

带 LED 驱动、AD、触控键增强型 Flash 单片机

CBM73xxF 系列芯片手册_V1.07

版本信息

版本	日期	说明
V1.00	20190603	初版
⋮		
V1.04	20200714	添加型号 CBM63082FS1A 的封装图
V1.05	20200717	1 修改 usr 寄存器位的清 0 操作说明；2 增加“附录Ⅶ、io 端口电流档位驱动特性”
V1.06	20210201	删除 CBM7332FQ1D 封装
V1.07	20210218	更正 CBM7320FS1D 的引脚标注（P43/K27）

最小起订量

芯片型号	封装	最小包装量	订单要求	备注
CBM7332FLQ1D	LQPF44	160pcs/盘	1600pcs 起订，按照 160 的倍数下订单。	
CBM7326FLQ1D	LQFP32	250pcs/盘	2000pcs 起订，按照 250 的倍数下订单。	
CBM7320FP1A	SSOP28	46pcs/管	506pcs 起订，按照 46 的倍数下订单。	
CBM7320FS1D	SOP28	27pcs/管	486pcs 起订，按照 27 的倍数下订单。	
CBM7312FS1D	SOP20	38pcs/管	494pcs 起订，按照 38 的倍数下订单。	
CBM7308FS1D	SOP16	50pcs/管	500pcs 起订，按照 50 的倍数下订单。	
CBM63082FS1D	SOP16	50pcs/管	500pcs 起订，按照 50 的倍数下订单。	

型号命名规则：

通用名：CBMx1x2x3x4x5x6x7x8； **例如：CBM 7 3 2 0 F S 1 D**

CBM：芯邦前缀。

x1：=6/7:51 内核。

x2：=3:三合一触控芯片；=6:MCU 芯片。

x3,x4：最大触摸按键数(或引脚数：32-44pin、26-32pin、20-28pin、18-24pin、12-20pin、08-16pin)。

x5：芯片版本。

x6：封装类型。=LQ:LQFP 封装，=Q:QFP 封装，=P:tssop 封装，=S:sop 封装。

x7,x8：内部序号。

目录

版本信息 2
最小起订量 3
目录 4
MCU 特性 7
MCU 选型表 8
引脚图一 CBM7332FLQ1D(LQFP44)： 9
引脚图二 CBM7326FLQ1D(LQFP32)： 10
引脚图三 CBM7320FS1D(SOP28)/CBM7320FP1A(SSOP28)： 11
引脚图四 CBM7312FS1D(SOP20)： 11
引脚图五 CBM7308FS1D(SOP16)： 13
引脚图六 CBM63082FS1D(SOP16)： 14
表 1：引脚汇总定义（参考 PxyIOCFG 寄存器的重映射功能描述） 15
1、系统概述 16
CBM73xxF 系统框图 16
2、存储器 17
2.1、程序存储器 17
2.2、数据存储器 17
3、特殊功能寄存器(SFR) 18
3.1、外设特殊功能寄存器 18
3.2、内核特殊功能寄存器 21
4、振荡器模块 26
4.1、系统总时钟框图 27
4.2、系统相关寄存器的定义 28
4.3、时钟配置例程 34
5、MCU 的特性 37
5.1、复位&LVD 37
5.2、中断 41
5.3、低功耗模式 51
5.4、ESD 静电防护 55
5.5、RAM 区偶校验 55
5.6、CRC16-CCITT 校验 56
5.7、编程加密 56
5.8、在线 Debug &烧录 57
5.9、在线更新升级用户程序 57
6、外部中断 60
6.1、外部中断寄存器的定义 60
6.2、外部中断使用说明 62
6.3、外部中断配置步骤 63
7、I/O 端口 64
7.1、IO 寄存器的定义 64
7.2、配置引脚为 GPIO 的操作步骤 70

8、看门狗定时器(WDT) 71
8.1、WDT 寄存器的定义 71
8.2、WDT 配置步骤 72
9、8 位定时器 Timer0 74
9.1、Timer0 支持的功能 74
9.2、Timer0 寄存器的定义 79
9.3、Timer0 中断 81
10、8 位定时器 Timer1 82
10.1、Timer1 支持的功能 82
10.2、Timer1 寄存器的定义 88
10.3、Timer1 中断 90
11、16 位定时器 Timer2 91
11.1、Timer2 支持的功能 91
11.2、Timer2 寄存器的定义 94
11.3、Timer2 中断 97
12、16 位定时器 Timer3 98
12.1、Timer3 的支持的功能 98
12.2、Timer3 寄存器的定义 113
12.3、Timer3 中断 117
13、PWM 模块 118
13.1、PWM 的操作步骤 118
13.2、PWM 寄存器的定义 119
14、实时时钟模块(RTC) 122
14.1、RTC 工作原理 122
14.2、RTC 时钟选择 122
14.3、RTC 的操作步骤 122
14.4、RTC 寄存器的定义 123
14.5、RTC 中断 126
15、ADC 模块 127
15.1、ADC 工作原理 127
15.2、ADC 的操作步骤 128
15.3、ADC 寄存器的定义 130
16、触摸按键功能(sensorADC) 134
sensorADC 工作原理 134
SensorADC 操作步骤 134
sensorADC 寄存器的定义 137
17、UART 串口模块 142
17.1、UART 的操作步骤 142
17.2、UART1 普通模式发送数据 143
17.3、UART1 普通模式接收数据 143
17.4、UART1 DMA 模式发送数据 144
17.5、UART1 DMA 模式接收数据 144
17.6、UART1 寄存器的定义 144

17.7、UART2 寄存器的定义 149
18、SPI 模块 150
18.1、SPI 的操作步骤 150
18.2、主控模式 152
18.3、从动模式 153
18.4、SPI 寄存器的定义 153
19、I2C 模块 157
19.1、I2C Slave(从动模块) 157
19.2、I2C Master(主控模块) 164
20、EFC 模块(类 EEPROM) 172
20.1、NVR 区的操作(类 EEPROM) 172
20.2、MAIN 区 CRC16-CCITT 校验 173
20.3、EFC 寄存器的定义 176
21、LED 驱动器 179
21.1、LED 的操作步骤 179
21.2、数码管的接法 180
21.3、LED 显示控制原理 181
21.4、LED 寄存器的定义 183
22、电气特性 186
极限参数 186
直流电气特性（T = 25℃） 186
交流电气特性（T = 25℃） 187
上电复位特性（T = 25℃） 188
触控按键电气特性（T = 25℃） 188
A/D 转换器电气特性（T = 25℃） 189
flash 电气特性 189
类 EEPROM 电气特性 189
23、封装信息 190
附录Ⅰ 汇编指令周期表 196
附录Ⅱ 内部高速时钟 48M 温漂曲线 199
附录Ⅲ 内部低速时钟 800K 温漂曲线 200
附录Ⅳ ADC 采样温漂曲线（包含 Vref 全温度偏差） 201
1、采样数据 201
2、常规方式温漂曲线（图表对应上表中的数据） 202
3、硬件方式温漂曲线（图表对应上表中的数据） 203
附录Ⅴ XBYTE 类型地址寄存器的操作说明 204
附录Ⅵ IO 端口电流档位（6mA、30mA、60mA）驱动特性 205

MCU 特性

- 基于 8051 指令的 8 位单片机
- RAM：
  -支持内部 RAM256 字节、外部 XRAM 1536 字节
  -支持使能 RAM、XRAM 数据自动偶校验
- Flash ROM：24K~64K 字节
  -擦写周期 100,000 次以上
  -数据保存时间长达 20 年（25℃）
- 类 EEPROM（EFC）：1024 字节
  -擦写周期 100,000 次以上
  -数据保存时间长达 20 年（25℃）
- 工作电压：VDD = 2.7V~5.5V
- 时钟源（-40℃~85℃）：
  -内部高频振荡器：48Mhz(精度±1%)
  -内部低频振荡器：800Khz(精度±1%)
  -外部高速 M 级晶振：4~16Mhz
  -外部低速 K 级晶振：32.768Khz
- 自动监测外部 M 级晶振工作状态的功能
- MCU 主时钟(MCLK)：可配置，最高 16Mhz
- 软件看门狗(WDT)：复位时间 320us~42s
- 内置低电压检测功能(LVD)、ESD 检测功能
- 支持省电模式
  -SLEEP 休眠模式
  -IDLE 空闲模式
- 支持代码编程加密保护，防止代码泄露
- 支持 CRC16-CCITT 校验功能（上电校验、动态校验）
- 支持硬件复位功能(低电平复位)
- 支持软件指令复位功能
- 支持在线更新升级程序功能
- 支持通过串口在线调试

低功耗特性：

- 休眠电流（sleep）：
  -5.0V 时典型值为 160uA(ADCCR_SLEEP =1<<4、ADCBUFEN=0、IO 高阻内部上拉）
- 工作电流：
  -16Mhz(MCLK)、 5.0V 时典型值为 5.0mA（ADCCR_SLEEP =1<<4、ADCBUFEN=0、空载）

外设特性：

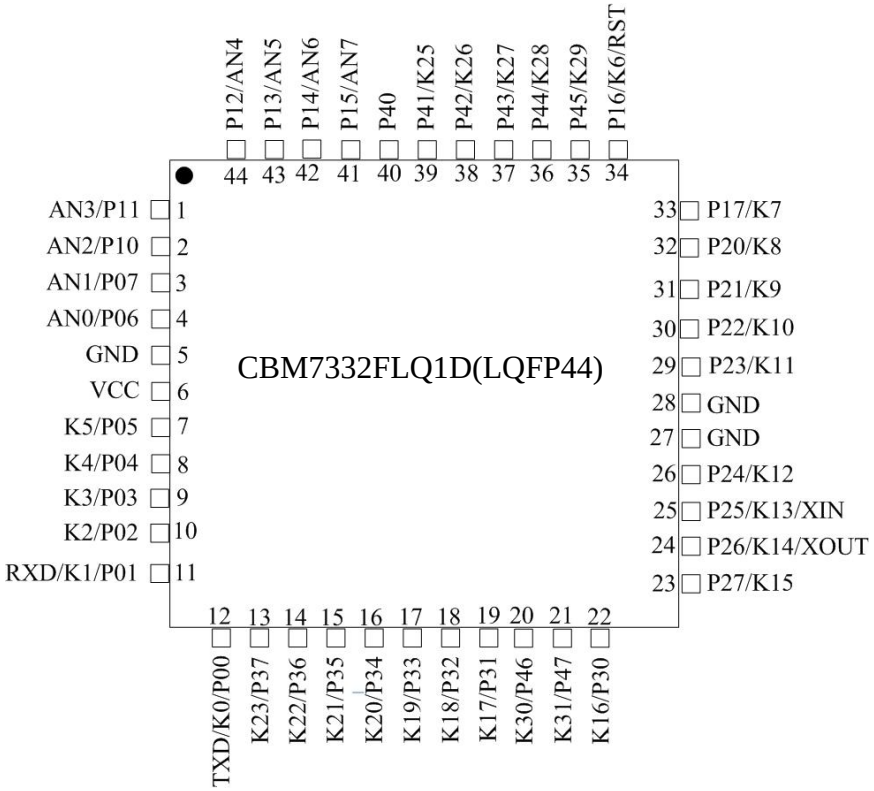
- 多达 40 个单独可控的 I/O 引脚：
  -大的灌/拉电流可直接驱动 LED 等器件，最大 60mA
  -可单独编程的电流大小控制寄存器(6mA、30mA、60mA)
  -可单独编程的方向控制寄存器
  -可单独编程的上拉电阻选择控制寄存器、上拉使能寄存器
  -可单独编程的开漏控制寄存器
  -支持 IO 口复用功能重映射
  -支持位操作控制的端口数据寄存器…
- A/D 转换器：
  -10 位 A/D 转换器
  -最大采样时钟为 1.4Mhz
  -最大支持 8 个通道
  -支持最大 64 次硬件滤波采样
  -支持选择内部 1v(精度±1%)为基准的绝对电压采样方式
- 触摸功能：
  -内建 12/16/24/31 路触控按键功能
  -支持 MCU 休眠状态下唤醒
  -支持低功耗扫描工作方式
- LED 数显驱动器：
  -硬件驱动方式，无需核参与、不占用用户软件指令资源
  -COM 端口支持最大 60mA 驱动电流 ，电流大小可配
  -4×13 段/5×12 段/6×11 段/7×10 段/9×8 段/10×7 段驱动共阴/共阳 LED
- 支持两路相互独立的通用 UART：
  -独立的波特率发生器
  -自动波特率校准
  -每路 UART 都支持全双工普通模式与半双工 DMA 模式
  -DMA 工作模式，自动从用户指定的 XRAM 缓存区收发数据
  -DMA 工作模式,不占用户软件资源,最多一次传输 128bytes
  -支持数据收发完成中断
- 定时器 timer：4 个独立定时器模块用于间隔定时、方波输出、PWM 输出、外部事件计数、捕捉输入及单脉冲输出等
- 支持多路 PWM 输出：
  -2 个专用 PWM 模块可提供 2 路 PWM 输出
  -4 个定时器 timer 模块也可配置提供 4 路 PWM 输出
- RTC：
  -支持实时时钟、闹钟定时功能
  -支持秒、分、时、天、定时 5 个中断
  -支持内部 800K 分频得到的 100Khz（±1%精度）时钟源
  -支持外部 32.768Khz 时钟源
  -支持外部 M 级晶振分频得到的~32.768Khz 时钟源
- 4 线 SPI(4 种数据传输极性模式)
- I^2C 主/从模式接口（主/从模块相互独立，可同时工作）

MCU 选型表

功能模块 / 芯片型号	程序存储器	数据存储器			I/O	10 位 A/D (通道数)	触摸按键	外部中断	复位 IO
	Flash (字节)	IRAM (字节)	XRAM (字节)	类 EEPROM (字节)					
CBM7332FLQ1D	64K	256	1536	1024	40	8	31	6	Y
CBM7326FLQ1D	64K	256	1536	1024	28	8	20	4	Y
CBM7320FS1D	64K	256	1536	1024	26	6	20	2	Y
CBM7320FP1D	64K	256	1536	1024	26	6	20	2	Y
CBM7312FS1D	64K	256	1536	1024	18	4	14	1	Y
CBM7308FS1D	24K	256	1536	1024	14	2	12	1	Y
CBM63082FS1D	24K	256	1536	1024	14	3	11	1	Y

功能模块 / 芯片型号	UART	SPI	I2C		定时器	PWM	RTC	LED 驱动	外挂 K 级晶振	外挂 M 级晶振	封装
			主	从							
CBM7332FLQ1D	2 路	Y	Y	Y	4	2	Y	4×13	Y	Y	LQFP44
CBM7326FLQ1D	2 路	Y	Y	Y	4	2	Y	4×13	Y	Y	LQFP32
CBM7320FS1D	2 路	Y	Y	Y	4	2	Y	4×13	Y	N	SOP28
CBM7320FP1D	2 路	Y	Y	Y	4	2	Y	4×13	Y	N	SSOP28
CBM7312FS1D	2 路	Y	Y	Y	4	2	Y	Y	Y	N	SOP20
CBM7308FS1D	2 路	Y	Y	Y	4	2	Y	Y	Y	N	SOP16
CBM63082FS1D	2 路	Y	Y	Y	4	2	Y	Y	Y	N	SOP16

引脚图一 CBM7332FLQ1D(LQFP44)：



注：

1、封装图中已标识出的复用功能只能映射到该 IO 口上，不能重映射到其它 IO，具体如下：

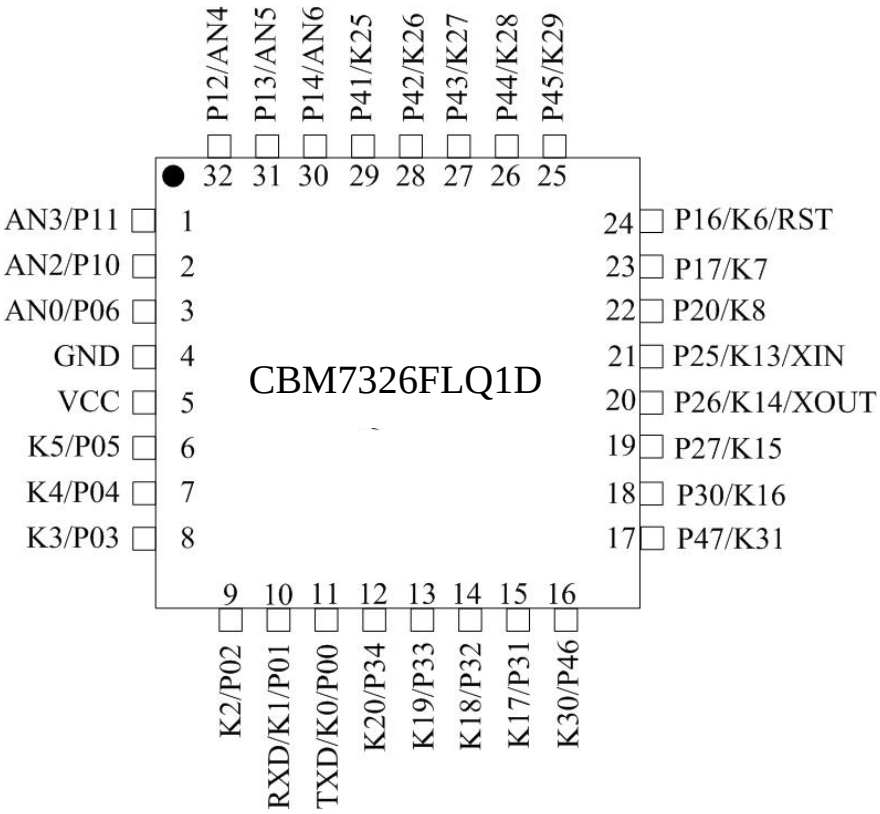
- Debug 调试/烧录串口：TXD 只能映射到 P00， RXD 只能映射到 P01；
- AD：AN0~AN7 只能对应映射到 P06~P15 口，即 AN0-P06、……、AN7-P15；
- RST：硬件复位功能只能映射在 P16 口；
- 外挂 M 或 K 级晶振二选一：输入引脚 XIN 只能映射到 P25，输出引脚 XOUT 只能映射到 P26；
- 触摸：除 AD(P06~P15)外的其它 IO 口均可映射成触摸按键，但只能一一对应映射（如 P00-K0、P01-K1、P02-K2 等）。

2、以下复用功能可以映射到除 VCC、GND 外的任意 IO 口，具体如下：

- SPI Slave: SSDI/SSDO/SSCK/SSCS；
- SPI Master: SMDI/SMDO/SMCK/SMCS；
- I2C Slave: SSCL/SSDA；
- I2C Master: SMCL/SMDA；
- uart1: TX1/RX1；
- uart2: TX2/RX2；
- timer: T0O、T1I/T1O、T2O、T3IO/T3OO/T3I1；
- PWM: PWM1/PWM2；
- 外部中断：INT0/ INT1/ INT2/ INT3/ INT4/ INT5；
- SEG：数码管段选端口；
- COM/GRID：数码管 COM 端口。

3、P00\P01 为 dubug 口默认 2.2k 上拉，P16 默认 4.5k 上拉，这 3 个 io 在上电时会有~40ms 高电平输出，因此不建议用作电平驱动类的控制引脚（比如不建议用来驱动控制 led、继电器、可控硅、蜂鸣器等等）。

引脚图二 CBM7326FLQ1D(LQFP32)：



注：

1、封装图中已标识出的复用功能只能映射到该 IO 口上，不能重映射到其它 IO，具体如下：

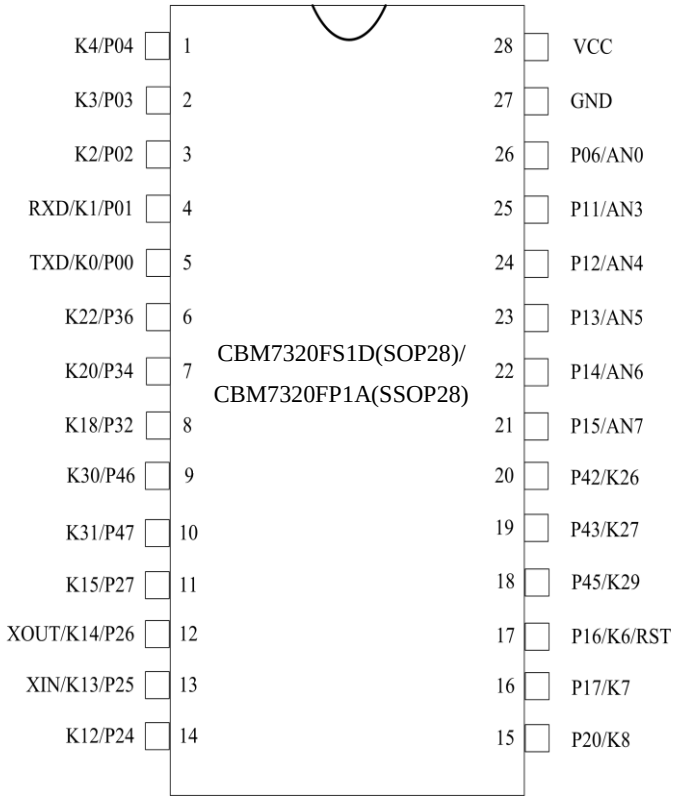
- Debug 调试/烧录串口：TXD 只能映射到 P00， RXD 只能映射到 P01；
- AD：AN0~AN7 只能对应映射到 P06~P15 口，即 AN0-P06、……、AN7-P15；
- RST：硬件复位功能只能映射在 P16 口；
- 外挂 M 或 K 级晶振二选一：输入引脚 XIN 只能映射到 P25，输出引脚 XOUT 只能映射到 P26；
- 触摸：除 AD(P06~P15)外的其它 IO 口均可映射成触摸按键，但只能一一对应映射（如 P00-K0、P01-K1、P02-K2 等）。

2、以下复用功能可以映射到除 VCC、GND 外的任意 IO 口，具体如下：

- SPI Slave: SSDI/SSDO/SSCK/SSCS；
- SPI Master: SMDI/SMDO/SMCK/SMCS；
- I2C Slave: SSCL/SSDA；
- I2C Master: SMCL/SMDA；
- uart1: TX1/RX1；
- uart2: TX2/RX2；
- timer: T0O、T1I/T1O、T2O、T3IO/T3OO/T3I1；
- PWM: PWM1/PWM2；
- 外部中断：INT0/ INT1/ INT2/ INT3/ INT4/ INT5；
- SEG：数码管段选端口；
- COM/GRID：数码管 COM 端口。

3、P00\P01 为 dubug 口默认 2.2k 上拉，P16 默认 4.5k 上拉，这 3 个 io 在上电时会有~40ms 高电平输出，因此不建议用作电平驱动类的控制引脚（比如不建议用来驱动控制 led、继电器、可控硅、蜂鸣器等等）。

引脚图三 CBM7320FS1D(SOP28)/CBM7320FP1A(SSOP28)：



注：

1、封装图中已标识出的复用功能只能映射到该 IO 口上，不能重映射到其它 IO，具体如下：

- Debug 调试/烧录串口：TXD 只能映射到 P00， RXD 只能映射到 P01；
- AD：AN0~AN7 只能对应映射到 P06~P15 口，即 AN0-P06、……、AN7-P15；
- RST：硬件复位功能只能映射在 P16 口；
- 外挂 M 或 K 级晶振二选一：输入引脚 XIN 只能映射到 P25，输出引脚 XOUT 只能映射到 P26；
- 触摸：除 AD(P06~P15)外的其它 IO 口均可映射成触摸按键，但只能一一对应映射（如 P00-K0、P01-K1、P02-K2 等）。

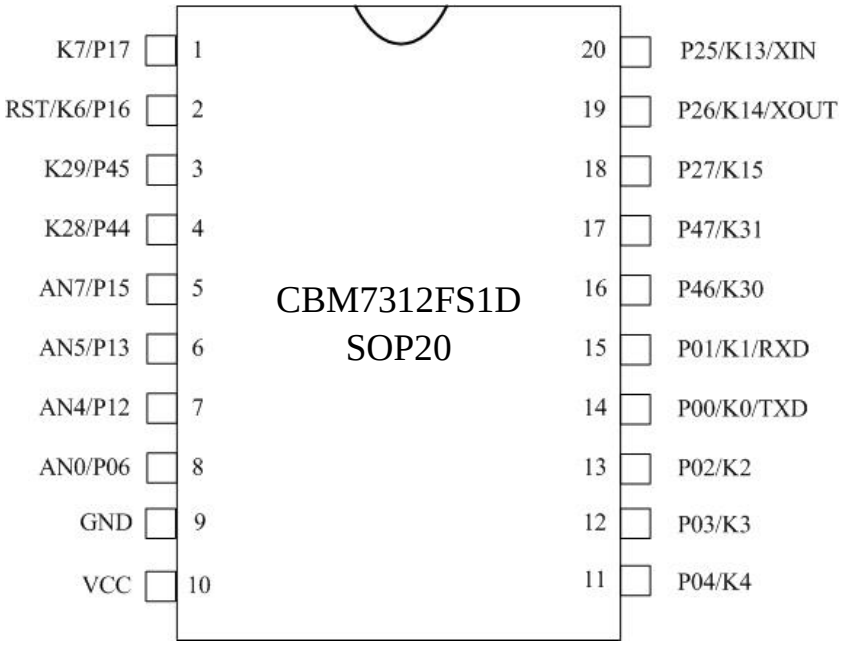
2、以下复用功能可以映射到除 VCC、GND 外的任意 IO 口，具体如下：

- SPI Slave: SSDI/SSDO/SSCK/SSCS；
- SPI Master: SMDI/SMDO/SMCK/SMCS；
- I2C Slave: SSCL/SSDA；
- I2C Master: SMCL/SMDA；
- uart1: TX1/RX1；
- uart2: TX2/RX2；
- timer: T0O、T1I/T1O、T2O、T3IO/T3OO/T3I1；
- PWM: PWM1/PWM2；
- 外部中断：INT0/ INT1/ INT2/ INT3/ INT4/ INT5；
- SEG：数码管段选端口；
- COM/GRID：数码管 COM 端口。

3、P00\P01 为 dubug 口默认 2.2k 上拉，P16 默认 4.5k 上拉，这 3 个 io 在上电时会有~40ms 高电平输出，因此不建议用作电

平驱动类的控制引脚（比如不建议用来驱动控制 led、继电器、可控硅、蜂鸣器等等）。

引脚图四 CBM7312FS1D(SOP20)：



注：

1、封装图中已标识出的复用功能只能映射到该 IO 口上，不能重映射到其它 IO，具体如下：

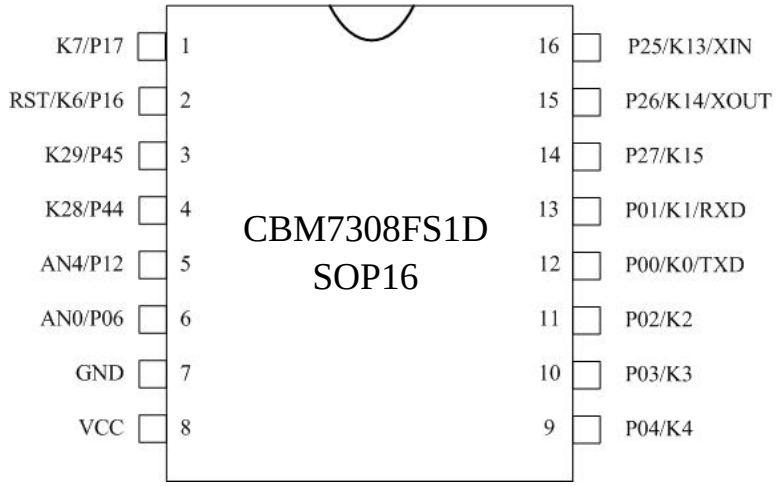
- Debug 调试/烧录串口：TXD 只能映射到 P00， RXD 只能映射到 P01；
- AD：AN0~AN7 只能对应映射到 P06~P15 口，即 AN0-P06、……、AN7-P15；
- RST：硬件复位功能只能映射在 P16 口；
- 外挂 M 或 K 级晶振二选一：输入引脚 XIN 只能映射到 P25，输出引脚 XOUT 只能映射到 P26；
- 触摸：除 AD(P06~P15)外的其它 IO 口均可映射成触摸按键，但只能一一对应映射（如 P00-K0、P01-K1、P02-K2 等）。

