SINO WEALTH

SH32F9001_32F9801

基于ARM® Cortex™-M3核心的32位微控制器

1. 特性

- 内核：ARM 32位的Cortex-M3 CPU
  - 最高72MHz工作频率
  - 支持硬件乘/除法
  - 高速存储器和指令加速，提高执行效率
- 存储器
  - 最高256K/128K字节FLASH程序存储器
  - 1K字节高速缓冲Cache
  - 16K字节SRAM+16K字节CRAM（Code RAM）
  - 6K字节内置类EEPROM区+6K字节BOOT ROM
  - 4K字节的厂商代码选项区（包含1024字节OTP区）
- 时钟、复位和电源管理
  - 2.0V~5.5V数字供电（VDD）和I/O引脚
  - 上电复位(POR)、掉电检测(BOD)、低电压复位(LVR)
  - 硬件独立看门狗(IWDT)、窗口看门狗(WWDT)
  - 4M~12MHz晶体振荡器/陶瓷振荡器
  - 32kHz晶体振荡器
  - 内嵌高频RC振荡器：24MHz（全温度1%精度）
  - 内嵌低频RC振荡器：128kHz
  - 内嵌PLL产生高频时钟
  - 时钟安全监控（CSM）
- 低功耗模式
  - 睡眠模式（仅CPU停止）
  - 停机模式（CPU停止，外设时钟切断；退出ShutDown后，系统时钟恢复成进入停机模式前的状态）
- 24通道12位高速模数转换器
  - 1MSPS最高采样速率
  - 支持单次/连续/同步/独立采样模式和事件触发
- 1个DMA控制器
  - 8个独立的DMA通道
  - 支持内存到内存、内存到外设、外设到内存传输
- 最多75个快速GPIO端口
  - 最多75个可用GPIO口
  - 所有GPIO内建上拉/下拉电阻
  - 12个引脚提供极强灌/拉电流能力
  - 引脚兼容TTL电平输入
- 16个外部中断输入通道（EXTI0~EXTI15）
  - 任何一个GPIO都可映射为外部中断输入
  - 支持边沿触发与电平触发、支持软件触发外部中断
- 中断源
  - 16个Cortex-M3中断
  - TIM5~8，PCA0~7，PWM0~3
  - 外部中断0~15，UART0~5，SPI0/1，TWI0/1
  - ADC，LED
  - CSM，BOD，DMA，CRC
- LED驱动器
  - 3~12COM X 16（1/1 ~1/12占空比）
  - 支持共阴共阳模式可选
  - 支持256级调光
- LCD驱动器
  - 8 X 36段（1/8占空比，1/4偏置）
  - 6 X 38段（1/6占空比，1/4偏置或者1/3偏置）
  - 5 X 39段（1/5占空比，1/3偏置）
  - 4 X 40段（1/4占空比，1/3偏置）
- 8个各包含两个比较/捕捉单元的16位PCA0~7
  - 可级联组合成4个32位PCA
- 4路16位带死区/支持互补输出的PWM0/1/2/3
- 4个16位基本定时器/计数器（TIM5/TIM6/TIM7/TIM8）
  - 可级联组合成2个32位定时器/计数器
  - 自动重载累加计数器
- 1个24位自减型系统时间定时器（SYSTICK）
- 多个串行通信接口
  - 6个通用异步串行口（UART0/1/2/3/4/5）
  - 2个独立串行外设接口（SPI0/1）
  - 2个独立两线串行接口（TWI0/1）
- 丰富的片上模拟功能模块（AMOC）
  - 内部温度传感器，支持校准操作
  - 内部稳压源，可产生多级比较基准电平
  - ADC内置参考电压，同时可输出作为外部电压基准
- CRC代码校验、CRC数据传输校验
- SRAM的March-C/March-X检测算法
- 96位芯片唯一识别码
- 标准SWJ烧写/仿真接口
  - 串行双线接口（SWDP）
- 工作环境温度：-40℃～+105℃
- 封装：
  - LQFP44, TQFP48, LQFP64, LQFP80

2. 概述

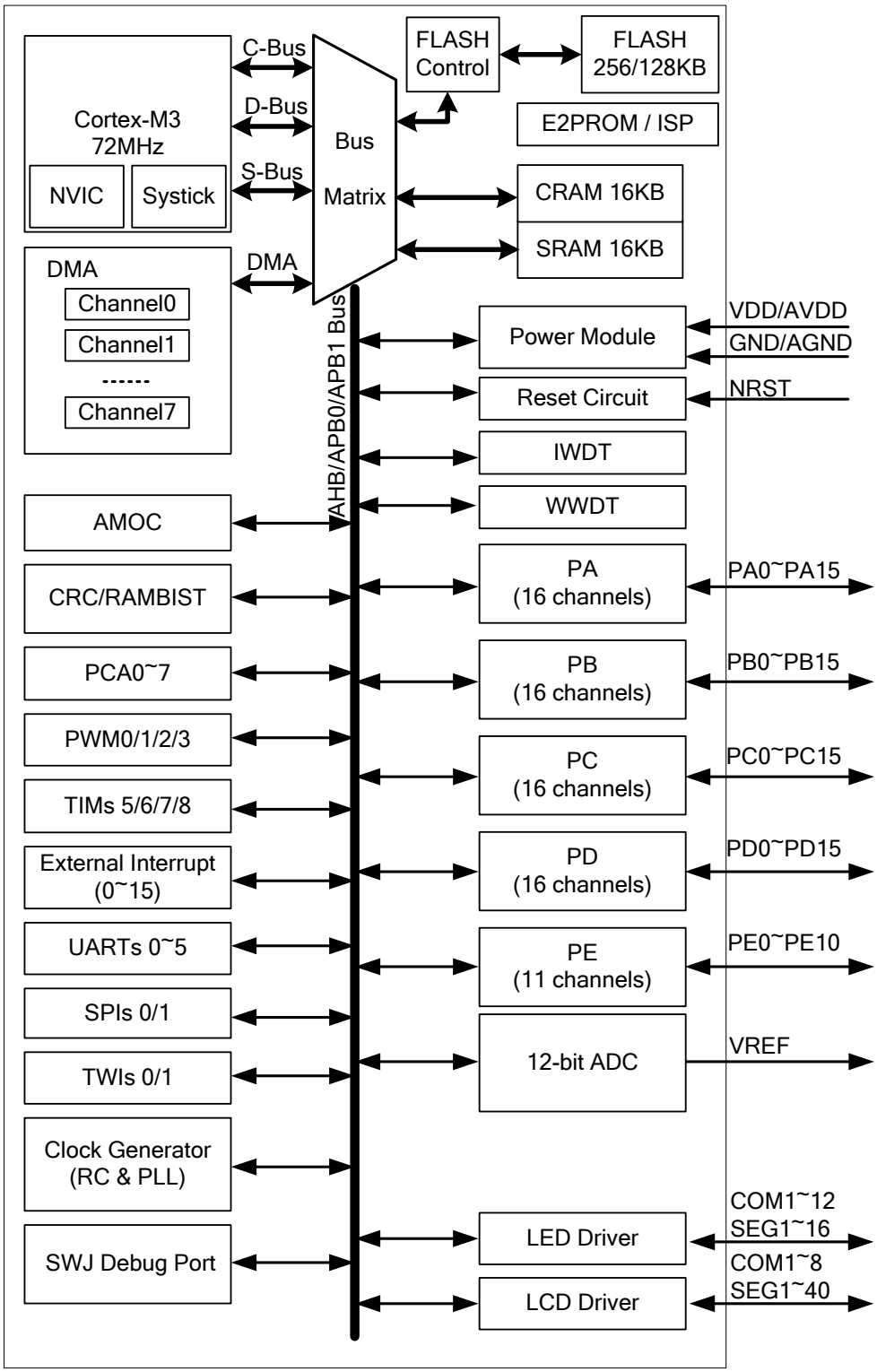
SH32F9001_32F9801是基于ARM Cortex-M3内核的高性能32位微控制器，支持72MHz最高主频，由于采用特殊结构的存储器指令加速模式，可使系统获得近乎零等待周期的高运行效率。

Cortex-M3内核具有成熟可靠的系统结构，高效的指令执行效率以及完备的生态系统，支持着传统控制系统向网络式智能化控制系统迈进。SH32F9001_32F9801充分发挥了Cortex-M3内核的优势，专注于提供高集成度、高运算能力的单片集成电路，提供高性价比的微控制器系统解决方案，特别适合开发智能家居、大家电以及电机控制方案。

在运算能力方面，Cortex-M3内核已内置了硬件32位乘法器、除法器和移位寄存器，具有强大的计算能力。同时SH32F9001_32F9801内置了8个可编程计数器阵列（PCA）和通用16位PWM定时器模块，集成了一个12位多通道高速ADC，使其应用范围更加广泛。

SH32F9001_32F9801支持低功耗应用模式，提供8/16/32位可选的CRC代码校验和数据校验，SRAM自检算法，内置独立看门狗、窗口看门狗、低电压复位电路、掉电检测电路等辅助模块，能够有效提升代码安全性与系统可靠性，可应用于高安规要求场合。

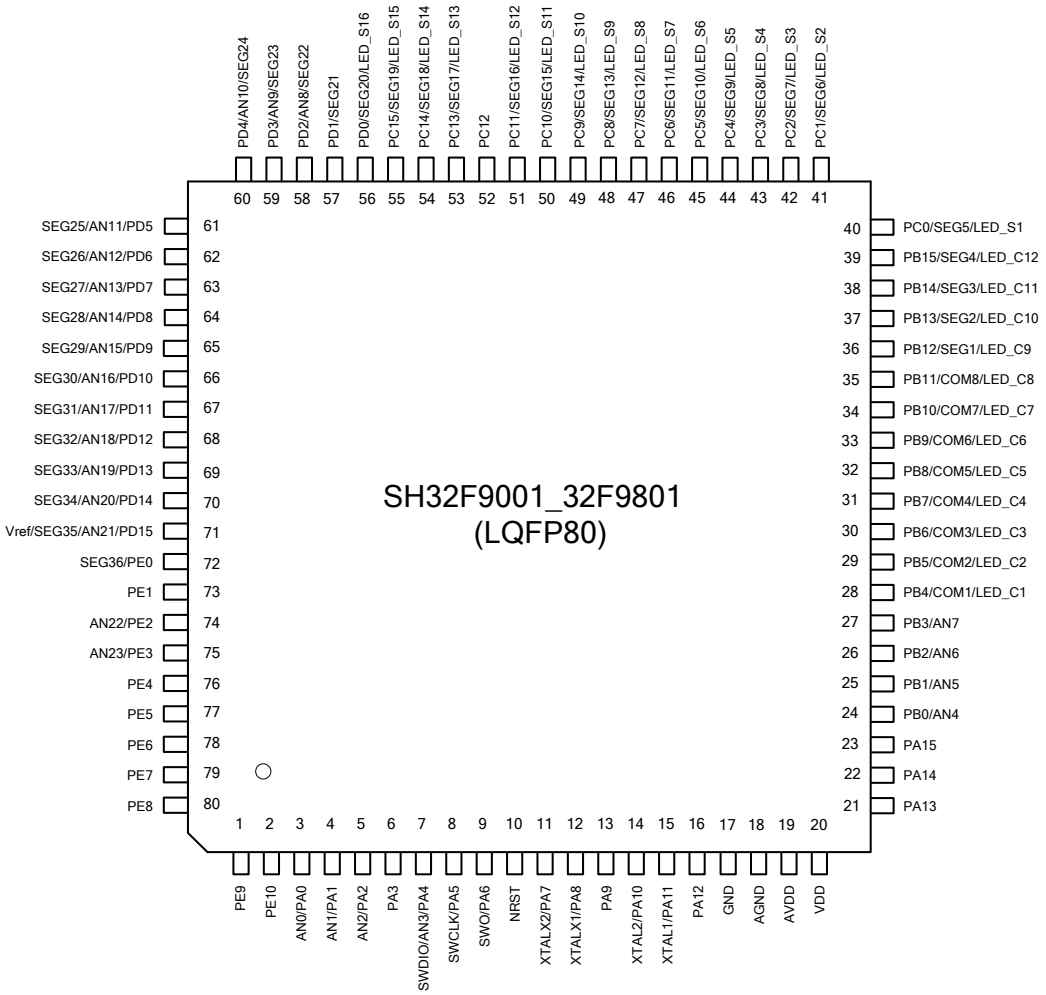
3. 方框图


Cortex-M3
72MHz
NVIC
Systick
C-Bus
D-Bus
S-Bus
DMA
Bus
Matrix
FLASH
Control
FLASH
256/128KB
E2PROM / ISP
CRAM 16KB
SRAM 16KB
DMA
Channel0
Channel1

Channel7
AHB/APB0/APB1 Bus
Power Module
Reset Circuit
IWDT
WWDT
VDD/AVDD
GND/AGND
NRST
AMOC
CRC/RAMBIST
PCA0~7
PWM0/1/2/3
TIMs 5/6/7/8
External Interrupt
(0~15)
UARTs 0~5
SPIs 0/1
TWIs 0/1
Clock Generator
(RC & PLL)
SWJ Debug Port
PA
(16 channels)
PB
(16 channels)
PC
(16 channels)
PD
(16 channels)
PE
(11 channels)
12-bit ADC
LED Driver
LCD Driver
PA0~PA15
PB0~PB15
PC0~PC15
PD0~PD15
PE0~PE10
VREF
COM1~12
SEG1~16
COM1~8
SEG1~40

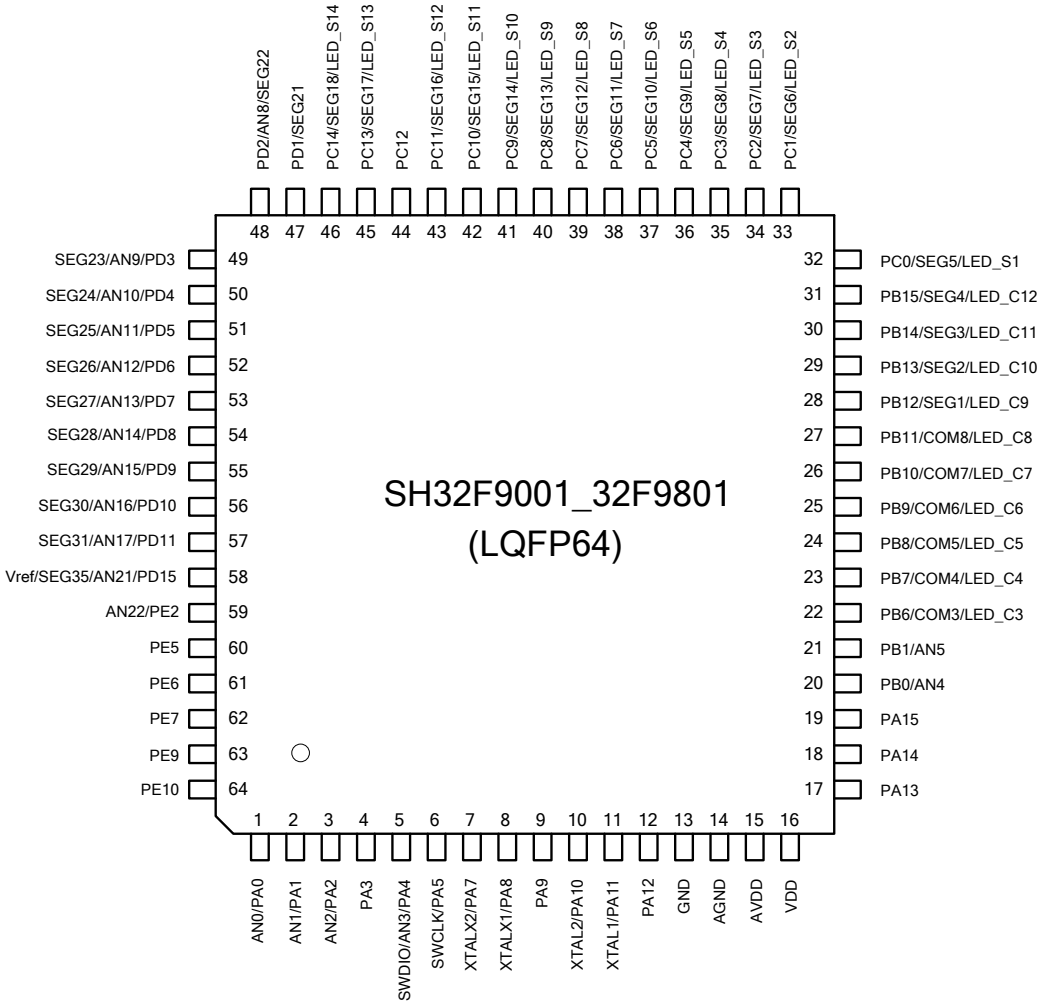
4. 引脚定义

80脚LQFP封装



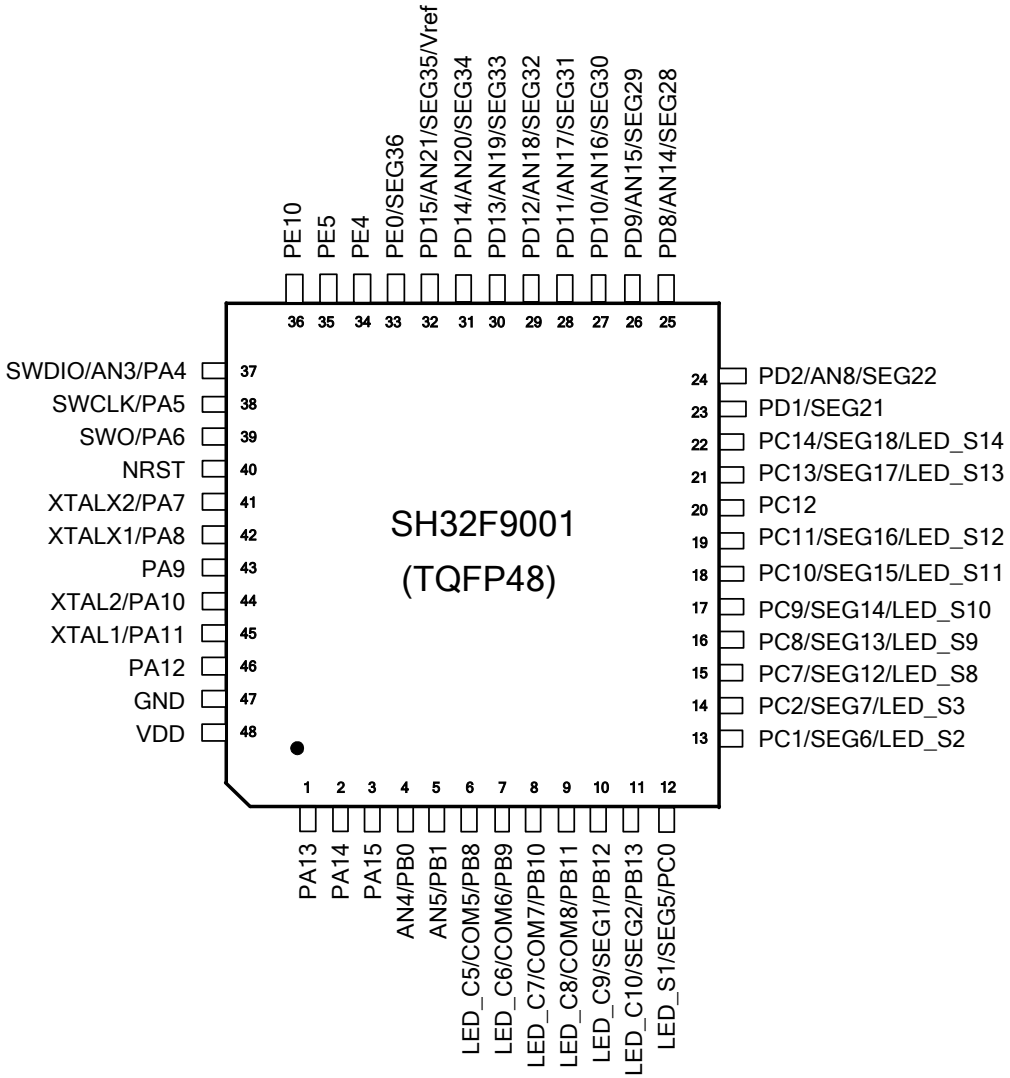
说明：上图中每个GPIO口只列出了典型复用选项，完整的复用（含重映像）选项请参考“复用功能映射表”。

64脚LQFP封装



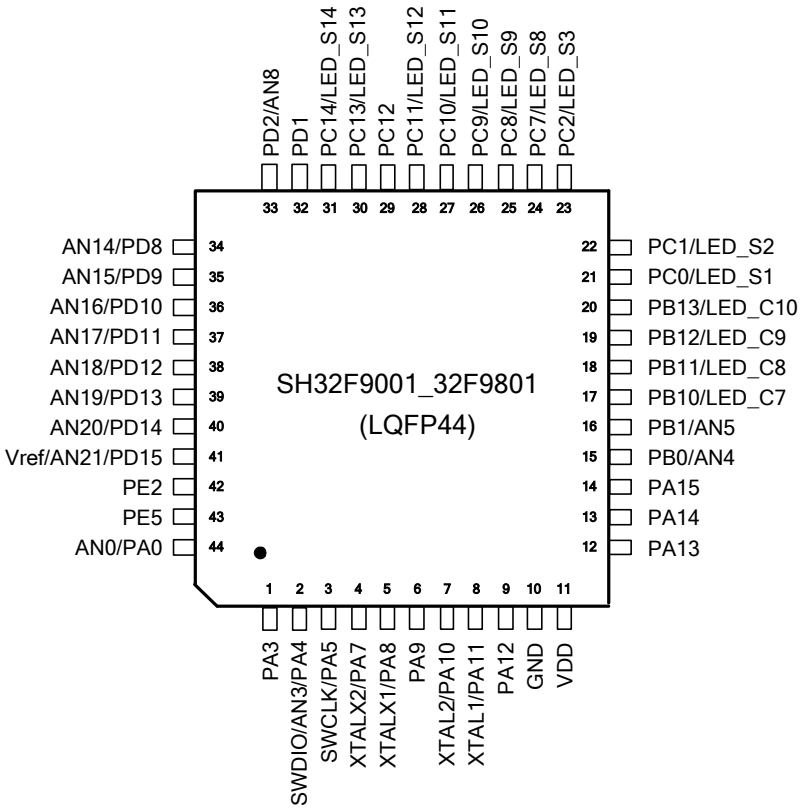
说明：上图中每个GPIO口只列出了典型复用选项，完整的复用（含重映像）选项请参考“复用功能映射表”。

48脚TQFP封装



说明：上图中每个GPIO口只列出了典型复用选项，完整的复用（含重映像）选项请参考“复用功能映射表”。

44脚LQFP封装



注意：

除个别电源与系统引脚外，所有引脚都有复用功能，复用功能通过"端口复用功能选择器（AFRL/AFRH）"选择设定。
除SWJ端口引脚外，所有复用引脚上电默认是GPIO口。
系统层面的SWJ端口和XTAL端口复用在SWJCFG@SYSCFG_SAFR和OSCCFG@SYSCFG_SAFR中定义。

复用功能映射表（Alternate Function Mapping）

AF1	AF2	AF3	AF4	AF5	AF6	AF7	AF8	AF9
GPIO	UART0/1/2 UART3/4/5	SPI0 TWI0	SPI1 TWI1	PCA0/1/2/3 PCA4/5/6/7	TIM5/6/7/8	PWM0/ 1/2/3	LED ADC	LCD
PA0	TXD3		SS1		T5		AN0	
PA1	RXD3		MISO1		T6		AN1	
PA2		SDA0	MOSI1		T7	FLT0	AN2	
PA3	TXD0	SCL0	SCK1		T8	PWM0A		
PA4	RXD0			P0CEX0		PWM0B	AN3	
PA5	TXD1			P0CEX1				
PA6	RXD1			ECI0				
PA7	TXD2			P2CEX0				
PA8	RXD2			P2CEX1				
PA9	TXD4			ECI2				
PA10	RXD4			P6CEX0				
PA11	TXD5			P6CEX1				
PA12	RXD5			ECI6	T5			
PA13		SCL0		ECI5	T6	FLT1		
PA14		SDA0		P5CEX0	T7	PWM1A		
PA15			SCL1	P5CEX1	T8	PWM1B		
PB0	TXD3		SDA1	ECI1		FLT2	AN4	
PB1	RXD3			P1CEX0		PWM2A	AN5	
PB2	TXD4			P1CEX1		PWM2B	AN6	
PB3	RXD4			P2CEX0		FLT3	AN7	
PB4	TXD5			P2CEX1		PWM3A	LED_C1	COM1
PB5	RXD5			ECI2		PWM3B	LED_C2	COM2
PB6	TXD0			P4CEX0			LED_C3	COM3
PB7	RXD0			P4CEX1		FLT0	LED_C4	COM4
PB8	TXD1			ECI4		PWM0A	LED_C5	COM5
PB9	RXD1					PWM0B	LED_C6	COM6
PB10	TXD2		SS1		T5		LED_C7	COM7
PB11	RXD2		MISO1	P7CEX0	T6		LED_C8	COM8
PB12		SDA0	MOSI1	P7CEX1	T7		LED_C9	SEG1
PB13		SCL0	SCK1	ECI7	T8		LED_C10	SEG2
PB14				P3CEX0			LED_C11	SEG3
PB15		SS0		P3CEX1		FLT1	LED_C12	SEG4
PC0	TXD3	SCK0	SCL1	ECI3		PWM1A	LED_S1	SEG5
PC1	RXD3	MOSI0	SDA1			PWM1B	LED_S2	SEG6
PC2	TXD4	MISO0				FLT2	LED_S3	SEG7
PC3	RXD4		MISO1			PWM2A	LED_S4	SEG8
PC4	TXD5	SDA0	MOSI1			PWM2B	LED_S5	SEG9
PC5	RXD5	SCL0	SCK1			FLT3	LED_S6	SEG10
PC6		SCK0	SS1	ECI0	T5	PWM3A	LED_S7	SEG11
PC7	TXD1	MISO0		P0CEX0	T6	PWM3B	LED_S8	SEG12
PC8	RXD1	MOSI0		P0CEX1	T7		LED_S9	SEG13
PC9		SS0	SCK1	P1CEX0	T8		LED_S10	SEG14
PC10	RXD0	SCL0	MOSI1	P1CEX1			LED_S11	SEG15
PC11	TXD0	SDA0	MISO1	ECI1			LED_S12	SEG16
PC12	TXD1	MISO0	SS1					
PC13	RXD1	MOSI0	SDA1	P6CEX0			LED_S13	SEG17
PC14	TXD2	SCK0	SCL1	P6CEX1			LED_S14	SEG18
PC15	RXD2	SS0		ECI6			LED_S15	SEG19
PD0	TXD3			P2CEX0		FLT0	LED_S16	SEG20
PD1	RXD3			P2CEX1		PWM0A		SEG21
PD2	TXD4			ECI2	T5	PWM0B	AN8	SEG22
PD3	RXD4			P3CEX0	T6	FLT1	AN9	SEG23
PD4	TXD5			P3CEX1	T7	PWM1A	AN10	SEG24
PD5	RXD5			ECI3	T8	PWM1B	AN11	SEG25
PD6				P7CEX0			AN12	SEG26
PD7				P7CEX1			AN13	SEG27
PD8				ECI7			AN14	SEG28
PD9				P3CEX0			AN15	SEG29
PD10				P3CEX1			AN16	SEG30
PD11				ECI3			AN17	SEG31
PD12							AN18	SEG32
PD13	RXD0			P4CEX0			AN19	SEG33
PD14	TXD0			P4CEX1			AN20	SEG34
PD15	TXD1	SS0		ECI4	T5		AN21	SEG35
PE0	RXD1	SCK0	SCL1	ECI5	T6			SEG36
PE1	TXD2	MOSI0	SDA1	P5CEX0	T7			
PE2	RXD2	MISO0		P5CEX1	T8		AN22	
PE3							AN23	
PE4						FLT0		
PE5	TXD2		MISO1			FLT1		
PE6	RXD2		MOSI1			PWM0B		
PE7			SCK1			PWM1B		
PE8			SS1			PWM0A		
PE9						PWM1A		
PE10								

注：AF0为引脚非选功能(I/O为高阻,仅上/下拉有效),AF1为GPIO，AF2～AF15为数字/模拟信号。原则上不建议将一个功能口配置到不同I/O引脚上。

5. 引脚描述

引脚编号	类型	说明
I/O端口		
PA0 – PA15	I/O	16位双向I/O端口
PB0 – PB15	I/O	16位双向I/O端口
PC0 – PC15	I/O	16位双向I/O端口
PD0 – PD15	I/O	16位双向I/O端口
PE0 – PE10	I/O	11位双向I/O端口
通用PWM定时器（PWM0~PWM3，支持双通道带死区互补输出）		
PWM0A – PWM3A	O	16位PWM0~PWM3通道A脉冲输出
PWM0B – PWM3B	O	16位PWM0~PWM3通道B脉冲输出
FLT0 – 3	I	PWM0~PWM3故障检测输入引脚
PCA控制器（PCA0~PCA3，16位可编程计数器阵列）		
P0CEX0 – P7CEX0	I/O	PCA0~PCA7模块0输入/出引脚
P0CEX1 – P7CEX1	I/O	PCA0~PCA7模块1输入/出引脚
ECI0 – ECI7	I	PCA0~PCA7外部时钟输入
基本定时器（TIM5~TIM8）		
T5 – T8	I/O	TIM5~TIM8外部时钟输入/时钟比较输出
ADC（24通道模数转换器）		
AN0 – 23	I	ADC输入通道
VREF	I/O	ADC内部参考电压滤波电容接口/内部参考电压输出/外部参考电压输入
TPS	I	温度传感器输入通道
中断，复位，时钟，电源		
NRST	I	该引脚上保持10μs以上的低电平，CPU将复位。由于有内建30kΩ上拉电阻连接到V_{DD}，所以仅接一个外部电容即可实现上电复位
BOOT*	I	启动模式选择引脚，BOOT=0为正常从闪存启动，BOOT=1为从系统存储器启动（ISP）
XTAL1	I	高频谐振器输入
XTAL2	O	高频谐振器输出
XTALX1	I	32kHz晶体谐振器输入
XTALX2	O	32kHz晶体谐振器输出
GND	P	数字接地
V_{DD}	P	数字电源（2.0V - 5.5V）
AGND	P	模拟接地
AV_{DD}	P	模拟电源（2.0V - 5.5V）
UART（UART0~UART5，通用同步异步串行口）		
RXD0 – 5	I/O	串口0~5数据接收
TXD0 – 5	O	串口0~5数据发送

SPI（SPI0/1，串行外设接口）		
MOSI0/1	I/O	SPI0/1主输出从输入引脚
MISO0/1	I/O	SPI0/1主输入从输出引脚
SCK0/1	I/O	SPI0/1串行时钟引脚
SS0/1	I	SPI0/1从设备选择引脚
TWI（TWI0/1，两线串行接口）		
SCL0/1	I/O	TWI0/1时钟信号（支持主机/从机模式）
SDA0/1	I/O	TWI0/1数据输入输出
LCD		
COM1~COM8	O	LCD输出COM口
SEG1~SEG36	O	LCD输出Segment口
LED		
LED_C1~LED_C12	O	LED输出COM口
LED_S1~LED_S16	O	LED输出Segment口
调试接口（SWDP，串行双线调试接）		
SWDIO	I/O	SWDP数据输入输出
SWCLK	I	SWDP时钟输入
SWO	I	SWDP跟踪数据输出

注意：

系统上电这3个接口默认是调试接口。

*注：BOOT引脚实际就是一个普通GPIO口，其功能在ISP程序中定义。

6. 产品信息

Part Num	RAM (KBytes) (Sram + C-RAM)	Flash (KBytes)	E2 (KBytes)	ISP (KBytes)	PWM (units/chls)	ADC (chls)	UART/SPI /TWI (units)	PCA (units/chls)	IO	Package
SH32F9001U/048UR	16 + 16	256	6	6	4/2	12	6/2/2	8/2	45	TQFP48
SH32F9001B/064BR	16 + 16	256	6	6	4/2	18	6/2/2	8/2	60	LQFP64(12X12)
SH32F9001P/044PR	16 + 16	256	6	6	4/2	13	6/2/2	8/2	42	LQFP44
SH32F9001P/064PR	16 + 16	256	6	6	4/2	18	6/2/2	8/2	60	LQFP64(10X10)
SH32F9001P/080PR	16 + 16	256	6	6	4/2	24	6/2/2	8/2	75	LQFP80(12X12)
SH32F9001Y/080YR	16 + 16	256	6	6	4/2	24	6/2/2	8/2	75	LQFP80(14X14)
SH32F9801P/044PR	16 + 16	128	6	6	4/2	13	6/2/2	8/2	42	LQFP44
SH32F9801B/064BR	16 + 16	128	6	6	4/2	18	6/2/2	8/2	60	LQFP64(12X12)
SH32F9801P/080PR	16 + 16	128	6	6	4/2	24	6/2/2	8/2	75	LQFP80(12X12)
SH32F9801Y/080YR	16 + 16	128	6	6	4/2	24	6/2/2	8/2	75	LQFP80(14X14)

SH32 F 9 0 0 1 P /080PR

产品序列
SH30 = ARM Cortex M0 32bit 系列
SH32 = ARM Cortex M3 32bit 系列
SH33 = ARM Cortex M4 32bit 系列
SH34 = ARM Cortex M7 32bit 系列

产品类型
F = Flash类型，无特殊功耗需求
L = 低功耗类型

产品子序列
2 = 高性能型/电机
7 = 低功耗/电表
9 = 多功能型/家电

ROM容量
1 = 8KB
2 = 16KB
3 = 24KB
4 = 32KB
5 = 48KB
6 = 64KB
7 = 96KB
8 = 128KB
9 = 192KB
0 = 256KB
A = 384KB
B = 512KB
C = 768KB
D = 1024KB
E = 1536KB
F = 2048KB

产品序号
芯片产品序列号

产品版本
第一版为空，第二版从A~Z开始递增

封装类型
D = DIP
F = QFP
P/B/Y = LQFP
Q = QFN
U = TQFP
X = TSSOP

生产代码
080： 引脚数
P ：封装类型，同上
R ：包装类型
R：Tray
U：Tube
A：Tape&Reel
E：Tape&Reel-Type-E
W：Tape&Reel-Type-W

7. 内核

7.1 ARM®的 CortexTM-M3 核心

ARM® Cortex™-M3 处理器是行业领先的 32 位RISC处理器，适用于具有较高确定性的实时应用，它经过专门开发，适用于广泛的高性能低成本平台应用（包括微控制器、工业控制系统以及无线网络和传感器）。Cortex™-M3处理器具有出色的计算性能以及对事件的优异系统响应能力，该处理器提供额外的代码效率，同时可应对实际中对低动态和静态功率需求的挑战。Cortex™-M3采用了哈佛结构，拥有独立的指令总线和数据总线，可以让取指与数据访问并行不悖，从而提升了性能。

SH32F9001_32F9801拥有ARM®Cortex™-M3最新的r2p1版本内核。

有关内核的更多信息请参考《Cortex™-M3技术参考手册》。

7.2 寄存器组

Cortex-M3 处理器拥有R0-R15的寄存器组。其中R13作为堆栈指针SP。SP有两个，但在同一时刻只能有一个可以看到（Banked寄存器）。

- R0-R12：通用寄存器

R0-R12都是32位通用寄存器，用于数据操作。

但是注意：绝大多数16位Thumb指令只能访问R0-R7，而32位Thumb-2指令可以访问所有寄存器。

- Banked R13：两个堆栈指针

Cortex-M3拥有两个堆栈指针，然而它们是banked，因此任一时刻只能使用其中的一个。

□ 主堆栈指针（MSP）：复位后缺省使用的堆栈指针，用于操作系统内核以及异常处理例程（包括中断服务例程）

□ 进程堆栈指针（PSP）：由用户的应用程序代码使用。

堆栈指针的最低两位永远是0，这意味着堆栈总是4字节对齐的。

在ARM 编程领域中，凡是打断程序顺序执行的事件，都被称为异常(Exception)。除了外部中断外，当有指令执行了“非法操作”，或者访问被禁的内存区间，因各种错误产生的Fault，以及不可屏蔽中断发生时，都会打断程序的执行，这些情况统称为异常。在不严格的上下文中，异常与中断也可以混用。另外，程序代码也可以主动请求进入异常状态的（常用于系统调用）。

- R14：连接寄存器

当呼叫一个子程序时，由R14存储返回地址。

不像大多数其它处理器，ARM为了减少访问内存的次数，把返回地址直接存储在寄存器中。这样足以使很多只有1级子程序调用的代码无需访问内存（堆栈内存），从而提高了子程序调用的效率。如果多于1级，则需要把前一级的R14值压到堆栈里。在ARM上编程时，应尽量只使用寄存器保存中间结果，迫不得已时才访问内存。

- R15：程序计数寄存器

指向当前的程序地址。如果修改它的值，就能改变程序的执行流。

- 特殊功能寄存器

Cortex-M3 还在内核水平上搭载了若干特殊功能寄存器，包括

□ **程序状态字寄存器组**（PSRs）

□ **中断屏蔽寄存器组**（PRIMASK, FAULTMASK, BASEPRI）

□ **控制寄存器**（CONTROL）

表7-1 Cortex-M3 特殊功能寄存器及其功能

寄存器	功能
xPSR	记录ALU标志（0 标志，进位标志，负数标志，溢出标志），执行状态，当前正服务的中断号
PRIMASK	除能所有的中断——除了不可屏蔽中断（NMI）
FAULTMASK	除能所有的Fault——除了NMI，被除能的Faults可以越级
BASEPRI	除能所有优先级不高于某个具体数值的中断。
CONTROL	定义特权状态，并且决定使用哪一个堆栈指针

8. 存储器和总线架构

ARM® Cortex®-M3的总线接口基于AHB-Lite和APB协议，它们的规则参考“AMBA2.0 规格（第4版）”。

8.1 系统架构

SH32F9001_32F9801主系统由多层AHB总线矩阵互联和驱动，有4个主机接口和6个从机接口，如下图所示：

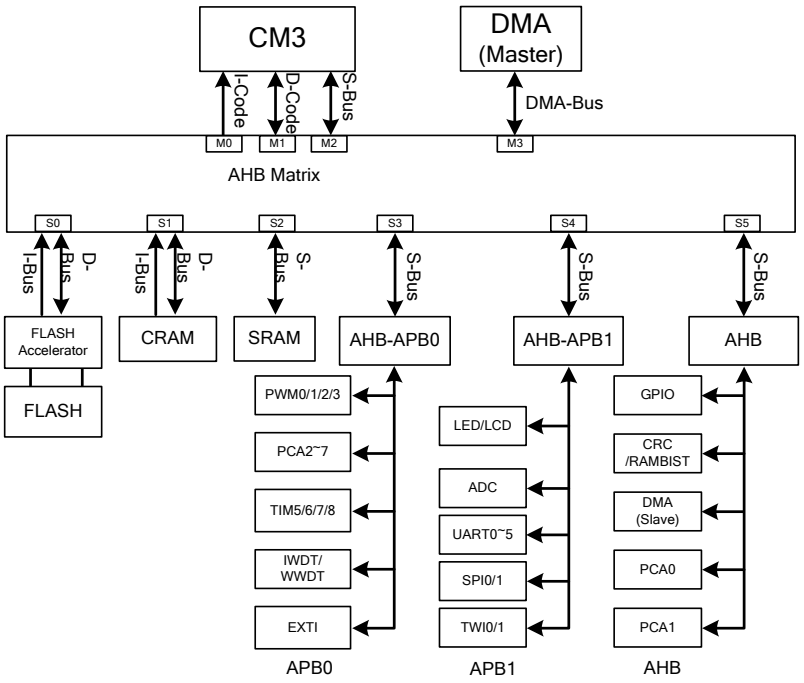


图8-1　　SH32F9001_32F9801 系统架构框图

ICode总线

该总线将Cortex™-M3内核的ICode总线与Flash/CRAM指令接口相连接。指令预取在此总线上完成。

DCode总线

该总线将Cortex™-M3内核的DCode总线与Flash/CRAM的数据接口相连接(常量加载和调试访问)。

System总线

该总线连接Cortex™-M3内核的系统总线(外设总线)到总线矩阵，总线矩阵协调着内核和DMA间的访问。

DMA总线

此总线将DMA的AHB主控接口与总线矩阵相联，总线矩阵协调着CPU和DMA到SRAM、CRAM、Flash和外设的访问。

注：此图表示DMA访问外设走总线矩阵，也可根据设计要求走独立总线。

总线矩阵

总线矩阵协调内核系统总线和DMA主控总线之间的访问仲裁。总线矩阵包含4个主机接口(CPU的ICode、DCode、系统总线和DMA总线)和6个从机接口(Flash存储器接口2个、指令SRAM接口2个、数据SRAM接口1个和AHB外设接口1个)。AHB外设通过总线矩阵与系统总线相连，允许DMA访问。

注：DMA控制器中无论是内存还是外设都走DMA总线。

AHB外设总线

AHB外设总线挂载在总线矩阵上，一些要求高速并和系统紧密耦合的外设挂载在该总线上。AHB/APB桥也挂载在此总线上。

AHB/APB桥

AHB-APB桥在AHB和APB总线间提供同步连接。APB0和APB1都是降速外设总线，两者最高速度均为36MHz。

在使用一个外设之前，必须设置时钟寄存器RCC_AHBENR/RCC_APB0ENR/RCC_APB1ENR的相应控制位来打开该外设的时钟，具体请参考“复位和时钟控制（RCC）”章节。

8.2 存储器组织

SH32F9001_32F9801遵循ARM® Cortex™-M3对存储器组织的规定，存储器基本组织架构如下：

表8-1 SH32F9001_32F9801 基本存储器组织表

地址范围	用途	地址范围细分	描述
0x0000 0000 – 0x0FFF FFFF	片上非易失性代码存储区（Flash）*	0x0000 0000 – 0x0003 FFFF	片上256K flash设备（SH32F9001）
		0x0000 0000 – 0x0001 FFFF	片上128K flash设备（SH32F9801）
	片上非易失性系统存储区（System Block）	0x0FFF 0000 – 0x0FFF 17FF	片上EEProm-Like存储区（共6K）
		0x0FFF 8000 – 0x0FFF 97FF	片上Boot Rom区（共6K，包含ISP）
		0x0FFF E000 – 0x0FFF EFFF	片上Information区（共4K，包含1024字节OTP）
0x1000 0000 – 0x1FFF FFFF	片上易失性数据存储区2（CRAM，可作为代码存储区）	0x1000 0000 – 0x1000 3FFF	片上16K CRAM设备
0x2000 0000 – 0x3FFF FFFF	片上易失性数据存储区1（SRAM）	0x2000 0000 – 0x2000 3FFF	片上16K SRAM设备（位于数据位带区）
		0x2200 0000 – 0x25FF FFFF	数据位带别名区，每一个地址映像为位带区的一个位
0x4000 0000 – 0x5FFF FFFF	片上外设区	0x4000 0000 – 0x4001 FFFF	片上APB0外设区，每个外设分配1K字节地址空间（位于外设位带区）
		0x4002 0000 – 0x4003 FFFF	片上APB1外设区，每个外设分配1K字节地址空间（位于外设位带区）
		0x4004 0000 – 0x400F FFFF	片上AHB外设区，每个外设分配1K字节地址空间（位于外设位带区）
		0x4200 0000 – 0x43FF FFFF	外设位带别名区，每一个地址映像为位带区的一个位
0x6000 0000 – 0x9FFF FFFF	保留区域	-	可在此区域扩展外部RAM设备
0xA000 0000 – 0xDFFF FFFF	保留区域	-	可在此区域扩展外部外设设备
0xE000 0000 – 0xE00F FFFF	Cortex-M3私有外设区	0xE000 0000 – 0xE000 0FFF	指令跟踪宏单元（ITM）
		0xE000 1000 – 0xE000 1FFF	数据观察点与跟踪（DWT）
		0xE000 2000 – 0xE000 2FFF	闪存地址重载与断点单元（FPB）
		0xE000 3000 – 0xE000 DFFF	保留区域
		0xE000 E000 – 0xE000 EFFF	嵌套向量中断控制器（NVIC）
		0xE000 F000 – 0xE003 FFFF	保留区域
		0xE004 0000 – 0xE004 0FFF	跟踪端口接口单元（TPIU）

8.2.1 寄存器映像

表8-2列出了所有SH32F9001_32F9801中内置外设（不包含Cortex-M3内核设备）的基地址。具体各个外设的寄存器映像请参考该外设“寄存器”章节中的寄存器地址映像表。

外设寄存器地址：外设基地址+寄存器偏移地址

表8-2 SH32F9001_32F9801 内置外设的基地址

基地址	外设	总线
0x4000 0000 ~ 0x4000 03FF	reserved	APB0总线: 0x4000 0000 - 0x4000 83FF
0x4000 0400 ~ 0x4000 07FF	TIM5	
0x4000 0800 ~ 0x4000 0BFF	TIM6	
0x4000 0C00 ~ 0x4000 0FFF	TIM7	
0x4000 1000 ~ 0x4000 13FF	TIM8	
0x4000 1400 ~ 0x4000 3BFF	reserved	
0x4000 3C00 ~ 0x4000 3FFF	PWM0	
0x4000 4000 ~ 0x4000 43FF	PWM1	
0x4000 4400 ~ 0x4000 47FF	PWM2	
0x4000 4800 ~ 0x4000 4BFF	PWM3	
0x4000 4C00 ~ 0x4000 5FFF	reserved	
0x4000 6000 ~ 0x4000 63FF	PCA2	
0x4000 6400 ~ 0x4000 67FF	PCA3	
0x4000 6800 ~ 0x4000 6BFF	PCA4	
0x4000 6C00 ~ 0x4000 6FFF	PCA5	
0x4000 7000 ~ 0x4000 73FF	PCA6	
0x4000 7400~ 0x4000 77FF	PCA7	
0x4000 7800 ~ 0x4000 7BFF	EXTI	
0x4000 7C00 ~ 0x4000 7FFF	IWDT	
0x4000 8000 ~ 0x4000 83FF	WWDT	
0x4002 0000 ~ 0x4002 03FF	UART0	APB1总线: 0x4002 0000 - 0x4002 3FFF
0x4002 0400 ~ 0x4002 07FF	UART1	
0x4002 0800 ~ 0x4002 0BFF	UART2	
0x4002 0C00 ~ 0x4002 0FFF	UART3	
0x4002 1000 ~ 0x4002 13FF	UART4	
0x4002 1400 ~ 0x4002 17FF	UART5	
0x4002 1800 ~ 0x4002 1BFF	SPI0	
0x4002 1C00 ~ 0x4002 1FFF	SPI1	
0x4002 2000 ~ 0x4002 23FF	TWI0	
0x4002 2400 ~ 0x4002 27FF	TWI1	
0x4002 2800 ~ 0x4002 2BFF	ADC	
0x4002 2C00 ~ 0x4002 2FFF	TPS	
0x4002 3000 ~ 0x4002 33FF	reserved	
0x4002 3400 ~ 0x4002 37FF	LED	
0x4002 3800 ~ 0x4002 3BFF	LCD	
0x4004 0000 ~ 0x4004 03FF	PCA0	AHB总线: 0x4004 0000 - 0x4004 5BFF
0x4004 0400 ~ 0x4004 07FF	PCA1	
0x4004 0800 ~ 0x4004 081F	GPIOA_CFG	
0x4004 0820 ~ 0x4004 083F	GPIOB_CFG	
0x4004 0840 ~ 0x4004 085F	GPIOC_CFG	
0x4004 0860 ~ 0x4004 087F	GPIOD_CFG	
0x4004 0880 ~ 0x4004 089F	GPIOE_CFG	
0x4004 0A00 ~ 0x4004 0A1F	GPIOA	
0x4004 0A20 ~ 0x4004 0A3F	GPIOB	
0x4004 0A40 ~ 0x4004 0A5F	GPIOC	
0x4004 0A60 ~ 0x4004 0A7F	GPIOD	
0x4004 0A80 ~ 0x4004 0A9F	GPIOE	
0x4004 0C00 ~ 0x4004 37FF	reserved	
0x4004 3800 ~ 0x4004 3BFF	SYSCFG	
0x4004 3C00 ~ 0x4004 3FFF	RCC	
0x4004 4000 ~ 0x4004 43FF	DMA	
0x4004 4400 ~ 0x4004 47FF	reserved	
0x4004 4800 ~ 0x4004 4BFF	FLASH	
0x4004 4C00 ~ 0x4004 4FFF	reserved	
0x4004 5000 ~ 0x4004 53FF	CRC	
0x4004 5400 ~ 0x4004 57FF	RAMBIST	
0x4004 5800 ~ 0x4004 5BFF	reserved	

8.2.2 数据存储器 1（SRAM）

SH32F9001_32F9801内置16K数据RAM（SRAM）。它可以以字节、半字(16位)或全字(32位)访问。SRAM的起始地址是0x2000 0000。

SRAM支持March检测算法。

8.2.3 位带

Cortex™-M3存储器映像包括两个位带(bit-band)区。这两个位带区的每一位（1位）将映射到位带别名区中的一个字(32位)，在位带别名区写入一个字具有对位带区的目标位执行读-改-写操作的相同效果。

位带区：支持位带操作的地址区

位带别名区：对别名地址的访问最终作用到位带区的访问上（这中间有一个地址映像过程）

在SH32F9001_32F9801里，SRAM区和片上外设区全部都分配在位带区，即都支持位带操作。

■ 对SRAM 位带区的某个比特，记它所在字节地址为A，位序号为n(0<=n<=7)，则该比特在别名区的地址为：

位带别名地址 = 0x22000000 + ((A-0x20000000)*8+n)*4 = 0x22000000+(A-0x20000000)*32 + n*4

■ 对于片上外设位带区的某个比特，记它所在字节的地址为A，位序号为n(0<=n<=7)，则该比特在别名区的地址为：

位带别名地址 = 0x42000000 + ((A-0x40000000)*8+n)*4 = 0x42000000+(A-0x40000000)*32 + n*4

下面的例子说明如何映像SRAM区地址为0x20000300字节的位2：

位带别名地址= 0x22000000 + (0x300×32) + (2×4) = 0x22006008

对0x22006008地址的写操作与对SRAM中地址0x20000300字节的位2执行读-改-写操作有着相同的效果。读0x22006008地址返回SRAM中地址0x20000300字节的位2的值(0x01或0x00)。

注意：CortexTM-M3中存储空间是按字节编址的，因此位带别名区的一个字（32位）占用4个字节地址，即根据位带映射得到的位带别名地址，其最低2位必然是0。

因SH32F9001_32F9801采用的是小端模式，对位带别名地址=X的位置进行字操作，实际上相当于对地址X、X+1、X+2、X+3地址进行字节操作。因此上述例子中更严谨的说法是对0x22006008地址写入0x00000000或0x00000001，或者读出0x00000000或0x00000001。

有关位带的更多信息请参考《Cortex™-M3技术参考手册》。

8.2.4 程序存储器

SH32F9001_32F9801内置有Flash存储器，分成主存储块和特殊信息块两部分：

- 主存储块为256K/128K字节，划分为256/128个1K字节的扇区
- 特殊信息块为16K字节，划分为16个1K字节的扇区

有关Flash程序存储器的详细信息，请参考下一节“Flash程序存储器”。

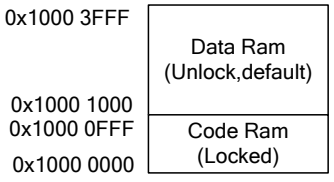
8.2.5 数据存储器 2（CRAM）

SH32F9001_32F9801另内置了16K程序RAM（CRAM），该CRAM可作为Data Ram使用，也可作为Code Ram使用，用于加载核心程序，支持程序全速运行。加载操作由用户软件完成（一般由IDE环境自动生成加载代码并放置STARTUP中）。

CRAM提供一个锁定方式用于保护程序代码，当设置为锁定方式时，CRAM变为只读。锁定按sector=2KB进行设置，通过CRAMLCKn@SYSCFG_CRAMLOCK控制位设置。

具体的CRAM代码搬移和运行方法可参考应用参考文档“SH32F9001_32F9801使用CRAM运行程序”。

如下例中通过设置4KB为Code Ram，其它保持为Data Ram，设置如下：



8.3 Flash 程序存储器

8.3.1 特性

- Flash 主存储器包括 256/128 X 1KB 扇区，总共 256/128KB
- Flash 特殊信息存储器包括 16 X 1KB 扇区，总共 16KB
- Flash 读操作位宽为 128 位，写操作位宽支持 32 位，16 位
- 在工作电压范围内都能进行编程和擦除操作
- 支持整体/扇区擦除和编程
- 支持 FLASH 划分区块功能，方便在线升级程序及备份
- Flash 读保护功能，防止非法访问
- Flash 写保护功能，阻止意外操作
- 编程/擦除次数：　程序区：至少 100,000 次
  类 EEPROM 区：至少 100,000 次
- 数据保存年限：至少 20 年
- 低功耗

8.3.2 简介

SH32F9001_32F9801 的 Flash 存储区包括两个部分，分别是 Flash 主程序存储区（Main Program Memory Block）和特殊信息存储区（Information Memory Block）。

SH32F9001_32F9801 内置 256/128K 的可编程 Flash 主程序存储区（Main Program Memory Block），每个扇区 1024 字节，256/128 个扇区都可单独进行擦除。

SH32F9001_32F9801 的特殊信息存储区包括 6KB 的内置类 EEPROM 存储区，用于存放用户数据，每个扇区 1024 字节。此外特殊信息存储区还包括 6KB 的引导区（Boot Rom Block），每个扇区 1024 字节。Flash 存储器结构见下图。

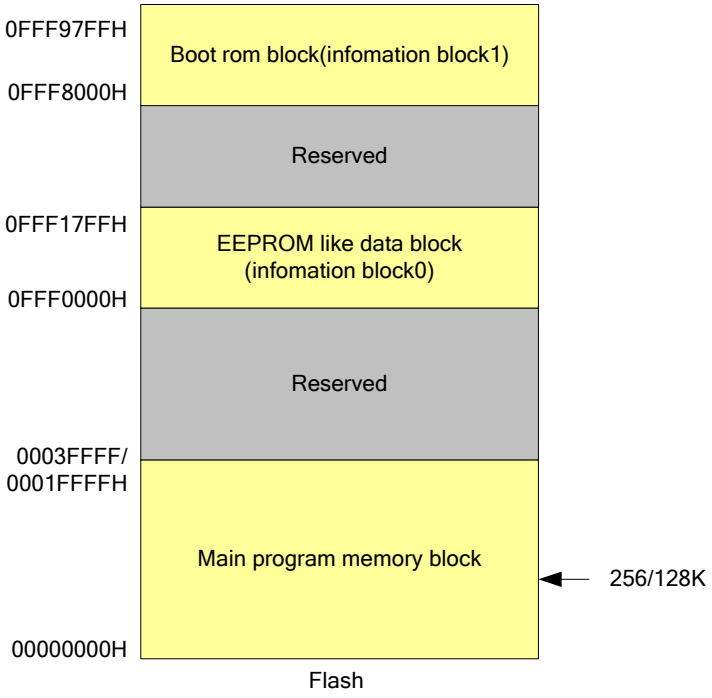


图8-2　　Flash 存储器结构

表8-3　　Flash 存储器结构表

Block	Name	Base address	Size（bytes）
Main memory (SH32F9001)	**Sector 0**	0x0000 0000 - 0x0000 03FF	1024
	Sector 1	0x0000 0400 - 0x0000 07FF	1024
	Sector 2	0x0000 0800 - 0x0000 0BFF	1024
	…	…	…
	…	…	…
	Sector 254	0x0003 F800 - 0x0003 FBFF	1024
	Sector 255	0x0003 FC00 - 0x0003 FFFF	1024
Main memory (SH32F9801)	**Sector 0**	0x0000 0000 - 0x0000 03FF	1024
	Sector 1	0x0000 0400 - 0x0000 07FF	1024
	Sector 2	0x0000 0800 - 0x0000 0BFF	1024
	…	…	…
	…	…	…
	Sector 126	0x0001 F800 - 0x0001 FBFF	1024
	Sector 127	0x0001 FC00 - 0x0001 FFFF	1024
Information Block	**EEPROM Block**	0x0FFF 0000 - 0x0FFF 17FF	6144
	Boot Rom	0x0FFF 8000 - 0x0FFF 97FF	6144
	Protect Block	0x0FFF E000 - 0x0FFF E7FF	1024
	Customer Block	0x0FFF E800 - 0x0FFF EFFF	1024
	Product Block	0x0FFF F000 - 0x0FFF F7FF	1024
	OTP Block	0x0FFF F800 - 0x0FFF FFFF	1024
Flash memory Operation registers	**FLASH_ACR**	0x4004 4800 - 0x4004 4803	4
	FLASH_MKYR	0x4004 4804 - 0x4004 4807	4
	FLASH_E2KYR	0x4004 4808 - 0x4004 480B	4
	FLASH_SR	0x4004 480C - 0x4004 480F	4
	FLASH_CR	0x4004 4810 - 0x4004 4813	4
	FLASH_OPR	0x4004 4814 - 0x4004 4817	4
	FLASH_RPR	0x4004 4818 - 0x4004 481B	4
	FLASH_WRPR	0x4004 481C- 0x4004 481F	4
	FLASH_CNTR	0x4004 4820 - 0x4004 4823	4
	FLASH_UPCNTR	0x4004 4824 - 0x4004 4827	4
	FLASH_CNTCR	0x4004 4828 - 0x4004 482B	4
	FLASH_IKYR	0x4004 482C - 0x4004 482F	4
	Reversed	0x4004 4830- 0x4004 48FF	208
	FLASH_MEMRMP	0x4004 4900- 0x4004 4913	4

8.3.3　Flash 存储器功能描述

8.3.3.1　Flash 擦除与编程

系统复位后，Flash 存储器操作寄存器处于锁定状态，所以对 Flash 进行擦除编程操作之前需要对 Flash 存储器操作寄存器进行解锁。不同 Flash 存储区具有不同的解锁寄存器，对不同区域操作需要解锁相应的寄存器。对解锁 Flash 操作也分为两种情况，一种是解锁后可对 Flash 进行单次操作，另一种是解锁后，可对 Flash 进行多次操作。如果 Flash 存储器操作寄存器处于解锁状态，可通过控制 FLASH_CR 寄存器，对 Flash 各存储区进行擦除、编程操作。但为防止 Flash 被误操作，对 Flash 存储器控制寄存器进行访问时，必须保证 Flash 操作定时器的计数值在一个有效的范围内，否则对 Flash 控制寄存器 FLASH_CR 的修改被禁止。

对 Flash 存储器，每次可以写入 16 位或 32 位。主存储区进行擦除操作后，每 16 位或 32 位仅可以被编程一次，即被编程的 16 位或 32 位数据的各个位如果不是 0，就不可以再次被烧写，否则相应错误标志置 1。

主存储区的整片擦除与按扇区擦除，主存储区的编程，类 EEPROM 存储区的按扇区擦除与编程，客户信息区的按扇区擦除与编程都可以通过运行在 RAM 中的程序，Boot 区的程序或者在 Debug 模式下直接访问 Flash 操作寄存器来实现。

主存储区的按扇区擦除及主存储区的编程、类 EEPROM 存储区的按扇区擦除与编程，客户信息区的按扇区擦除与编程可以通过主程序区中的程序访问 Flash 控制寄存器来实现。

产品信息区，Boot 区的扇区擦除与编程，在 Debug 模式下，只能通过直接访问 Flash 操作寄存器来实现。

Flash 中的主程序区可以通过设置读保护防止被非法的读出；同样可以对 Flash 存储区的各扇区设置写保护，防止在程序跑飞的情况下不被意外地改变。对于 256KB 的 Flash 存储区，写保护的基本单位是 8 个扇区为一个保护单元。

8.3.3.2 Flash 控制寄存器解锁

系统复位后，Flash 存储器操作寄存器进入锁定状态，所以对 Flash 进行擦除编程操作之前需要对 Flash 存储器操作寄存器进行解锁。不同 Flash 存储区具有不同的解锁寄存器，对不同区域操作需要解锁相应的寄存器。对 Flash 解锁需要对相应解锁寄存器写入两个解锁值，第一个键值写入错误或第二个键值写入错误，都不能对 Flash 相应区域进行解锁，并会产生一个总线错误；任意一个区域处于解锁状态，再对解锁寄存器进行解锁，也会产生一个总线错误。对解锁 Flash 操作也分为两种情况，一种是解锁后可对 Flash 进行单次操作，另一种是解锁后，可对 Flash 进行多次操作。如果 Flash 存储器操作寄存器处于解锁状态，可通过控制 FLASH_CR，对 Flash 各存储区进行擦除、编程操作。

复位后，位 MNLCK@FLASH_CR 置为 1。通过先向 FLASH_MKYR 寄存器写入 Flash 解锁值——0x8ACE 0246，再向 FLASH_MKYR 寄存器写入单次操作解锁值 0xC3C3 C3C3 或多次操作解锁值 0xB4B4 B4B4 后，才可以使得 FLASH_CR 解锁。两次正确的写解锁值操作完成后，位 MNLCK@FLASH_CR 被硬件清 0。软件可以将控制位 MNLCK@FLASH_CR 置 1 来锁定主程序区。FLASH_MKYR 寄存器锁定的是主程序区，解锁后，可以通过运行在 RAM 中的程序，主程序区中的程序，Boot 区的程序或者使用调试工具直接访问 Flash 操作寄存器的方式对主程序区进行操作。

特殊信息区的类 EEPROM 区可以被 FLASH_E2KYR 寄存器锁定。解锁过程也是两次写操作，通过先向 FLASH_E2KYR 寄存器写入 Flash 解锁值——0x9BDF 1357，再向 FLASH_E2KYR 寄存器写入单次操作解锁值 0xC3C3 C3C3 或多次操作解锁值 0xB4B4 B4B4 后，才可以使得该区域被解锁，同时控制位 E2LCK@FLASH_CR 被硬件清 0。软件可以将控制位 E2LCK@FLASH_CR 置 1 来锁定特殊信息区的类 EEPROM 区。FLASH_E2KYR 寄存器锁定的是特殊信息区的类 EEPROM 区，该区域解锁后，可以通过运行在 RAM 中的程序，主程序区中的程序，Boot 区的程序或者使用调试工具直接访问 Flash 操作寄存器的方式对其进行操作。

特殊信息区的代码保护区（Protect Block）、客户信息区（Customer Block）、OTP 区和 Boot 区（Boot Rom）可以被 FLASH_IKYR 寄存器锁定。解锁过程也是两次写操作，通过先向 FLASH_IKYR 寄存器写入 Flash 解锁值——0xABCD 5678，再向 FLASH_IKYR 寄存器写入单次操作解锁值 0xC3C3 C3C3 或多次操作解锁值 0xB4B4 B4B4 后，才可以使得该区域被解锁，同时控制位 INFLCK@FLASH_CR 被硬件清 0。软件可以将控制位 INFLCK@FLASH_CR 置 1 来锁定特殊信息区的代码保护区（Protect Block）、客户信息区（Customer Block）、OTP 区和 Boot 区（Boot Rom）。FLASH_IKYR 寄存器锁定的是特殊信息区的代码保护区、客户信息区、OTP 区和 Boot 区，其中代码保护区、客户信息区和 OTP 区可以通过运行在 RAM 中的程序，主程序区中的程序，Boot 区的程序或者使用调试工具直接访问 Flash 操作寄存器的方式来实现操作；而 Boot 区只能通过使用调试工具直接访问 Flash 操作寄存器来实现操作。

对于 Flash 单次解锁操作，当 MNLCK@FLASH_CR 或 E2WRE/INFLCK@FLASH_CR 被清 0 后，只允许对 Flash 进行一次编程或擦除操作，操作完成后 MNLCK@FLASH_CR 或 E2WRE/INFLCK@FLASH_CR 被硬件自动置 1，再次锁定 Flash 相关区块。

对于 Flash 多次解锁操作，MNLCK@FLASH_CR 或 E2WRE/ INFLCK@FLASH_CR 被清 0 后，允许对 Flash 进行多次编程或擦除操作，但操作完成后，必须将 MNLCK@FLASH_CR 或 E2WRE/INFLCK@FLASH_CR 位软件置 1，才能再次锁定 Flash 相关区块。

表8-4 各解锁寄存器解锁值及说明

寄存器	功能	Flash初始解锁值	Flash单次操作解锁值	Flash多次操作解锁值/锁定值	说明
FLASH_MKYR	解锁主程序区	0x8ACE 0246	0xC3C3 C3C3	0xB4B4 B4B4	通过运行在RAM中的程序，主程序区中的程序，Boot区的程序或者使用调试工具直接访问
FLASH_E2KYR	解锁特殊信息区的类EEPROM区	0x9BDF 1357	0xC3C3 C3C3	0xB4B4 B4B4	通过运行在RAM中的程序，主程序区中的程序，Boot区的程序或者使用调试工具直接访问
FLASH_IKYR	解锁特殊信息区的代码保护区（Protect Block）、客户信息区（Customer Block）和Boot区（Boot Rom）	0xABCD 5678	0xC3C3 C3C3	0xB4B4 B4B4	代码保护区、客户信息区和OTP区可以通过运行在RAM中的程序，主程序区中的程序，Boot区的程序或者使用调试工具直接访问Flash操作寄存器来实现；而Boot区只能通过使用调试工具直接访问Flash操作寄存器来实现。

8.3.3.3 Flash 操作定时器功能

Flash 控制存储器 FLASH_CR 具有一个操作定时器，Flash 操作定时器的计数值是以系统时钟为时钟源，Flash 操作定时器的计数值必须小于 Flash 操作定时器上限值，并且大于零，才可以对 Flash 控制寄存器 FLASH_CR 进行修改，或者向目的地址写入一个 32bit/16bit 数据。如果启动一次 Flash 操作时，即 STRT@FLASH_CR 置 1 或者启动对 Flash 的写操作时，Flash 操作定时器计数值不在有效的时间窗口内，Flash 操作不被执行，且位 PGWERR@FLASH_SR 置 1。Flash 操作定时器的具体用法可参考下图。

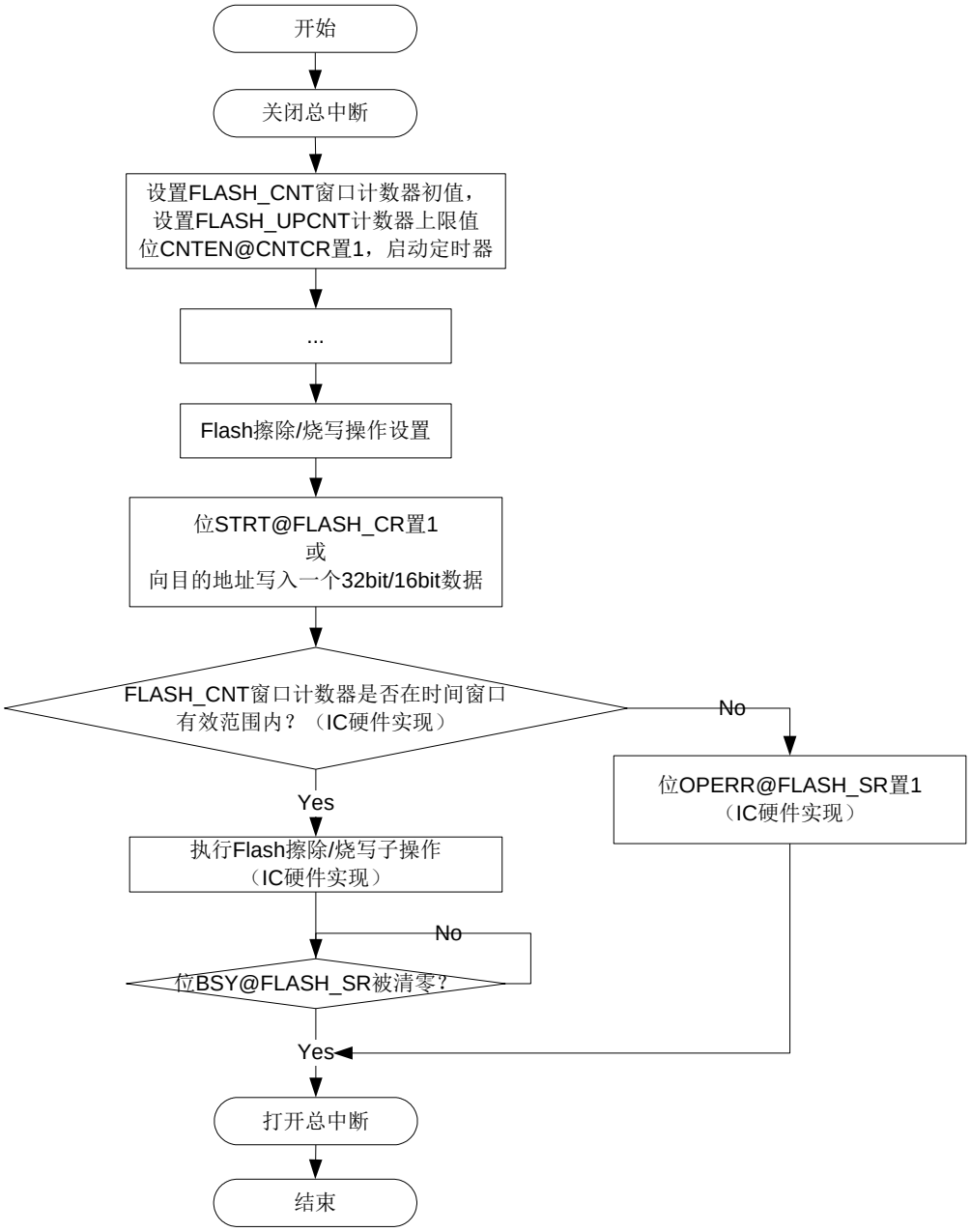


图8-3 Flash 操作定时器使用流程图

8.3.3.4 主程序区整体擦除

Flash 操作提供了整片擦除功能，可以擦除主存储块区的内容。具体步骤如下：

（1）FLASH_MKYR 依次写入 Flash 解锁值和单次操作解锁值，确保 Flash 主程序区不处于锁定状态；

（2）检查位 BSY@FLASH_SR 来判定 Flash 存储器是否处于运行状态；

（3）当位 BSY@FLASH_SR 为 0 时，写 Flash 主程序区整体擦除命令字到 FLASH_CR 寄存器；

（4）通过将位 STRT@FLASH_CR 置 1 来启动整片擦除操作；

（5）检查位 BSY@FLASH_SR 来判断擦除指令是否执行完毕；

（6）当位 BSY@FLASH_SR 为 0 时，操作完成。

位 EOP@FLASH_SR 预示操作的结束，该位置 1，表示操作执行完毕。所有 Flash 数据擦除后被复位为 0x0000 0000。

流程图如下：

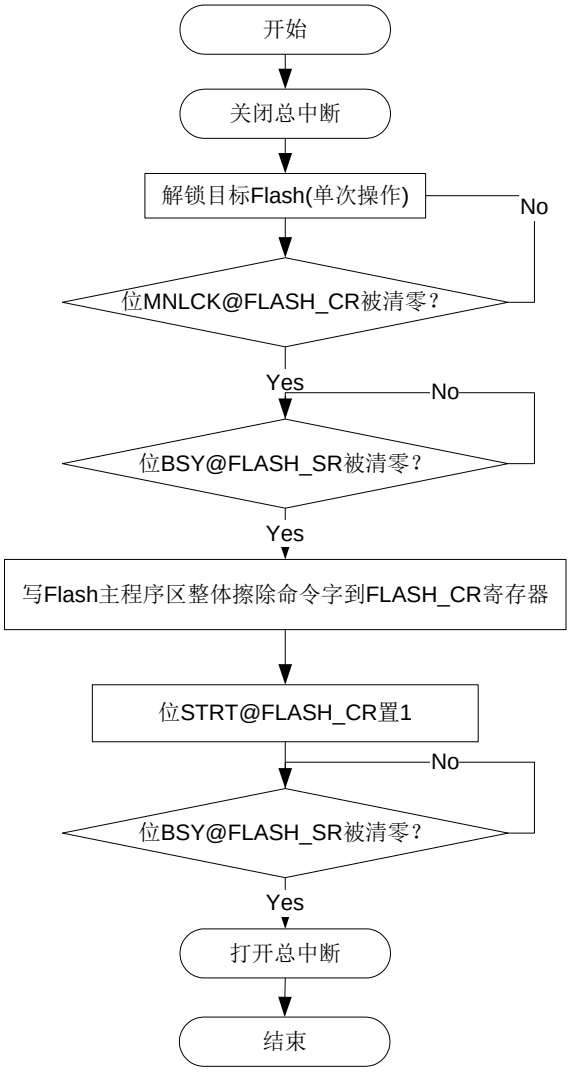


图8-4 主程序区整体擦除

8.3.3.5　　主程序区扇区擦除

Flash 存储器的每扇区都可以被独立擦除，不影响其它扇区内容。Flash 操作擦除扇区步骤如下：

（A）单独擦除某一扇区

（1）FLASH_MKYR 依次写入 Flash 解锁值和单次操作解锁值，确保 Flash 主程序区不处于锁定状态；

（2）检查位 BSY@FLASH_SR 来判定没有 Flash 存储器是否处于运行状态；

（3）当位 BSY@FLASH_SR 为 0 时，写主程序区扇区擦除命令字和待擦除扇区号码到 FLASH_CR 寄存器；

（4）通过将位 STRT@FLASH_CR 置 1 来启动扇区擦除操作；

（5）检查位位 BSY@FLASH_SR 来判断擦除指令是否执行完毕；

（6）当位 BSY@FLASH_SR 为 0 时，操作完成。

流程图如下：

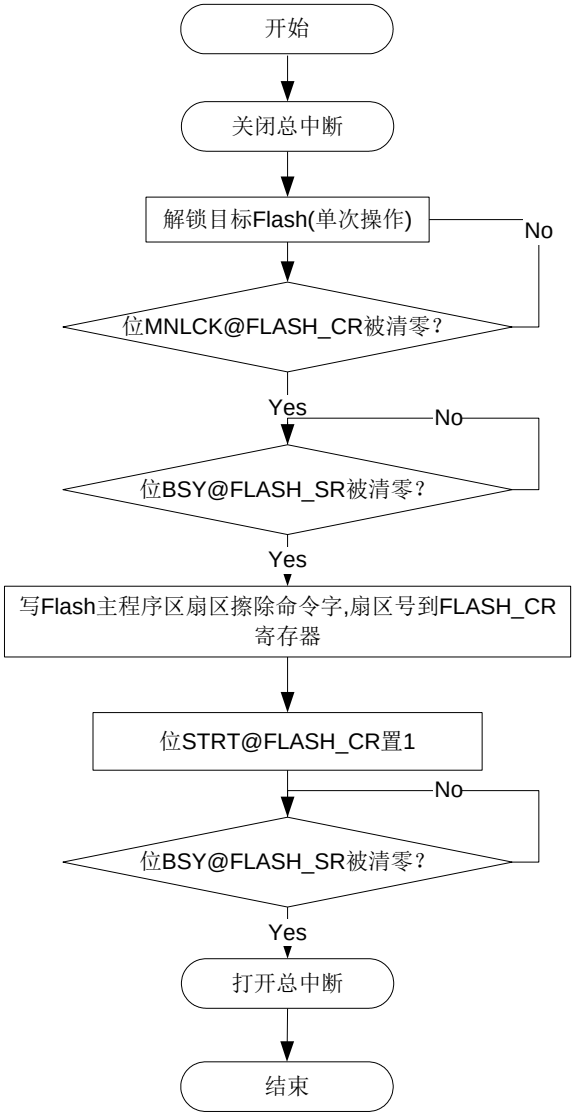


图8-5　　主程序区按扇区擦除（单一扇区）

（B）擦除多个扇区

（1）FLASH_MKYR 依次写入 Flash 解锁值和多次操作解锁值，确保 Flash 主程序区不处于锁定状态；
（2）检查位 BSY@FLASH_SR 来判定没有 Flash 存储器是否处于运行状态；
（3）当位 BSY@FLASH_SR 为 0 时，写主程序区扇区擦除命令字和扇区代码到 FLASH_CR 寄存器；
（4）通过将位 STRT@FLASH_CR 置 1 来启动扇区擦除操作；
（5）检查位 BSY@FLASH_SR 来判断擦除指令是否执行完毕；
（6）重复（3）-（5），擦除其它扇区；
（7）位 MNLCK@FLASH_CR 置 1，操作完成。
操作流程图如下图：

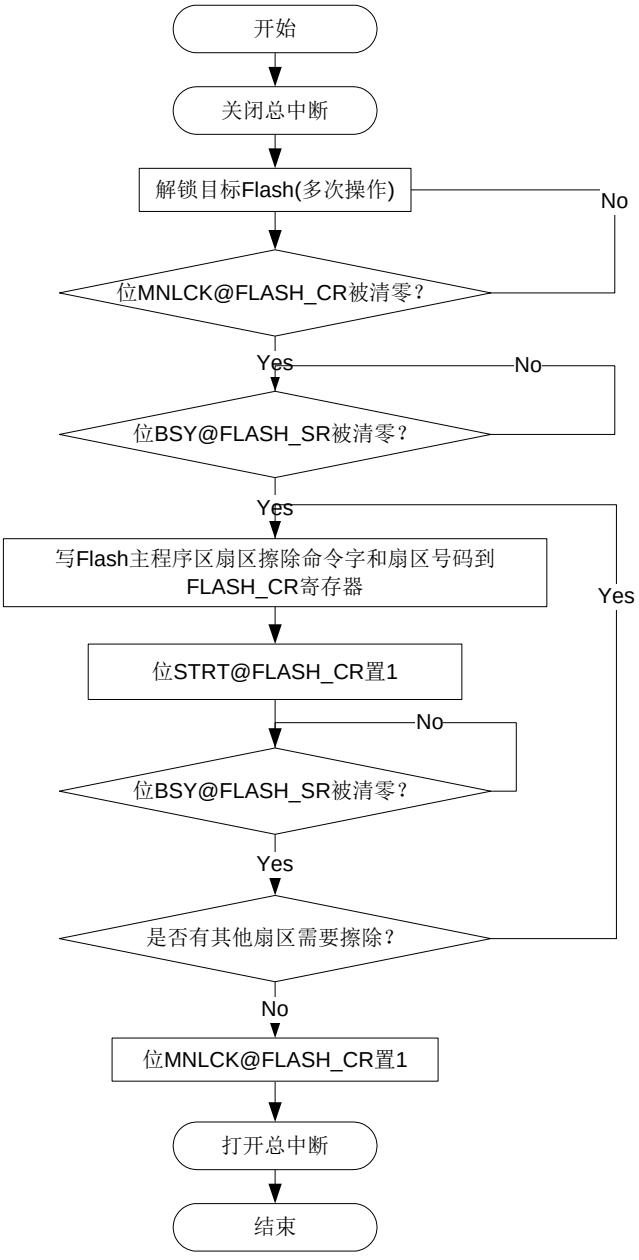


图8-6　　主程序区按扇区擦除（多个扇区）

当目标擦除扇区被用来取指令或访问数据时，相应的擦除操作无效，但 Flash 操作将不提供任何通知，因此确保目标擦除扇区地址的正确性是很有必要的。位 EOP@FLASH_SR 预示操作的结束。

8.3.3.6　　主程序区编程

Flash 操作提供了一个 32 位字/16 位半字编程功能，用来修改 Flash 主存储块内容。对 Flash 主程序区编程每次可以写入 32 位/16 位，地址必需按 32 位/16 位对齐，若未对齐，相应操作无效，相关错误标志位置 1。当 FLASH_CR 寄存器操作命令字设置为主程序区编程时，在一个 Flash 地址写入一个字或半字将启动一次编程。在编程过程中(位 BSY@FLASH_SR 位为'1')，任何读写 Flash 的操作都会使 CPU 暂停，直到此次 Flash 编程结束。

当数据缓冲区使能（位 DCEN@FLASH_ACR 置 1）时，对主程序区进行编程操作，需在操作后，CRST 位置 1。

下面的步骤显示了字编程操作寄存器过程。

（A）单字或半字编程

（1）FLASH_MKYR 依次写入 Flash 解锁值和单次操作解锁值，确保 Flash 主程序区不处于锁定状态；

（2）检查位 BSY@FLASH_SR 来判定没有 Flash 存储器是否处于运行状态；

（3）当位 BSY@FLASH_SR 为 0 时，写 Flash 主程序区编程操作命令字到 FLASH_CR 寄存器；

（4）向目的地址写入一个 32 位字/16 位半字；

（5）检查位 BSY@FLASH_SR 来判断编程指令是否执行完毕；

（6）当位 BSY@FLASH_SR 为 0 时，操作完成。

操作流程图如下图：

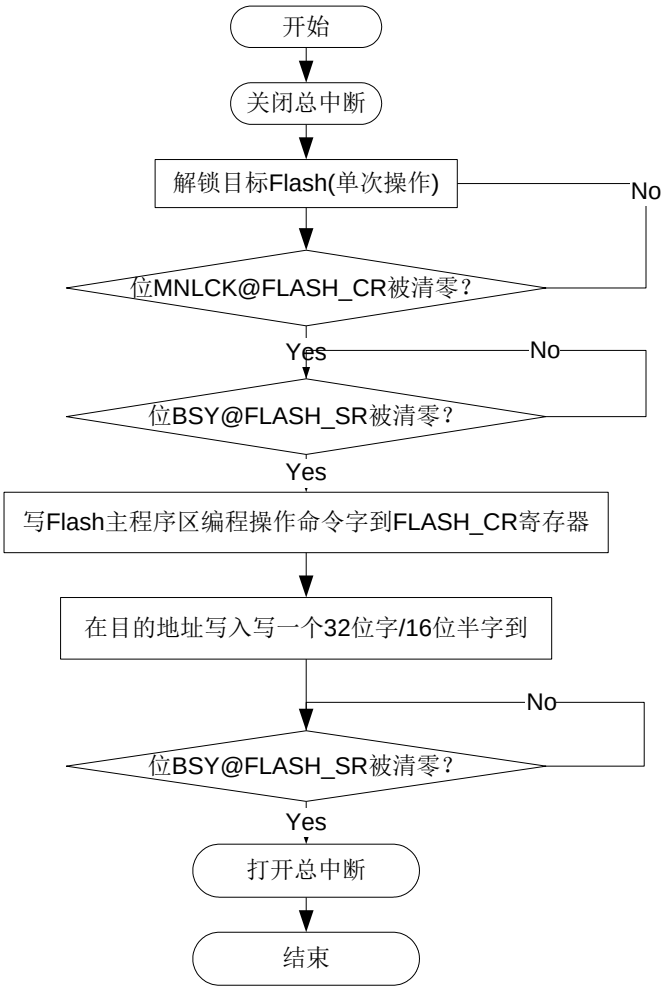


图8-7　　主程序区编程（单字或半字）

（B）连续写入多字编程

（1）FLASH_MKYR 依次写入 Flash 解锁值和多次操作解锁值，确保 Flash 主程序区不处于锁定状态；
（2）检查位 BSY@FLASH_SR 来判定 Flash 存储器是否处于运行状态；
（3）写 Flash 主程序区编程操作命令字到 FLASH_CR 寄存器；
（4）向目的地址写入一个 32 位字/16 位半字；
（5）检查位 BSY@FLASH_SR 来判断编程指令是否执行完毕；
（6）重复（4）-（5），直到多地址都编程结束；
（7）MNLCK@FLASH_CR 位置 1，操作完成。

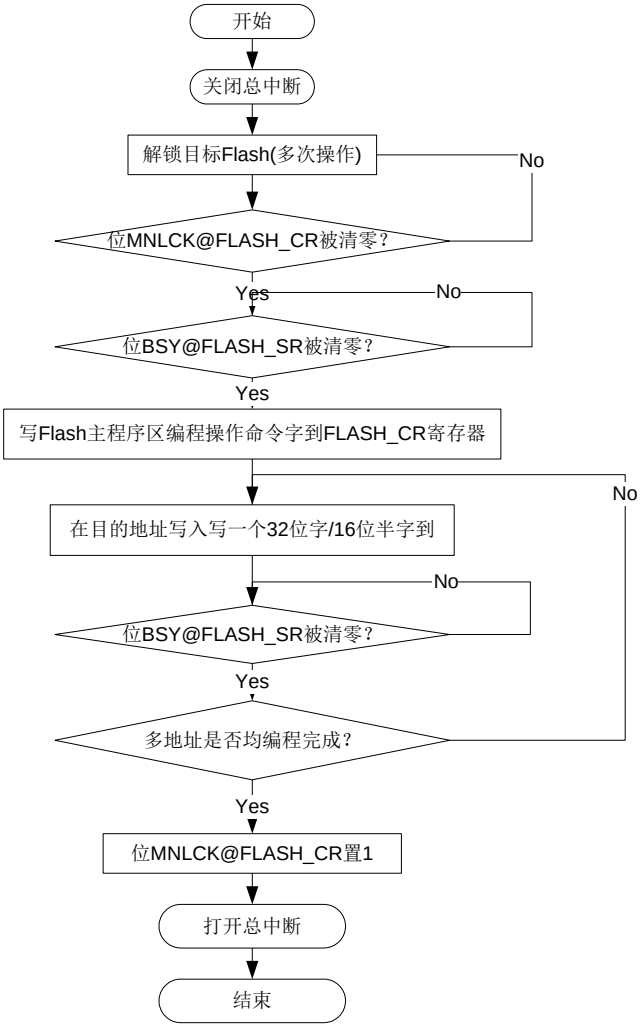


图8-8　　主程序区编程（连续写入多字编程）

对 Flash 存储器，每次可以写入 16 位或 32 位。主存储区擦除操作后，每 16 位或 32 位仅可以被编程一次，即被编程的 16 位或 32 位数据中有一个位如果不是 0，就不可以再次被烧写，否则编程错误标志位 PGWERR@FLASH_SR 被置 1。

如果 Flash 主程序区的某一扇区已经被写保护位保护，但仍对该扇区进行写操作，则写保护错误标志位 WRPRTERR @FLASH_SR 被置 1。

8.3.3.7　　类 EEPROM 存储区擦除编程

类 EEPROM 存储区擦除编程与主程序区按扇区擦除、编程相类似，不同的是 Flash 解锁寄存器不同，写入 FLASH_CR 寄存器的操作命令字不同，锁定存储区的标志位不同。

8.3.3.8 客户信息区（Customer Block）擦除编程

客户信息区（Customer Block）擦除编程与主程序区按扇区擦除、编程相类似，不同的是 Flash 解锁寄存器不同，写入 FLASH_CR 寄存器的操作命令字不同，锁定存储区的标志位不同。

8.3.3.9 OTP 区编程

OTP 区无法对其擦除，编程与主程序区编程相类似，不同的是 Flash 解锁寄存器不同，写入 FLASH_CR 寄存器的操作命令字不同，锁定存储区的标志位不同。

8.3.3.10 代码保护控制区（Protect Block）编程

代码保护控制区不支持按扇区擦除，代码保护控制区的编程与主程序区的编程相类似，不同的是 Flash 解锁寄存器不同，写入 FLASH_CR 寄存器的操作命令字不同，锁定存储区的标志位不同。(需要注意，增加了 Flash 区块划分功能，每个 sector 号与逻辑地址和物理地址的对应关系可能不同)

Flash 中的主程序区可以通过设置读保护防止被非法的读出；同样可以对 Flash 存储区的各扇区设置写保护，防止在程序跑飞的情况下不被意外地改变。对于 256KB 的 Flash 存储区，写保护的基本单位是 8 个扇区为一个保护单元。

(1) 读保护

读保护是通过设置读保护字节，写入后立即生效。

Flash 存储器提供了一个安全读保护功能来防止闪存中的用户代码被非法的读出，以保证用户代码安全。读保护有三个保护级别，分别是无读保护，低级别读保护和高级别读保护。

无读保护：当将读保护字节及其补码设为 0xA55A，不执行读保护，对 Flash、C-RAM 读操作不受限制。

低级别读保护：当设置读保护字节及其补码为 0xA55A 和 0xC33C 之外的其它值时，低级别读保护生效，此时读取到的读保护字节为 0xAAAA。从用户程序启动和从 Boot Rom 启动时，对 Flash 的访问无限制；在调试模式下（SWD）或在 RAM 中运行时，对 Flash 主存储器的访问受限，即使未选择写保护，也不可对 Flash 主存储器进行读、写、扇区擦除操作。但低级别读保护，在调试模式下（SWD）或在 SRAM 中运行时，整体擦除不受限制。低级别读保护状态下，执行内置 RAM 代码，通过 DMA 或其它任何方式间接地访问 Flash 主存储器的操作也被禁止。

在低级别保护下，可以通过调试模式下（SWD）向内置 SRAM 装载代码并执行代码来解除低级别读保护（整片擦除）,也可以通过使用中颖工具（SWD/ISP）执行整片擦除来解除低级别读保护。当保护字节被写入相应的值以后，只允许用户代码对 Flash 主程序存储器进行读操作(以非调试方式对 Flash 主存储器操作)。读保护状态下，可以通过在 Flash 主存储器中执行的代码进行编程(实现 IAP 或数据存储等功能)，但不允许在调试模式（SWD）下或在 SRAM 运行程序执行写或擦除操作(整片擦除除外)。

高级别读保护：当设置读保护字节及其补码为 0xC33C，高级别读保护生效。在高级别读保护下，调试功能（SWD）关闭，所以在调试模式下或在 SRAM 中运行代码，对 Flash 主存储器的访问都被禁止。此外，从 Boot Rom 区启动也被禁止。

(2) 写保护

写保护位用于对 Flash 各扇区进行擦除/编程保护，可以阻止对 Flash 存储器的意外操作。写保护字节的 WP[31:0]位为 1 可以使能指定扇区的写保护功能。对于小于或者等于 1MB 的 Flash 主程序存储区，是以 8 个扇区(1KB/扇区)Flash 存储区为单位提供保护，对被保护的扇区进行编程或擦除操作将不被执行，同时产生操作错误标志。对于超过 1MB 的 Flash 主程序存储区，是以 16 个扇区(1KB/扇区)为单位提供保护的。下表显示了 256KB Flash 主程序区，将 WP[31:0]位置 1 使能了哪些扇区保护。

表8-5 写保护位控制表

WP bit	被保护的页
WP[0]	Sector0 ~ Sector7
WP[1]	Sector8 ~ Sector15
WP[2]	Sector16 ~ Sector23
-	-
-	-
-	-
WP[30]	Sector240 ~ Sector247
WP[31]	Sector248 ~ Sector255

(3) 主程序区整体擦除保护

主程序区整体擦除保护位，该字节为 0x01，允许在普通运行模式（Normal）下执行 ME0 操作，否则在普通运行模式（Normal）下执行 ME0 操作无效。

8.4 启动配置（BootLoader）

对于 SH32F9001_32F9801 来说，系统上电或引脚复位后，系统硬件与引导区软件配合，根据当前系统的启动配置状态（读保护状态，引导区代码是否有效，复位时 Boot 引脚电平状态），选择 Flash 主程序存储区或者引导区来作为系统的启动区。

表8-6　　系统启动配置简表

系统配置状态			上电复位或引脚复位后状态		
读保护级别	引导区代码是否有效	复位时Boot引脚电平	系统启动区	Debug是否使能	是否进入ISP模式
无读保护	Yes	高	引导区	Yes	Yes
无读保护	Yes	低	Flash主程序存储区	Yes	No
无读保护	No	X	Flash主程序存储区	Yes	No
低级别读保护	Yes	高	引导区	Yes	Yes
低级别读保护	Yes	低	Flash主程序存储区	Yes	No
低级别读保护	No	X	Flash主程序存储区	Yes	No
高级别读保护	Yes	X	Flash主程序存储区	No	No
高级别读保护	Yes	X	Flash主程序存储区	No	No
高级别读保护	No	X	Flash主程序存储区	No	No

当选择 Flash 主程序存储区作为系统启动区，MEMMODE@FLASH_MEMRMP 硬件清 0，硬件自动将主程序存储区代码保持映像到地址 0x0000 0000。若选择引导区作为系统启动区，MEMMODE@ FLASH_MEMRMP 硬件置 1，硬件自动将引导区重映像到地址 0x0000 0000（只映射引导区前 1024 字节），而引导区 0x0FFF 8000 之后地址的内容仍可正常访问。系统启动流程图如下图：

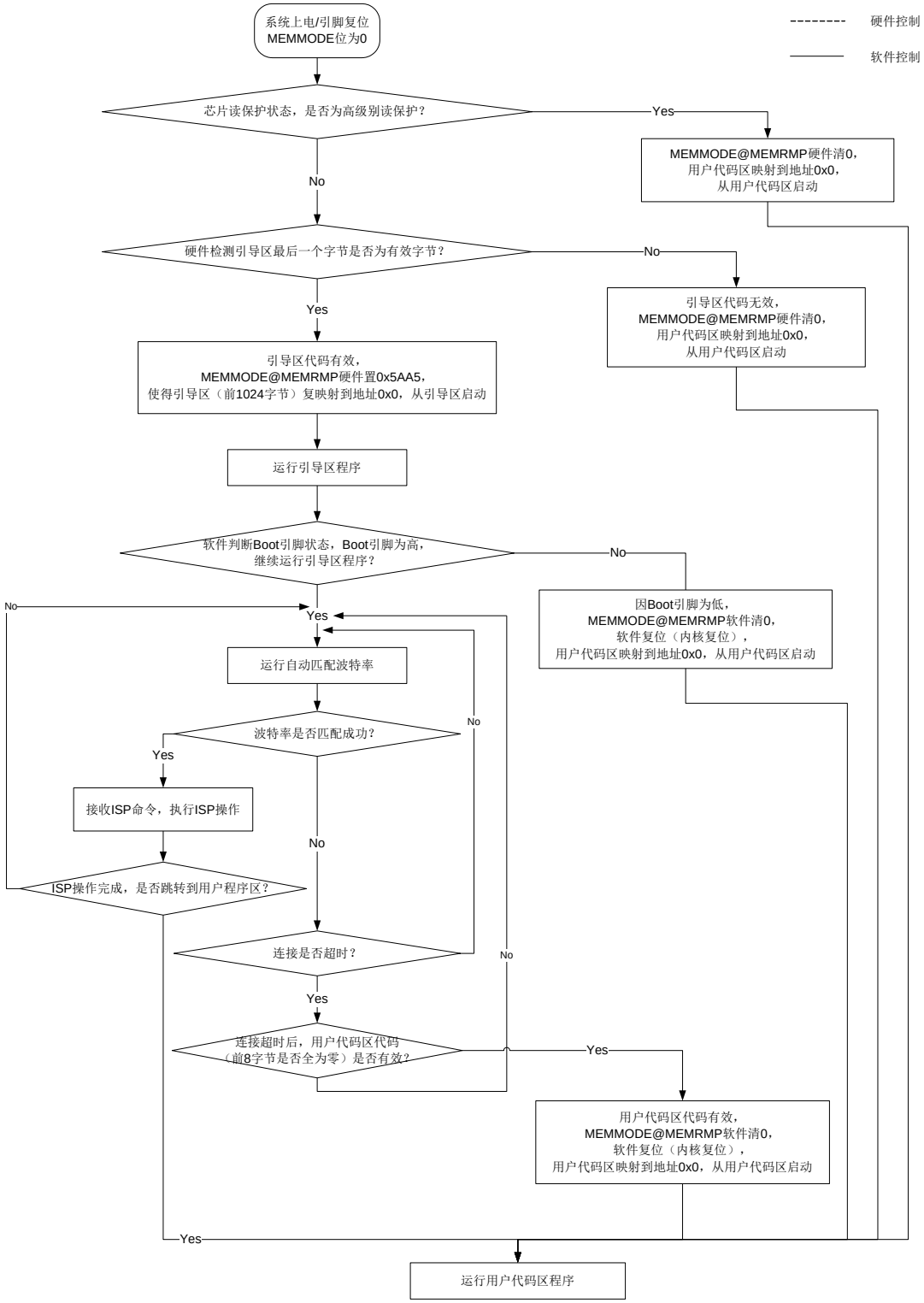

硬件控制
软件控制
系统上电/引脚复位
MEMMODE位为0
芯片读保护状态，是否为高级别读保护？
Yes
MEMMODE@MEMRMP硬件清0，
用户代码区映射到地址0x0，
从用户代码区启动
No
硬件检测引导区最后一个字节是否为有效字节？
No
引导区代码无效，
MEMMODE@MEMRMP硬件清0，
用户代码区映射到地址0x0，
从用户代码区启动
Yes
引导区代码有效，
MEMMODE@MEMRMP硬件置0x5AA5，
使得引导区（前1024字节）复映射到地址0x0，从引导区启动
运行引导区程序
软件判断Boot引脚状态，Boot引脚为高，
继续运行引导区程序？
No
因Boot引脚为低，
MEMMODE@MEMRMP软件清0，
软件复位（内核复位），
用户代码区映射到地址0x0，从用户代码区启动
No
Yes
运行自动匹配波特率
波特率是否匹配成功？
No
Yes
接收ISP命令，执行ISP操作
No
ISP操作完成，是否跳转到用户程序区？
连接是否超时？
No
Yes
连接超时后，用户代码区代码
（前8字节是否全为零）是否有效？
Yes
用户代码区代码有效，
MEMMODE@MEMRMP软件清0，
软件复位（内核复位），
用户代码区映射到地址0x0，从用户代码区启动
Yes
运行用户代码区程序

图8-9　　系统启动流程图

系统上电或引脚复位后，MEMMODE@ FLASH_MEMRMP硬件清0，硬件自动将主程序存储区代码保持映像到地址0x0，然后读取读保护位状态，若当前系统为高级别读保护状态，主程序存储区代码（用户代码）保持映像到地址0x0，系统从用户代码区启动。

若芯片非高级别读保护状态，硬件检测引导区最后一个字节是否为特定字节.若硬件检测到引导区最后一个字节为非特定字节，则保持主程序存储区代码（用户代码）保持映像到地址0x0000 0000，系统从用户代码区启动；若引导区最后一个字节为特定字节，表明引导区程序有效，MEMMODE@ FLASH_MEMRMP硬件置1，引导区（前1024字节）映像到地址0x0，系统从引导区启动，开始执行引导区程序。

