相关文档

PT 官方：

《PT32L00x 数据手册》

《PT32F00x 数据手册》

ARM 官方：

《Cortex™-M0 技术参考手册》

下载地址：https://developer.arm.com/documentation/ddi0432/c

《Cortex-M0 设备通用用户指南》

下载地址：https://developer.arm.com/documentation/dui0497/a

《ARM 调试接口架构规范 V5》

下载地址：https://developer.arm.com/documentation/ihi0031/f

《ARM CoreSight 技术参考手册》

下载地址：https://developer.arm.com/documentation/ddi0486/b

《ARMv6-M 架构参考手册》

下载地址：https://developer.arm.com/documentation/ddi0419/e

开发工具官方：

《CMSIS-Core (Cortex-M) Version 5.5.0》

下载地址：https://www.keil.com/pack/doc/cmsis/Core/html/index.html

目录

文中的缩写

寄存器描述表中使用的缩写列表

在对寄存器的描述中使用了下列缩写：

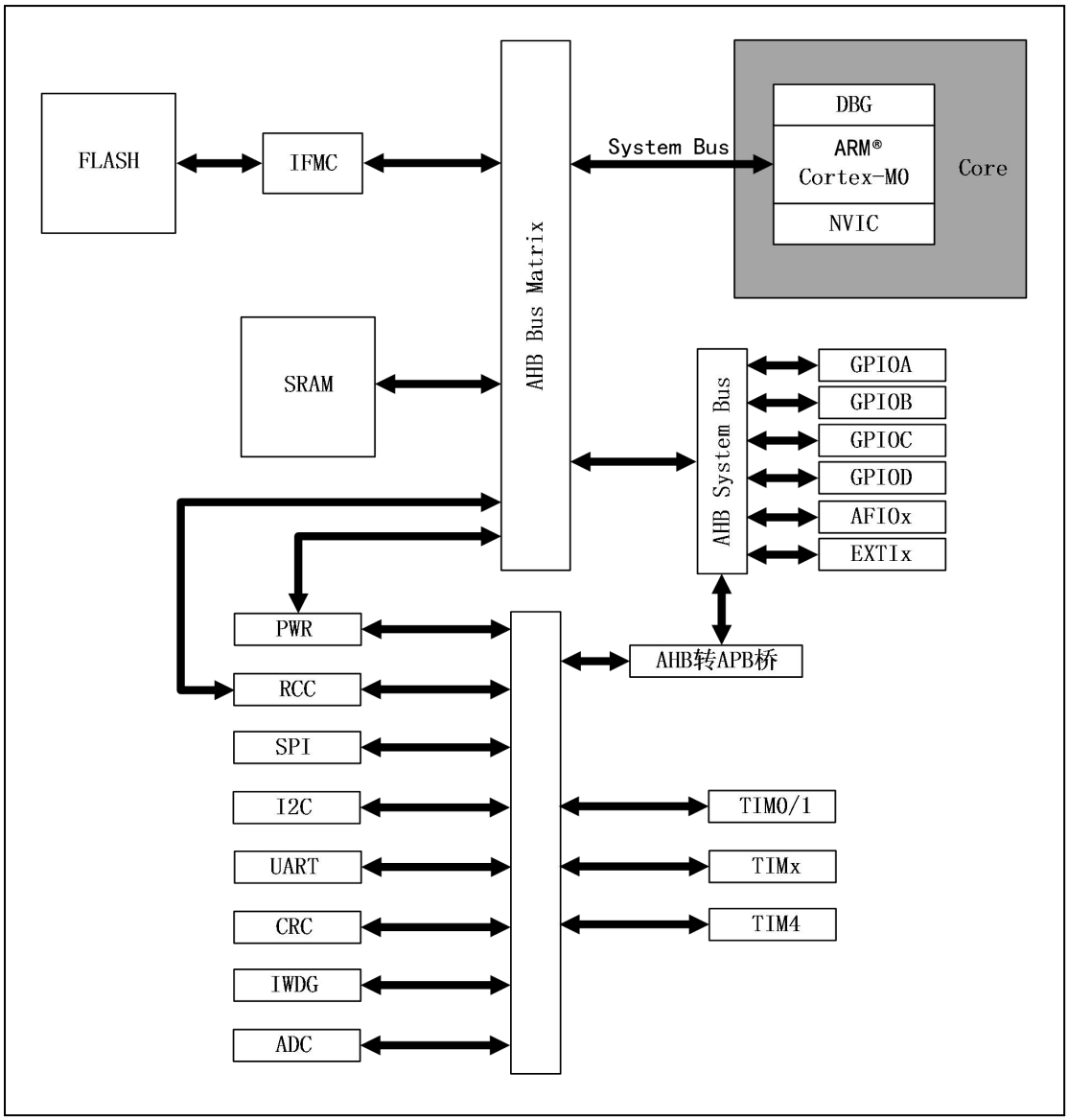
RW	read / write，软件能读写此位。
R	read-only，软件只能读此位。
W	write-only，软件只能写此位，读此位将返回复位值。
Rw1	read/clear，软件可以读此位，也可以通过写’1’清除此位，写’0’对此位无影响。
Rw0	read/clear，软件可以读此位，也可以通过写’0’清除此位，写’1’对此位无影响。
Res	Reserved，保留位，必须保持默认值不变
B	Byte
b	bit
UD	复位值 、User Define(用户定义)
MD	复位值 、Manufacturers Define(制造商定义)

1 存储器和总线构架

1.1 系统构架

- 一个驱动单元
  - Cortex™-M0 内核系统总线(S-bus)
- 四个被动单元
  - 内部 SRAM
  - 内部 Flash 闪存存储器
  - AHB 所连接的所有外设
  - AHB 到 APB 的桥，它连接的所有 APB 设备

图 1-1 系统结构



1.1.1 系统总线

此总线连接 Cortex™-M0 内核的系统总线(外设总线)到总线矩阵，总线矩阵协调内核与外部总线的访问。

1.1.2 AHB/APB 桥(APB)

“AHB/APB 桥”在 AHB 和 APB 总线间提供同步连接。AHB 和 APB 的操作速度均与系统时钟 SYS_CLK 同步。

连接到每个桥的不同外设的地址映射请参考表 1-1。在每一次复位过程当中，除 SRAM 以外的所有外设都被关闭。

注意：对 AHB 或 APB 总线上的寄存器进行 8 位或者 16 位操作时，由于数据字节以小端格式存放在存储器中，故有：

1. 读操作：该操作会被自动转换成 32 位的读，对应的数据将按照小端格式被存储在中间变量；

2. 写操作：总线和桥会自动将 8 位或者 16 位的写入数据扩展，并将缺失的高位补 0，以配合 32 位的向量。

1.2 存储器组织

程序存储器、数据存储器、寄存器和输入输出端口被组织在同一个 4GB 的线性地址空间内。

数据字节以小端格式存放在存储器中。一个字里的最低地址字节被认为是该字的最低有效字节，而最高地址字节是最高有效字节。

外设寄存器的映像请参考相关外设章节，下面列出芯片的具体地址分配。

表 1-1 AHB 地址分配

地址范围	大小	外设
0x0000_0000 – 0x0000_7FFF	32KB	指令 FLASH 空间
0x0000_8000 – 0x0000_F7FF	30KB(Max)	FLASH 主代码空间映射(1)
0x0000_F800 – 0x1FFF_FFFF		保留
0x2000_0000 – 0x2000_07FF	2KB	SRAM 数据区
0x2000_0800 – 0x3FFF_FFFF		保留
0x4000_0000 – 0x4001_7FFF	96KB	APB
0x4001_8000 – 0x4001_EFFF		保留
0x4001_F000 – 0x4001_FFFF	4KB	系统控制寄存器区
0x4002_0000 – 0x47FF_FFFF		保留
0x4800_0000 – 0x4800_0FFF	4KB	GPIOA 寄存器区
0x4800_1000 – 0x4800_1FFF	4KB	GPIOB 寄存器区
0x4800_2000 – 0x4800_2FFF	4KB	GPIOC 寄存器区
0x4800_3000 – 0x4800_3FFF	4KB	GPIOD 寄存器区
0x4800_4000 – 0xE000_DFFF		保留
0xE000_E000 – 0xE000_EFFF	4KB	内核控制寄存器区
0xE000_F000 – 0xFFFF_FFFF		保留
0xF000_0000 – 0xF000_0FFF	4KB	保留
0xF000_1000 – 0xFFFF_FFFF		保留

(1):仅在执行 bootloader 时有效，大小与 bootloader 大小相关，更多信息请参考”16 片内闪存控制器(IFMC)”

注意：访问非法的保留地址将导致 MCU 彻底失效，并且无法恢复。

表 1-2 APB 地址分配

地址范围	大小	外设
0x4000_0000 – 0x4000_07FF	2KB	IFMC
0x4000_0800 – 0x4000_0FFF	2KB	保留
0x4000_1000 – 0x4000_13FF	1KB	TIM2
0x4000_1400 – 0x4000_17FF	1KB	TIM3
0x4000_1800 – 0x4000_1BFF	1KB	外设占用(1)
0x4000 1C00 – 0x4000 1FFF	1KB	TIM4
0x4000_2000 – 0x4000_2FFF	4KB	保留
0x4000_3000 – 0x4000_37FF	2KB	IWDG
0x4000_3800 – 0x4000_3BFF	1KB	保留
0x4000_3C00 – 0x4000_3FFF	1KB	CRC
0x4000_4000 – 0x4000_43FF	1KB	保留
0x4000_4400 – 0x4000_47FF	1KB	UART0
0x4000_4800 – 0x4000_53FF	3KB	保留
0x4000_5400 – 0x4000_57FF	1KB	I2C
0x4000_5800 – 0x4000_5BFF	1KB	保留
0x4000_5C00 – 0x4001_23FF	50KB	保留
0x4001_2400 – 0x4001_27FF	1KB	ADC
0x4001_2800 – 0x4001_2BFF	1KB	保留
0x4001_2C00 – 0x4001_2FFF	1KB	TIM1
0x4001_3000 – 0x4001_33FF	1KB	SPI
0x4001_3400 – 0x4001_37FF	1KB	保留
0x4001_3800 – 0x4001_3BFF	1KB	UART1
0x4001_3C00 – 0x4001_3FFF	1KB	保留
0x4001_4000 – 0x4001_43FF	1KB	保留
0x4001_4400 – 0x4001_7FFF	15KB	保留

(1):该地址被分配到不同的外设，详见相关寄存器说明。

1.3 嵌入式 SRAM

PT32x00x 内置最大 2K 字节的 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。

1.4 嵌入式 Flash 闪存

PT32x00x 内置的闪存存储器可以用于在线编程(ICP)，在线编程(In-Circuit Programming –ICP)方式用于更新闪存存储器的全部内容，它通过 SWD 协议或系统加载程序(Bootloader)下载用户应用程序到微控制器中。ICP 是一种快速有效的编程方法，消除了封装和管座的困扰。

高性能的闪存模块有以下的主要特性：

最大 32K 字节的闪存存储器结构，下面罗列存储器的组成部分：

- 主程序区
- Bootloader 区
- 用户配置区

通过片内闪存控制器 IFMC 可以便捷的控制 Flash 闪存，有关 IFMC 的详细信息，请参考"16 片内闪存控制器(IFMC)"。

注意：PT32x00x 不支持中断向量表重映射，故不支持 IAP 应用。

1.5 启动配置

PT32x00x 支持两种启动模式：

- 从主程序区启动
- 从 Bootloader 区启动

这两种启动模式都基于 Flash 片内闪存，通过特定的程序配置以实现不同的启动模式，而无需外部硬件的介入。

注意：默认从主程序区启动，关于从 Bootloader 区启动的详细信息，请参考"16.3.6 系统启动配置"

2 CRC

2.1 综述

CRC 算法基于模 2 除法的原理，这需要占用一定的系统计算资源。
集成的硬件 CRC 是独立于系统的，它速度更快，更加灵活，且运行时并不占用系统计算资源,在一个时钟周期内，就可以完成 CRC 的计算。

2.2 特性

- 16 位的可编程的多项式，适用于多种不同的通信标准
- 16 位的可编程的种子(初始值)，适用于多种不同的通信标准
- 即算即得，单个 PCLK 的 CRC 计算时间
- 硬件反转，以支持不同类型的数据格式
  - 输入位序列反转
  - 输入字节序列反转
  - 输出位序列反转
- 独立的输入/输出数据寄存器

2.3 CRC 功能描述

CRC 即循环冗余校验，是一种数据传输检错功能，CRC 对一帧数据中的有效数据进行多项式计算，并将得到的结果附在这帧数据后面；它被广泛用于各种通信协议，常见的有 USB，Modbus 等等。

在实际通信中、通信的收发双方都约定了同一种 CRC 算法，如果数据帧在传输过程中没有被破坏，则双方通过 CRC 计算得出的结果将一致。

2.3.1 多项式

多项式用于生成有效数据的 CRC 校验码，又称为"生成多项式"。

收发双方约定了同一个多项式，利用这个多项式，对发送或接收的有效数据进行模 2 除法，以求出 CRC 校验码。

- 对于发送方，计算得出的 CRC 校验码被附在帧数据后面。
- 对于接收方，计算得出的 CRC 校验码会与帧数据之后的 CRC 校验码进行对比，当两者一致，才会接收本次数据。

常用的 CRC 多项式如下表所示：

表 2-1 常用的 CRC 多项式

名称	多项式	16 进制值	应用举例
CRC-16	$X^{16}+X^{15}+X^{2}+1$	0X18005	Modbus、USB
CRC-CCITT	$X^{16}+X^{12}+X^{5}+1$	0X11021	HDLC、PPP-FCS

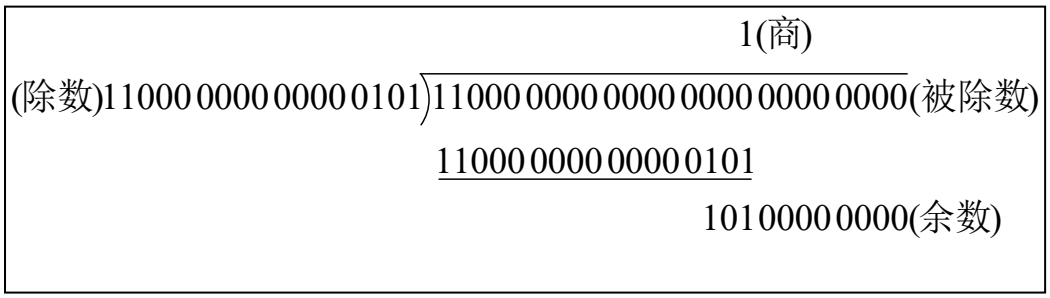
例如上表的 CRC-16，CRC 规定了多项式中有幂次为 1，没有幂次为 0，首尾一定要是 1，所以多项式 CRC-16 的二进制值为"1 1000 0000 0000 0101"，用 16 进制表示为 0X18005。

注意：由于最高位总是为 1，一般会将最高位忽略，但其本质是不变的。

示例，求有效数据"0x0180"的 CRC-16 结果：

- 0x0180 的二进制值为"1 1000 0000"
- CRC 为 16 位，0x0180 作为被除数，后面补上 16 个'0'："1 1000 0000 0000 0000 0000 0000"
- CRC-16 多项式的 16 进制值为 0x18005，作为除数，其二进制值为"1 1000 0000 0000 0101"

图 2-1 CRC-16 计算过程



最终的余数"101 0000 0000"就是 0x0180 的 CRC-16 结果，将其转换为 16 进制，结果为 0x500。

2.3.2 种子(初始值)

初始值的作用在于防止全'0'的数据 CRC 结果一直为 0，它执行一个"异或"的操作，对于有效数据：

- 当有效数据为'0'，初始值为'1'时，最终有效数据=0⊕1=1；
- 当有效数据为'1'，初始值为'1'时，最终有效数据=1⊕1=0；

合理的设置初始值有利于提高通信的鲁棒性，在一些标准的通信协议 CRC 中，对初始值也做出了相关要求。

注意：初始值的作用时机总是在反转功能作用之后。

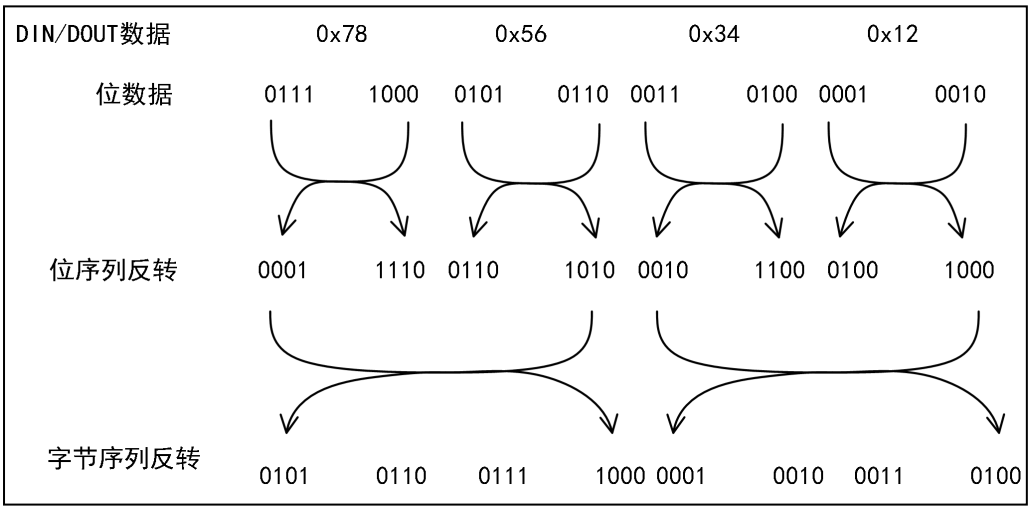
2.3.3 反转功能

CRC 模块的数据格式默认为 MSB(高位在前)，反转功能用于保证输入/输出数据格式间的兼容性：

- 对于输入：当输入数据为 LSB 格式时，可以使用输入位序列反转功能，将输入数据的格式转换成 MSB 格式。
- 对于输出：当输出数据需要为 LSB 格式时，可以使用输出位序列反转功能，将输出数据的格式转换成 LSB 格式。
- 特别地、当外部传入的数据是低字节在前，高字节在后的 16 位数据格式时，可以使用字节序列反转功能。

注意：当同时使能了输入字节序列反转功能和输入位序列反转功能时，输入字节序列反转功能被首先生效。

图 2-2　反转功能图



2.4 寄存器描述

2.4.1 CRC 控制寄存器(CRC_CR)

（地址： 0x4000_3C00）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	COSN	CBN	CISN	CIS	CRS	CEN
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	Rw1	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**CEN： CRC 模块使能控制(CRC enable)** 0： CRC 模块失能 1： CRC 模块使能
位 1	**CRS： CRC 复位控制(CRC reset control)** 0： 无效 1： 复位 CRC_DOUT 到默认状态
位 2	**CIS： CIS 输入选择(CRC input select)** 0： 8 位数据宽度 1： 16 位数据宽度
位 3	**CISN： 输入位序列反转(CRC input bit sequence reversal)** 无论 16 位和 8 位模式下，输入位序列反转只在字节内进行 0： 不反转 1： 反转
位 4	**CBN： 输入字节序列反转(CRC input byte sequence reversal)** 输入字节序列反转只对 16-bit 模式有效 0： 不反转 1： 反转
位 5	**COSN： 输出位序列反转(CRC output bit sequence reversal)** 输出位序列反转是对整个 16-bit 输出而言，而不是字节内部反转 0： 不反转 1： 反转
位 31：6	保留。必须保持为 0。

2.4.2 CRC 种子寄存器(CRC_SEED)

（地址： 0x4000_3C04）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	SEED[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位 15：0	**SEED[15:0]：CRC 种子(CRC seed)**
位 31：16	保留。必须保持为 0。

2.4.3 CRC 多项式寄存器(CRC_POLY)

（地址： 0x4000_3C08）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	POLY[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

位 15：0	**POLY[15:0]：CRC 多项式(CRC poly)** 0x8005： 表示 G(X) = X^16 + X^15 + X^2 + 1 0x1021： 表示 G(X) = X^16 + X^12 + X^5 + 1 其他： 保留 ***注：最高位的因子不需要写入寄存器***
位 31：16	保留。必须保持为 0。

2.4.4 CRC 数据输入寄存器(CRC_DIN)

（地址： 0x4000_3C0C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	DIN[15:0]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**DIN[15:0]：CRC 数据输入寄存器(CRC data input)** 该寄存器只写不可读，读恒为 0 ***注：*** ***1. 8 位模式下， 仅低 8 位有效*** ***2. 仿真模式下对该寄存器写入的数据无效***
位 31：16	保留。必须保持为 0。

2.4.5 CRC 数据输出寄存器(CRC_DOUT)

（地址： 0x4000_3C10）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	DOUT[15:0]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位 15：0	**DOUT[15:0]：CRC 数据输出寄存器(CRC data output)** 该寄存器只读不可写
位 31：16	保留。必须保持为 0。

2.4.6 寄存器列表

地址	寄存器	描述	备注
0x4000_3C00	CRC_CR	CRC 控制寄存器	CRC_CR 说明
0x4000_3C04	CRC_SEED	CRC 种子寄存器	CRC_SEED 说明
0x4000_3C08	CRC_POLY	CRC 多项式寄存器	CRC_POLY 说明
0x4000_3C0C	CRC_DIN	CRC 数据输入	CRC_DIN 说明
0x4000_3C10	CRC_DOUT	CRC 数据输出	CRC_DOUT 说明

3 电源控制(PWR)

3.1 综述

PWR 列举了芯片内部与电源相关的所有资源，包括内部的电源电压调节器、可编程的电源电压监测器、低功耗模式。

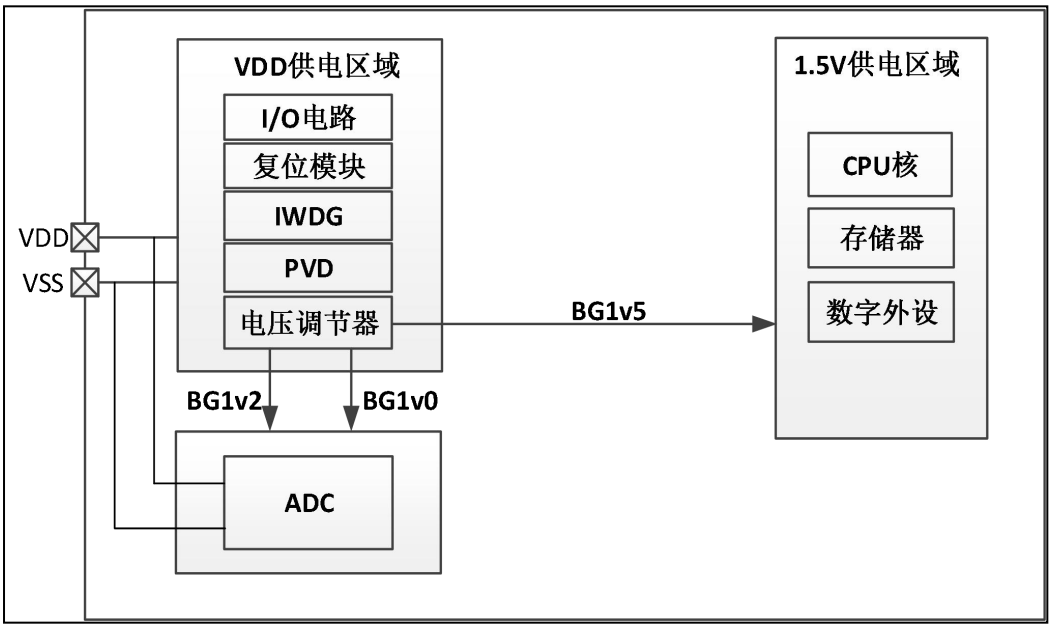
3.2 特性

- 集成的电源调节器，提供 3 路内部电源、均可通过 ADC 内部通道进行访问：
  - BG1v5 1.5V 精度 5%
  - BG1v2 1.2V 精度 5%
  - BG1v0 1.0V 精度 0.5%
- 多挡位、支持复位或中断的可编程电源电压检测器
- 在低功耗模式下，提供两种低功耗模式：
  - 睡眠模式
  - 深度睡眠模式
- 深度睡眠下、典型功耗仅为 0.8uA

3.3 电源调节器

PT32x00x 支持在超宽电压范围内工作、工作电压(VDD)详见《PT32x00x 数据手册》。
通过内置的电源调节器提供内部所需的 1.5V 电源 BG1v5、1.2V 参考电源 BG1v2 以及 1.0V 参考电源 BG1v0。
所有的内部电源均可通过 ADC 内部通道进行访问、详见 ADC(控制寄存器 ADC_CR)的"CHS"位描述。

图 3-1　电源框图



3.3.1 电源调节器

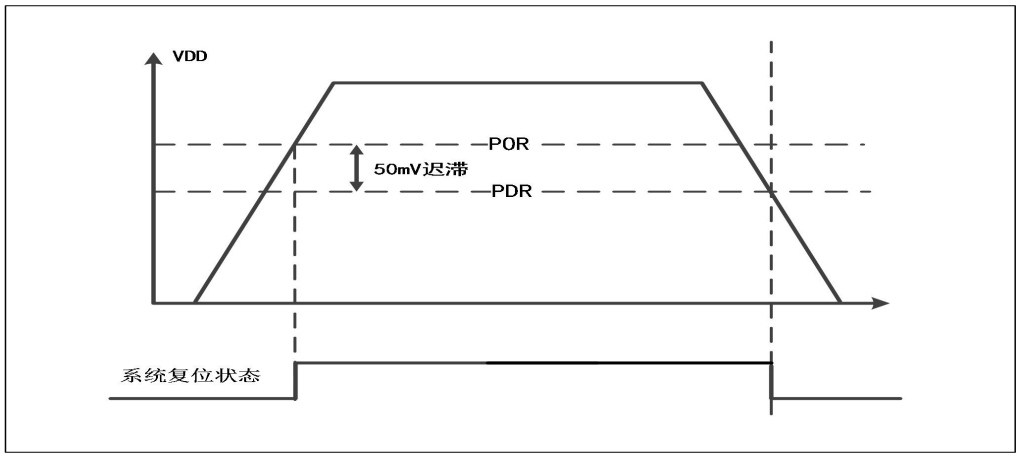
复位后电源调节器总是使能的。根据系统运行状态、它以 2 种不同的模式工作。
1.运行模式：调节器以正常功耗模式提供 BG1v5、BG1v2、BG1v0 三路电源；BG1v5 作为核心电源共给内部 CPU、存储器和数字外设，并且与 BG1v2、BG1v0 一起接入 ADC 的内部模拟通道中，如图 3-1 所示。
2.低功耗模式：提供睡眠和深度睡眠两种低功耗模式，电源调节器将递进的关闭片内电源、时钟、外设，以使系统功耗大大降低。

3.3.2 电源复位条件

PT32x00x 内部有一个完整的上电复位(POR)和掉电复位(PDR)电路，当 VDD 电压达到工作电压时系统就能正常工作。

当 VDD 低于指定的限位电压 V_{POR}/V_{PDR} 时，系统保持为复位状态。关于上电复位和掉电复位的更多细节请参考"4.3 复位功能描述"和《PT32x00x 数据手册》的电气特性部分。

图 3-2 上电复位和掉电复位的波形图



3.3.3 可编程的电源电压监测器(PVD)

用户可以利用 PVD 对 VDD 电压进行监测，(电源电压监测器配置寄存器 PWR_PVDR)中的"PLS"位用于设定 PVD 监控 VDD 电压的阈值。

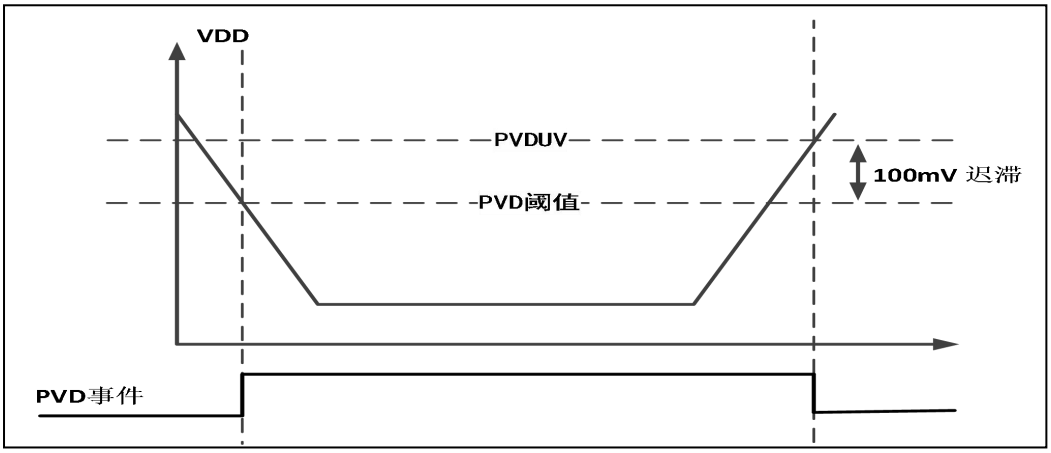
通过将(PWR_PVDR 寄存器)中的 PVDE 位置'1'以使能 PVD。

PVD 事件支持用户通过复位和中断两种形式进行访问，复位功能参考"4.3.3.1 电源低电压复位"的描述，其对应的中断事件在内部被连接到 NVIC 的第 20 号中断，如果该中断在系统中断使能寄存器 ISER 中被使能，一旦发生该事件、NVIC 就会响应 PVD 的中断请求。

PVD 事件的触发/解除机制由硬件自动控制，当 VDD 下降到"PLS 位"所设定的阈值时，PVD 中断被触发，当 VDD 上升到 PVDUV 时，PVD 中断被解除。

这一特性可用于用于执行紧急关闭任务。

图 3-3 PVD 的门限



3.4 低功耗模式

系统或电源复位以后，系统处于运行模式。当系统无需继续运行时，可以利用多种低功耗模式来节省功耗；

例如等待某个外部事件时，用户可以根据最低电源消耗、唤醒时间和可用的唤醒源等条件，选定一个最佳的低功耗模式。

PT32x00x 提供了两种低功耗模式：

- 睡眠状态(Cortex™-M0 内核停止，但包括 Cortex-M0 核心的外设在内的所有外设，如 NVIC、SysTick 等仍在运行，IFMC 无法访问)
- 深度睡眠状态(除 LSI 外的所有时钟均已停止工作，相关的片内电源以及外设被强制关闭)。

此外，在运行模式下，还可以通过以下方式降低功耗：

- 失能未被使用的时钟源
- 失能未被使用但已被使能的外设。
- 将不使用的 GPIO 做内部下拉处理

注意：BG1v5 电源的初始化依赖于 LSI，上电时，LSI 不应立刻被关闭，否则将导致 BG1v5 异常，详见《PT32x00x 数据手册》说明。

表 3-1 低功耗模式资源一览

低功耗状态	强制关闭的时钟	强制关闭的片内电源	强制关闭的外设
睡眠状态	CPU 核心时钟	无	IFMC
深度睡眠状态	HSI 时钟 PLL 时钟 SYS_CLK 挂载的所有外设时钟	BG1v5	PVD IFMC MCO

3.4.1 进入低功耗模式

通过将配置系统控制寄存器（SCR）中的 SLP 位置’1’以选定系统进入低功耗模式时、系统处于睡眠状态还是深度睡眠状态。

系统进入低功耗模式有两种方法：

- 执行 WFI 或 WFE 指令以进入低功耗模式
- 将系统控制寄存器（SCR）中的 SOE 位置’1’，当系统从最低优先级的中断处理程序中退出时，立即进入低功耗模式

注意：进入低功耗模式前，应该将所有未使用到的管脚配置为下拉输入，并清除所有管脚的复用功能可使系统功耗得到最大优化。

3.4.2 睡眠状态

表 3-1 罗列了系统处于睡眠状态时、片内资源的工作状态。

如需优化功耗、在进入睡眠状态前，用户可手动关闭无需在低功耗模式下工作的外设以降低功耗。

3.4.3 深度睡眠状态

表 3-1 罗列了系统处于深度睡眠状态时、片内资源的工作状态。
在深度睡眠状态下，只有 LSI 时钟挂载的外设可以正常运行，内核的寄存器、内存的信息仍保存，程序在唤醒后仍从上一次停止处执行；
除表 3-1 所罗列的在深度睡眠状态下被强制关闭外设外，其他外设均不会被强制关闭，在运行模式下使能的外设、在进入深度睡眠前应该手动关闭以避免额外功耗；
如需进一步的优化功耗，可将 LSI 时钟关闭，但 LSI 时钟挂载外设将无法工作，此时、低功耗模式的唤醒仅支持外部中断唤醒。

注意：GPIO 外部中断唤醒深度睡眠时，仅支持双沿中断、高电平中断和低电平中断。

3.4.4 唤醒方式

当系统处于低功耗模式时，如果是睡眠状态，根据进入低功耗模式的指令，有相应的唤醒方式：

- WFI 指令：任意一个被 NVIC 响应的外设中断都能将系统从睡眠模式唤醒。
- WFE 指令：包括所有中断在内、所有内核支持的事件

特别地、当(系统控制寄存器 SCR)的”SWC”位为’1’时：所有中断或事件均可唤醒。

注意：使用 WFE 唤醒睡眠状态，需要在(中断挂起清除寄存器 ICPR)中清除相应的挂起标志。

如果是深度睡眠状态，根据进入低功耗模式的方法，有相应的唤醒方式：

- WFI 指令：LSI 时钟挂载外设的中断或者外部中断
- WFE 指令：LSI 时钟挂载外设的中断、外部中断、复位事件或调试模式请求事件

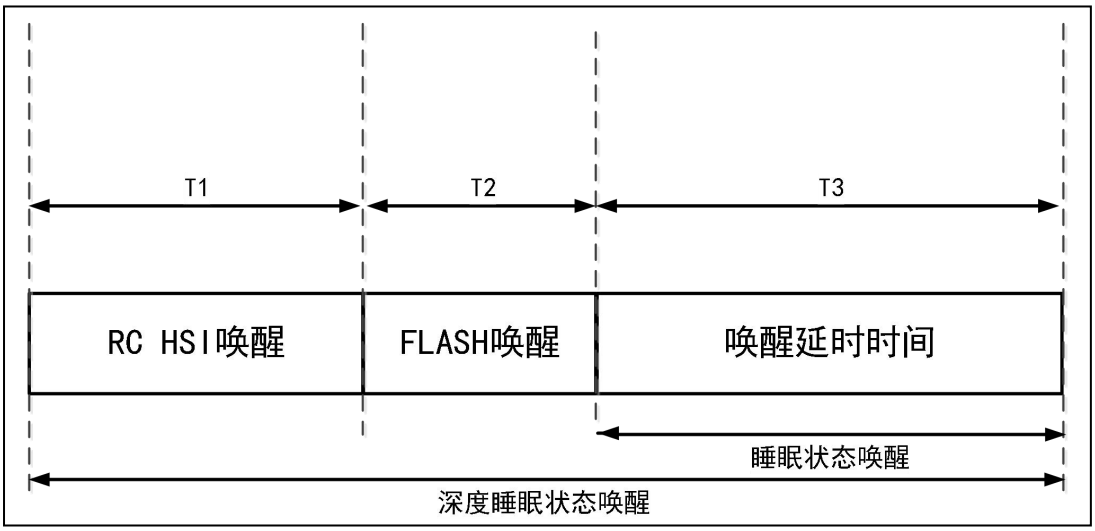
特别地、当(系统控制寄存器 SCR)的”SWC”位为’1’时：LSI 时钟挂载外设的中断、外部中断、复位事件或者调试模式请求事件均可唤醒。

3.4.5 唤醒时间

唤醒时间定义为：从唤醒事件开始至用户程序取读第一条指令。
系统从低功耗模式被唤醒时、时钟被最先唤醒，随后按照顺序唤醒 FLASH，进行唤醒延时，

最后系统重新进入运行模式。

图 3-4 低功耗唤醒过程



芯片在进入低功耗模式之前，可以配置时钟配置寄存器(RCC_CFGR)的”WKSK”位以选择系统时钟在低功耗唤醒后是否自动恢复到上一次的状态。

唤醒时间也是可配置的，通过配置时钟配置寄存器(RCC_CFGR)的”WKDL”位改变唤醒时间，如下表(T_{SYS_CLK} 表示单位系统时钟周期)：

表 3-2 低功耗唤醒过程

WKDL[1:0]	T1	T2	T3
00	16uS(1)	2uS(2)	320 (T_{SYS_CLK})
01			160 (T_{SYS_CLK})
10			132 (T_{SYS_CLK})
11			96 (T_{SYS_CLK})

注：

1. **RC HSI 唤醒时间取 VDD3.3V 时的典型值，详请参考《PT32x00x 数据手册》**
2. **FLASH 唤醒时间取 VDD3.3V 时的典型值，详请参考《PT32x00x 数据手册》**

3.5 寄存器描述

3.5.1 电源电压监测器配置寄存器 PWR_PVDR

（地址： 0x4000_1800）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	PLS[2:0]			PVDE
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

位 0	**PVDE**：电源电压监测器使能**(power voltage detect enable)** 0： 失能 PVD 1： 使能 PVD
位 3：1	**PLS[2:0]**：**PVD** 电平选择**(power voltage detect level select)** 这些位用于选择电源电压检测器的电压阈值 000： 4.0V 001： 1.7V 010： 2.2V 011： 2.75V 100： 3.5V ***注：当 LVD 阈值设为 2.2V 或 1.7V 时，LVD 将只能够复位，而不能够中断。***
位 31：4	保留，必须保持为 0。

3.5.2 系统控制寄存器(SCR)

（地址： 0x E000_ED10）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	SWC	-	SLP	SOE	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	Res	RW	RW	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	保留。必须保持为 0。
位 1	**SOE**：低功耗模式控制 0： 最低优先级的中断处理完毕后，仍处于运行模式 1： 最低优先级的中断处理完毕后，根据 SLP 位，立即进入睡眠或深度睡眠模式
位 2	**SLP**：低功耗模式选择 0： 睡眠状态 1： 深度睡眠状态
位 3	保留。必须保持为 0。
位 4	**SWC**：低功耗唤醒控制 0： 仅使能的中断或者事件可以唤醒低功耗模式 1： 所有中断或者事件都可以唤醒低功耗模式
位 31：0	保留。必须保持为 0。

3.5.3 寄存器列表

地址	寄存器	描述	备注
0x4000_1800	PWR_PVDR	电源电压监测器配置寄存器	PWR_PVDR 说明
0x E000_ED10	SCR	系统控制寄存器	SCR 说明

4 复位和时钟控制(RCC)

4.1 综述

PT32x00x 内部集成了一个多功能的复位和时钟控制器;
在复位功能上、该控制器支持 3 种复位：系统复位、电源复位、功能复位；除高级软件复位外，所有复位均支持复位后取读复位标志。
在时钟功能上、三种不同的时钟源可被用来驱动系统时钟(SYS_CLK)：

- HSI 振荡器时钟
- LSI 振荡器时钟
- PLL 时钟

当不被使用时，任一个时钟源都可被独立地启动或关闭，由此优化系统功耗。

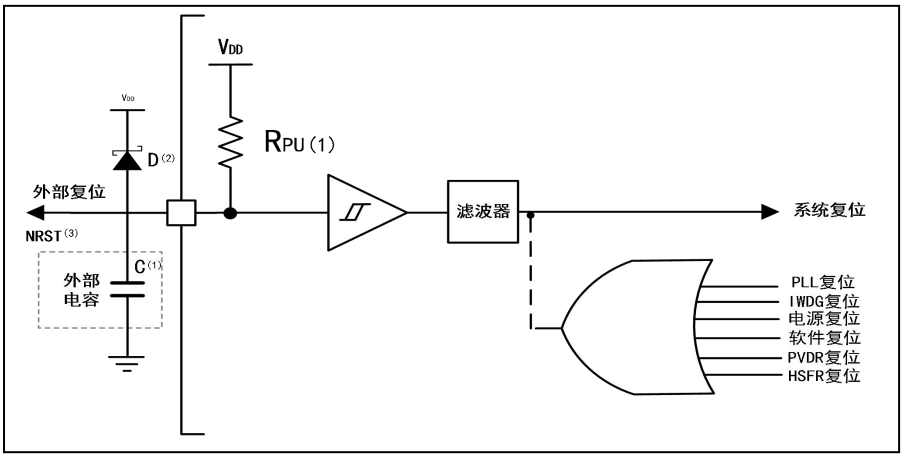
4.2 特性

- 多功能、配置灵活的复位机制
- 集成的高速 24Mhz RC 振荡器 HSI
- 集成的低功耗 32Khz RC 振荡器 LSI
- 集成的 PLL，可将 HSI 或 LSI 时钟 2 倍频作为系统时钟 SYS_CLK
- 鲁棒性优异的时钟安全机制
- 配置灵活的时钟输出

4.3 复位功能描述

PT32x00x 支持三种复位形式，分别为系统复位、上电复位和功能复位。

图 4-1　复位框图



(1):RPU 阻值详见《PT32x00x 数据手册》

注意：

1. 外部电容应该在 1uF~10uF 之间

2. 应当保证存在一个"IF=10mA 时，正向电压 VF 不超过 0.4V"肖特基二极管

3. 即使 NRST 引脚被复用，也应该保证存在外部电容。

4.3.1 系统复位

除了(RCC_RSR 寄存器)中的复位状态外，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)
- 软件复位(SF 复位)
- 独立看门狗计数终止(IWDG 复位)
- PLL 时钟失效复位（PLL 复位)

可通过查看(RCC_RSR 寄存器)中的复位状态以识别复位事件的来源。

4.3.1.1 NRST 引脚复位

当 NRST 引脚被拉低产生外部复位时，它将产生复位脉冲到系统复位。

注意：NRST 引脚支持复用为 IO，但上电时、该引脚仍默认作为 NRST 引脚。

4.3.1.2 软件复位

通过将 Cortex®-M0 的(中断应用和复位控制寄存器 AIRCR)中的 SYSRESETREQ 位置'1'，可实现软件复位。更多信息请参考《Cortex™-M0 技术参考手册》。

4.3.1.3 PLL 复位

将(RCC_RCR 寄存器)的 PREN 位置'1'以使能 PLL 时钟失效复位，当 PLL 时钟失效时，将发生 PLL 时钟失效复位、并将(RCC_RSR 寄存器)的 PLLR 位置'1'。

4.3.2 电源复位

包括(RCC_RSR 寄存器)中的复位标志，电源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，产生一个电源复位：

- 芯片上电/掉电复位（POR)

4.3.2.1 芯片上电/掉电复位

芯片上电/掉电复位产生条件请参考"3.3.2 电源复位条件"一节。
特别地、Flash 配置区中对配置字进行修改的操作需要产生一次 POR 复位或者高级软件复位，更多信息请参考"16.3.5 配置区操作"。

4.3.3 功能复位

功能复位通常具备可编程、多功能的特征；
包括(RCC_RSR 寄存器)中的复位标志，功能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，产生一个功能复位：

- 电源低电压复位（PVDR)
- 高级软件复位（HSFR)

4.3.3.1 电源低电压复位

在电源 PVD 模块使能的前提下、通过将(RCC_RCR 寄存器)中的”PDRE”位置’1’以使能该复位；
使能该复位、在电源电压低于 PVD 阈值时、将发生 PVD 复位、此时、(RCC_RSR 寄存器)中的”PVDR”将被置’1’；
通过为 PVD 模块配置相应的阈值，能够满足多档电源电压下的复位需求，更多信息请参考”3.3.3 可编程的电源电压监测器”节描述。

注意：

1. “PWR_PVDR 寄存器”和“RCC_RCR 寄存器”内的值在复位后即恢复到默认值 0，这意味着 PVD 的复位使能和阈值配置将在复位后失效。

2. 若需要 PVD 限制系统在某一电压范围才能开始工作，则必须在用户程序的最开始进行 PVD 阈值配置和 PVD 复位使能。

4.3.3.2 高级软件复位

在(RCC_RCR 寄存器)中的”HREN”位置’1’的前提下、往(RCC_HSFRR 寄存器)中写入高级软件复位密码’0xAB56’即可发生高级软件复位；
特别地、Flash 配置区中对配置字进行修改的操作需要产生一次 POR 复位或者高级软件复位，更多信息请参考”16.3.5 配置区操作”。

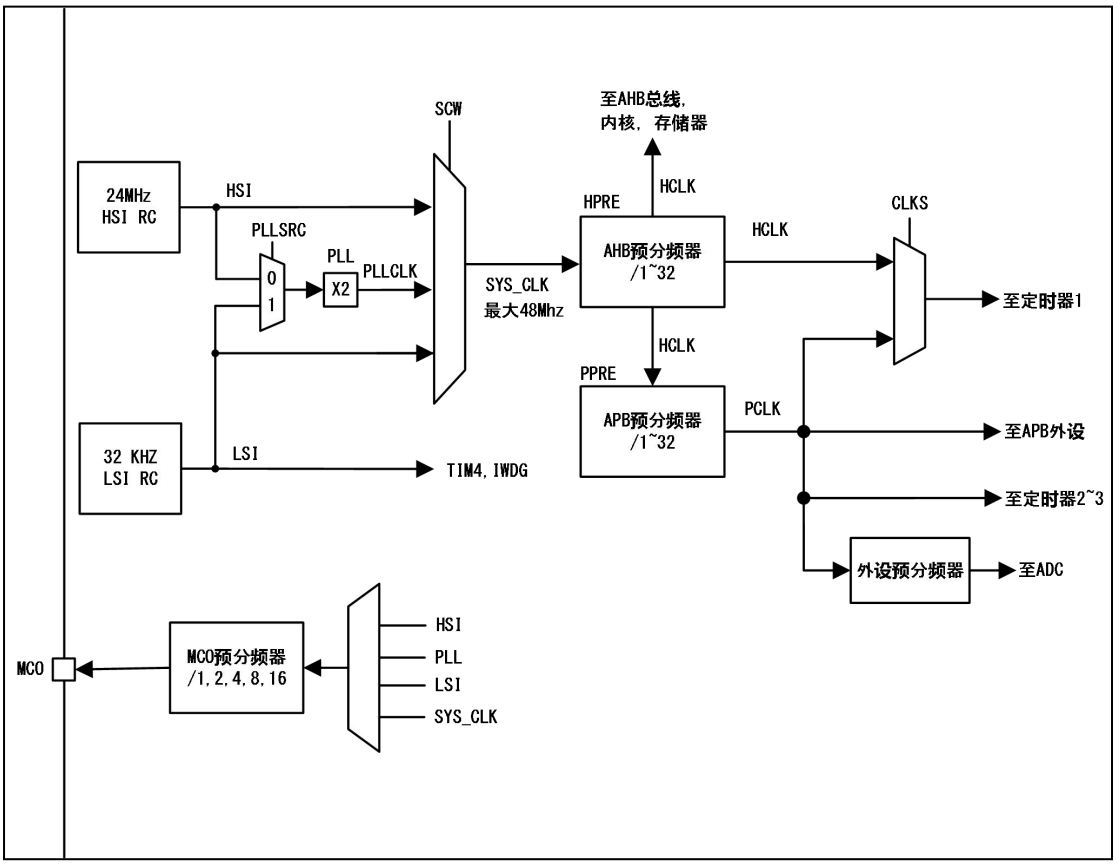
4.4 时钟功能描述

三种不同的时钟源可被用来驱动系统时钟(SYS_CLK)：

- HSI 振荡器时钟
- PLL 时钟
- LSI 振荡器时钟

当不被使用时，任一个时钟源都可被独立地启动或关闭，以优化系统功耗。

图 4-2　时钟树框图



1. **当 HSI 被用于作为 PLL 时钟的输入时，系统时钟 SYS_CLK 即得到最大频率 48Mhz。**
2. **对于时钟源的更多特性，请参考相应产品《PT32x00x 数据手册》中“电气特性”章节。**
3. **用户可通过多个预分频器配置 HCLK、PCLK 的频率。 HCLK 和 PCLK 时钟的最大频率是 48Mhz；**
4. **RCC 通过 HCLK 作为 Cortex M0 系统定时器(SysTick)的外部时钟。**
5. **外设时钟均由对应的外设分频器生成**

4.4.1 HSI 时钟

HSI 时钟能够在不需要任何外部器件的条件下提供系统时钟，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 SYS_CLK 或者作为 PLL 时钟的输入时钟源。

HSI 在满足时钟安全机制的前提下，可以通过时钟高频控制寄存器 RCC_HCR 里的 HSIEN 位来使能或禁用，详见时钟安全机制；

注意：HSI 时钟上电默认启用并作为系统时钟源 SYS_CLK，直到系统运行、SYS_CLK 才按照用户配置进行设置。

4.4.2 LSI 时钟

LSI 时钟信号由内部 32Khz 的 RC 振荡器产生，同样可以直接作为系统时钟或者作为 PLL 时钟的输入时钟源。

LSI 作为一个低功耗的时钟源，它可以在睡眠或深度睡眠模式下保持运行，为独立看门狗和自动唤醒定时器 TIM4 提供时钟。

LSI 时钟上电默认启用、在满足时钟安全机制的前提下，可以通过时钟低频控制寄存器 RCC_LCR 里的 LSIEN 位来使能或禁用，详见时钟安全机制。

注意：上电后，LSI 不应该被立刻关闭，应保证上电后至少 1mS 的延时才能够关闭 LSI。

4.4.3 PLL 时钟

内部 PLL 可以用来倍频 HSI RC 的输出时钟或 LSI RC 的输出时钟。参考图 4-2 和(时钟配置寄存器 RCC_CFGR)。

PLL 的配置(选择 HSI 或 LSI 作为 PLL 的输入时钟源、并选择 PLL 作为系统时钟)必须在其被激活前完成。一旦 PLL 被激活，这些参数就不能被改动。

PLL 时钟的失效将导致(RCC_CFGR 寄存器)内的 PLL 时钟失效标志位 PLLF 被硬件置'1'；此时、如果(RCC_RCR 寄存器)中的"PREN"位被置'1'、将发生系统复位、从(RCC_RSR 寄存器)中的"PLLR"位可获取 PLL 导致的系统复位信息。

4.4.4 系统时钟(SYS_CLK)选择

系统复位后，HSI 被选为系统时钟。

当时钟源被直接或者通过 PLL 间接作为系统时钟时，该时钟源将不能被停止。

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)，系统时钟从一个时钟源切换到另一个时钟源才能够发生；

在被选择时钟源没有就绪时，系统时钟的切换不会发生。直至目标时钟源就绪，才发生切换。

可以通过(RCC_CFGR 寄存器)里的”SCW”位获取或配置当前系统时钟的时钟源信息。

注意：睡眠唤醒后、系统时钟默认为“睡眠前的时钟配置”，可将(RCC_CFGR 寄存器)的”WKSCK”位清’0’以将时钟自动切换到 HSI。

4.4.5 时钟安全机制

默认激活的内部时钟安全机制保证即使极端恶劣工况下，芯片也能正常工作：

- 芯片从低功耗模式回到运行模式后，默认系统时钟均为上一次运行模式下的时钟。
- 当时钟源被直接或者通过 PLL 间接作为系统时钟时，该时钟源将不能被停止。
- PLL 时钟失效将导致系统时钟自动切换到 HSI 时钟，PLL 也将被关闭；
- PLL 时钟失效且使能了 PLL 失效复位、在复位动作之后、系统时钟将自动切换到 HSI 时钟。

注意：当使用 PLL 时钟作为系统主时钟进入休眠时，时钟安全机制仍在睡眠下仍作用，且带来了额外的功耗；为避免额外的睡眠消耗，应该在睡眠前首先将 PLL 时钟切换到 HSI 时钟运行。

4.4.6 看门狗时钟

如果启用 IWDG，LSI 时钟则不可被关闭，关闭 LSI 时钟将导致 IWDG 不可用。

4.4.7 TIM4 时钟

如果启用 TIM4，LSI 时钟则不可被关闭，关闭 LSI 时钟将导致 TIM4 不可用。

4.4.8 时钟输出(MCO)

微控制器允许输出时钟信号到外部的 MCO 引脚上。

使用 MCO 功能，对应的 GPIO 端口必须被正确的复用为 MCO 功能。以下四个时钟信号可被选作 MCO 时钟，下列是 MCO 输出时钟源的选择：

- HSI
- PLL
- LSI
- 当前系统时钟

MCO 输出的时钟选择由(时钟输出配置寄存器 RCC_MCOR)中的”COSRC”位配置，(RCC_MCOR 寄存器)中的”COPRE”位则用于配置 MCO 输出的时钟信号分频数；

注意：MCO 输出信号频率不应大于 10Mhz，否则输出信号将产生非预期的畸变。

4.5 RCC 寄存器描述

4.5.1 时钟高频控制寄存器(RCC_HCR)

（地址： 0x4000_1810）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	HSIEN
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位 0	**HSIEN**：内部高速时钟使能**(HSI enable)** 0： HSI 振荡器禁用 1： HSI 振荡器使能
位 31：1	保留。必须保持为 0。

4.5.2 时钟低频控制寄存器(RCC_LCR)

（地址： 0x4000_1814）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LSIEN
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位 0	**LSIEN**：内部低速时钟使能**(LSI enable)** 0： LSI 振荡器禁用 1： LSI 振荡器使能 **注：上电时，LSI 不应该被立刻关闭，如需优化功率以关闭 LSI，应参考数据手册，保证足够的延时**
位 31：1	保留。必须保持为 0。

4.5.3 时钟 PLL 控制寄存器(RCC_PCR)

（地址： 0x4000_1818）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PLL EN
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**PLLEN：PLL 时钟使能(PLL enable)** 0： PLL 时钟禁用 1： PLL 时钟使能
位 31：1	保留。必须保持为 0。

4.5.4 时钟输出配置寄存器(RCC_MCOR)

（地址： 0x4000_1824）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	COSRC[1:0]		-	COPRE[2:0]		
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 2：0	**COPRE[2:0]**：时钟输出预分频器**(clock output prescaler)** 000： 时钟输出不分频 001： 时钟输出 2 分频 010： 时钟输出 4 分频 011： 时钟输出 8 分频 100： 时钟输出 16 分频 其他： 无效
位 3	保留。必须保持为 0。
位 5：4	**COSRC[1:0]**：时钟输出源配置**(clock output source)** 00： 时钟输出源为 HSI 时钟 01： 时钟输出源为 PLL 时钟 10： 时钟输出源为 LSI 时钟 11： 时钟输出源为当前系统时钟
位 31：6	保留。必须保持为 0。

4.5.5 时钟配置寄存器(RCC_CFGR)

（地址： 0x4001_F00C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	PPRE[4:0]					-	-	-	WKCK	-	-	WKDL[1:0]	
R/W	Res	Res	Res	RW	RW	RW	RW	RW	Res	Res	Res	RW	Res	Res	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	PLLF	-	-	-	-	-	-	HPRE[4:0]					-	PLL SRC	SCW[1:0]	
R/W	RW	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 1：0	**SCW：系统时钟切换(system clock switch)** 0x： HSI 振荡器作为系统时钟 10： PLL 时钟作为系统时钟 11： LSI 振荡器作为系统时钟
位 2	**PLLSRC：PLL 输入时钟源 (PLL entry clock source)** 0： HSI 振荡器作为 PLL 输入时钟源 1： LSI 时钟作为 PLL 输入时钟源
位 3	保留。必须保持为 0。
位 8：4	**HPRE[4:0]：HCLK 预分频(HCLK prescaler)** 00000： HCLK 不分频 00001： HCLK 2 分频 00010： HCLK 3 分频 …… 11110： HCLK 31 分频 11111： HCLK 32 分频
位 14：9	保留。必须保持为 0。
位 15	**PLLF：PLL 时钟失效标志位(PLL fail flag)** 0： 无 PLL 时钟失效发生 1： 发生 PLL 时钟失效
位 17：16	**WKDL[1:0]：唤醒延时时间(wake up delay times)** 00： 最大延时，320 个 SYS_CLK 时钟周期 01： 次大延时，160 个 SYS_CLK 时钟周期 10： 次小延时，132 个 SYS_CLK 时钟周期 11： 最小延时， 96 个 SYS_CLK 时钟周期 ***注：详见电源控制一节 3.4.5 唤醒时间描述***
位 19：18	保留。必须保持为 0。
位 20	**WKSK：唤醒后默认系统时钟配置(system clock after wakeup)**

	0： 系统时钟默认为 HSI 时钟 1： 系统时钟默认为睡眠之前的系统时钟
位 23：21	保留。必须保持为 0。
位 28：24	**PPRE[4:0]：PCLK 预分频(PCLK prescaler)** 00000： PCLK 不分频 00001： PCLK 2 分频 00010： PCLK 3 分频 …… 11110： PCLK 31 分频 11111： PCLK 32 分频
位 31：29	保留。必须保持为 0。

4.5.6 复位状态寄存器 RCC_RSR

（地址： 0x4001_F010）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	PVDR	PINR	POR	PLLR	-	-	-	IWDGR	SFR
R/W	Res	Res	Res	Res	Res	Res	Res	Rw1	Rw1	Rw1	Rw1	Res	Res	Res	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**SFR**：软件复位标志**(software reset)** 0： 无软件复位发生 1： 发生软件复位，该位写 1 清除
位 1	**IWDGR**：独立看门狗复位标志**(iwdg reset)** 0： 无独立看门狗复位发生 1： 发生独立看门狗复位，该位写 1 清除
位 4：2	保留。必须保持为 0。
位 5	**PLLR**：**PLL** 时钟失效复位标志**(PLL fail reset)** 0： 无 PLL 时钟失效复位发生 1： 发生 PLL 时钟失效复位，该位写 1 清除
位 6	**POR**：上电/掉电复位标志**(power reset)** 0： 无上电/掉电复位发生 1： 发生上电/掉电复位，该位写 1 清除
位 7	**PINR**：**NRST** 引脚复位标志**(pin reset)** 0： 无 NRST 引脚复位发生 1： 发生 NRST 引脚复位，该位写 1 清除
位 8	**PVDR**：电源低电压复位标志**(PVD low voltage reset)** 0： 无电源低电压复位发生 1： 发生电源低电压复位，该位写 1 清除
位 31：9	保留。必须保持为 0。

4.5.7 高级软件复位控制寄存器(RCC_HSFRR)

（地址： 0x4001_F014）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	Key[15:0]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**Key[15:0]**：高级软件复位密码**(key)** 0xAB56：写入密码、发生高级软件复位 其他值： 无效 ***注：只有当(RCC_RCR 寄存器)的"HREN"位被使能时，写入密码才能发生高级软件复位***
位 31：0	保留。必须保持为 0。

4.5.8 复位控制寄存器 RCC_RCR

（地址： 0x4001_F018）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	HREN	-	PDRE	-	-	PREN	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	RW	Res	RW	Res	Res	RW	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0

位 4：0	保留。必须保持为 **0**。
位 5	**PREN**：**PLL** 时钟失效复位使能**(PLL fail reset enable)** 0： 复位禁止 1： 复位使能
位 7：6	保留。必须保持为 0。
位 8	**PDRE**：电源低电压检测触发复位使能**(PVD low voltage reset enable)** 0： 复位禁止 1： 复位使能
位 9	保留。必须保持为 0。
位 10	**HREN**：高级软件复位使能**(Advanced software reset enable)** 0： 复位禁止 1： 复位使能
位 31：11	保留。必须保持为 **0**。