2、以下复用功能可以映射到除 VCC、GND 外的任意 IO 口，具体如下：

- SPI Slave: SSDI/SSDO/SSCK/SSCS；
- SPI Master: SMDI/SMDO/SMCK/SMCS；
- I2C Slave: SSCL/SSDA；
- I2C Master: SMCL/SMDA；
- uart1: TX1/RX1；
- uart2: TX2/RX2；
- timer: T0O、T1I/T1O、T2O、T3IO/T3OO/T3I1；
- PWM: PWM1/PWM2；
- 外部中断：INT0/ INT1/ INT2/ INT3/ INT4/ INT5；
- SEG：数码管段选端口；
- COM/GRID：数码管 COM 端口。

3、P00\P01 为 dubug 口默认 2.2k 上拉，P16 默认 4.5k 上拉，这 3 个 io 在上电时会有~40ms 高电平输出，因此不建议用作电平驱动类的控制引脚（比如不建议用来驱动控制 led、继电器、可控硅、蜂鸣器等等）。

引脚图五 CBM7308FS1D(SOP16)：



注：

1、封装图中已标识出的复用功能只能映射到该 IO 口上，不能重映射到其它 IO，具体如下：

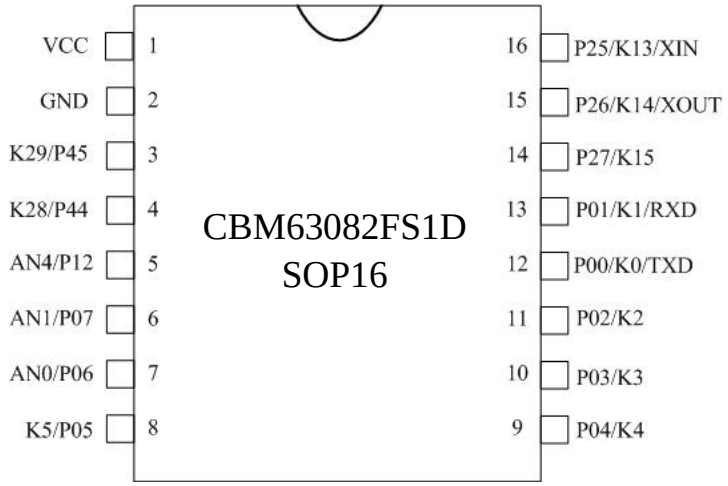
- Debug 调试/烧录串口：TXD 只能映射到 P00， RXD 只能映射到 P01；
- AD：AN0~AN7 只能对应映射到 P06~P15 口，即 AN0-P06、……、AN7-P15；
- RST：硬件复位功能只能映射在 P16 口；
- 外挂 M 或 K 级晶振二选一：输入引脚 XIN 只能映射到 P25，输出引脚 XOUT 只能映射到 P26；
- 触摸：除 AD(P06~P15)外的其它 IO 口均可映射成触摸按键，但只能一一对应映射（如 P00-K0、P01-K1、P02-K2 等）。

2、以下复用功能可以映射到除 VCC、GND 外的任意 IO 口，具体如下：

- SPI Slave: SSDI/SSDO/SSCK/SSCS；
- SPI Master: SMDI/SMDO/SMCK/SMCS；
- I2C Slave: SSCL/SSDA；
- I2C Master: SMCL/SMDA；
- uart1: TX1/RX1；
- uart2: TX2/RX2；
- timer: T0O、T1I/T1O、T2O、T3IO/T3OO/T3I1；
- PWM: PWM1/PWM2；
- 外部中断：INT0/ INT1/ INT2/ INT3/ INT4/ INT5；
- SEG：数码管段选端口；
- COM/GRID：数码管 COM 端口。

3、P00\P01 为 dubug 口默认 2.2k 上拉，P16 默认 4.5k 上拉，这 3 个 io 在上电时会有~40ms 高电平输出，因此不建议用作电平驱动类的控制引脚（比如不建议用来驱动控制 led、继电器、可控硅、蜂鸣器等等）。

引脚图六 CBM63082FS1D(SOP16)：



注：

1、封装图中已标识出的复用功能只能映射到该 IO 口上，不能重映射到其它 IO，具体如下：

- Debug 调试/烧录串口：TXD 只能映射到 P00， RXD 只能映射到 P01；
- AD：AN0~AN7 只能对应映射到 P06~P15 口，即 AN0-P06、……、AN7-P15；
- RST：硬件复位功能只能映射在 P16 口；
- 外挂 M 或 K 级晶振二选一：输入引脚 XIN 只能映射到 P25，输出引脚 XOUT 只能映射到 P26；
- 触摸：除 AD(P06~P15)外的其它 IO 口均可映射成触摸按键，但只能一一对应映射（如 P00-K0、P01-K1、P02-K2 等）。

2、以下复用功能可以映射到除 VCC、GND 外的任意 IO 口，具体如下：

- SPI Slave: SSDI/SSDO/SSCK/SSCS；
- SPI Master: SMDI/SMDO/SMCK/SMCS；
- I2C Slave: SSCL/SSDA；
- I2C Master: SMCL/SMDA；
- uart1: TX1/RX1；
- uart2: TX2/RX2；
- timer: T0O、T1I/T1O、T2O、T3IO/T3OO/T3I1；
- PWM: PWM1/PWM2；
- 外部中断：INT0/ INT1/ INT2/ INT3/ INT4/ INT5；
- SEG：数码管段选端口；
- COM/GRID：数码管 COM 端口。

3、P00\P01 为 dubug 口默认 2.2k 上拉，P16 默认 4.5k 上拉，这 3 个 io 在上电时会有~40ms 高电平输出，因此不建议用作电平驱动类的控制引脚（比如不建议用来驱动控制 led、继电器、可控硅、蜂鸣器等等）。

表 1：引脚汇总定义（参考 PxyIOCFG 寄存器的重映射功能描述）

引脚名称	说明	引脚名称	说明
VCC	电源正（2.7~5.5v）	TX1	串口 1 输出口
GND	电源地	RX1	串口 1 输入口
RST	硬件复位端口（只能映射到 P16）	TX2	串口 2 输出口
TXD	Debug 调试\烧录串口发送（只能映射到 P00）	RX2	串口 2 输入口
RXD	Debug 调试\烧录串口接收（只能映射到 P01）	T0O	定时器 0 输出口
P00~P47	GPIO 功能	T1IO	定时器 1 输入/输出口
K0~K31	触摸按键输入口	T2O	定时器 2 输出口
AN0~AN7	ADC 输入引脚（P06~P15）	T3IO	定时器 3 输入 0/输出口
XIN	外部晶振输入（只能映射到 P25）；（外部 M 或 K 级晶振二选一）	T3I1	定时器 3 输入 1
XOUT	外部晶振输出（只能映射到 P26）；（外部 M 或 K 级晶振二选一）	PWM1	PWM1 输出口
SSDI	SPI Slave 从机数据输入口	PWM2	PWM2 输出口
SSDO	SPI Slave 从机数据输出口	INT0~INT5	外部中断 0~5 输入口
SSCK	SPI Slave 从机时钟输入口	LEDSEG0~LEDSEG8	LED SEGx 口
SSCS	SPI Slave 从机片选信号	LEDSEG9_LEDGRID7	LED SEG9/GRID7 口
SMDI	SPI Master 主机数据输入口	LEDSEG10_LEDGRID6	LED SEG10/GRID6 口
SMDO	SPI Master 主机数据输出口	LEDSEG11_LEDGRID5	LED SEG11/GRID5 口
SMCK	SPI Master 主机时钟输出口	LEDSEG12_LEDGRID4	LED SEG12/GRID4 口
SMCS	SPI Master 主机片选信号	LEDGRID0~LEDGRID3	LED GRIDx 口(即 COM 口)
SSCL	I^2C Slave 从机时钟口		
SSDA	I^2C Slave 从机数据口		
SMCL	I^2C Master 主机时钟口		
SMDA	I^2C Master 主机数据口		

1、 系统概述

本用户手册涵盖了 CBM73xxF 系列所有型号芯片。上表 1 中给出了相应的引脚功能说明。下图 1-1 给出了 CBM73xxF 芯片的系统框图。

CBM73xxF 系列 MCU 支持 IO 大电流驱动、内置 LED 硬件驱动模块、触摸按键扫描、10 位高精度的 ADC（内置高精度 1v 参考电压、1v 误差 <= ±1%）、UART DMA 工作模式（直接从指定 XRAM 区收发数据，通迅只需占用极少的软件指令资源）、内部类 EEPROM（EFC 模块）、IO 重映射功能……；此外，还具有 LVD 检测、ESD 检测、CRC 校验、代码编程加密保护等提高 MCU 安全特性的功能。

CBM73xxF 系统框图

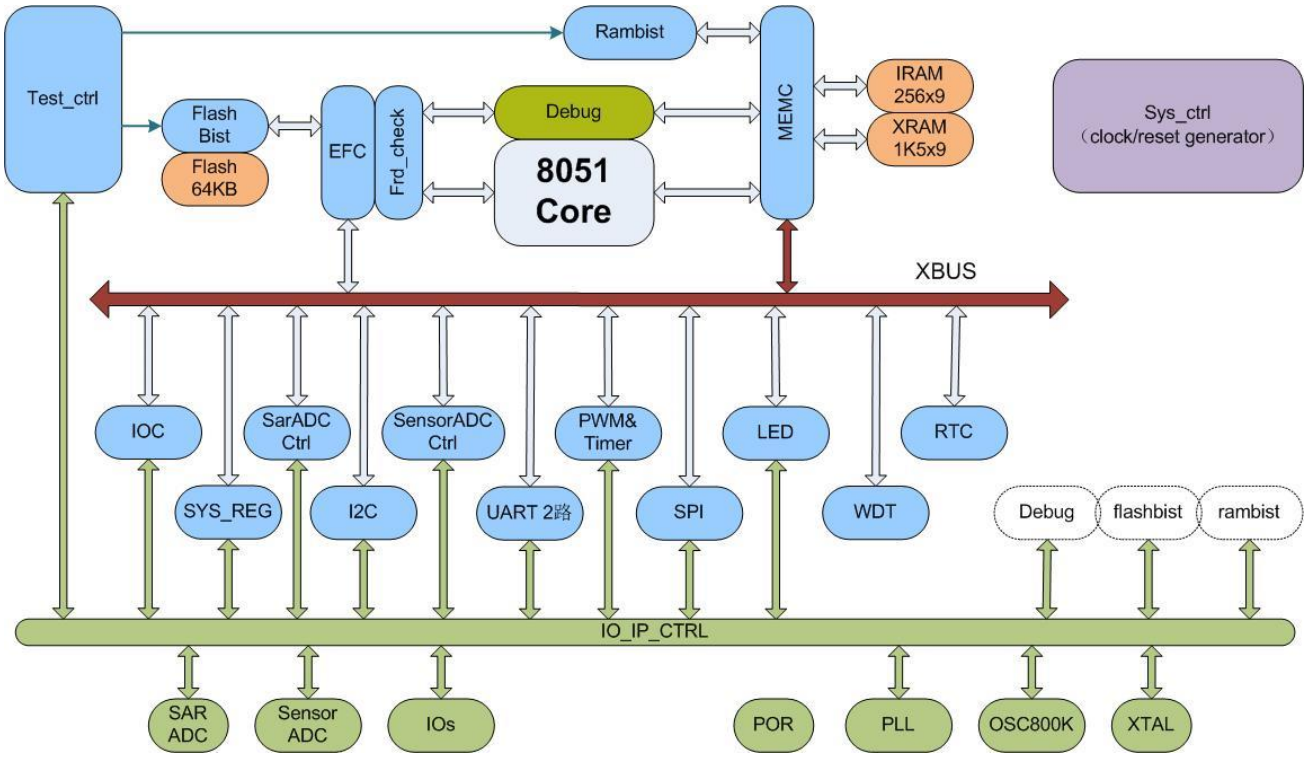


图 1-1、系统框图

2、 存储器

2.1、程序存储器

CBM73xxF 系列芯片具备 16 位程序计数器(PC 指针)，支持寻址 64KBytes(0000H - FFFFH)的程序存储空间。复位向量地址为 0000H，6 个中断向量默认地址分别为 0003H、000BH、0013H、001BH、0023H、002BH(可配置 VTOR 寄存器使中断向量产生偏移，详情请参考“3.2、中断向量偏移地址寄存器”)。64K 程序存储器映射如下图 2-1 所示。

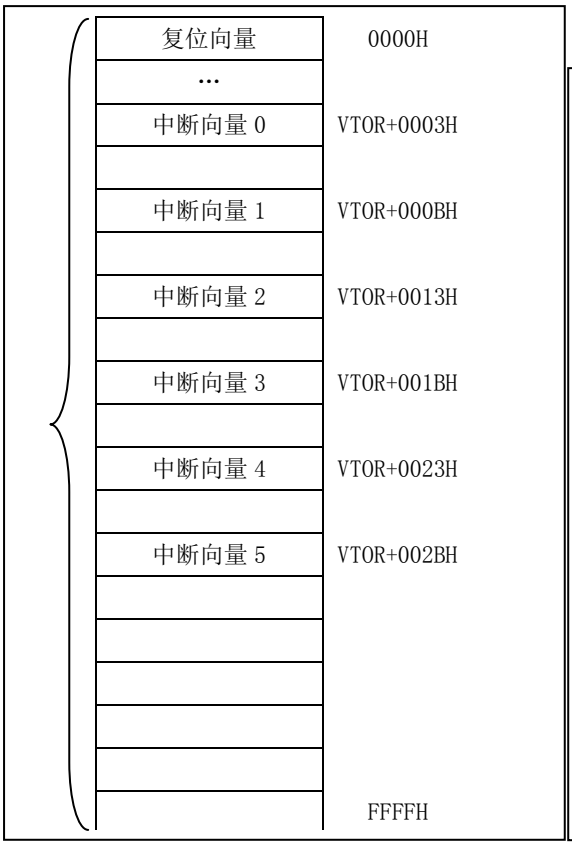


图 2-1：CBM73xxF 64KBytes 程序存储器映射图



图 2-2：CBM73xxF 片内 256 Bytes RAM 映射

2.2、数据存储器

数据存储器分为 8051 内核内置片内 256 字节 RAM（低 128 字节直接寻址区、高 128 字节间接寻址区）、片内 128 字节 RAM（高 128 地址的直接寻址特殊功能寄存器区）、扩展的 1536 字节外部 XRAM 区及片外特殊功能寄存器。且 CBM73xxF 系列 MCU 实现了对 RAM 和 XRAM 的偶校验功能，可有效避免一些干扰可能导致的读 RAM、XRAM 数据出错而触发错误响应动作的情况。

片内 256 字节 RAM（00H-FFH）

CBM73xxF 芯片的片内 256 字节 RAM 映射如上图 2-2 所示，片内 256 字节 RAM 具体分配如下：

a、工作寄存器区：共 4 组，每组 8 个存储单元，即 00H-07H、08H-0FH、10H-17H、18H-1FH，具体选择哪一个由程序状态字(PSW)中的 RS1 位和 RS0 位组合决定，默认为 00H-07H。

b、位寻址区：从 20H-2FH，共 16 个单元，每一位都可以进行位寻址(16*8=128 个位地址)。

c、便笺区：从 30H-FFH，共 208 个单元，作为数据变量区或堆栈区使用。

片内 128 字节 RAM(80H-FFH)

此区是内核特殊功能寄存器，如 P0~P4、B、ACC、PSW、IP1、IP2、IE、VTOR、PCON、PSEQ、DPH、DPL、DPTR、SP 等。

片外 XRAM（00H-600H）

CBM73xxF 系列芯片还在芯片内部集成了 1536 字节的片外 XRAM，可供用户用于定义变量等操作。（强烈建议用户变量都定义在外部 XRAM 区，内部 RAM 则用于程序堆栈处理，即 keil 工程设置成“Target—Memory Model—Large：variables in XDATA”）

3、 特殊功能寄存器(SFR)

特殊功能寄存器是 CPU 和外设模块用来控制芯片实现所需操作的寄存器，用户可操作配置。分成两类：外设特殊功能寄存器(见表 3-1)和内核特殊功能寄存器(见表 3-2)。

3.1、外设特殊功能寄存器

表 3-1：CBM73xxF 外设特殊功能寄存器总表

寄存器名	XRAM 地址	寄存器名	XRAM 地址	寄存器名	XRAM 地址	寄存器名	XRAM 地址
IO 控制寄存器：		P40IOCFG	1020H	P0OD	3AFFH	RAMCR	58FFH
P00IOCFG	1000H	P41IOCFG	1021H	P1OD	3BFFH	SRSTCR	5C00H
P01IOCFG	1001H	P42IOCFG	1022H	P2OD	3CFFH	OSC800KTRIMR	1040H
P02IOCFG	1002H	P43IOCFG	1023H	P3OD	3DFFH	PLLTRIMR	1041H
P03IOCFG	1003H	P44IOCFG	1024H	P4OD	3EFFH	XTALCR0	1042H
P04IOCFG	1004H	P45IOCFG	1025H	P0LDRVRL	40FFH	XTALCR1	1043H
P05IOCFG	1005H	P46IOCFG	1026H	P0LDRVRH	41FFH	XTALCR2	1044H
P06IOCFG	1006H	P47IOCFG	1027H	P1LDRVRL	42FFH	LDOCR0	1045H
P07IOCFG	1007H	reserve	20FFH	P1LDRVRH	43FFH	LDOCR1	1046H
P10IOCFG	1008H	reserve	21FFH	P2LDRVRL	44FFH		
P11IOCFG	1009H	reserve	22FFH	P2LDRVRH	45FFH	**中断寄存器：**	
P12IOCFG	100AH	reserve	23FFH	P3LDRVRL	46FFH	IFLTCNTR	F0FFH
P13IOCFG	100BH	reserve	24FFH	P3LDRVRH	47FFH	XINT01TRGR	F1FFH
P14IOCFG	100CH	P0PU	25FFH	P4LDRVRL	48FFH	XINT23TRGR	F2FFH
P15IOCFG	100DH	P1PU	26FFH	P4LDRVRH	49FFH	XINT45TRGR	F3FFH
P16IOCFG	100EH	P2PU	27FFH	P0SMEN	0x4BFF	IRQ0SR	F4FFH
P17IOCFG	100FH	P3PU	28FFH	P1SMEN	0x4CFF	IRQ1SR	F5FFH
P20IOCFG	1010H	P4PU	29FFH	P2SMEN	0x4DFF	IRQ2SR	F6FFH
P21IOCFG	1011H	P0OE	2AFFH	P3SMEN	0x4EFF	IRQ3SR	F7FFH
P22IOCFG	1012H	P1OE	2BFFH	P4SMEN	0x4FFF	IRQ4SR	F8FFH
P23IOCFG	1013H	P2OE	2CFFH	**系统相关寄存器：**		IRQ5SR	F9FFH
P24IOCFG	1014H	P3OE	2DFFH	CKDIVCR	50FFH	IRQ0ER	FAFFH
P25IOCFG	1015H	P4OE	2EFFH	MDLCKCR	51FFH	IRQ1ER	FBFFH
P26IOCFG	1016H	P0PURSELRL	30FFH	MISCR1	56FFH	IRQ2ER	FCFFH
P27IOCFG	1017H	P0PURSELRH	31FFH	MISCR2	5BFFH	IRQ3ER	FDFFH
P30IOCFG	1018H	P1PURSELRL	32FFH	MISCR3	59FFH	IRQ4ER	FEFFH
P31IOCFG	1019H	P1PURSELRH	33FFH	XCLKDIVR	E6FFH	IRQ5ER	FFFFH
P32IOCFG	101AH	P2PURSELRL	34FFH	ESDCR	57FFH		
P33IOCFG	101BH	P2PURSELRH	35FFH	SLPWKCR	52FFH	**EFC 模块寄存器：**	
P34IOCFG	101CH	P3PURSELRL	36FFH	SLPWKDR	53FFH	FDR	B0FFH
P35IOCFG	101DH	P3PURSELRH	37FFH	SLPWKSR	5DFFH	FADRL	B1FFH
P36IOCFG	101EH	P4PURSELRL	38FFH	RSTSR	54FFH	FADRH	B2FFH
P37IOCFG	101FH	P4PURSELRH	39FFH	LVDCR	55FFH	FCR	B3FFH

寄存器名	XRAM 地址	寄存器名	XRAM 地址	寄存器名	XRAM 地址	寄存器名	XRAM 地址
FPSR	B4FFH	SPIBPR	E3FFH	I2CMDR	CBFFH	LEDDR13	110DH
FDIV	B5FFH	SPISR	E4FFH	I2CMCTR	CCFFH		
FCRCER	B6FFH	SPIDR	E5FFH	I2CMSR	CDFFH	**PWM 寄存器：**	
FCRCSR	B7FFH					PWM1CR	68FFH
		WDT 寄存器：		**SensorAdc 寄存器：**		PWM2CR	69FFH
UART1寄存器：		WDTCR	60FFH	SCR0	90FFH	PWM1LRL	6AFFH
UCR1	D0FFH	WDTSETR	61FFH	SCR1	91FFH	PWM1LRH	6BFFH
USETR1	D1FFH	WDTSR	62FFH	SCR2	92FFH	PWM1HRL	6CFFH
UBPRL1	D2FFH	WDTENR	63FFH	SCR3	93FFH	PWM1HRH	6DFFH
UBPRH1	D3FFH			SSCAPL	94FFH	PWM2LR	6EFFH
UIER1	D4FFH	**RTC 寄存器：**		SSCAPH	95FFH	PWM2HR	6FFFH
USR1	D5FFH	RTCCR	E8FFH	SSR	96FFH		
UTX1R	D700H	RTCSR	E9FFH	SDRL	97FFH	**Timer 寄存器：**	
URX1R	D701H	RTCHDR	EAFFH	SDRH	98FFH	T0CKCR	70FFH
UTX1DMACR	D702H	RTCMDR	EBFFH	SFLTCNT	A9FFH	T0CR	71FFH
UTX1DMAADDRL	D703H	RTCSDR	ECFFH	SSELR	9AFFH	T0CMP0	72FFH
UTX1DMAADDRH	D704H	RTCAHDR	EDFFH	SNUMR	9BFFH	T0CMP1	73FFH
URX1DMACR	D705H	RTCAMDR	EEFFH	SWKCR	9EFFH	T0SR	74FFH
URX1DMAADDRL	D706H	RTCASDR	EFFFH			T0CNT	76FFH
URX1DMAADDRH	D707H					T1CKCR	78FFH
		ADC 寄存器：				T1CR	79FFH
UART2寄存器：		ADCDIVCR	B8FFH			T1CMPR	7AFFH
UCR2	18FFH	ADCCR1	B9FFH	**LED 寄存器：**		T1SR	7BFFH
USETR2	19FFH	ADCSR	BAFFH	LEDCR	C0FFH	T1CNT	7DFFH
UBPRL2	1AFFH	ADCCR2	BBFFH	LEDSETR	C1FFH	T2CKCR	80FFH
UBPRH2	1BFFH	ADCDRL	BCFFH	LEDDIVR	C2FFH	T2CR	81FFH
UIER2	1CFFH	ADCDRH	BDFFH	GRIDVLDL	C3FFH	T2CMPR0L	82FFH
USR2	1DFFH			GRIDVLDH	C4FFH	T2CMPR0H	83FFH
UTX2R	1F00H	**I2C Slaver 寄存器：**		LEDDR0	1100H	T2CMPR1L	84FFH
URX2R	1F01H	I2CSCR	D8FFH	LEDDR1	1101H	T2CMPR1H	85FFH
UTX2DMACR	1F02H	I2CSSR	D9FFH	LEDDR2	1102H	T2SR	86FFH
UTX2DMAADDRL	1F03H	revserve	DAFFH	LEDDR3	1103H	T2CNTL	AAFFH
UTX2DMAADDRH	1F04H	I2CSDR	DBFFH	LEDDR4	1104H	T2CNTH	ABFFH
URX2DMACR	1F05H	I2CSADDR	DCFFH	LEDDR5	1105H	T3CR	88FFH
URX2DMAADDRL	1F06H	I2CSRHADR	DDFFH	LEDDR6	1106H	T3CKCR	89FFH
URX2DMAADDRH	1F07H	I2CSHADR	DEFFH	LEDDR7	1107H	T3OCR	8AFFH
				LEDDR8	1108H	T3CMPR0L	8BFFH
SPI 寄存器：		**I2C Master 寄存器：**		LEDDR9	1109H	T3CMPR0H	8CFFH
SPICR	E0FFH	I2CMPRERL	C8FFH	LEDDR10	110AH	T3CMPR1L	8DFFH
SPISFMR	E1FFH	I2CMPRERH	C9FFH	LEDDR11	110BH	T3CMPR1H	8EFFH
SPIIER	E2FFH	I2CMCR	CAFFH	LEDDR12	110CH	T3SR	8FFFH

寄存器名	XRAM 地址	寄存器名	XRAM 地址	寄存器名	XRAM 地址	寄存器名	XRAM 地址
T3CNTL	ADFFH						
T3CNTH	AEFFH						

3.2、内核特殊功能寄存器

CBM73xxF 芯片有 16 字节内核特殊功能寄存器，内核特殊功能寄存器与片内 RAM 共用 80H-FFH 地址，但是它们拥有不同的物理存储区，使用不同的指令来访问。

表 3-2： CBM73xxF 内核特殊功能寄存器

内核特殊功能寄存器	地址
P0	80H
SP	81H
DPL	82H
DPH	83H
PSEQ	86H
PCON	87H
VTOR	88H
P1	90H
P2	A0H
IE	A8H
P3	B0H
IP2	B8H
IP1	C8H
P4	C0H
PSW	D0H
ACC	E0H
B	F0H
	FFH

累加器(ACC)

累加器 ACC 是一个常用的专用寄存器，指令系统中采用 A 作为累加器的助记符。

B 寄存器(B)

在乘除法指令中，会用到 B 寄存器。在其他指令中，B 寄存器可作为暂存器来使用。

栈指针(SP)

栈指针 SP 是一个 8 位专用寄存器，在执行 PUSH、各种子程序调用、中断响应等指令时，SP 先加一，再将数据压栈；执行 POP、RET、RETI 等指令时，数据退出堆栈后 SP 再减一。**堆栈栈顶可以是片上内部 RAM(30H-FFH)的任意地址**，系统复位后，SP 初始化为 07H，使得堆栈实际上从 08H 地址开始。

数据指针(DPTR)

数据指针是一个 16 位专用寄存器。编程时，DPTR 既可以作为一个 16 位寄存器使用，也可以作为 2 个独立的 8 位寄存器 DPH 和 DPL 分开使用，即：

DPH　DPTR 高 8 位字节

DPL　DPTR 低 8 位字节

DPTR 通常在访问外部数据存储器时作地址指针使用。由于外部数据存储器的寻址范围为 64KB，故把 DPTR 设计为 16 位。

中断向量偏移地址寄存器(VTOR)

VOTR 寄存器保存的是中断向量的偏移地址。设置该寄存器后，当中断发生时，程序跳转到新的地址上去执行。(详细说明见“5.2、中断入口地址”)

当设置编译器，使中断服务程序的入口不在程序空间的开始处(0x0000)时，可以选择设置此寄存器。

VTOR 寄存器上电复位值为 0x00，即复位时中断入口地址和 8051 中断入口地址一致。若用户开发的程序的起始地址为 0x0800，则可以将中断入口地址的高 8 位设置到中断入口偏移地址寄存器 VTOR 中。

程序状态字(PSW)

程序状态字是一个 8 位寄存器，用于存放程序运行中的各种状态信息。

其中有些位的状态是根据程序执行结果，由硬件自动设置的，而有些位的状态则使用软件方法设定。PSW 的位状态可以用专门指令进行测试，也可以用指令读出。一些条件转移指令根据 PSW 某些位的状态进行程序转移。

PSW 寄存器包含：

- 算术状态标志位
- 用户自定义标志位
- 工作寄存器页选择位
- 奇偶校验位

- **寄存器 PSW：**

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CY	AC	F0	RS1	RS0	OV	F1	P
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7　　CY:进位标志位
　　0 = 算术或逻辑运算中，没有进位或借位发生
　　1 = 算术或逻辑运算中，有进位或借位发生

bit 6　　AC：辅助进位标志位
　　0 = 算术或逻辑运算中，没有辅助进位或借位发生
　　1 = 算术或逻辑运算中，有辅助进位或借位发生

bit 5　　F0:用户自定义标志位
　　提供给用户自定义的标志位

bit 4-3　　RS[1:0]：R0-R7 工作寄存器页选择位
　　00 = 页 0(映射到 00H-07H)
　　01 = 页 1(映射到 08H-0FH)
　　10 = 页 2(映射到 10H-17H)
　　11 = 页 3(映射到 18H-1FH)

bit 2　　OV：溢出标志位
　　0 = 没有溢出发生
　　1 = 有溢出发生

bit 1　　F1:用户自定义标志位
　　提供给用户自定义的标志位

bit 0　　P：奇偶校验位
　　0 = 累加器 A 中值为 1 的位数为偶数
　　1 = 累加器 A 中值为 1 的位数为奇数

中断使能寄存器(IE)

