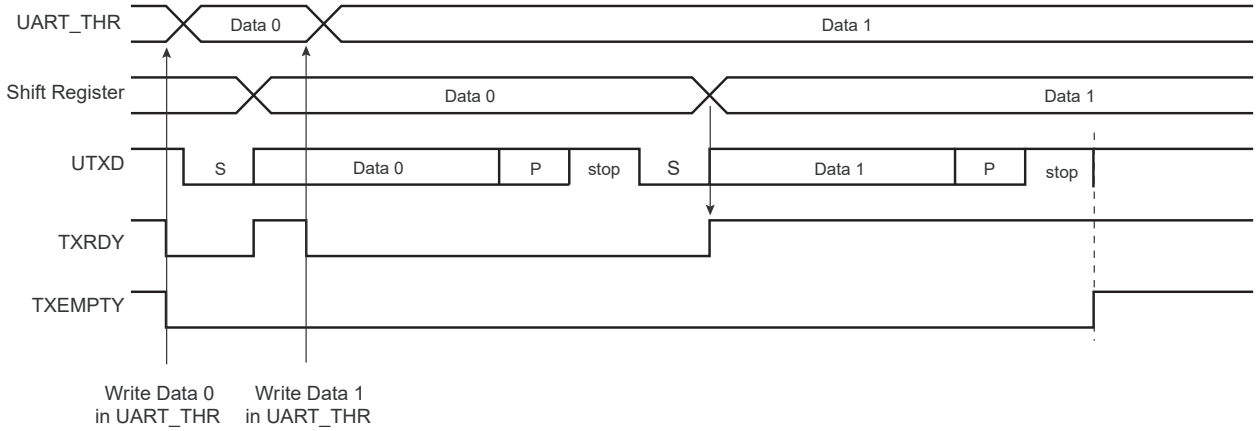

#### 47.5.3.3 Transmitter Control

When the transmitter is enabled, the bit TXRDY (Transmitter Ready) is set in UART_SR. The transmission starts when the programmer writes in the UART_THR, and after the written character is transferred from UART_THR to the internal shift register. The TXRDY bit remains high until a second character is written in UART_THR. As soon as the first character is completed, the last character written in UART_THR is transferred into the internal shift register and TXRDY rises again, showing that the holding register is empty.

When both the internal shift register and UART_THR are empty, i.e., all the characters written in UART_THR have been processed, the TXEMPTY bit rises after the last stop bit has been completed.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 47-10. Transmitter Control**




### 47.5.4 DMA Support

Both the receiver and the transmitter of the UART are connected to a DMA Controller (DMAC) channel.

The DMA Controller channels are programmed via registers that are mapped within the DMAC user interface.

### 47.5.5 Comparison Function on Received Character

When a comparison is performed on a received character, the result of the comparison is reported on the CMP flag in UART_SR when UART_RHR is loaded with the new received character. The CMP flag is cleared by writing a one to the RSTSTA bit in UART_CR.

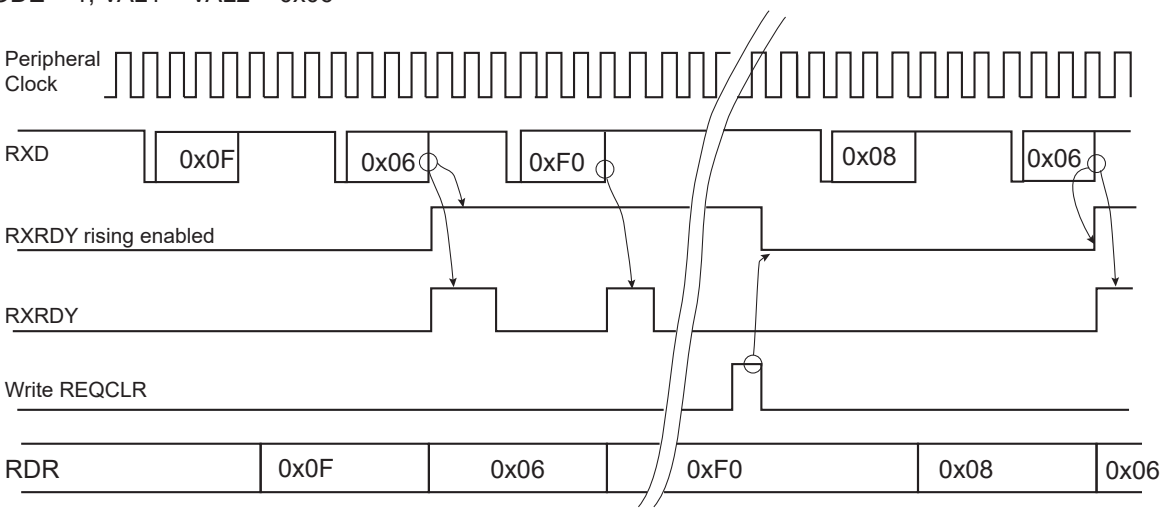
UART_CMPR (see UART Comparison Register) can be programmed to provide different comparison methods. These are listed below:

- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if either received character equals VAL1 or VAL2.

By programming the CMPMODE bit to 1, the comparison function result triggers the start of the loading of UART_RHR (see the figure below). The trigger condition occurs as soon as the received character value matches the condition defined by the programming of VAL1, VAL2 and CMPPAR in UART_CMPR. The comparison trigger event can be restarted by writing a one to the REQCLR bit in UART_CR.

**Figure 47-11. Receive Holding Register Management**

CMPMODE = 1, VAL1 = VAL2 = 0x06



### 47.5.6 Asynchronous and Partial Wake-up (SleepWalking)

Asynchronous and partial wake-up (SleepWalking) is a means of data pre-processing that qualifies an incoming event, thus allowing the UART to decide whether or not to wake up the system. SleepWalking is used primarily when the system is in Wait mode (refer to section “Power Management Controller (PMC)”) but can also be enabled when the system is fully running.

No access must be performed in the UART between the enable of asynchronous partial wake-up and the wake-up performed by the UART.

If the system is in Wait mode and asynchronous and partial wake-up is enabled, the maximum baud rate that can be achieved equals 19200.

If the system is running or in Sleep mode, the maximum baud rate that can be achieved equals 115200 or higher. This limit is bounded by the peripheral clock frequency divided by 16.

The UART_RHR must be read before enabling asynchronous and partial wake-up.

When SleepWalking is enabled for the UART (refer to section “Power Management Controller (PMC)”), the PMC decodes a clock request from the UART. The request is generated as soon as there is a falling edge on the RXD line as this may indicate the beginning of a start bit. If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the UART.

As soon as the clock is provided by the PMC, the UART processes the received frame and compares the received character with VAL1 and VAL2 in UART_CMPR (UART Comparison Register).

The UART instructs the PMC to disable the clock if the received character value does not meet the conditions defined by VAL1 and VAL2 fields in UART_CMPR (see Asynchronous Event Generating Only Partial Wake-up).

If the received character value meets the conditions, the UART instructs the PMC to exit the full system from Wait mode (see Asynchronous Wake-up Use Case Examples).

The VAL1 and VAL2 fields can be programmed to provide different comparison methods and thus matching conditions.

- If VAL1 equals VAL2, then the comparison is performed on a single value and the wake-up is triggered if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 wakes up the system.
- If VAL1 is strictly higher than VAL2, then the wake-up is triggered if the received character equals VAL1 or VAL2.
- If VAL1 = 0 and VAL2 = 255, the wake-up is triggered as soon as a character is received.

The matching condition can be configured to include the parity bit (CMPPAR in UART_CMPR). Thus, if the received data matches the comparison condition defined by VAL1 and VAL2 but a parity error is encountered, the matching condition is cancelled and the UART instructs the PMC to disable the clock (see Asynchronous Event Generating Only Partial Wake-up).

If the processor and peripherals are running, the UART can be configured in Asynchronous and partial wake-up mode by enabling the PMC_SLPWK_ER (see "Power Management Controller (PMC)"). When activity is detected on the receive line, the UART requests the clock from the PMC and the comparison is performed. If there is a comparison match, the UART continues to request the clock. If there is no match, the clock is switched off for the UART only, until a new activity is detected.

The CMPMODE configuration has no effect when Asynchronous and Partial Wake-up mode is enabled for the UART (see PMC_SLPWK_ER in "Power Management Controller (PMC)").

When the system is kept in active/running mode and the UART enters Asynchronous and Partial Wake-up mode, the flag CMP must be programmed as the unique source of the UART interrupt.

When the system exits Wait mode as the result of a matching condition, the RXRDY flag is used to determine if the UART is the source of exit.

**Note:** If the SleepWalking function is enabled on the UART, a divide by 8 of the peripheral clock versus the bus clock is not possible. Other dividers can be used with no constraints.

© 2021 Microchip Technology Inc. and its subsidiaries

**Figure 47-12. Asynchronous Wake-up Use Case Examples**

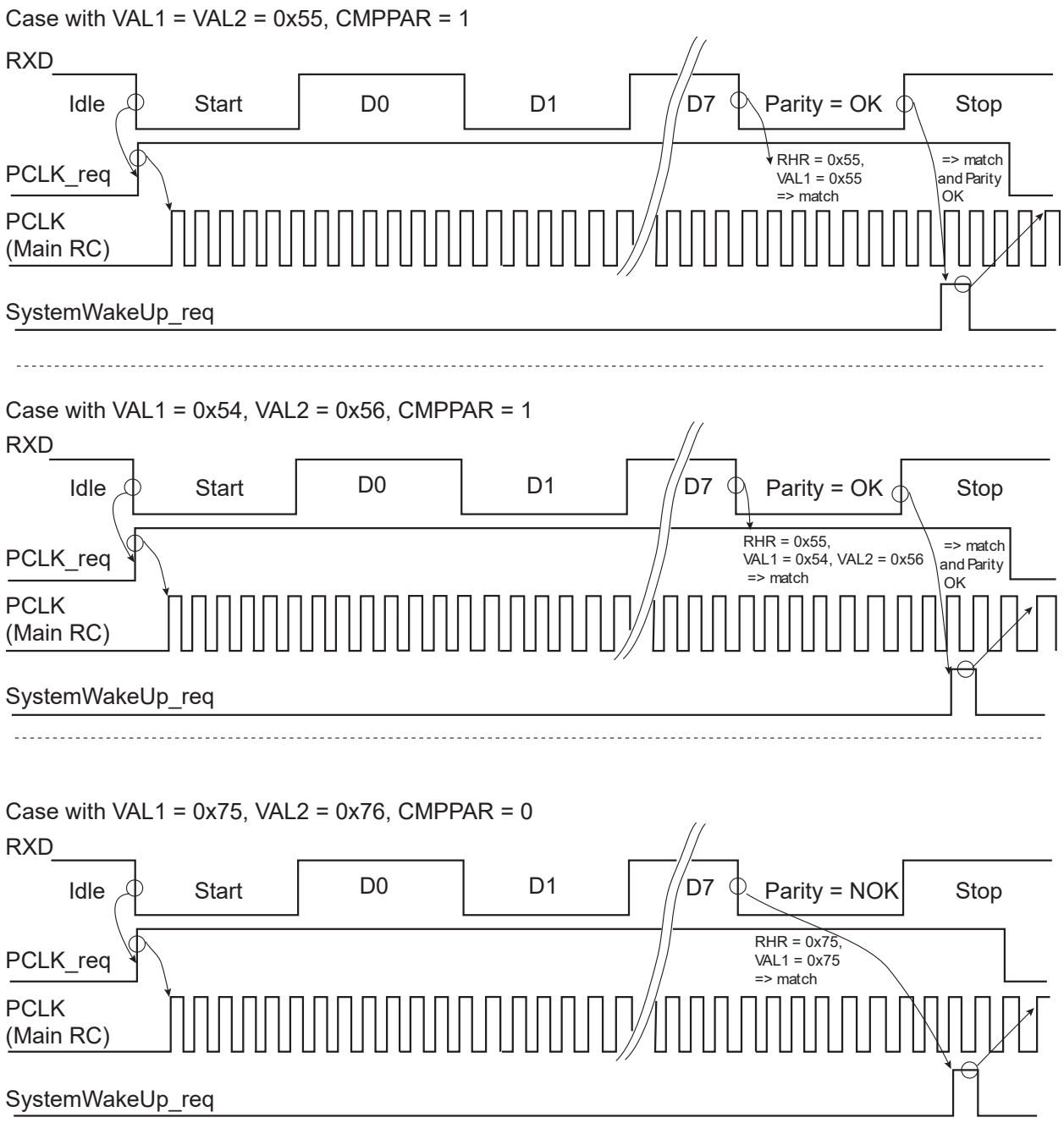
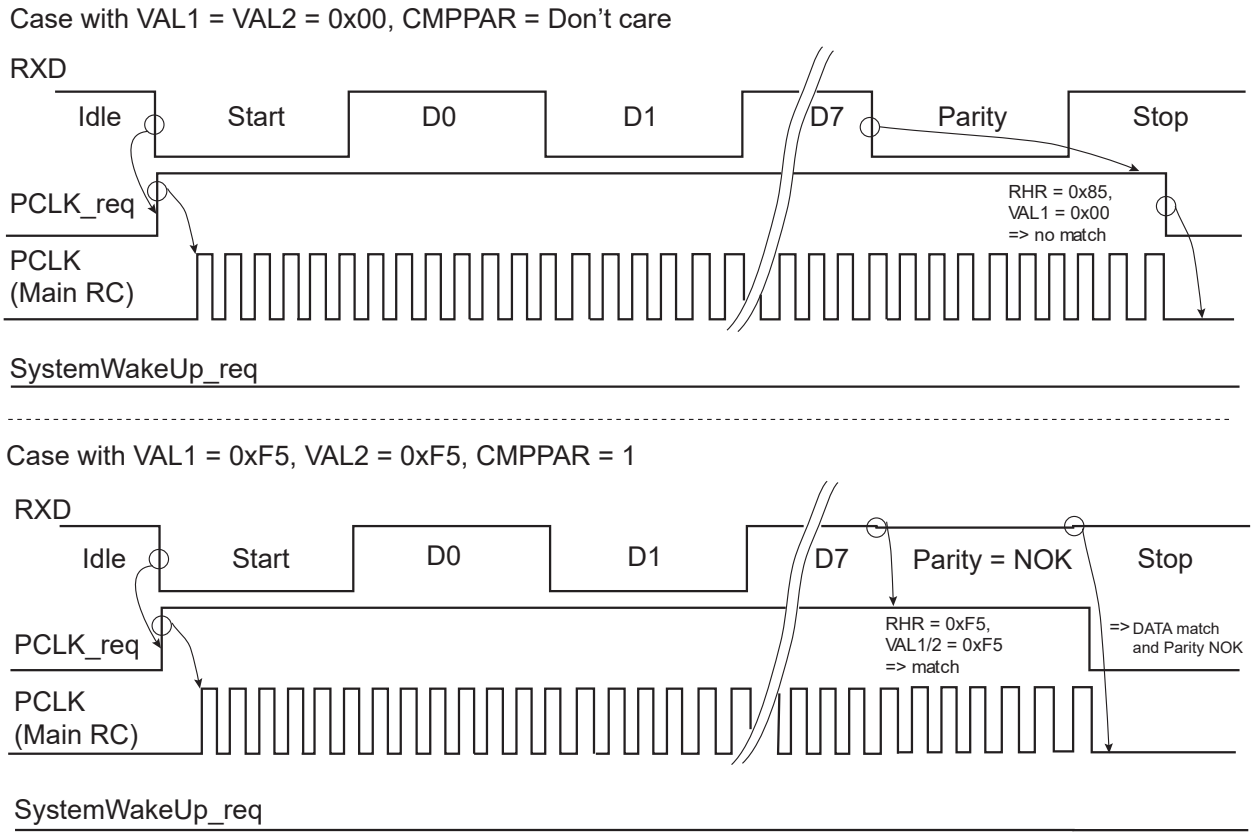


**Figure 47-13. Asynchronous Event Generating Only Partial Wake-up**




**Related Links**

31. Power Management Controller (PMC)

### 47.5.7 Register Write Protection

To prevent any single software error from corrupting UART behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the UART Write Protection Mode Register (UART_WPMR).

The following registers can be write-protected:

- UART Mode Register
- UART Baud Rate Generator Register
- UART Comparison Register

### 47.5.8 Test Modes

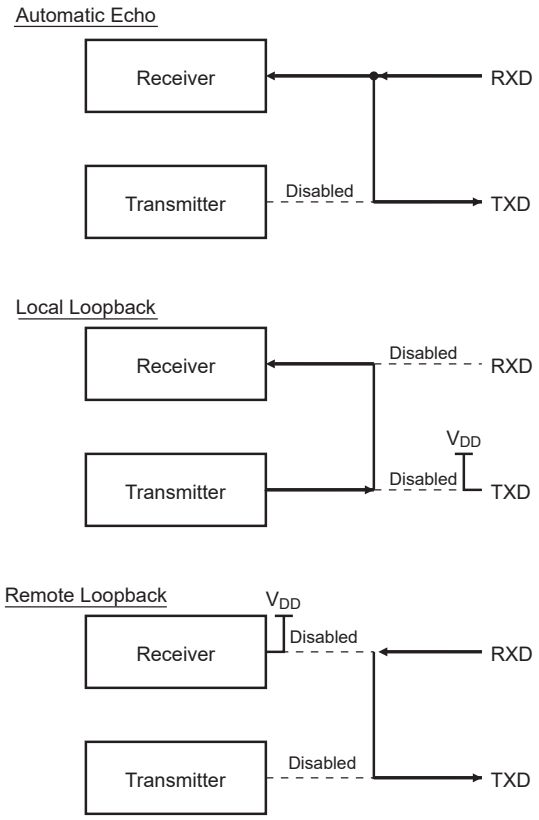
The UART supports three test modes. These modes of operation are programmed by using the CHMODE field in UART_MR.

The Automatic Echo mode allows a bit-by-bit retransmission. When a bit is received on the URXD line, it is sent to the UTXD line. The transmitter operates normally, but has no effect on the UTXD line.

The Local Loopback mode allows the transmitted characters to be received. UTXD and URXD pins are not used and the output of the transmitter is internally connected to the input of the receiver. The URXD pin level has no effect and the UTXD line is held high, as in idle state.

The Remote Loopback mode directly connects the URXD pin to the UTXD line. The transmitter and the receiver are disabled and have no effect. This mode allows a bit-by-bit retransmission.

**Figure 47-14. Test Modes**



## 47.6 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | UART_CR | 7:0 | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| | | 15:8 | | | | REQCLR | | | | RSTSTA |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | UART_MR | 7:0 | | | | FILTER | | | | |
| | | 15:8 | CHMODE[1:0] | | | BRSRCCK | PAR[2:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | UART_IER | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | CMP | | | | | | TXEMPTY | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | UART_IDR | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | CMP | | | | | | TXEMPTY | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | UART_IMR | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | CMP | | | | | | TXEMPTY | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | UART_SR | 7:0 | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| | | 15:8 | CMP | | | | | | TXEMPTY | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | UART_RHR | 7:0 | RXCHR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | UART_THR | 7:0 | TXCHR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | UART_BRGR | 7:0 | CD[7:0] | | | | | | | |
| | | 15:8 | CD[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | UART_CMPR | 7:0 | VAL1[7:0] | | | | | | | |
| | | 15:8 | | CMPPAR | | CMPMODE | | | | |
| | | 23:16 | VAL2[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | UART_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 47.6.1 UART Control Register

**Name:** UART_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | REQCLR | | | | RSTSTA |
| Access | | | | W | | | | W |
| Reset | | | | – | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXDIS | TXEN | RXDIS | RXEN | RSTTX | RSTRX | | |
| Access | W | W | W | W | W | W | | |
| Reset | – | – | – | – | – | – | | |

**Bit 12 – REQCLR** Request Clear

- SleepWalking enabled:

0: No effect.
1: Bit REQCLR clears the potential clock request currently issued by UART, thus the potential system wake-up is cancelled.

- SleepWalking disabled:

0: No effect.
1: Bit REQCLR restarts the comparison trigger to enable receive holding register loading.

**Bit 8 – RSTSTA** Reset Status

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the status bits PARE, FRAME, CMP and OVRE in the UART_SR. |

**Bit 7 – TXDIS** Transmitter Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The transmitter is disabled. If a character is being processed and a character has been written in the UART_THR and RSTTX is not set, both characters are completed before the transmitter is stopped. |

**Bit 6 – TXEN** Transmitter Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The transmitter is enabled if TXDIS is 0. |

**Bit 5 – RXDIS** Receiver Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The receiver is disabled. If a character is being processed and RSTRX is not set, the character is completed before the receiver is stopped. |

**Bit 4 – RXEN** Receiver Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The receiver is enabled if RXDIS is 0. |

**Bit 3 – RSTTX** Reset Transmitter

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The transmitter logic is reset and disabled. If a character is being transmitted, the transmission is aborted. |

**Bit 2 – RSTRX** Reset Receiver

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The receiver logic is reset and disabled. If a character is being received, the reception is aborted. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 47.6.2 UART Mode Register

**Name:** UART_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHMODE[1:0] | | | BRSRCCK | PAR[2:0] | | | |
| Access | R/W | R/W | | R/W | R/W | R/W | R/W | |
| Reset | 0 | 0 | | 0 | 0 | 0 | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | FILTER | | | | |
| Access | | | | R/W | | | | |
| Reset | | | | 0 | | | | |

**Bits 15:14 – CHMODE[1:0]** Channel Mode

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode |
| 1 | AUTOMATIC | Automatic echo |
| 2 | LOCAL_LOOPBACK | Local loopback |
| 3 | REMOTE_LOOPBACK | Remote loopback |

**Bit 12 – BRSRCCK** Baud Rate Source Clock
0 (PERIPH_CLK): The baud rate is driven by the peripheral clock
1 (PMC_PCK): The baud rate is driven by a PMC-programmable clock PCK (refer to section Power Management Controller (PMC)).

**Bits 11:9 – PAR[2:0]** Parity Type

| Value | Name | Description |
|---|---|---|
| 0 | EVEN | Even Parity |
| 1 | ODD | Odd Parity |
| 2 | SPACE | Space: parity forced to 0 |
| 3 | MARK | Mark: parity forced to 1 |
| 4 | NO | No parity |

**Bit 4 – FILTER** Receiver Digital Filter
0 (DISABLED): UART does not filter the receive line.
1 (ENABLED): UART filters the receive line using a three-sample filter (16x-bit clock) (2 over 3 majority).

### 47.6.3 UART Interrupt Enable Register

**Name:** UART_IER
**Offset:** 0x08
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMP | | | | | | TXEMPTY | |
| Access | W | | | | | | W | |
| Reset | – | | | | | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | W | W | W | | | | W | W |
| Reset | – | – | – | | | | – | – |

**Bit 15 – CMP** Enable Comparison Interrupt

**Bit 9 – TXEMPTY** Enable TXEMPTY Interrupt

**Bit 7 – PARE** Enable Parity Error Interrupt

**Bit 6 – FRAME** Enable Framing Error Interrupt

**Bit 5 – OVRE** Enable Overrun Error Interrupt

**Bit 1 – TXRDY** Enable TXRDY Interrupt

**Bit 0 – RXRDY** Enable RXRDY Interrupt

© 2021 Microchip Technology Inc.
and its subsidiaries

### 47.6.4 UART Interrupt Disable Register

**Name:** UART_IDR
**Offset:** 0x0C
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMP | | | | | | TXEMPTY | |
| Access | W | | | | | | W | |
| Reset | – | | | | | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | W | W | W | | | | W | W |
| Reset | – | – | – | | | | – | – |

**Bit 15 – CMP** Disable Comparison Interrupt

**Bit 9 – TXEMPTY** Disable TXEMPTY Interrupt

**Bit 7 – PARE** Disable Parity Error Interrupt

**Bit 6 – FRAME** Disable Framing Error Interrupt

**Bit 5 – OVRE** Disable Overrun Error Interrupt

**Bit 1 – TXRDY** Disable TXRDY Interrupt

**Bit 0 – RXRDY** Disable RXRDY Interrupt

© 2021 Microchip Technology Inc.
and its subsidiaries

### 47.6.5 UART Interrupt Mask Register

**Name:** UART_IMR
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMP | | | | | | TXEMPTY | |
| Access | R | | | | | | R | |
| Reset | 0 | | | | | | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | R | R | R | | | | R | R |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 15 – CMP** Mask Comparison Interrupt

**Bit 9 – TXEMPTY** Mask TXEMPTY Interrupt

**Bit 7 – PARE** Mask Parity Error Interrupt

**Bit 6 – FRAME** Mask Framing Error Interrupt

**Bit 5 – OVRE** Mask Overrun Error Interrupt

**Bit 1 – TXRDY** Disable TXRDY Interrupt

**Bit 0 – RXRDY** Mask RXRDY Interrupt

© 2021 Microchip Technology Inc.
and its subsidiaries

### 47.6.6 UART Status Register

**Name:** UART_SR
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMP | | | | | | TXEMPTY | |
| Access | R | | | | | | R | |
| Reset | 0 | | | | | | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PARE | FRAME | OVRE | | | | TXRDY | RXRDY |
| Access | R | R | R | | | | R | R |
| Reset | 0 | 0 | 0 | | | | 0 | 0 |

**Bit 15 – CMP** Comparison Match

| Value | Description |
|---|---|
| 0 | No received character matches the comparison criteria programmed in VAL1, VAL2 fields and in CMPPAR bit since the last RSTSTA. |
| 1 | The received character matches the comparison criteria. |

**Bit 9 – TXEMPTY** Transmitter Empty

| Value | Description |
|---|---|
| 0 | There are characters in UART_THR, or characters being processed by the transmitter, or the transmitter is disabled. |
| 1 | There are no characters in UART_THR and there are no characters being processed by the transmitter. |

**Bit 7 – PARE** Parity Error

| Value | Description |
|---|---|
| 0 | No parity error has occurred since the last RSTSTA. |
| 1 | At least one parity error has occurred since the last RSTSTA. |

**Bit 6 – FRAME** Framing Error

| Value | Description |
|---|---|
| 0 | No framing error has occurred since the last RSTSTA. |
| 1 | At least one framing error has occurred since the last RSTSTA. |

**Bit 5 – OVRE** Overrun Error

| Value | Description |
|---|---|
| 0 | No overrun error has occurred since the last RSTSTA. |
| 1 | At least one overrun error has occurred since the last RSTSTA. |

**Bit 1 – TXRDY** Transmitter Ready

| Value | Description |
|---|---|
| 0 | A character has been written to UART_THR and not yet transferred to the internal shift register, or the transmitter is disabled. |
| 1 | There is no character written to UART_THR not yet transferred to the internal shift register. |

**Bit 0 – RXRDY** Receiver Ready

| Value | Description |
|---|---|
| 0 | No character has been received since the last read of the UART_RHR, or the receiver is disabled. |
| 1 | At least one complete character has been received, transferred to UART_RHR and not yet read. |

### 47.6.7 UART Receiver Holding Register

**Name:** UART_RHR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXCHR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RXCHR[7:0]** Received Character
Last received character if RXRDY is set.

### 47.6.8 UART Transmit Holding Register

**Name:** UART_THR
**Offset:** 0x1C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TXCHR[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 7:0 – TXCHR[7:0]** Character to be Transmitted
Next character to be transmitted after the current character if TXRDY is not set.

### 47.6.9 UART Baud Rate Generator Register

**Name:** UART_BRGR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – CD[15:0]** Clock Divisor

| Value | Description |
|---|---|
| 0 | Baud rate clock is disabled |
| 1 to 65,535 | If BRSRCCK = 0:<br>$\mathrm{CD} = \frac{\mathrm{f_{peripheral\ clock}}}{16 \times \mathrm{Baud\ Rate}}$<br>If BRSRCCK = 1:<br>$\mathrm{CD} = \frac{\mathrm{f_{PCKx}}}{16 \times \mathrm{Baud\ Rate}}$ |

### 47.6.10 UART Comparison Register

**Name:** UART_CMPR
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | VAL2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | CMPPAR | | CMPMODE | | | | |
| Access | | R/W | | R/W | | | | |
| Reset | | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | VAL1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – VAL2[7:0]** Second Comparison Value for Received Character

| Value | Description |
|---|---|
| 0-255 | The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to set CMP flag in UART_SR. If asynchronous partial wake-up (SleepWalking) is enabled in PMC_SLPWK_ER, the UART requests a system wake-up if condition is met. |

**Bit 14 – CMPPAR** Compare Parity

| Value | Description |
|---|---|
| 0 | The parity is not checked and a bad parity cannot prevent from waking up the system. |
| 1 | The parity is checked and a matching condition on data can be cancelled by an error on parity bit, so no wake-up is performed. |

**Bit 12 – CMPMODE** Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | FLAG_ONLY | Any character is received and comparison function drives CMP flag. |
| 1 | START_CONDITION | Comparison condition must be met to start reception. |

**Bits 7:0 – VAL1[7:0]** First Comparison Value for Received Character

| Value | Description |
|---|---|
| 0-255 | The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to set CMP flag in UART_SR. If asynchronous partial wake-up (SleepWalking) is enabled in PMC_SLPWK_ER, the UART requests a system wake-up if the condition is met. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 47.6.11 UART Write Protection Mode Register

**Name:** UART_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x554152 | PASSWD | Writing any other value in this field aborts the write operation. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for the list of registers that can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII). |

© 2021 Microchip Technology Inc.
and its subsidiaries

# 48. Media Local Bus (MLB)

## 48.1 Description

The MediaLB (MLB) maps all the MOST Network data types (transport methods) into a single low-cost, scalable, and standardized hardware interface between a MediaLB Controller and at least one other MediaLB Device. The use of MediaLB simplifies the hardware interface, reduces the pin count, and facilitates the design of modular reusable hardware. From a software development perspective, the use of MediaLB relieves the system developer from the complexity of the MOST Network, which simplifies software development and enables the design of reusable software for different applications. This simplified, standardized interface shortens time-to-market and makes software maintenance effortless.

The link layer and three different physical layers are defined as part of this specification. The physical layer section describes pin configurations, operating speeds, and bus topology. The link layer section describes the compliance of the signaling and addressing protocol.

### 48.1.1 MediaLB Concept

The MediaLB topology supports communication among all MediaLB Devices, including the MediaLB Controller. The bus interface consists of a uni-directional line for clock (MLBC), a bi-directional line for signal information (MLBS), and a bi-directional line for data transfer (MLBD).

The MediaLB topology supports one Controller connected to one or more Devices, where the Controller is the interface between the MediaLB Devices and the MOST Network. The MediaLB Controller includes MediaLB Device functionality, and also generates the MediaLB clock (MLBC) that is synchronized to the MOST Network. This generated clock provides the timing for the entire MediaLB interface. The Controller will continue to generate MLBC even when the Controller loses lock with the MOST Network.

The MLBS line is a multiplexed signal which carries ChannelAddresses generated by the MediaLB Controller, as well as Command and RxStatus bytes from MediaLB Devices. Each ChannelAddress indicates which Device can transmit data and which Device (or Devices) can receive data on a particular logical channel.

The MLBD line is driven by the transmitting MediaLB Device and is received by all other MediaLB Devices, including the MediaLB Controller. The MLBD line carries the actual data (synchronous, asynchronous, control, or isochronous). For synchronous stream data transmission, multiple MediaLB Devices can receive the same data, in a broadcast fashion. The transmitting MediaLB Device indicates the particular type of data transmitted by sending the appropriate command on the MLBS line. The Link Layer section defines the different commands supported.

### 48.1.2 MediaLB Protocol

Once per MOST Network frame, the MediaLB Controller generates a unique FRAMESYNC pattern on the MLBS line. For all Devices on the bus, the end of the FRAMESYNC pattern defines the byte boundary and the channel boundary for the MLBS and MLBD lines.

Each four-byte wide block (quadlet) in a 3-pin MediaLB frame is defined as a physical channel. Physical channels can be grouped into multiple quadlets (which do not have to be consecutive) to form a logical channel. The MediaLB Controller handles channel arbitration, allocates channel bandwidth for MediaLB Devices, and manages the unique ChannelAddresses for referencing logical channels.

The MediaLB Controller initiates communication with MediaLB Devices by sending an assigned ChannelAddress on MLBS in each logical channel. This ChannelAddress indicates which MediaLB Device will transmit data and which MediaLB Devices will receive data in the following logical channel.

One physical channel after the ChannelAddress is sent on MLBS, the transmitting MediaLB Device associated with that ChannelAddress outputs a command byte (Command) on MLBS and respective data (Data) on MLBD, concurrently. The Command byte contains information about the data simultaneously being transmitted. The MediaLB Device receiving the data outputs a status byte (RxStatus) on MLBS after the transmitting Device sends the Command byte. This status response can indicate that the Device is ready to receive the data, or that the receiving Device is busy (e.g. cannot receive the data at present). Since synchronous stream data is sent in a broadcast fashion, Devices receiving synchronous data can never return a busy status response. In this situation, the RxStatus byte must not be actively driven onto the MLBS line by Devices receiving synchronous data.

© 2021 Microchip Technology Inc.
and its subsidiaries

The ChannelAddresses output by the Controller for each logical channel are used in normal data transport and can be statically or dynamically assigned. To support dynamic configuration of MediaLB Devices, a unique DeviceAddress must be assigned to all MediaLB Devices before startup. DeviceAddresses allow the External Host Controller (EHC) and MediaLB Controller to dynamically determine which Devices exist on the bus. At the request of a MediaLB Device (e.g. EHC), the Controller scans for DeviceAddresses in the System Channel. Once a Device is detected, a ChannelAddress for each logical channel can be assigned.

The DeviceAddress, ChannelAddress, Command, and RxStatus structures are described in the Link Layer section.

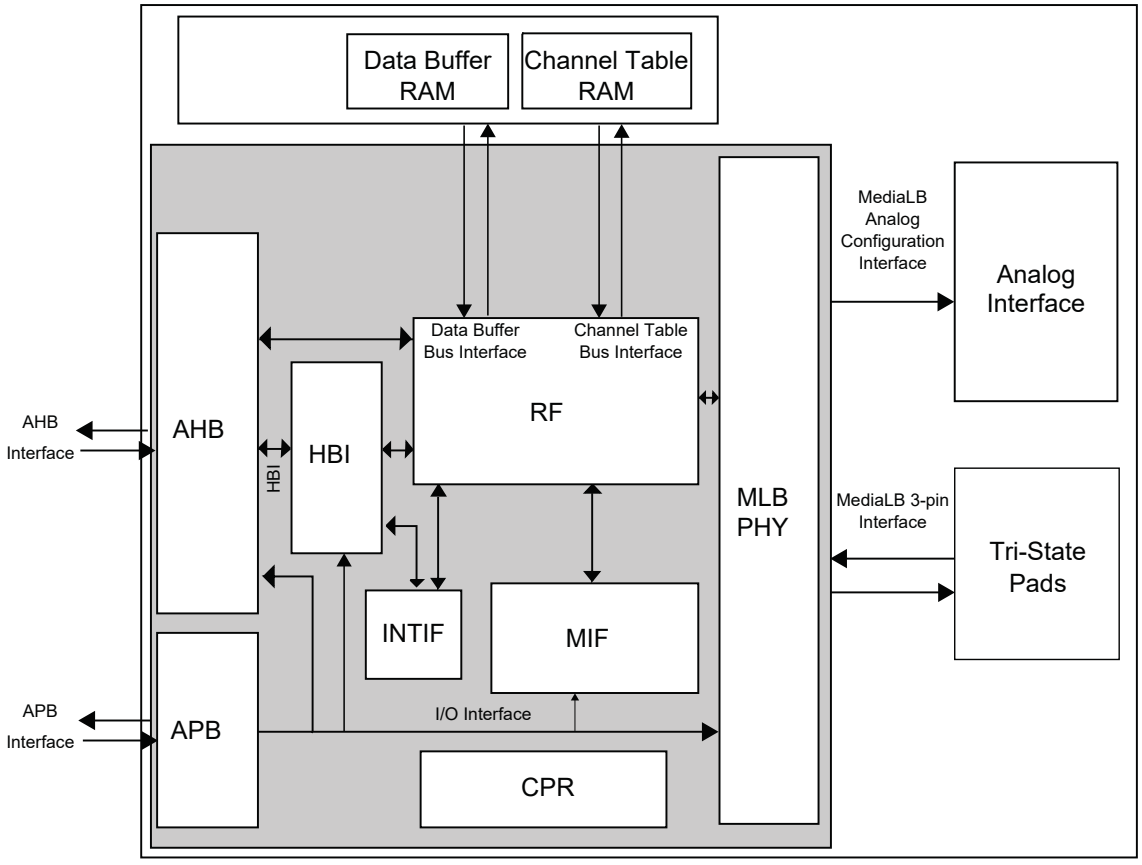
## 48.2 Embedded Characteristics

- Support of all MOST data transport methods: synchronous stream data, asynchronous packet data, control message data, and isochronous data
- Multiple clock rates supported
- Scalable data rate for all MOST Network data transport methods
- A frame synchronization pattern (FRAMESYNC) enables easy Device synchronization to MOST Networks
- Dedicated system-broadcast channel for administration
- Support of MediaLB Controller to MediaLB Device transfers and inter-MediaLB Devices transfers
- Broadcast support from one transmitter to multiple receivers for synchronous stream data

## 48.3 Block Diagram

The following figure is the top-level block diagram of the MLB behavioral models.

**Figure 48-1. 3-Pin MLB Block Diagram**



## 48.4 Signal Description

### 48.4.1 Definition of Terms

The following terms will be used when referring to specific implementations of MediaLB.

**Table 48-1. MediaLB Definition of Terms**

| Names | Description |
|---|---|
| Media Local Bus: | |
| MLBC | General reference to the Clock line of a Media Local Bus:<br>on a 3-pin MediaLB interface, connects to the MLBCLK pin |
| MLBS | General reference to the Signal line of a Media Local Bus:<br>on a 3-pin MediaLB interface, connects to the MLBSIG pin |
| MLBD | General reference to the Data line of a Media Local Bus:<br>on a 3-pin MediaLB interface, connects to the MLBDAT pin |
| 3-pin MediaLB Interface: | |
| MLBCLK | MediaLB Controller (output) pin connected to MLBC.<br>MediaLB Device (input) pin connected to MLBC. |
| MLBSIG | MediaLB Device (I/O) pin connected to MLBS. |
| MLBDAT | MediaLB Device (I/O) pin connected to MLBD. |

### 48.4.2 External Signals

The following table describes the external signals of the MLB.

**Table 48-2. MLB External Signals**

| Signal | Description | Direction |
|---|---|---|
| MLBCLK | 3-wire clock signal. | I |
| MLBDATA | 3-wire data signal. | I/O |
| MLBSIG | 3-wire signal. | I/O |

## 48.5 Product Dependencies

### 48.5.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines.

The programmer must first program the PIO controllers to assign the MLB pins to their peripheral functions.

### 48.5.2 Power Management

The MLB can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the MLB clock.

### 48.5.3 Interrupt Sources

The MLB interface has two interrupt lines connected to the interrupt controller. Handling the MLB interrupts requires programming the interrupt controller before configuring the MLB.

© 2021 Microchip Technology Inc.
and its subsidiaries

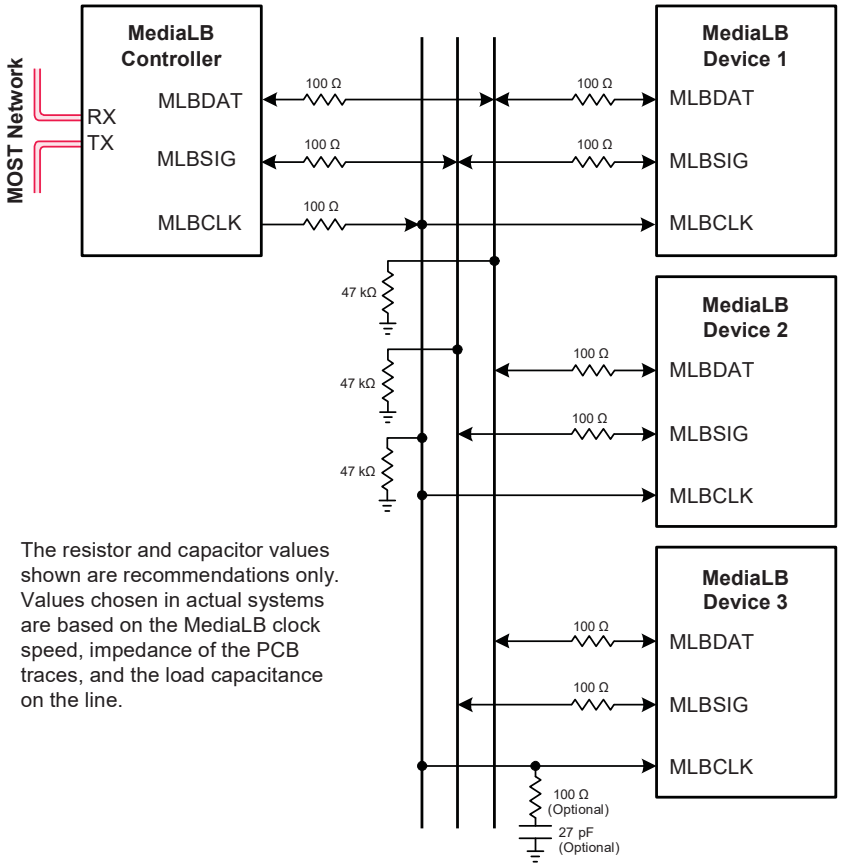
### 48.5.4 3-pin MediaLB Interface

#### 48.5.4.1 Pin Description

The MediaLB system clock is generated by a single MediaLB Controller. The MediaLB Controller outputs the clock on the MLBCLK pin, which is connected to the clock input of all other MediaLB Devices in the system. All MediaLB Devices (including the MediaLB Controller), share the signals connected to the MLBSIG and MLBDAT pins.

Once per physical channel (quadlet) on the MLBSIG line, the Controller outputs the ChannelAddress, the transmitting Device outputs Command, and the receiving Device outputs RxStatus. Therefore, each Device must set MLBSIG high impedance when not driving in order to allow the other Devices to drive it. Once per physical channel, the transmitting Device must also drive data onto the MLBDAT line, and set the line to high impedance for physical channels not allocated to that particular Device. As illustrated in the following figure, pull-down resistors are required on each signal to keep them in a known state when neither the Controller nor a Device is driving. Resistors are also recommended near the Controller and Device transmit lines for series termination and rise/fall time control. The clock line (MLBCLK) may optionally have AC-parallel termination near the farthest Device from the Controller to ensure a clean clock by minimizing reflections.

**Figure 48-2. 3-pin MediaLB Connection Diagram**




## 48.6 Functional Description

### 48.6.1 Link Layer

The MediaLB link layer uses the concept of ChannelAddress, Command, RxStatus, and Data to transport all MOST Network data types and manage MediaLB.

These terms are defined as follows:

- ChannelAddress:

A 16-bit token, which is sent on the MLBS line by the MediaLB Controller at the end of a physical channel. A unique ChannelAddress defines a logical channel and grants a particular physical channel to a transmitting (Tx) and a receiving (Rx) MediaLB Device.

- Command:

A byte-wide value sent by the transmitting (Tx) MediaLB Device on the MLBS line at the start of a physical channel. This command byte indicates the data type and additional control information to the Rx MediaLB Device. The Tx Device also outputs data on the MLBD signal during the same physical channel that Command is sent.

- RxStatus:

A byte-wide value sent by the receiving (Rx) MediaLB Device on the MLBS line, after Command is sent. This status response provides a hardware handshaking mechanism and signals other control information, such as transmission errors, back to the sender.

- Data:

The physical channel contains Data and is sent by the Tx MediaLB Device during the same physical channel in which Command is sent. This physical channel data must be transmitted left-justified, MSB first, most significant byte first. Note the Rx Device might return a status of busy, wherein the Tx Device must retransmit the same data in the next physical channel associated with the logical channel.

To dynamically configure ChannelAddresses for logical channels, a DeviceAddress can be pre-defined for MediaLB Devices. The DeviceAddress is a 16-bit address used in the System Channel with the MLBScan command to detect which MediaLB Devices exist.

#### 48.6.1.1 Channel Addresses

A MediaLB logical channel is defined as all physical channels associated with a single ChannelAddress. A logical channel on MediaLB is unidirectional; therefore, a single MediaLB Device sends data on a logical channel to one or more receiving Devices. If two Devices require bidirectional communication, then two MediaLB logical channels are required.

A ChannelAddress is 16-bits wide. Of the 16-bits, ChannelAddress (CA) bits 15 through 9 and the LSB are always zero. Only the eight bits CA[8:1] vary. A delay of one physical channel exists between the occurrence of the ChannelAddress and the actual physical channel granted. The 0x01FE ChannelAddress is defined as the FRAMESYNC pattern, where the end of the pattern determines the byte boundary, the physical channel boundary, and indicates that the MediaLB frame starts one physical channel later (PC0). The 0x0000 ChannelAddress is defined as the BusIdle state, which indicates that the corresponding physical channel is not assigned and not used by any Device. All odd ChannelAddresses are reserved; therefore, the LSB of a valid ChannelAddress is always zero. The MLBS line is in a consistent known state when not driven by any Device. For 3-pin MediaLB, this is achieved with the required weak pull-down.

**Table 48-3. MediaLB ChannelAddresses**

| ChannelAddress (1) | Description |
|---|---|
| 0x0000 | BusIdle - Indicates that the physical channel is not being used, not assigned. |
| 0x0002..0x007E | 63 ChannelAddresses - defines the logical channels used in normal operation (3-pin MediaLB) |
| 0x0080..0x01FC | Reserved |
| 0x01FE | FRAMESYNC - MediaLB frame alignment and System Channel ChannelAddress |
| 0x0200..0xFFFF | Reserved |

Note: 1. All odd ChannelAddresses are reserved (LSB must be zero for valid ChannelAddresses).

#### 48.6.1.2 Device Addresses

DeviceAddresses are 16-bits wide, must be pre-assigned, and must be unique for each MediaLB Device. Of the 16-bits, DeviceAddress (DA) bits 15 through 9 and the LSB are always zero. Only the eight bits DA[8:1] vary. At the request of the EHC, DeviceAddresses can be scanned for by the MediaLB Controller to dynamically determine which Devices exist on MediaLB. DeviceAddresses are only used with the MLBScan command in the System Channel and

are never assigned to physical channels. Once a Device is found, the ChannelAddresses used in normal operation can be assigned.

MediaLB Devices are encouraged to support dynamic configuration, where a preset DeviceAddress is used to assign the ChannelAddresses for each logical channel. Dynamic configuration avoids collisions of ChannelAddresses on different Devices.

To minimize collisions of DeviceAddresses, programmable Devices should assign the DeviceAddress via firmware. For non-programmable Devices, it is strongly recommended to have only the upper bits fixed, and have the lower bits configurable via pins on the Device. Having the lower bits configurable via pins minimizes collisions with other manufacturer's Devices, as well as allows multiple instances of the same Device to coexist on the same MediaLB bus.

**Table 48-4. DeviceAddress Grouping**

| Device Addresses | Range | Device Type |
|---|---|---|
| 0x0002..0x017E | – | Reserved |
| 0x0180..0x0186 | 4 | External Host Controller Processors |
| 0x0188..0x018E | 4 | General Processors |
| 0x0190..0x0196 | – | Reserved |
| 0x0198..0x019E | – | Reserved |
| 0x01A0..0x01A6 | 4 | Digital Signal Processors |
| 0x01A8..0x01AE | – | Reserved |
| 0x01B0..0x01B6 | 4 | Decoder Chips |
| 0x01B8..0x01BE | – | Reserved |
| 0x01C0..0x01C6 | 4 | Encoder Chips |
| 0x01C8..0x01CE | – | Reserved |
| 0x01D0..0x01DE | 8 | Digital-to-Analog Converters (DACs) |
| 0x01E0..0x01E6 | – | Reserved |
| 0x01E8..0x01EE | – | Reserved |
| 0x01F0..0x01FC | 7 | Analog-to-Digital Converters (ADCs) |

#### 48.6.1.3 Command Bytes

The MediaLB Command field is eight-bits wide and all odd values are reserved; therefore, the LSB of Command is always zero.

Transmitting MediaLB Devices (including the Controller) place Command on the MLBS line to indicate the type of data being transmitted on the MLBD line.

Two types of MediaLB commands are defined: Normal and System. Normal commands are those sent by the transmitting MediaLB Device (or Controller) in non-System Channels. System commands are those sent by the MediaLB Controller in the System Channel.

**Table 48-5. MediaLB RxStatus Responses**

| Value (see Note) | Command | Description |
|---|---|---|
| Normal Commands (TX Device sends in non-system channels): | | |
| 00h | NoData | No data to send out in this physical channel. |

.....continued

| Value (see Note) | Command | Description |
|---|---|---|
| 02h...0Eh | rsvd | Reserved |
| 10h | SyncData | Tx Device sends out SyncData command to indicate synchronous stream data. |
| 12h...1Eh | rsvd | Reserved |
| 20h | AsyncStart | Asynchronous logical channel. Start of a packet. |
| 22h | AsyncContinue | Asynchronous logical channel. Middle of a packet. |
| 24h | AsyncEnd | Asynchronous logical channel. End of a packet. |
| 26h | AsyncBreak | Asynchronous logical channel. Indicates a packet stop. No valid data present on the MLBD line. |
| 28h...2Eh | rsvd | Reserved |
| 30h | ControlStart | Control logical channel. Start of a message. |
| 32h | ControlContinue | Control logical channel. Middle of a message. |
| 34h | ControlEnd | Control logical channel. End of a message. |
| 36h | ControlBreak | Control logical channel. Indicates a message stop. No valid data present on the MLBD line. |
| 38h...3Eh | rsvd | Reserved |
| 40h | IsoNoData | Isochronous logical channel, no data valid. |
| 42h | Iso1Byte | Isochronous logical channel, one data byte valid. First byte (MSB) transmitted/received is valid. Last three bytes in physical channel are empty. |
| 44h | Iso2Bytes | Isochronous logical channel, first two data bytes valid. First byte transmitted/received is the MSB. Last two bytes in physical channel are empty. |
| 46h | Iso3Bytes | Isochronous logical channel, first three data bytes valid. First byte transmitted/received is the MSB. Last byte in physical channel is empty. |
| 48h | Iso4Bytes | Isochronous logical channel, all four data bytes valid. First byte transmitted/received is the MSB. |
| 4Ah...4Eh | rsvd | Reserved |
| 50h | IsoSync1Byte | Isochronous logical channel, one data byte valid and start of a block. First byte transmitted/received is valid. Last three bytes in physical channel are empty. |
| 52h | IsoSync2Bytes | Isochronous logical channel, two data bytes valid and start of a block. First byte transmitted/received is the MSB. Last two bytes in the physical channel are empty. |
| 54h | IsoSync3Bytes | Isochronous logical channel, three data bytes valid and start of a block. First byte transmitted/received is the MSB. Last byte in physical channel is empty. |
| 56h | IsoSync4Bytes | Isochronous logical channel, all four data bytes valid and start of a block. First byte transmitted/received is the MSB. |
| 58h...DEh | rsvd | Reserved |

© 2021 Microchip Technology Inc. and its subsidiaries

| .....continued | | |
|---|---|---|
| **Value (see Note)** | **Command** | **Description** |
| System Commands (Controller sends in System Channel): | | |
| 00h | NoData | The Controller has no System command to send out. |
| E0h | MOSTLock | The Controller issues a MOST Network lock command in the System Channel to notify Devices that the MOST Network is in lock. |
| E2h | MOSTUnlock | The Controller issues a MOST Network unlock command in the System Channel to notify Devices that the MOST Network is unlocked. |
| E4h | MLBScan | The Controller issues an MediaLB scan command in the System Channel and uses the MLBD line to indicate the DeviceAddress which is currently being scanned. All Devices supporting MLBScan must compare the received DeviceAddress against their internal DeviceAddress, and if a match occurs, a Device responds in the following System Channel with one of the System responses as specified in Table 48-6. |
| E6h | MLBSubCmd | The Controller outputs a sub-command in the System Channel. The sub-command is part of the data on the MLBD line. |
| E8h...FCh | rsvd | Reserved |
| FEh | MLBReset | The Controller outputs a MediaLB reset on the System Channel MLBS line. If the first two-bytes are zero on the MLBD line, then the system reset is a broadcast system reset and every Device should reset its MediaLB interface. Otherwise, the MLBD line contains the DeviceAddress of the Device being asked to reset its own MediaLB interface. |

**Note:** All odd values (LSB set) are reserved.

For synchronous logical channels, the NoData command indicates that the Tx Device assigned to that ChannelAddress has not setup the channel yet. For asynchronous and control logical channels, NoData is used during packet data transfer when there is no data available to transmit.

#### 48.6.1.4 RxStatus Bytes

The MediaLB RxStatus field is eight-bits wide and all odd values are reserved; therefore, the LSB of RxStatus is always zero. Receiving Devices must place RxStatus on the MLBS line after the Tx Device command byte (Command). The RxStatus status responses are divided into two categories: current state and feedback. The current state RxStatus indicates the status of the Rx Device in the current physical channel, whereas the feedback RxStatus is a response to a Command in the previous logical channel. For Normal responses, only the ReceiverProtocolError is a feedback RxStatus byte. All System responses are also feedback RxStatus bytes.

Two types of MediaLB status responses are defined: Normal and System. Normal status responses are sent by the receiving MediaLB Device (or Controller) in the non-System Channels. System status responses are sent by the receiving MediaLB Device in the System Channel.

For synchronous data reception, the Rx Device does not drive a response. For 3-pin MediaLB, the pull-down resistor on the MLBS line implements the ReceiverReady response automatically (cannot be delayed or stopped).

For control or asynchronous packet reception, the Rx Device responds to a control or asynchronous command with ReceiverReady if it can accept the quadlet on the MLBD line. If the Rx Device cannot accept the quadlet, then it will respond with a status of ReceiverBusy. If the Rx Device needs to stop or cancel the packet transmission, it can respond with a status of ReceiverBreak, in which case the Tx Device must stop transmitting the current packet.

When the Rx Device recognizes an error, the ReceiverProtocolError status response is sent in the next physical channel that is part of the logical channel. The status response of ReceiverProtocolError is issued by the Rx Device under certain conditions, see Data Transport Methods for details.

**Table 48-6. MediaLB RxStatus Responses**

| Value (see Note) | RxStatus | Description |
|---|---|---|
| Normal Responses (Rx Device response in non-System Channels): | | |
| 00h | ReceiverReady | Current state indicating the receiving Device is ready to receive the data. This is the default for the bus. The Rx Devices should not drive this response for broadcast channels. |
| 02h...0Eh | rsvd | Reserved |
| 10h | ReceiverBusy | Current state indicating the Rx Device is not ready to receive the data. The data must be retransmitted in the next physical channel associated with this logical channel. This response is not allowed on synchronous channels. |
| 12h...6Eh | rsvd | Reserved |
| 70h | ReceiverBreak | Current state indicating the Rx Device will not receive additional data quadlets and requests termination of the data transmission. Only allowed on control and asynchronous channels. |
| 72h | ReceiverProtocolError | Feedback indicating the command received in the prior physical channel (of this logical channel) did not match the pre-defined channel format or was out of sequence. Only allowed on control and asynchronous channels. |
| 74h...7Eh | rsvd | Reserved |
| System Responses (Rx Device response in System Channel): | | |
| 00h | DeviceNotPresent | |
| 80h | DevicePresent | |
| 82h | DeviceServiceRequest | Device response to DeviceAddress scan (MLBScan), where the scanned Device needs some or all its ChannelAddresses configured. |
| 84h...FEh | rsvd | Reserved |

**Note:** All odd values (LSB set) are reserved.

### 48.6.1.5 System Commands

The Controller sends out System commands in the physical channel associated with the FRAMESYNC MediaLB frame alignment ChannelAddress (PC0). The NoData command indicates no command exists on the System Channel for this frame. All System commands are optional and may or may not be implemented on the MediaLB Controller. Additionally, System responses (including dynamic configuration) are optional and may or may not be implemented on a specific MediaLB Device.

The MOSTLock and MOSTUnlock commands indicate the status of the Controller relative to the MOST Network. When the Controller is not locked to the MOST Network (MOSTUnlock), all MediaLB data being transferred to or from the MOST Network must also stop. Buffers in the Controller could delay the stopping point to beyond when MOSTUnlock shows up on MediaLB.

The MLBReset command is designed to place the MediaLB interface in one or all Devices in a known state. When a MediaLB Device receives the MLBReset command, it will look at the corresponding first two received (most significant) data bytes on the MLBD line:

- If the first two bytes are zero, then all MediaLB Devices must reset their MediaLB interface to an initialized known state (broadcast reset to all Devices).
- If the first two bytes match the local DeviceAddress, then only the Device with the matching DeviceAddress will reset its MediaLB interface to an initialized known state (reset targeted to only one Device).

© 2021 Microchip Technology Inc. and its subsidiaries

The MLBSubCmd command is used for configuration and status information from the Controller to Devices. A sub-command is contained in the first byte of the MLBD quadlet. When MediaLB Device interfaces receive the MLBSubCmd command, they will store the command and corresponding data quadlet (sub-command). Currently, only one sub-command is defined (scSetCA) and is used in dynamic configuration.

MediaLB Devices and ChannelAddresses can be configured using two methods: static or dynamic. When the EHC MediaLB Device uses the dynamic method, it instructs the Controller to scan for other MediaLB Devices. As Devices are found, the EHC then instructs the Controller to configure the found Device via the MLBSubCmd command.

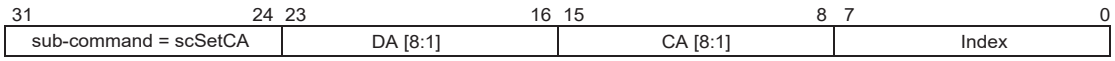
The EHC determines which DeviceAddresses to scan for and, once a Device is found, which ChannelAddresses to assign. The EHC uses the pre-defined logical channels opened when MediaLB was started to transfer messages to the Controller. The EHC sends a message to the Controller to start scanning for a particular DeviceAddress. The Controller then sends the MLBScan command into the System Channel, and places the DeviceAddress into the first two bytes (most significant or first two transmitted) of the System Channel on MLBD.

An Rx Device with a matching DeviceAddress must send a status response of DevicePresent in the next System Channel if the ChannelAddresses are already assigned or fixed. If the ChannelAddresses have not been assigned, then the Rx Device must respond with DeviceServiceRequest.

If a Device is found, the Controller sends a message to the EHC indicating the Device's presence and whether the Device needs to be configured or not. For Devices that need to be configured (requesting service), the EHC must then send a message to the Controller defining which ChannelAddresses to send to the Device. The Controller then sends this information to all Devices using the MLBSubCmd command in the System Channel.

The MLBSubCmd command data field contains four bytes that are defined as follows:

**Figure 48-3. Sub-Command scSetCA Quadlet**




The scSetCA (01h) sub-command (under the MediaLB MLBSubCmd command) supports dynamic configuration of MediaLB ChannelAddresses. The bytes are defined as follows:

- scSetCA (01h) - Sub-command to Set ChannelAddress. Indicates that the rest of the bytes are logical channel configuration information.
- DA[8:1] - DeviceAddress bits 8 through 1, where all other bits are zero. Matches the DeviceAddress found during the MLBScan command.
- CA[8:1] - ChannelAddress bits 8 through 1, where all other bits are zero. Assigned ChannelAddress associated with a specific Index (Device's logical channel) below.
- Index - Indicates which logical channel within a Device to associate the ChannelAddress with. This index enables a Device to support multiple logical channels. Index 0 and 1 are reserved for control channels. Devices that do not support control channels will start at Index 2 (with Indices 0 and 1 unused).

MediaLB Devices receiving this sub-command should check the DA[8:1] byte to determine whether this DeviceAddress matches its own. If the DeviceAddress matches, then the Device uses the ChannelAddress (CA[8:1] bits) for the logical channel associated with that Index. If a Device is reset or drops off MediaLB, it must reinitialize to its power-up state and discard any previously assigned ChannelAddresses.

MediaLB Device documentation must contain a table defining the relationship between the Index value, the particular logical channel associated with it, and the type and maximum bandwidth supported. In addition, the Device must indicate how many frames are needed to set the ChannelAddress once the scSetCA sub-command has been received. The EHC must use this data to determine the wait between setting Indices/Logical channels.

#### 48.6.1.6 Data Structure for 3-pin MediaLB

The 3-pin MediaLB data structure consists of a ChannelAddress, a Tx command (Command), an Rx response (RxStatus), and four data bytes (Data).

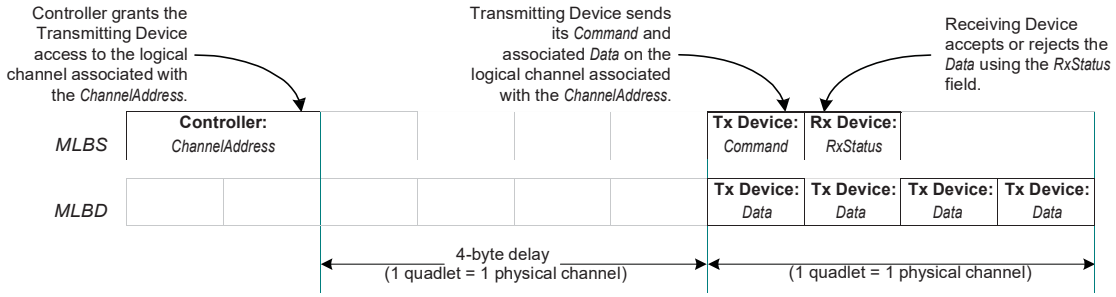
A MediaLB data structure flow is:

- The MediaLB Controller places a ChannelAddress on the MLBS line. This addresses two or more MediaLB Devices. One acts as a Tx MediaLB Device and the other or others act as Rx MediaLB Devices.
- After a fixed delay of 4 bytes (one quadlet or physical channel), the addressed Tx MediaLB Device responds by shifting out a command byte (Command) onto the MLBS line, coincident with the start of 4 bytes of data onto the MLBD line.

© 2021 Microchip Technology Inc.
and its subsidiaries

- The Rx MediaLB Device responds in the same physical channel by shifting out its status response (RxStatus) onto the MLBS line after the Tx Device's Command. The RxStatus reports the status of the receiving Device to the sender. For asynchronous, control and isochronous (non-broadcast) transmissions, the data sent is accepted if the receiver presents a status response of ReceiverReady or rejected if the receiver presents a status response of ReceiverBusy. For synchronous and isochronous (broadcast) transmissions, the receiving Device must not drive any RxStatus, thereby defaulting to ReceiverReady. Synchronous (and some isochronous) data is sent in a broadcast fashion and supports multiple receiving Devices.

**Figure 48-4. 3-pin MediaLB Data Structure**




During normal operation, the MediaLB Controller initiates a transfer by sending out the ChannelAddress on the MLBS line, and then stops driving (high-impedance) the MLBS line. When a MediaLB Device recognizes the ChannelAddress as related to one of its channels, the Tx Device will generate the Command on the MLBS line and place the data on the MLBD line. The Rx Device will generate the RxStatus on the MLBS line, after the Command. Both Command and RxStatus are output in the second quadlet after the matching ChannelAddress occurred. If the Rx Device reports a status response of ReceiverBusy, then the Tx Device must retransmit the Command and Data in the next physical channel assigned to that same ChannelAddress (next quadlet in the logical channel). If the Tx Device transmits the NoData command, the Rx Device ignores the data on the MLBD line.

This results in the following scheme:

Controller: ChannelAddress → Tx Device: Command → Rx Device: RxStatus

Since for synchronous data transmission (SyncData) the status response must always be ReceiverReady (bus default when signal not driven), synchronous data supports broadcast transmission to multiple Rx Devices.

After the Tx Device outputs Command, it must stop driving the MLBS line to allow the Rx Device to output RxStatus. At the end of the physical channel, the Tx Device must also stop driving the MLBD line unless the ChannelAddress for the next physical channel is also assigned to it. Likewise, after the Rx Device outputs RxStatus, it must stop driving the MLBS line to allow the Controller to output another ChannelAddress.

Figure 48-5 illustrates which Device is driving the MediaLB signal and data lines, using the 256Fs speed as an example. Depending on the number of physical channels that are grouped into logical channels, fewer unique ChannelAddresses may be seen in the frame. In Figure 48-5, each logical channel is one quadlet (one physical channel), mapping to seven ChannelAddresses (B through H). If one logical channel consisted of two quadlets and another consisted of three quadlets, then only four unique ChannelAddresses would be seen on MediaLB (B through E).

For MediaLB synchronization purposes, ChannelAddress 0x01FE is defined as the FRAMESYNC pattern. The MediaLB Controller generates this pattern once per MOST Network frame on the MLBS line. The MediaLB link layer is designed to ensure that this bit pattern is unique on the bus.

All MediaLB Devices must synchronize their byte boundary and their physical channel boundary upon receiving the FRAMESYNC pattern. The end of the FRAMESYNC pattern also indicates that four bytes later is the start of the MediaLB frame (PC0), and the System Channel. The actual number of physical channels supported is determined by the MediaLB clock speed. the following table illustrates the number of available quadlets and physical channels per frame for 3-pin MediaLB speed modes.

**Table 48-7. 3-pin MediaLB Valid Physical Channels**

| MediaLB Speed | Physical Channels per Frame | Available Physical Channels per Frame (see Note) |
|---|---|---|
| 256×Fs | 8 | 7 (PC1–PC7) |

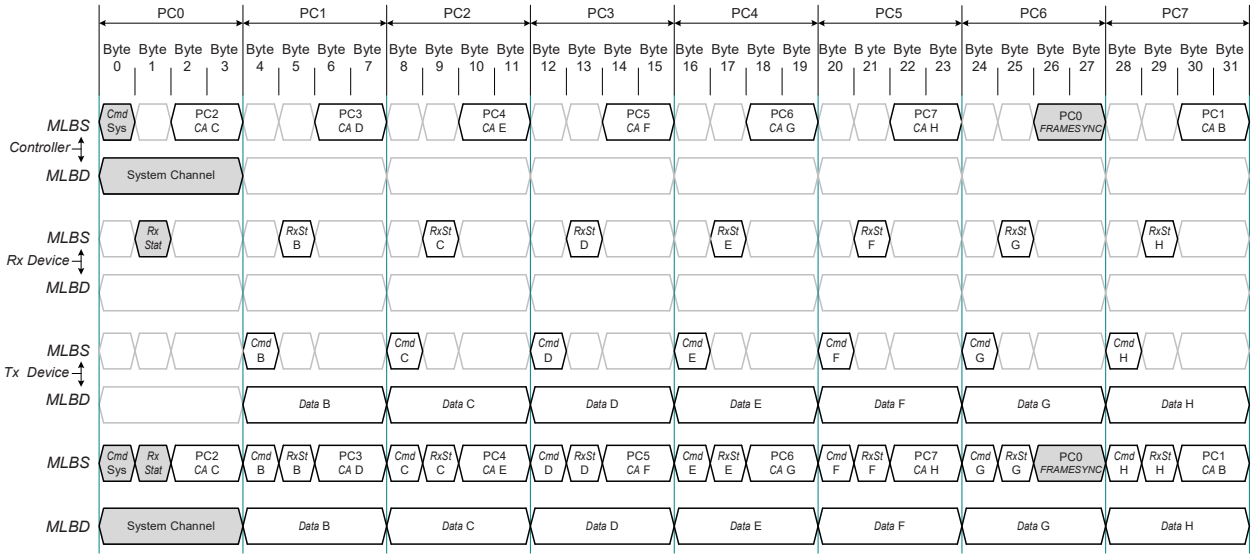
| .....continued | | |
|---|---|---|
| **MediaLB Speed** | **Physical Channels per Frame** | **Available Physical Channels per Frame (see Note)** |
| 512×Fs | 16 | 15 (PC1–PC15) |

**Note:** PC0 (first physical channel of the MediaLB frame) is always used as the System Channel.

The MLBS and MLBD physical channel associated with the FRAMESYNC ChannelAddress (PC0), is defined as the System Channel and can be used by the Controller to broadcast system control and status information to all Devices. Examples of System commands are MLBReset and MLBScan. Status examples include MOSTLock and MOSTUnlock which indicate the status of the MOST Network to MediaLB Devices.

MediaLB supports both static physical channel assignments or dynamic implementations. As an example of a static implementation, the Controller can automatically open a pair of logical channels at power-up. Through these channels, the rest of the MediaLB bandwidth can be configured by a MediaLB Device (generally the EHC). For a dynamic implementation, the EHC can request the Controller to scan for specific DeviceAddresses and then configure the Devices found (see the MLBScan System command).

**Figure 48-5. 3-pin MediaLB 256Fs Interface Example**




### 48.6.1.7 Initialization

At power up, the MediaLB Controller might output a MLBReset command in the System Channel (all System commands are optional). Upon reception of the MLBReset command, all MediaLB Devices will cancel any current transmissions or receptions and clear their buffers.

Two scenarios are supported to configure MediaLB Devices and ChannelAddresses:

- Static pre-configured before startup. The system implementor decides which ChannelAddresses are to be used for every communication path on MediaLB. This static MediaLB configuration can be communicated by the EHC to the Controller through pre-defined power-up logical channels or through a secondary port.
- Dynamically at run-time. Dynamic configuration allows the board designer to support multiple build options where the EHC can query to find out if a particular Device is present or not on a particular board. The EHC instructs the Controller to scan for a particular DeviceAddress in the System Channel. The Controller uses the MLBScan command to look for a Device. The Controller then notifies the EHC whether the Device is present or not. If the Device is present, then the EHC can instruct the Controller to set the ChannelAddresses for the Device found. The EHC sends messages to the Controller to set each Indices/Logical channel, and waits the appropriate amount of time between each message as specified in the Devices documentation. When that particular Device is configured, the EHC can instruct the Controller to scan for the next Device.

Since the MediaLB Controller is the interface between the MediaLB Devices and the MOST Network, the Controller provides the MLBC signal and will also continue to operate even when the MOST Network is unlocked. When

no activity exists on MediaLB, the Controller can shut off the MLBC placing MediaLB in a low-power state. The ChannelAddress assignments are not affected in low-power state; therefore, the same communication paths exists once MLBC is restarted.

MediaLB Devices are synchronously slaved to the MediaLB Controller through the MLBC signal. Since the Controller is synchronized to the MOST Network, the MLBC signal provides Network synchronization to all MediaLB Devices. Once the Controller starts up MLBC, all MediaLB Devices must synchronize to the MediaLB frame before communication can commence. When not frame-locked, Devices must search for the FRAMESYNC pattern, which defines a byte and physical channel boundary. Additionally, the start of the MediaLB frame (PC0) occurs one quadlet after FRAMESYNC is present on the bus. Even when a Device is frame-locked, it should check every frame continuing to validate that it remains frame-locked. While frame-locked, the Device can access MediaLB according the rules of the MediaLB protocol.

A MediaLB Device must perform the following operations:

- Rules for synchronization to MediaLB:
  - When locked, as long as FRAMESYNC is detected at the expected time, the Device must not synchronize to unexpected FRAMESYNC patterns.
  - When locked and FRAMESYNC is not detected at the expected time for two consecutive frames, declare unlock, and the Device stops driving MLBS and MLBD.
  - When unlocked and FRAMESYNC is detected at the same time for three consecutive frames, declare lock, and the Device can resume driving MLBS and MLBD when appropriate.
- When the Tx Device for a physical channel, it drives Command onto MLBS at the beginning of the physical channel and then sets MLBS to a high impedance state. In addition, the Tx Device drives the data quadlet onto MLBD line for the duration of the physical channel, and then sets the MLBD line to a high impedance state. The NoData command is the default for the MLBS line and does not need to be driven by the Tx Device.
- When the Rx Device for a physical channel, it drives RxStatus onto MLBS in the second byte of the physical channel and then sets MLBS to a high impedance state for asynchronous, control and isochronous (non-broadcast) transmissions. When no RxStatus is driven, the MLBS line defaults to ReceiverReady; however, it is recommended that the Rx Device drive the ReceiverReady response for non-broadcast transmissions.
- When the Rx Device for a physical channel, it must not drive any RxStatus (defaulting to ReceiverReady) for synchronous and isochronous (broadcast) transmissions.

#### 48.6.1.8 Data Transport Methods

MediaLB supports four data transport methods: synchronous stream data, asynchronous packet data, control message data and isochronous data. Synchronous stream data is transmitted in a broadcast fashion, where the only response allowed by an Rx Device is ReceiverReady (MLBS default). Control and asynchronous transport methods are packet based and support only one Rx Device at a time. Control and asynchronous transmissions require start and end commands to delineate the packets. Isochronous data can be broadcast if all Rx Devices do not use the status response of ReceiverBusy. Otherwise, isochronous transmissions must be to a single Rx Device.

Control and Asynchronous

Both the control and asynchronous commands define the boundaries of a packet message and work similarly. The following discussion on control packets also applies to asynchronous packets with the commands changed to the asynchronous versions.

For control packets, the ControlStart command is sent by the Tx Device at the start of a message. After the first quadlet of the message, middle quadlets will use the ControlContinue command. For the last quadlet of the packet, the Tx Device uses the ControlEnd command. If the command sequence is received out of order, the Rx Device sends the status response of ReceiverProtocolError in the next quadlet of the logical channel.

If the Tx Device must abort the packet while it's being transmitted, the ControlBreak command is sent. Assuming a message is to be retransmitted after the ControlBreak command is sent, the message must be restarted from the beginning (cannot resume with the ControlContinue command).

The protocol flow for a Tx Device is illustrated in Figure 48-6 through Figure 48-8. Although these diagrams illustrate control packet transmission, they also apply to asynchronous packets where the commands that start with Control are replaced by Async. The data transfer blocks (slanted rectangle shapes) occur only during a physical channel (PCn) associated with the logical channel defined by a single ChannelAddress.

© 2021 Microchip Technology Inc. and its subsidiaries

The flow diagram contains four states: Idle, Start, Continue, and End. Each state uses a different command when sending the data. The Idle state is the starting point, waiting for the application to initiate a packet transfer. When a quadlet is ready to be transferred, the flow diagram moves to the Start state.

Note: If a ControlEnd command is sent in the physical channel preceding a ReceiverProtocolError RxStatus (in either the Idle or Start state), the ReceiverProtocolError status response must be assigned to the previous packet transmitted. Alternatively, a status response of ReceiverProtocolError (in either the Idle or Start state) must not be assigned to the previous packet transmitted unless ControlEnd was sent in the preceding physical channel.

Once a quadlet has been sent successfully, the flow diagram moves to the Continue state, depicted in Figure 48-7, and stays there until all but the last quadlet has been transmitted. The last quadlet is transmitted in the End state, which is depicted in Figure 48-8.

The protocol flow for an Rx Device is illustrated in Figure 48-9. This flow diagram consists of only two states: Idle and Continue. The Idle state is the starting point where the Rx Device is waiting for a packet start command. Once a start command has been received (ControlStart or AsyncStart), the flow diagram moves to the Continue state. The reception of a ControlEnd command completes the transfer and moves the flow diagram back to the Idle state, where it waits for the next packet.

The protocol flow for an Rx Device, as described in Figure 48-9, should be used as a reference for standard MediaLB Devices. According to this flow, a ReceiverProtocolError status response may be sent by an Rx Device only in the Continue state; however, more enhanced MediaLB Devices can also conduct protocol checks in the Idle state. In this case, a ReceiverProtocolError status response could be sent for example, if a logical channel is setup for control data and an isochronous or synchronous command is received. Protocol checks in the Continue state may be expanded beyond the flow shown in Figure 48-9 when required by specific implementations.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 48-6. Control Packet Tx Device Protocol: Start**

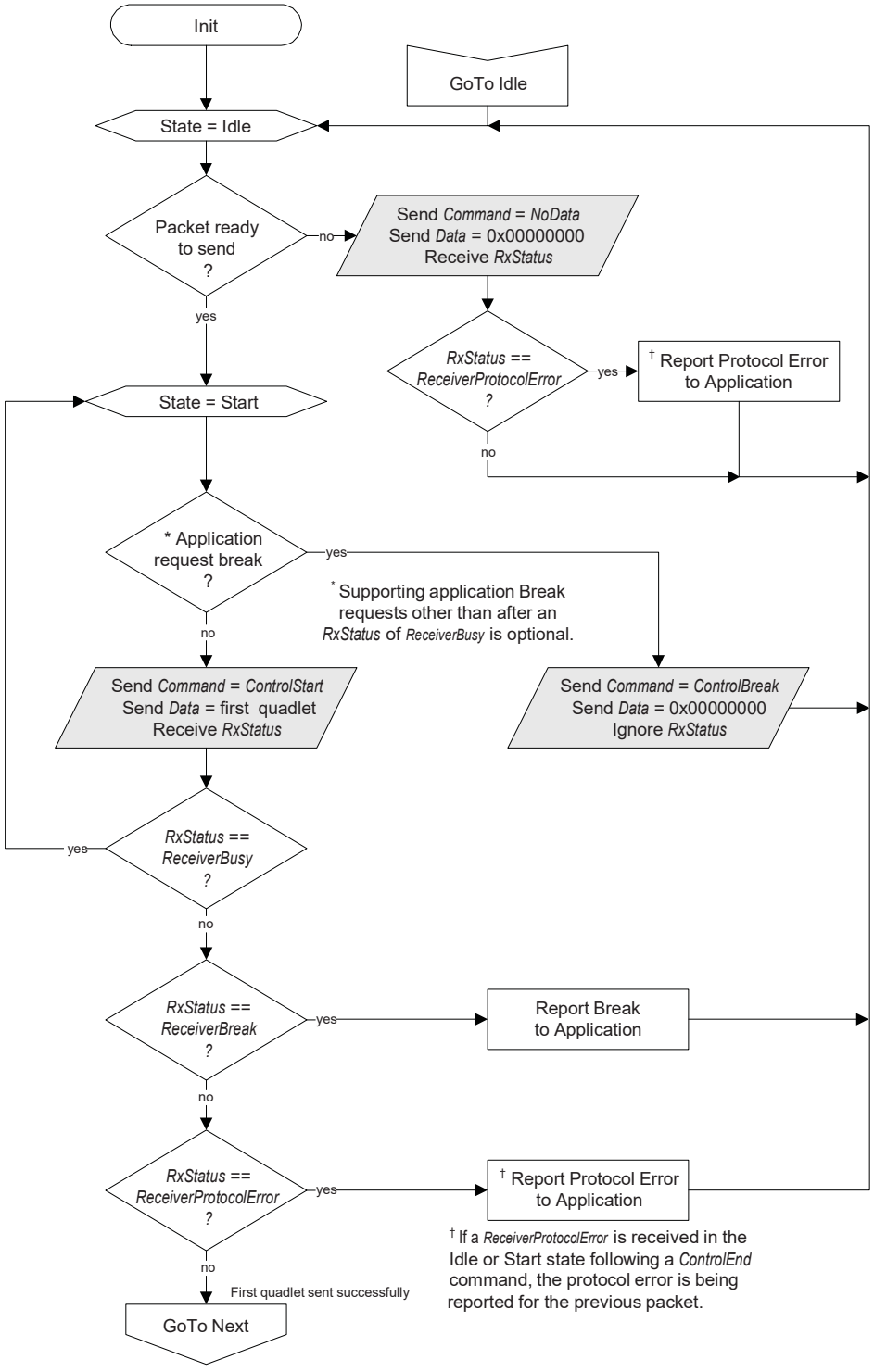


**Figure 48-7. Control Packet Tx Device Protocol: Middle**

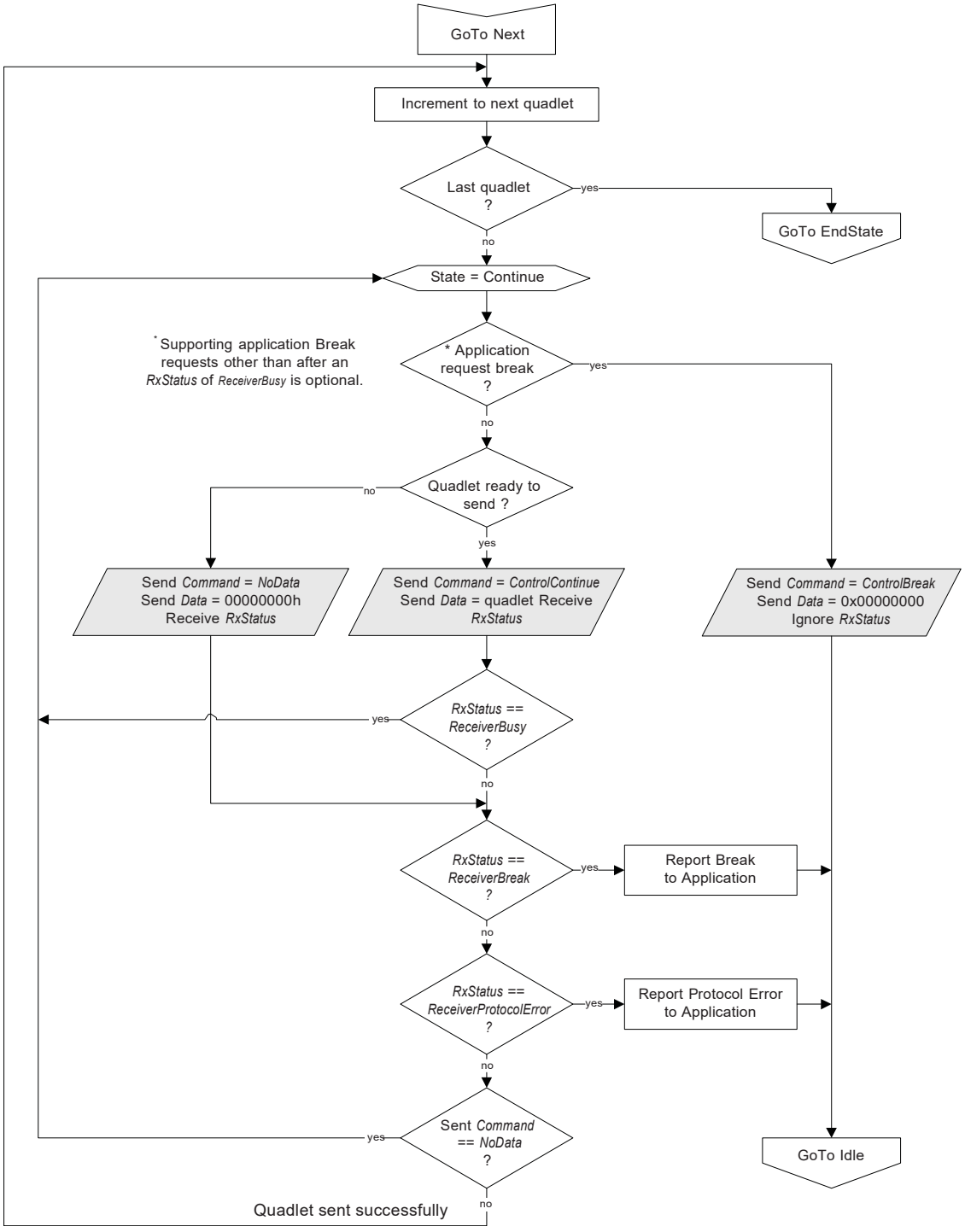


**Figure 48-8. Control Packet Tx Device Protocol: End**

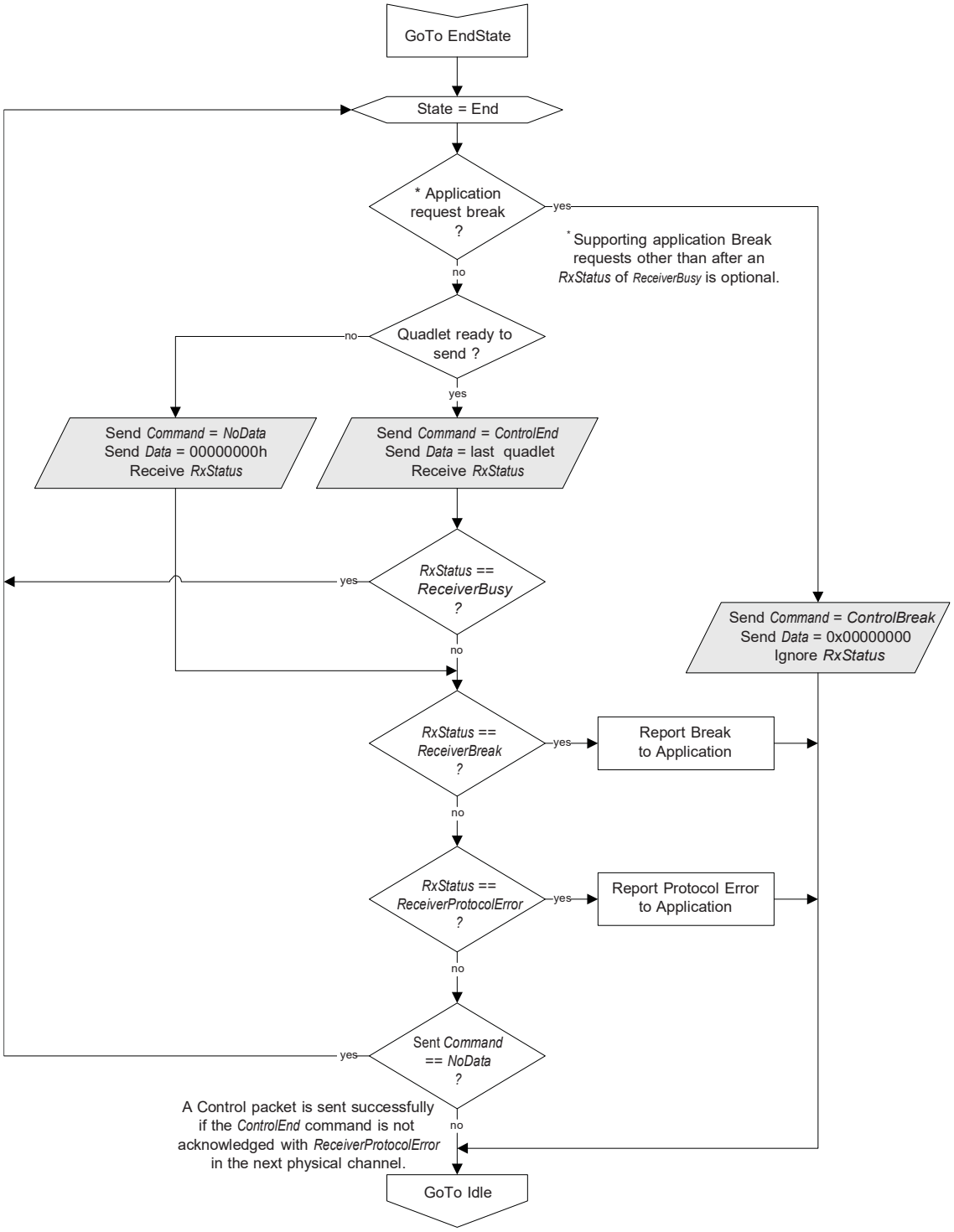
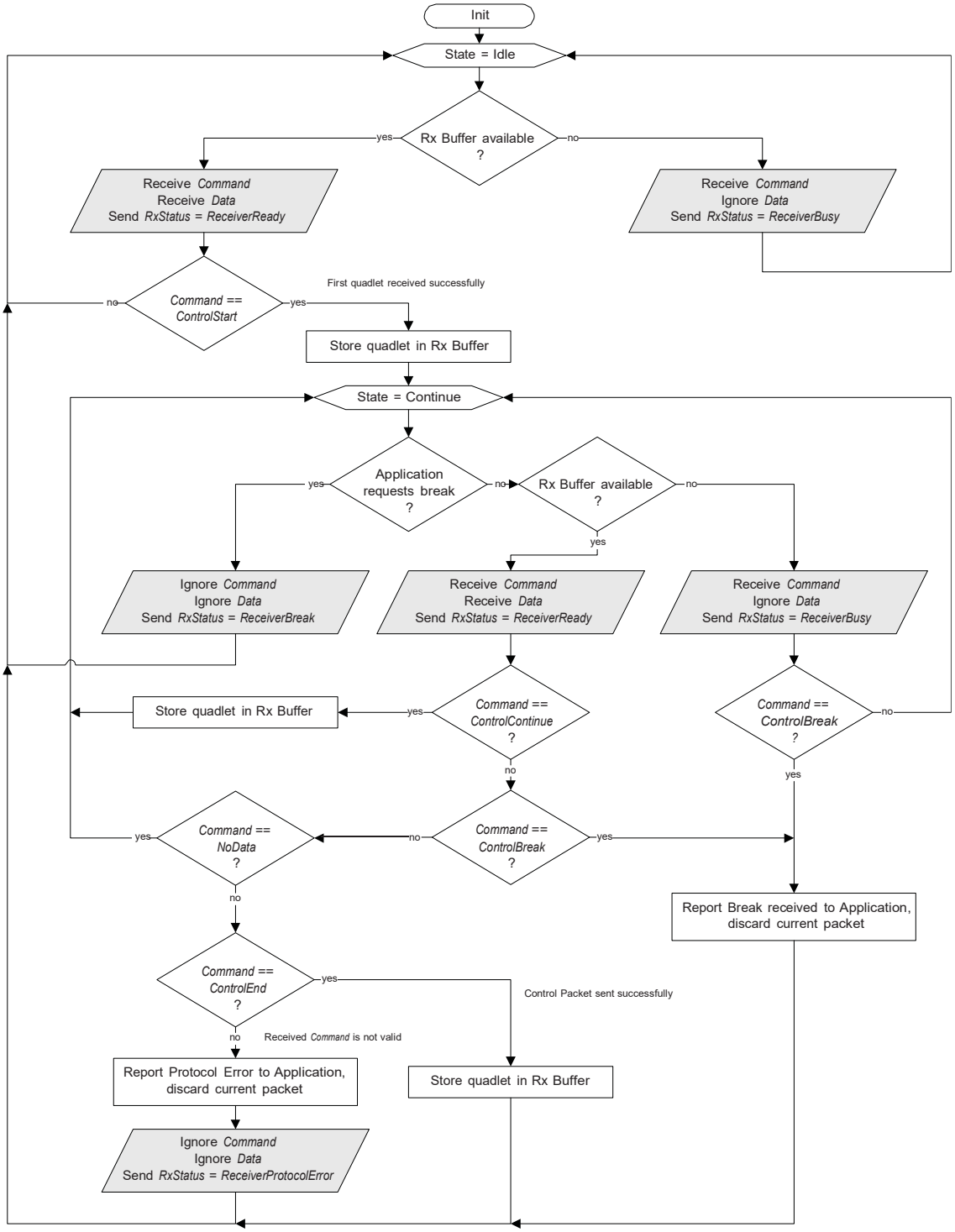


**Figure 48-9. Control Packet Rx Device Protocol**

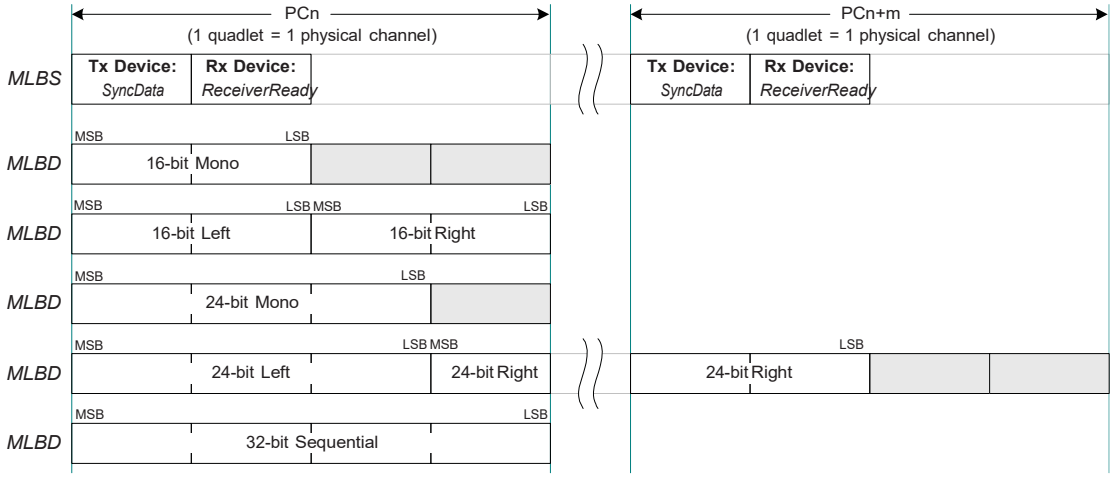



Synchronous

Synchronous stream data is sent in a continuous and broadcast fashion, without block information. Therefore, receiving Devices must not respond to the synchronous command; thereby leaving RxStatus in the ReceiverReady state (logic low). For 3-pin MediaLB, the required pull-down on MLBS leaves this signal in the ReceiverReady command when no synchronous data is transmitted on the MLBD line.

Figure 48-10 illustrates the synchronous data formats for MediaLB. For stereo 24-bit data, two physical channels (PCn) are needed per frame where the data is packed and left-justified in the two quadlets. In the 32-bit sequential format, data fills the entire quadlet with the internal data format determined by the system implementor.

**Figure 48-10. MediaLB Synchronous Data Structure**




The synchronous flow for a Tx Device is illustrated in Figure 48-11. The data transfer blocks (slanted rectangle shapes) occur only during a physical channel (PCn) associated with the logical channel defined by a single ChannelAddress. The flow diagram contains only one state: Transmit. Once a channel has been initialized, the Transmit state is entered. If a Tx Device has no data to transmit, it must still send the SyncData command and set the actual data to a safe value, such as all zeros. To stop sending synchronous data, the logical channel must be eliminated (ChannelAddress removed from MediaLB).

The synchronous flow for an Rx Device is illustrated in Figure 48-12. The flow diagram also contains only one state, Continue, where the Rx Device waits for data from the Tx Device. No command other than SyncData is expected or allowed. When the SyncData command is detected, the corresponding data quadlet sent with the command is received and stored in the Rx buffer. Any command received, other than SyncData, is a ProtocolError and should be reported to the application. Furthermore, the data quadlet received with the invalid command is discarded and replaced with a safe value.

Since the default bus state is ReceiverReady, the Rx Device must not drive the MLBS line with RxStatus since ReceiverReady is the only allowable response for synchronous data. The system stops the transfer of synchronous data by eliminating the logical channel (ChannelAddress) from the bus. If an Rx Device does not receive its ChannelAddress in the frame, it should assume that the channel is not setup yet, or that the logical channel has been eliminated and should respond accordingly (for example, mute outputs).

**Figure 48-11. Synchronous Data Tx Device Protocol**

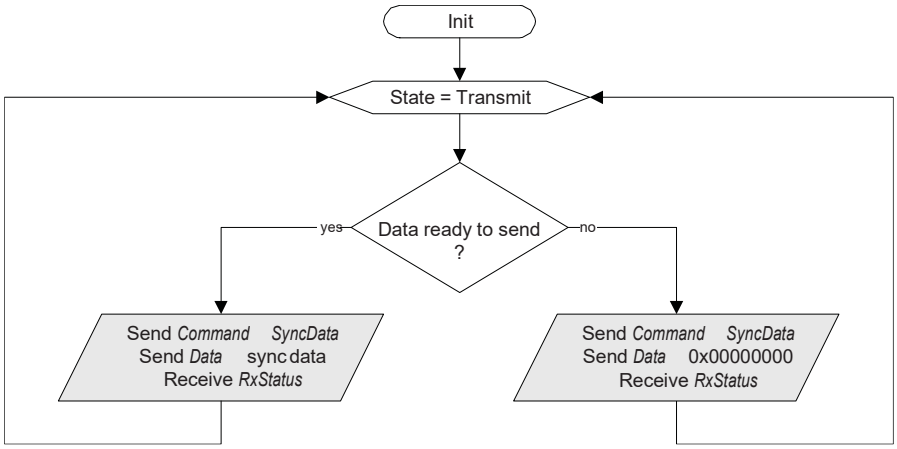
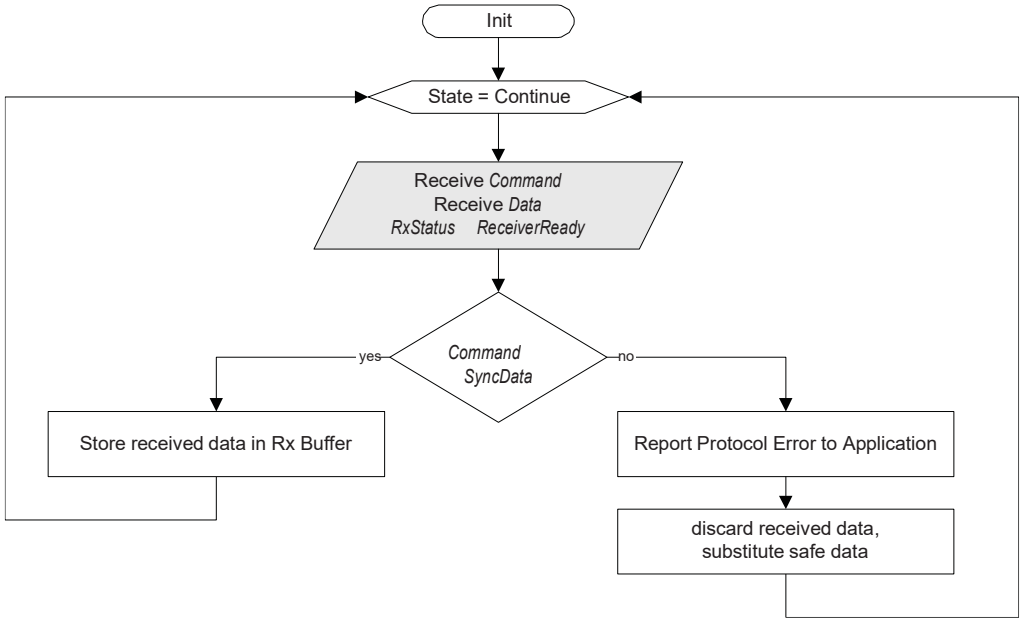


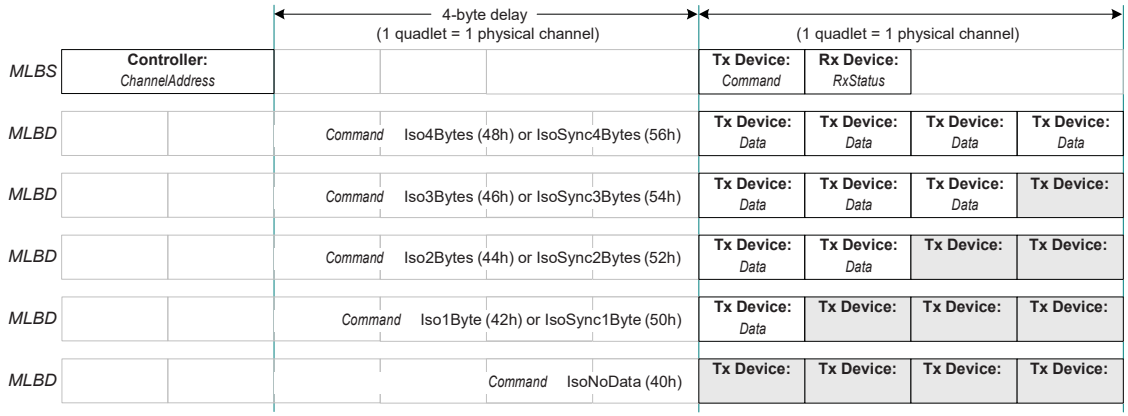
**Figure 48-12. Synchronous Data Rx Device Protocol**




Isochronous

Isochronous data is sent in a streaming fashion, similar to synchronous data. However, the isochronous commands indicate the start of a block and how many bytes are valid in the concurrent transmitted quadlet. Valid bytes are left-justified in the quadlet, as illustrated in Figure 48-13. When isochronous data is being transported (channel active), but no data is available for the current quadlet, the IsoNoData command is sent by the Tx Device.

**Figure 48-13. MediaLB Isochronous Data Structure**




The isochronous flow for a Tx Device is illustrated in Figure 48-14. The data transfer blocks (slanted rectangle shapes) occur only during a physical channel (PCn) associated with the logical channel defined by a single ChannelAddress. Similar to the synchronous flow, isochronous data immediately starts transmitting. When data exists from the application, the IsoSync?Bytes commands are used to indicate the start of a block, which provides alignment information to the Rx Device. The Iso?Bytes commands indicate the middle of a block of data. The definition of block for isochronous data is outside the scope of this document. For physical channels that transfer less than four bytes, the Rx Device must only use/store the number of valid bytes, and ignore the unused portion.

The isochronous flow for an Rx Device is illustrated in Figure 48-15. The NoData command indicates that the channel is not setup yet. Once an isochronous channel is setup, the Rx Device continually receives the channel data, similar to synchronous data. The only two valid responses for an isochronous channel are ReceiverBusy, and the default bus state of ReceiverReady. Although Rx Devices can respond with ReceiverBusy, its use should be minimized, since Tx Devices may not be able to store much isochronous data that gets backed up due to the ReceiverBusy responses. If any Rx Device uses ReceiverBusy, then only one Rx Device is allowed. If all targeted Rx Devices do

© 2021 Microchip Technology Inc.
and its subsidiaries

not drive RxStatus (default ReceiverReady response), then the isochronous stream can support multiple Rx Devices (broadcast).

**Figure 48-14. Isochronous Data Tx Device Protocol**

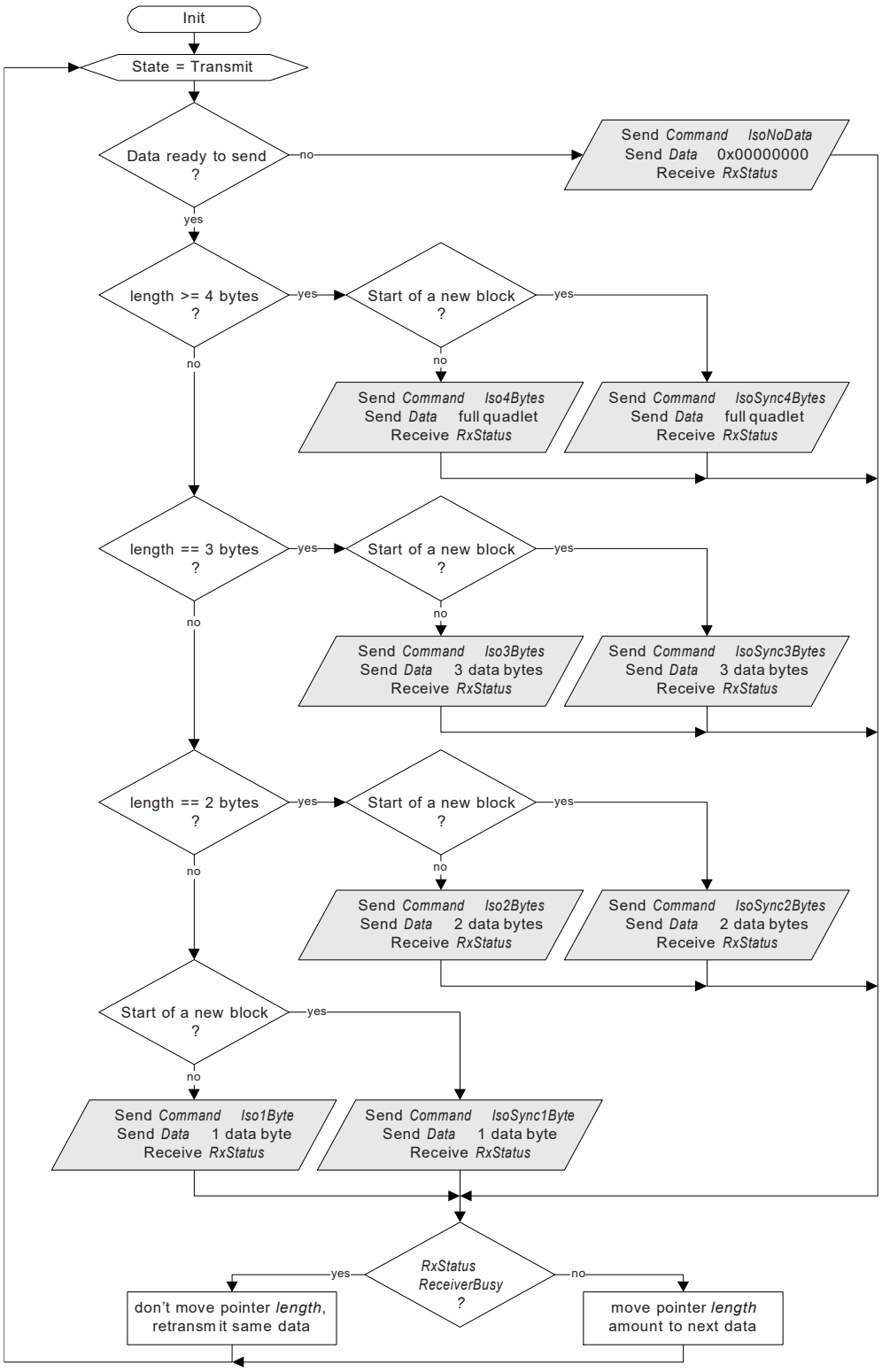
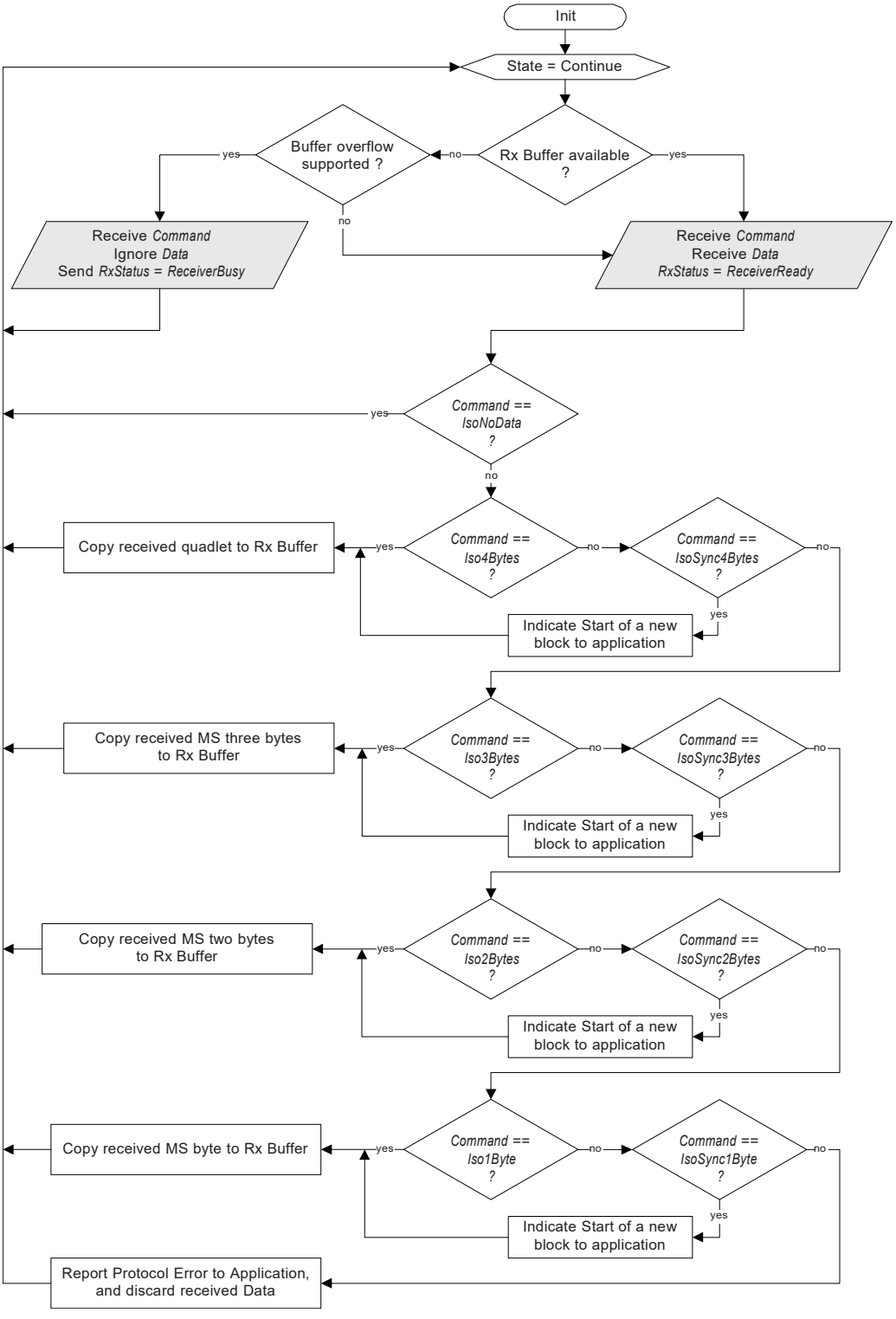


**Figure 48-15. Isochronous Data Rx Device Protocol**




## 48.6.2 Compliance

The MediaLB specification is targeted towards many levels of chip complexity and native intelligence. Therefore, different levels of implementation are allowed to support MediaLB and still remain compliant to this specification.

The Physical Layer portion of this specification must be met by all Devices for whichever speeds a particular Device supports. All MediaLB Devices must support the rules for synchronization to MediaLB.

For MediaLB Controllers, all System commands are optional, including support for dynamic system configuration and DeviceAddresses.

For MediaLB Devices, support for all transport methods is optional. If a MediaLB Device supports a particular transport method, it must fully support it including all Command bytes and RxStatus responses associated with that transport method. For asynchronous and control methods, the Protocol error responses can be expanded for additional error checking, based on specific implementations. Any extra error checking that causes a Protocol error to be transmitted must be listed in the Device documentation.

For MediaLB Devices, support for System responses and dynamic configuration are optional. If dynamic configuration is supported, it must comply with the specifications listed in this document.

All MediaLB Devices must specify clearly in documentation what MediaLB speeds, System commands, and transport methods they support. In addition, MediaLB Devices must clearly state the DeviceAddress as well as the Index and associated transport method used in configuring the ChannelAddress.

### 48.6.3 Internal Flow Description

The internal functional blocks of the MLB include:

- MediaLB Block (MLB PHY) - Implements the physical and link-layer requirements of a MediaLB 3-pin interface. Serial-to-parallel and parallel-to-serial data transformations are implemented, as well as MediaLB frame synchronization.
- Host Bus Interface Block (HBI) - Provides 16-bit parallel slave access to all MOST channels and data types for the external Host Controller (HC). The HBI supports up to 64 independent channels with a minimum access latency of 40 ns per word and a maximum bandwidth of 400 Mbps.
- Routing Fabric Block (RF) - Manages the flow of data between the MediaLB block and the HBI block, implementing a bus arbiter and multiplexing logic to the Channel Table RAM (CTR) and the Data Buffer RAM (DBR).
- Memory Interface Block (MIF) - Implements a bridge between the I/O bus and the customer-implemented RAMs (i.e. Channel Table and Data Buffer).
- Interrupt Interface Block (INTIF) - Sends notifications to HBI that there are changes to the channel descriptors.
- Clocks, Power, and Reset Block (CPR) - Implements clock and reset multiplexing and synchronization.
- AHB Block (AHB) - Implements a bus bridge between the AHB master and the HBI slave interfaces.
- APB Block (APB) - Implements a bus bridge that translates the two-cycle APB interface signals to the single-cycle I/O interface signals.

#### 48.6.3.1 MediaLB Block

The Media Local Bus (MediaLB) block supports a MediaLB 3-pin interface that provides real-time access to all network data types including streaming, packet, control, and isochronous data.

The MediaLB interface supports the MediaLB protocol for single-ended 3-pin mode, with a maximum data rate of 1024xFs (49.152 MHz at Fs=48 kHz).

MediaLB Channel Address to Logical Channel Mapping

The MediaLB channel addresses are mapped to the logical channels as follows:

**Table 48-8. MediaLB Channel Address to Logical Channel Mapping**

| Channel Address | Logical Channel |
|---|---|
| 0x0002 | 1 |
| 0x0004 | 2 |
| 0x0006 | 3 |
| .... | .... |
| 0x007C | 62 |

| .....continued | |
|---|---|
| **Channel Address** | **Logical Channel** |
| 0x007E | 63 |
| 0x01FE | 0(1) |

Note: 1. Logical Channel 0 is the System Channel and is reserved.

#### 48.6.3.2 Host Bus Interface Block

The Host Bus Interface (HBI) block provides a 16-bit parallel slave port that provides an external Host Controller (HC) with access to all MOST channels and data types.

Up to 64 independent HBI channels are available to the HC, each configurable for either transmitting or receiving a particular application data type (synchronous, isochronous, asynchronous, or control). The HBI block provides source and sink access to the full network data bandwidth.

**HBI Physical Addresses**

To access a particular HBI DMA channel, hardware must first translate the HBI channel address to a channel allocation table (CAT) physical address. This physical address is then used to retrieve the channel label (CL), which in turn retrieves the channel descriptor.

See the following table for more information on the mapping between the HBI channel address and physical address.

**Table 48-9. HBI Channel Address to Physical Address Mapping**

| HBI Channel | CAT Address | CAT Offset |
|---|---|---|
| 0x0 | 0x88 | 000 |
| 0x1 | 0x88 | 001 |
| 0x2 | 0x88 | 010 |
| 0x3 | 0x88 | 011 |
| 0x4 | 0x88 | 100 |
| 0x5 | 0x88 | 101 |
| 0x6 | 0x88 | 110 |
| 0x7 | 0x88 | 111 |
| 0x8 | 0x89 | 000 |
| ... | ... | ... |
| 0x3E | 0x8F | 110 |
| 0x3F | 0x8F | 111 |

#### 48.6.3.3 Routing Fabric Block

The Routing Fabric (RF) block manages the flow of data between the MediaLB Port and the HBI Port. Bus multiplexers and a bus arbiter are implemented in the RF block for accessing the channel table RAM (CTR) and data buffer RAM (DBR).

Each DMA controller in the routing fabric uses Channel Descriptors (stored in the CTR) to manage access to dynamic buffers in the DBR.

**Data Buffer RAM**

The MLB has an external data buffer RAM (DBR) that is 8-bit x 16k entries deep. The DBR provides dynamic circular buffering between the transmit and receive devices.

The size and location of each data buffer is defined by software in the channel descriptor table (CDT), which is located in the CTR.

Receive devices retain the write address pointer to the associated circular data buffer in the DBR, while transmit devices retain the read address pointer. The DMA controllers in the routing fabric are responsible for ensuring that the circular buffers do not overflow or underflow. Each channel type (e.g., synchronous, isochronous, asynchronous and control) has Full and Empty detection.

- Synchronous Channels
  For synchronous channels, two mechanisms prevent overflow and underflow of the data buffer:
  - Hardware aligns the read pointer (RPTR) to the write pointer (WPTR) to ensure an offset of two sub-buffers.
  - RPTR and WPTR are periodically synchronized to the start of the next sub-buffer (e.g. following a FRAMESYNC).
- Isochronous Channels
  For isochronous channels, hardware does not read from an empty data buffer or write to a full data buffer. The conditions used by hardware for detection include:

  Data buffer Empty condition: (RPTR = WPTR) AND (BF = 0), and

  Data buffer Full condition: (WPTR = RPTR) AND (BF = 1).
- Asynchronous and Control Channels
  For asynchronous and control channels, hardware does not read from an empty data buffer or write to a full data buffer. Hardware evaluates the DMA pointers (RPTR, WPTR) and packet count (RPC, WPC) to detect the data buffer condition, where:
  - Data buffer Empty condition: (RPTR = WPTR) AND (RPC = WPC), and
  - Data buffer Full condition: ((WPTR = RPTR) AND (WPC != RPC)) OR (WPC = (RPC - 1)).

**Channel Table RAM**

The MLB has an external Channel Table RAM (CTR) that is 128-bit x 144-entry. The CTR allows system software to dynamically configure channel routing and allocate data buffers in the DBR.

The CTR is logically divided into three sub-tables:

- Channel Descriptor Table (CDT)
- AHB Descriptor Table (ADT)
- Channel Allocation Table (CAT)

Address Mapping

**Table 48-10. CTR Address Mapping**

| Label | Address | Bits 127…96 | Bits 95…64 | Bits 63…32 | Bits 31…0 |
|---|---|---|---|---|---|
| Channel Descriptor Table (CDT): | | | | | |
| CDT | 0x00 | CDT0[127:0], CL = 0 | | | |
| | 0x01 | CDT1[127:0], CL = 1 | | | |
| | 0x02 | CDT2[127:0], CL = 2 | | | |
| | ... | ... | | | |
| | 0x3D | CDT61[127:0], CL = 61 | | | |
| | 0x3E | CDT62[127:0], CL = 62 | | | |
| | 0x3F | CDT63[127:0], CL = 63 | | | |
| AHB Descriptor Table (ADT): | | | | | |

| .....continued | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Label** | **Address** | **Bits 127…96** | | **Bits 95…64** | | **Bits 63…32** | | **Bits 31…0** | |
| ADT(1) | 0x40 | ADT0[127:0] | | | | | | | |
| | 0x41 | ADT1[127:0] | | | | | | | |
| | 0x42 | ADT2[127:0] | | | | | | | |
| | ... | ... | | | | | | | |
| | 0x7D | ADT61[127:0] | | | | | | | |
| | 0x7E | ADT62[127:0] | | | | | | | |
| | 0x7F | ADT63[127:0] | | | | | | | |
| Channel Allocation Table (CAT): | | | | | | | | | |
| CAT for MediaLB | 0x80 | CAT7 | CAT6 | CAT5 | CAT4 | CAT3 | CAT2 | CAT1 | CAT0 |
| | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| | 0x87 | CAT63 | CAT62 | CAT61 | CAT60 | CAT59 | CAT58 | CAT57 | CAT56 |
| CAT for HBI(1) | 0x88 | CAT71 | CAT70 | CAT69 | CAT68 | CAT67 | CAT66 | CAT65 | CAT64 |
| | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| | 0x8F | CAT127 | CAT126 | CAT125 | CAT124 | CAT123 | CAT122 | CAT121 | CAT120 |

Note: 1. A fixed relationship exists between ADT entries and HBI CAT entries. When using HBI channel 0 (CAT64) one should program ADT0. When using HBI channel 1 (CAT65) one should program ADT1, and so on.

Channel Allocation Table

The Channel Allocation Table (CAT) is comprised of 16 CTR entries (addresses 0x80–0x8F), as shown in Table 1-12. Each 16-bit CAT entry represents a logical connection to or from a transmit/receive device (e.g. MediaLB or HBI channel). All entries are indexed according to a fixed physical address assigned to every Rx/Tx channel (as shown in the following table). The value stored in a CAT entry includes a 6-bit Connection Label, which provides a pointer to the CDT. To complete a logical channel and form a routing connection, system software must assign the same Connection Label to both the Rx and Tx channels.

**Table 48-11. CAT Entry Map**

| Peripheral | Tx Channels | Rx Channels | CAT Start Index | CAT End Index | Entries |
|---|---|---|---|---|---|
| MediaLB | 0 to 64 | 64 - Tx Channels | 0 | 63 | 64 |
| HBI | 0 to 64 | 64 - Tx Channels | 64 | 127 | 64 |

The format of a full CAT entry is shown in Table 48-12, with field descriptions described in Table 48-13. All reserved bits of a CAT entry field should be written as zero.

**Table 48-12. CAT Entry Formats**

| Channel Type | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Isochronous | rsvd | FCE | rsvd | RNW | CE | CT[2:0] = 3 | | | rsvd | | CL[5:0] | | | | | |
| Asynchronous | rsvd | | MT | RNW | CE | CT[2:0] = 2 | | | rsvd | | CL[5:0] | | | | | |
| Control | rsvd | | MT | RNW | CE | CT[2:0] = 1 | | | rsvd | | CL[5:0] | | | | | |
| Synchronous | rsvd | MFE | MT | RNW | CE | CT[2:0] = 0 | | | rsvd | | CL[5:0] | | | | | |

**Table 48-13. CAT Field Definitions**

| Field | Description |
|---|---|
| CL[5:0] | Connection Label (offset into CDT) |
| CT[2:0] | Channel Type (Others):<br>111 = Reserved<br>110 = Reserved<br>101 = Reserved<br>100 = Reserved<br>011 = Isochronous<br>010 = Asynchronous<br>001 = Control<br>000 = Synchronous |
| CE | Channel Enable:<br>1 = Enabled<br>0 = Disabled |
| RNW | Read Not Write:<br>1 = Read<br>0 = Write |
| MT | Mute Enable (1):<br>1 = Enabled<br>0 = Disabled |
| FCE | Flow Control Enable (2):<br>1 = Enabled<br>0 = Disabled |
| MFE | Multi-Frame per Sub-buffer Enable(3): 1 = Enabled 0 = Disabled |
| rsvd | Reserved. Software writes a zero to all reserved bits when the entry is initialized. The reserved bits are Read-only after initialization. |

Notes: 1. When set for synchronous channels, the MT bit forces Rx channels to write zeros into the channel data buffer, and Tx channels to output zeros on the physical interface. When set for asynchronous and control channels, the MT bit causes DMA to halt at a packet boundary. Not valid for isochronous channels.

2. The FCE bit is used by MediaLB isochronous Rx channels only.

3. The MFE bit is used by MediaLB synchronous channels only.

Channel Setup

Data direction in the MLB is in reference to the DBR. Therefore, the data direction of CAT entries corresponding to the same channel is reversed for the HBI CAT and the MediaLB CAT.

For a Tx channel (from the HC to the MediaLB interface):

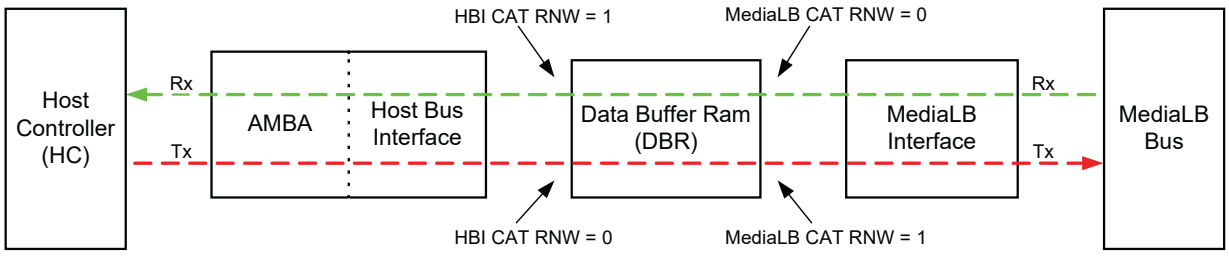
- HBI CAT entry: RNW = 0 (write)
- MediaLB CAT entry: RNW = 1 (read)

Conversely, for a Rx channel (data from MediaLB to HC):

- HBI CAT entry: RNW = 1 (read)
- MediaLB CAT entry: RNW = 0 (write)

The figure below illustrates the directional relationship in the MLB.

**Figure 48-16. MLB DBR Directional Relationship**




Channel Descriptor Table

The Channel Descriptor Table (CDT) is comprised of 64 CTR entries (addresses 0x00–0x3F), as shown in Table 48-10.

Each 128-bit CDT entry (also referred to as a Channel Descriptor) is referenced by a Connection Label and contains information about a data buffer in the DBR (e.g., buffer size, address pointers).

The format of each CDT entry (also referred to as a Channel Descriptor) depends on the channel type (e.g. synchronous, isochronous, asynchronous, or control).

Note: All reserved Channel Descriptor bits must be written to '0' by software when initialized.

Synchronous Channel Operation

The MLB provides two modes of operation (Standard and Multi-Frame per Sub- buffer) to provide flexibility for implementing synchronous channels.

Channels set up for Standard mode require less buffer space, but have higher interrupt rates and more stringent latency requirements. For channels configured for Standard mode, the Host Controller must transfer one full frame of streaming data in/out of each streaming channel's data buffer for each frame period.

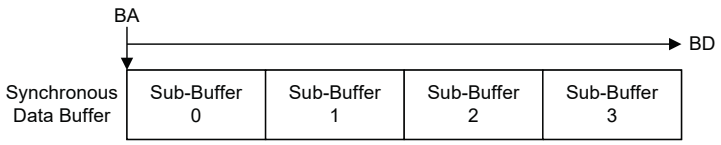
Channels set up for Multi-Frame per Sub-buffer mode require more buffer space, but have lower interrupt rates and less stringent latency requirements. For channels configured for Multi-Frame per Sub-buffer mode, the Host Controller must transfer N full frames of streaming data in/out of each streaming channel's data buffer for each frame period.

To set up a channel in Multi-Frame per Sub-buffer mode:

- Program MLB_MLBC0.FCNT[2:0] to select the number of frames per sub-buffer
- Program the CAT to enable multi-frame sub-buffering (MFE = 1) for each particular channel
- Set the buffer depth in the CDT: BD = 4 × m × bpf - 1,
  where m = frames per sub- buffer, bpf = bytes per frame
- Repeat for additional synchronous channels

A sample synchronous data buffer is shown in the following figure. Each data buffer contains four sub-buffers and each sub-buffer contains space for 1 to 64 frames of data, determined by MLB_MLBC0.FCNT[2:0].

**Figure 48-17. Synchronous Data Buffer Structure**




Synchronous Channel Descriptors

The format and field definitions for a synchronous CDT entry are shown in Table 48-14 and Table 48-14, respectively.

**Table 48-14. Synchronous CDT Entry Format**

| Bit Offset | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | WSBC | | Reserved | | | | | | | | | | | | | |
| 16 | RSBC | | Reserved | | | | | | | | | | | | | |

.....continued

| Bit Offset | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 32 | Reserved | | | | | | | | | | | | | | | |
| 48 | Reserved | | | | | | | | | | | | | | | |
| 64 | WSTS[3:0] | | | | WPTR[11:0] | | | | | | | | | | | |
| 80 | RSTS[3:0] | | | | RPTR[11:0] | | | | | | | | | | | |
| 96 | Reserved | | | | BD[11:0] | | | | | | | | | | | |
| 112 | Reserved | | BA[13:0] | | | | | | | | | | | | | |

**Table 48-15. Synchronous CDT Entry Field Definitions**

| Field | Description | Details | Accessibility |
|---|---|---|---|
| BA | Buffer Base Address | - BA can start at any byte in the 16k DBR | r,w |
| BD | Buffer Depth | - BD = size of buffer in bytes - 1<br>- Buffer end address = BA + BD<br>- BD = 4 x m x bpf - 1, where:<br>m = frames per sub-buffer (for MFE = 0, m = 1) bpf = bytes per frame. | r,w |
| RPTR | Read Pointer | - Software initializes to zero, hardware updates<br>- Counts the read address offset within a buffer<br>- DMA read address = BA + RPTR | r,w,u(1) |
| WPTR | Write Pointer | - Software initializes to zero, hardware updates<br>- Counts the write address offset within a buffer<br>- DMA write address = BA + WPTR | r,w,u (1) |
| RSBC | Read Sub-buffer Counter | - Software initializes to zero, hardware updates<br>- Counts the read sub-buffer offset<br>- DMA uses for pointer management | r,w,u (1) |
| WSBC | Write Sub-buffer Counter | - Software initializes to zero, hardware updates<br>- Counts the write sub-buffer offset<br>- DMA uses for pointer management | r,w,u (1) |
| RSTS | Read Status | - Software initializes to zero, hardware updates<br>- RSTS states:(2)<br>xxx0 = normal operation (no mute)<br>xxx1 = normal operation (mute)<br>xx0x = idle | r,w,u (1) |
| WSTS | Write Status | - Software initializes to zero, hardware updates<br>- WSTS states:(2)<br>xxx0 = normal operation (no mute)<br>xxx1 = normal operation (mute)<br>xx0x = idle<br>1xxx = command protocol error | r,w,u (1) |

| .....continued | | | |
|---|---|---|---|
| **Field** | **Description** | **Details** | **Accessibility** |
| Reserved | Reserved | - Software writes a zero to all reserved bits when the entry is initialized. The reserved bits are Read-only after initialization. | r,w,u (1) |

Notes: 1. “u” means “Updated periodically by hardware”.

2. Only valid for DMA pointers associated with the MediaLB block (Not valid for HBI block related pointers).

Isochronous Channel Descriptors

The format and field definitions for an isochronous CDT entry are shown in Table 48-16 and Table 48-17, respectively.

**Table 48-16. Isochronous CDT Entry Format**

| Bit Offset | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Reserved | | | | | | | | | | | | | | | |
| 16 | Reserved | | | | | | | | | | | | | | | |
| 32 | Reserved | | | | | | | BS[8:0] | | | | | | | | |
| 48 | Reserved | | | | | | | | | | | | | | | |
| 64 | WSTS[2:0] | | | WPTR[12:0] | | | | | | | | | | | | |
| 80 | RSTS[2:0] | | | RPTR[12:0] | | | | | | | | | | | | |
| 96 | Reserved | | | BD[12:0] | | | | | | | | | | | | |
| 112 | BF | rsvd | BA[13:0] | | | | | | | | | | | | | |

**Table 48-17. Isochronous CDT Entry Field Definitions**

| Field | Description | Details | Accessibility |
|---|---|---|---|
| BA | Buffer Base Address | - BA can start at any byte in the 16k DBR | r,w |
| BD | Buffer Depth | - BD = size of buffer in bytes - 1<br>- Buffer end address = BA + BD<br>- Isochronous buffers must be large enough to hold at least 3 blocks (packets) of data<br>- Buffer depth must be a integer multiple of blocks | r,w |
| BF | Buffer Full | - Software initializes to zero, hardware updates<br>- DMA write hardware sets BF when the buffer is full<br>- DMA read hardware clears BF when the buffer is empty<br>- BF is valid only when the buffer is full or empty, otherwise ignore | r,w,u (1) |
| BS | Block Size | - BS defines when to begin the DMA to the data buffer<br>- BS = buffer block size in bytes - 1<br>- For Rx channels, the DMA writes start when the number of empty bytes (SPACE) in the data buffer ≥ the block size<br>- For Tx channels, the DMA reads start when the number of valid bytes (VALID) in the data buffer ≥ the block size | r,w,u (1) |

| .....continued | | | |
|---|---|---|---|
| **Field** | **Description** | **Details** | **Accessibility** |
| RPTR | Read Pointer | - Software initializes to zero, hardware updates<br>- Counts the read address offset within a buffer<br>- DMA read address = BA + RPTR | r,w,u (1) |
| WPTR | Write Pointer | - Software initializes to zero, hardware updates<br>- Counts the write address offset within a buffer<br>- DMA write address = BA + WPTR | r,w,u (1) |
| RSTS | Read Status | - Software initializes to zero, hardware updates<br>- RSTS states:(2)<br>xx1 = active<br>xx0 = idle | r,w,u (1) |
| WSTS | Write Status | - Software initializes to zero, hardware updates<br>- WSTS states:(2)<br>xx1 = active<br>xx0 = idle<br>x1x = command protocol error<br>1xx = buffer overflow (FCE = 0 only) | r,w,u (1) |
| Reserved | Reserved | - Software writes a zero to all Reserved bits when the entry is initialized. The Reserved bits are Read-only after initialization. | r,w,u (1) |

Notes: 1. “u” means “Updated periodically by hardware”.

2. Only valid for DMA pointers associated with the MediaLB block (Not valid for HBI block related pointers).

Asynchronous and Control Channel Descriptors

The format and field definitions for asynchronous and control CDT entries are shown in Table 48-18 and Table 48-19, respectively.

**Table 48-18. Asynchronous/Control CDT Entry Format**

| Bit Offset | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | WPC[4:0] | | | | | Reserved | | | | | | | | | | |
| 16 | RPC[4:0] | | | | | Reserved | | | | | | | | | | |
| 32 | rsvd | WPC[7:5] | | | Reserved | | | | | | | | | | | |
| 48 | rsvd | RPC[7:5] | | | Reserved | | | | | | | | | | | |
| 64 | WSTS[3:0] | | | | WPTR[11:0] | | | | | | | | | | | |
| 80 | RSTS[3:0] | | | | RPTR[11:0] | | | | | | | | | | | |
| 96 | RSTS[4] | WSTS[4] | rsvd | | BD[11:0] | | | | | | | | | | | |
| 112 | Reserved | | BA[13:0] | | | | | | | | | | | | | |

**Table 48-19. Asynchronous/Control CDT Entry Field Definitions**

| Field | Description | Details | Accessibility |
|---|---|---|---|
| BA | Buffer Base Address | - BA can start at any byte in the 16k DBR | r,w |

.....continued

| Field | Description | Details | Accessibility |
|---|---|---|---|
| BD | Buffer Depth | - BD = size of buffer in bytes - 1<br>- Buffer end address = BA + BD<br>- BD ≥ max packet length - 1 | r,w |
| RPC | Read Packet Count | - Software initializes to zero, hardware updates<br>- Used in conjunction with WPC, RPTR and WPTR to determine if the buffer is empty or full | r,w,u (1) |
| WPC | Write Packet Count | - Software initializes to zero, hardware updates<br>- Used in conjunction with RPC, RPTR and WPTR to determine if the buffer is empty or full | r,w,u (1) |
| RPTR | Read Pointer | - Software initializes to zero, hardware updates<br>- Counts the read address offset within a buffer<br>- DMA read address = BA + RPTR | r,w,u (1) |
| WPTR | Write Pointer | - Software initializes to zero, hardware updates<br>- Counts the write address offset within a buffer<br>- DMA read address = BA + WPTR | r,w,u (1) |
| RSTS | Read Status | - Software initializes to zero, hardware updates<br>- Status states:(2)<br>x0x00 = idle<br>xx1xx = ReceiverProtocolError response received from Rx Device<br>1xxxx = ReceiverBreak command received from Rx Device | r,w,u (1) |
| WSTS | Write Status | - Software initializes to zero, hardware updates<br>- Status states:(2)<br>x0x00 = idle<br>xx1xx = command protocol error detected<br>1xxxx = AsyncBreak/ControlBreak command received from Tx Device | r,w,u (1) |
| Reserved | Reserved | Software writes a zero to all reserved bits when the entry is initialized. The reserved bits are Read-only after initialization. | r,w,u (1) |

Notes: 1. “u” means “Updated periodically by hardware”.

2. Only valid for DMA pointers associated with the MediaLB block (not valid for HBI block related pointers).

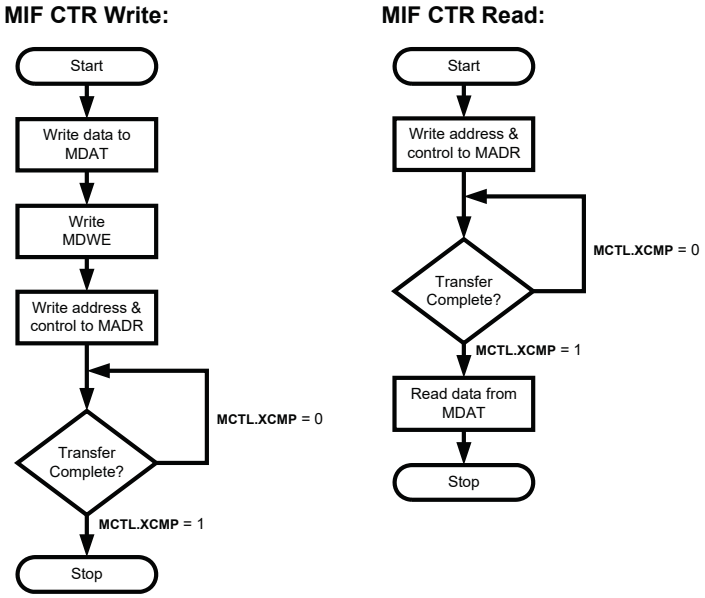
#### 48.6.3.4 Memory Interface Block

The Memory Interface (MIF) block implements a bridge between the I/O and the CTB or DBB interfaces.

**CTR Access**

The MIF block allows the HC to directly access the external Channel Table RAM (CTR) when MLB_MADR.TB is cleared. Any write to the MLB_MADR register triggers a single read or write cycle. Reading from the MLB_MADR register does not initiate read/write access.

**Figure 48-18. MIF CTR Read and Write Flow Diagrams**




**Direct CTR Writes**

For a direct write of the CTR, the HC first loads the 128-bit data entry into the MLB_MDAT0–3 registers. Bitwise write enable control is available via the MLB_MDWE0–3 registers.

After the MDATn and MDWEn registers are set up, a write cycle is initiated by writing the address and control information to MLB_MADR as follows:

- MLB_MADR.WNR = 1
- MLB_MADR.TB = 0
- MLB_MADR.ADDR[7:0] = 8-bit Target Address

The MIF block sets MLB_MCTL.XCMP = 1 to inform the HC when the write is complete.

**Direct CTR Reads**

For a direct read of the CTR, the HC initiates a read cycle by writing the address and control information to MLB_MADR as follows:

- MLB_MADR.WNR = 0
- MLB_MADR.TB = 0
- MLB_MADR.ADDR[7:0] = 8-bit Target Address

The MIF block sets MLB_MCTL.XCMP = 1 to inform the HC when the read is complete. The HC can then read the 128-bit data entry from the MLB_MDAT0–3 registers.

**CTR Addressing**

The CTR is addressed as a 128-bit wide value. However, the MIF block can only access 32 bits of the addressed CTR data in a single access. Therefore, four 32-bit accesses through the MIF block are required to access a single 128-bit value (e.g. CDT entry).

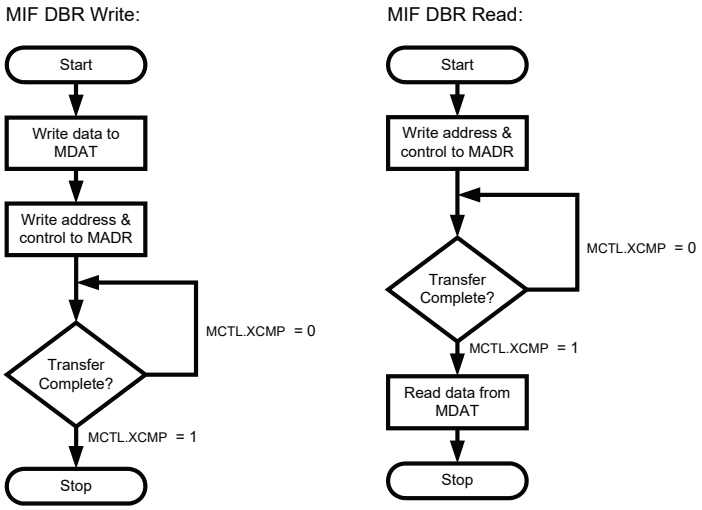
To access a 16-bit CAT entry in the CTR, only a single access through the MIF is required. For example, to load a CAT61 entry for an isochronous Tx channel with mute and flow control enabled:

- Write MLB_MDAT2 = 7B070000h (assumes Connection Label = 7)
- MLB_MDWE2 = FFFF0000h (bitwise write enable for 16 msbs; assumes MLB_MDWE0/1/3 =00000000h)
- MLB_MADR = 80000087h (write CTR address 87h)

**DBR Access**

The MIF block allows the HC to access the external Data Buffer RAM (DBR) directly when MLB_MADR.TB is set. Any write to the MLB_MADR triggers a single read or write cycle. Reading from the MLB_MADR register does not initiate read/write access.

**Figure 48-19. MIF DBR Read and Write Flow Diagrams**




**Direct DBR Writes**

For a direct write of the DBR, the HC first loads the 8-bit data entry into the MLB_MDAT0 register at bits[7:0]. MLB_MDAT1–3 and MLB_MDWE0–3 are not used for DBR access.

After the MLB_MDAT0 register is set up, a write cycle is initiated by writing the address and control information to MLB_MADR as follows:

- MLB_MADR.WNR = 1
- MLB_MADR.TB = 1
- MLB_MADR.ADDR[13:0] = 14-bit Target Address

The MIF block sets MLB_MCTL.XCMP = 1 to inform the HC when the write is complete.

**Direct DBR Reads**

For a direct read of the DBR, the HC initiates a read cycle by writing the address and control information to MLB_MADR as follows:

- MLB_MADR.WNR = 0
- MLB_MADR.TB = 1
- MLB_MADR.ADDR[13:0] = 14-bit target address

The MIF block sets MLB_MCTL.XCMP = 1 to inform the HC when the read is complete. The HC can then read the 8-bit data entry from the MLB_MDAT0 register at bits[7:0].

### 48.6.3.5 Interrupt Interface Block

The Interrupt Interface (INTIF) block performs a low-priority polling algorithm of each of the HBI channel descriptors.

The INTIF alerts the HBI block when specific changes to HBI Channel Descriptors occur.

- For asynchronous and control read/write channels:
  - a packet is available to read in the channel buffer, or
  - sufficient empty space is available in the channel buffer to accept a requested packet write.
- For isochronous read/write channels:
  - the number of valid bytes in the channel buffer exceeds the block size, or
  - the number of empty bytes in the channel buffer exceeds the block size.

#### 48.6.3.6 AHB Block

The AHB block manages data exchange between local channel data buffers within the MLB and the system memory buffer.

To support system memory buffering, a ping-pong memory structure is implemented on a per-channel basis using 128-bit descriptors for AHB Descriptor Table (ADT) entries.

**Note:** The 64 ADT entries are directly mapped to the 64 HBI physical channels.

Each logical channel is assigned a separate 128-bit descriptor, defining the data buffers in the system memory used by the DMA interface for that channel. The descriptors are stored at fixed addresses in the external CTR.

**AHB Descriptor Table**

The following table provides an overview of field definitions for ADT entries.