运行引导区固化的程序，首先读取Boot引脚状态，若Boot引脚为低，MEMMODE@ FLASH_MEMRMP软件清0，用户代码区重新映像到地址0x0，然后启动软件复位（内核复位），从用户代码区启动；若Boot引脚为高，继续执行引导区程序，可进行ISP操作。ISP烧写流程首先启动自动匹配波特率功能，若自动匹配波特率成功，则可接收ISP命令，执行ISP操作；若匹配波特率连接超时，判断用户代码是否有效（用户代码前两个WORD不为零），代码有效则将MEMMODE@ FLASH_MEMRMP软件清0，用户代码区重新映像到地址0x0000 0000，然后启动软件复位（内核复位），从用户代码区启动。

进入ISP流程后首先匹配波特率，然后进入ISP命令的接收与对闪存存储器的写入过程，直到ISP完成；之后可通过硬件复位退出ISP流程。

BOOT引脚就是普通GPIO引脚，任意一个GPIO口都可以定义为BOOT引脚，取决于BootLoader程序，与硬件无关。

某个GPIO口一旦定义为BOOT引脚，由于BOOT一般要外加上下拉电阻，故作为GPIO口使用时功能是受限的（BootLoader完成后BOOT引脚即可作为GPIO口使用，直到下次上电或复位），用户要确保引脚电平不会影响启动模式。

8.5 在系统编程（ISP）

SH32F9001_32F9801支持ISP编程，出厂引导区不烧写任何程序，需要客户自行开发。

8.6 Flash 访问加速控制

SH32F9001_32F9801增加了Flash访问加速控制模块，用以平衡Cortex-M3的高速性能与Flash内存访问慢的矛盾。通常在高速访问Flash 内存时需要插入等待一定的时间（LATENCY@FLASH_ACR）。为了保证Flash能够正常被读取，将时钟和等待状态之间的举例关系：

系统时钟<25MHz	;选择零个等待状态
25MHz<系统时钟<50MHz	;选择一个等待状态
50MHz<系统时钟<75MHz	;选择两个等待状态

以此类推，请根据系统时钟选择对应的等待状态，可以提高系统运行效率。

为此SH32F9001_32F9801增加了两个预读缓冲区(指令和数据)，在程序运行时可以根据预测机制，提前将指令及数据读取到缓冲区(1KB)，CPU实际取指或取数据时就可以直接从缓冲区读取，无需额外的等待。

8.7 Flash 切分区块功能

为保证片上备份程序与主程序能够快速切换，SH32F9001_32F9801配置有FLASH切分区块功能。

与FLASH区块切分的相关代码选项参考“用户代码选项”。其中：

FLASH_BACKUP为FLASH分区使能位，用于控制FLASH分区功能是否使能；

FLASH_SELECT为FLASH逻辑地址映射切换控制位，仅在FLASH_BACKUP为‘1’时有效；

在Boot Loader以及Main程序中，允许对FLASH_BACKUP和FLASH_SELECT进行“擦除”以及“擦除 - 编程”的操作，此时需要通过将FLASH模块的LOCK寄存器写“0x5AA5”解锁后才可修改。具体操作方式与“客户信息区”烧写/擦除方式相同。

在修改FLASH_SELECT后，可通过软件复位(SW复位)或上电复位，产生Vector复位，使PC指针对应的逻辑地址映射完成切换。

上电复位或其他系统复位后，硬件会自动根据读到的FLASH_BACKUP以及FLASH_SELECT值，对FLASH的逻辑地址进行相应的映射处理。

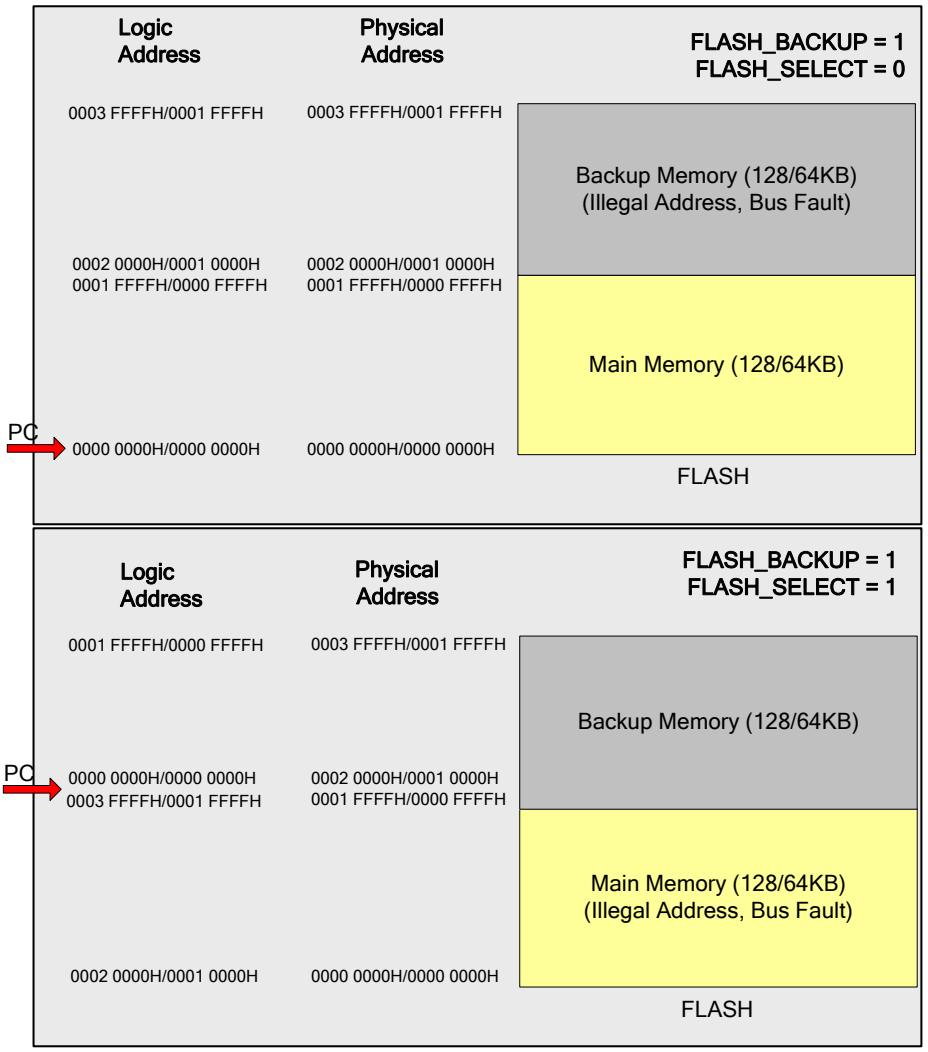


图8-10 256/128KB FLASH 区块切分后地址配置框图

（1） 当 FLASH_SELECT 为‘0’时，FLASH 区的逻辑地址与物理地址相符；且系统复位后，PC 指针内容为：指向逻辑地址 0000 0000H，物理地址 0000 0000H；

（2） 当 FLASH_SELECT 为‘1’时，FLASH 区的主 Memory 块和备份 Memory 块的逻辑地址调转，此时：

A. Main Memory 块的逻辑地址 = 物理地址 + 0002 0000H/0001 0000H；

B. Backup Memory 块的逻辑地址 = 物理地址 - 0002 0000H/0001 0000H；

且系统复位后，PC 指针内容为：指向逻辑地址 0000 0000H，物理地址 0002 0000H/0001 0000H

当FLASH_SELECT的值为‘0’时，PC指针指向Main Memory，当PC指针超过Main Memory的逻辑地址范围（0000 0000H~0001 FFFFH/0000 FFFFH）时，则认为程序跑飞，产生总线Fault；当FLASH_SELECT的值为‘1’时，使能FLASH主/备份区块地址交换功能，PC指针指向Backup Memory，当PC指针超过Backup Memory的逻辑地址范围（0000 0000H~0001 FFFFH/0000 FFFFH）时，则认为程序跑飞，产生总线Fault；

总之，当FLASH分区块功能开启时，通过FLASH_SELECT可选择对应区块作为PC指针访问区域，若PC指针访问超出选中区块的范围，则认为程序跑飞，发生总线Fault。但需要注意，此时非选中的FLASH区块仍可作为ROM进行读/写操作，只是不允许运行程序。

8.8 寄存器

FLASH 模块寄存器列表 (基地址: 0x4004 4800)

地址	寄存器名	说明
0x4004 4800	ACR	Flash访问控制寄存器
0x4004 4804	MKYR	Flash主程序存储区解锁控制寄存器
0x4004 4808	E2KYR	类EEPROM存储区解锁控制寄存器
0x4004 480C	SR	Flash状态和清除寄存器
0x4004 4810	CR	Flash控制寄存器
0x4004 4818	RPR	Flash读保护寄存器
0x4004 481C	WRPR	Flash写保护寄存器
0x4004 4820	CNTR	Flash操作定时器计数寄存器
0x4004 4824	UPCNTR	Flash操作定时器上限值
0x4004 4828	CNTCR	Flash操作定时器启动寄存器
0x4004 482C	IKYR	Flash特殊信息存储区解锁控制寄存器
0x4004 4900	MEMRMP	内存映射选择寄存器

8.8.1 Flash 访问控制寄存器 (FLASH_ACR)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留				CRST	DCEN	ICEN	PRFTEN	保留					LATENCY[2:0]		
-				RW	RW	RW	RW	-					RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LOCK[15:0]**	**寄存器锁定位** 0x5AA5：解锁此寄存器 其他：锁定此寄存器 配置此寄存器时解锁位必须同时输入0x5AA5，操作完成后自动锁定，读取此LOCK位固定读出0x0000。
15 ~ 12	**保留**	**-**
11	**CRST**	**缓冲区复位** 0：缓冲区无复位 1：缓冲区复位
10	**DCEN**	**数据缓冲区使能** 0：关闭数据缓冲区 1：启用数据缓冲区
9	**ICEN**	**指令缓冲区使能** 0：关闭指令缓冲区 1：启用指令缓冲区
8	**PRFTEN**	**预取缓冲区使能** 0：关闭预取缓冲区 1：启用预取缓冲区
7 ~ 3	**保留**	**-**

2 ~ 0	**LATENCY[2:0]**	**这些位表示SYSCLK(系统时钟)周期与Flash访问时间的比例** 000：0个等待状态 001：1个等待状态 010：2个等待状态 011：3个等待状态 100：4个等待状态 101：5个等待状态 110：6个等待状态 111：7个等待状态

8.8.2 Flash 主程序存储区解锁控制寄存器(FLASH_MKYR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
MKY[31:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MKY[31:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	**MKY[31:0]**	**Flash主程序存储区解锁控制位** 依次写正确的Flash解锁值、操作解锁值到MKY[31:0]可以解锁Flash主程序存储区的操作，解锁成功后，位MNLCK@FLASH_CR被硬件清0，MKY[31:0]硬件清零。 对于Flash单次解锁操作，当位MNLCK@FLASH_CR被清0后，只允许对Flash进行一次编程或擦除操作，操作完成后位MNLCK@FLASH_CR自动置1，再次锁定Flash主程序存储区。 对于Flash多次解锁操作，当位MNLCK@FLASH_CR被清0后，允许对Flash进行多次编程或擦除操作，但操作完成后，必须将位MNLCK@FLASH_CR软件置1，才能再次锁定Flash主程序存储区。 这些位仅能被软件写

8.8.3 类 EEPROM 存储区解锁控制寄存器 (FLASH_E2KYR)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
E2KY[31:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
E2KY[31:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	E2KY[31:0]	**类EEPROM存储区解锁控制位** 依次写正确的Flash解锁值、操作解锁值到E2KY[31:0]可以解锁类EEPROM区，解锁成功后，位E2LCK@FLASH_CR被硬件清0，E2KY[31:0]硬件清零。 对于Flash单次解锁操作，当解锁成功后，只允许对Flash进行一次编程或擦除操作，操作完成后位E2LCK@FLASH_CR自动置1，再次锁定Flash相关区块。 对于Flash多次解锁操作，当解锁成功后，允许对Flash进行多次编程或擦除操作，但操作完成后，必须将位E2LCK@FLASH_CR软件置1，才能再次锁定Flash相关区块。 这些位仅能被软件写

8.8.4 Flash 状态和清除寄存器 (FLASH_SR)

偏移地址: 0x000C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留								STAERRC	PGWERRC	PGPERRC	WRPRTERRC	FLSERRC	保留	OPERRC	EOPC
-								*WO*	*WO*	*WO*	*WO*	*WO*	-	*WO*	*WO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
BSY	保留							STAERR	PGWERR	PGPERR	WRPRTERR	FLSERR	保留	OPERR	EOP
RO	-							*RO*	*RO*	*RO*	*RO*	*RO*	-	*RO*	*RO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	保留	-
23	STAERRC	**Flash状态错误标志清除位** 0：无效 1：清除
22	PGWERRC	**编程窗口错误标志清除位** 0：无效 1：清除
21	PGPERRC	**编程错误标志清除位** 0：无效 1：清除
20	WRPRTERRC	**写保护错误标志清除位** 0：无效 1：清除
19	FLSERRC	**Flash硬件错误状态清除位** 0：无效 1：清除
18	保留	-
17	OPERRC	**Flash操作错误标志清除位** 0：无效 1：清除
16	EOPC	**操作执行标志位** 0：无效 1：清除

15	**BSY**	**忙标志位** 该位指示Flash操作正在进行。在Flash操作开始时，该位被设置为1；在操作结束或发生错误时该位被清除0。
14 ~ 8	**保留**	**-**
7	**STAERR**	**Flash状态错误标志位** 在Flash未正常解锁的情况下，当试图对Flash擦除或编程，该位硬件置1。STAERRC@FLASH_SR写入1可以清除这位状态。 在STRT@FLASH_CR未置1的情况下，试图对Flash编程，该位硬件置1。STAERRC@FLASH_SR写入1可以清除这位状态。
6	**PGWERR**	**编程窗口错误标志位** 该位置1，表示启动一次Flash操作时，即位STRT@FLASH_CR置1或者启动对Flash的写操作时，Flash操作定时器计数值不在有效的时间窗口内，Flash操作不被执行。PGWERRC@FLASH_SR写入1可以清除这位状态。
5	**PGPERR**	**编程错误标志位** 试图对已烧写过的地址编程（某一bit已不为0），硬件置1。PGPERRC@FLASH_SR写入1可以清除这位状态。 注：进行编程操作设置之前，必须先清除位STRT@FLASH_CR。
4	**WRPRTERR**	**写保护错误标志位** 试图对写保护的Flash地址编程/擦除时，硬件置1。WRPRTERRC@FLASH_SR写入1可以清除这位状态。
3	**FLSERR**	**Flash硬件错误** Flash操作超时，该错误标志置1。FLSERRC@FLASH_SR写入1可以清除这位状态。
2	**保留**	**-**
1	**OPERR**	**Flash操作错误标志位** Flash操作失败，硬件置1。OPERRC@FLASH_SR写入1可以清除这位状态。具体错误详见PGWERR、PGPERR、WRPRTERR、STAERR错误标志位。
0	**EOP**	**操作执行标志位** 当Flash正确执行完编程/擦除操作，硬件置1。EOPC@FLASH_SR写入1可以清除这位状态。 注：每次执行完编程或擦除都会设置EOP@FLASH_SR状态。

8.8.5 Flash 控制寄存器 (FLASH_CR)

偏移地址: 0x0010

复位值: 0x0000 D000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CMD[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MNLCK	**E2LCK**	**保留**	**INFLCK**	**PSIZE**	**保留**		**STRT**	**SNB[7:0]**							
RW1s	*RW1s*	-	*RW1s*	*RW*	-		*RW*	*RW*							
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**CMD[15:0]**	**Flash操作命令字** 0xAA55：主程序区整体擦除(ME0) 0xE619：主程序区扇区擦除(MSE) 0x6E91：主程序区存储单元编程(MPG) 0xB44B：特殊信息区E2Prom区编程(IPG) 0x4BB4：特殊信息区E2Prom区按扇区擦除(类E2Prom)（E2SE） 0xF00F：特殊信息区OTP区编程（OPG） 0xD22D：特殊信息区Protect区编程(SPG) 0xE11E：特殊信息区BootRom区编程(BRPG) 其它：保留值 *注：只有在Flash操作解锁时才能修改此位。*
15	**MNLCK**	**主程序区锁定标志** 依次向FLASH_MKYR写入正确的Flash解锁值、操作解锁值后，该位被硬件清零，此时才允许对Flash主程序存储区进行擦除、编程操作。 该位只能硬件清零，软件置1。 在一次不成功的解锁操作后，下次系统复位前，该位不能再被改变。
14	**E2LCK**	**类EEPROM存储区锁定标志位** 依次向FLASH_E2KYR写入正确的Flash解锁值、操作解锁值后，该位被硬件清零，此时允许对类EEPROM存储区进行擦除、编程操作。 该位只能硬件清零，软件置1。
13	**保留**	**-**
12	**INFLCK**	**特殊信息区锁定标志位(代码保护区、客户信息区、产品信息区和引导区)** 依次向FLASH_IKYR写入正确的Flash解锁值、操作解锁值后，该位被硬件清零，此时允许对特殊信息区(代码保护区、客户信息区、产品信息区和引导区)进行擦除、编程操作。 该位只能硬件清零，软件置1。
11	**PSIZE**	**烧写操作位宽选择位** 0：32位同时烧写 1：16位同时烧写 *注：只有在Flash操作解锁时才能修改此位。*
10 ~ 9	**保留**	**-**
8	**STRT**	**开始标志位** 当该位为1时将触发一次操作。该位只可由软件置为1并在BSY@FLASH_SR变为1时清为0。 *注：只有在Flash操作解锁时才能修改此位。*
7 ~ 0	**SNB[7:0]**	**扇区擦除扇区选择位:** 00000000：扇区0 00000001：扇区1 00000010：扇区2 …….. 11111110：扇区254 11111111：扇区255 注：该位仅当主程序区扇区擦除或类EEPROM区按扇区擦除时才有效，主程序区按扇区擦除时，每个扇区大小是1024字节；EEPROM区按扇区擦除时，每个扇区大小是1024字节。

8.8.6 Flash 读保护寄存器 (FLASH_RPR)

偏移地址: 0x0018

复位值: 0x0000 XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RDP[15:0]															
RO															
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	RDP[15:0]	读保护 该寄存器包含由复位后客户信息区加载器加载的读保护选项字节。

8.8.7 Flash 写保护寄存器 (FLASH_WRPR)

偏移地址: 0x001C

复位值: 0xXXXX XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
WRP[31:0]															
RO															
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WRP[31:0]															
RO															
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

位	符号	说明
31 ~ 0	WRP[31:0]	写保护该寄存器包含由复位后代码保护区加载器加载的写保护字节。 0：写保护失效 1：写保护生效 注意：这些位为只读。

8.8.8 Flash 操作定时器计数寄存器 (FLASH_CNTR)

偏移地址: 0x0020

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CNT[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CNT[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	**CNT[31:0]**	**Flash操作定时器计数值寄存器** 基于系统时钟的Flash操作定时器，递减计数，递减至Flash操作定时器上限值后，才允许修改Flash控制寄存器，当Flash操作定时器计数值递减至零后，禁止修改Flash控制寄存器。

8.8.9 Flash 操作定时器上限值 (FLASH_UPCNTR)

偏移地址: 0x0024
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
UPCNT[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
UPCNT[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	**UPCNT[31:0]**	**Flash操作定时器上限值** Flash操作定时器计数值递减至Flash操作定时器上限值后，才允许修改Flash控制寄存器，Flash操作定时器计数值递减至零后，禁止修改Flash控制寄存器。

8.8.10 Flash 操作定时器启动寄存器 (FLASH_CNTCR)

偏移地址: 0x0028
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留															CNTEN
-															*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 1	保留	-
0	**CNTEN**	**Flash操作定时器启动位** 0：Flash操作定时器禁能 1：Flash操作定时器使能 Flash操作定时器计数递减到零，该位清0，Flash操作定时器禁能。

8.8.11 Flash 特殊信息存储区解锁控制寄存器 (FLASH_IKYR)

偏移地址: 0x002C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
INFO_KEY[31:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
INFO_KEY[31:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	**INFO_KEY[31:0]**	**Flash特殊信息存储区解锁控制位** 依次写Flash解锁值，操作解锁值到INFO_KEY[31:0]可以解锁Flash特殊信息区（代码保护区、客户信息区、OTP区和引导区），解锁成功后，位INFLCK@FLASH_CR被硬件清0，INFO_KEY[31:0]硬件清零。 对于Flash单次解锁操作，当解锁成功后，只允许对Flash进行一次编程或擦除操作，操作完成后位INFLCK@FLASH_CR硬件自动置1，再次锁定Flash相关区块。 对于Flash多次解锁操作，当解锁成功后，允许对Flash进行多次编程或擦除操作，但操作完成后，必须将位INFLCK@FLASH_CR软件置1，才能再次锁定Flash相关区块。 这些位仅能被软件写

8.8.12 内存映射选择寄存器 (FLASH_MEMRMP)

偏移地址: 0x0100

复位值: 0x0000 XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MEMMODE[15:0]															
RW															
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**MEMMODE[15:0]**	**内存映射选择(Memory mapping selection)** 这些位在复位后由硬件设置，软件可读可写。 0x5AA5: Boot区映像到地址0x0 其他: 主闪存存储器映像到地址0x0 任何时刻只要不满足0x5AA5，即映射到主闪存存储器

9. 电源控制（PWR）

SH32F9001_32F9801的系统工作电压(VDD)为2.0～5.5V。通过内置的电压调节器提供系统内核电源。模拟电路工作电压(AVDD)不能超过VDD±0.3V。

SH32F9001_32F9801内部有一套完整的上电复位(POR)和掉电检测(BOD)电路，无需再配置外部复位和检测电路。掉电检测在一些电池应用中具有重要意义。

SH32F9001_32F9801保留了独立的外部复位引脚NRST。

9.1 电源与时钟预热

- 内建电源预热计数器消除电源上电不稳定状态
- 内建振荡器预热计数器消除振荡器起振不稳定状态

SH32F9001_32F9801内建有电源上电预热计数器，主要是用来消除上电电压建立时的不稳定态，同时完成内部一些初始化序列，如读取客户代码选项等。

SH32F9001_32F9801还内建振荡器预热计数器，它能消除振荡器在下列情况下起振时的不稳定状态：上电复位，引脚复位，低电压复位，从停机模式唤醒，独立看门狗复位。

上电后，SH32F9001_32F9801会先经过电源上电预热计数过程，等待溢出后再进行振荡器的预热计数过程，溢出后开始运行程序。

表9-1 电源上电预热计数时间

上电复位/ 按键复位/低压复位		**IWDT复位 （不在停机模式下）**		**IWDT复位 （从停机模式下唤醒）**		**从停机模式下唤醒 （中断或事件唤醒）**	
电源上电 预热计数时间	振荡器上电 预热计数[注]	电源上电 预热计数时间	振荡器上电 预热计数	电源上电 预热计数时间	振荡器上电 预热计数	电源上电 预热计数时间	振荡器上电 预热计数
11ms	≈1ms	≈1ms	无	≈1ms	有	≈500us	有

表9-2 振荡器上电预热计数时间

振荡器类型	**预热计数时间**
32kHz晶振（LSE）	2^{13} X Tosc ≈ 256ms
高频晶振/陶振（HSE）	2^{17} X Tosc
内部低频128kHz RC（LSI）	2^{7} X Tosc ≈ 1ms
内部高频24MHz RC（HSI）	2^{7} X Tosc ≈ 5.4us
PLL时钟	约2ms

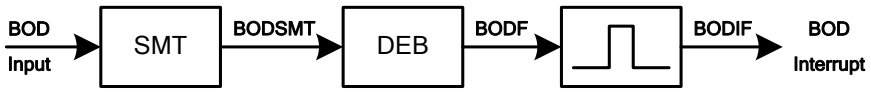
注：SH32F9001_32F9801上电时固定选择高频RC（HSI）作为系统时钟，振荡器预热计数定义为2^{7} X Tosc
说明：以上参数根据实际设计结果再进行调整。

9.2 掉电检测（BOD）

- 通过寄存器选择开启
- 上升和下降都能产生中断
- 可调的检测电压（档位误差V_{BOD}/2）
- 通过档位实现回差，上升有回差，下降无回差
- 硬件去抖动，去抖时间（T）约为40～80us（典型在60us左右）

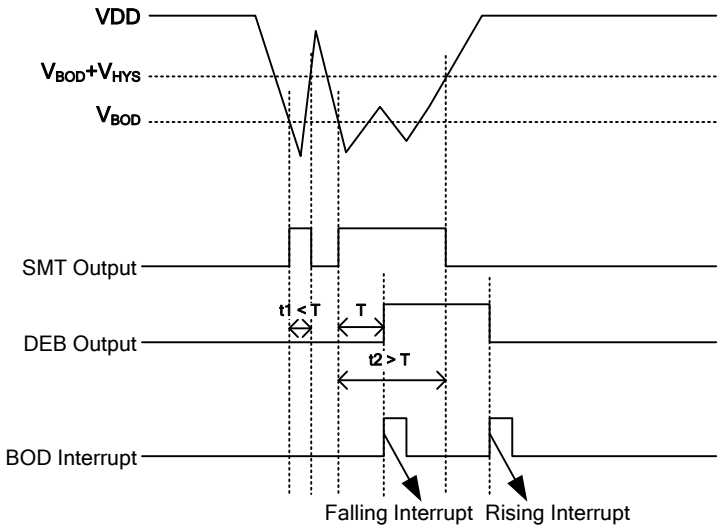
掉电检测用来监测电源电压，如果电压高于或低于指定值时产生内部标志，并可以产生中断。主要用于检测电源电压的变化，用于监控系统供电情况。在某些应用中，可用于低电压检测，系统软件可据此采取一些保护措施，如备份数据、保留现场等。

BOD内含带回差的比较电路，并具备硬件去抖动，基本结构如下：



下图给出了在开启上升和下降双向中断，上升带回差（BOD上档电压）条件下的BOD工作波形：

注意：档位电压仅是预定义值，允许进行微调，实际值以设计结果为准。



由于具备双向检测功能，BOD的使用变得很简单，典型使用步骤如下：

BOD中断开启步骤：

1）使能BOD模块（BODEN=1）；
2）设置BOD检测档位寄存器VBOD[3:0]；
3）设置BOD双沿检测BODMD[1:0]=10b；
4）清零BOD中断请求标志位BODIF；
5）置位BOD中断允许位BODIE（NVIC相应通道需配合开启）。

在 BOD 中断中，做如下工作：

1）清零BOD中断允许位BODIE；
2）查询BODF，确定刚才发生的是上升还是下降变化；
3）根据2）的查询结果执行相应的操作（如果是单向检测，则不需要查询BODF）；
4）清零BOD中断请求标志位BODIF；
5）置位BOD中断允许位BODIE；

BOD也能引起NMI中断，由IEN_BOD@SYSCFG_SAFR控制位使能。

9.3 低电压复位（LVR）

在交流电或大容量电池应用中，接通大负载后容易导致MCU供电暂时低于定义的工作电压。低电压复位可以应用于此，保护系统在低于设定电压下产生有效复位。

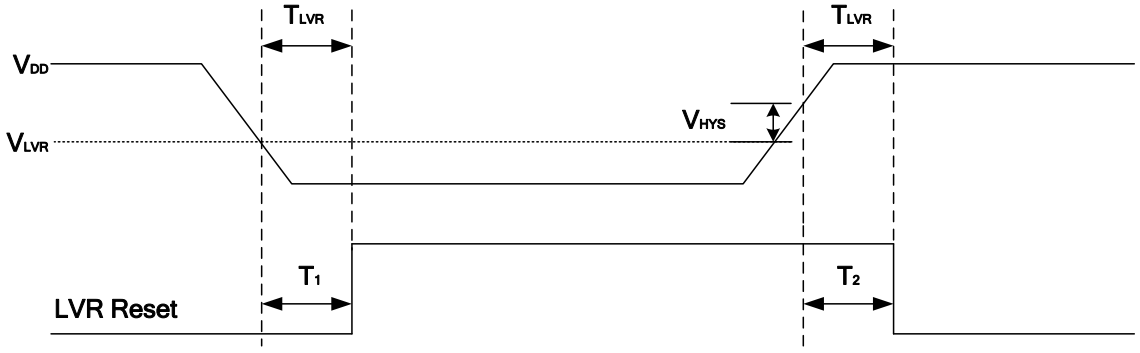
低电压复位（LVR）功能是为了监测供电电压，当供电电压低于设定电压V_{LVR}时，MCU将产生内部复位。

- 通过寄存器选择开启和设定 LVR 电压
- LVR 去抖动时间 T_{LVR} 为 40～80μs（典型在 60us 左右）
- 当供电电压低于设定电压 V_{LVR} 时，将产生内部复位

LVR功能打开后，具有以下特性（t表示电压低于设定电压V_{LVR}的时间）：

当$V_{DD} \le V_{LVR}$且$t \ge T_{LVR}$时产生系统复位。

当$V_{DD} > V_{LVR}$或$V_{DD} < V_{LVR}$，但持续时间 $< T_{LVR}$时不会产生系统复位。



上图中, V_{DD}为电源电压, V_{LVR}为LVR检测电压, V_{HYS}为回差。

9.4 低功耗管理

SH32F9001_32F9801在设计之初就注重了低功耗设计，重点降低系统运行功耗。除此之外，用户还可以关注如下几个方面，以进一步降低系统功耗：

- 使用低电压供电；
- 降低系统时钟；
- 关闭APB和AHB总线上未被使用的外设时钟；
- 降低GPIO和外设接口电路上电流；

用户需要根据最低电源消耗、最快速启动时间和可用的唤醒资源等条件，选定一个合适的低功耗模式。

- 睡眠模式(Sleep Mode)：基本低功耗模式，MCU内核停止运行，但所有外设包括内核的核心外设，如NVIC、SysTick等仍在运行；
- 停机模式(Stop Mode)：深度低功耗模式，除了内核停机，绝大部分外设都停止运行，除了担负唤醒任务的个别外设可开启，如IWDT、LVR等；

表9-3　　低功耗模式一览

模式	进入	唤醒	唤醒延时	VDD/AVDD电源域	LDO状态
睡眠模式	WFI/WFE 保持SLEEPDEEP=0	任一中断/唤醒事件/复位信号	无	1. Cortex-M3内核时钟停止； 2. 振荡器和电源保持原状态，外设时钟和部分模拟外设电源可软件关闭；	正常
停机模式	WFI/WFE 保持SLEEPDEEP=1	任一外部中断/复位信号	1. HSI唤醒时间； 2. HSE和PLL唤醒时间（假设有使用）； 3. LDO唤醒时间；	1. HSI、HSE以及PLL均被禁用，使用这部分时钟的外设全部停止； 2. LSI可开启，使用此时钟外设可正常工作； 3. 128kHz WDT RC正常工作，且不可关闭； 4. 部分模拟外设电源可软件关闭；	低功耗

9.4.1 睡眠模式

睡眠模式能够降低系统功耗，在此模式下，程序中止运行，CPU时钟停止，但外部设备时钟继续运行。睡眠模式下，CPU在确定的状态下停止，并在进入睡眠模式前所有CPU的状态都被保存，SRAM和寄存器内容被保留，GPIO保持运行时状态。

进入睡眠模式

保持CortexTM-M3系统控制寄存器中的SLEEPDEEP位处于复位状态，执行WFI或WFE指令就可以进入睡眠模式。根据Cortex-M3系统控制寄存器中的SLEEPONEXIT位的值，有2个CPU选项影响进入睡眠模式：

- SLEEP-NOW：如果SLEEPONEXIT位被清除，当WFI或WFE被执行时，微控制器立即进入睡眠模式；
- SLEEP-ON-EXIT：如果SLEEPONEXIT位被置位，系统从最低优先级的中断处理程序中退出时，微控制器就立即进入睡眠模式。

说明：SLEEP-ON-EXIT是CM3特有的“自动睡眠”的机制——它可以被编程为从中断服务例程返回后立即睡眠。这样一来，处理器的所有工作就只是响应中断，其它时间都在睡眠。

SLEEP-ON-EXIT只支持WFI指令进入，不支持WFE指令。

退出睡眠模式

如果执行WFI指令进入睡眠模式，任意一个被嵌套向量中断控制器响应的外设中断都能将系统从睡眠模式唤醒[注]。

如果执行WFE指令进入睡眠模式，则一旦发生唤醒事件时，微处理器都将从睡眠模式退出。唤醒事件可以通过下述方式产生：

- 在外设控制寄存器中使能一个中断，而不是在NVIC(嵌套向量中断控制器)中使能，并且在Cortex-M3系统控制寄存器中使能SEVONPEND位。当MCU从WFE中唤醒后，外设的中断挂起位和外设的NVIC中断通道挂起位(在NVIC中断清除挂起寄存器中)必须被清除。
- 配置一个外部或内部的EXTI线为事件模式。当MCU从WFE中唤醒后，因为与事件线对应的挂起位未被设置，不必清除外设的中断挂起位或外设的NVIC中断通道挂起位。该模式唤醒所需的时间最短，因为没有时间损失在中断的进入或退出上。

说明：

（1）当从WFI 唤醒时，要根据异常系统的优先级来决定是否唤醒。只有当该中断的优先级比当前优先级要高（如果是在中断服务例程中使用WFI），并且比BASEPRI 掩蔽的高时，才唤醒处理器并执行ISR。但如果PRIMASK 置位，则依然唤醒处理器，而ISR 不执行。

（2）WFE 则不管优先级和掩蔽情况如何，只要SETONPEND 置位，任何一个事件发生时一定把处理器唤醒。至于是否执行ISR，则与WFI 的情况相同。

关于如何退出睡眠模式，更多的细节参考下表。

表9-4　　SLEEP-NOW 模式

SLEEP-NOW模式	说明
进入	在以下条件下执行WFI(等待中断)或WFE(等待事件)指令： – SLEEPDEEP = 0 和 – SLEEPONEXIT = 0 参考Cortex-M3系统控制寄存器。
退出	如果执行WFI进入睡眠模式： 中断：参考中断向量表 如果执行WFE进入睡眠模式： 唤醒事件：参考唤醒事件管理
唤醒延时	无

表9-5　　SLEEP-ON-EXIT 模式

SLEEP-ON-EXIT模式	说明
进入	在以下条件下执行WFI指令： – SLEEPDEEP = 0 和 – SLEEPONEXIT = 1 参考Cortex-M3系统控制寄存器。 注：为了进入睡眠模式，所有的外部中断的请求位(挂起寄存器(EXTI_PR))标志必须被清除，否则睡眠模式的进入流程将会被跳过，程序继续运行。
退出	中断：参考中断向量表
唤醒延时	无

复位信号（RESET，WDT，BOD、LVR）都能退出睡眠模式。

系统唤醒后，立即恢复CPU时钟，如果中断唤醒还需要执行中断服务程序，随后跳转到进入睡眠模式指令之后的指令。唤醒后SRAM保持不变，而系统寄存器和外设寄存器的值根据不同功能模块设置可能会改变（时钟和使能都没关）。

如果是复位信号唤醒，则系统从地址0x0000 0000开始执行，SRAM和寄存器按照复位规则处理。

9.4.2　停机模式

停机模式可以使芯片进入功耗非常低的状态。停机模式将停止CPU和外围设备的所有时钟信号（除LSI和LSE可在停机模式下开启），并根据控制要求手动关闭部分Analog模块的供电，同时还能设置内核电压调节器进入低功耗状态。如果IWDT使能，IWDT模块将继续工作，LCD模块开启时也处于工作状态，LVR电路则任何模式都开启。在进入停机模式前所有CPU的状态都被保存，SRAM和寄存器内容被保留，GPIO保持运行时状态，但PWM输出（含PWM和PCA模块）将固定被切换到高阻状态。

说明：对普通外设，停机时虽然模块使能未关闭，但由于时钟切断，模块处于停止状态，模块引脚电平保持为停机前电平。但对PWM模块需要强行切换到高阻输出，否则可能会损坏外部电路。应用上，建议用户在进入停机前手动关闭PWM，在唤醒后重新初始化PWM。

注意：PWM模块在正常使能情况下，只要时钟被关闭（通过RCC_APB0ENR寄存器），即立刻切换到高阻输出状态。

停机模式时HSE、HSI、LSE、LSI和PLL电路都被自动关闭，若LCD电路开启，则LSI或LSE时钟仍保持输出，不会关闭。

系统进入停机模式时，RCC模块的SW、SWS、PLLON、HSEON、LSEON、PLLRDYIE、HSERDYIE、LSERDYIE等控制位保持原值不变，PLLRDY、HSERDY、LSERDY、PLLRDYIF、HSERDYIF、LSERDYIF等标志位还原为复位值，退出停机模式后硬件恢复系统时钟。

停机模式下，以下模块可以正常运行：

- 独立看门狗（IWDT），一旦启动IWDT，除了系统复位，它不能再被停止[注]；
- 低频RC时钟（128kHz RC），可以软件开启和关闭；
- 唤醒定时器（Wakeup Timer），使用TIM5-8定时器，要求配置为128kHz WDT RC或LSE时钟源，或外部异步时钟源；
- LCD驱动器，当LCD驱动器开启时，LSI/LSE在停机模式下不会被关闭；

注：默认情况下IWDT在停机模式下关闭，可以通过IWDTPD@IWDT_CR寄存器设置为停机模式下开启，这种情况下IWDT会唤醒停机；TTL功能在停机模式下关闭，CMOS输入被打开。

在该模式下如果ADC电路未关闭，则这些外设仍然消耗电流。可以设置这些模块的寄存器使能位来关闭这些外设。

停机模式下内部LDO固定进入低功耗状态。

停机模式下时钟安全监控（CSM）模块无效，睡眠模式下CSM可正常工作。

进入停机模式

关于如何进入停机模式，详见表9-6。

- 如果正在进行闪存编程，直到对闪存访问完成，系统才进入停机模式；
- 如果正在进行对AHB/APB的访问，直到对AHB/APB访问完成，系统才进入停机模式；
- 如果正在进行DMA传输，则DMA传输暂停直到唤醒，但当前正在进行的一次DMA操作要完成，系统才进入停机模式；
- 如果在中断服务程序中，或有中断在排队，系统正常进入停机模式，但EXTI、Wakeup Timer二类中断除外[注]；

注：如果在进入停机前有EXTI、Wakeup Timer中断产生时将跳过停机流程，除非清除掉相应的中断请求标志。

表9-6　　STOP 模式

STOP模式	说明
进入	在以下条件下执行WFI(等待中断)或WFE(等待事件)指令： – 设置Cortex-M3系统控制寄存器中的SLEEPDEEP位 注：为了进入停机模式，所有的外部中断的请求位(挂起寄存器(EXTI_PR))标志必须被清除，否则停机模式的进入流程将会被跳过，程序继续运行。
退出	如果执行WFI进入停止模式： 设置任一外部中断线为中断模式(在NVIC中必须使能相应的外部中断向量)。参见NVIC的“中断向量表”。 如果执行WFE进入停止模式： 设置任一外部中断线为事件模式。参见EXTI的“唤醒事件管理”。
唤醒延时	HSI RC唤醒时间 + 电压调节器从低功耗唤醒的时间。

退出停机模式

关于如何退出停机模式，详见表9-6。

当进入停机模式后，HSI 、HSE、PLL时钟会被自动关闭， LSE、LSI时钟不能被关闭（如果进入停机模式前被打开）。当一个中断或唤醒事件导致退出停机模式后，系统时钟仍然运行停机模式前的时钟，和系统上电时类似，需要等待电源Warmup和时钟Warmup。

复位信号（RESET，IWDT，LVR）能退出停机模式，复位后程序从0x0000 0000地址开始运行。

如果通过中断唤醒退出停机模式，还需要执行中断服务程序，随后跳转到进入停机模式指令之后的指令。

中断唤醒源包括外部输入中断（EXTI）、唤醒定时器中断（Wakeup Timer）。

如果中断使能未开启，还可以通过EXTI事件、唤醒定时器溢出事件进行唤醒。

说明：EXTI、Wakeup Timer都是既能产生中断又能产生事件的设备，Wakeup Timer事件是不可屏蔽的，只要开启并符合条件必定会发生，因此，Wakeup Timer唤醒停机模式实际上是事件的结果（先有事件后有中断），即使Wakeup Timer不开中断也能够唤醒，开中断的结果是唤醒以后还要进入相应中断服务程序处理。

9.4.3　低功耗模式下的调试

默认情况下，如果在进行调试微处理器时，使微处理器进入低功耗模式，将失去调试连接。这是因为Cortex-M3的内核失去了时钟。 然而，通过设置DBGMCU_CR寄存器中的某些配置位，可以在使用低功耗模式下调试软件。

更多的细节请参考调试接口有关“低功耗模式下调试支持”的内容。

9.5　寄存器

SYSCFG 模块寄存器列表 (基地址: 0x4004 3800)

地址	寄存器名	说明
0x4004 3800	PWRCR	电源控制寄存器
0x4004 3804	PWRSR	电源状态寄存器

9.5.1 电源控制寄存器 (SYSCFG_PWRCR)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留				LVREN	VLVR[1:0]		BODEN	BODIE	BODMD[1:0]		VBOD[4:0]				
-				*RW*	*RW*		*RW*	*RW*	*RW*		*RW*				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31~12	保留	-
11	LVREN	**LVR使能位** **0：禁止低电压复位功能** **1：允许低电压复位功能** 说明：LVR一旦开启，开启后关闭无效，只有在复位时才能关闭。
10 ~ 9	VLVR[1:0]	**LVR复位电压档位设置位** 00: 4.1 V 01: 3.7 V 10: 2.8 V 11: 2.3 V
8	BODEN	**BOD允许位** 0：禁止掉电检测 1：允许掉电检测
7	BODIE	**BOD中断使能位**
6 ~ 5	BODMD[1:0]	**BOD模式选择控制位** 00：仅当VDD电压从小于BOD门限到大于BOD门限时，BODIF标志置1 01：仅当VDD电压从大于BOD门限到小于BOD门限时，BODIF标志置1 10：当VDD电压从大于门限到小于门限，或从小于门限到大于门限，BODIF标志都置1 11：保留
4 ~ 0	VBOD[4:0]	**BOD门限电压设置位** 00000: 2.40 V 00001: 2.50 V 00010: 2.60 V 00011: 2.70 V 00100: 2.80 V 00101: 2.90 V 00110: 3.00 V 00111: 3.10 V 01000: 3.20 V 01001: 3.30 V 01010: 3.40 V 01011: 3.50 V 01100: 3.60 V 01101: 3.70 V 01110: 3.80 V 01111: 3.90 V 10000: 4.00 V 10001: 4.10 V 10010: 4.20 V 10011: 4.30 V 其他: 2.40V

9.5.2 电源状态寄存器 (SYSCFG_PWRSR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留														BODF	BODIF
-														*RO*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 2	保留	-
1	BODF	**BOD标志位** 0：当前电压高于在VBOD[3:0]中设置的BOD电压 1：当前电压低于在VBOD[3:0]中设置的BOD电压
0	BODIF	**BOD中断请求标志** 0：无中断挂起，清除挂起 1：中断挂起

10. 复位和时钟控制（RCC）

10.1 复位

SH32F9001_32F9801支持两种复位形式，分别为系统复位（SYSRESET）和电源复位（PWRESET）。

注：根据Cortex-M3说明，系统复位为Warm Reset，不会复位Debug电路，电源复位为Cold Reset，会复位Debug电路。

10.1.1 系统复位

除了时钟控制器的RCC_CSR寄存器中的复位标志位外，系统复位将复位所有寄存器至它们的复位状态。 当发生以下任一事件时，产生一个系统复位：

- NRST引脚上的低电平(外部复位)
- 独立看门狗计数终止(IWDT复位)
- 窗口看门狗计数终止(WWDT复位)
- 软件复位(SW复位)
- LOCKUP处理(CPU锁死跳转)

可通过查看RCC_RSTSTR控制状态寄存器中的复位状态标志位识别复位事件来源。

10.1.1.1 引脚复位

SH32F9001_32F9801有一个独立的复位引脚NRST，该引脚上的一段时间低电平将引起系统复位。

10.1.1.2 独立看门狗复位（IWDT）

独立看门狗定时器是一个12位递减计数器，独立内建RC振荡器作为其时钟源，上电默认打开，且不可关闭。

独立看门狗复位后产生IWDTRSTF@RCC_RSTSTR标志。

具体见“独立看门狗”章节介绍。

10.1.1.3 窗口看门狗复位 （WWDT）

窗口看门狗的喂狗时间是一个时间窗口，用来检测外部干扰或不可预测逻辑条件造成的应用程序背离正常运行序列而产生的软件故障。适合在精确计时窗口起作用的应用程序。当过早或过迟刷新窗口看门狗时，将产生复位信号。

窗口看门狗复位后产生WWDTRSTF@RCC_RSTSTR标志。

具体见“窗口看门狗”章节介绍。

10.1.1.4 软件复位

通过将Cortex™-M3中断应用和复位控制寄存器中的SYSRESETREQ位置’1’，可实现软件复位。请参考Cortex™-M3技术参考手册获得进一步信息。

另外，NVIC中的应用程序中断与复位控制寄存器（AIRCR）的VECTRESET位，可以只复位Cortex-M3内核，不复位外设。

10.1.1.5 CPU LOCKUP 锁死处理

发生CPU Lockup时，将PC指针指向FLASH最后一个字节，并等待WDT复位。有关LOCKUP请参考Cortex™-M3技术参考手册获得进一步信息。

10.1.2 电源复位

当以下事件中之一发生时，产生电源复位：

- 上电复位(POR复位)
- 低电压复位（LVR复位）

电源复位将复位所有寄存器。复位入口矢量被固定在地址0x0000_0004。

10.1.2.1 上电复位

SH32F9001_32F9801内部有一个完整的上电复位（POR）电路，当供电电压达到V_{POR}时系统即能正常工作。当VDD低于指定的限位电压V_{POR}时，系统保持为复位状态。

关于上电复位的细节请参考“电气特性”部分。

10.1.2.2 低电压复位

低电压复位（LVR）是一种强制性保护复位，可以通过寄存器LVREN@SYSCFG_PWRCR开启（LVR一旦开启，开启后关闭无效，只有在复位时才能关闭），当供电电压低于V_{LVR}时，MCU将产生复位。

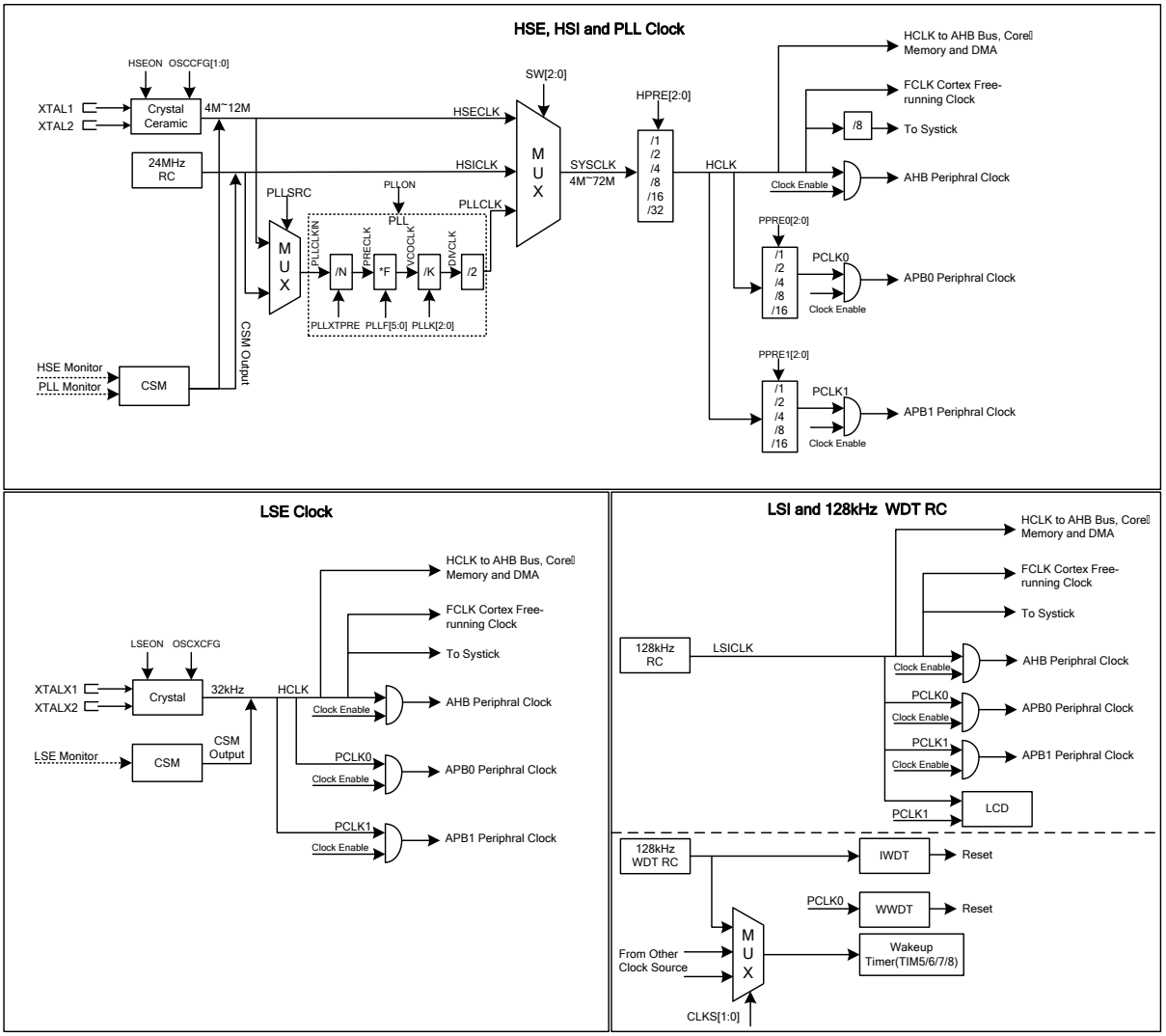
具体见“低电压复位（LVR）”章节介绍。

10.2 时钟

四种不同的时钟源可被用来驱动系统时钟(SYSCLK)：

- HSI振荡器时钟（24MHz）
- HSE振荡器时钟（4MHz～12MHz Crystal/Ceramic）
- LSI振荡器时钟（128kHz）
- LSE振荡器时钟（32kHz Crystal）
- PLL时钟 （MAX：72MHz）

其中PLL时钟可选择HSI和HSE作为时钟源。



SH32F9001_32F9801的时钟结构框图

注：此框图用于说明系统整体的时钟树结构，控制细节请参考本章内容。

用户可通过多个预分频器配置AHB、APB0、APB1域的频率。

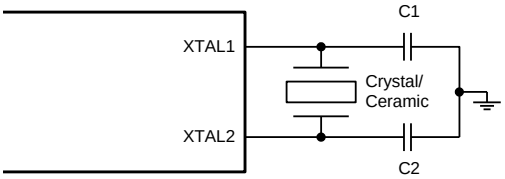
- HCLK：AHB域主时钟，最大频率是72MHz，包括Cortex-M3内核、内存、DMA等都由HCLK驱动。
- PCLK0：APB0域主时钟，最大频率是36MHz，APB0总线上的外设都由PCLK0驱动，建议用户不要设置超过36MHz频率；
- PCLK1：APB1域主时钟，最大频率是36MHz，APB1总线上的外设都由PCLK1驱动，建议用户不要设置超过36MHz频率；

RCC通过AHB时钟(HCLK)8分频后作为SysTick的外部时钟。通过对SysTick控制与状态寄存器的设置，可选择上述时钟或内核时钟作为SysTick时钟源。

10.2.1 HSE 时钟

高速外部时钟信号(HSE)由以下两种时钟源产生：

(1) 外部晶体/陶瓷谐振器：4MHz - 12MHz

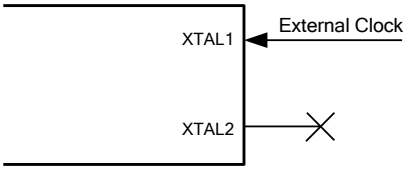


4MHz~12Mz外部振荡器（包含晶体谐振器和陶瓷谐振器）可为系统提供更为精确的主时钟。在时钟控制寄存器HSERDY@RCC_CR位用来指示高速外部振荡器是否稳定。在启动时，直到这一位被硬件置'1'，时钟才被释放出来。如果在时钟中断寄存器RCC_CIENR中允许产生中断，将会产生相应中断。

HSE晶体可以通过设置时钟控制寄存器里HSEON@RCC_CR位被启动和关闭。

为了减少时钟输出的失真和缩短启动稳定时间，晶体/陶瓷谐振器和负载电容器必须尽可能地靠近振荡器引脚。负载电容值必须根据所选择的振荡器来调整。

(2) 外部输入时钟：4MHz – 12MHz



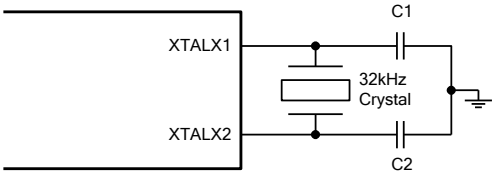
在这个模式里，必须提供外部时钟。它的频率最高可达12MHz。这种模式下用户先要在XTAL1上灌入外部时钟，然后开启HSEON位，此时XTAL2必须悬空。但可以通过设置OSCCFG@SYSCFG_SAFR位为10b来设置XTAL2复用为GPIO口使用。

和外部振荡器一样，外部时钟也存在预热时间和Ready信号。

外部时钟也受CSM监控。

10.2.2 LSE 时钟

低速外部时钟信号(LSE)由外部32kHz晶体谐振器产生：



32kHz外部振荡器可为系统提供精确的低频主时钟。在时钟控制寄存器LSERDY@RCC_CR位用来指示低速外部振荡器是否稳定。在启动时，直到这一位被硬件置'1'，时钟才被释放出来。如果在时钟中断寄存器RCC_CIENR中允许产生中断，将会产生相应中断。

LSE晶体可以通过设置时钟控制寄存器里LSEON@RCC_CR位被启动和关闭。

为了减少时钟输出的失真和缩短启动稳定时间，晶体谐振器和负载电容器必须尽可能地靠近振荡器引脚。负载电容值必须根据所选择的振荡器来调整。

10.2.3 HSI 时钟

HSI时钟信号由内部24MHz的RC振荡器产生，可直接作为系统时钟或作为PLL输入，作为PLL输入时，PLL预分频因子PLLXTPRE@RCC_CFGR要打开，即启动预二分频。

HSI还有如下特性：

- HSI RC振荡器的启动时间比HSE晶体振荡器短。
- 系统复位后，选择HSI作为系统时钟。
- 系统上电后默认选择HSI作为系统时钟。
- HSI RC在出厂时会被校准到0.3%(25°C)，在全温度范围不超过1%，系统复位时，工厂校准值被装载到时钟控制寄存器的HSICAL[7:0]位。 如果用户的应用基于不同的电压或环境温度，这将会影响RC振荡器的精度。可以通过时钟控制寄存器里的HSITRIM[2:0]位来进一步调整HSI频率。
- 如果HSE晶体振荡器失效，HSI时钟会被自动切换为系统时钟，切换频率可通过CSMSEL寄存器配置。

10.2.4 LSI 时钟

LSI时钟信号由内部128kHz的RC振荡器产生，提供一个低功耗时钟源，可直接作为系统时钟，但不可作为PLL输入。

LSI具有如下特性：

- 若128kHz RC作为LCD时钟源，则在停机模式下，仍保持开启。
- LSI不支持RC校准。

10.2.5 WDT 时钟

WDT RC提供一个低功耗时钟源，以用于停机唤醒模块的驱动，包括IWDT和Wakeup Timer。WDT时钟不能用于系统时钟。WDT时钟频率约为128KHz左右（注：128KHz WDT RC 精度为±50%）。

- WDT时钟上电后是常开的。
- 在停机模式下也保持开启。
- WDT时钟不支持校准。
- IWDT使用128KHz WDT RC作为工作时钟，Wakeup Timer（TIM5/TIM6/TIM7/TIM8）可选择128kHz WDT RC时钟作为唤醒定时器时钟源。

10.2.6 PLL 时钟

内部PLL可以用来倍频HSI的内部RC输出时钟或HSE的外部晶体振荡器输出时钟。

PLL的设置必须在其被激活前完成。这些参数包括PLLSRC、PLLXTPRE、PLLF、PLLK，一旦PLL被激活，这些参数就不能被改动。

如果PLL中断在时钟中断寄存器里被允许，当PLL准备就绪时，可产生中断申请。

PLL倍频系数由3个参数组成，分别是N、F和K，计算公式是：

$$PRECLK = \frac{PLLCLKIN}{N} \quad (1)$$

$$VCOCLK = PRECLK \times F \quad (2)$$

$$PLLCLK = \frac{VCOCLK}{K} \times \frac{1}{2} \quad (3)$$

N：由预分频因子PLLXTPRE决定，取值范围1～4；

F：由倍频因子PLLF指定，取值范围15～78；

K：由分频因子PLLK指定，取值范围1～8；

对于HSI时钟源，建议预分频因子N取2或4，以获得理想的50%占空比的时钟输入。

对于HSE时钟源，建议N取1，直接输入。

对PLL模块而言，PLLCLKIN输入时钟范围4MHz～24MHz，PRECLK预分频时钟要求满足4MHz～8MHz范围，VCOCLK倍频输出时钟要求满足150MHz～300MHz范围。PLL的设置除了要满足上述时钟范围要求，也需要满足系统主频范围要求（4MHz～72MHz）。

以设置72MHz系统主频为例，

（1） 选择24MHz HSI作为时钟源，设置PLLXTPRE=3（N=4），PLLF=33（F=48），PLLK=1（K=2），则PRECLK=6MHz，VCOCLK=288MHz，都满足要求，输出PLLCLK为：6×48/2/2=72MHz；

（2） 选择12MHz HSE作为时钟源，设置PLLXTPRE=1（N=2），PLLF=33（F=48），PLLK=1（K=2），则PRECLK=6MHz，VCOCLK=288MHz，都满足要求，输出PLLCLK为：6×48/2/2=72MHz。

10.2.7 系统时钟(SYSCLK)选择

系统复位后，HSI振荡器被选为系统时钟。当时钟源被直接或通过PLL间接作为系统时钟时，它将不能被停止。只有当目标时钟源准备就绪了(经过启动稳定阶段的延迟或PLL稳定)，从一个时钟源到另一个时钟源的切换才允许发生。

在时钟控制寄存器(RCC_CR)里的状态位指示哪个时钟已经准备好了，哪个时钟目前被用作系统时钟。

10.2.8 时钟安全监控（CSM）

为了增强系统的可靠性，防止因时钟失效造成系统出错甚至死机的严重后果，SH32F9001_32F9801增加了一个时钟安全监控（CSM）模块。CSM监控三个源头：

（1） HSE的振荡情况，包含晶振、陶振、或外部时钟输入的情况（开启HSE的情况，即使HSE未作为系统时钟源），一旦发生停振，则送出异常标志HSECSMF，并将时钟自动切换至HSI[*对应HSE频率（通过寄存器RCC->CSMSEL选择）*]；

（2） LSE的振荡情况，一旦32kHz晶振发生停振（开启LSE的情况，即使LSE未作为系统时钟源），则送出异常标志LSECSMF，并将时钟自动切换至内部32kHz RC（由LSI分频得到）；

（3） PLL的锁定情况（使用PLL的情况），一旦发生失锁，则送出异常标志PLLCSMF，并将时钟自动切换至HSI；

用户可以在RCC@CISTR来查询相应的中断标志以确定是何种异常。如果上述异常会影响系统时钟，则会申请CSM中断，该中断连接至NMI不可屏蔽中断。具体系统时钟和异常关系，以及何时产生CSM中断，参见下面的“CSM异常功能定义和处理表”。CSM引起NMI中断由IEN_CSM@SYSCFG.SAFR控制位使能。

CSM上电是默认关闭的，但可以通过CSMON@RCC_CR。CSM开启时只有在LSE Ready、HSE Ready或PLL Lock后监控才有效。

表10-1 CSM 异常功能定义和处理表

系统时钟	HSE异常情况	LSE异常情况	PLL异常情况	CSM标志	CSM处理（CSM中断不可屏蔽）	RCC寄存器变化
HSI	0	0	1	PLLCSMF=1	不做处理	CSMF置起的同时复位相应的RDY位
	0	1	0	LSECSMF=1	LSE时钟切换到LSI（128kHz四分频）	CSMF置起的同时复位相应的RDY位
	0	1	1	PLLCSMF=1 LSECSMF=1	LSE时钟切换到LSI（128kHz四分频）	CSMF置起的同时复位相应的RDY位
	1	0	0	HSECSMF=1	HSE时钟切换到HSI时钟（对应CSM选择档位）	CSMF置起的同时复位相应的RDY位
	1	0	1	HSECSMF=1 PLLCSMF=1	HSE时钟切换到HSI时钟（对应CSM选择档位）	CSMF置起的同时复位相应的RDY位
	1	1	0	HSECSMF=1 LSECSMF=1	（1）HSE时钟切换到HSI时钟（对应CSM选择档位）； （2）LSE时钟切换到LSI（128kHz四分频）	CSMF置起的同时复位相应的RDY位
	1	1	1	HSECSMF=1 LSECSMF=1	（1）HSE时钟切换到HSI时钟（对应CSM选择档	CSMF置起的同时复位相应的RDY位

				PLLCSMF=1	位）； （2）LSE时钟切换到LSI （128kHz四分频）	
HSI+PLL	0	0	1	PLLCSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断；	SW,SWS PLLON,PLLRDY PLLRDYIE PLLRDYIF复位[注1]
	0	1	0	LSECSMF=1	LSE时钟切换到LSI （128kHz四分频）	CSMF置起的同时复位相应的RDY位
	0	1	1	PLLCSMF=1 LSECSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断； （3）LSE时钟切换到LSI （128kHz四分频）	SW,SWS PLLON,PLLRDY PLLRDYIE PLLRDYIF复位[注1]
	1	0	0	HSECSMF=1	不做处理	CSMF置起的同时复位相应的RDY位
	1	0	1	HSECSMF=1 PLLCSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断； （3）HSE时钟切换到HSI时钟（对应CSM选择档位）；	SW,SWS PLLON,PLLRDY PLLRDYIE PLLRDYIF复位[注1]
	1	1	0	HSECSMF=1 LSECSMF=1	（1）HSE时钟切换到HSI时钟（对应CSM选择档位）； （2）LSE时钟切换到LSI （128kHz四分频）	CSMF置起的同时复位相应的RDY位
	1	1	1	HSECSMF=1 LSECSMF=1 PLLCSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断； （3）HSE时钟切换到HSI时钟（对应CSM选择档位）； （4）LSE时钟切换到LSI （128kHz四分频）	SW,SWS PLLON,PLLRDY PLLRDYIE PLLRDYIF复位[注1]
HSE	0	0	1	PLLCSMF=1	不做处理	CSMF置起的同时复位相应的RDY位
	0	1	0	LSECSMF=1	LSE时钟切换到LSI （128kHz四分频）	CSMF置起的同时复位相应的RDY位
	0	1	1	PLLCSMF=1 LSECSMF=1	LSE时钟切换到LSI （128kHz四分频）	CSMF置起的同时复位相应的RDY位
	1	0	0	HSECSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断；	SW,SWS HSEON,HSERDY HSERDYIE HSERDYIF复位
	1	0	1	HSECSMF=1 PLLCSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断；	SW,SWS HSEON,HSERDY HSERDYIE HSERDYIF复位
	1	1	0	HSECSMF=1 LSECSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断； （3）LSE时钟切换到LSI （128kHz四分频）	SW,SWS HSEON,HSERDY HSERDYIE HSERDYIF复位

	1	1	1	HSECSMF=1 LSECSMF=1 PLLCSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断； （3）LSE时钟切换到LSI（128kHz四分频）	SW,SWS HSEON,HSERDY HSERDYIE HSERDYIF复位
HSE+PLL	0	0	1	PLLCSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断；	SW,SWS PLLSRC[注2],HSEON HSERDY,HSERDYIE HSERDYIF复位
	0	1	0	LSECSMF=1	LSE时钟切换到LSI（128kHz四分频）	CSMF置起的同时复位相应的RDY位
	0	1	1	PLLCSMF=1 LSECSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断； （3）LSE时钟切换到LSI（128kHz四分频）	SW,SWS PLLSRC[注2],HSEON HSERDY,HSERDYIE HSERDYIF复位
	1	0	0	HSECSMF=1	（1）系统时钟切换到HSI，且硬件自动切到PLL；[注3] （2）申请CSM中断；	SW,SWS PLLSRC[注2],HSEON HSERDY,HSERDYIE HSERDYIF复位
	1	0	1	HSECSMF=1 PLLCSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断；	SW,SWS PLLSRC,PLLON PLLRDY,HSEON HSERDY,PLLRDYIE PLLRDYIF,HSERDYIE HSERDYIF复位
	1	1	0	HSECSMF=1 LSECSMF=1	（1）系统时钟切换到HSI，且硬件自动切到PLL；[注3] （2）申请CSM中断； （3）LSE时钟切换到LSI（128kHz四分频）	SW,SWS PLLSRC[注2],HSEON HSERDY,HSERDYIE HSERDYIF复位
	1	1	1	HSECSMF=1 LSECSMF=1 PLLCSMF=1	（1）系统时钟切换到HSI； （2）申请CSM中断； （3）LSE时钟切换到LSI（128kHz四分频）	SW,SWS PLLSRC,PLLON PLLRDY,HSEON HSERDY,PLLRDYIE PLLRDYIF,HSERDYIE HSERDYIF复位
LSI	0	0	1	PLLCSMF=1	不做处理	CSMF置起的同时复位相应的RDY位
	0	1	0	LSECSMF=1	LSE时钟切换到LSI（128kHz四分频）	CSMF置起的同时复位相应的RDY位
	0	1	1	PLLCSMF=1 LSECSMF=1	LSE时钟切换到LSI（128kHz四分频）	CSMF置起的同时复位相应的RDY位
	1	0	0	HSECSMF=1	HSE时钟切换到HSI时钟（对应CSM选择档位）；	CSMF置起的同时复位相应的RDY位
	1	0	1	HSECSMF=1 PLLCSMF=1	HSE时钟切换到HSI时钟（对应CSM选择档位）；	CSMF置起的同时复位相应的RDY位
	1	1	0	HSECSMF=1 LSECSMF=1	（1）HSE时钟切换到HSI时钟（对应CSM选择档位）； （2）LSE时钟切换到LSI	CSMF置起的同时复位相应的RDY位

					（128kHz四分频）	
	1	1	1	HSECSMF=1 LSECSMF=1 PLLCSMF=1	（1）HSE时钟切换到HSI时钟（对应CSM选择档位）； （2）LSE时钟切换到LSI（128kHz四分频）	CSMF置起的同时复位相应的RDY位
LSE	0	0	1	PLLCSMF=1	不做处理	CSMF置起的同时复位相应的RDY位
	0	1	0	LSECSMF=1	（1）系统时钟切换到LSI（128kHz四分频）； （2）申请CSM中断；	SW,SWS LSEON,LSERDY LSERDYIE LSERDYIF复位
	0	1	1	PLLCSMF=1 LSECSMF=1	（1）系统时钟切换到LSI（128kHz四分频）； （2）申请CSM中断；	SW,SWS LSEON,LSERDY LSERDYIE LSERDYIF复位
	1	0	0	HSECSMF=1	HSE时钟切换到HSI时钟（对应CSM选择档位）	CSMF置起的同时复位相应的RDY位
	1	0	1	HSECSMF=1 PLLCSMF=1	HSE时钟切换到HSI时钟（对应CSM选择档位）	CSMF置起的同时复位相应的RDY位
	1	1	0	HSECSMF=1 LSECSMF=1	（1）系统时钟切换到LSI（128kHz四分频）； （2）申请CSM中断； （3）HSE时钟切换到HSI时钟（对应CSM选择档位）	SW,SWS LSEON,LSERDY LSERDYIE LSERDYIF复位
	1	1	1	HSECSMF=1 LSECSMF=1 PLLCSMF=1	（1）系统时钟切换到LSI（128kHz四分频）； （2）申请CSM中断； （3）HSE时钟切换到HSI时钟（对应CSM选择档位）	SW,SWS LSEON,LSERDY LSERDYIE LSERDYIF复位

注1：执行CSM处理时清除相应的RDY位，并关闭该时钟源（ON位），SW和SWS控制位随着系统时钟切换到HSI而同时复位；

注2：PLLSRC=1时无法关闭HSEON，因此关闭HSEON要求同时把PLLSRC复位；

注3：当HSE停振且系统时钟为HSE+PLL时，硬件自动切换系统时钟为对应CSM选择档位HSI+PLL；

从上表中可以看出，即使HSE/LSE没有直接或间接作为系统时钟，一旦HSE/LSE停振，CSM也会置位异常标志，由IEN_CSM@SYSCFG.SAFR控制是否产生CSM中断。同样的，即使PLL未用作系统时钟，一旦PLL失锁，也会置位标志，由IEN_CSM@SYSCFG.SAFR控制是否产生CSM中断。

如果HSE关闭、LSE关闭或PLL关闭，则CSM检测相应模块失效的功能自动关闭。

注意：一旦CSM中断产生，引起NMI被不断执行，直到CSMF挂起位被清除，或是CSMON@RCC_CR或是IEN_CSM@SYSCFG.SAFR为0。因此，在NMI的处理程序中必须通过设置时钟中断寄存器RCC@CICLR中的CSMC位来清除CSM中断（将会同时清除3个异常标志）。

下面代码演示了时钟失效与恢复的过程。

HSE时钟失效时，产生NMI中断。

```
Void  NMIException(void)
{
    if (RCC_GetITStatus(RCC_IT_CSM) != RESET)    //产生了时钟失效中断
{
/*******************************************************************/
RCC_ITConfig(IEN_CSM@SYSCFG.SAFR, DISABLE);  // DISABLE IE_CSM
RCC_PLLConfig(RCC_PLL_OFF);       //手动关闭PLL

RCC_HSEConfig(RCC_HSE_OFF);      // 手动关闭HSE
RCC_HSEConfig(RCC_SW, HSI);       // 手动切换

/*******************************************************************/
            ……..       // 用户可以在此添加应用相关的保护代码
/*    下面为HSE恢复做准备，一旦HSE恢复将产生对应中断*/
/*******************************************************************/
RCC_HSEConfig(RCC_HSE_ON);      // 再次手动打开HSE
RCC_ClearITPendingBit(RCC_IT_CSM);      // 清除中断挂起位
/*******************************************************************/
```

10.2.9 系统节拍定时器（SysTick）

SysTick是一个24位倒计数定时器，在Cortex-M3内核中实现。可提供一个固定间隔的时间节拍，在主流程中可作为监控或定时使用。因为所有Cortex-M3芯片都有SysTick，有助于操作系统或程序进行跨平台移植。

SysTick可使用内核时钟，也可使用外部参考时钟，其节拍周期可由“重装载值寄存器”设定。具体寄存器定义可参考“Cortex-M3技术参考手册”。

SH32F9001_32F9801给SysTick提供了外部参考时钟，当系统时钟为高频振荡器时，SysTick时钟源固定为HCLK/8，其节拍标准值固定为90000（存放在系统定时器校准值寄存器STCALIB中），即在HCLK为72MHz频率下，产生10ms的时间基准；当系统时钟为32kHz振荡器时，SysTick时钟源固定为HCLK，其节拍标准值固定为320（存放在系统定时器校准值寄存器STCALIB中），即在HCLK为32kHz频率下，产生10ms的时间基准；

10.2.10 HSI 修整功能（HSITRIM）

SH32F9001_32F9801出厂时HSI已校准（定义为HSICAL）到一个很高的精度，而在运行时由于电压和温度影响，有必要对HSI进行进一步修整（定义为HSITRIM），RCC提供了3bit修整控制位HSITRIM[2:0]。

SH32F9001_32F9801内部也提供了一种使用外部晶振对HSI进行修整的特殊功能，该功能由TRIMRUN@RCC_HSICAL开启，方法是使用HSI去计量4096分频后的HSE时钟，结果存放在TRIMREF[12:0]@RCC_HSICAL中，由用户根据该参数调整HSITRIM，从而达到提升HSI精度、补偿电压和温度变化的目的。

使用内部修整时，要求用户预先开启HSE并等待稳定，系统内部不对此检查，计量完成后，TRIMRUN控制位恢复为0，否则保持为1。

如HSE使用8M晶振，则标准情况下TRIMREF将得到4095（表示4096个时钟）。

如果使用PLL，做内部修整时系统仍然可以工作于PLL下，但计量完成对HSI进行修整一般要求重开PLL，以避免HSI的抖动导致PLL失步。

10.3 寄存器

RCC 模块寄存器列表 (基地址:0x4004 3C00)

地址	寄存器名	说明
0x4004 3C00	CR	时钟控制寄存器
0x4004 3C04	CFGR	时钟配置寄存器
0x4004 3C08	CIENR	时钟中断使能寄存器
0x4004 3C0C	CISTR	时钟中断状态寄存器
0x4004 3C10	CICLR	时钟中断标志清除寄存器
0x4004 3C14	AHBRSTR	AHB外设复位寄存器
0x4004 3C18	APB1RSTR	APB1外设复位寄存器
0x4004 3C1C	APB0RSTR	APB0外设复位寄存器
0x4004 3C20	AHBENR	AHB外设时钟使能寄存器
0x4004 3C24	APB1ENR	APB1外设时钟使能寄存器
0x4004 3C28	APB0ENR	APB0外设时钟使能寄存器
0x4004 3C2C	RSTSTR	RESET状态寄存器
0x4004 3C30	RSTCLR	清除RESET状态寄存器
0x4004 3C34	HSICAL	HSI振荡器校准寄存器
0x4004 3C38	CSMSEL	CSM档位选择
0x4004 3C3C	RCCLOCK	RCC配置锁定寄存器
0x4004 3C44	HSERFB	外部晶振/陶振反馈电阻选择寄存器

10.3.1 时钟控制寄存器 (RCC_CR)

偏移地址: 0x0000

复位值: 0x0000 2000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															LSIRDY
-															*RO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LSION	HSIRDY	HSION	LSERDY	LSEON	HSERDY	HSEON	PLLRDY	PLLON	CSMON	SWS[2:0]			SW[2:0]		
RW	*RO*	*RW*	*RO*	*RW*	*RO*	*RW*	*RO*	*RW*	*RW*	*RO*			*RW*		
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 17	保留	-
16	LSIRDY	**LSI时钟就绪标志** LSI振荡稳定后由硬件置1 0：LSI振荡未稳定 1：LSI振荡稳定
15	LSION	**LSI使能 (LSI enable)** 由软件置1或清零以开启或关闭LSI 当LSI时钟被用作系统时钟时，该位不能被清零 0：LSI关闭 1：LSI使能

14	**HSIRDY**	**HSI时钟就绪标志** HSI振荡稳定后由硬件置1 0：HSI振荡未稳定 1：HSI振荡稳定
13	**HSION**	**HSI使能 (HSI enable)** 由软件置1或清零以开启或关闭HSI 当HSI时钟被用作系统时钟时，该位不能被清零 0：HSI关闭 1：HSI使能
12	**LSERDY**	**外部32kHz振荡器就绪标志** 由硬件置1来指示外部32kHz振荡器已经稳定 在LSEON位清零后，该位需要6个外部32kHz时钟周期清零 0：外部32kHz时钟没有就绪 1：外部32kHz时钟就绪 说明：该位依赖于LSEON，仅指示外部32kHz振荡器就绪状态。
11	**LSEON**	**外部32kHz振荡器使能** 由软件置1或清0 当LSE被直接或间接用作系统时钟时，该位不能被清零 0：LSE振荡器关闭 1：LSE振荡器开启
10	**HSERDY**	**外部高频振荡器就绪标志** 由硬件置1来指示外部高频振荡器已经稳定 在HSEON位清零后，该位需要6个外部4-12MHz时钟周期清零 0：外部4-12MHz时钟没有就绪 1：外部4-12MHz时钟就绪 说明：该位依赖于HSEON，仅指示外部高频振荡器就绪状态。
9	**HSEON**	**外部高频振荡器使能** 由软件置1或清0 当HSE被直接或间接用作系统时钟时，该位不能被清零 0：HSE振荡器关闭 1：HSE振荡器开启
8	**PLLRDY**	**PLL时钟就绪标志** PLL锁定后由硬件置1 0：PLL未锁定 1：PLL锁定
7	**PLLON**	**PLL使能 (PLL enable)** 由软件置1或清零以开启或关闭PLL 当PLL时钟被用作系统时钟时，该位不能被清零 0：PLL关闭 1：PLL使能
6	**CSMON**	**CSM时钟检测功能使能** 0：CSM时钟检测功能关闭 1：CSM时钟检测功能开启
5 ~ 3	**SWS[2:0]**	**系统时钟切换状态 (System clock switch status)** 由硬件置1或清0来指示系统时钟源。 000：当前系统时钟为HSI 001：当前系统时钟为PLL输出

		010：当前系统时钟为LSE 011：当前系统时钟为HSE 100：当前系统时钟为LSI
2 ~ 0	**SW[2:0]**	**系统时钟切换 (System clock switch)** 由软件置1或清0来选择系统时钟源。 000：HSI作为系统时钟 001：PLL输出作为系统时钟 010：LSE作为系统时钟 011：HSE作为系统时钟 100：LSI作为系统时钟

10.3.2 时钟配置寄存器 (RCC_CFGR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留					HSESEL[1:0]		HSIENREC	CRY32NOISEEN	CRY32SPDUP	PLLXTPRE[1:0]		PLLSRC	保留	PLLF[4:0]	
-					RW		RW	RW	RW	RW		RW	-	RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PLLF[5:0]			PLLK[2:0]			PLLSW	PPRE1[2:0]			PPRE0[2:0]			HPRE[2:0]		
RW			RW			RW	RW			RW			RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 27	保留	-
26 ~ 25	**HSESEL[1:0]**	**HSE驱动能力选择位(推荐选用Normal档)** 00：Normal 01：保留 10：保留 11：Strong
24	**HSIENREC**	**HSI死态电路控制开关位**
23	**CRY32NOISEEN**	**LSE振荡器扰动使能信号** 0：Disable 1：Enable
22	**CRY32SPDUP**	**32.768kHz晶体谐振器加速模式控制位** 0：32.768kHz振荡器常规模式，由软件清0。 1：32.768kHz振荡器加速模式，由软件或者硬件置1。 此位在系统发生任何形式的复位，如上电复位，看门狗复位等时，自动由硬件设置1，用以加速32.768kHz振荡器起振，缩短32.768kHz振荡器的起振时间。 如果有需要，本位也可以由软件置1或者清0。比如进入掉电模式（stop）前，可以将此位置1，掉电模式唤醒后再由软件清0。 应该注意的是关闭32.768kHz加速模式（此位清0），可以节省系统的耗电。只有LSE相关开启时才有效。
21 ~ 20	**PLLXTPRE[1:0]**	**PLL时钟输入预分频 (Pre-divider for PLL entry)** 由软件设置分频后作为PLL输入时钟。

		只能在关闭PLL时写入此位。 00：不预分频 01：2分频 10：3分频 11：4分频
19	**PLLSRC**	**PLL输入时钟源 (PLL entry clock source)** 由软件置1或清0来选择PLL输入时钟源。 只能在关闭PLL时才能写入此位。 0：HSI时钟作为PLL输入时钟 1：HSE时钟作为PLL输入时钟
18 ~ 13	**PLLF[5:0]**	**PLL倍频因子F (PLL multiplication factor F)** 由软件设置来确定PLL倍频参数 只有在PLL关闭的情况下才可被写入。 000000：F=15 000001：F=16 000010：F=17 ... ... 111101：F=76 111110：F=77 111111：F=78
12 ~ 10	**PLLK[2:0]**	**PLL倍频因子K (PLL multiplication factor K)** 由软件设置来确定PLL倍频参数 只有在PLL关闭的情况下才可被写入。 000：K=1 001：K=2 010：K=3 011：K=4 100：K=5 101：K=6 110：K=7 111：K=8
9	**PLLSW**	**PLL自动重启使能位** 只能在关闭PLL时才能写入此位；只在系统时钟为HSE+PLL时有效。 0：HSI时钟作为系统时钟 1：HSI时钟作为PLL输入时钟，待PLL稳定后硬件自动将PLL输出作为系统时钟 当PLLSW = 0时，若HSE停振，则系统时钟切换到CSM选择的对应档位HSI时钟，PLLON、PLLRDY等相关标志位硬件清零； 当PLLSW = 1时，若HSE停振，则系统时钟切换到CSM选择的对应档位HSI时钟，且硬件自动重启PLL时钟，待PLL RDY后将系统时钟由硬件自动切换到PLL；
8 ~ 6	**PPRE1[2:0]**	**APB1预分频** 由软件置1或清0来控制低速APB1时钟(PCLK1)的预分频系数。 000：PCLK1不分频 001：PCLK1 2分频 010：PCLK1 4分频 011：PCLK1 8分频 100：PCLK1 16分频

		其他：PCLK1 16分频 警告：软件必须保证APB1时钟频率不超过36MHz。 注1：系统时钟选择LSI或LSE时，APB1时钟不分频，预分频系数无效。
5 ~ 3	**PPRE0[2:0]**	**APB0预分频** 由软件置1或清0来控制低速APB1时钟(PCLK1)的预分频系数。 000：PCLK0不分频 001：PCLK0 2分频 010：PCLK0 4分频 011：PCLK0 8分频 100：PCLK0 16分频 其他：PCLK0 16分频 警告：软件必须保证APB0时钟频率不超过36MHz。 注1：系统时钟选择LSI或LSE时，APB0时钟不分频，预分频系数无效。
2 ~ 0	**HPRE[2:0]**	**AHB预分频 (AHB Prescaler)** 由软件置1或清0来控制AHB时钟的预分频系数。 000：SYSCLK不分频 001：SYSCLK 2分频 010：SYSCLK 4分频 011：SYSCLK 8分频 100：SYSCLK 16分频 101：SYSCLK 32分频 其他：SYSCLK 32分频 注1：系统时钟选择LSI或LSE时，AHB时钟不分频，预分频系数无效。

10.3.3 时钟中断使能寄存器 (RCC_CIENR)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留											PLLRDYIE	HSERDYIE	LSERDYIE	保留	
-											*RW*	*RW*	*RW*	-	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 5	保留	-
4	**PLLRDYIE**	**PLL就绪中断使能 (PLL ready interrupt enable)** 由软件置1或清0来使能或关闭PLL就绪中断。 0：PLL就绪中断关闭； 1：PLL就绪中断使能。
3	**HSERDYIE**	**HSE就绪中断使能 (HSE ready interrupt enable)** 由软件置1或清0来使能或关闭外部4-12MHz振荡器就绪中断。 0：HSE就绪中断关闭；

		1：HSE就绪中断使能。
2	**LSERDYIE**	**LSE就绪中断使能 (HSE ready interrupt enable)** 由软件置1或清0来使能或关闭外部32kHz振荡器就绪中断。 0：LSE就绪中断关闭； 1：LSE就绪中断使能。
1 ~ 0	**保留**	**-**

10.3.4 时钟中断状态寄存器 (RCC_CISTR)

偏移地址: 0x000C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								CSM PLL F	CSM HSE F	CSM LSE F	PLL RDYI F	HSE RDYI F	LSE RDYI F	保留	
-								*RO*	*RO*	*RO*	*RO*	*RO*	*RO*	-	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	**保留**	**-**
7	**CSMPLLF**	**PLL异常中断标志 (PLL clock security monitor interrupt flag)** PLL失锁时有效，由硬件置1。 由软件通过置1 CSMC位来清除。 0：无PLL异常导致的CSM中断； 1：有PLL异常导致的CSM中断；
6	**CSMHSEF**	**HSE异常中断标志 HSE clock security monitor interrupt flag)** HSE失效时有效，由硬件置1。 由软件通过置1 CSMC位来清除。 0：无HSE失效导致的CSM中断； 1：有HSE失效导致的CSM中断；
5	**CSMLSEF**	**LSE异常中断标志 LSE clock security monitor interrupt flag)** LSE失效时有效，由硬件置1。 由软件通过置1 CSMC位来清除。 0：无LSE失效导致的CSM中断； 1：有LSE失效导致的CSM中断；
4	**PLLRDYIF**	**PLL就绪中断标志 (PLL ready interrupt flag)** 由硬件置1。 由软件通过置1 PLLRDYC位来清除。 0：无PLL上锁产生的时钟就绪中断； 1：PLL上锁导致时钟就绪中断。
3	**HSERDYIF**	**HSE就绪中断标志 (HSE ready interrupt flag)** 由硬件置1。 由软件通过置1 HSERDYC位来清除。 0：无外部4-12MHz振荡器产生的时钟就绪中断； 1：外部4-12MHz振荡器导致时钟就绪中断。
2	**LSERDYIF**	**LSE就绪中断标志 (LSE ready interrupt flag)** 由硬件置1。 由软件通过置1 LSERDYC位来清除。 0：无外部32kHz振荡器产生的时钟就绪中断； 1：外部32kHz振荡器导致时钟就绪中断。

1 ~ 0	保留	-

10.3.5 时钟中断标志清除寄存器 (RCC_CICLR)

偏移地址: 0x0010

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								CSMC	保留		PLLRDYC	HSERDYC	LSERDYC	保留	
-								WO	-		WO	WO	WO	-	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7	CSMC	**清除CSM中断 (Clock security monitor interrupt clear)** 由软件置1来清除CSM中断标志位CSMIF。 0：无作用； 1：清除CSMPLLF和CSMHSEF中断标志位。
6 ~ 5	保留	-
4	PLLRDYC	**清除PLL就绪中断 (PLL ready interrupt clear)** 由软件置1来清除PLL就绪中断标志位PLLRDYF。 0：无作用； 1：清除PLL就绪中断标志位PLLRDYF。
3	HSERDYC	**清除HSE就绪中断 (HSE ready interrupt clear)** 由软件置1来清除HSE就绪中断标志位HSERDYF。 0：无作用； 1：清除HSE就绪中断标志位HSERDYF。
2	LSERDYC	**清除LSE就绪中断 (LSE ready interrupt clear)** 由软件置1来清除LSE就绪中断标志位LSERDYF。 0：无作用； 1：清除HSE就绪中断标志位LSERDYF。
1 ~ 0	保留	-

10.3.6 AHB 外设复位寄存器 (RCC_AHBRSTR)

偏移地址: 0x0014

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留							NTESTRST	RAMBISTRST	CRCRST	DMARST	SYSCFGRST	保留	GPIORST	PCA1RST	PCA0RST
-							*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	-	*RW1s*	*RW1s*	*RW1s*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 9	保留	-
8	NTESTRST	**NTEST模块复位 (NTEST module reset)** 由软件置1，硬件清0 0：无影响 1：NTEST模块复位
7	RAMBISTRST	**RAMBIST模块复位 (RAMBIST module reset)** 由软件置1，硬件清0 0：无影响 1：RAMBIST模块复位
6	CRCRST	**CRC模块复位 (CRC module reset)** 由软件置1，硬件清0 0：无影响 1：CRC模块复位
5	DMARST	**DMA模块复位 (DMA reset)** 由软件置1，硬件清0 0：无影响 1：DMA模块复位
4	SYSCFGRST	**系统配置模块复位 (SYSCFG reset)** 由软件置1，硬件清0 0：无影响 1：系统配置模块复位 注：无法复位此模块内的one-time寄存器。
3	保留	-
2	GPIORST	**通用I/O端口复位 (GPIO block reset)** 由软件置1，硬件清0 0：无影响 1：通用I/O端口复位
1	PCA1RST	**PCA1复位 (Programmable Counter Array 1 reset)** 由软件置1，硬件清0 0：无影响 1：PCA1复位
0	PCA0RST	**PCA0复位 (Programmable Counter Array 0 reset)**

		由软件置1，硬件清0 0：无影响 1：PCA0复位

10.3.7 APB1 外设复位寄存器 (RCC_APB1RSTR)

偏移地址: 0x0018

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留		LEDRST	LCDRST	ADCRST	AMOCRST	TWI1RST	TWI0RST	SPI1RST	SPI0RST	UART5RST	UART4RST	UART3RST	UART2RST	UART1RST	UART0RST
-		*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 14	保留	-
13	LEDRST	**LED复位 LED Driver reset)** 由软件置1，硬件清0 0：无影响 1：LED复位
12	LCDRST	**LCD复位 (LCD Driver reset)** 由软件置1，硬件清0 0：无影响 1：LCD复位
11	ADCRST	**ADC复位 (ADC reset)** 由软件置1，硬件清0 0：无影响 1：ADC复位
10	AMOCRST	**片上模拟功能模块复位(Analog module on-chip reset)** 由软件置1，硬件清0 0：无影响 1：片上模拟功能模块复位
9	TWI1RST	**TWI1复位 (TWI 1 reset)** 由软件置1，硬件清0 0：无影响 1：TWI1复位
8	TWI0RST	**TWI0复位 (TWI 0 reset)** 由软件置1，硬件清0 0：无影响 1：TWI0复位
7	SPI1RST	**SPI1复位 (SPI 1 reset)** 由软件置1，硬件清0 0：无影响

		1：SPI1复位
6	**SPI0RST**	**SPI0复位 (SPI 0 reset)** 由软件置1，硬件清0 0：无影响 1：SPI0复位
5	**UART5RST**	**UART5复位 (UART 5 reset)** 由软件置1，硬件清0 0：无影响 1：UART5复位
4	**UART4RST**	**UART4复位 (UART 4 reset)** 由软件置1，硬件清0 0：无影响 1：UART4复位
3	**UART3RST**	**UART3复位 (UART 3 reset)** 由软件置1，硬件清0 0：无影响 1：UART3复位
2	**UART2RST**	**UART2复位 (UART 2 reset)** 由软件置1，硬件清0 0：无影响 1：UART2复位
1	**UART1RST**	**UART1复位 (UART 1 reset)** 由软件置1，硬件清0 0：无影响 1：UART1复位
0	**UART0RST**	**UART0复位 (UART 0 reset)** 由软件置1，硬件清0 0：无影响 1：UART0复位

10.3.8 APB0 外设复位寄存器 (RCC_APB0RSTR)

偏移地址: 0x001C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WWDTRST	EXTIRST	PCA7RST	PCA6RST	PCA5RST	PCA4RST	PCA3RST	PCA2RST	PWM3RST	PWM2RST	PWM1RST	PWM0RST	TIM8RST	TIM7RST	TIM6RST	TIM5RST
RW1s	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*	*RW1s*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15	WWDTRST	**窗口看门狗复位(Window watchdog reset)** 由软件置1，硬件清0 0：无影响 1：窗口看门狗复位
14	EXTIRST	**EXTI复位 (External Interrupt reset)** 由软件置1，硬件清0 0：无影响 1：EXTI复位
13	PCA7RST	**PCA7复位 (Programmable Counter Array 7 reset)** 由软件置1，硬件清0 0：无影响 1：PCA7复位
12	PCA6RST	**PCA6复位 (Programmable Counter Array 6 reset)** 由软件置1，硬件清0 0：无影响 1：PCA6复位
11	PCA5RST	**PCA5复位 (Programmable Counter Array 5 reset)** 由软件置1，硬件清0 0：无影响 1：PCA5复位
10	PCA4RST	**PCA4复位 (Programmable Counter Array 4 reset)** 由软件置1，硬件清0 0：无影响 1：PCA4复位
9	PCA3RST	**PCA3复位 (Programmable Counter Array 3 reset)** 由软件置1，硬件清0 0：无影响 1：PCA3复位
8	PCA2RST	**PCA2复位 (Programmable Counter Array 2 reset)** 由软件置1，硬件清0 0：无影响 1：PCA2复位
7	PWM3RST	**PWM3复位 (Pulse Width Modulation 3 reset)** 由软件置1，硬件清0 0：无影响 1：PWM3复位
6	PWM2RST	**PWM2复位 (Pulse Width Modulation 2 reset)** 由软件置1，硬件清0 0：无影响 1：PWM2复位
5	PWM1RST	**PWM1复位 (Pulse Width Modulation 1 reset)** 由软件置1，硬件清0 0：无影响

		1：PWM1复位
4	**PWM0RST**	**PWM0复位 (Pulse Width Modulation 0 reset)** 由软件置1，硬件清0 0：无影响 1：PWM0复位
3	**TIM8RST**	**定时器8复位 (Timer 8 reset)** 由软件置1，硬件清0 0：无影响 1：定时器8复位
2	**TIM7RST**	**定时器7复位 (Timer 7 reset)** 由软件置1，硬件清0 0：无影响 1：定时器7复位
1	**TIM6RST**	**定时器6复位 (Timer 6 reset)** 由软件置1，硬件清0 0：无影响 1：定时器6复位
0	**TIM5RST**	**定时器5复位 (Timer 5 reset)** 由软件置1，硬件清0 0：无影响 1：定时器5复位

10.3.9 AHB 外设时钟使能寄存器 (RCC_AHBENR)

偏移地址: 0x0020

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留							NTESTEN	RAMBISTEN	CRCEN	DMAEN	SYSCFGEN	保留	IOCLKEN	PCA1EN	PCA0EN
-							*RW*	*RW*	*RW*	*RW*	*RW*	-	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 9	保留	-
8	**NTESTEN**	**NTEST模块时钟使能 (NTEST clock enable)** 由软件置1或清0 0：NTEST模块时钟关闭 1：NTEST模块时钟开启
7	**RAMBISTEN**	**RAMBIST模块时钟使能 (RAMBIST clock enable)** 由软件置1或清0 0：RAMBIST模块时钟关闭 1：RAMBIST模块时钟开启
6	**CRCEN**	**CRC模块时钟使能 (CRC clock enable)**

		由软件置1或清0 0：CRC模块时钟关闭 1：CRC模块时钟开启
5	**DMAEN**	**DMA模块时钟使能 (DMA clock enable)** 由软件置1或清0 0：DMA模块时钟关闭 1：DMA模块时钟开启
4	**SYSCFGEN**	**系统配置模块时钟使能 (SYSCFG clock enable)** 由软件置1或清0 0：系统配置模块时钟关闭 1：系统配置模块时钟开启 注：无法复位此模块内的one-time寄存器。
3	**保留**	**-**
2	**IOCLKEN**	**通用I/O端口配置区时钟使能 (GPIO configure block clock enable)** 由软件置1或清0 0：通用I/O端口配置区时钟关闭 1：通用I/O端口配置区时钟开启
1	**PCA1EN**	**PCA1时钟使能 (PCA 1 clock enable)** 由软件置1或清0 0：PCA1时钟关闭 1：PCA1时钟开启
0	**PCA0EN**	**PCA0时钟使能 (PCA 0 clock enable)** 由软件置1或清0 0：PCA0时钟关闭 1：PCA0时钟开启

10.3.10 APB1 外设时钟使能寄存器 (RCC_APB1ENR)

偏移地址: 0x0024

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留		LEDEN	LCDEN	ADCEN	AMOCEN	TWI1EN	TWI0EN	SPI1EN	SPI0EN	UART5EN	UART4EN	UART3EN	UART2EN	UART1EN	UART0EN
-		*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 14	保留	-
13	**LEDEN**	**LED时钟使能 (LED clock enable)** 由软件置1或清0 0：LED时钟关闭 1：LED时钟开启
12	**LCDEN**	**LCD时钟使能 (LCD clock enable)**

		由软件置1或清0 0：LCD时钟关闭 1：LCD时钟开启
11	**ADCEN**	**ADC时钟使能 (ADC clock enable)** 由软件置1或清0 0：ADC时钟关闭 1：ADC时钟开启
10	**AMOCEN**	**片上模拟功能模块时钟使能 (Analog module on-chip clock enable)** 由软件置1或清0 0：片上模拟功能模块时钟关闭 1：片上模拟功能模块时钟开启
9	**TWI1EN**	**TWI1时钟使能 (TWI1 clock enable)** 由软件置1或清0 0：TWI1时钟关闭 1：TWI1时钟开启
8	**TWI0EN**	**TWI0时钟使能 (TWI0 clock enable)** 由软件置1或清0 0：TWI0时钟关闭 1：TWI0时钟开启
7	**SPI1EN**	**SPI1时钟使能 (SPI1 clock enable)** 由软件置1或清0 0：SPI1时钟关闭 1：SPI1时钟开启
6	**SPI0EN**	**SPI0时钟使能 (SPI0 clock enable)** 由软件置1或清0 0：SPI0时钟关闭 1：SPI0时钟开启
5	**UART5EN**	**UART5时钟使能 (UART5 clock enable)** 由软件置1或清0 0：UART5时钟关闭 1：UART5时钟开启
4	**UART4EN**	**UART4时钟使能 (UART4 clock enable)** 由软件置1或清0 0：UART4时钟关闭 1：UART4时钟开启
3	**UART3EN**	**UART3时钟使能 (UART3 clock enable)** 由软件置1或清0 0：UART3时钟关闭 1：UART3时钟开启
2	**UART2EN**	**UART2时钟使能 (UART2 clock enable)** 由软件置1或清0 0：UART2时钟关闭 1：UART2时钟开启
1	**UART1EN**	**UART1时钟使能 (UART1 clock enable)** 由软件置1或清0 0：UART1时钟关闭

		1：UART1时钟开启
0	**UART0EN**	**UART0时钟使能 (UART0 clock enable)** 由软件置1或清0 0：UART0时钟关闭 1：UART0时钟开启