4.5.9 寄存器列表

地址	寄存器	描述	备注
0x4000_1810	RCC_HCR	时钟高频控制寄存器	RCC_HCR 说明
0x4000_1814	RCC_LCR	时钟低频控制寄存器	RCC_LCR 说明
0x4000_1818	RCC_PCR	时钟 PLL 控制寄存器	RCC_PCR 说明
0x4000_1824	RCC_MCOR	时钟输出配置寄存器	RCC_MCOR 说明
0x4001_F00C	RCC_CFGR	时钟配置寄存器	RCC_CFGR 说明
0x4001_F010	RCC_RSR	复位状态寄存器	RCC_RSR 说明
0x4001_F014	RCC_HSFRR	高级软件复位控制寄存器	RCC_HSFRR 说明
0x4001_F018	RCC_RCR	复位控制寄存器	RCC_RCR 说明

5 通用和复用功能 I/O(GPIO 和 AFIO)

5.1 综述

根据《PT32x00x 数据手册》中列出的每个 I/O 引脚和相关的硬件特征，每个 I/O 都可以由软件分别配置、按照输入输出模式可以有如下配置：

- 输入:
  - 输入浮空
  - 输入上拉
  - 输入下拉
  - 数字复用输入
  - 模拟复用输入
- 输出
  - 开漏输出
  - 开漏弱上拉/下拉输出
  - 推挽输出
  - 数字复用输出
  - 模拟复用输出

每个 I/O 可以被自由编程，然而 I/O 相关寄存器必须按 32 位字访问(不可以半字或字节访问)。

5.2 特征

- 所有的 I/O 都可配置为输入或输出
- 所有的 I/O 都可作为外部中断
- 可选的、多功能的数字功能复用
- 可选的、多功能的模拟功能复用
- 通过分开的使能和失能操作控制寄存器，提供中断安全的 GPIO 操作
- 提供内部弱上/下拉电阻，上/下拉电阻阻值详见《PT32x00x 数据手册》
- 端口映射操作，提供单指令的 I/O 操作，以保障中断安全

5.3 GPIO/AFIO 功能描述

复位期间和刚复位后，除 SWD 接口和 NRST 引脚外、所有的 I/O 默认为输入浮空(GPIOx_OES 寄存器的 ”IOy”=0，GPIOx_PUS 寄存器的 ”IOy”=0，GPIOx_PDS 寄存器的 ”IOy”=0)，直到进入运行模式，用户对 I/O 的配置才能够生效；

5.3.1 GPIO 输入配置

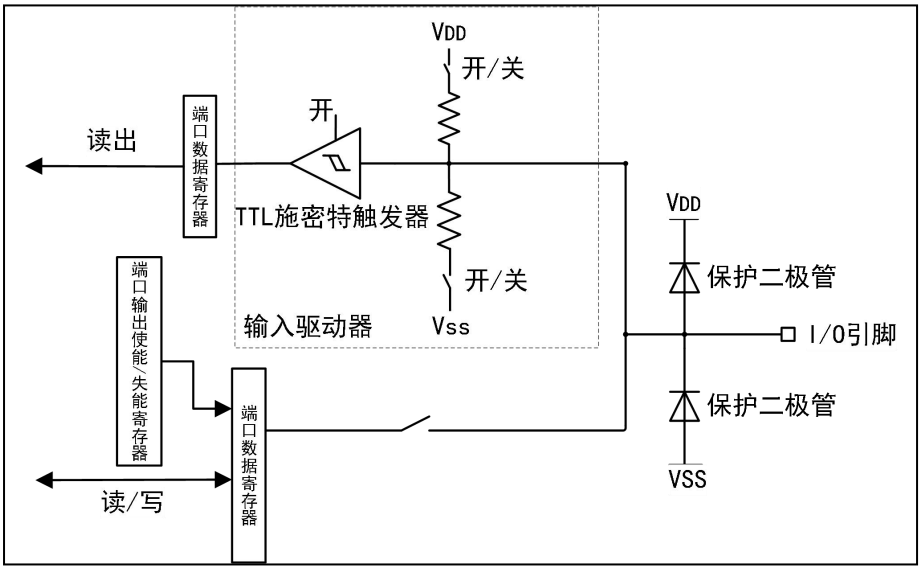
GPIO 上电默认输入；

配置 GPIO 为输出后，通过将(端口输出失能寄存器 GPIOx_OEC)的相关位置’1’以清除 GPIO 输出配置，并将 GPIO 恢复到输入模式；

当 GPIO 被配置为输入时：

- 输出驱动器被禁止
- 根据输入配置(上拉、下拉、施密特触发)的不同，弱上拉/下拉电阻和施密特触发器被连接
- 出现在 I/O 脚上的数据在每个 HCLK 时钟被采样到(端口数据寄存器 GPIOx_DR)
- 对(GPIOx_DR 寄存器)的读访问可得到当前 I/O 状态

图 5-1 输入配置



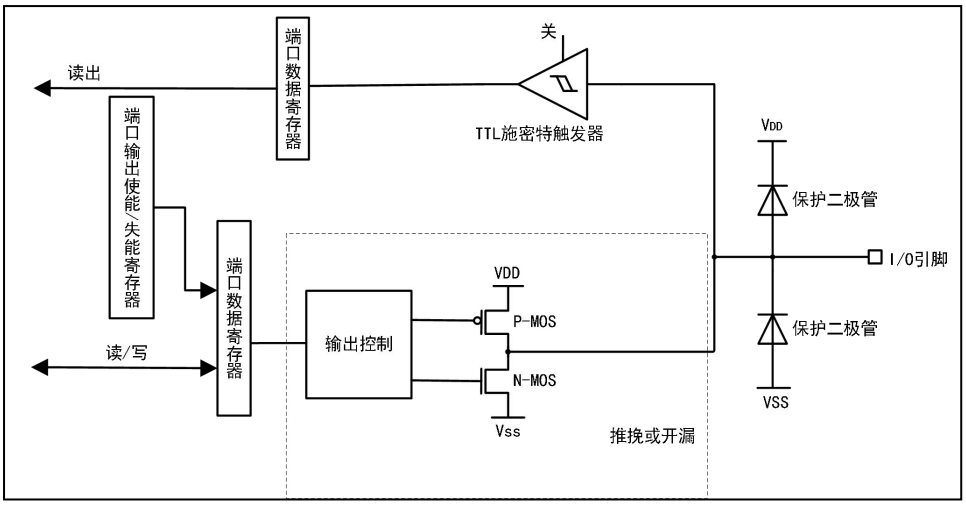
5.3.2 GPIO 输出配置

通过将(端口输出使能寄存器 GPIOx_OES)的相关位置'1'以将 GPIO 配置为输出模式；当 GPIO 被配置为输出时：

- 输出驱动器被开启
- 输出配置
  1. 开漏模式：(GPIOx_DR 寄存器)上的'0'开启 N-MOS，而(GPIOx_DR 寄存器)上的'1'将端口置于高阻状态(PMOS 从不被开启)。
  2. 推挽模式：(GPIOx_DR 寄存器)上的'0'开启 N-MOS，而(GPIOx_DR 寄存器)上的'1'将开启 P-MOS。
- 在推挽模式下、弱上拉/下拉电阻被禁止
- 出现在 I/O 脚上的数据在每个 HCLK 时钟被采样到(GPIOx_DR 寄存器)
- 对(GPIOx_DR 寄存器)的读访问可得到当前 I/O 状态

下图给出了 I/O 端口位的输出配置。

图 5-2 输出配置



5.3.3 外部中断

所有 I/O 都有外部中断能力。为了使用外部中断，端口必须配置成输入模式。

更多的关于外部中断的信息，请参考："外部中断控制器(EXTI)"一章

5.3.4 映射操作

通过(GPIO_DR 寄存器)对 I/O 操作时，由于寄存器的操作是按'字'进行的，为了避免对其他 I/O 的误操作，一般流程如下：

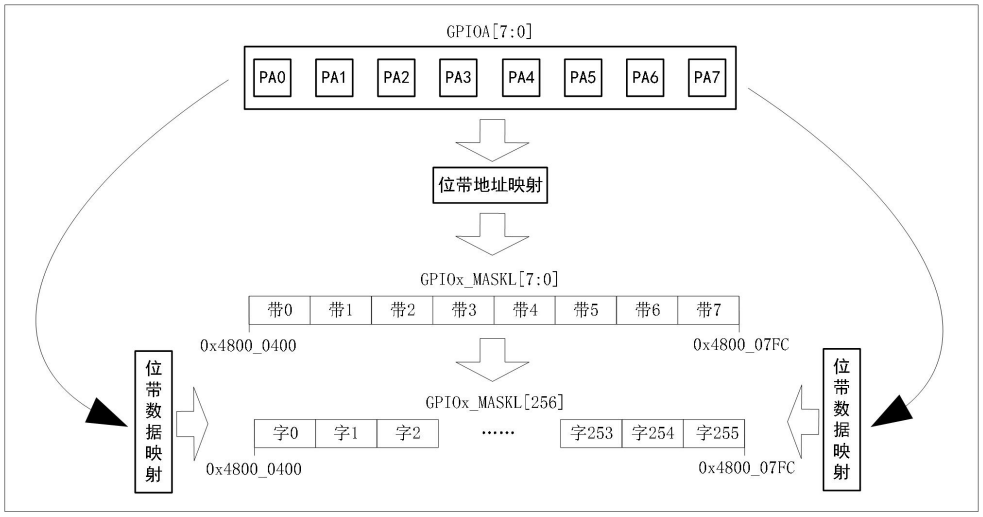
- 读 DR 寄存器
- DR 寄存器值与目标 I/O 的值进行'逻辑与'/'逻辑或'操作
- 操作结果赋值给 DR 寄存器

在一些 I/O 模拟并行总线或需要快速切换 I/O 的应用场合，上述流程的执行可能导致无法满足速率要求；

PT 提供了一种映射操作，这种操作有：

- 对 x 端口内的单个或多个 I/O 的操作是绝对的(这意味着 x 端口内的其他 I/O 不会被影响)
- 鉴于其操作的"绝对"特性，可以直接赋值，保障了中断安全。

图 5-3 映射操作，以 GPIOA 低 8 位 I/O 为例



对于 16 位宽的一个端口 GPIOx[15:0]，映射操作分低 8 位和高 8 位两组进行，分别映射至 GPIOx_MASKL[7:0]和 GPIOx_MASKH[7:0]。

示例：需要 PA0 和 PA1 输出高，PA6 和 PA7 输出低，不改变端口内的其他 I/O。

- 参考图 5-3 的位带"地址"映射，确定所需操作的 I/O 位于"带 0、1、6、7"，用二进制表示为"11000011"，故可以确定目标地址为 GPIOx_MASKL[0xC3]，该地址是绝对的，只对应 PA0、PA1、PA6 和 PA7，对 GPIOA 端口内的其他 I/O 是无意义的。
- 参考图 5-3 的位带"数据"映射，我们需要 PA0 和 PA1 输出高，则 GPIOA[1:0]=11,GPIOA[7:6]=00，用二进制表示为"00xxxx11(x 代表 0 或 1 任意值)"，最终，地址 GPIOx_MASKL[0xC3]=0x03，即一次实现了四个数据位同步置 1 和清零的操作。

注意：

1. *使用映射操作前，相关 I/O 应当被配置为输出模式*
2. *GPIOx_MASKL[7:0]和 GPIOx_MASKH[7:0]地址，仅低 8 位有意义。*

5.3.5 AFIO 复用 IO 功能功能描述

引脚是有限的，但功能却可以是多样的；

PT 为 GPIO 提供了多种功能，这些功能可以分为数字功能和模拟功能，通过对引脚功能的灵活配置，可以满足更多样化的需求。

复位期间和刚复位后，除必须的功能(SWD、可能的 NRST)外，所有的 GPIO 默认为主功能，直到进入运行模式，用户对引脚的复用才能够生效；

注意：关于 IO 引脚支持的复用功能、请参考《PT32x00x 数据手册》中的引脚定义部分。

5.3.5.1 数字功能复用配置

数字功能输入输出均为数字信号；

端口数字功能配置寄存器(AFIOx_AFS0)和(AFIOx_AFS1)用于配置 GPIO 的数字复用功能，每个 GPIO 引脚都有独立的 3bit 位宽的配置位，它最多支持 GPIO 引脚复用 7 个数字功能，详见(AFIOx_AFS0)或(AFIOx_AFS1)寄存器的说明。

端口复用功能清除寄存器(AFIOx_AFC)用于清除相应的 GPIO 引脚复用功能、以将相应的 GPIO 引脚功能恢复到主功能；

当 I/O 端口被复用为数字功能时：

- I/O 的输入输出模式内部自动配置
- 手动启用的施密特触发输入
- 手动启用的弱上拉和弱下拉电阻
- 在每个 HCLK 时钟周期，出现在 I/O 脚上的数据被采样到(GPIOx_DR 寄存器)
- 开漏模式下，读(GPIOx_DR 寄存器)可以得到 I/O 口状态
- 在推挽模式时，读(GPIOx_DR 寄存器)可以得到当前 I/O 状态

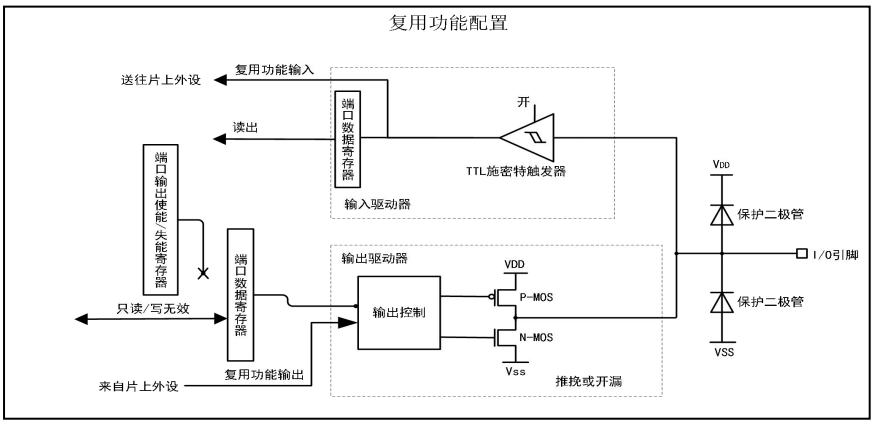
当资源允许，用户可以把一些复用功能重新映射到不同的引脚(参考《PT32x00x 数据手册》中的引脚定义部分)，下图展示了 I/O 端口位的复用功能配置。

注意：

1. 一个 GPIO 引脚可能支持多种数字功能和模拟功能，但同一时间内、GPIO 引脚只能使用其中一种，禁止将模拟功能和数字功能混用。

2. 切换不同的数字复用功能前，必须在(GPIOx_AFC)中先将之前的复用清除。

图 5-3 复用功能配置



5.3.5.2 模拟功能复用配置

模拟功能的输入输出均为模拟信号；

(端口模拟功能使能寄存器 ANAS)和(端口模拟功能失能寄存器 ANAC)用于管理 GPIO 的模拟复用功能；

当 GPIO 的模拟复用功能被使能后，只有相关的模块(如 ADC)被正确配置，GPIO 的模拟功能才能正常使用；

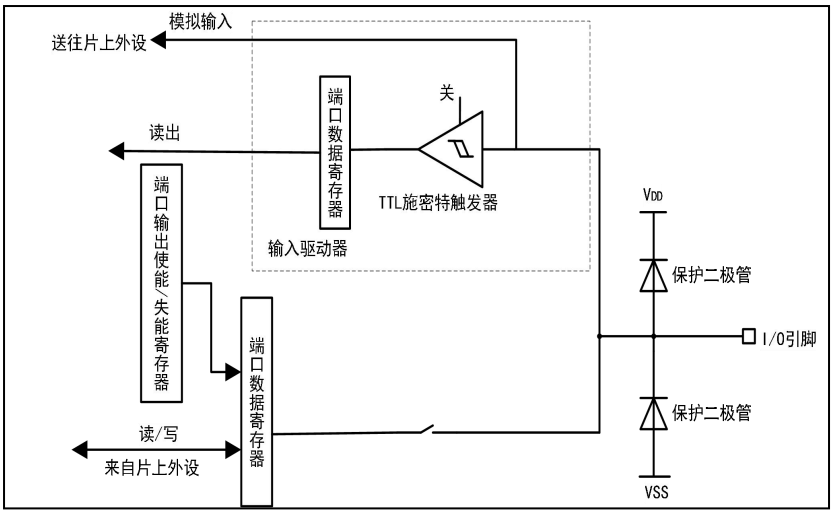
当 I/O 端口被配置为模拟输入配置时：

- 输出缓冲器被禁止；
- 弱上拉和下拉电阻被禁止；
- 读取(GPIOx_DR 寄存器)时数值为'0'。

下图示出了 I/O 端口的模拟输入配置：

注意：一个 GPIO 引脚可能支持多种模拟功能，但同一时间内、只能开启一个模拟功能，否则将导致功能异常。

图 5-4 模拟输入配置



(1)模拟复用输入模式仅在集成了 ADC 外设、OPA 外设、CMP 外设的产品上有效

(2)模拟复用输出模式仅在集成了 DAC 外设、LCD 外设、OPA 外设、CMP 外设的产品上有效

5.3.6 外设复用下的 IO 状态

下列表格列出了各个外设的引脚配置。

表 5-1 高级定时器 TIM1

TIM1 引脚	配置	IO 状态
TIM1_CHx	输入捕获通道 x	浮空输入
	输出比较通道 x	推挽输出
TIM1_CHxN	互补输出通道 x	推挽输出
TIM1_BKIN	刹车输入	浮空输入

表 5-2 UART

UART 引脚	配置	IO 状态
UARTx_TX	全双工模式	推挽输出
	单线半双工模式	推挽输出/浮空输入
	单线单工模式	推挽输出
UARTx_RX	全双工模式	浮空输入
	单线半双工模式	推挽输出
	单线单工模式	未使用，可作为通用 I/O

表 5-3 SPI

SPI 引脚	配置	IO 状态
SPI_SCK	主模式	推挽输出
	从模式	浮空输入
SPI_MOSI	主模式	推挽输出
	从模式	浮空输入
SPI_MISO	主模式	浮空输入
	从模式	推挽输出
SPI_CS	硬件主模式	推挽输出
	硬件从模式	浮空输入
	软件模式	未使用，可作为通用 I/O

表 5-4 I2C

I2C 引脚	配置	IO 状态
I2C_SCL	I2C 时钟	开漏输出
I2C_SDA	I2C 数据	开漏输出

表 5-5 ADC

I2C 引脚	配置	IO 状态
ADC_INx	ADC 输入通道	模拟输入

表 5-6 其他 I/O 功能

引脚	描述	IO 状态
MCO	系统时钟输出	推挽输出

5.4 GPIO 寄存器描述

请参考第 1 章中用到的缩写。
必须以字(32 位)的方式操作这些外设寄存器。

5.4.1 端口数据寄存器(GPIOx_DR) (x=A..D)

（地址： GPIOA：0x4800_0000；GPIOB：0x4800_1000；
GPIOC：0x4800_2000；GPIOD：0x4800_3000）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	DR15	DR14	DR13	DR12	DR11	DR10	DR9	DR8	DR7	DR6	DR5	DR4	DR3	DR2	DR1	DR0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**DRy**：端口 x 引脚 y 的输入/输出数据**(port x pin y data)** 0： 端口 x 引脚 y 输入/输出为低电平 1： 端口 x 引脚 y 输入/输出为高电平
位 31：16	保留，必须保持为 **0**。

5.4.2 端口输出使能寄存器(GPIOx_OES) (x=A..D)

（地址： GPIOA：0x4800_0008；GPIOB：0x4800_1008；
GPIOC：0x4800_2008；GPIOD：0x4800_3008）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IOy：端口 x 引脚 y 输出使能(port x pin y output enable) 0： 端口 x 引脚 y 为输入状态，该位写 0 无效 1： 端口 x 引脚 y 输出使能
位 31：16	保留，必须保持为 0。

5.4.3 端口输出失能寄存器(GPIOx_OEC) (x=A..D)

（地址： GPIOA：0x4800_000C；GPIOB：0x4800_100C；
GPIOC：0x4800_200C；GPIOD：0x4800_300C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IOy：端口 x 引脚 y 输出失能(port x pin y output disable) 0： 无效操作 1： 清除端口 x 引脚 y 的输出模式配置，端口自动切换到输入模式
位 31：16	保留，必须保持为 0。

5.4.4 输入上拉使能寄存器(GPIOx_PUS) (x=A..D)

（地址： GPIOA：0x4800_0040；GPIOB：0x4800_1040；
GPIOC：0x4800_2040；GPIOD：0x4800_3040）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IOy：端口 x 引脚 y 输入上拉使能(Port x pin y input pull-up enable) 0： 端口 x 引脚 y 输入上拉功能失能，该位写 0 无效 1： 端口 x 引脚 y 输入上拉使能
位 31：16	保留，必须保持为 0。

5.4.5 输入上拉失能寄存器(GPIOx_PUC) (x=A..D)

（地址： GPIOA：0x4800_0044；GPIOB：0x4800_1044；
GPIOC：0x4800_2044；GPIOD：0x4800_3044）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IOy：端口 x 引脚 y 输入上拉失能(Port x pin y input pull-up disable) 0： 无效操作 1： 端口 x 引脚 y 输入上拉失能
位 31：16	保留，必须保持为 0。

5.4.6 输入下拉使能寄存器(GPIOx_PDS) (x=A..D)

（地址： GPIOA：0x4800_0048；GPIOB：0x4800_1048；
GPIOC：0x4800_2048；GPIOD：0x4800_3048）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**IOy**：端口 x 引脚 y 输入下拉使能**(Port x pin y input pull-down enable)** 0： 端口 x 引脚 y 输入下拉功能失能，该位写 0 无效 1： 端口 x 引脚 y 输入下拉使能
位 31：16	保留，必须保持为 **0**。

5.4.7 输入下拉失能寄存器(GPIOx_PDC) (x=A..D)

（地址： GPIOA：0x4800_004C；GPIOB：0x4800_104C；
GPIOC：0x4800_204C；GPIOD：0x4800_304C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**IOy**： 端口 x 引脚 y 输入下拉失能**(Port x pin y input pull-down disable)** 0： 无效操作 1： 端口 x 引脚 y 输入下拉失能
位 31：16	保留，必须保持为 **0**。

5.4.8 输出开漏使能寄存器(GPIOx_ODS) (x=A..D)

（地址： GPIOA：0x4800_0050；GPIOB：0x4800_1050；
GPIOC：0x4800_2050；GPIOD：0x4800_3050）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IOy： 端口 x 引脚 y 输出开漏使能(Port x pin y output open-drain enable) 0： 端口 x 引脚 y 输出开漏功能失能，该位写 0 无效 1： 端口 x 引脚 y 输出开漏使能
位 31：16	保留，必须保持为 0。

5.4.9 输出开漏失能寄存器(GPIOx_ODC) (x=A..D)

（地址： GPIOA：0x4800_0054；GPIOB：0x4800_1054；
GPIOC：0x4800_2054；GPIOD：0x4800_3054）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IOy： 端口 x 引脚 y 输出开漏失能(Port x pin y output open-drain disable) 0： 无效操作 1： 端口 x 引脚 y 输出开漏失能
位 31：16	保留，必须保持为 0。

5.4.10 输入施密特触发使能寄存器(GPIOx_CSS) (x=A..D)

（地址： GPIOA：0x4800_0070；GPIOB：0x4800_1070；
GPIOC：0x4800_2070；GPIOD：0x4800_3070）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IOy： 端口 x 引脚 y 输入施密特触发使能(Port x pin y input Schmitt trigger enable) 0： 端口 x 引脚 y 输入施密特触发功能失能，该位写 0 无效 1： 端口 x 引脚 y 输入施密特触发使能
位 31：16	保留，必须保持为 0。

5.4.11 输入施密特触发失能寄存器(GPIOx_CSC) (x=A..D)

（地址： GPIOA：0x4800_0074；GPIOB：0x4800_1074；
GPIOC：0x4800_2074；GPIOD：0x4800_3074）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IOy： 端口 x 引脚 y 输入施密特触发失能(Port x pin y input Schmitt trigger disable) 0： 无效操作 1： 端口 x 引脚 y 输入施密特触发失能
位 31：16	保留，必须保持为 0。

5.4.12 端口低 8 位映射操作区域(GPIOx_MASKL) (x=A..D)

（地址： GPIOA：0x4800_0400；GPIOB：0x4800_1400；
GPIOC：0x4800_2400；GPIOD：0x4800_3400）

GPIOx_MASKL 为一块长度为 256 字的虚拟内存区，用于映射针对端口 GPIOx 低 8 位 I/O 中任意单个或多个 I/O 的一次性置 1 和清 0 组合操作(中断安全操作):

- x=A，针对 GPIOA 端口模块；
- x=B，针对 GPIOB 端口模块；
- x=C，针对 GPIOC 端口模块；
- x=D，针对 GPIOD 端口模块；

该区域只写，读回数据为无效．请参考“5.3.4 端口映射操作”章节。

5.4.13 端口高 8 位映射操作区域(GPIOx_MASKH) (x=A..D)

（地址： GPIOA：0x4800_0800；GPIOB：0x4800_1800；
GPIOC：0x4800_2800；GPIOD：0x4800_3800）

GPIOx_MASKH 为一块长度为 256 字的虚拟内存区，用于映射针对端口 GPIOx 低 8 位 I/O 中任意单个或多个 I/O 的一次性置 1 和清 0 组合操作(中断安全操作):

- x=A，针对 GPIOA 端口模块；
- x=B，针对 GPIOB 端口模块；
- x=C，针对 GPIOC 端口模块；
- x=D，针对 GPIOD 端口模块；

该区域只写，读回数据为无效．请参考“5.3.4 端口映射操作”章节。

5.4.14 寄存器列表

地址	寄存器	描述	备注
0x4800_0000	GPIOA_DR	GPIOA 端口数据寄存器	GPIOx_DR 说明
0x4800_0008	GPIOA_OES	GPIOA 端口输出使能置位寄存器	GPIOx_OES 说明
0x4800_000C	GPIOA_OEC	GPIOA 端口输出使能清除寄存器	GPIOx_OEC 说明
0x4800_0040	GPIOA_PUS	GPIOA 内部上拉使能置位寄存器	GPIOx_PUS 说明
0x4800_0044	GPIOA_PUC	GPIOA 内部上拉使能清除寄存器	GPIOx_PUC 说明
0x4800_0048	GPIOA_PDS	GPIOA 内部下拉使能置位寄存器	GPIOx_PDS 说明
0x4800_004C	GPIOA_PDC	GPIOA 内部下拉使能清除寄存器	GPIOx_PDC 说明
0x4800_0050	GPIOA_ODS	GPIOA 输出开漏使能置位寄存器	GPIOx_ODS 说明
0x4800_0054	GPIOA_ODC	GPIOA 输出开漏使能清除寄存器	GPIOx_ODC 说明
0x4800_0070	GPIOA_CSS	GPIOA 施密特功能置位寄存器	GPIOx_CSS 说明
0x4800_0074	GPIOA_CSC	GPIOA 施密特功能清除寄存器	GPIOx_CSC 说明
0x4800_0400 ~ 0x4800_07FC	GPIOA_MASKL	GPIOA 端口低 8 位映射操作区	GPIOx_MASKL 说明
0x4800_0800 ~ 0x4800_BFC	GPIOA_MASKH	GPIOA 端口高 8 位映射操作区	GPIOx_MASKH 说明
地址	寄存器	描述	备注
0x4800_1000	GPIOB_DR	GPIOB 端口数据寄存器	GPIOx_DR 说明
0x4800_1008	GPIOB_OES	GPIOB 端口输出使能置位寄存器	GPIOx_OES 说明
0x4800_100C	GPIOB_OEC	GPIOB 端口输出使能清除寄存器	GPIOx_OEC 说明
0x4800_1040	GPIOB_PUS	GPIOB 内部上拉使能置位寄存器	GPIOx_PUS 说明
0x4800_1044	GPIOB_PUC	GPIOB 内部上拉使能清除寄存器	GPIOx_PUC 说明
0x4800_1048	GPIOB_PDS	GPIOB 内部下拉使能置位寄存器	GPIOx_PDS 说明
0x4800_104C	GPIOB_PDC	GPIOB 内部下拉使能清除寄存器	GPIOx_PDC 说明
0x4800_1050	GPIOB_ODS	GPIOB 输出开漏使能置位寄存器	GPIOx_ODS 说明
0x4800_1054	GPIOB_ODC	GPIOB 输出开漏使能清除寄存器	GPIOx_ODC 说明
0x4800_1070	GPIOB_CSS	GPIOB 施密特功能置位寄存器	GPIOx_CSS 说明
0x4800_1074	GPIOB_CSC	GPIOB 施密特功能清除寄存器	GPIOx_CSC 说明
0x4800_1400 ~ 0x4800_17FC	GPIOB_MASKL	GPIOB 端口低 8 位映射操作区	GPIOx_MASKL 说明
0x4800_1800 ~ 0x4800_BFC	GPIOB_MASKH	GPIOB 端口高 8 位映射操作区	GPIOx_MASKH 说明

地址	寄存器	描述	备注
0x4800_2000	GPIOC_DR	GPIOC 端口数据寄存器	GPIOx_DR 说明
0x4800_2008	GPIOC_OES	GPIOC 端口输出使能置位寄存器	GPIOx_OES 说明
0x4800_200C	GPIOC_OEC	GPIOC 端口输出使能清除寄存器	GPIOx_OEC 说明
0x4800_2040	GPIOC_PUS	GPIOC 内部上拉使能置位寄存器	GPIOx_PUS 说明
0x4800_2044	GPIOC_PUC	GPIOC 内部上拉使能清除寄存器	GPIOx_PUC 说明
0x4800_2048	GPIOC_PDS	GPIOC 内部下拉使能置位寄存器	GPIOx_PDS 说明
0x4800_204C	GPIOC_PDC	GPIOC 内部下拉使能清除寄存器	GPIOx_PDC 说明
0x4800_2050	GPIOC_ODS	GPIOC 输出开漏使能置位寄存器	GPIOx_ODS 说明
0x4800_2054	GPIOC_ODC	GPIOC 输出开漏使能清除寄存器	GPIOx_ODC 说明
0x4800_2070	GPIOC_CSS	GPIOC 施密特功能置位寄存器	GPIOx_CSS 说明
0x4800_2074	GPIOC_CSC	GPIOC 施密特功能清除寄存器	GPIOx_CSC 说明
0x4800_2400 ~ 0x4800_27FC	GPIOC_MASKL	GPIOC 端口低 8 位映射操作区	GPIOx_MASKL 说明
0x4800_2800 ~ 0x4800_2BFC	GPIOC_MASKH	GPIOC 端口高 8 位映射操作区	GPIOx_MASKH 说明
地址	寄存器	描述	备注
0x4800_3000	GPIOD_DR	GPIOD 端口数据寄存器	GPIOx_DR 说明
0x4800_3008	GPIOD_OES	GPIOD 端口输出使能置位寄存器	GPIOx_OES 说明
0x4800_300C	GPIOD_OEC	GPIOD 端口输出使能清除寄存器	GPIOx_OEC 说明
0x4800_3040	GPIOD_PUS	GPIOD 内部上拉使能置位寄存器	GPIOx_PUS 说明
0x4800_3044	GPIOD_PUC	GPIOD 内部上拉使能清除寄存器	GPIOx_PUC 说明
0x4800_3048	GPIOD_PDS	GPIOD 内部下拉使能置位寄存器	GPIOx_PDS 说明
0x4800_304C	GPIOD_PDC	GPIOD 内部下拉使能清除寄存器	GPIOx_PDC 说明
0x4800_3050	GPIOD_ODS	GPIOD 输出开漏使能置位寄存器	GPIOx_ODS 说明
0x4800_3054	GPIOD_ODC	GPIOD 输出开漏使能清除寄存器	GPIOx_ODC 说明
0x4800_3070	GPIOD_CSS	GPIOD 施密特功能置位寄存器	GPIOx_CSS 说明
0x4800_3074	GPIOD_CSC	GPIOD 施密特功能清除寄存器	GPIOx_CSC 说明
0x4800_3400 ~ 0x4800_37FC	GPIOD_MASKL	GPIOD 端口低 8 位映射操作区	GPIOx_MASKL 说明
0x4800_3800 ~ 0x4800_3BFC	GPIOD_MASKH	GPIOD 端口高 8 位映射操作区	GPIOx_MASKH 说明

5.5 AFIO 寄存器描述

请参考第 1 章中用到的缩写。
必须以字(32 位)的方式操作这些外设寄存器。

5.5.1 端口数字功能配置寄存器 0(AFIOx_AFS0) (x=A..D)

（地址： AFIOA：0x4800_0010；AFIOB：0x4800_1010；
AFIOC：0x4800_2010；AFIOD：0x4800_3010）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	IO7[2:0]			-	IO6[2:0]			-	IO5[2:0]			-	IO4[2:0]		
R/W	Res	RW	RW	RW	Res	RW	RW	RW	Res	RW	RW	RW	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	IO3[2:0]			-	IO2[2:0]			-	IO1[2:0]			-	IO0[2:0]		
R/W	Res	RW	RW	RW	Res	RW	RW	RW	Res	RW	RW	RW	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 31：0	**IOy[2:0]：** 端口 x 引脚 y 复用功能配置**(Port x pin y digtal alternate function)** 软件配置这些位以选择端口 x 引脚 y 的复用功能。 000：不复用，端口 x 引脚 y 为主功能 001：复用功能 0 …… 111：复用功能 6

5.5.2 端口数字功能配置寄存器 1(AFIOx_AFS1) (x=A..D)

（地址： AFIOA：0x4800_0014；AFIOB：0x4800_1014；
AFIOC：0x4800_2014；AFIOD：0x4800_3014）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	IO15[2:0]			-	IO14[2:0]			-	IO13[2:0]			-	IO12[2:0]		
R/W	Res	RW	RW	RW	Res	RW	RW	RW	Res	RW	RW	RW	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	IO11[2:0]			-	IO10[2:0]			-	IO9[2:0]			-	IO8[2:0]		
R/W	Res	RW	RW	RW	Res	RW	RW	RW	Res	RW	RW	RW	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 31：0	**IOy[2:0]：** 端口 x 引脚 y 复用功能配置**(Port x pin y digtal alternate function)** 软件配置这些位以选择端口 x 引脚 y 的复用功能。 000：不复用，端口 x 引脚 y 为主功能 001：复用功能 0 …… 111：复用功能 6

5.5.3 端口数字功能清除寄存器(AFIOx_AFC) (x=A..D)

（地址： AFIOA：0x4800_0018；AFIOB：0x4800_1018；
AFIOC：0x4800_2018；AFIOD：0x4800_3018）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**IOy：** 端口 x 引脚 y 数字功能清除**(Port x pin y digtal alternate function clear)** 0： 无效操作 1： 端口 x 引脚 y 数字功能清除、自动切换到主功能 ***注：切换不同的数字复用功能前，必须在(GPIOx_AFC)中先将之前的复用清除。***
位 31：16	保留，必须保持为 0。

5.5.4 端口模拟功能使能寄存器(AFIOx_ANAS) (x=A..D)

（地址： AFIOA：0x4800_0058；AFIOB：0x4800_1058；
AFIOC：0x4800_2058；AFIOD：0x4800_3058）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**IOy：** 端口 x 引脚 y 模拟功能使能**(Port x pin y analog alternate function)** 0： 端口 x 引脚 y 模拟功能失能，该位写 0 无效 1： 端口 x 引脚 y 模拟功能使能
位 31：16	保留，必须保持为 **0**。

5.5.5 端口模拟功能失能寄存器(AFIOx_ANAC) (x=A..D)

（地址： AFIOA：0x4800_005C；AFIOB：0x4800_105C；
AFIOC：0x4800_205C；AFIOD：0x4800_305C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IO15	IO14	IO13	IO12	IO11	IO10	IO9	IO8	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**IOy：** 端口 x 引脚 y 模拟功能失能**(Port x pin y analog alternate function clear)** 0： 无效操作 1： 端口 x 引脚 y 模拟功能失能、自动切换到数字主功能
位 31：16	保留，必须保持为 **0**。

5.5.6 寄存器列表

地址	寄存器	描述	备注
0x4800_0010	AFIOA_AFS0	AFIO 端口数字功能配置寄存器 0	AFIOx_AFS0 说明
0x4800_0014	AFIOA_AFS1	AFIO 端口数字功能配置寄存器 1	AFIOx_AFS1 说明
0x4800_0018	AFIOA_AFC	AFIO 端口数字功能清除寄存器	AFIOx_AFC 说明
0x4800_0058	AFIOA_ANAS	AFIO 端口模拟功能使能寄存器	AFIOx_ANAS 说明
0x4800_005C	AFIOA_ANAC	AFIO 端口模拟功能失能寄存器	AFIOx_ANAC 说明
地址	寄存器	描述	备注
0x4800_1010	AFIOB_AFS0	AFIO 端口数字功能配置寄存器 0	AFIOx_AFS0 说明
0x4800_1014	AFIOB_AFS1	AFIO 端口数字功能配置寄存器 1	AFIOx_AFS1 说明
0x4800_1018	AFIOB_AFC	AFIO 端口数字功能清除寄存器	AFIOx_AFC 说明
0x4800_1058	AFIOB_ANAS	AFIO 端口模拟功能使能寄存器	AFIOx_ANAS 说明
0x4800_105C	AFIOB_ANAC	AFIO 端口模拟功能失能寄存器	AFIOx_ANAC 说明
地址	寄存器	描述	备注
0x4800_2010	AFIOC_AFS0	AFIO 端口数字功能配置寄存器 0	AFIOx_AFS0 说明
0x4800_2014	AFIOC_AFS1	AFIO 端口数字功能配置寄存器 1	AFIOx_AFS1 说明
0x4800_2018	AFIOC_AFC	AFIO 端口数字功能清除寄存器	AFIOx_AFC 说明
0x4800_2058	AFIOC_ANAS	AFIO 端口模拟功能使能寄存器	AFIOx_ANAS 说明
0x4800_205C	AFIOC_ANAC	AFIO 端口模拟功能失能寄存器	AFIOx_ANAC 说明
地址	寄存器	描述	备注
0x4800_3010	AFIOD_AFS0	AFIO 端口数字功能配置寄存器 0	AFIOx_AFS0 说明
0x4800_3014	AFIOD_AFS1	AFIO 端口数字功能配置寄存器 1	AFIOx_AFS1 说明
0x4800_3018	AFIOD_AFC	AFIO 端口数字功能清除寄存器	AFIOx_AFC 说明
0x4800_3058	AFIOD_ANAS	AFIO 端口模拟功能使能寄存器	AFIOx_ANAS 说明
0x4800_305C	AFIOD_ANAC	AFIO 端口模拟功能失能寄存器	AFIOx_ANAC 说明

6 嵌套向量中断控制器(NVIC)

6.1 综述

为了减少延迟并提高中断处理效率，Cortex®-M0 嵌套了向量中断控制器 NVIC。
所有外设中断的请求均在内部被连接到 NVIC，只有在 NVIC 中使能了相应的外设中断，相应的外设中断才能够被系统响应。
NVIC 对系统中断及外设中断提供的控制包括如使能/失能控制、优先级、清除、挂起、软件触发等功能，NVIC 和 M0 内核的接口紧密相连，可以实现低延迟、高效能的中断处理；
M0 内核还内置了简单的 24bit 向下计数定时器（SysTick）；
SysTick 可以用来作为实时操作系统（RTOS）的节拍定时器，或者作为一个简单的计数器。SysTick 从预设值向下计数，当它到达零时，产生一个系统中断，详细信息请参考《Cortex™-M0 技术参考手册》。
额外的，内核还提供了中断屏蔽寄存器组(PRIMASK，FAULTMASK 和 BASEPRI)可用于屏蔽/开放所有中断。
用户可以直接通过 CMSIS-Core 封装的内核寄存器访问函数：“__disable_irq()”和“__enable_irq()”函数来屏蔽/开放所有中断，详细信息请参考《Cortex-M0 设备通用用户指南》和《CMSIS-Core (Cortex-M) Version 5.5.0》

6.2 特性

- 多达 32 个可屏蔽的外设中断(不包含 16 个 Cortex™-M0 中断)
- 4 个可编程的优先等级(使用了 2 位中断优先级)
- IWDG 中断连接到不可屏蔽的 NMI 向量
- 内置的 24bit 系统定时器 Systick
  - 向下计数
  - 自动重载
  - 当计数器计数到 0，产生可屏蔽中断
  - Systick 的参考时钟来自 HCLK

6.3 中断和异常向量

下面的表格列出了 PT32x00x 产品的向量表，表中列出 16 个系统中断类型和最多 32 个的外设中断。

表 6-1 中断向量表

异常类型	中断号	优先级	名称	说明	地址
-	-	-	-	保留	0x00
1	-	固定	Reset	复位	0x04
2	-14	固定	NMI	不可屏蔽中断 **IWDG 中断连接到 NMI 向量**	0x08
3	-13	固定	HardFault	所有类型的故障	0x0C
4~10	-	-	-	保留	0x10~0x28
11	-5	可设置	SVCall	通过 SWI 指令的系统服务调用	0x2c
12~13	-	-	-	保留	0x30~0x24
14	-2	可设置	PendSV	可挂起的系统服务	0x38
15	-1	可设置	SysTick	SysTick 定时器中断	0x3C
16	0	可设置		保留	0x40
17	1	可设置	PLLFAIL	PLL 时钟失效中断	0x44
18	2	可设置	-	保留	0x48
19	3	可设置	IFMC	IFMC 全局中断	0x4C
20	4	可设置	-	保留	0x50
21	5	可设置	EXTIA	外部中断 GPIOA 线	0x54
22	6	可设置	EXTIB	外部中断 GPIOB 线	0x58
23	7	可设置	EXTIC	外部中断 GPIOC 线	0x5c
24	8	可设置	EXTID	外部中断 GPIOD 线	0x60
25	9	可设置	-	保留	0x64
26	10	可设置	-	保留	0x68
27	11	可设置	-	保留	0x6c
28	12	可设置	ADC	ADC 全局中断	0x70
29	13	可设置	TIM1	TIM1 全局中断	0x74
30	14	可设置	-	保留	0x78
31	15	可设置	TIM4	TIM4 全局中断	0x7c
32	16	可设置	TIM3	TIM3 全局中断	0x80
33	17	可设置	TIM2	TIM2 全局中断	0x84
34	18	可设置	-	保留	0x88
35	19	可设置	-	保留	0x8c
36	20	可设置	PVD	电源电压检测(PVD)中断	0x90
37	21	可设置	-	保留	0x94
38	22	可设置		保留	0x98
39	23	可设置	I2C	I2C 全局中断	0x9C

40	24	可设置	-	保留	0xA0
41	25	可设置	SPI	SPI 全局中断	0xA4
42	26	可设置	-	保留	0xA8
43	27	可设置	UART0	UART0 全局中断	0xAC
44	28	可设置	UART1	UART1 全局中断	0xB0
45	29	可设置	-	保留	0xB4
46	30	可设置	-	保留	0xB8
47	31	可设置	-	保留	0xBC

6.4 NVIC 寄存器描述

6.4.1 系统中断使能寄存器 ISER

（地址： 0xE000_E100）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	UART1	UART0	-	SPI	-	I2C	-	-	PVD	-	-	TIM2	TIM3
R/W	Res	Res	Res	Rw1	Rw1	Res	Rw1	Res	Rw1	Res	RW	Rw1	Res	Res	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TIM4	-	TIM1	ADC	-	-	-	EXTID	EXTIC	EXTIB	EXTIA	-	IFMC	-	PLL FAIL	-
R/W	Rw1	Res	Rw1	Rw1	Res	Res	Res	Rw1	Rw1	Rw1	Rw1	Res	Rw1	Res	Rw1	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	保留，必须保持为 0。
位 1	**PLLFAIL：PLL 时钟失效中断使能设定位(PLL fail interrupt enable)** 0： PLL 时钟失效中断禁止，该位写 0 无效 1： PLL 时钟失效中断使能
位 2	保留，必须保持为 0。
位 3	**IFMC：IFMC 控制中断使能(IFMC interrupt enable)** 0： IFMC 中断禁止，该位写 0 无效 1： IFMC 中断使能
位 4	保留，必须保持为 0。
位 5	**EXTIA：EXTIA 端口中断使能(EXTIA interrupt enable)** 0： EXTIA 端口中断禁止，该位写 0 无效 1： EXTIA 端口中断使能
位 6	**EXTIB：EXTIB 端口中断使能(EXTIB interrupt enable)** 0： EXTIB 端口中断禁止，该位写 0 无效 1： EXTIB 端口中断使能
位 7	**EXTIC：EXTIC 端口中断使能(EXTIC interrupt enable)** 0： EXTIC 端口中断禁止，该位写 0 无效 1： EXTIC 端口中断使能
位 8	**EXTID： EXTID 端口中断使能(EXTID interrupt enable)** 0： EXTID 端口中断禁止，该位写 0 无效 1： EXTID 端口中断使能
位 11：9	保留，必须保持为 0。
位 12	**ADC： ADC 中断使能(ADC interrupt enable)**

	0： ADC 中断禁止，该位写 0 无效 1： ADC 中断使能
位 13	TIM1： TIM1 中断使能(TIM1 interrupt enable) 0： TIM1 中断禁止，该位写 0 无效 1： TIM1 中断使能
位 14	保留，必须保持为 0。
位 15	TIM4： TIM4 中断使能(TIM4 interrupt enable) 0： TIM4 中断禁止，该位写 0 无效 1： TIM4 中断使能
位 16	TIM3： TIM3 中断使能(TIM3 interrupt enable) 0： TIM3 中断禁止，该位写 0 无效 1： TIM3 中断使能
位 17	TIM2： TIM2 中断使能(TIM2 interrupt enable) 0： TIM2 中断禁止，该位写 0 无效 1： TIM2 中断使能
位 19：18	保留，必须保持为 0。
位 20	PVD： PVD 中断使能(PVD interrupt enable) 0： PVD 中断禁止，该位写 0 无效 1： PVD 中断使能
位 22：21	保留，必须保持为 0。
位 23	I2C： I2C 中断使能(I2C interrupt enable) 0： I2C 中断禁止，该位写 0 无效 1： I2C 中断使能
位 24	保留，必须保持为 0。
位 25	SPI： SPI 中断使能(SPI interrupt enable) 0： SPI 中断禁止，该位写 0 无效 1： SPI 中断使能
位 26	保留，必须保持为 0。
位 27	UART0： UART0 中断使能(UART0 interrupt enable) 0： UART0 中断禁止，该位写 0 无效 1： UART0 中断使能
位 28	UART1： UART1 中断使能(UART1 interrupt enable) 0： UART1 中断禁止，该位写 0 无效 1： UART1 中断使能
位 31：29	保留，必须保持为 0。

6.4.2 系统中断禁止寄存器 ICER

（地址： 0xE000_E180）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	UART1	UART0	-	SPI	-	I2C	-	-	PVD	-	-	TIM2	TIM3
R/W	Res	Res	Res	Rw1	Rw1	Res	Rw1	Res	Rw1	Res	Res	Rw1	Res	Res	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TIM4	-	TIM1	ADC	-	-	-	EXTID	EXTIC	EXTIB	EXTIA	-	IFMC	-	PLL FAIL	-
R/W	Rw1	Res	Rw1	Rw1	Res	Res	Res	Rw1	Rw1	Rw1	Rw1	Res	Rw1	Res	Rw1	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	保留，必须保持为 0。
位 1	**PLL_FAIL：PLL 时钟失效中断禁止(PLL fail interrupt disable)** 0： PLL 时钟失效中断使能，该位写 0 无效 1： PLL 时钟失效中断禁止
位 2	保留，必须保持为 0。
位 3	**IFMC：IFMC 控制中断禁止(IFMC interrupt disable)** 0： IFMC 中断使能，该位写 0 无效 1： IFMC 中断禁止
位 4	保留，必须保持为 0。
位 5	**EXTIA： EXTIA 端口中断禁止(EXTIA interrupt disable)** 0： EXTIA 端口中断使能，该位写 0 无效 1： EXTIA 端口中断禁止
位 6	**EXTIB： EXTIB 端口中断禁止(EXTIB interrupt disable)** 0： EXTIB 端口中断使能，该位写 0 无效 1： EXTIB 端口中断禁止
位 7	**EXTIC： EXTIC 端口中断禁止(EXTIC interrupt disable)** 0： EXTIC 端口中断使能，该位写 0 无效 1： EXTIC 端口中断禁止
位 8	**EXTID： EXTID 端口中断禁止(EXTID interrupt disable)** 0： EXTID 端口中断使能，该位写 0 无效 1： EXTID 端口中断禁止
位 11：9	保留，必须保持为 0。
位 12	**ADC： ADC 中断禁止(ADC interrupt disable)** 0： ADC 中断使能，该位写 0 无效 1： ADC 中断禁止
位 13	**TIM1： TIM1 中断禁止(TIM1 interrupt disable)**

	0： TIM1 中断使能，该位写 0 无效 1： TIM1 中断禁止
位 14	保留，必须保持为 0。
位 15	TIM4： TIM4 中断禁止(TIM4 interrupt disable) 0： TIM4 中断使能，该位写 0 无效 1： TIM4 中断禁止
位 16	TIM3： TIM3 中断禁止(TIM3 interrupt disable) 0： TIM3 中断使能，该位写 0 无效 1： TIM3 中断禁止
位 17	TIM2： TIM2 中断禁止(TIM2 interrupt disable) 0： TIM2 中断使能，该位写 0 无效 1： TIM2 中断禁止
位 19：18	保留，必须保持为 0。
位 20	PVD： PVD 中断禁止(PVD interrupt disable) 0： PVD 中断使能，该位写 0 无效 1： PVD 中断禁止
位 22：21	保留，必须保持为 0。
位 23	I2C： I2C 中断禁止(I2C interrupt disable) 0： I2C 中断使能，该位写 0 无效 1： I2C 中断禁止
位 24	保留，必须保持为 0。
位 25	SPI： SPI 中断禁止(SPI interrupt disable) 0： SPI 中断使能，该位写 0 无效 1： SPI 中断禁止
位 26	保留，必须保持为 0。
位 27	UART0： UART0 中断禁止(UART0 interrupt disable) 0： UART0 中断使能，该位写 0 无效 1： UART0 中断禁止
位 28	UART1： UART1 中断禁止(UART1 interrupt disable) 0： UART1 中断使能，该位写 0 无效 1： UART1 中断禁止
位 31：29	保留，必须保持为 0。

6.4.3 系统中断挂起设定寄存器 ISPR

（地址： 0xE000_E200）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	UART1	UART0	-	SPI	-	I2C	-	-	PVD	-	-	TIM2	TIM3
R/W	Res	Res	Res	Rw1	Rw1	Res	Rw1	Res	Rw1	Res	Res	Rw1	Res	Res	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TIM4	-	TIM1	ADC	-	-	-	EXTID	EXTIC	EXTIB	EXTIA	-	IFMC	-	PLL FAIL	-
R/W	Rw1	Res	Rw1	Rw1	Res	Res	Res	Rw1	Rw1	Rw1	Rw1	Res	Rw1	Res	Rw1	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	保留，必须保持为 0。
位 1	**PLL_FAIL：PLL 时钟失效中断挂起等待(PLL fail interrupt pending)** 0： PLL 时钟失效中断无请求，该位写 0 无效 1： PLL 时钟失效中断挂起等待
位 2	保留，必须保持为 0。
位 3	**IFMC：IFMC 控制中断挂起等待(IFMC interrupt pending)** 0： IFMC 中断无请求，该位写 0 无效 1： IFMC 中断挂起等待
位 4	保留，必须保持为 0。
位 5	**EXTIA： EXTIA 端口中断挂起等待(EXTIA interrupt pending)** 0： EXTIA 端口中断无请求，该位写 0 无效 1： EXTIA 端口中断挂起等待
位 6	**EXTIB：EXTIB 端口中断挂起等待(EXTIB interrupt pending)** 0： PB 端口中断无请求，该位写 0 无效 1： PB 端口中断挂起等待
位 7	**EXTIC： EXTIC 端口中断挂起等待(EXTIC interrupt pending)** 0： EXTIC 端口中断无请求，该位写 0 无效 1： EXTIC 端口中断挂起等待
位 8	**EXTID： EXTID 端口中断挂起等待(EXTID interrupt pending)** 0： EXTID 端口中断无请求，该位写 0 无效 1： EXTID 端口中断挂起等待
位 11：9	保留，必须保持为 0。
位 12	**ADC： ADC 中断挂起等待(ADC interrupt pending)** 0： ADC 中断无请求，该位写 0 无效 1： ADC 中断挂起等待
位 13	**TIM1： TIM1 中断挂起等待(TIM1 interrupt pending)**

	0： TIM1 中断无请求，该位写 0 无效 1： TIM1 中断挂起等待
位 14	保留，必须保持为 0。
位 15	TIM4： TIM4 中断挂起等待(TIM4 interrupt pending) 0： TIM4 中断无请求，该位写 0 无效 1： TIM4 中断挂起等待
位 16	TIM3： TIM3 中断挂起等待(TIM3 interrupt pending) 0： TIM3 中断无请求，该位写 0 无效 1： TIM3 中断挂起等待
位 17	TIM2： TIM2 中断挂起等待(TIM2 interrupt pending) 0： TIM2 中断无请求，该位写 0 无效 1： TIM2 中断挂起等待
位 19：18	保留，必须保持为 0。
位 20	PVD： PVD 中断挂起等待(PVD interrupt pending) 0： PVD 中断无请求，该位写 0 无效 1： PVD 中断挂起等待
位 22：21	保留，必须保持为 0。
位 23	I2C： I2C 中断挂起等待(I2C interrupt pending) 0： I2C 中断无请求，该位写 0 无效 1： I2C 中断挂起等待
位 24	保留，必须保持为 0。
位 25	SPI： SPI 中断挂起等待(SPI interrupt pending) 0： SPI 中断无请求，该位写 0 无效 1： SPI 中断挂起等待
位 26	保留，必须保持为 0。
位 27	UART0： UART0 中断挂起等待(UART0 interrupt pending) 0： UART0 中断无请求，该位写 0 无效 1： UART0 中断挂起等待
位 28	UART1： UART1 中断挂起等待(UART1 interrupt pending) 0： UART1 中断无请求，该位写 0 无效 1： UART1 中断挂起等待
位 31：29	保留，必须保持为 0。

6.4.4 系统中断挂起清除寄存器 ICPR

（地址： 0xE000_E280）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	UART1	UART0	-	SPI	-	I2C	-	-	PVD	-	-	TIM2	TIM3
R/W	Res	Res	Res	Rw1	Rw1	Res	Rw1	Res	Rw1	Res	Res	Rw1	Res	Res	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TIM4	-	TIM1	ADC	-	-	-	EXTID	EXTIC	EXTIB	EXTIA	-	IFMC	-	PLL FAIL	-
R/W	Rw1	Res	Rw1	Rw1	Res	Res	Res	Rw1	Rw1	Rw1	Rw1	Res	Rw1	Res	Rw1	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	保留，必须保持为 0。
位 1	**PLL_FAIL：PLL 时钟失效中断挂起清除(PLL fail interrupt pending clear)** 0： 无意义 1： PLL 时钟失效中断挂起清除
位 2	保留，必须保持为 0。
位 3	**IFMC：IFMC 控制中断挂起清除(IFMC interrupt pending clear)** 0： IFMC 中断无请求，该位写 0 无效 1： IFMC 中断挂起清除
位 4	保留，必须保持为 0。
位 5	**EXTIA： EXTIA 端口中断挂起清除(EXTIA interrupt pending clear)** 0： EXTIA 端口中断无请求，该位写 0 无效 1： EXTIA 端口中断挂起清除
位 6	**EXTIB：EXTIB 端口中断挂起清除(EXTIB interrupt pending clear)** 0： EXTIB 端口中断无请求，该位写 0 无效 1： EXTIB 端口中断挂起清除
位 7	**EXTIC： EXTIC 端口中断挂起清除(EXTIC interrupt pending clear)** 0： EXTIC 端口中断无请求，该位写 0 无效 1： EXTIC 端口中断挂起清除
位 8	**EXTID： EXTID 端口中断挂起清除(EXTID interrupt pending clear)** 0： EXTID 端口中断无请求，该位写 0 无效 1： EXTID 端口中断挂起清除
位 11：9	保留，必须保持为 0。
位 12	**ADC： ADC 中断挂起清除(ADC interrupt pending clear)** 0： ADC 中断无请求，该位写 0 无效 1： ADC 中断挂起清除
位 13	**TIM1： TIM1 中断挂起清除(TIM1 interrupt pending clear)**

	0： TIM1 中断无请求，该位写 0 无效 1： TIM1 中断挂起清除
位 14	保留，必须保持为 0。
位 15	TIM4： TIM4 中断挂起清除(TIM4 interrupt pending clear) 0： TIM4 中断无请求，该位写 0 无效 1： TIM4 中断挂起清除
位 16	TIM3： TIM3 中断挂起清除(TIM3 interrupt pending clear) 0： TIM3 中断无请求，该位写 0 无效 1： TIM3 中断挂起清除
位 17	TIM2： TIM2 中断挂起清除(TIM2 interrupt pending clear) 0： TIM2 中断无请求，该位写 0 无效 1： TIM2 中断挂起清除
位 19：18	保留，必须保持为 0。
位 20	PVD： PVD 中断挂起清除(PVD interrupt pending clear) 0： PVD 中断无请求，该位写 0 无效 1： PVD 中断挂起清除
位 22：21	保留，必须保持为 0。
位 23	I2C： I2C 中断挂起清除(I2C interrupt pending clear) 0： I2C 中断无请求，该位写 0 无效 1： I2C 中断挂起清除
位 24	保留，必须保持为 0。
位 25	SPI： SPI 中断挂起清除(SPI interrupt pending clear) 0： SPI 中断无请求，该位写 0 无效 1： SPI 中断挂起清除
位 26	保留，必须保持为 0。
位 27	UART0： UART0 中断挂起清除(UART0 interrupt pending clear) 0： UART0 中断无请求，该位写 0 无效 1： UART0 中断挂起清除
位 28	UART1： UART1 中断挂起清除(UART1 interrupt pending clear) 0： UART1 中断无请求，该位写 0 无效 1： UART1 中断挂起清除
位 31：29	保留，必须保持为 0。