**Table 48-20. ADT Field Definitions**

| Field | No. of Bits | Description | Accessibility |
|---|---|---|---|
| CE | 1 | Channel enable:<br>0 = Disabled<br>1 = Enabled | r,w,u (1) |
| LE | 1 | Endianess select:<br>0 = Big Endian<br>1 = Little Endian | r,w |
| PG | 1 | Page pointer. Software initializes to zero, hardware writes thereafter.<br>0 = Ping buffer<br>1 = Pong buffer | r,w,u (1) |
| RDY1 | 1 | Buffer ready bit for ping buffer page:<br>0 = Not ready<br>1 = Ready | r,w |
| RDY2 | 1 | Buffer ready bit for pong buffer page:<br>0 = Not ready<br>1 = Ready | r,w |
| DNE1 | 1 | Buffer done bit for ping buffer page:<br>0 = Not done<br>1 = Done | r,u (1),c0 |
| DNE2 | 1 | Buffer done bit for pong buffer page:<br>0 = Not done<br>1 = Done | r,u (1),c0 |
| ERR1 | 1 | AHB error response detected for ping buffer page:<br>0 = No error<br>1 = Error | r,u (1),c0 (2) |
| ERR2 | 1 | AHB error response detected for pong buffer page:<br>0 = No error<br>1 = Error | r,u (1),c0 (2) |

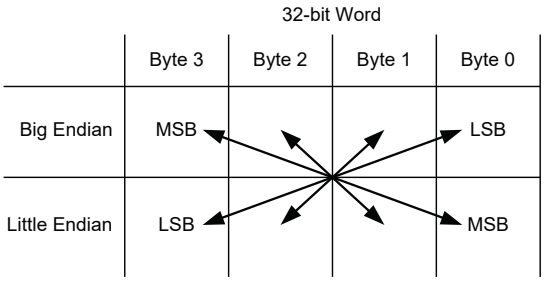
| .....continued | | | |
|---|---|---|---|
| **Field** | **No. of Bits** | **Description** | **Accessibility** |
| PS1 | 1 | Packet start bit for ping buffer page:<br>0 = No packet start<br>1 = Packet start<br>Reserved for synchronous and isochronous channels. | r,w,u (1) (both Tx and Rx) |
| PS2 | 1 | Packet start bit for pong buffer page:<br>0 = No packet start<br>1 = Packet start<br>Reserved for synchronous and isochronous channels. | r,w,u (1) (both Tx and Rx) |
| MEP1 | 1 | Most Ethernet Packet (MEP) indicator for ping buffer page:<br>0 = Not MEP<br>1 = MEP<br>MEP1 only valid for the first page of a segmented buffer.<br>Reserved for control, synchronous and isochronous channels. | Rsvd for Tx<br>r,u (1),c0 (2) for Rx |
| MEP2 | 1 | MEP packet indicator for pong buffer page:<br>0 = not MEP<br>1 = MEP MEP2 only valid for the first page of a segmented buffer.<br>Reserved for control, synchronous and isochronous channels. | Reserved for Tx<br>r,u (1),c0 (2) for Rx |
| BD1(2) | 11 to 13 | Buffer depth for ping buffer page:<br>11 or 12-bits for asynchronous and control channels.<br>13-bits for synchronous and isochronous channels. | r,w |
| BD2(2) | 11 to 13 | Buffer depth for pong buffer page:<br>11 or 12-bits for asynchronous and control channels.<br>13-bits for synchronous and isochronous channels. | r,w |
| BA1 | 32 | Buffer base address for ping buffer page | r,w |
| BA2 | 32 | Buffer base address for pong buffer page | r,w |
| Reserved | varies | Software writes a zero to all Reserved bits when the entry is initialized. The reserved bits are Read-only after initialization. | r,w,u (1) |

**Notes:**

1. “u” means “Updated periodically by hardware”.
2. “c0” means “Cleared by writing a 0”.
3. The buffer depth (BD1 and BD2) for synchronous channels must consider if Multi-Frame per Sub-buffer mode is enabled.

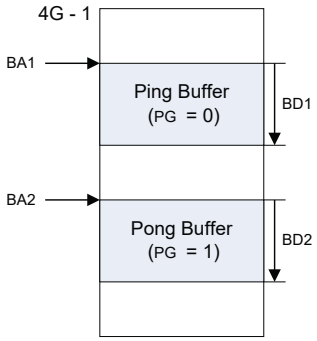
Data exchange across the AHB interface can be configured as Little Endian (LE = 1) or Big Endian (LE = 0). The following figure provides an overview of the endian options, chosen by an ADT descriptor field.

**Figure 48-20. Endianness Overview**




The following figure shows an example of the ping-pong system memory structure. This system memory structure is similar for all channel types and shows the relationship between the BAn, BDn, and PG descriptor fields.

**Figure 48-21. Ping-Pong System Memory Structure**




Each ADT entry holds a 32-bit BAn field which defines the start of each ping or pong buffer within system memory. The BDn field is used to indicate the size for the respective ping or pong page. The maximum size is 2k-entries for asynchronous and control channels; 8k-entries for isochronous and synchronous channels.

### AHB Synchronous Channel Descriptors

Table 48-21 shows the format for a synchronous ADT entry. The field definitions are defined in Table 48-22. Each synchronous channel buffer can be up to 8k-bytes deep.

**Table 48-21. Synchronous ADT Entry Format**

| Bit Offset | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | CE | LE | PG | Reserved | | | | | | | | | | | | |
| 16 | Reserved | | | | | | | | | | | | | | | |
| 32 | RDY1 | DNE1 | ERR1 | BD1[12:0] | | | | | | | | | | | | |
| 48 | RDY2 | DNE2 | ERR2 | BD2[12:0] | | | | | | | | | | | | |
| 64 | BA1[15:0] | | | | | | | | | | | | | | | |
| 80 | BA1[31:16] | | | | | | | | | | | | | | | |
| 96 | BA2[15:0] | | | | | | | | | | | | | | | |
| 112 | BA2[31:16] | | | | | | | | | | | | | | | |

### AHB Isochronous Channel Descriptors

The isochronous buffering scheme allows each ping or pong buffer to contain a single block or a multiple number of blocks. For this reason, the isochronous buffer depth (BDn) must be defined in terms of an integer number (n) and block size (BS) (e.g. BDn = n x (BS + 1) - 1).

Table 48-22 shows the format for an isochronous ADT entry. The field definitions are defined in Table 48-23. Each isochronous channel buffer can be up to 8k-bytes deep.

**Table 48-22. Isochronous ADT Entry Format**

| Bit Offset | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | CE | LE | PG | Reserved | | | | | | | | | | | | |
| 16 | Reserved | | | | | | | | | | | | | | | |
| 32 | RDY1 | DNE1 | ERR1 | BD1[12:0] | | | | | | | | | | | | |
| 48 | RDY2 | DNE2 | ERR2 | BD2[12:0] | | | | | | | | | | | | |
| 64 | BA1[15:0] | | | | | | | | | | | | | | | |
| 80 | BA1[31:16] | | | | | | | | | | | | | | | |
| 96 | BA2[15:0] | | | | | | | | | | | | | | | |
| 112 | BA2[31:16] | | | | | | | | | | | | | | | |

### AHB Asynchronous and Control Channel Descriptors

Every asynchronous and control packet adheres to the Port Message Protocol (PMP), which designates the first two bytes of each packet as the packet length (PML). Each packet must be no more than 2048 bytes.

Software must set the buffer ready bit (RDYn) for each buffer as it programs the DMA. As hardware processes each buffer, it sets the done bit (DNEn) and generates an interrupt to inform HC. When hardware finishes processing a buffer it can begin processing another buffer if RDYn is set. The application is responsible for setting up and configuring the channel buffer descriptor prior to every DMA access on the channel.

Two packet modes are supported by hardware for programming the DMA, single-packet mode and multiple-packet mode.

### Single-packet Mode

The single-packet mode asynchronous and control buffering scheme supports a maximum of one packet per buffer (e.g. ping or pong). Both non-segmented and segmented data packets are allowed while using single-packet mode.

Non-segmented packets are exchanged when only one buffer (e.g. ping or pong) is needed for packet transfer. Segmented packets are exchanged when a single packet is too long for one buffer and the packet must span multiple buffers. The following figure shows the memory space usage for both non-segmented and segmented asynchronous or control packets along with the packet start bit (PSn). While using single-packet mode, buffer done (DNEn) is set in hardware when a packet is done or the buffer is full.

shows the format for single-packet mode asynchronous and control ADT entries. The field definitions are defined in Table 48-23.

**Figure 48-22. Single-packet Asynchronous or Control System Memory Structure**

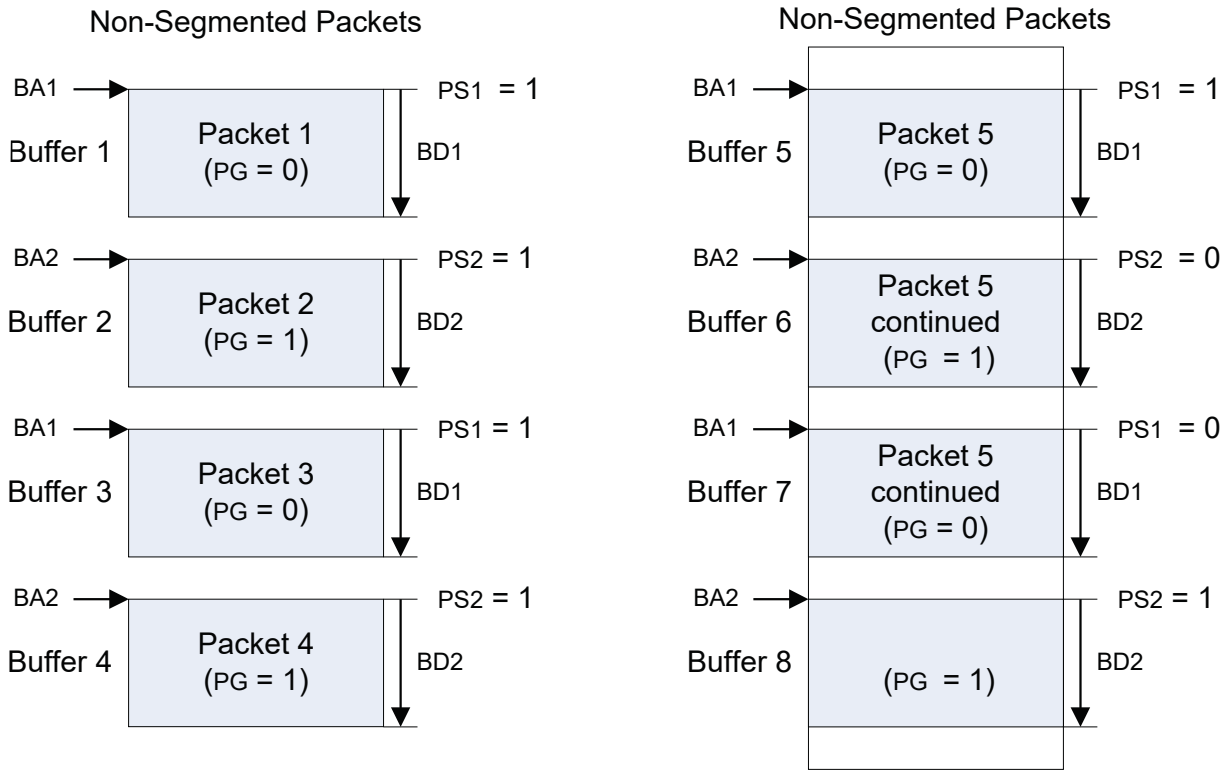



**Table 48-23. Single-packet Asynchronous and Control Entry Format**

| Bit Offset | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | CE | LE | PG | Reserved | | | | | | | | | | | | |
| 16 | Reserved | | | | | | | | | | | | | | | |
| 32 | RDY1 | DNE1 | ERR1 | PS1 | MEP1 | BD1[10:0] | | | | | | | | | | |
| 48 | RDY2 | DNE2 | ERR2 | PS2 | MEP2 | BD2[10:0] | | | | | | | | | | |
| 64 | BA1[15:0] | | | | | | | | | | | | | | | |
| 80 | BA1[31:16] | | | | | | | | | | | | | | | |
| 96 | BA2[15:0] | | | | | | | | | | | | | | | |
| 112 | BA2[31:16] | | | | | | | | | | | | | | | |

**Multiple-packet Mode**

The multiple-packet mode asynchronous and control buffering scheme supports more than one packet per system memory buffer, as shown in the following figure. Multiple- packet mode reduces the interrupt rate for packet channels at the cost of increasing buffering and latency.

For Tx packet channels in multiple-packet mode, software sets the packet start bit (PSn) for every buffer. Setting PSn informs hardware that the first two bytes of the buffer contains the port message length (PML) of the first packet. After the first packet, hardware keeps track of where packets start and end within the current buffer. Software should not write to PSn while the buffer is active (RDYn = 1 and DNEn = 0). For Tx packet channels, the buffer is done (DNEn= 1) when the last byte of the last packet in the buffer is read from system memory. Software should set the buffer depth to contain the exact number of complete packets for that buffer. Segmented buffers are not supported for Tx packet channels in multiple-packet mode.

© 2021 Microchip Technology Inc.
and its subsidiaries

For Rx packet channels in multiple-packet mode, PSn has no meaning and should be ignored. Software is responsible for keeping track of where each packet starts and ends within the multiple-packet buffer via the packet PML. The buffer done bit (DNEn) is set in hardware for Rx channels when a buffer is full (see Buffer 1 in Figure 48-23) or if a packet ends exactly 1-byte before the end of the buffer (see Buffer 2 in Figure 48-23). Multiple-packet mode also supports segmented Rx packets spanning two or more buffers (see Buffers 3–6 in Figure 48-23).

Table 48-24 shows the format for multiple-packet mode asynchronous and control ADT entries. The field definitions are defined in Table 48-20.

**Figure 48-23. Multiple-packet Asynchronous or Control System Memory Structure**

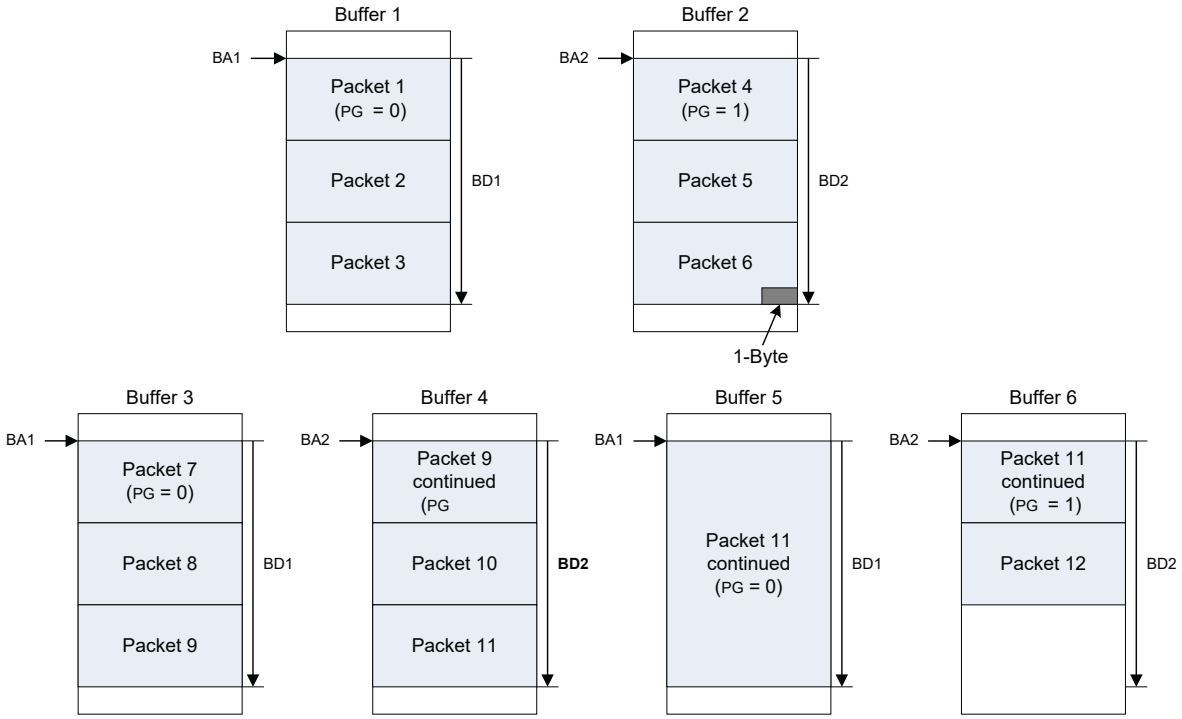



**Table 48-24. Multiple-packet Asynchronous and Control Entry Format**

| Bit Offset | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | CE | LE | PG | Reserved | | | | | | | | | | | | |
| 16 | Reserved | | | | | | | | | | | | | | | |
| 32 | RDY1 | DNE1 | ERR1 | PS1(1) | BD1[11:0] | | | | | | | | | | | |
| 48 | RDY2 | DNE2 | ERR2 | PS2(1) | BD2[11:0] | | | | | | | | | | | |
| 64 | BA1[15:0] | | | | | | | | | | | | | | | |
| 80 | BA1[31:16] | | | | | | | | | | | | | | | |
| 96 | BA2[15:0] | | | | | | | | | | | | | | | |
| 112 | BA2[31:16] | | | | | | | | | | | | | | | |

**Note:** PSn is only valid for TX channels. Set PSn = 1 at the start of the buffer.

### 48.6.4 Software Flow

The top-level software tasks the application must perform can be placed in two categories:

- Channel Initialization
- Channel Servicing

#### 48.6.4.1 Channel Initialization

The software flow required to initialize a channel must be performed in order to ensure proper operation.

For clarity, the software flow is grouped as follows:

- Configure the Hardware
- Program the Routing Fabric Block
- Program the AHB Block DMAs
- Synchronize and Unmute Synchronous Channel

**Configure the Hardware**

The MLB_MLBC0, HMCR0, HMCR1 and MLB_HCTL registers are accessible directly via APB reads and writes.

1. Initialize CTR and registers
   - 1.1. Clear CAT, CDT, and ADT bits in CTR
   - 1.2. Clear all bits of all registers
2. Configure the MediaLB interface
   - 2.1. Select MediaLB clock speed via MLB_MLBC0.MLBCLK
   - 2.2. Set MediaLB enable via MLB_MLBC0.MLBEN
3. Configure the HBI interface
   - 3.1. Set HMCR0 and HMCR1 = FFFFFFFFh to activate all channels
   - 3.2. Set the HBI enable bit: MLB_HCTL.EN = 1

**Program the Routing Fabric Block**

The CAT and CDT reside in the external CTR and are programmed indirectly via APB or I/O reads and writes to the MIF block.

1. Clear all bits of the CAT
2. Select a logical channel: N = 0–63
3. Program the CDT for channel N
   - 3.1. Set the 14-bit base address (BA)
   - 3.2. Set the 12-bit or 13-bit buffer depth (BD): BD = buffer depth in bytes - 1
     - 3.2.1. For synchronous channels: (BD + 1) = 4 x frames per sub-buffer (m) x bytes- per-frame (bpf)
     - 3.2.2. For isochronous channels: (BD + 1) mod (BS + 1) = 0
     - 3.2.3. For asynchronous channels: (BD + 1) ≥ max packet length (1024 for a MOST Data Packet (MDP); 1536 for a MOST Ethernet Packet (MEP))
     - 3.2.4. For control channels: (BD + 1) ≥ max packet length (64)
   - 3.3. For isochronous channels, set the block size (BS): BS = block size in bytes - 1
   - 3.4. Clear all other bits of the CDT
4. Program the CAT for the inbound DMA
   - 4.1. For Tx channels (to MediaLB) HBI is the inbound DMA
   - 4.2. For Rx channels (from MediaLB) MediaLB is the inbound DMA
   - 4.3. Set the channel direction: RNW = 0
   - 4.4. Set the channel type:
     CT[2:0] = 010 (asynchronous), 001 (control), 011 (isochronous), or 000 (synchronous)
   - 4.5. Set the connection label: CL[5:0] = N
   - 4.6. If CT[2:0] = 000 (synchronous), set the mute bit (MT = 1)
   - 4.7. Set the channel enable: CE = 1
   - 4.8. Set all other bits of the CAT to ‘0’
5. Program the CAT for the outbound DMA
   - 5.1. For Tx channels (to MediaLB) MediaLB is the outbound DMA
   - 5.2. For Rx channels (from MediaLB) HBI is the outbound DMA
   - 5.3. Set the channel direction: RNW = 1

5.4. Set the channel type: CT[2:0] = 010 (asynchronous), 001 (control), 011 (isochronous), or 000 (synchronous)
5.5. Set the channel label: CL[5:0] = N
5.6. If CT[2:0] = 000 (synchronous), set the mute bit (MT = 1)
5.7. Set the channel enable: CE = 1
5.8. Set all other bits of the CAT to '0'
6. Repeat steps 2–5 to initialize all logical channels

**Program the AHB Block DMAs**

The ADT resides in the external CTR and is programmed indirectly via APB reads and writes to the MIF.

1. Initialize all bits of the ADT to '0'
2. Select a logical channel: N = 0–63
3. Program the AHB block ping page for channel N
   3.1. Set the 32-bit base address (BA1)
   3.2. Set the 11-bit buffer depth (BD1): BD1 = buffer depth in bytes - 1
      3.2.1. For synchronous channels: (BD1 + 1) = n x frames per sub-buffer (m) x bytes-per-frame (bpf)
      3.2.2. For isochronous channels: (BD1 + 1) mod (BS + 1) = 0
      3.2.3. For asynchronous channels: 5 ≤ (BD1 + 1) ≤ 4096 (max packet length)
      3.2.4. For control channels: 5 ≤ (BD1 + 1) ≤ 4096 (max packet length)
   3.3. For asynchronous and control Tx channels set the packet start bit (PS1) iff the page contains the start of the packet
   3.4. Clear the page done bit (DNE1)
   3.5. Clear the error bit (ERR1)
   3.6. Set the page ready bit (RDY1)
4. Program the AHB block pong page for channel N
   4.1. Set the 32-bit base address (BA2)
   4.2. Set the 11-bit buffer depth (BD2): BD2 = buffer depth in bytes - 1
      4.2.1. For synchronous channels: (BD2 +1) = n x frames per sub-buffer (m) x bytes-per-frame (bpf)
      4.2.2. For isochronous channels: (BD2 + 1) mod (BS + 1) = 0
      4.2.3. For asynchronous channels: 5 ≤ (BD2 + 1) ≤ 4096 (max packet length)
      4.2.4. For control channels: 5 ≤ (BD2 + 1) ≤ 4096 (max packet length)
   4.3. For asynchronous and control Tx channels set the packet start bit (PS2) if the page contains the start of the packet
   4.4. Clear the page done bit (DNE2)
   4.5. Clear the error bit (ERR2)
   4.6. Set the page ready bit (RDY2)
5. Select Big Endian (LE = 0) or Little Endian (LE = 1)
6. Select the active page: PG = 0 (ping), PG = 1 (pong)
7. Set the channel enable (CE) bit for all active logical channels
8. Repeat steps 2–7 for all active logical channels

**Note:** All asynchronous and control packets must start with a PMP header. The first two bytes of the PMP header contains the Port Message Length (PML), which defines the length of the message that follows in bytes (not including PML itself). Hardware uses the PML to determine when a packet is complete. Asynchronous and control packets can also be segmented into two or more pages as well as contain multiple packets per page within system memory.

**Synchronize and Unmute Synchronous Channel**

The MLB_MLBC0 and MLB_MLBC1 registers are accessible directly via APB reads and writes.

1. Check that MediaLB clock is running (MLB_MLBC1.CLKM = 0)
2. If MLB_MLBC1.CLKM = 1, clear the register bit, wait one APB or I/O clock cycle and repeat step 1.

© 2021 Microchip Technology Inc. and its subsidiaries

3. Poll for MediaLB lock (MLB_MLBC0.MLBLK = 1)
4. Wait four frames
5. Unmute synchronous channel(s)

#### 48.6.4.2 Channel Servicing

After initialization, each channel will require periodic servicing.

The following software flows can be performed concurrently and in any order:

- Servicing the AHB Block (DMA) Interrupts
- Servicing the MediaLB Interrupts
- Polling for MediaLB System Commands

**Servicing the AHB Block (DMA) Interrupts**

The MLB_ACMR0, MLB_ACMR1, MLB_ACTL, MLB_ACSR0, and MLB_ACSR1 registers are accessible directly via APB reads and writes.

1. Program the MLB_ACMRn registers to enable interrupts from all active DMA channels.
2. Select the status clear method: MLB_ACTL.SCE = 0 (hardware clears on read), MLB_ACTL.SCE = 1 (software writes a '1' to clear).
3. Select 1 or 2 interrupt signals: MLB_ACTL.SMX = 0 (one interrupt for channels 0–31 on MediaLB IRQ0 and another interrupt for channels 32–63 on MediaLB IRQ1), MLB_ACTL.SMX = 1 (single interrupt for all channels on MediaLB IRQ0).
4. Wait for an interrupt from MediaLB IRQ[1:0].
5. Read the MLB_ACSRn registers to determine which channel or channels are causing the interrupt.
6. If MLB_ACTL.SCE = 1, write the results of step 5 back to MLB_ACSR0 and MLB_ACSR1 to clear the interrupt.
7. Select a logical channel (N = 0–63) with an interrupt to service.
8. Read the ADT entry for channel N
    8.1. Determine the active page (ping or pong) via the PG bit.
    8.2. Determine which page(s) are done via the DNEn bits.
    8.3. Determine which channels encountered an AHB error via the ERRn bit.
    8.4. Determine which asynchronous and control Rx channel pages contain a packet start via the PSn bit (extract the PML).
9. Reprogram the expired or broken AHB page(s) via steps 3 and 4 in Section "Program the AHB Block DMAs".
10. Repeat steps 6–9 for all channels with pending interrupts.
11. Repeat steps 4–10 while there are active channels.

**Note:** Channels that receive an AHB error response are disabled (CE = 0) by hardware.

**Servicing the MediaLB Interrupts**

1. Select the MediaLB Channel Status Register (MSn) to be cleared by software, writing a '0' to the appropriate bits.
2. Program MLB_MIEN to enable protocol error interrupts for all active MediaLB channels (MLB_MIEN.CTX_PE = 1, MLB_MIEN.CRX_PE = 1, MLB_MIEN.ATX_PE = 1, MLB_MIEN.ARX_PE = 1, MLB_MIEN.SYNC_PE = 1, and MLB_MIEN.ISOC_PE = 1)
3. Wait for an interrupt on the mlb_int signal.
4. Read the MSn registers to determine which channel(s) are causing the interrupt.
5. Read RSTS/WSTS of the appropriate CDT(s) to determine the interrupt type.
6. Clear RSTS/WSTS errors to resume channel operation.
    6.1. For synchronous channels: WSTS[3] = 0
    6.2. For isochronous channels: WSTS[2:1] = 00
    6.3. For asynchronous and control channels: RSTS[4]/WSTS[4] = 0 and RSTS[2]/ WSTS[2] = 0

**Polling for MediaLB System Commands**

© 2021 Microchip Technology Inc. and its subsidiaries

The MLB supports the MediaLB System Commands (e.g. MlbScan, MlbReset, MOST_Unlock). The MediaLB System Status (MLB_MSS) Register is used to detect a System Command received from the MediaLB Controller. The MLB automatically sends the appropriate system response to the MediaLB Controller.

The procedure for the application is:

1. The application periodically polls the MLB_MSS register.
2. Clear by writing a '0' to the appropriate bit in MLB_MSS register after the application finishes the service.
3. If MLB_MSS.SWSYSCMD = 1, read the MLB_MSD register to receive the system data sent from MediaLB Controller.

#### 48.6.4.3 Low Power Mode

MLB does not provide dedicated low power mode features.

In case the clocks of digital IP need to shut down to save power, the following operations are recommended before entering low power mode:

- Finish any active MLB transfer
- Disable MLB (clear the MLBEN and MLBPEN bits in MLB_MLBC0)
- Disable HBI (clear all bits in MLB_HCMR0 and MLB_HCMR1, clear EN bit in MLB_HCTL)
- Mask AHB interrupts (clear all bits in MLB_ACMR0 and MLB_ACMR1)

For information on configuring the MLB IP if the clocks are re-enabled, see Section "Configure the Hardware".

© 2021 Microchip Technology Inc.
and its subsidiaries

## 48.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | MLB_MLBC0 | 7:0 | MLBLK | | ZERO | MLBCLK[2:0] | | | | MLBEN |
| | | 15:8 | FCNT[0] | CTLRETRY | | ASYRETRY | | | | |
| | | 23:16 | | | | | | | FCNT[2:1] | |
| | | 31:24 | | | | | | | | |
| 0x04 ... 0x0B | Reserved | | | | | | | | | |
| 0x0C | MLB_MS0 | 7:0 | MCS: MediaLB Channel Status [31[7:0] | | | | | | | |
| | | 15:8 | MCS: MediaLB Channel Status [31[15:8] | | | | | | | |
| | | 23:16 | MCS: MediaLB Channel Status [31[23:16] | | | | | | | |
| | | 31:24 | MCS: MediaLB Channel Status [31[31:24] | | | | | | | |
| 0x10 ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | MLB_MS1 | 7:0 | MCS: MediaLB Channel Status [63[7:0] | | | | | | | |
| | | 15:8 | MCS: MediaLB Channel Status [63[15:8] | | | | | | | |
| | | 23:16 | MCS: MediaLB Channel Status [63[23:16] | | | | | | | |
| | | 31:24 | MCS: MediaLB Channel Status [63[31:24] | | | | | | | |
| 0x18 ... 0x1F | Reserved | | | | | | | | | |
| 0x20 | MLB_MSS | 7:0 | | | SERVREQ | SWSYSCMD | CSSYSCMD | ULKSYSCMD | LKSYSCMD | RSTSYSCMD |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | MLB_MSD | 7:0 | SD0[7:0] | | | | | | | |
| | | 15:8 | SD1[7:0] | | | | | | | |
| | | 23:16 | SD2[7:0] | | | | | | | |
| | | 31:24 | SD3[7:0] | | | | | | | |
| 0x28 ... 0x2B | Reserved | | | | | | | | | |
| 0x2C | MLB_MIEN | 7:0 | | | | | | | ISOC_BUFO | ISOC_PE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | ATX_BREAK | ATX_PE | ATX_DONE | ARX_BREAK | ARX_PE | ARX_DONE | SYNC_PE |
| | | 31:24 | | | CTX_BREAK | CTX_PE | CTX_DONE | CRX_BREAK | CRX_PE | CRX_DONE |
| 0x30 ... 0x3B | Reserved | | | | | | | | | |
| 0x3C | MLB_MLBC1 | 7:0 | CLKM | LOCK | | | | | | |
| | | 15:8 | NDA[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x40 ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | MLB_HCTL | 7:0 | | | | | | | RST1 | RST0 |
| | | 15:8 | EN | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 ... 0x87 | Reserved | | | | | | | | | |
| 0x88 | MLB_HCMR0 | 7:0 | CHM: Bitwise Channel Mask Bit [31[7:0] | | | | | | | |
| | | 15:8 | CHM: Bitwise Channel Mask Bit [31[15:8] | | | | | | | |
| | | 23:16 | CHM: Bitwise Channel Mask Bit [31[23:16] | | | | | | | |
| | | 31:24 | CHM: Bitwise Channel Mask Bit [31[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x8C | MLB_HCMR1 | 7:0 | CHM: Bitwise Channel Mask Bit [63[7:0] | | | | | | | |
| | | 15:8 | CHM: Bitwise Channel Mask Bit [63[15:8] | | | | | | | |
| | | 23:16 | CHM: Bitwise Channel Mask Bit [63[23:16] | | | | | | | |
| | | 31:24 | CHM: Bitwise Channel Mask Bit [63[31:24] | | | | | | | |
| 0x90 | MLB_HCER0 | 7:0 | CERR: Bitwise Channel Error Bit [31[7:0] | | | | | | | |
| | | 15:8 | CERR: Bitwise Channel Error Bit [31[15:8] | | | | | | | |
| | | 23:16 | CERR: Bitwise Channel Error Bit [31[23:16] | | | | | | | |
| | | 31:24 | CERR: Bitwise Channel Error Bit [31[31:24] | | | | | | | |
| 0x94 | MLB_HCER1 | 7:0 | CERR: Bitwise Channel Error Bit [63[7:0] | | | | | | | |
| | | 15:8 | CERR: Bitwise Channel Error Bit [63[15:8] | | | | | | | |
| | | 23:16 | CERR: Bitwise Channel Error Bit [63[23:16] | | | | | | | |
| | | 31:24 | CERR: Bitwise Channel Error Bit [63[31:24] | | | | | | | |
| 0x98 | MLB_HCBR0 | 7:0 | CHB: Bitwise Channel Busy Bit [31[7:0] | | | | | | | |
| | | 15:8 | CHB: Bitwise Channel Busy Bit [31[15:8] | | | | | | | |
| | | 23:16 | CHB: Bitwise Channel Busy Bit [31[23:16] | | | | | | | |
| | | 31:24 | CHB: Bitwise Channel Busy Bit [31[31:24] | | | | | | | |
| 0x9C | MLB_HCBR1 | 7:0 | CHB: Bitwise Channel Busy Bit [63[7:0] | | | | | | | |
| | | 15:8 | CHB: Bitwise Channel Busy Bit [63[15:8] | | | | | | | |
| | | 23:16 | CHB: Bitwise Channel Busy Bit [63[23:16] | | | | | | | |
| | | 31:24 | CHB: Bitwise Channel Busy Bit [63[31:24] | | | | | | | |
| 0xA0 ... 0xBF | Reserved | | | | | | | | | |
| 0xC0 | MLB_MDAT0 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0xC4 | MLB_MDAT1 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0xC8 | MLB_MDAT2 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0xCC | MLB_MDAT3 | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | DATA[23:16] | | | | | | | |
| | | 31:24 | DATA[31:24] | | | | | | | |
| 0xD0 | MLB_MDWE0 | 7:0 | MASK: Bitwise Write Enable for CTR Data - bits[31[7:0] | | | | | | | |
| | | 15:8 | MASK: Bitwise Write Enable for CTR Data - bits[31[15:8] | | | | | | | |
| | | 23:16 | MASK: Bitwise Write Enable for CTR Data - bits[31[23:16] | | | | | | | |
| | | 31:24 | MASK: Bitwise Write Enable for CTR Data - bits[31[31:24] | | | | | | | |
| 0xD4 | MLB_MDWE1 | 7:0 | MASK: Bitwise Write Enable for CTR Data - bits[39:32] | | | | | | | |
| | | 15:8 | MASK: Bitwise Write Enable for CTR Data - bits[47:40] | | | | | | | |
| | | 23:16 | MASK: Bitwise Write Enable for CTR Data - bits[55:48] | | | | | | | |
| | | 31:24 | MASK: Bitwise Write Enable for CTR Data - bits[63:56] | | | | | | | |
| 0xD8 | MLB_MDWE2 | 7:0 | MASK: Bitwise Write Enable for CTR Data - bits[71:64] | | | | | | | |
| | | 15:8 | MASK: Bitwise Write Enable for CTR Data - bits[79:72] | | | | | | | |
| | | 23:16 | MASK: Bitwise Write Enable for CTR Data - bits[87:80] | | | | | | | |
| | | 31:24 | MASK: Bitwise Write Enable for CTR Data - bits[95:88] | | | | | | | |
| 0xDC | MLB_MDWE3 | 7:0 | MASK: Bitwise Write Enable for CTR Data - Bits[103:96] | | | | | | | |
| | | 15:8 | MASK: Bitwise Write Enable for CTR Data - Bits[111:104] | | | | | | | |
| | | 23:16 | MASK: Bitwise Write Enable for CTR Data - Bits[119:112] | | | | | | | |
| | | 31:24 | MASK: Bitwise Write Enable for CTR Data - Bits[127:120] | | | | | | | |
| 0xE0 | MLB_MCTL | 7:0 | | | | | | | | XCMP |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xE4 | MLB_MADR | 7:0 | ADDR[7:0] | | | | | | | |
| | | 15:8 | | | ADDR[13:8] | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | WNR | TB | | | | | | |
| 0xE8 ... 0x03BF | Reserved | | | | | | | | | |
| 0x03C0 | MLB_ACTL | 7:0 | | | | MPB | | DMA_MODE | SMX | SCE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x03C4 ... 0x03CF | Reserved | | | | | | | | | |
| 0x03D0 | MLB_ACSR0 | 7:0 | CHS: Interrupt Status for Logical Channels [31[7:0] | | | | | | | |
| | | 15:8 | CHS: Interrupt Status for Logical Channels [31[15:8] | | | | | | | |
| | | 23:16 | CHS: Interrupt Status for Logical Channels [31[23:16] | | | | | | | |
| | | 31:24 | CHS: Interrupt Status for Logical Channels [31[31:24] | | | | | | | |
| 0x03D4 | MLB_ACSR1 | 7:0 | CHS[7:0] | | | | | | | |
| | | 15:8 | CHS[15:8] | | | | | | | |
| | | 23:16 | CHS[23:16] | | | | | | | |
| | | 31:24 | CHS[31:24] | | | | | | | |
| 0x03D8 | MLB_ACMR0 | 7:0 | CHM[7:0] | | | | | | | |
| | | 15:8 | CHM[15:8] | | | | | | | |
| | | 23:16 | CHM[23:16] | | | | | | | |
| | | 31:24 | CHM[31:24] | | | | | | | |
| 0x03DC | MLB_ACMR1 | 7:0 | CHM[7:0] | | | | | | | |
| | | 15:8 | CHM[15:8] | | | | | | | |
| | | 23:16 | CHM[23:16] | | | | | | | |
| | | 31:24 | CHM[31:24] | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 48.7.1 MediaLB Control 0 Register

**Name:** MLB_MLBC0
**Offset:** 0x000
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | 0 | 0 | | 0 | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | 0 | | 0 | 0 | 0 | 0 | | 0 |

**Bits 17:15 – FCNT[2:0]** The number of frames per sub-buffer for synchronous channels

| Value | Name | Description |
|---|---|---|
| 0 | 1_FRAME | 1 frame per sub-buffer (Operation is the same as Standard mode.) |
| 1 | 2_FRAMES | 2 frames per sub-buffer |
| 2 | 4_FRAMES | 4 frames per sub-buffer |
| 3 | 8_FRAMES | 8 frames per sub-buffer |
| 4 | 16_FRAMES | 16 frames per sub-buffer |
| 5 | 32_FRAMES | 32 frames per sub-buffer |
| 6 | 64_FRAMES | 64 frames per sub-buffer |

**Bit 14 – CTLRETRY** Control Tx Packet Retry

| Value | Description |
|---|---|
| 0 | A control packet that is flagged with a Break or ProtocolError by the receiver is skipped. |
| 1 | A control packet that is flagged with a Break or ProtocolError by the receiver is retransmitted. |

**Bit 12 – ASYRETRY** Asynchronous Tx Packet Retry

| Value | Description |
|---|---|
| 0 | An asynchronous packet that is flagged with a Break or ProtocolError by the receiver is skipped. |
| 1 | An asynchronous packet that is flagged with a Break or ProtocolError by the receiver is retransmitted. |

**Bit 7 – MLBLK** MediaLB Lock Status (read-only)

| Value | Description |
|---|---|
| 1 | indicates that the MediaLB block is synchronized to the incoming MediaLB frame.<br>If MLBLK is cleared (unlocked), MLBLK is set after FRAMESYNC is detected at the same position for three consecutive frames.<br>If MLBLK is set (locked), MLBLK is cleared after not receiving FRAMESYNC at the expected time for two consecutive frames. While MLBLK is set, FRAMESYNC patterns occurring at locations other than the expected one are ignored. |

**Bit 5 – ZERO** Must be Written to 0

**Bits 4:2 – MLBCLK[2:0]** MLBCLK (MediaLB clock) Speed Select

| Value | Name | Description |
|---|---|---|
| 0 | 256_FS | 256xFs (for MLBPEN = 0) |
| 1 | 512_FS | 512xFs (for MLBPEN = 0) |
| 2 | 1024_FS | 1024xFs (for MLBPEN = 0) |
| 3 | 2048_FS | 2048xFs (for MLBPEN = 0) |
| 4 | 3072_FS | 3072xFs (for MLBPEN = 0) |
| 5 | 4096_FS | 4096xFs (for MLBPEN = 0) |
| 6 | 6144_FS | 6144xFs (for MLBPEN = 0) |

**Bit 0 – MLBEN** MediaLB Enable

| Value | Description |
|---|---|
| 1 | MLBCLK (MediaLB clock), MLBSIG (signal), and MLBDATA (data) are received and transmitted on the appropriate MediaLB pins. |

### 48.7.2 MediaLB Channel Status 0 Register

**Name:** MLB_MS0
**Offset:** 0x00C
**Reset:** 0x00000000
**Property:** Read/Write

Each bit can be cleared by writing a 0.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MCS: MediaLB Channel Status [31[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MCS: MediaLB Channel Status [31[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MCS: MediaLB Channel Status [31[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MCS: MediaLB Channel Status [31[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MCS: MediaLB Channel Status [31[31:0]** 0] (cleared by writing a 0)
Indicates the channel status for MediaLB channels 31 to 0. Channel status bits are set by hardware and cleared by software. Status is only set if the appropriate bits in the MLB_MIEN register are set.

### 48.7.3 MediaLB Channel Status1 Register

**Name:** MLB_MS1
**Offset:** 0x014
**Reset:** 0x00000000
**Property:** Read/Write

Each bit can be cleared by writing a 0.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MCS: MediaLB Channel Status [63[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MCS: MediaLB Channel Status [63[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MCS: MediaLB Channel Status [63[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MCS: MediaLB Channel Status [63[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MCS: MediaLB Channel Status [63[31:0]** 32] (cleared by writing a 0)
Indicates the channel status for MediaLB channels 63 to 32. Channel status bits are set by hardware and cleared by software. Status is only set if the appropriate bits in the MLB_MIEN register are set.

### 48.7.4 MediaLB System Status Register

**Name:** MLB_MSS
**Offset:** 0x020
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | SERVREQ | SWSYSCMD | CSSYSCMD | ULKSYSCMD | LKSYSCMD | RSTSYSCMD |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 5 – SERVREQ** Service Request Enabled

| Value | Description |
|---|---|
| 0 | The MediaLB block responds with a “device present” system response. |
| 1 | The MediaLB block responds with a “device present, request service” system response if a matching channel scan system command is detected. |

**Bit 4 – SWSYSCMD** Software System Command Detected in the System Quadlet (cleared by writing a 0)
Set by hardware, cleared by software. Data is stored in the MLB_MSD register for this command.

**Bit 3 – CSSYSCMD** Channel Scan System Command Detected in the System Quadlet (cleared by writing a 0)
Set by hardware, cleared by software. If the node address specified in Data quadlet matches the value in MLB_MLBC1.NDA, the device responds either “device present” or “device present, request service” system response in the next system quadlet.

**Bit 2 – ULKSYSCMD** Network Unlock System Command Detected in the System Quadlet (cleared by writing a 0)
Set by hardware, cleared by software.

**Bit 1 – LKSYSCMD** Network Lock System Command Detected in the System Quadlet (cleared by writing a 0)
Set by hardware, cleared by software.

**Bit 0 – RSTSYSCMD** Reset System Command Detected in the System Quadlet (cleared by writing a 0)
Set by hardware, cleared by software.

### 48.7.5 MediaLB System Data Register

**Name:** MLB_MSD
**Offset:** 0x024
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | SD3[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SD2[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SD1[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SD0[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:24 – SD3[7:0]** System Data (Byte 3)
Updated with MediaLB Data[31:24] when a MediaLB software system command is received in the system quadlet. If MLB_MSS.SWSYSCMD is already set, then SD3 is not updated.

**Bits 23:16 – SD2[7:0]** System Data (Byte 2)
Updated with MediaLB Data[23:16] when a MediaLB software system command is received in the system quadlet. If MLB_MSS.SWSYSCMD is already set, then SD2 is not updated.

**Bits 15:8 – SD1[7:0]** System Data (Byte 1)
Updated with MediaLB Data[15:8] when a MediaLB software system command is received in the system quadlet. If MLB_MSS.SWSYSCMD is already set, then SD1 is not updated.

**Bits 7:0 – SD0[7:0]** System Data (Byte 0)
Updated with MediaLB Data[7:0] when a MediaLB software system command is received in the system quadlet. If MLB_MSS.SWSYSCMD is already set, then SD0 is not updated.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 48.7.6 MediaLB Interrupt Enable Register

**Name:** MLB_MIEN
**Offset:** 0x02C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | CTX_BREAK | CTX_PE | CTX_DONE | CRX_BREAK | CRX_PE | CRX_DONE |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | ATX_BREAK | ATX_PE | ATX_DONE | ARX_BREAK | ARX_PE | ARX_DONE | SYNC_PE |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ISOC_BUFO | ISOC_PE |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

**Bit 29 – CTX_BREAK** Control Tx Break Enable

| Value | Description |
|---|---|
| 1 | A ReceiverBreak response received from the receiver on a control Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 28 – CTX_PE** Control Tx Protocol Error Enable

| Value | Description |
|---|---|
| 1 | A ProtocolError generated by the receiver on a control Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 27 – CTX_DONE** Control Tx Packet Done Enable

| Value | Description |
|---|---|
| 1 | A packet transmitted with no errors on a control Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 26 – CRX_BREAK** Control Rx Break Enable
Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.

| Value | Description |
|---|---|
| 1 | A ControlBreak command received from the transmitter on a control. |

**Bit 25 – CRX_PE** Control Rx Protocol Error Enable

| Value | Description |
|---|---|
| 1 | A ProtocolError detected on a control Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 24 – CRX_DONE** Control Rx Packet Done Enable

| Value | Description |
|---|---|
| 1 | A packet received with no errors on a control Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 22 – ATX_BREAK** Asynchronous Tx Break Enable

| Value | Description |
|---|---|
| 1 | A ReceiverBreak response received from the receiver on an asynchronous Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 21 – ATX_PE** Asynchronous Tx Protocol Error Enable

| Value | Description |
|---|---|
| 1 | A ProtocolError generated by the receiver on an asynchronous Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 20 – ATX_DONE** Asynchronous Tx Packet Done Enable
Tx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set.

| Value | Description |
|---|---|
| 1 | A packet transmitted with no errors on an asynchronous |

**Bit 19 – ARX_BREAK** Asynchronous Rx Break Enable

| Value | Description |
|---|---|
| 1 | A AsyncBreak command received from the transmitter on an asynchronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 18 – ARX_PE** Asynchronous Rx Protocol Error Enable

| Value | Description |
|---|---|
| 1 | A ProtocolError detected on an asynchronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 17 – ARX_DONE** Asynchronous Rx Done Enable

| Value | Description |
|---|---|
| 1 | A packet received with no errors on an asynchronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 16 – SYNC_PE** Synchronous Protocol Error Enable

| Value | Description |
|---|---|
| 1 | A ProtocolError detected on a synchronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

**Bit 1 – ISOC_BUFO** Isochronous Rx Buffer Overflow Enable

| Value | Description |
|---|---|
| 1 | A buffer overflow on an isochronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. This occurs only when isochronous flow control is disabled. |

**Bit 0 – ISOC_PE** Isochronous Rx Protocol Error Enable

| Value | Description |
|---|---|
| 1 | A ProtocolError detected on an isochronous Rx channel causes the appropriate channel bit in the MLB_MS0 or MLB_MS1 registers to be set. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 48.7.7 MediaLB Control 1 Register

**Name:** MLB_MLBC1
**Offset:** 0x03C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NDA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLKM | LOCK | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | | | | | | |

**Bits 15:8 – NDA[7:0]** Node Device Address
Used for system commands directed to individual MediaLB nodes.

**Bit 7 – CLKM** MediaLB Clock Missing Status (cleared by writing a 0)
Set when MLBCLK (MediaLB clock) is not toggling at the pin; cleared by software.

**Bit 6 – LOCK** MediaLB Lock Error Status (cleared by writing a 0)
Set when MediaLB is unlocked; cleared by software.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 48.7.8 HBI Control Register

**Name:** MLB_HCTL
**Offset:** 0x080
**Reset:** 0x00000000
**Property:** Read/Write

The HC can control and monitor general operation of the HBI block by reading and writing the HBI Control Register (MLB_HCTL) through the I/O interface. Each bit of MLB_HCTL is read/write.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EN | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RST1 | RST0 |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

**Bit 15 – EN** HBI Enable

| Value | Description |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

**Bit 1 – RST1** Address Generation Unit 1 Software Reset

| Value | Description |
|---|---|
| 0 | Active |
| 1 | Reset |

**Bit 0 – RST0** Address Generation Unit 0 Software Reset

| Value | Description |
|---|---|
| 0 | Active |
| 1 | Reset |

### 48.7.9 HBI Channel Mask 0 Register

**Name:** MLB_HCMR0
**Offset:** 0x088
**Reset:** 0x00000000
**Property:** Read/Write

The HC can control which channel(s) are able to generate an HBI interrupt by writing the HBI Channel Mask Registers (HCMRn). Each bit of HCMRn is read/write.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHM: Bitwise Channel Mask Bit [31[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHM: Bitwise Channel Mask Bit [31[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHM: Bitwise Channel Mask Bit [31[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHM: Bitwise Channel Mask Bit [31[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CHM: Bitwise Channel Mask Bit [31[31:0]** 0]
CHM[n] = 1 indicates that channel n can generate an interrupt.

### 48.7.10 HBI Channel Mask 1 Register

**Name:** MLB_HCMR1
**Offset:** 0x08C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHM: Bitwise Channel Mask Bit [63[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHM: Bitwise Channel Mask Bit [63[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHM: Bitwise Channel Mask Bit [63[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHM: Bitwise Channel Mask Bit [63[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CHM: Bitwise Channel Mask Bit [63[31:0]** 32]
CHM[n] = 1 indicates that channel n can generate an interrupt.

### 48.7.11 HBI Channel Error 0 Register

**Name:** MLB_HCER0
**Offset:** 0x090
**Reset:** 0x00000000
**Property:** Read-only

The HBI Channel Error Registers (HCERn) indicate which channel(s) have encountered fatal errors.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CERR: Bitwise Channel Error Bit [31[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CERR: Bitwise Channel Error Bit [31[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CERR: Bitwise Channel Error Bit [31[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CERR: Bitwise Channel Error Bit [31[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CERR: Bitwise Channel Error Bit [31[31:0]** 0]
CERR[n] = 1 indicates that a fatal error occurred on channel n.

### 48.7.12 HBI Channel Error 1 Register

**Name:** MLB_HCER1
**Offset:** 0x094
**Reset:** 0x00000000
**Property:** Read-only

HCERn status bits are set when hardware detects hardware errors on the given logical channel, including:

- Channel opened, but not enabled,
- Channel programmed with invalid channel type, or
- Out-of-range PML for asynchronous or control Tx channels

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CERR: Bitwise Channel Error Bit [63[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CERR: Bitwise Channel Error Bit [63[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CERR: Bitwise Channel Error Bit [63[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CERR: Bitwise Channel Error Bit [63[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CERR: Bitwise Channel Error Bit [63[31:0]** 32]
CERR[n] = 1 indicates that a fatal error occurred on channel n.

### 48.7.13 HBI Channel Busy 0 Register

**Name:** MLB_HCBR0
**Offset:** 0x098
**Reset:** 0x00000000
**Property:** Read-only

The HC can determine which channel(s) are busy by reading the HBI Channel Busy Registers (HCBRn). An HBI channel is busy if:

• it is currently loaded into one of the two AGUs

• the channel is enabled, CE = 1 from the Channel Allocation Table (CTR Address Mapping), and

• the DMA is active

When an HBI channel is busy, hardware may write back its local copy of the channel descriptor at any time. System software should not write a CDT descriptor for a channel that is busy. Only two HBI channels can be busy at any given time. Each bit of HCBRn is read-only.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHB: Bitwise Channel Busy Bit [31[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHB: Bitwise Channel Busy Bit [31[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHB: Bitwise Channel Busy Bit [31[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHB: Bitwise Channel Busy Bit [31[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CHB: Bitwise Channel Busy Bit [31[31:0]** 0]
CHB[n] = 1 indicates that channel n is busy.

### 48.7.14 HBI Channel Busy 1 Register

**Name:** MLB_HCBR1
**Offset:** 0x09C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHB: Bitwise Channel Busy Bit [63[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHB: Bitwise Channel Busy Bit [63[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHB: Bitwise Channel Busy Bit [63[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHB: Bitwise Channel Busy Bit [63[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CHB: Bitwise Channel Busy Bit [63[31:0]** 32]
CHB[n] = 1 indicates that channel n is busy.

### 48.7.15 MIF Data 0 Register

**Name:** MLB_MDAT0
**Offset:** 0x0C0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** CRT or DBR Data
CTR data - bits[31:0] of 128-bit entry or
DBR data - bits[7:0] of 8-bit entry

### 48.7.16 MIF Data 1 Register

**Name:** MLB_MDAT1
**Offset:** 0x0C4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** CRT Data
CTR data - bits[63:32] of 128-bit entry

### 48.7.17 MIF Data 2 Register

**Name:** MLB_MDAT2
**Offset:** 0x0C8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** CRT Data
CTR data - bits[95:64] of 128-bit entry

### 48.7.18 MIF Data 3 Register

**Name:** MLB_MDAT3
**Offset:** 0x0CC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – DATA[31:0]** CRT Data
CTR data - bits[127:96] of 128-bit entry

### 48.7.19 MIF Data Write Enable 0 Register

**Name:** MLB_MDWE0
**Offset:** 0x0D0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[31[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[31[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[31[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[31[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MASK: Bitwise Write Enable for CTR Data - bits[31[31:0]** 0]
MASK[n] = 1 indicates that CTR data [n] is enabled.

### 48.7.20 MIF Data Write Enable 1 Register

**Name:** MLB_MDWE1
**Offset:** 0x0D4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[63:56] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[55:48] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[47:40] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[39:32] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MASK: Bitwise Write Enable for CTR Data - bits[63:32]**
MASK[n] = 1 indicates that CTR data [n] is enabled.

### 48.7.21 MIF Data Write Enable 2 Register

**Name:** MLB_MDWE2
**Offset:** 0x0D8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[95:88] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[87:80] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[79:72] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - bits[71:64] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MASK: Bitwise Write Enable for CTR Data - bits[95:64]**
MASK[n] = 1 indicates that CTR data [n] is enabled.

### 48.7.22 MIF Data Write Enable 3 Register

**Name:** MLB_MDWE3
**Offset:** 0x0DC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - Bits[127:120] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - Bits[119:112] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

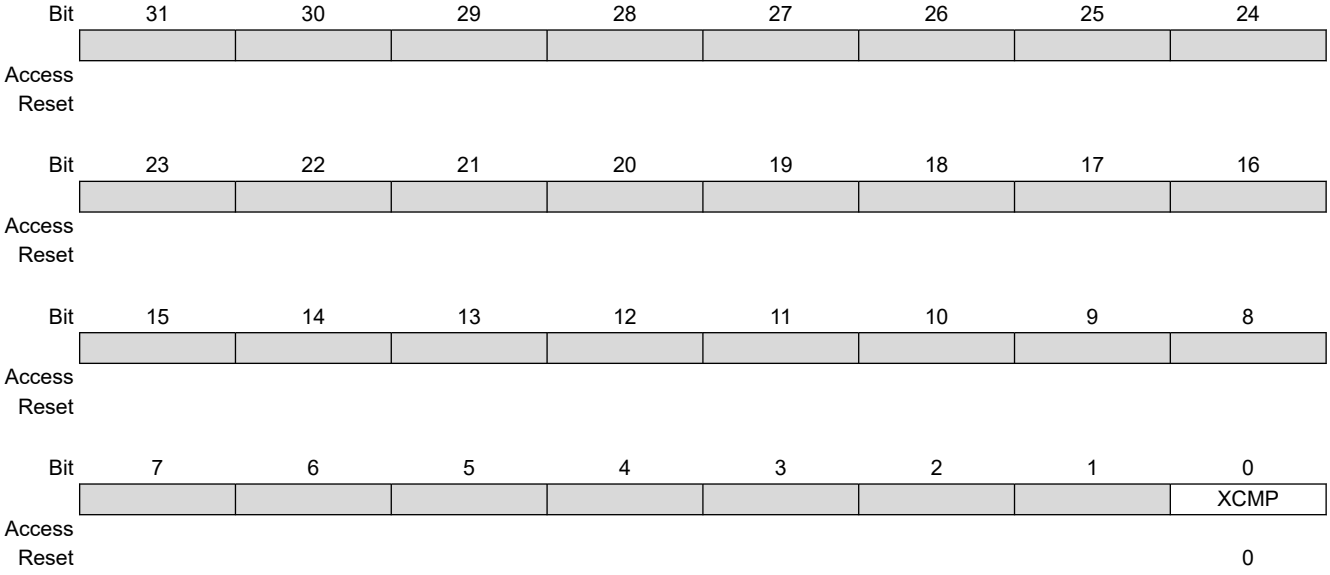
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - Bits[111:104] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MASK: Bitwise Write Enable for CTR Data - Bits[103:96] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – MASK: Bitwise Write Enable for CTR Data - Bits[127:96]**
MASK[n] = 1 indicates that CTR data [n] is enabled.

### 48.7.23 MIF Control Register

**Name:** MLB_MCTL
**Offset:** 0x0E0
**Reset:** 0x00000000
**Property:** Read/Write




**Bit 0 – XCMP** Transfer Complete (Write 0 to Clear)

### 48.7.24 MIF Address Register

**Name:** MLB_MADR
**Offset:** 0x0E4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WNR | TB | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | ADDR[13:8] | | | | | |
| Access | | | | | | | | |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ADDR[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – WNR** Write-Not-Read Selection

| Value | Description |
|---|---|
| 0 | Read |
| 1 | Write |

**Bit 30 – TB** Target Location Bit
0 (CTR): Selects CTR
1 (DBR): Selects DBR

**Bits 13:0 – ADDR[13:0]** CTR or DBR Address
CTR address of 128-bit entry or
DBR address of 8-bit entry - bits[7:0]

### 48.7.25 AHB Control Register

**Name:** MLB_ACTL
**Offset:** 0x3C0
**Reset:** 0x00000000
**Property:** Read/Write

The AHB Control (MLB_ACTL) register is written by the HC to configure the AHB block for channel interrupts. MLB_ACTL contains three configuration fields, one is used to select the DMA mode, one is used to multiplex channel interrupts onto a single interrupt signal, and the last selects the method of clearing channel interrupts (either software or hardware).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | MPB | | DMA_MODE | SMX | SCE |
| Access | | | | | | | | |
| Reset | | | | 0 | | 0 | 0 | 0 |

**Bit 4 – MPB** DMA Packet Buffering Mode
0 (SINGLE_PACKET): Single-packet mode
1 (MULTIPLE_PACKET): Multiple-packet mode

**Bit 2 – DMA_MODE** DMA Mode

| Value | Description |
|---|---|
| 0 | DMA Mode 0 |
| 1 | DMA Mode 1 |

**Bit 1 – SMX** AHB Interrupt Mux Enable

| Value | Description |
|---|---|
| 0 | MLB_ACSR0 generates an interrupt on MediaLB IRQ0; MLB_ACSR1 generates an interrupt on MediaLB IRQ1 |
| 1 | MLB_ACSR0 and MLB_ACSR1 generate an interrupts on MediaLB IRQ0 only |

**Bit 0 – SCE** Software Clear Enable

| Value | Description |
|---|---|
| 0 | Hardware clears interrupt after a MLB_ACSRn register read |
| 1 | Software writes a ‘1’ to clear |

### 48.7.26 AHB Channel Status 0 Register

**Name:** MLB_ACSR0
**Offset:** 0x3D0
**Reset:** 0x00000000
**Property:** Read/Write

The AHB Channel Status (ACSRn) registers contain interrupt bits for each of the 64 physical channels. When an MLB_ACSRn register bit is set, it indicates that the corresponding physical channel has an interrupt pending.

An AHB interrupt is triggered when either DNEn or ERRn is set within the AHB Channel Descriptor. The HC is notified of the channel interrupt via ahb_int[1:0]. When an interrupt occurs in MLB_ACSR0 (for channels 31 to 0) MediaLB IRQ0 is set. When an interrupt occurs in MLB_ACSR1 (for channels 63 to 32) MediaLB IRQ1 is set.

Interrupts in MLB_ACSR0 and MLB_ACSR1 can be optionally multiplexed onto a single interrupt signal, MediaLB IRQ0, if MLB_ACTL.SMX = 1. If MLB_ACTL.SCE = 0, hardware automatically clears the interrupt bit(s) after the HC reads the ACSRn register. Alternatively, if MLB_ACTL.SCE = 1, software must write a 1 to the appropriate bit(s) of MLB_ACSRn to clear the interrupt(s).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHS: Interrupt Status for Logical Channels [31[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHS: Interrupt Status for Logical Channels [31[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHS: Interrupt Status for Logical Channels [31[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHS: Interrupt Status for Logical Channels [31[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CHS: Interrupt Status for Logical Channels [31[31:0]** 0] (cleared by writing a 1)
CHS[n] = 1 indicates that an interrupt is pending on channel n.

### 48.7.27 AHB Channel Status 1 Register

**Name:** MLB_ACSR1
**Offset:** 0x3D4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHS[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHS[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHS[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHS[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CHS[31:0]** Interrupt Status for Logical Channels 63 to 32 (cleared by writing a 1)
CHS[n] = 1 indicates that an interrupt is pending on channel n.

### 48.7.28 AHB Channel Mask 0 Register

**Name:** MLB_ACMR0
**Offset:** 0x3D8
**Reset:** 0x00000000
**Property:** Read/Write

Using the AHB Channel Mask (ACMRn) register, the HC can control which channel(s) generate interrupts on ahb_int[1:0]. All ACMRn register bits default as '0' ("masked"); therefore, the HC must initially write ACMRn to enable interrupts. Each bit of ACMRn is read/write accessible.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHM[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHM[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHM[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHM[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CHM[31:0]** Bitwise Channel Mask Bits 31 to 0
CHM[n] = 1 indicates that channel n can generate an interrupt.

### 48.7.29 AHB Channel Mask 1 Register

**Name:** MLB_ACMR1
**Offset:** 0x3DC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CHM[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CHM[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CHM[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CHM[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CHM[31:0]** Bitwise Channel Mask Bits 63 to 32
CHM[n] = 1 indicates that channel n can generate an interrupt.

# 49. Controller Area Network (MCAN)

## 49.1 Description

The Controller Area Network (MCAN) performs communication according to ISO 11898-1:2015 and to Bosch CAN-FD specification. Additional transceiver hardware is required for connection to the physical layer.

All functions concerning the handling of messages are implemented by the Rx Handler and the Tx Handler. The Rx Handler manages message acceptance filtering, the transfer of received messages from the CAN core to the Message RAM, as well as providing receive message status information. The Tx Handler is responsible for the transfer of transmit messages from the Message RAM to the CAN core, as well as providing transmit status information.

Acceptance filtering is implemented by a combination of up to 128 filter elements, where each element can be configured as a range, as a bit mask, or as a dedicated ID filter.

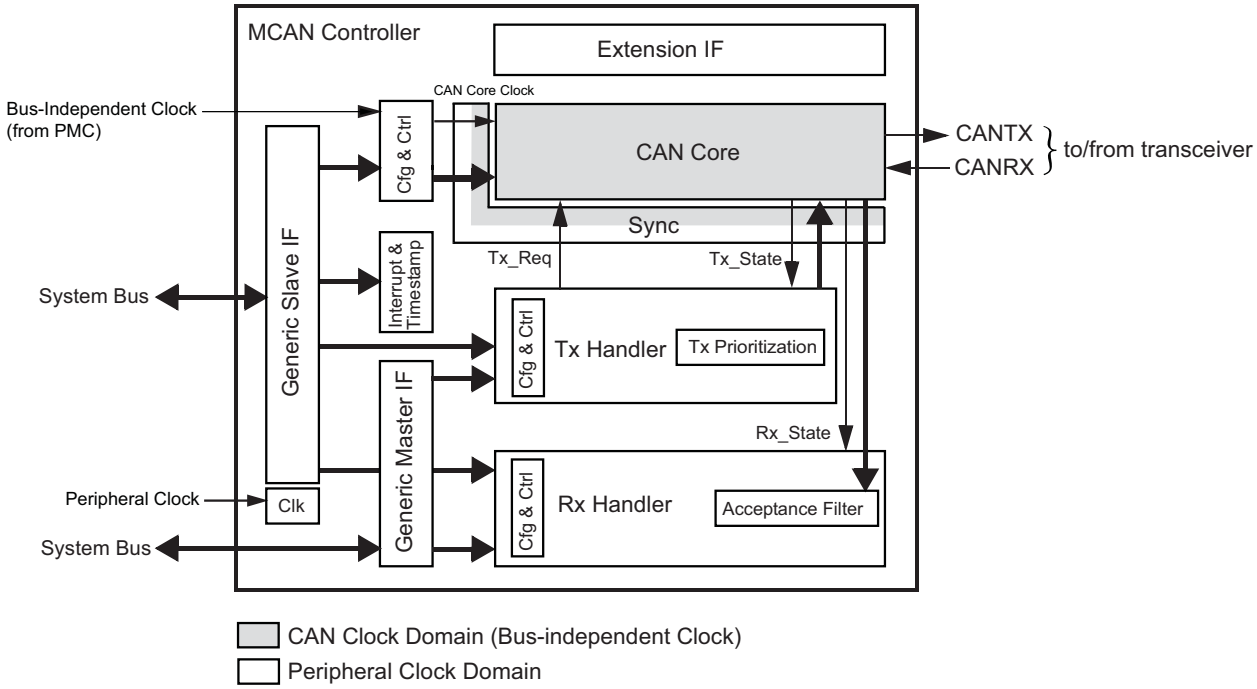
## 49.2 Embedded Characteristics

- Compliant with CAN Protocol Version 2.0 Part A, B and ISO 11898-1
- CAN-FD with up to 64 Data Bytes Supported
- CAN Error Logging
- AUTOSAR Optimized
- SAE J1939 Optimized
- Improved Acceptance Filtering
- Two Configurable Receive FIFOs
- Separate Signalling on Reception of High Priority Messages
- Up to 64 Dedicated Receive Buffers
- Up to 32 Dedicated Transmit Buffers
- Configurable Transmit FIFO
- Configurable Transmit Queue
- Configurable Transmit Event FIFO
- Direct Message RAM Access for Processor
- Multiple MCANs May Share the Same Message RAM
- Programmable Loop-back Test Mode
- Maskable Module Interrupts
- Support for Asynchronous CAN and System Bus Clocks
- Power-down Support
- Debug on CAN Support

© 2021 Microchip Technology Inc.
and its subsidiaries

## 49.3 Block Diagram

**Figure 49-1. MCAN Block Diagram**




**Note:** Refer to section “Power Management Controller (PMC)” for details about the bus-independent clock (PCK5).

**Related Links**

31. Power Management Controller (PMC)

## 49.4 Product Dependencies

### 49.4.1 I/O Lines

The pins used to interface to the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the CAN pins to their peripheral functions.

### 49.4.2 Power Management

The MCAN can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the MCAN clock.

In order to achieve a stable function of the MCAN, the system bus clock must always be faster than or equal to the CAN clock.

It is recommended to use the CAN clock at frequencies of 20, 40 or 80 MHz. To achieve these frequencies, PMC PCK5 must select the UPLLCK (480 MHz) as source clock and divide by 24,12, or 6. PCK5 allows the system bus and processor clock to be modified without affecting the bit rate communication.

### 49.4.3 Interrupt Sources

The two MCAN interrupt lines (MCAN_INT0, MCAN_INT1) are connected on internal sources of the Interrupt Controller.

Using the MCAN interrupts requires the Interrupt Controller to be programmed first.

Interrupt sources can be routed either to MCAN_INT0 or to MCAN_INT1. By default, all interrupt sources are routed to interrupt line MCAN_INT0/1. By programming MCAN_ILE.EINT0 and MCAN_ILE.EINT1, the interrupt sources can be enabled or disabled separately.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.4.4 Address Configuration

The LSBs [bits 15:2] for each section of the CAN Message RAM are configured in the respective buffer configuration registers as detailed in Message RAM.

The MSBs [bits 31:16] of the CAN Message RAM for CAN0 and CAN1 are configured in CCFG_CAN0 and CCFG_SYSIO registers.

### 49.4.5 Timestamping

Timestamping uses the value of CV in the TC Counter Value 0 register (TC_CV0) at address 0x4000C010. TC0 can use the programmable clocks PCK6 or PCK7 as input. Refer to the section “Timer Counter (TC)” for more details.

The selection between PCK6 and PCK7 is done in the Matrix Peripheral Clock Configuration Register (CCFG_PCCR), using the bit TC0CC. Refer to this register in the section “Bus Matrix (MATRIX)” for more details.

These clocks can be programmed in the the registers PMC Programmable Clock Registers PMC_PCK6 and PMC_PCK7, respectively. Refer to these registers in the section “Power Management Controller (PMC)” for more details.

**Related Links**

50. Timer Counter (TC)

31. Power Management Controller (PMC)

## 49.5 Functional Description

### 49.5.1 Operating Modes

#### 49.5.1.1 Software Initialization

Software initialization is started by setting bit MCAN_CCCR.INIT, either by software or by a hardware reset, when an uncorrected bit error was detected in the Message RAM, or by going Bus_Off. While MCAN_CCCR.INIT is set, message transfer from and to the CAN bus is stopped and the status of the CAN bus output CANTX is recessive (HIGH). The counters of the Error Management Logic EML are unchanged. Setting MCAN_CCCR.INIT does not change any configuration register. Resetting MCAN_CCCR.INIT finishes the software initialization. Afterwards the Bit Stream Processor BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (≡ Bus_Idle) before it can take part in bus activities and start the message transfer.

Access to the MCAN configuration registers is only enabled when both bits MCAN_CCCR.INIT and MCAN_CCCR.CCE are set (protected write).

MCAN_CCCR.CCE can only be configured when MCAN_CCCR.INIT = ‘1’. MCAN_CCCR.CCE is automatically cleared when MCAN_CCCR.INIT = ‘0’.

The following registers are cleared when MCAN_CCCR.CCE = ‘1’:

- High Priority Message Status (MCAN_HPMS)
- Receive FIFO 0 Status (MCAN_RXF0S)
- Receive FIFO 1 Status (MCAN_RXF1S)
- Transmit FIFO/Queue Status (MCAN_TXFQS)
- Transmit Buffer Request Pending (MCAN_TXBRP)
- Transmit Buffer Transmission Occurred (MCAN_TXBTO)
- Transmit Buffer Cancellation Finished (MCAN_TXBCF)
- Transmit Event FIFO Status (MCAN_TXEFS)

The Timeout Counter value MCAN_TOCV.TOC is loaded with the value configured by MCAN_TOCC.TOP when MCAN_CCCR.CCE = ‘1’.

In addition, the state machines of the Tx Handler and Rx Handler are held in idle state while MCAN_CCCR.CCE = ‘1’.

The following registers are only writeable while MCAN_CCCR.CCE = ‘0’

© 2021 Microchip Technology Inc.
and its subsidiaries

- Transmit Buffer Add Request (MCAN_TXBAR)
- Transmit Buffer Cancellation Request (MCAN_TXBCR)

MCAN_CCCR.TEST and MCAN_CCCR.MON can only be set when MCAN_CCCR.INIT = '1' and MCAN_CCCR.CCE = '1'. Both bits may be cleared at any time. MCAN_CCCR.DAR can only be configured when MCAN_CCCR.INIT = '1' and MCAN_CCCR.CCE = '1'.

#### 49.5.1.2 Normal Operation

Once the MCAN is initialized and MCAN_CCCR.INIT is cleared, the MCAN synchronizes itself to the CAN bus and is ready for communication.

After passing the acceptance filtering, received messages including Message ID and DLC are stored into a dedicated Rx Buffer or into Rx FIFO 0 or Rx FIFO 1.

For messages to be transmitted, dedicated Tx Buffers and/or a Tx FIFO or a Tx Queue can be initialized or updated. Automated transmission on reception of remote frames is not implemented.

#### 49.5.1.3 CAN FD Operation

There are two variants in the CAN FD frame format, first the CAN FD frame without bit rate switching where the data field of a CAN frame may be longer than 8 bytes. The second variant is the CAN FD frame where control field, data field, and CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and the end of the frame.

The previously reserved bit in CAN frames with 11-bit identifiers and the first previously reserved bit in CAN frames with 29-bit identifiers will now be decoded as FDF bit. FDF = recessive signifies a CAN FD frame, FDF = dominant signifies a Classic CAN frame. In a CAN FD frame, the two bits following FDF, res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. The coding of res = recessive is reserved for future expansion of the protocol. In case the MCAN receives a frame with FDF = recessive and res = recessive, it will signal a Protocol Exception Event by setting bit MCAN_PSR.PXE. When Protocol Exception Handling is enabled (MCAN_CCCR.PXHD = 0), this causes the operation state to change from Receiver (MCAN_PSR.ACT = 2) to Integrating (MCAN_PSR.ACT = 00) at the next sample point. In case Protocol Exception Handling is disabled (MCAN_CCCR.PXHD = 1), the MCAN will treat a recessive res bit as an form error and will respond with an error frame.

CAN FD operation is enabled by programming CCCR.FDOE. In case CCCR.FDOE = '1', transmission and reception of CAN FD frames is enabled. Transmission and reception of Classic CAN frames is always possible. Whether a CAN FD frame or a Classic CAN frame is transmitted can be configured via bit FDF in the respective Tx Buffer element. With CCCR.FDOE = '0', received frames are interpreted as Classic CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if bit FDF of a Tx Buffer element is set. CCCR.FDOE and CCCR.BRSE can only be changed while CCCR.INIT and CCCR.CCE are both set.

With MCAN_CCCR.FDOE = 0, the setting of bits FDF and BRS is ignored and frames are transmitted in Classic CAN format. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 0, only bit FDF of a Tx Buffer element is evaluated. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 1, transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significant higher than in the CAN FD arbitration phase. In this case disable the CAN FD bit rate switching option for transmissions.
- During system startup all nodes are transmitting according to ISO11898-1 until it is verified that they are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation.
- Wake-up messages in CAN Partial Networking have to be transmitted in Classic CAN format.
- End-of-line programming in case not all nodes are CAN FD-capable. Non-CAN FD nodes are held in Silent mode until programming has completed. Then all nodes revert to Classic CAN communication.

In the CAN FD format, the coding of the DLC differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN, the codes 9 to 15, which in standard CAN all code a data field of 8 bytes, are coded according to the table below.

**Table 49-1. Coding of DLC in CAN FD**

| DLC | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| Number of Data Bytes | 12 | 16 | 20 | 24 | 32 | 48 | 64 |

In CAN FD frames, the bit timing will be switched inside the frame, after the BRS (Bit Rate Switch) bit, if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by the Nominal Bit Timing and Prescaler register (MCAN_NBTP). In the following CAN FD data phase, the data phase CAN bit timing is used as defined by the Data Bit Timing and Prescaler register (MCAN_DBTP). The bit timing reverts back from the data phase timing at the CRC delimiter or when an error is detected, whichever occurs first.

The maximum configurable bit rate in the CAN FD data phase depends on the CAN core clock frequency. Example: with a CAN clock frequency of 20 MHz and the shortest configurable bit time of 4 $t_q$, the bit rate in the data phase is 5 Mbit/s.

In both data frame formats, CAN FD and CAN FD with bit rate switching, the value of the bit ESI (Error Status Indicator) is determined by the transmitter's error state at the start of the transmission. If the transmitter is error passive, ESI is transmitted recessive, else it is transmitted dominant.

### 49.5.1.4 Transmitter Delay Compensation

During the data phase of a CAN FD transmission only one node is transmitting, all others are receivers. The length of the bus line has no impact. When transmitting via pin CANTX the protocol controller receives the transmitted data from its local CAN transceiver via pin CANRX. The received data is delayed by the transmitter delay. In case this delay is greater than TSEG1 (time segment before sample point), a bit error is detected. In order to enable a data phase bit time that is even shorter than the transmitter delay, the delay compensation is introduced. Without delay compensation, the bit rate in the data phase of a CAN FD frame is limited by the delay.

#### 49.5.1.4.1 Description

The MCAN protocol unit has implemented a delay compensation mechanism to compensate the delay, thereby enabling transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver.

To check for bit errors during the data phase, the delayed transmit data is compared against the received data at the secondary sample point. If a bit error is detected, the transmitter will react to this bit error at the next following regular sample point. During arbitration phase the delay compensation is always disabled.

The transmitter delay compensation enables configurations where the data bit time is shorter than the transmitter delay, it is described in detail in the new ISO11898-1. It is enabled by setting bit MCAN_DBTP.TDC.

The received bit is compared against the transmitted bit at the SSP. The SSP position is defined as the sum of the measured delay from the MCAN's transmit output CANTX through the transceiver to the receive input CANRX plus the transmitter delay compensation offset as configured by MCAN_TDCR.TDCO. The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (e.g. half of the bit time in the data phase). The position of the secondary sample point is rounded down to the next integer number of CAN core clock periods.

MCAN_PSR.TDCV shows the actual transmitter delay compensation value. MCAN_PSR.TDCV is cleared when MCAN_CCCR.INIT is set and is updated at each transmission of an FD frame while MCAN_DBTP.TDC is set.

The following boundary conditions have to be considered for the delay compensation implemented in the MCAN:

- The sum of the measured delay from CANTX to CANRX and the configured delay compensation offset MCAN_TDCR.TDCO has to be less than 6 bit times in the data phase.
- The sum of the measured delay from CANTX to CANRX and the configured delay compensation offset MCAN_TDCR.TDCO has to be less or equal 127 CAN core clock periods. In case this sum exceeds 127 CAN core clock periods, the maximum value of 127 CAN core clock periods is used for delay compensation.
- The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the SSPs.

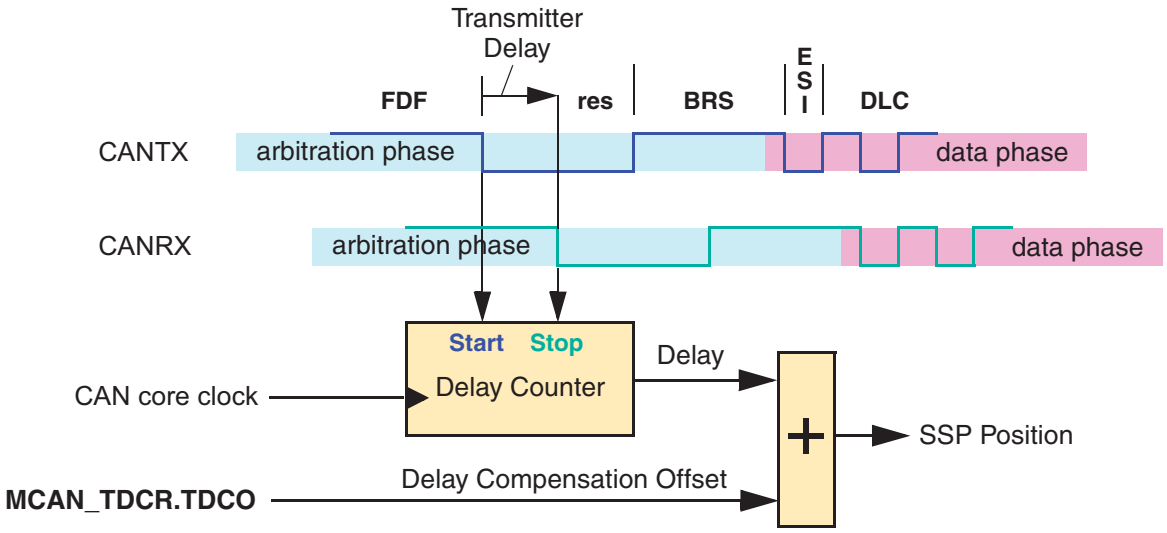
#### 49.5.1.4.2 Transmitter Delay Measurement

If transmitter delay compensation is enabled by programming MCAN_DBTP.TDC = '1', the measurement is started within each transmitted CAN FD frame at the falling edge of bit FDF to bit res. The measurement is stopped when this edge is seen at the receive input CANRX of the transmitter.

© 2021 Microchip Technology Inc. and its subsidiaries

The resolution of this measurement is one mtq.

**Figure 49-2. Transmitter Delay Measurement**




To avoid that a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received res bit, resulting in a to early SSP position, the use of a transmitter delay compensation filter window can be enabled by programming MCAN_TDCR.TDCF.

This defines a minimum value for the SSP position. Dominant edges on CANRX, that would result in an earlier SSP position are ignored for transmitter delay measurement. The measurement is stopped when the SSP position is at least MCAN_TDCR.TDCF AND CANRX is low.

### 49.5.1.5 Restricted Operation Mode

In Restricted Operation mode, the node is able to receive data and remote frames and to give acknowledge to valid frames, but it does not send data frames, remote frames, active error frames, or overload frames. In case of an error condition or overload condition, it does not send dominant bits, instead it waits for the occurrence of bus idle condition to resynchronize itself to the CAN communication. The error counters are not incremented. The processor can set the MCAN into Restricted Operation mode by setting bit MCAN_CCCR.ASM. The bit can only be set by the processor when both MCAN_CCCR.CCE and MCAN_CCCR.INIT are set to '1'. The bit can be reset by the processor at any time.

Restricted Operation mode is automatically entered when the Tx Handler was not able to read data from the Message RAM in time. To leave Restricted Operation mode, the processor has to reset MCAN_CCCR.ASM.

The Restricted Operation mode can be used in applications that adapt themselves to different CAN bit rates. In this case the application tests different bit rates and leaves the Restricted Operation mode after it has received a valid frame.

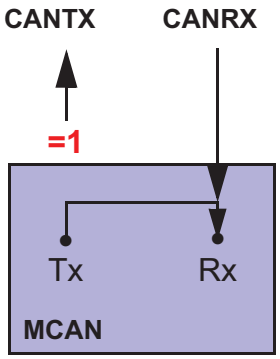
**Note:** The Restricted Operation Mode must not be combined with the Loop Back mode (internal or external).

### 49.5.1.6 Bus Monitoring Mode

The MCAN is set in Bus Monitoring mode by setting MCAN_CCCR.MON. In Bus Monitoring mode (see ISO11898-1, 10.12 Bus monitoring), the MCAN is able to receive valid data frames and valid remote frames, but cannot start a transmission. In this mode, it sends only recessive bits on the CAN bus. If the MCAN is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the MCAN monitors this dominant bit, although the CAN bus may remain in recessive state. In Bus Monitoring mode, the Tx Buffer Request Pending register (MCAN_TXBRP) is held in reset state.

The Bus Monitoring mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. The figure below shows the connection of signals CANTX and CANRX to the MCAN in Bus Monitoring mode.

**Figure 49-3. Pin Control in Bus Monitoring Mode**




**Bus Monitoring Mode**

#### 49.5.1.7 Disabled Automatic Retransmission

According to the CAN Specification (see ISO11898-1, 6.3.3 Recovery Management), the MCAN provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default automatic retransmission is enabled. To support time-triggered communication as described in ISO 11898-1, chapter 9.2, the automatic retransmission may be disabled via MCAN_CCCR.DAR.

##### 49.5.1.7.1 Frame Transmission in DAR Mode

In DAR mode, all transmissions are automatically cancelled after they start on the CAN bus. A Tx Buffer's Tx Request Pending bit TXBRP.TRPx is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

- Successful transmission:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx not set
- Successful transmission in spite of cancellation:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx set
- Arbitration lost or frame transmission disturbed:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx not set

  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx set

In case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event FIFO element is written with Event Type ET = "10" (transmission in spite of cancellation).

#### 49.5.1.8 Power-down (Sleep Mode)

The MCAN can be set into Power-down mode via bit MCAN_CCCR.CSR.

When all pending transmission requests have completed, the MCAN waits until bus idle state is detected. Then the MCAN sets MCAN_CCCR.INIT to prevent any further CAN transfers. Now the MCAN acknowledges that it is ready for power down by setting to one the bit MCAN_CCCR.CSA. In this state, before the clocks are switched off, further register accesses can be made. A write access to MCAN_CCCR.INIT will have no effect. Now the bus clock (peripheral clock) and the CAN core clock may be switched off.

To leave Power-down mode, the application has to turn on the MCAN clocks before clearing CC Control Register flag MCAN_CCCR.CSR. The MCAN will acknowledge this by clearing MCAN_CCCR.CSA. The application can then restart CAN communication by clearing the bit CCCR.INIT.

#### 49.5.1.9 Test Modes

To enable write access to the MCAN Test register (MCAN_TEST) (see Section 7.6), bit MCAN_CCCR.TEST must be set. This allows the configuration of the test modes and test functions.

Four output functions are available for the CAN transmit pin CANTX by programming MCAN_TEST.TX. Additionally to its default function – the serial data output – it can drive the CAN Sample Point signal to monitor the MCAN's

bit timing and it can drive constant dominant or recessive values. The actual value at pin CANRX can be read from MCAN_TEST.RX. Both functions can be used to check the CAN bus' physical layer.

Due to the synchronization mechanism between CAN clock and system bus clock domain, there may be a delay of several system bus clock periods between writing to MCAN_TEST.TX until the new configuration is visible at output pin CANTX. This applies also when reading input pin CANRX via MCAN_TEST.RX.

**Note:** Test modes should be used for production tests or self-test only. The software control for pin CANTX interferes with all CAN protocol functions. It is not recommended to use test modes for application.

#### 49.5.1.9.1 External Loop Back Mode

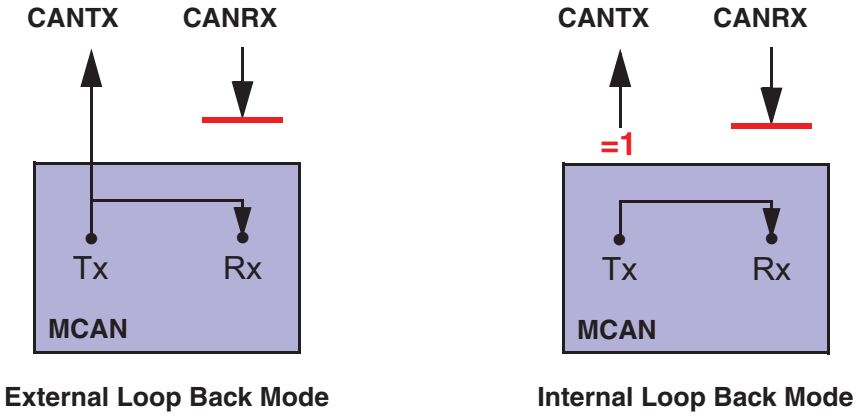
The MCAN can be set in External Loop Back mode by setting the bit MCAN_TEST.LBCK. In Loop Back mode, the MCAN treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into an Rx Buffer or an Rx FIFO. The figure below shows the connection of signals CANTX and CANRX to the MCAN in External Loop Back mode.

This mode is provided for hardware self-test. To be independent from external stimulation, the MCAN ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back mode. In this mode, the MCAN performs an internal feedback from its Tx output to its Rx input. The actual value of the CANRX input pin is disregarded by the MCAN. The transmitted messages can be monitored at the CANTX pin.

#### 49.5.1.9.2 Internal Loop Back Mode

Internal Loop Back mode is entered by setting bits MCAN_TEST.LBCK and MCAN_CCCR.MON. This mode can be used for a "Hot Selftest", meaning the MCAN can be tested without affecting a running CAN system connected to the pins CANTX and CANRX. In this mode, pin CANRX is disconnected from the MCAN, and pin CANTX is held recessive. The figure below shows the connection of CANTX and CANRX to the MCAN when Internal Loop Back mode is enabled.

**Figure 49-4. Pin Control in Loop Back Modes**




### 49.5.2 Timestamp Generation

For timestamp generation the MCAN supplies a 16-bit wrap-around counter. A prescaler TSCC.TCP can be configured to clock the counter in multiples of CAN bit times (1…16). The counter is readable via MCAN_TSCV.TSC. A write access to the Timestamp Counter Value register (MCAN_TSCV) resets the counter to zero. When the timestamp counter wraps around, interrupt flag MCAN_IR.TSW is set.

On start of frame reception / transmission the counter value is captured and stored into the timestamp section of an Rx Buffer / Rx FIFO (RXTS[15:0]) or Tx Event FIFO (TXTS[15:0]) element.

By programming bit MCAN_TSCC.TSS an external 16-bit timestamp can be used. See Timestamping for more details.

### 49.5.3 Timeout Counter

To signal timeout conditions for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO, the MCAN supplies a 16-bit Timeout Counter. It operates as down-counter and uses the same prescaler controlled by TSCC.TCP as the Timestamp Counter. The Timeout Counter is configured via the Timeout Counter Configuration register (MCAN_TOCC). The

© 2021 Microchip Technology Inc. and its subsidiaries

actual counter value can be read from MCAN_TOCV.TOC. The Timeout Counter can only be started while MCAN_CCCR.INIT = '0'. It is stopped when MCAN_CCCR.INIT = '1', e.g. when the MCAN enters Bus_Off state.

The operating mode is selected by MCAN_TOCC.TOS. When operating in Continuous mode, the counter starts when MCAN_CCCR.INIT is reset. A write to MCAN_TOCV presets the counter to the value configured by MCAN_TOCC.TOP and continues down-counting.

When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by MCAN_TOCC.TOP. Down-counting is started when the first FIFO element is stored. Writing to MCAN_TOCV has no effect.

When the counter reaches zero, interrupt flag MCAN_IR.TOO is set. In Continuous mode, the counter is immediately restarted at MCAN_TOCC.TOP.

**Note:** The clock signal for the Timeout Counter is derived from the CAN Core's sample point signal. Therefore the point in time where the Timeout Counter is decremented may vary due to the synchronization / re-synchronization mechanism of the CAN Core. If the bit rate switch feature in CAN FD is used, the timeout counter is clocked differently in arbitration and data field.

### 49.5.4 Rx Handling

The Rx Handler controls the acceptance filtering, the transfer of received messages to the Rx Buffers or to one of the two Rx FIFOs, as well as the Rx FIFO's Put and Get Indices.

#### 49.5.4.1 Acceptance Filtering

The MCAN offers the possibility to configure two sets of acceptance filters, one for standard identifiers and one for extended identifiers. These filters can be assigned to an Rx Buffer or to Rx FIFO 0,1. For acceptance filtering each list of filters is executed from element #0 until the first matching element. Acceptance filtering stops at the first matching element. The following filter elements are not evaluated for this message.

The main features are:

- Each filter element can be configured as
  - range filter (from - to)
  - filter for one or two dedicated IDs
  - classic bit mask filter
- Each filter element is configurable for acceptance or rejection filtering
- Each filter element can be enabled / disabled individually
- Filters are checked sequentially, execution stops with the first matching filter element

Related configuration registers are:

- Global Filter Configuration (MCAN_GFC)
- Standard ID Filter Configuration (MCAN_SIDFC)
- Extended ID Filter Configuration (MCAN_XIDFC)
- Extended ID and Mask (MCAN_XIDAM)

Depending on the configuration of the filter element (SFEC/EFEC) a match triggers one of the following actions:

- Store received frame in FIFO 0 or FIFO 1
- Store received frame in Rx Buffer
- Store received frame in Rx Buffer and generate pulse at filter event pin
- Reject received frame
- Set High Priority Message interrupt flag (MCAN_IR.HPM)
- Set High Priority Message interrupt flag (MCAN_IR.HPM) and store received frame in FIFO 0 or FIFO 1

Acceptance filtering is started after the complete identifier has been received. After acceptance filtering has completed, and if a matching Rx Buffer or Rx FIFO has been found, the Message Handler starts writing the received message data in portions of 32 bit to the matching Rx Buffer or Rx FIFO. If the CAN protocol controller has detected an error condition (e.g. CRC error), this message is discarded with the following impact on the effected Rx Buffer or Rx FIFO:

- Rx Buffer
  New Data flag of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data. For error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC.
- Rx FIFO
  Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data. For error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC. In case the matching Rx FIFO is operated in Overwrite mode, the boundary conditions described in Rx FIFO Overwrite Mode have to be considered.
  **Note:** When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filter(s) used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.

#### 49.5.4.1.1 Range Filter

The filter matches for all received frames with Message IDs in the range defined by SF1ID/SF2ID resp. EF1ID/EF2ID.

There are two possibilities when range filtering is used together with extended frames:

- EFT = “00”: The Message ID of received frames is ANDed with MCAN_XIDAM before the range filter is applied.
- EFT = “11”: MCAN_XIDAM is not used for range filtering.

#### 49.5.4.1.2 Filter for Specific IDs

A filter element can be configured to filter for one or two specific Message IDs. To filter for one specific Message ID, the filter element has to be configured with SF1ID = SF2ID resp. EF1ID = EF2ID.

#### 49.5.4.1.3 Classic Bit Mask Filter

Classic bit mask filtering is intended to filter groups of Message IDs by masking single bits of a received Message ID. With classic bit mask filtering SF1ID/EF1ID is used as Message ID filter, while SF2ID/EF2ID is used as filter mask.

A zero bit at the filter mask will mask out the corresponding bit position of the configured ID filter, e.g. the value of the received Message ID at that bit position is not relevant for acceptance filtering. Only those bits of the received Message ID where the corresponding mask bits are one are relevant for acceptance filtering.

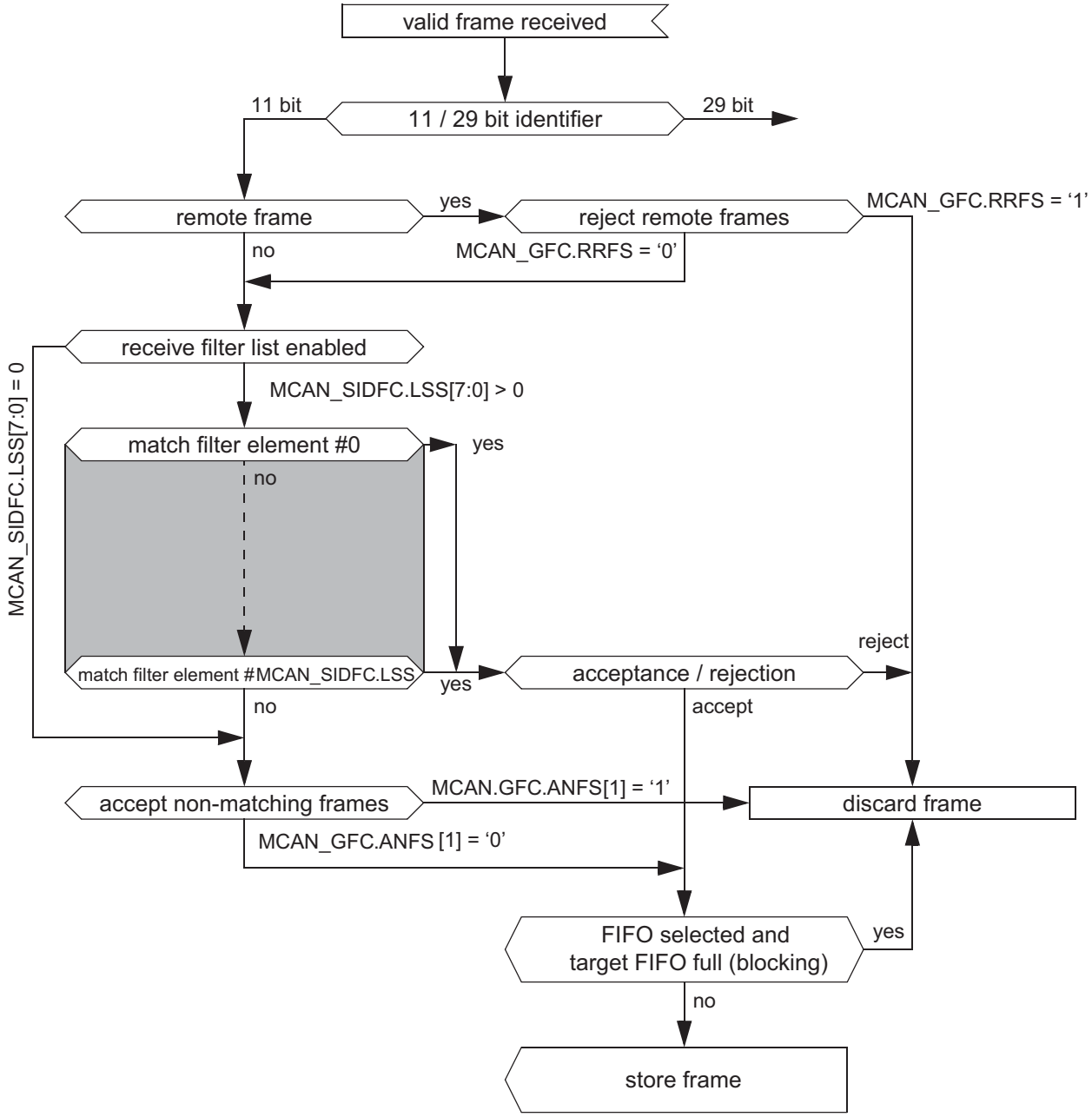
In case all mask bits are one, a match occurs only when the received Message ID and the Message ID filter are identical. If all mask bits are zero, all Message IDs match.

#### 49.5.4.1.4 Standard Message ID Filtering

The figure below shows the flow for standard Message ID (11-bit Identifier) filtering. The Standard Message ID Filter element is described in 49.5.7.5 Standard Message ID Filter Element.

Controlled by MCAN_GFC and MCAN_SIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

**Figure 49-5. Standard Message ID Filter Path**



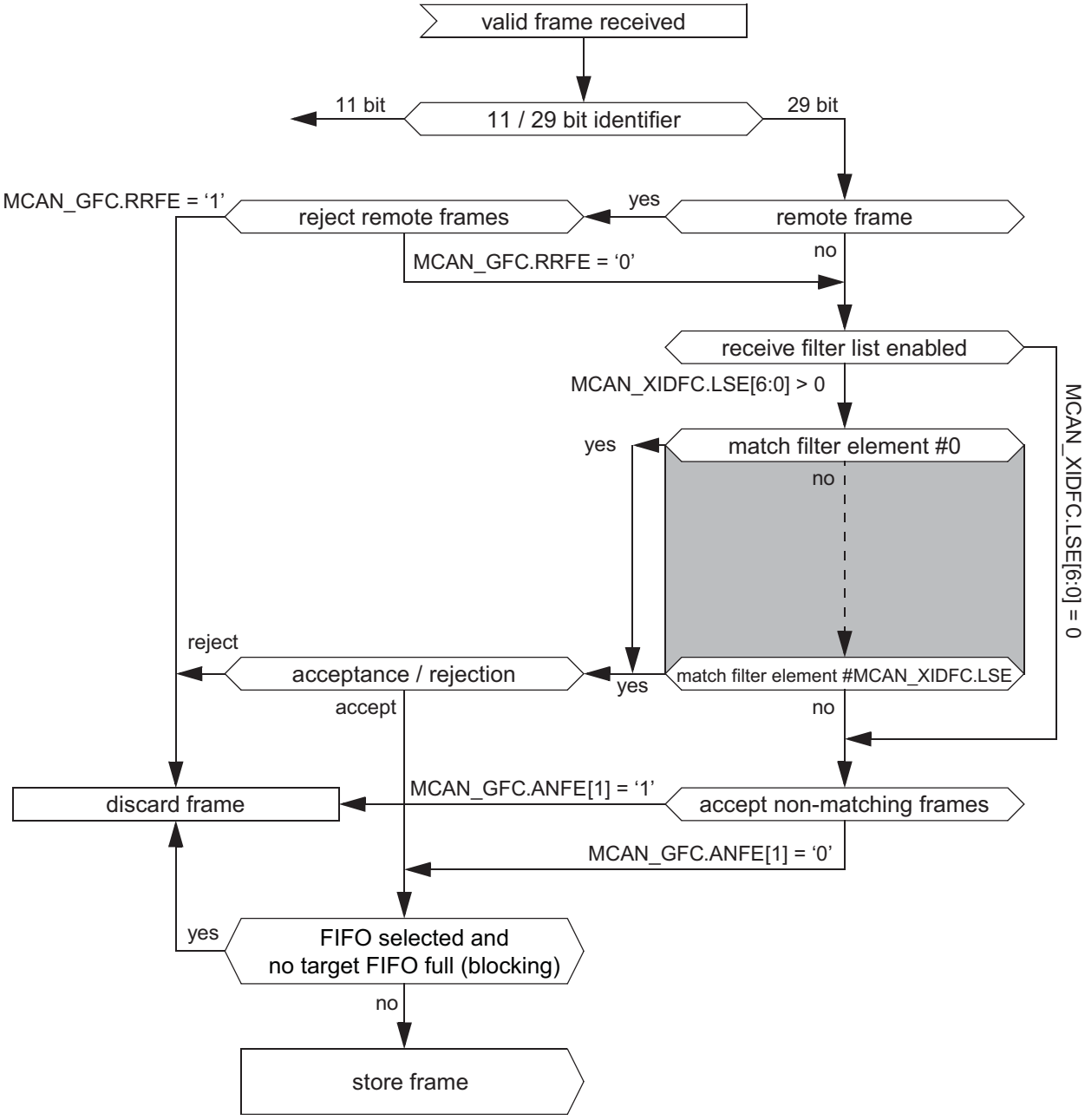

**Extended Message ID Filtering**

The figure below shows the flow for extended Message ID (29-bit Identifier) filtering. The Extended Message ID Filter element is described in 49.5.7.6 Extended Message ID Filter Element.

Controlled by MCAN_GFC and MCAN_XIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

MCAN_XIDAM is ANDed with the received identifier before the filter list is executed.

**Figure 49-6. Extended Message ID Filter Path**



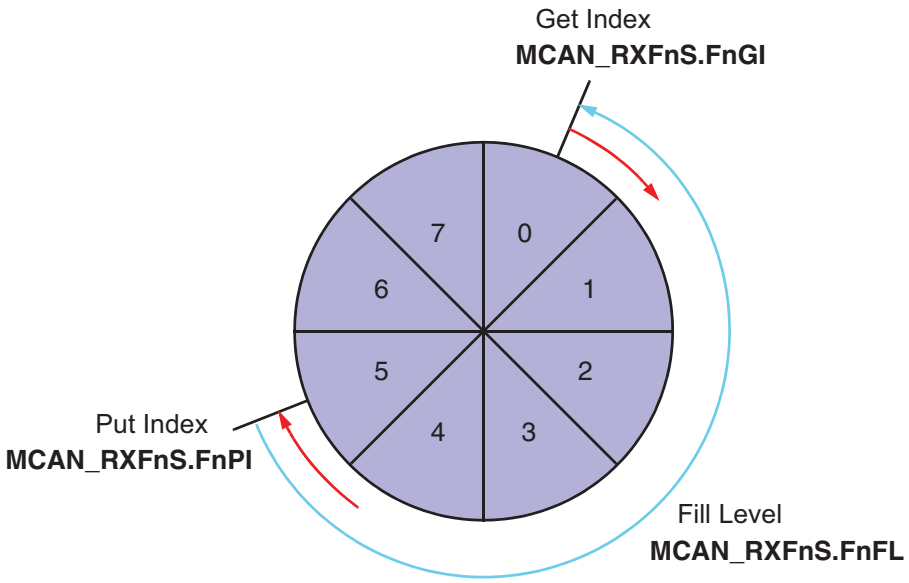

### 49.5.4.2 Rx FIFOs

Rx FIFO 0 and Rx FIFO 1 can be configured to hold up to 64 elements each. Configuration of the two Rx FIFOs is done via the Rx FIFO 0 Configuration register (MCAN_RXF0C) and the Rx FIFO 1 Configuration register (MCAN_RXF1C).

Received messages that passed acceptance filtering are transferred to the Rx FIFO as configured by the matching filter element. For a description of the filter mechanisms available for Rx FIFO 0 and Rx FIFO 1, see Acceptance Filtering. The Rx FIFO element is described in Rx Buffer and FIFO Element.

To avoid an Rx FIFO overflow, the Rx FIFO watermark can be used. When the Rx FIFO fill level reaches the Rx FIFO watermark configured by MCAN_RXFnC.FnWM, interrupt flag MCAN_IR.RFnW is set. When the Rx FIFO Put Index reaches the Rx FIFO Get Index, an Rx FIFO Full condition is signalled by MCAN_RXFnS.FnF. In addition, the interrupt flag MCAN_IR.RFnF is set.

**Figure 49-7. Rx FIFO Status**




When reading from an Rx FIFO, Rx FIFO Get Index MCAN_RXFnS.FnGI × FIFO Element Size has to be added to the corresponding Rx FIFO start address MCAN_RXFnC.FnSA.

**Table 49-2. Rx Buffer / FIFO Element Size**

| MCAN_RXESC.RBDS[2:0]<br>MCAN_RXESC.FnDS[2:0] | Data Field<br>[bytes] | FIFO Element Size<br>[RAM words] |
|---|---|---|
| 0 | 8 | 4 |
| 1 | 12 | 5 |
| 2 | 16 | 6 |
| 3 | 20 | 7 |
| 4 | 24 | 8 |
| 5 | 32 | 10 |
| 6 | 48 | 14 |
| 7 | 64 | 18 |

#### 49.5.4.2.1 Rx FIFO Blocking Mode

The Rx FIFO Blocking mode is configured by MCAN_RXFnC.FnOM = ‘0’. This is the default operating mode for the Rx FIFOs.

When an Rx FIFO full condition is reached (MCAN_RXFnS.FnPI = MCAN_RXFnS.FnGI), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signalled by MCAN_RXFnS.FnF = ‘1’. In addition, the interrupt flag MCAN_IR.RFnF is set.

In case a message is received while the corresponding Rx FIFO is full, this message is discarded and the message lost condition is signalled by MCAN_RXFnS.RFnL = ‘1’. In addition, the interrupt flag MCAN_IR.RFnL is set.

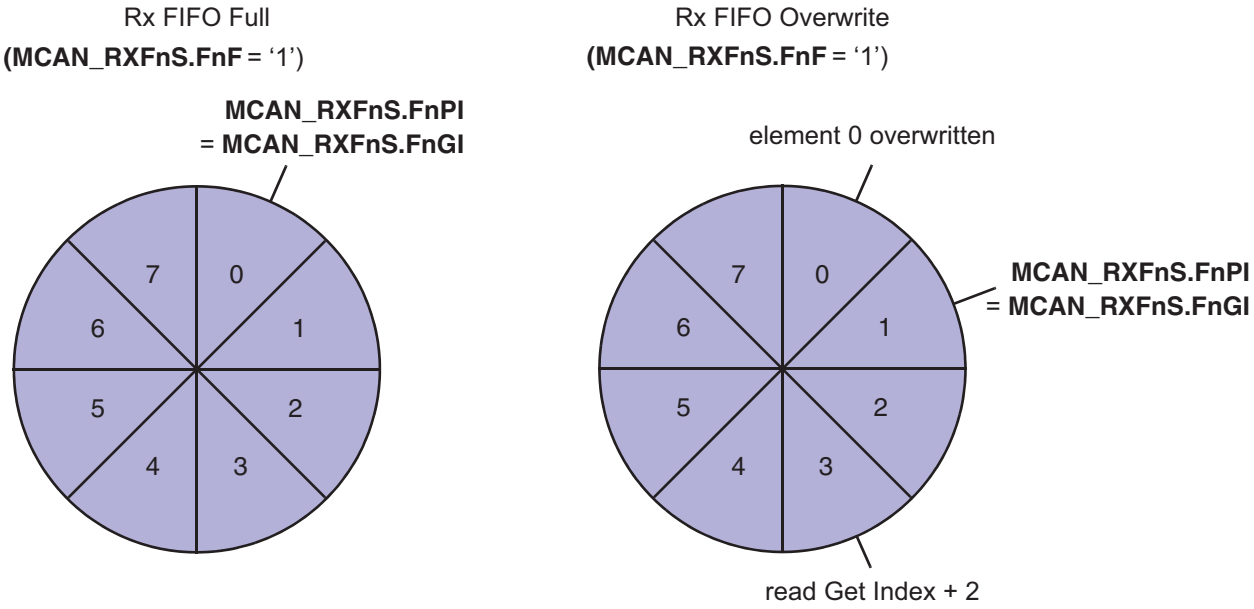
#### 49.5.4.2.2 Rx FIFO Overwrite Mode

The Rx FIFO Overwrite mode is configured by MCAN_RXFnC.FnOM = ‘1’.

When an Rx FIFO full condition (MCAN_RXFnS.FnPI = MCAN_RXFnS.FnGI) is signalled by MCAN_RXFnS.FnF = ‘1’, the next message accepted for the FIFO will overwrite the oldest FIFO message. Put and get index are both incremented by one.

When an Rx FIFO is operated in Overwrite mode and an Rx FIFO full condition is signalled, reading of the Rx FIFO elements should start at least at get index + 1. The reason for that is, that it might happen, that a received message is written to the Message RAM (put index) while the processor is reading from the Message RAM (get index). In this case inconsistent data may be read from the respective Rx FIFO element. Adding an offset to the get index when reading from the Rx FIFO avoids this problem. The offset depends on how fast the processor accesses the Rx FIFO. The figure below shows an offset of two with respect to the get index when reading the Rx FIFO. In this case the two messages stored in element 1 and 2 are lost.

**Figure 49-8. Rx FIFO Overflow Handling**




After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index MCAN_RXFnA.FnA. This increments the get index to that element number. In case the put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (MCAN_RXFnS.FnF = '0').

### 49.5.4.3 Dedicated Rx Buffers

The MCAN supports up to 64 dedicated Rx Buffers. The start address of the dedicated Rx Buffer section is configured via MCAN_RXBC.RBSA.

For each Rx Buffer, a Standard or Extended Message ID Filter Element with SFEC / EFEC = 7 and SFID2 / EFID2[10:9] = 0 has to be configured.

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element. The format is the same as for an Rx FIFO element. In addition, the flag MCAN_IR.DRX (Message stored in dedicated Rx Buffer) in MCAN_IR is set.

**Table 49-3. Example Filter Configuration for Rx Buffers**

| Filter Element | SFID1[10:0] EFID1[28:0] | SFID2[10:9] EFID2[10:9] | SFID2[5:0] EFID2[5:0] |
|---|---|---|---|
| 0 | ID message 1 | 0 | 0 |
| 1 | ID message 2 | 0 | 1 |
| 2 | ID message 3 | 0 | 2 |

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in the New Data 1 register (MCAN_NDAT1) and New Data 2 register (MCAN_NDAT2) is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the processor by writing a '1' to the respective bit position.

While an Rx Buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

#### 49.5.4.3.1 Rx Buffer Handling

- Reset interrupt flag IR.DRX
- Read New Data registers
- Read messages from Message RAM
- Reset New Data flags of processed messages

### 49.5.4.4 Debug on CAN Support

Debug messages are stored into Rx Buffers. For debug handling three consecutive Rx buffers (e.g. #61, #62, #63) have to be used for storage of debug messages A, B, and C. The format is the same as for an Rx Buffer or an Rx FIFO element (see Rx Buffer and FIFO Element).

Advantage: Fixed start address for the DMA transfers (relative to MCAN_RXBC.RBSA), no additional configuration required.

For filtering of debug messages Standard / Extended Filter Elements with SFEC / EFEC = '111' have to be set up. Messages matching these filter elements are stored into the Rx Buffers addressed by SFID2 / EFID2[5:0].

After message C has been stored, the DMA request output m_can_dma_req is activated and the three messages can be read from the Message RAM under DMA control. The RAM words holding the debug messages will not be changed by the MCAN while m_can_dma_req is activated. The behavior is similar to that of an Rx Buffer with its New Data flag set.

After the DMA has completed, the MCAN is prepared to receive the next set of debug messages.

#### 49.5.4.4.1 Filtering for Debug Messages

Filtering for debug messages is done by configuring one Standard / Extended Message ID Filter Element for each of the three debug messages. To enable a filter element to filter for debug messages SFEC / EFEC has to be programmed to "111". In this case fields SFID1 / SFID2 and EFID1 / EFID2 have a different meaning. While SFID2 / EFID2[10:9] controls the debug message handling state machine, SFID2 / EFID2[5:0] controls the location for storage of a received debug message.

When a debug message is stored, neither the respective New Data flag nor MCAN_IR.DRX are set. The reception of debug messages can be monitored via RXF1S.DMS.

**Table 49-4. Example Filter Configuration for Debug Messages**

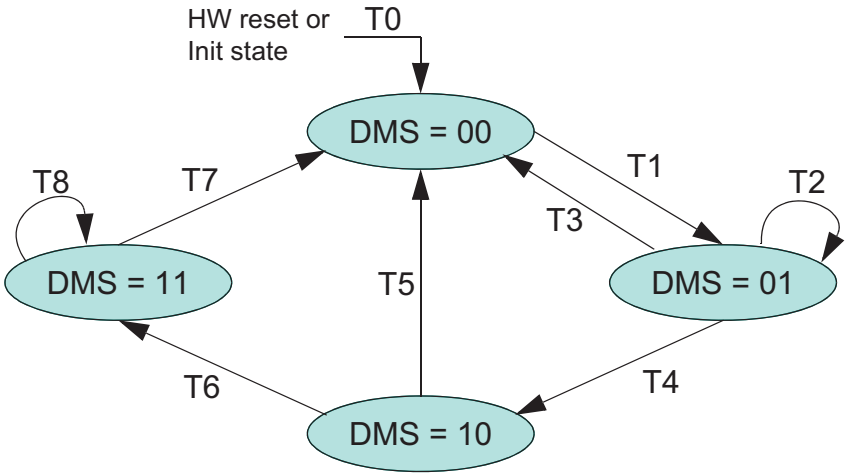
| Filter Element | SFID1[10:0] EFID1[28:0] | SFID2[10:9] EFID2[10:9] | SFID2[5:0] EFID2[5:0] |
|---|---|---|---|
| 0 | ID debug message A | 1 | 11 1101 |
| 1 | ID debug message B | 2 | 11 1110 |
| 2 | ID debug message C | 3 | 11 1111 |

#### 49.5.4.4.2 Debug Message Handling

The debug message handling state machine ensures that debug messages are stored to three consecutive Rx Buffers in the correct order. If some messages are missing, the process is restarted. The DMA request is activated only when all three debug messages A, B, C have been received in the correct order.

The status of the debug message handling state machine is signalled via MCAN_RXF1S.DMS.

**Figure 49-9. Debug Message Handling State Machine**




T0: reset m_can_dma_req output, enable reception of debug messages A, B, and C

T1: reception of debug message A

T2: reception of debug message A

T3: reception of debug message C

T4: reception of debug message B

T5: reception of debug messages A, B

T6: reception of debug message C

T7: DMA transfer completed

T8: reception of debug message A,B,C (message rejected)

### 49.5.5 Tx Handling

The Tx Handler handles transmission requests for the dedicated Tx Buffers, the Tx FIFO, and the Tx Queue. It controls the transfer of transmit messages to the CAN Core, the Put and Get Indices, and the Tx Event FIFO. Up to 32 Tx Buffers can be set up for message transmission. The CAN mode for transmission (Classic CAN or CAN FD) can be configured separately for each Tx Buffer element. The Tx Buffer element is described in Tx Buffer Element. The table below describes the possible configurations for frame transmission.

**Table 49-5. Possible Configurations for Frame Transmission**

| MCAN_CCCR | | Tx Buffer Element | | Frame Transmission |
|---|---|---|---|---|
| BRSE | FDOE | FDF | BRS | |
| ignored | 0 | ignored | ignored | Classic CAN |
| 0 | 1 | 0 | ignored | Classic CAN |
| 0 | 1 | 1 | ignored | FD without bit rate switching |
| 1 | 1 | 0 | ignored | Classic CAN |
| 1 | 1 | 1 | 0 | FD without bit rate switching |
| 1 | 1 | 1 | 1 | FD with bit rate switching |

**Note:** AUTOSAR requires at least three Tx Queue Buffers and support of transmit cancellation.

The Tx Handler starts a Tx scan to check for the highest priority pending Tx request (Tx Buffer with lowest Message ID) when MCAN_TXBRP is updated, or when a transmission has been started.

#### 49.5.5.1 Transmit Pause

The transmit pause feature is intended for use in CAN systems where the CAN message identifiers are (permanently) specified to specific values and cannot easily be changed. These message identifiers may have a higher CAN arbitration priority than other defined messages, while in a specific application their relative arbitration priority should be inverse. This may lead to a case where one ECU sends a burst of CAN messages that cause another ECU's CAN messages to be delayed because that other messages have a lower CAN arbitration priority.

If e.g. CAN ECU-1 has the transmit pause feature enabled and is requested by its application software to transmit four messages, it will, after the first successful message transmission, wait for two CAN bit times of bus idle before it is allowed to start the next requested message. If there are other ECUs with pending messages, those messages are started in the idle time, they would not need to arbitrate with the next message of ECU-1. After having received a message, ECU-1 is allowed to start its next transmission as soon as the received message releases the CAN bus.

The transmit pause feature is controlled by bit MCAN_CCCR.TXP. If the bit is set, the MCAN will, each time it has successfully transmitted a message, pause for two CAN bit times before starting the next transmission. This enables other CAN nodes in the network to transmit messages even if their messages have lower prior identifiers. Default is transmit pause disabled (MCAN_CCCR.TXP = '0').

This feature looses up burst transmissions coming from a single node and it protects against "babbling idiot" scenarios where the application program erroneously requests too many transmissions.

#### 49.5.5.2 Dedicated Tx Buffers

Dedicated Tx Buffers are intended for message transmission under complete control of the processor. Each dedicated Tx Buffer is configured with a specific Message ID. In case that multiple Tx Buffers are configured with the same Message ID, the Tx Buffer with the lowest buffer number is transmitted first.

If the data section has been updated, a transmission is requested by an Add Request via MCAN_TXBAR.ARn. The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx Queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

A dedicated Tx Buffer allocates Element Size 32-bit words in the Message RAM (see the table below). Therefore the start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index (0…31) × Element Size to the Tx Buffer Start Address TXBC.TBSA.

**Table 49-6. Tx Buffer / FIFO / Queue Element Size**

| TXESC.TBDS[2:0] | Data Field [bytes] | Element Size [RAM words] |
|---|---|---|
| 0 | 8 | 4 |
| 1 | 12 | 5 |
| 2 | 16 | 6 |
| 3 | 20 | 7 |
| 4 | 24 | 8 |
| 5 | 32 | 10 |
| 6 | 48 | 14 |
| 7 | 64 | 18 |

#### 49.5.5.3 Tx FIFO

Tx FIFO operation is configured by programming MCAN_TXBC.TFQM to '0'. Messages stored in the Tx FIFO are transmitted starting with the message referenced by the Get Index MCAN_TXFQS.TFGI. After each transmission the Get Index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO enables transmission of messages with the same Message ID from different Tx Buffers in the order these messages have been written to the Tx FIFO. The MCAN calculates the Tx FIFO Free Level MCAN_TXFQS.TFFL as difference between Get and Put Index. It indicates the number of available (free) Tx FIFO elements.

© 2021 Microchip Technology Inc. and its subsidiaries

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQPI. An Add Request increments the Put Index to the next free Tx FIFO element. When the Put Index reaches the Get Index, Tx FIFO Full (MCAN_TXFQS.TFQF = '1') is signalled. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

When a single message is added to the Tx FIFO, the transmission is requested by writing a '1' to the TXBAR bit related to the Tx Buffer referenced by the Tx FIFO's Put Index.

When multiple (n) messages are added to the Tx FIFO, they are written to n consecutive Tx Buffers starting with the Put Index. The transmissions are then requested via MCAN_TXBAR. The Put Index is then cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx Buffers as indicated by the Tx FIFO Free Level.

When a transmission request for the Tx Buffer referenced by the Get Index is cancelled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level is recalculated. When transmission cancellation is applied to any other Tx Buffer, the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates Element Size 32-bit words in the Message RAM (see the table Table 49-6). Therefore the start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQPI (0…31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA.

#### 49.5.5.4 Tx Queue

Tx Queue operation is configured by programming MCAN_TXBC.TFQM to '1'. Messages stored in the Tx Queue are transmitted starting with the message with the lowest Message ID (highest priority). In case that multiple Queue Buffers are configured with the same Message ID, the Queue Buffer with the lowest buffer number is transmitted first.

New messages have to be written to the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQPI. An Add Request cyclically increments the Put Index to the next free Tx Buffer. In case that the Tx Queue is full (MCAN_TXFQS.TFQF = '1'), the Put Index is not valid and no further message should be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been cancelled.

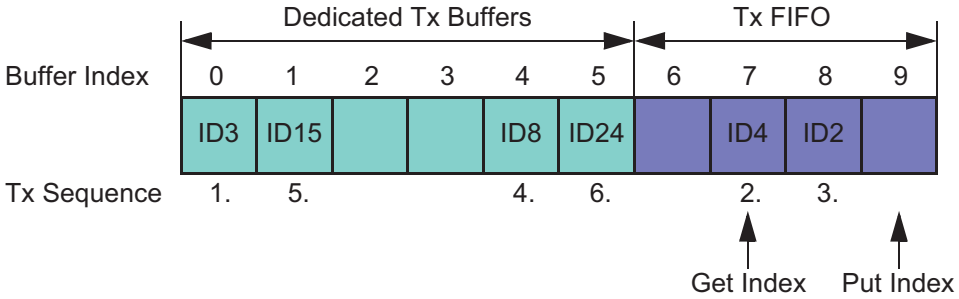
The application may use register MCAN_TXBRP instead of the Put Index and may place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates Element Size 32-bit words in the Message RAM (see the table Tx Buffer / FIFO / Queue Element Size). Therefore the start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQPI (0…31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA.

#### 49.5.5.5 Mixed Dedicated Tx Buffers / Tx FIFO

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx FIFO. The number of dedicated Tx Buffers is configured by MCAN_TXBC.NDTB. The number of Tx Buffers assigned to the Tx FIFO is configured by MCAN_TXBC.TFQS. In case MCAN_TXBC.TFQS is programmed to zero, only dedicated Tx Buffers are used.

**Figure 49-10. Example of Mixed Configuration Dedicated Tx Buffers / Tx FIFO**




Tx prioritization:

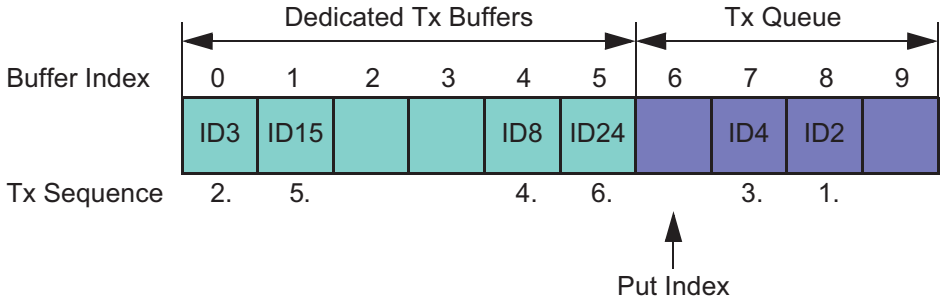
- Scan dedicated Tx Buffers and oldest pending Tx FIFO Buffer (referenced by MCAN_TXFS.TFGI)

- Buffer with lowest Message ID gets highest priority and is transmitted next

### 49.5.5.6 Mixed Dedicated Tx Buffers / Tx Queue

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx Queue. The number of dedicated Tx Buffers is configured by MCAN_TXBC.NDTB. The number of Tx Queue Buffers is configured by MCAN_TXBC.TFQS. In case MCAN_TXBC.TFQS is programmed to zero, only dedicated Tx Buffers are used.

**Figure 49-11. Example of Mixed Configuration Dedicated Tx Buffers / Tx Queue**




Tx prioritization:

- Scan all Tx Buffers with activated transmission request
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next

### 49.5.5.7 Transmit Cancellation

The MCAN supports transmit cancellation. This feature is especially intended for gateway applications and AUTOSAR-based applications. To cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue Buffer, the processor has to write a '1' to the corresponding bit position (=number of Tx Buffer) of register MCAN_TXBCR. Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signalled by setting the corresponding bit of register MCAN_TXBCF to '1'.

In case a transmit cancellation is requested while a transmission from a Tx Buffer is already ongoing, the corresponding TXBRP bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding MCAN_TXBTO and MCAN_TXBCF bits are set. If the transmission was not successful, it is not repeated and only the corresponding MCAN_TXBCF bit is set.

**Note:** In case a pending transmission is cancelled immediately before this transmission could have been started, there follows a short time window where no transmission is started even if another message is also pending in this node. This may enable another node to transmit a message which may have a lower priority than the second message in this node.

### 49.5.5.8 Tx Event Handling

To support Tx event handling the MCAN has implemented a Tx Event FIFO. After the MCAN has transmitted a message on the CAN bus, Message ID and timestamp are stored in a Tx Event FIFO element. To link a Tx event to a Tx Event FIFO element, the Message Marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

The Tx Event FIFO can be configured to a maximum of 32 elements. The Tx Event FIFO element is described in Debug on CAN Support.

When a Tx Event FIFO full condition is signalled by IR.TEFF, no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO Get Index has been incremented. In case a Tx event occurs while the Tx Event FIFO is full, this event is discarded and interrupt flag MCAN_IR.TEFL is set.

To avoid a Tx Event FIFO overflow, the Tx Event FIFO watermark can be used. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by MCAN_TXEFC.EFWM, interrupt flag MCAN_IR.TEFW is set.

When reading from the Tx Event FIFO, two times the Tx Event FIFO Get Index MCAN_TXEFS.EFGI has to be added to the Tx Event FIFO start address MCAN_TXEFC.EFSA.

© 2021 Microchip Technology Inc. and its subsidiaries

### 49.5.6 FIFO Acknowledge Handling

The Get Indices of Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index in the registers MCAN_RXF0A, MCAN_RXF1A and MCAN_TXEFA. Writing to the FIFO Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level. There are two use cases:

When only a single element has been read from the FIFO (the one being pointed to by the Get Index), this Get Index value is written to the FIFO Acknowledge Index.

When a sequence of elements has been read from the FIFO, it is sufficient to write the FIFO Acknowledge Index only once at the end of that read sequence (value: Index of the last element read), to update the FIFO's Get Index.

Due to the fact that the processor has free access to the MCAN's Message RAM, special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This might be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also alters the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

**Note:** The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The MCAN does not check for erroneous values.

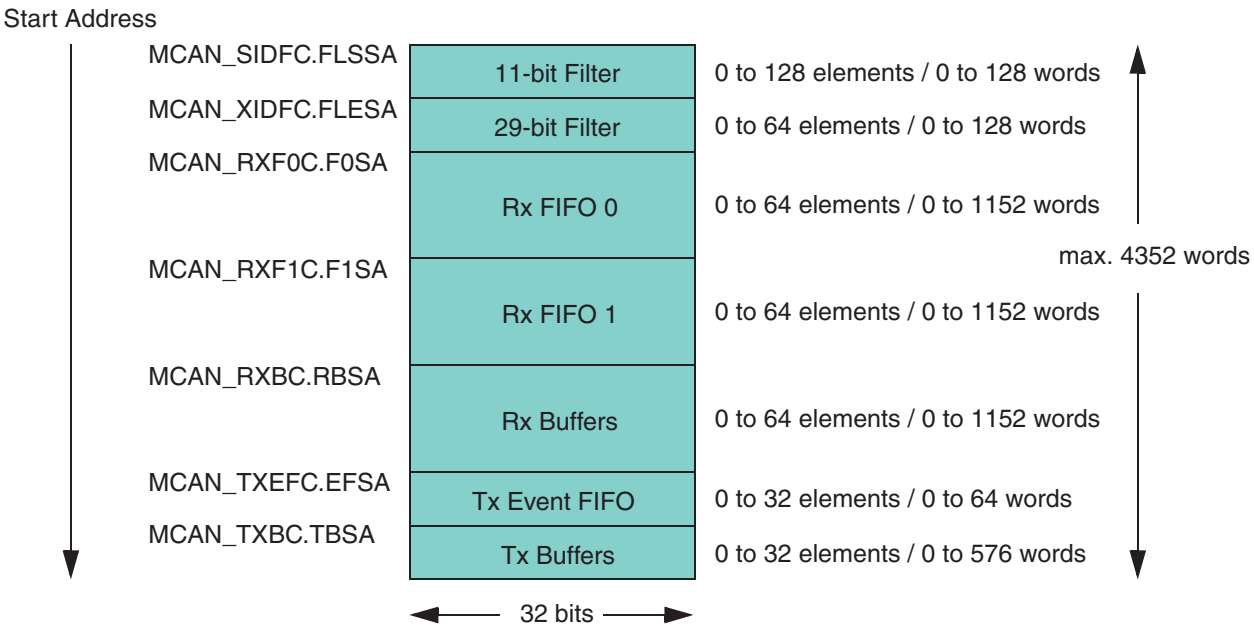
### 49.5.7 Message RAM

#### 49.5.7.1 Message RAM Configuration

The Message RAM has a width of 32 bits. The MCAN module can be configured to allocate up to 4352 words in the Message RAM. It is not necessary to configure each of the sections listed in the figure below, nor is there any restriction with respect to the sequence of the sections.

When operated in CAN FD mode, the required Message RAM size depends on the element size configured for Rx FIFO0, Rx FIFO1, Rx Buffers, and Tx Buffers via MCAN_RXESC.F0DS, MCAN_RXESC.F1DS, MCAN_RXESC.RBDS, and MCAN_TXESC.TBDS.

**Figure 49-12. Message RAM Configuration**




When the MCAN addresses the Message RAM, it addresses 32-bit words, not single bytes. The configurable start addresses are 32-bit word addresses; i.e., only bits 15 to 2 are evaluated, the two least significant bits are ignored.

**Note:** The MCAN does not check for erroneous configuration of the Message RAM. The configuration of the start addresses of the different sections and the number of elements of each section must be checked carefully to avoid falsification or loss of data.

© 2021 Microchip Technology Inc. and its subsidiaries

### 49.5.7.2 Rx Buffer and FIFO Element

Up to 64 Rx Buffers and two Rx FIFOs can be configured in the Message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The structure of a Rx Buffer / FIFO element is shown in the table below. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register MCAN_RXESC.

**Table 49-7. Rx Buffer and FIFO Element**

| | 31 | | | 24 | 23 | | | | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R0 | ESI | XTD | RTR | ID[28:0] | | | | | | | | | |
| R1 | ANMF | FIDX[6:0] | | | – | FDF | BRS | DLC[3:0] | | RXTS[15:0] | | | |
| R2 | DB3[7:0] | | | | DB2[7:0] | | | | | DB1[7:0] | | DB0[7:0] | |
| R3 | DB7[7:0] | | | | DB6[7:0] | | | | | DB5[7:0] | | DB4[7:0] | |
| ... | ... | | | | ... | | | | | ... | | ... | |
| Rn | DBm[7:0] | | | | DBm-1[7:0] | | | | | DBm-2[7:0] | | DBm-3[7:0] | |

• R0 Bit 31 ESI: Error State Indicator

0: Transmitting node is error active.

1: Transmitting node is error passive.

• R0 Bit 30 XTD: Extended Identifier

Signals to the processor whether the received frame has a standard or extended identifier.

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• R0 Bit 29 RTR: Remote Transmission Request

Signals to the processor whether the received frame is a data frame or a remote frame.

0: Received frame is a data frame.

1: Received frame is a remote frame.

**Note:** There are no remote frames in CAN FD format. In case a CAN FD frame was received (FDF = 1), bit RTR reflects the state of the reserved bit r1.

• R0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

• R1 Bit 31 ANMF: Accepted Non-matching Frame

Acceptance of non-matching frames may be enabled via MCAN_GFC.ANFS and MCAN_GFC.ANFE.

0: Received frame matching filter index FIDX.

1: Received frame did not match any Rx filter element.

• R1 Bits 30:24 FIDX[6:0]: Filter Index

0-127: Index of matching Rx acceptance filter element (invalid if ANMF = ‘1’).
Range is 0 to MCAN_SIDFC.LSS - 1 resp. MCAN_XIDFC.LSE - 1.

• R1 Bit 21 FDF: FD Format

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

• R1 Bit 20 BRS: Bit Rate Switch

0: Frame received without bit rate switching.

© 2021 Microchip Technology Inc.
and its subsidiaries

1: Frame received with bit rate switching.

**Note:**
Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled (MCAN_CCCR.FDOE = 1). Bit BRS is only evaluated when in addition MCAN_CCCR.BRSE = 1.

• R1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: received frame has 0-8 data bytes.

9-15: CAN: received frame has 8 data bytes.

9-15: CAN FD: received frame has 12/16/20/24/32/48/64 data bytes.

• R1 Bits 15:0 RXTS[15:0]: Rx Timestamp

Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.

• R2 Bits 31:24 DB3[7:0]: Data Byte 3

• R2 Bits 23:16 DB2[7:0]: Data Byte 2

• R2 Bits 15:8 DB1[7:0]: Data Byte 1

• R2 Bits 7:0 DB0[7:0]: Data Byte 0

• R3 Bits 31:24 DB7[7:0]: Data Byte 7

• R3 Bits 23:16 DB6[7:0]: Data Byte 6

• R3 Bits 15:8 DB5[7:0]: Data Byte 5

• R3 Bits 7:0 DB4[7:0]: Data Byte 4

... ... ...

• Rn Bits 31:24 DBm[7:0]: Data Byte m

• Rn Bits 23:16 DBm-1[7:0]: Data Byte m-1

• Rn Bits 15:8 DBm-2[7:0]: Data Byte m-2

• Rn Bits 7:0 DBm-3[7:0]: Data Byte m-3

**Note:** Depending on the configuration of the element size (MCAN_RXESC), between two and sixteen 32-bit words (Rn = 3 ..17) are used for storage of a CAN message's data field.

### 49.5.7.3 Tx Buffer Element

The Tx Buffers section can be configured to hold dedicated Tx Buffers as well as a Tx FIFO / Tx Queue. In case that the Tx Buffers section is shared by dedicated Tx buffers and a Tx FIFO / Tx Queue, the dedicated Tx Buffers start at the beginning of the Tx Buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue.
The Tx Handler distinguishes between dedicated Tx Buffers and Tx FIFO / Tx Queue by evaluating the Tx Buffer configuration TXBC.TFQS and TXBC.NDTB. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register TXESC.

**Table 49-8. Tx Buffer Element**

| | 31 | | | 24 | 23 | | | | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| T0 | ESI | XTD | RTR | ID[28:0] | | | | | | | | | |
| T1 | MM[7:0] | | | | EFC | reserved | FDF | BRS | DLC[3:0] | reserved | | | |
| T2 | DB3[7:0] | | | | DB2[7:0] | | | | | DB1[7:0] | | DB0[7:0] | |
| T3 | DB7[7:0] | | | | DB6[7:0] | | | | | DB5[7:0] | | DB4[7:0] | |
| ... | ... | | | | ... | | | | | ... | | ... | |
| Tn | DBm[7:0] | | | | DBm-1[7:0] | | | | | DBm-2[7:0] | | DBm-3[7:0] | |

• T0 Bit 30 ESI: Error State Indicator

T0 Bit 31 ESI: Error State Indicator

0: ESI bit in CAN FD format depends only on error passive flag

1: ESI bit in CAN FD format transmitted recessive

**Note:** The ESI bit of the transmit buffer is or'ed with the error passive flag to decide the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, an error active node may optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive. This feature can be used in gateway applications when a message from an error passive node is routed to another CAN network.

• T0 Bit 30 XTD: Extended Identifier

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• T0 Bit 29 RTR: Remote Transmission Request

0: Transmit data frame.

1: Transmit remote frame.

**Note:** When RTR = 1, the MCAN transmits a remote frame according to ISO11898-1, even if MCAN_CCCR.FDOE enables the transmission in CAN FD format.

• T0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier has to be written to ID[28:18].

• T1 Bits 31:24 MM[7:0]: Message Marker

Written by processor during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.

• T1 Bit 23 EFC: Event FIFO Control

0: Do not store Tx events.

1: Store Tx events.

• T1 Bit 21 FDF: FD Format

0: Frame transmitted in Classic CAN format

1: Frame transmitted in CAN FD format

• T1 Bit 20 BRS: Bit Rate Switching

0: CAN FD frames transmitted without bit rate switching

1: CAN FD frames transmitted with bit rate switching

**Note:**
Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled (MCAN_CCCR.FDOE = 1). Bit BRS is only evaluated when in addition MCAN_CCCR.BRSE = 1.

• T1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: transmit frame has 0-8 data bytes.

9-15: CAN: transmit frame has 8 data bytes.

9-15: CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes.

• T2 Bits 31:24 DB3[7:0]: Data Byte 3

• T2 Bits 23:16 DB2[7:0]: Data Byte 2

• T2 Bits 15:8 DB1[7:0]: Data Byte 1

• T2 Bits 7:0 DB0[7:0]: Data Byte 0

• T3 Bits 31:24 DB7[7:0]: Data Byte 7

© 2021 Microchip Technology Inc.
and its subsidiaries

• T3 Bits 23:16 DB6[7:0]: Data Byte 6

• T3 Bits 15:8 DB5[7:0]: Data Byte 5

• T3 Bits 7:0 DB4[7:0]: Data Byte 4

... ... ...

• Tn Bits 31:24 DBm[7:0]: Data Byte m

• Tn Bits 23:16 DBm-1[7:0]: Data Byte m-1

• Tn Bits 15:8 DBm-2[7:0]: Data Byte m-2

• Tn Bits 7:0 DBm-3[7:0]: Data Byte m-3

**Note:** Depending on the configuration of the element size (MCAN_TXESC), between two and sixteen 32-bit words (Tn = 3 ..17) are used for storage of a CAN message's data field.

#### 49.5.7.4 Tx Event FIFO Element

Each element stores information about transmitted messages. By reading the Tx Event FIFO the processor gets this information in the order the messages were transmitted. Status information about the Tx Event FIFO can be obtained from register TXEFS.

**Table 49-9. Tx Event FIFO Element**

| | 31 | | | 24 | 23 | | | | | | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E0 | ESI | XTD | RTR | ID[28:0] | | | | | | | | | | | |
| E1 | MM[7:0] | | | | ET [1:0] | FDF | BRS | DLC[3:0] | | | | TXTS[15:0] | | | |

• E0 Bit 31 ESI: Error State Indicator

0: Transmitting node is error active.

1: Transmitting node is error passive.

• E0 Bit 30 XTD: Extended Identifier

0: 11-bit standard identifier.

1: 29-bit extended identifier.

• E0 Bit 29 RTR: Remote Transmission Request

0: Data frame transmitted.

1: Remote frame transmitted.

• E0 Bits 28:0 ID[28:0]: Identifier

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

• E1 Bits 31:24 MM[7:0]: Message Marker

Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.

• E1 Bit 23:22 ET[1:0]: Event Type

0: Reserved

1: Tx event

2: Transmission in spite of cancellation (always set for transmissions in DAR mode)

3: Reserved

• E1 Bit 21 FDF: FD Format

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

© 2021 Microchip Technology Inc.
and its subsidiaries

• E1 Bit 20 BRS: Bit Rate Switch

0: Frame transmitted without bit rate switching.

1: Frame transmitted with bit rate switching.

• E1 Bits 19:16 DLC[3:0]: Data Length Code

0-8: CAN + CAN FD: frame with 0-8 data bytes transmitted.

9-15: CAN: frame with 8 data bytes transmitted.

9-15: CAN FD: frame with 12/16/20/24/32/48/64 data bytes transmitted

• E1 Bits 15:0 TXTS[15:0]: Tx Timestamp

Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.

#### 49.5.7.5 Standard Message ID Filter Element

Up to 128 filter elements can be configured for 11-bit standard IDs. When accessing a Standard Message ID Filter element, its address is the Filter List Standard Start Address MCAN_SIDFC.FLSSA plus the index of the filter element (0…127).

**Table 49-10. Standard Message ID Filter Element**

| | 31 | | | | | 24 | 23 | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S0 | SFT[1:0] | | SFEC [2:0] | | | SFID1[10:0] | | | – | SFID2[10:0] | | |

• Bits 31:30 SFT[1:0]: Standard Filter Type

0: Range filter from SF1ID to SF2ID (SF2ID ≥ SF1ID)

1: Dual ID filter for SF1ID or SF2ID

2: Classic filter: SF1ID = filter, SF2ID = mask

3: Reserved

• Bit 29:27 SFEC[2:0]: Standard Filter Element Configuration

All enabled filter elements are used for acceptance filtering of standard frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = “100”, “101”, or “110” a match sets interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case register HPMS is updated with the status of the priority match.

0: Disable filter element

1: Store in Rx FIFO 0 if filter matches

2: Store in Rx FIFO 1 if filter matches

3: Reject ID if filter matches

4: Set priority if filter matches

5: Set priority and store in FIFO 0 if filter matches

6: Set priority and store in FIFO 1 if filter matches

7: Store into Rx Buffer or as debug message, configuration of SFT[1:0] ignored

• Bits 26:16 SFID1[10:0]: Standard Filter ID 1

First ID of standard ID filter element.

When filtering for Rx Buffers or for debug messages this field defines the ID of a standard message to be stored. The received identifiers must match exactly, no masking mechanism is used.

• Bits 10:0 SFID2[10:0]: Standard Filter ID 2

© 2021 Microchip Technology Inc. and its subsidiaries

This field has a different meaning depending on the configuration of SFEC:

- SFEC = “001”...“110”–Second ID of standard ID filter element
- SFEC = “111”–Filter for Rx Buffers or for debug messages

SFID2[10:9] decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence.

0: Store message in a Rx buffer

1: Debug Message A

2: Debug Message B

3: Debug Message C

SFID2[5:0] defines the index of the dedicated Rx Buffer element to which a matching message is stored.

### 49.5.7.6 Extended Message ID Filter Element

Up to 64 filter elements can be configured for 29-bit extended IDs. When accessing an Extended Message ID Filter element, its address is the Filter List Extended Start Address MCAN_XIDFC.FLESA plus two times the index of the filter element (0…63).

**Table 49-11. Extended Message ID Filter Element**

| | 31 | | | 24 | 23 | 16 | 15 | 8 | 7 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| F0 | EFEC [2:0] | | | EFID1[28:0] | | | | | | |
| F1 | EFT[1:0] | | – | EFID2[28:0] | | | | | | |

• F0 Bit 31:29 EFEC[2:0]: Extended Filter Element Configuration

All enabled filter elements are used for acceptance filtering of extended frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = “100”, “101”, or “110”, a match sets the interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case, register MCAN_HPMS is updated with the status of the priority match.

0: Disable filter element

1: Store in Rx FIFO 0 if filter matches

2: Store in Rx FIFO 1 if filter matches

3: Reject ID if filter matches

4: Set priority if filter matches

5: Set priority and store in FIFO 0 if filter matches

6: Set priority and store in FIFO 1 if filter matches

7: Store into Rx Buffer or as debug message, configuration of EFT[1:0] ignored

• F0 Bits 28:0 EFID1[28:0]: Extended Filter ID 1

First ID of extended ID filter element.

When filtering for Rx Buffers or for debug messages this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only MCAN_XIDAM masking mechanism (see Extended Message ID Filtering) is used.

• F1 Bits 31:30 EFT[1:0]: Extended Filter Type

0: Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID)

1: Dual ID filter for EF1ID or EF2ID

2: Classic filter: EF1ID = filter, EF2ID = mask

3: Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID), MCAN_XIDAM mask not applied

© 2021 Microchip Technology Inc. and its subsidiaries

• F1 Bits 28:0 EFID2[28:0]: Extended Filter ID 2

This field has a different meaning depending on the configuration of EFEC:

• EFEC = “001”...“110”–Second ID of extended ID filter element

• EFEC = “111”–Filter for Rx Buffers or for debug messages

EFID2[10:9] decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence.

0: Store message in an Rx buffer

1: Debug Message A

2: Debug Message B

3: Debug Message C

EFID2[5:0] defines the index of the dedicated Rx Buffer element to which a matching message is stored.

### 49.5.8 Hardware Reset Description

After hardware reset, the registers of the MCAN hold the reset values listed in the register descriptions. Additionally the Bus_Off state is reset and the output CANTX is set to recessive (HIGH). The value 0x0001 (MCAN_CCCR.INIT = ‘1’) in the CC Control register enables software initialization. The MCAN does not influence the CAN bus until the processor resets MCAN_CCCR.INIT to ‘0’.

### 49.5.9 Access to Reserved Register Addresses

In case the application software accesses one of the reserved addresses in the MCAN register map (read or write access), interrupt flag MCAN_IR.ARA is set and, if enabled, the selected interrupt line is risen.