10.3.11 APB0 外设时钟使能寄存器 (RCC_APB0ENR)

偏移地址: 0x0028

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WWDTEN	EXTIEN	PCA7EN	PCA6EN	PCA5EN	PCA4EN	PCA3EN	PCA2EN	PWM3EN	PWM2EN	PWM1EN	PWM0EN	TIM8EN	TIM7EN	TIM6EN	TIM5EN
RW	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15	**WWDTEN**	**窗口看门狗时钟使能 (Window watchdog clock enable)** 由软件置1或清0 0：窗口看门狗时钟关闭 1：窗口看门狗时钟开启
14	**EXTIEN**	**EXTI时钟使能 (EXTI clock enable)** 由软件置1或清0 0：EXTI时钟关闭 1：EXTI时钟开启
13	**PCA7EN**	**PCA7时钟使能 (PCA 7 clock enable)** 由软件置1或清0 0：PCA7时钟关闭 1：PCA7时钟开启
12	**PCA6EN**	**PCA6时钟使能 (PCA 6 clock enable)** 由软件置1或清0 0：PCA6时钟关闭 1：PCA6时钟开启
11	**PCA5EN**	**PCA5时钟使能 (PCA 5 clock enable)** 由软件置1或清0 0：PCA5时钟关闭 1：PCA5时钟开启
10	**PCA4EN**	**PCA4时钟使能 (PCA 4 clock enable)** 由软件置1或清0 0：PCA4时钟关闭 1：PCA4时钟开启
9	**PCA3EN**	**PCA3时钟使能 (PCA 3 clock enable)**

		由软件置1或清0 0：PCA3时钟关闭 1：PCA3时钟开启
8	**PCA2EN**	**PCA2时钟使能 (PCA 2 clock enable)** 由软件置1或清0 0：PCA2时钟关闭 1：PCA2时钟开启
7	**PWM3EN**	**PWM3时钟使能 (PWM 3 clock enable)** 由软件置1或清0 0：PWM3时钟关闭 1：PWM3时钟开启
6	**PWM2EN**	**PWM2时钟使能 (PWM 2 clock enable)** 由软件置1或清0 0：PWM2时钟关闭 1：PWM2时钟开启
5	**PWM1EN**	**PWM1时钟使能 (PWM 1 clock enable)** 由软件置1或清0 0：PWM1时钟关闭 1：PWM1时钟开启
4	**PWM0EN**	**PWM0时钟使能 (PWM 0 clock enable)** 由软件置1或清0 0：PWM0时钟关闭 1：PWM0时钟开启
3	**TIM8EN**	**定时器8时钟使能 (Timer 8 clock enable)** 由软件置1或清0 0：定时器8时钟关闭 1：定时器8时钟开启
2	**TIM7EN**	**定时器7时钟使能 (Timer 7 clock enable)** 由软件置1或清0 0：定时器7时钟关闭 1：定时器7时钟开启
1	**TIM6EN**	**定时器6时钟使能 (Timer 6 clock enable)** 由软件置1或清0 0：定时器6时钟关闭 1：定时器6时钟开启
0	**TIM5EN**	**定时器5时钟使能 (Timer 5 clock enable)** 由软件置1或清0 0：定时器5时钟关闭 1：定时器5时钟开启

10.3.12 RESET 状态寄存器 (RCC_RSTSTR)

偏移地址: 0x002C

复位值: 0x0000 0004

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留									LVRSTF2	WWDTRSTF	IWDTRSTF	SWRSTF	PORSTF	LVRSTF	PINRSTF
-									RO	RO	RO	RO	RO	RO	RO
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

位	符号	说明
31 ~ 7	保留	-
6	LVRSTF2	**VCC上低电压复位标志 (LVR reset flag2)** 在VCC上有低电压复位发生时由硬件置1；由软件通过写LVRSTF2C位清除。 0：无VCC低电压复位发生 1：发生VCC低电压复位
5	WWDTRSTF	**窗口看门狗复位标志 (Window watchdog reset flag)** 在窗口看门狗复位发生时由硬件置1；由软件通过写WWDTRSTFC位清除。 0：无窗口看门狗复位发生 1：发生窗口看门狗复位
4	IWDTRSTF	**独立看门狗复位标志 (Independent watchdog reset flag)** 在独立看门狗复位发生时由硬件置1；由软件通过写IWDTRSTFC位清除。 0：无独立看门狗复位发生 1：发生独立看门狗复位
3	SWRSTF	**软件复位标志 (Software reset flag)** 在软件复位发生时由硬件置1；由软件通过写SWRSTFC位清除。 0：无软件复位发生 1：发生软件复位
2	PORSTF	**上电复位标志 (POR reset flag)** 在上电复位发生时由硬件置1；由软件通过写PORSTFC位清除。 0：无上电复位发生 1：发生上电复位
1	LVRSTF	**低电压复位标志 (LVR reset flag)** 在低电压复位发生时由硬件置1；由软件通过写LVRSTFC位清除。 0：无低电压复位发生 1：发生低电压复位
0	PINRSTF	**NRST引脚复位标志 (PIN reset flag)** 在NRST引脚复位发生时由硬件置1；由软件通过写PINRSTFC位清除。 0：无NRST引脚复位发生 1：发生NRST引脚复位

10.3.13 清除 RESET 状态寄存器 (RCC_RSTCLR)

偏移地址: 0x0030

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留									LVRSTF2C	WWDTRSTFC	IWDTRSTFC	SWRSTFC	PORSTFC	LVRSTFC	PINRSTFC
-									WO	WO	WO	WO	WO	WO	WO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 7	保留	-
6	LVRSTF2C	**清除VCC上低电压复位标志LVRSTF2** 由软件置1来清除复位标志。 0：无作用 1：清除复位标志
5	WWDTRSTFC	**清除WWDT复位标志** 由软件置1来清除复位标志。 0：无作用 1：清除复位标志
4	IWDTRSTFC	**清除IWDT复位标志** 由软件置1来清除复位标志。 0：无作用 1：清除复位标志
3	SWRSTFC	**清除SOFTWARE复位标志** 由软件置1来清除复位标志。 0：无作用 1：清除复位标志
2	PORSTFC	**清除POR复位标志** 由软件置1来清除复位标志。 0：无作用 1：清除复位标志
1	LVRSTFC	**清除LVR复位标志** 由软件置1来清除复位标志。 0：无作用 1：清除复位标志
0	PINRSTFC	**清除PIN复位标志** 由软件置1来清除复位标志。 0：无作用 1：清除复位标志

10.3.14 HSI 振荡器校准寄存器 (RCC_HSICAL)

偏移地址: 0x0034

复位值: 0xXXXX XX00

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TRIMRUN	保留		TRIMREF[12:0]												
RW	-		RO												
0	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
HSICAL[7:0]								保留					HSITRIM[2:0]		
RO								-					RW		
X	X	X	X	X	X	X	X	0	0	0	0	0	0	0	0

位	符号	说明
31	**TRIMRUN**	**HSI用户调整开关，只能软件置1，硬件或软件清0** 0：运算已结束或未启动 1：启动HSI ADJ，运算过程中标志为1,运算完成后标志由硬件清0。 注：允许运算过程中清零RUN以强行结束运算。 RUN启动时会把CORDIC内部flag清零。
30 ~ 29	**保留**	**-**
28 ~ 16	**TRIMREF[12:0]**	**HSI用户校准参数值** 在RUN运行完成后，用户可从该寄存器读取HSI用户校准参数值 （HSI计数HSE分频源）
15 ~ 8	**HSICAL[7:0]**	**内部高频RC振荡器校准** 在系统启动时，这些位被自动初始化
7 ~ 3	**保留**	**-**
2 ~ 0	**HSITRIM[2:0]**	**内部高频RC振荡器调整** 由软件写入来调整HSI频率，它们被叠加在HSICAL[7:0]数值上 这些位在HSICAL[7:0]的基础上，让用户可以输入一个调整数值，根据电压和温度的变化调整内部HSI RC振荡器的频率 000： +0 （default） 001： +0.25% 010：+0.5% 011：+0.75% 100：-1% 101：-0.75% 110：-0.5% 111：-0.25%

10.3.15 CSM 档位选择 (RCC_CSMSEL)

偏移地址: 0x0038

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留													CSMLVL[2:0]		
-													*RW*		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 3	保留	-
2 ~ 0	CSMLVL[2:0]	CSM频率选择控制 000：4MHz 001：6MHz 010：8MHz 011：12MHz 100：32kHz 其他：保留

10.3.16 RCC 配置锁定寄存器 (RCC_RCCLOCK)

偏移地址: 0x003C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LOCK[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	LOCK[15:0]	RCC 配置操作解锁位 (RCC register lock bit) 0x33CC： 解锁 其他：锁定

10.3.17 外部高频晶振/陶振驱动电路反馈电阻选择 (HSERFB)

偏移地址: 0x0044

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留														RFBSEL[1:0]	
-														*RW*	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 2	保留	-
1 ~ 0	RFBSEL[1:0]	HSE时钟反馈电阻选择控制(推荐选用500kΩ) 00：150kΩ 01：200kΩ 10：300kΩ 11：500kΩ

11. 看门狗定时器（WDT）

11.1 独立看门狗定时器（IWDT）

独立看门狗具有一个12位的递减计数器，使用内建128kHz WDT RC作为时钟源（注：128KHz WDT RC 精度为±50%）。独立看门狗默认上电开启，且不可关闭[注]。

注：在DEBUG全速仿真模式下，可通过DBG_IWDT@SYSCFG_DBGCR关闭。关闭后，IWDT计数器不再计数，直至下次通过DBG_IWDT@SYSCFG_DBGCR开启时，计数器清零并重新开始计数。特别注意，该控制寄存器仅在DEBUG仿真模式下有效，脱机运行模式下，IWDT始终开启，无法关闭。

IWDT可以工作在停机模式下，因此可以作为唤醒定时器，通过设置使能位IWDTPD@IWDT_CR可选择停机时开启或关闭看门狗（默认停机模式下关闭）。使能位IWDTPD@IWDT_CR一旦开启，开启后关闭无效，只有在复位时才能关闭。

在定时器溢出之前，通过对喂狗寄存器IWDT_CLR写入0xAAAA更新计数器以重新开始计数，避免产生溢出。

定时器溢出后将复位芯片，并置位IWDTRSTF@RCC_RSTSTR标志。

上电开启后，看门狗工作在较长溢出时间下，溢出时间大约为4096*32/128=1024ms。

在调试时，通过“DBG_IWDT@SYSCFG_DBGCR”可以控制调试时看门狗是否停止工作。

11.2 窗口看门狗定时器（WWDT）

窗口看门狗通常被用来监测，由外部干扰或不可预见的逻辑条件造成的应用程序背离正常的运行序列而产生的软件故障。

窗口看门狗是一个8位递减计数器，喂狗区间为一个窗口区，早于窗口喂狗区喂狗将产生超前异常事件，晚于窗口喂狗区将产生延迟异常事件，两者都会引起复位。另外，在计数器到达窗口喂狗区下边界时为最迟喂狗时刻，可申请WWDT中断，可作为窗口喂狗的最后补救时刻。

SH32F9001_32F9801中WWDT的窗口下边界固定为0。在计数器到达0xFF时产生复位。

窗口看门狗通过WWDTON@WWDT_CR开启，一旦开启，只有在复位时才能关闭。

窗口看门狗工作在APB0总线上，由APB0时钟驱动。

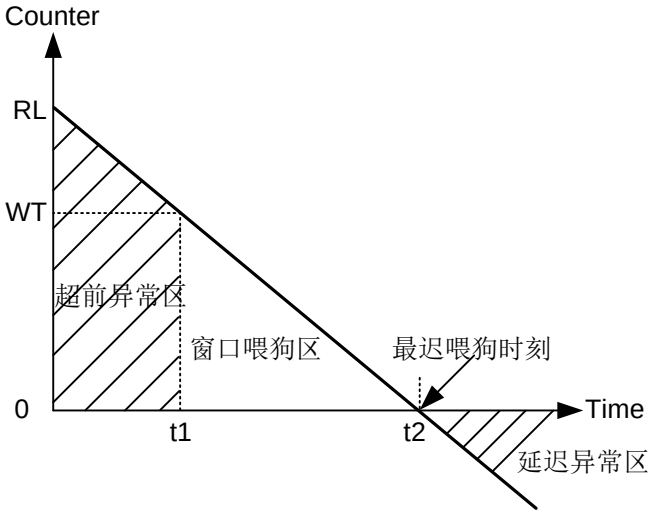


图11-1　　窗口看门狗示意图

应用WWDT时，先设置WWDT的窗口重载值WWDTRLR(图中RL)，窗口值WWDTWTR(图中WT)，然后开启WWDT，计数器开始运行（counter初始为重载值）。在正常喂狗区通过对喂狗寄存器WWDTCLR写入0x5555以更新计数器，避免产生溢出。

如设置RL>=WT则是正常窗口监控功能，如设置RL<WT，则没有超前异常监控，只能监控延迟异常。

注：如果设置RL=WT=0xFF，则窗口最大，无超前监控，功能类似普通看门狗。

当微控制器进入调试模式时(Cortex-M3核心停止)，根据调试模块中的DBG_WWDT配置位的状态，WWDT的计数器能够继续工作或停止。详见“调试接口”章节。

11.3 寄存器

IWDT 模块寄存器列表 (基地址: 0x4000 7C00)

地址	寄存器名	说明
0x4000 7C00	CR	独立看门狗控制寄存器
0x4000 7C04	CLR	独立看门狗喂狗寄存器

11.3.1 独立看门狗控制寄存器 (IWDT_CR)

偏移地址: 0x0000

复位值: 0x0000 7FFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IWDTPD	IWDTPR[2:0]			IWDTRLR[11:0]											
RW1t	RW			RW											
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	符号	说明
31 ~ 16	**LOCK[15:0]**	**寄存器锁定位** 0x5AA5：解锁此寄存器 其他：锁定此寄存器 配置此寄存器时解锁位必须同时输入 0x5AA5，操作完成后自动锁定，读取此 LOCK 位固定读出 0x0。
15	**IWDTPD**	**独立看门狗睡眠/停机模式下使能位** 0：睡眠/停机模式下禁止独立看门狗工作 1：睡眠/停机模式下允许独立看门狗工作 说明：该位一旦开启，开启后关闭无效，只有在复位时才能关闭。
14 ~ 12	**IWDTPR[2:0]**	**独立看门狗时钟预分频设置** 000: 预分频因子=4 001: 预分频因子=8 010: 预分频因子=16 011: 预分频因子=32（默认） 100: 预分频因子=64 101: 预分频因子=128 110: 预分频因子=256 111: 预分频因子=512 注：IWDTPR[2:0]上电默认值是 011b。
11 ~ 0	**IWDTRLR[11:0]**	**独立看门狗重载寄存器** 用于定义看门狗计数器的重装载值。 注：IWDTRLR[11:0]复位值为 0x0FFF。

11.3.2 独立看门狗喂狗寄存器 (IWDT_CLR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IWDTCLR[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	IWDTCLR[15:0]	独立看门狗喂狗寄存器 0xAAAA：喂狗（用 IWDTRLR 更新看门狗计数器） 其他：无意义 喂狗完成后读此寄存器固定返回 0x0。

WWDT 模块寄存器列表 (基地址:0x0x4000 8000)

地址	寄存器名	说明
0x4000 8000	CR	窗口看门狗控制寄存器
0x4000 8004	SR	窗口看门狗状态寄存器
0x4000 8008	CLR	窗口看门狗喂狗寄存器
0x4000 800C	WTR	窗口看门狗窗口寄存器

11.3.3 窗口看门狗控制寄存器 (WWDT_CR)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WWDTON	WWDTIE	保留			WWDTPR[2:0]			WWDTRLR[7:0]							
RW1t	RW	-			RW			RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	LOCK[15:0]	寄存器锁定位 0x5AA5：解锁此寄存器 其他：锁定此寄存器 配置此寄存器时解锁位必须同时输入0x5AA5，操作完成后自动锁定，读取此LOCK位固定读出0x0000。
15	WWDTON	窗口看门狗使能，此位由软件置1，但仅能由硬件在复位后清0。 0：禁止WWDT 1：启用WWDT 说明：该位一旦开启，开启后关闭无效，只有在复位时才能关闭。
14	WWDTIE	窗口看门狗最迟喂狗中断使能，此位由软件置1和清0。

		0：禁止WWDT中断 1：启用WWDT中断
13 ~ 11	保留	-
10 ~ 8	**WWDTPR[2:0]**	**窗口看门狗时钟预分频设置** 000: 预分频因子=1X256 001: 预分频因子=2X256 010: 预分频因子=4X256 011: 预分频因子=8X256 100: 预分频因子=16X256 101: 预分频因子=32X256 110: 预分频因子=64X256 111: 预分频因子=128X256
7 ~ 0	**WWDTRLR[7:0]**	**窗口看门狗重载寄存器** 用于定义看门狗计数器的重装载值。

11.3.4 窗口看门狗状态寄存器 (WWDT_SR)

偏移地址: 0x0004

复位值: 0x0000 00FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WWDTIF	保留							TCNT[7:0]							
RW	-							*RO*							
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

位	符号	说明
31 ~ 16	保留	-
15	**WWDTIF**	**窗口看门狗最迟喂狗中断标志位，此位由软件或硬件置1，软件清0。** 0：未产生WWDT中断 1：产生了WWDT中断
14 ~ 8	保留	-
7 ~ 0	**TCNT[7:0]**	**窗口看门狗计数寄存器** 指示当前看门狗计数值，启动时默认为最大值0xFF。

11.3.5 窗口看门狗喂狗寄存器 (WWDT_CLR)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
WWDTCLR[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**WWDTCLR[15:0]**	**窗口看门狗喂狗寄存器** 0x5555：喂狗（用WWDTRLR更新看门狗计数器） 其他：无意义 喂狗完成后读此寄存器固定返回0x0。

11.3.6 窗口看门狗窗口寄存器 (WWDT_WTR)

偏移地址: 0x000C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								WWDTWTR[7:0]							
-								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LOCK[15:0]**	**寄存器锁定位** 0x5AA5：解锁此寄存器 其他：锁定此寄存器 配置此寄存器时解锁位必须同时输入0x5AA5，操作完成后自动锁定，读取此LOCK位固定读出0x0000。
15 ~ 8	保留	-
7 ~ 0	**WWDTWTR[7:0]**	**窗口看门狗窗口值寄存器** 用于定义看门狗的窗口值。 该寄存器具有写保护，只有LOCK位为0x5AA5时才能写入，读出不受限制。

12. 代码和数据检测（Code and Data Checking）

12.1 CRC 模块（CRC）

12.1.1 CRC 简介

循环冗余校验(CRC)计算单元是根据可设置的生成多项式得到任一32位/16位/8位的CRC计算结果。CRC技术主要应用于核实数据传输或者数据存储的正确性和完整性。标准EN/IEC 60335-1需要提供一种核实闪存存储器完整性的方法，利用 CRC计算单元可以在程序运行时计算出软件的标识，与在连接时生成的参考标识比较，检查闪存存储器完整性。在通讯过程中，将CRC值添加到数据末尾进行传送，接收端将所接收到的数据逐个送入CRC单元，最后判断最后的CRC值是否为0来验证传输数据的完整性。

12.1.2 CRC 主要特性

- 支持多种CRC格式
  CRC-32: (0x04C11DB7)
  多项式: $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$
  CRC-16: (0x8005)
  多项式: $x^{16}+x^{15}+x^{2}+1$
  CRC-CCITT: (0x1021)
  多项式: $x^{16}+x^{12}+x^{5}+1$
  CRC-8: (0x7):
  多项式: $x^{8}+x^{2}+x+1$
- 支持修改初始值 CRC_INIT. 当计算CRC-16时，初始值CRC_INIT的低16位有效，当计算CRC-8时，初始值CRC_INIT的低8位有效。
- 可配置输入/输出数据转换，包括字节逆向，字节内位逆向，及1’s complement （ 按位取反)。
- 支持8/16/32位数据输入。输入8位则一次只计算8位数据，输入16位则一次只计算16位数据，输入32位则一次计算32位数据。此输入格式与CRC格式无关。计算数据的基本单元为8位数据，因此当输入多字节数据时是低字节先运算。
- 支持DMA及 PIO模式 (CRC_DR可作为DMA的目标地址，也可直接寄存器访问)
- 模块时钟可以独立打开/关闭
- 计算CRC-8需要1个时钟，CRC-16需要2个时钟，CRC-32需要4个时钟

循环冗余校验(CRC)计算单元是根据可设置的生成多项式得到任一32位/16位/8位的CRC计算结果。CRC技术主要应用于核实数据传输或者数据存储的正确性和完整性。标准EN/IEC 60335-1需要提供一种核实闪存存储器完整性的方法，利用 CRC计算单元可以在程序运行时计算出软件的标识，与在连接时生成的参考标识比较，检查闪存存储器完整性。在通讯过程中，将CRC值添加到数据末尾进行传送，接收端将所接收到的数据逐个送入CRC单元，最后判断最后的CRC值是否为0来验证传输数据的完整性。

12.1.3 CRC 功能描述

如图12.1所示。计算CRC时首先根据实际情况对CRC_CR进行格式设置。可设置的参数有(MODE, RBITR, RBITW, RBYTER, RBYTEW, COMPLR,COMPLW)@CRC_CR. 然后设置格式所需的生成多项式及CRC初始值。设置完毕后需要往RELOAD@CRC_CR写0，将初始值载入CRC_DR单元。之后就可以往CRC_DR中写数据并从CRC_DR中读取计算结果。每往CRC_DR中写一个数据就会触发一次CRC运算，使用者可以通过读取CRC_DR得到当前计算结果。

数据写入时支持8位，16位，32位写入，但要求写入/读取时必须按字对齐(4字节对齐,即必须是CRC_DR的首地址)，否则结果不可预测。一次写入时数据与具体的CRC结果的位数无关，但与CRC结果相关。当写入8位数据时只计算8位数据，当写入16位数据时只计算16位数据，当写入32位数据时一次计算32位数据。基本的计算单位为一个字节，当写入数据大于1个字节时低字节先参与运算(详见算法描述)。所以当输入转换设置为全0时(即不对写入数据作任何变换)，一次写入0x12345678与连续写入0x78,0x56,0x34,0x12得到的结果是一样的。

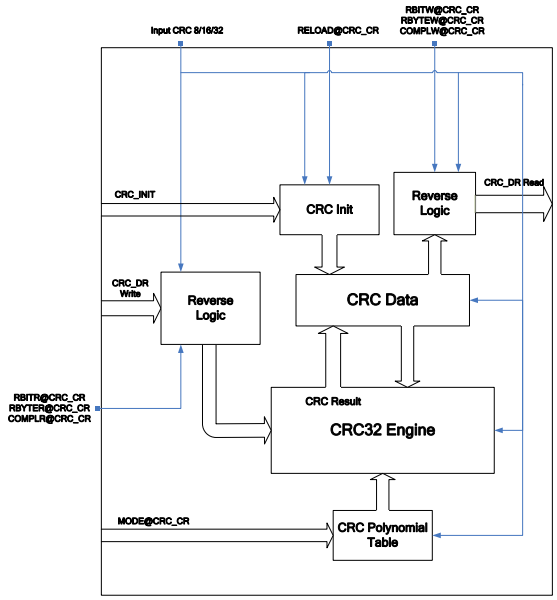


图12-1　　CRC 逻辑模块图

12.1.3.1　　CRC 格式转换描述

由于对寄存器操作时可以是8位，16位或32位。所以在对数据预处理时会根据输入数据的差异进行不同的处理。读取时只与CRC Mode有关。

表12-1　　数据格式转换描述

	8位写入			读取		
	COMPLW	RBITW	RBYTEW	COMPLR	RBITR	RBYTER
CRC_8 MODE=2	Byte0 = ~Byte0	Bit0<->Bit7	忽略	Byte0 = ~Byte0	Bit0<->Bit7	忽略
CRC_16 MODE=1	Byte0 = ~Byte0	Bit0<->Bit7	忽略	Byte0 = ~Byte0 Byte1 = ~Byte1	Bit0<->Bit7 Bit8<->Bit15	Byte1<->Byte0
CRC_32 MODE=0	Byte0 = ~Byte0	Bit0<->Bit7	忽略	Byte0 = ~Byte0 Byte1 = ~Byte1 Byte2 = ~Byte2 Byte3 = ~Byte3	Bit0<->Bit7 Bit8<->Bit15 Bit16<->Bit23 Bit24<->Bit31	Byte0<->Byte3 Byte1<->Byte2
	16位写入			读取		
	COMPLW	RBITW	RBYTEW	COMPLR	RBITR	RBYTER
CRC_8 MODE=2	Byte0 = ~Byte0 Byte1 = ~Byte1	Bit0<->Bit7 Bit8<->Bit15	Byte1<->Byte0	Byte0 = ~Byte0	Bit0<->Bit7	忽略
CRC_16 MODE=1	Byte0 = ~Byte0 Byte1 = ~Byte1	Bit0<->Bit7 Bit8<->Bit15	Byte1<->Byte0	Byte0 = ~Byte0 Byte1 = ~Byte1	Bit0<->Bit7 Bit8<->Bit15	Byte1<->Byte0
CRC_32 MODE=0	Byte0 = ~Byte0 Byte1 = ~Byte1	Bit0<->Bit7 Bit8<->Bit15	Byte1<->Byte0	Byte0 = ~Byte0 Byte1 = ~Byte1 Byte2 = ~Byte2 Byte3 = ~Byte3	Bit0<->Bit7 Bit8<->Bit15 Bit16<->Bit23 Bit24<->Bit31	Byte0<->Byte3 Byte1<->Byte2
	32位写入			读取		
	COMPLW	RBITW	RBYTEW	COMPLR	RBITR	RBYTER
CRC_8 MODE=2	Byte0 = ~Byte0 Byte1 = ~Byte1 Byte2 = ~Byte2 Byte3 = ~Byte3	Bit0<->Bit7 Bit8<->Bit15 Bit16<->Bit23 Bit24<->Bit31	Byte0<->Byte3 Byte1<->Byte2	Byte0 = ~Byte0	Bit0<->Bit7	忽略
CRC_16 MODE=1	Byte0 = ~Byte0 Byte1 = ~Byte1 Byte2 = ~Byte2 Byte3 = ~Byte3	Bit0<->Bit7 Bit8<->Bit15 Bit16<->Bit23 Bit24<->Bit31	Byte0<->Byte3 Byte1<->Byte2	Byte0 = ~Byte0 Byte1 = ~Byte1	Bit0<->Bit7 Bit8<->Bit15	Byte1<->Byte0
CRC_32	Byte0 = ~Byte0	Bit0<->Bit7	Byte0<->Byte3	Byte0 = ~Byte0	Bit0<->Bit7	Byte0<->Byte3

MODE=0	Byte1 = ~Byte1 Byte2 = ~Byte2 Byte3 = ~Byte3	Bit8<->Bit15 Bit16<->Bit23 Bit24<->Bit31	Byte1<->Byte2	Byte1 = ~Byte1 Byte2 = ~Byte2 Byte3 = ~Byte3	Bit8<->Bit15 Bit16<->Bit23 Bit24<->Bit31	Byte1<->Byte2

12.1.3.2 常见格式的配置

CRC32的设置 (0x04C11DB7)

$$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$$

CRC_INIT : 0xFFFFFFFF
CRC_CR.MODE : 0
CRC_CR.RBITW : 0
CRC_CR.RBYTEW : 0
CRC_CR.COMPLW : 0
CRC_CR.RBITR : 0
CRC_CR.RBYTER : 0
CRC_CR.COMPLR : 0
输入:
8位: 一次计算1个字节
16位: 一次计算2个字节，低字节先算
32位: 一次计算4个字节，低字节先算
CRC结果：CRC_DR的全部32-bit

CRC16 的设置 (0x8005)

$$x^{16}+x^{15}+x^{2}+1$$

CRC_INIT : 0xFFFF
CRC_CR.MODE : 1
CRC_CR.RBITW : 0
CRC_CR.RBYTEW : 0
CRC_CR.COMPLW : 0
CRC_CR.RBITR : 0
CRC_CR.RBYTER : 0
CRC_CR.COMPLR : 0
输入:
8位: 一次计算1个字节
16位: 一次计算2个字节，低字节先算
32位: 一次计算4个字节，低字节先算
CRC结果：CRC_DR的低16位

CRC16-CITT 的设置 (0x1021)

$$x^{16}+x^{12}+x^{5}+1$$

CRC_INIT : 0xFFFF
CRC_CR.MODE : 2
CRC_CR.RBITW : 0
CRC_CR.RBYTEW : 0
CRC_CR.COMPLW : 0
CRC_CR.RBITR : 0

CRC_CR.RBYTER : 0
CRC_CR.COMPLR : 0
输入:
8位: 一次计算1个字节
16位: 一次计算2个字节，低字节先算
32位: 一次计算4个字节，低字节先算
CRC结果：CRC_DR的低16位

CRC8的设置 (0x7)

$X^8 + X^2 + X + 1$

CRC_INIT : 0xFF
CRC_CR.MODE : 3
CRC_CR.RBITW : 0
CRC_CR.RBYTEW : 0
CRC_CR.COMPLW : 0
CRC_CR.RBITR : 0
CRC_CR.RBYTER : 0
CRC_CR.COMPLR : 0
输入:
8位: 一次计算1个字节
16位: 一次计算2个字节，低字节先算
32位: 一次计算4个字节，低字节先算
CRC结果：CRC_DR的低8位

12.1.4 CRC 寄存器

CRC 模块寄存器列表 (基地址: 0x4004 5000)

地址	寄存器名	说明
0x4004 5000	DR	CRC数据寄存器
0x4004 5004	CR	CRC控制寄存器
0x4004 5008	INIT	CRC初始值寄存器

12.1.4.1 CRC 数据寄存器 (CRC_DR)

偏移地址: 0x0000
复位值: 0xFFFF FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DR[31:0]															
RW															
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DR[31:0]															
RW															
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	符号	说明
31 ~ 0	**DR[31:0]**	1. 输入时用作写一个新的数据到CRC计算单元，并启动计算. 2. 读取时返回上一次CRC计算的结果 3. 允许一次写入8位，16位，32位。

12.1.4.2 CRC 控制寄存器 (CRC_CR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RBYTER	RBITR	COMPLR	RBYTEW	RBITW	COMPLW	MODE[1:0]		保留							RELOAD
RW	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*		-							*WO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15	**RBYTER**	**对DR中的数据进行字节顺序逆向后再读取** 0: 保持不变 1: 进行字节顺序逆向 (具体变换见表12-1 数据转换描述表)
14	**RBITR**	**对DR中的数据进行字节内位顺序逆向后再读取** 0: 保持不变 1: 进行字节内位顺序逆向(具体变换见表12-1 数据转换描述表)
13	**COMPLR**	**对DR中的数据进行1’s complement变换后再读取.(即对的数据进行按位取反)** 0:保持不变 1:进行1’s complement变换
12	**RBYTEW**	**对写入的数据进行字节顺序逆向** 0: 保持不变 1: 进行字节顺序逆向 (具体变换见表12-1 数据转换描述表)
11	**RBITW**	**对写入的数据进行字节内位顺序逆向** 0: 保持不变 1: 进行字节内位顺序逆向(具体变换见表12-1 数据转换描述表)
10	**COMPLW**	**对写入的数据进行1’s complement变换。(即对写入的数据进行按位取反)** 0:保持不变 1:进行1’s complement变换
9 ~ 8	**MODE[1:0]**	**CRC算法模式选择，因为用逆向算法，所以生成多项式的实际值是按位逆向的。** 00: CRC-32 (0x04C11DB7) – 选table crc32_04C11DB7 01: CRC-16 (生成多项式为0x8005 -- modbus) – 选table crc16_8005 10: CRC-CITT (生成多项式为0x1021) – 选table crc16_1021 11: CRC-8 (生成多项式为0x7) – 选table crc8_7
7 ~ 1	保留	-
0	**RELOAD**	**将CRC的initial值重新载入到运算寄存器** 1: 写1重新载入。将INIT@CRC_INIT重新载入DR@CRC_DR寄存器，其他寄存器值保持不变。 0: 写0不变，此Bit无法读取，始终为 0。

12.1.4.3 CRC 初始值寄存器 (CRC_INIT)

偏移地址: 0x0008

复位值: 0xFFFF FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
INIT[31:0]															
RW															
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
INIT[31:0]															
RW															
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	符号	说明
31 ~ 0	**INIT[31:0]**	**存放CRC初始值，往RELOAD@CRC_CR写1后，此值会自动载入到DR@CRC_DR寄存器中。** MODE=0: 32位有效 MODE=1: 低16位有效 MODE=2: 低16位有效 MODE=3: 低8位有效

12.2 SRAM 检测模块（RAMBIST）

12.2.1 SRAM 检测模块简介

为满足IEC60730对RAM区检测的要求，SH32F9001_32F9801提供了硬件实现的RAM检测模块。此模块采用了March-C及March-X算法，使用时可以选择其一。在检测模块允许在运行过程中被程序访问打断，与传统软件实现相比，极大的保证了应用程序的执行效率。

12.2.2 SRAM 检测模块功能描述

1. 此模块采用March-C算法或March-X算法对SRAM进行检测。March-C可以检测到SAF,TF,CFin,CFid,BF,SCF等错误,March-X可以检测到SAF,TF,CFin,AF错误。March-X算法检测速度比March-C的检测速度快。使用时可根据具体的应用要求选择合适的检测算法。
2. 此模块需要使用正在测试RAM的高地址区域作为RAM的数据备份区。在检测过程中，此区域用于保存需要检测RAM(CRAM或SRAM)的数据，等检测完成后再恢复到RAM。所以在检测过程中不允许用户程序访问此备份区域（由软件程序保证,硬件逻辑不做特别检查），否则可能会产生不可预测的错误。数据备份区的大小由BLKSZ@RAMBIST_CFG 寄存器决定。例如BLKSZ为1代表检测大小为32Bytes,所以备份区的大小也为32Bytes.则地址为0x20003FE0~0x20003FFF.
3. 检测范围包括CRAM及SRAM区。可以设置检查的起始地址及区域大小。起始地址必须按检测区域大小对齐。检测区域大小确定后，此模块会在待测区在高位地址占用同样大小的一块空间作为数据备份区域。所以在设置检测区域时必须避免与备份区域重叠，此冲突由软件避免，硬件逻辑不做任何判断。如果发生冲突，可能会导致不可预测的错误。
4. 具有数据备份区的自检功能。在此功能下无需进行数据备份，检测的大小有BLKSZ@RAMBIST_CFG寄存器决定。检测区域位置由RAMTYP决定。
5. 允许在此模块运行过程中用户程序访问除数据备份区域之外的区域。(注:备份区域的访问禁止需要由软件避免,硬件逻辑不做处理，如发生冲突，可能导致程序产生不可预知的错误)。当用户程序访问RAM时，此模块会暂停检测动作，释放RAM访问给用户程序，待用户程序访问完成后，继续之前的检测过程。如果用户程序访问的是被检测区域，此模块会将访问地址映像到备份区域，从备份区域存取程序所需数据，不影响程序的正常运行。用户程序的访问优先于模块的检测，尽可能的保证应用程序的应用功能。
6. 此模块在进行检测前后，会对被检测区域的内容进行备份及恢复(备份区自检除外). 在备份及恢复过程中，允许用户程序访问除备份区之外的区域。用户程序的访问会暂停正在进行的备份或恢复过程，待访问完成后，继续之前的过程。如果访问的是被检测区域，此模块会根据当前备份或恢复的位置判断访问地址是否已备份或恢复，从而决定是从实际RAM地址访问还是从数据备份区访问。用户程序的访问优先于模块的备份及恢复，尽可能的保证应用程序的应用效率。
7. 在备份及恢复过程中，需要对所操作的数进行校验。如果在备份过程中发生检验错误，需要置ERR@ RAMBIST_CSR位且停止此模块的运行。如果在恢复过程中发生校验错误，需要先置ERR@RAMBIST_CSR位并继续后续的恢复动作，直到恢复完成。
8. 在检测过程中发现错误后就立即置ERR@RAMBIST_CSR位，然后中断正在检测过程并启动恢复过程(如在进行备份区自检，无需启动恢复过程)。直到恢复完成后才清BSY@ RAMBIST_CSR.
9. 此模块由RUN@ RAMBIST_CSR启动，启动后自动置BSY@RAMBIST_CSR。（ERR@RAMBIST_CSR位需要由软件在模块启动前清。 允许清0与启动同时进行）. 在检测过程中(包括备份及恢复)发现错误就立即置ERR@RAMBIST_CSR位.待整个模块完成或停止后就清BSY@RAMBIST_CSR位。
10. 在此模块运行过程中，不允许程序修改此模块的寄存器但允许程序读取这些寄存器。此限制由硬件逻辑实现。

12.2.2.1 March-C 算法描述

Step1: 全写0

Step2: 从头到尾按Bit读0判断,然后写1

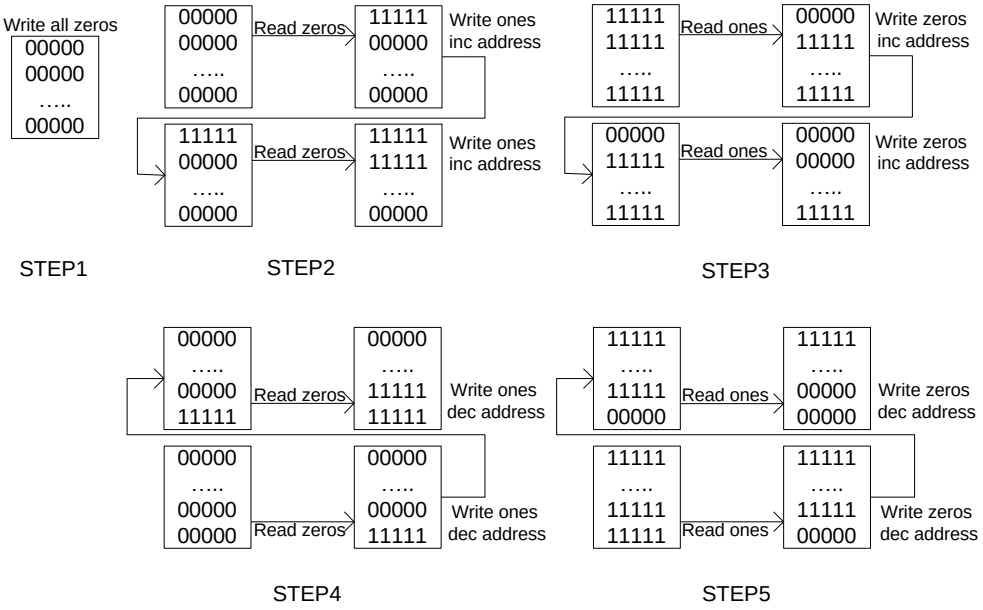
Step3: 从头到尾按Bit读1判断,然后写0

Step4: 从尾到头按Bit读0判断,然后写1

Step5: 从尾到头按Bit读1判断,然后写0

Step6: 读取后判断全0

12.2.2.2 March-C 算法逻辑图



12.2.2.3 March-X 算法描述

March-X 算法是 March-C 算法的子集。

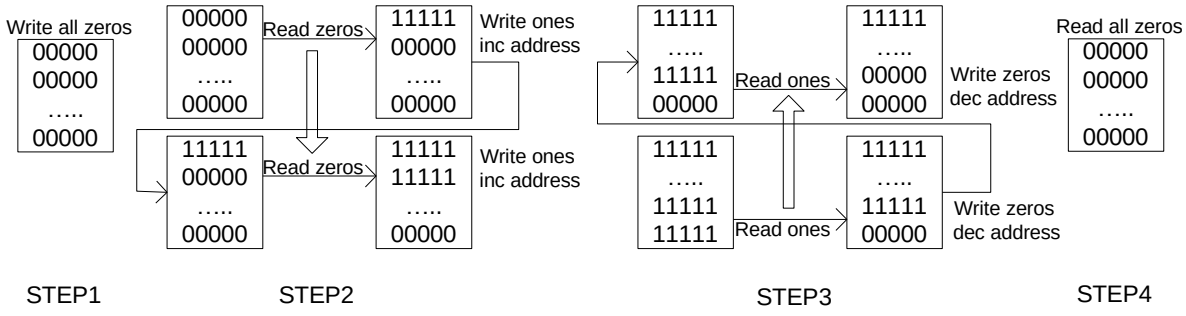
Step1: 全写0

Step2: 从头到尾按Bit读0判断，然后写1

Step3: 从尾到头按Bit读1判断，然后写0

Step4: 读取后判断全0

12.2.2.4 March-X 算法逻辑图



12.2.2.5 SRAM 检测模块操作步骤

软件操作步骤：

1. 备份区域自检
   a) 设置RAMTYP@RAMBIST_ADDR设定备份区域位置(C-RAM 或 D-RAM)
   b) 设置BLKSZ@RAMBIST_CFG设定备份区域大小
   c) 设置SEL@ RAMBIST_CFG选择所用的算法(March-C 或 March-X)
   d) 清ERR@ RAMBIST_CSR。清除错误标志
   e) 设置MOD@ RAMBIST_CSR,设定自检模式
   f) 往RUN@ RAMBIST_CSR写入0x59A6,启动模块运行
   g) 检测BYS@ RAMBIST_CSR,判断是否检测结束
   h) 检测ERR@ RAMBIST_CSR,判断是否出错

2. 普通RAM区域检测
   a) 设置RAMTYP@RAMBIST_ADDR设定检测位置(C-RAM 或 D-RAM)
   b) 设置BLKSZ@RAMBIST_CFG设定检测区域大小
   c) 设置SEL@ RAMBIST_CFG选择所用的算法(March-C 或 March-X)
   d) 清MOD@ RAMBIST_CSR,设定普通检测模式
   e) 设置ADDR@ RAMBIST_ADDR设定检测地址位置
   f) 清ERR@ RAMBIST_CSR。清除错误标志
   g) 往RUN@ RAMBIST_CSR写入0x59A6,启动模块运行
   h) 检测ERR@RAMBIST_CSR,判断是否出错。如果出错就进行错误处理
   i) 检测BYS@RAMBIST_CSR,判断是否检测结束
   j) 循环 e ~ i 直到RAM的所有区块(备份区除外)检查完毕

对应的硬件操作过程：

1. 往RUN@RAMBIST_CSR写入0x59A6启动模块
2. 硬件置BSY@RAMBIST_CSR位为1
3. 备份待检区域。将待检区的数据拷贝到备份区(如果是自检模式不需要此步骤)。按字拷贝，每拷贝完一个字后都将进行读取校验(重新从目标地址读取数据然后与原始数据比较)，校验完成后再拷贝下一个，直到备份完成。如果校验错误，就置错误标志(ERR@RAMBIST_CSR)且退出整个进程(不再进行后续步骤)，然后再清BSY@ RAMBISTCSR位。拷贝期间可以被来自程序的RAM访问打断。等程序访问完成后重新开始之前这个字的的拷贝动作(假设在一个字刚拷贝完还没有校验时被打断，那么恢复拷贝后这个字将被重新读取，写入及校验，只有校验通过才认为这个字备份成功)。如果程序访问的RAM正好是在拷贝的那个字，那么在没有完成校验之前被访问的数据还从原来的区域存取。如果已经校验完成且正确，那么就从备份区域存取。不在检测范围内的访问都从实际地址存取。
4. 备份成功后，对被检区域进行March-C或March-X的检测。具体检测动作参考上面的March-C及March-X算法描述。如果在检测过程中发现错误，就先置错误标志(ERR@ RAMBIST_CSR)，然后中断检测过程后再进入恢复过程。在检测过程中可以被来自程序的RAM访问打断，等程序访问完成后继续之前的检测动作。如果程序访问的区域在当前检测范围内，就从备份区域的相应地址进行存取，如果不在当前检测范围内就从RAM的实际地址存取。
5. 检测完毕后（不管是否有错），都进行备份数据的恢复。恢复时按字将数据从备份区域拷贝到检测好区域。如果恢复时ERR@ RAMBIST_CSR是1(有错误)将不进行恢复数据的校验动作。如果恢复时ERR@ RAMBIST_CSR是0(无错误)，则在每恢复一个字后都将进行校验(重新从目标地址读取数据然后与原始数据比较)，如果不一致就置ERR@ RAMBIST_CSR. 校验结束后,不管之前的校验是否正确，都将进行下一个字的恢复(出错后不再对恢复数据进行校验)，直到全部完成。恢复期间可以被来自程序的RAM访问打断。等程序访问完成后重新开始之前这个字的的恢复动作(假设在一个字刚拷贝完还没有校验时被打断，那么恢复后这个字将被重新读取，写入及校验，只有校验通过才认为这个字恢复成功)。如果程序访问的RAM正好是在拷贝的那个字，那么在没有完成校验之前被访问的数据还从备份区域存取。如果已经校验完成且正确，那么就从实际区域存取。不在检测范围内的访问都从实际地址存取。
6. 恢复完成后，清BSY@ RAMBIST_CSR位，结束此次检测。
7. 当BSY@ RAMBIST_CSR位为1时，不允许软件修改此模块中的寄存器值，此限制由硬件实现。

12.2.3 RAMBIST 寄存器

RAMBIST 模块寄存器列表 (基地址: 0x4004 5400)

地址	寄存器名	说明
0x4004 5400	ADDR	SRAM自检地址寄存器
0x4004 5404	CFG	SRAM自检配置寄存器
0x4004 5408	CSR	SRAM自检控制寄存器

12.2.3.1 SRAM 自检地址寄存器 (RAMBIST_ADDR)

偏移地址: 0x0000
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留		RAMTYP[1:0]		保留											
-		RW		-											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留		RAMADR[13:0]													
-		RW													
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 30	保留	-
29 ~ 28	RAMTYP[1:0]	选择自检区域CRAM及SRAM. 00:无效 01:选择CRAM区 10:选择SRAM区 11:无效
27 ~ 14	保留	-
13 ~ 0	RAMADR[13:0]	设置要检查RAM区域的地址[A13:A0]. 待检测区域的地址必须根据BLKSZ的大小对齐，如果不对齐会由硬件自动按BLKSZ的大小舍弃低位地址。 例: 检测区域的BLKSZ为 1 (表示32字节对齐),所以合法的RAMADDR为 00 RAMTYP[1:0] 0000 00xx xxxx xxx0 0000 如果输入00 RAMTYP[1:0] 0000 00xx xxxx xxx0 0101,在实际检查时也是从00 RAMTYP[1:0] 0000 00xx xxxx xxx0 000

12.2.3.2 SRAM 自检配置寄存器 (RAMBIST_CFG)

偏移地址: 0x0004
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SEL	保留												BLKSZ[2:0]		
RW	-												RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15	SEL	选择检测算法 0：March-C 1：March-X
14 ~ 3	保留	-
2 ~ 0	BLKSZ[2:0]	测试区域大小 0: 16Bytes 1: 32Bytes 2: 64Bytes 3: 128Bytes 4: 256Bytes 5: 512Bytes 6: 1024Bytes 7: 2048Bytes

12.2.3.3 SRAM 自检控制寄存器 (RAMBIST_CSR)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留													ERR	BSY	MOD
-													RW	RO	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RUN[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 19	保留	-
18	ERR	模块出错标志，软件清0.与RUN位同时进行. 0：没有检测到错误 1：检测到有错误 (包括:数据备份校验错误,RAM检测失败, 数据恢复校验错误, RAMTYP无效)
17	BSY	模块运行标志位，由模块自动置位及清除 0：表示此模块没有检测或检测已完成 1：表示此模块正在检测
16	MOD	选择检测模式 0: 检测有寄存器ADDR@RAMBIST及BLKSZ[2:0]@RAMBIST_CFG指定的RAM区域 1: 检测数据备份区域. 数据备份区域大小BLKSZ[2:0]@RAMBIST_CFG寄存器决定,位置在SRAM的高位地址. 在此模式下ADDR@RAMBIST的设置被忽略.
15 ~ 0	RUN[15:0]	启动检测，启动后BSY Flag自动置1 写0x59A6：启动检测，读取永远为0 其他： 不动作

13. 通用功能 I/O (GPIO)

13.1 简介

SH32F9001_32F9801提供最多75个通用I/O口，分成5组端口（A～E），每组最多包含16个输入输出引脚。每组I/O端口都有独立的配置寄存器和数据寄存器。

每一个I/O口都有复用功能，或者与片上数字外设复用，或者与片上模拟外设复用，或者与外部中断与调试接口复用。

所有I/O都支持弱上拉和下拉、以及推挽和开漏输出，I/O口输出驱动能力多档可调，个别I/O口提供超强的灌电流能力，以满足某些应用中直驱LED或小功率器件的要求。

13.2 主要特性

- 所有端口组都挂载在AHB总线上，支持快速输入输出
- 输入/输出方向控制
- 输入管脚可配置弱上拉/下拉
- 输出管脚可配置为推挽或开漏，驱动能力多档可调
- 基于寄存器的按位置位/复位控制
- 端口寄存器支持按字节操作
- 部分端口可用作外部中断，触发沿可设置（需结合EXTI 配置寄存器）
- I/O口可配置为普通输入输出功能/数字外设功能/模拟外设功能
- I/O口配置使用多路选择器，同一时刻只能配置为一种功能
- 复用为数字外设功能时若输入通路使能，允许读引脚输入状态
- 部分端口支持复用功能重映射
- 支持端口配置锁定功能

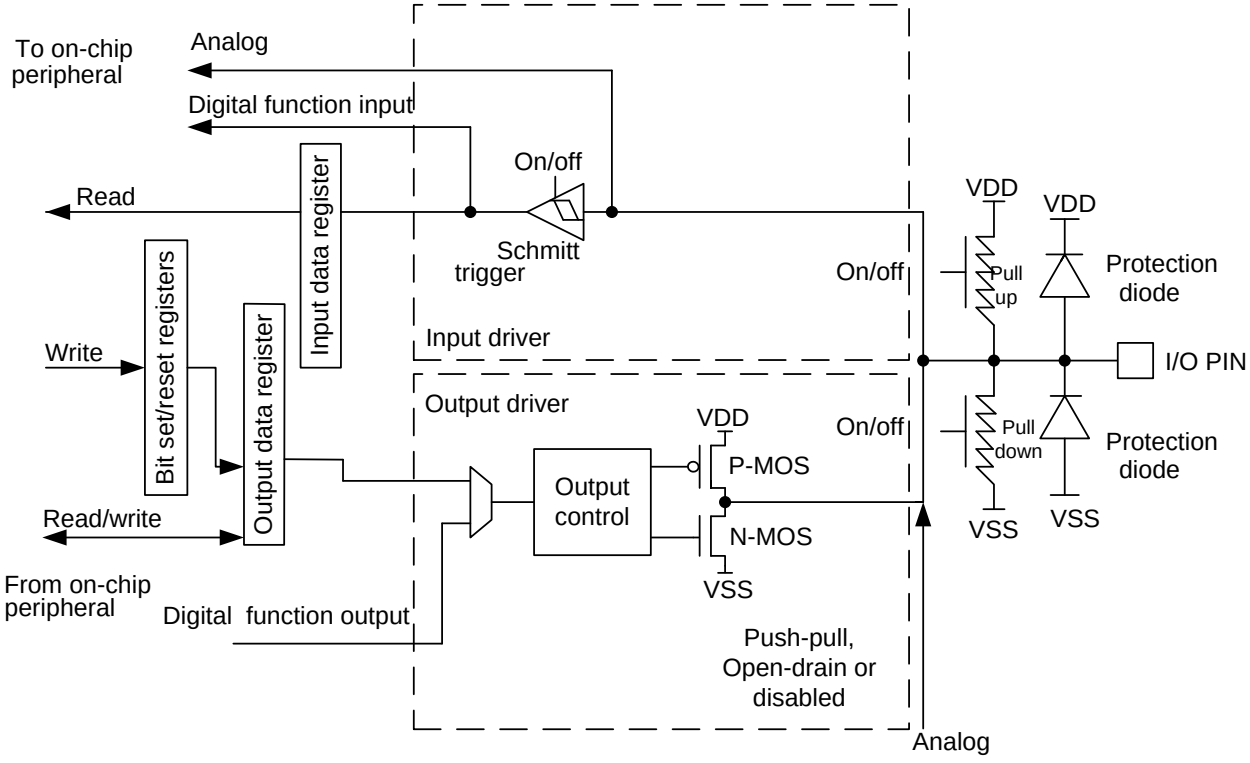


图13-1　　I/O 端口基本结构框图

13.3 功能描述

每个I/O模式配置是通过4个基本配置寄存器（MODER、OTYPER、ODRVR、PUPDR、TTLEN）完成的，每个寄存器都支持以32位字、16位半字、8位字节的方式读写，用于配置I/O的输入输出模式（方向）、输出推挽/开漏方式、输出驱动档位、端口上下拉、输入电平逻辑属性。

I/O端口的基本工作模式如下表所示：

表13-1　　I/O 端口基本工作模式一览

端口功能[注4]	AFRL/AFRH	MODER	OTYPE	ODRVR	PLDR	PHDR	I/O属性	
GPIO	AF0	x	x	x	0	0	GPIO高阻	高阻态
					0	1		上拉
					1	0		下拉
					1	1		上下拉[注1]
	AF1	0	x	x	0	0	GPIO输入	浮空
					0	1		上拉
					1	0		下拉
					1	1		上下拉[注1]
		1	0	ODRVR	0	0	GPIO推挽输出	推挽
					0	1		推挽+上拉
					1	0		推挽+下拉
					1	1		推挽+上下拉[注1]
			1		0	0	GPIO开漏输出	开漏
					0	1		开漏+上拉
					1	0		开漏+下拉
					1	1		开漏+上下拉[注1]
外设	AF2～AF15	x[注2]	x	x	0	0	数字输入	浮空
					0	1		上拉
					1	0		下拉
					1	1		上下拉[注1]
			0	ODRVR	0	0	数字推挽输出	推挽
					0	1		推挽+上拉
					1	0		推挽+下拉
					1	1		推挽+上下拉[注1]
			1		0	0	数字开漏输出	开漏
					0	1		开漏+上拉
					1	0		开漏+下拉
					1	1		开漏+上下拉[注1]
		x	x	x	0	0	模拟输入输出	浮空
					0	1		上拉[注3]
					1	0		下拉
					1	1		上下拉

注1：上拉和下拉在输入和输出模式中都可以开启。支持上下拉同时打开以输出近似VDD/2电平（仅个别引脚可输出较为精确的VDD/2电平），但不建议在输入时开启上下拉（会有漏电问题），但系统上不禁止这种组合搭配，由用户应用避免。

注2：数字外设的输入输出配置是与外设绑定的，即启用某个数字外设复用功能后，用户仅需配置OTYPE、ODRVR和PUPDR寄存器，无需配置MODER寄存器。

注3：模拟模式下默认不开启上下拉，特殊应用中可开启上拉或下拉或上下拉以提供一个特定输入电平。

注4：除了表中3种基本的端口功能外，端口还有一种系统接口（system port），包含调试接口SWJ和外部振荡器接口XTAL共7个端口。系统接口也可以复用为GPIO或数字外设接口。但在作为系统接口使用时上述基本配置寄存器无效，这两种接口在SYSCFG中有进一步说明。

13.3.1 I/O 引脚的默认配置

在复位期间和复位之后，I/O口功能默认配置为AF0通道“I/O为高阻,仅上/下拉有效”功能，除了SWJ接口。

系统复位后，SWJ作为默认功能被选中，为了保证正确进入调试模式，保持SWJ输入引脚非浮空是有必要的，为此复位后对应的SWJ引脚状态设置如下：

- SWDIO：输入上拉
- SWCLK：输入下拉[注1]
- SWO：输出状态

一旦JTAG接口被用户代码释放（见控制位SWJCFG@SYSCFG_SAFR），软件可以把这些I/O口作为普通的I/O口使用。则这些I/O口的状态将由对应的I/O配置寄存器决定，默认情况下所有I/O口都配置为“高阻”。

与SWJ接口不同，外部振荡器接口XTAL1/XTAL2默认是GPIO接口，需在系统启动后由软件开启XTAL接口属性（见控制位OSCCFG@SYSCFG_SAFR），更详细的定义见“SYSCFG”章节。

13.3.2 I/O 控制寄存器

每个I/O口都有4个32位寄存器用于配置工作模式：MODER，OTYPER， ODRVR， PUPDR，TTLEN。

MODER：设置输入输出方向；

OTYPER：设置输出类型，分别为推挽输出、开漏输出(当输出0时，只有N-MOS被打开)；

ODRVR：设置输出驱动能力，分为4档：普通档、弱驱动档、极弱驱动档、超强驱动档（仅少量引脚有此能力）；

PUPDR：设置内部弱上拉和弱下拉，在输入和输出时都能设置上下拉寄存器相应位开启和关闭；

TTLEN：设置输入电平逻辑为CMOS或TTL逻辑。

13.3.3 I/O 数据寄存器

每个GPIO口都有2个16位数据寄存器：IDR，ODR。

IDR：输入数据寄存器，当输入允许时，对输入数据寄存器的读访问可得到I/O状态

ODR：输出数据寄存器，写到输出数据寄存器上的值(ODR)输出到相应的I/O引脚。可以以推挽模式或开漏模式使用输出驱动器。

13.3.4 I/O 位置位和位清除

位置位和清除寄存器（BSRR）是1个32位寄存器，允许对输出寄存器（ODR）的每个位进行独立的置位和清除。

BSRR的低16位是置位操作，高16位是清除操作，允许在单次AHB写操作内对同一组端口的不同I/O口进行同时置位和清除操作，如果是对同一个I/O口同时置位和清除，置位操作具有高优先级。

BSRR的读写可以是字、半字或字节，下面的例子说明了如何用字访问来同时同向/反向切换两个I/O口：

```
；同时同向切换PA.0和PA.1
GPIOA_BSRR = 0x00000003
GPIOA_BSRR = 0x00030000
；同时反向切换PA.0和PA.1
GPIOA_BSRR = 0x00020001
GPIOA_BSRR = 0x00010002
```

如果用半字的方式来实现同时同向切换，方法如下：

```
#define GPIOA_BitSet_ADDR = 0xXX        ；指向GPIO_BSRR_A的基地址（用于set）
#define GPIOA_BitReset_ADDR = 0xXX + 2 ；指向GPIO_BSRR_A的高半字偏移地址（用于reset）
*(u16*)GPIOA_BitSet_ADDR = 0x03
 *(u16*)GPIOA_BitReset_ADDR = 0x03
```

BSRR对ODR的位操作是单次（one-shot）的，不会锁住ODR的相应位，用户仍然可直接对ODR寄存器进行操作。

13.3.5 I/O 锁定机制

锁定机制允许冻结IO配置。可锁定的寄存器包括：OTYPER，ODRVR，PUPDR，TTLEN，AFRL和AFRH。

每一个锁定位LCKy用于锁定一个I/O口，写入锁定位还需要通过向对应的解锁位（LOCK）写入特殊控制字才能写入。

13.3.6 I/O 配置为外部中断/唤醒线

所有端口都可以通过外部中断配置寄存器（EXTI_CFGL/ EXTI_CFGH，在EXTI模块中）来选择连接到外部中断输入线上。为了使用外部中断线，端口需要配置成输入模式[注]。

更多的关于外部中断的信息，参考“外部中断/事件控制器(EXTI)”章节。

注：作为EXTI时建议配置为GPIO输入模式，但如果配置为数字外设功能，也可以触发外部中断。不管配置为GPIO功能或数字外设功能，须输入使能（即IE寄存器为1）才可以触发外部中断。

13.3.7 I/O 复用功能

寄存器AFRL和AFRH定义了端口的复用功能，允许同一端口作为多个复用功能的共享输入输出口，每个端口都有一个16输入的I/O多路复用选择器用于选择需要的复用功能，即同一时刻每个端口只允许一个外设连接。

16个多路选择通道（AF0～AF15）分成3类：

- AF0：GPIO非选功能，高阻（默认）；
- AF1：GPIO基本功能；
- AF2～AF15：数字/模拟外设复用功能；

GPIO非选信道：AF0

这是I/O口上电和复位后的默认配置，所有I/O口上电都配置为GPIO非选功能，高阻，除了JTAG接口。

JTAG接口引脚只有通过SWJCFG@SYSCFG_SAFR关闭SWJ-DP接口后才能作为AF0功能使用。

GPIO基本信道：AF1

这是I/O口通用功能GPIO，需通过寄存器AFRL和AFRH才能配置为I/O基本信道AF1。

JTAG接口引脚只有通过SWJCFG@SYSCFG_SAFR关闭SWJ-DP接口后，默认为AF0功能，再配置为AF1才能正常使用I/O基本功能。

数字/模拟外设复用通道：AF2～AF15

数字外设复用功能由硬件自动设置输入输出方向，但输出类型(OTYPE)、上下拉(PUPDR)、输出驱动(ODRVR)、通道选择(AFRL/AFRH)参数需用户软件配置。

请参考“第4节 引脚定义”的“复用功能映射表”，以获取详细的引脚复用映射。

说明：对于某些外设要求的双向端口，可由硬件自动切换。外设会保证不受发送时反馈到接收信道的数据的影响（假如开启接收信道的话）。

模拟通道用于片上模拟外设，输入输出方向、输出类型和输出驱动配置无意义，但上下拉在某些应用中有特殊用途。

模拟通道会关闭输出缓冲器及输入施密特触发器，相当于断开了数字输入输出通路，可以避免漏电。

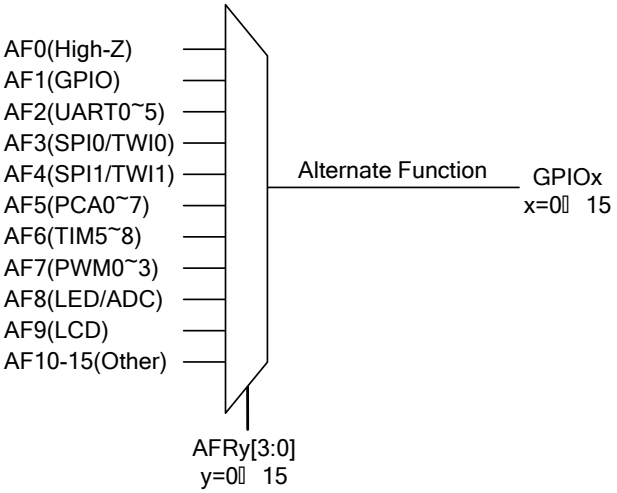


图13-2　　I/O 复用功能多路选择框图

基于上述复用基本功能，SH32F9001_32F9801也支持引脚重映射，即同一外设可映射到不同I/O口上[注]，以适应不同封装形式的资源优化配置，也方便应用布线。具体映射可对照“复用功能映射表”中定义。

注：系统允许同一外设被同时配置到多个不同I/O口上，比如某个外设被配置到PA0口， 并同时被配置到PA10口，则先输出的有效，PA10设置无效。原则上应软件避免将同一功能口配置到不同端口上。

13.3.8　GPIO 输入功能

当I/O端口配置为GPIO输入时：

- 输出缓冲器被禁止
- 当输入使能时，施密特触发输入被激活
- 根据上拉下拉配置的不同，弱上拉和下拉电阻被连接或断开
- 当输入使能时，对输入数据寄存器的读访问可得到I/O状态
- 当输入关闭时，对输入数据寄存器的读访问固定为“0”

下图给出了I/O端口位的GPIO输入配置

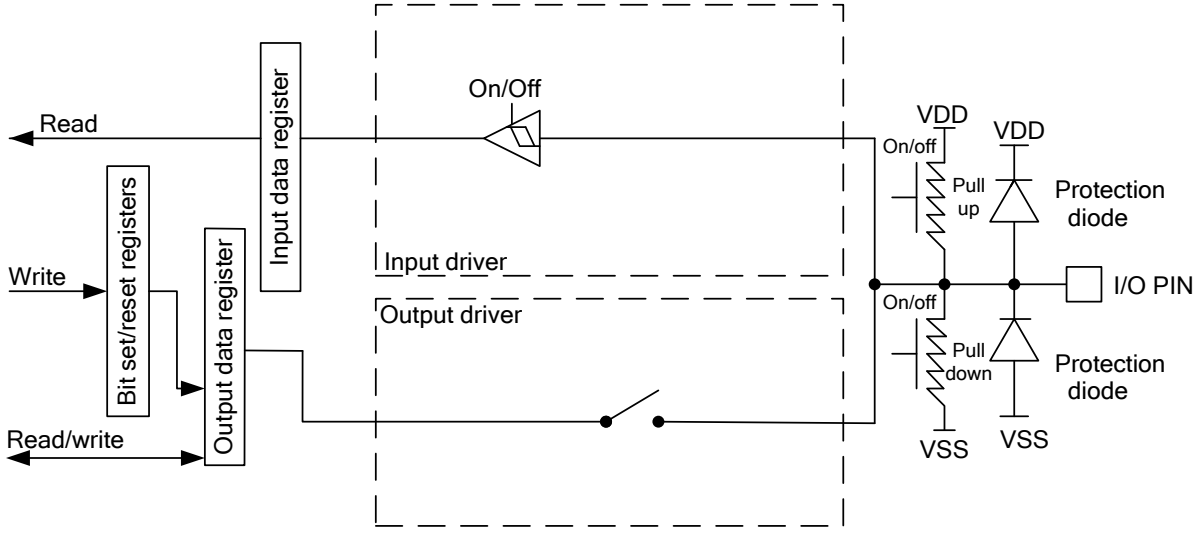


图13-3　　I/O 端口位的 GPIO 输入配置框图

13.3.9 GPIO 输出功能

当I/O端口被配置为GPIO输出时：

- 输出缓冲器被激活
  - 开漏模式：输出寄存器上的'0'激活N-MOS，而输出寄存器上的'1'将端口置于高阻状态(P-MOS不激活)。
  - 推挽模式：输出寄存器上的'0'激活N-MOS，而输出寄存器上的'1'将激活P-MOS。
- 当输入使能时，施密特触发输入被激活
- 根据上拉下拉配置的不同，弱上拉和下拉电阻被连接或断开
- 当输入使能时，对输入数据寄存器的读访问可得到I/O状态
- 当输入关闭时，对输入数据寄存器的读访问固定为“0”
- 对输出数据寄存器的读访问得到最后一次写的值

下图给出了I/O端口位的GPIO输出配置。

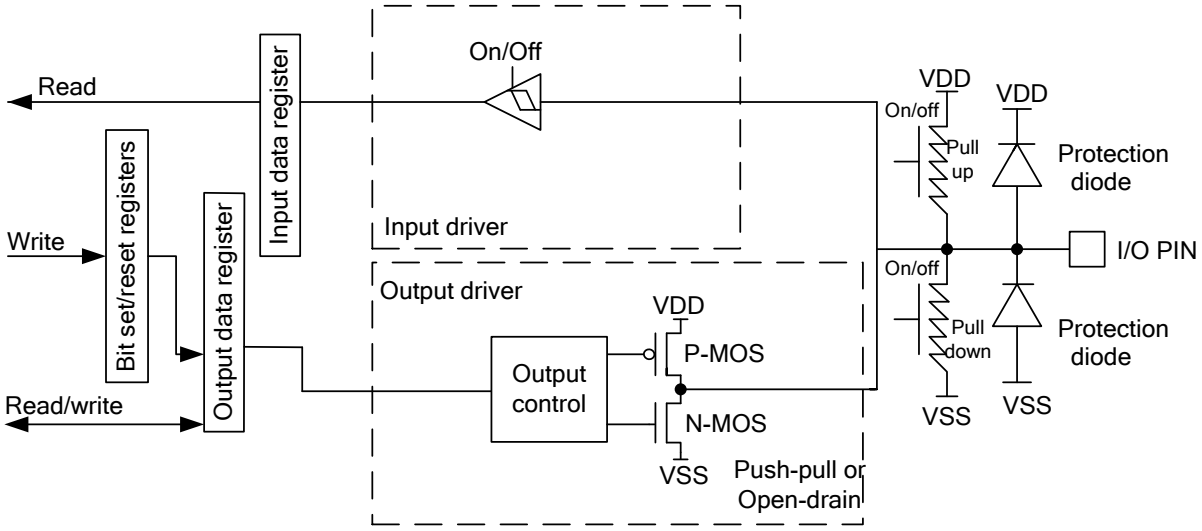


图13-4　　I/O 端口位的 GPIO 输出配置框图

13.3.10 数字复用功能

当I/O端口被配置为数字复用功能时：

- 作为数字复用输入时输出缓冲区被关闭
- 作为数字复用输出时输出缓冲器被打开（断开GPIO输出数据寄存器）
- 数字复用输入或输出属性是由外设决定的，在功能激活时自动配置[注2]
- 无论是输入还是输出，若输入使能，施密特触发输入都被激活
- 根据上拉下拉配置的不同，弱上拉和下拉电阻被连接或断开
- 若输入使能，读输入数据寄存器时可得到I/O口状态
- 当输入关闭时，对输入数据寄存器的读访问固定为“0”

下图示出了I/O端口位数字复用功能配置：

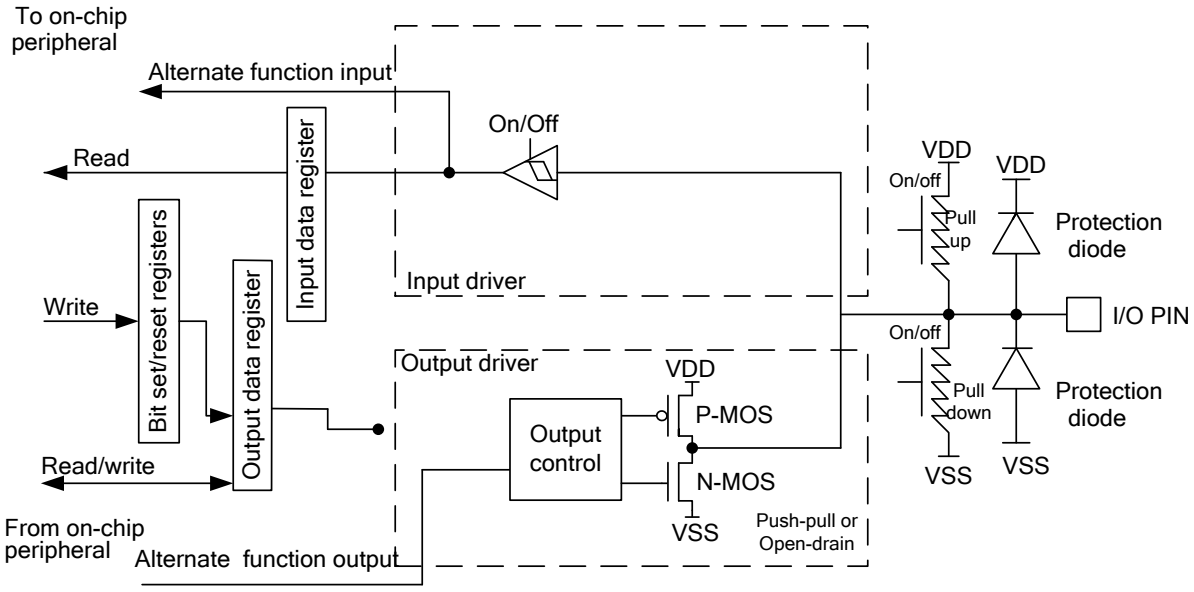


图13-5　　I/O 端口位的数字复用功能配置框图

如果软件把一个I/O脚配置成数字复用输出功能，但是外设没有被激活，此时输出缓冲器被关闭，输出通道呈现高阻状态。当然用户可通过配置上拉或下拉给出一个确定电平。

说明：GPIO输入功能、GPIO输出功能和数字复用功能的I/O输入通道均受IE寄存器控制，默认都是打开的，因此实际上当I/O口设置为GPIO端口（AF1），仅仅是额外打开了I/O输出通道。

13.3.11 模拟复用功能

当I/O端口被配置为模拟复用配置时：

- 输出缓冲器被禁止；
- 自动关闭施密特触发输入，实现了每个模拟I/O引脚上的零消耗，施密特触发输出值被强置为'0'；
- 读取输入数据寄存器时数值为'0'；
- 允许开启弱上拉和下拉；

下图示出了I/O端口位的模拟复用配置：

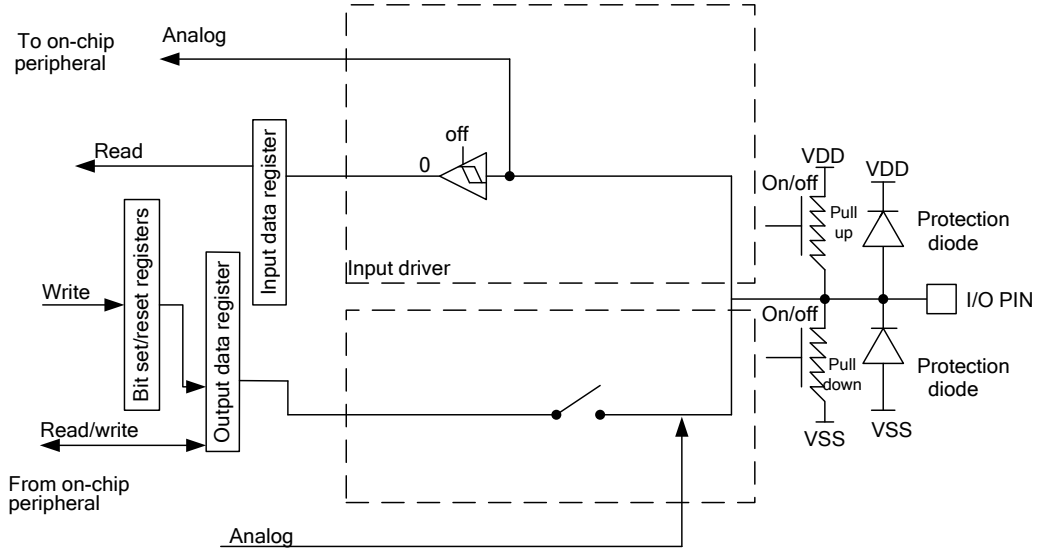


图13-6　　I/O 端口位的模拟复用功能配置框图

13.3.12 I/O 引脚的基本应用

从应用上来看，I/O引脚只有2种功能：GPIO功能、外设功能（模拟/数字）。使用数字外设功能时，需要根据不同外设类型对端口位进行配置，包括推挽/开漏、上/下拉、输入允许/禁止和输出驱动能力。

外设功能（模拟/数字）

- 模拟外设
  - 配置成模拟信道：AFRx = 1000b（AF8）
  - 关闭上/下拉电阻：PUPDRx = 00b （一般应用默认不开启）
- 数字外设
  - 配置成数字信道：AFRx = 0010b（AF2）～1111b（AF15）
  - 选择推挽/开漏、上/下拉和输出驱动能力：OTYPERx、PUPDRx、ODRVRx

数字外设的一般配置规则是：

1. 对于数字复用输入口，端口自动切换到输入，用户可配置输入浮空、上拉或下拉，输入引脚由外部驱动。
2. 对于数字复用输出口，端口自动切换到输出，用户可配置推挽或开漏、驱动能力、上下拉。显然，此时引脚和数据输出寄存器断开，并和片上外设的输出信号连接。
3. 对于数字复用双向口，端口自动在输入与输出间切换，用户可配置推挽或开漏、驱动能力、上下拉。
4. 无论是数字复用输入/输出/双向口，输入通路均受IE寄存器控制；对于数字复用输入/双向口，输入通路需打开（即IE=1）；对于数字复用输出口，输入通路是否打开，根据应用需求灵活配置。

一种推荐的外设配置流程为：

（1） 根据复用口的输入输出属性，配置好上下拉（如有必要），推挽或开漏（输出），驱动能力（输出）；

（2） 切换到目标外设通道；

（3） 根据需求，打开或关闭目标外设的输入通路；

（4） 开启对应外设（某些设备在开启后要求把上拉或下拉关闭以免影响信号），设备工作；

（5） 关闭外设时如有必要再配置上下拉以给出确定电平（关闭外设时输出通道会自动切换到高阻）。

实际应用中有些外设（2）和（4）顺序也可以交换，即先使能外设，再切换目标外设通道。

GPIO功能（输入/输出）

- 配置方向
  - 输入：MODERx = 0b
  - 输出：MODERx = 1b
- 对于输入
  - 配置上/下拉电阻：PUPDRx
  - 配置CMOS/TTL逻辑电平：TTLEN_x
  - 配置输入允许：IE
- 对于输出
  - 选择推挽/开漏、上/下拉和输出驱动：OTYPERx、PUPDRx、ODRVRx

以串口模块UART3举例说明： UART3为数字外设，配置如下：

复用引脚	功能定义	GPIO配置
PA0/TXD3	异步全双工模式，数据发送	推挽输出，关闭上下拉，驱动能力取默认， AFR0 = 0010b（AF2）
	同步半双工模式，时钟输出	推挽输出，关闭上下拉，驱动能力取默认， AFR0 = 0010b（AF2）
PA1/RXD3	异步全双工模式，数据接收	浮空输入或带上拉输入，输入允许 AFR1 = 0010b（AF2）
	同步半双工模式，双向数据	推挽输出与浮空输入自动切换，关闭上下拉或开启上拉，输入允许，驱动能力取默认。 AFR1 = 0010b（AF2）

13.4 寄存器

GPIOx_CFG 模块寄存器列表 (基地址: 0x4004 0800)

地址	寄存器名	说明
0x4004 0800 + 0x20*x	LCKR	端口x配置锁定寄存器 (端口A～E对应x:0~4)
0x4004 0804+ 0x20*x	IE	端口x输入使能配置寄存器 (端口A～E对应x:0~4)
0x4004 0808+ 0x20*x	OTYPER	端口x输出方式配置寄存器 (端口A～E对应x:0~4)
0x4004 080C+ 0x20*x	ODRVR	端口x输出驱动能力配置寄存器 (端口A～E对应x:0~4)
0x4004 0810+ 0x20*x	PUPDR	端口x上下拉配置寄存器 (端口A～E对应x:0~4)
0x4004 0814+ 0x20*x	TTLEN	端口x部分引脚TTL电平选择寄存器 (端口A～E对应x:0~4)
0x4004 0818+ 0x20*x	AFRL	端口x复用功能寄存器低位 (端口A～E对应x:0~4)
0x4004 081C+ 0x20*x	AFRH	端口x复用功能寄存器高位 (端口A～E对应x:0~4)

13.4.1 端口 x 配置锁定寄存器 (GPIOx_CFG_LCKR)

偏移地址: 0x0000 + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCKy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LOCK[15:0]**	**LCK控制位操作解锁位 (Port x reset bit)** 这些位是只写，读取这些位返回0x0000。 0x5AA5： 解锁 其他：锁定
15 ~ 0	**LCKy(y=15~0)**	**端口x锁定位(Port x lock bit)** 这些位可读可写，写入这些位必须要满足LOCK解锁条件。（读取不需要） 0: 端口位配置未锁定 1: 端口位配置锁定

13.4.2 端口 x 输入使能配置寄存器 (GPIOx_CFG_IE)

偏移地址: 0x0004 + 0x20*x

复位值: 0x0000 FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IEy(y=15~0)															
RW															
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	IEy(y=15~0)	端口x输入使能配置 (Port x Input Enable Control configuration) 由软件配置 0：输入关闭 1：输入使能（复位状态）

13.4.3 端口 x 输出方式配置寄存器 (GPIOx_CFG_OTYPER)

偏移地址: 0x0008 + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
OTy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	OTy(y=15~0)	端口x输出方式配置 (Port x output type configuration) 由软件配置 0：推挽输出 （复位状态） 1：开漏输出

13.4.4 端口 x 输出驱动能力配置寄存器 (GPIOx_CFG_ODRVR)

偏移地址: 0x000C + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
ODRVR15		ODRVR14		ODRVR13		ODRVR12		ODRVR11		ODRVR10		ODRVR9		ODRVR8	
RW		*RW*		*RW*		*RW*		*RW*		*RW*		*RW*		*RW*	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ODRVR7		ODRVR6		ODRVR5		ODRVR4		保留	ODR VR3	保留	ODR VR2	保留	ODR VR1	保留	ODR VR0
RW		*RW*		*RW*		*RW*		-	*RW*	-	*RW*	-	*RW*	-	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 30	**ODRVR15**	**端口x输出驱动能力配置(Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB15是有效的） 11：输出1强驱动，输出0弱驱动（仅PB15是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
29 ~ 28	**ODRVR14**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB14是有效的） 11：输出1强驱动，输出0弱驱动（仅PB14是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
27 ~ 26	**ODRVR13**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB13是有效的） 11：输出1强驱动，输出0弱驱动（仅PB13是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
25 ~ 24	**ODRVR12**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB12是有效的） 11：输出1强驱动，输出0弱驱动（仅PB12是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
23 ~ 22	**ODRVR11**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB11是有效的） 11：输出1强驱动，输出0弱驱动（仅PB11是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
21 ~ 20	**ODRVR10**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB10是有效的） 11：输出1强驱动，输出0弱驱动（仅PB10是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
19 ~ 18	**ODRVR9**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB9是有效的） 11：输出1强驱动，输出0弱驱动（仅PB9是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
17 ~ 16	**ODRVR8**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB8是有效的） 11：输出1强驱动，输出0弱驱动（仅PB8是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
15 ~ 14	**ODRVR7**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB7是有效的）

		11：输出1强驱动，输出0弱驱动（仅PB7是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
13 ~ 12	**ODRVR6**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB6是有效的） 11：输出1强驱动，输出0弱驱动（仅PB6是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
11 ~ 10	**ODRVR5**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB5是有效的） 11：输出1强驱动，输出0弱驱动（仅PB5是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
9 ~ 8	**ODRVR4**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 00：普通状态（复位值） 01：弱驱动 10: 输出0强驱动，输出1弱驱动（仅PB4是有效的） 11：输出1强驱动，输出0弱驱动（仅PB4是有效的） 详细驱动能力参数见DataSheet “电气特性”章节
7	**Reserved**	**-**
6	**ODRVR3**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 0：普通状态（复位值） 1：弱驱动 详细驱动能力参数见DataSheet “电气特性”章节
5	**Reserved**	**-**
4	**ODRVR2**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 0：普通状态（复位值） 1：弱驱动 详细驱动能力参数见DataSheet “电气特性”章节
3	**Reserved**	**-**
2	**ODRVR1**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 0：普通状态（复位值） 1：弱驱动 详细驱动能力参数见DataSheet “电气特性”章节
1	**Reserved**	**-**
0	**ODRVR0**	**端口x输出驱动能力配置 (Port x output driver configuration) 由软件配置** 0：普通状态（复位值） 1：弱驱动 详细驱动能力参数见DataSheet “电气特性”章节

13.4.5 端口 x 上下拉配置寄存器 (GPIOx_CFG_PUPDR)

偏移地址: 0x0010 + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PLDR15	PHDR15	PLDR14	PHDR14	PLDR13	PHDR13	PLDR12	PHDR12	PLDR11	PHDR11	PLDR10	PHDR10	PLDR9	PHDR9	PLDR8	PHDR8
RW	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PLDR7	PHDR7	PLDR6	PHDR6	PLDR5	PHDR5	PLDR4	PHDR4	PLDR3	PHDR3	PLDR2	PHDR2	PLDR1	PHDR1	PLDR0	PHDR0
RW	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31	**PLDR15**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
30	**PHDR15**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
29	**PLDR14**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
28	**PHDR14**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
27	**PLDR13**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
26	**PHDR13**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
25	**PLDR12**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
24	**PHDR12**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
23	**PLDR11**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
22	**PHDR11**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
21	**PLDR10**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
20	**PHDR10**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
19	**PLDR9**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
18	**PHDR9**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
17	**PLDR8**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
16	**PHDR8**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
15	**PLDR7**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
14	**PHDR7**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置**

		0：关闭上拉（复位状态） 1：使能上拉
13	**PLDR6**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
12	**PHDR6**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
11	**PLDR5**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
10	**PHDR5**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
9	**PLDR4**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
8	**PHDR4**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
7	**PLDR3**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
6	**PHDR3**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
5	**PLDR2**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
4	**PHDR2**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
3	**PLDR1**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
2	**PHDR1**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉
1	**PLDR0**	**端口x下拉配置 (Port x pull-down configuration) 由软件配置** 0：关闭下拉（复位状态） 1：使能下拉
0	**PHDR0**	**端口x上拉配置 (Port x pull-up configuration) 由软件配置** 0：关闭上拉（复位状态） 1：使能上拉

13.4.6 端口 x 部分引脚 TTL 电平选择寄存器 (GPIOx_CFG_TTLEN)

偏移地址: 0x0014 + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PX15	PX14	PX13	PX12	PX11	PX10	PX9	PX8	PX7	PX6	PX5	PX4	PX3	PX2	PX1	PX0
RW	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15	PX15	**PX15引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
14	PX14	**PX14引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
13	PX13	**PX13引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
12	PX12	**PX12引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
11	PX11	**PX11引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
10	PX10	**PX10引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
9	PX9	**PX9引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
8	PX8	**PX8引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
7	PX7	**PX7引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
6	PX6	**PX6引脚TTL/CMOS输入选择** 0 : 选择CMOS输入 1: 选择TTL输入

		TTL参数见“电气特性”，在停机模式下不做任何处理。
5	**PX5**	**PX5引脚TTL/CMOS输入选择** 0：选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
4	**PX4**	**PX4引脚TTL/CMOS输入选择** 0：选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
3	**PX3**	**PX3引脚TTL/CMOS输入选择** 0：选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
2	**PX2**	**PX2引脚TTL/CMOS输入选择** 0：选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
1	**PX1**	**PX1引脚TTL/CMOS输入选择** 0：选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。
0	**PX0**	**PX0引脚TTL/CMOS输入选择** 0：选择CMOS输入 1: 选择TTL输入 TTL参数见“电气特性”，在停机模式下不做任何处理。

13.4.7 端口 x 复用功能寄存器低位(GPIOx_CFG_AFRL)

(Port 复用信息参考“IOMUX”表，在 Reg 中不再具体说明)

偏移地址: 0x0018 + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AFR7[3:0]				AFR6[3:0]				AFR5[3:0]				AFR4[3:0]			
RW				RW				RW				RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
AFR3[3:0]				AFR2[3:0]				AFR1[3:0]				AFR0[3:0]			
RW				RW				RW				RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 28	**AFR7[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR7功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8

		1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
27 ~ 24	**AFR6[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR6功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
23 ~ 20	**AFR5[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR5功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
19 ~ 16	**AFR4[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR4功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8

		1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
15 ~ 12	**AFR3[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR3功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
11 ~ 8	**AFR2[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR2功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
7 ~ 4	**AFR1[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR1功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8

		1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
3 ~ 0	**AFR0[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR0功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15

13.4.8 端口 x 复用功能寄存器高位 (GPIOx_CFG_AFRH)

(Port 复用信息参考“IOMUX”表，在 Reg 中不再具体说明)

偏移地址: 0x001C + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
AFR15[3:0]				AFR14[3:0]				AFR13[3:0]				AFR12[3:0]			
RW				RW				RW				RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
AFR11[3:0]				AFR10[3:0]				AFR9[3:0]				AFR8[3:0]			
RW				RW				RW				RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 28	**AFR15[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR15功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
27 ~ 24	**AFR14[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR14功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
23 ~ 20	**AFR13[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR13功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11

		1100: AF12 1101: AF13 1110: AF14 1111: AF15
19 ~ 16	**AFR12[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR12功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
15 ~ 12	**AFR11[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR11功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
11 ~ 8	**AFR10[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR10功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
7 ~ 4	**AFR9[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR9功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8

		1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15
3 ~ 0	**AFR8[3:0]**	**端口x复用功能选择位 (Port x alternate function selection) 由软件配置** AFR8功能选择： 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7 1000: AF8 1001: AF9 1010: AF10 1011: AF11 1100: AF12 1101: AF13 1110: AF14 1111: AF15

GPIOx 模块寄存器列表 (基地址:0x0x4004 0A00)

地址	寄存器名	说明
0x4004 0A00 + 0x20*x	MODER	端口**x**模式寄存器 (x:0~4)
0x4004 0A04 + 0x20*x	IDR	端口**x**输入数据寄存器 (x:0~4)
0x4004 0A08 + 0x20*x	ODR	端口**x**输出数据寄存器 (x:0~4)
0x4004 0A0C + 0x20*x	BSRR	端口**x**位清除/置位寄存器 (x:0~4)

13.4.9 端口 x 模式寄存器 (GPIOx_MODER)

偏移地址: 0x0000+ 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
MODERy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**MODERy(y=15~0)**	**端口x模式配置 (Port x mode configuration) 由软件配置** 0：输入模式（复位状态） 1：输出模式

13.4.10 端口 x 输入数据寄存器 (GPIOx_IDR)

偏移地址: 0x0004 + 0x20*x

复位值: 0x0000 XXXX

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IDRy(y=15~0)															
RO															
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**IDRy(y=15~0)**	**端口x输入数据寄存器 (Port x input data register)** 该寄存器只读，存储端口读到的电平值。

13.4.11 端口 x 输出数据寄存器 (GPIOx_ODR)

偏移地址: 0x0008 + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ODRy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**ODRy(y=15~0)**	**端口x输出数据寄存器 (Port x output data register)** 该寄存器可以软件读写。通过后面的BSRR寄存器，ODR能够独立的被按位set和reset，以实现原子位操作。

13.4.12 端口 x 位清除/置位寄存器 (GPIOx_BSRR)

偏移地址: 0x000C + 0x20*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
BRy(y=15~0)															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
BSy(y=15~0)															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**BRy(y=15~0)**	**端口x位清除 (Port x reset bit)** 这些位是只写，读取这些位返回0x0000。 0: 无影响 1: 清除相应的ODR15位 注：如果同时设置同一个Port口的BR15和BS15，则BS15优先级高。
15 ~ 0	**BSy(y=15~0)**	**端口x位置位 (Port x set bit)** 这些位是只写，读取这些位返回0x0000。 0: 无影响 1: 置位相应的ODR15位 注：如果同时设置同一个Port口的同一位BR15和BS15，则BS15优先级高。

14. 系统配置模块(SYSCFG)

此模块用于系统层面的辅助模块和系统配置。包括SRAM检测模块定义，引脚功能重映像、调试接口等相关配置。SYSCFG挂在AHB总线上，有独立的使能与复位控制。

14.1 系统配置相关

14.1.1 SWJ 和 XTAL 引脚复用功能定义

调试器接口为两线SW接口（SWDIO和SWCLK），默认作为SWJ接口，通过SWJCFG@SYSCFG_SAFR可以配置为GPIO口。一旦配置完成，只能通过复位回到调试接口状态。

外部振荡器引脚XTAL1和XTAL2默认用作GPIO接口，通过OSCCFG@SYSCFG_SAFR可配置为外部振荡器接口或外灌时钟接口，一旦配置完成，只能通过复位回到默认接口状态。

SWJ接口和XTAL接口在作为非GPIO接口使用时，GPIO的配置参数无效（可以写入但无效果），包含输入输出方向的配置也无效。

说明：XTAL引脚作为振荡器接口或外灌时钟接口时，硬件上需屏蔽上下拉，否则由于软件误配置会造成振荡器停振的严重后果。

14.1.2 CRAM 锁定选项

一旦CRAM设置为锁定，将变为只读，此时对CRAM进行写入将触发总线Fault。

CRAM锁定以2K为扇区，通过CRAMLCKy(y=0～7)@SYSCFG_CRAMLOCK开启。具体参见“CRAM”章节。

14.1.3 调试接口寄存器

DBGCR为调试接口寄存器，此处仅定义了调试模式下看门狗状态以及低功耗模式下的调试选项，具体参见“调试接口”章节。

14.2 寄存器

SYSCFG 模块寄存器列表 (基地址: 0x4004 3800)

地址	寄存器名	说明
0x4004 3800	PWRCR	电源控制寄存器
0x4004 3804	PWRSR	电源状态寄存器
0x4004 3808	SAFR	系统配置寄存器
0x4004 380C	CRAMLOCK	CRAM扇区锁定控制寄存器
0x4004 3810	DBGCR	调试接口控制寄存器

14.2.1 电源控制寄存器 (SYSCFG_PWRCR)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留				LVREN	VLVR[1:0]		BODEN	BODIE	BODMD[1:0]		VBOD[4:0]				
-				*RW*	*RW*		*RW*	*RW*	*RW*		*RW*				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31~12	保留	-
11	LVREN	**LVR使能位** **0：禁止低电压复位功能** **1：允许低电压复位功能** 说明：LVR一旦开启，开启后关闭无效，只有在复位时才能关闭。
10 ~ 9	VLVR[1:0]	**LVR复位电压档位设置位** 00: 4.1 V 01: 3.7 V 10: 2.8 V 11: 2.3 V
8	BODEN	**BOD允许位** 0：禁止掉电检测 1：允许掉电检测
7	BODIE	**BOD中断使能位** 0：禁止BOD中断 1：允许BOD中断
6 ~ 5	BODMD[1:0]	**BOD模式选择控制位** 00：仅当VDD电压从小于BOD门限到大于BOD门限时，BODIF标志置1 01：仅当VDD电压从大于BOD门限到小于BOD门限时，BODIF标志置1 10：当VDD电压从大于门限到小于门限，或从小于门限到大于门限，BODIF标志都置1 11：保留
4 ~ 0	VBOD[4:0]	**BOD门限电压设置位** 00000: 2.40 V 00001: 2.50 V 00010: 2.60 V 00011: 2.70 V 00100: 2.80 V 00101: 2.90 V 00110: 3.00 V 00111: 3.10 V 01000: 3.20 V 01001: 3.30 V 01010: 3.40 V 01011: 3.50 V 01100: 3.60 V 01101: 3.70 V 01110: 3.80 V 01111: 3.90 V 10000: 4.00 V 10001: 4.10 V 10010: 4.20 V 10011: 4.30 V 其他: 2.40V

14.2.2 电源状态寄存器 (SYSCFG_PWRSR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留														BODF	BODIF
-														*RO*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 2	保留	-
1	**BODF**	**BOD标志位** 0：当前电压高于在VBOD[3:0]中设置的BOD电压 1：当前电压低于在VBOD[3:0]中设置的BOD电压
0	**BODIF**	**BOD中断请求标志** 0：无中断挂起，清除挂起 1：中断挂起

14.2.3 系统配置寄存器 (SYSCFG_SAFR)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留							IEN_CSM	IEN_BOD	IEN_EXTI0	SWJCFG	保留		OSCXCFG	OSCCFG[1:0]	
-							RW	RW	RW	RW1t	-		RW1t	RW1t	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LOCK[15:0]**	**此寄存器操作解锁位 (Control bit Lock register)** 这些位是只写，读取这些位返回0x00。 0x5AA5： 解锁 其他： 锁定 注：写入此寄存器需配合解锁控制，读此寄存器不需要解锁。
15 ~ 9	**保留**	**-**
8	**IEN_CSM**	**CSM引起NMI中断使能位**
7	**IEN_BOD**	**BOD引起NMI中断使能位**
6	**IEN_EXTI0**	**EXTI0引起NMI中断使能位**
5	**SWJCFG**	**串行线JTAG引脚复用功能定义** 这些位只可由软件写(读这些位，将返回未定义的数值)，用于配置SWJ和跟踪复用功能的I/O口。SWJ(串行线JTAG)支持SWD访问Cortex M3的调试端口。系统复位后的默认状态是启用SWJ但没有跟踪功能，这种状态下可以通过JTMS/JTCK脚上的特定信号选择JTAG或SW(串行线)模式。 为释放部分或全部调试口用做普通I/O口，用户软件在复位后可设置SWJCFG。 0：完全SWJ(SW-DP)，SWJ调试口无法复用为普通I/O。 1：关闭SW-DP；SWO可复用为普通I/O，但SWDIO，SWCLK在正常运行状态下可复用为普通I/O，在调试模式状态下仍强制作为调试接口。 注：此控制位一旦设置，只有reset才能修改；SW-DP包含SWDIO/SWCLK/SWO引脚。
4 ~ 3	**保留**	**-**
2	**OSCXCFG**	**XTALX1/XTALX2引脚复用功能定义** 0： XTALX1/XTALX2引脚不复用振荡器功能（默认） 1：外部振荡器接口 注：1、此控制位一旦设置，只有系统复位(NRST/IWDT/WWDT/软件复位)、电源复位(POR/LVR)和SYSCFG Reset才能修改此寄存器。
1 ~ 0	**OSCCFG[1:0]**	**XTAL1/XTAL2引脚复用功能定义** 00： XTAL1/XTAL2引脚不复用振荡器功能（默认） 01：外部振荡器接口（晶振和陶振） 10：XTAL1引脚上外灌时钟，XTAL2引脚不复用振荡器功能 11：保留 注：1、此控制位一旦设置，只有系统复位(NRST/IWDT/WWDT/软件复位)、电源复位(POR/LVR)和SYSCFG Reset才能修改此寄存器。

14.2.4 CRAM 扇区锁定控制寄存器 (SYSCFG_CRAMLOCK)

偏移地址: 0x000C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								CRAMLCKy(y=7~0)							
-								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LOCK[15:0]**	**CRAMLCKy控制位操作解锁位** 这些位是只写，读取这些位返回0x0000。 0x5AA5： 解锁 其他: 锁定
15 ~ 8	**保留**	**-**
7 ~ 0	**CRAMLCKy(y=7~0)**	**CRAM sector 锁定位(CRAM sector lock bit)** 这些位可读可写，写入这些位必须要满足LOCK解锁条件。（读取不需要） 0: CRAM sector y未锁定 1: CRAM sector y锁定 注： CRAM sector起始地址为0x1000 0000，每一个sector为2K字节，总共16K字节。

14.2.5 调试接口控制寄存器 (SYSCFG_DBGCR)

偏移地址: 0x0010

复位值: 0x0000 0020

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留		DBG_TWI	DBG_SPI	DBG_UART	DBG_PWM	DBG_TIM	DBG_PCA	DBG_WWDT	DBG_IWDT	DBG_DMA	DBG_LED	DBG_LCD	保留	DBG_STOP	DBG_SLEEP
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

位	符号	说明
31 ~ 16	**LOCK[15:0]**	**此寄存器操作解锁位 (Control bit Lock register)** 这些位是只写，读取这些位返回0x0000。 0x5AA5： 解锁 其他: 锁定 注：写入此寄存器需配合解锁控制，读此寄存器不需要解锁。
15 ~ 14	**保留**	**-**
13	**DBG_TWI**	**当内核进入调试状态时TWI状态控制** 0：TWI停止工作 1：TWI仍然正常工作
12	**DBG_SPI**	**当内核进入调试状态时SPI状态控制** 0：SPI停止工作 1：SPI仍然正常工作
11	**DBG_UART**	**当内核进入调试状态时UART状态控制** 0：UART停止工作

