高性能 RISC CPU

- **49 条指令**
- **指令周期可选：1T / 2T / 4T**
- **存储架构**
  - ✓ 程序 ROM: 8k x 14 bits
  - ✓ 数据 RAM: 1k x 8 bits
  - ✓ 数据 EEPROM: 256 x 8 bits
  - ✓ 扇区程序代码保护 (扇区=1k x 14bits)
  - ✓ 支持 IAP
- **16 层硬件堆栈**

特殊单片机特性

- **工作温度范围：-40 ─85°C**
- **宽工作电压范围：1.9 ─ 5.5V**
  - ✓ DC – 8MHz: VDD ≥ 1.9V
  - ✓ DC – 16MHz: VDD ≥ 2.7V
- **时钟源**
  - ✓ 16MHz 高速高精度 HIRC
  - ✓ 32kHz 低速低功耗 LIRC
  - ✓ 晶体振荡器和外部时钟输入
  - ✓ 晶体时钟缺失检测
  - ✓ 晶体时钟配置下的双速时钟启动
  - ✓ 慢时钟周期测量
- **带 7 位预分频的 16 位看门狗，时钟源可选**
- **上电复位延时计数器**
- **低功耗模式 SLEEP**
  - ✓ 系统时钟可选择保持运行或关闭
- **低电压复位 LVR:**
  - ✓ 2.0/2.2/2.5/2.8/3.1/3.6/4.1V
- **低电压检测 LVD:**
  - ✓ 内部电压：2.0/2.4/2.8/3.0/3.6/4.0V
  - ✓ 或检测外部输入，可当比较器用
- **支持 ISP 和在线调试 OCD**
  - ✓ 3 个硬件断点
  - ✓ 软复位，单步，暂停，跳跃等
- **封 装 形 式： SOP20，TSSOP20，SOP28，LQFP32**

外设特性

- **GPIO**
  - ✓ 30 个方向独立控制的通用 IO
  - ✓ 30 个唤醒管脚：边沿或电平检测
  - ✓ 30 个带上拉功能的管脚，独立控制
  - ✓ 30 个带下拉功能的管脚，独立控制
  - ✓ 30 个可编程源电流管脚：2/4/14/26mA@5V
  - ✓ 30 个可编程灌电流管脚：max. 62mA@5V
  - ✓ 支持管脚第二功能的重映射
- **通信接口**
  - ✓ USART
  - ✓ I2C，主从机
  - ✓ SPI，主从机
- **12 bit SAR ADC (x1)**
  - ✓ 8 个外部通道(AN0 – AN7) + 1 个¼ VDD 通道
  - ✓ 内部参考电压：VDD, 0.5V, 2V, 3V
  - ✓ 外部参考：VREFP, VREFN
  - ✓ 手动和自动触发方式
  - ✓ 支持延时触发
  - ✓ 支持自动校准
- **TIM1-16bit**
  - ✓ 带 16 位预分频的 16 位定时器
  - ✓ 自动重载
  - ✓ 时钟源：系统时钟，HIRC 以及倍频时钟（晶体或 HIRC 的二倍频），LIRC
  - ✓ 周期、占空比寄存器双缓冲设计
  - ✓ 4 个独立的捕捉/比较/PWM 通道
  - ✓ PWM 支持沿对齐，中心对齐，单次脉冲模式
  - ✓ 3 组带死区控制的互补 PWM 输出
  - ✓ 前沿消隐
  - ✓ 故障刹车控制
- **TIM2-16bit**
  - ✓ 带 15 位预分频的 16 位定时器
  - ✓ 自动重载
  - ✓ 时钟源：系统时钟，HIRC 以及倍频时钟（晶体或 HIRC 的二倍频），LIRC
  - ✓ 周期、占空比寄存器双缓冲设计
  - ✓ 3 个独立的捕捉/比较/PWM 通道
- **TIM4-8bit，带 8bit 预分频的基本定时器，时钟源可选**
- **触摸感应模块**
  - ✓ 15 个按键通道
  - ✓ 扫描次数可配
  - ✓ 防水功能

目录

1. 系统功能框图及脚位

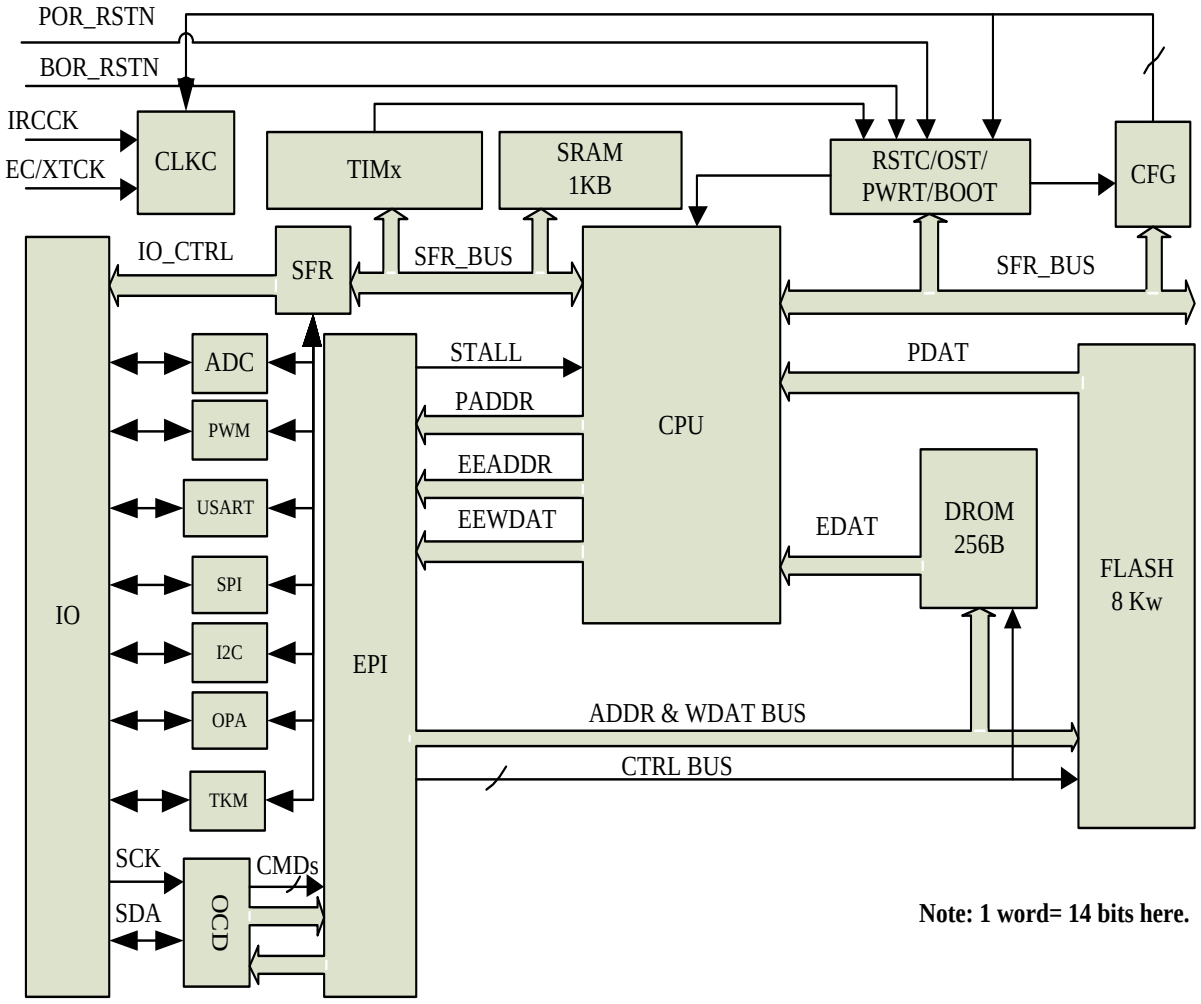


图 1.1 芯片整体功能框图

1.1. 脚位图

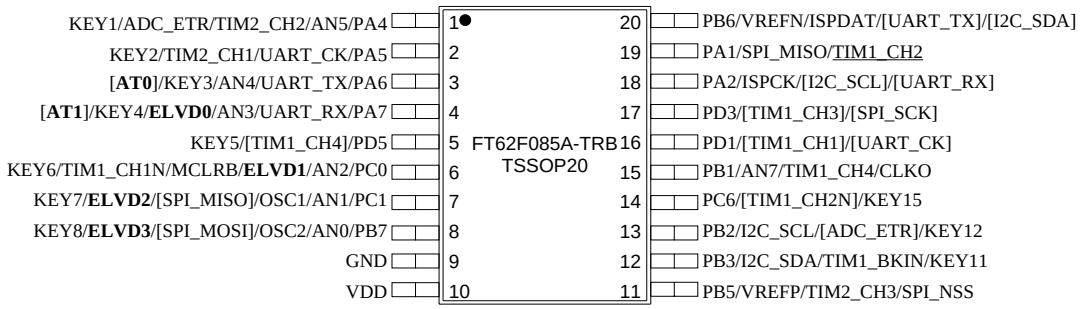


图 1.2 TSSOP20 脚位

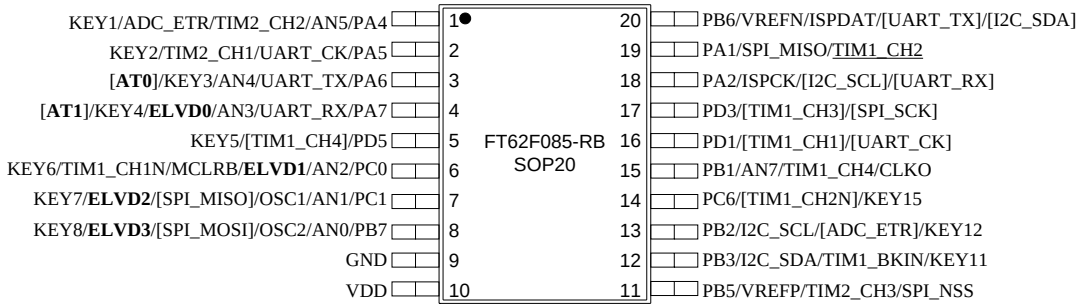


图 1.3 SOP20 脚位

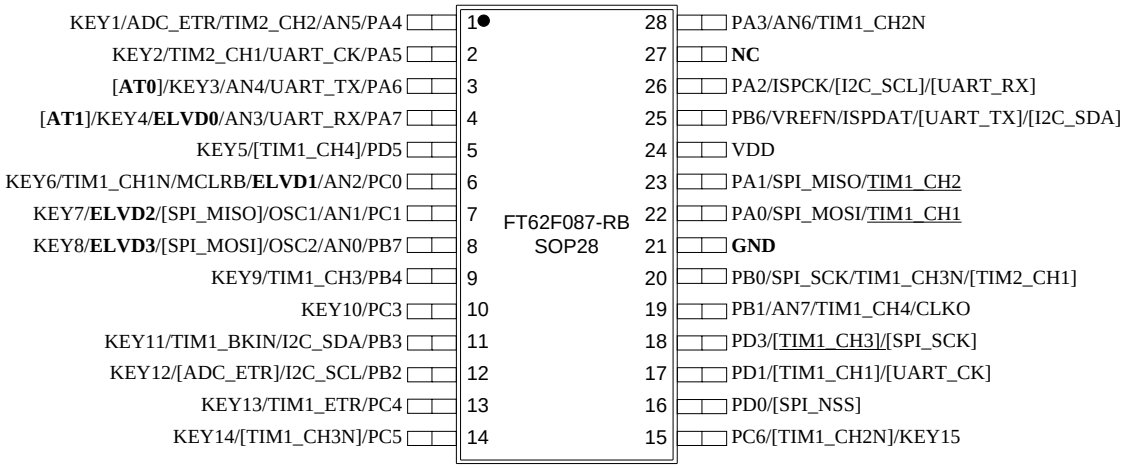


图 1.4 SOP28 脚位

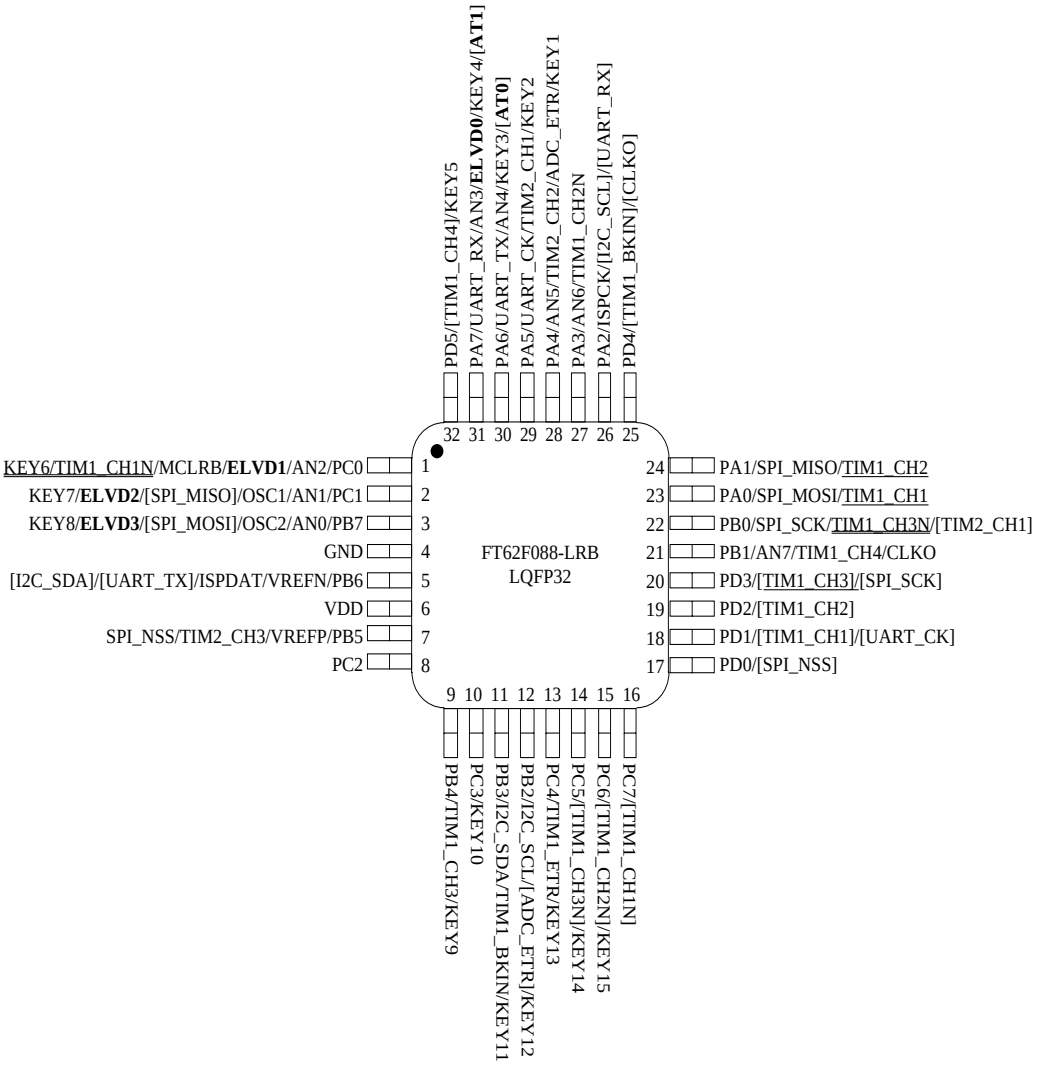


图 1.5 LQFP-32 脚位

1.2. 管脚描述

LQFP32	Pin name	Type	INT input	HS output	Main func.	Default AF
1	PC0/AN2/MCLRB/TIM1_CH1N/**ELVD1**/KEY6	IO	√	√	PC0	TIM1_CH1N
2	PC1/AN1/OSC1/[SPI_MISO]/**ELVD2**/KEY7	IO	√	√	PC1	SPI_MISO
3	PB7/AN0/OSC2/[SPI_MOSI]/**ELVD3**/KEY8	IO	√	√	PB7	SPI_MOSI
4	GND	Ground	—	√	Ground	
5	PB6/ISPDAT/[I2C_SDA]/[UART_TX]	IO	√	√	PB6	I2C_SDA
6	VDD	Supply	—	√	Power	
7	PB5/TIM2_CH3/SPI_NSS	IO	√	√	PB5	TIM2_CH3
8	PC2	IO	√	√	PC2	—
9	PB4/TIM1_CH3/KEY9	IO	√	√	PB4	TIM1_CH3
10	PC3/KEY10	IO	√	√	PC3	—
11	PB3/I2C_SDA/TIM1_BKIN/KEY11	IO	√	√	PB3	I2C_SDA
12	PB2/I2C_SCL/[ADC_ETR]/KEY12	IO	√	√	PB2	I2C_SCL
13	PC4/TIM1_ETR/KEY13	IO	√	√	PC4	TIM1_ETR
14	PC5/[TIM1_CH3N]/KEY14	IO	√	√	PC5	TIM1_CH3N
15	PC6/[TIM1_CH2N]/KEY15	IO	√	√	PC6	TIM1_CH2N
16	PC7/[TIM1_CH1N]	IO	√	√	PC7	TIM1_CH1N
17	PD0/[SPI_NSS]	IO	√	√	PD0	SPI_NSS
18	PD1/[TIM1_CH1]/[UART_CK]	IO	√	√	PD1	TIM1_CH1
19	PD2/[TIM1_CH2]	IO	√	√	PD2	TIM1_CH2
20	PD3/[TIM1_CH3]/[SPI_SCK]	IO	√	√	PD3	TIM1_CH3
21	PB1/TIM1_CH4/CLKO/AN7	IO	√	√	PB1	TIM1_CH4
22	PB0/SPI_SCK/TIM1_CH3N/[TIM2_CH1]	IO	√	√	PB0	SPI_SCK
23	PA0/SPI_MOSI/TIM1_CH1	IO	√	√	PA0	SPI_MOSI
24	PA1/SPI_MISO/TIM1_CH2	IO	√	√	PA1	SPI_MISO
25	PD4/[TIM1_BKIN]/[CLKO]	IO	√	√	PD4	TIM1_BKIN
26	PA2/ISPCK/[I2C_SCL]/[UART_RX]	IO	√	√	PA2	I2C_SCL
27	PA3/AN6/TIM1_CH2N	IO	√	√	PA3	AN6
28	PA4/AN5/TIM2_CH2/ADC_ETR/KEY1	IO	√	√	PA4	AN5
29	PA5/VREF/UART_CK/TIM2_CH1/KEY2	IO	√	√	PA5	UART_CK
30	PA6/UART_TX/AN4/[AT0]/KEY3	IO	√	√	PA6	UART_TX
31	PA7/UART_RX/AN3/**ELVD0**/[AT1]/KEY4	IO	√	√	PA7	UART_RX
32	PD5/[TIM1_CH4]/KEY5	IO	√	√	PD5	TIM1_CH4

2. 程序存储器

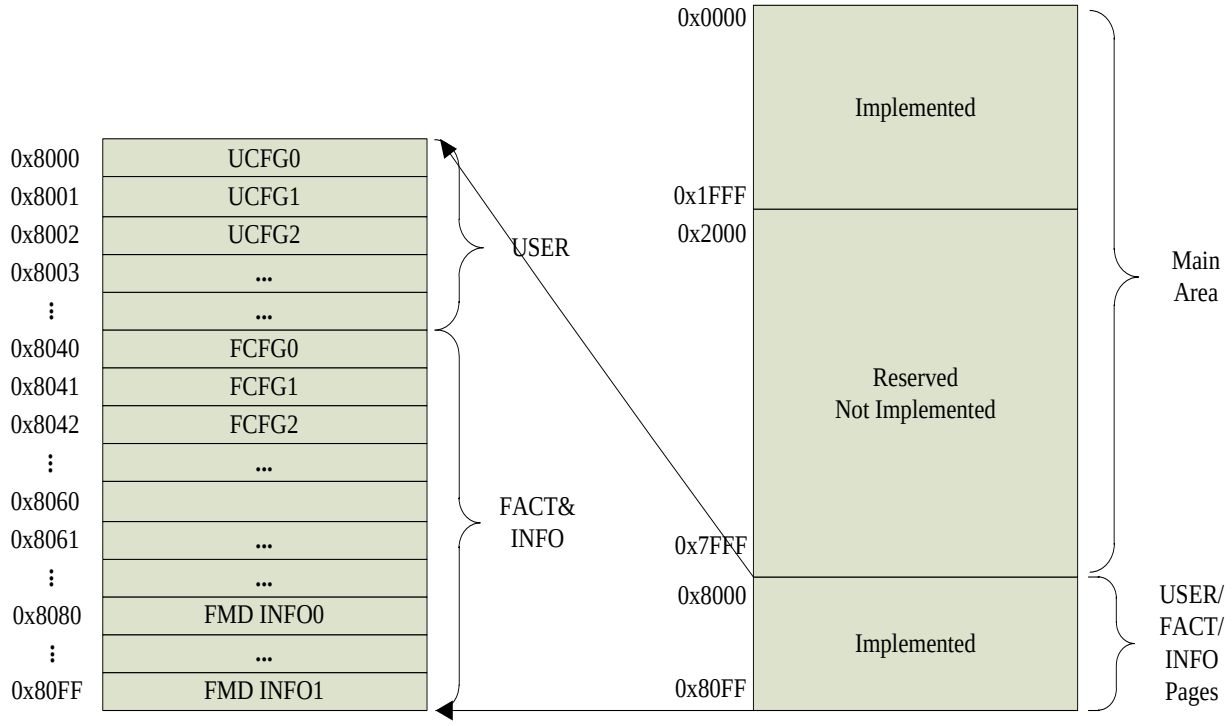


图 1.4 程序存储区地址空间

程序地址计数器 PC 为 15 位(0x0000 ~ 0x7FFF)，最多支持 32K 地址空间。芯片实现了 8K 的程序存储器（0x0000 ~ 0x1FFF），外加上 2 个额外的用户配置区，工厂配置区，以及 2 个信息区 INFO0/INFO1。

8K 程序存储器由 128 页组成，每页 64 个 word（1word= 14bits），地址范围为 0x0000~0x1FFF，当程序地址超过 0x1FFF 将导致回卷到 0x0000。

另外的 4 个 NVM 区分别占一个单独的页，每页 64 word，其编址从 0x8000 开始，到 0x80FF 结束。

2.1. 将程序存储器当作数据存储器读取

有两种方法可访问程序存储器中的常数。第一种方法是使用 RETW 指令表。第二种方法是设置 FSR 指向程序存储器。

2.1.1. RETW 指令

RETW 指令用于提供对常数表的访问。

例 2.1 给出了创建这种表的推荐方法。

BRW 指令使得这种类型的表实现起来非常简单。如果代码必须保持与前几代单片机之间的可移植性，则 BRW 指令不适用，因此必须使用较老的表读取方法。

```
Constants ; example 2.1
BRW ;Add Index in W to program counter to ;select data
RETW DATA0 ;Index0 data
RETW DATA1 ;Index1 data
RETW DATA2
RETW DATA3
my_function
;… LOTS OF CODE…
LDWI DATA_INDEX
call constants       : THE CONSTANT IS IN W
```

2.1.2. 使用 FSR 间接读取

可将程序存储器当作数据存储器进行访问，方法是将 FSRxH 寄存器的 bit7 置 1 并读取与之配对的 INDFx 寄存器。MOVIW 指令会将已寻址到的字的低 8 位保存在 W 寄存器中。无法通过 INDF 寄存器执行写程序存储器操作。通过 FSR 访问程序存储器的指令需要一个额外的指令周期才能完成。例 2.2 演示了通过 FSR 访问程序存储器的过程。

如果标号指向程序存储器中的单元，HIGH 伪指令将 bit<7>置 1。

```
constants     ; example 2.2
RETW DATA0 ; Index0 data
RETW DATA1 ; Index1 data
RETW DATA2
RETW DATA3
my_function
;… LOTS OF CODE…
LDWI LOW constants
STR FSR1L
LDWI HIGH constants
STR FSR1H
MOVIW 0[FSR1] ; THE PROGRAM MEMORY IS IN W
```

3. 数据存储器

3.1. 数据存储器构成

数据存储器划分为 32 存储区，每个存储区大小为 128 个字节，每个存储区由以下部分组成：

- 12 个内核寄存器
- 最多 20 个特殊功能寄存器
- 最多 80 字节的通用 RAM
- 16 字节的共用 RAM

可通过将存储区号写入存储区选择寄存器（Bank Select Register，BSREG）来选择有效存储区。未实现的存储器将读为 0。所有的数据存储器都可以直接访问（通过使用文件寄存器的指令），或通过 2 个文件选择寄存器（FSR）间接访问。更多信息，请参见第 3.4 节“间接寻址”。

数据存储器使用一个 12 位地址。地址的高 5 位用于定义存储区地址，低 7 位用于选择该存储区中的寄存器/RAM。

存储区（页）划分：

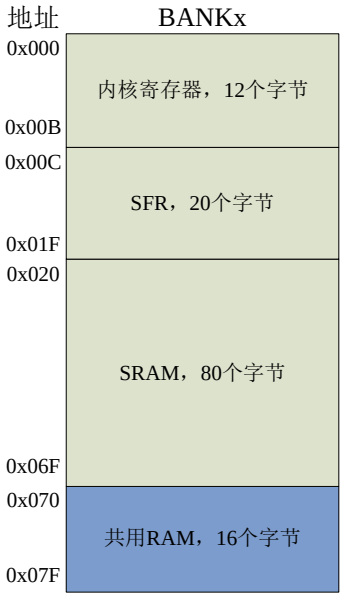


图 3.1 存储区构成

3.1.1. 内核寄存器

内核寄存器包含直接影响 MCU 基本操作的寄存器。这些寄存器包括：

- INDF0
- INDF1
- PCL
- STATUS
- FSR0 低字节
- FSR0 高字节
- FSR1 低字节
- FSR1 高字节
- BSREG
- WREG
- PCLATH
- INTCON

注：内核寄存器位于数据存储器每个存储区的前 12 个地址单元。

3.1.2. 状态寄存器

如寄存器 3.1 所示，状态（STATUS）寄存器包含：

- ALU 的算术运算状态
- 复位状态
- 数据存储器（SRAM）的存储区选择位

和其他寄存器一样，状态寄存器也可以作为任何指令的目标寄存器。如果一条影响 Z、DC 或 C 位的指令以状态寄存器作为目标寄存器，将禁止写这三位。根据器件逻辑，这些位会被置 1 或清零。 此外，也不能写 TO 和 PD 位。因此，当执行一条把状态寄存器作为目标寄存器的指令后，状态寄存器的结果可能和预想的不一样。

例如，执行 CLRR STATUS 指令会清零该寄存器的高 3 位并将 Z 位置 1。从而使状态寄存器的值为“000uu1uu”（其中 u 表示不变）。

因此，建议仅使用 BCR、BSR、SWAPR 和 STR 指令来改变状态寄存器，因为这些指令不影响任何状态位。欲知其他不会影响任何状态位的指令，请参见“指令集汇总”。

寄存器 3.1 STATUS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—			/TO	/PD	Z	DC	C
Reset	NA			1	1	x	x	x
Type	RO-0			RO	RO	RW	RW	RW

Bit	Name	Function
7:5	NA	**未实现，读 0**
4	/TO	超时标志位 1 = 在上电后，执行了CLRWDT指令或SLEEP指令 0 = 发生了 WDT 超时
3	/PD	掉电标志位 1 = 上电复位后或执行了CLRWDT指令 0 = 执行了 SLEEP 指令
2	Z	零状态位 1 = 算术运算或逻辑运算结果为零 0 = 算术运算或逻辑运算结果不为零
1	DC	半进位/ 半借位标志位（ADDWR、ADDWI、SUBWI 和SUBWR指令） (1) 1 = 结果的第4个低位向高位发生了进位 0 = 结果的第 4 个低位未向高位发生进位
0	C	进位/借位标志位(1) （ADDWR、ADDWI、SUBWI 和SUBWR指令） (1) 1 = 结果的最高位发生了进位 0 = 结果的最高位未发生进位

注意：

1. 对于借位 C，极性是相反的。减法指令通过加上第二个操作数的二进制补码实现。对于移位指令（RRF 和 RLF），此位值来自源寄存器的最高位或最低位。

3.2. 特殊功能寄存器

特殊功能寄存器是应用程序用来控制器件中外设功能所需操作的寄存器。与外设操作有关的寄存器将在本数据手册中的相应外设章节中讲述。

3.2.1. 通用 RAM

数据存储器的每个存储区中的 GPR 最多有 80 个字节。

3.2.2. GPR 的线性访问

可以通过 FSR 以非分区方法访问通用 RAM。这可以简化对大存储器结构的访问。更多信息，请参见第 3.6.2 节“线性数据存储器”。

3.2.3. 公共 RAM

从所有存储区都可以访问 16 字节的公共 RAM。

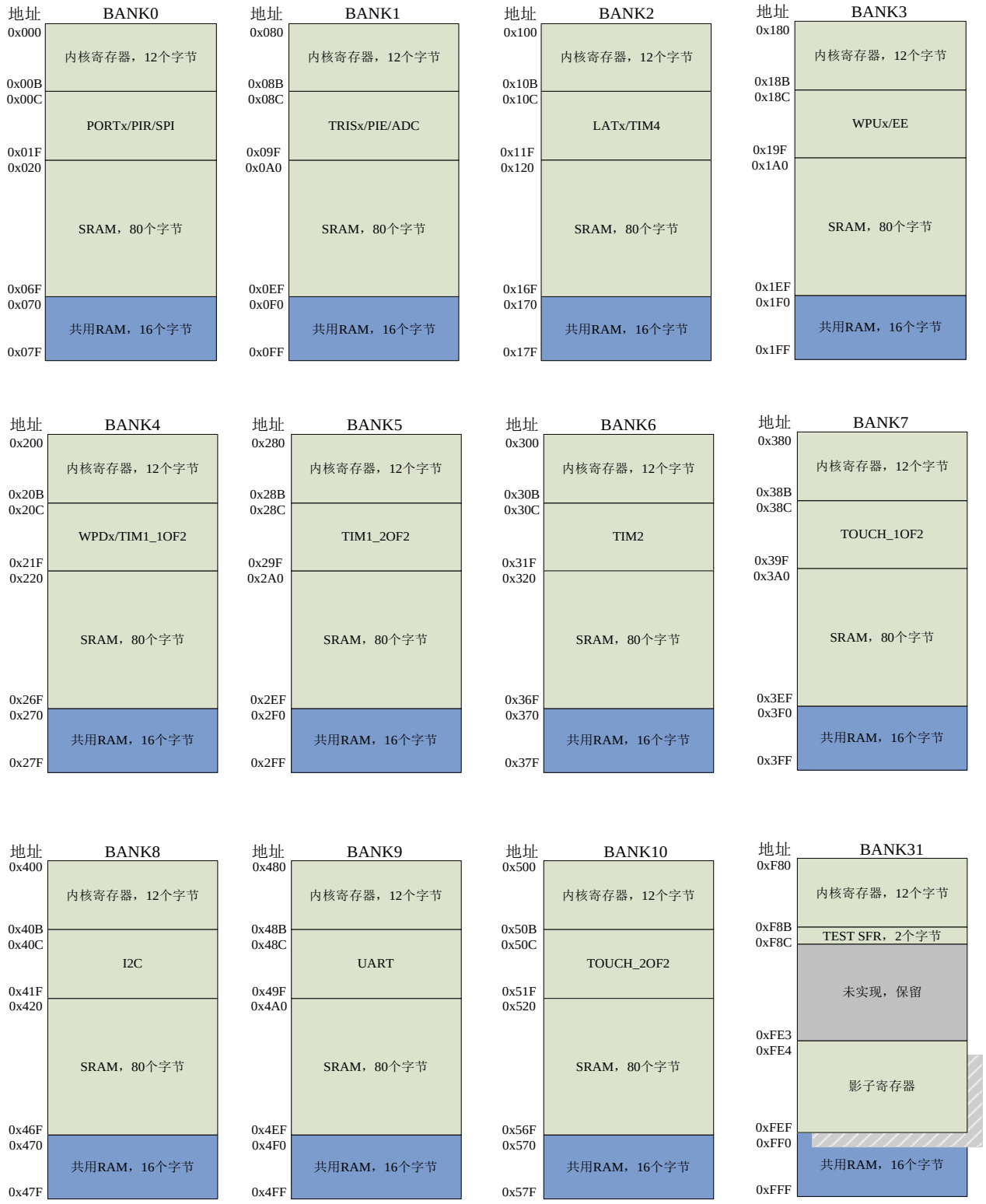


图 3.2 存储区地址映射

3.2.4. 存储区地址映射

BANK0~BANK5

地址	BANK0	地址	BANK1	地址	BANK2	地址	BANK3	地址	BANK4	地址	BANK5
000h	INDF0	080h	INDF0	100h	INDF0	180h	INDF0	200h	INDF0	280h	INDF0
001h	INDF1	081h	INDF1	101h	INDF1	181h	INDF1	201h	INDF1	281h	INDF1
002h	PCL	082h	PCL	102h	PCL	182h	PCL	202h	PCL	282h	PCL
003h	STATUS	083h	STATUS	103h	STATUS	183h	STATUS	203h	STATUS	283h	STATUS
004h	FSR0L	084h	FSR0L	104h	FSR0L	184h	FSR0L	204h	FSR0L	284h	FSR0L
005h	FSR0H	085h	FSR0H	105h	FSR0H	185h	FSR0H	205h	FSR0H	285h	FSR0H
006h	FSR1L	086h	FSR1L	106h	FSR1L	186h	FSR1L	206h	FSR1L	286h	FSR1L
007h	FSR1H	087h	FSR1H	107h	FSR1H	187h	FSR1H	207h	FSR1H	287h	FSR1H
008h	BSREG	088h	BSREG	108h	BSREG	188h	BSREG	208h	BSREG	288h	BSREG
009h	WREG	089h	WREG	109h	WREG	189h	WREG	209h	WREG	289h	WREG
00Ah	PCLATH	08Ah	PCLATH	10Ah	PCLATH	18Ah	PCLATH	20Ah	PCLATH	28Ah	PCLATH
00Bh	INTCON	08Bh	INTCON	10Bh	INTCON	18Bh	INTCON	20Bh	INTCON	28Bh	INTCON
00Ch	PORTA	08Ch	TRISA	10Ch	LATA	18Ch	WPUA	20Ch	WPDA	28Ch	TIM1CNTRH
00Dh	PORTB	08Dh	TRISB	10Dh	LATB	18Dh	WPUB	20Dh	WPDB	28Dh	TIM1CNTRL
00Eh	PORTC	08Eh	TRISC	10Eh	LATC	18Eh	WPUC	20Eh	WPDC	28Eh	TIM1PSCRH
00Fh	PORTD	08Fh	TRISD	10Fh	LATD	18Fh	WPUD	20Fh	WPDD	28Fh	TIM1PSCRL
010h	—	090h	—	110h	—	190h	—	210h	—	290h	TIM1ARRH
011h	PIR1	091h	PIE1	111h	**TIM4CR1**	191h	EEADRL	211h	TIM1CR1	291h	TIM1ARRL
012h	—	092h	—	112h	TIM4IER	192h	EEADRH	212h	—	292h	TIM1RCR
013h	—	093h	—	113h	TIM4SR	193h	EEDATL	213h	TIM1SMCR	293h	TIM1CCR1H
014h	EPIF0	094h	EPIE0	114h	TIM4EGR	194h	EEDATH	214h	—	294h	TIM1CCR1L
015h	SPIDATA	095h	CKOCON	115h	TIM4CNTR	195h	EECON1	215h	TIM1IER	295h	TIM1CCR2H
016h	SPICTRL	096h	PCON	116h	TIM4PSCR	196h	EECON2	216h	TIM1SR1	296h	TIM1CCR2L
017h	SPICFG	097h	WDTCON	117h	TIM4ARR	197h	ANSELA	217h	TIM1SR2	297h	TIM1CCR3H
018h	SPISCR	098h	OSCTUNE	118h	EPS0	198h	**EECON3**	218h	TIM1EGR	298h	TIM1CCR3L
019h	SPICRCPOL	099h	OSCCON	119h	EPS1	199h	LVDCON	219h	TIM1CCMR1	299h	TIM1CCR4H
01Ah	SPIRXCRC	09Ah	PCKEN	11Ah	PSRC0	19Ah	PSINK0	21Ah	TIM1CCMR2	29Ah	TIM1CCR4L
01Bh	SPITXCRC	09Bh	ADRESL	11Bh	PSRC1	19Bh	PSINK1	21Bh	TIM1CCMR3	29Bh	TIM1BKR
01Ch	SPIIER	09Ch	ADRESH	11Ch	MISC0	19Ch	PSINK2	21Ch	TIM1CCMR4	29Ch	TIM1DTR
01Dh	SPICTRL2	09Dh	ADCON0	11Dh	AFP2	19Dh	PSINK3	21Dh	TIM1CCER1	29Dh	TIM1OISR
01Eh	SPISTAT	09Eh	ADCON1	11Eh	ITYPE0	19Eh	AFP0	21Eh	TIM1CCER2	29Eh	TIM2CCR3H
01Fh	ADDLY	09Fh	ADCON2	11Fh	ITYPE1	19Fh	AFP1	21Fh	ODCON0	29Fh	TIM2CCR3L
020~06F	GPR, 80B	0A0~0EF	GPR, 80B	120~16F	GPR, 80B	1A0~1EF	GPR, 80B	220~26F	GPR, 80B	2A0~2EF	GPR, 80B
070~07F	公共 RAM	0F0~0FF	公共 RAM	170~17F	公共 RAM	1F0~1FF	公共 RAM	270~27F	公共 RAM	2F0~2FF	公共 RAM

BANK6~BANK10

地址	BANK6	地址	BANK7	地址	BANK8	地址	BANK9	地址	BANK10	地址	BANK31
300h	INDF0	380h	INDF0	400h	INDF0	480h	INDF0	500h	INDF0	F80h	INDF0
301h	INDF1	381h	INDF1	401h	INDF1	481h	INDF1	501h	INDF1	F81h	INDF1
302h	PCL	382h	PCL	402h	PCL	482h	PCL	502h	PCL	F82h	PCL
303h	STATUS	383h	STATUS	403h	STATUS	483h	STATUS	503h	STATUS	F83h	STATUS
304h	FSR0L	384h	FSR0L	404h	FSR0L	484h	FSR0L	504h	FSR0L	F84h	FSR0L
305h	FSR0H	385h	FSR0H	405h	FSR0H	485h	FSR0H	505h	FSR0H	F85h	FSR0H
306h	FSR1L	386h	FSR1L	406h	FSR1L	486h	FSR1L	506h	FSR1L	F86h	FSR1L
307h	FSR1H	387h	FSR1H	407h	FSR1H	487h	FSR1H	507h	FSR1H	F87h	FSR1H
308h	BSREG	388h	BSREG	408h	BSREG	488h	BSREG	508h	BSREG	F88h	BSREG
309h	WREG	389h	WREG	409h	WREG	489h	WREG	509h	WREG	F89h	WREG
30Ah	PCLATH	38Ah	PCLATH	40Ah	PCLATH	48Ah	PCLATH	50Ah	PCLATH	F8Ah	PCLATH
30Bh	INTCON	38Bh	INTCON	40Bh	INTCON	48Bh	INTCON	50Bh	INTCON	F8Bh	INTCON
30Ch	TIM2CR1	38Ch	TKTMR	40Ch	I2CCR1	48Ch	URDATAL	50Ch	—	F8Ch	—
30Dh	TIM2IER	38Dh	TKC0	40Dh	I2CCR2	48Dh	URDATAH	50Dh	—	F8Dh	—
30Eh	TIM2SR1	38Eh	TKC1	40Eh	I2CCR3	48Eh	URIER	50Eh	—	F8Eh	CF0OUT1L
30Fh	TIM2SR2	38Fh	WProof1	40Fh	I2COARL	48Fh	URLCR	50Fh	—	F8Fh	CF0OUT1H
310h	TIM2EGR	390h	WProof2	410h	I2COARH	490h	URLCREXT	510h	—	F90h	CF0OUT2L
311h	TIM2CCMR1	391h	WProof3	411h	I2CFREQ	491h	URMCR	511h	—	F91h	CF0OUT2H
312h	TIM2CCMR2	392h	M0analog	412h	I2CDR	492h	URLSR	512h	—	F92h	CF0OUT3L
313h	TIM2CCMR3	393h	M1analog	413h	I2CCMD	493h	URRAR	513h	—	F93h	CF0OUT3H
314h	TIM2CCER1	394h	M2analog	414h	I2CCCRL	494h	URDLL	514h	—	F94h	TKM016DL
315h	TIM2CCER2	395h	M3analog	415h	I2CCCRH	495h	URDLH	515h	—	F95h	TKM016DH
316h	TIM2CNTRH	396h	TKM0C0	416h	I2CITR	496h	URABCR	516h	—	F96h	CF1OUT1L
317h	TIM2CNTRL	397h	TKM0C1	417h	I2CSR1	497h	URSYNCR	517h	—	F97h	CF1OUT1H
318h	TIM2PSCR	398h	TKM1C0	418h	I2CSR2	498h	URLINCR	518h	—	F98h	CF1OUT2L
319h	TIM2ARRH	399h	TKM1C1	419h	I2CSR3	499h	URSDCR0	519h	—	F99h	CF1OUT2H
31Ah	TIM2ARRL	39Ah	TKM2C0	41Ah	ADCON3	49Ah	URSDCR1	51Ah	—	F9Ah	CF1OUT3L
31Bh	TIM2CCR1H	39Bh	TKM2C1	41Bh	ADCMPH	49Bh	URSDCR2	51Bh	—	F9Bh	CF1OUT3H
31Ch	TIM2CCR1L	39Ch	TKM3C0	41Ch	LEBCON	49Ch	URTC	51Ch	—	F9Ch	TKM116DL
31Dh	TIM2CCR2H	39Dh	TKM3C1	41Dh	MSCKCON	49Dh	—	51Dh	—	F9Dh	TKM116DH
31Eh	TIM2CCR2L	39Eh	—	41Eh	SOSCPRL	49Eh	—	51Eh	—	F9Eh	CF2OUT1L
31Fh	TCKSRC	39Fh	—	41Fh	SOSCPRH	49Fh	—	51Fh	—	F9Fh	CF2OUT1H
320~36F	GPR, 80B	3A0~3EF	GPR, 80B	420~46F	GPR, 80B	4A0~4EF	GPR, 80B	520~56F	GPR, 80B	FE4~FEF	Shadow reg
370~37F	公共 RAM	3F0~3FF	公共 RAM	470~47F	公共 RAM	4F0~4FF	公共 RAM	570~57F	公共 RAM	FF0~FFF	公共 RAM

3.2.5. Bank11，bank12 和 bank31

地址	BANK11	地址	BANK12	地址	BANK31	地址	BANK31
580h	INDF0	600h	INDF0	F90h	CF0OUT2L	FA7h	CF3OUT1H
581h	INDF1	601h	INDF1	F91h	CF0OUT2H	FA8h	CF3OUT2L
582h	PCL	602h	PCL	F92h	CF0OUT3L	FA9h	CF3OUT2H
583h	STATUS	603h	STATUS	F93h	CF0OUT3H	FAAh	CF3OUT3L
584h	FSR0L	604h	FSR0L	F94h	TKM016DL	FABh	CF3OUT3H
585h	FSR0H	605h	FSR0H	F95h	TKM016DH	FACh	TKM316DL
586h	FSR1L	606h	FSR1L	F96h	CF1OUT1L	FADh	TKM316DH
587h	FSR1H	607h	FSR1H	F97h	CF1OUT1H	…	—
588h	BSREG	608h	BSREG	F98h	CF1OUT2L	FE3h	—
589h	WREG	609h	WREG	F99h	CF1OUT2H	FE4h	STATUS_SHAD
58Ah	PCLATH	60Ah	PCLATH	F9Ah	CF1OUT3L	FE5h	WREG_SHAD
58Bh	INTCON	60Bh	INTCON	F9Bh	CF1OUT3H	FE6h	BSREG_SHAD
58Ch	—	60Ch	—	F9Ch	TKM116DL	FE7h	PCLATH_SHAD
…	—	…	—	F9Dh	TKM116DH	FE8h	FSR0L_SHAD
59Fh	—	61Fh	—	F9Eh	CF2OUT1L	FE9h	FSR0H_SHAD
5A0h	**GPR, 80B**	620h	**GPR, 48B**	F9Fh	CF2OUT1H	FEAh	FSR1L_SHAD
5A1h		…		FA0h	CF2OUT2L	FEBh	FSR1H_SHAD
5A2h		64Fh		FA1h	CF2OUT2H	FECh	—
5A3h		650h	—	FA2h	CF2OUT3L	FEDh	STKPTR
5A4h		…	—	FA3h	CF2OUT3H	FEEh	TOSL
…		…	—	FA4h	TKM216DL	FEFh	TOSH
5EFh		**66Fh**	—	FA5h	TKM216DH	FF0h ~ FFFh	公共 RAM
5F0~5FF	公共 RAM	670~67F	公共 RAM	FA6h	CF3OUT1L		

3.2.6. Bank31 的影子寄存器

CPU 在进入中断时，硬件会自动把 W 寄存器，STATUS 寄存器（TO 和 PD 状态标志位除外），BSR 寄存器，FSR 寄存器以及 PCLATH 寄存器保存到处于 bank31 的影子寄存中，在退出中断时再把它们恢复到对应的寄存器，节省软件的开销。

地址	寄存器名称
FE4h	STATUS_SHAD
FE5h	WREG_SHAD
FE6h	BSREG_SHAD
FE7h	PCLATH_SHAD
FE8h	FSR0L_SHAD
FE9h	FSR0H_SHAD
FEAh	FSR1L_SHAD
FEBh	FSR1H_SHAD
FECh	—
FEDh	STKPTR
FEEh	TOSL
FEFh	TOSH

3.3. 堆栈

FT62F08X 有一个 16 级深 x15 位宽的硬件堆栈。堆栈空间不属于程序存储空间或数据存储空间的一部分。当执行 LCALL 或 CALLW 指令或由于中断导致程序跳转时，PC 的值会被压入堆栈。当执行 RET、RETW 或 RETI 指令时，PC 值从堆栈弹出。PCLATH 的值不受压栈或出栈操作的影响。

连续压栈 17 次后将产生上溢，上溢标志位 STKOVF 上溢置 1。

同理，如果弹栈次数大于压栈次数，则产生下溢，标志位 STKUNF 将被置 1，无论是上溢还是下溢都将导致一次系统复位，**而且 16 级的堆栈将全部清 0**。

3.3.1. 访问堆栈

可通过 TOSH、TOSL 和 STKPTR 寄存器来使用堆栈。STKPTR 是堆栈指针的当前值。TOSH:TOSL 寄存器对指向栈顶。这两个寄存器都可读写。由于 PC 的大小为 15 位，故 TOS 划分为 TOSH 和 TOSL 两部分。要访问堆栈，可调整用来定位 TOSH:TOSL 的 STKPTR 值，然后对 TOSH:TOSL 执行读/写操作。STKPTR 为 5 位，允许检测上溢和下溢。

在正常程序运行期间，LCALL、CALLW 和中断会使 STKPTR 值递增 1，而 RETW、RET 和 RETI 会使 STKPTR 值递减 1。任何时候都可以检查 STKPTR，以查看可用堆栈空间。STKPTR 总是指向堆栈中的当前使用单元。因此，LCALL 或 CALLW 指令会使 STKPTR 值递增 1，然后写 PC，而返回操作则会卸载 PC，然后使 STKPTR 值递减 1。

注意：在允许中断的情况下修改 STKPTR 时应谨慎。

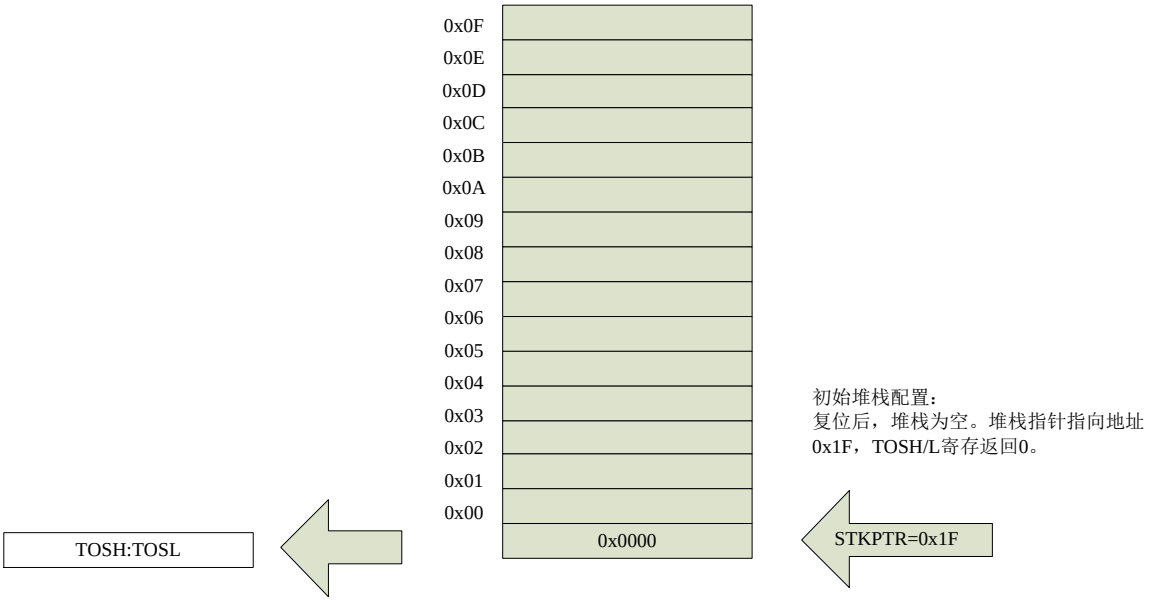


图 3.3 软件访问堆栈

3.3.2. 上溢/下溢复位

在压满 16 级后再执行压栈操作，或者在弹出第 1 级后再执行出栈操作，PCON 寄存器中的相应位（分别为 STKOVF 或 STKUNF）会置 1，从而使器件复位。

3.4. 间接寻址

INDFn 寄存器不是物理寄存器。任何访问 INDFn 寄存器的指令，实际上都是访问由文件选择寄存器（FSR）指定的地址处的寄存器。如果 FSRn 地址指定了 2 个 INDFn 寄存器中的任何一个，执行读操作会返回 0，而写操作无法实现（尽管状态位会受影响）。可通过 FSRnH 和 FSRnL 对来创建 FSRn 寄存器值。FSR 寄存器形成的 16 位地址允许对 65536 个地址单元的空间进行寻址。

这些地址单元可划分为 3 个存储区：

- 传统数据存储器
- 线性数据存储器
- 闪存程序存储器

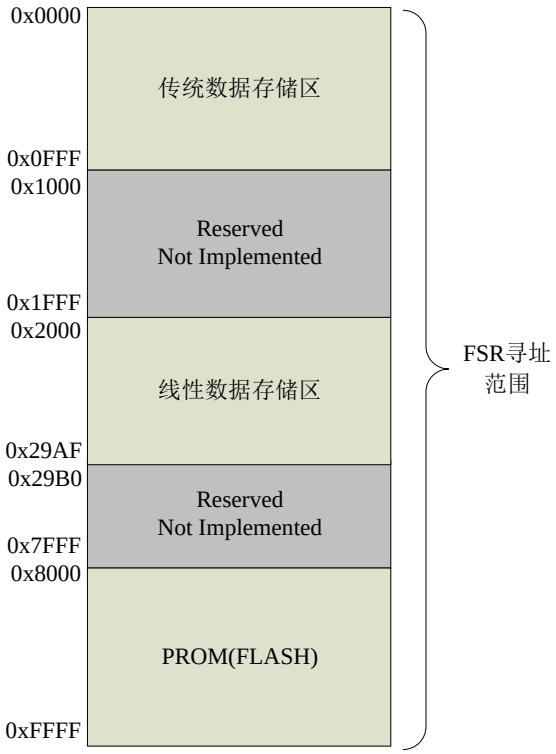


图 3.4 间接寻址

3.4.1. 传统数据存储器

传统数据存储器指的是从 FSR 地址 0x000 到 FSR 地址 0xFFF 的区域。此地址对应于所有 SFR、GPR 和公共寄存器的绝对地址。

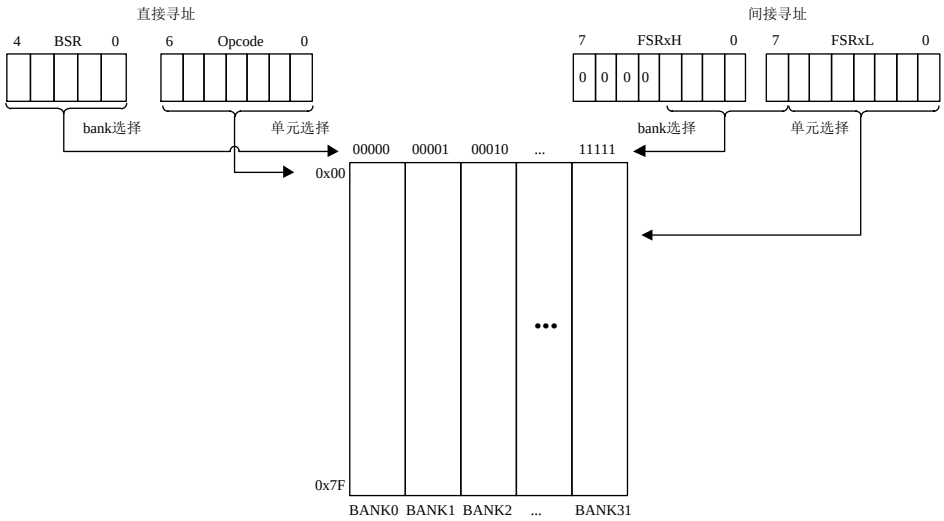


图 3.5 传统数据存储器映射

3.4.2. 线性数据存储器

线性数据存储器指的是从 FSR 地址 0x2000 到 FSR 地址 0x29AF 的区域。该区域为虚拟区域，它指向所有存储区中 80 字节的 GPR 存储区块。

未实现的存储区读为 0x00。使用线性数据存储器区域允许缓冲区大于 80 字节，因为当 FSR 增大到超过一个存储区时，会直接转到下一个存储区的 GPR 存储器。线性数据存储器区域不包含 16 字节的公共存储器。

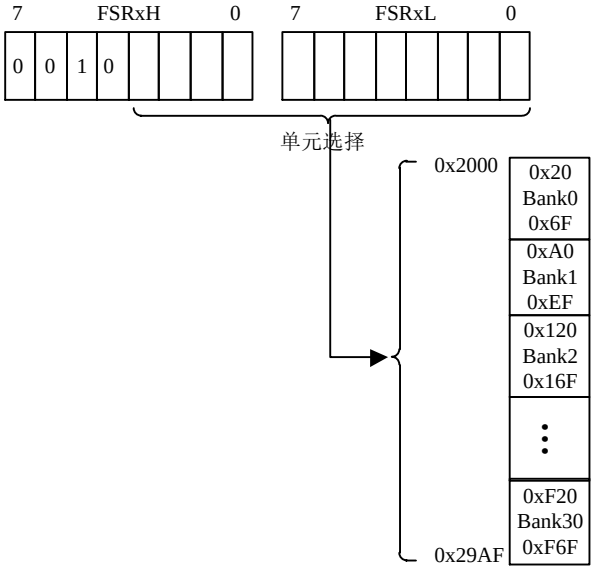


图 3.6 线性数据存储区映射

3.4.3. 闪存程序存储器

要使常数的访问更为容易，可将整个闪存程序存储器映射到 FSR 地址空间的高半部分。当 FSRnH 的 MSB 置 1 时，低 15 位就是可通过 INDF 进行访问的程序存储器的地址。只有每个存储单元的低 8 位可通过 INDF 进行访问。通过 FSR/INDF 接口无法对闪存程序存储器执行写操作。所有通过 FSR/INDF 接口对闪存程序存储器进行访问的指令都需要一个额外的指令周期才能完成。

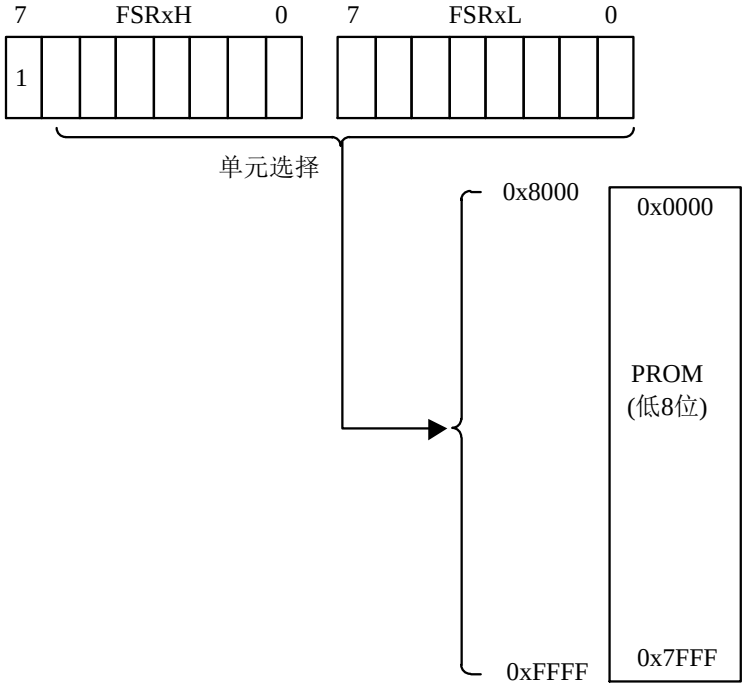


图 3.7 程序存储区映射

下图给出了 CPU 使用 FSRn 间接寻址 PROM 的时序，共占 2 个指令周期。

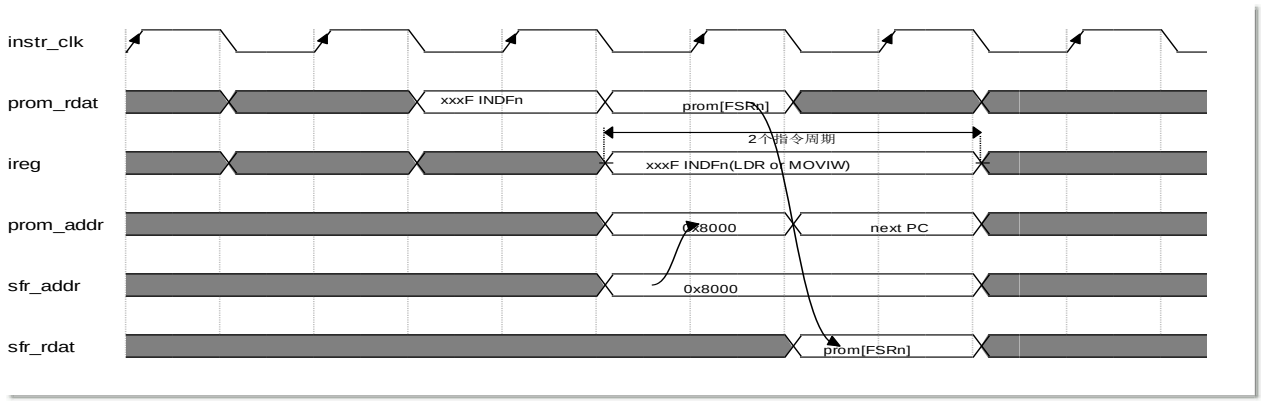


图 3.8 间接寻址读程序存储器

4. 复位源

有以下 7 种复位源：

- 上电复位
- 低电压复位
- 看门狗复位
- 非法指令复位
- 软件复位
- EMC 复位
- 外部管脚复位
- 堆栈溢出复位（见 3.3.2 小节）

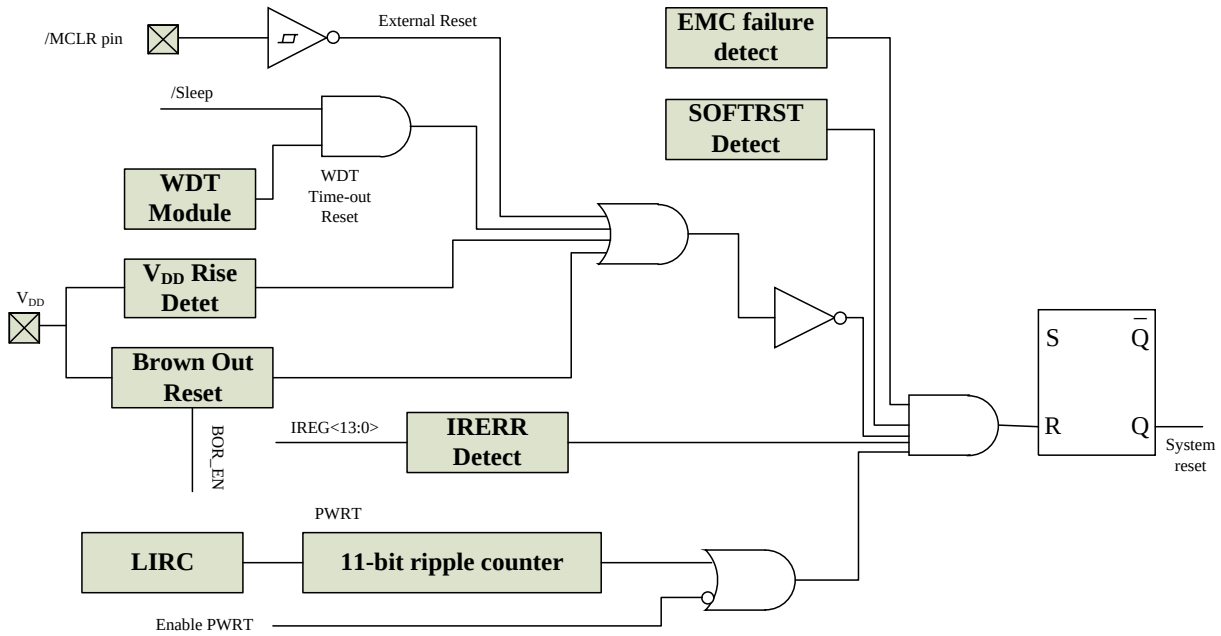


图 4.1 复位框图

4.1. 上电复位

片上的 POR 电路会将芯片保持在复位状态直到 VDD 电源电压达到足够高，上电复位释放后，系统复位不会立即释放，还要等一个约 4ms 的延时，期间数字电路保持在复位状态。

4.1.1. 上电复位流程

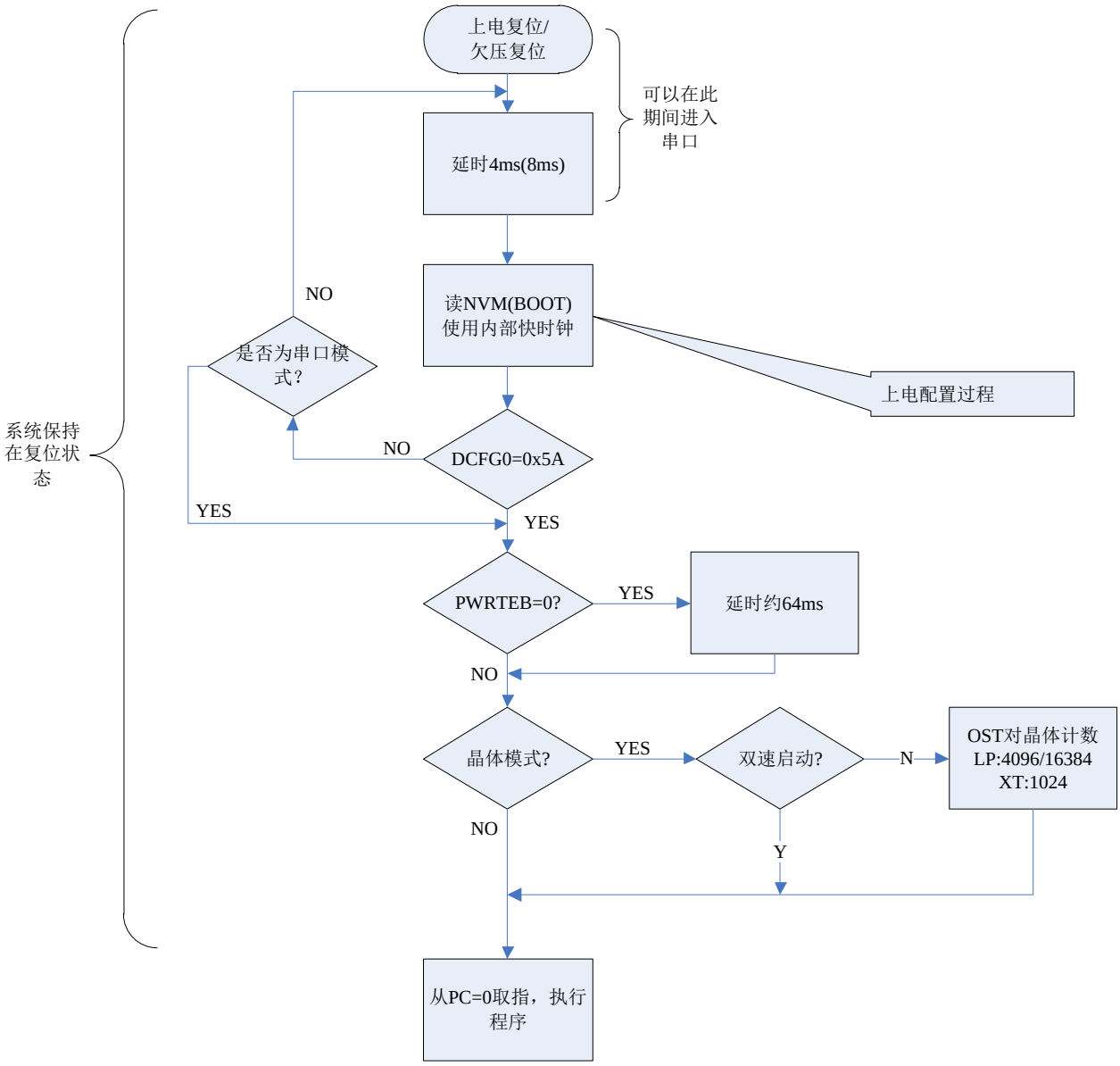


图 4.1.1 上电复位流程图

4.2. 低电压复位

低电压复位由 UCFG1<1:0>位和 SLVREN 位来控制。低电压复位就是指当电源电压低于 VBOR 门限电压时所产生的复位。不过当 VDD 电压低于 VBOR 不超过 T_{BOR}（4~5 个慢时钟周期）时间时，低电压复位可能不会发生。

如果 BOR（低电压复位）是使能（UCFG1<1:0>=00）的，那么最大 VDD 电压上升时间的要求就不存在。BOR 电路会将芯片控制在复位状态，一直到 VDD 电压达到 VBOR 门限电压以上。

当 UCFG1<1:0>=10 时，BOR 电路关闭将由 CPU 的运行状态决定：CPU 正常工作时 BOR 电路工作，CPU 处于睡眠模式时 BOR 电路自动关闭，这样可以方便的使系统功耗降至更低水平。

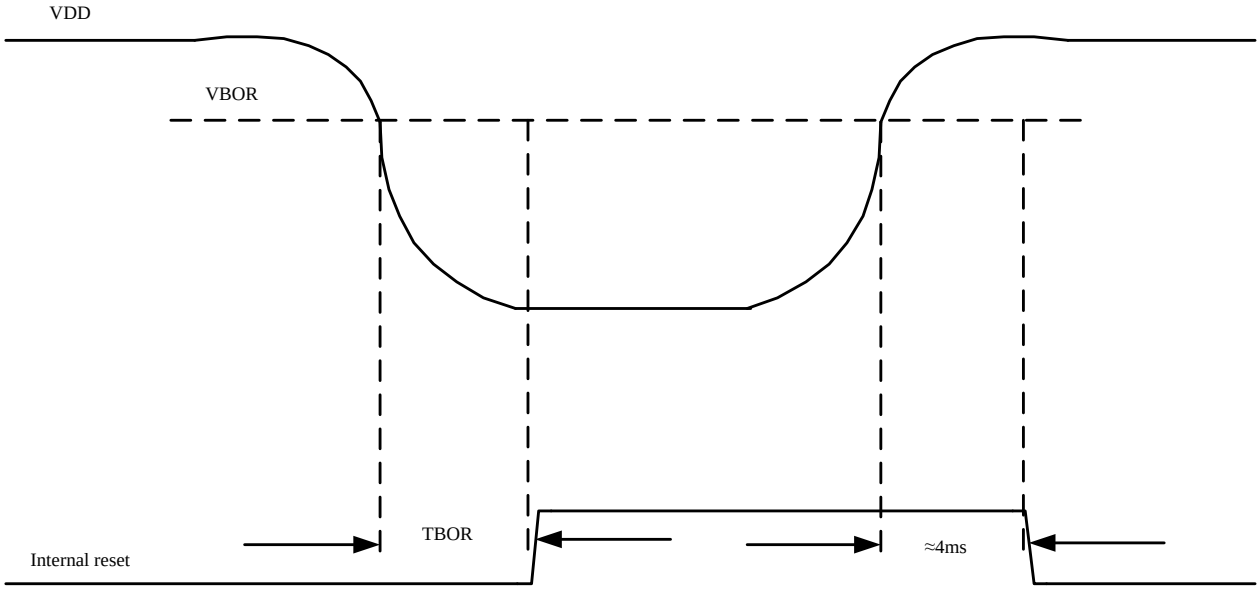


图 4.2.1 欠压复位

注意：

1. **T_{BOR} 时间约为 122~152μs；**
2. **电压恢复正常之后，内部复位不会立即释放，而是要等约为 4ms 的时间。**

4.3. 上电复位延时

复位模块内置了一个 11 位的上电复位定时器 PWRT 模块，它和 WDT 复用同一个计数器，它为上电复位和低电压复位提供一个固定的 64ms（正常情况下）的定时。这个定时器由内部慢时钟驱动。芯片在 PWRT 溢出之前都是被保持在复位状态，这段时间能保证 VDD 上升到足够高的电压使得系统能正常工作。

可通过系统寄存器（UCFG0）来使能。另外需要注意的是，由于由内部慢时钟驱动，定时的实际时间长度是随温度，电压等条件变化而变化的，这个时间不是一个精准参数。

4.4. 非法指令复位

当 CPU 的指令寄存器取指到非法指令（未定义的操作码）时，标志位 IERRF（PCON.4）将被置位，同时系统将进行复位。利用此功能可增加系统的抗干扰能力。

以下是非法指令汇总表格：

未定义指令编码			
b<13:12>	b<11:8>	b<7:4>	b<3:0>
00	0000	0000	**2~7**
00	0000	0000	C~F
00	0000	4~5	xxxx
00	0000	0110	0/1, 8~F
00	0000	0111	xxxx
00	**0001**	1~7	xxxx

4.5. 软件复位

增强型内核实现了一条软件复位指令，助记符为 RESET，它提供给软件执行硬件复位的方法，复位标志为 SRSTF（PCON.2）。

4.6. EMC 复位

配置区有一冗余寄存器 DCFG0，上电复位值为 0x5A，在配置过程中它被位于 NVM 区的 0x8047 单元低 8 位覆盖，如果其值不等于 0x5A，则 EMC 硬件检测逻辑会发出一次复位，标志位 EMCF 置 1，重启上电配置过程（读 NVM 区），直到读到正确的值为止（处于调试模式时，该功能自动禁止）。

在程序运行过程中，EMC 检测模块一直判断 DCFG0 的值，当发生某种 EMC 干扰导致其值发生变化时，（DCFG0≠0x5A）也将引发 EMC 复位，重启读 NVM 区的过程。

4.7. 上电配置过程(BOOT)

发生上电复位、低电压复位后，除了固有的 4ms 复位延时外，还有一个初始化配置寄存器 UCFGx 的动作。该动作从 PROM 的保留地址读取内容写到 UCFGx，待所有配置地址读取完成后，才可以释放系统复位，如图 4.7.1 和图 4.7.2 的所示，该过程大概需要 24μs。

4.7.1. 可触发 BOOT 过程的复位源汇总

复位源	可触发 Boot	可配置/配置位
上电复位	√	NA
低电压复位	√	NA
EMC 复位	√	NA
看门狗复位	√	Y，BRSTEN，FCFG0.0
非法指令复位	√	
外部管脚复位	√	
堆栈溢出复位	–	NA

4.8. LVD 低电压侦测

除了低电压复位功能外，芯片还内置有低电压侦测功能。当电源电压低于设置的电压档位（由 LVDCON 的 LVDL<2:0>选择）超过 T_{LVD}（4 到 5 个慢时钟周期）以上时，标志位 LVDW 将会被置 1，软件可以利用此位来监控电源电压。如果电源电压大于 LVDL 设置的电压档，该标志位会自动清除，**换言之，LVDW 位不具有锁存功能**。

4.8.1. 检测外部电压

除了可以监控片内 VDD 外，LVD 模块还具备检测外部电压的功能。寄存器位 LVDM 决定了 LVD 作用于 VDD 还是外部电压，当它为 1 时表示对外部管脚 ELVDx 进行监控。外部 LVD 管脚一共有 4 种选择，由寄存器 ELVDS<1:0>控制。当配置为 ELVD 功能时，管脚斯密特输入被关闭以防漏电。

检测外部电压（LVDM=1）时，LVD 的极性是相反的，即该模块只有检测到外部电压**高于**所设阈值且时间超过 T_{LVD} 时，LVDW 和 LVDIF 标志位才会置高。

注意：PC0 的外部复位功能优先级高于外部 LVD 功能，换言之，当配置为外部复位管脚时，外部 LVD 的检测是无效的，不管 ELVDS 以及 LVDM 为何值。

4.8.2. LVD 中断

除了通过轮询 LVDW 位了解低电压侦测事件外，软件还可以能过中断的方式来获得低电压的情况。当低电压侦测事件发生后，LVDIF 位自动置 1，它是一个电平触发锁存器，只能通过软件清 0，清 0 的前提是电源电压恢复到 LVDL<2:0>设置的水平以上，LVD 事件结束。

当 PEIE 和 LVDIE 被置 1 且 LVDIF 为 1 时，睡眠状态下 LVD 中断标志位还可以作为一个唤醒源，如果之前 GIE=1，则唤醒后 CPU 进入中断处理。

4.9. 复位时序

4.9.1. 上电复位时序 1

禁止 PWRT，内部时钟模式

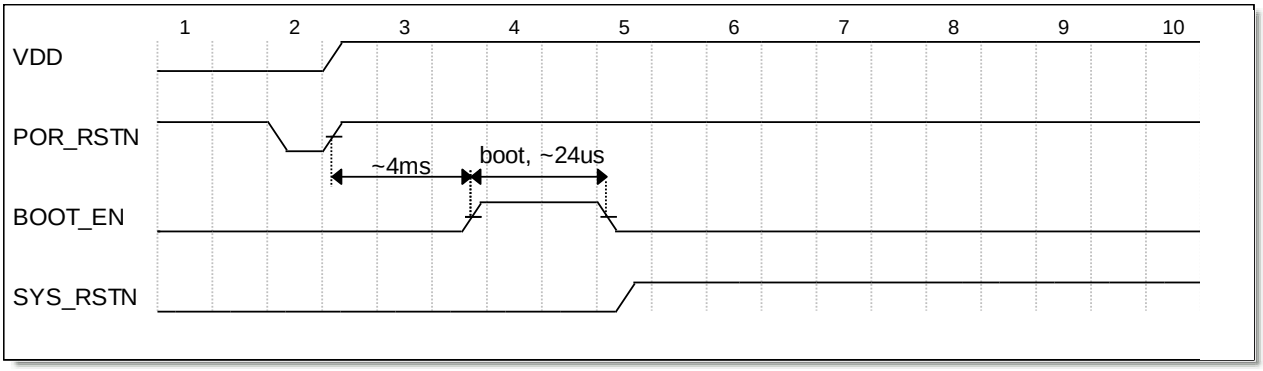


图 4.7.1 上电复位时序，使用内部时钟，PWRT 禁止

4.9.2. 上电复位时序 2

使能 PWRT，内部时钟模式

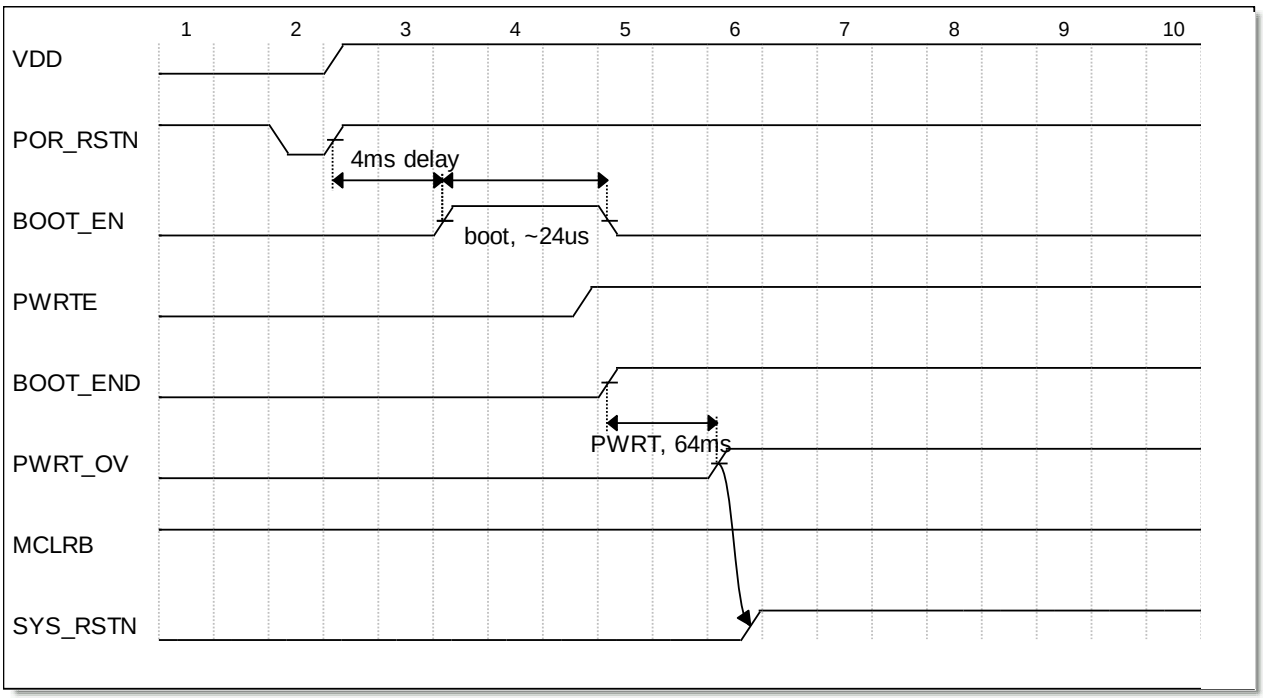


图 4.7.2 上电复位时序，使用内部时钟，PWRT 使能

4.9.3. 上电复位时序 3

使能 PWRT，晶体时钟模式，双速启动禁止

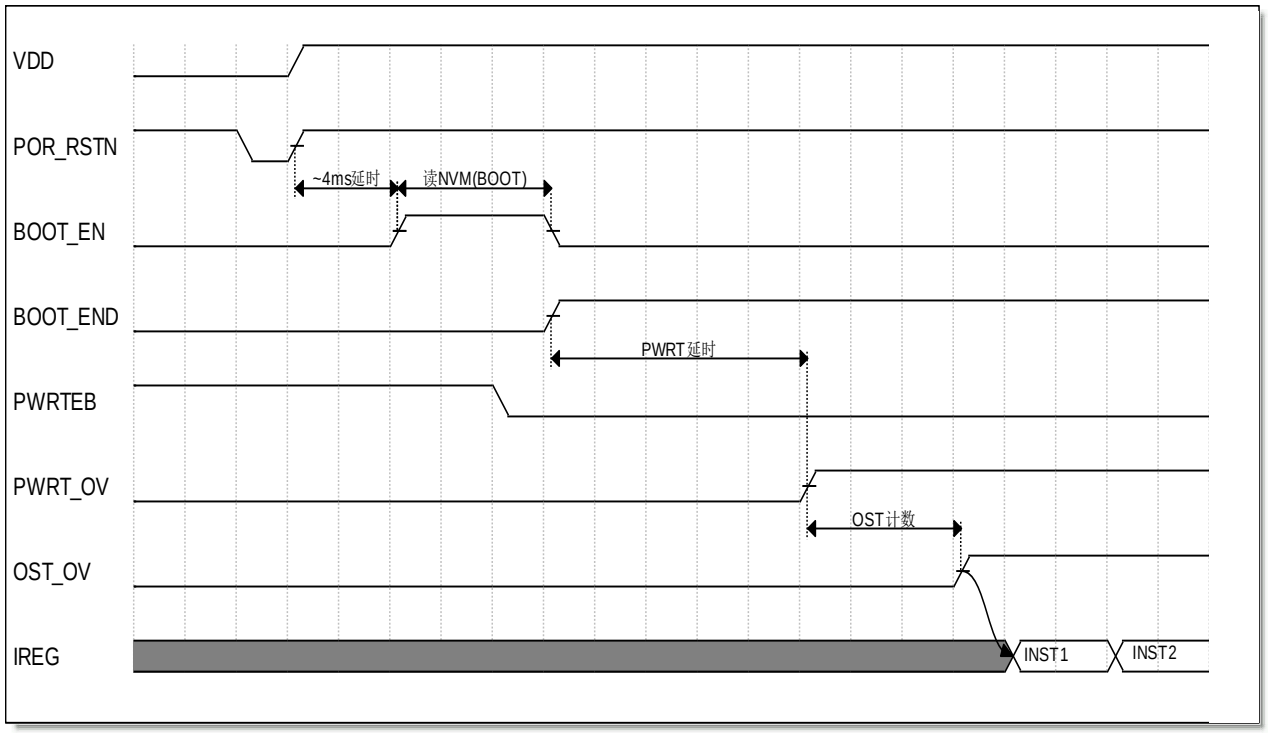


图 4.7.2 上电复位时序，使用内部时钟，PWRT 使能

4.10. 复位源标志位

以下复位事件都有对应的寄存器标志位记录，相关寄存器为 PCON，供软件查询。

上电复位（PORF）
欠压复位（BORF）
RESET 指令复位（SRSTF）
堆栈上溢复位（STKOVF）
堆栈下溢复位（STKUNF）
非法指令复位（IERRF）
EMC 复位（EMCF）
MCLR 复位（MCLRF）

4.10.1. PCON 寄存器，地址 0x96

Bit	7	6	5	4	3	2	1	0
Name	STKOVF	STKUNF	EMCF	IERRF	/MCLRF	/SRSTF	/PORF	/BORF
POR val.	0	0	0	0	1	1	0	1
BOR val.	0	0	0	0	1	1	1	0
Other rst.	Q	Q	Q	Q	Q	Q	U	U
Type	RW-0	RW-0	RW-0	RW-0	RW-1	RW-1	RW-1	RW-1

符号解释：
Q：取决于所发生的复位
U：保持不变
RW-0：软件只能写 0，不能写 1，只能由相关硬件事件置 1
RW-1：软件只能写 1，不能写 0，只能由相关硬件事件清 0

Bit	Name	Function
7	STKOVF	堆栈上溢标志位，高有效 0：未发生堆栈上溢，或该位由软件清 0 1：发生了堆栈上溢
6	STKUNF	堆栈下溢标志位，高有效 0：未发生堆栈下溢，或该位由软件清 0 1：发生了堆栈下溢
5	EMCF	EMC 复位标志，高有效 0：未发生 EMC 复位或由软件清 0 1：发生了 EMC 复位复位
4	IERRF	非法指令复位标志，高有效 0：未发生非法指令复位或由软件清 0 1：发生了非法指令复位复位
3	/MCLRF	外部复位标志，低有效 0：发生了 MCLR 复位 1：未发生 MCLR 复位或由软件置 1
2	/SRSTF	软件复位标志，低有效 0：执行了 RESET 指令 1：未执行 RESET 指令，或由软件置 1
1	/PORF	上电复位标志，低有效 0：发生了上电复位 1：没发生上电复位或者由软件置 1 PORF 在上电复位后值为 0，此后软件应该将其置 1
0	/BORF	低电压复位标志，低有效 0：发生了低电压复位 1：没发生低电压复位或者由软件置 1 /BOR 在上电复位后其值不确定，必须由软件置 1。发生后续复位后，通过查询此位来确定是否低电压复位

4.10.2. LVDCON 寄存器，地址 0x199

Bit	7	6	5	4	3	2	1	0
Name	SLVREN	LVDM	—	LVDEN	LVDW	LVDL		
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	SLVREN	软件控制 LVR 使能位，当 UCFG1<1:0>为 01 时： 1 = 打开 LVR 0 = 禁止 LVR 当 UCFG1<1:0>不为 01 时，此位无实际意义 **注意：发生欠压复位时，该位不会清 0。其它任何复位都可将其清 0**
6	LVDM	LVD 模块检测电压源选择 1 = 检测外部管脚 PC0（**这时如果 LVDEN 为 1，PC0 变为模拟管脚**） 0 = 检测内部电压
5	N/A	**保留位，不要写 1**
4	LVDEN	低电压侦测使能 1：开启 LVD 侦测功能 0：关闭 LVD 侦测功能
3	LVDW	低电压标志位，只读 **当 LVDM 为 0 时：** 1：VDD 掉到了 LVDL[2:0]所设置的电压超过 122μs 以上 0：VDD 正常，高于 LVDL[2:0]所设置的电压 **当 LVDM 为 1 时：** 1：所选外部管脚电压高于 LVDL[2:0]所设置的电压超过 122μs 以上 0：所选外部管脚电压低于 LVDL[2:0]所设置的电压
2:0	LVDL	低电压侦测选择位

值	检测电压
000	保留
001	保留
010	2.0V
011	2.4V
100	2.8V
101	3.0V
110	3.6V
111	4.0V

4.11. 配置寄存器汇总

4.11.1. UCFG0，PROM 地址 0x8000

Bit	7	6	5	4	3	2	1	0
Name	**Reserved**	CPB	MCLRE	PWRTEB	WDTE	FOSC		
POR val.	1	1	0	1	0	111		

位	名称	描述
7	**N/A**	**保留位，请保持该位置 1**
6	CPB	Flash 全区域（8K words）保护设置 1：不对 Flash 进行全区域保护 0：启用 Flash 全区域保护，除了 CPU 取指，CPU 或外部串口皆读返回 0 注意： 此位只能由 1 改写为 0，而不能由 0 改写为 1。由 0 改写成 1 的唯一方法是进行一次包括 USER_OPT 在内的片擦操作，并且重新上电后 CPB 才变为 1
5	MCLRE	1：PC0/MCLR 为复位脚功能 0：PC0/MCLR 脚为 GPIO
4	PWRTEB	1：PWRT 禁止 0：PWRT 使能
3	WDTE	1：WDT 使能，程序不能禁止 0：WDT 禁止，但程序可通过设置 WDTCON 的 SWDTEN 位将 WDT 使能
2:0	FOSC	000：LP 低速晶振模式，PC1/PB7 接低速晶体 001：XT 高速晶振模式，PC1/PB7 接高速晶体 010：外部时钟模式，PB7 为 IO 功能，PC1 接时钟输入 其它：INTOSCIO 模式，PC1 和 PB7 皆为 GPIO 引脚

4.11.2. UCFG1，PROM 地址 0x8001

Bit	7	6	5	4	3	2	1	0
Name	OSTPER		TSEL		IESO	FSCMEN	LVREN	
POR val.	0	1	2’b11		1	1	2’b11	

位	名称	描述
7:6	OSTPER	OST 定时器周期选择 00：512 01：1024(default) 10：2048 11：4096（LP 模式时为 32768）
5:4	TSEL	指令周期选择位 00，01：1T 10：2T 11：4T
3	IESO	双速时钟使能 1：使能双速时钟模式 0：禁止双速时钟模式
2	FSCMEN	时钟故障监视使能（当使用 EC 或者 XT、LP 模式时） 1：使能时钟故障监视 0：禁止时钟故障监视
1:0	LVREN	低电压复位选择 00：使能低电压复位 01：LVR 由 LVDCON 的 SLVREN 决定 10：MCU 正常模式时开启 LVR，睡眠模式时关闭 LVR，跟 SLVREN 位无关 11：禁止低电压复位