© 2021 Microchip Technology Inc.
and its subsidiaries

## 49.6 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | MCAN_CREL | 7:0 | DAY[7:0] | | | | | | | |
| | | 15:8 | MON[7:0] | | | | | | | |
| | | 23:16 | SUBSTEP[3:0] | | | | YEAR[3:0] | | | |
| | | 31:24 | REL[3:0] | | | | STEP[3:0] | | | |
| 0x04 | MCAN_ENDN | 7:0 | ETV[7:0] | | | | | | | |
| | | 15:8 | ETV[15:8] | | | | | | | |
| | | 23:16 | ETV[23:16] | | | | | | | |
| | | 31:24 | ETV[31:24] | | | | | | | |
| 0x08 | MCAN_CUST | 7:0 | CSV[7:0] | | | | | | | |
| | | 15:8 | CSV[15:8] | | | | | | | |
| | | 23:16 | CSV[23:16] | | | | | | | |
| | | 31:24 | CSV[31:24] | | | | | | | |
| 0x0C | MCAN_DBTP | 7:0 | DTSEG2[3:0] | | | | | DSJW[2:0] | | |
| | | 15:8 | | | | DTSEG1[4:0] | | | | |
| | | 23:16 | TDC | | | DBRP[4:0] | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | MCAN_TEST | 7:0 | RX | TX[1:0] | | LBCK | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | MCAN_RWD | 7:0 | WDC[7:0] | | | | | | | |
| | | 15:8 | WDV[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | MCAN_CCCR | 7:0 | TEST | DAR | MON | CSR | CSA | ASM | CCE | INIT |
| | | 15:8 | NISO | TXP | EFBI | PXHD | | | BRSE | FDOE |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | MCAN_NBTP | 7:0 | | NTSEG2[6:0] | | | | | | |
| | | 15:8 | NTSEG1[7:0] | | | | | | | |
| | | 23:16 | NBRP[7:0] | | | | | | | |
| | | 31:24 | NSJW[6:0] | | | | | | | NBRP[8] |
| 0x20 | MCAN_TSCC | 7:0 | | | | | | | TSS[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | TCP[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x24 | MCAN_TSCV | 7:0 | TSC[7:0] | | | | | | | |
| | | 15:8 | TSC[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | MCAN_TOCC | 7:0 | | | | | | TOS[1:0] | | ETOC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | TOP[7:0] | | | | | | | |
| | | 31:24 | TOP[15:8] | | | | | | | |
| 0x2C | MCAN_TOCV | 7:0 | TOC[7:0] | | | | | | | |
| | | 15:8 | TOC[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | MCAN_ECR | 7:0 | TEC[7:0] | | | | | | | |
| | | 15:8 | RP | REC[6:0] | | | | | | |
| | | 23:16 | CEL[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x44 | MCAN_PSR | 7:0 | BO | EW | EP | ACT[1:0] | | | LEC[2:0] | |
| | | 15:8 | | PXE | RFDF | RBRS | RESI | | DLEC[2:0] | |
| | | 23:16 | | TDCV[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x48 | MCAN_TDCR | 7:0 | | TDCF[6:0] | | | | | | |
| | | 15:8 | | TDCO[6:0] | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | MCAN_IR | 7:0 | RF1L | RF1F | RF1W | RF1N | RF0L | RF0F | RF0W | RF0N |
| | | 15:8 | TEFL | TEFF | TEFW | TEFN | TFE | TCF | TC | HPM |
| | | 23:16 | EP | ELO | | | DRX | TOO | MRAF | TSW |
| | | 31:24 | | | ARA | PED | PEA | WDI | BO | EW |
| 0x54 | MCAN_IE | 7:0 | RF1LE | RF1FE | RF1WE | RF1NE | RF0LE | RF0FE | RF0WE | RF0NE |
| | | 15:8 | TEFLE | TEFFE | TEFWE | TEFNE | TFEE | TCFE | TCE | HPME |
| | | 23:16 | EPE | ELOE | | | DRXE | TOOE | MRAFE | TSWE |
| | | 31:24 | | | ARAE | PEDE | PEAE | WDIE | BOE | EWE |
| 0x58 | MCAN_ILS | 7:0 | RF1LL | RF1FL | RF1WL | RF1NL | RF0LL | RF0FL | RF0WL | RF0NL |
| | | 15:8 | TEFLL | TEFFL | TEFWL | TEFNL | TFEL | TCFL | TCL | HPML |
| | | 23:16 | EPL | ELOL | | | DRXL | TOOL | MRAFL | TSWL |
| | | 31:24 | | | ARAL | PEDL | PEAL | WDIL | BOL | EWL |
| 0x5C | MCAN_ILE | 7:0 | | | | | | | EINT1 | EINT0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x60 ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | MCAN_GFC | 7:0 | | | ANFS[1:0] | | ANFE[1:0] | | RRFS | RRFE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 | MCAN_SIDFC | 7:0 | FLSSA[5:0] | | | | | | | |
| | | 15:8 | FLSSA[13:6] | | | | | | | |
| | | 23:16 | LSS[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x88 | MCAN_XIDFC | 7:0 | FLESA[5:0] | | | | | | | |
| | | 15:8 | FLESA[13:6] | | | | | | | |
| | | 23:16 | | LSE[6:0] | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x8C ... 0x8F | Reserved | | | | | | | | | |
| 0x90 | MCAN_XIDAM | 7:0 | EIDM[7:0] | | | | | | | |
| | | 15:8 | EIDM[15:8] | | | | | | | |
| | | 23:16 | EIDM[23:16] | | | | | | | |
| | | 31:24 | | | | EIDM[28:24] | | | | |
| 0x94 | MCAN_HPMS | 7:0 | MSI[1:0] | | BIDX[5:0] | | | | | |
| | | 15:8 | FLST | FIDX[6:0] | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 | MCAN_NDAT1 | 7:0 | ND7 | ND6 | ND5 | ND4 | ND3 | ND2 | ND1 | ND0 |
| | | 15:8 | ND15 | ND14 | ND13 | ND12 | ND11 | ND10 | ND9 | ND8 |
| | | 23:16 | ND23 | ND22 | ND21 | ND20 | ND19 | ND18 | ND17 | ND16 |
| | | 31:24 | ND31 | ND30 | ND29 | ND28 | ND27 | ND26 | ND25 | ND24 |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x9C | MCAN_NDAT2 | 7:0 | ND39 | ND38 | ND37 | ND36 | ND35 | ND34 | ND33 | ND32 |
| | | 15:8 | ND47 | ND46 | ND45 | ND44 | ND43 | ND42 | ND41 | ND40 |
| | | 23:16 | ND55 | ND54 | ND53 | ND52 | ND51 | ND50 | ND49 | ND48 |
| | | 31:24 | ND63 | ND62 | ND61 | ND60 | ND59 | ND58 | ND57 | ND56 |
| 0xA0 | MCAN_RXF0C | 7:0 | F0SA[5:0] | | | | | | | |
| | | 15:8 | F0SA[13:6] | | | | | | | |
| | | 23:16 | | F0S[6:0] | | | | | | |
| | | 31:24 | F0OM | F0WM[6:0] | | | | | | |
| 0xA4 | MCAN_RXF0S | 7:0 | | F0FL[6:0] | | | | | | |
| | | 15:8 | | | F0GI[5:0] | | | | | |
| | | 23:16 | | | F0PI[5:0] | | | | | |
| | | 31:24 | | | | | | | RF0L | F0F |
| 0xA8 | MCAN_RXF0A | 7:0 | | | F0AI[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xAC | MCAN_RXBC | 7:0 | RBSA[5:0] | | | | | | | |
| | | 15:8 | RBSA[13:6] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB0 | MCAN_RXF1C | 7:0 | F1SA[5:0] | | | | | | | |
| | | 15:8 | F1SA[13:6] | | | | | | | |
| | | 23:16 | | F1S[6:0] | | | | | | |
| | | 31:24 | F1OM | F1WM[6:0] | | | | | | |
| 0xB4 | MCAN_RXF1S | 7:0 | | F1FL[6:0] | | | | | | |
| | | 15:8 | | | F1GI[5:0] | | | | | |
| | | 23:16 | | | F1PI[5:0] | | | | | |
| | | 31:24 | DMS[1:0] | | | | | | RF1L | F1F |
| 0xB8 | MCAN_RXF1A | 7:0 | | | F1AI[5:0] | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xBC | MCAN_RXESC | 7:0 | | F1DS[2:0] | | | | F0DS[2:0] | | |
| | | 15:8 | | | | | | RBDS[2:0] | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC0 | MCAN_TXBC | 7:0 | TBSA[5:0] | | | | | | | |
| | | 15:8 | TBSA[13:6] | | | | | | | |
| | | 23:16 | | | NDTB[5:0] | | | | | |
| | | 31:24 | | TFQM | TFQS[5:0] | | | | | |
| 0xC4 | MCAN_TXFQS | 7:0 | | | TFFL[5:0] | | | | | |
| | | 15:8 | | | | TFGI[4:0] | | | | |
| | | 23:16 | | | TFQF | TFQPI[4:0] | | | | |
| | | 31:24 | | | | | | | | |
| 0xC8 | MCAN_TXESC | 7:0 | | | | | | TBDS[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xCC | MCAN_TXBRP | 7:0 | TRP7 | TRP6 | TRP5 | TRP4 | TRP3 | TRP2 | TRP1 | TRP0 |
| | | 15:8 | TRP15 | TRP14 | TRP13 | TRP12 | TRP11 | TRP10 | TRP9 | TRP8 |
| | | 23:16 | TRP23 | TRP22 | TRP21 | TRP20 | TRP19 | TRP18 | TRP17 | TRP16 |
| | | 31:24 | TRP31 | TRP30 | TRP29 | TRP28 | TRP27 | TRP26 | TRP25 | TRP24 |
| 0xD0 | MCAN_TXBAR | 7:0 | AR7 | AR6 | AR5 | AR4 | AR3 | AR2 | AR1 | AR0 |
| | | 15:8 | AR15 | AR14 | AR13 | AR12 | AR11 | AR10 | AR9 | AR8 |
| | | 23:16 | AR23 | AR22 | AR21 | AR20 | AR19 | AR18 | AR17 | AR16 |
| | | 31:24 | AR31 | AR30 | AR29 | AR28 | AR27 | AR26 | AR25 | AR24 |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xD4 | MCAN_TXBCR | 7:0 | CR7 | CR6 | CR5 | CR4 | CR3 | CR2 | CR1 | CR0 |
| | | 15:8 | CR15 | CR14 | CR13 | CR12 | CR11 | CR10 | CR9 | CR8 |
| | | 23:16 | CR23 | CR22 | CR21 | CR20 | CR19 | CR18 | CR17 | CR16 |
| | | 31:24 | CR31 | CR30 | CR29 | CR28 | CR27 | CR26 | CR25 | CR24 |
| 0xD8 | MCAN_TXBTO | 7:0 | TO7 | TO6 | TO5 | TO4 | TO3 | TO2 | TO1 | TO0 |
| | | 15:8 | TO15 | TO14 | TO13 | TO12 | TO11 | TO10 | TO9 | TO8 |
| | | 23:16 | TO23 | TO22 | TO21 | TO20 | TO19 | TO18 | TO17 | TO16 |
| | | 31:24 | TO31 | TO30 | TO29 | TO28 | TO27 | TO26 | TO25 | TO24 |
| 0xDC | MCAN_TXBCF | 7:0 | CF7 | CF6 | CF5 | CF4 | CF3 | CF2 | CF1 | CF0 |
| | | 15:8 | CF15 | CF14 | CF13 | CF12 | CF11 | CF10 | CF9 | CF8 |
| | | 23:16 | CF23 | CF22 | CF21 | CF20 | CF19 | CF18 | CF17 | CF16 |
| | | 31:24 | CF31 | CF30 | CF29 | CF28 | CF27 | CF26 | CF25 | CF24 |
| 0xE0 | MCAN_TXBTIE | 7:0 | TIE7 | TIE6 | TIE5 | TIE4 | TIE3 | TIE2 | TIE1 | TIE0 |
| | | 15:8 | TIE15 | TIE14 | TIE13 | TIE12 | TIE11 | TIE10 | TIE9 | TIE8 |
| | | 23:16 | TIE23 | TIE22 | TIE21 | TIE20 | TIE19 | TIE18 | TIE17 | TIE16 |
| | | 31:24 | TIE31 | TIE30 | TIE29 | TIE28 | TIE27 | TIE26 | TIE25 | TIE24 |
| 0xE4 | MCAN_TXBCIE | 7:0 | CFIE7 | CFIE6 | CFIE5 | CFIE4 | CFIE3 | CFIE2 | CFIE1 | CFIE0 |
| | | 15:8 | CFIE15 | CFIE14 | CFIE13 | CFIE12 | CFIE11 | CFIE10 | CFIE9 | CFIE8 |
| | | 23:16 | CFIE23 | CFIE22 | CFIE21 | CFIE20 | CFIE19 | CFIE18 | CFIE17 | CFIE16 |
| | | 31:24 | CFIE31 | CFIE30 | CFIE29 | CFIE28 | CFIE27 | CFIE26 | CFIE25 | CFIE24 |
| 0xE8 ... 0xEF | Reserved | | | | | | | | | |
| 0xF0 | MCAN_TXEFC | 7:0 | EFSA[5:0] | | | | | | | |
| | | 15:8 | EFSA[13:6] | | | | | | | |
| | | 23:16 | | | EFS[5:0] | | | | | |
| | | 31:24 | | | EFWM[5:0] | | | | | |
| 0xF4 | MCAN_TXEFS | 7:0 | | | EFFL[5:0] | | | | | |
| | | 15:8 | | | | EFGI[4:0] | | | | |
| | | 23:16 | | | | EFPI[4:0] | | | | |
| | | 31:24 | | | | | | | TEFL | EFF |
| 0xF8 | MCAN_TXEFA | 7:0 | | | | EFAI[4:0] | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 49.6.1 MCAN Core Release Register

**Name:** MCAN_CREL
**Offset:** 0x00
**Reset:** 0x32150320
**Property:** Read-only

Due to clock domain crossing, there is a delay between when a register bit or field is written and when the related status register bits are updated.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | REL[3:0] | | | | STEP[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SUBSTEP[3:0] | | | | YEAR[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MON[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DAY[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | x | x | x | x | x | x | x | x |

**Bits 31:28 – REL[3:0]** Core Release
One digit, BCD-coded.

**Bits 27:24 – STEP[3:0]** Step of Core Release
One digit, BCD-coded.

**Bits 23:20 – SUBSTEP[3:0]** Sub-step of Core Release
One digit, BCD-coded.

**Bits 19:16 – YEAR[3:0]** Timestamp Year
One digit, BCD-coded. This field is set by generic parameter on MCAN synthesis.

**Bits 15:8 – MON[7:0]** Timestamp Month
Two digits, BCD-coded. This field is set by generic parameter on MCAN synthesis.

**Bits 7:0 – DAY[7:0]** Timestamp Day
Two digits, BCD-coded. This field is set by generic parameter on MCAN synthesis.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.2 MCAN Endian Register

**Name:** MCAN_ENDN
**Offset:** 0x04
**Reset:** 0x87654321
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ETV[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ETV[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ETV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |

**Bits 31:0 – ETV[31:0]** Endianness Test Value
The endianness test value is 0x87654321.

### 49.6.3 MCAN Customer Register

**Name:** MCAN_CUST
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CSV[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CSV[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CSV[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CSV[31:0]** Customer-specific Value
Customer-specific value.

### 49.6.4 MCAN Data Bit Timing and Prescaler Register

**Name:** MCAN_DBTP
**Offset:** 0x0C
**Reset:** 0x00000A33
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

The CAN bit time may be programmed in the range of 4 to 25 time quanta. The CAN time quantum may be programmed in the range of 1 to 32 CAN core clock periods. $t_q$ = (DBRP + 1) CAN core clock periods.

DTSEG1 is the sum of Prop_Seg and Phase_Seg1. DTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [DTSEG1 + DTSEG2 + 3] $t_q$
or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] $t_q$.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

With a CAN core clock frequency of 8 MHz, the reset value of 0x00000A33 configures the MCAN for a fast bit rate of 500 kbit/s.

The bit rate configured for the CAN FD data phase via MCAN_DBTP must be higher than or equal to the bit rate configured for the arbitration phase via MCAN_NBTP.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TDC | | | DBRP[4:0] | | | | |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | DTSEG1[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 1 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DTSEG2[3:0] | | | | | DSJW[2:0] | | |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 1 | 1 | | 0 | 1 | 1 |

**Bit 23 – TDC** Transmitter Delay Compensation
0 (DISABLED): Transmitter Delay Compensation disabled.
1 (ENABLED): Transmitter Delay Compensation enabled.

**Bits 20:16 – DBRP[4:0]** Data Bit Rate Prescaler
The value by which the peripheral clock is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

**Bits 12:8 – DTSEG1[4:0]** Data Time Segment Before Sample Point
0: Forbidden.
1 to 31: The duration of time segment is $t_q$ x (DTSEG1 + 1).

**Bits 7:4 – DTSEG2[3:0]** Data Time Segment After Sample Point
The duration of time segment is $t_q$ x (DTSEG2 + 1).

**Bits 2:0 – DSJW[2:0]** Data (Re) Synchronization Jump Width
The duration of a synchronization jump is $t_q$ x (DSJW + 1).

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.5 MCAN Test Register

**Name:** MCAN_TEST
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

Write access to the Test Register has to be enabled by setting bit MCAN_CCCR.TEST to ‘1’.

All MCAN Test Register functions are set to their reset values when bit MCAN_CCCR.TEST is cleared.

Loop Back mode and software control of pin CANTX are hardware test modes. Programming of TX ≠ 0 disturbs the message transfer on the CAN bus.

The reset value for MCAN_TEST.RX is undefined.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RX | TX[1:0] | | LBCK | | | | |
| Access | R | R/W | R/W | R/W | | | | |
| Reset | x | 0 | 0 | 0 | | | | |

**Bit 7 – RX** Receive Pin (read-only)
Monitors the actual value of pin CANRX.
The reset value for this bit is undefined.

| Value | Description |
|---|---|
| 0 | The CAN bus is dominant (CANRX = ‘0’). |
| 1 | The CAN bus is recessive (CANRX = ‘1’). |

**Bits 6:5 – TX[1:0]** Control of Transmit Pin (read/write)

| Value | Name | Description |
|---|---|---|
| 0 | RESET | Reset value, CANTX controlled by the CAN Core, updated at the end of the CAN bit time. |
| 1 | SAMPLE_POINT_MONITORING | Sample Point can be monitored at pin CANTX. |
| 2 | DOMINANT | Dominant (‘0’) level at pin CANTX. |
| 3 | RECESSIVE | Recessive (‘1’) at pin CANTX. |

**Bit 4 – LBCK** Loop Back Mode (read/write)
0 (DISABLED): Reset value. Loop Back mode is disabled.
1 (ENABLED): Loop Back mode is enabled (see Test Modes).

### 49.6.6 MCAN RAM Watchdog Register

**Name:** MCAN_RWD
**Offset:** 0x14
**Reset:** 0x00000000
**Property:** Read/Write

The RAM Watchdog monitors the Message RAM response time. A Message RAM access via the MCAN's Generic Master Interface starts the Message RAM Watchdog Counter with the value configured by MCAN_RWD.WDC. The counter is reloaded with MCAN_RWD.WDC when the Message RAM signals successful completion by activating its READY output. In case there is no response from the Message RAM until the counter has counted down to zero, the counter stops and interrupt flag MCAN_IR.WDI is set. The RAM Watchdog Counter is clocked by the system bus clock (peripheral clock).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WDV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WDC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:8 – WDV[7:0]** Watchdog Value (read-only)
Watchdog Counter Value for the current message located in RAM.

**Bits 7:0 – WDC[7:0]** Watchdog Configuration (read/write)
Start value of the Message RAM Watchdog Counter. The counter is disabled when WDC is cleared.

### 49.6.7 MCAN CC Control Register

**Name:** MCAN_CCCR
**Offset:** 0x18
**Reset:** 0x00000001
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NISO | TXP | EFBI | PXHD | | | BRSE | FDOE |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TEST | DAR | MON | CSR | CSA | ASM | CCE | INIT |
| Access | R/W | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Bit 15 – NISO** Non-ISO Operation
If this bit is set, the MCAN uses the CAN FD frame format as specified by the Bosch CAN FD Specification V1.0.

| Value | Description |
|---|---|
| 0 | CAN FD frame format according to ISO11898-1 (default). |
| 1 | CAN FD frame format according to Bosch CAN FD Specification V1.0. |

**Bit 14 – TXP** Transmit Pause (read/write, write protection)
If this bit is set, the MCAN pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame (see Tx Handling).

| Value | Description |
|---|---|
| 0 | Transmit pause disabled. |
| 1 | Transmit pause enabled. |

**Bit 13 – EFBI** Edge Filtering during Bus Integration (read/write, write protection)

| Value | Description |
|---|---|
| 0 | Edge filtering is disabled. |
| 1 | Edge filtering is enabled. Two consecutive dominant tq required to detect an edge for hard synchronization. |

**Bit 12 – PXHD** Protocol Exception Event Handling (read/write, write protection)

| Value | Description |
|---|---|
| 0 | Protocol exception handling enabled. |
| 1 | Protocol exception handling disabled. |

**Bit 9 – BRSE** Bit Rate Switching Enable (read/write, write protection)
0 (DISABLED): Bit rate switching for transmissions disabled.
1 (ENABLED): Bit rate switching for transmissions enabled.

**Bit 8 – FDOE** CAN FD Operation Enable (read/write, write protection)
0 (DISABLED): FD operation disabled.
1 (ENABLED): FD operation enabled.

**Bit 7 – TEST** Test Mode Enable (read/write, write protection against '1')
0 (DISABLED): Normal operation, MCAN_TEST register holds reset values.
1 (ENABLED): Test mode, write access to MCAN_TEST register enabled.

**Bit 6 – DAR** Disable Automatic Retransmission (read/write, write protection)
0 (AUTO_RETX): Automatic retransmission of messages not transmitted successfully enabled.
1 (NO_AUTO_RETX): Automatic retransmission disabled.

**Bit 5 – MON** Bus Monitoring Mode (read/write, write protection against '1')
0 (DISABLED): Bus Monitoring mode is disabled.
1 (ENABLED): Bus Monitoring mode is enabled.

**Bit 4 – CSR** Clock Stop Request (read/write)
0 (NO_CLOCK_STOP): No clock stop is requested.
1 (CLOCK_STOP): Clock stop requested. When clock stop is requested, first INIT and then CSA will be set after all pending transfer requests have been completed and the CAN bus reached idle.

**Bit 3 – CSA** Clock Stop Acknowledge (read-only)

| Value | Description |
|---|---|
| 0 | No clock stop acknowledged. |
| 1 | MCAN may be set in power down by stopping the peripheral clock and the CAN core clock. |

**Bit 2 – ASM** Restricted Operation Mode (read/write, write protection against '1')
For a description of the Restricted Operation mode see Restricted Operation Mode.
0 (NORMAL): Normal CAN operation.
1 (RESTRICTED): Restricted Operation mode active.

**Bit 1 – CCE** Configuration Change Enable (read/write, write protection)
0 (PROTECTED): The processor has no write access to the protected configuration registers.
1 (CONFIGURABLE): The processor has write access to the protected configuration registers (while MCAN_CCCR.INIT = '1').

**Bit 0 – INIT** Initialization (read/write)
Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to ensure that the previous value written to INIT has been accepted by reading INIT before setting INIT to a new value.
0 (DISABLED): Normal operation.
1 (ENABLED): Initialization is started.

© 2021 Microchip Technology Inc. and its subsidiaries

### 49.6.8 MCAN Nominal Bit Timing and Prescaler Register

**Name:** MCAN_NBTP
**Offset:** 0x1C
**Reset:** 0x06000A03
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN_CCCR.

The CAN bit time may be programmed in the range of 4 to 385 time quanta. The CAN time quantum may be programmed in the range of 1 to 512 CAN core clock periods. $t_q = t_{core\ clock}$ x (NBRP + 1).

NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [NTSEG1 + NTSEG2 + 3] $t_q$
or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] $t_q$.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

With a CAN core clock frequency of 8 MHz, the reset value of 0x06000A03 configures the MCAN for a bit rate of 500 kbit/s.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NSJW[6:0] | | | | | | | NBRP[8] |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NBRP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NTSEG1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | NTSEG2[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

**Bits 31:25 – NSJW[6:0]** Nominal (Re) Synchronization Jump Width
0 to 127: The duration of a synchronization jump is $t_q$ x (NSJW + 1).

**Bits 24:16 – NBRP[8:0]** Nominal Bit Rate Prescaler
0 to 511: The value by which the oscillator frequency is divided for generating the CAN time quanta. The CAN time is built up from a multiple of this quanta. CAN time quantum (tq) = $t_{core\ clock}$ x (NBRP + 1)

**Bits 15:8 – NTSEG1[7:0]** Nominal Time Segment Before Sample Point

| Value | Description |
|---|---|
| 0 | Reserved; do not use. |
| 1 to 255 | The duration of time segment is $t_q$ x (NTSEG1 + 1). |

**Bits 6:0 – NTSEG2[6:0]** Nominal Time Segment After Sample Point

| Value | Description |
|---|---|
| 0 | Reserved; do not use. |
| 1 to 127 | The duration of time segment is $t_q$ x (NTSEG2 + 1). |

### 49.6.9 MCAN Timestamp Counter Configuration Register

**Name:** MCAN_TSCC
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read/Write

For a description of the Timestamp Counter see Timestamp Generation.

With CAN FD, an external counter is required for timestamp generation (TSS = 2).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | TCP[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | TSS[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 19:16 – TCP[3:0]** Timestamp Counter Prescaler
Configures the timestamp and timeout counters time unit in multiples of CAN bit times [ 1…16 ]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

**Bits 1:0 – TSS[1:0]** Timestamp Select

| Value | Name | Description |
|---|---|---|
| 0 | ALWAYS_0 | Timestamp counter value always 0x0000 |
| 1 | TCP_INC | Timestamp counter value incremented according to TCP |
| 2 | EXT_TIMESTAMP | External timestamp counter value used |
| 3 | ALWAYS_0 | Timestamp counter value always 0x0000 |

### 49.6.10 MCAN Timestamp Counter Value Register

**Name:** MCAN_TSCV
**Offset:** 0x24
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TSC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TSC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – TSC[15:0]** Timestamp Counter (cleared on write)
The internal/external Timestamp Counter value is captured on start of frame (both Receive and Transmit). When MCAN_TSCC.TSS = 1, the Timestamp Counter is incremented in multiples of CAN bit times [ 1…16 ] depending on the configuration of MCAN_TSCC.TCP. A wrap around sets interrupt flag MCAN_IR.TSW. Write access resets the counter to zero.
When MCAN_TSCC.TSS = 2, TSC reflects the external Timestamp Counter value. Thus a write access has no impact.
**Note:** A “wrap around” is a change of the Timestamp Counter value from non-zero to zero not caused by write access to MCAN_TSCV.

### 49.6.11 MCAN Timeout Counter Configuration Register

**Name:** MCAN_TOCC
**Offset:** 0x28
**Reset:** 0xFFFF0000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

For a description of the Timeout Counter, see Timeout Counter.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TOP[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TOP[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | TOS[1:0] | | ETOC |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 31:16 – TOP[15:0]** Timeout Period
Start value of the Timeout Counter (down-counter). Configures the Timeout Period.

**Bits 2:1 – TOS[1:0]** Timeout Select
When operating in Continuous mode, a write to MCAN_TOCV presets the counter to the value configured by MCAN_TOCC.TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by MCAN_TOCC.TOP. Down-counting is started when the first FIFO element is stored.

| Value | Name | Description |
|---|---|---|
| 0 | CONTINUOUS | Continuous operation. |
| 1 | TX_EV_TIMEOUT | Timeout controlled by Tx Event FIFO. |
| 2 | RX0_EV_TIMEOUT | Timeout controlled by Receive FIFO 0. |
| 3 | RX1_EV_TIMEOUT | Timeout controlled by Receive FIFO 1. |

**Bit 0 – ETOC** Enable Timeout Counter
0 (NO_TIMEOUT): Timeout Counter disabled.
1 (TOS_CONTROLLED): Timeout Counter enabled.
For use of timeout function with CAN FD, see Timeout Counter.

### 49.6.12 MCAN Timeout Counter Value Register

**Name:** MCAN_TOCV
**Offset:** 0x2C
**Reset:** 0x0000FFFF
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TOC[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TOC[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 15:0 – TOC[15:0]** Timeout Counter (cleared on write)
The Timeout Counter is decremented in multiples of CAN bit times [ 1…16 ] depending on the configuration of MCAN_TSCC.TCP. When decremented to zero, interrupt flag MCAN_IR.TOO is set and the Timeout Counter is stopped. Start and reset/restart conditions are configured via MCAN_TOCC.TOS.

### 49.6.13 MCAN Error Counter Register

**Name:** MCAN_ECR
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read-only

When MCAN_CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CEL[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RP | REC[6:0] | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TEC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – CEL[7:0]** CAN Error Logging (cleared on read)
The counter is incremented each time when a CAN protocol error causes the Transmit Error Counter or the Receive Error Counter to be incremented. It is reset by read access to CEL. The counter stops at 0xFF; the next increment of TEC or REC sets interrupt flag IR.ELO.

**Bit 15 – RP** Receive Error Passive

| Value | Description |
|---|---|
| 0 | The Receive Error Counter is below the error passive level of 128. |
| 1 | The Receive Error Counter has reached the error passive level of 128. |

**Bits 14:8 – REC[6:0]** Receive Error Counter
Actual state of the Receive Error Counter, values between 0 and 127.

**Bits 7:0 – TEC[7:0]** Transmit Error Counter
Actual state of the Transmit Error Counter, values between 0 and 255.

### 49.6.14 MCAN Protocol Status Register

**Name:** MCAN_PSR
**Offset:** 0x44
**Reset:** 0x00000707
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | TDCV[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | PXE | RFDF | RBRS | RESI | DLEC[2:0] | | |
| Access | | | | | | | | |
| Reset | | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BO | EW | EP | ACT[1:0] | | LEC[2:0] | | |
| Access | R | R | R | R | R | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

**Bits 22:16 – TDCV[6:0]** Transmitter Delay Compensation Value
0 to 127: Position of the secondary sample point, in CAN core clock periods, defined by the sum of the measured delay from CANTX to CANRX and MCAN_TDCR.TDCO.

**Bit 14 – PXE** Protocol Exception Event (cleared on read)

| Value | Description |
|---|---|
| 0 | No protocol exception event occurred since last read access |
| 1 | Protocol exception event occurred |

**Bit 13 – RFDF** Received a CAN FD Message (cleared on read)
This bit is set independently from acceptance filtering.

| Value | Description |
|---|---|
| 0 | Since this bit was reset by the CPU, no CAN FD message has been received |
| 1 | Message in CAN FD format with FDF flag set has been received |

**Bit 12 – RBRS** BRS Flag of Last Received CAN FD Message (cleared on read)
This bit is set together with RFDF, independently from acceptance filtering.

| Value | Description |
|---|---|
| 0 | Last received CAN FD message did not have its BRS flag set. |
| 1 | Last received CAN FD message had its BRS flag set. |

**Bit 11 – RESI** ESI Flag of Last Received CAN FD Message (cleared on read)
This bit is set together with RFDF, independently from acceptance filtering.

| Value | Description |
|---|---|
| 0 | Last received CAN FD message did not have its ESI flag set. |
| 1 | Last received CAN FD message had its ESI flag set. |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bits 10:8 – DLEC[2:0]** Data Phase Last Error Code (set to 111 on read)
Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field will be cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error.

**Bit 7 – BO** Bus_Off Status

| Value | Description |
|---|---|
| 0 | The MCAN is not Bus_Off. |
| 1 | The MCAN is in Bus_Off state. |

**Bit 6 – EW** Warning Status

| Value | Description |
|---|---|
| 0 | Both error counters are below the Error_Warning limit of 96. |
| 1 | At least one of error counter has reached the Error_Warning limit of 96. |

**Bit 5 – EP** Error Passive

| Value | Description |
|---|---|
| 0 | The MCAN is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected. |
| 1 | The MCAN is in the Error_Passive state. |

**Bits 4:3 – ACT[1:0]** Activity
Monitors the CAN communication state of the CAN module.

| Value | Name | Description |
|---|---|---|
| 0 | SYNCHRONIZING | Node is synchronizing on CAN communication |
| 1 | IDLE | Node is neither receiver nor transmitter |
| 2 | RECEIVER | Node is operating as receiver |
| 3 | TRANSMITTER | Node is operating as transmitter |

**Bits 2:0 – LEC[2:0]** Last Error Code (set to 111 on read)
The LEC indicates the type of the last error to occur on the CAN bus. This field is cleared when a message has been transferred (reception or transmission) without error.

| Value | Name | Description |
|---|---|---|
| 0 | NO_ERROR | No error occurred since LEC has been reset by successful reception or transmission. |
| 1 | STUFF_ERROR | More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed. |
| 2 | FORM_ERROR | A fixed format part of a received frame has the wrong format. |
| 3 | ACK_ERROR | The message transmitted by the MCAN was not acknowledged by another node. |
| 4 | BIT1_ERROR | During transmission of a message (with the exception of the arbitration field), the device tried to send a recessive level (bit of logical value ‘1’), but the monitored bus value was dominant. |
| 5 | BIT0_ERROR | During transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device tried to send a dominant level (data or identifier bit logical value ‘0’), but the monitored bus value was recessive. During Bus_Off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the processor to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). |
| 6 | CRC_ERROR | The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match the CRC calculated from the received data. |
| 7 | NO_CHANGE | Any read access to the Protocol Status Register re-initializes the LEC to ‘7’. When the LEC shows value ‘7’, no CAN bus event was detected since the last processor read access to the Protocol Status Register. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.15 MCAN Transmitter Delay Compensation Register

**Name:** MCAN_TDCR
**Offset:** 0x48
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | TDCO[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TDCF[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 14:8 – TDCO[6:0]** Transmitter Delay Compensation Offset
0 to 127: Offset value, in CAN core clock periods, defining the distance between the measured delay from CANTX to CANRX and the secondary sample point.

**Bits 6:0 – TDCF[6:0]** Transmitter Delay Compensation Filter
0 to 127: defines the minimum value for the SSP position, in CAN core clock periods. Dominant edges on CANRX that would result in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCO.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.16 MCAN Interrupt Register

**Name:** MCAN_IR
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read/Write

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the processor clears them. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. A hard reset will clear the register. The configuration of IE controls whether an interrupt is generated. The configuration of ILS controls on which interrupt line an interrupt is signalled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ARA | PED | PEA | WDI | BO | EW |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EP | ELO | | | DRX | TOO | MRAF | TSW |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TEFL | TEFF | TEFW | TEFN | TFE | TCF | TC | HPM |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RF1L | RF1F | RF1W | RF1N | RF0L | RF0F | RF0W | RF0N |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – ARA** Access to Reserved Address

| Value | Description |
|---|---|
| 0 | No access to reserved address occurred |
| 1 | Access to reserved address occurred |

**Bit 28 – PED** Protocol Error in Data Phase

| Value | Description |
|---|---|
| 0 | No protocol error in data phase |
| 1 | Protocol error in data phase detected (MCAN_PSR.DLEC differs from 0 or 7) |

**Bit 27 – PEA** Protocol Error in Arbitration Phase

| Value | Description |
|---|---|
| 0 | No protocol error in arbitration phase |
| 1 | Protocol error in arbitration phase detected (MCAN_PSR.LEC differs from 0 or 7) |

**Bit 26 – WDI** Watchdog Interrupt

| Value | Description |
|---|---|
| 0 | No Message RAM Watchdog event occurred. |
| 1 | Message RAM Watchdog event due to missing READY. |

**Bit 25 – BO** Bus_Off Status

| Value | Description |
|---|---|
| 0 | Bus_Off status unchanged. |
| 1 | Bus_Off status changed. |

**Bit 24 – EW** Warning Status

| Value | Description |
|---|---|
| 0 | Error_Warning status unchanged. |
| 1 | Error_Warning status changed. |

**Bit 23 – EP** Error Passive

| Value | Description |
|---|---|
| 0 | Error_Passive status unchanged. |
| 1 | Error_Passive status changed. |

**Bit 22 – ELO** Error Logging Overflow

| Value | Description |
|---|---|
| 0 | CAN Error Logging Counter did not overflow. |
| 1 | Overflow of CAN Error Logging Counter occurred. |

**Bit 19 – DRX** Message stored to Dedicated Receive Buffer

The flag is set whenever a received message has been stored into a dedicated Receive Buffer.

| Value | Description |
|---|---|
| 0 | No Receive Buffer updated. |
| 1 | At least one received message stored into a Receive Buffer. |

**Bit 18 – TOO** Timeout Occurred

| Value | Description |
|---|---|
| 0 | No timeout. |
| 1 | Timeout reached. |

**Bit 17 – MRAF** Message RAM Access Failure

The flag is set, when the Rx Handler

• has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message.

• was not able to write a message to the Message RAM. In this case message storage is aborted.

In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Receive Buffer is not set, a partly stored message is overwritten when the next message is stored to this location.

The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the MCAN is switched into Restricted Operation mode (see Restricted Operation Mode). To leave Restricted Operation mode, the processor has to reset MCAN_CCCR.ASM.

| Value | Description |
|---|---|
| 0 | No Message RAM access failure occurred. |
| 1 | Message RAM access failure occurred. |

**Bit 16 – TSW** Timestamp Wraparound

| Value | Description |
|---|---|
| 0 | No timestamp counter wrap-around. |
| 1 | Timestamp counter wrapped around. |

**Bit 15 – TEFL** Tx Event FIFO Element Lost

| Value | Description |
|---|---|
| 0 | No Tx Event FIFO element lost. |
| 1 | Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero. |

**Bit 14 – TEFF** Tx Event FIFO Full

| Value | Description |
|---|---|
| 0 | Tx Event FIFO not full. |
| 1 | Tx Event FIFO full. |

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 13 – TEFW** Tx Event FIFO Watermark Reached

| Value | Description |
|---|---|
| 0 | Tx Event FIFO fill level below watermark. |
| 1 | Tx Event FIFO fill level reached watermark. |

**Bit 12 – TEFN** Tx Event FIFO New Entry

| Value | Description |
|---|---|
| 0 | Tx Event FIFO unchanged. |
| 1 | Tx Handler wrote Tx Event FIFO element. |

**Bit 11 – TFE** Tx FIFO Empty

| Value | Description |
|---|---|
| 0 | Tx FIFO non-empty. |
| 1 | Tx FIFO empty. |

**Bit 10 – TCF** Transmission Cancellation Finished

| Value | Description |
|---|---|
| 0 | No transmission cancellation finished. |
| 1 | Transmission cancellation finished. |

**Bit 9 – TC** Transmission Completed

| Value | Description |
|---|---|
| 0 | No transmission completed. |
| 1 | Transmission completed. |

**Bit 8 – HPM** High Priority Message

| Value | Description |
|---|---|
| 0 | No high priority message received. |
| 1 | High priority message received. |

**Bit 7 – RF1L** Receive FIFO 1 Message Lost

| Value | Description |
|---|---|
| 0 | No Receive FIFO 1 message lost. |
| 1 | Receive FIFO 1 message lost, also set after write attempt to Receive FIFO 1 of size zero. |

**Bit 6 – RF1F** Receive FIFO 1 Full

| Value | Description |
|---|---|
| 0 | Receive FIFO 1 not full. |
| 1 | Receive FIFO 1 full. |

**Bit 5 – RF1W** Receive FIFO 1 Watermark Reached

| Value | Description |
|---|---|
| 0 | Receive FIFO 1 fill level below watermark. |
| 1 | Receive FIFO 1 fill level reached watermark. |

**Bit 4 – RF1N** Receive FIFO 1 New Message

| Value | Description |
|---|---|
| 0 | No new message written to Receive FIFO 1. |
| 1 | New message written to Receive FIFO 1. |

**Bit 3 – RF0L** Receive FIFO 0 Message Lost

| Value | Description |
|---|---|
| 0 | No Receive FIFO 0 message lost. |
| 1 | Receive FIFO 0 message lost, also set after write attempt to Receive FIFO 0 of size zero. |

**Bit 2 – RF0F** Receive FIFO 0 Full

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 0 | Receive FIFO 0 not full. |
| 1 | Receive FIFO 0 full. |

**Bit 1 – RF0W** Receive FIFO 0 Watermark Reached

| Value | Description |
|---|---|
| 0 | Receive FIFO 0 fill level below watermark. |
| 1 | Receive FIFO 0 fill level reached watermark. |

**Bit 0 – RF0N** Receive FIFO 0 New Message

| Value | Description |
|---|---|
| 0 | No new message written to Receive FIFO 0. |
| 1 | New message written to Receive FIFO 0. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.17 MCAN Interrupt Enable Register

**Name:** MCAN_IE
**Offset:** 0x54
**Reset:** 0x00000000
**Property:** Read/Write

The following configuration values are valid for all listed bit names of this register:

0: Disables the corresponding interrupt.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ARAE | PEDE | PEAE | WDIE | BOE | EWE |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EPE | ELOE | | | DRXE | TOOE | MRAFE | TSWE |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TEFLE | TEFFE | TEFWE | TEFNE | TFEE | TCFE | TCE | HPME |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RF1LE | RF1FE | RF1WE | RF1NE | RF0LE | RF0FE | RF0WE | RF0NE |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – ARAE** Access to Reserved Address Enable

**Bit 28 – PEDE** Protocol Error in Data Phase Enable

**Bit 27 – PEAE** Protocol Error in Arbitration Phase Enable

**Bit 26 – WDIE** Watchdog Interrupt Enable

**Bit 25 – BOE** Bus_Off Status Interrupt Enable

**Bit 24 – EWE** Warning Status Interrupt Enable

**Bit 23 – EPE** Error Passive Interrupt Enable

**Bit 22 – ELOE** Error Logging Overflow Interrupt Enable

**Bit 19 – DRXE** Message stored to Dedicated Receive Buffer Interrupt Enable

**Bit 18 – TOOE** Timeout Occurred Interrupt Enable

**Bit 17 – MRAFE** Message RAM Access Failure Interrupt Enable

**Bit 16 – TSWE** Timestamp Wraparound Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 15 – TEFLE** Tx Event FIFO Event Lost Interrupt Enable

**Bit 14 – TEFFE** Tx Event FIFO Full Interrupt Enable

**Bit 13 – TEFWE** Tx Event FIFO Watermark Reached Interrupt Enable

**Bit 12 – TEFNE** Tx Event FIFO New Entry Interrupt Enable

**Bit 11 – TFEE** Tx FIFO Empty Interrupt Enable

**Bit 10 – TCFE** Transmission Cancellation Finished Interrupt Enable

**Bit 9 – TCE** Transmission Completed Interrupt Enable

**Bit 8 – HPME** High Priority Message Interrupt Enable

**Bit 7 – RF1LE** Receive FIFO 1 Message Lost Interrupt Enable

**Bit 6 – RF1FE** Receive FIFO 1 Full Interrupt Enable

**Bit 5 – RF1WE** Receive FIFO 1 Watermark Reached Interrupt Enable

**Bit 4 – RF1NE** Receive FIFO 1 New Message Interrupt Enable

**Bit 3 – RF0LE** Receive FIFO 0 Message Lost Interrupt Enable

**Bit 2 – RF0FE** Receive FIFO 0 Full Interrupt Enable

**Bit 1 – RF0WE** Receive FIFO 0 Watermark Reached Interrupt Enable

**Bit 0 – RF0NE** Receive FIFO 0 New Message Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.18 MCAN Interrupt Line Select Register

**Name:** MCAN_ILS
**Offset:** 0x58
**Reset:** 0x00000000
**Property:** Read/Write

The Interrupt Line Select register assigns an interrupt generated by a specific interrupt flag from the Interrupt Register to one of the two module interrupt lines.

0: Interrupt assigned to interrupt line MCAN_INT0.

1: Interrupt assigned to interrupt line MCAN_INT1.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | ARAL | PEDL | PEAL | WDIL | BOL | EWL |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EPL | ELOL | | | DRXL | TOOL | MRAFL | TSWL |
| Access | R/W | R/W | | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TEFLL | TEFFL | TEFWL | TEFNL | TFEL | TCFL | TCL | HPML |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RF1LL | RF1FL | RF1WL | RF1NL | RF0LL | RF0FL | RF0WL | RF0NL |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 29 – ARAL** Access to Reserved Address Line

**Bit 28 – PEDL** Protocol Error in Data Phase Line

**Bit 27 – PEAL** Protocol Error in Arbitration Phase Line

**Bit 26 – WDIL** Watchdog Interrupt Line

**Bit 25 – BOL** Bus_Off Status Interrupt Line

**Bit 24 – EWL** Warning Status Interrupt Line

**Bit 23 – EPL** Error Passive Interrupt Line

**Bit 22 – ELOL** Error Logging Overflow Interrupt Line

**Bit 19 – DRXL** Message stored to Dedicated Receive Buffer Interrupt Line

**Bit 18 – TOOL** Timeout Occurred Interrupt Line

**Bit 17 – MRAFL** Message RAM Access Failure Interrupt Line

**Bit 16 – TSWL** Timestamp Wraparound Interrupt Line

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bit 15 – TEFLL** Tx Event FIFO Event Lost Interrupt Line

**Bit 14 – TEFFL** Tx Event FIFO Full Interrupt Line

**Bit 13 – TEFWL** Tx Event FIFO Watermark Reached Interrupt Line

**Bit 12 – TEFNL** Tx Event FIFO New Entry Interrupt Line

**Bit 11 – TFEL** Tx FIFO Empty Interrupt Line

**Bit 10 – TCFL** Transmission Cancellation Finished Interrupt Line

**Bit 9 – TCL** Transmission Completed Interrupt Line

**Bit 8 – HPML** High Priority Message Interrupt Line

**Bit 7 – RF1LL** Receive FIFO 1 Message Lost Interrupt Line

**Bit 6 – RF1FL** Receive FIFO 1 Full Interrupt Line

**Bit 5 – RF1WL** Receive FIFO 1 Watermark Reached Interrupt Line

**Bit 4 – RF1NL** Receive FIFO 1 New Message Interrupt Line

**Bit 3 – RF0LL** Receive FIFO 0 Message Lost Interrupt Line

**Bit 2 – RF0FL** Receive FIFO 0 Full Interrupt Line

**Bit 1 – RF0WL** Receive FIFO 0 Watermark Reached Interrupt Line

**Bit 0 – RF0NL** Receive FIFO 0 New Message Interrupt Line

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.19 MCAN Interrupt Line Enable

**Name:** MCAN_ILE
**Offset:** 0x5C
**Reset:** 0x00000000
**Property:** Read/Write

Each of the two interrupt lines to the processor can be enabled/disabled separately by programming bits EINT0 and EINT1.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | EINT1 | EINT0 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – EINT1** Enable Interrupt Line 1

| Value | Description |
|---|---|
| 0 | Interrupt line MCAN_INT1 disabled. |
| 1 | Interrupt line MCAN_INT1 enabled. |

**Bit 0 – EINT0** Enable Interrupt Line 0

| Value | Description |
|---|---|
| 0 | Interrupt line MCAN_INT0 disabled. |
| 1 | Interrupt line MCAN_INT0 enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.20 MCAN Global Filter Configuration

**Name:** MCAN_GFC
**Offset:** 0x80
**Reset:** 0x00000000
**Property:** Read/Write

Global settings for Message ID filtering. The Global Filter Configuration controls the filter path for standard and extended messages as illustrated in Standard Message ID Filter Path and Extended Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | ANFS[1:0] | | ANFE[1:0] | | RRFS | RRFE |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:4 – ANFS[1:0]** Accept Non-matching Frames Standard
Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated.

| Value | Name | Description |
|---|---|---|
| 0 | RX_FIFO_0 | Accept in Rx FIFO 0 |
| 1 | RX_FIFO_1 | Accept in Rx FIFO 1 |
| 2-3 | REJECTED | Message rejected |

**Bits 3:2 – ANFE[1:0]** Accept Non-matching Frames Extended
Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated.

| Value | Name | Description |
|---|---|---|
| 0 | RX_FIFO_0 | Accept in Rx FIFO 0 |
| 1 | RX_FIFO_1 | Accept in Rx FIFO 1 |
| 2-3 | REJECTED | Message rejected |

**Bit 1 – RRFS** Reject Remote Frames Standard
0 (FILTER): Filter remote frames with 11-bit standard IDs.
1 (REJECT): Reject all remote frames with 11-bit standard IDs.

**Bit 0 – RRFE** Reject Remote Frames Extended
0 (FILTER): Filter remote frames with 29-bit extended IDs.
1 (REJECT): Reject all remote frames with 29-bit extended IDs.

© 2021 Microchip Technology Inc. and its subsidiaries

### 49.6.21 MCAN Standard ID Filter Configuration

**Name:** MCAN_SIDFC
**Offset:** 0x84
**Reset:** 0x00000000
**Property:** Read/Write

Settings for 11-bit standard Message ID filtering. The Standard ID Filter Configuration controls the filter path for standard messages as illustrated in Standard Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | LSS[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FLSSA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLSSA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 23:16 – LSS[7:0]** List Size Standard
>128: Values greater than 128 are interpreted as 128.

| Value | Description |
|---|---|
| 0 | No standard Message ID filter. |
| 1-128 | Number of standard Message ID filter elements. |

**Bits 15:2 – FLSSA[13:0]** Filter List Standard Start Address
Start address of standard Message ID filter list (32-bit word address, see Message RAM Configuration).
Write FLSSA with the bits [15:2] of the 32-bit address.

### 49.6.22 MCAN Extended ID Filter Configuration

**Name:** MCAN_XIDFC
**Offset:** 0x88
**Reset:** 0x00000000
**Property:** Read/Write

Settings for 29-bit extended Message ID filtering. The Extended ID Filter Configuration controls the filter path for standard messages as described in Extended Message ID Filter Path.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | LSE[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FLESA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FLESA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 22:16 – LSE[6:0]** List Size Extended

| Value | Description |
|---|---|
| 0 | No extended Message ID filter. |
| 1-64 | Number of extended Message ID filter elements. |
| >64 | Values greater than 64 are interpreted as 64. |

**Bits 15:2 – FLESA[13:0]** Filter List Extended Start Address
Start address of extended Message ID filter list (32-bit word address, see Message RAM Configuration).
Write FLESA with the bits [15:2] of the 32-bit address.

### 49.6.23 MCAN Extended ID AND Mask

**Name:** MCAN_XIDAM
**Offset:** 0x90
**Reset:** 0x1FFFFFFF
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | EIDM[28:24] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 1 | 1 | 1 | 1 | 1 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | EIDM[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EIDM[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EIDM[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**Bits 28:0 – EIDM[28:0]** Extended ID Mask
For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active.

### 49.6.24 MCAN High Priority Message Status

**Name:** MCAN_HPMS
**Offset:** 0x94
**Reset:** 0x00000000
**Property:** Read-only

This register is updated every time a Message ID filter element configured to generate a priority event matches. This can be used to monitor the status of incoming high priority messages and to enable fast access to these messages.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FLST | FIDX[6:0] | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MSI[1:0] | | BIDX[5:0] | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 15 – FLST** Filter List
Indicates the filter list of the matching filter element.

| Value | Description |
|---|---|
| 0 | Standard filter list |
| 1 | Extended filter list |

**Bits 14:8 – FIDX[6:0]** Filter Index
Index of matching filter element. Range is 0 to MCAN_SIDFC.LSS - 1 resp. MCAN_XIDFC.LSE - 1.

**Bits 7:6 – MSI[1:0]** Message Storage Indicator

| Value | Name | Description |
|---|---|---|
| 0 | NO_FIFO_SEL | No FIFO selected. |
| 1 | LOST | FIFO message lost. |
| 2 | FIFO_0 | Message stored in FIFO 0. |
| 3 | FIFO_1 | Message stored in FIFO 1. |

**Bits 5:0 – BIDX[5:0]** Buffer Index
Index of Receive FIFO element to which the message was stored. Only valid when MSI[1] = ‘1’.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.25 MCAN New Data 1

**Name:** MCAN_NDAT1
**Offset:** 0x98
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ND31 | ND30 | ND29 | ND28 | ND27 | ND26 | ND25 | ND24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ND23 | ND22 | ND21 | ND20 | ND19 | ND18 | ND17 | ND16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ND15 | ND14 | ND13 | ND12 | ND11 | ND10 | ND9 | ND8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ND7 | ND6 | ND5 | ND4 | ND3 | ND2 | ND1 | ND0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – NDx** New Data
The register holds the New Data flags of Receive Buffers 0 to 31. The flags are set when the respective Receive Buffer has been updated from a received frame. The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register.

| Value | Description |
|---|---|
| 0 | Receive Buffer not updated |
| 1 | Receive Buffer updated from new message |

### 49.6.26 MCAN New Data 2

**Name:** MCAN_NDAT2
**Offset:** 0x9C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ND63 | ND62 | ND61 | ND60 | ND59 | ND58 | ND57 | ND56 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ND55 | ND54 | ND53 | ND52 | ND51 | ND50 | ND49 | ND48 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ND47 | ND46 | ND45 | ND44 | ND43 | ND42 | ND41 | ND40 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ND39 | ND38 | ND37 | ND36 | ND35 | ND34 | ND33 | ND32 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – NDx** New Data
The register holds the New Data flags of Receive Buffers 32 to 63. The flags are set when the respective Receive Buffer has been updated from a received frame. The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register.

| Value | Description |
|---|---|
| 0 | Receive Buffer not updated. |
| 1 | Receive Buffer updated from new message. |

### 49.6.27 MCAN Receive FIFO 0 Configuration

**Name:** MCAN_RXF0C
**Offset:** 0xA0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | F0OM | F0WM[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | F0S[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | F0SA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | F0SA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 31 – F0OM** FIFO 0 Operation Mode
FIFO 0 can be operated in Blocking or in Overwrite mode (see Rx FIFOs).

| Value | Description |
|---|---|
| 0 | FIFO 0 Blocking mode. |
| 1 | FIFO 0 Overwrite mode. |

**Bits 30:24 – F0WM[6:0]** Receive FIFO 0 Watermark

| Value | Description |
|---|---|
| 0 | Watermark interrupt disabled. |
| 1-64 | Level for Receive FIFO 0 watermark interrupt (MCAN_IR.RF0W). |
| >64 | Watermark interrupt disabled. |

**Bits 22:16 – F0S[6:0]** Receive FIFO 0 Size
The Receive FIFO 0 elements are indexed from 0 to F0S-1.

| Value | Description |
|---|---|
| 0 | No Receive FIFO 0 |
| 1-64 | Number of Receive FIFO 0 elements. |
| >64 | Values greater than 64 are interpreted as 64. |

**Bits 15:2 – F0SA[13:0]** Receive FIFO 0 Start Address
Start address of Receive FIFO 0 in Message RAM (32-bit word address, see Message RAM Configuration).
Write F0SA with the bits [15:2] of the 32-bit address.

### 49.6.28 MCAN Receive FIFO 0 Status

**Name:** MCAN_RXF0S
**Offset:** 0xA4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RF0L | F0F |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | F0PI[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | F0GI[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | F0FL[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 25 – RF0L** Receive FIFO 0 Message Lost
This bit is a copy of interrupt flag MCAN_IR.RF0L. When MCAN_IR.RF0L is reset, this bit is also reset.
Overwriting the oldest message when MCAN_RXF0C.F0OM = '1' will not set this flag.

| Value | Description |
|---|---|
| 0 | No Receive FIFO 0 message lost |
| 1 | Receive FIFO 0 message lost, also set after write attempt to Receive FIFO 0 of size zero |

**Bit 24 – F0F** Receive FIFO 0 Full

| Value | Description |
|---|---|
| 0 | Receive FIFO 0 not full. |
| 1 | Receive FIFO 0 full. |

**Bits 21:16 – F0PI[5:0]** Receive FIFO 0 Put Index
Receive FIFO 0 write index pointer, range 0 to 63.

**Bits 13:8 – F0GI[5:0]** Receive FIFO 0 Get Index
Receive FIFO 0 read index pointer, range 0 to 63.

**Bits 6:0 – F0FL[6:0]** Receive FIFO 0 Fill Level
Number of elements stored in Receive FIFO 0, range 0 to 64.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.29 MCAN Receive FIFO 0 Acknowledge

**Name:** MCAN_RXF0A
**Offset:** 0xA8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | F0AI[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – F0AI[5:0]** Receive FIFO 0 Acknowledge Index
After the processor has read a message or a sequence of messages from Receive FIFO 0 it has to write the buffer index of the last element read from Receive FIFO 0 to F0AI. This will set the Receive FIFO 0 Get Index MCAN_RXF0S.F0GI to F0AI + 1 and update the FIFO 0 Fill Level MCAN_RXF0S.F0FL.

### 49.6.30 MCAN Receive Buffer Configuration

**Name:** MCAN_RXBC
**Offset:** 0xAC
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RBSA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RBSA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 15:2 – RBSA[13:0]** Receive Buffer Start Address
Configures the start address of the Receive Buffers section in the Message RAM (32-bit word address, see Message RAM Configuration). Also used to reference debug messages A,B,C.
Write RBSA with the bits [15:2] of the 32-bit address.

### 49.6.31 MCAN Receive FIFO 1 Configuration

**Name:** MCAN_RXF1C
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | F1OM | F1WM[6:0] | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | F1S[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | F1SA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | F1SA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 31 – F1OM** FIFO 1 Operation Mode
FIFO 1 can be operated in Blocking or in Overwrite mode (see Rx FIFOs).

| Value | Description |
|---|---|
| 0 | FIFO 1 Blocking mode. |
| 1 | FIFO 1 Overwrite mode. |

**Bits 30:24 – F1WM[6:0]** Receive FIFO 1 Watermark

| Value | Description |
|---|---|
| 0 | Watermark interrupt disabled |
| 1-64 | Level for Receive FIFO 1 watermark interrupt (MCAN_IR.RF1W). |
| >64 | Watermark interrupt disabled. |

**Bits 22:16 – F1S[6:0]** Receive FIFO 1 Size
The elements in Receive FIFO 1 are indexed from 0 to F1S - 1.

| Value | Description |
|---|---|
| 0 | No Receive FIFO 1 |
| 1-64 | Number of elements in Receive FIFO 1. |
| >64 | Values greater than 64 are interpreted as 64. |

**Bits 15:2 – F1SA[13:0]** Receive FIFO 1 Start Address
Start address of Receive FIFO 1 in Message RAM (32-bit word address, see Message RAM Configuration).
Write F1SA with the bits [15:2] of the 32-bit address.

### 49.6.32 MCAN Receive FIFO 1 Status

**Name:** MCAN_RXF1S
**Offset:** 0xB4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DMS[1:0] | | | | | | RF1L | F1F |
| Access | R | R | | | | | R | R |
| Reset | 0 | 0 | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | F1PI[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | F1GI[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | F1FL[6:0] | | | | | | |
| Access | | R | R | R | R | R | R | R |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:30 – DMS[1:0]** Debug Message Status

| Value | Name | Description |
|---|---|---|
| 0 | IDLE | Idle state, wait for reception of debug messages, DMA request is cleared. |
| 1 | MSG_A | Debug message A received. |
| 2 | MSG_AB | Debug messages A, B received. |
| 3 | MSG_ABC | Debug messages A, B, C received, DMA request is set. |

**Bit 25 – RF1L** Receive FIFO 1 Message Lost
This bit is a copy of interrupt flag IR.RF1L. When IR.RF1L is reset, this bit is also reset.
Overwriting the oldest message when MCAN_RXF1C.F1OM = '1' will not set this flag.

| Value | Description |
|---|---|
| 0 | No Receive FIFO 1 message lost. |
| 1 | Receive FIFO 1 message lost, also set after write attempt to Receive FIFO 1 of size zero. |

**Bit 24 – F1F** Receive FIFO 1 Full

| Value | Description |
|---|---|
| 0 | Receive FIFO 1 not full. |
| 1 | Receive FIFO 1 full. |

**Bits 21:16 – F1PI[5:0]** Receive FIFO 1 Put Index
Receive FIFO 1 write index pointer, range 0 to 63.

**Bits 13:8 – F1GI[5:0]** Receive FIFO 1 Get Index
Receive FIFO 1 read index pointer, range 0 to 63.

**Bits 6:0 – F1FL[6:0]** Receive FIFO 1 Fill Level
Number of elements stored in Receive FIFO 1, range 0 to 64.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.33 MCAN Receive FIFO 1 Acknowledge

**Name:** MCAN_RXF1A
**Offset:** 0xB8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | F1AI[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 5:0 – F1AI[5:0]** Receive FIFO 1 Acknowledge Index
After the processor has read a message or a sequence of messages from Receive FIFO 1 it has to write the buffer index of the last element read from Receive FIFO 1 to F1AI. This will set the Receive FIFO 1 Get Index MCAN_RXF1S.F1GI to F1AI + 1 and update the FIFO 1 Fill Level MCAN_RXF1S.F1FL.

### 49.6.34 MCAN Receive Buffer / FIFO Element Size Configuration

**Name:** MCAN_RXESC
**Offset:** 0xBC
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Configures the number of data bytes belonging to a Receive Buffer / Receive FIFO element. Data field sizes >8 bytes are intended for CAN FD operation only.

In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Receive Buffer or Receive FIFO, only the number of bytes as configured by MCAN_RXESC are stored to the Receive Buffer resp. Receive FIFO element. The rest of the frame's data field is ignored.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RBDS[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | F1DS[2:0] | | | | F0DS[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 10:8 – RBDS[2:0]** Receive Buffer Data Field Size

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8-byte data field |
| 1 | 12_BYTE | 12-byte data field |
| 2 | 16_BYTE | 16-byte data field |
| 3 | 20_BYTE | 20-byte data field |
| 4 | 24_BYTE | 24-byte data field |
| 5 | 32_BYTE | 32-byte data field |
| 6 | 48_BYTE | 48-byte data field |
| 7 | 64_BYTE | 64-byte data field |

**Bits 6:4 – F1DS[2:0]** Receive FIFO 1 Data Field Size

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8-byte data field |
| 1 | 12_BYTE | 12-byte data field |
| 2 | 16_BYTE | 16-byte data field |
| 3 | 20_BYTE | 20-byte data field |
| 4 | 24_BYTE | 24-byte data field |
| 5 | 32_BYTE | 32-byte data field |
| 6 | 48_BYTE | 48-byte data field |
| 7 | 64_BYTE | 64-byte data field |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bits 2:0 – F0DS[2:0]** Receive FIFO 0 Data Field Size

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8-byte data field |
| 1 | 12_BYTE | 12-byte data field |
| 2 | 16_BYTE | 16-byte data field |
| 3 | 20_BYTE | 20-byte data field |
| 4 | 24_BYTE | 24-byte data field |
| 5 | 32_BYTE | 32-byte data field |
| 6 | 48_BYTE | 48-byte data field |
| 7 | 64_BYTE | 64-byte data field |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.35 MCAN Tx Buffer Configuration

**Name:** MCAN_TXBC
**Offset:** 0xC0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

The sum of TFQS and NDTB may not exceed 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | TFQM | TFQS[5:0] | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | NDTB[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TBSA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TBSA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bit 30 – TFQM** Tx FIFO/Queue Mode

| Value | Description |
|---|---|
| 0 | Tx FIFO operation. |
| 1 | Tx Queue operation. |

**Bits 29:24 – TFQS[5:0]** Transmit FIFO/Queue Size

| Value | Description |
|---|---|
| 0 | No Tx FIFO/Queue. |
| 1-32 | Number of Tx Buffers used for Tx FIFO/Queue. |
| >32 | Values greater than 32 are interpreted as 32. |

**Bits 21:16 – NDTB[5:0]** Number of Dedicated Transmit Buffers

| Value | Description |
|---|---|
| 0 | No dedicated Tx Buffers. |
| 1-32 | Number of dedicated Tx Buffers. |
| >32 | Values greater than 32 are interpreted as 32. |

**Bits 15:2 – TBSA[13:0]** Tx Buffers Start Address
Start address of Tx Buffers section in Message RAM (32-bit word address, see Message RAM Configuration).
Write TBSA with the bits [15:2] of the 32-bit address.

### 49.6.36 MCAN Tx FIFO/Queue Status

**Name:** MCAN_TXFQS
**Offset:** 0xC4
**Reset:** 0x00000000
**Property:** Read-only

The Tx FIFO/Queue status is related to the pending Tx requests listed in register MCAN_TXBRP. Therefore the effect of Add/Cancellation requests may be delayed due to a running Tx scan (MCAN_TXBRP not yet updated).

In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers.
Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | TFQF | TFQPI[4:0] | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | TFGI[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TFFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 21 – TFQF** Tx FIFO/Queue Full

| Value | Description |
|---|---|
| 0 | Tx FIFO/Queue not full. |
| 1 | Tx FIFO/Queue full. |

**Bits 20:16 – TFQPI[4:0]** Tx FIFO/Queue Put Index
Tx FIFO/Queue write index pointer, range 0 to 31.

**Bits 12:8 – TFGI[4:0]** Tx FIFO Get Index
Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (MCAN_TXBC.TFQM = ‘1’).

**Bits 5:0 – TFFL[5:0]** Tx FIFO Free Level
Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (MCAN_TXBC.TFQM = ‘1’).

### 49.6.37 MCAN Tx Buffer Element Size Configuration

**Name:** MCAN_TXESC
**Offset:** 0xC8
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Configures the number of data bytes belonging to a Tx Buffer element. Data field sizes > 8 bytes are intended for CAN FD operation only.

In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size MCAN_TXESC.TBDS, the bytes not defined by the Tx Buffer are transmitted as “0xCC” (padding bytes).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | TBDS[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – TBDS[2:0]** Tx Buffer Data Field Size

| Value | Name | Description |
|---|---|---|
| 0 | 8_BYTE | 8-byte data field |
| 1 | 12_BYTE | 12-byte data field |
| 2 | 16_BYTE | 16-byte data field |
| 3 | 20_BYTE | 20-byte data field |
| 4 | 24_BYTE | 24-byte data field |
| 5 | 32_BYTE | 32-byte data field |
| 6 | 48_BYTE | 48- byte data field |
| 7 | 64_BYTE | 64-byte data field |

### 49.6.38 MCAN Transmit Buffer Request Pending

**Name:** MCAN_TXBRP
**Offset:** 0xCC
**Reset:** 0x00000000
**Property:** Read-only

MCAN_TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this Add Request is cancelled immediately, the corresponding MCAN_TXBRP bit is reset.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TRP31 | TRP30 | TRP29 | TRP28 | TRP27 | TRP26 | TRP25 | TRP24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TRP23 | TRP22 | TRP21 | TRP20 | TRP19 | TRP18 | TRP17 | TRP16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TRP15 | TRP14 | TRP13 | TRP12 | TRP11 | TRP10 | TRP9 | TRP8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TRP7 | TRP6 | TRP5 | TRP4 | TRP3 | TRP2 | TRP1 | TRP0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – TRPx** Transmission Request Pending for Buffer x
Each Tx Buffer has its own Transmission Request Pending bit. The bits are set via register MCAN_TXBAR. The bits are reset after a requested transmission has completed or has been cancelled via register MCAN_TXBCR.
TXBRP bits are set only for those Tx Buffers configured via MCAN_TXBC. After a MCAN_TXBRP bit has been set, a Tx scan (see Tx Handling) is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID).
A cancellation request resets the corresponding transmission request pending bit of register MCAN_TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding TXBRP bit has been reset.
After a cancellation has been requested, a finished cancellation is signalled via MCAN_TXBCF.
• after successful transmission together with the corresponding MCAN_TXBTO bit.
• when the transmission has not yet been started at the point of cancellation.
• when the transmission has been aborted due to lost arbitration.
• when an error occurred during frame transmission.
In DAR mode, all transmissions are automatically cancelled if they are not successful. The corresponding MCAN_TXBCF bit is set for all unsuccessful transmissions.

| Value | Description |
|---|---|
| 0 | No transmission request pending |
| 1 | Transmission request pending |

### 49.6.39 MCAN Transmit Buffer Add Request

**Name:** MCAN_TXBAR
**Offset:** 0xD0
**Reset:** 0x00000000
**Property:** Read/Write

If an add request is applied for a Transmit Buffer with pending transmission request (corresponding MCAN_TXBRP bit already set), this Add Request is ignored.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AR31 | AR30 | AR29 | AR28 | AR27 | AR26 | AR25 | AR24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AR23 | AR22 | AR21 | AR20 | AR19 | AR18 | AR17 | AR16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AR15 | AR14 | AR13 | AR12 | AR11 | AR10 | AR9 | AR8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AR7 | AR6 | AR5 | AR4 | AR3 | AR2 | AR1 | AR0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – ARx** Add Request for Transmit Buffer x
Each Transmit Buffer has its own Add Request bit. Writing a '1' will set the corresponding Add Request bit; writing a '0' has no impact. This enables the processor to set transmission requests for multiple Transmit Buffers with one write to MCAN_TXBAR. MCAN_TXBAR bits are set only for those Transmit Buffers configured via TXBC. When no Transmit scan is running, the bits are reset immediately, else the bits remain set until the Transmit scan process has completed.

| Value | Description |
|---|---|
| 0 | No transmission request added. |
| 1 | Transmission requested added. |

### 49.6.40 MCAN Transmit Buffer Cancellation Request

**Name:** MCAN_TXBCR
**Offset:** 0xD4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CR31 | CR30 | CR29 | CR28 | CR27 | CR26 | CR25 | CR24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CR23 | CR22 | CR21 | CR20 | CR19 | CR18 | CR17 | CR16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CR15 | CR14 | CR13 | CR12 | CR11 | CR10 | CR9 | CR8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CR7 | CR6 | CR5 | CR4 | CR3 | CR2 | CR1 | CR0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – CRx** Cancellation Request for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Request bit. Writing a ‘1’ will set the corresponding Cancellation Request bit; writing a ‘0’ has no impact. This enables the processor to set cancellation requests for multiple Transmit Buffers with one write to MCAN_TXBCR. MCAN_TXBCR bits are set only for those Transmit Buffers configured via TXBC. The bits remain set until the corresponding bit of MCAN_TXBRP is reset.

| Value | Description |
|---|---|
| 0 | No cancellation pending. |
| 1 | Cancellation pending. |

### 49.6.41 MCAN Transmit Buffer Transmission Occurred

**Name:** MCAN_TXBTO
**Offset:** 0xD8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TO31 | TO30 | TO29 | TO28 | TO27 | TO26 | TO25 | TO24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TO23 | TO22 | TO21 | TO20 | TO19 | TO18 | TO17 | TO16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TO15 | TO14 | TO13 | TO12 | TO11 | TO10 | TO9 | TO8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TO7 | TO6 | TO5 | TO4 | TO3 | TO2 | TO1 | TO0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – TOx** Transmission Occurred for Buffer x
Each Transmit Buffer has its own Transmission Occurred bit. The bits are set when the corresponding MCAN_TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a ‘1’ to the corresponding bit of register MCAN_TXBAR.

| Value | Description |
|---|---|
| 0 | No transmission occurred. |
| 1 | Transmission occurred. |

### 49.6.42 MCAN Transmit Buffer Cancellation Finished

**Name:** MCAN_TXBCF
**Offset:** 0xDC
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CF31 | CF30 | CF29 | CF28 | CF27 | CF26 | CF25 | CF24 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CF23 | CF22 | CF21 | CF20 | CF19 | CF18 | CF17 | CF16 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CF15 | CF14 | CF13 | CF12 | CF11 | CF10 | CF9 | CF8 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CF7 | CF6 | CF5 | CF4 | CF3 | CF2 | CF1 | CF0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – CFx** Cancellation Finished for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Finished bit. The bits are set when the corresponding MCAN_TXBRP bit is cleared after a cancellation was requested via MCAN_TXBCR. In case the corresponding MCAN_TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register MCAN_TXBAR.

| Value | Description |
|---|---|
| 0 | No transmit buffer cancellation. |
| 1 | Transmit buffer cancellation finished. |

### 49.6.43 MCAN Transmit Buffer Transmission Interrupt Enable

**Name:** MCAN_TXBTIE
**Offset:** 0xE0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TIE31 | TIE30 | TIE29 | TIE28 | TIE27 | TIE26 | TIE25 | TIE24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TIE23 | TIE22 | TIE21 | TIE20 | TIE19 | TIE18 | TIE17 | TIE16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TIE15 | TIE14 | TIE13 | TIE12 | TIE11 | TIE10 | TIE9 | TIE8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TIE7 | TIE6 | TIE5 | TIE4 | TIE3 | TIE2 | TIE1 | TIE0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – TIEx** Transmission Interrupt Enable for Buffer x
Each Transmit Buffer has its own Transmission Interrupt Enable bit.

| Value | Description |
|---|---|
| 0 | Transmission interrupt disabled |
| 1 | Transmission interrupt enable |

### 49.6.44 MCAN Transmit Buffer Cancellation Finished Interrupt Enable

**Name:** MCAN_TXBCIE
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CFIE31 | CFIE30 | CFIE29 | CFIE28 | CFIE27 | CFIE26 | CFIE25 | CFIE24 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CFIE23 | CFIE22 | CFIE21 | CFIE20 | CFIE19 | CFIE18 | CFIE17 | CFIE16 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CFIE15 | CFIE14 | CFIE13 | CFIE12 | CFIE11 | CFIE10 | CFIE9 | CFIE8 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CFIE7 | CFIE6 | CFIE5 | CFIE4 | CFIE3 | CFIE2 | CFIE1 | CFIE0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31 – CFIEx** Cancellation Finished Interrupt Enable for Transmit Buffer x
Each Transmit Buffer has its own Cancellation Finished Interrupt Enable bit.

| Value | Description |
|---|---|
| 0 | Cancellation finished interrupt disabled. |
| 1 | Cancellation finished interrupt enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 49.6.45 MCAN Transmit Event FIFO Configuration

**Name:** MCAN_TXEFC
**Offset:** 0xF0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | EFWM[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | EFS[5:0] | | | | | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | EFSA[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EFSA[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | | |

**Bits 29:24 – EFWM[5:0]** Event FIFO Watermark

| Value | Description |
|---|---|
| 0 | Watermark interrupt disabled. |
| 1-32 | Level for Tx Event FIFO watermark interrupt (MCAN_IR.TEFW). |
| >32 | Watermark interrupt disabled. |

**Bits 21:16 – EFS[5:0]** Event FIFO Size
The Tx Event FIFO elements are indexed from 0 to EFS - 1.

| Value | Description |
|---|---|
| 0 | Tx Event FIFO disabled. |
| 1-32 | Number of Tx Event FIFO elements. |
| >32 | Values greater than 32 are interpreted as 32. |

**Bits 15:2 – EFSA[13:0]** Event FIFO Start Address
Start address of Tx Event FIFO in Message RAM (32-bit word address, see Message RAM Configuration).
Write EFSA with the bits [15:2] of the 32-bit address.

### 49.6.46 MCAN Tx Event FIFO Status

**Name:** MCAN_TXEFS
**Offset:** 0xF4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | TEFL | EFF |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | EFPI[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | EFGI[4:0] | | | | |
| Access | | | | R | R | R | R | R |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EFFL[5:0] | | | | | |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 25 – TEFL** Tx Event FIFO Element Lost
This bit is a copy of interrupt flag MCAN_IR.TEFL. When MCAN_IR.TEFL is reset, this bit is also reset.

| Value | Description |
|---|---|
| 0 | No Tx Event FIFO element lost. |
| 1 | Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero. |

**Bit 24 – EFF** Event FIFO Full

| Value | Description |
|---|---|
| 0 | Tx Event FIFO not full. |
| 1 | Tx Event FIFO full. |

**Bits 20:16 – EFPI[4:0]** Event FIFO Put Index
Tx Event FIFO write index pointer, range 0 to 31.

**Bits 12:8 – EFGI[4:0]** Event FIFO Get Index
Tx Event FIFO read index pointer, range 0 to 31.

**Bits 5:0 – EFFL[5:0]** Event FIFO Fill Level
Number of elements stored in Tx Event FIFO, range 0 to 32.

### 49.6.47 MCAN Tx Event FIFO Acknowledge

**Name:** MCAN_TXEFA
**Offset:** 0xF8
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | EFAI[4:0] | | | | |
| Access | | | | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | 0 | 0 | 0 | 0 | 0 |

**Bits 4:0 – EFAI[4:0]** Event FIFO Acknowledge Index
After the processor has read an element or a sequence of elements from the Tx Event FIFO, it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index MCAN_TXEFS.EFGI to EFAI + 1 and update the FIFO 0 Fill Level MCAN_TXEFS.EFFL.

# 50. Timer Counter (TC)

## 50.1 Description

A Timer Counter (TC) module includes three identical TC channels. The number of implemented TC modules is device-specific.

Each TC channel can be independently programmed to perform a wide range of functions including frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse width modulation.

Each channel has three external clock inputs, five internal clock inputs and two multipurpose input/output signals which can be configured by the user. Each channel drives an internal interrupt signal which can be programmed to generate processor interrupts.

The TC embeds a quadrature decoder (QDEC) connected in front of the timers and driven by TIOA0, TIOB0 and TIOB1 inputs. When enabled, the QDEC performs the input lines filtering, decoding of quadrature signals and connects to the timers/counters in order to read the position and speed of the motor through the user interface.

The TC block has two global registers which act upon all TC channels:

- Block Control register (TC_BCR)—allows channels to be started simultaneously with the same instruction
- Block Mode register (TC_BMR)—defines the external clock inputs for each channel, allowing them to be chained

## 50.2 Embedded Characteristics

- Total of 12 Channels
- 16-bit Channel Size
- Wide Range of Functions Including:
  - Frequency measurement
  - Event counting
  - Interval measurement
  - Pulse generation
  - Delay timing
  - Pulse Width Modulation
  - Up/down capabilities
  - Quadrature decoder
  - 2-bit Gray up/down count for stepper motor
- Each Channel is User-Configurable and Contains:
  - Three external clock inputs
  - Five Internal clock inputs
  - Two multipurpose input/output signals acting as trigger event
  - Trigger/capture events can be directly synchronized by PWM signals
- Internal Interrupt Signal
- Read of the Capture Registers by the DMAC
- Compare Event Fault Generation for PWM
- Register Write Protection

© 2021 Microchip Technology Inc.
and its subsidiaries

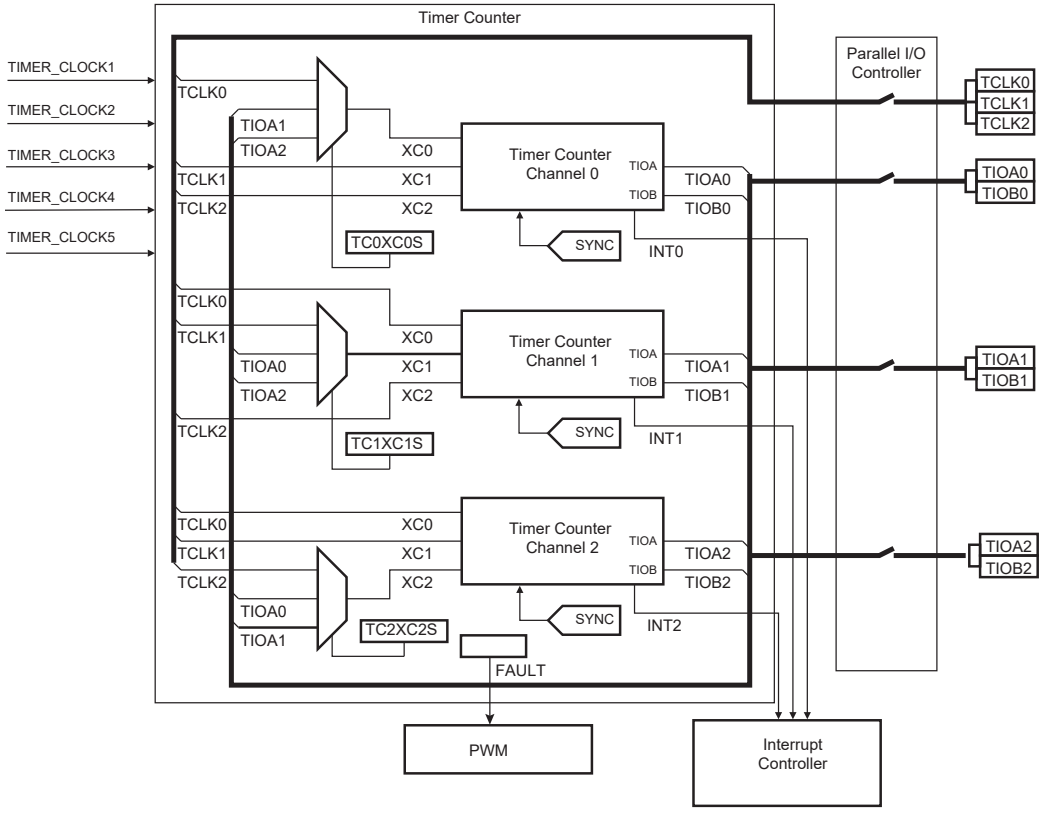
## 50.3 Block Diagram

**Table 50-1. Timer Counter Clock Assignment**

| Name | Definition |
|---|---|
| TIMER_CLOCK1 | PCK6 or PCK7 (TC0 only) |
| TIMER_CLOCK2 | MCK/8 |
| TIMER_CLOCK3 | MCK/32 |
| TIMER_CLOCK4 | MCK/128 |
| TIMER_CLOCK5 (1) | SLCK |

1. When SLCK is selected for Peripheral Clock (CSS = 0 in PMC Master Clock register), SLCK input is equivalent to Peripheral Clock.
2. The PCK6 or PCK7 (TC0 only) frequency must be at least three times lower than peripheral clock frequency.

**Figure 50-1. Timer Counter Block Diagram**




**Note:**
The QDEC connections are detailed in Predefined Connection of the Quadrature Decoder with Timer Counters.

**Table 50-2. Channel Signal Description**

| Signal Name | Description |
|---|---|
| XC0, XC1, XC2 | External Clock Inputs |
| TIOAx | Capture Mode: Timer Counter Input<br>Waveform Mode: Timer Counter Output |

| .....continued | |
|---|---|
| **Signal Name** | **Description** |
| TIOBx | Capture Mode: Timer Counter Input<br>Waveform Mode: Timer Counter Input/Output |
| INT | Interrupt Signal Output (internal signal) |
| SYNC | Synchronization Input Signal (from configuration register) |

## 50.4 Pin List

**Table 50-3. Pin List**

| Pin Name | Description | Type |
|---|---|---|
| TCLK0–TCLK2 | External Clock Input | Input |
| TIOA0–TIOA2 | I/O Line A | I/O |
| TIOB0–TIOB2 | I/O Line B | I/O |

## 50.5 Product Dependencies

### 50.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the TC pins to their peripheral functions.

### 50.5.2 Power Management

The TC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the Timer Counter clock of each channel.

### 50.5.3 Interrupt Sources

The TC has an interrupt line per channel connected to the interrupt controller. Handling the TC interrupt requires programming the interrupt controller before configuring the TC.

### 50.5.4 Synchronization Inputs from PWM

The TC has trigger/capture inputs internally connected to the PWM. Refer to "Synchronization with PWM" and to the implementation of the Pulse Width Modulation (PWM) in this product.

### 50.5.5 Fault Output

The TC has the FAULT output internally connected to the fault input of PWM. Refer to "Fault Mode" and to the implementation of the Pulse Width Modulation (PWM) in this product.

## 50.6 Functional Description

### 50.6.1 Description

All channels of the Timer Counter are independent and identical in operation except when the QDEC is enabled. The registers for channel programming are listed in 50.7 Register Summary.

### 50.6.2 16-bit Counter

Each 16-bit channel is organized around a 16-bit counter. The value of the counter is incremented at each positive edge of the selected clock. When the counter has reached the value $2^{16}$-1 and passes to zero, an overflow occurs and the COVFS bit in the Interrupt Status register (TC_SR) is set.

The current value of the counter is accessible in real time by reading the Counter Value register (TC_CV). The counter can be reset by a trigger. In this case, the counter value passes to zero on the next valid edge of the selected clock.

### 50.6.3 Clock Selection

At block level, input clock signals of each channel can be connected either to the external inputs TCLKx, or to the internal I/O signals TIOAx for chaining[1] by programming the Block Mode register (TC_BMR). See Clock Chaining Selection.

Each channel can independently select an internal or external clock source for its counter[2]:

- External clock signals: XC0, XC1 or XC2
- Internal clock signals: PCK6 or PCK7 (TC0 only), MCK/8, MCK/32, MCK/128, SLCK

This selection is made by the TCCLKS bits in the Channel Mode register (TC_CMRx).

The selected clock can be inverted with TC_CMRx.CLKI. This allows counting on the opposite edges of the clock.

The burst function allows the clock to be validated when an external signal is high. The BURST parameter in the TC_CMRx defines this signal (none, XC0, XC1, XC2). See Clock Selection.

**Notes:**

1. In Waveform mode, to chain two timers, it is mandatory to initialize some parameters:
   – Configure TIOx outputs to 1 or 0 by writing the required value to TC_CMRx.ASWTRG.
   – Bit TC_BCR.SYNC must be written to 1 to start the channels at the same time.
2. In all cases, if an external clock or asynchronous internal clock PCK6 or PCK7 (TC0 only) is used, the duration of each of its levels must be longer than the peripheral clock period, so the clock frequency will be at least 2.5 times lower than the peripheral clock.

**Figure 50-2. Clock Chaining Selection**

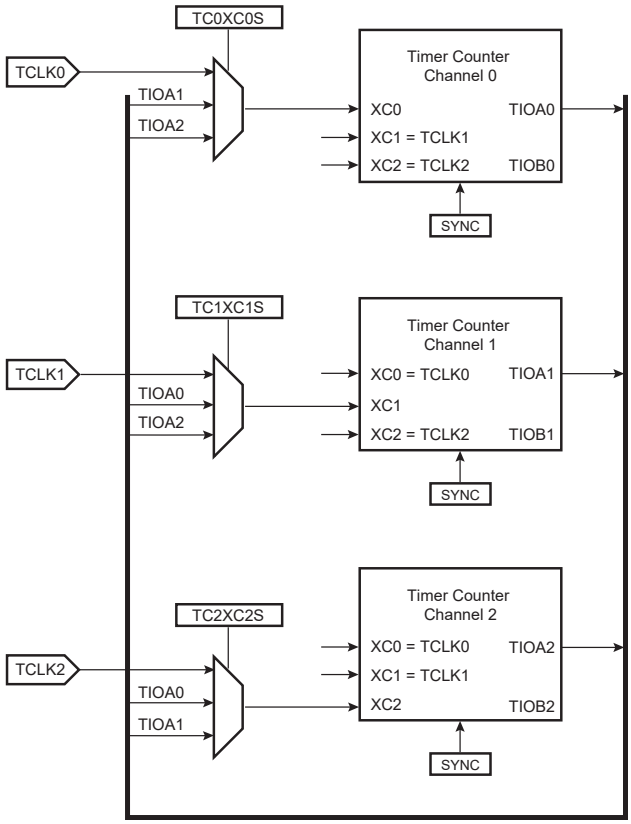
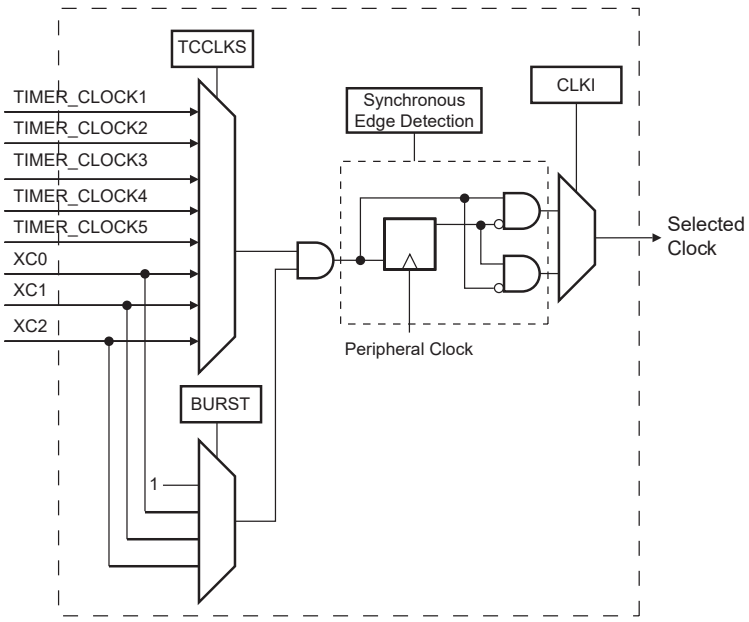


**Figure 50-3. Clock Selection**

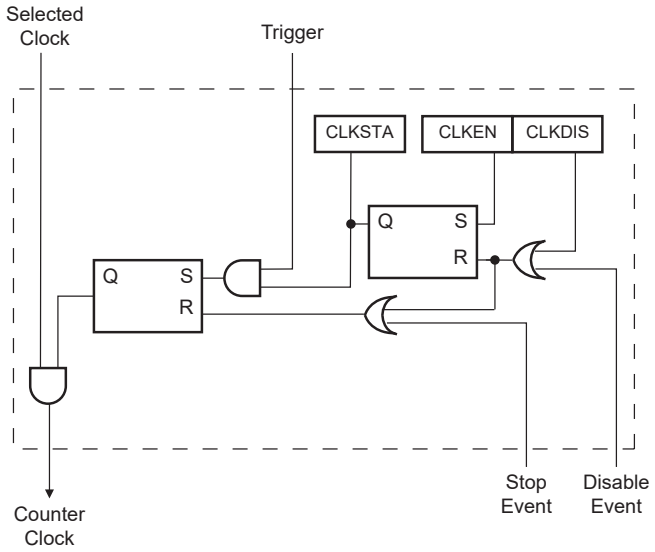



### 50.6.4 Clock Control

The clock of each counter can be controlled in two different ways: it can be enabled/disabled and started/stopped, as shown in the following figure.

- The clock can be enabled or disabled by the user with the CLKEN and the CLKDIS commands in the Channel Control register (TC_CCR). In Capture mode it can be disabled by an RB load event if TC_CMRx.LDBDIS is set to ‘1’. In Waveform mode, it can be disabled by an RC Compare event if TC_CMRx.CPCDIS is set to ‘1’. When disabled, the start or the stop actions have no effect: only a CLKEN command in the TC_CCR can reenable the clock. When the clock is enabled, TC_SR.CLKSTA is set.
- The clock can also be started or stopped: a trigger (software, synchro, external or compare) always starts the clock. The clock can be stopped by an RB load event in Capture mode (TC_CMRx.LDBSTOP = 1) or an RC compare event in Waveform mode (TC_CMRx.CPCSTOP = 1). The start and the stop commands are effective only if the clock is enabled.

**Figure 50-4. Clock Control**



### 50.6.5 Operating Modes

Each channel can operate independently in two different modes:

- Capture mode provides measurement on signals.
- Waveform mode provides wave generation.

The TC operating mode is programmed with TC_CMRx.WAVE.

In Capture mode, TIOAx and TIOBx are configured as inputs.

In Waveform mode, TIOAx is always configured to be an output and TIOBx is an output if it is not selected to be the external trigger.

### 50.6.6 Trigger

A trigger resets the counter and starts the counter clock. Three types of triggers are common to both modes, and a fourth external trigger is available to each mode.

Regardless of the trigger used, it will be taken into account at the following active edge of the selected clock. This means that the counter value can be read differently from zero just after a trigger, especially when a low frequency signal is selected as the clock.

The following triggers are common to both modes:

- Software Trigger: Each channel has a software trigger, available by setting TC_CCR.SWTRG.
- SYNC: Each channel has a synchronization signal SYNC. When asserted, this signal has the same effect as a software trigger. The SYNC signals of all channels are asserted simultaneously by writing TC_BCR with SYNC set.
- Compare RC Trigger: RC is implemented in each channel and can provide a trigger when the counter value matches the RC value if TC_CMRx.CPCTRG is set .

The channel can also be configured to have an external trigger. In Capture mode, the external trigger signal can be selected between TIOAx and TIOBx. In Waveform mode, an external event can be programmed on one of the following signals: TIOBx, XC0, XC1 or XC2. This external event can then be programmed to perform a trigger by setting TC_CMRx.ENETRG.

If an external trigger is used, the duration of the pulses must be longer than the peripheral clock period in order to be detected.

### 50.6.7 Capture Mode

Capture mode is entered by clearing TC_CMRx.WAVE.

Capture mode allows the TC channel to perform measurements such as pulse timing, frequency, period, duty cycle and phase on TIOAx and TIOBx signals which are considered as inputs.

The figure Figure 50-6 shows the configuration of the TC channel when programmed in Capture mode.

### 50.6.8 Capture Registers A and B

Registers A and B (TC_RA and TC_RB) are used as capture registers. They can be loaded with the counter value when a programmable event occurs on the signal TIOAx.

TC_CMRx.LDRA defines the TIOAx selected edge for the loading of TC_RA, and TC_CMRx.LDRB defines the TIOAx selected edge for the loading of TC_RB.

The subsampling ratio defined by TC_CMRx.SBSMPLR is applied to these selected edges, so that the loading of Register A and Register B occurs once every 1, 2, 4, 8 or 16 selected edges.

TC_RA is loaded only if it has not been loaded since the last trigger or if TC_RB has been loaded since the last loading of TC_RA.

TC_RB is loaded only if TC_RA has been loaded since the last trigger or the last loading of TC_RB.

Loading TC_RA or TC_RB before the read of the last value loaded sets TC_SR.LOVRS. In this case, the old value is overwritten.

© 2021 Microchip Technology Inc.
and its subsidiaries

When DMA is used (on channel 0), the Register AB (TC_RAB) address must be configured as source address of the transfer. TC_RAB provides the next unread value from TC_RA and TC_RB. It may be read by the DMA after a request has been triggered upon loading TC_RA or TC_RB.

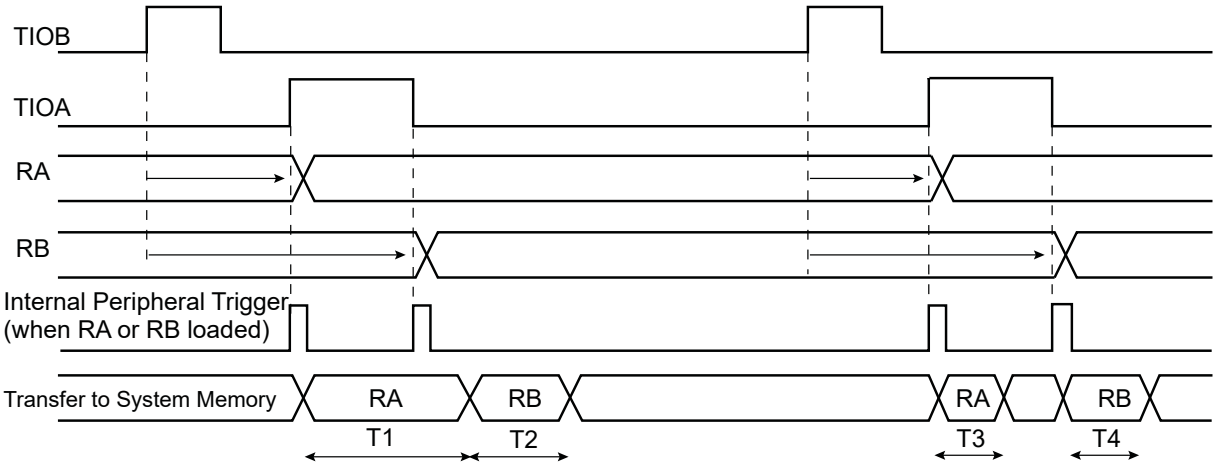
### 50.6.9 Transfer with DMAC in Capture Mode

The DMAC can perform access from the TC to system memory in Capture mode only.

The following figure illustrates how TC_RA and TC_RB can be loaded in the system memory without processor intervention.

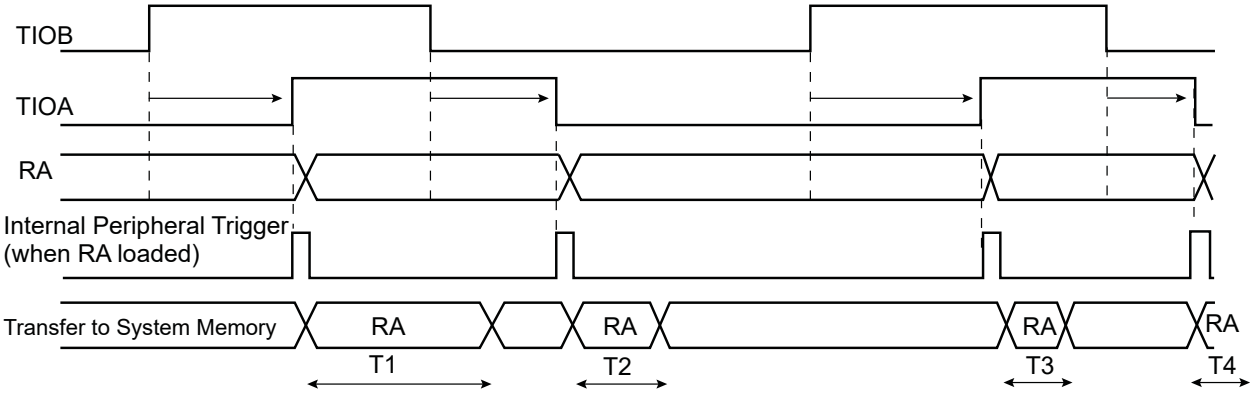
**Figure 50-5. Example of Transfer with DMAC in Capture Mode**

ETRGEDG = 1, LDRA = 1, LDRB = 2, ABETRG = 0




T1,T2,T3,T4 = System Bus load dependent ($t_{min}$ = 8 Peripheral Clocks)

ETRGEDG = 3, LDRA = 3, LDRB = 0, ABETRG = 0



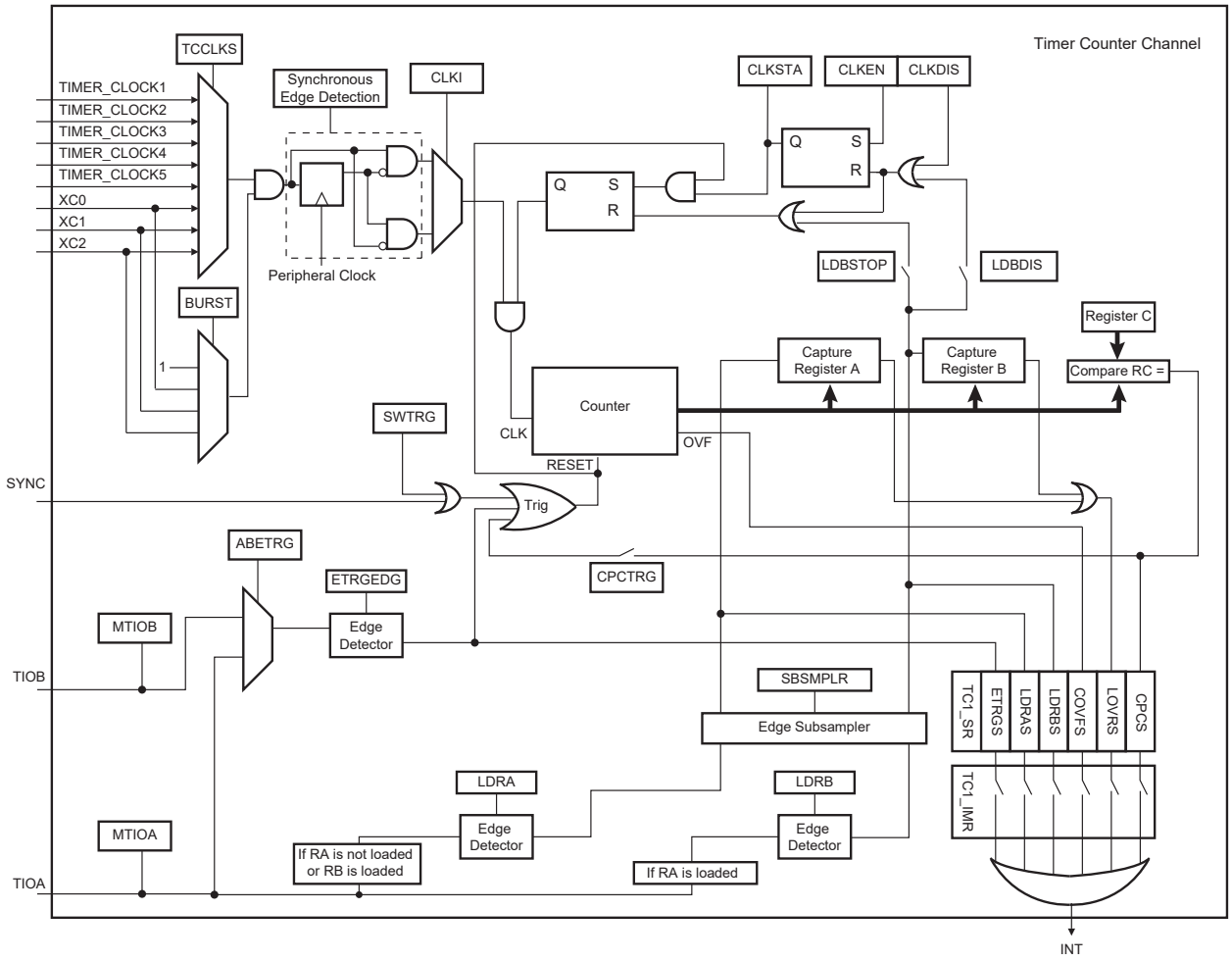

T1,T2,T3,T4 = System Bus load dependent ($t_{min}$ = 8 Peripheral Clocks)

### 50.6.10 Trigger Conditions

In addition to the SYNC signal, the software trigger and the RC compare trigger, an external trigger can be defined.

The ABETRG bit in the TC_CMR selects TIOAx or TIOBx input signal as an external trigger or the trigger signal from the output comparator of the PWM module. The External Trigger Edge Selection parameter (ETRGEDG field in TC_CMR) defines the edge (rising, falling, or both) detected to generate an external trigger. If ETRGEDG = 0 (none), the external trigger is disabled.

**Figure 50-6. Capture Mode**




### 50.6.11 Waveform Mode

Waveform mode is entered by setting the TC_CMRx.WAVE bit.

In Waveform mode, the TC channel generates one or two PWM signals with the same frequency and independently programmable duty cycles, or generates different types of one-shot or repetitive pulses.

In this mode, TIOAx is configured as an output and TIOBx is defined as an output if it is not used as an external event (EEVT parameter in TC_CMR).

Waveform Mode shows the configuration of the TC channel when programmed in Waveform operating mode.

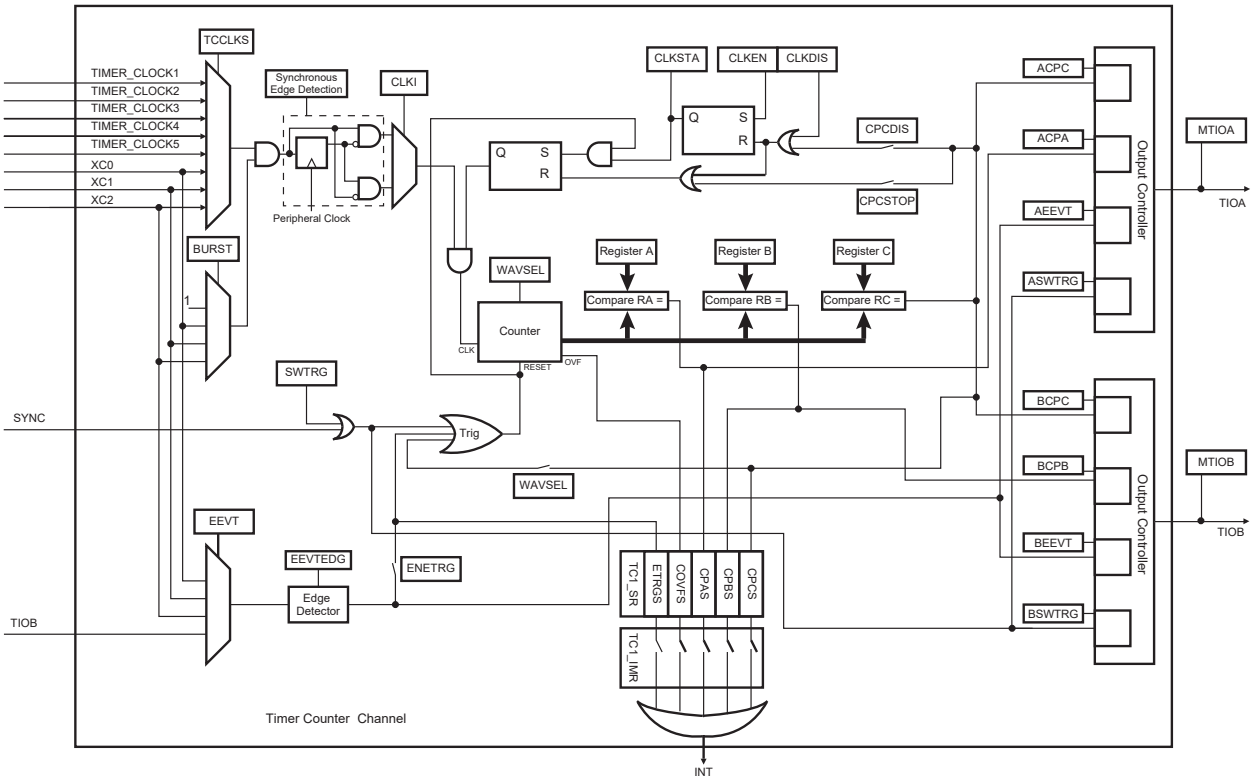
### 50.6.12 Waveform Selection

Depending on the WAVSEL parameter in TC_CMR, the behavior of TC_CV varies.

With any selection, TC_RA, TC_RB and TC_RC can all be used as compare registers.

RA Compare is used to control the TIOAx output, RB Compare is used to control the TIOBx output (if correctly configured) and RC Compare is used to control TIOAx and/or TIOBx outputs.

**Figure 50-7. Waveform Mode**




### 50.6.12.1 WAVSEL = 00

When WAVSEL = 00, the value of TC_CV is incremented from 0 to $2^{16}$-1. Once $2^{16}$-1 has been reached, the value of TC_CV is reset. Incrementation of TC_CV starts again and the cycle continues.

An external event trigger or a software trigger can reset the value of TC_CV. It is important to note that the trigger may occur at any time.

Refer to the figures below.

RC Compare cannot be programmed to generate a trigger in this configuration. At the same time, RC Compare can stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

**Figure 50-8. WAVSEL = 00 without Trigger**

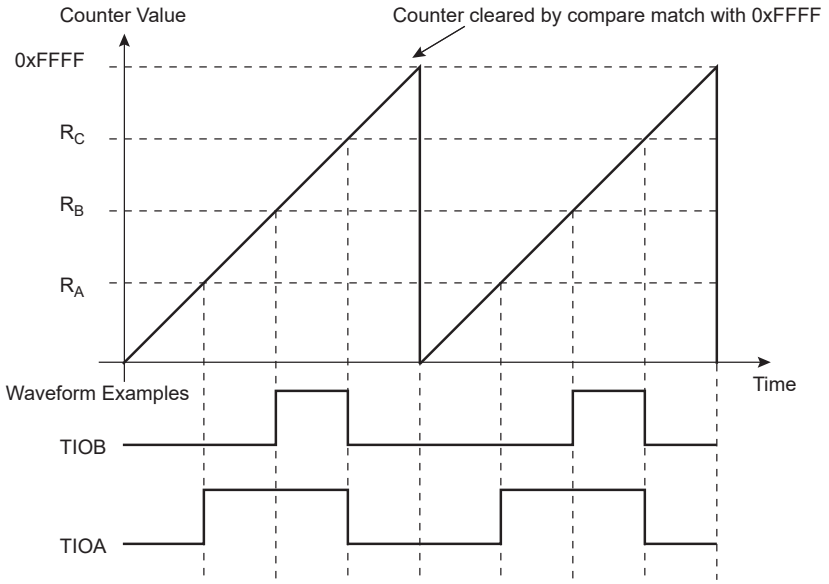
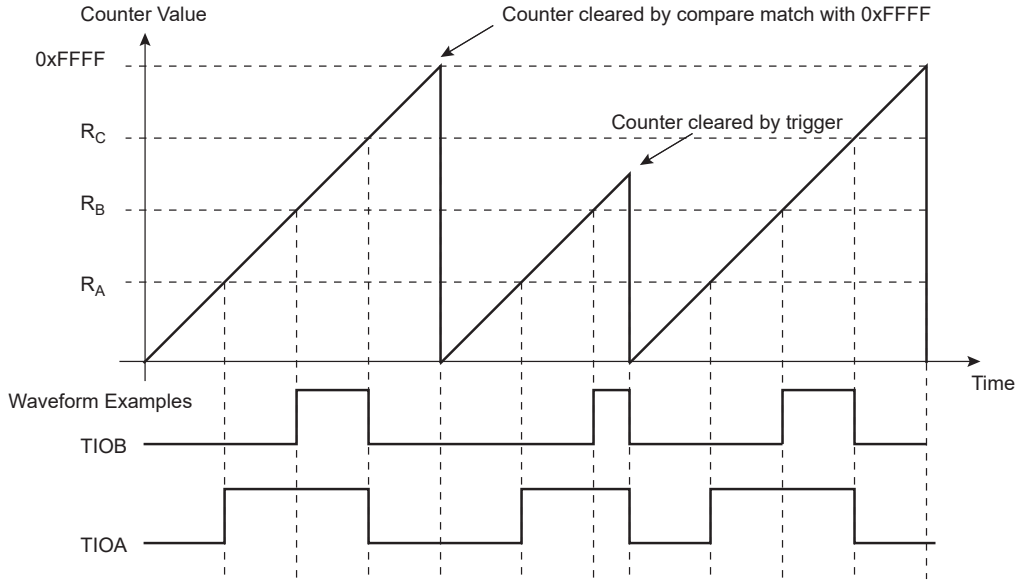



**Figure 50-9. WAVSEL = 00 with Trigger**




### 50.6.12.2 WAVSEL = 10

When WAVSEL = 10, the value of TC_CV is incremented from 0 to the value of RC, then automatically reset on a RC Compare. Once the value of TC_CV has been reset, it is then incremented and so on.

It is important to note that TC_CV can be reset at any time by an external event or a software trigger if both are programmed correctly.

Refer to the figures below.

In addition, RC Compare can stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

**Figure 50-10. WAVSEL = 10 without Trigger**

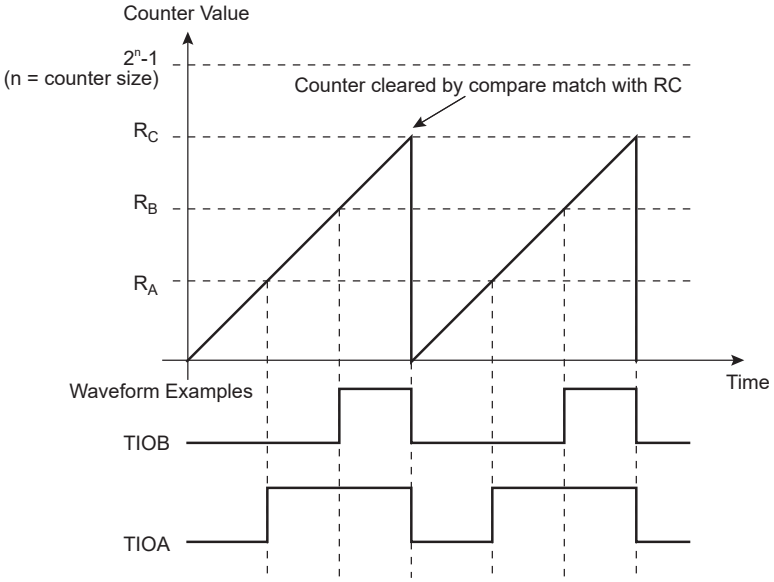
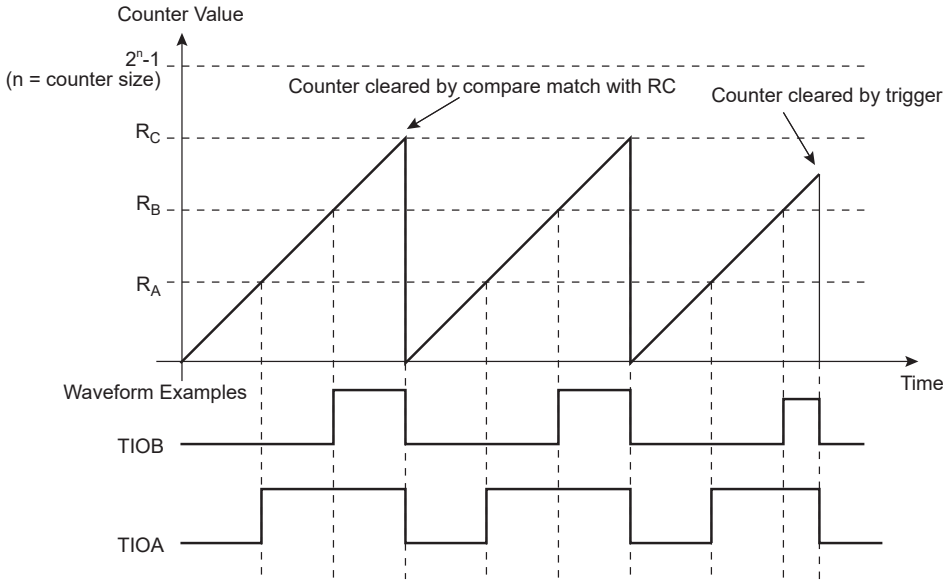



**Figure 50-11. WAVSEL = 10 with Trigger**




### 50.6.12.3 WAVSEL = 01

When WAVSEL = 01, the value of TC_CV is incremented from 0 to $2^{16}$-1 . Once $2^{16}$-1 is reached, the value of TC_CV is decremented to 0, then reincremented to $2^{16}$-1 and so on.

A trigger such as an external event or a software trigger can modify TC_CV at any time. If a trigger occurs while TC_CV is incrementing, TC_CV then decrements. If a trigger is received while TC_CV is decrementing, TC_CV then increments.

Refer to the figures below.

RC Compare cannot be programmed to generate a trigger in this configuration.

At the same time, RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

**Figure 50-12. WAVSEL = 01 without Trigger**

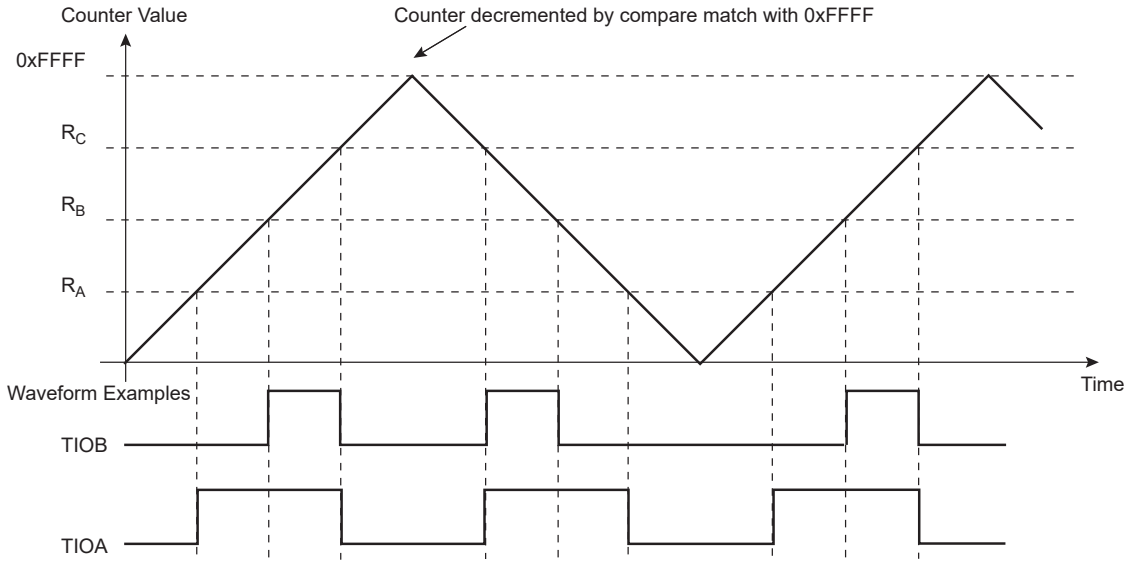
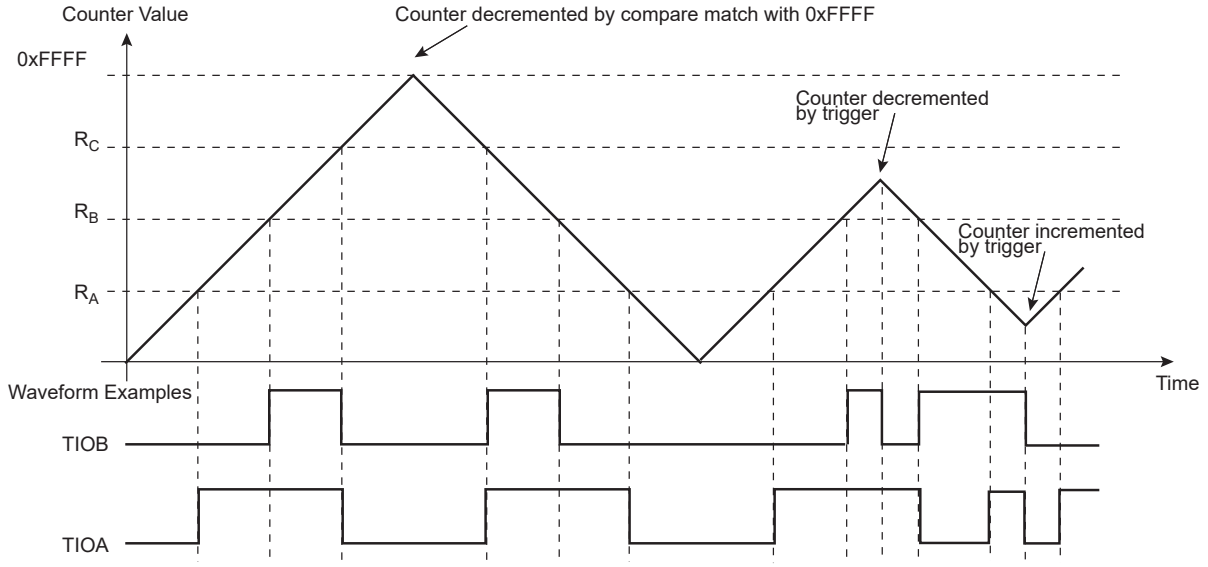



**Figure 50-13. WAVSEL = 01 with Trigger**




### 50.6.12.4 WAVSEL = 11

When WAVSEL = 11, the value of TC_CV is incremented from 0 to RC. Once RC is reached, the value of TC_CV is decremented to 0, then reincremented to RC and so on.

A trigger such as an external event or a software trigger can modify TC_CV at any time. If a trigger occurs while TC_CV is incrementing, TC_CV then decrements. If a trigger is received while TC_CV is decrementing, TC_CV then increments.

Refer to the figures below.

RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

**Figure 50-14. WAVSEL = 11 without Trigger**

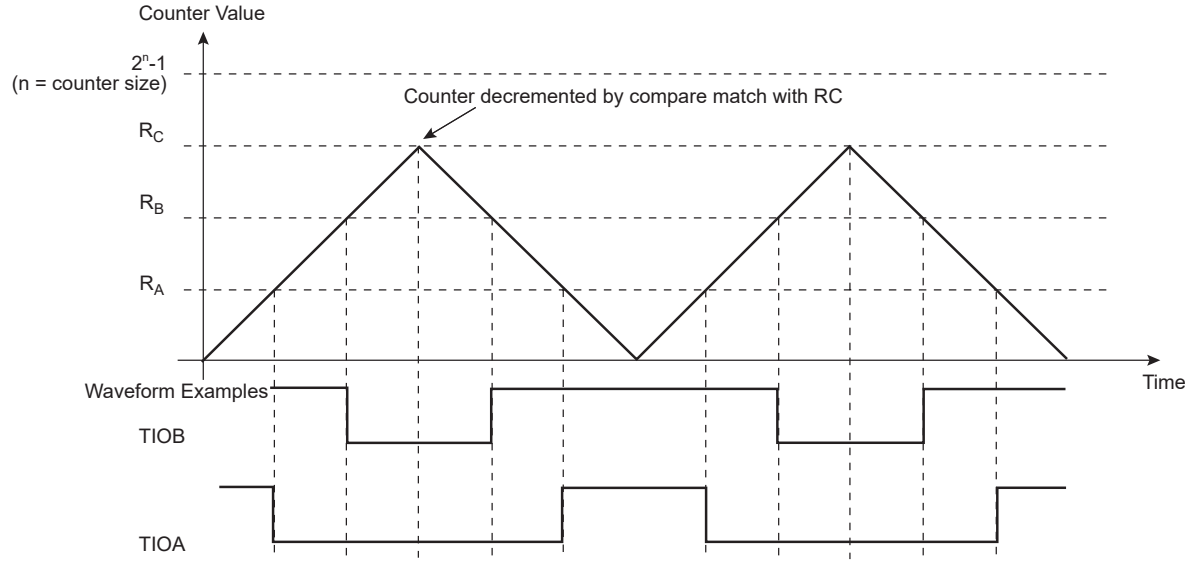
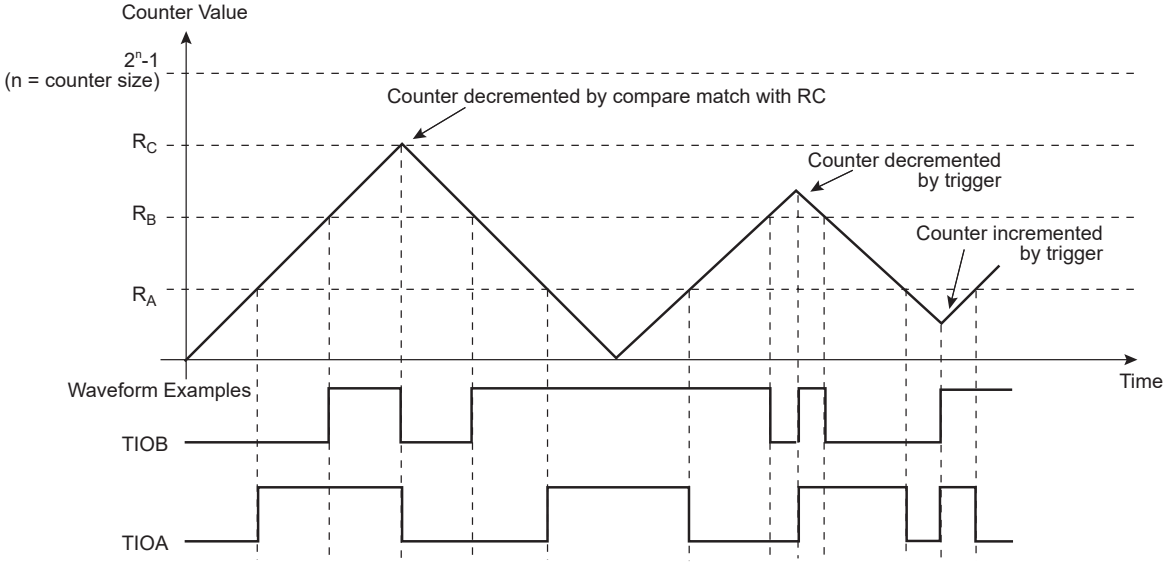



**Figure 50-15. WAVSEL = 11 with Trigger**




### 50.6.13 External Event/Trigger Conditions

An external event can be programmed to be detected on one of the clock sources (XC0, XC1, XC2) or TIOBx. The external event selected can then be used as a trigger.

The event trigger is selected using TC_CMR.EEVT. The trigger edge (rising, falling or both) for each of the possible external triggers is defined in TC_CMR.EEVTEDG. If EEVTEDG is cleared (none), no external event is defined.

If TIOBx is defined as an external event signal (EEVT = 0), TIOBx is no longer used as an output and the compare register B is not used to generate waveforms and subsequently no IRQs. In this case, the TC channel can only generate a waveform on TIOAx.

When an external event is defined, it can be used as a trigger by setting TC_CMR.ENETRG.

As in Capture mode, the SYNC signal and the software trigger are also available as triggers. RC Compare can also be used as a trigger depending on the parameter WAVSEL.

### 50.6.14 Synchronization with PWM

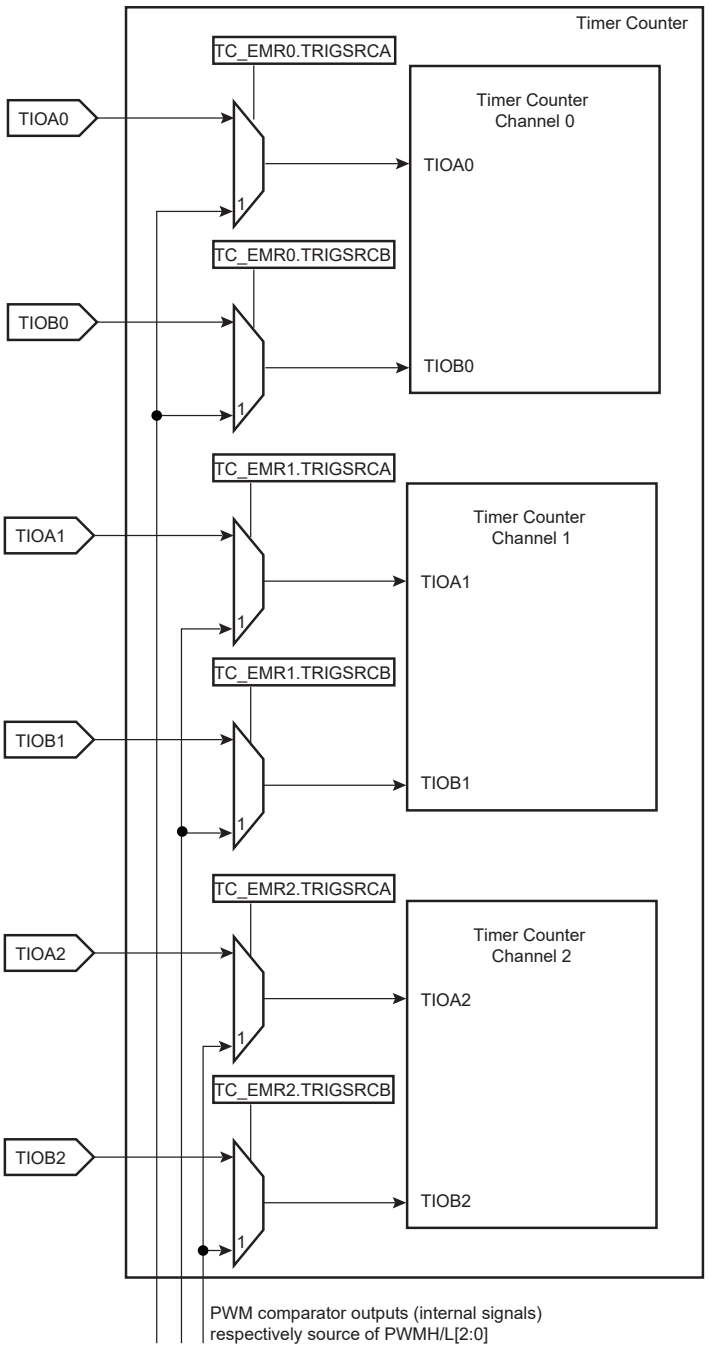
The inputs TIOAx/TIOBx can be bypassed, and thus channel trigger/capture events can be directly driven by the independent PWM module.

PWM comparator outputs (internal signals without dead-time insertion - OCx), respectively source of the PWMH/L[2:0] outputs, are routed to the internal TC inputs. These specific TC inputs are multiplexed with TIOA/B input signal to drive the internal trigger/capture events.

The selection is made in the Extended Mode register (TC_EMR) fields TRIGSRCA and TRIGSRCB (see "TC Extended Mode Register").

Each channel of the TC module can be synchronized by a different PWM channel as described in the following figure.

**Figure 50-16. Synchronization with PWM**



### 50.6.15 Output Controller

The output controller defines the output level changes on TIOAx and TIOBx following an event. TIOBx control is used only if TIOBx is defined as output (not as an external event).

The following events control TIOAx and TIOBx:

- Software trigger
- External event
- RC compare

RA Compare controls TIOAx, and RB Compare controls TIOBx. Each of these events can be programmed to set, clear or toggle the output as defined in the corresponding parameter in TC_CMR.

### 50.6.16 Quadrature Decoder

#### 50.6.16.1 Description

The quadrature decoder (QDEC) is driven by TIOA0, TIOB0 and TIOB1 input pins and drives the timer counter of channel 0 and 1. Channel 2 can be used as a time base in case of speed measurement requirements (refer to Predefined Connection of the Quadrature Decoder with Timer Counters).

When writing a ‘0’ to TC_BMR.QDEN, the QDEC is bypassed and the IO pins are directly routed to the timer counter function.

TIOA0 and TIOB0 are to be driven by the two dedicated quadrature signals from a rotary sensor mounted on the shaft of the off-chip motor.

A third signal from the rotary sensor can be processed through pin TIOB1 and is typically dedicated to be driven by an index signal if it is provided by the sensor. This signal is not required to decode the quadrature signals PHA, PHB.

TC_CMRx.TCCLKS must be configured to select XC0 input (i.e., 0x101). Field TC0XC0S has no effect as soon as the QDEC is enabled.

Either speed or position/revolution can be measured. Position channel 0 accumulates the edges of PHA, PHB input signals giving a high accuracy on motor position whereas channel 1 accumulates the index pulses of the sensor, therefore the number of rotations. Concatenation of both values provides a high level of precision on motion system position.

In Speed mode, position cannot be measured but revolution can be measured.

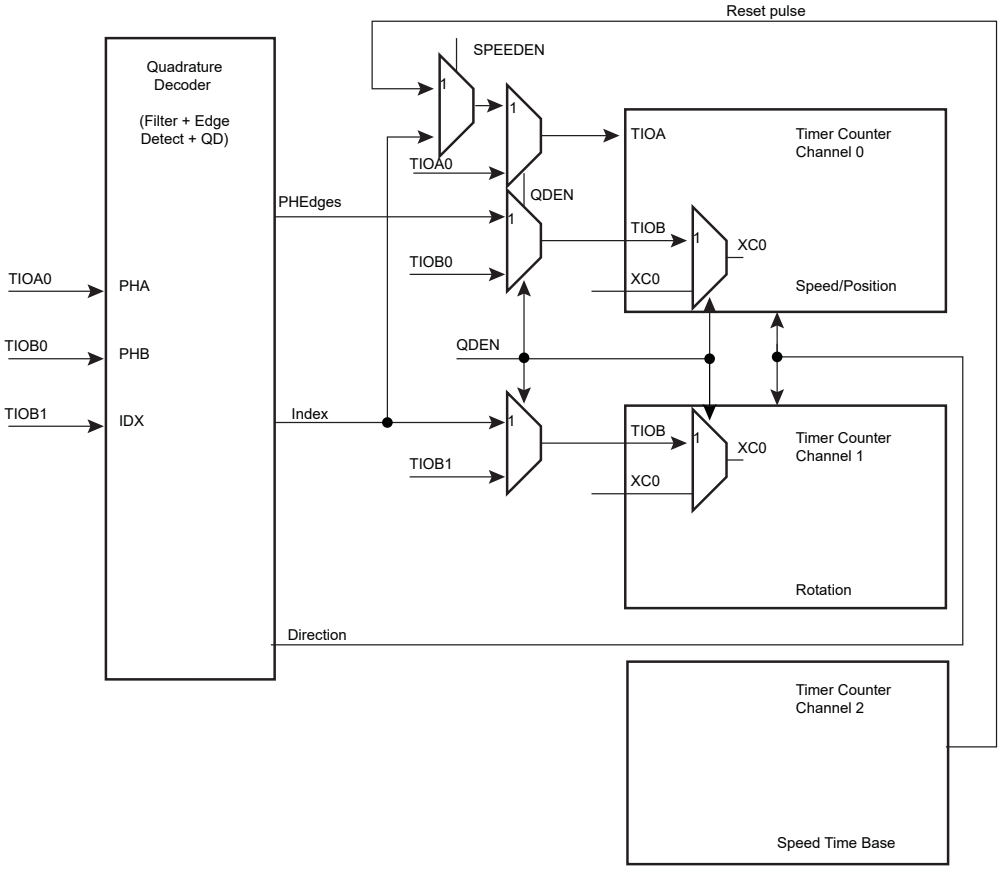
Inputs from the rotary sensor can be filtered prior to downstream processing. Accommodation of input polarity, phase definition and other factors are configurable.

Interruptions can be generated on different events.

A compare function (using TC_RC) is available on channel 0 (speed/position) or channel 1 (rotation) and can generate an interrupt by means of TC_SRx.CPCS.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 50-17. Predefined Connection of the Quadrature Decoder with Timer Counters**




### 50.6.16.2 Input Preprocessing

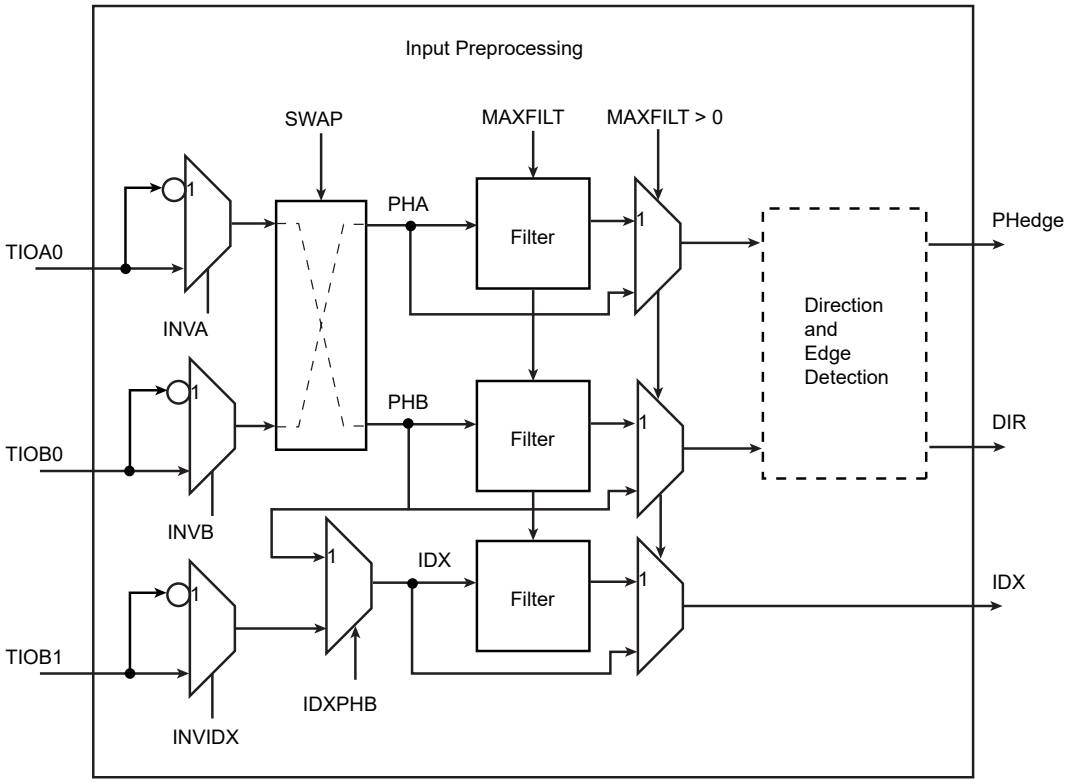
Input preprocessing consists of capabilities to take into account rotary sensor factors such as polarities and phase definition followed by configurable digital filtering.

Each input can be negated and swapping PHA, PHB is also configurable.

TC_BMR. MAXFILT is used to configure a minimum duration for which the pulse is stated as valid. When the filter is active, pulses with a duration lower than (MAXFILT +1) × $t_{peripheral\ clock}$ are not passed to downstream logic.

The value of (MAXFILT +1) × $t_{peripheral\ clock}$ must not be greater than 10% of the minimum pulse on PHA, PHB or index when the rotary encoder speed is at its maximum. This speed depends on the application.

**Figure 50-18. Input Stage**

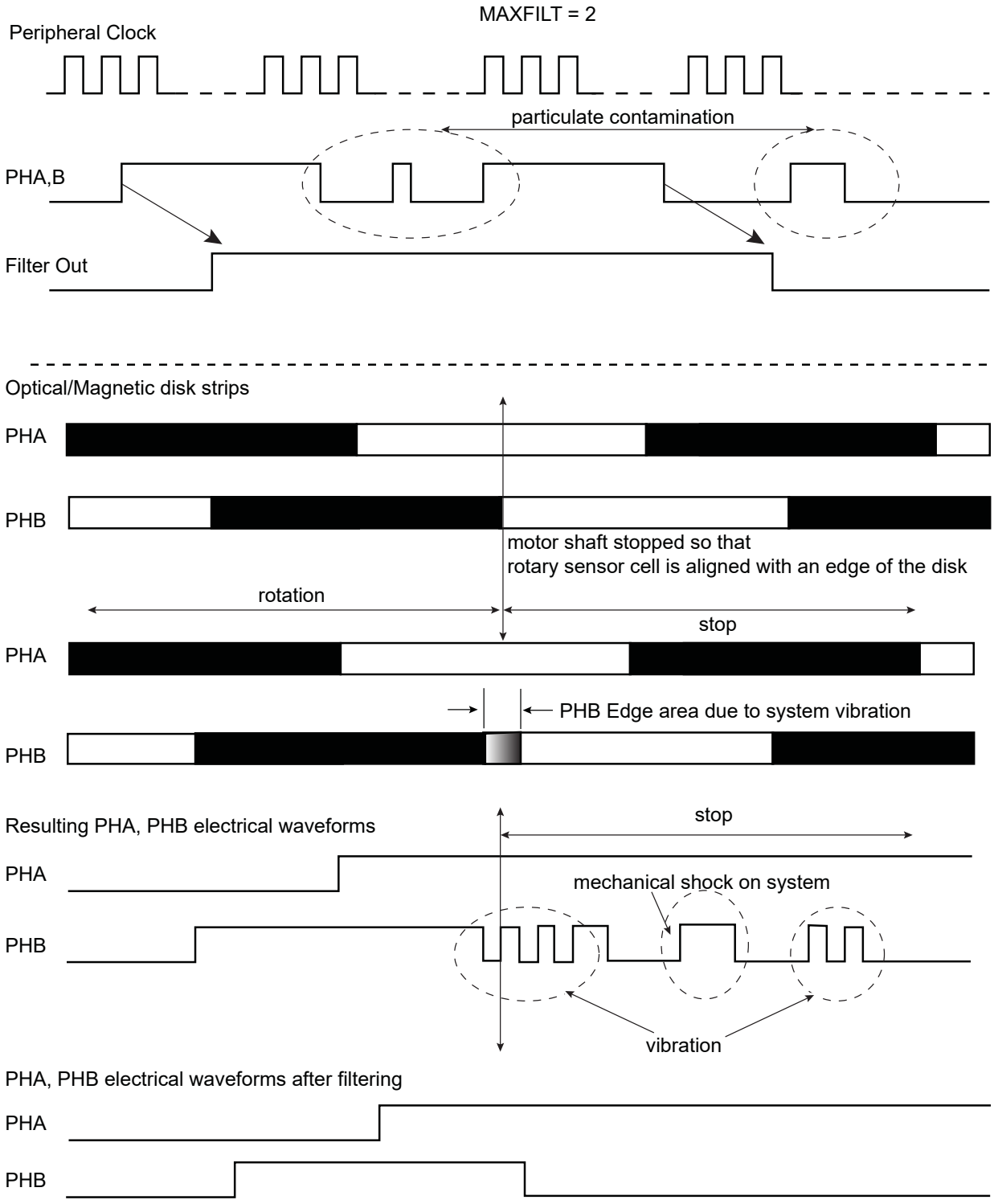



Input filtering can efficiently remove spurious pulses that might be generated by the presence of particulate contamination on the optical or magnetic disk of the rotary sensor.

Spurious pulses can also occur in environments with high levels of electromagnetic interference. Or, simply if vibration occurs even when rotation is fully stopped and the shaft of the motor is in such a position that the beginning of one of the reflective or magnetic bars on the rotary sensor disk is aligned with the light or magnetic (Hall) receiver cell of the rotary sensor. Any vibration can make the PHA, PHB signals toggle for a short duration.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 50-19. Filtering Examples**




### 50.6.16.3 Direction Status and Change Detection

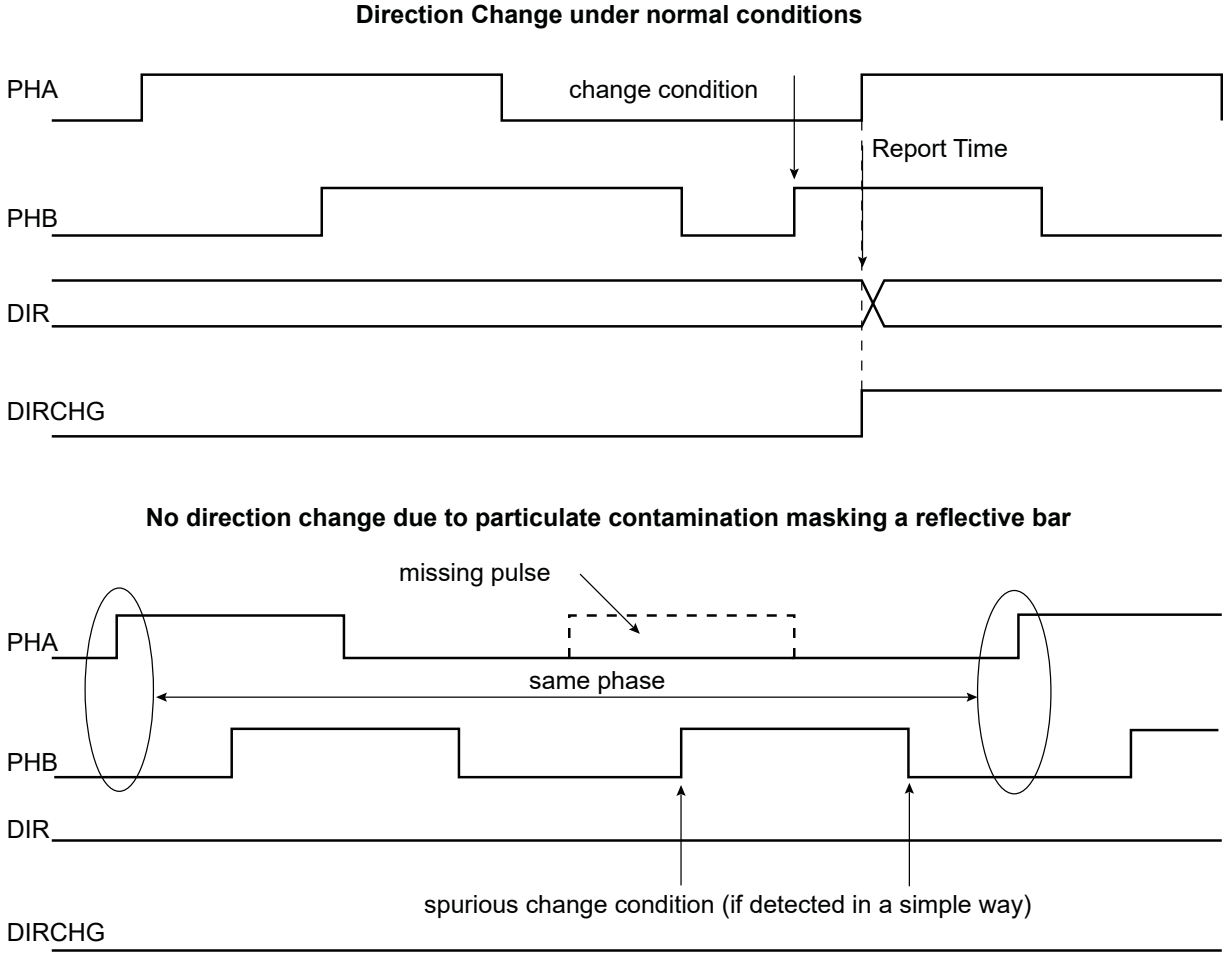
After filtering, the quadrature signals are analyzed to extract the rotation direction and edges of the two quadrature signals detected in order to be counted by TC logic downstream.