		1：UART仍然正常工作
10	**DBG_PWM**	**当内核进入调试状态时PWM状态控制** 0：PWM停止工作 1：PWM仍然正常工作
9	**DBG_TIM**	**当内核进入调试状态时TIM5-TIM8模块状态控制** 0：TIM5-TIM8模块停止工作 1：TIM5-TIM8模块仍然正常工作
8	**DBG_PCA**	**当内核进入调试状态时PCA状态控制** 0：PCA停止工作 1：PCA仍然正常工作
7	**DBG_WWDT**	**当内核进入调试状态时调试窗口看门狗状态控制** 0：窗口看门狗计数器停止工作 1：窗口看门狗计数器仍然正常工作
6	**DBG_IWDT**	**当内核进入调试状态时看门狗状态控制** 0：看门狗计数器停止工作 1：看门狗计数器仍然正常工作
5	**DBG_DMA**	**当内核进入调试状态时DMA状态控制** 0：DMA停止工作 1：DMA仍然正常工作
4	**DBG_LED**	**当内核进入调试状态时LED状态控制** 0：LED停止工作 1：LED仍然正常工作
3	**DBG_LCD**	**当内核进入调试状态时LCD状态控制** 0：LCD停止工作 1：LCD仍然正常工作
2	**保留**	**-**
1	**DBG_STOP**	**调试停机模式** 0：(FCLK开，HCLK开)在停机模式时，FCLK和HCLK时钟由内部RC振荡器提供。当退出停机模式时，软件必需重新配置时钟系统启动PLL，晶振等(与配置此比特位为0时的操作一样) 1：(FCLK关，HCLK关)在停机模式时，时钟控制器禁止一切时钟(包括HCLK和FCLK)。当从停机模式退出时，时钟的配置和复位之后的配置一样(微控制器由8MHz的内部RC振荡器(HSI)提供时钟)。因此，软件必需重新配置时钟控制系统启动PLL，晶振等
0	**DBG_SLEEP**	**调试睡眠模式** 0：(FCLK开，HCLK开)在睡眠模式时，FCLK和HCLK时钟都由原先配置好的系统时钟提供。 1：(FCLK开，HCLK关)在睡眠模式时，FCLK由原先已配置好的系统时钟提供，HCLK则关闭。由于睡眠模式不会复位已配置好的时钟系统，因此从睡眠模式退出时，软件不需要重新配置时钟系统。

15. 中断和事件

15.1 可嵌套中断向量控制器（NVIC）

15.1.1 简介

NVIC是Cortex-M3的主要组成部分，它与CPU紧密结合，降低了中断延时，并允许新进中断可以得到高效处理。

NVIC还可以处理系统异常：包括复位(Reset), 非屏蔽中断(NMI), 硬件错误(Hard Fault), 内存错误(MemManage Fault), 总线错误(Bus Fault), 使用错误(Usage Fault),调用系统服务(SVCall), 调试监控(Debug Monitor), 挂起服务(PendSV), 及系统嘀嗒(Systick)。

15.1.2 主要特性

- 嵌套向量控制器是ARM Cortex-M3的一个组成部分
- 高度耦合的中断控制器提供了低中断延时。
- 控制处理系统异常及外设中断
- SH32F支持 41 个可屏蔽中断(不包含16个Cortex-M3的中断)。
- 16个可编程中断优先级（使用4位中断优先级）
- 低延迟的异常和中断处理
- 可复位的向量表
- 中断可屏蔽
- 软件中断

可嵌套中断支持

可嵌套中断支持的作用范围很广，覆盖了所有的外部中断和绝大多数系统异常。外在表现是，这些异常都可以被赋予不同的优先级。当前优先级被存储在 xPSR 的专用字段中。当一个异常发生时，硬件会自动比较该异常的优先级是否比当前的异常优先级更高。如果发现来了更高优先级的异常，处理器就会中断当前的中断服务例程（或者是普通程序），而服务新来的异常——即立即抢占。

向量中断支持

当开始响应一个中断后，CM3 会自动定位一张向量表，并且根据中断号从表中找出 ISR 的入口地址，然后跳转过去执行。这么一来，中断延迟时间大为缩短。 动态优先级调整支持软件可以在运行时期更改中断的优先级。如果在某 ISR 中修改了自己所对应中断的优先级，而且这个中断又有新的实例处于悬起中（pending），也不会自己打断自己，从而没有重入(reentry)风险。

中断延迟大大缩短

Cortex-M3 为了缩短中断延迟，引入了好几个新特性。包括自动的现场保护和恢复，以及其它的措施，用于缩短中断嵌套时的 ISR 间延迟。详情请见后面关于“咬尾中断”和“晚到中断”的讲述。

中断可屏蔽

既可以屏蔽优先级低于某个阈值的中断/异常 (设置 BASEPRI 寄存器)，也可以全体封杀(设置PRIMASK和FAULTMASK寄存器)。这是为了让时间关键（time-critical）的任务能在死线(deadline，或最后期限)到来前完成，而不被干扰。

15.1.3 中断汇总

SH32F9001_32F9801中断的所有中断源与中断控制/状态信息汇总如下表：

NMI源包含CSM、EXTI0、BOD，有单独的控制位（见SYSCFG_SAFR寄存器）。EXTI0可以选择PA0、PB0、PC0、PD0、PE0作为输入。EXTI0除了NMI入口，还有独立的EXTI0入口，因此产生NMI时EXTI0中断仍需要处理（可以同时开启），BOD也有两个中断入口，而CSM只有NMI中断入口。

9 Exception + 43 HWINT

中断 ID	中断源	向量 地址	中断屏蔽位	中断使能位	中断标志位
-15	Reset	0x0004			
-14	NMI	0x0008	SYSCFG_SAFR_IEN_CSM SYSCFG_SAFR_IEN_BOD SYSCFG_SAFR_IEN_EXTI0		RCC_CISTR_CSMPLLF RCC_CISTR_CSMHSEF SYSCFG_PWRSR_BODIF EXTI_PR_Pry
-13	HardFault	0x000C	FAULTMASK		
-12	MemManage	0x0010	PRIMASK	SCB_SHCSR_MEMFAULTENA	SCB_SHCSR_MEMFAULTPENDED
-11	BusFault	0x0014	PRIMASK	SCB_SHCSR_BUSFAULTENA	SCB_SHCSR_BUSFAULTPENDED
-10	UsageFault	0x0018	PRIMAS0K	SCB_SHCSR_USGFAULTENA	SCB_SHCSR_USGFAULTPENDED
-9	(reserved)	0x001C			
-8	(reserved)	0x0020			
-7	(reserved)	0x0024			
-6	(reserved)	0x0028			
-5	SVCall	0x002C	PRIMASK		SCB_SHCSR_SVCALLPENDED
-4	DebugMonitor	0x0030	PRIMASK	DEMCR	SCB_SHCSR_MONITORACT
-3	(reserved)	0x0034			
-2	PendSV	0x0038	PRIMASK		SCB_SHCSR_PENDSVACT
-1	SysTick	0x003C	PRIMASK	SysTick_CTRL_TICKINT	SCB_SHCSR.SYSTICKACT
0	WWDT	0x0040	NVIC->ISER[0].0	WWDT_CR_WWDTIE	WWDT_SR_WWDTIF
1	BOD	0x0044	NVIC->ISER[0].1	SYSCFG_PWRCSR_BODIE	SYSCFG_PWRCSR_BODIF
2	RCC	0x0048	NVIC->ISER[0].2	RCC_CIENR_PLLRDYIE RCC_CIENR_HSERDYIE RCC_CIENR_LSERDYIE	RCC_CISTR_PLLRDYIF RCC_CISTR_HSERDYIF RCC_CISTR_LSERDYIF
3	EXTI0	0x004C	NVIC->ISER[0].3	EXTI_IMR_IMR0	EXTI_PR_PR0
4	EXTI1	0x0050	NVIC->ISER[0].4	EXTI_IMR_IMR1	EXTI_PR_PR1
5	EXTI2	0x0054	NVIC->ISER[0].5	EXTI_IMR_IMR2	EXTI_PR_PR2
6	EXTI3	0x0058	NVIC->ISER[0].6	EXTI_IMR_IMR3	EXTI_PR_PR3
7	EXTI4	0x005C	NVIC->ISER[0].7	EXTI_IMR_IMR4	EXTI_PR_PR4
8	DMA_CH0	0x0060	NVIC->ISER[0].8	DMA_CCR0_TEIE DMA_CCR0_BEIE DMA_CCR0_HTIE DMA_CCR0_TCIE	DMA_IFSR_TEIF0 DMA_IFSR_BEIF0 DMA_IFSR_HTIF0 DMA_IFSR_TCIF0
9	DMA_CH1	0x0064	NVIC->ISER[0].9	DMA_CCR1_TEIE DMA_CCR1_BEIE DMA_CCR1_HTIE DMA_CCR1_TCIE	DMA_IFSR_TEIF1 DMA_IFSR_BEIF1 DMA_IFSR_HTIF1 DMA_IFSR_TCIF1

10	DMA_CH2_7	0x0068	NVIC->ISER[0].10	DMA_CCR2_TEIE DMA_CCR2_BEIE DMA_CCR2_HTIE DMA_CCR2_TCIE DMA_CCR3_TEIE DMA_CCR3_BEIE DMA_CCR3_HTIE DMA_CCR3_TCIE DMA_CCR4_TEIE DMA_CCR4_BEIE DMA_CCR4_HTIE DMA_CCR4_TCIE DMA_CCR5_TEIE DMA_CCR5_BEIE DMA_CCR5_HTIE DMA_CCR5_TCIE DMA_CCR6_TEIE DMA_CCR6_BEIE DMA_CCR6_HTIE DMA_CCR6_TCIE DMA_CCR7_TEIE DMA_CCR7_BEIE DMA_CCR7_HTIE DMA_CCR7_TCIE	DMA_IFSR_TEIF2 DMA_IFSR_BEIF2 DMA_IFSR_HTIF2 DMA_IFSR_TCIF2 DMA_IFSR_TEIF3 DMA_IFSR_BEIF3 DMA_IFSR_HTIF3 DMA_IFSR_TCIF3 DMA_IFSR_TEIF4 DMA_IFSR_BEIF4 DMA_IFSR_HTIF4 DMA_IFSR_TCIF4 DMA_IFSR_TEIF5 DMA_IFSR_BEIF5 DMA_IFSR_HTIF5 DMA_IFSR_TCIF5 DMA_IFSR_TEIF6 DMA_IFSR_BEIF6 DMA_IFSR_HTIF6 DMA_IFSR_TCIF6 DMA_IFSR_TEIF7 DMA_IFSR_BEIF7 DMA_IFSR_HTIF7 DMA_IFSR_TCIF7
11	ADC	0x006C	NVIC->ISER[0].11	ADC_ADCON1_ADIE ADC_ADCMPCON_ADGIE ADC_ADCMPCON_ADLIE	ADC_ADCON1_ADIF ADC_ADCMPCON_ADGIF ADC_ADCMPCON_ADLIF
12	PWM0	0x0070	NVIC->ISER[0].12	PWM0_CR_PWMIE	PWM0_CR_PWMIF
13	PWM1	0x0074	NVIC->ISER[0].13	PWM1_CR_PWMIE	PWM1_CR_PWMIF
14	PWM2	0x0078	NVIC->ISER[0].14	PWM2_CR_PWMIE	PWM2_CR_PWMIF
15	PWM3	0x007C	NVIC->ISER[0].15	PWM3_CR_PWMIE	PWM3_CR_PWMIF
16	PCA0	0x0080	NVIC->ISER[0].16	PCA0_CMD_ECF	PCA0_CR_CF
17	PCA1	0x0084	NVIC->ISER[0].17	PCA1_CMD_ECF	PCA1_CR_CF
18	PCA2	0x0088	NVIC->ISER[0].18	PCA2_CMD_ECF	PCA2_CR_CF
19	PCA3	0x008C	NVIC->ISER[0].19	PCA3_CMD_ECF	PCA3_CR_CF
20	EXTI9_5	0x0090	NVIC->ISER[0].20	EXTI_IMR_IMR5 EXTI_IMR_IMR6 EXTI_IMR_IMR7 EXTI_IMR_IMR8 EXTI_IMR_IMR9	EXTI_PR_PR5 EXTI_PR_PR6 EXTI_PR_PR7 EXTI_PR_PR8 EXTI_PR_PR9
21	UART0	0x0094	NVIC->ISER[0].21	UART0_CR_TIE UART0_CR_RIE UART0_CR_TCIE	UART0_FR_TI UART0_FR_RI UART0_FR_TC
22	TWI0	0x0098	NVIC->ISER[0].22	TWI0_CR_TWINTE	TWI0_FR_TWINT TWI0_FR_TOUT TWI0_FR_TRFEE
23	TWI1	0x009C	NVIC->ISER[0].23	TWI1_CR_TWINTE	TWI1_FR_TWINT TWI1_FR_TOUT TWI1_FR_TRFEE
24	SPI0	0x00A0	NVIC->ISER[0].24	SPI0_CR_SPTIE SPI0_CR_SPRIE SPI0_CR_SSDIS	SPI0_FR_SPTI SPI0_FR_SPRI SPI0_FR_MODF
25	SPI1	0x00A4	NVIC->ISER[0].25	SPI1_CR_SPTIE SPI1_CR_SPRIE SPI1_CR_SSDIS	SPI1_FR_SPTI SPI1_FR_SPRI SPI1_FR_MODF
26	UART1	0x00A8	NVIC->ISER[0].26	UART1_CR_TIE UART1_CR_RIE UART1_CR_TCIE	UART1_FR_TI UART1_FR_RI UART1_FR_TC

27	UART2	0x00AC	NVIC->ISER[0].27	UART2_CR_TIE UART2_CR_RIE UART2_CR_TCIE	UART2_FR_TI UART2_FR_RI UART2_FR_TC
28	UART3	0x00B0	NVIC->ISER[0].28	UART3_CR_TIE UART3_CR_RIE UART3_CR_TCIE	UART3_FR_TI UART3_FR_RI UART3_FR_TC
29	UART4	0x00B4	NVIC->ISER[0].29	UART4_CR_TIE UART4_CR_RIE UART4_CR_TCIE	UART4_FR_TI UART4_FR_RI UART4_FR_TC
30	EXTI15_10	0x00B8	NVIC->ISER[0].30	EXTI_IMR_IMR10 EXTI_IMR_IMR11 EXTI_IMR_IMR12 EXTI_IMR_IMR13 EXTI_IMR_IMR14 EXTI_IMR_IMR15	EXTI_PR_PR10 EXTI_PR_PR11 EXTI_PR_PR12 EXTI_PR_PR13 EXTI_PR_PR14 EXTI_PR_PR15
31	UART5	0x00BC	NVIC->ISER[0].31	UART5_CR_TIE UART5_CR_RIE UART5_CR_TCIE	UART5_FR_TI UART5_FR_RI UART5_FR_TC
32	TIM5	0x00C0	NVIC->ISER[1].0	TIM5_CR_IE	TIM5_CR_TF
33	TIM6	0x00C4	NVIC->ISER[1].1	TIM6_CR_IE	TIM6_CR_TF
34	TIM7	0x00C8	NVIC->ISER[1].2	TIM7_CR_IE	TIM7_CR_TF
35	TIM8	0x00CC	NVIC->ISER[1].3	TIM8_CR_IE	TIM8_CR_TF
36	PCA4	0x00D0	NVIC->ISER[1].4	PCA4_CMD_ECF	PCA4_CR_CF
37	PCA5	0x00D4	NVIC->ISER[1].5	PCA5_CMD_ECF	PCA5_CR_CF
38	PCA6	0x00D8	NVIC->ISER[1].6	PCA6_CMD_ECF	PCA6_CR_CF
39	PCA7	0x00DC	NVIC->ISER[1].7	PCA7_CMD_ECF	PCA7_CR_CF
40	LED	0x00E0	NVIC->ISER[1].8	LED_CR_IE	LED_FR_LEDIF LED_FR_COMIF

对于上述中断源（系统复位中断除外），只有对应的中断屏蔽位和中断使能位都允许触发中断时，对应中断标志被硬件置1的同时才会触发中断，中断响应过程由中断优先级设置和嵌套设置决定。

15.2 外部中断/事件控制器（EXTI）

外部中断/事件控制器由16个产生事件/中断请求的边沿检测器组成，这16条线都可配置到GPIO口。每个输入线可以独立地配置输入类型(脉冲或挂起)和对应的触发事件(上升沿或下降沿或者双边沿都触发)。每个输入线都可以独立地被屏蔽。挂起寄存器保持着状态线的中断请求。

15.2.1 主要特性

- 每个中断/事件都有独立的触发和屏蔽
- 每个中断线都有专用的状态位
- 支持多达16个软件的中断请求
- 支持作为DMA触发源
- 输入检测采样参数可调，在normal运行和stop模式下都具备debounce能力

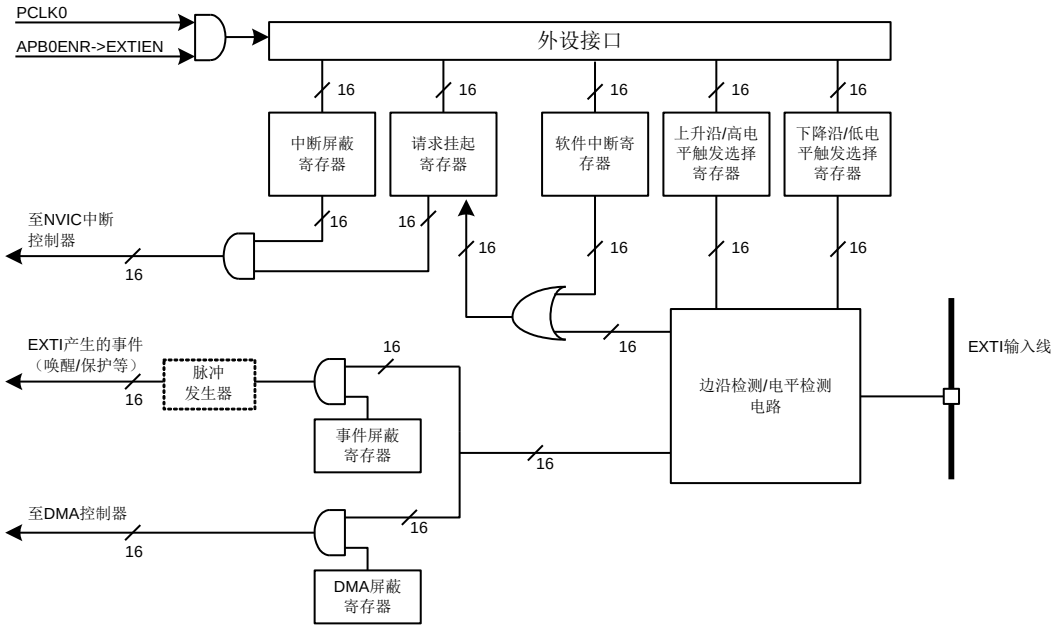


图15-1　　外部中断/事件控制器框图

上图说明如下：

1. 中断和事件在或门之前是同源的，边沿检测电路检测外部输入线的边沿变化事件，然后分成三路，一路进入中断系统，请求CPU进行中断响应，另一路进入事件系统，通过脉冲发生器产生一个硬件脉冲作为系统内其它模块的触发信号，第三路进入DMA控制器，可以触发DMA进行操作（实际上第二路与第三路本质都是事件，此电路中为清晰起见把二者独立开）；
2. 边沿检测电路可以检测上升沿、下降沿、上升下降沿；
3. 电平检测电路可以检测高电平、低电平、高低电平；
4. 软件中断优先于外部信号，即当软件中断位置为“1”时，无论外部输入是什么情况，或门总是输出有效信号；

说明：在中断系统的基础上保留事件发生路径，主要是作为一些特殊外设的触发源，比如DMA传输、比如缓冲寄存器的更新、AD采样触发、事件唤醒内核（WFE）等。这些触发操作直接作用到硬件，不需要经过CPU的中断处理途径。

说明：EXTI的请求挂起寄存器和中断屏蔽寄存器和NVIC是独立的。
EXTI0能引起NMI中断，由IEN_EXTI0@SYSCFG_SAFR控制位使能。

15.2.2 唤醒事件管理

SH32F9001_32F9801可以处理外部或内部事件来唤醒内核(WFE)。唤醒事件可以通过下述配置产生：

● 在外设的控制寄存器使能一个中断，但不在NVIC中使能，同时在Cortex-M3的系统控制寄存器中使能SEVONPEND位。当CPU从WFE恢复后，需要清除相应外设的中断挂起位（在EXTI挂起寄存器中）和外设NVIC中断通道挂起位(在NVIC中断挂起清除寄存器中)。

● 配置一个外部或内部EXTI线为事件模式，当CPU从WFE恢复后，因为对应事件线的挂起位没有被置位，不必清除相应外设的中断挂起位或NVIC中断通道挂起位。使用外部I/O端口作为唤醒事件，请参见GPIO章节的“GPIO配置为外部中断/唤醒线”说明。

15.2.3 功能说明

要产生中断，必须先配置好并使能中断线。根据需要的边沿检测设置2个触发寄存器，同时在中断屏蔽寄存器的相应位写’1’允许中断请求。当外部中断线上发生了期待的边沿时，将产生一个中断请求，对应的挂起位也随之被置’1’。在挂起寄存器的对应位写’1’，将清除该中断请求。

如果需要产生事件，必须先配置好并使能事件线。根据需要的边沿检测通过设置2个触发寄存器，同时在事件屏蔽寄存器的相应位写’1’允许事件请求。当事件线上发生了需要的边沿时，将产生一个事件请求脉冲，对应的挂起位不被置’1’。

SH32F9001_32F9801也支持电平触发，通过设置触发模式为电平触发，并设置高电平或低电平触发，EXTI可以工作在电平触发状态，与边沿触发不同，如果触发电平不改变，将会持续触发。

EXTI输入信号支持采样滤波，在正常运行模式下使用APB0时钟PCLK0作为时钟源，支持采样时钟预分频（最高128分频），并支持多次采样以达到滤波效果。在停机模式下，外部中断通道仍有效，但此时模块的预分频设置无效[注]。 具体见“EXTI采样控制寄存器”。

注：在停机模式下，PCLK0无时钟，EXTI的时钟源为128KHz，预分频无效，采样次数有效，EXTI输入通道经过Debounce后检测输入信号，产生中断/事件请求后退出ShutDown模式，并唤醒系统时钟。

EXTI支持作为DMA触发源，EXTI外部中断事件发生时，可选择送出DMA请求，触发DMA传输，DMA请求在DMA控制器响应并应答后释放。EXTI的DMA使能信号具体见“DMA请求屏蔽控制器”。

硬件中断选择

通过下面的过程来配置16个输入线做为中断源：

● 配置16个中断线的屏蔽位(EXTI_IMR)

● 配置所选中断线的触发选择位(EXTI_RTSR和EXTI_FTSR)；

● 配置对应到外部中断控制器(EXTI)的NVIC中断通道的使能和屏蔽位，使得16个中断线中的请求可以被正确地响应。

硬件事件选择

通过下面的过程，可以配置16个线路为事件源

● 配置16个事件线的屏蔽位(EXTI_EMR)

● 配置事件线的触发选择位(EXTI_RTSR和EXTI_FTSR)

软件中断的选择

16个线路可以被配置成软件中断线。下面是产生软件中断的过程：

● 配置16个中断/事件线屏蔽位(EXTI_IMR)

● 设置软件中断寄存器的请求位(EXTI_SWIER)

15.2.4 外部中断/事件线路映像

GPIO以下图的方式连接到16个外部中断/事件线上：

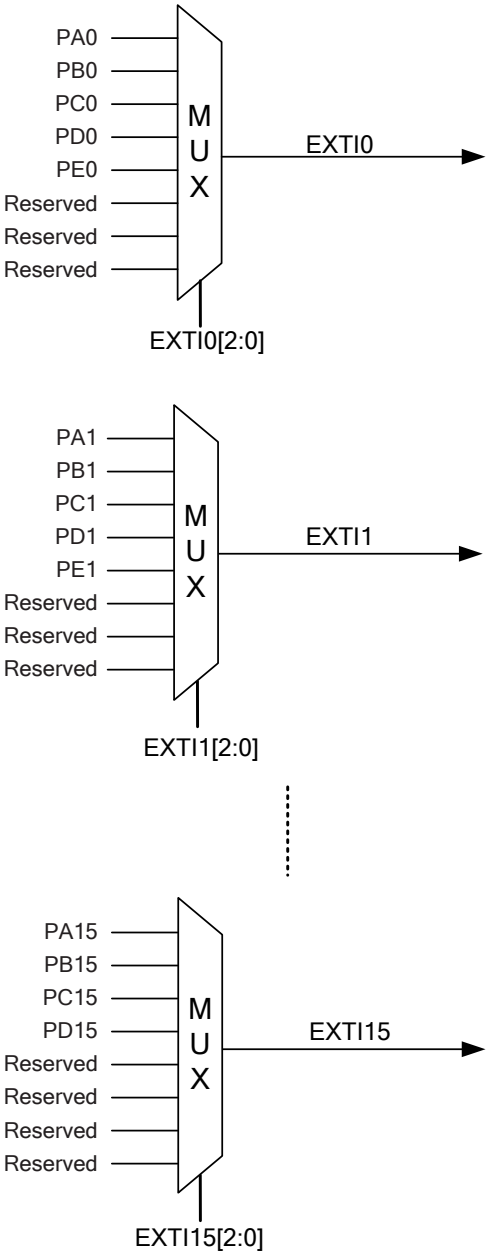


图15-2　　外部中断通用 I/O 映像

15.3 寄存器

EXTI 模块寄存器列表 (基地址: 0x4000 7800)

地址	寄存器名	说明
0x4000 7800	IMR	中断屏蔽寄存器
0x4000 7804	EMR	事件屏蔽寄存器
0x4000 7808	TMSR	触发模式选择寄存器
0x4000 780C	RTSR	上升沿/高电平触发模式选择寄存器
0x4000 7810	FTSR	下降沿/低电平触发模式选择寄存器
0x4000 7814	SWIER	软件中断事件寄存器
0x4000 7818	PR	挂起寄存器
0x4000 781C	CFGL	外部中断配置低寄存器
0x4000 7820	CFGH	外部中断配置高寄存器
0x4000 7824	SAMPL	EXTI采样控制低寄存器
0x4000 7828	SAMPH	EXTI采样控制高寄存器
0x4000 782C	DMR	DMA屏蔽寄存器

15.3.1 中断屏蔽寄存器 (EXTI_IMR)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
IMRy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	IMRy(y=15~0)	线y上的中断屏蔽(Interrupt Mask on line y) 0: 屏蔽来自线y上的中断请求 1: 开放来自线y上的中断请求

15.3.2 事件屏蔽寄存器 (EXTI_EMR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
EMRy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	EMRy(y=15~0)	线y上的事件屏蔽(Event Mask on line y) 0: 屏蔽来自线y上的事件请求 1: 开放来自线y上的事件请求

15.3.3 触发模式选择寄存器 (EXTI_TMSR)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TMRy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**TMRy(y=15~0)**	**线y上的触发模式选择位(Trigger Mode selection)** 0: 边沿触发 1: 电平触发 注：边沿触发为one-shot方式，触发后通过软件clear pending flag来清除标志位。电平触发为level方式，触发后只能通过外部电平变化来清除标志位，无法通过软件清除。

15.3.4 上升沿/高电平触发模式选择寄存器 (EXTI_RTSR)

偏移地址: 0x000C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RTRy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**RTRy(y=15~0)**	**线y上的上升沿/高电平触发事件配置位 (Rising/high level trigger event configuration bit of line y)** 0：禁止输入线y上的上升沿/高电平触发(中断和事件) 1：允许输入线y上的上升沿/高电平触发(中断和事件)

15.3.5 下降沿/低电平触发模式选择寄存器 (EXTI_FTSR)

偏移地址: 0x0010

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FTRy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	FTRy(y=15~0)	**线y上的下降沿/低电平触发事件配置位 (Falling/low level trigger event configuration bit of line y)** 0：禁止输入线y上的下降沿/低电平触发(中断和事件) 1：允许输入线y上的下降沿/低电平触发(中断和事件)

15.3.6 软件中断寄存器 (EXTI_SWIER)

偏移地址: 0x0014

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SWIERy(y=15~0)															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	SWIERy(y=15~0)	**线y上的软件中断 (Software interrupt on line y)** 当该位为0时，写1将设置EXTI_PR中相应的挂起位。如果在EXTI_IMR中允许产生该中断，则此时将产生一个中断。 0： 无意义 1：置位EXTI_PR相应pending位，引起软件触发 该信号由硬件自动清除。 如果对应的EXTI_PR已经置位，则本次操作无效。

15.3.7 挂起寄存器 (EXTI_PR)

偏移地址: 0x0018

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PRCy(y=15~0)															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PRy(y=15~0)															
RO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	PRCy(y=15~0)	**挂起清除位** 0：无效 1：清除
15 ~ 0	PRy(y=15~0)	**挂起位 (Pending bit)** 0：没有发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的触发事件，该位被置1。 注：对电平触发，如果外部电平没有变化，则清除pending位后将立刻又产生pending。

15.3.8 外部中断配置低寄存器 (EXTI_CFGL)

偏移地址: 0x001C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留	EXTI7[2:0]			保留	EXTI6[2:0]			保留	EXTI5[2:0]			保留	EXTI4[2:0]		
-	RW			-	RW			-	RW			-	RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留	EXTI3[2:0]			保留	EXTI2[2:0]			保留	EXTI1[2:0]			保留	EXTI0[2:0]		
-	RW			-	RW			-	RW			-	RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31	保留	-
30 ~ 28	EXTI7[2:0]	**EXTI 7 配置位** 用于选择 EXTI7 外部中断的源输入，由软件写入。 000：PA[7] 引脚 001：PB[7] 引脚 010：PC[7] 引脚 011：PD[7] 引脚 100：PE[7] 引脚 101：保留 110：保留 111：保留
27	保留	-
26 ~ 24	EXTI6[2:0]	**EXTI 6 配置位** 用于选择 EXTI6 外部中断的源输入，由软件写入。 000：PA[6] 引脚 001：PB[6] 引脚 010：PC[6] 引脚 011：PD[6] 引脚 100：PE[6] 引脚 101：保留 110：保留 111：保留
23	保留	-
22 ~ 20	EXTI5[2:0]	**EXTI 5 配置位** 用于选择 EXTI5 外部中断的源输入，由软件写入。 000：PA[5] 引脚 001：PB[5] 引脚 010：PC[5] 引脚 011：PD[5] 引脚 100：PE[5] 引脚 101：保留 110：保留 111：保留
19	保留	-
18 ~ 16	EXTI4[2:0]	**EXTI 4 配置位** 用于选择 EXTI4 外部中断的源输入，由软件写入。 000：PA[4] 引脚 001：PB[4] 引脚 010：PC[4] 引脚 011：PD[4] 引脚

		100：PE[4] 引脚 101：保留 110：保留 111：保留
15	**保留**	**-**
14 ~ 12	**EXTI3[2:0]**	**EXTI 3 配置位** 用于选择 EXTI3 外部中断的源输入，由软件写入。 000：PA[3] 引脚 001：PB[3] 引脚 010：PC[3] 引脚 011：PD[3] 引脚 100：PE[3] 引脚 101：保留 110：保留 111：保留
11	**保留**	**-**
10 ~ 8	**EXTI2[2:0]**	**EXTI 2 配置位** 用于选择 EXTI2 外部中断的源输入，由软件写入。 000：PA[2] 引脚 001：PB[2] 引脚 010：PC[2] 引脚 011：PD[2] 引脚 100：PE[2] 引脚 101：保留 110：保留 111：保留
7	**保留**	**-**
6 ~ 4	**EXTI1[2:0]**	**EXTI 1 配置位** 用于选择 EXTI1 外部中断的源输入，由软件写入。 000：PA[1] 引脚 001：PB[1] 引脚 010：PC[1] 引脚 011：PD[1] 引脚 100：PE[1] 引脚 101：保留 110：保留 111：保留
3	**保留**	**-**
2 ~ 0	**EXTI0[2:0]**	**EXTI 0 配置位** 用于选择 EXTI0 外部中断的源输入，由软件写入。 000：PA[0] 引脚 001：PB[0] 引脚 010：PC[0] 引脚 011：PD[0] 引脚 100：PE[0] 引脚 101：保留 110：保留 111：保留

15.3.9 外部中断配置高寄存器 (EXTI_CFGH)

偏移地址: 0x0020

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留	EXTI15[2:0]			保留	EXTI14[2:0]			保留	EXTI13[2:0]			保留	EXTI12[2:0]		
-	RW			-	RW			-	RW			-	RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留	EXTI11[2:0]			保留	EXTI10[2:0]			保留	EXTI9[2:0]			保留	EXTI8[2:0]		
-	RW			-	RW			-	RW			-	RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31	保留	-
30 ~ 28	EXTI15[2:0]	**EXTI 15 配置位** 用于选择 EXTI15 外部中断的源输入，由软件写入。 000：PA[15] 引脚 001：PB[15] 引脚 010：PC[15] 引脚 011：PD[15] 引脚 100：保留 101：保留 110：保留 111：保留
27	保留	-
26 ~ 24	EXTI14[2:0]	**EXTI 14 配置位** 用于选择 EXTI14 外部中断的源输入，由软件写入。 000：PA[14] 引脚 001：PB[14] 引脚 010：PC[14] 引脚 011：PD[14] 引脚 100：保留 101：保留 110：保留 111：保留
23	保留	-
22 ~ 20	EXTI13[2:0]	**EXTI 13 配置位** 用于选择 EXTI13 外部中断的源输入，由软件写入。 000：PA[13] 引脚 001：PB[13] 引脚 010：PC[13] 引脚 011：PD[13] 引脚 100：保留 101：保留 110：保留 111：保留
19	保留	-
18 ~ 16	EXTI12[2:0]	**EXTI 12 配置位** 用于选择 EXTI12 外部中断的源输入，由软件写入。 000：PA[12] 引脚 001：PB[12] 引脚 010：PC[12] 引脚 011：PD[12] 引脚

		100：保留 101：保留 110：保留 111：保留
15	**保留**	**-**
14 ~ 12	**EXTI11[2:0]**	**EXTI 11 配置位** 用于选择 EXTI11 外部中断的源输入，由软件写入。 000：PA[11] 引脚 001：PB[11] 引脚 010：PC[11] 引脚 011：PD[11] 引脚 100：保留 101：保留 110：保留 111：保留
11	**保留**	**-**
10 ~ 8	**EXTI10[2:0]**	**EXTI 10 配置位** 用于选择 EXTI10 外部中断的源输入，由软件写入。 000：PA[10] 引脚 001：PB[10] 引脚 010：PC[10] 引脚 011：PD[10] 引脚 100：PE[10] 引脚 101：保留 110：保留 111：保留
7	**保留**	**-**
6 ~ 4	**EXTI9[2:0]**	**EXTI 9 配置位** 用于选择 EXTI9 外部中断的源输入，由软件写入。 000：PA[9] 引脚 001：PB[9] 引脚 010：PC[9] 引脚 011：PD[9] 引脚 100：PE[9] 引脚 101：保留 110：保留 111：保留
3	**保留**	**-**
2 ~ 0	**EXTI8[2:0]**	**EXTI 8 配置位** 用于选择 EXTI8 外部中断的源输入，由软件写入。 000：PA[8] 引脚 001：PB[8] 引脚 010：PC[8] 引脚 011：PD[8] 引脚 100：PE[8] 引脚 101：保留 110：保留 111：保留

15.3.10 EXTI 采样控制低寄存器 (EXTI_SAMPL)

偏移地址: 0x0024

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PS7[1:0]		SN7[1:0]		PS6[1:0]		SN6[1:0]		PS5[1:0]		SN5[1:0]		PS4[1:0]		SN4[1:0]	
RW		RW		RW		RW		RW		RW		RW		RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PS3[1:0]		SN3[1:0]		PS2[1:0]		SN2[1:0]		PS1[1:0]		SN1[1:0]		PS0[1:0]		SN0[1:0]	
RW		RW		RW		RW		RW		RW		RW		RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 30	**PS7[1:0]**	**EXTI7采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
29 ~ 28	**SN7[1:0]**	**EXTI7连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
27 ~ 26	**PS6[1:0]**	**EXTI6采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
25 ~ 24	**SN6[1:0]**	**EXTI6连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
23 ~ 22	**PS5[1:0]**	**EXTI5采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
21 ~ 20	**SN5[1:0]**	**EXTI5连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
19 ~ 18	**PS4[1:0]**	**EXTI4采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
17 ~ 16	**SN4[1:0]**	**EXTI4连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4

15 ~ 14	**PS3[1:0]**	**EXTI3采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
13 ~ 12	**SN3[1:0]**	**EXTI3连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
11 ~ 10	**PS2[1:0]**	**EXTI2采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
9 ~ 8	**SN2[1:0]**	**EXTI2连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
7 ~ 6	**PS1[1:0]**	**EXTI1采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
5 ~ 4	**SN1[1:0]**	**EXTI1连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
3 ~ 2	**PS0[1:0]**	**EXTI0采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
1 ~ 0	**SN0[1:0]**	**EXTI0连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4

15.3.11 EXTI 采样控制高寄存器 (EXTI_SAMPH)

偏移地址: 0x0028

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PS15[1:0]		SN15[1:0]		PS14[1:0]		SN14[1:0]		PS13[1:0]		SN13[1:0]		PS12[1:0]		SN12[1:0]	
RW		RW		RW		RW		RW		RW		RW		RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PS11[1:0]		SN11[1:0]		PS10[1:0]		SN10[1:0]		PS9[1:0]		SN9[1:0]		PS8[1:0]		SN8[1:0]	
RW		RW		RW		RW		RW		RW		RW		RW	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 30	**PS15[1:0]**	**EXTI15采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
29 ~ 28	**SN15[1:0]**	**EXTI15连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
27 ~ 26	**PS14[1:0]**	**EXTI14采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
25 ~ 24	**SN14[1:0]**	**EXTI14连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
23 ~ 22	**PS13[1:0]**	**EXTI13采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
21 ~ 20	**SN13[1:0]**	**EXTI13连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
19 ~ 18	**PS12[1:0]**	**EXTI12采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
17 ~ 16	**SN12[1:0]**	**EXTI12连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
15 ~ 14	**PS11[1:0]**	**EXTI11采样时钟预分频比选择位（Prescaler select）**

		00：1/1 01：1/4 10：1/16 11：1/128
13 ~ 12	**SN11[1:0]**	**EXTI11连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
11 ~ 10	**PS10[1:0]**	**EXTI10采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
9 ~ 8	**SN10[1:0]**	**EXTI10连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
7 ~ 6	**PS9[1:0]**	**EXTI9采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
5 ~ 4	**SN9[1:0]**	**EXTI9连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4
3 ~ 2	**PS8[1:0]**	**EXTI8采样时钟预分频比选择位（Prescaler select）** 00：1/1 01：1/4 10：1/16 11：1/128
1 ~ 0	**SN8[1:0]**	**EXTI8连续采样次数选择位（sample number）** 00：1 01：2 10：3 11：4

15.3.12 DMA 屏蔽寄存器 (EXTI_DMR)

偏移地址: 0x002C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DMRy(y=15~0)															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	DMRy(y=15~0)	线y上的DMA屏蔽(DMA Mask on line y) 0: 屏蔽来自线y上的DMA请求 1: 开放来自线y上的DMA请求

16. DMA 控制器

16.1 简介

直接存储器存取(DMA)用来提供在外设和存储器之间或者存储器和存储器之间的高速数据传输。无须CPU干预，数据可以通过DMA快速地移动，这就节省了CPU的资源来做其它操作。DMA控制器有8个信道，每个信道专门用来管理来自于一个或多个外设对存储器访问的请求。还有一个仲裁器来协调CPU与DMA请求的优先权。

注：对于一个双向数据传输应用，需要2个DMA通道分别完成发送和接收。

16.2 主要特性

- 8个独立的可配置的信道，每个信道可支持一个单向传输
- 每个通道都直接连接专用的硬件DMA请求，每个通道都同样支持软件触发
- 支持One-Shot和To-End两种触发方式，支持软件触发方式
- 基本传输单位是突发（burst），突发长度可设，当burst长度是1时，相当于single传输
- 在同一个DMA模块上，多个请求间的优先权可以通过软件编程设置(共有四级：很高、高、中等和低)，同一优先权的访问顺序还会进行轮换处理
- 数据源和目标数据区的传输方式、传输位宽、循环方式、循环数量、指针增量都可独立配置
- 每个通道都有4个事件标志(半传输、传输完成、突发传输结束、传输出错)，这4个事件任一个都可产生中断请求
- DMA源和目标都支持存储器和外设，即支持存储器到存储器、外设到存储器、存储器到外设、外设到外设之间的传输
- SRAM、CRAM、APB0、APB1和AHB外设均可作为访问的源和目标，Flash只能作为源不能作为目标[注]
- 可编程的数据传输数目，最大为2048个突发（burst），支持自动重载
- 每个突发之间可强制释放DMA一段时间以保证CPU效率

注：Flash作为DMA目标时，在DMA配置上没有做限制，但FLASH接口上屏蔽了DMA写入操作，即不会产生效果。操作时不会产生DMA error，也不会产生flash write error。

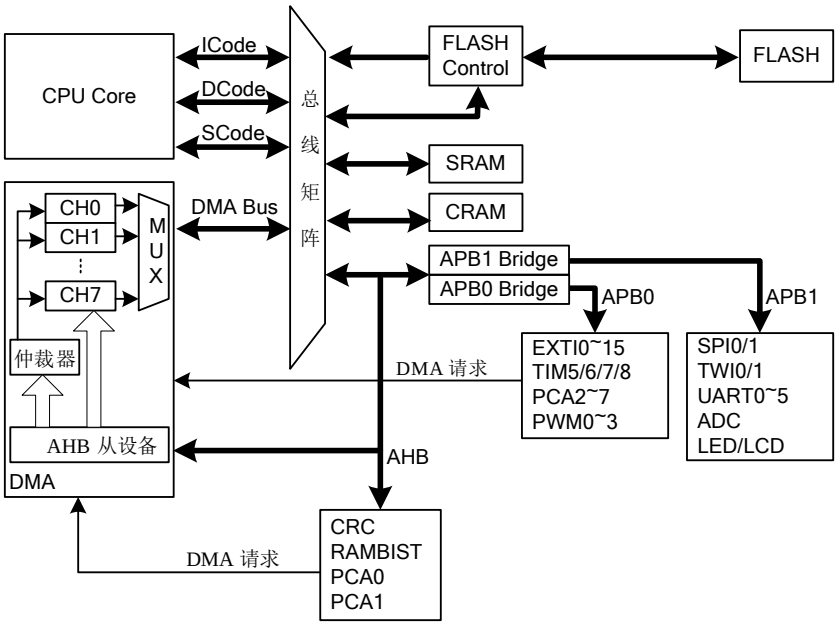


图16-1 DMA 控制器逻辑框图

说明：

（1） DMA 支持从 Flash 中读取表格数据，传输到 SRAM 或片上外设，但不支持向 Flash 写入数据，DMA 控制器没有禁止这种配置，只是执行向 Flash 写入时没有效果；

（2）DMA 模块既是 AHB 主机又是 AHB 从机，前者表示 DMA 与 CPU 拥有类似的权限可控制总线矩阵上数据的传输，后者表示 DMA 作为一个设备模块需要通过 CPU 进行参数配置。

16.3 功能描述

DMA控制器和CPU核心共享系统数据总线，执行直接存储器数据传输。当CPU和DMA同时访问相同的目标(RAM或外设)时，DMA请求会暂停CPU访问系统总线达若干个周期，总线仲裁器执行循环调度，以保证CPU至少可以得到一半的系统总线带宽。而在突发传输模式下，DMA暂停CPU的时间还要更长。

16.3.1 DMA 处理

在发生一个事件后，外设向DMA控制器发送一个请求信号。DMA控制器根据通道的优先权处理请求。当DMA控制器开始访问发出请求的外设时，DMA控制器立即发送给它一个应答信号。当从DMA控制器得到应答信号时，外设立即释放它的请求。一旦外设释放了这个请求，DMA控制器同时撤销应答信号。如果有更多的请求时，外设可以启动下一个周期。 总之，每次DMA传送由3个操作组成：

- 从源地址寄存器指示的存储器地址取数据，第一个源地址是DMA_SARx寄存器指定的外设基地址或存储器单元。
- 存数据到目标地址寄存器指示的目标存储器地址，第一个目标地址是DMA_DARx寄存器指定的外设基地址或存储器单元。
- 执行一次DMA_CPKTx寄存器的递增操作，该寄存器包含已完成的突发数目。

16.3.2 仲裁器

仲裁器有2个基本功能：

- 负责协调CPU与DMA控制器对共享总线的占用；
- 负责协调多个DMA通道的优先级处理；

16.3.2.1 总线占用方式

DMA占用总线方式称为“总线挪用方式”，这种情况下DMA每传输一次就需要暂停并且释放总线至少一个总线周期，将总线控制权交还给CPU。释放总线的周期数取决于BURSTIDLE参数，最小1个，最大32个。

DMA的优先级高于CPU，DMA请求到达时，CPU如果处于非锁定状态（锁定状态指当前CPU的执行不能被打断），则在执行完当前指令后就应该响应DMA请求，释放总线控制权。

由于SH32F9001_32F9801中DMA基本传输单位是突发（burst），在一个突发启动后，DMA将占用总线资源直到该突发传输完成，突发的长度由BURSTLEN设置，最小1个数据，最大16个数据（每个DMA数据可能为1字节/2字节/4字节）。

如果突发长度设置为最大，将会对CPU的执行效率造成一定影响，因此，大突发一般用于对CPU效率不敏感，但存在小块数据需要连续快速收发的场合。

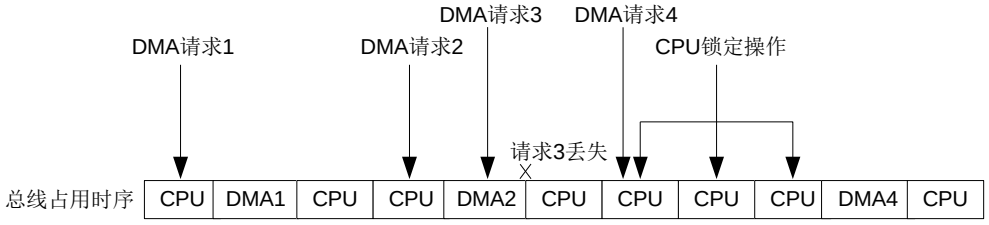


图16-2 DMA 总线挪用方式工作示意图（突发长度为 1）

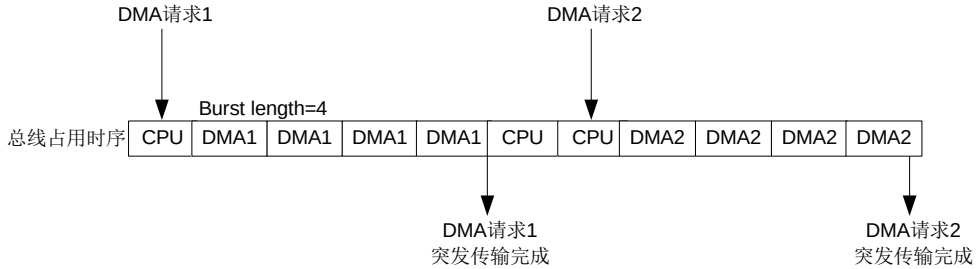


图16-3 DMA 总线挪用方式工作示意图（突发长度为 4）

16.3.2.2 多通道优先级处理

根据通道请求的优先级来启动设备的访问，每个通道的优先权可以在DMA_CCRx寄存器中设置，有4个等级：

— 最高优先级

— 高优先级

— 中等优先级

— 低优先级

高优先级始终优先于低优先级访问。

如果多个通道设置同一优先级，并且同时提出请求，按照通道序号决定访问顺序，即信道1先于信道2访问，依此类推，但如果已经有通道进行过访问，则会进行访问顺序轮换处理，即同优先级内某个通道访问完成后，访问顺序排到最后，依此类推，以保证同优先级内其它通道的访问机会。

举例来说，通道1，3，5为高优先级，2，4，6为低优先级。

（1）1，3触发同时发生，1>>3；

（2）之后3，5同时发生，由于3已经传输过一次，因此5>>3；

（3）之后1，3，5同时发生，则1>>5>>3；

（4）之后1，2，3，4同时发生，则1>>3>>2>>4，假如信道4开始前，1，2，6又发生了，那么1>>3>>2>>1>>4>>6>>2。

16.3.3 DMA 通道

每个信道都可以在有固定地址的源存储空间和目标存储空间之间执行DMA传输。 总共8个DMA信道，每个信道可连接多个外设，但每次只能选择一个外设，通过DMA_CCRx寄存器的STRMSEL[2:0]字段进行选择。

16.3.3.1 DMA 传输数量和自动重载

DMA传输数量寄存器DMA_NPKTx定义的是DMA传输的突发个数，表示NPKT+1个突发传输数量。最小设置0表示1个突发，最大设置511表示512个突发。DMA_CPKTx是DMA传输计数器，每次传输后计数器加1，直到计数到DMA_NPKTx时，根据DMA的自动重载功能设置，DMA将停止传输或者重载后继续传输。

当RELOADy@DMA_CSR为0时自动重载关闭，CPKTx在计数到达NPKTx后将停止传输，并自动清除DMA使能位，此时计数器DMA_CPKTx保持不变，如果使能DMA传输完成中断，则申请DMA中断。DMA_CPKTx在下一次DMA使能位开启时自动清0。

当RELOADy@DMA_CSR为1时自动重载开启，CPKTx在计数到达NPKTx后自动清0，DMA使能位保持开启，DMA传输源和目标地址恢复为初始设定值，DMA传输重新开始（不间断）。如果开启DMA传输完成中断，同样会申请DMA中断。在用户关闭RELOAD控制位后，DMA会等到当前这一轮重载传输完成后才会停止传输。

特殊：在设置To-end时，开启自动重载后不会自动开始传输，需要硬件或软件重新触发一下。

特殊：当DMA_NPKTx=0时，DMA使能打开，触发信号引起1次传输后使能关闭，DMA_CPKTx为0，表示1次突发传输。

注：DMA_CPKTx初始化为0，在DMA使能关闭时该初始化值并无意义，在DMA开启传输后以CPKT+1的方式表示实际发生的突发传输数量。

每个突发包含1～16个DMA数据，具体见DMA_CCRx@DMA_BURSTLEN位设置。

根据DMA_CCRx@SIZE设置的不同，每个DMA数据可能是1、2、4个字节。

DMA传输数量寄存器只能在DMA关闭时设置，在开启DMA通道时，该寄存器变为只读。

注意：DMA_NPKTx和DMA_CPKTx的计数单位都是突发（Burst）。

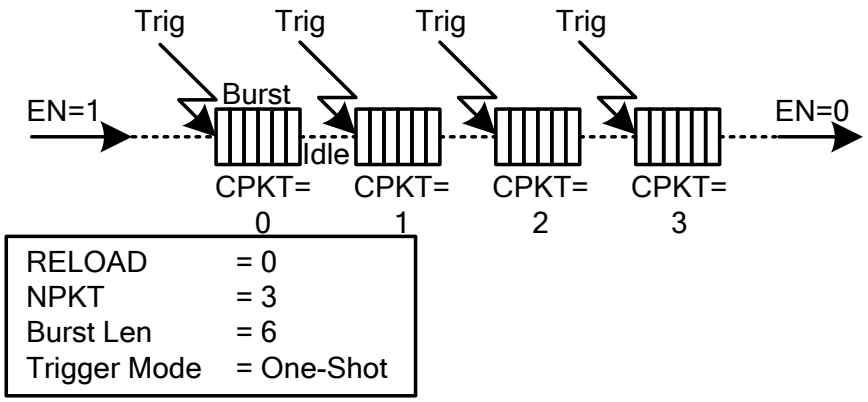


图16-4　　基于 Burst 的 DMA 传输示意图

16.3.3.2　可编程数据位宽

源端和目标端的传输数据位宽可以通过DMA_CCRx寄存器中的SIZE字段编程。根据DMA_CCRx@SIZE设置的不同，每个DMA数据可能是1、2、4个字节。

16.3.3.3　指针修改方式

通过设置DMA_CCRx寄存器中的SPTYP和DPTYP标志位，源端和目标端的内部当前地址指针（无法读出）在每次传输后可以有选择地完成自动修改。DMA支持指针INC、DEC、FIX和WRAP方式，其中WRAP方式用于循环模式。指针修改方式全部由软件设定。

当设置为INC方式时，下一个要传输的地址将是前一个地址加上增量值，增量值取决于所选的数据宽度不同而可取1、2或4三种情况。

DMA_SARx/DMA_DARx存放的是源端/目标端的初始指针，并且在DMA传输过程中保持不变，软件不能改变和读出当前正在传输的地址（它在内部的当前源端/目标端地址寄存器中）。

注：用户如想了解当前内部指针位置，只能通过（初始指针+突发传输计数×突发长度×SIZE参数）来间接计算得到。

16.3.3.4　循环模式

循环模式用于处理循环缓冲区和连续的数据传输(如ADC的连续转换模式)。支持源端和目标端独立开启循环。

循环周期固定为突发长度，循环传输完成后其地址指针会自动恢复成循环开始时的初值，无需软件更改指针。

当设置SPTYP/DPTYP=WRAP方式时就开启对应的循环模式。

下表给出了DMA循环模式的各种设置。

表16-1　　DMA 循环方式设置表

循环方式	设置	功能说明
非循环	SPTYP=00/01/10，DPTYP=00/01/10	非循环模式。完全由软件进行指针边界操作，需要CPU开销。
单循环	源端循环： SPTYP=11，DPTYP=00/01/10 目标端循环： SPTYP=00/01/10，DPTYP=11	单循环模式。比如接收多通道ADC采样数据，在ADC端按照通道数设置循环长度，可简化部分操作。
双循环	SPTYP=11，DPTYP=11	双循环模式。简化两端的CPU指针操作。

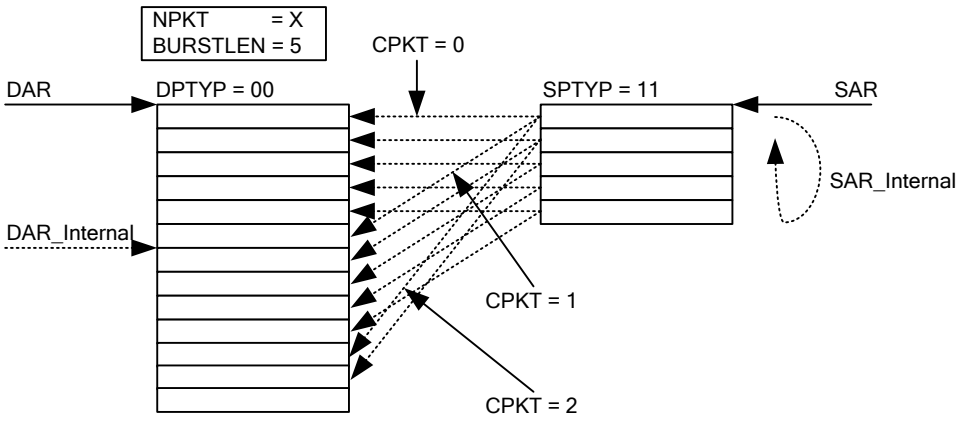


图16-5　单循环模式

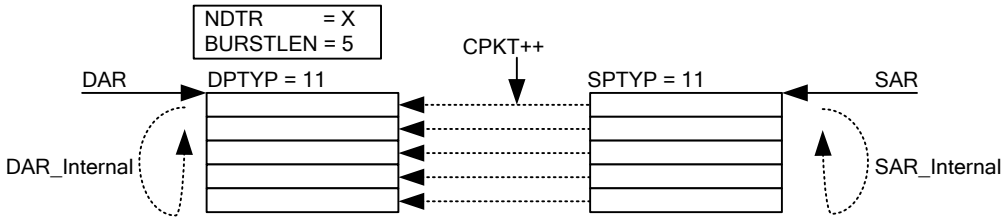


图16-6　双循环模式

16.3.3.5　软件触发方式

DMA通道的操作可以在没有外设请求的情况下进行，这种操作就是软件触发方式。

软件触发方式一般用于存储器到存储器之间的数据搬移，但并不局限于此。当设置了DMA_CCRx寄存器中的SWTRG位之后，将启动一次软件触发，其效果与脉冲触发相同。

在做存储器到存储器数据搬移时，需注意Flash只能作为源被读取，无法写入。

16.3.3.6　单次传输与突发传输

单次传输是突发传输的特例，突发长度为1即为单次传输。

单次传输是DMA的基本传输方式，DMA仲裁策略能够保证CPU至少获得一半的运行时间，从而在保证CPU运行时间与解放CPU干预外设负担之间达成较为理想的平衡，有助于提升系统运行效率。

突发传输（指突发长度大于1）适用于快速小块数据搬移，具体见“仲裁器”部分描述。

16.3.3.7　One-shot 触发与 To-end 触发

One-shot触发每次只进行一次DMA突发传输，To-end触发后将把DMA_NPKTx寄存器中的全部DMA传输数量的突发传输完成后才停止。

To-end触发时前后突发之间同样存在IDLE时隙供CPU运行。

16.3.3.8　BUSY 信号

DMA通道在传输时会置出busy信号，用户查询各通道的busy信号可确定当前哪一个通道在传输。BUSY信号在One-Shot时只指示一次突发传输，在To-End时需要持续到传输完成。

注：busy信号只指示burst正在传输，在burst idle期间busy恢复为空闲。

16.3.3.9 Burst Idle 设置

DMA在传输完成一个突发后，释放总线给CPU执行。如果有多个DMA通道开启，则需要等待当前通道的Burst Idle结束后才能执行下一个DMA通道，即CPU的执行时间不会被排队的DMA抢占，这种设计保证了CPU的执行效率。

16.3.3.10 信道配置过程

下面是配置DMA信道x的过程(x代表通道号，采用非循环单次传输单次触发模式说明)：

1. 在DMA_SARx寄存器中设置源数据区的首地址。
2. 在DMA_DARx寄存器中设置目标数据区的首地址。
3. 在DMA_NPKTx寄存器中设置要传输的数据量。
4. 在DMA_CCRx寄存器中设置DMA通道映像表STRMSEL[2:0]，每个DMA通道只能选择一个外设。
6. 在DMA_CCRx寄存器中设置突发长度、触发方式、指针修改方式、数据位宽、信道优先级、中断使能。
7. 设置DMA_CCRx寄存器的ENABLE位，启动该通道。

一旦启动了DMA通道，它即可响应连到该通道上的内存或外设的DMA请求（注）。

当传输一半的数据后，半传输标志(HTIF)被置1，当设置了允许半传输中断位(HTIE)时，将产生一个中断请求。在数据传输结束后，传输完成标志(TCIF)被置1，当设置了允许传输完成中断位(TCIE)时，将产生一个中断请求。

注：有关外设的DMA请求使能，可以通过设置相应外设寄存器中的控制位（在对应外设章节中），被独立地开启或关闭，如果配置并开启了对应的DMA通道，而相应的外设的DMA请求未开启，则硬件不做处理，DMA传输也不会发生。

16.3.4 可编程的数据传输宽度、对齐方式和数据大小端

当SSIZE和DSIZE不相同时，DMA模块按照下表进行数据对齐。

表16-2 可编程的数据传输宽度和大小端操作（当 SPTYP=DPTYP=00）

源端宽度	目标宽度	传输数目	源：地址/数据	传输操作	目标：地址/数据
8	**8**	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: 在0x0读B0[7:0]，在0x0写B0[7:0] 2: 在0x1读B1[7:0]，在0x1写B1[7:0] 3: 在0x2读B2[7:0]，在0x2写B2[7:0] 4: 在0x3读B3[7:0]，在0x3写B3[7:0]	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3
8	16	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: 在0x0读B0[7:0]，在0x0写00B0[15:0] 2: 在0x1读B1[7:0]，在0x2写00B1[15:0] 3: 在0x2读B2[7:0]，在0x4写00B2[15:0] 4: 在0x3读B3[7:0]，在0x6写00B3[15:0]	0x0 / 00B0 0x2 / 00B1 0x4 / 00B2 0x6 / 00B3
8	32	4	0x0 / B0 0x1 / B1 0x2 / B2 0x3 / B3	1: 在0x0读B0[7:0]，在0x0写000000B0[31:0] 2: 在0x1读B1[7:0]，在0x4写000000B1[31:0] 3: 在0x2读B2[7:0]，在0x8写000000B2[31:0] 4: 在0x3读B3[7:0]，在0xC写000000B3[31:0]	0x0 / 000000B0 0x4 / 000000B1 0x8 / 000000B2 0xC / 000000B3
16	8	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: 在0x0读B1B0[15:0]，在0x0写B0[7:0] 2: 在0x2读B3B2[15:0]，在0x1写B2[7:0] 3: 在0x4读B5B4[15:0]，在0x2写B4[7:0] 4: 在0x6读B7B6[15:0]，在0x3写B6[7:0]	0x0 / B0 0x1 / B2 0x2 / B4 0x3 / B6
16	**16**	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: 在0x0读B1B0[15:0]，在0x0写B1B0[15:0] 2: 在0x2读B3B2[15:0]，在0x2写B3B2[15:0] 3: 在0x4读B5B4[15:0]，在0x4写B5B4[15:0] 4: 在0x6读B7B6[15:0]，在0x6写B7B6[15:0]	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6
16	32	4	0x0 / B1B0 0x2 / B3B2 0x4 / B5B4 0x6 / B7B6	1: 在0x0读B1B0[15:0]，在0x0写0000B1B0[31:0] 2: 在0x2读B3B2[15:0]，在0x4写0000B3B2[31:0] 3: 在0x4读B5B4[15:0]，在0x8写0000B5B4[31:0] 4: 在0x6读B7B6[15:0]，在0xC写0000B7B6[31:0]	0x0 / 0000B1B0 0x4 / 0000B3B2 0x8 / 0000B5B4 0xC / 0000B7B6
32	8	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: 在0x0读B3B2B1B0[31:0]，在0x0写B0[7:0] 2: 在0x4读B7B6B5B4[31:0]，在0x1写B4[7:0] 3: 在0x8读BBBAB9B8[31:0]，在0x2写B8[7:0] 4: 在0xC读BFBEBDBC[31:0]，在0x3写BC[7:0]	0x0 / B0 0x1 / B4 0x2 / B8 0x3 / BC

32	16	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: 在0x0读B3B2B1B0[31:0]，在0x0写B1B0[15:0] 2: 在0x4读B7B6B5B4[31:0]，在0x2写B5B4[15:0] 3: 在0x8读BBBAB9B8[31:0]，在0x4写B9B8[15:0] 4: 在0xC读BFBEBDBC[31:0]，在0x6写BDBC[15:0]	0x0 / B1B0 0x2 / B5B4 0x4 / B9B8 0x6 / BDBC
32	**32**	4	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC	1: 在0x0读B3B2B1B0[31:0]，在0x0写B3B2B1B0[31:0] 2: 在0x4读B7B6B5B4[31:0]，在0x4写B7B6B5B4[31:0] 3: 在0x8读BBBAB9B8[31:0]，在0x8写BBBAB9B8[31:0] 4: 在0xC读BFBEBDBC[31:0]，在0xC写BFBEBDBC[31:0]	0x0 / B3B2B1B0 0x4 / B7B6B5B4 0x8 / BBBAB9B8 0xC / BFBEBDBC

16.3.4.1 操作一个不支持字节或半字写的AHB 设备

当DMA模块开始一个AHB设备的字节或半字写操作时，数据将在HWDATA[31:0]总线中未使用的部分重复。因此，如果DMA以字节或半字写入不支持字节或半字写操作的AHB设备时(即HSIZE不适于该模块)，不会发生错误，DMA将按照下面两个例子写入32位HWDATA数据：

- 当HSIZE=半字时，写入半字'0xABCD'，DMA将设置HWDATA总线为'0xABCDABCD'。
- 当HSIZE=字节时，写入字节'0xAB'，DMA将设置HWDATA总线为'0xABABABAB'。

假定AHB/APB桥是一个AHB的32位从设备，它不处理HSIZE参数，它将按照下述方式把任何AHB上的字节或半字按32位传送到APB上：

- 一个AHB上对地址0x0(或0x1、0x2或0x3)的写字节数据'0xB0'操作，将转换到APB上对地址0x0的写字数据'0xB0B0B0B0'操作。
- 一个AHB上对地址0x0(或0x2)的写半字数据'0xB1B0'操作，将转换到APB上对地址0x0的写字数据'0xB1B0B1B0'操作。

16.3.5 错误管理

读写一个保留的地址区域，将会产生DMA传输错误，具体包括

（1） 对CRAM区域（CRAM设置为Read Only时）进行写入

（2） 对Cortex-M3私有外设区进行读写

（3） 在DMA读写时发生的其它总线错误[注]

当在DMA读写操作时发生DMA传输错误，硬件会自动地清除发生错误的信道所对应的信道配置寄存器(DMA_CCRx)的EN位，该通道操作被停止（注：需等待当前Burst传输完成）。此时，在DMA_IFSR寄存器中对应该通道的传输错误中断标志位(TEIF)将被置位，如果在DMA_CCRx寄存器中设置了传输错误中断允许位，则将产生中断。

注：DMA读写保留地址区或非法读写时产生总线矩阵异常，该异常将会送给DMA控制器产生DMA错误。

16.3.6 DMA 中断

每个DMA通道都有4种传输事件，当传输事件发生时可以产生中断。即突发传输结束(BE)、半传输(HT)、传输完成(TC)和传输错误(TE)。DMA配置寄存器CCRx 中设置相关控制位可使能相关中断事件。如果BE、HT、TC 或TE 四个事件中有任意一个事件发生时，将产生中断请求。

在中断向量表中，DMA模块占用3个入口，分别为DMA_CH0、DMA_CH1、DMA_CH2-7。

表16-3 DMA 中断请求

中断事件	事件标志位	使能控制位
突发传输结束	BEIF	BEIE
传输过半	HTIF	HTIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE

16.3.7 DMA 请求映像

DMA控制器从外设(PCAx(x=0~7)、PWMx(x=0~3)、ADC、UARTx(x=0~5)、SPIx(x=1、2)、TWIx(x=1、2)、EXTIx(x=0~15)、TIMx(x=5、6、7、8)、LED、LCD产生的请求，通过输入源选择器（STRMSEL[2:0]）输入到仲裁器，这意味着每一个DMA通道同时只能有一个请求有效。参见下图的DMA请求映像。

表16-4　　各个通道的 DMA 请求一览

DMA输入源 STRMSEL[2:0]	**外设类型**	**DMA Channel 0**	**DMA Channel 1**	**DMA Channel 2**	**DMA Channel 3**	**DMA Channel 4**	**DMA Channel 5**	**DMA Channel 6**	**DMA Channel 7**
stream0	**UART**	UART0_TX	UART0_RX	UART1_TX	UART1_RX	UART2_TX	UART2_RX	UART3_TX	UART3_RX
stream1	**SPI**	SPI0_TX	SPI0_RX	SPI1_TX	SPI1_RX	-	-	-	-
stream2	**EXTI**	EXTI0/1	EXTI2/3	EXTI4/5	EXTI6/7	EXTI8/9	EXTI10/11	EXTI12/13	EXTI14/15
stream3	**TIM**	-	-	-	-	TIM5	TIM6	TIM7	TIM8
stream4	**ADC/ UART**	ADC	ADC	ADC	ADC	UART4_TX	UART4_RX	UART5_TX	UART5_RX
stream5	**PCA**	PCA0	PCA0	PCA1	PCA1	PCA2	PCA2	PCA3	PCA3
stream6	**PCA**	PCA4	PCA4	PCA5	PCA5	PCA6	PCA6	PCA7	PCA7
stream7	**PWM/ LED**	PWM0	PWM1	PWM2	PWM3	LED	LED	LED	LED

说明：DMA支持对GPIO的收发，但触发信号需要通过EXTI、TIM等其它外设产生；TIM5-8在低频RC128KHz、LSE32.768KHz和Tn作为时钟源计数时，不能触发DMA。

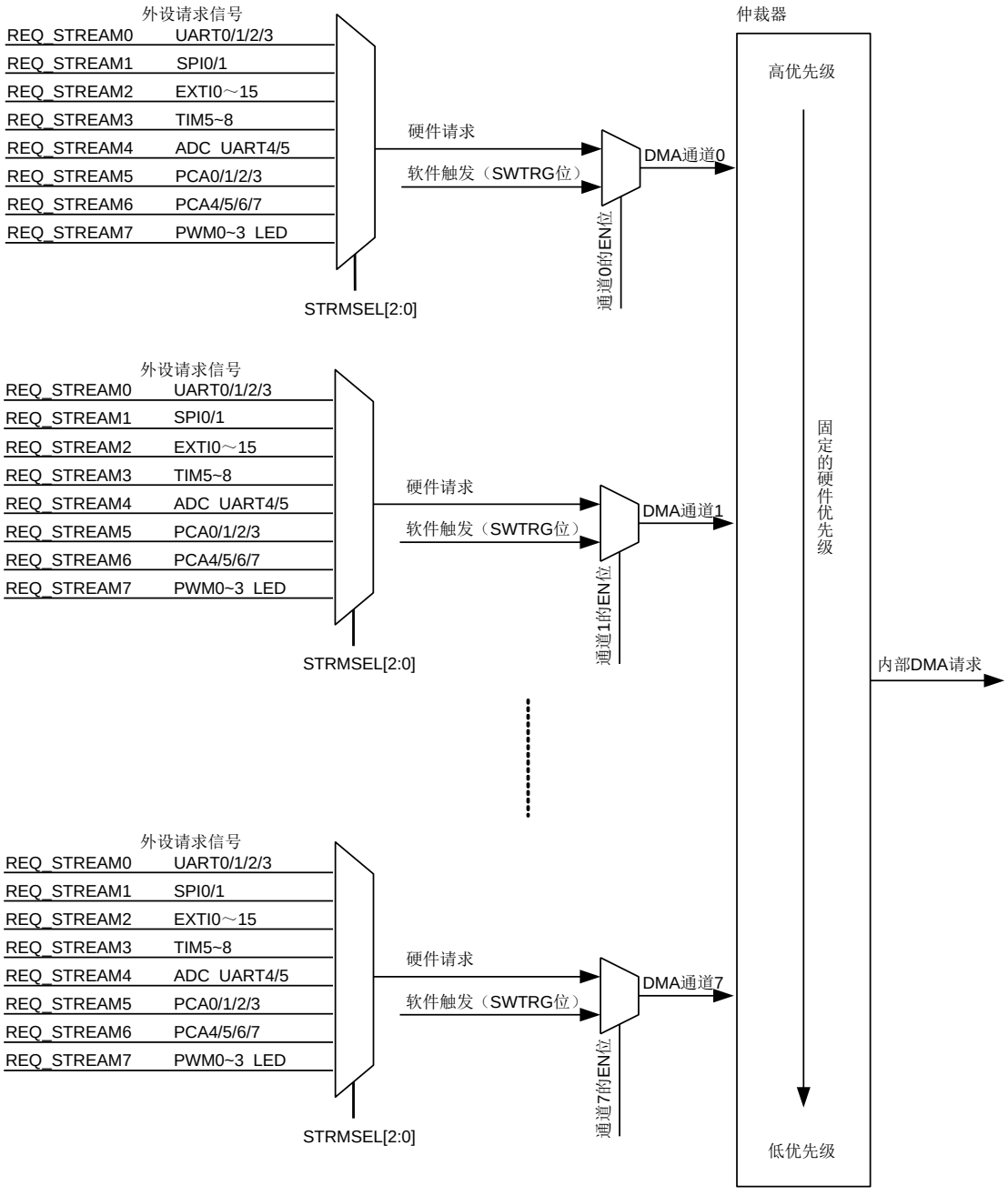


图16-7　　DMA 请求映像图

16.4 寄存器

DMA 模块寄存器列表 (基地址: 0x4004 4000)

地址	寄存器名	说明
0x4004 4000	IFSR	DMA中断状态寄存器
0x4004 4004	IFCR	DMA中断标志清除寄存器
0x4004 4008	CSR	DMA控制状态寄存器
0x4004 4010	CCR0	DMA通道0配置寄存器
0x4004 4014	NPKT0	DMA通道0传输数量寄存器
0x4004 4018	CPKT0	DMA通道0传输计数寄存器
0x4004 401C	SAR0	DMA通道0源地址寄存器
0x4004 4020	DAR0	DMA通道0目标地址寄存器
0x4004 4030	CCR1	DMA通道1配置寄存器
0x4004 4034	NPKT1	DMA通道1传输数量寄存器
0x4004 4038	CPKT1	DMA通道1传输计数寄存器
0x4004 403C	SAR1	DMA通道1源地址寄存器
0x4004 4040	DAR1	DMA通道1目标地址寄存器
0x4004 4050	CCR2	DMA通道2配置寄存器
0x4004 4054	NPKT2	DMA通道2传输数量寄存器
0x4004 4058	CPKT2	DMA通道2传输计数寄存器
0x4004 405C	SAR2	DMA通道2源地址寄存器
0x4004 4060	DAR2	DMA通道2目标地址寄存器
0x4004 4070	CCR3	DMA通道3配置寄存器
0x4004 4074	NPKT3	DMA通道3传输数量寄存器
0x4004 4078	CPKT3	DMA通道3传输计数寄存器
0x4004 407C	SAR3	DMA通道3源地址寄存器
0x4004 4080	DAR3	DMA通道3目标地址寄存器
0x4004 4090	CCR4	DMA通道4配置寄存器
0x4004 4094	NPKT4	DMA通道4传输数量寄存器
0x4004 4098	CPKT4	DMA通道4传输计数寄存器
0x4004 409C	SAR4	DMA通道4源地址寄存器
0x4004 40A0	DAR4	DMA通道4目标地址寄存器
0x4004 40B0	CCR5	DMA通道5配置寄存器
0x4004 40B4	NPKT5	DMA通道5传输数量寄存器
0x4004 40B8	CPKT5	DMA通道5传输计数寄存器
0x4004 40BC	SAR5	DMA通道5源地址寄存器
0x4004 40C0	DAR5	DMA通道5目标地址寄存器
0x4004 40D0	CCR6	DMA通道6配置寄存器
0x4004 40D4	NPKT6	DMA通道6传输数量寄存器
0x4004 40D8	CPKT6	DMA通道6传输计数寄存器
0x4004 40DC	SAR6	DMA通道6源地址寄存器
0x4004 40E0	DAR6	DMA通道6目标地址寄存器
0x4004 40F0	CCR7	DMA通道7配置寄存器
0x4004 40F4	NPKT7	DMA通道7传输数量寄存器
0x4004 40F8	CPKT7	DMA通道7传输计数寄存器
0x4004 40FC	SAR7	DMA通道7源地址寄存器
0x4004 4100	DAR7	DMA通道7目标地址寄存器

16.4.1 DMA 中断状态寄存器 (DMA_IFSR)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TEIFy(y=7~0)								HTIFy(y=7~0)							
RO								RO							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TCIFy(y=7~0)								BEIFy(y=7~0)							
RO								RO							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**TEIFy(y=7~0)**	**通道y的传输错误标志，硬件置位，在DMA_IFCR寄存器的相应位写入1可以清除对应标志。** 0： 在通道y没有传输错误（TE）事件 1：在通道y产生了传输错误（TE）事件
23 ~ 16	**HTIFy(y=7~0)**	**通道y的半传输标志，硬件置位，在DMA_IFCR寄存器的相应位写入1可以清除对应标志。** 0： 在通道y没有半传输（HT）事件 1：在通道y产生了半传输（HT）事件
15 ~ 8	**TCIFy(y=7~0)**	**通道y的传输完成标志，硬件置位，在DMA_IFCR寄存器的相应位写入1可以清除对应标志。** 0： 在通道y没有传输完成（TC）事件 1：在通道y产生了传输完成（TC）事件
7 ~ 0	**BEIFy(y=7~0)**	**通道y的块传输完成中断标志，硬件置位，在DMA_IFCR寄存器的相应位写入1可以清除对应标志。** 0： 在通道y没有块传输完成（BE）事件 1：在通道y产生了块传输完成（BE）事件

16.4.2 DMA 中断标志清除寄存器 (DMA_IFCR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CTEIFy(y=7~0)								CHTIFy(y=7~0)							
WO								WO							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CTCIFy(y=7~0)								CBEIFy(y=7~0)							
WO								WO							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**CTEIFy(y=7~0)**	**清除通道y的传输错误标志，软件清除** 0： 不起作用 1：清除DMA_ISR寄存器中的对应的TEIF标志
23 ~ 16	**CHTIFy(y=7~0)**	**清除通道y的半传输标志，软件清除** 0： 不起作用 1：清除DMA_ISR寄存器中的对应的HTIF标志
15 ~ 8	**CTCIFy(y=7~0)**	**清除通道y的传输完成标志，软件清除** 0： 不起作用 1：清除DMA_ISR寄存器中的对应的TCIF标志
7 ~ 0	**CBEIFy(y=7~0)**	**清除通道y的块传输完成中断标志，软件清除**

		0： 不起作用 1：清除DMA_ISR寄存器中的对应的BEIF标志

16.4.3 DMA 控制状态寄存器 (DMA_CSR)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RELOADy(y=7~0)								保留				BURSTIDLE[3:0]			
RW								-				RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DBUSYy(y=7~0)								SWTRGy(y=7~0)							
RO								RW1s							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**RELOADy(y=7~0)**	**DMA通道y的重载控制位，该位由软件设置和清除** 0：不自动重载 1：自动重载
23 ~ 20	**保留**	**-**
19 ~ 16	**BURSTIDLE[3:0]**	**DMA突发传输后释放周期设置位，该位由软件设置和清除** 0000：释放1个周期 0001：释放2个周期 0010：释放3个周期 0011：释放4个周期 0100：释放5个周期 0101：释放6个周期 0110：释放7个周期 0111：释放8个周期 1000：释放9个周期 1001：释放10个周期 1010：释放12个周期 1011：释放16个周期 1100：释放20个周期 1101：释放24个周期 1110：释放28个周期 1111：释放32个周期
15 ~ 8	**DBUSYy(y=7~0)**	**DMA通道y的状态，由硬件设置与清除，包含软件硬件触发状态** 0：idle 1：busy
7 ~ 0	**SWTRGy(y=7~0)**	**DMA通道y的软件触发，该位由软件设置，硬件清除** 0：无软件触发，可读0，写0无意义 1：置位软件触发，在DMA响应后自动清0

16.4.4 DMA 通道 n 配置寄存器 (DMA_CCRn)(n=0..7)

偏移地址: 0x0010
:0x0030
:0x0050
:0x0070
:0x0090
:0x00B0
:0x00D0
:0x00F0

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留								TRGMODE	STRMSEL[2:0]			BURSTLEN[3:0]			
-								RW	RW			RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PL[1:0]		SSIZE[1:0]		DSIZE[1:0]		SPTYP[1:0]		DPTYP[1:0]		保留	TEIE	BEIE	HTIE	TCIE	EN
RW		RW		RW		RW		RW		-	RW	RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	保留	-
23	TRGMODE	DMA触发模式选择位，该位由软件设置和清除 0：one-shot触发方式 1：to-end触发方式
22 ~ 20	STRMSEL[2:0]	DMA通道输入源选择控制位，这些位由软件设置和清除 000：选择stream0输入 001：选择stream1输入 ..... 111：选择stream7输入
19 ~ 16	BURSTLEN[3:0]	DMA突发长度设置位，该位由软件设置和清除 0000：每个burst包含1个数据（即single传输） 0001：每个burst包含2个数据 … 1111：每个burst包含16个数据 注：根据SIZE设置的不同，1个数据传输可能包含1字节/2字节/4字节。
15 ~ 14	PL[1:0]	通道优先级，这些位由软件设置和清除 00：低 01：中 10：高 11：最高
13 ~ 12	SSIZE[1:0]	源端数据宽度，这些位由软件设置和清除 00：8位 01：16位 10：32位 11：保留
11 ~ 10	DSIZE[1:0]	目标端数据宽度，这些位由软件设置和清除 00：8位 01：16位 10：32位 11：保留
9 ~ 8	SPTYP[1:0]	源端指针修改方式，该位由软件设置和清除 00：递增 01：递减

		10：固定 11：循环（递增循环）
7 ~ 6	**DPTYP[1:0]**	**目标端指针修改方式，该位由软件设置和清除** 00：递增 01：递减 10：固定 11：循环（递增循环）
5	**保留**	**-**
4	**TEIE**	**允许传输错误中断，该位由软件设置和清除** 0：禁止TE中断 1：允许TE中断
3	**BEIE**	**允许burst传输结束中断，该位由软件设置和清除** 0：禁止BE中断 1：允许BE中断 注：burst长度设置为1个数据时（即single模式），同样允许BE中断。
2	**HTIE**	**允许半传输中断，该位由软件设置和清除** 0：禁止HT中断 1：允许HT中断
1	**TCIE**	**允许传输完成中断，该位由软件设置和清除** 0：禁止TC中断 1：允许TC中断
0	**EN**	**通道使能控制位，由软件设置与清除** 0：通道关闭 1：通道开启

16.4.5 DMA 通道 n 传输数量寄存器 (DMA_NPKTn)(n=0..7)

偏移地址: 0x0014
:0x0054
:0x0074
:0x0094
:0x00B4
:0x00D4
:0x00F4

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留					NPKT[10:0]										
-					*RW*										
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 11	保留	-
10 ~ 0	**NPKT[10:0]**	**通道n数据传输数量（burst数）** 数据传输数量为NPKT，设置范围0～2047，表示传输burst数量从1至2048个。这个寄存器只能在通道不工作(DMA_CCRx的EN=0)时写入。通道开启后该寄存器变为只读。

16.4.6 DMA 通道 n 传输计数寄存器 (DMA_CPKTn)(n=0..7)

偏移地址: 0x0018
:0x0038
:0x0058
:0x0078
:0x0098
:0x00B8
:0x00D8
:0x00F8
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留					CPKT[10:0]										
-					RO										
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 11	保留	-
10 ~ 0	CPKT[10:0]	**通道n数据传输计数（burst数）** 该寄存器用于指示DMA传输数量，只读。开启DMA通道后，该计数器自动清0，并根据DMA传输进行计数，当计数到NPKT数量时，停止DMA传输，清除DMA通道使能位，并保持计数值不变。 该寄存器设置范围0～2047，表示传输burst数量从1至2048个。

16.4.7 DMA 通道 n 源地址寄存器 (DMA_SARn)(n=0..7)

偏移地址: 0x001C
:0x003C
:0x005C
:0x007C
:0x009C
:0x00BC
:0x00DC
:0x00FC
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
SAR[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SAR[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	SAR[31:0]	**DMA源地址寄存器 (Source address register)** 源数据寄存器的基地址 当SSIZE=01(16位)，不使用SAR[0]位。操作自动地与半字地址对齐。 当SSIZE=10(32位)，不使用SAR[1:0]位。操作自动地与字地址对齐。 当开启通道(DMA_CCRx的EN=1)时不能写该寄存器

16.4.8 DMA 通道 n 目标地址寄存器 (DMA_DARn)(n=0..7)

偏移地址: 0x0020
:0x0040
:0x0060
:0x0080
:0x00A0
:0x00C0
:0x00E0
:0x0100

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
DAR[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DAR[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	**DAR[31:0]**	**DMA目标地址寄存器 Destination address register)** 目标数据寄存器的基地址 当DSIZE=01(16位)，不使用DAR[0]位。操作自动地与半字地址对齐。 当DSIZE=10(32位)，不使用DAR[1:0]位。操作自动地与字地址对齐。 当开启通道(DMA_CCRx的EN=1)时不能写该寄存器

17. LED 驱动器

17.1 简介

LED驱动器包含一个控制器，12个COM输出引脚和16个SEG输出引脚。支持1/1~1/12占空比电压驱动方式。

LED驱动器支持共阴/共阳模式切换，通过设置CR_LEDMODSEL进行选择。切换LED显示驱动器共阴/共阳模式，只需设置CR_LEDMODSEL，LED驱动器会硬件自动将相关的COM和SEG的电平取反，无需修改其他寄存器设置。

LED驱动器有两种运行模式。

模式1：亮灭LED模式

当LED驱动器工作在亮灭LED模式时，每一个LEDCOMx [15:0]控制一个LED灯，当LEDCOMx[15:0]为0时，LED熄灭，当LEDCOMx [15:0]为1时，LED点亮；在LED一帧或者一个COM扫描结束后，LED驱动器对应的中断标志位LEDIF或者COMIF标志位置1。

模式2：调光LED模式

当LED驱动器工作在调光LED模式时，每一个LEDSEGx [7:0]控制正在扫描的COM周期内SEG的占空比；该占空比总共可以256档可选；当LEDSEGx [7:0]为0xff时，SEG输出最大占空比，当LEDSEGx [7:0]为0x00时，SEG输出最小占空比；当LLEDSEGx [7:0]为0x00~0xff中间值时，SEG输出 相对应的占空比；针对LEDSEGx [7:0]的修改，会在下一个COM扫描周期生效；

在LED每一个COM周期扫描结束后，立即进行下一个COM周期的扫描，LED驱动器对应的中断标志位FR_COMIF会置1，由于LED驱动器工作在模式2，所以在LED驱动使能后，FR_COMIF中断标志位也会置1，方便用户修改LEDSEGx [7:0]的值；在扫描完成一帧后，LED驱动器对应的中断标志位FR_LEDIF会置1；

无论LED驱动器工作在模式1或者模式2，COM和SEG 的非选电平都为浮动电平， COM的输出有效电平为低电平，SEG的输出有效电平为输出高电平。LEDSHARE_LEDCOM[11:0]寄存器控制LED驱动器的COM个数；LEDSHARE_LEDSEG1[7:0]和EDSHARE_LEDSGE2[7：0]寄存器控制LED驱动器SEG的个数；CR_DISCOM[7:0]寄存器可以设置每一个COM周期的宽度；为了防止两个COM显示切换之间的不确定状态，LED驱动器会在2个COM之间插入一段死区时间，在死区时间内，COM输出浮动电平，死区时间可以通过CR_LEDDZ[7:0]寄存器来设置；

在上电复位、引脚复位、低电压复位或看门狗复位期间，LED被关闭。当LED被关闭时，被选中COM和SEG输出为浮动电平。

注意：

模式1举例：

T_B为LED单COM扫描宽度，T_{APB}为APB时钟宽度, T_{LED}为LED扫描时间宽度

*T_B= T_{APB}*256*DISCOM*

*T_{LED} = T_B *S*

S为LED扫描的COM数量：扫描4COM即S=4，扫描5COM即S=5，以此类推。

以需要显示的LED帧频200HZ(5ms)为例,当LED为5COM且APB时钟为24MHz 时：

*T_B= T_{APB} *256* DISCOM*

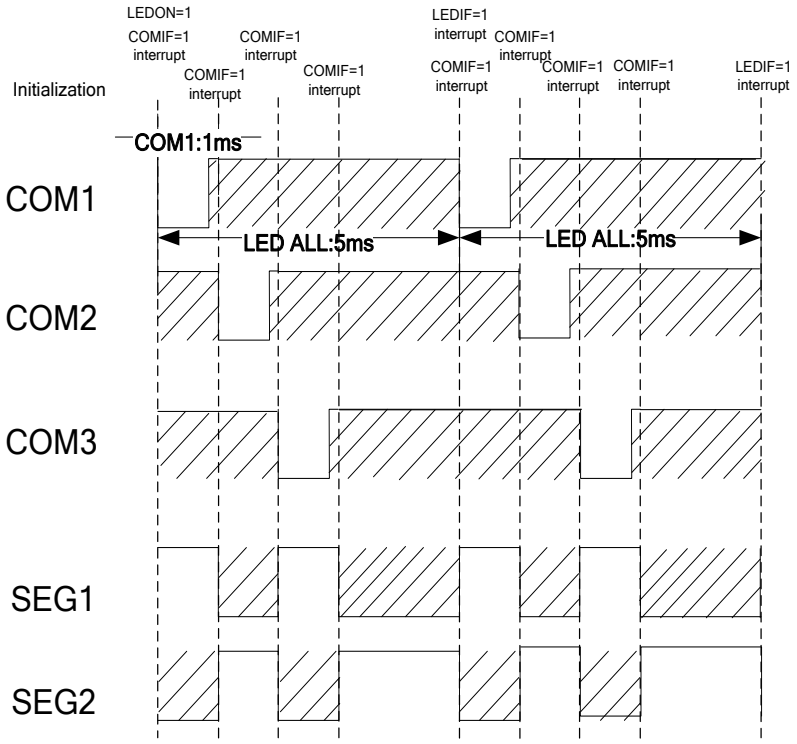
*=41.66ns*256*94(5EH)*

=1002506ns=1002.506u s=1.002ms

需要设置T_{LED} =5ms

*T_{LED}= T_B*S =1.002ms*5=5.010ms*

设置需要开启的中断功能(ELED=1)，可选择帧中断(LEDFY=1) (必须)或者COM中断(LEDCY=1)(可选)。开启LED模块并启动LED扫描，当一帧/一COM波形扫描结束后，相应选择的帧中断(LEDIF=1)/COM中断(COMIF=1)中断标志会置起。(SEG和COM的非选电平都为浮动电平，COM有效电平为低电平，SEG有效电平为高电平。)



模式2举例：

T_B为LED单COM扫描宽度，T_{APB}为APB时钟宽度, T_{LED}为LED扫描时间宽度

$T_B = T_{APB}*256*DISCOM$

$T_{LED} = T_B * S$

S为LED扫描的COM数量：扫描4COM即S=4，扫描5COM即S=5，以此类推。

以需要显示的LED帧频200HZ(5ms)为例,当LED为5COM且APB时钟时钟为24MHz 时：

$T_B = T_{APB} *256* DISCOM$

*=41.66ns*256*94(5EH)*

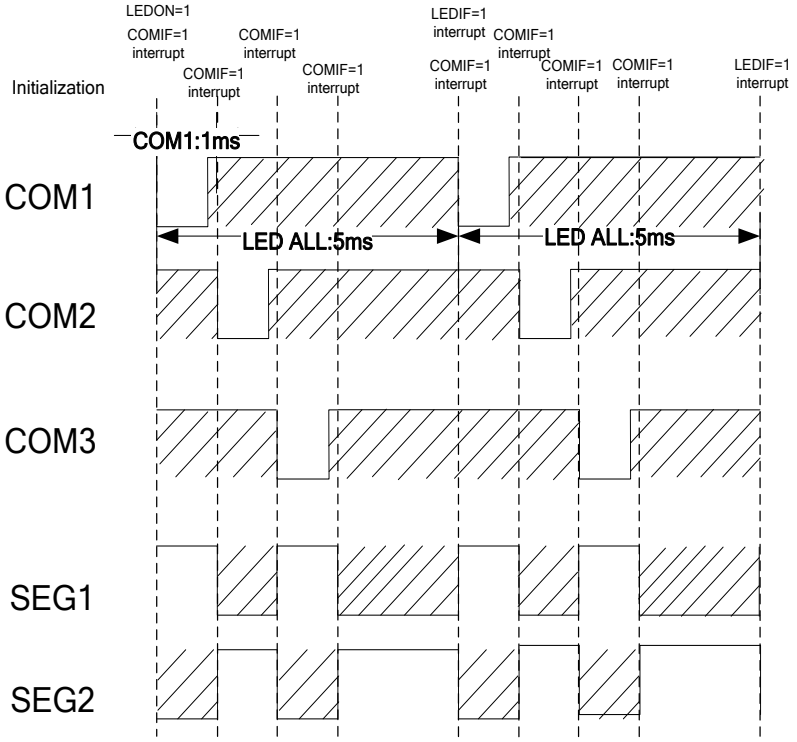
=1002506ns=1002.506u s=1.002ms

需要设置T_{LED} =5ms

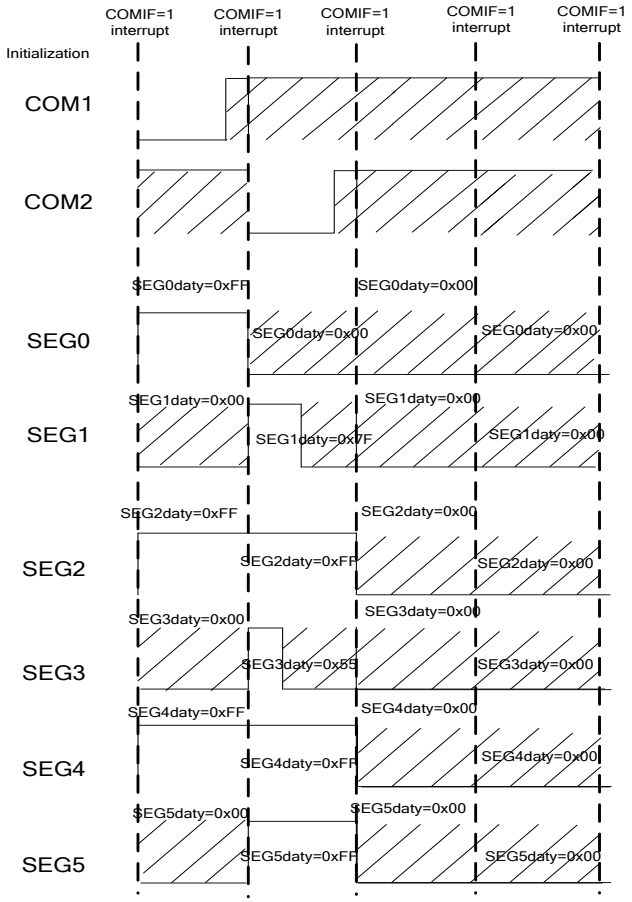
*$T_{LED} = T_B*S$ =1.002ms*5=5.010ms*

1. 第一个COM的SEG 的波形SEGXduty(X=0~16)(如果输出满幅设置0XFF，即第一个COM周期显示100%占空比高电平，如果不显示，设置0X00，即显示浮动电平，如果设置0X7F,SEG输出50%占空比高电平)，设置需要开启的中断功能（ELED=1），需要选择帧COM中断(LEDCY=1)(必须)，或加上帧中断(LEDFY=1)（可选）。开启LED中断和LED模块。

2. 开启LED 功能前，设置第一个COM的波形，启动LED功能，在LED中断(COMIF=1)来临时，改写第二个COM周期的SEG 的波形SEGXduty(X=0~16)。如果不修改，则显示上周期波形。以此类推，到一帧(LEDIF=1)结束。最后第二个COM中断来临时，填写下一帧的第一个COM的SEG 的波形。



模式2调光示意图：


COMIF=1 interrupt
COMIF=1 interrupt
COMIF=1 interrupt
COMIF=1 interrupt
COMIF=1 interrupt
Initialization
COM1
COM2
SEG0daty=0xFF
SEG0daty=0x00
SEG0daty=0x00
SEG0daty=0x00
SEG0
SEG1daty=0x00
SEG1daty=0x00
SEG1daty=0xFF
SEG1daty=0x00
SEG1
SEG2daty=0xFF
SEG2daty=0x00
SEG2daty=0xFF
SEG2daty=0x00
SEG2
SEG3daty=0x00
SEG3daty=0x00
SEG3daty=0x55
SEG3daty=0x00
SEG3
SEG4daty=0xFF
SEG4daty=0x00
SEG4daty=0xFF
SEG4daty=0x00
SEG4
SEG5daty=0x00
SEG5daty=0x00
SEG5daty=0xFF
SEG5daty=0x00
SEG5

17.2 LED RAM 配置

模式1：

寄存器	LEDCOM	7	6	5	4	3	2	1	0
		15	14	13	12	11	10	9	8
		23	22	21	20	19	18	17	16
		31	30	29	28	27	26	25	24
LED_BUFS0	LEDCOM1	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
	LEDCOM2	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
LED_BUFS1	LEDCOM3	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
	LEDCOM4	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
LED_BUFS2	LEDCOM5	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
	LEDCOM6	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
LED_BUFS3	LEDCOM7	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
	LEDCOM8	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
LED_BUFS4	LEDCOM9	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
	LEDCOM10	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
LED_BUFS5	LEDCOM11	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9
	LEDCOM12	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
		SEG16	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9

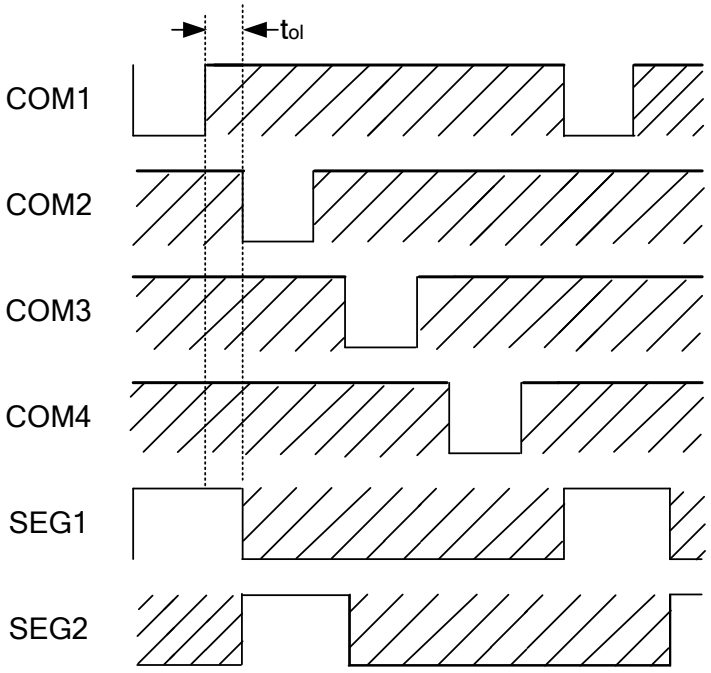
模式2：

寄存器	LEDSEG	7	6	5	4	3	2	1	0
LED_BUFS6	LED_SEG1	DY1.7	DY1.6	DY1.5	DY1.4	DY1.3	DY1.2	DY1.1	DY1.0
	LED_SEG2	DY2.7	DY2.6	DY2.5	DY2.4	DY2.3	DY2.2	DY2.1	DY2.0
	LED_SEG3	DY3.7	DY3.6	DY3.5	DY3.4	DY3.3	DY3.2	DY3.1	DY3.0
	LED_SEG4	DY4.7	DY4.6	DY4.5	DY4.4	DY4.3	DY4.2	DY4.1	DY4.0
LED_BUFS7	LED_SEG5	DY5.7	DY5.6	DY5.5	DY5.4	DY5.3	DY5.2	DY5.1	DY5.0
	LED_SEG6	DY6.7	DY6.6	DY6.5	DY6.4	DY6.3	DY6.2	DY6.1	DY6.0
	LED_SEG7	DY7.7	DY7.6	DY7.5	DY7.4	DY7.3	DY7.2	DY7.1	DY7.0
	LED_SEG8	DY8.7	DY8.6	DY8.5	DY8.4	DY8.3	DY8.2	DY8.1	DY8.0
LED_BUFS8	LED_SEG9	DY9.7	DY9.6	DY9.5	DY9.4	DY9.3	DY9.2	DY9.1	DY9.0
	LED_SEG10	DY10.7	DY10.6	DY10.5	DY10.4	DY10.3	DY10.2	DY10.1	DY10.0
	LED_SEG11	DY11.7	DY11.6	DY11.5	DY11.4	DY11.3	DY11.2	DY11.1	DY11.0
	LED_SEG12	DY12.7	DY12.6	DY12.5	DY12.4	DY12.3	DY12.2	DY12.1	DY12.0
LED_BUFS9	LED_SEG13	DY13.7	DY13.6	DY13.5	DY13.4	DY13.3	DY13.2	DY13.1	DY13.0
	LED_SEG14	DY14.7	DY14.6	DY14.5	DY14.4	DY14.3	DY14.2	DY14.1	DY14.0
	LED_SEG15	DY15.7	DY15.6	DY15.5	DY15.4	DY15.3	DY15.2	DY15.1	DY15.0
	LED_SEG16	DY16.7	DY16.6	DY16.5	DY16.4	DY16.3	DY16.2	DY16.1	DY16.0