4.11.3. UCFG2，PROM 地址 0x8002

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	LVRS[3:0]			
POR val.	0	0	1	1	4’b0000			

位	名称	描述
7:4	N/A	保留位
3:0	LVRS[3:0]	低电压复位阈值选择

数值	电压
1010	保留
1011	
1100	
1101	
1110	
1111	
0000	
0001	
0000	保留
0011	2.0V
0100	2.2V
0101	2.5V
0110	2.8V
0111	3.1V
1000	3.6V
1001	4.1V

4.11.4. UCFG3，PROM 地址 0x8003

Bit	7	6	5	4	3	2	1	0
Name	FSECPB0							
POR val.	1	1	1	1	1	1	1	1

位	名称	描述
7:0	FSECPB0	Flash 扇区保护设置，低有效 主程序区的 8K words 分成 8 个扇区，每扇区大小 1K words Bitx: 0：扇区 x 被保护，软件和串口皆不能读，编程，页擦除（64 words） 1：扇区 x 不被保护，软件和串口可自由读，编程，页擦除（64words）

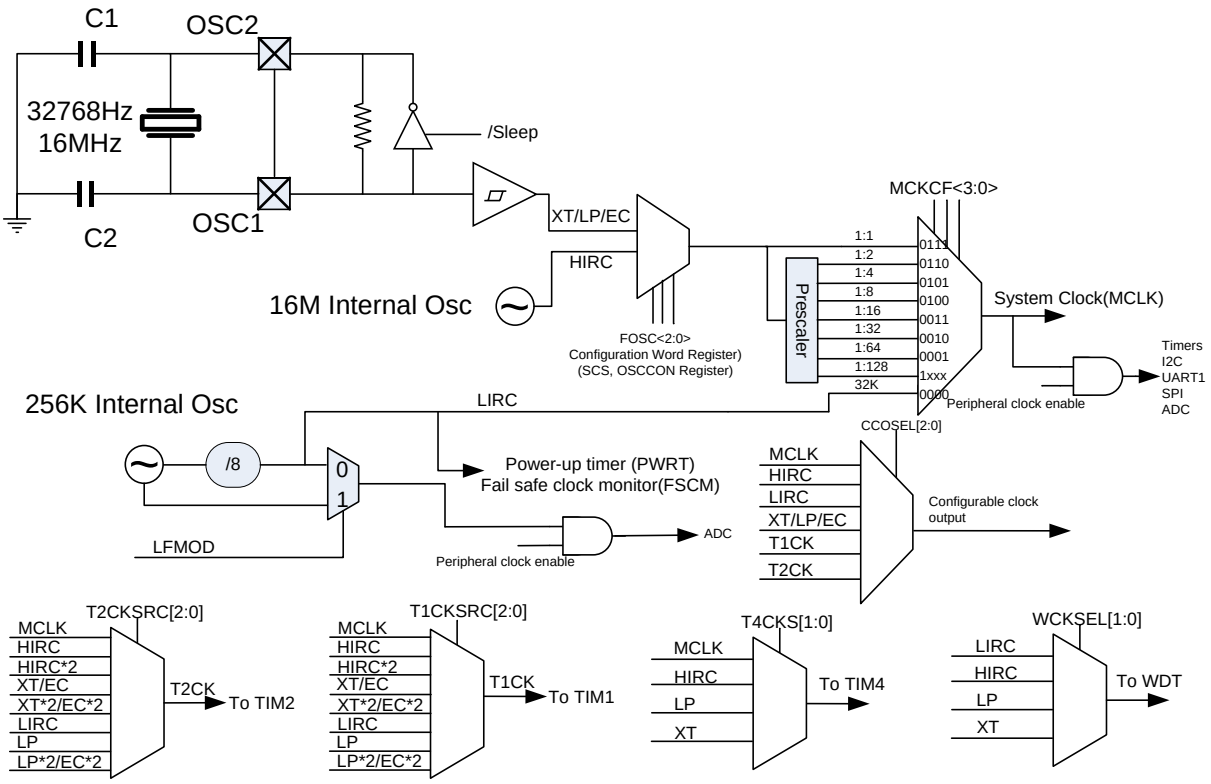
4.11.5. DCFG0，PROM 地址 0x8047

EMC 模块保护寄存器

Bit	7	6	5	4	3	2	1	0
Name	DCFG0							
POR val.	0	1	0	1	1	0	1	0

位	名称	描述
7:0	DCFG0	EMC 模块检测寄存器 0x5A：CPU 正常运行，EMC 不产生复位 其它值：在非串口模式下，EMC 将产生复位，并触发 boot，重读 NVM

5. 时钟源



5.1 系统时钟源框图

时钟源包含 4 个时钟源：2 个内置振荡器，1 个外部晶体振荡器，1 个外部时钟灌入源。内置振荡器包括 1 个内部 16M 高速精准振荡器(HIRC)，1 个内部 32K/256K(LIRC)低速低功耗振荡器。这些时钟或振荡器结合预分频器可以给系统提供各种频率的时钟源。同时内部高速振荡器可以通过 OSCTUNE 寄存器对振荡器的频率进行调节校准。

5.1. 时钟源模式

时钟源模式分为外部和内部模式。

外部时钟模式依靠外部电路提供时钟源，比如外部时钟 EC 模式，晶体谐振器 XT、LP 模式。

内部时钟模式内置于振荡器模块中，振荡器模块有 16MHz 高频振荡器和 32kHz 低频振荡器。可通过 OSCCON 寄存器的系统时钟选择位（SCS）来选择内部高速或者外部时钟源。

5.2. 外部时钟模式

5.2.1. 振荡器起振定时器（OST）

如果振荡器模块配置为 LP、XT 模式，振荡器起振定时器（OST）根据配置字 OSTPER<1:0>对来自 OSC1 的振荡计数。用户可以根据不同应用要求，通过设置 OSTPER 位调节 OST 计数的次数。这发生在上电复位（POR）之后或上电延时定时器（PWRT）延时结束（如果被使能）时，或从休眠中唤醒后，或故障保护条件清除后。在此期间，程序计数器不递增，程序执行暂停。OST 确保使用石英晶体谐振器或陶瓷谐振器的振荡器电路已经启动并向振荡器模块提供稳定的系统时钟信号。当在时钟源之间切换时，需要一定的延时以使新时钟稳定。

注意：OST 复用了 WDT 定时器，故在 OST 对晶体时钟计数时，WDT 功能被屏蔽，待 OST 发生溢出后，WDT 功能才恢复（如果此前 WDT 被使能的话）。当系统时钟切换到 LP 或者 XT 模式时，看门狗计数器会被清零。

5.2.2. EC 模式

外部时钟模式允许外部产生的逻辑电平作为系统时钟源。工作在此模式下时，外部时钟源连接到 OSC1 输入，OSC2 引脚可用作通用 I/O。

当选取 EC 模式时，振荡器起振定时器（OST）被禁止。因此，上电复位（POR）后或者从休眠中唤醒后的操作不存在延时。MCU 被唤醒后再次启动外部时钟，器件恢复工作，就好像没有停止过一样。

5.2.3. LP 和 XT 模式

LP 和 XT 模式支持连接到 OSC1 和 OSC2 的石英晶体谐振器或陶瓷谐振器，模式选择内部反相放大器的低或高增益设定，以支持各种谐振器类型及速度。

LP 振荡器模式选择内部反相放大器的最低增益设定。该模式设计仅用于驱动 32.768 kHz 音叉式晶振（钟表晶振）。

XT 振荡器模式选择内部反相放大器的高增益设定。

5.2.4. 内部时钟模式

振荡器模块有两个独立的内部振荡器，可配置或选取为系统时钟源。

1. HIRC（高频内部振荡器）出厂时已校准，工作频率为 16MHz。
2. LIRC（低频内部振荡器）未经校准，工作频率为 32 kHz。软件对 OSCCON 寄存器的系统时钟选择位 MCKCF<3:0>进行写操作，可选择系统时钟速度。

可通过 OSCCON 寄存器的系统时钟选择（SCS）位，在外部或内部高速时钟源之间选择系统时钟。

注意：OSCCON 寄存器的 LFMOD 可以选择 LIRC 是 32kHz 或者 256kHz，但看门狗固定使用 32kHz，不管 LFMOD 为何值。

5.2.5. 频率选择位（MCKCF）

外部晶体时钟，16MHz HIRC 和 32kHz LIRC 的输出连接到预分频器和多路复用器（见图 5.1）。OSCCON 寄存器的内部振荡器频率选择位 MCKCF<3:0>用于选择不同的分频输出。可通过软件选择以下各分频比或频率：

1:1
1:2
1:4
1:8（**复位后的缺省值**）
1:16
1:32
1:64
1:128
32 kHz（LIRC）

5.2.6. HIRC 和 LIRC 时钟切换时序

当在 LIRC 和 HIRC 之间切换时，新的振荡器可能为了省电已经关闭（见图 5.2 和图 5.3）。在这种情况下，OSCCON 寄存器的 MCKCF 位被修改之后、频率选择生效之前，存在一个延时。OSCCON 寄存器的 LTS 和 HTS 位将反映 LIRC 和 HIRC 振荡器的当前活动状态。频率选择时序如下：

1. OSCCON 寄存器的 MCKCF<3:0>位被修改
2. 如果新时钟是关闭的，会有一个时钟的启动延时
3. 时钟切换电路等待当前时钟的 2 个下降沿的到来
4. CLKOUT 保持为低，时钟切换电路等待两个新时钟下降沿的到来
5. 现在 CLKOUT 连接到新时钟。OSCCON 寄存器的 HTS 和 LTS 位被更新
6. 时钟切换完成

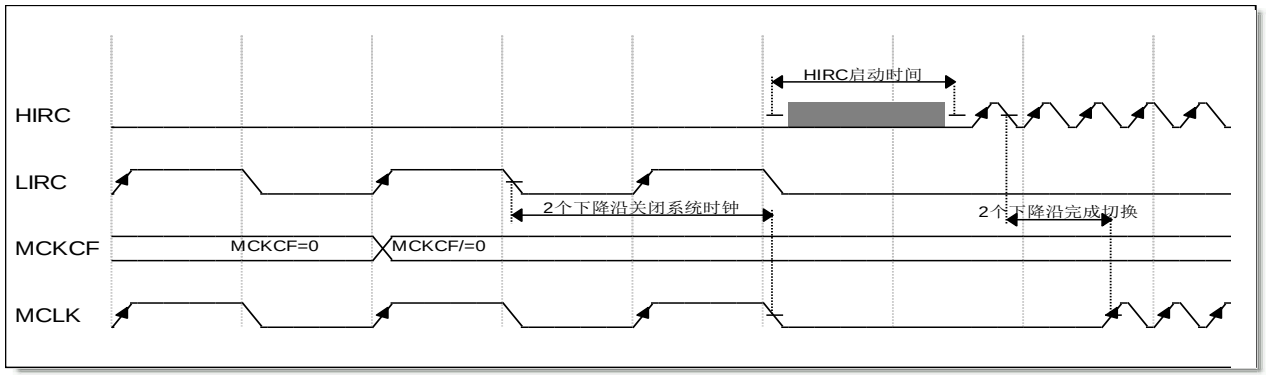


图 5.2 由慢时钟切换到快时钟

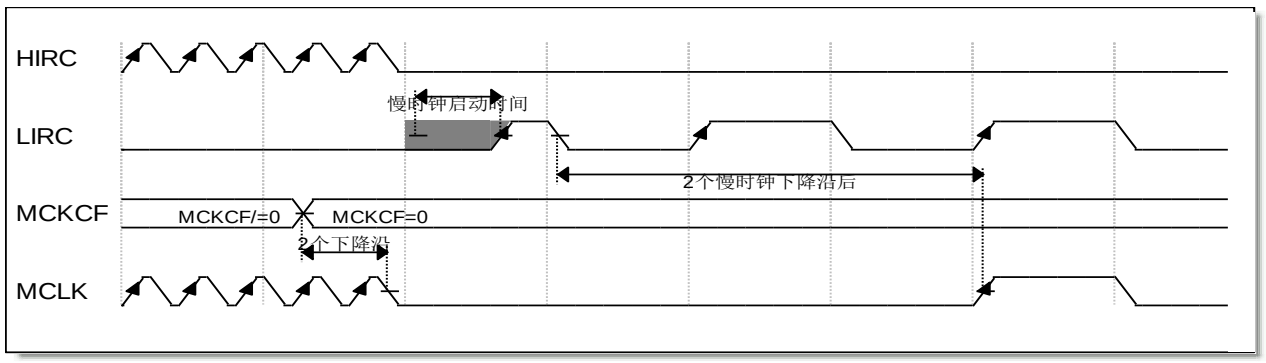


图 5.3 由快时钟切换到慢时钟

5.2.7. HIRC 时钟特殊功能

FT62F08X 内建了一个振荡频率可调的高精度 HIRC 作为系统时钟，出厂时被精确地调校至16MHz@5V/25℃，用户可以通过编程器的 Code Option 校准系统时钟。校准过程是过滤掉制程上的偏差对精度造成的影响。此 HIRC 受工作的环境温度和工作电压影响会有一定的漂移，对于压漂（4.5V~5.5V）以及（-20℃~85℃）的温漂一般状况会在±1%以内。

FT62F08X 有一个特殊的功能：用户可修改 OSCTUNE 的值来对 HIRC 频率作调整。
OSCTUNE 的值确保 HIRC 在上电后准确工作在 16MHz。此数值的初始值每颗 IC 都会有差异。初始值为OSCTUNE[s]，此时芯片工作在16MHz，每改变1个LSb则HIRC频率变化约为80kHz。OSCTUNE[6:0]和 HIRC 输出的关系如下：

OSCTUNE[6:0]值	HIRC 实际输出频率（16M 为例），单位 kHz
OSCTUNE[s]-n	(16000-n*80)
…..	…..
OSCTUNE[s]-2	16000-2*80=15840
OSCTUNE[s]-1	16000-1*80=15920
OSCTUNE[s]	16000
OSCTUNE[s]+1	16000+1*80=16080
OSCTUNE[s]+2	16000+2*80=16160
…..	…..
OSCTUNE[s]+n	(16000+n*80)

5.3. 时钟切换

通过软件对 OSCCON 寄存器的系统时钟选择（SCS）位进行操作，可将系统时钟源在外部和内部高速时钟源之间切换。

5.3.1. 系统时钟选择（SCS）位

OSCCON 寄存器的系统时钟选择（SCS）位选择用于 CPU 和外设的系统时钟源。

OSCCON 寄存器的位 SCS = 0 时，系统时钟源由配置字寄存器（UCFG0）中 FOSC<2:0>位的配置决定。
OSCCON 寄存器的位 SCS = 1 时，忽略 FOSC<2:0>位，根据 OSCCON 寄存器的 MCKCF<3:0>位决定系统时钟源：HIRC 的分频时钟或者 32k 时钟。

注：
1. 任何由硬件引起的时钟切换（可能产生自双速启动或故障保护时钟监控器）都不会更新 OSCCON 寄存器的 SCS 位。用户应该监控 OSCCON 寄存器的 OSTS 位以确定当前的系统时钟源；
2. 当 MCKCF<3:0>等于 0 时，无论 SCS 为何值，系统时钟都选择内部慢时钟。

5.3.2. 振荡器起振超时状态（OSTS）位

OSCCON 寄存器的振荡器起振超时状态（OSTS）位用于指示系统时钟是来自外部时钟源，还是来自内部时钟源。外部时钟源由配置字寄存器（UCFG0）的 FOSC<2:0>定义。OSTS 还特别指明在 LP 或 XT 模式下，振荡器起振定时器（OST）是否已超时。

5.3.3. 双速时钟启动模式

双速启动模式通过最大限度地缩短外部振荡器起振与代码执行之间的延时，进一步节省了功耗。对于频繁使用休眠模式的应用，双速启动模式将在器件唤醒后除去外部振荡器的起振时间，从而可降低器件的总体功耗。该模式使得应用能够从休眠中唤醒，将 INTOSC 用作时钟源执行数条指令，然后再返回休眠状态而无需等待主振荡器的稳定。

注：执行 SLEEP 指令将中止振荡器起振时间，并使 OSCCON 寄存器的 OSTS 位保持清零。

当振荡器模块配置为 LP 或 XT 模式时，振荡器起振定时器（OST）使能（见第 5.2.1 节“振荡器起振定时器”）。OST 将暂停程序执行，直到完成配置字 OSTPER<1:0>位要求的计数次数。双速启动模式在 OST 计数时使用内部振荡器进行工作，使代码执行的延时最大限度地缩短。当 OST 计数到 OSTPER<1:0>位要求的计数次数且 OSCCON 寄存器的 OSTS 位置 1 时，程序执行切换至外部振荡器。

注：
1.系统时钟配置为外部晶振模式时，同时使能了双速模式，在 OST 未计数到 OSTPER 要求的数值时，CLRWDT 指令不能清除看门狗计数器，也就是说此时 OST 计数不能被中断；
2.系统时钟配置为外部晶振模式时，同时使能了双速模式，在 OST 未计数到 OSTPER 要求的数值时，执行 sleep 指令时，看门狗计数器被清零，此时的 OST 计数被清零。

5.3.4. 双速启动模式配置

通过以下设定来配置双速启动模式：

- 配置字寄存器（UCFG1）中的位 IESO = 1；内部/外部切换位（使能双速启动模式）
- OSCCON 寄存器的位 SCS = 0
- 配置字寄存器（UCFG0）中的 FOSC<2:0>配置为 LP 或 XT 模式

在下列操作之后，进入双速启动模式：

- 上电复位（POR）且上电延时定时器（PWRT）
- 延时结束（使能时）后，或者从休眠状态唤醒

如果外部时钟振荡器配置为除 LP 或 XT 模式以外的任一模式，那么双速启动将被禁止。这是因为 POR 后或从休眠中退出时，外部时钟振荡器不需要稳定时间。

5.3.5. 双速启动顺序

1. 从上电复位或休眠中唤醒
2. 使用内部振荡器以 OSCCON 寄存器的 MCKCF<3:0>位设置的频率开始执行指令
3. OST 使能，计数 OSTPER<1:0>位要求的计数次数
4. OST 超时，等待内部振荡器下降沿的到来
5. OSTS 置 1
6. 系统时钟保持为低，直到新时钟下一个下降沿的到来（LP 或 XT 模式）
7. 系统时钟切换到外部时钟源

5.3.6. 故障保护时钟监控器

故障保护时钟监控器（FSCM）使得器件在出现外部振荡器故障时仍能继续工作。FSCM 能在振荡器起振延时定时器（OST）到期后的任一时刻检测振荡器故障。FSCM 通过将配置字寄存器（UCFG1）中的 FSCMEN 位置 1 来使能。FSCM 可用于所有外部振荡模式（LP、XT 和 EC）。

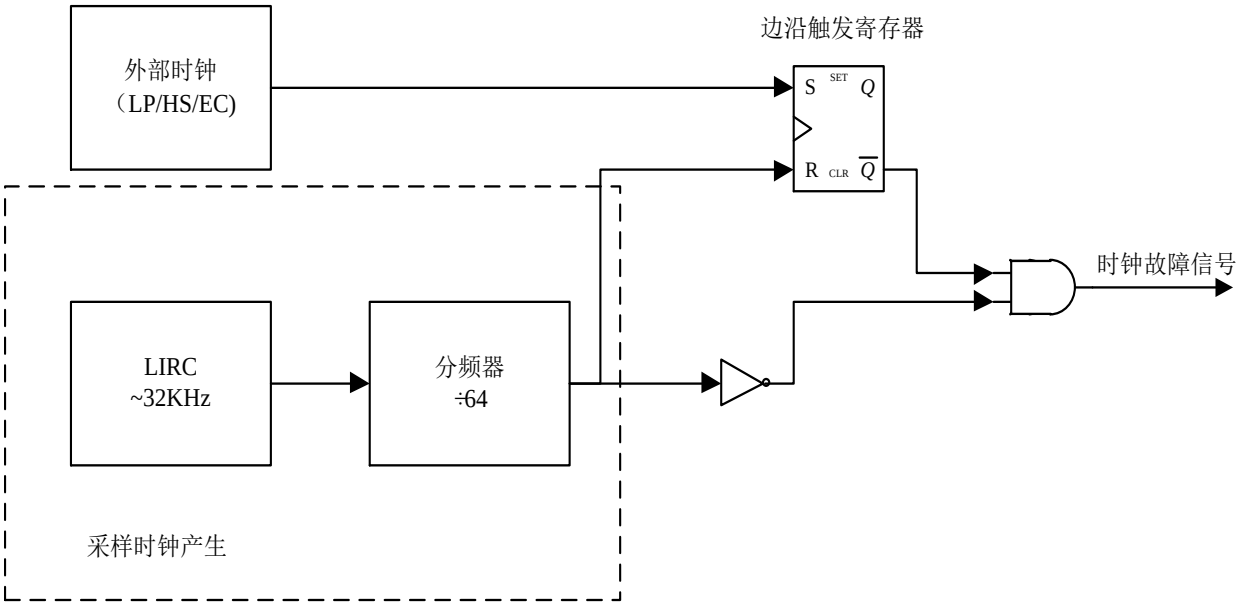


图 5.4 FSCM 原理框图

5.3.7. 故障保护检测

FSCM 模块通过将外部振荡器与 FSCM 采样时钟比较来检测振荡器故障。LIRC 除以 64，就产生了采样时钟。请参见图 5.4。故障检测器内部有一个锁存器。在外部时钟的每个下降沿，锁存器被置 1。在采样时钟的每个上升沿，锁存器被清零。如果采样时钟的整个半周期流逝而主时钟依然未进入低电平，就检测到故障。

5.3.8. 故障保护操作

当外部时钟出现故障时，FSCM 将器件时钟切换到内部时钟源，并将 PIR1 寄存器的 OSFIF 标志位置 1。如果在 PIR1 寄存器的 OSFIE 位置 1 的同时将该标志位置 1，将产生中断。固件随后会采取措施减轻可能由故障时钟所产生的问题。系统时钟将继续来自内部时钟源，直到器件固件成功重启外部振荡器并切换回外部操作。

FSCM 所选的内部时钟源由 OSCCON 寄存器的 MCKCF<3:0>位决定。这使内部振荡器可以在故障发生前就得以配置。

5.3.9. 故障保护条件清除

复位、执行 SLEEP 指令或翻转 OSCCON 寄存器的 SCS 位后，故障保护条件被清除。OSCCON 寄存器的 SCS 位被修改后，OST 将重新启动。OST 运行时，器件继续从 OSCCON 中选定的 INTOSC 进行操作。OST 超时后，故障保护条件被清除，器件将从外部时钟源进行操作。必须先清除故障保护条件，才能清零 OSFIF 标志位。

5.3.10. 复位或从休眠中唤醒

FSCM 设计为能在振荡器起振延时定时器（OST）到期后的任一时刻检测振荡器故障。OST 的使用场合为从休眠状态唤醒后以及任何类型的复位后。OST 不能在 EC 时钟模式下使用，所以一旦复位或唤醒完成，FSCM 就处于激活状态。当 FSCM 被使能时，双速启动也被使能。因此，当 OST 运行时，器件总是处于代码执行阶段。

注：由于振荡器起振时间的范围变化较大，在振荡器起振期间（从复位或休眠中退出时），故障保护电路不处于激活状态。经过一段适当的时间后，用户应检查 OSCCON 寄存器的 OSTS 位，以验证振荡器是否已成功起振以及系统时钟是否切换成功。

5.4. 外设时钟门控

关闭未使用外设的时钟可降低功耗。外设的时钟门控模式可以对以下外设的时钟随时打开或者关闭：

- ADC
- I2C
- SPI
- USART
- TIM1/2/3
- TOUCH

系统复位后，所有的外设时钟均处于关闭的状态。用户可以通过配置 PCKEN 寄存器对应的位来打开相应的外设时钟。但是如果需要禁止某个外设，必须在其时钟被停止之前进行。

如果要使能某个外设，则需要先使能对应外设的时钟，然后再使能对应外设。外设时钟控制只是控制其模块的系统时钟。

TIM1/2/4 计数时钟也受 PCKEN 寄存器控制，当 PCKEN 寄存器对应的未置 1 时，则 TIMER 的计数时钟和寄存器操作时钟同时打开。

注意：

1. 在睡眠模式下，当 SYSON=1 时，无论 TIM1/2 的时钟源是否为系统时钟，其时钟源都会打开；当 SYSON=0 时，TIM1/2 时钟源关闭。
2. ADC 的转换时钟源打开或者关闭与系统进入睡眠模式无关。
3. 慢时钟测量时，慢时钟的打开与关闭也与系统时钟进入睡眠模式无关。

5.5. 时钟输出

可以将芯片内部的时钟源输出到芯片的 CLKO 管脚上，可以选择以下几种时钟源输出：

- 系统时钟
- 内部高速时钟
- XT 晶振时钟
- LP 晶振时钟
- 外部时钟
- 内部慢时钟
- TIM1/TIM2 时钟

这些时钟的输出选择和输出控制位于 CKOCON 寄存器。当时钟正在输出时，CCORDY 被硬件置 1。

当输出某个时钟时，对应的时钟源也同时被打开。通过清零 CCOEN 位禁止时钟的输出。时钟输出关闭动作完成之后，CCORDY 位才能被硬件清零。

注意：

1. 在睡眠模式下，CCO 的输出与 SYSON 相关，当 SYSON=1 时，CCO 继续输出进入睡眠之前所选的时钟；当 SYSON=0 时，CCO 暂停输出时钟，当系统退出睡眠状态后，继续输出所选时钟；
2. 当系统时钟为 XT 模式时，即使输出时钟选择了 LP 时钟，输出时钟也是 XT 时钟；
3. 当系统时钟为 LP 模式时，即使输出时钟选择了 XT 时钟，输出时钟也是 LP 时钟；
4. 当 FOSC<2:0>选择为内部时钟源时，输出时钟选择了 LP、XT 或者 EC 时钟，输出时钟端口则不会输出时钟；

5.5.1. 时钟输出注意

MCU 处于工作模式时，被选择为时钟输出的时钟源将自动使能，例如当前系统时钟为 HSI，时钟输出选择为 XT 晶振时钟且 CCOEN 为 1，则对应的 XT 晶振电路打开，相关管脚也变成 OSC1, OSC2 功能。

UCFG0 配置选项的优先级要高于 CKOCON 寄存器的优先级，例如配置选项选择 XT 作为系统时钟，时钟输出配置 LP 或 EC 是不起作用的，反之亦然。

5.6. 与时钟源相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
UCFG0	0x8000	—	CPB	MCLRE	PWRTEB	WDTE	FOSC2	FOSC1	FOSC0	-qqq qqqq
OSCTUNE	0x98	—	TUN[6:0]							-xxx xxxx
OSCCON	0x99	MCKCF[3:0]				OSTS	HTS	LTS	SCS	0100 x000
PCKEN	0x9A	TKEN	I2CEN	UARTEN	SPICKEN	TIM4EN	TIM2EN	TIM1EN	ADCEN	0000 0000
CKOCON	0x95	SYSON	CCORDY	DTYSEL[1:0]		CCOSEL[2:0]			CCOEN	0010 0000
TCKSRC	0x31F	LFMOD	T2CKSRC[2:0]			—	T1CKSRC[2:0]			0000 -000

5.6.1. OSCCON 寄存器，地址 0x99

Bit	7	6	5	4	3	2	1	0
Name	MCKCF[3:0]				OSTS	HTS	LTS	SCS
Reset	4'b0100				x	0	0	0
TYPE	RW				RO	RO	RO	RW

Bit	Name	Function
7:4	MCKCF[3:0]	主时钟（系统时钟）分频比选择 值 / 分频比或 LIRC 选择 0111 / 1:1 0110 / 1:2 0101 / 1:4 0100 / **1:8(default)** 0011 / 1:16 0010 / 1:32 0001 / 1:64 1xxx / 1:128 0000 / 32kHz(LIRC)
3	OSTS	振荡器起振超时状态位 1 = 器件运行在 FOSC<2:0>指定的外部时钟之下 0 = 器件运行在内部振荡器之下
2	HTS	高速内部时钟状态 1 = HIRC is ready 0 = HIRC is not ready
1	LTS	低速内部时钟状态 1 = LIRC is ready 0 = LIRC is not ready
0	SCS	系统时钟选择位 1 = 系统时钟选择为内部振荡器 0 = 时钟源由 FOSC<2:0>决定

5.6.2. OSCTUNE 寄存器，地址 0x98

Bit	7	6	5	4	3	2	1	0
Name	—	TUN[6:0]						
Reset	0	7'bxxx_xxxx						
TYPE	RO-0	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	N/A	保留位，读 0
6:0	TUN[6:0]	内部高速时钟频率调节位，见HIRC时钟特殊功能

5.6.3. PCKEN 寄存器，地址 0x9A

Bit	7	6	5	4	3	2	1	0
Name	TKEN	I2CEN	UARTEN	SPICKEN	TIM4EN	TIM2EN	TIM1EN	ADCEN
Reset	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	TKEN	Touch 模块时钟使能位： 1 = 打开时钟 0 = 关闭时钟
6	I2CEN	I2C 模块时钟使能位： 1 = 打开时钟 0 = 关闭时钟
5	UARTEN	USART 模块时钟使能位： 1 = 打开时钟 0 = 关闭时钟
4	SPICKEN	SPI 模块时钟使能位： 1 = 打开时钟 0 = 关闭时钟
3	TIM4EN	TIM4 模块时钟使能位： 1 = 打开时钟 0 = 关闭时钟
2	TIM2EN	TIM2 模块时钟使能位： 1 = 打开时钟 0 = 关闭时钟
1	TIM1EN	TIM1 模块时钟使能位： 1 = 打开时钟 0 = 关闭时钟
0	ADCEN	ADC 模块时钟使能位： 1 = 打开时钟 0 = 关闭时钟

5.6.4. CKOCON 寄存器，地址 0x95

Bit	7	6	5	4	3	2	1	0
Name	SYSON	CCORDY	DTYSEL		CCOSEL[2:0]			CCOEN
Reset	0	0	1	0	0	0	0	0
TYPE	RW	RO	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	SYSON	睡眠状态下的系统时钟控制位 1 = 睡眠状态下，系统时钟保持运行 0 = 睡眠状态下，系统时钟关闭
6	CCORDY	时钟输出标志位，**只读** 1 = 时钟输出已准备好 0 = 时钟未输出
5:4	DTYSEL	TIM1/TIM2 倍频时钟占空比调节位 00：2ns 延迟 01：3ns 延迟 10：4ns 延迟 11：7ns 延迟
3:1	CCOSEL[2:0]	输出时钟选择位： 000：MCLK/系统时钟 001：HIRC 010：LIRC 011：XT 100：T1CK 101：T2CK 110：LP 111：EC
0	CCOEN	时钟输出使能位： 1 = 使能时钟输出 0 = 禁止时钟输出

5.6.5. TCKSRC 寄存器，地址 0x31F

Bit	7	6	5	4	3	2	1	0
Name	LFMOD	T2CKSRC			—	T1CKSRC		
Reset	0	0	0	0	—	0	0	0
TYPE	RW	RW	RW	RW	RO-0	RW	RW	RW

Bit	Name	Function
7	LFMOD	低频内振模式： 1 = 256K 振荡频率模式 0 = 32K 振荡频率模式
6:4	T2CKSRC	TIM2 时钟源选择位 值 / 时钟源 000 系统时钟/主时钟 001 HIRC 010 XT 时钟/外部时钟 011 HIRC 的 2 倍频 100 XT 时钟/外部时钟的 2 倍频 101 LIRC 110 LP 时钟/外部时钟 111 LP 时钟/外部时钟的 2 倍频
3	N/A	保留位，读 0
2:0	T1CKSRC	TIM1 时钟源选择位 值 / 时钟源 000 系统时钟/主时钟 001 HIRC 010 XT 时钟/外部时钟 011 HIRC 的 倍频 100 XT 时钟/外部时钟的 2 倍频 101 LIRC 110 LP 时钟/外部时钟 111 LP 时钟/外部时钟的 2 倍频

注意：

1. 当系统时钟配置选项选择 XT 模式时，TIMx 时钟源就不能选择 LP 或者 LP 的 2 倍频；

2. 同样地，当系统时钟配置为 LP 模式时，TIMx 时钟源就不能选择 XT 或者 XT 的 2 倍频；

3. 当 FOSC<2:0>配置为内部时钟时，TIMx 时钟源被配置为 LP、XT 或者 EC 时钟源，此时 TIMx 无时钟源输入；

6. 中断

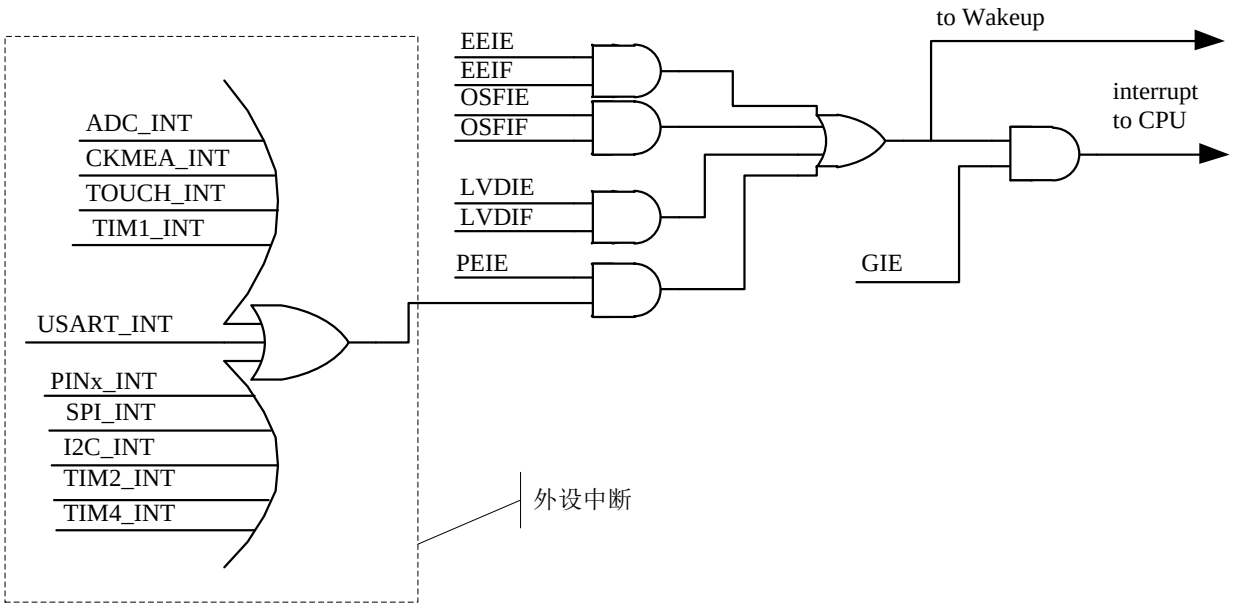


图 6.1 中断逻辑框图

FT62F08X 有以下中断源，部分中断可以把 CPU 从睡眠状态唤醒：

- 外部管脚中断
- ADC 中断
- LVD 中断
- EEPROM 写完成中断
- 慢时钟测量完成中断
- 时钟缺失中断
- TIMx 中断
- SPI 中断
- I2C 中断
- USART 中断
- 触摸中断

6.1. 中断的使能

由图 6.1 可以看出，要想使用中断，除了该中断的相关使能位（xxxIE）需要置 1 外，总中断开关（GIE）也需要打开。

ADC，慢时钟测量，触摸按键，TIMx，外部管脚，SPI，I2C 以及 USART 模块被归类了外设中断，除了其自身的中断使能要打开外，另外一个外设总中断开关 PEIE 也需要置 1。

另外，各中断标志位的置起并跟中断使能位无关。

6.2. 中断的响应时间

中断响应延时定义为从发生中断事件到开始执行中断向量处的代码所需要的时间。正常情况下（即不是处于 EEPROM 或 FLASH 的写周期），同步中断的响应延时为 3 或 4 个指令周期。对于异步中断，响应延时为 3 至 5 个指令周期，具体取决于中断发生的时间和正在执行的指令。

6.3. 睡眠下的中断

由于睡眠状态下系统时钟被关闭，部分使用系统时钟作为时钟的外设将停止工作。可以将 CPU 唤醒的中断源有：

- 外部管脚中断
- EEPROM 写完成中断
- LVD 中断
- TOUCH 中断
- TIMx 中断（使用慢时钟时）
- ADC 中断

需要注意的是，唤醒 CPU 并不要求 GIE 使能。GIE 为 0 时，CPU 唤醒后将执行 SLEEP 指令后面的代码，而非中断向量。

当 CKOCON.SYSON 为 1 时，系统时钟保持运行，所以其它直接使用系统时钟的外设也可以把 CPU 唤醒，**换言之，在这种条件下，所有中断都能唤醒 CPU。**

注意，由于同步带来的延时，在对中断标志位清 0 之后，至少要等两条指令才可以执行 SLEEP 指令，否则 MCU 将不会进入睡眠。

6.4. 现场保护

进入中断时，中断控制单元将返回的 PC 地址保存在堆栈中。而且，以下寄存器自动保存在影子寄存器中：

- W 寄存器
- STATUS 寄存器（TO 和 PD 状态标志位除外）
- BSREG 寄存器
- FSR 寄存器
- PCLATH 寄存器

退出中断服务程序时，这些寄存器将自动恢复。在 ISR 期间对这些寄存器所做的任何修改都将丢失。如果需要修改任何这些寄存器，应修改相应的影子寄存器，并在退出 ISR 时恢复此新值。影子寄存器在 Bank31 中，可读写。根据用户应用程序的要求，可能还需要保存其他寄存器，这些额外的寄存器则需要用户自行处理。

6.5. 与中断相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
INTCON	0x0B	GIE	PEIE	EEIE	LVDIE	OSFIE	EEIF	LVDIF	OSFIF	0000 0000
PIE1	0x91	—	—	—	—	—	TKIE	CKMIE	ADCIE	---- -000
PIR1	0x11	—	—	—	—	—	TKIF	CKMIF	ADCIF	---- -000
EPIF0	0x14	外部管脚中断标志位								0000 0000
EPIE0	0x94	外部管脚中断使能位								0000 0000

6.5.1. INTCON 寄存器，地址 0x0B

Bit	7	6	5	4	3	2	1	0
Name	GIE	PEIE	EEIE	LVDIE	OSFIE	EEIF	LVDIF	OSFIF
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	GIE	全局中断使能 1 = 打开全局中断 0 = 关闭全局中断
6	PEIE	外设中断使能 1 = 打开外设中断 0 = 关闭外设中断
5	EEIE	EEPROM写完成中断使能 1 = 打开EE中断 0 = 关闭 EE 中断
4	LVDIE	LVD中断使能 1 = 打开LVD中断t 0 = 关闭 LVD 中断
3	OSFIE	**时钟缺失中断使能** 1 = 打开时钟缺失中断 0 = 关闭时钟缺失中断 t
2	EEIF	EEPROM写完成标志位（**写1清0，写0无效**） 1 = EEPROM完成写操作 0 = EEPROM 写操作未完成，或已由软件清 0
1	LVDIF	LVD中断标志位（**写1清0，写0无效**） 1 = 发生欠压事件（当LVDM为1时，则表示外部管脚电压高于所设阈值） 0 = 未发生欠压事件，或已由软件清 0
0	OSFIF	1 = 发生了时钟缺失事件（**写1清0，写0无效**） 0 = 未发生时钟缺失事件，或已由软件清0

6.5.2. PIR1 寄存器，地址 0x11

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	TKIF	CKMIF	ADCIF
Reset	—	—	—	—	—	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW

Bit	Name	Function
7:3	N/A	未实现，读 0
2	TKIF	触摸中断标志位 1 = 有触摸检测事件（**写1清0，写0无效**） 0 = 没有触摸检测事件，或已由软件清 0
1	CKMIF	慢时钟测量完成中断标志位 1 = 测量慢时钟操作完成 （**写1清0，写0无效**） 0 = 测量慢时钟未完成，或已由软件清 0
0	ADCIF	ADC 转换完成中断标志位 1 = ADC转换完成（**写1清0，写0无效**） 0 = ADC 转换未完成，或已由软件清 0

6.5.3. PIE1 寄存器，地址 0x91

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	TKIE	CKMIE	ADCIE
Reset	—	—	—	—	—	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW

Bit	Name	Function
7:3	N/A	未实现，读 0
2	TKIE	触摸中断使能 1 = 允许触摸中断 0 = 禁止触摸中断
1	CKMIE	慢时钟测量完成中断使能 1 = 允许慢时钟测量中断 0 = 禁止慢时钟测量中断
0	ADCIE	ADC 转换完成中断使能 1 = 允许ADC转换中断 0 = 禁止 ADC 转换中断

7. 睡眠模式

器件通过执行 SLEEP 指令进入睡眠模式。
进入休眠模式时，MCU 的状态如下：
1. WDT 将清零但是保持运行（如果使能了在休眠期间工作）
2. STATUS 寄存器的 PD 位清零
3. STATUS 寄存器的 TO 位置 1
4. CPU 时钟停止
5. 32kHz LIRC 不受影响，并且由其提供时钟的外设可以在休眠模式下继续工作
6. LP 晶体振荡器不受影响（当 TIMx 使用它作为工作时钟时）
7. ADC 不受影响（如果选择了专用 FRC 时钟）
8. 电容触摸传感振荡器不受影响
9. I/O 端口保持执行 SLEEP 指令之前的状态（驱动为高电平、低电平或高阻态）
10. WDT 之外的复位不受休眠模式影响

关于外设在休眠期间工作的更多详细信息，请参见各个章节。

要最大程度地降低电流消耗，应考虑以下条件：
1. I/O 引脚不应悬空，I/O 作为输入时可打开内部的上拉或下拉
2. 外部电路从 I/O 引脚灌电流
3. 内部电路从 I/O 引脚拉电流
4. 内部弱上拉的引脚
5. 模块使用 31kHz LIRC
6. 模块使用 LP 振荡器

7.1. 睡眠的唤醒

可以通过下列任一事件将器件从休眠状态唤醒：
1. MCLR 引脚上的外部复位输入（如果使能）
2. BOR 复位（如果使能）
3. POR 复位
4. 看门狗定时器（如果使能）溢出
5. 任何外部中断
6. 能够在休眠期间运行的外设产生的中断（更多信息请参见各个外设）

前 3 个事件会使器件复位，后 3 个事件认为是程序执行的延续。

当执行 SLEEP 指令时，下一条指令（PC+1）被预先取出。如果希望通过中断事件唤醒器件，则必须允许相应的中断允许位。唤醒与 GIE 位的状态无关，如果 GIE 位被禁止，器件将继续执行 SLEEP 指令后的指令。**如果 GIE 位被允许，器件先执行 SLEEP 指令后的指令，然后将调用中断服务程序**。如果不想执行 SLEEP 指令后的指令，用户应该在 SLEEP 指令后面放置一条 NOP 指令。

器件从休眠模式唤醒时，WDT 清零，与唤醒的原因无关。

7.1.1. 使用中断唤醒

当禁止全局中断（GIE 被清零），并且有任一中断源中断标志位置 1 且其中断被使能时，将会发生下列某一事件：

- 如果在执行 SLEEP 指令之前发生中断
  - SLEEP 指令将作为 NOP 执行
  - WDT 和 WDT 预分频器不会清零
  - STATUS 寄存器的 TO 位不会置 1
  - STATUS 寄存器的 PD 位不会清零

- 如果在执行 SLEEP 指令期间或之后发生中断
  - SLEEP 指令将完全执行
  - 器件将立即从休眠模式唤醒
  - WDT 和 WDT 预分频器将清零
  - STATUS 寄存器的 TO 位将置 1
  - STATUS 寄存器的 PD 位将清零

要确定是否执行了 SLEEP 指令，可以测试 PD 位。如果 PD 位置 1，则说明 SLEEP 指令被作为一条 NOP 指令执行了。

7.2. 睡眠的系统时钟

进入睡眠状态后，CPU 时钟停止，PC 停留在 SLEEP 的下一条地址。默认情况下，系统时钟也会被关闭。但如果 SYSON 位置 1 时，系统时钟将一直保持运行，在这种情况下，被选择为系统时钟的 HIRC，XT，LIRC 振荡器将不会被关闭。

注意：如果要使用 FLASH 或数据 EEPROM 的写完成中断唤醒，SYSON 必须置 1。

7.3. 与睡眠模式相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
STATUS	0x03	—	—	—	TO	PD	Z	DC	C	---q quuu
INTCON	0x0B	GIE	PEIE	EEIE	LVDIE	OSFIE	EEIF	LVDIF	OSFIF	0000 0000
PIE1	0x91	—	—	—	—	—	TKIE	CKMIE	ADCIE	---- -000
PIR1	0x11	—	—	—	—	—	TKIF	CKMIF	ADCIF	---- -000
EPIF0	0x14	外部管脚中断标志位								0000 0000
EPIE0	0x94	外部管脚中断使能位								0000 0000

8. 数据 EEPROM 和 FLASH

数据 EEPROM 和闪存程序存储器在正常工作期间（整个 VDD 范围）是可读可写的。这两种存储器没有直接映射到数据存储器空间，而是通过特殊功能寄存器（SFR）间接寻址。有 7 个 SFR 用于访问这两种存储器：

- EECON1
- EECON2
- EECON3
- EEDATL
- EEDATH
- EEADRL
- EEADRH

当与数据 EEPROM 模块接口时，EEDATL 寄存器存放要读写的 8 位数据，EEADRL 寄存器存放被访问的 EEDATL 单元的地址。这些器件具有 256 字节的数据 EEPROM，地址范围从 0h 到 0FFh。

访问程序存储器模块时，EEDATH:EEDATL 寄存器对形成双字节字，存放要读/写的 14 位数据，而 EEADRL 和 EEADRH 寄存器形成双字节字，存放被读取的程序存储单元的 15 位地址。

EEPROM 数据存储器允许以字节为单位进行读写。EEPROM 字节写操作会自动擦除目标存储单元并写入新数据（在写入前擦除）。写入时间由片上定时器控制。写入和擦除电压由片上电荷泵产生，此电荷泵能在器件电压范围内正常工作，用于字节或字操作。

器件能否对程序存储器的特定块执行写操作取决于配置字寄存器位 FSECPB0<7:0>的设置。然而，始终允许程序存储器的读操作。

当器件被代码保护时，调试接口将不再能访问数据或程序存储器。在代码保护时，CPU 仍可继续读写数据 EEPROM 存储器和闪存程序存储器。

8.1. EEADRL 和 EEADRH 寄存器

EEADRH:EEADRL 寄存器对可以寻址最大 256 字节的数据 EEPROM 或最大 32K 字的程序存储器。

当选择程序地址值时，地址的高字节写入 EEADRH 寄存器而低字节被写入 EEADRL 寄存器。当选择 EEPROM 地址值时，只将地址的低字节写入 EEADRL 寄存器。

8.1.1. EECON1 和 EECON2 寄存器

EECON1 是访问 EE 存储器的控制寄存器。
控制位 EEPGD 决定访问的是程序存储器还是数据存储器。当它为 0 时，任何后续操作都将针对 EEPROM 存储器进行。当置 1 时，任何后续操作都将针对程序存储器进行。复位后，EEPGD 清 0，即默认选中 EEPROM。

控制位 RD 和 WR 分别启动读和写。用软件只能将这些位置 1 而无法清零。在读或写操作完成后，由硬件将它们清零。由于无法用软件将 WR 位清零，可避免因意外而过早地终止写操作。

当 WREN 位置 1 时，允许执行写操作。上电时，WREN 位被清零。在正常运行中当写操作被复位中断时，WRERR 位置 1。在这些情况下，复位后用户可以检查 WRERR 位并执行相应的错误处理程序。当写操作完成时，PIR2 寄存器的中断标志位 EEIF 被置 1。该标志位必须用软件清零。

读 EECON2 得到的是全 0。EECON2 寄存器仅在数据 EEPROM 写过程中使用。要使能写操作，必须将特定模式写入 EECON2。

8.2. 使用数据 EEPROM

数据 EEPROM 是高耐久性、可字节寻址的阵列，已将其优化以便存储频繁更改的信息（例如：程序变量或其他经常更新的数据）。软件开发者应将不频繁更改的变量（例如常量、ID 和校准值等）存储在闪存程序存储器中，以免超出 EEPROM 字节最大的写允许次数。

8.2.1. 读数据 EEPROM 存储器

要读取数据存储单元，**用户必须先把 EECON3 的 DRDEN 置 1** 并等待 0.2μs，然后把地址写入 EEADRL 寄存器，清零 EECON1 寄存器的 EEPGD 和 CFGS 控制位，再置 1 控制位 RD。在紧接着的下一个周期，EEDATL 寄存器中就有了数据；因此，该数据可由下一条指令读取。EEDATL 将把此值保持至下一次读取或用户向该单元写入数据时（在写操作过程中）为止。

例 8.2.1，读数据 EEPROM

```
BANKSEL EEADRL        ;
LDWI DATA_EE_ADDR ;
STR EEADRL            ;Data Memory
;Address to read
BCR EECON1, CFGS    ;Deselect Config space
BCR EECON1, EEPGD ;Point to DATA memory
BSR EECON1, RD       ;EE Read
LDR EEDATL, W       ;W =    EEDATL
```

注意：

1. **无论 CPB 为何值，软件总是可以读取 EEPROM；**
2. **所需数据读完后，用户要清 DRDEN 以节省功耗；**

8.2.2. 写数据 EEPROM 存储器

要写 EEPROM 数据存储单元，用户应首先将该单元的地址写入 EEADRL 寄存器并将数据写入 EEDATL 寄存器。然后用户必须按特定顺序开始写入每个字节，且要确保 EECON3.DRDEN 清 0。

如果未完全按照上述顺序（即，首先将 55h 写入 EECON2，随后将 AAh 写入 EECON2，最后将 WR 位置 1）逐字节写入，将不会启动写操作。在该代码段中应禁止中断。

此外，必须将 EECON1 中的 WREN 位置 1 以使能写操作。这种机制可防止由于代码执行错误（异常）导致误写数据 EEPROM。除了更新 EEPROM 时以外，用户应始终保持 WREN 位清零。**WREN 位不能由硬件清零**。

一个写序列启动后，清零 WREN 位将不会影响此写周期。除非 WREN 位置 1，否则 WR 位将无法置 1。写周期完成时，WR 位由硬件清零并且 EE 写完成中断标志位（EEIF）置 1。用户可以允许中断或查询此位。EEIF 必须用软件清零。

注意：
软件对 EECON1.WR 写 1 后，至少等待一个系统时钟（NOP 或者任何别的指令）软件才能对该位进行读判断，否则将读回 0，进而影响程序的流程（比如误认为写结束）。

8.2.3. 防止误写操作的保护措施

有些情况下，用户并不希望向数据 EEPROM 存储器写入数据。为了防止 EEPROM 误写操作，器件内建了各种保护机制。上电时，清零 WREN。同时，上电延时定时器（64ms 的延时）也会阻止对 EEPROM 进行写操作。

写启动序列和 WREN 位共同防止在以下情况下发生意外写操作：

- 欠压
- 电源故障
- 软件故障

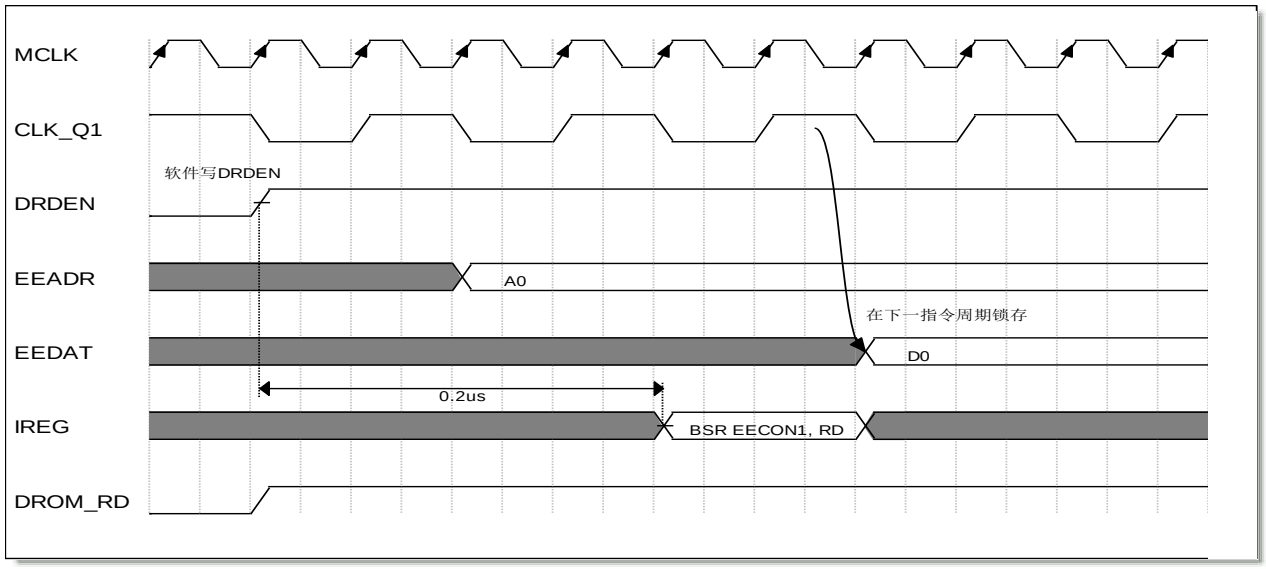


图 8.2.1 2T 模式下软件读 EEPROM 时序

例 8.2.2 写数据 EEPROM

该程序流不能被中断

```
BANKSEL EEADRL              ;
LDWI DATA_EE_ADDR           ;
STR EEADRL                  ;Data Memory Address to write
LDWI DATA_EE_DATA           ;
STR EEDATL                  ;Data Memory Value to write
BCR EECON1, CFGS            ;Deselect Configuration space
BCR EECON1, EEPGD           ;Point to DATA memory
BSR EECON1, WREN            ;Enable writes

BCR INTCON, GIE             ;Disable INTs.
LDWI 55h                    ;
STR EECON2                  ;Write 55h
LDWI 0AAh                   ;
STR EECON2                  ;Write AAh
BSR EECON1, WR               ;Set WR bit to begin write
BSR INTCON, GIE             ;Enable Interrupts
BCR EECON1, WREN            ;Disable writes
BTSC EECON1, WR             ;Wait for write to complete
LJUMP $-2                   ;Done
```

注意：

1. 数据 EEPROM 写周期并不暂停指令的执行。

8.3. 闪存程序存储器概述

了解闪存程序存储器结构对于擦除和编程操作非常重要。闪存程序存储器按行排列。每行包括固定数量的 14 位程序存储器字。行是用户软件可擦除的最小块大小。只有当目标地址位于未受写保护的存储器段内（由配置字寄存器的 CPB 和 FSECPB0 定义），才能对闪存程序存储器进行写或擦除操作。擦除某行后，用户可以对该行的部分或全部进行重新编程。写入程序存储器行的数据将被写入 14 位宽的数据写锁存器中。用户不能直接访问这些写锁存器，但是可以通过对 EEDATH:EEDATL 寄存器对的连续写入来加载写锁存器的内容。

数据写锁存器数并不等于行单元数，**例如 FT62F08X 一行有 64 个单元，但只有 1 个写锁存器**。编程时，用户软件需要填充该组写锁存器并多次启动编程操作，才能完全重新编程擦除的行（页）。例如，具有 64 字的行大小和 1 个写锁存器的器件需要将数据装入写锁存器并启动编程操作 64 次。

注：如果用户只想修改部分以前编程的行，那么必须读取整行的内容并保存在 RAM 中，然后进行擦除。

8.3.1. 读闪存程序存储器

要读程序存储单元，用户必须：
1) 将最低有效地址位和最高有效地址位写入 EEADRH:EEADRL 寄存器对
2) 将 EECON1 寄存器的 CFGS 位清零
3) 将 EECON1 寄存器的 EEPGD 控制位置 1
4) 然后，将 EECON1 寄存器的控制位 RD 置 1

一旦将读控制位置 1，闪存程序存储器控制器将使用第二个指令周期读取数据。这会导致紧跟“BSR EECON1,RD”指令后的第二条指令被忽略。在紧接着的下一个周期，EEDATH:EEDATL 寄存器对中就有数据了，因此可在随后的指令中将该数据读作两个字节。
EEDATH:EEDATL 寄存器对将把此值保存至下一次读操作或用户向该单元写入数据时为止。

注：
1. **要求程序存储器读操作后的两条指令为 NOP。这可以防止用户在 RD 位置 1 后的下一条指令执行双周期指令；**
2. **不管 CP 位的设置如何，软件都可以读取闪存程序存储器；**

例 8.3.1 读程序存储器

```
* This code block will read 1 word of program
* memory at the memory address: PROG_ADDR_HI: PROG_ADDR_LO
* data will be returned in the variables: PROG_DATA_HI, PROG_DATA_LO
BANKSEL EEADRL                  ; Select Bank for EEPROM registers
LDWI PROG_ADDR_LO               ;
STR EEADRL                      ; Store LSB of address
LDWI PROG_ADDR_HI               ;
STR EEADRH                      ; Store MSB of address

BCR EECON1,CFGS                 ; Do not select Configuration Space
BSR EECON1,EEPGD                ; Select Program Memory
BCR INTCON,GIE                  ; Disable interrupts
BSR EECON1,RD                   ; Initiate read
NOP                             ; Executed(Figure 8.3.1)
NOP                             ; Ignored(Figure 8.3.1)
BSR INTCON,GIE                  ; Restore interrupts
LDR EEDATL,W                    ; Get LSB of word
STR PROG_DATA_LO                ; Store in user location
LDR EEDATH,W                    ; Get MSB of word
STR PROG_DATA_HI                ; Store in user location
```

8.3.2. 擦除闪存程序存储器

当执行代码时，程序存储器只能按行擦除。要擦除行：

1) 将要擦除的新行地址装入 EEADRH:EEADRL 寄存器对
2) 将 EECON1 寄存器的 CFGS 位清零
3) 将 EECON1 寄存器的 EEPGD、FREE 和 WREN 位置 1
4) 依次将 55h 和 Aah 写入 EECON2（闪存编程解锁序列）
5) 将 EECON1 寄存器的控制位 WR 置 1，以开始擦除操作
6) 查询 EECON1 寄存器的 FREE 位，以确定行擦除何时结束

参见例 8.3.2。

例 8.3.2 程序存储器的行擦除

```
; This row erase routine assumes the following:
; 1. A valid address within the erase block is loaded in ADDRH:ADDRL
; 2. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)
BCR INTCON,GIE        ; Disable ints so required sequences will execute properly
BANKSEL EEADRL
LDR ADDRL,W           ; Load lower 8 bits of erase address boundary
STR EEADRL
LDR ADDRH,W           ; Load upper 6 bits of erase address boundary
STR EEADRH
BSR EECON1,EEPGD      ; Point to program memory
BCR EECON1,CFGS       ; Not configuration space
BSR EECON1,FREE       ; Specify an erase operation
BSR EECON1,WREN       ; Enable writes

LDWI 55h    ; Start of required sequence to initiate erase
STR EECON2            ; Write 55h
LDWI 0AAh             ;
STR EECON2            ; Write AAh
BSR EECON1,WR         ; Set WR bit to begin erase
NOP                   ; Any instructions here are ignored as processor
                      ; halts to begin erase sequence
NOP                   ; Processor will stop here and wait for erase complete.
                      ; after Erase processor continues with 3rd instruction
BCR EECON1,WREN       ; Disable writes
BSR INTCON,GIE         ; Enable interrupts
```

该程序流不能被中断

在“BSR EECON1,WR”指令后，处理器需要两个周期来设置擦除操作。用户必须在 WR 位置 1 后，执行两条 NOP 指令。处理器将暂停内部操作，通常为 2ms 擦除时间。这不是休眠模式，因为时钟和外设将继续运行。擦除周期后，处理器从 EECON1 写指令后的第三条指令继续操作。

8.3.3. 写闪存程序存储器

使用以下步骤编程程序存储器：

1) 装入要编程的字的起始地址
2) 将数据装入写锁存器
3) 启动编程操作
4) 重复第 1 至 3 步，直到写入所有数据

写入程序存储器之前，要写入的字必须已擦除或者以前未写入过。程序存储器一次只能擦除一行。启动写操作时不会自动擦除。

程序存储器一次只能写入一个字，请参见**图 8.3.2**（对带 1 个写锁存器的程序存储器进行字写操作）。程序存储器写操作完成时，写锁存器将复位为 0x3FFF。

应完成以下步骤，以装载写锁存器和编程程序存储块。可按以下步骤操作：装载数据到写锁存器，启动编程序列。需要特殊的解锁序列才能将数据装入写锁存器或启动闪存编程操作，不应中断此解锁序列。

1) 将 EECON1 寄存器的 EEPGD 和 WREN 位置 1
2) 将 EECON1 寄存器的 CFGS 位清零
3) 将要写入的单元地址装入 EEADRH:EEADRL 寄存器对
4) 依次将 55h 和 Aah 写入 EECON2，然后将 EECON1 寄存器的 WR 位置 1（闪存编程解锁序列）
5) 等待约 2ms 时间，锁存器数据写入程序存储器

例 8.3.3 给出了完整的单字写序列示例。将初始地址装入 EEADRH:EEADRL 寄存器对，数据通过间接寻址载入。

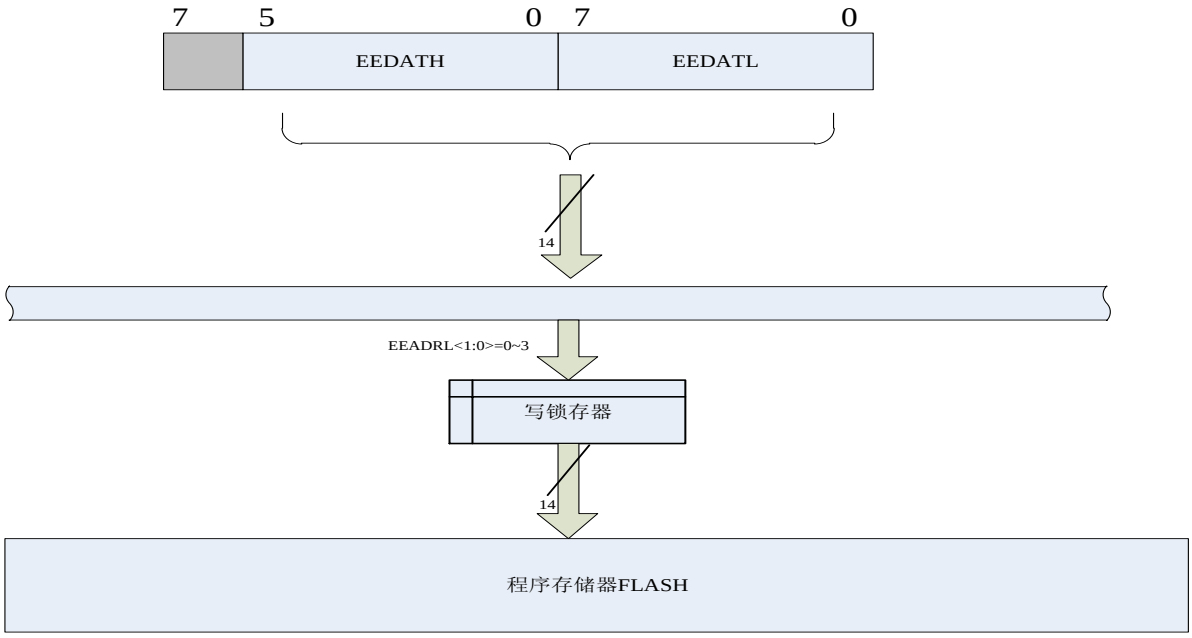


图 8.3.2 对带 1 个写锁存器的 FLASH 进行写操作

注： 例 8.3.3 中提供的代码序列必须重复多次，以完全编程擦除的程序存储器行。
例如，对于 FT62F08X 系列芯片，一行（页）有 64 个 WORD，该代码序列需要重复 64 遍。

例 8.3.3 对包含 1 个写锁存器的 FLASH 编程

```
; This write routine assumes the following:
; 1. The 2 bytes of data are loaded, starting at the address in DATA_ADDR
; 2. Each word of data to be written is made up of two adjacent bytes in DATA_ADDR,
;    stored in little endian format
; 3. A valid starting address is loaded in ADDRH:ADDRL
; 4. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)
;
   BCR INTCON,GIE      ; Disable ints so required sequences will execute properly
   BANKSEL EEADRH      ; Bank 3
   LDR ADDRH,W         ; Load target address
   STR EEADRH          ;
   LDR ADDRL,W         ;
   STR EEADRL          ;
   LDWI LOW DATA_ADDR ; Load initial data address
   STR FSR0L ;
   LDWI HIGH DATA_ADDR ; Load initial data address
   STR FSR0H ;
   BSR EECON1,EEPGD ; Point to program memory
   BCR EECON1,CFGS   ; Not configuration space
   BSR EECON1,WREN  ; Enable writes

   MOVIW FSR0++        ; Load first data byte into lower
   STR EEDATL          ;
   MOVIW FSR0++        ; Load second data byte into upper
   STR EEDATH          ;

START_WRITE

   LDWI 55h            ; Start of required write sequence:
   STR EECON2          ; Write 55h
   LDWI 0AAh           ;
   STR EECON2          ; Write AAh
   BSR EECON1,WR       ; Set WR bit to begin write
   NOP                 ; Any instructions here are ignored as processor
                       ; halts to begin write sequence
   NOP                 ; Processor will stop here and wait for write complete.
                       ; after write processor continues with 3rd instruction

   BCR EECON1,WREN ; Disable writes
   BSR INTCON,GIE      ; Enable interrupts
```

该程序流不能被中断

8.4. 修改闪存程序存储器

修改程序存储器行中的现有数据，并且该行内的数据必须保留时，必须首先读取它并将其保存在 RAM 映像中。

使用以下步骤修改程序存储器：

1) 装入要修改的行的起始地址
2) 从行中读取现有数据并将其保存到 RAM 映像中
3) 修改 RAM 映像以包含要写入到程序存储器的新数据
4) 装入要重新写入的行的起始地址
5) 擦除程序存储器行
6) 将来自 RAM 映像的数据装入写锁存器
7) 启动编程操作

根据需要重复第 6 至 7 步多次，以对擦除行进行重新编程。

8.5. 配置字 UCFGx/FCFGx 读访问

当 EECON1 寄存器中的 CFGS=1 时，不是访问程序存储器或 EEPROM 数据存储器，而是访问配置字 UCFGx。这是 PC<15>=1 时指向的区域，即当软件发起读操作时，地址是[0x8000+EADDR]，但不是所有的地址都可访问，对于未实现的单元，读返回未定义。

8.6. 写校验

根据具体应用，将写入数据 EEPROM 或者程序存储器中的值对照期望写入的值进行校验（见例 9.6.1）是一个良好的编程习惯。例 8.6.1 显示了如何校验对 EEPROM 的写操作。

例 8.6.1 对数据 EEPROM 写校验

```
BANKSEL EEDATL     ;
LDR EEDATL, W      ;EEDATL not changed from previous write
BSR EECON1, RD     ;YES, Read the value written
XORWR EEDATL, W    ;
BTSS STATUS, Z     ;Is data the same
LJUMP WRITE_ERR    ;No, handle error
                   ;Yes, continue
```

8.7. FLASH 内容保护

FLASH 控制器内置加密保护单元，具备以下特性：

- 全区加密，由 CPB 位控制
- 分扇区加密，1 扇区=1k words，由 FSECPB0 寄存器控制
- 解除加密只能通过执行一次包含 UCFG 页在内的全芯片擦除

全加密和分扇区加密区别如下表：

加密方式	CPU 取指	软件读	软件写	串口读	串口写
无	√	√	√(2)	√	√
全区	√	√	√(2)	×(1)	×(4)
分扇区	√	×(1)	×(3)	×(3)	×(5)

注意：

1) EEDAT 保持旧值不变；
2) 软件不可以编程或擦除 UCFG 页；
3) 只可以读未加密的扇区；
4) 只允许串口做包括 UCFG 在内的全芯片擦除；
5) 只允许串口做包括 UCFG 在内的全芯片擦除，或者对未加密的扇区做页擦除，编程；
6) 任何情况下，软件都不可以做全芯片擦除以及 FCFG 区的编程和擦除

8.8. 与 EEPROM 相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
INTCON	0x0B	GIE	PEIE	EEIE	LVDIE	OSFIE	EEIF	LVDIF	OSFIF	0000 0000
EECON1	0x195	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	00-0 x000
EECON2	0x196	EEPROM 写控制寄存器 2								---- ----
EECON3	0x198	—							DRDEN	---- ---0
EEADRH	0x192	—	EEPROM 地址高 7 位							-000 0000
EEADRL	0x191	EEPROM 地址低 8 位								0000 0000
EEDATL	0x193	EEPROM 数据低 8 位								xxxx xxxx
EEDATH	0x194	—	—	EEPROM 数据高 6 位						--xx xxxx

8.8.1. EEDAT 寄存器，地址 0x193, 0x194

EEDATL, SFR 地址 0x193

Bit	7	6	5	4	3	2	1	0
Name	EEDAT[7:0]							
Reset	x	x	x	x	x	x	x	x
Type	RW	RW	RW	RW	RW	RW	RW	RW

EEDATH, SFR 地址 0x194

Bit	7	6	5	4	3	2	1	0
Name	—	—	EEDAT[13:8]					
Reset	—	—	x	x	x	x	x	x
Type	RO-0	RO-0	RW	RW	RW	RW	RW	RW

Bit	Name	Function
15:14	N/A	保留位，读 0
13:0	EEDAT	EEPROM/FLASH 读写数据寄存器 在写周期（约 2ms）内，该寄存器不可写

8.8.2. EEADR 寄存器，地址 0x191, 0x192

EEADRL, SFR 地址 0x191

Bit	7	6	5	4	3	2	1	0
Name	EEADR[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

EEADRH, SFR 地址 0x192

Bit	7	6	5	4	3	2	1	0
Name	—	EEADR[14:8]						
Reset	—	0	0	0	0	0	0	0
Type	RO-0	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
15	N/A	保留位，读 0
14:0	EEADR	EEPROM/FLASH 读写地址寄存器 在写周期（约 2ms）内，该寄存器不可写 **注意：用该寄存器访问程序存储器时，地址范围必须位于 0~0x1FFF，否则无法完成读写访问**

8.8.3. EECON1 寄存器，地址 0x195

Bit	7	6	5	4	3	2	1	0
Name	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
Reset	0	0	0	0	x	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW-1	RW-1

Bit	Name	Function
7	EEPGD	FLASH/数据 EEPROM 存储器选择位 1 = 访问 FLASH 0 = 访问数据 EEPROM 存储器
6	CFGS	FLASH/数据 EEPROM 或配置寄存器选择位 1 = 访问配置寄存器，读访问 0 = 访问 FLASH 或数据 EEPROM 存储器
5	Reserved	**保留位，不要向该位写 1**
4	FREE	FLASH 擦除使能位 当 CFGS=0 且 EEPGD=1（FLASH）： 1 = 在下一条 WR 命令执行擦除操作（擦除完成后由硬件清零） 0 = 在下一条 WR 命令执行写操作 如果 EEPGD=0 且 CFGS=0（访问数据 EEPROM）： **该位不起作用**，下一条 WR 命令将启动一个擦除周期和一个写周期
3	WRERR	EEPROM 错误标志位 1 = 此状态表示试图进行不当的编程或擦除序列 或在写周期过程中被意外终止时，该位自动置 1 终止的事件可以是除 POR 之外的任何复位 0 = 编程或擦除操作正常完成
2	WREN	编程/擦除使能位 1 = 允许执行编程/擦除操作 0 = 禁止编程/擦除 FLASH 和数据 EEPROM **在写周期内，禁止对该寄存器位写**
1	WR	FLASH/EEPROM 写控制位 1 = 启动 FLASH 或数据 EEPROM 编程/擦除操作，软件写 1 后至少要等 1 个系统时钟才能回读 该操作是自定时的，且该位在操作完成时由硬件清零 **用软件只能将 WR 位置 1，但不能清零** 0 = 对闪存或数据 EEPROM 的编程/擦除操作已完成，当前不在进行中
0	RD	FLASH/数据 EEPROM 读控制位 1 = 启动对 FLASH 或数据 EEPROM 的读操作。读操作只占用一个周期 RD 由硬件清零，用软件只能将 RD 位置 1，但不能清零 0 = 不启动 FLASH 或数据 EEPROM 读操作

8.8.4. EECON2 寄存器，地址 0x196

Bit	7	6	5	4	3	2	1	0
Name	EEPROM 控制寄存器 2							
Reset	x	x	x	x	x	x	x	x
Type	WO	WO	WO	WO	WO	WO	WO	WO

Bit	Name	Function
7:0	EECON2	FLASH/数据 EEPROM 写操作解锁控制寄存器 要解锁写操作，在对 EECON1 寄存器的 WR 置位前，必须先写 55h，随后是 Aah。写入该寄存器的值用于解锁写操作。对这些写操作有特殊的时序要求，必须是在连续的指令周期完成

8.8.5. EECON3 寄存器，地址 0x198

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	DRDEN
Reset	—	—	—	—	—	—	—	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW

Bit	Name	Function
7:1	N/A	保留位，读 0
0	DRDEN	数据 EEPROM 读使能 1：允许软件读数据 EEPROM(DROM)，置 1 后至少要等 0.2μs 才能发起 DROM 读 0：禁止软件读数据 EEPROM(DROM)

9. 12bit ADC 模块

模数转换器（Analog-to-digital Converter，ADC）可将模拟输入信号转换为相应的 12 位二进制表征值。该系列器件采用多个模拟输入复用到一个采样保持电路。采样保持电路的输出与转换器的输入相连接。转换器通过逐次逼近法产生 12 位二进制值，并将转换结果保存在 ADC 结果寄存器（ADRESL:ADRESH）中。ADC 参考电压可用软件选择为 VDD、外部参考电压或内部产生的参考电压。ADC 可在转换完成时产生中断。该中断可用于将器件从休眠唤醒。

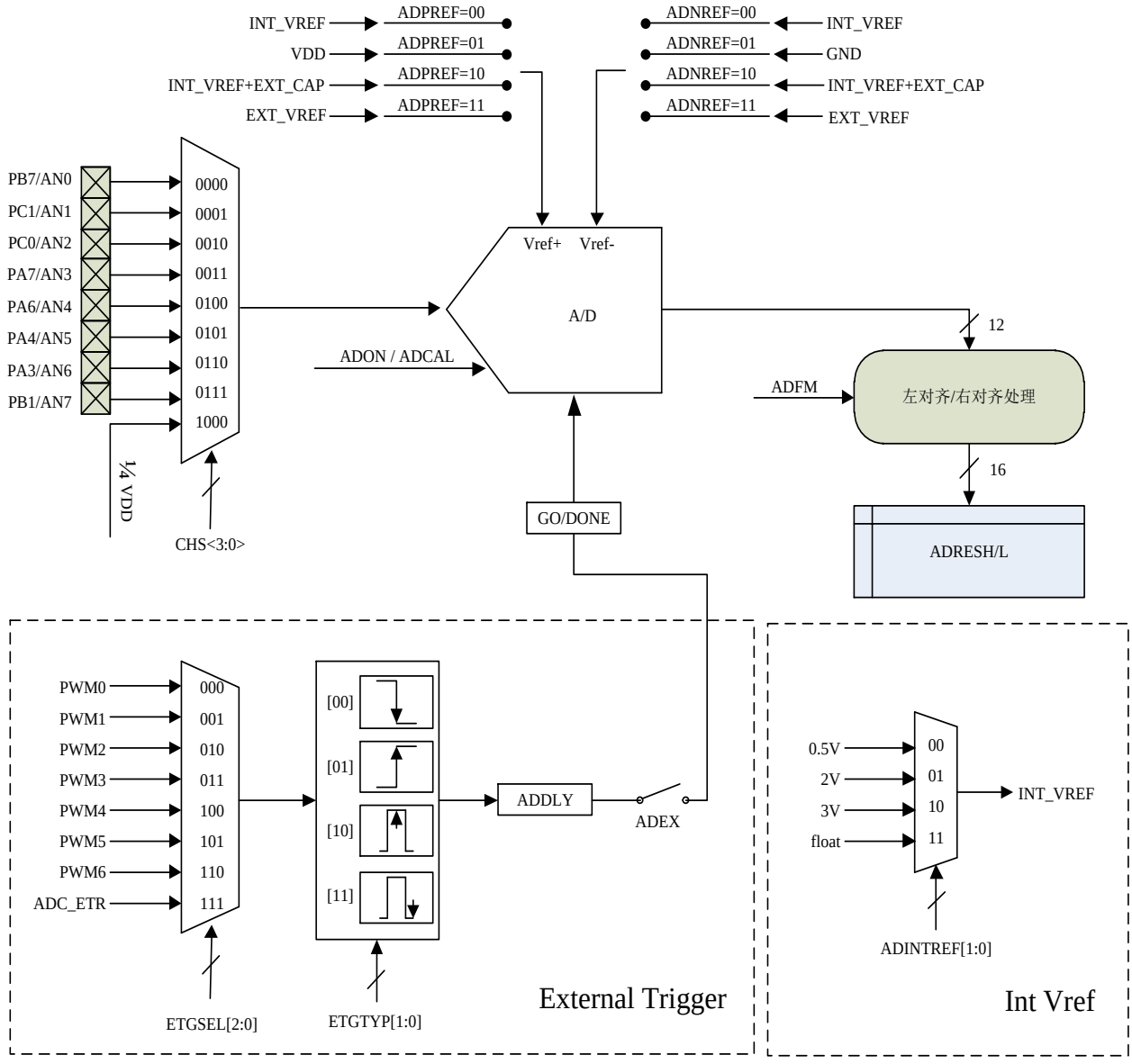


图 9.1 ADC 原理框图

9.1 ADC 的配置

配置和使用 ADC 时，必须考虑以下功能：

- 校准 ADC
- 端口配置
- 通道选择
- 触发方式选择
- 触发源选择
- 触发类型选择
- 触发延时配置
- ADC 参考电压的选择
- ADC 转换时钟源
- 中断控制
- 转换结果的格式
- 阈值比较

注意：在进行各项配置更改的时候，需要确保 AD 转换并未正在进行或外部触发功能未开启。建议在 ADON 关闭时进行更改。

9.1.1校准 ADC

ADC 在启动转换之前建议至少进行一次校准。校准值会一直保存但不可见，任何复位都会导致其丢失。ADCON0 寄存器的 ADCAL 设定为 1 可启动自动校准，不可以与 ADON 同时设定为 1。
自校准实现 ADC 偏移误差的自修正，实现原理是断开输入电压 Vin 选择的端口，将其与内部比较器的负参考电压 Vref-端口连接，以保证输入电压为 Vref-时能够输出零点转换值。

更多信息请参见第 9.2 节“ADC 的工作原理”。

9.1.2端口配置

ADC 可用于转换模拟和数字信号。转换模拟信号时，应将相关的 TRIS 和 ANSEL 位置 1 将 I/O 引脚应配置为模拟功能。更多信息请参见相应的端口章节。

注意：如果定义为数字输入的引脚上存在模拟电压，会导致输入缓冲器传导过大的电流。

9.1.3通道选择

ADCON0 寄存器的 CHS 位决定将哪个通道连接到采样保持电路。改变通道时，根据采样稳定的需要可在启动转换前加入一定延时，硬件已固定有 $1.5T_{AD}$ 的采样延时。更多信息请参见第 9.2 节“ADC 的工作原理”。

9.1.4触发方式选择

ADCON0 寄存器的 ADEX 位决定是否使用外部触发信号。
若 ADEX=0 时，ADGO 可由程序置位，AD 转换完成自动清零。
若 ADEX=1 时，ADGO 将由外部硬件触发置位，AD 转换完成清零。

注意：若选择了前沿消隐触发 ADC，即 LEBADT 设为 1 时，需要先置位 ADEX 和 ADON。

9.1.5触发源选择

在设定 ADEX 后，ADCON2 寄存器的 ETGSEL 位决定使用哪个外部触发信号。其中可选 I/O 引脚触发，需要配置相关寄存器。具体请参见相应的端口章节。

9.1.6触发类型选择

ADCON2 寄存器的 ETGTYP 位决定外部触发信号的触发类型。
其中选择 PWM 的中点或终点类型时，触发源将会默认选择 TIM1 中心对齐的 PWM 输出信号。具体请参见相应的 TIM1 章节。

9.1.7触发延时配置

ADCON2 寄存器的 ADDLY.8 位和 ADDLY 寄存器组成 9 位延时计数器，共同决定外部触发信号的触发延时时间。由于需要同步异步信号，实际延迟时间为：(ADDLY+6)/F_{ADC}。

注意：若选择了前沿消隐触发功能时，则实际延迟时间为：(ADDLY+3)/F_{TIM1} + 3/F_{ADC}。

9.1.8ADC 参考电压

ADCON1 寄存器的 ADPREF 位提供对正参考电压的控制，ADNREF 位提供对负参考电压的控制。正/负参考电压可以是内部参考电压、VDD/GND、内部参考电压加外部电容、外部参考电压。正/负参考电压可以有各种组合，但不可以同时选择内部参考电压。若发生则强制负参考电压选择 GND。
ADCON2 寄存器的 ADINTREF 位提供对内部参考电压的控制。内部参考电压可以选择 0.5V、2V、3V 或者悬空。

9.1.9转换时钟

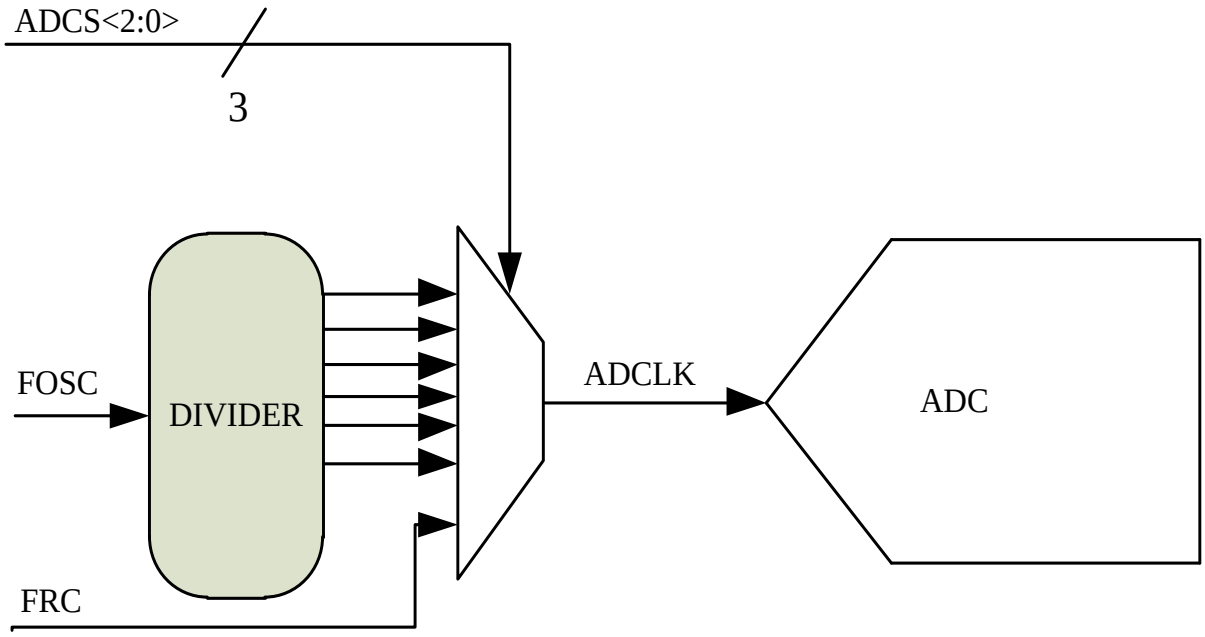


图 9.2 ADC 的时钟配置原理

转换时钟源可通过 ADCON1 寄存器的 ADCS 位用软件选择。有以下 7 种时钟选项：

- $F_{OSC}/2$
- $F_{OSC}/4$
- $F_{OSC}/8$
- $F_{OSC}/16$
- $F_{OSC}/32$
- $F_{OSC}/64$
- F_{RC}（内部慢时钟振荡器）

完成一位（bit）的转换时间定义为 T_{AD}。完成 12 位转换需要 15 个 TAD 周期（包括 $1.5T_{AD}$ 的采样时间和 $1T_{AD}$ 的数据传输处理时间），如图 9.3 和 9.6 所示。

进行正确的转换必须满足相应的 T_{AD} 规范。更多信息请参见第 21 节“电气特性”中的 A/D 转换要求。表 9.1 所示为正确选择 ADC 时钟的示例。

注意：

1. **除非使用的是 F_{RC}，否则任何系统时钟频率的变化均会改变 ADC 时钟频率，这将对 ADC 结果产生负面影响；**
2. **F_{RC} 可以是 256kHz 或者是 32kHz，取决于 LFMOD 为何值；**

ADC 时钟周期（T_{AD}）		系统时钟频率（F_{OSC}）			
ADC 时钟源	ADCS<2:0>	16MHz	8MHz	4MHz	1MHz
F_{OSC} /2	000	125ns	250ns	500ns	2.0μs
F_{OSC} /4	100	250ns	500ns	1.0μs	4.0μs
F_{OSC} /8	001	0.5μs	1.0μs	2.0μs	8.0μs
F_{OSC} /16	101	1.0μs	2.0μs	4.0μs	16.0μs
F_{OSC} /32	010	2.0μs	4.0μs	8.0μs	32.0μs
F_{OSC} /64	110	4.0μs	8.0μs	16.0μs	64.0μs
F_{RC}	x11	4.0μs	4.0μs	4.0μs	4.0μs

表 9.1 ADC 时钟周期和器件工作频率

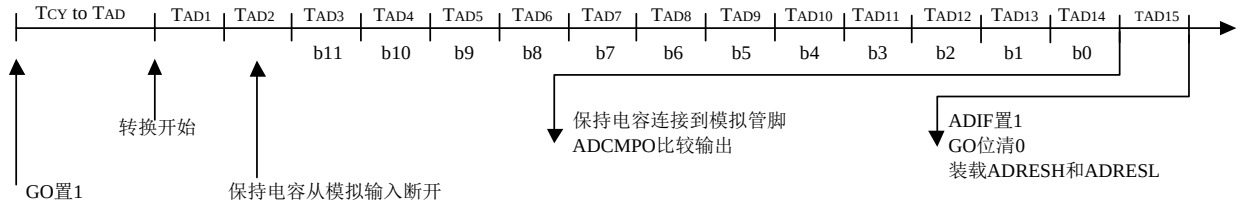


图 9.3 模数转换 T_{AD} 周期

9.1.10 中断

ADC 模块可使中断在模数转换完成时产生。ADC 中断标志为 PIR1 寄存器中的 ADIF 位。ADC 中断使能为 PIE1 寄存器中的 ADIE 位。ADIF 位必须用软件置 1 清零。

注意： 1、无论 ADC 中断是否被打开，ADIF 位在每次正常转换完成时均置 1。
2、自动校准完成、软件停止 AD 转换都不会置位 ADIF。
3、仅当在选择了 F_{RC} 振荡器或打开 SYSON bit 时，ADC 才能在休眠期间工作。

器件工作或处于休眠状态时均可产生中断。如果器件处于休眠状态，中断可唤醒器件。从休眠唤醒时，始终执行 SLEEP 指令后的那条指令。如果用户试图唤醒器件并恢复顺序执行代码，必须禁止全局中断。如果允许全局中断，代码执行将转至中断服务程序。