The direction status can be directly read at anytime in the TC_QISR. The polarity of the direction flag status depends on the configuration written in TC_BMR. INVA, INVB, INVIDX, SWAP modify the polarity of DIR flag.

Any change in rotation direction is reported in the TC_QISR and can generate an interrupt.

The direction change condition is reported as soon as two consecutive edges on a phase signal have sampled the same value on the other phase signal and there is an edge on the other signal. The two consecutive edges of one phase signal sampling the same value on other phase signal is not sufficient to declare a direction change, as particulate contamination may mask one or more reflective bars on the optical or magnetic disk of the sensor. Refer to the following figure for waveforms.

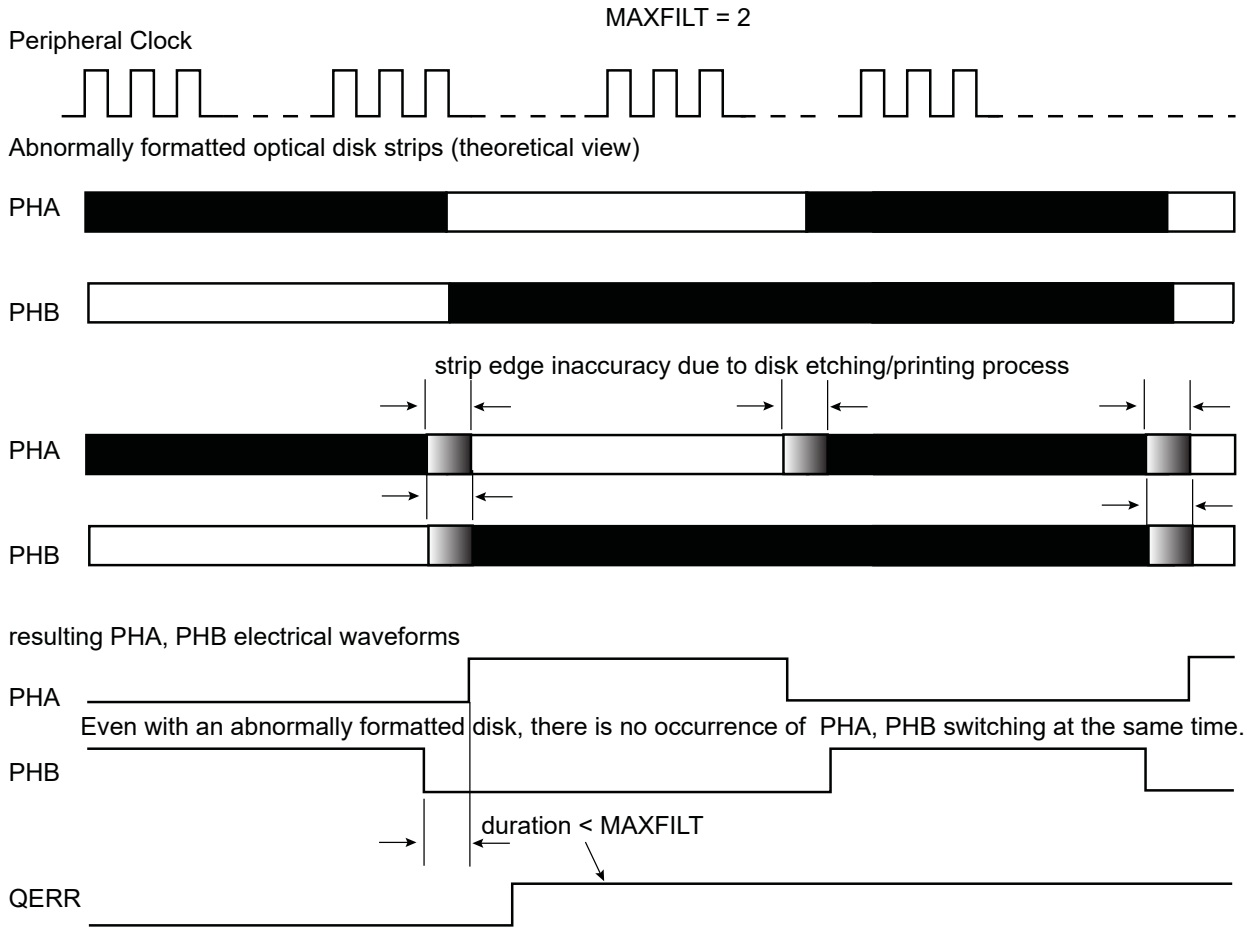
**Figure 50-20. Rotation Change Detection**




The direction change detection is disabled when TC_BMR.QDTRANS is set. In this case, the DIR flag report must not be used.

A quadrature error is also reported by the QDEC via TC_QISR.QERR. This error is reported if the time difference between two edges on PHA, PHB is lower than a predefined value. This predefined value is configurable and corresponds to (TC_BMR.MAXFILT + 1) × $t_{peripheral\ clock}$ ns. After being filtered, there is no reason to have two edges closer than (TC_BMR.MAXFILT + 1) × $t_{peripheral\ clock}$ ns under normal mode of operation.

**Figure 50-21. Quadrature Error Detection**




MAXFILT must be tuned according to several factors such as the peripheral clock frequency, type of rotary sensor and rotation speed to be achieved.

### 50.6.16.4 Position and Rotation Measurement

When TC_BMR.POSEN is set, the motor axis position is processed on channel 0 (by means of the PHA, PHB edge detections) and the number of motor revolutions are recorded on channel 1 if the IDX signal is provided on the TIOB1 input. If no IDX signal is available, the internal counter can be cleared for each revolution if the number of counts per revolution is configured in TC_RC0.RC and the TC_CMR.CPCTRG bit is written to '1'. The position measurement can be read in the TC_CV0 register and the rotation measurement can be read in the TC_CV1 register.

Channel 0 and 1 must be configured in Capture mode (TC_CMR0.WAVE = 0). 'Rising edge' must be selected as the External Trigger Edge (TC_CMR.ETRGEDG = 0x01) and 'TIOAx' must be selected as the External Trigger (TC_CMR.ABETRG = 0x1). The process must be started by configuring TC_CCR.CLKEN and TC_CCR.SWTRG.

In parallel, the number of edges are accumulated on TC channel 0 and can be read on the TC_CV0 register.

Therefore, the accurate position can be read on both TC_CV registers and concatenated to form a 32-bit word.

The TC channel 0 is cleared for each increment of IDX count value.

Depending on the quadrature signals, the direction is decoded and allows to count up or down in TC channels 0 and 1. The direction status is reported on TC_QISR.

### 50.6.16.5 Speed Measurement

When TC_BMR.SPEEDEN is set, the speed measure is enabled on channel 0.

A time base must be defined on channel 2 by writing the TC_RC2 period register. Channel 2 must be configured in Waveform mode (WAVE bit set) in TC_CMR2. The WAVSEL field must be defined with 0x10 to clear the counter by comparison and matching with TC_RC value. Field ACPC must be defined at 0x11 to toggle TIOAx output.

© 2021 Microchip Technology Inc.
and its subsidiaries

This time base is automatically fed back to TIOAx of channel 0 when QDEN and SPEEDEN are set.

Channel 0 must be configured in Capture mode (WAVE = 0 in TC_CMR0). TC_CMR0.ABETRG must be configured at 1 to select TIOAx as a trigger for this channel.

EDGTRG must be set to 0x01, to clear the counter on a rising edge of the TIOAx signal and field LDRA must be set accordingly to 0x01, to load TC_RA0 at the same time as the counter is cleared (LDRB must be set to 0x01). As a consequence, at the end of each time base period the differentiation required for the speed calculation is performed.

The process must be started by configuring bits CLKEN and SWTRG in the TC_CCR.

The speed can be read on field RA in TC_RA0.

Channel 1 can still be used to count the number of revolutions of the motor.

#### 50.6.16.6 Detecting a Missing Index Pulse

To detect a missing index pulse due contamination, dust, etc., the TC_SR0.CPCS flag can be used. It is also possible to assert the interrupt line if the TC_SR0.CPCS flag is enabled as a source of the interrupt by writing a ‘1’ to TC_IER0.CPCS.

The TC_RC0.RC field must be written with the nominal number of counts per revolution provided by the rotary encoder, plus a margin to eliminate potential noise (e.g., if nominal count per revolution is 1024, then TC_RC0.RC=1026).

If the index pulse is missing, the timer value is not cleared and the nominal value is exceeded, then the comparator on the RC triggers an event, TC_SR0.CPCS=1, and the interrupt line is asserted if TC_IER0.CPCS=1.

The missing index pulse detection is only valid if the bit TC_QISR.DIRCHG=0.

#### 50.6.16.7 Detecting Contamination/Dust at Rotary Encoder Low Speed

The contamination/dust that can be filtered when the rotary encoder speed is high may not be filtered at low speed, thus creating unsollicited direction change, etc.

At low speed, even a minor contamination may appear as a long pulse, and thus not filtered and processed as a standard quadrature encoder pulse.

This contamination can be detected by using the similar method as the missing index detection.

A contamination exists on a phase line if TC_SR.CPCS = 1 and TC_QISR.DIRCHG = 1 when there is no sollicited change of direction.

#### 50.6.16.8 Missing Pulse Detection and Autocorrection

The QDEC is equipped with a circuitry which detects and corrects some errors that may result from contamination on optical disks or other materials producing the quadrature phase signals.

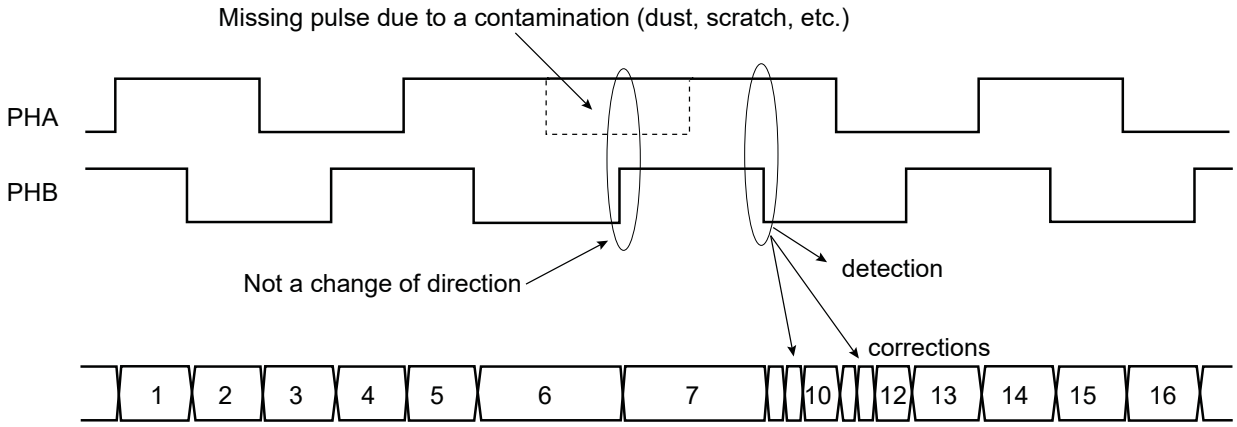
The detection and autocorrection only works if the Count mode is configured for both phases (EDGPHA = 1 in TC_BMR) and is enabled (AUTOC = 1 in TC_BMR).

If a pulse is missing on a phase signal, it is automatically detected and the pulse count reported in the CV field of the TC_CV0/1 is automatically corrected.

There is no detection if both phase signals are affected at the same location on the device providing the quadrature signals because the detection requires a valid phase signal to detect the contamination on the other phase signal.

© 2021 Microchip Technology Inc. and its subsidiaries

**Figure 50-22. Detection and Autocorrection of Missing Pulses**




If a quadrature device is undamaged, the number of pulses counted for a predefined period of time must be the same with or without detection and autocorrection feature.

Therefore, if the measurement results differ, a contamination exists on the device producing the quadrature signals.

This does not substitute the measurements of the number of pulses between two index pulses (if available) but provides a complementary method to detect damaged quadrature devices.

When the device providing quadrature signals is severely damaged, potentially leading to a number of consecutive missing pulses greater than 1, the downstream processing may be affected. It is possible to define the maximum admissible number of consecutive missing pulses before issuing a Missing Pulse Error flag (MPE in TC_QISR). The threshold triggering an MPE flag report can be configured in TC_BMR.MAXCMP. If the field MAXCMP is cleared, MPE never rises. The flag MAXCMP can trigger an interrupt while the QDEC is operating, thus providing a real time report of a potential problem on the quadrature device.

### 50.6.17 2-bit Gray Up/Down Counter for Stepper Motor

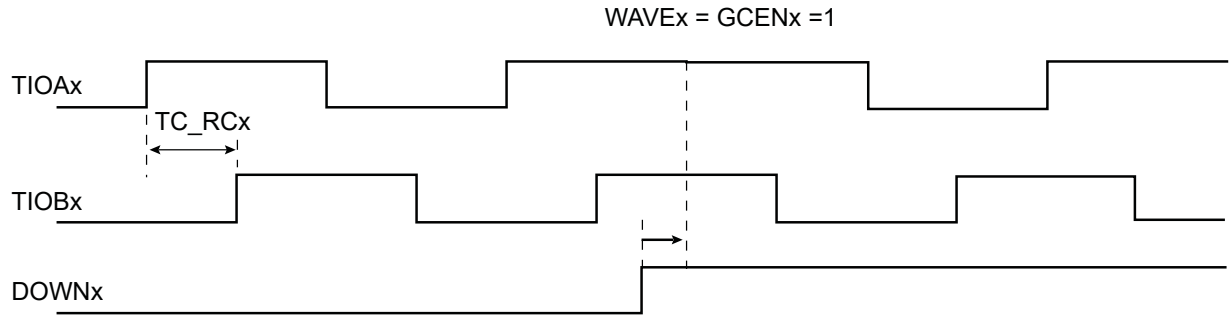
Each channel can be independently configured to generate a 2-bit Gray count waveform on corresponding TIOAx, TIOBx outputs by means of TC_SMMRx.GCEN.

Up or Down count can be defined by writing TC_SMMRx.DOWN.

It is mandatory to configure the channel in Waveform mode in the TC_CMR.

The period of the counters can be programmed in TC_RCx.

**Figure 50-23. 2-bit Gray Up/Down Counter**




### 50.6.18 Fault Mode

At any time, the TC_RCx registers can be used to perform a comparison on the respective current channel counter value (TC_CVx) with the value of TC_RCx register.

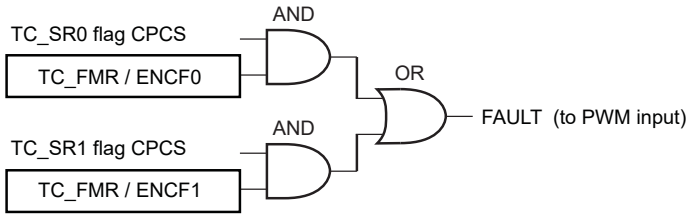
The CPCSx flags can be set accordingly and an interrupt can be generated.

This interrupt is processed but requires an unpredictable amount of time to be achieve the required action.

It is possible to trigger the FAULT output of the TIMER1 with CPCS from TC_SR0 and/or CPCS from TC_SR1. Each source can be independently enabled/disabled in the TC_FMR.

This can be useful to detect an overflow on speed and/or position when QDEC is processed and to act immediately by using the FAULT output.

**Figure 50-24. Fault Output Generation**




### 50.6.19 Register Write Protection

To prevent any single software error from corrupting TC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the TC Write Protection Mode Register (TC_WPMR).

The Timer Counter clock of the first channel must be enabled to access TC_WPMR.

The following registers can be write-protected when WPEN is set:

- TC Block Mode Register
- TC Channel Mode Register Capture Mode
- TC Channel Mode Register Waveform Mode
- TC Fault Mode Register
- TC Stepper Motor Mode Register
- TC Register A
- TC Register B
- TC Register C
- TC Extended Mode Register

## 50.7 Register Summary

**Note:** The register TC_CMR has two modes, Capture Mode and Waveform Mode. In this register summary, both modes are displayed

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | TC_CCR0 | 7:0 | | | | | | SWTRG | CLKDIS | CLKEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | TC_CMR0 | 7:0 | LDBDIS | LDBSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | CPCTRG | | | | ABETRG | ETRGEDG[1:0] | |
| | | 23:16 | | SBSMPLR[2:0] | | | LDRB[1:0] | | LDRA[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x04 | TC_CMR0 | 7:0 | CPCDIS | CPCSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | WAVSEL[1:0] | | ENETRG | EEVT[1:0] | | EEVTEDG[1:0] | |
| | | 23:16 | ASWTRG[1:0] | | AEEVT[1:0] | | ACPC[1:0] | | ACPA[1:0] | |
| | | 31:24 | BSWTRG[1:0] | | BEEVT[1:0] | | BCPC[1:0] | | BCPB[1:0] | |
| 0x08 | TC_SMMR0 | 7:0 | | | | | | | DOWN | GCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | TC_RAB0 | 7:0 | RAB[7:0] | | | | | | | |
| | | 15:8 | RAB[15:8] | | | | | | | |
| | | 23:16 | RAB[23:16] | | | | | | | |
| | | 31:24 | RAB[31:24] | | | | | | | |
| 0x10 | TC_CV0 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | CV[31:24] | | | | | | | |
| 0x14 | TC_RA0 | 7:0 | RA[7:0] | | | | | | | |
| | | 15:8 | RA[15:8] | | | | | | | |
| | | 23:16 | RA[23:16] | | | | | | | |
| | | 31:24 | RA[31:24] | | | | | | | |
| 0x18 | TC_RB0 | 7:0 | RB[7:0] | | | | | | | |
| | | 15:8 | RB[15:8] | | | | | | | |
| | | 23:16 | RB[23:16] | | | | | | | |
| | | 31:24 | RB[31:24] | | | | | | | |
| 0x1C | TC_RC0 | 7:0 | RC[7:0] | | | | | | | |
| | | 15:8 | RC[15:8] | | | | | | | |
| | | 23:16 | RC[23:16] | | | | | | | |
| | | 31:24 | RC[31:24] | | | | | | | |
| 0x20 | TC_SR0 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | MTIOB | MTIOA | CLKSTA |
| | | 31:24 | | | | | | | | |
| 0x24 | TC_IER0 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | TC_IDR0 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | TC_IMR0 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x30 | TC_EMR0 | 7:0 | | | TRIGSRCB[1:0] | | | | TRIGSRCA[1:0] | |
| | | 15:8 | | | | | | | | NODIVCLK |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 ... 0x3F | Reserved | | | | | | | | | |
| 0x40 | TC_CCR1 | 7:0 | | | | | | SWTRG | CLKDIS | CLKEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x44 | TC_CMR1 | 7:0 | LDBDIS | LDBSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | CPCTRG | | | | ABETRG | ETRGEDG[1:0] | |
| | | 23:16 | | SBSMPLR[2:0] | | | LDRB[1:0] | | LDRA[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x44 | TC_CMR1 | 7:0 | CPCDIS | CPCSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | WAVSEL[1:0] | | ENETRG | EEVT[1:0] | | EEVTEDG[1:0] | |
| | | 23:16 | ASWTRG[1:0] | | AEEVT[1:0] | | ACPC[1:0] | | ACPA[1:0] | |
| | | 31:24 | BSWTRG[1:0] | | BEEVT[1:0] | | BCPC[1:0] | | BCPB[1:0] | |
| 0x48 | TC_SMMR1 | 7:0 | | | | | | | DOWN | GCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x4C | TC_RAB1 | 7:0 | RAB[7:0] | | | | | | | |
| | | 15:8 | RAB[15:8] | | | | | | | |
| | | 23:16 | RAB[23:16] | | | | | | | |
| | | 31:24 | RAB[31:24] | | | | | | | |
| 0x50 | TC_CV1 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | CV[31:24] | | | | | | | |
| 0x54 | TC_RA1 | 7:0 | RA[7:0] | | | | | | | |
| | | 15:8 | RA[15:8] | | | | | | | |
| | | 23:16 | RA[23:16] | | | | | | | |
| | | 31:24 | RA[31:24] | | | | | | | |
| 0x58 | TC_RB1 | 7:0 | RB[7:0] | | | | | | | |
| | | 15:8 | RB[15:8] | | | | | | | |
| | | 23:16 | RB[23:16] | | | | | | | |
| | | 31:24 | RB[31:24] | | | | | | | |
| 0x5C | TC_RC1 | 7:0 | RC[7:0] | | | | | | | |
| | | 15:8 | RC[15:8] | | | | | | | |
| | | 23:16 | RC[23:16] | | | | | | | |
| | | 31:24 | RC[31:24] | | | | | | | |
| 0x60 | TC_SR1 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | MTIOB | MTIOA | CLKSTA |
| | | 31:24 | | | | | | | | |
| 0x64 | TC_IER1 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x68 | TC_IDR1 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x6C | TC_IMR1 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x70 | TC_EMR1 | 7:0 | | | TRIGSRCB[1:0] | | | | TRIGSRCA[1:0] | |
| | | 15:8 | | | | | | | | NODIVCLK |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 ... 0x7F | Reserved | | | | | | | | | |
| 0x80 | TC_CCR2 | 7:0 | | | | | | SWTRG | CLKDIS | CLKEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 | TC_CMR2 | 7:0 | LDBDIS | LDBSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | CPCTRG | | | | ABETRG | ETRGEDG[1:0] | |
| | | 23:16 | | SBSMPLR[2:0] | | | LDRB[1:0] | | LDRA[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x84 | TC_CMR2 | 7:0 | CPCDIS | CPCSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| | | 15:8 | WAVE | WAVSEL[1:0] | | ENETRG | EEVT[1:0] | | EEVTEDG[1:0] | |
| | | 23:16 | ASWTRG[1:0] | | AEEVT[1:0] | | ACPC[1:0] | | ACPA[1:0] | |
| | | 31:24 | BSWTRG[1:0] | | BEEVT[1:0] | | BCPC[1:0] | | BCPB[1:0] | |
| 0x88 | TC_SMMR2 | 7:0 | | | | | | | DOWN | GCEN |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x8C | TC_RAB2 | 7:0 | RAB[7:0] | | | | | | | |
| | | 15:8 | RAB[15:8] | | | | | | | |
| | | 23:16 | RAB[23:16] | | | | | | | |
| | | 31:24 | RAB[31:24] | | | | | | | |
| 0x90 | TC_CV2 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | CV[31:24] | | | | | | | |
| 0x94 | TC_RA2 | 7:0 | RA[7:0] | | | | | | | |
| | | 15:8 | RA[15:8] | | | | | | | |
| | | 23:16 | RA[23:16] | | | | | | | |
| | | 31:24 | RA[31:24] | | | | | | | |
| 0x98 | TC_RB2 | 7:0 | RB[7:0] | | | | | | | |
| | | 15:8 | RB[15:8] | | | | | | | |
| | | 23:16 | RB[23:16] | | | | | | | |
| | | 31:24 | RB[31:24] | | | | | | | |
| 0x9C | TC_RC2 | 7:0 | RC[7:0] | | | | | | | |
| | | 15:8 | RC[15:8] | | | | | | | |
| | | 23:16 | RC[23:16] | | | | | | | |
| | | 31:24 | RC[31:24] | | | | | | | |
| 0xA0 | TC_SR2 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | MTIOB | MTIOA | CLKSTA |
| | | 31:24 | | | | | | | | |
| 0xA4 | TC_IER2 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA8 | TC_IDR2 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xAC | TC_IMR2 | 7:0 | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xB0 | TC_EMR2 | 7:0 | | | TRIGSRCB[1:0] | | | | TRIGSRCA[1:0] | |
| | | 15:8 | | | | | | | | NODIVCLK |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xB4 ... 0xBF | Reserved | | | | | | | | | |
| 0xC0 | TC_BCR | 7:0 | | | | | | | | SYNC |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xC4 | TC_BMR | 7:0 | | | TC2XC2S[1:0] | | TC1XC1S[1:0] | | TC0XC0S[1:0] | |
| | | 15:8 | INVIDX | INVB | INVA | EDGPHA | QDTRANS | SPEEDEN | POSEN | QDEN |
| | | 23:16 | MAXFILT[3:0] | | | | | AUTOC | IDXPHB | SWAP |
| | | 31:24 | | | MAXCMP[3:0] | | | | MAXFILT[5:4] | |
| 0xC8 | TC_QIER | 7:0 | | | | | MPE | QERR | DIRCHG | IDX |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xCC | TC_QIDR | 7:0 | | | | | MPE | QERR | DIRCHG | IDX |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD0 | TC_QIMR | 7:0 | | | | | MPE | QERR | DIRCHG | IDX |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD4 | TC_QISR | 7:0 | | | | | MPE | QERR | DIRCHG | IDX |
| | | 15:8 | | | | | | | | DIR |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD8 | TC_FMR | 7:0 | | | | | | | ENCF1 | ENCF0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xDC ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | TC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |

### 50.7.1 TC Channel Control Register

**Name:** TC_CCRx
**Offset:** 0x00 + x*0x40 [x=0..2]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | SWTRG | CLKDIS | CLKEN |
| Access | | | | | | W | W | W |
| Reset | | | | | | – | – | – |

**Bit 2 – SWTRG** Software Trigger Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | A software trigger is performed: the counter is reset and the clock is started. |

**Bit 1 – CLKDIS** Counter Clock Disable Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the clock. |

**Bit 0 – CLKEN** Counter Clock Enable Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the clock if CLKDIS is not 1. |

### 50.7.2 TC Channel Mode Register: Capture Mode

**Name:** TC_CMRx
**Offset:** 0x04 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register can be written only if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | SBSMPLR[2:0] | | | LDRB[1:0] | | LDRA[1:0] | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WAVE | CPCTRG | | | | ABETRG | ETRGEDG[1:0] | |
| Access | R/W | R/W | | | | R/W | R/W | R/W |
| Reset | 0 | 0 | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LDBDIS | LDBSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 22:20 – SBSMPLR[2:0]** Loading Edge Subsampling Ratio

| Value | Name | Description |
|---|---|---|
| 0 | ONE | Load a Capture register each selected edge. |
| 1 | HALF | Load a Capture register every 2 selected edges. |
| 2 | FOURTH | Load a Capture register every 4 selected edges. |
| 3 | EIGHTH | Load a Capture register every 8 selected edges. |
| 4 | SIXTEENTH | Load a Capture register every 16 selected edges. |

**Bits 19:18 – LDRB[1:0]** RB Loading Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | RISING | Rising edge of TIOAx |
| 2 | FALLING | Falling edge of TIOAx |
| 3 | EDGE | Each edge of TIOAx |

**Bits 17:16 – LDRA[1:0]** RA Loading Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | RISING | Rising edge of TIOAx |
| 2 | FALLING | Falling edge of TIOAx |
| 3 | EDGE | Each edge of TIOAx |

**Bit 15 – WAVE** Waveform Mode

| Value | Description |
|---|---|
| 0 | Capture mode is enabled. |
| 1 | Capture mode is disabled (Waveform mode is enabled). |

© 2021 Microchip Technology Inc. and its subsidiaries

**Bit 14 – CPCTRG** RC Compare Trigger Enable

| Value | Description |
|---|---|
| 0 | RC Compare has no effect on the counter and its clock. |
| 1 | RC Compare resets the counter and starts the counter clock. |

**Bit 10 – ABETRG** TIOAx or TIOBx External Trigger Selection

| Value | Description |
|---|---|
| 0 | TIOBx is used as an external trigger. |
| 1 | TIOAx is used as an external trigger. |

**Bits 9:8 – ETRGEDG[1:0]** External Trigger Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | The clock is not gated by an external signal. |
| 1 | RISING | Rising edge |
| 2 | FALLING | Falling edge |
| 3 | EDGE | Each edge |

**Bit 7 – LDBDIS** Counter Clock Disable with RB Loading

| Value | Description |
|---|---|
| 0 | Counter clock is not disabled when RB loading occurs. |
| 1 | Counter clock is disabled when RB loading occurs. |

**Bit 6 – LDBSTOP** Counter Clock Stopped with RB Loading

| Value | Description |
|---|---|
| 0 | Counter clock is not stopped when RB loading occurs. |
| 1 | Counter clock is stopped when RB loading occurs. |

**Bits 5:4 – BURST[1:0]** Burst Signal Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | The clock is not gated by an external signal. |
| 1 | XC0 | XC0 is ANDed with the selected clock. |
| 2 | XC1 | XC1 is ANDed with the selected clock. |
| 3 | XC2 | XC2 is ANDed with the selected clock. |

**Bit 3 – CLKI** Clock Invert

| Value | Description |
|---|---|
| 0 | Counter is incremented on rising edge of the clock. |
| 1 | Counter is incremented on falling edge of the clock. |

**Bits 2:0 – TCCLKS[2:0]** Clock Selection

To operate at maximum peripheral clock frequency, refer to “TC Extended Mode Register”.

| Value | Name | Description |
|---|---|---|
| 0 | TIMER_CLOCK1 | Clock selected: internal PCK6 or PCK7 (TC0 only) clock signal (from PMC) |
| 1 | TIMER_CLOCK2 | Clock selected: internal MCK/8 clock signal (from PMC) |
| 2 | TIMER_CLOCK3 | Clock selected: internal MCK/32 clock signal (from PMC) |
| 3 | TIMER_CLOCK4 | Clock selected: internal MCK/128 clock signal (from PMC) |
| 4 | TIMER_CLOCK5 | Clock selected: internal SLCK clock signal (from PMC) |
| 5 | XC0 | Clock selected: XC0 |
| 6 | XC1 | Clock selected: XC1 |
| 7 | XC2 | Clock selected: XC2 |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.3 TC Channel Mode Register: Waveform Mode

**Name:** TC_CMRx
**Offset:** 0x04 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | BSWTRG[1:0] | | BEEVT[1:0] | | BCPC[1:0] | | BCPB[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ASWTRG[1:0] | | AEEVT[1:0] | | ACPC[1:0] | | ACPA[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WAVE | WAVSEL[1:0] | | ENETRG | EEVT[1:0] | | EEVTEDG[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPCDIS | CPCSTOP | BURST[1:0] | | CLKI | TCCLKS[2:0] | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:30 – BSWTRG[1:0]** Software Trigger Effect on TIOBx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 29:28 – BEEVT[1:0]** External Event Effect on TIOBx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 27:26 – BCPC[1:0]** RC Compare Effect on TIOBx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 25:24 – BCPB[1:0]** RB Compare Effect on TIOBx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bits 23:22 – ASWTRG[1:0]** Software Trigger Effect on TIOAx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 21:20 – AEEVT[1:0]** External Event Effect on TIOAx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 19:18 – ACPC[1:0]** RC Compare Effect on TIOAx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bits 17:16 – ACPA[1:0]** RA Compare Effect on TIOAx

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

**Bit 15 – WAVE** Waveform Mode

| Value | Description |
|---|---|
| 0 | Waveform mode is disabled (Capture mode is enabled). |
| 1 | Waveform mode is enabled. |

**Bits 14:13 – WAVSEL[1:0]** Waveform Selection

| Value | Name | Description |
|---|---|---|
| 0 | UP | UP mode without automatic trigger on RC Compare |
| 1 | UPDOWN | UPDOWN mode without automatic trigger on RC Compare |
| 2 | UP_RC | UP mode with automatic trigger on RC Compare |
| 3 | UPDOWN_RC | UPDOWN mode with automatic trigger on RC Compare |

**Bit 12 – ENETRG** External Event Trigger Enable
Whatever the value programmed in ENETRG, the selected external event only controls the TIOAx output and TIOBx if not used as input (trigger event input or other input used).

| Value | Description |
|---|---|
| 0 | The external event has no effect on the counter and its clock. |
| 1 | The external event resets the counter and starts the counter clock. |

**Bits 11:10 – EEVT[1:0]** External Event Selection
Signal selected as external event.

| Value | Name | Description | TIOB Direction |
|---|---|---|---|
| 0 | TIOB | TIOB | Input |
| 1 | XC0 | XC0 | Output |
| 2 | XC1 | XC1 | Output |
| 3 | XC2 | XC2 | Output |

**Note:** If TIOB is chosen as the external event signal, it is configured as an input and no longer generates waveforms and subsequently no IRQs.

**Bits 9:8 – EEVTEDG[1:0]** External Event Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | RISING | Rising edge |
| 2 | FALLING | Falling edge |
| 3 | EDGE | Each edge |

**Bit 7 – CPCDIS** Counter Clock Disable with RC Compare

| Value | Description |
|---|---|
| 0 | Counter clock is not disabled when counter reaches RC. |
| 1 | Counter clock is disabled when counter reaches RC. |

**Bit 6 – CPCSTOP** Counter Clock Stopped with RC Compare

| Value | Description |
|---|---|
| 0 | Counter clock is not stopped when counter reaches RC. |
| 1 | Counter clock is stopped when counter reaches RC. |

**Bits 5:4 – BURST[1:0]** Burst Signal Selection

| Value | Name | Description |
|---|---|---|
| 0 | NONE | The clock is not gated by an external signal. |
| 1 | XC0 | XC0 is ANDed with the selected clock. |
| 2 | XC1 | XC1 is ANDed with the selected clock. |
| 3 | XC2 | XC2 is ANDed with the selected clock. |

**Bit 3 – CLKI** Clock Invert

| Value | Description |
|---|---|
| 0 | Counter is incremented on rising edge of the clock. |
| 1 | Counter is incremented on falling edge of the clock. |

**Bits 2:0 – TCCLKS[2:0]** Clock Selection

To operate at maximum peripheral clock frequency, refer to “TC Extended Mode Register”.

| Value | Name | Description |
|---|---|---|
| 0 | TIMER_CLOCK1 | Clock selected: internal PCK6 or PCK7 (TC0 only) clock signal (from PMC) |
| 1 | TIMER_CLOCK2 | Clock selected: internal MCK/8 clock signal (from PMC) |
| 2 | TIMER_CLOCK3 | Clock selected: internal MCK/32 clock signal (from PMC) |
| 3 | TIMER_CLOCK4 | Clock selected: internal MCK/128 clock signal (from PMC) |
| 4 | TIMER_CLOCK5 | Clock selected: internal SLCK clock signal (from PMC) |
| 5 | XC0 | Clock selected: XC0 |
| 6 | XC1 | Clock selected: XC1 |
| 7 | XC2 | Clock selected: XC2 |

© 2021 Microchip Technology Inc. and its subsidiaries

### 50.7.4 TC Stepper Motor Mode Register

**Name:** TC_SMMRx
**Offset:** 0x08 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** R/W

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DOWN | GCEN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – DOWN** Down Count

| Value | Description |
|---|---|
| 0 | Up counter. |
| 1 | Down counter. |

**Bit 0 – GCEN** Gray Count Enable

| Value | Description |
|---|---|
| 0 | TIOAx [x=0..2] and TIOBx [x=0..2] are driven by internal counter of channel x. |
| 1 | TIOAx [x=0..2] and TIOBx [x=0..2] are driven by a 2-bit Gray counter. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.5 TC Register AB

**Name:** TC_RABx
**Offset:** 0x0C + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RAB[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RAB[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RAB[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RAB[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RAB[31:0]** Register A or Register B
RAB contains the next unread capture Register A or Register B value in real time. It is usually read by the DMA after a request due to a valid load edge on TIOAx.
When DMA is used, the RAB register address must be configured as source address of the transfer.

### 50.7.6 TC Counter Value Register

**Name:** TC_CVx
**Offset:** 0x10 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CV[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CV[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CV[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CV[31:0]** Counter Value
CV contains the counter value in real time.

**Important:**
For 16-bit channels, CV field size is limited to register bits 15:0.

### 50.7.7 TC Register A

**Name:** TC_RAx
**Offset:** 0x14 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register has access Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1.
This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RA[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RA[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RA[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RA[31:0]** Register A
RA contains the Register A value in real time.

**Important:**
For 16-bit channels, RA field size is limited to register bits 15:0.

### 50.7.8 TC Register B

**Name:** TC_RBx
**Offset:** 0x18 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register has access Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1.
This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RB[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RB[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RB[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RB[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RB[31:0]** Register B
RB contains the Register B value in real time.

**Important:**
For 16-bit channels, RB field size is limited to register bits 15:0.

### 50.7.9 TC Register C

**Name:** TC_RCx
**Offset:** 0x1C + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RC[31:24] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RC[23:16] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RC[15:8] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RC[7:0] | | | | | | | |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – RC[31:0]** Register C
RC contains the Register C value in real time.

**Important:**
For 16-bit channels, RC field size is limited to register bits 15:0.

### 50.7.10 TC Interrupt Status Register

**Name:** TC_SRx
**Offset:** 0x20 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | MTIOB | MTIOA | CLKSTA |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 18 – MTIOB** TIOBx Mirror

| Value | Description |
|---|---|
| 0 | TIOBx is low. If TC_CMRx.WAVE = 0, TIOBx pin is low. If TC_CMRx.WAVE = 1, TIOBx is driven low. |
| 1 | TIOBx is high. If TC_CMRx.WAVE = 0, TIOBx pin is high. If TC_CMRx.WAVE = 1, TIOBx is driven high. |

**Bit 17 – MTIOA** TIOAx Mirror

| Value | Description |
|---|---|
| 0 | TIOAx is low. If TC_CMRx.WAVE = 0, TIOAx pin is low. If TC_CMRx.WAVE = 1, TIOAx is driven low. |
| 1 | TIOAx is high. If TC_CMRx.WAVE = 0, TIOAx pin is high. If TC_CMRx.WAVE = 1, TIOAx is driven high. |

**Bit 16 – CLKSTA** Clock Enabling Status

| Value | Description |
|---|---|
| 0 | Clock is disabled. |
| 1 | Clock is enabled. |

**Bit 7 – ETRGS** External Trigger Status (cleared on read)

| Value | Description |
|---|---|
| 0 | External trigger has not occurred since the last read of the Status Register. |
| 1 | External trigger has occurred since the last read of the Status Register. |

**Bit 6 – LDRBS** RB Loading Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RB Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1. |
| 1 | RB Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0. |

**Bit 5 – LDRAS** RA Loading Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RA Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1. |
| 1 | RA Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0. |

**Bit 4 – CPCS** RC Compare Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RC Compare has not occurred since the last read of the Status Register. |
| 1 | RC Compare has occurred since the last read of the Status Register. |

**Bit 3 – CPBS** RB Compare Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RB Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0. |
| 1 | RB Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1. |

**Bit 2 – CPAS** RA Compare Status (cleared on read)

| Value | Description |
|---|---|
| 0 | RA Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0. |
| 1 | RA Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1. |

**Bit 1 – LOVRS** Load Overrun Status (cleared on read)

| Value | Description |
|---|---|
| 0 | Load overrun has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1. |
| 1 | RA or RB have been loaded at least twice without any read of the corresponding register since the last read of the Status Register, if TC_CMRx.WAVE = 0. |

**Bit 0 – COVFS** Counter Overflow Status (cleared on read)

| Value | Description |
|---|---|
| 0 | No counter overflow has occurred since the last read of the Status Register. |
| 1 | A counter overflow has occurred since the last read of the Status Register. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.11 TC Interrupt Enable Register

**Name:** TC_IERx
**Offset:** 0x24 + x*0x40 [x=0..2]
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 7 – ETRGS** External Trigger

**Bit 6 – LDRBS** RB Loading

**Bit 5 – LDRAS** RA Loading

**Bit 4 – CPCS** RC Compare

**Bit 3 – CPBS** RB Compare

**Bit 2 – CPAS** RA Compare

**Bit 1 – LOVRS** Load Overrun

**Bit 0 – COVFS** Counter Overflow

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.12 TC Interrupt Disable Register

**Name:** TC_IDRx
**Offset:** 0x28 + x*0x40 [x=0..2]
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 7 – ETRGS** External Trigger

**Bit 6 – LDRBS** RB Loading

**Bit 5 – LDRAS** RA Loading

**Bit 4 – CPCS** RC Compare

**Bit 3 – CPBS** RB Compare

**Bit 2 – CPAS** RA Compare

**Bit 1 – LOVRS** Load Overrun

**Bit 0 – COVFS** Counter Overflow

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.13 TC Interrupt Mask Register

**Name:** TC_IMRx
**Offset:** 0x2C + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ETRGS | LDRBS | LDRAS | CPCS | CPBS | CPAS | LOVRS | COVFS |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ETRGS** External Trigger

**Bit 6 – LDRBS** RB Loading

**Bit 5 – LDRAS** RA Loading

**Bit 4 – CPCS** RC Compare

**Bit 3 – CPBS** RB Compare

**Bit 2 – CPAS** RA Compare

**Bit 1 – LOVRS** Load Overrun

**Bit 0 – COVFS** Counter Overflow

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.14 TC Extended Mode Register

**Name:** TC_EMRx
**Offset:** 0x30 + x*0x40 [x=0..2]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | NODIVCLK |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TRIGSRCB[1:0] | | | | TRIGSRCA[1:0] | |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bit 8 – NODIVCLK** No Divided Clock

| Value | Description |
|---|---|
| 0 | The selected clock is defined by field TCCLKS in TC_CMRx. |
| 1 | The selected clock is peripheral clock and TCCLKS field (TC_CMRx) has no effect. |

**Bits 5:4 – TRIGSRCB[1:0]** Trigger Source for Input B

| Value | Name | Description |
|---|---|---|
| 0 | EXTERNAL_TIOBx | The trigger/capture input B is driven by external pin TIOBx |
| 1 | PWMx | For TC0 to TC10: The trigger/capture input B is driven internally by the comparator output (see Synchronization with PWM) of the PWMx.<br>For TC11: The trigger/capture input B is driven internally by the GTSUCOMP signal of the Ethernet MAC (GMAC). |

**Bits 1:0 – TRIGSRCA[1:0]** Trigger Source for Input A

| Value | Name | Description |
|---|---|---|
| 0 | EXTERNAL_TIOAx | The trigger/capture input A is driven by external pin TIOAx |
| 1 | PWMx | The trigger/capture input A is driven internally by PWMx |

### 50.7.15 TC Block Control Register

**Name:** TC_BCR
**Offset:** 0xC0
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SYNC |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – SYNC** Synchro Command

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Asserts the SYNC signal which generates a software trigger simultaneously for each of the channels. |

### 50.7.16 TC Block Mode Register

**Name:** TC_BMR
**Offset:** 0xC4
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | MAXCMP[3:0] | | | | MAXFILT[5:4] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MAXFILT[3:0] | | | | | AUTOC | IDXPHB | SWAP |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | INVIDX | INVB | INVA | EDGPHA | QDTRANS | SPEEDEN | POSEN | QDEN |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TC2XC2S[1:0] | | TC1XC1S[1:0] | | TC0XC0S[1:0] | |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 29:26 – MAXCMP[3:0]** Maximum Consecutive Missing Pulses

| Value | Description |
|---|---|
| 0 | The flag MPE in TC_QISR never rises. |
| 1–15 | Defines the number of consecutive missing pulses before a flag report. |

**Bits 25:20 – MAXFILT[5:0]** Maximum Filter
Pulses with a period shorter than MAXFILT+1 peripheral clock cycles are discarded. For more details on MAXFILT constraints, see "Input Preprocessing"

| Value | Description |
|---|---|
| 1–63 | Defines the filtering capabilities. |

**Bit 18 – AUTOC** AutoCorrection of missing pulses
0 (DISABLED): The detection and autocorrection function is disabled.
1 (ENABLED): The detection and autocorrection function is enabled.

**Bit 17 – IDXPHB** Index Pin is PHB Pin

| Value | Description |
|---|---|
| 0 | IDX pin of the rotary sensor must drive TIOA1. |
| 1 | IDX pin of the rotary sensor must drive TIOB0. |

**Bit 16 – SWAP** Swap PHA and PHB

| Value | Description |
|---|---|
| 0 | No swap between PHA and PHB. |
| 1 | Swap PHA and PHB internally, prior to driving the QDEC. |

**Bit 15 – INVIDX** Inverted Index

| Value | Description |
|---|---|
| 0 | IDX (TIOA1) is directly driving the QDEC. |

| Value | Description |
|---|---|
| 1 | IDX is inverted before driving the QDEC. |

**Bit 14 – INVB** Inverted PHB

| Value | Description |
|---|---|
| 0 | PHB (TIOB0) is directly driving the QDEC. |
| 1 | PHB is inverted before driving the QDEC. |

**Bit 13 – INVA** Inverted PHA

| Value | Description |
|---|---|
| 0 | PHA (TIOA0) is directly driving the QDEC. |
| 1 | PHA is inverted before driving the QDEC. |

**Bit 12 – EDGPHA** Edge on PHA Count Mode

| Value | Description |
|---|---|
| 0 | Edges are detected on PHA only. |
| 1 | Edges are detected on both PHA and PHB. |

**Bit 11 – QDTRANS** Quadrature Decoding Transparent

| Value | Description |
|---|---|
| 0 | Full quadrature decoding logic is active (direction change detected). |
| 1 | Quadrature decoding logic is inactive (direction change inactive) but input filtering and edge detection are performed. |

**Bit 10 – SPEEDEN** Speed Enabled

| Value | Description |
|---|---|
| 0 | Disabled. |
| 1 | Enables the speed measure on channel 0, the time base being provided by channel 2. |

**Bit 9 – POSEN** Position Enabled

| Value | Description |
|---|---|
| 0 | Disable position. |
| 1 | Enables the position measure on channel 0 and 1. |

**Bit 8 – QDEN** Quadrature Decoder Enabled
Quadrature decoding (direction change) can be disabled using QDTRANS bit.
One of the POSEN or SPEEDEN bits must be also enabled.

| Value | Description |
|---|---|
| 0 | Disabled. |
| 1 | Enables the QDEC (filter, edge detection and quadrature decoding). |

**Bits 5:4 – TC2XC2S[1:0]** External Clock Signal 2 Selection

| Value | Name | Description |
|---|---|---|
| 0 | TCLK2 | Signal connected to XC2: TCLK2 |
| 1 | – | Reserved |
| 2 | TIOA0 | Signal connected to XC2: TIOA0 |
| 3 | TIOA1 | Signal connected to XC2: TIOA1 |

**Bits 3:2 – TC1XC1S[1:0]** External Clock Signal 1 Selection

| Value | Name | Description |
|---|---|---|
| 0 | TCLK1 | Signal connected to XC1: TCLK1 |
| 1 | – | Reserved |
| 2 | TIOA0 | Signal connected to XC1: TIOA0 |
| 3 | TIOA2 | Signal connected to XC1: TIOA2 |

**Bits 1:0 – TC0XC0S[1:0]** External Clock Signal 0 Selection

| Value | Name | Description |
|---|---|---|
| 0 | TCLK0 | Signal connected to XC0: TCLK0 |
| 1 | – | Reserved |
| 2 | TIOA1 | Signal connected to XC0: TIOA1 |
| 3 | TIOA2 | Signal connected to XC0: TIOA2 |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.17 TC QDEC Interrupt Enable Register

**Name:** TC_QIER
**Offset:** 0xC8
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when an occurrence of MAXCMP consecutive missing pulses is detected. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when a quadrature error occurs on PHA, PHB. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when a change on rotation direction is detected. |

**Bit 0 – IDX** Index

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when a rising edge occurs on IDX input. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.18 TC QDEC Interrupt Disable Register

**Name:** TC_QIDR
**Offset:** 0xCC
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when an occurrence of MAXCMP consecutive missing pulses has been detected. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when a quadrature error occurs on PHA, PHB. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when a change on rotation direction is detected. |

**Bit 0 – IDX** Index

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when a rising edge occurs on IDX input. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.19 TC QDEC Interrupt Mask Register

**Name:** TC_QIMR
**Offset:** 0xD0
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | The interrupt on the maximum number of consecutive missing pulses specified in MAXCMP is disabled. |
| 1 | The interrupt on the maximum number of consecutive missing pulses specified in MAXCMP is enabled. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | The interrupt on quadrature error is disabled. |
| 1 | The interrupt on quadrature error is enabled. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | The interrupt on rotation direction change is disabled. |
| 1 | The interrupt on rotation direction change is enabled. |

**Bit 0 – IDX** Index

| Value | Description |
|---|---|
| 0 | The interrupt on IDX input is disabled. |
| 1 | The interrupt on IDX input is enabled. |

### 50.7.20 TC QDEC Interrupt Status Register

**Name:** TC_QISR
**Offset:** 0xD4
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DIR |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | MPE | QERR | DIRCHG | IDX |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 8 – DIR** Direction
Returns an image of the current rotation direction.

**Bit 3 – MPE** Consecutive Missing Pulse Error

| Value | Description |
|---|---|
| 0 | The number of consecutive missing pulses has not reached the maximum value specified in MAXCMP since the last read of TC_QISR. |
| 1 | An occurrence of MAXCMP consecutive missing pulses has been detected since the last read of TC_QISR. |

**Bit 2 – QERR** Quadrature Error

| Value | Description |
|---|---|
| 0 | No quadrature error since the last read of TC_QISR. |
| 1 | A quadrature error occurred since the last read of TC_QISR. |

**Bit 1 – DIRCHG** Direction Change

| Value | Description |
|---|---|
| 0 | No change on rotation direction since the last read of TC_QISR. |
| 1 | The rotation direction changed since the last read of TC_QISR. |

**Bit 0 – IDX** Index

| Value | Description |
|---|---|
| 0 | No Index input change since the last read of TC_QISR. |
| 1 | The IDX input has changed since the last read of TC_QISR. |

### 50.7.21 TC Fault Mode Register

**Name:** TC_FMR
**Offset:** 0xD8
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | ENCF1 | ENCF0 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bit 1 – ENCF1** Enable Compare Fault Channel 1

| Value | Description |
|---|---|
| 0 | Disables the FAULT output source (CPCS flag) from channel 1. |
| 1 | Enables the FAULT output source (CPCS flag) from channel 1. |

**Bit 0 – ENCF0** Enable Compare Fault Channel 0

| Value | Description |
|---|---|
| 0 | Disables the FAULT output source (CPCS flag) from channel 0. |
| 1 | Enables the FAULT output source (CPCS flag) from channel 0. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 50.7.22 TC Write Protection Mode Register

**Name:** TC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x54494D | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
The Timer Counter clock of the first channel must be enabled to access this register.
See "Register Write Protection" for a list of registers that can be write-protected and Timer Counter clock conditions.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x54494D ("TIM" in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x54494D ("TIM" in ASCII). |

# 51. Pulse Width Modulation Controller (PWM)

## 51.1 Description

The Pulse Width Modulation Controller (PWM) generates output pulses on 4 channels independently according to parameters defined per channel. Each channel controls two complementary square output waveforms. Characteristics of the output waveforms such as period, duty-cycle, polarity and dead-times (also called dead-bands or non-overlapping times) are configured through the user interface. Each channel selects and uses one of the clocks provided by the clock generator. The clock generator provides several clocks resulting from the division of the PWM peripheral clock. External triggers can be managed to allow output pulses to be modified in real time.

All accesses to the PWM are made through registers mapped on the peripheral bus. All channels integrate a double buffering system in order to prevent an unexpected output waveform while modifying the period, the spread spectrum, the duty-cycle or the dead-times.

Channels can be linked together as synchronous channels to be able to update their duty-cycle or dead-times at the same time.

The update of duty-cycles of synchronous channels can be performed by the DMA Controller channel which offers buffer transfer without processor Intervention.

The PWM includes a spread-spectrum counter to allow a constantly varying period (only for Channel 0). This counter may be useful to minimize electromagnetic interference or to reduce the acoustic noise of a PWM driven motor.

The PWM provides 8 independent comparison units capable of comparing a programmed value to the counter of the synchronous channels (counter of channel 0). These comparisons are intended to generate software interrupts, to trigger pulses on the 2 independent event lines (in order to synchronize ADC conversions with a lot of flexibility independently of the PWM outputs) and to trigger DMA Controller transfer requests.

PWM outputs can be overridden synchronously or asynchronously to their channel counter.

The PWM provides a fault protection mechanism with 8 fault inputs, capable to detect a fault condition and to override the PWM outputs asynchronously (outputs forced to ‘0’, ‘1’ or Hi-Z).

For safety usage, some configuration registers are write-protected.

## 51.2 Embedded Characteristics

- 4 Channels
- Common Clock Generator Providing Thirteen Different Clocks
  - A Modulo n Counter Providing Eleven Clocks
  - Two Independent Linear Dividers Working on Modulo n Counter Outputs
- Independent Channels
  - Independent 16-bit Counter for Each Channel
  - Independent Complementary Outputs with 12-bit Dead-Time Generator (Also Called Dead-Band or Non-Overlapping Time) for Each Channel
  - Independent Push-Pull Mode for Each Channel
  - Independent Enable Disable Command for Each Channel
  - Independent Clock Selection for Each Channel
  - Independent Period, Duty-Cycle and Dead-Time for Each Channel
  - Independent Double Buffering of Period, Duty-Cycle and Dead-Times for Each Channel
  - Independent Programmable Selection of The Output Waveform Polarity for Each Channel, with Double Buffering
  - Independent Programmable Center- or Left-aligned Output Waveform for Each Channel
  - Independent Output Override for Each Channel
  - Independent Interrupt for Each Channel, at Each Period for Left-Aligned or Center-Aligned Configuration

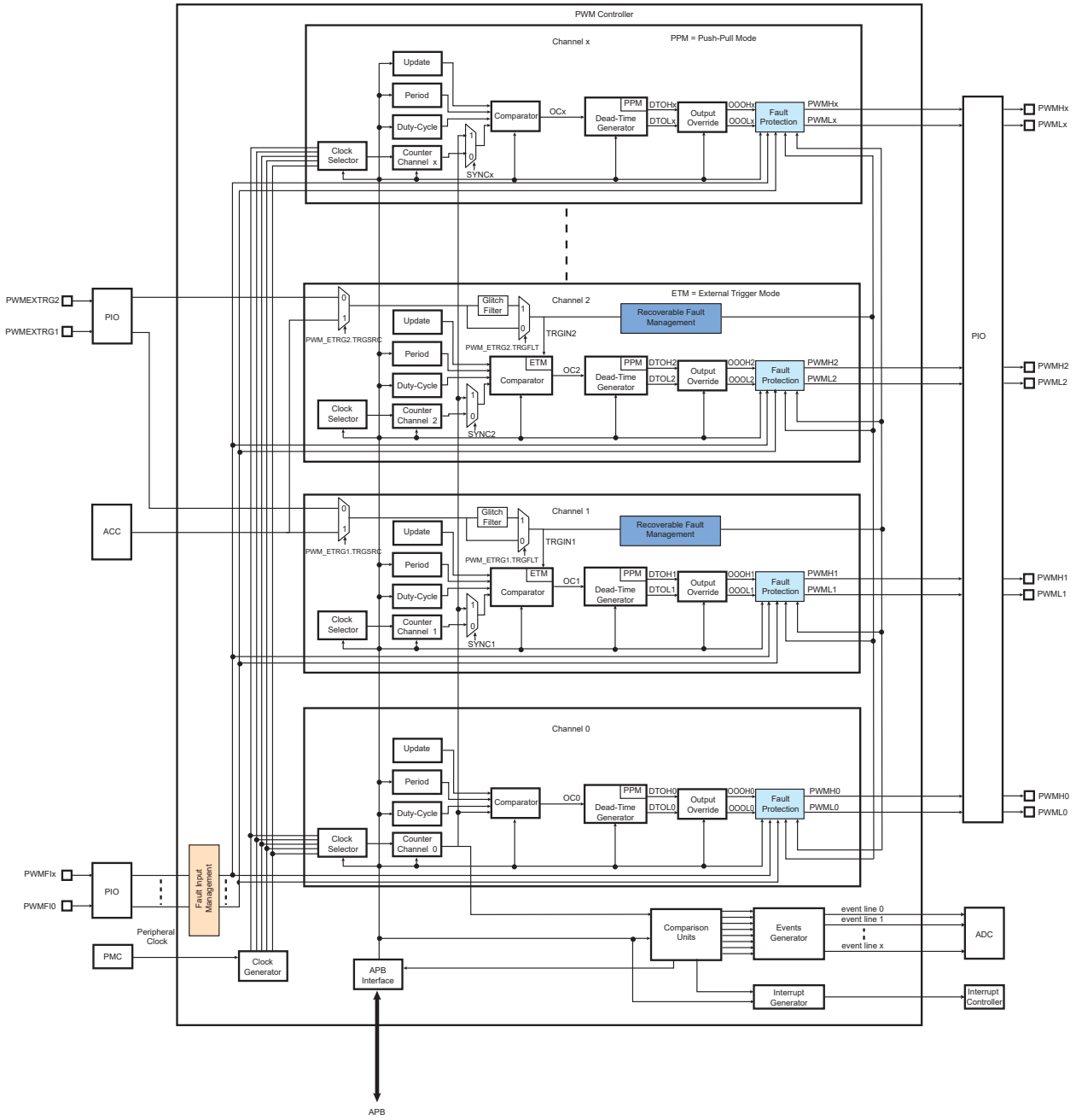
© 2021 Microchip Technology Inc.
and its subsidiaries

  - Independent Update Time Selection of Double Buffering Registers (Polarity, Duty Cycle) for Each Channel, at Each Period for Left-Aligned or Center-Aligned Configuration
- External Trigger Input Management (e.g., for DC/DC or Lighting Control)
  - External PWM Reset Mode
  - External PWM Start Mode
  - Cycle-By-Cycle Duty Cycle Mode
  - Leading-Edge Blanking
- 2 2-bit Gray Up/Down Channels for Stepper Motor Control
- Spread Spectrum Counter to Allow a Constantly Varying Duty Cycle (only for Channel 0)
- Synchronous Channel Mode
  - Synchronous Channels Share the Same Counter
  - Mode to Update the Synchronous Channels Registers after a Programmable Number of Periods
  - Synchronous Channels Supports Connection of one DMA Controller Channel Which Offers Buffer Transfer Without Processor Intervention To Update Duty-Cycle Registers
- 2 Independent Events Lines Intended to Synchronize ADC Conversions
  - Programmable delay for Events Lines to delay ADC measurements
- 8 Comparison Units Intended to Generate Interrupts, Pulses on Event LinesDMA Controller Transfer Requests
- 8 Programmable Fault Inputs Providing an Asynchronous Protection of PWM Outputs
  - 3 User Driven through PIO Inputs
  - PMC Driven when Crystal Oscillator Clock Fails
  - ADC Controller Driven through Configurable Comparison Function
  - Analog Comparator Controller Driven
  - Timer/Counter Driven through Configurable Comparison Function
- Register Write Protection

© 2021 Microchip Technology Inc.
and its subsidiaries

## 51.3 Block Diagram

**Figure 51-1. Pulse Width Modulation Controller Block Diagram**




**Note:** For a more detailed illustration of the fault protection circuitry, refer to "Fault Protection".

## 51.4 I/O Lines Description

Each channel outputs two complementary external I/O lines.

**Table 51-1. I/O Line Description**

| Name | Description | Type |
|---|---|---|
| PWMHx | PWM Waveform Output High for channel x | Output |
| PWMLx | PWM Waveform Output Low for channel x | Output |
| PWMFIx | PWM Fault Input x | Input |
| PWMEXTRGy | PWM Trigger Input y | Input |

## 51.5 Product Dependencies

### 51.5.1 I/O Lines

The pins used for interfacing the PWM are multiplexed with PIO lines. The programmer must first program the PIO controller to assign the desired PWM pins to their peripheral function. If I/O lines of the PWM are not used by the application, they can be used for other purposes by the PIO controller.

All of the PWM outputs may or may not be enabled. If an application requires only four channels, then only four PIO lines are assigned to PWM outputs.

### 51.5.2 Power Management

The PWM is not continuously clocked. The programmer must first enable the PWM clock in the Power Management Controller (PMC) before using the PWM. However, if the application does not require PWM operations, the PWM clock can be stopped when not needed and be restarted later. In this case, the PWM will resume its operations where it left off.

### 51.5.3 Interrupt Sources

The PWM interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the PWM interrupt requires the Interrupt Controller to be programmed first.

### 51.5.4 Fault Inputs

The PWM has the fault inputs connected to the different modules. Refer to the implementation of these modules within the product for detailed information about the fault generation procedure. The PWM receives faults from:

- PIO inputs
- the PMC
- the ADC controller
- the Analog Comparator Controller
- Timer/Counters

© 2021 Microchip Technology Inc.
and its subsidiaries

**Table 51-2. Fault Inputs**

| Fault Generator | External PWM Fault Input Number | Polarity Level[1] | Fault Input ID |
|---|---|---|---|
| PWM0 | | | |
| PA9 | PWMC0_PWMFI0 | User-defined | 0 |
| PD8 | PWMC0_PWMFI1 | User-defined | 1 |
| PD9 | PWMC0_PWMFI2 | User-defined | 2 |
| Main OSC (PMC) | – | To be configured to 1 | 3 |
| AFEC0 | – | To be configured to 1 | 4 |
| AFEC1 | – | To be configured to 1 | 5 |
| ACC | – | To be configured to 1 | 6 |
| Timer0 | – | To be configured to 1 | 7 |
| PWM1 | | | |
| PA21 | PWMC1_PWMFI0 | User-defined | 0 |
| PA26 | PWMC1_PWMFI1 | User-defined | 1 |
| PA28 | PWMC1_PWMFI2 | User-defined | 2 |
| Main OSC (PMC) | – | To be configured to 1 | 3 |
| AFEC0 | – | To be configured to 1 | 4 |
| AFEC1 | – | To be configured to 1 | 5 |
| ACC | – | To be configured to 1 | 6 |
| Timer1 | – | To be configured to 1 | 7 |

© 2021 Microchip Technology Inc. and its subsidiaries

**Note:**

1. FPOL field in PWMC_FMR.

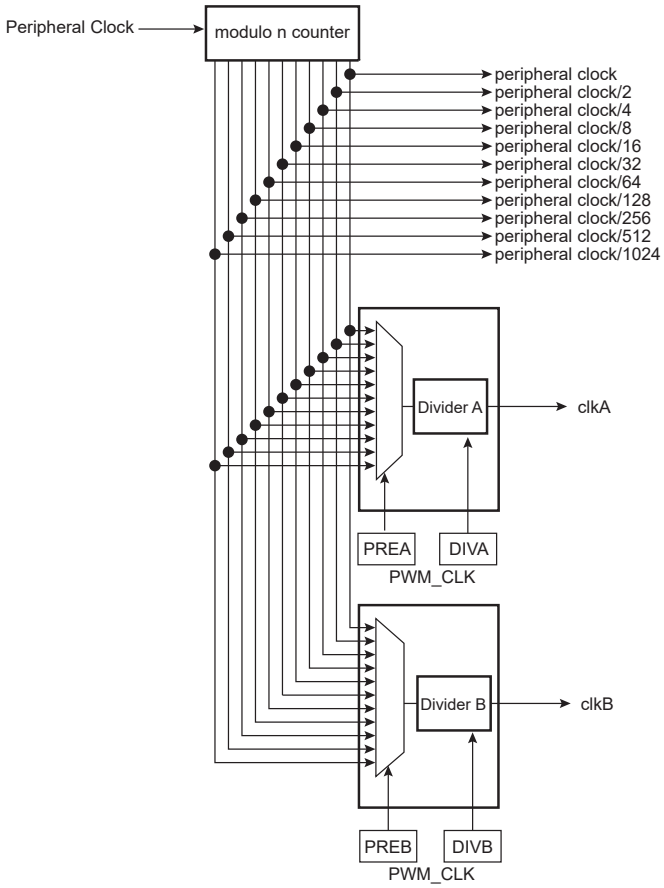
## 51.6 Functional Description

The PWM controller is primarily composed of a clock generator module and 4 channels.

- Clocked by the peripheral clock, the clock generator module provides 13 clocks.
- Each channel can independently choose one of the clock generator outputs.
- Each channel generates an output waveform with attributes that can be defined independently for each channel through the user interface registers.

### 51.6.1 PWM Clock Generator

**Figure 51-2. Functional View of the Clock Generator Block Diagram**




The PWM peripheral clock is divided in the clock generator module to provide different clocks available for all channels. Each channel can independently select one of the divided clocks.

The clock generator is divided into different blocks:

- a modulo n counter which provides 11 clocks: $f_{peripheral\ clock}$, $f_{peripheral\ clock}/2$, $f_{peripheral\ clock}/4$, $f_{peripheral\ clock}/8$, $f_{peripheral\ clock}/16$, $f_{peripheral\ clock}/32$, $f_{peripheral\ clock}/64$, $f_{peripheral\ clock}/128$, $f_{peripheral\ clock}/256$, $f_{peripheral\ clock}/512$, $f_{peripheral\ clock}/1024$
- two linear dividers (1, 1/2, 1/3, ... 1/255) that provide two separate clocks: clkA and clkB

Each linear divider can independently divide one of the clocks of the modulo n counter. The selection of the clock to be divided is made according to the PREA (PREB) field of the PWM Clock register (PWM_CLK). The resulting clock clkA (clkB) is the clock selected divided by DIVA (DIVB) field value.

After a reset of the PWM controller, DIVA (DIVB) and PREA (PREB) are set to ‘0’. This implies that after reset clkA (clkB) are turned off.

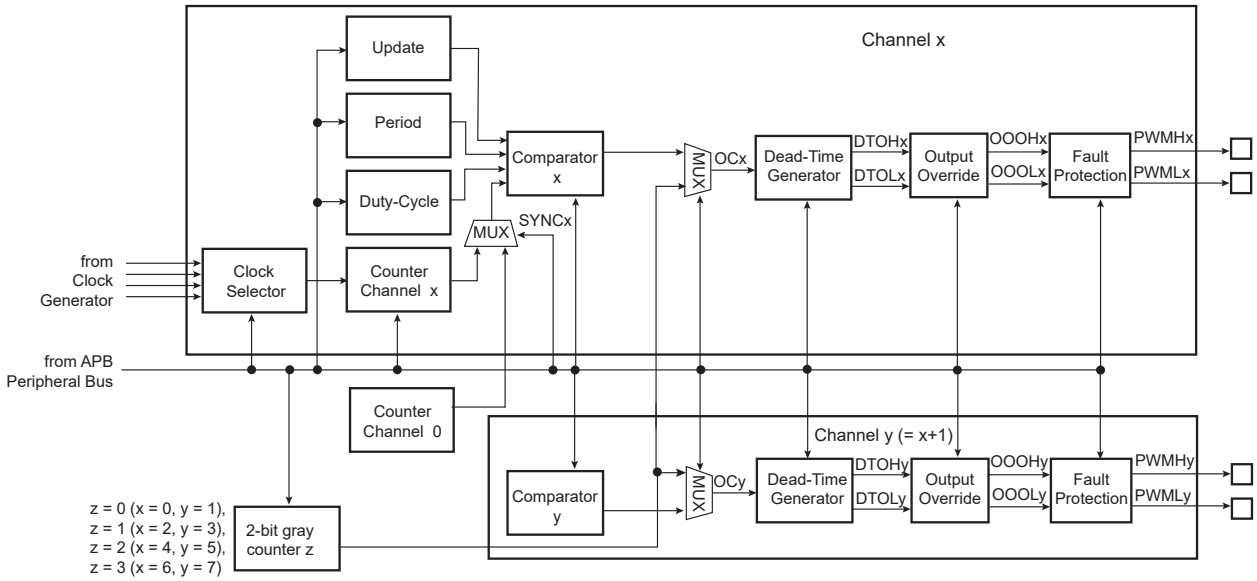
At reset, all clocks provided by the modulo n counter are turned off except the peripheral clock. This situation is also true when the PWM peripheral clock is turned off through the Power Management Controller.

**⚠ CAUTION** Before using the PWM controller, the programmer must first enable the peripheral clock in the Power Management Controller (PMC).

## 51.6.2 PWM Channel

### 51.6.2.1 Channel Block Diagram

**Figure 51-3. Functional View of the Channel Block Diagram**




Each of the 4 channels is composed of six blocks:

- A clock selector which selects one of the clocks provided by the clock generator (described in PWM Clock Generator ).
- A counter clocked by the output of the clock selector. This counter is incremented or decremented according to the channel configuration and comparators matches. The size of the counter is 16 bits.
- A comparator used to compute the OCx output waveform according to the counter value and the configuration. The counter value can be the one of the channel counter or the one of the channel 0 counter according to SYNCx bit in the PWM Sync Channels Mode Register (PWM_SCM).
- A 2-bit configurable gray counter enables the stepper motor driver. One gray counter drives 2 channels.
- A dead-time generator providing two complementary outputs (DTOHx/DTOLx) which allows to drive external power control switches safely.
- An output override block that can force the two complementary outputs to a programmed value (OOOHx/ OOOLx).
- An asynchronous fault protection mechanism that has the highest priority to override the two complementary outputs (PWMHx/PWMLx) in case of fault detection (outputs forced to ‘0’, ‘1’ or Hi-Z).

### 51.6.2.2 Comparator

The comparator continuously compares its counter value with the channel period defined by CPRD in the PWM Channel Period Register (PWM_CPRDx) and the duty-cycle defined by CDTY in the PWM Channel Duty Cycle Register (PWM_CDTYx) to generate an output signal OCx accordingly.

The different properties of the waveform of the output OCx are:

- the clock selection. The channel counter is clocked by one of the clocks provided by the clock generator described in the previous section. This channel parameter is defined in the CPRE field of the PWM Channel Mode Register (PWM_CMRx). This field is reset at '0'.
- the waveform period. This channel parameter is defined in the CPRD field of the PWM_CPRDx register.
  If the waveform is left-aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM peripheral clock divided by a given prescaler value "X" (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

  $$\frac{(X \times \mathrm{CPRD})}{f_{\mathrm{peripheral\ clock}}}$$

  By using the PWM peripheral clock divided by a given prescaler value "X" (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

  $$\frac{(X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

  If the waveform is center-aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM peripheral clock divided by a given prescaler value "X" (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

  $$\frac{(2 \times X \times \mathrm{CPRD})}{f_{\mathrm{peripheral\ clock}}}$$

  By using the PWM peripheral clock divided by a given prescaler value "X" (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

  $$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or}$$

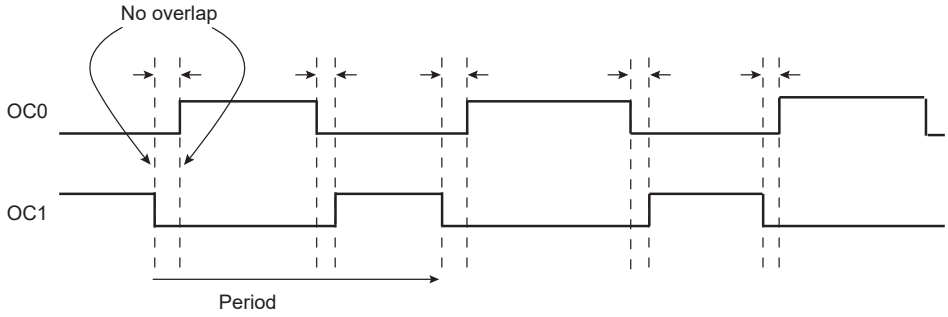
  $$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$
- the waveform duty-cycle. This channel parameter is defined in the CDTY field of the PWM_CDTYx register.
  If the waveform is left-aligned, then:

  $$\mathrm{duty\ cycle} = \frac{(\mathrm{period} - 1/\mathrm{fchannel_x_clock} \times \mathrm{CDTY})}{\mathrm{period}}$$

  If the waveform is center-aligned, then:

  $$\mathrm{duty\ cycle} = \frac{((\mathrm{period}/2) - 1/\mathrm{fchannel_x_clock} \times \mathrm{CDTY}))}{(\mathrm{period}/2)}$$
- the waveform polarity. At the beginning of the period, the signal can be at high or low level. This property is defined in the CPOL bit of PWM_CMRx. By default, the signal starts by a low level. The DPOLI bit in PWM_CMRx defines the PWM polarity when the channel is disabled (CHIDx = 0 in PWM_SR). For more details, see the figure *Waveform Properties.*
  - DPOLI = 0: PWM polarity when the channel is disabled is the same as the one defined for the beginning of the PWM period.
  - DPOLI = 1: PWM polarity when the channel is disabled is inverted compared to the one defined for the beginning of the PWM period.
- the waveform alignment. The output waveform can be left- or center-aligned. Center-aligned waveforms can be used to generate non-overlapped waveforms. This property is defined in the CALG bit of PWM_CMRx. The default mode is left-aligned.

**Figure 51-4. Non-Overlapped Center-Aligned Waveforms**




**Note:** See the figure *Waveform Properties* for a detailed description of center-aligned waveforms.

When center-aligned, the channel counter increases up to CPRD and decreases down to 0. This ends the period.

When left-aligned, the channel counter increases up to CPRD and is reset. This ends the period.

Thus, for the same CPRD value, the period for a center-aligned channel is twice the period for a left-aligned channel.

Waveforms are fixed at 0 when:

- CDTY = CPRD and CPOL = 0 (Note that if TRGMODE = MODE3, the PWM waveform switches to 1 at the external trigger event (see Cycle-By-Cycle Duty Mode)).
- CDTY = 0 and CPOL = 1

Waveforms are fixed at 1 (once the channel is enabled) when:

- CDTY = 0 and CPOL = 0
- CDTY = CPRD and CPOL = 1 (Note that if TRGMODE = MODE3, the PWM waveform switches to 0 at the external trigger event (see Cycle-By-Cycle Duty Mode)).

The waveform polarity must be set before enabling the channel. This immediately affects the channel output level.

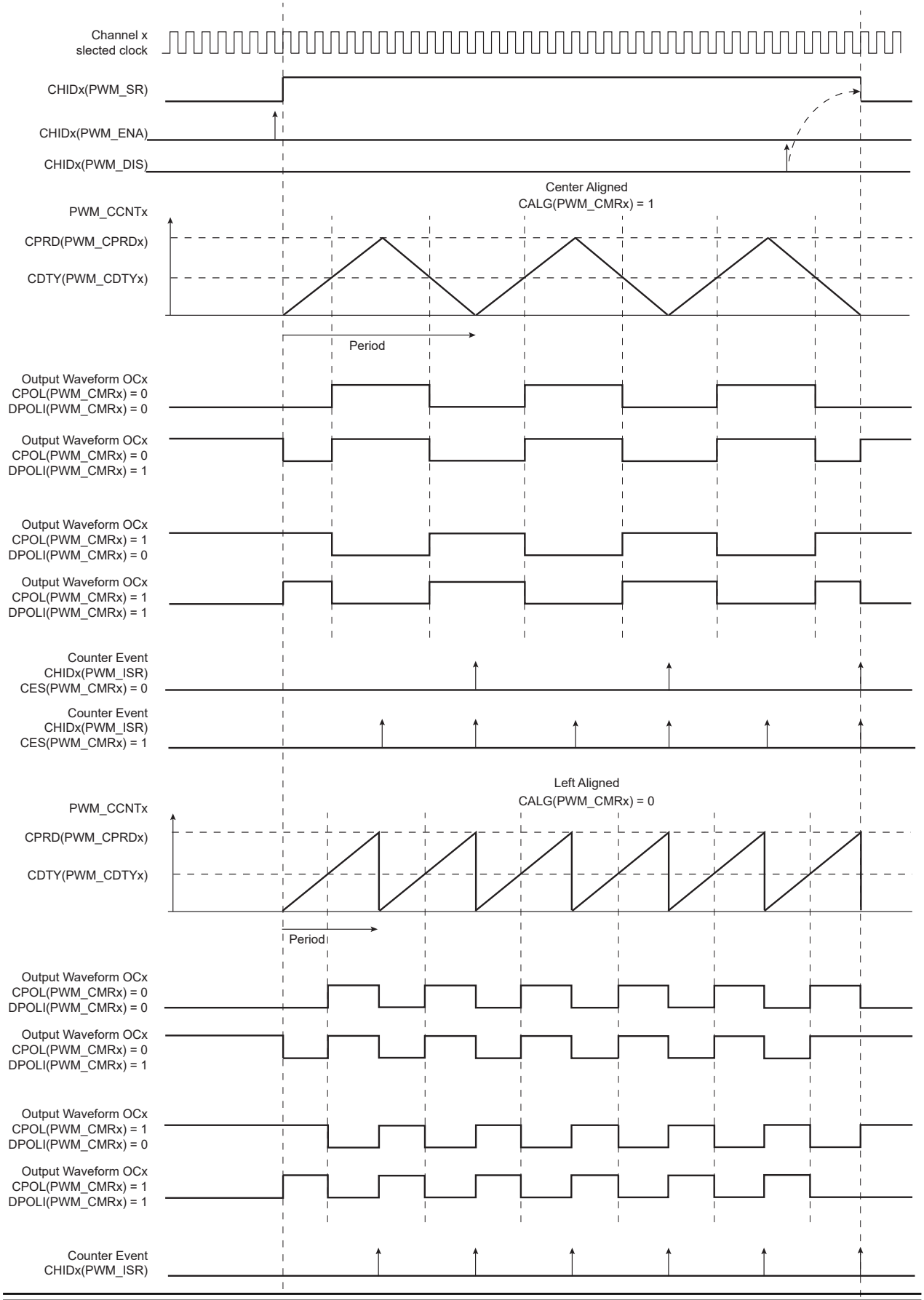
Modifying CPOL in PWM Channel Mode Register while the channel is enabled can lead to an unexpected behavior of the device being driven by PWM.

In addition to generating the output signals OCx, the comparator generates interrupts depending on the counter value. When the output waveform is left-aligned, the interrupt occurs at the end of the counter period. When the output waveform is center-aligned, the bit CES of PWM_CMRx defines when the channel counter interrupt occurs. If CES is set to ‘0’, the interrupt occurs at the end of the counter period. If CES is set to ‘1’, the interrupt occurs at the end of the counter period and at half of the counter period.

The figure below illustrates the counter interrupts depending on the configuration.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 51-5. Waveform Properties**



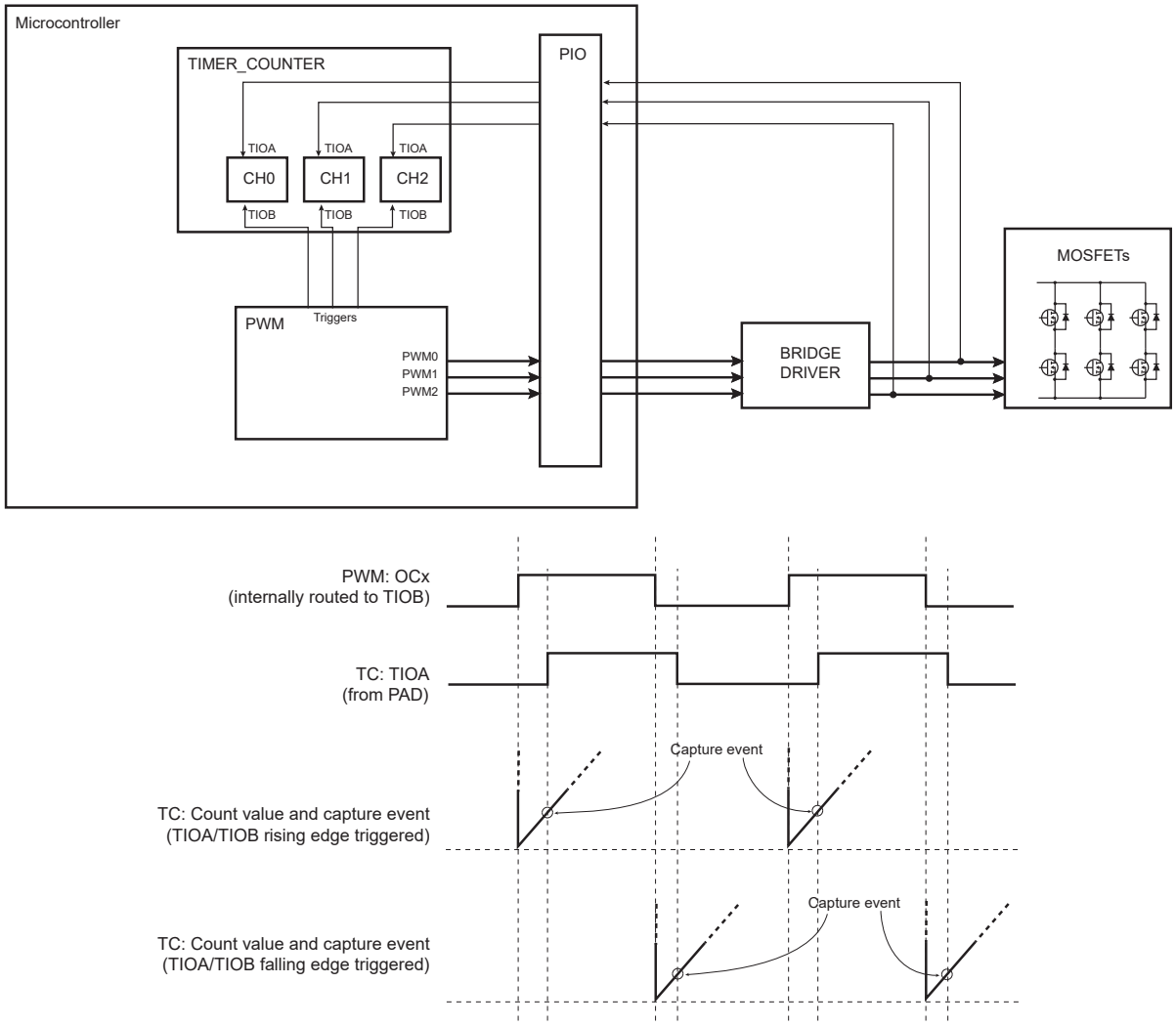
### 51.6.2.3 Trigger Selection for Timer Counter

The PWM controller can be used as a trigger source for the Timer Counter (TC) to achieve the two application examples described below.

#### 51.6.2.3.1 Delay Measurement

To measure the delay between the channel x comparator output (OCx) and the feedback from the bridge driver of the MOSFETs (see the figure below), the bit TCTS in the PWM Channel Mode Register must be at 0. This defines the comparator output of the channel x as the TC trigger source. The TIOB trigger (TC internal input) is used to start the TC; the TIOA input (from PAD) is used to capture the delay.

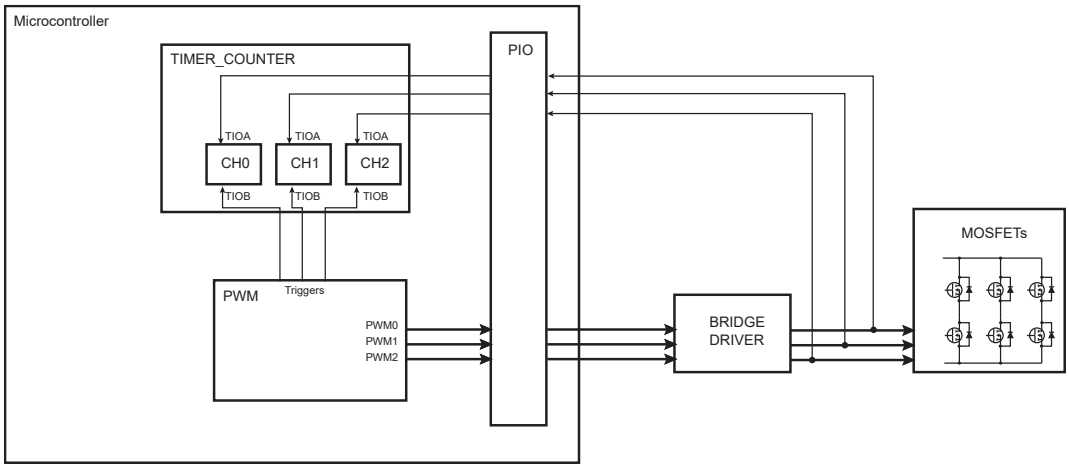
**Figure 51-6. Triggering the TC: Delay Measurement**

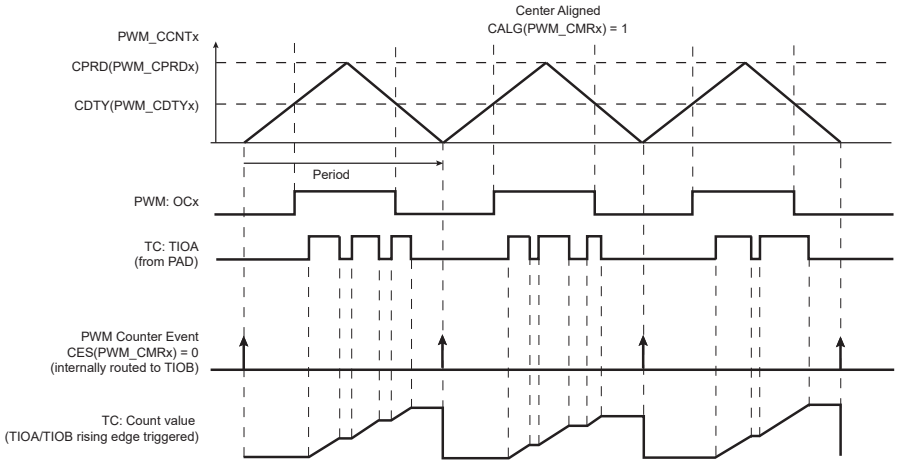


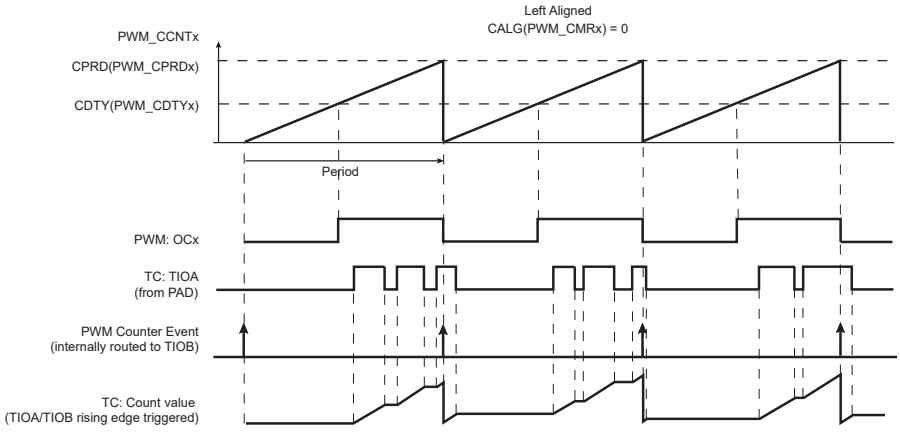

#### 51.6.2.3.2 Cumulated ON Time Measurement

To measure the cumulated “ON” time of MOSFETs (see the figure below), the bit TCTS of the PWM Channel Mode Register must be set to 1 to define the counter event (see the figure *Waveform Properties*) as the Timer Counter trigger source.

**Figure 51-7. Triggering the TC: Cumulated "ON" Time Measurement**



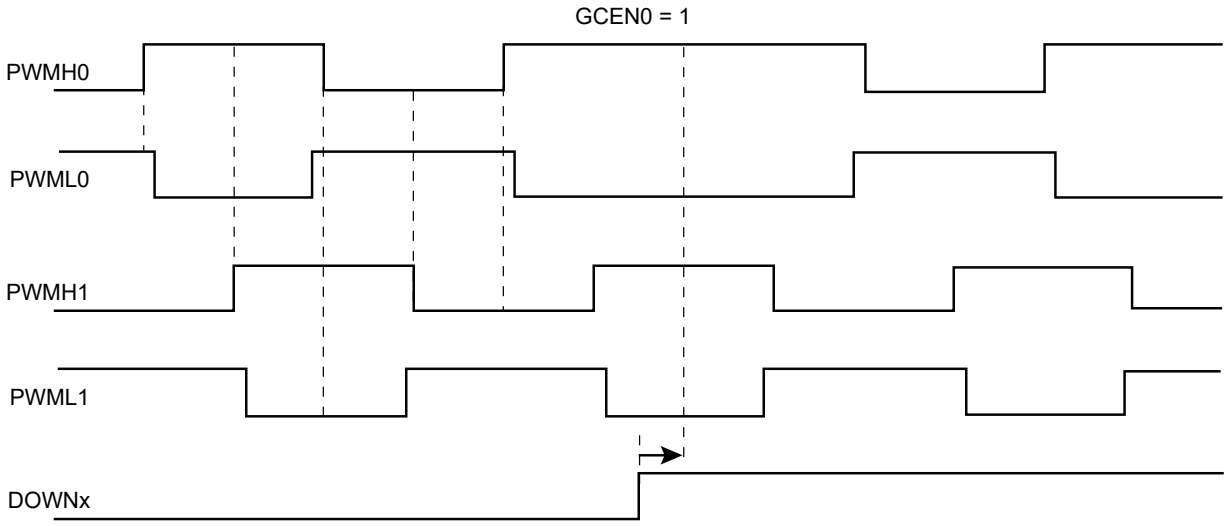

### 51.6.2.4 2-bit Gray Up/Down Counter for Stepper Motor

A pair of channels may provide a 2-bit gray count waveform on two outputs. Dead-time generator and other downstream logic can be configured on these channels.

Up or Down Count mode can be configured on-the-fly by means of PWM_SMMR configuration registers.

When GCEN0 is set to ‘1’, channels 0 and 1 outputs are driven with gray counter.

**Figure 51-8. 2-bit Gray Up/Down Counter**




### 51.6.2.5 Dead-Time Generator

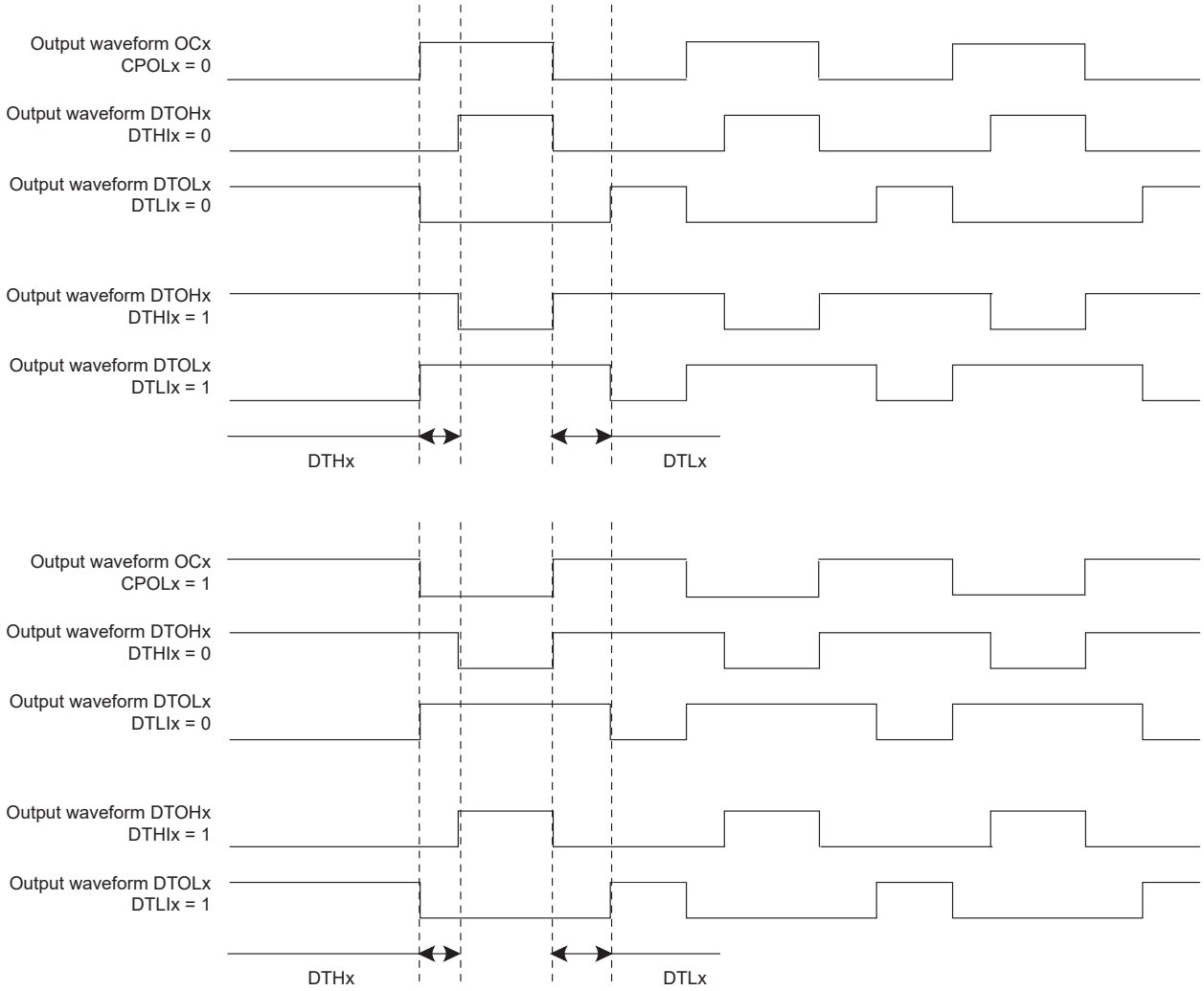
The dead-time generator uses the comparator output OCx to provide the two complementary outputs DTOHx and DTOLx, which allows the PWM macrocell to drive external power control switches safely. When the dead-time generator is enabled by setting the bit DTE to 1 or 0 in the PWM Channel Mode Register (PWM_CMRx), dead-times (also called dead-bands or non-overlapping times) are inserted between the edges of the two complementary outputs DTOHx and DTOLx. Note that enabling or disabling the dead-time generator is allowed only if the channel is disabled.

The dead-time is adjustable by the PWM Channel Dead Time Register (PWM_DTx). Each output of the dead-time generator can be adjusted separately by DTH and DTL. The dead-time values can be updated synchronously to the PWM period by using the PWM Channel Dead Time Update Register (PWM_DTUPDx).

The dead-time is based on a specific counter which uses the same selected clock that feeds the channel counter of the comparator. Depending on the edge and the configuration of the dead-time, DTOHx and DTOLx are delayed until the counter has reached the value defined by DTH or DTL. An inverted configuration bit (DTHI and DTLI bit in PWM_CMRx) is provided for each output to invert the dead-time outputs. The following figure shows the waveform of the dead-time generator.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 51-9. Complementary Output Waveforms**




#### 51.6.2.5.1 PWM Push-Pull Mode

When a PWM channel is configured in Push-Pull mode, the dead-time generator output is managed alternately on each PWM cycle. The polarity of the PWM line during the idle state of the Push-Pull mode is defined by the DPOLI bit in the PWM Channel Mode Register (PWM_CMRx). The Push-Pull mode can be enabled separately on each channel by writing a one to bit PPM in the PWM Channel Mode Register.

**Figure 51-10. PWM Push-Pull Mode**

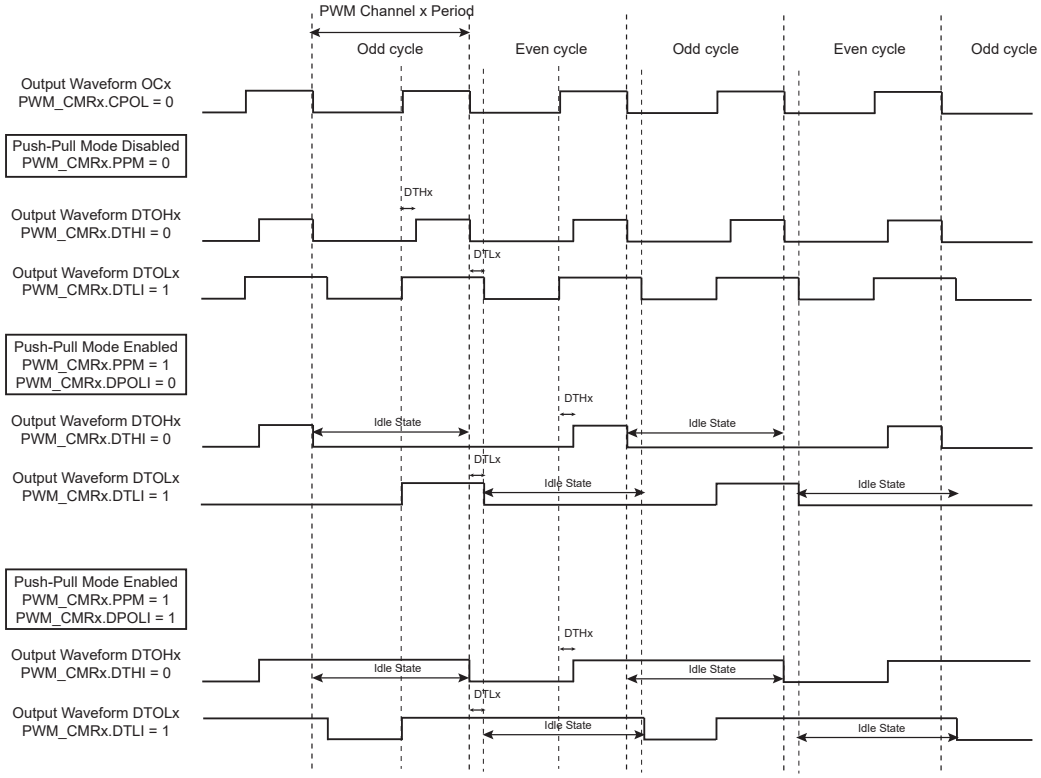


**Figure 51-11. PWM Push-Pull Waveforms: Left-Aligned Mode**

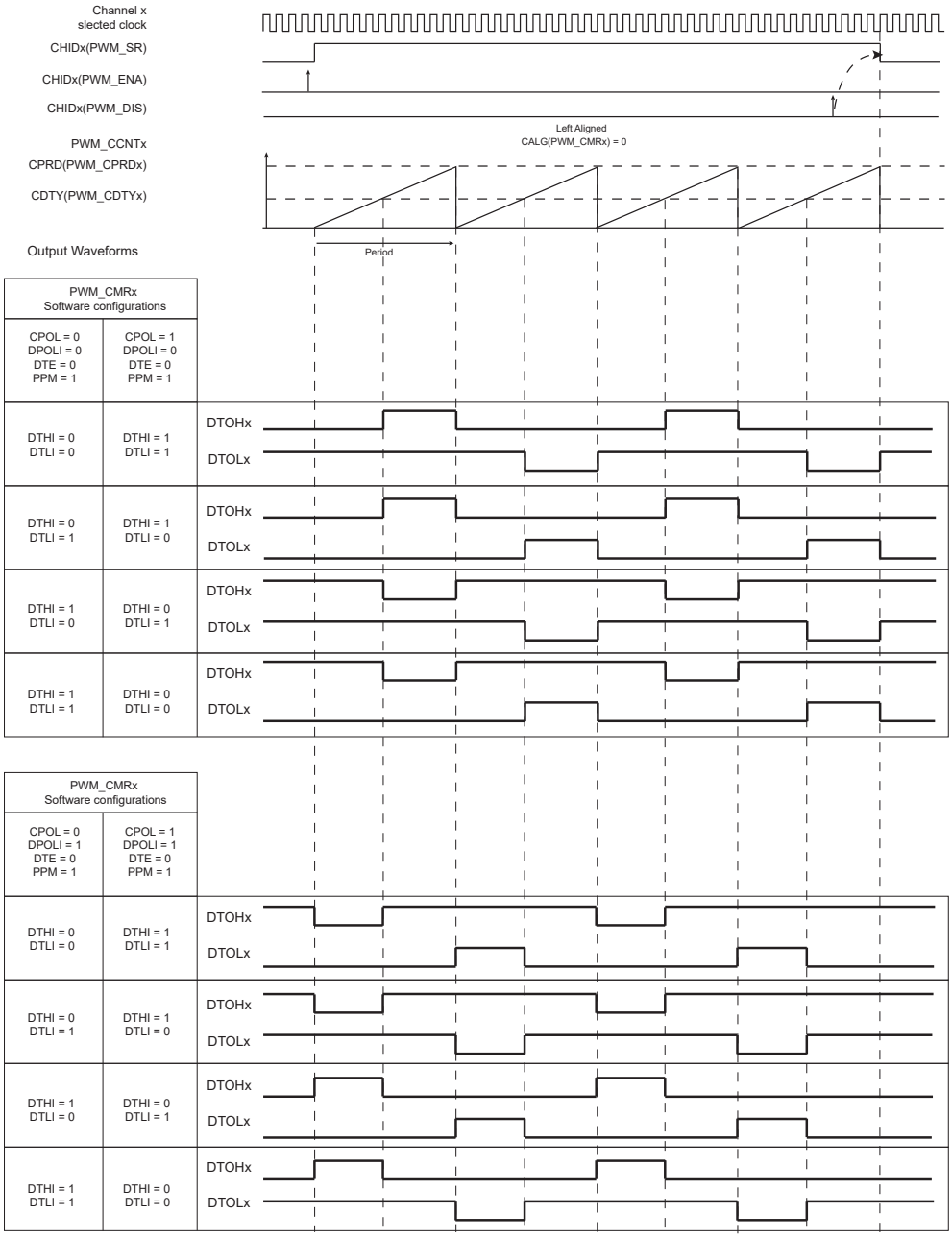
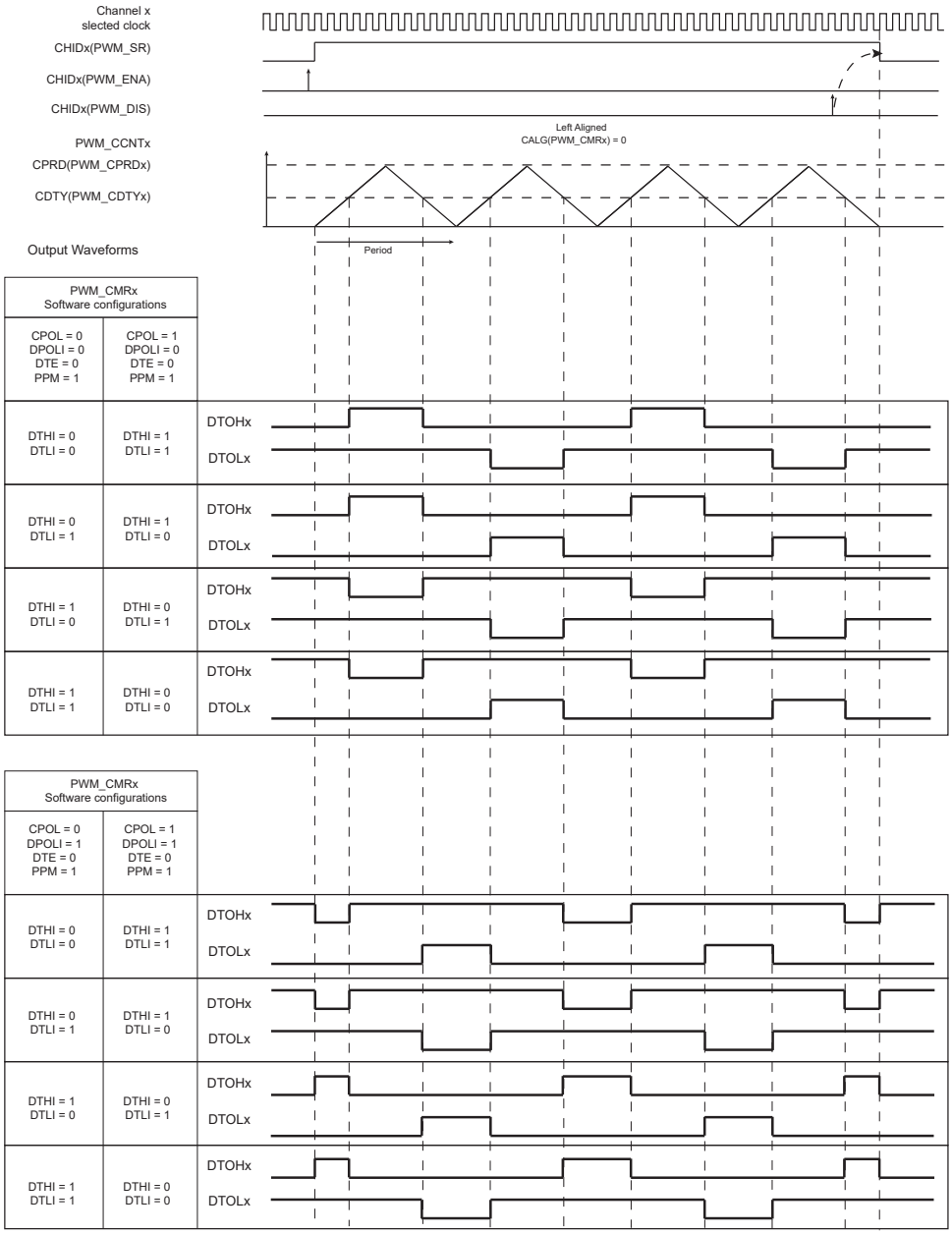


**Figure 51-12. PWM Push-Pull Waveforms: Center-Aligned Mode**




The PWM Push-Pull mode can be useful in transformer-based power converters, such as a half-bridge converter. The Push-Pull mode prevents the transformer core from being saturated by any direct current.

**Figure 51-13. Half-Bridge Converter Application: No Feedback Regulation**

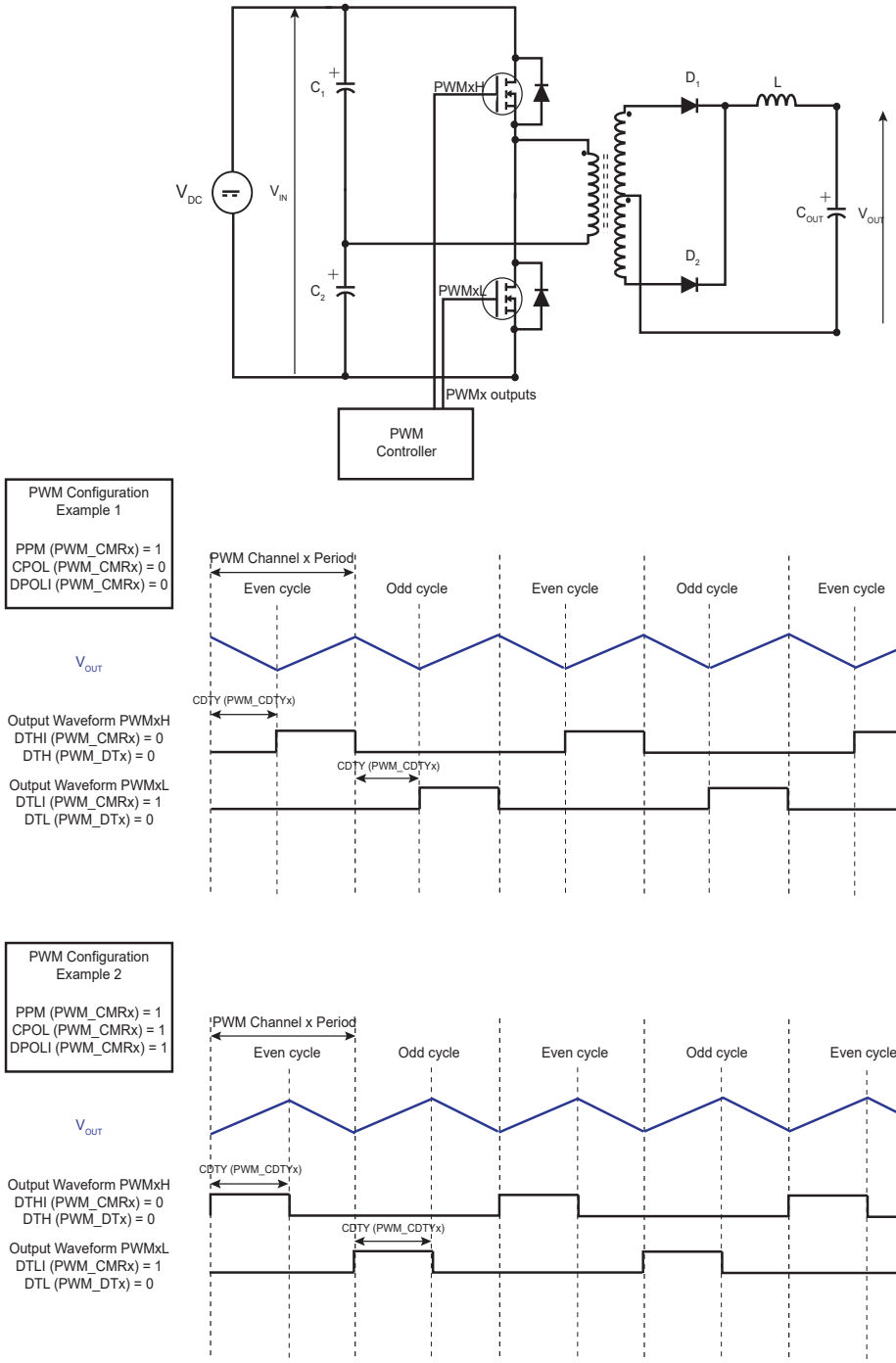
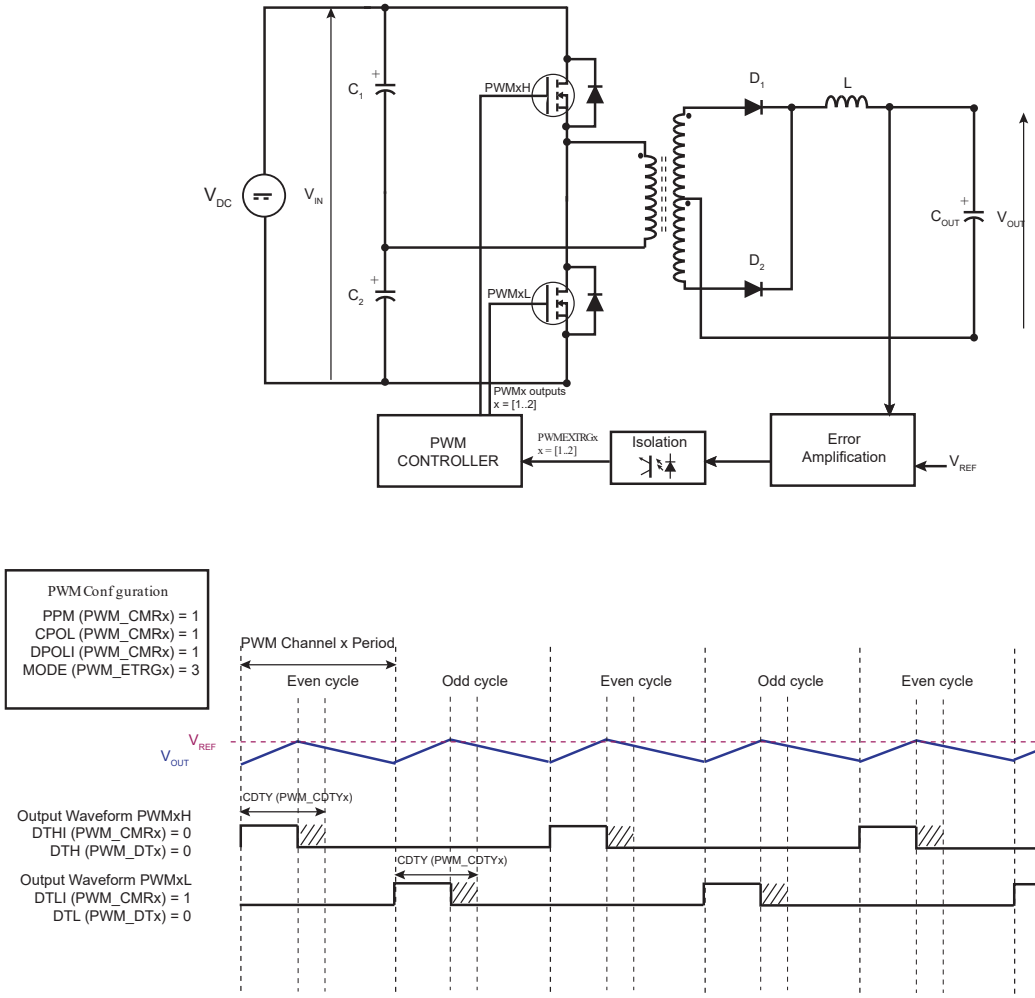


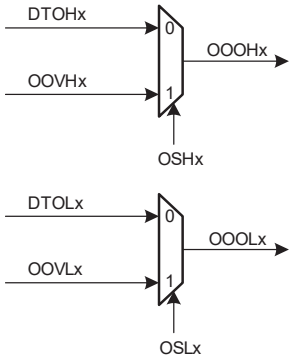
**Figure 51-14. Half-Bridge Converter Application: Feedback Regulation**




### 51.6.2.6 Output Override

The two complementary outputs DTOHx and DTOLx of the dead-time generator can be forced to a value defined by the software.

**Figure 51-15. Override Output Selection**




The fields OSHx and OSLx in the PWM Output Selection Register (PWM_OS) allow the outputs of the dead-time generator DTOHx and DTOLx to be overridden by the value defined in the fields OOVHx and OOVLx in the PWM Output Override Value Register (PWM_OOV).

The set registers PWM Output Selection Set Register (PWM_OSS) and PWM Output Selection Set Update Register (PWM_OSSUPD) enable the override of the outputs of a channel regardless of other channels. In the same way,

the clear registers PWM Output Selection Clear Register (PWM_OSC) and PWM Output Selection Clear Update Register (PWM_OSCUPD) disable the override of the outputs of a channel regardless of other channels.

By using buffer registers PWM_OSSUPD and PWM_OSCUPD, the output selection of PWM outputs is done synchronously to the channel counter, at the beginning of the next PWM period.

By using registers PWM_OSS and PWM_OSC, the output selection of PWM outputs is done asynchronously to the channel counter, as soon as the register is written.

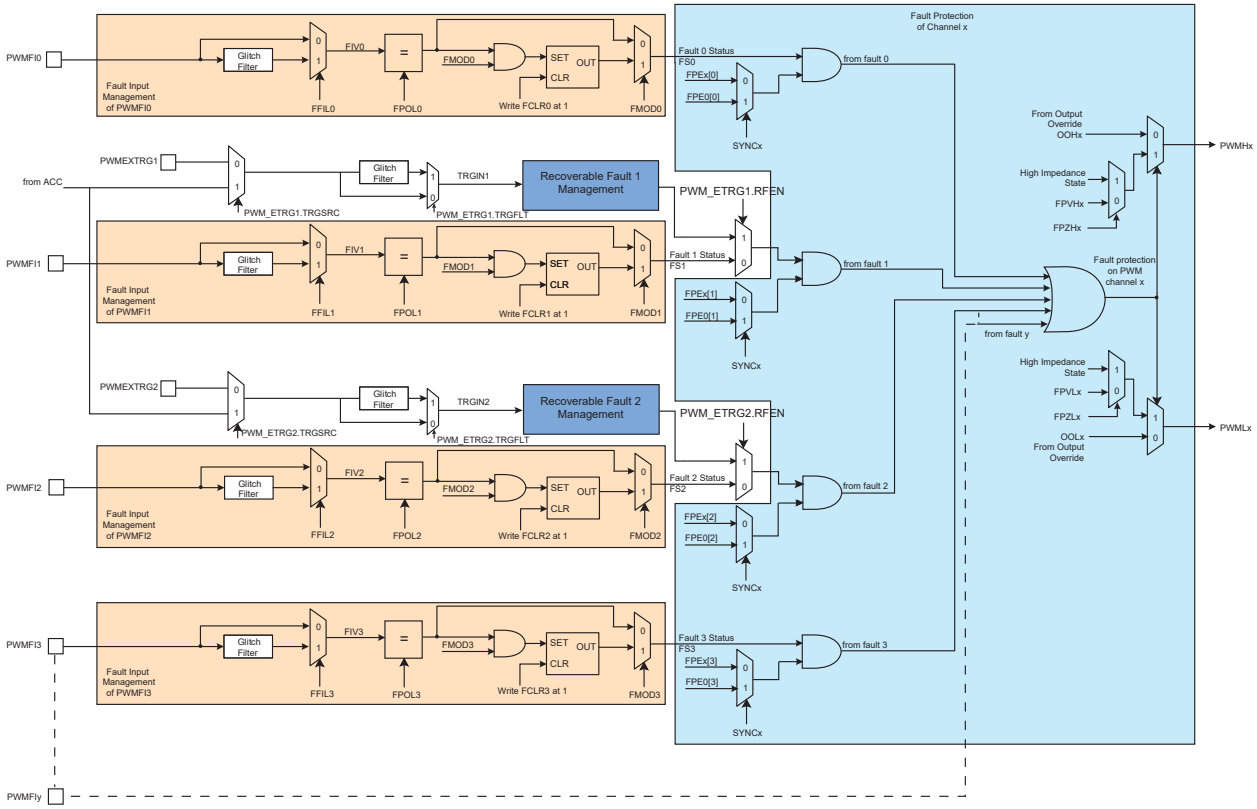
The value of the current output selection can be read in PWM_OS.

While overriding PWM outputs, the channel counters continue to run, only the PWM outputs are forced to user defined values.

### 51.6.2.7 Fault Protection

8 inputs provide fault protection which can force any of the PWM output pairs to a programmable value. This mechanism has priority over output overriding.

**Figure 51-16. Fault Protection**




The polarity level of the fault inputs is configured by the FPOL field in the PWM Fault Mode Register (PWM_FMR). For fault inputs coming from internal peripherals such as ADC or Timer Counter, the polarity level must be FPOL = 1. For fault inputs coming from external GPIO pins the polarity level depends on the user's implementation.

The configuration of the Fault Activation mode (FMOD field in PWMC_FMR) depends on the peripheral generating the fault. If the corresponding peripheral does not have “Fault Clear” management, then the FMOD configuration to use must be FMOD = 1, to avoid spurious fault detection. Refer to the corresponding peripheral documentation for details on handling fault generation.

Fault inputs may or may not be glitch-filtered depending on the FFIL field in PWM_FMR. When the filter is activated, glitches on fault inputs with a width inferior to the PWM peripheral clock period are rejected.

A fault becomes active as soon as its corresponding fault input has a transition to the programmed polarity level. If the corresponding bit FMOD is set to ‘0’ in PWM_FMR, the fault remains active as long as the fault input is at this polarity level. If the corresponding FMOD field is set to ‘1’, the fault remains active until the fault input is no longer at this polarity level and until it is cleared by writing the corresponding bit FCLR in the PWM Fault Clear Register

(PWM_FCR). In the PWM Fault Status Register (PWM_FSR), the field FIV indicates the current level of the fault inputs and the field FIS indicates whether a fault is currently active.

Each fault can be taken into account or not by the fault protection mechanism in each channel. To be taken into account in the channel x, the fault y must be enabled by the bit FPEx[y] in the PWM Fault Protection Enable register (PWM_FPE1). However, synchronous channels (see Synchronous Channels) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[y]).

The fault protection on a channel is triggered when this channel is enabled and when any one of the faults that are enabled for this channel is active. It can be triggered even if the PWM peripheral clock is not running but only by a fault input that is not glitch-filtered.

When the fault protection is triggered on a channel, the fault protection mechanism resets the counter of this channel and forces the channel outputs to the values defined by the fields FPVHx and FPVLx in the PWM Fault Protection Value Register 1 (PWM_FPV) and fields FPZHx/FPZLx in the PWM Fault Protection Value Register 2, as shown in the table below. The output forcing is made asynchronously to the channel counter.

**Table 51-3. Forcing Values of PWM Outputs by Fault Protection**

| FPZH/Lx | FPVH/Lx | Forcing Value of PWMH/Lx |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | – | High impedance state (Hi-Z) |

**⚠ CAUTION**

- To prevent any unexpected activation of the status flag FSy in PWM_FSR, the FMODy bit can be set to ‘1’ only if the FPOLy bit has been previously configured to its final value.
- To prevent any unexpected activation of the Fault Protection on the channel x, the bit FPEx[y] can be set to ‘1’ only if the FPOLy bit has been previously configured to its final value.

If a comparison unit is enabled (see PWM Comparison Units) and if a fault is triggered in the channel 0, then the comparison cannot match.

As soon as the fault protection is triggered on a channel, an interrupt (different from the interrupt generated at the end of the PWM period) can be generated but only if it is enabled and not masked. The interrupt is reset by reading the interrupt status register, even if the fault which has caused the trigger of the fault protection is kept active.

#### 51.6.2.7.1 Recoverable Fault

The PWM provides a Recoverable Fault mode on fault 1 and 2 (see figure *Fault Protection*).

The recoverable fault signal is an internal signal generated as soon as an external trigger event occurs (see PWM External Trigger Mode).

When the fault 1 or 2 is defined as a recoverable fault, the corresponding fault input pin is ignored and bits FFIL1/2, FMOD1/2 and FFIL1/2 are not taken into account.

The fault 1 is managed as a recoverable fault by the PWMEXTRG1 input trigger when PWM_ETRG1.RFEN = 1, PWM_ENA.CHID1 = 1, and PWM_ETRG1.TRGMODE ≠ 0.

The fault 2 is managed as a recoverable fault by the PWMEXTRG2 input trigger when PWM_ETRG2.RFEN = 1, PWM_ENA.CHID2 = 1, and PWM_ETRG2.TRGMODE ≠ 0.

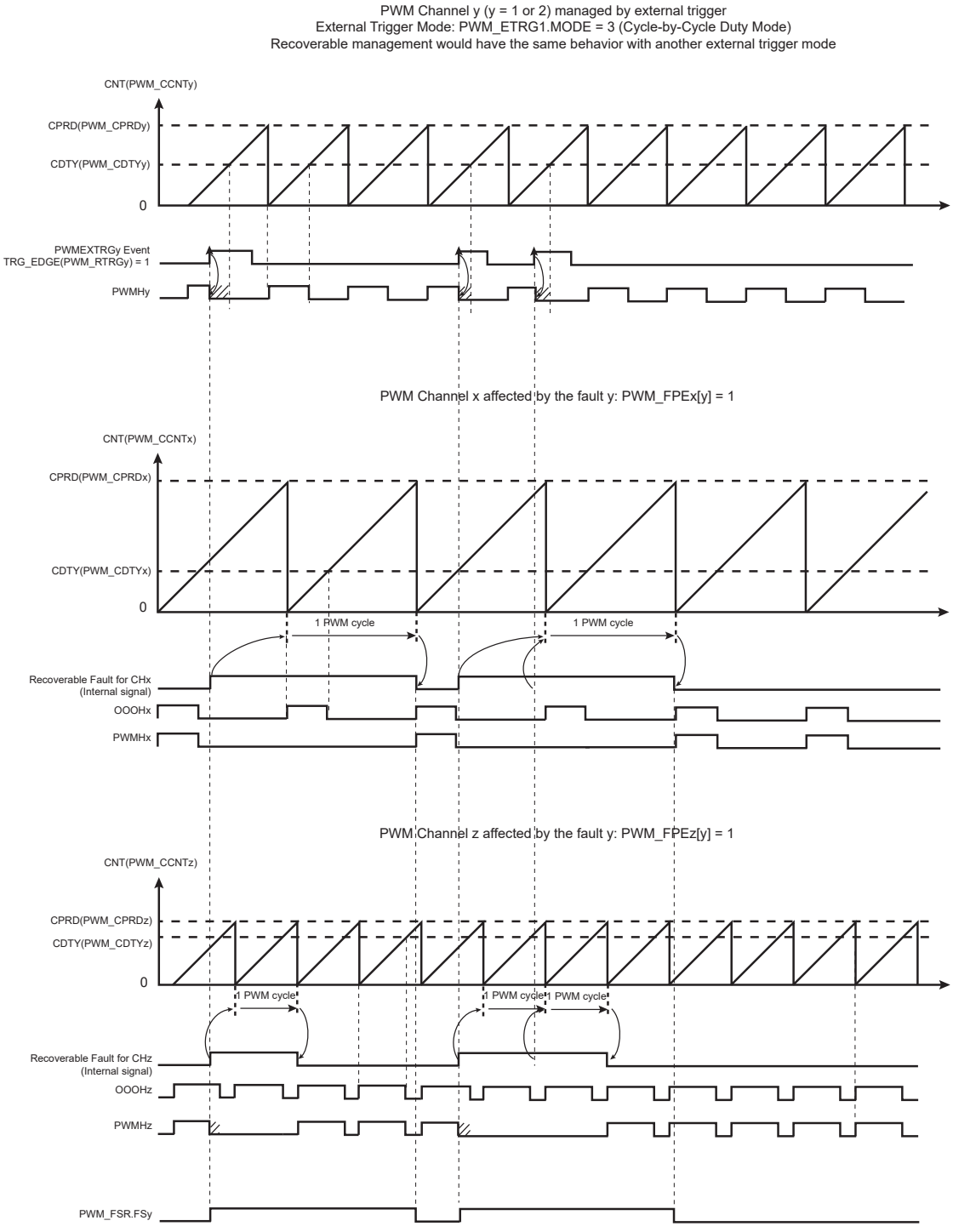
Recoverable fault 1 and 2 can be taken into account by all channels by enabling the bit FPEx[1/2] in the PWM Fault Protection Enable registers (PWM_FPEx). However the synchronous channels (see Synchronous Channels) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[1/2]).

When a recoverable fault is triggered (according to the PWM_ETRGx.TRGMODE setting), the PWM counter of the affected channels is not cleared (unlike in the classic fault protection mechanism) but the channel outputs are forced to the values defined by the fields FPVHx and FPVLx in the PWM Fault Protection Value Register 1 (PWM_FPV), as per table *Forcing Values of PWM Outputs by Fault Protection*. The output forcing is made asynchronously to the

channel counter and lasts from the recoverable fault occurrence to the end of the next PWM cycle (if the recoverable fault is no longer present) (see the figure below).

The recoverable fault does not trigger an interrupt. The Fault Status FSy (with y = 1 or 2) is not reported in the PWM Fault Status Register when the fault y is a recoverable fault.

**Figure 51-17. Recoverable Fault Management**



### 51.6.2.8 Spread Spectrum Counter

The PWM macrocell includes a spread spectrum counter allowing the generation of a constantly varying duty cycle on the output PWM waveform (only for the channel 0). This feature may be useful to minimize electromagnetic interference or to reduce the acoustic noise of a PWM driven motor.

This is achieved by varying the effective period in a range defined by a spread spectrum value which is programmed by the field SPRD in the PWM Spread Spectrum Register (PWM_SSPR). The effective period of the output waveform is the value of the spread spectrum counter added to the programmed waveform period CPRD in the PWM Channel Period Register (PWM_CPRD0).

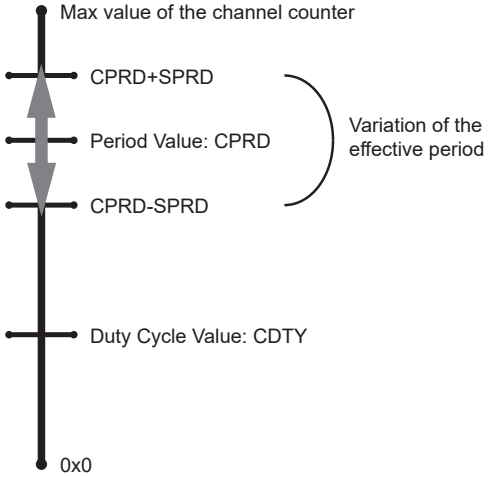
It will cause the effective period to vary from CPRD-SPRD to CPRD+SPRD. This leads to a constantly varying duty cycle on the PWM output waveform because the duty cycle value programmed is unchanged.

The value of the spread spectrum counter can change in two ways depending on the bit SPRDM in PWM_SSPR.

If SPRDM = 0, the Triangular mode is selected. The spread spectrum counter starts to count from -SPRD when the channel 0 is enabled or after reset and counts upwards at each period of the channel counter. When it reaches SPRD, it restarts to count from -SPRD again.

If SPRDM = 1, the Random mode is selected. A new random value is assigned to the spread spectrum counter at each period of the channel counter. This random value is between -SPRD and +SPRD and is uniformly distributed.

**Figure 51-18. Spread Spectrum Counter**




### 51.6.2.9 Synchronous Channels

Some channels can be linked together as synchronous channels. They have the same source clock, the same period, the same alignment and are started together. In this way, their counters are synchronized together.

The synchronous channels are defined by the SYNCx bits in the PWM Sync Channels Mode Register (PWM_SCM). Only one group of synchronous channels is allowed.

When a channel is defined as a synchronous channel, the channel 0 is also automatically defined as a synchronous channel. This is because the channel 0 counter configuration is used by all the synchronous channels.

If a channel x is defined as a synchronous channel, the fields/bits for the channel 0 are used instead of those of channel x:

- CPRE in PWM_CMR0 instead of CPRE in PWM_CMRx (same source clock)
- CPRD in PWM_CPRD0 instead of CPRD in PWM_CPRDx (same period)
- CALG in PWM_CMR0 instead of CALG in PWM_CMRx (same alignment)

Modifying the fields CPRE, CPRD and CALG of for channels with index greater than 0 has no effect on output waveforms.

Because counters of synchronous channels must start at the same time, they are all enabled together by enabling the channel 0 (by the CHID0 bit in PWM_ENA register). In the same way, they are all disabled together by disabling channel 0 (by the CHID0 bit in PWM_DIS register). However, a synchronous channel x different from channel 0 can be enabled or disabled independently from others (by the CHIDx bit in PWM_ENA and PWM_DIS registers).

© 2021 Microchip Technology Inc. and its subsidiaries

Defining a channel as a synchronous channel while it is an asynchronous channel (by writing the bit SYNCx to ‘1’ while it was at ‘0’) is allowed only if the channel is disabled at this time (CHIDx = 0 in PWM_SR). In the same way, defining a channel as an asynchronous channel while it is a synchronous channel (by writing the SYNCx bit to ‘0’ while it was ‘1’) is allowed only if the channel is disabled at this time.

The UPDM field (Update Mode) in the PWM_SCM register selects one of the three methods to update the registers of the synchronous channels:

- Method 1 (UPDM = 0): The period value, the duty-cycle values and the dead-time values must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).The update is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM Sync Channels Update Control Register (PWM_SCUC) is set to ‘1’.
- Method 2 (UPDM = 1): The period value, the duty-cycle values, the dead-time values and the update period value must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPD). The update of the period value and of the dead-time values is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM_SCUC register is set to ‘1’. The update of the duty-cycle values and the update period value is triggered automatically after an update period defined by the field UPR in the PWM Sync Channels Update Period Register (PWM_SCUP).
- Method 3 (UPDM = 2): Same as Method 2 apart from the fact that the duty-cycle values of ALL synchronous channels are written by the DMA Controller. The user can choose to synchronize the DMA Controller transfer request with a comparison match (see Section 7.3 “PWM Comparison Units”), by the fields PTRM and PTRCS in the PWM_SCM register. The DMA destination address must be configured to access only the PWM DMA Register (PWM_DMAR). The DMA buffer data structure must consist of sequentially repeated duty cycles. The number of duty cycles in each sequence corresponds to the number of synchronized channels. Duty cycles in each sequence must be ordered from the lowest to the highest channel index. The size of the duty cycle is 16 bits.

**Table 51-4. Summary of the Update of Registers of Synchronous Channels**

| Register | UPDM = 0 | UPDM = 1 | UPDM = 2 |
|---|---|---|---|
| Period Value (PWM_CPRDUPDx) | Write by the processor | | |
| | Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to ‘1’ | | |
| Dead-Time Values (PWM_DTUPDx) | Write by the processor | | |
| | Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to ‘1’ | | |
| Duty-Cycle Values (PWM_CDTYUPDx) | Write by the processor | Write by the processor | Write by the DMA Controller |
| | Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to ‘1’ | Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR. | |
| Update Period Value (PWM_SCUPUPD) | Not applicable | Write by the processor | |
| | Not applicable | Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR. | |

#### 51.6.2.9.1 Method 1: Manual write of duty-cycle values and manual trigger of the update

In this mode, the update of the period value, the duty-cycle values and the dead-time values must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).

To trigger the update, the user must use the bit UPDULOCK in the PWM_SCUC register which allows to update synchronously (at the same PWM period) the synchronous channels:

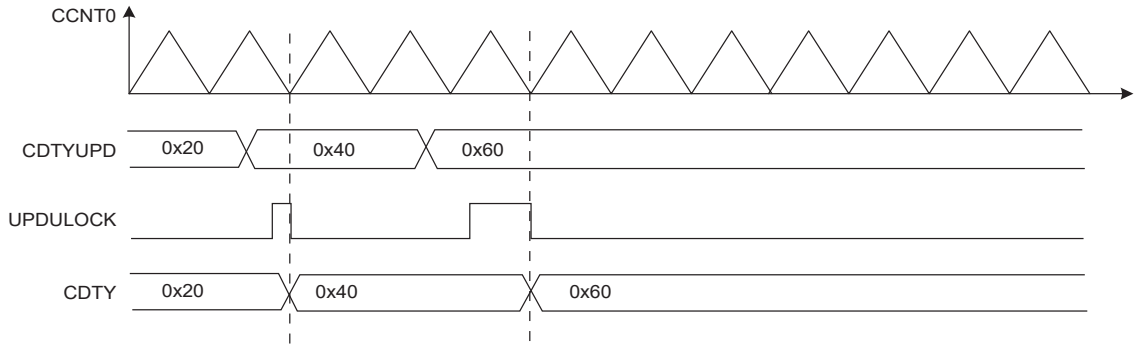
- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

Sequence for Method 1:

1. Select the manual write of duty-cycle values and the manual update by setting the UPDM field to ‘0’ in the PWM_SCM register.
2. Define the synchronous channels by the SYNCx bits in the PWM_SCM register.
3. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
4. If an update of the period value and/or the duty-cycle values and/or the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).
5. Set UPDULOCK to ‘1’ in PWM_SCUC.
6. The update of the registers will occur at the beginning of the next PWM period. When the UPDULOCK bit is reset, go to Step 4. for new values.

**Figure 51-19. Method 1 (UPDM = 0)**




#### 51.6.2.9.2 Method 2: Manual write of duty-cycle values and automatic trigger of the update

In this mode, the update of the period value, the duty-cycle values, the dead-time values and the update period value must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_CDTYUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK in the PWM_SCUC register, which updates synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the update period by the UPR field in the PWM_SCUP register. The PWM controller waits UPR+1 period of synchronous channels before updating automatically the duty values and the update period value.

The status of the duty-cycle value write is reported in the PWM Interrupt Status Register 2 (PWM_ISR2) by the following flags:

- WRDY: this flag is set to ‘1’ when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to ‘0’ when the PWM_ISR2 register is read.

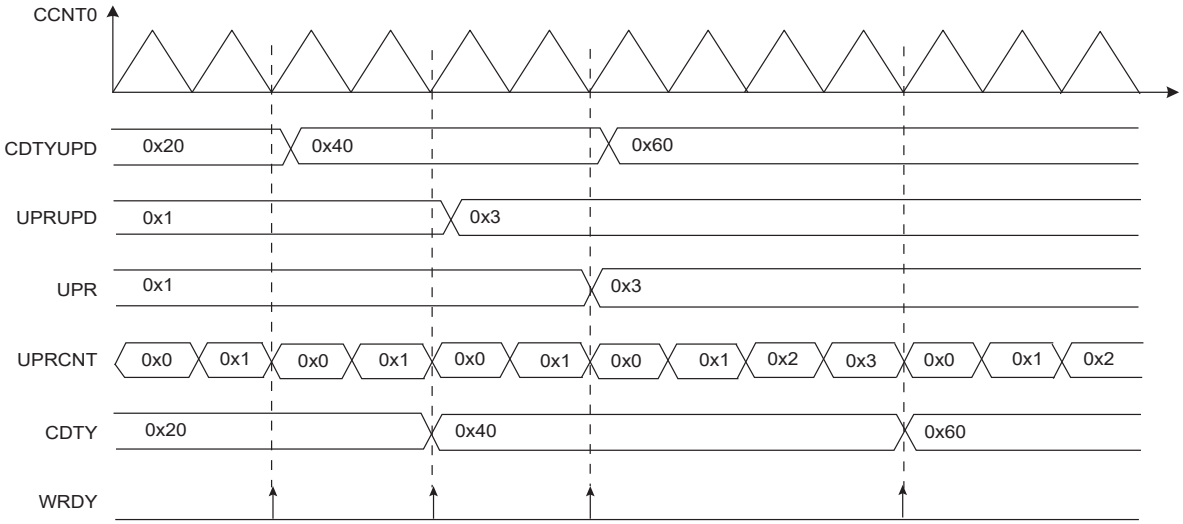
Depending on the interrupt mask in the PWM Interrupt Mask Register 2 (PWM_IMR2), an interrupt can be generated by these flags.

Sequence for Method 2:

1. Select the manual write of duty-cycle values and the automatic update by setting the field UPDM to ‘1’ in the PWM_SCM register
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
5. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 8.
6. Set UPDULOCK to ‘1’ in PWM_SCUC.

7. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 5. for new values.
8. If an update of the duty-cycle values and/or the update period is required, check first that write of new update values is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in PWM_ISR2.
9. Write registers that need to be updated (PWM_CDTYUPDx, PWM_SCUPUPD).
10. The update of these registers will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 8. for new values.

**Figure 51-20. Method 2 (UPDM = 1)**




#### 51.6.2.9.3 Method 3: Automatic write of duty-cycle values and automatic trigger of the update

In this mode, the update of the duty cycle values is made automatically by the DMA Controller. The update of the period value, the dead-time values and the update period value must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK which allows to update synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period value is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the Update Period by the field UPR in the PWM_SCUP register. The PWM controller waits UPR+1 periods of synchronous channels before updating automatically the duty values and the update period value.

Using the DMA Controller removes processor overhead by reducing its intervention during the transfer. This significantly reduces the number of clock cycles required for a data transfer, which improves microcontroller performance.

The DMA Controller must write the duty-cycle values in the synchronous channels index order. For example if the channels 0, 1 and 3 are synchronous channels, the DMA Controller must write the duty-cycle of the channel 0 first, then the duty-cycle of the channel 1, and finally the duty-cycle of the channel 3.

The status of the DMA Controller transfer is reported in PWM_ISR2 by the following flags:

- WRDY: this flag is set to ‘1’ when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to ‘0’ when PWM_ISR2 is read. The user can choose to synchronize the WRDY flag and the DMA Controller transfer request with a comparison match (see PWM Comparison Units), by the fields PTRM and PTRCS in the PWM_SCM register.
-

- UNRE: this flag is set to ‘1’ when the update period defined by the UPR field has elapsed while the whole data has not been written by the DMA Controller. It is reset to ‘0’ when PWM_ISR2 is read.

Depending on the interrupt mask in PWM_IMR2, an interrupt can be generated by these flags.

Sequence for Method 3:

1. Select the automatic write of duty-cycle values and automatic update by setting the field UPDM to 2 in the PWM_SCM register.
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Define when the WRDY flag and the corresponding DMA Controller transfer request must be set in the update period by the PTRM bit and the PTRCS field in the PWM_SCM register (at the end of the update period or when a comparison matches).
5. Define the DMA Controller transfer settings for the duty-cycle values and enable it in the DMA Controller registers
6. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
7. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 10.
8. Set UPDULOCK to ‘1’ in PWM_SCUC.
9. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 7. for new values.
10. If an update of the update period value is required, check first that write of a new update value is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in PWM_ISR2, else go to Step 14.
11. Write the register that needs to be updated (PWM_SCUPUPD).
12. The update of this register will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 10. for new values.
13. Wait for the DMA status flag indicating that the buffer transfer is complete. If the transfer has ended, define a new DMA transfer for new duty-cycle values. Go to Step 5.

**Figure 51-21. Method 3 (UPDM = 2 and PTRM = 0)**

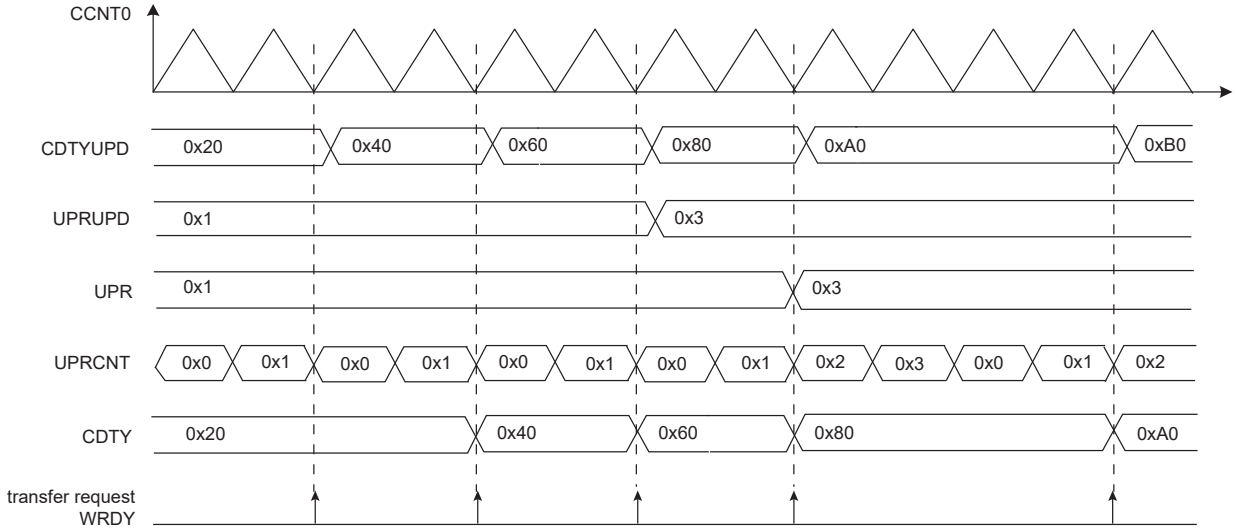
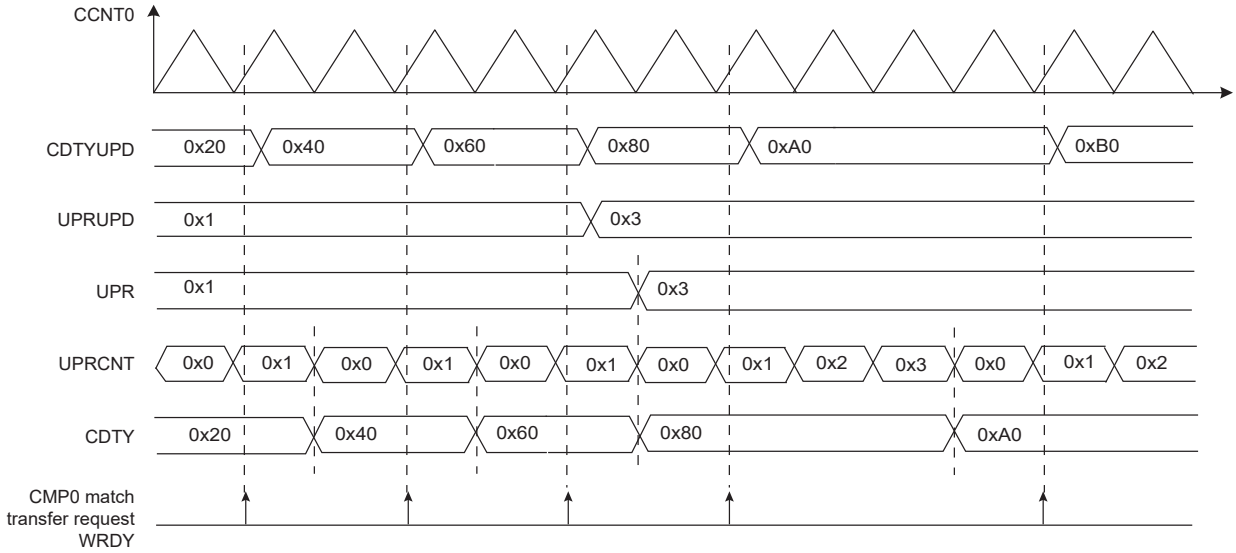


**Figure 51-22. Method 3 (UPDM = 2 and PTRM = 1 and PTRCS = 0)**




### 51.6.2.10 Update Time for Double-Buffering Registers

All channels integrate a double-buffering system in order to prevent an unexpected output waveform while modifying the period, the spread spectrum value, the polarity, the duty-cycle, the dead-times, the output override, and the synchronous channels update period.