注：

可调光LED显示功能：

*SEGx输出波形宽度=(T_{APB}*256*DISCOM)*（SEGxduty/256）*



注意：t_{OL}*为LED Common信号间的不重叠时间，宽度由LEDDZ（LEDDZ>0AH）来配置。图中阴影部分为COM口floating状态*

17.3 寄存器

LED 模块寄存器列表 (基地址: 0x4002 3400)

地址	寄存器名	说明
0x4002 3400	CR	LED控制寄存器
0x4002 3404	FR	LED中断标志和清除寄存器
0x4002 340C	LEDSHARE	LED的IO SHARE控制寄存器
0x4002 3410	LED_BUFS0	LED数据寄存器
0x4002 3414	LED_BUFS1	LED数据寄存器
0x4002 3418	LED_BUFS2	LED数据寄存器
0x4002 341C	LED_BUFS3	LED数据寄存器
0x4002 3420	LED_BUFS4	LED数据寄存器
0x4002 3424	LED_BUFS5	LED数据寄存器
0x4002 3428	LED_BUFS6	LED数据寄存器
0x4002 342C	LED_BUFS7	LED数据寄存器
0x4002 3430	LED_BUFS8	LED数据寄存器
0x4002 3434	LED_BUFS9	LED数据寄存器

17.3.1 LED 控制寄存器 (LED_CR)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留					LEDCDMA	LEDFDMA	CNTMOD	LEDDZ[7:0]							
-					RW	RW	RW	RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DISCOM[7:0]								LEDCY	LEDFY	LEDMODSEL	IE	MODSW	MODE	LEDMD	EN
RW								RW	RW	RW	RW	RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 27	保留	-
26	LEDCDMA	**LED_COM中断DMA允许位** 0：禁止DMA请求 1：允许DMA请求
25	LEDFDMA	**LED帧中断DMA允许位** 0：禁止DMA请求 1：允许DMA请求
24	CNTMOD	**LED扫描计数器模式控制位** 0：当MODSW为1时，LED扫描计数器继续计数 1：当MODSW为1时，LED扫描计数器暂停计数，保留当前COM值，再次打开LED时，从当前COM口继续扫描
23 ~ 16	LEDDZ[7:0]	**LED死区控制选择位** LED死区时间 = APB时钟*LEDDZ[0:7]
15 ~ 8	DISCOM[7:0]	**LED时钟选择位** LED时钟宽度 = APB时钟*256*DISCOM[7:0]
7	LEDCY	**LED_COM中断使能位** 0：禁止LED_COM中断 1：允许LED_COM中断
6	LEDFY	**LED帧中断使能位**

		0：禁止LED帧中断 1：允许LED帧中断
5	**LEDMODSEL**	**LED共阴共阳模式控制位** 0：选择LED工作模式为共阴模式 1: 选择LED工作模式为共阳模式
4	**IE**	**LED中断使能位** 0：禁止LED中断 1：允许LED中断
3	**MODSW**	**LED共享控制位** 0：LED端口控制位有效 1：所有LED共享端口设置为IO
2	**MODE**	**LED模式选择位** 0：模式1（不可调光LED显示） 1: 模式2（带可调光LED显示）
1	**LEDMD**	**LED中断运行模式控制为** 0：LEDIF = 1时，LED继续扫描 1: LEDIF = 1时，LED停止扫描，需要将LEDON置1开始下一帧
0	**EN**	**LED开/关控制位** 0：禁止LED驱动器 1：允许LED驱动器

17.3.2 LED 中断标志和清除寄存器 (LED_FR)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留														LEDIFC	COMIFC
-														*WO*	*WO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留														LEDIF	COMIF
-														*RO*	*RO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 18	保留	-
17	**LEDIFC**	**LED_FRAME中断标志位清除位** 0：无效 1：清除
16	**COMIFC**	**LEDCOM中断标志位清除位** 0：无效 1：清除
15 ~ 2	保留	-
1	**LEDIF**	**LED_FRAME中断标志** 0：无LED帧中断，可由软件清零 1：由硬件置1，表示已经完成一帧的LED扫描
0	**COMIF**	**LEDCOM中断标志** 0：无LED_COM中断标志，可由软件清零 1：由硬件置1，表示已经完成一个LED_COM扫描

17.3.3 LED 的 IO SHARE 控制寄存器 (LED_LEDSHARE)

偏移地址: 0x000C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDSEG2[7:0]								LEDSEG1[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留				LEDCOM[11:0]											
-				RW											
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LEDSEG2[7:0]**	**SEG口模式选择位2（x = 0-7）** 0：作为I/O 1：在LED模块开启时，作为LED的SEG9~16
23 ~ 16	**LEDSEG1[7:0]**	**SEG口模式选择位2（x = 0-7）** 0：作为I/O 1：在LED模块开启时，作为LED的SEG1~8
15 ~ 12	**保留**	**-**
11 ~ 0	**LEDCOM[11:0]**	**COM口模式选择位（x = 0-12）** 0：作为I/O 1：在LED模块开启时，作为LED的COM （LED_C1 - LED_C12）

17.3.4 LED 数据寄存器 (LED_LED_BUFS0)

偏移地址: 0x0010

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDCOM2[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDCOM1[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LEDCOM2[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 0	**LEDCOM1[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

17.3.5 LED 数据寄存器 (LED_LED_BUFS1)

偏移地址: 0x0014

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDCOM4[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDCOM3[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LEDCOM4[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 0	**LEDCOM3[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

17.3.6 LED 数据寄存器 (LED_LED_BUFS2)

偏移地址: 0x0018

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDCOM6[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDCOM5[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LEDCOM6[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 0	**LEDCOM5[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

17.3.7 LED 数据寄存器 (LED_LED_BUFS3)

偏移地址: 0x001C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDCOM8[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDCOM7[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LEDCOM8[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 0	**LEDCOM7[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

17.3.8 LED 数据寄存器 (LED_LED_BUFS4)

偏移地址: 0x0020

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDCOM10[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDCOM9[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LEDCOM10[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 0	**LEDCOM9[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

17.3.9 LED 数据寄存器 (LED_LED_BUFS5)

偏移地址: 0x0024

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDCOM12[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDCOM11[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**LEDCOM12[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 0	**LEDCOM11[15:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

17.3.10 LED 数据寄存器 (LED_LED_BUFS6)

偏移地址: 0x0028

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDSEG4[7:0]								LEDSEG3[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDSEG2[7:0]								LEDSEG1[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LEDSEG4[7:0]**	**SEG显示数据** SEG4占空比
23 ~ 16	**LEDSEG3[7:0]**	**SEG显示数据** SEG3占空比
15 ~ 8	**LEDSEG2[7:0]**	**SEG显示数据** SEG2占空比
7 ~ 0	**LEDSEG1[7:0]**	**SEG显示数据** SEG1占空比

17.3.11 LED 数据寄存器 (LED_LED_BUFS7)

偏移地址: 0x002C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDSEG8[7:0]								LEDSEG7[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDSEG6[7:0]								LEDSEG5[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LEDSEG8[7:0]**	**SEG显示数据** SEG8占空比
23 ~ 16	**LEDSEG7[7:0]**	**SEG显示数据** SEG7占空比
15 ~ 8	**LEDSEG6[7:0]**	**SEG显示数据** SEG6占空比
7 ~ 0	**LEDSEG5[7:0]**	**SEG显示数据** SEG5占空比

17.3.12 LED 数据寄存器 (LED_LED_BUFS8)

偏移地址: 0x0030

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDSEG12[7:0]								LEDSEG11[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDSEG10[7:0]								LEDSEG9[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LEDSEG12[7:0]**	**SEG显示数据** SEG12占空比
23 ~ 16	**LEDSEG11[7:0]**	**SEG显示数据** SEG11占空比
15 ~ 8	**LEDSEG10[7:0]**	**SEG显示数据** SEG10占空比
7 ~ 0	**LEDSEG9[7:0]**	**SEG显示数据** SEG9占空比

17.3.13 LED 数据寄存器 (LED_LED_BUFS9)

偏移地址: 0x0034

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LEDSEG16[7:0]								LEDSEG15[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LEDSEG14[7:0]								LEDSEG13[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LEDSEG16[7:0]**	**SEG显示数据** SEG16占空比
23 ~ 16	**LEDSEG15[7:0]**	**SEG显示数据** SEG15占空比
15 ~ 8	**LEDSEG14[7:0]**	**SEG显示数据** SEG14占空比
7 ~ 0	**LEDSEG13[7:0]**	**SEG显示数据** SEG13占空比

18. LCD 驱动器

18.1 简介

- 支持4*40、5*39、6*38 、8*36
- 支持1/3bias和1/4bias
- 支持1/4、1/5、1/6、1/8duty
- 支持快速充电模式（Fast Charge Mode）以降低功耗
- 未使能的 COM口可以共享成SEG口
- 在上电复位、引脚复位、低电压复位或看门狗复位期间，LCD被关闭。当LCD被关闭时， Common和Segment都输出低电平
- 支持内部电阻串分压方式实现Bias电压
- 支持在STOP模式下运行

传统电阻型LCD显示模式有以下特性：

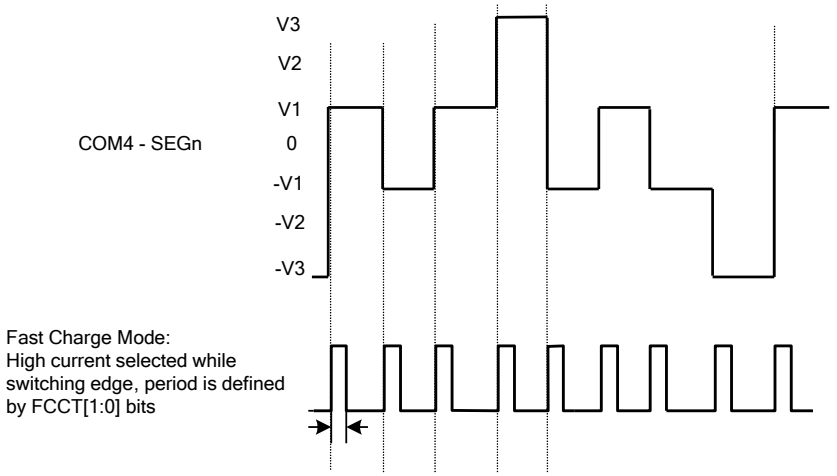
- LCD的驱动电压VLCD由VDD供电，VDD由CR寄存器的VOL[3:0]控制16级对比度调节，再由内部电阻分压得到对应的V1，V2，V3
- 帧频为64Hz；
- 支持快速充电模式（Fast Charge Mode）以降低功耗。

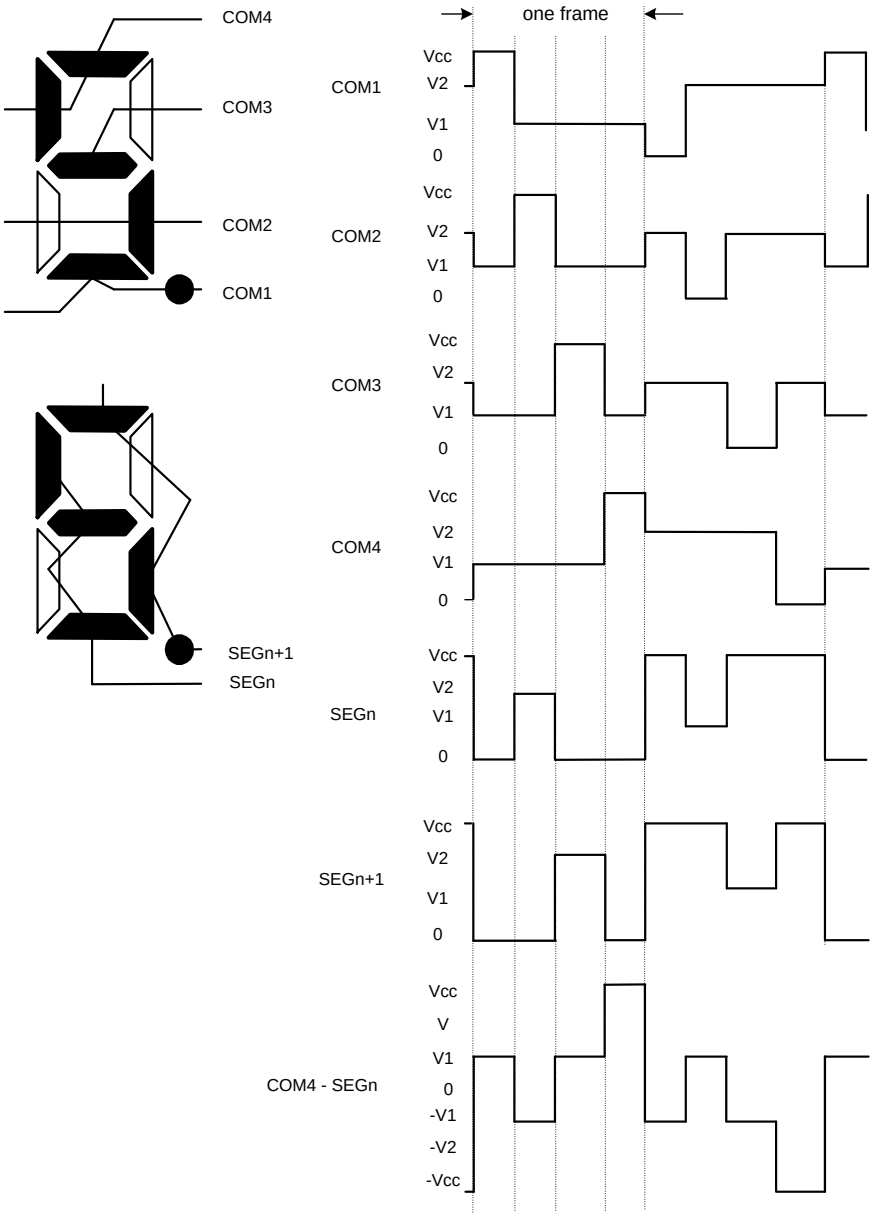
由LCD_CR寄存器的MOD[1:0]位控制可选择LCD偏置电阻（R_{LCD}）为60k或900k/1.5M。选择60k偏置电阻可以得到较好的显示效果，但电流相对会大一些，不适合低功耗的应用。将LCD_CR的MOD[1:0]位设置为00选择900k/1.5M偏置电阻，虽然可以达到较低的功耗，但LCD显示效果会变得差一些。900k和1.5M的选择由MOD [1:0]位控制。

因此，MCU提供了兼顾低功耗和显示效果的显示模式：快速充电模式。设置MOD [10]= 10可以选择此种显示方式，在显示数据刷新时刻选择60k偏置电阻，提供较大的驱动电流，在数据保持期间选择900k/1.5M偏置电阻，提供较小的驱动电流。

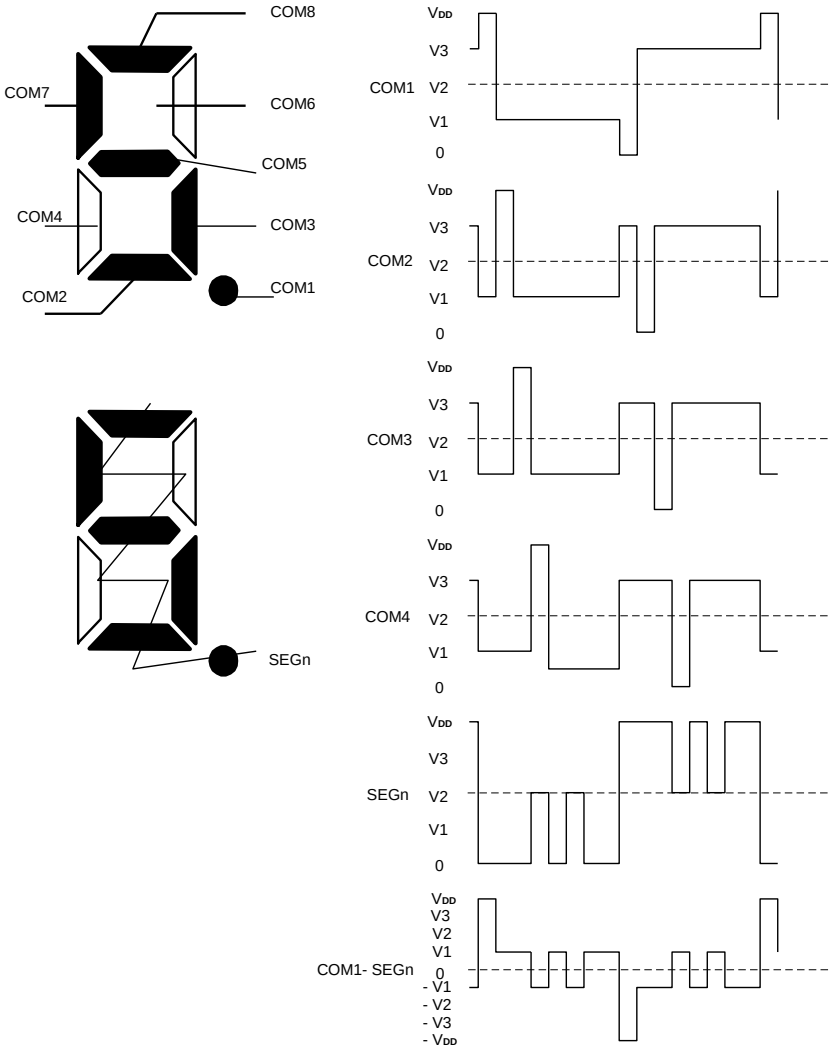
快速充电模式充电时间为LCD com周期的1/8、1/16、1/32或1/64可选，由LCD_CR寄存器的FCCTL[1:0]位选择。

快速充电模式波形





LCD波形（1/4占空比，1/3偏置）



LCD波形（1/8占空比，1/4偏置）

18.2 LCD RAM 配置

寄存器		7	6	5	4	3	2	1	0
		COM8	COM7	COM6	COM5	COM4	COM3	COM2	COM1
LCD_BUFS0	LCD_BUF0	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1
	LCD_BUF1	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2
	LCD_BUF2	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3
	LCD_BUF3	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4
LCD_BUFS1	LCD_BUF4	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5
	LCD_BUF5	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6
	LCD_BUF6	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7
	LCD_BUF7	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8
LCD_BUFS2	LCD_BUF8	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9
	LCD_BUF9	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10
	LCD_BUF10	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11
	LCD_BUF11	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12
LCD_BUFS3	LCD_BUF12	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13
	LCD_BUF13	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14
	LCD_BUF14	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15
	LCD_BUF15	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16
LCD_BUFS4	LCD_BUF16	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17
	LCD_BUF17	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18
	LCD_BUF18	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19
	LCD_BUF19	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20
LCD_BUFS5	LCD_BUF20	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21
	LCD_BUF21	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22
	LCD_BUF22	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23
	LCD_BUF23	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24
LCD_BUFS6	LCD_BUF24	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25
	LCD_BUF25	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26
	LCD_BUF26	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27
	LCD_BUF27	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28
LCD_BUFS7	LCD_BUF28	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29
	LCD_BUF29	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30	SEG30
	LCD_BUF30	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31	SEG31
	LCD_BUF31	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32	SEG32
LCD_BUFS8	LCD_BUF32	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33	SEG33
	LCD_BUF33	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34	SEG34
	LCD_BUF34	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35	SEG35
	LCD_BUF35	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36	SEG36
LCD_BUFS9	LCD_BUF36	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37	SEG37
	LCD_BUF37	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38	SEG38
	LCD_BUF38	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39	SEG39
	LCD_BUF39	SEG40	SEG40	SEG40	SEG40	SEG40	SEG40	SEG40	SEG40

18.3 寄存器

LCD 模块寄存器列表 (基地址: 0x4002 3800)

地址	寄存器名	说明
0x4002 3800	CR	LCD控制寄存器
0x4002 3804	LCDSHARE	LCD的IO SHARE控制寄存器
0x4002 3808	LCDSHARE1	LCD的IO SHARE控制寄存器
0x4002 3810	LCD_BUFS0	LCD数据寄存器
0x4002 3814	LCD_BUFS1	LCD数据寄存器
0x4002 3818	LCD_BUFS2	LCD数据寄存器
0x4002 381C	LCD_BUFS3	LCD数据寄存器
0x4002 3820	LCD_BUFS4	LCD数据寄存器
0x4002 3824	LCD_BUFS5	LCD数据寄存器
0x4002 3828	LCD_BUFS6	LCD数据寄存器
0x4002 382C	LCD_BUFS7	LCD数据寄存器
0x4002 3830	LCD_BUFS8	LCD数据寄存器
0x4002 3834	LCD_BUFS9	LCD数据寄存器

18.3.1 LCD 控制寄存器 (LCD_CR)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留													DCK[1:0]		CNTMOD
-													*RW*		*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
FCCTL[1:0]		MOD[1:0]		RLCD	MODSW	ELCC	VOL[3:0]				DUTY[2:0]			LCLK	EN
RW		*RW*		*RW*	*RW*	*RW*	*RW*				*RW*			*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 19	**保留**	**-**
18 ~ 17	**DCK[1:0]**	**LCD时钟选择控制位** 00：1/4分频 01：1/3分频 10：1/2分频 11：1/1分频 注意：此两位只在LCD时钟选择LSI时有效
16	**CNTMOD**	**LCD扫描计数器模式控制位** 0：当MODSW为1时，LCD扫描计数器继续计数 1：当MODSW为1时，LCD扫描计数器暂停计数，保留当前COM值，再次打开LCD时，从当前COM口继续扫描
15 ~ 14	**FCCTL[1:0]**	**充电时间控制位** 00：1/8 LCD com周期 01：1/16 LCD com周期 10：1/32 LCD com周期 11：1/64 LCD com周期
13 ~ 12	**MOD[1:0]**	**驱动模式选择位** 00：传统电阻型模式，偏置电阻总和为900k/1.5M 01：传统电阻型模式，偏置电阻总和为60k 10：快速充电模式，偏置电阻总和自动在60k和900k/1.5M之间切换 11：无意义

11	**RLCD**	**LCD偏置电阻选择控制位** 0：LCD偏置电阻为900k 1：LCD偏置电阻总和为1.5M
10	**MODSW**	**LCD共享总选择位** 0：LCD端口控制位有效 1：所有LCD共享端口设置为IO
9	**ELCC**	**LCD对比度控制使能位** 0：关闭LCD对比度控制 1：打开LCD对比度控制
8 ~ 5	**VOL[3:0]**	**LCD对比度控制位** 0000：VLCD = 0.531VDD 0001：VLCD = 0.563VDD 0010：VLCD = 0.594VDD 0011：VLCD = 0.625VDD 0100：VLCD = 0.656VDD 0101：VLCD = 0.688VDD 0110：VLCD = 0.719VDD 0111：VLCD = 0.750VDD 1000：VLCD = 0.781VDD 1001：VLCD = 0.813VDD 1010：VLCD = 0.844VDD 1011：VLCD = 0.875VDD 1100：VLCD = 0.906VDD 1101：VLCD = 0.938VDD 1110：VLCD = 0.969VDD 1111：VLCD = 1.000VDD
4 ~ 2	**DUTY[2:0]**	**LCD占空比选择位** 000：1/4占空比，1/3偏置（4 COM X 40 SEG） COM口：COM1-4 SEG口：SEG1-36 COM5-8共享为SEG37-40 001：1/8占空比，1/4偏置（8 COM X 36 SEG） COM口：COM1-8 SEG口：SEG1-36 010：1/4占空比，1/3偏置（4 COM X 40 SEG） COM口：COM5-8共享为COM1-4 SEG口：SEG1-36，COM1-4共享为SEG37-40 011：1/5占空比，1/3偏置（5 COM X 39 SEG） COM口：COM1-5 SEG口：SEG1-36，COM6-8共享为SEG37-39 100：1/6占空比，1/3偏置（6 COM X 38 SEG） COM口：COM1-6 SEG口：SEG1-36，COM7-8共享为SEG37-SEG38 101：1/6占空比，1/4偏置（6 COM X 39 SEG） COM口：COM1-6 SEG口：SEG1-36，COM7-8共享为SEG37-SEG38 其它：1/4占空比，1/3偏置（4 COM X 40 SEG） COM口：COM1-4 SEG口：SEG1-36 COM5-8共享为SEG37-40
1	**LCLK**	**LCD时钟选择位** 0：LCD时钟源=LSI

		1: LCD时钟源=32.768Khz Crystal
0	**EN**	**LCD开/关控制位** 0：禁止LCD驱动器 1：允许LCD驱动器

18.3.2 LCD 的 IO SHARE 控制寄存器 (LCD_LCDSHARE)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDSEG3[7:0]								LCDSEG2[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDSEG1[7:0]								LCDCOM[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDSEG3[7:0]**	**SEG口模式选择位3（x = 0-7）** 0：作为I/O 1：在LCD模块开启时，作为LCD的SEG17~24
23 ~ 16	**LCDSEG2[7:0]**	**SEG口模式选择位2（x = 0-7）** 0：作为I/O 1：在LCD模块开启时，作为LCD的SEG9~16
15 ~ 8	**LCDSEG1[7:0]**	**SEG口模式选择位1（x = 0-7）** 0：作为I/O 1：在LCD模块开启时，作为LCD的SEG1~7
7 ~ 0	**LCDCOM[7:0]**	**COM口模式选择位（x = 0-7）** 0：作为I/O 1：在LCD模块开启时，作为LCD的COM （COM1-COM8）

18.3.3 LCD 的 IO SHARE 控制寄存器 (LCD_LCDSHARE1)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留				LCDSEG5[3:0]				LCDSEG4[7:0]							
-				RW				RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 12	保留	-
11 ~ 8	LCDSEG5[3:0]	SEG口模式选择位5（x = 0-7） 0：作为I/O 1：在LCD模块开启时，作为LCD的SEG33~36
7 ~ 0	LCDSEG4[7:0]	SEG口模式选择位4（x = 0-7） 0：作为I/O 1：在LCD模块开启时，作为LCD的SEG25~32

18.3.4 LCD 数据寄存器 (LCD_LCD_BUFS0)

偏移地址: 0x0010

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF3[7:0]								LCDBUF2[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF1[7:0]								LCDBUF0[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	LCDBUF3[7:0]	SEG显示数据 0：对应单元不显示 1：对应单元显示
23 ~ 16	LCDBUF2[7:0]	SEG显示数据 0：对应单元不显示 1：对应单元显示
15 ~ 8	LCDBUF1[7:0]	SEG显示数据 0：对应单元不显示 1：对应单元显示
7 ~ 0	LCDBUF0[7:0]	SEG显示数据 0：对应单元不显示 1：对应单元显示

18.3.5 LCD 数据寄存器 (LCD_LCD_BUFS1)

偏移地址: 0x0014

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF7[7:0]								LCDBUF6[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF5[7:0]								LCDBUF4[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF7[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF6[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF5[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF4[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

18.3.6 LCD 数据寄存器 (LCD_LCD_BUFS2)

偏移地址: 0x0018

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF11[7:0]								LCDBUF10[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF9[7:0]								LCDBUF8[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF11[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF10[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF9[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF8[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

18.3.7 LCD 数据寄存器 (LCD_LCD_BUFS3)

偏移地址: 0x001C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF15[7:0]								LCDBUF14[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF13[7:0]								LCDBUF12[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF15[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF14[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF13[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF12[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

18.3.8 LCD 数据寄存器 (LCD_LCD_BUFS4)

偏移地址: 0x0020

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF19[7:0]								LCDBUF18[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF17[7:0]								LCDBUF16[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF19[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF18[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF17[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF16[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

18.3.9 LCD 数据寄存器 (LCD_LCD_BUFS5)

偏移地址: 0x0024

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF23[7:0]								LCDBUF22[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF21[7:0]								LCDBUF20[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF23[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF22[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF21[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF20[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

18.3.10 LCD 数据寄存器 (LCD_LCD_BUFS6)

偏移地址: 0x0028

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF27[7:0]								LCDBUF26[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF25[7:0]								LCDBUF24[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF27[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF26[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF25[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF24[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

18.3.11 LCD 数据寄存器 (LCD_LCD_BUFS7)

偏移地址: 0x002C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF31[7:0]								LCDBUF30[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF29[7:0]								LCDBUF28[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF31[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF30[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF29[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF28[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

18.3.12 LCD 数据寄存器 (LCD_LCD_BUFS8)

偏移地址: 0x0030

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF35[7:0]								LCDBUF34[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF33[7:0]								LCDBUF32[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF35[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF34[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF33[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF32[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

18.3.13 LCD 数据寄存器 (LCD_LCD_BUFS9)

偏移地址: 0x0034

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LCDBUF39[7:0]								LCDBUF38[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LCDBUF37[7:0]								LCDBUF36[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	**LCDBUF39[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
23 ~ 16	**LCDBUF38[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
15 ~ 8	**LCDBUF37[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示
7 ~ 0	**LCDBUF36[7:0]**	**SEG显示数据** 0：对应单元不显示 1：对应单元显示

19. 基本定时器（Basic Timer，TIM5～TIM8）

19.1 简介

SH32F9001_32F9801内置4个基本的16 位定时器，可级联成两个32位的定时器。每个时钟最大能以APB0时钟（PCLK0）运行，定时器可以在SHUTDOWN模式下运行（时钟源为内建128kHz RC或者LSE）。

主要特性

- 16位×4 个定时器
- 各定时器进行递增计数
- 各个定时器可以选择独立的时钟源
- 每个定时器可以选择外部时钟源，可做为计数器
- 能够同步运行
- 能够级联成两个32位定时器

19.2 功能

○：能　　—：不能

PCLK0为APB0时钟，ICLK为时钟源

项目	TIM5	TIM6	TIM7	TIM8
时钟源（ICLK）	(1) PCLK0 (2) T5引脚clock输入 (3) 128kHz WDT RC (4) 32kHz Cry（LSE） (5) TIM6（级联）	(1) PCLK0 (2) T6引脚clock输入 (3) 128kHz WDT RC (4) 32kHz Cry（LSE）	(1) PCLK0 (2) T7引脚clock输入 (3) 128kHz WDT RC (4) 32kHz Cry（LSE） (5) TIM8（级联）	(1) PCLK0 (2) T8引脚clock输入 (3) 128kHz WDT RC (4) 32kHz Cry（LSE）
时钟源分频	ICLK/(PSQ[15:0]+1)	ICLK/(PSQ[15:0]+1)	ICLK/(PSQ[15:0]+1)	ICLK/(PSQ[15:0]+1)
周期设定寄存器	TIM5.TPR	TIM6.TPR	TIM7.TPR	TIM8.TPR
输入引脚	T5	T6	T7	T8
输出引脚	T5	T6	T7	T8
级联运行	○	○	○	○
DMA启动	全部的中断源			
A/D转换开始触发	—	—	•TCNT溢出 （**TIM7.TF**）	—
中断源	• TCNT溢出 （**TIM5.TF**）	• TCNT溢出 （**TIM6.TF**）	• TCNT溢出 （**TIM7.TF**）	• TCNT溢出 （**TIM8.TF**）

注：(1) 当32kHz Cry作为TIM时钟，如果出现停振，则CSM切换时钟不能切换成内建32kHz RC；

(2) TIM5/6/7/8中的时钟源128kHz WDT RC是常开的，且精度为±50%；

19.2.1 计数器/定时器运行

SH32F9001_32F9801有4个基本定时器，每个定时器有一个16位数据寄存器TIMx_TCNT（x=5~8），有一个2bit时钟源选择器CLKS[1:0]@TIMx_CR，有一个时钟源分频器TIMx_PSQ[15:0]（x=5~8）。TIMx_CR寄存器的IE位置1能允许定时器TIMx_TCNT（x=5~8）中断。

当开始计时后，定时器开始递增计数，当达到寄存器TIMx.TPR（x=5~8）时溢出，系统置起定时器溢出标志TFx，TIMx_TCNT（x=5~8）被清除。如果定时器x中断被允许，将会产生一个中断。

CLKS[1:0]@TIMx位选择定时器的时钟源：可以选择PCLK0、定时器n输入引脚（Tn）的下降沿、32kHz Cry（LSE）、128kHz WDT RC和另外一个TIMER的溢出标志位。其中只有TIM5和TIM7可以分别选择TIM6和TIM8的溢出标志位作为时钟源（级联）。

STR@TIMx置1时不复位定时器，这意味着如果STR@TIMx位置1，定时器寄存器将从TIMx_TCNT开始计数。所以在允许定时器之前，应该设定定时器寄存器的初始值。

当作为定时器应用时，可配置寄存器TIMx_CR中的TC位使定时器x溢出时Tn脚输出电平自动翻转。如果TC被置1，Tn引脚自动设置为输出。

注1：当写入TIMx_TCNT初值大于TIMx_TPR时，TIMx_TCNT会继续向上计数，直到计到0xFFFF后回到0x0000。

注2：定时器溢出指的是TIMx_TCNT递增计数过程中与TIMx_TPR相等时。

注3：预分频寄存器只能在定时器停止的时候能够写。

*注4：Tn输入的最快频率1/2*PCLK0，滤波时钟选择PCLK0。*

注5：Tn输出电平的初始状态为低电平。

下图为定时器框图，另外TIM5/6/7/8可以唤醒低功耗。

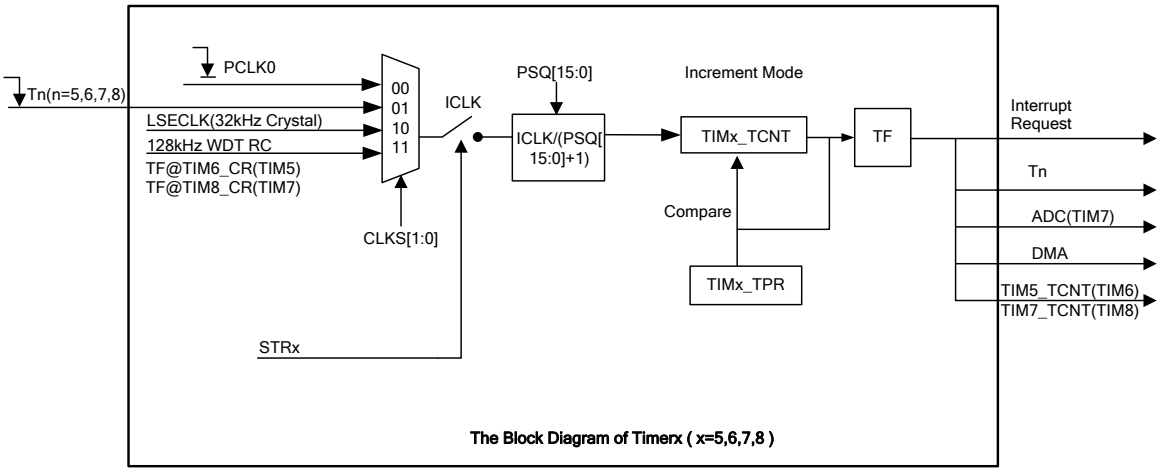


图19-1　　定时器框图

注1：定时器5/6/7/8可以工作在低功耗模式，并且可以以级联的方式唤醒低功耗。

注2：在Stop模式下，Tn口也可以工作，Tn口的滤波用一个128K做一个滤波。

注3：修改TIMx_TCNT时，请确保STR@ TIMx_CR为0（定时器5/6/7，开启后只需配置STR@TIMx_CR=0即可关闭定时器）；

注4：对于定时器8来说，当PPRE0[2:0]@RCC_CFGR ≠1时即PCLK0时钟非2分频时，定时器开启后只需配置STR@TIMx_CR=0即可关闭定时器；

注5：对于定时器8来说，当PPRE0[2:0]@RCC_CFGR = 1 即PCLK0时钟2分频时，定时器开启后配置STR@ TIMx_CR = 0无效，需要设置TIM8RST@RCC_APB0RSTR = 1后重新配置定时器8的所有寄存器。

如果CLKS[1:0]@TIMx_CR为00，定时器x不能工作在低功耗模式下。如果CLKS[1:0]@TIMx_CR为01，Tn端口输入外部时钟，定时器x可以工作在普通模式或低功耗模式。当CLKS[1:0]@TIMx_CR为10时，定时器x选择LSE 32K Cry时钟源，定时器x可以工作在普通模式或低功耗模式。当CLKS[1:0]@TIMx_CR为11时，定时器x选择128kHz WDT RC时钟源，定时器x可以工作在普通模式或低功耗模式。

19.2.2　级联运行

级联运行是将2个通道的16位计数器连接为32位计数器的功能。

通过CASCEN@TIMx_CR（x为5或7）设为1将TIMx（x为5或7）的计数器时钟设定为通过TIMx_TCNT（x为6或8）的溢出进行计数，实现级联运行的功能。

级联的两种组合如下表：

组合	高16位	低16位
定时器5和定时器6	TIM5	TIM6
定时器7和定时器8	TIM7	TIM8

计数器级联运行的例子：

设定TIM5的定时器的级联时钟源为TIM6的溢出，设定TIM5的时钟源为T5的下降沿，计数如下图：

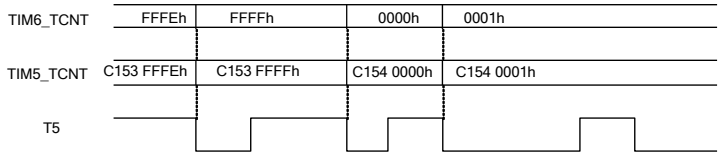


图19-2　　级联运行示例图

注1：级联之后，TIM5_TCNT，TIM5_TPR，TIM5_PSQ的高16位寄存器有效，变成一个32bit的Timer，所以读取只需要读取TIM5的32位寄存器。同时Timer6的相关寄存器全部失效。TIM7与TIM8也一样。

注2：当级联作为32位Timer使用时，仅需开启高位TIM(TIM5和TIM7)的STR@CR位即可。

19.3　寄存器

TIMER5-8 模块寄存器列表 (基地址: 0x4000 0400)

地址	寄存器名	说明
0x4000 0400 + 0x400*(x-5)	CR	TIMx控制寄存器 (TIM5～8对应:0~3)
0x4000 0404 + 0x400*(x-5)	TCNT	TIMx计数寄存器 (TIM5～8对应:0~3)
0x4000 0408 + 0x400*(x-5)	TPR	TIMx周期寄存器 (TIM5～8对应:0~3)
0x4000 040C + 0x400*(x-5)	PSQ	TIMx预分频寄存器 (TIM5～8对应:0~3)
0x4000 0410 + 0x400*(x-5)	TIMINTF	TIMx中断标志和清除寄存器 (TIM5～8对应:0~3)

19.3.1 TIM 控制寄存器 (TIMx_CR)

偏移地址: 0x0000 + 0x400*(x-5)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CASCEN	保留				TC	ETEN	TRIGEN	保留	IE	CLKS[1:0]		OPM	保留		STR
RW	-				RW	RW	RW	-	RW	RW		RW	-		RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15	CASCEN	**TIMx级联(cascade)使能位(x=5,7有效，只在级联的高位TIMER有效)** 0：禁止TIM5/TIM6或者TIM7/TIM8级联功能 1：使能TIM5/TIM6或者TIM7/TIM8级联功能 注：只有在级联的两个TIMER都停止计数时才能修改此位
14 ~ 11	保留	-
10	TC	**比较输出功能允许位** 0：禁止定时器x比较输出功能 1：允许定时器x比较输出功能 注：当定时器允许比较输出时(TC=1)，时钟源不能选择Tn(n=5,6,7,8)口。
9	ETEN	**TIMx溢出事件使能位** 0：禁止定时器溢出唤醒CM3内核 1：允许定时器溢出唤醒CM3内核 注：级联之后为32位溢出时才允许唤醒
8	TRIGEN	**TIMx溢出触发外部模块使能位** 0：定时器x溢出禁止触发其他模块 1：定时器x溢出允许触发其他模块 注1：其他模块有DMA、ADC 注2：所有模块都能触发DMA，但是只有TIM7能够触发ADC
7	保留	-
6	IE	**TIMx溢出中断使能位** 0：禁止定时器TIMx溢出中断 1：允许定时器TIMx溢出中断
5 ~ 4	CLKS[1:0]	**TIMx时钟源（ICLK）选择位** 00：内部APB0时钟（PCLK0） 01：Tn(n=5,6,7,8)口输入，自动上拉 10：外接32kHz Crystal（LSECLK） 11：内部低频RC振荡器(内建128kHz WDT RC) (精度为±50%)
3	OPM	**单脉冲模式** 0：定时器x溢出时，定时器不停止 1：定时器x溢出时，定时器停止 注：级联时高16位TIM的此位无效，只读属性，与低位TIM的此位一样 当CLKS[1:0]=01/10/11，且进入Stop模式时，单脉冲模式开启，单次计数结束后，STR仍为1，需要设置STR=0；然而，不进入Stop模式，则STR=0
2 ~ 1	保留	-
0	STR	**TIMx开始位(x=5,6,7,8)** 注1：可以软件触发，也可以由外部模块触发 注2：级联时高16位TIM的此位无效，只读属性，与低位TIM的此位一样

19.3.2 TIM 计数寄存器 (TIMx_TCNT)

偏移地址: 0x0004 + 0x400*(x-5)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TCNTH[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TCNTL[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**TCNTH[15:0]**	**定时器计数值高16位： 只有TIM5和TIM7位有效** 注：只有级联时有效
15 ~ 0	**TCNTL[15:0]**	**定时器计数值低16位： 全部TIM有效**

19.3.3 TIM 周期寄存器 (TIMx_TPR)

偏移地址: 0x0008 + 0x400*(x-5)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TPRH[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TPRL[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**TPRH[15:0]**	**定时器自动重载高16位： 只有TIM5和TIM7位有效** 注：只有级联时有效
15 ~ 0	**TPRL[15:0]**	**定时器自动重载寄存器** 注：低16位，全部TIM有效

19.3.4 TIM 预分频寄存器 (TIMx_PSQ)

偏移地址: 0x000C + 0x400*(x-5)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
PSQH[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PSQL[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**PSQH[15:0]**	**定时器分频器高16位： 只有TIM5和TIM7位有效** 注：只有级联时有效。
15 ~ 0	**PSQL[15:0]**	**定时器分频器低16位：全部TIM有效** 注：定时器时钟为(ICLK/(PSQ[15:0]+1))

注：当写入TCNT值时，PSQ的内部CNT值也要清零。

19.3.5 TIMx 中断标志和清除寄存器 (TIMx_TIMINTF)

偏移地址: 0x0010 + 0x400*(x-5)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															TFC
-															*WO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留															TF
-															*RO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 17	保留	-
16	**TFC**	**TIMx溢出标志位清除位** 0：无效 1：清除
15 ~ 1	保留	-
0	**TF**	**TIMx溢出标志位** 0：定时器x无溢出，可由软件清0 1：定时器x溢出，由硬件置1 注：级联之后，32位溢出才置起标志位

20. 16bit 脉冲宽度调制（PWM0/1/2/3）

20.1 特性

- 4×2路带死区控制的互补输出
- 提供每个PWMx（x = 0,1,2,3）周期溢出中断
- 两路输出极性可独立选择
- 提供出错侦测功能可紧急关闭PWM输出
- 提供保护寄存器可使重要寄存器免受干扰出错

SH32F9001_32F9801集成了四个16位PWM模块。PWM模块可以产生周期和占空比分别可调整的脉宽调制波形。

如果EFLT置位，PWMx（x = 0,1,2,3）输出能由FLT引脚的输入信号变化自动关闭。

PWM定时器也为PWMx（x = 0,1,2,3）提供4个中断源，在每个PWM周期都会产生中断。这样用户可以实现每个PWM周期中更改下一次循环的周期或占空比。

20.2 PWM 使能控制

当PWMEN清0后，PWM输出立即关闭。

FLT端口主要用于检测异常信号，快速关闭PWM输出。FLT探测到故障后，由硬件执行使PWM输出关闭，所以当故障发生后，它可以快速响应，使得PWM输出无效以保护连接PWM的大功率器件。FLT引脚没有内建上拉电阻。

如果EFLT位清0，则表示FLT端口对PWM定时器输出控制无效。

详情请见PWM控制寄存器章节。

20.3 PWM 保护寄存器

PWM保护寄存器用来控制对PWM控制寄存器、PWM周期寄存器、PWM占空比寄存器和PWM死区时间控制寄存器的更改。只有当保护寄存器中的数据为5AA5H时，才允许修改这些寄存器内容，否则不能修改。

这个寄存器能增强SH32F9001_32F9801的抗干扰能力。

详情请见PWM保护寄存器章节。

20.4 16 位 PWM 定时器

SH32F9001_32F9801包含四个16位PWM模块。PWM模块可以产生周期和占空比分别可调整的脉宽调制波形。PWMCR寄存器用于控制PWM模块的时钟，PWMPR寄存器用于控制PWM输出波形的周期，PWMDR寄存器用于控制PWM模块输出波形的占空比。

在PWM输出允许期间可以修改这三个寄存器，但在下一个PWM周期修改才会起作用。

注意：

(1) PWMx输出关闭时（包括PWME关闭和FLT发生）,PWMxA和PWMxB（x = 0,1,2,3）输出固定低电平（PWMSA = 0, PWMSB = 1）或高电平（PWMSA =1, PWMSB =0）。

(2) 一旦检测到FLTx引脚的高/低电平，内部状态会保持，PWMx输出会关闭。

(3) 当FLTx输入信号有效期间，FLTS位无法清除。只有当FLTx输入信号消失后，才能清除FLTS状态位。

PWM输出周期 = [PP.15，PP.0] X PWMx时钟。

当[PP.15，PP.0] = 000H，如果PWMSA = 0，不管PWMx占空比为多少，PWMxA（x = 0,1,2,3）输出低电平。

当[PP.15，PP.0] = 000H，如果PWMSA = 1，不管PWMx占空比为多少，PWMxA（x = 0,1,2,3）输出高电平。

PWMx输出占空比 = [PD.15，PD.0] X PWMx时钟

当[PP.15，PP.0] ≤ [PD.15，PD.0]，如果PWMSA = 0，PWMxA（x = 0,1,2,3）输出高电平。

当[PP.15，PP.0] ≤ [PD.15，PD.0]，如果PWMSA = 1，PWMxA（x = 0,1,2,3）输出低电平。

编程注意事项：

(1) 设置PWMx->PWMLOCK（x = 0,1,2,3，下同）寄存器内容为0x5AA5；选择PWMx模块时钟源。

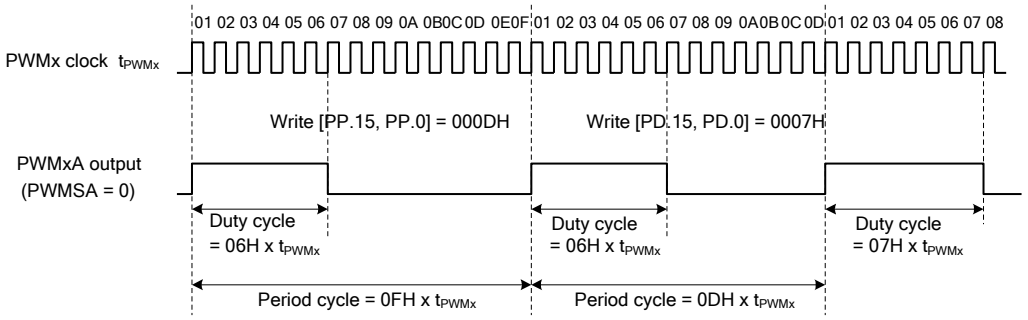
(2) 通过写适当的值到PWM周期控制寄存器（PWMx->PWMPR）或PWM占空比寄存器（PWMx->PWMDR）设置PWM周期/占空比。

(3) 通过设置PWMx控制寄存器（PWMx->CR）的PWMSA和PWMSB位选择PWMxA和PWMxB输出模式（高电平有效或低电平有效）。

(4) 通过设置PWMx控制寄存器（PWMx->CR）中的EPWMA或EPWMB位为"1"来允许PWMx端口输出。

(5) 如果PWMx周期或者占空比需要改变，操作流程如同步骤2或者步骤3说明。修改后的重载计数器的值在下一个周期开始有效。

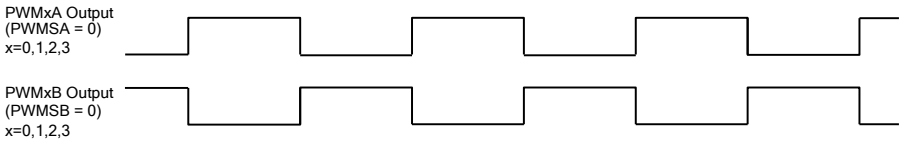
(6) 为避免干扰，设置PWMx->PWMLOCK寄存器中的数据不等于0x5AA5。



PWMx输出周期或占空比更改示例

20.5 PWMxB 互补输出（x = 0,1,2,3）

如下图所示，一般的，当没有插入死区时间时，PWMxB输出波形的周期及占空比与PWMxA相同。通过设置PWMSB，PWMxB（x = 0,1,2,3）可以设置为输出波形与PWMxA输出波形互补。当PWMx控制寄存器中EPWMxB位置1时，PWMxB的有效电平由PWMSB设置。



PWMxA和PWMxB引脚输出波形

注意：

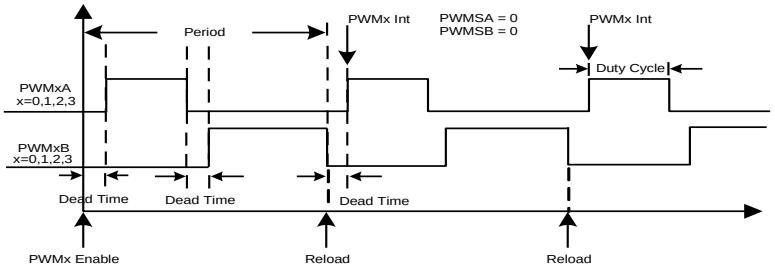
(1) 尽管PWMxA（x = 0,1,2,3）被禁止，但是如果PWMxB被允许，则它们仍然会有输出信号。

(2) 如果EFLT置位，当FLTx端口有效时，PWMxB（x = 0,1,2,3）和PWMxA（x = 0,1,2,3）都输出低（PWMSA = 0, PWMSB = 1）或者都输出高（PWMSA = 1, PWMSB = 0）。

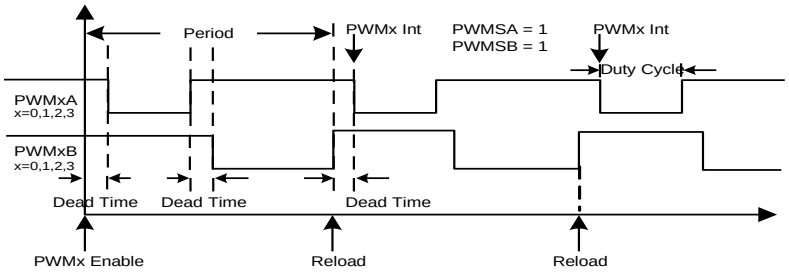
20.6 死区时间

SH32F9001_32F9801 PWMx提供死区时间控制功能。

当PWMSA = 0, PWMSB = 0时，死区时间产生如下图所示。



当PWMSA = 1, PWMSB = 1时，死区时间产生如下图所示。



通过写PWMx死区时间控制寄存器，在PWMxA（x = 0,1,2,3）和PWMxB（x = 0,1,2,3）之间产生死区时间。

PWMxA与PWMxB周期相同。

注意：

(1) 死区时间必须在PWMx（x = 0,1,2,3）输出允许前置位，否则，死区时间不会改变。所以为了修改死区时间，先解锁PWMxLOCK寄存器（PWMx->PWMLOCK = 5AA5H）并禁止PWMx输出，然后改变死区时间，允许PWMx输出。最后，为保证PWMx相关寄存器不受干扰影响而改变，修改PWMLOCK寄存器内容不等于5AA5H。

(2) 为了产生死区时间，请确保（PWMxA周期 - PWMxA占空比）> 2（PWMxB的死区时间）。否则，PWMxB当PWMSB = 1时输出高电平，当PWMSB = 0时输出低电平。*

(3) PWMDT寄存器用于控制死区时间，它的时基为PCLK0时钟，而周期和占空比的时基由TCK[2:0]控制，最小为2个PCLK0时钟。

(4) 每次重新开启PWMx前，须先将死区控制寄存器清0，然后再重新设置此寄存器。

20.7 寄存器

PWM0-3 模块寄存器列表 (基地址:0x0x4000 3C00)

地址	寄存器名	说明
0x4000 3C00+ 0x400*x	CR	PWMx控制寄存器(PWM0～3对应x:0~3)
0x4000 3C04+ 0x400*x	PWMLOCK	PWMx保护寄存器(PWM0～3对应x:0~3)
0x4000 3C08+ 0x400*x	PWMPR	PWMx周期寄存器(PWM0～3对应x:0~3)
0x4000 3C0C+ 0x400*x	PWMDR	PWMx周期寄存器(PWM0～3对应x:0~3)
0x4000 3C10+ 0x400*x	PWMDTR	PWMx死区寄存器(PWM0～3对应x:0~3)
0x4000 3C14+ 0x400*x	PWMINTF	PWMx中断标志和清除寄存器(PWM0～3对应x:0~3)

20.7.1 PWM 控制寄存器 (PWMx_CR)

偏移地址: 0x0000 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留		FLTDMA	PWMDMA	EPWMA	EPWMB	PWMIE	EFLTIE	PWMSA	PWMSB	EFLT	FLTC	TCK[2:0]			PWMEN
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 14	保留	-
13	FLTDMA	**FLT通道1DMA允许位** 0：禁止DMA请求 1：允许DMA请求
12	PWMDMA	**PWM溢出DMA允许位** 0：禁止DMA请求 1：允许DMA请求
11	EPWMA	**16位PWMxA输出使能位** 0：PWMxA输出禁止，用作I/O功能 1：PWMxA输出允许
10	EPWMB	**16位PWMxB输出使能位** 0：PWMxB输出禁止，用作I/O功能 1：PWMxB输出允许
9	PWMIE	**PWM溢出中断允许位** 0：禁止PWM溢出中断 1：允许PWM溢出中断
8	EFLTIE	**PWM引脚电平中断允许位** 0：禁止PWM引脚错误中断 1：允许PWM引脚错误中断
7	PWMSA	**PWMxA输出模式** 0：高电平驱动，PWMxA占空比期间输出高电平，占空比溢出后输出低电平 1：低电平驱动，PWMxA占空比期间输出低电平，占空比溢出后输出高电平
6	PWMSB	**PWMxB输出模式** 0：低电平驱动，PWMxB占空比期间输出低电平，占空比溢出后输出高电平 1：高电平驱动，PWMxB占空比期间输出高电平，占空比溢出后输出低电平
5	EFLT	**FLT引脚配置位** 0：普通I/O端口 1：PWM故障检测输入引脚
4	FLTC	**FLT引脚电平配置位** 0：FLT为低电平时，PWM输出关闭 1：FLT为高电平时，PWM输出关闭
3 ~ 1	TCK[2:0]	**16位PWM时钟源选择位** 000：PCLK0时钟/2 001：PCLK0时钟/4 010：PCLK0时钟/8 011：PCLK0时钟/16 100：PCLK0时钟/1 101：PCLK0时钟/32 110：PCLK0时钟/64 111：PCLK0时钟/256
0	PWMEN	**PWM模块使能位** 0：Disable 1：Enable

20.7.2 PWM 保护寄存器 (PWMx_PWMLOCK)

偏移地址: 0x0004 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PWMLO[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	PWMLO[15:0]	PWM锁定位 只有当PWMLO = 0x5AA5，才能允许修改其他PWMx寄存器

20.7.3 PWM 周期寄存器 (PWMx_PWMPR)

偏移地址: 0x0008 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PP[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	PP[15:0]	16位PWM周期寄存器

20.7.4 PWM 占空比寄存器 (PWMx_PWMDR)

偏移地址: 0x000C + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
PD[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	PD[15:0]	16位PWM占空比寄存器

20.7.5 PWM 死区寄存器 (PWMx_PWMDTR)

偏移地址: 0x0010 + 0x400*x
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
DT[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**DT[15:0]**	**16位PWM死区时间控制** 死区时间为DT[15:0] × tosc

20.7.6 PWM 中断标志和清除寄存器 (PWMx_PWMINTF)

偏移地址: 0x0014 + 0x400*x
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留												PWMIFC	FLTSC	PWMIF	FLTS
-												*WO*	*WO*	*RO*	*RO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 4	保留	-
3	**PWMIFC**	**PWM中断标志位清除位** 0：无效 1：清除
2	**FLTSC**	**FLT状态位清除位** 0：无效 1：清除
1	**PWMIF**	**PWM中断标志位** 0：PWM周期计数器溢出，需由软件清0 1：PWM周期计数器溢出，硬件置1
0	**FLTS**	**FLT状态位** 0：PWM正常状态，需由软件清0 1：PWM异常，PWM输出关闭，硬件置1

21. 可编程计数器阵列（PCAx（x=0~7））

21.1 简介

SH32F9001_32F9801内置有8个高级的16位可编程计数器阵列（Programable Counter Array），其中PCA0/1计数器最大能以AHB（HCLK）运行，PCA2~PCA7计数器最大能以APB0时钟（PCLK0）运行。可编程计数器阵列PCAx提供增强的定时器功能，它仅需要较少的CPU干预。

21.2 主要特性

- SH32F9001_32F9801有8个16位定时器PCA
- PCAx（x=0~7）具有两路独立比较捕捉模块；
- 支持输入捕捉、比较匹配(输出)
- 支持频率可调的方波输出
- 支持PWM调制输出，可选8位/16位PWM调制器
- 16位PWM调制器支持相位修正功能和相频修正功能
- PCA0/1、PCA2/3、PCA4/5、PCA6/7两两能够级联拓展成32位PCA模块

说明：①小下标x为PCA序号，例如PCAx（x=0~7），下文将统一使用PCAx，不再说明x的值。

②小下标n为比较捕捉模块序号，例如P0CEXn（n=0,1），下文将统一使用PxCEXn，不再说明n的值。

PCAx由一个专用的16位计数器/定时器和2个16位捕捉/比较模块组成，PCAx的原理框图如下图所示，每个捕捉/比较模块有其自己的I/O线PxCEXn。

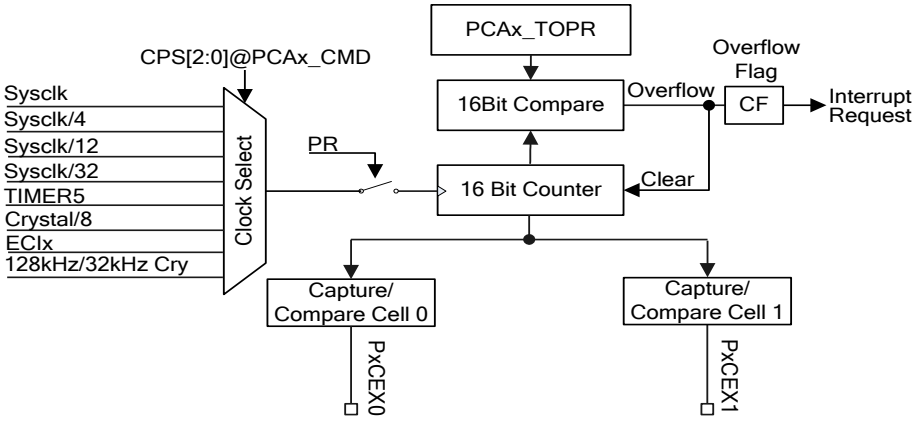


图21-1 PCAx原理框图

21.3 时钟配置

PCAx的计数器/定时器有一个可编程选择时钟源：总线时钟、总线时钟/4、总线时钟/12、总线时钟/32、外部振荡器时钟/8、定时器5溢出或或ECIx输入引脚上的外部时钟信号、内建128kHz RC/外部32kHz晶振，通过PCAx_CMD寄存器中的CPS[2:0]位选择定时器/计数器的时钟源，如下表所示。

CPS2	CPS1	CPS0	时钟源
0	0	0	总线时钟
0	0	1	总线时钟的4分频
0	1	0	总线时钟的12分频
0	1	1	总线时钟的32分频
1	0	0	定时器5溢出
1	0	1	外部高频振荡器的8分频
1	1	0	ECIx下降沿（最大速率=总线时钟频率/4）
1	1	1	内建128kHzRC/外部32kHz晶振

注意：1、外部高频振荡器8分频与总线时钟同步。

2、总线时钟周期不得低于计数时钟周期的4倍频（计数时钟为总线时钟时除外），否则PCAx Counter将不能正确计数；

3、若CPS[2:0]为111时，LSE(32kHz晶振)和LSI(内建128kHz RC)同时存在，则PCAx的时钟源为LSE(32kHz晶振)

4、当LSERDY为‘1’时，外部32.768kHz晶振作为计数源时钟；当LSERDY为‘0’时：若CSMLSEF为‘1’，则CSM切换时钟（内建32kHz RC）作为计数源时钟，若CSMLSEF为‘0’，则内建128kHz作为计数源时钟。

5、当定时器5溢出作为PCA2-7时钟源时，定时器5的时钟源只能选择内部APB0时钟PCLK0，然而，对于PCA0-1来说，定时器5可以选择PCLK0、定时器5输入引脚（T5）的下降沿、32kHz Cry（LSE）、128kHz WDT RC。

21.4 工作原理

16位的PCAx计数器/定时器内部集成16位的计数单元，16位的计数溢出值寄存器由TOPH和TOPL组成，用户可以自由地配置TOP寄存器定义计数器的溢出值，TOP寄存器上电初始值为0xFFFF。

16位定时器/计数器是PCAx最基本，也是各个模块正常运转必不可少的单元。通过PCAx_CR寄存器的PR位可以开启/禁止定时器/计数器工作，当PR设置为逻辑‘0’时，定时器/计数器16位Counter也被强制清‘0’。当计时器从0x0000向TOP计数溢出（TOP到0x0000的同一个总线时钟里，即锯齿波模式）或计数器从TOP递减计数为0x0000（PCAx Counter工作在三角波模式）时，PCAx_PCAINTF寄存器中的溢出标志（CF）被置为逻辑’1’并产生一个中断请求（PCAx_CMD中ECF位设置为逻辑‘1’即可允许CF标志产生中断请求）。当CPU转向中断服务程序时，CF位不能被硬件自动清除，必须用软件清除（PCAx_PCAINTF寄存器中CFC位置1）。这里需要说明的是，当PCAx两个模块工作在不同工作模式（但计数波形模式相同），CF有可能在PCA Counter到达TOP和0x0000时都会被置为逻辑“1”。

PCAx_TOPR、CPRn为16位寄存器，可以直接进行16位读写，但仍然可以以8位的方式操作它们，在本文中标注为TOPH、TOPL、CPHn、CPLn。

读PCAx_TOPR、CPRn对PCAx计数无影响。

时序图21-2为为PCAx计数单元时序波形图，图中短横杠表示PCAx_TOPR值。图21-2-a中计数器从0X0000开始计数到PCAx_TOPR溢出称为一个计数周期，溢出周期的结束发生在PCAx_TOPR向0x0000清零时刻。图21-2-b中计数器从PCAx_TOPR递减计数到0X0000溢出称为一个计数周期。此时PCAx中断标志CF信号会被拉高，若中断允许，系统将转向相应PCAx中断服务子程序，但PCAx的计数不受中断影响而停止计数。

注：从生成波形的形状上，向上计数模式也可称作锯齿波模式，中央对齐计数模式也可称作三角波模式。

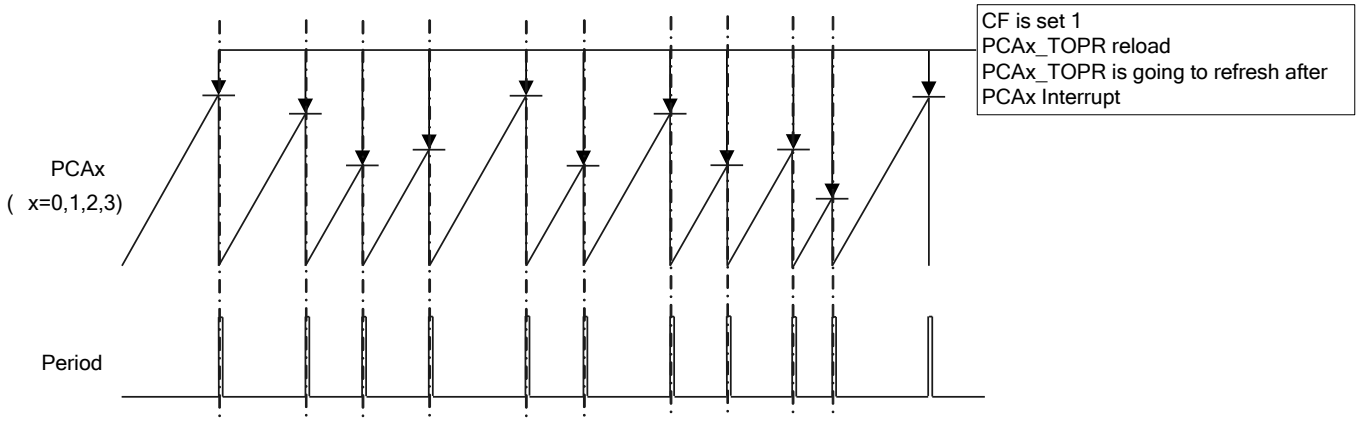


图21-2-a PCAx 计数器向上计数（锯齿波）模式下的波形图

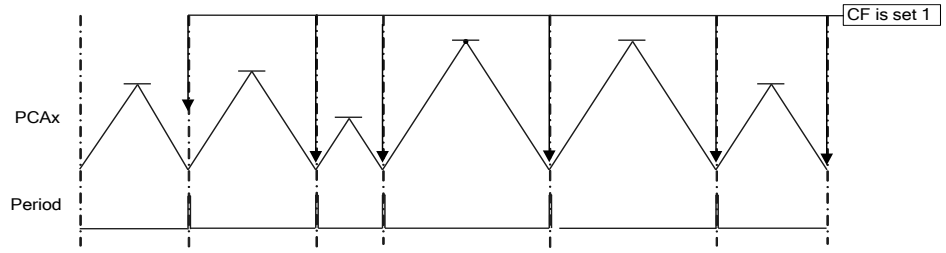


图21-2-b PCAx 计数器在中央对齐计数（三角波）模式下的波形图

21.4.1 PCAx 模式配置

PCAx计数器挂载2路捕捉/比较模块均可以实现增强功能。每个捕捉/比较模块都可被配置为独立工作（注：1个捕捉/比较模块对应1路通道），每个模块在系统控制器中都有属于自己的控制寄存器，这些寄存器用于配置模块的工作方式和与模块交换数据。可以通过配置各自模块PCAx_CPMn（n = 0或1）寄存器中SMn[1]和SMn[0]两位使能该模块工作在以下4种工作模式之一：边沿触发捕捉、比较匹配输出、频率输出、PWM输出模式。

工作方式选择如下表Table21-1所示：

表21-1 PCAx 模式选择表

模式	SDEN @PCAx_CMD	SMn[1] @PCAx_CPMn	SMn[0] @PCAx_CPMn	FSn[1] @PCAx_CPMn	FSn[0] @PCAx_CPMn	功能说明
Mode0	0	0	0	0	X	正沿触发捕捉（锯齿波）
				1	0	负沿触发捕捉（锯齿波）
				1	1	任意沿触发捕捉（锯齿波）
Mode1	0	0	1	0	X	连续比较匹配（输出）（锯齿波）
				1	X	单次比较匹配（输出）（锯齿波）
Mode2	0	1	0	X	X	频率输出（锯齿波）
Mode3	0	1	1	0	0	8位PWM（锯齿波）
				0	1	16位PWM（锯齿波）
	1			1	0	16位相位修正PWM（三角波）
	1			1	1	16位相频修正PWM（三角波）
其他						PCAx Counter正确计数，但比较/捕捉模块不工作。

注：X表示任意值;

当PCAx被设定为两种波形（锯齿波或三角波）中的一种时，比较捕捉模块的另一种波形模式即使被配置也是无效的。(注：即如果配置为锯齿波模式，则三角波相关的寄存器配置无效)

使用比较/捕捉模块时需注意：

①改变PCAx_TOPR值必须保证新的PCAx_TOPR值不小于所有比较寄存器PCAx_CPRn的数值。

②比较/捕捉模块作为PWM输出功能时，若PCAx_CPRn等于0x0000，输出一直保持为低电平；若PCAx_CPRn等于PCAx_TOPR，输出则保持为高电平。引脚取反则输出正好相反；

③PCAx的所有比较/捕捉模块只能工作于同一模式（例：PCA0的比较/捕捉模块0和比较/捕捉模块1只能工作在同一模式）。

PCAx_CPMn寄存器用于配置PCAx捕捉/比较模块的工作方式，Table21-1概述了模块工作在不同方式时该寄存器相关位的设置情况。PCAx_CPMn寄存器中的ECCFn位置‘1’，将允许模块的CCFn中断。注意：要使单个的CCFn中断得到响应，必须先整体允许PCAx中断。PCAx中断配置的详细信息见图21-3。

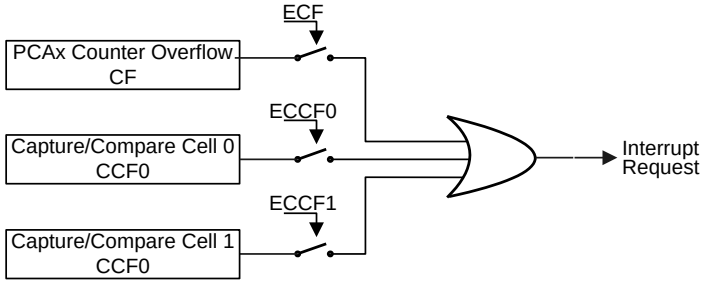


图21-3 PCAx 中断原理框图

21.4.2 Mode0：边沿触发的捕捉模式

在该方式，PxCEXn引脚上出现的电平跳变将捕捉PCAx计数器/定时器的当前计数值并将其装入到对应模块的16位捕捉/比较寄存器（PCAx_CPRn）中（通过配置SMn[1:0] =00使能该方式）。在此模式下，PCAx_CPMn寄存器中的FSn[1]和FSn[0]位用于选择触发捕捉的电平变化类型：低电平到高电平（正沿触发FSn[1:0] =0X）、高电平到低电平（负沿触发FSn[1:0]=10）或任何变化（正沿或负沿触发FSn[1:0]=11）。当捕捉发生时，PCAINTF中的捕捉/比较标志（CCFn）被置为逻辑‘1’并产生一个中断请求（如果CCFn中断被允许）。当CPU转向中断服务程序时，CCFn位不能被硬件自动清除，必须用软件清0（PCAx_PCAINTF寄存器中CCFnC位置1）。如果FSn[1]和FSn[0]位都被设置为逻辑‘1’，可以通过读PCAx_CPMn寄存器的TCPn位来确定本次捕捉是由上升沿触发还是由下降沿触发。

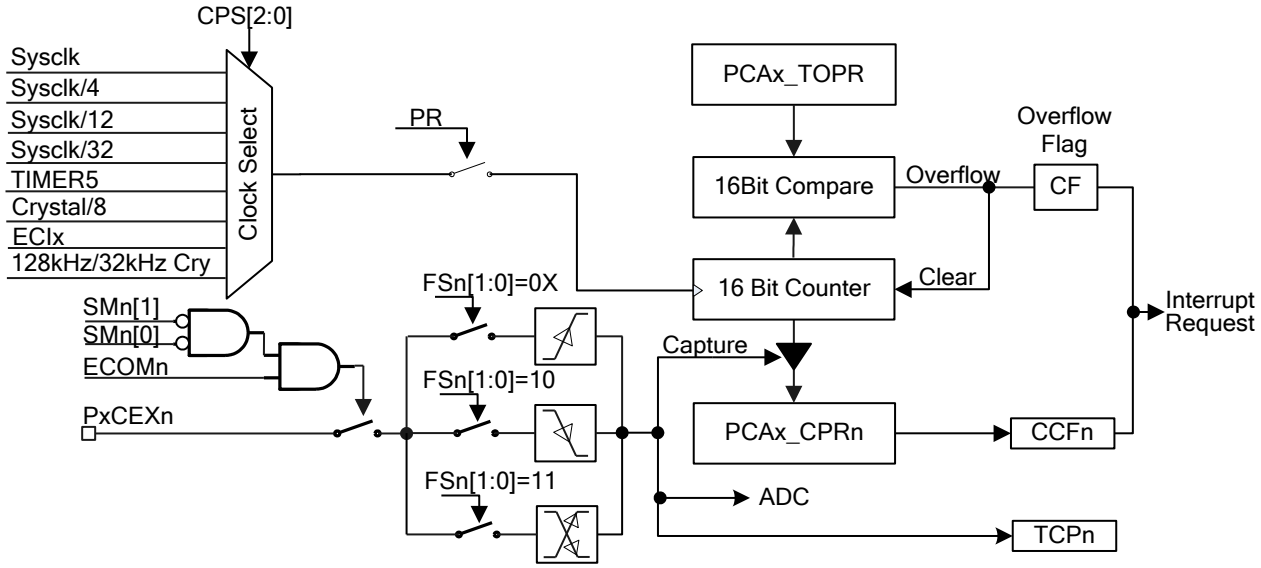


图21-4 PCAx 捕捉方式原理框图

21.4.3 Mode1：比较匹配输出（软件定时器）方式

软件定时器方式也称为比较输出方式（通过配置SMn[1:0]=01使能该方式）。在该方式，FSn[1:0] = 0x，可以实现连续匹配，计数器会从0到PCAx_TOPR值重复计数。PCAx将计数器/定时器的计数值与模块的16位捕捉/比较寄存器（PCAx_CPRn）进行比较。当发生匹配时，PCAx_PCAINTF中的捕捉/比较标志（CCFn）被置为逻辑‘1’（当MATn = 1时）并产生一个中断请求（如果CCFn中断被允许），并且模块的PxCEXn引脚上的逻辑电平将发生变化（设置TCPn位使能该功能）。当CPU转向中断服务程序时，CCFn位不能被硬件自动清除，必须用软件清0（PCAx_PCAINTF寄存器中CCFnC位置1）。

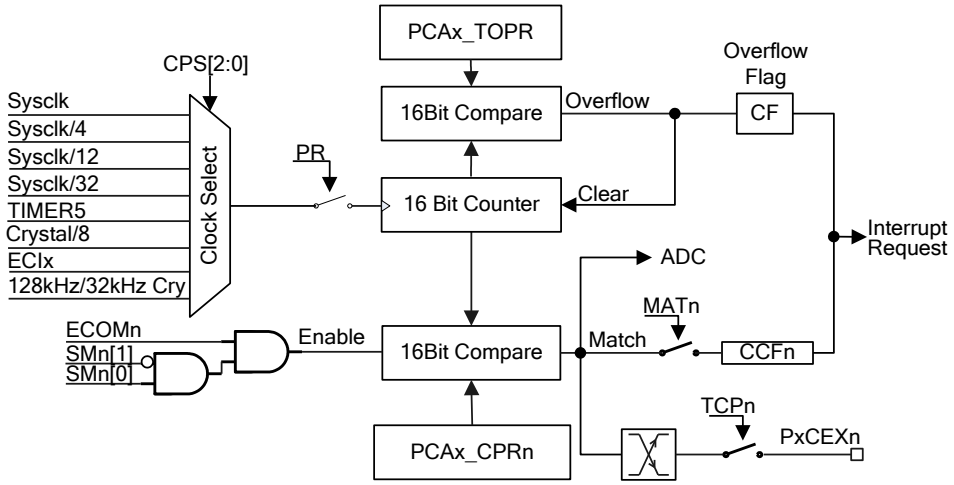


图21-5 比较匹配方式（软件定时器）原理框图

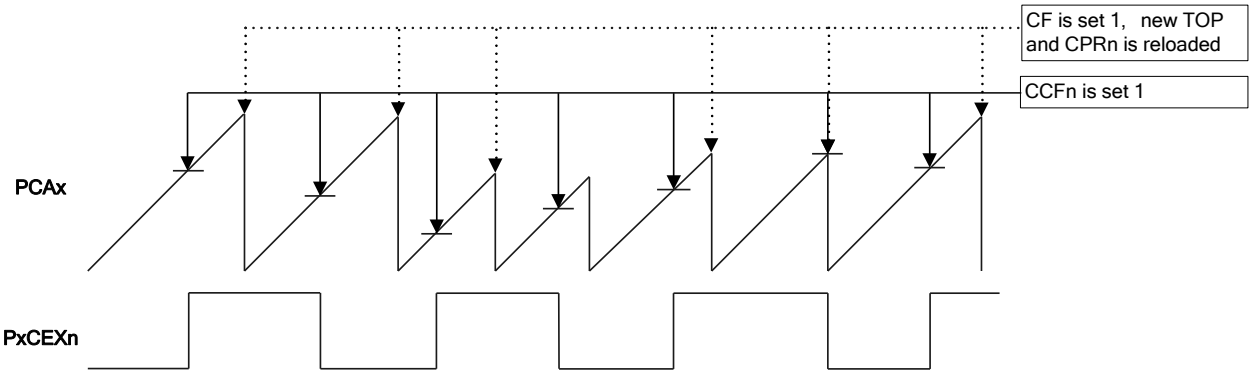


图21-6 比较匹配方式（软件定时器）波形图

当FSn[1:0] =1x时，模块将工作于单次匹配模式，可以通过设置PCAx_FORCE寄存器OSCn位来实现产生单次上升沿或者下降沿。即计数器从0计数到PCAx_TOPR值后溢出重新计回0且不断重复计数，但是输出引脚PxCEXn的电平也只在第一次匹配时发生翻转，之后就保持翻转后的电平不再改变。相应的标志位CCFn会被置1，如果ECCFn为1，则会触发一次中断。

另外，当模块工作在比较匹配输出（软件定时器）方式时，可通过设置PCAx_FORCE寄存器FCOn位强制产生一次比较匹配，但是，这样的匹配并不是真正的匹配，他不会置起比较匹配标志，也不会影响寄存器数值，只是会在PxCEXn引脚产生电平翻转输出（单次触发模式时发生强制匹配会输出相应电平）。当产生一次强制匹配（写FCOn为'1'）生效后，该位会被硬件自动清'0'。

21.4.4 Mode2：频率输出方式

频率输出方式可在模块的PxCEXn引脚产生可编程频率的方波(配置SMn[1:0]=10使能该模式，在此模式下，PCAx_CPRn寄存器的更新不使用双缓冲机制)。捕捉/比较模块的高字节CPHn位@PCAx_CPRn保持输出电平改变前要计的PCA时钟数。所产生的方波的频率$F_{PxCEXn}= F_{PCAx} /(2 \times CPHn)$

注：对于该方程，CPHn中的值为0x00时，相当于256。

其中：FPCAx是由PCAx方式寄存器（PCAx_CMD）中的CPS[2:0]位选择的PCAx时钟的频率。

如下图21-7，捕捉/比较模块的低字节CPLn位@PCAx_CPRn与PCAx计数器的低8位字节进行比较；若两者匹配时，PxCEXn引脚的电平发生改变，同时高字节CPHn中的偏移值被加到CPLn，PLx继续计数直到再次匹配，PxCEXn引脚的电平改变，周而复始，PxCEXn引脚输出频率由CPHn控制。如果PCAx的某个比较/捕捉模块使能该模式，TOPL的值固定为0XFF，用户可以配置TOPH值来改变计数最大值。

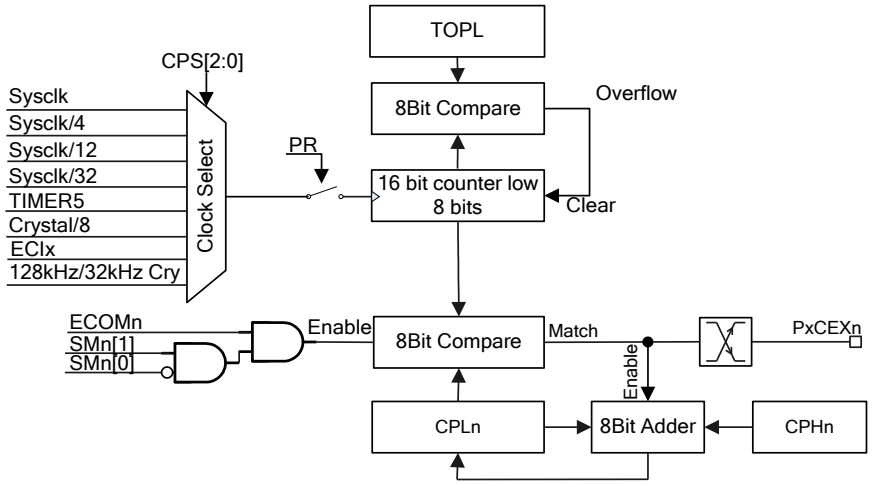


图21-7 频率输出方式原理框图

21.4.5 Mode3：PWM 模式

PCAx的每个模块都可以被独立地产生脉宽调制（PWM）输出。配置SMn[1:0]=11将使能比较/捕捉模块n工作在PWM模式。在该模式下，通过配置FSn[1]和FSn[0]两位可以使比较/捕捉模块n工作在一下4种PWM功能。

PxFSn[1]	PxFSn[0]	功能说明
0	0	8位PWM（锯齿波）
0	1	16位PWM（锯齿波）
1	0	16位相位修正PWM（三角波）
1	1	16位相频修正PWM（三角波）

8 位脉宽调制器（PWM8）功能

当比较/捕捉模块工作在8位PWM功能时，PCAx 低8位 Counter从0x00向TOPL递增计数（锯齿波模式），当PLx溢出时（从0xFF到0x00），保存在CPHn中的值被自动装入到CPLn，这个过程不需软件干预（如下图21-8 8位脉宽调制器（PWM）方式原理框图）。如果PCAx的某个比较/捕捉模块使能该模式，TOPL的值固定为0XFF，用户可以配置TOPH值来改变计数最大值，但不影响8位PWM输出周期。

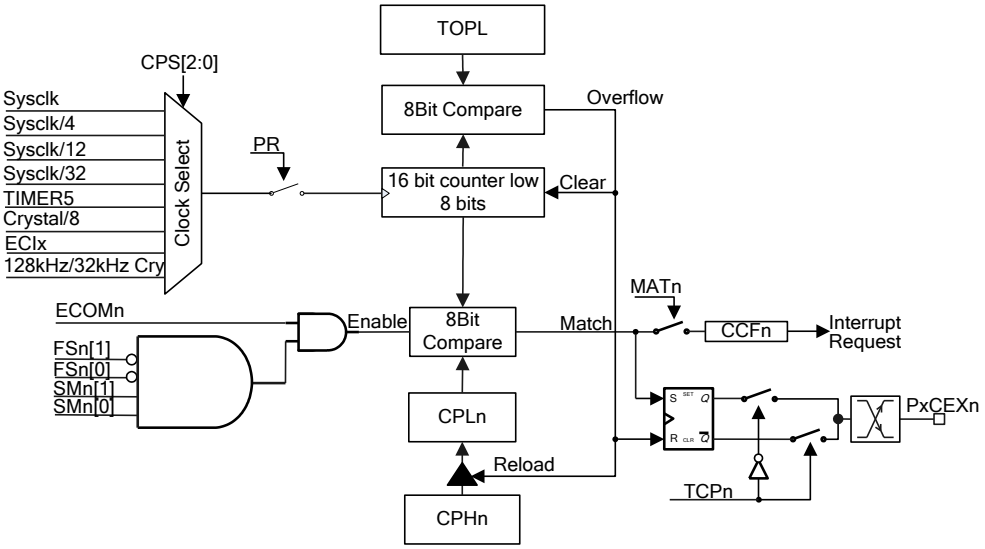
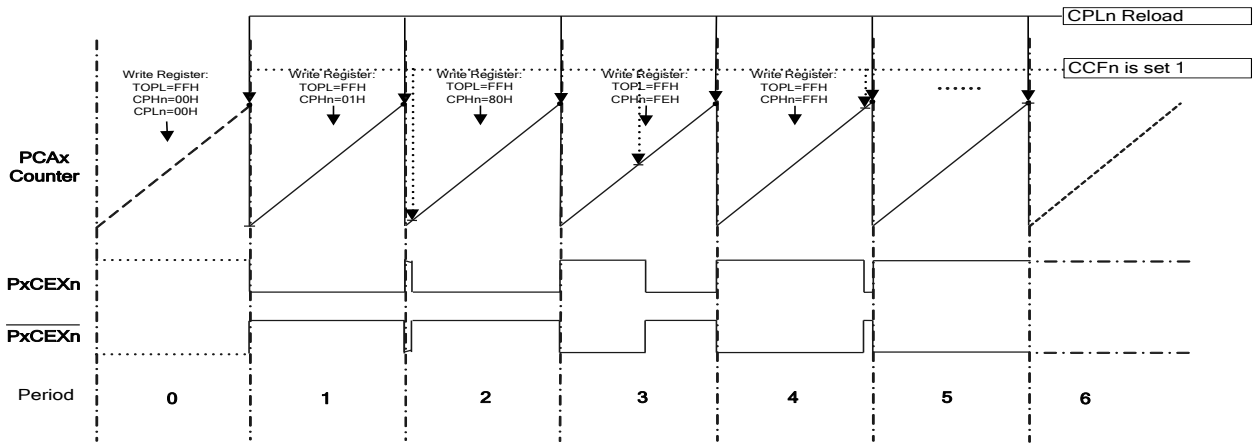


图21-8 8 位脉宽调制器（PWM）方式原理框图

模块的捕捉/比较寄存器CPHn用来改变PWM 输出信号的占空比（在此模式下，CPRn寄存器的更新不使用双缓冲机制）。当TCPn =0时，PCA计数器/定时器的低8位字节与CPLn中的值相等时，PxCEXn引脚上的输出被清‘0’；当计数值低8位字节溢出时，PxCEXn输出被置‘1’（见波形图21-9）；当TCPn =1时，PxCEXn引脚输出极性相反的波形。8位PWM方式的占空比**Duty=（256-（CPHn+1））/256**。

如下波形图21-9，周期1为上电PxCEXn引脚的default值；周期1为CPLn = 00H时，PxCEXn引脚输出电平；周期2~4分别为CPLn = 01H、80H、FEH时，PxCEXn引脚相应占空比的波形；周期5为CPLn = FFH引脚PxCEXn被拉高。



图中短横杠表示CPLn值，圆点表示TOPL值

图21-9 8位PWM波形图

16位脉宽调制器（PWM16）功能

16位脉宽调制PWM同8位PWM模式类似，都是基于PCAx Counter的单沿计数模式。在该方式下，16位捕捉/比较模块CPRn用来定义PWM信号低电平时间的PCAx时钟数。当TCPn =0时，PCAx计数器与模块的匹配寄存器CPRn值匹配时，PxCEXn的输出被置为清‘0’；当计数器溢出时，PxCEXn输出被置‘1’，当TCPn =1时，PxCEXn引脚输出极性相反的波形。实现原理框图如下图21-10所示。

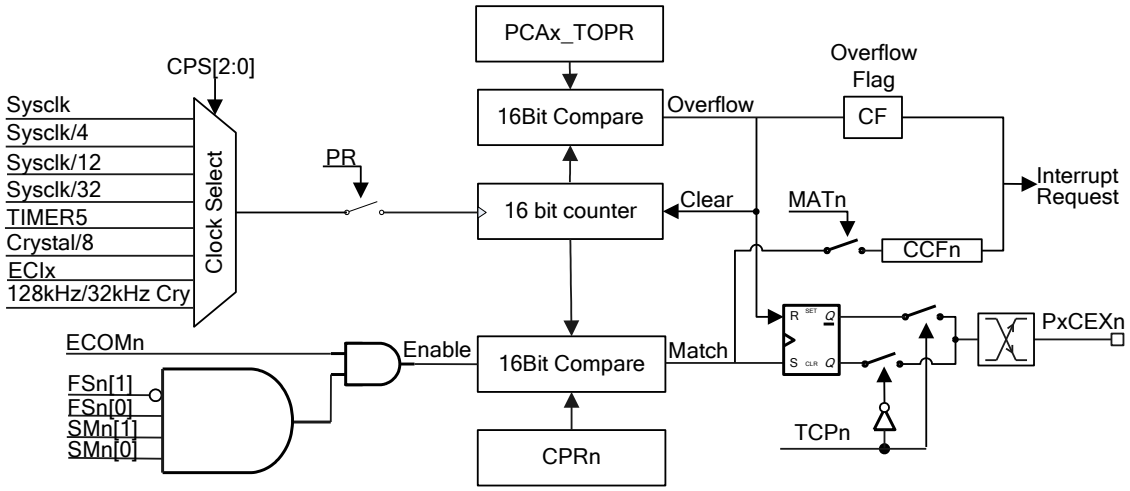


图21-10 16位PWM模式框图（单沿模式）

16位PWM方式的占空比Duty= (TOPR - CPRn) / (TOPR + 1)

输出波形图如下图21-11所示，周期1为上电PxCEXn引脚的default值；周期1为CPRn = 0000H时，PxCEXn引脚输出电平；周期2~4分别为CPRn = 0001H、8000H、FFFEH时，PxCEXn引脚相应占空比的波形；周期5为CPRn = FFFFH引脚PxCEXn被拉高；周期6~8为改变PCAx_TOPR和CPRn值时，PxCEXn引脚的输出波形。

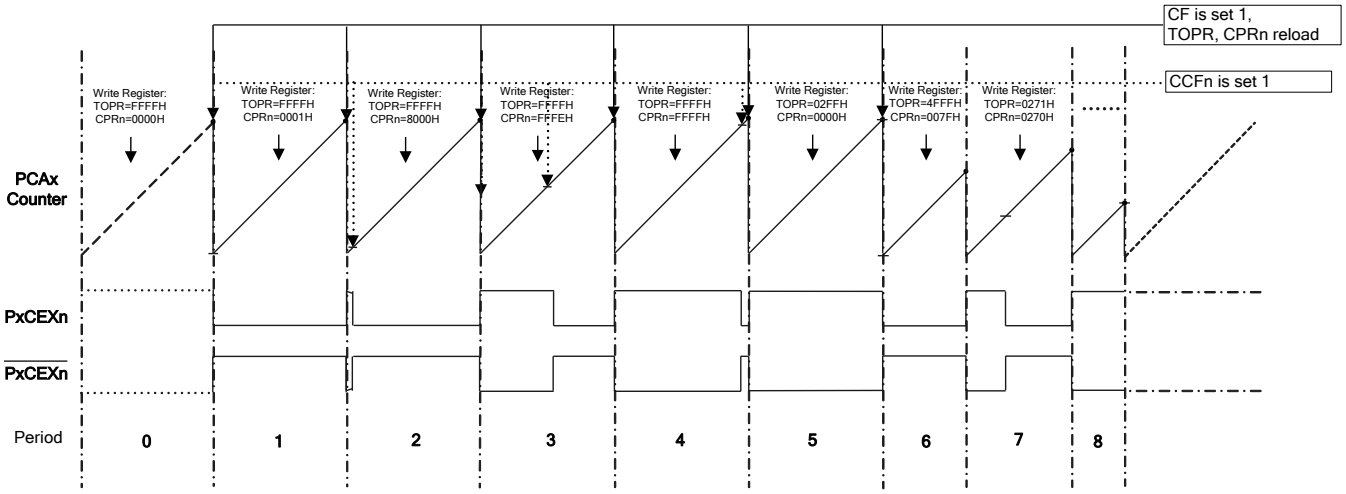


图21-11 16位PWM波形图

16位相位修正脉宽调制器（XPWM16）功能

相位修正PWM(XPWM)功能为用户提供了一个获得高精度的、相位准确的PWM波形的方法。与相位和频率修正模式类似，此模式是基于三角波操作。即计时器重复地从0x0000计到PCAx_TOPR，然后又从PCAx_TOPR倒退回到0x0000。当TCPn=0时，当计时器往PCAx_TOPR计数时若PCAx Counter与CPRn匹配，PxCEXn将清零为低电平；而在计时器往0x0000计数时若PCAx Counter与CPRn匹配，PxCEXn将置位为高电平。当TCPn =1时，PxCEXn引脚输出极性相反的波形。实现原理框图如图21-12所示。当匹配发生时，匹配标志位CCFn被置'1'（MATn=1），若中断允许，则会产生一次PCAx中断响应。当PCAx从TOP递减计数回到0x0000溢出时，PCAx中断标志CF也会被置'1'，若中断允许，则会产生一次PCAx中断响应。

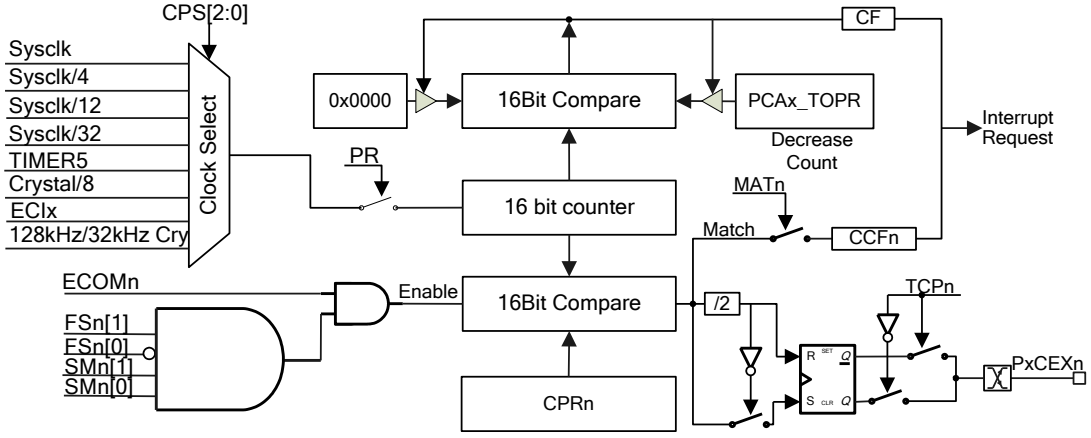


图21-12 16位相位修正模式框图

与锯齿波操作相比，三角波操作可获得的最大频率要小。但其对称特性十分适合于电机控制。

相位修正PWM模式的PWM分辨率可由PCAx_TOPR定义。最小分辨率为 2 比特(PCAx_TOPR设为 0x0003)，最大分辨率为16位。PWM 分辨率位数R_{PxXPWM}=log（PCAx_TOPR+1）/log(2)

工作于相位修正PWM模式时，计数器的数值一直累加到PCAx_TOPR值，然后改变计数方向。在一个定时器时钟周期里PCAx值等于PCAx_TOPR值，然后在下一次计数到来时PCAx_TOPR和CPRn将得到更新。如下图21-13所示为16位XPWM波形图，由图中可以看出，PWM周期起始于TOP点，也终止于TOP点，并且PCAx_TOPR和CPRn寄存器的值在PCAx到达TOP值时得到更新。由于为PCAx_TOPR和CPRn寄存器写操作使用的是双缓冲机制，使得在PCAx计数过程中改变这两个寄存器的值将不会影响当前周期的计数最大值和比较捕获值。

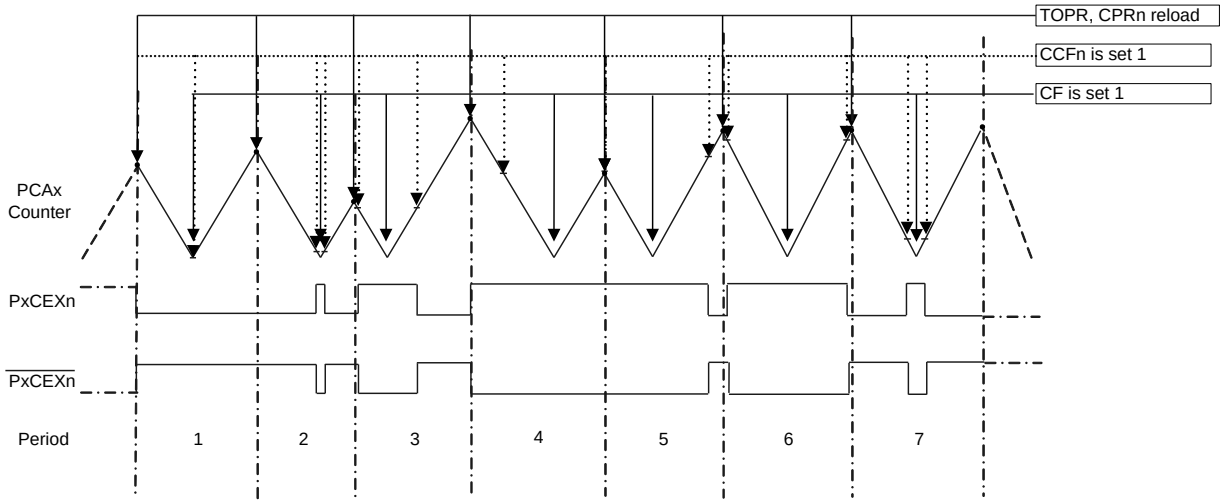


图21-13 16位XPWM波形图

若要在PCAx运行时改变TOP值，最好用相位与频率修正模式代替相位修正模式。若PCAx_TOPR保持不变，那么这两种工作模式实际上没有区别。

工作于相位修正模式时PWM频率$F_{XPWM} = F_{PCAx} / (2 * PCAx_TOPR)$

16 位相频修正脉宽调制器（XPPWM16）方式

相位与频率修正PWM(XPPWM)模式以下简称相频修正PWM模式。与相位修正模式类似，该功能也是基于三角波操作。XPPWM可以产生高精度的、相位与频率都准确的PWM波形。其实现原理框图如图21-14所示。计时器重复地从0x0000计到PCAx_TOPR，然后又从PCAx_TOPR倒计数到0X0000。TCPn=0时，当计时器往PCAx_TOPR计数时若PCAx Counter与CPRn匹配，PxCEXn将清零为低电平；而在计时器往0x0000计数时若PCAx Counter与CPRn匹配，PxCEXn将置位为高电平。当TCPn=1时，PxCEXn引脚输出极性相反的波形。当匹配发生时，匹配标志位CCFn被置'1'（MATn=1），若中断允许，则会产生一次PCAx中断响应。当PCAx Counter从PCAx_TOPR递减计数回到0x0000溢出时，PCAx中断标志CF也会被置'1'，若中断允许，则会产生一次PCAx中断响应。