9.1.11 转换结果的格式

12 位 A/D 转换结果有两种格式，即左对齐和右对齐。ADCON1 寄存器的 ADFM 位控制输出格式。AD 自动校准值也受输出格式影响。

图 9.4 所示为两种输出格式。

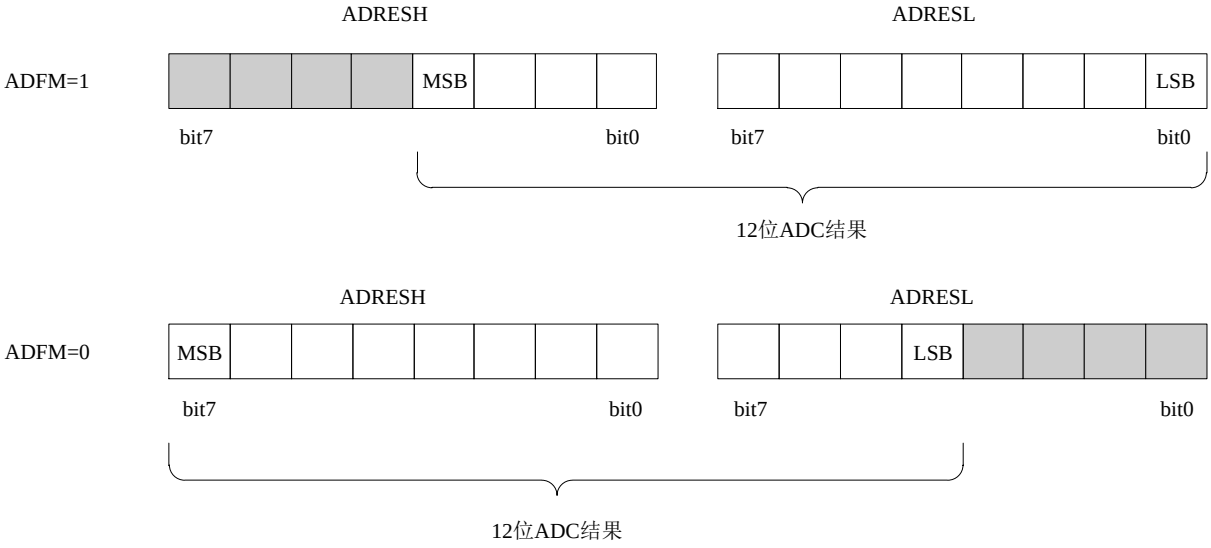


图 9.4 ADC 转换结果格式示意

9.1.12 阈值比较

ADCMPH 寄存器为 ADC 结果比较阈值，ADCON3 寄存器的 ADCMPEN 位控制比较功能使能，ADCMPOP 位控制比较极性，ADCMPO 指示比较结果。
ADC 可以在每次转换完成时进行比较。比较结果会一直保持，直到下次转换完成被更新。ADCMPEN 或 ADON 的清零可以关闭比较功能或 AD 模块，同时可以清零 ADCMPO。进入睡眠不会清零 ADCMPO。

在每次比较完成时可以产生故障刹车事件，由 ADCON3 寄存器的 ADFBEN 控制。

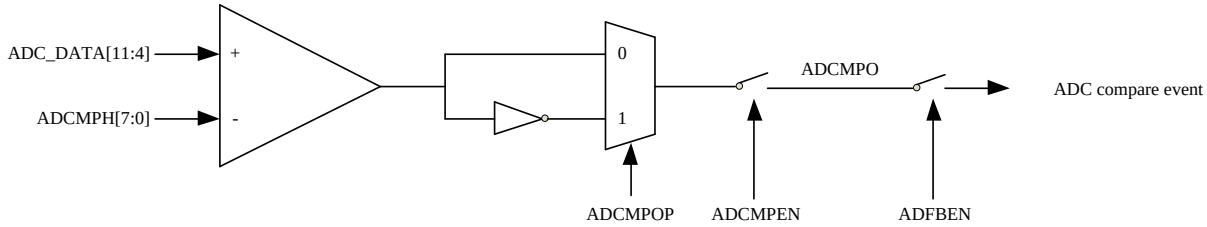


图 9.5 ADC 阈值比较功能框图

9.2 ADC 的工作原理

9.2.1启动自动校准

要启动 ADC 模块的自动校准功能，必须将 ADCON0 寄存器的 ADCAL 位置 1。
校准完成后会自动将 ADCAL 清零，并且将校准值更新到 ADRESH/L。
校准完成后 ADC 模块处于已校准状态，直到器件掉电或复位前均有效。

注意：ADCAL 不能跟 ADON 同时为 1。

校准步骤

1. **查询 ADON 位是否为 1，为 1 则清零；**
2. **配置正确的 VREFP 和 VREFN，这点尤其重要，因为校准结果直接影响后面的 ADC 转换；**
3. **把 ADCAL 位置 1；**
4. **等待 ADCAL 位清 0，至此，自动校准过程结束；**

9.2.2启动转换

要使能 ADC 模块，必须将 ADCON0 寄存器的 ADON 位置 1。
若 ADEX=0 时，将 ADCON0 寄存器的 GO/DONE 位置 1 将启动 AD 转换,。
若 ADEX=1 时，需要外部触发信号才能启动，并且硬件置位 GO/DONE 位，程序置位 ADGO 无效。

注意：

1. **不应在打开 ADC 的那条指令中将 GO/DONE 位置 1。请参见第 9.2.7 节“A/D 转换步骤”**
2. **不应在启动 ADC 转换后或等待外部触发时更改 AD 配置。**
3. **置位 ADGO 后需要等待一个系统周期才可读回 ADGO 标志。**

9.2.3转换完成

转换完成时，ADC 模块将：

- 将 GO/DONE 位清零
- 将 ADIF 标志位置 1
- 用新的转换结果更新 ADRESH:ADRESL 寄存器
- 若使能阈值比较功能，则更新 ADCMPO 比较结果

9.2.4终止转换

如果转换必须在完成前被终止，可用软件将 GO/DONE 清零。ADRESH:ADRESL 将使用部分完成的模数转换结果进行更新。未完成位将用最后转换的一位填充。
终止转换需要处理时间，实际处理时间为 $4/F_{AD}$ ，即在这个时间后才会更新 AD 转换结果，若终止时 AD 还未开始转换，则不更新 AD 转换结果。
终止转换不会产生中断。

注意：器件复位将强制所有寄存器回到其复位状态。这样，ADC 模块就被关闭，并且任何待处理的转换均被终止。

9.2.5休眠模式下 ADC 的工作

ADC 模块可在休眠期间工作，这要求将 ADC 时钟源置于 F_{RC} 选项或打开 SYSON 位。
ADC 需要等待 $4*T_{AD}$ 后才开始转换。这允许软件在设置 ADGO 后，执行一个 SLEEP 指令置 MCU 于 SLEEP 模式，从而降低 ADC 转换期间的系统噪声。通过配置 ADC 时钟为 F_{RC} 和清零 SYSON，可进一步降低系统噪声。
如果允许 ADC 中断，转换完成后器件将从休眠唤醒。如果禁止 ADC 中断，ADC 模块在转换完成后关闭，尽管 ADON 位保持置 1 状态。
如果 ADC 时钟源不是 F_{RC} 并且 SYSON 未打开，执行一条 SLEEP 指令将使当前转换强制中止，ADC 模块被直接关闭，尽管 ADON 位保持置 1 状态。

如果需要在休眠模式下搭配其它模块一起使用，具体请参见相应的功能配置章节，如 TIMER、GPIO、CLK 管理模块。

9.2.6外部触发器

除了通过软件启动 AD 转换外，还可以通过硬件触发方式启动 AD 转换。在 ADEX 置 1 后，可选择 PWM 通道的边沿或周期、管脚边沿等触发信号自动触发启动 AD 转换（硬件自动置位 GO/DONE）。这允许在没有软件介入的情况下，定期进行 AD 转换。
通过 ETGSEL (ADCON2[2:0]) 和 ETGTYP (ADCON2[5:4]) 设置来选择触发源和触发类型。同时，还可以在外部触发信号与启动 AD 转换之间插入触发延时。
在 AD 模块转换过程中(GO/DONE = 1)，任何软件或硬件触发信号都是无效的。若这时停止转换，清零 ADGO 并不会停止触发延时计数。可清零 ADEX 停止触发延时计数。

只有配置 TIMER 为 PWM 输出模式并且使能 PWM 输出时，才会产生 AD 触发信号。更多信息请参见相应的 TIMER 章节。

注意：
当 LEBEN=1 时，外部触发器被禁止。这种情况下，可以选择把 LEBADT 置 1，此时 ADON 和 ADEX 必须设置为 1。在消隐周期结束后会触发一次 AD 转换（硬件自动置位 GO/DONE）。

9.2.7A/D 转换步骤

以下是使用 ADC 进行模数转换的步骤示例：

1. 配置端口：
   - 禁止引脚输出驱动器（见 TRIS 寄存器）
   - 将引脚配置为模拟
2. 配置 ADC 模块：
   - 选择 ADC 转换时钟
   - 配置参考电压
   - 选择 ADC 输入通道
   - 配置触发源、类型及延时
   - 选择转换结果的格式
   - 配置 ADC 结果阈值比较
3. 配置 ADC 中断（可选）：
   - 将 ADC 中断标志清零
   - 允许 ADC 中断
   - 允许外设中断
   - 允许全局中断
4. 进行 ADC 自校准（可选）：
   - 将 ADCAL 置 1 启动自校准
   - 等待并查询 ADCAL 位，为 0 则校准结束
5. 打开 ADC 模块，并等待所需稳定时间 T_{ST}(1)；
6. 将 GO/DONE 置 1 启动转换或等待硬件触发；
7. 等待一个系统周期才可回读 GO/DONE；
8. 通过以下情况之一等待 ADC 转换完成：
   - 查询 GO/DONE 位
   - 等待 ADC 中断（允许中断时）
9. 读取 ADC 结果；
10. 将 ADC 中断标志清零（在允许了中断的情况下这一步是必需的）。

以下是一段示例代码：

```
BANKSEL   TRISB                ;
BSR       TRISB,7              ;Set PB7 to input
BANKSEL   ANSELA               ;
BSR       ANSELA,0             ;Set PB7 to analog
BANKSEL   ADCON1               ;
LDWI      B'11110101'          ;Right justify, ADC Frc clock
STR       ADCON1               ;Vref+ VDD , Vref- GND
BANKSEL   ADCON0               ;
LDWI      B'00000000'          ;Select channel AN0,
STR       ADCON0               ;
BSR       ADCON0,ADCAL         ;Start ADC Self-Calibration
BTSC      ADCON0,ADCAL         ;Is Self-Calibration done?
GOTO      $-1                  ;No, test again
BSR       ADCON0,ADON          ;Turn ADC On
CALL      StableTime           ;ADC stable time
BSR       ADCON0,ADGO          ;Start conversion
NOP                            ;ADGO ReadBack WaitTime
BTSC      ADCON0,ADGO          ;Is conversion done?
GOTO      $-1                  ;No, test again
BANKSEL   ADRESH               ;
LDR       ADRESH,W             ;Read upper 4 bits
STR       RESULTHI             ;store in GPR space
BANKSEL   ADRESL               ;
LDR       ADRESL,W             ;Read lower 8 bits
STR       RESULTLO             ;Store in GPR space
```

注意：

1. T_{ST} 时间是 ADC 的稳定时间(15 个 T_{AD})，每次关闭 AD 转换再启动时需要等待 T_{ST}；

9.3 A/D 采集时间要求

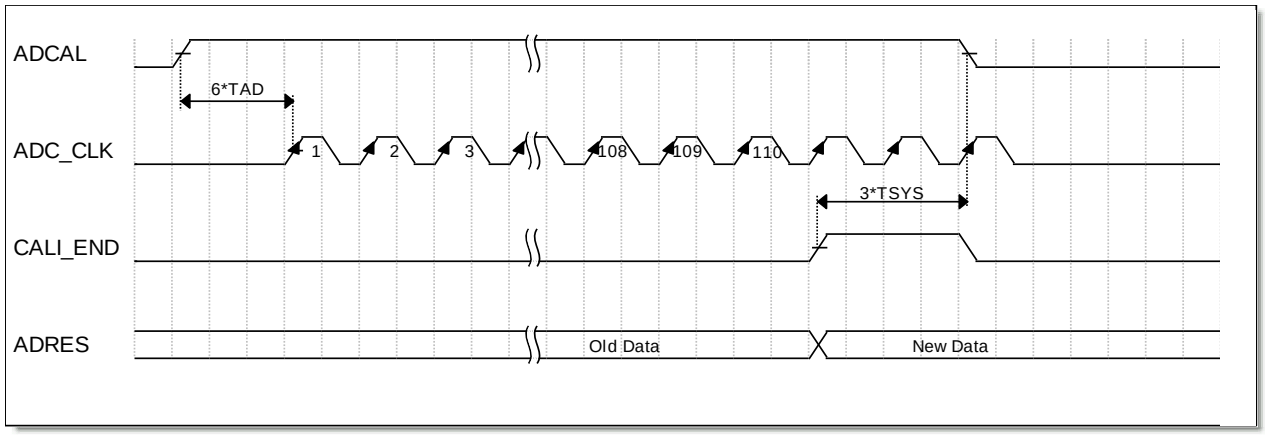


图 9.6 ADC 自校准时序图

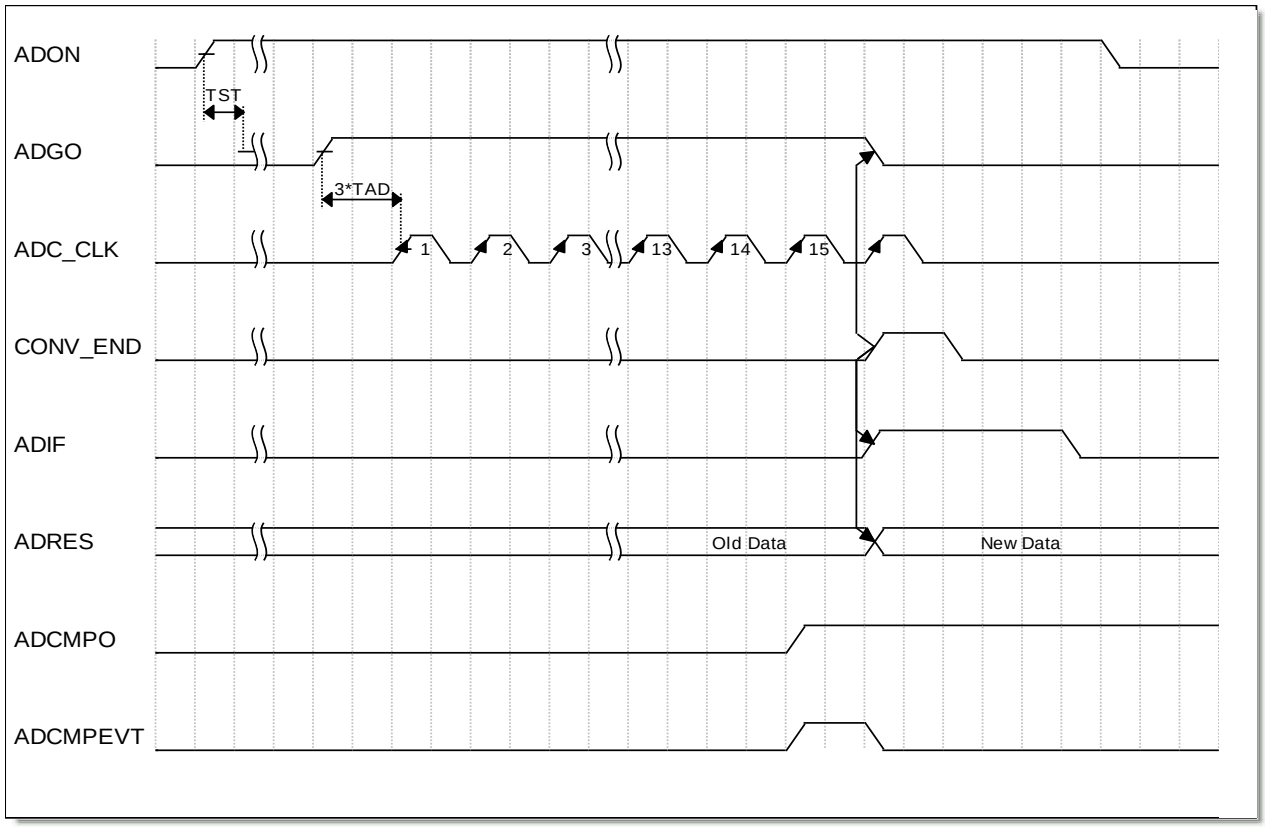
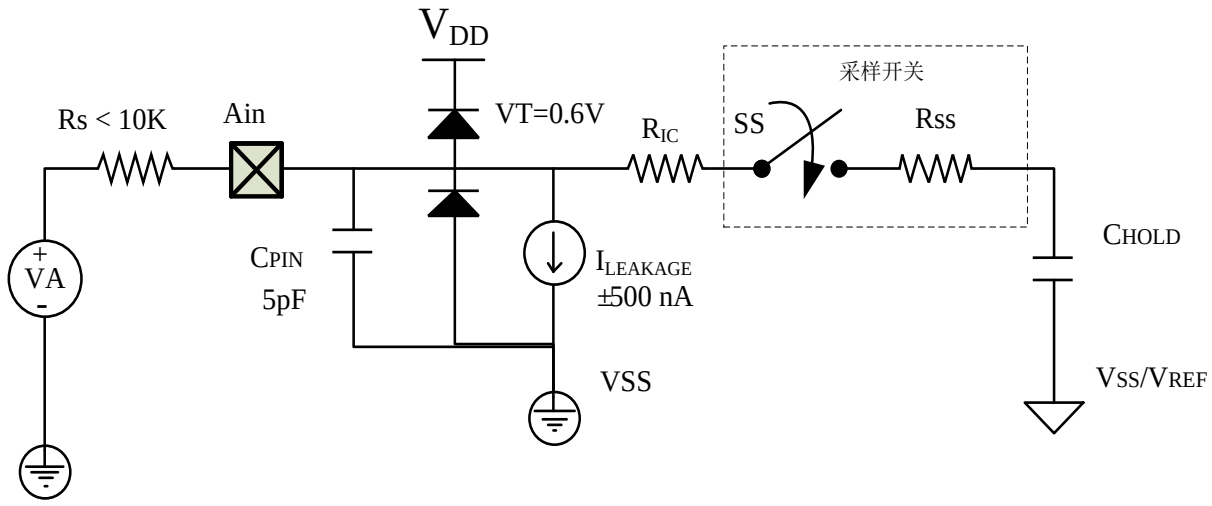


图 9.7 ADC 软件触发转换时序图

为了使 ADC 达到规定的精度，必须使充电保持电容（CHOLD）充满至输入通道的电平。模拟输入模型请参见图 9.8。源阻抗（RS）和内部采样开关（RSS）阻抗直接影响电容 CHOLD 的充电时间。采样开关（RSS）阻抗随器件电压（VDD）的变化而变化，参见图 9.8。建议模拟信号源的最大阻抗为 10kΩ。采集时间随着源阻抗的降低而缩短。在选择（或改变）模拟输入通道后，必须在开始转换前完成采集。



图注：

CPIN = 输入电容

VT =门限电压

ILEAKAGE =结点漏电流

RIC =互联电阻

SS =采样开关

CHOLD =采样保持电容

图 9.8 模拟输入模型

9.4 与 ADC 相关寄存器汇总

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	复位值
ADRESL	0x09B	A/D 结果寄存器的低字节								0000 0000
ADRESH	0x09C	A/D 结果寄存器的高字节								0000 0000
ADCON0	0x09D	CHS<3:0>				ADCAL	ADEX	GO/DONE	ADON	0000 0000
ADCON1	0x09E	ADFM	ADCS<2:0>			ADNREF<1:0>		ADPREF<1:0>		0000 0000
ADCON2	0x09F	ADINTREF<1:0>		ETGTYP<1:0>		ADDLY.8	ETGSEL<2:0>			0000 0000
ADDLY	0x01F	ADDLY<7:0> / LEBPRL<7:0>								0000 0000
ADCON3	0x41A	ADFBEN	ADCMPOP	ADCMPEN	ADCMPO	LEBADT	—	ELVDS<1:0>		0000 0-00
ADCMPH	0x41B	ADCMPH<7:0>								0000 0000
LEBCON	0x41C	LEBEN	LEBCH		EDGS		BKS2	BKS1	BKS0	0000 0000

注：ADCON2、ADDLY、ADCON3、LEBCON 即使在 PCKEN 的 ADCEN 位为 0 时也可以读写。

9.4.1ADRESL，地址 0x9B

Bit	7	6	5	4	3	2	1	0
Name	ADRESL<7:0>							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ADRESL<7:0>	ADC 结果寄存器的低字节 ADFM=0 时，ADRESL[7:4]为 12 位转换结果的低 4 位，其余为 0。 ADFM=1 时，ADRESL[7:0]为 12 位转换结果的低 8 位。

9.4.2ADRESH，地址 0x9C

Bit	7	6	5	4	3	2	1	0
Name	ADRESH<7:0>							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ADRESH<7:0>	ADC 结果寄存器的高字节 ADFM=0 时，ADRESH[7:0]为 12 位转换结果的高 8 位。 ADFM=1 时，ADRESH[3:0]为 12 位转换结果的高 4 位，其余为 0。

9.4.3ADCON0，地址 0x9D

Bit	7	6	5	4	3	2	1	0
Name	CHS<3:0>				ADCAL	ADEX	GO/DONE	ADON
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:4	CHS<3:0>	模拟通道选择位 0000 = AN0 0001 = AN1 0010 = AN2 0011 = AN3 0100 = AN4 0101 = AN5 0110 = AN6 0111 = AN7 1000 = 1/4 V_{DD} 其余保留
3	ADCAL	ADC 自动校准使能位（ADON 为 0 时可设定） 该位由软件设置来启动 ADC 校准。当校准完成后，由硬件清零。 0 = 校准完成。 1 = 写 1 时校准 ADC，读为 1 时意味着校准仍在进行中。
2	ADEX	ADC 触发信号类型选择 该位决定启动 ADC 的触发条件 0 = 当软件设定 GO/DONE 位，启动 AD 转换 1 = 需要外部触发信号触发才可启动 AD 转换，触发事件置位 GO/DONE 位。 外部触发信号条件由寄存器 ETGSEL<2:0>和 ETGTYP<1:0>决定。
1	GO/DONE	AD 转换状态位（硬件触发事件直接置位） 将该位置 1 可启动 A/D 转换周期。当 A/D 转换完成以后，该位由硬件自动清零。 0 = A/D 转换完成/未进行。 1 = A/D 转换正在进行或硬件触发延时正在计数。
0	ADON	ADC 使能位 0 = ADC 被禁止且不消耗工作电流 1 = ADC 被使能

9.4.4ADCON1，地址 0x9E

Bit	7	6	5	4	3	2	1	0
Name	ADFM	ADCS<2:0>			ADNREF<1:0>		ADPREF<1:0>	
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	ADFM	ADC 结果格式选择位 1 = 右对齐。装入转换结果时，ADRESH 的高 4 位被设置为 0。 0 = 左对齐。装入转换结果时，ADRESL 的低 4 位被设置为 0。
6:4	ADCS<2:0>	ADC 转换时钟选择位 000 = F_{OSC}/2 001 = F_{OSC}/8 010 = F_{OSC}/32 011 = F_{RC}（由专用 RC 振荡器提供时钟） 100 = F_{OSC}/4 101 = F_{OSC}/16 110 = F_{OSC}/64 111 = F_{RC}（由专用 RC 振荡器提供时钟）
3:2	ADNREF	ADC 负参考电压配置位（使用 PB6 连接外部参考电压或外部电容） 00 = Int Vref（内部参考电压） 01 = GND 10 = Int Vref + Ext Cap（内部参考电压 + 外部电容） 11 = Ext Vref（外部参考电压）
1:0	ADPREF	ADC 正参考电压配置位（使用 PB5 连接外部参考电压或外部电容） 00 = Int Vref（内部参考电压） 01 = V_{DD} 10 = Int Vref + Ext Cap（内部参考电压 + 外部电容） 11 = Ext Vref（外部参考电压）

9.4.5ADCON2，地址 0x9F

Bit	7	6	5	4	3	2	1	0
Name	ADINTREF<1:0>		ETGTYP<1:0>		ADDLY.8	ETGSEL<2:0>		
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:6	ADINTREF<1:0>	ADC 内部参考电压配置位 00 = 0.5V 01 = 2V 10 = 3V 11 = float（悬空）
5:4	ETGTYP<1:0>	外部触发信号类型选择 当 ADEX 置 1，该位决定响应外部触发的类型 00 = PWM 或 ADC_ETR 脚的下降沿 01 = PWM 或 ADC_ETR 脚的上升沿 10 = 一个 PWM 周期的中点 11 = 一个 PWM 周期的终点 注：PWM 周期中点或终点触发仅适用于中心对齐模式的 PWM 输出
3	ADDLY.8 /LEBPR9	ADC 外部触发延时计数器阈值 第 8 位 详见 ADDLY 寄存器描述
2:0	ETGSEL<2:0>	外部触发源选择 当 ADEX 为 1，该位选择外部触发 ADC 的来源 选择 PWM 源时需要配置 TIMER 为 PWM 输出模式并使能输出。 000 = PWM0 001 = PWM1 010 = PWM2 011 = PWM3 100 = PWM4 101 = PWM5 110 = PWM6 111 = ADC_ETR

9.4.6ADDLY/LEBPRL，地址 0x1F

Bit	7	6	5	4	3	2	1	0
Name	ADDLY<7:0>							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ADDLY<7:0>	ADC 外部触发启动延时计数器阈值低位 该 8 位寄存器与 ADCON2.7 组成 9 位计数器，用于在外部触发启动 ADC 之前加入一段延迟。延迟计数器结束再开始 ADC 转换 外部延迟时间 = (ADDLY+6)/F_{ADC} 注，该延时仅当 ADEX 置 1 时有效。如果启用 PWM 输出触发 ADC 功能，在 PWM 运行过程中不得更改 ADDLY 计数值。同时复用为前沿消隐计数阈值

9.4.7ADCON3，地址 0x41A

Bit	7	6	5	4	3	2	1	0
Name	ADFBEN	ADCMPOP	ADCMPEN	ADCMPO	LEBADT	—	ELVDS	
Reset	0	0	0	0	0	—	0	0
Type	RW	RW	RW	RO	RW	RO-0	RW	RW

Bit	Name	Function
7	ADFBEN	ADC 比较结果响应故障刹车使能 0 = 禁止 1 = ADC 触发故障刹车功能使能
6	ADCMPOP	ADC 比较器输出极性选择位 0 = 若 ADC 结果的高八位大于或等于 ADCMPH[7:0]，ADCMPO 为 1 1 = 若 ADC 结果的高八位小于 ADCMPH[7:0]，ADCMPO 为 1
5	ADCMPEN	ADC 结果比较使能位 0 = ADC 结果比较功能关闭 1 = ADC 结果比较功能打开
4	ADCMPO	ADC 比较结果输出位 该位输出 ADCMPOP 设定的比较输出结果。每次 AD 转换结束都会更新输出
3	LEBADT	前沿消隐周期结束后，ADC 触发使能 1 = 触发 ADC 转换 0 = 不触发 ADC 转换
2	N/A	保留：读为 0
1:0	ELVDS	外部 LVD 管脚输入选择，只有当 LVDM 为 1 时才有效 00 = ELVD0 01 = ELVD1 10 = ELVD2 11 = ELVD3

9.4.8ADCMPH，地址 0x41B

Bit	7	6	5	4	3	2	1	0
Name	ADCMPH<7:0>							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ADCMPH<7:0>	ADC 比较阈值 仅 8 位，用于 ADC 结果高 8 位比较。

10. 高级定时器 TIM1

10.1. 特性

- 16bit 的向上计数、向下计数或者上/下计数器，支持自动重载；
- 支持可编程预分频的计数时钟；
- 支持 4 个独立的捕捉比较通道，通道可支持：
  - 输入捕捉
  - 输出比较
  - 边沿或中心对称 PWM
  - 单脉冲输出
  - 6 步 PWM
- PWM 互补输出和可编程死区时间；
- 可编程的重复计数器；
- 刹车功能，使输出停止在一个复位态或者一个预设状态
- 中断事件：
  - 更新事件：计数器溢出，计数器初始化
  - 触发事件：触发计数开始与停止，计数器初始化或外部触发事件
  - 输入捕捉事件
  - 输出比较事件
  - 刹车输入有效事件
- 外部时钟的触发计数
- 前沿消隐

10.2. 原理框图

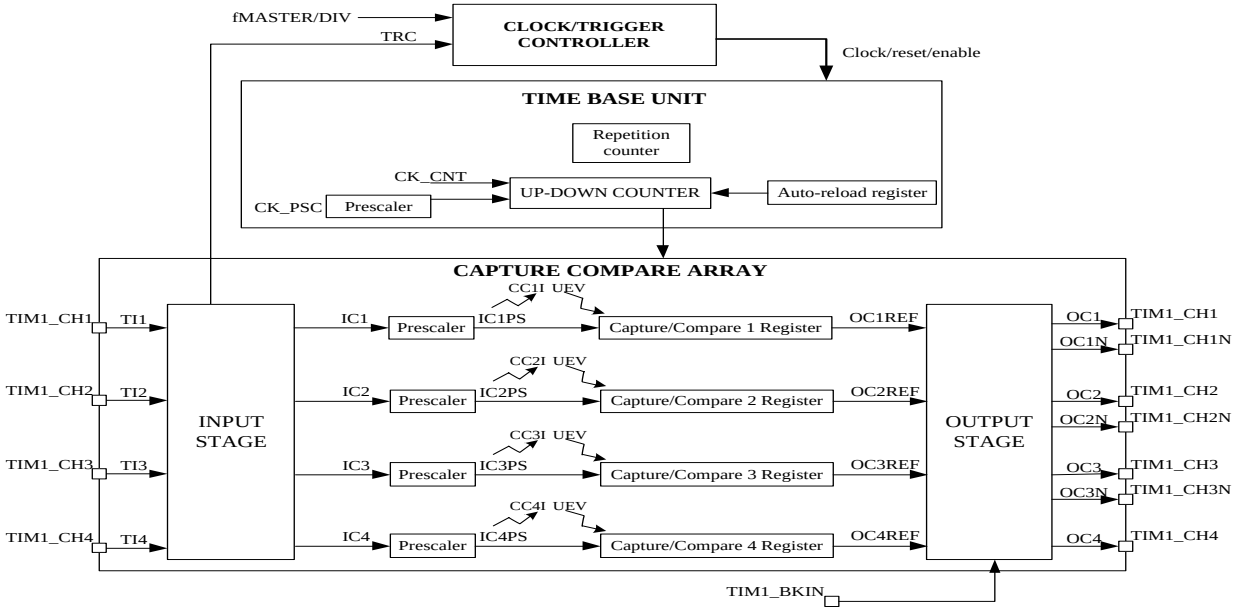


图 10.1 TIM1 原理框图

10.3. 功能描述

整个 TIM1 可以分为三个大的功能部分：计数基本单元、计数控制和捕捉比较通道。计数基本单元分为向上/向下计数器、自动加载寄存器、重复计数器和预分频器；计数控制器又分为计数触发源，模式控制；捕捉比较通道分为捕捉输入通道，输出比较通道，死区产生和输出控制。

10.3.1. 计数基本单元

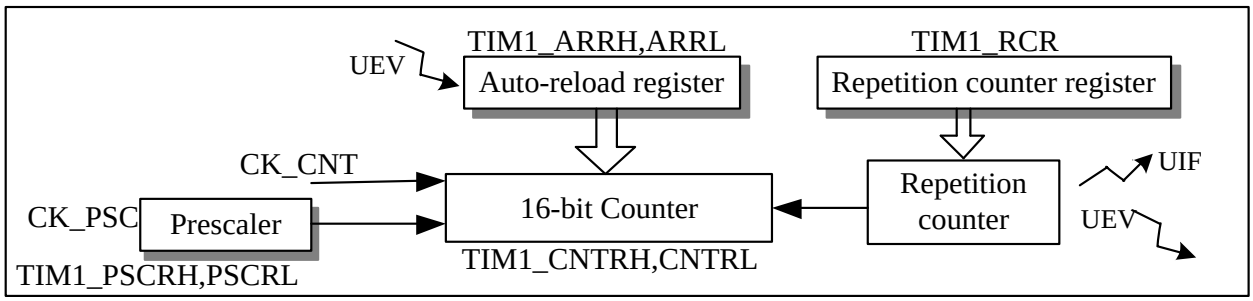


图 10.2 计数基本单元

16 位计数器，预分频器，自动重载寄存器和重复计数寄存器都能由软件进行读写。

10.3.1.1. 计数基本单元组成

10.3.1.1.1. 16 位计数器

16 位计数器的读写：

- TIM1CNTRH/L 能在任何时候进行写操作；但是建议为了避免出现不正确的中间状态，不要在计数器运行的时候进行写操作
- TIM1CNTRH/L 的写操作是没有顺序限制的；可以先写高位也可以先写低位
- TIM1CNTRH/L 能在任何时候进行读操作；但是因为此设计是异步设计，所以在计数器运行期间进行读操作可能读出不正确的数值，需要读两次，比较两次数值是否一致；如果一致，则读出的数值是正确的数值；否则，读出数值是错误的。

10.3.1.1.2. 预分频器

计数时钟可以进行 16bit 的时钟预分频，分频系数为 1~65536。

下列是计算计数器时钟分频的公式：
$f_{CK\_CNT} = f_{CK\_PSC}/(PSCR[15:0]+1)$；PSCR 为实际装入预分频器影子寄存器的值

预分频支持分频自动更新，即在更新事件发生后，能够自动改变预分频值。当 T1CEN 为 0 时，写入预分频寄存器的值也能直接加载实际应用的预分频寄存器中。

配置步骤示例：
1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 配置计数周期
3. 配置占空比
4. 配置预分频
5. 使能计数器

示例对应时序图：

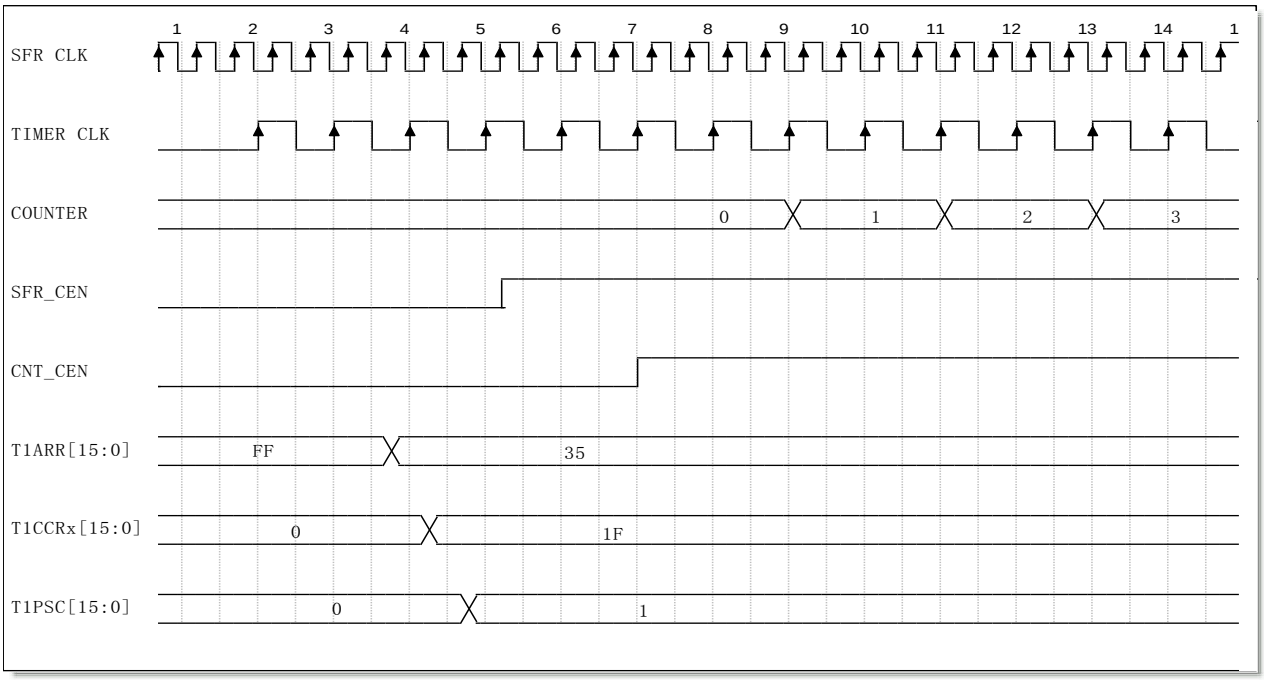


图 10.3 预分频为 1 时的计数器计数图

注意：
在配置的时候，需要先将周期，占空比，模式等寄存器配置完成后并且在 T1CEN 使能之前配置预分频寄存器。

10.3.1.1.3. 自动重载寄存器

自动重载寄存器由一个预加载寄存器和一个影子寄存器组成。
写自动重载寄存器的三种方式：

- 方式 1：计数器使能位打开并且周期预加载使能(T1ARPE=1)。在这种模式下，写入自动重载寄存器的值保存在预加载寄存器中，并在下一个更新事件到来时传送到影子寄存器中进行使用。如下图所示：

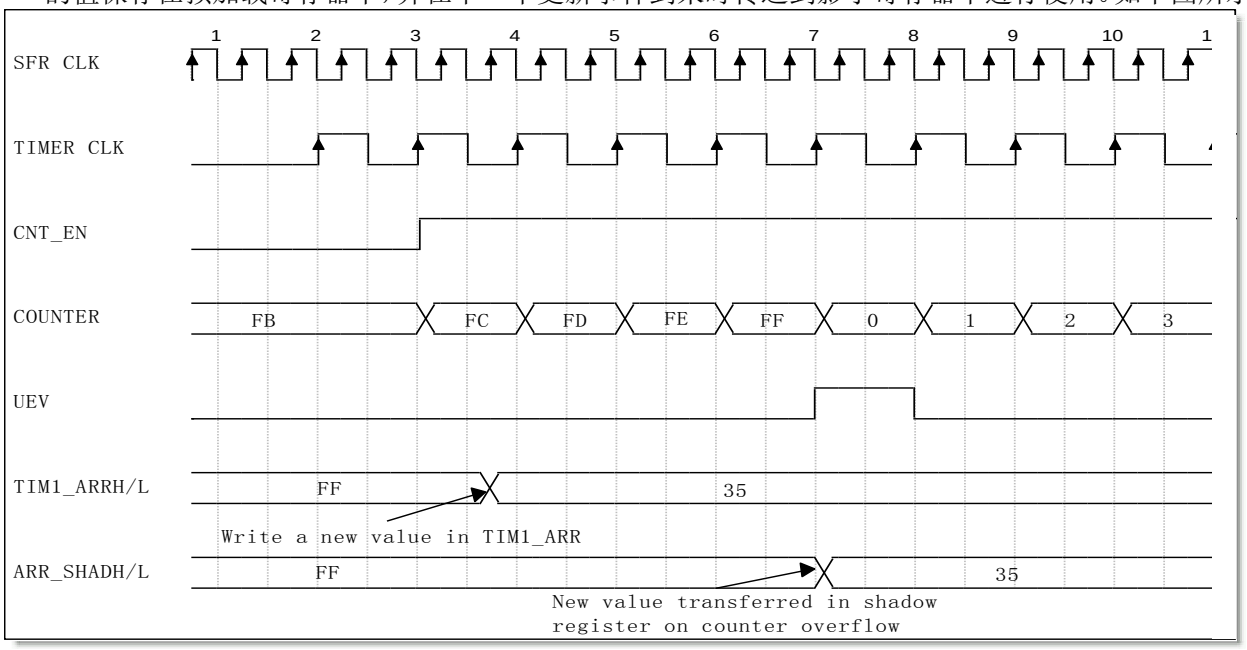


图 10.4 T1CEN=1 且 T1ARPE=1，周期寄存器(T1ARR)加载图

- 方式 2：计数器使能位打开并且周期预加载关闭(T1ARPE=0)。在这种模式下，写入自动重载寄存器的数值直接传送到影子寄存器中进行使用。如下图所示：

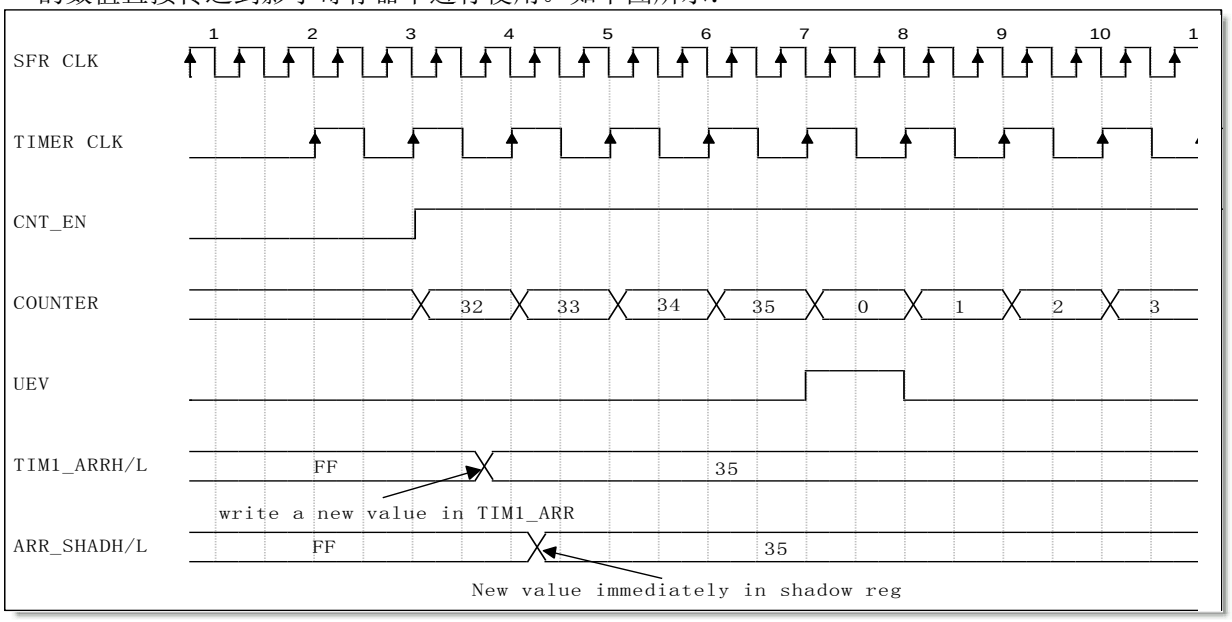


图 10.5 T1CEN=1 且 T1ARPE=0，周期寄存器(T1ARR)加载图

- 方式 3：当计数器使能位(T1CEN)关闭时，不管周期预加载(T1ARPE)使能还是关闭，写入自动重载寄存器的数值直接传送到影子寄存器中进行使用。如下图所示：

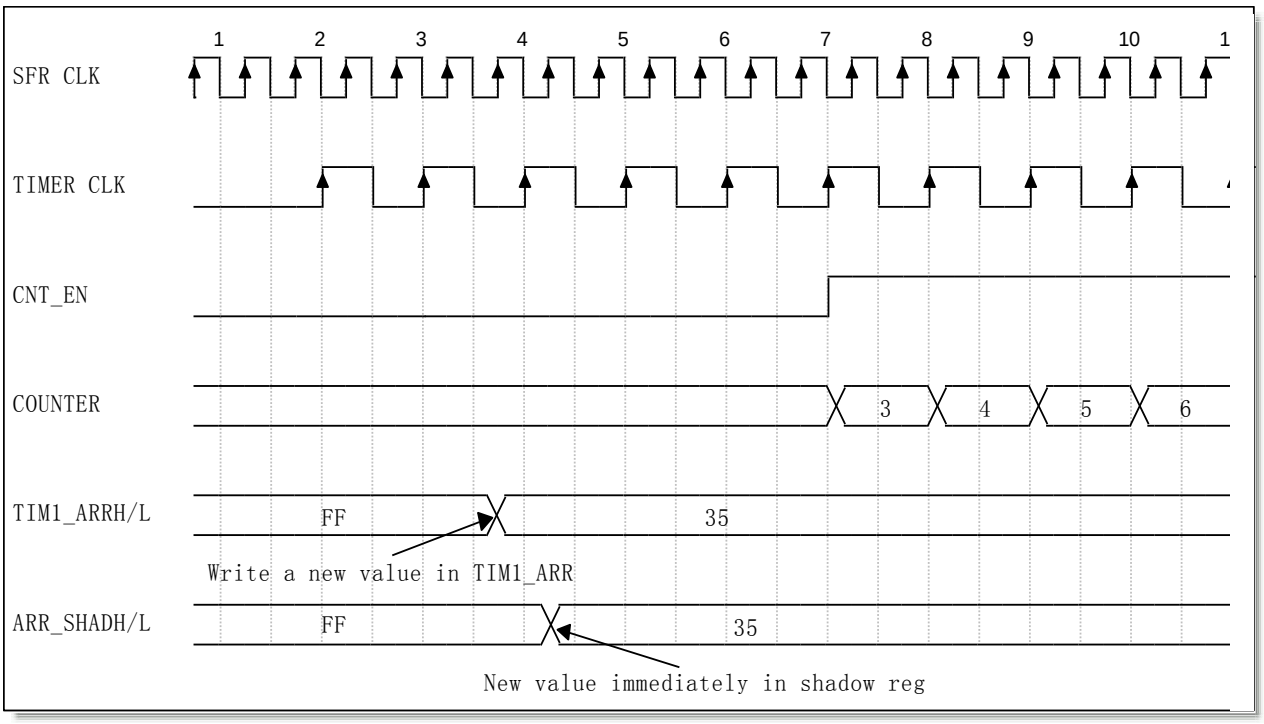


图 10.6 T1CEN=0，周期寄存器(T1ARR)加载图

10.3.1.1.4. 更新事件

更新事件的产生：

- 计数器上溢或下溢
- 配置为复位模式(必须在输入捕捉模式下才能配置为复位模式)时，触发事件的到来

更新事件的影响：

- 影响 1：某些预加载的寄存器(具体寄存器可查看寄存器表格)在预加载使能的情况下都能被更新为最新值。各类预加载寄存器的总结如下表所示：

在更新事件下，可进行预加载的寄存器	TIMARRH/L	TIM1PSCRH/L	TIM1CCRxH/L
相应的预加载使能位	T1ARPE	没有使能位，预加载在计数器使能(T1CEN=1)时一直有效	T1OCxPE

表 10.1 更新事件相关的预加载寄存器 vs 预加载使能位

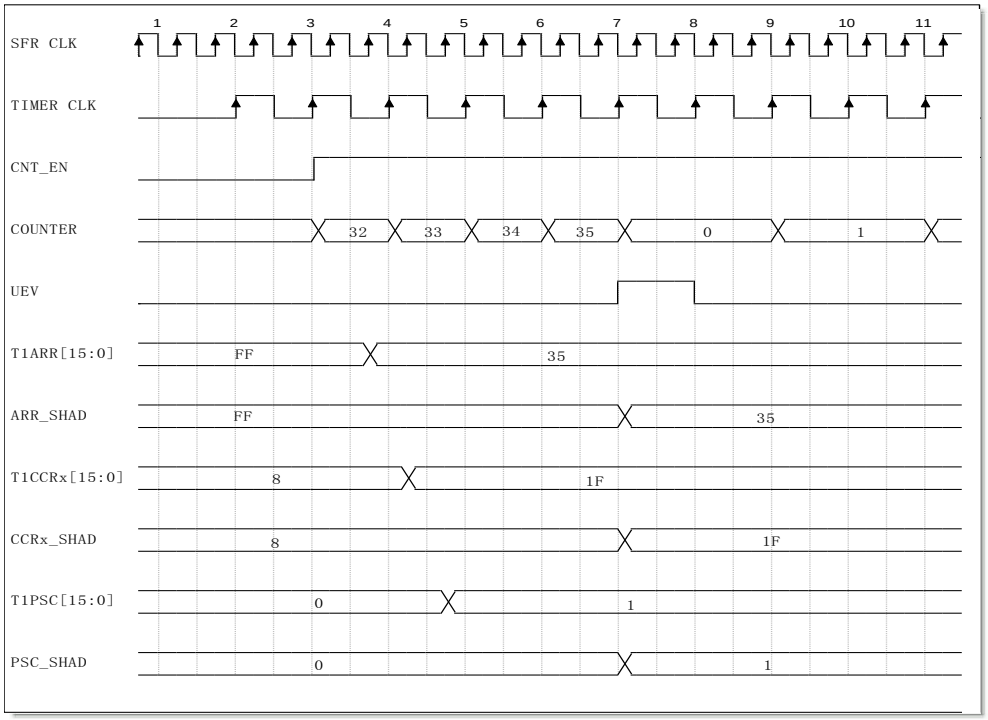


图 10.7 更新事件下，预加载寄存器的更新图

- 影响 2：若 T1UDIS=0，当产生更新事件时，更新标志位(T1UIF)被置位；反之，T1UDIS=1 时，不产生更新事件，更新标志位(T1UIF)也不会被置位。如下图所示：

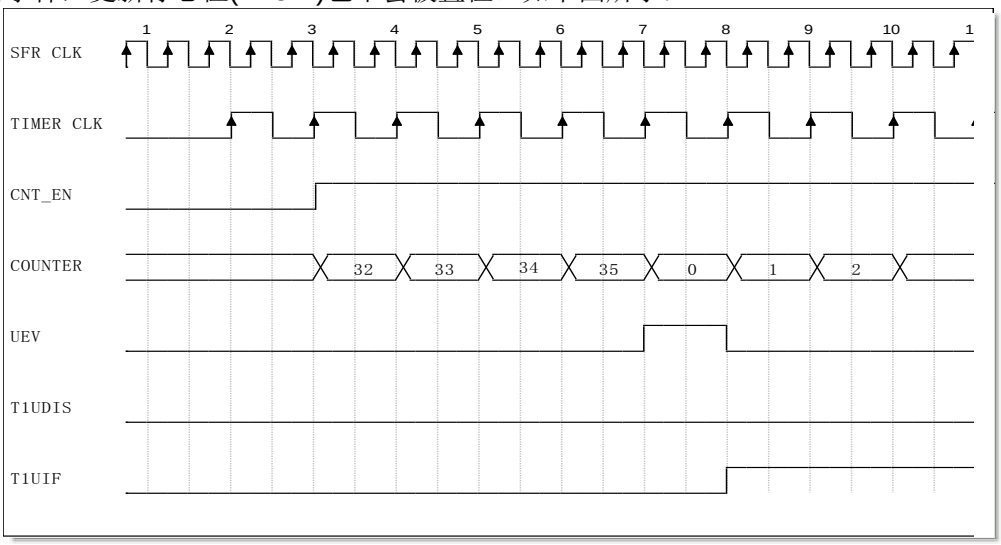


图 10.8 更新事件下且 T1UDIS=0，更新标志位变化图

- 影响 3：单次脉冲模式下，更新事件的到来会使计数器使能位(T1CEN)关闭，计数器停止计数。关于单次脉冲模式的详细说明可查看 10.3.3.3 章节内容。

- 影响 4：故障事件撤消后，如果 T1AOE=1，PWM 将在更新事件后恢复正常输出。关于故障刹车事件的详细说明可查看 10.3.5 章节内容。

10.3.1.2. 向上计数模式

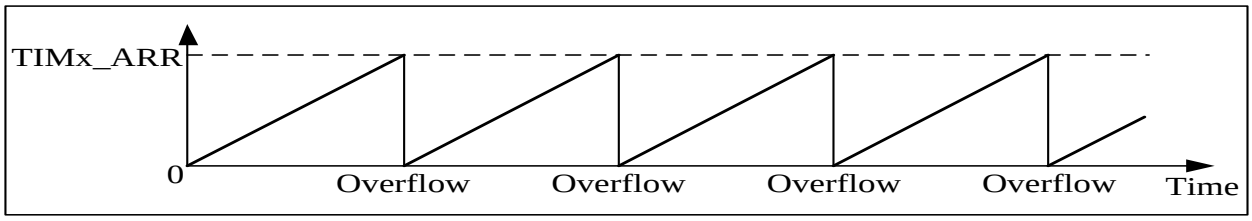


图 10.9 向上计数模式

在向上计数模式中，计数器从 0 开始计数向上计数，计到 TIM1_ARR 寄存器所设数值。然后重新从 0 开始计数并产生一个计数器上溢事件；如果 T1UDIS 设为 0，那么还会产生一个更新事件 UEV。

10.3.1.3. 向下计数模式

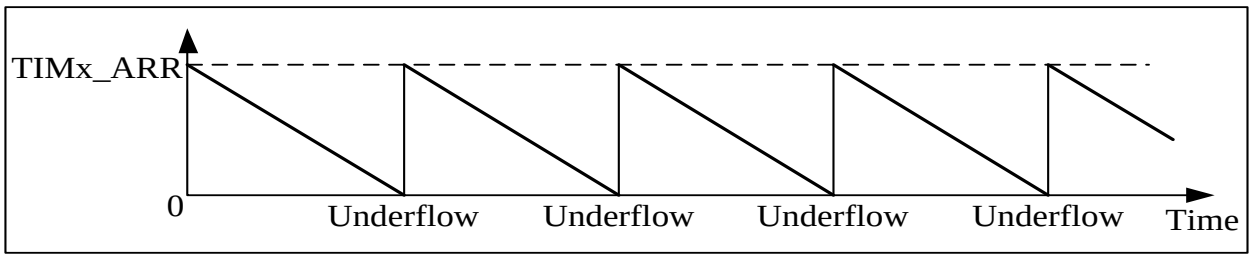


图 10.10 向下计数模式

在向下计数模式中，计数器从 TIM1_ARR 寄存器设置的自动重载值开始向下计数，直到计到 0。然后重新从自动重载值开始计数并产生一个计数器下溢事件；如果 T1UDIS 设为 0，那么还会产生一个更新事件 UEV。

10.3.1.4. 中心对齐模式

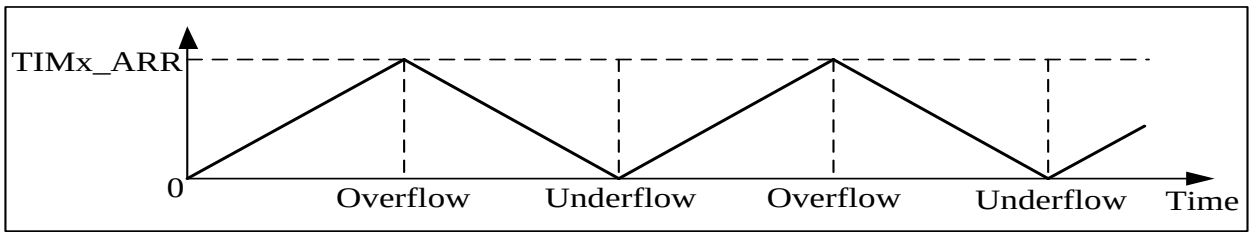


图 10.11 中心对齐模式，T1DIR 初始化为 0

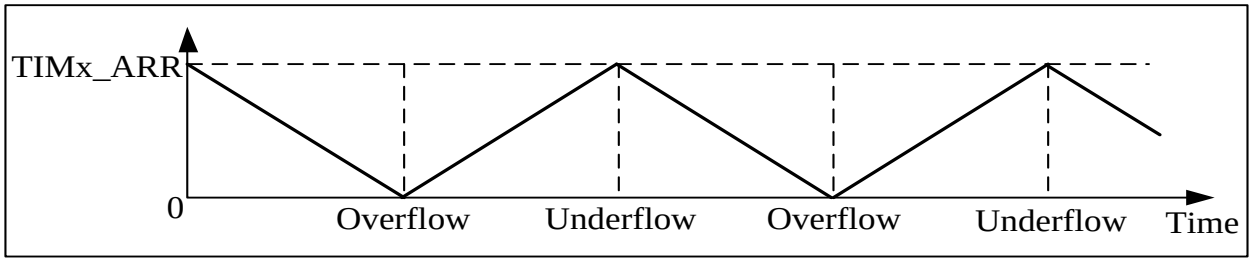


图 10.12 中心对齐模式，T1DIR 初始化为 1

在中心对齐模式中，计数器从 0 开始向上计数，计到自动重载值。这时会产生一个计数器上溢事件。然后计数器开始向下计数计到 0，产生一个下溢事件。计数器不断地重复上述的计数过程。

在这个模式下，方向位(T1DIR)不能进行写操作。方向位会由硬件设置成当前计数器的计数方向。

中心对齐模式所需注意事项：

- 当在中心对齐模式下开始计数时，当前的配置会被使用 – 计数开始值为写入 TIM1CNTRH/L 中的值，计数开始方向决定于写入 TIM1CR1 寄存器中的 T1DIR 位。注意 T1DIR 位和 T1CMS 值不能被软件同时改写。
- T1DIR 位在 T1CMS 不等于 00 时，为只读寄存器，无法进行写操作；所以如果想要配置计数器的初始计数方向，需要先配置计数方向(T1DIR)，再配置计数模式(TICMS)。
- 运行在中心对齐模式下时，不建议写计数器值(TIM1CNTRH/L)，因为可能会产生意想不到的结果。
  如果写入计数器的值大于自动加载值(TIM1_CNT > T1ARR)，计数方向可能不会进行更新。
  如果写入计数器的值为 0 或为 T1ARR，计数方向会进行更新但更新事件(UEV)不会产生。

配置步骤示例：

1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 配置计数周期预加载使能(T1ARPE=1)
3. 配置计数周期(T1ARR=06H)，占空比
4. 配置初始计数方向为向上计数(T1DIR=0)
5. 配置计数模式为中央对齐模式 1(T1CMS=01)
6. 配置预分频(T1PSC=0)
7. 使能计数器

示例对应时序图：

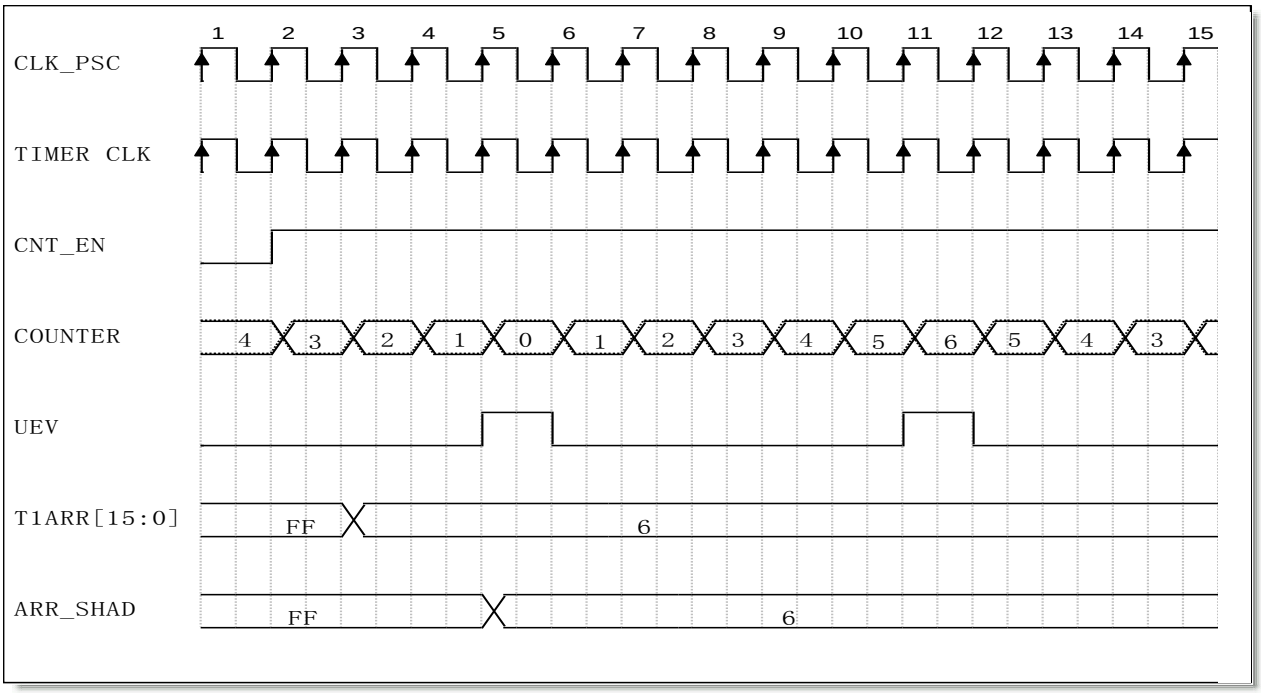


图 10.12 中心对齐模式下，计数时序图

10.3.1.5. 重复向下计数器

重复计数器是 8bit 的向下计数器，会在每次 TIMER 上溢或下溢时-1；只有当重复向下计数器减到 0 时，计数器上溢或下溢才会产生更新事件(UEV)；使用重复计数器能够设定更新事件的频率，这在产生特定数量 PWM 信号时非常有用，如图 10.14 所示。

重复向下计数器自减事件：

- 计数器向上计数模式下，每个计数上溢事件都会使重复计数器减 1。
- 计数器向下计数模式下，每个计数下溢事件都会使重复计数器减 1。
- 计数器中心对齐模式下，每个计数上溢或下溢事件都会使重复计数器减 1。

重复向下计数器是自动重载的，当发生了更新事件(UEV)时，会将 TIM1RCR 寄存器中的值自动重载到重复向下计数器中。如下图所示：

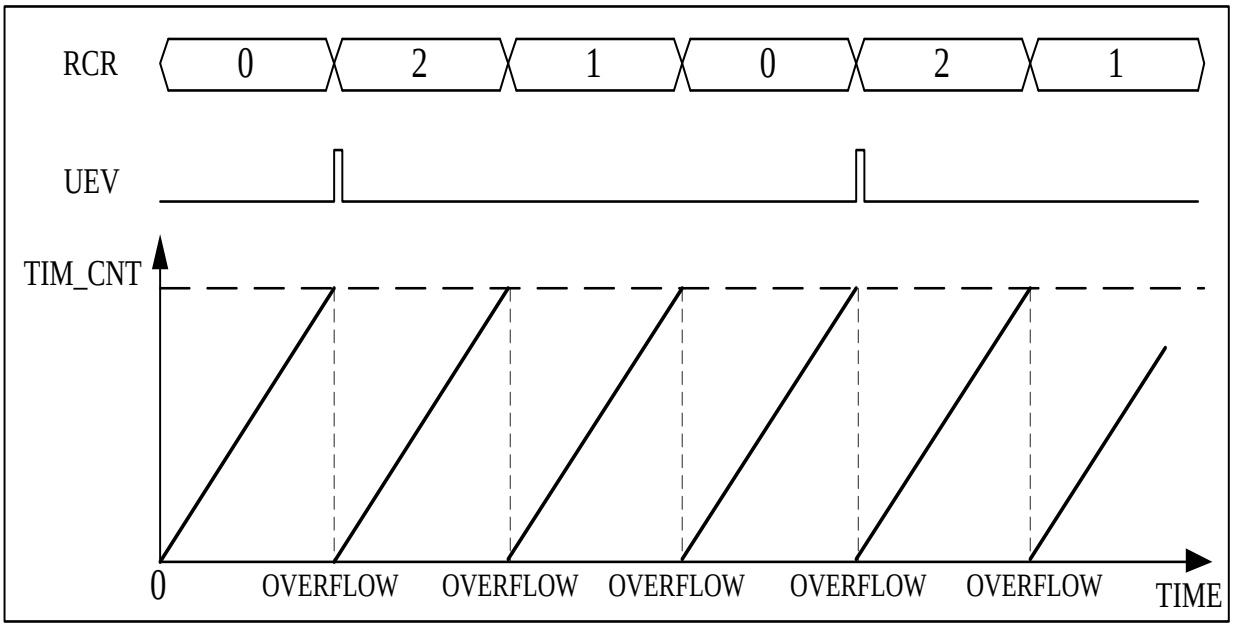


图 10.13 T1REP=2，重复计数器计数时序图

配置产生特定个数 PWM 信号的步骤示例：

1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 配置 TIM1 通道相对应的端口为输出端口
3. 开启更新事件中断
4. 配置计数周期(T1ARR)，占空比(T1CCRx)
5. 需要开启周期预加载(T1ARPE)和占空比预加载功能(T1OCxPE)
6. 配置计数方向为向上计数(T1DIR=0)
7. 配置输出比较模式(T1OCxM=3’b111)为 PWM2 输出模式，并配置通道使能
8. 打开自动主输出使能(T1AOE=1)位
9. 使能计数器
10. 在更新事件中断中，重新更改计数周期，占空比等配置

以下是一段示例代码：

```
BANKSEL     PCKEN
BSR         PCKEN,0     ; 使能 TIM1 模块时钟
BANKSEL     INTCON
LDWI        H'C0'
STR         INTCON      ; 开启全局中断使能和外设中断使能
BANKSEL     TCKSRC
LDWI        H'01'
STR         TCKSRC      ; 选择 TIM1 时钟源为 HIRC
BANKSEL     TRISA
LDWI        H'FE'
STR         TRISA       ; 配置 PA0 为通道 1 的输出通道
BANKSEL     TIM1ARRL
LDWI        H'1F'
STR         TIM1ARRL    ; 将输出波形周期配置为 32
LDWI        H'10'
STR         TIM1CCR1L   ; 将输出波形占空比配置为 16
LDWI        H'02'
STR         TIM1RCR     ; 将重复计数器配置为 2
BSR         TIM1BKR,6   ; 打开自动主输出使能位
BANKSEL     TIM1CCMR1
LDWI        H'70'
STR         TIM1CCMR1   ; 配置通道 1 为 PWM2 模式输出
BSR         TIM1IER,0   ; 开启更新事件中断
LDWI        H'01'
STR         TIM1CCER1   ; 使能通道 1
BANKSEL     TIM1CR1
LDWI        H'81'       ; 开启计数器计数使能位
STR         TIM1CR1     ; 开启计数器计数使能位和周期预加载使能位
INT:
BANKSEL     TIM1ARRL
LDWI        H'14'
STR         TIM1ARRL    ; 将输出波形周期配置为 20
```

上述示例对应示意图：

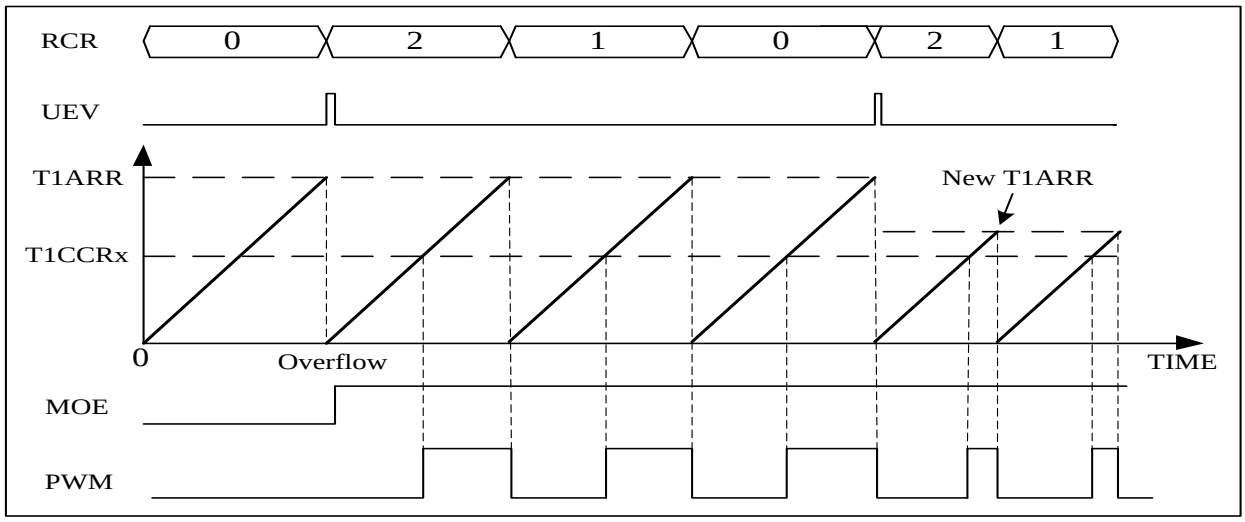


图 10.14 利用重复计数器输出 3 个特定的 PWM 的时序图

注意：

由于重复计数器只有在周期更新事件(UEV)发生时才重载 T1REP 值，对 TIM1_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用，所以建议当配置 T1REP 不为 0 时，在第一个更新事件(计数器上溢或下溢)之后再打开更新事件中断。

10.3.2. 计数控制器

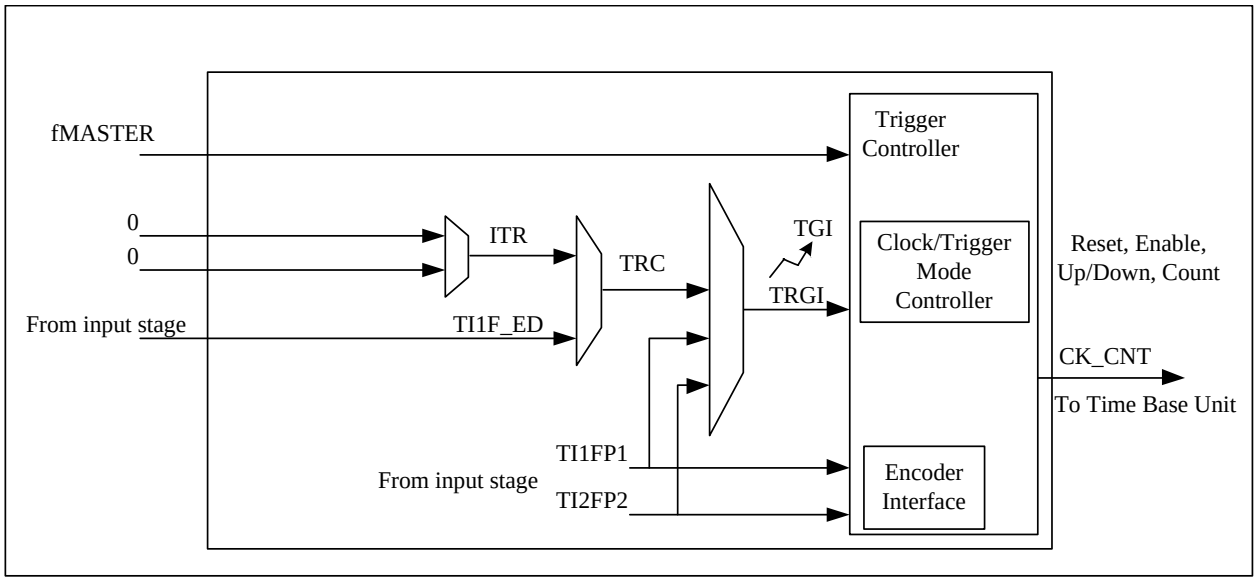


图 10.15 时钟/触发控制器框图

时钟/触发控制器允许配置各种计数器时钟源，输入触发和输出触发。

10.3.2.1. 计数器时钟源

计数器的计数时钟(CK_CNT)可由 TCKSRC 寄存器进行选择，总共有以下 8 种时钟源：

- 系统时钟/主时钟
- HIRC
- XT 时钟/外部时钟
- HIRC 的 2 倍频
- XT 时钟/外部时钟的 2 倍频
- LIRC
- LP 时钟/外部时钟
- LP 时钟/外部时钟的 2 倍频

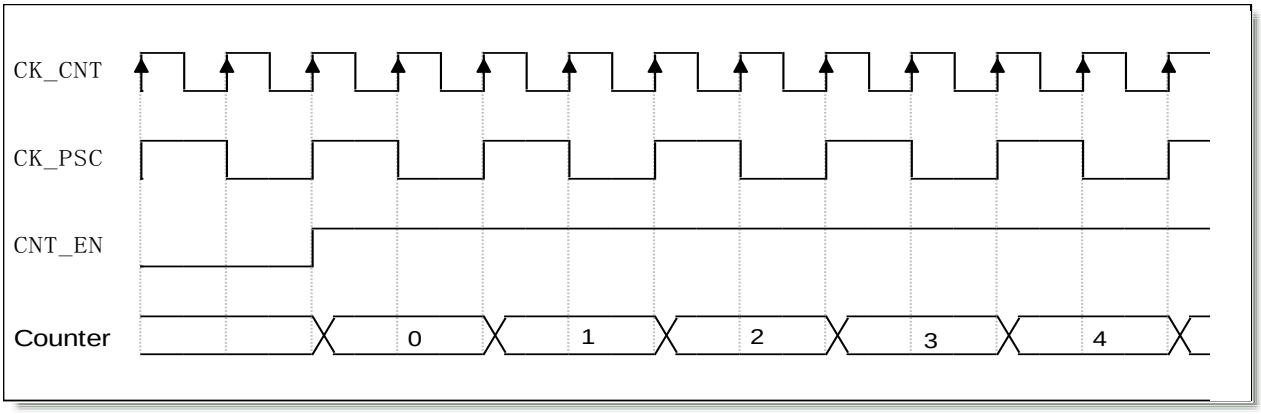


图 10.16 预分频为 1 时，计数器计数时序图

10.3.2.2. 计数触发源

本章节所述的触发源是计数触发源，而非捕捉源；计数触发源只能来自通道 1/2 对应的输入端口，来源与通道 1/2 的捕捉源的来源一致；计数触发源不能来自通道 3/4，通道 3/4 的输入只能作为对应的通道捕捉源。关于捕捉源的详细描述可查看 10.3.3.1 章节内容。

当T1SMS=000时，计数由内部时钟驱动，使能T1CEN即可触发计数，但是在slave模式(T1SMS=101/110)下，需要触发源，这些触发源由 TIM1CR1 寄存器中的 T1TS[2:0]位进行选择，总共有以下 3 种计数触发源：

（1） 输入源 T1 的边沿检测(TI1F_ED)；
（2） 滤波后的通道 1 输入(TI1FP1)；
（3） 滤波后的通道 2 输入(FI2FP2)；

注意：
只有当 T1SMS=101/110 时，计数触发源才有意义；否则，不管端口是否有触发信号输入都是没有意义的。

10.3.2.3. 计数控制模式选择

TIM1 除了有向上计数、向下计数、中央对齐计数方式之外，还有 4 种计数控制模式，需要计数触发源的配合使用；计数模式的选择由 TIM1SMCR 寄存器中的 T1SMS[2:0]去控制，下列是 4 种计数模式：

（1） 内部时钟模式：
计数由内部时钟(CK_CNT)驱动。
（2） 复位模式(必须在输入捕捉模式下才能配置为此模式)：
在选中的触发输入(TRGI)的上升沿时重新初始化计数器，并且产生一个更新寄存器的信号。
（3） 门控模式(T1SMS=3’b101)：
当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。
（4） 触发模式(T1SMS=3’b110)：
计数器在触发输入 TRGI 的上升沿启动(但不复位)，只有计数器的启动是受控的。

- 内部时钟模式：

内部时钟(CK_CNT)模式下，在软件配置计数器使能(T1CEN)之后，计数器开始由内部时钟(CK_CNT)驱动下进行计数；如图 10.16 所示。

- 复位模式：

当触发输入事件到来时，计数器和计数器预分频都会被初始化。如果此时 T1URS 为 0 且 T1UDIS 也为 0，则会产生一个更新事件，同时所有的预加载寄存器都会被更新。

复位模式的步骤示例：

1. 配置输入捕捉寄存器的值 – 配置输入捕捉滤波器 T1IC1F=000；配置捕捉预分频器 T1IC1PSC=0
2. 将通道配置为输入捕捉通道 T1CC1S=01，并将 IC1 映射在 TI1FP1 上
3. 写 T1CC1P=0，选择检测触发上升沿的到来
4. 通过写 T1SMS=100，将 TIM1 配置为复位模式。同时写 T1TS=101，选择 TI1 为输入触发源
5. 置位 T1CEN，启动计数器

以下是一段示例代码：

```
BANKSEL     PCKEN
BSR         PCKEN,0     ; 使能 TIM1 模块时钟
BANKSEL     TCKSRC
LDWI        H'01'
STR         TCKSRC      ; 选择 TIM1 时钟源为 HIRC
BANKSEL     TRISA
LDWI        H'FF'
STR         TRISA       ; PA0 为通道 1 的输入通道
BANKSEL     TIM1CCMR1
LDWI        H'01'
STR         TIM1CCMR1   ; 配置通道 1 的 IC1 映射在 TI1FP1 上
LDWI        H'54'
STR         TIM1SMCR    ; 配置 TIM1 为复位模式，触发源为 TI1FP1
LDWI        H'01'
STR         TIM1CCER1   ; 使能通道 1 并且为上升沿触发
BANKSEL     TIM1CR1
BSR         TIM1CR1,0   ; 开启计数器计数使能位
```

当 TI1 的上升沿到来时，计数器被清 0 并从 0 开始重新计数。与此同时与此同时，触发标志位(TIF)会被置位，在触发中断使能的情况下还会产生一个中断请求。如下列示例对应时序图所示：

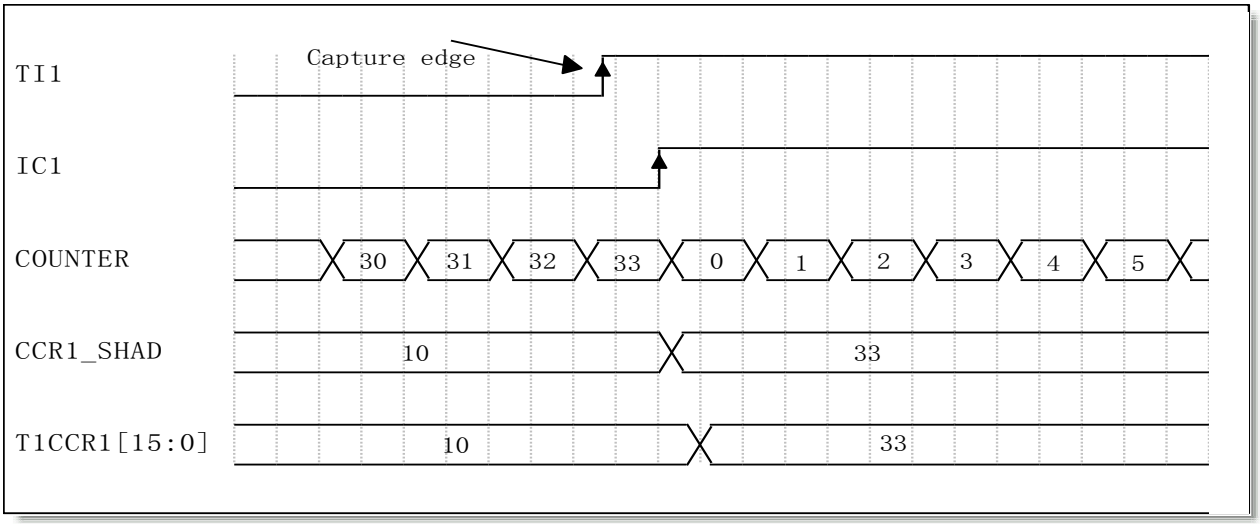


图 10.17 复位模式下，计数器计数时序图

- 门控模式：

依据选择的触发输入的电平值，计数器会被使能。此模式下，计数器的运行和停止都是受控的。

门控模式的步骤示例：

1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 配置输入捕捉寄存器的值 – 配置输入捕捉滤波器 T1IC1F=000；配置捕捉预分频器 T1IC1PSC=0
3. 将通道配置为输入捕捉通道 T1CC1S=01，并将 IC1 映射在 TI1FP1 上
4. 写 T1CC1P=1，选择检测输入低电平的到来
5. 通过写 T1SMS=101，将 TIM1 配置为门控模式。同时写 T1TS=101，选择 TI1 为输入源
6. 置位 T1CEN，使能计数器(在门控模式下，需要开启 T1CEN；在此基础上，才能由输入源控制计数器的运行与停止)

以下是一段示例代码：

```
BANKSEL     PCKEN
BSR         PCKEN,0     ; 使能 TIM1 模块时钟
BANKSEL     TCKSRC
LDWI        H'01'
STR         TCKSRC      ; 选择 TIM1 时钟源为 HIRC
BANKSEL     TRISA
LDWI        H'FF'
STR         TRISA       ; PA0 为通道 1 的输入通道
BANKSEL     TIM1CCMR1
LDWI        H'01'
STR         TIM1CCMR1   ; 配置通道 1 的 IC1 映射在 TI1FP1 上
LDWI        H'55'
STR         TIM1SMCR    ; 配置 TIM1 为门控模式，触发源为 TI1FP1
LDWI        H'03'
STR         TIM1CCER1   ; 使能通道 1 并且低电平输入为有效电平
BANKSEL     TIM1CR1
BSR         TIM1CR1,0   ; 开启计数器计数使能位
BTSS        TIM1SR1,6   ; 判断触发中断标志位是否为高
LJUMP       $-1
BCR         TIM1SR1,6   ; 将触发中断标志位清零
```

当 TI1 为低电平时，计数器在内部时钟的驱动下进行计数；当 TI1 变为高电平时，计数器停止计数。触发标志位(T1TIF)会在计数器启动或停止时被置位。如下列示例对应时序图所示：

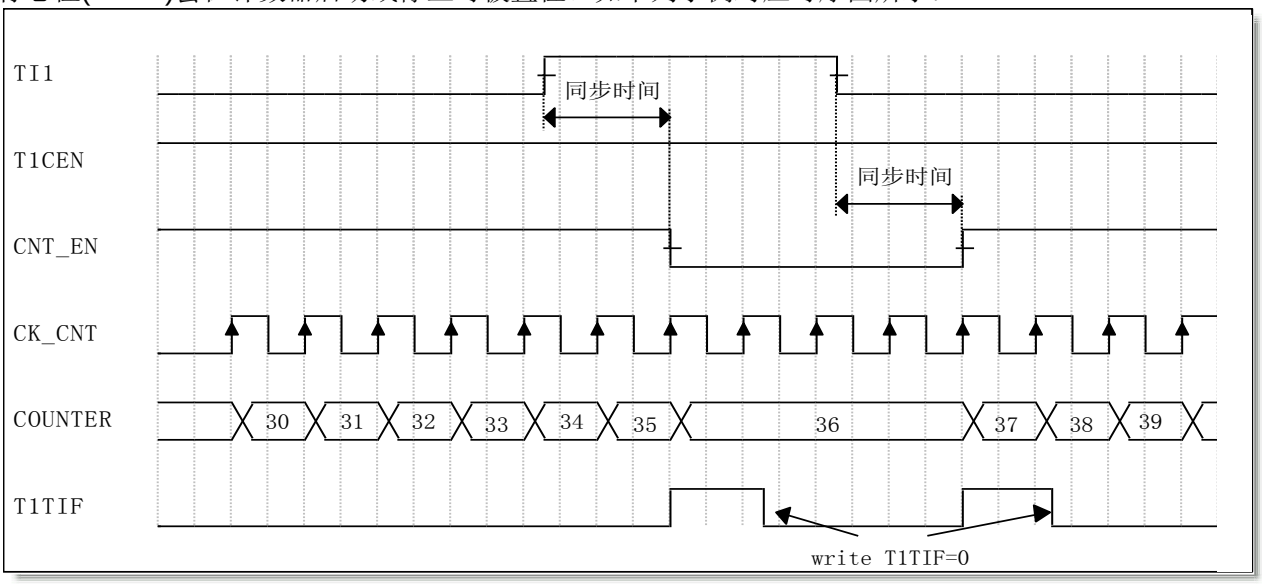


图 10.18 门控模式下，计数器计数时序图

- 触发模式：

依据选择的触发输入的电平值，计数器会被启动(T1CEN 被置位)。

触发模式的步骤示例：

1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 配置输入捕捉寄存器的值 – 配置输入捕捉滤波器 T1IC2F=000；配置捕捉预分频器 T1IC2PSC=0
3. 将通道配置为输入捕捉通道 T1CC2S=01，并将 IC2 映射在 TI2FP2 上
4. 写 T1CC2P=0，选择检测触发上升沿的到来
5. 通过写 T1SMS=110，将 TIM1 配置为触发模式。同时写 T1TS=110，选择 TI2 为输入触发源

以下是一段示例代码：

```
BANKSEL     PCKEN       ;
BSR         PCKEN,0     ; 使能 TIM1 模块时钟
BANKSEL     TCKSRC      ;
LDWI        H'01'       ;
STR         TCKSRC      ; 选择 TIM1 时钟源为 HIRC
BANKSEL     TRISA       ;
LDWI        H'FF'       ;
STR         TRISA       ; PA1 为通道 2 的输入通道
BANKSEL     TIM1CCMR2   ;
LDWI        H'01'       ;
STR         TIM1CCMR2   ; 配置通道 2 的 IC2 映射在 TI2FP2 上
LDWI        H'66'       ;
STR         TIM1SMCR    ; 配置 TIM1 为触发控制模式，触发源为 TI2FP2
LDWI        H'10'       ;
STR         TIM1CCER1   ; 使能通道 2 并且为上升沿触发
BANKSEL     TIM1CR1     ;
BSR         TIM1CR1,0   ; 开启计数器计数使能位
```

当 TI2 的上升沿到来时，计数器在内部时钟的驱动下启动计数，并且触发标志位(T1TIF)被置位。如下列示例对应时序图所示：

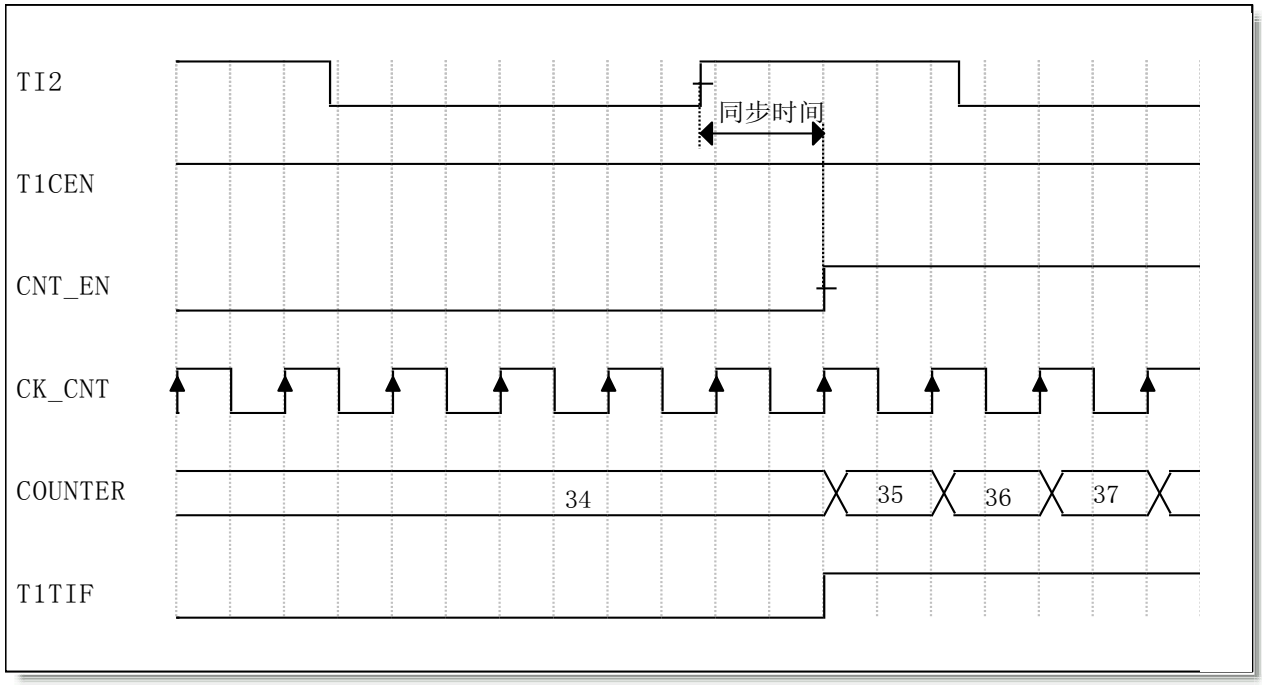


图 10.19 触发模式下，计数器计数时序图

10.3.3. 捕捉比较通道

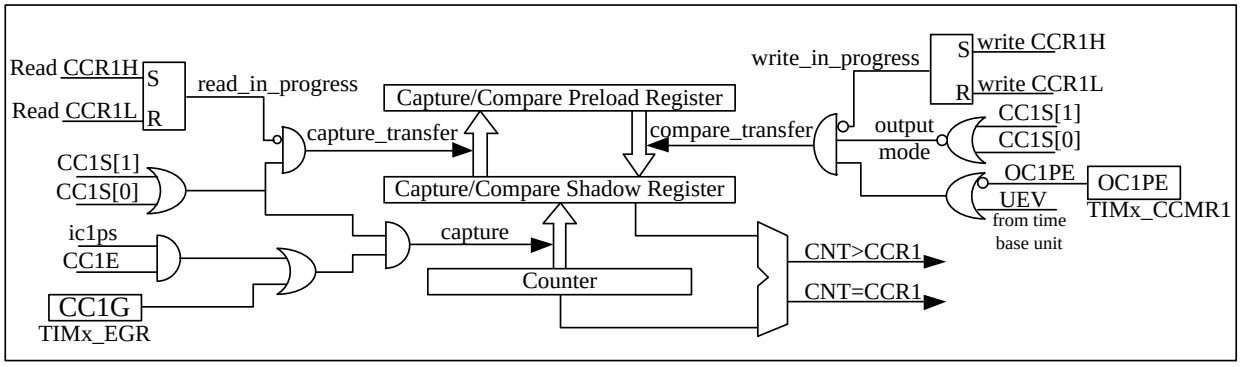


图 10.20 捕捉/比较通道 1 框图

TIMER 的 I/O 口能被配置为输入捕捉或输出比较功能。这个配置由 T1CCxS 通道选择位进行设定；对单个通道而言，输入捕捉功能和输出比较功能是互斥的两个功能。但每个通道都有独立的配置寄存器，所以可以将某些通道配置为输入捕捉功能，另一些通道配置为输出比较功能；例如：配置 T1CC1S=2'b00，T1CC2S=2'b00，T1CC3S=2'b01，T1CC3S=2'b10，这样通道 1 和通道 2 为输出比较通道，可在输出波形，而通道 3 和通道 4 为输入捕捉通道，可进行捕捉功能。