IE 是内核中断使能寄存器，包含中断 0、中断 1、…、中断 5 的使能位以及全局中断使能位。

注：当有中断条件产生时，无论对应的中断使能位或全局使能位(IE 中的 EA)的状态如何，中断标志位都将置 1。**用户软件应在使能某个中断之前，确保先将相应的中断标志位清零。**

寄存器 IE：

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
EA	-	EX5	EX4	EX3	EX2	EX1	EX0
R/W-0	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 EA：全局中断使能位
0 = 禁止所有中断
1 = 使能所有未被屏蔽的中断

bit 6 保留未用

bit 5 EX5：中断 5 使能位
0 = 禁止中断 5 中断
1 = 使能中断 5 中断

bit 4 EX4：中断 4 使能位
0 = 禁止中断 4 中断
1 = 使能中断 4 中断

bit 3 EX3：中断 3 使能位
0 = 禁止中断 3 中断
1 = 使能中断 3 中断

bit 2 EX2：中断 2 使能位
0 = 禁止中断 2 中断
1 = 使能中断 2 中断

bit 1 EX1：中断 1 使能位
0 = 禁止中断 1 中断
1 = 使能中断 1 中断

bit 0 EX0：中断 0 使能位
0 = 禁止中断 0 中断
1 = 使能中断 0 中断

中断优先级控制寄存器(IP1、IP2)

CBM73xxF 中断优先级有 4 级，IP 寄存器控制 6 个中断的优先级，如下所示。

寄存器 IP1：

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
IP1_BIT[7:6]		IP1_BIT[5:4]		IP1_BIT[3:2]		IP1_BIT[1:0]	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 IP1_BIT[7:6]：中断 3 优先级控制位

同下

bit 5-4 IP1_BIT[5:4]：中断 2 优先级控制位

同下

bit 3-2 IP1_BIT[3:2]：中断 1 优先级控制位

同下

bit 1-0 IP1_BIT[1:0]：中断 0 优先级控制位

00 = 0 级中断优先级（最低）

01 = 1 级中断优先级

10 = 2 级中断优先级

11 = 3 级中断优先级（最高）

寄存器 IP2：

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	IP2_BIT[3:2]		IP2_BIT[1:0]	
U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-4 保留未用

bit 3-2 IP2_BIT[3:2]：中断 5 优先级控制位

同下

bit 1-0 IP2_BIT[1:0]：中断 4 优先级控制位

00 = 0 级中断优先级（最低）

01 = 1 级中断优先级

10 = 2 级中断优先级

11 = 3 级中断优先级（最高）

注：

1、高优先级的中断可以打断低优先级中断函数的处理，反之不行；

2、同一级的中断，后产生的中断不能打断目前正在处理的中断；

3、同时产生多个中断时，同一级的中断优先级为 IP1_BIT[1:0]> IP1_BIT[3:2] > IP1_BIT[5:4] > IP1_BIT[7:6] > IP2_BIT[1:0]>IP2_BIT[3:2]；

4、3 级中断优先级最高，0 级中断优先级最低。

低电耗控制寄存器(PCON)

低功耗控制(PCON)寄存器包含：

- 休眠模式使能位
- 空闲模式使能位

寄存器 PCON：

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	SLEEP	IDLE
U-x	U-x	U-x	U-x	U-x	U-x	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-2 保留未用

bit 1 SLEEP：SLEEP 模式使能位

0 = 禁止 SLEEP 模式

1 = 使能 SLEEP 模式

bit 0 IDLE：IDLE 模式使能位

0 = 禁止 IDLE 模式

1 = 使能 IDLE 模式

注：

1、芯片处于调试模式下，不能通过调试行为改变该寄存器的值，仅可以在运行模式下通过软件代码改变。

2、写 PCON 寄存器前，必需先依次 写 PSEQ = 0x0A ，PSEQ = 0x05。

3、不能同时使能 SLEEP 与 IDLE 模式，即不能写 PCON=0x03。

写 PCON 前的保护序列寄存器(PSEQ)

PSEQ 是操作 PCON 寄存器前的保护序列寄存器。写 PCON 寄存器前，必需先依次写 PSEQ = 0x0A ，PSEQ = 0x05。

寄存器 PSEQ：

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-				
U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-4 保留未用

bit 3-0 PSEQ[3:0]：写 PCON 前的保护序列寄存器

必需先依次 写 PSEQ[3:0] = 0x0A、PSEQ[3:0] = 0x05，才能写 PCON 寄存器。

注：芯片处于在线调试模式下，不能通过调试行为改变该寄存器的值，仅可以在运行模式下通过软件代码改变。

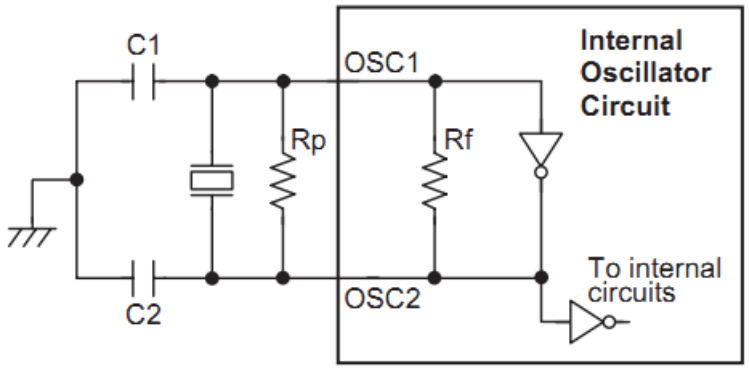
4、 振荡器模块

CBM73xxF 系列 MCU 支持内部时钟和外接晶振时钟，系统共有四个时钟源分别是：内部高速 48M（IHCLK）、内部低速 800K（ILCLK）、外部高速 M 级晶振（EHCLK）、外部低速 K 级晶振（ELCLK）。

表 4-1、时钟源参考值大小

时钟源简称	最小值	典型值	最大值	时钟源精度（-40℃~85℃）	引脚
IHCLK（内部高速时钟）	-	48M	-	±1%	-
ILCLK（内部低速时钟）	-	800K	-	±1%	-
EHCLK（外部高速时钟）	4M	8M	16M	取决于晶体精度	P25/P26 （P25IOCFG=0x09 P26IOCFG=0x09）
ELCLK（外部低速时钟）	-	32.768K	-	取决于晶体精度	

外部晶体电路(M 级、K 级晶振只能二选一，不能同时外挂)：



（芯片内部提供 C1 C2 Rp，通过寄存器 XTALCR0、XTALCR1、XTALCR2 配置其大小，使用内部电容电阻则不需要外接了）

用户根据需求选择相应的时钟源（内部或外部时钟），再经相关时钟配置寄存器、时钟分频寄存器最终会产生两大时钟：系统主时钟（MCLK）、系统副时钟（SCLK1、SCLK2、SCLK3）；其中，MCLK 为系统指令执行时钟给内核使用；SCLK 为系统副时钟给各外设模块作计数时钟用。详细请见下表 4-2、图 4-1，时钟相关寄存器见表 4-3。

表 4-2、时钟关系映射总表

两大时钟		可选择时钟源	供使用的模块	备注
系统主时钟（MCLK）		IHCLK、EHCLK、ILCLK	系统内核、PWM、CRC16 校验模块	系统指令执行时钟
系统副时钟	SCLK1	IHCLK、EHCLK、ILCLK	Timer、SPI、I2C	各模块计数时钟
	SCLK2	EHCLK、ELCLK	RTC、Timer	
	SCLK3	ILCLK	LED	

注：

1、上表 4-2 中未描述出的模块，该些模块的时钟来自于 SYSCLK(系统所选时钟源)或直接来自于 IHCLK（内部 48M）、ILCLK（内部 800K）,详情请参考下图 4-1 描述；

2、M 级、K 级晶振只能二选一，不能同时外挂；

3、外挂 M/K 级晶振必须配置 P25IOCFG=0x09,P26IOCFG=0x09，以使能 IO 的晶振引脚功能。

4.1、系统总时钟框图

振荡器模块具有多种时钟源，经过不同的分频器供给 MCU 内核和外设工作。下图 4-1 为系统总时钟框图：

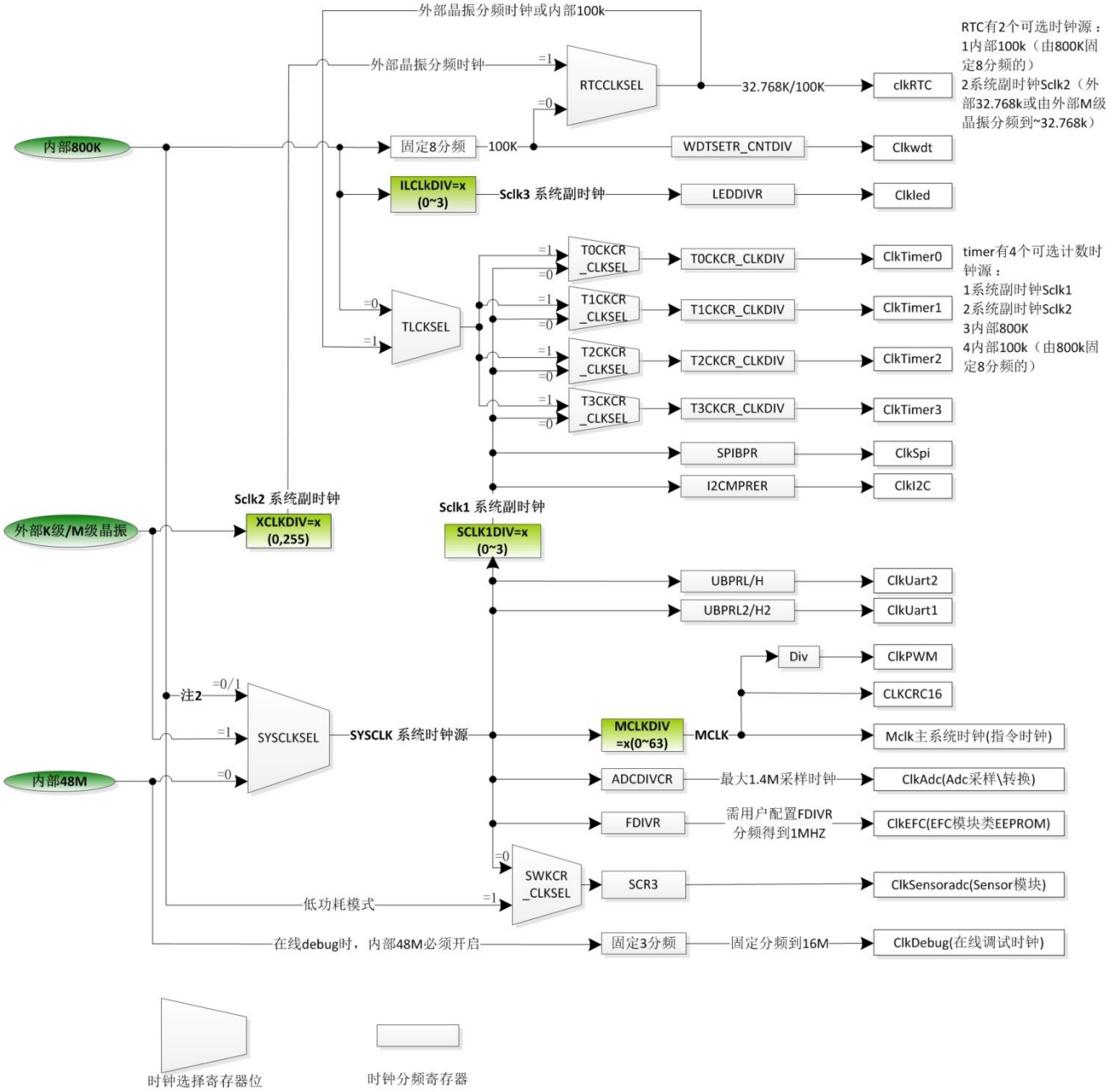


图 4-1、MCU 系统总时钟框图

注：1、上图 4-1 中的 Clk_SW 表示时钟选择寄存器；SYSCLK 为系统所选择的系统时钟源（IHCLK、EHCLK、ILCLK）；Div 表示时钟分频寄存器（MCLKDIV、SCLK1DIV、XCLKDIV、ILCLKDIV 为系统级时钟分频寄存器，其它 Div 为模块级时钟分频寄存器）；

2、上图 4-1 中“注 2”表示：当选择外部 M 级晶振为系统时钟源时且当外部 M 级晶振由于某种干扰意外停振时，系统可以通过配置使用自动切换内部低速 800K 为系统时钟源，待外部 M 级时钟恢复稳定后，再自动从 800K 切回到外部 M 级时钟为系统时钟（要实现此功能需要配置使能相关控制寄存器,详细请见控制寄存器“MISCR3”）；

3、MCLK = SYSCLK /(CKDIVCR_MCLKDIV[5:0] + 1)；

4、SCLK1 = SYSCLK /(MISCR2_SCLK1DIV[1:0] + 1)；

5、SCLK2 = (EHCLK 或 ELCLK)/(XCLKDIVR[7:0] + 1)；

6、SCLK3 = SYSCLK /$2^{\text{CKDIVCR_ILCLKDIV[1:0]}}$；

4.2、系统相关寄存器的定义

振荡器模块有 4 个时钟源：内部高速 48M（IHCLK）、内部低速 800K（ILCLK）、外部高速 M 级晶振（EHCLK）、外部低速 K 级晶振（ELCLK）。系统时钟相关控制寄存器如下表 4-3 所示，具体作用请见各寄存器描述。

MCU 上电时，系统默认选择 IHCLK 为时钟源，IHCLK 经 4 分频后得到 12MHZ 时钟作为 MCLK。MCLK 最高支持为 16MHZ，即 IHCLK 的 3 分频，所以配置 MCLKDIV[5:0]时应注意值不能小于 2。

表 4-3、时钟寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
CKDIVCR	0x50FF	XRAM	系统时钟分频控制寄存器	1100 0011
MDLCKCR	0x51FF	XRAM	各功能模块时钟控制寄存器	0000 1000
MISCR1	0x56FF	XRAM	系统控制寄存器1	0000 0000
MISCR2	0x5BFF	XRAM	系统控制寄存器2	1000 0101
MISCR3	0x59FF	XRAM	系统控制寄存器3	0000 0000
XCLKDIVR	0xE6FF	XRAM	外部晶振的分频寄存器	0000 0000
XTALCR0	0x1042	XRAM	外部晶振控制寄存器0	0100 0111
XTALCR1	0x1043	XRAM	外部晶振控制寄存器1	0111 1111
XTALCR2	0x1044	XRAM	外部晶振控制寄存器2	1111 1111

注：上表 4-3 中寄存器地址 0x1042~0x1044 对应寄存器均不支持按位操作，只能对整个寄存器进行字节的操作；表中其它寄存器支持按位操作或直接对字节操作，可参考“附录VI XRAM 类型地址寄存器的操作说明”；**且配置寄存器 OSC800KTRIMR、PLLTRIMR、XTALCR0、XTALCR1、XTALCR2、LDOCR0、LDOCR1 之前必须打开 IOC 模块的时钟。**

CKDIVCR：时钟分频控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
ILCLKDIV1	ILCLKDIV0	MCLKDIV5	MCLKDIV4	MCLKDIV3	MCLKDIV2	MCLKDIV1	MCLKDIV0
R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1；0：清零；x：未知值）							

bit 7-6 ILCLKDIV[1:0]：内部低速 800K 时钟分频器，分频得到 SCLK3=800K/$2^{ILCLKDIV[1:0]}$ 给 LED 模块

00 = 1 分频

01 = 2 分频

10 = 4 分频

11 = 8 分频

bit 5-0 MCLKDIV[5:0]：系统所选择时钟源的分频控制位，分频后供给系统主时钟（MCLK）

000000 = 1 分频

000001 = 2 分频

……

111111 = 64 分频

注：

1、系统主时钟 MCLK=SYSCLK/(MCLKDIV[5:0]+1)；(SYSCLK 为系统所选择的时钟源，时钟框图请参考“4.1、系统总时钟框图”)

2、由于 CPU 最快只能运行在 16M 时钟下，所以 SYSDIV[5:0]不能小于 000010(3 分频)；

3、ILCLKDIV 对内部 800K 分频后，得到系统副时钟 SCLK3=800K/$2^{ILCLKDIV[1:0]}$ 给 LED 提供工作时钟源。

MDLCKCR：模块时钟控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TLCKSEL	PWMEN	SPIEN	UART1EN	IOCEN	TIMEREN	LEDEN	WDTEN
R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 TLCKSEL：Timer 模块的低速时钟选择位（详见“4.1、系统总时钟框图”）

0 = Timer 模块的低速时钟源选择为内部 800K

1 = Timer 模块的低速时钟源选择为内部 100K 或外部晶振（32.768K\M 级晶振）。（如果 MISCR2_RTCCLKSEL=0，则 Timer 模块的低速时钟源为内部 100K；如果 MISCR2_RTCCLKSEL=1，则低速时钟源选择为外部晶振。）

bit 6 PWMEN：PWM 时钟使能位

0 = 关闭 PWM 时钟，PWM 不工作

1 = 使能 PWM 时钟，PWM 工作

bit 5 SPIEN：SPI 时钟使能位

0 = 关闭 SPI 时钟，SPI 不工作

1 = 使能 SPI 时钟，SPI 工作

bit 4 UART1EN：UART1 时钟使能位

0 = 关闭 UART1 时钟，UART1 不工作

1 = 使能 UART1 时钟，UART1 工作

bit 3 IOCEN：GPIO 时钟使能位

0 = 关闭 GPIO 时钟，GPIO 不工作

1 = 使能 GPIO 时钟，GPIO 工作

bit 2 TIMEREN：Timer 时钟使能位

0 = 关闭 Timer 时钟，Timer 不工作

1 = 使能 Timer 时钟，Timer 工作

bit 1 LEDEN：LED 时钟使能位

0 = 关闭 LED 时钟，LED 不工作

1 = 使能 LED 时钟，LED 工作

bit 0 WDTEN：WDT 时钟使能位

0 = 关闭 WDT 时钟，WDT 不工作

1 = 使能 WDT 时钟，WDT 工作

注：

1、如果要配置某个模块的寄存器时，必需先配置 MDLCKCR 或 MISCR1，使能相应模块的时钟，否则配置无效；

2、工作状态下 IO 时钟使能控制位必须打开，不能关闭； 且在配置寄存器 OSC800KTRIMR、PLLTRIMR、XTALCR0、XTALCR1、XTALCR2、LDOCR0、LDOCR1 之前必须打开 IOC 模块的时钟。

MISCR1：系统控制寄存器 1

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	SENSORCKEN	UART2CKEN	-	FMFIFOEN	FMULRDEN	NVR2CRCEN	POREN
U-x	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 -保留未用

bit 6 SENSORCKEN：触摸模块时钟使能位

0 = 关闭 SENSOR 时钟，SENSOR 不工作

1 = 使能 SENSOR 时钟，SENSOR 工作

bit 5 UART2CKEN：UART2 时钟使能位

0 = 关闭 UART2 时钟，UART2 不工作

1 = 使能 UART2 时钟，UART2 工作

bit 4 -保留未用

bit 3 FMFIFOEN：Flash 预取指机制控制位（只有 bit2 的 FMULRDEN=1 时才能配置 FMFIFOEN=1，否则置 FMFIFOEN=0）

0 = 关闭 Flash 预取指机制，对系统时钟 MCLK 频率无影响

1 = 使能 Flash 预取指机制，对系统时钟 MCLK 频率产生 3 倍频率效果（见下表 4-4）

bit 2 FMULRDEN：Flash 重读机制控制位，建议使能且必须配置完时钟源后就使能

0 = 关闭 Flash 重读机制，对系统时钟 MCLK 频率无影响

1 = 使能 Flash 重读机制，对系统时钟 MCLK 频率产生 4 分频效果（见下表 4-4）

bit 1 NVR2CRCEN：NVR2 区的 CRC 动态校验使能控制位

0 = 不使能 NVR2 区 CRC 动态校验

1 = 使能 NVR2 区 CRC 动态校验

bit 0 POREN：上电复位禁止位

0 = 当掉电到 POR 复位电压门限时，复位系统

1 = 当掉电到 POR 复位电压门限时，不复位系统

注：

1、如果要配置某个模块的寄存器时，必需先配置 **MDLCKCR 或 MISCR1** 使能相应模块的时钟位，否则配置无效；

2、建议使能 flash 重读机制 FMULRDEN 和 flash 预取指机制 FMFIFOEN，且必须在配置完时钟源后再软件使能该两位。

3、表 4-4：预取指 FMFIFOEN 和读判决 FMULRDEN 对 mclk 的影响如下表（例如：MCLKDIV[5:0]=2、SYSCLK=48M）

FMFIFOEN（bit3）	FMULRDEN(bit2)	MCLK
0	0	16M
0	1（4 分频效果）	4M
1	0	×(不允许配置)
1（3 倍频效果）	1（4 分频效果）	12M

MISCR2：系统控制寄存器 2

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SCLK1DIV1	SCLK1DIV0	RTCCLKSEL	SYSCLKSEL	SYSCLKSWITCHEN	XTALEN	IHCLKCHKEN	IHCLKEN
R/W-1	R/W-0	R/W-0	R/W-0	W-0	R/W-1	R/W-0	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 SCLK1DIV[1:0]：副时钟 SCLK1 分频寄存器，对系统所选时钟源 SYSCLK 进行分频后给到 Timer、I2C、SPI 的工作时钟,最大 16MHZ， SCLK1 = SYSCLK/ (SCLK1DIV[1:0] + 1) 。(SYSCLK 为系统所选择的时钟源，时钟框图请参考“4.1、系统总时钟框图”)

00 = 1 分频

01 = 2 分频

10 = 3 分频

11 = 4 分频

bit 5 RTCCLKSEL：RTC 时钟源选择控制位

0 = RTC 选择内部 100K 时钟（由内部 800K 固定 8 分频得到的 100K 时钟）

1 = RTC 选择外部 32.768K 时钟（外部 32.768K 晶振、外部 M 级晶振经 XCLKDIV 分频后产生的~32.768K 时钟）

bit 4 SYSCLKSEL：系统时钟源选择控制位

0 = 选择内部 48M 高速时钟（IHCLK）为系统时钟源

1 = 选择外部 M 级高速时钟（EHCLK）为系统时钟源

bit 3 SYSCLKSWITCHEN：48M 作系统时钟时且 IHCLKCHKEN =1，当 48M 被干扰时是否自动切到内部 800k 作系统时钟源

0：48M 时钟异常时不会自动切换到 800k

1：48M 时钟异常时自动切换到 800k 作系统时钟源

bit 2 XTALEN：外部晶振使能控制位

0 = 关闭外部晶振

1 = 使能外部晶振

bit 1 IHCLKCHKEN：监测内部高速 48M 时钟

0 = 不开启监测 IHCLK 是否被干扰的功能

1 = 开启监测 IHCLK 功能，当监测到 IHCLK 被干扰到时自动重新开启 IHCLK

bit 0 IHCLKEN：内部高速时钟 48M 使能控制位

0 = 关闭内部高速 48M 时钟

1 = 开启内部高速 48M 时钟

注：

1、副时钟 SCLK1 = SYSCLK / (SCLK1DIV[1:0] + 1)；SCLK1 给 Timer、I2C、SPI 提供时钟源，最大 16MHZ；(SYSCLK 为系统所选择的时钟源，时钟框图请参考“4.1、系统总时钟框图”)

2、切换系统所选时钟源之前必需保证该时钟源已经开启且稳定，所以应该在开启该时钟源后并等待~10us 再切换；

3、必须始终保证 MCLK<=16MHZ,因此从低频率时钟源切到高频率时钟源时，必须先配置 CKDIVCR_MCLKDIV，以确保切换之后 MCLK<=16MHZ；而从高频率时钟源切到低频率时钟源时，必须先切换然后再配置 MCLKDIV，以确保切换之前的 MCLK<=16MHZ。

MISCR3：系统控制寄存器 3

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SYSCLKSR	SYSCLKFINTSR	SYSCLKFINTEN	XTALCHKEN	ESD1EN	ESD1SR	NVR2CRCSR	LVDSTOPWDT
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 SYSCLKSR：系统时钟源的工作状态标志（当系统时钟为外部时钟时，必须配置 XCLKDIR 把外部晶振分频到 12k~400k 范围才能检测实时状态）

0 = 所选系统时钟源正常

1 = 所选系统时钟源出错(当系统时钟为外部时钟时,使能 XTALCHKEN=1 则只要外部时钟出错就会自动选择 800K 为系统时钟源，此时支持软件切换到 48M 为系统时钟源；当系统时钟为 48M 时钟时，使能 IHCLKCHKEN=1 及 SYSCLKSWITCHEN=1 则只要 48M 出错就会自动选择 800K 为系统时钟源，此时支持软件重启 48M 为系统时钟源）

bit 6 SYSCLKFINTSR：系统时钟源工作异常时的中断标志位（前提是使能外部时钟的监测功能即 XTALCHKEN=1 或使能内部 48M 监测功能 IHCLKCHKEN=1）

0 = 所选系统时钟源工作正常或出错后已由软件清 0

1 = 所选系统时钟源工作出错异常，需要软件写 0 清 0（当系统时钟为外部时钟时，使能 XTALCHKEN=1 则只要外部时钟出错就会自动选择 800K 为系统时钟源，此时支持软件切换到 48M 为系统时钟源；当系统时钟为 48M 时钟时，使能 IHCLKCHKEN=1 及 SYSCLKSWITCHEN=1 则只要 48M 出错就会自动选择 800K 为系统时钟源，此时支持软件重启 48M 为系统时钟源）

bit 5 SYSCLKFINTEN：系统时钟源工作异常时的中断使能控制位（前提是使能外部时钟的监测功能即 XTALCHKEN=1 或使能内部 48M 监测功能 IHCLKCHKEN=1）

0 = 当所选系统时钟源为外部时钟或 48M 时钟且其发生异常时，不产生中断

1 =当所选系统时钟源为外部时钟或 48M 时钟且其发生异常时，产生中断

bit 4 XTALCHKEN：外部时钟状态监测功能控制位

0 = 不使能监测功能（当外部时钟作为系统时钟源时，若外部时钟停振，则系统停止工作）

1 =使能监测功能（当外部时钟作为系统时钟源时，若外部时钟停振，则系统自动切换内部 800K 为系统时钟源，系统继续工作；等外部时钟恢复正常后再自动切回外部时钟为系统时钟；或者用户软件检测到在发生外部时钟工作异常中断时，用户也可以软件控制切换内部 48M 为系统时钟,当外部晶振正常（XTALSR=0）时再切换到外部晶振为系统时钟源。）

bit 3 ESD1EN：ESD 功能使能控制位

0 = 关闭 ESD 功能

1 = 使能 ESD 功能

bit 2 ESD1SR：ESD 状态标志位

0 = 没有 ESD 事件发生

1 = 有 ESD 事件发生，需软件写 0 清 0

bit 1 NVR2CRCSR ：NVR2 区 CRC 动态校验错误标志

0 = NVR2 区 CRC 动态校验正确

1 = NVR2 区 CRC 动态校验错误

bit 0 LVDSTOPWDT：LVD 发现时是否停止 WDT 计数，防止 LVD 停止工作模式时 WDT 引发复位动作

0 = LVD 发生时，不停止 WDT 计数，WDT 继续正常工作

1 = LVD 发生时，停止 WDT 计数，WDT 停止工作，待 LVD 撤消后，WDT 再恢复工作

注：MISCR3 的 bit[7:4]为外部晶振或 48M 时钟的工作状态的相关控制位及相关工作状态位；当需要监测内部 48M 时，还需要配置使能 MISCR2 寄存器的 bit1 和 bit3。