6.4.5 系统中断优先级寄存器 IPRx

x：表示 IPR0~IPR7。

(地址： IPR0：0xE000_E400，IPR1：0xE000_E404，IPR2：0xE000_E408
IPR3：0xE000_E40C，IPR4：0xE000_E410，IPR5：0xE000_E414
IPR6：0xE000_E418，IPR7：0xE000_E41C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	PRI_(4x+3)								PRI_(4x+2)							
R/W	RW	RW	R	R	R	R	R	R	RW	RW	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	PRI_(4x+1)								PRI_(4x+0)							
R/W	RW	RW	R	R	R	R	R	R	RW	RW	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 7：0 **位 15：8** **位 23：16** **位 31：24**	**PRI_(4x+y)**：中断号 **IRQn** 的优先级配置**(priority of the interrupt number)** n：表示对应的 IRQ 中断号,n=(4x+y) y：表示 0~3，一个 IPRx 寄存器可以配置 4 个中断号的优先级 x：表示 IPR0~IPR7，8 个 IPR 寄存器，每个寄存器可配置 4 个中断号的优先级，支持最大 32 个外设中断的优先级配置 每个 PRI_(4x+y)的 8 个位中只有最高两位[7：6]有效，四级优先级，数值越低优先级越高，其中 0 的优先级是最高的。 例： 配置 TIM4 的中断优先级，TIM4 的中断号为 15，故 15=(4*3+3)，应当配置 IPR3 的 PRI_(4x+3)； 配置 UART0 的中断优先级，UART0 的中断号为 27，故 27=(4*6+3)，应当配置 IPR6 的 PRI_(4x+3)； ***注：*** ***1. 中断优先级寄存器的配置必须在中断使能之前完成*** ***2. 写 PRI_(4x+(0~3))的低 5 位[5：0]无效，且读恒为 0***

6.4.6 NVIC 寄存器列表

地址	寄存器	描述	备注
0xE000_E010	SYST_CSR	SysTick 控制状态寄存器	了解更多 NVIC 寄存器相关信息，请参考《Cortex™-M0 技术参考手册》
0xE000_E014	SYST_RVR	SysTick 重载寄存器	
0xE000_E018	SYST_CVR	SysTick 当前计数值寄存器	
0xE000_E01C	SYST_CALIB	SysTick 校准值寄存器	
0xE000_E100	ISER	系统中断使能寄存器	ISER 说明
0xE000_E180	ICER	系统中断禁止寄存器	ICER 说明
0x E000_E200	ISPR	系统中断挂起设定寄存器	ISPR 说明
0xE000_E280	ICPR	系统中断挂起清除寄存器	ICPR 说明
0xE000_E400	IPR0	系统中断 0~3 优先级寄存器	IPRx 说明
0xE000_E404	IPR1	系统中断 4~7 优先级寄存器	IPRx 说明
0xE000_E408	IPR2	系统中断 8~11 优先级寄存器	IPRx 说明
0xE000_E40C	IPR3	系统中断 12~15 优先级寄存器	IPRx 说明
0xE000_E410	IPR4	系统中断 16~19 优先级寄存器	IPRx 说明
0xE000_E414	IPR5	系统中断 20~23 优先级寄存器	IPRx 说明
0xE000_E418	IPR6	系统中断 24~27 优先级寄存器	IPRx 说明
0xE000_E41C	IPR7	系统中断 28~31 优先级寄存器	IPRx 说明

7 外部中断控制器(EXTI)

7.1 综述

每个 GPIO 内部都集成了一个独立的“电平和边沿检测器”，在 GPIO 配置为输入模式的前提下，当外部输入信号满足这些检测器的触发条件时、这些检测器都可以产生中断请求，它们共同构成了外部中断控制器(EXTI)。

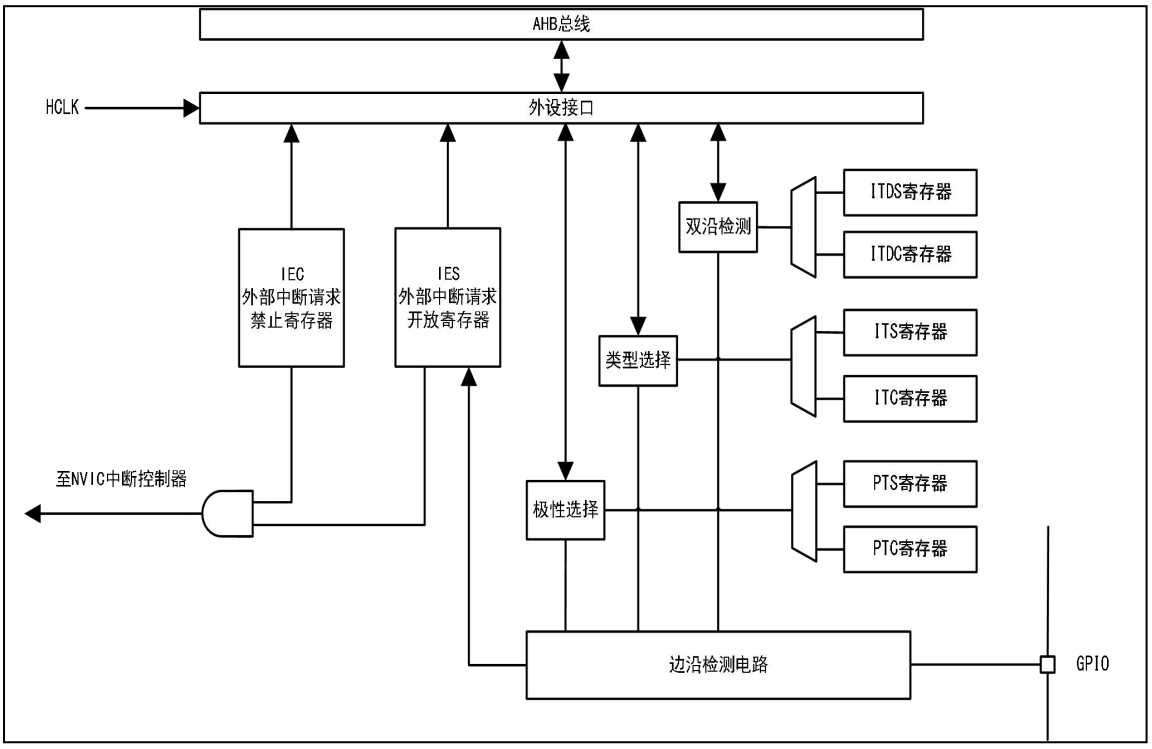
每个检测器的中断请求都可以被单独的使能与失能，用户可以独立的配置检测器中断的触发类型(上升沿、下降沿、高电平、低电平、双沿)，EXTI 中断标志寄存器保持了相应检测器的中断请求。

7.2 特性

- 以下五种信号类型可被电平和边沿检测器识别，并产生中断：
  - 上升沿信号
  - 下降沿信号
  - 高电平信号
  - 低电平信号
  - 双沿信号(任意电平沿)
- 边沿检测器支持对频率高达 20Mhz 的外部信号进行检测。
- GPIO 方向为输出时，边沿检测器仍可正常工作
- GPIO 功能被复用时，边沿检测器仍可正常工作

7.3 EXTI 功能描述

图 7-1 外部中断框图



7.3.1 有效的外部中断

每个 GPIO 内部都集成了一个独立的“电平和边沿检测器”，EXTI 相关寄存器的低 16 位用于配置相关 GPIO 的外部中断，对不存在的 GPIO 引脚所配置的信息将不会被生效；

PT32x00x 定义的 GPIO 信息参考《PT32x00x 数据手册》中的“引脚定义”部分。

7.3.2 外部中断唤醒低功耗模式

通过外部中断将内核从低功耗模式唤醒(WFE/WFI)，执行下列操作，以获得最优功耗：

- 正确的配置外部中断，外部中断类型设为上升沿或高电平触发，相关引脚外部或内部下拉。
- 保障”3.4 低功耗模式”中，优化功耗的方式被正确执行
- 此时可以关闭内部 LSI，以获得最优功耗。

注意：GPIO 外部中断唤醒深度睡眠时，仅支持双沿中断、高电平中断和低电平中断。

7.3.3 外部中断的配置

外部中断支持对 5 种信号类型进行检测，按照信号类型，可以分成三类信号：

- “沿”信号，上升沿或下降沿
- “电平”信号，高电平或低电平
- “双沿”信号，任意电平沿

EXTI 提供了 6 个寄存器用于配置外部信号的检测类型，它们分别是：

- 外部中断类型设置寄存器(EXTIx_ITS)
- 外部中断类型清除寄存器(EXTIx_ITC)
- 外部中断双沿类型配置寄存器(EXTIx_ITDS)
- 外部中断双沿类型清除寄存器(EXTIx_ITDC)
- 外部中断极性配置寄存器(EXTIx_PTS)
- 外部中断极性清除寄存器(EXTIx_PTC)

外部中断检测的信号类型与寄存器的配置关系如下表：

表 7-1

	类型配置		双沿配置		极性配置	
信号类型	ITS	ITC	ITDS	ITDC	PTS	PTC
上升沿	1	0	0	1	1	0
下降沿	1	0	0	1	0	1
高电平	0	1	0	1	1	0
低电平	0	1	0	1	0	1
双沿	X	X	1	0	X	X

7.4 EXTI 寄存器描述

7.4.1 外部中断请求开放寄存器(EXTIx_IES) (x=A..D)

（地址： EXTIA：0x4800_001C；EXTIB：0x4800_101C；
EXTIC：0x4800_201C；EXTID：0x4800_301C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**IEy：** 端口 x 引脚 y 输入中断的请求**(port x pin y input interrupt request)** 0： 禁止端口 x 引脚 y 的输入中断请求，该位写 0 无效 1： 开放端口 x 引脚 y 的输入中断请求
位 31：16	保留，必须保持为 0。

7.4.2 外部中断请求禁止寄存器(EXTIx_IEC) (x=A..D)

（地址： EXTIA：0x4800_0020；EXTIB：0x4800_1020；
EXTIC：0x4800_2020；EXTID：0x4800_3020）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8	IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**IEy：**端口 x 引脚 y 输入中断的请求**(port x pin y input interrupt request)** 0： 无效操作 1： 禁止端口 x 引脚 y 的输入中断请求
位 31：16	保留，必须保持为 0。

7.4.3 外部中断类型配置寄存器(EXTIx_ITS) (x=A..D)

（地址： EXTIA：0x4800_0024；EXTIB：0x4800_1024；
EXTIC：0x4800_2024；EXTID：0x4800_3024）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IT15	IT14	IT13	IT12	IT11	IT10	IT9	IT8	IT7	IT6	IT5	IT4	IT3	IT2	IT1	IT0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	ITy：端口 x 引脚 y 输入中断的请求类型(port x pin y input interrupt request type) 0： 端口 x 引脚 y 的中断类型配置为“电平”类型，该位写 0 无效 1： 端口 x 引脚 y 的中断类型配置为“沿”类型。
位 31：8	保留，必须保持为 0。

7.4.4 外部中断类型清除寄存器(EXTIx_ITC) (x=A..D)

（地址： EXTIA：0x4800_0028；EXTIB：0x4800_1028；
EXTIC：0x4800_2028；EXTID：0x4800_3028）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IT15	IT14	IT13	IT12	IT11	IT10	IT9	IT8	IT7	IT6	IT5	IT4	IT3	IT2	IT1	IT0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	ITy：端口 x 引脚 y 输入中断的请求类型(port x pin y input interrupt request type) 0： 无效操作 1： 清除端口 x 引脚 y 的中断类型设置，并将端口 x 引脚 y 的中断类型配置为“电平”类型。
位 31：16	保留，必须保持为 0。

7.4.5 外部中断双沿类型配置寄存器(EXTIx_ITDS) (x=A..D)

（地址： EXTIA：0x4800_002C；EXTIB：0x4800_102C；
EXTIC：0x4800_202C；EXTID：0x4800_302C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IT15	IT14	IT13	IT12	IT11	IT10	IT9	IT8	IT7	IT6	IT5	IT4	IT3	IT2	IT1	IT0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	ITy：端口 x 引脚 y 输入中断的请求类型(port x pin y input interrupt request type) 0： 端口 x 引脚 y 的中断类型非“双沿”类型，该位写 0 无效 1： 端口 x 引脚 y 的中断类型配置为“双沿”类型。
位 31：16	保留，必须保持为 0。

7.4.6 外部中断双沿类型清除寄存器(EXTIx_ITDC) (x=A..D)

（地址： EXTIA：0x4800_0030；EXTIB：0x4800_1030；
EXTIC：0x4800_2030；EXTID：0x4800_3030）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IT15	IT14	IT13	IT12	IT11	IT10	IT9	IT8	IT7	IT6	IT5	IT4	IT3	IT2	IT1	IT0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	ITy：端口 x 引脚 y 输入中断的请求类型(port x pin y input interrupt request type) 0： 无效操作 1： 清除端口 x 引脚 y 的“双沿”中断类型设置，中断类型由“中断类型设置/清除寄存器”配置
位 31：16	保留，必须保持为 0。

7.4.7 外部中断极性配置寄存器(EXTIx_PTS) (x=A..D)

（地址： EXTIA：0x4800_0034；EXTIB：0x4800_1034；
EXTIC：0x4800_2034；EXTID：0x4800_3034）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	HT15	HT14	HT13	HT12	HT11	HT10	HT9	HT8	HT7	HT6	HT5	HT4	HT3	HT2	HT1	HT0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	HTy：端口 x 引脚 y 输入中断极性(port x pin y input interrupt polarity) 0： 端口 x 引脚 y 的输入中断极性设置为低电平或下降沿有效，该位写 0 无效 1： 端口 x 引脚 y 的输入中断极性设置为高电平或上升沿有效
位 31：16	保留，必须保持为 0。

7.4.8 外部中断极性清除寄存器(EXTIx_PTC) (x=A..D)

（地址： EXTIA：0x4800_0038；EXTIB：0x4800_1038；
EXTIC：0x4800_2038；EXTID：0x4800_3038）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	LT15	LT14	LT13	LT12	LT11	LT10	LT9	LT8	LT7	LT6	LT5	LT4	LT3	LT2	LT1	LT0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	LTy：端口 x 引脚 y 输入中断极性(port x pin y input interrupt polarity) 0： 无效操作 1： 清除端口 x 引脚 y 的触发极性设置，将端口 x 引脚 y 的输入中断极性设置为低电平或下降沿有效
位 31：16	保留，必须保持为 0。

7.4.9 外部中断标志寄存器(EXTIx_IF) (x=A..D)

（地址： EXTIA：0x4800_003C；EXTIB：0x4800_103C；
EXTIC：0x4800_203C；EXTID：0x4800_303C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IF15	IF14	IF13	IF12	IF11	IF10	IF9	IF8	IF7	IF6	IF5	IF4	IF3	IF2	IF1	IF0
R/W	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	IFy：端口 x 引脚 y 外部中断标志位(port x pin y input interrupt flag) 0： 端口 x 引脚 y 无外部中断发生 1： 端口 x 引脚 y 有外部中断发生,软件可对该位写 1 将其清除
位 31：16	保留，必须保持为 0。

7.4.10 寄存器列表

地址	寄存器	描述	备注
EXTIA			
0x4800_001C	EXTIA_IES	EXTIA 外部中断请求开放寄存器	EXTIx_IES 说明
0x4800_0020	EXTIA_IEC	EXTIA 外部中断请求禁止寄存器	EXTIx_IEC 说明
0x4800_0024	EXTIA_ITS	EXTIA 外部中断类型配置寄存器	EXTIx_ITS 说明
0x4800_0028	EXTIA_ITC	EXTIA 外部中断类型清除寄存器	EXTIx_ITC 说明
0x4800_002C	EXTIA_ITDS	EXTIA 外部中断双沿类型配置寄存器	EXTIx_ITDS 说明
0x4800_0030	EXTIA_ITDC	EXTIA 外部中断双沿类型清除寄存器	EXTIx_ITDC 说明
0x4800_0034	EXTIA_PTS	EXTIA 外部中断极性配置寄存器	EXTIx_HTE 说明
0x4800_0038	EXTIA_PTC	EXTIA 外部中断极性清除寄存器	EXTIx_LTE 说明
0x4800_003C	EXTIA_IF	EXTIA 外部中断标志寄存器	EXTIx_IF 说明
EXTIB			
0x4800_101C	EXTIB_IES	EXTIB 外部中断请求开放寄存器	EXTIx_IES 说明
0x4800_1020	EXTIB_IEC	EXTIB 外部中断请求禁止寄存器	EXTIx_IEC 说明
0x4800_1024	EXTIB_ITS	EXTIB 外部中断类型配置寄存器	EXTIx_ITS 说明
0x4800_1028	EXTIB_ITC	EXTIB 外部中断类型清除寄存器	EXTIx_ITC 说明
0x4800_102C	EXTIB_ITDS	EXTIB 外部中断双沿类型配置寄存器	EXTIx_ITDS 说明
0x4800_1030	EXTIB_ITDC	EXTIB 外部中断双沿类型清除寄存器	EXTIx_ITDC 说明
0x4800_1034	EXTIB_PTS	EXTIB 外部中断极性配置寄存器	EXTIx_HTE 说明
0x4800_1038	EXTIB_PTC	EXTIB 外部中断极性清除寄存器	EXTIx_LTE 说明
0x4800_103C	EXTIB_IF	EXTIB 外部中断标志寄存器	EXTIx_IF 说明
EXTIC			
0x4800_201C	EXTIC_IES	EXTIC 外部中断请求开放寄存器	EXTIx_IES 说明
0x4800_2020	EXTIC_IEC	EXTIC 外部中断请求禁止寄存器	EXTIx_IEC 说明
0x4800_2024	EXTIC_ITS	EXTIC 外部中断类型配置寄存器	EXTIx_ITS 说明
0x4800_2028	EXTIC_ITC	EXTIC 外部中断类型清除寄存器	EXTIx_ITC 说明
0x4800_202C	EXTIC_ITDS	EXTIC 外部中断双沿类型配置寄存器	EXTIx_ITDS 说明
0x4800_2030	EXTIC_ITDC	EXTIC 外部中断双沿类型清除寄存器	EXTIx_ITDC 说明
0x4800_2034	EXTIC_PTS	EXTIC 外部中断极性配置寄存器	EXTIx_HTE 说明
0x4800_2038	EXTIC_PTC	EXTIC 外部中断极性清除寄存器	EXTIx_LTE 说明
0x4800_203C	EXTIC_IF	EXTIC 外部中断标志寄存器	EXTIx_IF 说明
EXTID			
0x4800_301C	EXTID_IES	EXTID 外部中断请求开放寄存器	EXTIx_IES 说明
0x4800_3020	EXTID_IEC	EXTID 外部中断请求禁止寄存器	EXTIx_IEC 说明
0x4800_3024	EXTID_ITS	EXTID 外部中断类型配置寄存器	EXTIx_ITS 说明
0x4800_3028	EXTID_ITC	EXTID 外部中断类型清除寄存器	EXTIx_ITC 说明
0x4800_302C	EXTID_ITDS	EXTID 外部中断双沿类型配置寄存器	EXTIx_ITDS 说明

0x4800_3030	EXTID_ITDC	EXTID 外部中断双沿类型清除寄存器	EXTIx_ITDC 说明
0x4800_3034	EXTID_PTS	EXTID 外部中断极性配置寄存器	EXTIx_HTE 说明
0x4800_3038	EXTID_PTC	EXTID 外部中断极性清除寄存器	EXTIx_LTE 说明
0x4800_303C	EXTID_IF	EXTID 外部中断标志寄存器	EXTIx_IF 说明

8 模拟/数字转换(ADC)

8.1 综述

PT32x00x 有一个 12 位的逐次逼近型模数转换器 ADC，该 ADC 有多达 11 个通道，允许 ADC 测量 8 个外部和 3 个内部信号源。各通道的 A/D 转换可以按照单次、连续或定时器触发模式执行。ADC 的结果可以左对齐或右对齐方式存储在 16 位数据寄存器中。

ADC 的输入时钟由 PCLK 经分频产生,最大频率不得超过 10Mhz。

8.2 特性

- 12 位的模数转换分辨率
- 最快 19 个 ADC 时钟周期的采样转换时间
- 单次、连续或定时触发的转换模式
- 独立编程的采样时间
- 支持转换完成中断
- 带内嵌数据一致性的数据对齐
- 集成的多路高精度基准电压源
- ADC 参考电压：VDD
- ADC 模拟通道输入电压范围：0~VDD

8.3 功能描述

下图是 ADC 模块框图，表 1 为 ADC 引脚说明。

图 8-1 ADC 框图

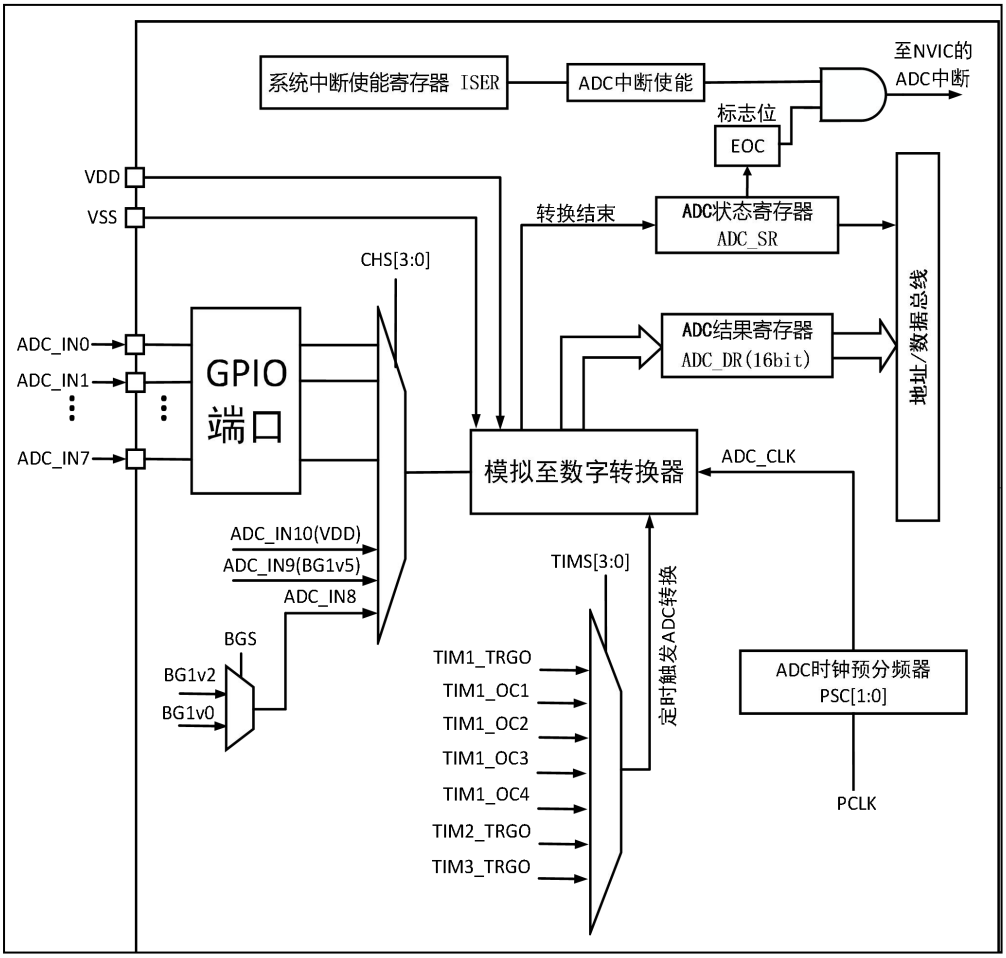


表 8-1 ADC 通道

名称	信号类型	描述
ADC_IN[7:0]	输入，外部模拟信号	8 个外部模拟输入通道
ADC_IN8(BGS)	输入，BG1v2(内部 1.2V 电压)	1.2V 内部带隙基准电压，精度 5%
	输入，BG1v0(内部 1.0V 电压)	1.0V 内部带隙基准电压，精度 0.5%
ADC_IN9(BG1v5)	输入，BG1v5(内部 1.5V 电压)	1.5V 内部带隙基准电压，精度 5%
ADC_IN10(VDD)	输入，电源	电源

8.3.1 ADC 开关控制

通过(ADC_CR 寄存器)的”ADEN”位可以将 ADC 使能；ADEN 位置 1 时、ADC 将从断电状态下唤醒。

ADC 上电延迟一段时间后(t_{RDY})、待内部参考源处于稳定状态，(ADC_SR 寄存器)的”RDY”位将会置 1，此时设置(ADC_CR 寄存器)的”SOC”位即可启动 ADC 转换。

通过清除 ADEN 位可以停止转换，并将 ADC 置于断电模式，在该模式，ADC 几乎不耗电(非睡眠模式时，仅几个 μA)。

注意：使用任意触发方式触发 ADC 转换前，都应优先保证 ADC 已经 RDY 完毕。

8.3.2 ADC 时钟

由时钟控制器提供的 ADC_CLK 时钟和 PCLK 时钟同步，ADC 内部自带了一个专用的可编程预分频器、可通过对(ADC_CR 寄存器)的”PSC”位的配置以分频。

注意：ADC_CLK 频率不得超过 10Mhz。

8.3.3 通道选择

多达 11 个通道可以选择，可以通过(ADC_CR 寄存器)的 CHS[3:0]位选择模拟输入通道。详见 ADC_CR 寄存器描述。

注意：ADC_DR 是公共的，如果在切换通道前不对 ADC_DR 内的数据进行读取保存，切换通道后，上一个通道保存在 ADC_DR 内的采样结果将丢失。

8.3.4 转换模式

ADC 有三种转换模式：

- 单次转换模式
- 连续转换模式
- 定时触发转换模式

软件可通过配置(ADC_CR 寄存器)的”MODE”位以选择相应的转换模式。

8.3.4.1 单次转换

使用该模式、需要将(ADC_CR 寄存器)的”MODE”位配置为 0x00。

单次转换模式下、软件将(ADC_CR 寄存器)的”SOC”置 1 以触发一次 ADC 单次转换，转换完成、则 ADC 停止转换，直到下一次 SOC 位被置 1。

每一次转换完成：

- 转换数据被储存在 16 位 ADC_DR 寄存器中
- EOC(转换完成)标志被设置
- 如果使能 NVIC 中的 ADC 中断，则产生中断。

8.3.4.2 连续转换

使用该模式、需要将(ADC_CR 寄存器)的”MODE”位配置为 0x02。
连续转换模式下，软件将 ADC_CR 的 SOC 位置 1 以触发一次 ADC 连续转换，转换完成、则 ADC 自动开始下一次转换。
每一次转换完成：

- 转换数据被储存在 16 位 ADC_DR 寄存器中
- EOC(转换完成)标志被设置
- 如果使能 NVIC 中的 ADC 中断，则产生中断。

8.3.4.3 定时触发转换

配置(ADC_CR 寄存器)的”MODE”位为 0x01，并选择定时器事件触发模式，转换可以由定时器的 OC 或者 TRGO 事件触发(例如定时器 TIM1，TIM2，TIM3)。
(ADC_CR 寄存器)的”TIMS”位允许选择 ADC 定时触发源中 7 个可能的事件以触发 ADC 采样。详细可查看定时器章节对于下列信号的描述。
表 8-2 ADC 定时触发源选择

触发源	描述	TIMS[3:0]
TIM1_TRGO 事件	来自片上定时器的内部信号	0101
TIM1_OC1 事件		0110
TIM1_OC2 事件		0111
TIM1_OC3 事件		1000
TIM1_OC4 事件		1001
TIM2_TRGO 事件		1010
TIM3_TRGO 事件		1011

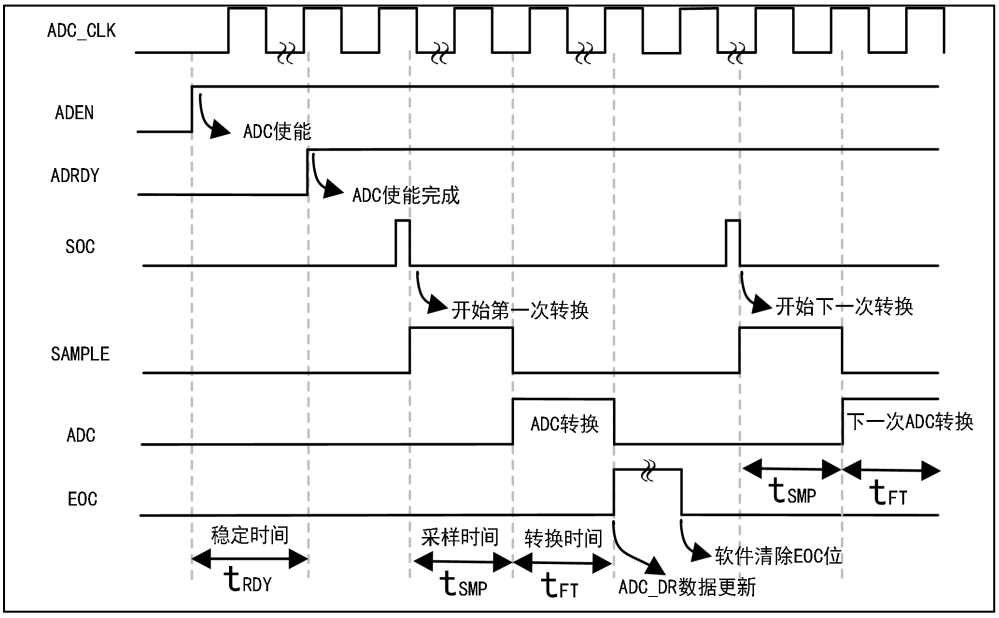
8.3.5 ADC 中断

ADC 内部的转换完成事件 EOC 被默认连接到 NVIC；转换结束后，当 NVIC 中的 ADC 中断被使能，则产生一个 ADC 中断请求到 NVIC。

8.3.6 时序图

如下图所示，ADC 在使能之后，需要一个>2uS 的上电稳定时间 t_{RDY}，在启动 ADC 转换并经过一段可编程的采样时间 t_{SAM} 后，等待 16 个 ADC_CLK 周期的转换时间 t_{FT}，最终 EOC 标志被硬件置 1，此时在 ADC_DR 寄存器中，就更新了本次转换的结果。

图 8-2 ADC 转换时序图



注意：ADC 转换时间 t_{FT} 固定为 16 个 ADC_CLK

8.3.7 数据对齐

ADC_CR 寄存器中的 ALIGN 位选择转换后数据储存的对齐方式。数据可以左对齐或右对齐，如图 8-3 和图 8-4 所示。

图 8-3 数据右对齐

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

图 8-4 数据左对齐

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0

注意：转换结果只有 12 个位有效

8.3.8 可编程的采样时间

在进行转换前，ADC 需要对输入电压进行一段时间的采样，在这段采样时间 t_{SMP} 内，输入电压越稳定，ADC 转换后的值也就越平稳；

t_{SMP} 基于 ADC_CLK，可以通过配置(ADC_SAMPLE 寄存器)的”SMP”位以修改采样时间的长度。

$t_{SMP}=t_{ADC\_CLK}*SMP$

- t_{ADC_CLK} 为 ADC 时钟的周期
- SMP 为”SMP”位的值

采样时间应配置在 3~255 之间，详见寄存器(ADC_SAMPLE)说明。

8.3.9 ADC 采样转换时间

结合 8.3.6 节的时序图和 8.3.8 可编程的采样时间的描述，可知：

ADC 采样转换时间 Tconv 为：

$Tconv = t_{SAM} + t_{FT} = t_{ADC\_CLK} * (SMP+16)$

由于 SMP 最低为 3，故 ADC 采样转换时间最快为 19 个 ADC_CLK 时钟。

8.4 寄存器描述

8.4.1 ADC 控制寄存器 ADC_CR

（地址：0x4001_2400）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	TIMS[3:0]			
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	BGS	ALIGN	MODE[1:0]		-	PSC[1:0]		SOC	-	ADEN	-	CHS[3:0]			
R/W	Res	R	RW	RW	RW	Res	RW	RW	W	Res	RW	Res	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

位 3：0	**CHS[3:0]：ADC 通道选择(ADC channel selection)** 0000： ADC_IN0 (ADC 模拟输入通道 0) 0001： ADC_IN1 (ADC 模拟输入通道 1) 0010： ADC_IN2 (ADC 模拟输入通道 2) 0011： ADC_IN3 (ADC 模拟输入通道 3) 0100： ADC_IN4 (ADC 模拟输入通道 4) 0101： ADC_IN5 (ADC 模拟输入通道 5) 0110： ADC_IN6 (ADC 模拟输入通道 6) 0111： ADC_IN7 (ADC 模拟输入通道 7) 1000： ADC_IN8 (ADC 模拟输入通道 8) 1001： ADC_IN9 (ADC 模拟输入通道 9) 1010： ADC_IN10 (ADC 模拟输入通道 10) 其他： 保留 ***注：ADC_IN8ADC 模拟输入通道 8)在芯片内部连到了模拟通道电压，模拟通道电压源由 BGS 位决定。*** ***ADC_IN9(ADC 模拟输入通道 9)在芯片内部连到了 BG1v5。*** ***ADC_IN10(ADC 模拟输入通道 10)与 VDD 相连.***
位 4	保留。必须保持为 0。
位 5	**ADEN：ADC 使能控制(ADC enable)** 该位由软件设置和清除。当该位为‘0’时，写入‘1’将把 ADC 从断电模式下唤醒。 当该位为‘1’时，写入‘1’将启动转换。应用程序需注意，在转换器上电至转换开始有一个延迟 t_{RDY}，参见图 8-2。 0： ADC 禁止 1： ADC 使能
位 6	保留。必须保持为 0。
位 7	**SOC：ADC 转换启动控制(ADC start on control)**

	该位只写， 读恒为 0， 写 0 无效， ADEN 使能后， 需要等待 RDY 状态位置 1， 才能开始启动 ADC 转换。 0： 无效操作 1： ADC 启动转换
位 9：8	PSC[1:0]：ADC 时钟频率控制(ADC prescaler) 00： 2 分频 01： 4 分频 10： 8 分频 11： 16 分频
位 10	保留。必须保持为 0。
位 12：11	MODE[1:0]：ADC 转换模式控制(ADC mode) 00： 单次转换模式 01： TIM 事件触发转换模式 10： 连续转换模式 11： 保留
位 13	ALIGN：ADC 转换结果对齐格式选择(ADC ALIGN) 0： 右对齐 1： 左对齐
位 14	BGS：模拟通道电压选择(ADC bg voltage selection) 0： BG1v2 1： BG1v0
位 15	保留。必须保持为 0。
位 19：16	TIMS[3:0]：ADC 定时触发源选择(ADC timing trigger source selection) 这些位选择用于启动 ADC 转换的外部事件。 0101： TIM1_TRGO 事件 0110： TIM1_OC1 事件 0111： TIM1_OC2 事件 1000： TIM1_OC3 事件 1001： TIM1_OC4 事件 1010： TIM2_TRGO 事件 1011： TIM3_TRGO 事件 其他： 保留
位 31：20	保留。必须保持为 0。

8.4.2 ADC 状态寄存器 ADC_SR

（地址： 0x4001_2404）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EOC	RDY
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**RDY：ADC 使能就绪标志位(ADC ready)** 0： ADC 使能未完成 1： ADC 使能完成
位 1	**EOC：ADC 转换完成标志位(ADC end of conversion)** 0： ADC 转换未完成 1： ADC 转换完成，读取 ADC_DR 寄存器以清 0 转换完成标志
位 31：2	保留。必须保持为 0。

8.4.3 ADC 结果寄存器 ADC_DR

（地址： 0x4001_2408）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	DATA[15:0]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**DATA[15:0]：ADC 转换结果(ADC data)** 左对齐： 位[15:4]为 12 位 ADC 转换结果 右对齐： 位[11:0]为 12 位 ADC 转换结果 ***注：应保证每次转换完毕，都要读取一次 ADC 转换结果，避免引用上一次转换完成未被清 0 的 EOC 标志而导致意外的结果。***
位 31：16	保留。必须保持为 0。

8.4.4 ADC 采样时间寄存器 ADC_SAMPLE

（地址： 0x4001_240C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	SMP[7:0]							
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

位 7：0	**SMP[7:0]：ADC 采样时间(ADC sample time)** LSB 为 1 个 ADC 时钟周期，ADC 时钟频率由 ADC_CR 寄存器的 PSC 设定 SMP 设定值应介于 3~255 之间，低于 3 的配置值（采样时间过短）可能造成 AD 转换结果不准确。
位 31：8	保留。必须保持为 0。

8.4.5 寄存器列表

地址	寄存器	描述	备注
0x4001_2400	ADC_CR	ADC 控制寄存器	ADC_CR 说明
0x4001_2404	ADC_SR	ADC 状态寄存器	ADC_SR 说明
0x4001_2408	ADC_DR	ADC 结果寄存器	ADC_DR 说明
0x4001_240C	ADC_SAMPLE	ADC 采样时间寄存器	ADC_SAMPLE 说明

9 高级定时器(TIM1)

9.1 综述

高级控制定时器(TIM1)由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。

它适合多种用途，包含测量输入信号的脉冲宽度或周期(输入捕获)，或者产生输出波形(输出比较、PWM、嵌入死区时间的互补 PWM 等)。

使用定时器预分频器，可以实现脉冲宽度和波形周期从微秒级到毫秒级的任意调节。

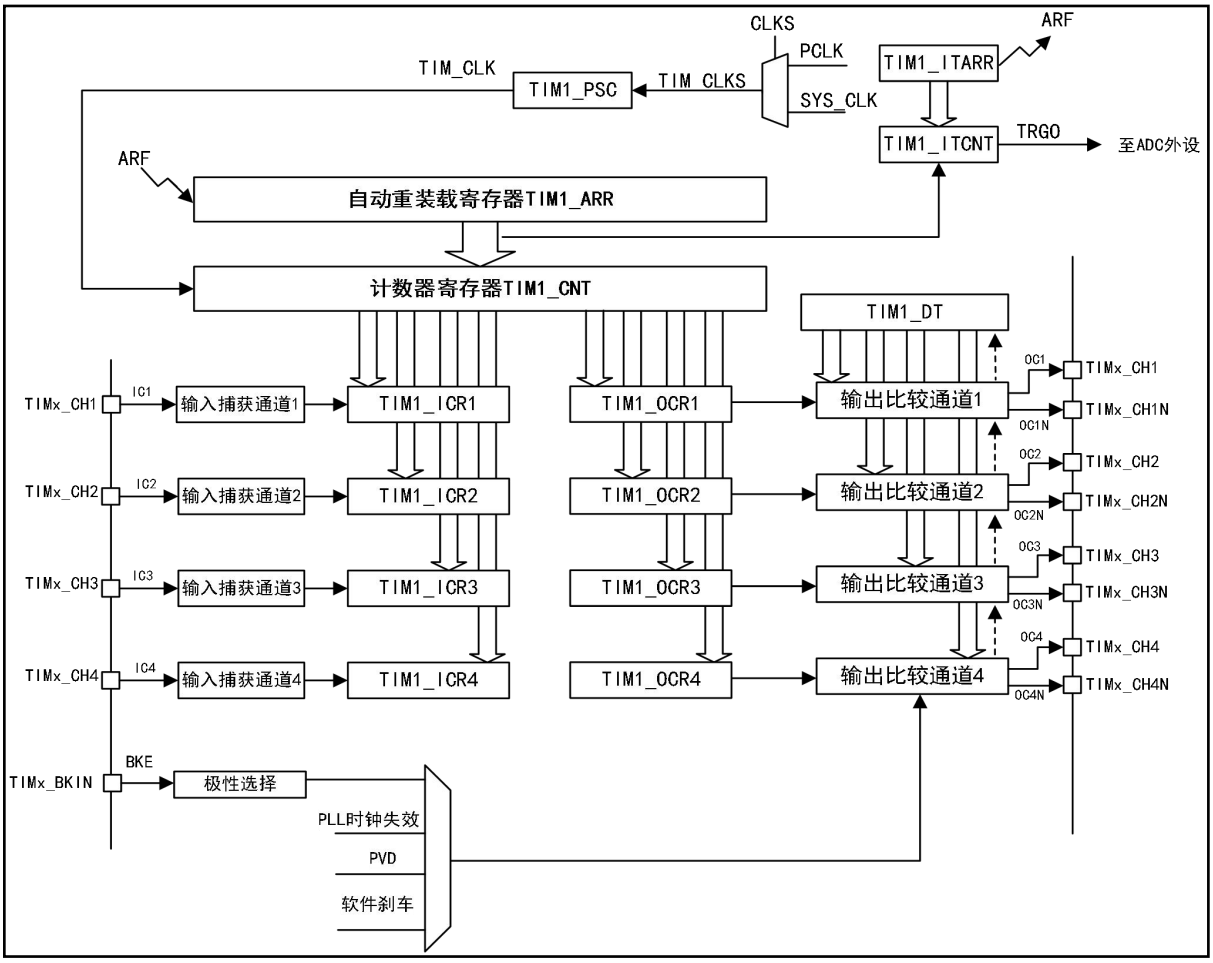
高级控制定时器(TIM1)和基本定时器(TIMx)是完全独立的，它们并不共享任何资源。

9.2 特性

- 16 位向上、向下、中央计数的自动装载计数器
- 16 位可编程预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 多达 4 个独立通道，支持：
  - 输入捕获
  - 输出比较
  - PWM 生成(向上/向下的边沿或者中间对齐模式)
  - 单脉冲模式输出
- 4 个输出比较通道分别配备独立的输出比较值寄存器
- 4 个输入捕获通道分别配备独立的输入捕获值寄存器
- 死区时间可编程的互补 PWM 输出
- 4 位可编程的中断重复计数器，支持最多计数 16 次更新事件以产生中断。
- 刹车输入信号可以将定时器置于关闭状态，或将输出信号置于空闲状态
- 如下事件发生时产生中断：
  - 计数器重载：计数器向上溢出/向下溢出
  - 各个输入捕获通道的捕获事件
  - 各个输出比较通道的比较事件
  - 刹车
- 触发 ADC 采样的同步信号
- 更新事件 UEV 由下列事件或操作产生：
  - 计数重载 AR
  - 产生更新事件 UG

9.3 TIM1 功能描述

图 9-1 高级控制定时器框图



9.3.1 时基单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动重装载寄存器。这个计数器可以向上计数、向下计数或者中央对齐(向上/向下)计数。此计数器时钟由预分频器分频得到。

计数器、自动重装载寄存器和预分频寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器 (TIMx_CNT)
- 预分频寄存器 (TIMx_PSC)
- 自动重装载寄存器 (TIMx_ARR)
- 中断重复计数值寄存器 (TIMx_ITARR)
- 中断重复计数器 (TIMx_ITCNT)

预分频寄存器和自动重装载寄存器中分别集成了一个缓冲器，未达到更新事件之前，所有对预分频寄存器和自动重装载寄存器的修改均会被存放到内部缓冲器中，直到更新事件 UEV 时生效、缓冲器中的内容被更新到预分频寄存器和自动重装载寄存器中。

当计数器达到溢出条件(由计数方向决定)时，产生计数器重载事件 AR。

计数器由预分频器的时钟输出 TIM_CLK 驱动，仅当设置了(TIMx_CR1 寄存器)中的计数器使能位(CEN)，计数器才有效。

注意：在设置了 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器才开始计数。

9.3.1.1 预分频器

高级定时器的时基单元集成了一个 16 位的预分频器，预分频器支持 1 至 65536 之间的任意数值对计数器时钟进行分频。

计数器的时钟频率 TIM_CLK 等于 TIM_CLKS/ (PSC[15:0]+1)。

TIMx_PSC 寄存器内部存在缓冲，因此可以在运行过程中改变它的数值；新的预分频数值将在下一个更新事件时生效。

以下两图是在运行过程中改变预分频系数的例子。

图 9-2 预分频系数从 1 变到 2 的计数器时序图

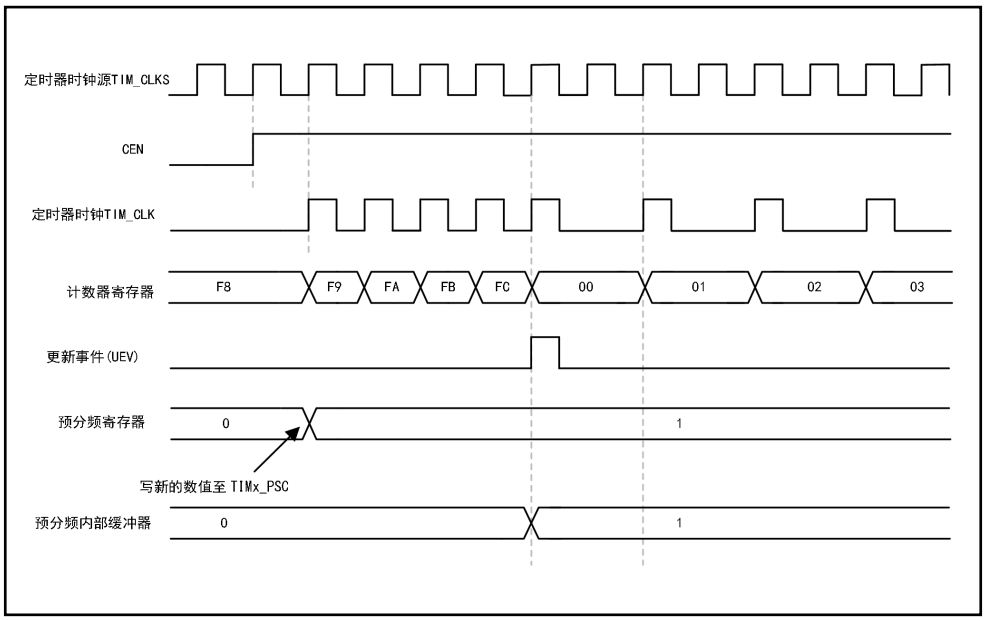
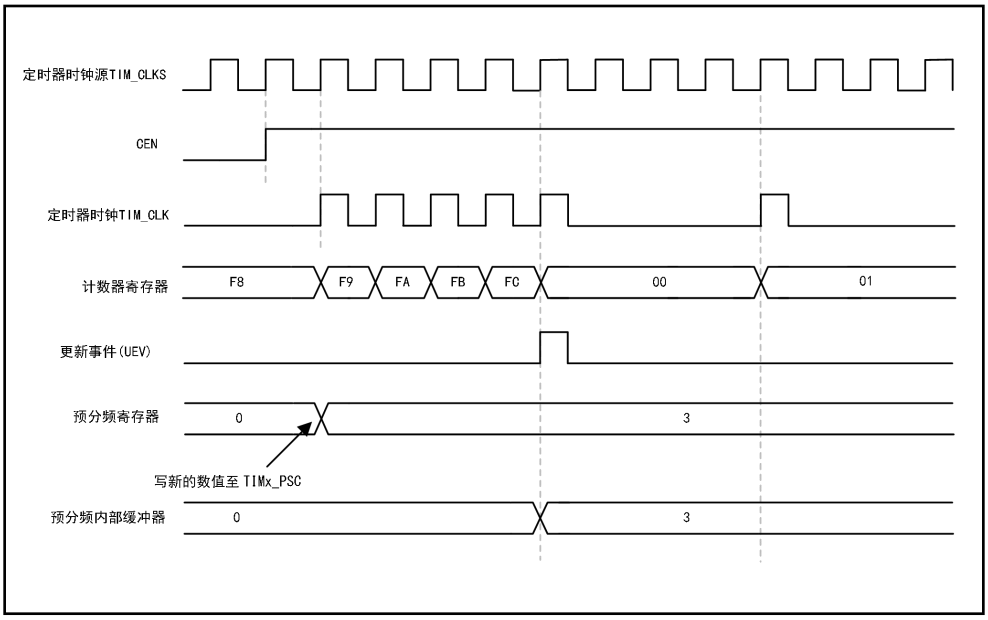


图 9-3 预分频系数从 1 变到 4 的计数器时序图



9.3.2 计数模式

注意：

1.计数模式的变更、仅在定时器关闭(CEN=0)时；

2.定时器使能之后，任何对于计数器模式的变更操作均不会被生效，软件对计数模式相关位(DIR\CMS\OPM)的操作只读。

9.3.2.1 向上计数模式

在向上计数模式中，计数器从 0 计数到自动重装载值(TIMx_ARR 寄存器中的值)，然后计数器溢出、重新从 0 开始计数并且产生一个计数重载事件 AR。

每次计数器溢出时可以产生计数重载事件，当发生一个计数重载事件时：

- 硬件将计数重载标志位(TIMx_SR 寄存器中的 ARF 位)置'1'
- 预分频器的内部缓冲器更新(TIMx_PSC 寄存器中的当前值被生效)。
- 自动重装载内部缓冲器更新(TIMx_ARR 寄存器中的当前值被生效)。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

图 9-4 计数器时序图，内部时钟分频因子为 1

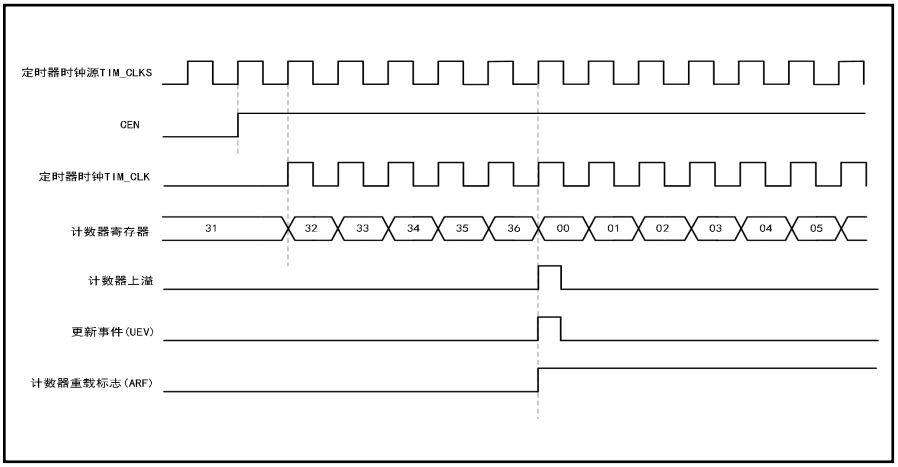


图 9-5 计数器时序图，内部时钟分频因子为 N

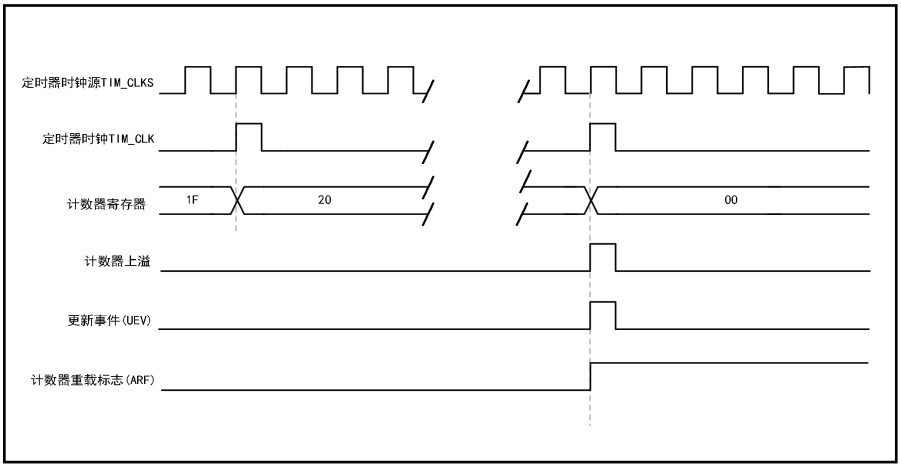
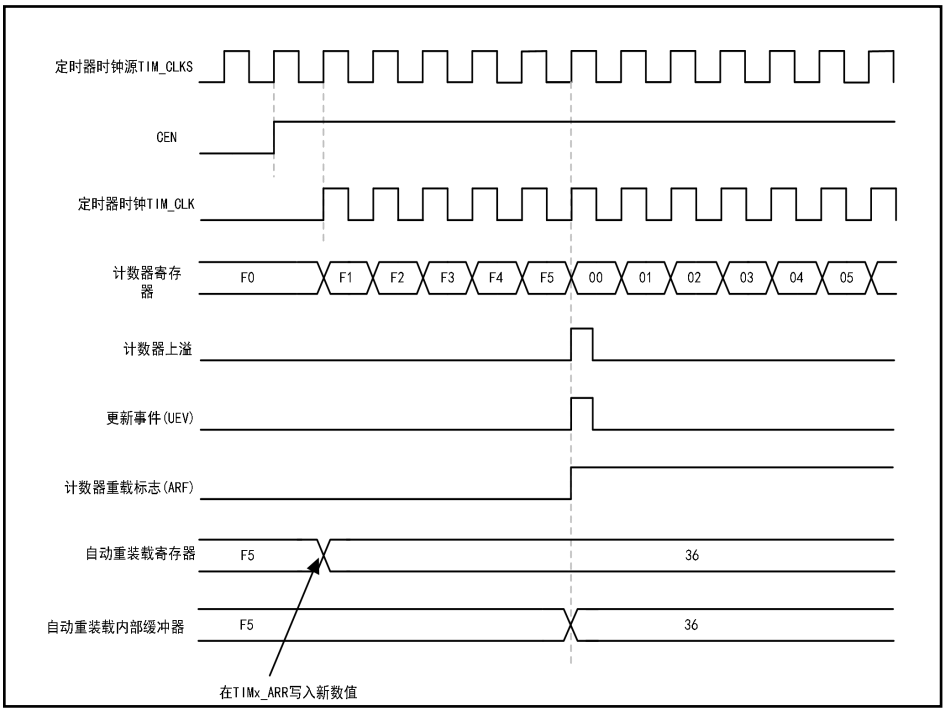


图 9-6 计数器时序图，修改自动重装载寄存器时的更新事件



9.3.2.2 向下计数模式

在向下模式中，计数器从自动装入的值(TIMx_ARR 计数器的值)开始向下计数到 0，然后计数器溢出、从自动装入的值重新开始计数、并且产生计数重载事件。

每次计数器溢出时可以产生计数重载事件，当发生一个计数重载事件时：

- 硬件将计数重载标志位(TIMx_SR 寄存器中的 ARF 位)置’1’
- 预分频器的内部缓冲器更新(TIMx_PSC 寄存器中的当前值被生效)。
- 自动重装载内部缓冲器更新(TIMx_ARR 寄存器中的当前值被生效)。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

图 9-7 计数器时序图，内部时钟分频因子为 1

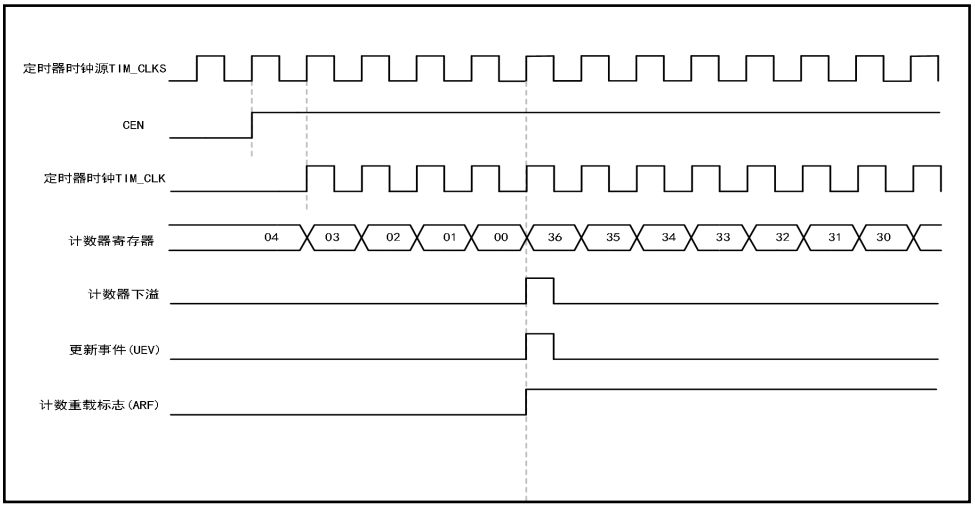


图 9-8 计数器时序图，内部时钟分频因子为 N

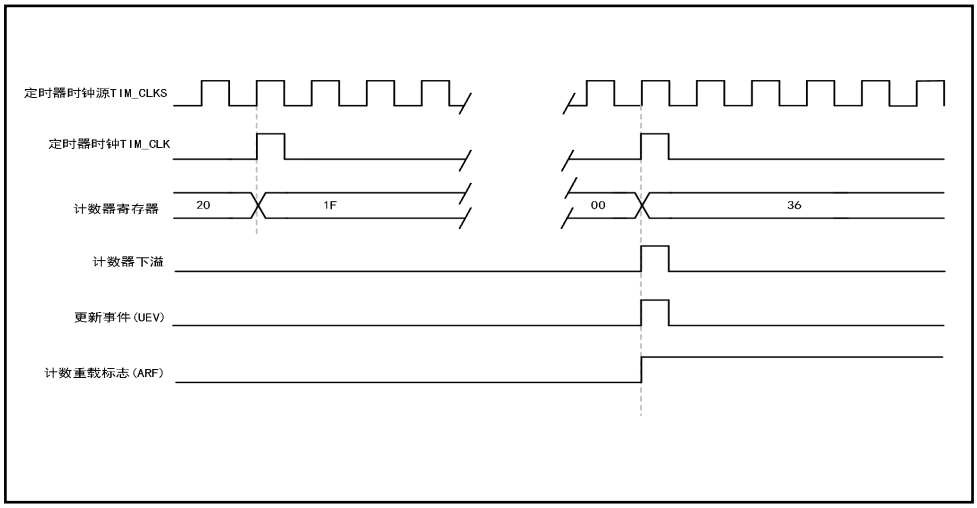
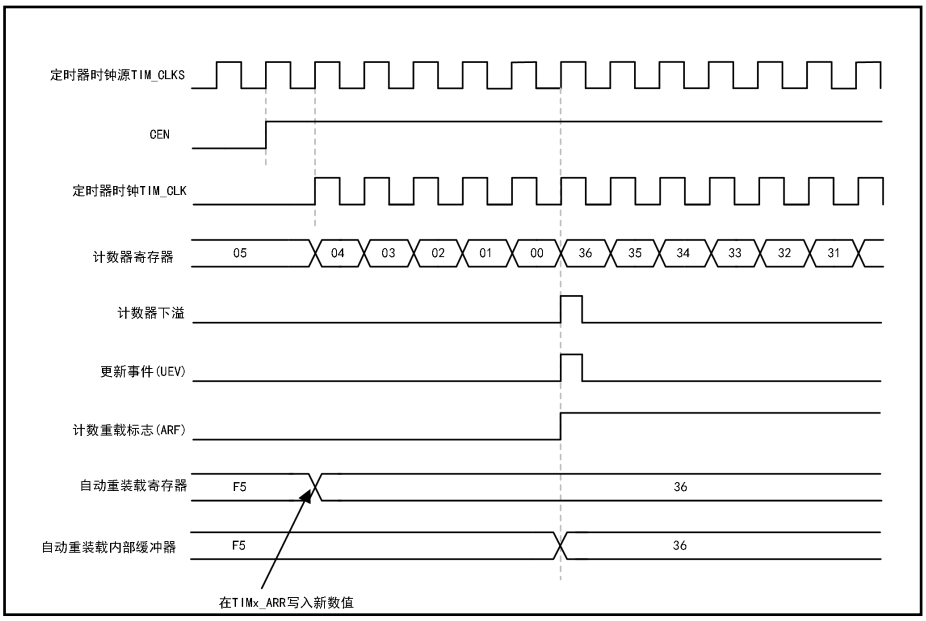


图 9-9 计数器时序图，修改自动重装载寄存器时的更新事件



9.3.2.3 中央对齐模式(向上/向下计数)