TIM1CCRxH/L 寄存器的读写：
TIM1CCRxH/L 寄存器由一个预加载寄存器和一个影子寄存器组成。
TIM1CCRxH/L 寄存器在输出比较模式和输入捕捉模式下的读写有所不同；在输出比较模式下，为可读可写寄存器；而在输入捕捉模式下，为只读寄存器。

- 在输出比较模式下：
  TIM1CCRxH/L 寄存器的访问没有任何限制，可读可写。
  读 TIM1CCRxH/L： 读出的值来自 CCRx 预加载寄存器的值，跟先前写入 TIM1CCRxH/L 寄存器的值保持一致。
  写 TIM1CCRxH/L：有预加载使能位(T1OCxPE)；如果预加载使能(T1OCxPE=0)关闭，则写入 TIM1CCRxH/L 寄存器的值直接由 CCRx 预加载寄存器传递到 CCRx 影子寄存器。反之，写入 TIM1CCRxH/L 寄存器的值在下一次更新事件发生时才会从 CCRx 预加载寄存器传递到 CCRx 影子寄存器。
- 在输入捕捉模式下：
  TIM1CCRxH/L 寄存器为只读寄存器。在捕捉事件发生时，计数器值会被写入到 CCRx 影子寄存器中，而后再写回到 CCRx 预加载寄存器中。
  读 TIM1CCRxH/L 寄存器时，必须先读高 8 位，再读低 8 位。读高 8 位时，CCRx 预加载寄存器被冻结，此时计数器值无法写回到 CCRx 预加载寄存器中；只有按顺序读完低 8 位后，CCRx 预加载寄存器才能更新为最新一次捕捉值。

注意：
TIM1CCMRx 寄存器是复用寄存器。
当作为输出比较通道时，TIM1CCMRx 寄存器作为输出配置寄存器，并且第 7 位和第 2 位禁止配置，保持为默认值；下表为 TIM1CCMRx 作为输出配置寄存器时的具体意义：

Bit	7	6	5	4	3	2	1	0
Name	reserved	T1OCxM[2:0]			T1OCxPE	reserved	T1CCxS[1:0]	
Rese	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RW	RW	RO-0	RW	RW

表 10.2 TIM1CCMRx 作为输出配置寄存器

当作为输入捕捉通道时，TIM1CCMRx 寄存器作为输入配置寄存器；下表为 TIM1CCMRx 作为输出配置寄存器时的具体意义：

Bit	7	6	5	4	3	2	1	0
Name	T1ICxF[3:0]				T1ICxPSC[1:0]		T1CCxS[1:0]	
Rese	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

表 10.3 TIM1CCMRx 作为输入配置寄存器

10.3.3.1. 捕捉输入通道

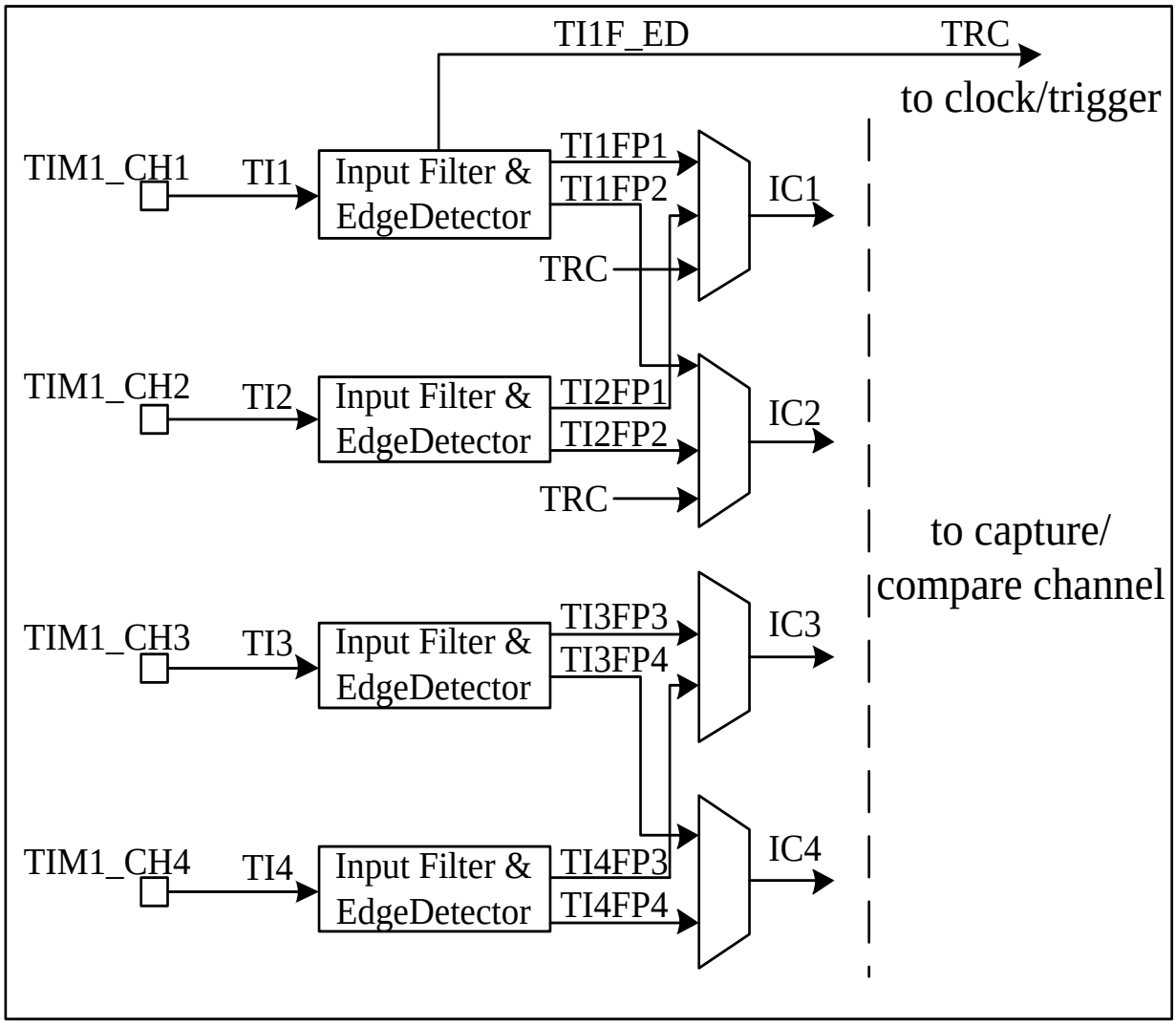


图 10.21 通道输入框图

信号名称	详细说明
TIM1_CH1/2/3/4	通道 1/2/3/4 对应 I/O 口的输入
IC1/2/3/4	通过选择后的真正的通道捕捉源
TI1FP1	来自通道 1 对应 I/O 的输入捕捉信号，作为通道 1 的捕捉源之一
TI1FP2	来自通道 1 对应 I/O 的输入捕捉信号，作为通道 2 的捕捉源之一
TI2FP2	来自通道 2 对应 I/O 的输入捕捉信号，作为通道 2 的捕捉源之一
TI2FP1	来自通道 2 对应 I/O 的输入捕捉信号，作为通道 1 的捕捉源之一
TI3FP3	来自通道 3 对应 I/O 的输入捕捉信号，作为通道 3 的捕捉源之一
TI3FP4	来自通道 3 对应 I/O 的输入捕捉信号，作为通道 4 的捕捉源之一
TI4FP4	来自通道 4 对应 I/O 的输入捕捉信号，作为通道 4 的捕捉源之一
TI4FP3	来自通道 4 对应 I/O 的输入捕捉信号，作为通道 3 的捕捉源之一
TRC	来自通道 1 对应 I/O 的输入双沿捕捉信号，作为通道 1 和通道 2 的捕捉源之一

表 10.4 对应图 10.9 中信号说明列表

当一个通道被配置成输入捕捉通道并且输入捕捉事件有效时，可以将当前的计数值保存在 TIM1CCRx 寄存器。每个通道都有一个数字滤波单元，可配置采样频率(T1ICxF[3:0])，捕捉预分频(T1IC1PSC[1:0])，捕捉极性选择(T1CCxP)和捕捉触发源(T1CCxS)。每个通道都有各自的捕捉源，如下表所示：

T1CCxS (捕捉源选择)	**通道 1**	**通道 2**	**通道 3**	**通道 4**
2'b00	TI1FP1	TI2FP2	TI3FP3	TI4FP4
2'b01	TI2FP1	TI1FP2	TI4FP3	TI3FP4
2'b10	TRC	TRC	—	—

表 10.5 各通道输入捕捉源列表

当一个输入捕捉发生时：

- TIM1CCR1H/L 寄存器得到捕捉发生时计数器的值。
- 输入捕捉标志位(T1CCxIF)被置位。如果当 T1CCxIF 保持为 1 时，又一次发生了输入捕捉事件，那么溢出捕捉标志位(T1CCxOF)也会被置位。
- 如果 T1CCxIE 为 1，那么捕捉将产生一个中断事件。

配置为输入捕捉通道的示例步骤：

1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 将通道相应的端口配置为输入端口
3. 选择输入触发源(T1CCxS)
4. 配置采样频率(T1ICxF[3:0])，捕捉预分频(T1IC1PSC[1:0])
5. 配置捕捉源的捕捉极性(T1CCxP)
6. 使能捕捉通道(T1CCxE)
7. 使能计数器(T1CEN)

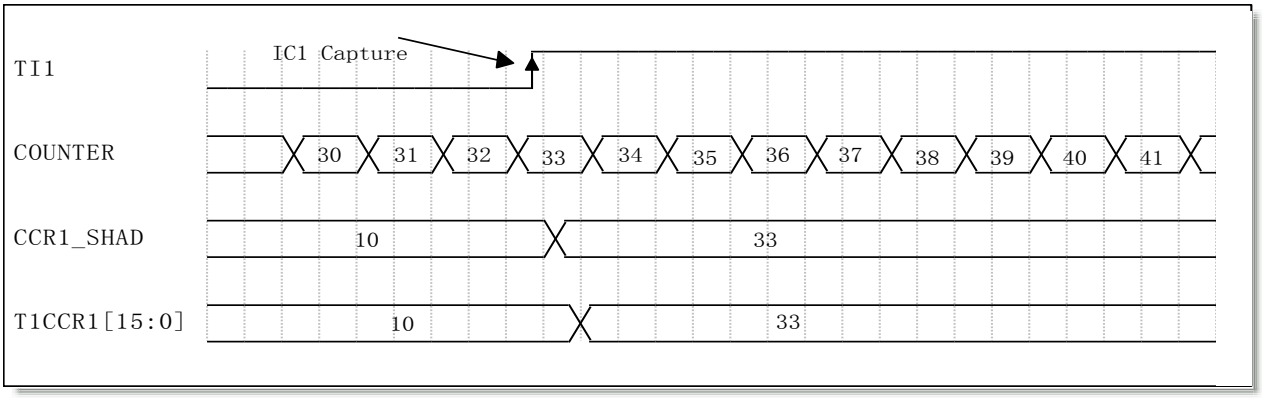


图 10.22 简单输入捕捉时序图

PWM 输入信号测量的应用：

利用捕捉输入模式和复位模式，并且将两个通道的输入捕捉源都选择为同一个通道的 PWM 信号输入；这样就可以测量从通道输入的 PWM 信号的周期以及占空比。

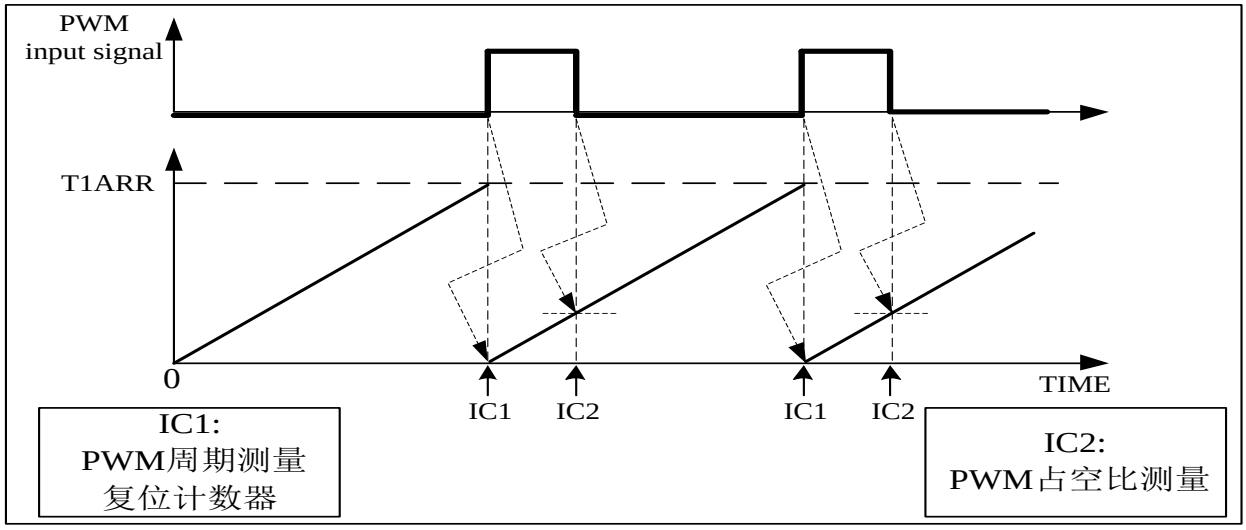


图 10.23 测量 PWM 信号的示意图

具体测量 PWM 的配置步骤如下：

1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 将通道 1/2 相应的端口配置为输入端口
3. 通道 1 配置将 IC1 映射在 TI1FP1 上；通道 2 配置将 IC2 映射在 TI2FP1 上
4. 配置通道 1 为上升沿捕捉(T1CC1P=0)；通道 2 为下降沿捕捉(T1CC2P=1)
5. 配置采样频率(T1ICxF[3:0]=4'b0000)，捕捉预分频(T1IC1PSC[1:0]=2'b00)
6. 将计数控制模式配置为复位模式(T1SMS=101)，计数触发源配置为 TI1FP1(T1TS=101)
7. 使能计数器(T1CEN)
8. 开启通道 1 和通道 2 的输入捕捉功能(T1CC1E=1 且 T1CC2E=1)

注意：

因为捕捉沿先于复位触发源两个计数时钟周期，所以需要软件进行一下操作才能等到准确测量值：

- 当预分频为 0 时，PWM 周期等于 T1CCR1H/L+2，PWM 占空比等于 T1CCR2H/L+2
- 当预分频为 1 时，PWM 周期等于 T1CCR1H/L+1，PWM 占空比等于 T1CCR2H/L+1

- 当预分频大于 1 时，PWM 周期等于 T1CCR1H/L，PWM 占空比等于 T1CCR2H/

以下是一段示例代码：

```
BANKSEL     PCKEN           ;
BSR         PCKEN,0         ; 使能 TIM1 模块时钟
BANKSEL     TCKSRC          ;
LDWI        H'01'           ;
STR         TCKSRC          ; 选择 TIM1 时钟源为 HIRC
BANKSEL     TRISA           ;
LDWI        H'FF'           ;
STR         TRISA           ; 配置 PA0 为通道 1 的输入通道，PA1 为通道 2 的输入通道
BANKSEL     TIM1CCMR1       ;
LDWI        H'01'           ;
STR         TIM1CCMR1       ; 配置通道 1 为 IC1 映射在 TI1FP1 上
LDWI        H'02'           ;
STR         TIM1CCMR2       ; 配置通道 2 的 IC2 映射在 TI1FP2 上
LDWI        H'54'           ;
STR         TIM1SMCR        ; 配置 TIM1 为复位控制模式，触发源为 TI1FP1
LDWI        H'31'           ;
STR         TIM1CCER1       ; 使能通道 1/2，通道 1 为上升沿捕捉，通道 2 为下降沿捕捉
BANKSEL     TIM1CR1         ;
BSR         TIM1CR1,0       ; 开启计数器计数使能位
```

上述示例代码对应波形图：

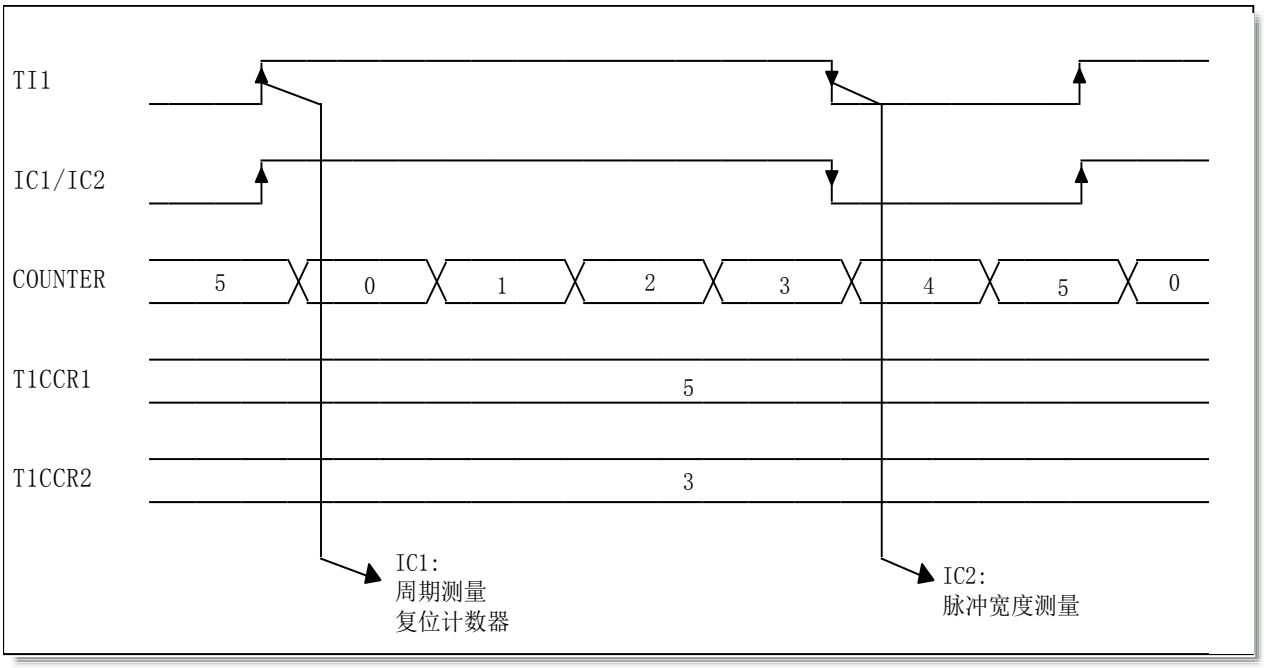


图 10.24 测量 PWM 信号的时序图

10.3.3.2. 输出比较通道

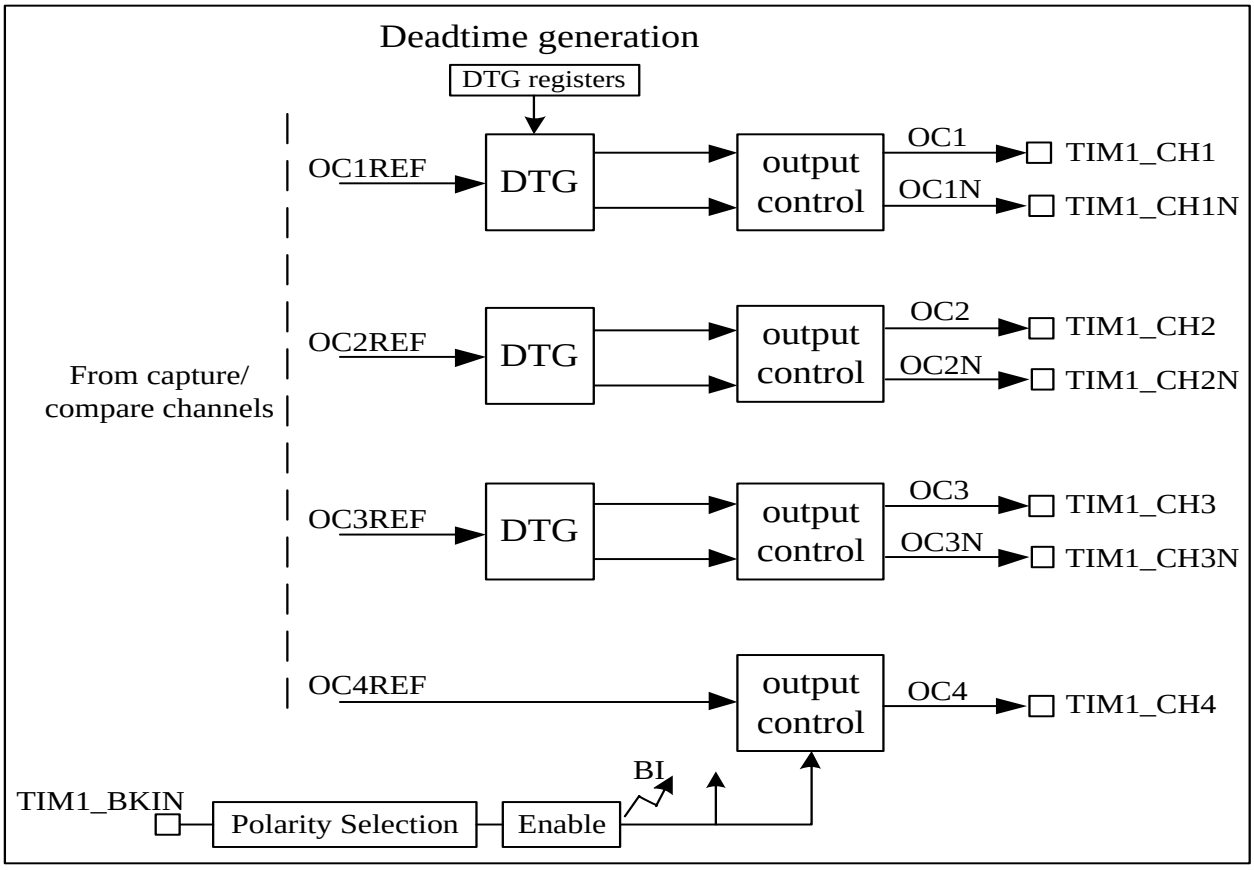


图 10.25 通道输出框图

输出阶段产生立即响应的波形，用来作为参考波形，叫做 OCxREF 信号(高有效)。刹车功能，极性选择和其他输出控制位都在参考波形之后去做控制。
输出比较通道根据计数值与比较值 CCRx，产生 OCxREF 输出并送到死区产生模块，经过死区产生模块后再经过其他输出控制位的控制将波形输出到端口。
具体的输出控制位以及可达到的输出效果可以查看 10.3.3.5 章节内容。

输出比较模式下，可选择不同的输出模式去输出 PWM 波形；输出模式由 T1OCxM[3:0]选择，总共有以下 8 种不同的输出模式(最终的输出还需要取决于极性选择(T1CCxP))：

（1） 冻结模式：输出冻结，输出实际比较值(CCRx_SHAD)与计数器 TIM1_CNT 间的比较对 OC1REF 不起作用
（2） 匹配有效：当计数值 CNT 与实际比较值(CCRx_SHAD)匹配时，OCxREF 为高电平；
（3） 匹配无效：当计数值 CNT 与实际比较值(CCRx_SHAD)匹配时，OCxREF 为低电平；
（4） 翻转：当计数值 CNT 与实际比较值(CCRx_SHAD)匹配时，输出翻转；
（5） 强制无效：OCxREF 强制为低电平；
（6） 强制有效：OCxREF 强制为高电平；
（7） PWM1：向上计数时，当 CNT<实际比较值(CCRx_SHAD)时，OCxREF 有效；
向下计数时，CNT>实际比较值(CCRx_SHAD)时，OCxREF 无效；
（8） PWM2：向上计数时，当 CNT<实际比较值(CCRx_SHAD)时，OCxREF 无效；
向下计数时，CNT>实际比较值(CCRx_SHAD)时，OCxREF 有效；

配置为输出比较通道的示例步骤：

1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 将通道相应的端口配置为输出端口
3. 配置输出波形的周期(T1ARR)和占空比(T1CCRx)
4. 配置输出比较模式(T1OCxM)和输出极性(T1CCxP)
5. 使能比较输出通道(T1CCxE)
6. 打开主输出自动使能位(T1AOE)，在更新事件发生时硬件会自动使能主输出(T1MOE)
7. 使能计数器(T1CEN)

以下是一段示例代码：

```
BANKSEL   PCKEN
BSR       PCKEN,0     ; 使能 TIM1 模块时钟
BANKSEL   TCKSRC
LDWI      H'01'
STR       TCKSRC      ; 选择 TIM1 时钟源为 HIRC
BANKSEL   TRISA
LDWI      H'FE'
STR       TRISA       ; 配置 PA0 为通道 1 的输出通道
BANKSEL   TIM1ARRL
LDWI      H'05'
STR       TIM1ARRL    ; 将输出波形周期配置为 6
LDWI      H'03'
STR       TIM1CCR1L   ; 将输出波形占空比配置为 3
BANKSEL   TIM1CCMR1
LDWI      H'10'
STR       TIM1CCMR1   ; 配置通道 1 为匹配有效模式输出
LDWI      H'01'
STR       TIM1CCER1   ; 使能通道 1
BANKSEL   TIM1BKR
BSR       TIM1BKR,6   ; 打开主输出自动使能位 T1AOE
BANKSEL   TIM1CR1
BSR       TIM1CR1,0   ; 开启计数器计数使能位
```

上述示例代码对应波形图：

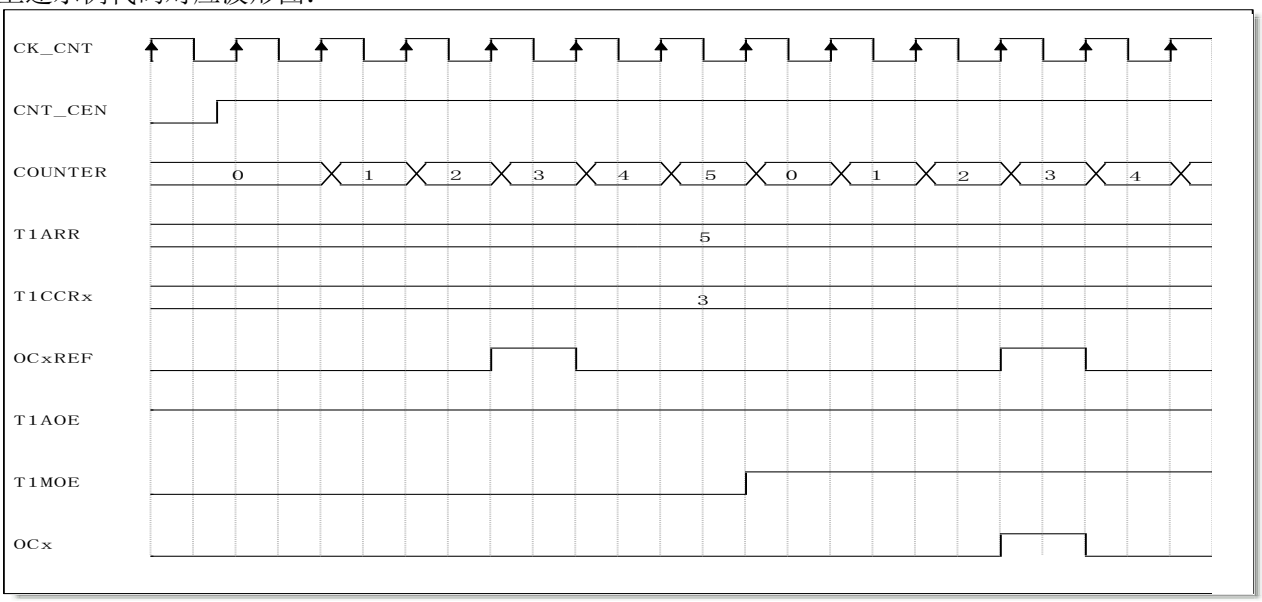


图 10.26 T1OCxM 为匹配有效模式下的输出时序图

以下是一段示例代码：

```
BANKSEL     PCKEN           ;
BSR         PCKEN,0         ; 使能 TIM1 模块时钟
BANKSEL     TCKSRC          ;
LDWI        H'01'           ;
STR         TCKSRC          ; 选择 TIM1 时钟源为 HIRC
BANKSEL     TRISA           ;
LDWI        H'FE'           ;
STR         TRISA           ; 配置 PA0 为通道 1 的输出通道
BANKSEL     TIM1ARRL        ;
LDWI        H'05'           ;
STR         TIM1ARRL        ; 将输出波形周期配置为 6
LDWI        H'03'           ;
STR         TIM1CCR1L       ; 将输出波形占空比配置为 3
BANKSEL     TIM1CCMR1       ;
LDWI        H'30'           ;
STR         TIM1CCMR1       ; 配置通道 1 为翻转模式输出
LDWI        H'01'           ;
STR         TIM1CCER1       ; 使能通道 1
BANKSEL     TIM1BKR         ;
BSR         TIM1BKR,6       ; 打开主输出自动使能位 T1AOE
BANKSEL     TIM1CR1         ;
BSR         TIM1CR1,0       ; 开启计数器计数使能位
```

上述示例代码对应波形图：

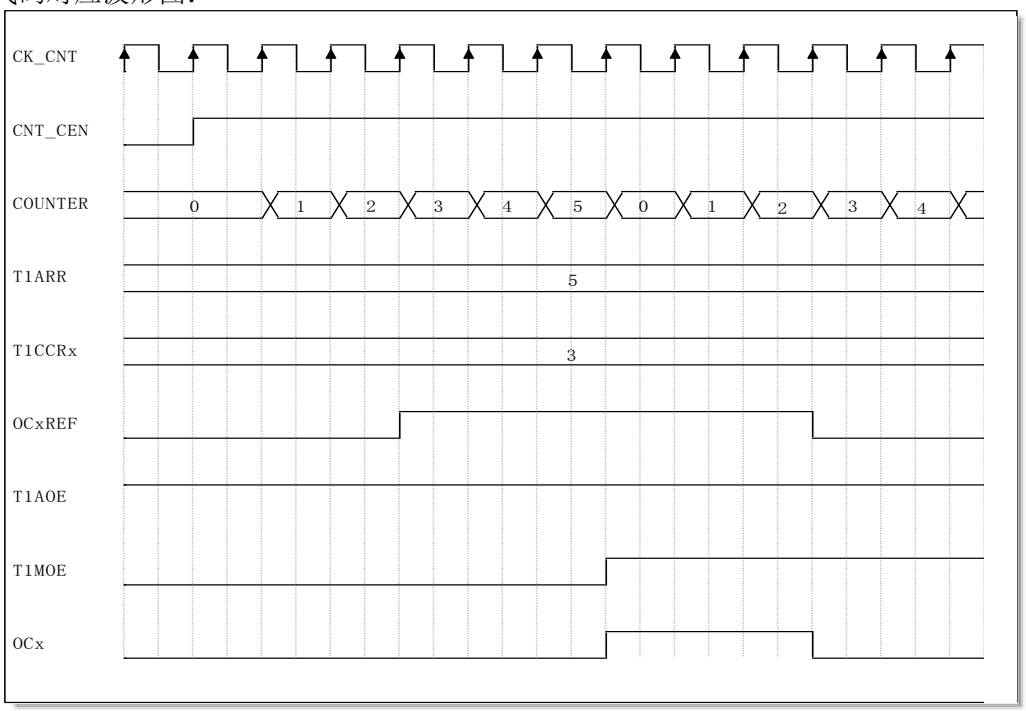


图 10.27 T1OCxM 为翻转模式下的输出时序图

以下是一段示例代码：

```
BANKSEL   PCKEN
BSR       PCKEN,0     ; 使能 TIM1 模块时钟
BANKSEL   TCKSRC
LDWI      H'01'
STR       TCKSRC      ; 选择 TIM1 时钟源为 HIRC
BANKSEL   TRISA
LDWI      H'FE'
STR       TRISA       ; 配置 PA0 为通道 1 的输出通道
BANKSEL   TIM1ARRL
LDWI      H'05'
STR       TIM1ARRL    ; 将输出波形周期配置为 6
LDWI      H'03'
STR       TIM1CCR1L   ; 将输出波形占空比配置为 3
BANKSEL   TIM1CCMR1
LDWI      H'70'
STR       TIM1CCMR1   ; 配置通道 1 为 PWM2 模式输出
LDWI      H'01'
STR       TIM1CCER1   ; 使能通道 1
BANKSEL   TIM1BKR
BSR       TIM1BKR,6   ; 打开主输出自动使能位 T1AOE
BANKSEL   TIM1CR1
BSR       TIM1CR1,0   ; 开启计数器计数使能位
```

上述示例代码对应波形图：

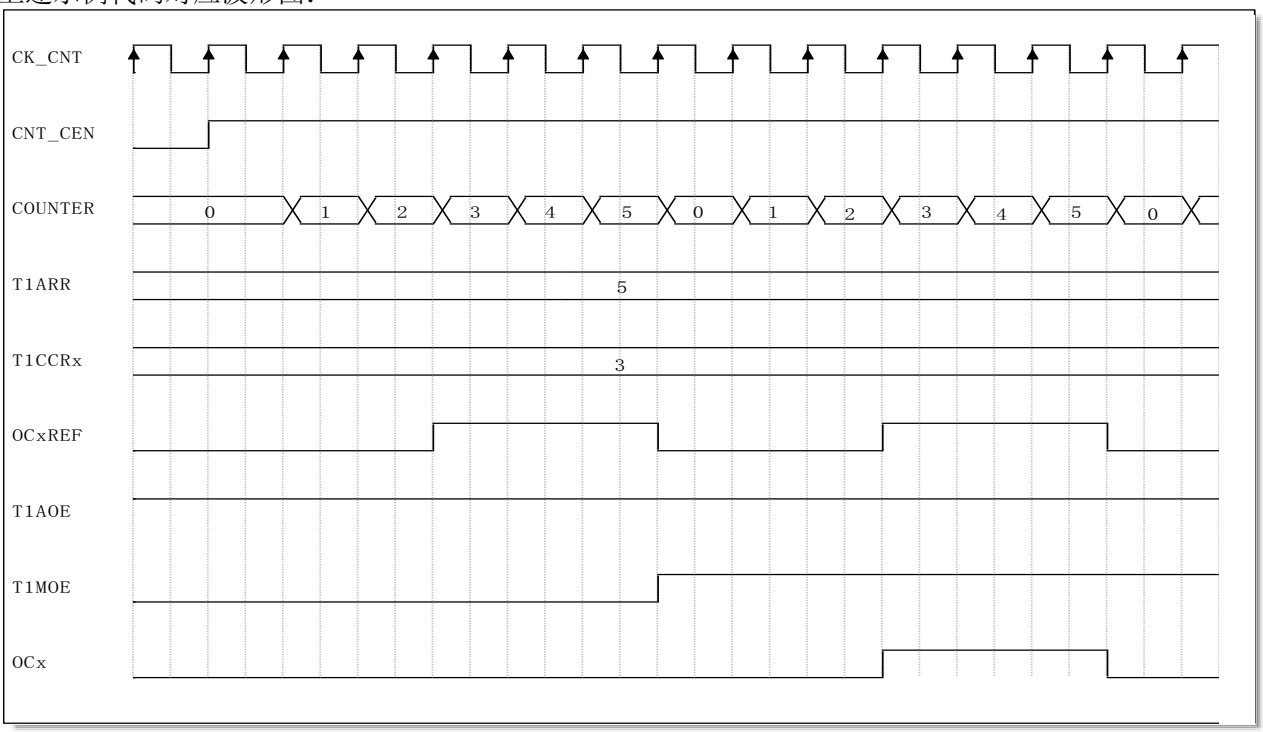


图 10.28 T1OCxM 为 PWM2 模式下的输出时序图

注意：

- 刹车事件的到来只会影响输出，并不影响 TIMER 本身的计数。
- 为了避免在计数器还未开启时，就开始输出脉冲；需要在计数器使能之后再开启 T1CCxE 和 T1CCxNE

10.3.3.3. 单次脉冲模式

开启单次脉冲模式和不开单次脉冲模式的区别在于：开启单次脉冲模式(T1OPM=1)并且下一次更新事件到来时，硬件会自动关闭计数器使能位(T1CEN)，计数器停止计数。

想要产生一个正确的脉冲，比较值(T1CCRx)必须与计数器初始值(T1ARR)不同；所以在开始计数之前必须满足以下配置：

- 在向上计数模式下：COUNTER < T1CCRx < T1ARR
- 在向下计数模式下：COUNTER > T1CCRx

想要产生一个正确的脉冲，还必须满足以下配置：

- 在输出模式为 PWM1 模式(T1OCxM=110)下：T1CCxP 必须为 1
  如果输出模式为 PWM1 模式并且 T1CCxP 为 0，更新事件之后 PWM 输出会一直为有效值
- 在输出模式为 PWM2 模式(T1OCxM=111)下：T1CCxP 必须为 0
  如果输出模式为 PWM2 模式并且 T1CCxP 为 1，更新事件之后 PWM 输出会一直为有效值

单次脉冲模式可以配合触发模式在特定的时间点产生单个特定的 PWM 输出；配置步骤如下所示：

1. 使能 TIM1 模块时钟并选择 TIM1 时钟源
2. 将通道 2 相应的端口配置为输入端口，通道 1 相应的端口配置为输出端口
3. 通道 2 配置 T1CC2S 为 01，IC2 映射在 TI2FP2 上；并配置通道 2 为上升沿捕捉(T1CC2P=0)
4. 将计数控制模式配置为触发模式(T1SMS=110)，计数触发源配置为 TI2FP2(T1TS=110)
5. 通道 1 配置为输出通道(T1CC1S=00)
6. 通道 1 的比较输出模式配置为 PWM2 模式(T1OC1M=111)，输出极性配置为高电平有效(T1CC1P=0)
7. **打开主输出使能(T1MOE)并且**使能计数器(T1CEN)
8. 开启通道 2 的输入捕捉功能 T1CC2E=1)和通道 1 的输出比较功能(T1CC1E)

以下是一段示例代码：

```
BANKSEL     PCKEN       ;
BSR         PCKEN,0     ; 使能 TIM1 模块时钟
BANKSEL     TCKSRC      ;
LDWI        H'01'       ;
STR         TCKSRC      ; 选择 TIM1 时钟源为 HIRC
BANKSEL     TRISA       ;
LDWI        H'FE'       ;
STR         TRISA       ; 配置 PA0 为通道 1 的输出通道，PA1 为通道 2 的输入通道
BANKSEL     TIM1CCMR1   ;
LDWI        H'70'       ;
STR         TIM1CCMR1   ; 配置通道 1 为 PWM2 模式输出
LDWI        H'01'       ;
STR         TIM1CCMR2   ; 配置通道 2 的 IC2 映射在 TI2FP2 上
LDWI        H'66'       ;
STR         TIM1SMCR    ; 配置 TIM1 为触发控制模式，触发源为 TI2FP2
LDWI        H'11'       ;
STR         TIM1CCER1   ; 使能通道 1 和通道 2
BANKSEL     TIM1BKR     ;
BSR         TIM1BKR,7   ; 打开主输出使能 T1MOE
BANKSEL     TIM1CR1     ;
BSR         TIM1CR1,0   ; 开启计数器计数使能位
```

上述示例代码对应示意图：

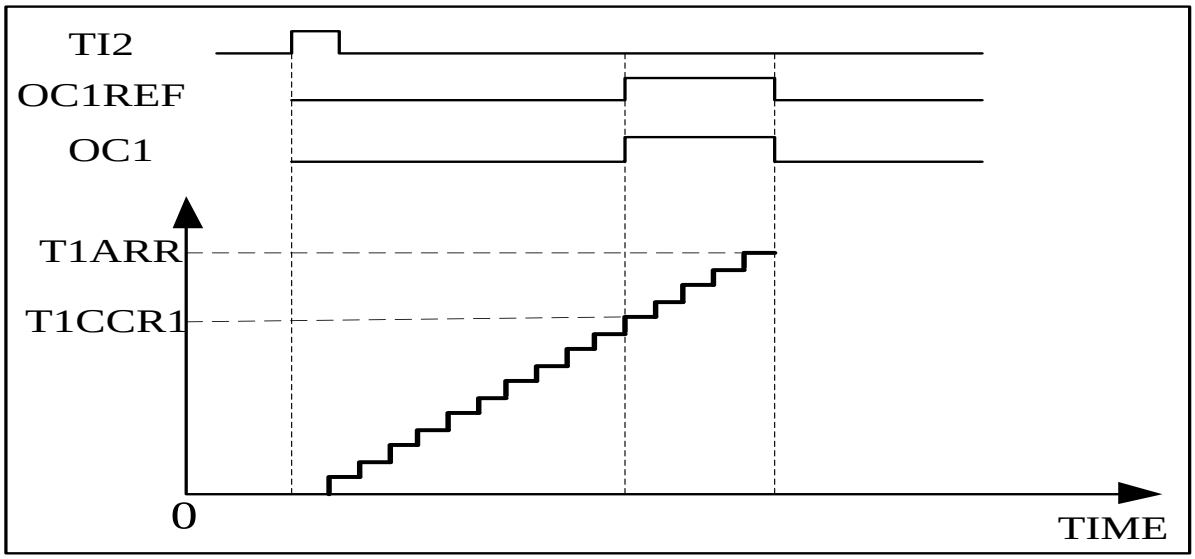


图 10.29 单次脉冲应用示意图

10.3.3.4. 死区产生

当把通道的互补输出使能时，就自动使能死区功能。每当一个输出信号(正向输出信号或互补输出信号)出现下降沿时，就会将另一个信号的上升沿后延一个死区时间长度。如图 10.26 和图 10.27 所示：

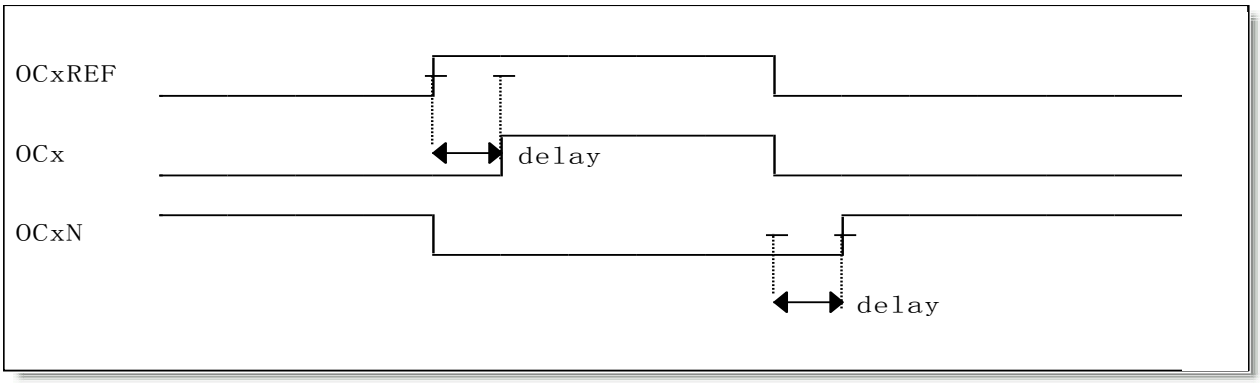


图 10.30 正向输出插入死区时序图

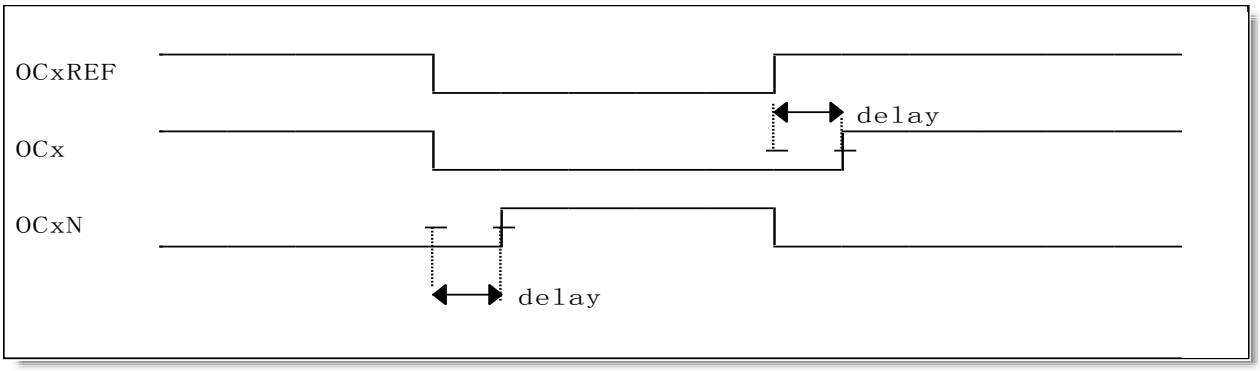


图 10.31 互补输出插入死区时序图

死区时间可以编程:根据寄存器位 T1DTG[7:0]，可以配置死区时间长度，具体参考寄存器描述 TIM1DTR 的 T1DTG[7:0]。

有的 OCXREF 输出的脉冲时间很短（小于死区时间），有可能某一脉冲信号(正向输出信号或反向输出信号)会被死区覆盖，导致输出不变化。如图 10.28 和图 10.29 所示：

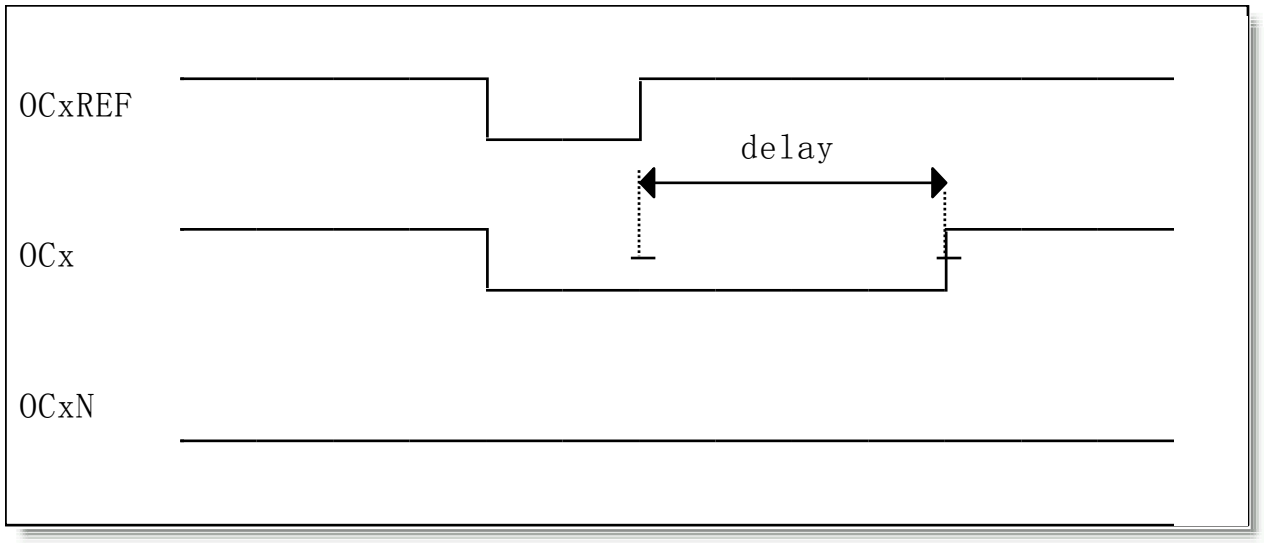


图 10.32 正向输出被死区覆盖时序图

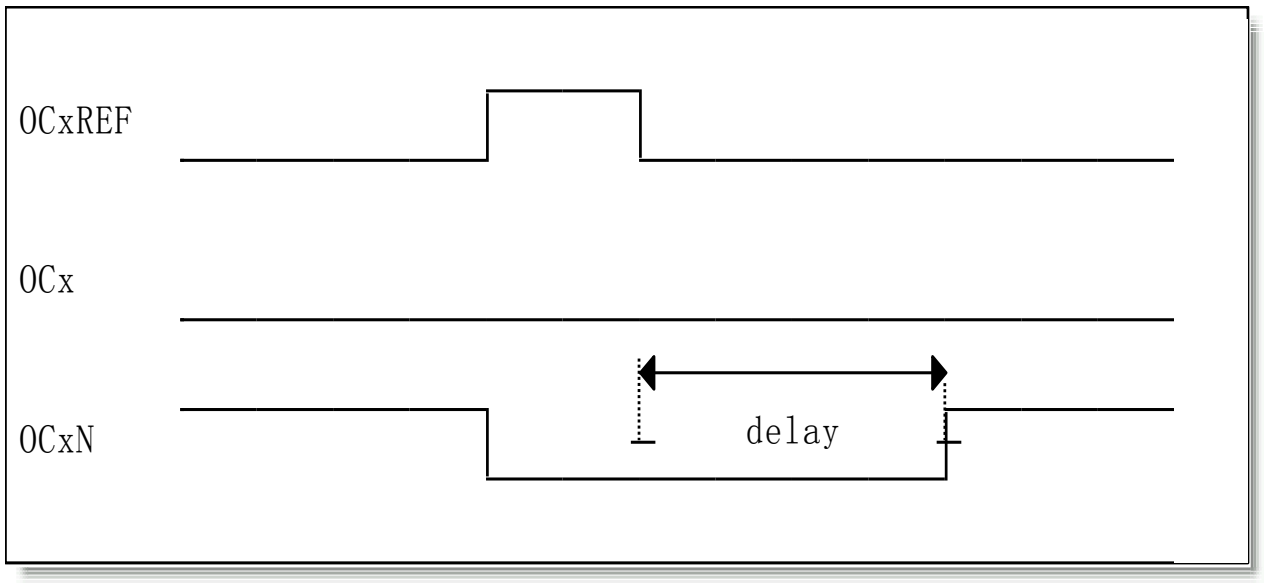


图 10.33 互补输出被死区覆盖时序图

10.3.3.5. 输出控制

参考波形 OCxREF 产生后不是直接输出到端口，会先经过死区控制模块和通过极性选择，最后再由以下 5 位控制信号进行组合控制之后才送到端口上。具体的组合控制内容如下表所示：

控制位					输出状态	
T1MOE	**T1OSSI**	**T1OSSR**	**T1CCxE**	**T1CCxNE**	**OCx 输出状态**	**OCxN 输出状态**
1	X	0	0	0	输出关闭(不由 TIMER 驱动) OCx=0, OCx_EN=0	输出关闭(不由 TIMER 驱动) OCxN=0, OCxN_EN=0
		0	0	1	输出关闭(不由 TIMER 驱动) OCx=0, OCx_EN=0	OCxREF + 极性选择 OCxN=OCxREF ^ T1CCxNP OCxN_EN=1
		0	1	0	OCxREF + 极性选择 OCx=OCxREF ^ T1CCxNP OCx_EN=1	输出关闭(不由 TIMER 驱动) OCxN=0, OCxN_EN=0
		0	1	1	OCxREF + 极性选择 + 死区时间 OCx_EN=1	OCREF 的互补信号 + 极性选择 + 死区时间 OCxN_EN=1
		1	0	0	输出关闭(不由 TIMER 驱动) OCx=T1CCxP, OCx_EN=0	输出关闭(不由 TIMER 驱动) OCxN=T1CCxNP,OCxN_EN=0
		1	0	1	关闭状态(运行模式下输出使能) OCx=T1CCxP, OCx_EN=1	OCxREF + 极性选择 OCxN=OCxREF ^ T1CCxNP OCxN_EN=1
		1	1	0	OCxREF + 极性选择 OCx=OCxREF ^ T1CCxNP OCx_EN=1	关闭状态(运行模式下输出使能) OCxN=T1CCxNP, OCx_EN=1
		1	1	1	OCxREF + 极性选择 + 死区时间 OCx_EN=1	OCREF 的互补信号 + 极性选择 + 死区时间 OCxN_EN=1
0	0	X	0	0	输出关闭(不由 TIMER 驱动) OCx=T1CCxP, OCx_EN=0	输出关闭(不由 TIMER 驱动) OCxN=T1CCxNP,OCxN_EN=0
	0		0	1	输出关闭(不由 TIMER 驱动) 一开始 OCx=T1CCxP,OCx_EN=0,OCxN=T1CCxNP,OCxN_EN=0 在死区时间之后 OCx=T1OISx,OCxN=T1OISxN	
	0		1	0		
	0		1	1		
	1		0	0	输出关闭(不由 TIMER 驱动) OCx=T1CCxP, OCx_EN=0	输出关闭(不由 TIMER 驱动) OCxN=T1CCxNP,OCxN_EN=0
	1		0	1	关闭状态(空闲模式下输出使能) 一开始 OCx=T1CCxP,OCx_EN=1,OCxN=T1CCxNP,OCxN_EN=1 在死区时间之后 OCx=T1OISx,OCxN=T1OISxN	
	1		1	0		
	1		1	1		

表 10.6 输出控制与输出状态

可以根据 T1MOE、T1OSSI、T1OSSR、T1CCxNE 和 CCxE 进行输出的控制。

注意：
输出的状态转换实际是刹车事件异步将 T1MOE 清零实现的，T1MOE 异步清零，但要注意刹车事件撤销时需要同步 2 个 CK_CNT 时钟（如果有时钟的情况下）。

10.3.4. TIM1 中断

TIM1 有以下 7 个中断请求源：
- 刹车中断
- 触发中断
- 捕捉/比较 4 中断
- 捕捉/比较 3 中断
- 捕捉/比较 2 中断
- 捕捉/比较 1 中断
- 更新中断(例如：上溢、下溢、计数初始化)

在用这些中断之前需要提前打开 TIM1IER 寄存器中的中断使能位(T1BIE、T1TIE、T1CCxIE 和 T1UIE)。

不同的中断源还可以配置通过 TIM1EGR 寄存器来产生(软件产生中断)。

10.3.5. 故障刹车源

TIM1 有以下 3 种刹车事件：
- BKIN 管脚事件
- LVD 事件
- ADC 比较事件

当故障事件有效且被选择为刹车源（由 BKS0~2 决定），如果 BKE 位为 1，PWM 输出管脚将被置于预设的状态，预设状态由寄存器 TIM1OISR 决定。

当一个刹车事件发生时：
- T1MOE 位会被异步清 0，强制输出进入无效状态，空闲状态或复位状态。甚至在 MCU 振荡器关闭的情况下，T1MOE 也会被刹车事件清 0。
- 在 T1MOE=0 之后，每个输出通道都会先将输出值置为无效值，等死区时间到之后变成提前设置好的 T1OISx 位的值。如果 T1OSSI=0，TIMER 会将输出关闭。
- 当互补输出使能时：
  输出首先会设置为无效值(根据极性选择位)。此操作是异步清 0 的，所以即使 TIM1 没有时钟驱动也能进行。
  如果 TIM1 是有时钟进行驱动的，那么死区时间到来之后就会进入由 T1OISx 和 T1OISxN 提前设定的预设状态。(由于 T1MOE 的同步，所以此情况下真正的死区时间会比死区设置值长 2 个 CK_CNT 时钟)
- 刹车状态标志位(T1BIF)被置位。如果 T1BIE 位为 1，那么将会产生一个中断事件。
- 如果 T1AOE 位配置为 1，那么 T1MOE 位在下次更新事件(UEV)到来时，将会由硬件自动置位。如果 T1AOE 位为 0，那么只能由软件将 T1MOE 位重新置位。

当故障事件有效时，T1MOE 清 0，PWM 输出将一直置于预设状态；
故障事件撤消后，如果 T1AOE=1，PWM 将在下一次更新事件后恢复正常输出，否则，软件需要自动打开 T1MOE。如图 10.30 和图 10.31 所示。

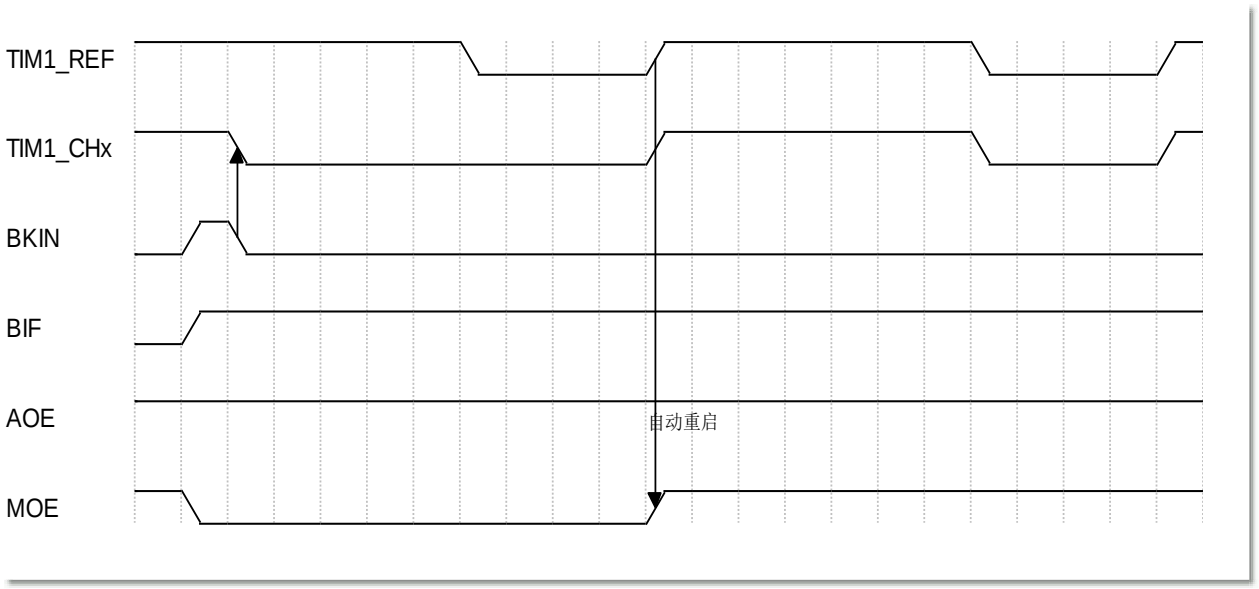


图 10.34 PWM 的自动重启

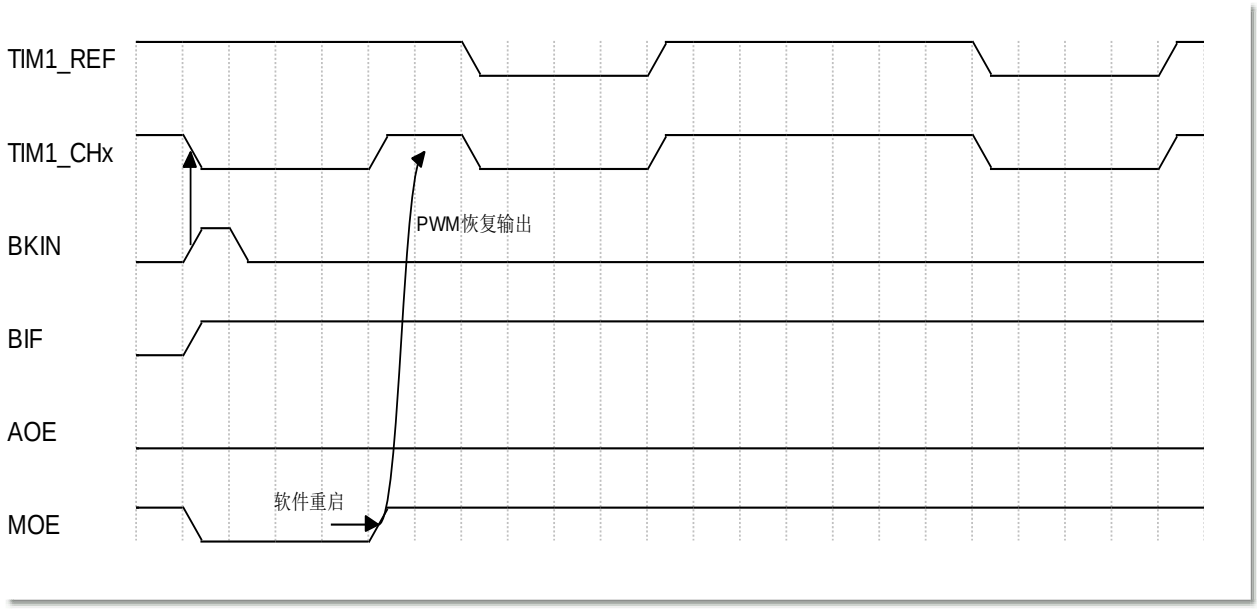


图 10.35 PWM 的软件重启

10.3.6. 前沿消隐

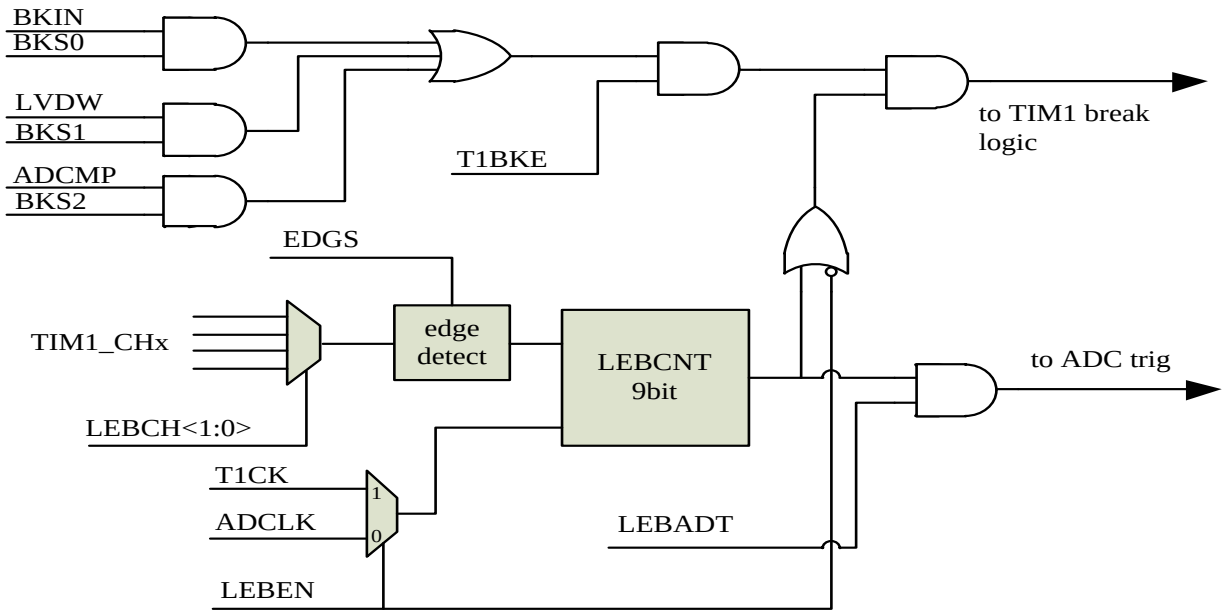


图 10.36 前沿消隐原理框图

在高速开关应用中，开关通常会产生极大的瞬变，这些瞬变可能会导致测量误差。利用前沿消隐（LEB）功能，应用程序可以忽略 PWM 输出边沿附近发生的瞬变。

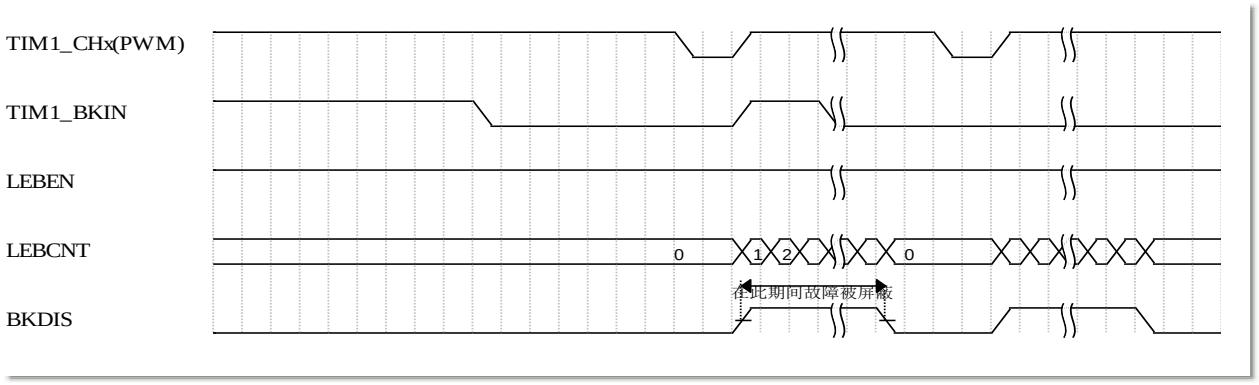


图 10.37 前沿消隐时序示意图

LEBCH 用于选择被消隐 TIM1 的 PWM 通道，EDGS 选择边沿类型。当 LEBEN 为 1，PWM 边沿将触发 LEB 定时器计数，时钟源为 TIM1 时钟，直到计数值等于 LEBPR，LEB 定时器停止计数，这段时间为消隐周期，期间所发生的刹车事件将被忽略；在消隐周期内如果再次发生有效的 PWM 边沿，则 LEB 定时器将清 0，重新开始计数。

注意：

(1) LEB 定时器和 ADC 延时定时器复用了同一个 9bit 计数器，当 LEBEN 为 1 时，原 ADC 的延时触发功能被禁止，但如果 LEBADT 为 1，LEB 定时器溢出将触发一次 AD 转换。

(2) 寄存器 ADCON3 中 ADCMPEN 位的关闭能将 ADCMP 产生的刹车事件清除。

10.4. 与 TIM1 相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
TIM1CR1	0x211	T1ARPE	T1CMS[1:0]		T1DIR	T1OPM	T1URS	T1UDIS	T1CEN	0000 0000
TIM1SMCR	0x213	—	T1TS[2:0]			—	T1SMS[2:0]			-000 -000
TIM1IER	0x215	T1BIE	T1TIE	—	T1CC4IE	T1CC3IE	T1CC2IE	T1CC1IE	T1UIE	00-0 0000
TIM1SR1	0x216	T1BIF	T1TIF	—	T1CC4IF	T1CC3IF	T1CC2IF	T1CC1IF	T1UIF	00-0 0000
TIM1SR2	0x217	—	—	—	T1CC4OF	T1CC3OF	T1CC2OF	T1CC1OF	—	---0 000-
TIM1EGR	0x218	T1BG	—	—	T1CC4G	T1CC3G	T1CC2G	T1CC1G	—	0--0 000-
TIM1CCMR1 (output mode)	0x219	—	T1OC1M[2:0]			T1OC1PE	—	T1CC1S[1:0]		-000 0-00
TIM1CCMR1 (input mode)		T1IC1F[3:0]				T1IC1PSC[1:0]		T1CC1S[1:0]		0000 0000
TIM1 CCMR2 (output mode)	0x21A	—	T1OC2M[2:0]			T1OC2PE	—	T1CC2S[1:0]		-000 0-00
TIM1CCMR2 (input mode)		T1IC2F[3:0]				T1IC2PSC[1:0]		T1CC2S[1:0]		0000 0000
TIM1CCMR3 (output mode)	0x21B	—	T1OC3M[2:0]			T1OC3PE	—	T1CC3S[1:0]		-000 0-00
TIM1CCMR3 (input mode)		T1IC3F[3:0]				T1IC3PSC[1:0]		T1CC3S[1:0]		0000 0000
TIM1CCMR4 (output mode)	0x21C	—	T1OC4M[2:0]			T1OC4PE	—	T1CC4S[1:0]		-000 0-00
TIM1CCMR4 (input mode)		T1IC4F[3:0]				T1IC4PSC[1:0]		T1CC4S[1:0]		0000 0000
TIM1CCER1	0x21D	T1CC2NP	T1CC2NE	T1CC2P	T1CC2E	T1CC1NP	T1CC1NE	T1CC1P	T1CC1E	0000 0000
TIM1CCER2	0x21E	—	—	T1CC4P	T1CC4E	T1CC3NP	T1CC3NE	T1CC3P	T1CC3E	--00 0000
TIM1CNTRH	0x28C	T1CNT[15:8]								0000 0000
TIM1CNTRL	0x28D	T1CNT[7:0]								0000 0000
TIM1PSCRH	0x28E	T1PSC[15:8]								0000 0000
TIM1PSCRL	0x28F	T1PSC[7:0]								0000 0000
TIM1ARRH	0x290	T1ARR[15:8]								1111 1111
TIM1ARRL	0x291	T1ARR[7:0]								1111 1111
TIM1RCR	0x292	T1REP[7:0]								0000 0000
TIM1CCR1H	0x293	T1CCR1[15:8]								0000 0000
TIM1CCR1L	0x294	T1CCR1[7:0]								0000 0000
TIM1CCR2H	0x295	T1CCR2[15:8]								0000 0000
TIM1CCR2L	0x296	T1CCR2[7:0]								0000 0000
TIM1CCR3H	0x297	T1CCR3[15:8]								0000 0000
TIM1CCR3L	0x298	T1CCR3[7:0]								0000 0000
TIM1CCR4H	0x299	T1CCR4[15:8]								0000 0000
TIM1CCR4L	0x29A	T1CCR4[7:0]								0000 0000
TIM1BKR	0x29B	T1MOE	T1AOE	T1BKP	T1BKE	T1OSSR	T1OSSI	T1LOCK[1:0]		0000 0000
TIM1DTR	0x29C	T1DTG[7:0]								0000 0000
TIM1OISR	0x29D	—	T1OIS4	T1OIS3N	T1OIS3	T1OIS2N	T1OIS2	T1OIS1N	T1OIS1	-000 0000
LEBCON	0x41C	LEBEN	LEBCH[1:0]		—	EDGS	BKS[2:0]			000- 0000

注意：

TIM1 寄存器中的保留位必须保持为复位值，不能更改，否则可能出现预想不到的情况。

10.4.1. TIM1CR1，地址：0x211

Bit	7	6	5	4	3	2	1	0
Name	T1ARPE	T1CMS[1:0]		T1DIR	T1OPM	T1URS	T1UDIS	T1CEN
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

7	**T1ARPE**：自动预装载允许位 0：TIM1ARRH/L寄存器没有缓冲，它可以被直接写入； 1：TIM1ARRH/L 寄存器由预装载缓冲器缓冲。
6:5	**T1CMS[1:0]**：选择中央对齐模式 00：边沿对齐模式。计数器依据方向位(T1DIR)向上或向下计数。 01：中央对齐模式1。计数器交替地向上和向下计数。配置为输出的通道(TIM1CCMRx寄存器中CCxS=00)的输出比较中断标志位，只在计数器向下计数时被置1。 10：中央对齐模式2。计数器交替地向上和向下计数。配置为输出的通道(TIM1CCMRx寄存器中CCxS=00)的输出比较中断标志位，只在计数器向上计数时被置1。 11：中央对齐模式3。计数器交替地向上和向下计数。配置为输出的通道(TIM1CCMRx寄存器中CCxS=00)的输出比较中断标志位，在计数器向上和向下计数时均被置1。 注1：在计数器开启时(T1CEN=1)，不允许从边沿对齐模式转换到中央对齐模式。
4	**T1DIR**：方向 0：计数器向上计数； 1：计数器向下计数。 注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。
3	**T1OPM**：单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除T1CEN位)时，计数器停止。
2	**T1URS**：更新请求源 0：如果T1UDIS允许产生更新事件，则下述任一事件产生一个更新中断： 寄存器被更新(计数器上溢/下溢) 复位触发事件产生的更新 1：如果T1UDIS允许产生更新事件，则只有当下列事件发生时才产生更新中断，并T1UIF置1： 计数器上溢/下溢
1	**T1UDIS**：禁止更新 0：一旦下列事件发生，产生更新(UEV)事件： 计数器溢出/下溢 时钟/触发模式控制器产生的硬件复位被缓存的寄存器被装入它们的预装载值 1：不产生更新事件，影子寄存器(ARR_SHAD、PSC_SHAD、CCRx_SHAD)保持它们的值。如果触发复位模式下触发事件到来时，计数器和预分频器会被重新初始化。
0	**T1CEN**：允许计数器 0：禁止计数器； 1：使能计数器。 注：在软件设置了T1CEN位后，门控模式才能工作。

10.4.2. TIM1SMCR，地址：0x213

Bit	7	6	5	4	3	2	1	0
Name	reserved	T1TS[2:0]			reserved	T1SMS[2:0]		
Reset	—	0	0	0	—	0	0	0
Type	RO-0	RW	RW	RW	RO-0	RW	RW	RW

7	**保留位**
6:4	**T1TS[2:0]**：触发选择 这3位选择用于选择同步计数器的触发输入。 000：内部触发ITR0连接到TIM6 TRGO **(此设计没有TIM6，所以固定接0)** 001：保留 010：内部触发ITR2连接到TIM5 TRGO**(此设计没有TIM5，所以固定接0)** 011：保留 100：TI1的边沿检测器(TI1F_ED) 101：滤波后的定时器输入1(TI1FP1) 110：滤波后的定时器输入2(TI2FP2) 111：**禁止配置** 注：这些位只能在SMS=3'b000时被改变，以避免在改变时产生错误的边沿检测。
3	**保留位**
2:0	**T1SMS**：时钟/触发/从模式选择 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：时钟/触发控制器禁止 – 如果T1CEN=1，则预分频器直接由内部时钟驱动。 100：复位模式 – 在选中的触发输入(TRGI)的上升沿时重新初始化计数器，并且产生一个更新寄存器的信号。 101：门控模式 – 当触发输入(TRGI)为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110：触发模式 – 计数器在触发输入TRGI的上升沿启动(但不复位)，只有计数器的启动是受控的。 **注：(1)其他值禁止配置** **(2)只有在输入捕捉模式下才能配置为复位模式；在输出比较模式下，禁止配置为复位模式** **(3)配置成门控模式/触发模式时，捕捉功能可正常使用**

10.4.3. TIM1IER，地址：0x215

Bit	7	6	5	4	3	2	1	0
Name	T1BIE	T1TIE	reserved	T1CC4IE	T1CC3IE	T1CC2IE	T1CC1IE	T1UIE
Reset	0	0	—	0	0	0	0	0
Type	RW	RW	RO-0	RW	RW	RW	RW	RW

7	**T1BIE**：允许刹车中断 0：禁止刹车中断； 1：允许刹车中断。
6	**T1TIE**：触发中断使能 0：禁止触发中断； 1：使能触发中断。
5	**保留位**
4	**T1CC4IE**：允许捕获/比较4中断 0：禁止捕获/比较4中断； 1：允许捕获/比较4中断。
3	**T1CC3IE**：允许捕获/比较3中断 0：禁止捕获/比较3中断； 1：允许捕获/比较3中断。
2	**T1CC2IE**：允许捕获/比较2中断 0：禁止捕获/比较2中断； 1：允许捕获/比较2中断。
1	**T1CC1IE**：允许捕获/比较1中断 0：禁止捕获/比较1中断； 1：允许捕获/比较1中断。
0	**T1UIE**：允许更新中断 0：禁止更新中断； 1：允许更新中断。

10.4.4. TIM1SR1，地址：0x216

Bit	7	6	5	4	3	2	1	0
Type	T1BIF	T1TIF	reserved	T1CC4IF	T1CC3IF	T1CC2IF	T1CC1IF	T1UIF
Reset	0	0	—	0	0	0	0	0
Type	R-W0	R-W0	RO-0	R-W0	R-W0	R-W0	R-W0	R-W0

7	**T1BIF**：刹车中断标记**(写1清0，写0无效)** 一旦刹车输入有效，由硬件对该位置1。如果刹车输入无效，则该位可由软件清0。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。
6	**T1TIF**：触发器中断标记**(写1清0，写0无效)** 当发生触发事件(当从模式控制器处于除门控模式外的其它模式时，在TRGI输入端检测到有效边沿，或门控模式下的任一边沿)时由硬件对该位置1。它由软件清0。 0：无触发器事件产生； 1：触发中断等待响应。
5	**保留位**
4	**T1CC4IF**：捕获/比较4中断标记**(写1清0，写0无效)** 参考CC1IF描述。
3	**T1CC3IF**：捕获/比较3中断标记**(写1清0，写0无效)** 参考CC1IF描述。
2	**T1CC2IF**：捕获/比较2中断标记**(写1清0，写0无效)** 参考CC1IF描述。
1	**T1CC1IF**：捕获/比较1中断标记　如果通道1配置为输出模式：**(写1清0，写0无效)** 当计数器值与比较值匹配时该位由硬件置1，但在中心对称模式下除外(参考TIM1_CR1寄存器的T1CMS位)。它由软件清0。 0：无匹配发生； 1：CNT的值与T1CCR1值匹配。 注：在中心对称模式下，当计数器值为0时，向上计数，当计数器值为T1ARR时，向下计数（它从0向上计数到T1ARR-1，再由T1ARR向下计数到1）。因此，对所有的T1SMS位值，这两个值都不置标记。但是，如果T1CCR1>T1ARR，则当CNT达到T1ARR值时，T1CC1IF置1。 如果通道1配置为输入模式：当捕获事件发生时该位由硬件置1，它由软件清0或通过读TIM1CCR1L清0。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至TIM1CCR1H/L(在IC1上检测到与所选极性相同的边沿)。
0	**T1UIF**：更新中断标记**(写1清0，写0无效)** 当产生更新事件时该位由硬件置1。它由软件清0。 0：无更新事件产生； 1：更新事件等待响应。

10.4.5. TIM1SR2，地址：0x217

Bit	7	6	5	4	3	2	1	0
Name	reserved			T1CC4OF	T1CC3OF	T1CC2OF	T1CC1OF	reserved
Reset	—	—	—	0	0	0	0	—
Type	RO-0	RO-0	RO-0	RW	RW	RW	RW	RO-0

7:5	**保留位**
4	**T1CC4OF**：捕获/比较4重复捕获标记**(写1清0，写0无效)** 参见CC1OF描述。
3	**T1CC3OF**：捕获/比较3重复捕获标记**(写1清0，写0无效)** 参见CC1OF描述。
2	**T1CC2OF**：捕获/比较2重复捕获标记**(写1清0，写0无效)** 参见CC1OF描述。
1	**T1CC1OF**：捕获/比较1重复捕获标记**(写1清0，写0无效)** 仅当相应的通道被配置为输入捕获时，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到TIM1CCR1H/L寄存器时，T1CC1IF的状态已经为1。
0	**保留位**

10.4.6. TIM1EGR，地址：0x218

Bit	7	6	5	4	3	2	1	0
Name	T1BG	reserved	reserved	T1CC4G	T1CC3G	T1CC2G	T1CC1G	reserved
Reset	0	—	—	0	0	0	0	—
Type	R0-W	RO-0	RO-0	R0-W	R0-W	R0-W	R0-W	RO-0

7	**T1BG**：产生刹车事件 该位由软件置1，用于产生一个刹车事件，由硬件自动清0。 0：无动作； 1：产生一个刹车事件。此时T1MOE=0、T1BIF=1，若开启对应的中断(T1BIE=1)，则产生相应的中断。
6:5	**保留位**
4	**T1CC4G**：产生捕获/比较4事件 参考CC1G描述。
3	**T1CC3G**：产生捕获/比较3事件 参考CC1G描述。
2	**T1CC2G**：产生捕获/比较2事件 参考CC1G描述。
1	**T1CC1G**：产生捕获/比较1事件 该位由软件置1，用于产生一个捕获/比较事件，由硬件自动清0。 0：无动作； 1：在通道1上产生一个捕获/比较事件： 若通道1配置为输出： 设置T1CC1IF=1，若开启对应的中断，则产生相应的中断。若通道1配置为输入： 当前的计数器值被捕获至TIM1CCR1H/L寄存器，设置T1CC1IF=1，若开启对应的中断，则产生相应的中断。若T1CC1IF已经为1，则设置T1CC1OF=1。
0	**保留位**

10.4.7. TIM1CCMR1，地址：0x219

配置为输出比较模式：

Bit	7	6	5	4	3	2	1	0
Name	reserved	T1OC1M[2:0]			T1OC1PE	reserved	T1CC1S[1:0]	
Rese	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RW	RW	RO-0	RW	RW

7	**保留位**
6:4	**T10C1M[2:0]**：输出比较1模式 该3位定义了输出参考信号OC1REF的动作，而OC1REF决定了OC1的值。OC1REF是高电平有效，而OC1的有效电平取决于CC1P位。 000：冻结。输出实际比较值(CCRx_SHAD)与计数器TIM1_CNT间的比较对OC1REF不起作用； 001：匹配时设置通道1的输出为有效电平。当计数器TIM1_CNT的值与捕获/比较寄存器1(TIM1_CCR1)相同时，强制OC1REF为高。 010：匹配时设置通道1的输出为无效电平。当计数器TIM1_CNT的值与捕获/比较寄存器1(TIM1_CCR1)相同时，强制OC1REF为低。 011：翻转。当TIM1_CCR1=TIM1_CNT时，翻转OC1REF的电平。 100：强制为无效电平。强制OC1REF为低。 101：强制为有效电平。强制OC1REF为高。 110：PWM模式1－ 在向上计数时，一旦TIM1_CNT<实际比较值(CCRx_SHAD)时OC1REF为有效电平，否则为无效电平；在向下计数时，一旦TIM1_CNT>实际比较值(CCRx_SHAD)时，OC1REF为无效电平，否则为有效电平。 111：PWM模式2－ 在向上计数时，一旦TIM1_CNT<实际比较值(CCRx_SHAD)时OC1REF为无效电平，否则为有效电平；在向下计数时，一旦TIM1_CNT>实际比较值(CCRx_SHAD)时，OC1REF为有效电平，否则为无效电平。 注1：一旦LOCK级别设为3(TIM1_BKR寄存器中的LOCK位)并且T1CC1S=00(该通道配置成输出) 则该位不能被修改。 注2：在PWM模式1或PWM模式2中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到PWM模式时，OC1REF电平才改变。(参考17.5.7PWM模式)
3	**T1OC1PE**：输出比较1预装载使能 0：禁止TIM1CCR1H/L寄存器的预装载功能，可随时写入T1CCR1预加载寄存器，并且新写入的数值立即起作用。 1：开启TIM1CCR1H/L寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM1CCR1H/L的预装载值在更新事件到来时被加载至当前寄存器中。 注1：一旦LOCK级别设为3(TIM1BKR寄存器中的T1LOCK位)并且T1CC1S=00(该通道配置成输出) 则该位不能被修改。 注2：为了操作正确，在PWM模式下必须使能预装载功能。但在单脉冲模式下(TIM1CR1寄存器的T1OPM=1)，它不是必须的。
2	**保留位**
1:0	**T1CC1S[1:0]**：捕获/比较1 选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道1被配置为输出； 01：通道1被配置为输入，IC1映射在TI1FP1上； 10：通道1被配置为输入，IC1映射在TI2FP1上； 11：通道1被配置为输入，IC1映射在TRC上。此模式仅工作在内部触发器输入被选中时(由TIM1SMCR寄存器的T1TS位选择)。 注：T1CC1S仅在通道关闭时(TIM1CCER1寄存器的T1CC1E=0，T1CC1NE=0且已被更新)才是可写的。

配置为输入捕捉模式：

Name	T1IC1F[3:0]				T1IC1PSC[1:0]		T1CC1S[1:0]	
Type	RO	RO	RO	RO	RO	RO	RO	RO
7:4	**T1IC1F[3:0]**：输入捕获1滤波器 这几位定义了TI1输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，只有发生了N个事件后输出的跳变才被认为有效。 0000：无滤波器，fSAMPLING=fMASTER 0001：采样频率fSAMPLING=fMASTER，N=2 0010：采样频率fSAMPLING=fMASTER，N=4 0011：采样频率fSAMPLING=fMASTER，N=8 0100：采样频率fSAMPLING=fMASTER/2，N=6 0101：采样频率fSAMPLING=fMASTER/2，N=8 0110：采样频率fSAMPLING=fMASTER/4，N=6 0111：采样频率fSAMPLING=fMASTER/4，N=8 1000：采样频率fSAMPLING=fMASTER/8，N=6 1001：采样频率fSAMPLING=fMASTER/8，N=8 1010：采样频率fSAMPLING=fMASTER/16，N=5 1011：采样频率fSAMPLING=fMASTER/16，N=6 1100：采样频率fSAMPLING=fMASTER/16，N=8 1101：采样频率fSAMPLING=fMASTER/32，N=5 1110：采样频率fSAMPLING=fMASTER/32，N=6 1111：采样频率fSAMPLING=fMASTER/32，N=8							
3:2	**T1IC1PSC[1:0]**：输入/捕获1预分频器 这2位定义了通道1输入(IC1)的预分频系数。 一旦T1CC1E=0(TIM1CCER寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每2个事件触发一次捕获； 10：每4个事件触发一次捕获； 11：每8个事件触发一次捕获。							
1:0	**T1CC1S[1:0]**：捕获/比较1 选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道1被配置为输出； 01：通道1被配置为输入，IC1映射在TI1FP1上； 10：通道1被配置为输入，IC1映射在TI2FP1上； 11：通道1被配置为输入，IC1映射在TRC上。此模式仅工作在内部触发器输入被选中时(由TIM1SMCR寄存器的T1TS位选择)。 注：T1CC1S仅在通道关闭时(TIM1CCER1寄存器的T1CC1E=0，T1CC1NE=0且已被更新)才是可写的。							

10.4.8. TIM1CCMR2，地址：0x21A

配置为输出比较模式：

Bit	7	6	5	4	3	2	1	0
Name	reserved	T1OC2M[2:0]			T1OC2PE	reserved	T1CC2S[1:0]	
Reset	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RW	RW	RO-0	RW	RW
7	**保留位**							
6:4	**T1OC2M[2:0]：**输出比较2模式							
3	**T1OC2PE：**输出比较2预装载使能							
2	**保留位**							
1:0	**T1CC2S[1:0]：**捕获/比较2选择。 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道2被配置为输出； 01：通道2被配置为输入，IC2映射在TI2FP2上； 10：通道2被配置为输入，IC2映射在TI1FP2上； 11：预留 注：T1CC2S仅在通道关闭时(TIM1CCER1寄存器的T1CC2E=0，T1CC2NE=0且已被更新)才是可写的。							

配置为输入捕捉模式：

Name	T1IC2F[3:0]				T1IC2PSC[1:0]		T1CC2S[1:0]	
Type	RO	RO	RO	RO	RO	RO	RO	RO
7:4	**T1IC2F[3:0]：**输入捕获2滤波器							
3:2	**T1IC2PSC[1:0]：**输入/捕获2预分频器							
1:0	**T1CC2S[1:0]：**捕获/比较2选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道2被配置为输出； 01：通道2被配置为输入，IC2映射在TI2FP2上； 10：通道2被配置为输入，IC2映射在TI1FP2上； 11：通道2被配置为输入，IC2映射在TRC上。此模式仅工作在内部触发器输入被选中时(由TIM1SMCR寄存器的T1TS位选择)。 注：T1CC2S仅在通道关闭时(TIM1CCER1寄存器的T1CC2E=0，T1CC2NE=0且已被更新)才是可写的。							

10.4.9. TIM1CCMR3，地址：0x21B

配置为输出比较模式：

Bit	7	6	5	4	3	2	1	0
Type	reserved	T1OC3M[2:0]			T1OC3PE	reserved	T1CC3S[1:0]	
Reset	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RO-0	RW	RO-0	RW	RW
7	**保留位**							
6:4	**T1OC3M[2:0]**：输出比较3模式							
3	**T1OC3PE**：输出比较3预装载使能							
2	**保留位**							
1:0	**T1CC3S[1:0]**：捕获/比较3选择。 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道3被配置为输出； 01：通道3被配置为输入，IC3映射在TI3FP3上； 10：通道3被配置为输入，IC3映射在TI4FP3上； 11：预留 注：T1CC3S仅在通道关闭时(TIM1CCER2寄存器的T1CC3E=0，T1CC3NE=0且已被更新)才是可写的。							