XCLKDIVR：外部时钟分频寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
DIV7	DIV6	DIV5	DIV4	DIV3	DIV2	DIV1	DIV0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 XCLKDIVR [7:0]：对外部晶振进行分频得到 SCLK2，SCLK2= (EHCLK 或 ELCLK)/(XCLKDIVR [7:0] +1)分频后给 Timer、RTC 时钟。(SYSCLK 为系统所选择的时钟源，时钟框图请参考“4.1、系统总时钟框图”)

00000000 = 1 分频

00000001 = 2 分频

00000010 = 3 分频

……

11111101 = 254 分频

11111110 = 255 分频

11111111 = 256 分频

注：

1、系统副时钟 SCLK2 =（EHCLK、ELCLK） / (XCLKDIV[7:0] + 1)；SCLk2 给 Timer、RTC 提供时钟；

2、Timer 可以选择用 SCLK2（K 级或 M 级）作计数时钟；

3、RTC 可以选择 SCLK2 的 K 级时钟（外部 32.768K 晶振或由外部 M 级晶振分频得到的~32.768K）作时钟。

XTALCR0：外部晶振控制寄存器 0

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	DOSEL	RSEL	-	ICURRENTSEL	AVDSEL	RSDL1	RSDL0
U-x	R/W-1	R/W-0	U-0	R/W-0	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 保留未用

bit 6 DOSEL：Comparator 输出还是 Schmitt 输出选择**（必须为 1）**

0 = Schmitt 输出

1 = Comparator 输出

bit 5 RSEL：外部晶振振荡电阻选择位(建议选内部电阻)

0 = 外挂振荡电阻（如果选择了外接电阻则外挂 1~4M）

1 = 内置振荡电阻

bit 4 IRVALSEL：内置振荡电阻大小选择

0 = 2.6 MΩ

1 = 5.2 MΩ

bit 3 ICURRENTSEL：晶振内部 regulator 电流选择（晶振的频率越大，需要配置大的电流档位 0）

0 = 默认电流

1 = 1/2 * 默认电流

bit 2 AVDSEL：晶振内部 AVD 电压选择（必须为 1）

0 = 选择内部 regulator 2.0v

1 = 选择 VDD 电压

bit 1-0 RSDL [1:0]：K 或 M 级晶振限流电阻选择(K 级-固定为 11，M 级-值越大振幅越大)

XTALCR1：外部晶振控制寄存器 1

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
XTALSEL1	XTALSEL0	-	-	-	-	-	-
R/W-0	R/W-1	U-1	U-1	U-1	U-1	U-1	U-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 XTALSEL[1:0]：外部晶振大小选择

00 = 单 PIN K 级晶振（32.768K）

01 = 双 PIN K 级晶振（32.768K）

1x = 双 PIN M 级晶振

bit 5-0 XTALC1S[5:0]：Xin 端负载电容值调节

每个 bit 表示一个电容大小：bit5=12pf, bit4=8pf, bit3=4pf, bit2=2pf, bit1=1pf, bit0=0.5pf；

例如 XTALC1S[5:0]=b100001,则表示选择 12.5pf 电容

XTALCR2：外部晶振控制寄存器 2

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
XTALDRS1	XTALDRS0	-	-	-	-	-	-
R/W-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 XTALDRS[1:0]：晶振限流电阻选择,固定配为 11

bit 5-0 XTALC2S[5:0]：外部晶振 Xout 端负载电容值调节

每个 bit 表示一个电容大小：bit5=12pf, bit4=8pf, bit3=4pf, bit2=2pf, bit1=1pf, bit0=0.5pf；

例如 XTALC2S[5:0]=b100011,则表示选择 13.5pf 电容

注：外部晶振寄存器的参考配置如下：

1>、8M：XTALCR0 = 0x77 ; XTALCR1 = 0xFF ; XTALCR2 = 0xFF ;//用内部晶振电容电阻，则不需要再外挂电容电阻

2>、32.768K：XTALCR0 = 0x77 ; XTALCR1 = 0x7F ; XTALCR2 = 0xFF ;//用内部晶振电容电阻，则不需要再外挂电容电阻

4.3、时钟配置例程

例 4-1、内部 48M 作系统时钟源

```
MDLCKCR_IOCEN = 1<<3; //开启 IOC 模块时钟
//MISCR2_IHCLKCHKEN = 1<<1;//使内部 48M 时钟监测功能，若出错会自动重启 48M 时钟(可选)
//MISCR2_SYSCLKSWITCHEN = 1<<3 ;//使内部 48M 时钟监测出错时自动切换到 800k 作系统时钟源(可选)
MISCR2_IHCLKEN = 1<<0; //开启内部 48M 时钟
_nop_();_nop_();_nop_();_nop_();_nop_();//等待时钟稳定
CKDIVCR_MCLKDIV = 2<<0; //配置系统主时钟（MCLK）的分频数；MCLK = 48/(2+1) = 16M
MISCR2_SCLK1DIV = 2<<6 ; //配置外设模块的计数时钟（系统副时钟 SCLK1）的分频数；SCLK1=48/(2+1) = 16M
MISCR2_SYSCLKSEL = 0<<4; //选择内部高速时钟 48M 为系统时钟源
CKDIVCR_ILCLKDIV = 0<<6; //配置外设模块的计数时钟（系统副时钟 SCLK3）的分频数；SCLK3 = 800K/2^0 = 800K
MISCR2_RTCCLKSEL = 0<<5;//0：RTC 选择内部 100K，1：选择外部时钟（XCLKDIVR 分频后）得到的 32.768K
MDLCKCR_TLCKSEL = 0<<7 ;//timer 模块低速时钟选择，0-内部 800K；1-外部晶振经 XCLKDIVCR 分频后的时钟
/*//以下则根据实际情况是否配置外部 K 级晶振及 RTC 的时钟
```

```
P25IOCFG = 0x09;//K 级晶振引脚
P26IOCFG = 0x09;//K 级晶振引脚
P2OE_5  = 0x20;
P2OE_6  = 0x40;
P2PU_5  = 0x00;
P2PU_6  = 0x00;
XTALCR0 = 0x77;
XTALCR1 = 0x7f;
XTALCR2 = 0xff;
MISCR2_XTALEN = 1<<2; //使能外部晶振
_nop_() ;_nop_() ;_nop_() ;_nop_() ;_nop_() ;//延时等待外部晶振稳定
XCLKDIVR = 0; //配置外设模块的计数时钟（系统副时钟 SCLK2）的分频数；SCLK2 = 32.768K/(0+1) = 32.768K
MISCR2_RTCCLKSEL = 1<<5;//0：RTC 选择内部 100K，1：选择外部时钟（XCLKDIVR 分频后）得到的 32.768K
*/
```

注：1、当系统时钟源由较慢的时钟源（外部晶振）切换到较快时钟（内部 48M）时，必须先配置 MCLK、SCLK1 对应的分频数，再切换系统时钟源；

2、配置寄存器 OSC800KTRIMR、PLLTRIMR、XTALCR0、XTALCR1、XTALCR2、LDOCR0、LDOCR1 前必须打开 IOC 模块的时钟。

例 4-2、外部 8M 晶振作系统时钟源

```
MDLCKCR_IOCEN = 1<<3; //开启 IOC 模块时钟
P25IOCFG = 0x09;
P26IOCFG = 0x09;
P2OE_5  = 0x20;
P2OE_6  = 0x40;
P2PU_5  = 0x00;
P2PU_6  = 0x00;
XTALCR0 = 0x77;
XTALCR1 = 0xff;
XTALCR2 = 0xff;
MISCR2_XTALEN = 1<<2;//使能外部晶振
_nop_();_nop_();_nop_();_nop_();_nop_();//延时等待外部晶振稳定
MISCR2_SYSCLKSEL = 1<<4;//选择外部晶振作为系统时钟源 SYSCLK=8M
CKDIVCR_MCLKDIV = 0<<0;//系统主时钟(MCLK) = SYSCLK / (CKDIVCR_MCLKDIV + 1)=8M
MISCR2_SCLK1DIV = 0<<6 ;//系统副时钟 1（SCLK1）= SYSCLK / (MISCR2_SCLK1DIV + 1) =8M
XCLKDIVR = 0; //系统副时钟 2（SCLK2）= EHCLK / (XCLKDIVR + 1)=8M
CKDIVCR_ILCLKDIV = 0<<6;//系统副时钟 3（SCLK3）=ILCLK/2^ILCLKDIV[1:0]=800 / 2^ILCLKDIV[1:0]=800K
MISCR2_RTCCLKSEL = 0<<5;//选择 RTC 外部时钟，0:选择内部 100K 时钟；1：选择外部晶振分频后的时钟 SCLK2（32.768K）
MDLCKCR_TLCKSEL = 0<<7 ;//Timer 模块低速时钟选择内部 800K（0-内部 800K；1-外部晶振经 XCLKDIVR 分频后时钟）
```

注：

1、当系统时钟源由较快时钟（内部 48M）切换到较慢的时钟源（外部晶振）时，必须先切换系统时钟源，再配置 MCLK、SCLK1 对应的分频数；

2、选择外部晶振为系统时钟源时，RTC 只能选择内部 100K 为时钟或者是由 M 级晶振分频得到的~32.768K 时钟；

3、配置寄存器 OSC800KTRIMR、PLLTRIMR、XTALCR0、XTALCR1、XTALCR2、LDOCR0、LDOCR1 前必须先打开 IOC 模块的时钟。

例 4-3、外部 8M 作系统时钟源，并使能对外部晶振监测功能

```
MDLCKCR_IOCEN = 1<<3 ;//开启 IOC 模块时钟
P25IOCFG = 0x09;
P26IOCFG = 0x09;
P2OE_5  = 0x20;
P2OE_6  = 0x40;
P2PU_5  = 0x00;
P2PU_6  = 0x00;
XTALCR0 = 0x77;
XTALCR1 = 0xff;
XTALCR2 = 0xff;
MISCR2_XTALEN = 1<<2 ;//1:外部晶振使能
_nop_();_nop_();_nop_();_nop_();_nop_();//延时等待外部晶振稳定
MISCR2_SYSCLKSEL = 1<<4;//选择外部晶振作为系统时钟源 SYSCLK=8M
CKDIVCR_MCLKDIV = 0<<0 ;//MCLK = SYSCLK / (CKDIVCR_MCLKDIV + 1)=8M
MISCR2_SCLK1DIV = 0<<6 ;//SCLK1= SYSCLK / (MISCR2_SCLK1DIV + 1)=8M
XCLKDIVR = 0 ;//SCLK2 = EHCLK / (XCLKDIVR + 1)=8M
CKDIVCR_ILCLKDIV = 0<<6 ;//SCLK3 = ILCLK/2^ILCLKDIV[1:0]=800 / 2^ILCLKDIV[1:0]=800K
MISCR2_RTCCLKSEL = 0<<5 ;//0:RTC 内部 100K
MDLCKCR_TLCKSEL = 0<<3 ;//定时器低速时钟选择
//以下是使能外部晶振监测功能的命令代码
MISCR3_XTALCHKEN = 1<<4 ;//使能对外部晶振监测功能
MISCR3_SYSCLKFINTEN = 1<<5 ;//使能外部晶振出错时，产生中断
IRQ2ER_SYSCLKFINT = 1<<4 ;//使外设模块中断
EX2 = 1 ;//内核中断 2
EA = 1 ;//使能总中断
```

注：

1、当系统时钟源由较快时钟（内部 48M）切换到较慢的时钟源（外部晶振）时，必须先切换系统时钟源，再配置 MCLK、SCLK1 对应的分频数；

2、选择外部晶振为系统时钟源时，RTC 只能选择内部 100K 为时钟或者是由 M 级晶振分频得到的~32.768K 时钟；

3、配置寄存器 OSC800KTRIMR、PLLTRIMR、XTALCR0、XTALCR1、XTALCR2、LDOCR0、LDOCR1 前必须先打开 IOC 模块的时钟。

5、 MCU 的特性

CBM73xxF 系列芯片的特性功能旨在最大限度地提高系统的可靠性、稳定性、抗干扰性能等指标；通过减少外部分立元件降低设计成本；充分的考虑和提高用户对 MCU 资料配置的灵活度，降低了开发周期等。

特性功能包括如下：

- 复位
- 低电压监测功能（LVD）
- 中断
- 低功耗模式
- 静电防护（ESD）
- RAM、XRAM 偶校验
- CRC16 校验（有助于 class B 认证）
- 代码编程加密保护
- 在线 Debug &烧录
- 在线升级更新用户程序

5.1、复位&LVD

CBM73xxF 系例 MCU 支持多种复位，如下：

- POR 复位
- WDT 复位
- LVD 复位
- 硬件复位 $\overline{\text{RST_PAD}}$
- 软件指令复位

片上复位电路的简化框图如下图 5-1，复位相关寄存器如下表 5-1：

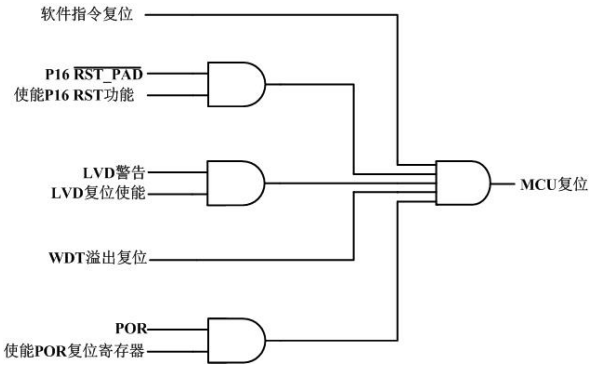


图 5-1、复位电路简化框图

当发生以上任何一种复位时，MCU 的内核、IO、WDT、timer、PWM、rtc、adc、sensoradc、spi、ii2、EFC、led、xtal 等所有外设模块都会停止运行；需要注意的是：

1、在任何一种复位过程中 P16 都会映射成复位引脚功能（P16 io 口在芯片内部有上拉，因此 P16 io 外部不能接下拉，否则 P16 可能为低电平会导致 MCU 一直在复位中）；

2、在任何一种复位过程中除 P16 外的其它 IO 均为高阻状态；

3、在任何一种复位完成后，所有 IO 均为高阻状态；

4、在任何一种复位完成后 RSTSR 寄存器会记录下当前的复位源标志、而其它所有的用户寄存器都会恢复为上电的默认值；

5、发生 por 复位后，iram、xram 的内容是不确定的；而发生 pad/lvd/wdt/srst 复位后，iram、xram 的内容则保持为复位前的内容状态；因此发生任何一种复位后都建议用户程序在定义变量时对变量进行初始化，否则可能因为变量值异常而引发程序逻辑错误。

表 5-1、复位寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
RSTSR	0x54FF	XRAM	系统复位标志寄存器	0000 0001
MISCR1	0x56FF	XRAM	系统控制寄存器1(控制位：MISCR1_POREN)	0000 0000
LVDCR	0x55FF	XRAM	低电压控制寄存器	0000 0010
MISCR3	0x59FF	XRAM	系统控制寄存器3(控制位：MISCR3_LVDSTOPEN)	0000 0000
P16IOCFG	0x100E	XRAM	P16功能复用寄存器 P16IOCFG .bit7=0:RST；bit7=1:其它功能，详见下表7-2）	1100 0000
SRSTCR	0x5C00	XRAM	软件指令复位控制寄存器	0000 0000

以上表中寄存器 P16IOCFG、SRSTCR 均不支持按位操作，只能对整个寄存器进行字节的操作；其它寄存器支持按位操作或直接对字节操作。

RSTSR：系统复位标志寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	SRST	HRST	LVD	WDT	POR
U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 4 SRST：软件指令复位标志位
0 = 无
1 = 发生软件指令复位，可写 0 清 0

bit 3 HRST：硬件复位标志位
0 = 无
1 = 发生硬件复位，可写 0 清 0

bit 2 LVD：低电压复位标志位
0 = 无
1 = 发生 LVD 复位，可写 0 清 0

bit 1 WDT：看门狗复位标志位
0 = 无
1 = 发生 WDT 复位，可写 0 清 0

bit 0 POR：上电复位标志位
0 = 无
1 = 发生 POR 复位，可写 0 清 0

注：

1、MCU 在上电初始化时，可通过软件查询此寄存器的标志，可以知道当前发生复位的原因。

POR 复位

当 MCU 的供电电压 VDD 达到足以使芯片正常工作的电压之前，MCU 的上电复位电路会使芯片保持在复位状态。直到 VDD 电压满足后，MCU 开始初始化系统各资源器件（工作寄存器组\SP\DPTR 等内核特殊功能寄存器、外设特殊功能寄存器、RAM、XRAM 等），初始化完成后从 0000H 地址开始运行程序。

MISCR1 寄存器 bit0 位（MISCR1_POREN）是上电复位使能禁止位，即 MISCR1_POREN = 0 时，上电复位功能使能；当 MISCR1_POREN = 1,上电复位功能禁止（详情请见“4.2、MISCR1”寄存器描述），建议 MISCR1_POREN = 0，使能上电复位功能。

WDT 复位

使能 WDT 功能，如果 WDT 发生连续第二溢出时，则 MCU 发生复位，WDT 溢出时间可设 160us~21s，则 WDT 复位时间可设为 320us~42s；WDT 的详细操作请参考“8、WDT”章节；

LVD 复位&LVD 监测功能

LVDCR：低电压控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
LVDSR	FLTLEN2	FLTLEN1	FLTLEN0	VOLSEL2	VOLSEL1	OPER1	OPER0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 LVDSR：LVD 状态标志位

0 = 无 LVD 告警发生

1 = 检测到低电压告警事件（跟随 LVD 信号），需软件写 0 清 0

bit 6-4 FLTLEN[2:0]：对 LVD 告警信号毛刺滤波宽度设置，滤波时间 =（2 的 n 次方）us

000 = 1us

001 = 2us

010 = 4us

……

111 = 128us

bit 3-2 VOLSEL[3:2]：低电压监测阈值设置

00 = 2.9v

01 = 3.3v

10 = 3.7v

11 = 4.2v

bit 1-0 OPER[1:0]：当监测到低压事件时的处理措施设置

00 = 系统暂停工作，直到电压恢复到监测阈值之上（发生 LVD 事件系统暂停工作时，可通过配置 MISCR3_LVDSTOPEN 位来选择 WDT 是否暂停工作，如果 WDT 继续工作则有可能在系统暂停期间发生 WDT 溢出复位）

01 = 系统发生复位，直到电压恢复到监测阈值之上

10 = 仅记录状态

11 = 仅产生 LVD 中断

注：必须要打开 LDOCR1.bit6=1，LVD 低电压检测模块才能工作。

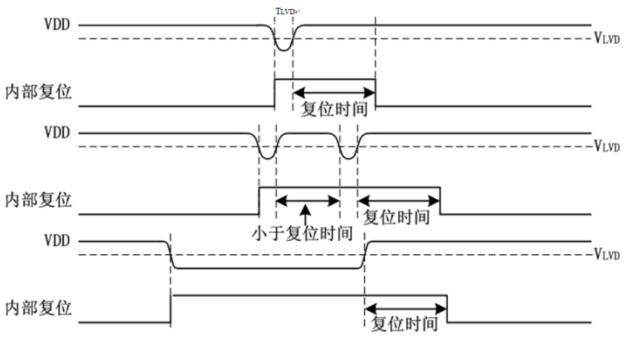


图 5-2、内部低压复位逻辑图

如上图 5-2 中，当低压复位使能时，一旦 VDD 下降到 VLVD 以下，则 MCU 会产生内部复位信号；当内部复位信号持续时间超过 TLVD 时，则芯片将会发生复位（不管 VDD 的变化速率如何，都会发生复位）。如果 VDD 低于 VLVD 的时间少于 TLVD，则芯片不会发生复位。（TLVD 为 FLTLEN[2:0]设置的毛刺滤波宽度）

另外，任何复位(POR 复位、LVD 复位、WDT 溢出复位、软件复位、硬件复位等)都会使芯片保持复位状态，直到 VDD 上升到 VLVD 以上(如上图 5-2)。

MISCR3_LVDSTOPEN：LVD 停止时钟时是否关闭 WDT

寄存器 MISCR3 的 bit0 位控制是否关闭 WDT（0：不关闭；1：关闭），详情请参考“4.2 章节 MISCR3 寄存器”描述。

硬件复位 RST_PAD

CBM73xxF 系列 MCU 支持 IO 引脚外部硬件复位功能，为低电平复位。典型硬件复位电路如下图 5-3 所示

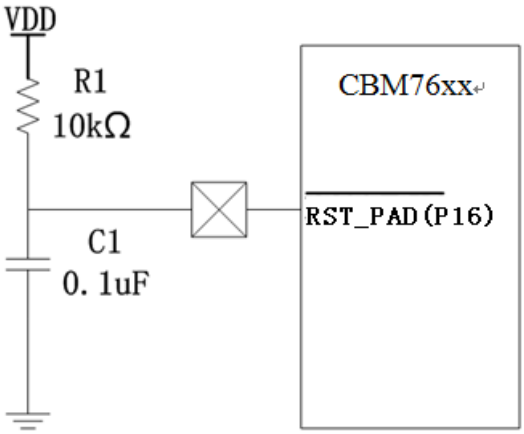


图 5-3、典型硬件复位电路

CBM73xxF 外部硬件复位引脚固定在 P16 端口，且必须设置寄存器 P16IOCFG=0x00(详见下表 7-2)，才能使能 P16 端口的硬件复位功能，同时还须配置 P16 方向寄存器为输入方向（P10E_6 =0x40）；P16IOCFG=0x80（bit7=1）时表示 P16 为普通 IO 口，不具有硬件复位功能；P16IOCFG=其它值，此时 P16 端口可配置为其它复用功能使用，详情请见“寄存器 PxyIOCFG”描述。

软件指令复位

SRSTCR：系统复位标志寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

CBM73xxF 系列 MCU 支持软件指令复位功能，通过依次对寄存器 SRSTCR 写入复位命令，可实现软件复位，如下：

```
SRSTCR = 0x55;
SRSTCR = 0xaa;
SRSTCR = 0x5a;
```

则 MCU 发生软件复位。

5.2、中断

CBM73xxF 系列 MCU 由内核中断和对应各模块中断源构成。MCU 总共有 6 个内核中断，内核中断入口地址分别为 0003H、000BH、0013H、001BH、0023H、002BH；每个内核中断包含多个模块中断源，各模块中断源均可触发内核中断事件；CBM73xxF 系列 MCU 的所有模块中断源如下：

- 外部中断 0、1、2、3、4、5
- 看门狗溢出中断
- 低压监测中断
- ESD 监测中断
- Timer0 中断
- Timer1 中断
- Timer2 中断
- Timer3 中断 0、中断 1
- SensorADC 中断
- ADC 中断
- RTC 中断
- UART1、UART2 中断
- I2C Master 中断
- I2C Slave 中断
- SPI 中断

中断入口地址

CBM73xxF 系列 MCU 的中断入口地址由 16bit 数据构成；其中高 8 位为可偏移地址控制寄存器 VTOR 决定，低 8 位是标准的 8051 的中断入口地址。

VTOR 寄存器上电复位值为 0x00，即复位时中断入口地址和 8051 中断入口地址一致。若用户开发程序的起始地址为 0x0800，则可以将中断入口地址的高 8 位设置到中断入口偏移地址寄存器 VTOR 中，使中断发生时，程序跳转到新的中断入口处。若配置 VTOR = 0x08，则新的中断入口地址，如下表 5-2 所示：

表 5-2、中断向量入口地址

中断向量	中断入口地址(原始)	中断入口地址(加偏移)
INT0	0x0003	0x0803
INT1	0x000b	0x080b
INT2	0x0013	0x0813
INT3	0x001b	0x081b
INT4	0x0023	0x0823
INT5	0x002b	0x082b

中断优先级

CBM73xxF 系列芯片内核中断具有两级优先级可控（高优先级和低优先级），由 IP 寄存器对应位控制 6 个内核中断的优先级，详细请参考“3.2 章节 IP 寄存器”介绍（每个内核中断包含多个外设模块中断，属于同一内核中断的外设中断处于同一优先级。）；中断优先级规则如下：

1、高优先级的中断可以打断低优先级中断的处理；反之，不行。

2、同一级的中断，后产生的中断不能打断目前正在处理的中断。

3、同一级的中断也存在优先级；若同一级的中断同时产生，高优先级的先处理；优先级：IP1_bit[1:0]>IP1_bit[3:2]>IP1_bit[5:4]>IP1_bit[7:6]>IP2_bit[1:0]>IP2_bit[3:2]。

中断寄存器的定义

如下表 5-3 所示，IE 为内核中断使能寄存器，控制 6 个内核中断开关及总中断开关；IP 为内核中断优先级控制寄存器，对应 bit 分别控制 6 个内核中断的优先级顺序；IRQnER 为模块中断使能寄存器，控制对应内核中断 n 的各模块中断源的中断开关；IRQnSR 为外设模块中断状态寄存器，标志对应内核中断 n 的各外设模块中断源的中断状态。

表 5-3、中断寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
IE	0xA8	SFR	内核中断使能寄存器	0000 0000
IP1	0xB8	SFR	内核中断优先级控制寄存器	0000 0000
IP2	0xC8	SFR	内核中断优先级控制寄存器	0000 0000
IRQ0SR	0xF4FF	XRAM	外设模块中断状态寄存器0	0000 0000
IRQ1SR	0xF5FF	XRAM	外设模块中断状态寄存器1	0000 0000
IRQ2SR	0xF6FF	XRAM	外设模块中断状态寄存器2	0000 0000
IRQ3SR	0xF7FF	XRAM	外设模块中断状态寄存器3	0000 0000
IRQ4SR	0xF8FF	XRAM	外设模块中断状态寄存器4	0000 0000
IRQ5SR	0xF9FF	XRAM	外设模块中断状态寄存器5	0000 0000
IRQ0ER	0xFAFF	XRAM	外设模块中断使能寄存器0	0000 0000
IRQ1ER	0xFBFF	XRAM	外设模块中断使能寄存器1	0000 0000
IRQ2ER	0xFCFF	XRAM	外设模块中断使能寄存器2	0000 0000
IRQ3ER	0xFDFF	XRAM	外设模块中断使能寄存器3	0000 0000
IRQ4ER	0xFEFF	XRAM	外设模块中断使能寄存器4	0000 0000
IRQ5ER	0xFFFF	XRAM	外设模块中断使能寄存器5	0000 0000

以上表中寄存器 IE、IP 均不支持按位操作，只能对整个寄存器进行字节的操作；其它寄存器支持按位操作或直接对字节操作。

IE & IP1 & IP2：内核中断使能 & 优先级控制寄存器

请参考“3.2 章节 IE、IP 寄存器”。

IRQ0ER：外设模块中断使能寄存器 0

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	XRAMERRINT	LVDINT	WDTINT	TCM3INT0	XINT0
U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-5 保留未用

bit 4 XRAMERRINT：XRAM 偶校验出错中断使能位

0 = 关闭

1 = 使能

bit 3 LVDINT：LVD 中断使能位

0 = 关闭

1 = 使能

bit 2 WDTINT：WDT 溢出中断使能位

0 = 关闭

1 = 使能

bit 1　　TCM3INT0：Timer3 中断 0 中断使能位

0 = 关闭

1 = 使能

bit 0　　XINT0：外部中断 0 中断使能位

0 = 关闭

1 = 使能

IRQ1ER：外设模块中断使能寄存器 1

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	IRAMERRINT	ESD1INT	SADCINT	TCM3INT1	XINT1
U-x	U-x	U-x	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-5　　保留未用

bit 4　　IRAMERRINT：IRAM 偶校验出错中断使能位

0 = 关闭

1 = 使能

bit 3　　ESD1INT：ESD1INT 中断使能位

0 = 关闭

1 = 使能

bit 2　　SADCINT：SADCINT 中断使能位

0 = 关闭

1 = 使能

bit 1　　TCM3INT1：Timer3 中断 1 中断使能位

0 = 关闭

1 = 使能

bit 0　　XINT1：外部中断 1 中断使能位

0 = 关闭

1 = 使能

IRQ2ER：外设模块中断使能寄存器 2

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	SYSCLKFINT	-	RTCINT	TCM0INT	XINT2
U-x	U-x	U-x	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-5　　保留未用

bit 4　　SYSCLKFINT：系统时钟源出错中断使能位

0 = 关闭

1 = 使能

bit 3　　保留未用

bit 2　　RTCINT：RTCINT 中断使能位

0 = 关闭

1 = 使能

bit 1　　TCM0INT：Timer0 中断使能位
　　0 = 关闭
　　1 = 使能
bit 0　　XINT2：XINT2 中断使能位
　　0 = 关闭
　　1 = 使能