在中央对齐模式，计数器按“向上再向下”的两个步骤进行计数：

- 步骤 1：计数器从 0 计数到”自动重装载值(TIMx_ARR 寄存器中的值)-1”，然后计数器溢出、并产生一个计数重载事件 ARF；
- 步骤 2：满足步骤 1 后，计数器由自动重装载值向下计数到 1，然后计数器溢出，并产生一个计数重载事件 ARF；然后再重复步骤 1。

当发生计数重载事件时，所有的寄存器都被更新，并且计数重载标志位(TIMx_SR 寄存器中的 ARF 位)也被设置，此时：

- 预分频器内部缓冲器被加载为预装载(TIMx_PSC 寄存器)的值。
- 自动重装载内部缓冲器被加载为预装载值(TIMx_ARR 寄存器中的内容)。

注意：

1. 计数器溢出时，自动重装载寄存器的更新总是在计数器更新之前

2. 不建议运行在中央对齐模式时改写计数器

以下是一些计数器在不同时钟频率下的操作的例子：

图 9-10 计数器时序图，内部时钟分频因子为 1

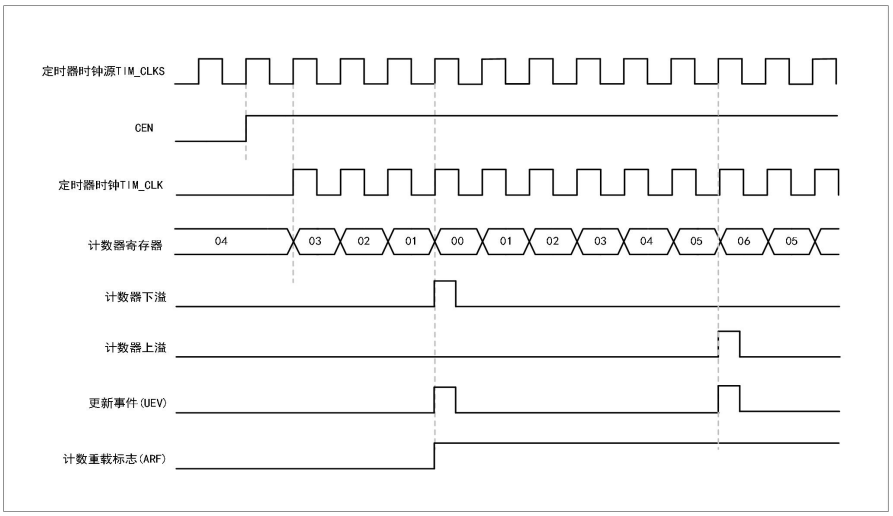


图 9-11 计数器时序图，内部时钟分频因子为 N

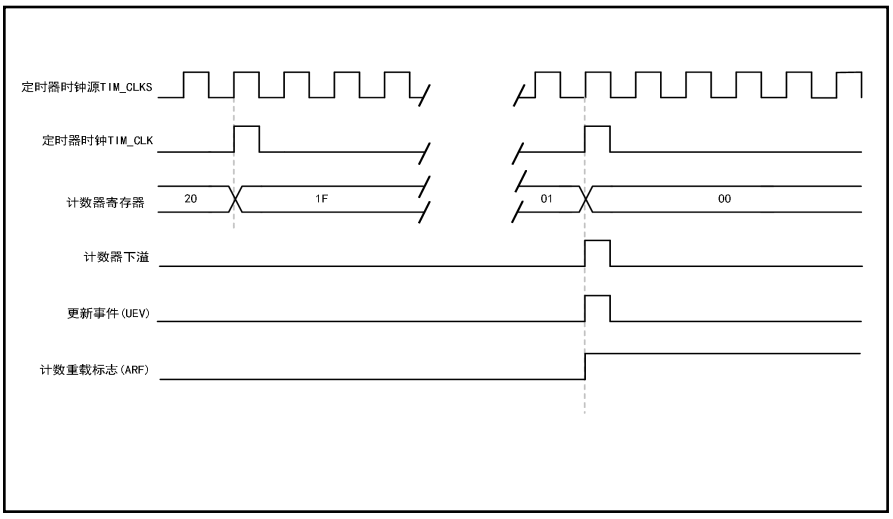
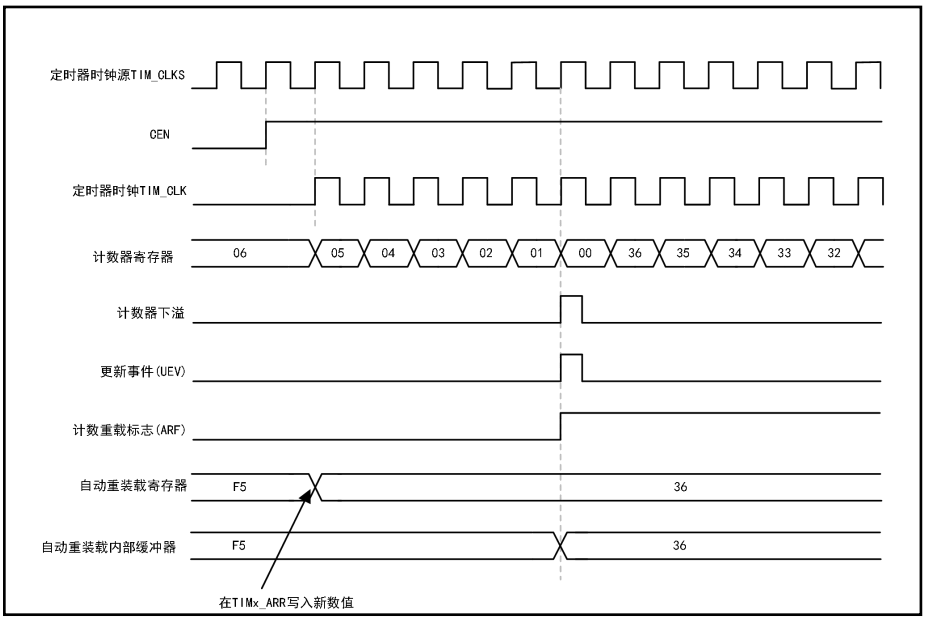


图 9-12 计数器时序图，修改自动重装载寄存器时的更新事件



9.3.2.4 单脉冲模式

相较于前述中的计数模式、单脉冲模式(OPM)是一个特例。

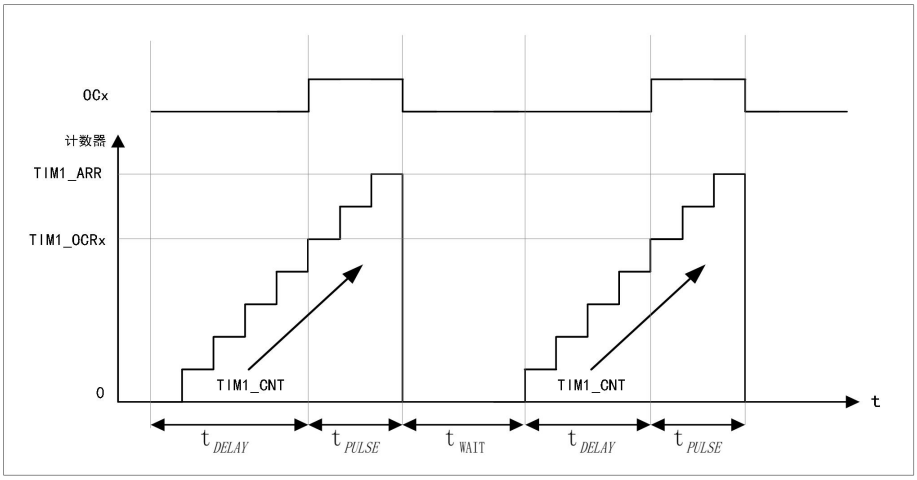
设置(TIMx_CR1 寄存器)中的 OPM 位以使能单脉冲模式、高级定时器的单脉冲模式(OPM)允许定时器产生一个脉宽可程序控制的脉冲或者一个脉冲时间。

单脉冲模式下、计数器计数到自动重装载值后溢出、并产生了计数重载事件 ARF、紧接着计时器被自动停止。

仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前(当定时器正在等待触发)，必须如下配置：

- 向上计数方式：计数器 CNT < OCRx ≤ ARR (特别地,OCRx>0)
- 向下计数方式：计数器 CNT > OCRx。

图 9-13 单脉冲模式的例子



- 通过设定合适的 TIM1_ARR 和 TIM1_OCR 以产生相应的 t_{DELAY} 和 t_{PULSE} 并在 OCx 上生成一段可控的脉冲；
- 通过自定义的 t_{WAIT}(软件或定时器延时)可产生一段连续可控的脉冲；

9.3.3 时钟源

通过(TIM1_CR1 寄存器)的 CLKS 位可以选择高级定时器的时钟源：

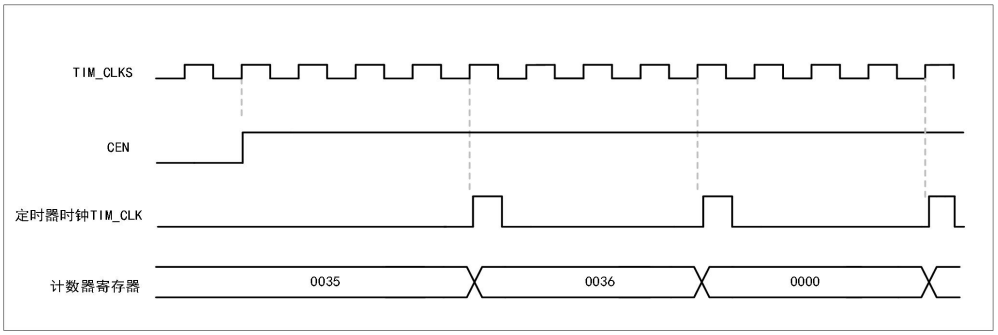
- PCLK
- HCLK

配置好高级定时器的时钟源后，由相应的时钟源经预分频器分频得到最终的高级定时器时钟(TIM_CLK)。

(TIMx_CR1)寄存器的 CEN 位是实际的控制位，只能通过软件改变它们。一旦 CEN 位置'1'，时钟源即向预分频器提供时钟。

下图示出当 TIMx_ARR=0x36，计数器在向上计数模式下，内部时钟分频因子为 4 时的操作。

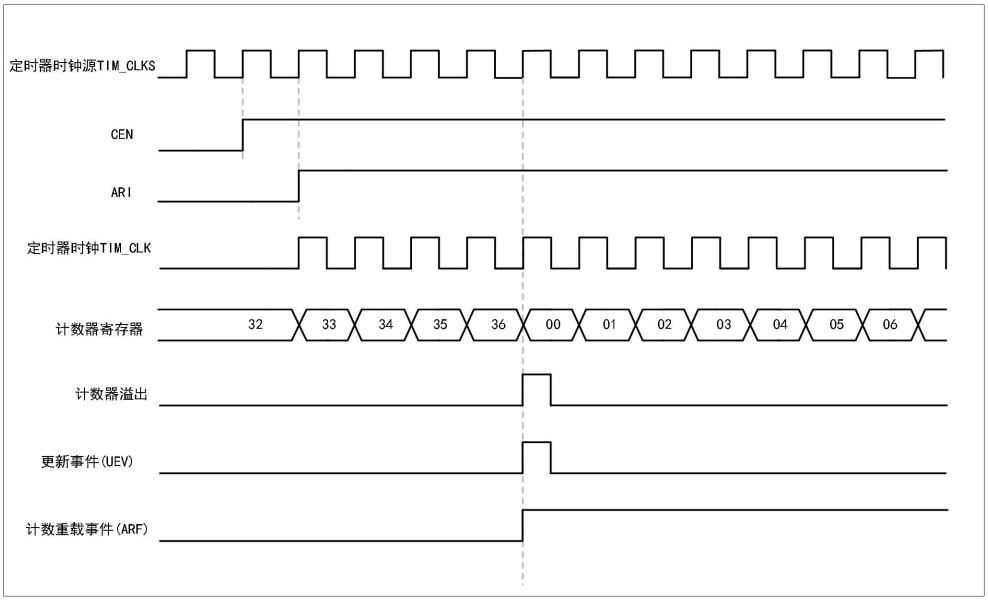
图 9-14 向上计数模式下的定时器时序图，内部时钟分频因子为 4



9.3.4 计数重载中断

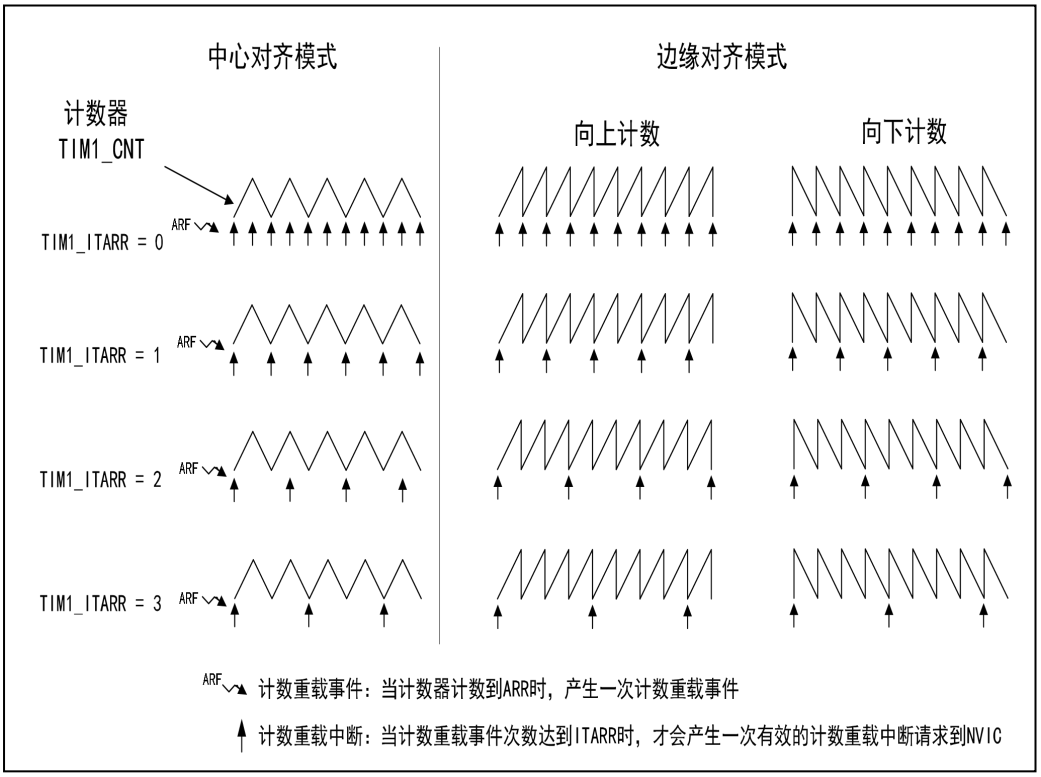
通过将 TIMx 控制寄存器 2(TIMx_CR2)中的“ARI”位置'1'、定时器在产生计数重载事件后将同步生成一个更新中断请求到 NVIC，当 NVIC 中的 TIM1 中断被使能时，系统将产生对应的中断。

图 9-15 计数重载事件与中断标志



9.3.5 中断重复计数器

9.3.1 节”时基单元”解释了计数器上溢/下溢时计数重载事件(ARF)是如何产生的、
9.3.4 节”计数重载中断”则描述了计数重载事件与计数重载中断的关系；
传统的定时器，更新事件与更新中断是同步的，但在一些应用中，要求更新中断频率的可控，此时，就可以使用中断重复计数器。
高级定时器集成了一个 4 位的中断重复计数器，支持 0 至 16 之间的任意数值对计数重载事件进行计数。
中断重复计数器 ITCNT 按照中断重复计数器值寄存器(TIM1_ITARR)中的值，对计数重载中断的次数进行计数，并将计数结果存放在中断重复计数器寄存器(TIM1_ITCNT)中，直到 ITCNT 溢出，才产生一次计数重载中断请求到 NVIC。
图 9-16 TIMx_ITARR 的寄存器设置，以及更新中断请求发出的频率



9.3.6 输入捕获功能

输入捕获功能围绕着一个输入捕获值寄存器(TIM1_ICRx)，每个输入捕获通道内部集成的输入滤波器、边沿检测器可用于检测一段脉冲、或任意电平沿的宽度。

最多支持 4 个捕获通道、每个捕获通道都支持独立的中断请求，以下两种信号类型可被电平和边沿检测器识别：

- 上升沿信号
- 下降沿信号

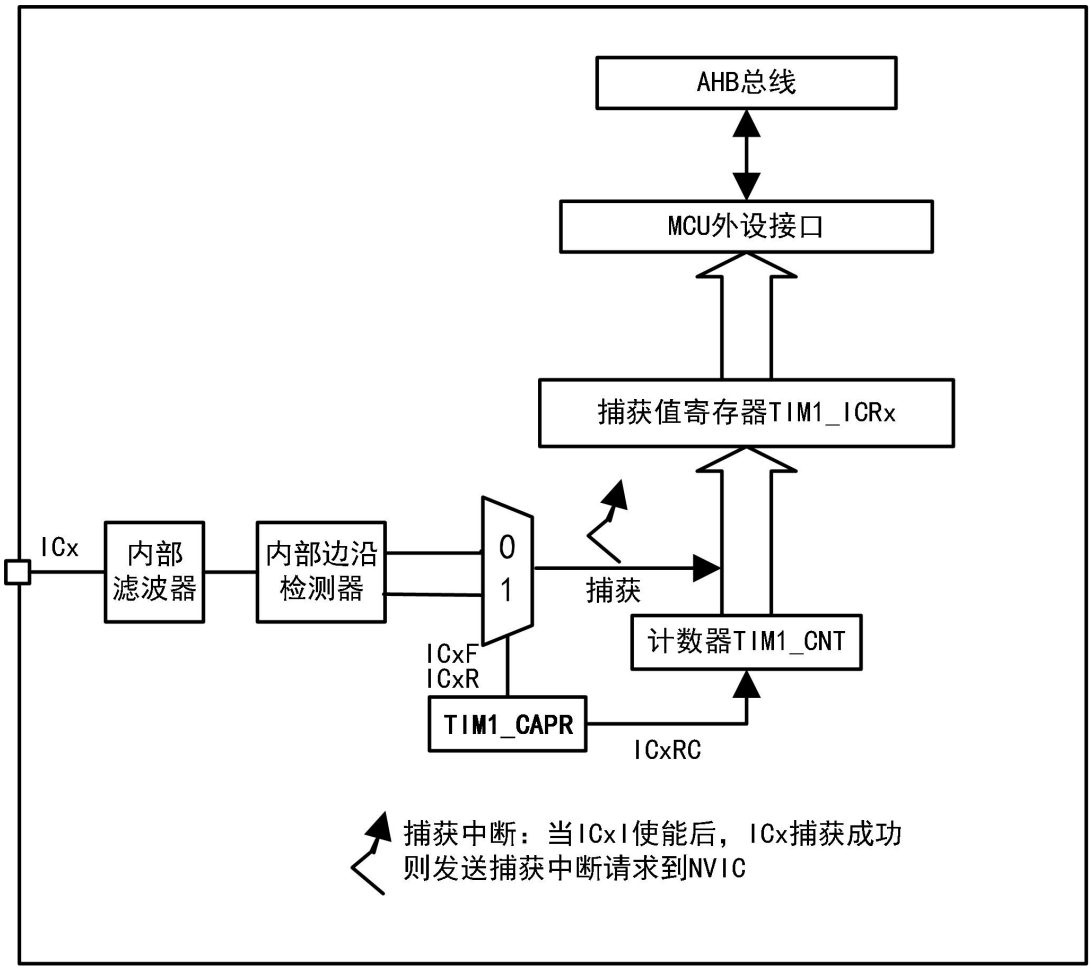
注意：一个 TIM1_CHx 复用功能引脚上，不应该同时使能输入捕获功能和输出比较功能。

9.3.6.1 输入捕获通道

输入捕获通道 ICx 在内部接到了复用功能引脚上，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到输入捕获值寄存器(TIMx_ICRx)中。

每个输入捕获通道内部都集成了一个滤波器，这个滤波器带宽大于 2 个 PCLK 时钟周期，小于 2 个 PCLK 的输入信号将被过滤。

图 9-17 输入捕获通道的框图



9.3.6.2 输入捕获模式

输入捕获模式的配置步骤：

- 在相应的复用功能 GPIO 上、复用 TIM1_CHx 功能。
- 配置高级定时器的时基单元
- 选择输入捕获信号的类型，例如：
  - 将(TIM1_CAPR)寄存器中的 ICxR 位置'1'，使能通道 x 上升沿捕获
  - 将(TIM1_CAPR)寄存器中的 ICxF 位置'1'，使能通道 x 下降沿捕获
  - 将(TIM1_CAPR)寄存器中的 IC1RC 位置'1'，使能通道 x 捕获复位计数器
- 设置 TIMx_CR1 寄存器的 CEN 位启动计数器
- 如果要产生一个输入捕获中断请求，设置(TIM1_CAPR)寄存器中的 ICxI 位。

当发生一个输入捕获时：

- 计数器的值被同步到 TIMx_ICRx 寄存器。
- (TIMx_SR)寄存器中对应的 ICxy(x 表示输入捕获通道，y 表示捕获信号沿类型)标志将被硬件置'1'
- 如设置了(TIM1_CAPR)寄存器中的 ICxI 位，则会产生一个中断请求到 NVIC。

注意：建议在读取捕获标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

9.3.7 输出比较功能

输出比较功能围绕着一个输出比较值寄存器，每个输出比较通道内部集成了输出模式控制器、刹车控制器、死区发生器、输出控制器、互补输出控制器，可以用来控制一个输出波形，或者指示一段给定的的时间已经到时。

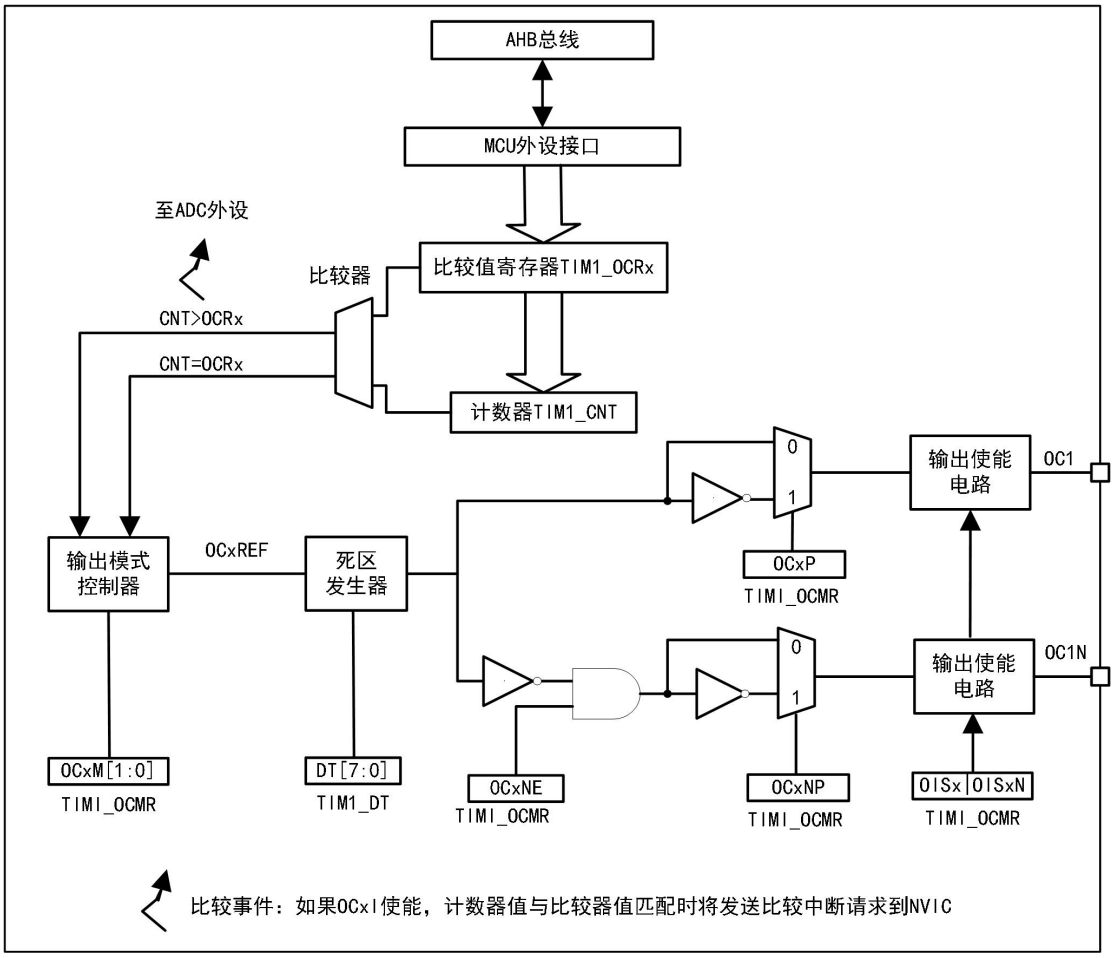
最多支持 4 个输出通道和 4 个互补输出通道，4 个输出通道支持独立的中断请求，当计数器值与比较值匹配时产生中断请求,这些中断请求在内部还被默认接入到 ADC，可用于触发 ADC 采样。

注意：一个 TIM1_CHx 复用功能引脚上，不应该同时使能输入捕获功能和输出比较功能。

9.3.7.1 输出比较通道

输出比较通道的极性控制"OCxP\OCxNP"在内部接到了复用功能引脚上，根据(TIM1_OCMR)寄存器中 OCxP 和 OCxNP 位的配置，当计数器值大于/小于输出比较值寄存器(TIM1_OCCRx)中的值时，OCxP\OCxNP 将输出不同的电平信号。

图 9-18 输出比较通道的框图



9.3.7.2 输出比较模式

输出比较模式的配置步骤：

- 在相应的复用功能 GPIO 上、复用 TIM1_CHx 功能。
- 配置高级定时器的时基单元
- 将相应的数据写入(TIM1_OCRx)寄存器中
- 通过配置(TIM1_OCMR)寄存器的 OCxM 位，以选择输出比较模式：
  - 00： 输出禁止
  - 01： PWM 模式 1
  - 10： PWM 模式 2
  - 11: 翻转电平
- 如有需要、配置(TIM1_OCMR 寄存器)的 OCxP 位和 OISx 位，以选择输出通道正常运行时的有效电平及空闲状态时的输出电平。
- 如果需要、配置(TIM1_OCMR 寄存器)的”OCxNE”位、”OCxNP”位和”OISxN”位，这些位用于使能互补通道输出、选择互补通道输出正常运行时的有效电平及空闲状态时的输出电平
- 如有需要、配置(TIM1_OCMR 寄存器)的”BKE”位以使能定时器刹车输入功能；再通过配置(TIM1_CR1 寄存器)的”BKIS”位、可选择多达 6 种的定时器刹车输入源。
- 设置 TIMx_CR1 寄存器的 CEN 位启动计数器
- 如果要产生一个比较输出中断请求，设置(TIM1_CR2)寄存器中的 OCxI 位。

当发生一个输出比较(OCx 事件)时：

- 通道 OCx 根据 OCxM、OCxP 的配置输出电平
- 如果使能了互补通道 OCxN，则 OCxN 根据 OCxM、OCxPN 的配置输出电平
- 如设置了(TIM1_CR2)寄存器中的 OCxI 位，则会产生一个 TIM1 中断请求到 NVIC。
- OCx 事件被同步到 ADC 外设

9.3.8 PWM 模式

脉冲宽度调制模式可以产生一个由(TIMx_ARR 寄存器)确定频率、由(TIMx_OCRx 寄存器)确定占空比的信号。

在(TIMx_OCMRx 寄存器)中的 OCxM 位写入二进制'01'(PWM 模式 1)或'10'(PWM 模式 2)，能够独立地设置每个 OCx 输出通道产生一路 PWM。

在 PWM 模式(模式 1 或模式 2)下， TIMx_CNT 和 TIMx_OCRx 始终在进行比较， (依据计数器的计数方向)以确定是否符合 TIMx_CNT≥TIMx_OCRx 的条件。

根据(TIMx_CR2 寄存器)中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号，在边沿对齐模式下，通过 DIR 位可以设置计数器向上或向下计数。

9.3.8.1 PWM 边沿对齐模式

- 向上计数配置

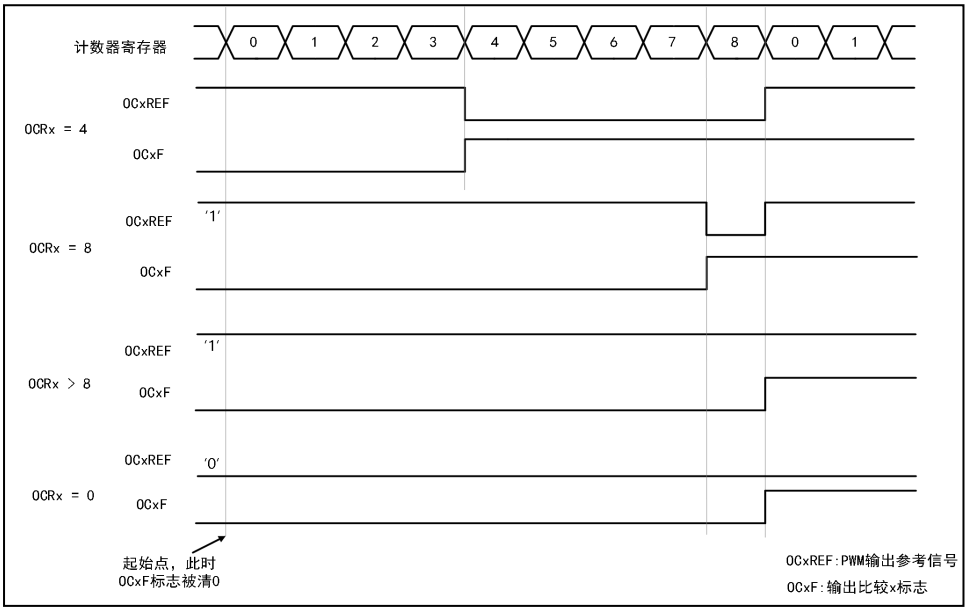
将(TIMx_CR2 寄存器)中的 DIR 位清'0'的时候执行向上计数。参考 9.3.2.1 节。

下面是当 OC1P 为'0'时，一个 PWM 模式 1 的例子：

1. 当 TIMx_CNT<TIMx_OCRx 时，PWM 输出参考信号 OCxREF 输出高电平，否则输出低电平。
2. 当 TIMx_OCRx 的值大于自动重装载值(TIMx_ARR)，则 OCxREF 保持为高电平；
3. 当 TIMx_OCRx 的值为 0，则 OCxREF 保持为低电平；

下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

图 9-19 边沿对齐的 PWM 波形(ARR=8)



- 向下计数的配置

将(TIMx_CR2)寄存器中的 DIR 位置'1'的时候执行向下计数。参考 9.3.2.2 节。

下面是当 OC1P 为'0'时，一个 PWM 模式 1 的例子：

4. 当 TIMx_CNT>TIMx_OCRx 时，OCx 输出低电平，否则输出高电平。
5. 当 TIMx_OCRx 中的值大于自动重装载值(TIMx_ARR)，则 OCx 保持为高电平；
6. 当 TIMx_OCRx 中的值为 0，则 OCx 保持为低电平；

9.3.8.2 PWM 中央对齐模式

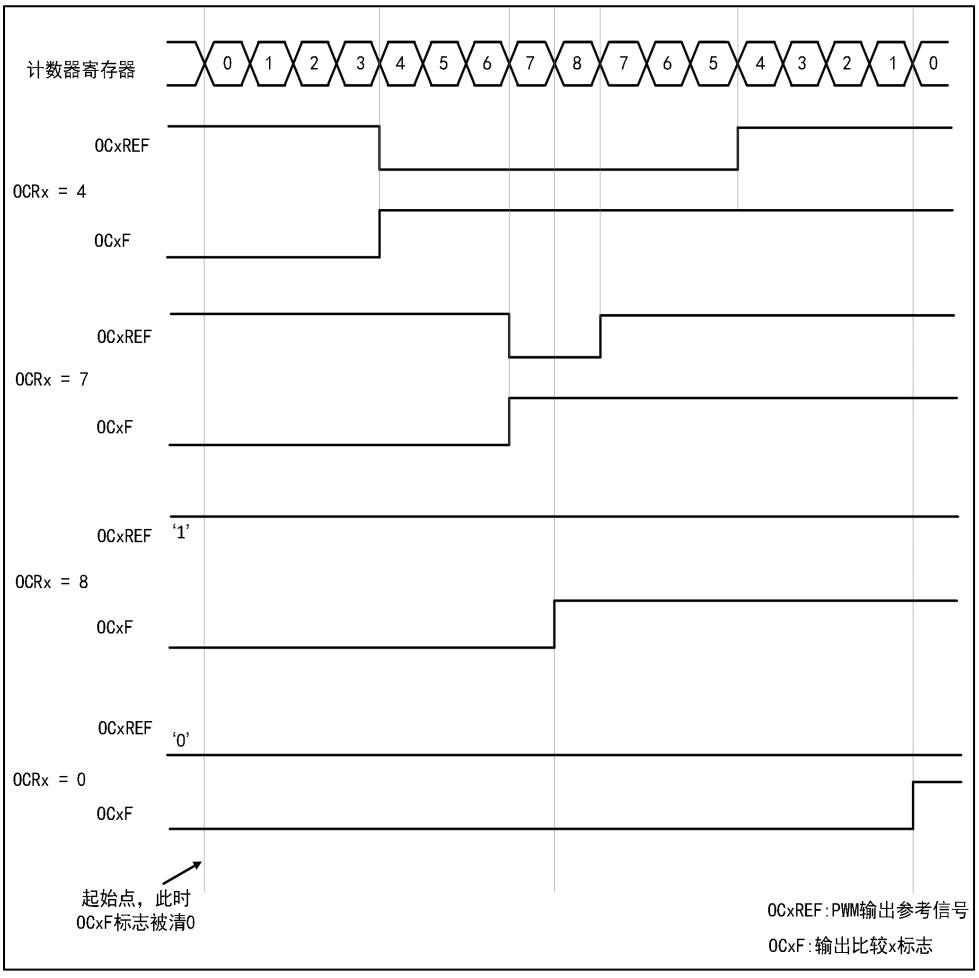
当(TIM1_CR2 寄存器)中的”CMS”位为’1’时执行中央对齐模式计数。

中央对齐模式下，输出比较标志分别在计数器向上和向下计数时被分别置’1’，一个中央计数周期内将产生两次计数重载事件，(TIM1_CR2)寄存器中的计数方向位(DIR)此时将由硬件更新，软件写无效。参照“中央对齐模式”描述。

下图给出了一些中央对齐的 PWM 波形例子：

- TIMx_ARR=8
- PWM 模式 1
- OC1P 为’1’
- CMS 为’1’

图 9-20 中央对齐的 PWM 波形



使用中央对齐模式的提示：

- 首次进入中央对齐模式时，计数器计数方向为(TIMx_CR1)寄存器中 DIR 位的值。
- (TIMx_CR1)寄存器中 DIR 位的值随中央计数方向的切换而改变，软件写无效。
- 当运行在中央对齐模式时禁止改写计数器，因为这会产生不可预知的结果：
  - 当(TIMx_CNT>TIMx_ARR)，则方向不会被更新，例如，如果计数器正在向上计数，它就会继续向上计数。
  - 将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生计数重载事件 ARF。

9.3.8.3 互补输出和死区插入

高级控制定时器(TIM1)能够输出两路互补信号；通过内部集成的死区控制器、还能够管理输出的瞬时状态，以避免推挽式连接的两个晶体管开关同时导通。
避免推挽式连接的两个晶体管开关同时导通的这段时间通常被称为死区时间，用户应该根据连接的输出器件和它们的特性(电平转换的延时、电源开关的延时等)来调整死区时间。

互补输出的配置步骤：

- 在相应的复用功能 GPIO 上、复用 TIM1_CHx 和 TIM1_CHxN 功能。
- 配置高级定时器的时基单元
- 将相应的数据写入(TIMx_OCRx)寄存器中
- 配置(TIM1_OCMR)寄存器的 OCxM 位，从下列两种输出比较模式中选择：
  - 01： PWM 模式 1
  - 10： PWM 模式 2
- 配置(TIM1_OCMR)寄存器的 OCxP 位和 OCxNP 位，选择 PWM 有效电平
- 配置(TIM1_OCMR)寄存器的 OISx 位和 OISxN 位，选择空闲状态时的输出电平
- 配置(TIM1_OCMR)寄存器的 OCxNE 位，使能互补通道输出
- 如有需要、配置(TIM1_OCMR)寄存器的 BKE 以使能定时器刹车输入功能；再通过配置(TIM1_CR1)寄存器的 BKIS[2:0]位、可选择多达 6 种的定时器刹车输入源。
- 设置 TIMx_CR1 寄存器的 CEN 位启动计数器
- 如果要产生一个输出比较中断请求，设置(TIM1_CR2)寄存器中的 OCxI 位。

死区插入的配置步骤：

- 在互补输出配置完毕的基础上，配置 TIM1 死区时间控制寄存器(TIM1_DT)，(TIM1_DT)提供了一个长度 8 位的死区时间配置位'DT'，支持数值 0~255 的范围内配置死区时间，单位为定时器时钟源 TIM_CLKS(可选的 SYS_CLK 或 PCLK)。

注意：死区时间的配置仅在互补输出时生效。如果死区时间大于当前有效电平的持续时间(OCx 或者 OCxN)，则不会产生相应的脉冲。

每一个通道的死区延时都是相同的，下面几张图显示了输出比较源信号(OCxREF)和 OCxREF 通过死区控制器输出的信号(OCx,OCxN)之间的关系。(假设 CCxP=0、CCxNP=0 并且 CCxNE=1)

图 9-21 带死区插入的互补输出

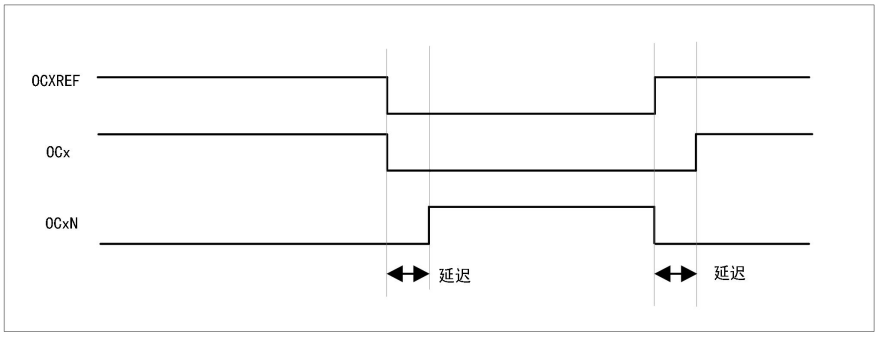


图 9-22 死区波形延迟大于负脉冲(负占空比 < 50%，DT > (ARR - OCx))

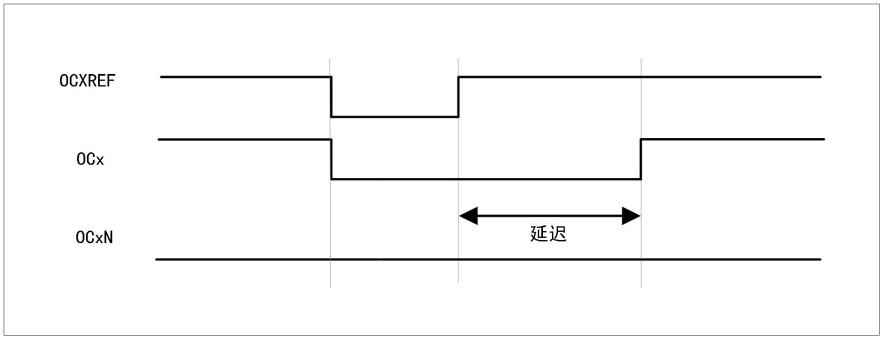


图 9-23 死区波形延迟大于正脉冲(正占空比 < 50%，DT > OCx)

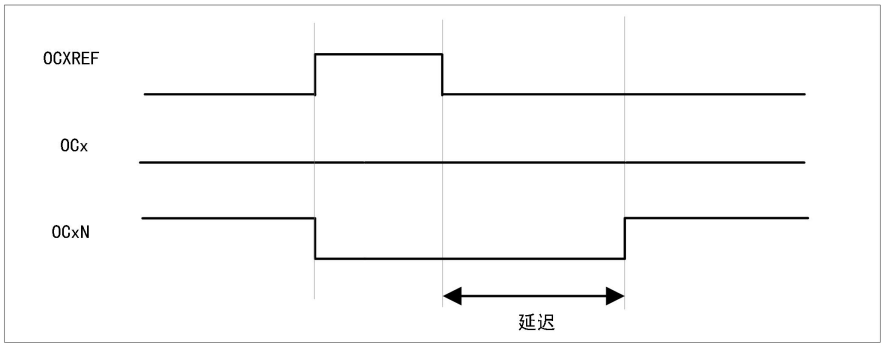
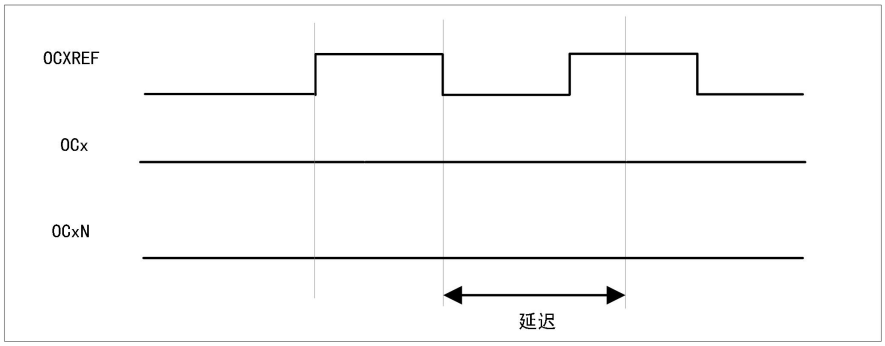


图 9-24 死区波形延迟大于正脉冲(占空比 = 50%，DT > OCx)



9.3.8.4 刹车功能

刹车功能用于保障系统可以第一时间将 PWM 关闭，并将各引脚的输出电平置于预期的空闲状态，或强制关闭定时器。

刹车功能的配置步骤：

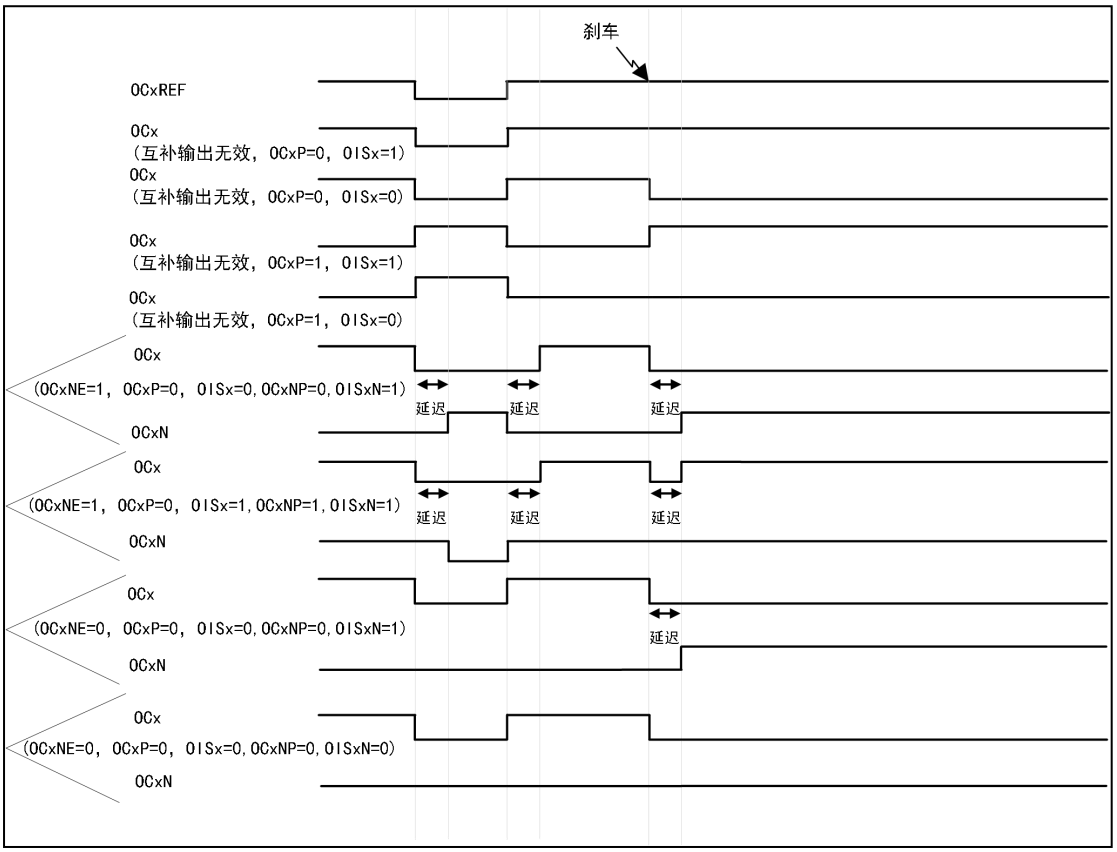
- 配置(TIM1_CR1)寄存器的 BKIC 位，选择发生刹车后，定时器的状态。
- 配置(TIM1_CR1)寄存器的 BKIS[2:0]位，以选择当前的刹车输入源
- 设置(TIM1_OCMR)寄存器的 BKP 位，以选择刹车输入信号的极性
- 设置(TIM1_OCMR)寄存器的 BKE 位，以使能刹车功能
- 如果要产生一个刹车中断请求，设置(TIM1_OCMR)寄存器中的 BKI 位

当发生一个刹车事件时：

- 通道 OCx 和 OCxN 处于空闲状态
- 通道 OCx 根据 OISx 的配置输出空闲状态电平
- 如果使能了互补通道 OCxN，则死区后、则 OCxN 根据 OISxN 的配置输出空闲状态电平
- (TIMx_SR)寄存器中对应的 BIF 标志将被硬件置'1'
- 如果使能了刹车中断 BKI，则会产生一个 TIM1 中断请求到 NVIC。

下图显示响应刹车的输出实例。

图 9-25 响应刹车的输出



9.3.9 外设间同步信号

9.3.9.1 TRGO 信号

由高级定时器产生的 TRGO 信号在内部被接到 ADC 的触发源中，TRGO 信号与计数重载事件同步，当计数重载事件发生时，也会产生一个 TRGO 信号输出到内部相应外设；

注意：选择 TRGO 信号触发外设时，TRGO 信号的频率应该在小于外设的最大工作频率。

9.3.9.2 OCx 事件

OCx 事件在内部被接到 ADC 的触发源中，OCx 事件与输出比较事件同步，当 CNT=OCRx 时，也会产生一个 OCx 事件输出到内部相应外设；

注意：选择 OCx 信号触发外设时，OCx 信号的频率应该在小于外设的最大工作频率。

9.3.10 调试模式

当微控制器进入调试模式(Cortex-M0 核心停止)时，根据(TIM1_CR1 寄存器)中 DBGE 位的设置，TIM1 计数器将继续计数或者停止计数。

注意：在使用高级定时器进行电机控制的应用时，应当谨慎处理调试模式，避免在调试模式下，功率器件短路引起的意外故障。

9.4 TIM1 寄存器描述

9.4.1 TIM1 状态寄存器(TIM1_SR)

（地址：0x4001_2C00）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	BIF	IC4F	IC4R	IC3F	IC3R	IC2F	IC2R	IC1F	IC1R	OC4F	OC3F	OC2F	OC1F	ARF
R/W	Res	Res	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1	Rw1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**ARF**：计数重载标志**(auto reload flag)** 当计数器值与重载值匹配时，该位由硬件置’1’，软件对该位置’1’将其清除 0：　　无匹配事件产生，计数器未发生过溢出。 1：　　计数器发生溢出，计数器重载事件产生。
位 1	**OC1F**：输出比较 1 标志**(output compare 1 flag)** 当计数器值与比较值匹配时该位由硬件置 1，软件可对该位置’1’将其清除。 0：　　无匹配发生； 1：　　TIMx_CNT 的值与 TIMx_OCR1 的值匹配。
位 2	**OC2F**：输出比较 2 标志**(output compare 2 flag)** 参考 OC1F 描述。
位 3	**OC3F**：输出比较 3 标志**(output compare 3 flag)** 参考 OC1F 描述。
位 4	**OC4F**：输出比较 4 标志**(output compare 4 flag)** 参考 OC1F 描述。
位 5	**IC1R**：输入捕获 1 上升沿标志**(input capture 1 rising edge flag)** 当输入捕获 1 通道捕获到上升沿信号时该位由硬件置 1，软件可对该位置’1’将其清除。 0：　　无上升沿捕获事件发生 1：　　捕获 1 通道发生上升沿捕获事件
位 6	**IC1F**：输入捕获 1 下降沿标志**(input capture 1 falling edge flag)** 当输入捕获 1 通道捕获到下降沿信号时该位由硬件置 1，软件可对该位置’1’将其清除。 0：　　无下降沿捕获事件发生 1：　　捕获 1 通道发生下降沿捕获事件
位 7	**IC2R**：输入捕获 2 上升沿标志**(input capture 2 rising edge flag)** 参考 IC1R 描述。
位 8	**IC2F**：输入捕获 2 下降沿标志**(input capture 2 falling edge flag)**

	参考 **IC1F** 描述。
位 9	**IC3R**：输入捕获 **3** 上升沿标志**(input capture 3 rising edge flag)** 参考 **IC1R** 描述。
位 10	**IC3F**：输入捕获 **3** 下降沿标志**(input capture 3 falling edge flag)** 参考 **IC1F** 描述。
位 11	**IC4R**：输入捕获 **4** 上升沿标志**(input capture 4 rising edge flag)** 参考 IC1R 描述。
位 12	**IC4F**：输入捕获 **4** 下降沿标志**(input capture 4 falling edge flag)** 参考 **IC1F** 描述。
位 13	**BIF**：刹车事件标志位**(Break input flag)** 刹车事件发生时由硬件对该位置‘1’。软件可对该位置’1’将其清除。 0：　　无刹车事件发生 1：　　发生刹车事件 ***注：刹车事件的输入源由 TIM1_CR 寄存器的 BKIS[2:0]位决定。***
位 31：14	保留。必须保持为 **0**。

9.4.2 TIM1 控制寄存器 1(TIM1_CR1)

（地址：0x4001_2C04）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	**DBGE**	**BKIC**	**BKIS[2:0]**			-	**CLKS**	**UG**	**CEN**
R/W	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	Res	RW	Rw1	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**CEN**：定时器使能控制位**(Counter enable)** 0： 关闭定时器 1： 使能定时器
位 1	**UG**：产生更新事件**(Update generation)** 0： 无动作 1： 重新初始化计数器，并产生一个更新事件。 ***注 1：预分频内部缓冲器也被清’0’(但是预分频系数不变)。*** ***注 2：中央计数模式下或 DIR=0(向上计数)时计数器被清'0'；当 DIR=1(向下计数)时，计数器取 TIMx_ARR 的值;***
位 2	**CLKS**：定时器时钟选择控制位**(TIM clock source selection)** 0： PCLK 1： SYS_CLK
位 3	保留。必须保持为 0。
位 6：4	**BKIS[2:0]**：定时器刹车输入源**(TIM Break source selection)** 000： TIMx_BKIN 001： 保留，不应使用这个配置 011： PLL 时钟失效 100： 保留，不应使用这个配置 101： 保留，不应使用这个配置 110： PVD 111： 软件刹车
位 7	**BKIC**：定时器刹车控制**(TIM break control)** 0： 刹车事件发生时，OCx/OCNx 为空闲状态，定时器不会被强制关闭,直到刹车事件失效，PWM 按照输出比较配置寄存器中的配置恢复输出。 1： 刹车事件发生时，定时器被强制关闭，EN 位为 0
位 8	**DBGE**：定时器调试挂起控制位**(TIM debug)** 0： 定时器在调试暂停时被挂起 1： 定时器在调试暂停时仍旧工作
位 31：9	保留。必须保持为 0。

9.4.3 TIM1 中断重复计数值寄存器(TIM1_ITARR)

（地址：0x4001_2C08）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	ITARR[3:0]			
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 3：0	**ITARR[3:0]**：中断重复计数值**(TIM interrupt auto reload)** 当 ITARR 值不为 0 时，中断重复计数器 ITCNT 将对更新事件 UEV 计数，直到更新事件次数达到 ITARR 值，定时器才向 NVIC 发送 UEV 中断请求。 这意味着在不同的计数模式中，NVIC 接收到来自 TIM1 的更新中断请求频率为： ● 在边沿对齐模式下，计数整周期； ● 在中心对称模式下，计数半周期；
位 31：4	保留。必须保持为 0。

9.4.4 TIM1 中断重复计数器(TIM1_ITCNT)

（地址：0x4001_2C0C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	ITCNT[3:0]			
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 3：0	**ITCNT[3:0]**：中断计数器的值**(TIM interrupt counter value)** 当软件更新 **ITARR** 时，该位会清零并重新开始对更新事件 **UEV** 进行计数，直到 **ITCNT=ITARR**，**TIM1** 向 **NVIC** 发送一次 **UEV** 中断请求，**ITCNT** 则自动从 **0** 开始重新计数。 更多细节请参考 9.3.5 节：有关 **ITCNT** 的更新和动作。
位 31：4	保留。必须保持为 0。

9.4.5 TIM1 预分频寄存器(TIM1_PSC)

（地址：0x4001_2C10）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	PSC[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**PSC[15:0]**：预分频器的值 **(Prescaler value)** **PSC** 包含了当更新事件产生时装入当前预分频器寄存器的值。
位 31：16	保留。必须保持为 **0**。

9.4.6 TIM1 计数器寄存器(TIM1_CNT)

（地址：0x4001_2C18）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CNT[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**CNT[15:0]**：计数器的值 **(Counter value)**
位 31：16	保留。必须保持为 **0**。

9.4.7 TIM1 控制寄存器 2 (TIM1_CR2)

（地址：0x4001_2C1C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	CMS	DIR	OPM	OC4I	OC3I	OC2I	OC1I	ARI
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**ARI**：计数重载中断使能**(Auto reload interrupt enable)** 当计数器值与自动重装载值匹配时产生中断请求 0：　禁止计数器重载中断 1：　使能计数器重载中断
位 1	**OC1I**：输出比较 1 中断使能**(output compare 1 interrupt enable)** 当计数器值与比较值匹配时产生中断请求 0：　不产生中断 1：　产生中断
位 2	**OC2I**：输出比较 2 中断使能**(output compare 2 interrupt enable)** 参考 OC1I 描述。
位 3	**OC3I**：输出比较 3 中断使能**(output compare 3 interrupt enable)** 参考 OC1I 描述。
位 4	**OC4I**：输出比较 4 中断使能**(output compare 4 interrupt enable)** 参考 OC1I 描述。
位 5	**OPM**：单脉冲模式 **(One-pulse mode)** 0：　在发生更新事件时，计数器不停止 1：　在发生下次更新事件时，计数器停止计数(清除 CEN 位)。
位 6	**DIR**：计数方向控制 **(Direction)** 0：　计数器向上计数 1：　计数器向下计数
位 7	**CMS**：选择中央对齐模式 **(Center-aligned mode selection)** 0：　边沿对齐模式。计数器依据方向位**(DIR)**向上或向下计数。 1：　中央对齐模式。计数器交替的向上和向下计数。
位 31：8	保留。必须保持为 0。

9.4.8 TIM1 自动重装载寄存器(TIM1_ARR)

（地址： 0x4001_2C20）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	ARR[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**ARR[15:0]**：自动重装载的值 **(Auto reload value)** 详细参考 9.3.1 节：有关 **ARR** 的更新和动作。 ***注：当自动重装载的值为 0 时，计数器不工作。***
位 31：16	保留。必须保持为 **0**。

9.4.9 TIM1 输出比较值寄存器(TIM1_OCRx)(x =1, 2, 3, 4)

（地址： OCR1：0x4001_2C24；OCR2：0x4001_2C28；
OCR3：0x4001_2C2C；OCR4：0x4001_2C30）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	OCR[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**OCR[15:0]**：输出比较通道 **x** 的值**(output compare value)** 取值范围 **0x0000~0xFFFF** **ARR** 寄存器用于设定 **PWM** 的周期，**OCRx** 寄存器用于设定四路 **PWM** 输出的占空比
位 31：16	保留。必须保持为 **0**。

9.4.10 TIM1 输入捕获配置寄存器(TIM1_CAPR)

（地址：0x4001_2C34）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IC4I	IC4RC	IC4F	IC4R	IC3I	IC3RC	IC3F	IC3R	IC2I	IC2RC	IC2F	IC2R	IC1I	IC1RC	IC1F	IC1R
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**IC1R**：通道 1 上升沿捕获使能**(input capture 1 rising capture enable)** 0：　　通道 1 上升沿捕获禁止 1：　　通道 1 上升沿捕获使能
位 1	**IC1F**：通道 1 下降沿捕获使能**(input capture 1 falling capture enable)** 0：　　通道 1 下降沿捕获禁止 1：　　通道 1 下降沿捕获使能
位 2	**IC1RC**：通道 1 捕获复位计数器**(input capture 1 capture reset conter enable)** 0：　　通道 1 捕获时不复位计数器 1：　　通道 1 捕获时复位计数器 ***注：计数器是公共的，这意味着在 A 通道使能捕获且开启复位计数器功能，将影响 B 通道的 PWM 输出。应当避免这样的应用 。***
位 3	**IC1I**：通道 1 捕获中断**(input capture 1 interrupt enable)** 0：　　通道 1 捕获中断禁止 1：　　通道 1 捕获中断使能
位 4	**IC2R**：通道 2 上升沿捕获使能**(input capture 2 rising capture enable)** 参考 IC1R 的描述。
位 5	**IC2F**：通道 2 下降沿捕获使能**(input capture 2 falling capture enable)** 参考 IC1F 的描述。
位 6	**IC2RC**：通道 2 捕获复位计数器**(input capture 2 capture reset conter enable)** 参考 IC1RC 的描述。
位 7	**IC2I**：通道 2 捕获中断**(input capture 2 interrupt enable)** 参考 IC1I 的描述。
位 8	**IC3R**：通道 3 上升沿捕获使能**(input capture 3 rising capture enable)** 参考 IC1R 的描述。

位 9	IC3F：通道 3 下降沿捕获使能(input capture 3 falling capture enable) 参考 IC1F 的描述。
位 10	IC3RC：通道 3 捕获复位计数器(input capture 3 capture reset conter enable) 参考 IC1RC 的描述。
位 11	IC3I：通道 3 捕获中断(input capture 3 interrupt enable) 参考 IC1I 的描述。
位 12	IC4R：通道 4 上升沿捕获使能(input capture 4 rising capture enable) 参考 IC1R 的描述。
位 13	IC4F：通道 4 下降沿捕获使能(input capture 4 falling capture enable) 参考 IC1F 的描述。
位 14	IC4RC：通道 4 捕获复位计数器(input capture 4 capture reset conter enable) 参考 IC1RC 的描述。
位 15	IC4I：通道 4 捕获中断(input capture 4 interrupt enable) 参考 IC1I 的描述。
位 31：16	保留。必须保持为 0。