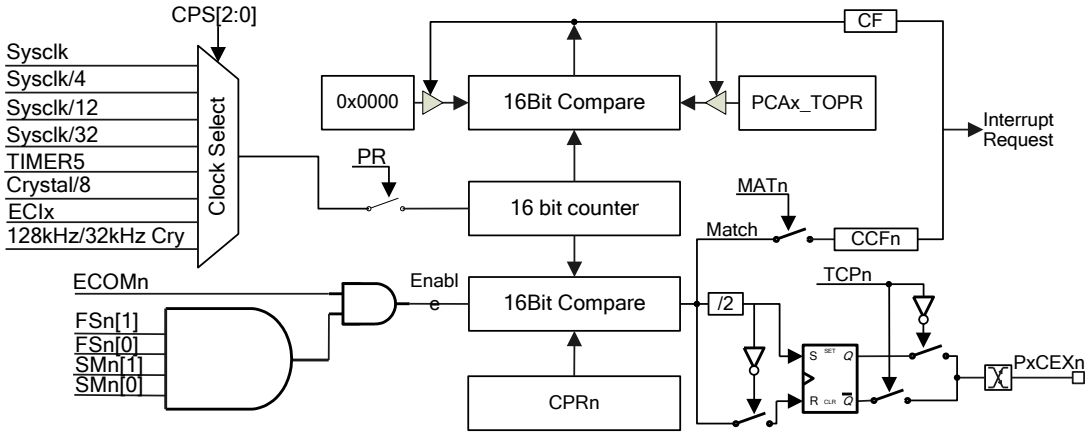


图21-14 16位相频修正模式框图

与锯齿波操作相比，三角波操作可获得的最大频率要小。但其对称特性十分适合于电机控制。

相频修正PWM模式与相位修正PWM模式的主要区别在于CPRn和PCAx_TOPR寄存器的更新时间，详见图21-13与图21-15。相频修正PWM输出的PWM在所有的周期中均为对称的信号，这是由于相位修正PWM是在PCAx_TOPR点更新CPRn和PCAx_TOPR寄存器值，而相频修正PWM则是在0x0000点更新CPRn和PCAx_TOPR寄存器的值。因此相频修正PWM一个周期内的上升和下降斜坡的长度始终相等，因此输出的脉冲是绝对对称的，这样就确保了频率的正确性。在CPRn寄存器通过双缓冲方式得到更新的同一个时钟周期里PCAx溢出标志CF置位。该中断标志位可用来在每次计数器达到0x0000时产生中断。

具体波形图如图21-15所示。如果XPPWM基频不断变化 (通过改变PCAx_TOPR值)，CPRn的双缓冲特性使其更适合于这个应用。

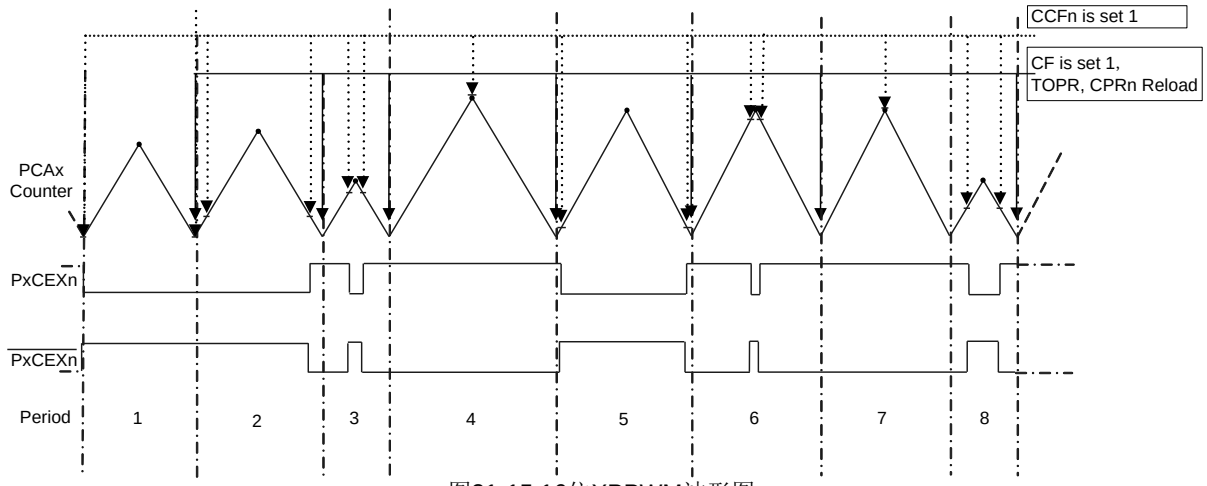


图21-15 16位XPPWM波形图

相频修正修正PWM模式的PWM分辨率由PCAx_TOPR定义。最小分辨率为2比特(PCAx_TOPR设为 0x0003)，最大分辨率为16位 (PCAx_TOPR设为0xFF)。

输出的PWM频率 $F_{XPPWM} = F_{PCAx} / (2 * PCAx_TOPR)$

输出占空比 Duty = (PCAx_TOPR - CPRn) / PCAx_TOPR

21.4.6 Mode4：PCA 级联运行

级联运行是将2个16位PCA连接为一个32位PCA的功能，从而扩展PCA的TOP值及捕捉模块比较值CPn可设置范围。

通过设置PCAx_CR->CASCEN（x=0,2,4,5）为'1'开启PCA两两级联功能。当PCA两两级联后，PCA相关配置均由PCA0/2/4/6寄存器设置，而PCA1/3/5/7的相关控制寄存器设置无效。32位PCA模式同样具有Mode0~Mode3工作方式。

级联的两种组合如下表：

级联组合	高16位	低16位
PCA0和PCA1	PCA1	PCA0
PCA2和PCA3	PCA3	PCA2
PCA4和PCA5	PCA5	PCA4
PCA6和PCA7	PCA7	PCA6

PCA级联运行的例子：

设定PCA0与PCA1级联，PCA1的定时器的时钟源为PCA0的溢出，设定PCA0的时钟源为ECI0的下降沿，计数如下图：

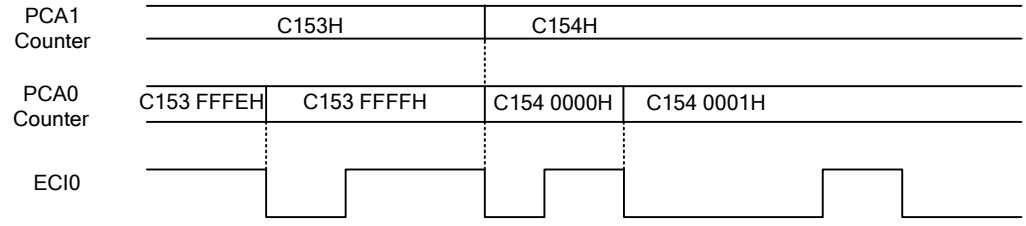


图21-16 级联运行示例图

注：级联之后，PCA1的TOPR值和CPR0/CPR1被自动映像到PCA0对应寄存器的高16位，所以读取只需要读取PCA0的32位寄存器，其余PCA级联同理。

21.5 寄存器

PCA0-1 模块寄存器列表 (基地址: 0x4004 0000)

地址	寄存器名	说明
0x4004 0000 + 0x400*x	CR	PCAx控制寄存器 (PCA0～1对应x:0~1)
0x4004 0004 + 0x400*x	CMD	PCAx方式配置寄存器 (PCA0～1对应x:0~1)
0x4004 0008 + 0x400*x	CPM0	CPM0：PCAx 捕捉/比较寄存器 (PCA0～1对应x:0~1)
0x4004 000C + 0x400*x	CPM1	CPM1：PCAx 捕捉/比较寄存器 (PCA0～1对应x:0~1)
0x4004 0010 + 0x400*x	FORCE	PCAx FORCE强制输出控制寄存器 (PCA0～1对应x:0~1)
0x4004 0014 + 0x400*x	TOPR	PCAx计数最大值 (PCA0～1对应x:0~1)
0x4004 0018 + 0x400*x	CPR0	PCAx通道0比较/捕捉模块寄存器 (PCA0～1对应x:0~1)
0x4004 001C + 0x400*x	CPR1	PCAx通道1比较/捕捉模块寄存器 (PCA0～1对应x:0~1)
0x4004 0020 + 0x400*x	PCAINTF	PCAx中断标志和清除寄存器 (PCA0～1对应x:0~1)

21.5.1 PCA 控制寄存器 (PCAx_CR)

偏移地址: 0x0000 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留														CASCEN	PR
-														*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 2	保留	-
1	**CASCEN**	**PCAx级联(Cascade)使能位(x=0，只在级联的低位PCA0有效)** 0：禁止PCA0/1级联功能 1：使能PCA0/1级联功能 注：只有在级联的两个PCA都停止运行（PR=0）时才能修改此位
0	**PR**	**PCA计数器/定时器运行控制位** 0：禁止PCA计数器/定时器 1：允许PCA计数器/定时器

21.5.2 PCA 方式配置寄存器 (PCAx_CMD)

偏移地址: 0x0004 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								CFDMA	CF1DMA	CF0DMA	ECF	SDEN	CPS[2:0]		
-								*RW*	*RW*	*RW*	*RW*	*RW*	*RW*		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7	**CFDMA**	**PCA计数器/定时器DMA允许位** 0：禁止DMA请求 1：允许DMA请求
6	**CF1DMA**	**PCA捕捉通道1DMA允许位** 0：禁止DMA请求 1：允许DMA请求
5	**CF0DMA**	**PCA捕捉通道0DMA允许位** 0：禁止DMA请求 1：允许DMA请求
4	**ECF**	**PCA计数器/定时器溢出中断允许位** 0：禁止CF中断 1：当CF被置位时，允许 PCA计数器/定时器溢出的中断请求 该位是PCA计数器/定时器溢出（CF）中断的屏蔽位
3	**SDEN**	**PCA计数模式选择位** 0：PCA工作在向上计数（锯齿波）模式, 此时该PCA其它比较捕捉模块的三角波模式均被禁止 1：PCA工作在中央对齐计数（三角波）模式，此时该PCA其它比较捕捉模块的锯齿波模式均被禁止 当PCA设定为两种计数模式中的一种时，比较捕捉模块的另一种计数模式即使被配置也是无效的。
2 ~ 0	**CPS[2:0]**	**PCA计数器/定时器时钟选择** 这些位选择 PCA计数器的时钟源。具体请参照21.3时钟配置。

21.5.3 CPM0：PCA 捕捉/比较寄存器 (PCAx_CPMn)(n=0..1)

偏移地址: 0x0008 + 0x400*x

:0x000C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								SMn[1:0]		FSn[1:0]		ECOMn	TCPn	MATn	ECCFn
-								*RW*		*RW*		*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7 ~ 6	SMn[1:0]	**PCA模式选择位** 00： 捕捉方式 01：比较匹配输出（软件定时器） 10：频率输出 11：PWM输出 具体请参照表XXXX
5 ~ 4	FSn[1:0]	**当SMn[1:0] =00时：捕捉方式选择位** 0X：比较/捕捉模块0工作在正沿触发方式 10：比较/捕捉模块0工作在负沿触发方式 11：比较/捕捉模块0工作在任意沿触发方式 当SMn[1:0] =01时：软件定时器选择位 0X：连续软件定时模式 1X：单次软件定时模式 当SMn[1:0] =11时：PWM方式选择。 00：选择 8位 PWM方式 01：选择 16位 PWM方式 10：选择16位相位修正PWM方式 11：选择16位相频修正PWM方式 当SMn[1:0] =10（频率输出方式）时此位才无效。
3	ECOMn	**比较/捕捉模块功能使能位** 0：禁止比较/捕捉模块0 1：使能比较/捕捉模块0
2	TCPn	当SMn[1:0] =00时，该位为捕捉信号边沿指示位 0：CEXn引脚为低电平 1：CEXn引脚为高电平 当SMn[1:0] =01时，该位为波形输出控制位 0：CEXn引脚不输出软件波形 1：CEXn引脚允许软件波形输出 当SMn[1:0] =11时，该位为PWM输出取反使能位 0：PWM正常输出波形（Duty为低电平有效） 1：PWM反向输出波形（Duty为高电平有效）
1	MATn	**匹配标志控制位** 0：不置起相应模块的匹配标志 1：置起相应模块的匹配标志
0	ECCFn	**捕捉/比较标志中断允许位** 0：禁止CCFn中断 1：当CCFn位被置1时，允许捕捉/比较标志的中断请求

21.5.4 FORCE 强制输出控制寄存器 (PCAx_FORCE)

偏移地址: 0x0010 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留												OSC1	OSC0	FCO1	FCO0
-												*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 4	保留	-
3	OSC1	**模块1引脚CEX1输出寄存器，该位只有当SMn[1:0] =01且FSn[1:0] =1x时有效** 给该位写0，CEX1引脚输出高电平，当发生比较匹配时，CEX1引脚被清低电平； 给该位写1，CEX1引脚输出低电平，当发生比较匹配时，CEX1引脚被置高电平；
2	OSC0	**模块0引脚CEX0输出寄存器，该位只有当SMn[1:0] =01且FSn[1:0] =1x时有效** 给该位 写0，CEX0引脚输出高电平，当发生比较匹配时，CEX0引脚被清低电平； 给该位写1，CEX0引脚输出低电平，当发生比较匹配时，CEX0引脚被置高电平；
1	FCO1	**模块1强制匹配控制位（该位只有当SMn[1:0] =01时有效）** 0：不使能强制匹配 1：使能强制匹配，匹配后硬件自动清零
0	FCO0	**模块0强制匹配控制位（该位只有当SMn[1:0] =01时有效）** 0：不使能强制匹配 1：使能强制匹配，匹配后硬件自动清零

21.5.5 PCA 计数最大值 (PCAx_TOPR)

偏移地址: 0x0014 + 0x400*x

复位值: 0x0000 FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TOPCAS[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TOPH[7:0]								TOPL[7:0]							
RW								RW							
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	符号	说明
31 ~ 16	TOPCAS[15:0]	**32位PCA TOP值高16位：只有PCA0该位有效** 注：只有级联时有效，例如PCA1.[TOPH:TOPL]与PCA0.TOPCAS的读写值都一样。
15 ~ 8	TOPH[7:0]	**TOPH：PCA TOP高8位设置**
7 ~ 0	TOPL[7:0]	**TOPL：PCA TOP低8位设置**

21.5.6 通道 n 比较/捕捉模块寄存器 (PCAx_CPRn)(n=0..1)

偏移地址: 0x0018 + 0x400*x

:0x001C + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CPnCAS[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CPHn[7:0]								CPLn[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**CPnCAS[15:0]**	**32位PCA CPn捕捉模块n高16位：只有PCA0该位有效** 注：只有级联时有效，例如PCA1.[CPHn:CPLn]与PCA0.CPnCAS的读写值都一样。
15 ~ 8	**CPHn[7:0]**	**CPHn寄存器保存捕捉模块n高8位的值**
7 ~ 0	**CPLn[7:0]**	**CPLn寄存器保存捕捉模块n低8位的值**

21.5.7 PCA 中断标志和清除寄存器 (PCAx_PCAINTF)

偏移地址: 0x0020 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留										CFC	CCF 1C	CCF 0C	CF	CCF 1	CCF 0
-										WO	WO	WO	RO	RO	RO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 6	保留	-
5	**CFC**	**PCA计数器/定时器溢出标志清除位** 0：无效 1：清除
4	**CCF1C**	**CCF1：PCA模块1捕捉/比较标志清除位** 0：无效 1：清除
3	**CCF0C**	**CCF0：PCA模块0捕捉/比较标志清除位** 0：无效 1：清除
2	**CF**	**PCA计数器/定时器溢出标志** 当PCA计数器/定时器从0xFFFF到0x0000溢出时由硬件置位。在计数器/定时器溢出（CF）中断被允许时，该位置1将导致 CPU 转向PCA中断服务程序。该位不能由硬件自动清0，必须用软件清0。
1	**CCF1**	**CCF1：PCA模块1捕捉/比较标志** 在发生一次匹配或捕捉时该位由硬件置位。当CCF1中断被允许时，该位置1将导致CPU转向PCA中断服务程序。该位不能由硬件自动清0必须用软件清0。
0	**CCF0**	**CCF0：PCA模块0捕捉/比较标志** 在发生一次匹配或捕捉时该位由硬件置位。当CCF中断被允许时，该位置1将导致CPU转向PCA中断服务程序。该位不能由硬件自动清0必须用软件清0。

PCA2-7 模块寄存器列表 (基地址: 0x4000 6000)

地址	寄存器名	说明
0x4000 6000 + 0x400*(x-2)	CR	PCAx控制寄存器 (PCA2～7对应x:0~5)
0x4000 6004 + 0x400*(x-2)	CMD	PCAx方式配置寄存器 (PCA2～7对应x:0~5)
0x4000 6008 + 0x400*(x-2)	CPM0	CPM0：PCAx 捕捉/比较寄存器 (PCA2～7对应x:0~5)
0x4000 600C + 0x400*(x-2)	CPM1	CPM1：PCAx 捕捉/比较寄存器 (PCA2～7对应x:0~5)
0x4000 6010 + 0x400*(x-2)	FORCE	PCAx FORCE强制输出控制寄存器 (PCA2～7对应x:0~5)
0x4000 6014 + 0x400*(x-2)	TOPR	PCAx计数最大值 (PCA2～7对应x:0~5)
0x4000 6018 + 0x400*(x-2)	CPR0	PCAx通道0比较/捕捉模块寄存器 (PCA2～7对应x:0~5)
0x4000 601C + 0x400*(x-2)	CPR1	PCAx通道1比较/捕捉模块寄存器 (PCA2～7对应x:0~5)
0x4000 6020 + 0x400*(x-2)	PCAINTF	PCAx中断标志和清除寄存器 (PCA2～7对应x:0~5)

21.5.8 PCA 控制寄存器 (PCAx_CR)

偏移地址: 0x0000 + 0x400*(x-2)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留														CASCEN	PR
-														*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 2	保留	-
1	CASCEN	**PCAx级联(Cascade)使能位(x=2,4,6，只在级联的低位PCA2/4/6有效)** 0：禁止PCA2/3(或PCA4/5或PCA6/7）级联功能 1：使能PCA2/3(或PCA4/5或PCA6/7）级联功能 注：只有在级联的两个PCA都停止运行（PR=0）时才能修改此位
0	PR	**PCA计数器/定时器运行控制位** 0：禁止PCA计数器/定时器 1：允许PCA计数器/定时器

21.5.9 PCA 方式配置寄存器 (PCAx_CMD)

偏移地址: 0x0004 + 0x400*(x-2)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								CFDMA	CF1DMA	CF0DMA	ECF	SDEN	CPS[2:0]		
-								*RW*	*RW*	*RW*	*RW*	*RW*	*RW*		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7	**CFDMA**	**PCA计数器/定时器DMA允许位** 0：禁止DMA请求 1：允许DMA请求
6	**CF1DMA**	**PCA捕捉通道1DMA允许位** 0：禁止DMA请求 1：允许DMA请求
5	**CF0DMA**	**PCA捕捉通道0DMA允许位** 0：禁止DMA请求 1：允许DMA请求
4	**ECF**	**PCA计数器/定时器溢出中断允许位** 0：禁止CF中断 1：当CF被置位时，允许 PCA计数器/定时器溢出的中断请求 该位是PCA计数器/定时器溢出（CF）中断的屏蔽位
3	**SDEN**	**PCA计数模式选择位** 0：PCA工作在向上计数（锯齿波）模式, 此时该PCA其它比较捕捉模块的三角波模式均被禁止 1：PCA工作在中央对齐计数（三角波）模式，此时该PCA其它比较捕捉模块的锯齿波模式均被禁止 当PCA设定为两种计数模式中的一种时，比较捕捉模块的另一种计数模式即使被配置也是无效的。
2 ~ 0	**CPS[2:0]**	**PCA计数器/定时器时钟选择** 这些位选择 PCA计数器的时钟源。具体请参照21.3时钟配置。

21.5.10 CPM0：PCA 捕捉/比较寄存器 (PCAx_CPMn)(n=0..1)

偏移地址: 0x0008 + 0x400*(x-2)

:0x000C + 0x400*(x-2)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								SMn[1:0]		FSn[1:0]		ECOMn	TCPn	MATn	ECCFn
-								*RW*		*RW*		*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7 ~ 6	SMn[1:0]	**PCA模式选择位** 00： 捕捉方式 01：比较匹配输出（软件定时器） 10：频率输出 11：PWM输出 具体请参照表XXXX
5 ~ 4	FSn[1:0]	**当SMn[1:0] =00时：捕捉方式选择位** 0X：比较/捕捉模块0工作在正沿触发方式 10：比较/捕捉模块0工作在负沿触发方式 11：比较/捕捉模块0工作在任意沿触发方式 当SMn[1:0] =01时：软件定时器选择位 0X：连续软件定时模式 1X：单次软件定时模式 当SMn[1:0] =11时：PWM方式选择。 00：选择 8位 PWM方式 01：选择 16位 PWM方式 10：选择16位相位修正PWM方式 11：选择16位相频修正PWM方式 当SMn[1:0] =10（频率输出方式）时此位才无效。
3	ECOMn	**比较/捕捉模块功能使能位** 0：禁止比较/捕捉模块0 1：使能比较/捕捉模块0
2	TCPn	**当SMn[1:0] =00时，该位为捕捉信号边沿指示位** 0：CEXn引脚为低电平 1：CEXn引脚为高电平 当SMn[1:0] =01时，该位为波形输出控制位 0：CEXn引脚不输出软件波形 1：CEXn引脚允许软件波形输出 当SMn[1:0] =11时，该位为PWM输出取反使能位 0：PWM正常输出波形（Duty为低电平有效） 1：PWM反向输出波形（Duty为高电平有效）
1	MATn	**匹配标志控制位** 0：不置起相应模块的匹配标志 1：置起相应模块的匹配标志
0	ECCFn	**捕捉/比较标志中断允许位** 0：禁止CCFn中断 1：当CCFn位被置1时，允许捕捉/比较标志的中断请求

21.5.11 FORCE 强制输出控制寄存器 (PCAx_FORCE)

偏移地址: 0x0010 + 0x400*(x-2)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留												OSC1	OSC0	FCO1	FCO0
-												*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 4	保留	-
3	OSC1	**模块1引脚CEX1输出寄存器，该位只有当SMn[1:0] =01且FSn[1:0]=1x时有效** 给该位写0，CEX1引脚输出高电平，当发生比较匹配时，CEX1引脚被清低电平； 给该位写1，CEX1引脚输出低电平，当发生比较匹配时，CEX1引脚被置高电平；
2	OSC0	**模块0引脚CEX0输出寄存器，该位只有当SMn[1:0] =01且FSn[1:0]=1x时有效** 给该位 写0，CEX0引脚输出高电平，当发生比较匹配时，CEX0引脚被清低电平； 给该位写1，CEX0引脚输出低电平，当发生比较匹配时，CEX0引脚被置高电平；
1	FCO1	**模块1强制匹配控制位（该位只有当SMn[1:0] =01时有效）** 0：不使能强制匹配 1：使能强制匹配，匹配后硬件自动清零
0	FCO0	**模块0强制匹配控制位（该位只有当SMn[1:0] =01时有效）** 0：不使能强制匹配 1：使能强制匹配，匹配后硬件自动清零

21.5.12 PCA 计数最大值 (PCAx_TOPR)

偏移地址: 0x0014 + 0x400*(x-2)

复位值: 0x0000 FFFF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TOPCAS[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TOPH[7:0]								TOPL[7:0]							
RW								RW							
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	符号	说明
31 ~ 16	TOPCAS[15:0]	**32位PCA TOP值高16位：只有PCA2/4/6该位有效** 注：只有级联时有效，例如PCA3/5/7.[TOPH:TOPL]与PCA2/4/6.TOPCAS的读写值都一样。
15 ~ 8	TOPH[7:0]	**TOPH：PCA TOP高8位设置**
7 ~ 0	TOPL[7:0]	**TOPL：PCA TOP低8位设置**

21.5.13 通道 0 比较/捕捉模块寄存器 (PCAx_CPR0)

偏移地址: 0x0018 + 0x400*(x-2)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CP0CAS[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CPH0[7:0]								CPL0[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**CP0CAS[15:0]**	**32位PCA CP0捕捉模块0高16位：只有PCA2/4/6该位有效** 注：只有级联时有效，例如PCA3/5/7.[CPH0:CPL0]与PCA2/4/6.CP0CAS的读写值都一样。
15 ~ 8	**CPH0[7:0]**	**CPH0寄存器保存捕捉模块0高8位的值**
7 ~ 0	**CPL0[7:0]**	**CPL0寄存器保存捕捉模块0低8位的值**

21.5.14 PCA 通道 1 比较/捕捉模块寄存器 (PCAx_CPR1)

偏移地址: 0x001C + 0x400*(x-2)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
CP1CAS[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
CPH1[7:0]								CPL1[7:0]							
RW								*RW*							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	**CP1CAS[15:0]**	**32位PCA CP1捕捉模块1高16位：只有PCA0/2/4/6该位有效** 注：只有级联时有效，例如PCA3/5/7.[CPH1:CPL1]与PCA2/4/6.CP1CAS的读写值都一样。
15 ~ 8	**CPH1[7:0]**	**CPH1寄存器保存捕捉模块1高8位的值**
7 ~ 0	**CPL1[7:0]**	**CPL1寄存器保存捕捉模块1低8位的值**

21.5.15 PCA 中断标志和清除寄存器 (PCAx_PCAINTF)

偏移地址: 0x0020 + 0x400*(x-2)

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留										CFC	CCF 1C	CCF 0C	CF	CCF 1	CCF 0
-										*WO*	*WO*	*WO*	*RO*	*RO*	*RO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 6	保留	-
5	CFC	**PCA计数器/定时器溢出标志清除位** 0：无效 1：清除
4	CCF1C	**CCF1：PCA模块1捕捉/比较标志清除位** 0：无效 1：清除
3	CCF0C	**CCF0：PCA模块0捕捉/比较标志清除位** 0：无效 1：清除
2	CF	**PCA计数器/定时器溢出标志** 当PCA计数器/定时器从0xFFFF到0x0000溢出时由硬件置位。在计数器/定时器溢出（CF）中断被允许时，该位置1将导致 CPU 转向PCA中断服务程序。该位不能由硬件自动清0，必须用软件清0。
1	CCF1	**CCF1：PCA模块1捕捉/比较标志** 在发生一次匹配或捕捉时该位由硬件置位。当CCF1中断被允许时，该位置1将导致CPU转向PCA中断服务程序。该位不能由硬件自动清0必须用软件清0。
0	CCF0	**CCF0：PCA模块0捕捉/比较标志** 在发生一次匹配或捕捉时该位由硬件置位。当CCF中断被允许时，该位置1将导致CPU转向PCA中断服务程序。该位不能由硬件自动清0必须用软件清0。

22. 串行通信接口（UART/SPI/TWI）

22.1 通用异步收发器（UART）

22.1.1 简介

通用异步收发传输器（Universal Asynchronous Receiver/ Transmitter)，通常称作 UART，是一种异步收发传输器。

22.1.2 主要特性

- 自带波特率发生器
- 1/2 位停止位可设
- UART 有四种工作方式
- 增强功能包括帧出错检测及自动地址识别
- 奇偶校验
- 支持 LIN 总线模式
- 支持 DMA 通信

22.1.3 功能描述

22.1.3.1 UART 工作方式

UART 有 4 种工作方式。在通信之前用户必须先初始化 UARTx_CR，选择方式和波特率以及停止位。

在所有四种方式中，在使能了 TEN 后，任何将 UARTx_TDR 作为目标寄存器的写操作都会启动发送。在方式 0 中由条件 RI = 0 和 REN = 1 初始化接收。这会在 TXD 引脚上产生一个时钟信号，然后在 RXD 引脚上移 8 位数据。在其它方式中由输入的起始位初始化接收（如果 RI = 0 和 REN = 1）。外部发送器通信以发送起始位开始。

表22-1 UART 工作方式列表

SM0	SM1	方式	类型	波特率	帧长度	起始位	停止位	第 9 位
0	0	0	同步半双工	PCLK1/（48 或 12）	8 位	无	无	无
0	1	1	异步全双工	自带波特率发生器的溢出率/16	10 位(或 11)	1	1(或 2)	无
1	0	2	异步全双工	PCLK1/（256 或 128）	11 位(或 12)	1	1(或 2)	0，1
1	1	3	异步全双工	自带波特率发生器的溢出率/16	11 位(或 12)	1	1(或 2)	0，1

方式 0：同步，半双工通讯

方式 0 支持与外部设备的同步通信。在 RXD 引脚上收发串行数据，TXD 引脚发送移位时钟。SH32F9001_32F9801 提供 TXD 引脚上的移位时钟，因此这种方式是串行通信的半双工方式。在这个方式中，每帧收发 8 位，低位先接收或发送。

通过置 SM2 位@UARTx_CR 为 0 或 1，波特率固定为 PCLK1 时钟的 1/48 或 1/12。当 SM2 位等于 0 时，串行端口以 PCLK1 时钟的 1/48 运行。当 SM2 位等于 1 时，串行端口以 PCLK1 时钟的 1/12 运行。

功能块框图如下图所示。数据通过 RXD 引脚移入和移出串行端口，移位时钟由 TXD 引脚输出。

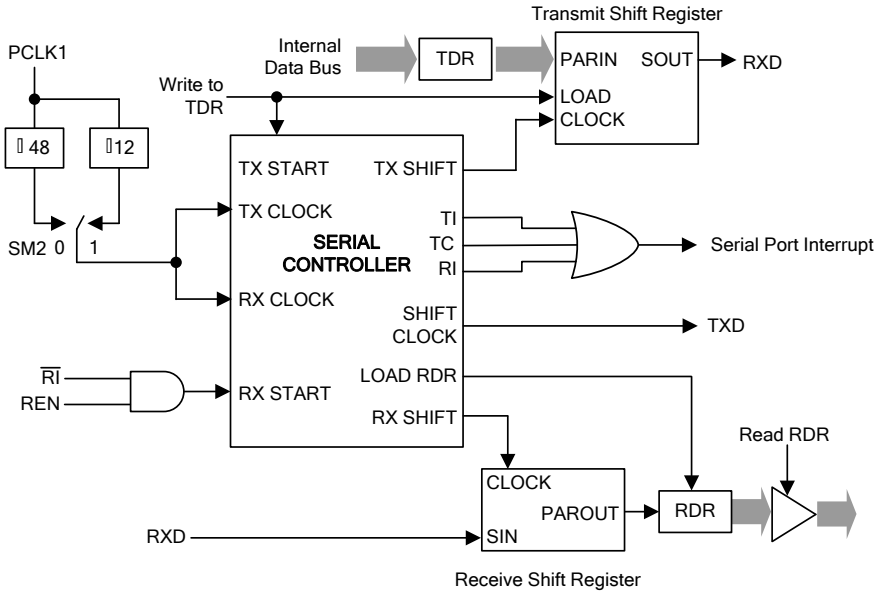


图22-1　　方式 0 功能块框图

在使能了 TEN@UARTx_CR 后,UARTx_TDR 寄存器为空，写操作可以被执行。写入数据下一个 PCLK1 时钟 TX 控制块开始发送。数据转换发生在移位时钟的下降沿，移位寄存器的内容逐次从左往右移位，空位置 0。当移位寄存器中的所有 8 位都发送后，TX 控制模块停止发送操作，然后在下一个 PCLK1 时钟的上升沿将数据从 UARTx_TDR 寄存器搬移置 SHIFT 寄存器，此时，UARTx_TDR 又为空，TI @UARTx_FR 硬件置位（TI 为发送中断标志，实时反映发送寄存器状态，置位表示发送寄存器为空）。

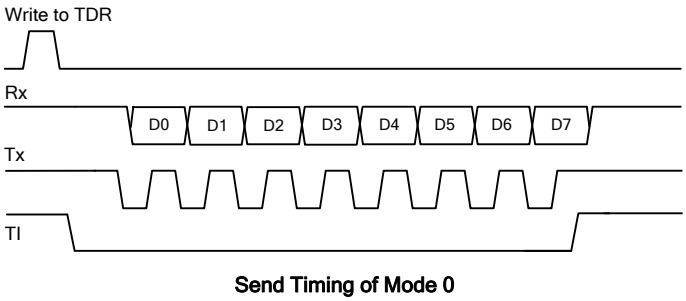
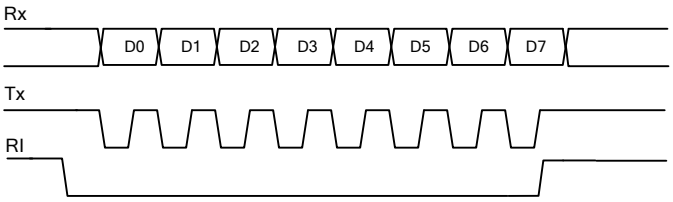


图22-2　　方式 0 发送时序图

REN@UARTx_CR 置 1 和 RI@UARTx_FR 写 1 清零初始化接收。下一个 PCLK1 时钟启动接收，在移位时钟的上升沿锁存数据，接收转换寄存器的内容逐次向左移位。当所有 8 位数据都移到移位寄存器中后，RX 控制块停止接收，在下一个 PCLK1 时钟的上升沿 RI 置位，直到被软件读取 RDR[7:0]@UARTx_RDR，RI 随之被清零（或写 1 清零 RI）才允许下一次接收。（RI 为接收中断标志，实时反映发送寄存器状态，置位表示发送寄存器为满。）否则接收到 SHIFT 内的数据将会覆盖 UARTx_RDR 内未被取走的数据，且会产生 RXOV 错误标志。



Receive Timing of Mode 0

图22-3　　方式 0 接收时序图

方式 1：8 位 UART，可变波特率，异步全双工

方式 1 提供 10/11 位全双工异步通信，10 位由一个起始位（逻辑 0），8 个数据位（低位为第一位）和 1/2 位停止位（逻辑 1）组成。在接收时，这 8 个数据位存储在 RDR[7:0]@UARTx_RDR 中而停止位储存在 RB8@UARTx_CR 中。方式 1 中的波特率固定为自带波特率发生器溢出率的 1/16。功能块框图如下图所示：

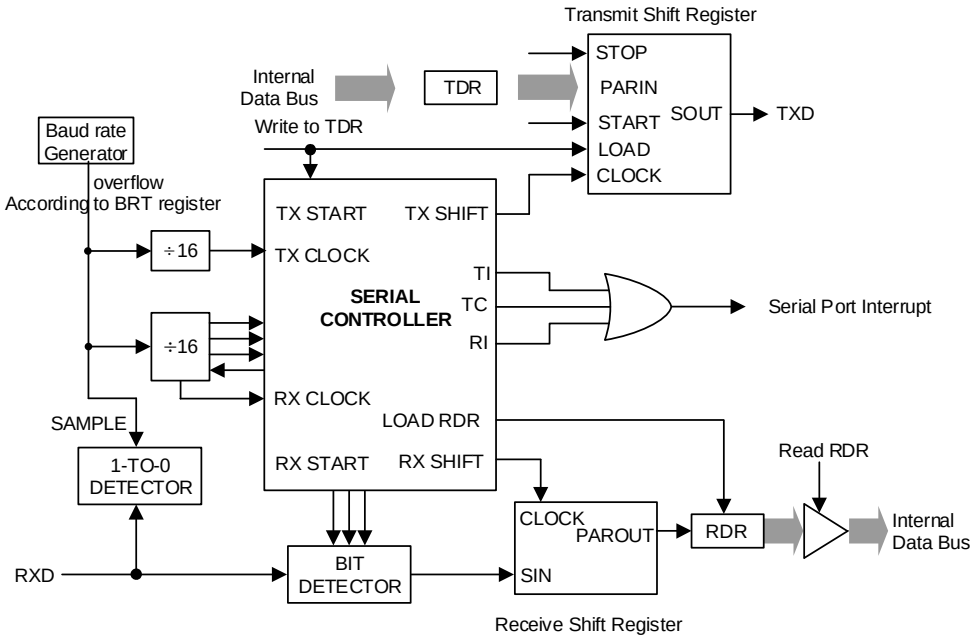
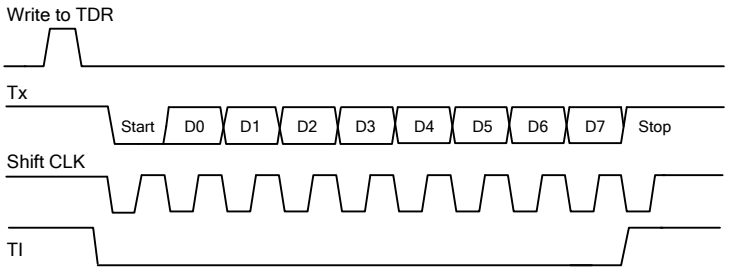


图22-4　　方式 1 功能块框图

在使能了 TEN@UARTx_CR 后, UARTx_TDR 寄存器为空，写操作可以被执行，实际上 TXD 引脚上发送是从 16 分频计数器中的下一次跳变之后的 PCLK1 时钟开始的，因此位时间与 16 分频计数器是同步的。起始位首先在 TXD 引脚上移出，然后是 8 位数据位。在发送移位寄存器中的所有 8 位数据都发送完后，停止位在 TXD 引脚上移出，在停止位发送开始的同时 TI 标志置位。



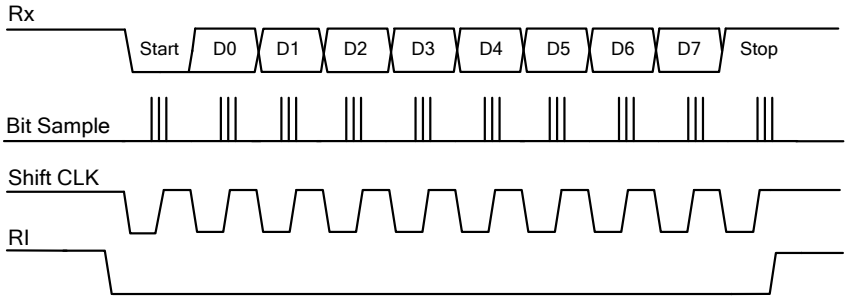
Transmit Timing of Mode 1

图22-5　　方式 1 发送时序图(STOP=1BIT)

只有 REN 置位时才允许接收。当 RXD 引脚检测到下降沿时串行口开始接收串行数据。为此，CPU 对 RXD 不断采样，采样速率为波特率的 16 倍。当检测下降沿时，16 分频计数器立即复位，这有助于 16 分频计数器与 RXD 引脚上的串行数据位同步。16 分频计数器把每一位的时间分为 16 个状态，在第 7、8、9 状态时，位检测器对 RXD 端的电平进行采样。为抑制噪声，在这 3 个状态采样中至少有 2 次采样值一致数据才被接收。如果所接收的第一位不是 0，说明这位不是一帧数据的起始位，该位被忽略，接收电路被复位，等待 RXD 引脚上另一个下降沿的到来。若起始位有效，则移入移位寄存器，并接着移入其它位到移位寄存器。8 个数据位和 1 个停止位移入之后，移位寄存器的内容被分别装入 RDR[7:0]@UARTx_RDR 和 RB8 中，RI 置位，但必须满足下列条件：

1. RI = 0
2. SM2 = 0 或者接收的停止位 = 1

如果这些条件被满足，那么停止位装入 RB8，8 个数据位装入 RDR[7:0]@UARTx_RDR，RI 被置位。否则接收的帧会丢失，继续检测下一个起始位。这时，接收器将重新去探测 RXD 端是否另一个下降沿。用户应将 RDR[7:0]@UARTx_RDR 内数据取走（或用软件写 1 清零 RI），然后才能再次接收，否则接收到 SHIFT 内的数据将会覆盖 UARTx_RDR 内未被取走的数据，且会产生 RXOV 错误标志。



Receive Timing of Mode 1

图22-6　　方式 1 接收时序图(STOP=1BIT)

方式 2：9 位 UART，固定波特率，异步全双工

这个方式使用异步全双工通信中的 11/12 位。一帧由一个起始位（逻辑 0），8 个数据位（低位为第一位），一个可编程的第 9 数据位和一个/两个停止位（逻辑 1）组成。方式 2 支持多机通信和硬件地址识别（详见多机通讯章节）。在数据传送时，第 9 数据位（TB8@UARTx_CR）可以写 0 或 1，该位的设置需要在写入相应 TDR[7:0]@UARTx_TDR 之前完成。第 9 位可用作多机通信中的数据/地址标志位，以及奇偶校验位。当接收到数据时，第 9 数据位移入 RB8 而停止位不保存。若使用软件读取 RB8 位需要在读取 RDR[7:0]@UARTx_RDR 之前读取。通过 SMOD@UARTx_CR 位选择波特率为 PCLK1 的 1/256 或 1/128。功能块框图如下所示：

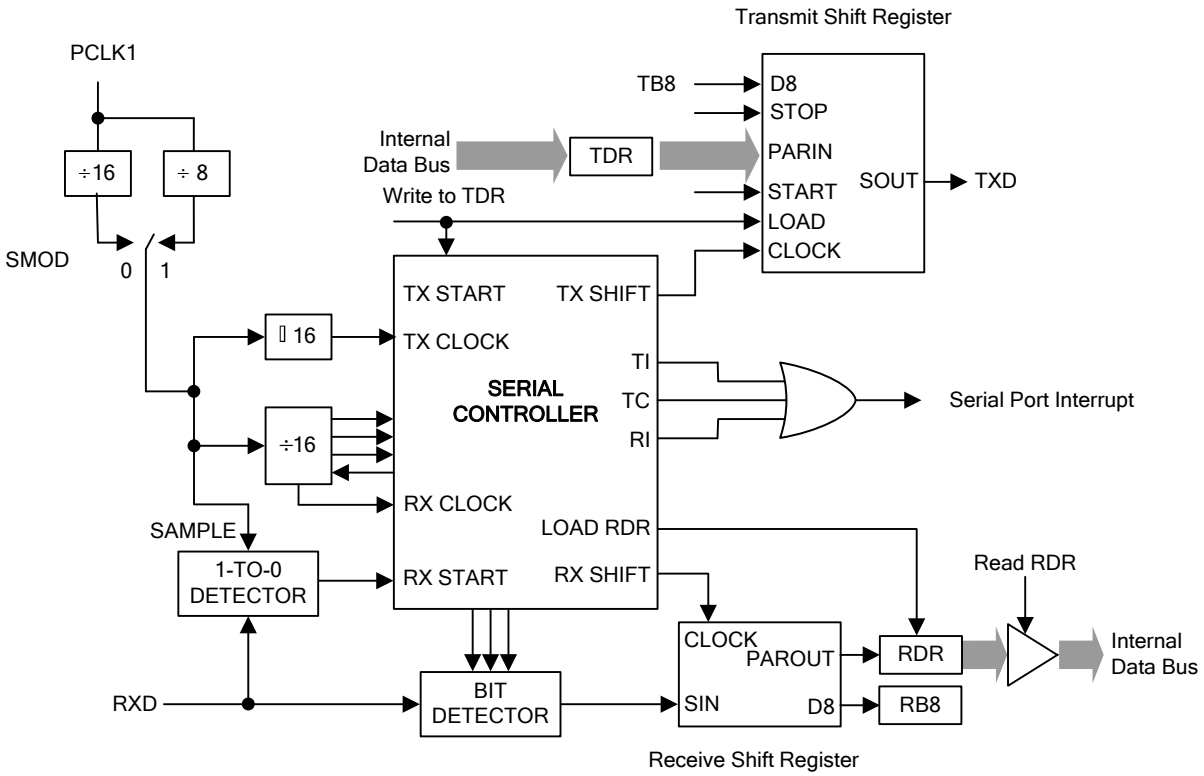
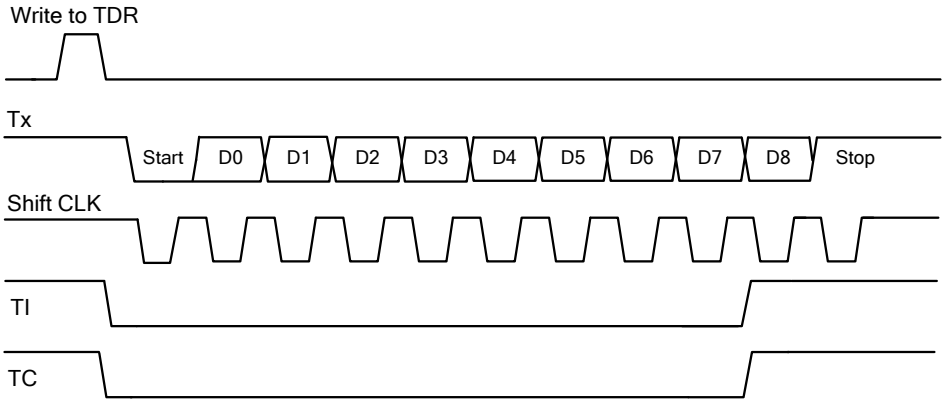


图22-7　　方式 2 功能块框图

在使能了 TEN@UARTx_CR 后, UARTx_TDR 寄存器为空，写操作可以被执行，同时也将 TB8 载入到发送移位寄存器的第 9 位中。实际上 TXD 发送是从 16 分频计数器中的下一次跳变之后的 PCLK1 时钟开始的，因此位时间与 16 分频计数器是同步的。起始位首先在 TXD 引脚上移出，然后是 9 位数据。在发送转换寄存器中的所有 9 位数据都发送完后，停止位在 TXD 引脚上移出，在第 8 位数据发送结束后 TI 标志置位。



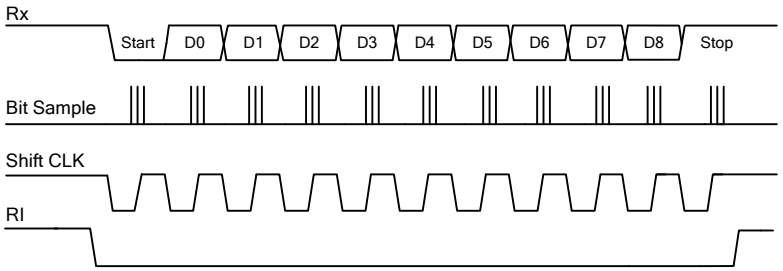
Transmit Timing of Mode 2

图22-8　　方式 2 发送时序图(STOP=1BIT)

只有 REN 置位时才允许接收。当 RXD 引脚检测到下降沿时串行口开始接收串行数据。为此，CPU 对 RXD 不断采样，采样速率为波特率的 16 倍。当检测下降沿时，16 分频计数器立即复位。这有助于 16 分频计数器与 RXD 引脚上的串行数据位同步。16 分频计数器把每一位的时间分为 16 个状态，在第 7、8、9 状态时，位检测器对 RXD 端的电平进行采样。为抑制噪声，在这 3 个状态采样中至少有 2 次采样值一致数据才被接收。如果所接收的第一位不是 0，说明这位不是一帧数据的起始位，该位被忽略，接收电路被复位，等待 RXD 引脚上另一个下降沿的到来。若起始位有效，则移入移位寄存器，并接着移入其它位到移位寄存器。9 个数据位和 1 个停止位移入之后，移位寄存器的内容被分别装入 UARTx_RDR 和 RB8 中，RI 置位，但必须满足下列条件：

1. RI = 0
2. SM2 = 0 或者接收的第 9 位 = 1，且接收的字节符合约定从机地址

如果这些条件被满足，那么第 9 位移入 RB8，8 位数据移入 UARTx_RDR，RI 被置位。否则接收的数据帧会丢失。在停止位的当中，接收器回到寻找 RXD 引脚上的另一个下降沿。用户应将 UARTx_RDR 内数据取走使（或用软件写 1 清除 RI），然后才能再次接收。否则接收到 SHIFT 内的数据将会覆盖 UARTx_RDR 内未被取走的数据，且会产生 RXOV 错误标志。



Receive Timing of Mode 2

图22-9　　方式 2 接收时序图(STOP=1BIT)

方式 3：9 位 UART，可变波特率，异步全双工

方式 3 使用方式 2 的传输协议以及方式 1 的波特率产生方式。

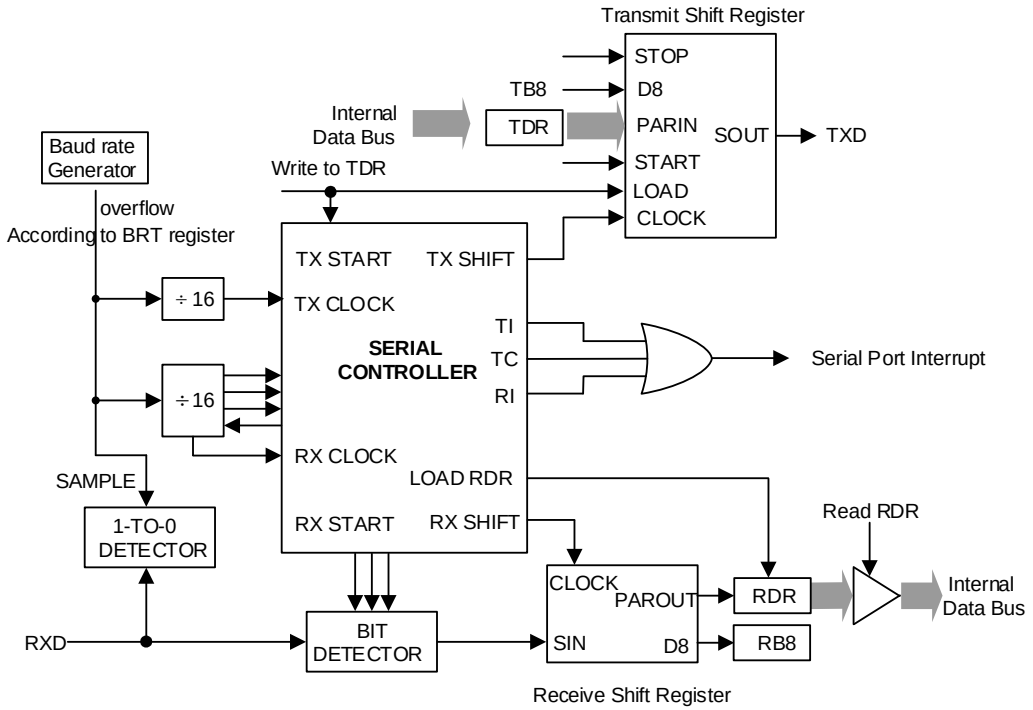


图22-10　　方式 3 功能块框图

关于TI,TC标志的说明

TI@UARTx_FR为发送中断标志位。若TDR[7:0]@UARTx_TDR为空，TI将被置位。对TDR[7:0]@UARTx_TDR写入数据后，TI将自动清零。但是需要注意的是，在连续发送数据时，第一个数据是直接写穿到发送移位寄存器里，第二个数据才被写入到TDR[7:0]@UARTx_TDR里。所以在第二个数据写入到TDR[7:0]@UARTx_TDR内后，TI标志位才会清零。（如图22-11所示）

TC@UARTx_FR为连续帧发送结束标志位。该位通过以下操作序列来实现清零：step1，读UARTx_FR寄存器; step2，写UARTx_TDR寄存器。对TCC@UARTx_FR写1也可以清零TC位。当连续发送的一帧数据的最后一Byte数据发送结束后（停止位发送结束后），并且检测到TI@UARTx_FR为高的情形下，TC标志位置起。（如图22-11所示）

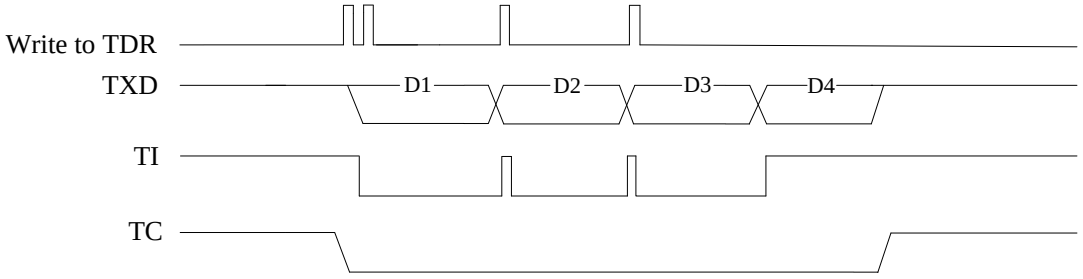


图22-11　　TC 工作时序图

22.1.3.2　　可微调波特率

UART 自带一个波特率发生器，它实质上就是一个 15 位递增计数器。

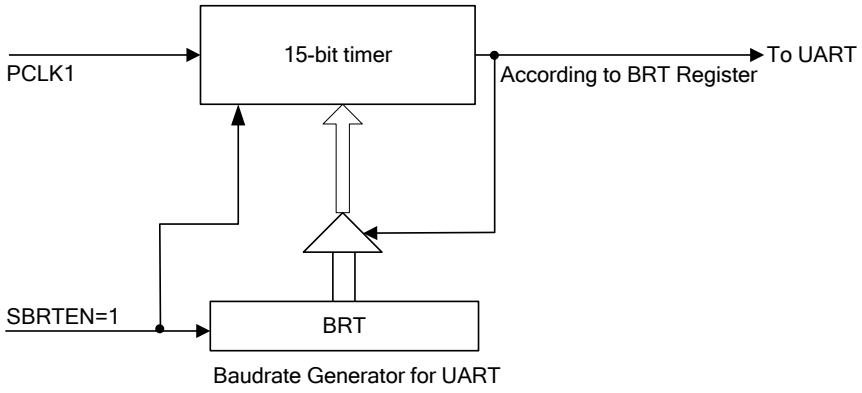


图22-12　　波特率发生器功能块框图

由图得到，波特率发生器的溢出率为

$$SBRToverflowrate = \frac{PCLK1}{BRT}$$

，

因此，UART 在各模式下的波特率计算公式如下。

在方式 0 中，波特率可编程为 APB1 时钟的 1/48 或 1/12，由 SM2 位决定。当 SM2 为 0 时，串行端口在 PCLK1 时钟的 1/48 下运行。当 SM2 为 1 时，串行端口在 PCLK1 时钟的 1/12 下运行。

在方式 1 和方式 3 中，波特率可微调，精度为一个 PCLK1 时钟，公式如下：

$$BaudRate = \frac{PCLK1}{16 \times (SBRT+1)+BFINE}$$

例如：PCLK1 = 16MHz，需要得到 9600Hz 的波特率，SBRT 和 BFINE 值计算方法如下：
16000000/16/9600 = 104.17
SBRT = 104 - 1 = 103
由 BaudRate 计算公式：9600 = 16000000/(16 X 104 + BFINE)
得到：BFINE = 3
此微调方式计算出的实际波特率为 9598，误差为 0.02%。

在方式 2 中，波特率固定为 PCLK1 时钟的 1/256 或 1/128，由 SMOD 位@UARTx_CR 中决定。当 SMOD 位为 0 时，UART 以 PCLK1 时钟的 1/256 运行。当 SMOD 位为 1 时，UART 以 PCLK1 时钟的 1/128 运行。

$$BaudRate = 2^{SMOD} \times (\frac{PCLK1}{256})$$

注意：无论使用哪种方式进行通讯，必须使能 UART 波特率发生器，即 SBRTEN=1。

22.1.3.3 多机通讯

自动（硬件）地址识别

在方式 2 和方式 3 中，SM2 置位，UART 运行状态如下：接收到停止位，RB8 的第 9 位为 1（地址字节标志），且接收到的数据字节符合 UART 的从机地址，UART 将开始接收后续数据字节。

第 9 位为 1 表明该字节是地址而非数据。当主机要发送一组数据给几个从机中的一个时，必须先发送目标从机地址。所有从机等待接收地址字节。自动地址识别的特点是只有地址匹配的从机才能产生后续接收数据的中断标志，硬件完成地址比较。

地址匹配的从机才能继续接收数据字节并产生接收中断标志。地址不匹配的从机不受影响，将继续等待接收和它匹配的地址字节。

使用自动地址识别功能时，主机可以通过调用给定的从机地址选择与一个或多个从机通信。主机使用广播地址可以寻址所有从机。有个特殊功能寄存器 UARTx_ADDR，包含了从机地址（SADDR）和地址屏蔽（SAMR）。从机地址是一个 8 位的字节,SAMR 用于定义 SADDR 各位的有效与否，如果 SAMR 中某一位为 0，则 SADDR 中相应位被忽略，如果 SAMR 中某一位置位，则 SADDR 中相应位将用于产生约定地址。这可以使用户在不改变 SADDR 寄存器中的从机地址的情况下灵活地寻址多个从机。

表22-2 自动地址识别范例表

	从机 1	**从机 2**
SADDR	10100100	10100111
SAMR（为 0 的位被忽略）	11111010	11111001
约定地址	10100x0x	10100xx1
广播地址（SADDR 或 SAMR）	1111111x	11111111

从机 1 和从机 2 的约定地址最低位是不同的。从机 1 忽略了最低位，而从机 2 的最低位是 1。因此只与从机 1 通讯时，主机必须发送最低位为 0 的地址（10100000）。类似地，从机 1 的第 1 位为 0，从机 2 的第 1 位被忽略。因此，只与从机 2 通讯时，主机必须发送第 1 位为 1 的地址（10100011）。如果主机需要同时与两从机通讯，则第 0 位为 1，第 1 位为 0，第 2 位被两从机都忽略，两个不同的地址用于选定两个从机（1010 0001 和 1010 0101）。

主机可以通过广播地址与所有从机同时通讯。这个地址等于 SADDR 和 SAMR 的位或，结果中的 0 表示该位被忽略。多数情况下，广播地址为 0xFFh，该地址可被所有从机应答。

系统复位后，SADDR 和 SAMR 两个寄存器初始化为 0，这两个结果设定了约定地址和广播地址为 XXXXXXXX（所有位都被忽略）。这有效地去除了多从机通讯的特性，禁止了自动寻址方式。这样的 UART 将对任何地址都产生应答。用户可以按照上面提到的方法实现软件地址识别的多机通讯。

22.1.3.4 帧出错检测

UART 具有帧出错检测功能。错误标志位被置位后，只能通过软件对相应清除位写 1 来清零，尽管后续接收的帧没有任何错误也不会自动清零。该错误标志的置位不会触发中断，不影响串口数据继续发送，只供软件查询使用。

发送冲突

如果 TDR[7:0]@UARTx_TDR 内有数据，用户软件继续写数据到 TDR[7:0]@UARTx_TDR 时，发送冲突位 TXCOL@UARTx_FR 置位。如果发生了冲突，新数据会被忽略，不能被写入发送缓冲器。该位的清零通过对 TXCOLC@UARTx_FR 写 1 来实现。

接收溢出

如果在 UARTx_RDR 中的数据未被读取之前，又有新的数据存入 UARTx_RDR，那么接收溢出位 RXOV@UARTx_FR 置位。如果发生了接收溢出，UARTx_RDR 中未被取走的数据将丢失。该位的清零通过对 RXOVC@UARTx_FR 写 1 来实现。

奇偶校验出错

如果检测到一个错误的奇偶校验位，那么奇偶校验位 PE@UARTx_FR 置位。该位的清零通过对 PEC@UARTx_FR 写 1 来实现。

帧出错

如果检测到一个无效（低）停止位，那么帧出错位 FE@UARTx_FR 置位。该位的清零通过对 FEC@UARTx_FR 写 1 来实现。

注意：*在发送之前，TXD 引脚必须被设置为输出高电平。*

22.1.3.5 奇偶校验

奇校验：校验位与传输数据（二进制表示）中‘1’的个数为奇数。

偶校验：校验位与传输数据（二进制表示）中‘1’的个数为偶数。

选择方式 0 和方式 1 时，没有奇偶校验功能。选择方式 2 和方式 3 时，若奇偶校验允许位 PCE@UARTx_CR 为 0 且 SM2@UARTx_CR 为 0，第九位 TB8@UARTx_CR 将用为奇偶校验位，该位数据由移位寄存器生成并自动赋值到第九位。校验方式选择由 PS@UARTx_CR 设定。

奇偶校验位为硬件自动检测，在每个字节传送完成后，若发生奇偶校验错误自动将就校验错误 PE@UARTx_FR 位置 1，该位由软件对 PEC@UARTx_FR 写 1 实现清零。

注：无论校验是否正确，数据都会存入 RDR[7:0]@UARTx_RDR，校验位都会存入 RB8@UARTx_RDR。

22.1.3.6 LIN 总线模式

LIN 模式通过 LINEN@UARTx_CR 进行控制。LIN 模式下，使用 UART 的 TXD 和 RXD 引脚作为 LIN 发送接收功能管脚，UART 的模式应选择 MODE1。

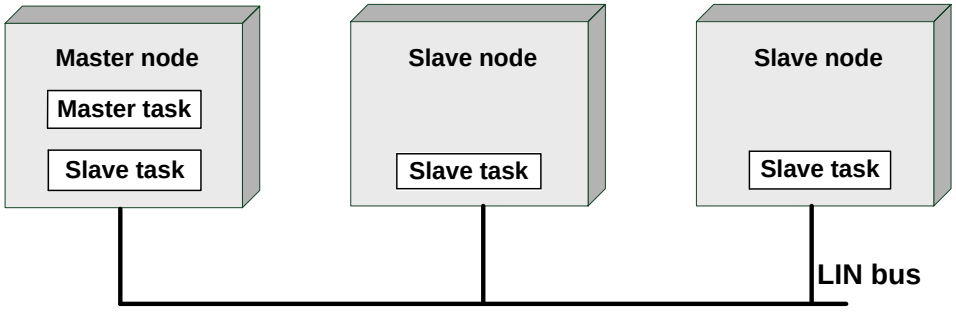


图22-13 LIN 总线模式

LIN 发送，传送字节与普通的 UART 类似。依照 LIN 总线协议，先发送同步间隔（SBK@UARTx_CR）作为 LIN 总线上信号传输的开始，然后发送 0X55 作为波特率适应的同步域数据，最后发送标识符场来控制总线上的数据通讯。主从机后续通信的数据格式均为 8 位数据（LSB），无校验位。在数据帧末端带有一个校验和场。校验和场有两种：仅根据数据字节进行计算的校验和被称为传统的校验和，它可以用于 LIN1.3 从机通讯；根据数据字节和标识符进行计算的校验和被称为增强校验和，可以与 LIN2.0 从机进行通讯。主机根据校验和判断有没有达到目的，进而决定要不要再次发起 LIN 总线通讯。

LIN 接收，在 UART 的 RXD 接口有同步间隔的检测电路，当检测到长度大于为 11/10 个 bit 位的低电平（通过 LBDL@UARTx_CR 设置），判定为打断分隔符。随即产生一个 LBD 中断标志，若响应中断允许位 LBDIE 使能则进入中断服务程序。在 LBD 通过 LBDC 写 1 清零的时候需要将接收移位寄存器与接收缓存清空，并且关闭 LINEN,以便接收后续移入的串行数据。如还需要重新检测同步间断信号，需要重新使能 LINEN。

22.1.3.7 利用 DMA 通信

用户可选择通过使用 DMA 操作 UART 的发送和或接收。DMA 模式由 UARTx_CR 寄存器中的 DMAT 和 DMAR 模式选择位分别决定决定发送与接收是否采用 DMA 总线进行通讯。选择 DMA 模式后，建议将 UART 的相应中断使能位关闭，因为 DMA 模式下 DMA 请求无需通过 CPU 参与便可以自行处理数据流传输。

UART 发送

在 DMA 模式中，当发送数据缓存 UARTx_TDR 数据发送完毕为空的情况下，串口发送的 DMA 请求就会生效，以请求 DMA 将后续数据传输到发送数据缓存 UARTx_TDR 内数据。串口发送 DMA 请求由 DMA 控制器清除。

UART 接收

在 DMA 模式中，当接收数据缓存 UARTx_RDR 数据更新完毕为满的情况下，串口接收的 DMA 请求就会生效，以请求 DMA 将接收数据缓存 UARTx_RDR 内数据取走。串口接收 DMA 请求由 DMA 控制器清除。

22.1.3.8 中断控制

UART 的中断（TI,RI）在 UARTx_TDR 为空和 UARTx_RDR 为满时产生。注意：当 TEN 未使能，TIE 使能时，会触发中断将数据搬移到发送缓存内，但是数据将会在使能 TEN 后才发送。

采用 DMA 通道时，UART 的读写请求信号直接传送至 DMA 控制器，中断信号由 DMA 控制器负责清 0；

当包含有数据的一帧发送完成后，并且 TI=1 时，由硬件将 TC 位置 1。

LBD 中断发生在使能 LIN 总线功能后，在 RXD 上检测到 break 信号时发出。

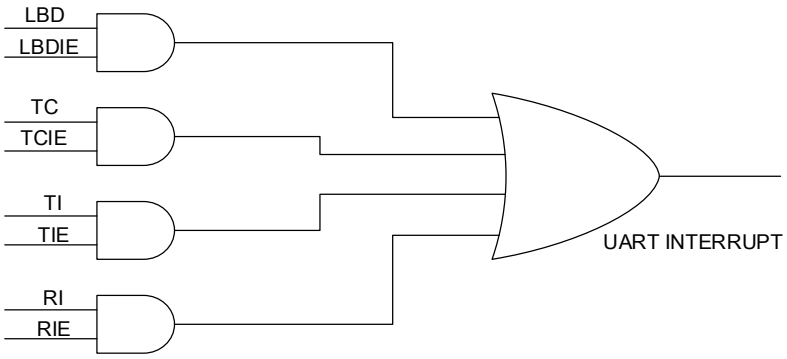


图22-14 UART 中断控制图

22.1.4 UART 寄存器

UART0-5 模块寄存器列表 (基地址: 0x4002 0000)

地址	寄存器名	说明
0x4002 0000 + 0x400*x	FR	UARTx中断标志寄存器 (UART0～5对应x:0~5)
0x4002 0008 + 0x400*x	TDR	UARTx发送数据寄存器 (UART0～5对应x:0~5)
0x4002 000C + 0x400*x	RDR	UARTx接收数据寄存器 (UART0～5对应x:0~5)
0x4002 0010 + 0x400*x	ADDR	UARTx地址配置寄存器 (UART0～5对应x:0~5)
0x4002 0014 + 0x400*x	BRT	UARTx波特率配置寄存器 (UART0～5对应x:0~5)
0x4002 0018 + 0x400*x	CR	UARTx控制寄存器 (UART0～5对应x:0~5)

22.1.4.1 UART 中断标志寄存器 (UARTx_FR)

偏移地址: 0x0000 + 0x400*x

复位值: 0x0000 0006

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留								LBDC	PEC	FEC	RXOVC	TXCOLC	TCC	保留	RIC
-								*WO*	*WO*	*WO*	*WO*	*WO*	*WO*	-	*WO*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								LBD	PE	FE	RXOV	TXCOL	TC	TI	RI
-								*RO*	*RO*	*RO*	*RO*	*RO*	*RO*	*RO*	*RO*
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

位	符号	说明
31 ~ 24	保留	-
23	**LBDC**	**LIN同步间隔中断标志清零位** 0：无操作 1：清零LBD
22	**PEC**	**UART奇偶校验出错标志位** 0：无操作 1：清零PE
21	**FEC**	**UART帧出错标志清零位** 0：无操作 1：清零FE
20	**RXOVC**	**UART接收溢出标志清零位** 0：无操作 1：清零RXOV
19	**TXCOLC**	**UART发送冲突标志清零位** 0：无操作 1：清零TXCOL
18	**TCC**	**UART全部一帧数据发送完毕标志清零位** 0：无操作 1：清零TC 注：TC通过对TCC写入1来清零，也可以先读FR再写TDR的操作序列亦可清零
17	保留	-
16	**RIC**	**UART一字节数据接收中断标志清零位**

		0：无操作 1：清零RI 注：RI通过对RIC写入1来清零，也可以对RDR读操作来清零
15 ~ 8	**保留**	**-**
7	**LBD**	**LIN同步间隔中断标志位** 0：未接收到同步间隔打断信号 1：接收到同步间隔打断信号 通过写LBDC清零。
6	**PE**	**UART奇偶校验出错标志位** 0：奇偶校验正确 1：奇偶校验出错，由硬件置位 注：对PEC写1清零
5	**FE**	**UART帧出错标志位** 0：无帧出错 1：帧出错，由硬件置位 注：对FEC写1清零
4	**RXOV**	**UART接收溢出标志位** 0：无接收溢出 1：接收溢出，由硬件置位 注：对RXOVC写1清零
3	**TXCOL**	**UART发送冲突标志位** 0：无发送冲突 1：发送冲突，由硬件置位 注：对TXCOLC写1清零
2	**TC**	**UART全部一帧数据发送完毕标志位** 0：全部一帧数据发送中，无置位 1：全部一帧数据发送完毕，由硬件置位 注：TC通过对TCC写入1来清零，也可以也可以写TDR的操作序列清零
1	**TI**	**UART一字节数据发送中断标志位** 0：单字节数据发送中，无置位 1：单字节数据发送完毕，由硬件置位 注：TI只能通过对TDR写操作来清零
0	**RI**	**UART一字节数据接收中断标志位** 0：单字节数据发接收中，无置位 1：单字节数据接收完毕，由硬件置位 注：RI通过对RIC写入1来清零，也可以对RDR读操作来清零

22.1.4.2 UART 发送数据寄存器 (UARTx_TDR)

偏移地址: 0x0008 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								**TDR[7:0]**							
-								*RW*							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	**保留**	**-**
7 ~ 0	**TDR[7:0]**	**UART的发送数据寄存器**

22.1.4.3 UART 接收数据寄存器 (UARTx_RDR)

偏移地址: 0x000C + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								RDR[7:0]							
-								RO							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7 ~ 0	RDR[7:0]	UART的接收数据寄存器

22.1.4.4 UART 地址配置寄存器 (UARTx_ADDR)

偏移地址: 0x0010 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SMAR[7:0]								SADDR[7:0]							
RW								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 8	SMAR[7:0]	位屏蔽寄存器，决定了SADDR的哪些位会被检测
7 ~ 0	SADDR[7:0]	UART的硬件地址

22.1.4.5 UART 波特率配置寄存器 (UARTx_BRT)

偏移地址: 0x0014 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留												BFINE[3:0]			
-												RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留	SBRT[14:0]														
-	RW														
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 20	保留	-
19 ~ 16	BFINE[3:0]	波特率发生器微调数据
15	保留	-
14 ~ 0	SBRT[14:0]	波特率发生器调节数据

22.1.4.6 UART 控制寄存器 (UARTx_CR)

偏移地址: 0x0018 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留											DMAT	DMAR	TEN	REN	LINEN
-											RW	RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SBK	SM[1:0]		SM2	PCE	PS	TB8	RB8	LBDL	LBDIE	TCIE	TIE	RIE	SMOD	SBRTEN	STOP
RW	RW		RW	RW	RW	RW	RO	RW	RW	RW	RW	RW	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 21	保留	-
20	DMAT	**UART发送DMA通道使能位** 0：发送数据不通过DMA传输 1：发送数据通过DMA传输
19	DMAR	**UART接收DMA通道使能位** 0：接收数据不通过DMA传输 1：接收数据通过DMA传输
18	TEN	**UART发送功能使能位** 0：禁用发送功能 1：使能发送功能
17	REN	**UART接收功能使能位** 0：禁用接收功能 1：使能接收功能
16	LINEN	**LIN模式功能使能位** 0：禁止LIN模式 1：使能LIN模式
15	SBK	**LIN同步间隔使能位** 0：不发送同步间隔 1：发送同步间隔，发送完成后自动清零
14 ~ 13	SM[1:0]	**UART串行方式控制位** 00：方式0，同步方式，固定波特率 01：方式1，8位异步方式，可变波特率 10：方式2，9位异步方式，固定波特率 11：方式3，9位异步方式，可变波特率
12	SM2	**多机处理通讯允许位** 0：在方式0下，波特率是PCLK1时钟的1/48 在方式1下，禁止停止位确认检验，任何停止位都会置位RI 在方式2和3下，任何字节都会置位RI 1：在方式0下，波特率是PCLK1时钟的1/12 在方式1下，允许停止位确认检验，只有有效的停止位(1)才能置位RI 在方式2和3下，只有地址字节（第9位=1）才能置位RI
11	PCE	**奇偶校验使能位** 0：禁止奇偶校验 1：使能奇偶校验
10	PS	**奇偶校验选择位** 0：奇校验 1：偶校验
9	TB8	**在UART的方式2和3下发送的第9位**
8	RB8	**在UART的方式2和3下接收的第9位**

7	**LBDL**	**同步间隔检测阈值长度** 0：低电平长度大于10BIT低电平 1：低电平长度大于11BIT低电平
6	**LBDIE**	**LIN打断分隔符中断使能位** 0：禁止LIN打断分隔符中断功能 1：允许LIN打断分隔符中断功能
5	**TCIE**	**全部数据发送完成中断使能位** 0：禁止数据发送完成标志位TC触发中断功能 1：允许数据发送完成标志位TC触发中断功能
4	**TIE**	**单字节传送中断使能位** 0：禁止传输中断标志位TI触发中断 1：允许传输中断标志位TI触发中断
3	**RIE**	**单字节接收中断使能位** 0：禁止接收中断标志位RI触发中断 1：允许接收中断标志位RI触发中断
2	**SMOD**	**UART方式2波特率控制位** 0：在方式2中，波特率为PCLK1时钟的1/256 1：在方式2中，波特率为PCLK1时钟的1/128
1	**SBRTEN**	**UART波特率发生器使能控制位** 0：禁用波特率发生器 1：使能波特率发生器
0	**STOP**	**停止位位数选择位** 0：选择使用1位停止位 1：选择使用2位停止位

22.2 串行外设接口（SPI）

22.2.1 简介

串行外部设备接口（SPI）是一种高速串行通信接口，允许 MCU 与外围设备（包括其它 MCU）进行全双工，同步串行通讯。

下图所示即为典型的由一个主设备和若干从属外部设备组成的SPI总线网络，主设备通过 3 条线连接所有从设备，主设备控制连接从属设备$\overline{SS}$引脚的 4 个并行端口来选中其中一个从属设备进行通讯。

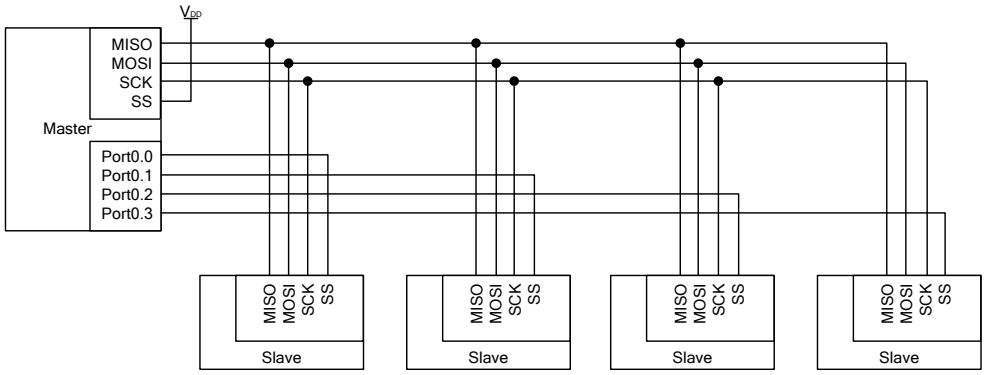


图22-15　　SPI 总线网络图

22.2.2 主要特性

- 全双工，三线同步传输
- 主从机操作
- 10 个可编程主时钟频率
- 极性相位可编程的串行时钟
- 数据字位宽 8、16 可选
- 支持 DMA 通信
- 带 MCU 中断的主模式故障出错标志
- 写入冲突标志保护
- 可选择 LSB 或 MSB 传输

22.2.3 功能描述

22.2.3.1　　信号描述

主输出从输入（MOSI）

该路信号连接主设备和从设备。数据通过 MOSI 从主设备串行传送到从设备，主设备输出，从设备输入。

主输入从输出（MISO）

该路信号连接从设备和主设备。数据通过MISO从从设备串行传送到主设备，从设备输出，主设备输入。当SPI配置为从设备并未被选中（$\overline{SS}$引脚为高电平），从设备的MISO引脚处于高阻状态。

SPI 串行时钟（SCK）

SCK信号用作控制MOSI和MISO线上输入输出数据的同步移动。每 8 时钟周期线上传送一个字节。如果从设备未被选中（$\overline{SS}$引脚为高电平），SCK信号被此从设备忽略。

从设备选择引脚（$\overline{SS}$）

每个从属外围设备由一个从选择引脚（$\overline{SS}$引脚）选择，当引脚信号为低电平时，表明该从设备被选中。主设备可以通过软件控制连接于从设备$\overline{SS}$引脚的端口电平选择每个从设备，很明显，只有一个主设备可以驱动通讯网络。为了防止MISO总

线冲突，同一时间只允许一个从设备与主设备通讯。在主设备模式中，$\overline{SS}$引脚状态关联SPI标志寄存器SPIx_FR中MODF标志位以防止多个主设备驱动MOSI和SCK。

下列情况，$\overline{SS}$引脚可以作为普通端口或其它功能使用：

(1) 设备配置为主设备，SPI 控制寄存器 SPIx_CR 寄存器的 SSDIS 位置 1。这种配置仅仅存在于通讯网络中只有一个主设备的情况，因此， MODF@SPIx_FR 标志位不会被置 1。

(2) 设备配置为从设备，SPI控制寄存器SPIx_CR的CPHA位和SSDIS位置 1。这种配置情况存在于只有一个主设备一个从设备的通讯网络中，因此，设备总是被选中的，主设备也不需要控制从设备的$\overline{SS}$引脚选择其作为通讯目标。

注意：*当CPHA = '0'时，$\overline{SS}$引脚产生下降沿表示启动发送。*

22.2.3.2 波特率

在主模式下，SPI 的波特率有十种可选择的频率，分别是内部时钟的 2，4，8，16，32，64，128，256，512，1024 分频，可以通过设定 SPIx_CR 寄存器的 SPR[3:0]位进行选择。

22.2.3.3 原理描述

下图所示是 SPI 模块的详细结构。

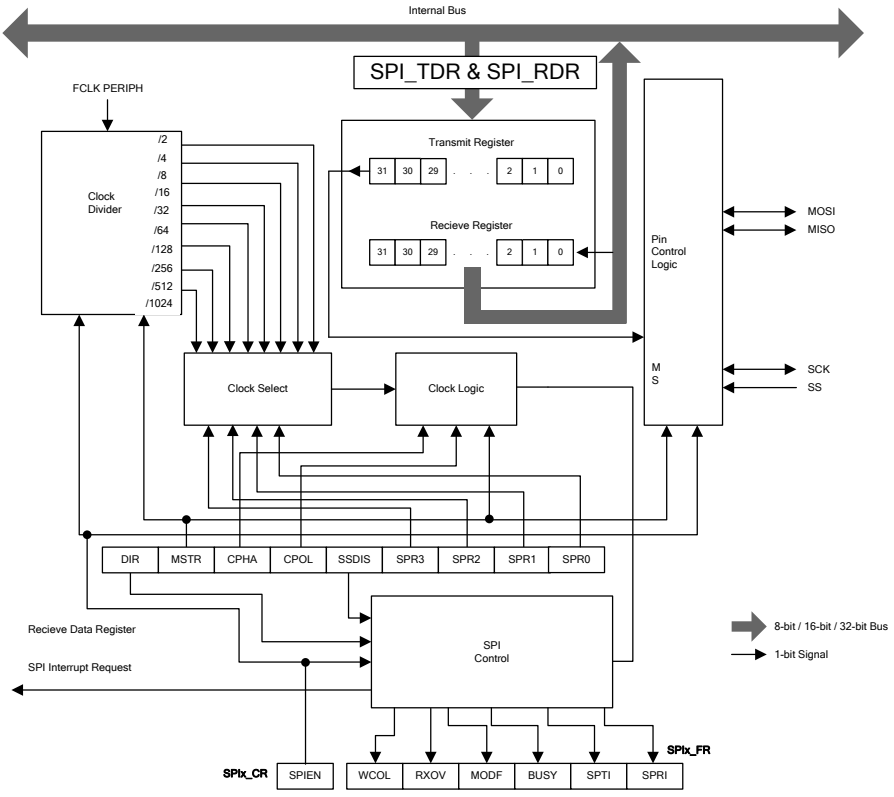


图22-16 SPI 模块框图

22.2.3.4 工作模式

SPI 可配置为主模式或从属模式中的一种。SPI 模块的配置和初始化通过设置 SPIx_CR 寄存器（串行外围设备控制寄存器）来完成。配置完成后，通过设置 SPIx_CR,SPIx_TDR,SPIx_RDR（串行外围设备数据寄存器）来完成数据传送。

在SPI通讯期间，数据同步地被串行的移进移出。串行时钟线（SCK）使两条串行数据线（MOSI和MISO）上数据的移动和采样保持同步。从设备选择线（$\overline{SS}$）可以独立地选择SPI从属设备；如果从设备没有被选中，则不能参与SPI总线上的活动。

当 SPI 主设备通过 MOSI 线传送数据到从设备时，从设备通过 MISO 线发送数据到主设备作为响应，这就实现了在同一时钟下数据发送和接收的同步全双工传输。对 SPI 数据发送寄存器 SPIx_TDR 进行写操作，当缓存器为空时将写入发送缓存器，当缓存器内有数据时将写入发送数据寄存器 SPIx_TDR。对 SPIx_RDR 寄存器进行读操作将获得接收缓存内的数据。

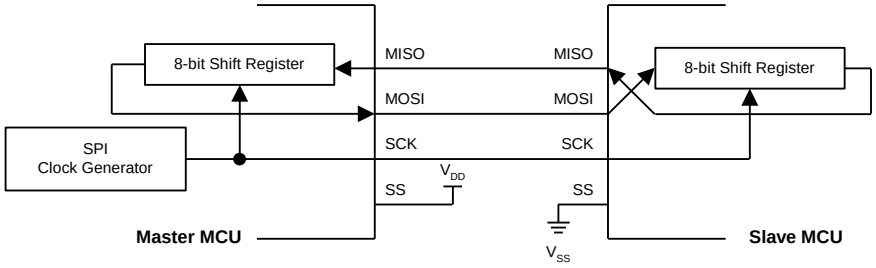


图22-17　　全双工主从互联图

主模式

(1) 模式启动

SPI 主设备控制 SPI 总线上所有数据传送的启动。当 SPIx_CR 寄存器中的 MSTR 位置 1 时，SPI 在主模式下运行，只有一个主设备可以启动传送。

(2) 发送

在 SPI 主模式下，写一个字节数据到 SPI 发送数据寄存器 SPIx_TDR，若发送缓存器为空，数据将会写入发送缓存器，若发送缓存器非空，数据将被放在发送数据寄存器内，当发送缓存器发送完毕，数据会从发送数据寄存器自动调入发送缓存器，并产生 SPTI 中断。如果发送数据寄存器为满时还向 SPIx_TDR 写数据，那么主 SPI 产生一个 WCOL 信号以表明写入太快。但是发送数据寄存器和发送缓存器中的数据不会受到影响，发送也不会中断。另外如果发送缓存器为非空，那么主设备立即按照 SCK 上的 SPI 时钟频率串行地移出发送缓存器中的数据到 MOSI 线上。当发送数据寄存器 SPIx_TDR 为空时，SPIx_FR 寄存器中的 SPTI 位被置 1。如果 SPTIE 中断被允许，当 SPTI 位置 1 时，会产生一个中断。SPTI@SPIx_FR 将在 SPIx_TDR 内被填充数据后自动清零。

(3) 接收

当主设备通过 MOSI 线传送数据给从设备时，相对应的从设备同时也通过 MISO 线将其发送缓存器的内容传送给主设备的接收缓存器，实现全双工操作。从设备接收的数据按照 MSB 或 LSB 优先的传送方向存入主设备的接收缓存器。当一个字节的数据完全被移入接收寄存器时，数据将被自动移入 SPIx_RDR，并产生 SPRI 中断。处理器可以通过读 SPIx_RDR 寄存器获得该数据。接收中断标志位 SPRI 在 SPIx_RDR 为满时被置 1，当 SPIx_RDR 内数据被读走为空后，SPRI 将自动清零。如果发生超限（接收到的数据未被及时取走，SPIx_RDR 与接收缓存器内都有数据，就试图开始下一次接收），RXOV 位置 1，表示发生数据超限，此时接收移位寄存器保持原有数据并且 SPRI 位置 1，这样直到 SPRI 位被清 0，SPI 主设备将不会接收任何数据。

从模式

(1) 模式启动

当SPIx_CR寄存器中的MSTR位清 0，SPI在从模式下运行。在数据传送之前，从设备的$\overline{SS}$引脚必须被置低，而且必须保持低电平直到一个字节数据传送完毕。

(2) 发送与接收

从属模式下，按照主设备控制的 SCK 信号，数据通过 MOSI 引脚移入，MISO 引脚移出。一个位计数器记录 SCK 的边沿数，当接收移位寄存器移入 8 位数据（一个字节），开始发送移位寄存器移出 8 位数据（一个字节）。数据可以通过读取 SPIx_RDR 寄存器获得。如果 SPRIE 中断被允许，当 SPRI 置 1 时，也会产生一个中断。

SPI 从设备不能启动数据传送，所以 SPI 从设备必须在主设备开始一次新的数据传送之前将要传送的数据写入发送寄存器 SPIx_TDR（当只写入一个数据时，直接写入发送缓存器，当写入两个数据时，分别存放在发送缓存器与发送数据寄存器 SPIx_TDR 内）。SPI 模块复位后，如果从设备在第一次开始发送之前未写入数据，从设备将传送“0x00”字节给主设备。如果写 SPIx_TDR 操作时，发送数据寄存器 SPIx_TDR 与发送缓存器内都有数据，那么 SPI 从设备的 WCOL 标志位置 1，即如果发送数据寄存器 SPIx_TDR 已经含有数据，SPI 从设备的 WCOL 位置 1，表示写 SPIx_TDR 冲突。但是发送数据寄存器与发送缓存器内的数据不受影响，传送也不会被中断。

22.2.3.5　传送形式

通过软件设置 SPIx_CR 寄存器的 CPOL 位和 CPHA 位，用户可以选择 SPI 时钟极性和相位的四种组合方式。CPOL 位定义时钟的极性，即空闲时的电平状态，它对 SPI 传输格式影响不大。CPHA 位定义时钟的相位，即定义允许数据采样移位的时钟边沿。在主从通讯的两个设备中，时钟极性相位的设置应一致。

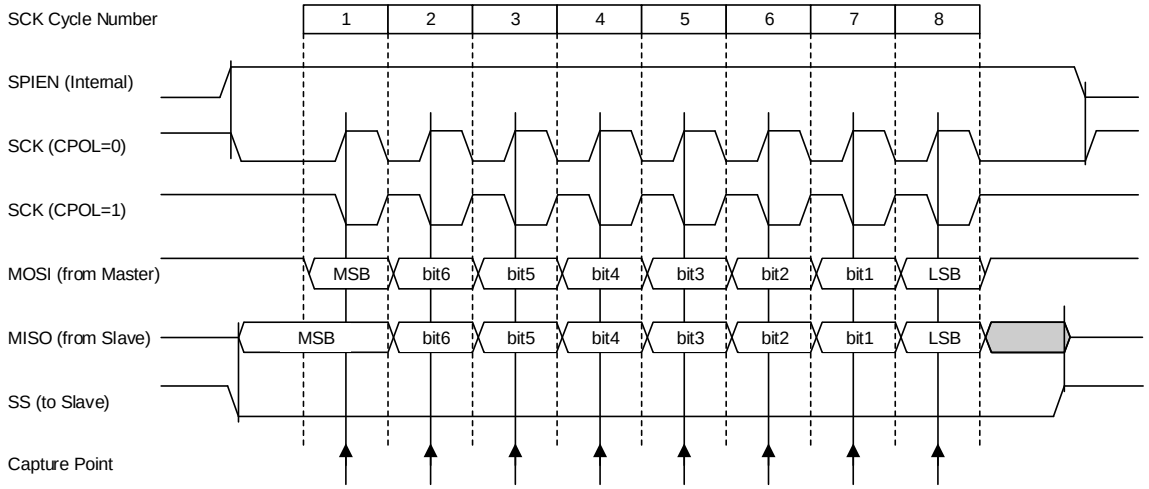


图22-18　数据传送形式（CPHA = 0）图

如果CPHA = 0，SCK的第一个沿捕获数据，从设备必须在SCK的第一个沿之前将数据准备好，因此，在从机模式下，$\overline{SS}$引脚的下降沿从设备开始发送数据。$\overline{SS}$引脚在每次传送完一个字节之后必须被拉高，在发送下一个字节之前重新设置为低电平，因此当CPHA = 0，SSDIS不起作用。另外, 针对该模式，SPI还添加了SPSFF@SPIx_CR控制位，置位该控制位后，第一byte数据还需要$\overline{SS}$引脚的下降沿载入，从第二byte数据开始将使用前一byte最后一个CLK沿载入数据，以达到连续发送的目的，同时，使能该功能后还可以使用DMA方式来传输数据。

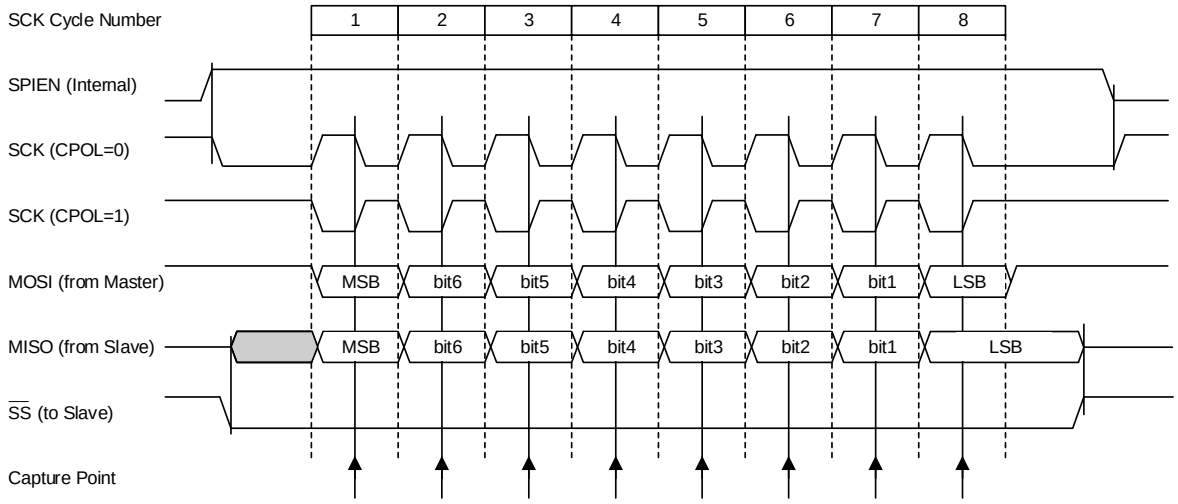


图22-19　数据发送形式（CPHA = 1）图

如果CPHA = 1，主设备在SCK的第一个沿将数据输出到MOSI线上，从设备把SCK的第一个沿作为开始发送信号。用户必须在第一个SCK的第二个沿之前完成写SPIx_TDR的操作。$\overline{SS}$引脚在每个字节数据的传送过程始终保持低电平。这种数据传输形式是一个主设备一个从设备之间通信的首选形式。

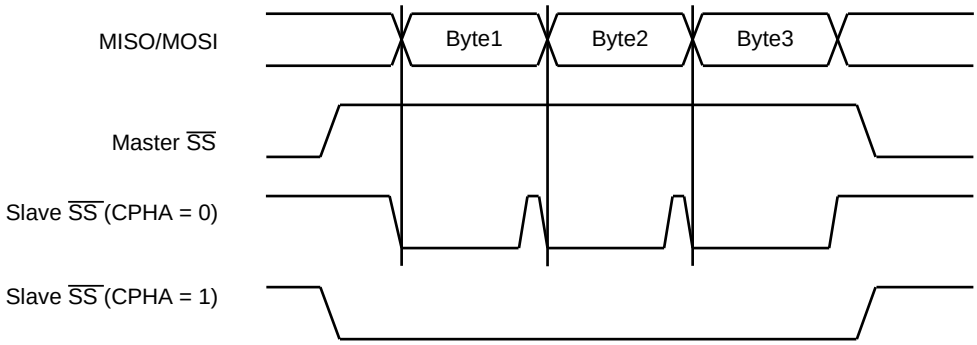


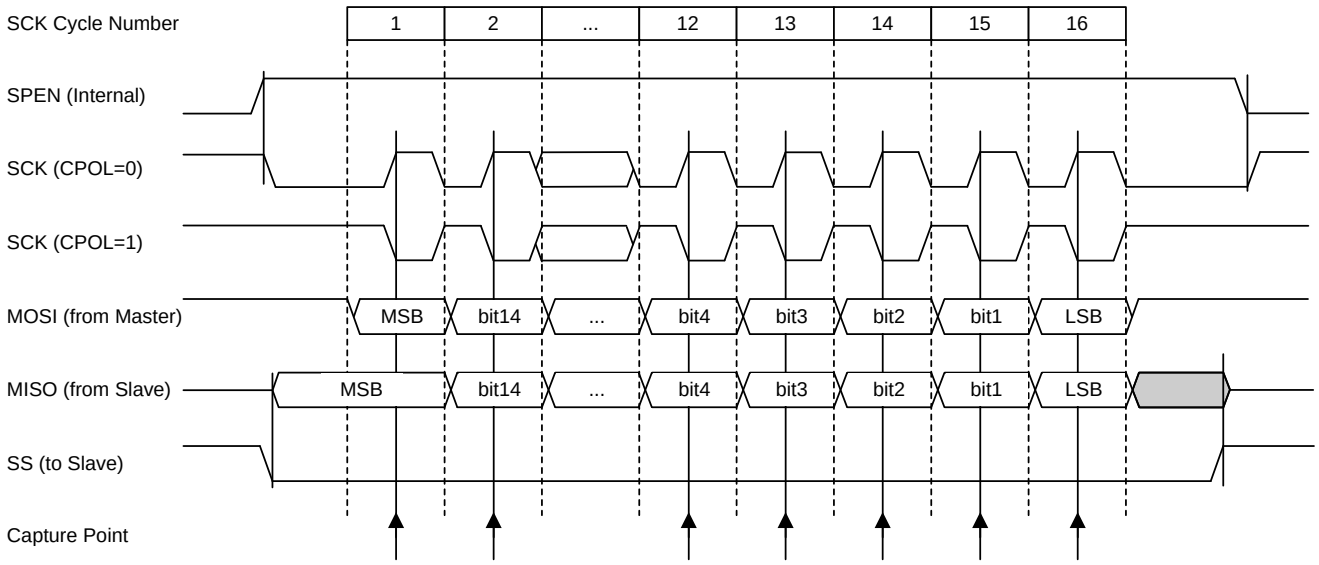
图22-20　　CPHA/NSS 时序图

注意： *(1) 当 SPI 用作从设备模式，且 SPIx_CR 寄存器的 CPOL 位清 0，SCK 端口必须设置为输入模式，并在数据传输使能位置 1 前打开上拉电阻；*

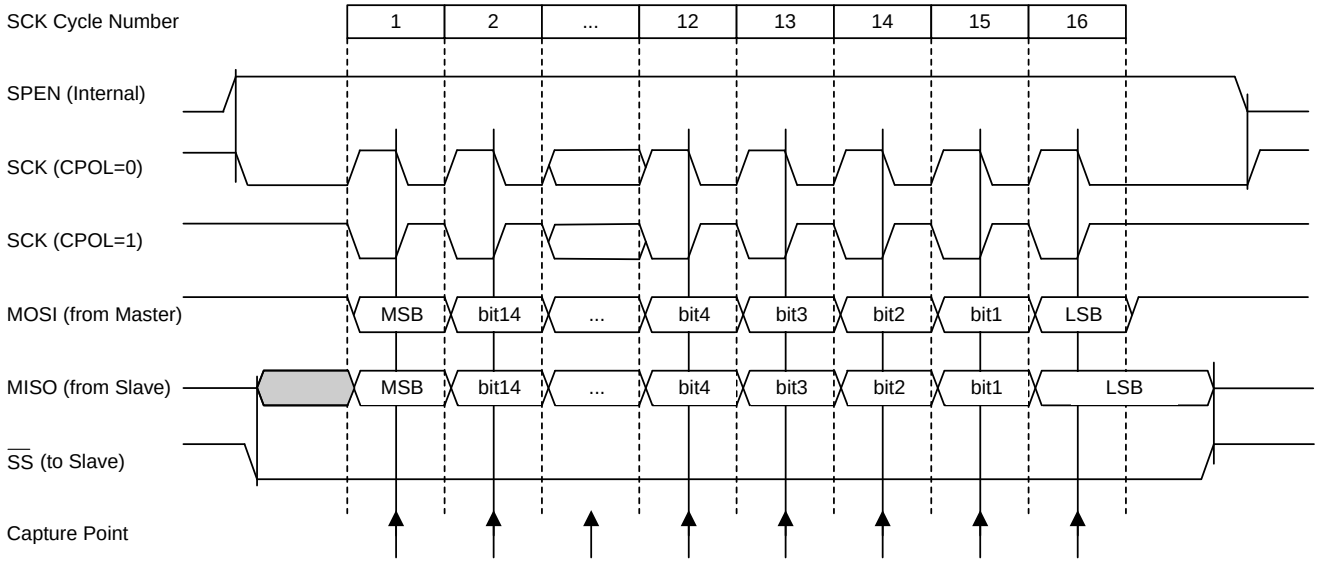
(2) 当 SPI 用作主设备模式，如果 SCK 引脚设置为开漏，SCK 时钟的上升沿要小于 3 个 PCLK 时钟。