This double-buffering system comprises the following update registers:

- PWM Sync Channels Update Period Update Register
- PWM Output Selection Set Update Register
- PWM Output Selection Clear Update Register
- PWM Spread Spectrum Update Register
- PWM Channel Duty Cycle Update Register
- PWM Channel Period Update Register
- PWM Channel Dead Time Update Register
- PWM Channel Mode Update Register

When one of these update registers is written to, the write is stored, but the values are updated only at the next PWM period border. In Left-aligned mode (CALG = 0), the update occurs when the channel counter reaches the period value CPRD. In Center-aligned mode, the update occurs when the channel counter value is decremented and reaches the 0 value.

In Center-aligned mode, it is possible to trigger the update of the polarity and the duty-cycle at the next half period border. This mode concerns the following update registers:

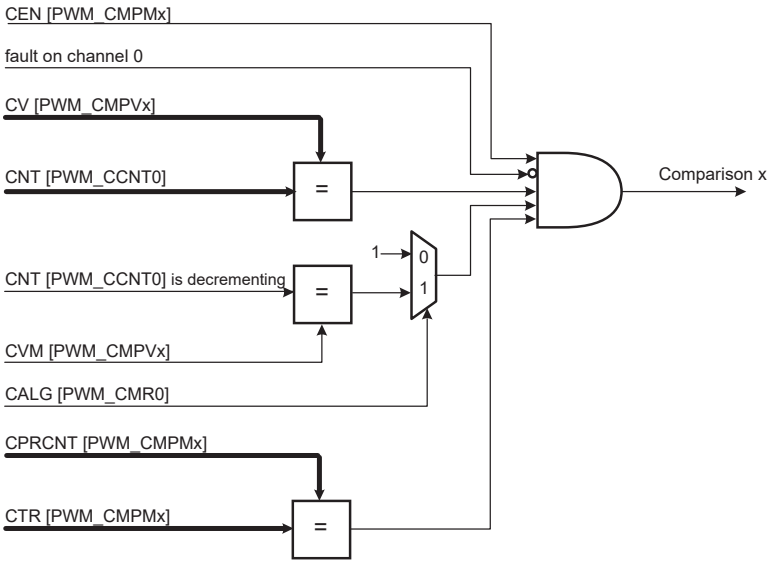
- PWM Channel Duty Cycle Update Register
- PWM Channel Mode Update Register

The update occurs at the first half period following the write of the update register (either when the channel counter value is incrementing and reaches the period value CPRD, or when the channel counter value is decrementing and reaches the 0 value). To activate this mode, the user must write a one to the bit UPDS in the PWM Channel Mode Register.

## 51.6.3 PWM Comparison Units

The PWM provides 8 independent comparison units able to compare a programmed value with the current value of the channel 0 counter (which is the channel counter of all synchronous channels, “Synchronous Channels”). These comparisons are intended to generate pulses on the event lines (used to synchronize ADC, see PWM Event Lines), to generate software interrupts and to trigger DMA Controller transfer requests for the synchronous channels (see Method 3: Automatic write of duty-cycle values and automatic trigger of the update).

**Figure 51-23. Comparison Unit Block Diagram**




The comparison x matches when it is enabled by the bit CEN in the PWM Comparison x Mode Register (PWM_CMPMx for the comparison x) and when the counter of the channel 0 reaches the comparison value defined by the field CV in PWM Comparison x Value Register (PWM_CMPVx for the comparison x). If the counter of the channel 0 is center-aligned (CALG = 1 in PWM Channel Mode Register), the bit CVM in PWM_CMPVx defines if the comparison is made when the counter is counting up or counting down (in Left-alignment mode CALG = 0, this bit is useless).

If a fault is active on the channel 0, the comparison is disabled and cannot match (see Fault Protection).

The user can define the periodicity of the comparison x by the fields CTR and CPR in PWM_CMPMx. The comparison is performed periodically once every CPR+1 periods of the counter of the channel 0, when the value of the comparison period counter CPRCNT in PWM_CMPMx reaches the value defined by CTR. CPR is the maximum value of the comparison period counter CPRCNT. If CPR = CTR = 0, the comparison is performed at each period of the counter of the channel 0.

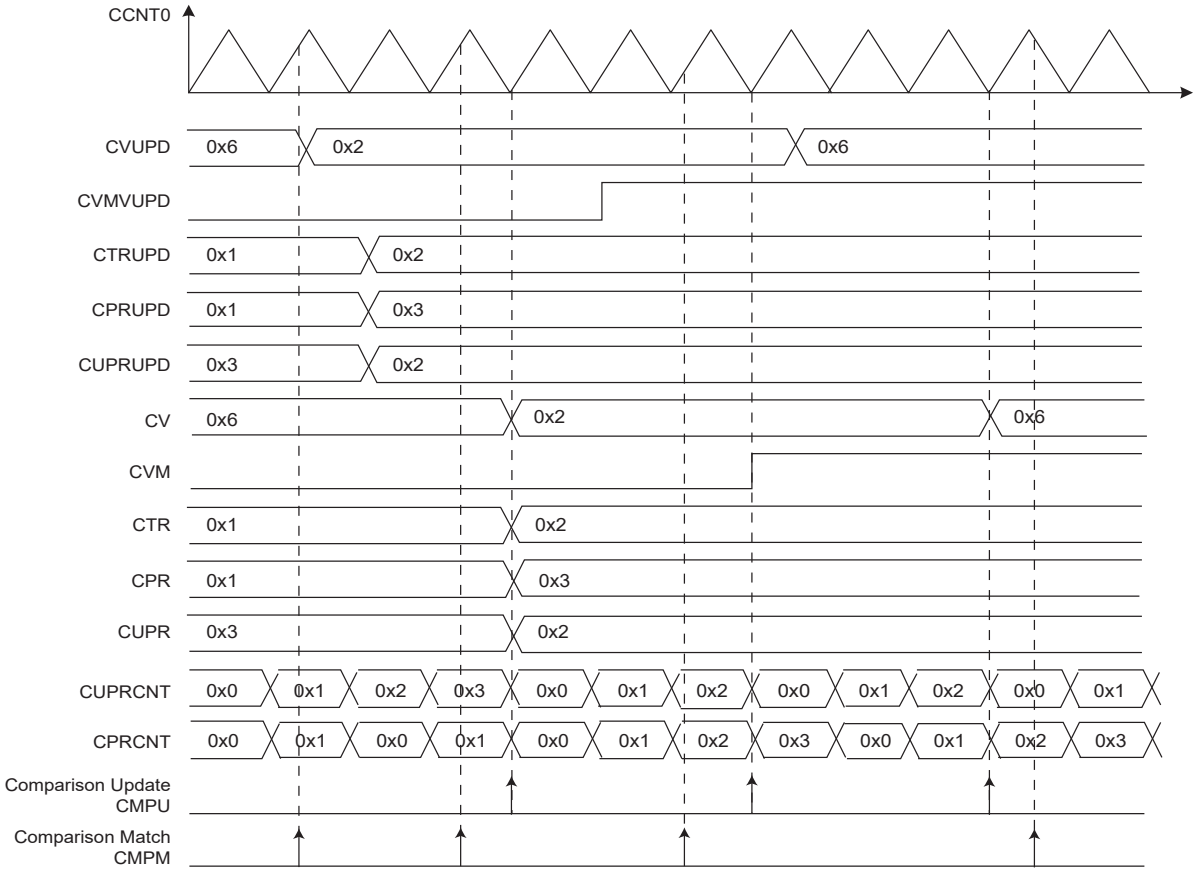
The comparison x configuration can be modified while the channel 0 is enabled by using the PWM Comparison x Mode Update Register (PWM_CMPMUPDx registers for the comparison x). In the same way, the comparison x value can be modified while the channel 0 is enabled by using the PWM Comparison x Value Update Register (PWM_CMPVUPDx registers for the comparison x).

The update of the comparison x configuration and the comparison x value is triggered periodically after the comparison x update period. It is defined by the field CUPR in PWM_CMPMx. The comparison unit has an update period counter independent from the period counter to trigger this update. When the value of the comparison update period counter CUPRCNT (in PWM_CMPMx) reaches the value defined by CUPR, the update is triggered. The comparison x update period CUPR itself can be updated while the channel 0 is enabled by using the PWM_CMPMUPDx register.

**⚠ CAUTION** The write of PWM_CMPVUPDx must be followed by a write of PWM_CMPMUPDx.

The comparison match and the comparison update can be source of an interrupt, but only if it is enabled and not masked. These interrupts can be enabled by the PWM Interrupt Enable Register 2 and disabled by the PWM Interrupt Disable Register 2. The comparison match interrupt and the comparison update interrupt are reset by reading the PWM Interrupt Status Register 2.

**Figure 51-24. Comparison Waveform**




### 51.6.4 PWM Event Lines

The PWM provides 2 independent event lines intended to trigger actions in other peripherals (e.g., for the Analog-to-Digital Converter (ADC)).

A pulse (one cycle of the peripheral clock) is generated on an event line, when at least one of the selected comparisons is matching. The comparisons can be selected or unselected independently by the CSEL bits in the PWM Event Line x Register (PWM_ELMRx for the Event Line x).

An example of event generation is provided in the figure Event Line Generation Waveform (Example).

**Figure 51-25. Event Line Block Diagram**

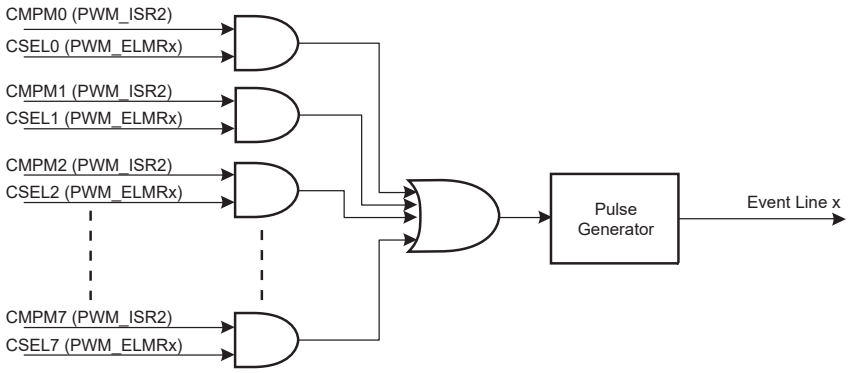
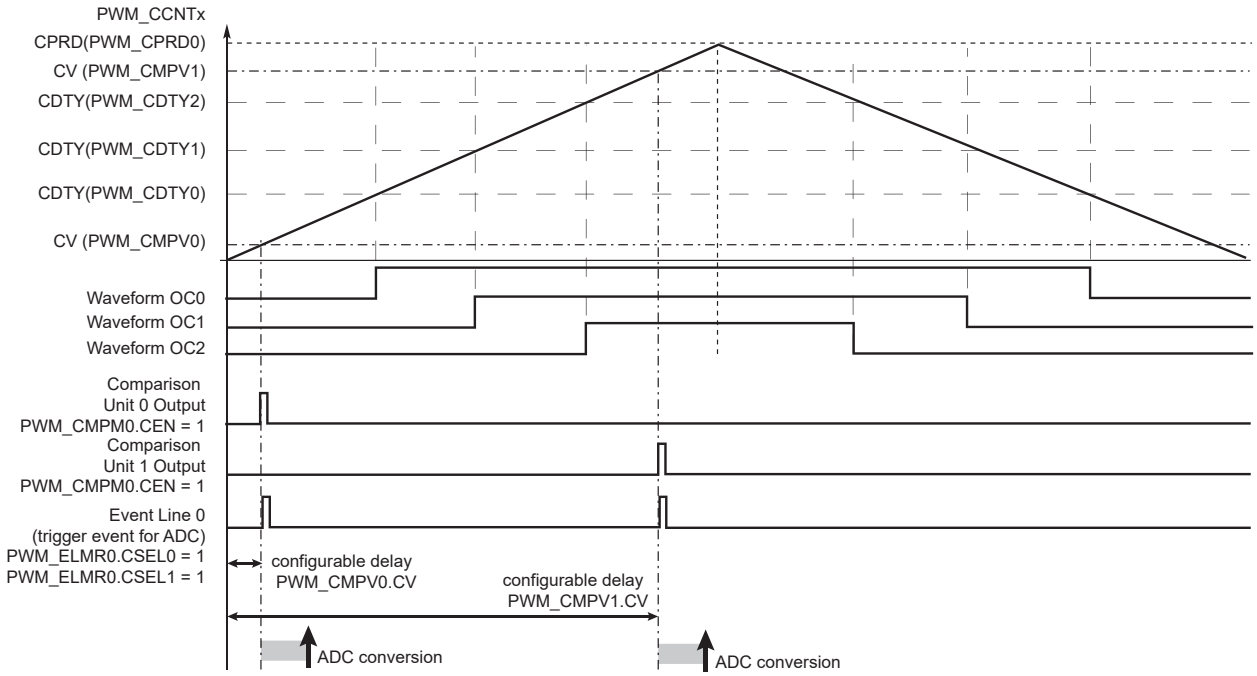


**Figure 51-26. Event Line Generation Waveform (Example)**




### 51.6.5 PWM External Trigger Mode

The PWM channels 1 and 2 can be configured to use an external trigger for generating specific PWM signals. The external trigger source can be selected through the bit TRGSRC of the PWM External Trigger Register (see the table below).

**Table 51-5. External Event Source Selection**

| Channel | Trigger Source Selection | Trigger Source |
|---|---|---|
| 1 | PWM_ETRG1.TRGSRC = 0 | From PWMEXTRG1 input |
| | PWM_ETRG1.TRGSRC = 1 | From Analog Comparator Controller |
| 2 | PWM_ETRG2.TRGSRC = 0 | From PWMEXTRG2 input |
| | PWM_ETRG2.TRGSRC = 1 | From Analog Comparator Controller |

Each external trigger source can be filtered by writing a one to the TRGFILT bit in the corresponding PWM External Trigger Register (PWM_ETRGx).

Each time an external trigger event is detected, the corresponding PWM channel counter value is stored in the MAXCNT field of the PWM_ETRGx register if it is greater than the previously stored value. Reading the PWM_ETRGx register will clear the MAXCNT value.

Three different modes are available for channels 1 and 2 depending on the value of the TRGMODE field of the PWM_ETRGx register:

- TRGMODE = 1: External PWM Reset Mode
- TRGMODE = 2: External PWM Start Mode
- TRGMODE = 3: Cycle-By-Cycle Duty Mode

See the following sections.

This feature is disabled when TRGMODE = 0.

This feature should only be enabled if the corresponding channel is left-aligned (CALG = 0 in PWM Channel Mode Register of channel 1 or 2) and not managed as a synchronous channel (SYNCx = 0 in PWM Sync Channels Mode

Register where x = 1 or 2). Programming the channel to be center-aligned or synchronous while TRGMODE is not 0 could lead to unexpected behavior.

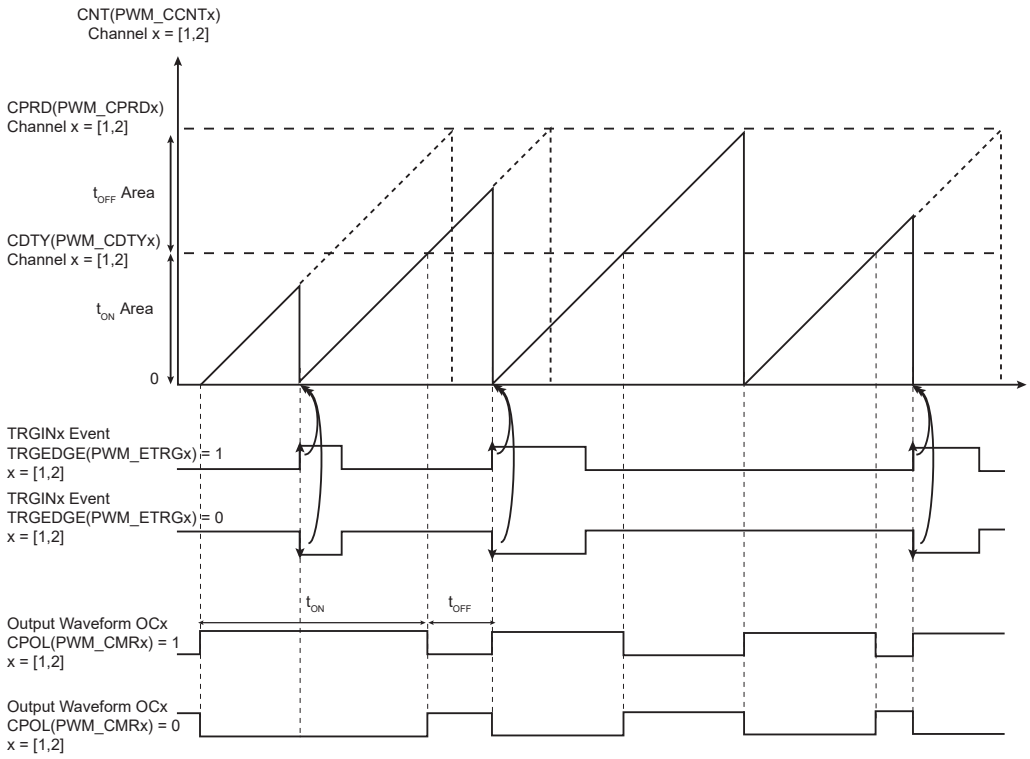
### 51.6.5.1 External PWM Reset Mode

External PWM Reset mode is selected by programming TRGMODE = 1 in the PWM_ETRGx register.

In this mode, when an edge is detected on the PWMEXTRGx input, the internal PWM counter is cleared and a new PWM cycle is restarted. The edge polarity can be selected by programming the TRGEDGE bit in the PWM_ETRGx register. If no trigger event is detected when the internal channel counter has reached the CPRD value in the PWM Channel Period Register, the internal counter is cleared and a new PWM cycle starts.

Note that this mode does not guarantee a constant $t_{ON}$ or $t_{OFF}$ time.

**Figure 51-27. External PWM Reset Mode**



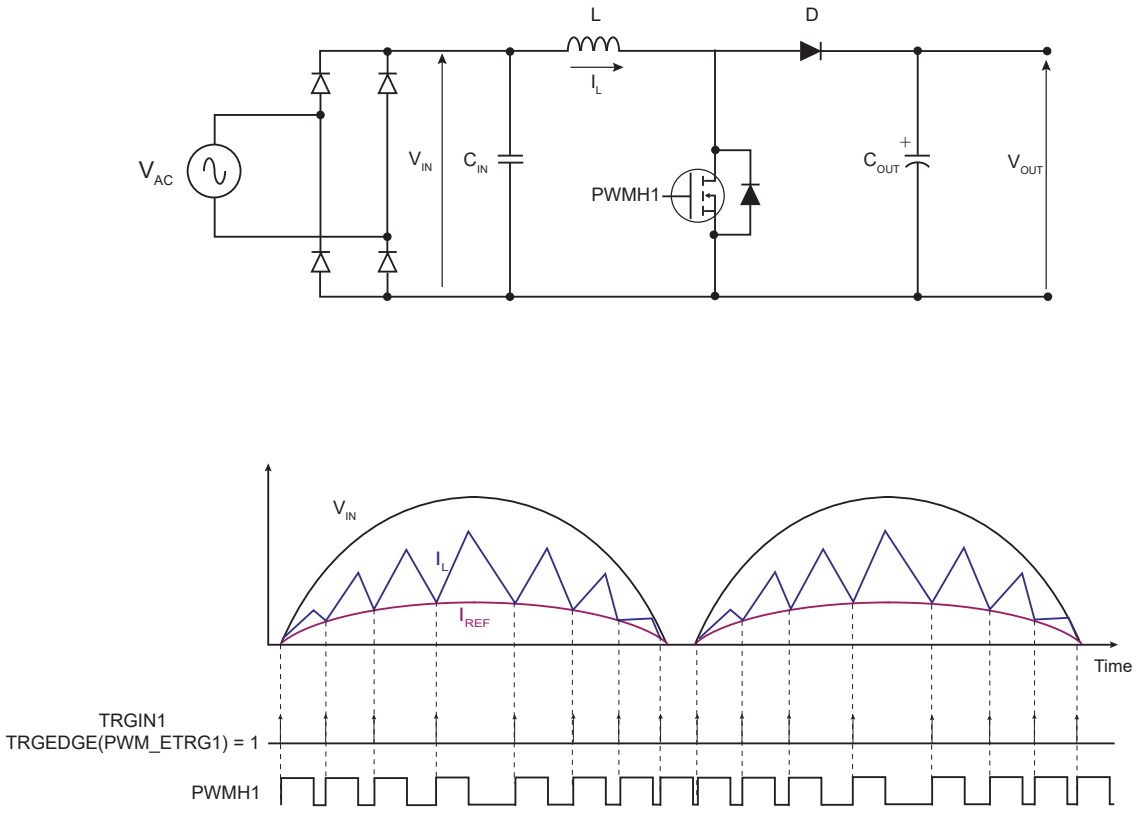

#### 51.6.5.1.1 Application Example

The external PWM Reset mode can be used in power factor correction applications.

In the example below, the external trigger input is the PWMEXTRG1 (therefore the PWM channel used for regulation is the channel 1). The PWM channel 1 period (CPRD in the PWM Channel Period Register of the channel 1) must be programmed so that the TRGIN1 event always triggers before the PWM channel 1 period elapses.

In the figure below, an external circuit (not shown) is required to sense the inductor current $I_L$. The internal PWM counter of the channel 1 is cleared when the inductor current falls below a specific threshold ($I_{REF}$). This starts a new PWM period and increases the inductor current.

**Figure 51-28. External PWM Reset Mode: Power Factor Correction Application**



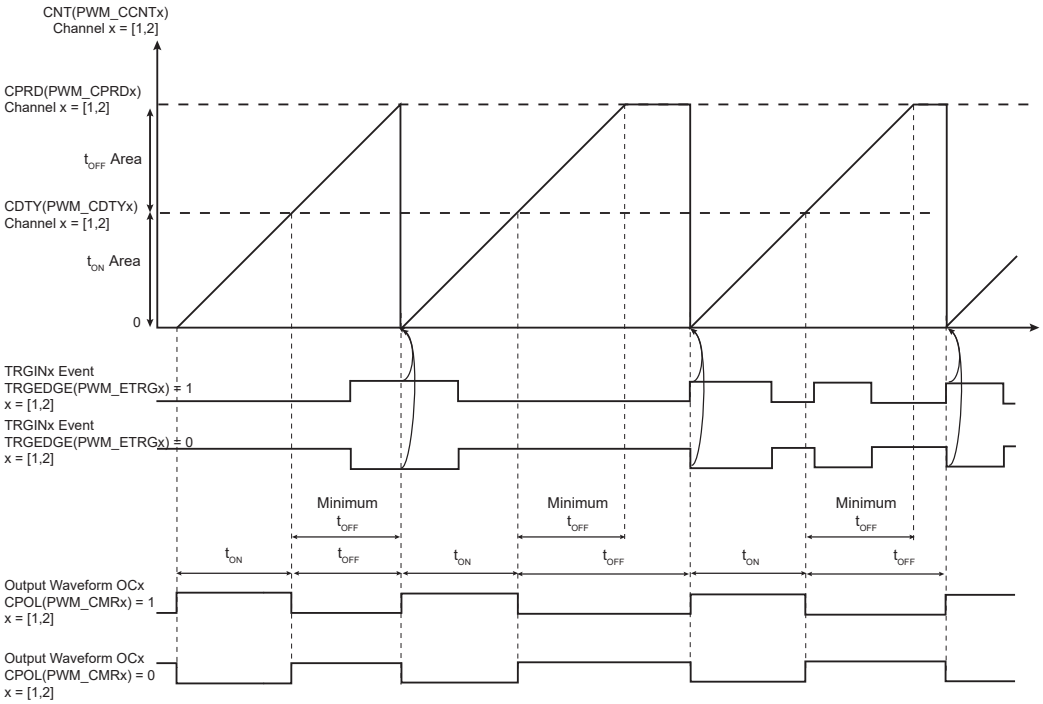

### 51.6.5.2 External PWM Start Mode

External PWM Start mode is selected by programming TRGMODE = 2 in the PWM_ETRGx register.

In this mode, the internal PWM counter can only be reset once it has reached the CPRD value in the PWM Channel Period Register and when the correct level is detected on the corresponding external trigger input. Both conditions have to be met to start a new PWM period. The active detection level is defined by the bit TRGEDGE of the PWM_ETRGx register.

Note that this mode guarantees a constant $t_{ON}$ time and a minimum $t_{OFF}$ time.

**Figure 51-29. External PWM Start Mode**



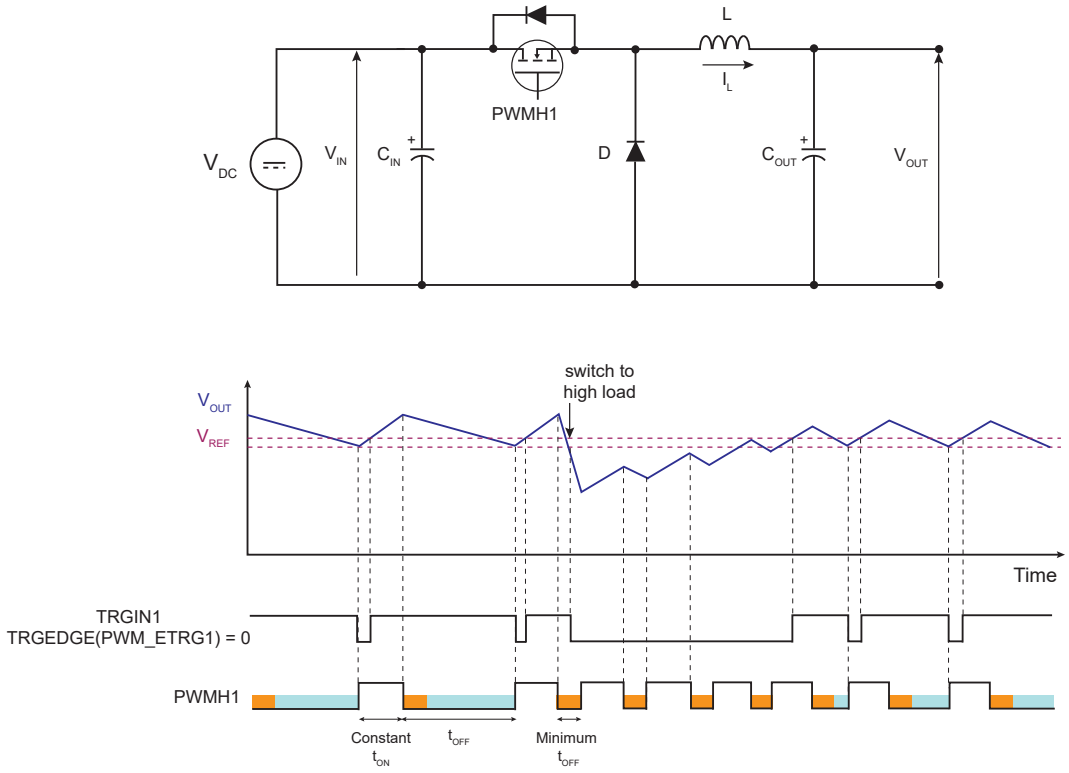

#### 51.6.5.2.1 Application Example

The external PWM Start mode generates a modulated frequency PWM signal with a constant active level duration ($t_{ON}$) and a minimum inactive level duration (minimum $t_{OFF}$).

The $t_{ON}$ time is defined by the CDTY value in the PWM Channel Duty Cycle Register. The minimum $t_{OFF}$ time is defined by CDTY - CPRD (PWM Channel Period Register). This mode can be useful in Buck DC/DC Converter applications.

When the output voltage $V_{OUT}$ is above a specific threshold (Vref), the PWM inactive level is maintained as long as $V_{OUT}$ remains above this threshold. If $V_{OUT}$ is below this specific threshold, this mode guarantees a minimum $t_{OFF}$ time required for MOSFET driving (see the figure below).

**Figure 51-30. External PWM Start Mode: Buck DC/DC Converter**




### 51.6.5.3 Cycle-By-Cycle Duty Mode

#### 51.6.5.3.1 Description

Cycle-by-cycle duty mode is selected by programming TRGMODE = 3 in PWM_ETRGx.

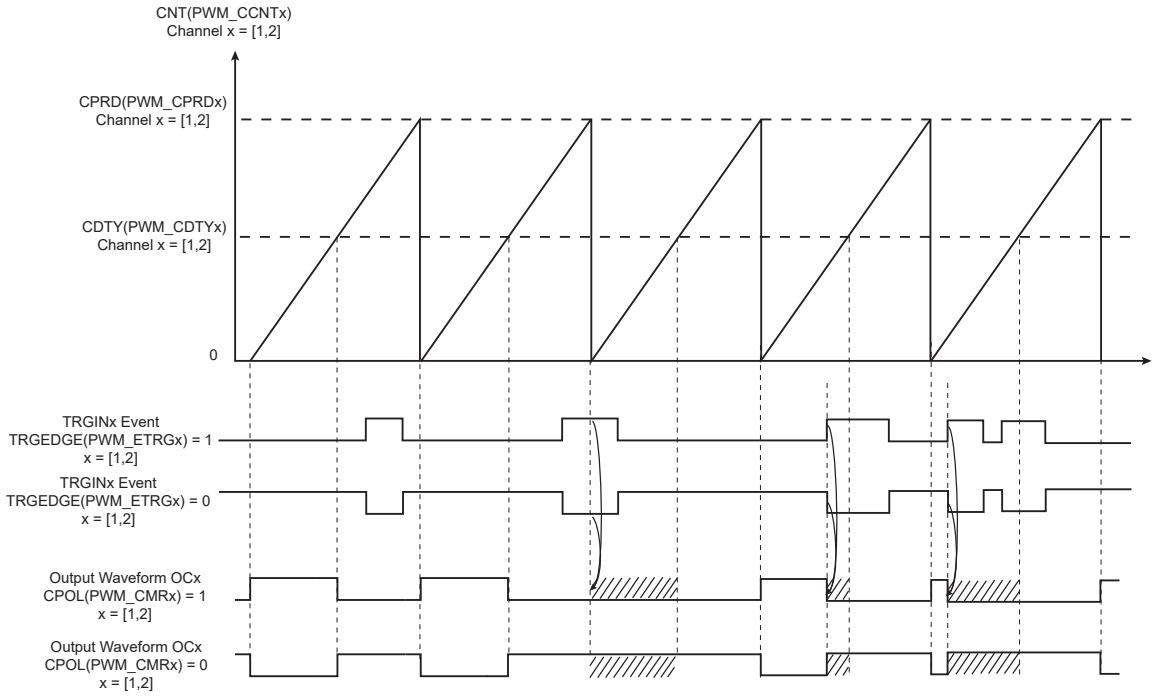
In this mode, the PWM frequency is constant and is defined by the CPRD value in the PWM Channel Period Register.

An external trigger event has no effect on the PWM output if it occurs while the internal PWM counter value is above the CDTY value of the PWM Channel Duty Cycle Register.

If the internal PWM counter value is below the value of CDTY of the PWM Channel Duty Cycle Register, an external trigger event makes the PWM output inactive.

The external trigger event can be detected on rising or falling edge according to the TRGEDGE bit in PWM_ETRGx.

**Figure 51-31. Cycle-By-Cycle Duty Mode**

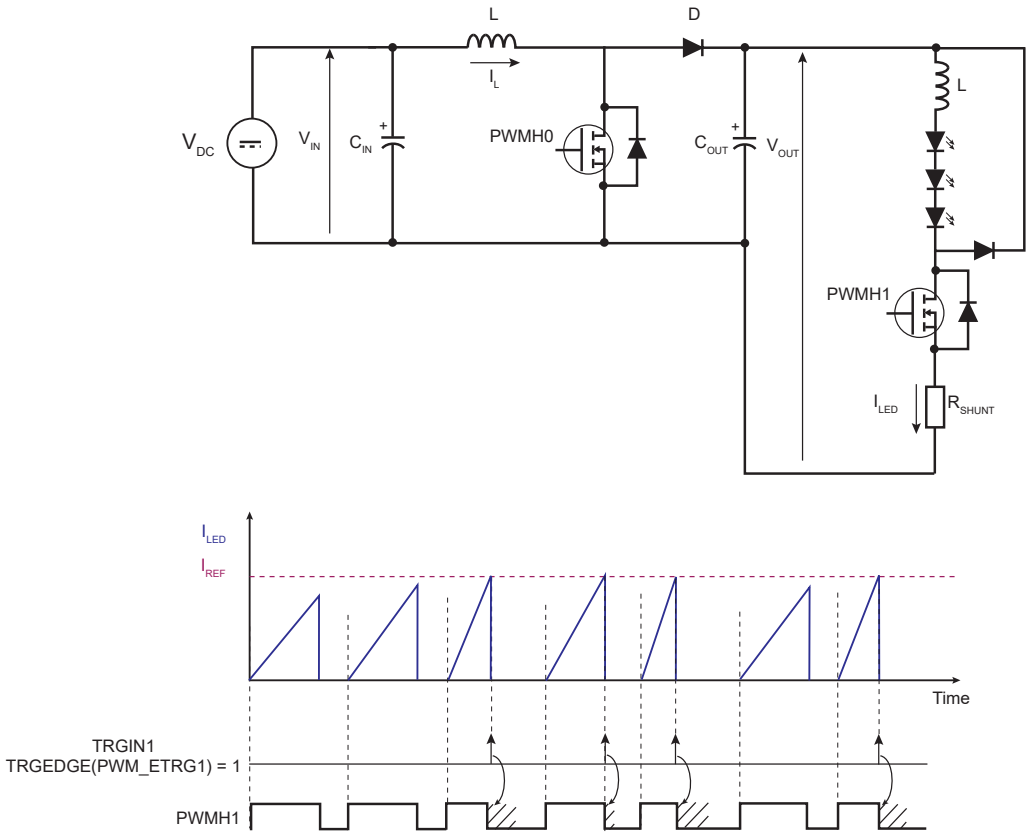



#### 51.6.5.3.2 Application Example

The figure below illustrates an application example of the Cycle-by-cycle Duty mode.

In an LED string control circuit, Cycle-by-cycle Duty mode can be used to automatically limit the current in the LED string.

**Figure 51-32. Cycle-By-Cycle Duty Mode: LED String Control**




### 51.6.5.4 Leading-Edge Blanking (LEB)

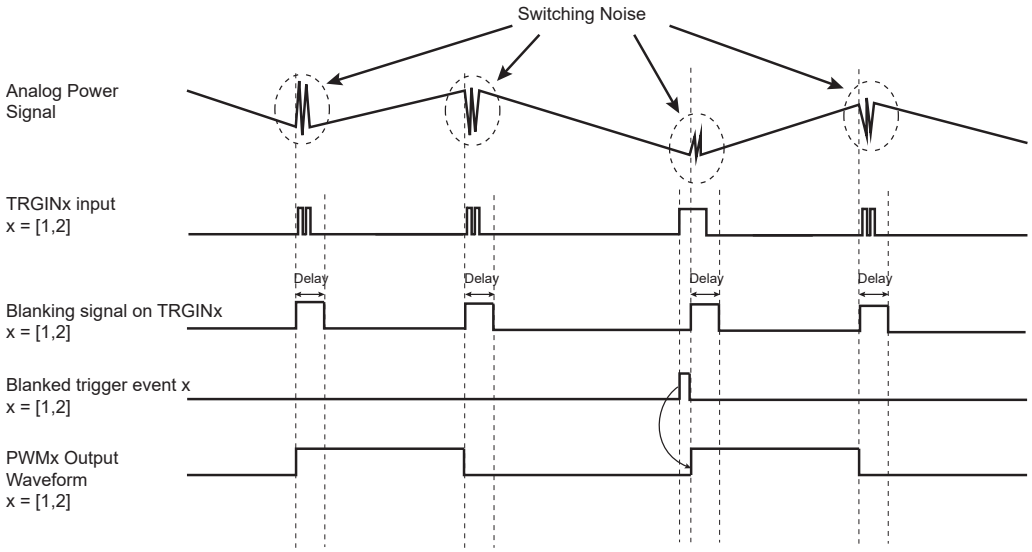
PWM channels 1 and 2 support leading-edge blanking. Leading-edge blanking masks the external trigger input when a transient occurs on the corresponding PWM output. It masks potential spurious external events due to power transistor switching.

The blanking delay on each external trigger input is configured by programming the LEBDELAYx in the PWM Leading-Edge Blanking Register.

The LEB can be enabled on both the rising and the falling edges for the PWMH and PWML outputs through the bits PWMLFEN, PWMLREN, PWMHFEN, PWMHREN.

Any event on the PWMEXTRGx input which occurs during the blanking time is ignored.

**Figure 51-33. Leading-Edge Blanking**




## 51.6.6 PWM Controller Operations

### 51.6.6.1 Initialization

Before enabling the channels, they must be configured by the software application as described below:

- Unlock User Interface by writing the WPCMD field in PWM_WPCR.
- Configuration of the clock generator (DIVA, PREA, DIVB, PREB in the PWM_CLK register if required).
- Selection of the clock for each channel (CPRE field in PWM_CMRx)
- Configuration of the waveform alignment for each channel (CALG field in PWM_CMRx)
- Selection of the counter event selection (if CALG = 1) for each channel (CES field in PWM_CMRx)
- Configuration of the output waveform polarity for each channel (CPOL bit in PWM_CMRx)
- Configuration of the period for each channel (CPRD in the PWM_CPRDx register). Writing in PWM_CPRDx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CPRDUPDx register to update PWM_CPRDx as explained below.
- Configuration of the duty-cycle for each channel (CDTY in the PWM_CDTYx register). Writing in PWM_CDTYx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CDTYUPDx register to update PWM_CDTYx as explained below.
- Configuration of the dead-time generator for each channel (DTH and DTL in PWM_DTx) if enabled (DTE bit in PWM_CMRx). Writing in the PWM_DTx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_DTUPDx register to update PWM_DTx
- Selection of the synchronous channels (SYNCx in the PWM_SCM register)
- Selection of the moment when the WRDY flag and the corresponding DMA Controller transfer request are set (PTRM and PTRCS in the PWM_SCM register)
- Configuration of the Update mode (UPDM in PWM_SCM register)
- Configuration of the update period (UPR in PWM_SCUP register) if needed
- Configuration of the comparisons (PWM_CMPVx and PWM_CMPMx)
- Configuration of the event lines (PWM_ELMRx)
- Configuration of the fault inputs polarity (FPOL in PWM_FMR)
- Configuration of the fault protection (FMOD and FFIL in PWM_FMR, PWM_FPV and PWM_FPE)
- Enable of the interrupts (writing CHIDx and FCHIDx in PWM_IER1, and writing WRDY, UNRE, CMPMx and CMPUx in PWM_IER2)
- Enable of the PWM channels (writing CHIDx in the PWM_ENA register)

© 2021 Microchip Technology Inc. and its subsidiaries

#### 51.6.6.2 Source Clock Selection Criteria

The large number of source clocks can make selection difficult. The relationship between the value in the PWM Channel Period Register (PWM_CPRDx) and the PWM Channel Duty Cycle Register (PWM_CDTYx) helps the user select the appropriate clock. The event number written in the Period Register gives the PWM accuracy. The Duty-Cycle quantum cannot be lower than 1/CPRDx value. The higher the value of PWM_CPRDx, the greater the PWM accuracy.

For example, if the user sets 15 (in decimal) in PWM_CPRDx, the user is able to set a value from between 1 up to 14 in PWM_CDTYx. The resulting duty-cycle quantum cannot be lower than 1/15 of the PWM period.

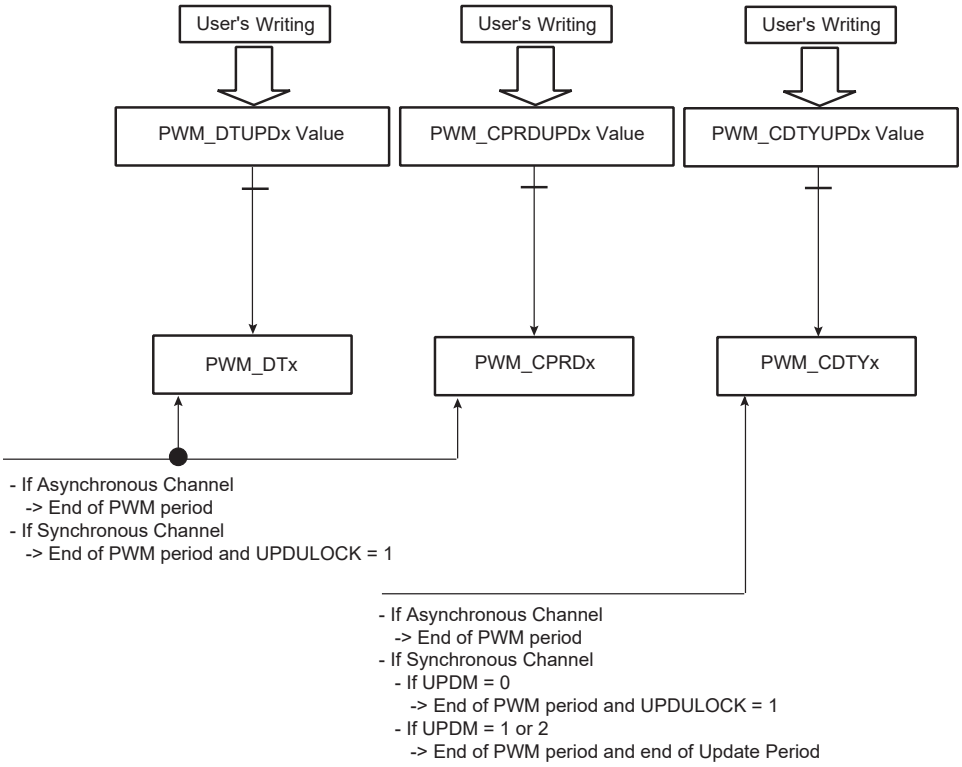
#### 51.6.6.3 Changing the Duty-Cycle, the Period and the Dead-Times

It is possible to modulate the output waveform duty-cycle, period and dead-times.

To prevent unexpected output waveform, the user must use the PWM Channel Duty Cycle Update Register (PWM_CDTYUPDx), the PWM Channel Period Update Register (PWM_CPRDUPDx) and the PWM Channel Dead Time Update Register (PWM_DTUPDx) to change waveform parameters while the channel is still enabled.

- If the channel is an asynchronous channel (SYNCx = 0 in PWM Sync Channels Mode Register (PWM_SCM)), these registers hold the new period, duty-cycle and dead-times values until the end of the current PWM period and update the values for the next period.
- If the channel is a synchronous channel and update method 0 is selected (SYNCx = 1 and UPDM = 0 in PWM_SCM register), these registers hold the new period, duty-cycle and dead-times values until the bit UPDULOCK is written at ‘1’ (in PWM Sync Channels Update Control Register (PWM_SCUC)) and the end of the current PWM period, then update the values for the next period.
- If the channel is a synchronous channel and update method 1 or 2 is selected (SYNCx = 1 and UPDM = 1 or 2 in PWM_SCM register):
  - registers PWM_CPRDUPDx and PWM_DTUPDx hold the new period and dead-times values until the bit UPDULOCK is written at ‘1’ (in PWM_SCUC) and the end of the current PWM period, then update the values for the next period.
  - register PWM_CDTYUPDx holds the new duty-cycle value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in PWM Sync Channels Update Period Register (PWM_SCUP)) and the end of the current PWM period, then updates the value for the next period.
    **Note:** If the update registers PWM_CDTYUPDx, PWM_CPRDUPDx and PWM_DTUPDx are written several times between two updates, only the last written value is taken into account.

**Figure 51-34. Synchronized Period, Duty-Cycle and Dead-Time Update**




### 51.6.6.4 Changing the Update Period of Synchronous Channels

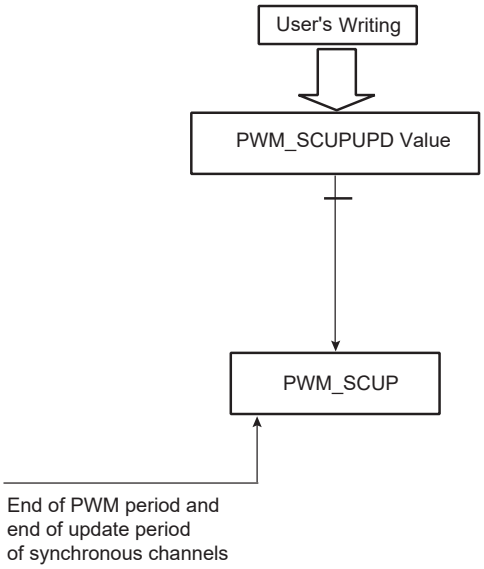
It is possible to change the update period of synchronous channels while they are enabled. See Method 2: Manual write of duty-cycle values and automatic trigger of the update and Method 3: Automatic write of duty-cycle values and automatic trigger of the update .

To prevent an unexpected update of the synchronous channels registers, the user must use the PWM Sync Channels Update Period Update Register (PWM_SCUPUPD) to change the update period of synchronous channels while they are still enabled. This register holds the new value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in PWM_SCUP) and the end of the current PWM period, then updates the value for the next period.

**Notes:**

1. If the update register PWM_SCUPUPD is written several times between two updates, only the last written value is taken into account.
2. Changing the update period does make sense only if there is one or more synchronous channels and if the update method 1 or 2 is selected (UPDM = 1 or 2 in PWM Sync Channels Mode Register).

**Figure 51-35. Synchronized Update of Update Period Value of Synchronous Channels**




### 51.6.6.5 Changing the Comparison Value and the Comparison Configuration

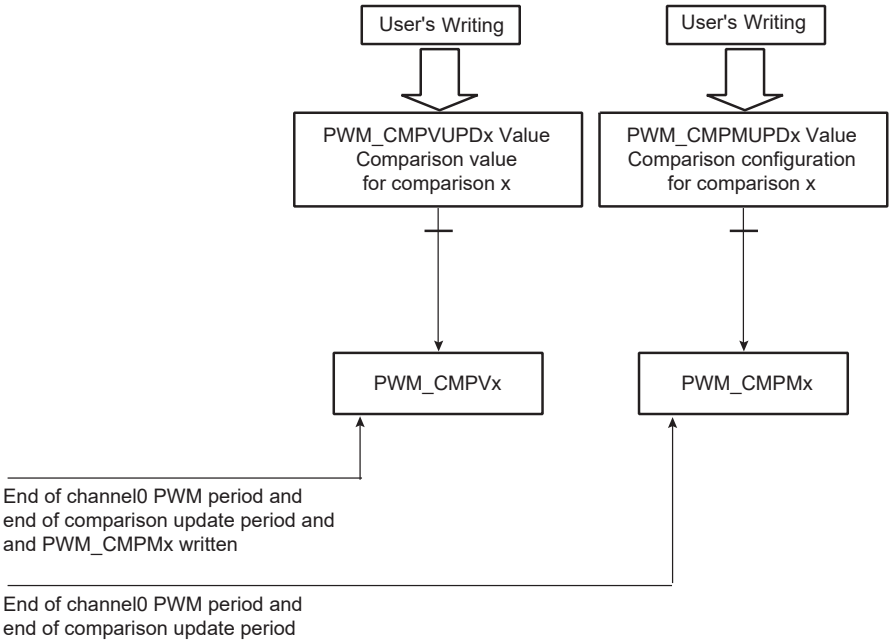
It is possible to change the comparison values and the comparison configurations while the channel 0 is enabled (see PWM Comparison Units).

To prevent unexpected comparison match, the user must use the PWM Comparison x Value Update Register (PWM_CMPVUPDx) and the PWM Comparison x Mode Update Register (PWM_CMPMUPDx) to change, respectively, the comparison values and the comparison configurations while the channel 0 is still enabled. These registers hold the new values until the end of the comparison update period (when CUPRCNT is equal to CUPR in PWM Comparison x Mode Register (PWM_CMPMx) and the end of the current PWM period, then update the values for the next period.

**⚠ CAUTION** The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

**Note:** If the update registers PWM_CMPVUPDx and PWM_CMPMUPDx are written several times between two updates, only the last written value are taken into account.

**Figure 51-36. Synchronized Update of Comparison Values and Configurations**




### 51.6.6.6 Interrupt Sources

Depending on the interrupt mask in PWM_IMR1 and PWM_IMR2, an interrupt can be generated at the end of the corresponding channel period (CHIDx in the PWM Interrupt Status Register 1 (PWM_ISR1)), after a fault event (FCHIDx in PWM_ISR1), after a comparison match (CMPMx in PWM_ISR2), after a comparison update (CMPUx in PWM_ISR2) or according to the Transfer mode of the synchronous channels (WRDY and UNRE in PWM_ISR2).

If the interrupt is generated by the flags CHIDx or FCHIDx, the interrupt remains active until a read operation in PWM_ISR1 occurs.

If the interrupt is generated by the flags WRDY or UNRE or CMPMx or CMPUx, the interrupt remains active until a read operation in PWM_ISR2 occurs.

A channel interrupt is enabled by setting the corresponding bit in PWM_IER1 and PWM_IER2. A channel interrupt is disabled by setting the corresponding bit in PWM_IDR1 and PWM_IDR2.

## 51.6.7 Register Write Protection

To prevent any single software error that may corrupt PWM behavior, the registers listed below can be write-protected by writing the field WPCMD in the PWM Write Protection Control Register (PWM_WPCR). They are divided into six groups:

- Register group 0:
  - PWM Clock Register
- Register group 1:
  - PWM Disable Register
  - PWM Interrupt Enable Register 1
  - PWM Interrupt Disable Register 1
  - PWM Interrupt Enable Register 2
  - PWM Interrupt Disable Register 2
- Register group 2:
  - PWM Sync Channels Mode Register
  - PWM Channel Mode Register
  - PWM Stepper Motor Mode Register
  - PWM Fault Protection Value Register 2
  - PWM Leading-Edge Blanking Register

  - PWM Channel Mode Update Register
- Register group 3:
  - PWM Spread Spectrum Register
  - PWM Spread Spectrum Update Register
  - PWM Channel Period Register
  - PWM Channel Period Update Register
- Register group 4:
  - PWM Channel Dead Time Register
  - PWM Channel Dead Time Update Register
- Register group 5:
  - PWM Fault Mode Register
  - PWM Fault Protection Value Register 1

There are two types of write protection:

- SW write protection—can be enabled or disabled by software
- HW write protection—can be enabled by software but only disabled by a hardware reset of the PWM controller

Both types of write protection can be applied independently to a particular register group by means of the WPCMD and WPRGx fields in PWM_WPCR. If at least one type of write protection is active, the register group is write-protected. The value of field WPCMD defines the action to be performed:

- 0: Disables SW write protection of the register groups of which the bit WPRGx is at ‘1’
- 1: Enables SW write protection of the register groups of which the bit WPRGx is at ‘1’
- 2: Enables HW write protection of the register groups of which the bit WPRGx is at ‘1’

At any time, the user can determine whether SW or HW write protection is active in a particular register group by the fields WPSWS and WPHWS in the PWM Write Protection Status Register (PWM_WPSR).

If a write access to a write-protected register is detected, the WPVS flag in PWM_WPSR is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS and WPVSRC fields are automatically cleared after reading PWM_WPSR.

© 2021 Microchip Technology Inc.
and its subsidiaries

## 51.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | PWM_CLK | 7:0 | DIVA[7:0] | | | | | | | |
| | | 15:8 | | | | | PREA[3:0] | | | |
| | | 23:16 | DIVB[7:0] | | | | | | | |
| | | 31:24 | | | | | PREB[3:0] | | | |
| 0x04 | PWM_ENA | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | PWM_DIS | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C | PWM_SR | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x10 | PWM_IER1 | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| | | 31:24 | | | | | | | | |
| 0x14 | PWM_IDR1 | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| | | 31:24 | | | | | | | | |
| 0x18 | PWM_IMR1 | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| | | 31:24 | | | | | | | | |
| 0x1C | PWM_ISR1 | 7:0 | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| | | 31:24 | | | | | | | | |
| 0x20 | PWM_SCM | 7:0 | | | | | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | PTRCS[2:0] | | | PTRM | | | UPDM[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x24 | PWM_DMAR | 7:0 | DMADUTY[7:0] | | | | | | | |
| | | 15:8 | DMADUTY[15:8] | | | | | | | |
| | | 23:16 | DMADUTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | PWM_SCUC | 7:0 | | | | | | | | UPDULOCK |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | PWM_SCUP | 7:0 | UPRCNT[3:0] | | | | UPR[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | PWM_SCUPUPD | 7:0 | | | | | UPRUPD[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 | PWM_IER2 | 7:0 | | | | | UNRE | | | WRDY |
| | | 15:8 | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| | | 23:16 | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x38 | PWM_IDR2 | 7:0 | | | | | UNRE | | | WRDY |
| | | 15:8 | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| | | 23:16 | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| | | 31:24 | | | | | | | | |
| 0x3C | PWM_IMR2 | 7:0 | | | | | UNRE | | | WRDY |
| | | 15:8 | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| | | 23:16 | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| | | 31:24 | | | | | | | | |
| 0x40 | PWM_ISR2 | 7:0 | | | | | UNRE | | | WRDY |
| | | 15:8 | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| | | 23:16 | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| | | 31:24 | | | | | | | | |
| 0x44 | PWM_OOV | 7:0 | | | | | OOVH3 | OOVH2 | OOVH1 | OOVH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OOVL3 | OOVL2 | OOVL1 | OOVL0 |
| | | 31:24 | | | | | | | | |
| 0x48 | PWM_OS | 7:0 | | | | | OSH3 | OSH2 | OSH1 | OSH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSL3 | OSL2 | OSL1 | OSL0 |
| | | 31:24 | | | | | | | | |
| 0x4C | PWM_OSS | 7:0 | | | | | OSSH3 | OSSH2 | OSSH1 | OSSH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSSL3 | OSSL2 | OSSL1 | OSSL0 |
| | | 31:24 | | | | | | | | |
| 0x50 | PWM_OSC | 7:0 | | | | | OSCH3 | OSCH2 | OSCH1 | OSCH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSCL3 | OSCL2 | OSCL1 | OSCL0 |
| | | 31:24 | | | | | | | | |
| 0x54 | PWM_OSSUPD | 7:0 | | | | | OSSUPH3 | OSSUPH2 | OSSUPH1 | OSSUPH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSSUPL3 | OSSUPL2 | OSSUPL1 | OSSUPL0 |
| | | 31:24 | | | | | | | | |
| 0x58 | PWM_OSCUPD | 7:0 | | | | | OSCUPH3 | OSCUPH2 | OSCUPH1 | OSCUPH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | OSCUPL3 | OSCUPL2 | OSCUPL1 | OSCUPL0 |
| | | 31:24 | | | | | | | | |
| 0x5C | PWM_FMR | 7:0 | FPOL[7:0] | | | | | | | |
| | | 15:8 | FMOD[7:0] | | | | | | | |
| | | 23:16 | FFIL[7:0] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x60 | PWM_FSR | 7:0 | FIV[7:0] | | | | | | | |
| | | 15:8 | FS[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x64 | PWM_FCR | 7:0 | FCLR[7:0] | | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x68 | PWM_FPV1 | 7:0 | | | | | FPVH3 | FPVH2 | FPVH1 | FPVH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FPVL3 | FPVL2 | FPVL1 | FPVL0 |
| | | 31:24 | | | | | | | | |
| 0x6C | PWM_FPE | 7:0 | FPE0[7:0] | | | | | | | |
| | | 15:8 | FPE1[7:0] | | | | | | | |
| | | 23:16 | FPE2[7:0] | | | | | | | |
| | | 31:24 | FPE3[7:0] | | | | | | | |
| 0x70 ... 0x7B | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x7C | PWM_ELMR0 | 7:0 | CSEL7 | CSEL6 | CSEL5 | CSEL4 | CSEL3 | CSEL2 | CSEL1 | CSEL0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x80 | PWM_ELMR1 | 7:0 | CSEL7 | CSEL6 | CSEL5 | CSEL4 | CSEL3 | CSEL2 | CSEL1 | CSEL0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x84 ... 0x9F | Reserved | | | | | | | | | |
| 0xA0 | PWM_SSPR | 7:0 | SPRD[7:0] | | | | | | | |
| | | 15:8 | SPRD[15:8] | | | | | | | |
| | | 23:16 | SPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | SPRDM |
| 0xA4 | PWM_SSPUP | 7:0 | SPRDUP[7:0] | | | | | | | |
| | | 15:8 | SPRDUP[15:8] | | | | | | | |
| | | 23:16 | SPRDUP[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA8 ... 0xAF | Reserved | | | | | | | | | |
| 0xB0 | PWM_SMMR | 7:0 | | | | | | | GCEN1 | GCEN0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | DOWN1 | DOWN0 |
| | | 31:24 | | | | | | | | |
| 0xB4 ... 0xBF | Reserved | | | | | | | | | |
| 0xC0 | PWM_FPV2 | 7:0 | | | | | FPZH3 | FPZH2 | FPZH1 | FPZH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | FPZL3 | FPZL2 | FPZL1 | FPZL0 |
| | | 31:24 | | | | | | | | |
| 0xC4 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | PWM_WPCR | 7:0 | WPRG5 | WPRG4 | WPRG3 | WPRG2 | WPRG1 | WPRG0 | WPCMD[1:0] | |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | PWM_WPSR | 7:0 | WPVS | | WPSWS5 | WPSWS4 | WPSWS3 | WPSWS2 | WPSWS1 | WPSWS0 |
| | | 15:8 | | | WPHWS5 | WPHWS4 | WPHWS3 | WPHWS2 | WPHWS1 | WPHWS0 |
| | | 23:16 | WPVSRC[7:0] | | | | | | | |
| | | 31:24 | WPVSRC[15:8] | | | | | | | |
| 0xEC ... 0x012F | Reserved | | | | | | | | | |
| 0x0130 | PWM_CMPV0 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0134 | PWM_CMPVUPD0 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0138 | PWM_CMPM0 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x013C | PWM_CMPMUPD0 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0140 | PWM_CMPV1 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0144 | PWM_CMPVUPD1 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0148 | PWM_CMPM1 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x014C | PWM_CMPMUPD1 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0150 | PWM_CMPV2 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0154 | PWM_CMPVUPD2 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0158 | PWM_CMPM2 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x015C | PWM_CMPMUPD2 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0160 | PWM_CMPV3 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0164 | PWM_CMPVUPD3 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0168 | PWM_CMPM3 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x016C | PWM_CMPMUPD3 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0170 | PWM_CMPV4 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0174 | PWM_CMPVUPD4 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0178 | PWM_CMPM4 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x017C | PWM_CMPMUPD4 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0180 | PWM_CMPV5 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0184 | PWM_CMPVUPD5 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0188 | PWM_CMPM5 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x018C | PWM_CMPMUPD5 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x0190 | PWM_CMPV6 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x0194 | PWM_CMPVUPD6 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x0198 | PWM_CMPM6 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x019C | PWM_CMPMUPD6 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x01A0 | PWM_CMPV7 | 7:0 | CV[7:0] | | | | | | | |
| | | 15:8 | CV[15:8] | | | | | | | |
| | | 23:16 | CV[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVM |
| 0x01A4 | PWM_CMPVUPD7 | 7:0 | CVUPD[7:0] | | | | | | | |
| | | 15:8 | CVUPD[15:8] | | | | | | | |
| | | 23:16 | CVUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | CVMUPD |
| 0x01A8 | PWM_CMPM7 | 7:0 | CTR[3:0] | | | | | | | CEN |
| | | 15:8 | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| | | 23:16 | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x01AC | PWM_CMPMUPD7 | 7:0 | CTRUPD[3:0] | | | | | | | CENUPD |
| | | 15:8 | | | | | CPRUPD[3:0] | | | |
| | | 23:16 | | | | | CUPRUPD[3:0] | | | |
| | | 31:24 | | | | | | | | |
| 0x01B0 ... 0x01FF | Reserved | | | | | | | | | |
| 0x0200 | PWM_CMR0 | 7:0 | | | | | CPRE[3:0] | | | |
| | | 15:8 | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| | | 23:16 | | | | | PPM | DTLI | DTHI | DTE |
| | | 31:24 | | | | | | | | |
| 0x0204 | PWM_CDTY0 | 7:0 | CDTY[7:0] | | | | | | | |
| | | 15:8 | CDTY[15:8] | | | | | | | |
| | | 23:16 | CDTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0208 | PWM_CDTYUPD0 | 7:0 | CDTYUPD[7:0] | | | | | | | |
| | | 15:8 | CDTYUPD[15:8] | | | | | | | |
| | | 23:16 | CDTYUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x020C | PWM_CPRD0 | 7:0 | CPRD[7:0] | | | | | | | |
| | | 15:8 | CPRD[15:8] | | | | | | | |
| | | 23:16 | CPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0210 | PWM_CPRDUPD0 | 7:0 | CPRDUPD[7:0] | | | | | | | |
| | | 15:8 | CPRDUPD[15:8] | | | | | | | |
| | | 23:16 | CPRDUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0214 | PWM_CCNT0 | 7:0 | CNT[7:0] | | | | | | | |
| | | 15:8 | CNT[15:8] | | | | | | | |
| | | 23:16 | CNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0218 | PWM_DT0 | 7:0 | DTH[7:0] | | | | | | | |
| | | 15:8 | DTH[15:8] | | | | | | | |
| | | 23:16 | DTL[7:0] | | | | | | | |
| | | 31:24 | DTL[15:8] | | | | | | | |
| 0x021C | PWM_DTUPD0 | 7:0 | DTHUPD[7:0] | | | | | | | |
| | | 15:8 | DTHUPD[15:8] | | | | | | | |
| | | 23:16 | DTLUPD[7:0] | | | | | | | |
| | | 31:24 | DTLUPD[15:8] | | | | | | | |
| 0x0220 | PWM_CMR1 | 7:0 | | | | | CPRE[3:0] | | | |
| | | 15:8 | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| | | 23:16 | | | | | PPM | DTLI | DTHI | DTE |
| | | 31:24 | | | | | | | | |
| 0x0224 | PWM_CDTY1 | 7:0 | CDTY[7:0] | | | | | | | |
| | | 15:8 | CDTY[15:8] | | | | | | | |
| | | 23:16 | CDTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0228 | PWM_CDTYUPD1 | 7:0 | CDTYUPD[7:0] | | | | | | | |
| | | 15:8 | CDTYUPD[15:8] | | | | | | | |
| | | 23:16 | CDTYUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x022C | PWM_CPRD1 | 7:0 | CPRD[7:0] | | | | | | | |
| | | 15:8 | CPRD[15:8] | | | | | | | |
| | | 23:16 | CPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0230 | PWM_CPRDUPD1 | 7:0 | CPRDUPD[7:0] | | | | | | | |
| | | 15:8 | CPRDUPD[15:8] | | | | | | | |
| | | 23:16 | CPRDUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0234 | PWM_CCNT1 | 7:0 | CNT[7:0] | | | | | | | |
| | | 15:8 | CNT[15:8] | | | | | | | |
| | | 23:16 | CNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0238 | PWM_DT1 | 7:0 | DTH[7:0] | | | | | | | |
| | | 15:8 | DTH[15:8] | | | | | | | |
| | | 23:16 | DTL[7:0] | | | | | | | |
| | | 31:24 | DTL[15:8] | | | | | | | |
| 0x023C | PWM_DTUPD1 | 7:0 | DTHUPD[7:0] | | | | | | | |
| | | 15:8 | DTHUPD[15:8] | | | | | | | |
| | | 23:16 | DTLUPD[7:0] | | | | | | | |
| | | 31:24 | DTLUPD[15:8] | | | | | | | |
| 0x0240 | PWM_CMR2 | 7:0 | | | | | CPRE[3:0] | | | |
| | | 15:8 | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| | | 23:16 | | | | | PPM | DTLI | DTHI | DTE |
| | | 31:24 | | | | | | | | |
| 0x0244 | PWM_CDTY2 | 7:0 | CDTY[7:0] | | | | | | | |
| | | 15:8 | CDTY[15:8] | | | | | | | |
| | | 23:16 | CDTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0248 | PWM_CDTYUPD2 | 7:0 | CDTYUPD[7:0] | | | | | | | |
| | | 15:8 | CDTYUPD[15:8] | | | | | | | |
| | | 23:16 | CDTYUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x024C | PWM_CPRD2 | 7:0 | CPRD[7:0] | | | | | | | |
| | | 15:8 | CPRD[15:8] | | | | | | | |
| | | 23:16 | CPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0250 | PWM_CPRDUPD2 | 7:0 | CPRDUPD[7:0] | | | | | | | |
| | | 15:8 | CPRDUPD[15:8] | | | | | | | |
| | | 23:16 | CPRDUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0254 | PWM_CCNT2 | 7:0 | CNT[7:0] | | | | | | | |
| | | 15:8 | CNT[15:8] | | | | | | | |
| | | 23:16 | CNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0258 | PWM_DT2 | 7:0 | DTH[7:0] | | | | | | | |
| | | 15:8 | DTH[15:8] | | | | | | | |
| | | 23:16 | DTL[7:0] | | | | | | | |
| | | 31:24 | DTL[15:8] | | | | | | | |
| 0x025C | PWM_DTUPD2 | 7:0 | DTHUPD[7:0] | | | | | | | |
| | | 15:8 | DTHUPD[15:8] | | | | | | | |
| | | 23:16 | DTLUPD[7:0] | | | | | | | |
| | | 31:24 | DTLUPD[15:8] | | | | | | | |
| 0x0260 | PWM_CMR3 | 7:0 | | | | | CPRE[3:0] | | | |
| | | 15:8 | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| | | 23:16 | | | | | PPM | DTLI | DTHI | DTE |
| | | 31:24 | | | | | | | | |
| 0x0264 | PWM_CDTY3 | 7:0 | CDTY[7:0] | | | | | | | |
| | | 15:8 | CDTY[15:8] | | | | | | | |
| | | 23:16 | CDTY[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0268 | PWM_CDTYUPD3 | 7:0 | CDTYUPD[7:0] | | | | | | | |
| | | 15:8 | CDTYUPD[15:8] | | | | | | | |
| | | 23:16 | CDTYUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x026C | PWM_CPRD3 | 7:0 | CPRD[7:0] | | | | | | | |
| | | 15:8 | CPRD[15:8] | | | | | | | |
| | | 23:16 | CPRD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0270 | PWM_CPRDUPD3 | 7:0 | CPRDUPD[7:0] | | | | | | | |
| | | 15:8 | CPRDUPD[15:8] | | | | | | | |
| | | 23:16 | CPRDUPD[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0274 | PWM_CCNT3 | 7:0 | CNT[7:0] | | | | | | | |
| | | 15:8 | CNT[15:8] | | | | | | | |
| | | 23:16 | CNT[23:16] | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0278 | PWM_DT3 | 7:0 | DTH[7:0] | | | | | | | |
| | | 15:8 | DTH[15:8] | | | | | | | |
| | | 23:16 | DTL[7:0] | | | | | | | |
| | | 31:24 | DTL[15:8] | | | | | | | |
| 0x027C | PWM_DTUPD3 | 7:0 | DTHUPD[7:0] | | | | | | | |
| | | 15:8 | DTHUPD[15:8] | | | | | | | |
| | | 23:16 | DTLUPD[7:0] | | | | | | | |
| | | 31:24 | DTLUPD[15:8] | | | | | | | |
| 0x0280 ... 0x03FF | Reserved | | | | | | | | | |
| 0x0400 | PWM_CMUPD0 | 7:0 | | | | | | | | |
| | | 15:8 | | | CPOLINVUP | | | | CPOLUP | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0404 ... 0x041F | Reserved | | | | | | | | | |
| 0x0420 | PWM_CMUPD1 | 7:0 | | | | | | | | |
| | | 15:8 | | | CPOLINVUP | | | | CPOLUP | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0424 ... 0x042B | Reserved | | | | | | | | | |
| 0x042C | PWM_ETRG1 | 7:0 | MAXCNT[7:0] | | | | | | | |
| | | 15:8 | MAXCNT[15:8] | | | | | | | |
| | | 23:16 | MAXCNT[23:16] | | | | | | | |
| | | 31:24 | RFEN | TRGSRC | TRGFILT | TRGEDGE | | | TRGMODE[1:0] | |
| 0x0430 | PWM_LEBR1 | 7:0 | | LEBDELAY[6:0] | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | PWMHREN | PWMHFEN | PWMLREN | PWMLFEN |
| | | 31:24 | | | | | | | | |
| 0x0434 ... 0x043F | Reserved | | | | | | | | | |
| 0x0440 | PWM_CMUPD2 | 7:0 | | | | | | | | |
| | | 15:8 | | | CPOLINVUP | | | | CPOLUP | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0444 ... 0x044B | Reserved | | | | | | | | | |
| 0x044C | PWM_ETRG2 | 7:0 | MAXCNT[7:0] | | | | | | | |
| | | 15:8 | MAXCNT[15:8] | | | | | | | |
| | | 23:16 | MAXCNT[23:16] | | | | | | | |
| | | 31:24 | RFEN | TRGSRC | TRGFILT | TRGEDGE | | | TRGMODE[1:0] | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x0450 | PWM_LEBR2 | 7:0 | | LEBDELAY[6:0] | | | | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | PWMHREN | PWMHFEN | PWMLREN | PWMLFEN |
| | | 31:24 | | | | | | | | |
| 0x0454 ... 0x045F | Reserved | | | | | | | | | |
| 0x0460 | PWM_CMUPD3 | 7:0 | | | | | | | | |
| | | 15:8 | | | CPOLINVUP | | | | CPOLUP | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 51.7.1 PWM Clock Register

**Name:** PWM_CLK
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS0 and WPHWS0 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PREB[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIVB[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PREA[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIVA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:24 – PREB[3:0]** CLKB Source Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | CLK | Peripheral clock |
| 1 | CLK_DIV2 | Peripheral clock/2 |
| 2 | CLK_DIV4 | Peripheral clock/4 |
| 3 | CLK_DIV8 | Peripheral clock/8 |
| 4 | CLK_DIV16 | Peripheral clock/16 |
| 5 | CLK_DIV32 | Peripheral clock/32 |
| 6 | CLK_DIV64 | Peripheral clock/64 |
| 7 | CLK_DIV128 | Peripheral clock/128 |
| 8 | CLK_DIV256 | Peripheral clock/256 |
| 9 | CLK_DIV512 | Peripheral clock/512 |
| 10 | CLK_DIV1024 | Peripheral clock/1024 |
| Other | – | Reserved |

**Bits 23:16 – DIVB[7:0]** CLKB Divide Factor

| Value | Name | Description |
|---|---|---|
| 0 | CLKB_POFF | CLKB clock is turned off |
| 1 | PREB | CLKB clock is clock selected by PREB |
| 2-255 | PREB_DIV | CLKB clock is clock selected by PREB divided by DIVB factor |

**Bits 11:8 – PREA[3:0]** CLKA Source Clock Selection

| Value | Name | Description |
|---|---|---|
| 0 | CLK | Peripheral clock |
| 1 | CLK_DIV2 | Peripheral clock/2 |
| 2 | CLK_DIV4 | Peripheral clock/4 |

| Value | Name | Description |
|---|---|---|
| 3 | CLK_DIV8 | Peripheral clock/8 |
| 4 | CLK_DIV16 | Peripheral clock/16 |
| 5 | CLK_DIV32 | Peripheral clock/32 |
| 6 | CLK_DIV64 | Peripheral clock/64 |
| 7 | CLK_DIV128 | Peripheral clock/128 |
| 8 | CLK_DIV256 | Peripheral clock/256 |
| 9 | CLK_DIV512 | Peripheral clock/512 |
| 10 | CLK_DIV1024 | Peripheral clock/1024 |
| Other | – | Reserved |

**Bits 7:0 – DIVA[7:0]** CLKA Divide Factor

| Value | Name | Description |
|---|---|---|
| 0 | CLKA_POFF | CLKA clock is turned off |
| 1 | PREA | CLKA clock is clock selected by PREA |
| 2-255 | PREA_DIV | CLKA clock is clock selected by PREA divided by DIVA factor |

### 51.7.2 PWM Enable Register

**Name:** PWM_ENA
**Offset:** 0x04
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 0, 1, 2, 3 – CHIDx** Channel ID

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enable PWM output for channel x. |

### 51.7.3 PWM Disable Register

**Name:** PWM_DIS
**Offset:** 0x08
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 0, 1, 2, 3 – CHIDx** Channel ID

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disable PWM output for channel x. |

### 51.7.4 PWM Status Register

**Name:** PWM_SR
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3 – CHIDx** Channel ID

| Value | Description |
|---|---|
| 0 | PWM output for channel x is disabled. |
| 1 | PWM output for channel x is enabled. |

### 51.7.5 PWM Interrupt Enable Register 1

**Name:** PWM_IER1
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – FCHIDx** Fault Protection Trigger on Channel x Interrupt Enable

**Bits 0, 1, 2, 3 – CHIDx** Counter Event on Channel x Interrupt Enable

### 51.7.6 PWM Interrupt Disable Register 1

**Name:** PWM_IDR1
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – FCHIDx** Fault Protection Trigger on Channel x Interrupt Disable

**Bits 0, 1, 2, 3 – CHIDx** Counter Event on Channel x Interrupt Disable

### 51.7.7 PWM Interrupt Mask Register 1

**Name:** PWM_IMR1
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – FCHIDx** Fault Protection Trigger on Channel x Interrupt Mask

**Bits 0, 1, 2, 3 – CHIDx** Counter Event on Channel x Interrupt Mask

### 51.7.8 PWM Interrupt Status Register 1

**Name:** PWM_ISR1
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

**Note:** Reading PWM_ISR1 automatically clears CHIDx and FCHIDx flags.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FCHID3 | FCHID2 | FCHID1 | FCHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHID3 | CHID2 | CHID1 | CHID0 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – FCHIDx** Fault Protection Trigger on Channel x

| Value | Description |
|---|---|
| 0 | No new trigger of the fault protection since the last read of PWM_ISR1. |
| 1 | At least one trigger of the fault protection since the last read of PWM_ISR1. |

**Bits 0, 1, 2, 3 – CHIDx** Counter Event on Channel x

| Value | Description |
|---|---|
| 0 | No new counter event has occurred since the last read of PWM_ISR1. |
| 1 | At least one counter event has occurred since the last read of PWM_ISR1. |

### 51.7.9 PWM Sync Channels Mode Register

**Name:** PWM_SCM
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | PTRCS[2:0] | | | PTRM | | | UPDM[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | SYNC3 | SYNC2 | SYNC1 | SYNC0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 23:21 – PTRCS[2:0]** DMA Controller Transfer Request Comparison Selection
Selection of the comparison used to set the flag WRDY and the corresponding DMA Controller transfer request.

**Bit 20 – PTRM** DMA Controller Transfer Request Mode

| UPDM | PTRM | WRDY Flag and DMA Controller Transfer Request |
|---|---|---|
| 0 | x | The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are never set to ‘1’. |
| 1 | x | The WRDY flag in PWM Interrupt Status Register 2 is set to ‘1’ as soon as the update period is elapsed, the DMA Controller transfer request is never set to ‘1’. |
| 2 | 0 | The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to ‘1’ as soon as the update period is elapsed. |
| | 1 | The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to ‘1’ as soon as the selected comparison matches. |

**Bits 17:16 – UPDM[1:0]** Synchronous Channels Update Mode

| Value | Name | Description |
|---|---|---|
| 0 | MODE0 | Manual write of double buffer registers and manual update of synchronous channels 1[1]. |
| 1 | MODE1 | Manual write of double buffer registers and automatic update of synchronous channels 2[2]. |
| 2 | MODE2 | The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to ‘1’ as soon as the update period is elapsed. |

**Notes:**

1. The update occurs at the beginning of the next PWM period, when the UPDULOCK bit in PWM Sync Channels Update Control Register is set.
2. The update occurs when the Update Period is elapsed.

**Bits 0, 1, 2, 3 – SYNCx** Synchronous Channel x

| Value | Description |
|---|---|
| 0 | Channel x is not a synchronous channel. |
| 1 | Channel x is a synchronous channel. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.10 PWM DMA Register

**Name:** PWM_DMAR
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DMADUTY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DMADUTY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DMADUTY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – DMADUTY[23:0]** Duty-Cycle Holding Register for DMA Access
Each write access to PWM_DMAR sequentially updates PWM_CDTYUPDx.CDTYUPD with DMADUTY (only for channel configured as synchronous). See “Method 3: Automatic write of duty-cycle values and automatic trigger of the update” .

### 51.7.11 PWM Sync Channels Update Control Register

**Name:** PWM_SCUC
**Offset:** 0x28
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | UPDULOCK |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – UPDULOCK** Synchronous Channels Update Unlock
This bit is automatically reset when the update is done.

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | If the UPDM field is set to ‘0’ in PWM Sync Channels Mode Register, writing the UPDULOCK bit to ‘1’ triggers the update of the period value, the duty-cycle and the dead-time values of synchronous channels at the beginning of the next PWM period. If the field UPDM is set to ‘1’ or ‘2’, writing the UPDULOCK bit to ‘1’ triggers only the update of the period value and of the dead-time values of synchronous channels. |

### 51.7.12 PWM Sync Channels Update Period Register

**Name:** PWM_SCUP
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | UPRCNT[3:0] | | | | UPR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:4 – UPRCNT[3:0]** Update Period Counter
Reports the value of the update period counter.

**Bits 3:0 – UPR[3:0]** Update Period
Defines the time between each update of the synchronous channels if automatic trigger of the update is activated (UPDM = 1 or UPDM = 2 in PWM Sync Channels Mode Register). This time is equal to UPR+1 periods of the synchronous channels.

### 51.7.13 PWM Sync Channels Update Period Update Register

**Name:** PWM_SCUPUPD
**Offset:** 0x30
**Reset:** –
**Property:** Write-only

This register acts as a double buffer for the UPR value. This prevents an unexpected automatic trigger of the update of synchronous channels.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UPRUPD[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 3:0 – UPRUPD[3:0]** Update Period Update
Defines the wanted time between each update of the synchronous channels if automatic trigger of the update is activated (UPDM = 1 or UPDM = 2 in PWM Sync Channels Mode Register). This time is equal to UPR+1 periods of the synchronous channels.

### 51.7.14 PWM Interrupt Enable Register 2

**Name:** PWM_IER2
**Offset:** 0x34
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UNRE | | | WRDY |
| Access | | | | | W | | | W |
| Reset | | | | | – | | | – |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx** Comparison x Update Interrupt Enable

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx** Comparison x Match Interrupt Enable

**Bit 3 – UNRE** Synchronous Channels Update Underrun Error Interrupt Enable

**Bit 0 – WRDY** Write Ready for Synchronous Channels Update Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.15 PWM Interrupt Disable Register 2

**Name:** PWM_IDR2
**Offset:** 0x38
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UNRE | | | WRDY |
| Access | | | | | W | | | W |
| Reset | | | | | – | | | – |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx** Comparison x Update Interrupt Disable

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx** Comparison x Match Interrupt Disable

**Bit 3 – UNRE** Synchronous Channels Update Underrun Error Interrupt Disable

**Bit 0 – WRDY** Write Ready for Synchronous Channels Update Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.16 PWM Interrupt Mask Register 2

**Name:** PWM_IMR2
**Offset:** 0x3C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UNRE | | | WRDY |
| Access | | | | | R | | | R |
| Reset | | | | | 0 | | | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx** Comparison x Update Interrupt Mask

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx** Comparison x Match Interrupt Mask

**Bit 3 – UNRE** Synchronous Channels Update Underrun Error Interrupt Mask

**Bit 0 – WRDY** Write Ready for Synchronous Channels Update Interrupt Mask

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.17 PWM Interrupt Status Register 2

**Name:** PWM_ISR2
**Offset:** 0x40
**Reset:** 0x00000000
**Property:** Read-only

Reading PWM_ISR2 automatically clears flags WRDY, UNRE and CMPSx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CMPU7 | CMPU6 | CMPU5 | CMPU4 | CMPU3 | CMPU2 | CMPU1 | CMPU0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CMPM7 | CMPM6 | CMPM5 | CMPM4 | CMPM3 | CMPM2 | CMPM1 | CMPM0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | UNRE | | | WRDY |
| Access | | | | | R | | | R |
| Reset | | | | | 0 | | | 0 |

**Bits 16, 17, 18, 19, 20, 21, 22, 23 – CMPUx** Comparison x Update

| Value | Description |
|---|---|
| 0 | The comparison x has not been updated since the last read of the PWM_ISR2 register. |
| 1 | The comparison x has been updated at least one time since the last read of the PWM_ISR2 register. |

**Bits 8, 9, 10, 11, 12, 13, 14, 15 – CMPMx** Comparison x Match

| Value | Description |
|---|---|
| 0 | The comparison x has not matched since the last read of the PWM_ISR2 register. |
| 1 | The comparison x has matched at least one time since the last read of the PWM_ISR2 register. |

**Bit 3 – UNRE** Synchronous Channels Update Underrun Error

| Value | Description |
|---|---|
| 0 | No Synchronous Channels Update Underrun has occurred since the last read of the PWM_ISR2 register. |
| 1 | At least one Synchronous Channels Update Underrun has occurred since the last read of the PWM_ISR2 register. |

**Bit 0 – WRDY** Write Ready for Synchronous Channels Update

| Value | Description |
|---|---|
| 0 | New duty-cycle and dead-time values for the synchronous channels cannot be written. |
| 1 | New duty-cycle and dead-time values for the synchronous channels can be written. |

### 51.7.18 PWM Output Override Value Register

**Name:** PWM_OOV
**Offset:** 0x44
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OOVL3 | OOVL2 | OOVL1 | OOVL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OOVH3 | OOVH2 | OOVH1 | OOVH0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – OOVLx** Output Override Value for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | Override value is 0 for PWML output of channel x. |
| 1 | Override value is 1 for PWML output of channel x. |

**Bits 0, 1, 2, 3 – OOVHx** Output Override Value for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | Override value is 0 for PWMH output of channel x. |
| 1 | Override value is 1 for PWMH output of channel x. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.19 PWM Output Selection Register

**Name:** PWM_OS
**Offset:** 0x48
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSL3 | OSL2 | OSL1 | OSL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSH3 | OSH2 | OSH1 | OSH0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – OSLx** Output Selection for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | Dead-time generator output DTOLx selected as PWML output of channel x. |
| 1 | Output override value OOVLx selected as PWML output of channel x. |

**Bits 0, 1, 2, 3 – OSHx** Output Selection for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | Dead-time generator output DTOHx selected as PWMH output of channel x. |
| 1 | Output override value OOVHx selected as PWMH output of channel x. |

### 51.7.20 PWM Output Selection Set Register

**Name:** PWM_OSS
**Offset:** 0x4C
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSSL3 | OSSL2 | OSSL1 | OSSL0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSSH3 | OSSH2 | OSSH1 | OSSH0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – OSSLx** Output Selection Set for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Output override value OOVLx selected as PWML output of channel x. |

**Bits 0, 1, 2, 3 – OSSHx** Output Selection Set for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Output override value OOVHx selected as PWMH output of channel x. |

### 51.7.21 PWM Output Selection Clear Register

**Name:** PWM_OSC
**Offset:** 0x50
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCL3 | OSCL2 | OSCL1 | OSCL0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCH3 | OSCH2 | OSCH1 | OSCH0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – OSCLx** Output Selection Clear for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Dead-time generator output DTOLx selected as PWML output of channel x. |

**Bits 0, 1, 2, 3 – OSCHx** Output Selection Clear for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Dead-time generator output DTOHx selected as PWMH output of channel x. |

### 51.7.22 PWM Output Selection Set Update Register

**Name:** PWM_OSSUPD
**Offset:** 0x54
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSSUPL3 | OSSUPL2 | OSSUPL1 | OSSUPL0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSSUPH3 | OSSUPH2 | OSSUPH1 | OSSUPH0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – OSSUPLx** Output Selection Set for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Output override value OOVLx selected as PWML output of channel x at the beginning of the next channel x PWM period. |

**Bits 0, 1, 2, 3 – OSSUPHx** Output Selection Set for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Output override value OOVHx selected as PWMH output of channel x at the beginning of the next channel x PWM period. |

### 51.7.23 PWM Output Selection Clear Update Register

**Name:** PWM_OSCUPD
**Offset:** 0x58
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCUPL3 | OSCUPL2 | OSCUPL1 | OSCUPL0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OSCUPH3 | OSCUPH2 | OSCUPH1 | OSCUPH0 |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

**Bits 16, 17, 18, 19 – OSCUPLx** Output Selection Clear for PWML output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Dead-time generator output DTOLx selected as PWML output of channel x at the beginning of the next channel x PWM period. |

**Bits 0, 1, 2, 3 – OSCUPHx** Output Selection Clear for PWMH output of the channel x

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Dead-time generator output DTOHx selected as PWMH output of channel x at the beginning of the next channel x PWM period. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.24 PWM Fault Mode Register

**Name:** PWM_FMR
**Offset:** 0x5C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

See Fault Inputs for details on fault generation.

**⚠ CAUTION** To prevent an unexpected activation of the status flag FSy in the PWM Fault Status Register, the bit FMODy can be set to '1' only if the FPOLy bit has been previously configured to its final value.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FFIL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FMOD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FPOL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:16 – FFIL[7:0]** Fault Filtering
For each bit y of FFIL, where y is the fault input number:
0: The fault input y is not filtered.
1: The fault input y is filtered.

**Bits 15:8 – FMOD[7:0]** Fault Activation Mode
For each bit y of FMOD, where y is the fault input number:
0: The fault y is active until the fault condition is removed at the peripheral(1) level.
1: The fault y stays active until the fault condition is removed at the peripheral level(1)AND until it is cleared in the PWM Fault Clear Register.
**Note:**
1. The peripheral generating the fault.

**Bits 7:0 – FPOL[7:0]** Fault Polarity
For each bit y of FPOL, where y is the fault input number:
0: The fault y becomes active when the fault input y is at 0.
1: The fault y becomes active when the fault input y is at 1.

### 51.7.25 PWM Fault Status Register

**Name:** PWM_FSR
**Offset:** 0x60
**Reset:** 0x00000000
**Property:** Read-only

Refer to Fault Inputs for details on fault generation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FS[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FIV[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:8 – FS[7:0]** Fault Status
For each bit y of FS, where y is the fault input number:
0: The fault y is not currently active.
1: The fault y is currently active.

**Bits 7:0 – FIV[7:0]** Fault Input Value
For each bit y of FIV, where y is the fault input number:
0: The current sampled value of the fault input y is 0 (after filtering if enabled).
1: The current sampled value of the fault input y is 1 (after filtering if enabled).

### 51.7.26 PWM Fault Clear Register

**Name:** PWM_FCR
**Offset:** 0x64
**Reset:** –
**Property:** Write-only

See Fault Inputs for details on fault generation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FCLR[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 7:0 – FCLR[7:0]** Fault Clear
For each bit y of FCLR, where y is the fault input number:
0: No effect.
1: If bit y of FMOD field is set to ‘1’ and if the fault input y is not at the level defined by the bit y of FPOL field, the fault y is cleared and becomes inactive (FMOD and FPOL fields belong to PWM Fault Mode Register), else writing this bit to ‘1’ has no effect.

### 51.7.27 PWM Fault Protection Value Register 1

**Name:** PWM_FPV1
**Offset:** 0x68
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

See Fault Inputs for details on fault generation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FPVL3 | FPVL2 | FPVL1 | FPVL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FPVH3 | FPVH2 | FPVH1 | FPVH0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 16, 17, 18, 19 – FPVLx** Fault Protection Value for PWML output on channel x
This bit is taken into account only if the bit FPZLx is set to ‘0’ in PWM Fault Protection Value Register 2.

| Value | Description |
|---|---|
| 0 | PWML output of channel x is forced to ‘0’ when fault occurs. |
| 1 | PWML output of channel x is forced to ‘1’ when fault occurs. |

**Bits 0, 1, 2, 3 – FPVHx** Fault Protection Value for PWMH output on channel x
This bit is taken into account only if the bit FPZHx is set to ‘0’ in PWM Fault Protection Value Register 2.

| Value | Description |
|---|---|
| 0 | PWMH output of channel x is forced to ‘0’ when fault occurs. |
| 1 | PWMH output of channel x is forced to ‘1’ when fault occurs. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 51.7.28 PWM Fault Protection Enable Register

**Name:** PWM_FPE
**Offset:** 0x6C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Only the first 8 bits (number of fault input pins) of the register fields are significant.

Refer to Section 6.4 “Fault Inputs” for details on fault generation.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | FPE3[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | FPE2[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | FPE1[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FPE0[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:7, 8:15, 16:23, 24:31 – FPEx** Fault Protection Enable for channel x
For each bit y of FPEx, where y is the fault input number:
0: Fault y is not used for the fault protection of channel x.
1: Fault y is used for the fault protection of channel x.

**⚠ CAUTION** To prevent an unexpected activation of the fault protection, the bit y of FPEx field can be set to ‘1’ only if the corresponding FPOL field has been previously configured to its final value in PWM Fault Mode Register.

### 51.7.29 PWM Event Line x Mode Register

**Name:** PWM_ELMRx
**Offset:** 0x7C + x*0x04 [x=0..1]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CSEL7 | CSEL6 | CSEL5 | CSEL4 | CSEL3 | CSEL2 | CSEL1 | CSEL0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7 – CSELy** Comparison y Selection

| Value | Description |
|---|---|
| 0 | A pulse is not generated on the event line x when the comparison y matches. |
| 1 | A pulse is generated on the event line x when the comparison y match. |

### 51.7.30 PWM Spread Spectrum Register

**Name:** PWM_SSPR
**Offset:** 0xA0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SPRDM |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SPRD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SPRD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPRD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – SPRDM** Spread Spectrum Counter Mode

| Value | Description |
|---|---|
| 0 | Triangular mode. The spread spectrum counter starts to count from -SPRD when the channel 0 is enabled and counts upwards at each PWM period. When it reaches +SPRD, it restarts to count from -SPRD again. |
| 1 | Random mode. The spread spectrum counter is loaded with a new random value at each PWM period. This random value is uniformly distributed and is between -SPRD and +SPRD. |

**Bits 23:0 – SPRD[23:0]** Spread Spectrum Limit Value
The spread spectrum limit value defines the range for the spread spectrum counter. It is introduced in order to achieve constant varying PWM period for the output waveform.

### 51.7.31 PWM Spread Spectrum Update Register

**Name:** PWM_SSPUP
**Offset:** 0xA4
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the SPRD value. This prevents an unexpected waveform when modifying the spread spectrum limit value.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SPRDUP[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SPRDUP[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | SPRDUP[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – SPRDUP[23:0]** Spread Spectrum Limit Value Update
The spread spectrum limit value defines the range for the spread spectrum counter. It is introduced in order to achieve constant varying period for the output waveform.

### 51.7.32 PWM Stepper Motor Mode Register

**Name:** PWM_SMMR
**Offset:** 0xB0
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DOWN1 | DOWN0 |
| Access | | | | | | | | |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | GCEN1 | GCEN0 |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

**Bits 16, 17 – DOWNx** Down Count

| Value | Description |
|---|---|
| 0 | Up counter. |
| 1 | Down counter. |

**Bits 0, 1 – GCENx** Gray Count Enable

| Value | Description |
|---|---|
| 0 | Disable gray count generation on PWML[2*x], PWMH[2*x], PWML[2*x +1], PWMH[2*x +1] |
| 1 | Enable gray count generation on PWML[2*x], PWMH[2*x], PWML[2*x +1], PWMH[2*x +1]. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.33 PWM Fault Protection Value Register 2

**Name:** PWM_FPV2
**Offset:** 0xC0
**Reset:** 0x000F000F
**Property:** Read/Write

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FPZL3 | FPZL2 | FPZL1 | FPZL0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | FPZH3 | FPZH2 | FPZH1 | FPZH0 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 1 | 1 | 1 | 1 |

**Bits 16, 17, 18, 19 – FPZLx** Fault Protection to Hi-Z for PWML output on channel x

| Value | Description |
|---|---|
| 0 | When fault occurs, PWML output of channel x is forced to value defined by the bit FPVLx in PWM Fault Protection Value Register 1. |
| 1 | When fault occurs, PWML output of channel x is forced to high-impedance state. |

**Bits 0, 1, 2, 3 – FPZHx** Fault Protection to Hi-Z for PWMH output on channel x

| Value | Description |
|---|---|
| 0 | When fault occurs, PWMH output of channel x is forced to value defined by the bit FPVHx in PWM Fault Protection Value Register 1. |
| 1 | When fault occurs, PWMH output of channel x is forced to high-impedance state. |

### 51.7.34 PWM Write Protection Control Register

**Name:** PWM_WPCR
**Offset:** 0xE4
**Reset:** –
**Property:** Write-only

See Register Write Protection for the list of registers that can be write-protected.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WPRG5 | WPRG4 | WPRG3 | WPRG2 | WPRG1 | WPRG0 | WPCMD[1:0] | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | – | 0 | – |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x50574D | PASSWD | Writing any other value in this field aborts the write operation of the WPCMD field.<br>Always reads as 0 |

**Bits 2, 3, 4, 5, 6, 7 – WPRGx** Write Protection Register Group x

| Value | Description |
|---|---|
| 0 | The WPCMD command has no effect on the register group x. |
| 1 | The WPCMD command is applied to the register group x. |

**Bits 1:0 – WPCMD[1:0]** Write Protection Command
This command is performed only if the WPKEY corresponds to 0x50574D (“PWM” in ASCII).

| Value | Name | Description |
|---|---|---|
| 0 | DISABLE_SW_PROT | Disables the software write protection of the register groups of which the bit WPRGx is at ‘1’. |
| 1 | ENABLE_SW_PROT | Enables the software write protection of the register groups of which the bit WPRGx is at ‘1’. |
| 2 | ENABLE_HW_PROT | Enables the hardware write protection of the register groups of which the bit WPRGx is at ‘1’. Only a hardware reset of the PWM controller can disable the hardware write protection. Moreover, to meet security requirements, the PIO lines associated with the PWM can not be configured through the PIO interface. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.35 PWM Write Protection Status Register

**Name:** PWM_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | WPHWS5 | WPHWS4 | WPHWS3 | WPHWS2 | WPHWS1 | WPHWS0 |
| Access | | | R | R | R | R | R | R |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WPVS | | WPSWS5 | WPSWS4 | WPSWS3 | WPSWS2 | WPSWS1 | WPSWS0 |
| Access | R | | R | R | R | R | R | R |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – WPVSRC[15:0]** Write Protect Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bits 8, 9, 10, 11, 12, 13 – WPHWSx** Write Protect HW Status

| Value | Description |
|---|---|
| 0 | The HW write protection x of the register group x is disabled. |
| 1 | The HW write protection x of the register group x is enabled. |

**Bit 7 – WPVS** Write Protect Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of PWM_WPSR. |
| 1 | At least one write protection violation has occurred since the last read of PWM_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

**Bits 0, 1, 2, 3, 4, 5 – WPSWSx** Write Protect SW Status

| Value | Description |
|---|---|
| 0 | The SW write protection x of the register group x is disabled. |
| 1 | The SW write protection x of the register group x is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.36 PWM Comparison x Value Register

**Name:** PWM_CMPVx
**Offset:** 0x0130 + x*0x10 [x=0..7]
**Reset:** 0x00000000
**Property:** Read/Write

Only the first 16 bits (channel counter size) of field CV are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CVM |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CV[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CV[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CV[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – CVM** Comparison x Value Mode

| Value | Description |
|---|---|
| 0 | The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is incrementing. |
| 1 | The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is decrementing.<br>Note: This bit is not relevant if the counter of the channel 0 is left-aligned (CALG = 0 in PWM Channel Mode Register) |

**Bits 23:0 – CV[23:0]** Comparison x Value
Define the comparison x value to be compared with the counter of the channel 0.

### 51.7.37 PWM Comparison x Value Update Register

**Name:** PWM_CMPVUPDx
**Offset:** 0x0134 + x*0x10 [x=0..7]
**Reset:** –
**Property:** Write-only

This register acts as a double buffer for the CV and CVM values. This prevents an unexpected comparison x match.

Only the first 16 bits (channel counter size) of field CVUPD are significant.

**⚠ CAUTION** The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CVMUPD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CVUPD[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CVUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CVUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bit 24 – CVMUPD** Comparison x Value Mode Update
**Note:** This bit is not relevant if the counter of the channel 0 is left-aligned (CALG = 0 in PWM Channel Mode Register)

| Value | Description |
|---|---|
| 0 | The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is incrementing. |
| 1 | The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is decrementing. |

**Bits 23:0 – CVUPD[23:0]** Comparison x Value Update
Define the comparison x value to be compared with the counter of the channel 0.

### 51.7.38 PWM Comparison x Mode Register

**Name:** PWM_CMPMx
**Offset:** 0x0138 + x*0x10 [x=0..7]
**Reset:** 0x00000000
**Property:** R/W

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CUPRCNT[3:0] | | | | CUPR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CPRCNT[3:0] | | | | CPR[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CTR[3:0] | | | | | | | CEN |
| Access | R/W | R/W | R/W | R/W | | | | R/W |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

**Bits 23:20 – CUPRCNT[3:0]** Comparison x Update Period Counter
Reports the value of the comparison x update period counter.
Note: The field CUPRCNT is read-only

**Bits 19:16 – CUPR[3:0]** Comparison x Update Period
Defines the time between each update of the comparison x mode and the comparison x value. This time is equal to CUPR+1 periods of the channel 0 counter.

**Bits 15:12 – CPRCNT[3:0]** Comparison x Period Counter
Reports the value of the comparison x period counter.
Note: The field CPRCNT is read-only

**Bits 11:8 – CPR[3:0]** Comparison x Period
CPR defines the maximum value of the comparison x period counter (CPRCNT). The comparison x value is performed periodically once every CPR+1 periods of the channel 0 counter.

**Bits 7:4 – CTR[3:0]** Comparison x Trigger
The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

**Bit 0 – CEN** Comparison x Enable

| Value | Description |
|---|---|
| 0 | The comparison x is disabled and can not match. |
| 1 | The comparison x is enabled and can match. |

### 51.7.39 PWM Comparison x Mode Update Register

**Name:** PWM_CMPMUPDx
**Offset:** 0x013C + x*0x10 [x=0..7]
**Reset:** –
**Property:** W

This register acts as a double buffer for the CEN, CTR, CPR and CUPR values. This prevents an unexpected comparison x match.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CUPRUPD[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPRUPD[3:0] | | | |
| Access | | | | | W | W | W | W |
| Reset | | | | | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CTRUPD[3:0] | | | | | | | CENUPD |
| Access | W | W | W | W | | | | W |
| Reset | 0 | 0 | 0 | – | | | | – |

**Bits 19:16 – CUPRUPD[3:0]** Comparison x Update Period Update
Defines the time between each update of the comparison x mode and the comparison x value. This time is equal to CUPR+1 periods of the channel 0 counter.

**Bits 11:8 – CPRUPD[3:0]** Comparison x Period Update
CPR defines the maximum value of the comparison x period counter (CPRCNT). The comparison x value is performed periodically once every CPR+1 periods of the channel 0 counter.

**Bits 7:4 – CTRUPD[3:0]** Comparison x Trigger Update
The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

**Bit 0 – CENUPD** Comparison x Enable Update

| Value | Description |
|---|---|
| 0 | The comparison x is disabled and can not match. |
| 1 | The comparison x is enabled and can match. |

### 51.7.40 PWM Channel Mode Register

**Name:** PWM_CMRx
**Offset:** 0x0200 + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PPM | DTLI | DTHI | DTE |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | TCTS | DPOLI | UPDS | CES | CPOL | CALG |
| Access | | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CPRE[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bit 19 – PPM** Push-Pull Mode
The Push-Pull mode is enabled for channel x.

| Value | Description |
|---|---|
| 0 | The Push-Pull mode is disabled for channel x. |
| 1 | The Push-Pull mode is enabled for channel x. |

**Bit 18 – DTLI** Dead-Time PWMLx Output Inverted

| Value | Description |
|---|---|
| 0 | The dead-time PWMLx output is not inverted. |
| 1 | The dead-time PWMLx output is inverted. |

**Bit 17 – DTHI** Dead-Time PWMHx Output Inverted

| Value | Description |
|---|---|
| 0 | The dead-time PWMHx output is not inverted. |
| 1 | The dead-time PWMHx output is inverted. |

**Bit 16 – DTE** Dead-Time Generator Enable

| Value | Description |
|---|---|
| 0 | The dead-time generator is disabled. |
| 1 | The dead-time generator is enabled. |

**Bit 13 – TCTS** Timer Counter Trigger Selection

| Value | Description |
|---|---|
| 0 | The comparator of the channel x (OCx) is used as the trigger source for the Timer Counter (TC). |
| 1 | The counter events of the channel x is used as the trigger source for the Timer Counter (TC). |

**Bit 12 – DPOLI** Disabled Polarity Inverted

| Value | Description |
|---|---|
| 0 | When the PWM channel x is disabled (CHIDx(PWM_SR) = 0), the OCx output waveform is the same as the one defined by the CPOL bit. |
| 1 | When the PWM channel x is disabled (CHIDx(PWM_SR) = 0), the OCx output waveform is inverted compared to the one defined by the CPOL bit. |

**Bit 11 – UPDS** Update Selection
If the PWM period is center-aligned (CALG=1):
0: The update occurs at the next end of the PWM period after writing the update register(s).
1: The update occurs at the next end of the PWM half period after writing the update register(s).
If the PWM period is left-aligned (CALG=0), the update always occurs at the end of the PWM period after writing the update register(s).

**Bit 10 – CES** Counter Event Selection
The bit CES defines when the channel counter event occurs when the period is center-aligned (flag CHIDx in PWM Interrupt Status Register 1).
If the PWM period is center-aligned (CALG=1):
0: The channel counter event occurs at the end of the PWM period.
1: The channel counter event occurs at the end of the PWM period and at half the PWM period.
If the PWM period is left-aligned (CALG=0), the channel counter event occurs at the end of the period and the CES bit has no effect.

**Bit 9 – CPOL** Channel Polarity

| Value | Description |
|---|---|
| 0 | The OCx output waveform (output from the comparator) starts at a low level. |
| 1 | The OCx output waveform (output from the comparator) starts at a high level. |

**Bit 8 – CALG** Channel Alignment

| Value | Description |
|---|---|
| 0 | The period is left-aligned. |
| 1 | The period is center-aligned. |

**Bits 3:0 – CPRE[3:0]** Channel Prescaler

| Value | Name | Description |
|---|---|---|
| | MCK | Peripheral clock |
| 1 | MCK_DIV_2 | Peripheral clock/2 |
| 2 | MCK_DIV_4 | Peripheral clock/4 |
| 3 | MCK_DIV_8 | Peripheral clock/8 |
| 4 | MCK_DIV_16 | Peripheral clock/16 |
| 5 | MCK_DIV_32 | Peripheral clock/32 |
| 6 | MCK_DIV_64 | Peripheral clock/64 |
| 7 | MCK_DIV_128 | Peripheral clock/128 |
| 8 | MCK_DIV_256 | Peripheral clock/256 |
| 9 | MCK_DIV_512 | Peripheral clock/512 |
| 10 | MCK_DIV_1024 | Peripheral clock/1024 |
| 11 | CLKA | Clock A |
| 12 | CLKB | Clock B |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.41 PWM Channel Duty Cycle Register

**Name:** PWM_CDTYx
**Offset:** 0x0204 + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CDTY[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CDTY[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CDTY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – CDTY[23:0]** Channel Duty-Cycle
Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

### 51.7.42 PWM Channel Duty Cycle Update Register

**Name:** PWM_CDTYUPDx
**Offset:** 0x0208 + x*0x20 [x=0..3]
**Reset:** –
**Property:** Write-only

This register acts as a double buffer for the CDTY value. This prevents an unexpected waveform when modifying the waveform duty-cycle.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CDTYUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – CDTYUPD[23:0]** Channel Duty-Cycle Update
Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

### 51.7.43 PWM Channel Period Register

**Name:** PWM_CPRDx
**Offset:** 0x020C + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CPRD[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CPRD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPRD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – CPRD[23:0]** Channel Period
If the waveform is left-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:
– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(X \times \mathrm{CPRD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

If the waveform is center-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:
– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(2 \times X \times \mathrm{CPRD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(2 \times X \times \mathrm{CPRD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

### 51.7.44 PWM Channel Period Update Register

**Name:** PWM_CPRDUPDx
**Offset:** 0x0210 + x*0x20 [x=0..3]
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the CPRD value. This prevents an unexpected waveform when modifying the waveform period.

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CPRDUPD[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CPRDUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CPRDUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 23:0 – CPRDUPD[23:0]** Channel Period Update
If the waveform is left-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:
– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(X \times \mathrm{CPRDUPD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(X \times \mathrm{CPRDUPD} \times \mathrm{DIVA})}{f_{\mathrm{peripheral\ clock}}} \text{ or } \frac{(X \times \mathrm{CPRDUPD} \times \mathrm{DIVB})}{f_{\mathrm{peripheral\ clock}}}$$

If the waveform is center-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:
– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = $2^{PREA}$ is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(2 \times X \times \mathrm{CPRDUPD})}{f_{\mathrm{peripheral\ clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(2 \times X \times \text{CPRDUPD} \times \text{DIVA})}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times \text{CPRDUPD} \times \text{DIVB})}{f_{\text{peripheral clock}}}$$

### 51.7.45 PWM Channel Counter Register

**Name:** PWM_CCNTx
**Offset:** 0x0214 + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

Only the first 16 bits (channel counter size) are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CNT[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CNT[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CNT[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 23:0 – CNT[23:0]** Channel Counter Register
Channel counter value. This register is reset when:
• the channel is enabled (writing CHIDx in the PWM_ENA register).
• the channel counter reaches CPRD value defined in the PWM_CPRDx register if the waveform is left-aligned.

### 51.7.46 PWM Channel Dead Time Register

**Name:** PWM_DTx
**Offset:** 0x0218 + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if bits WPSWS4 and WPHWS4 are cleared in the PWM Write Protection Status Register.

Only the first 12 bits (dead-time counter size) of fields DTH and DTL are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DTL[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DTL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DTH[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DTH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – DTL[15:0]** Dead-Time Value for PWMLx Output
Defines the dead-time value for PWMLx output. This value must be defined between 0 and CDTY (PWM_CDTYx).

**Bits 15:0 – DTH[15:0]** Dead-Time Value for PWMHx Output
Defines the dead-time value for PWMHx output. This value must be defined between 0 and the value (CPRD – CDTY) (PWM_CPRDx and PWM_CDTYx).

### 51.7.47 PWM Channel Dead Time Update Register

**Name:** PWM_DTUPDx
**Offset:** 0x021C + x*0x20 [x=0..3]
**Reset:** 0x00000000
**Property:** Write-only

This register can only be written if bits WPSWS4 and WPHWS4 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the DTH and DTL values. This prevents an unexpected waveform when modifying the dead-time values.

Only the first 12 bits (dead-time counter size) of fields DTHUPD and DTLUPD are significant.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DTLUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DTLUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DTHUPD[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DTHUPD[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:16 – DTLUPD[15:0]** Dead-Time Value Update for PWMLx Output
Defines the dead-time value for PWMLx output. This value must be defined between 0 and CDTY (PWM_CDTYx). This value is applied only at the beginning of the next channel x PWM period.

**Bits 15:0 – DTHUPD[15:0]** Dead-Time Value Update for PWMHx Output
Defines the dead-time value for PWMHx output. This value must be defined between 0 and the value (CPRD – CDTY) (PWM_CPRDx and PWM_CDTYx). This value is applied only at the beginning of the next channel x PWM period.

### 51.7.48 PWM Channel Mode Update Register

**Name:** PWM_CMUPDx
**Offset:** 0x0400 + x*0x20 [x=0..3]
**Reset:** –
**Property:** Write-only

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the CPOL value. This prevents an unexpected waveform when modifying the polarity value.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CPOLINVUP | | | | CPOLUP | |
| Access | | | W | | | | W | |
| Reset | | | – | | | | – | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

**Bit 13 – CPOLINVUP** Channel Polarity Inversion Update
If this bit is written at ‘1’, the write of the bit CPOLUP is not taken into account.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The OCx output waveform (output from the comparator) is inverted. |

**Bit 9 – CPOLUP** Channel Polarity Update
The write of this bit is taken into account only if the bit CPOLINVUP is written at ‘0’ at the same time.

| Value | Description |
|---|---|
| 0 | The OCx output waveform (output from the comparator) starts at a low level. |
| 1 | The OCx output waveform (output from the comparator) starts at a high level. |

### 51.7.49 PWM External Trigger Register

**Name:** PWM_ETRGx
**Offset:** 0x042C + (x-1)*0x20 [x=1..2]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RFEN | TRGSRC | TRGFILT | TRGEDGE | | | TRGMODE[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | MAXCNT[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | MAXCNT[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | MAXCNT[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 31 – RFEN** Recoverable Fault Enable

| Value | Description |
|---|---|
| 0 | The TRGINx signal does not generate a recoverable fault. |
| 1 | The TRGINx signal generate a recoverable fault in place of the fault x input. |

**Bit 30 – TRGSRC** Trigger Source

| Value | Description |
|---|---|
| 0 | The TRGINx signal is driven by the PWMEXTRGx input. |
| 1 | The TRGINx signal is driven by the Analog Comparator Controller. |

**Bit 29 – TRGFILT** Filtered input

| Value | Description |
|---|---|
| 0 | The external trigger input x is not filtered. |
| 1 | The external trigger input x is filtered. |

**Bit 28 – TRGEDGE** Edge Selection

| Value | Name | Description |
|---|---|---|
| 0 | FALLING_ZERO | TRGMODE = 1: TRGINx event detection on falling edge.<br>TRGMODE = 2, 3: TRGINx active level is 0 |
| 1 | RISING_ONE | TRGMODE = 1: TRGINx event detection on rising edge.<br>TRGMODE = 2, 3: TRGINx active level is 1 |

**Bits 25:24 – TRGMODE[1:0]** External Trigger Mode

| Value | Name | Description |
|---|---|---|
| 0 | OFF | External trigger is not enabled. |
| 1 | MODE1 | External PWM Reset Mode |
| 2 | MODE2 | External PWM Start Mode |
| 3 | MODE3 | Cycle-by-cycle Duty Mode |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Bits 23:0 – MAXCNT[23:0]** Maximum Counter value
Maximum channel x counter value measured at the TRGINx event since the last read of the register.
At the TRGINx event, if the channel x counter value is greater than the stored MAXCNT value, then MAXCNT is updated by the channel x counter value.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 51.7.50 PWM Leading-Edge Blanking Register

**Name:** PWM_LEBRx
**Offset:** 0x0430 + (x-1)*0x20 [x=1..2]
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PWMHREN | PWMHFEN | PWMLREN | PWMLFEN |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | LEBDELAY[6:0] | | | | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 19 – PWMHREN** PWMH Rising Edge Enable
Leading-edge blanking is enabled on PWMHx output rising edge.

| Value | Description |
|---|---|
| 0 | Leading-edge blanking is disabled on PWMHx output rising edge. |
| 1 | Leading-edge blanking is enabled on PWMHx output rising edge. |

**Bit 18 – PWMHFEN** PWMH Falling Edge Enable

| Value | Description |
|---|---|
| 0 | Leading-edge blanking is disabled on PWMHx output falling edge. |
| 1 | Leading-edge blanking is enabled on PWMHx output falling edge. |

**Bit 17 – PWMLREN** PWML Rising Edge Enable

| Value | Description |
|---|---|
| 0 | Leading-edge blanking is disabled on PWMLx output rising edge. |
| 1 | Leading-edge blanking is enabled on PWMLx output rising edge. |

**Bit 16 – PWMLFEN** PWML Falling Edge Enable

| Value | Description |
|---|---|
| 0 | Leading-edge blanking is disabled on PWMLx output falling edge. |
| 1 | Leading-edge blanking is enabled on PWMLx output falling edge. |

**Bits 6:0 – LEBDELAY[6:0]** Leading-Edge Blanking Delay for TRGINx
Leading-edge blanking duration for external trigger x input. The delay is calculated according to the following formula:
LEBDELAY = ($f_{peripheral\ clock}$ × Delay) + 1

# 52. Analog Front-End Controller (AFEC)

## 52.1 Description

The Analog Front-End Controller (AFEC) is based on an Analog Front-End (AFE) cell integrating a 12-bit Analog-to-Digital Converter (ADC), a Programmable Gain Amplifier (PGA), a Digital-to-Analog Converter (DAC) and two 6-to-1 analog multiplexers, making possible the analog-to-digital conversions of 12 analog lines (in single Sample-and-Hold mode) or two simultaneous conversions of 6 analog lines (in dual Sample-and-Hold mode). The conversions extend from 0V to VREFP. The AFEC supports a 12-bit resolution mode which can be extended up to a 16-bit resolution by digital averaging.

Conversion results are reported in a common register for all channels, as well as in a channel-dedicated register.

Software trigger, external trigger on rising edge of the AFE_ADTRG pin or internal triggers from Timer Counter output(s) are configurable.

The comparison circuitry allows automatic detection of values below a threshold, higher than a threshold, in a given range or outside the range. Thresholds and ranges are fully configurable.

The AFEC internal fault output is directly connected to PWM Fault input. This input can be asserted by means of comparison circuitry in order to immediately put the PWM outputs in a safe state (pure combinational path).

The AFEC also integrates a Sleep mode and a conversion sequencer and connects with a DMA channel. These features reduce both power consumption and processor intervention.

The AFEC has a selectable single-ended or fully differential input and benefits from a 2-bit programmable gain. A set of reference voltages is generated internally from a single external reference voltage node that may be equal to the analog supply voltage. An external decoupling capacitance is required for noise filtering.

A digital error correction circuit based on the multi-bit redundant signed digit (RSD) algorithm is employed in order to reduce INL and DNL errors.

Finally, the user can configure AFE timings, such as startup time and tracking time.

## 52.2 Embedded Characteristics

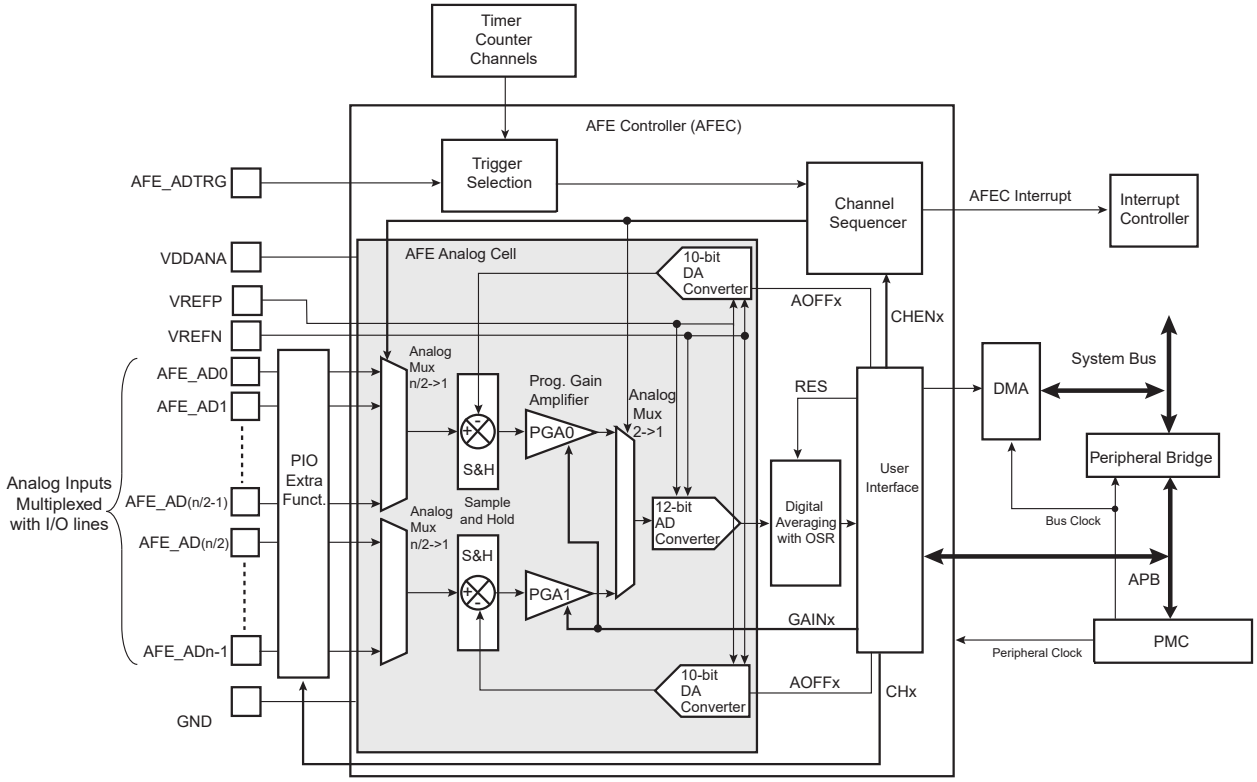
- 12-bit Resolution up to 16-bit Resolution by Digital Averaging
- Wide Range of Power Supply Operation
- Selectable Single-ended or Differential Input Voltage
- Selectable Single or Dual Sample-and-Hold Mode
- Programmable Gain for Maximum Full-Scale Input Range 0–$V_{DD}$
- Programmable Offset Per Channel
- Automatic Correction of Offset and Gain Errors
- Integrated Multiplexers Offering Up to 12 Independent Analog Inputs
- Individual Enable and Disable of Each Channel
- Hardware or Software Trigger
  - External trigger pin
  - Timer counter outputs (corresponding TIOA trigger)
  - PWM event line
- Drive of PWM Fault Input
- DMA Support
- Possibility of AFE Timings Configuration
- Two Sleep Modes and Conversion Sequencer
  - Automatic wakeup on trigger and back to sleep mode after conversions of all enabled channels
  - Possibility of customized channel sequence

© 2021 Microchip Technology Inc.
and its subsidiaries

- Standby Mode for Fast Wakeup Time Response
  - Powerdown capability
- Automatic Window Comparison of Converted Values
- Register Write Protection

## 52.3 Block Diagram

**Figure 52-1. Analog Front-End Controller Block Diagram**




## 52.4 Signal Description

**Table 52-1. AFEC Signal Description**

| Pin Name | Description |
|---|---|
| VREFP | Reference voltage |
| VREFN | Reference voltage |
| AFE_AD0—AFE_AD11[1] | Analog input channels |
| AFE_ADTRG | External trigger |

**Note:**

1. AFE_AD11 is not an actual pin but is connected to a temperature sensor.

## 52.5 Product Dependencies

### 52.5.1 I/O Lines

The digital input AFE_ADTRG is multiplexed with digital functions on the I/O line and the selection of AFE_ADTRG is made using the PIO Controller.

The analog inputs AFE_ADx are multiplexed with digital functions on the I/O lines. AFE_ADx inputs are selected as inputs of the AFEC when writing a one in the corresponding CHx bit of AFEC_CHER and the digital functions are not selected.

### 52.5.2 Power Management

The AFEC is not continuously clocked. The programmer must first enable the AFEC peripheral clock in the Power Management Controller (PMC) before using the AFEC. However, if the application does not require AFEC operations, the peripheral clock can be stopped when not needed and restarted when necessary.

When the AFEC is in Sleep mode, the peripheral clock must always be enabled.

### 52.5.3 Interrupt Sources

The AFEC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the AFEC interrupt requires the interrupt controller to be programmed first.

### 52.5.4 Temperature Sensor

The temperature sensor is connected to Channel 11 of the AFEC.

The temperature sensor provides an output voltage $V_T$ that is proportional to the absolute temperature (PTAT).

### 52.5.5 Timer Triggers

Timer Counters may or may not be used as hardware triggers depending on user requirements. Thus, some or all of the timer counters may be unconnected.

### 52.5.6 PWM Event Lines

PWM event lines may or may not be used as hardware triggers, depending on user requirements.

### 52.5.7 Fault Output

The AFEC has the Fault output connected to the FAULT input of PWM. See Fault Output and implementation of the PWM in the product.

### 52.5.8 Conversion Performances

For performance and electrical characteristics of the AFE, refer to the AFE Characteristics in the section “Electrical Characteristics”.

## 52.6 Functional Description

### 52.6.1 Analog Front-End Conversion

The AFE embeds programmable gain amplifiers that must be enabled prior to any conversion. The bits PGA0EN and PGA1EN in the Analog Control register (AFEC_ACR) must be set.

The AFE uses the AFE clock to perform conversions. In order to guarantee a conversion with minimum error, after any start of conversion, the AFEC waits a number of AFE clock cycles (called transfer time) before changing the channel selection again (and so starts a new tracking operation).

AFE conversions are sequenced by two operating times: the tracking time and the conversion time.

- The tracking time represents the time between the channel selection change and the time for the controller to start the AFEC. The AFEC allows a minimum tracking time of 15 AFE clock periods.

- The conversion time represents the time for the AFEC to convert the analog signal.

The AFE clock frequency is selected in the PRESCAL field of the AFEC_MR. The tracking phase starts during the conversion of the previous channel. If the tracking time is longer than the conversion time of the12-bit AD converter ($t_{CONV}$), the tracking phase is extended to the end of the previous conversion.

The AFE clock frequency ranges from $f_{peripheral\ clock}/2$ if PRESCAL is 1, and $f_{peripheral\ clock}/256$ if PRESCAL is set to 255 (0xFF). PRESCAL must be programmed to provide the AFE clock frequency given in the section “Electrical Characteristics”.

The AFE conversion time ($t_{AFE_conv}$) is applicable for all modes and is calculated as follows:

$$t_{\mathrm{AFE_conv}} = 23 \times t_{\mathrm{AFE\ Clock}}$$

When the averager is activated, the AFE conversion time is multiplied by the OSR value.

In Free Run mode, the sampling frequency ($f_S$) is calculated as $1/t_{AFE_conv}$.

**Figure 52-2. Sequence of AFE Conversions when Tracking Time > Conversion Time**

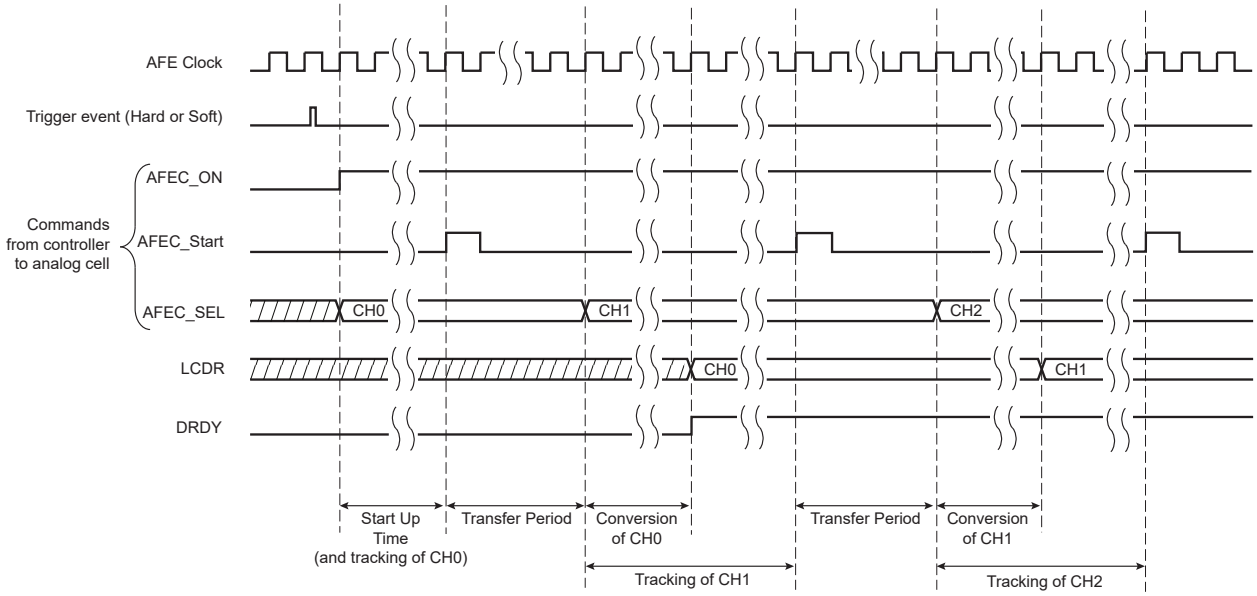
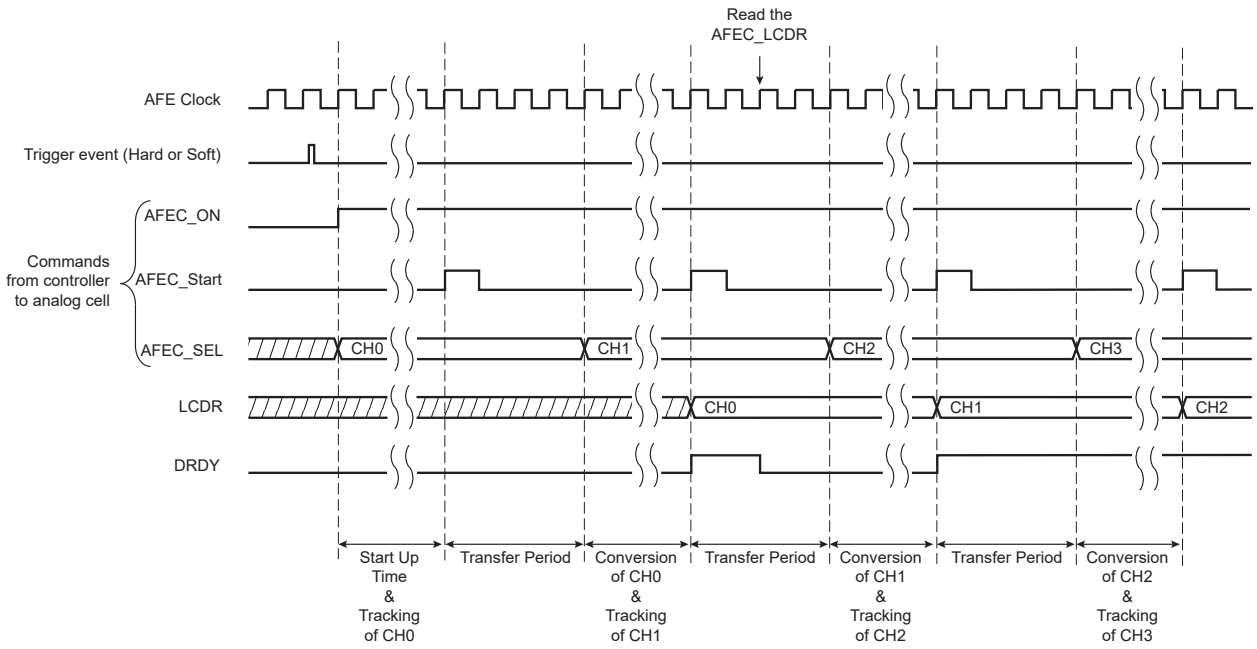



**Figure 52-3. Sequence of AFE Conversions when Tracking Time < Conversion Time**




### 52.6.2 Conversion Reference

The conversion is performed on a full range between 0V and the reference voltage carried on pin VREFP. Analog inputs between these voltages convert to values based on a linear conversion.

### 52.6.3 Conversion Resolution

The AFEC supports 12-bit native resolutions. Writing ‘2’ or greater to the RES field in the Extended Mode register (AFEC_EMR) automatically enables the Enhanced Resolution mode. For details on this mode, see 52.6.14 Enhanced Resolution Mode and Digital Averaging Function.

Moreover, when a DMA channel is connected to the AFEC, a resolution lower than 16 bits sets the transfer request size to 16 bits.

**Note:** If ADTRG is asynchronous to the AFEC peripheral clock, the internal resynchronization introduces a jitter of 1 peripheral clock. This jitter may reduce the resolution of the converted signal. Refer to the formula below, where $f_{IN}$ is the frequency of the analog signal to convert and $t_J$ is the half-period of 1 peripheral clock.

$$\mathrm{SNR} = 20 \times \log 10\left(\frac{1}{2\pi f_{\mathrm{IN}} t_J}\right)$$

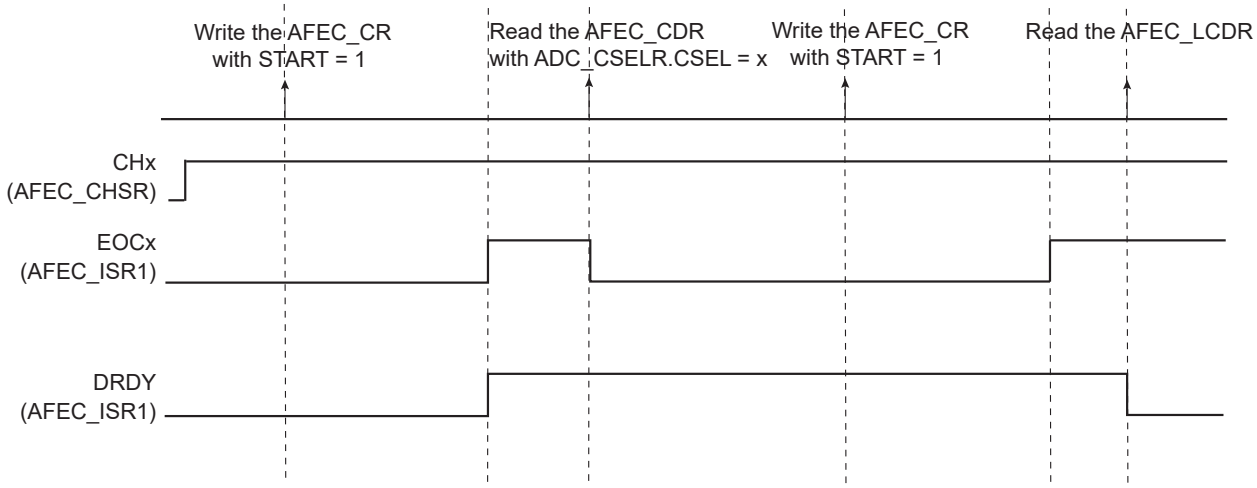
### 52.6.4 Conversion Results

When a conversion is completed, the resulting 12-bit digital value is stored in an internal register (one register for each channel) that can be read by means of the Channel Data Register (AFEC_CDR) and the Last Converted Data Register (AFEC_LCDR). By setting the bit TAG in the AFEC_EMR, the AFEC_LCDR presents the channel number associated with the last converted data in the CHNB field.

The bits EOCx, where 'x' corresponds to the value programmed in the CSEL bit of AFEC_CSELR, and DRDY in the Interrupt Status Register (AFEC_ISR) are set. In the case of a connected DMA channel, DRDY rising triggers a data transfer request. In any case, either EOCx or DRDY can trigger an interrupt.

Reading the AFEC_CDR clears the EOCx bit. Reading AFEC_LCDR clears the DRDY bit and the EOCx bit corresponding to the last converted channel.

**Figure 52-4. EOCx and DRDY Flag Behavior**



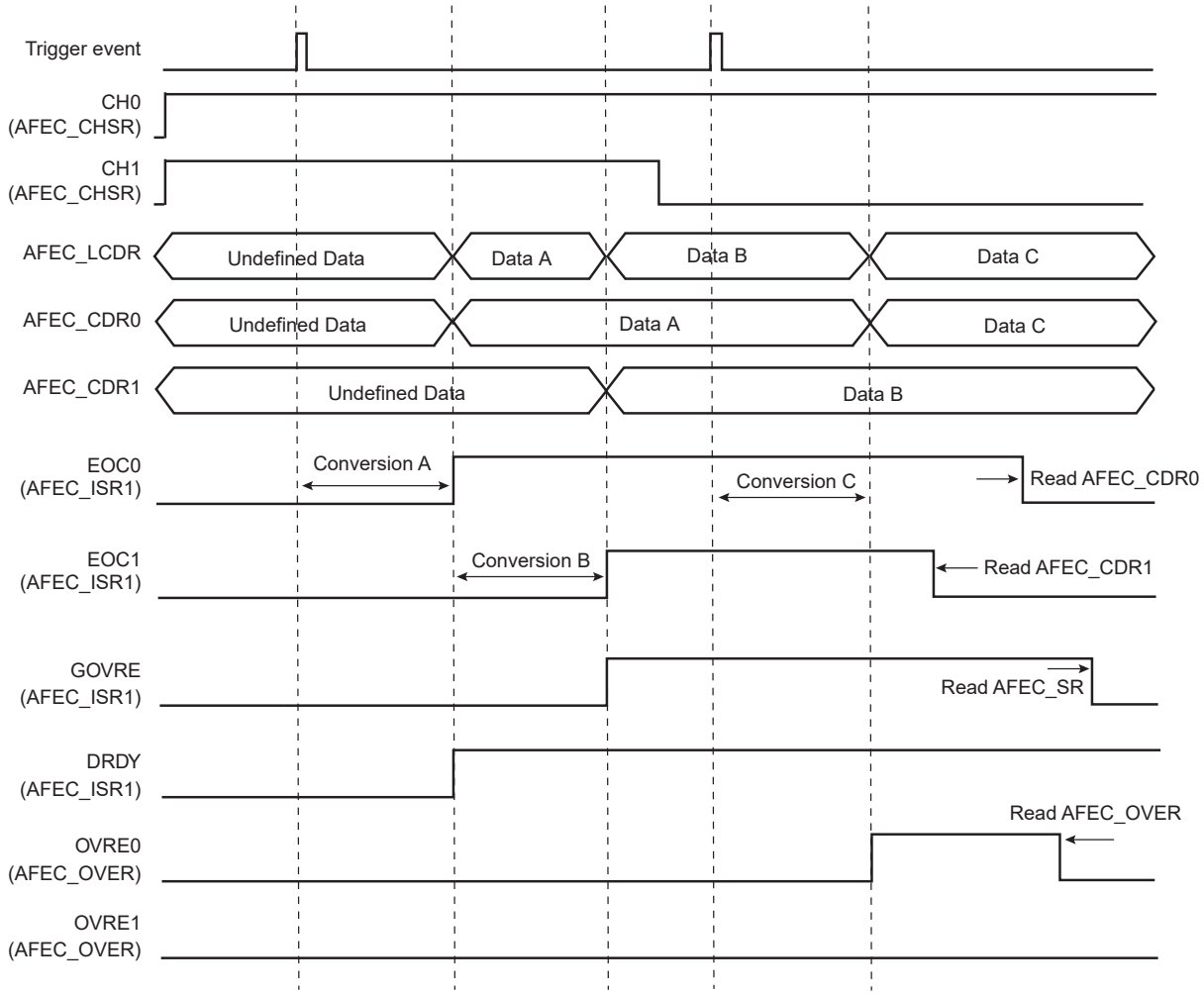

If AFEC_CDR is not read before further incoming data is converted, the corresponding OVREx flag is set in the Overrun Status Register (AFEC_OVER).

New data converted when DRDY is high sets the GOVRE bit in AFEC_ISR.

The OVREx flag is automatically cleared when AFEC_OVER is read, and the GOVRE flag is automatically cleared when AFEC_ISR is read.

© 2021 Microchip Technology Inc. and its subsidiaries

**Figure 52-5. EOCx, GOVRE and OVREx Flag Behavior**




**WARNING** If the corresponding channel is disabled during a conversion, or if it is disabled and then reenabled during a conversion, its associated data and its corresponding EOCx and GOVRE flags in AFEC_ISR and OVREx flags in AFEC_OVER are unpredictable.

### 52.6.5 Conversion Results Format

The conversion results can be signed (2's complement) or unsigned depending on the value of the SIGNMODE field in AFEC_EMR.

Four modes are available:

- Results of channels configured in Single-ended mode are unsigned; results of channels configured in Differential mode are signed.
- Results of channels configured in Single-ended mode are signed; results of channels configured in Differential mode are unsigned.
- Results of all channels are unsigned.
- Results of all channels are signed.

If conversion results are signed and resolution is less than 16 bits, the sign is extended up to the bit 15 (e.g., 0xF43 for 12-bit resolution will be read as 0xFF43 and 0x467 will be read as 0x0467).

### 52.6.6 Conversion Triggers

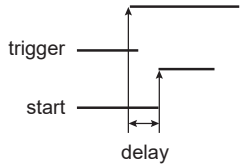
Conversions of the active analog channels are started with a software or hardware trigger. The software trigger is provided by writing a ‘1’ to the bit START in the Control Register (AFEC_CR).

The hardware trigger can be one of the TIOA outputs of the Timer Counter channels, PWM Event line, or the external trigger input of the AFEC (ADTRG). The hardware trigger is selected with AFEC_MR.TRGSEL. The selected hardware trigger is enabled with AFEC_MR.TRGEN

The minimum time between two consecutive trigger events must be strictly greater than the duration of the longest conversion sequence according to configuration of registers AFEC_MR, AFEC_CHSR, AFEC_SEQ1R, AFEC_SEQ2R.

If a hardware trigger is selected, the start of a conversion is triggered after a delay starting at each rising edge of the selected signal. Due to asynchronous handling, the delay may vary in a range of two peripheral clock periods to one AFE clock period. This delay varies from trigger to trigger and so introduces a jitter error leading to a reduced Signal-to-Noise ratio performance.

**Figure 52-6. Conversion Start with the Hardware Trigger**




If one of the TIOA outputs is selected, the corresponding Timer Counter channel must be programmed in Waveform mode.

Only one start command is necessary to initiate a conversion sequence on all the channels. The AFEC hardware logic automatically performs the conversions on the active channels, then waits for a new request. The Channel Enable (AFEC_CHER) and Channel Disable (AFEC_CHDR) registers permit the analog channels to be enabled or disabled independently.

If the AFEC is used with a DMA, only the transfers of converted data from enabled channels are performed and the resulting data buffers should be interpreted accordingly.

### 52.6.7 Sleep Mode and Conversion Sequencer

The AFEC Sleep mode maximizes power saving by automatically deactivating the AFE when it is not being used for conversions. Sleep mode is selected by setting AFEC_MR.SLEEP.

Sleep mode is managed by a conversion sequencer, which automatically processes the conversions of all channels at lowest power consumption.

This mode can be used when the minimum period of time between two successive trigger events is greater than the startup period of the AFEC. Refer to the AFE Characteristics in the section “Electrical Characteristics”.

When a start conversion request occurs, the AFE is automatically activated. As the analog cell requires a startup time, the logic waits during this lapse and starts the conversion on the enabled channels. When all conversions are complete, the AFE is deactivated until the next trigger. Triggers occurring during the sequence are not taken into account.

A fast wakeup mode is available in the AFEC_MR as a compromise between power-saving strategy and responsiveness. Setting the FWUP bit enables the Fast Wakeup mode. In Fast Wakeup mode, the AFE is not fully deactivated while no conversion is requested, thereby providing lower power savings but faster wakeup.

The conversion sequencer allows automatic processing with minimum processor intervention and optimized power consumption. Conversion sequences are performed periodically using a Timer/Counter output or the PWM event line.

The DMA can automatically process the periodic acquisition of several samples without processor intervention.

The sequence can be customized by programming the Channel Sequence registers AFEC_SEQ1R and AFEC_SEQ2R and setting AFEC_MR.USEQ. The user selects a specific order of channels and can program up to 12 conversions by sequence. The user may create a personal sequence by writing channel numbers in AFEC_SEQ1R and AFEC_SEQ2R. Channel numbers can be written in any order and repeated several times. Only enabled USCHx fields are converted. Thus, to program a 15-conversion sequence, the user disables AFEC_CHSR.CH15, thus disabling AFEC_SEQ2R.USCH15.

© 2021 Microchip Technology Inc.
and its subsidiaries

**Note:** The reference voltage pins always remain connected in Normal mode as in Sleep mode.

### 52.6.8 Comparison Window

The AFEC features automatic comparison functions. It compares converted values to a low threshold, a high threshold or both, depending on the value of AFEC_EMR.CMPMODE. The comparison can be done on all channels or only on the channel specified in AFEC_EMR.CMPSEL. To compare all channels, AFEC_EMR.CMPALL must be set.

Moreover, a filtering option can be set by writing the number of consecutive comparison errors needed to raise the flag. This number can be written and read in AFEC_EMR.CMPFILTER.

The flag can be read on AFEC_ISR.COMPE and can trigger an interrupt.

The high threshold and the low threshold can be read/written in the Compare Window Register (AFEC_CWR).

Depending on the sign of the conversion, chosen by setting the SIGNMODE bit in the AFEC Extended Mode Register, the high threshold and low threshold values must be signed or unsigned to maintain consistency during the comparison. If the conversion is signed, both thresholds must also be signed; if the conversion is unsigned, both thresholds must be unsigned. If comparison occurs on all channels, the SIGNMODE bit must be set to ALL_UNSIGNED or ALL_SIGNED and thresholds must be set accordingly.

### 52.6.9 Differential Inputs

The AFE can be used either as a single-ended AFE (AFEC_DIFFR.DIFF = 0) or as a fully differential AFE (AFEC_DIFFR.DIFF = 1). By default, after a reset, the AFE is in Single-ended mode.

The AFEC can apply a different mode on each channel.

The same inputs are used in Single-ended or Differential mode.

Depending on the AFE mode, the analog multiplexer selects one or two inputs to map to a channel. The table below provides input mapping for both modes.