配置为输入捕捉模式：

Name	T1IC3F[3:0]				T1IC3PSC[1:0]		T1CC3S[1:0]	
Type	RO	RO	RO	RO	RO	RO	RO	RO
7:4	**T1IC3F[3:0]**：输入捕获3滤波器							
3:2	**T1IC3PSC[1:0]**：输入/捕获3预分频器							
1:0	**T1CC3S[1:0]**：捕获/比较3选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道3被配置为输出； 01：通道3被配置为输入，IC3映射在TI3FP3上； 10：通道3被配置为输入，IC3映射在TI4FP3上； 11：预留 注：T1CC3S仅在通道关闭时(TIM1CCER2寄存器的T1CC3E=0，T1CC3NE=0且已被更新)才是可写的。							

10.4.10. TIM1CCMR4，地址：0x21C

配置为输出比较模式：

Bit	7	6	5	4	3	2	1	0
Name	reserved	T1OC4M[2:0]			T1OC4PE	reserved	T1CC4S[1:0]	
Reset	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RO-0	RW	RO-0	RW	RW
7	**保留位**							
6:4	**T1OC4M[2:0]**：输出比较4模式							
3	**T1OC4PE**：输出比较4预装载使能							
2	**保留位**							
1:0	**T1CC4S[1:0]**：捕获/比较4选择。 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道4被配置为输出； 01：通道4被配置为输入，IC4映射在TI3FP4上； 10：通道4被配置为输入，IC4映射在TI4FP4上； 11：预留 注：T1CC4S仅在通道关闭时(TIM1CCER2寄存器的T1CC4E=0)才是可写的。							

配置为输入捕捉模式：

Name	T1IC4F[3:0]				T1IC4PSC[1:0]		T1CC4S[1:0]	
Type	RO	RO	RO	RO	RO	RO	RO	RO
7:4	**T1IC4F[3:0]**：输入捕获4滤波器							
3:2	**T1IC4PSC[1:0]**：输入/捕获4预分频器							
1:0	**T1CC4S[1:0]**：捕获/比较4选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道4被配置为输出； 01：通道4被配置为输入，IC4映射在TI3FP4上； 10：通道4被配置为输入，IC4映射在TI4FP4上； 11：预留 注：T1CC4S仅在通道关闭时(TIM1CCER2寄存器的T1CC4E=0)才是可写的。							

10.4.11. TIM1CCER1，地址：0x21D

Bit	7	6	5	4	3	2	1	0
Name	T1CC2NP	T1CC2NE	T1CC2P	T1CC2E	T1CC1NP	T1CC1NE	T1CC1P	T1CC1E
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

7	**T1CC2NP**：输入捕获/比较2互补输出极性。参考T1CC1NP的描述。
6	**T1CC2NE**：输入捕获/比较2互补输出使能。参考T1CC1NE的描述。
5	**T1CC2P**：输入捕获/比较2输出极性。参考T1CC1P的描述。
4	**T1CC2E**：输入捕获/比较2输出使能。参考T1CC1E的描述。
3	**T1CC1NP**：输入捕获/比较1互补输出极性 0：OC1N高电平有效； 1：OC1N低电平有效。 注1：一旦LOCK级别(TIM1BKR寄存器中的T1LOCK位)设为3或2且T1CC1S=00(通道配置为输出) 则该位不能被修改。
2	**T1CC1NE**：输入捕获/比较1互补输出使能 0：关闭－ OC1N禁止输出，因此OC1N的输出电平依赖于T1MOE、T1OSSI、T1OSSR、T1OIS1、T1OIS1N和T1CC1E位的值。 1：开启－ OC1N信号输出到对应的输出引脚，其输出电平依赖于T1MOE、T1OSSI、T1OSSR、T1OIS1、T1OIS1N和T1CC1E位的值。
1	**T1CC1P**：输入捕获/比较1输出极性选择 通道1配置为输出： 0：OC1高电平有效； 1：OC1低电平有效。 通道1配置为触发输入： 0：触发发生在TI1F的高电平或上升沿； 1：触发发生在TI1F的低电平或下降沿。 CC1通道配置为捕捉输入： 0：捕捉发生在TI1F的高电平或上升沿； 1：捕捉发生在TI1F的低电平或下降沿。 注1：一旦LOCK级别(TIM1BKR寄存器中的T1LOCK位)设为3或2，则该位不能被修改。
0	**T1CC1E**：输入捕获/比较1输出使能 CC1通道配置为输出： 0： 关闭－ OC1禁止输出，因此OC1的输出电平依赖于T1MOE、T1OSSI、T1OSSR、T1OIS1、T1OIS1N和T1CC1NE位的值。 1： 开启－ OC1信号输出到对应的输出引脚，其输出电平依赖于T1MOE、T1OSSI、T1OSSR、T1OIS1 、 T1OIS1N 和 T1CC1NE 位 的 值 。 通道1配置为输入： 该位决定了计数器的值是否能被捕获到TIM1CCR1寄存器中。 0：捕获禁止； 1：捕获使能。

10.4.12. TIM1CCER2，地址：0x21E

Bit	7	6	5	4	3	2	1	0
Name	reserved	reserved	T1CC4P	T1CC4E	T1CC3NP	T1CC3NE	T1CC3P	T1CC3E
Reset	—	—	0	0	0	0	0	0
Type	RO-0	RO-0	RW	RW	RW	RW	RW	RW
7:6	**保留位**							
5	**T1CC4P**：输入捕获/比较4输出极性。参考T1CC1P的描述。							
4	**T1CC4E**：输入捕获/比较4输出使能。参考T1CC1E 的描述。							
3	**T1CC3NP**：输入捕获/比较3互补输出极性。参考T1CC1NP的描述。							
2	**T1CC3NE**：输入捕获/比较3互补输出使能。参考T1CC1NE的描述。							
1	**T1CC3P**：输入捕获/比较3输出极性。参考T1CC1P的描述。							
0	**T1CC3E**：输入捕获/比较3输出使能。参考T1CC1E 的描述。							

10.4.13. TIM1CNTRH，地址：0x28C

Bit	7	6	5	4	3	2	1	0
Name	T1CNT[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CNT[15:8]**：计数器的高8位值							

10.4.14. TIM1CNTRL，地址：0x28D

Bit	7	6	5	4	3	2	1	0
Name	T1CNT[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CNT[7:0]**：计数器的低8位值							

10.4.15. TIM1PSCRH，地址：0x28E

Bit	7	6	5	4	3	2	1	0
Name	T1PSC[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1PSC[15:7]**：预分频器的高8位值 预分频器用于对CK_PSC进行分频。 计数器的时钟频率(f_{CK_CNT})等于f_{CK_PSC}/(PSCR[15:0]+1)。 PSCR为实际装入预分频器影子寄存器的值。这意味着为了使新的值起作用，必须产生一个更新事件或者T1CEN=0。							

10.4.16. TIM1PSCRL，地址：0x28F

Bit	7	6	5	4	3	2	1	0
Name	T1PSC[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1PSC[7:0]：**预分频器的低8位值 预分频器用于对CK_PSC进行分频。 计数器的时钟频率(f_{CK_CNT})等于f_{CK_PSC}/(PSCR[15:0]+1)。 PSCR为实际装入预分频器影子寄存器的值。这意味着为了使新的值起作用，必须产生一个更新事件或者T1CEN=0。 注：在配置的时候，需要先将周期，占空比，模式等寄存器配置完成后并且在T1CEN使能之前配置预分频寄存器。							

10.4.17. TIM1ARRH，地址：0x290

Bit	7	6	5	4	3	2	1	0
Name	T1ARR[15:8]							
Reset	1	1	1	1	1	1	1	1
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1ARR[15:8]:** 自动重装载的高8位值 T1ARR为将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。							

10.4.18. TIM1ARRL，地址：0x291

Bit	7	6	5	4	3	2	1	0
Name	T1ARR[7:0]							
Reset	1	1	1	1	1	1	1	1
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1ARR[7：0]:** 自动重装载的低8位值 T1ARR为将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。							

10.4.19. TIM1RCR，地址：0x292

Bit	7	6	5	4	3	2	1	0
Name	T1REP[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1REP[7:0]:** 重复计数器的值 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下重复计数器达到0，会产生一个更新事件并且重复计数器重新从T1REP值开始计数。由于重复计数器只有在周期更新事件(UEV)发生时才重载T1REP值，因此对TIM1RCR寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在PWM模式中，(T1REP+1)对应着： --- 在边沿对齐模式下，PWM周期的数目； --- 在中心对称模式下，PWM半周期的数目；							

10.4.20. TIM1CCR1H，地址：0x293

Bit	7	6	5	4	3	2	1	0
Name	T1CCR1[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CCR1[15:8]:** 捕获/比较1的高8位值 若通道1配置为输出(TIM1CCMR1的T1CC1S=00)： T1CCR1H/L为装入当前捕获/比较1寄存器的值(预装载值)。 如果在TIM1CCMR1寄存器(T1OC1PE位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。 当前捕获/比较寄存器的值同计数器TIM1_CNT的值相比较，并在OC1端口上产生输出信号。 若通道1配置为输入： T1CCR1H/L包含了上一次输入捕获1事件(IC1)发生时的计数器值（此时该寄存器为只读）。							

10.4.21. TIM1CCR1L，地址：0x294

Bit	7	6	5	4	3	2	1	0
Name	T1CCR1[7:0]							
Reset	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CCR1[7:0]**: 捕获/比较1的低8位值							

10.4.22. TIM1CCR2H，地址：0x295

Bit	7	6	5	4	3	2	1	0
Name	T1CCR2[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CCR2[15:8]:** 捕获/比较2的高8位值 若通道2配置为输出(TIM1CCMR2的T1CC2S=00)： T1CCR2H/L包含了装入当前捕获/比较2寄存器的值(预装载值)。 如果在TIM1CCMR2寄存器(T1OC2PE位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。 当前捕获/比较寄存器的值同计数器TIM1_CNT的值相比较，并在OC2端口上产生输出信号。 若通道2配置为输入： T1CCR2H/L包含了由上一次输入捕获2事件(IC2)传输的计数器值(此时该寄存器为只读)。							

10.4.23. TIM1CCR2L，地址：0x296

Bit	7	6	5	4	3	2	1	0
Name	T1CCR2[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CCR2[7:0]**: 捕获/比较2的低8位值							

10.4.24. TIM1CCR3H，地址：0x297

Bit	7	6	5	4	3	2	1	0
Name	T1CCR3[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CCR3[15:8]:** 捕获/比较3的高8位值 若通道2配置为输出(TIM1CCMR3的T1CC3S=00)： T1CCR3H/L包含了装入当前捕获/比较3寄存器的值(预装载值)。 如果在TIM1CCMR3寄存器(T1OC3PE位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。 当前捕获/比较寄存器的值同计数器TIM1_CNT的值相比较，并在OC3端口上产生输出信号。 若通道3配置为输入： T1CCR3H/L包含了由上一次输入捕获3事件(IC3)传输的计数器值(此时该寄存器为只读)。							

10.4.25. TIM1CCR3L，地址：0x298

Bit	7	6	5	4	3	2	1	0
Name	T1CCR3[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CCR3[7:0]**：捕获/比较3的低8位值							

10.4.26. TIM1CCR4H，地址：0x299

Bit	7	6	5	4	3	2	1	0
Name	T1CCR4[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CCR4[15:8]:** 捕获/比较4的高8位值 若通道2配置为输出(TIM1CCMR4的T1CC4S=00)： T1CCR4H/L包含了装入当前捕获/比较4寄存器的值(预装载值)。 如果在TIM1CCMR4寄存器(T1OC4PE位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。 当前捕获/比较寄存器的值同计数器TIM1_CNT的值相比较，并在OC4端口上产生输出信号。 若通道4配置为输入： T1CCR4H/L包含了由上一次输入捕获4事件(IC4)传输的计数器值(此时该寄存器为只读)。							

10.4.27. TIM1CCR4L，地址：0x29A

Bit	7	6	5	4	3	2	1	0
Name	T1CCR4[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1CCR4[7:0]**：捕获/比较4的低8位值							

10.4.28. TIM1BKR，地址：0x29B

Bit	7	6	5	4	3	2	1	0
Name	T1MOE	T1AOE	T1BKP	T1BKE	T1OSSR	T1OSSI	T1LOCK[1:0]	
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

7	**T1MOE:** 主输出使能 一旦刹车输入有效，该位被硬件异步清0。根据T1AOE位的设置值，该位可以由软件置1或被自动置1。它仅对配置为输出的通道有效。 0：禁止OC和OCN输出或强制为空闲状态； 1：如果设置了相应的使能位(TIM1CCERx寄存器的T1CCxE位)，则使能OC和OCN输出。
6	**T1AOE:** 自动输出使能 0：T1MOE只能被软件置1； 1：T1MOE能被软件置1或在下一个更新事件被自动置1(如果刹车输入无效)。 注：一旦LOCK级别(TIM1BKR寄存器中的T1LOCK位)设为1，则该位不能被修改。
5	**T1BKP:** 刹车输入极性（**只对故障源TIM1_BKIN有效**） 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦LOCK级别(TIM1BKR寄存器中的T1LOCK位)设为1，则该位不能被修改。
4	**T1BKE:** 刹车功能使能 0：禁止刹车输入(BRK)； 1：开启刹车输入(BRK)。 注：一旦LOCK级别(TIM1BKR寄存器中的T1LOCK位)设为1，则该位不能被修改。
3	**T1OSSR:** 运行模式下“关闭状态”选择 该位用于当T1MOE=1且通道为互补输出时。 0：当定时器不工作时，禁止OC/OCN输出(OC/OCN使能输出信号=0)； 1：当定时器不工作时，一旦CcxE=1或CcxNE=1，首先开启OC/OCN并输出无效电平，然后置OC/OCN使能输出信号=1。 注：一旦LOCK级别(TIM1BKR寄存器中的T1LOCK位)设为2，则该位不能被修改。
2	**T1OSSI:**空闲模式下“关闭状态”选择 该位用于当T1MOE=0且通道设为输出时。 0：当定时器不工作时，禁止OC/OCN输出(OC/OCN使能输出信号=0)； 1：当定时器不工作时，一旦CcxE=1或CcxNE=1，OC/OCN首先输出其空闲电平，然后OC/OCN使能输出信号=1。 注：一旦LOCK级别(TIM1BKR寄存器中的T1LOCK位)设为2，则该位不能被修改。
1:0	**T1LOCK[1:0]:** 锁定设置 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别1，不能写入TIM1BKR寄存器的T1BKE、T1BKP、T1AOE位和TIM1OISR寄存器的T1OISI位； 10：锁定级别2，不能写入锁定级别1中的各位，也不能写入输出极性位(一旦相关通道通过T1CCxS位设为输出，输出极性位是TIM1CCERx寄存器的T1CCxP位)以及T1OSSR/T1OSSI位； 11：锁定级别3，不能写入锁定级别2中的各位，也不能写入输出控制位(一旦相关通道通过T1CCxS位设为输出，输出控制位是TIM1CCMRx寄存器的T1OCIM/T1OCIPE位)； 注：在系统复位后，只能写一次LOCK位，一旦写入TIM1BDR寄存器，则其内容保持不变直至复位。

10.4.29. TIM1DTR，地址：0x29C

Bit	7	6	5	4	3	2	1	0
Name	T1DTG[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T1DTG[7:0]:** 死区发生器设置 这些位定义了插入互补输出之间的死区持续时间。假设DT表示其持续时间，tCK_PSC为TIM1的时钟脉冲： T1DTG[7:5]=0xx => DT=DTG[7:0]x tdtg，其中：tdtg =$t_{CK\_PSC.}$ (f1) T1DTG[7:5]=10x => DT=(64+DTG[5:0])x tdtg，其中：tdtg = $t_{CK\_PSC.}$ (f2) T1DTG[7:5]=110 => DT=(32+DTG[4:0])x t_{dtg}， 其 中 ：tdtg=8x $t_{CK\_PSC.}$ (f3) T1DTG[7:5]=111 => DT=(32+DTG[4:0])x tdtg， 其 中 ：tdtg=16x $t_{CK\_PSC.}$ (f4) 举列: 如果t_{CK_PSC} =125 ns (8 MHz)，可能的死区时间为： T1DTG[7:0] = 0到7Fh， 0到15875 ns， 步长时间为125 ns (参考f1) T1DTG[7:0] = 80h到BFh ，16μs到31750ns， 步长时间为250 ns (参考f2) T1DTG[7:0] = C0h到DFh， 32μs到63μs， 步长时间为1μs (参考f3) T1DTG[7:0] = E0h到FFh， 64μs到126μs， 步长时间为2 μs (参考f4) 注：一旦LOCK级别(TIM1BKR寄存器中的T1LOCK位)设为1、2或3，则不能修改这些位。							

10.4.30. TIM1OISR，地址：0x29D

Bit	7	6	5	4	3	2	1	0
Name	reserved	T1OIS4	T1OIS3N	T1OIS3	T1OIS2N	T1OIS2	T1OIS1N	T1OIS1
Reset	—	0	0	0	0	0	0	0
Type	RO-0	RW	RW	RW	RW	RW	RW	RW
7	**保留位**							
6	**T1OIS4**：输出空闲状态4(OC4输出)。参见T1OIS1位。							
5	**T1OIS3N**：输出空闲状态3(OC3N输出)。参见T1OIS1N位。							
4	**T1OIS3**：输出空闲状态3(OC3输出)。参见T1OIS1位。							
3	**T1OIS2N**：输出空闲状态2(OC2N输出)。参见T1OIS1N位。							
2	**T1OIS2**：输出空闲状态2(OC2输出)。参见T1OIS1位。							
1	**T1OIS1N**：输出空闲状态1(OC1N输出)。 0：当T1MOE=0时，则在一个死区时间后，OC1N=0； 1：当T1MOE=0时，则在一个死区时间后，OC1N=1。 注：已经设置了LOCK(TIM1BKR寄存器)级别1、2或3后，该位不能被修改。							
0	**T1OIS1**：输出空闲状态1(OC1输出)。 0：当T1MOE=0时，如果OC1N使能，则在一个死区后，OC1=0； 1：当T1MOE=0时，如果OC1N使能，则在一个死区后，OC1=1。 注：已经设置了LOCK(TIM1BKR寄存器)级别1、2或3后，该位不能被修改。							

10.4.31. LEBCON 寄存器，地址 0x41C

Bit	7	6	5	4	3	2	1	0
Name	LEBEN	LEBCH[1:0]		reserved	EDGS	BKS[2:0]		
Reset	0	0	0	—	0	0	0	0
Type	RW	RW	RW	RO-0	RW	RW	RW	RW

Bit	Name	Function
7	LEBEN	前沿消隐使能位（**仅当 ADGO=0 时可进行切换，否则 ADC 工作异常**） 1 = 使能 0 = 禁止
6:5	LEBCH[1:0]	前沿消隐通道选择 00 = TIM1_CH1 01 = TIM1_CH2 10 = TIM1_CH3 11 = TIM1_CH4
4	N/A	保留位，读 0
3	EDGS	PWM 消隐沿选择 0 = PWM 上升沿 1 = PWM 下降沿
2	BKS[2:0]	BKS[2:0]，TIM1 的故障源使能，高有效 BKS2：选择 ADC 阈值比较 BKS1：选择 LVD 检测 BKS0：选择 BKIN 管脚
1		
0		

11. 通用定时器 TIM2

11.1. 特性

tiemr2 的功能除捕捉比较通道数量不同以外，其他相同：

- 16bit 的向上计数，支持自动重载；
- 计数时钟预分频；
- 支持 1/2 个独立的捕捉比较通道，通道可支持：
  - 输入捕捉
  - 输出比较
  - PWM 产生
- 中断事件：
  - 更新事件：计数器溢出，计数器初始化
  - 输入捕捉事件
  - 输出比较事件

11.2. 原理框图

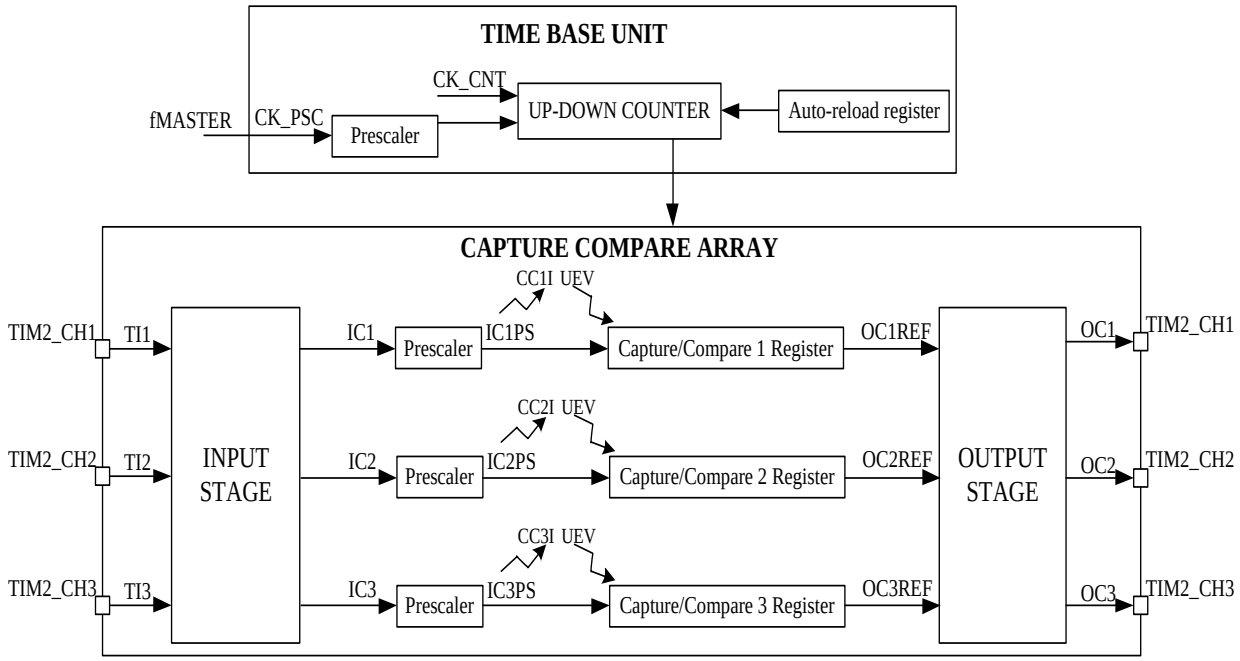


图 11.1 TIM2 原理框图

11.3. 功能描述

整个 TIM2 可以分为两个大的功能部分：计数基本单元和捕捉比较通道。计数基本单元分为向上计数器、自动加载寄存器、预分频器；捕捉比较通道分为捕捉输入通道，输出比较通道和输出控制。

11.3.1. 计数基本单元

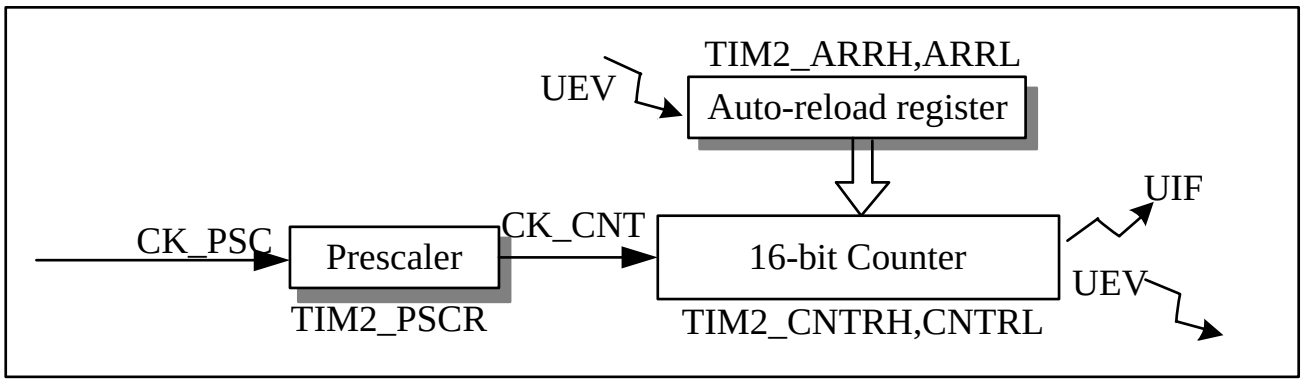


图 11.2 计数基本单元框图

计数基本单元包括：

- 16 位向上计数器
- 16 位自动重加载寄存器
- 4 位可编程预分频器

TIM2 没有重复计数器

11.3.1.1. 时钟源选择

时钟源可由 TCKSRC 寄存器进行配置：

- T2CKSRC[2:0]=000 时，系统时钟/主时钟为 TIM2 时钟
- T2CKSRC[2:0]=001 时，HIRC 为 TIM2 时钟
- T2CKSRC[2:0]=010 时，XT 时钟/外部时钟为 TIM2 时钟
- T2CKSRC[2:0]=011 时，HIRC 的 2 倍频为 TIM2 时钟
- T2CKSRC[2:0]=100 时，XT 时钟/外部时钟的 2 倍频为 TIM2 时钟
- T2CKSRC[2:0]=101 时，LIRC 为 TIM2 时钟
- T2CKSRC[2:0]=110 时，LP 时钟/外部时钟为 TIM2 时钟
- T2CKSRC[2:0]=111 时，LP 时钟/外部时钟的 2 倍频为 TIM2 时钟

11.3.1.2. 向上计数器

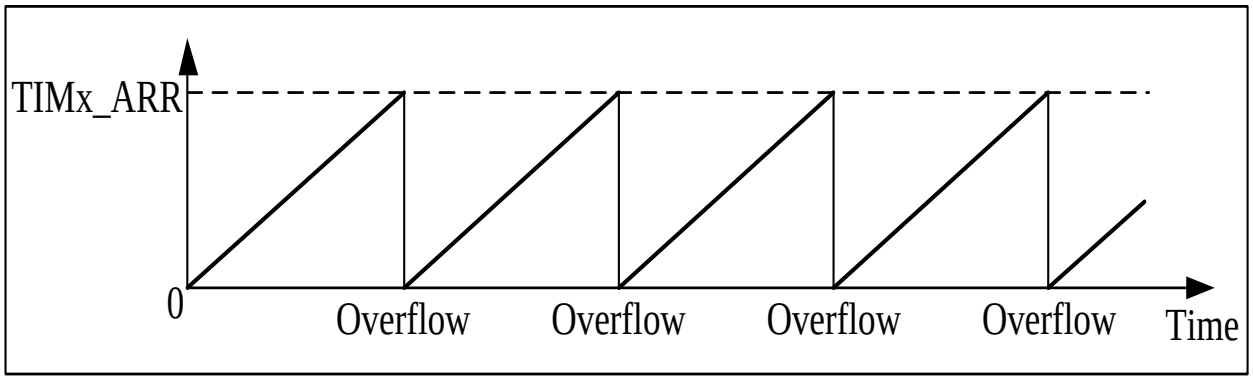


图 11.3 向上计数器

TIM2 计数器只能向上计数。计数器从 0 开始计数向上计数，计到 TIM1_ARR 寄存器所设数值。然后重新从 0 开始计数并产生一个计数器上溢事件；如果 T2UDIS 设为 0，那么还会产生一个更新事件 UEV。

11.3.1.3. 预分频器

计数时钟可以进行 4bit 的时钟预分频：

$f_{CK\_CNT} = f_{CK\_PSC}/2^{(PSCR[3:0])}$

预分频支持分频自动更新，即在更新事件发生后，能够自动改变预分频值。当 T2CEN 为 0 时，写入预分频寄存器的值也能直接加载实际应用的预分频寄存器中。

11.3.2. 捕捉比较通道

TIM2CCMRx 寄存器是复用寄存器。
当作为输出比较通道时，TIM2CCMRx 寄存器作为输出配置寄存器，并且第 7 位和第 2 位禁止配置，保持为默认值；下表为 TIM2CCMRx 作为输出配置寄存器时的具体意义：

Bit	7	6	5	4	3	2	1	0
Name	reserved	T2OcxM[2:0]			T2OcxPE	reserved	T2CCxS[1:0]	
Rese	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RW	RW	RO-0	RW	RW

表 11.1 TIM2CCMRx 作为输出配置寄存器

当作为输入捕捉通道时，TIM2CCMRx 寄存器作为输入配置寄存器；下表为 TIM2CCMRx 作为输出配置寄存器时的具体意义：

Bit	7	6	5	4	3	2	1	0
Name	T2IcxF[3:0]				T2ICxPSC[1:0]		T2CCxS[1:0]	
Rese	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

表 10.2 TIM2CCMRx 作为输入配置寄存器

11.3.2.1. 捕捉输入通道

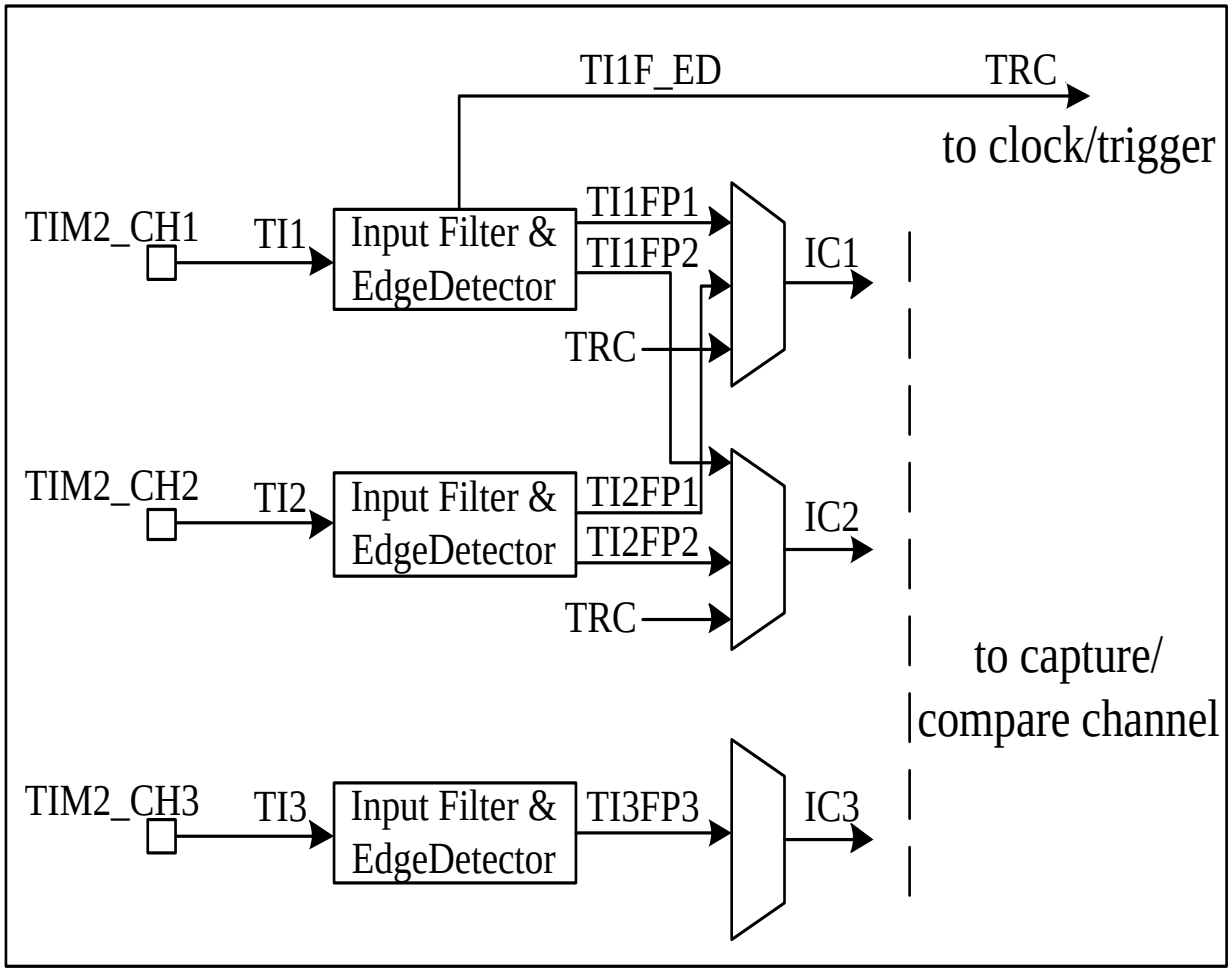


图 11.4 输入通道框图

11.3.2.2. 输出比较通道

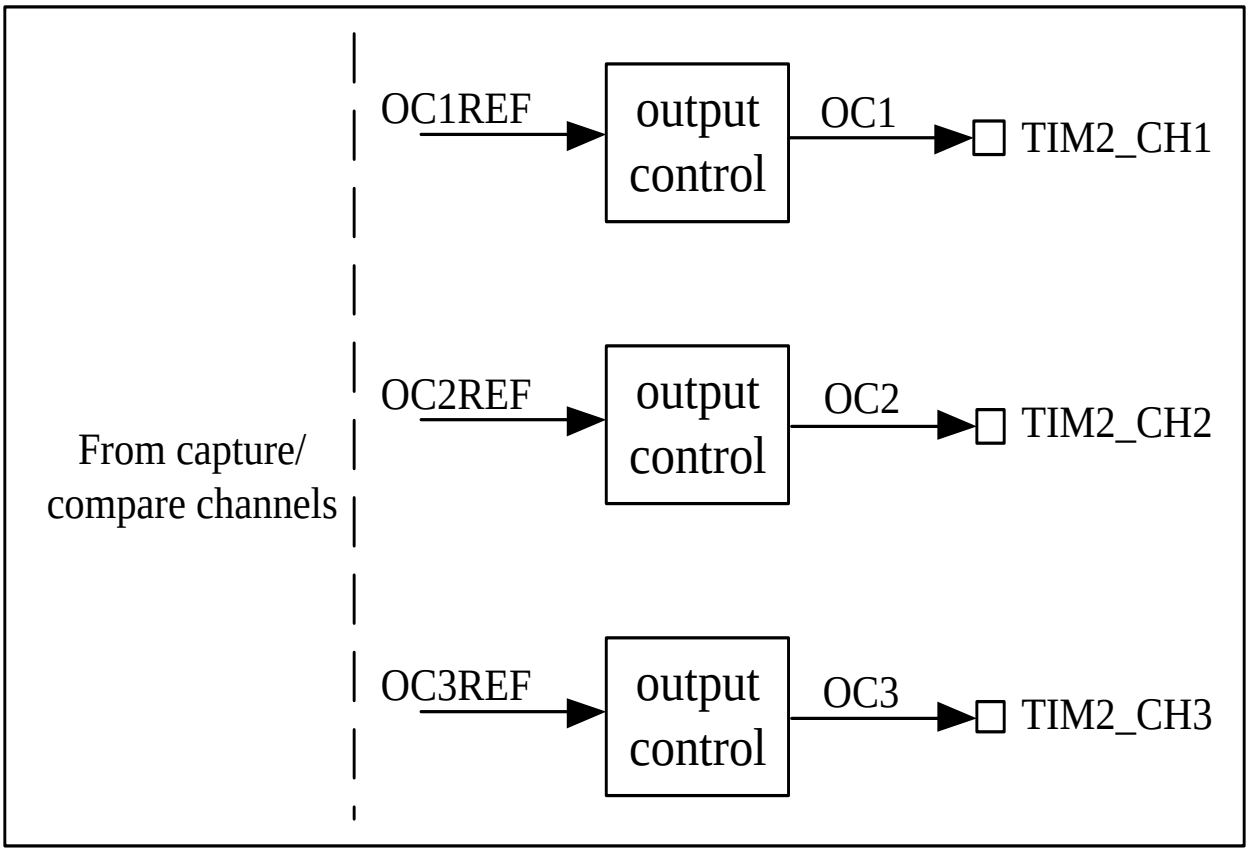


图 11.5 输出通道框图

TIM2 的输出没有死区功能，没有互补输出功能，也没有刹车功能。

11.3.3. TIM2 中断

TIM2 有以下 4 个中断请求源：

- 捕捉/比较 3 中断
- 捕捉/比较 2 中断
- 捕捉/比较 1 中断
- 更新中断

在用这些中断之前需要提前打开 TIM2IER 寄存器中的中断使能位(T2CCxIE 和 T2UIE)。

不同的中断源还可以配置通过 TIM2EGR 寄存器来产生(软件产生中断)。

11.4. 与 TIM2 相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
TIM2CR1	0x30C	T2ARPE	—	—	—	T2OPM	T2URS	T2UDIS	T2CEN	0--- 0000
TIM2IER	0x30D	—	—	—	—	T2CC3IE	T2CC2IE	T2CC1IE	T2UIE	---- 0000
TIM2SR1	0x30E	—	—	—	—	T2CC3IF	T2CC2IF	T2CC1IF	T2UIF	---- 0000
TIM2SR2	0x30F	—	—	—	—	T2CC3OF	T2CC2OF	T2CC1OF	—	---- 000-
TIM2EGR	0x310	—	—	—	—	T2CC3G	T2CC2G	T2CC1G	T2UG	---- 0000
TIM2CCMR1 (output mode)	0x311	—	T2OC1M[2:0]			T2OC1PE	—	T2CC1S[1:0]		-000 0-00
TIM2CCMR1 (input mode)		T2IC1F[3:0]				T2IC1PSC[1:0]		T2CC1S[1:0]		0000 0000
TIM2CCMR2 (output mode)	0x312	—	T2OC2M[2:0]			T2OC2PE	—	T2CC2S[1:0]		-000 0-00
TIM2CCMR2 (input mode)		T2IC2F[3:0]				T2IC2PSC[1:0]		T2CC2S[1:0]		0000 0000
TIM2CCMR3 (output mode)	0x313	—	T2OC2M[2:0]			OC3PE	—	T2CC3S[1:0]		-000 0-00
TIM2_CCMR3 (input mode)		T2IC3F[3:0]				T2IC3PSC[1:0]		T2CC3S[1:0]		0000 0000
TIM2CCER1	0x314	—	—	T2CC2P	T2CC2E	—	—	T2CC1P	T2CC1E	--00 —00
TIM2CCER2	0x315	—	—	—	—	—	—	T2CC3P	T2CC3E	---- --00
TIM2CNTRH	0x316	T2CNT[15:8]								0000 0000
TIM2CNTRL	0x317	T2CNT[7:0]								0000 0000
TIM2PSCR	0x318	—	—	—	—	T2PSC[3:0]				---- 0000
TIM2ARRH	0x319	T2ARR[15:8]								1111 1111
TIM2ARRL	0x31A	T2ARR[7:0]								1111 1111
TIM2CCR1H	0x31B	T2CCR1[15:8]								0000 0000
TIM2CCR1L	0x31C	T2CCR1[7:0]								0000 0000
TIM2CCR2H	0x31D	T2CCR2[15:8]								0000 0000
TIM2CCR2L	0x31E	T2CCR2[7:0]								0000 0000
TIM2CCR3H	0x29E	T2CCR3[15:8]								0000 0000
TIM2CCR3L	0x29F	T2CCR3[7:0]								0000 0000

11.4.1. TIM2CR1，地址 0x30C

Bit	7	6	5	4	3	2	1	0
Name	T2ARPE	reversed			T2OPM	T2URS	T2UDIS	T2CEN
Reset	0	—	—	—	0	0	0	0
Type	RW	RO-0	RO-0	RO-0	RW	RW	RW	RW

7	**T2ARPE**：自动预装载允许位 0：TIM2ARRH/L寄存器没有缓冲，它可以被直接写入； 1：TIM2ARRH/L 寄存器由预装载缓冲器缓冲。
6:4	**保留位**
3	**T2OPM**：单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除T2CEN位)时，计数器停止。
2	**T2URS**：更新请求源 0：如果T2UDIS允许产生更新事件，则下述任一事件产生一个更新中断： 寄存器被更新(计数器上溢/下溢) 1：如果T2UDIS允许产生更新事件，则只有当下列事件发生时才产生更新中断，并UIF置1： 寄存器被更新(计数器上溢/下溢)
1	**T2UDIS**：禁止更新 0：一旦下列事件发生，产生更新(UEV)事件： 计数器溢出/下溢 产生软件更新事件 1：不产生更新事件，影子寄存器(ARR_SHAD、PSC_SHAD、CCRx_SHAD)保持它们的值。
0	**T2CEN**：允许计数器 0：禁止计数器； 1：使能计数器。

11.4.2. TIM2IER，地址 0x30D

Bit	7	6	5	4	3	2	1	0
Name	reserved				T2CC3IE	T2CC2IE	T2CC1IE	T2UIE
Reset	—	—	—	—	0	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	R-W0	R-W0	R-W0	R-W0

7:4	**保留位**
3	**T2CC3IE**：允许捕获/比较3中断 0：禁止捕获/比较3中断； 1：允许捕获/比较3中断。
2	**T2CC2IE**：允许捕获/比较2中断 0：禁止捕获/比较2中断； 1：允许捕获/比较2中断。
1	**T2CC1IE**：允许捕获/比较1中断 0：禁止捕获/比较1中断； 1：允许捕获/比较1中断。
0	**T2UIE**：允许更新中断 0：禁止更新中断； 1：允许更新中断。

11.4.3. TIM2SR1，地址 0x30E

Bit	7	6	5	4	3	2	1	0
Name	reserved				T2CC3IF	T2CC2IF	T2CC1IF	T2UIF
Reset	—	—	—	—	0	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	R-W0	R-W0	R-W0	R-W0

7:4	**保留位**
3	**T2CC3IF**：捕获/比较3中断标记**(写1清0，写0无效)** 参考T2CC1IF描述。
2	**T2CC2IF**：捕获/比较2中断标记**(写1清0，写0无效)** 参考T2CC1IF描述。
1	**T2CC1IF**：捕获/比较1中断标记　如果通道1配置为输出模式：**(写1清0，写0无效)** 如果通道1配置为输出比较模式：当计数器值与比较值匹配时该位由硬件置1；它由软件清0。 0：无匹配发生； 1：TIM2_CNT的值与TIM2CCR1H/L的值匹配。 如果通道1配置为输入捕捉模式：当捕获事件发生时该位由硬件置1，它由软件清0或通过读TIM2CCR1L清0。 0：无输入捕获产生； 1：计数器值已被捕获(拷贝)至TIM2CCR1H/L(在IC1上检测到与所选极性相同的边沿)。
0	**T2UIF**：更新中断标记**(写1清0，写0无效)** 当产生更新事件时该位由硬件置1。它由软件清0。 0：无更新事件产生； 1：更新事件等待响应。

11.4.4. TIM2SR2，地址 0x30F

Bit	7	6	5	4	3	2	1	0
Name	reserved				T2CC3OF	T2CC2OF	T2CC1OF	reserved
Reset	—	—	—	—	0	0	0	—
Type	RO-0	RO-0	RO-0	RO-0	R-W0	R-W0	R-W0	RO-0

7:4	**保留位**
3	**T2CC3OF**：捕获/比较3重复捕获标记**(写1清0，写0无效)** 参见T2CC1OF描述。
2	**T2CC2OF**：捕获/比较2重复捕获标记**(写1清0，写0无效)** 参见T2CC1OF描述。
1	**T2CC1OF**：捕获/比较1重复捕获标记**(写1清0，写0无效)** 仅当相应的通道被配置为输入捕获时，该标记可由硬件置1。写0可清除该位。 0：无重复捕获产生； 1：计数器的值被捕获到TIM2CCR1H/L寄存器时，T2CC1IF的状态已经为1。
0	**保留位**

11.4.5. TIM2EGR，地址 0x310

Bit	7	6	5	4	3	2	1	0
Name	reserved				T2CC3G	T2CC2G	T2CC1G	T2UG
Reset	—	—	—	—	0	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW

7:4	**保留位**
3	**T2CC3G：**产生捕获/比较3事件 参考T2CC1G描述。
2	**T2CC2G：**产生捕获/比较2事件 参考T2CC1G描述。
1	**T2CC1G：**产生捕获/比较1事件 该位由软件置1，用于产生一个捕获/比较事件，由硬件自动清0。 0：无动作； 1：在通道1上产生一个捕获/比较事件。 若通道1配置为输出： 设置T2CC1IF=1，若开启对应的中断，则产生相应的中断。 若通道1配置为输入： 当前的计数器值被捕获至TIM2CCR1H/L寄存器，设置T2CC1IF=1，若开启对应的中断，则产生相应的中断。 若T2CC1IF已经为1，则设置T2CC1OF=1。
0	**T2UG：**产生更新事件 该位由软件置1，由硬件自动清0。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。 注意： 预分频器的计数器也被清0(但是预分频系数不变)。若在T2DIR=0(向上计数)则计数器被清0；若T2DIR=1(向下计数)则计数器初始化为TIM1ARRH/L的值。

11.4.6. TIM2CCMR1，地址 0x311

配置为输出比较模式：

Bit	7	6	5	4	3	2	1	0
Name	reserved	T2OC1M[2:0]			T2OC1PE	reserved	T2CC1S[1:0]	
Reset	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RO-0	RW	RO-0	RW	RW

7	**保留位**
6:4	**T2OC1M[2:0]：**输出比较1模式 该3位定义了输出参考信号OC1REF的动作，而OC1REF决定了OC1的值。OC1REF是高电平有效，而OC1的有效电平取决于T2CC1P位。 000：冻结。输出实际比较值(CCRx_SHAD)与计数器TIM2_CNT间的比较对OC1REF不起作用； 001：匹配时设置通道1的输出为有效电平。当计数器TIM2_CNT的值与实际比较值(CCRx_SHAD)相同时，强制OC1REF为高。 010：匹配时设置通道1的输出为无效电平。当计数器TIM2_CNT的值与实际比较值(CCRx_SHAD)相同时，强制OC1REF为低。 011：翻转。当TIM2_CCR1=TIM2_CNT时，翻转OC1REF的电平。 100：强制为无效电平。强制OC1REF为低。 101：强制为有效电平。强制OC1REF为高。 110：PWM模式1－ 在向上计数时，一旦TIM2_CNT<实际比较值(CCRx_SHAD)时OC1REF为有效电平，否则为无效电平；在向下计数时，一旦TIM2_CNT>实际比较值(CCRx_SHAD)时OC1REF为无效电平， 否则为有效电平。 111：PWM模式2－ 在向上计数时，一旦TIM2_CNT<实际比较值(CCRx_SHAD)时OC1REF为无效电平，否则为有效电平；在向下计数时，一旦TIM2_CNT>实际比较值(CCRx_SHAD)时OC1REF为有效电平，否则为无效电平。 注1：在PWM模式1或PWM模式2中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到PWM模式时，OC1REF电平才改变。
3	**T2OC1PE：**输出比较1预装载使能 0：禁止TIM2CCR1H/L寄存器的预装载功能，可随时写入T2CCR1预加载寄存器，并且新写入的数值立即起作用。 1：开启TIM2CCR1H/L寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM2CCR1H/L的预装载值在更新事件到来时被加载至当前寄存器中。
2	**保留位**
1:0	**T2CC1S[1:0]：**捕获/比较1 选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道1被配置为输出； 01：通道1被配置为输入，IC1映射在TI1FP1上； 10：通道1被配置为输入，IC1映射在TI2FP1上； 11：预留 注：T2CC1S仅在通道关闭时(TIM2CCER1寄存器的T2CC1E=0)才是可写的。

配置为输入捕捉模式：

Name	T2IC1F[3:0]				T2IC1PSC[1:0]		T2CC1S[1:0]	
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

7:4	**T2IC1F[3:0]：**输入捕获1滤波器 这几位定义了TI1输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，只有发生了N个事件后输出的跳变才被认为有效。 0000：无滤波器，fSAMPLING=fMASTER　　1000：采样频率fSAMPLING=fMASTER/8，N=6 0001：采样频率fSAMPLING=fMASTER，N=2　　1001：采样频率fSAMPLING=fMASTER/8，N=8 0010：采样频率fSAMPLING=fMASTER，N=4　　1010：采样频率fSAMPLING=fMASTER/16，N=5 0011：采样频率fSAMPLING=fMASTER，N=8　　1011：采样频率fSAMPLING=fMASTER/16，N=6 0100：采样频率fSAMPLING=fMASTER/2，N=6　　1100：采样频率fSAMPLING=fMASTER/16，N=8 0101：采样频率fSAMPLING=fMASTER/2，N=8　　1101：采样频率fSAMPLING=fMASTER/32，N=5 0110：采样频率fSAMPLING=fMASTER/4，N=6　　1110：采样频率fSAMPLING=fMASTER/32，N=6 0111：采样频率fSAMPLING=fMASTER/4，N=8　　1111：采样频率fSAMPLING=fMASTER/32，N=8
3:2	**T2IC1PSC[1:0]：**输入/捕获1预分频器 这2位定义了通道1输入(IC1)的预分频系数。 一旦T2CC1E=0(TIM2CCER1寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每2个事件触发一次捕获； 10：每4个事件触发一次捕获； 11：每8个事件触发一次捕获。
1:0	**T2CC1S[1:0]：**捕获/比较1 选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道1被配置为输出； 01：通道1被配置为输入，IC1映射在TI1FP1上； 10：通道1被配置为输入，IC1映射在TI2FP1上； 11：预留 注：T2CC1S仅在通道关闭时(TIM2CCER1寄存器的T2CC1E=0)才是可写的。

11.4.7. TIM2CCMR2，地址 0x312

配置为输出比较模式：

Bit	7	6	5	4	3	2	1	0
Name	reserved	T2OC2M[2:0]			T2OC2PE	reserved	T2CC2S[1:0]	
Reset	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RW	RW	RO-0	RW	RW
7	**保留位**							
6:4	**T2OC2M[2:0]**：输出比较2模式							
3	**T2OC2PE**：输出比较2预装载使能							
2	**保留位**							
1:0	**T2CC2S[1:0]**：捕获/比较2选择。 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道2被配置为输出； 01：通道2被配置为输入，IC2映射在TI2FP2上； 10：通道2被配置为输入，IC2映射在TI1FP2上； 11：预留 注：T2CC2S仅在通道关闭时(TIM2CCER1寄存器的T2CC2E=0，T2CC2NE=0且已被更新)才是可写的。							

配置为输入捕捉模式：

Name	T2IC2F[3:0]				T2IC2PSC[1:0]		T2CC2S[1:0]	
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO
7:4	**T2IC2F[3:0]**：输入捕获2滤波器							
3:2	**T2IC2PSC[1:0]**：输入/捕获2预分频器							
1:0	**T2CC2S[1:0]**：捕获/比较2选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道2被配置为输出； 01：通道2被配置为输入，IC2映射在TI2FP2上； 10：通道2被配置为输入，IC2映射在TI1FP2上； 11：预留 注：T2CC2S仅在通道关闭时(TIM2CCER1寄存器的T2CC2E=0，T2CC2NE=0且已被更新)才是可写的。							

11.4.8. TIM2CCMR3，地址 0x313

配置为输出比较模式：

Bit	7	6	5	4	3	2	1	0
Name	reserved	T2OC3M[2:0]			T2OC3PE	reserved	T2CC3S[1:0]	
Reset	—	0	0	0	0	—	0	0
Type	RO-0	RW	RW	RW	RW	RO-0	RW	RW
7	**保留位**							
6:4	**T2OC3M[2:0]**：输出比较3模式							
3	**T2OC3PE**：输出比较3预装载使能							
2	**保留位**							
1:0	**T2CC3S[1:0]**：捕获/比较3选择。 该位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道3被配置为输出； 01：通道3被配置为输入，IC3映射在TI3FP3上； 10：通道3被配置为输入，IC3映射在TI4FP3上； 11：预留 注：T2CC3S仅在通道关闭时(TIM2CCER2寄存器的T2CC3E=0，T2CC3NE=0且已被更新)才是可写的。							

配置为输入捕捉模式：

Name	T2IC3F[3:0]				T2IC3PSC[1:0]		T2CC3S[1:0]	
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO
7:4	**T2IC3F[3:0]**：输入捕获3滤波器							
3:2	**T2IC3PSC[1:0]**：输入/捕获3预分频器							
1:0	**T2CC3S[1:0]**：捕获/比较3选择。 这2位定义通道的方向(输入/输出)，及输入脚的选择： 00：通道3被配置为输出； 01：通道3被配置为输入，IC3映射在TI3FP3上； 10：通道3被配置为输入，IC3映射在TI4FP3上； 11：预留 注：T2CC3S仅在通道关闭时(TIM2CCER2寄存器的T2CC3E=0，T2CC3NE=0且已被更新)才是可写的。							

11.4.9. TIM2CCER1，地址 0x314

Bit	7	6	5	4	3	2	1	0
Name	reserved		T2CC2P	T2CC2E	reserved		T2CC1P	T2CC1E
Reset	—	—	0	0	—	—	0	0
Type	RO-0	RO-0	RW	RW	RO-0	RO-0	RW	RW
7:6	**保留位**							
5	**T2CC2P**：输入捕获/比较2输出极性。参考T2CC1P的描述。							
4	**T2CC2E**：输入捕获/比较2输出使能。参考T2CC1E的描述。							
3:2	**保留位**							
1	**T2CC1P**：输入捕获/比较1输出极性 通道1配置为输出： 0：OC1高电平有效； 1：OC1低电平有效。 通道1配置为触发输入： 0：触发发生在TI1F的高电平或上升沿； 1：触发发生在TI1F的低电平或下降沿。 通道1配置为捕捉输入： 0：捕捉发生在TI1F的高电平或上升沿； 1：捕捉发生在TI1F的低电平或下降沿。							
0	**T2CC1E**：输入捕获/比较1输出使能 CC1通道配置为输出： 0： 关闭－ OC1禁止输出。 1： 开启－ OC1信号输出到对应的输出引脚。 CC1通道配置为输入： 该位决定了计数器的值是否能捕获入TIM2CCR1寄存器。 0：捕获禁止； 1：捕获使能。							

11.4.10. TIM2CCER2，地址 0x315

Bit	7	6	5	4	3	2	1	0
Name	reserved						T2CC3P	T2CC3E
Reset	—	—	—	—	—	—	0	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW
7:2	**保留位**							
1	**T2CC3P**：输入捕获/比较3输出极性。参考T2CC1P的描述。							
0	**T2CC3E**：输入捕获/比较3输出使能。参考T2CC1E 的描述。							

11.4.11. TIM2CNTRH，地址 0x316

Bit	7	6	5	4	3	2	1	0
Name	T2CNT[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2CNT[15:8]**：计数器的高8位值							

11.4.12. TIM2CNTRL，地址 0x317

Bit	7	6	5	4	3	2	1	0
Name	T2CNT[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2CNT[7:0]**：计数器的低8位值							

11.4.13. TIM2PSCR，地址 0x318

Bit	7	6	5	4	3	2	1	0
Name	reserved				T2PSC[3:0]			
Reset	—	—	—	—	0	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	RW	RW	RW	RW
7:4	**保留位**							
3:0	**T2PSC[3:0]**：预分频器的值 预分频器对输入的CK_PSC时钟进行分频。 计数器的时钟频率f_{CK_CNT}等于$f_{CK_PSC}/2^{(PSCR[3:0])}$。 PSCR为实际装入预分频器影子寄存器的值(包括由于清除TIM2EGR寄存器的UG位产生的计数器清除事件)。这意味着如要新的预分频值生效，必须产生更新事件或者T2CEN=0。							

11.4.14. TIM2ARRH，地址 0x319

Bit	7	6	5	4	3	2	1	0
Name	T2ARR[15:8]							
Reset	1	1	1	1	1	1	1	1
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2ARR[15:8]:** 自动重装载的高8位值 T2ARR为将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。							

11.4.15. TIM2ARRL，地址 0x31A

Bit	7	6	5	4	3	2	1	0
Name	T2ARR[7:0]							
Reset	1	1	1	1	1	1	1	1
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2ARR[7：0]:** 自动重装载的低8位值 T2ARR为将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。							

11.4.16. TIM2CCR1H，地址 0x31B

Bit	7	6	5	4	3	2	1	0
Name	T2CCR1[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2CCR1[15:8]:** 捕获/比较1的高8位值 若通道1配置为输出(TIM2CCMR1的T2CC1S=00)： T2CCR1H/L为装入当前捕获/比较1寄存器的值(预装载值)。 如果在TIM2CCMR1寄存器(T2OC1PE位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较1寄存器中。 当前捕获/比较寄存器的值同计数器TIM2_CNT的值相比较，并在OC1端口上产生输出信号。 若CC1通道配置为输入： T2CCR1H/L包含了上一次输入捕获1事件(IC1)发生时的计数器值（此时该寄存器为只读）。							

11.4.17. TIM2CCR1L，地址 0x31C

Bit	7	6	5	4	3	2	1	0
Name	T2CCR1[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2CCR1[7:0]**: 捕获/比较1的低8位值							

11.4.18. TIM2CCR2H，地址 0x31D

Bit	7	6	5	4	3	2	1	0
Name	T2CCR2[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2CCR2[15:8]:** 捕获/比较2的高8位值 若通道2配置为输出(TIM2CCMR2的T2CC2S=00)： T2CCR2H/L为装入当前捕获/比较2寄存器的值(预装载值)。 如果在TIM2CCMR2寄存器(T2OC2PE位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较2寄存器中。 当前捕获/比较寄存器的值同计数器TIM2_CNT的值相比较，并在OC2端口上产生输出信号。 若通道2配置为输入： T2CCR2H/L包含了上一次输入捕获2事件(IC2)发生时的计数器值（此时该寄存器为只读）。							

11.4.19. TIM2CCR2L，地址 0x31E

Bit	7	6	5	4	3	2	1	0
Name	T2CCR2[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2CCR2[7:0]**: 捕获/比较2的低8位值							

11.4.20. TIM2CCR3H，地址 0x29E

Bit	7	6	5	4	3	2	1	0
Name	T2CCR3[15:8]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2CCR3[15:8]:** 捕获/比较3的高8位值 若通道3配置为输出(TIM2CCMR3的T2CC3S=00)： T2CCR3H/L为装入当前捕获/比较3寄存器的值(预装载值)。 如果在TIM2CCMR3寄存器(T2OC3PE位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较3寄存器中。 当前捕获/比较寄存器的值同计数器TIM2_CNT的值相比较，并在OC3端口上产生输出信号。 若通道3配置为输入： T2CCR3H/L包含了上一次输入捕获3事件(IC3)发生时的计数器值（此时该寄存器为只读）。							

11.4.21. TIM2CCR3L，地址 0x29F

Bit	7	6	5	4	3	2	1	0
Name	T2CCR3[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T2CCR3[7:0]**: 捕获/比较3的低8位值							

12. 基本定时器 TIM4

12.1. 特性

- 8bit 自动重载向上计数器
- 计数时钟可编程预分频
- 中断
  - ●计数器溢出

12.2. 原理框图

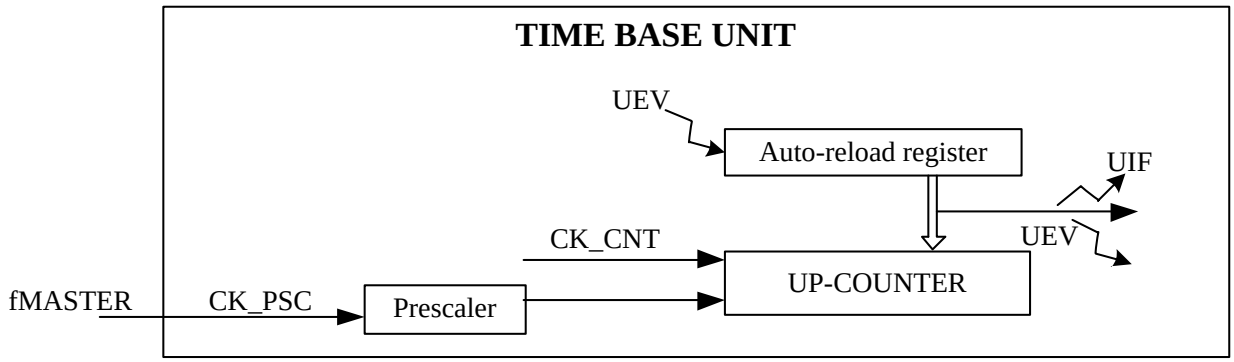


图 12.1 TIM4 原理框图

12.3. TIM4 时钟源

TIM4 有 4 种时钟源可选，由寄存器位 T4CKS 设置。在 TIM4 的被使能（PCKEN.TIM4EN=1）的情况下，所选择的时钟源被自动使能。

注意：

1. 如果要选择 LP 晶体时钟，系统时钟配置寄存器位 FOSC 必须选择 LP 模式，否则对应的时钟源将不被使能；

2. 同理，如果要选择 XT 晶体时钟，系统时钟配置寄存器位 FOSC 必须选择 XT 模式，否则对应的时钟源将不被使能；

SLEEP 模式下，如果 SYSON 为 1，且 TIM4EN=1，则所选择的时钟源将保持振荡，TIM4 将继续工作；否则，所选的时钟源取决于其他模块的设置情况。

12.4. 预分频器

计数时钟可以进行 3bit 的时钟预分频：

$f_{CK\_CNT} = f_{CK\_PSC}/2^{(PSCR[2:0])}$

预分频支持分频自动更新，即在更新事件发生后，能够自动改变预分频值。当 T4CEN 为 0 时，写入预分频寄存器的值也能直接加载实际应用的预分频寄存器中。

12.5. TIM4 中断

TIM4 只有一个中断请求源：

- 更新中断(计数器上溢或计数器初始化)

在用这些中断之前需要提前打开 TIM4IER 寄存器中的中断使能位(T4UIE)。

不同的中断源还可以配置通过 TIM4EGR 寄存器来产生(软件产生中断 T4UG)。

12.6. TIM4 寄存器表

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
TIM4CR1	0x111	T4ARPE	—	T4CKS[1:0]		T4OPM	T4URS	T4UDIS	T4CEN	0-00 0000
TIM4IER	0x112	—	—	—	—	—	—	—	T4UIE	---- ---0
TIM4SR	0x113	—	—	—	—	—	—	—	T4UIF	---- ---0
TIM4EGR	0x114	—	—	—	—	—	—	—	T4UG	---- ---0
TIM4CNTR	0x115	T4CNT[7:0]								0000 0000
TIM4PSCR	0x116	—	—	—	—	—	T4PSC[2:0]			---- -000
TIM4ARR	0x117	T4ARR[7:0]								1111 1111

12.6.1. TIM4CR1，地址 0x111

Bit	7	6	5	4	3	2	1	0
Name	T4ARPE	reserved	T4CKS[1:0]		T4OPM	T4URS	T4UDIS	T4CEN
Reset	0	—	0	0	0	0	0	0
Type	RW	RO-0	RW	RW	RW	RW	RW	RW

7	**T4ARPE**：自动预装载允许位 0：TIM4ARRH/L寄存器没有缓冲，它可以被直接写入； 1：TIM4ARRH/L 寄存器由预装载缓冲器缓冲。
6	**保留位**
5:4	**T4CKS**：TIM4时钟选择位 00：系统时钟/主时钟 01：内部快时钟HIRC 10：LP时钟，只有当FOSC选择LP模式时才有意义 11：XT时钟，只有当FOSC选择XT模式时才有意义
3	**T4OPM**：单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件(清除T4CEN位)时，计数器停止。
2	**T4URS**：更新请求源 0：如果T4UDIS允许产生更新事件，则下述任一事件产生一个更新中断： 寄存器被更新(计数器上溢) 软件设置T4UG位 1：如果T4UDIS允许产生更新事件，则只有当下列事件发生时才产生更新中断，并T4UIF置1： 寄存器被更新(计数器上溢)
1	**T4UDIS**：禁止更新 0：一旦下列事件发生，产生更新(UEV)事件： 计数器溢出 产生软件更新事件 1：不产生更新事件，影子寄存器(ARR_SHAD、PSC_SHAD)保持它们的值。如果设置了T4UG位，则计数器和预分频器被重新初始化。
0	**T4CEN**：允许计数器 0：禁止计数器； 1：使能计数器。

12.6.2. TIM4IER，地址 0x112

Bit	7	6	5	4	3	2	1	0
Name	reserved							T4UIE
Reset	—	—	—	—	—	—	—	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW

7:1	**保留位**
0	**T4UIE**：允许更新中断 0：禁止更新中断； 1：允许更新中断。

12.6.3. TIM4SR，地址 0x113

Bit	7	6	5	4	3	2	1	0
Name	reserved							T4UIF
Reset	—	—	—	—	—	—	—	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW-0
7:1	**保留位**							
0	**T4UIF**：更新中断标记 当产生更新事件时该位由硬件置1。它由软件清0。 0：无更新事件产生； 1：更新事件等待响应。							

12.6.4. TIM4EGR，地址 0x114

Bit	7	6	5	4	3	2	1	0
Name	reserved							T4UG
Reset	—	—	—	—	—	—	—	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW
7:1	**保留位**							
0	**T4UG**：产生更新事件 该位由软件置1，由硬件自动清0。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。							

12.6.5. TIM4CNTR，地址 0x115

Bit	7	6	5	4	3	2	1	0
Name	T4CNT[7:0]							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T4CNT[7:0]**：计数器的8位值							

12.6.6. TIM4PSCR，地址 0x116

Bit	7	6	5	4	3	2	1	0
Name	reserved					T4PSC[2:0]		
Reset	—	—	—	—	—	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW
7:3	**保留位**							
2:0	**T4PSC[2:0]**：预分频器的值 预分频器对输入的CK_PSC时钟进行分频。 计数器的时钟频率f_{CK_CNT}等于$f_{CK_PSC}/2^{(PSCR[2:0])}$。 PSCR为实际装入预分频器影子寄存器的值(包括由于清除TIMxEGR寄存器的T4UG位产生的计数器清除事件)。这意味着如要新的预分频值生效，必须产生更新事件或者T4CEN=0。							

12.6.7. TIM4ARR，地址 0x117

Bit	7	6	5	4	3	2	1	0
Name	T4ARR[7:0]							
Reset	1	1	1	1	1	1	1	1
Type	RW	RW	RW	RW	RW	RW	RW	RW
7:0	**T4ARR[7：0]:** 自动重装载的8位值 T4ARR为将要装载入实际的自动重装载寄存器的值。 当自动重装载的值为空时，计数器不工作。							

13. SPI 接口

13.1. 功能特性

- 3 线全双工同步传输
- 2 线半双工同步传输，或单向传输
- 主机模式或从机模式操作
- nss pin 软件或硬件管理
- 可编程的同步时钟极性和相位控制
- 可编程的 LSB first 或 MSB first
- 配置模式错误和 overrun 标志
- 硬件 CRC 校验支持
- Wakeup 唤醒支持

13.2. 功能描述

13.2.1. 一般描述

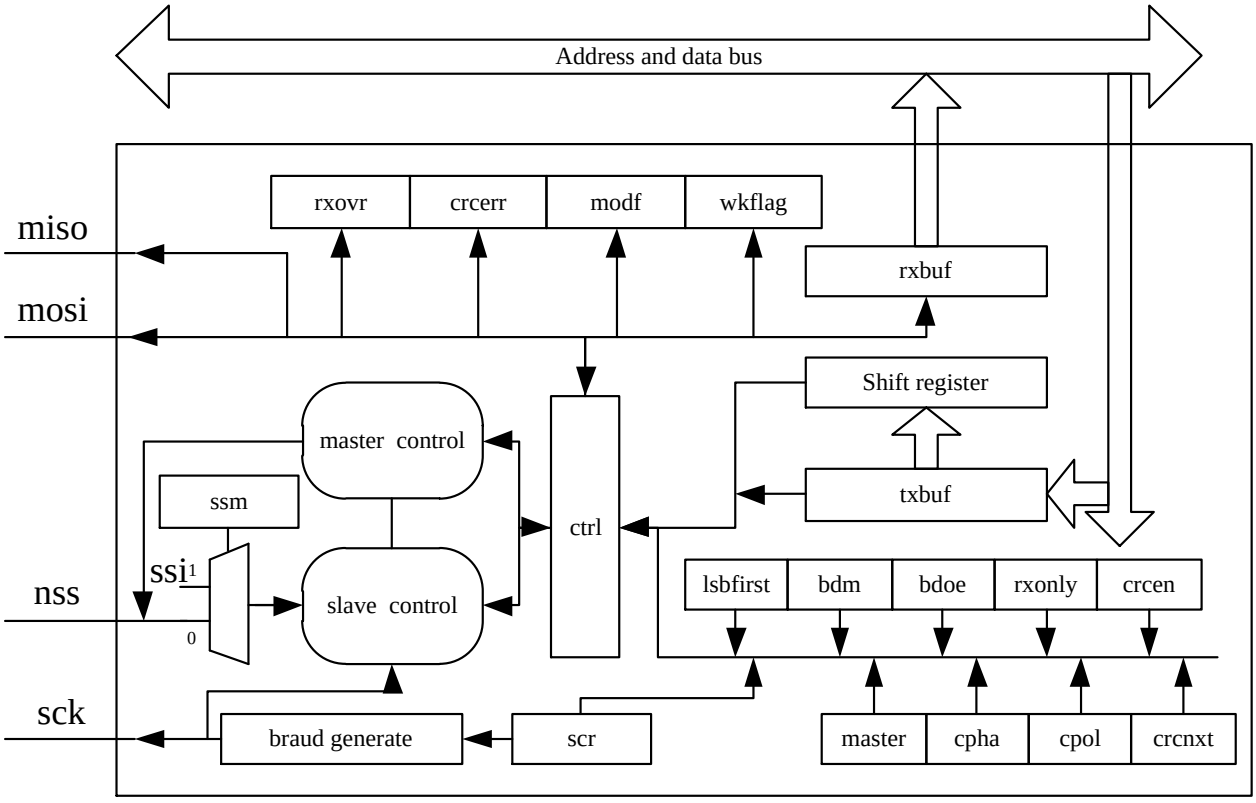


图 13.1 SPI 结构框图

SPI 接口总共有四个引脚：

- NSS:从机选择引脚低电平有效，在 NSS 引脚采用硬件管理用作输入时，如果 NSS 引脚为零则表示选择该从机，如果该接口配置为主机模式，则会产生 MODF 标志位，表示配置错误，这时该接口自动进入从机模式，这种特性用于兼容多主机通信；
- MOSI:主机数据输出/从机数据输入；
- MISO:主机数据输出/从机数据输入；
- SCK:主机串行时钟输出/从机串行时钟输入；

SPI 接口引脚通常的接法是采用三线全双工和四线全双工，如下图所示的引脚接法，NSS 引脚可以根据 NSSM 的值选择作为输入，或者是输出；当 NSS 引脚用作输入时，输入的值取决于 SSM 的值，当 SSM 置 1 时，NSS 引脚的值无效，SSI 的值会被传输到通信控制单元。SPI 模块支持双线半双工模式，主机模式时采用的数据通信引脚是 MOSI，从机模式时采用的数据通信引脚是 MISO。

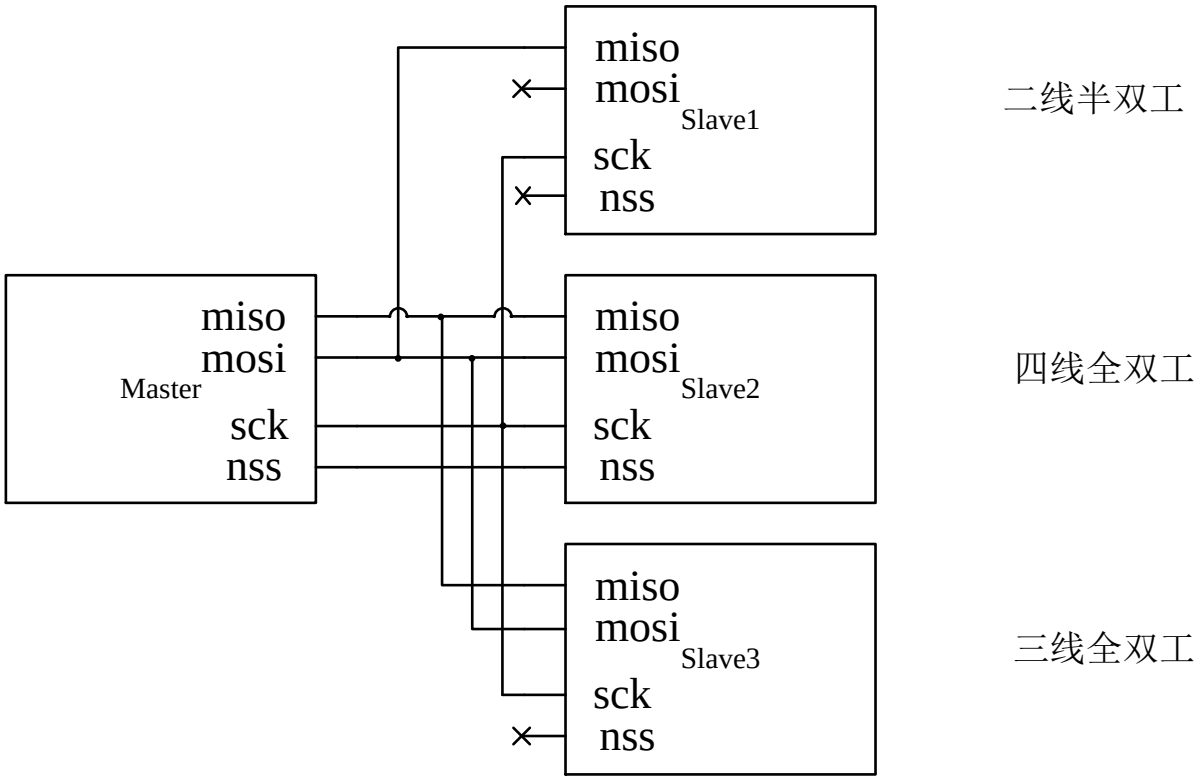


图 13.2 SPI 模块连接示意图

SPI 通信的时钟相位和极性设置如下图所示，时钟和相位的控制总共有四种情况，可以通过 CPHA 和 CPOL 设定相应配置，其中 CPOL 是控制模块空闲时的 SCK 的电平，当 CPOL 为 1 时，SCK 空闲时的电平为高电平，相反为低电平，图中所示箭头的位置是接收数据采样点；当使用了 NSS 引脚时，NSS 为低电平时才会接收数据；发送方串行数据的发送格式可以根据 LSBFIRST 来控制，默认情况下是先发送的八位数据的高位，最后发送的是八位数据的低位。

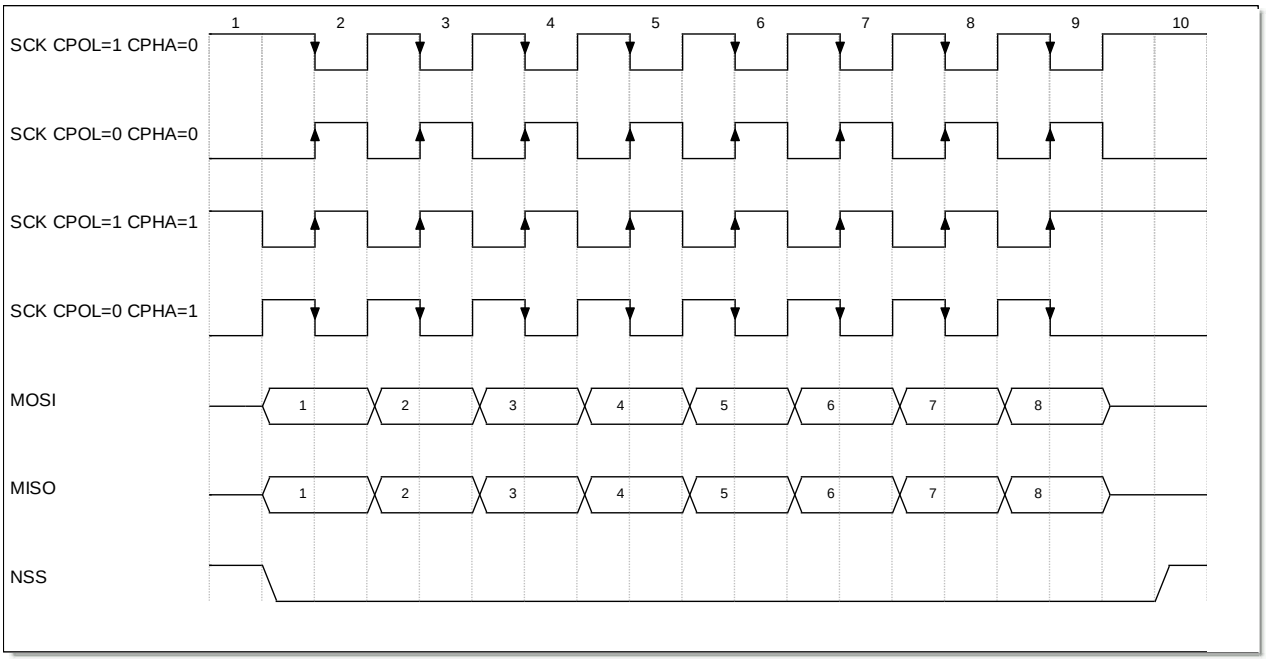


图 13.3 时钟极性和相位时序图

CRC 校验模块用来增强数据传输的可靠性，计算模块初始化的值采用的是零，默认多项式 CRCPOL 是 0x07，每次 CRCEN 从零到置位的时候都会对 CRC 模块进行初始化（该初始化不会影响 CRCPOL 的值），模块内部中间运算的值会丢失（RXCRC 和 TXCRC 的值会被置为零）。当 CRCEN 使能的时候，每次正常写入到 TXBUF 的值都会被送到 CRC 模块用来生成 TXCRC 的值，同样的在接收数据时每次正常写入到 RXBUF 的值也会被送到 CRC 模块，用来生产 RXCRC 的值；当需要传输 CRC 字节的时候，可以置位 CRCNXT，在正常数据传输完成后，下一次传输就会自动把 TXCRC 的值写入到 TXBUF(本次写入到 TXBUF 中的值不会送到 CRC 模块进行计算)，同时 CRCNXT 的值自动清零，在传输最后的 CRC 检验码的同时，也会接收对方的 CRC 检验码（本次接收到的数据不会写入 RXBUF），在接收完成时会比较 RXCRC 与接收到的校验码值，如果不匹配就会产生 CRCERR 标志位。

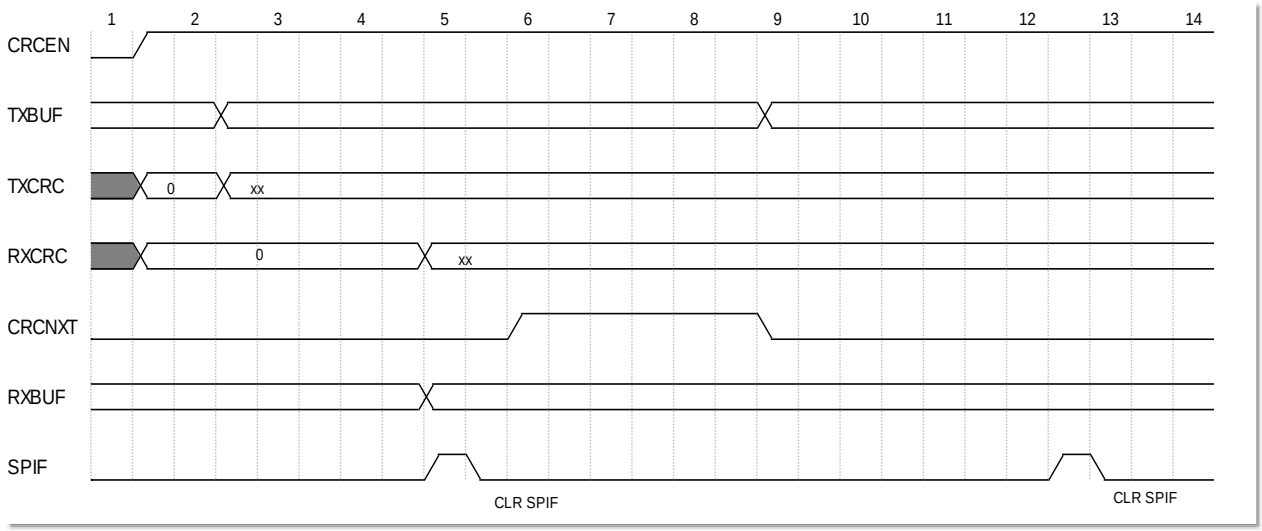


图 13.4 CRC 模块的工作时序图

13.2.2. 配置 SPI

主机模式的 SCK 是由主机产生的，从机的 SCK 是从机输入的，下面是 SPI 配置流程:

1. 根据主从机的需求配置 SPICFG 寄存器中的 MSTEN 位，置 0 是用作从机，置 1 是用作主机
2. 如果用作主机需要配置 SCR 寄存器，来配置通信的速率，通信的比特率=Fmater/(2*(SCR+1)), 该位对从机没有作用，只是在要求从机接收速率较快时，要提高 Fmaster 的频率，以便从机模块可以采样到 SCK 的上升沿或下降沿
3. 配置 NSSM 来设定如何使用 NSS 引脚，如果是用作四线主机模式则 NSS 引脚应配置为输出模式；如果用作从机模式，则可以配置成输入模式。如果只是用三线通信则可以设置禁用 NSS 引脚；另外 NSS 引脚在配置为输入模式时，可以置位 SSM 来启用软件管理 NSS 引脚的输入值，屏蔽掉实际的 NSS 引脚的值
4. 配置 SPICFG 中的 CPOL 和 CPHA 来配置 SCK 的相位和极性
5. 配置 SPICTRL2 中的 LSBFIRST 来设置数据的传输格式
6. 配置 CRCPOL 寄存器和 CRCEN，使能 CRC 校验
7. 置位 SPICTRL2 中的 RXONLY 只允许接收或者置位 BDM 来启用半双工通信
8. 置位 SPICFG 中的 SPIEN 来启用 SPI 通信接口，这时相应的 GPIO 接口会用作 SPI 通信接口，同时 SPIEN 从低电平到高电平的变化会导致清零 RXOVER，CRCERR，MODF，SPIF，WCF 标志位，置位 TXBMT，RXBMT 标志位
9. 如果使用中断模式进行通信则需配置 SPIIER 寄存器来使能相应的中断

13.2.3. 数据处理流程

数据通信的流程大致分为阻塞模式通信和非阻塞式模式通信，大致的处理方式是一样的，只是非阻模式是在中断中进行的：

1. 阻塞通信时向 DATA 寄存器写入数据后需要查询 TXBMT，在查询到 TXBMT 为 1 时可以写入下一个数据；在 TXE 中断使能允许的时候，TXBMT 置 1 会直接进入中断
2. 阻塞模式接收数据时，需要一直查询 RXMBT，在查询到该位为 1 时，则可以读取 DATA 寄存器的值；在 RXNE 中断允许时，RXMBT 置 1 时则会进入中断
3. 阻塞模式接收的过程中需要查询 RXOVRN 和 CRCERR 位，在查询到相应的位置 1 后则需要写零清理相应的错误标志位；在 RXERR 中断使能时，发生相应的错误标志位会直接进入中断
4. 非阻塞模式中，进入中断后查询一次状态信息，然后根据相应的状态信息处理发送接收流程，处理完成以后退出中断子程序，继续处理其他的事情

无论是阻塞模式还是非阻塞模式，通信模块中的相关标识位变化如下图所示，在数据写入到发送数据寄存器后，TXBMT 从 1 变为 0，然后数据寄存器中的数据会传送到内部的移位寄存器，移位寄存器标志位从 1 变为 0，直到数据从移位寄存器中完全移出后变为 1；在发送的过程中，BUSY 状态位一直为 1；发送完成标志位在当前字节传输完成后会变为 1，同时 RXBMT 的值会从 1 变为 0。

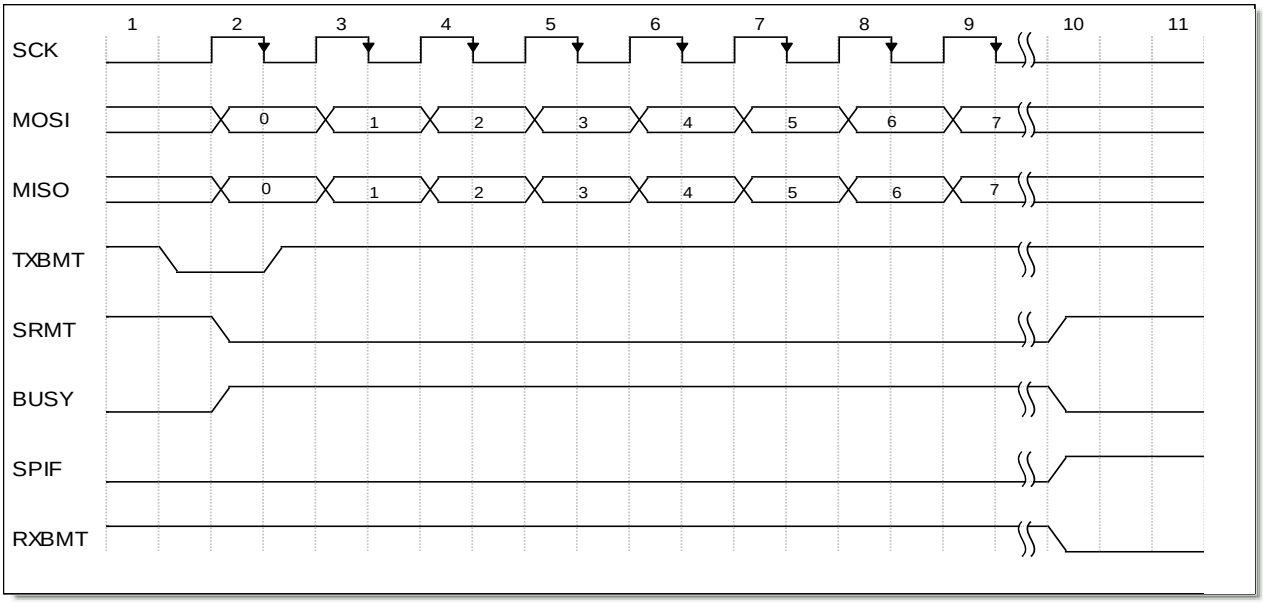


图 13.5 数据处理时序图

13.2.4. 睡眠模式唤醒

当系统进入睡眠模式，外设时钟存在时，SPI 模块有能力唤醒 MCU；如图所示，SPI 模块作为从机模块使用，开启了 WAKUP 中断使能，从机在接收到第一比特的数据时就会产生 WAKEUP 中断信号，在中断控制模块中如果使能了外设接口中断，就会直接唤醒 MCU；