22.2.3.6　　数据字 8、16 位可设置

单次传输的数据字长度可通过寄存器 SPIx_CR 的 SPDATL 位配置为 8 位或 16 位。采用 16 位字长时，移位寄存器与缓存寄存器均被扩展为 16 位模式，对 SPIx_TDR 与 SPIx_RDR 的读写均为 16 位操作。（注：TDR、RDR 的真实硬件为 16 位，8 位模式时，忽略高 8 位即可），时序图如下：



数据传送形式（CPHA = 0）



数据发送形式（CPHA = 1）

22.2.3.7　利用 DMA 通信

为了达到最大通信速度，需要及时往 SPI 发送缓冲器填数据，同样接收缓冲器中的数据也必须及时读走以防止溢出。为了方便高速率的数据传输，SPI 采用了一种采用简单的请求/应答的 DMA 机制。

当 SPIx_CR 寄存器上的对应使能位 SPDMAT、SPDMAR@SPIx_CR 被设置时，SPI 模块可以发出 DMA 传输请求。发送缓冲器和接收缓冲器有各自的 DMA 请求。选择 DMA 模式后，建议将 SPI 的相应中断使能位关闭，因为 DMA 模式下 DMA 请求无需通过 CPU 参与便可以自行处理数据流传输。

SPI 发送

发送时，在每次 SPTI 被置 1 时发出 DMA 请求，DMA 控制器则写数据至 SPIx_TDR 寄存器，SPTI 标志因此而被清 0。

SPI 接收

接收时，在每次 SPRI 被置 1 时发出 DMA 请求，DMA 控制器则从 SPIx_RDR 寄存器读出数据，SPRI 标志因此而被清 0。

当只使用 SPI 发送数据时，只需使能 SPI 的发送 DMA 通道。此时，若因为没有读取收到的数据，RXOV 被置为'1'(软件不必理会这个标志)。

22.2.3.8　SPSTA 状态异常情况

SPIx_FR 寄存器中的 MODF,WCOL,RXOV 标志位表示在 SPI 通讯中的出错情况：

(1) 模式故障（MODF)

SPI主模式下的模式故障出错表明$\overline{SS}$引脚上的电平状态与实际的设备模式不一致。SPIx_FR寄存器中MODF位置 1 后，表明系统控制存在多主设备冲突的问题。这种情况下，SPI系统受到如下影响：

产生 SPI 接收/错误 CPU 中断请求，在中断中需要软件完成以下操作；

SPI 被禁止，要重新使能 SPI 需要对 SPIx_CR 寄存器的 SPIEN 位写 1 操作；

SPIx_CR 寄存器的 MSTR 位清 0。

当SPIx_CR寄存器的$\overline{SS}$引脚禁止位（SSDIS）清 0，$\overline{SS}$引脚信号为低时，MODF标志位置 1。然而，对于只有一个主设备的系统来说，主设备的$\overline{SS}$引脚被拉低，那决不是另外一个主设备试图驱动网络。这种情况下，为防止MODF置 1，可使SPIx_CR寄存器中的SSDIS位置 1，$\overline{SS}$引脚作为普通I/O口或是其它功能引脚。

重新启动串行通信时，用户必须将 MODF 位软件清 0，将 SPIx_CR 寄存器中的 MSTR 位和 SPIEN 位置 1，重新启动主模式。

(2) 写冲突（WCOL）

在发送数据序列期间写入 SPIx_TDR 寄存器会引起的写冲突，SPIx_FR 寄存器中的 WCOL 位置 1。WCOL 位置 1 不会引起中断，发送也不会中止。WCOL 位需由软件清 0。

(3) 超限情况（RXOV）

主设备或从设备尚未清除 SPRI 位，主或从设备又试图发送几个数据字节时，超限情况发生。在这种情况下，接收移位寄存器保持原有数据，SPRI 置 1，同样 SPI 设备直到 SPRI 被清除后才会再接收数据。在 SPRI 位被清除之前继续调用中断，发送也不会中止。RXOV 位置 1 不会引起中断，RXOV 位需由软件清 0。

22.2.3.9　中断

三种 SPI 状态标志 SPTI、SPRI & MODF 能产生一个 CPU 中断请求。

串行外围设备数据发送接收标志：

SPI 的中断（SPTI、SPRI）在 SPIx_TDR 和 SPIx_RDR 缓存完成一字节数据传输后（发送时为发送寄存器为空，接收时为接收寄存器为满）产生；

使能 DMA 通道时，SPI 的 DMA 请求信号在完成一字节传输后（发送时为发送寄存器为空，接收时为接收寄存器为满）产生，直接传送至 DMA 控制器，请求信号由 DMA 控制器负责清 0；

模式故障标志，MODF：该位被置 1 表示$\overline{SS}$引脚上的电平与SPI模式不一致的。SSDIS位为 0 并且MODF置 1 将产生SPI接收器/出错CPU中断请求。当SSDIS置 1 时，无MODF中断请求产生。

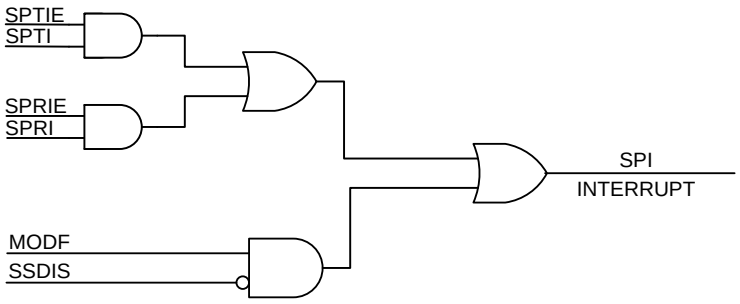


图22-21　　SPI 中断控制图

22.2.4　SPI 寄存器

SPI0-1 模块寄存器列表 (基地址: 0x4002 1800)

地址	寄存器名	说明
0x4002 1800 + 0x400*x	FR	SPIx中断标志寄存器 (SPI0～1对应x:0~1)
0x4002 1804 + 0x400*x	TDR	SPIx发送数据寄存器 (SPI0～1对应x:0~1)
0x4002 1808 + 0x400*x	RDR	SPIx接收数据寄存器 (SPI0～1对应x:0~1)
0x4002 180C + 0x400*x	CR	SPIx控制寄存器 (SPI0～1对应x:0~1)

22.2.4.1 SPI 中断标志寄存器 (SPIx_FR)

偏移地址: 0x0000 + 0x400*x

复位值: 0x0000 0002

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留										WCOLC	RXOVC	MODFC	保留	SPTIC	SPRIC
-										WO	WO	WO	-	WO	WO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留										WCOL	RXOV	MODF	BUSY	SPTI	SPRI
-										RO	RO	RO	RO	RO	RO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

位	符号	说明
31 ~ 22	保留	-
21	**WCOLC**	**写入冲突标志清除位** 0：无效 1：清除
20	**RXOVC**	**接收溢出标志清除位** 0：无效 1：清除
19	**MODFC**	**模式故障标志清除位** 0：无效 1：清除
18	保留	-
17	**SPTIC**	**SPI一字节数据传送中断标志清除位** 0：无效 1：清除
16	**SPRIC**	**SPI一字节数据传送中断标志清除位** 0：无效 1：清除
15 ~ 6	保留	-
5	**WCOL**	**写入冲突标志位** 0：已处理或没有写入冲突，由软件对WCOLC位写1清零 1：检测到写入冲突，由硬件置位
4	**RXOV**	**接收溢出标志位** 0：已处理或无接收溢出，由软件对RXOVC位写1清零 1：发生接收溢出，由硬件置位
3	**MODF**	**模式故障标志位** 0：无故障，由软件对MODFC位写1清零 1：NSS引脚电平与SPI模式不一致，由硬件置位
2	**BUSY**	**主机BUSY状态标志位** 0：主机处于空闲状态 1：主机处于发送状态，BUSY
1	**SPTI**	**SPI一字节数据传送中断标志位** 0：发送数据寄存器TDR为满的状态，由软件对SPTIC位写1清零 1：发送数据寄存器TDR为空的状态，由硬件置位
0	**SPRI**	**SPI一字节数据接收中断标志位** 0：接收数据寄存器RDR为空的状态，由软件对SPRIC位写1清零 1：接收数据寄存器RDR为满的状态，由硬件置位

22.2.4.2 SPI 发送数据寄存器 (SPIx_TDR)

偏移地址: 0x0004 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TDR[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TDR[31:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	**TDR[31:0]**	**发送缓存寄存器** 当SPDATW=0，且SPDATL=0，为8位SPI发送缓存寄存器 当SPDATW=0，且SPDATL=1，为16位SPI发送缓存寄存器 当SPDATW=1，且SPDATL=0/1，为32位SPI发送缓存寄存器

22.2.4.3 SPI 接收数据寄存器 (SPIx_RDR)

偏移地址: 0x0008 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RDR[31:0]															
RO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RDR[31:0]															
RO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 0	**RDR[31:0]**	**接收缓存寄存器** 当SPDATW=0，且SPDATL=0，为8位SPI接收缓存寄存器 当SPDATW=0，且SPDATL=1，为16位SPI接收缓存寄存器 当SPDATW=1，且SPDATL=0/1，为32位SPI接收缓存寄存器

22.2.4.4 SPI 控制寄存器 (SPIx_CR)

偏移地址: 0x000C + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															SPDATW
-															RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
SPSFF	SPIEN	SPDMAT	SPDMAR	SPTIE	SPRIE	SPDATL	DIR	MSTR	CPHA	CPOL	SSDIS	SPR[3:0]			
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 17	保留	-
16	**SPDATW**	**缓存寄存器长度选择位** 0:TDR/RDR寄存器长度为8或16位，具体由SPDATL选择

		1:TDR/RDR寄存器长度为32位
15	**SPSFF**	**从机快速发送模式控制位（CPHA=0时有效）** 0:普通模式，从机的SS引脚在每次传送完一个数据（8位或16位）后必须被拉高，发送下一个数据前必须从新拉低SS以载入数据到移位寄存器 1:快速模式，从机发送的第一个数据需要用SS的下降沿来触发，第二个数据则利用上个数据最后一个CLK来加载到移位寄存器（注，若发送缓存内没有数据则不会加载）
14	**SPIEN**	**SPI使能位** 0：关闭SPI 1：使能SPI
13	**SPDMAT**	**SPI数据发送DMA通道使能** 0：不使用DMA来传输SPI的发送数据 1：使用DMA来传输SPI的发送数据
12	**SPDMAR**	**SPI数据接收DMA通道使能** 0：不使用DMA来传输SPI的接收数据 1：使用DMA来传输SPI的接收数据
11	**SPTIE**	**SPI的传送中断标志使能位** 0：禁止传送中断标志位SPTI触发中断 1：允许传送中断标志位SPTI触发中断
10	**SPRIE**	**SPI的接收中断标志使能位** 0：禁止接收中断标志位SPRI触发中断 1：允许接收中断标志位SPRI触发中断
9	**SPDATL**	**数据字长度选择位** 0：每次发送8bit数据 1：每次发送16bit数据
8	**DIR**	**数据传送方向选择位** 0：MSB优先发送 1：LSB优先发送
7	**MSTR**	**SPI主从设备配置位** 0：配置SPI为从设备 1：配置SPI为主设备
6	**CPHA**	**时钟相位控制位** 0：CK周期的第一沿采集数据 1：CK周期的第二沿采集数据
5	**CPOL**	**时钟极性控制位** 0：在IDLE状态下CK处于低电平 1：在IDLE状态下CK处于高电平
4	**SSDIS**	**NSS引脚控制位** 0：在主/从模式下，打开NSS引脚 1：在主/从模式下，关闭NSS引脚，将不会产生MODF中断请求 在从模式下，如果CPHA=0，该位不起作用
3 ~ 0	**SPR[3:0]**	**时钟分频控制位** 0000：PCLK1 /2 0001：PCLK1 /4 0010：PCLK1 /8 0011：PCLK1 /16 0100：PCLK1 /32 0101：PCLK1 /64 0110：PCLK1 /128 0111：PCLK1 /256 1000：PCLK1 /512 1001：PCLK1 /1024 其他：保留

22.3 两线串行接口（TWI）

22.3.1 简介

TWI（Two-wire Serial Interface）接口是对 I²C 总线接口的继承和发展，完全兼容 I²C 总线，具有硬件实现简单、软件设计方便、运行可靠和成本低廉的优点。TWI 由一根时钟线和一根数据线组成，以字节为单位进行传输，可兼容 SMBus 总线规范，自动对字节传输进行处理，并对串行通信进行跟踪。

SCL/SDA 是 TWI 总线的信号线。 SDA 是双向数据线，SCL 是时钟线。在 TWI 总线上传送数据，首先送最高位(MSB)，由主机发出起始信号，然后由主机发送一个或多个字节的数据，数据传送完毕，由主机发出停止信号，完成一次最简单的 TWI 传输。

典型 TWI 总线应用如下图所示，最高支持 128 个不同的器件进行通讯。

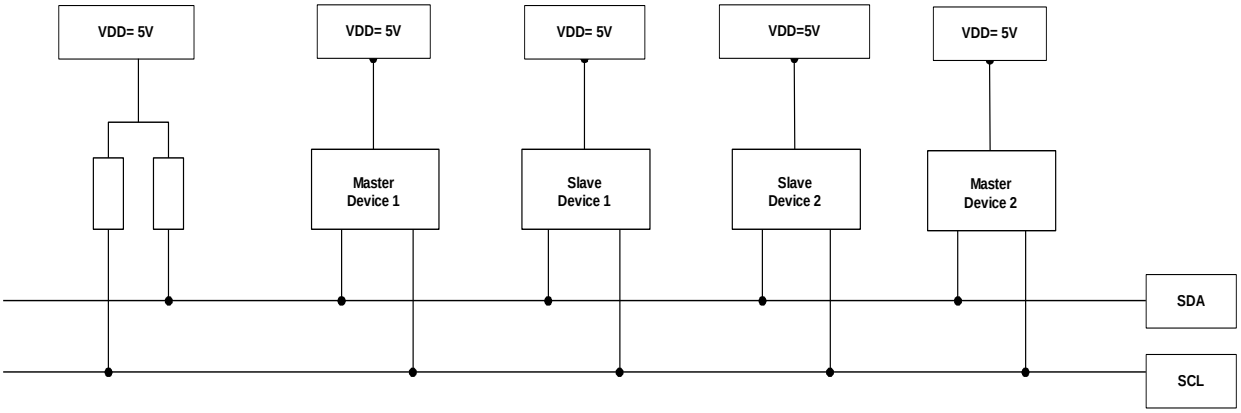


图22-22　　典型 TWI 总线应用框图

22.3.2 主要特性

- 两线模式
- OD 输出(需在 GPIO 中配置)，通讯电平不受 VDD 影响，也不会影响到 VDD
- 支持主机模式(Master)和从机模式(Slave)
- 允许发送(Transmitter)和接收(Receiver)
- 支持多主机通讯的仲裁功能
- 具有低电平总线超时判断(Timeout)
- 在睡眠模式下可唤醒系统，不支持停机模式唤醒
- 地址可编程，带多个地址屏蔽位，支持广播功能
- 支持标准模式和快速模式

22.3.3 功能描述

22.3.3.1　总线信号时序

总线初始状态为总线空闲，即 SDA 和 SCL 两条信号线都处于高电平，总线上所有的器件都释放总线，两条信号线各自的上拉电阻把电平拉高。

数据传输中数据线上每一位的传输均需要时钟线上一个脉冲。在时钟高电平时数据线应保持稳定，在时钟低电平时允许数据线改变。但起始条件和停止条件是特殊的，前者指时钟高电平期间的数据线的下降沿，后者指时钟高电平期间的数据线上升沿。

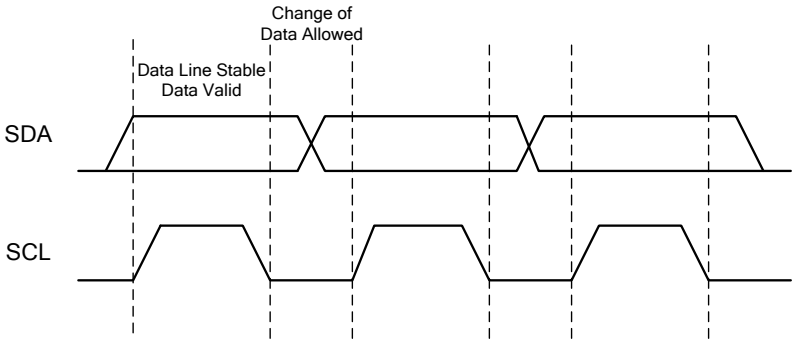


图22-23　　TWI 总线的基本位传输特性

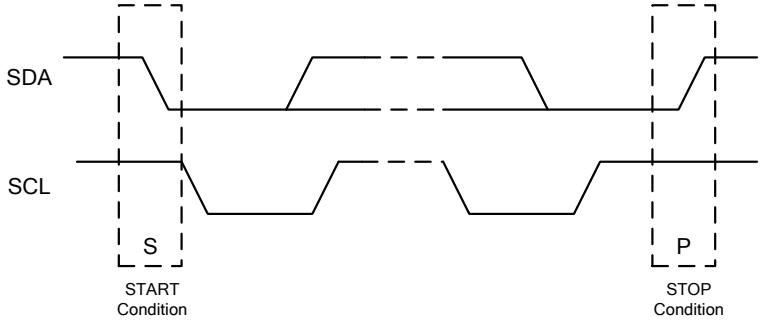


图22-24　　TWI 总线的起始条件（起始信号）和停止条件（停止信号）

在 TWI 总线中，起始条件和停止条件均由主机发出。

主机可以发起和终结一次传输。当主机发送一个起始条件时开始一次传输， 发送一个停止条件时结束本次传输。在起始条件和停止条件之间，总线定义为“忙碌”状态。其它主机不应该去试图发起传输。在“忙碌”状态下，如果主机再次发送起始条件，则定义为“重复起始条件”，表示主机希望不放弃总线的情况下开始一次新的传输。发送重复起始条件后，总线仍处于“忙碌”状态，一直到总线出现停止条件。鉴于重复起始条件和起始条件性质完全一致，除非特别声明，本文中将采用起始条件来代替两者。

所有数据包（包括地址包）均有 9 位组成，包括 1 个字节和一个应答位。主机负责发出时钟，起始条件和停止条件，接收者负责给出应答信号。接收者通过在第九个时钟脉冲处将数据线拉低发出“应答(ACK)”信号；或维持第九个脉冲处维持高电平表示“不应答(NACK)”信号。当接收方接收到最后一个字节，或因某种原因无法继续接收数据时，应回应“不应答(NACK)”信号。

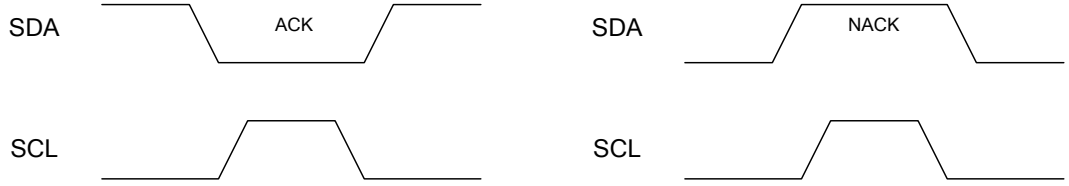


图22-25　　ACK 和 NACK 信号

由于某种原因从机不对主机寻址信号应答时（如从机正在进行实时性处理工作而无法响应总线通信），它必需返回一个 NACK，再由主机产生一个停止信号以结束总线的数据传送。

如果从机对主机进行了应答，但在数据传送一段时间后无法继续接收更多数据时，可以通过返回一个 NACK 通知主机，主机则应发出停止信号以结束数据的继续传送。

当主机接收数据时收到最后一个数据字节后，必须向从机发送一个 NACK 通知从机停止数据发送，然后从机释放 SDA 线，以允许主机产生一个停止信号。

注：TWI 是双向数据通信，一节点要通过 SDA 发送数据时要求其它节点释放 SDA 线，即 SDA 线处于外部上拉状态。

在 TWI 总线中，采用从高到低逐位进行传输。

一次传输通常包括一个起始条件，地址+读/写，一个或多个数据包和一个停止条件。**仅包含起始条件和停止条件的数据格式是不合通讯规则的。**值得注意的是“线与”结构给主机和从机之间的握手信号提供了方便。**当主机相对太快或从机需要处理其它事务时，从机可以通过拉低时钟线来拉长时钟线的低电平时间，从而降低通讯频率**。从机可以拉长时钟线的低电平周期但不会影响到时钟线高电平的周期。

在产生应答信号时，SH32F9001_32F9801 拉低 SDA 信号线。中断标志位置起期间，SH32F9001_32F9801 拉低 SCL 信号线，释放 SDA 信号线。中断处理完毕后清除 TWINT 标志，释放 SCL 线。

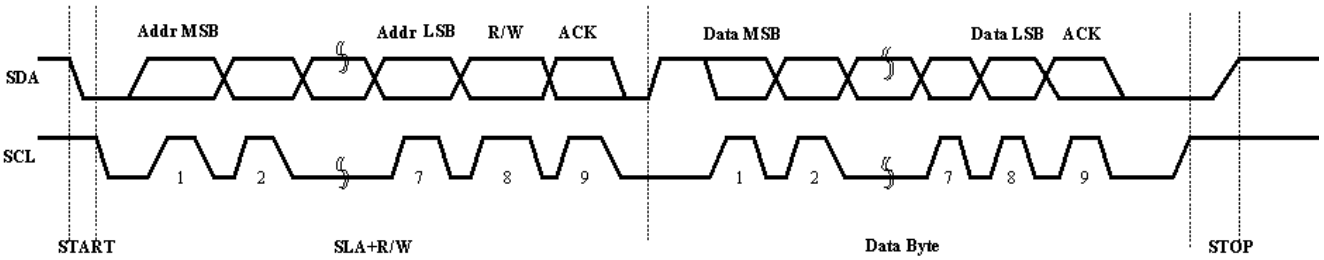


图22-26　一次典型的 TWI 数据传输时序图

主控器完成一次通信后还想继续占用总线进行通信，不释放总线，就要利用重复起始条件 Rs。它既作为前一次数据传输的结束，又作为后一次传输的开始。

特殊情况下，若需禁止所有发生在 TWI 总线上的通信，可以封锁或关闭总线，具体操作为总线上任一器件将 SCL 锁定在低电平即可。

22.3.3.2　时钟同步

当多个主机同时希望控制总线时，总线将依据“线与”原则决定时钟线高低电平。对于所有参与传输的主机来说，定义清楚每一个时钟脉冲的起始是相当重要的。

时钟线电平的由高到低跳变将导致所有参与传输的器件开始低电平计时。每一个器件计时达到自己低电平要求时释放时钟线，在时钟线变为高电平之前进入高电平等待期；当所有器件均计满低电平周期时，时钟线才变为高电平。之后所有器件开始对高电平进行计时，第一个计满高电平周期的器件将拉低时钟线，进入下一个时钟周期。

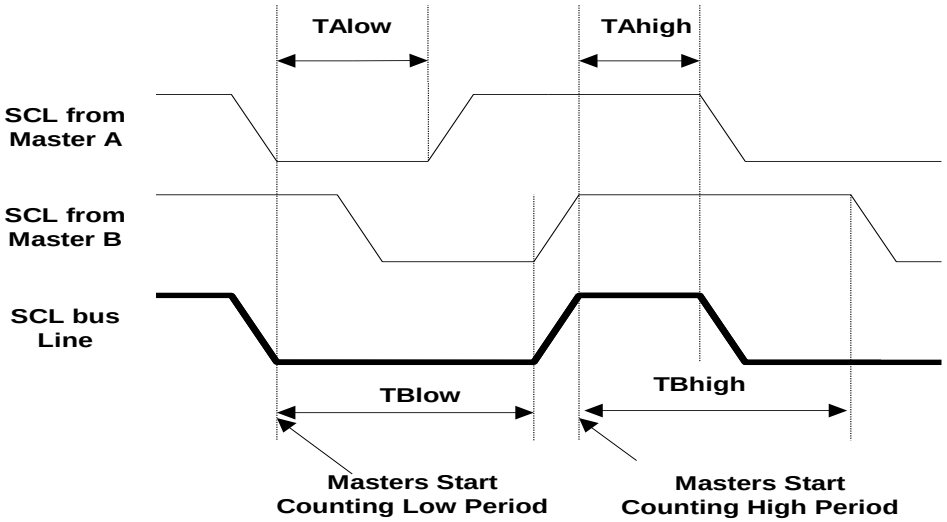


图22-27　时钟同步时序图

注：当多个主机同时控制 SCL 线时，“线与”会导致 SCL 线的低电平周期由低电平时间最长的主机决定，而高电平周期由高电平时间最短的主机决定。

22.3.3.3 总线仲裁

主机只有在总线处于“空闲”状态时才能开始一次传输。两个或多个主机可能在最小持续时间($t_{HOLD:STA}$))内同时发送起始条件，结果在总线上会得到一个正常的起始条件，多个主机将开始发送，后续的传输需要进行仲裁。

由于发送起始条件的主机无法知道是否有其它主机在竞争总线，只能靠 SCL 高电平期间对 SDA 线的电平检测来判断，即仲裁逻辑会检查每个 SDA 发送的逻辑 1 是否真正出现在总线上，如果有其它主机传输低电平并把 SDA 线拉低，则传输高电平的主机将失去仲裁，必须放弃总线，这就是基本的仲裁逻辑。

注：这是一种“低电平优先”的仲裁原则，即把总线判给在 SDA 上先发送低电平的主器件。

失去仲裁的主机将从发送状态转为接收状态，并且继续发送时钟脉冲（SCL 上），直到当前传输字节发送完毕。

注：除了主机发送模式，在主机接收模式中，当主机向从机返回一个 NACK 信号（逻辑 1）时，也有可能发生仲裁丢失，由于 NACK 在串行字节结束时出现，因此模块不会再发送时钟脉冲。

仲裁是按 bit 逐位进行的，它的第一个阶段是比较地址位，如果多个主机寻址相同的器件，则可能会顺利通过地址阶段仲裁，则仲裁将会继续比较数据位（主机发送模式），或者响应位（主机接收模式）。因为总线上的地址和数据信息是由赢得仲裁的主机决定，仲裁过程中不会丢失信息。

如果该主机同时开启了从机模式，在发送地址阶段失去仲裁后应检测线上的地址是否与自己相匹配；如果是对自己的访问，应立即切换到从机模式，接收信息。

每次传输中，仍要检测线上的“重复起始条件”，当检测到并非自己发出的“重复起始条件”时，应立即退出当前传输。

当一个主机发送重复起始条件或停止条件时，另一个主机可能仍然在发送数据，这会造成一个未定的状态（规范要求在帧格式相同位置重发这个重复起始条件或停止条件），为此定义如下 3 种情况下总线不进行仲裁：

1. 主机 1 发送重复起始条件，主机 2 发送数据
2. 主机 1 发送停止条件，主机 2 发送数据
3. 主机 1 发送重复起始条件，主机 2 发送停止条件

另外，仲裁只发生在主机之间，从机不参与仲裁。

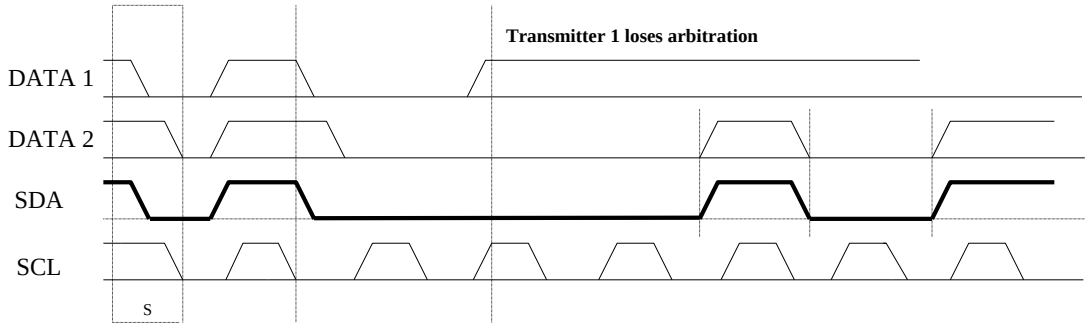


图22-28 数据仲裁时序图

22.3.3.4 TWI 通讯模块的详细结构

下图描述了 TWI 通讯模块的详细结构。

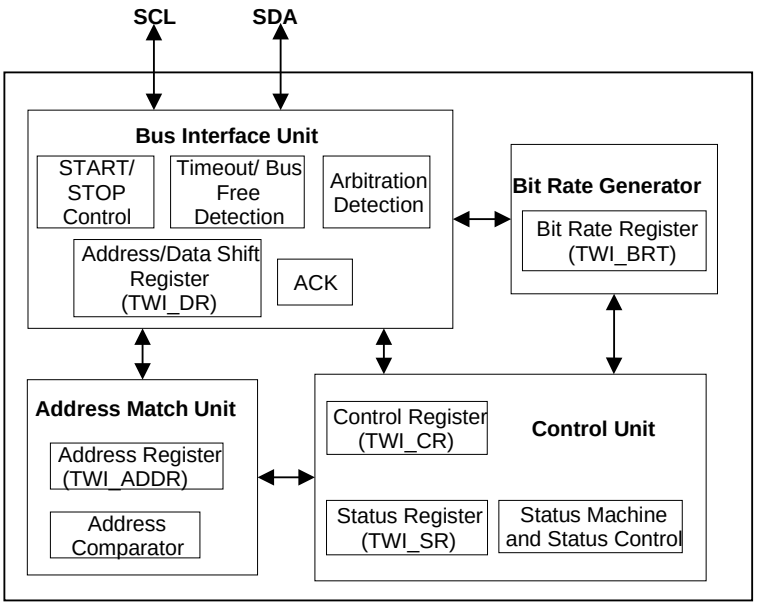


图22-29 TWI 模块结构图

总线接口单元

总线接口单元包括数据和地址移位寄存器(TWI_DR)，开始/停止条件控制器，仲裁和总线超时检测单元。

寄存器 TWI_DR 存储了即将发送的数据或地址和接收到的数据和地址。

开始/停止条件控制器负责发送和检测总线上的起始条件，重复起始条件和停止条件。

如果 SH32F9001_32F9801 已经作为主机开始一次传输，仲裁单元将始终检测是否有仲裁发生。当失去仲裁时，控制单元可以进行合适的动作，并产生相应的状态码。

SH32F9001_32F9801 在传输数据/地址时，必须在 SCL 由低跳高前维持数据稳定。

SH32F9001_32F9801 在传输 ACK/NACK 时，在 SCL 由低跳高后产生 TWINT 中断，并在 SCL 由高跳低时拉低 SCL，在 TWINT 中断清零时释放 SCL。

SH32F9001_32F9801 在传输 ACK/NACK 信号时，若 TWINT 已被清零，SCL 仍为高电平时，SDA 产生跳变，则重新产生 TWINT 中断，状态为 00H。SH32F9001_32F9801 当前通讯终止，该状态与普通 00H 状态处理一致。

SH32F9001_32F9801 在传输 ACK/NACK 信号时，若 TWINT 未被清零，SCL 仍为高电平时，SDA 产生跳变，则状态直接切换到 00H，不会再次产生中断。SH32F9001_32F9801 作为从机进入该状态，则当前通讯终止，可发生 STA 开始主机传输，或重新接受 STA+ADR 对自己地址的访问。SH32F9001_32F9801 作为主机进入该状态，则当前通讯终止，可发生 STA 开始主机传输，或重新接受 STA+ADR 对自己的访问。

SH32F9001_32F9801 在当前通讯终止后，不会再参与当前传输。SH32F9001_32F9801 若作为主机存在，请开启 EFREE 功能，防止进入逻辑死区。

SH32F9001_32F9801 规定总线维持高电平超过 TFREE =Tsys*TWI_HOC*1024（必须保证 TFREE 大于 $t_{SCL}/2$（t_{SCL} 为时钟线的周期））所定义的 PCLK1 时钟个数时为“空闲”状态，释放总线（此处“PCLK1 时钟”Tsys 实际为 TWI 模块工作时钟）。该功能仅适用于一个数据包传输过程中（8+1 个位）。SH32F9001_32F9801 处于从机发送模式，且所传输的第一个字节为低电平时适用该功能。起始条件（STA、RSTA）不适用于该功能。SH32F9001_32F9801 产生中断，寄存器 TWI_CR 中的 TFREE 会被置位(如果控制位 EFREE 已置位)。

如果时钟线 SCL 被从机拉低时，通讯会暂时中止；而主机也没有办法将时钟线拉高。为解决此问题，TWI 协议规定参与传输的所有器件，将时钟线维持低电平超过 N*Tsys（N 值由 CNT 寄存器决定）所定义的时钟个数时为“总线超时”，寄存器 TWI_FR 中的 TOUT 会被置位(如果控制位 ETOT 已置位)。软件通过此标记位的变化，复位 TWI 模块，硬件会自动释放总线。

频率生成单元

在主机模式下，可通过寄存器 TWI_CR 的 CR[1:0]分频系数及 TWI_BRT[7:0]寄存器来设定通讯频率。

SCL 频率为 f_{TWI} /（16+2*CR*TWI_BRT）。

注：此处 f_{TWI} 为 TWI 模块工作时钟，即 APB1 总线时钟。

地址匹配单元

地址匹配单元检验所收到的地址是否与 TWIADR 中的 7 位地址相匹配，如果地址屏蔽位 TWIAMR 内对应位置 1，则该位地址不检测。如果通用地址使能位 GC 被置位，还将检测是否与通用地址 00H 相匹配。当地址匹配时，控制单元将产生合适的动作及相应的状态码。

控制单元

控制单元监视 TWI 总线，并依据控制寄存器 TWI_CR 的设置进行相应的回应。当 TWI 总线有需要应用层注意的事件时，TWI 中断标志被置起，标明当前事件的状态码会被写入状态寄存器 TWI_STAT。状态寄存器 TWI_STAT 只表示 TWI 通讯中断产生时的通讯状态信息；其它情况下状态寄存器内是一个用于表示没有有效状态码的状态码。在中断清除之前，时钟线将维持低电平。应用软件可在处理完任务后才允许 TWI 通讯继续。

22.3.3.5 TWI 操作模式

在指定应用中，TWI 模块可用作主机(master)、从机(slave)或同时用作主机和从机。在从机模式中，TWI 硬件查找从机地址以及通用地址（GC=1）。如果检测到其中一个地址，则产生中断请求。如果处理器想成为总线主机，那么在进入主机模式前硬件必须等到总线空闲，使得从机操作不被中止。如果在主机模式中总线仲裁丢失，则 TWI 模块进入等待状态（纯主机），或者立刻切换到从机模式并在相同的串行传输中检测自身的从机地址（主从机）。

TWI 的四种基本操作模式：

- 主机发送模式
- 主机接收模式
- 从机接收模式
- 从机发送模式

有关这四种模式下文会有详细介绍。

SH32F9001_32F9801 的 TWI 通讯是以底层驱动电路和以中断为基础的应用软件共同完成的。诸如接收到一个字节或发送一个起始条件的所有总线事件均会产生一个中断。所以在字节传输期间，应用软件可以进行其它的操作。需注意的是，除了基本的 TWINT 中断外，有时还需要开启总线超时和 SCL 高电平超时检测，这两种情况也会产生中断，三个中断源共享一个中断入口，具体见“中断”章节。

当 TWINT 位置起时，表示一次 TWI 传输已完成，等待应用软件的回应，此时状态寄存器 TWI_STAT 包含了当前的状态。应用软件可通过寄存器 TWI_CR 和 TWI_STAT 决定 TWI 进行哪种通讯。

下面将分别介绍 TWI 通讯的四种主要模式，并对所有可能的状态码进行了描述。下图中有如下缩写：

S　　：起始条件
Rs　　：重复起始条件
R　　：读控制位
W　　：写控制位
A　　：应答位
Ā　　：无应答位
DATA：8 位数据
P　　：停止条件
SLA　：从机地址

图中圆形用于表示中断标志已被置起。其中的数字表示当前状态寄存器 TWI_STAT 中的状态码。

在 TWINT 被清除之前，TWI 通讯会暂停，应用软件必须决定是继续通讯还是终止当前传输。对每一个状态码，所需要的软件动作和随后的传输细节均有描述。

- **主机发送模式**

主机发送模式中，主机发送一系列数据到从机。为进入主机发送模式，一个起始条件，随后一个从机地址+写控制字(SLA+W)地址包表示进入主机发送模式(MT)。

通过设置控制寄存器 TWI_CR 中的 TWIEN 和 STA，清除 STO 和 TWINT，TWI 逻辑将检测 TWI 总线并在允许时发出一个起始条件(STA)。当起始条件(STA)传输完毕，通讯中断(TWINT)被置起，状态寄存器(TWI_STAT)为 08H，中断服务程序应将从机地址和写控制字(SLA+W)写入数据寄存器 TWI_DR。在开启下一个传输前清除 TWINT 标志。

当从机地址和写控制字传输完毕并收到一个“应答”信息时，中断(TWINT)被置起，状态寄存器 TWI_STAT 中有几个可能的状态：对主机模式有 18H，20H 和 38H，对从机模式有 68H，78H 和 B0H。

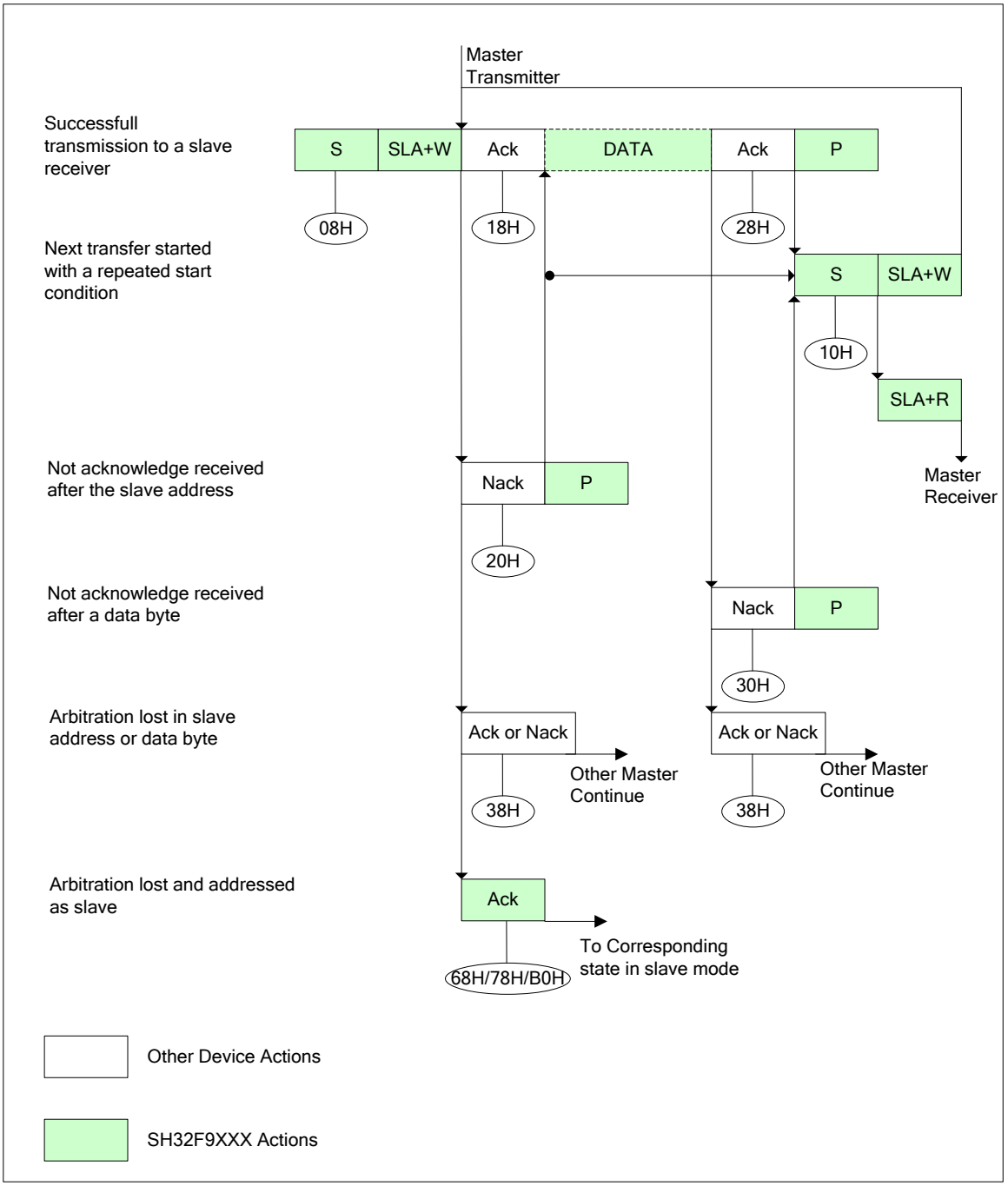


图22-30　　主机发送模式状态图

注：上图中设定SH32F9001_32F9801是主机。

表22-3　　主机发送模式状态码

状态码	TWI总线和硬件接口状态	应用软件响应					TWI执行的下一个动作
		读/写数据寄存器TWI_DR操作	控制位操作				
			STA	STO	TWINT	AA	
08H	已发送起始条件	写入SLA+W	X	0	0	X	发送SLA+W，接收ACK 或NACK
10H	已发送重复起始条件	写入SLA +W	X	0	0	X	发送SLA+W，接收ACK或NACK
		写入SLA +R	X	0	0	X	发送SLA+R，TWI将切换到主机接收模式
18H	已发送SLA +W；已接收ACK	写入数据字节	0	0	0	X	发送数据，接收ACK或NACK
		无TWI_DR动作	1	0	0	X	发送重复起始条件
			0	1	0	X	发送停止条件；清除STO标志
			1	1	0	X	发送停止条件，之后发送起始条件；STO被清除
20H	已发送SLA +W；已接收NACK	写入数据字节	0	0	0	X	发送数据，接收ACK或NACK
		无TWI_DR动作	1	0	0	X	发送重复起始条件
			0	1	0	X	发送停止条件；清除STO标志
			1	1	0	X	发送停止条件，之后发送起始条件；STO被清除
28H	已发送TWI_DR中数据；已接收ACK	写入数据字节	0	0	0	X	发送数据，接收ACK或NACK
		无TWI_DR动作	1	0	0	X	发送重复起始条件
			0	1	0	X	发送停止条件；清除STO标志
			1	1	0	X	发送停止条件，之后发送起始条件；STO被清除
30H	已发送TWI_DR中数据；已接收NACK	写入数据字节	0	0	0	X	发送数据，接收ACK或NACK
		无TWI_DR动作	1	0	0	X	发送重复起始条件
			0	1	0	X	发送停止条件；清除STO标志
			1	1	0	X	发送停止条件，之后发送起始条件；STO被清除
38H	在SLA+W或数据传输中丢失仲裁位	无TWI_DR动作	0	0	0	X	TWI总线被释放；进入非寻址从机模式
			1	0	0	X	在总线空闲时发送起始条件

- **主机接收模式**

主机接收模式中，主机从从机接收一系列数据。为进入主机接收模式，一个起始条件，随后一个从机地址+读控制字(SLA+R)地址包表示进入主机接受模式(MR)。

通过设置控制寄存器 TWI_CR 中的 TWIEN 和 STA，清除 STO 和 TWINT，TWI 逻辑将检测 TWI 总线并在允许时发出一个起始条件(STA)。当起始条件(STA)传输完毕，通讯中断(TWINT)被置起，状态寄存器(TWI_STAT)为 08H，中断服务程序应将从机地址和读控制字(SLA+R)写入数据寄存器 TWI_DR。在开启下一个传输前清除 TWINT 标志。

当从机地址和写控制字传输完毕并收到一个“应答”信息时，中断(TWINT)被置起，状态寄存器 TWI_STAT 中有几个可能的状态：对主机模式有 40H，48H 和 38H，对从机模式有 68H，78H 和 B0H。

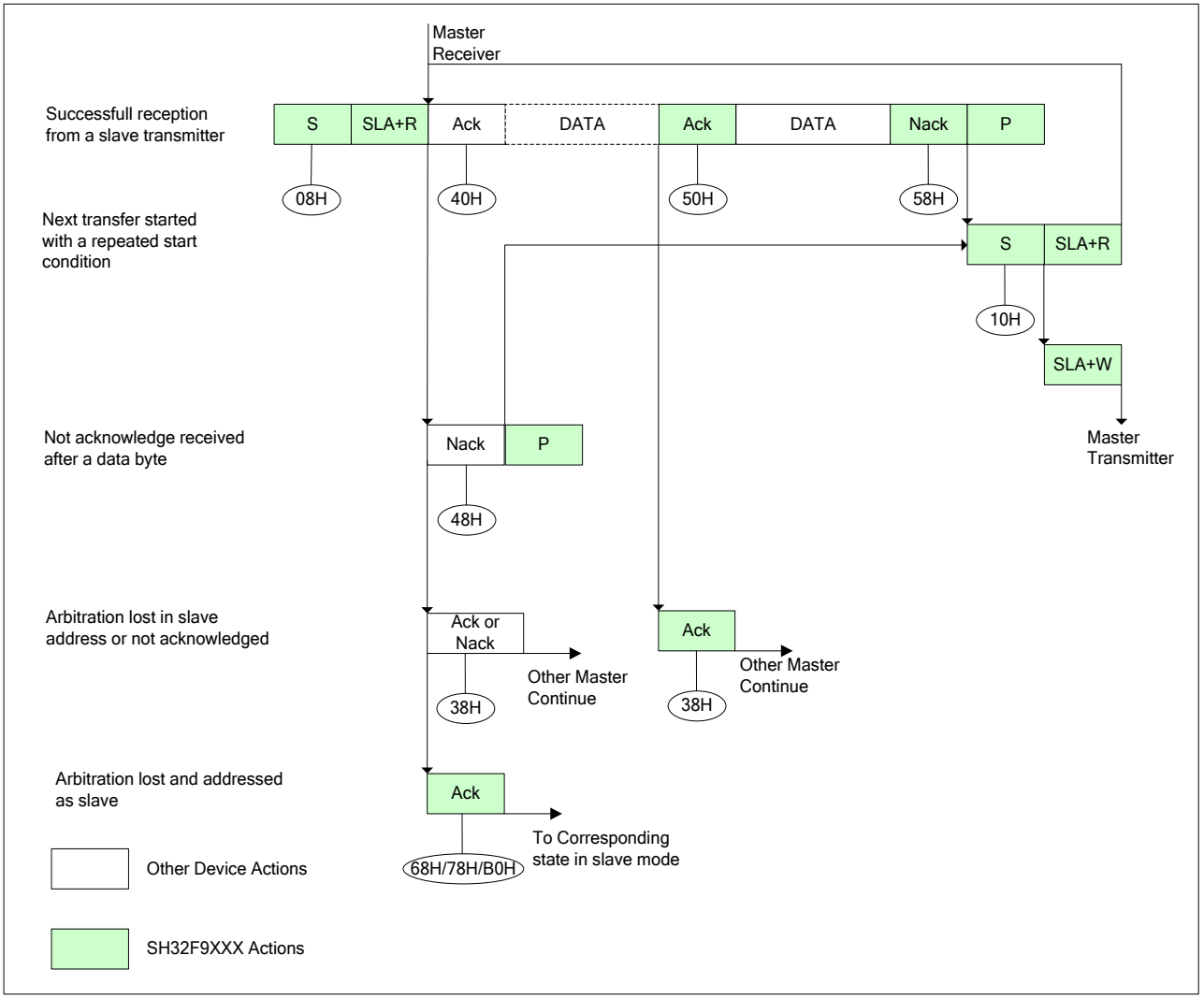


图22-31　　主机接收模式状态图

注：上图中设定SH32F9001_32F9801是主机。

表22-4　　主机接收模式状态码

状态码	TWI总线和硬件接口状态	应用软件响应					TWI执行的下一个动作
		读/写数据寄存器TWI_DR操作	控制位操作				
			STA	STO	TWINT	AA	
08H	已发送起始条件	写入SLA+R	X	0	0	X	发送SLA+R，接收ACK或NACK
10H	已发送重复起始条件	写入SLA+R	X	0	0	X	发送SLA+R，接收ACK或NACK
		写入SLA+W	X	0	0	X	发送SLA+W，TWI将切换到主机发送模式
38H	发送SLA+R或NACK时失去仲裁	无TWI_DR动作	0	0	0	X	TWI总线被释放；进入非寻址从机模式
			1	0	0	X	在总线空闲时发送起始条件
40H	已发送SLA+R；已接收ACK	无TWI_DR动作	0	0	0	0	接收数据，返回NACK
			0	0	0	1	接收数据，返回ACK
48H	已发送SLA+R；已接收NACK	无TWI_DR动作	1	0	0	X	发送重复起始条件
			0	1	0	X	发送停止条件；清除STO标志
			1	1	0	X	发送停止条件，之后发送起始条件；STO被清除
50H	数据已接收；已回应ACK	读取数据	0	0	0	0	接收数据，返回NACK
			0	0	0	1	接收数据，返回ACK
58H	数据已接收；已回应NACK	读取数据	1	0	0	X	发送重复起始条件
			0	1	0	X	发送停止条件；清除STO标志
			1	1	0	X	发送停止条件，之后发送起始条件；STO被清除

- **从机发送模式**

从机发送模式中，从机发送一系列数据到主机。为初始化从机发送模式，必须对控制寄存器 TWI_CR 和地址寄存器 TWI_ADDR 进行初始化：置位控制寄存器 TWI_CR 中的 TWIEN 和 AA，清除 STA、STO 和 TWINT；地址位 TWI_ADDR 为 SH32F9001_32F9801 准备相应的地址。如果 GC 置位，SH32F9001_32F9801 也将响应通用地址(00H)；否则将不响应通用地址。

在 TWIADR 和 TWI_CR 初始化后，SH32F9001_32F9801 将等待总线对自己地址或通用地址(如果 GC 被置位)的响应。如果方向标志位是“读”，则 TWI 进入从机发送模式，否则将进入从机接收模式。在地址和读标志位接收完毕后，中断标志(TWINT)置位，状态寄存器 TWI_STAT 有效。

在传输中，如果将应答使能位“AA”清零，TWI 将传送最后一个字节，并依据主机接收方发送的应答或不应答信息位进入 C0H 或 C8H 状态。总线将切换到非地址从机模式，不在响应主机传输。从而主机接收方将接收到一串“1”。最后一个字节发送完毕后，如果主机仍需额外的数据(传输“应答”信号)，则进入 C8H 状态。

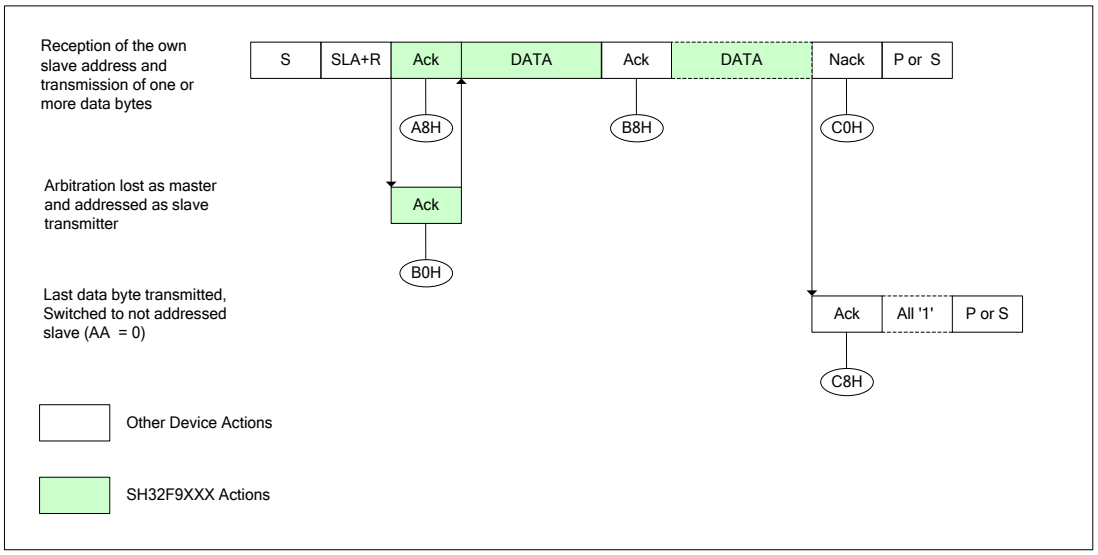


图22-32　　从机发送模式状态图

注：上图中设定SH32F9001_32F9801是从机。

表22-5　　从机发送模式状态码

状态码	TWI总线和硬件接口状态	应用软件响应					TWI执行的下一个动作
		读/写数据寄存器TWI_DR操作	控制位操作				
			STA	STO	TWINT	AA	
A8H	已收到自己SLA+R；已经回应ACK	写入数据字节	X	0	0	0	发送最后数据；等待ACK或NACK回应
			X	0	0	1	发送数据；等待ACK或NACK回应
B0H	作为主机发送SLA+R/W时失去仲裁，收到主机SLA+R；已回应ACK	写入数据字节	X	0	0	0	发送最后数据；等待ACK或NACK回应
			X	0	0	1	发送数据；等待ACK或NACK回应
B8H	已发送TWI_DR数据；已接收ACK回应	写入数据字节	X	0	0	0	发送最后数据；等待ACK或NACK回应
			X	0	0	1	发送数据；等待ACK或NACK回应
C0H	已发送TWI_DR数据；已接收NACK回应	无TWI_DR动作	0	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址
			0	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置
			1	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址；总线空闲时发送“起始条件”
			1	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置；总线空闲时发送“起始条件”
C8H	已发送最后一个TWI_DR数据(AA=0)；已接收ACK回应	无TWI_DR动作	0	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址
			0	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置
			1	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址；总线空闲时发送“起始条件”
			1	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置；总线空闲时发送“起始条件”

- **从机接收模式**

从机接收模式中，从机从主机接收一系列数据。为初始化从机接收模式，必须对控制寄存器 TWI_CR 和地址寄存器 TWI_ADDR 进行初始化：置位控制寄存器 TWI_CR 中的 TWIEN 和 AA，清除 STA、STO 和 TWINT；地址寄存器 TWIADR 中高 7 位为 SH32F9001_32F9801 准备相应的地址。如果 GC 置位，SH32F9001_32F9801 也将响应通用地址(00H)；否则将不响应通用地址。

在 TWI_ADDR 和 TWI_CR 初始化后，SH32F9001_32F9801 将等待总线对自己地址或通用地址(如果 GC 被置位)的响应。如果方向标志位是'写'，则 TWI 进入从机接收模式，否则将进入从机发送模式。在地址和写标志位接收完毕后，中断标志(TWINT)置位，状态寄存器 TWI_STAT 有效。

在传输中，如果将应答使能位"AA"清零，TWI 将接收最后一个字节并回应"不应答"信息。回应"不应答"可以表示当前从机无法接收更多字节。当 AA=0 时，SH32F9001_32F9801 无法回应对自己地址的访问；但仍然监视总线状态，并可以通过 AA=1 恢复对自己地址的相应。可以通过 AA=0 暂时将 SH32F9001_32F9801 从总线隔离。

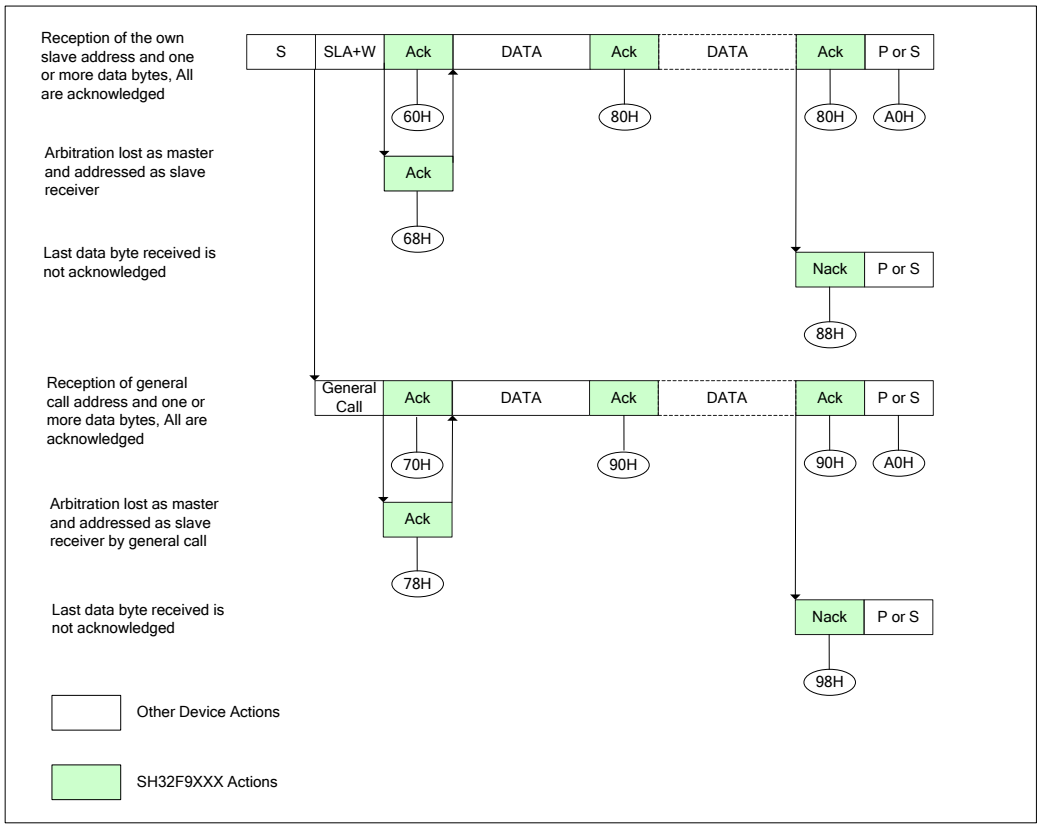


图22-33　　从机接收模式图

注：上图中设定SH32F9001_32F9801是从机。

表22-6　　从机接收模式状态码

状态码	TWI总线和硬件接口状态	应用软件响应					TWI执行的下一个动作
		读/写数据寄存器TWI_DR操作	控制位操作				
			STA	STO	TWINT	AA	
60H	已收到自己SLA+W； 已回应ACK	无TWI_DR动作	X	0	0	0	接收数据；发送NACK回应
			X	0	0	1	接收数据；发送ACK回应
68H	作为主机发送SLA+R/W时失去仲裁，收到主机SLA+W； 已回应ACK	无TWI_DR动作	X	0	0	0	接收数据；发送NACK回应
			X	0	0	1	接收数据；发送ACK回应
70H	收到主机发送通用地址； 已回应 ACK	无TWI_DR动作	X	0	0	0	接收数据；发送NACK回应
			X	0	0	1	接收数据；发送ACK回应
78H	作为主机发送SLA+R/W时失去仲裁，收到主机发送通用地址； 已回应ACK	无TWI_DR动作	X	0	0	0	接收数据；发送NACK回应
			X	0	0	1	接收数据；发送ACK回应
80H	处于已寻址状态； 已收到数据； 已回应ACK	读取数据	X	0	0	0	接收数据；发送NACK回应
			X	0	0	1	接收数据；发送ACK回应
88H	处于已寻址状态； 已收到数据； 已回应NACK	读取数据	0	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址
			0	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置
			1	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址；总线空闲时发送“起始条件”
			1	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置；总线空闲时发送“起始条件”
90H	处于通用地址已寻址状态； 已收到数据； 已回应ACK	读取数据	X	0	0	0	接收数据；发送NACK回应
			X	0	0	1	接收数据；发送ACK回应
98H	处于通用地址已寻址状态； 已收到数据； 已回应NACK	读取数据	0	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址
			0	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置
			1	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址；总线空闲时发送“起始条件”
			1	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置；总线空闲时发送“起始条件”
A0H	作为从机时收到停止条件或重复起始条件	无TWI_DR动作	0	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址
			0	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置
			1	0	0	0	切换至非寻址从机模式；不响应自己地址和通用地址；总线空闲时发送“起始条件”
			1	0	0	1	切换至非寻址从机模式；响应自己地址，是否响应通用地址依赖于寄存器TWI_ADDR中GC的设置；总线空闲时发送“起始条件”

- **其它模式**

除上述状态码外，两个状态码没有明确的 TWI 状态。状态 0F8H 表示由于中断标志 TWINT 未置起，没有相应的状态信息。当中断 TWINT 未置起，即在清除一个状态后到一个新状态建立前均由 0F8H 进行填充。

状态 00H 表示在 TWI 总线通讯中有错误发生，即传输中有非法的起始条件或停止条件发生。例如在传输地址，数据或回应 ACK 应答时有起始条件或停止条件发生。当总线扰乱了内部逻辑时也会产生 00H 状态。当非法状态出现时，会置起中断标志位 TWINT。可通过置起 STO 并清除 TWINT 标志恢复到正常通讯，SH32F9001_32F9801 将进入非寻址从机模式，并自动清除 STO 标志。数据线和时钟线将被释放，线上无停止条件传送。

表22-7　　其它模式状态码

状态码	TWI总线和硬件接口状态	应用软件响应					TWI执行的下一个动作
		读/写数据寄存器TWI_DR操作	控制位操作				
			STA	STO	TWINT	AA	
F8H	没有有效状态码；TWINT=0	无TWI_DR动作	无TWI_CR动作				等待或处理当前传输
00H	在主机或寻址从机模式下有非法起始条件或停止条件发送； 接口导致TWI内部逻辑混乱	无TWI_DR动作	0	1	0	X	只有内部硬件受影响；释放总线；切换到非寻址从机模式；清除STO

22.3.3.6　　支持快速模式 400kbps

TWI 模块兼容两档常用的传输速率：标准模式 100kHz、快速模式 400kHz。通过应用软件配置分频系数 CR[1:0]和比特率配置位 BRT[7:0]来获得两种传输速率。

22.3.3.7　　中断

总线超时标志TOUT，SCL高电平超时标志TFREE，TWI基础功能中断标志TWINT，三者任一标志置位都会引起TWI中断，三者共享中断向量，标志必须由软件清0。

TOUT：总线超时标志，该功能由ETOT控制位开启。当TWI总线低电平超过N*T_{PCLK1}时置位。N值由CNT[1:0]@TWI_CR寄存器位决定。

TFREE：SCL高电平超时标志，该功能由EFREE控制位开启。 参与总线传输时，如时钟线高电平超过TFREE = T_{PCLK1}* TWI_HOC *256所定义的PCLK1时钟个数时硬件置位。HOC值由HOC[7:0]@TWI_HOC寄存器位决定。

TWINT：TWI基础功能中断标志，当TWI模块使能时即开启。当TWI状态改变时TWINT置位。但是，进入状态F8不会使TWINT置位，因为在那种情况下中断服务程序不起作用。

当TWINT置位时，SCL线上的串行时钟低电平周期扩展，且串行传输被中止。当SCL为高时，TWINT标志的状态不受影响。

TWINTE： TWI中断使能位，控制TOUT/TFREE/TWINT三个中断源是否产生中断请求，如果关闭该中断使能位，则应用软件只能通过查询方式进行TWI通信状态处理。（注：TWINTE是SH32F9001_32F9801新增功能）

当进入TWI中断时，TWI状态寄存器TWI_STAT存入状态代码，可识别要执行的多个状态服务中的其中一个（对I2C典型有26个状态）。

注：SH32F9001_32F9801没有STA中断，该中断用于TWI唤醒MCU。

22.3.3.8　　TWI 通信服务程序开发

SH32F9001_32F9801 的 TWI 模块需配合用户应用程序才能实现完整的针对 TWI 的通信协议（根据应用的不同通信协议都有不同），保证了最大的应用灵活性。用户应用程序通过 TWI 中断进行驱动，不会影响主流程和其它控制。具体可参考四种主要模式的状态图和状态码进行开发。对于 MCU 最基本的操作 TWI/I2C 接口外设的情况，只需要参考“主机发送模式”进行开发即可，

一个完整的 TWI 服务程序必须执行的操作包括：

1. 复位后 TWI 模块初始化，包括配置从机地址和通用地址位（GC），使能中断，设置串行时钟频率等。
2. TWI 中断服务，识别状态代码并按分支进入相应的状态服务程序，另外也需要处理超时情况来限制无效的总线或丢失的服务程序。
3. 支持 4 种 TWI 操作模式的多种状态服务程序（I2C 有 26 种），有些模式有一种或几种没用到则相应的状态服务可忽略（只要小心处理，那些状态就不会出现）。

22.3.4 TWI 寄存器

TWI0-1 模块寄存器列表 (基地址: 0x4002 2000)

地址	寄存器名	说明
0x4002 2000 + 0x400*x	FR	TWIx中断标志寄存器 (TWI0～1对应x:0~1)
0x4002 2004 + 0x400*x	STAT	TWIx状态寄存器 (TWI0～1对应x:0~1)
0x4002 2008 + 0x400*x	HOC	TWIx高电平超时检测配置寄存器 (TWI0～1对应x:0~1)
0x4002 200C + 0x400*x	BRT	TWIx比特率配置寄存器 (TWI0～1对应x:0~1)
0x4002 2010 + 0x400*x	DR	TWIx数据寄存器 (TWI0～1对应x:0~1)
0x4002 2014 + 0x400*x	ADDR	TWIx地址配置寄存器 (TWI0～1对应x:0~1)
0x4002 2018 + 0x400*x	CR	TWIx控制寄存器 (TWI0～1对应x:0~1)

22.3.4.1 TWI 中断标志寄存器 (TWIx_FR)

偏移地址: 0x0000 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留												TOUTC	TFREEC	保留	TWINTC
-												WO	WO	-	WO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留												TOUT	TFREE	保留	TWINT
-												RO	RO	-	RO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	*符号*	*说明*
31 ~ 20	**保留**	**-**
19	**TOUTC**	**总线超时标志清除位** 0：无效 1：清除
18	**TFREEC**	**SCL 高电平超时标志清除位** 0：无效 1：清除
17	**保留**	**-**
16	**TWINTC**	**TWI 一字节数据传送中断标志清除位** 0：无效 1：清除
15 ~ 4	**保留**	**-**
3	**TOUT**	**总线超时标志位** 0：无超时发生，由软件对 TOUTC 写 1 清零 1：当 TWI 总线低电平超过 N*T_{PCLK1} 时置位。N 值由 TWITOUT 寄存器决定
2	**TFREE**	**SCL 高电平超时标志位** 0：无超时，由软件写 1 清零 1：参与总线传输时，如时钟线高电平超过 TFREE=T_{PCLK1}* TWI_HOC *256 所定义的 PCLK1 时钟个数时硬件置位
1	**保留**	**-**

0	TWINT	TWI 一字节数据传送中断标志位 0：单字节数据传送中，由软件写 1 清零 1：单字节数据传送完毕，由硬件置位

22.3.4.2 TWI 状态寄存器 (TWIx_STAT)

偏移地址: 0x0004 + 0x400*x

复位值: 0x0000 00F8

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								SR[4:0]					保留		
-								RO					-		
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0

位	符号	说明
31 ~ 8	保留	-
7 ~ 3	SR[4:0]	TWI 串行通讯状态位
2 ~ 0	保留	-

22.3.4.3 TWI 高电平超时检测配置寄存器 (TWIx_HOC)

偏移地址: 0x0008 + 0x400*x

复位值: 0x0000 00FF

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								HOC[7:0]							
-								RW							
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

位	符号	说明
31 ~ 8	保留	-
7 ~ 0	HOC[7:0]	TWI 的 SCL 总线高电平超时检测配置位

22.3.4.4 TWI 比特率配置寄存器 (TWIx_BRT)

偏移地址: 0x000C + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								BRT[7:0]							
-								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7 ~ 0	BRT[7:0]	TWI 的比特率配置位

22.3.4.5 TWI 数据寄存器 (TWIx_DR)

偏移地址: 0x0010 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								DR[7:0]							
-								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7 ~ 0	DR[7:0]	TWI 数据寄存器

22.3.4.6 TWI 地址配置寄存器 (TWIx_ADDR)

偏移地址: 0x0014 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留								TWIAMR[6:0]							保留
-								RW							-
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								ADDR[6:0]							GC
-								RW							RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 24	保留	-
23 ~ 17	TWIAMR[6:0]	TWI 地址屏蔽位 TWIAMR 的每一位可屏蔽 TWIADR 地址寄存器中相应的地址位。如果屏蔽位置 1，地址匹配逻辑忽略输入的地址位与 TWIADR 中的相应地址位的比较结果；如果屏蔽位置 0，不忽略相应位的比较结果
16 ~ 8	保留	-
7 ~ 1	ADDR[6:0]	TWI 地址配置位
0	GC	通用地址使能位 0：禁止响应通用地址 1：允许响应通用地址

22.3.4.7 TWI 控制寄存器 (TWIx_CR)

偏移地址: 0x0018 + 0x400*x

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TWIEN	保留				TWINTE	ETOT	EFREE	CNT[1:0]		CR[1:0]		保留	STA	STO	AA
RW	-				RW	RW	RW	RW		RW		-	RW	RW	RW
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15	TWIEN	**TWI 使能位** 0：关闭 TWI 1：使能 TWI
14 ~ 11	保留	-
10	TWINTE	**TWI 的发送接收中断标志使能位** 0：禁止 TWI 中断标志位 TWINT 触发中断 1：允许 TWI 中断标志位 TWINT 触发中断
9	ETOT	**总线超时中断标志使能位** 0：禁止 TOUT 触发 TWI 中断，但总线超时判断一直有效 1：当 TOUT 位被置‘1’时，允许 TOUT 触发 TWI 中断
8	EFREE	**SCL 高电平超时中断标志使能位** 0：禁止 TFREE 触发 TWI 中断，但 SCL 总线高电平超时判断一直有效 1：当 TFREE 位被置‘1’时，允许 TFREE 触发 TWI 中断
7 ~ 6	CNT[1:0]	**总线超时计数控制位** 00: N=25000 01: N=50000 10: N=100000 11: N=200000 N 值在总线超时功能中使用
5 ~ 4	CR[1:0]	**TWI 分频系数** 00：64 01：16 10： 4 11： 1
3	保留	-
2	STA	**起始位** 0：不发送起始条件 1：当总线空闲时发送起始条件 注：当 STA=1 时发送起始条件并进入主模式，如果已经在主模式，则发送重复起始条件 当 STA 和 STO 都置位时，如果处于主模式，将向总线发送一个停止条件，然后发送一个起始条件。如果接口处于从模式，则产生一个内部停止条件，但不发送到总线上。
1	STO	**终止位** 0：不发送终止条件 1：作为主机时发送终止条件；作为从机时不向总线发送终止条件，但状态恢复到非寻址从机状态。硬件将自动清除该标志位 注：在主模式中，当 STO 为 1 时，会发送一个停止条件或在从模式中从错误状态中恢复。当主模式中 STO=1 时，向总线发送停止条件。当总线检测到停止条件时，STO 自动清零。在从模式中，置位 STO 位可从错误状态中恢复。这种情况下不向总线发送停止条件。硬件的表现就好像是接收到一个停止条件并切换到不可寻址的从接收模式。STO 标志由硬件自动清零。
0	AA	**声明应答标志位** 0：回复“不应答”信号（SDA 高电平） 1：回复“应答”信号（SDA 低电平）

23. 模数转换器（Analog to Digital Converter，ADC）

23.1 简介

SH32F9001_32F9801包含一个单端型12位高速逐次逼近型模数转换器（ADC，Analog-to-Digit Converter），模块图如图23-1所示。ADC基准电压在芯片复位后默认使用AVDD，用户也可以选择外部VREF端口的输入或者内建VREF作为基准电压。

ADC有26路模拟输入通道（ADC.AN0 – ADC.AN23，VREF，TPS）。ADC可编入序列中自动进行转换，结果储存在对应的结果寄存器ADDRx（x = 0 - 7）中，每转换一次序列，结果寄存器的值更新一次，并作一次比较。结果寄存器和模拟输入通道之间的映像关系可以随意编程，从而组成一个转换序列，而且可以将某一模拟输入通道在序列中重复编程，从而在结果寄存器中获得此模拟输入通道连续多次转换的结果。

当启动连续转换功能时，序列自动循环启动转换，结果寄存器也不断更新，并将每次的更新值与极限值作比较。

对于单个信道，转换速率最高可达1MSPS，可由寄存器TS[3:0]@ADCON2设置ADC时钟速率以及采样时间。序列中相邻通道之间的时间间隔亦可通过寄存器TGAP[2:0]@ADCON2设置。

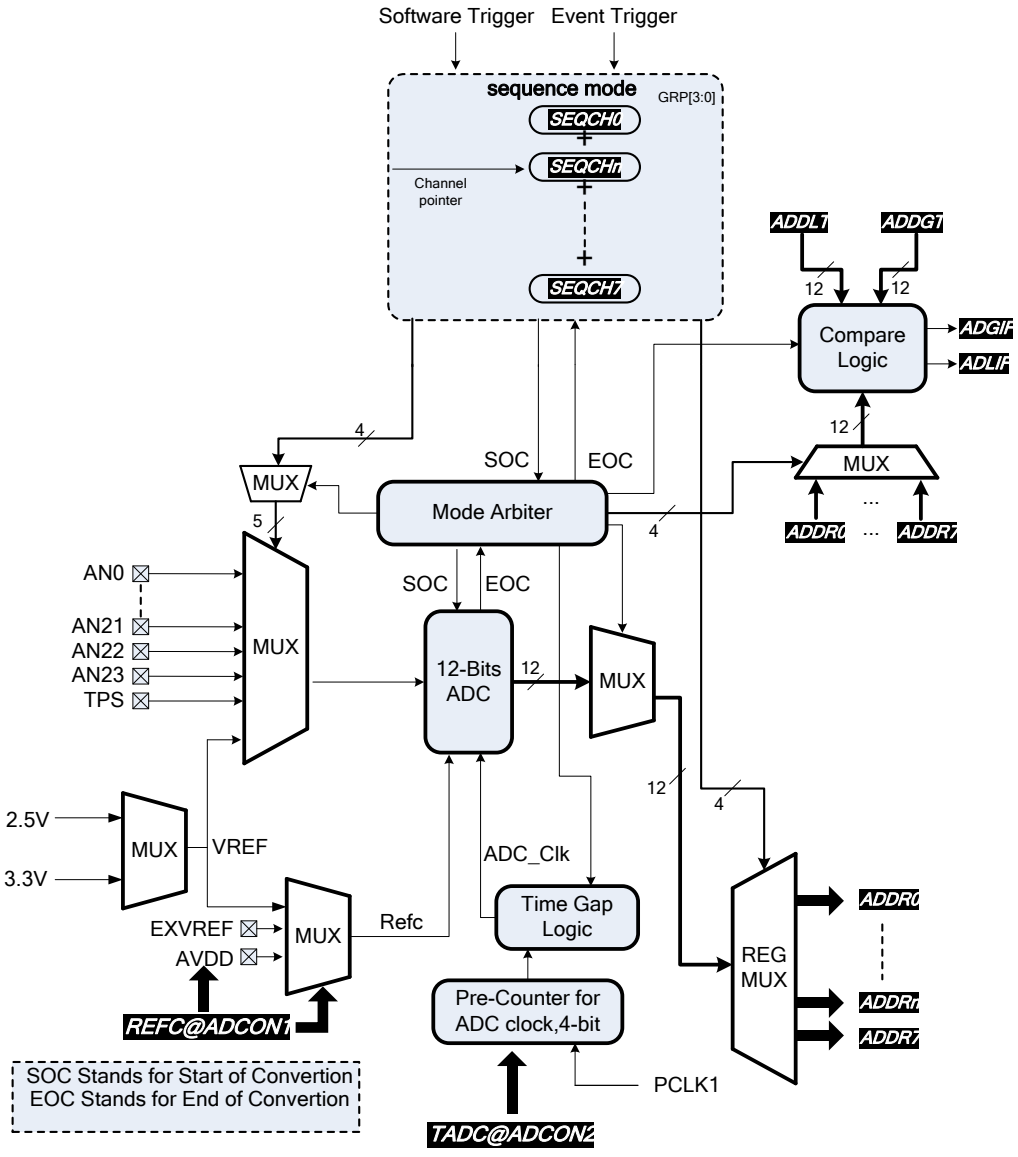


图23-1 ADC 系统框图

23.2 主要特性

- SH32F9001_32F9801内置1个逐次逼近方式的12位A/D转换器
- 参考电压可选内建VREF，外接VREF或AVDD
- 内建参考电压Vref为3.3V和2.5V两种可选(*2.7V以下不支持ADC内建参考电压输出*)
  (详见AMOC章节寄存器TPS_AVREFCON)
- 24路模拟输入口、一个温度传感器通道和一个内建参考电压共26个ADC模拟输入通道
- 启动一次ADC可以自动完成多通道转换（序列），而且每个信道都可以配置为多路模拟输入中的任意一路
- 序列可配置为单信道或多信道，一个序列最多可包含8个通道，转换结果保存在8个结果寄存器中
- 存在三种转换方式：序列转换方式、间断转换方式和连续转换方式
- 序列转换时相邻通道转换之间的时间间隔可由软件设定
- 可由两个PWM模块（PWM0/1）、八个PCA定时器（PCA0/1/2/3/4/5/6/7）、一个普通定时器（TIM7）和外部中断（EXTI0/1/2）的触发信号自动启动AD转换
- 单信道转换速率最高可达1MSPS
- 三种转换方式都带比较功能
- 带DMA功能，为防止ADC触发丢失，建议ADC单独配置一个DMA通道

23.3 功能

23.3.1 单次转换方式

ADCTU[1:0] = 00，ADC配置为单次序列转换模式，一次转换一个序列。序列由单个信道或多个信道组成，对序列进行转换即对序列中的通道进行逐个转换。在硬件上，使得多个信号在同一时间点上转换得以实现（2通道间最短采样间隔0.5us，可近似看作同时）。

转换的结果储存在对应的结果寄存器中，序列转换完成的同时，把指定的某个结果寄存器的值与ADDGT和ADDLT的值比较，并以标志指示比较的结果。

单次序列转换模式下， ADSOC被置起后开始从通道0开始转换，每次单通道转换结束后ADPCH寄存器自动加1，随后进行下一通道转换，当单次序列转换通道个数达到设定值(ADMAXCH[2:0])时，本次序列转换结束，ADSOC位清零，表明本次序列转换完成，同时自动复位ADPCH寄存器为0；同时ADINTF中的ADIF位将由硬件置起，此时若ADCON1中ADIE位为1（注意：ADC的中断不要被屏蔽），则将触发序列转换完成中断，ADIF标志需要软件清除。

例：有4路通道需要转换，按优先级排序为AN0，AN2，AN4，AN7，则设置为ADMAXCH[2:0] = 3，SEQCH0 = 0，SEQCH1 = 2，SEQCH2 = 4，SEQCH3 = 7，则每次启动AD转换（ADSOC=1）时，随后从通道0开始的4个通道将依次转换，结果将依次存在ADDR0，ADDR1，ADDR2，ADDR3中。

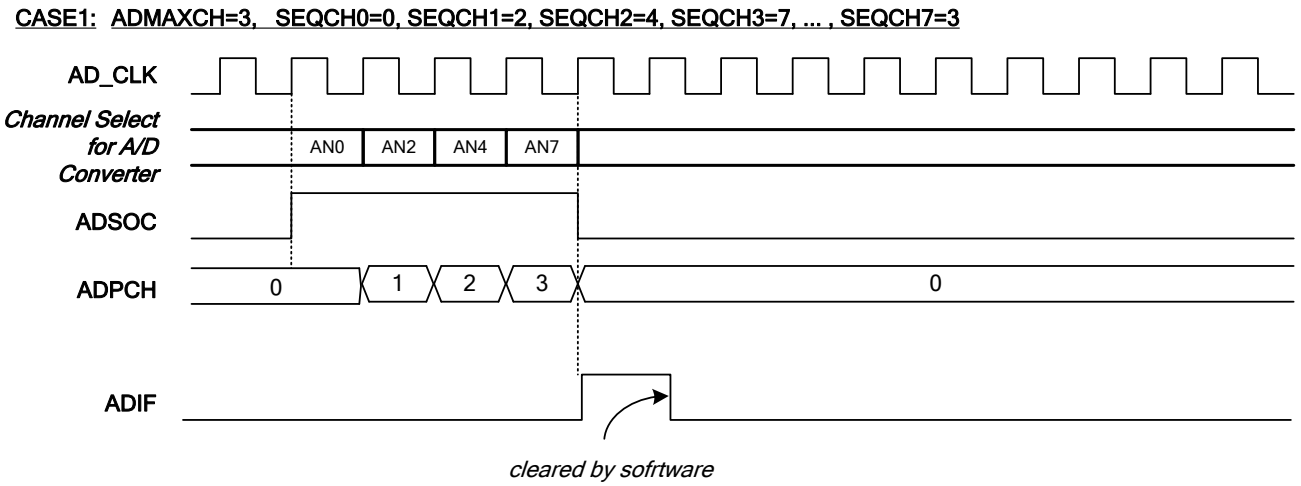


图23-2　　单次转换方式波形示例图

注：在单次转换方式下，ADSOC置1开始序列转换，在转换过程中保持为1，转换完成后硬件自动清0（若转换期间清0此位会立即终止序列转换。）

23.3.2 间断转换方式

ADCTU[1:0] = 01，则ADC工作在间断转换工作模式下，ADC直接从由ADPCH寄存器指定的通道开始转换，当转换个数达到(ADMAXCH[2:0])设定值时，本次序列转换结束。ADC采样通道序号最大为7，因此当转换通道指针寄存器ADPCH超过7时，会自动归0。一个序列转换结束后，如果此时将ADPCH软件改写为0，则下次转换恢复从通道0开始；若不改写ADPCH，则下次转换通道从ADPCH的值开始，若通道的预设模拟通道未进行过设置，则可能得到非预期结果，因此程序对该寄存器的改写要小心处理（ADC转换期间（ADSOC =1），ADPCH无法修改）。

例如，SEQCH0=2，SEQCH1=4，SEQCH2=7，SEQCH3=3，SEQCH4=5，SEQCH5=6，SEQCH6=1，SEQCH7=0，当ADMAXCH=3时，若前次转换结束时ADPCH=6，在单序列顺序采样模式的非连续转换方式下，SEQCH6，SEQCH7，SEQCH0和SEQCH1为有效的通道选择寄存器，由此组成的通道序列为AN1，AN0，AN2和AN4，采样结果将依次存在ADDR6，ADDR7，ADDR0，ADDR1中。转换结束后ADPCH值应该为2，而再次启动AD转换时，AN7，AN3，AN5和AN6将依次转换，结果将依次存在ADDR2，ADDR3，ADDR4，ADDR5中，转换结束后ADPCH值应该为6，以此类推。该示例中，各关键信号的时序关系如图23-3所示：

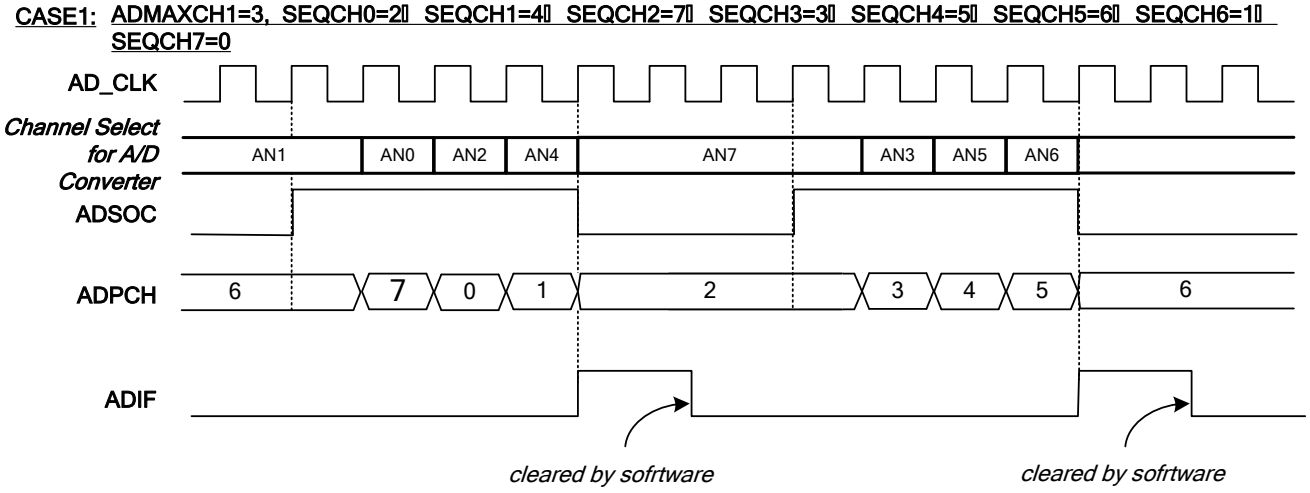


图23-3　　间断转换方式波形示例图

注1：ADPCH只能在ADC不采样（ADSOC等于0）时，能够被软件写入，并且只有在间断模式下才能被软件写入。
注2：ADPCH在上一次转换完成把采样结果送出之后更改，也就是说ADPCH更改时间与ADC结果送出时间一致。

23.3.3 连续转换方式

ADCTU[1:0] = 1x选择连续转换方式，序列循环转换，转换完一个序列便进行此序列的下一次转换。连续转换的一些寄存器设置可参考“单次转换方式”，和序列转换不同的是该模式将循环转换序列，一个序列转换完成到下个序列开始转换的时间也是由TGAP[3:0]控制的。每次序列转换完成的同时，把指定的某个结果寄存器的值与ADDGT和ADDLT的值比较，并以标志指示比较的结果。当软件清除ADSOC位时，AD转换立即停止。

例如，SEQCH0=2，SEQCH1=5，SEQCH2=0，SEQCH3=4，…，SEQCH6=1，SEQCH7=6，当ADMAXCH=3时，在转换开始后，SEQCH0，SEQCH1，SEQCH2和SEQCH3为有效的通道选择寄存器，由此组成的通道序列为AN2，AN5，AN0，AN4。该示例中，各关键信号的时序关系如图23-4所示：

注意：ADPCH的值在采样结束时被清零。

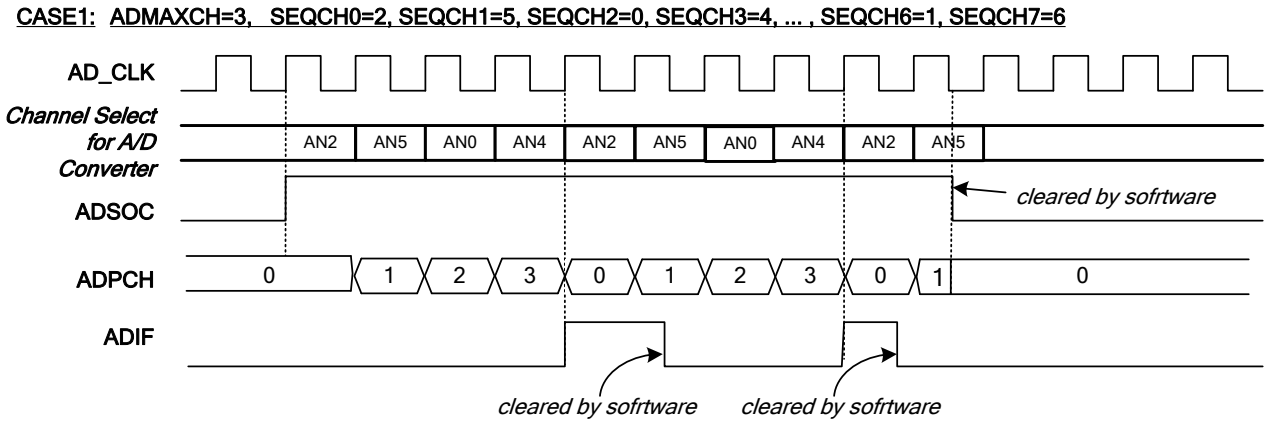


图23-4　　连续转换方式波形示例图

23.3.4　其它功能描述

23.3.4.1　序列转换中通道的选择设置

一个序列可包含单个或多个通道，将待转换的通道号分别存放在通道寄存器SEQCHx中（x = 0 - 7），寄存器SEQCHx有8组，所以序列转换时，一次最多可转换8路通道。而每次转换的通道数量由寄存器ADCON2中的ADMAXCH[2:0]的值决定。例：ADMAXCH[2:0] = 0，即单通道转换，将SEQCH0中存放的通道进行转换；ADMAXCH[2:0] = 3，则序列中有4路通道，将SEQCH0到SEQCH3中存放的通道依次转换。

通道寄存器SEQCHx（x = 0 - 7）中存放的即是待转换的通道号。例：有3路通道需要转换，按优先级排序为AN8，AN2，AN7，则设置为ADMAXCH[2:0] = 2，SEQCH0 = 8，SEQCH1 = 2，SEQCH2 = 7，将依次转换。

注意：*需要转换的信道必须配置为AD信道功能。SEQCHx 中也可设置相同的通道号，比如将SEQCHx 中的值全设置为AN3，结果寄存器中储存的将是不同时间段的AN3的转换值。*

23.3.4.2　序列转换模式 ADC 结果寄存器 ADDRx（x = 0 - 7）

序列转换的结果依次储存在结果寄存器ADDRx（x = 0 - 7）中，结果寄存器为只读寄存器。结果寄存器固定为右对齐的方式，带符号位。如果采样结果为负数，则按照补码的方式存储结果。例如当ADC的基准电压选择VDD是，正的采样结果最大值0x0FFF代表(4095/4096)*VDD，负的采样结果最大值0xF000代表-VDD，如下图。一个序列转换完毕，结果寄存器ADDRx中的值也更新了一次。

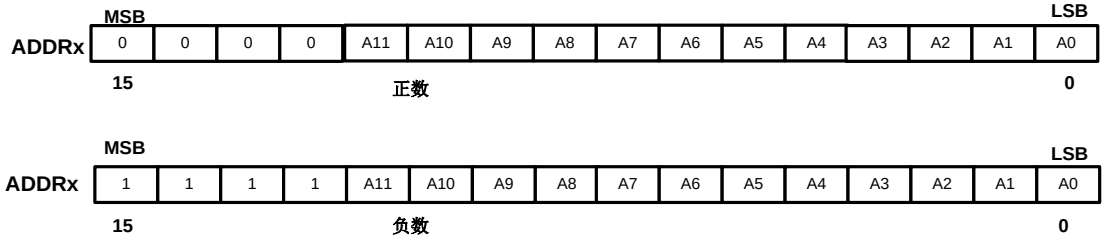


图23-5　　结果寄存器示例图

23.3.4.3　序列转换时通道之间时间间隔（Gap Time）的设置

序列转换时，从上一个通道转换完毕时刻到下一个通道开始采样时刻之间的时间可以通过寄存器 TGAP@ADCON2 位段来设置。当 TGAP[3:0] = 0 时，一个通道转换完毕后立即开始下一个通道的采样，之间没有等待时间。

23.3.4.4　序列转换模式的启动和停止

序列转换模式的启动可以分为软件启动和硬件启动。

软件启动，通过将ADCON1中的ADON位置1，这样可以使能ADC模块的时钟ADC_CLK，同时令ADON位置1可以令ADC模块中的模拟电路上电。当ADCON1寄存器中的ADSOC位置1，则启动转换。当一次序列转换完成，由硬件将ADSOC清0，同时进行比较（在23.3.5节中详细阐述）。当读取ADSOC为1时，则标志转换进行未完成。如果在转换过程中清零ADSOC位，则终止转换。

注：置起ADCON1中的ADON位置1后，需要等待10us的时间之后再开始采样。

硬件启动有5种方式，外部中断（EXTI0~2）引脚下降沿触发启动、普通定时器**(TIM7)**模块触发信号触发启动、PWM0-1模块触发信号触发启动和PCA0-7模块触发信号触发启动。具体通过设置ADCON1寄存器中的ADSTRS位和普通定时器**(TIM7)**模块的相关寄存器。详见表23-1所示。

表23-1　ADC 触发源选择表

ADSTRS[5:0]				发生源	AD开始源	开始条件
-	-	-	-	软件	ADSOC	软件启动
000	x	x	1	外部中断	EXTI0	外部中断0
	x	1	x		EXTI1	外部中断1
	1	x	x		EXTI2	外部中断2
001	x	x	x	定时器7	TIMTRG	定时器7溢出
010	x	x	x	PWM0	PWM0TRG	PWM0溢出
011	x	x	x	PWM1	PWM1TRG	PWM1溢出
100	x	x	1	PCA0	PCA0TRG	PCA0中断标志比较触发
	x	1	x	PCA1	PCA1TRG	PCA1中断标志比较触发
101	x	x	1	PCA2	PCA2TRG	PCA2中断标志比较触发
	x	1	x	PCA3	PCA3TRG	PCA3中断标志比较触发
110	x	x	1	PCA4	PCA4TRG	PCA4中断标志比较触发
	x	1	x	PCA5	PCA5TRG	PCA5中断标志比较触发
111	x	x	1	PCA6	PCA6TRG	PCA6中断标志比较触发
	x	1	x	PCA7	PCA7TRG	PCA7中断标志比较触发

注1：若触发源同时触发，则按照表23-1中开始条件顺序为优先级，依次相应触发源转换申请；

注2：当使用PWM信号触发ADC时，必须要设置死区时间为非零状态（PWMDTR≠0）才能触发ADC转换，如果不设置PWMDTR（上电默认值为0），则PWM无法触发ADC。

硬件触发信号优先于软件触发信号。当软件已经处于序列转换中时，硬件触发会终止先前的序列转换而重新启动一次序列转换，转换值将会覆盖先前的结果。这种情况下，硬件转换在单次模式和连续模式都是从信道0开始转换，间断模式从最开始写入的通道开始转换。

注1：任何硬件触发单次序列、间断转换和连续转换过程中，又来了一次硬件触发，则后一次硬件触发转换请求被忽略。

注2：在转换过程中，即ADSOC为1时，对除了ADCON1、ADCMPCON、ADDGT、ADDLT以外的所有ADC寄存器的写入都将视为无效操作；ADCMPCON寄存器的CSEL[2:0]位在ADSOC为1时，只有在连续转换方式可以修改，其它方式不可修改。

23.3.4.5　序列转换完成中断

序列转换完毕后，ADIF@ADINTF将由硬件置起，此时若对应的ADIE@ADCON1为1，则将触发序列转换完成中断，ADIF@ADINTF位只能由软件清除。

23.3.5 比较功能

23.3.5.1 比较用结果寄存器的指定

由寄存器CSEL@ADCMPCON来指定被比较的结果寄存器，若CSEL = n，则结果寄存器ADDRn将与ADDGT和ADDLT的值比较。以图23-6的配置为例，若设置CSEL = 2，每次序列转换完成的同时将ADDR2的值与ADDGT和ADDLT的值比较。

需要注意的是，若指定一个序列转换不使用的结果寄存器，则不会发生任何比较动作。如上例中令CSEL = 3，转换后未使用结果寄存器ADDR3，故没有任何比较动作发生，标志位ADGIF和ADLIF的值不会发生变化。

注意：*在写入比较值 ADDGT 和 ADDLT 时，应注意结果寄存器的存储方式是左对齐还是右对齐，数据写入格式应和结果寄存器的存储格式一致。ADDGT 和 ADDLT 写入值有即时生效性，比较时会采用 ADDGT 和 ADDLT 最近一次的更新值。*

23.3.5.2 比较过程

设CSEL = n，ADDGT = Max，ADDLT = Min，当序列转换完成时，立即将指定结果寄存器ADDRn的值与Max和Min比较。若ADDRn >= Max，则ADCMPCON寄存器中ADGIF位将置1，若ADGIE位为1，则可以触发一个*比较中断*（与序列转换完成共享一个中断向量），ADGIF位将保持为1直到软件将其清除；若ADDRn <= Min，则ADCMPCON寄存器中ADLIF位将置1，若ADLIE位为1，也可以触发一个*比较中断*（与序列转换完成共享一个中断向量），ADLIF位将保持为1直到软件将其清除。

设ADDGT = 0x3DF，ADDLT = 0x8FE，CSEL = n，则ADDRn的值与ADGIF、ADLIF位中值的关系如图23-6的左半部分所示；设ADDGT = 0x8FE，ADDLT = 0x3DF，CSEL = n，则ADDRn的值与ADGIF、ADLIF位中值的关系如图23-6的右半部分所示。