**Table 52-2. Input Pins and Channel Numbers**

| Input Pins | Channel Numbers | |
|---|---|---|
| | Single-ended Mode | Differential Mode |
| AFE_AD0 | CH0 | CH0 |
| AFE_AD1 | CH1 | |
| ... | ... | ... |
| AFE_AD10 | CH10 | CH10 |
| AFE_AD11 | CH11 | |

### 52.6.10 Sample-and-Hold Modes

The AFE can be configured in either single Sample-and-Hold mode (AFEC_SHMR.DUALx = 0) or dual Sample-and-Hold mode (AFEC_SHMR.DUALx = 1). By default, after a reset, the AFE is in single Sample-and-Hold mode.

The AFEC can apply a different mode on each channel.

The same inputs are used in single Sample-and-Hold mode or in dual Sample-and-Hold mode. Single-ended/Differential mode and single/dual Sample-and-Hold mode can be combined. See the following tables.

**Table 52-3. Input Pins and Channel Numbers In Dual Sample-and-Hold Mode**

| Single-Ended Input Pins | Differential Input Pins | Channel Numbers |
|---|---|---|
| AFE_AD0 & AFE_AD6 | AFE_AD0-AD1 & AFE_AD6–AFE_AD7 | CH0 |
| AFE_AD1 & AFE_AD7 | – | CH1 |
| ... | ... | ... |

© 2021 Microchip Technology Inc. and its subsidiaries

| .....continued | | |
|---|---|---|
| **Single-Ended Input Pins** | **Differential Input Pins** | **Channel Numbers** |
| AFE_AD4 & AFE_AD10 | AFE_AD4–AFE_AD5 & AFE_AD10–AFE_AD11 | CH4 |
| AFE_AD5 & AFE_AD11 | – | CH5 |

**Table 52-4. Input Pins and Channel Numbers in Single Sample-and-Hold Mode**

| Single-Ended Input Pins | Differential Input Pins | Channel Numbers |
|---|---|---|
| AFE_AD0 | AFE_AD0-AFE_AD1 | CH0 |
| AFE_AD1 | – | CH1 |
| ... | ... | ... |
| AFE_AD10 | AFE_AD10–AFE_AD11 | CH10 |
| AFE_AD11 | – | CH11 |

### 52.6.11 Input Gain and Offset

The AFE has a built-in programmable gain amplifier (PGA) and programmable offset per channel through a DAC.

The programmable gain amplifier can be set to gains of 1, 2 and 4 and can be used for single-ended applications or for fully differential applications.

The AFEC can apply different gain and offset on each channel.

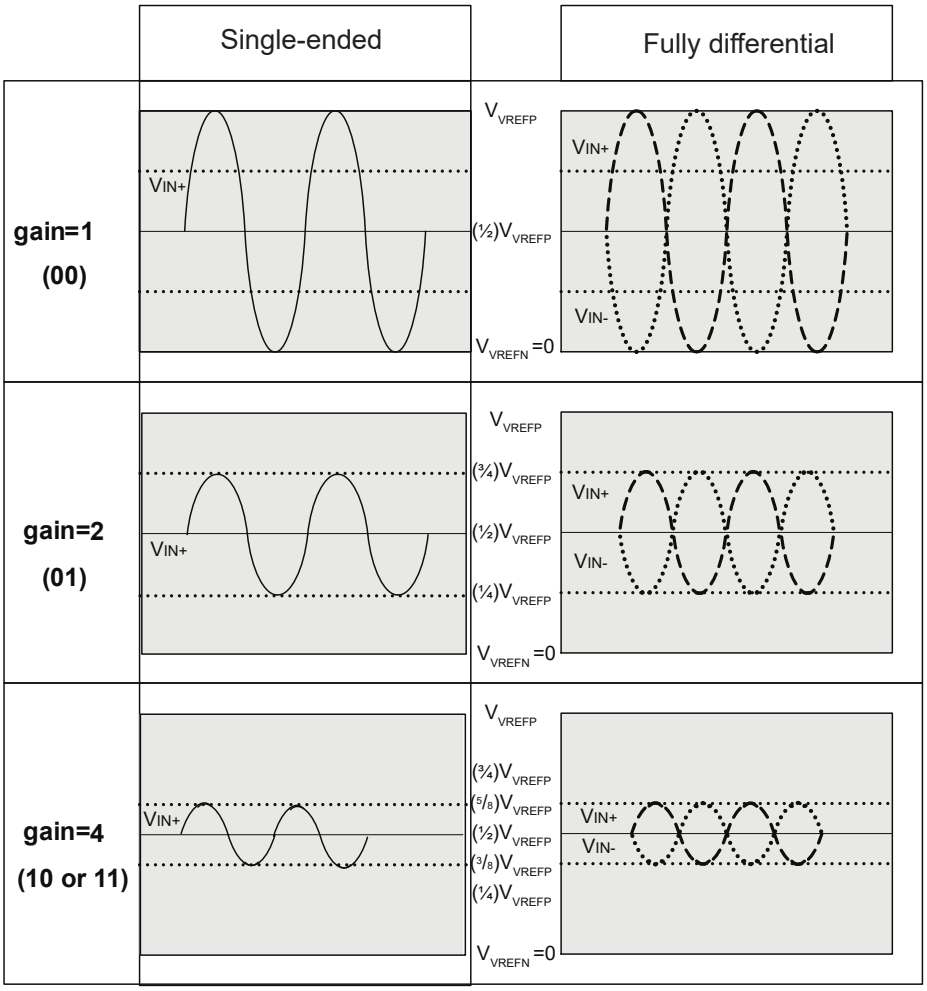
The gain is configured in the GAIN field of the Channel Gain Register (AFEC_CGR) as shown in the following table.

**Table 52-5. Gain of the Sample-and-Hold Unit**

| GAIN | GAIN (DIFFx = 0) | GAIN (DIFFx = 1) |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 2 | 2 |
| 2 | 4 | 4 |
| 3 | 4 | 4 |

The analog offset of the AFE is configured in the AOFF field in the Channel Offset Compensation register (AFEC_COCR). The offset is only available in Single-ended mode. The field AOFF must be configured to 512 (mid scale of the DAC) when there is no offset error to compensate.To compensate for an offset error of n LSB (positive or negative), the field AOFF must be configured to 512 + n.

**Figure 52-7. Analog Full Scale Ranges in Single-Ended/Differential Applications Versus Gain**




### 52.6.12 AFE Timings

Each AFE has its own minimal startup time configured in AFEC_MR.STARTUP.

**WARNING** No input buffer amplifier to isolate the source is included in the AFE. This must be taken into consideration.

### 52.6.13 Temperature Sensor

The temperature sensor is internally connected to channel index 11.

The AFEC manages temperature measurement in several ways. The different methods of measurement depend on the configuration bits TRGEN in the AFEC_MR and CH11 in AFEC_CHSR.

Temperature measurement can be triggered at the same rate as other channels by enabling the conversion channel 11.

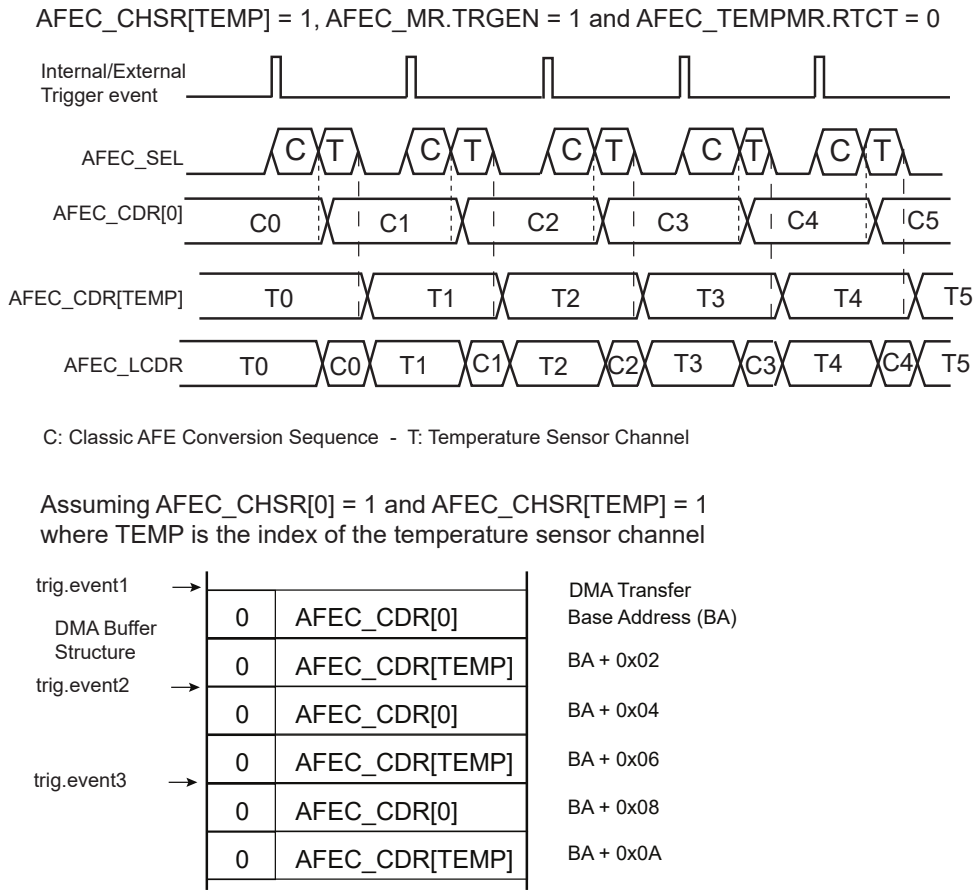
If AFEC_CHSR.CH11 is enabled, the temperature sensor analog cell is switched on. If a user sequence is used, the last converted channel of the sequence is always the temperature sensor channel.

A manual start can be performed only if AFEC_MR.TRGEN is disabled. When AFEC_CR.START is set, the temperature sensor channel conversion is scheduled together with the other enabled channels (if any). The result of the conversion is placed in an internal register that can be read in the AFEC_CDR (AFEC_CSELR must be programmed accordingly prior to reading AFEC_CDR) and the associated flag EOC11 is set in the AFEC_ISR.

© 2021 Microchip Technology Inc. and its subsidiaries

The channel of the temperature sensor is periodically converted together with the other enabled channels and the result is placed into AFEC_LCDR and an internal register (can be read in AFEC_CDR). Thus the temperature conversion result is part of the Peripheral DMA Controller buffer. The temperature channel can be enabled/disabled at any time, but this may not be optimal for downstream processing.

**Figure 52-8. Non-Optimized Temperature Conversion**



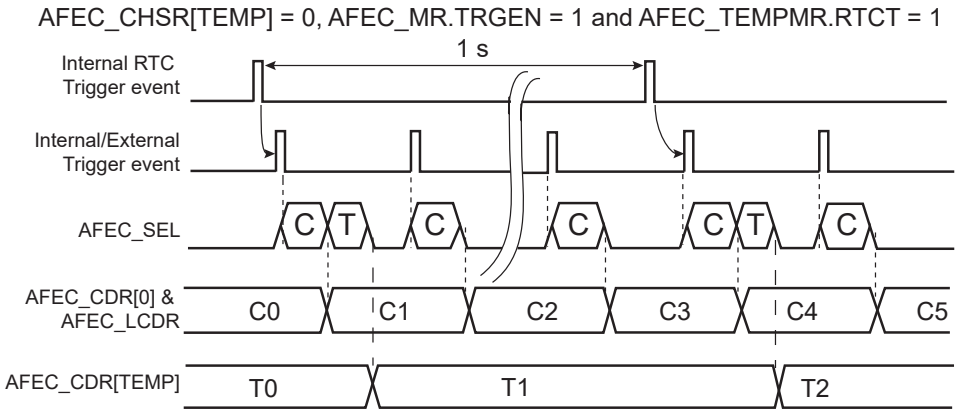

The temperature factor has a slow variation rate and may be different from other conversion channels. As a result, the AFEC allows a different way of triggering temperature measurement when AFEC_TEMPMR.RTCT is set but AFEC_CHSR.CH11 is cleared.

In this configuration, the measurement is triggered every second by means of an internal trigger generated by the RTC. This trigger is always enabled and independent of the triggers used for other channels. In this mode of operation, the temperature sensor is only powered for a period of time covering startup time and conversion time.

Every second, a conversion is scheduled for channel 11 but the result of the conversion is only uploaded to an internal register read by means of AFEC_CDR, and not to AFEC_LCDR. Therefore, the temperature channel is not part of the Peripheral DMA Controller buffer; only the enabled channel are kept in the buffer. The end of conversion of the temperature channel is reported by means of the EOC11 flag in AFEC_ISR.

© 2021 Microchip Technology Inc. and its subsidiaries

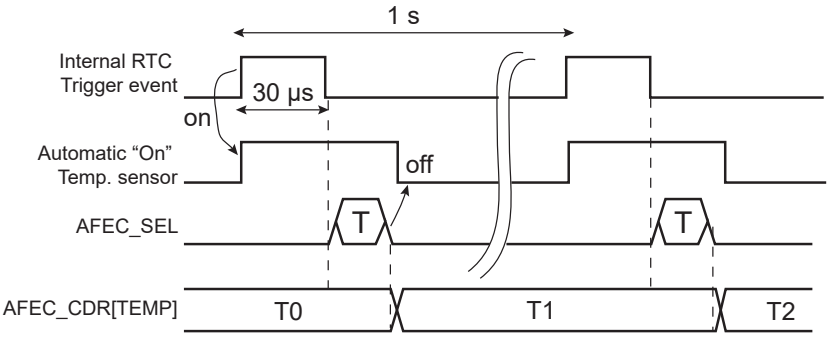
**Figure 52-9. Optimized Temperature Conversion Combined with Classical Conversions**




If RTCT is set and TRGEN is cleared, then all channels are disabled (AFEC_CHSR = 0) and only channel 11 is converted at a rate of one conversion per second.

This mode of operation, when combined with Sleep mode operation, provides a low-power mode for temperature measurement assuming there is no other AFE conversion to schedule at a higher sampling rate or no other channel to convert.

**Figure 52-10. Temperature Conversion Only**




Moreover, it is possible to raise a flag only if there is predefined change in the temperature measurement. The user can define a range of temperature or a threshold in AFEC_TEMPCWR and the mode of comparison in AFEC_TEMPMR. These values define the way the TEMPCHG flag will be raised in AFEC_ISR.

The TEMPCHG flag can be used to trigger an interrupt if there is an update/modification to be made in the system resulting from a temperature change.

In any case, if temperature sensor measurement is configured, the temperature can be read at any time in AFEC_CDR (AFEC_CSELR must be programmed accordingly prior to reading AFEC_CDR) .

### 52.6.14 Enhanced Resolution Mode and Digital Averaging Function

The Enhanced Resolution mode is enabled when AFEC_EMR.RES is set to 13-bit resolution or higher. In this mode, the AFEC trades conversion performance for accuracy by averaging multiple samples, thus providing a digital low-pass filter function. The resolution mode selected determines the oversampling, which represents the performance reduction factor.

To increase the accuracy by averaging multiple samples, some noise must be present in the input signal. The noise level should be between one and two LSB peak-to-peak to get good averaging performance.

The following table summarizes the oversampling ratio depending on the resolution mode selected.

**Table 52-6. Resolution and Oversampling Ratio**

| Resolution Mode | Oversampling Ratio |
|---|---|
| 13-bit | 4 |
| 14-bit | 16 |
| 15-bit | 64 |
| 16-bit | 256 |

Free Run mode is not supported if Enhanced Resolution mode is used.

The selected oversampling ratio applies to all enabled channels except the temperature sensor channel if triggered by an RTC event. See 52.5.4 Temperature Sensor.

The average result is valid into an internal register (read by means of the AFEC_CDR) only if EOCx (x corresponding to the index of the channel) flag is set in AFEC_ISR and OVREx flag is cleared in the AFEC_OVER. The average result is valid for all channels in the AFEC_LCDR only if DRDY is set and GOVRE is cleared in the AFEC_ISR.

Note that the AFEC_CDR is not buffered. Therefore, when an averaging sequence is ongoing, the value in this register changes after each averaging sample. However, overrun flags in the AFEC_OVER rise as soon as the first sample of an averaging sequence is received. Thus the previous averaged value is not read, even if the new averaged value is not ready.

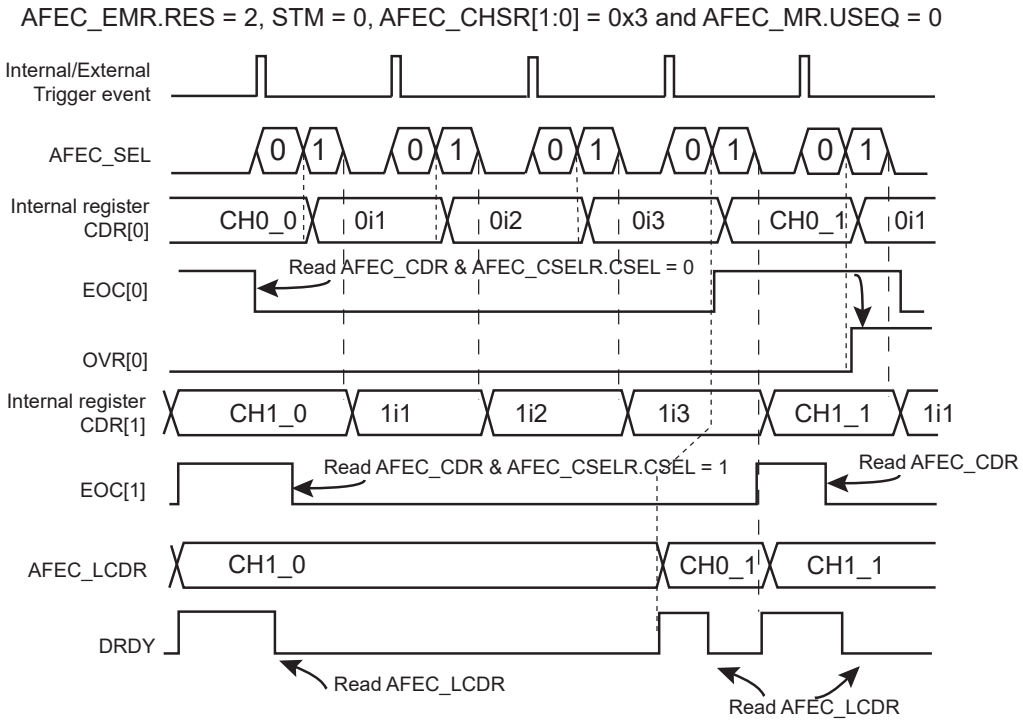
As a result, when an overrun flag rises in the AFEC_OVER, this indicates only that the previous unread data is lost. It does not indicate that this data has been overwritten by the new averaged value, as the averaging sequence concerning this channel can still be on-going.

The samples can be defined in different ways for the averaging function depending on the configuration of AFEC_EMR.STM and AFEC_MR.USEQ.

When USEQ is cleared, there are two possible ways to generate the averaging through the trigger event. If AFEC_EMR.STM is cleared, every trigger event generates one sample for each enabled channel, as described in the figure below. Therefore, four trigger events are requested to get the result of averaging if RES = 2.

© 2021 Microchip Technology Inc.
and its subsidiaries

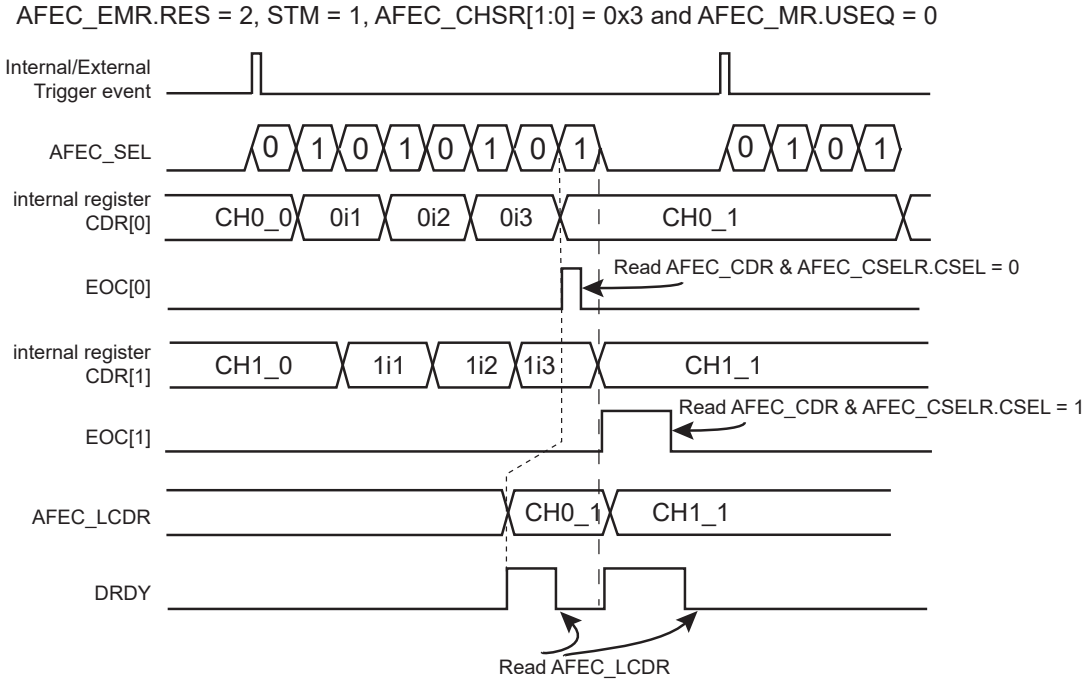
**Figure 52-11. Digital Averaging Function Waveforms over Multiple Trigger Events**




Note: 0i1,0i2,0i3, 1i1, 1i2, 1i3 are intermediate results and CH0/1_0/1 are final result of average function.

If AFEC_EMR.STM is set and AFEC_MR.USEQ is cleared, the sequence to be converted, defined in the AFEC_CHSR, is automatically repeated n times, where n corresponds to the oversampling ratio defined AFEC_EMR.RES. As a result, only one trigger is required to get the result of the averaging function as shown in the figure below.

**Figure 52-12. Digital Averaging Function Waveforms on a Single Trigger Event**




Note: 0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0/1_0/1 are final result of average function.

When USEQ is set, the user can define the channel sequence to be converted by configuring AFEC_SEQxR and AFEC_CHER so that channels are not interleaved during the averaging period. Under these conditions, a sample is defined for each end of conversion as described in the figure below.

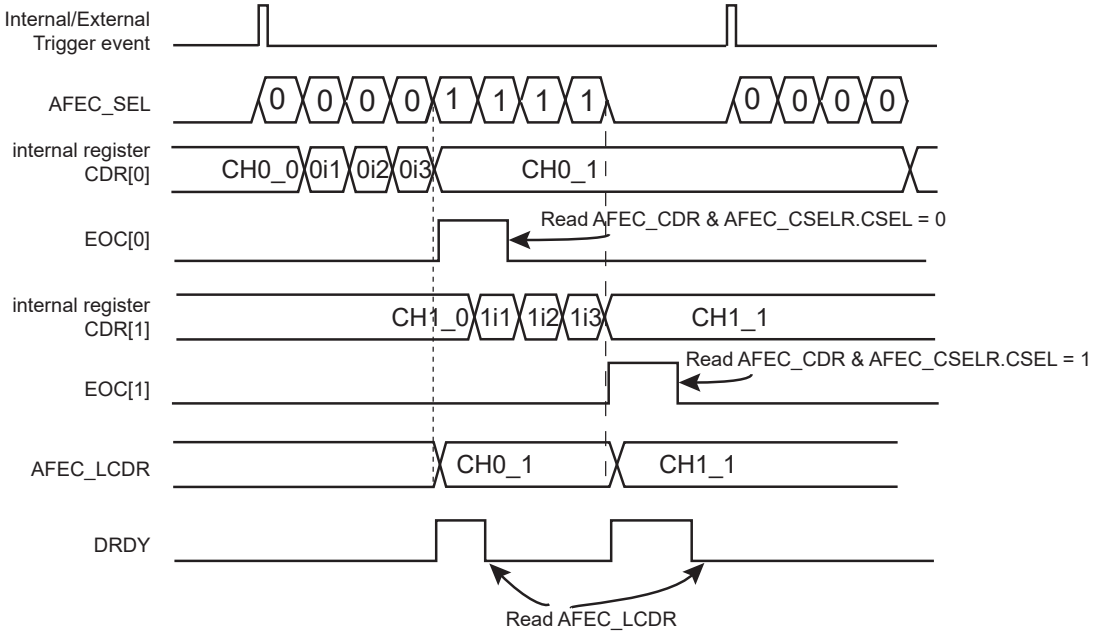
Therefore, if the same channel is configured to be converted four times consecutively and AFEC_EMR.RES = 2, the averaging result is placed in the corresponding channel internal data register (read by means of the AFEC_CDR) and the AFEC_LCDR for each trigger event.

In this case, the AFE real sample rate remains the maximum AFE sample rate divided by 4.

When USEQ is set and the RES field enables the Enhanced Resolution mode, it is important to note that the user sequence must be a sequence being an integer multiple of 4 (i.e., the number of the enabled channel in the Channel Status register (AFEC_CHSR) must be an integer multiple of 4 and the AFEC_SEQxR must be a series of 4 times the same channel index).

**Figure 52-13. Digital Averaging Function Waveforms on a Single Trigger Event, Non-interleaved**

AFEC_EMR.EMR = 2, STM = 1, AFEC_CHSR[7:0] = 0xFF and AFEC_MR.USEQ = 1
AFEC_SEQ1R = 0x1111_0000




Note: 0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0/1_0/1 are final results of average function.

### 52.6.15 Automatic Error Correction

The AFEC features automatic error correction of conversion results. Offset and gain error corrections are available. The correction can be enabled for each channel and correction values (offset and gain) are defined per Sample & Hold unit.

To enable error correction, the ECORR bit must be set in the AFEC Channel Error Correction register (AFEC_CECR). The offset and gain values used to compensate the results are set per Sample & Hold unit basis using the AFEC Correction Select register (AFEC_COSR) and the AFEC Correction Values register (AFEC_CVR). AFEC_COSR is used to select the Sample & Hold unit to be displayed in AFEC_CVR. This selection applies both to read and write operations in AFEC_CVR.

AFEC_CVR.OFFSETCORR and AFEC_CVR.GAINCORR must be filled with the values of corrective data. This data is computed from two measurement points in signed format. The correction is the same for all functional modes.

The final conversion result after error correction is obtained using the following formula, which is implemented after averaging in 2's complement format, with:

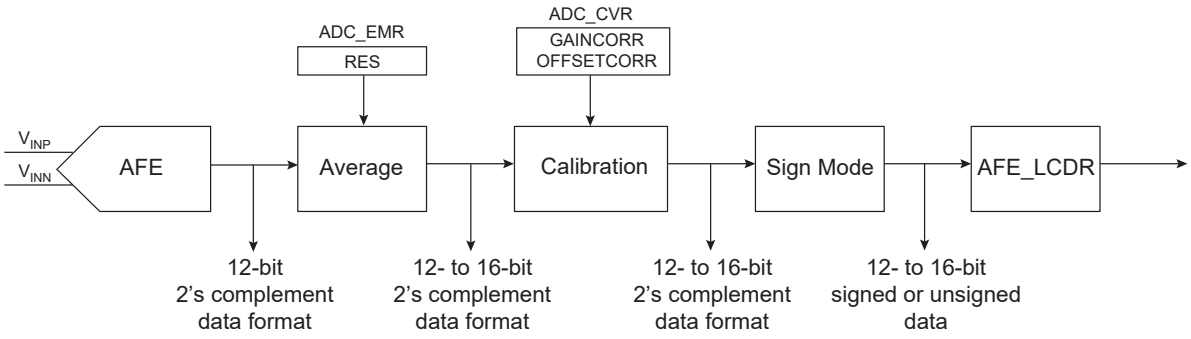
- OFFSETCORR—the offset correction value. OFFSETCORR is a signed value.
- GAINCORR—the gain correction value

© 2021 Microchip Technology Inc. and its subsidiaries

- Gs—the value 15

$$\text{Corrected Data} = (\text{Converted Data} + \text{OFFSETCORR}) \times \frac{\text{GAINCORR}}{2^{(\text{Gs})}}$$

**Figure 52-14. AFE Digital Signal Processing**




### 52.6.16 Buffer Structure

The DMA read channel is triggered each time a new data is stored in AFEC_LCDR. The same structure of data is repeatedly stored in AFEC_LCDR each time a trigger event occurs. Depending on the user mode of operation (AFEC_MR, AFEC_CHSR, AFEC_SEQ1R, AFEC_SEQ2R) the structure differs. When TAG is cleared, each data transferred to DMA buffer is carried on a half-word (16-bit) and consists of the last converted data right-aligned. When TAG is set, this data is carried on a word buffer (32-bit) and CHNB carries the channel number, thus simplifying post-processing in the DMA buffer and ensuring the integrity of the DMA buffer.

### 52.6.17 Fault Output

The AFEC internal fault output is directly connected to the PWM fault input. Fault output may be asserted depending on the configuration of AFEC_EMR, AFEC_CWR, AFEC_TEMPMR and AFEC_TEMPCWR and converted values.

Two types of comparison can trigger a compare event (fault output pulse). The first comparison type is based on AFEC_CWR settings, thus on all converted channels except the last one; the second type is linked to the last channel where temperature is measured. As an example, overcurrent and temperature exceeding limits can trigger a fault to PWM.

When the compare occurs, the AFEC fault output generates a pulse of one peripheral clock cycle to the PWM fault input. This fault line can be enabled or disabled within the PWM. If it is activated and asserted by the AFEC, the PWM outputs are immediately placed in a safe state (pure combinational path).

Note that the AFEC fault output connected to the PWM is not the COMPE bit. Thus the Fault Mode (FMOD) within the PWM configuration must be FMOD = 1.

### 52.6.18 Register Write Protection

To prevent any single software error from corrupting AFEC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the AFEC Write Protection Mode Register (AFEC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the AFEC Write Protection Status Register (AFEC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS flag is automatically cleared by reading the AFEC_WPSR.

The protected registers are:

- AFEC Mode Register
- AFEC Extended Mode Register
- AFEC Channel Sequence 1 Register
- AFEC Channel Sequence 2 Register
- AFEC Channel Enable Register
- AFEC Channel Disable Register

- AFEC Compare Window Register
- AFEC Channel Gain 1 Register
- AFEC Channel Differential Register
- AFEC Channel Selection Register
- AFEC Channel Offset Compensation Register
- AFEC Temperature Sensor Mode Register
- AFEC Temperature Compare Window Register
- AFEC Analog Control Register
- AFEC Sample & Hold Mode Register
- AFEC Correction Select Register
- AFEC Correction Values Register
- AFEC Channel Error Correction Register


© 2021 Microchip Technology Inc.
and its subsidiaries

## 52.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | AFEC_CR | 7:0 | | | | | | | START | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | AFEC_MR | 7:0 | FREERUN | FWUP | SLEEP | | TRGSEL[2:0] | | | TRGEN |
| | | 15:8 | PRESCAL[7:0] | | | | | | | |
| | | 23:16 | ONE | | | | STARTUP[3:0] | | | |
| | | 31:24 | USEQ | | TRANSFER[1:0] | | TRACKTIM[3:0] | | | |
| 0x08 | AFEC_EMR | 7:0 | CMPSEL[4:0] | | | | | | CMPMODE[1:0] | |
| | | 15:8 | | | CMPFILTER[1:0] | | | | CMPALL | |
| | | 23:16 | | | | | | RES[2:0] | | |
| | | 31:24 | | | SIGNMODE[1:0] | | | | STM | TAG |
| 0x0C | AFEC_SEQ1R | 7:0 | USCH1[3:0] | | | | USCH0[3:0] | | | |
| | | 15:8 | USCH3[3:0] | | | | USCH2[3:0] | | | |
| | | 23:16 | USCH5[3:0] | | | | USCH4[3:0] | | | |
| | | 31:24 | USCH7[3:0] | | | | USCH6[3:0] | | | |
| 0x10 | AFEC_SEQ2R | 7:0 | USCH9[3:0] | | | | USCH8[3:0] | | | |
| | | 15:8 | USCH11[3:0] | | | | USCH10[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | AFEC_CHER | 7:0 | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| | | 15:8 | | | | | CH11 | CH10 | CH9 | CH8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | AFEC_CHDR | 7:0 | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| | | 15:8 | | | | | CH11 | CH10 | CH9 | CH8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | AFEC_CHSR | 7:0 | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| | | 15:8 | | | | | CH11 | CH10 | CH9 | CH8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 | AFEC_LCDR | 7:0 | LDATA[7:0] | | | | | | | |
| | | 15:8 | LDATA[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | CHNB[3:0] | | | |
| 0x24 | AFEC_IER | 7:0 | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| | | 15:8 | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | TEMPCHG | | | | COMPE | GOVRE | DRDY |
| 0x28 | AFEC_IDR | 7:0 | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| | | 15:8 | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | TEMPCHG | | | | COMPE | GOVRE | DRDY |
| 0x2C | AFEC_IMR | 7:0 | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| | | 15:8 | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | TEMPCHG | | | | COMPE | GOVRE | DRDY |
| 0x30 | AFEC_ISR | 7:0 | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| | | 15:8 | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | TEMPCHG | | | | COMPE | GOVRE | DRDY |
| 0x34 ... 0x4B | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x4C | AFEC_OVER | 7:0 | OVRE7 | OVRE6 | OVRE5 | OVRE4 | OVRE3 | OVRE2 | OVRE1 | OVRE0 |
| | | 15:8 | | | | | OVRE11 | OVRE10 | OVRE9 | OVRE8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x50 | AFEC_CWR | 7:0 | LOWTHRES[7:0] | | | | | | | |
| | | 15:8 | LOWTHRES[15:8] | | | | | | | |
| | | 23:16 | HIGHTHRES[7:0] | | | | | | | |
| | | 31:24 | HIGHTHRES[15:8] | | | | | | | |
| 0x54 | AFEC_CGR | 7:0 | GAIN3[1:0] | | GAIN2[1:0] | | GAIN1[1:0] | | GAIN0[1:0] | |
| | | 15:8 | GAIN7[1:0] | | GAIN6[1:0] | | GAIN5[1:0] | | GAIN4[1:0] | |
| | | 23:16 | GAIN11[1:0] | | GAIN10[1:0] | | GAIN9[1:0] | | GAIN8[1:0] | |
| | | 31:24 | | | | | | | | |
| 0x58 ... 0x5F | Reserved | | | | | | | | | |
| 0x60 | AFEC_DIFFR | 7:0 | DIFF7 | DIFF6 | DIFF5 | DIFF4 | DIFF3 | DIFF2 | DIFF1 | DIFF0 |
| | | 15:8 | | | | | DIFF11 | DIFF10 | DIFF9 | DIFF8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x64 | AFEC_CSELR | 7:0 | | | | | CSEL[3:0] | | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x68 | AFEC_CDR | 7:0 | DATA[7:0] | | | | | | | |
| | | 15:8 | DATA[15:8] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x6C | AFEC_COCR | 7:0 | AOFF[7:0] | | | | | | | |
| | | 15:8 | | | | | | | AOFF[9:8] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x70 | AFEC_TEMPMR | 7:0 | | | TEMPCMPMOD[1:0] | | | | | RTCT |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x74 | AFEC_TEMPCWR | 7:0 | TLOWTHRES[7:0] | | | | | | | |
| | | 15:8 | TLOWTHRES[15:8] | | | | | | | |
| | | 23:16 | THIGHTHRES[7:0] | | | | | | | |
| | | 31:24 | THIGHTHRES[15:8] | | | | | | | |
| 0x78 ... 0x93 | Reserved | | | | | | | | | |
| 0x94 | AFEC_ACR | 7:0 | | | | | PGA1EN | PGA0EN | | |
| | | 15:8 | | | | | | | IBCTL[1:0] | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 ... 0x9F | Reserved | | | | | | | | | |
| 0xA0 | AFEC_SHMR | 7:0 | DUAL7 | DUAL6 | DUAL5 | DUAL4 | DUAL3 | DUAL2 | DUAL1 | DUAL0 |
| | | 15:8 | | | | | DUAL11 | DUAL10 | DUAL9 | DUAL8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xA4 ... 0xCF | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0xD0 | AFEC_COSR | 7:0 | | | | | | | | CSEL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xD4 | AFEC_CVR | 7:0 | OFFSETCORR[7:0] | | | | | | | |
| | | 15:8 | OFFSETCORR[15:8] | | | | | | | |
| | | 23:16 | GAINCORR[7:0] | | | | | | | |
| | | 31:24 | GAINCORR[15:8] | | | | | | | |
| 0xD8 | AFEC_CECR | 7:0 | ECORR7 | ECORR6 | ECORR5 | ECORR4 | ECORR3 | ECORR2 | ECORR1 | ECORR0 |
| | | 15:8 | | | | | ECORR11 | ECORR10 | ECORR9 | ECORR8 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0xDC ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | AFEC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | AFEC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | WPVSRC[15:8] | | | | | | | |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.1 AFEC Control Register

**Name:** AFEC_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | START | SWRST |
| Access | | | | | | | W | W |
| Reset | | | | | | | – | – |

**Bit 1 – START** Start Conversion

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Begins Analog Front-End conversion. |

**Bit 0 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the AFEC simulating a hardware reset. |

### 52.7.2 AFEC Mode Register

**Name:** AFEC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | USEQ | | TRANSFER[1:0] | | TRACKTIM[3:0] | | | |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ONE | | | | STARTUP[3:0] | | | |
| Access | R/W | | | | R/W | R/W | R/W | R/W |
| Reset | 0 | | | | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | PRESCAL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | FREERUN | FWUP | SLEEP | | TRGSEL[2:0] | | | TRGEN |
| Access | R/W | R/W | R/W | | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | | 0 | 0 | 0 | 0 |

**Bit 31 – USEQ** User Sequence Enable

| Value | Name | Description |
|---|---|---|
| 0 | NUM_ORDER | Normal mode: the controller converts channels in a simple numeric order. |
| 1 | REG_ORDER | User Sequence mode: the sequence is as defined in AFEC_SEQ1R and AFEC_SEQ1R. |

**Bits 29:28 – TRANSFER[1:0]** Transfer Period
Set to 2 to optimize transfer time.

**Bits 27:24 – TRACKTIM[3:0]** Tracking Time
Inherent tracking time is always 15 AFE clock cycles. Do not modify this field.

**Bit 23 – ONE** One
This bit must be written to 1.

**Bits 19:16 – STARTUP[3:0]** Startup Time

| Value | Name | Description |
|---|---|---|
| 0 | SUT0 | 0 periods of AFE clock |
| 1 | SUT8 | 8 periods of AFE clock |
| 2 | SUT16 | 16 periods of AFE clock |
| 3 | SUT24 | 24 periods of AFE clock |
| 4 | SUT64 | 64 periods of AFE clock |
| 5 | SUT80 | 80 periods of AFE clock |
| 6 | SUT96 | 96 periods of AFE clock |
| 7 | SUT112 | 112 periods of AFE clock |
| 8 | SUT512 | 512 periods of AFE clock |
| 9 | SUT576 | 576 periods of AFE clock |

© 2021 Microchip Technology Inc. and its subsidiaries

| Value | Name | Description |
|---|---|---|
| 10 | SUT640 | 640 periods of AFE clock |
| 11 | SUT704 | 704 periods of AFE clock |
| 12 | SUT768 | 768 periods of AFE clock |
| 13 | SUT832 | 832 periods of AFE clock |
| 14 | SUT896 | 896 periods of AFE clock |
| 15 | SUT960 | 960 periods of AFE clock |

**Bits 15:8 – PRESCAL[7:0]** Prescaler Rate Selection
PRESCAL = $f_{peripheral\ clock}$/ $f_{AFE\ Clock}$ - 1
When PRESCAL is cleared, no conversion is performed.

**Bit 7 – FREERUN** Free Run Mode

| Value | Name | Description |
|---|---|---|
| 0 | OFF | Normal mode |
| 1 | ON | Free Run mode: never wait for any trigger. |

**Bit 6 – FWUP** Fast Wakeup

| Value | Name | Description |
|---|---|---|
| 0 | OFF | Normal Sleep mode: the sleep mode is defined by the SLEEP bit. |
| 1 | ON | Fast Wakeup Sleep mode: the voltage reference is ON between conversions and AFE is OFF. |

**Bit 5 – SLEEP** Sleep Mode

| Value | Name | Description |
|---|---|---|
| 0 | NORMAL | Normal mode: the AFE and reference voltage circuitry are kept ON between conversions. |
| 1 | SLEEP | Sleep mode: the AFE and reference voltage circuitry are OFF between conversions. |

**Bits 3:1 – TRGSEL[2:0]** Trigger Selection

| Value | Name | Description |
|---|---|---|
| 0 | AFEC_TRIG0 | AFE0_ADTRG for AFEC0 / AFE1_ADTRG for AFEC1 |
| 1 | AFEC_TRIG1 | TIOA Output of the Timer Counter Channel 0 for AFEC0/TIOA Output of the Timer Counter Channel 3 for AFEC1 |
| 2 | AFEC_TRIG2 | TIOA Output of the Timer Counter Channel 1 for AFEC0/TIOA Output of the Timer Counter Channel 4 for AFEC1 |
| 3 | AFEC_TRIG3 | TIOA Output of the Timer Counter Channel 2 for AFEC0/TIOA Output of the Timer Counter Channel 5 for AFEC1 |
| 4 | AFEC_TRIG4 | PWM0 event line 0 for AFEC0 / PWM1 event line 0 for AFEC1 |
| 5 | AFEC_TRIG5 | PWM0 event line 1 for AFEC0 / PWM1 event line 1 for AFEC1 |
| 6 | AFEC_TRIG6 | Analog Comparator |
| 7 | Reserved | |

**Bit 0 – TRGEN** Trigger Enable

| Value | Name | Description |
|---|---|---|
| 0 | DIS | Hardware triggers are disabled. Starting a conversion is only possible by software. |
| 1 | EN | The hardware trigger selected by the TRGSEL field is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.3 AFEC Extended Mode Register

**Name:** AFEC_EMR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | SIGNMODE[1:0] | | | | STM | TAG |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | RES[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | CMPFILTER[1:0] | | | | CMPALL | |
| Access | | | R/W | R/W | | | R/W | |
| Reset | | | 0 | 0 | | | 0 | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CMPSEL[4:0] | | | | | | CMPMODE[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | | 0 | 0 |

**Bits 29:28 – SIGNMODE[1:0]** Sign Mode
If conversion results are signed and resolution is below 16 bits, the sign is extended up to the bit 15 (for example, 0xF43 for 12-bit resolution will be read as 0xFF43 and 0x467 will be read as 0x0467). See Conversion Results Format.

| Value | Name | Description |
|---|---|---|
| 0 | SE_UNSG_DF_SIGN | Single-Ended channels: unsigned conversions.<br>Differential channels: signed conversions. |
| 1 | SE_SIGN_DF_UNSG | Single-Ended channels: signed conversions.<br>Differential channels: unsigned conversions. |
| 2 | ALL_UNSIGNED | All channels: unsigned conversions. |
| 3 | ALL_SIGNED | All channels: signed conversions. |

**Bit 25 – STM** Single Trigger Mode

| Value | Description |
|---|---|
| 0 | Multiple triggers are required to get an averaged result. |
| 1 | Only a single trigger is required to get an averaged value. |

**Bit 24 – TAG** Tag for AFEC_LCDR

| Value | Description |
|---|---|
| 0 | Clears CHNB in AFEC_LCDR. |
| 1 | Appends the channel number to the conversion result in AFEC_LCDR. |

**Bits 18:16 – RES[2:0]** Resolution

| Value | Name | Description |
|---|---|---|
| 0 | NO_AVERAGE | 12-bit resolution, AFE sample rate is maximum (no averaging). |
| 1 | LOW_RES | 10-bit resolution, AFE sample rate is maximum (no averaging). |
| 2 | OSR4 | 13-bit resolution, AFE sample rate divided by 4 (averaging). |

© 2021 Microchip Technology Inc. and its subsidiaries

| Value | Name | Description |
|---|---|---|
| 3 | OSR16 | 14-bit resolution, AFE sample rate divided by 16 (averaging). |
| 4 | OSR64 | 15-bit resolution, AFE sample rate divided by 64 (averaging). |
| 5 | OSR256 | 16-bit resolution, AFE sample rate divided by 256 (averaging). |

**Bits 13:12 – CMPFILTER[1:0]** Compare Event Filtering
Number of consecutive compare events necessary to raise the flag = CMPFILTER+1.
When programmed to '0', the flag rises as soon as an event occurs.

**Bit 9 – CMPALL** Compare All Channels

| Value | Description |
|---|---|
| 0 | Only the channel indicated in CMPSEL field is compared. |
| 1 | All channels are compared. |

**Bits 7:3 – CMPSEL[4:0]** Comparison Selected Channel
If CMPALL = 0: CMPSEL indicates which channel has to be compared.
If CMPALL = 1: No effect.

**Bits 1:0 – CMPMODE[1:0]** Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | LOW | Generates an event when the converted data is lower than the low threshold of the window. |
| 1 | HIGH | Generates an event when the converted data is higher than the high threshold of the window. |
| 2 | IN | Generates an event when the converted data is in the comparison window. |
| 3 | OUT | Generates an event when the converted data is out of the comparison window. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.4 AFEC Channel Sequence 1 Register

**Name:** AFEC_SEQ1R
**Offset:** 0x0C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | USCH7[3:0] | | | | USCH6[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | USCH5[3:0] | | | | USCH4[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | USCH3[3:0] | | | | USCH2[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | USCH1[3:0] | | | | USCH0[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:3, 4:7, 8:11, 12:15, 16:19, 20:23, 24:27, 28:31 – USCHx** User Sequence Number x
The sequence number x (USCHx) can be programmed by the Channel number CHy where y is the value written in this field. The allowed range is 0 up to 11. So it is only possible to use the sequencer from CH0 to CH11.
This register activates only if AFEC_MR.USEQ is set.
Any USCHx field is taken into account only if AFEC_CHSR.CHx is set, else any value written in USCHx does not add the corresponding channel in the conversion sequence.
Configuring the same value in different fields leads to multiple samples of the same channel during the conversion sequence. This can be done consecutively, or not, depending on user requirements.

### 52.7.5 AFEC Channel Sequence 2 Register

**Name:** AFEC_SEQ2R
**Offset:** 0x10
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | USCH11[3:0] | | | | USCH10[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | USCH9[3:0] | | | | USCH8[3:0] | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:3, 4:7, 8:11, 12:15 – USCHx** User Sequence Number x
The sequence number x (USCHx) can be programmed by the Channel number CHy where y is the value written in this field. The allowed range is 0 up to 11. So it is only possible to use the sequencer from CH0 to CH11.
This register activates only if AFEC_MR.USEQ is set.
Any USCHx field is taken into account only if AFEC_CHSR.CHx is written to one, else any value written in USCHx does not add the corresponding channel in the conversion sequence.
Configuring the same value in different fields leads to multiple samples of the same channel during the conversion sequence. This can be done consecutively, or not, depending on user requirements.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.6 AFEC Channel Enable Register

**Name:** AFEC_CHER
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CH11 | CH10 | CH9 | CH8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – CHx** Channel x Enable
If AFEC_MR.USEQ = 1, CHx corresponds to the xth channel of the sequence described in AFEC_SEQ1R, AFEC_SEQ2R.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding channel. |

### 52.7.7 AFEC Channel Disable Register

**Name:** AFEC_CHDR
**Offset:** 0x18
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CH11 | CH10 | CH9 | CH8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – CHx** Channel x Disable

**WARNING** If the corresponding channel is disabled during a conversion, or if it is disabled and then reenabled during a conversion, its associated data and its corresponding EOCx and GOVRE flags in AFEC_ISR and OVREx flags in AFEC_OVER are unpredictable.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding channel. |

### 52.7.8 AFEC Channel Status Register

**Name:** AFEC_CHSR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CH11 | CH10 | CH9 | CH8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CH7 | CH6 | CH5 | CH4 | CH3 | CH2 | CH1 | CH0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – CHx** Channel x Status

| Value | Description |
|---|---|
| 0 | The corresponding channel is disabled. |
| 1 | The corresponding channel is enabled. |

### 52.7.9 AFEC Last Converted Data Register

**Name:** AFEC_LCDR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CHNB[3:0] | | | |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LDATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LDATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 27:24 – CHNB[3:0]** Channel Number
Indicates the last converted channel when AFEC_EMR.TAG is set. If AFEC_EMR.TAG is cleared, CHNB = 0.

**Bits 15:0 – LDATA[15:0]** Last Data Converted
The AFE conversion data is placed into this register at the end of a conversion and remains until a new conversion is completed.

### 52.7.10 AFEC Interrupt Enable 1 Register

**Name:** AFEC_IER
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | TEMPCHG | | | | COMPE | GOVRE | DRDY |
| Access | | W | | | | W | W | W |
| Reset | | – | | | | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 30 – TEMPCHG** Temperature Change Interrupt Enable

**Bit 26 – COMPE** Comparison Event Interrupt Enable

**Bit 25 – GOVRE** General Overrun Error Interrupt Enable

**Bit 24 – DRDY** Data Ready Interrupt Enable

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – EOCx** End of Conversion Interrupt Enable x

### 52.7.11 AFEC Interrupt Disable Register

**Name:** AFEC_IDR
**Offset:** 0x28
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | TEMPCHG | | | | COMPE | GOVRE | DRDY |
| Access | | W | | | | W | W | W |
| Reset | | – | | | | – | – | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| Access | | | | | W | W | W | W |
| Reset | | | | | – | – | – | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| Access | W | W | W | W | W | W | W | W |
| Reset | – | – | – | – | – | – | – | – |

**Bit 30 – TEMPCHG** Temperature Change Interrupt Disable

**Bit 26 – COMPE** Comparison Event Interrupt Disable

**Bit 25 – GOVRE** General Overrun Error Interrupt Disable

**Bit 24 – DRDY** Data Ready Interrupt Disable

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – EOCx** End of Conversion Interrupt Disable x

### 52.7.12 AFEC Interrupt Mask Register

**Name:** AFEC_IMR
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | TEMPCHG | | | | COMPE | GOVRE | DRDY |
| Access | | R | | | | R | R | R |
| Reset | | 0 | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – TEMPCHG** Temperature Change Interrupt Mask

**Bit 26 – COMPE** Comparison Event Interrupt Mask

**Bit 25 – GOVRE** General Overrun Error Interrupt Mask

**Bit 24 – DRDY** Data Ready Interrupt Mask

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – EOCx** End of Conversion Interrupt Mask x

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.13 AFEC Interrupt Status Register

**Name:** AFEC_ISR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | TEMPCHG | | | | COMPE | GOVRE | DRDY |
| Access | | R | | | | R | R | R |
| Reset | | 0 | | | | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | EOC11 | EOC10 | EOC9 | EOC8 |
| Access | | | | | | | | |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | EOC7 | EOC6 | EOC5 | EOC4 | EOC3 | EOC2 | EOC1 | EOC0 |
| Access | | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – TEMPCHG** Temperature Change (cleared on read)

| Value | Description |
|---|---|
| 0 | No comparison match (defined in AFEC_TEMPCMPR) occurred since the last read of AFEC_ISR. |
| 1 | The temperature value reported on AFEC_CDR (AFEC_CSELR.CSEL = 11) has changed since the last read of AFEC_ISR, according to what is defined in the Temperature Mode register (AFEC_TEMPMR) and the Temperature Compare Window register (AFEC_TEMPCWR). |

**Bit 26 – COMPE** Comparison Error (cleared by reading AFEC_ISR)

| Value | Description |
|---|---|
| 0 | No comparison error since the last read of AFEC_ISR. |
| 1 | At least one comparison error has occurred since the last read of AFEC_ISR. |

**Bit 25 – GOVRE** General Overrun Error (cleared by reading AFEC_ISR)

| Value | Description |
|---|---|
| 0 | No general overrun error occurred since the last read of AFEC_ISR. |
| 1 | At least one general overrun error has occurred since the last read of AFEC_ISR. |

**Bit 24 – DRDY** Data Ready (cleared by reading AFEC_LCDR)

| Value | Description |
|---|---|
| 0 | No data has been converted since the last read of AFEC_LCDR. |
| 1 | At least one data has been converted and is available in AFEC_LCDR. |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – EOCx** End of Conversion x (cleared by reading AFEC_CDRx)

| Value | Description |
|---|---|
| 0 | The corresponding analog channel is disabled, or the conversion is not finished. This flag is cleared when reading the AFEC_CDR if the CSEL bit is programmed with ‘x’ in the AFEC_CSELR. |
| 1 | The corresponding analog channel is enabled and conversion is complete. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 52.7.14 AFEC Overrun Status Register

**Name:** AFEC_OVER
**Offset:** 0x4C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | OVRE11 | OVRE10 | OVRE9 | OVRE8 |
| Access | | | | | R | R | R | R |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OVRE7 | OVRE6 | OVRE5 | OVRE4 | OVRE3 | OVRE2 | OVRE1 | OVRE0 |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – OVREx** Overrun Error x
An overrun error does not always mean that the unread data has been replaced by a new valid data. See Enhanced Resolution Mode and Digital Averaging Function for details.

| Value | Description |
|---|---|
| 0 | No overrun error on the corresponding channel since the last read of AFEC_OVER. |
| 1 | There has been an overrun error on the corresponding channel since the last read of AFEC_OVER. |

### 52.7.15 AFEC Compare Window Register

**Name:** AFEC_CWR
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HIGHTHRES[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HIGHTHRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LOWTHRES[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | LOWTHRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – HIGHTHRES[15:0]** High Threshold
High threshold associated to compare settings of AFEC_EMR. For comparisons lower than 16 bits and signed, the sign should be extended up to the bit 15.

**Bits 15:0 – LOWTHRES[15:0]** Low Threshold
Low threshold associated to compare settings of AFEC_EMR. For comparisons lower than 16 bits and signed, the sign should be extended up to the bit 15.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.16 AFEC Channel Gain Register

**Name:** AFEC_CGR
**Offset:** 0x54
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GAIN11[1:0] | | GAIN10[1:0] | | GAIN9[1:0] | | GAIN8[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GAIN7[1:0] | | GAIN6[1:0] | | GAIN5[1:0] | | GAIN4[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GAIN3[1:0] | | GAIN2[1:0] | | GAIN1[1:0] | | GAIN0[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0:1, 2:3, 4:5, 6:7, 8:9, 10:11, 12:13, 14:15, 16:17, 18:19, 20:21, 22:23 – GAINx** Gain for Channel x
Gain applied on input of Analog Front-End.
See section AFEC Channel Differential Register for a description of DIFFx.

| GAINx | Gain Applied | |
|---|---|---|
| | DIFFx = 0 | DIFFx = 1 |
| 0 | 1 | 1 |
| 1 | 2 | 2 |
| 2 | 4 | 4 |
| 3 | 4 | 4 |

### 52.7.17 AFEC Channel Differential Register

**Name:** AFEC_DIFFR
**Offset:** 0x60
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DIFF11 | DIFF10 | DIFF9 | DIFF8 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DIFF7 | DIFF6 | DIFF5 | DIFF4 | DIFF3 | DIFF2 | DIFF1 | DIFF0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – DIFFx** Differential Inputs for Channel x

| Value | Description |
|---|---|
| 0 | Single-ended mode. |
| 1 | Fully differential mode. |

### 52.7.18 AFEC Channel Selection Register

**Name:** AFEC_CSELR
**Offset:** 0x64
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | CSEL[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 3:0 – CSEL[3:0]** Channel Selection

| Value | Description |
|---|---|
| 0-11 | Selects the channel to be displayed in AFEC_CDR and AFEC_COCR. To be configured with the appropriate channel number. |

### 52.7.19 AFEC Channel Data Register

**Name:** AFEC_CDR
**Offset:** 0x68
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – DATA[15:0]** Converted Data
Returns the AFE conversion data corresponding to channel CSEL (configured in the AFEC Channel Selection Register).
At the end of a conversion, the converted data is loaded into one of the 12 internal registers (one for each channel) and remains in this internal register until a new conversion is completed on the same channel index. The AFEC_CDR together with AFEC_CSELR allows to multiplex all the internal channel data registers.
The data carried on AFEC_CDR is valid only if AFEC_CHSR.CHx bit is set (where x = AFEC_CSELR.CSEL field value).

### 52.7.20 AFEC Channel Offset Compensation Register

**Name:** AFEC_COCR
**Offset:** 0x6C
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | AOFF[9:8] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AOFF[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 9:0 – AOFF[9:0]** Analog Offset
Defines the analog offset to be used for channel CSEL (configured in the AFEC Channel Selection Register). This value is used as an input value for the DAC included in the AFE.
**Note:**
The field AOFF must be configured to 512 (mid scale of the DAC) when there is no offset error to compensate. To compensate for an offset error of n LSB (positive or negative), the field AOFF must be configured to 512 + n.

### 52.7.21 AFEC Temperature Sensor Mode Register

**Name:** AFEC_TEMPMR
**Offset:** 0x70
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | TEMPCMPMOD[1:0] | | | | | RTCT |
| Access | | | R/W | R/W | | | | R/W |
| Reset | | | 0 | 0 | | | | 0 |

**Bits 5:4 – TEMPCMPMOD[1:0]** Temperature Comparison Mode

| Value | Name | Description |
|---|---|---|
| 0 | LOW | Generates an event when the converted data is lower than the low threshold of the window. |
| 1 | HIGH | Generates an event when the converted data is higher than the high threshold of the window. |
| 2 | IN | Generates an event when the converted data is in the comparison window. |
| 3 | OUT | Generates an event when the converted data is out of the comparison window. |

**Bit 0 – RTCT** Temperature Sensor RTC Trigger Mode

| Value | Description |
|---|---|
| 0 | The temperature sensor measure is not triggered by RTC event. |
| 1 | The temperature sensor measure is triggered by RTC event (if TRGEN = 1). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.22 AFEC Temperature Compare Window Register

**Name:** AFEC_TEMPCWR
**Offset:** 0x74
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | THIGHTHRES[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | THIGHTHRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TLOWTHRES[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TLOWTHRES[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – THIGHTHRES[15:0]** Temperature High Threshold
High threshold associated to compare settings of the AFEC_TEMPMR. For comparisons less than 16 bits and signed, the sign should be extended up to the bit 15.

**Bits 15:0 – TLOWTHRES[15:0]** Temperature Low Threshold
Low threshold associated to compare settings of the AFEC_TEMPMR. For comparisons less than 16 bits and signed, the sign should be extended up to the bit 15.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.23 AFEC Analog Control Register

**Name:** AFEC_ACR
**Offset:** 0x94
**Reset:** 0x00000100
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | IBCTL[1:0] | |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 1 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PGA1EN | PGA0EN | | |
| Access | | | | | R/W | R/W | | |
| Reset | | | | | 0 | 0 | | |

**Bits 9:8 – IBCTL[1:0]** AFE Bias Current Control
Adapts performance versus power consumption. (Refer to AFE Characteristics in section “Electrical Characteristics”.)

**Bit 3 – PGA1EN** PGA1 Enable

| Value | Description |
|---|---|
| 0 | Programmable Gain Amplifier is disabled. |
| 1 | Programmable Gain Amplifier is enabled. |

**Bit 2 – PGA0EN** PGA0 Enable

| Value | Description |
|---|---|
| 0 | Programmable Gain Amplifier is disabled. |
| 1 | Programmable Gain Amplifier is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 52.7.24 AFEC Sample & Hold Mode Register

**Name:** AFEC_SHMR
**Offset:** 0xA0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | DUAL11 | DUAL10 | DUAL9 | DUAL8 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DUAL7 | DUAL6 | DUAL5 | DUAL4 | DUAL3 | DUAL2 | DUAL1 | DUAL0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – DUALx** Dual Sample & Hold for Channel x

| Value | Description |
|---|---|
| 0 | Single Sample-and-Hold mode. |
| 1 | Dual Sample-and-Hold mode. |

### 52.7.25 AFEC Correction Select Register

**Name:** AFEC_COSR
**Offset:** 0xD0
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – CSEL** Sample & Hold unit Correction Select
Selects the Sample & Hold unit to be displayed in the AFEC_CVR.

### 52.7.26 AFEC Correction Values Register

**Name:** AFEC_CVR
**Offset:** 0xD4
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | GAINCORR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GAINCORR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | OFFSETCORR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | OFFSETCORR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:16 – GAINCORR[15:0]** Gain Correction
Gain correction to apply on converted data. Only bits 0 to 15 are relevant (other bits are ignored and read as 0).

**Bits 15:0 – OFFSETCORR[15:0]** Offset Correction
Offset correction to apply on converted data. The offset is signed (2's complement), only bits 0 to 11 are relevant (other bits are ignored and read as 0).

### 52.7.27 AFEC Channel Error Correction Register

**Name:** AFEC_CECR
**Offset:** 0xD8
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the AFEC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | ECORR11 | ECORR10 | ECORR9 | ECORR8 |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ECORR7 | ECORR6 | ECORR5 | ECORR4 | ECORR3 | ECORR2 | ECORR1 | ECORR0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 – ECORRx** Error Correction Enable for Channel x

| Value | Description |
|---|---|
| 0 | Automatic error correction is disabled for channel x. |
| 1 | Automatic error correction is enabled for channel x. |

### 52.7.28 AFEC Write Protection Mode Register

**Name:** AFEC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protect KEY

| Value | Name | Description |
|---|---|---|
| 0x414443 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

See Register Write Protection for the list of registers which can be protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x414443 (“ADC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x414443 (“ADC” in ASCII). |

### 52.7.29 AFEC Write Protection Status Register

**Name:** AFEC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 23:8 – WPVSRC[15:0]** Write Protect Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protect Violation Status

| Value | Description |
|---|---|
| 0 | No Write Protect Violation has occurred since the last read of the AFEC_WPSR. |
| 1 | A Write Protect Violation has occurred since the last read of the AFEC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 53. Digital-to-Analog Converter Controller (DACC)

## 53.1 Description

The Digital-to-Analog Converter Controller (DACC) offers up to two single-ended analog outputs or one differential analog output, making it possible for the digital-to-analog conversion to drive up to two independent analog lines.

The DACC supports 12-bit resolution.

The DACC operates in Free-running mode, Max speed mode, Trigger mode or Interpolation mode.

The DACC integrates a Bypass mode which minimizes power consumption in case of a limited sampling rate conversion.

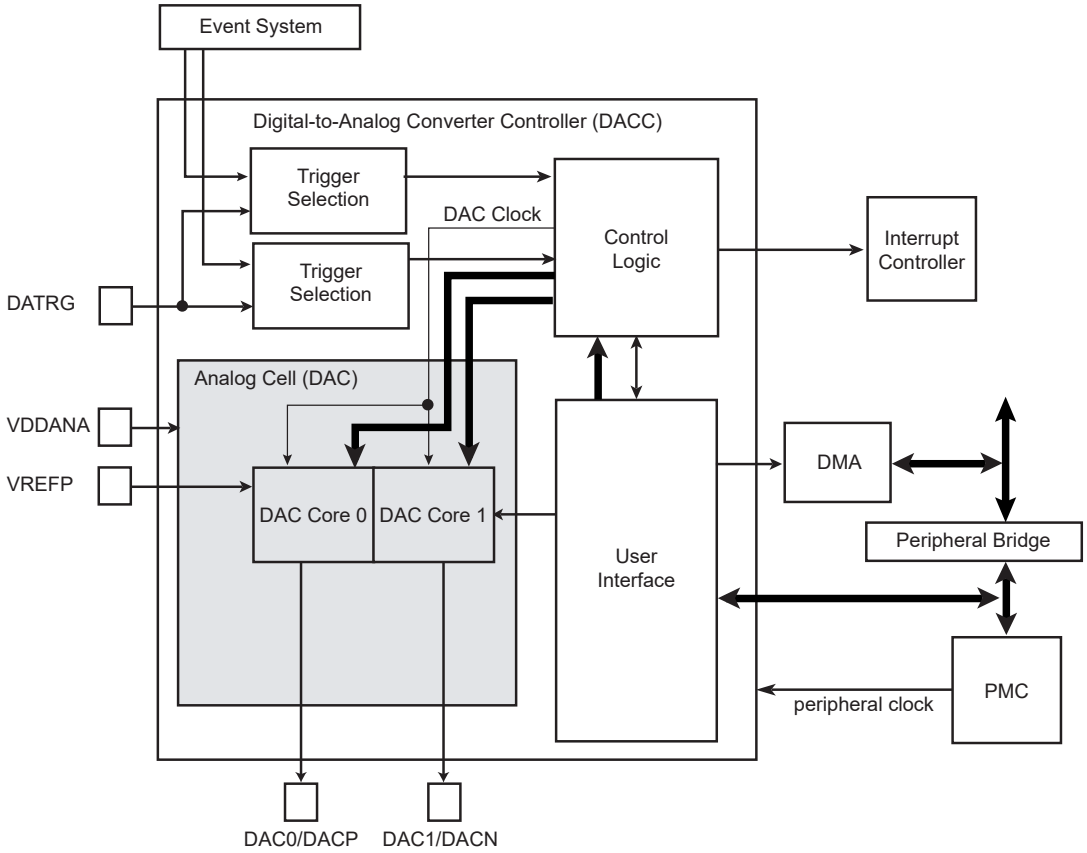
Each channel connects with a separate DMA channel. This feature reduces both power consumption and processor intervention.

## 53.2 Embedded Characteristics

- Up to Two Independent Single-Ended Analog Outputs or One Differential Analog Output
- 12-bit Resolution
- Integrated Interpolation Filter with 2×, 4×, 8×, 16× or 32× Oversampling Ratio (OSR)
- Reduced Number of System Bus Accesses (Word Transfer Mode)
- Individual Control of Each Analog Channel
- Hardware Triggers
  - One Trigger Selection Per Channel
    – External trigger pin
    – Internal events
- DMA Support
- One Internal FIFO per Channel
- Register Write Protection

## 53.3 Block Diagram

**Figure 53-1. Block Diagram**




## 53.4 Signal Description

**Table 53-1. DACC Signal Description**

| Name | Description | Direction |
|---|---|---|
| DAC0/DACP | Single-ended analog output channel 0 / Positive channel of differential analog output channel | Output |
| DAC1/DACN | Single-ended analog output channel 1 / Negative channel of differential analog output channel | Output |
| DATRG | Trigger | Input |
| VREFP | Positive reference voltage connected to VREFP | Input |
| VREFN | Negative reference voltage connected to VREFN | Input |

## 53.5 Product Dependencies

### 53.5.1 I/O Lines

The digital input DATRG is multiplexed with digital functions on the I/O line and is selected using the PIO Controller.

The analog outputs DAC0/DACP, DAC1/DACN are multiplexed with digital functions on the I/O lines .The analog outputs of the DACC drive the pads and the digital functions are not selected when the corresponding DACC channels are enabled by writing to the DACC Channel Enable Register (DACC_CHER).

### 53.5.2 Power Management

The programmer must first enable the DACC Clock in the Power Management Controller (PMC) before using the DACC.

The DACC becomes active as soon as a conversion is requested and at least one channel is enabled. The DACC is automatically deactivated when no channels are enabled.

### 53.5.3 Interrupt Sources

The DACC interrupt line is connected on one of the internal sources of the Interrupt controller. Using the DACC interrupt requires the Interrupt controller to be programmed first.

### 53.5.4 Conversion Performances

For performance and electrical characteristics of the DACC, see the DACC Characteristics in the section “Electrical Characteristics”.

## 53.6 Functional Description

### 53.6.1 Digital-to-Analog Conversion

To perform conversions, the DACC_CHSR.CHx bit must be set by writing a one to DACC_CHER.CHx. If both DACC_CHSR.CHx bits are cleared, the DAC analog cell is switched off. The DACx is ready to convert once DACC_CHSR.DACRDYx is read at ‘1’.

The DACC divides the peripheral clock to perform conversions. This divided clock is named DAC clock. Once a conversion starts, the DACC takes 12 DAC clock periods to provide the analog result on the selected analog output.

The minimum conversion period of the DAC is exactly 12 DAC clock periods when the Max speed mode is enabled (MAXSx = 1 in the DACC Mode Register (DACC_MR)). In this case the DAC is always clocked, slightly increasing the power consumption of the DAC.

When the Max speed mode is not used (Trigger or Free-running mode), the DAC is only clocked when a conversion is requested and a new conversion can only occur when the DAC has ended its previous conversion. The power consumption is lower but the sampling rate is lower as the controller waits for the end of conversion of the previously sent data. In this case, one conversion lasts 12 DAC clock periods plus 2 cycles of resynchronization stage.

The conversion mode of a channel can be modified only if this channel has been previously disabled.

Power consumption of the DAC can be adapted to its sampling rate via the DACC_ACR.IBCTLCHx fields.

In Bypass mode, the maximum sample rate and the power consumption of the DAC are lowered.

### 53.6.2 Conversion Results

When a conversion is completed, the resulting analog value is available at the selected DAC channel output. The EOC bit in the DACC Interrupt Status Register (DACC_ISR) is set.

Reading DACC_ISR clears the EOC bit.

### 53.6.3 Analog Output Mode Selection

The analog outputs can be set to either Single-ended or Differential mode with the DIFF bit in the DACC_MR.

© 2021 Microchip Technology Inc.
and its subsidiaries

When set to Single-ended mode (DIFF = 0), each DAC channel can be configured independently.

When set to Differential mode (DIFF = 1), the analog outputs DACP and DACN are located on DAC0 and DAC1 outputs, respectively. All operations are driven by channel 0 and activating this channel automatically activates channel 1. Sending a value on channel 0 (DACP) automatically generates the complementary signal to be sent to channel 1 (DACN). The signal sent to the DAC is centered around 2048. For example, sending 3000 = 2048 + 952 to the DAC0 channel will automatically send 1096 = 2048 - 952 to the DAC1 channel.

### 53.6.4 Conversion Modes

The conversion modes available in the DACC are described below.

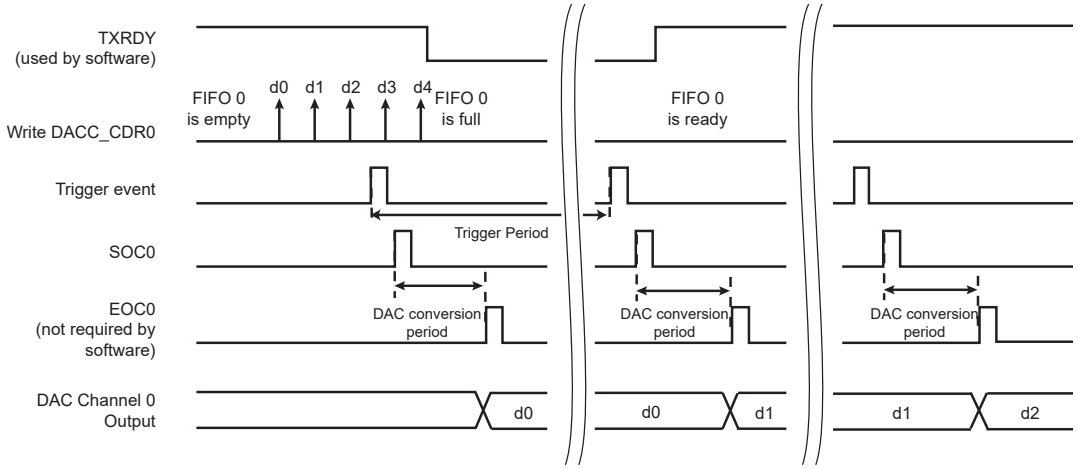
#### 53.6.4.1 Trigger Mode

Trigger mode is enabled by setting DACC_TRIGR.TRGENx.

The conversion waits for a rising edge on the selected trigger to send the data to the DAC. In this mode, the maximum data rate (i.e., the maximum trigger event frequency) cannot exceed 12 DAC clock periods plus 2 cycles of resynchronization stage.

**Note:** Disabling Trigger mode (TRGENx = 0) automatically sets the DACC in Free-running or Max speed mode depending on the status of DACC_MR.MAXSx.

**Figure 53-2. Conversion Sequence in Trigger Mode**



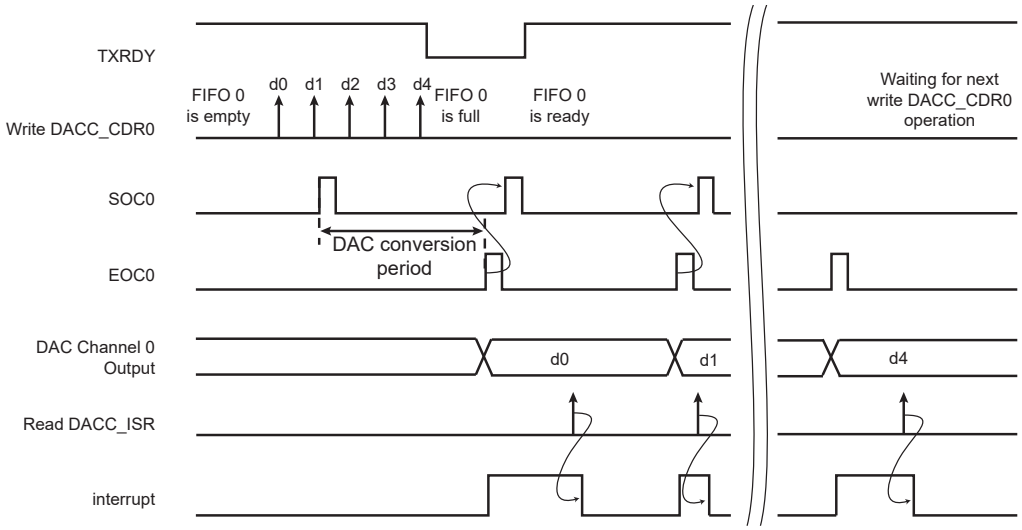

#### 53.6.4.2 Free-Running Mode

Free-running mode is enabled by clearing DACC_TRIGR.TRGENx and DACC_MR.MAXSx.

The conversion starts as soon as at least one channel is enabled. Once data is written in the DACC Conversion Data Register (DACC_CDRx), 12 DAC clock periods later, the converted data is available at the corresponding analog output. The next data is converted only when the EOC of the previous data is set.

If the FIFO is emptied, no conversion occurs and the data is maintained at the output of the DAC.

**Figure 53-3. Conversion Sequence in Free-running Mode**




### 53.6.4.3 Max Speed Mode

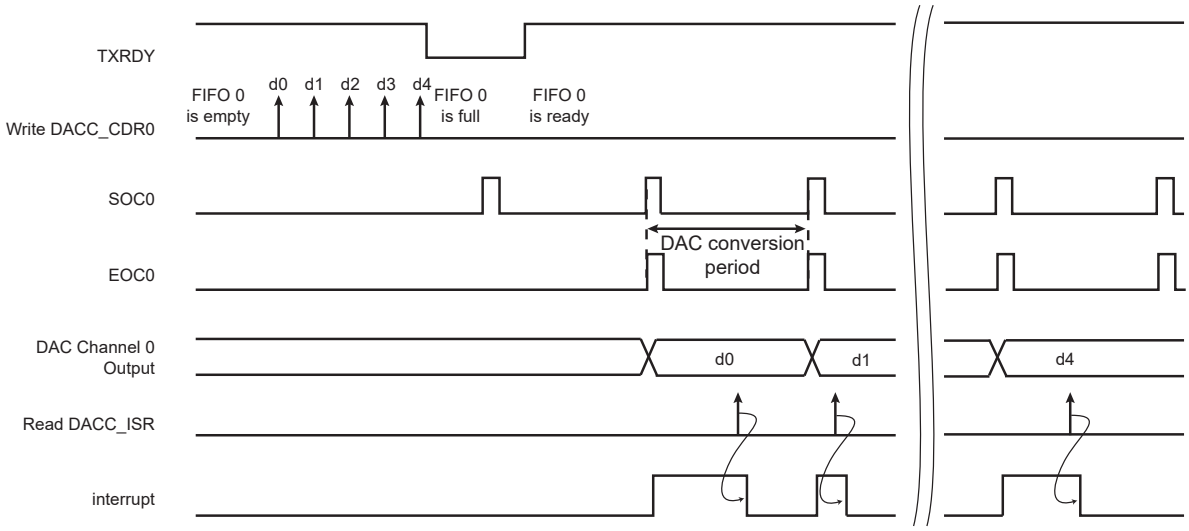
Max speed mode is enabled by setting DACC_TRIGR.TRGENx and DACC_MR.MAXSx.

The conversion rate is forced by the controller, which starts one conversion every 12 DAC clock periods. The controller does not wait for the EOC of the previous data to send a new data to the DAC and the DAC is always clocked.

If the FIFO is emptied, the controller send the last converted data to the DAC at a rate of 12 DAC clock periods.

The DACC_ACR.IBCTLCHx field must be configured for 1 MSps (see the section “Electrical Characteristics”).

**Figure 53-4. Conversion Sequence in Max Speed Mode**




### 53.6.4.4 Bypass Mode

Bypass mode disables the DAC output buffer and thus minimizes power consumption. This mode can be used to generate slow varying signals. Refer to the DAC Characteristics in the section “Electrical Characteristics” of this datasheet.

To enter this mode, Free-running mode must be selected and the DACC_ACR.IBCTLCHx field configured in Bypass mode.

### 53.6.4.5 Interpolation Mode

The DACC integrates interpolation filters that allow OSR of 2×, 4×, 8×, 16× or 32×. This mode can be used only if Trigger mode is enabled and value in the field OSRx is not '0'. The OSR of the interpolator is configured in the OSRx field in the DACC Trigger Register (DACC_TRIGR).

The data is sampled once every OSR trigger event and then recomputed at the trigger sample rate using a third-order SINC filter. This reduces the number of accesses to the DACC and increases the signal-to-noise ratio (SNR) of the converted output signal.

The figures below show the spectral mask of the SINC filter depending on the selected OSR. $f_s$ is the sampling frequency of the input signal which corresponds to the trigger frequency divided by OSR.

**Figure 53-5. Interpolator Spectral Mask for OSR = 2**

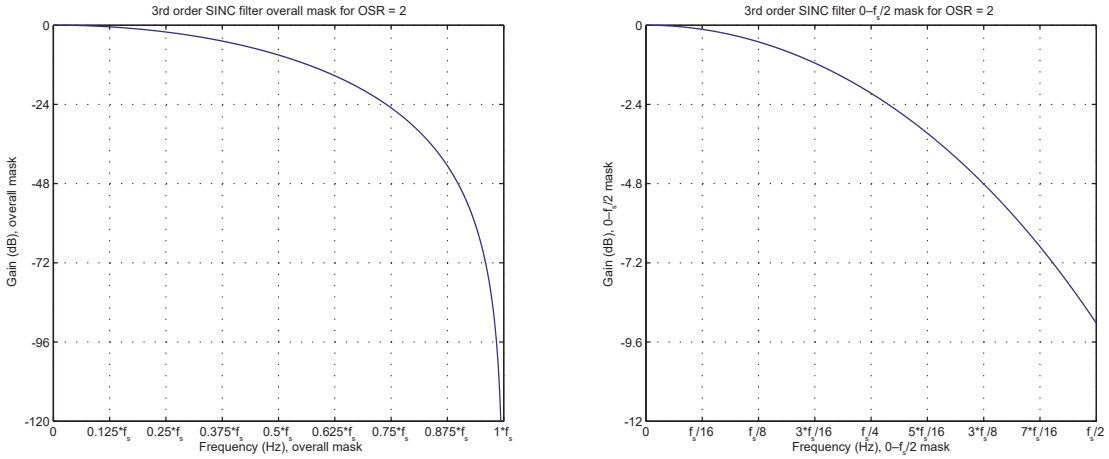



**Figure 53-6. Interpolator Spectral Mask for OSR = 4**

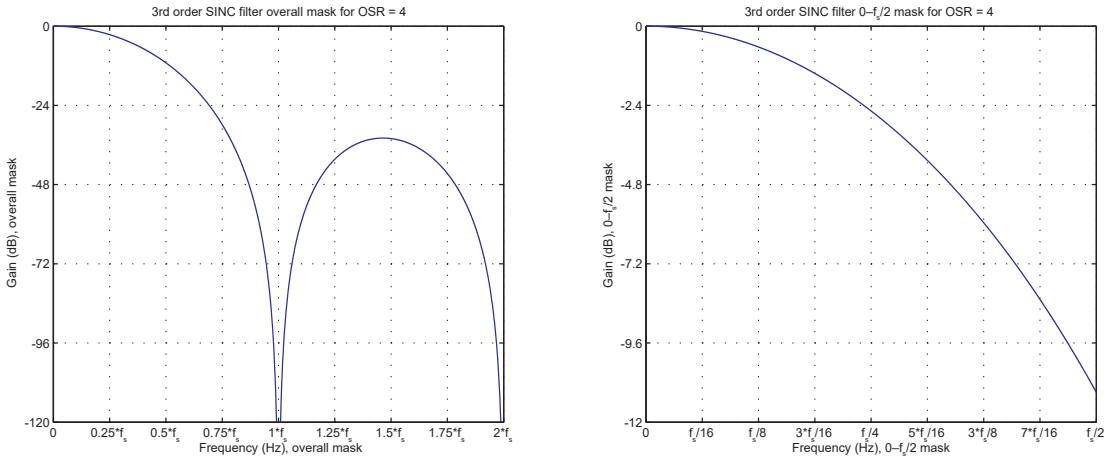


**Figure 53-7. Interpolator Spectral Mask for OSR = 8**

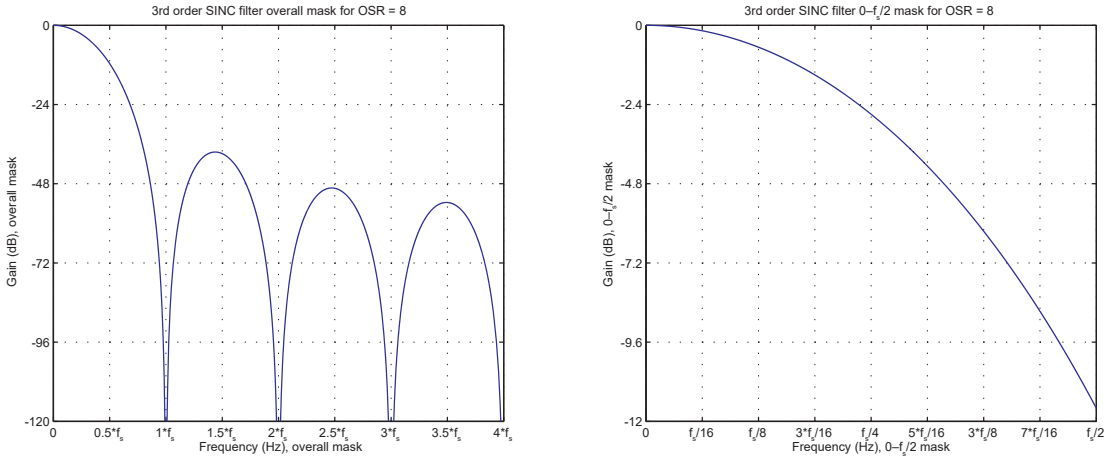



**Figure 53-8. Interpolator Spectral Mask for OSR = 16**

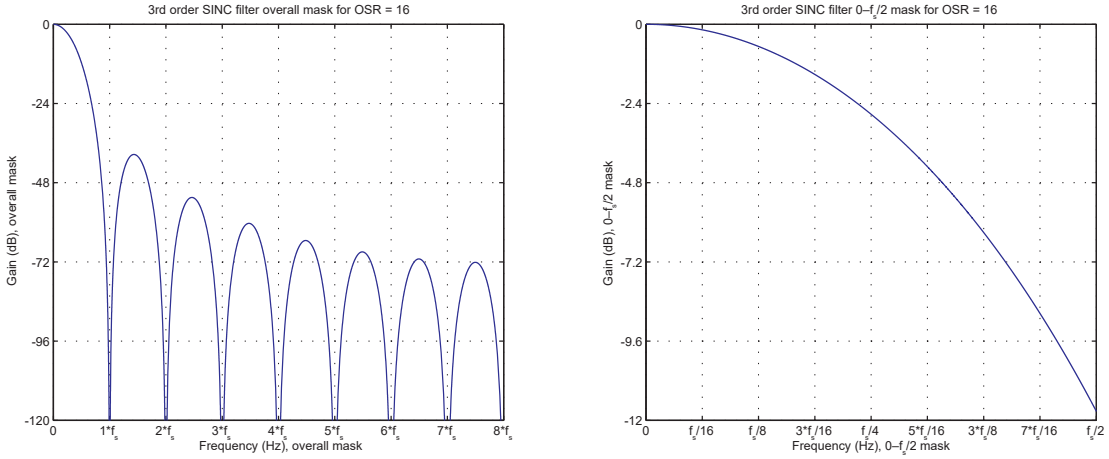
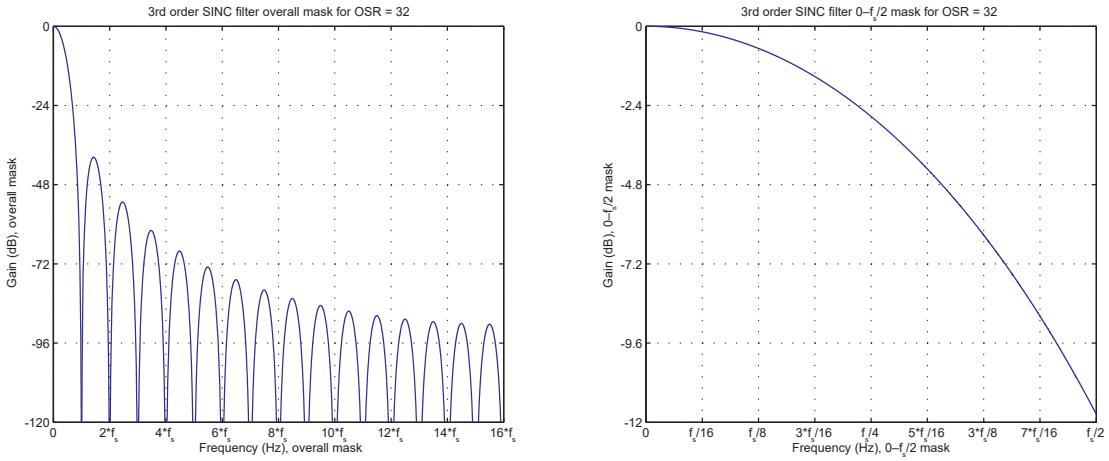



**Figure 53-9. Interpolator Spectral Mask for OSR = 32**




### 53.6.5 Conversion FIFO

Each channel embeds a four half-word FIFO to handle the data to be converted.

When the TXRDY flag of a channel in the DACC_ISR is active, the DACC is ready to accept conversion requests by writing data into the corresponding DACC_CDRx. Data which cannot be converted immediately are stored in the FIFO of the corresponding channel.

When the FIFO is full or the DACC is not ready to accept conversion requests, the TXRDY flag is inactive.

The DACC also offers the possibility of writing two data words in one access by setting the bit WORD in the DACC_MR. In this case, bits 11:0 contain the first data to be converted and bits 27:16 contain the second data to be converted. The two data are written into the FIFO of the selected channel. The TXRDY flag takes into account this double write access. Changing this access mode implies first switching off all channels.

**⚠WARNING** Writing in DACC_CDRx while TXRDY flag is inactive will corrupt FIFO data.

### 53.6.6 Register Write Protection

To prevent any single software error from corrupting DACC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the DACC Write Protection Mode Register (DACC_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the DACC Write Protection Status Register (DACC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the DACC_WPSR.

The following registers can be write-protected :

- DACC Mode Register
- DACC Channel Enable Register
- DACC Channel Disable Register
- DACC Analog Current Register
- DACC Trigger Register

## 53.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | DACC_CR | 7:0 | | | | | | | | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | DACC_MR | 7:0 | | | ZERO | WORD | | | MAXS1 | MAXS0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | DIFF | | | | | | | |
| | | 31:24 | | | | | PRESCALER[3:0] | | | |
| 0x08 | DACC_TRIGR | 7:0 | | TRGSEL0[2:0] | | | | | TRGEN1 | TRGEN0 |
| | | 15:8 | | | | | | TRGSEL1[2:0] | | |
| | | 23:16 | | OSR1[2:0] | | | | OSR0[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | DACC_CHER | 7:0 | | | | | | | CH1 | CH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | DACC_CHDR | 7:0 | | | | | | | CH1 | CH0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | DACC_CHSR | 7:0 | | | | | | | CH1 | CH0 |
| | | 15:8 | | | | | | | DACRDY1 | DACRDY0 |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | DACC_CDR0 | 7:0 | DATA0[7:0] | | | | | | | |
| | | 15:8 | DATA0[15:8] | | | | | | | |
| | | 23:16 | DATA1[7:0] | | | | | | | |
| | | 31:24 | DATA1[15:8] | | | | | | | |
| 0x20 | DACC_CDR1 | 7:0 | DATA0[7:0] | | | | | | | |
| | | 15:8 | DATA0[15:8] | | | | | | | |
| | | 23:16 | DATA1[7:0] | | | | | | | |
| | | 31:24 | DATA1[15:8] | | | | | | | |
| 0x24 | DACC_IER | 7:0 | | | EOC1 | EOC0 | | | TXRDY1 | TXRDY0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | DACC_IDR | 7:0 | | | EOC1 | EOC0 | | | TXRDY1 | TXRDY0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | DACC_IMR | 7:0 | | | EOC1 | EOC0 | | | TXRDY1 | TXRDY0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | DACC_ISR | 7:0 | | | EOC1 | EOC0 | | | TXRDY1 | TXRDY0 |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x34 ... 0x93 | Reserved | | | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x94 | DACC_ACR | 7:0 | | | | | IBCTLCH1[1:0] | | IBCTLCH0[1:0] | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | DACC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | DACC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | WPVSRC[7:0] | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

### 53.7.1 DACC Control Register

**Name:** DACC_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the DACC simulating a hardware reset. |

### 53.7.2 DACC Mode Register

**Name:** DACC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the DACC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | PRESCALER[3:0] | | | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DIFF | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | ZERO | WORD | | | MAXS1 | MAXS0 |
| Access | | | R/W | R/W | | | R/W | R/W |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 27:24 – PRESCALER[3:0]** Peripheral Clock to DAC Clock Ratio
This field defines the division ratio between the peripheral clock and the DAC clock as per the following formula:

$$\text{PRESCALER} = \left(\frac{f_{\text{peripheral clock}}}{f_{\text{DAC}}}\right) - 2$$

**Bit 23 – DIFF** Differential Mode

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | DAC0 and DAC1 are single-ended outputs. |
| 1 | ENABLED | DACP and DACN are differential outputs. The differential level is configured by the channel 0 value. |

**Bit 5 – ZERO** Must always be written to 0.

**Bit 4 – WORD** Word Transfer Mode

| Value | Name | Description |
|---|---|---|
| 0 | DISABLED | One data to convert is written to the FIFO per access to DACC. |
| 1 | ENABLED | Two data to convert are written to the FIFO per access to DACC (reduces the number of requests to DMA and the number of system bus accesses). |

**Bits 0, 1 – MAXSx** Max Speed Mode for Channel x

| Value | Name | Description |
|---|---|---|
| 0 | TRIG_EVENT | Trigger mode or Free-running mode enabled. (See TRGENx.DACC_TRIGR.) |
| 1 | MAXIMUM | Max speed mode enabled. |

### 53.7.3 DACC Trigger Register

**Name:** DACC_TRIGR
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the DACC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | OSR1[2:0] | | | | OSR0[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | TRGSEL1[2:0] | | |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TRGSEL0[2:0] | | | | | TRGEN1 | TRGEN0 |
| Access | | R/W | R/W | R/W | | | R/W | R/W |
| Reset | | 0 | 0 | 0 | | | 0 | 0 |

**Bits 16:18, 20:22 – OSRx** Oversampling Ratio of Channel x

| Value | Name | Description |
|---|---|---|
| 0 | OSR_1 | OSR = 1 |
| 1 | OSR_2 | OSR = 2 |
| 2 | OSR_4 | OSR = 4 |
| 3 | OSR_8 | OSR = 8 |
| 4 | OSR_16 | OSR = 16 |
| 5 | OSR_32 | OSR = 32 |

**Bits 4:6, 8:10 – TRGSELx** Trigger Selection of Channel x

| Value | Name | Description |
|---|---|---|
| 0 | TRGSEL0 | DATRG |
| 1 | TRGSEL1 | TC0 output |
| 2 | TRGSEL2 | TC1 output |
| 3 | TRGSEL3 | TC2 output |
| 4 | TRGSEL4 | PWM0 Event 0 |
| 5 | TRGSEL5 | PWM0 Event 1 |
| 6 | TRGSEL6 | PWM1 Event 0 |
| 7 | TRGSEL7 | PWM 1 Event 1 |

**Bits 0, 1 – TRGENx** Trigger Enable of Channel x

| Value | Name | Description |
|---|---|---|
| 0 | DIS | Trigger mode disabled. DACC is in Free-running mode or Max speed mode. |
| 1 | EN | Trigger mode enabled. |

### 53.7.4 DACC Channel Enable Register

**Name:** DACC_CHER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the DACC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CH1 | CH0 |
| Access | | | | | | | W | W |
| Reset | | | | | | | 0 | – |

**Bits 0, 1 – CHx** Channel x Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding channel. |

### 53.7.5 DACC Channel Disable Register

**Name:** DACC_CHDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

This register can only be written if the WPEN bit is cleared in the DACC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CH1 | CH0 |
| Access | | | | | | | W | W |
| Reset | | | | | | | 0 | – |

**Bits 0, 1 – CHx** Channel x Disable

**WARNING** If the corresponding channel is disabled during a conversion or if it is disabled then re-enabled during a conversion, its associated analog value and its corresponding EOC flags in DACC_ISR are unpredictable.

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding channel. |

### 53.7.6 DACC Channel Status Register

**Name:** DACC_CHSR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | DACRDY1 | DACRDY0 |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | CH1 | CH0 |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bits 8, 9 – DACRDYx** DAC Ready Flag

| Value | Description |
|---|---|
| 0 | The DACx is not yet ready to receive data. |
| 1 | The DACx is ready to receive data. |

**Bits 0, 1 – CHx** Channel x Status

| Value | Description |
|---|---|
| 0 | Corresponding channel is disabled. |
| 1 | Corresponding channel is enabled. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 53.7.7 DACC Conversion Data Register

**Name:** DACC_CDRx
**Offset:** 0x1C + x*0x04 [x=0..1]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DATA1[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DATA1[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DATA0[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA0[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:16 – DATA1[15:0]** Data to Convert for channel x
If DACC_MR.WORD is set, DATA1 is written to the FIFO of channel x after DATA0.

**Bits 15:0 – DATA0[15:0]** Data to Convert for channel x
DATA0 is written to the FIFO of channel x.

### 53.7.8 DACC Interrupt Enable Register

**Name:** DACC_IER
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EOC1 | EOC0 | | | TXRDY1 | TXRDY0 |
| Access | | | W | W | | | W | W |
| Reset | | | 0 | – | | | 0 | – |

**Bits 4, 5 – EOCx** End of Conversion Interrupt Enable of channel x

**Bits 0, 1 – TXRDYx** Transmit Ready Interrupt Enable of channel x

### 53.7.9 DACC Interrupt Disable Register

**Name:** DACC_IDR
**Offset:** 0x28
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EOC1 | EOC0 | | | TXRDY1 | TXRDY0 |
| Access | | | W | W | | | W | W |
| Reset | | | 0 | – | | | 0 | – |

**Bits 4, 5 – EOCx** End of Conversion Interrupt Disable of channel x

**Bits 0, 1 – TXRDYx** Transmit Ready Interrupt Disable of channel x

### 53.7.10 DACC Interrupt Mask Register

**Name:** DACC_IMR
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EOC1 | EOC0 | | | TXRDY1 | TXRDY0 |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 4, 5 – EOCx** End of Conversion Interrupt Mask of channel x

**Bits 0, 1 – TXRDYx** Transmit Ready Interrupt Mask of channel x

### 53.7.11 DACC Interrupt Status Register

**Name:** DACC_ISR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | EOC1 | EOC0 | | | TXRDY1 | TXRDY0 |
| Access | | | R | R | | | R | R |
| Reset | | | 0 | 0 | | | 0 | 0 |

**Bits 4, 5 – EOCx** End of Conversion Interrupt Flag of channel x

| Value | Description |
|---|---|
| 0 | No conversion has been performed since the last read of DACC_ISR. |
| 1 | At least one conversion has been performed since the last read of DACC_ISR. |

**Bits 0, 1 – TXRDYx** Transmit Ready Interrupt Flag of channel x

| Value | Description |
|---|---|
| 0 | DACC is not ready to accept new conversion requests. |
| 1 | DACC is ready to accept new conversion requests. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 53.7.12 DACC Analog Current Register

**Name:** DACC_ACR
**Offset:** 0x94
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the DACC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | IBCTLCH1[1:0] | | IBCTLCH0[1:0] | |
| Access | | | | | R/W | R/W | R/W | R/W |
| Reset | | | | | 0 | 0 | 0 | 0 |

**Bits 0:1, 2:3 – IBCTLCHx** Analog Output Current Control
Allows to adapt the slew rate of the analog output. For more details, refer to the DAC Characteristics in the section “Electrical Characteristics” of this datasheet.

### 53.7.13 DACC Write Protection Mode Register

**Name:** DACC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protect Key

| Value | Name | Description |
|---|---|---|
| 0x444143 | PASSWD | Writing any other value in this field aborts the write operation of bit WPEN.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable
See Register Write Protection for list of write-protected registers.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x444143 (“DAC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x444143 (“DAC” in ASCII). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 53.7.14 DACC Write Protection Status Register

**Name:** DACC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPVSRC[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bits 15:8 – WPVSRC[7:0]** Write Protection Violation Source
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

**Bit 0 – WPVS** Write Protection Violation Status

| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of the DACC_WPSR. |
| 1 | A write protection violation has occurred since the last read of the DACC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC. |

# 54. Analog Comparator Controller (ACC)

## 54.1 Description

The Analog Comparator Controller (ACC) configures the analog comparator and generates an interrupt depending on user settings. The analog comparator embeds two 8-to-1 multiplexers that generate two internal inputs. These inputs are compared, resulting in a compare output. The hysteresis level, edge detection and polarity are configurable.

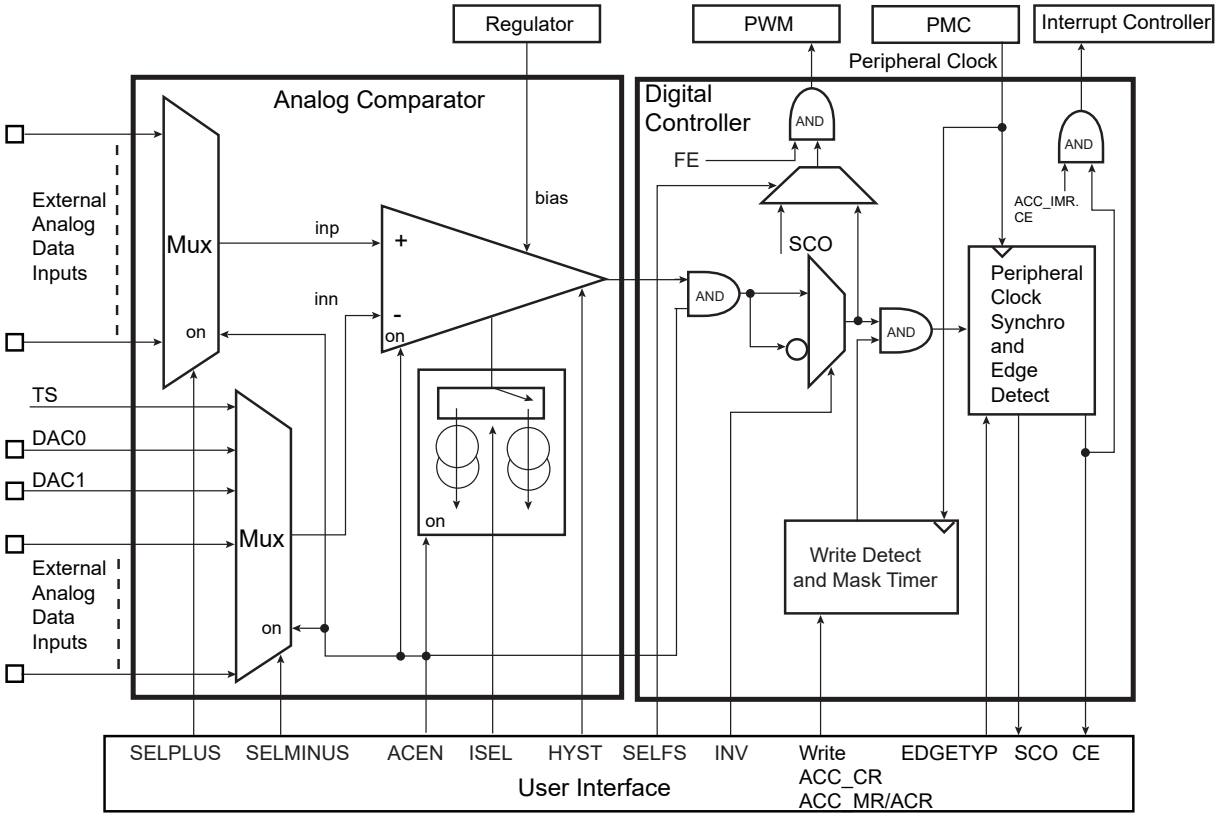
The ACC also generates a compare event which can be used by the Pulse Width Modulator (PWM).

## 54.2 Embedded Characteristics

- ANA_INPUTS User Analog Inputs Selectable for Comparison
- VOLT_REF Voltage References Selectable for Comparison: External Voltage Reference, DAC0, DAC1, Temperature Sensor (TS)
- Interrupt Generation
- Compare Event Fault Generation for PWM

## 54.3 Block Diagram

**Figure 54-1. Analog Comparator Controller Block Diagram**



## 54.4 Signal Description

**Table 54-1. ACC Signal Description**

| Pin Name | Description | Type |
|---|---|---|
| AFE0_AD[5:0] | External analog data inputs | Input |
| AFE1_AD[1:0] | | |
| TS | On-chip temperature sensor | Input |
| VREFP | AFE and DAC voltage reference | Input |
| DAC0, DAC1 | On-chip DAC outputs | Input |

## 54.5 Product Dependencies

### 54.5.1 I/O Lines

The analog input pins are multiplexed with digital functions (PIO) on the IO line. By writing the SELMINUS and SELPLUS fields in the ACC Mode Register (ACC_MR), the associated IO lines are set to Analog mode.

### 54.5.2 Power Management

The ACC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the ACC clock.

Note that the voltage regulator must be activated to use the analog comparator.

### 54.5.3 Interrupt Sources

The ACC has an interrupt line connected to the Interrupt Controller (IC). In order to handle interrupts, the Interrupt Controller must be programmed before configuring the ACC.

### 54.5.4 Fault Output

The ACC has the FAULT output connected to the FAULT input of PWM. See Fault Mode and the implementation of the PWM in the product.

## 54.6 Functional Description

### 54.6.1 Description

The Analog Comparator Controller (ACC) controls the analog comparator settings and performs postprocessing of the analog comparator output.

When the analog comparator settings are modified, the output of the analog cell may be invalid. The ACC masks the output for the invalid period.

A comparison flag is triggered by an event on the output of the analog comparator and an interrupt is generated. The event on the analog comparator output can be selected among falling edge, rising edge or any edge.

The ACC registers are listed in the Register Summary.

### 54.6.2 Analog Settings

The user can select the input hysteresis and configure two different options, characterized as follows:

- High-speed: shortest propagation delay/highest current consumption
- Low-power: longest propagation delay/lowest current consumption

Refer to ACC Analog Control Register.

### 54.6.3 Output Masking Period

As soon as the analog comparator settings change, the output is invalid for a duration depending on ISEL current.

A masking period is automatically triggered as soon as a write access is performed on the ACC_MR or ACC Analog Control Register (ACC_ACR) (regardless of the register data content).

When ISEL = 0, the mask period is 8 × $t_{peripheral\ clock}$.

When ISEL = 1, the mask period is 128 × $t_{peripheral\ clock}$.

The masking period is reported by reading a negative value (bit 31 set) on the ACC Interrupt Status Register (ACC_ISR).

### 54.6.4 Fault Mode

In Fault mode, a comparison match event is communicated by the ACC fault output which is directly and internally connected to a PWM fault input.

The source of the fault output can be configured as either a combinational value derived from the analog comparator output or as the peripheral clock resynchronized value. Refer to Analog Comparator Controller Block Diagram.

### 54.6.5 Register Write Protection

To prevent any single software error from corrupting ACC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the ACC Write Protection Mode Register (ACC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the ACC Write Protection Status Register (ACC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the ACC_WPSR register.

The following registers can be write-protected:

- ACC Mode Register
- ACC Analog Control Register

## 54.7 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ACC_CR | 7:0 | | | | | | | | SWRST |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | ACC_MR | 7:0 | | SELPLUS[2:0] | | | | SELMINUS[2:0] | | |
| | | 15:8 | | FE | SELFS | INV | | EDGETYP[1:0] | | ACEN |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x23 | Reserved | | | | | | | | | |
| 0x24 | ACC_IER | 7:0 | | | | | | | | CE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x28 | ACC_IDR | 7:0 | | | | | | | | CE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x2C | ACC_IMR | 7:0 | | | | | | | | CE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x30 | ACC_ISR | 7:0 | | | | | | | SCO | CE |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | MASK | | | | | | | |
| 0x34 ... 0x93 | Reserved | | | | | | | | | |
| 0x94 | ACC_ACR | 7:0 | | | | | | HYST[1:0] | | ISEL |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x98 ... 0xE3 | Reserved | | | | | | | | | |
| 0xE4 | ACC_WPMR | 7:0 | | | | | | | | WPEN |
| | | 15:8 | WPKEY[7:0] | | | | | | | |
| | | 23:16 | WPKEY[15:8] | | | | | | | |
| | | 31:24 | WPKEY[23:16] | | | | | | | |
| 0xE8 | ACC_WPSR | 7:0 | | | | | | | | WPVS |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 54.7.1 ACC Control Register

**Name:** ACC_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the module. |

### 54.7.2 ACC Mode Register

**Name:** ACC_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in the ACC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | FE | SELFS | INV | | EDGETYP[1:0] | | ACEN |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SELPLUS[2:0] | | | | SELMINUS[2:0] | | |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bit 14 – FE** Fault Enable
0 (DIS): The FAULT output is tied to 0.
1 (EN): The FAULT output is driven by the signal defined by SELFS.

**Bit 13 – SELFS** Selection Of Fault Source
0 (CE): The CE flag is used to drive the FAULT output.
1 (OUTPUT): The output of the analog comparator flag is used to drive the FAULT output.

**Bit 12 – INV** Invert Comparator Output
0 (DIS): Analog comparator output is directly processed.
1 (EN): Analog comparator output is inverted prior to being processed.

**Bits 10:9 – EDGETYP[1:0]** Edge Type

| Value | Name | Description |
|---|---|---|
| 0 | RISING | Only rising edge of comparator output |
| 1 | FALLING | Falling edge of comparator output |
| 2 | ANY | Any edge of comparator output |

**Bit 8 – ACEN** Analog Comparator Enable
0 (DIS): Analog comparator disabled.
1 (EN): Analog comparator enabled.

**Bits 6:4 – SELPLUS[2:0]** Selection For Plus Comparator Input
0..7: Selects the input to apply on analog comparator SELPLUS comparison input.

| Value | Name | Description |
|---|---|---|
| 0 | AFE0_AD0 | Select AFE0_AD0 |
| 1 | AFE0_AD1 | Select AFE0_AD1 |
| 2 | AFE0_AD2 | Select AFE0_AD2 |

© 2021 Microchip Technology Inc. and its subsidiaries

| Value | Name | Description |
|---|---|---|
| 3 | AFE0_AD3 | Select AFE0_AD3 |
| 4 | AFE0_AD4 | Select AFE0_AD4 |
| 5 | AFE0_AD5 | Select AFE0_AD5 |
| 6 | AFE1_AD6 | Select AFE1_AD0 |
| 7 | AFE1_AD7 | Select AFE1_AD1 |

**Bits 2:0 – SELMINUS[2:0]** Selection for Minus Comparator Input
0..7: Selects the input to apply on analog comparator SELMINUS comparison input.

| Value | Name | Description |
|---|---|---|
| 0 | TS | Select Temperature Sensor (TS) output voltage |
| 1 | VREFP | Select VREFP |
| 2 | DAC0 | Select DAC0 |
| 3 | DAC1 | Select DAC1 |
| 4 | AFE0_AD0 | Select AFE0_AD0 |
| 5 | AFE0_AD1 | Select AFE0_AD1 |
| 6 | AFE0_AD2 | Select AFE0_AD2 |
| 7 | AFE0_AD3 | Select AFE0_AD3 |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 54.7.3 ACC Interrupt Enable Register

**Name:** ACC_IER
**Offset:** 0x24
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – CE** Comparison Edge

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the interrupt when the selected edge (defined by EDGETYP) occurs. |

### 54.7.4 ACC Interrupt Disable Register

**Name:** ACC_IDR
**Offset:** 0x28
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – CE** Comparison Edge

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the interrupt when the selected edge (defined by EDGETYP) occurs. |

### 54.7.5 ACC Interrupt Mask Register

**Name:** ACC_IMR
**Offset:** 0x2C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | CE |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – CE** Comparison Edge

| Value | Description |
|---|---|
| 0 | The interrupt is disabled. |
| 1 | The interrupt is enabled. |

### 54.7.6 ACC Interrupt Status Register

**Name:** ACC_ISR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | MASK | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0 | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SCO | CE |
| Access | | | | | | | R | R |
| Reset | | | | | | | 0 | 0 |

**Bit 31 – MASK** Flag Mask

| Value | Description |
|---|---|
| 0 | The CE flag and SCO value are valid. |
| 1 | The CE flag and SCO value are invalid. |

**Bit 1 – SCO** Synchronized Comparator Output
Returns an image of the analog comparator output after being preprocessed (refer to ACC Block Diagram).
If INV = 0

- SCO = 0 if inn > inp
- SCO = 1 if inp > inn

If INV = 1

- SCO = 1 if inn > inp
- SCO = 0 if inp > inn

**Bit 0 – CE** Comparison Edge (cleared on read)

| Value | Description |
|---|---|
| 0 | No edge occurred (defined by EDGETYP) on analog comparator output since the last read of ACC_ISR. |
| 1 | A selected edge (defined by EDGETYP) on analog comparator output occurred since the last read of ACC_ISR. |

### 54.7.7 ACC Analog Control Register

**Name:** ACC_ACR
**Offset:** 0x94
**Reset:** 0
**Property:** Read/Write

This register can only be written if the WPEN bit is cleared in ACC Write Protection Mode Register.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | HYST[1:0] | | ISEL |
| Access | | | | | | R/W | R/W | R/W |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:1 – HYST[1:0]** Hysteresis Selection
Refer to the ACC characteristics in the section “Electrical Characteristics”.

| Value | Name | Description |
|---|---|---|
| 0 | NONE | No hysteresis |
| 1 | MEDIUM | Medium hysteresis |
| 2 | MEDIUM | Medium hysteresis |
| 3 | HIGH | High hysteresis |

**Bit 0 – ISEL** Current Selection
Refer to the ACC characteristics in the section “Electrical Characteristics”.
0 (LOPW): Low-power option.
1 (HISP): High-speed option.

### 54.7.8 ACC Write Protection Mode Register

**Name:** ACC_WPMR
**Offset:** 0xE4
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WPKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPEN |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bits 31:8 – WPKEY[23:0]** Write Protection Key

| Value | Name | Description |
|---|---|---|
| 0x414343 | PASSWD | Writing any other value in this field aborts the write operation of the WPEN bit.<br>Always reads as 0. |

**Bit 0 – WPEN** Write Protection Enable

Refer to Register Write Protection for the list of registers that can be write-protected.

| Value | Description |
|---|---|
| 0 | Disables the write protection if WPKEY corresponds to 0x414343 (“ACC” in ASCII). |
| 1 | Enables the write protection if WPKEY corresponds to 0x414343 (“ACC” in ASCII). |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 54.7.9 ACC Write Protection Status Register

**Name:** ACC_WPSR
**Offset:** 0xE8
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | WPVS |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – WPVS** Write Protection Violation Status

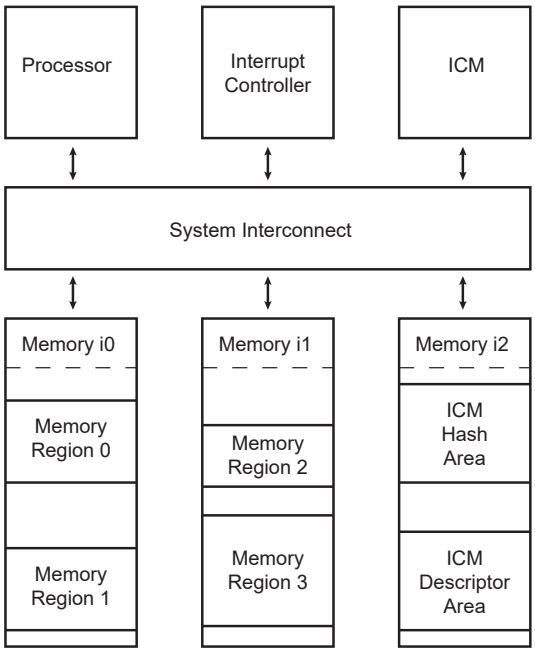
| Value | Description |
|---|---|
| 0 | No write protection violation has occurred since the last read of ACC_WPSR. |
| 1 | A write protection violation (WPEN = 1) has occurred since the last read of ACC_WPSR. |

# 55. Integrity Check Monitor (ICM)

## 55.1 Description

The Integrity Check Monitor (ICM) is a DMA controller that performs hash calculation over multiple memory regions through the use of transfer descriptors located in memory (ICM Descriptor Area). The Hash function is based on the Secure Hash Algorithm (SHA). The ICM integrates two modes of operation. The first one is used to hash a list of memory regions and save the digests to memory (ICM Hash Area). The second mode is an active monitoring of the memory. In that mode, the hash function is evaluated and compared to the digest located at a predefined memory address (ICM Hash Area). If a mismatch occurs, an interrupt is raised. See the figure below for an example of four-region monitoring. Hash and Descriptor areas are located in Memory instance i2, and the four regions are split in memory instances i0 and i1.

**Figure 55-1. Four-region Monitoring Example**




The ICM SHA engine is compliant with the American FIPS (Federal Information Processing Standard) Publication 180-2 specification.

The following terms are concise definitions of the ICM concepts used throughout this document:

- Region—a partition of instruction or data memory space
- Region Descriptor—a data structure stored in memory, defining region attributes
- Region Attributes—region start address, region size, region SHA engine processing mode, Write Back or Compare function mode
- Context Registers—a set of ICM non-memory-mapped, internal registers which are automatically loaded, containing the attributes of the region being processed
- Main List—a list of region descriptors. Each element associates the start address of a region with a set of attributes.
- Secondary List—a linked list defined on a per region basis that describes the memory layout of the region (when the region is non-contiguous)
- Hash Area—predefined memory space where the region hash results (digest) are stored

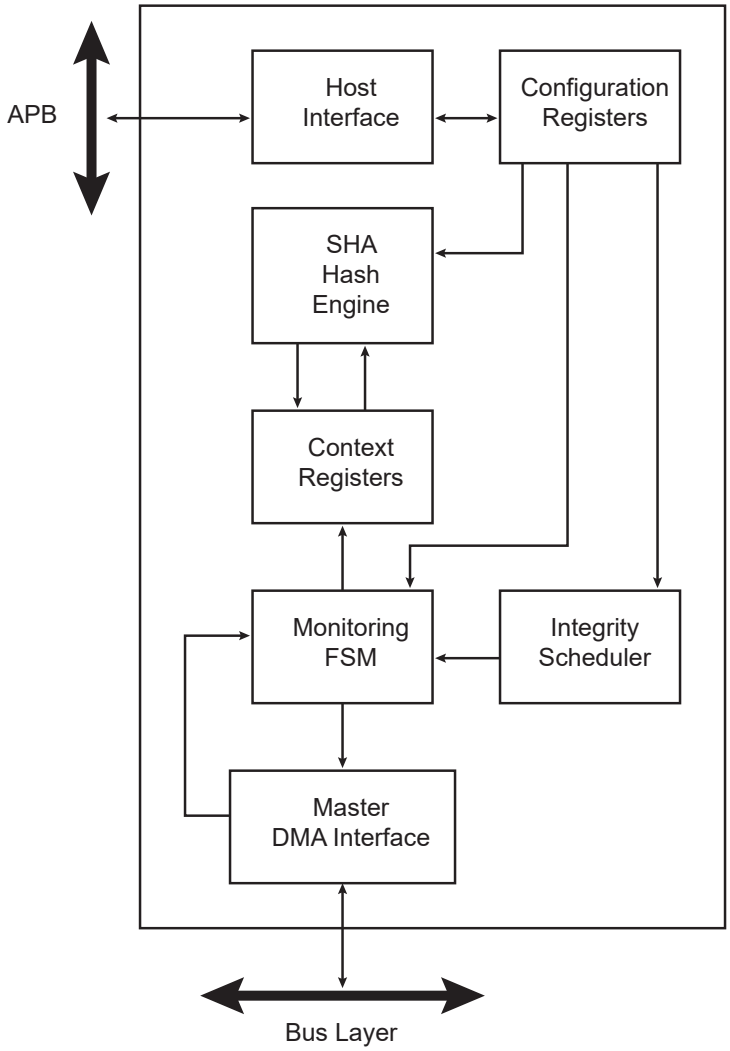
© 2021 Microchip Technology Inc. and its subsidiaries

## 55.2 Embedded Characteristics

- DMA AHB Master Interface
- Supports Monitoring of up to 4 Non-Contiguous Memory Regions
- Supports Block Gathering Using Linked Lists
- Supports Secure Hash Algorithm (SHA1, SHA224, SHA256)
- Compliant with FIPS Publication 180-2
- Configurable Processing Period:
  - When SHA1 algorithm is processed, the runtime period is either 85 or 209 clock cycles.
  - When SHA256 or SHA224 algorithm is processed, the runtime period is either 72 or 194 clock cycles.
- Programmable Bus Burden

## 55.3 Block Diagram

**Figure 55-2. Integrity Check Monitor Block Diagram**



## 55.4 Product Dependencies

### 55.4.1 Power Management

The peripheral clock is not continuously provided to the ICM. The programmer must first enable the ICM clock in the Power Management Controller (PMC) before using the ICM.

### 55.4.2 Interrupt Sources

The ICM interface has an interrupt line connected to the Interrupt Controller.

Handling the ICM interrupt requires programming the interrupt controller before configuring the ICM.

## 55.5 Functional Description

### 55.5.1 Overview

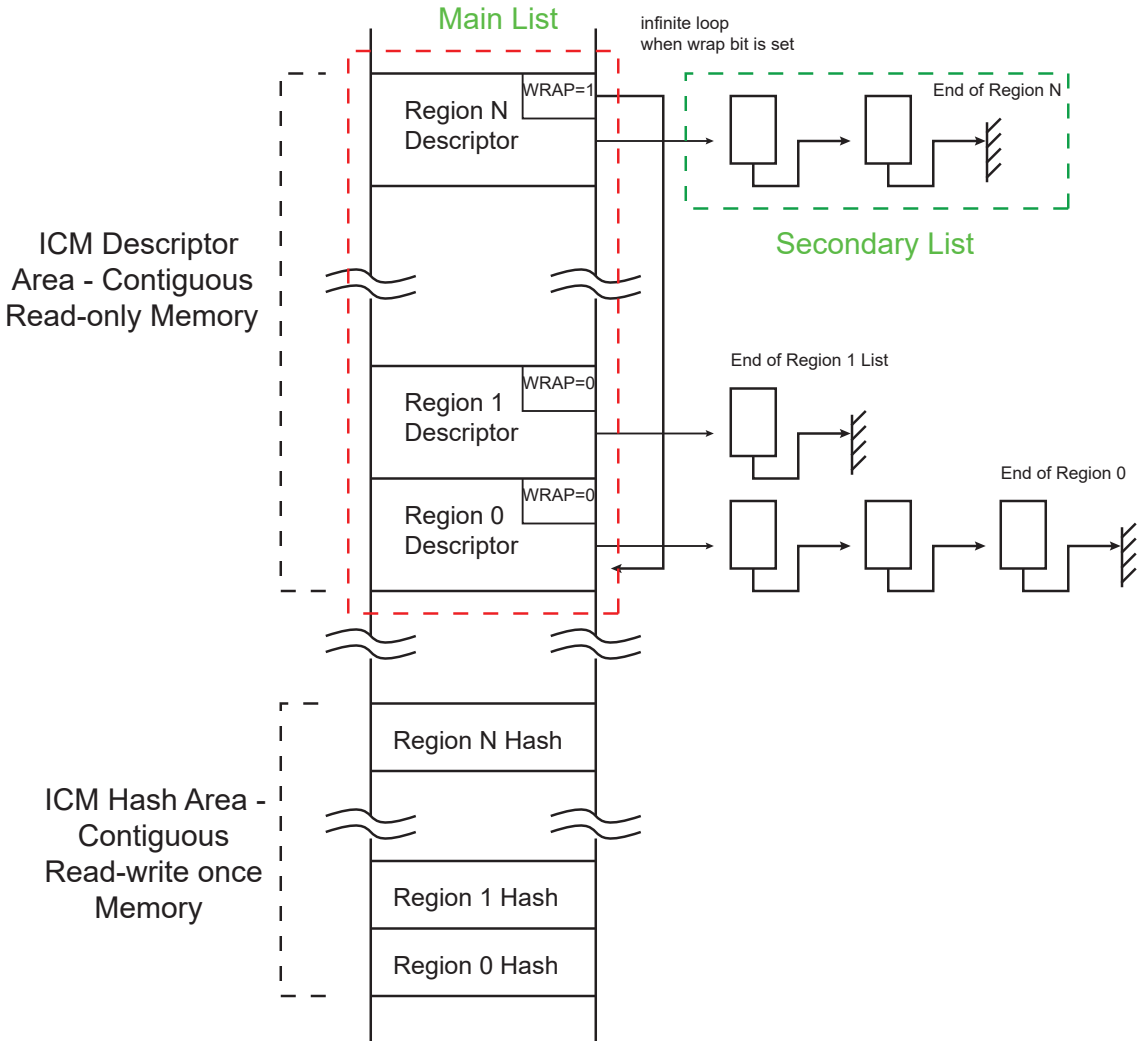
The Integrity Check Monitor (ICM) is a DMA controller that performs SHA-based memory hashing over memory regions. As shown in figure Integrity Check Monitor Block Diagram, it integrates a DMA interface, a Monitoring Finite State Machine (FSM), an integrity scheduler, a set of context registers, a SHA engine, an interface for configuration and status registers.

The ICM integrates a Secure Hash Algorithm engine (SHA). This engine requires a message padded according to FIPS180-2 specification when used as a SHA calculation unit only. Otherwise, if the ICM is used as integrated check for memory content, the padding is not mandatory. The SHA module produces an N-bit message digest each time a block is read and a processing period ends. N is 160 for SHA1, 224 for SHA224, 256 for SHA256.

When the ICM module is enabled, it sequentially retrieves a circular list of region descriptors from the memory (Main List described in figure ICM Region Descriptor and Hash Areas). Up to four regions may be monitored. Each region descriptor is composed of four words indicating the layout of the memory region (see figure Region Descriptor ). It also contains the hashing engine configuration on a per-region basis. As soon as the descriptor is loaded from the memory and context registers are updated with the data structure, the hashing operation starts. A programmable number of blocks (see TRSIZE field of the ICM_RCTRL structure member) is transferred from the memory to the SHA engine. When the desired number of blocks have been transferred, the digest is either moved to memory (Write Back function) or compared with a digest reference located in the system memory (Compare function). If a digest mismatch occurs, an interrupt is triggered if unmasked. The ICM module passes through the region descriptor list until the end of the list marked by an end of list marker (WRAP or EOM bit in ICM_RCFG structure member set to one). To continuously monitor the list of regions, the WRAP bit must be set to one in the last data structure and EOM must be cleared.

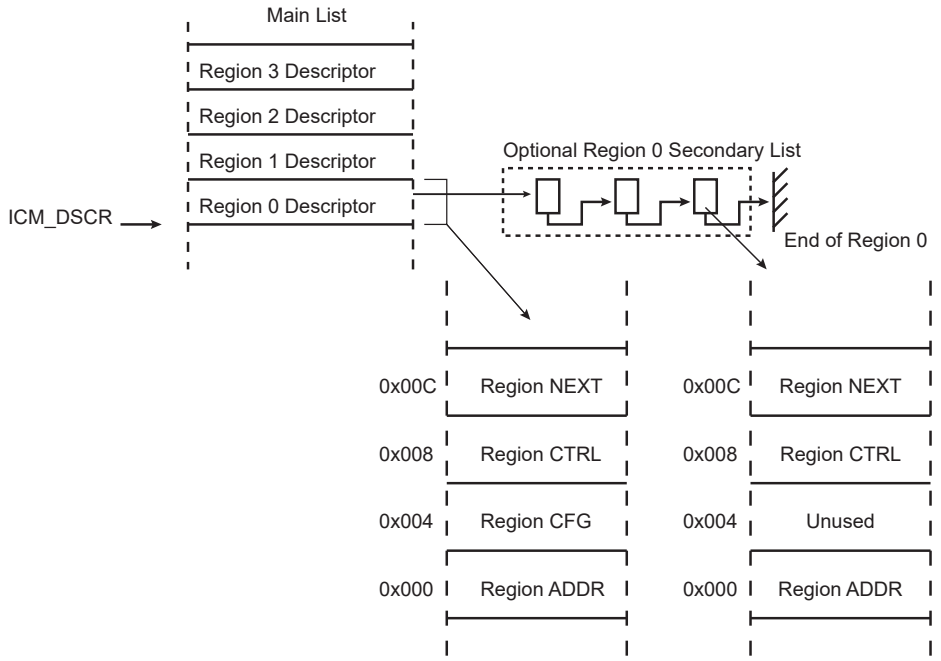
© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 55-3. ICM Region Descriptor and Hash Areas**



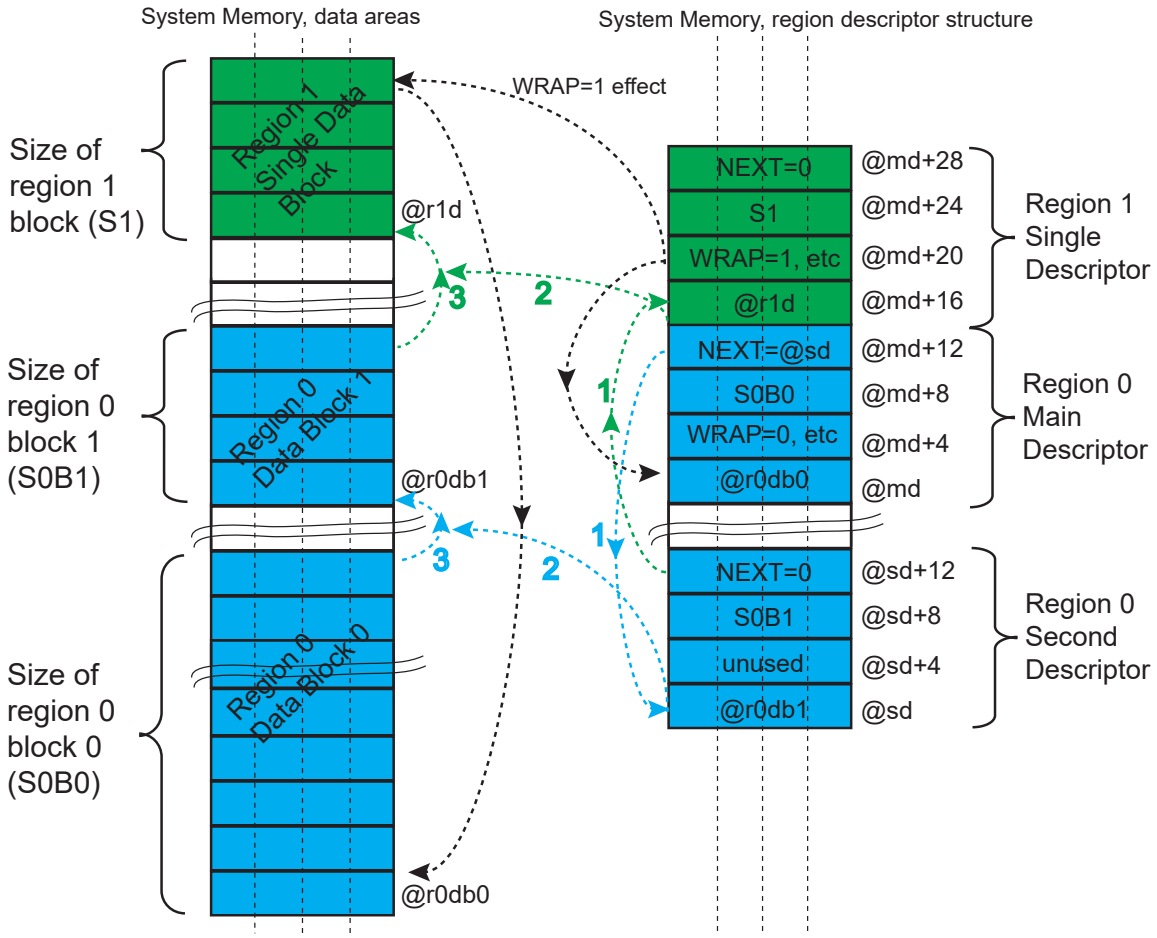

Each region descriptor supports gathering of data through the use of the Secondary List. Unlike the Main List, the Secondary List cannot modify the configuration attributes of the region. When the end of the Secondary List has been encountered, the ICM returns to the Main List. Memory integrity monitoring can be considered as a background service and the mandatory bandwidth shall be very limited. In order to limit the ICM memory bandwidth, use ICM_CFG.BBC to control the ICM memory load.

**Figure 55-4. Region Descriptor**




The figure below shows an example of the mandatory ICM settings required to monitor three memory data blocks of the system memory (defined as two regions) with one region being not contiguous (two separate areas) and one contiguous memory area. For each region, the SHA algorithm may be independently selected (different for each region). The wrap allows continuous monitoring.

**Figure 55-5. Example: Monitoring of 3 Memory Data Blocks (Defined as 2 Regions)**




### 55.5.2 ICM Region Descriptor Structure

The ICM Region Descriptor Area is a contiguous area of system memory that the controller and the processor can access. When the ICM is activated, the controller performs a descriptor fetch operation at *(ICM_DSCR) address. If the Main List contains more than one descriptor (i.e., more than one region is to be monitored), the fetch address is *(ICM_DSCR) + (RID<<4) where RID is the region identifier.

**Table 55-1. Region Descriptor Structure (Main List)**

| Offset | Structure Member | Name |
|---|---|---|
| ICM_DSCR+0x000+RID*(0x10) | ICM Region Start Address | ICM_RADDR |
| ICM_DSCR+0x004+RID*(0x10) | ICM Region Configuration | ICM_RCFG |
| ICM_DSCR+0x008+RID*(0x10) | ICM Region Control | ICM_RCTRL |
| ICM_DSCR+0x00C+RID*(0x10) | ICM Region Next Address | ICM_RNEXT |

#### 55.5.2.1 ICM Region Start Address Structure Member

**Name:** ICM_RADDR
**Property:** Read/Write

Register offset is calculated as ICM_DSCR+0x000+RID*(0x10).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | RADDR[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RADDR[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RADDR[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RADDR[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 31:0 – RADDR[31:0]** Region Start Address
This field indicates the first byte address of the region.

### 55.5.2.2 ICM Region Configuration Structure Member

**Name:** ICM_RCFG
**Property:** Read/Write

Register offset is calculated as ICM_DSCR+0x004+RID*(0x10).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | ALGO[2:0] | | | | PROCDLY | SUIEN | ECIEN |
| Access | | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | WCIEN | BEIEN | DMIEN | RHIEN | | EOM | WRAP | CDWBN |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 14:12 – ALGO[2:0]** SHA Algorithm
Values which are not listed in the table must be considered as “reserved”.

| Value | Name | Description |
|---|---|---|
| 0 | SHA1 | SHA1 algorithm processed |
| 1 | SHA256 | SHA256 algorithm processed |

**Bit 10 – PROCDLY** Processing Delay
When SHA1 algorithm is processed, the runtime period is either 85 or 209 clock cycles.
When SHA256 algorithm is processed, the runtime period is either 72 or 194 clock cycles.

| Value | Name | Description |
|---|---|---|
| 0 | SHORTEST | SHA processing runtime is the shortest one. |
| 1 | LONGEST | SHA processing runtime is the longest one. |

**Bit 9 – SUIEN** Monitoring Status Updated Condition Interrupt (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.RSU[i] flag is set when the corresponding descriptor is loaded from memory to ICM. |
| 1 | | The ICM_ISR.RSU[i] flag remains cleared even if the setting condition is met. |

**Bit 8 – ECIEN** End Bit Condition Interrupt (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.REC[i] flag is set when the descriptor with the EOM bit set is processed. |
| 1 | | The ICM_ISR.REC[i] flag remains cleared even if the setting condition is met. |

**Bit 7 – WCIEN** Wrap Condition Interrupt Disable (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.RWC[i] flag is set when the WRAP bit is set in a descriptor of the main list. |
| 1 | | ICM_ISR.RWC[i] flag remains cleared even if the setting condition is met. |

**Bit 6 – BEIEN** Bus Error Interrupt Disable (Default Enabled)

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Name | Description |
|---|---|---|
| 0 | | The flag is set when an error is reported on the system bus by the bus matrix. |
| 1 | | The flag remains cleared even if the setting condition is met. |

**Bit 5 – DMIEN** Digest Mismatch Interrupt Disable (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.RBE[i] flag is set when the hash value just calculated from the processed region differs from expected hash value. |
| 1 | | The ICM_ISR.RBE[i] flag remains cleared even if the setting condition is met. |

**Bit 4 – RHIEN** Region Hash Completed Interrupt Disable (Default Enabled)

| Value | Name | Description |
|---|---|---|
| 0 | | The ICM_ISR.RHC[i] flag is set when the field NEXT = 0 in a descriptor of the main or second list. |
| 1 | | The ICM_ISR.RHC[i] flag remains cleared even if the setting condition is met. |

**Bit 2 – EOM** End Of Monitoring

| Value | Name | Description |
|---|---|---|
| 0 | | The current descriptor does not terminate the monitoring. |
| 1 | | The current descriptor terminates the Main List. WRAP value has no effect. |

**Bit 1 – WRAP** Wrap Command

| Value | Name | Description |
|---|---|---|
| 0 | | The next region descriptor address loaded is the current region identifier descriptor address incremented by 0x10. |
| 1 | | The next region descriptor address loaded is ICM_DSCR. |

**Bit 0 – CDWBN** Compare Digest or Write Back Digest

| Value | Name | Description |
|---|---|---|
| 0 | | The digest is written to the Hash area. |
| 1 | | The digest value is compared to the digest stored in the Hash area. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 55.5.2.3 ICM Region Control Structure Member

**Name:** ICM_RCTRL
**Property:** Read/Write

Register offset is calculated as ICM_DSCR+0x008+RID*(0x10).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TRSIZE[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TRSIZE[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

**Bits 15:0 – TRSIZE[15:0]** Transfer Size for the Current Chunk of Data
ICM performs a transfer of (TRSIZE + 1) blocks of 512 bits.

#### 55.5.2.4 ICM Region Next Address Structure Member

**Name:** ICM_RNEXT
**Property:** Read/Write

Register offset is calculated as ICM_DSCR+0x00C+RID*(0x10).

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[29:22] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[21:14] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[13:6] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | NEXT[5:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | | |
| Reset | | | | | | | | |

**Bits 31:2 – NEXT[29:0]** Region Transfer Descriptor Next Address
When configured to 0, this field indicates that the current descriptor is the last descriptor of the Secondary List, otherwise it points at a new descriptor of the Secondary List.

### 55.5.3 ICM Hash Area

The ICM Hash Area is a contiguous area of system memory that the controller and the processor can access. The physical location is configured in the ICM hash area start address register. This address is a multiple of 128 bytes. If the CDWBN bit of the context register is cleared (i.e., Write Back activated), the ICM performs a digest write operation at the following starting location: *(ICM_HASH) + (RID<<5), where RID is the current region context identifier. If the CDWBN bit of the context register is set (i.e., Digest Comparison activated), the ICM performs a digest read operation at the same address.

#### 55.5.3.1 Message Digest Example

Considering the following 512-bit message (example given in FIPS 180-2):

“61626380000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000018”

The message is written to memory in a Little Endian (LE) system architecture.

**Table 55-2. 512 bits Message Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | |
|---|---|---|---|---|
| | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | 80 | 63 | 62 | 61 |
| 0x004–0x038 | 00 | 00 | 00 | 00 |
| 0x03C | 18 | 00 | 00 | 00 |

The digest is stored at the memory location pointed at by the ICM_HASH pointer with a Region Offset.

**Table 55-3. LE Resulting SHA-160 Message Digest Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | |
|---|---|---|---|---|
| | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | 36 | 3e | 99 | a9 |
| 0x004 | 6a | 81 | 06 | 47 |
| 0x008 | 71 | 25 | 3e | ba |
| 0x00C | 6c | c2 | 50 | 78 |
| 0x010 | 9d | d8 | d0 | 9c |

**Table 55-4. Resulting SHA-224 Message Digest Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | |
|---|---|---|---|---|
| | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | 22 | 7d | 09 | 23 |
| 0x004 | 22 | d8 | 05 | 34 |
| 0x008 | 77 | a4 | 42 | 86 |
| 0x00C | b3 | 55 | a2 | bd |
| 0x010 | e4 | bc | ad | 2a |
| 0x014 | f7 | b3 | a0 | bd |
| 0x018 | a7 | 9d | 6c | e3 |

**Table 55-5. Resulting SHA-256 Message Digest Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | |
|---|---|---|---|---|
| | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | bf | 16 | 78 | ba |
| 0x004 | ea | cf | 01 | 8f |
| 0x008 | de | 40 | 41 | 41 |
| 0x00C | 23 | 22 | ae | 5d |
| 0x010 | a3 | 61 | 03 | b0 |
| 0x014 | 9c | 7a | 17 | 96 |
| 0x018 | 61 | ff | 10 | b4 |
| 0x01C | ad | 15 | 00 | f2 |

Considering the following 1024-bit message (example given in FIPS 180-2):

“61626380000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000018”

The message is written to memory in a Little Endian (LE) system architecture.

**Table 55-6. 1024 bits Message Memory Mapping**

| Memory Address | Address Offset / Byte Lane | | | |
|---|---|---|---|---|
| | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 | 80 | 63 | 62 | 61 |
| 0x004–0x078 | 00 | 00 | 00 | 00 |
| 0x07C | 18 | 00 | 00 | 00 |

### 55.5.4 Using ICM as SHA Engine

The ICM can be configured to only calculate a SHA1, SHA224, SHA256 digest value.

#### 55.5.4.1 Settings for Simple SHA Calculation

The start address of the system memory containing the data to hash must be configured in the transfer descriptor of the DMA embedded in the ICM.

The transfer descriptor is a system memory area integer multiple of 4 x 32-bit words and the start address of the descriptor must be configured in ICM_DSCR (the start address must be aligned on 64-bytes; six LSB must be cleared). If the data to hash is already padded according to SHA standards, only a single descriptor is required, and ICM_RCFG.EOM must be written to 1. If the data to hash does not contain a padding area, it is possible to define the padding area in another system memory location, the ICM can be configured to automatically jump from a memory area to another one by configuring the descriptor register ICM_RNEXT with a value that differs from 0. Configuring ICM_RNEXT.NEXT with the start address of the padding area forces the ICM to concatenate both areas, thus providing the SHA result from the start address of the hash area configured in ICM_HASH.

Whether the system memory is configured as a single or multiple data block area, ICM_RCFG.CDWBN and ICM_RCFG.WRAP must be cleared. The bits WBDIS, EOMDIS, SLBDIS must be cleared in ICM_CFG.

ICM_RCTRL.RHIEN and ICM_RCTRL.ECIEN must be written to 1. The flag RHC[i], i being the region index, is set (if RHIEN is set) when the hash result is available at address defined in ICM_HASH. The flag REC[i], i being the region index, is set (if ECIEN is set) when the hash result is available at the address defined in ICM_HASH.

An interrupt is generated if the bit RHC[i] is written to 1 in the ICM_IER (if RHC[i] is set in ICM_RCTRL of region i) or if the bit REC[i] is written to 1 in the ICM_IER (if REC[i] is set in ICM_RCTRL of region i).

#### 55.5.4.2 Processing Period

The SHA engine processing period can be configured.

The short processing period allows to allocate bandwidth to the SHA module whereas the long processing period allocates more bandwidth on the system bus to other applications.

In SHA mode, the shortest processing period is 85 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA256 and SHA224 modes, the shortest processing period is 72 clock cycles + 2 clock cycles for start command synchronization. The longest period is 194 clock cycles + 2 clock cycles.

### 55.5.5 ICM Automatic Monitoring Mode

ICM_CFG.ASCD is used to activate the ICM Automatic Monitoring mode. When ICM_CFG.ASCD is set and bits CDWBN and EOM in ICM.RCFG equal 0, the ICM performs the following actions:

1. The ICM passes through the Main List once to calculate the message digest of the monitored area.
2. When WRAP = 1 in ICM_RCFG, the ICM begins monitoring. CDWBN in ICM_RCFG is now automatically set and EOM is cleared. These bits have no effect during the monitoring period that ends when EOM is set.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 55.5.6 Programming the ICM

**Table 55-7. Region Attributes**

| Transfer Type | | Main List | ICM_RCFG | | | ICM_RNEXT | Comments |
|---|---|---|---|---|---|---|---|
| | | | CDWBN | WRAP | EOM | NEXT | |
| Single Region | Contiguous list of blocks<br>Digest written to memory<br>Monitoring disabled | 1 item | 0 | 0 | 1 | 0 | The Main List contains only one descriptor. The Secondary List is empty for that descriptor. The digest is computed and saved to memory. |
| | Non-contiguous list of blocks<br>Digest written to memory<br>Monitoring disabled | 1 item | 0 | 0 | 1 | Secondary List address of the current region identifier | The Main List contains only one descriptor. The Secondary List describes the layout of the non-contiguous region. |
| | Contiguous list of blocks<br>Digest comparison enabled<br>Monitoring enabled | 1 item | 1 | 1 | 0 | 0 | When the hash computation is terminated, the digest is compared with the one saved in memory. |
| Multiple Regions | Contiguous list of blocks Digest written to memory<br>Monitoring disabled | More than one item | 0 | 0 | 1 for the last, 0 otherwise | 0 | ICM passes through the list once. |
| | Contiguous list of blocks<br>Digest comparison is enabled<br>Monitoring is enabled | More than one item | 1 | 1 for the last, 0 otherwise | 0 | 0 | ICM performs active monitoring of the regions. If a mismatch occurs, an interrupt is raised. |
| | Non-contiguous list of blocks<br>Digest is written to memory<br>Monitoring is disabled | More than one item | 0 | 0 | 1 | Secondary List address | ICM performs hashing and saves digests to the Hash area. |
| | Non-contiguous list of blocks<br>Digest comparison is enabled<br>Monitoring is enabled | More than one item | 1 | 1 | 0 | Secondary List address | ICM performs data gathering on a per region basis. |

© 2021 Microchip Technology Inc. and its subsidiaries

### 55.5.7 Security Features

When an undefined register access occurs, the URAD bit in the Interrupt Status Register (ICM_ISR) is set if unmasked. Its source is then reported in the Undefined Access Status Register (ICM_UASR). Only the first undefined register access is available through the ICM_UASR.URAT field.