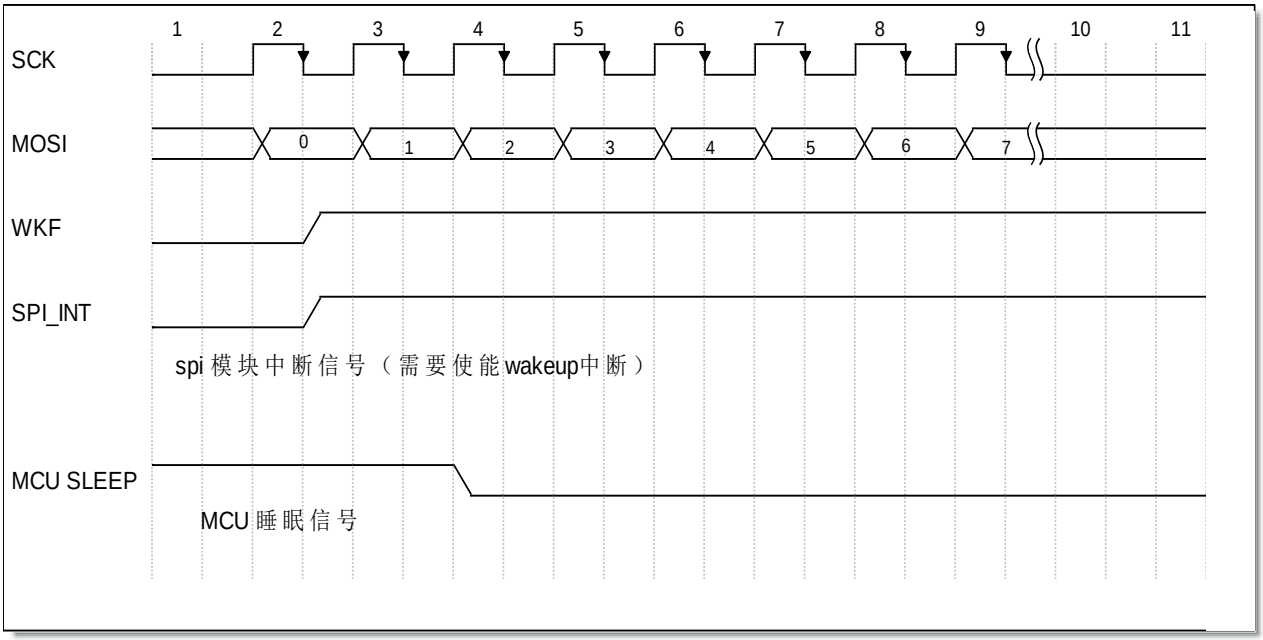


图 13.6 睡眠唤醒时序图

13.2.5. CRC 处理流程

阻塞模式传输数据时，在传输完成最后一个数据时，查询 TXBMT 的状态，在 TXBMT 位 1 时，则置位 CRCNXT，这时 TXCRC 的值就会自动传输到 DATA 寄存器，然后 CRCNXT 就会自动清零， 查询 CRCNXT，查询到为 0 则清零 SPIF 状态位，然后再查询 SPIF 位，如果状态位置 1，则表示 CRC 校验码发送完成，接着查询 CRCERR 状态位，如果该状态位为 1，表示 CRC 校验码不匹配，写零清零相应的状态位；采用非阻塞模式通信时，TXBMT 为 1 进入中断时，若数据已经发送完整，则置位 CRCNXT，CRC 发送完成后如果产生了 CRCERR，则会直接进入中断，查询相关的标志位，然后写零清零。

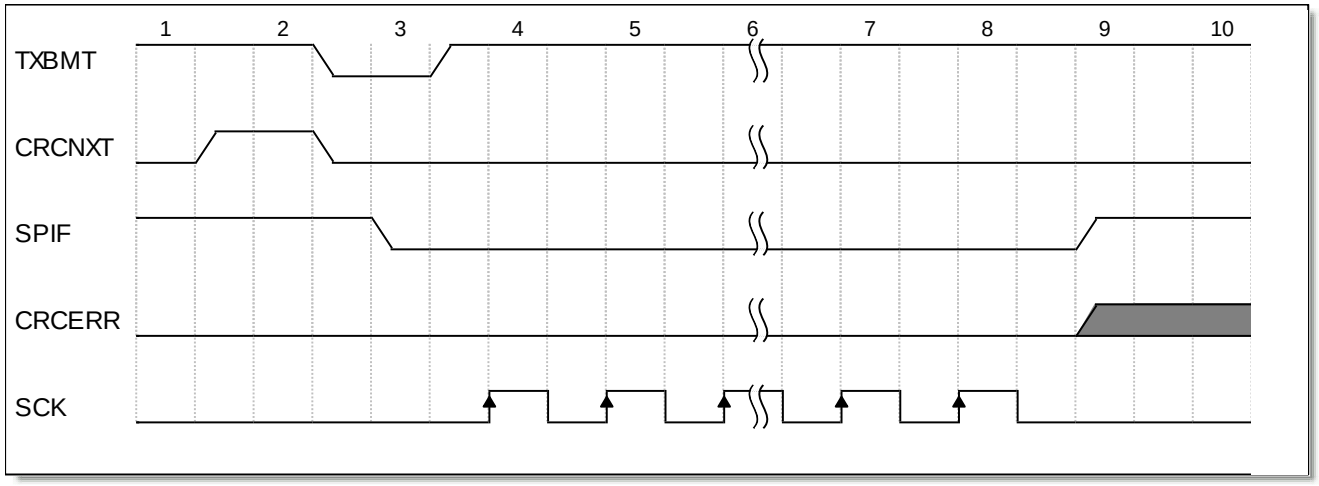


图 13.7 CRC 模块标志位时序图

13.3. 与 SPI 相关寄存器汇总

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	复位值
SPIDATA	015h	DATA[7:0]								0000 0000
SPICTRL	016h	SPIF	WCOL	MODF	RXOVRN	NSSM		TXBMT	SPIEN	0000 0110
SPICFG	017h	BUSY	MSTEN	CPHA	CPOL	SLAS	NSSVAL	SRMT	RXBMT	0000 0000
SPISCR	018h	SCR[7:0]								0000 0000
SPICRCPOL	019h	CRCPOL[7:0]								0000 0111
SPIRXCRC	01Ah	RXCRC[7:0]								0000 0000
SPITXCRC	01Bh	TXCRC[7:0]								0000 0000
SPIIER	01Ch	—				WAKUP	RXERR	RXNE	TXE	---- 0000
SPICTRL2	01Dh	BDM	BDOE	RXONLY	SSI	SSM	CRCNXT	CRCEN	LSBFIRST	0000 0000
SPISTAT	01Eh	—	**SMODF**	**SRXOVRN**	**SBUSY**	**SRXBMT**	**STXBMT**	WKF	CRCERR	-000 1100

13.3.1. SPIDATA 寄存器，地址 0x015

Bit	7:0
Name	DATA
Reset	0x00
Type	RW

Bit	Name	Function
7:0	DATA	数据发送/接收寄存器（BUF）

13.3.2. SPICTRL 寄存器，地址 0x016

Bit	7	6	5	4	3:2	1	0
Name	SPIF	WCOL	MODF	RXOVRN	NSSM	TXBMT	SPIEN
Reset	0x0	0x0	0x0	0x0	0x01	0x1	0x0
Type	RW	RW	RW	RW	RW	RO	RW

Bit	Name	Function
7	SPIF	传输完成标志 0：表示没有传输完成或者已经清零 1：传输完成标志位，写零清理，写 1 无效
6	WCOL	BUF 写入失败标识， 0：BUF 写入正常 1：BUF 为非空时，进行写入会置位该位，写零清理，写 1 无效
5	MODF	工作模式错误标识, 0：工作模式正常 1：当 SPI 配置为主机模式，并且 NSS 用作输入引脚时，若 NSS 引进为低电平就会产生置位该位 写零清零，写 1 无效
4	RXOVRN	接收溢出标志 0：接收正常 1：接收溢出，写零清理，写 1 无效
3:2	NSSM	NSS 引脚模式选择，当用作输出模式时，相应的 IO 脚会被用作 NSS 输出 00:禁用 NSS 引脚 01:NSS 引脚用作输入 1x:NSS 引脚用作输出，输出的值等于 NSSM[0]的值 注：NSS 引脚配置为输入状态时，可以被 SSM 的软件管理模式屏蔽掉
1	TXBMT	发送 BUFF 为空状态 0：发送 BUF 非空 1：发送 BUF 位空
0	SPIEN	SPI 接口使能 0：禁用 SPI 模块 1：启用 SPI 模块，相应的 IO 会被用作 SPI 的功能 **注：在使用完 SPI 以后，禁用 SPI 不会对已经产生的标志位进行复位，只有再次打开 SPI 时会对标志位进行复位操作；SPIEN 从低电平到高电平的变化会导致复位标志位**

13.3.3. SPICFG 寄存器，地址 0x017

Bit	7	6	5	4	3	2	1	0
Name	BUSY	MSTEN	CPHA	CPOL	SLAS	NSSVAL	SRMT	RXBMT
Reset	0x00	0x0	0x0	0x0	0x0	0x1	0x1	0x1
Type	RO	RW	RW	RW	RO	RO	RO	RO

Bit	Name	Function
7	BUSY	SPI BUSY 状态 0：SPI 模块空闲 1：表示 SPI 模块忙绿中
6	MSTEN	MASTER 使能位 0：工作在 SLAVE 模式 1：工作在 MSATER 模式
5	CPHA	SCK 相位选择 0：第一个时钟转换的沿是数据采样点 1：第二个时钟转换的沿是数据采样点
4	CPOL	SCK 极性选择 0：SPI 空闲时，SCK 的时钟是处于低电平状态 1：SPI 空闲时，SCK 的时钟是处于高电平状态
3	SLAS	SLAVE 选择标志 0：该模块未被选中 1：该模块被选中 注：当 NSS 用作输入时，该值可以被 SSM 软件管理，当 SSM 为 1 时，这里的值表示的是 SSI 的值取反
2	NSSVAL	NSS 引脚的输入值状态 注：当 NSS 用作输入时，该值可以被 SSM 软件管理，当 SSM 为 1 时，这里的值表示的是 SSI 的值
1	SRMT	移位寄存器为空状态 0：内部串行移位寄存器非空 1：内部串行移位寄存器为空
0	RXBMT	接受 BUFFER 为空状态 0：表示接收 BUF 非空 1：接收 BUF 为空状态

13.3.4. SPISCR 寄存器，地址 0x018

Bit	7:0
Name	SCR
Reset	0x00
Type	RW

Bit	Name	Function
7:0	SCR	波特率设置寄存器,波特率=Fmaster/(2*(SCR+1)) 注：Fmaster 指的是外设时钟

13.3.5. SPICRCPOL 寄存器，地址 0x019

Bit	7:0
Name	CRCPOL
Reset	0x07
Type	RW

Bit	Name	Function
7:0	CRCPOL	CRC 计算多项式,默认值为 0x07

13.3.6. SPIRXCRC 寄存器，地址 0x01A

Bit	7:0
Name	RXCRC
Reset	0x00
Type	RO

Bit	Name	Function
7:0	RXCRC	接收数据的 CRC 计算结果 注：该寄存器在 CRCEN 发生从零到 1 的变化时会清零

13.3.7. SPITXCRC 寄存器，地址 0x01B

Bit	7:0
Name	TXCRC
Reset	0x00
Type	RO

Bit	Name	Function
7:0	TXCRC	发送数据的 CRC 计算结果 注：该寄存器在 CRCEN 发生从零到一的变化时会清零

13.3.8. SPIIER 寄存器，地址 0x01C

Bit	7:4	3	2	1	0
Name	—	WAKUP	RXERR	RXNE	TXE
Reset	—	0x0	0x0	0x0	0x0
Type	RO-0	RW	RW	RW	RW

Bit	Name	Function
7:4	N/A	保留位，读 0
3	WAKUP	唤醒中断使能 0：禁用唤醒中断 1：启用中断唤醒
2	RXERR	接收错误中断使能，包括 CRC 错误,接收溢出错误，模式错误 0：禁用接收数据出现 CRC 错误，溢出错误和模式错误中断 1：允许接收数据出现 CRC 错误，溢出错误和模式错误中断
1	RXNE	接收 BUF 不为空中断使能 0：禁用接收 BUF 不为空中断 1：允许接收 BUF 不为空中断
0	TXE	发送 BUF 为空中断使 0：禁用发送 BUF 为空中断 1：使能发送 BUF 为空中断

13.3.9. SPICTRL2 寄存器，地址 0x01D

Bit	7	6	5	4	3	2	1	0
Name	BDM	BDOE	RXONLY	SSI	SSM	CRCNXT	CRCEN	LSBFIRST
Reset	0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	BDM	半双工使能 0：禁用半双工模式 1：使能半双工模式
6	BDOE	半双工模式的接收使能 0：半双工模式接收使能 1：半双工模式发送使能
5	RXONLY	全双工模式只允许接收使能 0：全双工模式允许发送和接收 1：全双工模式只允许接收
4	SSI	NSS 输入管脚的值，仅当 SSM 置 1 时有效 0：输入到 NSS 引脚的值是 0 1：输入发哦 NSS 引脚的值是 1
3	SSM	软件 SLAVE 模式管理，使能后 NSS 引脚的值由 SSI 替代 0：禁用 NSS 引脚的软件管理模式 1：启用 NSS 引脚的关键管理模式，如果 NSS 引脚用作输入，则 NSS 引脚实际的值有 SSI 取代
2	CECNXT	置位后在 TXBUFF 为空时会把 TXCRC 的值写入 TXBUFF 0：不传送 TXCRC 的值到 TXBUF 1：等待 TXBMT 为 1 时，传送 TXCRC 的值到 TXBUF，写入完成后改位自动清零
1	CRCEN	CRC 模块计算使能 0：禁用 CRC 校验模块 1：启用 CRC 校验模块
0	LSBFIRST	发送低比特位使能 0：高比特位优先发送 1：低比特位优先发送

13.3.10. SPISTAT 寄存器，地址 0x01E

Bit	7	6	5	4	3	2	1	0
Name	—	SMODF	SRXOVRN	SBUSY	SRXBMT	STXBMT	WKF	CRCERR
Reset	—	0x0	0x0	0x0	0x1	0x1	0x0	0x0
Type	RO-0	RO	RO	RO	RO	RO	RW	RW

Bit	Name	Function
7	N/A	保留位，读 0
6	SMODF	同 SPICTRL[5]
5	SRXOVRN	同 SPICTRL[4]
4	SBUSY	同 SPICFG[7]
3	SRXBMT	同 SPICFG[0]
2	STXBMT	同 CTRL[1]
1	WKF	睡眠模式下，从机在接收数据时，会产生 WAKEUP 唤醒标志，写 0 清零，写 1 无效 0：没有发生 WAKEUP 唤醒或者已被清零 1：发生了 WAKEUP 唤醒事件
0	CRCERR	CRC 错误标志，写 0 清零，写 1 无效 0：传输过程中没有发生 CRC 校验错误或者已被清零 1：传输过程中发生了 CRC 校验错误

14. I2C 接口

I2C 模块通过 SDA 和 SCL 管脚与外部 I2C 器件进行通信，数据以字节格式按照先传高位的原则进行传输。

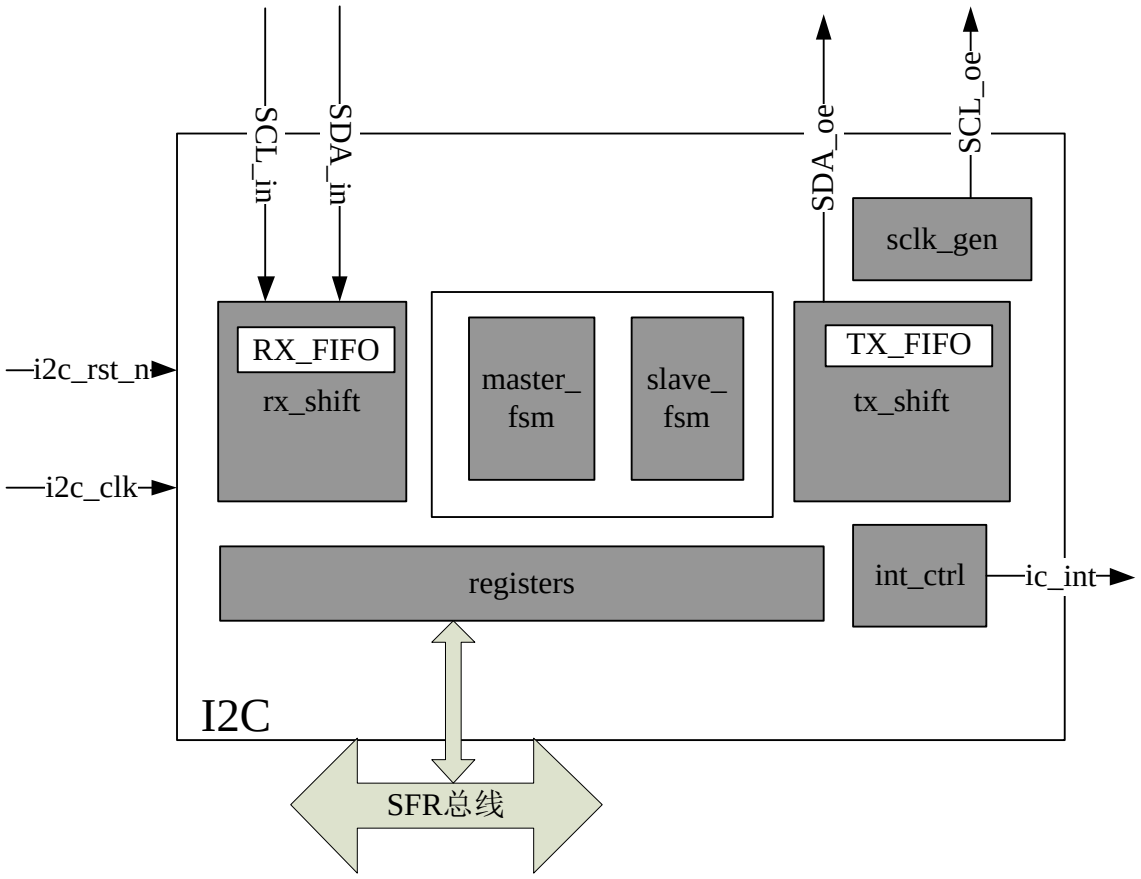


图 14.1 I2C 原理框图

I2C 模块包含以下功能：

- 主机模式和从机模式
- 多主机支持
- 标准模式（100kHz）和快速模式（400kHz）
- 7 位和 10 位地址模式
- General call 支持
- Clock stretching
- 发送 NACK（从机模式）

14.1. I2C 的工作原理

I2C 模块主要有四种工作模式，即主机接收、主机发送、从机发送、从机接收。每种模式下又包含了 7 位地址模式和 10 位地址格式。

14.1.1. 主机发送

主机发送模式下，输出串行数据到 SDA，输出时钟到 SCL。主机发送的第一个字节包括从机地址和读写位，此模式下读写位为 0。然后主机发送 8 位的串行数据，每个数据字节后会接收到 ACK。同时，主机也会产生 Start 和 Stop。

当 I2CCR1 寄存器中 MST10B 位为 0 时，主机发送 7 位地址格式：

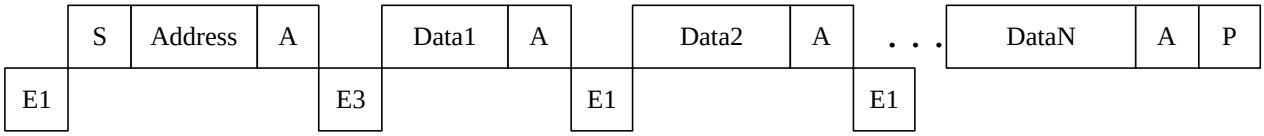


图 14.2 7 比特地址模式主机发送流程图

其中：
E1：IICTXE=1，写 DR 和 CMD 寄存器清零 IICTXE；
E3：ADDF=1，写零到 ADDF 清零 ADDF；
S：表示 START 信号；
A：表示 ACK 信号；
P：表示 STOP 信号；

当 I2CCR1 寄存器中 MST10B 位为 1 时，主机发送 10 位地址格式：

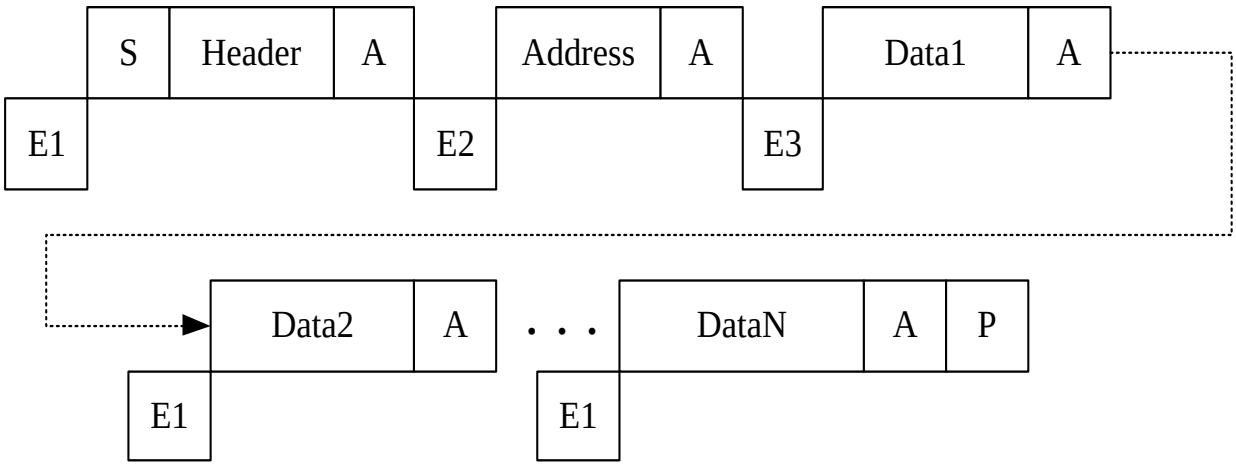


图 14.3 10 比特地址模式主机发送流程图

其中：
E1：IICTXE=1，写 DR 和 CMD 寄存器清零 IICTXE；
E2：ADD10F=1，写零到 ADD10F 清零 ADD10F；
E3：ADDF=1，写零到 ADDF 清零 ADD；

注意：主机发送模式下，软件可以不处理 SBF/ADDF/ADD10F 标志位，只关心 IICTXE。

14.1.2. 主机接收

主机接收模式下，从 SDA 线上接收串行数据，输出时钟到 SCL。主机首先发送从机地址和读写位，此模式下读写位为 1。然后主机接收 8 位的串行数据，每个数据字节后主机需要发送 ACK。同时，主机也会产生 Start 和 Stop。

当 I2CCR1 寄存器中 MST10B 位为 0 时，主机发送 7 位地址格式：

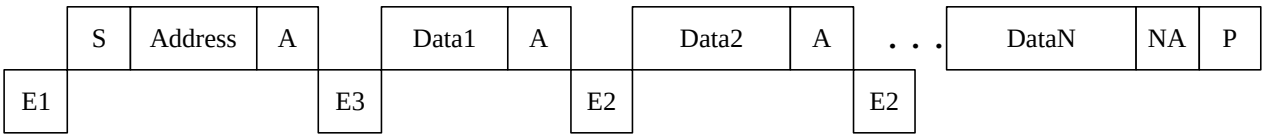


图 14.4 7 比特地址模式主机接收流程图

其中：
E1：IICTXE=1，写 DR 和 CMD 寄存器清零 IICTXE；
E3：ADDF=1，写零到 ADDF 清零 ADDF；
E2：IICRXNE=1，读 DR 寄存器清零 IICRXNE，然后写 CMD=0x1，继续读数据，或者写 0x20，再接收完成一个数据后停止；

当 I2CCR1 寄存器中 MST10B 位为 1 时，主机发送 10 位地址格式：

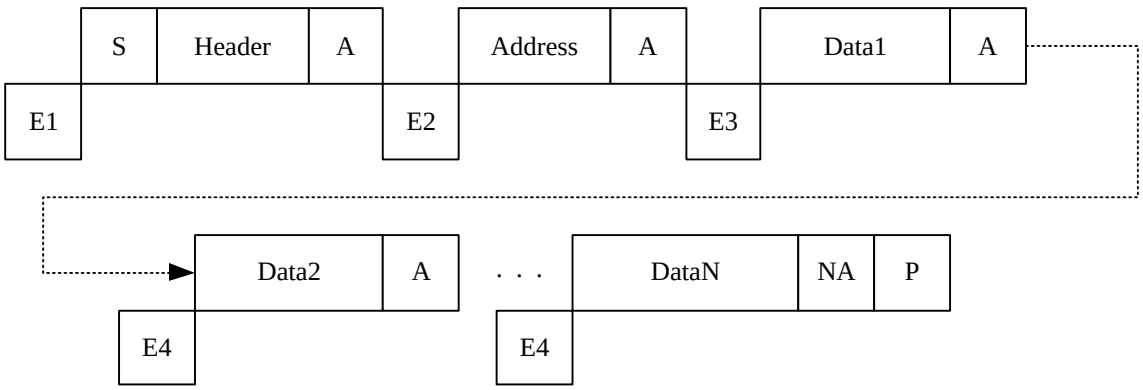


图 14.5 10 比特地址模式主机接收流程图

其中：
E1：IICTXE=1，写 DR 和 CMD 寄存器清零 IICTXE；
E2：ADD10F=1，写 1 到 ADD10F 清零 ADD10F；
E3：ADDF=1，写零到 ADDF 清零 ADDF；
E4：IICRXNE=1，读 DR 寄存器清零 IICRXNE，然后写 CMD=0x1，继续读数据，或者写 0x20，再接收完成一个数据后停止；
注意：主机接收模式下，软件可以不处理 SBF/ADDF/ADD10F 标志位

14.1.3. 从机发送

从机发送模式下，发送串行数据到 SDA。从机在检测到 Start 条件后，首先接收地址字节和读写位，此模式下读写位应为 1。然后发送 8 位的数据字节，每个数据字节后会接收 ACK。同时如果检测到 Stop 条件，从机会结束通信，等待下一次 Start。

当 I2CCR1 寄存器中 SLV10B 位为 0 时，从机响应 7 位地址格式：

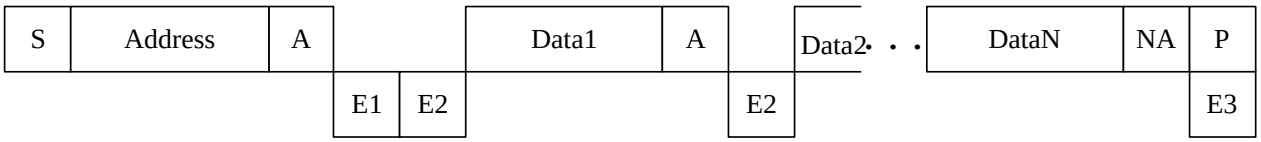


图 14.6 7 比特地址模式从机发送流程图

E1：ADDR=1,拉低 SCL 线，写零到 ADDR 清零 ADDF；
E2：IICTXE=1，拉低 SCL 线，读 SR3 寄存器 rd_req 位为 1，需向 DR 寄存器中写数据清零 IICTXE；
E3：AF=1，写 I2CSR2 寄存器中 AF 位为 0 清零；

当 I2CCR1 寄存器中 SLV10B 位为 1 时，从机响应 10 位地址格式：

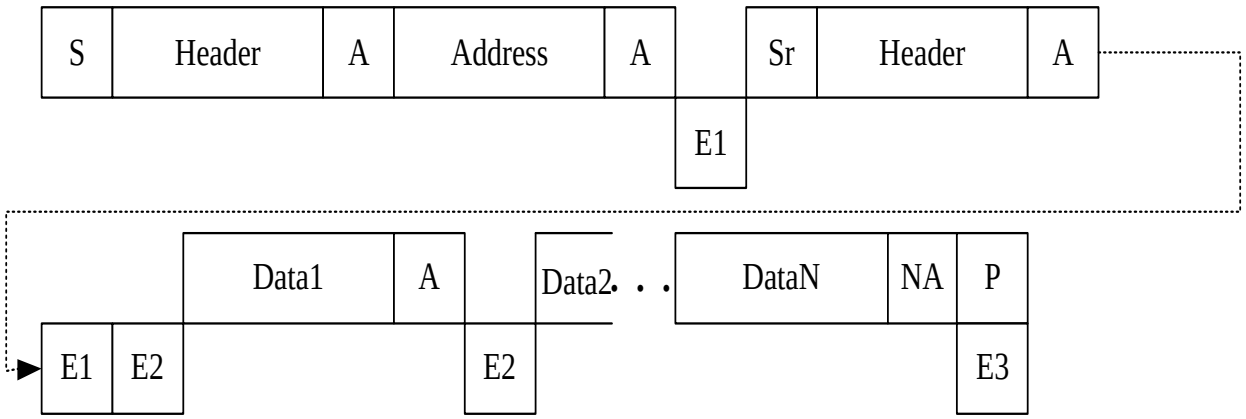


图 14.7 10 比特地址模式从机发送流程图

E1：ADDF=1，拉低 SCL 线，写零到 ADDF 清零；
E2：IICTXE=1，拉低 SCL 线，读 SR3 寄存器 rd_req 位为 1，需向 DR 寄存器中写数据清零 IICTXE；
E3：AF=1，写 SR2 寄存器中 AF 位为 0 清零；

14.1.4. 从机接收

从机接收模式下，从 SDA 线接收串行数据。从机在检测到 Start 条件后，首先接收地址字节和读写位，此模式下读写位应为 0。然后接收 8 位的数据字节，每个数据字节后需要发送 ACK。同时如果检测到 Stop 条件，从机会结束通信，等待下一次 Start。

当 I2CCR1 寄存器中 SLV10B 位为 0 时，从机响应 7 位地址格式：

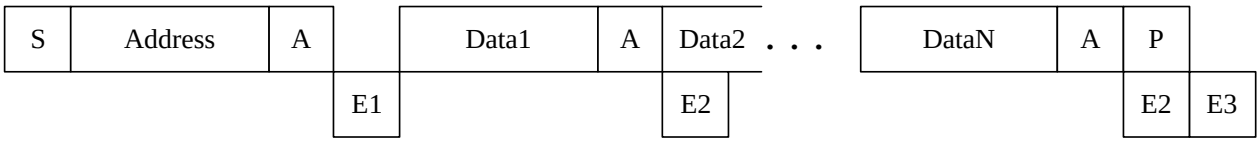


图 14.7 7 比特地址模式从机接收流程图

E1：ADDF=1，写零到 ADDF 清零；
E2：IICRXNE=1，读取 DR 寄存器中数据清零 IICRXNE；
E3：STOPF=1，写零到 STOPF 清零；

当 I2CCR1 寄存器中 SLV10B 位为 1 时，从机响应 10 位地址格式：

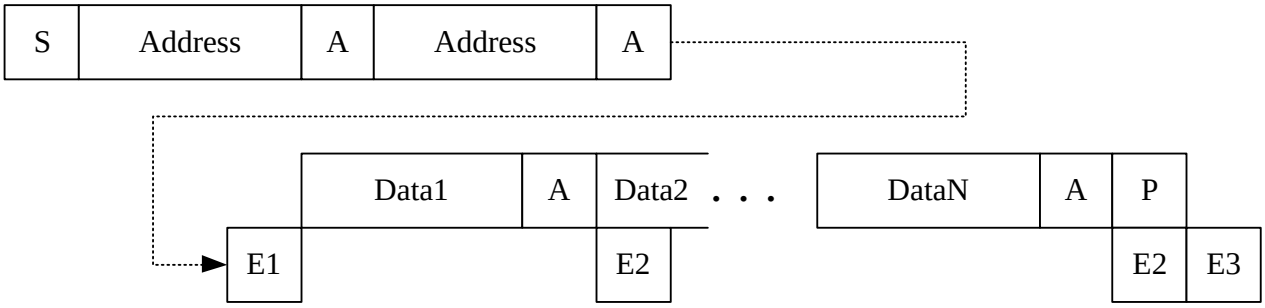


图 14.8 10 比特地址模式从机接收流程图

E1：ADDF=1，写零到 ADDF 清零；
E2：IICRXNE=1，读取 DR 寄存器中数据清零 IICRXNE；
E3：STOPF=1，写零到 STOPF 清零 STOPF；

14.1.5. General Call

General Call 模式在主机置位了 AGCALL 以后，就会向地址为 0x00 的地址发送数据，这种模式下主机只允许进行写数据，不允许读数据；从机模式下在置位了 AGCALL 以后，就会响应主机发来的 General Call；通信的过程跟主机发送，从机接收模式相同。

14.2. 与 I2C 相关寄存器汇总

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	复位值
I2CCR1	0x40C	—	—	—	MST10B	SLV10B	—	SPEED	MASTER	---0 0-00
I2CCR2	0x40D	—	SOFTRST	AGCALL	SNACK	—	—	RXHLD	—	-000 —0-
I2CCR3	0x40E	—					EVSTRE	—	ENABLE	---- -000
I2COARL	0x40F	ADD[7:0]								0000 0000
I2COARH	0x410	—	—	—	—	—	—	ADD[9:8]		---- --00
I2CFREQ	0x411	—	—	FREQ[5:0]						--00 0000
I2CDR	0x412	DR[7:0]								0000 0000
I2CCMD	0x413	—	—	—	—	—	RESTART	STOP	MSTDIR	---- -000
I2CCCRL	0x414	CCR[7:0]								0000 0000
I2CCCRH	0x415	—	DUTY	—	—	CCR[11:8]				-0—0000
I2CITR	0x416	—					ITBUFEN	ITEVEN	ITERREN	---- -000
I2CSR1	0x417	IICTXE	IICRXNE	—	STOPF	ADD10F	—	ADDF	SBF	00-0 0-00
I2CSR2	0x418	—	—	—	TXABRT	OVR	AF	ARLO	BERR	---0 0000
I2CSR3	0x419	—	—	GCALL	—	—	RDREQ	ACTIVE	RXHOLD	--0- -000

14.2.1. I2CCR1 寄存器，地址 0x40C

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	MST10B	SLV10B	—	SPEED	MASTER
Reset	—	—	—	0	0	—	0	0
Type	RO-0	RO-0	RO-0	RW	RW	RO-0	RW	RW

Bit	Name	Function
7:5	N/A	保留位，读 0
4	MST10B	主机模式下地址格式，该位只有在 I2C 模块禁用时才能进行写操作 0：发送 7 位地址格式； 1：发送 10 位地址格式；
3	SLV10B	从机模式下地址格式，该位只有在 I2C 模块禁用时才能进行写操作 0：响应 7 位地址格式； 1：响应 10 位地址格式；
2	N/A	保留位，读 0
1	SPEED	I2C 通信速度模式，该位只有在 I2C 模块禁用时才能进行写操作 0：标准模式（100kHz）； 1：快速模式（400kHz）；
0	MASTER	主从机模式，该位只有在 I2C 模块禁用时才能进行写操作 0：从机模式； 1：主机模式；

14.2.2. I2CCR2 寄存器，地址 0x40D

Bit	7	6	5	4	3	2	1	0
Name	—	SOFTRST	AGCALL	SNACK	—	—	RXHLD	—
Reset	—	0	0	0	—	—	0	—
Type	RO-0	RW	RW	RW	RO-0	RO-0	RW	RO-0

Bit	Name	Function
7	N/A	保留位，读 0
6	SOFTRST	软件复位，对 I2C 模块进行复位；该位只有在 ACTIVE 为 1 时才能进行写 1 操作 0：没有影响； 1：复位 I2C 控制模块； 注意：该复位不会复位寄存器的值；由于一些异常原因导致 I2C 模块一直处于活动状态时，可以置位该位对发送和接收模块进行复位
5	AGCALL	从机时，应答 General call 使能，该位只有在 I2C 模块禁用时才能进行写操作 0：不响应 General call； 1：响应 General call（在使能了 SNACK 时才会有效） 主机时，发送 General call 使能 0：发送正常的从机地址； 1：发送 General call 地址(0x00)；
4	SNACK	从机接收是否发送 NACK，该位只有在 I2C 模块禁用时才能进行写操作 0：字节接收后发送 ACK（地址匹配或数据字节）； 1：发送 NACK；
3:2	N/A	保留位，读 0
1	RXHLD	RX-FIFO 满控制位，该位只有在 I2C 模块禁用时才能进行写操作 0：RX-FIFO 满时不拉低 SCL，新接收的数据将会丢失； 1：RX-FIFO 满时拉低 SCL；
0	N/A	保留位，读 0

14.2.3. I2CCR3 寄存器，地址 0x40E

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	EVSTRE	—	ENABLE
Reset	—	—	—	—	—	0	—	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RO-0	RW

Bit	Name	Function
7:1	N/A	保留位，读 0
2	EVSTRE	SBF/ADD/ADD10 拉低 SCL 使能 0：SBF/ADD/ADD10 不拉低 SCL 1：SBF/ADD/ADD10 拉低 SCL
1	N/A	保留位，读 0
0	ENABLE	I2C 模块使能 0：禁用 I2C 模块； 1：使能 I2C 模块，相应的 IO 管脚会用作 I2C 的功能；

14.2.4. I2COARL 寄存器，地址 0x40F

Bit	7	6	5	4	3	2	1	0
Name	ADD[7:0]							
Reset	0x00							
Type	RW							

Bit	Name	Function
7:0	ADD[7:0]	从机地址，，该位只有在 I2C 模块禁用时才能进行写操作 7 位地址格式时：bit[7] 不关心； 10 位地址格式时：10 位地址的低 8 位； 注：如果模块用作主机，则该寄存器存储的是目标从机的地址，而在用作从机时表示的是本机的地址

14.2.5. I2COARH 寄存器，地址 0x410

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	ADD[9:8]	
Reset	—	—	—	—	—	—	2’b00	
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	

Bit	Name	Function
7:2	N/A	保留位，读 0
1:0	ADD[9:8]	从机地址：10 位地址的高 2 位，该位只有在 I2C 模块禁用时才能进行写操作 注：如果模块用作主机，则该寄存器存储的是目标从机的地址，而在用作从机时表示的是本机的地址

14.2.6. I2CFREQ 寄存器，地址 0x411

Bit	7	6	5	4	3	2	1	0
Name	—	—	FREQ[5:0]					
Reset	—	—	6'h0					
Type	RO-0	RO-0	RW					

Bit	Name	Function
7:6	N/A	保留位，读 0
5:0	FREQ[5:0]	外设时钟频率，请根据实际的外设时钟频率来设定该位的值；该位只有在 I2C 模块禁用时才能进行写操作 6'b000000：不允许； 6'b000001：1MHz； 6'b000010：2MHz； … 6'b011000：24MHz； Higher values：不允许；

14.2.7. I2CDR 寄存器，地址 0x412

Bit	7	6	5	4	3	2	1	0
Name	DR[7:0]							
Reset	0x00							
Type	RW							

Bit	Name	Function
7:0	DR[7:0]	数据寄存器 写时：将该数据 push 到 TX-FIFO 中； 读时：读出 RX-FIFO 中数据； 注意： 主机模式下写数据时，push 动作是在写完 I2CCMD 寄存器之后，因此需要先写 I2CDR 寄存器，再写 I2CCMD 寄存器；发送和接收的 FIFO 深度都为 1

14.2.8. I2CCMD 寄存器，地址 0x413

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	RESTART	STOP	MSTDIR
Reset	—	—	—	—	—	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	W	W	W

Bit	Name	Function
7:3	N/A	保留位，读 0
2	RESTART	发送 Restart，读到的值没有意义 0：字节传输后不发送 Restart； 1：字节传输后发送 Restart；
1	STOP	发送 Stop，读到的值没有意义 0：字节传输之后不发送 Stop； 1：字节传输之后发送 Stop；
0	MSTDIR	主机模式时的传输方向，读到的值没有意义 0：发送数据； 1：读取数据； **注意：请不要对该寄存器进行单比特操作，建议使用字节传输的指令对该寄存器进行操作，因为任何对该寄存器的写操作（无论是单比特还是字节操作）都会立即把这个数据写到 FIFO 中**

14.2.9. I2CCCRL 寄存器，地址 0x414

Bit	7	6	5	4	3	2	1	0
Name	CCR[7:0]							
Reset	0x00							
Type	RW							

Bit	Name	Function
7:0	CCR[7:0]	主机模式时 SCL 时钟周期的低 8 位；该位只有在 I2C 模块禁用时才能进行写操作

14.2.10. I2CCCRH 寄存器，地址 0x415

Bit	7	6	5	4	3	2	1	0
Name	—	DUTY	—	—	CCR[11:8]			
Reset	—	0	—	—	4'h0			
Type	RO-0	RW	RO-0	RO-0	RW			

Bit	Name	Function
7	N/A	保留位，读 0
6	DUTY	快速模式下占空比选择，该位只有在 I2C 模块禁用时才能进行写操作 0： Tlow/Thigh = 2； 1： Tlow/Thigh = 16/9； **注意：标准模式下 Tlow/Thigh = 1；**
5:4	N/A	保留位，读 0
3:0	CCR[11:8]	主机模式时 SCL 时钟周期的高 4 位，该位只有在 I2C 模块禁用时才能进行写操作 下表为具体的 SCL 时钟周期公式： 其中，Fmaster 为外设时钟频率；

	周期	SCLH	SCLL
标准模式	2*CCR*Fmaster	CCR*Fmaster	CCR*Fmaster
快速模式（DUTY=0）	3*CCR*Fmaster	CCR*Fmaster	2*CCR*Fmaster
快速模式（DUTY=1）	25*CCR*Fmaster	16*CCR*Fmaster	9*CCR*Fmaster

14.2.11. I2CITR 寄存器，地址 0x416

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	ITBUFEN	ITEVEN	ITERREN
Reset	—	—	—	—	—	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW

Bit	Name	Function
7:3	N/A	保留位，读 0
2	ITBUFEN	FIFO 状态中断使能 0：IICTXE=1 或 IICRXNE=1 不产生中断； 1：IICTXE=1 或 IICRXNE=1 产生中断；
1	ITEVEN	事件中断使能 0：Disabled 事件中断使能； 1：使能事件中断； 事件中断产生条件： SB=1（master） ADDR=1（master/slave） ADD10=1（master） STOPF=1（slave）
0	ITERREN	错误中断使能 0：Disable 错误中断； 1：使能错误中断； 错误中断产生条件： BERR=1 ARLO=1 AF=1 OVR=1

14.2.12. I2CSR1 寄存器，地址 0x417

Bit	7	6	5	4	3	2	1	0
Name	IICTXE	IICRXNE	—	STOPF	ADD10F	—	ADDF	SBF
Reset	0	0	—	0	0	—	0	0
Type	RO	RO	RO-0	RO	RO	RO-0	RO	RO

Bit	Name	Function
7	IICTXE	TX-FIFO 空 0：TX-FIFO 非空； 1：TX-FIFO 空； --发送条件下 TX-FIFO 空时置位； --软件写 I2CDR 寄存器，或者 disable I2C 时硬件清零； **注意：该寄存器位在数据传输过程中一直是 0，在当前数据传输完成后会置 1，因为 FIFO 的深度为 1，则推荐在该标志位为 1 时在向 FIFO 中写入数据，否则会导致数据写入失败，置位 OVR 标志位**
6	IICRXNE	RX-FIFO 非空 0：RX-FIFO 为空； 1：RX-FIFO 非空； --接收条件下 RX-FIFO 非空时置位； --软件读 I2CDR 寄存器，或者 disable I2C 时硬件清零；
5	N/A	保留位，读 0
4	STOPF	从机模式下 Stop 检测 0：没有检测到 Stop； 1：检测到 Stop； --从机模式时 ACK 之后检测到 Stop 时置位； --软件读 I2CSR1 寄存器，或者 disable I2C 时硬件清零；
3	ADD10F	主机模式下发送 10 位地址 Header； 0：没有发送 10 位地址 Header； 1：主机发送 10 位地址 Header； --主机发送 10 位地址 Header 时置位（ACK 之后）； --软件读 I2CSR1 寄存器，或者 disable I2C 时硬件清零；
2	N/A	保留位，读 0
1	ADDF	地址发送（主机）/地址匹配（从机）： 软件读 I2CSR1 寄存器，或者 disable I2C 时硬件清零； --地址匹配（从机） 0：接收地址不匹配； 1：接收地址匹配； --接收地址匹配或者识别 General Call 后置位； --地址发送（主机）： 0：地址传输没有完成； 1：地址传输完成； --对于 10 位地址：第二个地址字节之后置位（ACK 后）； --对于 7 位地址：地址字节之后置位（ACK 后）； **注意：NACK 后不会置位 ADDF；**
0	SBF	主机模式下 Start 产生 0：没有发送 Start； 1：发送 Start； --主机模式时发送 Start 置位； --软件读 I2CSR1 寄存器，或者 disable I2C 时硬件清零；

14.2.13. I2CSR2 寄存器，地址 0x418

Bit	7	6	5	4	3	2	1	0
Name	—			TXARBT	OVR	AF	ARLO	BERR
Reset	—			0	0	0	0	0
Type	RO-0			RW0	RW0	RW0	RW0	RW0

RW0：只能写 0，不能写 1

Bit	Name	Function
7:3	N/A	保留位，读 0
4	TXABRT	发送过程中由于出错或异常原因导致发送终止，写零清零或者 disable I2C 时硬件清零； 0：传输未发生终止 1：传输发生终止
3	OVR	Overrun 产生 0：没有 Overrun； 1：产生 Overrun； --以下条件时置位： tx-over：当 TX-FIFO 中有数据时仍写 I2CDR 寄存器； rx_over：RX-FIFO 中有数据时仍接收数据； rx_under：RX-FIFO 空时进行读操作； --软件该位写 0，或者 disable I2C 时硬件清零；
2	AF	无 ACK： 0：ACK 正常； 1：NACK 产生； --当产生 NACK 时硬件置位； --软件该位写 0，或者 disable I2C 时硬件清零；
1	ARLO	主机仲裁失败 0：无仲裁失败产生； 1：产生仲裁失败； --主机仲裁失败时置位； --软件该位写 0，或者 disable I2C 时硬件清零；
0	BERR	总线错误 0：没有检测到错位的 Start/Stop； 1：检测到错位的 Start/Stop； --字节传输阶段检测到 Start/Stop 时置位； --软件该位写 0，或者 disable I2C 时硬件清零；

14.2.14. I2CSR3 寄存器，地址 0x419

Bit	7	6	5	4	3	2	1	0
Name	—	—	GCALL	—	—	RDREQ	ACTIVE	RXHOLD
Reset	—	—	0	—	—	0	0	0
Type	RO-0	RO-0	RO	RO-0	RO-0	RO	RO	RO

Bit	Name	Function
7:6	N/A	保留位，读 0
5	GCALL	从机模式接收到 General call --从机模式接收并且 ACK General call 时置位； --检测到 Start/Stop，或者 disable I2C 时硬件清零；
4:3	N/A	保留位，读 0
2	RDREQ	从机模式读请求 0：从机接收数据； 1：从机发送数据； --从机接收地址字节的读写位为 1 时置位； --检测到 Start/Stop，或者 disable I2C 时硬件清零；
1	ACTIVE	主从状态机状态 0：主从状态机处于 IDLE 状态，总线处于空闲状态； 1：主从状态机处于 Busy 状态； --主从状态机处于 IDLE 状态时置位； --主从状态机处于 Busy 状态时清零；
0	RXHOLD	接收满保持状态，此时 SCL 被拉低，在读取数据寄存器后释放 0：接收未满，未出现 SCL 被拉低的状态 1：接收 FIFO 满，SCL 被拉低状态

15. USART 接口

15.1. 功能特性

- 同步模式
  - 产生同步时钟输出
- 多芯片通信模式
  - 哑模式唤醒之后，才可以接收数据
  - 可以通过地址匹配和 IDLE 帧唤醒哑模式
- 异步模式
  - 可编程的 7, 8, 9 比特数据模式
  - 支持 1, 2, 1.5 bit 停止位
  - 支持红外 1.0 模式
  - 单线半双工
  - 发送接收使能控制
  - 16bit 波特率设置
  - RXNE 中断，TXE 中断，IDLE 帧中断，break 帧中断，奇偶校验错误，overrun 中断，发送完成中断
- 智能卡模式
  - 1.5 bit 停止位
  - 时钟输出
  - guard time
- LIN 主机模式
  - 支持断开帧的发送与检测
- 自动波特率检测

15.2. 功能描述

15.2.1. 一般描述

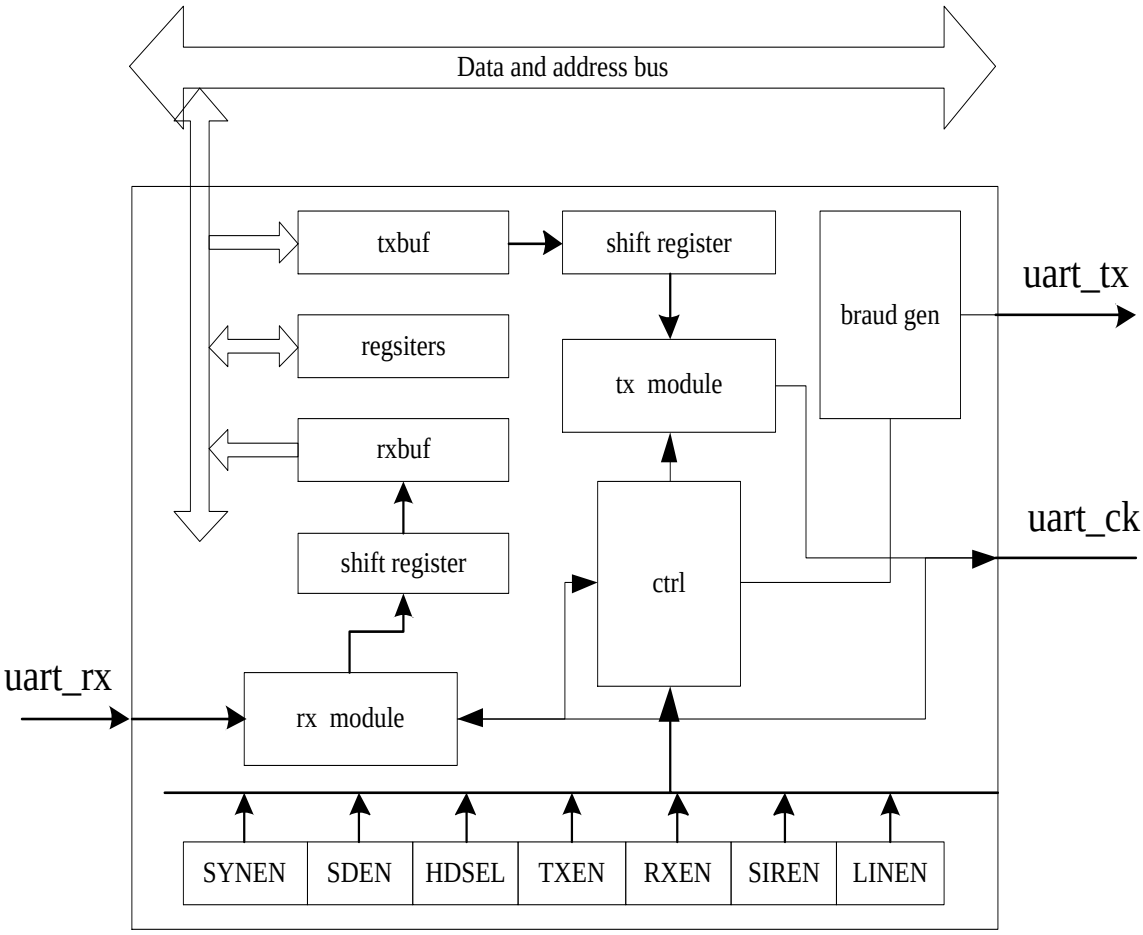


图 15.1 USART 原理框图

串口模块总共有三只引脚：

1. uart_rx：用作串口数据的输入引脚
2. uart_tx：用作串口数据的输出引脚，在用作半双工模式时也用作串行数据的输入引脚
3. uart_ck：在同步模式时用作同步时钟输出，在智能卡模式时用系统作分频时钟输出

该模块支持同步模式，异步模式，半双工模式，LIN Master 模式，红外模式和智能卡模式，默认的状态是工作于异步全双工模式，在确定使用一种模式后，请确保其他模式的相关使能位已关闭。

15.2.2. 异步工作模式

异步工作模式的串口采用异步的方式进行通信，配置的步骤如下：

1. 配置 DLH/DLL 产生相应的波特率进行通信，DLH 和 DLL 共同组成 16 位的波特率分频器，通信的波特率=fmaster/(16*(DL*))，其中 fmaster 为系统时钟，16 位的波特率分频器的值最小位 1，DL*表示的是 DLL 和 DLH 的组合，设置为零时串口不工作；
2. 配置 LCR 寄存器中的 LTH 位和 LCREXT 寄存器中的 EXTEN 来设置通信的数据长度，配置 LCR 寄存器中的 STOP 位来配置停止位的长度，配置 LCR 寄存器中的 PEN 和 EVEN 来配置奇偶校验位，配置 IER 寄存器中的中断使能位来允许中断；
3. 配置 MCR 寄存器中的 TXEN 和 RXEN 来使能允许发送和接收；

异步模式通信的数据格式是先发送低位数据位，最后发送高比特位，如下图所示的 8 比特数据格式不带奇偶校验和带奇偶校验的帧格式。

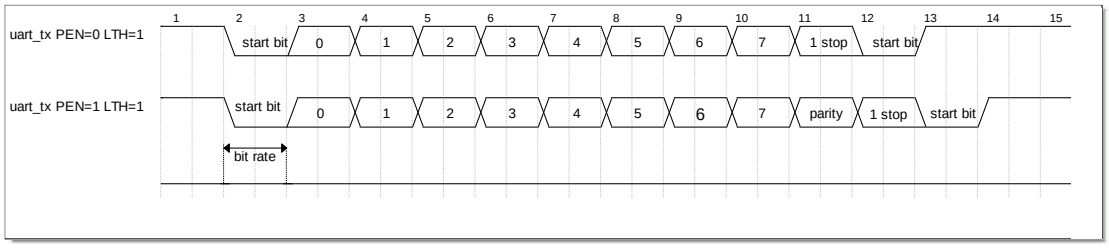


图 15.2 异步模式时序图

异步模式的数据处理流程包括阻塞模式处理和非阻塞模式处理，大致的处理流程如下：

1. 配置完波特率和相关控制位以后，发送端可以向 DATAL/H 发送 buf 寄存器写入数据，在阻塞模式下可以查询 TXEF 标志位，如果查询到 TXEF 为 1，则可以继续向 DATAL/H 写入需要发送的数据；在非阻塞模式下，使能发送为空中断，则在 TXEF 为 1 时，就会自动进入中断，向 DATAL/H 写入数据就可清除 TXEF 标志位，当在向 txbuf 写入最后一个要发送的数据时，禁用发送为空中断；
2. 接收端在阻塞模式下可以查询 RXNEF 标志位，在查询到该标志位为 1 时，表示接收到了数据，通过读取 DATAL/H 来清零 RXNEF 标志位；采用非阻塞模式接收数据时，需要使能 RXNE 中断，在串口接收到数据后，直接进入中断，读取 rxbuf 后清零 RXNEF 标志位；在采用非阻塞模式接收数据时，建议打开 RXSE 中断使能，在接收数据的过程中如遇到接收错误就会直接进入中断进行相关的处理；
3. 在串口发送数据的时候也可以使用 TCF 标志位来处理，在 TCF 标志位为 1 时，表示当前的数据发送已经完成，可以向 txbuf 写入下一个要发送的数据，这时 TCF 标志位会自动清零；

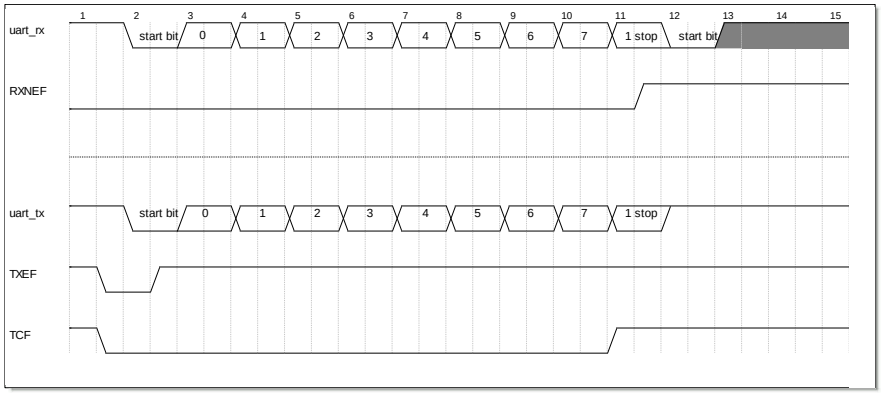


图 15.3 异步模式标志位时序图

15.2.3. 同步工作模式

同步工作模式用于串口模拟 SPI 通信的功能，在串口数据输出的同时，输出一个与数据相关的同步时钟，同步时钟的极性和相位可以通过 URSYNCR 寄存器中的 CPOL 和 CPHA 来配置；URSYNCR 寄存器中的 LBCL 控制的是最后一比特数据的时钟是否输出，该位为零时，只输出数据长度减一个有效同步时钟，最后一个有效时钟不会输出；SYNEN 就是同步时钟输出使能位，在该位为 1 时相应的 IO 会用作同步时钟输出；该模块只能模拟 SPI 主机模式，数据输出是先发送低位数据，然后发送的高位数据，并且时钟引脚只能输出同步时钟，并不能用作时钟输入；

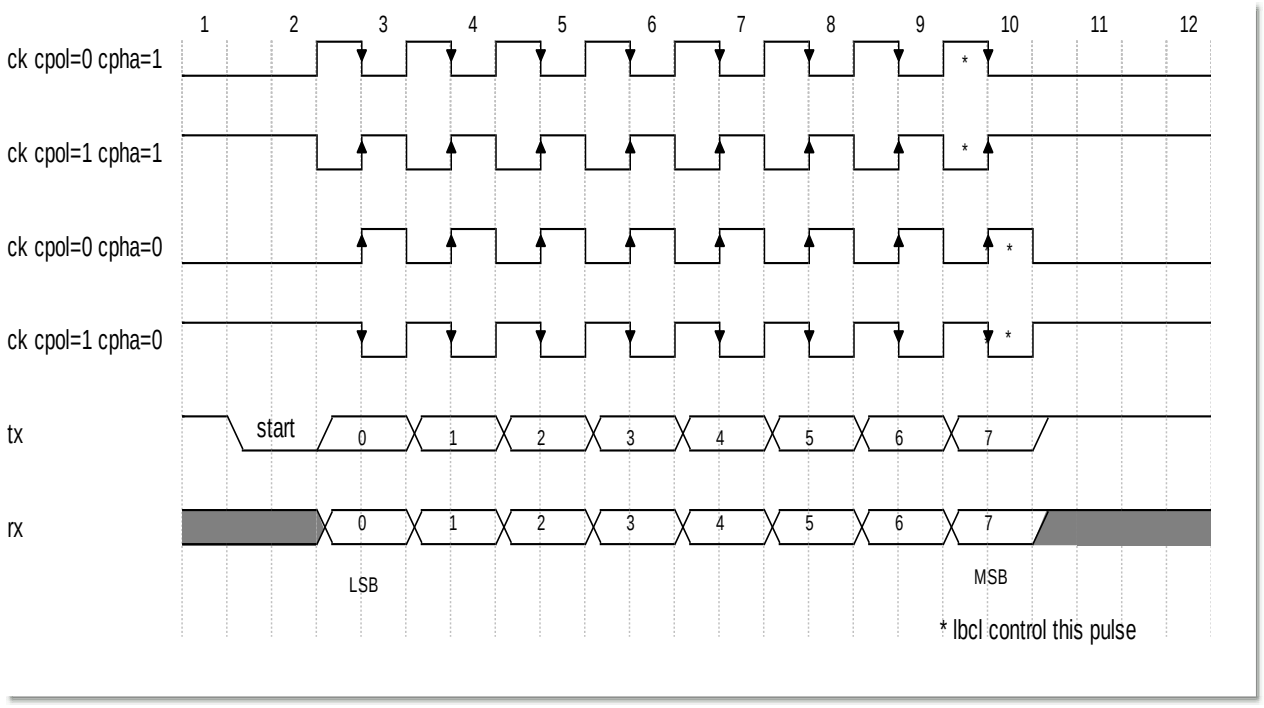


图 15.4 同步模式时序图

如图所示八比特数据格式的同步时钟输出，同步模式中如果没有使能 TXEN 也会产生同步脉冲输出，这时候同步模式只用于接收数据，写入到 DATAL/H 寄存器中的数据会发送到内部的移位寄存器中，用于产生同步脉冲输出，tx 引脚的值一直保持为 1，在同步发送的同时如果使能了 RXEN 接收位，则可以同步接收数据。

15.2.4. 半双工模式

半双工模式属于异步工作方式的一种，只是在通信时只用到了 tx 引脚，tx 引脚 IO 应该配置成开漏模式，发送与接收的处理由软件控制 RXEN 和 TXEN 来实现；需要注意的是如果在发送过程中同时使能了接收，则发送的数据也会被本机接收到；置位 HDSEL 即可启用半双工模式。

15.2.5. 红外工作模式

红外模式用于红外通信，置位 SIREN 位可以使能红外模式，同时 LTH 位置位为 1，启用八比特数据格式；通信的波特率设置跟异步串口的配置方法相同。

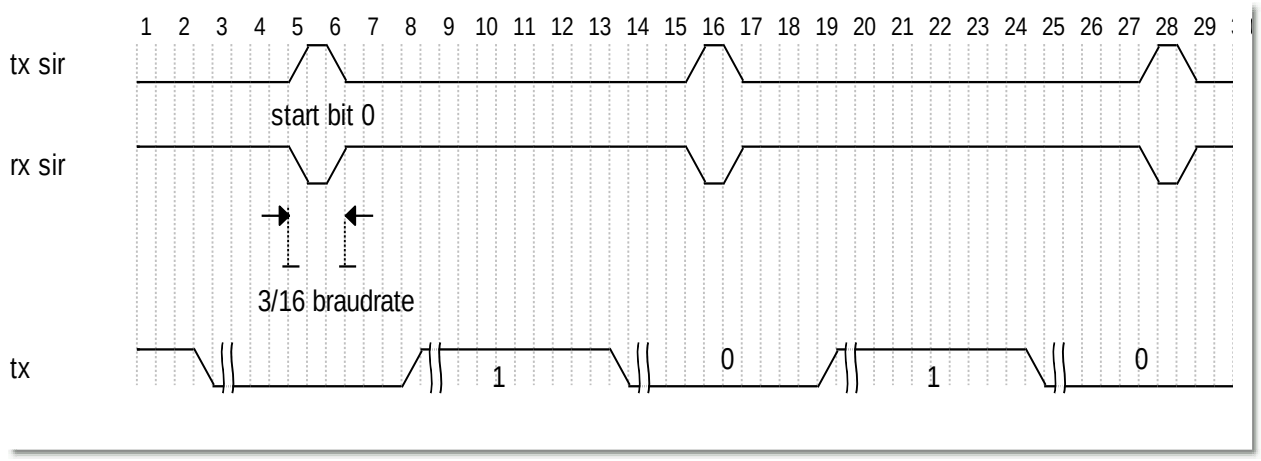


图 15.5 红外模式时序图

如图所示红外模块发送的脉冲宽度是比特周期的 3/16，当发送的数据为零时会产生一个高脉冲；接收时的低脉冲会被解释成零；接收与发送的总线极性是相反的，发送空闲时总线保持低电平，接收空闲时总线保持为高电平。

红外模式可以工作在低功耗模式，红外模式通常工作在系统的时钟频率下，红外的通信波特率 =fmaster/(16*DL*)；当使能了 SIRLP 以后，红外的通信波特率= fmaster/(PSC*16*DL*)；这里的 DL*表示 DLL 和 DLH 的组合，在 psc 设置为 0 或 1 时，psc 分频模块无效，波特产生模块直接使用 Fmaster，如下图所示。

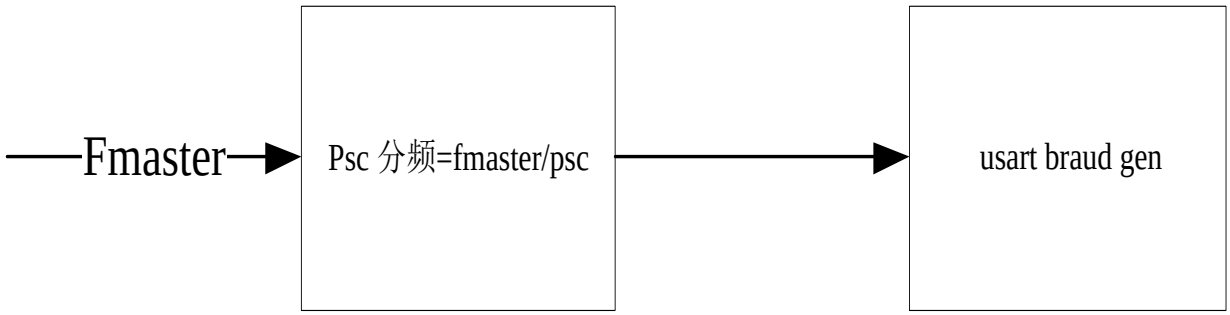


图 15.6 红外低功耗模式原理框图

15.2.6. 智能卡模式

智能卡模式属于半双工模式，支持 ISO7816-3 标准，置位 SDEN 来启用智能卡模式，除此之外根据协议要求需要使能 1.5 比特停止位 STOP 和奇偶校验位 PEN，同时需要配置相应的 IO 为开漏模式。

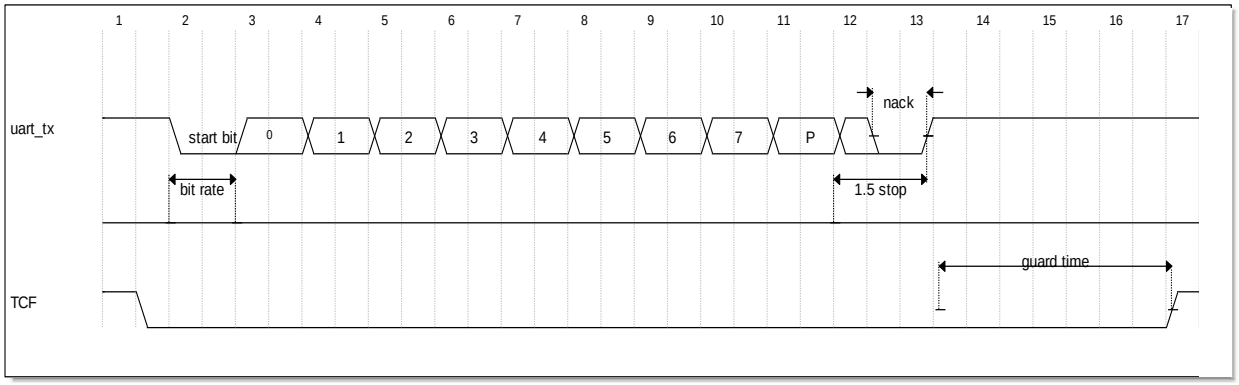


图 15.7 智能卡模式时序图

在使能了 NACK 位以后，接收方在检测到奇偶校验出错以后，会在 0.5 个停止位之后拉低总线一个比特周期，同时发送方会在停止位处检测总线是否被拉低，若检测到总线被拉低，则会产生帧错误标志 FEF，发送方根据要求可以选择重发当前的数据，发送次数由用户决定。在没有使能 NACK 位时，接收方在检测到奇偶校验错误时，不会拉低总线而是会产生一个奇偶检验错误标志位 PEF。

智能卡模式的发送方发送完成数据后，TCF 标志位会在经过 GT 个波特周期后置位；发送与接收的处理由软件控制 TXEN 和 RXEN 来处理。

智能卡模式中，可以通过使能 CKOE 来输出一个时钟供给使能卡使用，输出的时钟频率详见 URSDCR2 寄存器说明；**需要注意的是在置位 CKOE 后，请配置 PSC 的值为有效值，否则不应该置位 CKOE。**

15.2.7. LIN Master 模式

串口模块支持 LIN Master 模式，使能 LINEN 后进入 LIN Master 模式，在发送断开帧之前请先配置一下断开帧的长度 BLTH；如图所示，在置位 BKREQ 后，tx 引脚会发送 BLTH 个连续的低电平，发送完成后自动清零，在使能该位后，可以查询 BKREQ 的状态，等到 BKREQ 为 0 时则表示断开帧发送完成；在发送断开帧的过程中请勿手动清零 BKREQ。

接收端在接收到大于起始位+数据长度+停止位个数的连续低电平以后，会被认为接收到了断开帧，BKF 会被置 1。

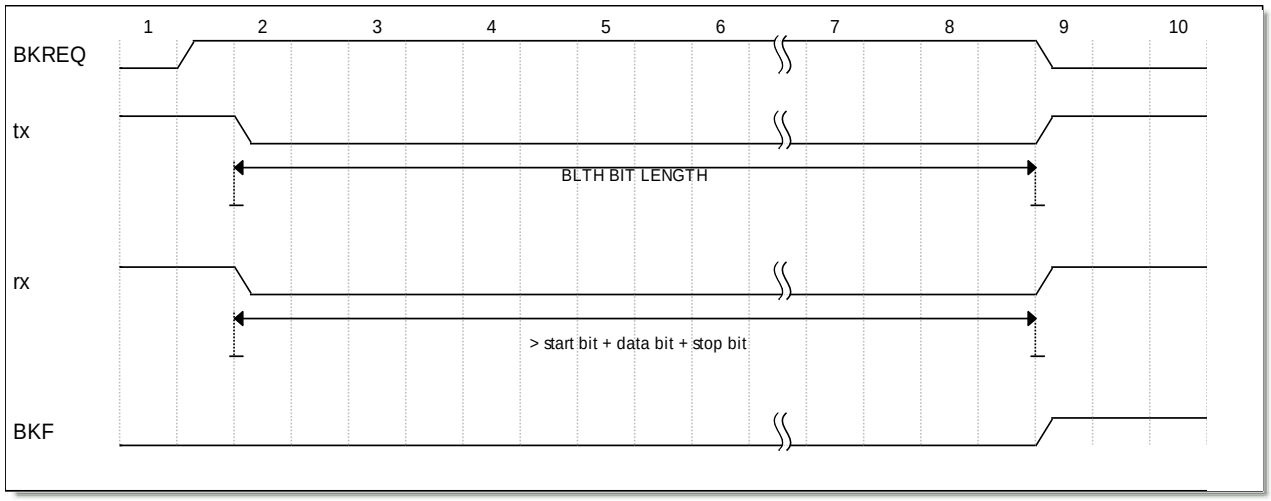


图 15.8 LIN Master 模式时序图

需要注意的是断开帧的接收与发送并不局限于 LIN mode，其他异步模式，红外模式等也是可以应用。

15.2.8. 多芯片通信模式

多处理器通信用于一个芯片用作主机模式，其他芯片用作从机模式，从机的发送引脚通过逻辑与的方式连接到主机的 RX 引脚，这种模式中主机希望接收特定的消息，只有在特定的条件触发后才会接收数据。

置位 RWU 后即可进入哑模式，屏蔽一切接收，根据 WAKE 的配置，可以唤醒接收主机接收数据：

1. WAKE 置零，在接收到起始位+数据位+停止位总个数波特周期后唤醒；
2. WAKE 置一，在接收到匹配的地址后唤醒；

- 地址空闲唤醒，在置位 RWU 后，如果总线数据一致繁忙，则不唤醒接收，在检测到连续的一帧空闲时间（起始位+数据位+停止位）后唤醒开始接收数据。

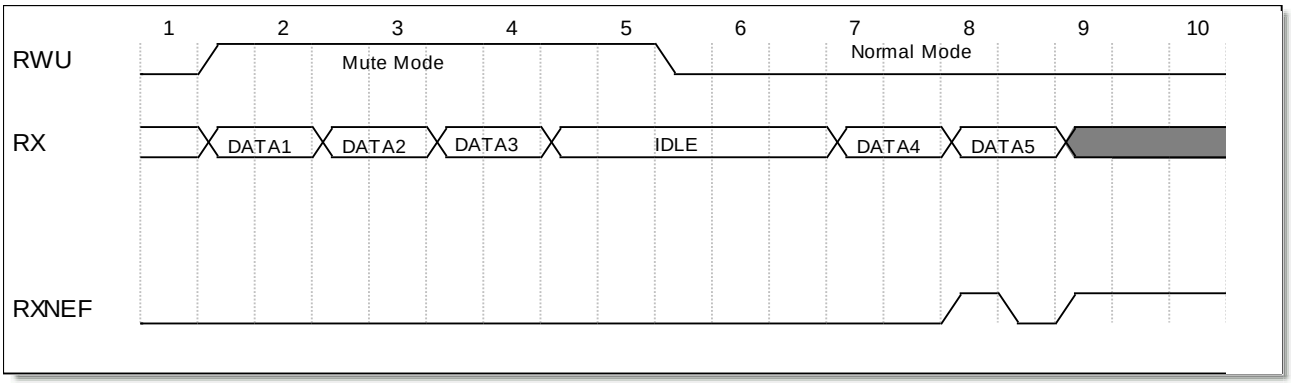


图 15.9 哑模式空闲唤醒时序图

- 地址匹配唤醒，在置位 RWU 后，每次接收到数据后都会判断数据的高位是否为 1，若为 1 则把数据的低四位与 URRAR 的值进行比较，若相等则退出哑模式开始接收之后的数据，后续如果再次接收到地址数据（该模式下数据的高位为 1 则表示接收到的数据为地址数据），则还是会与本机的地址 URRAR 进行比较，若不同立即进入哑模式；该模式下每次匹配到地址后 ADDF 会被置 1，反之没有匹配到地址时一直为零。

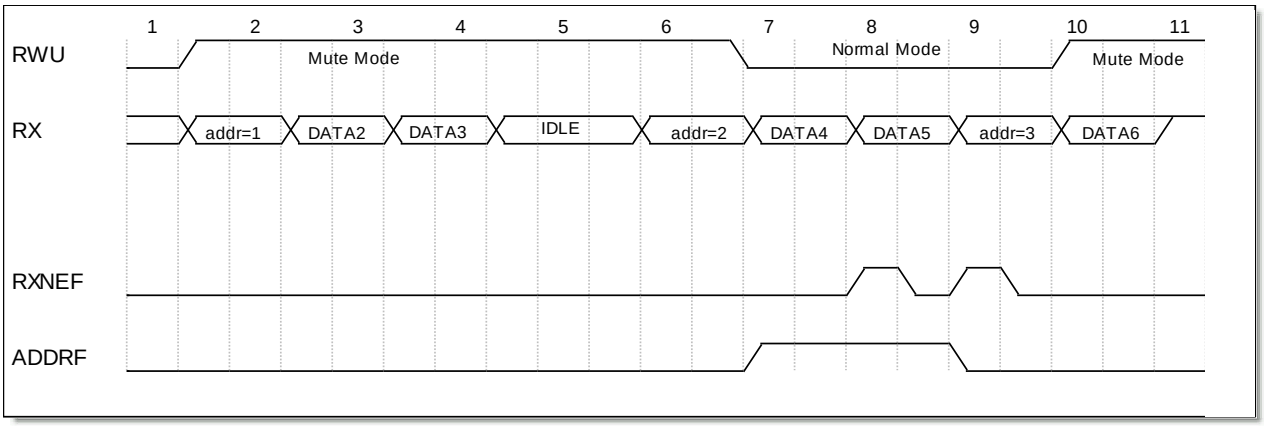


图 15.10 哑模式地址匹配唤醒时序图

15.2.9. 自动波特率检测

自动波特率检测功能用于接收端校准通信波特率，保持与发送端波特率相同，串口模块实现的波特率检测模块有两种模式：

1. 检测起始位的长度（model0）；这种模式要求数据的第一比特为一，例如数据 0x03、0x55 等；
2. 检测起始比特和第一比特的长度（model1）；这种模式要求第一比特的数据为 1，第二比特的数据为 0，例如数据 0x55，0x01 等；

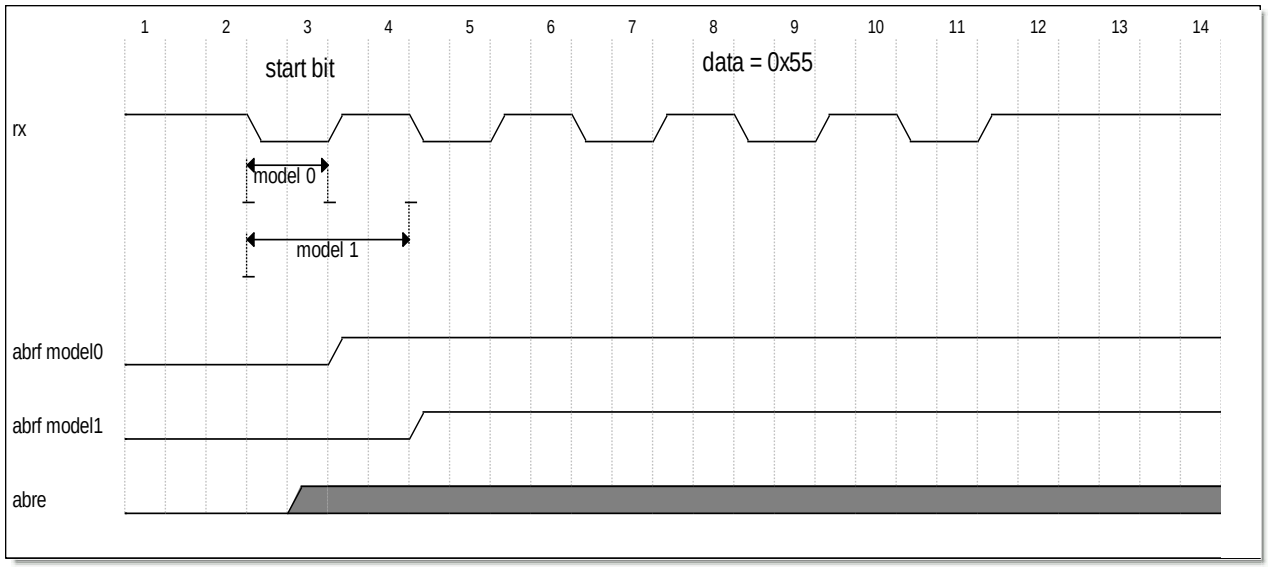


图 15.11 自动波特率检测时序图

使用波特率检测功能，首先使能 ABREN，然后根据自己使用的检测模式配置 ABRM，读取 ABRF 是否为 1（上次使用过后未清零），如果为 1，则写零清零；然后开始接收数据，波特率检测完成后 ABRF 就会置 1，在 ABRF 置 1 后，不要立即清零 ABRF，因为清理 ABRF 会立即在当前传输的位置（可能已经不是起始比特的位置）进行波特率检测，这样会导致错误的结果；当前数据接收完成后会产生 RXNEF 标志位，然后可以清零 ABRF，开始下一次波特率的检测，假如不清零 ABRF 标志位，则下次接收数据时不会启动波特率检测；如果波特率检测超出了范围则会产生 ABRE 标志位 ，表示波特率检测出错。

波特率检测完成后，如果后续还需要检测波特率则不需要立即清零 ABRF，只有在需要再次检测的时候清零 ABRF 即可。

需要注意的是波特率检测的数据是用来配置 DLL/DLH 寄存器用的，如果发送方波特率数据不靠近 Fbraudrate=Fmaster/(16*{DLH,DLL})，则波特率检测模块会自动配置本机为靠近支持的波特率，串口模块并不支持小数波特率，因此该模块的波特率检测存在误差。

15.3. 与 USART 相关寄存器汇总

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	复位值
URDATAL	0x48C	DATAL[7:0]								0000 0000
URDATAH	0x48D	—							DATAH	---- ---0
URIER	0x48E	—		TCEN	—	IDELE	RXSE	URTE	URRXNE	--0- 0000
URLCR	0x48F	—	BKREQ	—	EVEN	PEN	URSTOP	—	LTH	-0-0 00-0
URLCREXT	0x490	—						RWU	EXTEN	---- --00
URMCR	0x491	—		SIRLP	TXEN	RXEN	WAKE	HDSEL	SIREN	---0 0000
URLSR	0x492	ADDRF	IDLEF	TXEF	BKF	FEF	PEF	OVERF	RXNEF	0000 0000
URRAR	0x493	—				RAR[3:0]				---- 0000
URDLL	0x494	DLL[7:0]								0000 0000
URDLH	0x495	DLH[7:0]								0000 0000
URABCR	0x496	—				ABRE	ABRM	ABRF	ABREN	---- 0000
URSYNCR	0x497	—				LBCL	URCPHA	URCPOL	SYNEN	---- 0000
URLINCR	0x498	—			LINEN	BLTH[3:0]				---0 0000
URSDCR0	0x499	—	NACK	CKOE	SDEN	—				-000 ----
URSDCR1	0x49A	GT[7:0]								0000 0000
URSDCR2	0x49B	PSC[7:0]								0000 0000
URTC	0x49C	—							TCF	---- ---1

15.3.1. URDATAL 寄存器，地址 0x48C

Bit	7:0
Name	DATAL
Reset	0x00
Type	RW

Bit	Name	Function
7:0	DATAL	数据发送/接收寄存器低八位，请使用字节传输指令对该寄存器进行操作

15.3.2. URDATAH 寄存器，地址 0x48D

Bit	7:1	0
Name	—	DATAH
Reset	—	0x0
Type	RO-0	RW

Bit	Name	Function
7:1	N/A	保留位，读 0
0	DATAH	数据发送/接受寄存器高八位，这一位只有启用 9 比特数据格式才有用，这种模式下，一定要先写低八位再写高 1 位。 1：表示 DATAL 是地址 0：表示 DATAL 是数据

15.3.3. URIER 寄存器，地址 0x48E

Bit	7:6	5	4	3	2	1	0
Name	—	TCEN	—	IDELE	RXSE	URTE	URRXNE
Reset	—	0x0	—	0x0	0x0	0x0	0x0
Type	RO-0	RW	RO-0	RW	RW	RW	RW

Bit	Name	Function
7:6	N/A	保留位，读 0
5	TCEN	发送完成中断使能 1：使能发送完成中断 0：禁用发送完成中断
4	N/A	-保留位，读 0
3	IDELE	空闲帧中断使能 1：使能空闲帧中断 0：禁用空闲帧中断
2	RXSE	接收状态使能，包括帧断开，帧错误，奇偶校验错误，接收溢出错误 1：使能接收到断开帧/帧错误/奇偶校验错误/接收溢出错误状态中断 0：禁用状态信息中断
1	URTE	发送 buf 为空中断使能 1：使能发送为空中断 0：禁用发送为空中断
0	URRXNE	接受到数据中断使能 1：使能接收到数据中断 0：禁用接收到数据中断

15.3.4. URLCR 寄存器，地址 0x48F

Bit	7	6	5	4	3	2	1	0
Name	—	BKREQ	—	EVEN	PEN	URSTOP	—	LTH
Reset	—	0x0	—	0x0	0x0	0x0	—	0x0
Type	RO-0	RW	RO-0	RW	RW	RW	RO-0	RW

Bit	Name	Function
7	N/A	保留位，读 0
6	BKREQ	发送断开帧使能，发送完成后改位自动清零，断开帧发送过程中不应该写零该位；发送断开帧之前请先设置断开帧的长度，在 lincr 寄存器的 lth 字段 1：请求发送断开帧/断开帧发送中 0：未请求发送断开帧/断开帧发送完成
5	N/A	保留位，读 0
4	EVEN	置 1 表示偶检验使能，置零表示奇校验使能 1：表示使用偶检验 0：表示使用奇校验
3	PEN	校验位使能 1：使能校验位 0：禁用校验位
2	URSTOP	停止位长度设定 0：表示 1 个停止位 1：智能卡模式启用时表示 1.5 个停止位，否则表示两个停止位
1	N/A	保留位，读 0
0	LTH	通信数据长度设定 0：表示数据长度是 7 位，此位不包含检验位长度 1：表示数据长度是 8 位，此位不包含检验位长度

15.3.5. URLCREXT 寄存器，地址 0x490

Bit	7:2	1	0
Name	—	RWU	EXTEN
Reset	—	0x0	0x0
Type	RO-0	RW	RW

Bit	Name	Function
7:2	N/A	保留位，读 0
1	RWU	多处理器模式接收唤醒 1：设置进入哑模式 0：未设置进入哑模式或者已经退出哑模式
0	EXTEN	发送的数据是否时 9 比特长度 1：发送的数据长度是 9 比特长度 0：发送的数据不是 9 比特长度

15.3.6. URMCR 寄存器，地址 0x491

Bit	7:6	5	4	3	2	1	0
Name	—	SIRLP	TXEN	RXEN	WAKE	HDSEL	SIREN
Reset	—	0x0	0x0	0x0	0x0	0x0	0x0
Type	RO-0	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:6	N/A	保留位，读 0
5	SIRLP	红外低功耗模式使能 1：使能红外低功耗模式 0：禁用红外低功耗模式
4	TXEN	发送使能 1：使能接口的发送，相应的 IO 会被用作 TX 引脚 0：禁用接口的发送
3	RXEN	接收使能 1：允许接口接收，相应的 IO 会被用作 RX 引脚 0：禁用接口接收
2	WAKE	哑模式唤醒方式选择 1：选择地址匹配 0：选择 IDLE 帧
1	HDSEL	半双工使能 1：使能半双工模式 0：禁用半双工模式
0	SIREN	红外模式使能 1：使能红外模式 0：禁用红外模式

15.3.7. URLSR 寄存器，地址 0x492

Bit	7	6	5	4	3	2	1	0
Name	ADDRF	IDLEF	TXEF	BKF	FEF	PEF	OVERF	RXNEF
Reset	0x0	0x0	0x1	0x0	0x0	0x0	0x0	0x0
Type	RO	W0	RO	W0	W0	W0	W0	RO

Bit	Name	Function
7	ADDRF	哑模式地址匹配标志位，指示位，软件不可清 1：哑模式地址匹配唤醒模式中，匹配到了地址 0：哑模式地址匹配唤醒模式中，未匹配到地址
6	IDLEF	空闲帧标志，写 0 清 0，写 1 无效 1：检测到空闲帧 0：未检测到空闲帧
5	TXEF	发送寄存器的状态，在启用了 9 比特数据格式时，写 DATAH 寄存器清零，否则写取 DATAL 寄存器清零 1：发送寄存器为空 0：发送寄存器不为空
4	BKF	断开帧标志，写 0 清 0，写 1 无效 1：接收到了断开帧 0：未接收到断开帧或已清零
3	FEF	帧错误标志，写 0 清 0，写 1 无效 1：接收出现了帧错误 0：未接收到帧错误会已清零
2	PEF	奇偶校验错误标识，写 0 清 0，写 1 无效零 1：接收到了奇偶校验错误 0：未接收到奇偶校验错误或已清零
1	OVERF	溢出错误标志，写 0 清 0，写 1 无效 1：接收寄存器出现溢出 0：接收未出现溢出或已清零
0	RXNEF	在启用 9 比特数据格式时，读取 DATAH 寄存器清零，否则读取 DATAL 寄存器清零 1：接收寄存器非空 0：接收寄存器为空或已清零

15.3.8. URRAR 寄存器，地址 0x493

Bit	7:4	3:0
Name	—	Receive address
Reset	—	0x0
Type	RO-0	RW

Bit	Name	Function
7:4	N/A	保留位，读 0
3:0	Receive address	多处理器模式中的本机地址

15.3.9. URDLL 寄存器，地址 0x494

Bit	7:0
Name	DLL
Reset	0x0
Type	RW

Bit	Name	Function
7:0	DLL	波特率分频计数器低八位

15.3.10. URDLH 寄存器，地址 0x495

Bit	7:0
Name	DLH
Reset	0x0
Type	RW

Bit	Name	Function
7:0	DLH	波特率分频计数器高八位 波特率=Fmaster/(16*{DLH,DLL})，默认值{DLH，DLL}为 0x0000 时，串口不工作； {DLH，DLL}最小值为 0x0001

15.3.11. URABCR 寄存器，地址 0x496

Bit	ABCR	3	2	1	0
Name	—	ABRE	ABRM	ABRF	ABREN
Reset	—	0x0	0x0	0x0	0x0
Type	RO-0	RW	RW	RW	RW

Bit	Name	Function
7:4	N/A	保留位，读 0
3	ABRE	波特率检测溢出 1：波特率检测超出范围 0：波特率检测未超出范围
2	ABRM	波特率检测模式 0：只检测起始位长度；这种模式要求第一比特为 1 1：检测起始位和数据第一位的长度，然后除以 2；这种模式要求第一比特为 1，第二比特为零
1	ABRF	检测到波特率标识，写零清零；该位在清零后，会立即再次进入波特率检测，为了保证每次检测的都是起始位，可以在 RXNEF 置位后，再清零 1：检测到波特率 0：未检测到波特率
0	ABREN	自动波特率检测使能 1：使能波特率检测功能 0：禁用波特率检测功能

15.3.12. URSYNCR 寄存器，地址 0x497

Bit	7:4	3	2	1	0
Name	—	LBCL	URCPHA	URCPOL	SYNEN
Reset	—	0x0	0x0	0x0	0x0
Type	RO-0	RW	RW	RW	RW

Bit	Name	Function
7:4	N/A	保留位，读 0
3	LBCL	同步模式最后一个比特时钟使能 1：同步模式中最后一比特的时钟输出 0：同步模式中最后一比特的时钟不输出
2	URCPHA	同步模式时钟相位设置 1：主机在时钟变化的第二个沿开始采样第一个数据 0：主机在时钟变化的第一个沿开始采样第一个数据
1	URCPOL	同步模式时钟极性设置 1：同步模式中时钟空闲时为高电平 0：同步模式中时钟空闲时为低电平
0	SYNEN	同步模式使能 1：使能同步传输模式，相应的 IO 会用作同步时钟输出 0：禁用时钟同步模式