9.4.11 TIM1 输入捕获值寄存器(TIM1_ICRx)(x = 1, 2, 3, 4)

（地址： ICR1：0x4001_2C38；ICR2：0x4001_2C3C；
ICR3：0x4001_2C40；ICR4：0x4001_2C44）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	ICR[15:0]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	ICR[15:0]：通道 x 的输入捕获值(input capture value) 当 x 通道的捕获事件发生时，计数器 CNT[15:0]中的值被同步到 ICRx[15:0]
位 31：16	保留。必须保持为 0。

9.4.12 TIM1 输出比较配置寄存器(TIM1_OCMR)

（地址： 0x4001_2C48）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	SWBK	BKI	BKP	BKE	OC4NE	OC3NE	OC2NE	OC1NE	OIS4N	OIS3N	OIS2N	OIS1N	OIS4	OIS3	OIS2	OIS1
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	OC4NP	OC4P	OC4M[1:0]		OC3NP	OC3P	OC3M[1:0]		OC2NP	OC2P	OC2M[1:0]		OC1NP	OC1P	OC1M[1:0]	
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 1：0	**OC1M[1:0]**：通道 1 输出比较模式**(output compare 1 mode)** OC1M 定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、 OC1N 的值。 OC1REF 是高电平有效，而 OC1、 OC1N 的有效电平取决于 OC1P、 OC1NP 位。 00： 输出禁止，OCI/OC1N 通道的 GPIO 处于输入状态。 01： PWM 模式 1 向上计数期间 TIMx_CNT≤TIMx_OCR1 时，OC1REF 输出有效电平，否则输出无效电平 向下计数期间 TIMx_CNT≥TIMx_OCR1 时， OC1REF 输出无效电平，否则输出有效电平 10： PWM 模式 2 向上计数期间 TIMx_CNT≤TIMx_OCR1 时，OC1REF 输出无效电平，否则输出有效电平 向下计数期间 TIMx_CNT≥TIMx_OCR1 时， OC1REF 输出有效电平，否则输出无效电平 11: 翻转电平，TIMx_CNT=TIMx_OCRx 时，OC1 输出电平翻转 ***注 1：输出禁止时，被复用的 GPIO 处于输入状态，但复用前对上/下拉电阻的配置仍生效。*** ***注 2：输出禁止后，比较值寄存器仍在与计数器进行比较，切换 OCIM 模式即可恢复 PWM 输出，若要完全禁用 OC1/OC1N 输出，可将当前复用功能清除(详见 AFIO 描述)，或者关闭 TIMR1。***
位 2	**OC1P**：输出通道 1 极性选择**(output compare 1 polarity)** **0**： **OC1** 高电平为有效电平，低电平为无效电平 **1**： **OC1** 低电平为有效电平，高电平为无效电平
位 3	**OC1NP**：输出通道 1 互补输出极性选择**(output compare 1 complementary output polarity)** **0**： **OC1N** 低电平为有效电平，高电平为无效电平 **1**： **OC1N** 高电平为有效电平，低电平为无效电平 ***注 1：互补输出时，应该始终确保互补输出极性与输出极性相反。***
位 5：4	**OC2M[1:0]**：输出通道 2 比较输出模式**(output compare 2 mode)** 参考 OC1M[1:0]的描述。
位 6	**OC2P**：输出通道 2 极性选择**(output compare 2 polarity)** 参考 OC1P 的描述。
位 7	**OC2NP**：输出通道 2 互补输出极性选择**(output compare 2 complementary output polarity)**

	参考 OC1NP 的描述。
位 9：8	OC3M[1:0]：输出通道 3 比较输出模式(output compare 3 mode) 参考 OC1M[1:0]的描述。
位 10	OC3P：输出通道 3 极性选择(output compare 3 polarity) 参考 OC1P 的描述。
位 11	OC3NP：输出通道 3 互补输出极性选择(output compare 3 complementary output polarity) 参考 OC1NP 的描述。
位 13：12	OC4M[1:0]：输出通道 4 比较输出模式(output compare 4 mode) 参考 OC1M[1:0]的描述。
位 14	OC4P：输出通道 4 极性选择(output compare 4 polarity) 参考 OC1P 的描述。
位 15	OC4NP：输出通道 4 互补输出极性选择(output compare 4 complementary output polarity) 参考 OC1NP 的描述。
位 16	OIS1：输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：　　空闲状态为低电平 1：　　空闲状态为高电平 *注 1：TIM1 使能且 OC1M 不为 0，发生刹车事件时，OC1 即为空闲状态。* *注 2：TIM1 失能且 OC1M 不为 0，OC1 即为空闲状态。*
位 17	OIS2：输出空闲状态 2(OC2 输出) (Output Idle state 2) 参考 OIS1 的描述。
位 18	OIS3：输出空闲状态 3(OC3 输出) (Output Idle state 3) 参考 OIS1 的描述。
位 19	OIS4：输出空闲状态 4(OC4 输出) (Output Idle state 4) 参考 OIS1 的描述。
位 20	OIS1N：互补输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：　　空闲状态为低电平 1：　　空闲状态为高电平 *注 1：TIM1 使能且 OC1M 不为 0 ，互补输出使能，发生刹车事件时，死区后 OC1N 为空闲状态。* *注 2：TIM1 使能且 OC1M 不为 0，互补输出失能，死区后 OC1N 为空闲状态。*
位 21	OIS2N：互补输出空闲状态 2(OC2N 输出) (Output Idle state 2) 参考 OIS1N 的描述。
位 22	OIS3N：互补输出空闲状态 3(OC3N 输出) (Output Idle state 3) 参考 OIS1N 的描述。
位 23	OIS4N：互补输出空闲状态 4(OC4N 输出) (Output Idle state 4) 参考 OIS1N 的描述。
位 24	OC1NE：比较输出 1 互补输出使能 (Compare output 1 complementary output enable) 0：　　关闭，OC1N 禁止输出。 1：　　开启，OC1N 信号输出到对应的输出引脚。 *注：OC1N 输出电平依赖于 OC1NP、OIS1N 和 OC1NE 位的值。*
位 25	OC2NE：比较输出 2 互补输出使能 (Compare output 2 complementary output enable)

	参考 OIS1N 的描述。
位 26	OC3NE：比较输出 3 互补输出使能 (Compare output 3 complementary output enable) 参考 OIS1N 的描述。
位 27	OC4NE：比较输出 4 互补输出使能 (Compare output 4 complementary output enable) 参考 OIS1N 的描述。
位 28	BKE：刹车功能使能(Break enable) 0： 刹车输入禁止 1： 刹车输入使能
位 29	BKP：刹车输入极性(Break polarity) 0： 刹车输入低电平有效 1： 刹车输入高电平有效
位 30	BKI：刹车中断(Break interrupt enable) 0： 刹车输入中断禁止 1： 刹车输入中断使能
位 31	SWBK： 软件刹车(Soft ware Break) 当定时器刹车输入源被配置为软件刹车时，将该位置’1’，即可发生刹车事件。 0： 软件刹车无效 1： 软件刹车有效

9.4.13 TIM1 死区时间控制寄存器(TIM1_DT)

（地址： 0x4001_2C4C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	DT[7:0]							
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 7：0	**DT[7:0]**：死区时间配置**(Dead-time value)** 这些位定义了插入互补输出之间的死区持续时间，单位:TIM_CLKS(可选的 SYS_CLK 或 PCLK)
位 31：8	保留。必须保持为 **0**。

9.4.14 寄存器列表

地址	寄存器	描述	备注
0x4001_2C00	TIM1_SR	TIM1 状态寄存器	TIM1_SR 说明
0x4001_2C04	TIM1_CR1	TIM1 控制寄存器 1	TIM1_CR1 说明
0x4001_2C08	TIM1_ITARR	TIM1 中断重复计数值寄存器	TIM1_ITARR 说明
0x4001_2C0C	TIM1_ITCNT	TIM1 中断重复计数器	TIM1_ITCNT 说明
0x4001_2C10	TIM1_PSC	TIM1 预分频寄存器	TIM1_PSC 说明
0x4001_2C18	TIM1_CNT	TIM1 计数器寄存器	TIM1_CNT 说明
0x4001_2C1C	TIM1_CR2	TIM1 控制寄存器 2	TIM1_CR2 说明
0x4001_2C20	TIM1_ARR	TIM1 自动重装载寄存器	TIM1_ARR 说明
0x4001_2C24	TIM1_OCR1	TIM1 输出比较值寄存器 1	TIM1_OCR1 说明
0x4001_2C28	TIM1_OCR2	TIM1 输出比较值寄存器 2	TIM1_OCR2 说明
0x4001_2C2C	TIM1_OCR3	TIM1 输出比较值寄存器 3	TIM1_OCR3 说明
0x4001_2C30	TIM1_OCR4	TIM1 输出比较值寄存器 4	TIM1_OCR4 说明
0x4001_2C34	TIM1_CAPR	TIM1 输入捕获配置寄存器	TIM1_CAPR 说明
0x4001_2C38	TIM1_ICR1	TIM1 输入捕获值寄存器 1	TIM1_ICR1 说明
0x4001_2C3C	TIM1_ICR2	TIM1 输入捕获值寄存器 2	TIM1_ICR2 说明
0x4001_2C40	TIM1_ICR3	TIM1 输入捕获值寄存器 3	TIM1_ICR3 说明
0x4001_2C44	TIM1_ICR4	TIM1 输入捕获值寄存器 4	TIM1_ICR4 说明
0x4001_2C48	TIM1_OCMR	TIM1 输出比较配置寄存器	TIM1_OCMR 说明
0x4001_2C4C	TIM1_DT	TIM1 死区时间控制寄存器	TIM1_DT 说明

10 基本定时器(TIMx)

10.1 综述

基本定时器 TIM2 和 TIM3 各包含一个 16 位自动装载计数器，由各自的可编程预分频器驱动。

它们可以作为通用定时器提供时间基准，特别地、它们的更新事件被内部连接到了 ADC，可用于触发 ADC 转换。

这 2 个定时器是互相独立的，不共享任何资源。

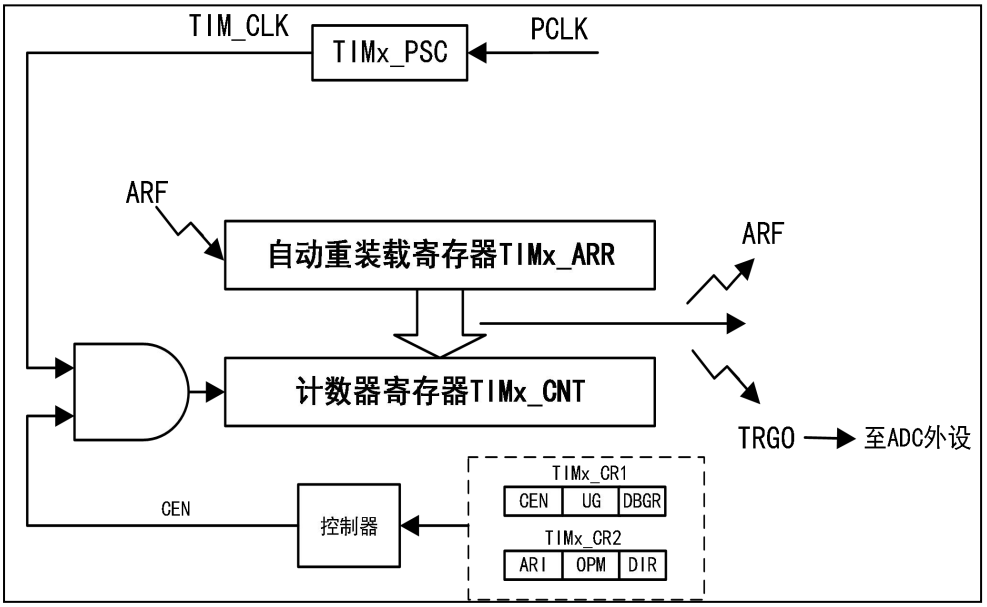
10.2 特性

TIM2 和 TIM3 定时器的主要功能包括：

- 16 位向上、向下自动重装载计数器
- 16 位可编程预分频器，支持对输入的时钟按 1～65536 之间的任意数值进行分频
- 触发 ADC 的同步电路
- 支持在计数器重载(计数器溢出)时产生中断
- 更新事件 UEV 由下列事件或操作产生：
  - 计数重载 AR
  - 产生更新事件 UG

10.3 TIMx 功能描述

图 10-1 基本定时器框图



10.3.1 时基单元

这个可编程定时器的主要部分是一个带有自动重装载功能的 16 位累加计数器，计数器的时钟通过一个预分频器得到。

软件可以读写计数器、自动重装载寄存器和预分频寄存器，这种读写操作是即时的，意味着当定时器处在运行状态时仍可以操作。

时基单元包含：

- 计数器寄存器(TIMx_CNT)
- 预分频寄存器(TIMx_PSC)
- 自动重装载寄存器(TIMx_ARR)

计数器由预分频输出 TIM_CLK 驱动，设置 TIMx_CR1 寄存器中的计数器使能位(CEN)使能计数器计数。

10.3.1.1 预分频器

基本定时器的时基单元集成了一个 16 位的预分频器，预分频器支持 1 至 65536 之间的任意数值对计数器时钟进行分频。

计数器的时钟频率 TIM_CLK 等于 PCLK/(PSC[15:0]+1)。

TIMx_PSC 寄存器内部无缓冲，因此可以在运行过程中实时的改变它的数值；新的预分频数值将立即生效。

以下两图是在运行过程中改变预分频系数的例子。

图 10-2 预分频系数从 1 变到 2 的计数器时序图

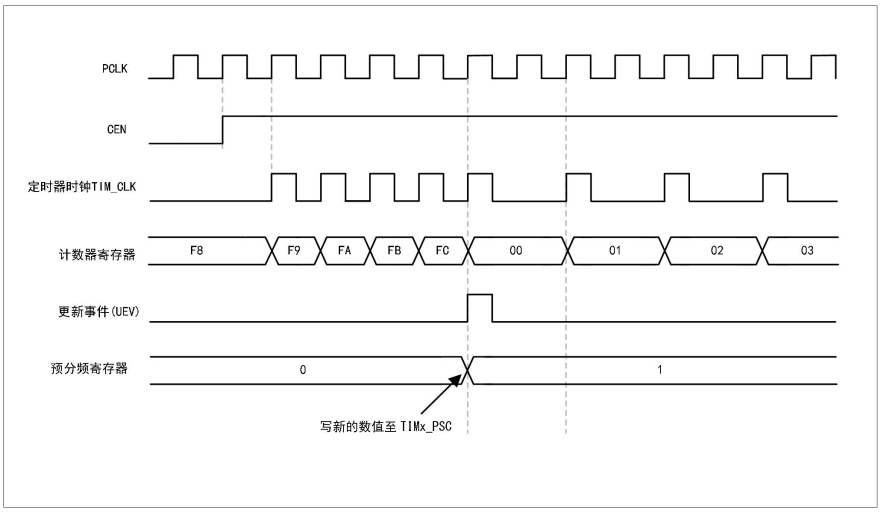
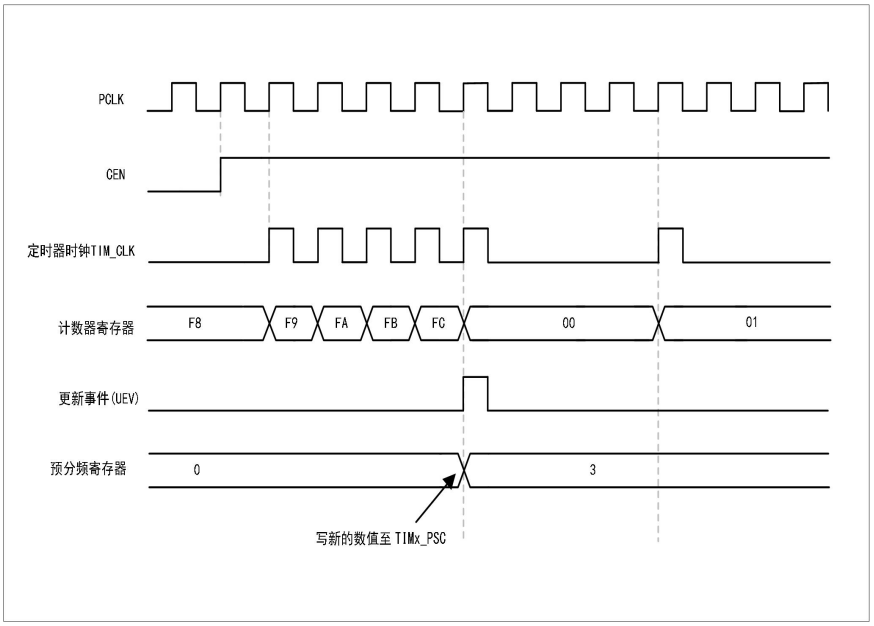


图 10-3 预分频系数从 1 变到 4 的计数器时序图



10.3.2 计数模式

在设置计数模式时，需要确保(TIMx_CR2 寄存器)的”RSTC”位始终为 1。

注意：计数模式的变更、仅在定时器关闭(CEN=0)时，定时器使能之后，任何对于计数器模式的变更操作均不会被生效，软件对计数模式相关位(DIR\OPM)的操作只读。

10.3.2.1 向上计数模式

在向上计数模式中，计数器从 0 计数到自动重装载值(TIMx_ARR 寄存器中的值)，然后重新从 0 开始计数并且产生一个计数重载事件 AR。

每次计数器溢出时可以产生计数重载事件，当发生一个计数重载事件时：

- 硬件将计数重载标志位(TIMx_SR 寄存器中的 ARF 位)置’1’
- 预装载寄存器的值更新(TIMx_ARR 寄存器中的当前值被生效)。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

图 10-4 计数器时序图，内部时钟分频因子为 1

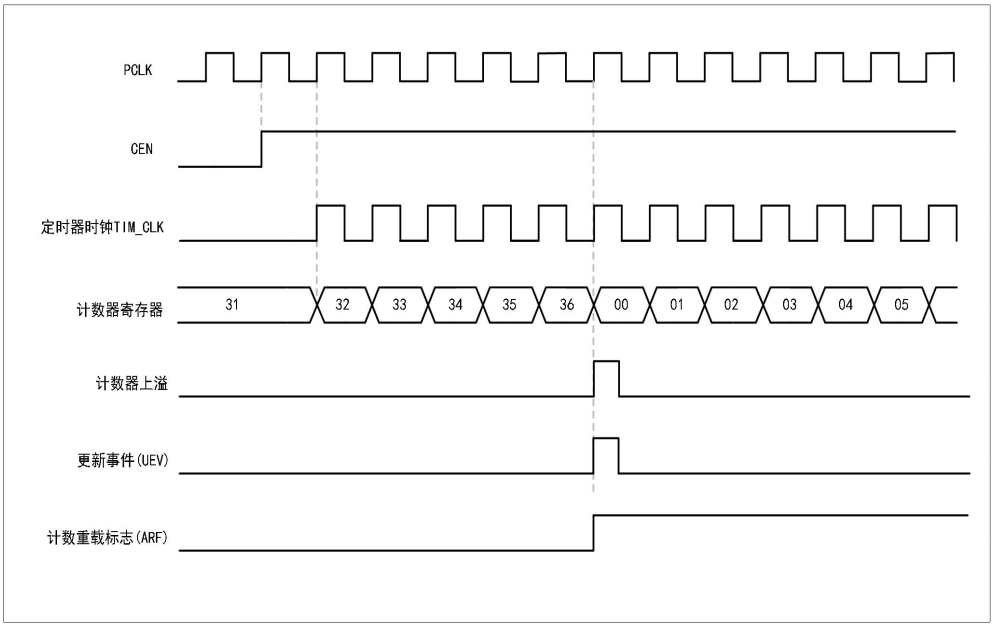


图 10-5 计数器时序图，内部时钟分频因子为 N

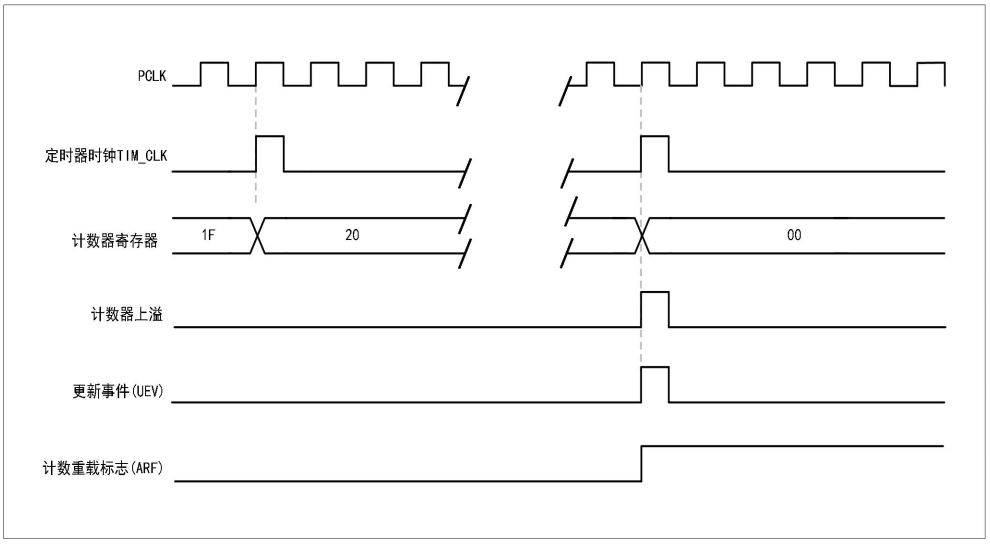
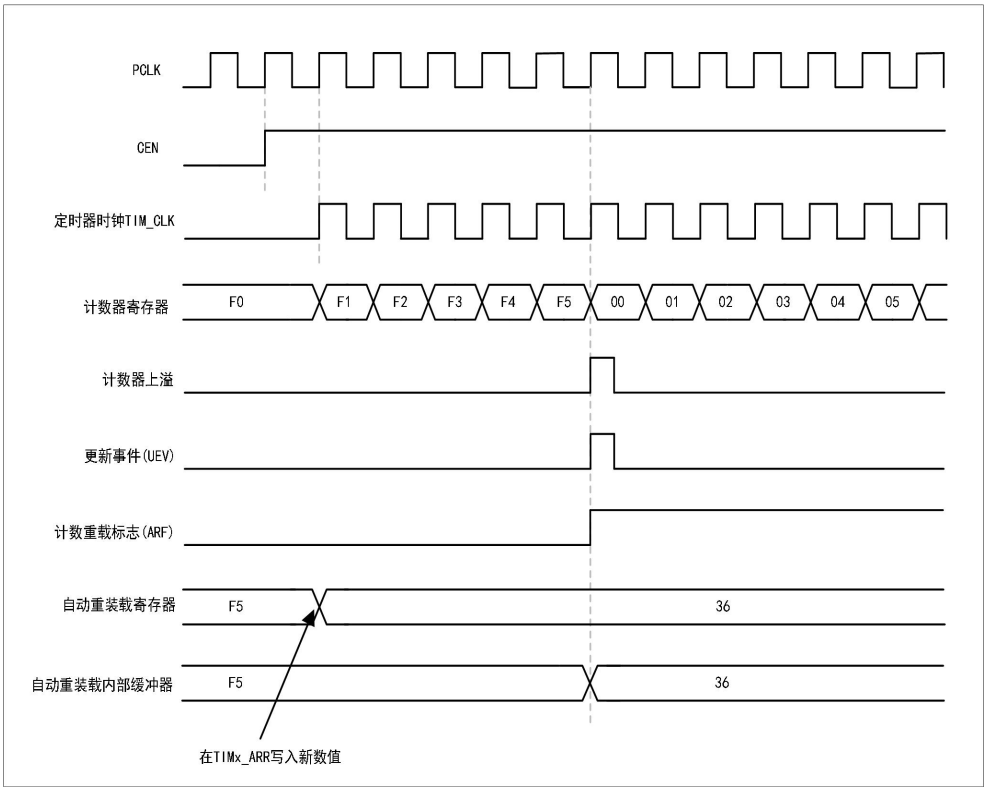


图 10-6 计数器时序图，修改自动重装载寄存器时的更新事件



10.3.2.2 向下计数模式

在向下模式中，计数器从自动装入的值(TIMx_ARR 计数器的值)开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下计数重载事件。

每次计数器溢出时可以产生计数重载事件，当发生一个计数重载事件时：

- 硬件将计数重载标志位(TIMx_SR 寄存器中的 ARF 位)置’1’
- 预装载寄存器的值更新(TIMx_ARR 寄存器中的当前值被生效)。当前的自动加载寄存器被更新为预装载值(TIMx_ARR 寄存器中的内容)。

注：自动重装载值在计数器重载入之前被更新，因此下一个周期将是预期的值。

下图给出一些例子，当 TIMx_ARR=0x36 时计数器在不同时钟频率下的动作。

图 10-7 计数器时序图，内部时钟分频因子为 1

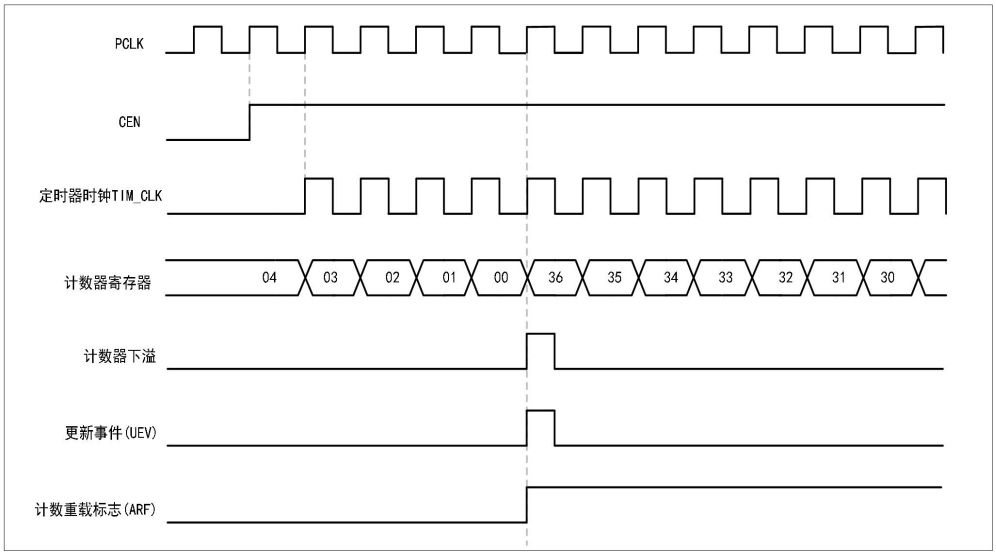


图 10-8 计数器时序图，内部时钟分频因子为 N

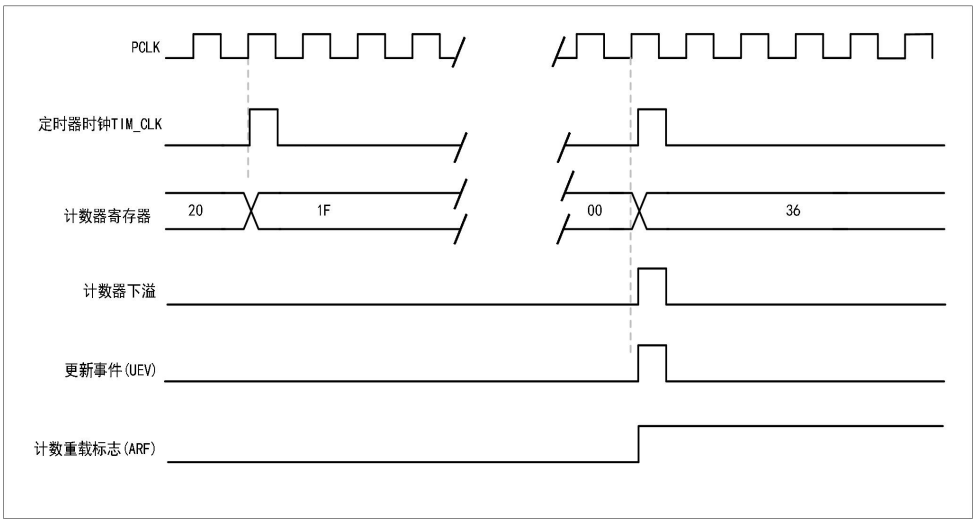
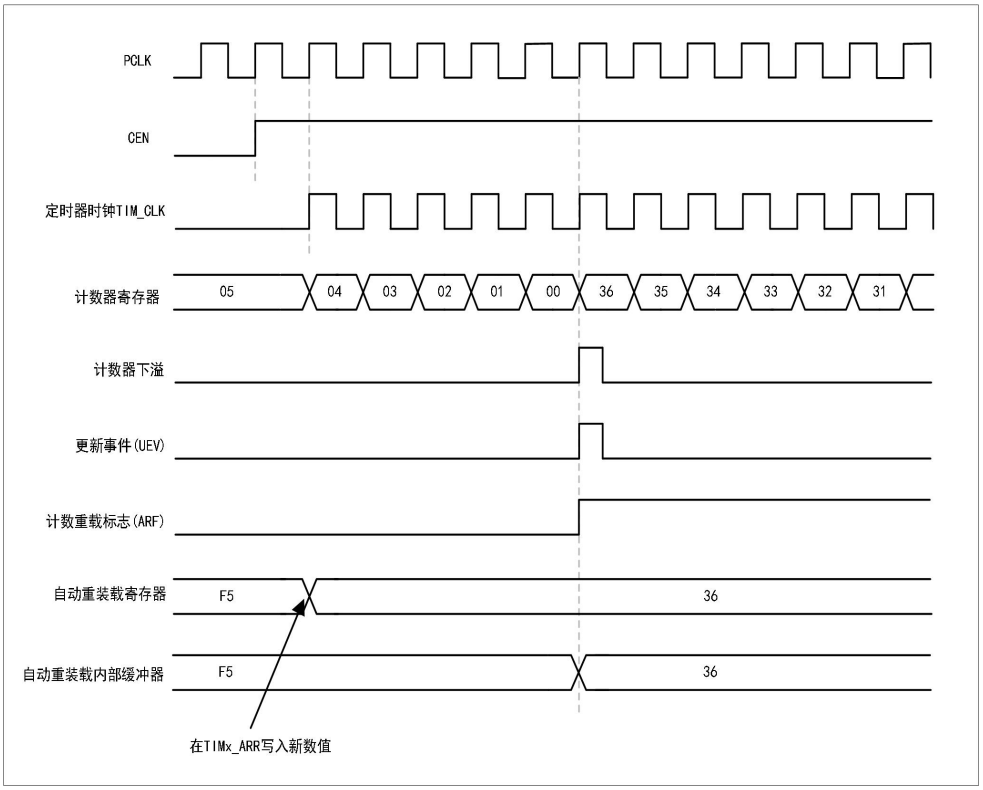


图 10-9 计数器时序图，修改自动重装载寄存器时的更新事件



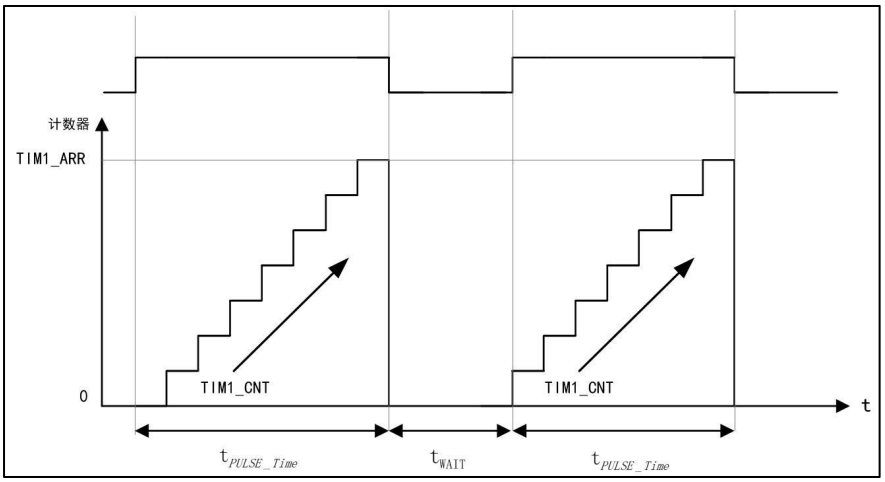
10.3.2.3 单脉冲模式

相较于前述中的计数模式、单脉冲模式(OPM)是一个特例。

设置 TIMx_CR1 寄存器中的 OPM 位以使能单脉冲模式、单脉冲模式(OPM)允许基本定时器产生一个脉冲时间，它通常可以被用于高精度的软件延时。

单脉冲模式下、计数器计数到自动重装载值后、产生了更新事件 UEV、紧接着被自动停止。

图 10-10 单脉冲模式的例子



- 通过设定合适的 TIM1_ARR 以产生相应的脉冲时间 t_{PULSE_Time}。
- 通过自定义的 t_{WAIT} 以产生一段连续可控的脉冲时间；

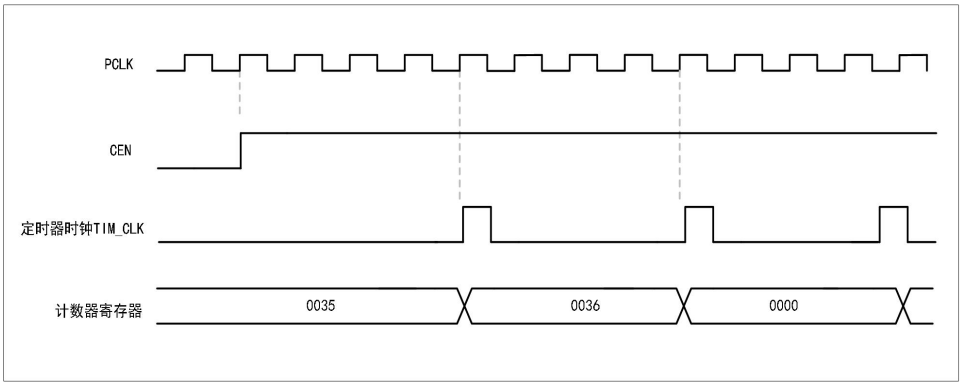
10.3.3 时钟源

定时器的时钟由 PCLK 经预分频器分频得到的(TIM_CLK)提供,。

TIMx_CR1 寄存器的 CEN 位是实际的控制位，只能通过软件改变它们。一旦 CEN 位置'1'，时钟源即向预分频器提供时钟。

下图示出控制电路和向上计数器在普通模式下，预分频值为 1 时的操作。

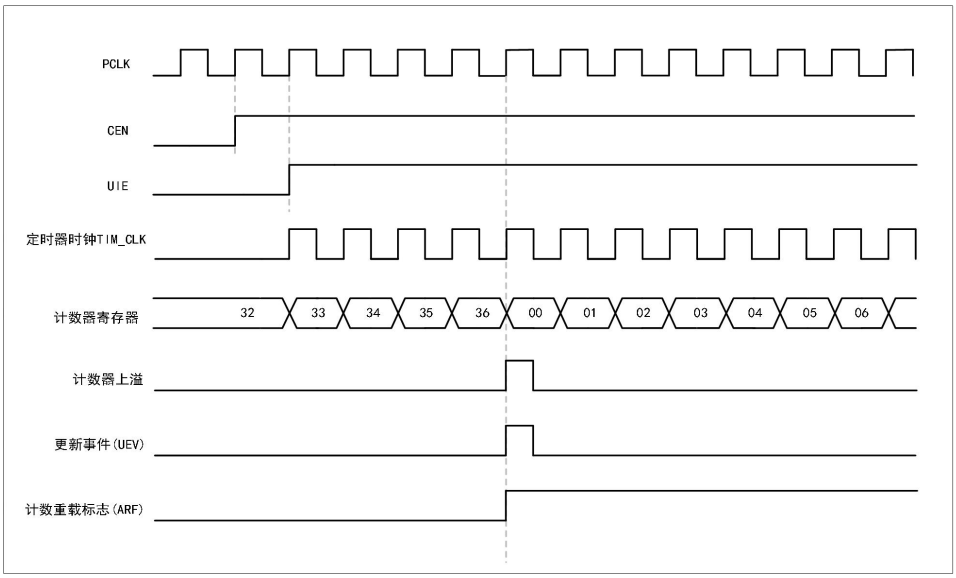
图 10-11 普通模式下的定时器时序图，内部时钟分频系数为 4



10.3.4 计数重载中断

通过将 TIMx 控制寄存器 2(TIMx_CR2)中的“ARI”位置'1'、定时器在产生更新事件后将同步生成一个更新中断请求到 NVIC，当 NVIC 中的 TIM2 或 TIM3 中断被使能时，系统将产生对应的中断。

图 10-12 更新事件与中断标志



10.3.5 同步信号 TRGO

由基本定时器产生的 TRGO 信号在内部被接到 ADC 的触发源中，TRGO 信号与更新事件同步，当更新事件发生时，也会产生一个 TRGO 信号输出到内部相应外设；

注意：选择 TRGO 信号触发外设时，TRGO 信号的频率应该在小于外设的最大工作频率。

10.3.6 调试模式

当微控制器进入调试模式(Cortex-M0 核心停止)时，根据(TIMx_CR1 寄存器)中 DBGE 位的设置，TIMx 计数器将继续计数或者停止计数。

注意：在调用到 TIMx 控制外部功率器件时，应当谨慎处理调试模式，避免在调试模式下，功率器件短路引起的意外故障。

10.4 TIMx 寄存器描述

10.4.1 TIMx 状态寄存器(TIMx_SR)(x = 2, 3)

（地址： TIM2：0x4000_1000；TIM3：0x4000_1400）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ARF
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**ARF：** 计数重载标志**(auto reload flag)** 当计数器值与重载值匹配时， 该位由硬件置‘1’，软件对该位写“1”以将标记清除 0： 无匹配事件产生， 计数器未发生过溢出。 1： 计数器发生溢出， 计数器重载事件产生。
位 31：1	保留。必须保持为 0。

10.4.2 TIMx 控制寄存器 1(TIMx_CR1)（x = 2, 3）

（地址： TIM2：0x4000_1004；TIM3：0x4000_1404）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	DBGE	-	-	-	-	-	-	UG	CEN
R/W	Res	Res	Res	Res	Res	Res	Res	RW	Res	Res	Res	Res	Res	Res	Rw1	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**CEN：** 使能定时器**(Counter enable)** 0： 关闭定时器 1： 使能定时器
位 1	**UG：** 产生更新事件 **(Update generation)** 0： 无动作 1： 重新初始化计数器， 并产生一个更新事件。 ***注 1： DIR=0(向上计数)时计数器被清’0’； 当 DIR=1(向下计数)时， 计数器取 TIMx_ARR 的值;***
位 7：2	保留。必须保持为 0。
位 8	**DBGE：** 计数器调试挂起控制位**(TIM debug)** 0： 定时器在调试暂停时被挂起，计数器停止计数 1： 定时器在调试暂停时继续工作，计数器仍旧计数。
位 31：9	保留。必须保持为 0。

10.4.3 TIMx 预分频寄存器(TIMx_PSC)（x = 2, 3）

（地址： TIM2：0x4000_1008；TIM3：0x4000_1408）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	PSC[15:0]															
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**PSC[15:0]**：预分频器的值**(Prescaler value)** PSC 包含了当更新事件产生时装入当前预分频器寄存器的值。
位 31：8	保留。必须保持为 0。

10.4.4 TIMx 计数器寄存器(TIMx_CNT)（x= 2, 3）

（地址： TIM2：0x4000_1010；TIM3：0x4000_1410；）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CNT[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**CNT[15:0]**：计数器的值 **(Counter value)**
位 31：16	保留。必须保持为 0。

10.4.5 TIMx 控制寄存器 2(TIMx_CR2)（x= 2, 3）

（地址： TIM2：0x4000_1014；TIM3：0x4000_1414）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	DIR	OPM	RSTC	ARI
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**ARI**：计数重载中断使能**(Auto reload interrupt enable)** 当计数器值与自动重装载值匹配时产生中断请求 0： 禁止计数器重载中断 1： 使能计数器重载中断
位 1	**RSTC**：复位计数器**(Reset counter)** 当计数器值与自动重装载值匹配时复位计数器 0： 计数重载时，不复位计数器 1： 计数重载时，复位计数器 ***注：当该位为 0 时：*** ***在向上计数：计数器将一直计数到 0XFFFF，接着计数器+1 溢出，才会将计数器值置为'0'。*** ***在向下计数：计数器从自动重装载值一直计数到 0，接着计数器-1 溢出，由"0xFFFF"开始重新向下计数。***
位 2	**OPM**：单脉冲模式 **(One-pulse mode)** 0： 在发生更新事件时，计数器不停止 1： 在发生下次更新事件时，计数器停止计数(清除 CEN 位)。
位 3	**DIR**：计数方向控制 **(Direction)** 0： 计数器向上计数 1： 计数器向下计数
位 31：4	保留。必须保持为 0。

10.4.6 TIMx 自动重装载寄存器(TIMx_ARR)（x = 2, 3）

（地址： TIM2：0x4000_1018；TIM3：0x4000_1418；）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	ARR[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**ARR[15:0]**：自动重装载的值 **(Auto reload value)** 详细参考 10.3.1 节：有关 ARR 的更新和动作。 ***注：当自动重装载的值为 0 时，计数器不工作。***
位 31：16	保留。必须保持为 **0**。

10.4.7 寄存器列表

地址	寄存器	描述	备注
0x4000_1000	TIM2_SR	TIM2 状态寄存器	TIM2_SR 说明
0x4000_1004	TIM2_CR1	TIM2 控制寄存器	TIM2_CR1 说明
0x4000_1008	TIM2_PSC	TIM2 预分频寄存器	TIM2_PSC 说明
0x4000_1010	TIM2_CNT	TIM2 计数器寄存器	TIM2_CNT 说明
0x4000_1014	TIM2_CR2	TIM2 控制寄存器 2	TIM2_CR2 说明
0x4000_1018	TIM2_ARR	TIM2 自动重装载寄存器	TIM2_ARR 说明
0x4000_1400	TIM3_SR	TIM3 状态寄存器	TIM3_SR 说明
0x4000_1404	TIM3_CR1	TIM3 控制寄存器	TIM3_CR1 说明
0x4000_1408	TIM3_PSC	TIM3 预分频寄存器	TIM3_PSC 说明
0x4000_1410	TIM3_CNT	TIM3 计数器寄存器	TIM3_CNT 说明
0x4000_1414	TIM3_CR2	TIM3 控制寄存器 2	TIM3_CR2 说明
0x4000_1418	TIM3_ARR	TIM3 自动重装载寄存器	TIM3_ARR 说明

11 低功耗定时器(TIM4)

11.1 综述

低功耗定时器 TIM4 工作在 LSI 时钟下，它包含一个 16 位自动装载计数器，由各自的可编程预分频器驱动。

在运行模式下，TIM4 可作为通用定时器提供秒级的时间基准；

在低功耗模式下(睡眠状态或深度睡眠状态)，TIM4 仍可正常工作，提供精确的时间以控制睡眠唤醒。

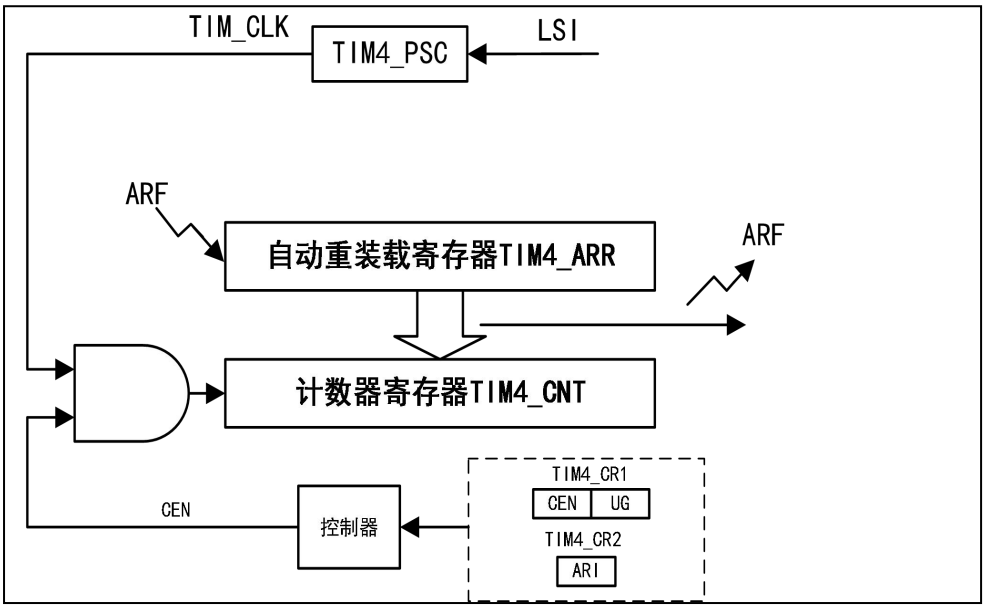
11.2 特性

TIM4 定时器的主要特性包括：

- 16 位向上自动重装载计数器
- 16 位可编程预分频器，支持对输入的时钟按 1～65536 之间的任意数值进行分频
- 在 LSI 时钟下工作
- 支持在更新事件(计数器溢出)时产生中断
- 在低功耗模式下工作(睡眠状态或深度睡眠状态)
- 更新事件 UEV 由下列事件或操作产生：
  - 计数重载 AR
  - 产生更新事件 UG

11.3 TIM4 功能描述

图 11-1 低功耗定时器框图



11.3.1 时基单元

这个低功耗定时器的主要部分是一个带有自动重装载功能的 16 位向上计数器，计数器的时钟通过一个预分频器得到。

软件可以读写计数器、自动重装载寄存器和预分频寄存器，这种读写操作是即时的，意味着当定时器处在运行状态时仍可以操作。

时基单元包含：

- 计数器寄存器(TIM4_CNT)
- 预分频寄存器(TIM4_PSC)
- 自动重装载寄存器(TIM4_ARR)

计数器由预分频输出 TIM_CLK 驱动，设置 TIM4_CR1 寄存器中的计数器使能位(CEN)使能计数器计数。

11.3.1.1 预分频器

低功耗定时器的时基单元集成了一个 16 位的预分频器，预分频器支持 1 至 65536 之间的任意数值对计数器时钟进行分频。

计数器的时钟频率 TIM_CLK 等于 LSI/(PSC[15:0]+1)。

TIM4_PSC 寄存器内部无缓冲，因此可以在运行过程中实时的改变它的数值；新的预分频数值将立即生效。

以下两图是在运行过程中改变预分频系数的例子。

图 11-2　TIMx_ARR=0xFF 时，预分频系数从 1 变到 2 的计数器时序图

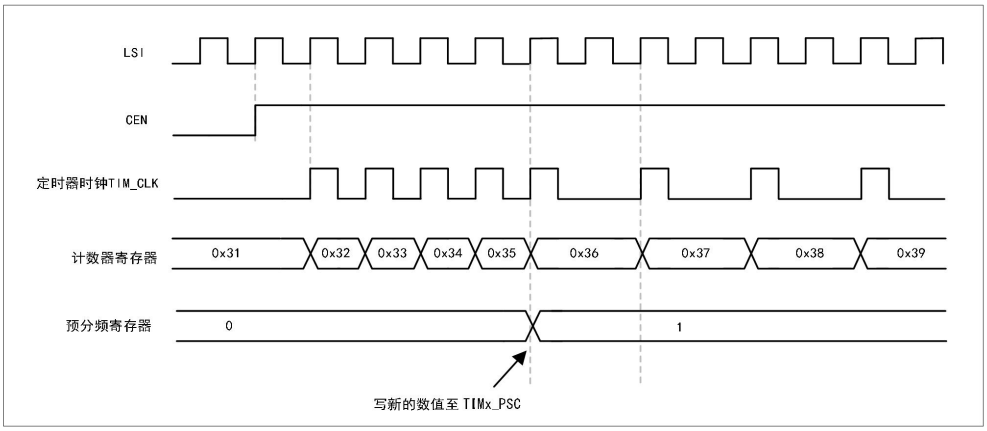
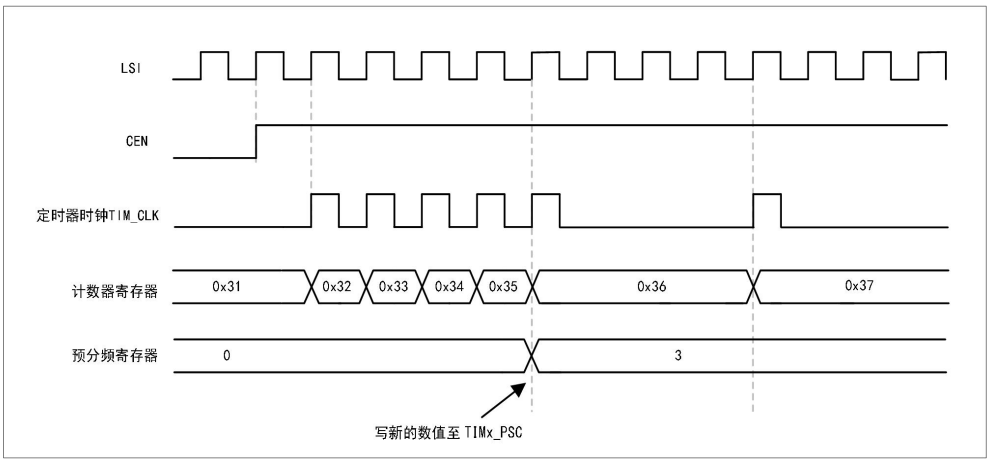


图 11-3 TIMx_ARR=0xFF 时，预分频系数从 1 变到 4 的计数器时序图



11.3.2 计数模式

TIM4 仅支持向上计数。需要确保(TIM4_CR2 寄存器)的”RSTC”位始终为 1。

11.3.2.1 向上计数模式

在向上模式中，计数器从 0 计数到自动重装载值(TIM4_ARR 寄存器中的值)，然后计数器溢出，重新从 0 开始计数并且产生一个计数重载事件 UEV。
每次计数器溢出时可以产生计数重载事件，当发生一个计数重载事件时：

- 硬件将计数重载标志位(TIM4_SR 寄存器中的 ARF 位)置’1’

下图给出一些例子，当 TIM4_ARR=0x36 时计数器在不同时钟频率下的动作。

图 11-4 计数器时序图，内部时钟分频因子为 1

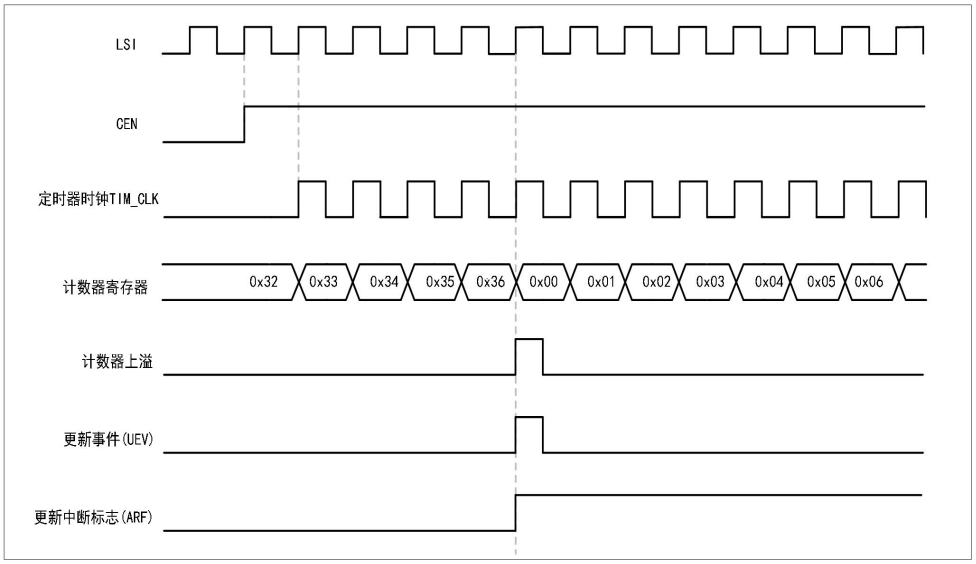


图 11-5 计数器时序图，内部时钟分频因子为 N

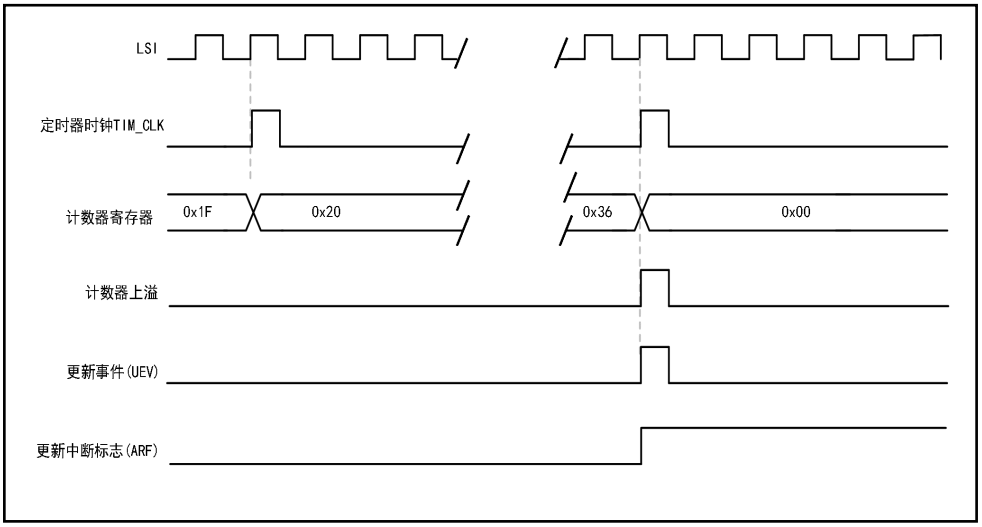
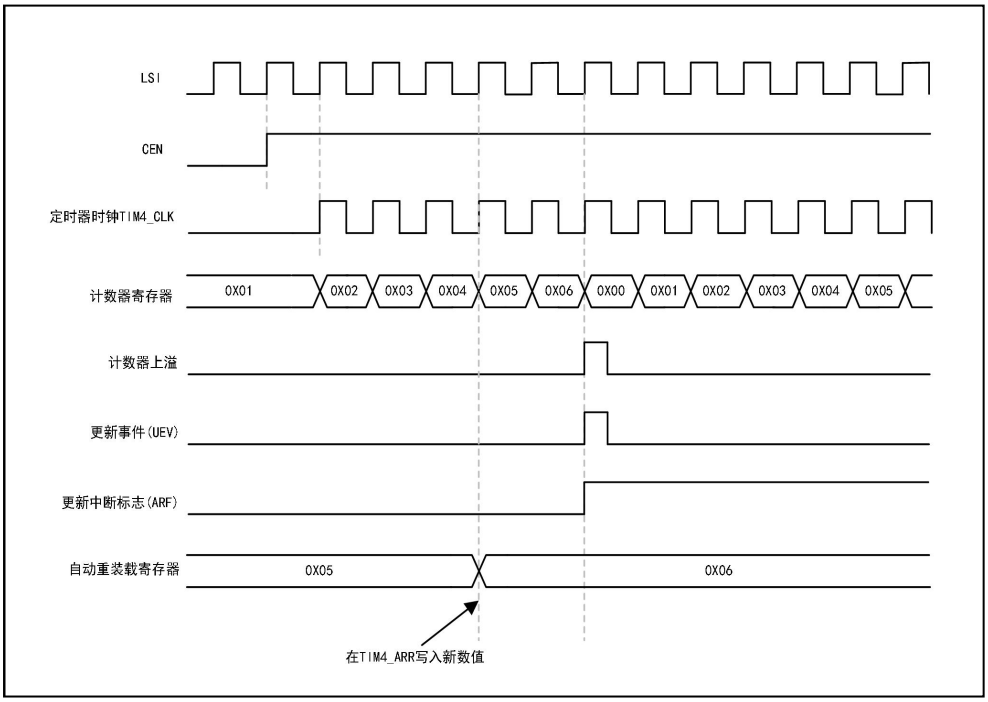


图 11-6 计数器时序图，修改自动重装载寄存器时的更新事件



注意：

1. 当 TIM4_ARR 中写入的新值小于当前计数值时，计数值将一直计数到 0xFFFF 溢出，再从 0X00 开始，计数到新的 TIM4_ARR 值。

2. 应当避免在 TIM4 计数器运行过程中修改 TIM4_ARR。

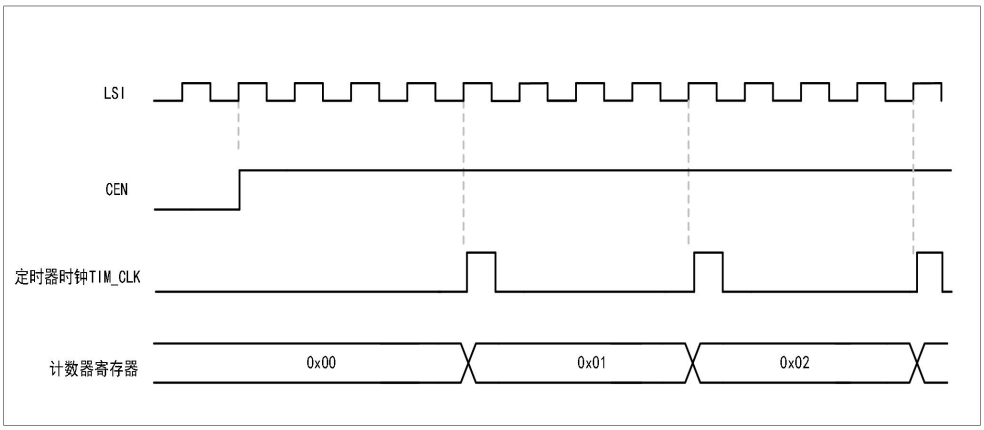
11.3.3 时钟源

定时器的时钟由 LSI 经预分频器分频得到的(TIM_CLK)提供。

TIM4_CR1 寄存器的 CEN 位是实际的控制位，只能通过软件改变它们。一旦 CEN 位置'1'，时钟源即向预分频器提供时钟。

下图示出控制电路和向下计数器在普通模式下，预分频值为 1 时的操作。

图 11-7 普通模式下的定时器时序图，内部时钟分频系数为 4

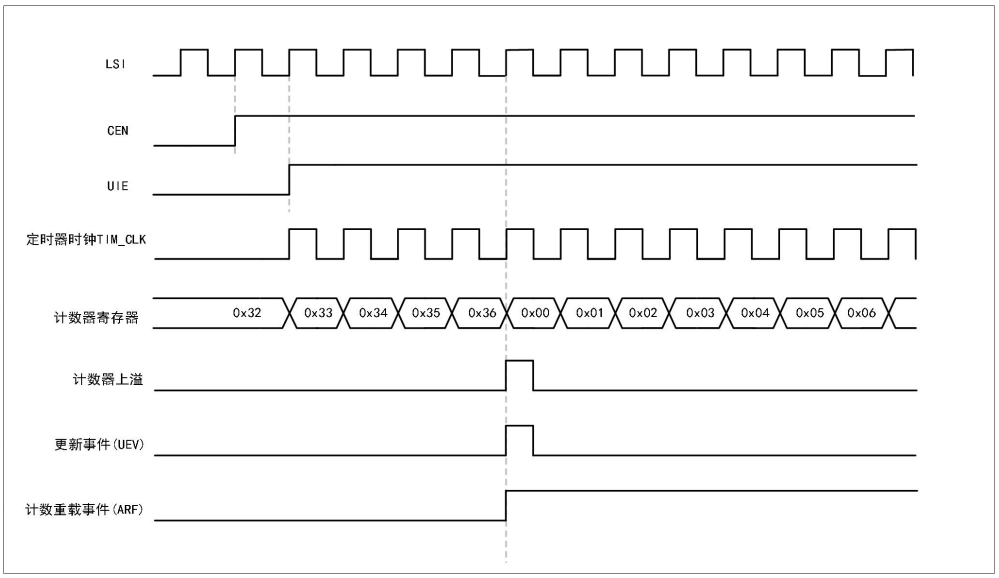


11.3.4 计数重载中断

通过将 TIM4 控制寄存器 2(TIM4_CR2)中的“ARI”位置'1'、定时器在产生计数重载事件后将同步生成一个更新中断请求到 NVIC，当 NVIC 中的 TIM4 中断被使能时，系统将产生对应的中断。