IRQ3ER：外设模块中断使能寄存器 3

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	UARTINT	TCM1INT	XINT3
U-x	U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-3 保留未用
bit 2　　UARTINT：uart1 模块中断使能位
　　0 = 关闭
　　1 = 使能
bit 1　　TCM1INT：Timer1 中断使能位
　　0 = 关闭
　　1 = 使能
bit 0　　XINT3：外部中断 3 中断使能位
　　0 = 关闭
　　1 = 使能

IRQ4ER：外设模块中断使能寄存器 4

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	UART2INT	ADCINT	TCM2INT	XINT4
U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-4　　保留未用
bit 3　　UART2INT：UART2 模块中断使能位
　　0 = 关闭
　　1 = 使能
bit 2　　ADCINT：ADC 中断使能位
　　0 = 关闭
　　1 = 使能
bit 1　　TCM2INT：Timer2 中断使能位
　　0 = 关闭
　　1 = 使能
bit 0　　XINT4：外部中断 4 中断使能位
　　0 = 关闭
　　1 = 使能

IRQ5ER：外设模块中断使能寄存器 5

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	I2CMINT	SPIINT	I2CSINT	XINT5
U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit7-4 保留未用

bit 3 I2CMINT：I2C Master 中断使能位

0 = 关闭

1 = 使能

bit 2 SPIINT：SPI 中断使能位

0 = 关闭

1 = 使能

bit 1 I2CSINT：I2C Slaver 中断使能位

0 = 关闭

1 = 使能

bit 0 XINT5：外部中断 5 中断使能位

0 = 关闭

1 = 使能

IRQ0SR：外设模块中断状态寄存器 0

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	XRAMERRINT	LVDINT	WDTINT	TCM3INT0	XINT0
U-x	U-x	U-x	R-0	R-0	R-0	R-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-5 保留未用

bit 4 XRAMERRINT：XRAM 偶校验出错中断标志位

0 = 无

1 = XRAM 偶校验出错

bit 3 LVDINT：LVD 中断标志位

0 = 无

1 = 已产生 LVD 中断告警

bit 2 WDTINT：WDT 溢出中断标志位

0 = 无

1 = WDT 已溢出

bit 1 TCM3INT0：Timer3 中断 0 中断标志位

0 = 无

1 = 已产生 Timer3 中断 0 中断条件

bit 0 XINT0：外部中断 0 中断标志位

0 = 无

1 = 已产生外部中断 0 中断条件，需软件写 0 清 0

IRQ1SR：外设模块中断状态寄存器 1

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	IRAMERRINT	ESD1INT	SADCINT	TCM3INT1	XINT1
U-x	U-x	U-x	R-0	R-1	R-0	R-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-5 保留未用

bit 4 IRAMERRINT：IRAM 偶校验出错中断标志位

0 = 无

1 = IRAM 偶校验出错

bit 3 ESD1INT：ESD1INT 中断标志位

0 = 无

1 = 已产生 ESD 中断条件

bit 2 SADCINT：SADCINT 中断标志位

0 = 无

1 = 已完成 SADC 采样

bit 1 TCM3INT1：Timer3 中断 1 中断标志位

0 = 无

1 = 已产生 Timer3 中断 1 的中断条件

bit 0 XINT1：外部中断 1 中断标志位

0 = 无

1 = 已产生外部中断 1 的中断条件，需软件写 0 清 0

IRQ2SR：外设模块中断状态寄存器 2

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	SYSCLKFINT	-	RTCINT	TCM0INT	XINT2
U-x	U-x	U-x	R-0	U-x	R-0	R-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit7-5 保留未用

bit 4 SYSCLKFINT：系统时钟源出错中断标志位

0 = 无

1 = 外部晶振出错，可触发中断

bit 3 保留未用

bit 2 RTCINT：RTCINT 中断标志位

0 = 无

1 = 已产生 RTC 中断条件

bit 1 TCM0INT：Timer0 中断标志位

0 = 无

1 = 已产生 Timer0 的中断条件

bit 0 XINT2：XINT2 中断标志位

0 = 无

1 = 已产生外部中断 2 的中断条件，需软件写 0 清 0

IRQ3SR：外设模块中断状态寄存器 3

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	UARTINT	TCM1INT	XINT3
U-x	U-x	U-x	U-x	U-x	R-0	R-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-3 保留未用

bit 2 UARTINT：uart1 中断标志位

0 = 无

1 = Uart1 已发送完成或接收完成数据

bit 1 TCM1INT：Timer1 中断标志位

0 = 无

1 =已产生 Timer1 中断条件

bit 0 XINT3：外部中断 3 中断标志位

0 = 无

1 = 已产生外部中断 3 的中断条件，需软件写 0 清 0

IRQ4SR：外设模块中断状态寄存器 4

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	UART2INT	ADCINT	TCM2INT	XINT4
U-x	U-x	U-x	U-x	R-0	R-0	R-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-4 保留未用

bit 3 UART2INT：UART2 模块中断标志位

0 = 无

1 = UART2 已发送完成或接收完成数据

bit 2 ADCINT：ADC 中断标志位

0 = 无

1 = ADC 采样转换完成

bit 1 TCM2INT：Timer2 中断标志位

0 = 无

1 = 已产生 Timer2 中断的条件

bit 0 XINT4：外部中断 4 中断标志位

0 = 无

1 = 已产生外部中断 4 的中断条件，需软件写 0 清 0

IRQ5SR：外设模块中断状态寄存器 5

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	I2CMINT	SPIINT	I2CSINT	XINT5
U-x	U-x	U-x	U-x	R-0	R-0	R-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-4　保留未用

bit 3　I2CMINT：I2C Master 中断标志位

0 = 无

1 = 已产生 I2C Master 中断条件

bit 2　SPIINT：SPI 中断状态位

0 = 无

1 = 已产生 SPI 中断条件

bit 1　I2CSINT：I2C Slaver 中断标志位

0 = 无

1 = 已产生 I2C Slaver 中断条件

bit 0　XINT5：外部中断 5 中断标志位

0 = 无

1 = 已产生外部中断 5 中断条件，需软件写 0 清 0

中断源汇总表

表 5-4、外设模块中断源汇总表

中断总使能位	内核中断使能位	内核优先级控制位	内核中断	中断源	相关寄存器
EA	EX0	IP1_bit[1:0]	内核中断0	XRAM 偶校验出错中断	IRQ0SR_XRAMERRINT，IRQ0ER_XRAMERRINT
				低电压检测中断	IRQ0SR_LVDINT，IRQ0ER_LVDINT
				看门狗中断	IRQ0SR_WDTINT，IRQ0ER_WDTINT
				定时器3中断0	IRQ0SR_TCM3INT0，IRQ0ER_TCM3INT0
				外部中断0	IRQ0SR_XINT0，IRQ0ER_XINT0
	EX1	IP1_bit[3:2]	内核中断1	IRAM 偶校验出错中断	IRQ0SR_IRAMERRINT，IRQ0ER_IRAMERRINT
				ESD 中断	IRQ1SR_ESD1INT，IRQ1ER_ESD1INT
				SensorADC 中断	IRQ1SR_SADCINT，IRQ1ER_SADCINT
				定时器3中断1	IRQ1SR_TCM3INT1，IRQ1ER_TCM3INT1
				外部中断1	IRQ1SR_XINT1，IRQ1ER_XINT1
	EX2	IP1_bit[5:4]	内核中断2	外部晶振出错中断	IRQ2SR_SYSCLKFINT，IRQ2ER_SYSCLKFINT
				RTC 中断	IRQ2SR_RTCINT，IRQ2ER_RTCINT
				定时器0中断	IRQ2SR_TCM0INT，IRQ2ER_TCM0INT
				外部中断2	IRQ2SR_XINT2，IRQ2ER_XINT2
	EX3	IP1_bit[7:6]	内核中断3		
				串口1中断	IRQ3SR_UARTINT，IRQ3ER_UARTINT
				定时器1中断	IRQ3SR_TCM1INT，IRQ3ER_TCM1INT
				外部中断3	IRQ3SR_XINT3，IRQ3ER_XINT3
	EX4	IP2_bit[1:0]	内核中断4	串口2中断	IRQ3SR_UART2INT，IRQ3ER_UART2INT
				通用 ADC 中断	IRQ4SR_ADCINT，IRQ4ER_ADCINT
				定时器2中断	IRQ4SR_TCM2INT，IRQ4ER_TCM2INT
				外部中断4	IRQ4SR_XINT4，IRQ4ER_XINT4
	EX5	IP2_bit[3:2]	内核中断5	I2C Master 中断	IRQ5SR_I2CMINT，IRQ5ER_I2CMINT
				SPI 中断	IRQ5SR_SPIINT，IRQ5ER_SPIINT
				I2C Slave 中断	IRQ5SR_I2CSINT，IRQ5ER_I2CSINT
				外部中断5	IRQ5SR_XINT5，IRQ5ER_XINT5

中断控制逻辑图

下图描述了内核中断 0 的外设模块从触发中断事件到的中断响应的控制逻辑，其它内核中断也是同样的控制逻辑，图中 n（0~5）表示 6 个内核中断。

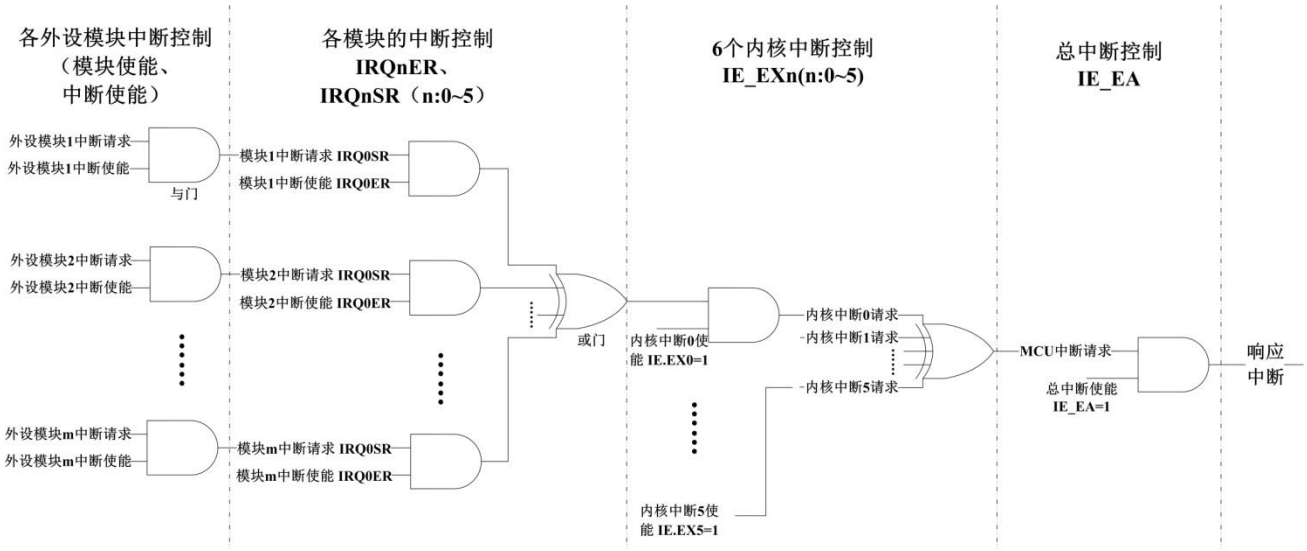


图 5-4、中断控制逻辑图

中断使能步骤

要确保外设模块的中断请求能成功触发 MCU 的中断响应(打断内核执行，跳转到内核中断服务程序入口)，必须进行以下步骤设置：

1、使能对应外设模块，打开对应外设模块的中断使能位；

2、打开该模块对应的“外设模块中断使能寄存器 IRQnER(n=0~5)”中对应中断使能控制位；

3、配置内核中断优先级寄存器 IP1\IP2（IP1_bit[1:0]、IP1_bit[3:2]…IP2_bit[3:2]位对应 6 个内核中断的优先级控制)；

4、打开内核中断使能寄存器 IE 中对应的内核中断使能位（EX0、EX1……EX5)；

5、打开内核总中断使能位(IE 寄存器的 EA 位控制)。

例：使能 UART1 接收数据完成中断如下：

1、将 UIER1 寄存器的 RXIE 位置 1，使能 UART1 接收数据完成中断；

2、将 IRQ3ER 寄存器的 UARTINT1 位置 1，使能 UART1 模块中断；(UART 模块包括发送/接收数据完成中断)

3、配置 IP 寄存器的 IP1_BIT[7:6]位，设置内核中断 3 的中断优先级；(0：低优先级；1：高优先级置)

4、将 IE 寄存器的 EX3 置 1，使能内核中断 3 中断；

5、将 IE 寄存器的 EA 位置 1，使能内核总中断。

5.3、低功耗模式

CBM73xxF 系列 MCU 有两种低功耗的工作模式，分别是休眠模式(SLEEP)和空闲模式(IDLE)。

5.3.1、空闲模式(IDLE)

软件配置 PCON=0x01(配置 PCON 前必须先写 PSEQ，具体请见“3.2 章节 PCON、PSEQ 寄存器”说明)；进入空闲模式后，51 内核、WDT、ADC、I²C Master 停止工作，但是 OSC48M、OSC800K、外部晶振可以继续工作。

在空闲模式下，以下模块会停止工作：

- 51 内核（指令不再执行）
- WDT
- ADC
- I²C Master

用户可以通过以下方法把 MCU 从空闲模式中唤醒：

- Timer0 中断
- Timer1 中断
- Timer2 中断
- Timer3 中断 0
- Timer3 中断 1
- sensorADC 中断
- RTC 中断
- SPI 中断
- I2C Slave 中断
- UART1 中断
- UART2 中断
- 外部中断 0、1、2、3、4、5

如果用户要配置某个模块为空闲模式唤醒源，则只要配置该模块并使能该模块的中断，则当满足该模块中断条件时就能唤醒 MCU。

5.3.2、休眠模式(SLEEP)

软件配置 PCON = 0x02(配置 PCON 前必须先写 PSEQ，具体请见“3.2 章节 PCON、PSEQ 寄存器”说明)；进入休眠模式后，51 内核、所有外设模块、OSC48M 时钟停止工作，但 OSC800K、外部晶振会继续工作。

可以通过下面五种方式将 MCU 从休眠状态唤醒：

- 外部中断 0 唤醒
- 自动计数唤醒
- RTC 定时唤醒
- 特定触摸按键唤醒

休眠唤醒寄存器的定义

表 5-5、休眠唤醒寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
SLPWKCR	0x52FF	XRAM	休眠模式唤醒控制寄存器	0000 0000
SLPWKDR	0x53FF	XRAM	休眠模式唤醒参数寄存器	0000 0000
SLPWKSR	0x5DFF	XRAM	休眠模式唤醒状态寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

SLPWKCR:休眠模式唤醒控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
XINT0EDGESEL	DRSEL2	DRSEL1	DRSEL0	XINT0WKEN	SENSORWKEN	RTCWKEN	AUTOWKEN
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 XINT0EDGESEL：外部中断 0 唤醒沿方式设置
0 = 外部中断 0 下降沿唤醒
1 = 外部中断 0 上降沿唤醒

bit 6-4 DRSEL[6:4]：XRAM 偶校验出错中断使能位
000 = 使能读写 AUTOWKCNT 的低 8 位
001 = 使能读写 AUTOWKCNT 的中间 8 位
010 = 使能读写 AUTOWKCNT 的高 8 位
011 = 保留
100 = 使能读写 SADCMIN 的低 8 位
101 = 使能读写 SADCMIN 的高 8 位
110 = 使能读写 SADCMAX 的低 8 位
111 = 使能读写 SADCMAX 的高 8 位

bit 3 XINT0WKEN：外部中断 0 唤醒使能位
0 = 关闭
1 = 使能

bit 2 SENSORWKEN：触摸按键唤醒使能位
0 = 关闭
1 = 使能

bit 1 RTCWKEN：RTC 唤醒使能位
0 = 关闭
1 = 使能

bit 0 AUTOWKEN：自动计数唤醒使能位
0 = 关闭
1 = 使能

SLPWKDR:休眠模式唤醒参数寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 休眠唤醒参数配置寄存器

SLPWKDR 寄存器用于设置两类 SLEEP 唤醒时的参数，需与 SLPWKCR 寄存器的 DRSEL<2:0>位配合使用。详情请见“计数自动唤醒”、“触摸按键唤醒”描述。

SLPWKSR:休眠模式唤醒状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	XINTOWKS	SENSORWKS	RTCWKS	AUTOWKS
U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 4 保留未用

bit 3 XINTOWKEN：外部中断 0 唤醒休眠模式标志位

0 = 无

1 = 休眠模式唤醒源为外部中断 0 唤醒

bit 2 SENSORWKEN：触摸按键唤醒休眠模式标志位

0 = 无

1 = 休眠模式唤醒源为触摸按键唤醒

bit 1 RTCWKEN：RTC 唤醒休眠模式标志位

0 = 无

1 = 休眠模式唤醒源为 RTC 唤醒

bit 0 AUTOWKEN：自动计数唤醒休眠模式标志位

0 = 无

1 = 休眠模式唤醒源为自动计数唤醒

外部中断 0 唤醒

外部中断 0 唤醒操作步骤：

1、端口配置：

通过 PxyIOCFG 寄存器将 I/O 配置成外部中断 0 功能时，会自动将引脚方向配置为输入，不需要配置对应的 PnOE 寄存器选择外部中断 0 引脚方向为输入。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)

2、外部中断 0 唤醒的有效沿选择：

SLPWKCR 寄存器的 EDGESEL 位可以选择外部中断 0 唤醒的有效沿，EDGESEL=0 为下降沿唤醒，EDGESEL=1 为上升沿唤醒。

3、使能外部中断 0 唤醒：

将 SLPWKCR 寄存器的 XINTOWKEN 位置 1，使能外部中断 0 唤醒。当外部中断 0 输入引脚有有效沿输入时，就会唤醒 CPU。

计数自动唤醒

计数自动唤醒是以 OSC800K 时钟为时基，每个时钟计数值加 1，当计数值与 AUTOWKCNT 匹配时就会唤醒 CPU。

计数自动唤醒操作步骤：

1、配置计数自动唤醒的等待周期数

计数自动唤醒等待周期数 AUTOWKCNT 是一个 24 位的数。通过 SLPWKCR 寄存器的 DRSEL[2:0]位和 SLPWKDR 寄存器的配合使用，可以配置定时自动唤醒等待周期数 ATWKCNT。

例如：若要将 ATWKCNT 设置成 0x123456，需执行如下 3 个操作(不分先后)：

1)、设置 SLPWKCR_DRSEL=000，写 SLPWKDR=0x56

2)、设置 SLPWKCR_DRSEL=001，写 SLPWKDR=0x34

3)、设置 SLPWKCR_DRSEL=010，写 SLPWKDR=0x12

若固件要读取 AUTOWKCNT 的数值，需执行如下 4 个操作：

1)、设置 SLPWKCR_DRSEL=000，读 SLPWKDR；

2)、设置 SLPWKCR_DRSEL=001，读 SLPWKDR；

3)、设置 SLPWKCR_DRSEL=010，读 SLPWKDR；

4)、将前 3 次读到的数据拼接为 24 位数

2、使能计数自动唤醒

将 SLPWKCR 寄存器的 AUTOWKEN 位置 1，使能定时自动唤醒。

RTC 唤醒

RTC 定时唤醒操作步骤：

1、配置 RTC 模块

使能 RTC 功能，具体配置请参考“14、RTC 模块”；

2、使能 RTC 定时唤醒

将 SLPWKCR 寄存器的 RTCWKEN 位置 1，使能 RTC 定时唤醒。当 RTC 模块时间寄存器与闹钟寄存器的值匹配时，就会唤醒 CPU。

触摸按键唤醒

特定触摸按键唤醒操作步骤：

1、配置 sensorADC 模块

配置特定触摸按键唤醒操作步骤基本上与正常工作模式下配置 sensorADC 的操作步骤一致。只有将禁止休眠唤醒相关配置要改为使能休眠唤醒相关配置。也就是将 SWKCR 寄存器的 CLKSEL 位置 1，选择内部低速 OSC80OK 经过 CKDIVCR 寄存器的 SLOWDIV[1:0]位分频后再 4 分频的时钟作为 sensorADC 的时钟。

配置 SWKCR 寄存器的 CMPRSET[2:0]位，设置采样值连续落入阈值内的次数。

2、按键电容采样值范围设置

按键电容采样值是一个 16 位的数，通过 SLPWKCR 寄存器的 DRSEL[2:0]位和 SLPWKDR 寄存器配合使用来配置按键电容采样值的范围。

例如：若要将电容采样值的范围设置成 0x1234~ 0x5678，需执行如下 4 个操作(不分先后)：

1)、设置 SLPWKCR_DRSEL=100，写 SLPWKDR=0x34

2)、设置 SLPWKCR_DRSEL=101，写 SLPWKDR=0x12

3)、设置 SLPWKCR_DRSEL=110，写 SLPWKDR=0x78

4)、设置 SLPWKCR_DRSEL=111，写 SLPWKDR=0x56

若固件要读取电容采样值的范围，需执行如下 5 个操作：

1)、设置 SLPWKCR_DRSEL=100，读 SLPWKDR

2)、设置 SLPWKCR_DRSEL=101，读 SLPWKDR

3)、设置 SLPWKCR_DRSEL=110，读 SLPWKDR

4)、设置 SLPWKCR_DRSEL=111，读 SLPWKDR

5)将前 4 次读到的数据拼接为 2 个 16 位数 SADCMIN、SADCMAX。

3、使能特定触摸按键唤醒

将 SLPWKCR 寄存器的 SENSORWKEN 位置 1，使能特定触摸按键唤醒。

使能特定触摸按键唤醒后，sensorADC 模块会不断对设定的 sensor 通道进行采样，当 sensorADC 采样值连续 N 次(由 SWKCR 寄存器的 CMPRSET[2:0]位配置)落入到预先设置的阈值范围内，就会唤醒 CPU。

5.4、ESD 静电防护

CBM73xxF 系列 MCU 内部带有 ESD 检测电路，当发生 ESD 事件时，MCU 会采取特殊处理措施，增强芯片的抗 ESD 性能。

ESD 寄存器的定义

表 5-6、ESD 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
MISCR3	0x59FF	XRAM	系统控制寄存器3	0000 0000
ESDCR	0x57FF	XRAM	ESD 控制寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

如上表 5-6 中，MISCR3 寄存器的 bit3（MISCR3.ESD1EN）为 ESD 使能控制位；MISCR3 寄存器的 bit2（MISCR3.ESD1SR）为 ESD 中断标志位，详情请见“4.2 章节 MISCR3 寄存器”；ESDCR 控制寄存器可设置 ESD 滤波级数、ESD 警告级别等；详情请见如下：

ESDCR：ESD 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CLKSTOPEN	CLRCNT2	CLRCNT1	CLRCNT0	VOUTSEL1	VOUTSEL0	RES1	RES0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 CLKSTOPEN：当 ESD 事件发生时，是否停止总时钟，指令不再运行

0 = 不停止

1 = 停止总时钟，系统指令不再运行；待 ESD 信号撤销后，时钟自动恢复

bit 6-4 CLRCNT[6:4]：当 ESD 信号滤波周期，滤波时间 = （CNT+1）*T800K

000 =1*T800K

001 = 2*T800K

002 =3*T800K

……

110 = 7*T800K

111 = 8*T800K

bit 3-2 VOUTSEL[3:2]：ESD 告警级别设置

bit 1-0 RES[1:0]：ESD 检测电路对地电阻配置位

00 =无

01 =2K

10 = 1K

11 = 500

注：

1、如果使能 ESD 发生时停止时钟的功能，当 ESD 事件发生后，系统指令停止运行但 WDT 会继续工作，此时需要注意最好满足：WDT 清狗时间<= 1/2*WDT 溢出时间。

5.5、RAM 区偶校验

RAM 区分 IRAM(256 Bytes)和 XRAM(1536 Bytes)，主要用于定义变量、分配缓存等功能，因此数据的正确性至关重要；当然 RAM 区数据本身出错是不太可能发生的，除非一些极端恶劣使用环境下的强干扰（电机引起强浪涌等）。因此，从硬件系统稳定性、可靠性考虑，CBM73xxF 系列 MCU 增加了对 IRAM、XRAM 区读写数据自动偶校验功能，可以软件选择开

启或关闭。

RAMCR：RAM 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	IRAMERRSR	XRAMERRSR	-	-	IRAMEN	XRAMEN
U-x	U-x	R/W-0	R/W-0	U-x	U-x	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 保留未用

bit 5 IRAMERRSR：IRAM 偶校验标志位

0 = 无

1 = IRAM 偶校验出错，需软件写 1 清 0

bit 4 XRAMERRSR：XRAM 偶校验标志位

0 = 无

1 = XRAM 偶校验出错，需软件写 1 清 0

bit 3-2 保留未用

bit 1 IRAMEN：IRAM 偶校验使能控制位

0 = 不使能

1 = 使能 IRAM 偶校验功能

bit 0 XRAMEN：XRAM 偶校验使能控制位

0 = 不使能

1 = 使能 XRAM 偶校验功能

注：

1、由于偶校验功能加了 1bit 校验位，所以 IRAM 实际大小为 256*9bit；XRAM 实际大小为 1536*9bit；最后 1bit 固定为偶校验位，不能作数据位使用。

2、如果使能 RAM 偶校验功能，则 MCU 上电时必须先初始化 RAM 区，因为上电后 RAM 是未知态，读操作可能会导致错误的偶校验结果；

RAM 偶校验的使能步骤如下

1>、上电初始化整个 IRAM、XRAM 区（默认 startup 文件中已配置为上电自动初始化全 RAM 区；禁止改写 startup 文件）；

2>、使能对应的 RAM 偶校验功能（RAMCR_IRAMEN=1 或 RAMCR_XRAMEN=1）；

3>、使能对应的 RAM 的中断功能（IRQ0ER_IRAMERRINT=1 或 IRQ0ER_XRAMERRINT=1）。

5.6、CRC16-CCITT 校验

CBM73xxF 系列 MCU 采用标准的 CRC16-CCITT 校验算法，支持对 main flash code 进行上电时的 CRC16 校验和运行中的动态 CRC16 校验；上电 CRC16 校验功能可以通过上位机编程软件进行配置使能（详情参考推广资料里的《CBM7X 系列烧录器使用手册_V1.1.pdf》）；而动态 CRC16 校验可以由用户软件控制使能与否。详情操作请参考“20.2、MAIN 区 CRC 校验”，校验算法请参考“20.2.3 和 20.2.4”。

CRC16 校验功能有助于 class B 认证。

5.7、编程加密

CBM73xxF 系列 MCU 支持对用户代码进行编程加密保护。编程秘钥由 2 字节组成，用户可以通过 PC 上位机（CBM73xxF 编程器）使能和配置编程秘钥并通过烧录器编程代码时写入（详情参考推广资料里的《CBM7X 系列烧录器使用手册_V1.1.pdf》）；用户使能编程加密后，芯片会开启外部对 MCU flash 的读操作保护等保护措施。

5.8、在线 Debug &烧录

CBM73xxF 系列 MCU 支持通过 UART 口实现在线 Debug 与烧录功能；在线 Debug 与烧录的硬件接口由如下 4 线组成：

- 电源线(VDD)
- 接地线(VSS)
- 数据发送线(TXD)
- 数据接收线(RXD)

通过以上接口，用户可方便实现以下功能：

1、使用 CBM73xxF 调试器与 CBM73xxF MCU 连接，通过 keil 软件实现在线调试、在线下载功能；满足产品开发、维护阶段调试需求；

2、使用 CBM73xxF 调试器与 CBM73xxF MCU 连接，通过 PC 上位机软件（CBM73xxF 编程器）可以在线烧录程序，满足小批量生产烧录需求；