15.3.13. URLINCR 寄存器，地址 0x498

Bit	7:5	4	3:0
Name	—	LINEN	BLTH
Reset	—	0x0	0x0
Type	RO-0	RW	RW

Bit	Name	Function
7:5	N/A	保留位，读 0
4	LINEN	Lin 模式使能 1：使能 LIN Master 模式 0：禁用 LIN Master 模式
3:0	BLTH	断开帧比特长度，大于零有效，一般设置为 12/13 比特长度，设置太短会被认为接收到的是正常的帧

15.3.14. URSDCR0 寄存器，地址 0x499

Bit	7	6	5	4	3:0
Name	—	NACK	CKOE	SDEN	—
Reset	—	0x0	0x0	0x0	—
Type	RO-0	RW	RW	RW	RO-0

Bit	Name	Function
6	NACK	智能卡回复 Nack 使能 1：使能检测到奇偶校验出错时，发送 NACK 0：检测到奇偶校验位错误时不发送 NACK
5	CKOE	给智能卡提供时钟时的使能 1：使能时钟输出，请配置 PSC 寄存器的值为有效值 0：禁止时钟输出
4	SDEN	智能卡模式使能，启用智能卡模式必须设定停止位为 1.5 位 1：使能智能卡模式 0：禁用智能卡模式
3:0	N/A	保留位，读 0

15.3.15. URSDCR1 寄存器，地址 0x49A

Bit	7:0
Name	GT
Reset	0x0
Type	RW

Bit	Name	Function
7:0	GT	智能卡模式:两字符之间波特间隔，最小为 1，即使设置为 0，也有一个间隔

15.3.16. URSDCR2 寄存器，地址 0x49B

Bit	7:0
Name	PSC
Reset	0x0
Type	RW

Bit	Name	Function
7:0	PSC	-给智能卡提供时钟时的时钟分频系数 0：无效 1：2 分频 2：3 分频 3：4 分频 …… -红外低功耗模式系统时钟分频 0:无效 1：1 分频 2：2 分频 3：3 分频 ……

15.3.17. URTC 寄存器，地址 0x49C

Bit	7:1	0
Name	—	TCF
Reset	—	0x1
Type	RO-0	RW

Bit	Name	Function
7:1	N/A	保留位，读 0
0	TCF	发送完成状态标志 1：数据发送完成 0：数据发送未完成，写 1 清零或写 DATAL/DATAH 寄存器后清零（在使能了 9 比特数据格式时，写 DATAH 寄存器后清零，否则写 DATAL 寄存器后清零）

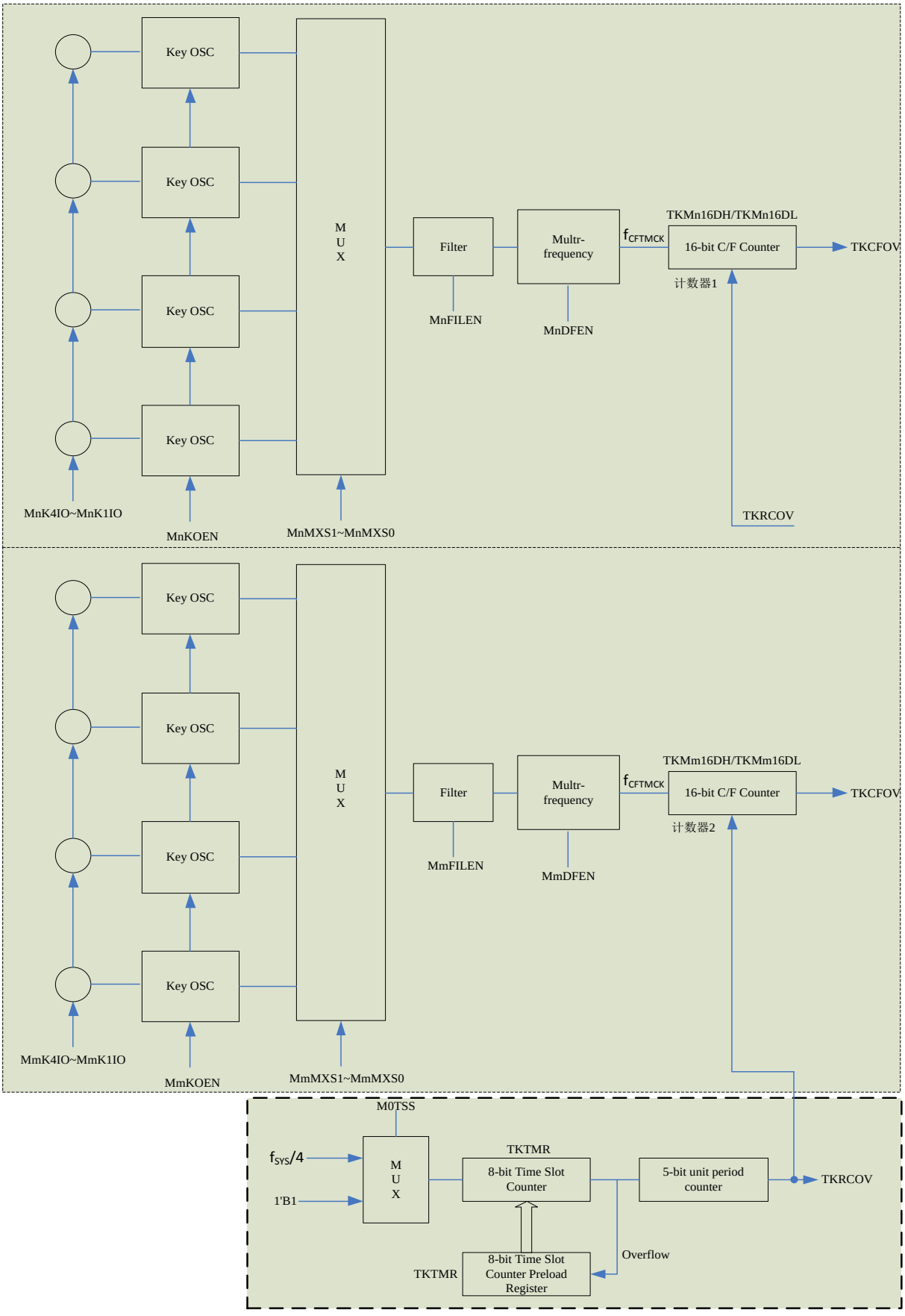
16. 电容按键模块

16.1. 触摸按键功能

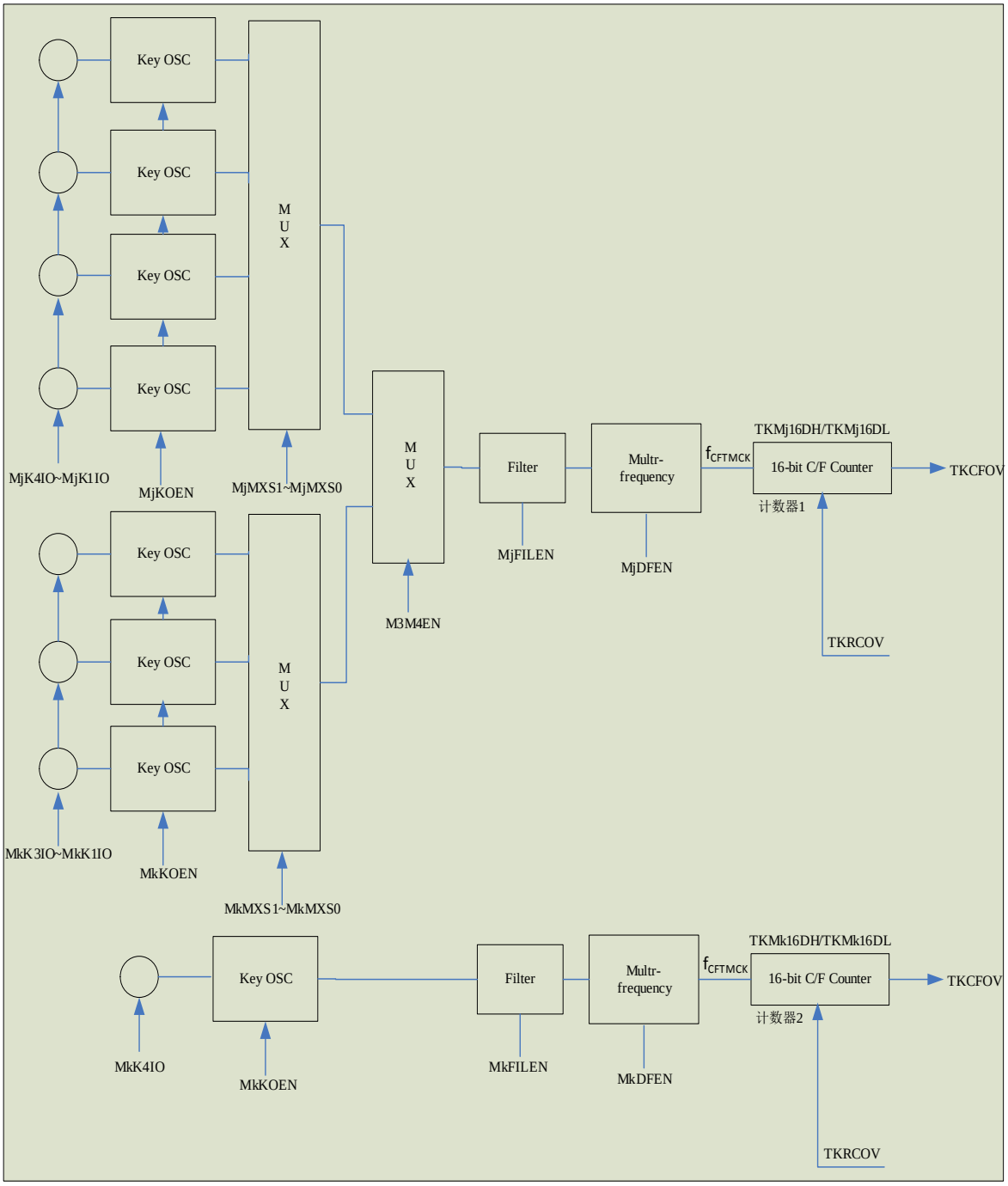
该系列单片机均提供多触控按键功能，触控按键功能完全内部集成只需外接少量元件，可通过对内部寄存器的简单操作来实现此功能。

16.2. 触摸按键结构

触控按键与 I/O 引脚共用，通过相应选择寄存器的位来选择此功能。按键被分成两组，每组为一个模块，编号为 M0~M1。每个模块为独立的一组，包含八个触控按键且每一个按键使用一个振荡器。每个模块具有单独的控制逻辑电路和配套的寄存器系列。M0 的 key1-key4 与 key5-key8 的控制逻辑电路和配套的寄存器系列不同。M1 的 key9-key15 与 key16 的控制逻辑电路和配套的寄存器系列不同。



模块 0 触控按键功能方框图（n=0 m=1）



模块 1 触控按键功能方框图（j=2 k=3）

注：

1. 虚线方框内时隙计数器部分为公共部分；
2. .当M0TSS=0时选择的时钟为fSYS/4，时隙计数器才能正常计数。

16.3. 触摸按键操作

手指接近或接触到触控面板时，面板的电容量会增大，电容量的变化会轻微改变内部感应振荡器的频率，通过测量频率的变化可以感知触控动作。参考时钟通过内部可编程分频器能够产生一个固定的时间周期。在这个时间周期内，通过在此固定时间周期内对感应振荡器产生的时钟周期计数，可确定触控按键的动作。

每个触控按键模块包含8个与I/O引脚共用的触控按键，通过寄存器可设置相应引脚功能。每个触控按键用一个感应振荡器，因此每个模块包含八个感应振荡器。

在参考时钟固定的时间间隔内，感应振荡器产生的时钟周期数是可以测量的。测到的周期数可以用于判断触控动作是否有效发生。在此固定的时间间隔结束后，会产生一个触控按键中断信号。

所有模块共用同一个时隙计数器，且所有触控按键模块共用一个起始信号，即 TKC0 寄存器中的 TKST。在此位清零时，所有模块的 16-bit C/F 计数器、触控按键功能 16-bit 计数器和 5-bit 时隙单位周期计数器会自动清零，而 8-bit 可编程时隙计数器不清零，由用户设置溢出时间。在 TKST 位由低变高时，16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器会自动开启。

当时隙计数器溢出，所有模块的按键振荡器都会自动停止且 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器也会自动停止。时隙计数器时钟源可通过 TKMnC1 寄存器中的 M0TSS 位选择来自参考振荡器或 fSYS/4。通过设置 TKMnC1 寄存器中的 MnROEN 位和 MnKOEN 为“1”，可使能参考振荡器和按键振荡器。

当时隙计数器溢出时，将产生一个触控按键中断。这里所有的触控按键是指已使能的触控按键。

每 8 个按键为一个模块，所以 KEY1~KEY8 为模块 0，KEY9~KEY16 为模块 1，模块 0 和模块 1 的架构不相同。

16.4. 触摸按键中断

触控按键只有一个中断，当时隙计数器溢出时将置位中断标志位，这里所有的触控按键是指已使能的触控按键。此时所有模块的16-bit C/F计数器、16-bit计数器、5-bit时隙单位周期计数器和8-bit时隙计数器会自动清零。

任何一个触控按键模块的16-bit C/F计数器溢出就会把16-bit C/F计数器溢出标志位TKCFOV置高。由于此标志位无法被自动清零，需通过应用程序将此位清零。16-bit计数器溢出就会把其溢出标志位TK16OV置高。由于此标志位无法被自动清零，需通过应用程序将此位清零。

16.5. 编程注意事项

相关寄存器设置后，将TKST位由低电平变为高电平会启动触控按键检测程序。此时所有相关的振荡器将使能并同步。当计数器溢出时，时隙计数器标志位TKRCOV将变为高电平，同时还会产生一个中断信号。当外部触控按键的大小和布局确定时，其相关的电容将决定感应振荡器的频率。

16.6. 工作模式

16.6.1. 单次扫描

使用默认的 TKC1 寄存器设置可以实现单次扫描的功能。TKST 从 0 到 1 跳变后触控按键模块 16-bit C/F 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器从 0 开始计数。当时隙计数器溢出，则 TKRCOV 位及触控按键中断请求标志位 TKMF 将会被置位且所有模块按键振荡器将自动停止。

16.6.2. 多次扫描

配置 TKC1 寄存器 bit[6:5]可以实现 n 次扫描的功能。TKST 从 0 到 1 跳变后触控按键模块 16-bit C/F 计数器从 0 开始计数、5-bit 时隙单位周期计数器从 0 开始计数、8-bit 时隙计数器从加载的 TKTMR 的值开始计数。当时隙计数器溢出 n 次时，则 TKRCOV 位及触控按键中断请求标志位 TKMF 将会被置位且所有模块按键振荡器将自动停止。

16.6.3. 4 段跳频

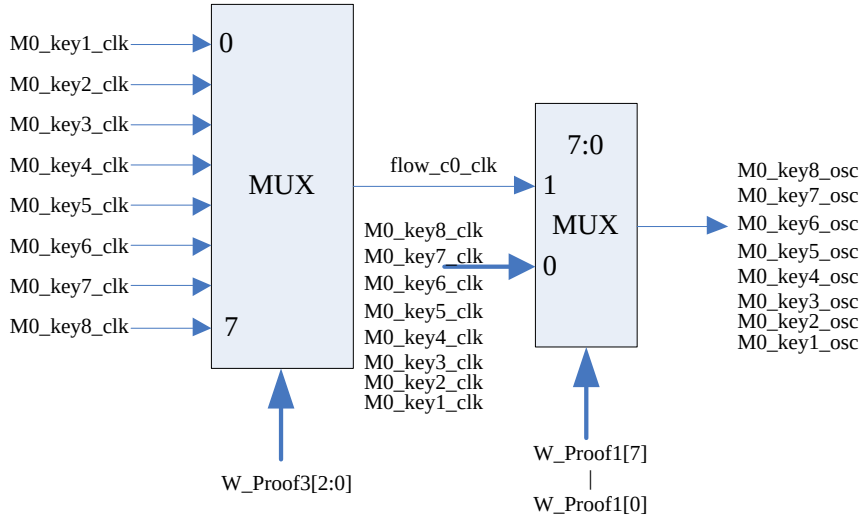
配置 TKM0C0 、TKM1C0、 TKM2C0 、TKM3C0 寄存器的 bit[3]位可以实现软件跳频或者硬件自动跳频功能，在硬件自动跳频功能下，模块 0 的 key1-key4、key5-key8 模块 1 的 key9-key15、key16 实现 4 段自动跳频配置，自动跳频配置如下：

	第 1 段	第 2 段	第 3 段	第 4 段
M0SOF[1:0]	3	1	2	0
M1SOF[1:0]	1	3	0	2
M2SOF[1:0]	2	0	3	1
M3SOF[1:0]	0	2	1	3

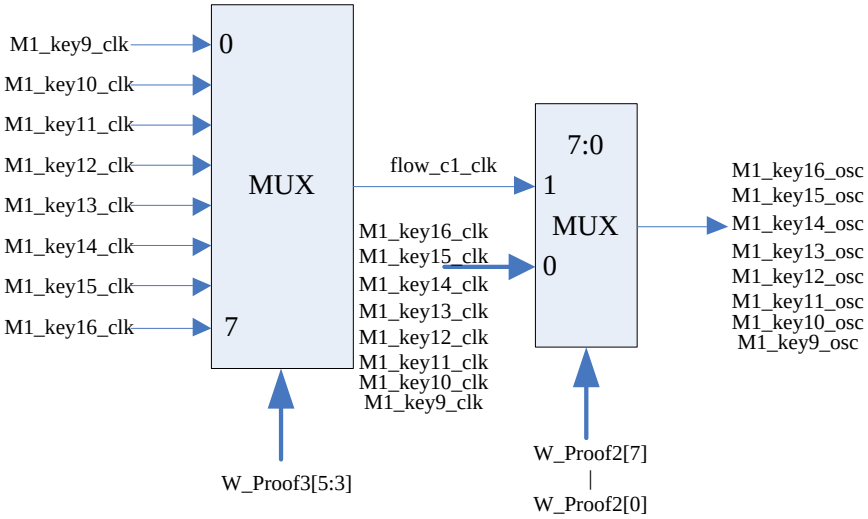
16.6.4. 防水模式

在按键防水使能开启的时候，跟随的按键时钟将选择被跟随的按键时钟进行计数。

在防水的情况下，按键被分成 2 组进行防水跟随。Key1-Key8 为一组，Key9-Key16 为一组。例如当防水使能 Wproof1[7]为 1 时，第八个按键时钟输入不是自己的振荡器产生的时钟而是 Wproof3[2:0]选择的时钟源，如果 Wproof3[2:0]为 0，那么第八个按键时钟输入为第一个按键振荡器产生的时钟。防水跟随示意图如下。



防水跟随图一



防水跟随图二

注意：如果按键自己防水，则不能跟随自己。

16.7. 与 TOUCH 相关寄存器汇总

寄存器名称	说明
TKTMR	触控按键时隙 8-bit 计数器预载寄存器
TKC0	触控按键功能控制寄存器 0
TKC1	触控按键功能控制寄存器 1
TKMnC0	触控按键模块控制寄存器 0
TKMnC1	触控按键模块控制寄存器 1
Wproof1	KEY8~KEY1 的防水使能控制 bits
Wproof2	KEY15~KEY9 的防水使能控制 bits
Wproof3	防水 8 选 1 控制 bits
Mnanalog	按键模拟配置位
CFnOUT1L	触控按键模块 16-bit C/F 计数器第一次计数值低 8 位的值
CFnOUT1H	触控按键模块 16-bit C/F 计数器第一次计数值高 8 位的值
CFnOUT2L	触控按键模块 16-bit C/F 计数器第二次计数值低 8 位的值
CFnOUT2H	触控按键模块 16-bit C/F 计数器第二次计数值高 8 位的值
CFnOUT3L	触控按键模块 16-bit C/F 计数器第三次计数值低 8 位的值
CFnOUT3H	触控按键模块 16-bit C/F 计数器第三次计数值高 8 位的值
TKMn16DL	触控按键模块 16-bit C/F 计数器第 n 次计数值低 8 位的值
TKMn16DH	触控按键模块 16-bit C/F 计数器第 n 次计数值高 8 位的值

每个触控按键模块包含八个触控按键功能，且都有其配套的寄存器。以下表格显示了每个触控按键模块的寄存器系列。寄存器名称里的 Mn 当 n =0 时代表 key1-key4，当 n=1 时代表 key5-key8，当 n=2 时代表 key9-key15，当 n=3 时代表 key16。一共有两个触控按键模块，key1-key8 为模块 0；key9-key16 为模块 1。

名称	地址	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	复位值
TKTMR	0x38C	TKTMR[7:0]								0000 0000
TKC0	0x38D	—	TKRCOV	TKST	TKCFOV	—	TDMY2	TDMY1	TDMY0	-000 -000
TKC1	0x38E	—	SEG[1:0]		—	—	TDMY3	option1	option0	-00- -000
Wproof1	0x38F	KEY8	KEY7	KEY6	KEY5	KEY4	KEY3	KEY2	KEY1	0000 0000
Wproof2	0x390	—	KEY15	KEY14	KEY13	KEY12	KEY11	KEY10	KEY9	0000 0000
Wproof3	0x391	TDMY5	TDMY4	MX5	MX4	MX3	MX2	MX1	MX0	0000 0000
M0analog	0x392	M0VIHVIL[2:0]			M0FILCS[2:0]			M0RSET[1:0]		0000 0100
M1analog	0x393	M1VIHVIL[2:0]			M1FILCS[2:0]			M1RSET[1:0]		0000 0100
M2analog	0x394	M2VIHVIL[2:0]			M2FILCS[2:0]			M2RSET[1:0]		0000 0100
M3analog	0x395	M3VIHVIL[2:0]			M3FILCS[2:0]			M3RSET[1:0]		0000 0100
TKM0C0	0x396	M0MXS1	M0MXS0	M0DFEN	M0FILEN	M0SOFC	—	M0SOF1	M0S0F0	0000 0-00
TKM0C1	0x397	M0TSS	—	—	M0KOEN	M0K4IO	M0K3IO	M0K2IO	M0K1IO	0—0 0000
TKM1C0	0x398	M1MXS1	M1MXS0	M1DFEN	M1FILEN	M1SOFC	—	M1SOF1	M1S0F0	0000 0-10
TKM1C1	0x399	TDMY6	—	—	M1KOEN	M1K4IO	M1K3IO	M1K2IO	M1K1IO	0—0 0000
TKM2C0	0x39A	M2MXS1	M2MXS0	M2DFEN	M2FILEN	M2SOFC	—	M2SOF1	M2S0F0	0000 0—0
TKM2C1	0x39B	M3M4EN	—	—	M2KOEN	M2K4IO	M2K3IO	M2K2IO	M2K1IO	0—0 0000
TKM3C0	0x39C	M3MXS1	M3MXS0	M3DFEN	M3FILEN	M3SOFC	—	M3SOF1	M3S0F0	0000 0-11
TKM3C1	0x39D	TDMY7	—	—	M3KOEN	M3K4IO	M3K3IO	M3K2IO	M3K1IO	0—0 0000
CF0OUT1L	0xF8E	CF0OUT1[7:0]								0000 0000
CF0OUT1H	0xF8F	CF0OUT1[15:8]								0000 0000
CF0OUT2L	0xF90	CF0OUT2[7:0]								0000 0000
CF0OUT2H	0xF91	CF0OUT2[15:8]								0000 0000
CF0OUT3L	0xF92	CF0OUT3[7:0]								0000 0000
CF0OUT3H	0xF93	CF0OUT3[15:8]								0000 0000
TKM016DL	0xF94	TKM016D[7:0]								0000 0000
TKM016DH	0xF95	TKM016D[15:8]								0000 0000
CF1OUT1L	0xF96	CF1OUT1[7:0]								0000 0000
CF1OUT1H	0xF97	CF1OUT1[15:8]								0000 0000
CF1OUT2L	0xF98	CF1OUT2[7:0]								0000 0000
CF1OUT2H	0xF99	CF1OUT2[15:8]								0000 0000
CF1OUT3L	0xF9A	CF1OUT3[7:0]								0000 0000
CF1OUT3H	0xF9B	CF1OUT3[15:8]								0000 0000
TKM116DL	0xF9C	TKM116D[7:0]								0000 0000
TKM116DH	0xF9D	TKM116D[15:8]								0000 0000
CF2OUT1L	0xF9E	CF12UT1[7:0]								0000 0000
CF2OUT1H	0xF9F	CF12UT1[15:8]								0000 0000
CF2OUT2L	0xFA0	CF12UT2[7:0]								0000 0000
CF2OUT2H	0xFA1	CF12UT2[15:8]								0000 0000
CF2OUT3L	0xFA2	CF12UT3[7:0]								0000 0000
CF2OUT3H	0xFA3	CF12UT3[15:8]								0000 0000
TKM216DL	0xFA4	TKM216D[7:0]								0000 0000
TKM216DH	0xFA5	TKM216D[15:8]								0000 0000
CF3OUT1L	0xFA6	CF3OUT1[7:0]								0000 0000
CF3OUT1H	0xFA7	CF3OUT1[15:8]								0000 0000
CF3OUT2L	0xFA8	CF3OUT2[7:0]								0000 0000
CF3OUT2H	0xFA9	CF3OUT2[15:8]								0000 0000
CF3OUT3L	0xFAA	CF3OUT3[7:0]								0000 0000
CF3OUT3H	0xFAB	CF3OUT3[15:8]								0000 0000
TKM316DL	0xFAC	TKM316D[7:0]								0000 0000
TKM316DH	0xFAD	TKM316D[15:8]								0000 0000

16.7.1. TKTMR 寄存器，地址 0x038C

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	D7-D0	触控按键时隙 8-bit 计数器预载寄存器。触控按键时隙计数器预载寄存器用于确定触控按键时隙溢出时间。时隙单位周期通过一个 5-bit 计数器获得，等于 32 个时隙时钟周期。因此，时隙计数器溢出时间可由下面的等式算出。时隙计数器溢出时间=(256-TKTMR[7:0])×32tTSC，其中 tTSC 为时隙计数器时钟周期。

16.7.2. TKC0 寄存器，地址 0x038D

it	7	6	5	4	3	2	1	0
Name	—	TKRCOV	TKST	TKCFOV	—	TDMY[2:0]		
Reset	—	0	0	0	—	0	0	0
Type	RO-0	R/W	R/W	R/W	RO-0	R/W	R/W	R/W

Bit	Name	Function
7	N/A	保留位，读为”0“
6	TKRCOV	触控按键时隙计数器溢出标志位。 0：无溢出 1：溢出 此位可通过应用程序读/ 写。当触控按键时隙计数器溢出将此位置为“1”时，相应的触控按键中断请求标志位也会同时置位。然而若是通过应用程序将此位设置为“1”时，相应的中断请求标志位不会受到影响。因此，此位不能通过应用程序置位但必须通过应用程序清零。 若时隙计数器溢出，则 TKRCOV 位及触控按键中断请求标志位 TKMF 将会被置位且所有模块按键振荡器停止。此时触控按键模块 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器都会自动关闭。
5	TKST	触控按键检测开启控制位 0：停止或无操作 0 → 1：开始检测 当该位为“0”时，所有模块的 16-bit C/F 计数器、触控按键功能 16-bit 计数器和 5-bit 时隙单位周期计数器会自动清零，但 8-bit 可编程时隙计数器不会被清零。当该位由 0 到 1 转变时，触控按键模块 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器都会自动开启，并使能按键振荡器以驱动相应的计数器。
4	TKCFOV	触控按键模块 16-bit C/F 计数器溢出标志位。 0：无溢出 1：溢出 该位由触控按键模块 16-bit C/F 计数器溢出置位，必须通过应用程序清零。
3	N/A	保留位，读为”0“
2:0	TDMY[2:0]	保留寄存器

16.7.3. TKC1 寄存器，地址 0x038E

Bit	7	6	5	4	3	2	1	0
Name	—	SEG[1:0]		—	—	TDMY3	option1	option0
Reset	—	0	0	—	—	0	0	0
Type	RO-0	R/W	R/W	RO-0	RO-0	R/W	R/W	R/W

Bit	Name	Function
7	N/A	保留位，读为”0“。
6:5	SEG[1:0]	选择时隙计数器计数几次。 00：时隙计数器计数 1 次。 01：时隙计数器计数 2 次。 10：时隙计数器计数 3 次。 11：时隙计数器计数 4 次。
4:3	N/A	保留位，读为”0“。
2	TDMY3	保留寄存器。
1:0	option1~ option0	当滤波使能开启时,Filter 模块的滤毛刺大小选择位。 00:0.5ns 01:3.5ns 10:6.5ns 11:9.5ns

16.7.4. Wproof1 寄存器，地址 0x038F

Bit	7	6	5	4	3	2	1	0
Name	KEY8	KEY7	KEY6	KEY5	KEY4	KEY3	KEY2	KEY1
Reset	0	0	0	0	0	0	0	0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7:0	KEY8-KEY1	KEY7~KEY0 的防水使能控制 bits。 1：对应 key 的输出信号跟随选中的 8 选 1 的输出信号。 0：为自己对应 key 的振荡器输出。

16.7.5. Wproof2 寄存器，地址 0x0390

Bit	7	6	5	4	3	2	1	0
Name	—	KEY15	KEY14	KEY13	KEY12	KEY11	KEY10	KEY9
Reset	0	0	0	0	0	0	0	0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	N/A	保留位
6:0	KEY15-KEY9	KEY15~KEY8 的防水使能控制 bits。 1：对应 key 的输出信号跟随选中的 8 选 1 的输出信号。 0：为自己对应 key 的振荡器输出。

16.7.6. Wproof3 寄存器，地址 0x0391

Bit	7	6	5	4	3	2	1	0
Name	TDMY5	TDMY4	MX5	MX4	MX3	MX2	MX1	MX0
Reset	0	0	0	0	0	0	0	0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7:3	TDMY[5:4]	保留寄存器。
5:3	MX5-MX3	Key15-key9 的 8 选 1 控制 bits 000: key9 001: key10 …… 110: key15 **111: key16（内部参考）**
2:0	MX2-MX0	Key8-key1 的 8 选 1 控制 bits 000: key1 001: key2 …… 110: key7 111: key8

16.7.7. Mnanalog 寄存器，地址 0x0392+n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	MnVIHVIL[2:0]			MnFILCS[2:0]			MnRSET[1:0]	
Reset	0	0	0	0	0	1	0	0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7:5	MnVIHVIL[2:0]	n=0 key1-4 的施密特 Vih\Vil 选择控制 bits n=1 key5-8 的施密特 Vih\Vil 选择控制 bits n=2 key9-15 的施密特 Vih\Vil 选择控制 bits n=3 key16 的施密特 Vih\Vil 选择控制 bits
4:2	MnFILCS[2:0]	n=0 key1-4 的 filter Cap 选择控制 bits n=1 key5-8 的 filter Cap 选择控制 bits n=2 key9-15 的 filter Cap 选择控制 bits n=3 key16 的 filter Cap 选择控制 bits
1:0	MnRSET[1:0]	n=0 key1-4 的 RES 选择控制 bits n=1 key5-8 的 RES 选择控制 bits n=2 key9-15 的 RES 选择控制 bits n=3 key16 的 RES 选择控制 bits

16.7.8. TKMnC0 寄存器，地址 0x0396+2n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	MnMXS1	MnMXS0	MnDFEN	MnFILE	MnSOFC	—	MnSOF1	MnSOF0
Reset	0	0	0	0	0	—	0	0
Type	R/W	R/W	R/W	R/W	R/W	RO-0	R/W	R/W

Bit	Name	Function
7:6	MnMXS1~ MnMXS0	MnMXS1~MnMXS0：多路复用按键选择。 00:KEYOSC1 01:KEYOSC2 10:KEYOSC3 11:KEYOSC4
5	MnDFEN	触控按键模块倍频功能控制位 0：除能 1：使能 此位用于控制触控按键振荡器的倍频功能。当此位置 1，按键振荡器频率将为原来的两倍。
4	MnFILE	触控按键模块滤波器功能控制位 0：除能 1：使能
3	MnSOFC	触控按键模块 C/F 振荡器跳频功能控制位 0：由 MnSOF1~MnSOF0 位控制 1：由硬件电路控制 该位用来选择触控按键振荡器跳频功能控制方式。当此位置 1，按键振荡器跳频功能由硬件电路控制，MnSOF1~MnSOF0 位的设置无效。
2	N/A	保留位，读出为 0。
1:0	MnSOF1~ MnSOF0	触控按键模块按键振荡器跳频选择位,内部等效电容=20pF 00：0.821MHz 01：0.841MHz 10：0.860MHz 11：0.876MHz 这些位用于触控按键振荡器跳频功能的频率选择。注意，只有当 MnSOFC 位为零时，这些位的选择才有效。上述频率会随着外部或内部电容值的不同而变化。若触控按键振荡器频率选择 1MHz，用户选择其它频率时可依比例调整。

16.7.9. TKMnC1 寄存器，地址 0x0397+2n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	MnTSS	—	—	MnKOEN	MnK4IO	MnK3IO	MnK2IO	MnK1IO
Reset	0	—	—	0	0	0	0	0
Type	R/W	RO-0	RO-0	R/W	R/W	R/W	R/W	R/W

Bit	Name	Function
7	MnTSS	n=0 为 M0TSS　0：时隙计数器计数使能。 1：时隙计数器计数不使能。 N=1 为保留寄存器 TDMY6 n=2 为 M3M4EN　0：按键 9、10、11、12 的时钟给 M1 模块的第一个计数器。 1：按键 13、14、15、16 的时钟给 M1 模块的第一个计数器。 N=3 为保留寄存器 TDMY7
6:5	N/A	保留位，读为“0”
4	MnKOEN	触控按键模块按键振荡器使能控制位 0：除能 1：使能
3	MnK4IO	触控按键模块 Key 4 使能控制。 0：除能，I/O 或其它功能 1：使能，KEY4
2	MnK3IO	触控按键模块 Key 3 使能控制。 0：除能，I/O 或其它功能 1：使能，KEY3
1	MnK2IO	触控按键模块 Key 2 使能控制。 0：除能，I/O 或其它功能 1：使能，KEY2
0	MnK1IO	触控按键模块 Key 1 使能控制。 0：除能，I/O 或其它功能 1：使能，KEY1

16.7.10. CFnOUT1L 寄存器，地址 0x0F8E+8n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	D7-D0	启动一次 TKST 计数配置的计数次数大于 1，该寄存器用于存储触控按键四个 16-bit C/F 计数器第一次计数值低 8 位的值。当启动计数触控按键时隙计数器第一次溢出，寄存器存入 16-bit C/F 计数器第一次计数值低 8 位值。当 TKST 位为“0”时，该寄存器将被清零。

16.7.11. CFnOUT1H 寄存器，地址 0x0F8F+8n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	D7-D0	启动一次 TKST 计数配置的计数次数大于 1，该寄存器用于存储触控按键四个 16-bit C/F 计数器第一次计数值高 8 位的值。当启动计数触控按键时隙计数器第一次溢出，寄存器存入 16-bit C/F 计数器第一次计数值高 8 位值。当 TKST 位为“0”时，该寄存器将被清零。

16.7.12. CFnOUT2L 寄存器，地址 0x0F90+8n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	D7-D0	启动一次 TKST 计数配置的计数次数大于 2，该寄存器用于存储触控按键四个 16-bit C/F 计数器第二次计数值低 8 位的值。当启动计数触控按键时隙计数器第二次溢出，寄存器存入 16-bit C/F 计数器第二次计数值低 8 位值。当 TKST 位为“0”时，该寄存器将被清零。

16.7.13. CFnOUT2H 寄存器，地址 0x0F91+8n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	D7-D0	启动一次 TKST 计数配置的计数次数大于 2，该寄存器用于存储触控按键四个 16-bit C/F 计数器第二次计数值高 8 位的值。当启动计数触控按键时隙计数器第二次溢出，寄存器存入 16-bit C/F 计数器第二次计数值高 8 位值。当 TKST 位为“0”时，该寄存器将被清零。

16.7.14. CFnOUT3L 寄存器，地址 0x0F92+8n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	D7-D0	启动一次 TKST 计数配置的计数次数大于 3，该寄存器用于存储触控按键四个 16-bit C/F 计数器第三次计数值低 8 位的值。当启动计数触控按键时隙计数器第三次溢出，寄存器存入 16-bit C/F 计数器第三次计数值低 8 位值。当 TKST 位为“0”时，该寄存器将被清零。

16.7.15. CFnOUT3H 寄存器，地址 0x0F93+8n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	D7-D0	启动一次 TKST 计数配置的计数次数大于 3，该寄存器用于存储触控按键四个 16-bit C/F 计数器第三次计数值高 8 位的值。当启动计数触控按键时隙计数器第三次溢出，寄存器存入 16-bit C/F 计数器第三次计数值高 8 位值。当 TKST 位为“0”时，该寄存器将被清零。

16.7.16. TKMn16DL 寄存器，地址 0x0F94+8n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	D7-D0	启动一次 TKST 计数配置的计数次数为 n，该寄存器用于存储触控按键四个 16-bit C/F 计数器第 n 次计数值低 8 位的值。当启动计数触控按键时隙计数器第 n 次溢出，寄存器存入 16-bit C/F 计数器第 n 次计数值低 8 位值。当 TKST 位为“0”时，该寄存器将被清零。

16.7.17. TKMn16DH 寄存器，地址 0x0F95+8n(n=0~3)

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
Reset	0	0	0	0	0	0	0	0
Type	RO	RO	RO	RO	RO	RO	RO	RO

Bit	Name	Function
7:0	D7-D0	启动一次 TKST 计数配置的计数次数为 n，该寄存器用于存储触控按键四个 16-bit C/F 计数器第 n 次计数值高 8 位的值。当启动计数触控按键时隙计数器第 n 次溢出，寄存器存入 16-bit C/F 计数器第 n 次计数值高 8 位值。当 TKST 位为“0”时，该寄存器将被清零。

17. GPIO

本芯片共包含 30 个 GPIO。这些 IO 除了作为普通输入/输出端口以外还通常具备一些与内核周边电路通讯的功能。

每个端口有 8 个标准寄存器供其操作使用。这些寄存包括：

- TRISx 寄存器（数据方向寄存器）
- PORTx 寄存器（用于读器件引脚上的电平）
- LATx 寄存器（输出锁存器）
- WPUx 寄存器（上拉控制）
- WPDx 寄存器（下拉控制）
- PSRCx 寄存器（源电流选择）
- PSINKx 寄存器（灌电流选择）
- ITYPEx 寄存器（中断类型选择）

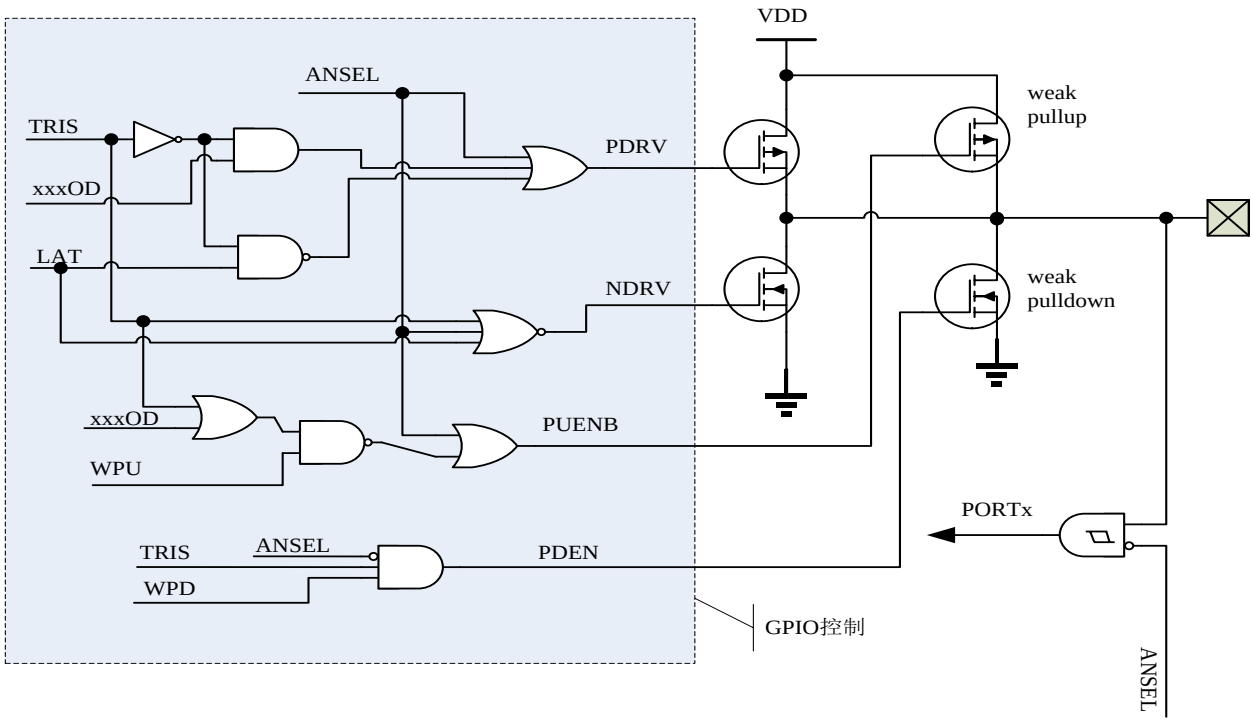


图 17.1 I/O 的结构原理

某些端口可能有以下额外寄存器：

- ANSELx（模拟选择寄存器）

通常情况下，当某个端口上的外设使能时，其相关引脚可能不能用作通用输出引脚，但可读取该引脚。

数据锁存器（LATx 寄存器）用于对 I/O 引脚所驱动的值进行“读-修改-写”操作。对 LATx 寄存器的写操作与对相应 PORTx 寄存器的写操作具有相同的效果。读 LATx 寄存器将读取保存在 I/O 端口锁存器中的值，而读 PORTx 寄存器将读取实际的 I/O 引脚值。

支持模拟输入的端口具有相关的 ANSELx 寄存器。当 ANSEL 位置 1 时，禁止与该位相关的数字输入缓冲器。禁止输入缓冲器可防止逻辑输入电路产生短路电流。

17.1. 端口和 TRIS 寄存器

所有的管脚 PORTx.y 都是双向端口，其方向控制寄存器就是 TRISx.y。将 TRISx.y 设置为“0”会将该对应 PORTx.y 端口设置为输出端口。在置为输出端口时，输出驱动电路会被打开，输出寄存器里的数据会被放置到输出端口。当 IO 处于输入状态时（TRISA=1），对 PORTx 进行读反映的是输入端口的状态。在 PORTx 上进行写动作时，PORTx 内容会被写入输出寄存器。所有的写操作都是“读-更改-写”这样一个微流程，即数据被读，然后更改，再写入输出寄存器的过程。

当 MCLRE 为 1 时，PORTC[0]读的值为 0，此时它是作为外部复位管脚。

17.2. 弱上拉

每个端口都有一个可以单独设置的内部弱上拉功能。控制 WPUx 寄存器里的位就可使能或关断这些弱上拉电路。当 GPIO 被设置为输出时，这些弱上拉电路会被自动关断。弱上拉电路在上电复位期间被置为关断状态，因为 WPUx 寄存器复位为无效状态。

当 PORTC[0]配置为复位脚时，内部上拉是自动打开的，此时 WPUC[0]不起作用。

17.3. 弱下拉

跟弱上拉功能类似，每个管脚处于数字输入时具有内部弱下拉功能，由寄存器 WPDx 控制。需要注意的是，弱上拉和弱下拉不是互斥的，即它们可以同时打开。

另外，PORTC[0]作为复位脚时，弱上拉自动打开，**但它不反映到 WPUC[0]上**，同时弱下拉自动关闭，WPDC[0]不起作用。

17.4. 开漏输出

以下 3 种功能管脚支持开漏输出：
SPI_MISO, SPI_MOSI
I2C_SCL, I2C_SDA
UART_TX

开漏输出由寄存器 ODCON0 相关位控制，当相关位为 1 时，该功能所在的管脚即配置为开漏。

注意：

1. **管脚的开漏功能和内部上拉功能可以同时打开；**
2. **对于拥有重映射功能的 I2C，开漏设置只应用在对应的被映射的管脚上；**

17.5. ANSELA 寄存器

ANSELA 寄存器用于控制 IO 的数模输入，当 ANSELA.x 为 1 时，对应的 IO 口为模拟引脚，IO 的输入上拉、下拉被自动禁止，软件读该 IO 返回的是 0。

ANSELA 寄存器位对数字输出驱动没有影响，换言之，TRIS 位的优先级更高，即当 TRIS 为 0 时，不管相关的 ANSELA.x 是 0 还是 1，对应的 IO 为数字输出 IO。要想配置真正的模拟管脚，TRIS 要置 1，把数字输出驱动关闭。

17.6. 源电流选择

每个 I/O 口都支持不同的源电流驱动能力。通过配置相应的选择寄存器 PSRCx，指定的 I/O 端口可支持 4 种级别的源电流驱动能力。仅当对应的引脚被设为输出时，其源电流选择位才有效。否则，这些选择位无效。用户可参考 I/O 电气特性章节为不同应用选择所需的源电流。

17.7. 灌电流选择

每个 I/O 都支持 2 种不同的灌电流驱动能力，设置寄存器为 PSINKx，当 I/O 设置为输出管脚时，其灌电流设置位才有效。

17.8. 管脚输出的优先级

每个 I/O 管脚均复用了多个功能，当某管脚复用的功能模块都使能输出的情况下，就存在优先级的问题。

因为输入是连到各个功能模块的，故输入不存在优先级问题，例如 PB0 作为 GPIO 输入功能时，同时也作为 TIM2 的捕捉输入。

17.9. PORTx 功能及优先级

管脚名称	功能优先级（由高到低排序）	管脚名称	功能优先级（由高到低排序）
PA0	TIM1_CH1 SPI_MISO PA0	PB0	TIM1_CH3N TIM2_CH1 SPI_SCK PB0
PA1	TIM1_CH2 SPI_MOSI PA1	PB1	CLKO TIM1_CH4 PB1
PA2	ISPCK（处于调试模式） I2C_SCL **UART_RX** PA2	PB2	I2C_SCL PB2
PA3	TIM1_CH2N PA3	PB3	I2C_SDA PB3
PA4	TIM2_CH2 PA4	PB4	TIM1_CH3 PB4
PA5	TIM2_CH1 UART_CK PA5	PB5	TIM2_CH3 SPI_NSS PB5
PA6	UART_TX PA6	PB6	ISPDAT（处于调试模式） I2C_SDA **UART_TX** PB6
PA7	PA7	PB7	OSC2（XT 模式） **SPI_MOSI** PB7

管脚名称	功能优先级（由高到低排序）	管脚名称	功能优先级（由高到低排序）
PC0	MCLRB（复位脚） TIM1_CH1N PC0	PD0	SPI_NSS PD0
PC1	OSC1（XT 模式） **SPI_MISO** PC1	PD1	TIM1_CH1 UART_CK PD1
PC2	PC2	PD2	TIM1_CH2 PD2
PC3	PC3	PD3	TIM1_CH3 **SPI_SCK** PD3
PC4	PC4	PD4	CLKO PD4
PC5	TIM1_CH3N PC5	PD5	TIM1_CH4 PD5
PC6	TIM1_CH2N PC6		
PC7	TIM1_CH1N PC7		

17.10. 管脚功能重映射

为提高应用的灵活性，有部分功能输入/输出支持重映射功能，即可以在两个管脚之间选择输入或输出，由寄存器 AFP0，AFP1 和 AFP2 设置。

17.11. 外部中断

所有 I/O 都可被选择为外部中断源，但同一时刻最多只有 8 个 IO 可以作为外部中断管脚，它们具备以下特性：

- 上升沿中断
- 下降沿中断
- 双边沿中断
- 低电平中断

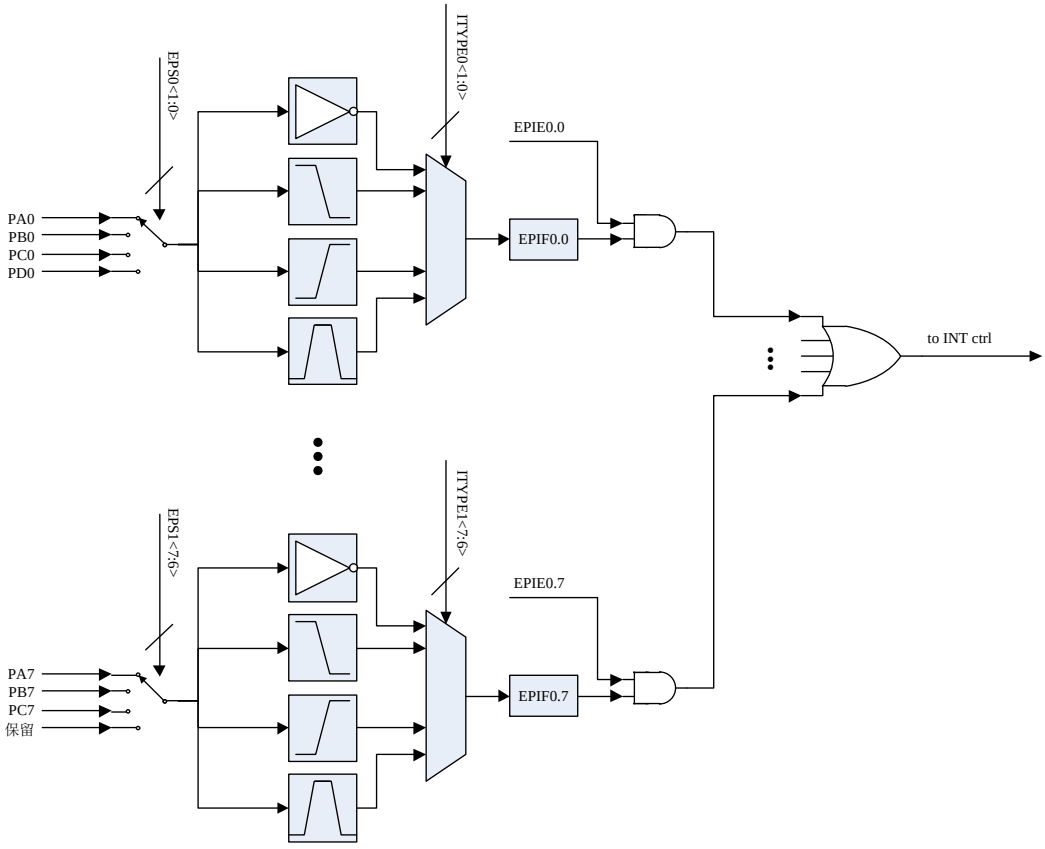


图 17.2 外部中断结构框图

中断类型的选择通过寄存器 ITYPE0、1 设置。

ITYPEx[1:0]/[3:2]值	中断触发类型
00	低电平
01	上升沿
10	下降沿
11	双边沿

外部中断源的选择通过 EPS0，EPS1 设置。

17.12. 关于读端口 PORTx

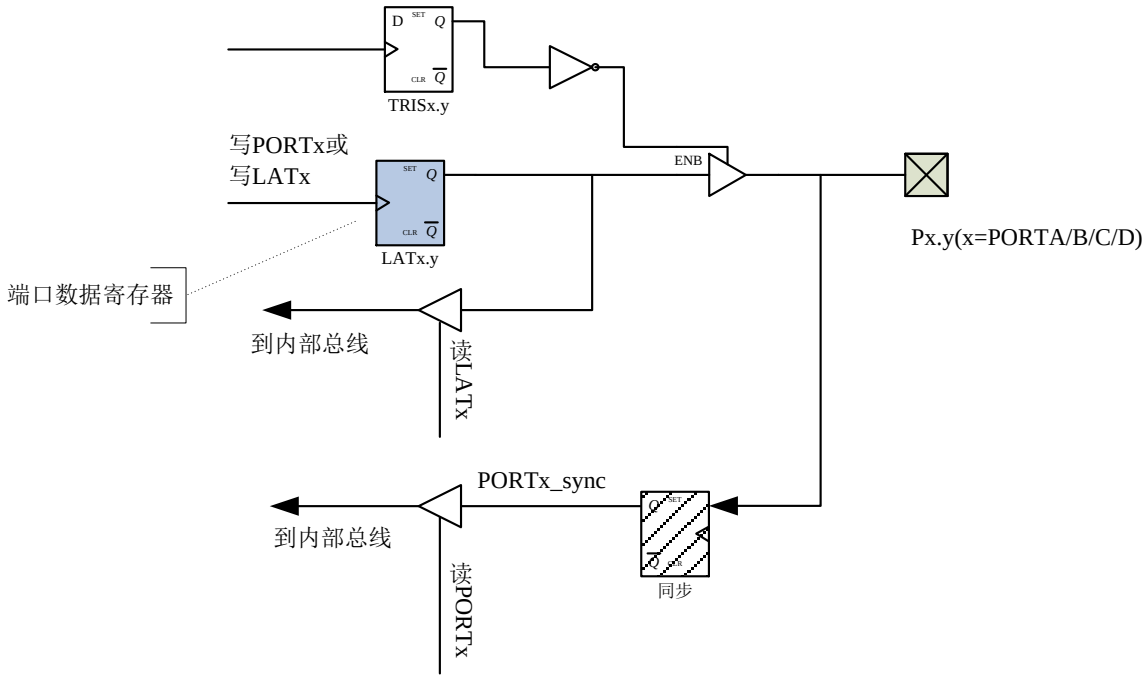


图 17.3 端口读操作原理框图

在 FT62F08x 系列芯片中，操作 GPIO 有两种方式：访问 PORTx 寄存器或者 LATx 寄存器，它们有不同的 SFR 地址。

对于读操作，“**读 PORTx**”返回的是管脚经过同步寄存器后的值，而“**读 LATx**”返回的是端口数据寄存器的值；换言之，软件对端口数据寄存器写操作之后，至少要经过一个系统时钟之后，才能通过“**读 PORTx**”的方式得到新值，而“**读 LATx**”则无需等待；

对于写操作，无论是写 PORTx 还是 LATx，都是对端口数据寄存器进行写；

由于以上特性，当软件使用“读-修改-写”（可参阅 20.1 小节）指令对 PORTx 进行写操作时，需要特别注意以下情形：

BSR PORTx, n; 对 PORTx 第 n 位置 1

BSR PORTx, m; 对 PORTx 第 m 位置 1

…

软件期望的波形如下：

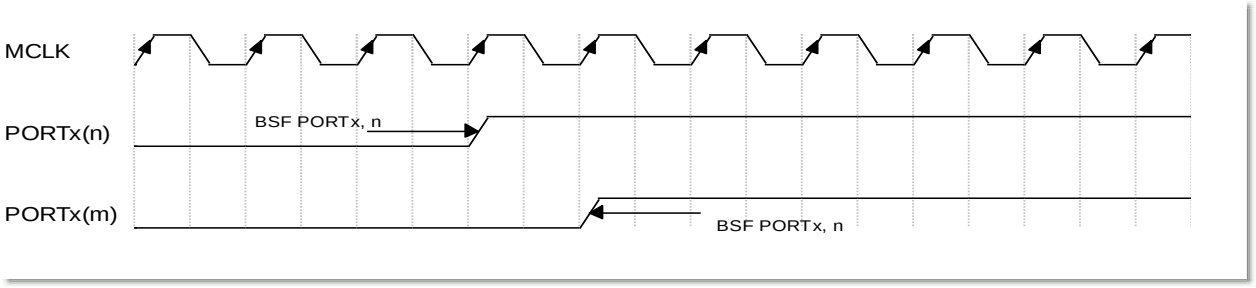


图 17.4 连续使用 RMW 指令对 PORTx 写操作的期望时序

实际输出波形如下：

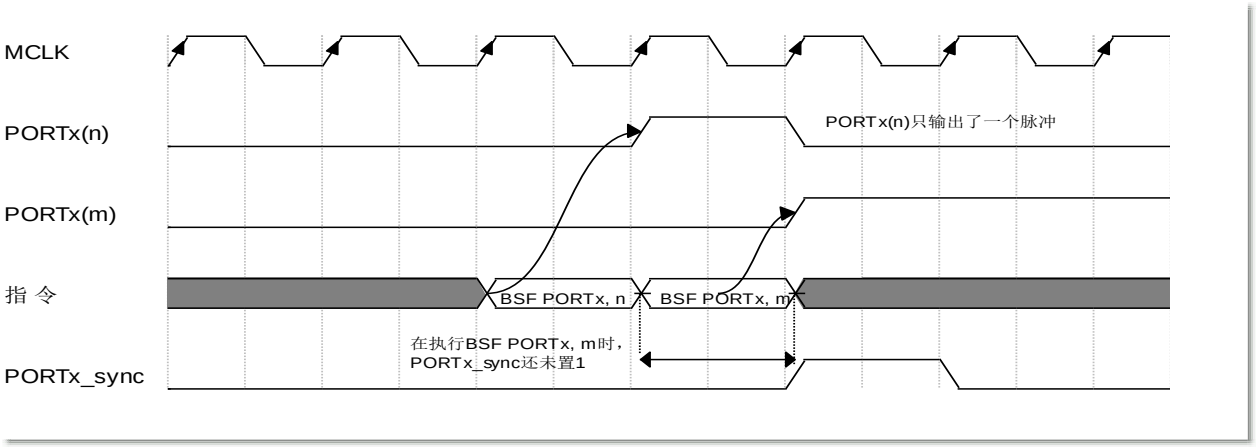


图 17.5 连续使用 RMW 指令对 PORTx 写操作的输出时序

出现这个现象的原因是在执行“BSR PORTx, m”（回顾一下 RMW 指令的执行流程：先读取 PORTx，修改数据，写 PORTx（LATx））时，由于同步的原因，PORTx_sync 还保持为 0，那么写回 PORTx 时刻，这一位的“0”又被写回到 LATx，导致管脚 PORTx.n 只有一个高脉冲。

有以下两种方式解决这一问题：

a) 在 PORTx 连续写操作中间插入一个 NOP；

```
BSR PORTx, n;   对 PORTx 第 n 位置 1
NOP             ;   插入 NOP 等待
BSR PORTx, m;   对 PORTx 第 m 位置 1
```

b) 或者，写操作用 LATx 寄存器而不是 PORTx；

```
BSR LATx, n ;     直接操作端口数据寄存器 LATx
BSR LATx, m;      直接操作端口数据寄存器 LATx
```

注意：只有 1T 速度模式下才有该现象，不存在于其它 2T/4T 模式，原因是处于 2T/4T 模式下，执行后续指令时，PORTx_sync 已经同步到最新的值。

17.13. 与端口相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
PSRC0	0x11A	管脚的源电流设置 0								1111 1111
PSRC1	0x11B	管脚的源电流设置 1								1111 1111
PSINK0	0x19A	管脚的灌电流设置 0								0000 0000
PSINK1	0x19B	管脚的灌电流设置 1								0000 0000
PSINK2	0x19C	管脚的灌电流设置 2								0000 0000
PSINK3	0x19D	—	—	管脚的灌电流设置 3						--00 0000
ITYPE0	0x11E	管脚中断类型设置 0								0000 0000
ITYPE1	0x11F	管脚中断类型设置 1								0000 0000
AFP0	0x19E	管脚重映射寄存器 0								0000 0000
AFP1	0x19F	—	管脚重映射寄存器 1							-000 0000
AFP2	0x11D	—	—	—	管脚重映射寄存器 2					---0 0000
EPS0	0x118	外部中断管脚选择 0								0000 0000
EPS1	0x119	外部中断管脚选择 1								0000 0000
EPIF0	0x14	外部管脚中断标志位								0000 0000
EPIE0	0x94	外部管脚中断使能位								0000 0000
ODCON0	0x21F	—	—	—	—	—	SPIOD	I2COD	UROD	---- -000
PORTA	0x0C	端口 A 读管脚寄存器								xxxx xxxx
PORTB	0x0D	端口 B 读管脚寄存器								xxxx xxxx
PORTC	0x0E	端口 C 读管脚寄存器								xxxx xxxx
PORTD	0x0F	—	—	端口 D 读管脚寄存器						--xx xxxx
TRISA	0x8C	端口 A 方向控制								1111 1111
TRISB	0x8D	端口 B 方向控制								1111 1111
TRISC	0x8E	端口 C 方向控制								1111 1111
TRISD	0x8F	—	—	端口 D 方向控制						--11 1111
LATA	0x10C	端口 A 数据锁存器								xxxx xxxx
LATB	0x10D	端口 B 数据锁存器								xxxx xxxx
LATC	0x10E	端口 C 数据锁存器								xxxx xxxx
LATD	0x10F	—	—	端口 D 数据锁存器						--xx xxxx
ANSELA	0x197	模拟管脚设置								0000 0000

17.13.1. PSRC0，地址 0x11A

Bit	7	6	5	4	3	2	1	0
Name	PSRCB				PSRCA			
Reset	1	1	1	1	1	1	1	1
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:4	PSRCB	PSRCB[3:2]，控制 PORTB[7:4]源电流能力 PSRCB[1:0]，控制 PORTB[3:0]源电流能力
3:0	PSRCA	PSRCA[3:2]，控制 PORTA[7:4]源电流能力 PSRCA[1:0]，控制 PORTA[3:0]源电流能力

17.13.2. PSRC1，地址 0x11B

Bit	7	6	5	4	3	2	1	0
Name	PSRCD				PSRCC			
Reset	1	1	1	1	1	1	1	1
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:4	PSRCD	PSRCD[3:2]，控制 PORTD[5:4]源电流能力 PSRCD[1:0]，控制 PORTD[3:0]源电流能力
3:0	PSRCC	PSRCC[3:2]，控制 PORTC[7:4]源电流能力 PSRCC[1:0]，控制 PORTC[3:0]源电流能力

PSRCx[1:0]/[3:2]值	源电流能力
00	L0（最小）
01	L1
10	L2
11	L3（最大）

17.13.3. PSINK0，地址 0x19A

Bit	7	6	5	4	3	2	1	0
Name	PSINK0							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	PSINK0	PORTA 的灌电流能力设置 0: L0, 53mA 1: L1, 62mA

17.13.4. PSINK1，地址 0x19B

Bit	7	6	5	4	3	2	1	0
Name	PSINK1							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	PSINK1	PORTB 的灌电流能力设置 0: L0, 53mA 1: L1, 62mA

17.13.5. PSINK2，地址 0x19C

Bit	7	6	5	4	3	2	1	0
Name	PSINK2							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	PSINK2	PORTC 的灌电流能力设置 0: L0, 53mA 1: L1, 62mA

17.13.6. PSINK3，地址 0x19D

Bit	7	6	5	4	3	2	1	0
Name	PSINK3							
Reset	—	—	0	0	0	0	0	0
Type	RO-0	RO-0	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:6	NA	保留位
5:0	PSINK3	PORTD 的灌电流能力设置 0: L0, 53mA 1: L1, 62mA

17.13.7. ITYPE0，地址 0x11E

Bit	7	6	5	4	3	2	1	0
Name	ITYPE0							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:4	ITYPE0[7:4]	ITYPE0[7:6]，控制 PORTx.3 中断类型
3:0	ITYPE0[3:0]	ITYPE0[3:2]，控制 PORTx.1 中断类型

17.13.8. ITYPE1，地址 0x11F

Bit	7	6	5	4	3	2	1	0
Name	ITYPE1							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:4	ITYPE1[7:4]	ITYPE1[7:6]，控制 PORTx.7 中断类型（**仅 PORTA/B/C**） ITYPE1[5:4]，控制 PORTx.6 中断类型（**仅 PORTA/B/C**）
3:0	ITYPE1[3:0]	ITYPE1[3:2]，控制 PORTx.5 中断类型 ITYPE1[1:0]，控制 PORTx.4 中断类型

17.13.9. AFP0，地址 0x19E

Bit	7	6	5	4	3	2	1	0
Name	AFP0							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function

位	值	受控功能	复用管脚
bit0	0	I2C_SDA	PB3
	1		PB6
bit1	0	ADC_ETR	PA4
	1		PB2
bit2	0	TIM1_CH3N	PB0
	1		PC5
bit3	0	TIM1_CH2N	PA3
	1		PC6
bit4	0	TIM1_CH1N	PC0
	1		PC7
bit5	0	SPI_NSS	PB5
	1		PD0
bit6	0	TIM1_CH1	PA0
	1		PD1
bit7	0	UART_CK	PA5
	1		PD1

7:0 AFP0

17.13.10. AFP1，地址 0x19F

Bit	7	6	5	4	3	2	1	0
Name	AFP1							
Reset	—	0	0	0	0	0	0	0
Type	RO-0	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7	NA	保留位
6:0	AFP1	

位	值	受控功能	复用管脚
bit0	0	TIM1_CH2	PA1
	1		PD2
bit1	0	TIM1_CH3	PB4
	1		PD3
bit2	0	TIM2_CH1	PA5
	1		PB0
bit3	0	TIM1_BKIN	PB3
	1		PD4
bit4	0	I2C_SCL	PB2
	1		PA2
bit5	0	TIM1_CH4	PB1
	1		PD5
bit6	0	CLKO	**PB1**
	1		PD4

17.13.11. AFP2，地址 0x11D

Bit	7	6	5	4	3	2	1	0
Name	AFP2							
Reset	—	—	—	0	0	0	0	0
Type	RO-0	RO-0	RO-0	RW	RW	RW	RW	RW

Bit	Name	Function
7:5	NA	保留位，读 0
4:0	AFP2	

位	值	受控功能	复用管脚
bit0	0	UART_TX	PA6
	1		PB6
bit1	0	UART_RX	PA7
	1		PA2
bit2	0	SPI_MISO	PA1
	1		PC1
bit3	0	SPI_MOSI	PA0
	1		PB7
bit4	0	SPI_SCK	PB0
	1		PD3

17.13.12. EPS0，地址 0x118

Bit	7	6	5	4	3	2	1	0
Name	EPS0							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	EPS0	外部中断 EINT3~0 的管脚选择

EPS0[1:0]值	**EINT0 管脚**	**EPS0[3:2]值**	**EINT1 管脚**
00	PA0	00	PA1
01	PB0	01	PB1
10	PC0	10	PC1
11	PD0	11	PD1
EPS0[5:4]值	**EINT2 管脚**	**EPS0[7:6]值**	**EINT3 管脚**
00	PA2	00	PA3
01	PB2	01	PB3
10	PC2	10	PC3
11	PD2	11	PD3

17.13.13. EPS1，地址 0x119

Bit	7	6	5	4	3	2	1	0
Name	EPS1							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	EPS1	外部中断 EINT7~4 的管脚选择

EPS1[1:0]值	**EINT4 管脚**	**EPS1[3:2]值**	**EINT5 管脚**
00	PA4	00	PA5
01	PB4	01	PB5
10	PC4	10	PC5
11	PD4	11	PD5
EPS1[5:4]值	**EINT6 管脚**	**EPS1[7:6]值**	**EINT7 管脚**
00	PA6	00	PA7
01	PB6	01	PB7
10	PC6	10	PC7
11	保留（接 PC6）	11	保留（接 PC7）

17.13.14. EPIF0，地址 0x14

Bit	7	6	5	4	3	2	1	0
Name	EPIF0							
Reset	0	0	0	0	0	0	0	0
Type	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

Bit	Name	Function
Bit[7:0]	EPIF0	外部中断 x 标志位 0：外部管脚 x 没触发中断，或已由软件清 0 1：外部管脚 x 触发了中断 写操作： **写 1 清 0，**建议只使用 STR, MOVWI 指令，而不是 BSR 位操作（**该寄存器不能由软件方式置 1**）

17.13.15. EPIE0，地址 0x94

Bit	7	6	5	4	3	2	1	0
Name	EPIE0							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
Bit[7:0]	EPIE0	外部中断 x 允许位 0：禁止外部中断 x 1：允许外部中断 x，当相关标志位 EPIF0.x 为 1 且 GIE 为 1 时，CPU 将执行中断程序

17.13.16. ODCON0，地址 0x21F

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	SPIOD	I2COD	UROD
Reset	—	—	—	—	—	0	0	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW	RW

Bit	Name	Function
7:3	N/A	保留位，读 0
2	SPIOD	SPI_MISO, SPI_MOSI 管脚的开漏输出设置，高有效
1	I2COD	I2C_SCL, I2C_SDA 管脚的开漏输出设置，高有效
0	UROD	UART_TX 管脚的开漏输出设置，高有效

17.13.17. PORTx，地址 0x0C, 0D, 0E, 0F

Bit	7	6	5	4	3	2	1	0
Name	PORTA/B/C/D							
Reset	x	x	x	x	x	x	x	x
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	PORTx	PORTx 管脚寄存器 读返回的是管脚上的电平，写是写到相应的 LATx 寄存器

17.13.18. TRISx，地址 0x8C, 8D, 8E, 8F

Bit	7	6	5	4	3	2	1	0
Name	TRISA/B/C/D							
Reset	1	1	1	1	1	1	1	1
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	TRISx	PORTx 方向控制寄存器 1 = 输入 0 = 输出

17.13.19. LATx，地址 0x10C, 10D, 10E, 10F

Bit	7	6	5	4	3	2	1	0
Name	LATA/B/C/D							
Reset	x	x	x	x	x	x	x	x
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	LATx	PORTx 数据寄存器

17.13.20. WPUx，地址 0x18C, 18D, 18E, 18F

Bit	7	6	5	4	3	2	1	0
Name	WPUA/B/C/D							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	WPUx	PORTx 弱上拉控制寄存器 1 = 使能弱上拉 0 = 关闭弱上拉

17.13.21. WPDx，地址 0x20C, 20D, 20E, 20F

Bit	7	6	5	4	3	2	1	0
Name	WPDA/B/C/D							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	WPDx	PORTx 弱下拉控制寄存器 1 = 使能弱下拉 0 = 关闭弱下拉

17.13.22. ANSELA，地址 0x197

Bit	7	6	5	4	3	2	1	0
Name	模拟管脚设置寄存器							
Reset	0	0	0	0	0	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:0	ANSELA	模拟选择寄存器 A，控制 Anx 相关的管脚 1 = Anx 为模拟管脚 0 = Anx 为数字管脚

18. 看门狗定时器

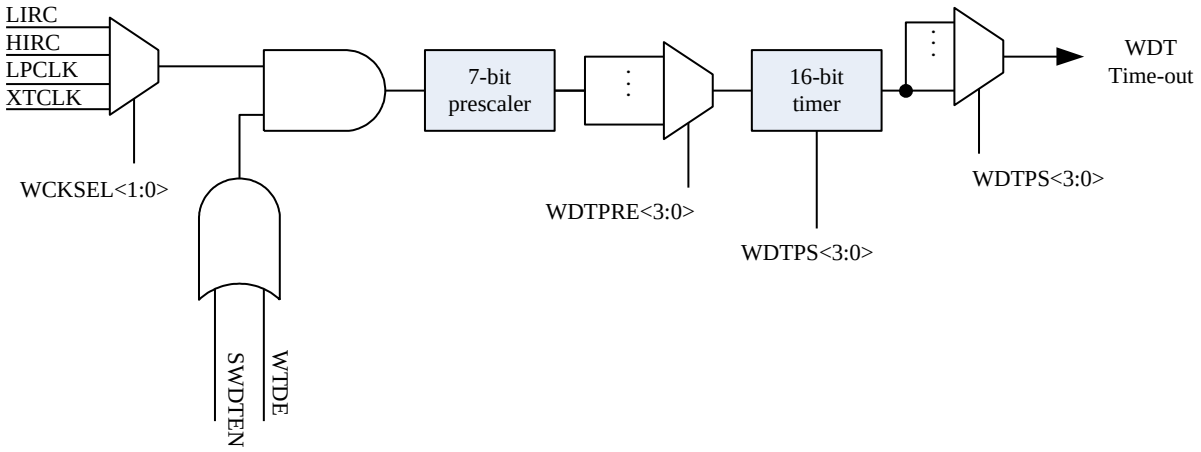


图 18.1 看门狗结构框图

看门狗的时钟源为内部慢时钟（32kHz），它是一个带 7 位预分频的 16 位计数器，其中预分频和周期可编程，分别由 WDTPRE 和 WDTPS 设置。

WDT 的硬件使能位位于配置寄存器 UCFG0 的第 3 位，WDTEN，软件使能位位于 WDTCON 寄存器的第 0 位，为 1 时表示使能看门狗，为 0 时禁止。

指令 CLRWDT、SLEEP 会清除看门狗计数器。

在使能了看门狗的情况下，处于睡眠时看门狗溢出事件可以作为一个唤醒源，而 MCU 正常工作时 WDT 则是作为一个复位源。

条件	看门狗状态
WDTEN 和 SWDTEN 同时为 0	清零
CLRWDT 指令	
进入 SLEEP、退出 SLEEP 时刻	
写 WDTCON	
写 WCKSEL	

注意：

1. **如果内部慢时钟从 32K 切换到 256K 模式（或反之从 256K 切换到 32K 模式，由 LFMOD 位控制），都不影响看门狗计时，因为 WDT 固定使用 32K 时钟源，见 5.1 小节的时钟框图；**
2. **PWRT 和 OST 复用了 WDT 定时器，故 PWRT 或 OST 工作时，看门狗的复位功能是暂时屏蔽的；**

18.1. 看门狗时钟源

WDT 有 4 种时钟源可选，由寄存器 MISC0 的 WCKSEL 位设置。在 WDT 使能的情况下，所选择的时钟源被自动使能，并在 SLEEP 模式下保持。

注意：

1. 如果要选择 LP 晶体时钟，系统时钟配置寄存器位 FOSC 必须选择 LP 模式，否则对应的时钟源将不被使能；

2. 同理，如果要选择 XT 晶体时钟，系统时钟配置寄存器位 FOSC 必须选择 XT 模式，否则对应的时钟源将不被使能；

18.2. 与看门狗相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
WDTCON	0x97	WDTPRE[2:0]			WDTPS[3:0]				SWDTEN	1110 1000
UCFG0	0x2000	—	CPB	MCLRE	PWRTEB	WDTE	FOSC2	FOSC1	FOSC0	qqqq qqqq
MISC0	0x11C	—	—	—	—	—	—	WCKSEL		---- --00

18.2.1. WDTCON 寄存器，地址 0x97

Bit	7	6	5	4	3	2	1	0
Name	WDTPRE[2:0]			WDTPS[3:0]				SWDTEN
Reset	1	1	1	0	1	0	0	0
Type	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Name	Function
7:5	WDTPRE	看门狗预分频设置位 000：1:1 001：1:2 010：1:4 011：1:8 100：1:16 101：1:32 110：1:64 **111：1:128（复位值）**
4:1	WDTPS	看门狗定时器周期选择： 0000 = 1:32 0001 = 1:64 0010 = 1:128 0011 = 1:256 **0100 = 1:512（复位值）** 0101 = 1:1024 0110 = 1:2048 0111 = 1:4096 1000 = 1:8192 1001 = 1:16384 1010 = 1:32768 1011 = 1:65536 1100 = 1:65536 1101 = 1:65536 1110 = 1:65536 1111 = 1:65536
0	SWDTEN	看门狗软件使能位 1 = 使能 0 = 禁止

18.2.2. MISC0 寄存器，地址 0x11C

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	WCKSEL	
Reset	—	—	—	—	—	—	0	0
Type	RO-0	RO-0	RO-0	RO-0	RO-0	RO-0	RW	RW

Bit	Name	Function
7:2	N/A	保留位，读 0
1:0	WCKSEL	WDT 时钟源选择 00 = LIRC 01 = HIRC 10 = LP，只有当 FOSC 选择 LP 模式时才有效 11 = XT，只有当 FOSC 选择 XT 模式时才有效

19. 慢时钟测量

芯片集成了两个内部 RC 振荡器，一个是经过出厂校准的高速高精度的 16M 快时钟 HIRC，一个是低速低功耗的 32K 时钟 LIRC，利用慢时钟测量功能可以把 LIRC 的周期用系统时钟计算出来。此功能可以比较精准的测量内部慢时钟周期。

19.1. 测量原理

慢时钟测量类似于定时器的捕捉模式，处于这种模式下，被测量时钟 LIRC 的边沿（任意沿）将会触发定时器，在另一高速时钟（如 HIRC）的作用下开始计数，在此后的第 2 个（或第 8 个，平均模式时）LIRC 边沿到来时，定时器停止计数，同时把定时器的值锁存到 SOSCPRL/H 寄存器。

注意：慢时钟测量使用的定时器是 TIM2。

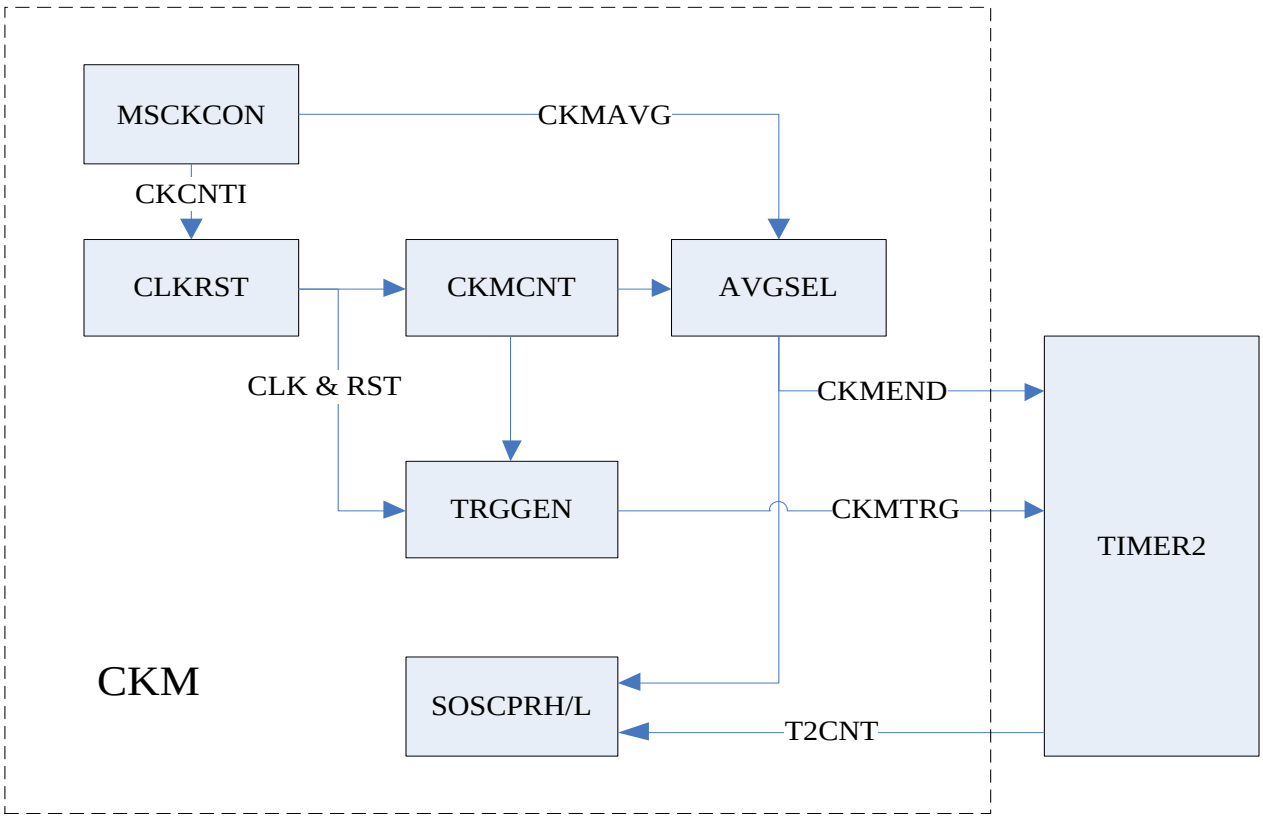


图 19.1 慢时钟测量模式原理框图

注意：

1. **在慢时钟测量过程中软件不要写 SOSCPRH/L；**
2. **不要在单步调试下做慢时钟测量，因为暂停模式下 TIM2 被停止，这样会导致测量结果不正确；**
3. **若 SYSON bit 为 0 时，慢时钟测量无法在 SLEEP 模式下进行，不要在测量运行时进入 SLEEP 模式。**

19.2. 上电自动测量

在上电后慢时钟测量将会自动启动，此时 CKCNTI 置 1，CKMAVG 为 0，打开 LIRC 和 HIRC。TIM2 的时钟被自动配置为 16M 的内部高速时钟，即类似设置 T2CKSRC 为 001、TIM2EN=1 的功能，但未配置这些位。TIM2 使用默认配置，无需置位 CEN 使能 TIM2 计数，此时不能配置 TIM2。
在测量过程中，实际应用程序已在运行，若要使用 TIM2 则需要查询 CKCNTI。若 CKCNTI 为 0 即可使用 TIM2，此时 SOSCPR 寄存器的值为有效值，其单位为 F_{HSI} 时钟的个数。

注意：

1. 上电自动测量不会置位 CKM 中断标志；

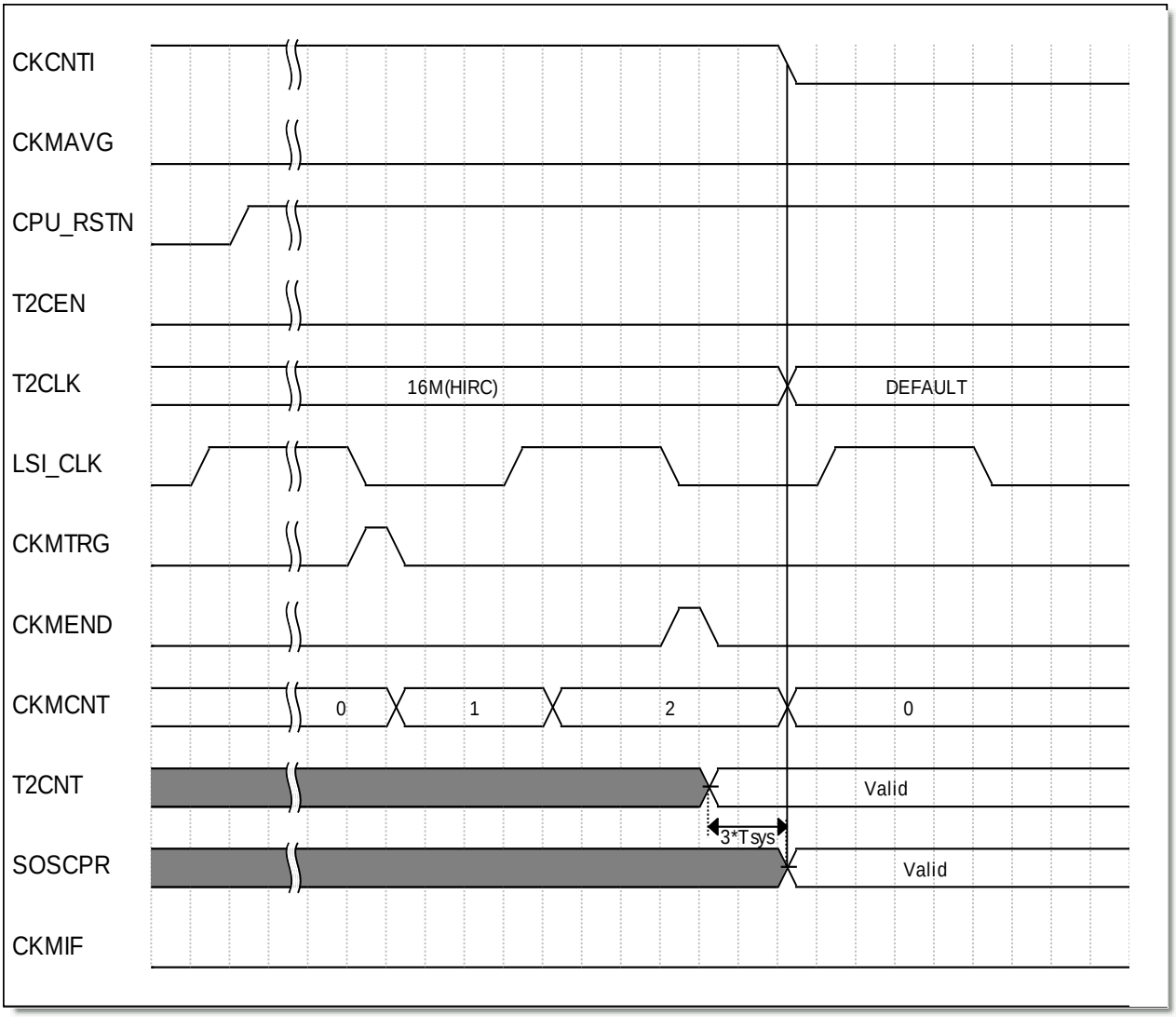


图 19.2 上电慢时钟自动测量时序图

19.3. 操作步骤

1. 为提高计量精度，建议设置 T2CKSRC 为 001，TIM2EN=1，选择 16M 的内部高速时钟；
2. 关闭 TIM2 的相关中断使能，设置 TIM2ARRH/L 为最大值，设置 TIM2PSC 为 0000；
3. 设置 TIM2CR1 为复位值，再将 CEN 置 1，使能 TIM2；
4. 如果选择 4 次平均，则把 MSCKCON.1 置 1，否则把它清 0；
5. 置位 MSCKCON.0，开始测量；
6. 测量结束后 MSCKCON.0 自动清 0，中断标志置 1；
7. 可以用查询或中断的方式等待结束；
8. 当查询到中断标志为 1 时读取得到的 SOSCPR 即为最终结果。

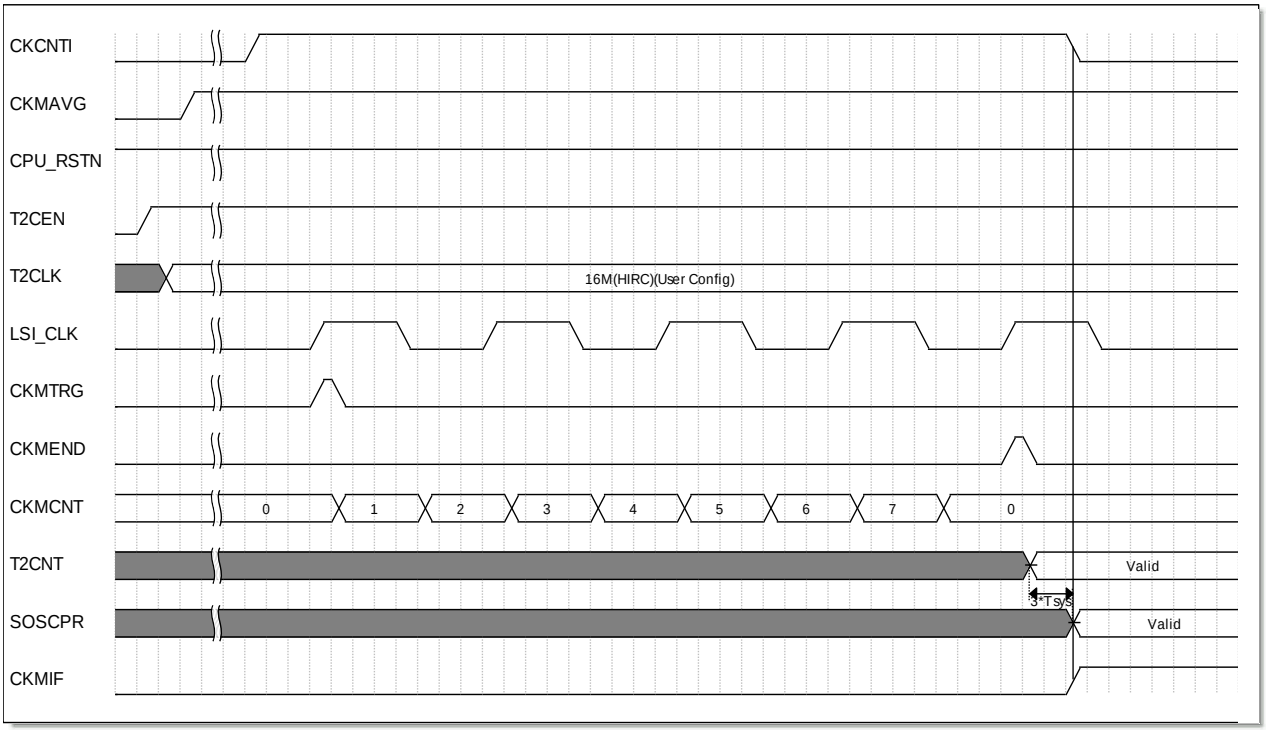


图 19.3 慢时钟测量模式时序图

19.4. 与慢时钟测量相关寄存器汇总

名称	地址	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	复位值
MSCKCON	0x41D	—	—	—	—	—	—	CKMAVG	CKCNTI	---- --01
SOSCPRL	0x41E	SOSCPR[7:0]								1111 1111
SOSCPRH	0x41F	—				SOSCPR[11:8]				---- 1111