Several kinds of unspecified register accesses can occur:

- Unspecified structure member set to one detected when the descriptor is loaded
- Configuration register (ICM_CFG) modified during active monitoring
- Descriptor register (ICM_DSCR) modified during active monitoring
- Hash register (ICM_HASH) modified during active monitoring
- Write-only register read access

The URAD bit and the URAT field can only be reset by writing a 1 to the ICM_CTRL.SWRST bit.

© 2021 Microchip Technology Inc.
and its subsidiaries

## 55.6 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | ICM_CFG | 7:0 | BBC[3:0] | | | | | SLBDIS | EOMDIS | WBDIS |
| | | 15:8 | UALGO[2:0] | | | UIHASH | | | DUALBUFF | ASCD |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | ICM_CTRL | 7:0 | REHASH[3:0] | | | | | SWRST | DISABLE | ENABLE |
| | | 15:8 | RMEN[3:0] | | | | RMDIS[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x08 | ICM_SR | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | RMDIS[3:0] | | | | RAWRMDIS[3:0] | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x0C ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | ICM_IER | 7:0 | RDM[3:0] | | | | RHC[3:0] | | | |
| | | 15:8 | RWC[3:0] | | | | RBE[3:0] | | | |
| | | 23:16 | RSU[3:0] | | | | REC[3:0] | | | |
| | | 31:24 | | | | | | | | URAD |
| 0x14 | ICM_IDR | 7:0 | RDM[3:0] | | | | RHC[3:0] | | | |
| | | 15:8 | RWC[3:0] | | | | RBE[3:0] | | | |
| | | 23:16 | RSU[3:0] | | | | REC[3:0] | | | |
| | | 31:24 | | | | | | | | URAD |
| 0x18 | ICM_IMR | 7:0 | RDM[3:0] | | | | RHC[3:0] | | | |
| | | 15:8 | RWC[3:0] | | | | RBE[3:0] | | | |
| | | 23:16 | RSU[3:0] | | | | REC[3:0] | | | |
| | | 31:24 | | | | | | | | URAD |
| 0x1C | ICM_ISR | 7:0 | RDM[3:0] | | | | RHC[3:0] | | | |
| | | 15:8 | RWC[3:0] | | | | RBE[3:0] | | | |
| | | 23:16 | RSU[3:0] | | | | REC[3:0] | | | |
| | | 31:24 | | | | | | | | URAD |
| 0x20 | ICM_UASR | 7:0 | | | | | | URAT[2:0] | | |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x24 ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | ICM_DSCR | 7:0 | DASA[1:0] | | | | | | | |
| | | 15:8 | DASA[9:2] | | | | | | | |
| | | 23:16 | DASA[17:10] | | | | | | | |
| | | 31:24 | DASA[25:18] | | | | | | | |
| 0x34 | ICM_HASH | 7:0 | HASA[0] | | | | | | | |
| | | 15:8 | HASA[8:1] | | | | | | | |
| | | 23:16 | HASA[16:9] | | | | | | | |
| | | 31:24 | HASA[24:17] | | | | | | | |
| 0x38 | ICM_UIHVAL0 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x3C | ICM_UIHVAL1 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x40 | ICM_UIHVAL2 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x44 | ICM_UIHVAL3 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x48 | ICM_UIHVAL4 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x4C | ICM_UIHVAL5 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x50 | ICM_UIHVAL6 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |
| 0x54 | ICM_UIHVAL7 | 7:0 | VAL[7:0] | | | | | | | |
| | | 15:8 | VAL[15:8] | | | | | | | |
| | | 23:16 | VAL[23:16] | | | | | | | |
| | | 31:24 | VAL[31:24] | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 55.6.1 ICM Configuration Register

**Name:** ICM_CFG
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | UALGO[2:0] | | | UIHASH | | | DUALBUFF | ASCD |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BBC[3:0] | | | | | SLBDIS | EOMDIS | WBDIS |
| Access | R/W | R/W | R/W | R/W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |

**Bits 15:13 – UALGO[2:0]** User SHA Algorithm

| Value | Name | Description |
|---|---|---|
| 0 | SHA1 | SHA1 algorithm processed |
| 1 | SHA256 | SHA256 algorithm processed |
| 4 | SHA224 | SHA224 algorithm processed |

**Bit 12 – UIHASH** User Initial Hash Value

| Value | Description |
|---|---|
| 0 | The secure hash standard provides the initial hash value. |
| 1 | The initial hash value is programmable. Field UALGO provides the SHA algorithm. The ALGO field of the ICM_RCFG structure member has no effect. |

**Bit 9 – DUALBUFF** Dual Input Buffer

| Value | Description |
|---|---|
| 0 | Dual Input Buffer mode is disabled. |
| 1 | Dual Input Buffer mode is enabled (better performances, higher bandwidth required on system bus). |

**Bit 8 – ASCD** Automatic Switch To Compare Digest

| Value | Description |
|---|---|
| 0 | Automatic monitoring mode is disabled. |
| 1 | The ICM passes through the Main List once to calculate the message digest of the monitored area. When WRAP = 1 in ICM_RCFG, the ICM begins monitoring. |

**Bits 7:4 – BBC[3:0]** Bus Burden Control
This field is used to control the burden of the ICM system bus. The number of system clock cycles between the end of the current processing and the next block transfer is set to $2^{BBC}$. Up to 32,768 cycles can be inserted.

**Bit 2 – SLBDIS** Secondary List Branching Disable

| Value | Description |
|---|---|
| 0 | Branching to the Secondary List is permitted. |
| 1 | Branching to the Secondary List is forbidden. The NEXT field of the ICM_RNEXT structure member has no effect and is always considered as zero. |

**Bit 1 – EOMDIS** End of Monitoring Disable

| Value | Description |
|---|---|
| 0 | End of Monitoring is permitted. |
| 1 | End of Monitoring is forbidden. The EOM bit of the ICM_RCFG structure member has no effect. |

**Bit 0 – WBDIS** Write Back Disable
When ASCD is set, WBDIS has no effect.

| Value | Description |
|---|---|
| 0 | Write Back operations are permitted. |
| 1 | Write Back operations are forbidden. Context register CDWBN bit is internally set to one and cannot be modified by a linked list element. ICM_RCFG.CDWBN has no effect. |

### 55.6.2 ICM Control Register

**Name:** ICM_CTRL
**Offset:** 0x04
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RMEN[3:0] | | | | RMDIS[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | REHASH[3:0] | | | | | SWRST | DISABLE | ENABLE |
| Access | W | W | W | W | | W | W | W |
| Reset | 0 | 0 | 0 | – | | – | – | – |

**Bits 15:12 – RMEN[3:0]** Region Monitoring Enable
Monitoring is activated by default.

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | When bit RMEN[i] is set to one, the monitoring of region with identifier i is activated. |

**Bits 11:8 – RMDIS[3:0]** Region Monitoring Disable

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | When bit RMDIS[i] is set to one, the monitoring of region with identifier i is disabled. |

**Bits 7:4 – REHASH[3:0]** Recompute Internal Hash

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | When REHASH[i] is set to one, Region i digest is re-computed. This bit is only available when region monitoring is disabled. |

**Bit 2 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | Resets the ICM. |

**Bit 1 – DISABLE** ICM Disable Register

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | The ICM is disabled. If a region is active, this region is terminated. |

**Bit 0 – ENABLE** ICM Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 0 | No effect |
| 1 | When set to one, the ICM is activated. |

### 55.6.3 ICM Status Register

**Name:** ICM_SR
**Offset:** 0x08
**Reset:** –
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RMDIS[3:0] | | | | RAWRMDIS[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | R |
| Reset | | | | | | | | – |

**Bits 15:12 – RMDIS[3:0]** Region Monitoring Disabled Status

| Value | Description |
|---|---|
| 0 | Region i is being monitored (occurs after integrity check value has been calculated and written to Hash area). |
| 1 | Region i monitoring is not being monitored. |

**Bits 11:8 – RAWRMDIS[3:0]** Region Monitoring Disabled Raw Status

| Value | Description |
|---|---|
| 0 | Region i monitoring has been activated by writing a 1 in RMEN[i] of ICM_CTRL. |
| 1 | Region i monitoring has been deactivated by writing a 1 in RMDIS[i] of ICM_CTRL. |

**Bit 0 – ENABLE** ICM Enable Register

| Value | Description |
|---|---|
| 0 | ICM is disabled. |
| 1 | ICM is activated. |

### 55.6.4 ICM Interrupt Enable Register

**Name:** ICM_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSU[3:0] | | | | REC[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWC[3:0] | | | | RBE[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDM[3:0] | | | | RHC[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

**Bit 24 – URAD** Undefined Register Access Detection Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | The Undefined Register Access interrupt is enabled. |

**Bits 23:20 – RSU[3:0]** Region Status Updated Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RSU[i] is set to one, the region i Status Updated interrupt is enabled. |

**Bits 19:16 – REC[3:0]** Region End bit Condition Detected Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When REC[i] is set to one, the region i End bit Condition interrupt is enabled. |

**Bits 15:12 – RWC[3:0]** Region Wrap Condition detected Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RWC[i] is set to one, the Region i Wrap Condition interrupt is enabled. |

**Bits 11:8 – RBE[3:0]** Region Bus Error Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RBE[i] is set to one, the Region i Bus Error interrupt is enabled. |

**Bits 7:4 – RDM[3:0]** Region Digest Mismatch Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | When RDM[i] is set to one, the Region i Digest Mismatch interrupt is enabled. |

**Bits 3:0 – RHC[3:0]** Region Hash Completed Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RHC[i] is set to one, the Region i Hash Completed interrupt is enabled. |

### 55.6.5 ICM Interrupt Disable Register

**Name:** ICM_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSU[3:0] | | | | REC[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWC[3:0] | | | | RBE[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDM[3:0] | | | | RHC[3:0] | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | – | 0 | 0 | 0 | – |

**Bit 24 – URAD** Undefined Register Access Detection Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Undefined Register Access Detection interrupt is disabled. |

**Bits 23:20 – RSU[3:0]** Region Status Updated Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RSU[i] is set to one, the region i Status Updated interrupt is disabled. |

**Bits 19:16 – REC[3:0]** Region End bit Condition detected Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When REC[i] is set to one, the region i End bit Condition interrupt is disabled. |

**Bits 15:12 – RWC[3:0]** Region Wrap Condition Detected Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RWC[i] is set to one, the Region i Wrap Condition interrupt is disabled. |

**Bits 11:8 – RBE[3:0]** Region Bus Error Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RBE[i] is set to one, the Region i Bus Error interrupt is disabled. |

**Bits 7:4 – RDM[3:0]** Region Digest Mismatch Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | When RDM[i] is set to one, the Region i Digest Mismatch interrupt is disabled. |

**Bits 3:0 – RHC[3:0]** Region Hash Completed Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | When RHC[i] is set to one, the Region i Hash Completed interrupt is disabled. |

### 55.6.6 ICM Interrupt Mask Register

**Name:** ICM_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSU[3:0] | | | | REC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWC[3:0] | | | | RBE[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDM[3:0] | | | | RHC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – URAD** Undefined Register Access Detection Interrupt Mask

| Value | Description |
|---|---|
| 0 | Interrupt is disabled |
| 1 | Interrupt is enabled. |

**Bits 23:20 – RSU[3:0]** Region Status Updated Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RSU[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RSU[i] is set to one, the interrupt is enabled for region i. |

**Bits 19:16 – REC[3:0]** Region End Bit Condition Detected Interrupt Mask

| Value | Description |
|---|---|
| 0 | When REC[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When REC[i] is set to one, the interrupt is enabled for region i. |

**Bits 15:12 – RWC[3:0]** Region Wrap Condition Detected Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RWC[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RWC[i] is set to one, the interrupt is enabled for region i. |

**Bits 11:8 – RBE[3:0]** Region Bus Error Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RBE[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RBE[i] is set to one, the interrupt is enabled for region i. |

**Bits 7:4 – RDM[3:0]** Region Digest Mismatch Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RDM[i] is set to zero, the interrupt is disabled for region i. |

© 2021 Microchip Technology Inc.
and its subsidiaries

| Value | Description |
|---|---|
| 1 | When RDM[i] is set to one, the interrupt is enabled for region i. |

**Bits 3:0 – RHC[3:0]** Region Hash Completed Interrupt Mask

| Value | Description |
|---|---|
| 0 | When RHC[i] is set to zero, the interrupt is disabled for region i. |
| 1 | When RHC[i] is set to one, the interrupt is enabled for region i. |

### 55.6.7 ICM Interrupt Status Register

**Name:** ICM_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | RSU[3:0] | | | | REC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | RWC[3:0] | | | | RBE[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RDM[3:0] | | | | RHC[3:0] | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 24 – URAD** Undefined Register Access Detection Status
The URAD bit is only reset by the SWRST bit in ICM_CTRL.
The URAT field in ICM_UASR indicates the unspecified access type.

| Value | Description |
|---|---|
| 0 | No undefined register access has been detected since the last SWRST. |
| 1 | At least one undefined register access has been detected since the last SWRST. |

**Bits 23:20 – RSU[3:0]** Region Status Updated Detected
When RSU[i] is set, it indicates that a region status updated condition has been detected.

**Bits 19:16 – REC[3:0]** Region End Bit Condition Detected
When REC[i] is set, it indicates that an end bit condition has been detected.

**Bits 15:12 – RWC[3:0]** Region Wrap Condition Detected
When RWC[i] is set, it indicates that a wrap condition has been detected.

**Bits 11:8 – RBE[3:0]** Region Bus Error
When RBE[i] is set, it indicates that a bus error has been detected while hashing memory region i.

**Bits 7:4 – RDM[3:0]** Region Digest Mismatch
When RDM[i] is set, it indicates that there is a digest comparison mismatch between the hash value of the region with identifier i and the reference value located in the Hash Area.

**Bits 3:0 – RHC[3:0]** Region Hash Completed
When RHC[i] is set, it indicates that the ICM has completed the region with identifier i.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 55.6.8 ICM Undefined Access Status Register

**Name:** ICM_UASR
**Offset:** 0x20
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | URAT[2:0] | | |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bits 2:0 – URAT[2:0]** Undefined Register Access Trace
Only the first Undefined Register Access Trace is available through the URAT field.
The URAT field is only reset by the SWRST bit in the ICM_CTRL register.

| Value | Name | Description |
|---|---|---|
| 0 | UNSPEC_STRUCT_MEMBER | Unspecified structure member set to one detected when the descriptor is loaded. |
| 1 | ICM_CFG_MODIFIED | ICM_CFG modified during active monitoring. |
| 2 | ICM_DSCR_MODIFIED | ICM_DSCR modified during active monitoring. |
| 3 | ICM_HASH_MODIFIED | ICM_HASH modified during active monitoring |
| 4 | READ_ACCESS | Write-only register read access |

### 55.6.9 ICM Descriptor Area Start Address Register

**Name:** ICM_DSCR
**Offset:** 0x30
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | DASA[25:18] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | DASA[17:10] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | DASA[9:2] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DASA[1:0] | | | | | | | |
| Access | R/W | R/W | | | | | | |
| Reset | 0 | 0 | | | | | | |

**Bits 31:6 – DASA[25:0]** Descriptor Area Start Address
The start address is a multiple of the total size of the data structure (64 bytes).

### 55.6.10 ICM Hash Area Start Address Register

**Name:** ICM_HASH
**Offset:** 0x34
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | HASA[24:17] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | HASA[16:9] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | HASA[8:1] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | HASA[0] | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0 | | | | | | | |

**Bits 31:7 – HASA[24:0]** Hash Area Start Address
This field points at the Hash memory location. The address must be a multiple of 128 bytes.

### 55.6.11 ICM User Initial Hash Value Register

**Name:** ICM_UIHVALx
**Offset:** 0x38 + x*0x04 [x=0..7]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | VAL[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | VAL[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | VAL[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | VAL[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – VAL[31:0]** Initial Hash Value
When ICM_CFG.UIHASH is set, the Initial Hash Value is user-programmable.
To meet the desired standard, use the following example values.
For ICM_UIHVAL0 field:

| Example | Comment |
|---|---|
| 0x67452301 | SHA1 algorithm |
| 0xC1059ED8 | SHA224 algorithm |
| 0x6A09E667 | SHA256 algorithm |

For ICM_UIHVAL1 field:

| Example | Comment |
|---|---|
| 0xEFCDAB89 | SHA1 algorithm |
| 0x367CD507 | SHA224 algorithm |
| 0xBB67AE85 | SHA256 algorithm |

For ICM_UIHVAL2 field:

| Example | Comment |
|---|---|
| 0x98BADCFE | SHA1 algorithm |
| 0x3070DD17 | SHA224 algorithm |
| 0x3C6EF372 | SHA256 algorithm |

For ICM_UIHVAL3 field:

| Example | Comment |
|---|---|
| 0x10325476 | SHA1 algorithm |
| 0xF70E5939 | SHA224 algorithm |

| .....continued | |
|---|---|
| **Example** | **Comment** |
| 0xA54FF53A | SHA256 algorithm |

For ICM_UIHVAL4 field:

| Example | Comment |
|---|---|
| 0xC3D2E1F0 | SHA1 algorithm |
| 0xFFC00B31 | SHA224 algorithm |
| 0x510E527F | SHA256 algorithm |

For ICM_UIHVAL5 field:

| Example | Comment |
|---|---|
| 0x68581511 | SHA224 algorithm |
| 0x9B05688C | SHA256 algorithm |

For ICM_UIHVAL6 field:

| Example | Comment |
|---|---|
| 0x64F98FA7 | SHA224 algorithm |
| 0x1F83D9AB | SHA256 algorithm |

For ICM_UIHVAL7 field:

| Example | Comment |
|---|---|
| 0xBEFA4FA4 | SHA224 algorithm |
| 0x5BE0CD19 | SHA256 algorithm |

Example of Initial Value for SHA-1 Algorithm

| Register Address | Address Offset / Byte Lane | | | |
|---|---|---|---|---|
| | 0x3 / 31:24 | 0x2 / 23:16 | 0x1 / 15:8 | 0x0 / 7:0 |
| 0x000 ICM_UIHVAL0 | 01 | 23 | 45 | 67 |
| 0x004 ICM_UIHVAL1 | 89 | ab | cd | ef |
| 0x008 ICM_UIHVAL2 | fe | dc | ba | 98 |
| 0x00C ICM_UIHVAL3 | 76 | 54 | 32 | 10 |
| 0x010 ICM_UIHVAL4 | f0 | e1 | d2 | c3 |

© 2021 Microchip Technology Inc.
and its subsidiaries

# 56. True Random Number Generator (TRNG)

## 56.1 Description

The True Random Number Generator (TRNG) passes the American *NIST Special Publication 800-22 (A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications) and the Diehard Suite of Tests.*

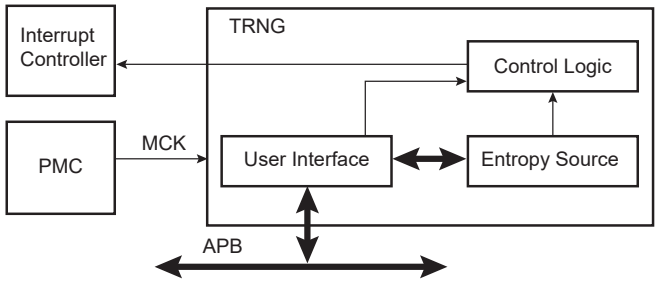
The TRNG may be used as an entropy source for seeding an NIST approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3.

## 56.2 Embedded Characteristics

- Passes *NIST Special Publication 800-22 Test Suite*
- Passes *Diehard Suite of Tests*
- May be Used as Entropy Source for seeding a NIST-approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3
- Provides a 32-bit Random Number Every 84 Clock Cycles

## 56.3 Block Diagram

**Figure 56-1. TRNG Block Diagram**




## 56.4 Product Dependencies

### 56.4.1 Power Management

The TRNG interface may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the TRNG user interface clock. The user interface clock is independent from any clock that may be used in the entropy source logic circuitry. The source of entropy can be enabled before enabling the user interface clock.

### 56.4.2 Interrupt Sources

The TRNG interface has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TRNG.

## 56.5 Functional Description

As soon as the TRNG is enabled in the Control register (TRNG_CR), the generator provides one 32-bit random value every 84 clock cycles.

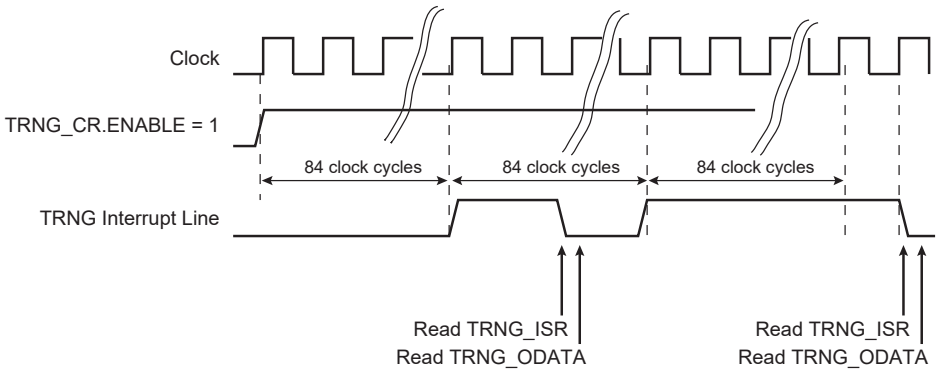
The TRNG interrupt line can be enabled in the Interrupt Enable register (TRNG_IER), and disabled in the Interrupt Disable register (TRNG_IDR). This interrupt is set when a new random value is available and is cleared when the

© 2021 Microchip Technology Inc. and its subsidiaries

Status register (TRNG_ISR) is read. The flag TRNG_ISR.DATRDY is set when the random data is ready to be read out on the 32-bit Output Data register (TRNG_ODATA).

The normal mode of operation checks that the flag in TRNG_ISR equals ‘1’ before reading TRNG_ODATA when a 32-bit random value is required by the software application.

**Figure 56-2. TRNG Data Generation Sequence**



## 56.6 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | TRNG_CR | 7:0 | | | | | | | | ENABLE |
| | | 15:8 | WAKEY[7:0] | | | | | | | |
| | | 23:16 | WAKEY[15:8] | | | | | | | |
| | | 31:24 | WAKEY[23:16] | | | | | | | |
| 0x04 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | TRNG_IER | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x14 | TRNG_IDR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x18 | TRNG_IMR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x1C | TRNG_ISR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x20 ... 0x4F | Reserved | | | | | | | | | |
| 0x50 | TRNG_ODATA | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |

### 56.6.1 TRNG Control Register

**Name:** TRNG_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | WAKEY[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | WAKEY[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | WAKEY[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | ENABLE |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bits 31:8 – WAKEY[23:0]** Register Write Access Key

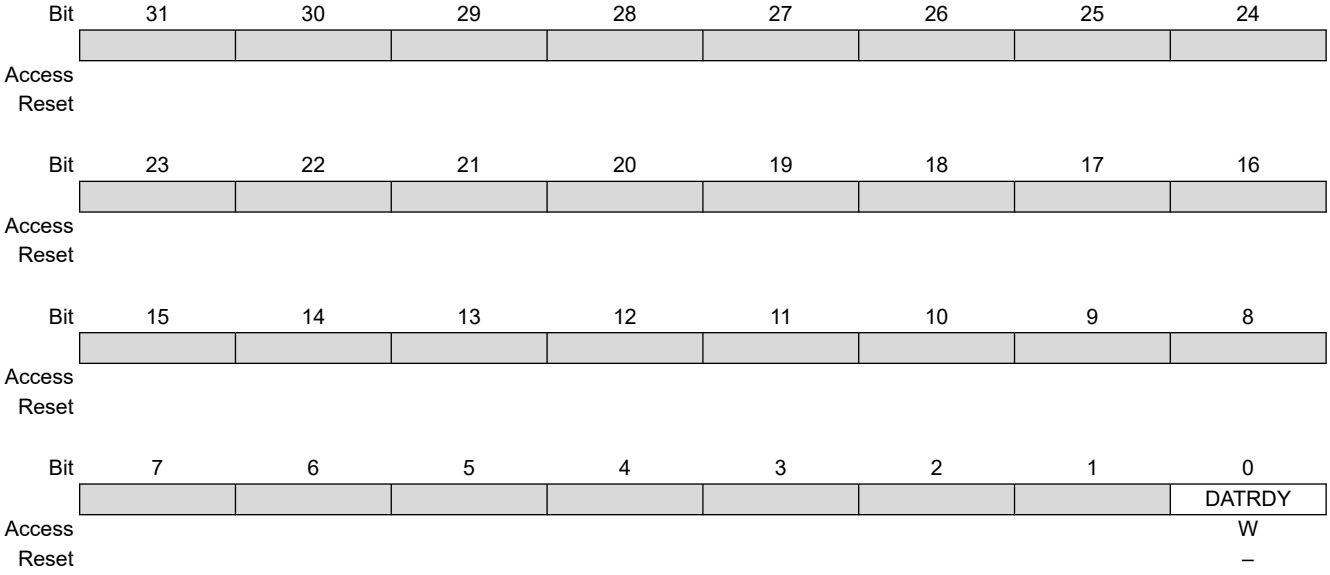
| Value | Name | Description |
|---|---|---|
| 0x524E47 | PASSWD | Writing any other value in this field aborts the write operation. |

**Bit 0 – ENABLE** Enables the TRNG to Provide Random Values

| Value | Description |
|---|---|
| 0 | Disables the TRNG. |
| 1 | Enables the TRNG if 0x524E47 (“RNG” in ASCII) is written in KEY field at the same time. |

### 56.6.2 TRNG Interrupt Enable Register

**Name:** TRNG_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only




**Bit 0 – DATRDY** Data Ready Interrupt Enable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Enables the corresponding interrupt. |

### 56.6.3 TRNG Interrupt Disable Register

**Name:** TRNG_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 0 – DATRDY** Data Ready Interrupt Disable

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Disables the corresponding interrupt. |

### 56.6.4 TRNG Interrupt Mask Register

**Name:** TRNG_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – DATRDY** Data Ready Interrupt Mask

| Value | Description |
|---|---|
| 0 | The corresponding interrupt is not enabled. |
| 1 | The corresponding interrupt is enabled. |

### 56.6.5 TRNG Interrupt Status Register

**Name:** TRNG_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 0 – DATRDY** Data Ready (cleared on read)

| Value | Description |
|---|---|
| 0 | Output data is not valid or TRNG is disabled. |
| 1 | New random value is completed since the last read of TRNG_ODATA. |

### 56.6.6 TRNG Output Data Register

**Name:** TRNG_ODATA
**Offset:** 0x50
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ODATA[31:0]** Output Data
The 32-bit Output Data register contains the 32-bit random data.

# 57. Advanced Encryption Standard (AES)

## 57.1 Description

The Advanced Encryption Standard (AES) is compliant with the American FIPS (Federal Information Processing Standard) Publication 197 specification.

The AES supports the following confidentiality modes of operation for symmetrical key block cipher algorithms: ECB, CBC, OFB, CFB, CTR), as specified in the NIST Special Publication 800-38A Recommendation, as well as Galois/Counter Mode (GCM) as specified in the NIST Special Publication 800-38D Recommendation. It is compatible with all these modes via DMA Controller channels, minimizing processor intervention for large buffer transfers.

The AES key loaded by the software.

The 128-bit/192-bit/256-bit AES key is stored in the AES Key Register made of four/six/eight 32-bit write-only AES Key Word registers (AES_KEYWR0–7).

The 128-bit input data and initialization vector (for some modes) are each stored in four 32-bit write-only AES Input Data registers (AES_IDATAR0–3) and AES Initialization Vector registers (AES_IVR0–3).

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data are ready to be read out on the four 32-bit AES Output Data registers (AES_ODATAR0–3) or through the DMA channels.

## 57.2 Embedded Characteristics

- Compliant with FIPS Publication 197, Advanced Encryption Standard (AES)
- 128-bit/192-bit/256-bit Cryptographic Key
- 10/12/14 Clock Cycles Encryption/Decryption Inherent Processing Time with a 128-bit/192-bit/256-bit Cryptographic Key
- Double Input Buffer Optimizes Runtime
- Support of the Modes of Operation Specified in the NIST Special Publication 800-38A and NIST Special Publication 800-38D:
  - Electronic Codebook (ECB)
  - Cipher Block Chaining (CBC) including CBC-MAC
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
  - Counter (CTR)
  - Galois/Counter Mode (GCM)
- 8, 16, 32, 64 and 128-bit Data Sizes Possible in CFB Mode
- Last Output Data Mode Allows Optimized Message Authentication Code (MAC) Generation
- Connection to DMA Optimizes Data Transfers for all Operating Modes

## 57.3 Product Dependencies

### 57.3.1 Power Management

The AES is clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the AES clock.

### 57.3.2 Interrupt Sources

The AES interface has an interrupt line connected to the Interrupt Controller.

Handling the AES interrupt requires programming the Interrupt Controller before configuring the AES.

© 2021 Microchip Technology Inc. and its subsidiaries

## 57.4 Functional Description

The Advanced Encryption Standard (AES) specifies a FIPS-approved cryptographic algorithm that can be used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information.

Encryption converts data to an unintelligible form called ciphertext. Decrypting the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in the AES Mode register (AES_MR) allows selection between the encryption and the decryption processes.

The AES is capable of using cryptographic keys of 128/192/256 bits to encrypt and decrypt data in blocks of 128 bits. This 128-bit/192-bit/256-bit key is defined in the user interface AES_KEYWRx register.

The input to the encryption processes of the CBC, CFB, and OFB modes includes, in addition to the plaintext, a 128-bit data block called the initialization vector (IV), which must be set in AES_IVRx. The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message. AES_IVRx are also used by the CTR mode to set the counter value.

### 57.4.1 AES Register Endianness

In ARM processor-based products, the system bus and processors manipulate data in little-endian form. The AES interface requires little-endian format words. However, in accordance with the protocol of the FIPS 197 specification, data is collected, processed and stored by the AES algorithm in big-endian form.

The following example illustrates how to configure the AES:

If the first 64 bits of a message (according to FIPS 197, i.e., big-endian format) to be processed is 0xcafedeca_01234567, then AES_IDATAR0 and AES_IDATAR1 registers must be written with the following pattern:

- AES_IDATAR0 = 0xcadefeca
- AES_IDATAR1 = 0x67452301

### 57.4.2 Operating Modes

The AES supports the following modes of operation:

- ECB: Electronic Codebook
- CBC: Cipher Block Chaining
  - CBC-MAC: Useful for CMAC hardware acceleration
- OFB: Output Feedback
- CFB: Cipher Feedback
  - CFB8 (CFB where the length of the data segment is 8 bits)
  - CFB16 (CFB where the length of the data segment is 16 bits)
  - CFB32 (CFB where the length of the data segment is 32 bits)
  - CFB64 (CFB where the length of the data segment is 64 bits)
  - CFB128 (CFB where the length of the data segment is 128 bits)
- CTR: Counter
- GCM: Galois/Counter Mode

The data preprocessing, data postprocessing and data chaining for the concerned modes are performed automatically. Refer to the NIST Special Publication 800-38A and NIST Special Publication 800-38D for more complete information.

Mode selection is done by configuring the OPMOD field in AES_MR.

In CFB mode, five data sizes are possible (8, 16, 32, 64 or 128 bits), configurable by means of AES_MR.CFBS.

In CTR mode, the size of the block counter embedded in the module is 16 bits. Therefore, there is a rollover after processing 1 Mbyte of data. If the file to be processed is greater than 1 Mbyte, this file must be split into fragments of 1 Mbyte or less for the first fragment if the initial value of the counter is greater than 0. Prior to loading the first fragment into AES_IDATARx, AES_IVRx must be fully programmed with the initial counter value. For any fragment, after the transfer is completed and prior to transferring the next fragment, AES_IVRx must be programmed with the appropriate counter value.

© 2021 Microchip Technology Inc.
and its subsidiaries

### 57.4.3 Last Output Data Mode (CBC_MAC)

This mode is used to generate cryptographic checksums on data (MAC) by means of cipher block chaining encryption algorithm (CBC-MAC algorithm for example).

The CMAC algorithm is a variant of CBC-MAC with post-processing requiring one-block encryption in ECB mode. Thus CBC-MAC is useful to accelerate CMAC.

After each end of encryption/decryption, the output data are available either on AES_ODATARx for Manual and Auto mode, or at the address specified in the receive buffer pointer for DMA mode (see the table "Last Output Data Mode Behavior versus Start Modes").

AES_MR.LOD allows retrieval of only the last data of several encryption/decryption processes.

Therefore, there is no need to define a read buffer in DMA mode.

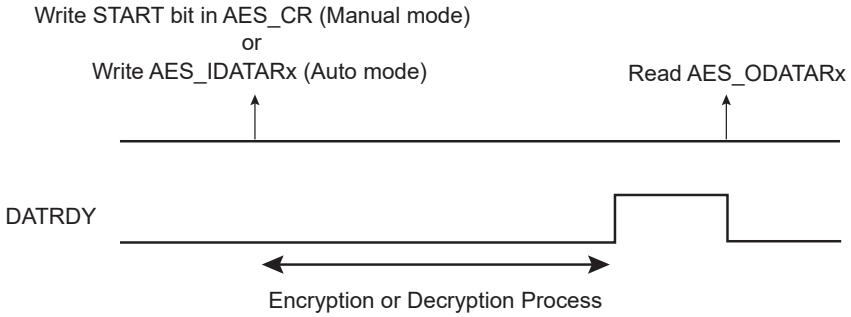
This data are only available in AES_ODATARx.

#### 57.4.3.1 Manual and Auto Modes

##### 57.4.3.1.1 If AES_MR.LOD = 0

The DATRDY flag is cleared when at least one AES_ODATARx is read (see the figure below).

**Figure 57-1. Manual and Auto Modes with AES_MR.LOD = 0**



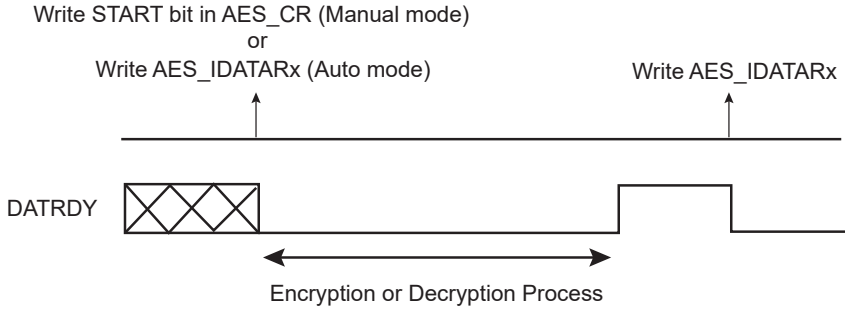

If the user does not want to read AES_ODATARx between each encryption/decryption, the DATRDY flag will not be cleared. If the DATRDY flag is not cleared, the user cannot know the end of the following encryptions/decryptions.

##### 57.4.3.1.2 If AES_MR.LOD = 1

This mode is optimized to process AES CBC-MAC operating mode.

The DATRDY flag is cleared when at least one AES_IDATAR is written (see the figure below). No additional AES_ODATAR reads are necessary between consecutive encryptions/decryptions.

**Figure 57-2. Manual and Auto Modes with AES_MR.LOD = 1**



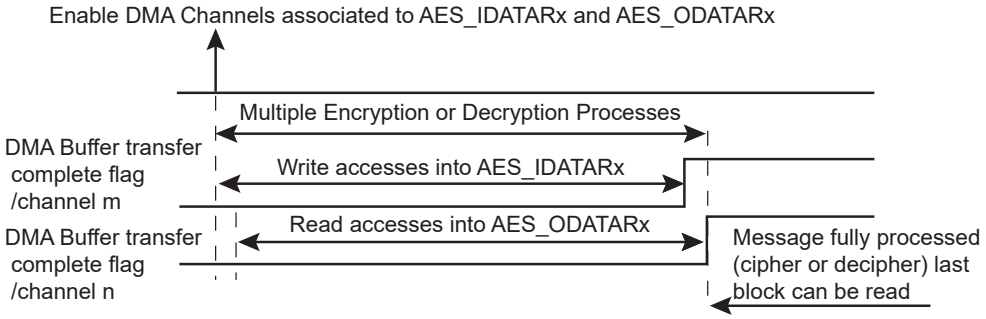

#### 57.4.3.2 DMA Mode

##### 57.4.3.2.1 If AES_MR.LOD = 0

This mode may be used for all AES operating modes except CBC-MAC where AES_MR.LOD = 1 mode is recommended.

The end of the encryption/decryption is indicated by the end of DMA transfer associated to AES_ODATARx (see the figure below). Two DMA channels are required: one for writing message blocks to AES_IDATARx and one to obtain the result from AES_ODATARx.

**Figure 57-3. DMA Transfer with AES_MR.LOD = 0**




#### 57.4.3.2.2 If AES_MR.LOD = 1

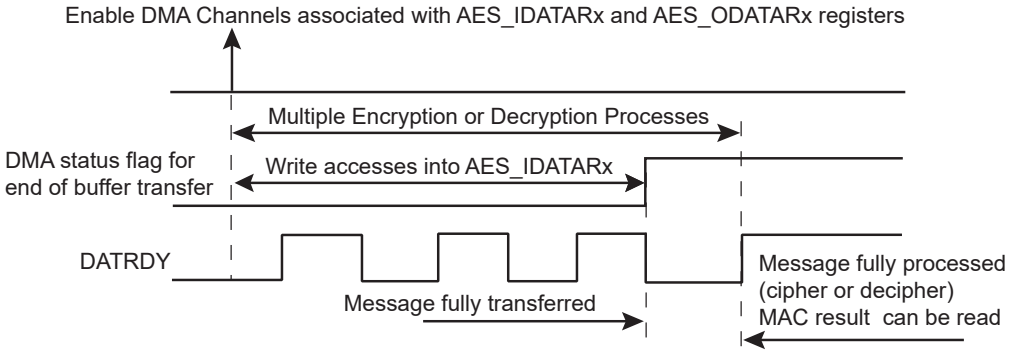
This mode is optimized to process AES CBC-MAC operating mode.

The user must first wait for the DMA buffer transfer complete flag, then for the flag DATRDY to rise to ensure that the encryption/decryption is completed (see the figure below).

In this case, no receive buffers are required.

The output data are only available on AES_ODATARx.

**Figure 57-4. DMA Transfer with AES_MR.LOD = 1**




The table below summarizes the different cases.

**Table 57-1. Last Output Data Mode Behavior versus Start Modes**

| Sequence | Manual and Auto Modes | | DMA Transfer | |
|---|---|---|---|---|
| | AES_MR.LOD = 0 | AES_MR.LOD = 1 | AES_MR.LOD = 0 | AES_MR.LOD = 1 |
| DATRDY Flag Clearing Condition[(1)] | At least one AES_ODATAR must be read | At least one AES_IDATAR must be written | Not used | Managed by the DMA |
| End of Encryption/ Decryption Notification | DATRDY | DATRDY | 2 DMA Buffer transfer complete flags (channel m and channel n) | DMA buffer transfer complete flag, then AES DATRDY flag |
| Encrypted/ Decrypted Data Result Location | In AES_ODATARx | In AES_ODATARx | At the address specified in the Channel Buffer Transfer Descriptor | In AES_ODATARx |

**Note:**

1. Depending on the mode, there are other ways of clearing the DATRDY flag. See AES Interrupt Status Register.

**WARNING** In DMA mode, reading AES_ODATARx before the last data transfer may lead to unpredictable results.

### 57.4.4 Galois/Counter Mode (GCM)

#### 57.4.4.1 Description

GCM comprises the AES engine in CTR mode along with a universal hash function (GHASH engine) that is defined over a binary Galois field to produce a message authentication tag (the AES CTR engine and the GHASH engine are depicted in the figure below.

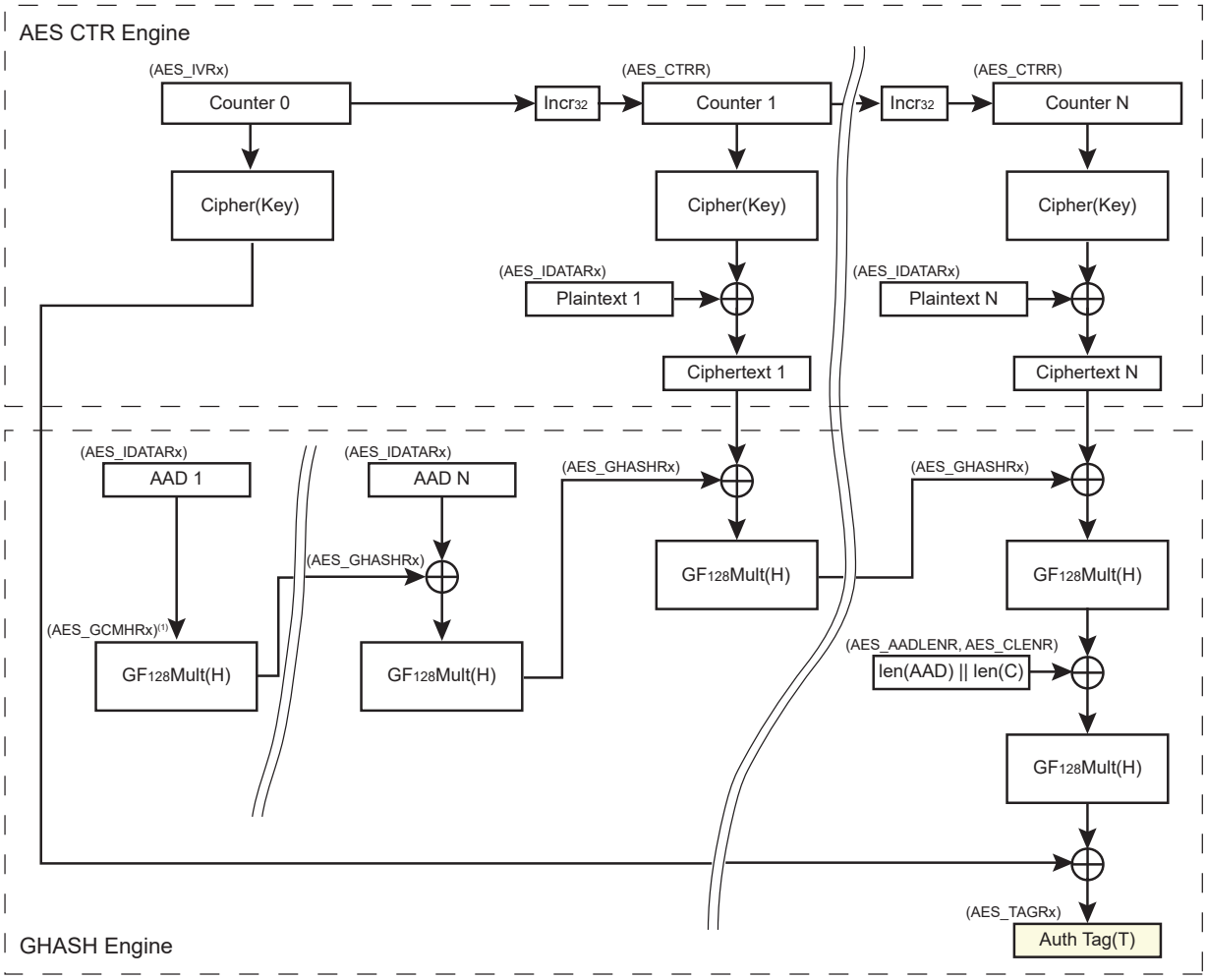
The GHASH engine processes data packets after the AES operation. GCM assures the confidentiality of data through the AES Counter mode of operation for encryption. Authenticity of the confidential data is assured through the GHASH engine. GCM can also provide assurance of data that is not encrypted. Refer to the NIST Special Publication 800-38D for more complete information.

GCM can be used with or without the DMA master. Messages may be processed as a single complete packet of data or they may be broken into multiple packets of data over time.

GCM processing is computed on 128-bit input data fields. There is no support for unaligned data. The AES key length can be whatever length is supported by the AES module.

The recommended programming procedure when using DMAPDC is described in the section GCM Processing.

**Figure 57-5. GCM Block Diagram**




Note: 1. Optional

### 57.4.4.2 Key Writing and Automatic Hash Subkey Calculation

Whenever a new key is written to the hardware, two automatic actions are processed:

- GCM Hash Subkey *H* generation—The GCM hash subkey (*H*) is automatically generated. The GCM hash subkey generation must be complete before doing any other action. AES_ISR.DATRDY indicates when the subkey generation is complete (with interrupt if configured). The GCM hash subkey calculation is processed with the formula *H* = CIPHER(Key, <128 bits to zero>). The generated GCM *H* value is then available in AES_GCMHRx. If the application software requires a specific hash subkey, the automatically generated *H* value can be overwritten in AES_GCMHRx.
  AES_GCMHRx can be written after the end of the hash subkey generation (see AES_ISR.DATRDY) and prior to starting the input data feed.
- AES_GHASHRx Clear—AES_GHASHRx are automatically cleared. If a hash initial value is needed for the GHASH, it must be written to AES_GHASHRx
  – after a write to the AES Key Register, if any
  – before starting the input data feed

### 57.4.4.3 GCM Processing

GCM processing is made up of three phases:

1. Processing the Additional Authenticated Data (*AAD*), hash computation only.
2. Processing the Ciphertext (C), hash computation + ciphering/deciphering.

3. Generating the Tag using length of *AAD*, length of C and $J_0$ (refer to NIST documentation for details).

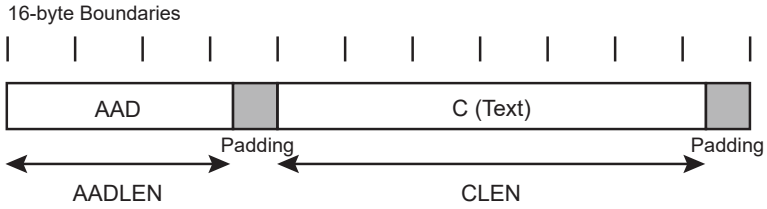
The Tag generation can be done either automatically, after the end of *AAD*/C processing if AES_MR.GTAGEN is set, or manually using AES_GHASHRx.GHASH (see subsections Processing a Complete Message with Tag Generation and Manual GCM Tag Generation for details).

#### 57.4.4.3.1 Processing a Complete Message with Tag Generation

Use this procedure only if J0 four LSB bytes ≠ 0xFFFFFFFF.

**Note:** If $J_0$ four LSB bytes = 0xFFFFFFFF or if the value is unknown, use the procedure described in Processing a Complete Message without Tag Generation followed by the procedure in Manual GCM Tag Generation.

**Figure 57-6. Full Message Alignment**




To process a complete message with Tag generation, the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '1'.
2. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. See Key Writing and Automatic Hash Subkey Calculation.
3. Calculate the $J_0$ value as described in NIST documentation $J_0 = IV \,||\, 0^{31} \,||\, 1$ when len(IV) = 96 and $J_0 = GHASH_H(IV \,||\, 0^{s+64} \,||\, [len(IV)]64)$ if len(IV) ≠ 96. See Processing a Message with only AAD (GHASHH) for $J_0$ generation.
4. Set AES_IVRx.IV with inc32($J_0$) ($J_0$ + 1 on 32 bits).
5. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN.
6. Fill AES_IDATARx.IDATA with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Wait for TAGRDY to be set (use interrupt if needed), then read AES_TAGRx.TAG to obtain the authentication tag of the message.

#### 57.4.4.3.2 Processing a Complete Message without Tag Generation

Processing a message without generating the Tag can be used to customize the Tag generation, or to process a fragmented message. To manually generate the GCM Tag, see Manual GCM Tag Generation.

To process a complete message without Tag generation, the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Calculate the J0 value as described in NIST documentation $J_0 = IV \,||\, 0^{31} \,||\, 1$ when len(IV) = 96 and $J_0 = GHASH_H(IV \,||\, 0^{s+64} \,||\, [len(IV)]64)$ if len(IV) ≠ 96. See Processing a Message with only AAD (GHASHH) for $J_0$ generation example when len(IV) ≠ 96.
4. Set AES_IVRx.IV with inc32($J_0$) ($J_0$ + 1 on 32 bits).
5. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN.
6. Fill AES_IDATARx.IDATA with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if AES_CLENR.CLEN ≠ 0 (or wait for DATRDY), then read AES_GHASHRx.GHASH to obtain the hash value after the last processed data.

#### 57.4.4.3.3 Processing a Fragmented Message without Tag Generation

If needed, a message can be processed by fragments, in such case automatic GCM Tag generation is not supported.

To process a message by fragments, the sequence is as follows:

- First fragment:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait for AES_ISR.DATRDY to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Calculate the $J_0$ value as described in NIST documentation $J_0 = IV || 0^{31} || 1$ when len(IV) = 96 and $J_0 = GHASH_H(IV || 0^{s+64} || [len(IV)]64)$ if len(IV) ≠ 96. See Processing a Message with only AAD (GHASHH) for $J_0$ generation example when len(IV) ≠ 96.
4. Set AES_IVRx.IV with inc32($J_0$) ($J_0$ + 1 on 32 bits).
5. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN according to the length of the first fragment, or set the fields with the full message length (both configurations work).
6. Fill AES_IDATARx.IDATA with the first fragment of the message to process (aligned on 16-byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if the fragment ends in C phase (or wait for DATRDY if the fragment ends in *AAD* phase), then read AES_GHASHRx.GHASH to obtain the value of the hash after the last processed data and finally read AES_CTR.CTR to obtain the value of the CTR encryption counter (not needed when the fragment ends in AAD phase).

- Next fragment (or last fragment):

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Set AES_IVRx.IV as follows:
   – If the first block of the fragment is a block of Additional Authenticated data, set AES_IVRx.IV with the $J_0$ initial value
   – If the first block of the fragment is a block of Plaintext data, set AES_IVRx.IV with a value constructed as follows: 'LSB96($J_0$) || CTR' value, (96 bit LSB of $J_0$ concatenated with saved CTR value from previous fragment).
4. Configure AES_AADLENR.AADLEN and AES_CLENR.CLEN according to the length of the current fragment, or set the fields with the remaining message length, both configurations work.
5. Fill AES_GHASHRx.GHASH with the value stored after the previous fragment.
6. Fill AES_IDATARx.IDATA with the current fragment of the message to process (aligned on 16 byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if the fragment ends in C phase (or wait for DATRDY if the fragment ends in AAD phase), then read AES_GHASHRx.GHASH to obtain the value of the hash after the last processed data and finally read AES_CTR.CTR to obtain the value of the CTR encryption counter (not needed when the fragment ends in *AAD* phase).

**Note:** Step 1 and 2 are required only if the value of the concerned registers has been modified.

Once the last fragment has been processed, the GHASH value will allow manual generation of the GCM tag. See Manual GCM Tag Generation.

#### 57.4.4.3.4 Manual GCM Tag Generation

This section describes the last steps of the GCM Tag generation.

The Manual GCM Tag Generation is used to complete the GCM Tag Generation when the message has been processed without Tag Generation.

**Note:** The Message Processing without Tag Generation must be finished before processing the Manual GCM Tag Generation.

To generate a GCM Tag manually, the sequence is as follows:

Processing S = GHASH$_H$ (*AAD* || 0v || C || 0u || [len(*AAD*)]64 || [len(C)]64):

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.
2. Set the AES Key Register and wait for AES_ISR.DATRDY to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
3. Configure AES_AADLENR.AADLEN to 0x10 (16 bytes) and AES_CLENR.CLEN to '0'. This will allow running a single GHASH$_H$ on a 16-byte input data (see the figure below).
4. Fill AES_GHASHRx.GHASH with the state of the GHASH field stored at the end of the message processing.
5. Fill AES_IDATARx.IDATA according to the SMOD configuration used with 'len(*AAD*)64 || len(*C*)64' value as described in the NIST documentation and wait for DATRDY to be set; use interrupt if needed.
6. Read AES_GHASHRx.GHASH to obtain the current value of the hash.

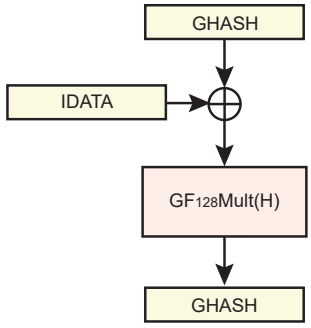
Processing *T* = GCTRK($J_0$, S):

1. Set AES_MR.OPMOD to CTR.
2. Set AES_IVRx.IV with '$J_0$' value.
3. Fill AES_IDATARx.IDATA with the GHASH value read at step 6 and wait for DATRDY to be set (use interrupt if needed).
4. Read AES_ODATARx.ODATA to obtain the GCM Tag value.

**Note:** Step 4 is optional if the GHASH field is to be filled with value '0' (0 length packet for instance).

#### 57.4.4.3.5 Processing a Message with only AAD (GHASHH)

**Figure 57-7. Single GHASH$_H$ Block Diagram (AADLEN ≤ 0x10 and CLEN = 0)**




It is possible to process a message with only *AAD* setting the CLEN field to '0' in AES_CLENR, this can be used for J0 generation when len(*IV*) ≠ 96 for instance.

Example: Processing $J_0$ when len(*IV*) ≠ 96

To process $J_0$ = GHASH$_H$(*IV* || $0^{s+64}$ || [len(*IV*)]64), the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to '0'.

1. Set the AES Key Register and wait until AES_ISR.DATRDY is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation.
2. Configure AES_AADLENR.AADLEN with 'len(*IV* || $0^{s+64}$ || [len(*IV*)]64)' in and AES_CLENR.CLEN to '0'. This will allow running a GHASH$_H$ only.
3. Fill AES_IDATARx.IDATA with the message to process (*IV* || $0^{s+64}$ || [len(*IV*)]64) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a GHASH$_H$ step is over (use interrupt if needed).
4. Read AES_GHASHRx.GHASH to obtain the $J_0$ value.
   Note: The GHASH value can be overwritten at any time by writing the value of AES_GHASHRx.GHASH, used to perform a GHASH$_H$ with an initial value for GHASH (write GHASH field between step 3 and step 4 in this case).

#### 57.4.4.3.6 Processing a Single GF128 Multiplication

The AES can also be used to process a single multiplication in the Galois field on 128 bits (GF$_{128}$) using a single GHASH$_H$ with custom *H* value (see the figure above).

To run a $GF_{128}$ multiplication (A x B), the sequence is as follows:

1. Set AES_MR.OPMOD to GCM and AES_MR.GTAGEN to ‘0’.
    1. Configure AES_AADLENR.AADLEN with 0x10 (16 bytes) and AES_CLENR.CLEN to ‘0’. This will allow running a single $GHASH_H$.
    2. Fill AES_GCMHRx.H with B value.
    3. Fill AES_IDATARx.IDATA with the A value according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a GHASHH computation is over (use interrupt if needed).
    4. Read AES_GHASHRx.GHASH to obtain the result.

**Note:** AES_GHASHRx.GHASH can be initialized with a value C between step 3 and step 4 to run a ((A XOR C) x B) $GF_{128}$ multiplication.

### 57.4.5 Double Input Buffer

AES_IDATARx can be double-buffered to reduce the runtime of large files.

This mode allows a new message block to be written when the previous message block is being processed. This is only possible when DMA accesses are performed (AES_MR.SMOD = 2).

AES_MR.DUALBUFF must be set to ‘1’ to access the double buffer.

### 57.4.6 Start Modes

AES_MR.SMOD allows selection of the encryption (or decryption) Start mode.

#### 57.4.6.1 Manual Mode

The sequence of actions is as follows:

1. Write AES_MR with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 128-bit/192-bit/256-bit AES key in AES_KEYWRx.
3. Write the initialization vector (or counter) in AES_IVRx.
   Note: AES_IVRx concerns all modes except ECB.
4. Set the bit DATRDY (Data Ready) in the AES Interrupt Enable register (AES_IER), depending on whether an interrupt is required or not at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized AES_IDATARx (see the table below).
6. Set the START bit in the AES Control register (AES_CR) to begin the encryption or the decryption process.
7. When processing completes, the DATRDY flag in the AES Interrupt Status register (AES_ISR) is raised. If an interrupt has been enabled by setting AES_IER.DATRDY, the interrupt line of the AES is activated.
8. When software reads one of AES_ODATARx, AES_IER.DATRDY is automatically cleared.

**Table 57-2. Authorized Input Data Registers**

| Operating Mode | Input Data Registers to Write |
|---|---|
| ECB | All |
| CBC | All |
| OFB | All |
| 128-bit CFB | All |
| 64-bit CFB | AES_IDATAR0 and AES_IDATAR1 |
| 32-bit CFB | AES_IDATAR0 |
| 16-bit CFB | AES_IDATAR0 |
| 8-bit CFB | AES_IDATAR0 |
| CTR | All |
| GCM | All |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Notes:**

1. In 64-bit CFB mode, writing to AES_IDATAR2 and AES_IDATAR3 is not allowed and may lead to errors in processing.
2. In 32, 16, and 8-bit CFB modes, writing to AES_IDATAR1, AES_IDATAR2 and AES_IDATAR3 is not allowed and may lead to errors in processing.

#### 57.4.6.2 Auto Mode

The Auto Mode is similar to the manual one, except that in this mode, as soon as the correct number of AES_IDATARx is written, processing is automatically started without any action in AES_CR.

#### 57.4.6.3 DMA Mode

The DMA Controller can be used in association with the AES to perform an encryption/decryption of a buffer without any action by software during processing.

AES_MR.SMOD must be configured to 2 and the DMA must be configured with non-incremental addresses.

The start address of any transfer descriptor must be configured with the address of AES_IDATAR0.

The DMA chunk size configuration depends on the AES mode of operation and is listed in the table below.

When writing data to AES with a first DMA channel, data are first fetched from a memory buffer (source data). It is recommended to configure the size of source data to “words” even for CFB modes. On the contrary, the destination data size depends on the mode of operation. When reading data from the AES with the second DMA channel, the source data is the data read from AES and data destination is the memory buffer. In this case, the source data size depends on the AES mode of operation and is listed in the table below.

**Table 57-3. DMA Data Transfer Type for the Different Operating Modes**

| Operating Mode | Chunk Size | Destination/Source Data Transfer Type |
|---|---|---|
| ECB | 4 | Word |
| CBC | 4 | Word |
| OFB | 4 | Word |
| CFB 128-bit | 4 | Word |
| CFB 64-bit | 1 | Word |
| CFB 32-bit | 1 | Word |
| CFB 16-bit | 1 | Half-word |
| CFB 8-bit | 1 | Byte |
| CTR | 4 | Word |
| GCM | 4 | Word |

### 57.4.7 Security Features

#### 57.4.7.1 Unspecified Register Access Detection

When an unspecified register access occurs, AES_ISR.URAD is raised. Its source is then reported in AES_ISR.URAT. Only the last unspecified register access is available through the AES_ISR.URAT.

Several kinds of unspecified register accesses can occur:

- Input Data register written during the data processing when SMOD = IDATAR0_START
- Output Data register read during data processing
- Mode register written during data processing
- Output Data register read during sub-keys generation
- Mode register written during sub-keys generation
- Write-only register read access

AES_ISR.URAD and AES_ISR.URAT can only be reset by AES_CR.SWRST.

## 57.5 Register Summary

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | AES_CR | 7:0 | | | | | | | | START |
| | | 15:8 | | | | | | | | SWRST |
| | | 23:16 | | | | | | | | |
| | | 31:24 | | | | | | | | |
| 0x04 | AES_MR | 7:0 | PROCDLY[3:0] | | | | DUALBUFF | | GTAGEN | CIPHER |
| | | 15:8 | LOD | OPMOD[2:0] | | | KEYSIZE[1:0] | | SMOD[1:0] | |
| | | 23:16 | CKEY[3:0] | | | | | CFBS[2:0] | | |
| | | 31:24 | | | | | | | | |
| 0x08 ... 0x0F | Reserved | | | | | | | | | |
| 0x10 | AES_IER | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | TAGRDY |
| | | 31:24 | | | | | | | | |
| 0x14 | AES_IDR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | TAGRDY |
| | | 31:24 | | | | | | | | |
| 0x18 | AES_IMR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | | | | | | | | URAD |
| | | 23:16 | | | | | | | | TAGRDY |
| | | 31:24 | | | | | | | | |
| 0x1C | AES_ISR | 7:0 | | | | | | | | DATRDY |
| | | 15:8 | URAT[3:0] | | | | | | | URAD |
| | | 23:16 | | | | | | | | TAGRDY |
| | | 31:24 | | | | | | | | |
| 0x20 | AES_KEYWR0 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x24 | AES_KEYWR1 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x28 | AES_KEYWR2 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x2C | AES_KEYWR3 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x30 | AES_KEYWR4 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x34 | AES_KEYWR5 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x38 | AES_KEYWR6 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x3C | AES_KEYWR7 | 7:0 | KEYW[7:0] | | | | | | | |
| | | 15:8 | KEYW[15:8] | | | | | | | |
| | | 23:16 | KEYW[23:16] | | | | | | | |
| | | 31:24 | KEYW[31:24] | | | | | | | |
| 0x40 | AES_IDATAR0 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x44 | AES_IDATAR1 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x48 | AES_IDATAR2 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x4C | AES_IDATAR3 | 7:0 | IDATA[7:0] | | | | | | | |
| | | 15:8 | IDATA[15:8] | | | | | | | |
| | | 23:16 | IDATA[23:16] | | | | | | | |
| | | 31:24 | IDATA[31:24] | | | | | | | |
| 0x50 | AES_ODATAR0 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x54 | AES_ODATAR1 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x58 | AES_ODATAR2 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x5C | AES_ODATAR3 | 7:0 | ODATA[7:0] | | | | | | | |
| | | 15:8 | ODATA[15:8] | | | | | | | |
| | | 23:16 | ODATA[23:16] | | | | | | | |
| | | 31:24 | ODATA[31:24] | | | | | | | |
| 0x60 | AES_IVR0 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x64 | AES_IVR1 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x68 | AES_IVR2 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x6C | AES_IVR3 | 7:0 | IV[7:0] | | | | | | | |
| | | 15:8 | IV[15:8] | | | | | | | |
| | | 23:16 | IV[23:16] | | | | | | | |
| | | 31:24 | IV[31:24] | | | | | | | |
| 0x70 | AES_AADLENR | 7:0 | AADLEN[7:0] | | | | | | | |
| | | 15:8 | AADLEN[15:8] | | | | | | | |
| | | 23:16 | AADLEN[23:16] | | | | | | | |
| | | 31:24 | AADLEN[31:24] | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Offset | Name | Bit Pos. | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x74 | AES_CLENR | 7:0 | CLEN[7:0] | | | | | | | |
| | | 15:8 | CLEN[15:8] | | | | | | | |
| | | 23:16 | CLEN[23:16] | | | | | | | |
| | | 31:24 | CLEN[31:24] | | | | | | | |
| 0x78 | AES_GHASHR0 | 7:0 | GHASH[7:0] | | | | | | | |
| | | 15:8 | GHASH[15:8] | | | | | | | |
| | | 23:16 | GHASH[23:16] | | | | | | | |
| | | 31:24 | GHASH[31:24] | | | | | | | |
| 0x7C | AES_GHASHR1 | 7:0 | GHASH[7:0] | | | | | | | |
| | | 15:8 | GHASH[15:8] | | | | | | | |
| | | 23:16 | GHASH[23:16] | | | | | | | |
| | | 31:24 | GHASH[31:24] | | | | | | | |
| 0x80 | AES_GHASHR2 | 7:0 | GHASH[7:0] | | | | | | | |
| | | 15:8 | GHASH[15:8] | | | | | | | |
| | | 23:16 | GHASH[23:16] | | | | | | | |
| | | 31:24 | GHASH[31:24] | | | | | | | |
| 0x84 | AES_GHASHR3 | 7:0 | GHASH[7:0] | | | | | | | |
| | | 15:8 | GHASH[15:8] | | | | | | | |
| | | 23:16 | GHASH[23:16] | | | | | | | |
| | | 31:24 | GHASH[31:24] | | | | | | | |
| 0x88 | AES_TAGR0 | 7:0 | TAG[7:0] | | | | | | | |
| | | 15:8 | TAG[15:8] | | | | | | | |
| | | 23:16 | TAG[23:16] | | | | | | | |
| | | 31:24 | TAG[31:24] | | | | | | | |
| 0x8C | AES_TAGR1 | 7:0 | TAG[7:0] | | | | | | | |
| | | 15:8 | TAG[15:8] | | | | | | | |
| | | 23:16 | TAG[23:16] | | | | | | | |
| | | 31:24 | TAG[31:24] | | | | | | | |
| 0x90 | AES_TAGR2 | 7:0 | TAG[7:0] | | | | | | | |
| | | 15:8 | TAG[15:8] | | | | | | | |
| | | 23:16 | TAG[23:16] | | | | | | | |
| | | 31:24 | TAG[31:24] | | | | | | | |
| 0x94 | AES_TAGR3 | 7:0 | TAG[7:0] | | | | | | | |
| | | 15:8 | TAG[15:8] | | | | | | | |
| | | 23:16 | TAG[23:16] | | | | | | | |
| | | 31:24 | TAG[31:24] | | | | | | | |
| 0x98 | AES_CTRR | 7:0 | CTR[7:0] | | | | | | | |
| | | 15:8 | CTR[15:8] | | | | | | | |
| | | 23:16 | CTR[23:16] | | | | | | | |
| | | 31:24 | CTR[31:24] | | | | | | | |
| 0x9C | AES_GCMHR0 | 7:0 | H[7:0] | | | | | | | |
| | | 15:8 | H[15:8] | | | | | | | |
| | | 23:16 | H[23:16] | | | | | | | |
| | | 31:24 | H[31:24] | | | | | | | |
| 0xA0 | AES_GCMHR1 | 7:0 | H[7:0] | | | | | | | |
| | | 15:8 | H[15:8] | | | | | | | |
| | | 23:16 | H[23:16] | | | | | | | |
| | | 31:24 | H[31:24] | | | | | | | |
| 0xA4 | AES_GCMHR2 | 7:0 | H[7:0] | | | | | | | |
| | | 15:8 | H[15:8] | | | | | | | |
| | | 23:16 | H[23:16] | | | | | | | |
| | | 31:24 | H[31:24] | | | | | | | |
| 0xA8 | AES_GCMHR3 | 7:0 | H[7:0] | | | | | | | |
| | | 15:8 | H[15:8] | | | | | | | |
| | | 23:16 | H[23:16] | | | | | | | |
| | | 31:24 | H[31:24] | | | | | | | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 57.5.1 AES Control Register

**Name:** AES_CR
**Offset:** 0x00
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | SWRST |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | START |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 8 – SWRST** Software Reset

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Resets the AES. A software-triggered hardware reset of the AES interface is performed. |

**Bit 0 – START** Start Processing

| Value | Description |
|---|---|
| 0 | No effect. |
| 1 | Starts manual encryption/decryption process. |

### 57.5.2 AES Mode Register

**Name:** AES_MR
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CKEY[3:0] | | | | | CFBS[2:0] | | |
| Access | W | W | W | W | | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | – | | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | LOD | OPMOD[2:0] | | | KEYSIZE[1:0] | | SMOD[1:0] | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | PROCDLY[3:0] | | | | DUALBUFF | | GTAGEN | CIPHER |
| Access | R/W | R/W | R/W | R/W | R/W | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | | 0 | 0 |

**Bits 23:20 – CKEY[3:0]** Key

| Value | Name | Description |
|---|---|---|
| 0xE | PASSWD | This field must be written with 0xE the first time AES_MR is programmed. For subsequent programming of AES_MR, any value can be written, including that of 0xE.<br>Always reads as 0. |

**Bits 18:16 – CFBS[2:0]** Cipher Feedback Data Size

| Value | Name | Description |
|---|---|---|
| 0 | SIZE_128BIT | 128-bit |
| 1 | SIZE_64BIT | 64-bit |
| 2 | SIZE_32BIT | 32-bit |
| 3 | SIZE_16BIT | 16-bit |
| 4 | SIZE_8BIT | 8-bit |

**Bit 15 – LOD** Last Output Data Mode

**WARNING** In DMA mode, reading to the Output Data registers before the last data encryption/decryption process may lead to unpredictable results.

| Value | Description |
|---|---|
| 0 | No effect.<br>After each end of encryption/decryption, the output data are available either on the output data registers (Manual and Auto modes) or at the address specified in the Channel Buffer Transfer Descriptor for DMA mode.<br>In Manual and Auto modes, the DATRDY flag is cleared when at least one of the Output Data registers is read. |

| Value | Description |
|---|---|
| 1 | The DATRDY flag is cleared when at least one of the Input Data Registers is written.<br>No more Output Data Register reads are necessary between consecutive encryptions/decryptions (see Last Output Data Mode). |

**Bits 14:12 – OPMOD[2:0]** Operating Mode
For CBC-MAC operating mode, set OPMOD to CBC and LOD to 1.

| Value | Name | Description |
|---|---|---|
| 0 | ECB | ECB: Electronic Codebook mode |
| 1 | CBC | CBC: Cipher Block Chaining mode |
| 2 | OFB | OFB: Output Feedback mode |
| 3 | CFB | CFB: Cipher Feedback mode |
| 4 | CTR | CTR: Counter mode (16-bit internal counter) |
| 5 | GCM | GCM: Galois/Counter mode |

**Bits 11:10 – KEYSIZE[1:0]** Key Size

| Value | Name | Description |
|---|---|---|
| 0 | AES128 | AES Key Size is 128 bits |
| 1 | AES192 | AES Key Size is 192 bits |
| 2 | AES256 | AES Key Size is 256 bits |

**Bits 9:8 – SMOD[1:0]** Start Mode
If a DMA transfer is used, configure SMOD to 2. See DMA Mode for more details.

| Value | Name | Description |
|---|---|---|
| 0 | MANUAL_START | Manual Mode |
| 1 | AUTO_START | Auto Mode |
| 2 | IDATAR0_START | AES_IDATAR0 access only Auto Mode (DMA) |

**Bits 7:4 – PROCDLY[3:0]** Processing Delay
Processing Time = N × (PROCDLY + 1)
where

- N = 10 when KEYSIZE = 0
- N = 12 when KEYSIZE = 1
- N = 14 when KEYSIZE = 2

The processing time represents the number of clock cycles that the AES needs in order to perform one encryption/decryption.
**Note:** The best performance is achieved with PROCDLY equal to 0.

**Bit 3 – DUALBUFF** Dual Input Buffer

| Value | Name | Description |
|---|---|---|
| 0 | INACTIVE | AES_IDATARx cannot be written during processing of previous block. |
| 1 | ACTIVE | AES_IDATARx can be written during processing of previous block when SMOD = 2. It speeds up the overall runtime of large files. |

**Bit 1 – GTAGEN** GCM Automatic Tag Generation Enable

| Value | Description |
|---|---|
| 0 | Automatic GCM Tag generation disabled. |
| 1 | Automatic GCM Tag generation enabled. |

**Bit 0 – CIPHER** Processing Mode

| Value | Description |
|---|---|
| 0 | Decrypts data. |
| 1 | Encrypts data. |

### 57.5.3 AES Interrupt Enable Register

**Name:** AES_IER
**Offset:** 0x10
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TAGRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 16 – TAGRDY** GCM Tag Ready Interrupt Enable

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Enable

**Bit 0 – DATRDY** Data Ready Interrupt Enable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 57.5.4 AES Interrupt Disable Register

**Name:** AES_IDR
**Offset:** 0x14
**Reset:** –
**Property:** Write-only

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TAGRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | W |
| Reset | | | | | | | | – |

**Bit 16 – TAGRDY** GCM Tag Ready Interrupt Disable

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Disable

**Bit 0 – DATRDY** Data Ready Interrupt Disable

© 2021 Microchip Technology Inc.
and its subsidiaries

### 57.5.5 AES Interrupt Mask Register

**Name:** AES_IMR
**Offset:** 0x18
**Reset:** 0x00000000
**Property:** Read-only

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TAGRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | URAD |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 16 – TAGRDY** GCM Tag Ready Interrupt Mask

**Bit 8 – URAD** Unspecified Register Access Detection Interrupt Mask

**Bit 0 – DATRDY** Data Ready Interrupt Mask

### 57.5.6 AES Interrupt Status Register

**Name:** AES_ISR
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | TAGRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | URAT[3:0] | | | | | | | URAD |
| Access | R | R | R | R | | | | R |
| Reset | 0 | 0 | 0 | 0 | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATRDY |
| Access | | | | | | | | R |
| Reset | | | | | | | | 0 |

**Bit 16 – TAGRDY** GCM Tag Ready

| Value | Description |
|---|---|
| 0 | GCM Tag is not valid. |
| 1 | GCM Tag generation is complete (cleared by reading GCM Tag, starting another processing or when writing a new key). |

**Bits 15:12 – URAT[3:0]** Unspecified Register Access (cleared by writing SWRST in AES_CR)
Only the last Unspecified Register Access Type is available through the URAT field.

| Value | Name | Description |
|---|---|---|
| 0 | IDR_WR_PROCESSING | Input Data register written during the data processing when SMOD = 2 mode. |
| 1 | ODR_RD_PROCESSING | Output Data register read during the data processing. |
| 2 | MR_WR_PROCESSING | Mode register written during the data processing. |
| 3 | ODR_RD_SUBKGEN | Output Data register read during the sub-keys generation. |
| 4 | MR_WR_SUBKGEN | Mode register written during the sub-keys generation. |
| 5 | WOR_RD_ACCESS | Write-only register read access. |

**Bit 8 – URAD** Unspecified Register Access Detection Status (cleared by writing SWRST in AES_CR)

| Value | Description |
|---|---|
| 0 | No unspecified register access has been detected since the last SWRST. |
| 1 | At least one unspecified register access has been detected since the last SWRST. |

**Bit 0 – DATRDY** Data Ready (cleared by setting bit START or bit SWRST in AES_CR or by reading AES_ODATARx)

| Value | Description |
|---|---|
| 0 | Output data not valid. |
| 1 | Encryption or decryption process is completed. |

**Note:** If AES_MR.LOD = 1: In Manual and Auto mode, the DATRDY flag can also be cleared by writing at least one AES_IDATARx.

### 57.5.7 AES Key Word Register x

**Name:** AES_KEYWRx
**Offset:** 0x20 + x*0x04 [x=0..7]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | KEYW[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – KEYW[31:0]** Key Word
The four/six/eight 32-bit Key Word registers set the 128-bit/192-bit/256-bit cryptographic key used for AES encryption/decryption.
AES_KEYWR0 corresponds to the first word of the key and respectively AES_KEYWR3/AES_KEYWR5/ AES_KEYWR7 to the last one.
Whenever a new key (AES_KEYWRx) is written to the hardware, two automatic actions are processed:

- GCM hash subkey generation
- AES_GHASHRx Clear

See Key Writing and Automatic Hash Subkey Calculation for details.
These registers are write-only to prevent the key from being read by another application.

### 57.5.8 AES Input Data Register x

**Name:** AES_IDATARx
**Offset:** 0x40 + x*0x04 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IDATA[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – IDATA[31:0]** Input Data Word
The four 32-bit Input Data registers set the 128-bit data block used for encryption/decryption.
AES_IDATAR0 corresponds to the first word of the data to be encrypted/decrypted, and AES_IDATAR3 to the last one.
These registers are write-only to prevent the input data from being read by another application.

### 57.5.9 AES Output Data Register x

**Name:** AES_ODATARx
**Offset:** 0x50 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ODATA[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – ODATA[31:0]** Output Data
The four 32-bit Output Data registers contain the 128-bit data block that has been encrypted/decrypted. AES_ODATAR0 corresponds to the first word, AES_ODATAR3 to the last one.

### 57.5.10 AES Initialization Vector Register x

**Name:** AES_IVRx
**Offset:** 0x60 + x*0x04 [x=0..3]
**Reset:** –
**Property:** Write-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | IV[31:24] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | IV[23:16] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | IV[15:8] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | IV[7:0] | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | – |

**Bits 31:0 – IV[31:0]** Initialization Vector
The four 32-bit Initialization Vector registers set the 128-bit Initialization Vector data block that is used by some modes of operation as an additional initial input.
AES_IVR0 corresponds to the first word of the Initialization Vector, AES_IVR3 to the last one.
These registers are write-only to prevent the Initialization Vector from being read by another application.
For CBC, OFB and CFB modes, the IV input value corresponds to the initialization vector.
For CTR mode, the IV input value corresponds to the initial counter value.
**Note:** These registers are not used in ECB mode and must not be written.

### 57.5.11 AES Additional Authenticated Data Length Register

**Name:** AES_AADLENR
**Offset:** 0x70
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | AADLEN[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | AADLEN[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | AADLEN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | AADLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – AADLEN[31:0]** Additional Authenticated Data Length
Length in bytes of the Additional Authenticated Data (AAD) that is to be processed.
**Note:** The maximum byte length of the AAD portion of a message is limited to the 32-bit counter length.

### 57.5.12 AES Plaintext/Ciphertext Length Register

**Name:** AES_CLENR
**Offset:** 0x74
**Reset:** 0x00000000
**Property:** Read/Write

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CLEN[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CLEN[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CLEN[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CLEN[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CLEN[31:0]** Plaintext/Ciphertext Length
Length in bytes of the plaintext/ciphertext (C) data that is to be processed.
**Note:** The maximum byte length of the C portion of a message is limited to the 32-bit counter length.

### 57.5.13 AES GCM Intermediate Hash Word Register x

**Name:** AES_GHASHRx
**Offset:** 0x78 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** R/W

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | GHASH[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | GHASH[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | GHASH[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | GHASH[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – GHASH[31:0]** Intermediate GCM Hash Word x
The four 32-bit Intermediate Hash Word registers expose the intermediate GHASH value. May be read to save the current GHASH value so processing can later be resumed, presumably on a later message fragment. Whenever a new key is written to the AES Key Register two automatic actions are processed:

- GCM hash subkey generation
- AES_GHASHRx Clear

See Key Writing and Automatic Hash Subkey Calculation for details.
If an application software-specific hash initial value is needed for the GHASH, it must be written to AES_GHASHRx:

- after a write to the AES Key Register, if any,
- before starting the input data feed.

### 57.5.14 AES GCM Authentication Tag Word Register x

**Name:** AES_TAGRx
**Offset:** 0x88 + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | TAG[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | TAG[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | TAG[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | TAG[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – TAG[31:0]** GCM Authentication Tag x
The four 32-bit Tag registers contain the final 128-bit GCM Authentication tag (*T*) when GCM processing is complete. TAG0 corresponds to the first word, TAG3 to the last word.

### 57.5.15 AES GCM Encryption Counter Value Register

**Name:** AES_CTRR
**Offset:** 0x98
**Reset:** 0x00000000
**Property:** Read-only

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | CTR[31:24] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | CTR[23:16] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | CTR[15:8] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | CTR[7:0] | | | | | | | |
| Access | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – CTR[31:0]** GCM Encryption Counter
Reports the current value of the 32-bit GCM counter.

### 57.5.16 AES GCM H Word Register x

**Name:** AES_GCMHRx
**Offset:** 0x9C + x*0x04 [x=0..3]
**Reset:** 0x00000000
**Property:** R/W

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | H[31:24] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | H[23:16] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | H[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | H[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 31:0 – H[31:0]** GCM H Word x
The four 32-bit H Word registers contain the 128-bit GCM hash subkey *H*value.
Whenever a new key is written to the AES Key Register, two automatic actions are processed:

- GCM hash subkey *H* generation
- AES_GHASHRx Clear

If the application software requires a specific hash subkey, the automatically-generated H value can be overwritten in AES_GCMHRx. See Key Writing and Automatic Hash Subkey Calculation for details.
Generating a GCM hash subkey *H* by a write in AES_GCMHRx enables to:

- select the GCM hash subkey *H* for GHASH operations,
- select one operand to process a single GF128 multiply.

# 58. Electrical Characteristics

## 58.1 Absolute Maximum Ratings

**Table 58-1. Absolute Maximum Ratings***

| | | |
|---|---|---|
| Storage Temperature | -60°C to + 150°C | *Notice: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. |
| Voltage on Input Pins with Respect to Ground | -0.3V to + 4.0V | |
| Maximum Operating Voltage VDDPLL, VDDUTMIC, VDDCORE | 1.4V | |
| Maximum Operating Voltage VDDIO, VDDUTMII, VDDPLLUSB, VDDIN | 4.0V | |
| Total DC Output Current on all I/O lines | 150 mA | |

**Table 58-2. Recommended Thermal Operating Conditions**

| Symbol | Parameter | Conditions | Minimum | Typical | Maximum | Unit |
|---|---|---|---|---|---|---|
| $T_A$ | Operating Temperature | – | -55 | - | +125 | °C |
| $R_{JA}$ | Junction-to-ambient Thermal Resistance | CQFP144 | – | - | 8 | °C/W |
| | | LQFP144 | – | – | 40 | °C/W |

## 58.2 DC Characteristics

The following characteristics are applicable to the specified operating temperature range, unless otherwise specified.

**Table 58-3. DC Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{DDCORE}$ | DC Supply Core | – | 1.20 | – | 1.32 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 20 MHz | – | – | 20 | mV |
| | Rising Slope | – | 1.20 | – | 30 | V/ms |
| $V_{DDIO}$ | DC Supply I/Os, Backup | (See **Note 1**) | 3.0 | 3.3 | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 30 | mV |
| | Rising Slope | – | 6.5 | – | 30 | V/ms |
| $V_{DDIN}$ | DC Supply Voltage Regulator | (See **Note 1**) | 3.0 | 3.3 | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 20 MHz | – | – | 20 | mv |
| $V_{DDPLL}$ | PLL A and Main Oscillator Supply | – | 1.20 | – | 1.32 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| | | rms value > 10 MHz | – | – | 10 | |

© 2021 Microchip Technology Inc. and its subsidiaries

.....continued

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{DDUTMIC}$ | DC Supply UDPHS and UHPHS UTMI+ Core | – | 1.20 | – | 1.32 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 10 | mV |
| $V_{DDUTMII}$ | DC Supply UDPHS and UHPHS UTMI+ Interface | – | 3.0 | 3.3 | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 20 | mV |
| $V_{DDPLLUSB}$ | DC Supply UTMI PLL | – | 3.0 | 3.3 | 3.6 | V |
| | Allowable Voltage Ripple | rms value 10 kHz to 10 MHz | – | – | 10 | mV |
| $V_{IL}$ | Low-level Input Voltage | GPIO_MLB | -0.3 | – | 0.7 | V |
| | | GPIO_AD, GPIO_CLK | -0.3 | – | 0.8 | |
| | | GPIO, CLOCK, RST, TEST | -0.3 | – | $V_{DDIO}$ x 0.3 | |
| $V_{IH}$ | High-level Input Voltage | GPIO_MLB | 1.80 | – | $V_{DDIO}$ + 0.3 | V |
| | | GPIO_AD, GPIO_CLK | 2 | | $V_{DDIO}$ + 0.3 | |
| | | GPIO, CLOCK, RST, TEST | $V_{DDIO}$ x 0.7 | – | $V_{DDIO}$ + 0.3 | |
| $V_{OH}$ | High-level Output Voltage | GPIO_MLB, $I_{OH}$ = 6 mA | 2 | – | – | V |
| | | GPIO_AD, GPIO, RST, TEST, CLOCK , $I_{OH}$ = 2 mA, Low drive | $V_{DDIO}$ - 0.4 | – | – | |
| | | GPIO_AD, GPIO, RST, TEST, CLOCK , $I_{OH}$ = 4 mA, High drive | $V_{DDIO}$ - 0.4 | – | – | |
| | | GPIO_CLK, $I_{OH}$ = 3 mA, Low drive | $V_{DDIO}$ - 0.4 | – | – | |
| | | GPIO_CLK, $I_{OH}$ = 6 mA, High drive | $V_{DDIO}$ - 0.4 | – | – | |
| $V_{OL}$ | Low-level Output Voltage | GPIO_MLB, $I_{OL}$= -6 mA | – | – | 0.4 | V |
| | | GPIO_AD, GPIO, RST, TEST, CLOCK,$I_{OL}$ = -2 mA, Low drive | – | – | 0.4 | |
| | | GPIO_AD, GPIO, RST, TEST, CLOCK,$I_{OL}$ = -5 mA, High drive | – | – | 0.4 | |
| | | GPIO_CLK, IOL= -3 mA, Low drive | – | – | 0.4 | |
| | | GPIO_CLK, $I_{OL}$= -6 mA, High drive | – | – | 0.4 | |
| $V_{hys}$ | Hysteresis Voltage | GPIO with Hysteresis mode enabled | 150 | – | – | mV |
| $I_{IL}$ | Low-level Input Current | Pullup OFF | -1 | – | 1 | µA |
| | | Pullup ON | 10 | – | 55 | |
| $I_{IH}$ | High-level Input Current | Pulldown OFF | -1 | – | 1 | µA |
| | | Pulldown ON | 10 | – | 55 | |
| $R_{SERIAL}$ | Serial Resistor | GPIO_MLB | – | 9 | – | Ohm |
| | | GPIO_AD, GPIO_CLK | – | 14 | – | |
| | | GPIO, CLOCK, RST, TEST | – | 26 | – | |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Note:**

1. $V_{DDIO}$ voltage must be equal to $V_{DDIN}$ voltage.

**Table 58-4. Voltage Regulator Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{DDOUT}$ | DC Output Voltage | Normal mode, $I_{LOAD}$ = 100 mA | 1.2 | 1.23 | 1.26 | V |
| | | Standby mode | – | 0 | – | |
| $I_{LOAD}$ | Maximum DC Output Current | | – | – | 150 | mA |
| $C_{DIN}$ | Input Decoupling Capacitor | (See **Note 1**) | – | 4.7 | – | µF |
| $C_{DOUT}$ | Output Decoupling Capacitor | (See **Note 2**) | – | 1 | – | µF |
| | | ESR | – | – | 2 | Ohm |
| $t_{ON}$ | Turn-on Time | $C_{DOUT}$ = 1 µF, $V_{DDOUT}$ reaches DC output voltage | – | 1 | 2.5 | ms |

**Notes:**

1. A 4.7 µF (±20%) or higher ceramic capacitor must be connected between $V_{DDIN}$ and the closest GND pin of the device. This large decoupling capacitor is mandatory to reduce start-up current, improving transient response and noise rejection.
2. To ensure stability, an external 1 µF (±20%) output capacitor, $C_{DOUT}$, must be connected between $V_{DDOUT}$ and the closest GND pin of the device. Solid tantalum and multilayer ceramic capacitors are suitable as output capacitors. A 100 nF bypass capacitor between $V_{DDOUT}$ and the closest GND pin of the device helps decrease output noise and improves the load transient response.

**Table 58-5. Core Power Supply Brownout Detector Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{T-}$ | Supply Falling Threshold (see **Note 1**) | – | 0.97 | 1.0 | 1.04 | V |
| $V_{hys}$ | Hysteresis Voltage | – | – | 25 | 50 | mV |
| $t_{START}$ | Start-up Time | From Disabled state to Enabled state | – | – | 400 | µs |

**Note:**

1. The Brownout Detector is configured using the BODDIS bit in the SUPC_MR register.

**Figure 58-1. Core Brownout Output Waveform**

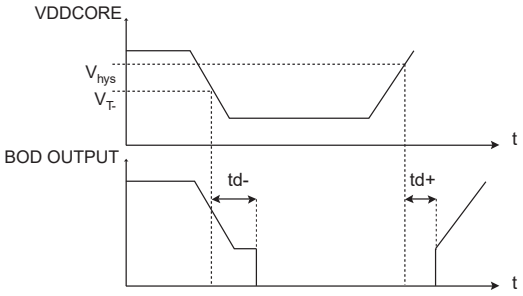



**Table 58-6. VDDCORE Power-on Reset Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{T+}$ | Threshold Voltage Rising | – | 0.79 | 0.95 | 1.07 | V |
| $V_{T-}$ | Threshold Voltage Falling | – | 0.66 | 0.89 | – | V |
| $V_{hys}$ | Hysteresis Voltage | – | 10 | 60 | 115 | mV |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | | | | | | |
|---|---|---|---|---|---|---|
| **Symbol** | **Parameter** | **Conditions** | **Min** | **Typ** | **Max** | **Unit** |
| $t_{RES}$ | Reset Timeout Period | – | 240 | 350 | 800 | µs |

**Figure 58-2. VDDCORE Power-On Reset Characteristics**

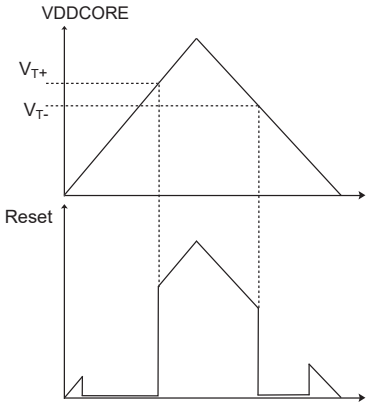



**Table 58-7. VDDIO Supply Monitor**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_T$ | Supply Monitor Threshold | 16 selectable steps (see the Threshold Selection table below) | – | – | – | V |
| $T_{ACC}$ | Threshold Accuracy | – | -4 | – | 4 | % |
| $V_{hys}$ | Hysteresis Voltage | – | – | 38 | 45 | mV |
| $t_{START}$ | Start-up Time | From Disabled state to Enabled state | – | – | 300 | µs |

**Table 58-8. Threshold Selection**

| Symbol | Parameter | Digital Code | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_T$ | Supply Monitor Threshold | 0 | – | 1.6 | – | V |
| | | 1 | – | 1.72 | – | |
| | | 10 | – | 1.84 | – | |
| | | 11 | – | 1.96 | – | |
| | | 100 | – | 2.08 | – | |
| | | 101 | – | 2.2 | – | |
| | | 110 | – | 2.32 | – | |
| | | 111 | – | 2.44 | – | |
| | | 1000 | – | 2.56 | – | |
| | | 1001 | – | 2.68 | – | |
| | | 1010 | – | 2.8 | – | |
| | | 1011 | – | 2.92 | – | |
| | | 1100 | – | 3.04 | – | |
| | | 1101 | – | 3.16 | – | |
| | | 1110 | – | 3.28 | – | |
| | | 1111 | – | 3.4 | – | |

**Figure 58-3. VDDIO Supply Monitor**

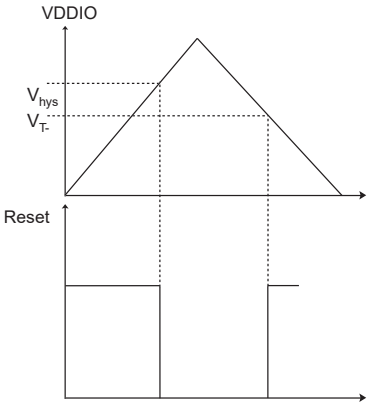
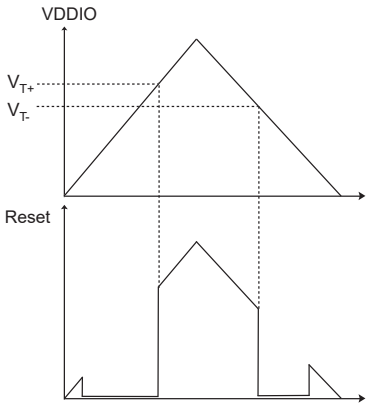



**Table 58-9. VDDIO Power-On Reset Characteristics**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{T+}$ | Threshold Voltage Rising | – | 1.45 | 1.53 | 1.61 | V |
| $V_{T-}$ | Threshold Voltage Falling | – | 1.37 | 1.46 | – | V |
| $V_{hys}$ | Hysteresis | – | 40 | 80 | 130 | mV |
| $t_{RES}$ | Reset Time-out Period | – | 240 | 320 | 800 | µs |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 58-4. VDDIO Power-On Reset Characteristics**




## 58.3 Power Consumption

- Power consumption of the device depending on the different Low-Power modes (Backup, Wait, Sleep) and Active mode
- Power consumption on power supply in different modes: Backup, Wait, Sleep and Active
- Static and dynamic power consumption of the I/Os

### 58.3.1 Backup Mode Current Consumption and Wakeup Time

The Backup mode configurations and measurements are defined as follows:

- Embedded slow clock RC oscillator is enabled
- Supply Monitor on $V_{DDIO}$ is disabled
- RTC is running
- RTT is enabled on 1 Hz mode
- BOD is disabled
- One WKUPx enabled
- Current measurement on AMP1 with and without the 1 Kbyte backup SRAM
- Measurements are made at ambient temperature

**Figure 58-5. Measurement Setup**

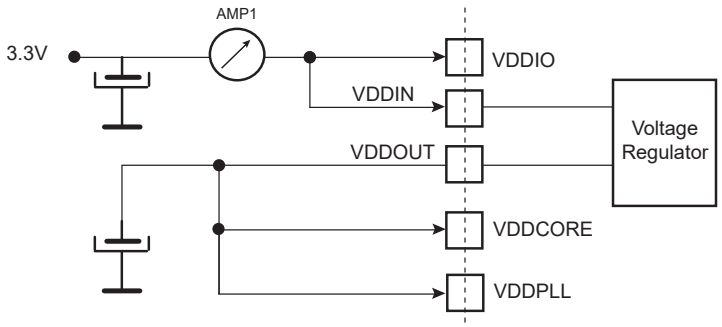



**Table 58-10. Worst Case Power Consumption for Backup Mode with 1 Kbyte Backup SRAM On**

| Total Consumption | Worst Case Value | | |
|---|---|---|---|
| | at 25°C | at 125°C | |
| Conditions | AMP1 | | Unit |
| $V_{DDIO}$ = 3.6V | 8.4 | 80 | µA |

**Table 58-11. Worst Case Power Consumption for Backup Mode with 1 Kbyte Backup SRAM Off**

| Total Consumption | Worst Case Value | | |
|---|---|---|---|
| | at 25°C | at 125°C | |
| Conditions | | AMP1 | Unit |
| $V_{DDIO}$ = 3.6V | 5.1 | 38 | µA |

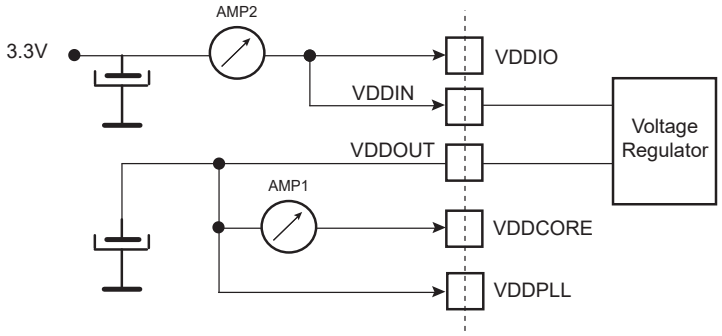
### 58.3.2 Sleep Mode Current Consumption and Wakeup Time

The Sleep mode configuration and measurements are defined as follows:

- Core clock OFF
- $V_{DDIO} = V_{DDIN} = 3.3V$
- Master Clock (MCK) running at various frequencies with PLLA or the fast RC oscillator
- Fast startup through WKUP0–13 pins
- Current measurement as shown below and associated wakeup time(1)
- All peripheral clocks deactivated
- $T_A$ = 25°C

  Note: 1. Wakeup time is defined as the delay between the WKUP event and the execution of the first instruction.

**Figure 58-6. Measurement Setup for Sleep Mode**




The following tables give current consumption and wakeup time in Sleep mode.

**Table 58-12. Typical Sleep Mode Current Consumption vs. Master Clock (MCK) Variation with PLLA**

| Core Clock/MCK (MHz) | VDDCORE Consumption (AMP1) | Total Consumption (AMP2) | Unit | Wakeup Time | Unit |
|---|---|---|---|---|---|
| 300/150 | 20 | 24 | mA | 0.85 | µs |
| 250/125 | 17 | 20 | | 1.05 | |
| 150/150 | 20 | 24 | | 0.9 | |
| 96/96 | 12.5 | 15 | | 1.4 | |
| 96/48 | 7.5 | 10 | | 2.5 | |
| 48/48 | 7 | 9.5 | | 2.8 | |
| 24/24 | 3.5 | 5 | | 5.6 | |
| 24/12 | 2 | 3 | | 10 | |
| 12/12 | 2 | 3 | | 11.2 | |
| 8/8 | 1.5 | 2 | | 16.8 | |
| 4/4 | 1.0 | 1.5 | | 32.9 | |
| 4/2 | 0.9 | 1 | | 60 | |
| 4/1 | 0.8 | 1 | | 112.6 | |

**Table 58-13. Typical Sleep Mode Current Consumption vs. Master Clock (MCK) Variation with Fast RC**

| Core Clock/MCK (MHz) | VDDCORE Consumption (AMP1) | Total Consumption (AMP2) | Unit | Wakeup Time | Unit |
|---|---|---|---|---|---|
| 12 | 2.0 | 2.0 | mA | 12 | µs |
| 8 | 1.5 | 1.5 | | 18 | |
| 4 | 1.0 | 1.1 | | 31 | |
| 2 | 0.8 | 0.8 | | 62 | |
| 1 | 0.6 | 0.7 | | 123 | |
| 0.5 | 0.6 | 0.6 | | 247 | |
| 0.25 | 0.5 | 0.5 | | 494 | |

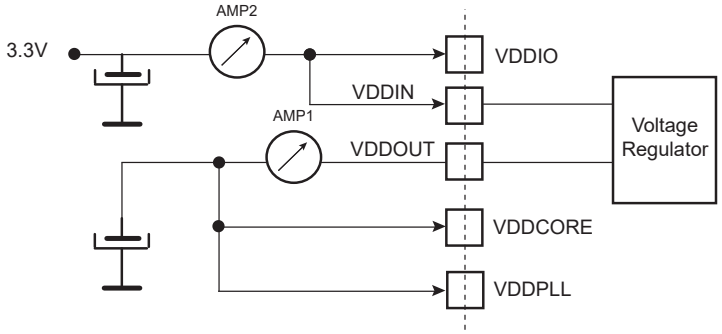
### 58.3.3 Wait Mode Current Consumption and Wake-up Time

The Wait mode configuration and measurements are defined as follows:

- Core clock and Master clock stopped
- Current measurement as shown below
- All peripheral clocks deactivated
- BOD disabled
- RTT enabled

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 58-7. Measurement Setup for Wait Mode**




The following tables give current consumption and wake-up time in Wait mode, where wake-up time is defined as the delay between the WKUP event and the execution of the first instruction..

**Table 58-14. Typical Current Consumption in Wait Mode at VDDIO=3.3V**

| Wait Mode Consumption | Typical Value | | | Unit |
|---|---|---|---|---|
| | at 25°C | | at 125°C | |
| Conditions | VDDOUT Consumption AMP1 | Total Consumption AMP2 | Total Consumption AMP2 | |
| No activity on the I/Os of the device | – | 0.5 | 15 | mA |

**Table 58-15. Typical Wake-up Time to Resume from Wait Mode**

| Conditions | Wake-up Time from Wait Mode | Unit |
|---|---|---|
| Resume from internal Flash with Cache enabled | 8.1 | µs |
| Resume from internal Flash with Cache disabled | 8.5 | |
| Resume from internal SRAM with Cache disabled | 8 | |

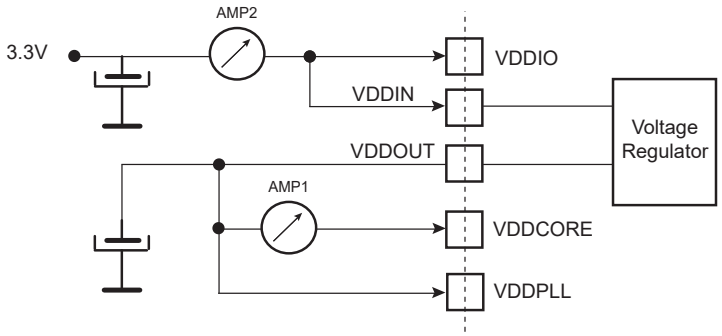
### 58.3.4 Active Mode Power Consumption

The conditions for measurement are defined as follows:

- $V_{DDIO}$ = $V_{DDIN}$ = 3.3V
- $V_{DDCORE}$ is provided by the Internal Voltage Regulator
- $T_A$ = 25°C
- Application running from Flash memory with 128-bit access mode
- All peripheral clocks are deactivated.
- Master Clock (MCK) running at various frequencies with PLLA or the fast RC oscillator.
- Current measurement on AMP1 (VDDCORE) and total current on AMP2

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 58-8. Active Mode Measurement Setup**




The following table gives current consumption in Active mode in typical conditions.

**Table 58-16. Typical Total Active Power Consumption with VDDCORE at 1.2V Running from Embedded Memory (AMP2)**

| Core Clock/MCK (MHz) | Cortex-M7 Running CoreMark | | | Unit |
|---|---|---|---|---|
| | Flash | | TCM | |
| | Cache Enable (CE) CoreMark = 4.9/MHz | Cache Disable (CD) CoreMark = 1.0/MHz | CoreMark = 5.0/MHz | |
| 300/150 | 90 | 57 | 83 | mA |
| 250/125 | 77 | 48 | 70 | |
| 150/150 | 52 | 40 | 48 | |
| 96/96 | 35 | 27 | 33 | |
| 96/48 | 31 | 20 | 28 | |
| 48/48 | 18 | 15 | 17 | |
| 24/24 | 10 | 8 | 9 | |
| 24/12 | 9 | 6 | 8 | |
| 12/12 | 5 | 4 | 5 | |
| 8/8 | 4 | 3 | 4 | |
| 4/4 | 2 | 2 | 2.5 | |
| 4/2 | 2 | 1.5 | 2 | |
| 4/1 | 1.5 | 1.5 | 1.5 | |
| 2/2 | 1.5 | 1.5 | 1.5 | |

**Note:** Flash Wait State (FWS) in EEFC_FMR is adjusted depending on core frequency.

## 58.4 Oscillator Characteristics

### 58.4.1 32 kHz RC Oscillator Characteristics

**Table 58-17. 32 kHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $f_{OSC}$ | Operating Frequency | – | 20 | 32 | 57 | kHz |

| .....continued | | | | | | |
|---|---|---|---|---|---|---|
| **Symbol** | **Parameter** | **Conditions** | **Min.** | **Typ.** | **Max.** | **Unit** |
| $t_{START}$ | Startup Time | – | – | – | 120 | µs |
| $I_{DDON}$ | Current Consumption | After startup time | – | 540 | – | nA |

### 58.4.2 4/8/12 MHz RC Oscillator

The 4/8/12 MHz RC oscillator is calibrated in production. This calibration can be read through the Get CALIB bit command (refer to the 22. Enhanced Embedded Flash Controller (EEFC)) and the frequency can be trimmed by software through the PMC.

**Table 58-18. 4/8/12 MHz RC Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $ACC_4$ | 4 MHz Total Accuracy(2) | 4 MHz output selected (see **Note 1**) | -35 | – | 46 | % |
| | | 4 MHz output selected at 25°C (see **Notes 1, 3**) | -1.2 | – | 0.8 | % |
| $ACC_8$ | 8 MHz Total Accuracy | 8 MHz output selected at 25°C (see **Notes 1, 3**) | -1.2 | – | 0.8 | % |
| $ACC_{12}$ | 12 MHz Total Accuracy | 12 MHz output selected at 25°C (see **Notes 1, 3**) | -1.6 | – | 0.8 | % |
| | Temp dependency | (see **Note 3**) | – | 0.07 | 0.12 | %/°C |
| $t_{START}$ | Startup Time | – | – | – | 20 | µs |

**Notes:**

1. Frequency range can be configured in the Supply Controller registers.
2. Not trimmed from factory.
3. After trimming at 25°C and VDDCORE = 1.2V.

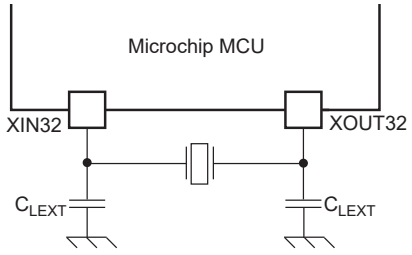
### 58.4.3 32.768 kHz Crystal Oscillator Characteristics

**Table 58-19. 32.768 kHz Crystal Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $1/(t_{CPXIN})$ | XIN32 Clock Frequency | (see **Note**) | – | – | 32 | kHz |
| $t_{CHXIN}$ | XIN32 Clock High Half-period | (see **Note**) | 15 | – | – | ns |
| $t_{CLXIN}$ | XIN32 Clock Low Half-period | (see **Note**) | 15 | – | – | ns |
| $V_{XIN_IL}$ | $V_{XIN}$ Input Low-level Voltage | (see **Note**) | Min of $V_{IL}$ for CLOCK pad | – | Max of $V_{IL}$ for CLOCK pad | V |
| $V_{XIN_IH}$ | $V_{XIN}$ Input High-level Voltage | (see **Note**) | Min of $V_{IH}$ for CLOCK pad | – | Max of $V_{IH}$ for CLOCK pad | V |
| $P_{ON}$ | Drive Level | – | – | – | 0.2 | µW |

**Note:** These characteristics apply only when the 32.768 kHz crystal oscillator is in Bypass mode.

**Figure 58-9. 32.768 kHz Crystal Oscillator Schematics**



$C_{LEXT} = 2 \times (C_{CRYSTAL} - C_{PARA} - C_{PCB})$

where $C_{PCB}$ is the capacitance of the printed circuit board (PCB) track layout from the crystal to the pin.

### 58.4.4 32.768 kHz Crystal Characteristics

**Table 58-20. 32.768 kHz Crystal Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| ESR | Equivalent Series Resistor | Crystal at 32.768 kHz | – | 50 | 100 | kOhm |
| $C_M$ | Motional Capacitance | Crystal at 32.768 kHz | 2 | – | 4 | fF |
| $C_{SHUNT}$ | Shunt Capacitance | Crystal at 32.768 kHz | 0.6 | – | 2 | pF |
| $C_{CRYSTAL}$ | Allowed Crystal Capacitance Load | From crystal specification | 6 | – | 12.5 | pF |

### 58.4.5 XIN32 Clock Characteristics in Bypass Mode

**Table 58-21. XIN32 Clock Characteristics in Bypass Mode**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $1/(t_{CPXIN})$ | XIN32 Clock Frequency | (see **Note**) | – | – | 32 | kHz |
| $t_{CHXIN}$ | XIN32 Clock High Half-period | (see **Note**) | 15 | – | – | ns |
| $t_{CLXIN}$ | XIN32 Clock Low Half-period | (see **Note**) | 15 | – | – | ns |
| $V_{XIN_IL}$ | $V_{XIN}$ Input Low-level Voltage | (see **Note**) | Min of $V_{IL}$ for CLOCK pad | – | Max of $V_{IL}$ for CLOCK pad | V |
| $V_{XIN_IH}$ | $V_{XIN}$ Input High-level Voltage | (see **Note**) | Min of $V_{IH}$ for CLOCK pad | – | Max of $V_{IH}$ for CLOCK pad | V |

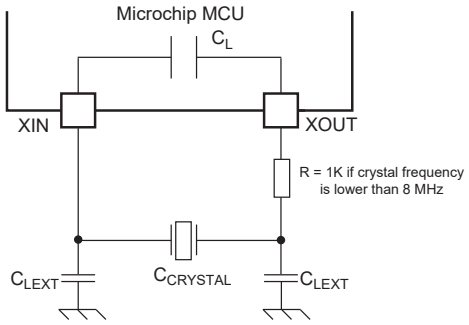
**Note:** These characteristics apply only when the 32.768 kHz crystal oscillator is in Bypass mode.

### 58.4.6 3 to 20 MHz Crystal Oscillator Characteristics

**Table 58-22. 3 to 20 MHz Crystal Oscillator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $f_{OSC}$ | Operating Frequency | Normal mode with crystal | 3 | – | 20 | MHz |
| $t_{START}$ | Startup Time | 3 MHz, $C_{SHUNT}$ = 3 pF | – | – | 40 | ms |
| | | 12 MHz, $C_{SHUNT}$ = 7 pF with $C_M$ = 1.6 fF | – | – | 6 | ms |
| | | 20 MHz, $C_{SHUNT}$ = 7 pF with $C_M$ = 1.6 fF | – | – | 5.7 | ms |
| $I_{DDON}$ | Current Consumption (on VDDIO) | 3 MHz | – | 230 | – | µA |
| | | 12 MHz | – | 390 | – | µA |
| | | 20 MHz | – | 450 | – | µA |
| $C_L$ | Internal Equivalent Load Capacitance | Integrated Load Capacitance ($X_{IN}$ and $X_{OUT}$ in series) | 7.5 | 9 | 10.5 | pF |
| $P_{ON}$ | Drive Level | 3 MHz | – | – | 15 | µW |
| | | 8 MHz | – | – | 30 | |
| | | 12 MHz, 20 MHz | – | – | 50 | |

**Figure 58-10. 3 to 20 MHz Crystal Oscillator Schematics**




$C_{LEXT} = 2 \times (C_{CRYSTAL} - C_L - C_{PCB})$

where, $C_{PCB}$ is the capacitance of the printed circuit board (PCB) track layout from the crystal to the pin.

### 58.4.7 3 to 20 MHz Crystal Characteristics

**Table 58-23. 3 to 20 MHz Crystal Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| ESR | Equivalent Series Resistor | Fundamental at 3 MHz | – | – | 150 | Ohm |
| | | Fundamental at 8 MHz | | | 140 | |
| | | Fundamental at 12 MHz | | | 120 | |
| | | Fundamental at 16 MHz | | | 80 | |
| | | Fundamental at 20 MHz | | | 50 | |
| $C_M$ | Motional capacitance | Fundamental at 3 MHz | 3 | – | 8 | fF |
| | | Fundamental at 8–20 MHz | 1.6 | – | 8 | |
| $C_{SHUNT}$ | Shunt capacitance | – | – | – | 7 | pF |
| $C_{CRYSTAL}$ | Allowed Crystal Capacitance Load | From crystal specification | 12.5 | – | 17.5 | pF |

### 58.4.8 3 to 20 MHz XIN Clock Input Characteristics in Bypass Mode

**Table 58-24. 3 to 20 MHz XIN Clock Input Characteristics in Bypass Mode**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $1/(t_{CPXIN})$ | XIN Clock Frequency | (see **Note 1**) | – | – | 20 | MHz |
| $t_{CHXIN}$ | XIN Clock High Half-period | (see **Note 1**) | 25 | – | – | ns |
| $t_{CLXIN}$ | XIN Clock Low Half-period | (see **Note 1**) | 25 | – | – | ns |
| $V_{XIN_IL}$ | $V_{XIN}$ Input Low-level Voltage | (see **Note 1**) | Min of $V_{IL}$ for CLOCK pad | – | Max of $V_{IL}$ for CLOCK pad | V |
| $V_{XIN_IH}$ | $V_{XIN}$ Input High-level Voltage | (see **Note 1**) | Min of $V_{IH}$ for CLOCK pad | – | Max of $V_{IH}$ for CLOCK pad | V |

**Note:** 1.These characteristics apply only when the 3–20 MHz crystal oscillator is in Bypass mode.

### 58.4.9 Crystal Oscillator Design Considerations

#### 58.4.9.1 Choosing a Crystal

When choosing a crystal for the 32768 Hz Slow Clock Oscillator or for the 3–20 MHz oscillator, several parameters must be taken into account. Important parameters between crystal and product specifications are as follows:

- Crystal Load Capacitance
  - The total capacitance loading the crystal, including the oscillator's internal parasitics and the PCB parasitics, must match the load capacitance for which the crystal's frequency is specified. Any mismatch in the load capacitance with respect to the crystal's specification will lead to inaccurate oscillation frequency
- Drive Level
  - Crystal drive level ≥ Oscillator Drive Level. Having a crystal drive level number lower than the oscillator specification may damage the crystal.
- Equivalent Series Resistor (ESR)
  - Crystal ESR ≤ Oscillator ESR Max. Having a crystal with ESR value higher than the oscillator may cause the oscillator to not start.
- Shunt Capacitance
  - Max. crystal shunt capacitance ≤ Oscillator Shunt Capacitance ($C_{SHUNT}$). Having a crystal with $C_{SHUNT}$ value higher than the oscillator may cause the oscillator to not start.

#### 58.4.9.2 Printed Circuit Board (PCB)

To minimize inductive and capacitive parasitics associated with XIN, XOUT, XIN32 and XOUT32 nets, it is recommended to route them as short as possible. Additionally, it is of prime importance to keep these nets away from noisy switching signals (clock, data, PWM, etc.). A good practice is to shield them with a quiet ground net to avoid coupling to neighboring signals.

## 58.5 PLLA Characteristics

**Table 58-25. PLLA Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input Frequency | – | 8 | – | 32 | MHz |
| $f_{OUT}$ | Output Frequency | – | 160 | – | 500 | MHz |
| $I_{PLL}$ | Current Consumption | Active mode at 160 MHz at 1.2V | – | 2.2 | 3 | mA |
| | | Active mode at 500 MHz at 1.2V | – | 8 | 12 | |
| $t_{START}$ | Startup Time | – | – | – | 300 | µs |

## 58.6 PLLUSB Characteristics

**Table 58-26. PLLUSB Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $f_{IN}$ | Input Frequency | – | – | 12 or 16 | – | MHz |
| $f_{OUT}$ | Output Frequency | – | – | 480 | – | MHz |
| $I_{PLLUSB}$ | Current Consumption | In Active mode, on VDDPLLUSB | – | 4.9 | 6.4 | mA |
| | | In Active mode, on VDDCORE | – | 0.4 | 1 | mA |
| $t_{START}$ | Startup Time | – | – | – | 50 | µs |

## 58.7 USB Transceiver Characteristics

The device conforms to all voltage, power, and timing characteristics and specifications as set forth in the USB 2.0 specification. Refer to the USB 2.0 specification for additional information.

© 2021 Microchip Technology Inc. and its subsidiaries

**Table 58-27. USB Transceiver Dynamic Power Consumption**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $I_{BIAS}$ | Bias Current Consumption on VBG | – | – | – | 12 | mA |
| $I_{VDDUTMII}$ | HS Transceiver Current Consumption | HS transmission | – | – | 44 | mA |
| | HS Transceiver Current Consumption | HS reception | – | – | 24 | mA |
| | LS / FS Transceiver Current Consumption | FS transmission 0m cable (see **Note 1**) | – | – | 5 | mA |
| | LS / FS Transceiver Current Consumption | FS transmission 5m cable (see **Note 1**) | – | – | 30 | mA |
| | LS / FS Transceiver Current Consumption | FS reception (see **Note 1**) | – | – | 1 | mA |
| $I_{VDDUTMIC}$ | Core | – | – | – | 10 | mA |

**Note:** 1. Including 1 mA due to pull-up or pull-down current consumption.

## 58.8 AFE Characteristics

Electrical data are in accordance with the specified operating temperature range, unless otherwise specified.

VREFP is the positive reference of the AFE. The VREFN pin must be connected to ground.

DAC1 and DAC0 provide an analog output voltage ($V_{DAC}$) in the range [0 : VREFP] with an accuracy equal to 10 bits. The DAC output voltage is single-ended and is used as a reference node by the sampling stage S/H0 and S/H1 (Sample-and-Hold PGA), relative to the single-ended input signal being sampled on the selected channel.

As a consequence, programming the DAC output voltage offers a capability to compensate for a DC offset on the input signal being sampled. DC offset compensation is effective in single-ended operation and is not effective in fully differential operation.

During fully differential operation, the DAC10 output voltage can be programmed at VREFP/2, by using the 10-bit code 512. The DAC value does not affect the AFE output code.

VREFP/2 on DAC0 and DAC1 is not automatically set and must be programed as the code 512 into the channel corresponding DAC0 and DAC1.

The following figures illustrate the architecture of the AFE in Single-ended and in Differential modes.

**Figure 58-11. Single-ended Mode AFE**

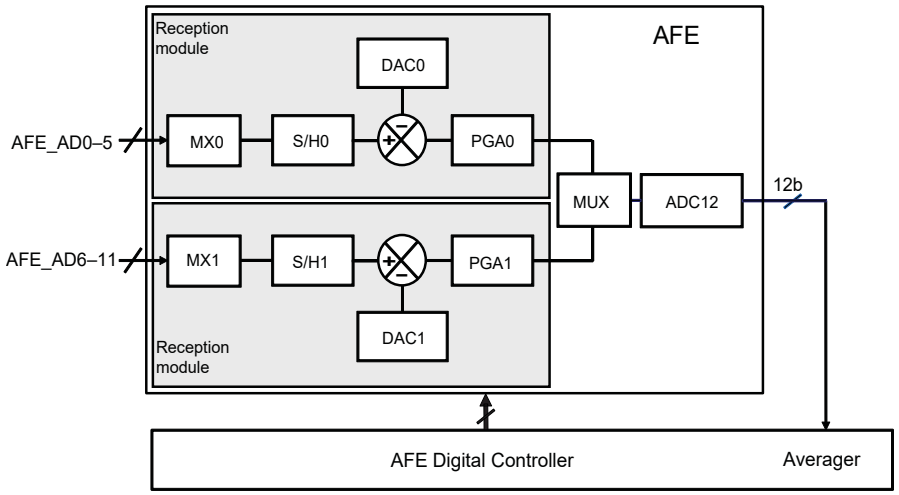
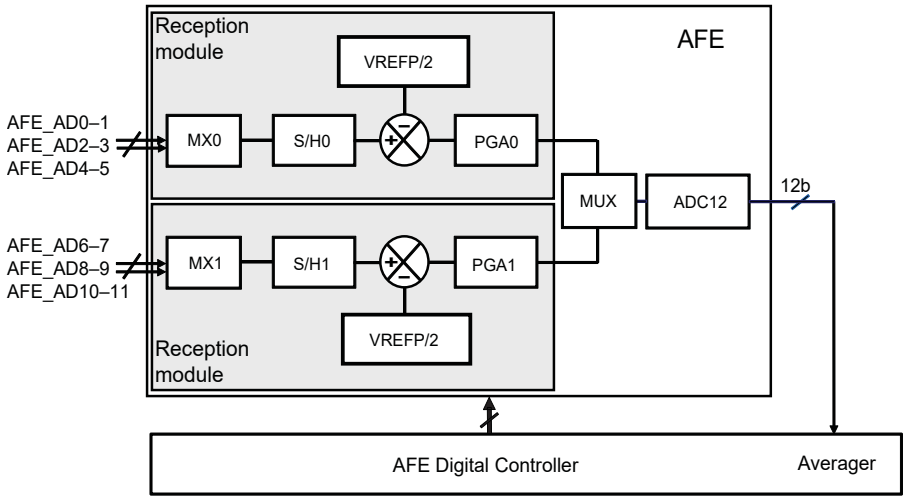


**Figure 58-12. Differential Mode AFE**




## 58.8.1 AFE Power Supply

### 58.8.1.1 Power Supply Characteristics

**Table 58-28. Power Supply Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $I_{VDDIN}$ | Analog Current Consumption | Sleep mode (see **Note 2**) | – | 2 | – | µA |
| | | Fast wake-up mode (see **Note 3**) | | 0.4 | | mA |
| | | Normal mode, single sampling | | 3.4 | | mA |
| | | Normal mode, dual sampling | | 4.2 | | mA |
| $I_{VDDCORE}$ | Digital Current Consumption | Sleep mode (see **Note 2**) | – | 1 | – | µA |
| | | Normal mode | | 80 | | |

**Notes:**

1. Current consumption is measured with AFEC_ACR.IBCTL=10.
2. In Sleep mode, the AFE core, the Sample and Hold and the internal reference operational amplifier are off.
3. In Fast Wake-up mode, only the AFE core is off.

### 58.8.1.2 ADC Bias Current

AFEC_ACR.IBCTL controls the ADC bias current with the nominal setting IBCTL = 10.

IBCTL = 10 is the mandatory configuration suitable for a sampling frequency of up to 1 MHz. If the sampling frequency is below 500 kHz, IBCTL = 01 can be used to reduce the current consumption.

If the sampling frequency is more than 1 MHz, then the setting must be IBCTL=11.

**Note:** The default value in the register is 01, and it must be modified according to the defined sampling frequency.

## 58.8.2 External Reference Voltage

$V_{VREFP}$ is an external reference voltage applied on the pin VREFP. The quality of the reference voltage $V_{VREFP}$ is critical to the performance of the AFE. A DC variation of the reference voltage $V_{VREFP}$ is converted to a gain error by the AFE. The noise generated by $V_{VREFP}$ is converted by the AFE to count noise.

© 2021 Microchip Technology Inc. and its subsidiaries

**Table 58-29. VREFP Electrical Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{VREFP}$ | Voltage Range | Full operational | 1.7 | – | VDDIN | V |
| | RMS Noise (see **Note 2**) | Bandwidth up to 1.74MHz VREFP=1.7V | – | – | 120 | µV |
| $R_{VREFP}$ | Input DC Impedance | AFE reference resistance bridge (see **Note 1**) | – | 4.7 | – | kOhm |
| Vin | Input Linear Range (see **Note 3**) | Operational Range | 2 | – | 98 | %VVREFP |
| $I_{VREFP}$ | Current | $V_{VREFP}$ = 3.3V | – | 0.8 | – | mA |

**Notes:**

1. When the AFE is in Sleep mode, the VREFP impedance has a minimum of 10 MOhm.
2. Requested noise on VREFP.
3. Electrical parameters specified inside the operational range. Exceeding this range can introduce additional INL error up to +/- 5 LSB and temperature dependency up to +/-10 LSB.

### 58.8.3 AFE Timings

**Table 58-30. AFE Timing Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $f_{AFE\ Clock}$ | Clock Frequency | – | 4 | 20 | 40 | MHz |
| $t_{AFE\ Clock}$ | Clock Period | – | 25 | 50 | 250 | ns |
| $f_S$ | Sampling Frequency (see **Note 1**) | – | – | – | 1.74 | MHz |
| $t_{START}$ | AFE Startup Time | Sleep mode to Normal mode | – | – | 4 | µs |
| | | Fast Wake-up mode to Normal mode | – | – | 2 | µs |

**Note:** 1.$f_s$ = 1 / $t_{AFE_conv}$ in Free Run mode; otherwise defined by the trigger timing.

### 58.8.4 AFE Transfer Function

The first operation of the AFE is a sampling function relative to $V_{DAC}$. $V_{DAC}$ is generated by an internal DAC0 or DAC1. All operations after the Sample-and-Hold are differential relative to an internal common mode voltage $V_{CM}$ = $V_{VREFP}$/2.

In Differential mode, the Sample-and-Hold common mode voltage is equal to $V_{DAC}$ = $V_{VREFP}$/2 (set by software DAC0 and DAC1 to code 512).

In Single-ended mode, $V_{DAC}$ is the common mode voltage. $V_{DAC}$ is the output of DAC0 or DAC1 voltage. All operations after the Sample-and-Hold are differential, including those in Single-ended mode.

For the formula example, the internal DAC0 or DAC1 is set for the code 512.

The DATA code in AFEC_CDR is up to 16-bit positive integer or two's complement (signed integer).

The code does not exceed 4095 when the field AFEC_EMR.RES=0 (12-bit mode, no averaging).

#### 58.8.4.1 Differential Mode (12-bit mode)

A differential input voltage $V_{IN}$ = $V_{INP}$ - $V_{INN}$ can be applied between two selected differential pins, e.g. AFE0_AD0 and AFE0_AD1.The ideal code $C_i$ is calculated by using the following formula and rounding the result to the nearest positive integer.

$$C_i = \frac{4096}{V_{\mathrm{VREFP}}} \times V_{\mathrm{IN}} \times \mathrm{Gain} + (2047)$$

For the other resolution defined by RES, the code $C_i$ is extended to the corresponding resolution.

The table below is a computation example for the above formula, where $V_{VREFP}$ = 3V.

**Table 58-31. Input Voltage Values in Differential Mode, Nonsigned Output**

| $C_i$ | | Gain | | |
|---|---|---|---|---|
| Signed | Nonsigned | 1 | 2 | 4 |
| -2048 | 0 | -3 | -1.5 | -0.75 |
| 0 | 2047 | 0 | 0 | 0 |
| 2047 | 4095 | 3 | 1.5 | 0.75 |

#### 58.8.4.2 Single-ended Mode (12-bit mode)

A single input voltage $V_{IN}$ can be applied to selected pins, e.g. AFE0_AD0 or AFE0_AD1. The ideal code $C_i$ is calculated using the following formula and rounding the result to the nearest positive integer.

The single-ended ideal code conversion formula is:

$$C_i = \frac{4096}{V_{\mathrm{VREFP}}} \times (V_{\mathrm{IN}} - V_{\mathrm{DAC}}) \times \mathrm{Gain} + 2047$$

For the other resolution defined by RES, the code $C_i$ is extended to the corresponding resolution.

The table below is a computation example for the above formula, where $V_{VREFP}$ = 3V:

**Table 58-32. Input Voltage Values in Single-ended Mode**

| $C_i$ | | Gain | | |
|---|---|---|---|---|
| Signed | Nonsigned | 1 | 2 | 4 |
| -2048 | 0 | 0 | 0.75 | 1.125 |
| 0 | 2047 | 1.5 | 1.5 | 1.5 |
| 2047 | 4095 | 3 | 2.25 | 1.875 |

#### 58.8.4.3 Example of LSB Computation

The LSB is relative to the analog scale $V_{VREFP}$.

The term LSB expresses the quantization step in volts, also used for one AFE code variation.

- Single-ended (SE) (ex: $V_{VREFP}$ = 3.0V)
  - Gain = 1, LSB = (3.0V / 4096) = 732 µV
  - Gain = 2, LSB = (1.5V / 4096) = 366 µV
  - Gain = 4, LSB = (750 mV / 4096) = 183 µV
- Differential (DIFF) (ex: $V_{VREFP}$ = 3.0V)
  - Gain = 1, LSB = (6.0V / 4096) = 1465 µV
  - Gain = 2, LSB = (3.0V / 4096) = 732 µV
  - Gain = 4, LSB = (1.5V / 4096) = 366 µV

The data include the AFE performances, as the PGA and AFE core cannot be separated. The temperature and voltage dependency are given as separate parameters.

#### 58.8.4.4 Gain and Offset Errors

For:

- a given gain error: $E_G$ (%)
- a given ideal code ($C_i$)
- a given offset error: $E_O$ (LSB of 12 bits)

in 12-bit mode, the actual code ($C_A$) is calculated using the following formula

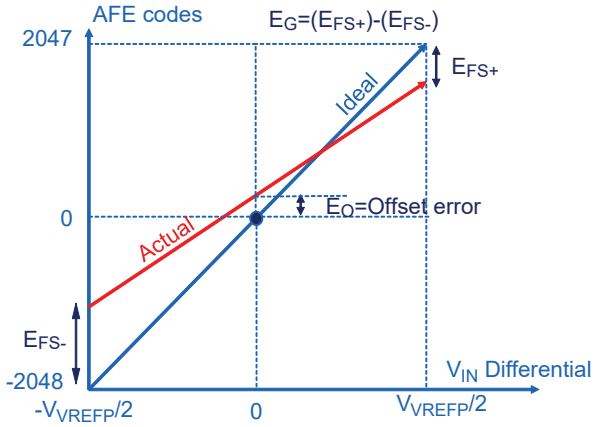
$$C_A = \left(1 + \frac{E_G}{100}\right) \times (C_i - 2047) + 2047 + E_O$$

For higher resolutions, the code can be extended to the corresponding resolution defined by RES.

#### 58.8.4.4.1 Differential Mode

In Differential mode, the offset is defined when the differential input voltage is zero.

**Figure 58-13. Gain and Offset Errors in Differential Mode**




where:

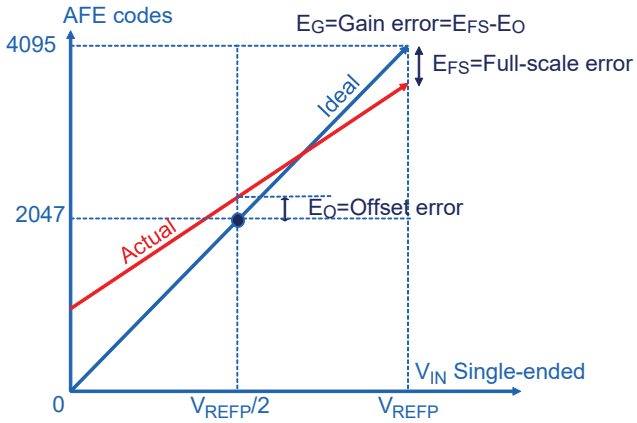
- Full-scale error $E_{FS}$ =($E_{FS+}$)-($E_{FS-}$), unit is LSB code
- Offset error $E_O$ is the offset error measured for $V_{IN}$=0V
- Gain error $E_G$=100 × $E_{FS}$ /4096, unit in %

The error values in the tables below include the sample-and-hold error as well as the PGA gain error.

#### 58.8.4.4.2 Single-ended Mode

The figure below illustrates the AFE output code relative to an input voltage $V_{IN}$ between 0V (Ground) and $V_{VREFP}$. The AFE is configured in Single-ended mode by connecting internally the negative differential input to $V_{VREFP}$/2. As the AFE continues to work internally in Differential mode, the offset is measured at $V_{VREFP}$/2. The offset at $V_{INP}$=0 can be computed using the transfer function and the corresponding $E_G$ and $E_O$.

**Figure 58-14. Gain and Offset Errors in Single-ended Mode**




where:

- Full-scale error $E_{FS}$ =($E_{FS+}$)-($E_{FS-}$), unit is LSB code
- Offset error $E_O$ is the offset error measured for $V_{REFP/2}$= 0V
- Gain error $E_G$=100 x $E_{FS}$ /4096, unit in %

The error values in the tables below include the DAC, the sample-and-hold error as well as the PGA gain error.

### 58.8.5 AFE Electrical Characteristics

**Table 58-33. AFE INL and DNL, $f_{AFE\ CLOCK}$ = < 20 MHz Maximum, IBCTL = 10**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Differential Mode** | | | | | | |
| INL | Integral Non-Linearity | Gain = 1 | -12 | ±1 | 14 | LSB |
| | | Gain = 2 | | ±1.5 | | |
| | | Gain = 4 | | ±1.8 | | |
| DNL | Differential Non-Linearity | – | -2 | ±0.6 | 2 | LSB |
| **Single-Ended Mode** | | | | | | |
| INL | Integral Non-Linearity | Gain = 1 | -14 | ±1.5 | 14 | LSB |
| | | Gain = 2 | | ±1.9 | | |
| | | Gain = 4 | | ±2.5 | | |
| DNL | Differential Non-Linearity | – | -2 | ±1.6 | 2 | LSB |

**Note:** INL/DNL given inside the linear range of the AFE: 5% to 95% of VREFP.

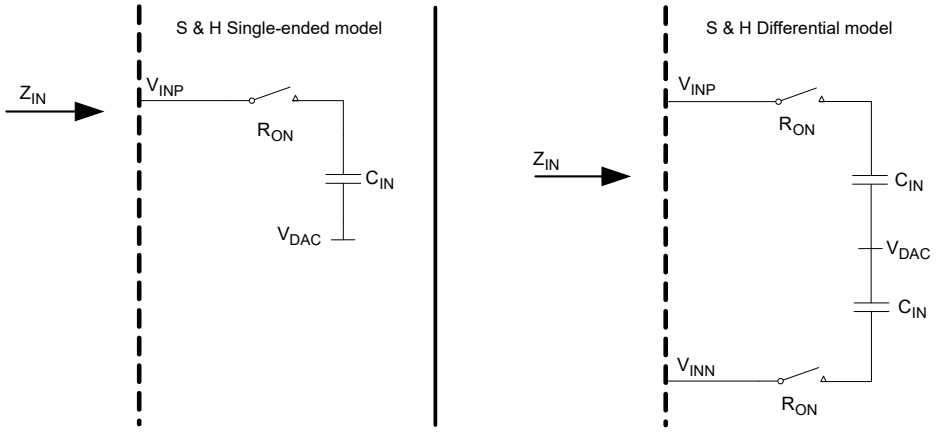
**Table 58-34. AFE Offset and Gain Error, $V_{VREFP}$ = 2.0V to 3.3V**

| Symbol | Parameter | Conditions | Min | Typ(1) | Max | Unit |
|---|---|---|---|---|---|---|
| **Differential Mode** | | | | | | |
| $E_O$ | Differential Offset Error (see **Note 1**) | Gain=1 | -20 | – | 20 | LSB |
| $E_G$ | Differential Gain Error | Gain=1 | -0.3 | 0 | 1 | % |
| | | Gain=2 | -0.3 | 0.3 | 2 | |
| | | Gain=4 | -0.3 | 0.7 | 5 | |
| **Single-Ended Mode** | | | | | | |
| $E_O$ | Single-ended Offset Error (see **Note 1**) | Gain=1 | -20 | – | 20 | LSB |
| $E_G$ | Single-ended Gain Error | Gain=1 | -0.3 | 1 | 2.5 | % |
| | | Gain=2 | -0.3 | 1.7 | 5 | |
| | | Gain=4 | -0.3 | 2.2 | 6.5 | |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 58.8.6 AFE Channel Input Impedance

**Figure 58-15. Input Channel Model**




where:

- $Z_{IN}$ is input impedance in Single-ended or Differential mode
- $C_{IN}$ = 2 to 8 pF ±20% depending on the gain value and mode (SE or DIFF); temperature dependency is negligible
- $R_{ON}$ is typical 2 kΩ and 8 kΩ max (worst case process and high temperature)

The following formula is used to calculate input impedance:

$$Z_{\mathrm{IN}} = \frac{1}{f_S \times C_{\mathrm{IN}}}$$

where:

- $f_S$ is the sampling frequency of the AFE channel
- Typ values are used to compute AFE input impedance $Z_{IN}$

**Table 58-35. Input Capacitance ($C_{IN}$) Values**

| Gain Selection | Single-ended | Differential | Unit |
|---|---|---|---|
| 1 | 2 | 2 | pF |
| 2 | 4 | 4 | |
| 4 | 8 | 8 | |

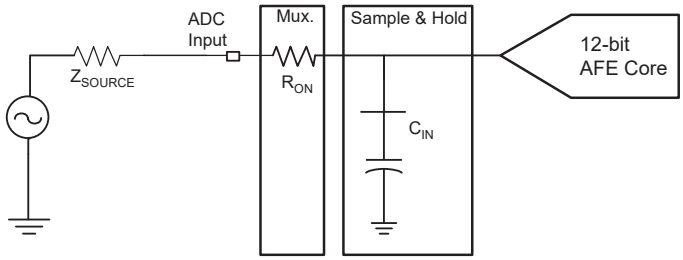
**Table 58-36. $Z_{IN}$ Input Impedance**

| $f_S$ (MHz) | 1 | 0.5 | 0.25 | 0.125 | 0.0625 | 0.03125 | 0.015625 | 0.007813 |
|---|---|---|---|---|---|---|---|---|
| $C_{IN}$ = 2 pF | | | | | | | | |
| $Z_{IN}$ (MΩ) | 0.5 | 1 | 2 | 4 | 8 | 16 | 32 | 64 |
| $C_{IN}$ = 4 pF | | | | | | | | |
| $Z_{IN}$ (MΩ) | 0.25 | 0.5 | 1 | 2 | 4 | 8 | 16 | 32 |
| $C_{IN}$ = 8 pF | | | | | | | | |
| $Z_{IN}$ (MΩ) | 0.125 | 0.25 | 0.5 | 1 | 2 | 4 | 8 | 16 |

#### 58.8.6.1 Track and Hold Time versus Source Output Impedance

The figure below shows a simplified acquisition path.

**Figure 58-16. Simplified Acquisition Path**




During the tracking phase, the AFE tracks the input signal during the tracking time shown below:

$t_{TRACK} = n \times C_{IN} \times (R_{ON}+Z_{SOURCE})/1000$

- Tracking time expressed in ns and $Z_{SOURCE}$ expressed in Ω.
- n depends on the expected accuracy
- $R_{ON}$= 2 kOhm

**Table 58-37. Number of Tau:n**

| Resolution (bits) | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|
| RES | 0 | 2 | 3 | 4 | 5 |
| n | 8 | 9 | 10 | 11 | 12 |

The AFEC already includes a tracking time of 15 $t_{AFE\ Clock}$.

### 58.8.6.2 AFE DAC Offset Compensation

**Table 58-38. DAC Static Performances (see Note 1)**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| N | Resolution (see **Note 2**) | – | – | 9 | 10 | LSB |
| INL | Integral Non Linearity | – | -3 | ±0.7 | 2.5 | LSB |
| DNL | Differential Non Linearity | – | -3.5 | ±0.5 | 3 | LSB |

**Notes:**

1. DAC Offset is included in the AFE EO performances.
2. 10 bits LSB relative to $V_{REFP}$ scale, LSB = $V_{VREFP}$ / 210 = 2.93 mV, with $V_{VREFP}$ = 3V.

## 58.9 Analog Comparator Characteristics

**Table 58-39. Analog Comparator Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{IR}$ | Input Voltage Range | – | GND + 0.2 | – | VDDIN - 0.2 | V |
| $V_{IO}$ | Input Offset Voltage | Comparator only | – | – | 10 | mV |
| $I_{VDDIN}$ | Current Consumption (VDDIN) | Low-power option (ACC_ACR.ISEL = 0) | – | 20 | – | µA |
| | | High-speed option (ACC_ACR.ISEL = 1) | – | 120 | – | |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | | | | | | |
|---|---|---|---|---|---|---|
| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
| $V_{hys}$ | Hysteresis | ACC_ACR.HYST = 1 or 2 ACC_ACR.ISEL = 0 | – | 20 | – | mV |
| | | ACC_ACR.HYST = 3 ACC_ACR.ISEL = 0 | – | 40 | – | |
| | | ACC_ACR.HYST = 1 or 2 ACC_ACR.ISEL = 1 | – | 25 | – | mV |
| | | ACC_ACR.HYST = 3 ACC_ACR.ISEL = 1 | – | 45 | – | |
| $t_S$ | Settling Time | Overdrive > 100 mV (ACC_ACR.ISEL = 0) | – | – | 1.5 | µs |
| | | Overdrive > 100 mV (ACC_ACR.ISEL = 1) | – | – | 0.15 | |

## 58.10 Temperature Sensor

The temperature sensor is connected to channel 11 of the AFE0.

The temperature sensor provides an output voltage ($V_{TEMP}$) that is proportional to absolute temperature (PTAT).

Improvement of the raw performance of the temperature sensor acquisition can be achieved by performing a single temperature point calibration to remove the initial inaccuracies ($V_{TEMP}$ and ADC offsets).

**Table 58-40. Temperature Sensor Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{TEMP}$ | Output Voltage via AD11 | $T_A$ = 25°C | 0.64 | 0.72 | 0.8 | V |
| $dV_{TEMP}/dT$ | Temperature Sensitivity (Slope Voltage versus Temperature) | – | 2.06 | 2.33 | 2.60 | mV/°C |
| $t_S$ | VTEMP Settling Time | When $V_{TEMP}$ is sampled by the AFEC, the required track-and-hold time to ensure 1°C accurate settling | – | – | 1 | µs |
| – | Temperature Accuracy | After offset calibration over $T_A$ range | -10 | – | 10 | °C |
| $t_{START}$ | Startup Time | – | – | – | 30 | µs |
| $I_{VDDIN}$ | Current Consumption | – | – | 130 | 270 | µA |

**Note:** AFE Gain Error and Offset error considered calibrated. This calibration at ambient temperature is not a feature of the product and is performed by the user's application.

© 2021 Microchip Technology Inc. and its subsidiaries

## 58.11 12-bit DAC Characteristics

**Table 58-41. Analog Power Supply Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $I_{VDDIN}$ | Current Consumption | Sleep mode (Clock OFF) | – | 10 | – | µA |
| | | Normal mode with one output on,<br>DACC_ACR.IBCTLCHx =3 (see **Note 1**)<br>FS = 1 MSps, no $R_{LOAD}$, $V_{DDIN}$ = 3.3V | – | 200 | 800 | |
| | | Normal mode with one output on,<br>DACC_ACR.IBCTLCHx =1 (see **Note 1**)<br>FS = 500 KSps, no $R_{LOAD}$, $V_{DDIN}$ = 3.3V | – | 100 | 400 | |
| | | Bypass mode (output buffer off) with one output on,<br>DACC_ACR.IBCTLCHx =0 (see **Note 1**)<br>FS = 500 KSps, no $R_{LOAD}$, $V_{DDIN}$ = 3.3V | – | 10 | 30 | |
| PSRR | Power Supply RejectionRatio ($V_{DDIN}$) | $V_{DDIN}$ ±10 mV<br>Up to 10 kHz | – | 70 | – | dB |

**Note:** 1. The maximum conversion rate versus the configuration of DACC_ACR.IBCTL is shown in the following table.

**Table 58-42. Maximum Conversion Rate vs. Configuration of DACC_ACR.IBCTL**

| DACC_ACR.IBCTLCHx | Maximum Conversion Rate |
|---|---|
| 0 | Bypass |
| 1 | 500 ks/s |
| 2 | N/A |
| 3 | 1 Ms/s |

**Table 58-43. Voltage Reference**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $V_{VREFP}$ | Positive Voltage Reference | Externally decoupled 1 µF | 2.0 | – | $V_{DDIN}$ | V |
| $I_{VREFP}$ | DC Current on VREFP | – | – | 2.5 | – | µA |

**Note:** $V_{REFP}$ is the positive reference shared with AFE and may have a different value for AFE. Refer to the AFE electrical characteristics if AFE is used. The $V_{REFN}$ pin must be connected to ground.