3、使用 CBM73xxF 脱机烧录器，可以对 CBM73xxF MCU 进行离线烧录程序，满足产线的批量生产需求。

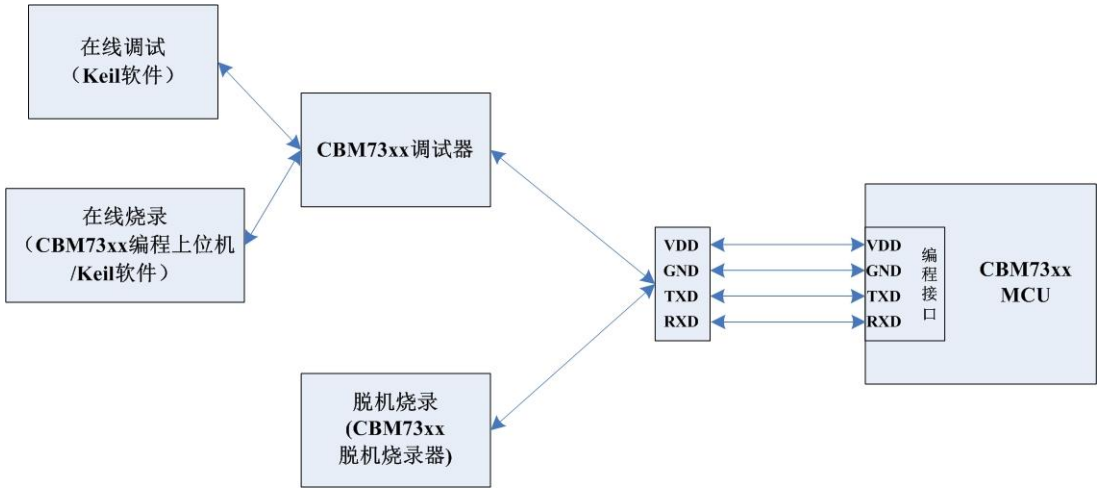


图 5-5、在线 debug、烧录示意图

5.9、在线更新升级用户程序

CBM73xxF 系列 MCU 支持在 MCU 工作过程中，由“更新上位机 app”通过与 MCU 建立的通迅方式（uart/iic/spi 等），将“待更新的用户程序”发送给 MCU，由 MCU 当前运行的用户程序来完成更新升级。芯片提供的“更新上位机 app”为“CBM_UARTUpdataTool.exe”,通过 uart 与 MCU 进行数据通迅。用户也可以选择自定义开发自己的 app 工具。



图 5-6、在线更新示意图

5.9.1、如何使能 mcu 的在线更新功能

用“CBM73xx 编程器”对“CBM7x 系列脱机烧录器”进行离线下载"待更新的用户程序"时,**必须勾选上“CBM73xx 编程器-离线配置参数”的 remap 项**（如下图 5-7），然后通过“CBM7x 系列脱机烧录器”对 mcu 进行烧录，则可以使能 MCU 的在线更新功能。

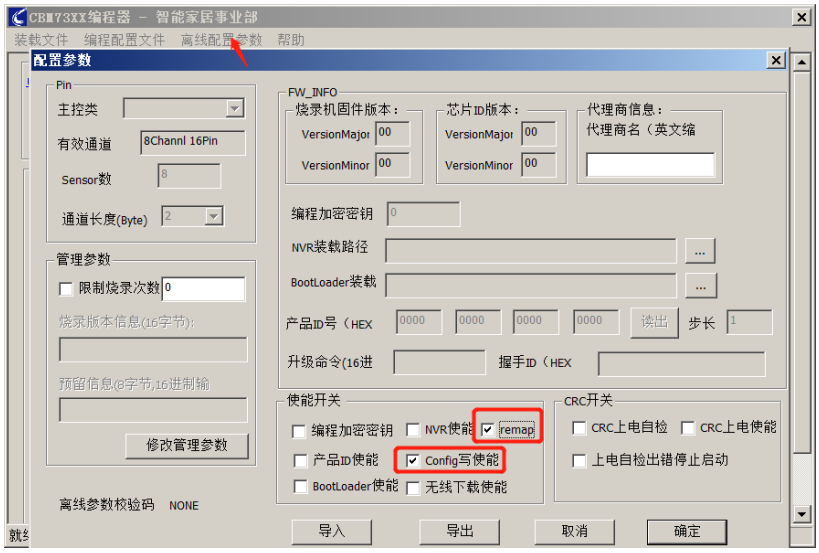


图 5-7、勾选 remap 项使能在线升级功能

5.9.2、使能在线更新功能后 main flash 区块的划分及用户程序大小限制

当使能在线更新功能（5.9.1 所述）后，main flash 地址被分为如下 3 个可被用户程序访问的功能区域地址块：

1>、31.5k 当前运行的用户程序的区块：访问地址为 0000H~7DFFH（一般不建议用户程序去访问操作此区）；

2>、31.5k 可用来更新用户程序代码的区块：访问地址为 8000H~FDFFH，用户程序可以将"待更新的用户程序"写到此区；

3>、0.5k 在线更新功能的相关控制字区块：访问地址为 FE00H~FFFFH，在用户程序成功更新完"待更新的用户程序"后，用户程序还需要更改此区的相关控制字的内容。

因此，**当使能在线更新功能后，用户程序和“待更新的用户程序”的大小都不能超过 31.5k。并且以上 3 个区块都可以由当前运行的用户程序通过 EFC 模块来对其进行 read/write/erase 的访问操作，操作函数详见“EFC 模块-20.1 章节”。**

5.9.3、在线更新功能相关控制字的定义

flash main 区 FE00H~FFFFH 上的地址 FE00H~FE09H 的内容用来作为在线更新功能的相关控制字，此区的内容均是在**更新成功后由用户程序写入更改**，具体定义如下：

1>、FE00H 地址上的内容：CRC16 校验控制字（可选写）

表示 MCU 在上电时是否使能对应块 crc16 校验，对应 bit=0 表示不使能该块的 crc16 校验，=1 表示使能。

bit[7:4]必须写全 0 即 b0000；

bit[3:0]的 bit0~bit3 分别对应第 1~4 块 main flash 的 crc16 校验使能控制，第 1~3 块大小均为 8k，第 4 块大小为 7.5k。

2>、FE01H 地址上的内容：更新状态控制字（必须改写）

bit[7:2]表示在线更新是否成功，**bit[7:2]= b101010 表示更新成功**，bit[7:2]="其它值"表示更新失败；

bit[1:0]表示在线更新成功的奇偶次数，**bit[1:0]=b00 表示偶数次，bit[1:0]=b01 表示奇数次，禁止写入"其它值"；**（5.9.1 使能在线更新功能后，bit[1:0]的默认值为 bit[1:0]=b00）。

因此，对 bit[1:0]的操作应该如下：先读出当前的 bit[1:0]值，对 bit[1:0]进行判断,然后再根据当前值来改写。

例如读出当前 bit[1:0]是 b00 则改写成 bit[1:0]=b01；读出当前 bit[1:0]是 b01，则改写成 bit[1:0]=b00。

3>、FE02H~FE09H 地址上内容：存放“待更新的用户程序”的 crc16 码值，crc 算法参考“20.2.3 和 20.2.4”（可选写）

{FE03H,FE02H}存放第 1 块（8k 大小：0000H~1FFFH）的 crc16 码值，crc16 码值的高 8 位写到 FE03H；

{ FE05H,FE04H }存放第 2 块（8k 大小：2000H~3FFFH）的 crc16 码值；

{ FE07H,FE06H }存放第 3 块（8k 大小：4000H~5FFFH）的 crc16 码值；

{ FE09H,FE08H }存放第 4 块（7.5k 大小：6000H~7DFFH）的 crc16 码值。

5.9.4、对"待更新的用户程序"的两种校验方式

1>、由用户自定义校验算法来校验

即用户程序在接收到"更新上位机"发来的"待更新的用户程序"数据包时，由用户程序根据自定义的校验算法来进行校验处理，校验成功则进行编程，校验失败则进行通迅校验失败处理等；所有数据代码更新完成后可以通过 MCU 提供的软件复位命令进行复位重启。

2>、利用 MCU 提供的 CRC16-CCITT 校验机制进行校验，校验算法参考“20. 2. 3 和 20.2.4”

首先，在进行脱机烧录器烧录程序时，需要按如下图 5-8 中所示勾选上**“CBM73xx 编程器-离线配置参数”**。如下图 5-8 的勾选项：表示 MCU 使能上电启动时的 CRC16 校验，且如果校验出错则停止启动，以避免运行错误的程序引发错误动作，如果校验正确则正常启动并开始执行更新后的用户程序。（详情参考推广资料里的《CBM7X 系列烧录器使用手册_V1.1.pdf》）

其次，是用户程序的处理：即在更新成功后，用户程序要使能“5.9.3、在线更新功能相关控制字”的 crc16 校验控制字（FE00H）的对应内容，同时还要把“待更新的用户程序”的 crc16 码值写入到 FE02H~FE09H 地址（crc16 码值是由“更新上位机 app”计算并发送给 MCU），然后通过 MCU 提供的软件复位命令进行复位。

通过以上操作，则在复位重启时，MCU 会对更新后的程序进行 crc16 上电校验。

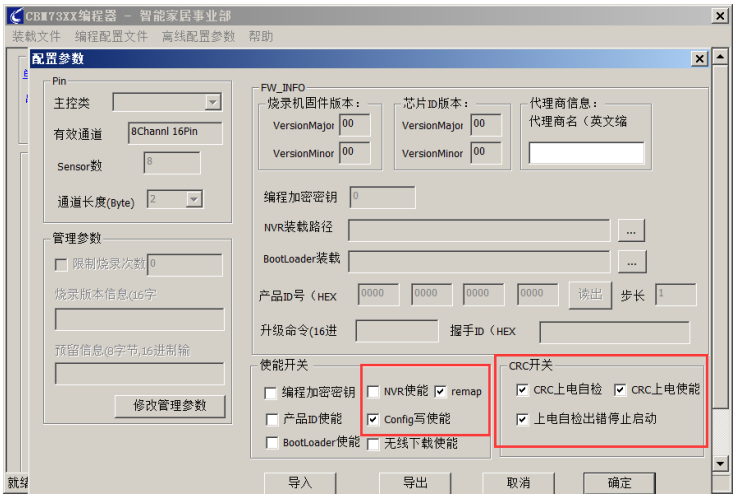


图 5-8、使能在线升级功能时 crc16 校验方式的勾选项

5.9.5、在线更新的参考步骤（可参考推广例程）

第一步：使能 MCU 的在线更新功能，即按 5.9.1 所述操作；

第二步：使能 MCU 的上电 crc16 校验功能，并且一旦校验出错则停止启动，即按 5.9.1 所述操作，但是需要将**“CBM73xx 编程器-离线配置参数”按图 5-8 进行勾选；**（根据需要选择此步骤，如果不需要开启上电 crc 校验则可以跳过第二步）

第三步：“更新上位机 app”发起更新命令，开始执行更新；

第四步：MCU 收到“更新上位机 app”发来的更新命令，校验数据帧正确，则擦除 main flash 的待编程区块(31.5k 可用来更新用户程序代码的区块 8000H~FDFFH)，擦除成功并应答“更新上位机 app”；

第五步：“更新上位机 app”将“待更新的用户程序”拆成多包数据依次发送给 mcu，由 mcu 用户程序进行编程写入（写入地址为 8000H~FDFFH）；

第六步：“更新上位机 app”从 mcu 读出“第五步”写入的“待更新用户程序”的校验结果，校验成功则执行“第七步”；

第七步：用户程序更改在线更新功能相关控制字内容（如 5.9.3 中定义）。其中 FE00H、FE02H~FE09H 的内容是根据第二步来决定是否需要写入，CRC16 码值的校验算法参考“20. 2. 3 和 20.2.4”，并只有执行了第二步时，则需要由“更新上位机 app”计算得出并发送给 MCU 写入更改，否则不需要更改；FE01H 的内容参考 5.9.3 来操作（**FE01H 的内容必须要改写**）。

第八步：用户程序执行软件复位指令，参考“5.1、软件指令复位”，重启 mcu。

第九步：如果使能了第二步的 crc16 校验功能，则 MCU 重启时会对更新程序进行 crc16 校验，校验失败则会停止启动，校验成功正常启动。

6、 外部中断

CBM73xxF 系列芯片最多支持 6 路外部 IO 口中断，通过 XINTxxTRGR 寄存器可以配置各个外部中断的触发方式，包括高电平触发、低电平触发、上升沿触发、下降沿触发、双边沿触发、不触发共 6 种方式。

同时，为避免外部干扰，MCU 片上硬件电路可配置对外部信号进行过滤。即，外部 pad 上的电平需持续某个长度的时间，才会认为是正确的电平。所以，即使外部 pad 上发生真实的电平变化，也会滞后一段时间进入中断服务程序。若持续时间小于滤波时间，则会被认为是干扰信号而被忽略。即使是边沿触发，亦是处理过滤后的边沿情况(见图 6-1)。信号过滤的时间长度由 IFLTCNTR 寄存器设置，默认是过滤 16 个系统时钟。

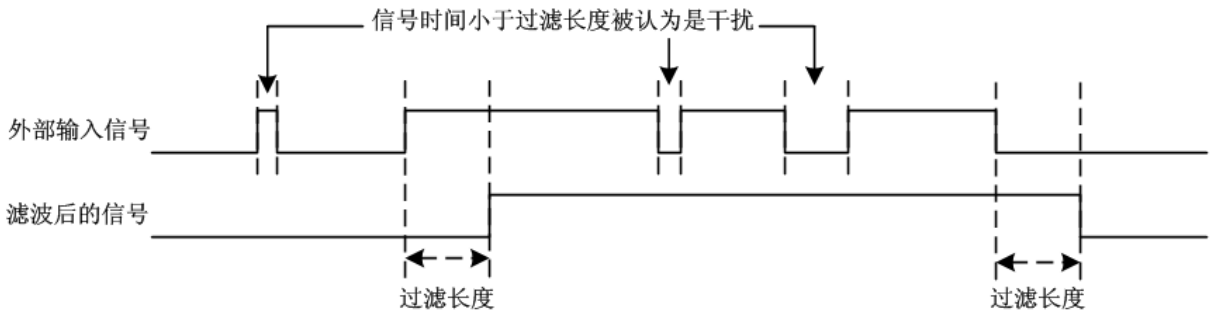


图 6-1、外部中断信号滤波处理

6.1、外部中断寄存器的定义

表 6-1、外部中断寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
IFLTCNTR	0xF0FF	XRAM	外部中断信号过滤长度配置寄存器	0000 1111
XINT01TRGR	0xF1FF	XRAM	外部中断0、1触发方式配置寄存器	0000 0000
XINT23TRGR	0xF2FF	XRAM	外部中断2、3触发方式配置寄存器	0000 0000
XINT45TRGR	0xF3FF	XRAM	外部中断4、5触发方式配置寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

IFLTCNTR：外部中断信号滤波时间配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 IFLTCNT[7:0]：外部中断信号过滤长度配置位

0000 0000 = 1 * Tsys

0000 0001 = 2 * Tsys

0000 0010 = 3 * Tsys

0000 0011 = 4 * Tsys

0000 0100 = 5 * Tsys

……

1111 1110 = 255 * Tsys

1111 1111 = 256 * Tsys

XINT01TRGR：外部中断 0、1 触发方式配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	XINT1TRGR2	XINT1TRGR1	XINT1TRGR0	-	XINT0TRGR2	XINT0TRGR1	XINT0TRGR0
U-x	R/W-0	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit7 保留未用

bit 6-4 XINT1TRGR[6:4]：外部中断 1 触发方式配置位

000 = 高电平触发，当外部中断 pad 为高电平时，触发中断。

001 = 低电平触发，当外部中断 pad 为低电平时，触发中断。

010 = 不触发

011 = 不触发

100 = 不触发

101 = 上升沿触发

110 = 下降沿触发

111 = 双沿触发(上升沿和下降沿都可以触发中断)

bit3 保留未用

bit 2-0 XINT1TRGR[2:0]：外部中断 0 触发方式配置位

000 = 高电平触发，当外部中断 pad 为高电平时，触发中断。

001 = 低电平触发，当外部中断 pad 为低电平时，触发中断。

010 = 不触发

011 = 不触发

100 = 不触发

101 = 上升沿触发

110 = 下降沿触发

111 = 双沿触发(上升沿和下降沿都可以触发中断)

XINT23TRGR：外部中断 2、3 触发方式配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	XINT3TRGR2	XINT3TRGR1	XINT3TRGR0	-	XINT2TRGR2	XINT2TRGR1	XINT2TRGR0
U-x	R/W-0	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit7 保留未用

bit 6-4 XINT3TRGR[6:4]：外部中断 3 触发方式配置位

000 = 高电平触发，当外部中断 pad 为高电平时，触发中断。

001 = 低电平触发，当外部中断 pad 为低电平时，触发中断。

010 = 不触发

011 = 不触发

100 = 不触发

101 = 上升沿触发

110 = 下降沿触发

111 = 双沿触发(上升沿和下降沿都可以触发中断)

bit3　　保留未用

bit 2-0　　XINT2TRGR[2:0]：外部中断 2 触发方式配置位

000 = 高电平触发，当外部中断 pad 为高电平时，触发中断。

001 = 低电平触发，当外部中断 pad 为低电平时，触发中断。

010 = 不触发

011 = 不触发

100 = 不触发

101 = 上升沿触发

110 = 下降沿触发

111 = 双沿触发(上升沿和下降沿都可以触发中断)

XINT45TRGR：外部中断 4、5 触发方式配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	XINT5TRGR2	XINT5TRGR1	XINT5TRGR0	-	XINT4TRGR2	XINT4TRGR1	XINT4TRGR0
U-x	R/W-0	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit7　　保留未用

bit 6-4　　XINT5TRGR[6:4]：外部中断 5 触发方式配置位

000 = 无效。

001 = 无效。

010 = 不触发

011 = 不触发

100 = 不触发

101 = 上升沿触发

110 = 下降沿触发

111 = 双沿触发(上升沿和下降沿都可以触发中断)

bit3　　保留未用

bit 2-0　　XINT4TRGR[2:0]：外部中断 4 触发方式配置位

000 = 高电平触发，当外部中断 pad 为高电平时，触发中断。

001 = 低电平触发，当外部中断 pad 为低电平时，触发中断。

010 = 不触发

011 = 不触发

100 = 不触发

101 = 上升沿触发

110 = 下降沿触发

111 = 双沿触发(上升沿和下降沿都可以触发中断)

6.2、外部中断使用说明

电平触发中断和边沿触发中断的区别：

1、电平触发就是只要电平变为有效电平，在中断使能的前提下，软件会进入中断服务程序。由于内核的 6 个中断入口都复用了至少 4 个中断源(见 IRQnER 寄存器)，所以在中断服务程序中，软件需查询中断状态寄存器，以判断是由哪个外设模块中断源引发的中断。从触发中断到软件查询中断寄存器状态，需要一定的时间，在这段时间内，触发中断的有效电平需要

持续保持。若查询到之前电平失效，会导致软件查询不到电平中断而无法正确处理。

电平触发的中断需要外部主动清除(将外部中断 pad 上的输入电平置为无效电平)。若外部中断 pad 上的中断有效电平一直持续，内核在处理完中断后，会再次进入中断服务程序，导致反复多次进入中断服务程序。如果不希望再次进入中断，则需要外部在首次进入中断服务程序且退出中断服务程序前主动清除此外部中断电平，或者在进入中断服务程序后软件关闭中断使能位，在下次需要时再使能。

2、边沿触发中断是靠外部中断 pad 的边沿触发的，即只要出现有效的沿变，硬件会记录下来曾经发生过中断请求。例如，配置成上升沿触发，外部 pad 在从低变高后，硬件即会记录下从低变高的事件，外部 pad 可以恢复到低电平，也不影响此次中断处理。但是，这种情况下的中断服务程序中需要由软件清除中断状态，否则处理完此次中断后，会再次进入中断服务程序。

6.3、外部中断配置步骤

1、配置 IO 引脚

首先将 IO 口映射成外部中断引脚，同一个 IO 口可以配置为外部中断 0~5 中的任何一个，即 PxyIOCFG = 0x38~0x3D，但一个外部中断不能同时映射到多个 IO 口上；然后将 IO 方向寄存器配置为输入态。

```
P02IOCFG = 0x38;//配置 P02 外部中断 0
POE_2 = 0x04;//配置 P02 输入方向
```

2、配置该外部中断的滤波时间、触发方式

```
IFLTCNTR = 0x80 ;//配置适当即可
XINT01TRGR_XINT0TRG    = 0x06;//配置外部中断 0 为上升沿触发
```

3、使能该外部中断模块的中断功能

```
IRQ0ER_XINT0 = 0x01;//使外部中断 0 的中断功能
```

4、使能对应内核中断、内核总中断、配置中断优先级

```
EX0 = 1;//使能内核中断 0 的中断功能
IP1_BIT[1:0] = 1; //设置内核中断 0 为高优先级
EA = 1;//使能内核总中断
```

7、 I/O 端口

CBM73xxF 系列 MCU 的所有 IO 口均支持重映射功能，功能重映射特点如下：

- 所有 IO 口都能复位为普通 GPIO 口用；
- 调试串口输出只能是 P00 口，调试串口输入只能是 P01 口；
- P06、P07、P10、P11、P12、P13、P14、P15 这 8 个 IO 口可重映射作为 ADC 通道；
- 除 ADC 通道 IO 口（P06、P07、P10~P15）外，所有 IO 口可重映射作为触摸 IO 口功能；
- IO 口可配置电流档位：6mA、30mA 或 60mA；
- P16 端口可复用为硬件复位 IO 口（详见下表 7-2）；
- P25、P26 可以映射为外部 K 级晶振引脚，其它 IO 口不支持外部 K 级晶振功能；
- 除上述功能只能重映射到指定 IO 外，其它功能可以任意重映射到任意 IO 上，如 UART 接口、SPI 接口、PWM 等等；
- 当某个 IO 口重映射用于硬件复位 IO、AD、SENSOR（触摸）这 3 个功能之一时，需要软件配置该 IO 口的方向为输入模式、上拉使能控制寄存器配置为 0；重映射为其它功能时不需要配置方向寄存器和上拉使能控制寄存器；
- 上电时，P00/P01 默认为 debug 功能，而其它 IO 口默认为输入高阻状态，需要用户对相关 I/O 寄存器进行初始化配置。

7.1、IO 寄存器的定义

表 7-1、IO 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
P0~P4	0x80、0x90~0xC0	SFR	P0~P4口数据寄存器	1111 1111
P00IOCFG	0x1000	XRAM	P0.0口功能控制寄存器	0000 0001
P01IOCFG	0x1001	XRAM	P0.1口功能控制寄存器	0011 0000
P02IOCFG	0x1002	XRAM	P0.2口功能控制寄存器	0100 0000
P03IOCFG	0x1003	XRAM	P0.3口功能控制寄存器	0100 0000
P04IOCFG	0x1004	XRAM	P0.4口功能控制寄存器	0100 0000
P05IOCFG	0x1005	XRAM	P0.5口功能控制寄存器	0100 0000
P06IOCFG	0x1006	XRAM	P0.6口功能控制寄存器	0100 0000
P07IOCFG	0x1007	XRAM	P0.7口功能控制寄存器	0100 0000
P10IOCFG	0x1008	XRAM	P1.0口功能控制寄存器	0100 0000
P11IOCFG	0x1009	XRAM	P1.1口功能控制寄存器	0100 0000
P12IOCFG	0x100A	XRAM	P1.2口功能控制寄存器	0100 0000
P13IOCFG	0x100B	XRAM	P1.3口功能控制寄存器	0100 0000
P14IOCFG	0x100C	XRAM	P1.4口功能控制寄存器	0100 0000
P15IOCFG	0x100D	XRAM	P1.5口功能控制寄存器	0100 0000
P16IOCFG	0x100E	XRAM	P1.6口功能控制寄存器	1100 0000
P17IOCFG	0x100F	XRAM	P1.7口功能控制寄存器	0100 0000
P20IOCFG	0x1010	XRAM	P2.0口功能控制寄存器	0100 0000
P21IOCFG	0x1011	XRAM	P2.1口功能控制寄存器	0100 0000
P22IOCFG	0x1012	XRAM	P2.2口功能控制寄存器	0100 0000
P23IOCFG	0x1013	XRAM	P2.3口功能控制寄存器	0100 0000
P24IOCFG	0x1014	XRAM	P2.4口功能控制寄存器	0100 0000
P25IOCFG	0x1015	XRAM	P2.5口功能控制寄存器	0100 0000
P26IOCFG	0x1016	XRAM	P2.6口功能控制寄存器	0100 0000
P27IOCFG	0x1017	XRAM	P2.7口功能控制寄存器	0100 0000
P30IOCFG	0x1018	XRAM	P3.0口功能控制寄存器	0100 0000

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
P31IOCFG	0x1019	XRAM	P3.1口功能控制寄存器	0100 0000
P32IOCFG	0x101A	XRAM	P3.2口功能控制寄存器	0100 0000
P33IOCFG	0x101B	XRAM	P3.3口功能控制寄存器	0100 0000
P34IOCFG	0x101C	XRAM	P3.4口功能控制寄存器	0100 0000
P35IOCFG	0x101D	XRAM	P3.5口功能控制寄存器	0100 0000
P36IOCFG	0x101E	XRAM	P3.6口功能控制寄存器	0100 0000
P37IOCFG	0x101F	XRAM	P3.7口功能控制寄存器	0100 0000
P40IOCFG	0x1020	XRAM	P4.0口功能控制寄存器	0100 0000
P41IOCFG	0x1021	XRAM	P4.1口功能控制寄存器	0100 0000
P42IOCFG	0x1022	XRAM	P4.2口功能控制寄存器	0100 0000
P43IOCFG	0x1023	XRAM	P4.3口功能控制寄存器	0100 0000
P44IOCFG	0x1024	XRAM	P4.4口功能控制寄存器	0100 0000
P45IOCFG	0x1025	XRAM	P4.5口功能控制寄存器	0100 0000
P46IOCFG	0x1026	XRAM	P4.6口功能控制寄存器	0100 0000
P47IOCFG	0x1027	XRAM	P4.7口功能控制寄存器	0100 0000
P0PU	0x25FF	XRAM	P0端口上拉使能控制寄存器	1111 1111
P1PU	0x26FF	XRAM	P1端口上拉使能控制寄存器	1111 1111
P2PU	0x27FF	XRAM	P2端口上拉使能控制寄存器	1111 1111
P3PU	0x28FF	XRAM	P3端口上拉使能控制寄存器	1111 1111
P4PU	0x29FF	XRAM	P4端口上拉使能控制寄存器	1111 1111
P0OE~P4OE	0x2AFF~0x2EFF	XRAM	P0口 GPIO 方向选择寄存器	1111 1111
P0PURSELRL	0x30FF	XRAM	P0口内部上拉电阻大小选择寄存器	0000 0101
P0PURSELRH	0x31FF	XRAM	P0口内部上拉电阻大小选择寄存器	0000 0000
P1PURSELRL	0x32FF	XRAM	P1口内部上拉电阻大小选择寄存器	0000 0000
P1PURSELRH	0x33FF	XRAM	P1口内部上拉电阻大小选择寄存器	0010 0000
P2PURSELRL	0x34FF	XRAM	P2口内部上拉电阻大小选择寄存器	0000 0000
P2PURSELRH	0x35FF	XRAM	P2口内部上拉电阻大小选择寄存器	0000 0000
P3PURSELRL	0x36FF	XRAM	P3口内部上拉电阻大小选择寄存器	0000 0000
P3PURSELRH	0x37FF	XRAM	P3口内部上拉电阻大小选择寄存器	0000 0000
P4PURSELRL	0x38FF	XRAM	P4口内部上拉电阻大小选择寄存器	0000 0000
P4PURSELRH	0x39FF	XRAM	P4口内部上拉电阻大小选择寄存器	0000 0000
P0OD	0x3AFF	XRAM	P0口开漏输出使能寄存器	0000 0000
P1OD	0x3BFF	XRAM	P1口开漏输出使能寄存器	0000 0000
P2OD	0x3CFF	XRAM	P2口开漏输出使能寄存器	0000 0000
P3OD	0x3DFF	XRAM	P3口开漏输出使能寄存器	0000 0000
P4OD	0x3EFF	XRAM	P4口开漏输出使能寄存器	0000 0000
P0LDRVRL	0x40FF	XRAM	P0口驱动电流选择寄存器	0000 0000
P0LDRVRH	0x41FF	XRAM	P0口驱动电流选择寄存器	0000 0000
P1LDRVRL	0x42FF	XRAM	P1口驱动电流选择寄存器	0000 0000
P1LDRVRH	0x43FF	XRAM	P1口驱动电流选择寄存器	0000 0000
P2LDRVRL	0x44FF	XRAM	P2口驱动电流选择寄存器	0000 0000