注意：使用 TIM4 中断时，应当考虑指令的操作是基于系统时钟 SYS_CLK 的，而 TIM4 的运行则是基于 LSI 的，在清除 TIM4 中断标志时，需要等待相关标志被清除，避免意外的结果。

图 11-8 计数重载事件与中断标志



11.4 TIM4 寄存器描述

11.4.1 TIM4 状态寄存器（TIM4_SR）

（地址： 0x4000_1C00 ）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ARF
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**ARF：** 计数重载标志**(auto reload flag)** 当计数器值与重载值匹配时， 该位由硬件置‘1’，软件对该位写“1”以将标记清除 0： 无匹配事件产生， 计数器未发生过溢出。 1： 计数器发生溢出， 计数器重载事件产生。
位 31：1	保留。必须保持为 0。

11.4.2 TIM4 控制寄存器 1(TIM4_CR1)

（地址： 0x4000_1C04 ）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	DBGE	-	-	-	-	-	-	UG	CEN
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Rw1	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**CEN**：使能定时器**(Counter enable)** 0： 关闭定时器 1： 使能定时器
位 1	**UG**：产生更新事件 **(Update generation)** 0： 无动作 1： 重新初始化计数器， 并产生一个更新事件。 ***注 1：计数器复位后，TIM4 的 CNT 始终由 0 开始计数;***
位 7：2	保留。必须保持为 0。
位 8	**DBGE**：TIM4 调试挂起位**(TIM debug)** 0： TIM4 在调试暂停时被挂起 1： TIM4 在调试暂停时仍旧工作
位 31：9	保留。必须保持为 0。

11.4.3 TIM4 预分频寄存器(TIM4_PSC)

（地址： 0x4000_1C08 ）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	PSC[15:0]															
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**PSC[15:0]**：预分频器的值 **(Prescaler value)** PSC 包含了当前实时装入预分频器寄存器的值
位 31：8	保留。必须保持为 0。

11.4.4 TIM4 计数器寄存器(TIM4_CNT)

（地址： 0x4000_1C10 ）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CNT[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**CNT[15:0]**：计数器的值 **(Counter value)**
位 31：16	保留。必须保持为 0。

11.4.5 TIM4 控制寄存器 2(TIM4_CR2)

（地址： 0x4000_1C14 ）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RSTC	ARI
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**ARI**：计数重载中断使能**(Auto reload interrupt enable)** 当计数器值与自动重装载值匹配时产生中断请求 0： 禁止计数器重载中断 1： 使能计数器重载中断
位 1	**RSTC**：复位计数器**(Reset counter)** 当计数器值与自动重装载值匹配时复位计数器 0： 计数重载时，不复位计数器 1： 计数重载时，复位计数器 ***注：当该位为 0 时：*** ***在向上计数：计数器将一直计数到 0XFFFF，接着计数器+1 溢出，才会将计数器值置为'0'。***
位 31：1	保留。必须保持为 0。

11.4.6 TIM4 自动重装载寄存器(TIM4_ARR)

（地址： 0x4000_1C18 ）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	ARR[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**ARR[15:0]**：自动重装载的值 **(Auto reload value)** 详细参考 11.3.1 节：有关 ARR 的更新和动作。 ***注：当自动重装载的值为 0 时，计数器不工作。***
位 31：16	保留。必须保持为 **0**。

11.4.7 寄存器列表

地址	寄存器	描述	备注
0x4000_1C00	TIM4_SR	TIM4 状态寄存器	TIM4_SR 说明
0x4000_1C04	TIM4_CR1	TIM4 控制寄存器 1	TIM4_CR1 说明
0x4000_1C08	TIM4_PSC	TIM4 预分频系数寄存器	TIM4_PSC 说明
0x4000_1C10	TIM4_CNT	TIM4 计数器寄存器	TIM4_CNT 说明
0x4000_1C14	TIM4_CR2	TIM4 控制寄存器 2	TIM4_CR2 说明
0x4000_1C18	TIM4_ARR	TIM4 自动重装载寄存器	TIM4_ARR 说明

12 独立看门狗(IWDG)

12.1 综述

独立看门狗模块可以用于解决处理器因为硬件或软件的故障所发生的错误。当计数器达到给定的超时值时，产生系统中断或者复位系统。

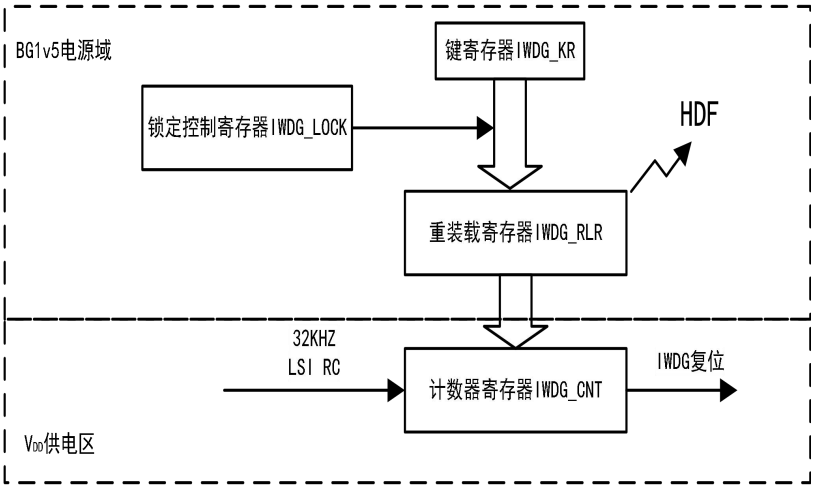
独立看门狗工作于片内低速振荡时钟 LSI，因此即使是主时钟失效时它仍然照常工作。适用于需要看门狗作为一个在主程序之外运行，能够完全独立工作，并且对时间精度要求较低的场合。

12.2 特性

- 递减计数器
- 时钟由片内低速振荡时钟 LSI 提供
- 睡眠模式下正常工作
- 看门狗被激活后，则在计数器计数至 0x000 时产生饿狗事件

12.3 IWDG 功能描述

图 12-1 独立看门狗框图



12.3.1 时基单元

独立看门狗本质上是一个基于 LSI 时钟的递减定时器，这个可编程定时器的主要部分是一个带有自动重装载功能的 32 位递减计数器，计数器的时钟固定为 32Khz 的 LSI 时钟。

时基单元包含：

- 自动重装载寄存器(IWDG_RLR)
- 计数器寄存器(IWDG_CNT)

设置 IWDG_CR 寄存器中的独立看门狗使能位(EN)使能计数器计数。

12.3.1.1 计时时间

32 位的独立看门狗计数器与自动重装载寄存器支持定义最大 131072S 的计时时间，通过下式决定合适的看门狗计时时间：

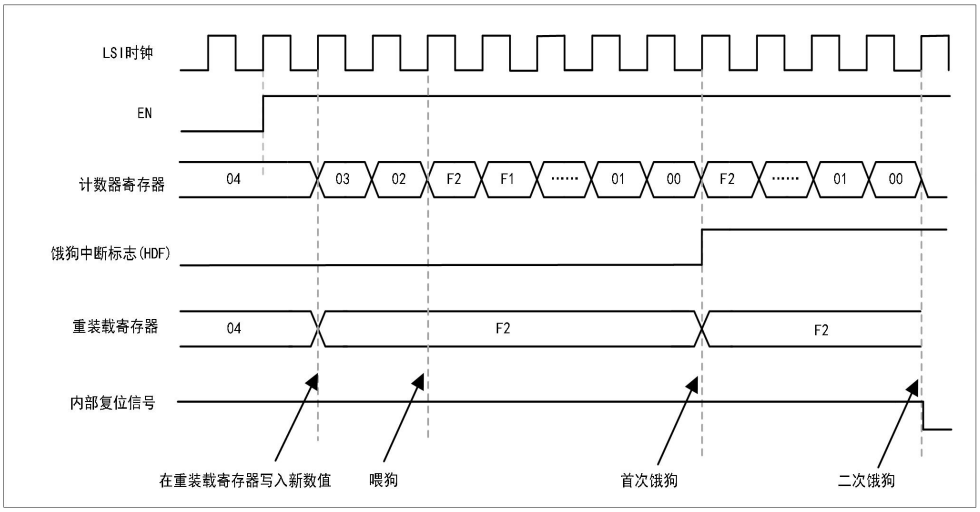
T_{IWDG}= RL[31:0] / f_{LSI}

其中：

- T_{IWDG} 为看门狗的计时时间，单位 S(秒)
- f_{LSI} 为 LSI 时钟的频率，单位 Hz

12.3.2 喂狗与饿狗

图 12-2 重装载寄存器从 04 到 F2 时，喂狗与饿狗时序图



看门狗用于在极端情况下，为单片机提供最后的保护；
在计时时间到达之前，应当通过(IWDG_KR)键寄存器写入一个任意值给予看门狗一个信号，用于将计数器复位至重装载值，这个信号称为”喂狗”；
在计时时间达到时，如果还没喂狗，则产生饿狗事件：

- 首次饿狗时，看门狗自动复位计数器至重装载值，(IWDG_SR 状态寄存器)中的饿狗事件标志被硬件置 1。
- 在饿狗中断标志被置 1 的情况下，产生二次饿狗，则系统发生一次看门狗复位。

产生饿狗事件，可能的原因有：

- 设定的计时时间过短
- 单片机遭受严重干扰导致程序跑飞

12.3.3 看门狗寄存器保护机制

看门狗提供了一种安全锁定的保护机制，可避免在系统运行过程中，因软件设计缺陷或程序跑飞时意外篡改 IWDG 外设的配置．一旦保护机制生效，IWDG 的所有寄存器都无法被软件改写。

12.3.3.1 生效保护机制

IWDG 的保护机制复位后默认生效，当保护机制生效时，读取(IWDG_LOCK)锁定控制寄存器的第 0 位”LOCK”位为 1；
当保护机制失效后需要再次生效保护机制，对(IWDG_LOCK)寄存器写入任意值即可。

12.3.3.2 失效保护机制

往(IWDG_LOCK)寄存器写入”0x1ACCE551”即可失效保护机制，当保护机制失效时，读取(IWDG_LOCK)锁定控制寄存器第 0 位”LOCK”位为 0；

12.3.4 独立看门狗中断

独立看门狗中断被连接到 NVIC 不可屏蔽中断中，它是默认启用且不可屏蔽的，当看门狗复位被禁止时，IWDG 计数器下溢出将产生一个中断请求到 NVIC；
喂狗以清除 IWDG 中断。

12.3.5 调试模式

当微控制器进入调试模式(Cortex-M0 核心停止)时，根据(IWDG_CR 寄存器)中”DBGE 位”的设置，独立看门狗计数器将继续计数或者停止计数。

注意：应当谨慎处理调试模式，避免在调试模式下，看门狗计数器溢出导致意外的复位。

12.4 寄存器描述

12.4.1 IWDG 重装载寄存器(IWDG_RLR)

（地址： 0x4000_3000）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	RL[31:16]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	RL[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位 31：0	**RL[31:0]**：看门狗计数器重装载值**(Reload value)** 该值的设定使用方法请参考 12.3.1.1 计时时间的描述

12.4.2 IWDG 计数器寄存器(IWDG_CNT)

（地址： 0x4000_3004）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	CNT[31:16]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CNT[15:0]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位 31：0	**CNT[31:0]**：独立看门狗计数器的值**(Counter value)**

12.4.3 IWDG 控制寄存器(IWDG_CR)

（地址： 0x4000_3008）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	DBGE	RSTE	EN
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**EN**：独立看门狗使能控制**(enable)** 0：　　IWDG 模块禁止，无中断响应 1：　　IWDG 模块使能，同时 IWDG 中断被使能
位 1	**RSTE**：独立看门狗复位控制**(reset enable)** 0：　　IWDG 复位禁止 1：　　IWDG 复位使能
位 2	**DBGE**：独立看门狗调试挂起控制**(IWDG debug)** 0：　　IWDG 在调试暂停时被挂起 1：　　IWDG 在调试暂停时仍旧工作
位 31：3	保留。必须保持为 0。

12.4.4 IWDG 键寄存器(IWDG_KR)

（地址： 0x4000_300C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	KEY[31:16]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	KEY[15:0]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 31：0	**KEY[31:0]：键值(key value)** 该寄存器为只写，读为 0 在保护机制失效后，对该寄存器写入任意数值都会： ● 将(IWDG_RLR)中的值重新载入到(IWDG_CNT)寄存器中，并重新开始向下计数； ● 将(IWDG_SR)中的计数溢出标志位清 0，如果已经被置 1；

12.4.5 IWDG 状态寄存器(IWDG_SR)

（地址： 0x4000_3010）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	HDF
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**HDF：饿狗中断标志(hungry dog flag)** 0： IWDG 未发生饿狗 1： IWDG 发生饿狗，将保护机制失效后，写 KR 寄存器以将其清 0
位 31：1	保留。必须保持为 0。

12.4.6 IWDG 保护机制控制寄存器(IWDG_LOCK)

（地址： 0x4000_3400）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	LOCK[31:16]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	LOCK[15:0]															KEY[0] (W) LOCK (R)
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位 31：0(W)	**KEY[31:0]**：保护机制控制位**(lock key value)** 0x1ACCE551：　IWDG 保护机制失效 其他：　IWDG 保护机制生效
位 0(R)	**LOCK**：锁状态**(Lock status)** 0：　当前 IWDG 保护机制失效中 1：　当前 IWDG 保护机制生效中 ***注：保护机制默认生效***

12.4.7 寄存器列表

地址	寄存器	描述	备注
0x4000_3000	IWDG_RLR	IWDG 重装载寄存器	IWDG_RLR 说明
0x4000_3004	IWDG_CNT	IWDG 计数器寄存器	IWDG_CNT 说明
0x4000_3008	IWDG_CR	IWDG 控制寄存器	IWDG_CR 说明
0x4000_300C	IWDG_KR	IWDG 键寄存器	IWDG_KR 说明
0x4000_3010	IWDG_SR	IWDG 状态寄存器	IWDG_SR 说明
0x4000_3400	IWDG_LOCK	IWDG 保护机制控制寄存器	IWDG_LOCK 说明

13 串行外设接口(SPI)

13.1 综述

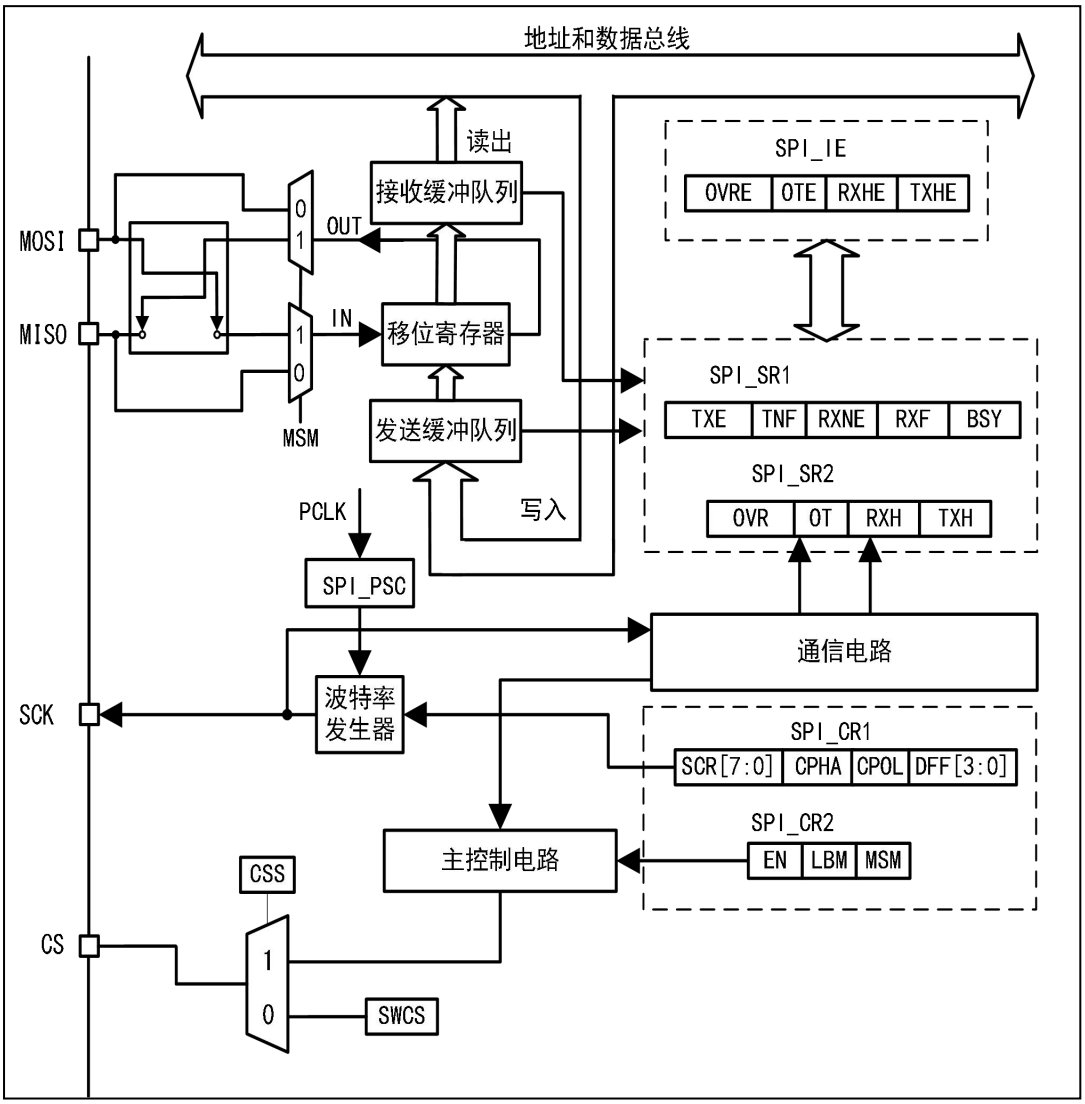
串行外设接口(SPI)允许芯片与外部设备以全双工、同步、串行方式通信。此接口可以被配置成主机或从机模式，并为外部从机设备提供通信时钟(SCK)。

13.2 特性

- 4 线(CS & MISO & MOSI & SCK)全双工同步传输
- 4~16 位可编程的数据帧格式选择
- 主机或从机工作模式
- 8 位的时钟预分频系数(SCR 位)
- 8 位的波特率预分频系数(通过 BR 位)
- 主模式和从模式下均可以选择由软件或硬件进行 CS 管理
- 可编程的时钟极性和相位
- 可触发中断的发送和接收标志
- SPI 总线忙状态标志
- 最大 8 级接收/发送缓冲队列
- 可触发中断的缓冲队列状态标志

13.3 功能描述

图 13-1 SPI 框图



通常 SPI 通过 4 个引脚与外部器件相连：

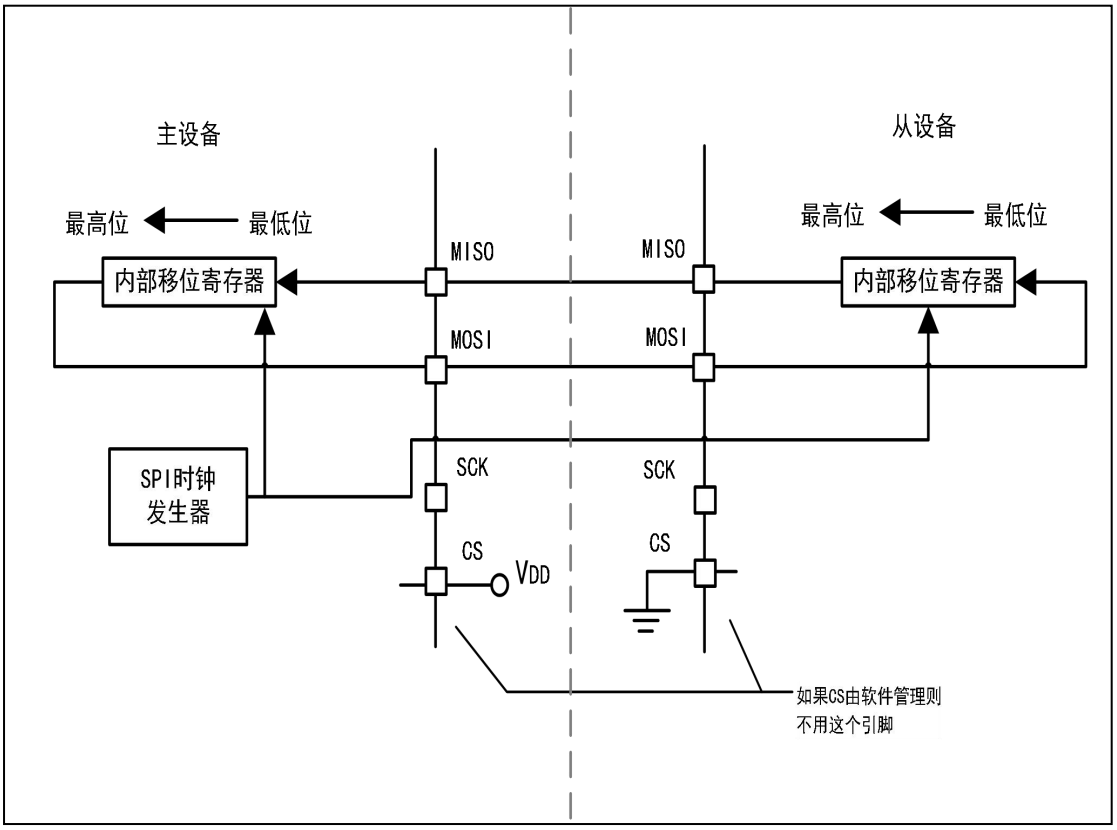
MISO:主机输入/从机输出引脚。该引脚在从机模式发送数据、在主机模式接收数据。

MOSI:主机输出/从机输入引脚。该引脚在主机模式发送数据，在从机模式接收数据。

SCK:串口时钟引脚。作为主机的输出，从机的输入。

CS：从机设备选择引脚。这是一个可选的引脚，用于选择从机。它作为"片选引脚"，让主设备可以单独地与特定从设备通讯，避免数据线上的冲突。

图 13-2 单个主机和单个从机的应用



MOSI 脚相互连接、MISO 脚相互连接。数据在主和从之间串行地传输(高位在前)。

13.3.1 SPI 通讯

通信总是由主机发起。主机通过 MOSI 脚把数据发送给从机，从机通过 MISO 引脚回传数据。这意味全双工通信的数据输出和数据输入是用同一个时钟信号同步的；时钟信号由主机通过 SCK 脚提供。

13.3.1.1 SPI 波特率

SPI 波特率由 PCLK 经 BR[7:0]和 SCR{7:0}分频而来：

$f_{SCK} = f_{PCLK} / (BR^* (1+SCR))$

注意：

1.BR[7:0]的第 0 位必须始终保持为 0，取值范围必须是 2~254 之间的偶数。

2.SPI 通讯波特率不应该超过数据手册当中规定的上限

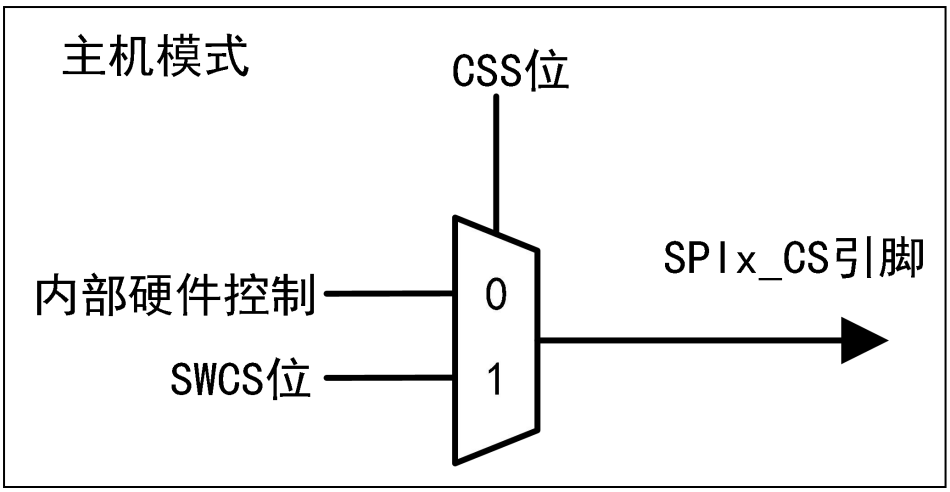
13.3.1.2 从机选择信号(CS)控制

有 2 种 CS 控制方式：

- 软件 CS 控制：通过将(SPI_CSS)寄存器的 CSS 位置'1'来使能该模式(见图 13-3)。在该模式下、SPI_CS 引脚可以通过 SWCS 来控制，按照工作模式有：
  - 配置 SPI 为主机模式时、SWCS 的状态对应着 SPI_CS 引脚的状态；
  - 配置 SPI 为从机模式时、SWCS 的状态对 SPI 的通信无影响。

- 硬件 CS 模式，通过将 CSS 位清'0'来使能该模式、按照工作模式有：
  - 配置 SPI 为主机模式时、CS 输出被使能、CS 引脚被自动拉低；
  - 配置 SPI 为从机模式时、CS 输入被使能、由外部主机管理 CS 信号。

图 13-3 硬件/软件的从机选择管理



13.3.1.3 时钟信号的相位和极性

(SPI_CR1)寄存器的 CPOL 和 CPHA 位，能够组合成四种可能的时序关系。

CPOL(时钟极性)位控制在没有数据传输时、时钟的空闲状态电平；CPHA(时钟相位)位控制在第几个边沿进行数据的采样和锁存：

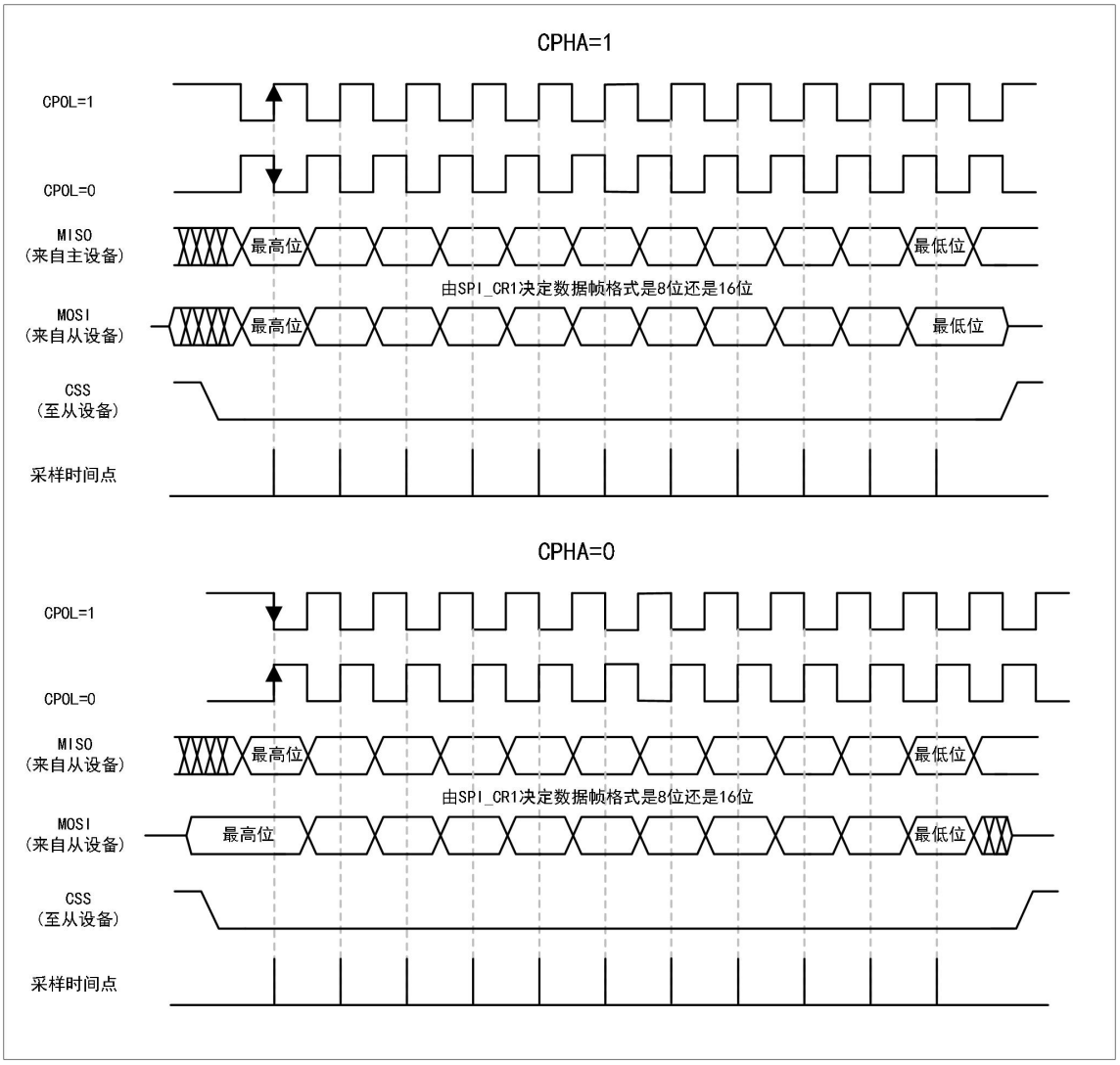
- 如果 CPOL 被清'0'，SCK 引脚在空闲状态保持低电平；
- 如果 CPOL 被置'1'，SCK 引脚在空闲状态保持高电平；
- 如果 CPHA 被置'1'，SCK 时钟在第二个边沿(CPOL='0'时为下降沿，CPOL='1'时为上升沿)进行数据位的采样，采样完毕由内部移位寄存器锁存数据。
- 如果 CPHA 被清'0'，SCK 时钟在第一个边沿(CPOL='0'时为上升沿，CPOL='1'时为下降沿)进行数据位采样，采样完毕由内部移位寄存器锁存数据。

图 13-4 显示了 SPI 通讯的时序图。

注意： *1. 在改变 CPOL/CPHA 位之前，必须清除 SPE 位将 SPI 禁止。*

2. 主机和从机必须确保处于相同的时序模式和帧数据格式。

图 13-4 时序图



13.3.1.4 数据帧格式

每一帧数据的输出，总是以最高有效位(MSB)开始。

根据(SPI_CR1)寄存器的 DFF 位，每个数据帧可以是 4 位到 16 位。所选择的数据帧格式对发送和接收均生效。

13.3.1.5 缓冲队列(FIFO)

SPI 的发送和接收均集成了一个 8 级的先入先出（FIFO）缓冲队列，这个缓冲队列中，每一级缓冲有 16 位的数据有效位，它们数据格式都是右对齐的，并且与 SPI_DR 寄存器实时同步；

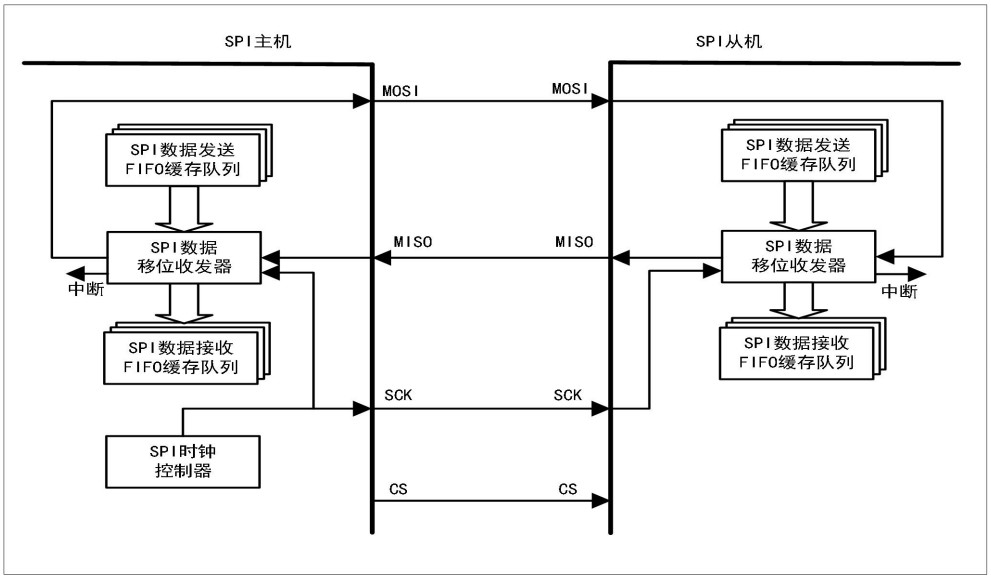
(SPI_CR1)寄存器 DFF 位定义的数据帧、不论大小(4 位~16 位)，每一帧数据均占用一级缓冲，缓冲队列在发送和接收时具体的表现为：

- 在发送时:
  - 发送的数据以'帧'为单位、首先存放到缓冲队列当中；
  - 每往(SPI_DR)寄存器写入一帧数据、缓冲队列都会被更新，直到存满 8 帧数据，(SPI_SR1)的'TNF'位将被清'0'(发送 FIFO 全满)。
  - 缓冲队列最多缓冲 8 帧数据，发送缓冲全满时、往(SPI_DR)寄存器继续写入的帧数据将被丢弃。

- 在接收时：
  - 接收到的数据以'帧'为单位、通过内部移位寄存器首先存放在缓冲当中；
  - 当读取(SPI_DR)寄存器时，先收到的数据会被先同步到(SPI_DR)寄存器中，直到缓冲为空，SPI_DR 的值将不会被更新，这意味着此时读到的数据为最后一帧数据；
  - 缓冲队列最多缓冲 8 帧数据，如果没有及时读取 SPI_DR 中的数据，则当第 9 帧数据来临时，该帧数据被丢弃。

特别的、SPI 还提供了发送和接收缓冲队列半满的状态，可提高对 SPI 总线的利用率。

图 13-5 缓冲队列的结构



13.3.2 状态标志

应用程序通过 3 个状态标志可以完全监控 SPI 总线的状态。

13.3.2.1 发送缓冲器空闲(TXE)

此标志为'1'时、表示发送缓冲队列为空，当写入一帧数据到(SPI_DR)寄存器时，TXE 标志被清'0'。

13.3.2.2 接收缓冲器非空(RXNE)

此标志为'1'时、表明在接收缓冲队列中包含至少一帧的有效数据。读(SPI_DR)寄存器可以清除此标志。

13.3.2.3 忙(BSY)

BSY 标志由硬件设置与清除，此标志表明 SPI 通信层的状态。

此标志为'1'时，表明 SPI 正忙于通信，在软件要关闭 SPI 模块之前，可以使用 BSY 标志检测传输是否结束，这样可以避免破坏最后一次传输。

当传输开始时， BSY 标志被置'1'，以下情况时此标志将被清除为'0'：

- 当传输结束；
- 当关闭 SPI 模块；

如果通信不是连续的，则在每个数据项的传输之间， BSY 标志为低。

当通信是连续时：

- 主模式下：在整个传输过程中， BSY 标志保持为高；
- 从模式下：在每个数据项的传输之间， BSY 标志在一个 SPI 时钟周期中为低。

注意：不要使用 BSY 标志处理每一帧数据项的发送和接收，最好使用 TXE 和 RXNE 标志。

13.3.3 错误标志

应用程序通过 2 个错误状态标志可以保障 SPI 通信的鲁棒性。

13.3.3.1 溢出错误

缓冲队列最多缓冲 8 帧数据。

如果没有及时读取 SPI_DR 中的数据，则当第 9 帧数据来临时，该帧数据被丢弃，SPI 产生 OVR 事件，如果将 OVRE 置'1'，则产生 OVR 中断。

即时读出 SPI_DR 寄存器中由缓冲队列同步的帧数据、以避免 OVR 错误；

在 OVR 错误产生时，将(SPI 中断标志清除寄存器(SPI_IFC)中)的"OVR"位置'1'以清除 OVR 错误。

13.3.3.2 超时错误

缓冲队列最多缓冲 8 帧数据。

当接收缓冲队列非空(缓冲队列中至少有一帧数据)，但软件没有在一定时间内(32 个 SPI 通讯时钟)进行任何的读取数据操作，SPI 产生 OT 事件，如果将 OTE 置'1'，则产生 OT 中断。

即时读出 SPI_DR 寄存器中由缓冲队列同步的帧数据、可将 OT 清除。

13.3.4 配置 SPI 为主机模式

13.3.4.1 配置步骤

- 通过(SPI_CR1)的 SCR[7:0]位和(SPI_BR)的 BR[7:0]位共同定义 SPI 波特率。
- 选择 CPOL 和 CPHA 位，定义数据传输和串行时钟间的相位关系(见图 13-3)，主机和从机的相位与极性必须相同。
- 设置 DFF 位来定义 4 位到 16 位的数据帧格式。
- 通过(SPI_CSS)寄存器的 CSS 位，选择 CS 引脚的控制方式。
- 将(SPI_CR2)寄存器的 SPE 位置'1'，以使能 SPI。

在这个配置中， MOSI 引脚是数据输出，而 MISO 引脚是数据输入。

13.3.4.2 数据发送过程

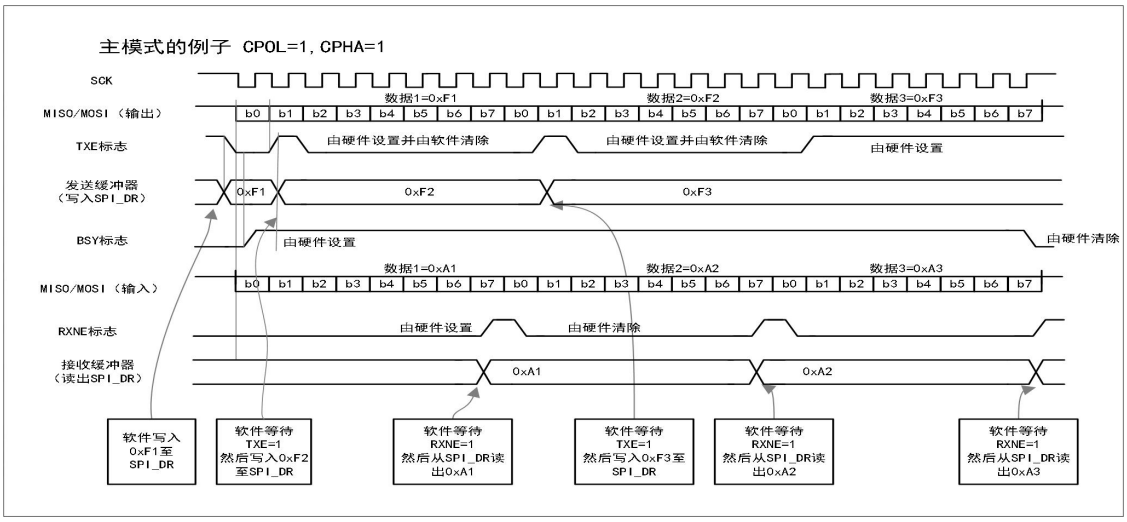
当写入数据至(SPI_DR)时，数据首先被同步到发送缓冲队列，接着发送过程开始。
在发送第一个数据位时，数据被并行地(通过内部总线)传入到内部移位寄存器，而后串行地移出到 MOSI 脚上；
当缓冲队列中的第 4 帧数据被传输到移位寄存器后、TXH 标志将被置位，如果设置了(SPI_IE)寄存器中的 TXHE 位，将产生 TXH 中断；
应当谨慎使用 TXH 中断，除非数据是在中断里被更新到(SPI_DR)寄存器中。

13.3.4.3 数据接收过程

对于接收器，当数据传输完成时：

- 内部移位寄存器里的数据被同步到接收缓冲队列，RXNE 标志被硬件置'1'。
- 当缓冲队列中接收到第 4 帧数据后、RXH 标志将被置位，如果设置了 SPI_IE 寄存器中的 RXHE 位，则产生中断。

当缓冲队列中数据为空时，硬件将自动清除 RXNE 位。
应当谨慎使用 RXH 中断，除非数据是在中断里被更新到(SPI_DR)寄存器中。
图 13-6 主模式、全双工模式下连续传输时，TXE/RXNE/BSY 的变化示意图



13.3.5 配置 SPI 为从机模式

在从模式配置中，从 SPIx_CLK 引脚上接收主器件的串行时钟。SPIx_CR1 寄存器的”PSC[2:0]”位和 SPIx_BR 寄存器的”BR[7:0]”位中设置的值不会影响传输率。

13.3.5.1 配置步骤

- 设置 DFF 位来定义 4 位到 16 位的数据帧格式。
- 选择 CPOL 和 CPHA 位，定义数据传输和串行时钟间的相位关系(见图 1-3)，主机和从机的相位与极性必须相同。
- 通过(SPI_CSS)寄存器的 CSS 位，选择 CS 引脚的控制方式。
- 将(SPI_CR2)寄存器的 MSM 位置’1’，以将 SPI 配置为从机模式
- 将(SPI_CR2)寄存器的 SPE 位置’1’，以使能 SPI。

注意：在外部主机发送时钟之前、从机应该首先使能，否则从机将可能收到意外数据。

13.3.5.2 数据发送过程

当从设备收到时钟信号，并且在 MOSI 引脚上出现第一个数据位时，发送过程开始。主机和从机通讯是同步的，这意味着接收移位寄存器每接收到一个位的数据，内部发送移位寄存器也同样要发送一个位的数据到主机。

当缓冲队列中的第 4 帧数据被传输到移位寄存器后、TXH 标志将被置位，如果设置了(SPI_IE)寄存器中的 TXHE 位，将产生 TXH 中断；

应当谨慎使用 TXH 中断，除非数据是在中断里被更新到(SPI_DR)寄存器中。

13.3.5.3 数据接收过程

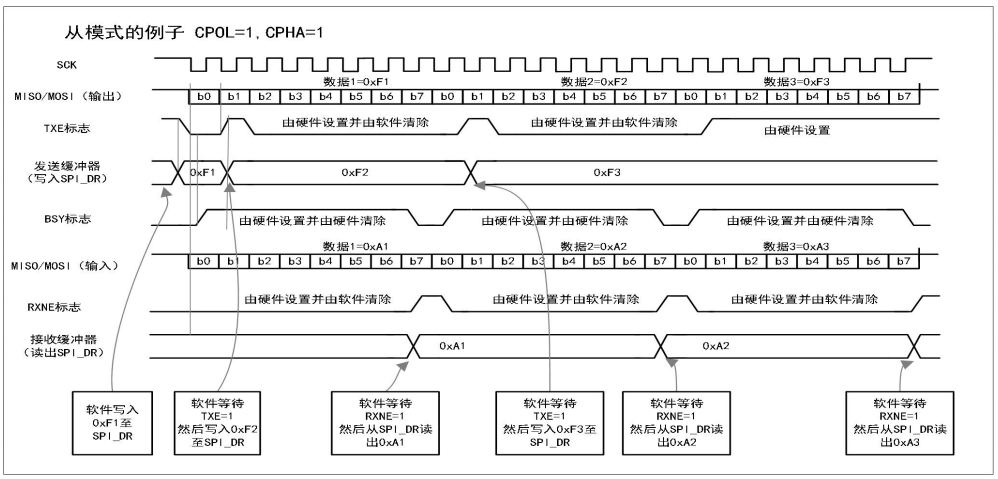
对于接收器，当数据传输完成时：

- 内部移位寄存器里的数据被同步到接收缓冲队列，RXNE 标志被硬件置’1’。
- 当缓冲队列中接收到第 4 帧数据后、RXH 标志将被置位，如果设置了 SPI_IE 寄存器中的 RXHE 位，则产生中断。

当缓冲队列中数据为空时，硬件将自动清除 RXNE 位。

应当谨慎使用 RXH 中断，除非数据是在中断里被更新到(SPI_DR)寄存器中。

图 13-7 从模式、全双工模式下连续传输时，TXE/RXNE/BSY 的变化示意图



13.3.6 连续和非连续传输

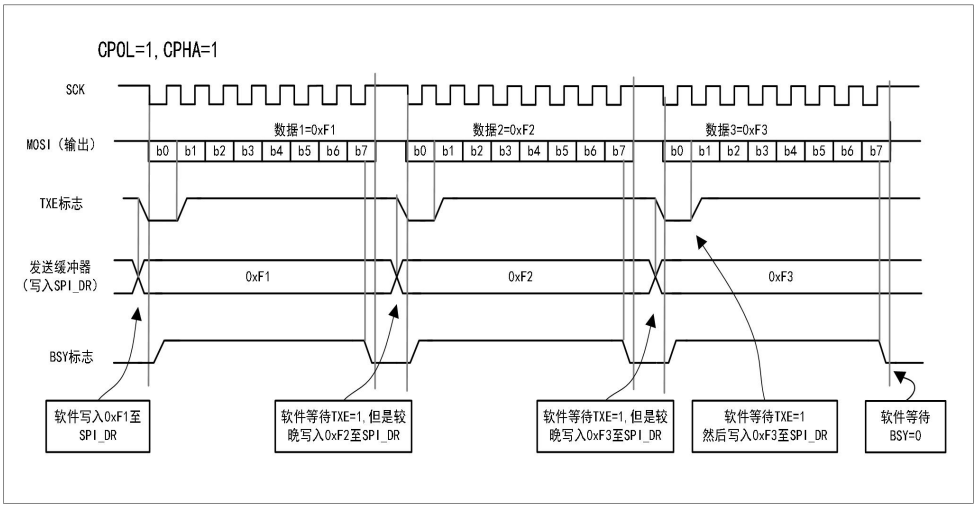
通过(SPI_SR2)寄存器 TXH 位的状态(或中断)和对 8 级缓冲队列的灵活利用，可以确保在满足 SPI 最大频率的条件下，当缓冲队列中最后一帧数据被发送前，能有足够的时间往(SPI_DR)寄存器写入新的数据，以实现数据的连续传输。

如果 SPI 频率够低，或者程序足够快，那么也可以通过查询(SPI_SR1)寄存器 TNF 位的状态来实现连续传输。

在连续传输时，SPI 时钟将保持连续、BSY 也不会被清除。

如果上述条件不被满足，将导致不连续的通信，此时、SPI 时钟将不连续、BSY 在每帧数据传输之间都会被清除。

图 13-8 非连续传输发送时，TXE/BSY 的变化示意图



13.3.7 关闭 SPI

当通讯结束，可以通过关闭 SPI 模块来终止通讯。清除 SPE 位即可关闭 SPI。

在某些配置下，如果传输还未完成就关闭 SPI 模块并进入停机模式，则可能导致当前的传输被破坏，而且 BSY 标志也变得不可信。

为了避免发生这种情况，关闭 SPI 模块时，建议按照下述步骤操作：

- 等待 RXNE=1 并接收最后一个数据；
- 等待 BSY=0；
- 关闭 SPI(SPE=0)，最后进入停机模式(或关闭该模块的时钟)。

13.3.8 SPI 中断

SPI 模块的中断请求：

中断事件	事件标志	使能控制位
接收 FIFO 溢出中断	OVR	OVRE
接收 FIFO 超时中断	OT	OTE
接收 FIFO 半满中断	RXH	RXHE
发送 FIFO 过半中断	TXH	TXHE

上述事件的状态也可以通过(SPI_CR2)寄存器查询；

13.3.8.1 接收 FIFO 溢出

接收缓冲队列最多缓冲 8 帧数据，如果没有及时读取(SPI_DR)中的数据，则当第 9 帧数据来临时，该帧数据被丢弃，SPI 产生接收 FIFO 溢出，其状态只能由软件清除。

13.3.8.2 接收 FIFO 超时

当接收缓冲队列非空(缓冲队列存在一帧或多帧数据)，且软件没有及时(在 32 个 SPI 时钟内)读取(SPI_DR)中的数据，SPI 产生接收 FIFO 超时，其状态只能由软件清除。

13.3.8.3 接收 FIFO 半满

当接收缓冲队列中半满(帧数据≥4)时，SPI 产生接收 FIFO 半满；
软件可以多次读取(SPI_DR)中的数据、直到缓冲队列中的帧数据<4，接收 FIFO 半满状态由硬件自动清除。

13.3.8.4 发送 FIFO 过半

当发送缓冲队列半满(帧数据≥4)时，SPI 产生发送 FIFO 过半；
发送缓冲队列中的帧数据按照“先入先出”的规则，通过内部移位寄存器发送出去，直到缓冲队列中的帧数据<4，发送 FIFO 过半状态由硬件自动清除。

13.4 寄存器描述

13.4.1 SPI 控制寄存器 1 (SPI_CR1)

（地址：0x4001_3000）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	SCR[7:0]								CPHA	CPOL	-	-	DFF[3:0]			
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 3：0	**DFF[3:0]**：帧数据格式**(Data frame format)** 0000： 保留 0001： 保留 0010： 保留 0011： 数据帧长度为 4 位 0100： 数据帧长度为 5 位 0101： 数据帧长度为 6 位 0110： 数据帧长度为 7 位 0111： 数据帧长度为 8 位 1000： 数据帧长度为 9 位 1001： 数据帧长度为 10 位 1010： 数据帧长度为 11 位 1011： 数据帧长度为 12 位 1100： 数据帧长度为 13 位 1101： 数据帧长度为 14 位 1110： 数据帧长度为 15 位 1111： 数据帧长度为 16 位 ***注：仅当 SPI 禁止（SPE=0）时，才能写入改位***
位 5：4	保留。必须保持为 **0**。
位 6	**CPOL**：时钟极性**(Clock polarity)** 0： 空闲状态时， SCK 保持低电平； 1： 空闲状态时， SCK 保持高电平。 ***注：仅当 SPI 禁止（SPE=0）时，才能写入改位***
位 7	**CPHA**：时钟相位**(Clock phase)**

	0： 数据采样从第一个时钟边沿开始； 1： 数据采样从第二个时钟边沿开始。 ***注：仅当 SPI 禁止（SPE=0）时，才能写入改位***
位 15：8	**SCR[7:0]：SPI 时钟分频系数(Prescaler value)** 详见 13.3.1.1 SPI 波特率介绍 ***注：仅当 SPI 禁止（SPE=0）时，才能写入改位***
位 31：16	保留。必须保持为 0。

13.4.2 SPI 控制寄存器 2(SPI_CR2)

（地址：0x4001_3004）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	MSM	SPE	LBM
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**LBM：环回模式控制(loopback mode)** 0： 正常工作模式 1： 环回传输模式（自发自收） ***注：仅当 SPI 禁止（SPE=0）时，才能写入改位***
位 1	**SPE：SPI 使能(SPI Enable)** 0： SPI 禁止 1： SPI 使能 ***注：当关闭 SPI 设备时，参考"关闭 SPI"一节描述。***
位 2	**MSM：主从模式(master-slave mode)** 0： 主机模式 1： 从机模式 ***注：仅当 SPI 禁止（SPE=0）时，才能写入改位***
位 31：3	保留。必须保持为 0。

13.4.3 SPI 数据寄存器(SPI_DR)

（地址：0x4001_3008）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	DR[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 15：0	**DR[15:0]**：数据寄存器**(SPI data)** 待发送或者已经收到的数据 数据寄存器对应两个缓冲队列：一个用于写(发送缓冲)，另外一个用于读(接收缓冲)： ● 写操作时硬件将数据同步发送缓冲队列； ● 读操时作硬件将返回接收缓冲队列里的数据； 数据的格式均为右对齐，根据 (SPI_CR1)中 DFF 位对于数据帧格式的选择，数据的发送和接收可以是 4 位~16 位的： ● 对于 4 位的数据、发送和接收时只会用到 SPI_DR[3:0]； ● 对于 5 位的数据、发送和接收时只会用到 SPI_DR[4:0]； ● 以此类推，对于 16 位的数据、发送和接收时将会用到 SPI_DR[15:0]；
位 31：16	保留。必须保持为 0。

13.4.4 SPI 状态寄存器(SPI_SR1)

（地址： 0x4001_300C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	BSY	RXF	RXNE	TNF	TXE
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

位 0	**TXE**：发送 **FIFO** 为空**(Transmit FIFO is empty)** 0： 发送 FIFO 非空 1： 发送 FIFO 为空
位 1	**TNF**：发送 **FIFO** 未满**(Transmit FIFO is not full)** 0： 发送 FIFO 全满 1： 发送 FIFO 未满
位 2	**RXNE**：接收 **FIFO** 非空**(Receive FIFO is not empty)** 0： 接收 FIFO 为空 1： 接收 FIFO 非空
位 3	**RXF**：接收 **FIFO** 全满**(Receive FIFO is full)** 0： 接收 **FIFO** 未满 1： 接收 **FIFO** 全满
位 4	**BSY**：数据传输状态标志位**(busy flag)** 0： SPI 处于空闲状态 1： SPI 处于传输数据状态或者发送 FIFO 非空
位 31：5	保留。必须保持为 0。

13.4.5 SPI 波特率预分频寄存器(SPI_BR)

（地址：0x4001_3010）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	BR[7:0]							
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 7：0	**BR[7:0]：SPI 波特率分频系数(Baud rate control)** BR 必须是 2~254 之间的偶数(bit0 只读，恒为 0)
位 31：8	保留。必须保持为 0。

13.4.6 SPI 中断使能寄存器(SPI_IE)

（地址：0x4001_3014）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	TXHE	RXHE	OTE	OVRE
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**OVRE**：接收 **FIFO** 溢出中断使能**(Receive FIFO overflow interrupt enable)** 0：　禁止接收 FIFO 溢出错误中断 1：　允许接收 FIFO 溢出错误中断
位 1	**OTE**：接收 **FIFO** 超时中断使能**(Receive FIFO over time interrupt enable)** 0：　禁止接收 FIFO 超时错误中断 1：　允许接收 FIFO 超时错误中断
位 2	**RXHE**：接收 **FIFO** 半满中断使能**(Receive FIFO half full interrupt enable)** 0：　禁止接收 FIFO 半满中断 1：　允许接收 FIFO 半满中断
位 3	**TXHE**：发送 **FIFO** 过半中断使能**(Transmit FIFO more than half interrupt enable)** 0：　禁止发送 FIFO 过半中断 1：　允许发送 FIFO 过半中断
位 31：4	保留。必须保持为 0。

13.4.7 SPI 状态寄存器 2(SPI_SR2)

（地址： 0x4001_3018）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	TXH	RXH	OT	OVR
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

位 0	**OVR**：接收 **FIFO** 溢出错误**(Receive FIFO overflow error flag)** 0： 接收 FIFO 未发生溢出错误 1： 接收 FIFO 溢出错误
位 1	**OT**：接收 **FIFO** 超时错误**(Receive FIFO overtime error flag)** 0： 接收 FIFO 未发生超时错误 1： 接收 FIFO 超时错误
位 2	**RXH**：接收 **FIFO** 数据半满**(Receive FIFO half full flag)** 0： 接收 FIFO 未半满 1： 接收 FIFO 半满
位 3	**TXH**：发送 **FIFO** 数据过半标志位**(Transmit FIFO more than half flag)** 0： 发送 FIFO 数据未过半 1： 发送 FIFO 数据已过半
位 31：4	保留。必须保持为 0。

13.4.8 SPI 中断标志清除寄存器(SPI_IFC)

（地址：0x4001_3020）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OT	OVR
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**OVR**：清除接收 FIFO 溢出中断标志**(Clear receive FIFO overflow error flag)** 0：　无效 1：　清除接收 FIFO 溢出错误中断
位 1	**OT**：清除接收 FIFO 超时中断标志**(Clear receive FIFO overtime error flag)** 0：　无效 1：　清除接收 FIFO 超时错误中断
位 31：2	保留。必须保持为 0。

13.4.9 SPI 片选信号控制寄存器(SPI_CSS)

（地址：0x4001_3028）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	SWCS	CSS	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 1:0	保留。必须保持为 0。
位 2	**CSS**：片选信号控制方式选择**(Chip select signal selection)** **0**：　　**SPI** 模块硬件自动控制 **1**：　　软件通过配置 **SWCS** 位控制
位 3	**SWCS**：软件片选信号控制**(Software Chip select signal control)** 0：　　CS 输出低电平 1：　　CS 输出高电平
位 31：4	保留。必须保持为 0。

13.4.10 寄存器列表

地址	寄存器	描述	备注
0x4001_3000	SPI_CR1	SPI 控制寄存器 1	SPI0_CR1 说明
0x4001_3004	SPI_CR2	SPI 控制寄存器 2	SPI0_CR2 说明
0x4001_3008	SPI_DR	SPI 数据寄存器	SPI0_DR 说明
0x4001_300C	SPI_SR1	SPI 状态寄存器 1	SPI0_SR1 说明
0x4001_3010	SPI_BR	SPI 波特率分频寄存器	SPI0_BR 说明
0x4001_3014	SPI_IE	SPI 中断使能寄存器	SPI0_IE 说明
0x4001_3018	SPI_SR2	SPI 状态寄存器 2	SPI0_SR2 说明
0x4001_3020	SPI_IFC	SPI 中断标志清除寄存器	SPI0_IFC 说明
0x4001_3028	SPI_CSS	SPI 片选信号控制寄存器	SPI0_CSS 说明

14 I2C 接口

14.1 综述

I2C 总线接口连接微控制器和串行 I2C 总线。
它提供多主机功能，控制所有 I2C 总线特定的时序、协议、仲裁和定时。通过一个可编程的分频器、支持最大 1Mbps 速率的多种通信模式。