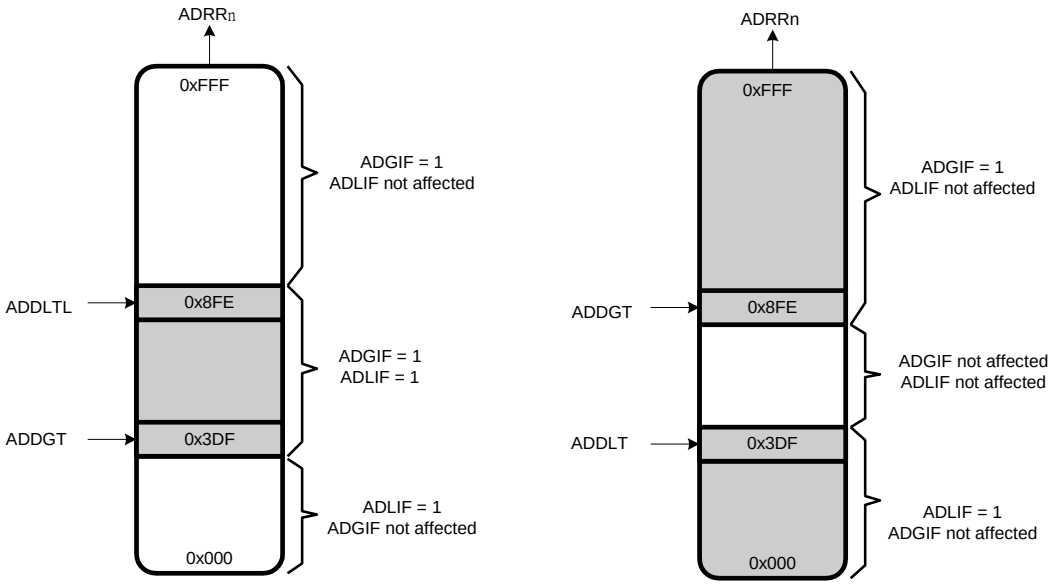


图23-6 比较过程图

23.3.5.3 比较中断

每次序列转换完成时，都会将指定结果寄存器的值去作比较，如果超限，可以产生一个比较中断。因为比较和序列转换完成是同时发生的，所以中断的产生也是同时。

23.3.6 ADC 转换时间设置

通过寄存器ADCON2可以设置ADC的时钟以及采样时间。通过设置ADCON2中的TADC[3:0]位段可以设置ADC的时钟。

寄存器ADCON2中的TS[3:0]位段可以设置每个通道的采样时间t_{SAMP}，t_{SAMP} = (TS[3:0] + TGAP[2:0]* GAPEN +1) * t_{AD}，具体见ADC寄存器章节。需要注意的是，无论TS[3:0]取何值，采样时间的最小值不会小于1 * t_{AD}，最大值不会大于16 * t_{AD}。即若TS[3:0] = 0，采样时间亦为1 * t_{AD}；若TS[3:0] = 15，采样时间亦为16 * t_{AD}。

对于12BIT模式AD转换每个通道的AD转换时间固定为15 * t_{AD}。因此每个通道的总共转换时间 = t_{SAMP} + 15 * t_{AD}。对于一个序列，若寄存器ADCON2中的位段TGAP[3:0]不为零，则序列中相邻的两个通道转换之间将加入一段时间T_G，这样若一个序列总共转换的通道数为n，则完成整个序列的转换所需时间为：n * (t_{SAMP} + 15 * t_{AD}) + (n - 1) * T_G。如果有3个序列采样，则序列的转换时间为3 * (t_{SAMP} + 15 * t_{AD}) + 2 * T_G。

对于10BIT模式AD转换每个通道的AD转换时间固定为13 * tAD。因此每个通道的总共转换时间 = t_{SAMP} + 13 * t_{AD}。

23.3.7 ADC 模块参考电压的设置

可以选择芯片AV_{DD}、内建V_{REF}或者V_{REF}引脚上的输入电压作为ADC模块的参考电压。芯片复位后，ADC模块的参考电压为芯片的AV_{DD}。令寄存器ADCON1中的REFC[1:0]位置1将使ADC模块使用V_{REF}引脚上的电压作为参考电压。

可以选择输出内建参考电压和不输出内建参考电压。

23.3.8 ADC 信道与 IO 口功能设置

注1：ADC输入I/O口设为模拟口。

注2：以前的I/O口配置寄存器在ADC模块中，现在在GPIO模块中。

23.3.9 DMA 请求

能够通过DMA存储ADC采样结果，在每个通道采样结束后，触发DMA功能，实现连续转换而不需要软件干预。可以通过ADDE@ADNCON1位来选择是否开启此功能。

23.3.10 AD 转换时对传感器输出阻抗的要求

如图23-7所示，为了保证AD转换在一定的转换速率的条件下能转换出精确的AD结果，需要在一定时间内（采样时间TS）对SH32F9001_32F9801内部采样电容C完成充电。如果在规定的采样时间内，采样电容C未能充满电，即V0 < V1，AD转换结果则会产生精度误差。因为R0（SH32F9001_32F9801内部电阻）和C（内部采样电容）是常量，已经由芯片内定，能否在转换出允许精度范围内的AD结果就取决于传感器内部等效电阻Rx。设AD转换结果的分辨率为M（12bit结果的分辨率为4096），如果AD转换的精度要求为N，则对传感器电阻的选择有如下要求：

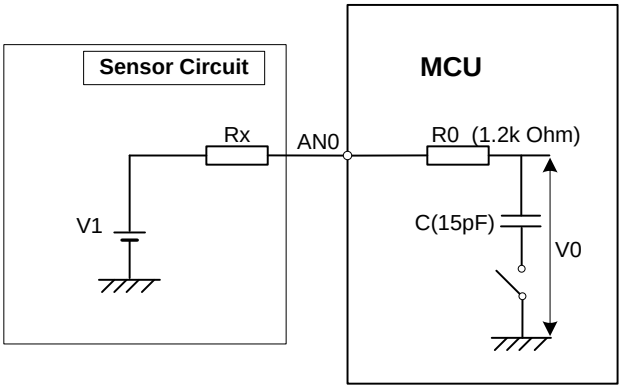


图23-7　　输出阻抗

$$Rx \le -\frac{T}{C * Ln\frac{V1-V0}{V1}} - R0$$

上式也可表示为：

$$Rx \le -\frac{T}{C * Ln\frac{N}{M}} - R0$$

对于12bit模式下即4096分辨率，标准采样时间为2us，结果精度误差控制在0.1LSB内时，按上述公式带入有

$$Rx \le -\frac{2\times10^{-6}}{15\times10^{-12} Ln\frac{0.1}{4096}} - 1.2\times10^{3} \approx 11.3K\Omega$$

即传感器选型时的内部等效电阻必须小于11.3KΩ。

23.3.11 注意事项

ADC转换，最好将ADON@ADCON1打开10us后再将ADSOC@ADCON1(GO/$\overline{DONE}$)位置1，进行转换。因为ADC模块打开后有个稳定时间。

转换模拟通路时要先将端口设置为AD信道功能才能顺利转换。

当ADIE，ADLIE，ADGIE的都开启时，ADIF，ADLIF，ADGIF任意一个置位都能引起中断，且共享一个中断向量，通过判位ADIF，ADLIF，ADGIF哪个为1，来确定具体的中断源并定制特定的操作。

连续转换方式：可以通过软件和硬件启动。

注1：在单次转换过程中，即ADSOC为1时，对除了ADCON1.ADON、ADCON2、ADDGT、ADDLT以外的所有ADC寄存器的写入都将视为无效操作。

注2：在循环转换过程中，即ADSOC为1时，对除了ADCON1.ADON、ADCON2、ADCMPCON、ADDGT、ADDLT以外的所有ADC寄存器的写入都将视为无效操作。

23.4 ADC 寄存器

ADC 模块寄存器列表 (基地址: 0x4002 2800)

地址	寄存器名	说明
0x4002 2800	ADCON1	ADC控制寄存器1
0x4002 2804	ADCON2	ADC控制寄存器2
0x4002 2808	ADPCH	ADC采样转换通道指针寄存器
0x4002 280C	ADDR0	ADC结果寄存器0
0x4002 2810	ADDR1	ADC结果寄存器1
0x4002 2814	ADDR2	ADC结果寄存器2
0x4002 2818	ADDR3	ADC结果寄存器3
0x4002 281C	ADDR4	ADC结果寄存器4
0x4002 2820	ADDR5	ADC结果寄存器5
0x4002 2824	ADDR6	ADC结果寄存器6
0x4002 2828	ADDR7	ADC结果寄存器7
0x4002 282C	ADCMPCON	ADC比较控制寄存器
0x4002 2830	ADDGT	ADC上限比较寄存器
0x4002 2834	ADDLT	ADC下限比较寄存器
0x4002 2838	SEQCHSEL0	ADC通道选择寄存器0
0x4002 283C	SEQCHSEL1	ADC通道选择寄存器1
0x4002 2840	ADGAPON	ADC通道GAP控制位
0x4002 2844	ADINTF	ADC中断标志和清除寄存器

23.4.1 ADC 控制寄存器 1 (ADC_ADCON1)

偏移地址: 0x0000

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADON	ADSTRS[5:0]						保留	ADIE	ADDE	REFC[1:0]		MOD	ADCTU[1:0]		ADSOC
RW	*RW*						-	*RW*	*RW*	*RW*		*RW*	*RW*		*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15	**ADON**	**ADC允许位** 0：禁止ADC模块 1：允许ADC模块
14 ~ 9	**ADSTRS[5:0]**	**ADC触发源选择位：** 为ADC的通道序列启动转换使能位。 见SPEC文档中的表23-1
8	保留	-
7	**ADIE**	**通道序列转换结束中断使能位：** 0：禁止ADC转换结束中断 1：使能ADC转换结束中断
6	**ADDE**	**通道序列转换结束DMA触发使能位：** 0：禁止ADC转换结束触发DMA 1：使能ADC转换结束触发DMA
5 ~ 4	**REFC[1:0]**	**基准电压选位** 00：选择AVDD为基准电压

		01：选择外部VREF端口输入为基准电压 10：选择内建的VREF为基准电压 11：保留 注：因为内建VREF和外部VREF共用引脚，ADC不能既选择内建VREF又选外接VREF
3	**MOD**	**ADC工作模式选择位：** 0：12bit 1：10bit
2 ~ 1	**ADCTU[1:0]**	**ADC通道序列的转换方式控制位：** 00：单次序列转换方式 01：间断序列转换方式 10：连续转换方式 11：保留 注：修改该控制位，会在下一次启动AD转换过程时才产生效果
0	**ADSOC**	**ADC通道序列启动AD转换请求位：** 0：未进行序列转换/连续转换或已完成。写0可取消现在正在进行的转换。 1：在单次和间断转换方式下，置1开始序列转换，在转换过程中保持为1，转换完成后硬件自动清0（若转换期间清0此位会立即终止序列转换。） 在连续转换方式下，置1开始连续转换，则该位不会由硬件清0，可以由软件清0立即终止转换

23.4.2 ADC 控制寄存器 2 (ADC_ADCON2)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留													TGAP [2:0]		
-													RW		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留				TS [3:0]				保留	ADMAXCH[2:0]			TADC[3:0]			
-				RW				-	RW			RW			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 19	保留	-
18 ~ 16	**TGAP [2:0]**	**序列中时相邻通道之间时间间隔设置位段** 在一次序列转换中，一个通道转换完毕到下一个通道开始采样之间的时间间隔 000：无等待时间 001：2ADC时钟周期 010：4ADC时钟周期 011：8ADC时钟周期 100：16ADC时钟周期 101：32ADC时钟周期 110：64ADC时钟周期 111：128ADC时钟周期
15 ~ 12	保留	-
11 ~ 8	**TS [3:0]**	**采样时间选择** 1 tAD ≤ 采样时间 = (TS [3:0]+1) * tAD ≤ 16 tAD 注1：即使TS[3:0] = 0000，最小采样时间为1tAD；即使TS[3:0] = 1111，最大采样时间为16tAD； 注2：在设置TS[3:0]前，请估算连接到ADC输入引脚的串联电阻； 注3：选择1 * tAD为采样时间时，请确保连接到ADC输入引脚的串联电阻满足输入阻抗需求，详见23.3.10； 注4：TADC设置需保证ADC时钟周期 Tadc ≥ 50ns

		注5：一个通道的总共转换时间 = 15tAD + 采样时间
7	保留	-
6 ~ 4	ADMAXCH[2:0]	**ADC通道序列的总长度设置位（设置值0~7）：** 一次AD转换过程的模拟通道总数为ADMAXCH+1
3 ~ 0	TADC[3:0]	**ADC时钟周期选择** 0000：ADC时钟周期tAD = 1 t_{PCLK1} 0001：ADC时钟周期tAD = 2 t_{PCLK1} 0010：ADC时钟周期tAD = 3 t_{PCLK1} 0011：ADC时钟周期tAD = 4 t_{PCLK1} 0100：ADC时钟周期tAD = 5 t_{PCLK1} 0101：ADC时钟周期tAD = 6 t_{PCLK1} 0110：ADC时钟周期tAD = 8 t_{PCLK1} 0111：ADC时钟周期tAD = 12 t_{PCLK1} 1000：ADC时钟周期tAD = 16 t_{PCLK1} 1001：ADC时钟周期tAD = 24 t_{PCLK1} 1010：ADC时钟周期tAD = 32 t_{PCLK1} 1011：ADC时钟周期tAD = 48 t_{PCLK1} 1100：ADC时钟周期tAD = 64 t_{PCLK1} 1101：ADC时钟周期tAD = 128 t_{PCLK1} 1110：ADC时钟周期tAD = 256 t_{PCLK1} 1111：ADC时钟周期tAD = 320 t_{PCLK1} 注：t_{PCLK1}为APB1的时钟周期

23.4.3 ADC 采样转换通道指针寄存器 (ADC_ADPCH)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留													ADPCH[2:0]		
-													*RW*		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 3	保留	-
2 ~ 0	ADPCH[2:0]	**ADC采样转换通道指针寄存器** 000：ADC当前采样通道为SEQCH0 001：ADC当前采样通道为SEQCH1 010：ADC当前采样通道为SEQCH2 011：ADC当前采样通道为SEQCH3 100：ADC当前采样通道为SEQCH4 101：ADC当前采样通道为SEQCH5 110：ADC当前采样通道为SEQCH6 111：ADC当前采样通道为SEQCH7 注1：ADPCH只能在ADC不采样（ADSOC等于0）时，能够被软件写入，并且只有在间断模式下才能被软件写入。 注2：ADPCH在上一次转换完成把采样结果送出之后更改，也就是说ADPCH更改时间与ADC结果送出时间一致。

23.4.4 ADC 结果寄存器 n (ADC_ADDRn)(n=0..7)

偏移地址: 0x000C
:0x0010
:0x0014
:0x0018
:0x001C
:0x0020
:0x0024
:0x0028
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ADDRn[15:0]															
RO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	**ADDRn[15:0]**	一个通道转换完毕后，数据立即更新并存放在ADDRn（x = 0..7）中序列转换一次所有结果寄存器也都更新了一次

23.4.5 ADC 比较控制寄存器 (ADC_ADCMPCON)

偏移地址: 0x002C
复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								ADGIE	ADGDE	ADLIE	ADLDE	保留	CSEL[2:0]		
-								*RW*	*RW*	*RW*	*RW*	-	*RW*		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7	**ADGIE**	**上限比较中断使能位** 0：禁止上限比较中断 1：使能上限比较中断
6	**ADGDE**	**上限比较DMA使能位** 0：禁止上限比较触发DMA 1：使能上限比较触发DMA
5	**ADLIE**	**下限比较中断使能位** 0：禁止下限比较中断 1：使能下限比较中断
4	**ADLDE**	**下限比较DMA使能位** 0：禁止下限比较触发DMA 1：使能下限比较触发DMA
3	保留	-
2 ~ 0	**CSEL[2:0]**	**用于比较的结果寄存器选择位段** 0：选择ADDR0中的值与ADDGT、ADDLT比较

		1：选择ADDR1中的值与ADDGT、ADDLT比较 2：选择ADDR2中的值与ADDGT、ADDLT比较 3：选择ADDR3中的值与ADDGT、ADDLT比较 4：选择ADDR4中的值与ADDGT、ADDLT比较 5：选择ADDR5中的值与ADDGT、ADDLT比较 6：选择ADDR6中的值与ADDGT、ADDLT比较 7：选择ADDR7中的值与ADDGT、ADDLT比较

23.4.6 ADC 上限比较寄存器 (ADC_ADDGT)

偏移地址: 0x0030

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
GT[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	GT[15:0]	**ADC上限比较寄存器** 当一次序列转换完成时，ADCMPCON寄存器中CSEL位段指定结果寄存器的值立刻与ADDGT中的值比较，若大于等于ADDGT中的值则令ADCMPCON中的ADGIF位置1，ADGIF位将保持为1直到软件将其清除

23.4.7 ADC 下限比较寄存器 (ADC_ADDLT)

偏移地址: 0x0034

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
LT[15:0]															
RW															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 16	保留	-
15 ~ 0	LT[15:0]	**ADC下限比较寄存器** 当一次序列转换完成时，ADCMPCON寄存器中CSEL位段指定结果寄存器的值立刻与ADDLT中的值比较，若小于等于ADDLT中的值则令ADCMPCON中的ADLIF位置1，ADLIF位将保持为1直到软件将其清除

23.4.8 ADC 通道选择寄存器 0 (ADC_SEQCHSEL0)

偏移地址: 0x0038

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留			SEQCH3[4:0]					保留			SEQCH2[4:0]				
-			RW					-			RW				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留			SEQCH1[4:0]					保留			SEQCH0[4:0]				
-			RW					-			RW				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 29	保留	-
28 ~ 24	**SEQCH3[4:0]**	**模拟通道3选择寄存器组，分别指定对应的模拟输入通道：** 00000 - AN0 00001 - AN1 00010 - AN2 00011 - AN3 00100 - AN4 00101 - AN5 00110 - AN6 00111 - AN7 01000 - AN8 01001 - AN9 01010 - AN10 01011 - AN11 01100 - AN12 01101 - AN13 01110 - AN14 01111 - AN15 10000 - AN16 10001 - AN17 10010 - AN18 10011 - AN19 10100 - AN20 10101 - AN21 10110 - AN22 10111 - AN23 11000 - Vref 11001 - TPS(温度传感器) 11010~11111 - 保留
23 ~ 21	保留	-
20 ~ 16	**SEQCH2[4:0]**	**模拟通道2选择寄存器组，分别指定对应的模拟输入通道：** 00000 - AN0 00001 - AN1 00010 - AN2 00011 - AN3 00100 - AN4 00101 - AN5 00110 - AN6 00111 - AN7 01000 - AN8 01001 - AN9 01010 - AN10 01011 - AN11 01100 - AN12

		01101 - AN13 01110 - AN14 01111 - AN15 10000 - AN16 10001 - AN17 10010 - AN18 10011 - AN19 10100 - AN20 10101 - AN21 10110 - AN22 10111 - AN23 11000 - Vref 11001 - TPS(温度传感器) 11010~11111 - 保留
15 ~ 13	保留	-
12 ~ 8	**SEQCH1[4:0]**	**模拟通道1选择寄存器组，分别指定对应的模拟输入通道：** 00000 - AN0 00001 - AN1 00010 - AN2 00011 - AN3 00100 - AN4 00101 - AN5 00110 - AN6 00111 - AN7 01000 - AN8 01001 - AN9 01010 - AN10 01011 - AN11 01100 - AN12 01101 - AN13 01110 - AN14 01111 - AN15 10000 - AN16 10001 - AN17 10010 - AN18 10011 - AN19 10100 - AN20 10101 - AN21 10110 - AN22 10111 - AN23 11000 - Vref 11001 - TPS(温度传感器) 11010~11111 - 保留
7 ~ 5	保留	-
4 ~ 0	**SEQCH0[4:0]**	**模拟通道0选择寄存器组，分别指定对应的模拟输入通道：** 00000 - AN0 00001 - AN1 00010 - AN2 00011 - AN3 00100 - AN4 00101 - AN5 00110 - AN6 00111 - AN7 01000 - AN8 01001 - AN9 01010 - AN10 01011 - AN11 01100 - AN12 01101 - AN13 01110 - AN14 01111 - AN15

		10000 - AN16 10001 - AN17 10010 - AN18 10011 - AN19 10100 - AN20 10101 - AN21 10110 - AN22 10111 - AN23 11000 - Vref 11001 - TPS(温度传感器) 11010~11111 - 保留

23.4.9 ADC 通道选择寄存器 1 (ADC_SEQCHSEL1)

偏移地址: 0x003C

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留			SEQCH7[4:0]					保留			SEQCH6[4:0]				
-			RW					-			RW				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留			SEQCH5[4:0]					保留			SEQCH4[4:0]				
-			RW					-			RW				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 29	保留	-
28 ~ 24	**SEQCH7[4:0]**	**模拟通道7选择寄存器组，分别指定对应的模拟输入通道：** 00000 - AN0 00001 - AN1 00010 - AN2 00011 - AN3 00100 - AN4 00101 - AN5 00110 - AN6 00111 - AN7 01000 - AN8 01001 - AN9 01010 - AN10 01011 - AN11 01100 - AN12 01101 - AN13 01110 - AN14 01111 - AN15 10000 - AN16 10001 - AN17 10010 - AN18 10011 - AN19 10100 - AN20 10101 - AN21 10110 - AN22 10111 - AN23 11000 - Vref 11001 - TPS(温度传感器) 11010~11111 - 保留
23 ~ 21	保留	-
20 ~ 16	**SEQCH6[4:0]**	**模拟通道6选择寄存器组，分别指定对应的模拟输入通道：** 00000 - AN0 00001 - AN1 00010 - AN2

		00011 - AN3 00100 - AN4 00101 - AN5 00110 - AN6 00111 - AN7 01000 - AN8 01001 - AN9 01010 - AN10 01011 - AN11 01100 - AN12 01101 - AN13 01110 - AN14 01111 - AN15 10000 - AN16 10001 - AN17 10010 - AN18 10011 - AN19 10100 - AN20 10101 - AN21 10110 - AN22 10111 - AN23 11000 - Vref 11001 - TPS(温度传感器) 11010~11111 - 保留
15 ~ 13	保留	-
12 ~ 8	**SEQCH5[4:0]**	**模拟通道5选择寄存器组，分别指定对应的模拟输入通道：** 00000 - AN0 00001 - AN1 00010 - AN2 00011 - AN3 00100 - AN4 00101 - AN5 00110 - AN6 00111 - AN7 01000 - AN8 01001 - AN9 01010 - AN10 01011 - AN11 01100 - AN12 01101 - AN13 01110 - AN14 01111 - AN15 10000 - AN16 10001 - AN17 10010 - AN18 10011 - AN19 10100 - AN20 10101 - AN21 10110 - AN22 10111 - AN23 11000 - Vref 11001 - TPS(温度传感器) 11010~11111 - 保留
7 ~ 5	保留	-
4 ~ 0	**SEQCH4[4:0]**	**模拟通道4选择寄存器组，分别指定对应的模拟输入通道：** 00000 - AN0 00001 - AN1 00010 - AN2 00011 - AN3 00100 - AN4 00101 - AN5

		00110 - AN6 00111 - AN7 01000 - AN8 01001 - AN9 01010 - AN10 01011 - AN11 01100 - AN12 01101 - AN13 01110 - AN14 01111 - AN15 10000 - AN16 10001 - AN17 10010 - AN18 10011 - AN19 10100 - AN20 10101 - AN21 10110 - AN22 10111 - AN23 11000 - Vref 11001 - TPS(温度传感器) 11010~11111 - 保留

23.4.10 ADC 通道 GAP 控制位 (ADC_ADGAPON)

偏移地址: 0x0040

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留								GAPENy(y=7~0)							
-								RW							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 8	保留	-
7 ~ 0	GAPENy(y=7~0)	ADC通道y的GAP时间使能位： 0：禁止 1：使能

23.4.11 ADC 中断标志和清除寄存器 (ADC_ADINTF)

偏移地址: 0x0044

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留													ADIFC	ADGIFC	ADLIFC
-													WO	WO	WO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留													ADIF	ADGIF	ADLIF
-													RO	RO	RO
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 19	保留	-
18	ADIFC	通道序列转换结束中断标志位清除位 0：无效 1：清除
17	ADGIFC	上限比较中断标志位清除位 0：无效 1：清除
16	ADLIFC	下限比较中断标志位清除位 0：无效 1：清除
15 ~ 3	保留	-
2	ADIF	通道序列转换结束中断标志位： 0：无通道序列转换结束中断发生 1：有通道序列转换结束中断发生
1	ADGIF	上限比较中断标志位 0：无上限中断发生，CSEL位段所指定结果寄存器中的最近一次更新值小于ADDGT中的值 1：发生上限中断，CSEL位段所指定结果寄存器中的最近一次更新值大于等于ADDGT中的值 该标志置起后需由软件清0
0	ADLIF	下限比较中断标志位 0：无下限中断发生，CSEL位段所指定结果寄存器中的最近一次更新值大于ADDLT中的值 1：发生下限中断，CSEL位段所指定结果寄存器中的最近一次更新值小于等于ADDLT中的值 该标志置起后需由软件清0

24. 片上模拟功能模块（Analog Module On Chip，AMOC）

24.1 简介

SH32F9001_32F9801片上集成了一个温度传感器。

24.2 主要特性

- 内置温度传感器，可软件校准
- 温度传感器可作为ADC输入通道，信号可通过内部独立放大

24.3 温度传感器

24.3.1 功能介绍

SH32F9001_32F9801包含一个温度传感器转换器，可通过TPSEN位控制，温度传感器由独立的内部运放放大后再送入ADC进行采样，转换为数字值。

为了更准确读取温度传感器数值，建议同时开启内部运放的CHOP功能，即在每次转换时同时切换温度传感器与内部运放的CHOP的开启与关闭，在CHOP开启/关闭状态下分别得到温度传感器AD转换数值，取平均后即可得到更准确的温度转换结果。

温度 T = ((VTPS-V25) / Tslope) + 25

V25 为25度时温度传感器输出电压

VTPS 为温度传感器输出电压

Tslope 为温度传感器斜率

24.3.2 温度传感器校准

SH32F9001_32F9801每颗芯片出厂后都校准温度传感器，分别将生产时实际温度及TPSCHOP关闭和开启时采样的AD值存放在地址信息存储区，可以由程序或编程工具读出。将实时AD转换温度与校准时存储AD值相减后加上校准时温度值则可以得到真实的温度。

校准时温度传感器实际温度值存于0FFF F070H – 0FFF F071H，校准时SH32F9001_32F9801温度传感器TPSCHOP关闭和开启时AD值分别0FFF F072H – 0FFF F075H。校准时实际温度保留1位小数，按Q4格式存放（即储存值=温度×2^4，16进制）；采样AD值存放格式为12bit模式下右对齐值。

程序读出示例：

```
unsigned short Temp，TempAD0，TempAD1
Temp = ReadAddr[0x0FFFF070];      //校准时温度值，16bits，Q4格式，工作电压5V
TempAD0 = ReadAddr [0x0FFFF072]; //校准时存储的AD值 CHOP=0，16bits，12位AD采样值
TempAD1 = ReadAddr [0x0FFFF074]; //校准时存储的AD值 CHOP=1，16bits，12位AD采样值
```

温度传感器输出电压随温度线性变化，但不同芯片的温度特性曲线会有不同，尽管提供了上述校准方法，但实际温度仍然会受不同芯片的不同温度特性差异，以及芯片本身功耗差异造成温升差异等因素影响，因此温度传感器更适合检测温度的变化，而不是测量绝对温度。如需要测量精确的绝对温度，应该使用专用的外置温度传感器。

24.4 寄存器

TPS 模块寄存器列表 (基地址: 0x4002 2C00)

地址	寄存器名	说明
0x4002 2C04	AVREFCON	模拟模块内建Vref控制寄存器
0x4002 2C08	TPSCON	温度传感器控制寄存器

24.4.1 模拟模块内建 Vref 控制寄存器 (TPS_AVREFCON)

偏移地址: 0x0004

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留												CHOPEN	CHOPS	VREFSEL	VREFEN
-												*RW*	*RW*	*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 4	保留	-
3	**CHOPEN**	**内建VREF的CHOP功能使能位：** 0：关闭 1：使能 注：自动实现CHOP校正功能
2	**CHOPS**	**内建VREF的CHOP关闭时，CHOP内部OP的输入极性：** 0：正极 1：负极
1	**VREFSEL**	**内建电压输出选择位** 0：选择2.5V为内建VREF 1：选择3.3V为内建VREF
0	**VREFEN**	**内建电压开启位** 0：停止内建VREF 1：开启内建VREF 注：基准源开启需要建立时间，最长不超过100us，因此当需要用到VREF时，需要提前100us将此标志位置1，VREF引脚需接4.7uF陶瓷电容

24.4.2 温度传感器控制寄存器 (TPS_TPSCON)

偏移地址: 0x0008

复位值: 0x0000 0000

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
保留															
-															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留														TPSCHOP	TPSEN
-														*RW*	*RW*
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	符号	说明
31 ~ 2	保留	-
1	TPSCHOP	温度传感器CHOP开关 0：温度传感器CHOP功能关闭 1：温度传感器CHOP功能开启
0	TPSEN	温度传感器开关 0：关闭温度传感器 1：开启温度传感器 注：温度传感器从关闭到开启，有一个建立过程，从TPSEN置为1开始，至少等待100us再开始AD转换读取转换结果

25. 器件电子签名

25.1 产品唯一身份标识寄存器（96 位）

产品唯一的身份标识非常适合：

- 用来作为序列号(例如产品序列号满足终端应用)
- 用来作为密码，在编写闪存时，将此唯一标识与软件加解密算法结合使用，提高代码在闪存存储器内的安全性。
- 用来激活带安全机制的自举过程

96 位的产品唯一身份标识所提供的参考号码对任意一个 SH32F9001_32F9801 微控制器，在任何情况下都是唯一的。用户在何种情况下，都不能修改这个身份标识。

这个 96 位的产品唯一身份标识，按照用户不同的用法，可以以字节(8 位)为单位读取，也可以以半字(16 位)或者全字(32 位)。

26. 调试接口

26.1 主要特性

SH32F9001_32F9801 使用 Cortex™-M3 内核，该内核内含硬件调试模块，支持复杂的调试操作。硬件调试模块允许内核在取指(指令断点)或访问数据(数据断点)时停止。内核停止时，内核的内部状态和系统的外部状态都是可以查询的。完成查询后，内核和外设可以被复原，程序将继续执行。

SH32F9001_32F9801 支持 SWJ 调试接口，当 SH32F9001_32F9801 微控制器连接到调试器并开始调试时，调试器将使用内核的硬件调试模块进行调试操作。

ARM Cortex™-M3 内核提供集成的片上调试功能。它由以下部分组成：

- SWJ-DP：串行/JTAG 调试端口
- AHP-AP： AHB 访问端口
- ITM：执行跟踪单元
- FPB：闪存指令断点
- DWT：数据触发
- TPUI：跟踪单元接口(仅较大封装的芯片支持)

26.2 ARM 参考文献

- Cortex™-M3 技术参考手册(TRM)
- ARM 调试接口 V5.2
- ARM CoreSight 开发工具集技术参考手册

26.3 SWJ 调试端口(Serial Wire)

SH32F9001_32F9801 内核集成了串行调试接口(SWJ-DP)，这是标准的 ARM CoreSight 调试接口。

- 串行调试接口(SW-DP)为 AHP-AP 模块提供 2 针(时钟＋数据)接口。

26.4 SWJ 调试端口引脚

SH32F9001_32F9801 的 3 个普通 I/O 口可用作 SWJ-DP 接口引脚。这些引脚在所有的封装里都存在。

表26-1 SWJ 调试端口引脚

SWJ-DP 端口引脚名称	SW 调试接口		引脚分配
	类型	调试功能	
SWDIO	输入/输出	串行数据输入/输出	PA4
SWCLK	输入	串行时钟	PA5
SWO	-	-	PA6

26.5 SWJ-DP 引脚分配

复位(SYSRESETn 或 POWRESETn)以后，属于 SWJ-DP 的所有 3 个引脚都立即被初始化为可被调试器使用的专用引脚(注意，并没有初始化跟踪输出脚，除非调试器对此脚进行定义)。

然而， SH32F9001_32F9801 微控制器可以用复用重映像和调试 I/O 配置寄存器(**SYSCFG_SAFR**)寄存器来禁止 SWJ-DP 接口的部分或所有引脚的功能，这些专用引脚将被释放以用作普通 I/O 口。此寄存器被映射到和 Cortex™-M3 系统总线相连接的 APB 桥上。对此寄存器的设置将由用户代码而不是调试器完成。

1 个控制位用来配置 SWJ-DP 接口的引脚，这 1 个位在系统复位时复位。

- **SYSCFG_SAFR** (SH32F9001_32F9801 微控制器中的地址是 0x4004 3808)

  ─ 读： APB，无等待状态

  ─ 写： APB，如果 AHB-APB 桥的写缓冲器满了，则一个等待状态位 26=SWJCFG 由软件置位和复位.

  这 1 位用来设置分配给 SWJ 调试接口的专用引脚数目，目的是在使用不同的调试接口时能释放尽可能多的引脚用作普通 I/O 口。

26.6 利用串行接口并释放不用的调试脚作为普通 I/O 口

为了利用串行调试接口来释放一些普通 I/O 口，用户软件必须在复位后设置 SWJCFG@ SAFR，从而释放部分或全部调试口用做普通 I/O 口。此控制位一旦设置，只有 Reset 才能修改。

对于用户软件设计，应注意在复位后，这些专用引脚仍然处于带上拉的输入(SWDIO)，带下拉的输入(SWCLK)，和输出(SWO)状态，并持续一段时间，直到用户代码释放这些引脚。

当这些引脚仍被配置成调试接口时(SW)，修改相应的普通 I/O 口配置寄存器是无效的。

表26-2 SWJ_DP 引脚分配

SWJCFG	配置为调试专用的引脚	SWJ 接口的 I/O 口分配			SW 调试接口是否可用
		PA4 /SWDIO	PA5 /SWCLK	PA6 /SWO	
0	所有的 SWJ 引脚(SW-DP)复位状态	调试专用	调试专用	调试专用	可用
1	SW-DP 接口禁止	用户可用	用户可用	用户可用	可用

26.7 MCU 调试模块

MCU 调试模块协助调试器提供以下功能：

- 低功耗模式
- 在断点时提供 PWM0-3、PCA0-7、TIM5-8、UART0-5、SPI0/1、TWI0/1、DMA、IWDT、WWDT、LED、LCD 的时钟控制

26.7.1 低功耗模式的调试支持

使用 WFI 和 WFE 可以进入低功耗模式。

MCU 支持多种低功耗模式，分别可以关闭 CPU 时钟，或降低 CPU 的能耗。内核不允许在调试期间关闭 FCLK 或 HCLK。这些时钟对于调试操作是必要的，因此在调试期间，它们必须工作。 MCU 使用一种特殊的方式，允许用户在低功耗模式下调试代码。为实现这一功能，调试器必须先设置一些配置寄存器来改变低功耗模式的特性。

- 在睡眠模式下，调试器必须先置位 DBG_SLEEP@DBGCR 寄存器。这将为 HCLK 提供与 FCLK(由代码配置的系统时钟)相同的时钟。
- 在停止模式下，调试器必须先置位 DBG_STOP@DBGCR 寄存器。这将激活内部 RC 振荡器，在停止模式下为 FCLK 和 HCLK 提供时钟。

26.7.2 带定时功能的模块调试支持

在产生断点时，有必要根据各模块的不同用途选择相应计数器的工作模式，对于 PWM0-3、PCA0-7、TIM5-8、UART0-5、SPI0/1、TWI0/1、DMA、IWDT、WWDT、LED、LCD 各个模块，在产生断点时，可选择各模块的计数器是否继续计数。

26.7.3 调试接口控制寄存器 (SYSCFG_DBGCR)

偏移地址: 0x0010

复位值: 0x0000 0020

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
LOCK[15:0]															
WO															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
保留		DBG_TWI	DBG_SPI	DBG_UART	DBG_PWM	DBG_TIM	DBG_PCA	DBG_WWDT	DBG_IWDT	DBG_DMA	DBG_LED	DBG_LCD	保留	DBG_STOP	DBG_SLEEP
-		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	-	RW	RW
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

位	符号	说明
31 ~ 16	**LOCK[15:0]**	**此寄存器操作解锁位 (Control bit Lock register)** 这些位是只写，读取这些位返回0x0000。 0x5AA5：解锁 其他: 锁定 注：写入此寄存器需配合解锁控制，读此寄存器不需要解锁。
15 ~ 14	保留	-
13	**DBG_TWI**	**当内核进入调试状态时TWI状态控制** 0：TWI停止工作 1：TWI仍然正常工作
12	**DBG_SPI**	**当内核进入调试状态时SPI状态控制** 0：SPI停止工作 1：SPI仍然正常工作
11	**DBG_UART**	**当内核进入调试状态时UART状态控制** 0：UART停止工作 1：UART仍然正常工作
10	**DBG_PWM**	**当内核进入调试状态时PWM状态控制** 0：PWM停止工作 1：PWM仍然正常工作
9	**DBG_TIM**	**当内核进入调试状态时TIM5-TIM8模块状态控制** 0：TIM5-TIM8模块停止工作 1：TIM5-TIM8模块仍然正常工作
8	**DBG_PCA**	**当内核进入调试状态时PCA状态控制** 0：PCA停止工作 1：PCA仍然正常工作
7	**DBG_WWDT**	**当内核进入调试状态时调试窗口看门狗状态控制** 0：窗口看门狗计数器停止工作 1：窗口看门狗计数器仍然正常工作
6	**DBG_IWDT**	**当内核进入调试状态时看门狗状态控制** 0：看门狗计数器停止工作 1：看门狗计数器仍然正常工作
5	**DBG_DMA**	**当内核进入调试状态时DMA状态控制** 0：DMA停止工作 1：DMA仍然正常工作
4	**DBG_LED**	**当内核进入调试状态时LED状态控制** 0：LED停止工作 1：LED仍然正常工作
3	**DBG_LCD**	**当内核进入调试状态时LCD状态控制** 0：LCD停止工作 1：LCD仍然正常工作
2	保留	-
1	**DBG_STOP**	**调试停机模式** 0：(FCLK开，HCLK开)在停机模式时，FCLK和HCLK时钟由内部RC振荡器提供。当退出停机模式时，软件必需重新配置时钟系统启动PLL，晶振等 1：(FCLK关，HCLK关)在停机模式时，时钟控制器禁止一切时钟(包括HCLK和FCLK)。当从停机模式退出时，时钟的配置和复位之后的配置一样(微控制器由8MHz的内部RC振荡器(HSI)提供时钟)。因此，软件必需重新配置时钟控制系统启动PLL，晶振等
0	**DBG_SLEEP**	**调试睡眠模式** 0：(FCLK开，HCLK开)在睡眠模式时，FCLK和HCLK时钟都由原先配置好的系统时钟提供。 1：(FCLK开，HCLK关)在睡眠模式时，FCLK由原先已配置好的系统时钟提供，HCLK则关闭。由于睡眠模式不会复位已配置好的时钟系统，因此从睡眠模式退出时，软件不需要重新配置时钟系统。

27. 用户代码选项

FLASH_BACKUP：

0：FLASH区不分区块（默认）

1：FLASH区分为两块，Main Memory以及Backup Memory

FLASH_SELECT：（只有当FLASH_BACKUP为1时有效）

0：PC指针从Main Memory区取址（默认）

1：PC指针从Backup Memory区取址

28. 电气特性

极限参数*

数字供电电压. . . . . –0.3V to +6.5V
模拟供电电压. . . . . –0.3V to +6.5V
数字输入/输出电压. . . . . GND-0.3V to VDD+0.3V
模拟输入电压. . . . . AGND-0.3V to AVDD +0.3V
工作环境温度. . . . . .-40℃ to +105℃
存储温度. . . . . .–55℃ to +125℃
FLASH存储器写/擦除操作 . . . . . -40℃ to +105℃

***注释**

如果器件的工作条件超过左列“极限参数”的范围，将造成器件永久性破坏。只有当器件工作在说明书所规定的范围内时功能才能得到保障。器件在极限参数列举的条件下工作将会影响到器件工作的可靠性。

直流电气特性 (VDD = 2.0 - 5.5V, AVDD = VDD, GND = AGND = 0V, TA = 25°C, 除非另有说明)

参数	符号	最小值	典型值	最大值	单位	条件
数字工作电压	V_{DD}	2.0	5.0	5.5	V	4MHz ≤ f_{sys} ≤ 72MHz
模拟工作电压	AV_{DD}	VDD-0.1	VDD	VDD+0.1	V	
工作电流	I_{OP1}	-	15	40	mA	f_{sys} = 72MHz，PLL开，V_{DD}= AV_{DD} =5.0V，使用晶振（8MHz）。所有输出引脚无负载，所有输入引脚不浮动 CPU打开(执行本地循环跳转)，IWDT打开，LVR打开，关闭其它所有功能
	I_{OP2}	-	3	10	mA	f_{sys} = 24MHz，PLL关，V_{DD}= AV_{DD} =5.0V，使用内部RC振荡器。所有输出引脚无负载，所有输入引脚不浮动 CPU打开(执行本地循环跳转)，IWDT打开，LVR打开，关闭其它所有功能
	I_{OP3}	-	400	600	μA	f_{sys} = 32kHz，PLL关，高频振荡器关闭，V_{DD}= AV_{DD} =5.0V，使用晶振（32.768kHz）。所有输出引脚无负载，所有输入引脚不浮动 CPU打开(执行本地循环跳转)，IWDT打开，LVR打开，关闭其它所有功能
待机电流 (睡眠模式)	I_{SB1}	-	7	18	mA	f_{sys} = 72MHz，PLL开，V_{DD}= AV_{DD} =5.0V，使用晶振（8MHz）。所有输出引脚无负载，所有输入引脚不浮动 CPU停止(执行WFI进入sleep)，IWDT打开，LVR打开，关闭其它所有功能
	I_{SB2}	-	2	7	mA	f_{sys} = 24MHz，PLL关，V_{DD}= AV_{DD} =5.0V，使用内部RC振荡器。所有输出引脚无负载，所有输入引脚不浮动 CPU停止(执行WFI进入sleep)，IWDT打开，LVR打开，关闭其它所有功能
	I_{SB3}	-	400	600	μA	f_{sys} = 32kHz，PLL关，高频振荡器关闭，V_{DD}= AV_{DD} =5.0V，使用晶振（32.768kHz）。所有输出引脚无负载，所有输入引脚不浮动 CPU停止(执行WFI进入sleep)，IWDT打开，LVR打开，关闭其它所有功能
待机电流 (停机模式)	I_{SB4}	-	12	15	μA	高频振荡器关闭，PLL关，V_{DD}= AV_{DD} =5.0V，所有输出引脚无负载，所有输入引脚不浮动 CPU停止(执行WFI进入stop)，IWDT打开，LVR关闭，关闭其它所有功能
	I_{SB5}	-	12	18	μA	f_{sys} = 32kHz，高频振荡器关闭，PLL关 V_{DD}= AV_{DD} =5.0V，所有输出引脚无负载，所有输入引脚不浮动

						CPU停止(执行WFI进入stop)，IWDT打开，LVR关闭，关闭其它所有功能
LVR电流	I_{LVR1}	-	1	3	μA	LVR打开，LVR电平 = 4.1V/3.7V/2.8V/2.3V
IWDT 电流	I_{IWDT}	-	-	1	uA	所有输出引脚无负载；独立看门狗打开V_{DD} =5.0V
输入低电压1	V_{IL1}	GND		0.3 X V_{DD}	V	I/O 端口（所有端口都是施密特输入） V_{DD} = 3.0 - 5.5V，窗口不小于0.15VDD
		GND		0.2 X V_{DD}	V	I/O端口（所有端口都是施密特输入） V_{DD} = 2.0 – 3.0V，窗口不小于0.2VDD
输入高电压1	V_{IH1}	0.6 X V_{DD}	-	V_{DD}	V	I/O 端口（所有端口都是施密特输入） V_{DD} = 3.0 - 5.5V，窗口不小于0.15VDD
		0.7 X V_{DD}	-	V_{DD}	V	I/O端口（所有端口都是施密特输入） V_{DD} = 2.0 – 3.0V，窗口不小于0.2VDD
输入低电压2	V_{IL2}	GND	-	0.2 X V_{DD}		仅SWDIO、SWCLK 引脚 V_{DD} = 2.0 – 5.5V
输入高电压2	V_{IH2}	0.8 X V_{DD}	-	V_{DD}		仅SWDIO、SWCLK 引脚 V_{DD} = 2.0 – 5.5V
输入低电压3	V_{IL3}	GND	-	0.8	V	I/O端口（所有端口都是TTL输入） V_{DD} = 4.5 - 5.5V，TTL输入功能打开（Note2）
		GND	-	0.15 X V_{DD}	V	I/O端口（所有端口都是TTL输入） V_{DD} = 2.0 - 4.5V，TTL输入功能打开（Note2）
输入高电压3	V_{IH3}	2.0	-	V_{DD}	V	I/O端口（所有端口都是TTL输入） V_{DD} = 4.5 - 5.5V，TTL输入功能打开（Note2）
		0.25 X V_{DD}+0.8	-	V_{DD}	V	I/O端口（所有端口都是TTL输入） V_{DD} = 2.0 - 4.5V，TTL输入功能打开（Note2）
输入漏电流	I_{IL}	-1		1	uA	输入无上拉, V_{IN}= V_{DD} or GND
上拉/下拉电阻	R_{PH1}	15	30	45	kΩ	V_{DD} =5.0V, V_{IN}=GND
IO口输出电流1	I_{OH1}	-10	-12	-	mA	I/O端口, V_{OH} = 4.3V, V_{DD} = 5.0V， GPIOx_ODRVR = 0b (Note3) (对于PB4~PB15，选择GPIOB_ODRVR = 00b档位) (翻转速率小于500kHz)
	I_{OH1MAX}(1)	-	-	-20	mA	I/O端口, V_{DD} = 5.0V，驱动能力档位选择同上
IO口输出电流2	I_{OH2}	-1	-1.5	-	mA	I/O端口, V_{OH} = 4.3V, V_{DD} = 5.0V GPIOx_ODRVR = 1b (Note3) (对于PB4~PB15，选择GPIOB_ODRVR = 01b档位) (翻转速率小于1MHz)
	I_{OH2MAX}(1)	-	-	-3	mA	I/O端口, V_{DD} = 5.0V，驱动能力档位选择同上
IO口灌电流1	I_{OL1}	15	20	-	mA	I/O 端口, V_{OL} = 0.6V, V_{DD} = 5.0V GPIOx_ODRVR = 0b (Note3) (对于PB4~PB15，选择GPIOB_ODRVR = 00b档位) (翻转速率小于500kHz)
	I_{OL1MAX}(1)	-	-	30	mA	I/O端口, V_{DD} = 5.0V，驱动能力档位选择同上
IO口灌电流2	I_{OL2}	3.5	6	-	mA	I/O 端口, V_{OL} = 0.6V, V_{DD} = 5.0V GPIOx_ODRVR = 1b (Note3) (对于PB4~PB15，选择GPIOB_ODRVR = 01b档位) (翻转速率小于1MHz)
	I_{OL2MAX}(1)	-	-	8		I/O端口, V_{DD} = 5.0V，驱动能力档位选择同上
IO大驱动口灌电流能力	I_{OL3}	88	100	-	mA	I/O端口, V_{DD} = 5.0V，V_{OL} = 0.5V GPIOx_ODRVR = 10b (Note3) 仅PB4~PB15 (翻转速率1kHz)

	I_{OL3MAX}(1)	-	-	250	mA	I/O端口, V_{DD} = 5.0V，驱动能力档位选择同上
大驱动口拉电流能力	I_{OH3}	-88	-100	-	mA	I/O端口，V_{DD} = 5.0V，V_{OH} = 4.2V GPIOB_ODRVR = 11b (Note3) 仅PB4~PB15 (翻转速率1kHz)
	I_{OH3MAX}(1)	-	-	-200	mA	I/O端口, V_{DD} = 5.0V，驱动能力档位选择同上
VDD电流	I_{VDD1}	-	-	-200	mA	V_{DD} = 5.0V
GND电流	I_{VSS1}	-	-	250	mA	V_{DD} = 5.0V

注意：

(1). 设计保证，实际测试时未包含温度，量产不做测试。

(2). TTL输入功能参考GPIO模块的GPIOx_CFG_TTLEN寄存器定义。

(3). 驱动能力选择参考GPIO模块的GPIOx_CFG_ODRVR寄存器定义。

高速12BIT模/数转换器电气特性（1LSB = AV_{DD}/4096）

转换速率为12KSPS～1MSPS。（AV_{DD} = 2.7V～5.5V，AGND = 0V，T_A = +25°C，除非另有说明）

参数	符号	最小值	典型值	最大值	单位	条件
工作电压范围	V_{AD}	2.7	5.0	5.5	V	
精度	N_R	-	12	-	bit	V_{REF} = 5.0V
A/D输入电压	V_{AIN}	GND	-	V_{REF}	V	
A/D输入电阻*	R_{AIN}	2	-	-	MΩ	V_{IN} = 5.0V
外部模拟参考电压	V_{REF}	2.0	-	AV_{DD}	V	
模拟电压源推荐阻抗	Z_{AIN}	-	-	5	kΩ	
A/D转换电流	I_{AD}	-	1.5	3	mA	ADC模块工作，AV_{DD} = 5.0V
微分非线性误差	D_{LE}	-	-	±1	LSB	AV_{DD} = 5.0V，V_{REF} = 5.0V
积分非线性误差	I_{LE}	-	-	±3	LSB	AV_{DD} = 5.0V，V_{REF} = 5.0V
满刻度误差	E_F	-	-	±7	LSB	AV_{DD} = 5.0V，V_{REF} = 5.0V
偏移量误差	E_Z	-	-	±7	LSB	AV_{DD} = 5.0V，V_{REF} = 5.0V
总绝对误差	E_{AD}	-	-	±8	LSB	AV_{DD} = 5.0V，V_{REF} = 5.0V
ADC时钟周期	t_{AD}	0.050	-	10.88	μs	
ADC采样时间	t_{SAMP}	0.100	-	10.88	μs	
总转换时间	T_{CON}	16	-	159	t_{AD}	（$1t_{AD}$ ～ $16t_{AD}$） + t_{GAP} + $t_{CONVERT}$

注意：

(1) ""表示ADC输入电阻就是直流条件下ADC自身的输入电阻。*

(2) 外部参考源阻抗R_{REF}大于30Ω时，建议加uF级电容，提高驱动能力。

温度传感器特性（AV_{DD} = 2.7 - 5.5V，AGND = 0V，T_A = +25°C，除非另有说明）

参数	符号	最小值	典型值	最大值	单位	条件
工作温度*	T	-40		150	℃	
温度传感器线性度	T_L		±3	±5	℃	AV_{DD} = 5V , T_A = -40 ~ +150°C
温度传感器斜率	T_{Slope}		5.153		mV/℃	
25度时输出电压	V_{25}		1.5		V	

注意：

1. *“*”温度范围在105°C以上由设计保证，实际不测试。*
2. *AV_{DD}小于2.7V时，不保证温度传感器的功能。*

内部电压基准源特性（AVDD = 2.7 - 5.5V，AGND = 0V，TA = +25°C，除非另有说明），VREF引脚接4.7uF陶瓷电容

参数	符号	最小值	典型值	最大值	单位	条件
基准源输出精度	V_O	2.488	2.5	2.512	V	AV_{DD} = 5.0V，开启2.5V参考电压输出，T_A = 25°C
		3.284	3.3	3.316	V	AV_{DD} = 5.0V，开启3.3V参考电压输出，T_A = 25°C
负载调整率	ΔV_{O1}	-	0.05	0.14	%	AV_{DD} = 5.0V，V_O = 2.5V or 3.3V，T_A = 25°C I_{LOAD}= 0 ~ 2mA
输入电压调整率	ΔV_{O2}	-	0.1	0.2	%	AV_{DD} = 3.0~ 5.5V，V_O = 2.5V，T_A = 25°C ILOAD=1mA
		-	0.1	0.2	%	AV_{DD} = 3.8~ 5.5V，V_O = 3.3V，T_A = 25°C ILOAD=1mA
参考源输出温度系数	TCV_O	-	50	300	ppm/°C	AV_{DD} =5.0V，V_O = 2.5V or 3.3V，T_A = -40 ~ +105°C（Note1）

注意：

1. *设计保证，实际测试时未包含温度。*
2. *AVDD电压若低于所选输出电源+0.4V，不保证基准源输出的精度。*

交流电气特性（V_{DD} = 2.0 - 5.5V，GND = 0V，T_A = +25°C，除非另有说明）

参数	符号	最小值	典型值	最大值	单位	条件
复位脉冲宽度	t_{RESET}	10	-	-	μs	低电平有效
复位引脚上拉电阻	R_{RPH}	15	30	45	kΩ	V_{DD} = 5.0V，V_{IN} = GND
频率稳定性 (HSI)	|Δ F1|/F1	-	-	0.3	%	内部高频RC振荡器频率精度:（1024个周期求平均值） |F_{RC1} – 24MHz|/24MHz X 100% （V_{DD} = 2.0 – 5.5V，T_A = 25°C）
	|Δ F1|/F1	-	-	0.5	%	内部高频RC振荡器频率精度:（1024个周期求平均值） |F_{RC1} – 24MHz|/24MHz X 100% （V_{DD} = 2.0 - 5.5V，T_A = -10°C至60°C， 设计保证，不在生产中测试）
	|Δ F1|/F1	-	-	1.0	%	内部高频RC振荡器频率精度:（1024个周期求平均值） |F_{RC1} – 24MHz|/24MHz X 100% （V_{DD} = 2.0 - 5.5V，T_A = -40°C至+105°C， 设计保证，不在生产中测试）
频率稳定性 (LSI)	|Δ F2|/F2	-	-	4.0	%	内部低频RC振荡器频率精度:（1024个周期求平均值） |F_{RC2} – 128kHz|/128kHz X 100% （V_{DD} = 2.0 – 5.5V，T_A = 25°C）
	|Δ F2|/F2	-	-	7.0	%	内部低频RC振荡器频率精度:（1024个周期求平均值） |F_{RC2} – 128kHz|/128kHz X 100% （V_{DD} = 2.0 - 5.5V，T_A = -10°C至60°C， 设计保证，不在生产中测试）
	|Δ F2|/F2	-	-	15.0	%	内部低频RC振荡器频率精度:（1024个周期求平均值） |F_{RC2} – 128kHz|/128kHz X 100% （V_{DD} = 2.0 - 5.5V，T_A = -40°C至+105°C， 设计保证，不在生产中测试）

PLL特性（V_{DD} = 2.0 - 5.5V，GND = 0V，T_A = +25°C，除非另有说明）

参数	符号	最小值	典型值	最大值	单位	条件
PLL输入时钟频率	f_{PLL_IN}	4	-	16	MHz	
PLL输入时钟Duty	D_{PLL_IN}	40	-	60	%	
PLL输出时钟频率	f_{PLL_OUT}			72	MHz	
PLL建立时间	T_{PLL}		2		ms	
PLL频率稳定性			±0.05	±0.1	%	仅限于HSE作为PLL的时钟源 （1024个周期求平均值 V_{DD} = 2.4 - 5.5V，T_A = 25°C）

掉电检测（BOD）电气特性（V_{DD} = 2.0 - 5.5V，GND = 0V，T_A = +25°C，除非另有说明）

参数	符号	最小值	典型值	最大值	单位	条件
BOD电压范围	V_{BOD}	2.4	-	4.3	V	
BOD门限等级	|ΔV_{BOD}|	-	150	-	mV	V_{BOD} = 2.4V～4.3V
BOD电压检测迟滞窗口*	V_{SMTBD1}	-	50	-	mV	V_{BOD} = 2.4V～4.3V
BOD去抖动时间	T_{BOD}	40	-	80	μs	

低电压复位（LVR）电气特性（V_{DD} = 2.0 - 5.5V，GND = 0V，T_A = +25°C，除非另有说明）

参数	符号	最小值	典型值	最大值	单位	条件
LVR电压1*	V_{LVR1}	3.95	4.1	4.25	V	LVR1使能
LVR电压2*	V_{LVR2}	3.55	3.7	3.85	V	LVR2使能
LVR电压3	V_{LVR3}	2.7	2.8	2.9	V	LVR3使能
LVR电压4	V_{LVR4}	2.2	2.3	2.4	V	LVR4使能
LVR电压检测迟滞窗口	V_{SMTLV}	-	80	-	mv	
LVR低电压复位宽度	T_{LVR}	40	-	80	μs	

上电复位电气特性（VDD = 2.0 - 5.5V，GND = 0V，TA = 25°C，除非另有说明）

参数	符号	最小值	典型值	最大值	单位	条件
电源上电复位电压	V_{POR}	1.82	1.90	1.98	V	
电源电压上升斜率	S_{VDD} (1)	0.005	-	1000	V/ms	LVR使能，V_{DD} = 2.0V - 5.5V

注意：

(1). 该项目只是设计保证，量产不做全斜率范围测试。

ESD特性

参数	符号	测试值	单位	条件
人体放电模型HBM	V_{HBM}	4000	V	TA = 25 °C，符合JEDEC JS-001-2017标准
充电器件模型CDM	V_{CDM}	1000	V	TA = 25 °C，符合JEDEC JS-002-2018标准
机器模型MM	V_{MM}	200	V	TA = 25 °C，符合JESD22-A115C标准
LatchUp电流	I_{LU}	200	mA	TA = 25 °C，符合JESD78E标准

(1)基于特性，不在生产中测试

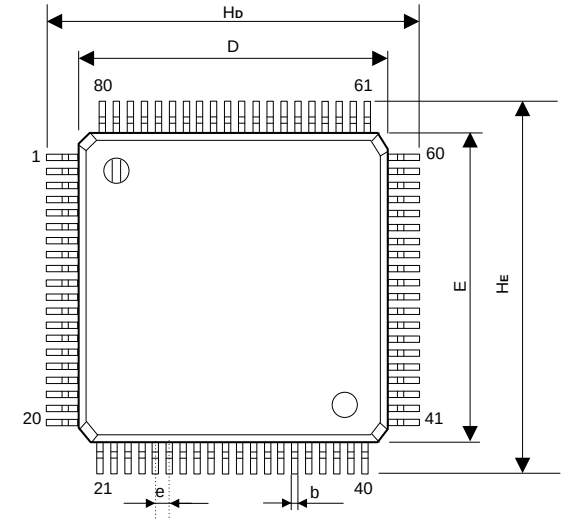
EFT特性

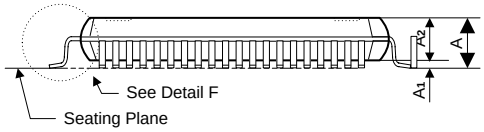
参数	符号	等级/类型	条件
EFT	V_{EFT}	4A	VDD=5.0V，TA = 25 °C，符合GB/T 17626.4-2008标准

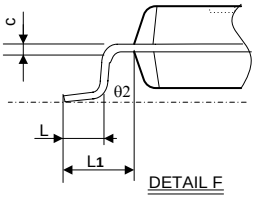
(1)基于特性，不在生产中测试

29. 封装信息

LQFP80（14×14）外形尺寸 **单位: 英寸/毫米**





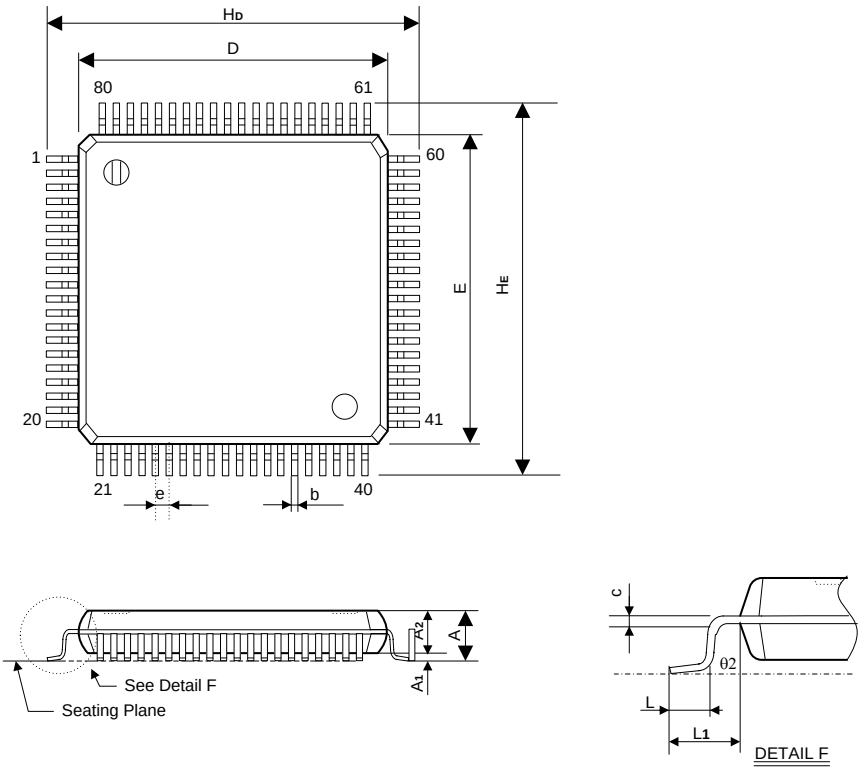


Symbol	Dimensions in inches		Dimensions in mm	
	MIN	MAX	MIN	MAX
A	---	0.063	---	1.6
A1	0.002	0.006	0.05	0.15
A2	0.053	0.057	1.35	1.45
D	0.547	0.555	13.90	14.10
E	0.547	0.555	13.90	14.10
H_D	0.622	0.938	15.80	16.20
H_E	0.622	0.938	15.80	16.20
b	0.010	0.015	0.26	0.38
e	0.026		0.65BSC	
c	0.004	0.008	0.09	0.20
L	0.018	0.030	0.45	0.75
L1	0.033	0.045	0.85	1.15
θ2	0°	10°	0°	10°

注意:

1. 封装尺寸不包括模的毛边凸起或门毛刺。
2. 如无特殊规定，容差为±0.1毫米。
3. 共面性：0.1毫米。
4. 控制尺寸为毫米。

LQFP80（12×12）外形尺寸 **单位: 英寸/毫米**

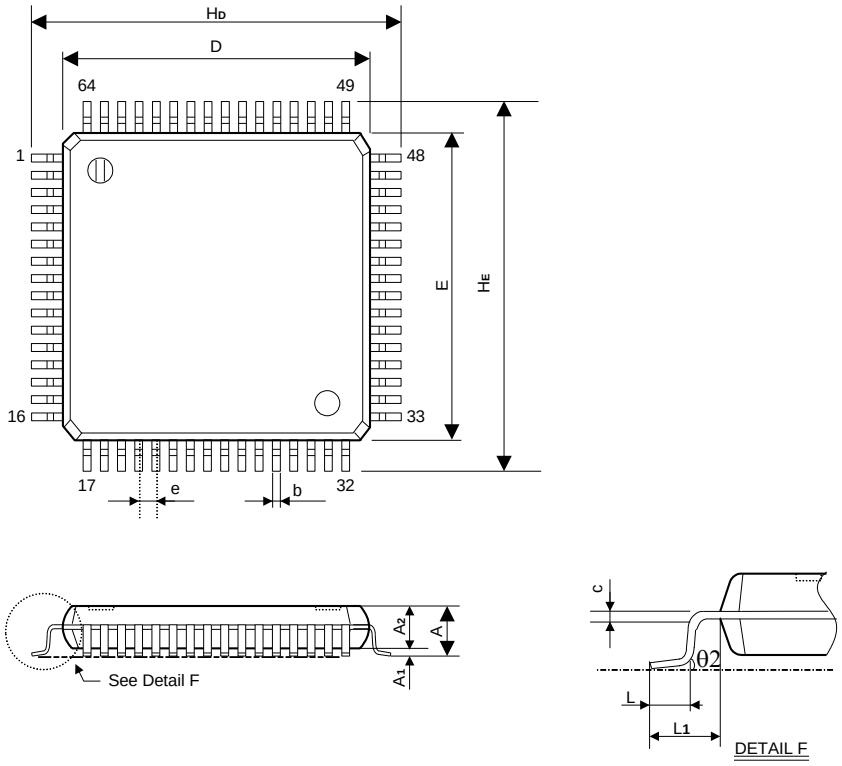


Symbol	Dimensions in inches		Dimensions in mm	
	MIN	MAX	MIN	MAX
A	---	0.063	---	1.60
A1	0.002	0.006	0.05	0.15
A2	0.053	0.057	1.35	1.45
D	0.467	0.478	11.85	12.15
E	0.467	0.478	11.85	12.15
H_D	0.543	0.559	13.80	14.20
H_E	0.543	0.559	13.80	14.20
b	0.007	0.011	0.17	0.27
e	0.020BSC		0.50BSC	
c	0.004	0.008	0.09	0.20
L	0.018	0.030	0.45	0.75
L1	0.033	0.045	0.85	1.15
θ2	0°	10°	0°	10°

注意:

1. 封装尺寸不包括模的毛边凸起或门毛刺。
2. 如无特殊规定，容差为±0.1毫米。
3. 共面性：0.1毫米。
4. 控制尺寸为毫米。

LQFP64（12×12）外形尺寸 **单位: 英寸/毫米**



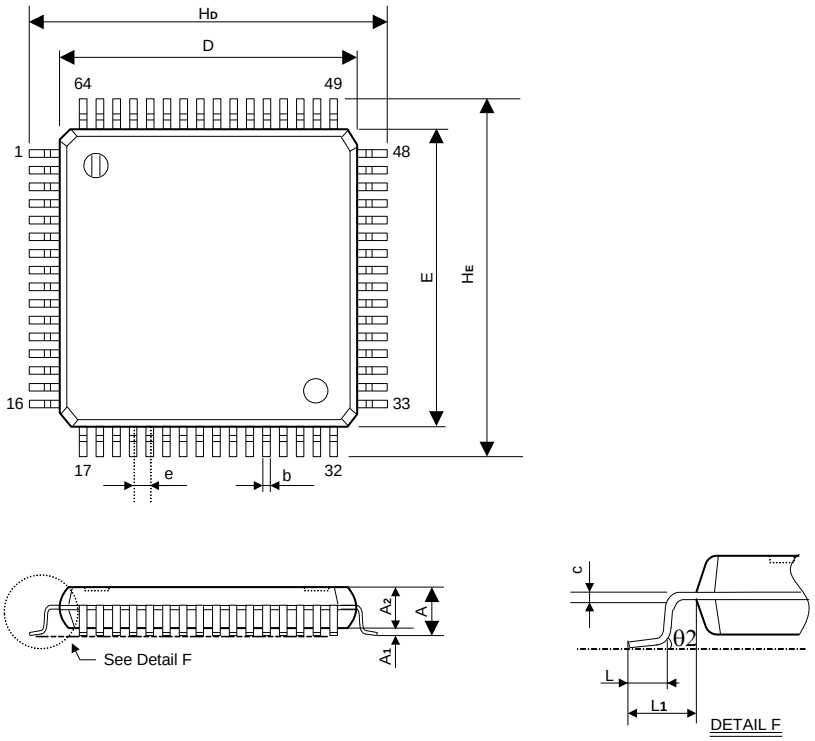
Symbol	Dimensions in inches		Dimensions in mm	
	MIN	MAX	MIN	MAX
A	---	0.063	---	1.6
A1	0.002	0.006	0.05	0.15
A2	0.053	0.057	1.35	1.45
D	0.468	0.476	11.9	12.1
E	0.468	0.476	11.9	12.1
H_D	0.543	0.560	13.8	14.2
H_E	0.543	0.560	13.8	14.2
b	0.010	0.015	0.26	0.38
e	0.026BSC		0.65BSC	
c	0.005	0.007	0.13	0.18
L	0.018	0.030	0.45	0.75
L1	0.033	0.045	0.85	1.15
θ2	0°	10°	0°	10°

注意:

1. 封装尺寸不包括模的毛边凸起或门毛刺。
2. 如无特殊规定，容差为±0.1毫米。
3. 共面性：0.1毫米。
4. 控制尺寸为毫米。

LQFP64（10×10）外形尺寸

单位: 英寸/毫米



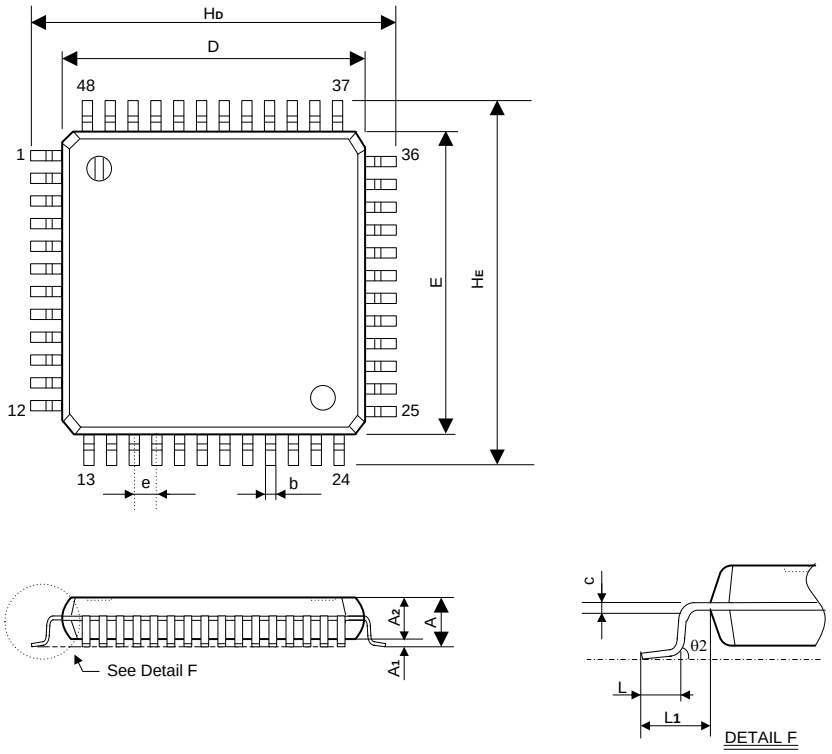
Symbol	Dimensions in inches		Dimensions in mm	
	MIN	MAX	MIN	MAX
A	---	0.063	---	1.6
A1	0.002	0.006	0.05	0.15
A2	0.053	0.057	1.35	1.45
D	0.388	0.400	9.85	10.15
E	0.388	0.400	9.85	10.15
H_D	0.465	0.480	11.8	12.2
H_E	0.465	0.480	11.8	12.2
b	0.007	0.011	0.17	0.27
e	0.016BSC		0.50BSC	
c	0.004	0.008	0.09	0.20
L	0.018	0.030	0.45	0.75
L1	0.033	0.045	0.850	1.150
θ2	0°	10°	0°	10°

注意:

1. 封装尺寸不包括模的毛边凸起或门毛刺。
2. 如无特殊规定，容差为±0.1毫米。
3. 共面性：0.1毫米。
4. 控制尺寸为毫米。

TQFP48外形尺寸

单位: 英寸/毫米



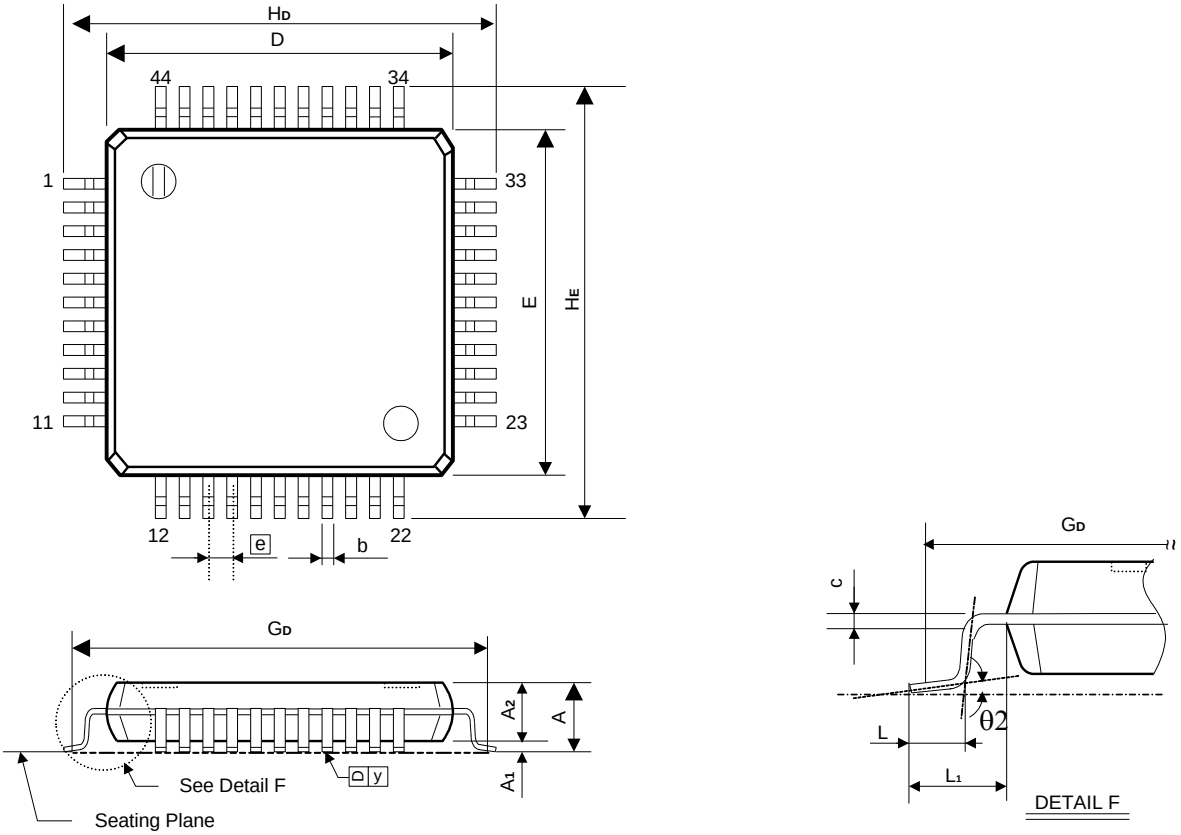
Symbol	Dimensions in inches		Dimensions in mm	
	MIN	MAX	MIN	MAX
A	---	0.047	---	1.2
A1	0.002	0.006	0.05	0.15
A2	0.035	0.041	0.9	1.05
D	0.270	0.281	6.85	7.15
E	0.270	0.281	6.85	7.15
H_D	0.346	0.362	8.8	9.2
H_E	0.346	0.362	8.8	9.2
b	0.005	0.011	0.15	0.27
e	0.020 TYP		0.500 TYP	
c	0.004	0.008	0.090	0.200
L	0.018	0.030	0.45	0.75
L1	0.033	0.045	0.85	1.15
θ2	0°	10°	0°	10°

注意:

1. 封装尺寸不包括模的毛边凸起或门毛刺。
2. 如无特殊规定，容差为±0.1毫米。
3. 共面性：0.1毫米。
4. 控制尺寸为毫米。

LQFP44外形尺寸

单位: 英寸/毫米



Symbol	Dimensions in inches		Dimensions in mm	
	MIN	MAX	MIN	MAX
A	0.057	0.065	1.45	1.65
A1	0.000	0.001	0.01	0.21
A2	0.051	0.059	1.3	1.5
D	0.388	0.400	9.85	10.15
E	0.388	0.400	9.85	10.15
H_D	0.465	0.480	11.8	12.2
H_E	0.465	0.480	11.8	12.2
b	0.010	0.018	0.25	0.45
e	0.031 TYP		0.8 TYP	
c	0.004	0.008	0.09	0.20
L	0.017	0.031	0.42	0.78
L1	0.037	0.045	0.95	1.15
θ2	0°	10°	0°	10°

注意:

1. 封装尺寸不包括模的毛边凸起或门毛刺。
2. 如无特殊规定，容差为±0.1毫米。
3. 共面性：0.1毫米。
4. 控制尺寸为毫米。

30. 规格更改记录

版本	记录	日期
2.5	添加定时器8开启后在PCLK0时钟2分频情况下关闭的注意事项 PCA单斜坡、双斜坡更改为锯齿波、三角波； 添加ESD和EFT特性； 修改笔误。	2025年6月
2.4	GPIO章节寄存器页GPIOx_IOD更改为GPIOx； 增加FLASH读取时钟等待状态设置； HSE驱动能力选择位，推荐更改为Normal档； BOD不能唤醒Stop模式； 修改程序区编程/擦除次数笔误，更改为100000次； 增加电源上电复位电压参数； 修改电气特性章节IO驱动能力； 当TIM5溢出作为PCA2-7时钟源时，TIM5只能选择内部APB0时钟。	2022年10月
2.3	产品信息增加SH32F9001Y/080YR； 增加HSE驱动能力选择位，推荐使用Strong档； 增加LSE振荡器扰动使能位； 增加32.768kHz晶体谐振器加速模式； 增加外部高频晶振/陶振驱动电路反馈电阻选择 (HSERFB)，推荐使用500kΩ。	2021年11月
2.2	更改LQFP44封装（删除引脚LCD）； 系统启动流程图中前256字节更改为前1024字节； 在系统编程ISP更改为出厂不烧写任何程序，需要客户自行开发； 增加Flash操作ME0、Protect区编程命令字和BootRom区编程命令字； 电气特性FLASH存储器写/擦除操作上限温度由85℃更改为105℃； LVREN、IWDTPD、WWDTON位开启后关闭无效，只有在复位时才能关闭； 更改DMA请求映像章节SPI_MOSI为SPI_TX，SPI_MISO为SPI_RX。	2021年10月
2.1	产品信息增加SH32F9801P/080PR； AD转换时间固定由14t_{AD}更改为15t_{AD}。	2021年4月
2.0	IWDT和TIMER 使用的128KHz RC命名为128KHz WDT RC； 更改UART奇偶校验选择位：0奇校验、1偶校验。	2021年1月
1.0	初始版本	2020年12月

重要声明

本手册为中颖电子股份有限公司及其关联公司（“公司”）的财产。本手册，包括本手册中描述的本公司的任何产品（“产品”），均为本公司根据相关可适用法律或条约所拥有。本公司保留该等法律和条约下的所有权利，不授予其专利、版权、商标或其他知识产权下的任何许可。

本手册内的任何技术信息，包括功能介绍和原理图，不应理解为使用或执行任何知识产权的许可。本手册若引述相关第三方的名称和品牌（如有）等为其各自所有者的财产，仅供识别用途。

本公司不对本手册或任何产品作任何明示或暗示的保证，包括但不限于对适销性和适合特定用途的暗示保证。本公司不承担因超出规格书或我司产品标准的保证范围使用本手册所述任何产品而产生的任何责任。除适用协议中明确规定的定制产品外，产品仅为普通商业、工业、个人和/或家庭应用而设计、开发和制造。禁止用于军事、国防、核能、医疗以及可能导致人身伤害、死亡，或是环境破坏等领域。用户应采取任何和所有行动，确保按照适用的法律法规使用和销售产品。

半导体产品自身存在一定的失效概率。为防止因故障或误工作而产生的人身损害、火灾事故或其他社会性损害，请注意冗余设计、消防设计以及其他安全防护设计。特别说明：参考应用电路不保证能够适用于特定应用的量产。

若用户违反上述声明，本公司不承担全部或部分责任，用户应在此免除本公司及其供应商和/或分销商因产品的所有非预期用途有关的任何索赔、损害或其他责任；与此同时，用户应赔偿并确保本公司及其供应商和/或分销商免受因产品的任何非预期用途的使用而产生的与之相关的所有索赔、成本、损害赔偿和其他责任，包括人身伤害或死亡的索赔。

本手册中的信息仅与产品有关。本公司保留随时对本手册及所述的产品和服务进行更改、修改或改进的权利，恕不另行通知。订购前建议用户咨询销售代表。

本公司对本手册拥有最终解释权。

31. 目录

1. 特性 1
2. 概述 2
3. 方框图 3
4. 引脚定义 4
5. 引脚描述 9
6. 产品信息 11
7. 内核 13
7.1 ARM®的CortexTM-M3核心 13
7.2 寄存器组 13
8. 存储器和总线架构 14
8.1 系统架构 14
8.2 存储器组织 15
8.2.1 寄存器映像 16
8.2.2 数据存储器1（SRAM） 17
8.2.3 位带 17
8.2.4 程序存储器 17
8.2.5 数据存储器2（CRAM） 17
8.3 Flash程序存储器 18
8.3.1 特性 18
8.3.2 简介 18
8.3.3 Flash存储器功能描述 19
8.4 启动配置（BootLoader） 28
8.5 在系统编程（ISP） 30
8.6 Flash访问加速控制 30
8.7 Flash切分区块功能 30
8.8 寄存器 32
8.8.1 Flash访问控制寄存器 (FLASH_ACR) 32
8.8.2 Flash主程序存储区解锁控制寄存器(FLASH_MKYR) 33
8.8.3 类EEPROM存储区解锁控制寄存器 (FLASH_E2KYR) 33
8.8.4 Flash状态和清除寄存器 (FLASH_SR) 34
8.8.5 Flash控制寄存器 (FLASH_CR) 35
8.8.6 Flash读保护寄存器 (FLASH_RPR) 37
8.8.7 Flash写保护寄存器 (FLASH_WRPR) 37
8.8.8 Flash操作定时器计数寄存器 (FLASH_CNTR) 37
8.8.9 Flash操作定时器上限值 (FLASH_UPCNTR) 38
8.8.10 Flash操作定时器启动寄存器 (FLASH_CNTCR) 38
8.8.11 Flash特殊信息存储区解锁控制寄存器 (FLASH_IKYR) 39
8.8.12 内存映射选择寄存器 (FLASH_MEMRMP) 39
9. 电源控制（PWR） 40
9.1 电源与时钟预热 40
9.2 掉电检测（BOD） 41

9.3 低电压复位（LVR）.....42
9.4 低功耗管理.....42
9.4.1 睡眠模式.....43
9.4.2 停机模式.....44
9.4.3 低功耗模式下的调试.....45
9.5 寄存器.....45
9.5.1 电源控制寄存器 (SYSCFG_PWRCR).....46
9.5.2 电源状态寄存器 (SYSCFG_PWRSR).....47
10. 复位和时钟控制（RCC）.....48
10.1 复位.....48
10.1.1 系统复位.....48
10.1.2 电源复位.....49
10.2 时钟.....50
10.2.1 HSE时钟.....51
10.2.2 LSE时钟.....51
10.2.3 HSI时钟.....52
10.2.4 LSI时钟.....52
10.2.5 WDT时钟.....52
10.2.6 PLL时钟.....52
10.2.7 系统时钟(SYSCLK)选择.....53
10.2.8 时钟安全监控（CSM）.....53
10.2.9 系统节拍定时器（SysTick）.....57
10.2.10 HSI修整功能（HSITRIM）.....57
10.3 寄存器.....58
10.3.1 时钟控制寄存器 (RCC_CR).....58
10.3.2 时钟配置寄存器 (RCC_CFGR).....60
10.3.3 时钟中断使能寄存器 (RCC_CIENR).....62
10.3.4 时钟中断状态寄存器 (RCC_CISTR).....63
10.3.5 时钟中断标志清除寄存器 (RCC_CICLR).....64
10.3.6 AHB外设复位寄存器 (RCC_AHBRSTR).....65
10.3.7 APB1外设复位寄存器 (RCC_APB1RSTR).....66
10.3.8 APB0外设复位寄存器 (RCC_APB0RSTR).....67
10.3.9 AHB外设时钟使能寄存器 (RCC_AHBENR).....69
10.3.10 APB1外设时钟使能寄存器 (RCC_APB1ENR).....70
10.3.11 APB0外设时钟使能寄存器 (RCC_APB0ENR).....72
10.3.12 RESET状态寄存器 (RCC_RSTSTR).....74
10.3.13 清除RESET状态寄存器 (RCC_RSTCLR).....75
10.3.14 HSI振荡器校准寄存器 (RCC_HSICAL).....76
10.3.15 CSM档位选择 (RCC_CSMSEL).....77
10.3.16 RCC配置锁定寄存器 (RCC_RCCLOCK).....77
10.3.17 外部高频晶振/陶振驱动电路反馈电阻选择 (HSERFB).....78
11. 看门狗定时器（WDT）.....79
11.1 独立看门狗定时器（IWDT）.....79
11.2 窗口看门狗定时器（WWDT）.....79
11.3 寄存器.....80
11.3.1 独立看门狗控制寄存器 (IWDT_CR).....80

11.3.2 独立看门狗喂狗寄存器 (IWDT_CLR) 81
11.3.3 窗口看门狗控制寄存器 (WWDT_CR) 81
11.3.4 窗口看门狗状态寄存器 (WWDT_SR) 82
11.3.5 窗口看门狗喂狗寄存器 (WWDT_CLR) 82
11.3.6 窗口看门狗窗口寄存器 (WWDT_WTR) 83
12. 代码和数据检测（Code and Data Checking） 84
12.1 CRC模块（CRC） 84
12.1.1 CRC简介 84
12.1.2 CRC 主要特性 84
12.1.3 CRC 功能描述 84
12.1.4 CRC寄存器 87
12.2 SRAM检测模块（RAMBIST） 90
12.2.1 SRAM检测模块简介 90
12.2.2 SRAM检测模块功能描述 90
12.2.3 RAMBIST寄存器 93
13. 通用功能I/O (GPIO) 95
13.1 简介 95
13.2 主要特性 95
13.3 功能描述 96
13.3.1 I/O引脚的默认配置 97
13.3.2 I/O控制寄存器 97
13.3.3 I/O数据寄存器 97
13.3.4 I/O位置位和位清除 97
13.3.5 I/O锁定机制 98
13.3.6 I/O配置为外部中断/唤醒线 98
13.3.7 I/O复用功能 98
13.3.8 GPIO输入功能 99
13.3.9 GPIO输出功能 100
13.3.10 数字复用功能 100
13.3.11 模拟复用功能 101
13.3.12 I/O引脚的基本应用 102
13.4 寄存器 103
13.4.1 端口x配置锁定寄存器 (GPIOx_CFG_LCKR) 103
13.4.2 端口x输入使能配置寄存器 (GPIOx_CFG_IE) 104
13.4.3 端口x输出方式配置寄存器 (GPIOx_CFG_OTYPER) 104
13.4.4 端口x输出驱动能力配置寄存器 (GPIOx_CFG_ODRVR) 104
13.4.5 端口x上下拉配置寄存器 (GPIOx_CFG_PUPDR) 106
13.4.6 端口x部分引脚TTL电平选择寄存器 (GPIOx_CFG_TTLEN) 109
13.4.7 端口x复用功能寄存器低位(GPIOx_CFG_AFRL) 110
13.4.8 端口x复用功能寄存器高位 (GPIOx_CFG_AFRH) 114
13.4.9 端口x模式寄存器 (GPIOx_MODER) 117
13.4.10 端口x输入数据寄存器 (GPIOx_IDR) 117
13.4.11 端口x输出数据寄存器 (GPIOx_ODR) 118
13.4.12 端口x位清除/置位寄存器 (GPIOx_BSRR) 118
14. 系统配置模块(SYSCFG) 119
14.1 系统配置相关 119

14.1.1 SWJ和XTAL引脚复用功能定义 119
14.1.2 CRAM锁定选项 119
14.1.3 调试接口寄存器 119

14.2 寄存器 119
14.2.1 电源控制寄存器 (SYSCFG_PWRCR) 119
14.2.2 电源状态寄存器 (SYSCFG_PWRSR) 121
14.2.3 系统配置寄存器 (SYSCFG_SAFR) 122
14.2.4 CRAM扇区锁定控制寄存器 (SYSCFG_CRAMLOCK) 123
14.2.5 调试接口控制寄存器 (SYSCFG_DBGCR) 123
15. 中断和事件 125

15.1 可嵌套中断向量控制器（NVIC） 125
15.1.1 简介 125
15.1.2 主要特性 125
15.1.3 中断汇总 125

15.2 外部中断/事件控制器（EXTI） 129
15.2.1 主要特性 129
15.2.2 唤醒事件管理 130
15.2.3 功能说明 130
15.2.4 外部中断/事件线路映像 131

15.3 寄存器 132
15.3.1 中断屏蔽寄存器 (EXTI_IMR) 132
15.3.2 事件屏蔽寄存器 (EXTI_EMR) 132
15.3.3 触发模式选择寄存器 (EXTI_TMSR) 133
15.3.4 上升沿/高电平触发模式选择寄存器 (EXTI_RTSR) 133
15.3.5 下降沿/低电平触发模式选择寄存器 (EXTI_FTSR) 133
15.3.6 软件中断寄存器 (EXTI_SWIER) 134
15.3.7 挂起寄存器 (EXTI_PR) 134
15.3.8 外部中断配置低寄存器 (EXTI_CFGL) 135
15.3.9 外部中断配置高寄存器 (EXTI_CFGH) 137
15.3.10 EXTI采样控制低寄存器 (EXTI_SAMPL) 139
15.3.11 EXTI采样控制高寄存器 (EXTI_SAMPH) 141
15.3.12 DMA屏蔽寄存器 (EXTI_DMR) 143
16. DMA控制器 144

16.1 简介 144

16.2 主要特性 144

16.3 功能描述 145
16.3.1 DMA处理 145
16.3.2 仲裁器 145
16.3.3 DMA 通道 146
16.3.4 可编程的数据传输宽度、对齐方式和数据大小端 149
16.3.5 错误管理 150
16.3.6 DMA中断 150
16.3.7 DMA请求映像 151

16.4 寄存器 153
16.4.1 DMA中断状态寄存器 (DMA_IFSR) 154
16.4.2 DMA中断标志清除寄存器 (DMA_IFCR) 154

16.4.3 DMA控制状态寄存器 (DMA_CSR) 155
16.4.4 DMA通道n配置寄存器 (DMA_CCRn)(n=0..7) 156
16.4.5 DMA通道n传输数量寄存器 (DMA_NPKTn)(n=0..7) 157
16.4.6 DMA通道n传输计数寄存器 (DMA_CPKTn)(n=0..7) 158
16.4.7 DMA通道n源地址寄存器 (DMA_SARn)(n=0..7) 158
16.4.8 DMA通道n目标地址寄存器 (DMA_DARn)(n=0..7) 159
17. LED驱动器 160
17.1 简介 160
17.2 LED RAM配置 164
17.3 寄存器 166
17.3.1 LED控制寄存器 (LED_CR) 166
17.3.2 LED中断标志和清除寄存器 (LED_FR) 167
17.3.3 LED的IO SHARE控制寄存器 (LED_LEDSHARE) 168
17.3.4 LED数据寄存器 (LED_LED_BUFS0) 168
17.3.5 LED数据寄存器 (LED_LED_BUFS1) 169
17.3.6 LED数据寄存器 (LED_LED_BUFS2) 169
17.3.7 LED数据寄存器 (LED_LED_BUFS3) 169
17.3.8 LED数据寄存器 (LED_LED_BUFS4) 170
17.3.9 LED数据寄存器 (LED_LED_BUFS5) 170
17.3.10 LED数据寄存器 (LED_LED_BUFS6) 171
17.3.11 LED数据寄存器 (LED_LED_BUFS7) 171
17.3.12 LED数据寄存器 (LED_LED_BUFS8) 172
17.3.13 LED数据寄存器 (LED_LED_BUFS9) 172
18. LCD驱动器 173
18.1 简介 173
18.2 LCD RAM配置 176
18.3 寄存器 177
18.3.1 LCD控制寄存器 (LCD_CR) 177
18.3.2 LCD的IO SHARE控制寄存器 (LCD_LCDSHARE) 179
18.3.3 LCD的IO SHARE控制寄存器 (LCD_LCDSHARE1) 179
18.3.4 LCD数据寄存器 (LCD_LCD_BUFS0) 180
18.3.5 LCD数据寄存器 (LCD_LCD_BUFS1) 180
18.3.6 LCD数据寄存器 (LCD_LCD_BUFS2) 181
18.3.7 LCD数据寄存器 (LCD_LCD_BUFS3) 181
18.3.8 LCD数据寄存器 (LCD_LCD_BUFS4) 182
18.3.9 LCD数据寄存器 (LCD_LCD_BUFS5) 182
18.3.10 LCD数据寄存器 (LCD_LCD_BUFS6) 183
18.3.11 LCD数据寄存器 (LCD_LCD_BUFS7) 183
18.3.12 LCD数据寄存器 (LCD_LCD_BUFS8) 184
18.3.13 LCD数据寄存器 (LCD_LCD_BUFS9) 184
19. 基本定时器（Basic Timer，TIM5～TIM8） 186
19.1 简介 186
19.2 功能 186
19.2.1 计数器/定时器运行 186
19.2.2 级联运行 187

19.3 寄存器 188
19.3.1 TIM控制寄存器 (TIMx_CR) 189
19.3.2 TIM计数寄存器 (TIMx_TCNT) 190
19.3.3 TIM周期寄存器 (TIMx_TPR) 190
19.3.4 TIM预分频寄存器 (TIMx_PSQ) 190
19.3.5 TIMx中断标志和清除寄存器 (TIMx_TIMINTF) 191
20. 16bit脉冲宽度调制（PWM0/1/2/3） 192
20.1 特性 192
20.2 PWM使能控制 192
20.3 PWM保护寄存器 192
20.4 16位PWM定时器 192
20.5 PWMxB互补输出（x = 0,1,2,3） 193
20.6 死区时间 193
20.7 寄存器 194
20.7.1 PWM控制寄存器 (PWMx_CR) 194
20.7.2 PWM保护寄存器 (PWMx_PWMLOCK) 196
20.7.3 PWM周期寄存器 (PWMx_PWMPR) 196
20.7.4 PWM占空比寄存器 (PWMx_PWMDR) 196
20.7.5 PWM死区寄存器 (PWMx_PWMDTR) 197
20.7.6 PWM中断标志和清除寄存器 (PWMx_PWMINTF) 197
21. 可编程计数器阵列（PCAx（x=0~7）） 198
21.1 简介 198
21.2 主要特性 198
21.3 时钟配置 198
21.4 工作原理 199
21.4.1 PCAx模式配置 200
21.4.2 Mode0：边沿触发的捕捉模式 201
21.4.3 Mode1：比较匹配输出（软件定时器）方式 201
21.4.4 Mode2：频率输出方式 202
21.4.5 Mode3：PWM模式 203
21.4.6 Mode4：PCA级联运行 207
21.5 寄存器 208
21.5.1 PCA控制寄存器 (PCAx_CR) 208
21.5.2 PCA方式配置寄存器 (PCAx_CMD) 209
21.5.3 CPM0：PCA 捕捉/比较寄存器 (PCAx_CPMn)(n=0..1) 209
21.5.4 FORCE强制输出控制寄存器 (PCAx_FORCE) 210
21.5.5 PCA计数最大值 (PCAx_TOPR) 211
21.5.6 通道n比较/捕捉模块寄存器 (PCAx_CPRn)(n=0..1) 211
21.5.7 PCA中断标志和清除寄存器 (PCAx_PCAINTF) 212
21.5.8 PCA控制寄存器 (PCAx_CR) 213
21.5.9 PCA方式配置寄存器 (PCAx_CMD) 214
21.5.10 CPM0：PCA 捕捉/比较寄存器 (PCAx_CPMn)(n=0..1) 214
21.5.11 FORCE强制输出控制寄存器 (PCAx_FORCE) 215
21.5.12 PCA计数最大值 (PCAx_TOPR) 216

21.5.13 通道0比较/捕捉模块寄存器 (PCAx_CPR0)....216
21.5.14 PCA通道1比较/捕捉模块寄存器 (PCAx_CPR1)....217
21.5.15 PCA中断标志和清除寄存器 (PCAx_PCAINTF)....217
22. 串行通信接口（UART/SPI/TWI）....219
22.1 通用异步收发器（UART）....219
22.1.1 简介....219
22.1.2 主要特性....219
22.1.3 功能描述....219
22.1.4 UART寄存器....229
22.2 串行外设接口（SPI）....234
22.2.1 简介....234
22.2.2 主要特性....234
22.2.3 功能描述....234
22.2.4 SPI寄存器....240
22.3 两线串行接口（TWI）....244
22.3.1 简介....244
22.3.2 主要特性....244
22.3.3 功能描述....244
22.3.4 TWI寄存器....259
23. 模数转换器（Analog to Digital Converter，ADC）....263
23.1 简介....263
23.2 主要特性....264
23.3 功能....264
23.3.1 单次转换方式....264
23.3.2 间断转换方式....265
23.3.3 连续转换方式....265
23.3.4 其它功能描述....266
23.3.5 比较功能....268
23.3.6 ADC转换时间设置....269
23.3.7 ADC模块参考电压的设置....269
23.3.8 ADC信道与IO口功能设置....269
23.3.9 DMA请求....269
23.3.10 AD转换时对传感器输出阻抗的要求....269
23.3.11 注意事项....270
23.4 ADC寄存器....271
23.4.1 ADC控制寄存器1 (ADC_ADCON1)....271
23.4.2 ADC控制寄存器2 (ADC_ADCON2)....272
23.4.3 ADC采样转换通道指针寄存器 (ADC_ADPCH)....273
23.4.4 ADC结果寄存器n (ADC_ADDRn)(n=0..7)....274
23.4.5 ADC比较控制寄存器 (ADC_ADCMPCON)....274
23.4.6 ADC上限比较寄存器 (ADC_ADDGT)....275
23.4.7 ADC下限比较寄存器 (ADC_ADDLT)....275
23.4.8 ADC通道选择寄存器0 (ADC_SEQCHSEL0)....276
23.4.9 ADC通道选择寄存器1 (ADC_SEQCHSEL1)....278
23.4.10 ADC通道GAP控制位 (ADC_ADGAPON)....280
23.4.11 ADC中断标志和清除寄存器 (ADC_ADINTF)....280

24. 片上模拟功能模块（Analog Module On Chip，AMOC） 282
24.1 简介 282
24.2 主要特性 282
24.3 温度传感器 282
24.3.1 功能介绍 282
24.3.2 温度传感器校准 282
24.4 寄存器 283
24.4.1 模拟模块内建Vref控制寄存器 (TPS_AVREFCON) 283
24.4.2 温度传感器控制寄存器 (TPS_TPSCON) 284
25. 器件电子签名 285
25.1 产品唯一身份标识寄存器（96位） 285
26. 调试接口 286
26.1 主要特性 286
26.2 ARM参考文献 286
26.3 SWJ调试端口(Serial Wire) 286
26.4 SWJ调试端口引脚 286
26.5 SWJ-DP引脚分配 286
26.6 利用串行接口并释放不用的调试脚作为普通I/O口 287
26.7 MCU调试模块 287
26.7.1 低功耗模式的调试支持 287
26.7.2 带定时功能的模块调试支持 287
26.7.3 调试接口控制寄存器 (SYSCFG_DBGCR) 287
27. 用户代码选项 289
28. 电气特性 290
29. 封装信息 296
30. 规格更改记录 302
31. 目录 304