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
P2LDRVRH	0x45FF	XRAM	P2口驱动电流选择寄存器	0000 0000
P3LDRVRL	0x46FF	XRAM	P3口驱动电流选择寄存器	0000 0000
P3LDRVRH	0x47FF	XRAM	P3口驱动电流选择寄存器	0000 0000
P4LDRVRL	0x48FF	XRAM	P4口驱动电流选择寄存器	0000 0000
P4LDRVRH	0x49FF	XRAM	P4口驱动电流选择寄存器	0000 0000
P0SMEN	0x4BFF	XRAM	P0口施密特使能控制寄存器	0000 0000
P1SMEN	0x4CFF	XRAM	P1口施密特使能控制寄存器	0100 0000
P2SMEN	0x4DFF	XRAM	P2口施密特使能控制寄存器	0000 0000
P3SMEN	0x4EFF	XRAM	P3口施密特使能控制寄存器	0000 0000
P4SMEN	0x4FFF	XRAM	P4口施密特使能控制寄存器	0000 0000

以上表中寄存器地址 0x1014~0x1027 对应寄存器均不支持按位操作，只能对整个寄存器进行字节的操作；其它寄存器支持按位操作或直接对字节操作。

Pn:数据寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
7	6	5	4	3	2	1	0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 Pn[7:0]：IO 口数据寄存器

0 = 输出低电平

1 = 输出高电平

注:当 Pn 端口为输出方向(POEn=0x00)时,向 Pn 寄存器写值将会输出到对应 IO 端口上;而当 Pn 端口为输入方向(POEn=0xff)时，则读 Pn 寄存器可以得到对应 IO 端口上的电平状态。

PxyIOCFG:功能复用控制寄存器(x:0~4;y:0~7)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 P16 的硬件 RST 功能控制位，此位只对 P16IO 有效，对其它 IO 无效（其它 io 时配置为 0）

0 = 使能 P16 的硬件 RST 功能

1 = 禁止 P16 的硬件 RST 功能

bit 6-0 PxyIOCFG[6:0]：IO 功能重映射控制寄存器（当 P16IOCFG.bit7=0 时则 P16 为 RST； =1 时则 P16 为如下设置功能）

0x00 = GPIO（当 P16IOCFG=0x00 则 P16 为 RST，当 P16IOCFG=0x80 则 P16 为 GPIO）

0x01 = 调试串口输出。P0.0 默认为调试串口输出，其他引脚不可以配置为调试串口输出

0x02 = 定时器 1 的输出或输入

0x03 = 定时器 3 的输出或输入 0

0x04 = I2C Slave 数据线

0x05 = I2C Master 时钟线

0x06 = I2C Master 数据线

0x07 = 未定义

0x08 = sensor 通道，除 P06、P07、P10~P15 这 8 个 IO 口外，其它所有引脚都可配置为 sensor 通道

0x09 = 模拟输入仅 P06、P07、P10、P11、P12、P13、P14、P15 可作为 ADC 采样通道（当配置 P25IOCFG=0x09,P26IOCFG=0x09 时，P25、P26 为外部晶振功能，M/K 只能二选一）

0x0A = 未定义

……

0x0F = 未定义

0x10 = SPIs 数据输出(主模式：从机输出主机输入)

0x11 = SPIm 片选输出(主模式：主机输出从机输入)

0x12 = SPIm 时钟输出(主模式：主机输出从机输入)

0x13 = SPIm 数据输出(主模式：主机输出从机输入)

0x14 = LED SEG0

0x15 = LED SEG1

0x16 =LED SEG2

0x17 =LED SEG3

0x18 =LED SEG4

0x19 =LED SEG5

0x1A =LED SEG6

0x1B =LED SEG7

0x1C =LED SEG8

0x1D = LED SEG9_GRID7

0x1E = LED SEG10_GRID6

0x1F =LED SEG11_GRID5

0x20 =LED SEG12_GRID4

0x21 = LED GRID0

0x22 = LED GRID1

0x23 =LED GRID2

0x24 =LED GRID3

0x25 = 定时器 0 输出

0x26 = 定时器 2 输出

0x27 = 通用串口 UART1 输出

0x28 = PWM1 输出

0x29 = PWM2 输出

0x2A = 通用串口 UART2 输出

0x2B = 未定义

……

0x2F = 未定义

0x30 = 调试串口输入。P0.1 默认为调试串口输入，其他引脚不可配置为调试串口输入

0x31 = I2C Slave 时钟线

0x32 = 定时器 3 输入 1

0x33 = 通用串口 UART1 输入

0x34 = SPIs 片选输入(从模式：从机输入主机输出)

0x35 = SPIs 时钟输入(从模式：从机输入主机输出)

0x36 = SPIs 数据输入(从模式：从机输入主机输出)

0x37 = SPIm 数据输入(从模式：主机输入从机输出)

0x38 = 外部中断 0

0x39 = 外部中断 1

0x3A = 外部中断 2

0x3B = 外部中断 3

0x3C = 外部中断 4

0x3D = 外部中断 5

0x3E = 通用串口 UART2 输入

0x3F = 未定义

0x40 = 高阻态（上电时的状态）

注：

1、调试烧录串口 uart 的输出只能是 P00 口，输入只能是 P01 口；

2、ADC 通道只能是 P0.6、P0.7、P1.0、P1.1、P1.2、P1.3、P1.4、P1.5；

3、P06～P15 不能作触摸 IO 功能；

4、除 P00、P01 外的其它 IO 口上电默值为 0x40 即高阻状态，上电后用户需要根据实际需求进行重新配置；

5、 P16IOCFG 的 bit7 位控制 P16 IO 的 RST 功能是否有效：bit7=0-使能 P16 的 RST 功能；bit7=1-禁止 P16 的 RST 功能；当 P16IOCFG=0x00 时，P16 为 RST，当 P16IOCFG.bit7=1 时 P16 为其它功能（详见下表 7-2）。

表 7-2：P16IOCFG 复用功能说明（此表不适用于其它 IO 口）

P16IOCFG	**P1OE_6**	**P16 口的复用功能**
0000 0000	1	P16 复用为硬件 RST 引脚
	0	禁止配置
0000 0001~0111 1111	x	禁止配置
1000 0000	x	P16 复用为 GPIO
1000 0001~1111 1111	x	P16 复用为除 RST、GPIO 外的其它功能（如上 PxyIOCFG 中所定义的功能）

注：x 表示 0 或 1

PnOE:方向控制寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
7	6	5	4	3	2	1	0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 PnOE[7:0]：IO 口方向控制寄存器

0 = 为输出方向

1 = 为输入方向（当 PxyIOCFG 配置为 GPIO 功能且 PnOE 为输入方向悬空时，必须进行内部上拉或外部上\下拉）

PnPU:上拉使能控制寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
7	6	5	4	3	2	1	0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　PnPU[7:0]：GPIO 内部上拉使能控制寄存器

0 = 禁止内部上拉

1 = 使能内部上拉(当 PxyIOCFG=0x40 或 0x09 时，上拉无效不会输出到 IO 端口)

PnPURSELRL：内部上拉电阻选择控制寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
3		2		1		0	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

PnPURSELRH：内部上拉电阻大小选择寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
7		6		5		4	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit[x:x-1]　Pn_PURSELRH/L[1:0]：对应 IO 口上拉电阻大小选择寄存器(x=1、3、5、7)

00 = 88K 上拉

01 = 2.2K 上拉

10 = 4.5K 上拉

11 = 10K 上拉

注：1、P0_0、P0_1 电阻配置寄存器上电默认值为 01-2.2K，P1_6 电阻配置寄存器上电默认值为 10-4.5K；因此这 3 个 IO 口不建议用作电平驱动类的控制引脚（比如不建议用来驱动控制 led、继电器、可控硅、蜂鸣器等等）；

2、其它 IO 电阻配置寄存器上电默认值为 00-88K；

PnOD：开漏输出控制寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　PnOD[1:0]：IO 口开漏输出控制寄存器

0 = 不使能开漏

1 = 使能开漏

PnLDRVRL：IO 驱动电流配置寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
3		2		1		0	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

PnLDRVRH：IO 驱动电流配置寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

7		6		5		4	
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit x-(x-1) PnLDRVRH/L[1:0]：对应 IO 口驱动电流配置寄存器(x=1、3、5、7)

00 = 6mA

01 = 30mA

10 = 60mA

11 = 6mA

PnSM：IO 口施密特使能寄存器 n(0~4)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 Pn_x：IO 口电平回读增强控制寄存器(x=0、1、2、3、4、5、6、7)

0 = 不使能施密电平回读增强特功能；

1 = 使能施密特电平回读增强功能（芯片有大电流时需使能）

注：P1SM_6 上电默认为 1，其它默认为 0。

7.2、配置引脚为 GPIO 的操作步骤

1、使能 GPIO 模块的时钟

MDLCKCR_IOCEN = 0x08;

2、配置功能复用寄存器：

PxyIOCFG = 0x00;//将引脚配置为 GPIO 功能

3、配置数据寄存器：

配置 Pn 对应位，选择输出电平。

4、配置驱动电流大小：

配置驱动电流选择寄存器 PnLDRVRH：PnLDRVRL，选择驱动电流大小。

5、配置内部上拉电阻大小：

配置 PnPURSELRH：PnPURSELRL 寄存器，选择内部上拉电阻大小。P0_0、P0_1、P1_6 默认为 01=2.2K,其它 IO 口默认为 00=88K。

6、内部上拉使能配置

配置 PnPU 寄存器，选择是否使能内部上拉。

7、配置开漏输出：

配置 PnOD 寄存器，选择是否使能开漏输出。

8、方向选择：

配置 PnOE 寄存器，选择 GPIO 方向。

例 7-1：初始化 P1.0、P1.1

```
MDLCKCR_IOCEN = 1<<3 ;//使能 GPIO 模块时钟
P10IOCFG = 0x00 ;//初始化 P1.0 为 GPIO
P11IOCFG = 0x00 ;//初始化 P1.1 为 GPIO
P1 = 0x00 ;//初始化 P1
P1LDRVRL_0 = 0x02 ;//配置 P1.0 驱动电流 30mA
P1LDRVRL_1 = 0x08 ;//配置 P1.1 驱动电流 30mA
P1PURSELRL_0 = 0x03 ;//配置 P1.0 上拉电阻 10K
P1PURSELRL_1 = 0x0C ;//配置 P1.1 上拉电阻 10K
P1PU_0 = 0x01 ;//使能 P1.0 内部上拉电阻
P1PU_1 = 0x02 ;//使能 P1.1 内部上拉电阻
P10D_0 = 0x00 ;//关闭 P1.0 开漏输出
P10D_1 = 0x00 ;//关闭 P1.1 开漏输出
P10E_0 = 0x00 ;//P1.0 输出
P10E_1 = 0x00 ;//P1.1 输出
```

8、 看门狗定时器(WDT)

CBM73xxF 系列 MCU 具有软件看门狗溢出复位功能，可软件开启、关闭；WDT 具有以下特性：

- WDT 的时钟源来自于内部低速时钟 800K 的 8 分频，即固定为 100K 的计数时钟；
- WDT 时基 100K 配合计数周期配置寄存器 WDTSETR，可得到宽范围溢出复位周期 320us~42s，满足大都数使用情况；
- WDT 具有溢出中断与溢出复位两种功能，溢出中断时间为 160us~21s；即 WDT 第一次溢出时可以产生溢出中断（如果使能了溢出中断）；而第二次溢出时则会发生溢出复位（必定发生）；
- WDT 的溢出中断可以实现定时功能，但每次在 WDT 中断程序中都需要执行喂狗动作，否则第二次溢出则会发生复位；
- WDT 在进入空闲、休眠模式下时会自动停止计数，唤醒后自动开启；另外，LVD 功能开启时，可以软件选择是否停止 WDT 计数，详情请见“5.1 章节 MISCR3 LVDSTOPEN”描述。

注：如果 WDT 作溢出复位功能使用时，则清除 WDT 计数器的行为(即“喂狗”)不能放在 WDT 的中断服务程序中，应该放 main 函数中定时执行喂狗动作。(否则当程序跑飞时，MCU 仍然能正常响应中断，达不到恢复程序的目的)

8.1、WDT 寄存器的定义

表 8-1、WDT 寄存器表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
MDLCKCR	0x51FF	XRAM	模块时钟控制寄存器	0000 1000
WDTCR	0x60FF	XRAM	WDT 控制寄存器	0000 1111
WDTSETR	0x61FF	XRAM	WDT 周期配置寄存器	0001 0011
WDTSR	0x62FF	XRAM	WDT 状态寄存器	0000 0000
WDTENR	0x63FF	XRAM	WDT 使能寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

如上表 8-1 所示，寄存器 MDLCKCR 的 bit0 为 WDT 的时钟使能位，操作 WDT 前必须先写 MDLCKCR_WDTEN=0x01；寄存器 WDTCR 可以实现 WDT 计数清零（清狗动作）；寄存器 WDTSETR 可以配置 WDT 计数器溢出时间；寄存器 WDTSR 为 WDT 中断状态标志；寄存器 WDTENR 可以使 WDT 计数功能。

任何时刻，软件对 WDTCR 寄存器依次写 0x5A、0xA5 可清 WDT 计数器，计数器将重新从 0 开始计数(写 WDTCR 顺序不能变)。

WDTCR:WDT 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
WDPC6	WDPC5	WDPC4	WDPC3	WDPC2	WDPC1	WDPC0	WDP
W-0	W-0	W-0	W-0	W-1	W-1	W-1	R-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1　　WDPC[6:0]：清狗控制命令位

依次写入 0x5A、0xA5 可清除计数器，使计数器重新从 0 开始计数。

bit 0　　WDP：状态标志位

软件写 WDTCR 会翻转 WDP 的状态；溢出或写 WDTSETR 时恢复到 0

注：依次写 0x5A、0xA5 的顺序不能乱，否则无效。

WDTSETR:WDT 周期配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CNTDIV3	CNTDIV2	CNTDIV1	CNTDIV0	CNTLEN3	CNTLEN 2	CNTLEN 1	CNTLEN 0
R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-4　　CNTDIV [3:0]：WDT 时基分频器

0000 = 1 分频
0001 = 2 分频
0010 = 4 分频
0011 = 8 分频
0100 = 16 分频
……
1010 = 1024 分频
1011 = 2048 分频
1100 = 4096 分频
1101~1111 = 8192 分频

bit 3-0 CNTLEN[3:0]：配置溢出周期的计数长度，0 <= CNTLEN <= 15，对应计数时基长度为(CNTLEN + 1) * 16。

注：

1、WDT 溢出周期 = 2^{CNTDIV}* (CNTLEN + 1) * 16 * 10us(T_{100K})。（160us~21s）

2、WDT 复位周期 = 2 * 溢出周期 = 2 * 2^{CNTDIV}* (CNTLEN + 1) * 16 * 10us(T_{100K})。（320us~42s）

WDTSR:WDT 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	-	WDTINT
U-x	U-x	U-x	U-x	U-x	U-x	U-x	R-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 保留未用

bit 0 WDTINT：WDT 溢出中断标志位,自动清 0

0 = 无

1 = 发生溢出

WDTENR:WDT 计数使能寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	-	CNTEN
U-x	U-x	U-x	U-x	U-x	U-x	U-x	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 保留未用

bit 0 WDT 计数器使能位

0 = 不使能

1 = 使能

8.2、WDT 配置步骤

1、WDT 模块时钟使能：

MDLCKCR_WDTEN = 0x01;//将 MDLCKCR 寄存器的 WDTEN 位置 1，使能 WDT 模块时钟。

2、清 WDT：

配置 WDT 开始工作前，先对 WDTCR 寄存器依次 5A、A5，清 WDT。

WDTCR = 0x5A;

WDTCR = 0xA5;

3、配置 WDT 溢出周期：

- 计数时钟分频选择：WDTSETR 寄存器的 CNTDIV<3:0>位可以对 WDT 计数时钟进行分频。(2^{CNTDIV} 最大为 8192)
- 计数长度选择：WDTSETR 寄存器的 CNTLEN<3:0>位可以选择 WDT 计数长度，0 <= CNTLEN <= 15，对应计数时基长度为(CNTLEN + 1) * 16。

WDT 溢出周期 = 2^{CNTDIV}* (CNTLEN + 1) * 16 * 10us(T_{100K})。（160us~21s）

WDT 复位周期 = 2 * 溢出周期 = 2 * 2^{CNTDIV}* (CNTLEN + 1) * 16 * 10us(T_{100K})。（320us~42s）

4、中断控制：

WDT 模块中没有中断使能位，只要依次将 IRQ0ER 寄存器的 WDTINT 位、IE 寄存器的 EX0 位、IE 寄存器的 EA 位置 1，就可以使能 WDT 溢出中断。

IRQ0ER_ WDTINT = 0x04;//使能 WDT 模块中断

EX0 = 1;//使能内核中断 0

EA = 1;//使能内核总中断

5、使能 WDT 计数：

WDTENR_CNTEN = 0x01;//将 WDTENR 寄存器的 CNTEN 位置 1，WDT 计数器开始计数。

WDT 工作时序如图 8-1 所示。

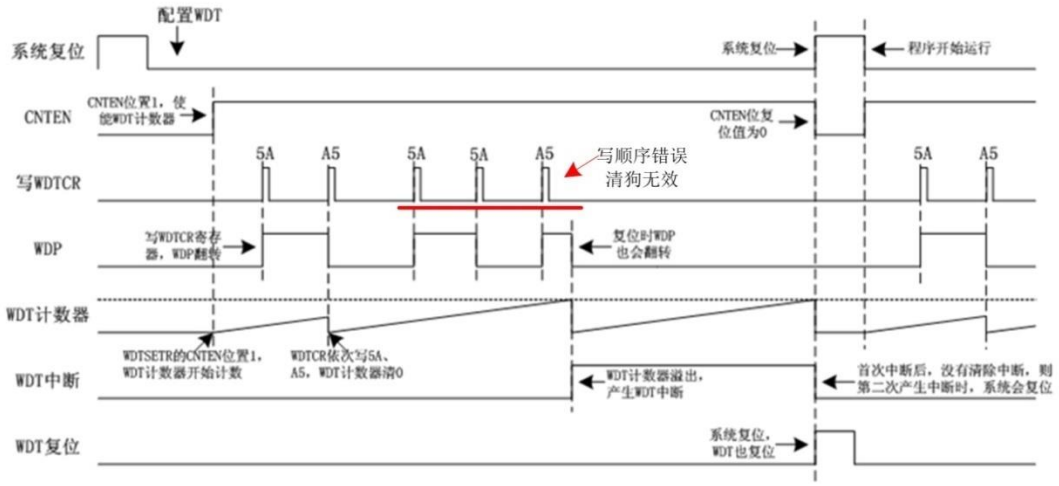


图 8-1、WDT 工作时序

9、 8 位定时器 Timer0

Timer0 模块是一个 8 位定时器，具有如下特点：

- 8 位定时器
- 3 位预分频器
- 8 位周期寄存器(TOCMP0)
- 8 位 PWM 脉宽调制寄存器(TOCMP1)
- Timer0 有高速时钟、低速时钟可供选择如下：高速时钟（系统副时钟 SCLK1）、低速时钟（系统副时钟 SCLK2、内部低速 800K、由内部 800k 固定 8 分频来的 100K）
- 支持实时读取计数器值

表 9-1、Timer0 时钟构成

Timer0 时钟	可选时钟	时钟源
高速时钟	SCLK1	IHCLK（48M）、EHCLK（M 级晶振）、ILCLK（800K）
低速时钟	SCLK2	EHCLK（M 级晶振）、ELCLK（K 级晶振）
	内部低速时钟 800K	ILCLK（800K）
	内部 100K	由内部 800k 固定 8 分频来的 100K

Timer0 时钟框图详情请参考“4.1、系统总时钟框图”描述。

注：T0 定时器的时钟源选择与以下寄存器位有关：TOCKCR_CLKSEL、MDLCKCR_TLCKSEL、MISCR2_RTCCLKSEL，参考下表 9-2。

表 9-2：T0 时钟源选择寄存器关系表

TOCKCR_CLKSEL	MDLCKCR_TLCKSEL	MISCR2_RTCCLKSEL	T0 定时器所选时钟源
0	x	x	SCLK1
1	0	x	内部低速时钟 800K
1	1	0	内部 100K(由内部低速时钟 800K 固定 8 分频来)
		1	SCLK2

9.1、Timer0 支持的功能

Timer0 支持以下 3 种功能：

- 间隔定时(定时器模式下)
- 方波输出(定时器模式下输出)
- PWM 输出(PWM 模式下输出)

间隔定时的操作步骤

配置 Timer0 为间隔定时功能的操作步骤如下：

1、Timer 模块时钟使能

MDLCKCR_TIMEREN = 0x04;//将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

2、关闭 Timer0

当 Timer0 在停止工作状态下才能完成 Timer0 的以下配置，T0CR_TEN = 0 即关闭 Timer0。

3、清中断标志：T0SR = 0x00 ;

4、间隔时间配置

Timer0 的间隔定时时间是通过 TOCMP0 寄存器配置，具体的间隔时间计算如下：

间隔时间 = (TOCMP0) * Timer0 时钟周期 = (TOCMP0) * Timer0 计数时钟分频 * Timer0 计数时钟周期。

Timer0 计数时钟周期由选择的计数时钟决定(“第 2 步_计数时钟选择”)，共有四种时钟：

- 内部高速 OSC48M 时钟
- 内部低速 800K

- 外部 32.768K\100K 晶振(必须确保外接了 32.768K\100K 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振）

5、Timer0 计数时钟参数配置

- 计数时钟选择

  1)、TOCKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer0 的时钟；(0：高速时钟；1：低速时钟)

  2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer0 的时钟。

- 计数时钟分频配置

  TOCKCR 寄存器的 CLKDIV[3:0]位可以对选择的时钟源进行分频。

- 计数时钟使能

  TOCKCR_CLKEN = 0x40;//将 TOCKCR 寄存器的 CLKEN 位置 1，使能 Timer0 计数时钟。

6、工作模式选择

Timer0 有定时器/方波、PWM 两种工作模式，通过 TOCR 寄存器的 MODE 位控制；这两种模式都可以实现间隔定时中断功能。

7、Timer0 模块中断控制

Timer0 只有一种中断，就是当 Timer0 计数器值等于(TOCMP0)时，产生 Timer0 中断。在间隔定时、方波输出、PWM 输出三种模式下都可以产生 Timer0 中断。TOCR 寄存器的 INTEN 位是 Timer0 中断使能位，TOSR 寄存器的 INTSR 位是 Timer0 中断标志位。

8、使能外设模块中断

使能寄存器 IRQ2ER 的 bit1 位，IRQ2ER_TCM0INT = 0x02。

9、使能内核中断

1)、EX2 = 1；//使能内核中断 2

2)、配置内核中断 2 中断优先级 IP1_bit[5:4]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

10、Timer0 使能

置 TOCR_TEN = 1 使能 Timer0 开始工作(注意：在初始化定时器或 Timer0 关闭情况下，给 TOCMP0 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 TOCR_TEN=1 使能 T0 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 TOCKCR_CLKSEL 的时钟)。

方波输出的操作步骤

配置 Timer0 为方波输出功能的操作步骤如下：

1、端口配置

P10IOCFG = 0x25;//配置 PxyIOCFG 寄存器将 I/O 配置成定时器 0 输出功能，PxOE_y 方向寄存器会自动配置为输出

注：操作 IO 端口寄存器前应先使能 IO 模块的时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1

2-10、与以上 timer0 的“间隔定时的操作步骤”相同。(注意：在初始化定时器或 Timer0 关闭情况下，给 TOCMP0 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 TOCR_TEN=1 使能 T0 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 TOCKCR_CLKSEL 的时钟)。

Timer0 间隔定时/方波输出模式工作时序图

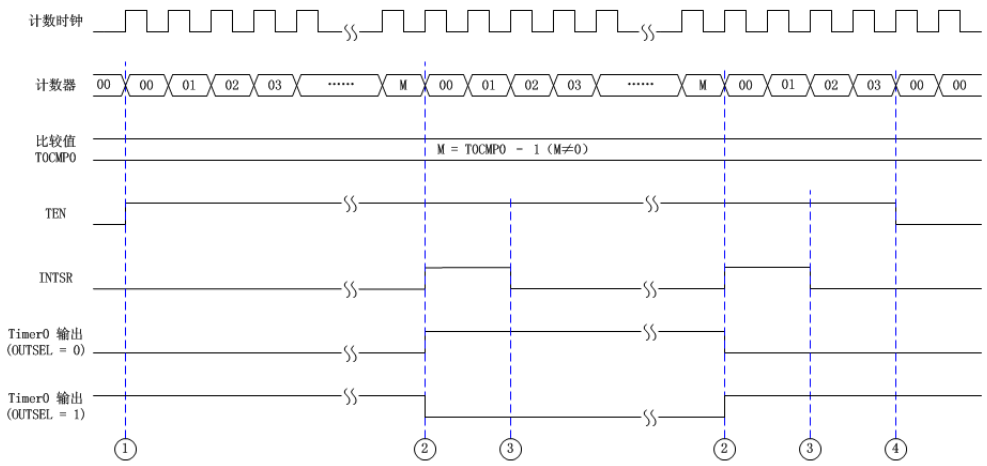


图 9-1、Timer0 间隔定时/方波输出工作时序

波形说明：

① 使能定时器，计数器开始计数；

② 计数器值等于 M，产生 Timer0 中断(即 T0SR 寄存器的 INTSR 位置 1)，同时 Timer0 输出翻转；

③ 软件处理并清除中断；

④ 不使能定时器，计数器清 0 并停止计数；

间隔时间 = (T0CMP0) * Timer0 时钟周期

= (T0CMP0) * Timer0 计数时钟分频 * Timer0 计数时钟周期

方波周期 = 2 * 间隔时间。

PWM 输出的操作步骤

配置 Timer0 为 PWM 输出功能的操作步骤如下：

1、端口配置

P10IOCFG = 0x25;//通过设置 PxyIOCFG 寄存器将 I/O 配置成定时器 0 输出功能，Pxy 引脚方向寄存器会自动配置为输出

注：操作 IO 端口寄存器前应先使能 IO 模块的时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1

2、Timer 模块时钟使能

MDLCKCR_TIMEREN = 0x04;//将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer0

当 Timer0 在停止工作状态下才能完成 Timer0 的以下配置，T0CR_TEN = 0 即关闭 Timer0。

4、清中断标志：T0SR = 0x00 ;

5、PWM 占空比配置

Timer0 的 PWM 占空比通过 T0CMP0 与 T0CMP1 寄存器值来配置;其中 T0CMP0 寄存器的值决定了整个 PWM 的周期,而 T0CMP1 寄存器的值决定了 PWM 电平输出的翻转时间，具体计算如下：

PWM 周期 = (T0CMP0) * Timer0 时钟周期 = (T0CMP0) * Timer0 计数时钟分频 * Timer0 计数时钟周期

PWM 占空比 = T0CMP1 / T0CMP0 (或者是（T0CMP0-T0CMP1）/ T0CMP0).（占空比公式取决于配置的初始电平的极性）

Timer0 计数时钟周期由选择的计数时钟决定(“第 3 步_计数时钟选择”)，共有四种时钟：

- 内部高速 OSC48M 时钟
- 内部低速 800K

- 外部 32.768K\100K 晶振(必须确保外接了 32.768K\100K 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振

注：

1)、当 TOCMP0 = 0 时，按 TOCMP0 = 256 计算；

2)、当 TOCMP0 = 1 时，没有方波输出，也不产生 Timer0 中断；

3)、寄存器 TOCMP1 的值必须小于寄存器 TOCMP0 的值；

4)、PWM 模式下，可以直接修改 TOCMP0、TOCMP1 来改变 PWM 的占空比；必须先修改 TOCMP0 再修改 TOCMP1，新的占空比配置将在下个 PWM 周期生效；

6、Timer0 计数时钟参数配置

- 计数时钟选择

  1)、TOCKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer0 的时钟；(0：高速时钟；1：低速时钟)

  2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer0 的时钟。

- 计数时钟分频配置

  TOCKCR 寄存器的 CLKDIV<3:0>位可以对选择的时钟源进行分频。

- 计数时钟使能

  TOCKCR_CLKEN = 0x40;//将 TOCKCR 寄存器的 CLKEN 位置 1，使能 Timer0 计数时钟。

7、Timer0 的 PWM 模式配置

- 工作模式选择：TOCR_MODE = 0x04;//将 TOCR 寄存器的 MODE 位置 1，工作模式选择为 PWM 模式；
- 输出初始电平选择:TOCR 寄存器的 OUTSEL 位是 Timer0 输出的初始电平选择位,对于方波输出和 PWM 输出都有效,OUTSEL 位决定输出初始电平的极性。