14.2 特性

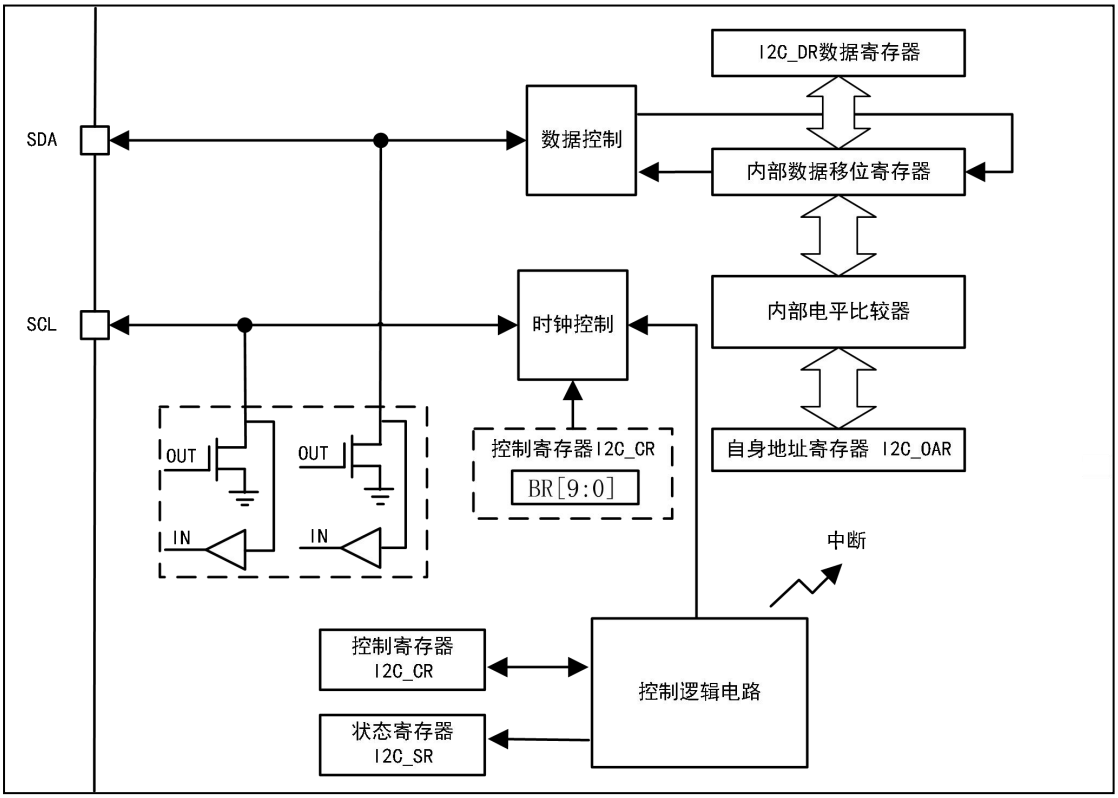
- 并行总线 I2C 总线协议转换器
- 支持不同的通讯速度
  - 标准速度(高达 100 kHz)
  - 快速(高达 400 kHz)
- 27 种通信状态，除空闲状态外，均可产生中断请求
- 高电平检测时间控制
- 分别支持工作在主机模式和从机模式：
  - I2C 主机功能
    - 产生通信时钟
    - 实现总线仲裁
    - 8 位的地址字，寻址 7 位地址从机、并控制数据读写方向
    - 通信状态监测
    - 产生起始和停止信号
  - I2C 从机功能
    - 可编程的 I2C 地址检测
    - 可响应一个 7 位地址
    - 通信状态监测
    - 起始信号和停止信号检测

14.3 I2C 功能描述

I2C 模块接收和发送数据、并将数据从串行转换成并行(从机模式下)，或并行转换成串行(主机模式下)。可以开启或禁止中断。

接口通过数据引脚(SDA)和时钟引脚(SCL)连接到 I2C 总线。允许连接到速率高达 1Mbps 的 I2C 总线。

图 14-1 I2C 的功能框图



14.3.1 I2C 通信

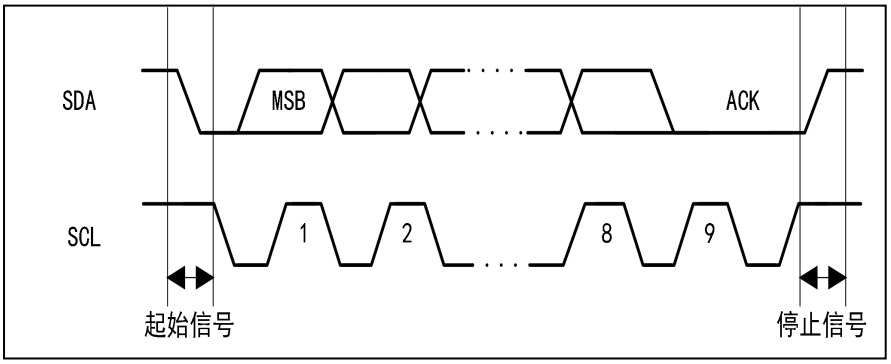
通信总是由主机发起。主机通过 SDA 脚将数据从最高有效位(MSB)开始发送给从机，从机则通过 SDA 脚回传数据，这意味半双工通信的数据输出和数据输入是用同一个时钟信号同步的；时钟信号则由主机通过 SCL 脚提供。

I2C 的通信总是以起始信号开始并以停止信号结束，下面是一次完整的 I2C 通信时，主机和从机所执行的步骤：

- 主机模式下：
  - 由软件控制产生起始信号
  - 发送起始信号后、主机发送 1 字节的地址字
  - 开始数据传输(主机发送或者接收若干个数据字)。
  - 由软件控制产生停止信号
- 从机模式下：
  - 接收主机发出的地址字并与自身地址作比较。
  - 在地址匹配的前提下，开始数据传输(从机接收或者发送)

一个 8 位的字节数据传输完成后、在第 9 个时钟期间，接收方应回送一个应答信号(ACK)给发送方。参考下图。

图 14-2 I2C 总线协议



软件可以开启或禁止应答信号(ACK)，也可以设置 I2C 接口的地址(7 位)。

14.3.1.1 I2C 波特率

I2C 通信波特率由 PCLK 经(I2C_CR)寄存器的"BR[9:0]"位分频而来：

$f_{SCL} = f_{PCLK}/((BR+1)*4)$

其中：

- f_{SCL} 为 I2C 通信的波特率
- f_{PCLK} 为 PCLK 时钟的频率

注意：

1.BR 存在最小值限制，BR 必须>5；BR<5 时，实际分频系数仍为 24。

2.I2C 波特率不应超过 1Mbps

14.3.1.2 空闲状态

当 SDA 和 SCL 两条信号线同时处于高电平时，规定为总线的空闲状态。
该状态下，没有任何主机或从机设备占用总线，读 SR 位恒为 0X1F；

注意：当整个 I2C 通信系统上存在多个主机/从机设备时，首次 I2C 通信前，应当保证所有设备均处于空闲状态，否则可能引发意外的状态；

14.3.1.3 起始信号(Start)

空闲状态下、主机通过发送起始信号以发起通信。
如(图 14-2)所示、起始信号定义为：当 SCL 在高电平时、SDA 从高到低的跳变。
该信号表示开始新的数据传输(每次数据传输可能包含几个字节的数据)，并使所有从机退出空闲状态。
通过将(I2C_CR)的 STAR 位置'1'以产生一次起始信号；
主机起始信号发送完毕，其 SR 位被硬件设置为 0x01。

14.3.1.4 应答信号(ACK)

一个 8 位的字节数据传输完成后、在第 9 个时钟期间，接收方应回送一个应答信号(ACK)给发送方，(如图 14-2)。
如果接收方无应答(NACK)，则按照主机和从机有：

- 主机发送数据(地址字或数据字)，从机返回 NACK，主机将判定为失败的数据传输；
- 从机发送数据，主机返回 NACK，从机将判定数据传输结束，并释放总线给主机。

对于以上两种情况，数据传输都被中止，主机会进行以下两种操作之一：

- 发送停止位、终止数据传输、并释放总线
- 发送重复起始位、建立新的通信

通过将(I2C_CR)的 ACK 位置'1'以使能应答信号(ACK)，接收方的应答状态决定了 I2C 在通信时 SR 位的一些状态值和中断请求类型，详见 14.3.4 状态信息和中断请求。

14.3.1.5 地址字(ADDR)与方向

主机在发送完起始信号之后、首先发送一个 1 字节的地址字。
地址字共 8 位，高 7 位是从机的地址码，最低位为"R/W"，决定总线上数据传输的方向：

- 1 = 读取传输， 从机向主机传输数据
- 0 = 写入传输， 主机向从机传输数据

从机的地址可以通过寄存器 I2C_OAR 的 ADDR 位配置；
当主机发送的地址与从机自身地址匹配时，从机寻址完毕，回送 ACK 响应之后有：

- 从机的 SR 位被硬件设置为 0x15
- 主机收到 ACK 后，SR 位被硬件设置为 0x03

14.3.1.6 数据字(DATA)

实现从机寻址后，寻址有效的从机就可以按照主机发送的“R/W”位指定的方向与主机逐字节地进行数据字传输；
通过主机或从机的发送器、(I2C_DR)寄存器中的字节经由内部的移位寄存器被发送到 SDA 线上。

14.3.1.7 停止信号(Stop)

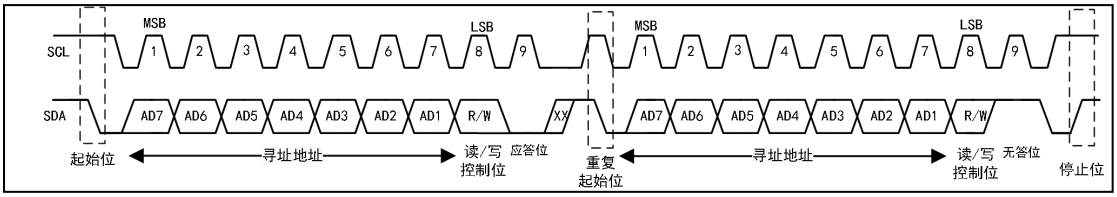
任何时候、主机都可以发送停止信号，以将总线恢复至空闲状态。
特别地、主机还可以直接发送起始位或者广播，而无需首先发送停止位，这被称为重复起始信号。
停止信号的定义是 SCL 在逻辑 1 时的从低到高的 SDA 电平跳变(见图 14-2)。
通过将(I2C_CR)的 STOP 位置’1’以产生一次停止信号
停止信号在 I2C 通信的不同阶段(地址字、数据字或广播)会产生相应的状态，详见 14.3.4 状态信息和中断请求。

14.3.1.8 重复起始信号

在无需终止通信，又需要从机释放总线的情况下，可以再发送一个起始信号(Start)以产生一个重复起始信号，该信号适用于以下场合：

- 主机需要与另外一个从机进行通信
- 使用不同的传输方向(R/W)与同一从机进行通信

图 14-3 重复起始信号



14.3.1.9 广播

广播功能包括特定的广播地址字”0x00”与广播数据字，支持主机用来与 IIC 总线上所有的从机设备进行通信(仅主机作为发送方)；
主机将地址字设置为广播地址”0x00”后、广播功能被启用；从机将(I2C_OAR)寄存器中的 BC 位置’1’以使能广播响应；BC 位置’1’后，当从机接收到广播、并回送 ACK 信号之后有：

- 从机的 SR 位被硬件设置为 0x0E
- 主机收到 ACK 后，SR 位被硬件设置为 0x03

当主机使用广播功能、后续的数据字传输也被视作广播数据字，I2C 总线上所有开启广播响应的从机均可接收广播数据字，接收到广播数据字的从机有：

- 回送 ACK，从机的 SR 位被硬件设置为 0x12，主机的 SR 位则为 0x05
- 回送 NACK，从机的 SR 位被硬件设置为 0x13，主机的 SR 位则为 0x06

当需要返回到非广播的 1 对 1 通信时，可由主机发送一个停止信号或者重复起始信号。

14.3.1.10 总线仲裁(SDA 仲裁)

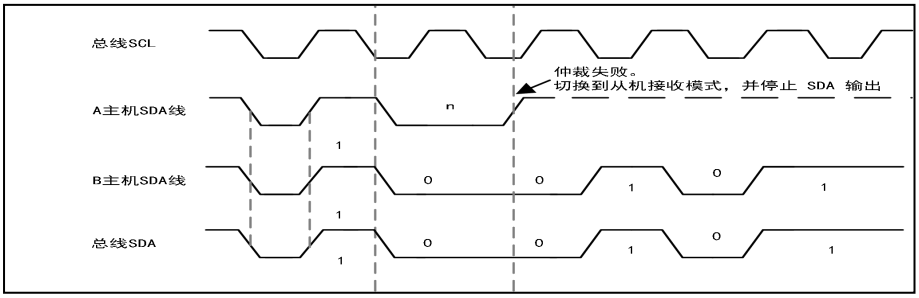
发送器在发送信号的同时也会检测自身发出的信号；在 SDA 线上、低电平总是比高电平具有更高的优先级，它们具有"与"的关系，这是内部总线仲裁的基础。

当两个、甚至多个主机试图同时控制 SDA 总线时，鉴于内部总线仲裁的基础有:

- 当 A 主机的 SDA 为逻辑 1，而 B 主机的 SDA 也为逻辑 1，此时总线仲裁机制失效，总线上的 SDA 为逻辑 1。
- 当 A 主机的 SDA 为逻辑 1，而 B 主机的 SDA 却为逻辑 0，此时总线仲裁机制生效，总线上的 SDA 为逻辑 0，A 主机被仲裁为失败。

当主机被仲裁失败时，将立即切换到从机接收模式，并停止 SDA 输出。这种情况下，从主模式到从模式的转换将不会生成停止条件，与此同时，仲裁失败的主机、其状态寄存器(I2C_SR)的 SR 位被硬件设置为 0x07。

图 14-4 总线仲裁



14.3.1.11 时钟同步(SCL 同步)

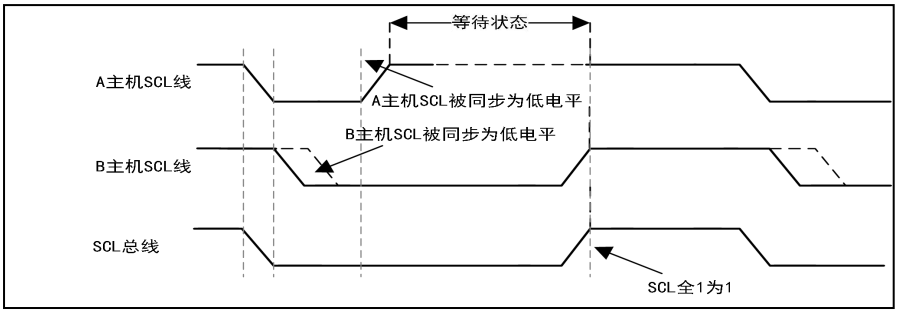
时钟信号发生器在发送时钟信号的同时也会检测自身发出的信号；在 SCL 线上、低电平总是比高电平具有更高的优先级，它们具有"与"的关系，这是内部时钟同步的基础。数据仅在时钟的高电平期间有效，所以当两个、甚至多个主机试图同时控制 SCL 总线时，鉴于内部时钟同步的基础有：

- 在某一时刻、当 A 主机的 SCL 为逻辑 1，而 B 主机的 SCL 也为逻辑 1，此时时钟同步机制失效，总线上的 SCL 为逻辑 1。
- 在某一时刻、当 A 主机的 SCL 为逻辑 1，而 B 主机的 SCL 却为逻辑 0，此时时钟同步机制生效，总线上的 SCL 为逻辑 0，A 主机的 SCL 被时钟同步机制强拉为逻辑 0。

鉴于上述流程、可以确定时钟同步机制有：

- SCL 线的低电平周期由具有"最长低电平周期"的主机决定
- SCL 线的高电平周期由具有"最短高电平周期"的主机决定

图 14-5 时钟同步



14.3.2 从机模式

默认情况下，I2C 接口总是工作在从模式。将从模式切换到主模式，需要产生一个起始信号。

一旦检测到起始条件，在 SDA 线上接收到的地址被送到内部移位寄存器。然后与从机自己的地址”ADDR”或者广播地址”0x00”(当 GC 位=1)相比较。

当地址不匹配时：

- I2C 接口将该地址忽略
- I2C 接口等待另一个起始条件

当地址匹配时：

- 如果 ACK 被置’1’，则回送 ACK，SR 位被硬件设置为”0x0C”
- 如果在 NVIC 中使能 I2C 中断，则产生相应的中断请求

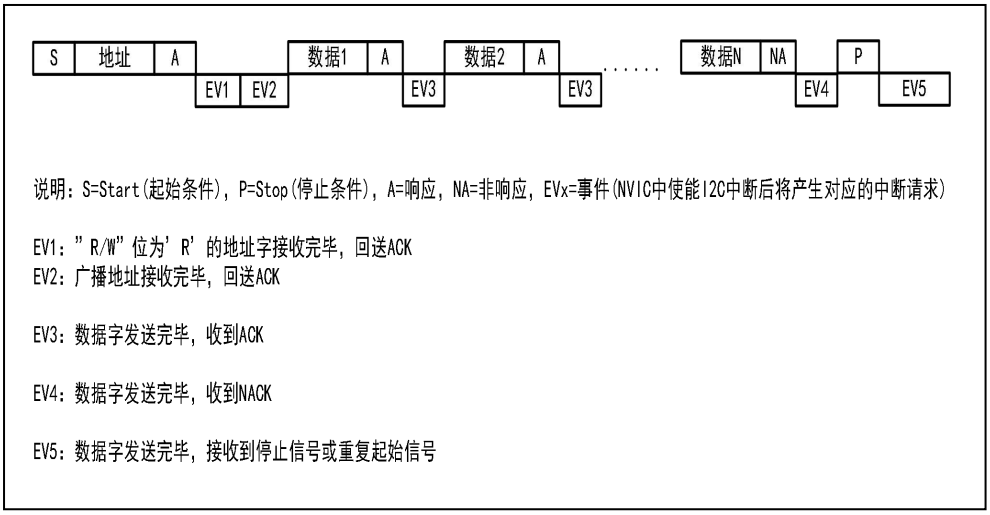
14.3.2.1 从发送器

实现从机寻址后，当从机接收到的“R/W”位为 1，在回送 ACK 后，SCL 线被从机置为逻辑’0’电平、接着从机进入发送器模式。

通过从发送器、(I2C_DR)寄存器中的字节经由内部的移位寄存器被发送到 SDA 线上、SCL 线被释放，从发送器数据发送完成后有：

- 主机回送 ACK,则从机的 SR 位被硬件设置为”0x17”
- 主机回送 NACK,则从机的 SR 位被硬件设置为”0x18”
- 如果在 NVIC 中使能 I2C 中断，则产生相应的中断请求

图 14-6 从发送器的传送序列



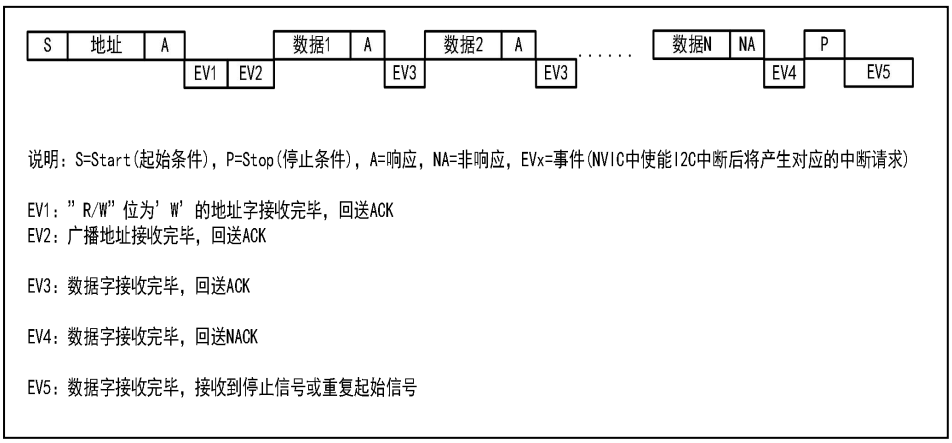
14.3.2.2 从接收器

实现从机寻址后，当从机接收到的“R/W”位为 0，在回送 ACK 后、从机进入接收器模式。

通过从接收器、SDA 线上的字节经由内部的移位寄存器被同步到(I2C_DR)寄存器中、从接收器数据接收完毕后有：

- 如果 ACK 被置’1’，则回送 ACK，SR 位被硬件设置为”0x10”
- 如果 ACK 被置’0’，则回送 NACK，SR 位被硬件设置为”0x11”
- 如果在 NVIC 中使能 I2C 中断，则产生相应的中断请求

图 14-7 从接收器的传送序列



14.3.2.3 通信终止

在传输完最后一个数据字节后，主机产生一个停止信号，从机检测到该信号时：

- SR 位被硬件设置为”0x19”
- 如果在 NVIC 中使能 I2C 中断，则产生相应的中断请求

14.3.3 主机模式

在主机模式时，I2C 接口启动数据传输并产生时钟信号。

I2C 通信总是以起始信号开始并以停止信号结束。当通过 START 位在总线上产生了起始条件，设备就进入了主模式，以下是主模式所要求的操作顺序：

- 在 I2C_CR 寄存器中设定 I2C 波特率分频系数”BR”以产生正确的时序
- 将 I2C_CR 寄存器中的”PE”位置’1’，启动 I2C 外设
- 将 I2C_CR 寄存器中的”STAR”位置’1’，产生起始信号
- 往 I2C_DR 寄存器中的”DR”位写入第一个 8 位数据，该数据为地址字。
- 继续往 I2C_DR 寄存器中的”DR”写入数据，这些数据为数据字。

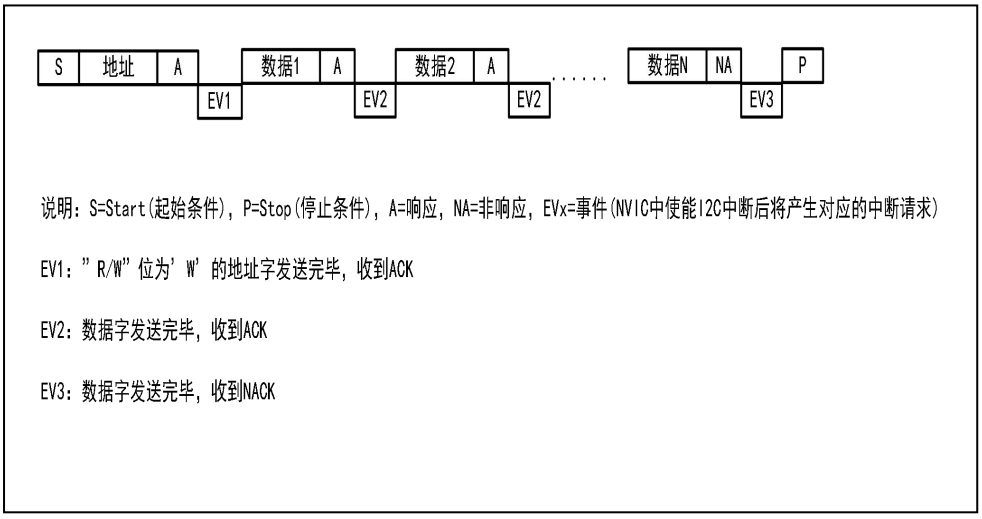
14.3.3.1 主发送器

主机发送完地址字后，当地址字的“R/W”位为 1，在接收到从机回送 ACK 后，SCL 线被主机置为逻辑’0’电平、接着主机进入发送器模式。

通过主发送器、(I2C_DR)寄存器中的字节经由内部的移位寄存器被发送到 SDA 线上、SCL 线被释放，主发送器数据发送完成后有：

- 从机回送 ACK,则从机的 SR 位被硬件设置为”0x05”
- 从机回送 NACK,则从机的 SR 位被硬件设置为”0x06”
- 如果在 NVIC 中使能 I2C 中断，则产生相应的中断请求

图 14-8 主发送器的传送序列



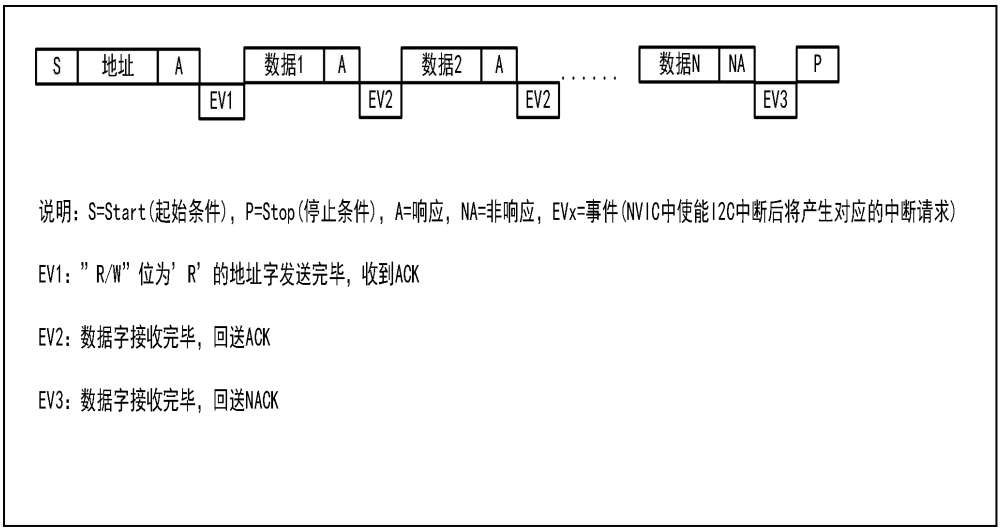
14.3.3.2 主接收器

主机发送完地址字后，当地址字的“R/W”位为 0，在接收到从机回送 ACK 后、主机进入接收器模式。

通过主接收器、SDA 线上的字节经由内部的移位寄存器被同步到(I2C_DR)寄存器中、主接收器数据接收完毕后有：

- 如果 ACK 被置'1'，则回送 ACK，SR 位被硬件设置为"0x0A"
- 如果 ACK 被置'0'，则回送 NACK，SR 位被硬件设置为"0x0B"
- 如果在 NVIC 中使能 I2C 中断，则产生相应的中断请求

图 14-9 主接收器的传送序列



14.3.3.3 通信终止

在 DR 寄存器中写入最后一个数据字后，通过设置 STOP 位产生一个停止信号(见图 14-8，14-9 的'P')，然后 I2C 接口将自动回到从模式。

14.3.4 状态信息和中断请求

I2C 模块支持 27 个状态，除总线空闲状态以外、每种状态均默认支持中断请求，(I2C_CR)寄存器的”SI”为这些状态的总标志位，清’0’以清除响应的状态或中断标志；状态寄存器(I2C_SR)的”SR[4:0]”位体现了所有支持的状态，参见表 14-1

表 14-1 I2C 通信状态信息

SR	I2C 总线状态	R/W 状态
0x00	主机或从机，在地址字或数据字正常传输期间 I2C 总线上出现一个停止信号	
0x01	主机，起始信号发送完毕	
0x02	主机，重复起始信号发送完毕	
0x03	主机，地址字发送完毕，收到 ACK	W
0x04	主机，地址字发送完毕，收到 NACK	W
0x05	主机，数据字发送完毕，收到 ACK	
0x06	主机，数据字发送完毕，收到 NACK	
0x07	主机，发送地址字或数据字时总线仲裁失败	
0x08	主机，地址字发送完毕，收到 ACK	R
0x09	主机，地址字发送完毕，收到 NACK	R
0x0A	主机，数据字接收完毕，回送 ACK	
0x0B	主机，数据字接收完毕，回送 NACK	
0x0C	从机，地址字接收完毕，回送 ACK	W
0x0D	从机，主机总线仲裁失败转化的从机 地址字接收完毕，回送 ACK	W
0x0E	从机，广播地址接收完毕，回送 ACK	
0x0F	从机，主机总线仲裁失败转化的从机 广播地址接收完毕，回送 ACK	
0x10	从机，数据字接收完毕，回送 ACK	
0x11	从机，数据字接收完毕，回送 NACK	
0x12	从机，广播模式下、数据字接收完毕，回送 ACK	
0x13	从机，广播模式下、数据字接收完毕，回送 NACK	
0x14	从机，数据字接收完毕 接收到停止信号或重复起始信号	
0x15	从机，地址字接收完毕，回送 ACK	R
0x16	从机，主机总线仲裁失败转化的从机 地址字接收完毕，回送 ACK	R
0x17	从机，数据字发送完毕，收到 ACK	
0x18	从机，数据字发送完毕，收到 NACK	
0x19	从机，数据字接收完毕，回送 ACK 后、 接收到停止信号或重复起始信号	
0x1F	总线空闲	

14.4 寄存器描述

14.4.1 I2C 控制寄存器 I2C_CR

（地址：0x4000_5400）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	BR[9:0]									
R/W	Res	Res	Res	Res	Res	Res	RW1	RW1	RW1	RW1	RW1	RW1	RW1	RW1	RW1	RW1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	HCT[3:0]				-	-	-	-	-	PE	STAR	STOP	SI	ACK	-	-
R/W	RW1	RW1	RW1	RW1	Res	Res	Res	Res	Res	RW1	RW1	RW1	RW1	RW1	Res	Res
复位	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 1：0	保留。必须保持为 0。
位 2	**ACK**：应答使能**(Acknowledge enable)** 软件可以设置该位，在 **I2C_CCR** 中清除该位 0：　　无应答返回 1：　　在接收到一个字节后返回一个应答(匹配的地址或数据) ***注：I2C 使能后，对于该位的修改，仅在 SI 为 1 时有效***
位 3	**SI**：I2C 状态总标志位**(I2C status flag)** 0：　　I2C 总线空闲或旧的状态被清除 1：　　I2C 总线有状态更新，写 1 以清除状态标志 ***注：如果在 NVIC 中使能 I2C 中断，该位将作为 I2C 中断总标志位。***
位 4	**STOP**：I2C 发送停止信号**(Stop generation)** 软件可以设置该位；当停止信号发送完毕、该位由硬件清除； 0：　　无停止条件产生； 1：　　产生一个停止信号
位 5	**STAR**：I2C 发送起始信号**(Start generation)** 软件可以设置该位，在 I2C_CCR 中清除该位 0:　　无起始信号产生； 1：　　产生起始信号。
位 6	**PE**：I2C 模块使能**(I2C enable)** 软件可以设置该位，在 **I2C_CCR** 中清除该位 0：　　禁用 I2C 模块； 1：　　启用 I2C 模块：相应的 I/O 口需配置为复用功能。 ***注：　为避免当前传输的数据被破坏、在通信结束之前，不应清除该位。***
位 15：7	保留。必须保持为 0。
位 25：16	**BR[9:0]**：I2C 波特率分频系数**(Baud rate control)**

	软件可以设置该位，在 I2C_CCR 中清除该位 详见 14.3.1.1 I2C 波特率章节介绍 ***注：BR 存在最小值限制，BR 必须>5；BR<5 时，实际分频系数仍为 24。***
位 31：26	保留。必须保持为 0。

14.4.2 I2C 状态寄存器 I2C_SR

（地址：0x4000_5404）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	SR[4:0]					-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	R	R	R	R	R	Res	Res	Res
复位	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0

位 2：0	保留。必须保持为 0。
位 7：3	**SR[4:0]：I2C 总线状态(I2C bus status)** 支持 27 种状态，除总线空闲外、所有状态均可产生中断请求，详见 14.3.4 状态信息和中断请求
位 31：8	保留。必须保持为 0。

14.4.3 I2C 数据寄存器 I2C_DR

（地址：0x4000_5408）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	DR[7:0]							
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 7：0	**DR[7:0]：8 位数据寄存器(I2C data)** 保存 I2C 总线发送或接收的数据 ***注：在从模式下，地址字不会被拷贝到 DR 中，而是直接与 ADDR 做对比；*** ***注：新数据字被同步到 DR 前，如果旧数据字仍未被取读，则旧数据字将被新数据字覆盖***
位 31：8	保留。必须保持为 0。

14.4.4 I2C 地址寄存器 I2C_OAR

（地址：0x4000_540C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	ADDR[6:0]							BC
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**BC**：广播响应**(Broadcast response control)** 0：　　禁止从机响应广播，以 NACK 响应广播 1：　　使能从机响应广播，以 ACK 响应广播
位 7：1	**ADDR[6:0]**：从机地址**(I2C addr)** 地址的 7~1 位。
位 31：8	保留。必须保持为 0。

14.4.5 I2C 控制清除寄存器 I2C_CCR

（地址：0x4000_5418）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	BR[9:0]									
R/W	Res	Res	Res	Res	Res	Res	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	HCT[3:0]				-	-	-	-	-	PE	STAR	STOP	SI	ACK	-	-
R/W	W	W	W	W	Res	Res	Res	Res	Res	w	w	w	w	w	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 1：0	保留。必须保持为 0。
位 2	ACK：应答位控制(Clear Acknowledge control) 0：　无效 1：　ACK 位清 0
位 3	**SI：I2C 中断标志位(Clear I2C interrupt flag)** 0：　无效 1：　SI 标志清 0
位 4	**STOP：I2C 发送停止位(Clear stop generation)** 0：　无效 1：　STOP 清 0
位 5	**STAR：I2C 发送起始位(Clear start generation)** 0：　无效 1：　STA 清 0
位 6	PE：I2C 模块使能控制(I2C disable) 0：　无效 1：　EN 清 0
位 11：7	保留。必须保持为 0。
位 15：12	HCT[3:0]：高电平检测时间系数(Clear high level detection time) 0：　无效 1：　HCT 对应位清 0
位 25：16	BR[9:0]：I2C 波特率分频系数清除(Clear baud rate) 0：　无效 1：　BR 对应位清 0
位 31：26	保留。必须保持为 0。

14.4.6 寄存器列表

地址	寄存器	描述	备注
0x4000_5400	I2C_CR	I2C控制寄存器	I2C_CR说明
0x4000_5404	I2C_SR	I2C状态寄存器	I2C_SR说明
0x4000_5408	I2C_DR	I2C数据寄存器	I2C_DR说明
0x4000_540C	I2C_OAR	I2C地址寄存器	I2C_OAR说明
0x4000_5418	I2C_CCR	I2C控制清除寄存器	I2C_CCR说明

15 通用异步收发器(UART)

15.1 综述

通用异步收发器(UART)提供了一种灵活的方法与使用工业标准的异步串行数据格式(RS232、RS485 等)的外部设备之间进行全双工的数据交换。
UART 利用分数波特率发生器提供宽范围的波特率选择。
它支持单线半双工通信，也支持 IrDA(红外数据组织)规范。

15.2 特性

- 全双工的异步通信
  - 发送/接收极性控制
- 支持最大 3Mbps 的可编程通信速率
- 可编程数据帧长度(7 位至 9 位)
- 可配置的停止位，支持 0.5~2 个停止位
- 单工仅发送的红外串行协议调制器
- 单线的半双工通信
- 最大 4 级的接收/发送 FIFO 缓冲队列
- 可触发中断的通信状态检测标志
  - 接收 FIFO 非空
  - 接收 FIFO 全满
  - 发送 FIFO 全空
  - 发送 FIFO 全满
  - 发送数据完毕
- 校验控制
  - 发送校验位
  - 对接收数据进行校验
- 可触发中断的错误检测标志
  - 校验错误
  - 帧错误
  - 接收 FIFO 溢出

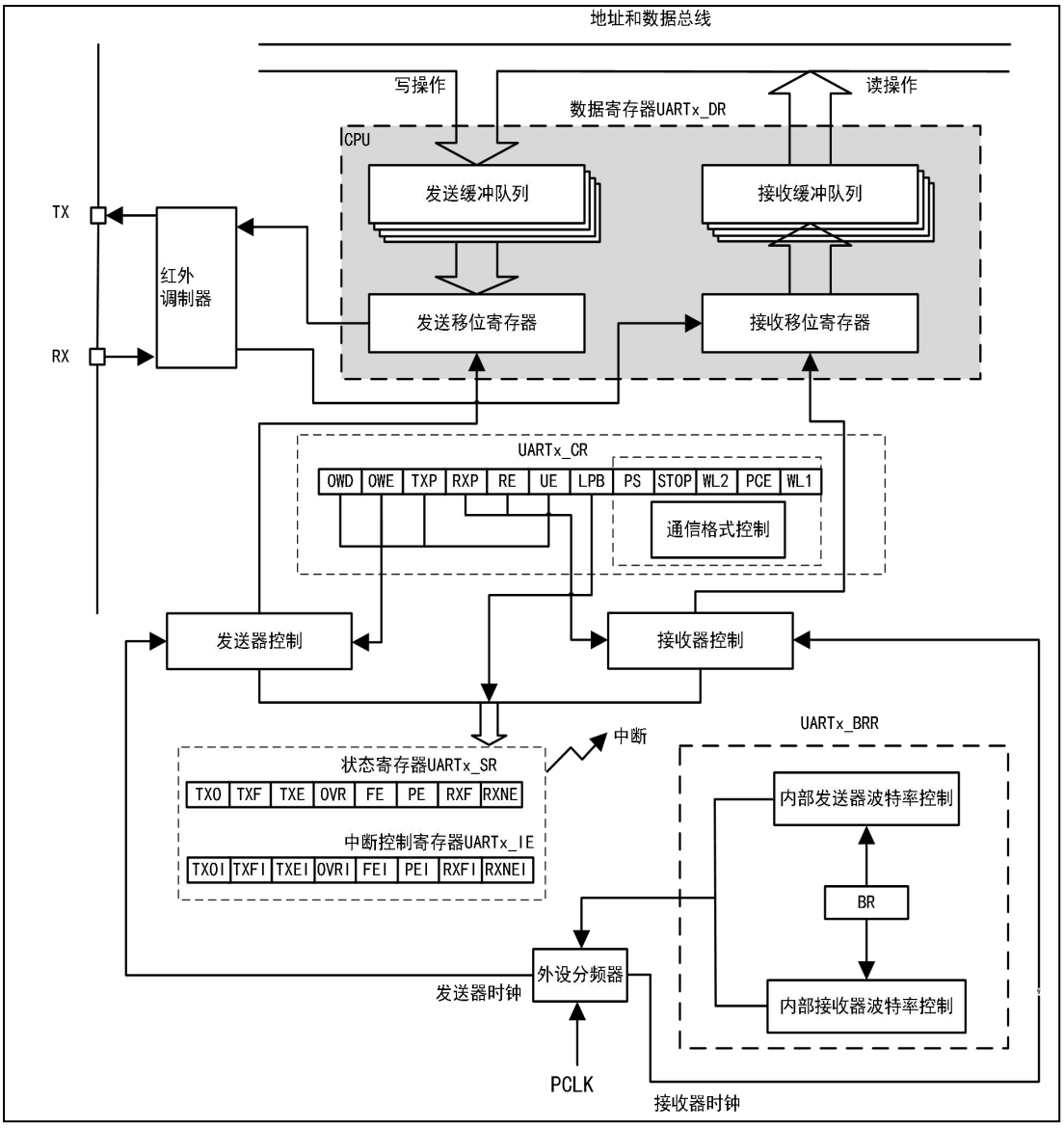
15.3 UART 功能描述

任何 UART 双向通信至少需要两个脚：接收数据输入(RX)和发送数据输出(TX)。

RX：接收数据串行输入。通过"过采样"技术来区别数据和噪音，从而恢复数据。

TX：发送数据输出。当发送器被激活、并且不发送数据时，TX 引脚处于高电平。在单线模式里，此 I/O 口被同时用于数据的发送和接收。

图 15-1 UART 框图



15.3.1 UART 通讯

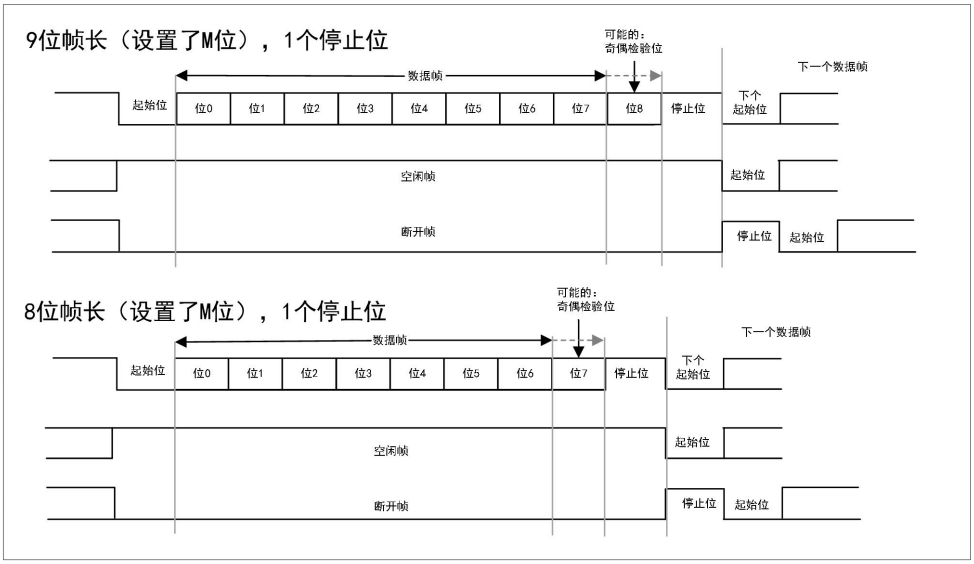
UART 是异步的串行数据收发器，它是没有时钟线的，两个设备通讯时，A 设备的 RX 接 B 设备的 TX，B 设备的 RX 接 A 设备的 TX，对于每个设备而言、发送和接收的数据分别在独立的两条线上传输，是一种异步全双工的通信方式。

对于两个 UART 设备之间什么时候开始传输、数据格式是什么、又是什么时候传输结束，UART 都有着严格的规范。

UART 的通信总是以起始位开始并以停止位结束的、下面是一次完整的 UART 通信时，UART 模块所执行的步骤：

- 通信线(RX/TX)处于空闲状态
- TX 线产生一个起始位，RX 线接收到一个起始位
- TX 线发送一帧数据帧，RX 线接收数据帧
- TX 线发送一个校验位，RX 线接收检验位
- TX 线发送一个停止位，RX 线接收停止位
- 通信线(RX/TX)处于空闲状态，直到下一个起始位来临

图 15-2 UART 帧长设置



15.3.1.1 UART 波特率

UART 通信波特率由 PCLK 经(UART_BRR)寄存器的"BR[15:0]"位分频而来：

$$f_{BPS} = \left(\frac{(f_{PCLK} / BR)}{16} \right)$$

其中：

- f_{BPS} 为 UART 通信的波特率
- f_{PCLK} 为 PCLK 时钟的频率

注意：UART 波特率不应超过 3Mbps

15.3.1.2 标准波特率误差

常用的标准串口通信波特率有 9600,38400,115200 等，这里的误差，指 UART 波特率与标准波特率的差值，它用百分比表示，设标准波特率为 S_{BPS}，则波特率误差为：

$$err_{BPS}\% = \left(\frac{(f_{BPS} - S_{BPS})}{S_{BPS}} \right) * 100$$

表 15-1 设置波特率时的误差计算

波特率		f_{PCLK}=24Mhz	
序号	bps	BR[15:0]值	误差
1	115200	0x0D	0.16%
2	38400	0x27	0.16%
3	28800	0x34	0.16%
4	19200	0x4E	0.16%
5	9600	0x9C	0.16%
6	4800	0x139	0.16%
7	2400	0x271	0
8	1200	0x4E2	0

注意：

1. 标准波特率误差不应超过 1%

2. 实际中，如需保证高可靠，高稳定的串口通信，还应考虑 PCLK 时钟源的频率误差，与实际线路干扰等因素。

15.3.1.3 空闲状态

当 RX 线或 TX 线处于高电平时，规定为该通信线处于空闲状态。

该状态下，没有任何设备占用通信线；

15.3.1.4 起始位(Start)

空闲状态下、TX 线通过发送一个起始位以发起一次通信。

如(图 15-2)所示、起始信号定义为：当通信线在高电平时、线上有一个从高到低的跳变。

该信号表示开始新的数据传输(每一个数据帧前必须有一个起始位)，并使 RX 退出空闲状态。

15.3.1.5 数据帧(DATA)

每一帧数据的输出， 总是以最低有效位(LSB)开始。

根据(UART_CR)寄存器的"WP[2:0]"位， 每个数据帧可以是 7 位到 9 位。 所选择的数据帧格式对发送和接收均生效。

15.3.1.6 校验位

如图 15-2，校验位是紧跟在数据帧的最后一个位发出的(如果为数据帧长为 9，最高位就是第 9 位；如果数据帧长为 8，最高位就是第 8 位)。

通过(UART_CR)寄存器的"WP[2:0]"位以使能校验位，使能校验位后，对于传输双方：

- 发送方将产生校验位
- 接收方将进行校验位的检测

通过"PS"位选择校验方式，可选的两种校验方式有：

- 偶校验：当数据帧中、二进制'1'的个数不为偶数时，校验位置'1'以满足偶数条件。
  - 例：数据帧 00110101，有 4 个'1'，为偶数，则校验位为'0';
  - 例：数据帧 00110100，有 3 个'1'，为奇数，则校验位为'1';
- 奇校验：当数据帧中、二进制'1'的个数不为奇数时，校验位置'1'以满足奇数条件。
  - 例：数据帧 00110101，有 4 个'1'，为偶数，则校验位为'1';
  - 例：数据帧 00110100，有 3 个'1'，为奇数，则校验位为'0';

15.3.1.7 停止位(Stop)

在数据帧或检验位发送完毕之后、TX 线通过发送一个停止位以结束一次通信。

如(图 15-2)所示、该信号表示结束一次数据传输(每帧数据后或每一个校验位后"当使能了校验位"，必须有一个停止位)，它将使通信线返回空闲状态。

15.3.1.8 缓冲队列(FIFO)

UART 的发送和接收均集成了一个 4 级的先入先出（FIFO）缓冲队列，这个缓冲队列中，每一级缓冲有最高 9 位的数据有效位，它们数据格式都是左对齐的，并且与 UART_DR 寄存器实时同步；

每帧数据不论大小(7 位~9 位)，每一帧数据均占用一级缓冲，缓冲队列在发送和接收时具体的表现为：

- 在发送时:
  - 发送的数据以'帧'为单位、首先存放到缓冲队列当中；
  - 每往(UART_DR)寄存器写入一帧数据、缓冲队列都会被更新，直到存满 4 帧数据，UART 将产生 TXF(发送 FIFO 全满)。
  - 缓冲队列最多缓冲 4 帧数据，发送缓冲全满时、往(UART_DR)寄存器继续写入的帧数据将被丢弃。

- 在接收时：
  - 接收到的数据以'帧'为单位、通过内部移位寄存器首先存放在缓冲当中；
  - 当读取(UART_DR)寄存器时，先收到的数据会被先同步到(UART_DR)寄存器中，直到缓冲为空，UART_DR 的值将不会被更新，这意味着此时读到的数据为最后一帧数据；
  - 缓冲队列最多缓冲 4 帧数据，如果没有及时读取 UART_DR 中的数据，则当第 5 帧数据来临时，该数据帧被丢弃。

15.3.2 启用 UART 功能

15.3.2.1 配置步骤

- 通过(UART_BRR)寄存器的”BR[15:0]”位定义 UART 的波特率。
- 可选的、通过(UART_CR)寄存器的”RXP”和”TXP”位以配置 TX 线和 RX 线的极性。
- 通过(UART_CR)寄存器的”WP[2:0]”位配置数据帧的格式
- 可选的、通过(UART_CR)寄存器的”WP[2:0]”位使能校验位，并通过”PS”位选择校验方式
- 通过(UART_CR)寄存器的”STOP”位以选择停止位的长度
- 可选的、将(UART_CR)寄存器的”RE”位置’1’，以使能 UART 接收功能
- 将(UART_CR)寄存器的”UE”位置’1’，以使能 UART

15.3.2.2 发送器

当写入数据至(UART_DR)时，数据首先被同步到发送缓冲队列，接着发送过程开始。在发送第一个数据位时，数据被并行地(通过内部总线)传入到内部移位寄存器，而后串行地移出到 TX 脚上；

当缓冲队列中的所有数据都被发送完毕、TXO 标志将被硬件置位，如果设置了(UART_IE)寄存器中的 TXOI 位，将产生 TXO 中断；

15.3.2.3 接收器

对于接收器，当数据帧传输完成时：

- 内部移位寄存器里的数据被同步到接收缓冲队列，RXNE 标志被硬件置’1’。当缓冲队列中数据为空时，硬件将自动清除 RXNE 位。
- 当缓冲队列中接收到第 4 帧数据后、RXF 标志将被硬件置位，如果设置了(UART_IE)寄存器中的 RXFI 位，则产生中断。

15.3.3 状态和中断

通过下述的状态，应用程序得以完全监控 UART，UART 所有的状态均支持中断请求，将(UARTx_IE)寄存器中相应状态的中断使能、在状态产生之后，将同步向 NVIC 发出一个中断请求；当 NVIC 中的 UARTx 中断被使能，系统将处理这些中断请求。

15.3.3.1 接收 FIFO 非空(RXNE)

此标志为'1'时、表明在接收缓冲队列中包含至少一帧的有效数据。当使能了 RXNE 中断"RXNEI"位后，将同步产生一个中断请求到 NVIC；
读(UART_DR)寄存器、直到接收 FIFO 为空，硬件自动清除此标志，相应的中断请求被同步清除。

15.3.3.2 接收 FIFO 全满(RXF)

此标志为'1'时、表明在接收缓冲队列中存在 4 帧的有效数据，并且从未被取读。当使能了 RXF 中断"RXFI"位后，将同步产生一个中断请求到 NVIC；
读(UART_DR)寄存器、直到接收 FIFO 中的有效帧数据<4，硬件自动清除此标志，相应的中断请求被同步清除。

15.3.3.3 发送 FIFO 为空(TXE)

此标志为'1'时、表示发送缓冲队列为空，当使能了 TXE 中断"TXEI"位后，将同步产生一个中断请求到 NVIC；
当写入一帧数据到(UART_DR)寄存器时，TXE 标志被硬件自动清除。

15.3.3.4 发送 FIFO 全满(TXF)

此标志为'1'时、表明在发送缓冲队列中存在 4 帧的有效数据，一旦此时再往(UART_DR)中写一帧新的数据，那么该帧数据将被丢弃。当使能了 TXF 中断"TXFI"位后，将同步产生一个中断请求到 NVIC；
参考 15.3.2.1 配置步骤使能 UART 以开始发送数据帧、直到发送 FIFO 中的有效帧数据<4，硬件自动清除此标志，相应的中断请求被同步清除。

15.3.3.5 发送数据完毕(TXO)

此标志为'1'时、表明此前在发送缓冲队列中存在的所有有效数据被发送完毕。当使能了 TXO 中断"TXOI"位后，将同步产生一个中断请求到 NVIC；
当写入一帧数据到(UART_DR)寄存器时，TXO 标志被硬件自动清除。

15.3.4 错误标志与中断

通过下述错误标志，应用程序可以管理 UART 通信时任何错误的情况。

注意：FIFO 中存储了一帧数据的所有信息，当 FIFO 中某帧存在错误时(校验错误或者帧错误)，仅当程序读取到该帧信息时，相应的错误标志才会置位。

15.3.4.1 校验错误(PE)

此标志为'1'时、表明模块使能了校验位，并且在当前这一次的通信中、检测到校验错误。当使能了 PE 中断"PEI"位后，将同步产生一个中断请求到 NVIC；

根据校验位的定义有：

- 偶校验错误：当数据帧中、二进制'1'的个数不为偶数时，校验位为二进制'0'。
  - 例：数据帧 00110101，有 4 个'1'，为偶数，但校验位却为'1';
  - 例：数据帧 00110100，有 3 个'1'，为奇数，但校验位却为'0';
- 奇校验错误：当数据帧中、二进制'1'的个数不为奇数时，校验位为二进制'0'。
  - 例：数据帧 00110101，有 4 个'1'，为偶数，但校验位却为'0';
  - 例：数据帧 00110100，有 3 个'1'，为奇数，但校验位却为'1';

将"PE"写 1 可以清除此标志，相应的中断请求被同步清除。

15.3.4.2 帧错误(FE)

此标志为'1'时、表明在当前这一次的通信中、检测到帧错误。当使能了 FE 中断"FEI"位后，将同步产生一个中断请求到 NVIC；

由于通信线上存在大量噪声或时序没有及时同步，造成的停止位没有在预期的时间上接和收识别出来，这被称为帧错误。

将"FE"写 1 可以清除此标志，相应的中断请求被同步清除。

15.3.4.3 溢出错误(OVR)

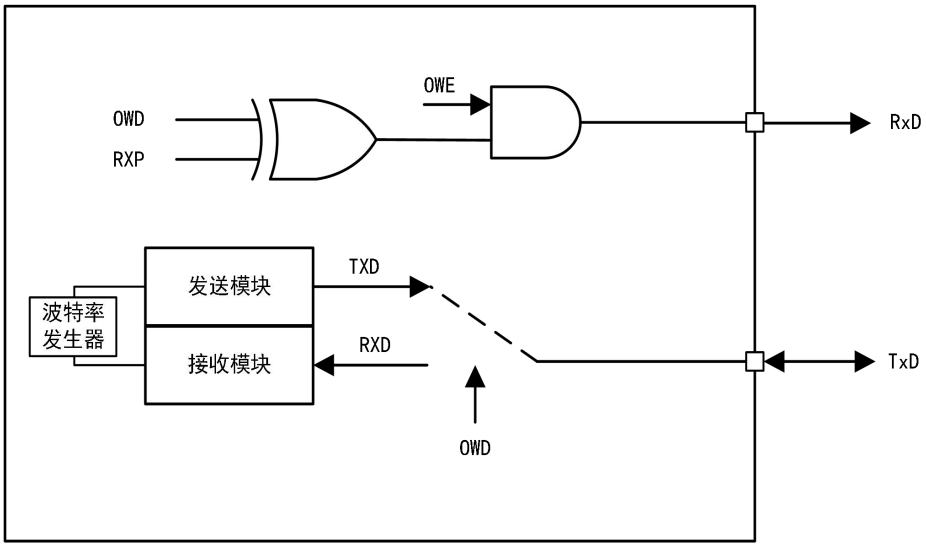
此标志为'1'时、表明在接收缓冲队列中存在 4 帧的有效数据从未被取读，并紧接着收到了第 5 帧数据，导致接收缓冲队列溢出，第 5 帧数据被丢弃。当使能了 OVR 中断"OVRI"位后，将同步产生一个中断请求到 NVIC；

"OVR"写 1 可以清除此标志，相应的中断请求被同步清除。

15.3.5 单线半双工通信

UART 可以配置为单线半双工通信。在单线半双工模式下，发送器和接收器在芯片内部通过(UART_CR)寄存器的”OWD”位连接到 TX 线上，RX 线用于输出 TX 线上数据的方向，如下图所示。

图 15-3 单线半双工模式框图



使用(UART_CR)寄存器的”OWE”位以选择 UART 以单线半双工或双线全双工通模式工作。当”OWE”位为’1’时，通过”OWD”位以切换 TX 线上数据的传输方向，RX 线如果被复用、则被用于指示当前数据传输方向，其输出状态见下表。

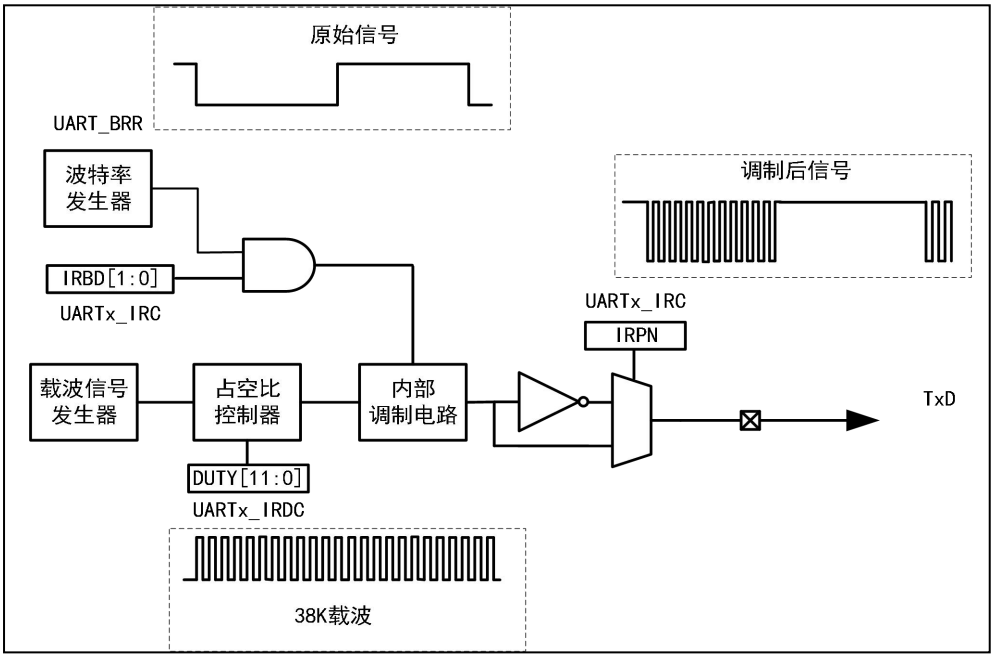
表 15-2 单线半双工下、RX 真值表

OWE	OWD	RXP	RX 输出	TX 引脚状态
1	0	0	0	数据接收
1	1	0	1	数据发送
1	0	1	1	数据接收
1	1	1	0	数据发送

15.3.6 红外通信功能

UART 可以配置为单线单工只发送的红外通信模式。在红外通信下，数据经过内部固定的 38Khz 载波信号的调制后，经由 TxD 输出到红外发射管，如下图所示。

图 15-4 红外通信框图



注意：红外通信的波特率同样由(UART_BRR)产生，但仅支持 600bps\1200bps\2400bps。

15.3.6.1 红外调制信号的占空比

38K 的载波信号由 PCLK 内部分频而来，红外调制占空比控制寄存器(UARTx_IRDC)的”DUTY[11:0]”位则决定了 38K 载波信号的占空比，其占空比单位为：

$$D_{UNIT} = \left(\frac{1}{f_{PCLK} / 38Khz} \right)$$

其中：f_{PCLK} 为 PCLK 频率。

示例：

在 UARTx_IRC 寄存器”IRPN”位为 0，且 f_{PCLK} 为 48Mhz 时，如果要获得 20%的正占空比：

通过上式，可知 D_{UNIT}=0.08%，20%占空比需要”20/0.08=250”单位，将 250 赋值给 DUTY[11:0]即可。

注意：

1. DUTY[11:0]默认为 0，占空比为 50%

2. 若配置的 DUTY[11:0]所表示的脉宽宽度超出了 38Khz 载波周期，则可能会出现无法预计的异常调制波形

15.4 UART 寄存器描述

15.4.1 UART 数据寄存器(UARTx_DR)

（地址： UART0：0x4000_4400；UART1：0x4001_3800）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	DR[8:0]								
R/W	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 8：0	**DR[8:0]：数据寄存器(UART data)** 待发送或者已经收到的数据 数据寄存器对应两个缓冲队列：一个用于写(发送缓冲)，另外一个用于读(接收缓冲)： ● 写操作时硬件将数据同步发送缓冲队列； ● 读操时作硬件将返回接收缓冲队列里的数据； 数据帧的格式均为右对齐，根据 (UARTx_CR)中 WP[2:0]位对于数据帧格式的选择，数据的发送和接收可以是 7 位~9 位的： ● 对于 7 位的数据、发送和接收时只会用到 DR[6:0]； ● 对于 8 位的数据、发送和接收时只会用到 DR[7:0]； ● 对于 9 位的数据、发送和接收时将会用到 DR[8:0]；
位 31：9	保留。必须保持为 0。