**Table 58-44. DAC Clock**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $f_{DAC}$ | DAC Clock Frequency | – | – | – | 12 | MHz |
| $f_S$ | Sampling Frequency | – | – | $f_{DAC}$ / 12 | – | MHz |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Table 58-45. Static Performance Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| INL | Integral Non-linearity (see **Note 1**) | No $R_{LOAD}$ | -10 | ±2 | 10 | LSB |
| | | $C_{LOAD}$ = 50 pF | | | | |
| | | DACC_ACR.IBTLCHx = 3 | | | | |
| DNL | Differential Non-linearity (see **Note 1**) | No $R_{LOAD}$ | -4 | ±2 | 4 | LSB |
| | | $C_{LOAD}$ = 50 pF | | | | |
| | | DACC_ACR.IBTLCHx = 3 | | | | |
| $E_O$ | Offset Error (see **Note 2**) | – | -8 | 1 | 8 | mV |
| $E_G$ | Gain Error | No $R_{LOAD}$ | -1 | – | 1 | %.FSR |
| | | $C_{LOAD}$ = 50 pF | | | | |
| | | DACC_ACR.IBTLCHx = 3 | | | | |

**Notes:**

1. Best-fit Curve from 0x080 to 0xF7F.
2. Difference between DACx at 0x800 and $V_{VREFP}/2$.

**Table 58-46. Dynamic Performance Characteristics**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $t_{START}$ | Startup Time | From DAC on (CHER.CHx) to DAC ready to convert (CHSR.DACRDYx) | – | 10 | – | µs |
| $t_S$ | Settling Time Code to Code; i.e., code(n-1) to code(n) ± 0.5 LSB | $R_{LOAD}$ = 5 Kohm<br>$C_{LOAD}$ = 50 pF | – | 0.5 | – | µs |
| | Settling Time Full-scale; i.e., 0x000 to 0xFFF ±0.5 LSB | DACC_ACR.IBCTLCHx = 3 | – | 1 | – | |
| | | FS = 1 MSps | | | | |
| Slew Rate | – | – | – | 3 | – | V/µs |

**Table 58-47. Analog Outputs**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $R_{LOAD}$ | Output Resistor Load | Output load resistor | 5 | – | – | kOhm |
| $C_{LOAD}$ | Output Capacitor Load | Output load capacitor | – | – | 50 | pF |
| $V_{DACx_MIN}$ | Minimum Output Voltage on DACx | Code = 0x000 No $R_{LOAD}$, $C_{LOAD}$ = 50 pF, DACC_ACR.IBCTLCHx =3 | – | 0.1 | 0.5 | %. $V_{VREFP}$ |
| $V_{DACx_MAX}$ | Maximum Output Voltage on DACx | Code = 0xFFF No $R_{LOAD}$ $C_{LOAD}$ = 50 pF, DACC_ACR.IBCTLCHx =3 | 99.5 | 99.9 | – | %. $V_{VREFP}$ |
| FSR | Full Scale Range | Code = 0x000 to 0xFFF No $R_{LOAD}$ $C_{LOAD}$ = 50 pF, DACC_ACR.IBCTLCHx =3 | 99 | 99.8 | – | %. $V_{VREFP}$ |

© 2021 Microchip Technology Inc. and its subsidiaries

| .....continued | | | | | | |
|---|---|---|---|---|---|---|
| **Symbol** | **Parameter** | **Conditions** | **Min.** | **Typ.** | **Max.** | **Unit** |
| $R_{OUT}$ | DAC Output Resistor | 0.3 < $V_{DACx}$ < $V_{DDIN}$ -0.3V, DACC_ACR.IBCTLCHx =3, $R_{LOAD}$ = 5 KOhm | – | 15 | – | Ohm |
| | | $V_{DACx}$ > $V_{DDIN}$ -0.3V, DACC_ACR.IBCTLCHx =3, $R_{LOAD}$ = 5 kOhm | – | 550 | – | Ohm |
| | | $V_{DACx}$ < 0.3V, DACC_ACR.IBCTLCHx = 3, $_{RLOAD}$= 5 kOhm | – | 550 | – | Ohm |
| | | $V_{DACx}$ = $V_{VREFP}$/2, DACC_ACR.IBCTLCHx = 0 (Bypass mode, buffer off), No $R_{LOAD}$ | – | 300 | – | kOhm |

## 58.12 Embedded Flash Characteristics

**Table 58-48. Flash Characteristics**

| **Parameter** | **Conditions** | | **Min** | **Typ** | **Max** | **Unit** |
|---|---|---|---|---|---|---|
| ERASE Line Assertion Time | – | | 230 | – | – | ms |
| Program Cycle Time | Write Page | | – | 1.5 | – | ms |
| | Erase Page | | – | 10 | 50 | ms |
| | Erase Small Sector (8 Kbytes) | | – | 80 | 200 | ms |
| | Erase Larger Sector (112 or 128 Kbytes) | | | 800 | 1500 | ms |
| Full Chip Erase | 512 Kbytes | | – | 3 | 6 | s |
| | 1 Mbytes | | – | 6 | 12 | s |
| | 2 Mbytes | | – | 13 | 24 | s |
| Data Retention | At $T_A$ = 95°C, after 1K cycles (see **Note 1**) | | 62 | – | – | Years |
| | At $T_A$ = 110°C, after 1K cycles (see **Note 1**) | | 26 | – | – | |
| | At $T_A$ = 125°C, after 1K cycles (see **Note 1**) | | 12 | – | – | |
| Endurance | Write/Erase cycles per page, block or sector within temperature range | | 10K | | | Cycles |
| Flash Active Current | Random 128-bit read at maximum frequency at 25°C | on $V_{DDCORE}$ =1.2V | – | 16 | 20 | mA |
| | | on $V_{DDIO}$ | – | 2 | 10 | |
| | Program at 25°C | on $V_{DDCORE}$ =1.2V | – | 2 | 3 | |
| | | on $V_{DDIO}$ | – | 8 | 12 | |
| | Erase at 25°C | on $V_{DDCORE}$ =1.2V | – | 2 | 2 | |
| | | on $V_{DDIO}$ | – | 8 | 12 | |

**Note:**

1. Calculation taking into account $R_{TH}$ and Ea=0.7eV.

Maximum operating frequencies are shown in the following table, but are limited by the Embedded Flash access time when the processor is fetching code out of it. These tables provide the device maximum operating frequency defined by the field FWS of the EEFC_FMR register. This field defines the number of wait states required to access the Embedded Flash Memory.

**Table 58-49. Embedded Flash Wait States for Worst-Case Conditions**

| FWS | Read Operations | Maximum Operating Frequency (MHz) - VDDIO 3.0V |
|---|---|---|
| 0 | 1 cycle | 23 |
| 1 | 2 cycles | 46 |
| 2 | 3 cycles | 69 |
| 3 | 4 cycles | 92 |
| 4 | 5 cycles | 115 |
| 5 | 6 cycles | 138 |
| 6 | 7 cycles | 150 |

## 58.13 Timings for STH Conditions

### 58.13.1 AC Characteristics

#### 58.13.1.1 Processor Clock Characteristics

**Table 58-50. Processor Clock Waveform Parameters**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $1/(t_{CPPCK})$ | Processor Clock Frequency | Worst case | – | 300 | MHz |

#### 58.13.1.2 Master Clock Characteristics

**Table 58-51. Master Clock Waveform Parameters**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $1/(t_{CPMCK})$ | Master Clock Frequency | Worst case | – | 150 | MHz |

#### 58.13.1.3 I/O Characteristics

Criteria used to define the maximum frequency of the I/Os:

- Output duty cycle (40%-60%)
- Minimum output swing: 100 mV to $V_{DDIO}$ - 100 mV
- Addition of rising and falling time inferior to 75% of the period

**Table 58-52. I/O Characteristics**

| Symbol | Parameter | Conditions | | | Min | Max | Unit |
|---|---|---|---|---|---|---|---|
| | | Load | $V_{DDIO}$ | Drive Level | | | |
| FreqMax1 | Pin Group 1 (1) Maximum output frequency | 10 pF | 3.0V | Low | – | 65 | MHz |
| | | | | High | – | 115 | |
| | | 25 pF | | Low | – | 28 | |
| | | | | High | – | 55 | |
| $PulseminH_1$ | Pin Group 1 (1) High Level Pulse Width | 10 pF | 3.0V | High | 6.1 | 9.2 | ns |
| $PulseminL_1$ | Pin Group 1 (1) Low Level Pulse Width | 10 pF | 3.0V | High | 6.1 | 9.2 | ns |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | | | | | | | |
|---|---|---|---|---|---|---|---|
| Symbol | Parameter | Conditions | | | Min | Max | Unit |
| | | Load | $V_{DDIO}$ | Drive Level | | | |
| FreqMax2 | Pin Group 2 (2)Maximum output frequency | 10 pF | 3.0V | High | – | 125 | MHz |
| | | | | Low | – | 100 | |
| $PulseminH_2$ | Pin Group 2 (2) High Level Pulse Width | 10 pF | 3.0V | High | 3.4 | 4.1 | ns |
| $PulseminL_2$ | Pin Group 2 (2) Low Level Pulse Width | 10 pF | 3.0V | High | 3.4 | 4.1 | ns |
| FreqMax3 | Pin Group3(3) Maximum output frequency | 30 pF | 3.0V | High | – | 75 | MHz |
| | | | | Low | – | 50 | |
| $PulseminH_3$ | Pin Group 3 (3) High Level Pulse Width | 30 pF | 3.0V | High | 6.0 | 7.3 | ns |
| $PulseminL_3$ | Pin Group 3 (3) Low Level Pulse Width | 30 pF | | High | 6.0 | 7.3 | ns |
| FreqMax4 | Pin Group 4(4) Maximum output frequency | 40 pF | 3.0V | – | – | 51 | MHz |
| $PulseminH_4$ | Pin Group 4(4) High Level Pulse Width | 40 pF | 3.0V | – | 7.8 | 11.2 | ns |
| $PulseminL_4$ | Pin Group 4(4) Low Level Pulse Width | 40 pF | 3.0V | – | 7.8 | 11.2 | ns |

**Notes:**

1. Pin Group 1 = GPIO, CLOCK
2. Pin Group 2 = GPIO_CLK
3. Pin Group 3 = GPIO_AD
4. Pin Group 4 = GPIO_MLB

### 58.13.1.4 MediaLB Characteristics

The system has been constrained to achieve the timings in 256×Fs and 512×Fs in compliance with the MediaLB (MLB) specification.

Note: 1024×Fs timings are achieved under STH conditions only.

### 58.13.1.5 QSPI Characteristics

**Figure 58-17. QSPI Master Mode with (CPOL= NCPHA = 0) or (CPOL= NCPHA= 1)**

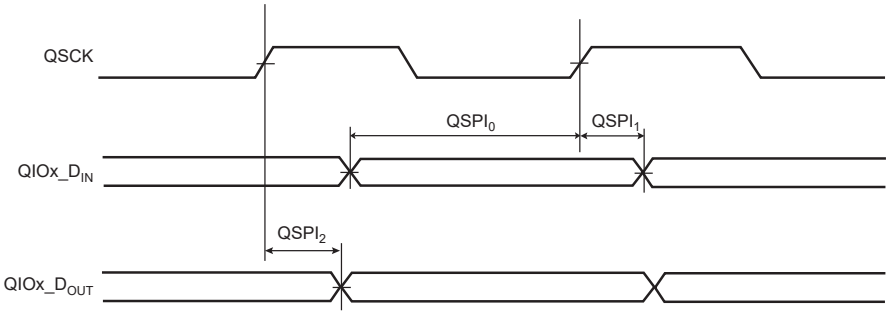
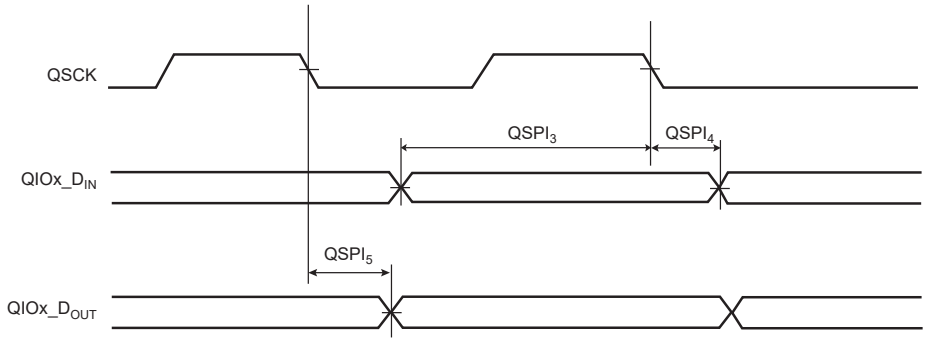


**Figure 58-18. QSPI Master Mode with (CPOL = 0 and NCPHA=1) or (CPOL=1 and NCPHA= 0)**




#### 58.13.1.5.1 Maximum QSPI Frequency

The following sections provide maximum QSPI frequency in master read and write modes.

**Master Write Mode**

The QSPI sends data to a slave device only, for example, an LCD. The limit is given by $QSPI_2$ (or $QSPI_5$) timing. Because it gives a maximum frequency above the maximum pad speed (Refer to the I/O Characteristics), the maximum QSPI frequency is the one from the pad.

**Master Read Mode**

$$f_{\mathrm{QSCK}}\mathrm{max} = \frac{1}{QSPI_0(\mathrm{or}\ QSPI_3) + t_{\mathrm{VALID}}}$$

$t_{VALID}$ is the slave time response to output data after detecting a QSCK edge.

For a QSPI slave device with $t_{VALID}$ (or $t_V$) = 12 ns, $f_{QSCK}$max = 66 MHz at VDDIO = 3.3V.

For a QSPI Flash memory device with $t_{VALID}$ (or $t_V$) = 6 ns, the formula returns a value of 112 MHz. In worst case conditions, this exceeds 66 MHz, which is the maximum allowed frequency of the QSPI master. In this case, the limitation is due to the controller and not the slave.

#### 58.13.1.5.2 QSPI Timings

Timings are given in the following domains:

- 3.3V domain: VDDIO from 2.85V to 3.6V, maximum external capacitor = 40 pF.

**Table 58-53. QSPI Timings**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $QSPI_0$ | QIOx data in to QSCK rising edge (input setup time) | 3.3V domain | 2.5 | – | ns |
| $QSPI_1$ | QIOx data in to QSCK rising edge (input hold time) | 3.3V domain | 0 | – | ns |
| $QSPI_2$ | QSCK rising edge to QIOx data out valid | 3.3V domain | -1.3 | 1.9 | ns |
| $QSPI_3$ | QIOx data in to QSCK falling edge (input setup time) | 3.3V domain | 2.9 | – | ns |
| $QSPI_4$ | QIOx data in to QSCK falling edge(input hold time) | 3.3V domain | 0 | – | ns |
| $QSPI_5$ | QSCK falling edge to QIOx data out valid | 3.3V domain | -1.6 | 1.8 | ns |

Timings are given for the 3.3V domain, with VDDIO from 2.85V to 3.6V, maximum external capacitor = 40 pF.

### 58.13.1.6 SPI Characteristics

In the figures below, the MOSI line shifting edge is represented with a hold time equal to 0. However, it is important to note that for this device, the MISO line is sampled prior to the MOSI line shifting edge. As shown further below, the device sampling point extends the propagation delay ($t_p$) for slave and routing delays to more than half the SPI clock period, whereas the common sampling point allows only less than half the SPI clock period.

As an example, an SPI Slave working in Mode 0 can be safely driven if the SPI Master is configured in Mode 0.

**Figure 58-19. MISO Capture in Master Mode**

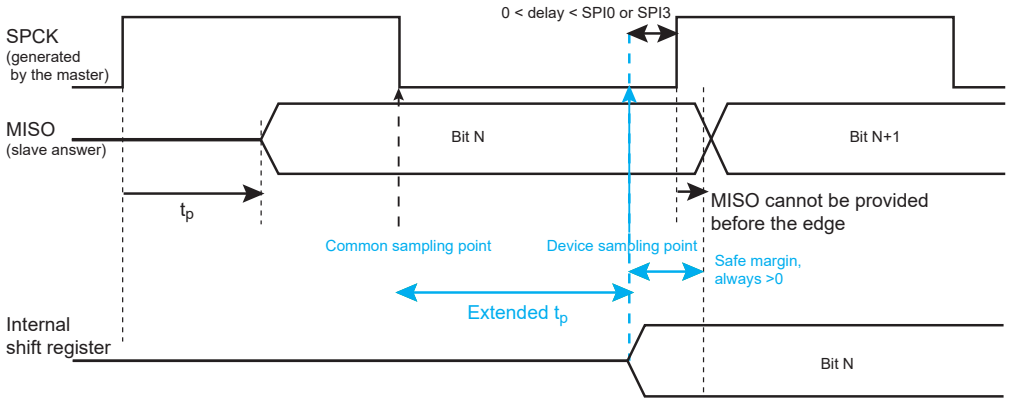



**Figure 58-20. SPI Master Mode with (CPOL= NCPHA = 0) or (CPOL= NCPHA= 1)**

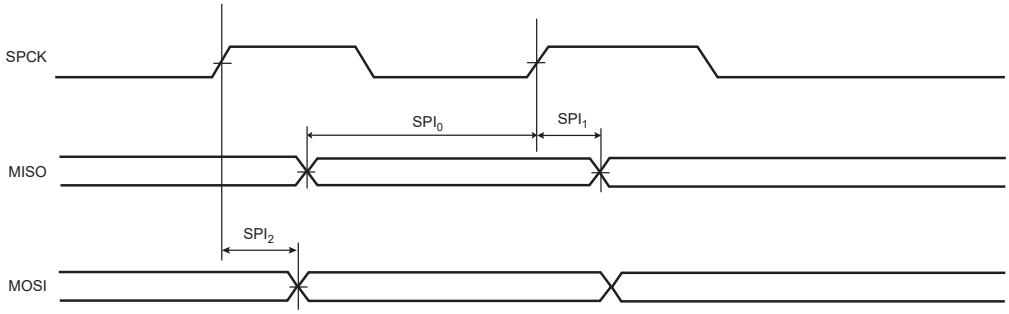



**Figure 58-21. SPI Master Mode with (CPOL = 0 and NCPHA=1) or (CPOL=1 and NCPHA= 0)**

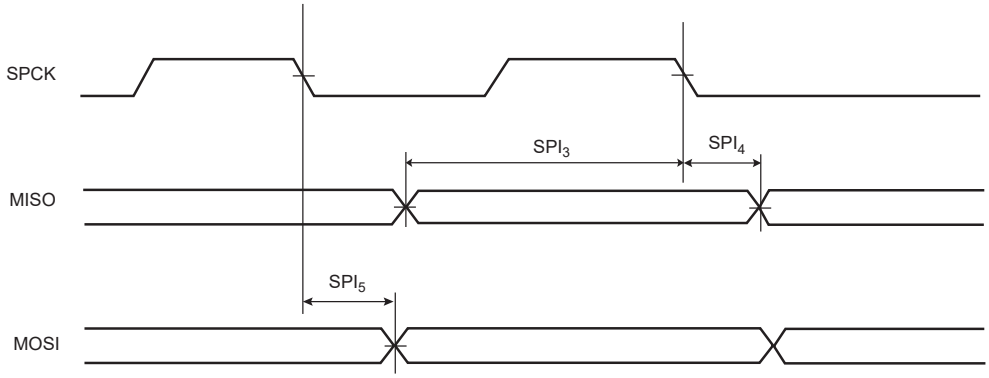


**Figure 58-22. SPI Slave Mode with (CPOL=0 and NCPHA=1) or (CPOL=1 and NCPHA=0)**

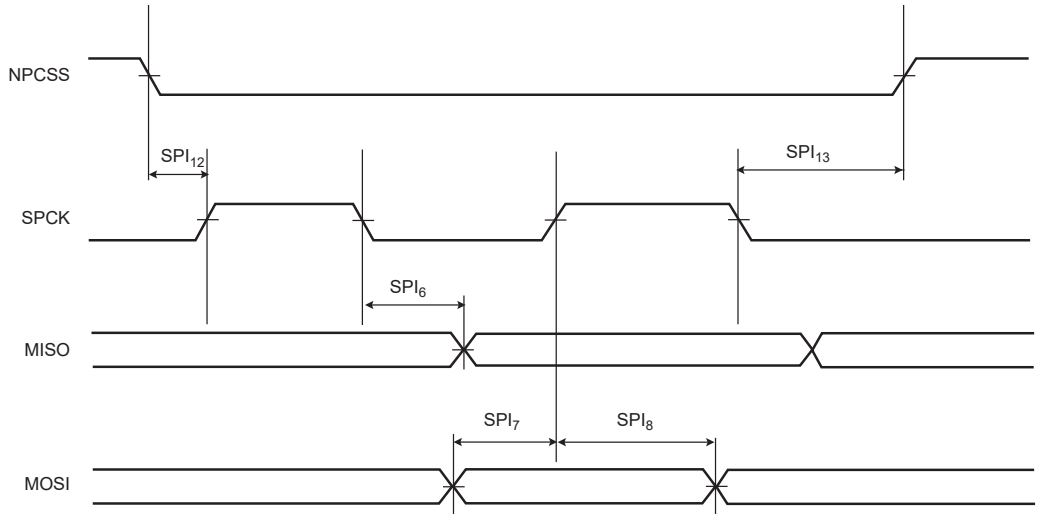
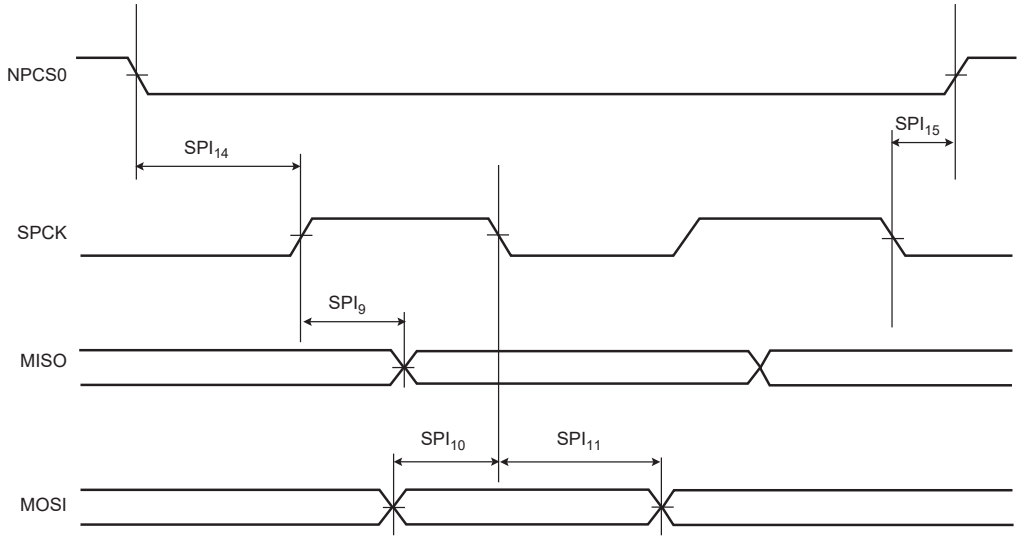



**Figure 58-23. SPI Slave Mode with (CPOL = NCPHA = 0) or (CPOL= NCPHA= 1)**




#### 58.13.1.6.1 Maximum SPI Frequency

The following formulas give maximum SPI frequency in Master read and write modes and in Slave read and write modes.

**Master Write Mode**

The SPI sends data to a slave device only, e.g. an LCD. The limit is given by $SPI_2$ (or $SPI_5$) timing. Since it gives a maximum frequency above the maximum pad speed (see I/O Characteristics), the max SPI frequency is the one from the pad.

**Master Read Mode**

$$f_{SPCK}\max = \frac{1}{SPI_0(\text{or } SPI_3) + t_{valid}}$$

$t_{valid}$ is the slave time response to output data after detecting an SPCK edge.

For a nonvolatile memory with $t_{valid}$ (or $t_v$) = 5 ns, $f_{SPCK}$max = 57 MHz at $V_{DDIO}$ = 3.3V.

**Slave Read Mode**

In slave mode, SPCK is the input clock for the SPI. The max SPCK frequency is given by setup and hold timings $SPI_7$/$SPI_8$(or $SPI_{10}$/$SPI_{11}$). Since this gives a frequency well above the pad limit, the limit in slave read mode is given by SPCK pad.

**Slave Write Mode**

$$f_{SPCK}\max = \frac{1}{2x(SPI_{6max}(\text{or } SPI_{9max}) + t_{setup})}$$

$t_{setup}$ is the setup time from the master before sampling data.

#### 58.13.1.6.2 SPI Timings

Timings are given for the 3.3V domain, with $V_{DDIO}$ from 2.85V to 3.6V, maximum external capacitor = 40 pF.

**Table 58-54. SPI Timings**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $SPI_0$ | MISO Setup time before SPCK rises (master) | 3.3V domain | 12.4 | – | ns |
| $SPI_1$ | MISO Hold time after SPCK rises (master) | 3.3V domain | 0 | – | ns |
| $SPI_2$ | SPCK rising to MOSI Delay (master) | 3.3V domain | -3.7 | 2.2 | ns |
| $SPI_3$ | MISO Setup time before SPCK falls (master) | 3.3V domain | 12.6 | – | ns |
| $SPI_4$ | MISO Hold time after SPCK falls (master) | 3.3V domain | 0 | – | ns |
| $SPI_5$ | SPCK falling to MOSI Delay (master) | 3.3V domain | -3.6 | 2.0 | ns |
| $SPI_6$ | SPCK falling to MISO Delay (slave) | 3.3V domain | 3.0 | 11.9 | ns |
| $SPI_7$ | MOSI Setup time before SPCK rises (slave) | 3.3V domain | 1.2 | – | ns |
| $SPI_8$ | MOSI Hold time after SPCK rises (slave) | 3.3V domain | 0.6 | – | ns |
| $SPI_9$ | SPCK rising to MISO Delay (slave) | 3.3V domain | 3.0 | 12.0 | ns |
| $SPI_{10}$ | MOSI Setup time before SPCK falls (slave) | 3.3V domain | 1.2 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SPCK falls (slave) | 3.3V domain | 0.6 | – | ns |
| $SPI_{12}$ | NPCS setup to SPCK rising (slave) | 3.3V domain | 3.9 | – | ns |
| $SPI_{13}$ | NPCS hold after SPCK falling (slave) | 3.3V domain | 0 | – | ns |
| $SPI_{14}$ | NPCS setup to SPCK falling (slave) | 3.3V domain | 4.0 | – | ns |
| $SPI_{15}$ | NPCS hold after SPCK falling (slave) | 3.3V domain | 0 | – | ns |

Note that in SPI master mode, the device does not sample the data (MISO) on the opposite edge where the data clocks out (MOSI), but the same edge is used. See Figure 58-19 and Figure 58-20.

### 58.13.1.7 HSMCI Timings

The High-speed MultiMedia Card Interface (HSMCI) supports the MultiMedia Card (MMC) Specification V4.3, the SD Memory Card Specification V2.0, the SDIO V2.0 specification and CE-ATA V1.1.

### 58.13.1.8 SDRAM Timings

The SDRAM Controller satisfies the timings of standard SDR-133 and LP-SDR-133 modules. SDR-133 and LP-SDR-133 timings are specified by the JEDEC standard.

### 58.13.1.9 SMC Timings

Timings are given in the 3.3V domain, with VDDIO from 2.85V to 3.6V, maximum external capacitor = 50 pF.

Timings are given assuming a capacitance load on data, control and address pads.

In the tables that follow, $t_{CPMCK}$ is MCK period.

#### 58.13.1.9.1 Read Timings

**Table 58-55. SMC Read Signals - NRD Controlled (READ_MODE = 1)**

| Symbol | Parameter | VDDIO Supply | | Unit |
|---|---|---|---|---|
| | | Min | Max | |
| NO HOLD Settings (NRD_HOLD = 0) | | | | |
| $SMC_1$ | Data Setup before NRD High | 14.3 | – | ns |
| $SMC_2$ | Data Hold after NRD High | 0 | – | ns |
| HOLD Settings (NRD_HOLD ≠ 0) | | | | |
| $SMC_3$ | Data Setup before NRD High | 12.1 | – | ns |
| $SMC_4$ | Data Hold after NRD High | 0 | – | ns |
| HOLD or NO HOLD Settings (NRD_HOLD ≠ 0, NRD_HOLD = 0) | | | | |
| $SMC_5$ | A0–A22 Valid before NRD High | (NRD_SETUP + NRD_PULSE) × $t_{CPMCK}$ - 4.3 | – | ns |
| $SMC_6$ | NCS low before NRD High | (NRD_SETUP + NRD_PULSE - NCS_RD_SETUP) × $t_{CPMCK}$ - 2.4 | – | ns |
| $SMC_7$ | NRD Pulse Width | NRD_PULSE × $t_{CPMCK}$ - 0.3 | – | ns |

**Table 58-56. SMC Read Signals - NCS Controlled (READ_MODE = 0)**

| Symbol | Parameter | VDDIO Supply | | Unit |
|---|---|---|---|---|
| | | Min | Max | |
| NO HOLD Settings (NCS_RD_HOLD = 0) | | | | |
| $SMC_8$ | Data Setup before NCS High | 21.4 | – | ns |
| $SMC_9$ | Data Hold after NCS High | 0 | – | ns |
| HOLD Settings (NCS_RD_HOLD ≠ 0) | | | | |
| $SMC_{10}$ | Data Setup before NCS High | 11.7 | – | ns |
| $SMC_{11}$ | Data Hold after NCS High | 0 | – | ns |
| HOLD or NO HOLD Settings (NCS_RD_HOLD ≠ 0, NCS_RD_HOLD = 0) | | | | |
| $SMC_{12}$ | A0–A22 valid before NCS High | (NCS_RD_SETUP + NCS_RD_PULSE) × $t_{CPMCK}$ - 3.9 | – | ns |
| $SMC_{13}$ | NRD low before NCS High | (NCS_RD_SETUP + NCS_RD_PULSE - NRD_SETUP) × $t_{CPMCK}$ - 4.2 | – | ns |
| $SMC_{14}$ | NCS Pulse Width | NCS_RD_PULSE length × $t_{CPMCK}$ - 0.2 | – | ns |

#### 58.13.1.9.2 Write Timings

**Table 58-57. SMC Write Signals - NWE Controlled (WRITE_MODE = 1)**

| Symbol | Parameter | VDDIO Supply | | Unit |
|---|---|---|---|---|
| | | Min | Max | |
| HOLD or NO HOLD Settings (NWE_HOLD ≠ 0, NWE_HOLD = 0) | | | | |
| $SMC_{15}$ | Data Out Valid before NWE High | NWE_PULSE × $t_{CPMCK}$ - 4.6 | – | ns |
| $SMC_{16}$ | NWE Pulse Width | NWE_PULSE × $t_{CPMCK}$ - 0.3 | – | ns |

© 2021 Microchip Technology Inc.
and its subsidiaries

| .....continued | | | | |
|---|---|---|---|---|
| Symbol | Parameter | VDDIO Supply | | Unit |
| | | Min | Max | |
| $SMC_{17}$ | A0–A22 valid before NWE low | NWE_SETUP × $t_{CPMCK}$ - 4.2 | – | ns |
| $SMC_{18}$ | NCS low before NWE high | (NWE_SETUP - NCS_RD_SETUP + NWE_PULSE) × $t_{CPMCK}$ - 2.2 | – | ns |
| HOLD Settings (NWE_HOLD ≠ 0) | | | | |
| $SMC_{19}$ | NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 change | NWE_HOLD × $t_{CPMCK}$ - 3.9 | – | ns |
| $SMC_{20}$ | NWE High to NCS Inactive (1) | (NWE_HOLD - NCS_WR_HOLD) × $t_{CPMCK}$ - 3.6 | – | ns |
| NO HOLD Settings (NWE_HOLD = 0) | | | | |
| $SMC_{21}$ | NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25, NCS change(1) | 1.5 | – | ns |

**Note:**
Hold length = total cycle duration - setup duration - pulse duration. “hold length” is for “NCS_WR_HOLD length” or “NWE_HOLD length”

**Table 58-58. SMC Write NCS Controlled (WRITE_MODE = 0)**

| Symbol | Parameter | VDDIO Supply | | Unit |
|---|---|---|---|---|
| | | Min | Max | |
| $SMC_{22}$ | Data Out Valid before NCS High | NCS_WR_PULSE × $t_{CPMCK}$ - 3.9 | – | ns |
| $SMC_{23}$ | NCS Pulse Width | NCS_WR_PULSE × $t_{CPMCK}$ - 0.2 | – | ns |
| $SMC_{24}$ | A0–A22 valid before NCS low | NCS_WR_SETUP × $t_{CPMCK}$ - 4.6 | – | ns |
| $SMC_{25}$ | NWE low before NCS high | (NCS_WR_SETUP - NWE_SETUP + NCS pulse) × $t_{CPMCK}$ - 4.6 | – | ns |
| $SMC_{26}$ | NCS High to Data Out, A0–A25, change | NCS_WR_HOLD × $t_{CPMCK}$ - 3.4 | – | ns |
| $SMC_{27}$ | NCS High to NWE Inactive | (NCS_WR_HOLD - NWE_HOLD) × $t_{CPMCK}$ - 2.4 | – | ns |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 58-24. SMC Timings - NCS Controlled Read and Write**

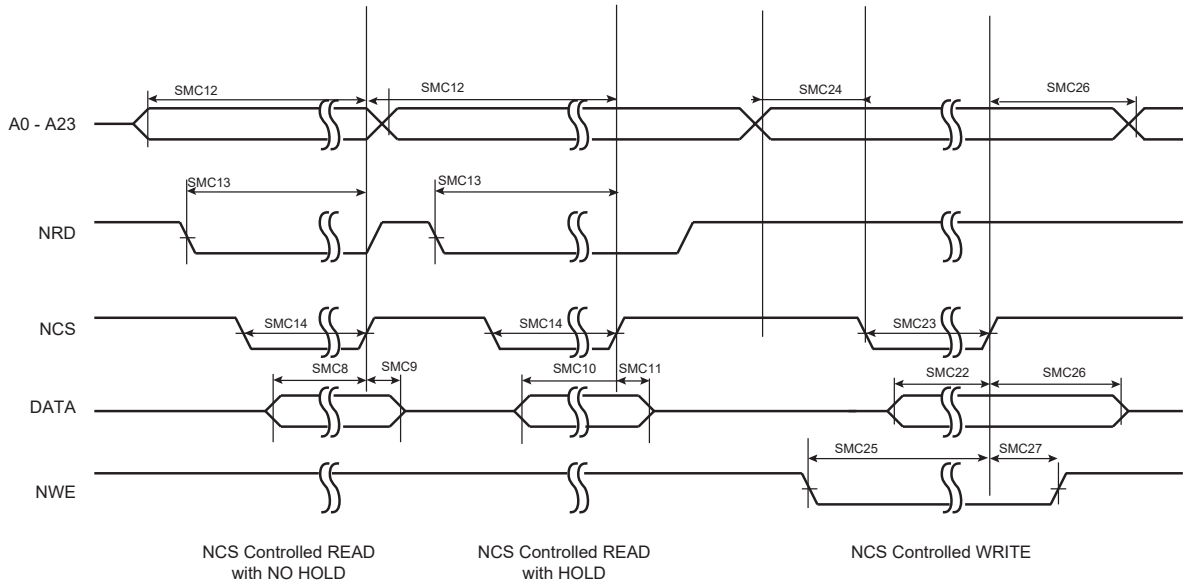
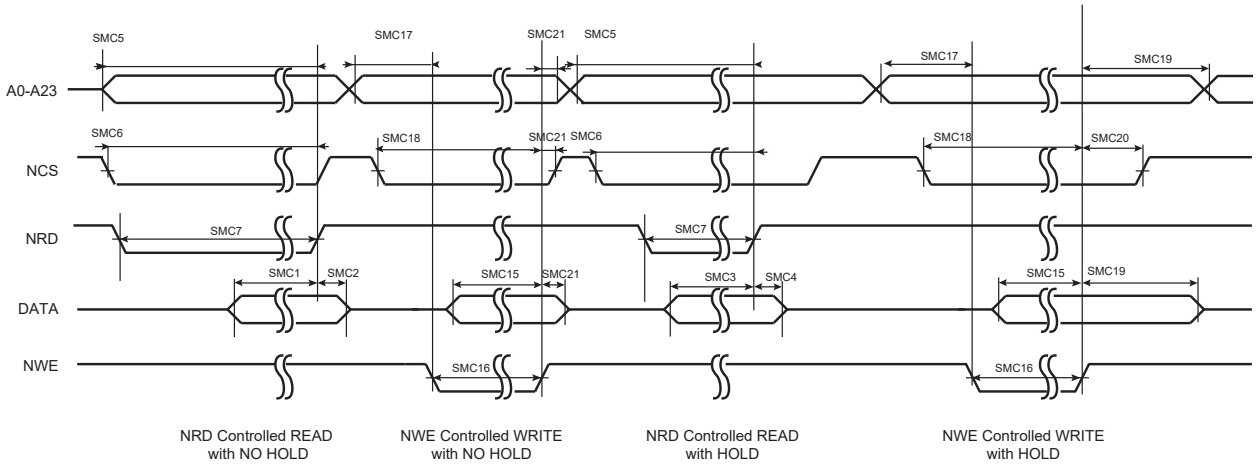



**Figure 58-25. SMC Timings - NRD Controlled Read and NWE Controlled Write**




### 58.13.1.10 USART in Asynchronous Modes

In Asynchronous modes, the maximum baud rate that can be achieved is MCK2/8, if the bit USART_MR.OVER = 1.

Example: if peripheral clock = 150 MHz, the maximum achievable baud rate is 18.75 MBit/s.

### 58.13.1.11 USART in SPI Mode Timings

**Figure 58-26. USART SPI Master Mode**

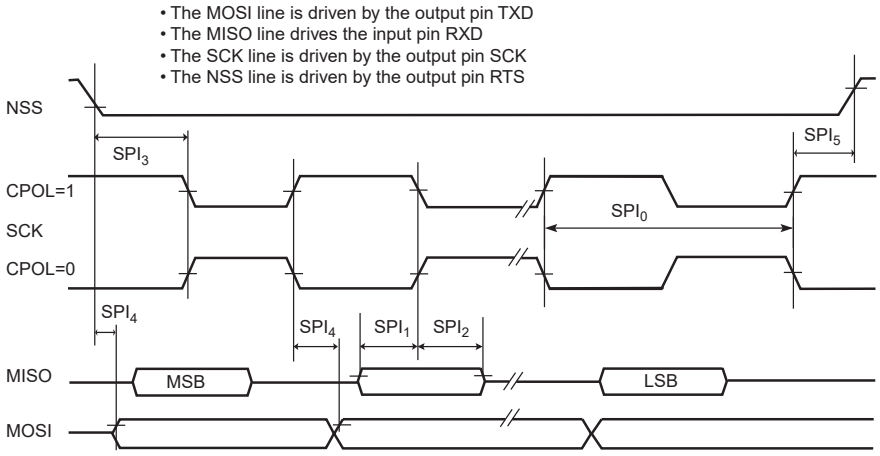



**Figure 58-27. USART SPI Slave Mode (Mode 1 or 2)**

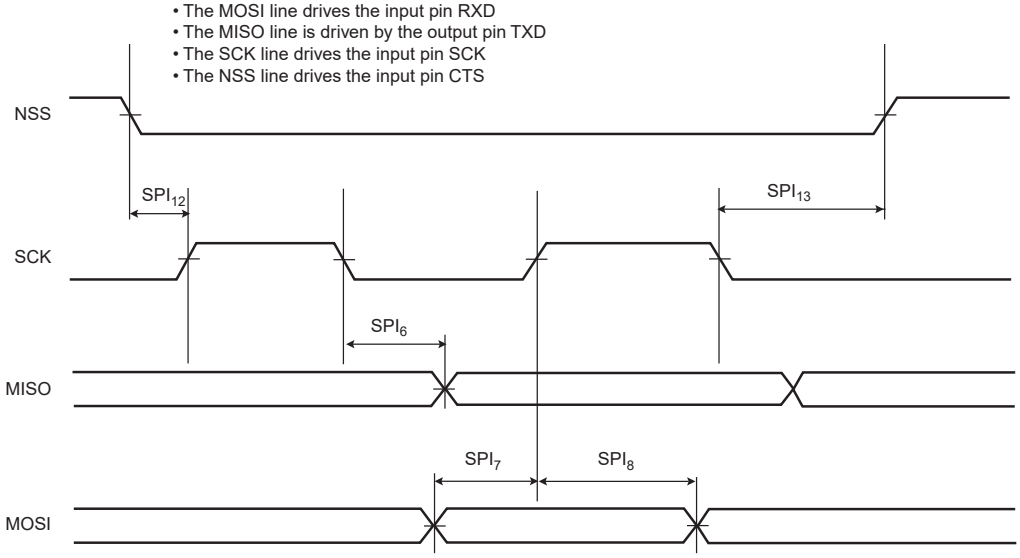
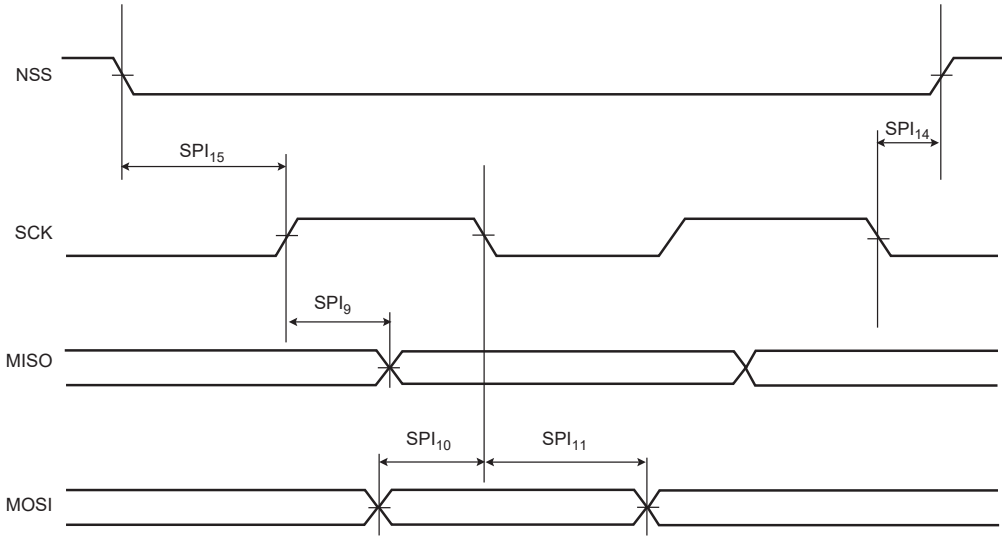



**Figure 58-28. USART SPI Slave Mode (Mode 0 or 3)**



#### 58.13.1.11.1 USART SPI Timings

Timings are given for the 3.3V domain, with VDDIO from 2.85V to 3.6V, maximum external capacitor = 40 pF.

**Table 58-59. USART SPI Timings**

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| Master Mode | | | | |
| $SPI_0$ | SCK Period | MCK/6 | – | ns |
| $SPI_1$ | Input Data Setup Time | 2.5 | – | ns |
| $SPI_2$ | Input Data Hold Time | 0.2 | – | ns |
| $SPI_3$ | Chip Select Active to Serial Clock | -0.9 | – | ns |
| $SPI_4$ | Output Data Setup Time | -1.9 | 10.4 | ns |
| $SPI_5$ | Serial Clock to Chip Select Inactive | -2.4 | -1.9 | ns |
| Slave Mode | | | | |
| $SPI_6$ | SCK falling to MISO | 2.9 | 13.9 | ns |
| $SPI_7$ | MOSI Setup time before SCK rises | 2.0 | – | ns |
| $SPI_8$ | MOSI Hold time after SCK rises | 0.2 | – | ns |
| $SPI_9$ | SCK rising to MISO | 3.0 | 13.5 | ns |
| $SPI_{10}$ | MOSI Setup time before SCK falls | 2.1 | – | ns |
| $SPI_{11}$ | MOSI Hold time after SCK falls | 0.4 | – | ns |
| $SPI_{12}$ | NPCS0 setup to SCK rising | 0.6 | – | ns |
| $SPI_{13}$ | NPCS0 hold after SCK falling | 0.6 | – | ns |
| $SPI_{14}$ | NPCS0 setup to SCK falling | 0.6 | – | ns |
| $SPI_{15}$ | NPCS0 hold after SCK rising | 0.7 | – | ns |

#### 58.13.1.12 Two-wire Serial Interface Characteristics

The follow table describes the requirements for devices connected to the Two-wire Serial Bus.

For timing symbols, refer to the following figure.

**Table 58-60. Two-wire Serial Bus Requirements**

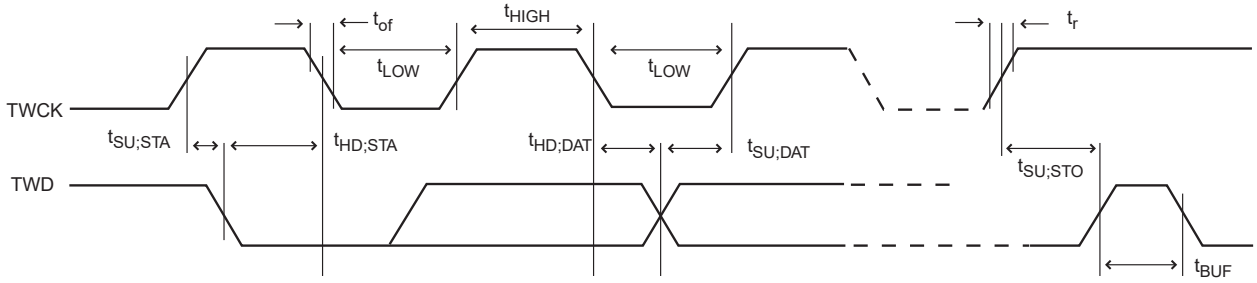
| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{IL}$ | Low-level Input Voltage | – | -0.3 | 0.3 $V_{DDIO}$ | V |
| $V_{IH}$ | High-level Input Voltage | – | 0.7 × $V_{DDIO}$ | $V_{CC}$ + 0.3 | V |
| $V_{hys}$ | Hysteresis of Schmitt Trigger Inputs | – | 0.150 | – | V |
| $V_{OL}$ | Low-level Output Voltage | 3 mA sink current | – | 0.4 | V |
| $t_R$ | Rise Time for both TWD and TWCK | | 20 + 0.1$C_b$(1)(2) | 300 | ns |
| $t_{OF}$ | Output Fall Time from $V_{IHmin}$ to $V_{ILmax}$ | 10 pF < $C_b$ < 400 pF see the figure below | 20 + 0.1$C_b$(1)(2) | 250 | ns |
| $C_i$(1) | Capacitance for each I/O Pin | – | – | 10 | pF |
| $f_{TWCK}$ | TWCK Clock Frequency | – | 0 | 400 | kHz |

| .....continued | | | | | |
|---|---|---|---|---|---|
| **Symbol** | **Parameter** | **Condition** | **Min** | **Max** | **Unit** |
| $R_P$ | Value of Pull-up resistor | $f_{TWCK} \leq 100$ kHz | $(V_{DDIO} - 0.4V) \div 3mA$ | $1000ns \div C_b$ | Ω |
| | | $f_{TWCK} > 100$ kHz | | $300ns \div C_b$ | Ω |
| $t_{LOW}$ | Low Period of the TWCK clock | $f_{TWCK} \leq 100$ kHz | (3) | – | µs |
| | | $f_{TWCK} > 100$ kHz | (3) | – | µs |
| $t_{HIGH}$ | High period of the TWCK clock | $f_{TWCK} \leq 100$ kHz | (4) | – | µs |
| | | $f_{TWCK} > 100$ kHz | (4) | – | µs |
| $t_{HD;STA}$ | Hold Time (repeated) START Condition | $f_{TWCK} \leq 100$ kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK} > 100$ kHz | $t_{HIGH}$ | – | µs |
| $t_{SU;STA}$ | Set-up time for a repeated START condition | $f_{TWCK} \leq 100$ kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK} > 100$ kHz | $t_{HIGH}$ | – | µs |
| $t_{HD;DAT}$ | Data hold time | $f_{TWCK} \leq 100$ kHz | 0 | $3 \times t_{CPMCK}$(5) | µs |
| | | $f_{TWCK} > 100$ kHz | 0 | $3 \times t_{CPMCK}$(5) | µs |
| $t_{SU;DAT}$ | Data setup time | $f_{TWCK} \leq 100$ kHz | $t_{LOW} - 3 \times t_{CPMCK}$(5) | – | ns |
| | | $f_{TWCK} > 100$ kHz | $t_{LOW} - 3 \times t_{CPMCK}$(5) | – | ns |
| $t_{SU;STO}$ | Setup time for STOP condition | $f_{TWCK} \leq 100$ kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK} > 100$ kHz | $t_{HIGH}$ | – | µs |
| $t_{HD;STA}$ | Hold Time (repeated) START Condition | $f_{TWCK} \leq 100$ kHz | $t_{HIGH}$ | – | µs |
| | | $f_{TWCK} > 100$ kHz | $t_{HIGH}$ | – | µs |

**Notes:**

1. Required only for $f_{TWCK} > 100$ kHz.
2. $C_b$ = capacitance of one bus line in pF. Per $I^2C$ standard, $C_b$ max = 400pF.
3. The TWCK low period is defined as follows: $t_{LOW} = ((CLDIV \times 2^{CKDIV}) + 4) \times t_{MCK}$.
4. The TWCK high period is defined as follows: $t_{HIGH} = ((CHDIV \times 2^{CKDIV}) + 4) \times t_{MCK}$.
5. $t_{CPMCK}$ = MCK bus period

**Figure 58-29. Two-wire Serial Bus Timing**



### 58.13.1.13 GMAC Characteristics

#### 58.13.1.13.1 Timing Conditions

**Table 58-61. Load Capacitance on Data, Clock Pads**

| Supply | $C_L$ | |
|---|---|---|
| | Max | Min |
| 3.3V | 20 pF | 0 pF |

#### 58.13.1.13.2 Timing Constraints

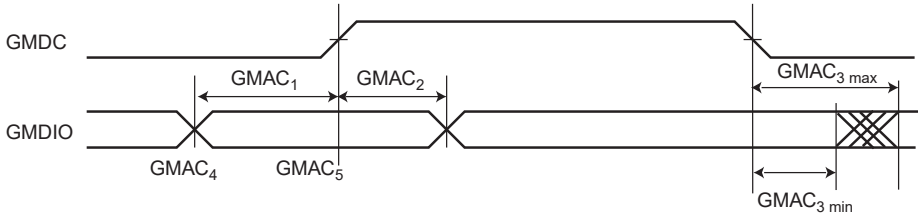
The GMAC must be constrained so as to satisfy the timings of standards shown below and in 58.13.1.13.3 MII Mode, in MAX corner.

**Table 58-62. GMAC Signals Relative to GMDC**

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $GMAC_1$ | Setup for GMDIO from GMDC rising | 10 | – | ns |
| $GMAC_2$ | Hold for GMDIO from GMDC rising | 10 | – | |
| $GMAC_3$ | GMDIO toggling from GMDC falling | 0(1) | 10(1) | |

Note: 1. For GMAC output signals, min and max access time are defined. The min access time is the time between the GMDC falling edge and the signal change. The max access timing is the time between the GMDC falling edge and the signal stabilizes. The figure below illustrates min and max accesses for $GMAC_3$.

**Figure 58-30. Min and Max Access Time of GMAC Output Signals**



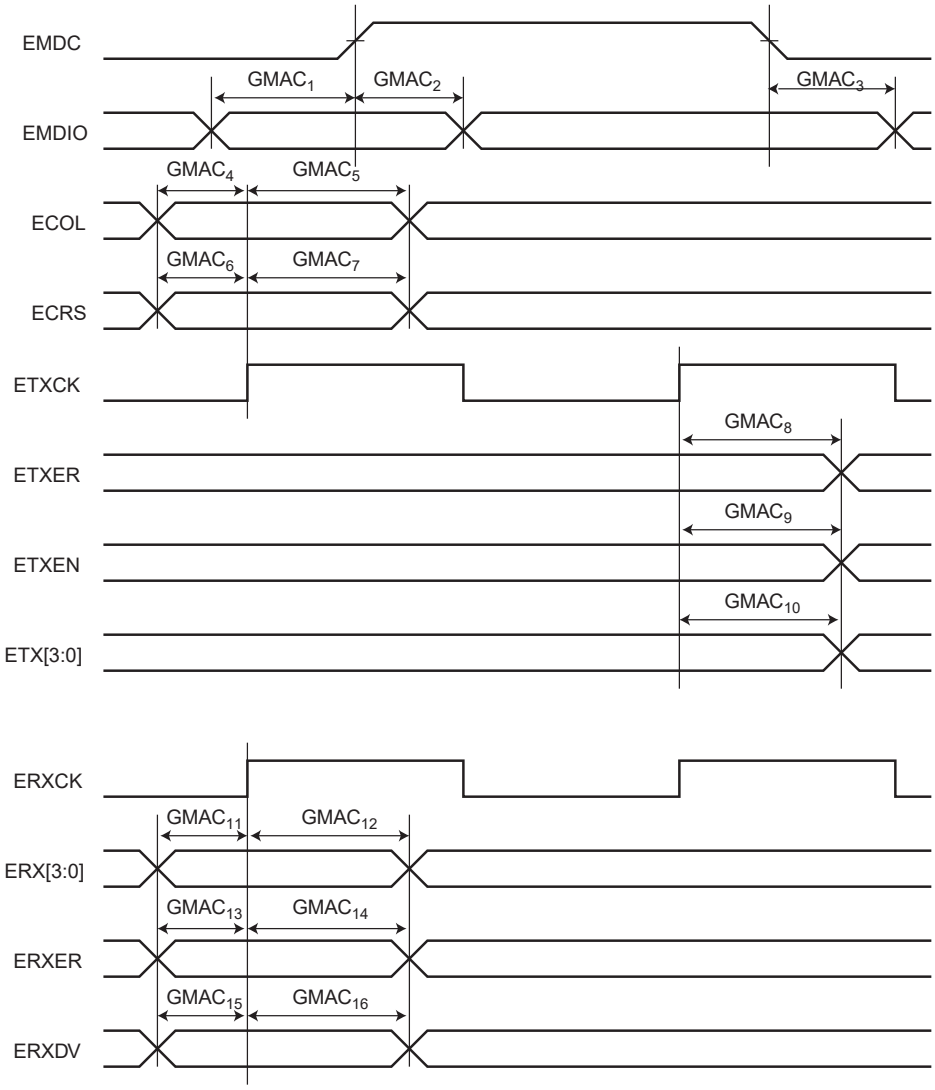
#### 58.13.1.13.3 MII Mode

**Table 58-63. GMAC MII Mode Timings**

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $GMAC_4$ | Setup for GCOL from GTXCK rising | 10 | – | ns |
| $GMAC_5$ | Hold for GCOL from GTXCK rising | 10 | – | |
| $GMAC_6$ | Setup for GCRS from GTXCK rising | 10 | – | |
| $GMAC_7$ | Hold for GCRS from GTXCK rising | 10 | – | |
| $GMAC_8$ | GTXER toggling from GTXCK rising | 10 | 25 | |
| $GMAC_9$ | GTXEN toggling from GTXCK rising | 10 | 25 | |
| $GMAC_{10}$ | GTX toggling from GTXCK rising | 10 | 25 | |
| $GMAC_{11}$ | Setup for GRX from GRXCK | 10 | – | |
| $GMAC_{12}$ | Hold for GRX from GRXCK | 10 | – | |
| $GMAC_{13}$ | Setup for GRXER from GRXCK | 10 | – | |
| $GMAC_{14}$ | Hold for GRXER from GRXCK | 10 | – | |
| $GMAC_{15}$ | Setup for GRXDV from GRXCK | 10 | – | |
| $GMAC_{16}$ | Hold for GRXDV from GRXCK | 10 | – | |

© 2021 Microchip Technology Inc.
and its subsidiaries

**Figure 58-31. GMAC MII Mode Signals**

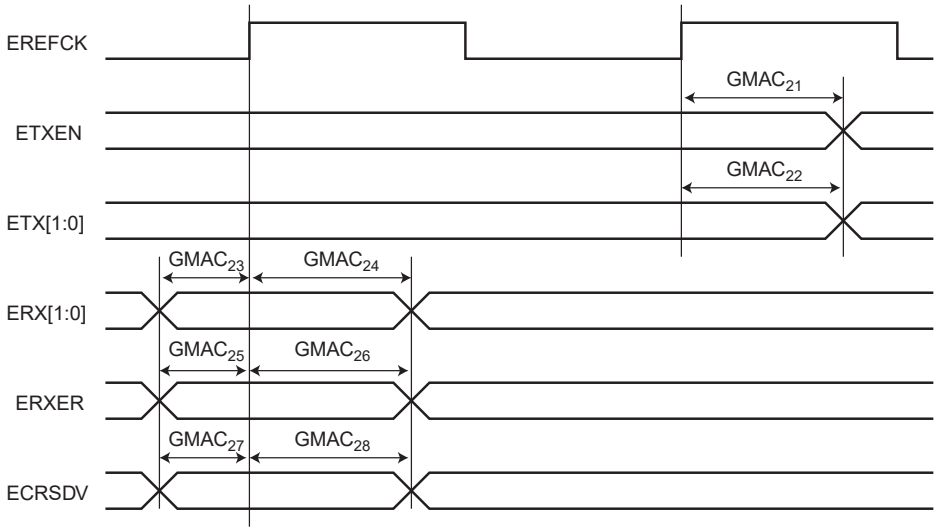



### 58.13.1.13.4 RMII Mode

**Table 58-64. GMAC RMII Mode Timings**

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $GMAC_{21}$ | ETXEN toggling from EREFCK rising | 2 | 16 | ns |
| $GMAC_{22}$ | ETX toggling from EREFCK rising | 2 | 16 | |
| $GMAC_{23}$ | Setup for ERX from EREFCK rising | 4 | – | |
| $GMAC_{24}$ | Hold for ERX from EREFCK rising | 2 | – | |
| $GMAC_{25}$ | Setup for ERXER from EREFCK rising | 4 | – | |
| $GMAC_{26}$ | Hold for ERXER from EREFCK rising | 2 | – | |
| $GMAC_{27}$ | Setup for ECRSDV from EREFCK rising | 4 | – | |
| $GMAC_{28}$ | Hold for ECRSDV from EREFCK rising | 2 | – | |

**Figure 58-32. GMAC RMII Mode Signals**




### 58.13.1.14 SSC Timings

#### 58.13.1.14.1 Timing Conditions

Timings are given assuming the load capacitance in the following table.

**Table 58-65. Load Capacitance**

| Supply | $C_L$ Max |
|---|---|
| 3.3V | 30 pF |

#### 58.13.1.14.2 Timing Extraction

**Figure 58-33. SSC Transmitter, TK and TF in Output**

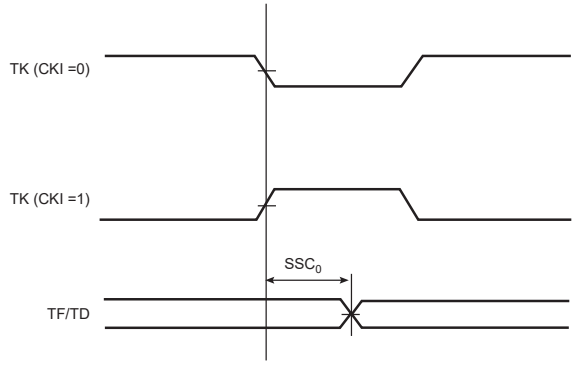



**Figure 58-34. SSC Transmitter, TK in Input and TF in Output**

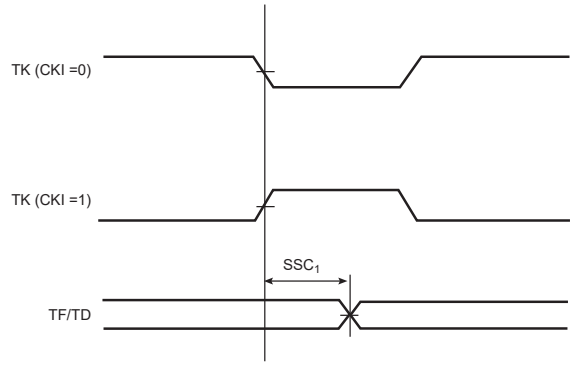


**Figure 58-35. SSC Transmitter, TK in Output and TF in Input**

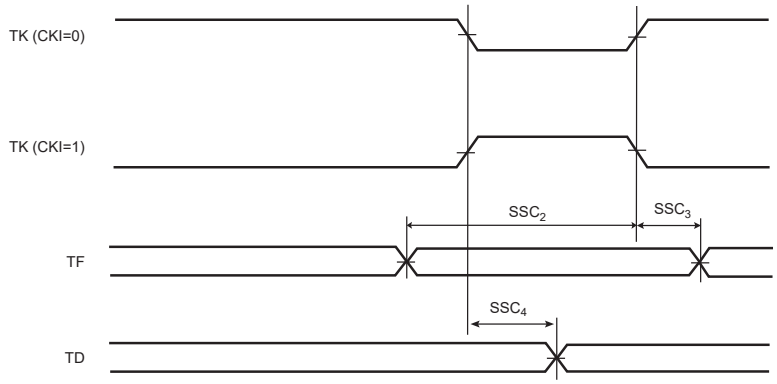



**Figure 58-36. SSC Transmitter, TK and TF in Input**

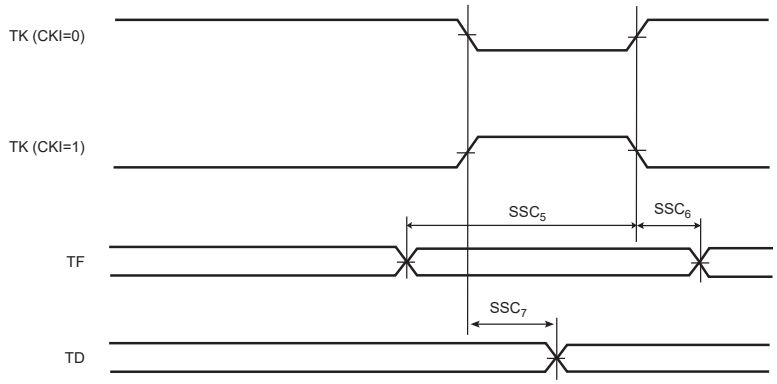



**Figure 58-37. SSC Receiver RK and RF in Input**

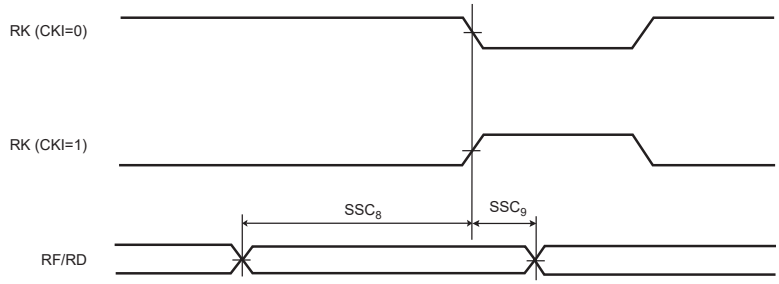



**Figure 58-38. SSC Receiver, RK in Input and RF in Output**

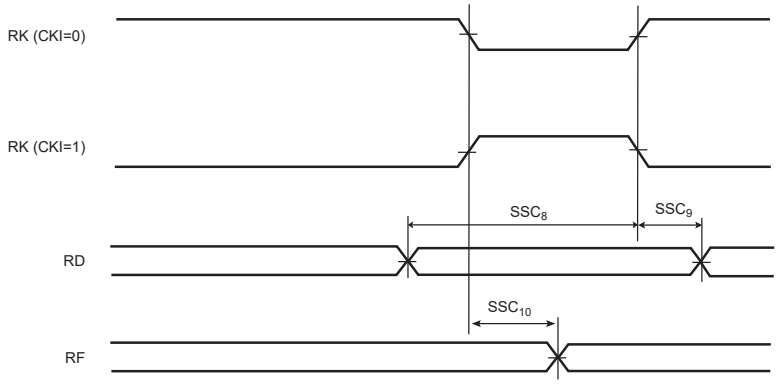


**Figure 58-39. SSC Receiver, RK and RF in Output**

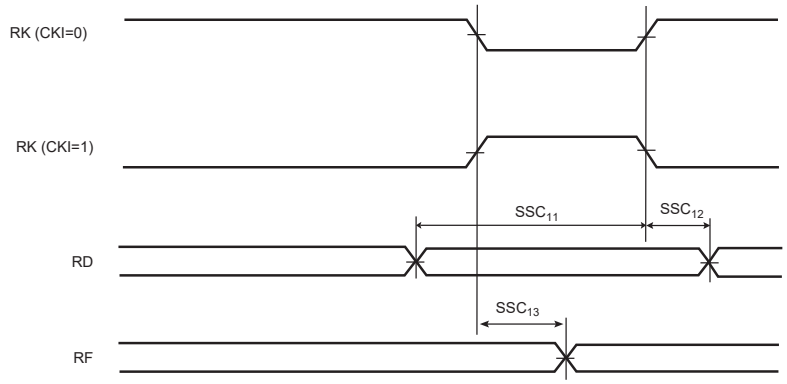



**Figure 58-40. SSC Receiver, RK in Output and RF in Input**

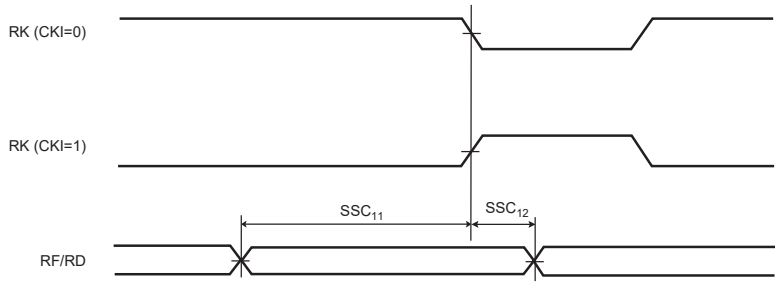


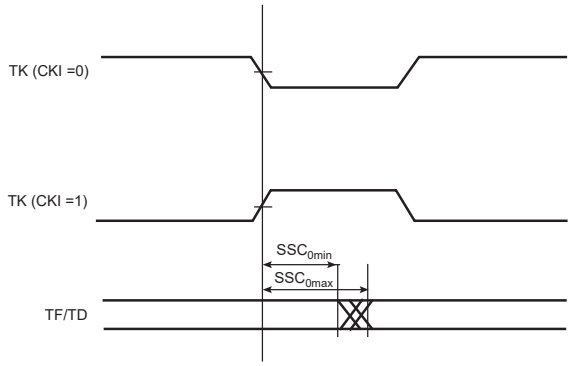

**Table 58-66. SSC Timings with 3.3V Peripheral Supply**

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| Transmitter | | | | | |
| $SSC_0$ | TK edge to TF/TD (TK output, TF output) | – | -3.9(1) | 4.0 (1) | ns |
| $SSC_1$ | TK edge to TF/TD (TK input, TF output) | – | 3.1(1) | 12.7(1) | ns |
| $SSC_2$ | TF setup time before TK edge (TK output) | – | 13.6 | – | ns |
| $SSC_3$ | TF hold time after TK edge (TK output) | – | 0 | – | ns |
| $SSC_4$ | TK edge to TF/TD (TK output, TF input) | – | -3.9(1) | 3.0(1) | ns |
| | | STTDLY = 0 START = 4, 5 or 7 | -3.9 + (2× $t_{CPMCK}$)(1) | 3.0 + (2 × $t_{CPMCK}$)(1) | |
| $SSC_5$ | TF setup time before TK edge (TK input) | – | 0 | – | ns |
| $SSC_6$ | TF hold time after TK edge (TK input) | – | $t_{CPMCK}$ | – | ns |
| $SSC_7$ | TK edge to TF/TD (TK input, TF input) | – | 3.1(1) | 11.8(1) | ns |
| | | STTDLY = 0 START = 4, 5 or 7 | 3.1 + (3 × $t_{CPMCK}$)(1) | 11.8 + (3 × $t_{CPMCK}$)(1) | |
| Receiver | | | | | |
| $SSC_8$ | RF/RD setup time before RK edge (RK input) | – | 0 | – | ns |
| $SSC_9$ | RF/RD hold time after RK edge (RK input) | – | $t_{CPMCK}$ | – | ns |
| $SSC_{10}$ | RK edge to RF (RK input) | – | 2.9(1) | 9.2(1) | ns |
| $SSC_{11}$ | RF/RD setup time before RK edge (RK output) | – | 10.1 - $t_{CPMCK}$ | – | ns |

| .....continued | | | | | |
|---|---|---|---|---|---|
| Symbol | Parameter | Condition | Min | Max | Unit |
| $SSC_{12}$ | RF/RD hold time after RK edge (RK output) | – | $t_{CPMCK}$ - 2.8 | – | ns |
| $SSC_{13}$ | RK edge to RF (RK output) | – | -2.1(1) | 1.9(1) | ns |

**Note:** For output signals (TF, TD, RF), min and max access times are defined. The min access time is the time between the TK (or RK) edge and the signal change. The max access timing is the time between the TK edge and the signal stabilization. The information below illustrates min and max accesses for SSC0. The same applies for SSC1, SSC4, and SSC7, SSC10 and SSC13.

**Figure 58-41. Min and Max Access Time of Output Signals**




### 58.13.1.15 ISI Timings

#### 58.13.1.15.1 Timing Conditions

Timings are given assuming the load capacitance in the following table.

**Table 58-67. Load Capacitance**

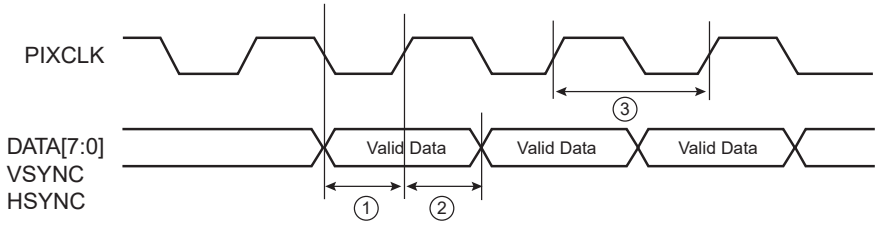
| Supply | $C_L$ Max |
|---|---|
| 3.3V | 30 pF |

#### 58.13.1.15.2 Timing Extraction

**Table 58-68. ISI Timings with Peripheral Supply 3.3V**

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $ISI_1$ | DATA/VSYNC/HSYNC setup time | 1.5 | – | ns |
| $ISI_2$ | DATA/VSYNC/HSYNC hold time | -1.2 | – | ns |
| $ISI_3$ | PIXCLK frequency | – | 75 | MHz |

**Figure 58-42. ISI Timing Diagram**



# 59. Schematic Checklist

The schematic checklist provides the user with the requirements regarding the different pin connections that must be considered before starting any new board design. It also provides information on the minimum hardware resources required to quickly develop an application with the SAMV71Q21RT device. It does not consider PCB layout constraints.

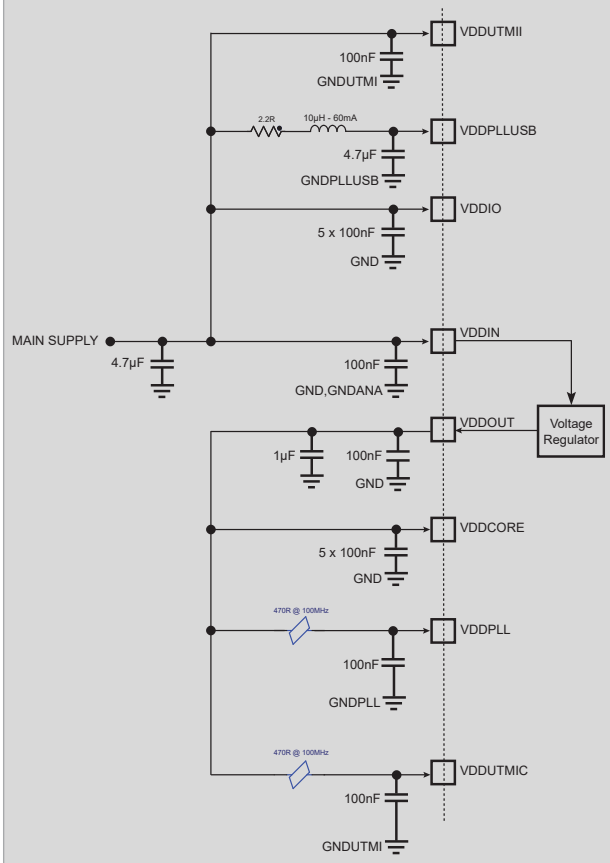
This information is not intended to be exhaustive. Its objective is to cover as many configurations of use as possible. The checklist contains a column for use by designers, making it easy to track and verify each line item.

## 59.1 Power Supplies

### 59.1.1 Supplying the Device With Only One Supply

**⚠ CAUTION** To guarantee reliable operation of the device, the board design must comply with power-up and power-down sequence guidelines provided in the "Power Considerations" chapter.

Power Supplies Schematic Example with Internal Regulator Use




Note: Component values are given only as a typical example.

Note: Restrictions
With main supply < 2.5V, USB and DACC are not usable.
With main supply > 2.5V and < 3V, USB is not usable.
With main supply > 3.0 V, all peripherals are usable.

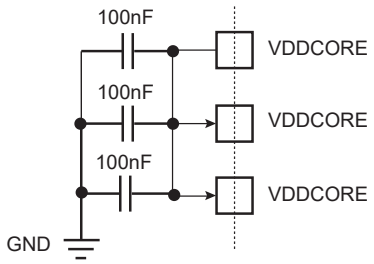
| Check | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | VDDIN | Decoupling/filtering capacitors (100 nF and 4.7 µF)(1)(2) | Powers the voltage regulator, AFE, DAC, and Analog comparator power supply<br>Supply ripple must not exceed 20 mVrms for 10 kHz to 20 MHz range.<br>**WARNING** VDDIN and VDDIO must have the same level and must be higher than VDDCORE.<br>**WARNING** Powerup and powerdown sequences given in the "Power Considerations" chapter must be respected. |
| | VDDIO | Decoupling/filtering capacitors (100 nF)(1)(2) | Powers the Peripheral I/O lines (Input/Output Buffers), backup part, 1 Kbyte of Backup SRAM, 32 kHz crystal oscillator, oscillator pads<br>Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.<br>Supply ripple must not exceed 30 mVrms for 10 kHz to 10 MHz range.<br>**WARNING** VDDIN and VDDIO must have the same level and must be higher than VDDCORE.<br>**WARNING** Powerup and powerdown sequences given in the "Power Considerations" chapter must be respected. |
| | VDDUTMII | Decoupling capacitor (100 nF)(1)(2) | Powers the USB transceiver interface. Must be connected to VDDIO.<br>For USB operations, VDDUTMII and VDDIO voltage ranges must be from 3.0V to 3.6V.<br>Must always be connected even when the USB is not used.<br>Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.<br>Supply ripple must not exceed 20 mVrms for 10 kHz to 10 MHz range. |
| | VDDPLLUSB | Decoupling/filtering RLC circuit(1) | Powers the UTMI PLL and the 3 to 20 MHz oscillator.<br>The VDDPLLUSB power supply pin draws small current, but it is noise sensitive. Care must be taken in VDDPLLUSB power supply routing, decoupling and also on bypass capacitors.<br>Supply ripple must not exceed 10 mVrms for 10 kHz to 10 MHz range. |
| | VDDOUT | Decoupling capacitor (100 nF + 1 µF)(1)(2) | Voltage Regulator Output |

.....continued

| Check | Signal Name | Recommended Pin Connection | Description |
|---|---|---|---|
| | VDDCORE | Decoupling capacitor (100 nF)(1)(2) | Powers the core, embedded memories and peripherals. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.<br><br>⚠WARNING Powerup and powerdown sequences given in the "Power Considerations" chapter must be respected. |
| | VDDPLL | Decoupling/filtering capacitors ferrite beads (100 nF and 470 Ohm @ 100MHz)(1)(2) | Powers the PLLA and the fast RC oscillator.<br>The VDDPLL power supply pin draws small current, but it is noise sensitive. Care must be taken in VDDPLL power supply routing, decoupling and also on bypass capacitors. |
| | VDDUTMIC | Decoupling/filtering capacitors ferrite beads (100 nF and 470 Ohm @ 100MHz)(1)(2) | Powers the USB transceiver core.<br>Must always be connected even if the USB is not used.<br>Decoupling/filtering capacitors/ferrite beads must be added to improve startup stability and reduce source voltage drop. |
| | GND | Voltage Regulator, Core Chip and Peripheral I/O lines ground | GND pins are common to VDDIN, VDDCORE and VDDIO pins. GND pins should be connected as shortly as possible to the system ground plane. |
| | GNDUTMI | UDPHS and UHPHS UTMI+ Core and interface ground | GNDUTMI pins are common to VDDUTMII and VDDUTMIC pins. GNDUTMI pins should be connected as shortly as possible to the system ground plane. |
| | GNDPLL | PLLA cell and Main Oscillator ground | GNDPLL pin is provided for VDDPLL pin. GNDPLL pin should be connected as shortly as possible to the system ground plane. |
| | GNDANA | Analog ground | GNDANA pins are common to AFE, DAC and ACC supplied by VDDIN pin.<br>GNDANA pins should be connected as shortly as possible to the system ground plane. |
| | GNDPLLUSB | USB PLL ground | The GNDPLLUSB pin is provided for VDDPLLUSB pin. The GNDPLLUSB pin should be connected as shortly as possible to the system ground plane. |

**Notes:**

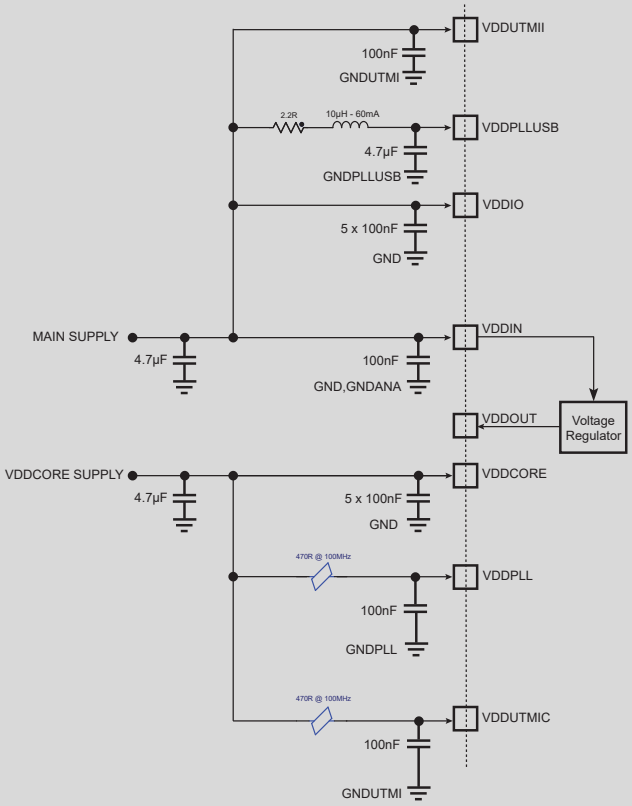
1. These values are given only as a typical example.
2. Decoupling capacitors must be connected as close as possible to the microcontroller and on each concerned pin, vias should be avoided.



### 59.1.2 Supplying the Device With Two Separate Supplies

**⚠ CAUTION** The board design must comply with power-up and power-down sequence guidelines provided in the "Power Considerations" chapter.

Power Supplies Schematic Example With Separate Power Supplies 59.3 Boot Program Hardware Constraints




Note: Component values are given only as a typical example.

Note: Restrictions
With main supply < 3.0 V, USB is not usable.
With main supply < 2.0 V, AFE, DAC and Analog comparator are not usable.
With main supply and VDDIN > 3.0 V, all peripherals are usable.

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| VDDIN | Decoupling/filtering capacitors (100 nF and 4.7 µF) (1, 2) | Powers the voltage regulator, AFE, DAC, and Analog comparator power supply<br>Supply ripple must not exceed 20 mVrms for 10 kHz to 20 MHz range.<br>**WARNING** VDDIN and VDDIO must have the same level and must always be higher than VDDCORE.<br>**WARNING** Power up and power down sequences given in the "Power Considerations" chapter must be respected. |
| VDDIO | Decoupling/filtering capacitors (100 nF) (1, 2) | Powers the Peripheral I/O lines (Input/Output Buffers), backup part, 1 Kbytes of Backup SRAM, 32 kHz crystal oscillator, oscillator pads Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.<br>Supply ripple must not exceed 30 mVrms for 10 kHz to 10 MHz range.<br>**WARNING** VDDIN and VDDIO must have the same level and must always be higher than VDDCORE.<br>**WARNING** Powerup and powerdown sequences given in the "Power Considerations" chapter must be respected. |
| VDDUTMII | Decoupling capacitor (100 nF) (1) (2) | Powers the USB transceiver interface. Must be connected to VDDIO. For USB operations, VDDUTMII and VDDIO voltage ranges must be from 3.0V to 3.6V.<br>Must always be connected even if the USB is not used.<br>Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.<br>Supply ripple must not exceed 20 mVrms for 10 kHz to 10 MHz range. |
| VDDPLLUSB | Decoupling/filtering RLC circuit (1) | Powers the UTMI PLL and the 3 to 20 MHz oscillator. For USB operations, VDDPLLUSB should be between 3.0V and 3.6V.<br>The VDDPLLUSB power supply pin draws small current, but it is noise sensitive. Care must be taken in VDDPLLUSB power supply routing, decoupling and also on bypass capacitors.<br>Supply ripple must not exceed 10 mVrms for 10 kHz to 10 MHz range. |
| VDDOUT | Left unconnected | Voltage Regulator Output |

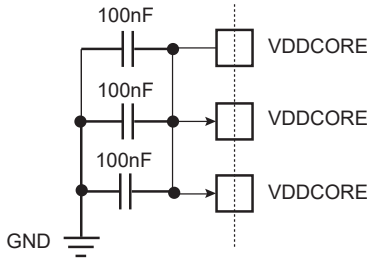
.....continued

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| VDDCORE | Decoupling capacitor (100 nF) (1) (2) | Powers the core, embedded memories and peripherals.<br>Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.<br>Supply ripple must not exceed 20 mVrms for 10 kHz to 20 MHz range.<br>**WARNING** Powerup and powerdown sequences given in the "Power Considerations" chapter must be respected. |
| VDDPLL | Decoupling/filtering capacitors ferrite beads (100 nF and 470 Ohm @ 100 MHz) (1) (2) | Powers the PLLA and the fast RC oscillator.<br>The VDDPLL power supply pin draws small current, but it is noise sensitive. Care must be taken in VDDPLL power supply routing, decoupling and also on bypass capacitors.<br>Supply ripple must not exceed 20 mVrms for 10 kHz to 10 MHz range and 10 mVrms for higher frequencies. |
| VDDUTMIC | Decoupling/filtering capacitors ferrite beads (100 nF and 470 Ohm @ 100 MHz) (1) (2) | Powers the USB transceiver core.<br>Must always be connected even if the USB is not used.<br>Decoupling/filtering capacitors/ferrite beads must be added to improve startup stability and reduce source voltage drop.<br>Supply ripple must not exceed 10 mVrms for 10 kHz to 10 MHz range. |
| GND | Voltage Regulator, Core Chip and Peripheral I/O lines ground | GND pins are common to VDDIN, VDDCORE and VDDIO pins.<br>GND pins should be connected as shortly as possible to the system ground plane. |
| GNDUTMI | UDPHS and UHPHS UTMI+ Core and interface ground | GNDUTMI pins are common to VDDUTMII and VDDUTMIC pins.<br>GNDUTMI pins should be connected as shortly as possible to the system ground plane. |
| GNDPLL | PLLA cell and Main Oscillator ground | GNDPLL pin is provided for VDDPLL pin. GNDPLL pin should be connected as shortly as possible to the system ground plane. |
| GNDANA | Analog ground | GNDANA pins are common to AFE, DAC and ACC supplied by VDDIN pin.<br>GNDANA pins should be connected as shortly as possible to the system ground plane. |
| GNDPLLUSB | USB PLL ground | GNDPLLUSB pin is provided for VDDPLLUSB pin. GNDPLLUSB pin should be connected as shortly as possible to the system ground plane. |

**Notes:**

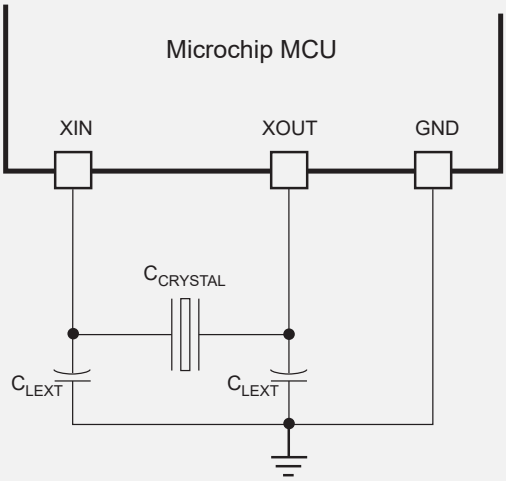
1. These values are given only as a typical example.
2. Decoupling capacitors must be connected as close as possible to the microcontroller and on each concerned pin, vias should be avoided.

© 2021 Microchip Technology Inc. and its subsidiaries

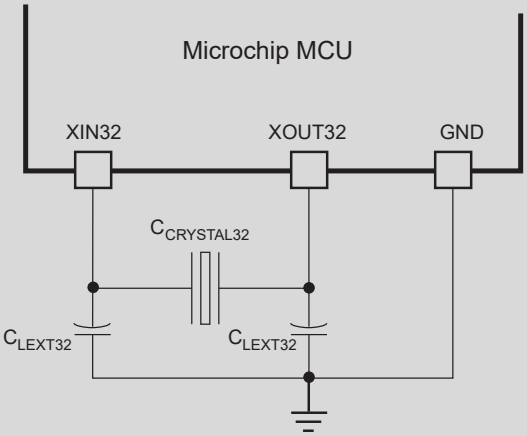



## 59.2 General Hardware Recommendations

### 59.2.1 Crystal Oscillators

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| XIN<br>XOUT<br>3 to 20 MHz Crystal Oscillator in Normal Mode | Crystals between 3 and 20 MHz<br>USB High/Full Speed Host/Device peripherals require a 12 or 16 MHz clock.<br>Capacitors on XIN and XOUT<br>(Crystal Load Capacitance dependent) | Crystal Load Capacitance to check ($C_{CRYSTAL}$).<br><br>Example: for a 12 MHz crystal with a load capacitance of $C_{CRYSTAL}$ = 15 pF, external capacitors are required:<br>$C_{LEXT}$ = 12 pF.<br>Refer to the Electrical Characteristics section. |
| XIN<br>XOUT<br>3 to 20 MHz Crystal Oscillator in Bypass Mode | XIN: external clock source<br>XOUT: can be left unconnected<br>USB High/Full speed Host/Device peripherals require a 12 or 16 MHz clock. | VDDIO square wave signal<br>External clock source up to 20 MHz<br>Duty Cycle: 40 to 60%<br>Refer to the Electrical Characteristics section. |
| XIN<br>XOUT<br>3 to 20 MHz Crystal Oscillator Disabled | XIN: can be left unconnected<br>XOUT: can be left unconnected<br>USB High/Full-speed Host/Device peripherals are not functional with Main RC oscillator. | Typical nominal frequency 12 MHz<br>(Main RC Oscillator)<br>Duty Cycle: 45 to 55%<br>Refer to the Electrical Characteristics section. |

.....continued

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| XIN32<br>XOUT32<br>Slow Clock Oscillator | 32.768 kHz Crystal<br>Capacitors on XIN32 and XOUT32<br>(Crystal Load Capacitance dependent) | Crystal load capacitance to check ($C_{CRYSTAL32}$).<br><br>Example: for a 32.768 kHz crystal with a load capacitance of $C_{CRYSTAL32}$ = 7 pF, external capacitors are required:<br>$C_{LEXT32}$ = 11 pF.<br>Refer to the Electrical Characteristics section. |
| XIN32<br>XOUT32<br>Slow Clock Oscillator in Bypass Mode | XIN32: external clock source<br>XOUT32: can be left unconnected | VDDIO square wave signal<br>External clock source up to 44 kHz<br>Duty Cycle: 40 to 60%<br>Refer to the Electrical Characteristics section. |
| XIN32<br>XOUT32<br>Slow Clock Oscillator Disabled | XIN32: can be left unconnected<br>XOUT32: can be left unconnected | Typical nominal frequency 32 kHz (internal 32 kHz RC oscillator)<br>Duty Cycle: 45 to 55%<br>Refer to the Electrical Characteristics section. |

### 59.2.2 Serial Wire Debug Interface

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| SWCLK/TCK | Pullup (100 kOhm)(1)<br>If debug mode is not required, this pin can be used as GPIO. | Serial Wire Clock / Test Clock (Boundary scan mode only)<br>This pin is a Schmitt trigger input.<br>No internal pullup resistor at reset. |
| SWDIO/TMS | Pullup (100 kOhm) (1)<br>If debug mode is not required, this pin can be used as GPIO. | Serial Wire Input-Output / Test Mode Select (Boundary scan mode only).<br>This pin is a Schmitt trigger input.<br>No internal pullup resistor at reset. |
| TDI | Floating.<br>If boundary mode is not required, this pin can be used as GPIO. | Test Data In (Boundary scan mode only)<br>This pin is a Schmitt trigger input.<br>No internal pullup resistor at reset. |

| .....continued | | |
|---|---|---|
| **Signal Name** | **Recommended Pin Connection** | **Description** |
| TRACESWO/TDO | Floating.<br>If boundary mode is not required, this pin can be used as GPIO. | Test Data Out (Boundary scan mode only)<br>Output driven at up to VDDIO |
| JTAGSEL | In harsh environments(2), it is strongly recommended to tie this pin to GND if not used or to add an external low-value resistor (such as 1 kOhm). | JTAG Selection.<br>Internal permanent pulldown resistor to GNDBU (15 kOhm).<br>Must be tied to VDDIO to enter JTAG Boundary Scan with TST tied to VDDIO and PD0 tied to GND. |

**Figure 59-1. SWD Schematic Example with a 10-pin Connector**

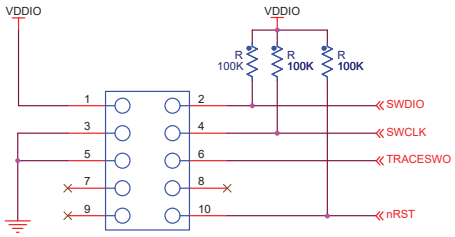
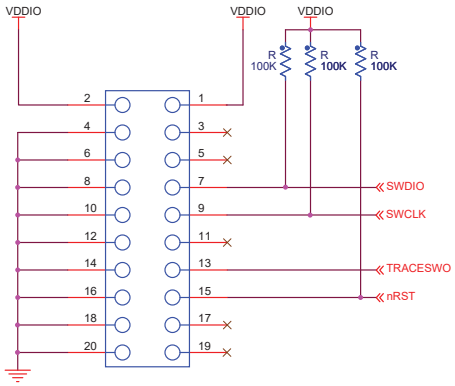



**Figure 59-2. SWD Schematic Example with a 20-pin Connector**




**Notes:**

1. These values are given only as a typical example.
2. In a well-shielded environment subject to low magnetic and electric field interference, the pin may be left unconnected. In noisy environments, a connection to ground is recommended.

### 59.2.3 Flash Memory

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| ERASE | If ERASE mode is not required, this pin can be used as GPIO. | Low level at startup is mandatory when not used. |

### 59.2.4 Reset and Test Pins

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| NRST | Application dependent.<br>Can be connected to a push button for hardware reset. | NRST is a bidirectional pin (Schmitt trigger input).<br>It is handled by the on-chip reset controller and can be driven low to provide a reset signal to the external components or asserted low externally to reset the microcontroller.<br>By default, the user reset is enabled after a general reset so that it is possible for a component to assert low and reset the microcontroller.<br>A permanent internal pullup resistor to VDDIO (100 kOhm) is available for user reset and external reset control. |
| TST | TST pin can be left unconnected in normal mode.<br>To enter in FFPI mode, TST pin must be tied to VDDIO.<br>In harsh environments (1), it is strongly recommended to tie this pin to GND if not used or to add an external low-value resistor (such as 10 kOhm). | This pin is a Schmitt trigger input.<br>Permanent internal pulldown resistor to GND (15 kOhm).<br>Must be tied to VDDIO to enter JTAG Boundary Scan, with JTAGSEL tied to VDDIO and PD0 tied to GND. |

Note: 1. In a well-shielded environment subject to low magnetic and electric field interference, the pin may be left unconnected. In noisy environments, a connection to ground is recommended.

### 59.2.5 PIOs

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| PAx<br>PBx<br>PCx<br>PDx<br>PEx | Application dependent.<br>(Pullup at VDDIO if needed) | All PIOs are pulled-up inputs (100 kOhm) at reset except those which are multiplexed with Oscillators Drivers and Debug interface that require to be enabled as peripherals:<br>Refer to the column “Reset State” of the pin description tables in the section "Package and Pinout".<br>Schmitt trigger on all inputs.<br>To reduce power consumption if not used, the concerned PIO can be configured as an output, driven at ‘0’ with internal pullup disabled. |

### 59.2.6 Parallel Capture Mode

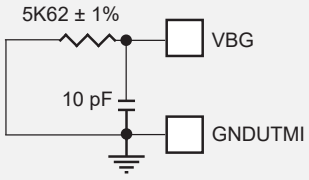
| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| PIODC0–7 | Application dependent.<br>(Pullup at VDDIO) | Parallel mode capture data<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |
| PIODCCLK | Application dependent.<br>(Pullup at VDDIO) | Parallel mode capture clock<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| PIODCEN1–2 | Application dependent.<br>(Pullup at VDDIO) | Parallel mode capture mode enable<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |

© 2021 Microchip Technology Inc.
and its subsidiaries

### 59.2.7 Analog Reference, Analog Front-End and DAC

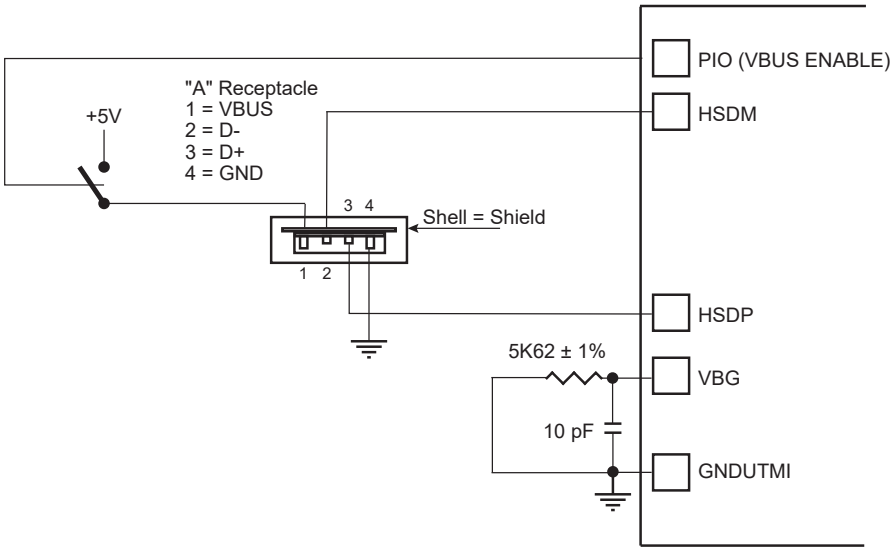
| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| Analog Voltage References | | |
| VREFP | 2.0V to VDDIN<br>LC Filter is required. | Positive reference voltage.<br>VREFP is a pure analog input.<br>VREFP is the voltage reference for the AFEC (ADC, PGA DAC and Analog Comparator).<br>To reduce power consumption, if analog features are not used, connect VREFP to GND.<br>Noise must be lower than 100 µVrms |
| VREFN | Analog Negative Reference | AFE, DAC and Analog Comparator negative reference<br>VREFN must be connected to GND or GNDANA. |
| 12-bit Analog Front-End | | |
| AFEx_AD0–<br>AFEx_AD11 | 0 to VREFP | AFE inputs channels<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |
| AFEx_ADTRG | Application dependent.<br>(Pulled-up on VREFP) | AFE external trigger input<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |
| 12-bit Digital-to-Analog Converter | | |
| DAC0–DAC1 | Application dependent.<br>0 to VREFP | Analog output<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |
| DATRG | Application dependent. | DAC external trigger input<br>Pulled-up input (100 kOhm) to VDDIO at reset. |

### 59.2.8 USB Host/Device

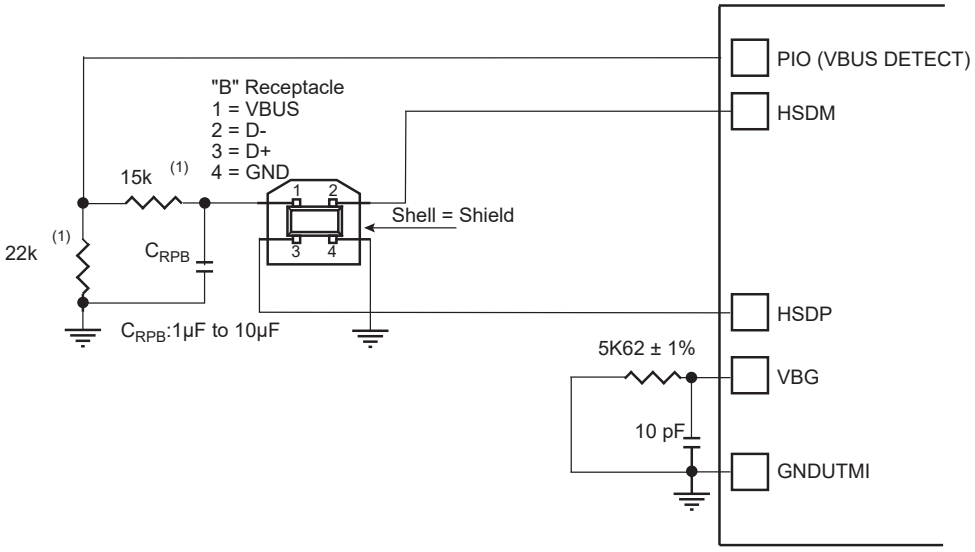
| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| VBG | 0.9 - 1.1V (1) (2) | Bias Voltage Reference for USB<br>To reduce the noise on the VBG pin to a minimum, implement the layout considerations below:<br>- Keep the VBG path as short as possible<br>- Ensure a ground connection to GNDUTMI<br><br>VBG can be left unconnected if USB is not used. |
| HSDM / HSDP | Application dependent (1) (2) | USB High Speed Data<br>Pull-down output at reset. |

**Notes:**

1. The following schematic shows an example of USB High Speed host connection. For more information, refer to the section *USB High-Speed Interface (USBHS)*.




2. The following schematic shows a typical USB High Speed device connection: For more information, refer to the section *USB High-Speed Interface (USBHS)*.




**Note:** The values shown on the 22 kΩ and 15 kΩ resistors are only valid with 3.3V supplied PIOs.

### 59.2.9 Memory Controllers

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| External Bus Interface | | |
| D[15:0] | Application dependent. | Data Bus (D0 to D15)<br>All data lines are pullup inputs to VDDIO at reset. |
| A[23:0] | Application dependent. | Address Bus (A0 to A23)<br>All address lines pullup inputs to VDDIO at reset. |
| NWAIT | Application dependent. | External Wait Signal.<br>Pulled-up input (100 kOhm) to VDDIO at reset. |

.....continued

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| Static Memory Controller | | |
| NCS0-NCS3 | Application dependent.<br>(Pullup at VDDIO) | Chip Select Lines<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |
| NRD | Application dependent. | Read Signal<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| NWE | Application dependent. | Write Enable<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |
| NWR0–NWR1 | Application dependent. | Write Signals<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |
| NBS0–NBS1 | Application dependent. | Byte Mask Signals<br>All are pulled-up inputs (100 kOhm) to VDDIO at reset. |
| NAND Flash Logic | | |
| NANDOE | Application dependent. | NAND Flash Output Enable<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| NANDWE | Application dependent. | NAND Flash Write Enable<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| SDR-SDRAM Controller Logic | | |
| SDCK | Application dependent. | SDRAM Clock<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| SDCKE | Application dependent. | SDRAM Clock Enable<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| SDCS | Application dependent.<br>(Pullup at VDDIO) | SDRAM Controller Chip Select<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| BA0–BA1 | Application dependent. | Bank Select<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| SDWE | Application dependent. | SDRAM Write Enable<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| RAS–CAS | Application dependent. | Row and Column Signal<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| SDA10 | Application dependent. | SDRAM Address 10 Line<br>Pulled-up input (100 kOhm) to VDDIO at reset. |

© 2021 Microchip Technology Inc.
and its subsidiaries

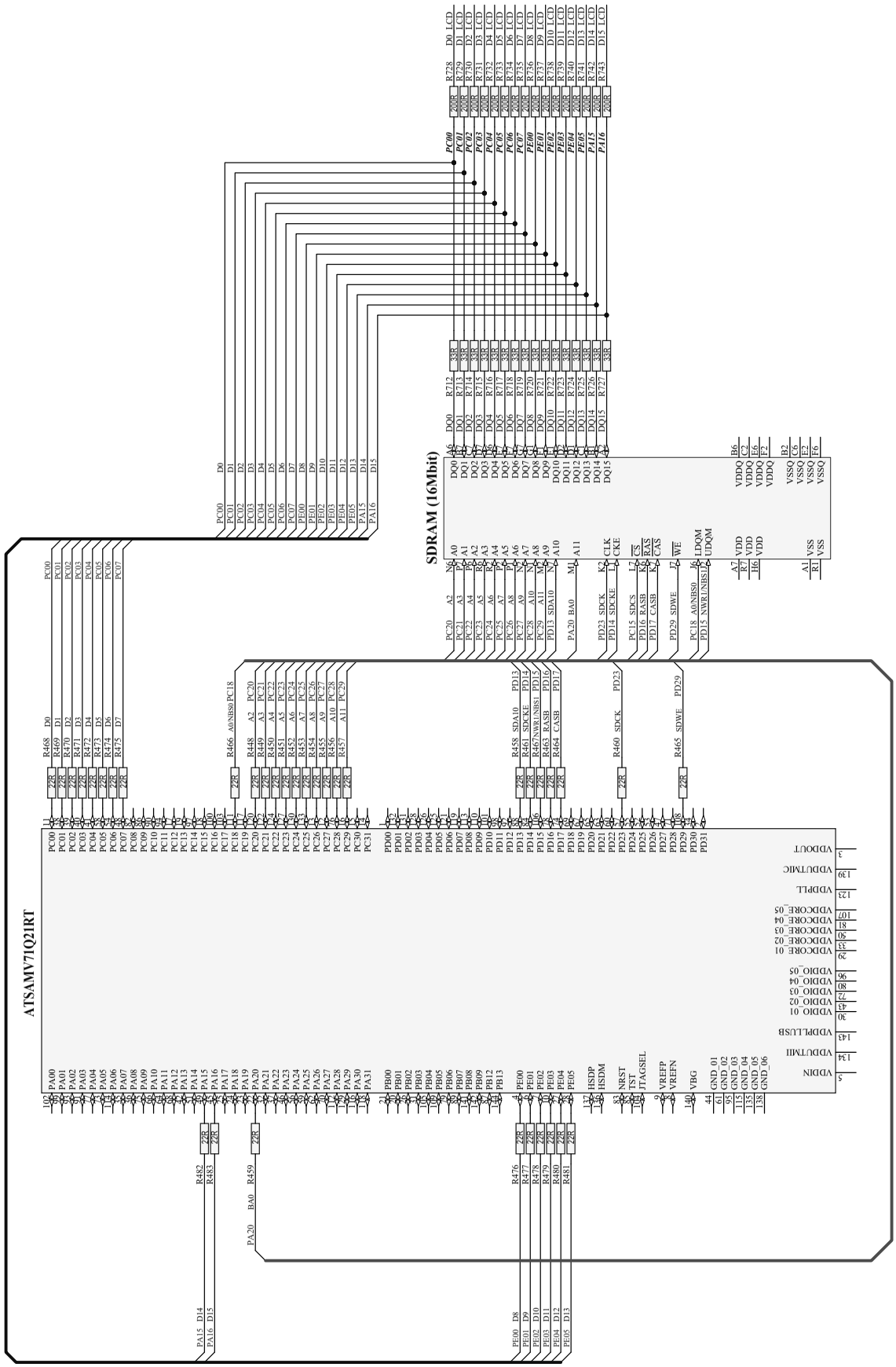
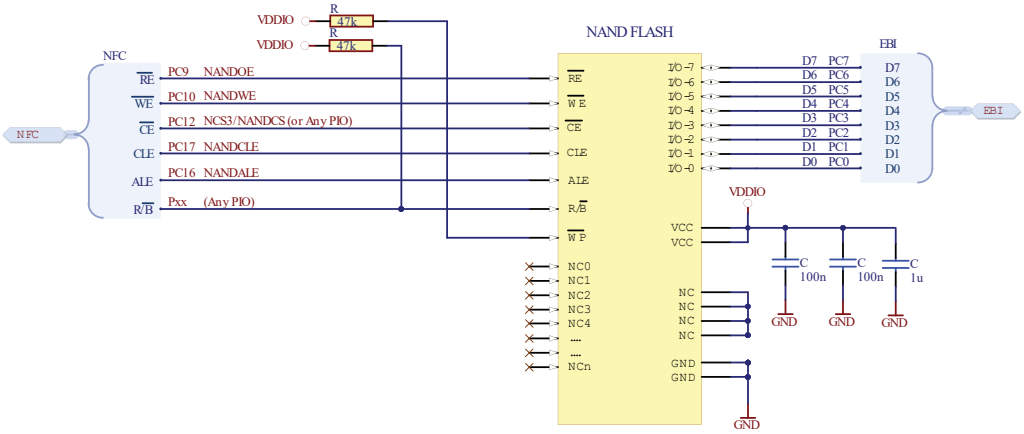



**Figure 59-3. Schematic Example with a 16 Mb/16-bit SDRAM**

**Figure 59-4. Schematic Example with a 2 Gb/8-bit NAND Flash**

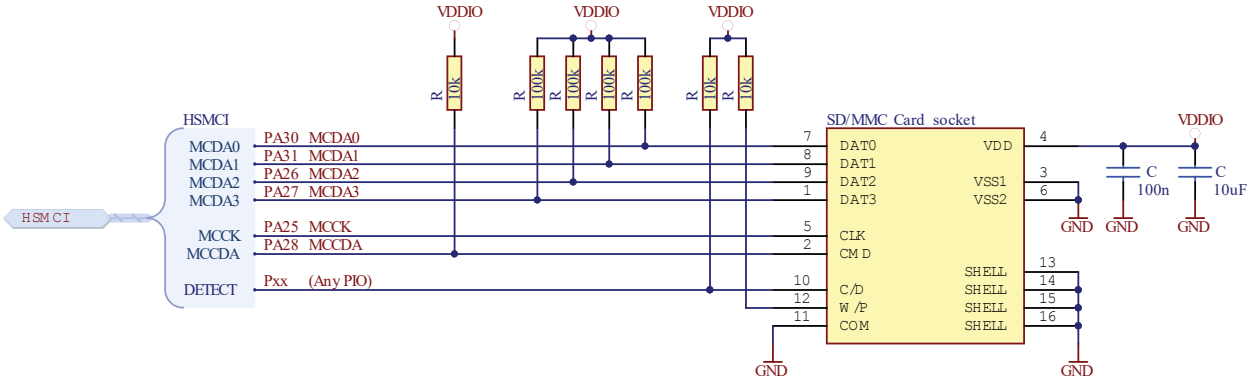



**Note:** For more details on the pin configuration of the EBI, refer to Table 33-3.

### 59.2.10 High-Speed Multimedia Card Interface (HSMCI)

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| MCCK | Application dependent | Multimedia Card Clock<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| MCCDA | Application dependent<br>(Pullup at VDDIO) | Multimedia Card Slot A Command<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| MCDA0–MCDA3 | Application dependent<br>(Pullup at VDDIO) | Multimedia Card Slot A Data<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |

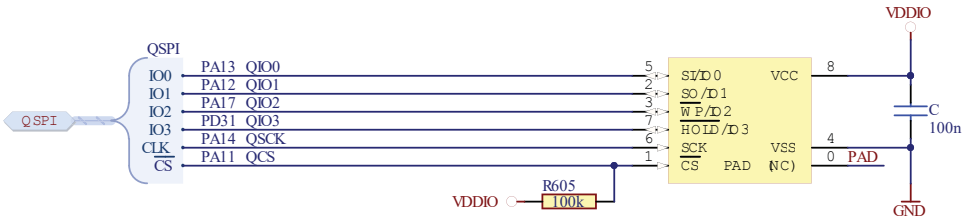
**Figure 59-5. Schematic Example with SD/MMC Card Interface**




### 59.2.11 QSPI Interface

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| QSCK | Application dependent. | QSPI Serial Clock<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| QCS | Application dependent.<br>(Pullup at VDDIO) | QSPI Chip Select<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| QIO0–QIO3 | Application dependent. | QSPI I/O<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |

**Figure 59-6. Schematic Example with QSPI Data Flash**




### 59.2.12 Other Interfaces

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| Universal Synchronous Asynchronous Receiver Transmitter | | |
| SCKx | Application dependent. | USARTx Serial Clock<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| TXDx | Application dependent. | USARTx Transmit Data<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| RXDx | Application dependent. | USARTx Receive Data<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| RTSx | Application dependent. | USARTx Request To Send<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| CTSx | Application dependent. | USARTx Clear To Send<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| DTRx | Application dependent. | USARTx Data Terminal Ready<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| DSRx | Application dependent. | USARTx Data Set Ready<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| DCDx | Application dependent. | USARTx Data Carrier Detect<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| RIx | Application dependent. | USARTx Ring Indicator<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| LONCOL1 | Application dependent. | LON Collision Detection<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| Synchronous Serial Controller | | |
| TD | Application dependent. | SSC Transmit Data<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| RD | Application dependent. | SSC Receive Data<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| TK | Application dependent. | SSC Transmit Clock<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| RK | Application dependent. | SSC Receive Clock I<br>Pulled-up input (100 kOhm) to VDDIO at reset. |

© 2021 Microchip Technology Inc.
and its subsidiaries

.....continued

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| TF | Application dependent. | SSC Transmit Frame Sync<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| RF | Application dependent. | SSC Receive Frame Sync<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| Image Sensor Interface | | |
| ISI_D0–ISI_D11 | Application dependent.<br>(Signal can be level-shifted depending on the image sensor characteristics) | Image Sensor Data<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| ISI_MCK | Application dependent.<br>(Signal can be level-shifted depending on the image sensor characteristics) | Image sensor reference clock. No dedicated signal, PCK1 can be used.<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| ISI_HSYNC | Application dependent.<br>(Signal can be level-shifted depending on the image sensor characteristics) | Image sensor horizontal synchro<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| ISI_VSYNC | Application dependent.<br>(Signal can be level-shifted depending on the image sensor characteristics) | Image sensor vertical synchro<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| ISI_PCK | Application dependent.<br>(Signal can be level-shifted depending on the image sensor characteristics) | Image sensor data clock<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| Timer/Counter | | |
| TCLKx | Application dependent. | TC Channel x External Clock Input<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| TIOAx | Application dependent. | TC Channel x I/O Line A<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| TIOBx | Application dependent. | TC Channel x I/O Line B<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| Pulse Width Modulation Controller | | |
| PWMC0_PWMHx<br>PWMC1_PWMHx | Application dependent. | Waveform Output High for Channel x<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| PWMC0_PWMLx<br>PWMC1_PWMLx | Application dependent. | Waveform Output Low for Channel x<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| PWMC0_PWMFI0–<br>PWMC0_PWMFI2<br>PWMC1_PWMFI0–<br>PWMC1_PWMFI2 | Application dependent. | Fault Inputs<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |

.....continued

| Signal Name | Recommended Pin Connection | Description |
|---|---|---|
| PWMC0_PWMEXTRG0<br>PWMC0_PWMEXTRG1<br>PWMC1_PWMEXTRG0<br>PWMC1_PWMEXTRG1 | Application dependent. | External Trigger Inputs<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| Serial Peripheral Interface | | |
| SPIx_MISO | Application dependent. | Master In Slave Out<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| SPIx_MOSI | Application dependent. | Master Out Slave In<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| SPIx_SPCK | Application dependent. | SPI Serial Clock<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| SPIx_NPCS0 | Application dependent.<br>(Pullup at VDDIO) | SPI Peripheral Chip Select 0<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| SPIx_NPCS1–<br>SPIx_NPCS3 | Application dependent.<br>(Pullup at VDDIO) | SPI Peripheral Chip Select<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| Two-Wire Interface | | |
| TWDx | Application dependent.<br>(4.7kOhm Pulled-up on VDDIO) | TWIx Two-wire Serial Data<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| TWCKx | Application dependent.<br>(4.7kOhm Pulled-up on VDDIO) | TWIx Two-wire Serial Clock<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| Fast Flash Programming Interface | | |
| PGMEN0–PGMEN1 | To enter in FFPI mode TST pins must be tied to VDDIO. | Programming Enabling<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| PGMM0–PGMM3 | Application dependent. | Programming Mode<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| PGMD0–PGMD15 | Application dependent. | Programming Data<br>Pulled-up inputs (100 kOhm) to VDDIO at reset. |
| PGMRDY | Application dependent. | Programming Ready<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| PGMNVALID | Application dependent. | Data Direction<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| PGMNOE | Application dependent. | Programming Read<br>Pulled-up input (100 kOhm) to VDDIO at reset. |
| PGMNCMD | Application dependent. | Programming Command<br>Pulled-up input (100 kOhm) to VDDIO at reset. |

## 59.3 Boot Program Hardware Constraints

Refer to 17. SAM-BA Boot Program for more details on the boot program.

### 59.3.1 Boot Program Supported Crystals (MHz)

A 12 MHz or a 16 MHz crystal or external clock (in Bypass mode) is mandatory in order to generate USB and PLL clocks correctly for the following boots.

### 59.3.2 SAM-BA Boot

The SAM-BA Boot Assistant supports serial communication via the UART or USB device port:

- UART0 hardware requirements: None
- USB Device hardware requirements: Not supported

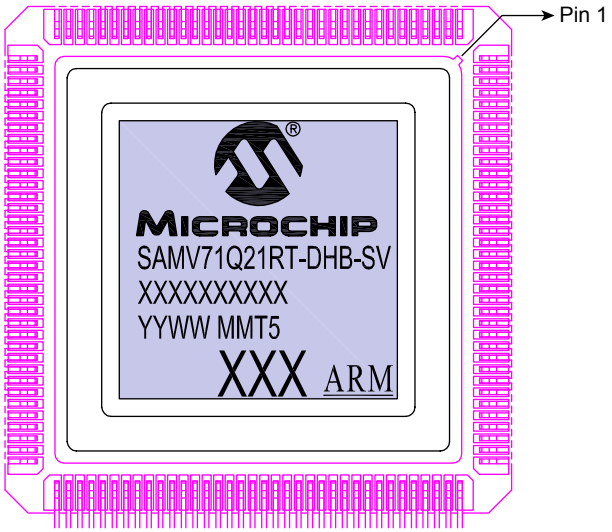
**Table 59-1. Pins Driven During SAM-BA Boot Program Execution**

| Peripheral | Pin |
|---|---|
| UART0 | URXD0 |
| UART0 | UTXD0 |

# 60. Marking

## 60.1 CQFP144

Top marking follows the example shown below:




where:

Line 1 = Microchip Logo

Line 2 = Device Name

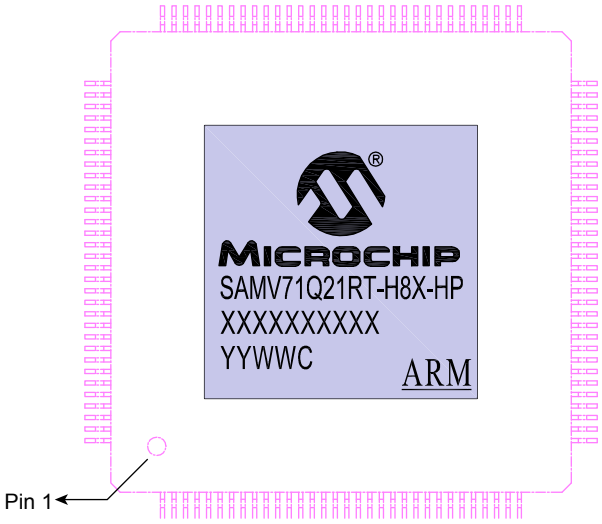
Line 3 = Lot Reference Number

Line 4 = Date Code (YYWW), Location Code

Line 5 = Serialization, ARM

## 60.2 LQFP144

Top marking follows the example shown below:




where:

Line 1 = Microchip Logo

Line 2 = Device Name

Line 3 = Lot Reference Number

Line 4 = Date Code (YYWW), Location Code

Line 5 = ARM

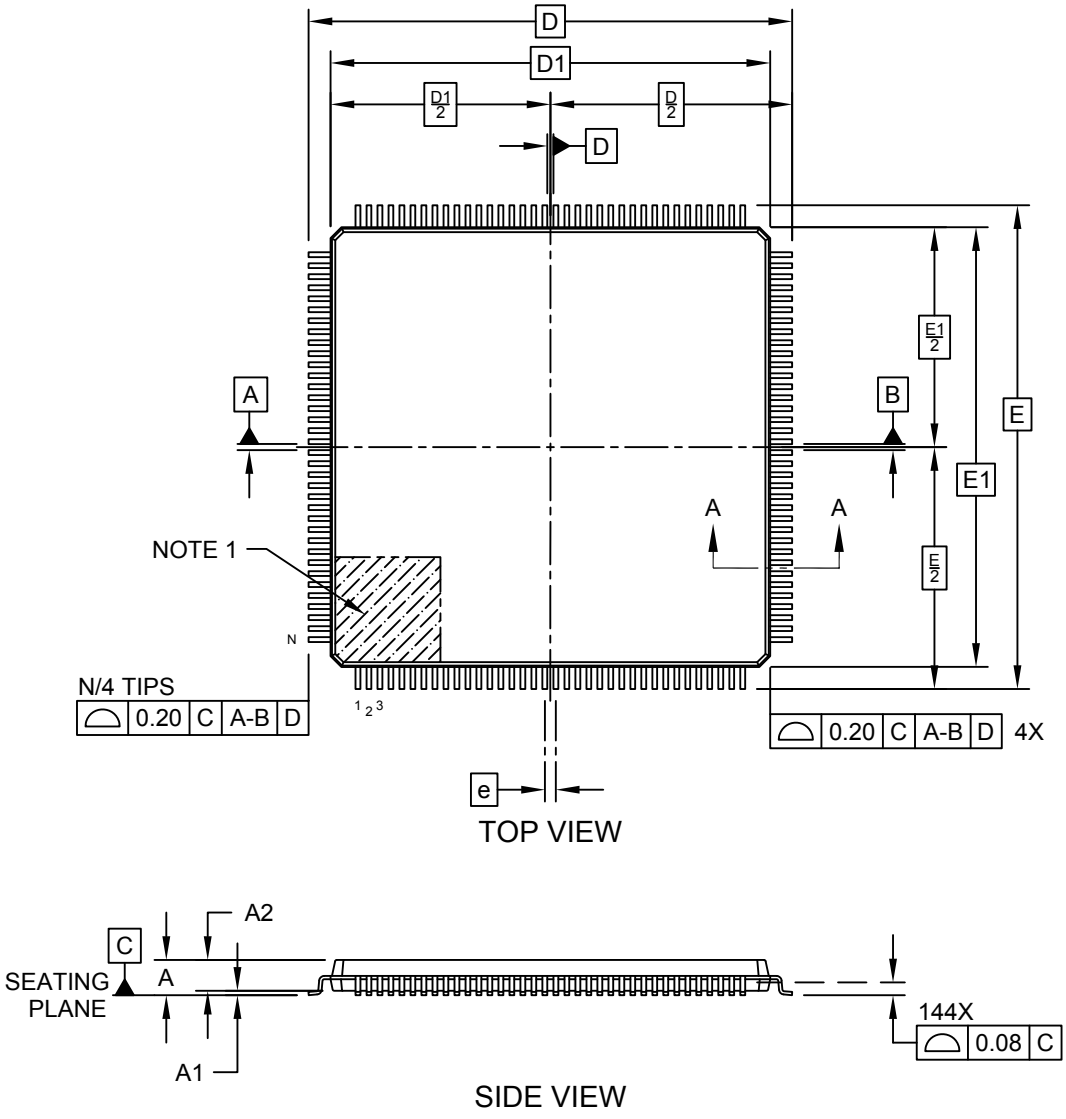
© 2021 Microchip Technology Inc.
and its subsidiaries

# 61. Mechanical Characteristics

## 61.1 144-Lead Plastic Quad Flatpack (2SB) - 20x20x1.4 mm Body [LQFP] Atmel Legacy Global Package Code AEI

**Figure 61-1. 144-pin LQFP Package Mechanical Drawing**

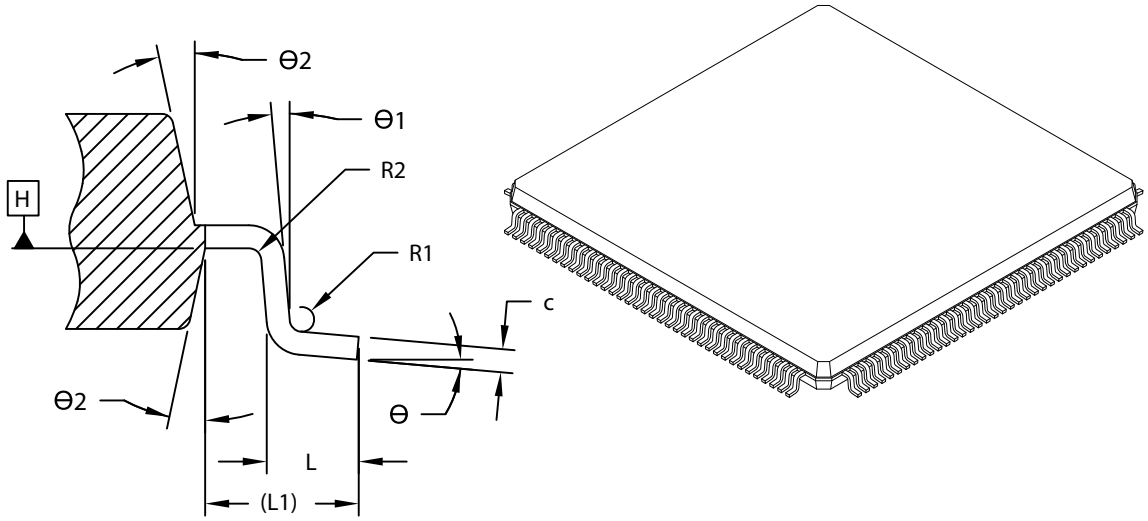
| **Note:** | For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging |
|---|---|




Microchip Technology Drawing C04-21010 Rev A Sheet 1 of 2

## 144-Lead Plastic Quad Flatpack (2SB) - 20x20x1.4 mm Body [LQFP]
## Atmel Legacy Global Package Code AEI

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




SECTION A-A

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Terminals | N | | 144 | |
| Pitch | e | | 0.50 BSC | |
| Overall Height | A | - | - | 1.60 |
| Standoff | A1 | 0.05 | 0.02 | 0.15 |
| Molded Plastic Height | A2 | 1.35 | 1.40 | 1.45 |
| Overall Length | D | | 22.00 BSC | |
| Exposed Pad Length | D1 | | 20.00 BSC | |
| Overall Width | E | | 22.00 BSC | |
| Exposed Pad Width | E1 | | 20.00 BSC | |
| Terminal Width | b | 0.17 | 0.22 | 0.27 |
| Terminal Width | c | 0.09 | 0.15 | 0.20 |
| Terminal Length | L | 0.45 | 0.60 | 0.75 |
| Footprint | L1 | | 1.00 REF | |
| Terminal Bend Radius | R1 | 0.08 | - | - |
| Terminal Bend Radius | R2 | 0.08 | - | 0.20 |
| Terminal Angle | Θ | 0° | 3.5° | 7° |
| Terminal Angle | Θ1 | 0° | - | - |
| Mold Draft Angle | Θ2 | 11° | 12° | 13° |

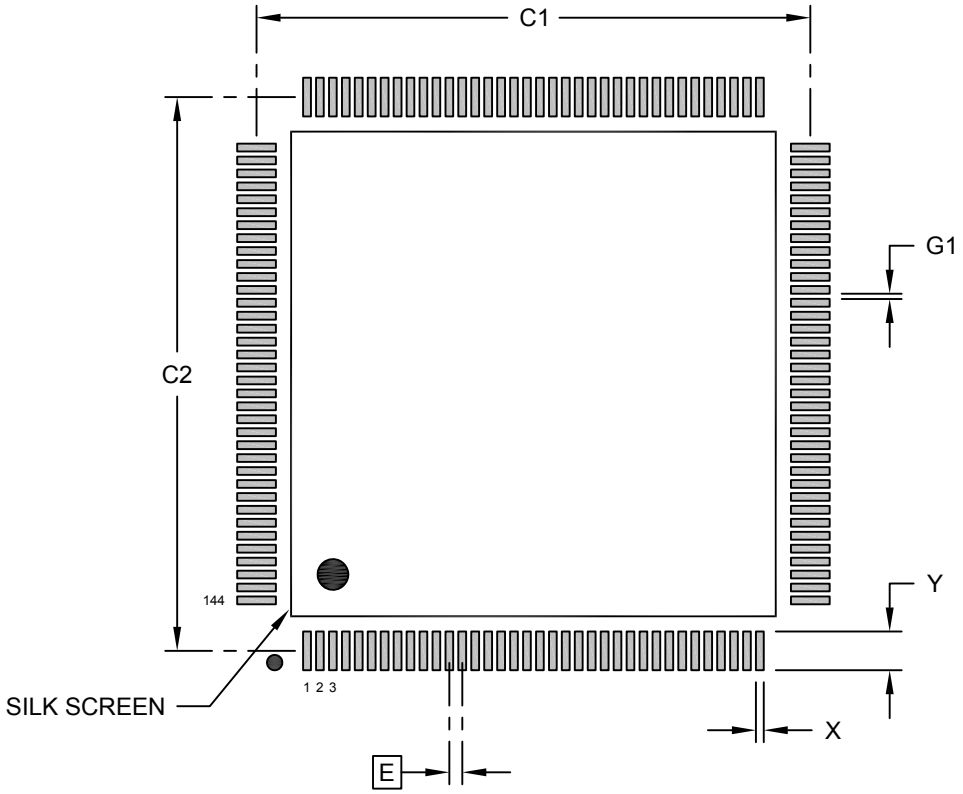
Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-21010 Rev A Sheet 2 of 2

### 144-Lead Plastic Quad Flatpack (2SB) - 20x20x1.4 mm Body [LQFP]
### Atmel Legacy Global Package Code AEI

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging




RECOMMENDED LAND PATTERN

| Units | | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.50 BSC | |
| Contact Pad Spacing | C1 | | 21.40 | |
| Contact Pad Spacing | C2 | | 21.40 | |
| Contact Pad Width (X144) | X1 | | | 0.30 |
| Contact Pad Length (X144) | Y1 | | | 1.50 |
| Contact Pad to Contact Pad (X140) | G1 | 0.20 | | |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-23010 Rev A

**Table 61-1. Device and LQFP Package Maximum Weight**

| | |
|---|---|
| 1365 | mg |

**Table 61-2. LQFP Package Reference**

| | |
|---|---|
| JEDEC Drawing Reference | JEDEC |
| J-STD-609 Classification | e3 |

**Table 61-3. LQFP Package Characteristics**

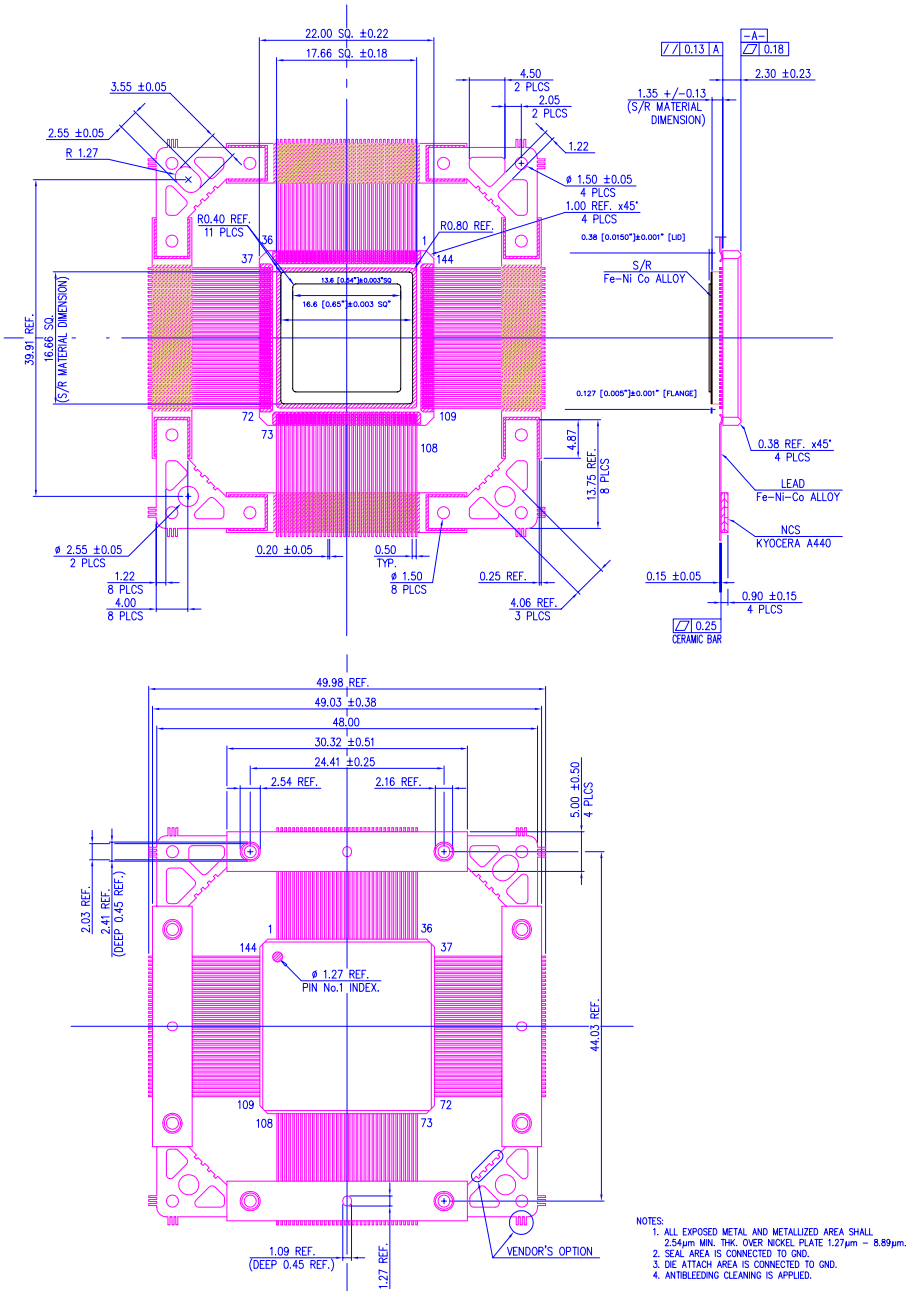
| | |
|---|---|
| Moisture Sensitivity Level | 3 |

This package respects the recommendations of the NEMI User Group.

## 61.2 144-pin CQFP Package

**Note:**
For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging.

**Figure 61-2. 144-pin CQFP Package Mechanical Drawing**




### 61.2.1 Soldering Profile

Please refer to the device's Qualpack document, which is available upon request from your Microchip Technology sales office.

### 61.2.2 Packaging Resources

Land Pattern Definition.

Refer to the following IPC Standard:

- IPC-7351B (Generic Requirements for Surface Mount Design and Land Pattern Standards) www.ipc.org
- Microchip Technology Packaging Specification, www.microchip.com/packaging

# 62. Errata

SAMV71Q21RT is derived from the automotive product SAMV71Q21 Rev. B.

As such, it inherits the same product ID.

| Device Identification | |
|---|---|
| **CHPID_CIDR[31:0]** | **CHIPID_EXID[31:0]** |
| 0xA122_0E01 | 0x00000002 |

## 62.1 USB High-Speed (USBHS)

The USBHS module functionality is adversely affected by concurrent accesses of the SDRAM.

Therefore the USBHS and UTMI PLL functions cannot be used as they have not been qualified and are not supported on this device.

**Work around**
None.

## 62.2 Other Modules

Refer to the document *SAM-E70S70V70V71-Family-Errata-Sheet-80000767* available on the Microchip Web Site.



**Important:**
In *SAM-E70S70V70V71-Family-Errata-Sheet-80000767*, only the errata related to SAMV71Q21 Rev. B apply, with the exception of errata related to the USBHS module, which is not supported.

# 63. Revision History

## 63.1 Rev. D - 7/2021

| |
|---|
| Added the Hirel Ceramic option in the ordering information. |
| Updated the Features topic. |
| Updated the Configuration Summary topic. |
| Updated the Space Quality Grade topic. |
| Updated the Erase Pin topic. |

## 63.2 Rev. C - 8/2019

| |
|---|
| Extends minimum operating temperature to -55°C for all package types throughout the data sheet. |
| Removes previous temperature range for Flash programming total Write/Erase specifications. |
| Eliminates USB option for SAM-BA boot programming. |
| Moves Drive Level specification ($P_{ON}$) for 32.768 kHz crystal oscillator from Table 58-20 to Table 58-19. |
| Moves Drive Level specification ($P_{ON}$) for 3 to 20 MHz crystal oscillator from Table 58-23 to Table 58-22. |
| Other minor typographic corrections throughout the document. |

## 63.3 Rev. B - 6/2019

| |
|---|
| Adds information for LQPF 144-pin packages throughout the data sheet. |
| Removes Erase Memory Plane option from Enhanced Flash Controller. |
| Expands Erase Pages options to 4, 8, 16 and 32 for Enhanced Flash Controller. |
| Removes specifications for $T_J$ and $P_D$ from Table 58-2. |
| Changes minimum and maximum specifications for INL and DNL in Table 58-38. |
| Adds “Soldering Profile” to Section 61.3. |
| Replaces reference to Atmel documentation with Microchip packaging reference, in Section 61.4. |

## 63.4 Rev. A - 12/2018

| |
|---|
| Initial revision based on the data sheet *SAM-E70S70V70V71-Family-DataSheet-DS60001527C*. |

© 2021 Microchip Technology Inc.
and its subsidiaries

## The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

## Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

## Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

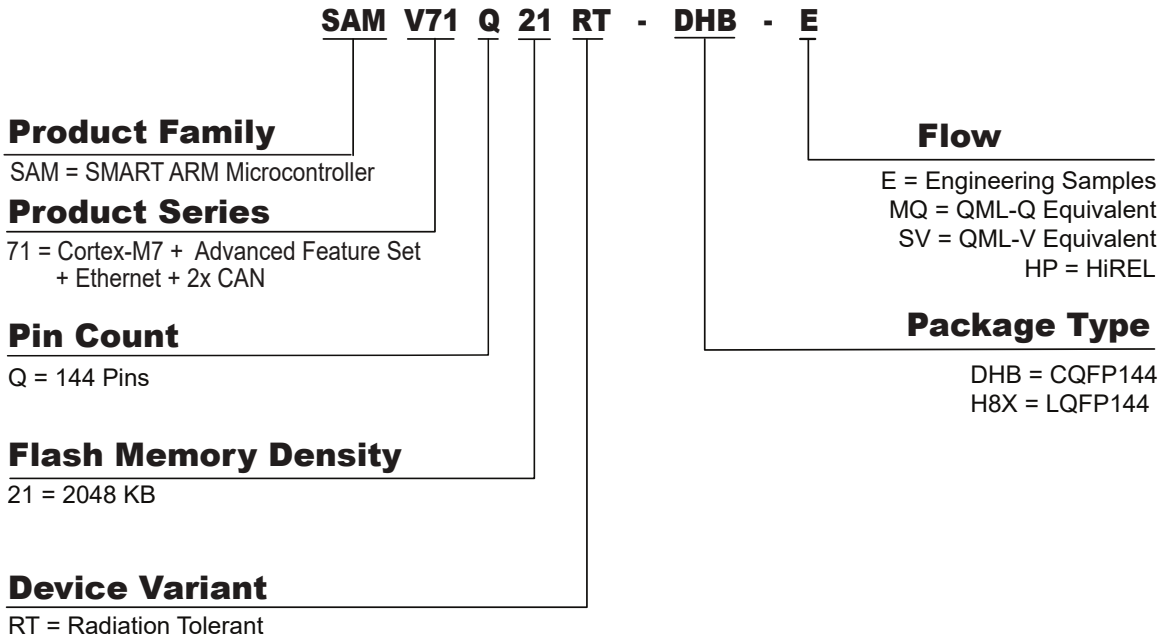
Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

© 2021 Microchip Technology Inc.
and its subsidiaries

## Product Identification System

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.




## Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

## Legal Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend,

© 2021 Microchip Technology Inc. and its subsidiaries

indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

## Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AnyRate, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PackeTime, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TempTrackr, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, FlashTec, Hyper Speed Control, HyperLight Load, IntelliMOS, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet-Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, Vite, WinPath, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BlueSky, BodyCom, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, INICnet, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2019, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-8708-1

## Quality Management System

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

# Worldwide Sales and Service


| AMERICAS | ASIA/PACIFIC | ASIA/PACIFIC | EUROPE |
|---|---|---|---|
| **Corporate Office**<br>2355 West Chandler Blvd.<br>Chandler, AZ 85224-6199<br>Tel: 480-792-7200<br>Fax: 480-792-7277<br>Technical Support:<br>www.microchip.com/support<br>Web Address:<br>www.microchip.com<br>**Atlanta**<br>Duluth, GA<br>Tel: 678-957-9614<br>Fax: 678-957-1455<br>**Austin, TX**<br>Tel: 512-257-3370<br>**Boston**<br>Westborough, MA<br>Tel: 774-760-0087<br>Fax: 774-760-0088<br>**Chicago**<br>Itasca, IL<br>Tel: 630-285-0071<br>Fax: 630-285-0075<br>**Dallas**<br>Addison, TX<br>Tel: 972-818-7423<br>Fax: 972-818-2924<br>**Detroit**<br>Novi, MI<br>Tel: 248-848-4000<br>**Houston, TX**<br>Tel: 281-894-5983<br>**Indianapolis**<br>Noblesville, IN<br>Tel: 317-773-8323<br>Fax: 317-773-5453<br>Tel: 317-536-2380<br>**Los Angeles**<br>Mission Viejo, CA<br>Tel: 949-462-9523<br>Fax: 949-462-9608<br>Tel: 951-273-7800<br>**Raleigh, NC**<br>Tel: 919-844-7510<br>**New York, NY**<br>Tel: 631-435-6000<br>**San Jose, CA**<br>Tel: 408-735-9110<br>Tel: 408-436-4270<br>**Canada - Toronto**<br>Tel: 905-695-1980<br>Fax: 905-695-2078 | **Australia - Sydney**<br>Tel: 61-2-9868-6733<br>**China - Beijing**<br>Tel: 86-10-8569-7000<br>**China - Chengdu**<br>Tel: 86-28-8665-5511<br>**China - Chongqing**<br>Tel: 86-23-8980-9588<br>**China - Dongguan**<br>Tel: 86-769-8702-9880<br>**China - Guangzhou**<br>Tel: 86-20-8755-8029<br>**China - Hangzhou**<br>Tel: 86-571-8792-8115<br>**China - Hong Kong SAR**<br>Tel: 852-2943-5100<br>**China - Nanjing**<br>Tel: 86-25-8473-2460<br>**China - Qingdao**<br>Tel: 86-532-8502-7355<br>**China - Shanghai**<br>Tel: 86-21-3326-8000<br>**China - Shenyang**<br>Tel: 86-24-2334-2829<br>**China - Shenzhen**<br>Tel: 86-755-8864-2200<br>**China - Suzhou**<br>Tel: 86-186-6233-1526<br>**China - Wuhan**<br>Tel: 86-27-5980-5300<br>**China - Xian**<br>Tel: 86-29-8833-7252<br>**China - Xiamen**<br>Tel: 86-592-2388138<br>**China - Zhuhai**<br>Tel: 86-756-3210040 | **India - Bangalore**<br>Tel: 91-80-3090-4444<br>**India - New Delhi**<br>Tel: 91-11-4160-8631<br>**India - Pune**<br>Tel: 91-20-4121-0141<br>**Japan - Osaka**<br>Tel: 81-6-6152-7160<br>**Japan - Tokyo**<br>Tel: 81-3-6880- 3770<br>**Korea - Daegu**<br>Tel: 82-53-744-4301<br>**Korea - Seoul**<br>Tel: 82-2-554-7200<br>**Malaysia - Kuala Lumpur**<br>Tel: 60-3-7651-7906<br>**Malaysia - Penang**<br>Tel: 60-4-227-8870<br>**Philippines - Manila**<br>Tel: 63-2-634-9065<br>**Singapore**<br>Tel: 65-6334-8870<br>**Taiwan - Hsin Chu**<br>Tel: 886-3-577-8366<br>**Taiwan - Kaohsiung**<br>Tel: 886-7-213-7830<br>**Taiwan - Taipei**<br>Tel: 886-2-2508-8600<br>**Thailand - Bangkok**<br>Tel: 66-2-694-1351<br>**Vietnam - Ho Chi Minh**<br>Tel: 84-28-5448-2100 | **Austria - Wels**<br>Tel: 43-7242-2244-39<br>Fax: 43-7242-2244-393<br>**Denmark - Copenhagen**<br>Tel: 45-4450-2828<br>Fax: 45-4485-2829<br>**Finland - Espoo**<br>Tel: 358-9-4520-820<br>**France - Paris**<br>Tel: 33-1-69-53-63-20<br>Fax: 33-1-69-30-90-79<br>**Germany - Garching**<br>Tel: 49-8931-9700<br>**Germany - Haan**<br>Tel: 49-2129-3766400<br>**Germany - Heilbronn**<br>Tel: 49-7131-72400<br>**Germany - Karlsruhe**<br>Tel: 49-721-625370<br>**Germany - Munich**<br>Tel: 49-89-627-144-0<br>Fax: 49-89-627-144-44<br>**Germany - Rosenheim**<br>Tel: 49-8031-354-560<br>**Israel - Ra'anana**<br>Tel: 972-9-744-7705<br>**Italy - Milan**<br>Tel: 39-0331-742611<br>Fax: 39-0331-466781<br>**Italy - Padova**<br>Tel: 39-049-7625286<br>**Netherlands - Drunen**<br>Tel: 31-416-690399<br>Fax: 31-416-690340<br>**Norway - Trondheim**<br>Tel: 47-72884388<br>**Poland - Warsaw**<br>Tel: 48-22-3325737<br>**Romania - Bucharest**<br>Tel: 40-21-407-87-50<br>**Spain - Madrid**<br>Tel: 34-91-708-08-90<br>Fax: 34-91-708-08-91<br>**Sweden - Gothenberg**<br>Tel: 46-31-704-60-40<br>**Sweden - Stockholm**<br>Tel: 46-8-5090-4654<br>**UK - Wokingham**<br>Tel: 44-118-921-5800<br>Fax: 44-118-921-5820 |


© 2021 Microchip Technology Inc. and its subsidiaries