19.4.1. MSCKCON 寄存器，地址 0x41D

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CKMAVG	CKCNTI
Reset	—	—	—	—	—	—	0	1
Type	RW	RW	RW	RW	RO-0	RW	RW	RW

Bit	Name	Function
7:2	N/A	保留位，读 0
1	CKMAVG	快时钟测量慢时钟周期的测量平均模式 1 = 打开平均模式（自动测量并累加 4 次） 0 = 关闭平均模式
0	CKCNTI	Clock Count Init –使能快时钟测量慢时钟周期 1 = 使能快时钟测量慢时钟周期 0 = 关闭快时钟测量慢时钟周期 注：这一位在测量完毕后会自动归零

19.4.2. SOSCPR 寄存器，地址 0x41E，41F

SOSCPRL，地址 0x41E

Bit	7	6	5	4	3	2	1	0
Name	SOSCPR[7:0]							
Reset	8’hFF							
Type	RW							

SOSCPRH，地址 0x41F

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	SOSCPR[11:8]			
Reset	—	—	—	—	4’hF			
Type	RO-0	RO-0	RO-0	RO-0	RW			

Bit	Name	Function
0x41E：7:0 0x41F：3:0	SOSCPR[11:0]	低速振荡器周期（单位：TIM2 时钟周期数） 用于慢时钟测量功能，T_{LSI} = SOSCPR * T_{TIM2}

20. 指令集汇总

一共 49 条指令，大部分指令为单周期。

助记符		说明	周期	影响标志
STR	f	将 W 送至 f	1	
NOP	–	空操作	1	
MOVLB	k	将立即数传送到 BSREG	1	
CLRWDT	–	清零看门狗定时器	1	TO, PD
RETI	–	中断返回	2	
RET	–	从子程序返回	2	
BRW	–	将 W 寄存器的内容作为偏移量进行相对跳转	2	
CALLW	–	调用地址由 W 寄存器指定的子程序	2	
RESET	–	软件器件复位	1	
MOVIW	n mm	将间接 FSRn 的内容传送到 W 寄存器，带预/后增/减修改符，mm	1	Z
MOVWI	n mm	将 W 寄存器的内容传送到间接 FSRn，带预/后增/减修改符，mm	1	
SLEEP	–	进入待机模式	1	TO, PD
CLRR	f	将 f 清零	1	Z
CLRW	–	将 W 清零	1	Z
SUBWR	f, d	f 减 W	1	C, DC, Z
DECR	f, d	f 减 1 操作	1	Z
IORWR	f, d	W 与 f 同或	1	Z
ANDWR	f, d	W 与 f 相与	1	Z
XORWR	f, d	W 与 f 异或	1	Z
ADDWR	f, d	W 与 f 相加	1	C, DC, Z
LDR	f, d	传送 f	1	Z
COMR	f, d	求 f 的补码	1	Z
INCR	f, d	f 加 1 操作	1	Z
DECRSZ	f, d	f 减 1 操作，若为 0 则跳过	1	
RRR	f, d	f 寄存器带进位位右循环	1	C
RLR	f, d	f 寄存器带进位位左循环	1	C
SWAPR	f, d	f 半字节交换	1	
INCRSZ	f, d	f 加 1 操作，若为 0 则跳过	1	
BCR	f, b	将 f 位清零	1	
BSR	f, b	将 f 位置 1	1	
BTSC	f, b	测试 f 位，若为 0 则跳过	1	
BTSS	f, b	测试 f 位，若为 1 则跳过	1	
LCALL	k	调用子程序	2	
LJUMP	k	转移	2	
LDWI	k	立即数移至 W	1	
MOVLP	k	将立即数传送到 PCLATH	1	
ADDFSR	n, k	立即数 k 与 FSRn 相加	1	
BRA	k	相对跳转	2	
RETW	k	立即数送到 W 中返回	2	
LSLF	f, d	逻辑左移	1	C 和 Z
LSRF	f, d	逻辑右移	1	C 和 Z
ASRF	f, d	算术右移	1	C 和 Z
IORWI	k	立即数与 W 同或	1	Z
ANDWI	k	立即数与 W 相与	1	Z
XORWI	k	立即数与 W 异或	1	Z
SUBWFB	f, d	f 减去 W（带借位）	1	C, DC, Z
SUBWI	k	立即数减 W	1	C, DC, Z
ADDWFC	f, d	W 和 f 进行带进位的相加	1	C, DC, Z
ADDWI	k	立即数与 W 相加	1	C, DC, Z
MOVIW	k[n]	将 INDFn 的内容传送到 W 寄存器，采用变址间接寻址模式	1	Z
MOVWI	k[n]	将 W 寄存器的内容传送到 INDFn，采用变址间接寻址模式	1	

表 20.1 指令集

操作码字段说明

字段	说明
f	文件寄存器地址（0x00 到 0x7F）
W	工作寄存器（累加器）
b	8 位文件寄存器内的位地址
k	立即数字段、常数或标号
x	忽略（或 0 或 1），汇编器将生成 x = 0 的代码。
D	目标寄存器选择，d = 0：结果保存至 W；d = 1：结果保存至文件寄存器 f。默认值为 d = 1。
N	FSR 或 INDF 编号（0-1）
mm	预/后增/减模式选择

缩写说明

字段	说明
PC	程序计数器
TO	WDT 超时位
C	进位位
DC	半进位位
Z	全零标志位
PD	掉电位（睡眠）

20.1. 读-修改-写（RMW）指令

所有需要使用文件寄存器（表格 20.1 中助记符带 f 的指令）的指令都会执行读-修改-写（RMW）操作，即先把目标寄存器内容取出，根据指令修改数据，再把数据写回到目标寄存器或 W（取决于 d 和具体指令）。

举例说明：

BSR FSR0L, 0;

上述指令在 CPU 的执行过程如下：

1）把 FSR0L 读出到临时寄存器 T；

2）把寄存器 T 或上”0000 0001”形成新数据；

3）再把新数据写回 FSR0L；

20.2. 指令详细描述

ADDFSR	立即数与FSRn 相加
语法：	[标号] ADDFSR FSRn, k
操作数：	-32≤k≤31 n∈[0,1]
操作：	FSR(n)+k → FSR(n)
受影响的状态位：	无
说明：	将有符号6 位立即数k与FSRnH:FSRnL寄存器对的内容相加。 FSRn 地址范围限制为0000h-FFFFh。传送地址超出该边界时，FSR会发生折回。

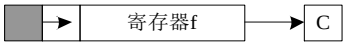
ADDWI	立即数与W相加
语法：	[标号] ADDWI k
操作数：	0≤k≤255
操作：	(W)+k → (W)
受影响的状态位：	C、DC和Z
说明：	将W寄存器的内容与8位立即数k相加，结果存入W寄存器。

ADDWR	W与f相加
语法：	[标号]ADDWR f,d
操作数：	0≤f≤127 d∈[0,1]
操作：	(W)+(f)→(目标寄存器)
受影响的状态位：	C、DC 和Z
说明：	将W寄存器的内容与寄存器f的内容相加。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。

ANDWI	立即数和W作逻辑与运算
语法：	[标号] ANDWI k
操作数：	0≤k≤255
操作：	(W).AND.(k) → (W)
受影响的状态位：	Z
说明：	将W寄存器的内容与8 位立即数k进行逻辑与运算。结果存入W寄存器。

ANDWR	W和f作逻辑与运算
语法：	[标号] ANDWR f,d
操作数：	0≤f≤127 d∈[0,1]
操作：	(W).AND.(f) →(目标寄存器)
受影响的状态位：	Z
说明：	将W 寄存器的内容与寄存器f的内容进行逻辑与运算。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。

ASRF	算术右移
语法：	[标号] ASRF f {,d}
操作数：	0≤f≤127 d∈[0,1]
操作：	(f<7>)→目标寄存器<7> (f<7:1>)→目标寄存器<6:0>， (f<0>)→C
受影响的状态位：	C和Z
说明：	将寄存器f的内容连同进位标志位一起右移1位。MSb保持不变。如果d为0，结果存入W寄存器。如果d为1,结果存回寄存器f。



ADDWFC	W与f相加（带进位）
语法：	[标号] ADDWFC f {,d}
操作数：	0≤f≤127 d∈[0,1]
操作：	(W)+(f)+(C)→目标寄存器
受影响的状态位：	C、DC 和Z
说明：	将W的内容、进位标志位与数据存储单元f的内容相加。如果d为0，结果存入W。如果d为1，结果存入数据存储单元f。

BRW	将W寄存器的内容作为偏移量进行相对跳转
语法：	[标号] BRW
操作数：	无
操作：	(PC)+(W)→PC
受影响的状态位：	无
说明：	将W的内容（无符号）与PC相加。由于PC将递增1以取出下一条指令，所以新地址将为PC+1+(W)。该指令为一条双周期指令。

BCR	将f寄存器中的某位清零
语法：	[标号] BCR f, b
操作数：	0≤f≤127 0≤b≤7
操作：	0→(f<b>)
受影响的状态位：	无
说明：	将寄存器f中的位b清零。

BSR	将f中的某位置1
语法：	[标号] BSR f,b
操作数：	0≤f≤127 0≤b≤7
操作：	1→(f<b>)
受影响的状态位：	无
说明：	将寄存器f的位b置1。

BRA	相对跳转
语法：	[标号] BRA 标号 [标号] BRA $+k
操作数：	-256≤标号 - PC+1≤255 -256≤k≤255
操作：	(PC)+1+k→PC
受影响的状态位：	无
说明：	将有符号9位立即数k与PC相加。由于PC 将递增1以便取下一条指令，所以新地址将为PC+1+k。该指令为一条双周期指令。该跳转的地址范围存在限制。

BTSC	测试f中某位，为0则跳过
语法：	[标号] BTSC f,b
操作数：	0≤f≤127 0≤b≤7
操作：	如果(f<b>)=0，则跳过
受影响的状态位：	无
说明：	如果寄存器f的位b为1，则执行下一条指令。如果寄存器f的位b为0，则丢弃下一条指令，转而执行一条NOP指令，从而使该指令成为双周期指令。

BTSS	测试f中某位,为1则跳过
语法：	[标号] BTSS f,b
操作数：	0≤f≤127 0≤b≤7
操作：	如果(f<b>)=1，则跳过
受影响的状态位：	无
说明：	如果寄存器f的位b为0，则执行下一条指令。如果位b为1，则丢弃下一条指令，转而执行一条NOP指令，从而使该指令成为双周期指令。

CLRW	将W寄存器清零
语法：	[标号] CLRW
操作数：	无
操作：	00h→(W) 1→Z
受影响的状态位：	Z
说明：	W寄存器被清零。全零位（Z）被置1。

LCALL	调用子程序
语法：	[标号] LCALL k
操作数：	0≤k≤2047
操作：	(PC)+1→TOS， k→PC<10:0>， (PCLATH<4:3>)→PC<12:11>
受影响的状态位：	无
说明：	调用子程序。首先，将返回地址（PC+1）压入堆栈。将11位立即数地址装入PC的<10:0>位。将PCLATH的内容装入PC的高位。LCALL是双周期指令。

CALLW	调用地址由W寄存器指定的子程序
语法：	[标号] CALLW
操作数：	无
操作：	(PC)+1→TOS， (W)→PC<7:0>， (PCLATH<6:0>)→PC<14:8>
受影响的状态位：	无
说明：	调用地址由W寄存器指定的子程序。首先，将返回地址（PC+1）压入返回堆栈。然后，W的内容被装入PC<7:0>，将PCLATH的内容装入PC<14:8>。CALLW是双周期指令。

CLRR	将f清零
语法：	[标号] CLRR f
操作数：	0≤f≤127
操作：	00h→(f) 1→Z
受影响的状态位：	Z
说明：	寄存器f的内容被清零，并且Z位被置1。

CLRWDT	将看门狗定时器清零
语法：	[标号] CLRWDT
操作数：	无
操作：	00h→WDT 0→WDT 预分频器 1→/TO 1→/PD
受影响的状态位：	/TO和/PD
说明：	CLRWDT指令复位看门狗定时器及其预分频器。 状态位/TO和/PD均被置1。

COMR	f取反
语法：	[标号] COMR f,d
操作数：	0≤f≤127 d∈[0,1]
操作：	$(\bar{f})$→(目标寄存器)
受影响的状态位：	Z
说明：	将寄存器f的内容取反。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。

LJUMP	无条件跳转
语法：	[标号] LJUMP k
操作数：	0≤k≤2047
操作：	k→PC<10:0> PCLATH<4:3>→PC<12:11>
受影响的状态位：	无
说明：	LJUMP是无条件跳转指令。将11位立即数值装入PC 的<10:0> 位。PC的高位从PCLATH<4:3>装入。LJUMP是双周期指令。

DECR	f递减1
语法：	[标号] DECR f,d
操作数：	0≤f≤127 d∈[0,1]
操作：	(f)-1→(目标寄存器)
受影响的状态位：	Z
说明：	将寄存器f的内容递减1。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。

INCR	f递增1
语法：	[标号] INCR f,d
操作数：	0≤f≤127 d∈[0,1]
操作：	(f)+1→(目标寄存器)
受影响的状态位：	Z
说明：	将寄存器f的内容递增1。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。

DECRSZ	f递减1，为0则跳过
语法：	[标号]DECRSZ f,d
操作数：	0≤f≤127 d∈[0,1]
操作：	(f)-1→(目标寄存器)； 结果=0则跳过
受影响的状态位：	无
说明：	将寄存器f的内容递减1。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。如果结果为1，则执行下一条指令。如果结果为0，则转而执行一条NOP指令，从而使该指令成为双周期指令。

INCRSZ	f递增1，为0则跳过
语法：	[标号] INCRSZ f,d
操作数：	0≤f≤127 d∈[0,1]
操作：	(f)+1→(目标寄存器)， 结果=0则跳过
受影响的状态位：	无
说明：	将寄存器f的内容递增1。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。如果结果为1，则执行下一条指令。如果结果为0，则转而执行NOP指令，从而使该指令成为双周期指令。

IORWI 立即数和W作逻辑或运算

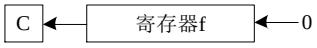
语法：[标号] IORWI k
操作数：0≤k≤255
操作：(W).OR.k→(W)
受影响的状态位：Z
说明：将W寄存器的内容与8位立即数k进行逻辑或运算。结果存入W寄存器。

IORWR W和f作逻辑或运算

语法：[标号] IORWR f,d
操作数：0≤f≤127
d∈[0,1]
操作：(W).OR.(f)→(目标寄存器)
受影响的状态位：Z
说明：将W寄存器的内容与寄存器f的内容进行逻辑或运算。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。

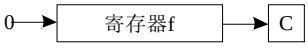
LSLF 逻辑左移

语法：[标号] LSLF f {,d}
操作数：0≤f≤127
d∈[0,1]
操作：(f<7>)→C
(f<6:0>)→目标寄存器<7:1>
0→目标寄存器<0>
受影响的状态位：C和Z
说明：将寄存器f的内容连同进位标志位一起左移1位。0 移入LSb。如果d为0，结果存入W。如果d为1，结果存回寄存器f。



LSRF 逻辑右移

语法：[标号] LSRF f {,d}
操作数：0≤f≤127
d∈[0,1]
操作：0→目标寄存器<7>
(f<7:1>)→目标寄存器<6:0>
(f<0>)→C
受影响的状态位：C和Z
说明：将寄存器f的内容连同进位标志位一起右移1位。0 移入MSb。如果d为0，结果存入W。如果d为1，结果存回寄存器f。



LDR 传送f

语法：[标号] LDR f,d
操作数：0≤f≤127
d∈[0,1]
操作：(f)→(目标寄存器)
受影响的状态位：Z
说明：根据d的状态，将寄存器f的内容传送到目标寄存器。如果d=0，目标寄存器为W寄存器。如果d=1，目标寄存器为文件寄存器f。由于状态标志位Z要受影响，可用d=1 检测文件寄存器。
指令字数：1
指令周期数：1

示例：LDR FSR, 0
执行指令后
W=FSR寄存器的值
Z=1

MOVIW 将INDFn的内容传送到W

语法： [标号] MOVIW ++FSRn
[标号] MOVIW --FSRn
[标号] MOVIW FSRn++
[标号] MOVIW FSRn--
[标号] MOVIW k[FSRn]

操作数： n∈[0,1]
mm∈[00,01,10,11]
-32≤k≤31

操作： INDFn→W
有效地址通过以下方式确定：
•FSR+1（预递增1）
•FSR-1（预递减1）
•FSR+k（相对偏移）
执行传送指令后，FSR值为以下任一项：
•FSR+1（所有值都加1）
•FSR-1（所有值都减1）
•不变

受影响的状态位：Z

模式	语法	mm
预递增	++FSRn	00
与递减	--FSRn	01
后递增	FSRn++	10
后递减	FSRn--	11

说明： 该指令用于在W寄存器和任一间接寄存器（INDFn）之间传送数据。执行该传送指令之前/之后，将通过预/后增/减来更新指针（FSRn）。

注：INDFn 寄存器不是物理寄存器。访问INDFn寄存器的指令实际上访问的是由FSRn指定的地址处的寄存器。

FSRn地址范围限制为0000h-FFFFh。地址递增/递减到超出边界时，将导致它发生折回。

MOVWI 将W 的内容传送到INDFn

语法： [标号] MOVWI ++FSRn
[标号] MOVWI --FSRn
[标号] MOVWI FSRn++
[标号] MOVWI FSRn--
[标号] MOVWI k[FSRn]

操作数： n∈[0,1]
mm∈[00,01,10,11]
-32≤k≤31

操作： W→INDFn
有效的地址由以下项决定:
•FSR+1（预递增1）
•FSR-1（预递减1）
•FSR+k（相对偏移）
执行传送指令后，FSR值为以下任一项：
•FSR+1（所有值都加1）
•FSR-1（所有值都减1）
不变

受影响的状态位：无

模式	语法	mm
预递增	++FSRn	00
与递减	--FSRn	01
后递增	FSRn++	10
后递减	FSRn--	11

说明： 该指令用于在W寄存器和任一个间接寄存器（INDFn）之间传送数据。执行该传送指令之前/之后，将通过预/后增/减来更新指针（FSRn）。

注：INDFn寄存器不是物理寄存器。访问INDFn寄存器的指令实际上访问的是由FSRn指定的地址处的寄存器。

FSRn地址范围限制为0000h-FFFFh。地址递增/递减到超出边界时，将导致它发生折回。
对于FSRn的递增/递减操作不会影响任何状态位。

MOVLB 将立即数传送到BSR

语法：	[标号] MOVLB k
操作数：	0≤k≤15
操作：	k→BSR
受影响的状态位：	无
说明：	将5位立即数k装入存储区选择寄存器（BSR）。

MOVLP 将立即数传送到PCLATH

语法：	[标号] MOVLP k
操作数：	0≤k≤127
操作：	k→PCLATH
受影响的状态位：	无
说明：	将7位立即数k装入PCLATH寄存器。

LDWI 将立即数传送到W

语法：	[标号] MOVLW k
操作数：	0≤k≤255
操作：	k→(W)
受影响的状态位：	无
说明：	将8位立即数k装入W寄存器。其余无关位均汇编为0。
指令字数：	1
指令周期数：	1
示例：	LDWI 0x5A 执行指令后 W = 0x5A

NOP 空操作

语法：	[标号] NOP
操作数：	无
操作：	空操作
受影响的状态位：	无
说明：	不执行任何操作。
指令字数：	1
指令周期数：	1
示例：	NOP

STR 将W的内容传送到f

语法：	[标号] STR f
操作数：	0≤f≤127
操作：	(W) →(f)
受影响的状态位：	无
说明：	将W寄存器的数据传送到寄存器f。
指令字数：	1
指令周期数：	1
示例：	STR OPTION 执行指令前 OPTION = 0xFF W = 0x4F 执行指令后 OPTION = 0x4F W = 0x4F

RESET 软件复位

语法：	[标号] RESET
操作数：	无
操作：	执行器件复位。复位PCON 寄存器的nRI标志。
受影响的状态位：	无
说明：	此指令可实现用软件执行硬件复位。

RET 从子程序返回

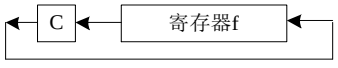
语法：	[标号] RET
操作数：	无
操作：	TOS→PC
受影响的状态位：	无
说明：	从子程序返回。执行出栈操作，将栈顶（TOS）内容装入程序计数器。这是一条双周期指令。

RETI 从中断返回

语法： [标号] RETI
操作数： 无
操作： TOS→PC，
1→GIE
受影响的状态位：无
说明： 从中断返回。执行出栈操作，将栈顶（Top-of-Stack，TOS）的内容装入PC。通过将全局中断允许位GIE（INTCON<7>）置1，来允许中断。这是一条双周期指令。
指令字数： 1
指令周期数： 2
示例： RETI
中断后
PC = TOS
GIE = 1

RLR 对f执行带进位的循环左移

语法： [标号] RLR f,d
操作数： 0≤f≤127
d∈[0,1]
操作： 参见如下说明
受影响的状态位：C
说明： 将寄存器f 的内容连同进位标志位一起循环左移1位。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。
指令字数： 1
指令周期数： 1
示例： RLF REG1,0
执行指令前:
REG1 = 1110 0110
C = 0
执行指令后:
REG1 = 1110 0110
W = 1100 1100
C = 1



RETW 返回并将立即数送入W

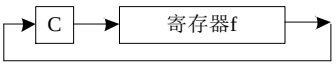
语法： [标号] RETW k
操作数： 0≤k≤255
操作： k→(W);
TOS→PC
受影响的状态位：无
说明： 将8位立即数k装入W寄存器。将栈顶内容（返回地址）装入程序计数器。这是一条双周期指令。
指令字数： 1
指令周期数： 2
示例：

```
        LCALL TABLE;W contains
    ;table offset value
        • ;W now has table value
        •
TABLE   •
        ADDWR   PC   ;W = offset
        RETW    k1   ;Begin table
        RETW    k2   ;
        •
        •
        •
        RETW    kn   ;End of table
```

执行指令前
W = 0x07
执行指令后
W = k8的值

RRR 对f执行带进位的循环右移

语法： [标号] RRR f,d
操作数： 0≤f≤127
d∈[0,1]
操作： 参见如下说明
受影响的状态位：C
说明： 将寄存器f的内容连同进位标志位一起循环右移1位。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f



SLEEP 进入休眠模式

语法：[标号] SLEEP
操作数：无
操作：00h→WDT，
0→WDT 预分频器，
1→/TO，
0→/PD
受影响的状态位：/TO和/PD
说明：掉电状态位/PD被清零。超时状态位/TO被置1。看门狗定时器及其预分频器被清零。振荡器停振，处理器进入休眠模式。

SUBWI 从立即数中减去W

语法：[标号] SUBWI k
操作数：0≤k≤255
操作：k-(W)→(W)
受影响的状态位：C、DC和Z
说明：用8位立即数k减去W寄存器的内容（通过二进制补码方式进行运算）。结果存入W寄存器。

C= 0	W＞k
C= 1	W≤k
DC=0	W<3:0> > k<3:0>
DC=1	W<3:0> ≤ k<3:0>

SUBWFB f减去W（带借位）

语法：SUBWFB f {,d}
操作数：0≤f≤127
d∈[0,1]
操作：(f)–(W)–(/B)→目标寄存器
受影响的状态位：C、DC和Z
说明：用f寄存器的内容减去W的内容和借位标志（进位）（通过二进制补码方式进行运算）。如果d为0，结果存入W。如果d为1，结果存回寄存器f。

SWAPR 将f中的两个半字节交换

语法：[标号] SWAPR f,d
操作数：0≤f≤127
d∈[0,1]
操作：(f<3:0>)→(目标寄存器<7:4>)，
(f<7:4>)→(目标寄存器<3:0>)
受影响的状态位：无
说明：寄存器f的高半字节和低半字节相互交换。如果d为0，结果存入W 寄存器。如果d为1，结果存回寄存器f。

SUBWR f减去W

语法：[标号] SUBWR f,d
操作数：0≤f≤127
d∈[0,1]
操作：(f)-(W)→(目标寄存器)
受影响的状态位：C、DC和Z
说明：用寄存器f的内容减去W寄存器的内容（通过二进制补码方式进行运算）。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。

C= 0	W＞f
C= 1	W≤f
DC=0	W<3:0> > f<3:0>
DC=1	W<3:0> ≤ f<3:0>

XORWR W和f作逻辑异或运算

语法：[标号] XORWR f,d
操作数：0≤f≤127
d∈[0,1]
操作：(W).XOR.(f)→(目标寄存器)
受影响的状态位：Z
说明：将W寄存器的内容与寄存器f的内容进行逻辑异或运算。如果d为0，结果存入W寄存器。如果d为1，结果存回寄存器f。

XORWI 立即数和W作逻辑异或运算

语法：	[标号] XORWI k
操作数：	0≤k≤255
操作：	(W).XOR.k→(W)
受影响的状态位：	Z
说明：	将W寄存器的内容与8位立即数k进行逻辑异或运算。结果存入W寄存器。

21. 芯片的电气特性

21.1. 极限参数

工作温度……-40~+85°C
存储温度……-40~+125°C
电源电压……V_{SS}-0.3V~V_{SS}+6.0V
端口输入电压……V_{SS}-0.3V~V_{DD}+0.3V

21.2. 内置高频振荡器（HIRC）

电气参数	最小值(1)	典型值	最大值(1)	单位	条件/备注
校准范围	15.84	16	16.16	MHz	25°C, V_{DD} = 2.5V
随温度变化范围	-2.0%	—	2.0%	—	-40~85°C, V_{DD} = 2.5V
随电源电压变化范围	-0.5%	—	0.5%	—	25°C, V_{DD} = 1.9~5.5V
I_{HIRC} 工作电流	—	40	—	μA	25°C, V_{DD} = 3.0V
启动时间	—	2.5	—	μs	25°C, V_{DD} = 3.0V

(1) 数据基于特性值，并未生产测试。

21.3. 内置低频振荡器（LIRC）

低频振荡器有双模模式，一种模式下振动频率为 32kHz，另一种模式下振动频率为 256kHz。振荡频率模式由 OSCCON 寄存器中的 LFMOD 位控制，0 为 32kHz 模式，1 为 256kHz 模式。

电气参数	最小值(1)	典型值	最大值(1)	单位	条件/备注
振荡频率	30.4	32	33.6	kHz	25°C, V_{DD} = 2.5V
随温度变化范围	-2.0%	—	2.0%	—	-40~85°C, V_{DD} = 2.5V
随电源电压变化范围	-4.5%	—	1.0%	—	25°C, V_{DD} = 1.9~5.5V
I_{LIRC} 工作电流	—	1.3	—	μA	25°C, V_{DD} = 3.0V
启动时间	—	4.6	—	μs	25°C, V_{DD} = 3.0V

(1) 数据基于特性值，并未生产测试。

21.4. 低电压复位电路（LVR）

电气参数	最小值(1)	典型值	最大值(1)	单位	条件/备注
I_{LVR} 工作电流	—	15.18	—	μA	V_{DD} = 3.3V
V_{LVR}，LVR 阈值	1.94	2.0	2.06	V	25°C
	2.13	2.2	2.27		
	2.42	2.5	2.58		
	2.72	2.8	2.88		
	3.01	3.1	3.19		
	3.49	3.6	3.71		
	3.98	4.1	4.22		
LVR delay	—	125	157	μs	25°C, V_{DD} = 1.9~5.5V

(1) 数据基于特性值，并未生产测试。

21.5. 低电压侦测电路（LVD）

电气参数	最小值(1)	典型值	最大值(1)	单位	条件/备注
I_{LVD} 工作电流	—	21.54	—	μA	V_{DD} = 3.3V
V_{LVD}，LVD 阈值	1.94	2.0	2.06	V	25°C
	2.33	2.4	2.47		
	2.72	2.8	2.88		
	2.91	3.0	3.09		
	3.49	3.6	3.71		
	3.88	4.0	4.12		
LVD delay	—	125	157	μs	25°C, V_{DD} = 1.9~5.5V

(1) 数据基于特性值，并未生产测试。

21.6. 上电复位电路（POR）

电气参数	最小值	典型值(1)	最大值	单位	条件/备注
I_{POR} 工作电流	—	140	—	nA	25°C, V_{DD} = 3.3V
V_{POR}	—	1.65	—	V	25°C

21.7. I/O PAD 电路

电气参数		最小值	典型值(1)	最大值	单位	条件/备注
V_{IL}		0	—	0.3*V_{DD}	V	
V_{IH}		0.7*V_{DD}	—	V_{DD}	V	
漏电流		-1	—	1	μA	V_{DD} = 5V
源电流(source)	L0	—	-2	—	mA	25°C, V_{DD} = 5V, V_{OH} = 4.5V
	L1	—	-4	—		
	L2	—	-14	—		
	L3	—	-26	—		
灌电流(sink)	L0	—	53	—	mA	25°C, V_{DD} = 5V, V_{OL}= 0.5V
	L1	—	62	—		
上拉电阻		—	21	—	kΩ	
下拉电阻		—	21	—	kΩ	

(1) 数据基于特性值，并未生产测试。

21.8. 总体工作电流（I_{DD}）

电气参数	Sysclk	典型值@V_{DD}(1)			单位
		2.0V	3.0V	5.5V	
正常模式(1T)，I_{DD}	16MHz	—	4.143	4.402	mA
	8MHz	1.897	2.648	2.808	
	4MHz	1.293	1.887	1.981	
	2MHz	0.871	1.130	1.183	
	1MHz	0.561	0.727	0.755	
	32kHz	0.036	0.051	0.054	
正常模式(2T)，I_{DD}	16MHz	2.170	3.000	3.181	mA
	8MHz	1.435	2.074	2.169	
	4MHz	0.947	1.224	1.284	
	2MHz	0.596	0.778	0.810	
	1MHz	0.420	0.560	0.581	
	32kHz	0.032	0.046	0.048	
休眠模式（Sleep, WDT OFF, LVR OFF），I_{SB}	—	0.087	0.136	0.240	μA
休眠模式（Sleep, WDT ON, LVR OFF）	—	1.294	2.420	2.854	
休眠模式（Sleep, LVR ON, WDT OFF）	—	11.257	15.318	20.777	
休眠模式（Sleep, LVR ON, WDT ON）	—	12.457	17.551	23.240	
休眠模式（Sleep, LVD ON, LVR OFF, WDT OFF）	—	17.793	21.672	27.133	

(1) 数据基于特性值，并未生产测试。

注意：

1. **测试环境温度为 25°C；**
2. **睡眠电流的测试条件为 I/O 处于输入模式并外部下拉到 0；**

21.9. AC 电气参数

电气参数		最小值	典型值	最大值	单位	条件/备注
Fsys (系统时钟频率)	1T/2T/4T	—	—	8	MHz	-40~85°C , V_{DD} = 1.9~5.5V
		—	—	16	MHz	-40~85°C , V_{DD} = 2.7~5.5V
指令周期（T_{INS}）	1T	62.5	—	—	ns	系统时钟 HIRC
	2T	125	—	—	ns	
	4T	250	—	—	ns	
	1T	30.5	—	—	μs	系统时钟 LIRC
	2T	61	—	—	μs	
	4T	122	—	—	μs	
上电复位保持时间（T_{DRH}）		—	4.2	—	ms	25°C, PWRT disable
外部复位脉冲宽度（T_{MCLRB}）		2000	—	—	ns	25°C
WDT 周期（T_{WDT}）		—	1	—	ms	无预分频, WDTPS<3:0>=0000

注：除特殊说明，特性测试条件为：T=-40~85°C，V_{DD}=1.9~5.5V。

21.10. 12bit ADC 特性

ADC 特性参数

电气参数	最小值(1)	典型值	最大值(1)	单位	条件/备注
ADC 工作电压 V_{DD}	2.7	—	5.5	V	
ADC 工作电流 I_{VDD}	—	630	—	μA	25°C, V_{REFP} = V_{DD} = 2.5V， ADC 转换时钟频率为 250kHz
	—	750	—	μA	25°C, V_{REFP} = V_{DD} = 3.0V， ADC 转换时钟频率为 250kHz
	—	1350	—	μA	25°C, V_{REFP} = V_{DD} = 5.5V， ADC 转换时钟频率为 250kHz
模拟输入电压 V_{AIN}	VREFN	—	VREFP	V	
外部参考电压 V_{REF}	—	—	V_{DD}	V	
分辨率	—	—	12	位	
积分误差 E_{IL}	—	±2	—	LSB	25°C, V_{REFP} = V_{DD} = 5.0V，V_{REFN} = GND，ADC 转换时钟频率为 250kHz
微分误差 E_{DL}	—	±2	—	LSB	
偏移误差 E_{OFF}	—	±1	—	LSB	V_{REFP} = V_{DD} = 5.0V，V_{REFN} = GND
增益误差 E_{GN}	—	±1.5	—	LSB	
转换时钟周期 TAD	—	1	—	μs	V_{REFP} > 3.0V, V_{DD} > 3.0V
转换时钟数	—	15	—	TAD	
稳定时间(T_{ST})	—	1	—	μs	
采样时间(T_{ACQ})	—	1.5	—	TAD	
建议的模拟电压源阻抗（ZAI）	—	—	10	kΩ	—

(1) 数据基于特性值，并未生产测试。

ADC Vref 特性参数

电气参数	最小值(1)	典型值(1)	最大值(1)	单位	条件/备注
内置参考电压 ADCVref	0.492	0.5	0.508	V	25°C, V_{DD} =5V
	1.992	2.0	2.008	V	25°C, V_{DD} =5V
	2.988	3.0	3.012	V	25°C, V_{DD} =5V
内置参考电压 0.5V	—	250	—	μs	25°C, V_{DD} =5V
稳定时间 T_{VRINT}	—	550	—	μs	25°C, V_{DD} =5V, 1μF
内置参考电压 2.0V	—	450	—	μs	25°C, V_{DD} =5V
稳定时间 T_{VRINT}	—	1500	—	μs	25°C, V_{DD} =5V, 1μF
内置参考电压 3.0V	—	300	—	μs	25°C, V_{DD} =5V
稳定时间 T_{VRINT}	—	550	—	μs	25°C, V_{DD} =5V, 1μF

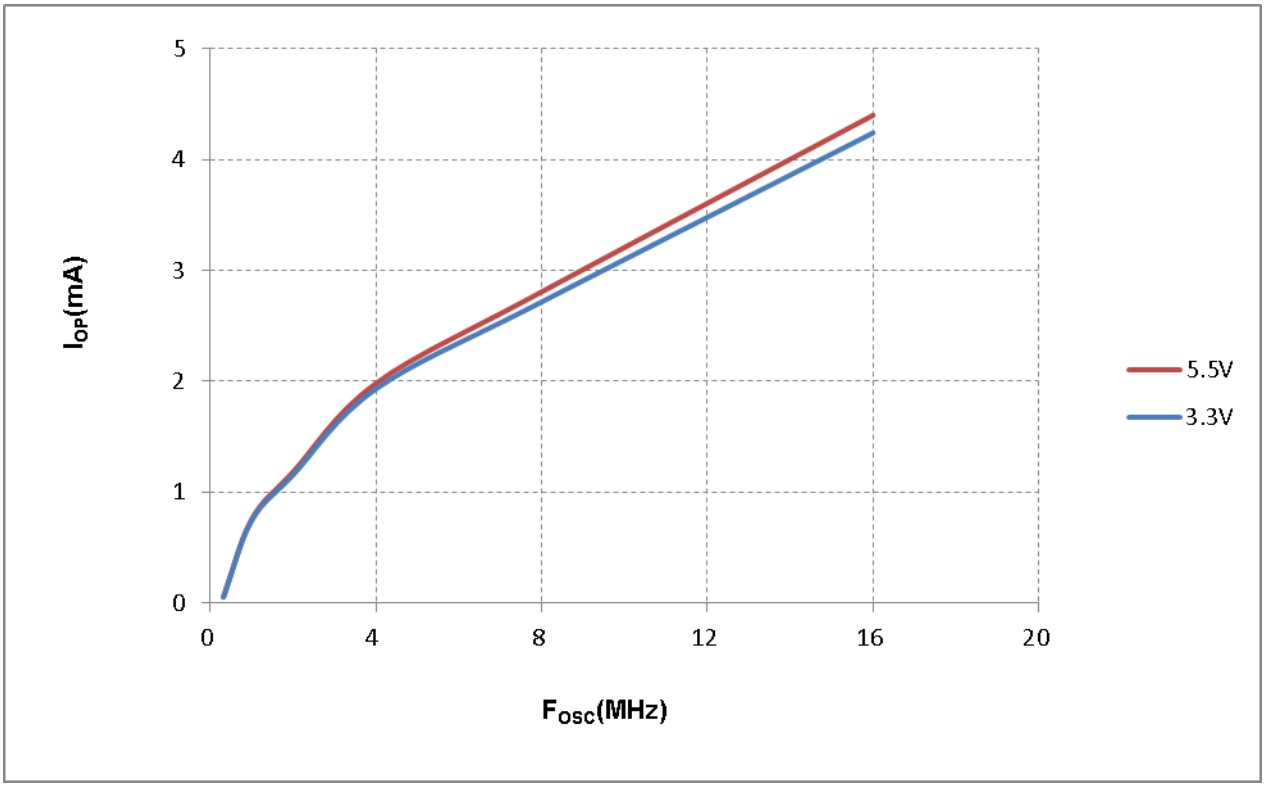
(1) 数据基于特性值，并未生产测试。

除非另外说明，否则“典型”值一栏的数据都是在 5.0V，25°C 的条件下给出。

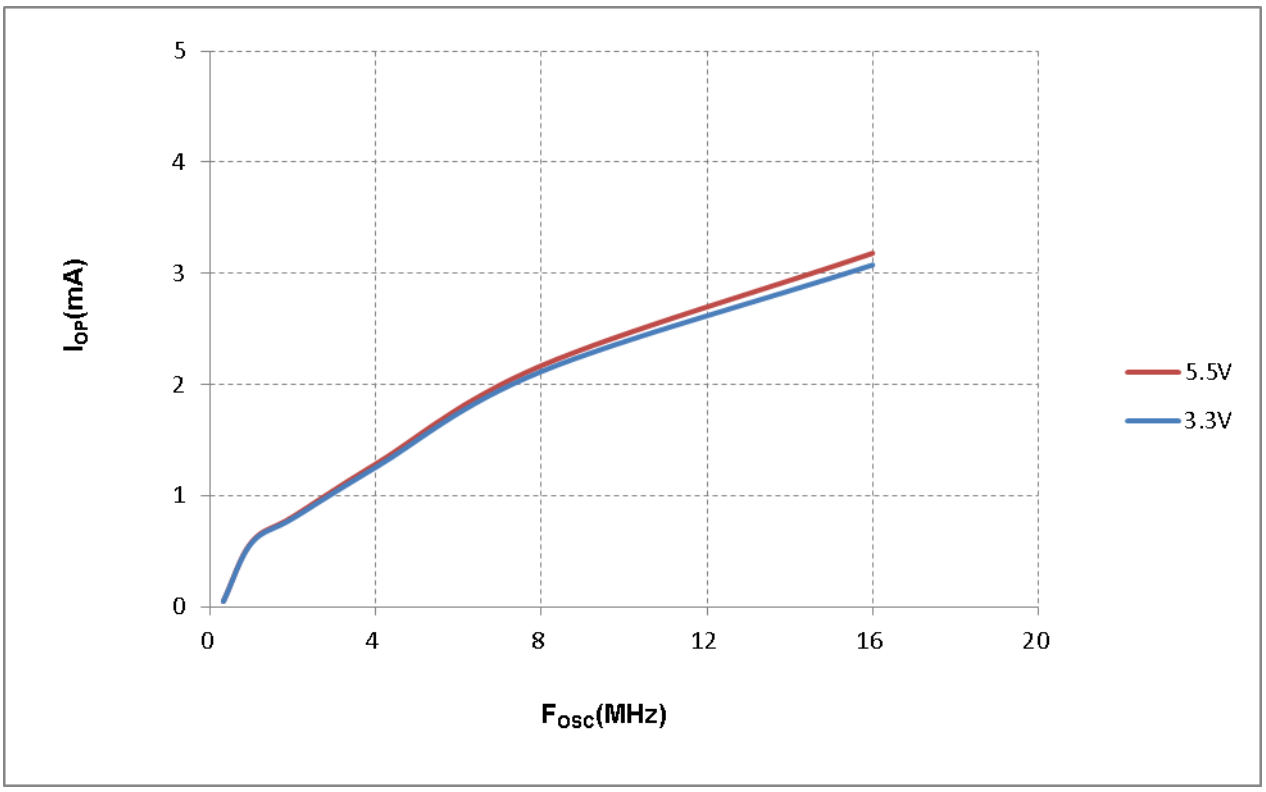
21.11. 直流和交流特性曲线图

注意：本节提供的图表基于特性值，仅用作设计参考，未经生产测试。

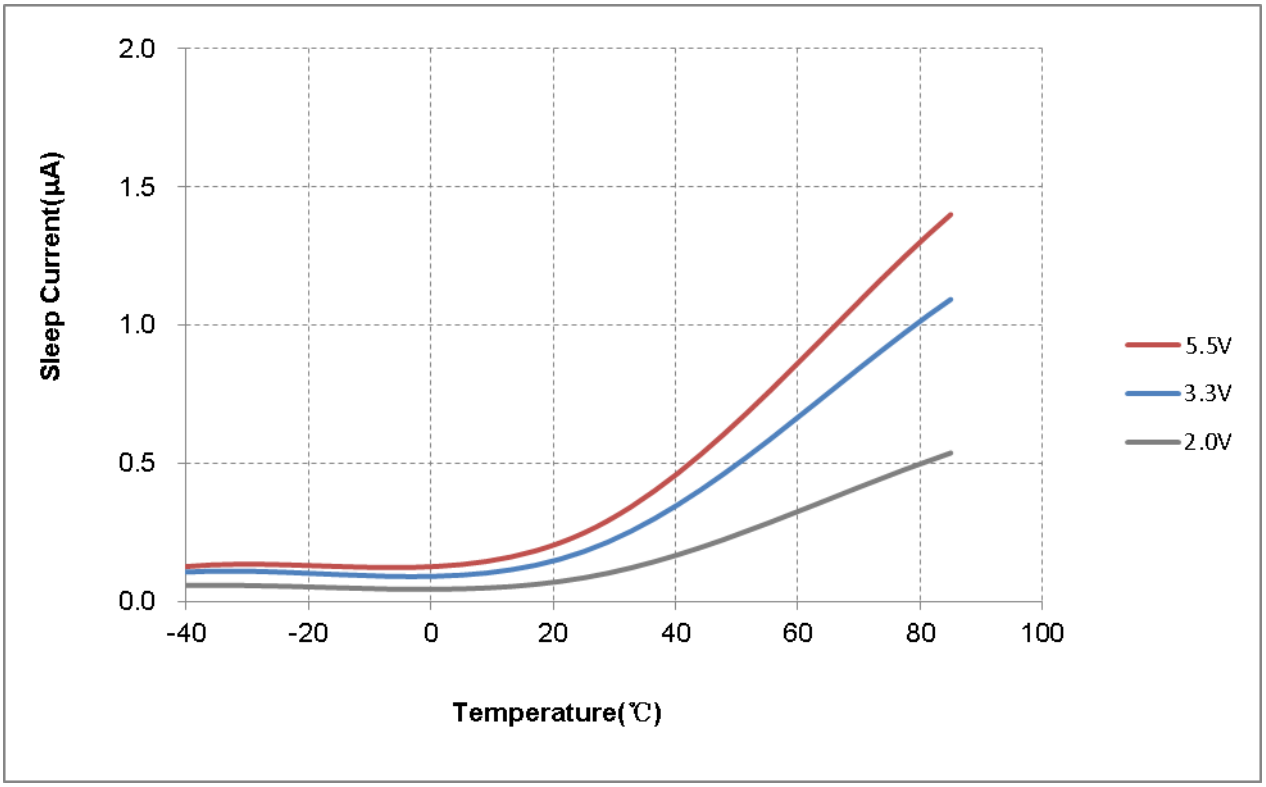
21.11.1. 不同 V_{DD} 下， I_{DD} vs Freq (1T, T_A=25°C)



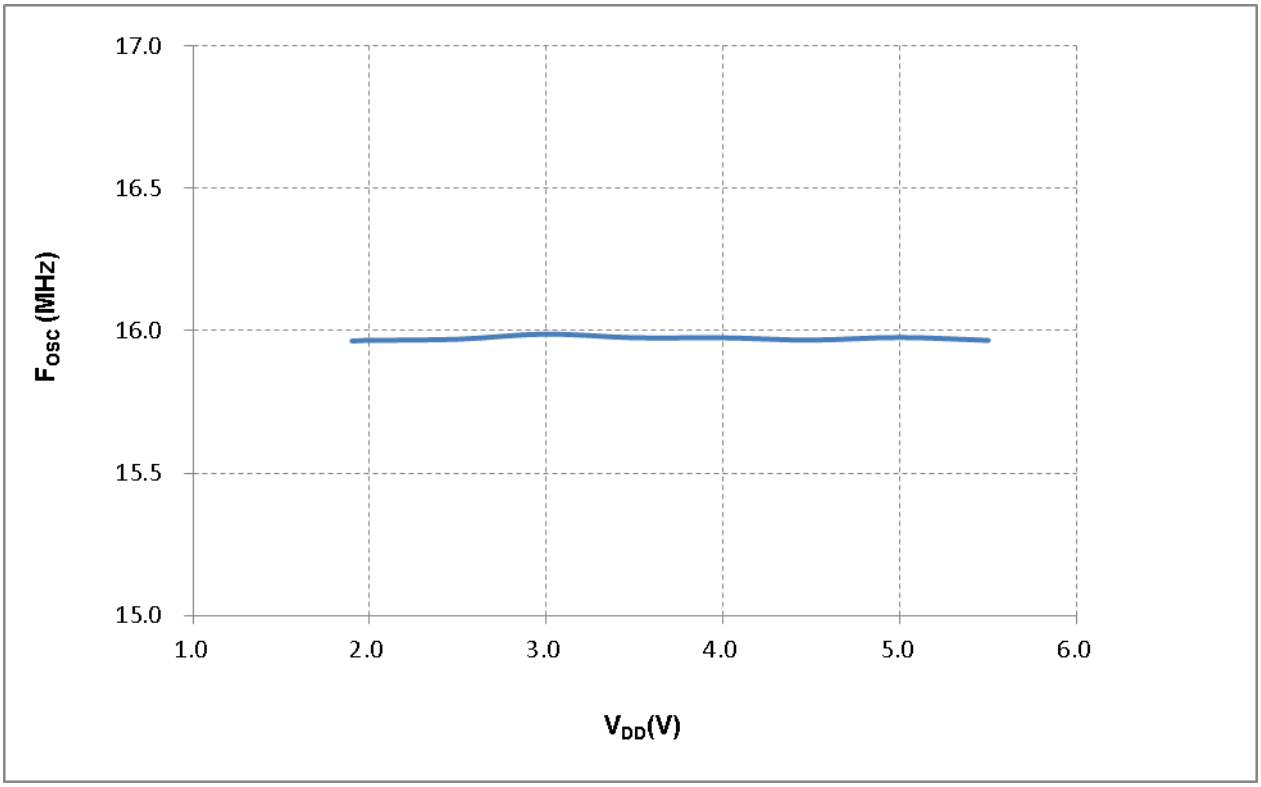
21.11.2. 不同 V_{DD} 下， I_{DD} vs Freq (2T, T_A=25°C)



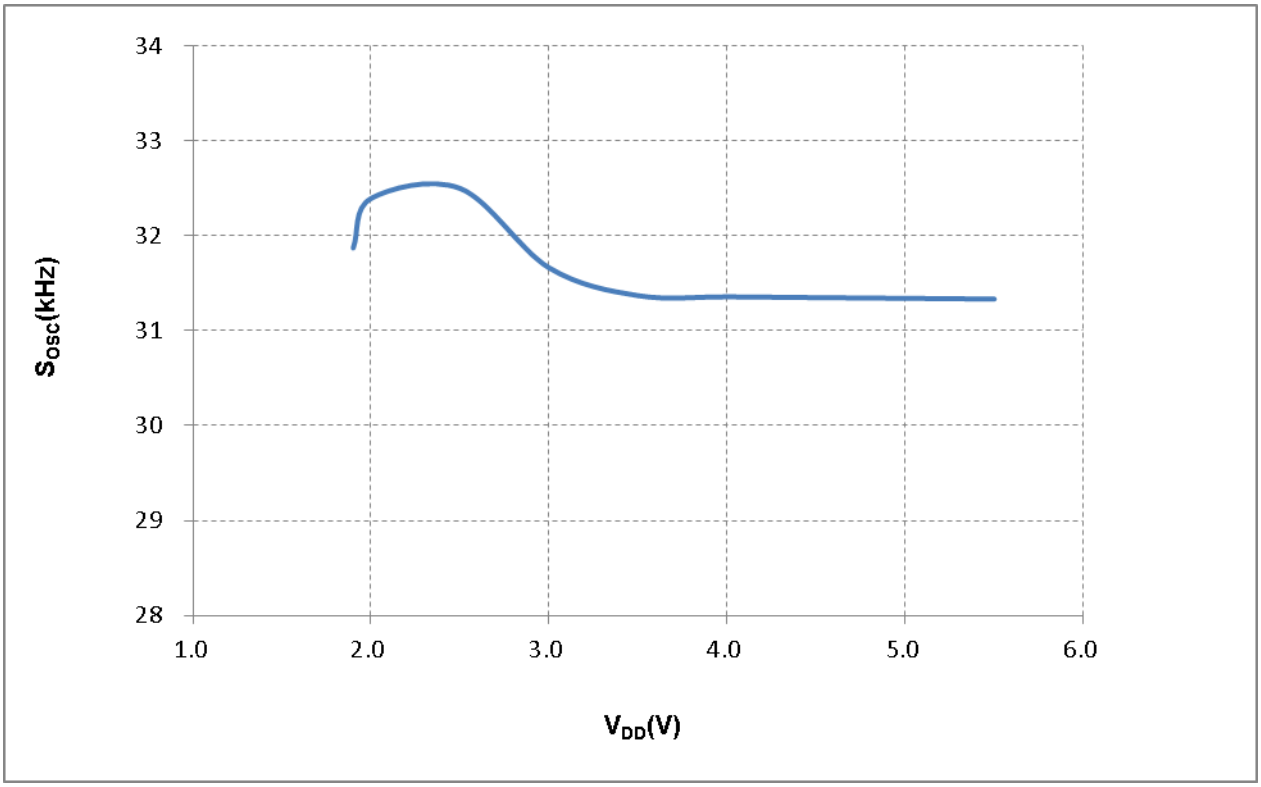
21.11.3. 不同 V_{DD} 下，I_{SB} (睡眠电流) 随温度变化曲线



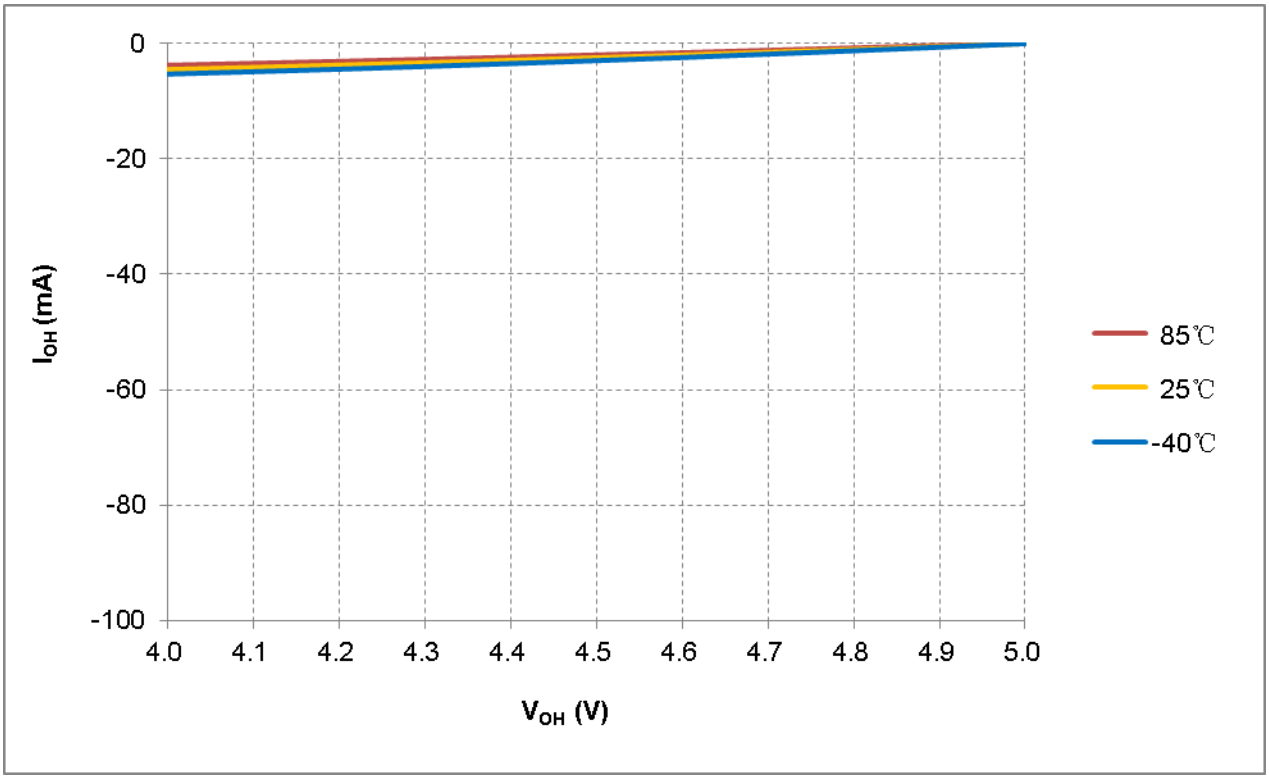
21.11.4. HIRC vs V_{DD} (TA=25°C)



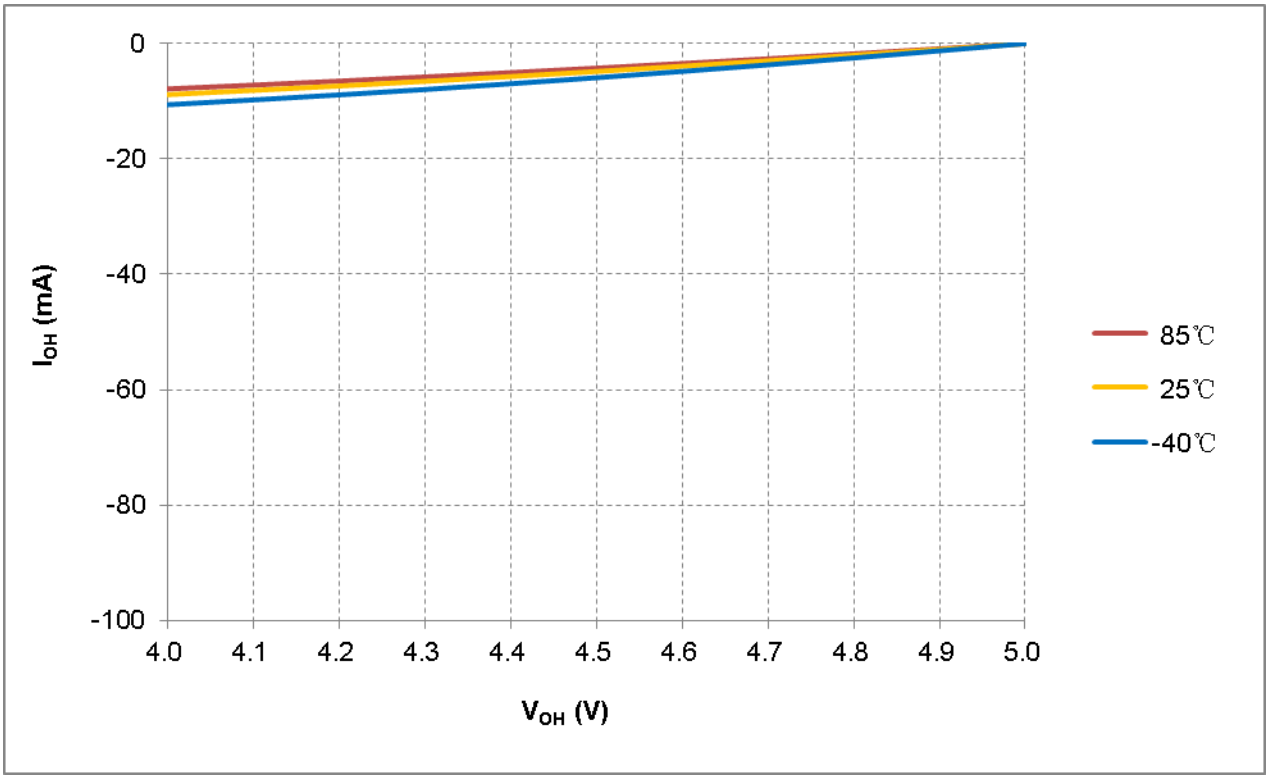
21.11.5. LIRC vs V_{DD} (T_A=25°C)



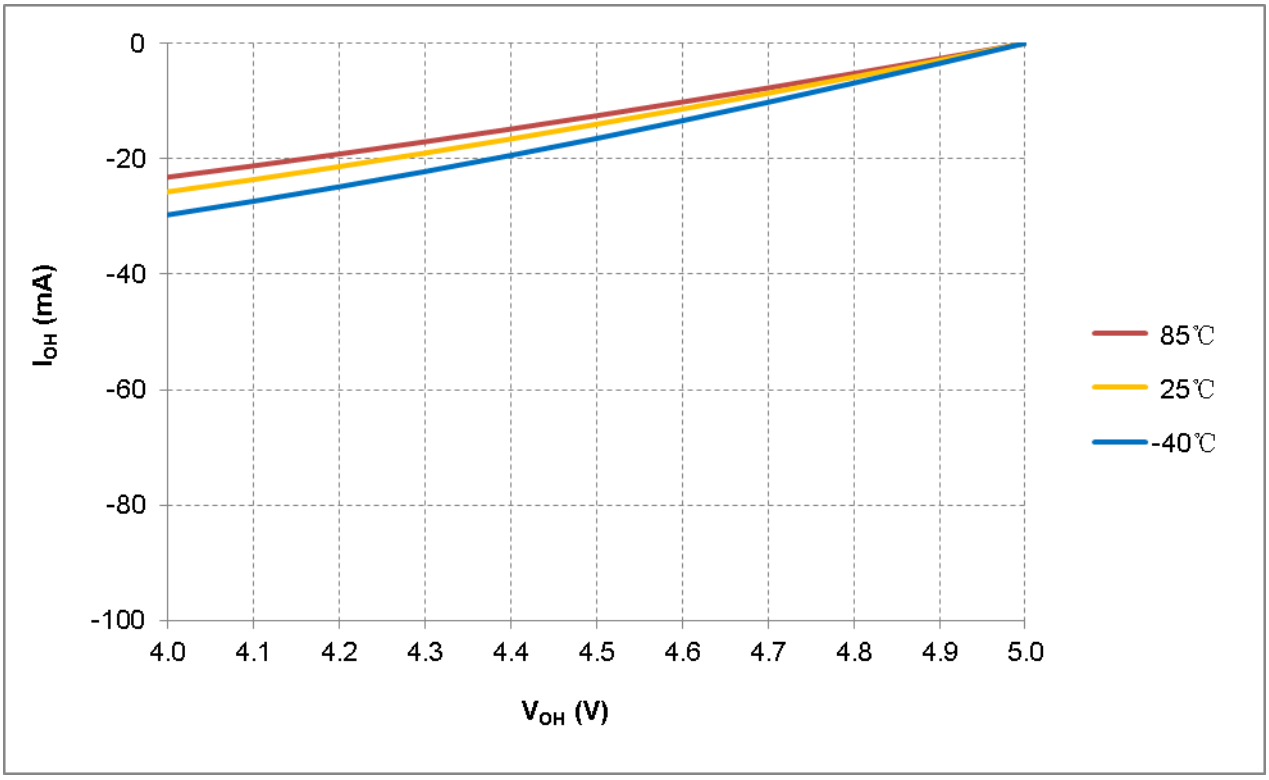
21.11.6. I_{OH} (level -2mA) vs V_{OH} @V_{DD}=5V



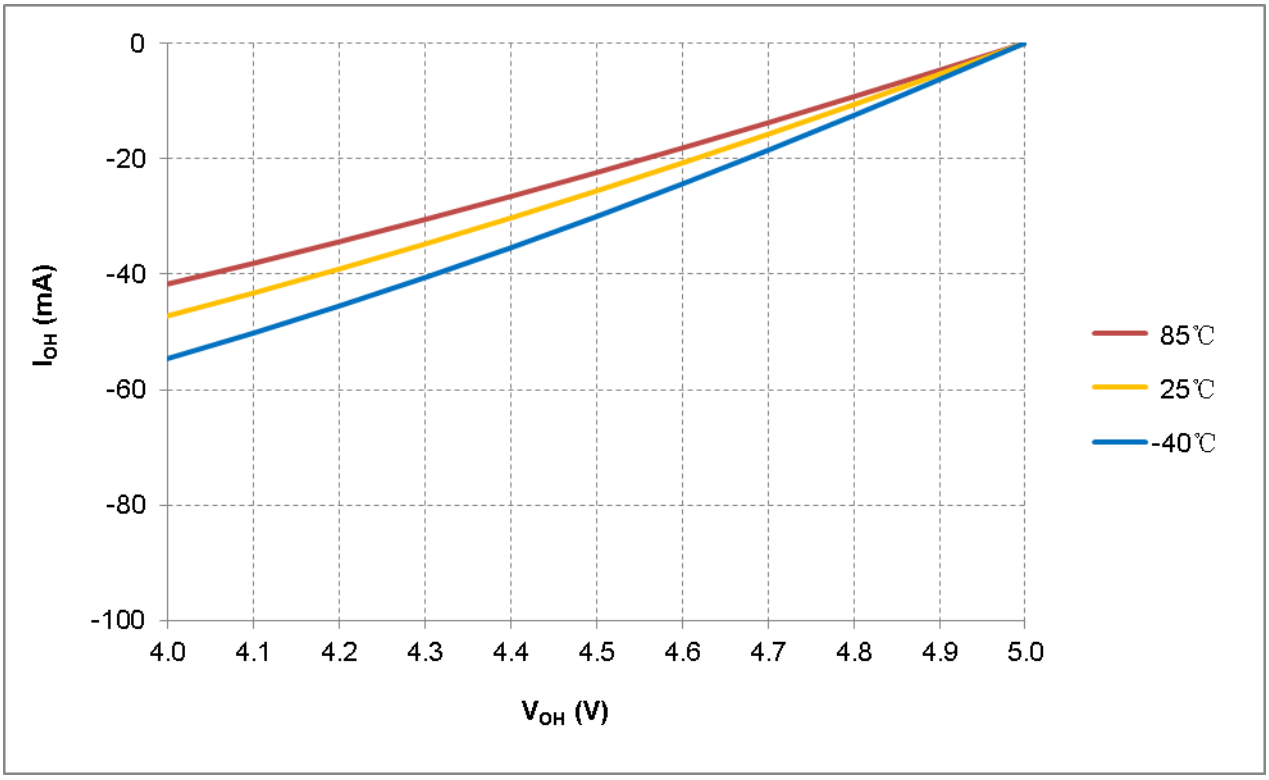
21.11.7. I_{OH} (level -4mA) vs V_{OH} @V_{DD}=5V



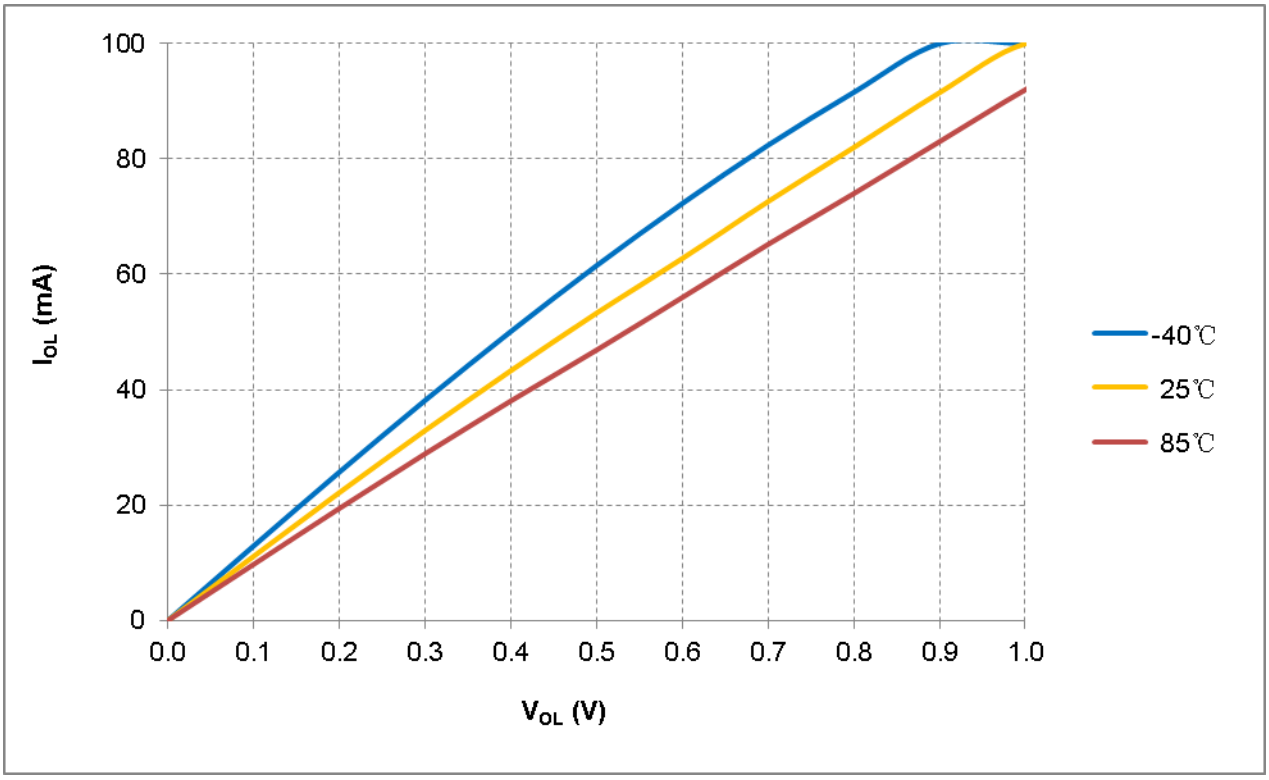
21.11.8. I_{OH} (level -14mA) vs V_{OH} @V_{DD}=5V



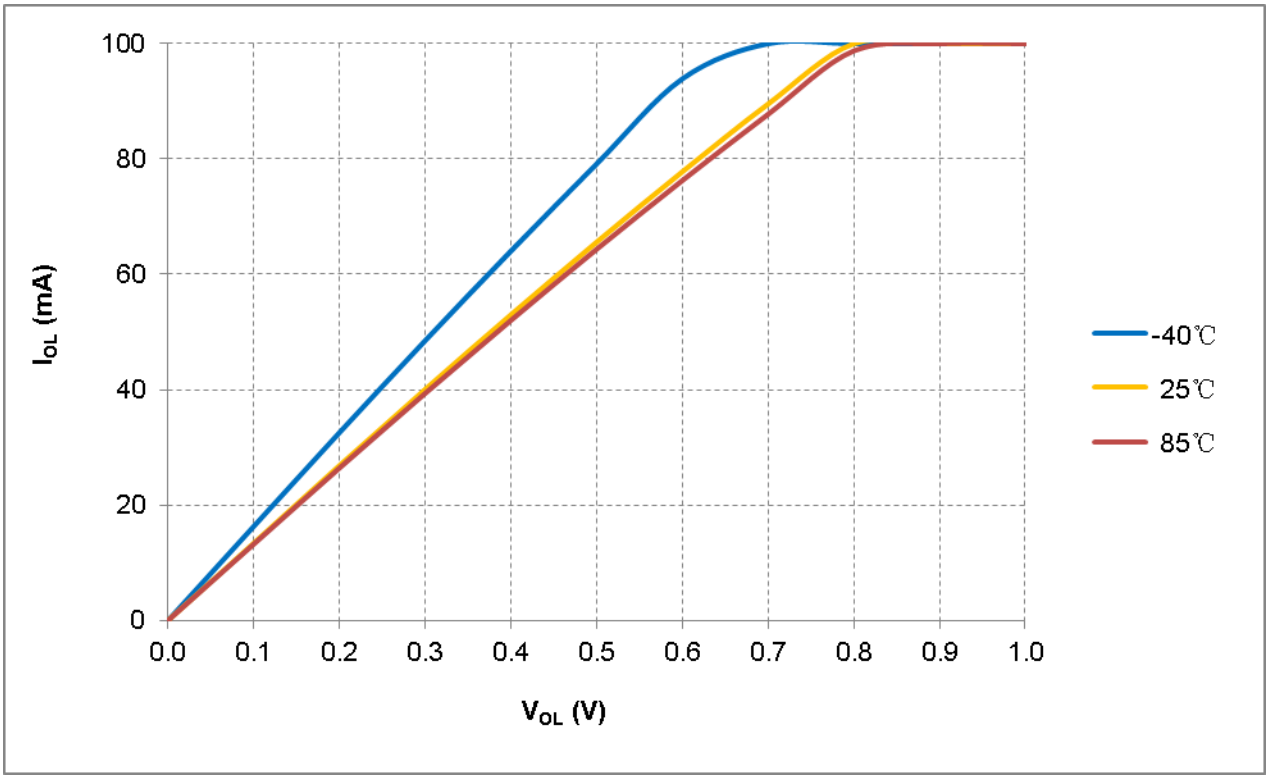
21.11.9. I_{OH} (level -26mA) vs V_{OH} @V_{DD}=5V



21.11.10. I_{OL} (L0) vs V_{OL} @V_{DD}=5V

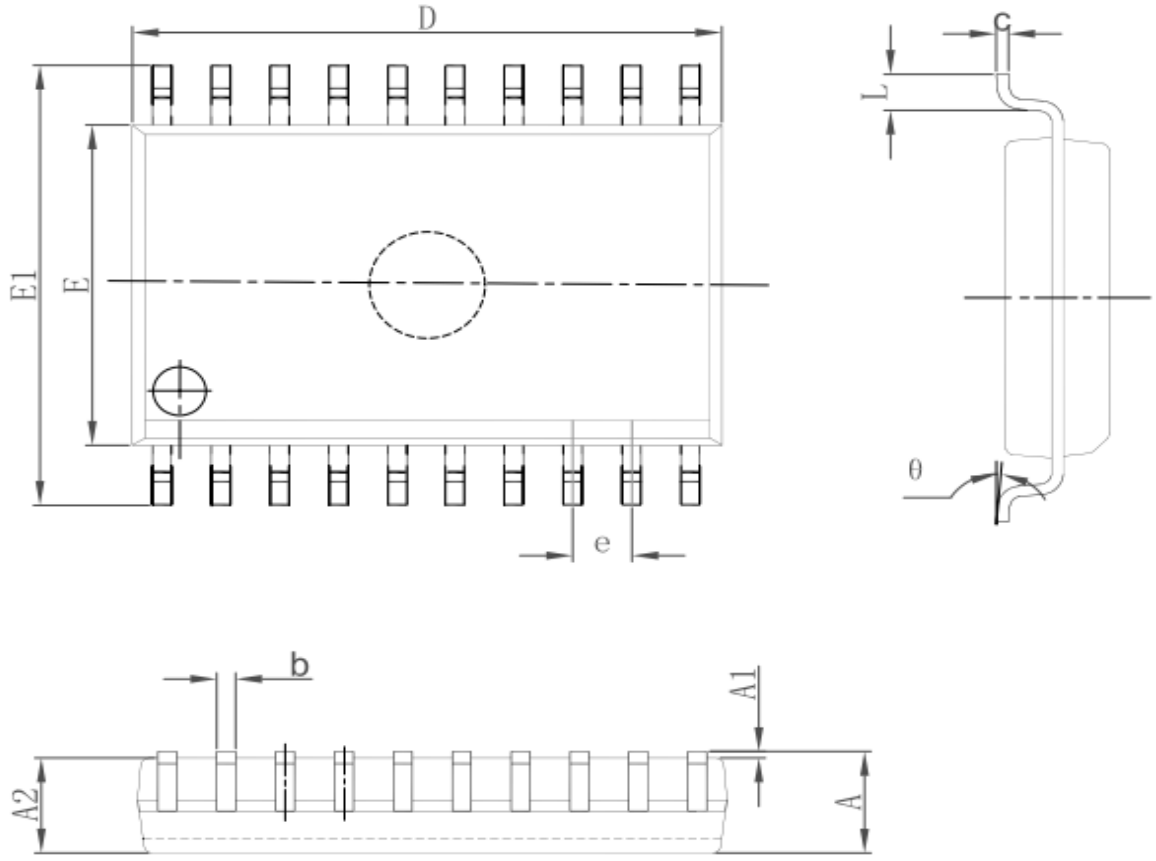


21.11.11. I_{OL} (L1) vs V_{OL} @V_{DD}=5V



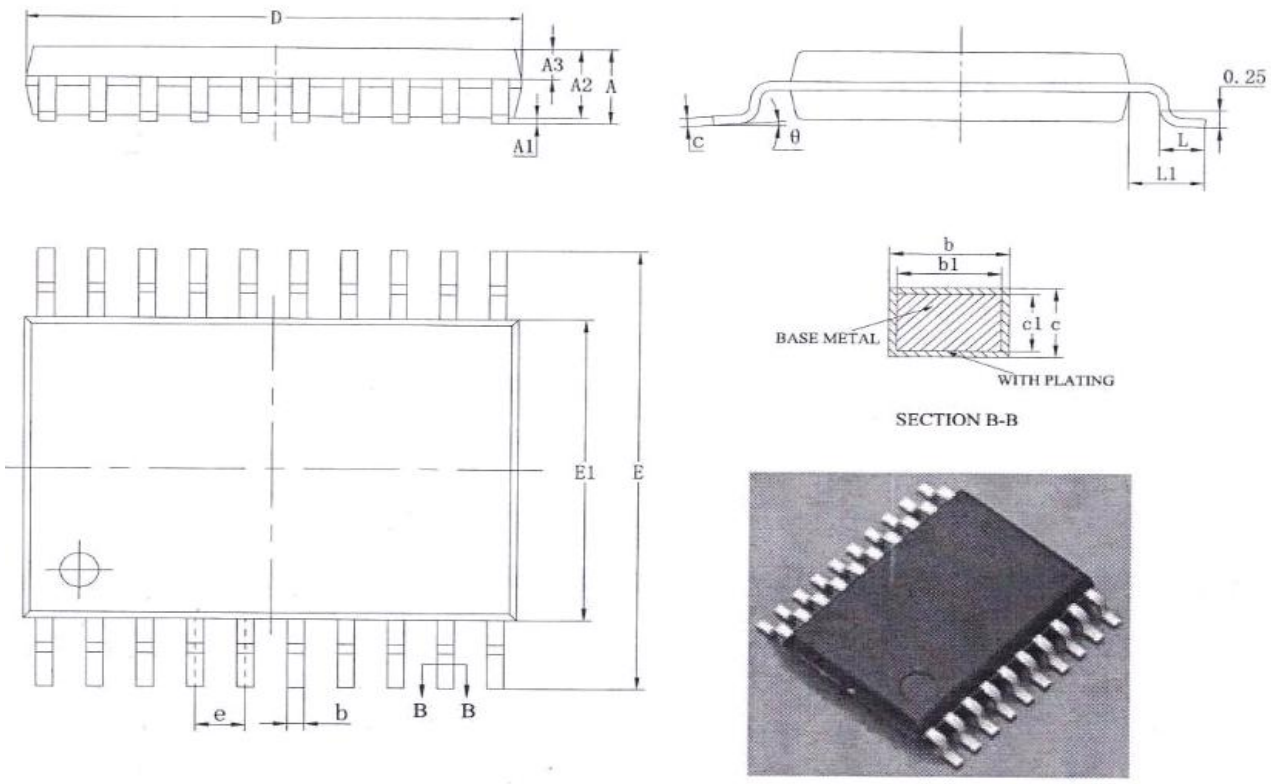
22. 芯片封装信息

SOP20 封装尺寸如下：



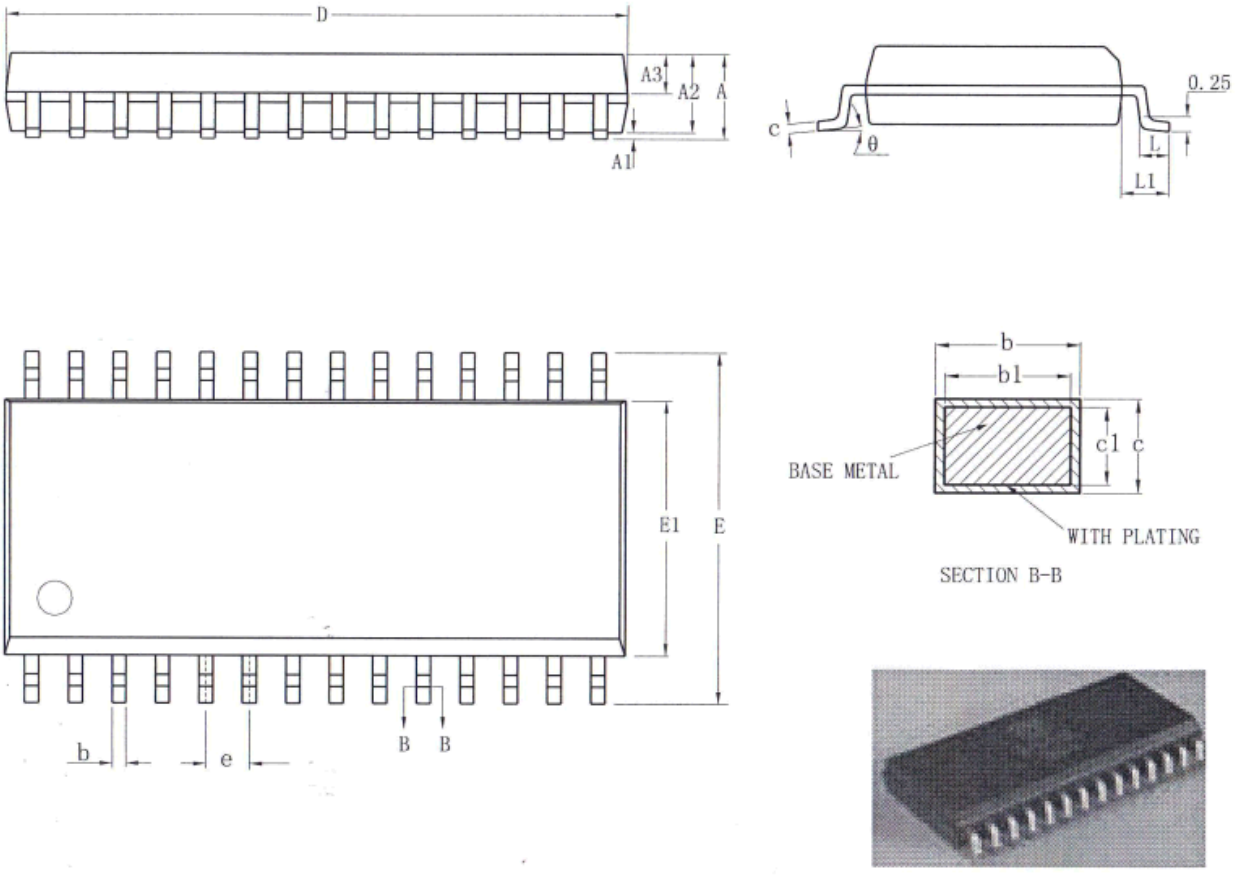
Symbol	Dimensions In Millimeters		Dimensions In Inches	
	Min	Max	Min	Max
A	2.350	2.650	0.093	0.104
A1	0.100	0.300	0.004	0.012
A2	2.100	2.500	0.083	0.098
b	0.330	0.510	0.013	0.020
c	0.204	0.330	0.008	0.013
D	12.520	13.000	0.493	0.512
E	7.400	7.600	0.291	0.299
E1	10.210	10.610	0.402	0.418
e	1.270（BSC）		0.050（BSC）	
L	0.400	1.270	0.016	0.050
θ	0°	8°	0°	8°

TSSOP20 封装尺寸如下：



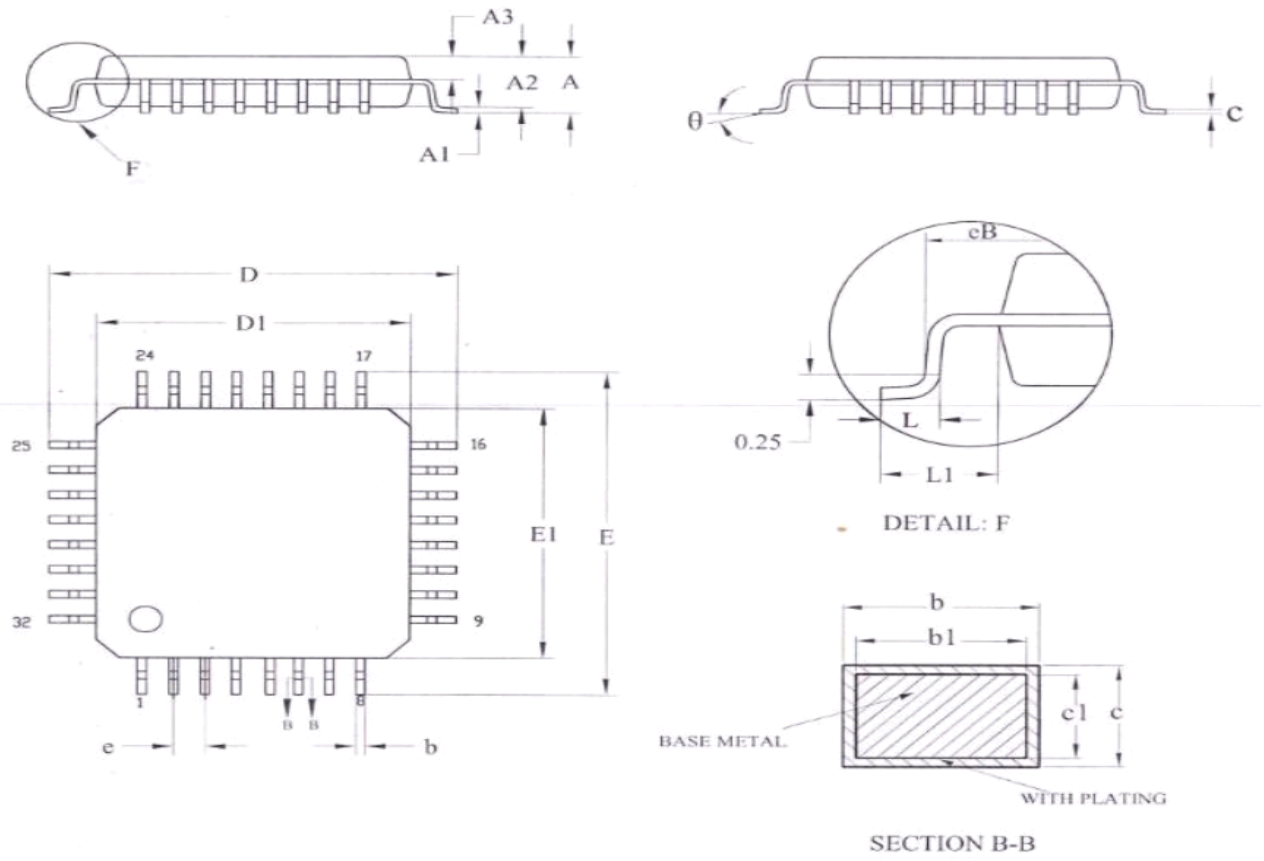
Symbol	Dimensions In Millimeters		Dimensions In Inches	
	Min	Max	Min	Max
A	-	1.20	-	0.047
A1	0.05	0.15	0.002	0.006
A2	0.80	1.05	0.031	0.041
A3	0.39	0.49	0.015	0.019
b	0.20	0.28	0.008	0.011
b1	0.19	0.25	0.007	0.010
c	0.13	0.17	0.005	0.007
c1	0.12	0.14	0.005	0.006
D	6.40	6.60	0.252	0.260
E1	4.30	4.50	0.169	0.177
E	6.20	6.60	0.244	0.259
e	0.65(BSC)		0.026(BSC)	
L	0.45	0.75	0.018	0.030
L1	1.00REF		0.039REF	
θ	0	8°	0	8°

SOP28 封装尺寸如下：



Symbol	Dimensions In Millimeters		Dimensions In Inches	
	Min	Max	Min	Max
A	-	2.65	-	0.104
A1	0.10	0.30	0.004	0.012
A2	2.25	2.35	0.089	0.093
A3	0.97	1.07	0.038	0.042
b	0.39	0.47	0.015	0.019
b1	0.38	0.44	0.015	0.017
c	0.25	0.29	0.010	0.011
c1	0.24	0.26	0.009	0.010
D	17.90	18.10	0.704	0.712
E	10.10	10.50	0.397	0.413
E1	7.40	7.60	0.290	0.299
e	1.27(BSC)		0.05(BSC)	
L	0.70	1.00	0.027	0.039
L1	1.40REF		0.055REF	
θ	0	8°	0	8°

LQFP32 封装尺寸如下：



Symbol	Dimensions In Millimeters		Dimensions In Inches	
	Min	Max	Min	Max
A	-	1.60	-	0.063
A1	0.05	0.15	0.002	0.006
A2	1.35	1.45	0.053	0.057
A3	0.59	0.69	0.023	0.027
b	0.33	0.41	0.013	0.016
b1	0.32	0.38	0.013	0.015
c	0.13	0.17	0.005	0.006
c1	0.12	0.14	0.005	0.006
D	8.80	9.20	0.346	0.362
D1	6.90	7.10	0.272	0.280
E	8.80	9.20	0.346	0.362
E1	6.90	7.10	0.272	0.280
eB	8.10	8.25	0.319	0.324
e	0.80(BSC)		0.031(BSC)	
L	0.45	0.75	0.018	0.030
L1	1.00REF		0.039REF	
θ	0	7°	0	7°

附录 1，文档更改历史

日期	版本	内容
2019-4-24	1.00	初版
2019-6-14	1.01	更正笔误“CLRF STATUS” 修改 11.3 章节的标题级别 添加 SOP20, SOP28 脚位图 修改 I2C 章节的笔误： ADDRF 改名为 ADDF TXE 改名 IICTXE，RXNE 改名 II2CRXNE 添加 BANK11 到 3.2.5 小节 Timerx 改名为 TIMx 更正若干笔误 添加 5.2.7 小节 添加校准步骤到 9.2.1 小节 更新图 3.2 的地址笔误 更正 TIM2CCR2L，TIM2CCR3H 寄存器描述笔误 SPICFG2 改为 SPICTRL2 开放 1T 模式 更正 9.2.1 小节中的 ADC 自校准步骤及该小节中的笔误 添加 ADC 自校准步骤到 9.2.7 小节，并修改示例代码 更新电气参数 更新特性页 更新若干格式问题 添加 FT62F085A-TRB 脚位图 添加注意事项到 8.2.2 小节，“写数据 EEPROM 存储器”
2019-9-3	1.02	TIM1BRK 更正为 TIM1BKR T1UTG 更正为 T1DTG MnMX1 更正为 MnMXS1 删除 FT62F085-TRB 脚位 更新电气特性的 I/O 源电流 URLCR.2 改名为 URSTOP
2019-11-18	1.03	更改电气特性 6.3 小节睡眠下的中断最后添加注意 PSINKx 寄存器描述中，把 L0（最小）L1（最大）改为：L0, xxmA, L1, xxmA 时钟框图中的“=12pF”删掉 TIM4CR1 的 T4CKS 改为可读写 KEYx 改为从 KEY1 开始编号

		更新 14.1.2 主机接收小节 添加 20.2，指令详细描述