15.4.2 UART 控制寄存器(UARTx_CR)

（地址： UART0：0x4000_4404；UART1：0x4001_3804）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	OWD	OWE	-	-	TXP	RXP
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	Res	Res	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	RE	UE	LPB	PS	STOP[1:0]		WP[2:0]		
R/W	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 2:0	**WP[2:0]**：数据帧与校验位配置**(Word length and parity config)** **WP[2:0]**决定了数据帧的字长和校验位控制，该位由软件设置和清零。 001：8 个数据位 011：7 个数据位+1 位校验位 100：9 个数据位 111：8 个数据位+1 位校验位 ***注：*** ***1. 在数据传输过程中(发送或者接收时)，不应修改这个位。*** ***2. 校验位使能后，数据帧最高位将被替换为校验位，意味着 8 位数据位时仅 7 位数据有效，9 位数据位时仅 8 位有效。***
位 4：3	**STOP[1:0]**：停止位长度**(Stop bit length)** 这 2 位用来设置停止位的位数 00： 0.5 个停止位 01： 1 个停止位 10： 1.5 个停止位 11： 2 个停止位
位 5	**PS**：校验方式**(parity mode)** 0： 偶校验 1： 奇校验
位 6	**LPB**：回绕模式控制**(loopback mode)** 0： 正常数据接收和发送模式 1： 内部自发自收模式（内部数据回绕）
位 7	**UE**：**UART** 使能**(UART enable)** 0： UART 禁止 1： UART 使能
位 8	**RE**：接收使能**(Receiver enable)**

	0： 禁止接收 1： 使能接收，并开始搜寻 RX 引脚上的起始位。
位 15：9	保留。必须保持为 0。
位 16	RXP：URAT 数据接收极性控制(Receiver polarity control) 0： 标准数据极性（空闲为高电平，起始位为低电平；数据 1=高电平，0=低电平） 1： 反相数据极性（空闲为低电平，起始位为高电平；数据 1=低电平，0=高电平）
位 17	TXP：URAT 数据发送极性控制(Transmitter polarity control) 0： 标准数据极性（空闲为高电平，起始位为低电平；数据 1=高电平，0=低电平） 1： 反相数据极性（空闲为低电平，起始位为高电平；数据 1=低电平，0=高电平）
位 19：18	保留。必须保持为 0。
位 20	OWE：URAT 单线通讯模式使能控制(single-line communication mode enable) 0： 标准两线全双工模式，模块对应的 TX 引脚为数据发送，RX 引脚为数据接收 1： 单线半双工模式，模块对应的 TX 引脚为数据接收或发送，RX 引脚则为发送/接收状态指示（见下面 OWD 位说明）
位 21	OWD：URAT 单线通讯接收/发送方向设置(single-line communication direction setting) 该位只有在 OWE 置为 1 使能单线通讯模式时才有意义 0： 单线模式下模块 TX 引脚为串行数据接收 1： 单线模式下模块 TX 引脚为串行数据发送
位 31：22	保留。必须保持为 0。

15.4.3 UART 波特率寄存器(UARTx_BRR)

（地址： UART0：0x4000_4408；UART1：0x4001_3808）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	BR[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

位 15：0	**BR[15:0]**：波特率分频系数**(Baud rate control)** 该值的设定使用方法请参考 UART 波特率计算和设定章节描述
位 31：16	保留。必须保持为 0。

15.4.4 UART 中断控制寄存器(UARTx_IE)

（地址： UART0：0x4000_440C；UART1：0x4001_380C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	TXOI	TXFI	-	TXEI	OVRI	FEI	PEI	-	-	RXFI	-	RX NEI
R/W	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**RXNEI**：接收 **FIFO** 非空中断使能**(Receive FIFO not empty interrupt enable)** 0： 禁止中断请求 1： 允许中断请求
位 1	保留。必须保持为 0。
位 2	**RXFI**：接收 **FIFO** 全满中断使能**(Receive FIFO full interrupt enable)** 0： 禁止中断请求 1： 允许中断请求
位 4:3	保留。必须保持为 0。
位 5	**PEI**：校验错误中断使能**(parity error interrupt enable)** 0： 禁止中断请求 1： 允许中断请求
位 6	**FEI**：帧错误中断使能**(frame error interrupt enable)** 0： 禁止中断请求 1： 允许中断请求
位 7	**OVRI**：接收 **FIFO** 溢出错误中断使能**(Receive FIFO overflow interrupt enable)** 0： 禁止中断请求 1： 允许中断请求
位 8	**TXEI**：发送 **FIFO** 为空中断使能**(Transmit FIFO empty interrupt enable)** 0： 禁止中断请求 1： 允许中断请求
位 9	保留。必须保持为 0。
位 10	**TXFI**：发送 **FIFO** 全满中断使能**(Transmit FIFO full interrupt enable)** 0： 禁止中断请求 1： 允许中断请求
位 11	**TXOI**：发送数据完毕中断使能**(Transmit end interrupt enable)**

	0： 禁止中断请求 1： 允许中断请求 ***注：TXO 默认为 1，且当 FIFO 中没有数据时，一直为 1；为避免意外的中断，应当谨慎使用 TXOI 中断。***
位 31：12	保留。必须保持为 0。

15.4.5 UART 状态寄存器(UARTx_SR)

（地址： UART0：0x4000_4410；UART1：0x4001_3810）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	TXO	TXF	-	TXE	OVR	FE	PE	-	-	RXF	-	RXNE
R/W	Res	Res	Res	Res	R	R	Res	R	RW1	RW1	RW1	Res	Res	R	Res	R
复位	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0

位 0	**RXNE**：接收 FIFO 非空**(Receive FIFO not empty flag)** 0： 接收 FIFO 为空 1： 接收 FIFO 非空
位 1	保留。必须保持为 0。
位 2	**RXF**：接收 FIFO 全满**(Receive FIFO full flag)** 0： 接收 FIFO 未满 1： 接收 FIFO 全满
位 4:3	保留。必须保持为 0。
位 5	**PE**：校验错误**(parity error flag)** 该位由硬件置**'1'**，软件写**'1'**清除 0： 未检测到奇偶校验错误 1： 检测到奇偶校验错误
位 6	**FE**：帧错误**(frame error flag)** 该位由硬件置**'1'**，软件写**'1'**清除 0： 未检测到帧错误 1： 检测到帧错误
位 7	**OVR**：接收 FIFO 溢出错误**(Receive FIFO overflow flag)** 该位由硬件置**'1'**，软件写**'1'**清除 0： 接收 FIFO 未发生溢出错误 1： 接收 FIFO 发生溢出错误
位 8	**TXE**：发送 FIFO 为空**(Transmit FIFO empty flag)** 0： 发送 FIFO 非空 1： 发送 FIFO 为空
位 9	保留。必须保持为 0。
位 10	**TXF**：发送 FIFO 全满**(Transmit FIFO full flag)** 0： 发送 FIFO 未满 1： 发送 FIFO 全满

位 11	TXO：发送数据完毕(Transmit end flag) 0： 正在发送数据 1： 发送数据完毕
位 31：12	保留。必须保持为 0。

15.4.6 UART 发送 FIFO 复位寄存器(UARTx_TXFR)

（地址： UART0：0x4000_441C；UART1：0x4001_381C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	TXFR[31:16]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TXFR[15:0]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 31：0	**TXFR[31:0]**：发送 FIFO 复位**(Transmit FIFO reset)** 对 TXFR 写入任意值，发送 FIFO 都将立即被复位清空。 该位只写，读恒为零。

15.4.7 UART 接收 FIFO 复位寄存器(UARTx_RXFR)

（地址： UART0：0x4000_4420；UART1：0x4001_3820）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	RXFR[31:16]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	RXFR[15:0]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 31：0	**RXFR[31:0]**：接收 FIFO 复位**(Receive FIFO reset)** 对 RXFR 写入任意值，接收 FIFO 都将立即被复位清空。 该位只写，读恒为零。

15.4.8 UART 红外调制控制寄存器(UARTx_IRC)

（地址： UART0：0x4000_4428；UART1：0x4001_3828）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	IRBD[1:0]		-	-	IRPN	IRE
R/W	Res	Res	Res	Res	RW	Res	Res	Res	Res	RW	RW	RW	Res	Res	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**IRE:** 红外功能使能控制**(Infrared enable)** 0： 红外功能禁止 1： 红外功能使能
位 1	**IRPN:** 红外发送极性控制**(Infrared polarity control)** **0**： 正常极性 **1**： 反极性
位 3：2	保留。必须保持为 0。
位 5：4	**IRBD[1:0]:** 红外速率选择控制**(Infrared baud rate)** 00： 600bps 01： 1200bps 10： 2400bps 11： 保留
位 31：6	保留。必须保持为 0。

15.4.9 UART 红外调制占空比控制寄存器(UARTx_IRDC)

（地址： UART0：0x4000_442C；UART1：0x4001_382C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	DUTY[11:0]											
R/W	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 11：0	**DUTY[11:0]**：红外调制占空比控制**(Infrared duty cycle)** 详见 15.3.6.1 节描述。
位 31：12	保留。必须保持为 0。

15.4.10 寄存器列表

地址	寄存器	描述	备注
0x4000_4400	UART0_DR	UART0 数据寄存器	UART0_DR 说明
0x4000_4404	UART0_CR	UART0 控制寄存器	UART0_CR 说明
0x4000_4408	UART0_BRR	UART0 波特率寄存器	UART0_BRR 说明
0x4000_440C	UART0_IE	UART0 中断控制寄存器	UART0_IE 说明
0x4000_4410	UART0_SR	UART0 状态寄存器	UART0_SR 说明
0x4000_441C	UART0_TXFR	UART0 发送 FIFO 复位寄存器	UART0_TXFR 说明
0x4000_4420	UART0_RXFR	UART0 接收 FIFO 复位寄存器	UART0_RXFR 说明
0x4000_4428	UART0_IRC	UART0 红外调制控制寄存器	UART0_IRC 说明
0x4000_442C	UART0_IRDC	UART0 红外调制占空比控制寄存器	UART0_IRDC 说明
地址	寄存器	描述	备注
0x4001_3800	UART1_DR	UART1 数据寄存器	UART1_DR 说明
0x4001_3804	UART1_CR	UART1 控制寄存器	UART1_CR 说明
0x4001_3808	UART1_BRR	UART1 波特率寄存器	UART1_BRR 说明
0x4001_380C	UART1_IE	UART1 中断控制寄存器	UART1_IE 说明
0x4001_3810	UART1_SR	UART1 状态寄存器	UART1_SR 说明
0x4001_381C	UART1_TXFR	UART1 发送 FIFO 复位寄存器	UART1_TXFR 说明
0x4001_3820	UART1_RXFR	UART1 接收 FIFO 复位寄存器	UART1_RXFR 说明
0x4001_3828	UART1_IRC	UART1 红外调制控制寄存器	UART1_IRC 说明
0x4001_382C	UART1_IRDC	UART1 红外调制占空比控制寄存器	UART1_IRDC 说明

16 片内闪存控制器(IFMC)

16.1 综述

PT32x00x 提供最大 32K 字节的片内 Flash 闪存存储空间。
除了存储用户程序之外，这块片内 Flash 闪存还可以被用于存储一些数据；通过 IFMC 可以对 Flash 进行访问。

16.2 特性

IFMC 基本特性如下：

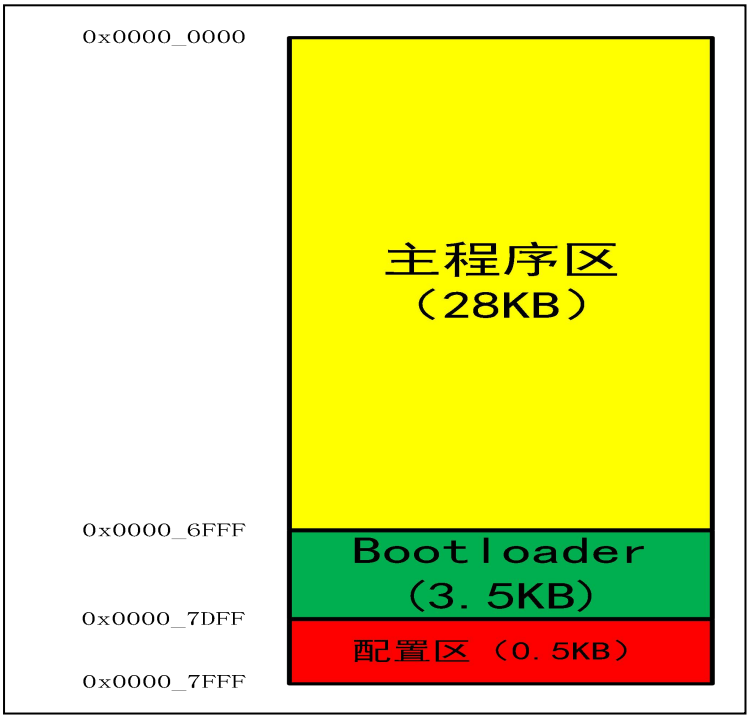
- 最大 32K 字节的片内 Flash 闪存可用空间
- 32 位的(字)编程操作
- 512 字节的(页)擦除操作
- 全片擦除操作
- 严格的读保护机制，当读保护解除，主程序区的所有内容将被擦除
- IFMC 快速操作：
  - 全片擦除最大 280mS
  - 页擦除最大 6mS
  - 写操作最大 7.5uS
  - 读操作最大 42nS

16.3 IFMC 功能描述

16.3.1 闪存模块组织

PT32x00x 提供 20K 字节/32K 字节两种 Flash 规格，Flash 的空间在不同的 Flash 规格下，有着不同的区域划分，如下图所示

图 16-1 32KB 规格下的 Flash 规格划分



16.3.1.1 20K 字节规格

表 16-1

区域	大小	地址范围
Bootloader 区	0~3.5KB，自定义	0x0000_4000~ 0x0000_4DFF
主程序区	19.5~16KB，自定义	0x0000_0000~ 0x0000_4DFF
配置区	0.5KB	0x0000_7E00 ~ 0x0000_7FFF

16.3.1.2 32K 字节规格

表 16-2

区域	大小	地址范围
Bootloader 区	0~3.5KB，自定义	0x0000_7000~ 0x0000_7DFF
主程序区	31.5~28KB，自定义	0x0000_0000~ 0x0000_7DFF
配置区	0.5KB	0x0000_7E00 ~ 0x0000_7FFF

注意：Bootloader 区和主程序区的大小均通过烧录器或配置字实现，烧录器可联系供应商提供。

表 16-3 闪存模块组织(20KB，主程序区 16KB，Bootloader 区 3.5KB，配置区 0.5KB)

区域	名称	地址范围	长度(字节)
主程序区	页 0	0x0000_0000~0x0000_01FF	512
	页 1	0x0000_0200~0x0000_03FF	512
	页 2	0x0000_0400~0x0000_05FF	512
	页 3	0x0000_0600~0x0000_07FF	512
	页 4	0x0000_0800~0x0000_09FF	512
	. . .	. . .	. . .
	31 页	0x0000_3E00~0x0000_3FFF	512
Bootloader 区	页 0	0x0000_4000~0x0000_41FF	512
	页 1	0x0000_4200~0x0000_43FF	512
	页 2	0x0000_4400~0x0000_45FF	512
	页 3	0x0000_4600~0x0000_47FF	512
	页 4	0x0000_4800~0x0000_49FF	512
	页 5	0x0000_4A00~0x0000_4BFF	512
	页 6	0x0000_4C00~0x0000_4DFF	512
配置区	页 1	0x0000_7E00 ~ 0x0000_7FFF	512

表 16-4 闪存模块组织(32KB，主程序区 28KB，Bootloader 区 3.5KB，配置区 0.5KB)

区域	名称	地址范围	长度(字节)
主程序区	页 0	0x0000_0000~0x0000_01FF	512
	页 1	0x0000_0200~0x0000_03FF	512
	页 2	0x0000_0400~0x0000_05FF	512
	页 3	0x0000_0600~0x0000_07FF	512
	页 4	0x0000_0800~0x0000_09FF	512
	. . .	. . .	. . .
	55 页	0x0000_6E00~0x0000_6FFF	512
Bootloader 区	页 0	0x0000_7000~0x0000_71FF	512
	页 1	0x0000_7200~0x0000_73FF	512
	页 2	0x0000_7400~0x0000_75FF	512
	页 3	0x0000_7600~0x0000_77FF	512
	页 4	0x0000_7800~0x0000_79FF	512
	页 5	0x0000_7A00~0x0000_7BFF	512
	页 6	0x0000_7C00~0x0000_7DFF	512
配置区	页 1	0x0000_7E00 ~ 0x0000_7FFF	512

16.3.2 读操作

IFMC 封装了一个读接口，这个接口的主要工作是产生读闪存的控制信号并获取 CPU 要求的指令，数据和指令通过这个接口被 CPU 取读或执行。
可以在地址空间直接对片内 Flash 寻址，任何 32 位数据的读操作都能访问闪存模块的内容并得到相应的数据。

16.3.2.1 访问时间调节器

为了保证读接口的控制信号，IFMC 在(控制寄存器 IFMC_CR)中提供了一个可配置的'WAIT'位以控制读接口的时钟周期与闪存访问时间的比例。
'WAIT'位给出了能够正确地读取数据时，闪存控制信号所需的时钟周期数目，芯片复位后，该值为'0'，闪存访问仅为一个时钟周期。

注意：HCLK 频率高于 24Mhz 时, WAIT 必须被设置为 1 周期等待；

16.3.3 IFMC 安全机制

为避免错误的操作将 Flash 闪存中存储的用户程序覆盖而导致非预期的结果、在芯片复位后，IFMC 模块是被保护的；
这个保护 IFMC 的安全机制在用户操作(控制寄存器 IFMC_CR)时，执行下列操作：

- 获取 KEY[15:0]的值
- 判断 KEY[15:0]值与(地址寄存器 IFMC_AR)中的地址所属的 FLASH 区域是否一一对应
- 不满足对应的条件，将(IFMC_SR 寄存器)中的 KERR 位置'1'

16.3.3.1 IFMC 解锁安全机制

通过写入特定的 KEY 值到(IFMC_CR)寄存器以解锁 IFMC 模块，这个特定的 KEY 值根据要操作的 Flash 闪存的区域有：

- 主程序区： 0xADEB
- 配置区： 0xC5AE

注意：

1、KERR 位将被硬件置'1'后，软件清除 KERR 位，并写入正确的 KEY 值或地址以恢复到正常工作。

16.3.4 IFMC 写入和擦除操作

IFMC 控制器集成了对内部 Flash 闪存进行擦除和编程的接口，使得应用程序无需关注内部 Flash 闪存在编程和擦除时的物理时序和逻辑控制。

只需对 IFMC 的相关寄存器进行读写操作，设定操作命令、地址和数据，即可对内部 Flash 闪存进行如下操作：

- 写操作
- 页擦除
- 全片擦除(仅主程序区)

注意：

1：写入前应确保单元数据为 0xFFFFFFF（擦除状态），否则将产生意外的数据

2：IFMC 对内部 Flash 闪存的擦写操作均以’字(32bit)’为单位进行。

3：程序运行在 FLASH 并且执行 IFMC 操作时，程序将等待 IFMC 操作直到操作完毕

16.3.4.1 写入和擦除时钟配置

IFMC 的”写入和擦除”时钟由 HCLK 经(IFMC_PSC)寄存器的”PSC[5:0]”分频而来：

$$f_{IFMC} = \frac{f_{HCLK}}{PSC[5:0]}$$

其中：

- f_{IFMC} 为 IFMC 时钟的频率
- f_{HCLK} 为 HCLK 的频率

注意：IFMC 擦写时钟不应超过 1Mhz

16.3.4.2 写操作

写操作以”字(32bit)”为单位进行，执行写操作前，应当确保下列前提：

- (IFMC_SR)寄存器 BUSY 位为’0’

一次完整的写操作应该如下列所示：

- (IFMC_AR)寄存器 AR 位置为正确的地址值
- (IFMC_DR)寄存器 DR 位写入数据
- (IFMC_CR)寄存器 CMD 位置为’0x00’
- (IFMC_CR)寄存器 MODE 位置’1’
- (IFMC_CR)寄存器 KEY 位置为正确的 KEY 值。
- (IFMC_CR)寄存器 STAR 位置’1’，执行 IFMC 操作。

特别地、当(IFMC_IE)寄存器中的 WOVI 位使能，完成一次写操作后，将产生一个中断请求到 NVIC。

注意：对于(IFMC_CR)寄存器的写入操作，应该是一次性的，可通过一个临时变量，存储所有对于(IFMC_CR)寄存器的配置，再一次性的赋值给(IFMC_CR)寄存器。

16.3.4.3 页擦除

页擦除以”512Byte”为单位进行，执行页擦除前，应当确保下列前提：

- (IFMC_SR)寄存器 BUSY 位为’0’

一次完整的页擦除应该如下列所示：

- (IFMC_AR)寄存器 AR 位置为目标页的起始地址值
- (IFMC_CR)寄存器 CMD 位置为’0x01’
- (IFMC_CR)寄存器 MODE 位置’1’
- (IFMC_CR)寄存器 KEY 位置为正确的 KEY 值。
- (IFMC_CR)寄存器 STAR 位置’1’，执行 IFMC 操作。

特别地、当(IFMC_IE)寄存器中的 POV 位使能，完成一次页擦除后，将产生一个中断请求到 NVIC。

注意：应当谨慎的使用页擦除操作，避免擦除到主程序的存放地址，引发意外的后果。

16.3.4.4 全片擦除

全片擦除将擦除主程序区的所有数据，执行全片擦除前，应当确保下列前提：

- (IFMC_SR)寄存器 BUSY 位为’0’
- 一次完整的全片擦除应该如下列所示：
- (IFMC_CR)寄存器 CMD 位置为’0x11’或’0x10’
- (IFMC_CR)寄存器 MODE 位置’1’
- (IFMC_CR)寄存器 KEY 位置为正确的 KEY 值。
- (IFMC_CR)寄存器 STAR 位置’1’，执行 IFMC 操作。

特别地、当(IFMC_IE)寄存器中的 COV 位使能，完成一次全片擦除后，将产生一个中断请求到 NVIC。

注意：

1、全片擦除仅针对主程序区

2、执行全片擦除后，主程序区的所有数据将被清除，包括用户烧录到 Flash 中的程序。

16.3.5 配置区操作

无论 Flash 闪存的规格为 20KB 还是 32KB，配置区的地址总是为 0x0000_7E00 ~ 0x0000_7FFF。

配置区的访问基于 IFMC 控制器，且读无效；

在配置区中，有 3 个配置字用于配置下列功能：

- 用户定义 ID 值
- Bootloader 配置
- Flash 读保护

配置字的修改，均在发生一次上电复位或者高级软件复位后生效，此时，相关的参数将被映射到对应的寄存器中。

注意：烧录器提供了完整而便捷的配置区操作、ICP 应用和普通烧录支持，推荐使用烧录器对配置区操作，烧录器可联系附近供应商提供。

表 16-5 用户配置字说明

地址	名称	说明	位域
0x0000_7E00	UDID	**用户定义 ID 值(UDID)：** UDID 值在芯片上电之后被映射到”用户定义 ID 寄存器(ID_UDID)”中，用户可以通过”用户定义 ID 寄存器(ID_UDID)”便捷的获取 UDID 值，这个 32bit 的值可以方便的支持用户自身的版本管理。	位[31:0]
0x0000_7E04	BCF	**系统启动方式(BOOT)：** BOOT=0xC4：系统从 Bootloader 区启动 BOOT=other：系统从主程序区启动	位[7:0]
		Bootloader 区域的大小(BSZ)： BSZ=0x0：Bootloader 区域大小为 0，无 Bootloader BSZ=0x1：Bootloader 区域大小为 1.5KB BSZ=0x0：Bootloader 区域大小为 2.5KB BSZ=0x0：Bootloader 区域大小为 3.5KB	位[9:8]
		保留	位[31:10]
0x0000_7E08	RDP	**Flash 读保护(RDP)：** RDP=0x15EC1CCA：Flash 将处于读保护状态 RDP=other: Flash 不处于读保护状态	位[31:0]

16.3.6 用户定义 ID(UDID)

通过配置字 UDID 或者烧录器以修改 UDID 值。
UDID 值被映射到”用户定义 ID 寄存器(ID_UDID)”中，用户可以通过(寄存器 ID_UDID)便捷的获取 UDID 值，这个 32bit 的值可以方便的支持用户自身的版本管理。

16.3.7 系统启动配置

通过配置字 BCF 或者烧录器，用户可以配置系统启动方式，PT32x00x 提供两种启动方式：

表 16-6 启动方式

启动方式	说明
系统从主程序区启动	主程序区被选为启动区域，默认的启动方式
系统从 Bootloader 区启动	Bootloader 区被选为启动区域(ICP 方式烧录所需的启动方式)

系统启动方式的信息被映射到”IFMC BOOT 状态寄存器(IFMC_BSR)”中，不同的启动方式将导致 Flash 的地址映射不同，如下图所示。

图 16.2 启动方式与地址映射

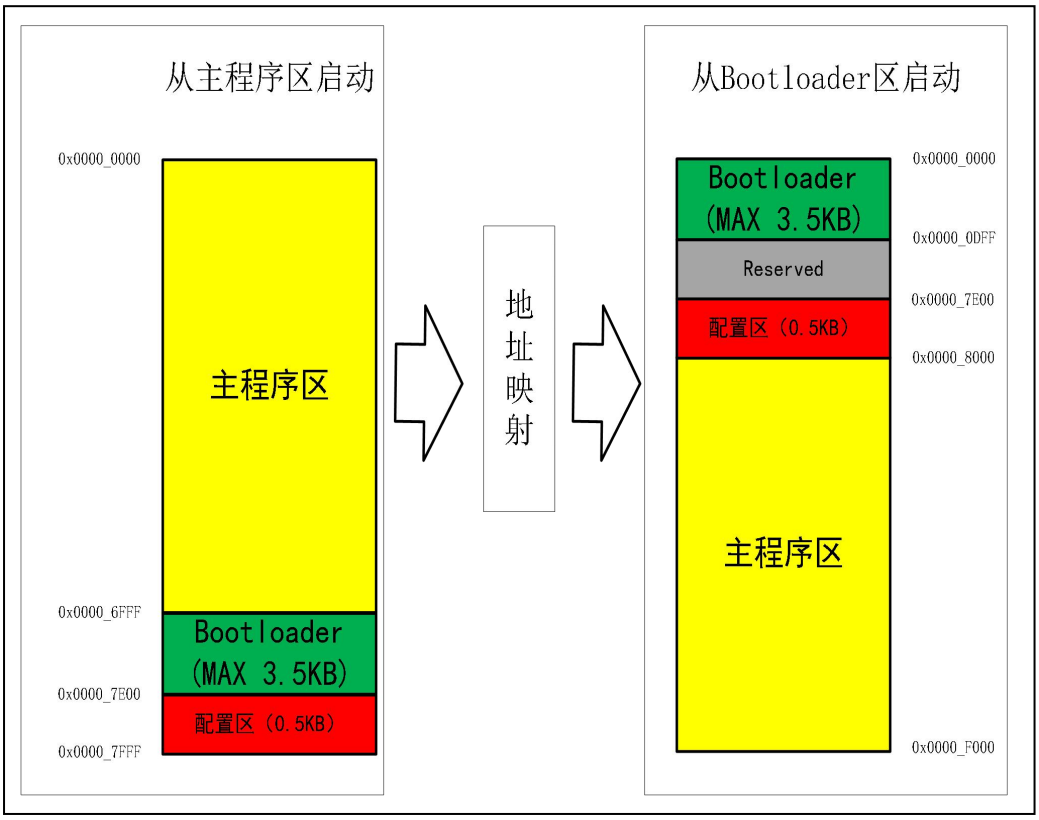
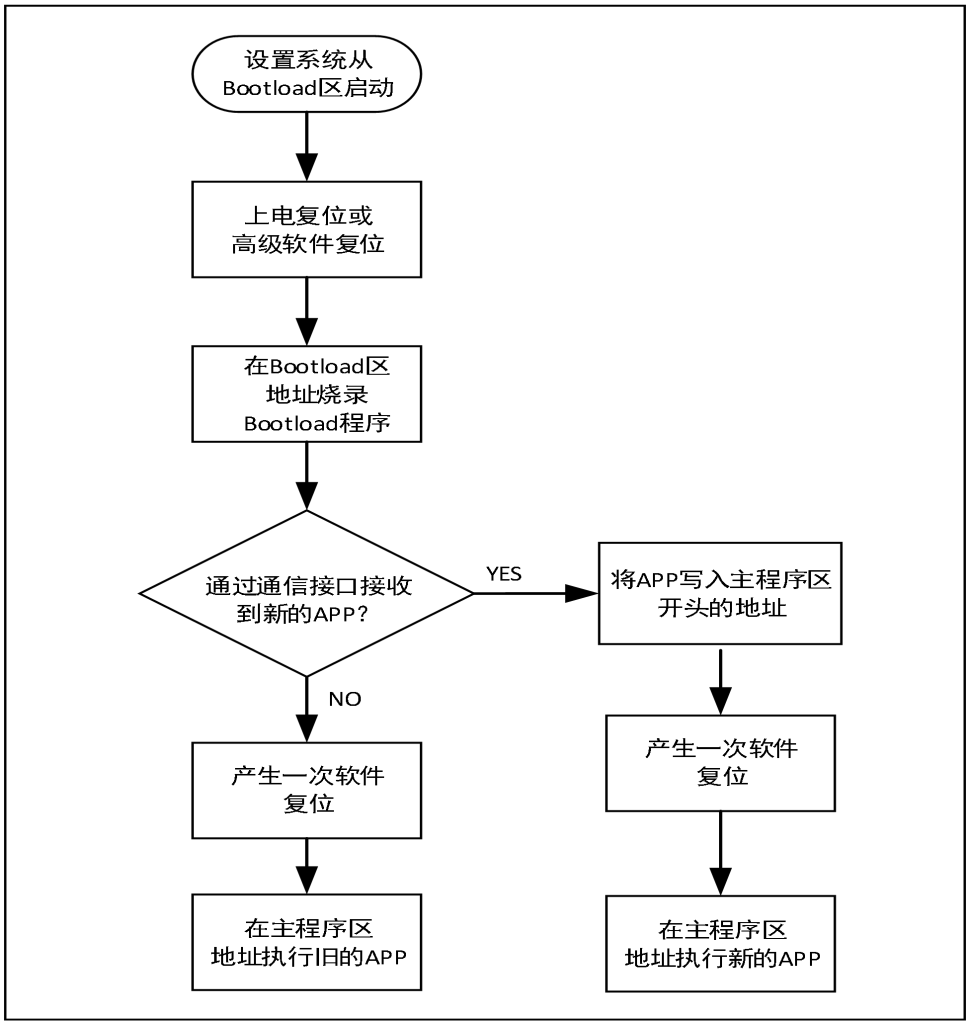


图 16.3 ICP 应用



16.3.8 Flash 读保护

PT32x00x 提供高级别的 Flash 程序区域代码安全保护机制。通过配置字 RDP 或者烧录器使能读保护后，执行下列机制：

- 仍允许用户代码中对片内 Flash 的读操作
- 禁止通过 SWD 以调试方式从片内 Flash 读取任何数据
- 严格的读保护机制，当读保护解除，主程序区的所有内容将被擦除

Flash 读保护状态被映射到”IFMC 读保护状态寄存器(IFMC_RPT)”中，用户代码可以通过(寄存器 IFMC_RPT)便捷的获取到 Flash 的读保护状态。

16.3.9 状态和中断

通过下述的状态，应用程序得以完全监控 IFMC，除 BUSY 和 KERR 状态以外、IFMC 所有的状态均支持中断请求。
将(IFMC_IE)寄存器中相应状态的中断使能、在状态产生之后，将同步向 NVIC 发出一个中断请求；当 NVIC 中的 IFMC 中断被使能，系统将处理这些中断请求。

16.3.9.1 IFMC 写操作命令完成(WOV)

此标志为'1'时、表明前一个执行的写操作完成。当使能了 WOV 中断"WOVI"位后，将同步产生一个中断请求到 NVIC；
对(IFMC_SR)寄存器的"WOV"位置'1'以清除此标志，相应的中断请求被同步清除。

16.3.9.2 IFMC 页擦除操作命令完成(POV)

此标志为'1'时、表明前一个执行的页擦除操作完成。当使能了 POV 中断"POVI"位后，将同步产生一个中断请求到 NVIC；
对(IFMC_SR)寄存器的"POV"位置'1'以清除此标志，相应的中断请求被同步清除。

16.3.9.3 IFMC 全片擦除操作命令完成(COV)

此标志为'1'时、表明前一个执行的全片擦除操作完成。当使能了 COV 中断"COVI"位后，将同步产生一个中断请求到 NVIC；
对(IFMC_SR)寄存器的"COV"位置'1'以清除此标志，相应的中断请求被同步清除。

16.3.9.4 IFMC 操作忙(BUSY)

此标志只读，当此标志为'1'时、表明当前正在执行 IFMC 操作，此时、IFMC 不会响应任何操作，直到 BUSY 空闲。

16.3.9.5 IFMC 操作错误(CERR)

操作错误有下述情况：

- 当 FLASH 处于读保护时，软件通过 swd 接口对主程序区的地址进行写操作
- 当 IFMC_AR 寄存器的地址不在主程序区时，执行 Flash 全片擦除命令
- 当系统从主程序区启动，并对 Bootloader 程序区域执行写操作。

此标志为'1'时、表明触发了上述的任一错误。当使能了 CERR 中断"CERI"位后，将同步产生一个中断请求到 NVIC；
对(IFMC_SR)寄存器的"CERR"位置'1'以清除此标志，相应的中断请求被同步清除。

16.3.9.6 IFMC 密码错误(KERR)

参考 16.3.2 IFMC 安全机制

此标志为'1'时、表明触发了上述错误。当使能了 KERR 中断"KERI"位后，将同步产生一个中断请求到 NVIC；

对(IFMC_SR)寄存器的"KERR"位置'1'以清除此标志，相应的中断请求被同步清除。

16.3.9.7 IFMC 地址错误(AERR)

地址错误有下述情况：

- 对超出当前 Flash 闪存规格的地址进行 IFMC 操作
- 当系统从 Bootloader 区启动时，对 Reserved 区域进行 IFMC 操作

此标志为'1'时、表明触发了上述的任一错误。当使能了 AERR 中断"AERI"位后，将同步产生一个中断请求到 NVIC；

对(IFMC_SR)寄存器的"AERR"位置'1'以清除此标志，相应的中断请求被同步清除。

16.4 IFMC 寄存器描述

16.4.1 IFMC 控制寄存器（IFMC_CR）

（地址： 0x4000_0000）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	KEY[15:0]															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	AINC	-	-	-	-	MODE	-	-	WAIT	-	-	CMD[1:0]		STAR
R/W	Res	Res	RW	Res	RW	Res	Res	RW	Res	Res	RW	Res	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

位 0	**STAR**：开始 **IFMC** 操作**(Start generation)** 0： 执行 IFMC 操作完毕 1： 执行 IFMC 操作
位 2：1	**CMD[1:0]**：**IFMC** 操作命令**(IFMC operation command)** 00： 写操作 01： 页擦除 1xb： 全片擦除
位 4：3	保留。必须保持为 0。
位 5	**WAIT**：**IFMC** 取指令等待周期**(instruction fetch wait cycle)** 设定 IFMC 取指令等待周期 0： 0 周期等待 1： 1 周期等待
位 7：6	保留。必须保持为 0。
位 8	**MODE**：**IFMC** 操作模式**(IFMC operation mode)** 0： IFMC 擦写模式关 1： IFMC 擦写模式开
位 12：9	保留。必须保持为 0。
位 13	**AINC**：**IFMC** 地址自动递增**(Address auto increment control)** 0： IFMC 写一次 Flash 后，IFMC_AR 寄存器值保持不变 1： IFMC 写一次 Flash 后，IFMC_AR 寄存器值自动加 4
位 15：14	保留。必须保持为 0。
位 31：16	**KEY[15:0]**：**IFMC** 闪存锁 **KEY** 值**(IFMC key)** 根据要操作的 **Flash** 闪存的区域指定合适的 **KEY** 值，参考"IFMC 解锁安全机制"一节描述

16.4.2 IFMC 状态寄存器（IFMC_SR）

（地址： 0x4000_0004）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	AERR	KERR	CERR	BUSY	COV	POV	WOV
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW1	RW1	RW1	R	RW1	RW1	RW1
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**WOV：IFMC 写操作完成(Write operation complete)** 0： 写操作未完成 1： 写操作完成，置 1 以清 0 状态位
位 1	**POV：IFMC 页擦除操作完成(Page erase operation complete)** 0： 页擦除操作未完成 1： 页擦除操作完成，置 1 以清 0 状态位
位 2	**COV：IFMC 全片擦除操作完成(Chip erase operation complete)** 0： 全片擦除操作未完成 1： 全片擦除操作完成，置 1 以清 0 状态位
位 3	**BUSY：IFMC 操作忙(Operation busy)** 0： IFMC 处于空闲状态 1： IFMC 操作忙，该位由硬件自动清除
位 4	**CERR：IFMC 操作错误(Operation command error)** 0： 操作正确 1： 操作错误，置 1 以清 0 状态位
位 5	**KERR：IFMC 密码错误(key error)** 0： 密码正确 1： 密码错误，置 1 以清 0 状态位
位 6	**AERR：Flash 地址错误(Address error)** 0： 地址正确 1： 地址错误，置 1 以清 0 状态位
位 31：7	保留。必须保持为 0。

16.4.3 IFMC 中断控制寄存器（IFMC_IE）

（地址： 0x4000_0008）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	AERI	KERI	CERI	-	COVI	POVI	WOVI
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	Res	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**WOVI：IFMC 写操作完成中断使能(Write operation complete interrupt enable)** 0： 中断禁止 1： 中断使能
位 1	**POVI：IFMC 擦除操作完成中断使能(Page erase operation complete interrupt enable)** 0： 中断禁止 1： 中断使能
位 2	**COVI：IFMC 全片擦除操作完成中断使能(Chip erase operation complete interrupt enable)** 0： 中断禁止 1： 中断使能
位 3	保留。必须保持为 0。
位 4	**CERI：IFMC 操作错误中断使能(Operation command error interrupt enable)** 0： 中断禁止 1： 中断使能
位 5	**KERI：IFMC 操作密码错误中断使能(key error interrupt enable)** 0： 中断禁止 1： 中断使能
位 6	**AERI：IFMC 地址错误中断使能(Address error interrupt enable)** 0： 中断禁止 1： 中断使能
位 31：7	保留。必须保持为 0。

16.4.4 IFMC 地址寄存器（IFMC_AR）

（地址： 0x4000_000C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	AR[13:0]														-	-
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 1：0	保留。必须保持为 0。
位 16：2	**AR[13:0]：Flash 逻辑地址(IFMC addr)** 这些位定义了 IFMC 执行操作时所需要的地址，参考 16.3.1 闪存模块组织描述。
位 31：17	保留。必须保持为 0。

16.4.5 IFMC 数据寄存器（IFMC_DR）

（地址： 0x4000_0010）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	DR[31:16]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	DR[15:0]															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 31：0	**DR[31:0]：Flash 编程数据(IFMC data)** 这些位定义了 IFMC 执行写操作时所要写入目标地址的数据，参考 16.3.3.2 写操作描述。

16.4.6 IFMC 预分频寄存器（IFMC_PSC）

（地址： 0x4000_0028）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	PSC[5:0]					
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

位 5：0	**PSC[5:0]：Flash 擦写时钟分频系数(Prescaler value)** PSC 用于对 HCLK 进行分频，以产生 1Mhz 的 IFMC 时钟，详见 16.3.2.1 节描述 ***注：PSC 默认值为 24***
位 31：6	保留。必须保持为 0。

16.4.7 IFMC BOOT 状态寄存器（IFMC_BSR）

（地址： 0x4001_F000）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BSR
R/W	Res	Res	Res	Res	Res	Res	Res	Res	RW	RW	RW	RW	RW	RW	RW	R
复位	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

位 0	**BSR：当前 BOOT 状态(Boot status)** 0： 当前程序在主程序区中执行 1： 当前程序在 Bootloader 区中执行 ***注：通过烧录器或配置字可改变程序运行区域，以实现 ICP。更多内容请参考”配置区操作”一节描述***
位 31：1	保留。必须保持为 0。

16.4.8 IFMC 读保护状态寄存器（IFMC_RPT）

（地址： 0x4001_F024）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PSR
R/W	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	Res	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位 0	**PSR：Flash 数据保护状态(Data protection status)** 0： Flash 数据未保护 1： Flash 数据处于保护状态 ***注：Flash 读保护通过烧录器或配置字实现，更多内容请参考”1.6.3.4 配置区操作”，烧录器可联系供应商提供***
位 31：1	保留。必须保持为 0。

16.5 寄存器列表

地址	寄存器	描述	备注
0x4000_0000	IFMC_CR	IFMC 控制寄存器	IFMC_CR 说明
0x4000_0004	IFMC_SR	IFMC 状态寄存器	IFMC_SR 说明
0x4000_0008	IFMC_IE	IFMC 中断使能寄存器	IFMC_IE 说明
0x4000_000C	IFMC_AR	IFMC 地址寄存器	IFMC_AR 说明
0x4000_0010	IFMC_DR	IFMC 数据寄存器	IFMC_DR 说明
0x4000_0028	IFMC_PSC	IFMC 预分频寄存器	IFMC_PSC 说明
0x4001_F000	IFMC_BSR	IFMC BOOT 状态寄存器	IFMC_BSR 说明
0x4001_F024	IFMC_RPT	IFMC 读保护状态寄存器	IFMC_RPT 说明

17 器件电子签名(ID)

17.1 综述

电子签名存放在片内 FLASH 闪存当中，上电之后，它们被映射到特定的寄存器内、用户可以通过 SWD 或者用户程序直接读取它们。
这些电子签名所包含的芯片识别信息在出厂时被编写，通过电子签名，可以自动匹配不同配置的微控制器。

17.2 特性

电子签名的基本特性如下：

- 32bit 的用户自定义信息空间
- 32bit 的芯片配置信息空间
- 96bit 的产品唯一身份标识

17.3 用户定义 ID 寄存器(ID_UDID)

（地址： 0x4001_F020）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	ID[31:16]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	ID[15:0]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD	UD

位 31:0	**ID[31:0]**：用户定义 **ID(User define identifier)** ***注： 用户自定义 ID 通过烧录器或配置字实现，参考”配置区操作”一节描述，烧录器可联系供应商提供***

17.4 产品唯一身份标识 ID 寄存器 1(ID_UID1)

（地址： 0x4001_F030）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	UID[31:16]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	UID[15:0]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD

位 31:0	**UID[31:0]**：产品唯一身份标识 **ID1(unique identifier 1)** ***注： 96 位的产品唯一身份标识只读，对于 MCU，UID 在任何情况下都是唯一的，用户无法修改这个身份标识。***

17.5 产品唯一身份标识 ID 寄存器 2(ID_UID2)

（地址： 0x4001_F034）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	UID[63:48]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	UID[47:32]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD

位 31:0	**UID[63:32]**：产品唯一身份标识 **ID2(unique identifier 2)** ***注： 96 位的产品唯一身份标识只读，对于 MCU，UID 在任何情况下都是唯一的，用户无法修改这个身份标识。***

17.6 产品唯一身份标识 ID 寄存器 3(ID_UID3)

（地址： 0x4001_F038）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	UID[95:80]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	UID[79:64]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD

位 31:0	**UID[95:64]**：产品唯一身份标识 **ID3(unique identifier 3)** ***注： 96 位的产品唯一身份标识只读，对于 MCU，UID 在任何情况下都是唯一的，用户无法修改这个身份标识。***

17.7 芯片配置 ID 寄存器(ID_CID)

（地址： 0x4001_F03C）

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	RES[15:0]															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	C_TYPE[3:0]				-	-	B_SIZE[1:0]		-	-	-	S SIZE	-	-	-	F SIZE
R/W	R	R	R	R	Res	Res	R	R	Res	Res	Res	R	Res	Res	Res	R
复位	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD	MD

位 0	**F_SIZE：FLASH 闪存容量(Size of flash memory)** 0： 16KB 1： 32KB
位 3：1	保留。必须保持为 1。
位 4	**S_SIZE：SRAM 存储器容量(Size of ram memory)** 0： 1KB 1： 2KB
位 7：5	保留。必须保持为 1。
位 9：8	**B_SIZE[1:0]：Bootloader 空间大小(Bootloader size of space)** 00： 0KB 01： 1.5KB 10： 2.5KB 11： 3.5KB ***注： 通过烧录器或配置字可改变 Bootloader 区大小，以实现 ICP。参考”配置区操作”一节描述***
位 15：12	**C_TYPE[3:0]：芯片型号(Chip type)** 1111： F 系列 other： L 系列
位 31:16	**RES[15:0]：默认信息** ***注： 无特殊说明，该位为 0xFFFF。***

18 调试支持(DBG)

18.1 综述

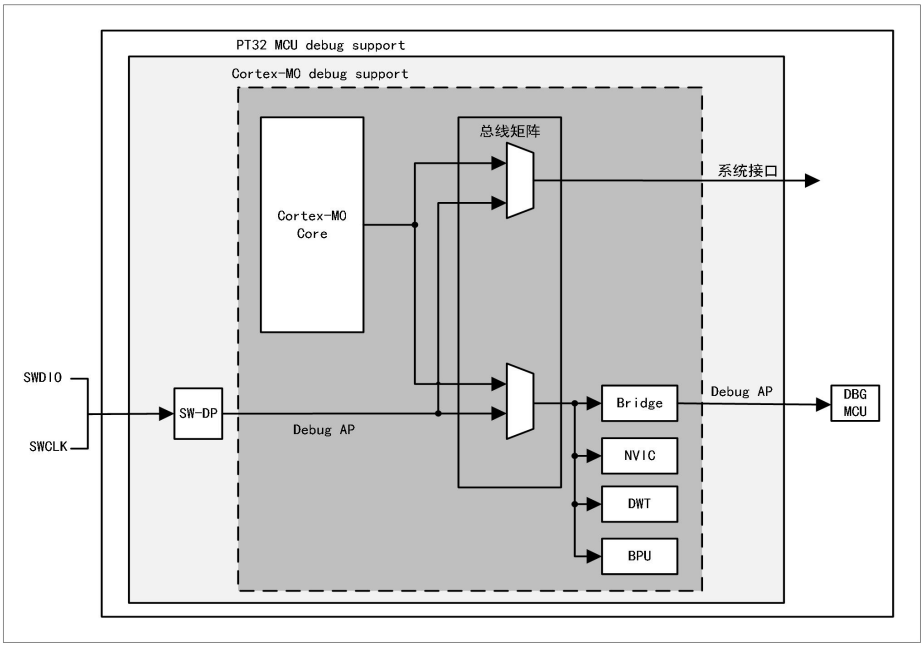
PT32x00x 器件是围绕 Cortex®-M0 内核构建的，该内核包含高级硬件调试模块、支持复杂的调试操作。

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。

当 PT32x00x 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。

PT32x00x 提供了“SW 串行调试接口”用于进行调试操作，如下图所示。

图 18-1 PT32x00x 级别和 Cortex™-M0 级别的调试框图



注意： Cortex™-M0 内核内含的硬件调试模块是 ARM CoreSight 开发工具集的子集。

ARM Cortex™-M0 内核提供集成的片上调试功能。它由以下部分组成：

- SW-DP：串行调试端口
- BPU: 断点单元
- DWT：数据触发

专用于 PT32x00x 的调试特性：

- 灵活的灵活的调试引脚分配
- 在断点时提供高级定时器控制
- 在断点时提供基本定时器控制
- 在断点时提供看门狗控制

注意：更多 Cortex™-M0 内核的调试功能信息，请参考《Cortex™-M0 技术参考手册》。

18.2 ARM 参考文献

- 《Cortex™-M0 技术参考手册》
- 《ARM 调试接口架构规范 V5》
- 《ARM CoreSight 技术参考手册》

18.3 引脚分布和 SW 端口引脚

两个引脚被用作 SW-DP 的输出。作为通用 I/O 的替代功能，这些引脚在所有封装上都可用，更多详细信息参考《PT32x00x 数据手册》

18.3.1 灵活的 SW 端口引脚分配

复位(系统复位、上电复位或功能复位)之后，属于 SW 端口的引脚都被立即初始化为专用的 SW 端口引脚。

然而，PT32x00x 微控制器可以使用 AFIO 功能来禁止 SW 端口部分或者所有引脚的功能，当相应的 SW 端口引脚被 AFIO 禁用，这些引脚将被释放以用作普通 I/O 使用。

注意：复位后，SWD 引脚的复用功能总是为 SW 调试接口，详见 AFIO 章节描述

18.3.2 内部上拉和下拉

一旦 SW 引脚被 AFIO 释放，GPIO 控制器就会对这些引脚进行控制。GPIO 将使它们处于复位状态：

- SWDIO：输入上拉
- SWCLK：输入下拉

18.4 SW 调试端口

18.4.1 SW 协议介绍

此同步串行协议使用 2 个引脚：

- SWCLK：从主机到目标的时钟信号
- SWDIO：双向数据信号

协议允许读写 2 个寄存器组(DPACC 和 APACC 寄存器组)。

数据位按 LSB 传输。

由于 SWDIO 为双向口，该引脚需有上拉(ARM 建议使用 100KΩ电阻)。

按照协议，每次 SWDIO 方向改变时，需插入一个转换时间。在该期间内主机和目标都不驱动此信号线。转换时间的默认值是 1 个比特，但可以通过配置 SWCLK 频率来调节。

注意：DPACC 和 APACC 寄存器的详细资料，请参考《Cortex™-M0 技术参考手册》。

18.4.2 SW 协议序列

每个序列由 3 个阶段组成：

1. 主机发送请求包(8 位)
2. 目标发送确认响应 ACK(3 位)
3. 主机或目标发送数据(33 位)

表 18-1 请求包

比特位	名称	描述
0	起始	必须为 1
1	APnDP	0：访问 DP 1：访问 AP
2	RnW	0：写请求 1：读请求
4：3	A(3:2)	DP 或 AP 寄存器的地址(参考表 18-5 和表 18-6)
5	Parity	请求包[4:0]数据的校验位
6	Stop	0
7	Park	不能由主机驱动，由于有上拉，目标永远读为 1

包请求后总是跟一个(默认为 1 位)转换时间，此时主机和目标都不驱动线路。

表 18-2 ACK 定义(3bit)

比特位	名称	描述
2：0	ACK	001：失败 010：等待 100：成功

当 ACK 为失败或等待，或者是一个回复读操作的 ACK，此 ACK 后有一个转换时间。

表 18-3 传输数据(33bit)

比特位	名称	描述
31：0	WDATA/RDATA	写或读的数据
32	Parity	32 位数据的奇偶校验位

读操作的数据传输操作后有一个转换时间。

18.4.3 SW-DP 状态机

SW-DP 状态机有一个内部 ID 编码用来识别 SW-DP，它遵守 JEP-106 标准。此 ID 编码是 ARM 默认的编码，值为 0x0BB11477(对应于 Cortex-M0)。

注意： 在调试器读这个 ID 编码之前， SW-DP 的状态机是不工作的

- 在上电复位后，或有超过 50 个周期的高电平后，SW-DP 状态机都将处于 RESET 状态。
- 当状态机处于 RESET 状态时，如果有至少 2 个周期的低电平，状态机将切换到 IDLE 状态。
- 当状态机处于 RESET 状态后，必需首先进入 IDLE 状态，并执行一个读 DP-SW ID 寄存器的操作。否则，调试器在执行其他传输时，只能获得一个失败的 ACK 响应。

更详细的 SW-DP 状态机资料请参考《Cortex™-M0 技术参考手册》和《ARM CoreSight 技术参考手册》。

18.4.4 DP 和 AP 读/写访问

- 对 DP 寄存器的读操作没有延迟：调试器将直接获得数据(如果 ACK＝成功)，或者等待(如果 ACK=等待)。
- 对 AP 寄存器的读操作具有延迟。即前一次读操作的结果只能在下一次操作时获得。如果下一次的操作不是对 AP 的访问，则必需读 DP-RDBUFF 寄存器来获得上一次读操作的结果。
- DP-CTRL/STAT 寄存器的 READOK 标志位会在每次 AP 读操作和 RDBUFF 读操作后更新，以通知调试器 AP 的读操作是否成功。
- SW-DP 具有写缓冲区(DP 和 AP 都有写缓冲)，这使得其他传输在进行时，仍然可以接受写操作。如果写缓冲区满，调试器将获得一个等待的 ACK 响应。读 IDCODE 寄存器，读 CTRL/STAT 寄存器和写 ABORT 寄存器操作在写缓冲区满时仍被接受。
- 由于 SWCLK 和 HCLK 的异步性，需要在写操作后(在奇偶校验位后)插入 2 个额外的 SWCLK 周期，以确保内部写操作正确完成。这两个额外的时钟周期需要在线路为低时插入(IDLE 状态下)。这个操作步骤在写 CTRL/STAT 寄存器以提出一个上电请求时尤其重要，否则下一个操作(在内核上电后才有效的操作)会立即执行，这将导致失败。

18.4.5 SW-DP 寄存器

当 APnDP=0 时，可以访问以下这些寄存器。

表 18-4 SW-DP 寄存器

A(3:2)	读/写	SELECT 寄存器的 CTRLSEL 位	寄存器	描述
00	读		IDCODE	固定为 0x0BB11477(用于识别 SW-DP)
00	写		ABORT	
01	读/写	0	DP-CTRL/STAT	－请求一个系统或调试的上电操作 －配置 AP 访问的操作模式 －控制比较，校验操作 －读取一些状态位(溢出，上电响应)
01	读/写	1	WIRE CONTROL	配置串行通信物理层协议(如转换时间长度等)。
10	读		READ RESEND	允许从一个错误的调试传输中恢复数据，而不用重复最初的 AP 传输。
10	写		SELECT	选择当前的访问端口和有效的 4 字长寄存器窗口
11	读/写		READ BUFFER	由于 AP 的访问具有传递性(当前 AP 读操作的结果会在下次 AP 传输时传出)，因此这个寄存器非常必要。 这个寄存器会从 AP 捕获上一次读操作的数据结果，因此可以获得数据而不必再启动一个新的 AP 传输。

18.4.6 SW-AP 寄存器

当 APnDP=1 时，可以访问以下这些寄存器。

AP 寄存器的访问地址由以下两部分组成：

- A[3:2]的值
- DP SELECT 寄存器的当前值

表 18-5 SW-AP 寄存器

Address	A[3:2]value	描述
0x0	00	保留，必须保持重置值。
0x4	01	DP CTRL/STAT 寄存器。用于： - 要求系统或调试开机 - 配置 AP 访问的传输操作 - 控制”推送比较”和”推送验证”操作。 - 读取一些状态标志（超限、上电确认）。
0x8	10	DP SELECT 寄存器：用来选择当前的访问端口和 4 字寄存器窗口。 - 位 31:24: APSEL：选择当前的 AP - 位 23:8: 保留 - 位 7:4: APBANKSEL：选择当前 AP 上的活动的 4 字寄存器窗口。 - 位 3:0: 保留
0xC	11	DP RDBUFF 寄存器： 用于允许调试器在一系列操作后获得最终结果（无需请求新的 SW-DP 操作）

18.5 内核调试

内核调试通过内核调试寄存器访问。 对这些寄存器的调试访问是通过调试访问端口进行的。它由四个寄存器组成：

表 18-6 内核调试寄存器

寄存器	描述
DHCSR	32 位的调试控制和状态寄存器 此寄存器提供内核状态信息，允许内核进入调试模式，和提供单步功能
DCRSR	17 位的内核寄存器调试选择寄存器 此寄存器选择需要进行读写操作的内核寄存器
DCRDR	32 位的内核寄存器调试数据寄存器 此寄存器存放由 DCRSR 选择的内核寄存器读出的或需要写入的数据
DEMCR	32 位异常调试和监视控制寄存器 此寄存器提供向量传输和监视调试控制功能。 TRCENA 位启动 TRACE 功能

注意： 这些寄存器在系统复位时不复位，仅在上电复位时复位。

更多详细资料请参考《Cortex™-M0 技术参考手册》。

为了在复位后立即使内核进入调试状态，需要：

- 使能调试和异常监视控制寄存器(Debug and Exception Monitor Control Register)的位 0(VC_CORRESET)。
- 使能调试控制和状态寄存器(Debug Halting Control and Status Register)的位 0(C_DEBUGEN)。

18.6 BPU (Break Point Unit)

Cortex-M0 BPU 实现提供了四个断点寄存器。 BPU 是 ARMv7~M（Cortex-M3 和 Cortex-M4）中可用的闪存补丁和断点(FPB)块的子集。

18.6.1 BPU 功能

处理器断点实现基于 PC 的断点功能。
有关 BPU CoreSight 标识寄存器及其地址和访问类型的详细信息，请参阅《ARMv6-M 架构参考手册》和《ARM CoreSight 技术参考手册》。

18.7 DWT (数据观察点触发 data watchpoint trigger)

Cortex-M0 DWT 实现提供了两个观察点寄存器集。

18.7.1 DWT 功能

处理器观察点实现了数据地址和基于 PC 的观察点功能、PC 采样寄存器，并支持比较器地址屏蔽，如《ARMv6-M 架构参考手册》中所述。

18.7.2 DWT 程序计数器采样寄存器

实现数据观察点单元的处理器还实现了 ARMv6-M 可选 DWT 程序计数器采样寄存器(DWT_PCSR)。
该寄存器允许调试器在不停止处理器的情况下定期对 PC 进行采样。 这提供了粗粒度的分析。 有关详细信息，请参阅《ARMv6-M 架构参考手册》。
Cortex-M0 DWT_PCSR 记录通过其条件代码和失败的指令。

18.8 MCU 调试模块

MCU 调试模块协助调试器提供以下功能：

- 在断点时提供高级定时器控制
- 在断点时提供基本定时器控制
- 在断点时提供看门狗控制

这些功能可在各个外设功能描述的“调试模式”描述中获得详细信息。

19 版本历史

表 7.1 文档版本历史

日期	版本	变更
2022-02-27	1.0	初始发行
2022-06-15	1.1	1. 优化一些细节描述 2. 16.3.2.1 访问时间调节器“补充一条注意” 系统时钟频率高于 24Mhz 时, WAIT 必须被设置为 1 周期等待； ” 3. 修正“TIM1 状态寄存器(TIM1_SR)”中，ICxR 和 ICxF 的顺序 4. 修正 3.5.1 电源电压监测器配置寄存器 PWR_PVDR 中，关于 PVD 挡位的错误描述
2022-10-13	1.2	1. 优化一些细节描述 2. 修正了一些错误
2023-5-10	1.3	1. 优化一些细节描述 2. 修正了一些错误