8、中断控制

Timer0 只有一种中断，就是 Timer0 计数器值等于(TOCMP0 - 1)时，产生 Timer0 中断。在间隔定时、方波输出、PWM 输出三种模式下都可以产生 Timer0 中断。TOCR 寄存器的 INTEN 位是 Timer0 中断使能位，TOSR 寄存器的 INTSR 位是 Timer0 中断标志位。

9、使能外设模块中断

使能寄存器 IRQ2ER 的 bit2 位，IRQ2ER_TCM0INT = 0x04;

10、使能内核中断

1)、EX2 = 1; //使能内核中断 2

2)、配置内核中断 2 中断优先级 IP1_bit[5:4]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

11、Timer0 使能

置 TOCR_TEN = 1 即可使能 Timer0 开始工作(注意：在初始化定时器或 Timer0 关闭情况下，给 TOCMP0、TOCMP1 寄存器赋值后必须等待>=3*$T_{所选时钟源}$的时间后才能 TOCR_TEN=1 使能 T0 工作,$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 TOCKCR_CLKSEL 的时钟)。

Timer0 PWM 输出模式工作时序图

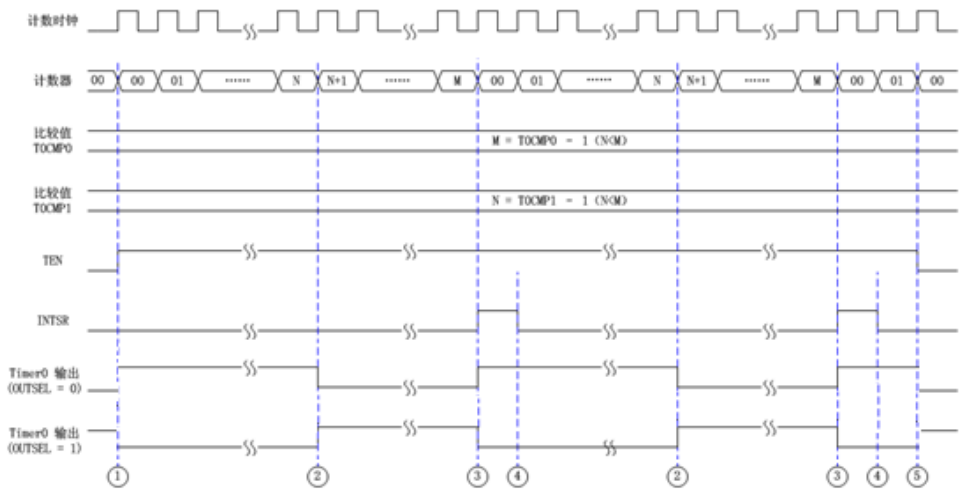


图 9-2、Timer0 间隔定时/方波输出工作时序

波形说明：

① 使能定时器，计数器开始计数；

② 计数器值等于 N，Timer0 输出翻转；

③ 计数器值等于 M，产生 Timer0 中断(即 TOSR 寄存器的 INTSR 位置 1)，同时 Timer0 输出翻转；

④ 软件处理并清除中断；

⑤ 不使能定时器，计数器清 0 并停止计数。

PWM 周期 = (TOCMP0)x Timer0 时钟周期

= (TOCMP0) x Timer0 计数时钟分频 x Timer0 计数时钟周期

PWM 占空比 = TOCMP1 ：TOCMP0(或者是（TOCMP0-TOCMP1）：TOCMP0)

9.2、Timer0 寄存器的定义

表 9-3、Timer0 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
T0CKCR	0x70FF	XRAM	Timer0时钟控制寄存器	0000 0000
T0CR	0x71FF	XRAM	Timer0模式控制寄存器	0000 0000
T0CMP0	0x72FF	XRAM	Timer0比较寄存器0	0000 0000
T0CMP1	0x73FF	XRAM	Timer0比较寄存器1	0000 0000
T0SR	0x74FF	XRAM	Timer0状态寄存器	0000 0010
T0CNT	0x76FF	XRAM	Timer0计数寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

T0CKCR:Timer0 时钟寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	CLKEN	CLKSEL	-	CLKDIV2	CLKDIV1	CLKDIV0
U-x	U-x	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 保留未用

bit 5 CLKEN：Timer0 计数时钟使能位

0 = 关闭 Timer0 计数时钟

1 = 打开 Timer0 计数时钟

bit 4 CLKSEL：Timer0 计数时钟选择位

0 = 选择高速时钟（系统副时钟 SCLK1）作为 Timer0 计数时钟

1 = 选择低速时钟（系统副时钟 SCLK2、内部低速 800K）作为 Timer0 计数时钟(具体是 800K 还是外部晶振需要配置 MDLCKCR_TLCKSEL 来决定，详情请参考“4.2、MDCLKCR 寄存器”描述)。

bit 3 保留未用

bit 2-0 CLKDIV：Timer0 计数时钟分频选择位

000 = 2 分频

001 = 4 分频

……

101 = 64 分频

110 = 128 分频

111 = 1 分频

T0CR:Timer0 模式控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	READMODE	INTEN	OUTSEL	MODE	TEN
U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-3 保留未用

bit4 READMODE：读 timer0 计数器模式选择，如果要读取 T0CNT 的值，则必须根据 Timer0 的时钟配置此位为正确的模式值，否则读到的 T0CNT 值与真实值之间可能存在误差

0 = Timer0 的时钟是 M 级时钟源

1 = Timer0 的时钟是 K 级时钟源（32.768K、100K、800K 或外部 M 级时钟经过 XCLKDIV 分频后的 K 级时钟）

bit 3 INTEN：Timer0 中断使能位

0 = 禁止计数器与 T0CMP0 匹配中断

1 = 使能计数器与 T0CMP0 匹配中断

bit 2 OUTSEL：Timer0 输出初始电平选择位

0 = Timer0 输出初始电平为低电平

1 = Timer0 输出初始电平为高电平

bit 1 MODE：Timer0 工作模式选择位

0 = 定时器/方波模式

1 = PWM 模式

bit 0 TEN：Timer0 使能位

0 = Timer0 停止工作

1 = Timer0 开始工作

T0CMP0:Timer0 比较寄存器 0

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T0CMP0 [7:0]：Timer0 比较寄存器 0

注：1、当 T0CMP0 = 0 时，按 T0CMP0 = 256 计算；

2、当 T0CMP0 = 1 时，没有方波输出，也不产生 Timer0 中断；

3、寄存器 T0CMP0 单独使用时，可用作间隔定时中断功能；寄存器 T0CMP0 与寄存器 T0CMP1 配合使用时，可以输出 PWM，其中 T0CMP0 控制 PWM 输出周期，T0CMP1 控制 PWM 输出占空比；

4、T0CMP0 寄存器可在 timer0 工作状态下实时更新，更新值将在下个计数周期生效；

5、在初始化 T0 或定时器关闭情况下，设置 T0CMP0、T0CMP1 后需要延时等待 3*T $_{所选时钟源}$的时间才能使能 T0 工作，T $_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T0CKCR_CLKSEL 的时钟。

例 9-1、更新 T0CMP0 的操作例程：

```
T0CMP0 = 0x55 ;//实时修改 T0CMP0 寄存器
```

T0CMP1:Timer0 比较寄存器 1

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	CLKEN	CLKSEL	-	CLKDIV2	CLKDIV1	CLKDIV0
U-x	U-x	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 T0CMP1<7:0>：Timer0 比较寄存器 1，用于控制 PWM 的占空比

注：1、寄存器 T0CMP1 的值必须小于寄存器 T0CMP0 的值；

2、寄存器 T0CMP0 决定 PWM 的输出周期，寄存器 T0CMP1 决定 PWM 的输出占空比，在 Timer0 工作期间，可以修改 T0CMP0、T0CMP1 寄存器改变输出 PWM 的频率或占空比，但必须先修改 T0CMP0 再修改 T0CMP1；

3、在初始化 T0 或定时器关闭情况下，设置 T0CMP0、T0CMP1 后需要延时等待 3*T $_{所选时钟源}$的时间才能使能 T0 工作，T $_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T0CKCR_CLKSEL 的时钟。

例 9-1、更新占空比的操作例程：

```
T0CMP0 = 0xA0 ;//先修改 T0CMP0 寄存器
T0CMP1 = 0x40 ;//再修改 T0CMP1 寄存器
```

T0SR:Timer0 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	-	INTSR
U-x	U-x	U-x	U-x	U-x	U-x	U-x	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1　　保留未用

bit 0　　INTSR：Timer0 计数器与 T0CMP0 匹配中断标志位

0 = Timer0 计数器与 T0CMP0 不匹配

1 = 发生了 Timer0 计数器与 T0CMP0 匹配，需软件写 0 清 0

T0CNT:Timer0 计数寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　T0CNT[7:0]: Timer0 当前计数值

读寄存器 T0CNT，可以得到 timer0 当前的计数值大小，但必须配合 T0CR 的 READMODE 位使用，如下例程。

（重启定时器可以清零计数值重新开始计时：T0CR_TEN = 0x00 ; T0CR_TEN = 0x01 ;）

例 9-2、Timer0 的时钟是 M 级时钟时，读 T0CNT 计数值

```
//初始化 timer0 配置
T0CR_TEN = 0x00 ;
T0CR_READMODE = 0x00;//M 级时钟
T0CR_TEN = 0x01;
……
temp = TCNT;//读 Timer0 计数值
……
```

例 9-3、Timer0 的时钟是 K 级时钟时，读 T0CNT 计数值

```
//初始化 timer0 配置
T0CR_TEN = 0x00;
T0CR_READMODE = 0x10; //K 级时钟
T0CR_TEN = 0x01;
……
temp = TCNT;//读 Timer0 计数值
……
```

9.3、Timer0 中断

不论是在定时器模式还是 PWM 模式下，当 Timer0 计数器值等于(T0CMP0 - 1)时，都会产生 Timer0 中断。

只要有 Timer0 中断条件产生，不论是否使能 Timer0 中断，T0SR 寄存器的 INTSR 位(Timer0 中断标志位)都会置 1，INTSR 位必须在软件中清零。Timer0 的中断使能位是 T0CR 寄存器的 INTEN 位。

注：1、在休眠模式下，Timer0 中断无法唤醒处理器。

2、在空闲模式下，Timer0 中断可以唤醒 处理器。

10、8 位定时器 Timer1

Timer1 模块是一个 8 位定时器，具有如下特点：

- 8 位定时器
- 3 位预分频器
- 8 位周期与脉宽调制寄存器(T1CMP)
- 可编程外部时钟边沿选择
- Timer1 还有高速时钟、低速时钟可供选择如下：高速时钟（系统副时钟 SCLK1）、低速时钟（系统副时钟 SCLK2、内部低速 800K、由内部 800k 固定 8 分频来的 100K）
- 支持实时读取计数器值

表 10-1、Timer1 时钟构成

Timer1 时钟	可选时钟	时钟源
高速时钟	SCLK1	IHCLK（48M）、EHCLK（M 级晶振）、ILCLK（800K）
低速时钟	SCLK2	EHCLK（M 级晶振）、ELCLK（K 级晶振）
	内部低速时钟 800K	ILCLK（800K）
	内部 100K	由内部 800k 固定 8 分频来的 100K

Timer1 时钟框图详情请参考“4.1 系统总时钟框图”描述。

注：T1 定时器的时钟源选择与以下寄存器位有关：T1CKCR_CLKSEL、MDLCKCR_TLCKSEL、MISCR2_RTCCLKSEL，参考下表 10-2。

表 10-2：T1 时钟源选择寄存器关系表

T1CKCR_CLKSEL	MDLCKCR_TLCKSEL	MISCR2_RTCCLKSEL	T1 定时器所选时钟源
0	x	x	SCLK1
1	0	x	内部低速时钟 800K
1	1	0	内部 100K(由内部低速时钟 800K 固定 8 分频来)
		1	SCLK2

10.1、Timer1 支持的功能

Timer1 支持以下 4 种功能：

- 间隔定时(定时器模式下)
- 方波输出(定时器模式下输出)
- PWM 输出(PWM 模式下输出)
- 外部事件计数(定时器模式下对外部输入电平跳变计数)

间隔定时的操作步骤

配置 Timer1 为间隔定时功能的操作步骤如下：

1、Timer 模块时钟使能

将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

2、关闭 Timer1

当 Timer1 在停止工作状态下才能完成 Timer1 的以下配置，T1CR_TEN = 0 即关闭 Timer1。

3、清中断标志：T1SR = 0x00 ；

4、间隔时间配置

Timer1 的间隔定时时间是通过 T1CMP 寄存器配置，具体的间隔时间计算如下：

间隔时间 = T1CMP * Timer1 时钟周期 = T1CMP * Timer1 计数时钟分频 * Timer1 计数时钟周期

Timer1 计数时钟周期由选择的计数时钟决定(“第 2 步_计数时钟选择”)，共有三种时钟：

- 内部高速 OSC48M
- 内部低速 800K
- 外部 32.768K/100k 晶振(必须确保外接了 32.768K/100k 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振）

5、Timer1 计数时钟参数配置

- 计数时钟选择：

1)、T1CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer1 的时钟。(0：高速时钟；1：低速时钟)

2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer1 的时钟。

- 计数时钟分频配置：

T1CKCR 寄存器的 CLKDIV[3:0]位可以对选择的时钟源进行分频。(此时不能配置为外部信号的有效沿选项。)

- 计数时钟使能：

将 T1CKCR 寄存器的 CLKEN 位置 1，使能 Timer1 计数时钟。

6、工作模式选择

将 T1CR 寄存器的 MODE 位清 0，工作模式选择为定时器模式。

7、Timer1 模块中断控制：

Timer1 只有一个中断，但是在间隔定时、方波输出、PWM 输出、外部事件计数模式下，产生 Timer1 中断的条件不同：

1)、在间隔定时、方波输出、外部事件计数模式下：

当 Timer1 计数器值等于(T1CMP - 1)时，会产生 Timer1 中断。

2)、在 PWM 输出模式下：

当 Timer1 计数器值溢出，即等 于 0xFF 时，会产生 Timer1 中断。

T1CR 寄存器的 INTEN 位是 Timer1 中断使能位，T1SR 寄存器的 INTSR 位是 Timer1 中断标志位

8、使能外设模块中断

使能寄存器 IRQ3ER 的 bit1 位，IRQ3ER_TCM1INT = 0x02。

9、使能内核中断

1)、EX3 = 1; //使能内核中断 3

2)、配置内核中断 3 中断优先级 IP1_bit[7:6]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

10、Timer1 使能：

置 T1CR_TEN = 1 使能 Timer1 开始工作(注意:在初始化定时器或 Timer1 关闭情况下,给 T1CMP 寄存器赋值后必须等待>=3*$T_{所选时钟源}$的时间后才能 T1CR_TEN=1 使能 T1 工作，$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T1CKCR_CLKSEL 的时钟)。

方波输出的操作步骤

配置 Timer1 为方波输出功能的操作步骤如下：

1、端口配置：

1)、P10IOCFG = 0x02;//通过设置 PxyIOCFG 寄存器将 I/O 配置为定时器 1 输出/输入功能

2)、P10E_0 = 0x00;//配置 PxOE_0 方向寄存器为输出（方波输出）

注：操作 IO 端口寄存器前应先使能 IO 模块的时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1

2-10、与以上 timer1 的“间隔定时的操作步骤”相同。(注意：在初始化定时器或 Timer1 关闭情况下，给 T1CMP 寄存器赋值后必须等待>=3*$T_{所选时钟源}$的时间后才能 T1CR_TEN=1 使能 T1 工作，$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T1CKCR_CLKSEL 的时钟)。

Timer1 间隔定时/方波输出工作时序图

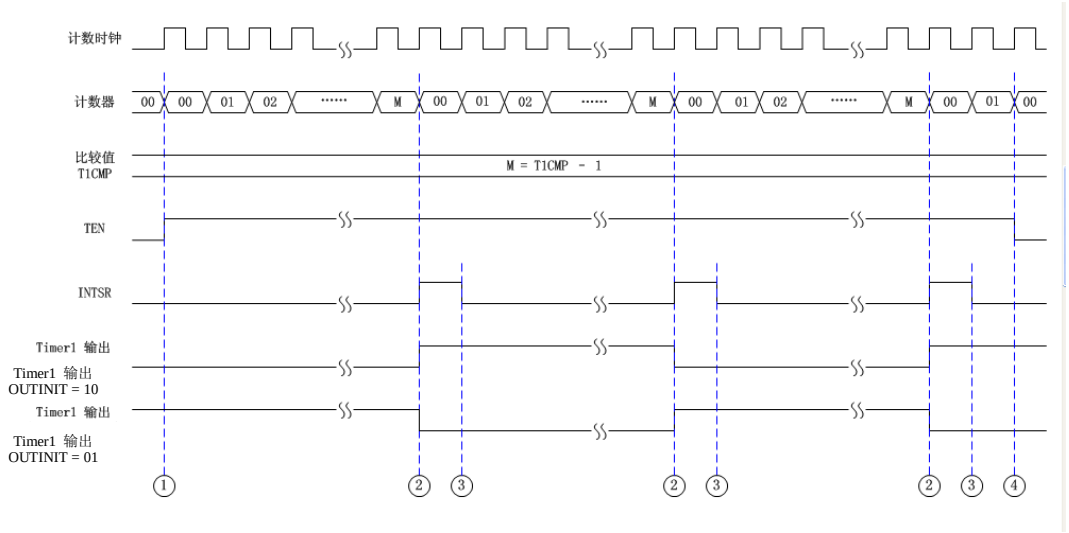


图 10-1、Timer1 间隔定时/方波输出工作时序

波形说明：

① 使能定时器，计数器开始计数；

② 计数器值等于 M，产生 Timer1 中断(即 T1SR 寄存器的 INTSR 位置 1)，同时 Timer1 输出翻转；

③ 软件处理并清除中断；

④ 不使能定时器，计数器清 0 并停止计数。

间隔时间 = (T1CMP) x Timer1 时钟周期

= (T1CMP) x Timer1 计数时钟分频 x Timer1 计数时钟周期。

方波周期 = 2 x 间隔时间。

注：1、当 T1CMP = 0 时，按 T1CMP = 256 计算

2、当 T1CMP = 1 时，没有方波输出，也不产生 Timer1 中断。

PWM 输出的操作步骤

配置 Timer1 为 PWM 输出功能的操作步骤如下：

1、端口配置

1)、P10IOCFG = 0x02;//通过设置 PxyIOCFG 寄存器将 I/O 配置为定时器 1 输出/输入功能

2)、P10E_0 = 0x00;//配置 PxOE_0 方向寄存器为输出（方波输出）

注：操作 IO 端口寄存器前应先使能 IO 模块的时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1

2、Timer 模块时钟使能

MDLCKCR_TIMEREN = 0x04;//将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer1

当 Timer1 在停止工作状态下才能完成 Timer1 的以下配置，T1CR_TEN = 0 即关闭 Timer1。

4、清中断标志：T1SR = 0x00 ；

5、PWM 占空比配置

Timer1 的 PWM 占空比配置与 Timer0 有所区别，PWM 周期固定，占空比可通过 T1CMP 配置调节，具体计算如下：

PWM 周期 = (0xFF + 1) * Timer1 时钟周期

= 256 * Timer1 计数时钟分频 * Timer1 计数时钟周期

PWM 占空比 = T1CMP / 256 或 (0x100 - T1CMP) / 256 .（占空比公式取决于配置的初始电平的极性）

Timer1 计数时钟周期由选择的计数时钟决定(“第 3 步_计数时钟选择”)，共有三种时钟：

- 内部高速 OSC48M
- 内部低速 800K
- 外部 32.768K\100K 晶振(必须确保外接了 32.768K\100K 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振）

注：

1)、Timer1 PWM 的周期是固定的(0xFF + 1)，不能配置；占空比可以通过配置 T1CMP 来调节，所以 T1CMP 必须小于(0xFF + 1)，也就是不能等于 00。(当输出反转位置 1 时，T1CMP 配置的是高电平的时间。)

2)、T1CMP 在 PWM 模式下，Timer1 工作期间可以直接修改。

6、Timer1 计数时钟参数配置

- 计数时钟选择：

  1)、T1CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer1 的时钟。(0：高速时钟；1：低速时钟)

  2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer1 的时钟。

- 计数时钟分频配置：

  T1CKCR 寄存器的 CLKDIV<3:0>位可以对选择的时钟源进行分频。(此时不能配置为外部信号的有效沿选项。)

- 计数时钟使能：

  将 T1CKCR 寄存器的 CLKEN 位置 1，使能 Timer1 计数时钟。

7、Timer1 模式配置

- 输出电平翻转使能：

  T1CR 寄存器的 INVEN 置 1，使能当 Timer1 计数器值等于(T1CMP - 1)时输出电平翻转。(不使能没有 PWM 波形输出)

- 输出反转选择：

  T1CR 寄存器的 OUTSEL 位是 Timer1 输出反转使能位，只对 PWM 输出都有效，OUTSEL 位置 1 后输出波形会发生反转。

- 工作模式选择：

  将 T1CR 寄存器的 MODE 位置 1，工作模式选择为 PWM 模式。

8、Timer1 中断控制

Timer1 只有一个中断，但是在间隔定时、方波输出、PWM 输出、外部事件计数模式下，产生 Timer1 中断的条件不同：

1)、在间隔定时、方波输出、外部事件计数模式下：当 Timer1 计数器值等于(T1CMP - 1)时，会产生 Timer1 中断。

2)、在 PWM 输出模式下：当 Timer1 计数器值溢出，即等于 0xFF 时，会产生 Timer1 中断。

T1CR 寄存器的 INTEN 位是 Timer1 中断使能位，T1SR 寄存器的 INTSR 位是 Timer1 中断标志位

9、使能外设模块中断

使能寄存器 IRQ3ER 的 bit1 位，IRQ3ER_TCM1INT = 0x02。

10、使能内核中断

1)、EX3 = 1; //使能内核中断 3

2)、配置内核中断 3 中断优先级 IP1_bit[7:6]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

11、Timer1 使能

置 T1CR_TEN = 1 即可使能 Timer1 开始工作(注意：在初始化定时器或 Timer1 关闭情况下，给 T1CMP 寄存器赋值后必须等待>=3*$T_{所选时钟源}$的时间后才能 T1CR_TEN=1 使能 T1 工作，$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T1CKCR_CLKSEL 的时钟)。

Timer1 PWM 输出工作时序图(T1CMP ≠ 0)

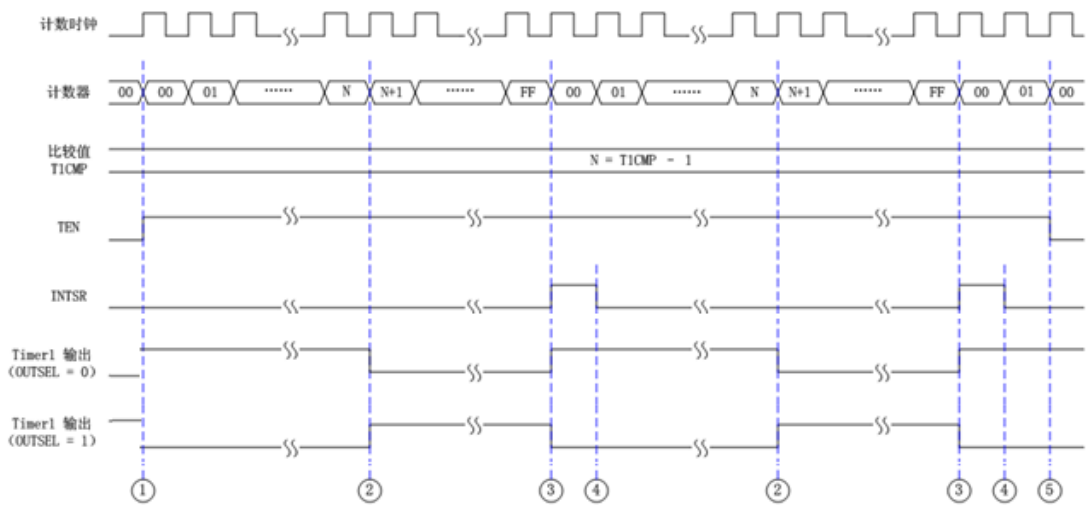


图 10-2、Timer1 PWM 输出工作时序图

波形说明：

① 使能定时器，计数器开始计数；

② 计数器值等于 N，Timer1 输出翻转；

③ 计数器值等于 0xFF，产生 Timer1 中断(即 T1SR 寄存器的 INTSR 位置 1)，同时 Timer1 输出翻转；

④ 软件处理并清除中断；

⑤ 不使能定时器，计数器清 0 并停止计数。

PWM 周期 = (0xFF + 1) * Timer1 时钟周期 = 256 * Timer1 计数时钟分频 * Timer1 计数时钟周期。

PWM 占空比 = T1CMP /256 或 （256-T1CMP）/ 256。

注：

1、当 T1CMP = 0 时，按 T1CMP = 256 计算，此时产生方波。

2、若需要产生可变占空比的 PWM 波形，可以修改 T1CMP 寄存器。软件修改 T1CMP 寄存器后，在下个 PWM 周期才会生效。

外部事件计数的操作步骤

配置 Timer1 为外部事件计数功能的操作步骤如下：

1、端口配置

1)、P10IOCFG = 0x02;//通过设置 PxyIOCFG 寄存器将 I/O 配置为定时器 1 输出/输入功能

2)、P10E_0 = 0x01;//配置 PxOE_0 方向寄存器为输入（外部时钟计数输入）

注：操作 IO 端口寄存器前应先使能 IO 模块的时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1。

2、Timer 模块时钟使能

MDLCKCR_TIMEREN = 0x04;//将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer1

当 Timer1 在停止工作状态下才能完成 Timer1 的以下配置，T1CR_TEN = 0 即关闭 Timer1。

4、清中断标志：T1SR = 0x00 ；

5、匹配值设置

在外部事件计数模式下，Timer1 计数器在定时器 1 输入引脚的每个有效沿都会加 1，当计数器值等于(T1CMP - 1)时，会产生中断。另外 T1CMP ≠ 1，T1CMP = 0 时，按 256 计算。

注：在定时器模式下，Timer1 工作期间不能修改 T1CMP，要修改 T1CMP 必须先把 T1CR 寄存器的 TEN 位清零，等待 T1SR 寄存器的 WREN 位置 1 后，再修改 T1CMP；

6、Timer1 计数时钟参数配置

- 外部信号的有效沿选择：

  T1CKCR 寄存器的 CLKDIV[3:0]位可以对外部信号的有效沿进行选择，有上升沿、下降沿、双边沿。此时不能配置为计数时钟分频选项。

- 计数时钟使能：

  将 T1CKCR 寄存器的 CLKEN 位置 1，使能 Timer1 计数时钟。

7、工作模式选择

将 T1CR 寄存器的 MODE 位清 0，工作模式选择为定时器模式。

8、Timer1 中断控制

Timer1 只有一个中断，但是在间隔定时、方波输出、PWM 输出、外部事件计数模式下，产生 Timer1 中断的条件不同：

1)、在间隔定时、方波输出、外部事件计数模式下：当 Timer1 计数器值等于(T1CMP - 1)时，会产生 Timer1 中断。

2)、在 PWM 输出模式下：当 Timer1 计数器值溢出，即等于 0xFF 时，会产生 Timer1 中断。

T1CR 寄存器的 INTEN 位是 Timer1 中断使能位，T1SR 寄存器的 INTSR 位是 Timer1 中断标志位

9、使能外设模块中断

使能寄存器 IRQ3ER 的 bit1 位，IRQ3ER_TCM1INT = 0x02。

10、使能内核中断

1)、EX3 = 1; //使能内核中断 3

2)、配置内核中断 2 中断优先级 IP1_bit[7:6]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

11、Timer1 使能

置 T1CR_TEN = 1 使能 Timer1 开始工作(注意:在初始化定时器或 Timer1 关闭情况下,给 T1CMP 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T1CR_TEN=1 使能 T1 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T1CKCR_CLKSEL 的时钟)。

注：可以直接读 T1CNT 寄存器的值，得到当前的外部时钟的计数个数。

Timer1 外部时钟计数工作时序图

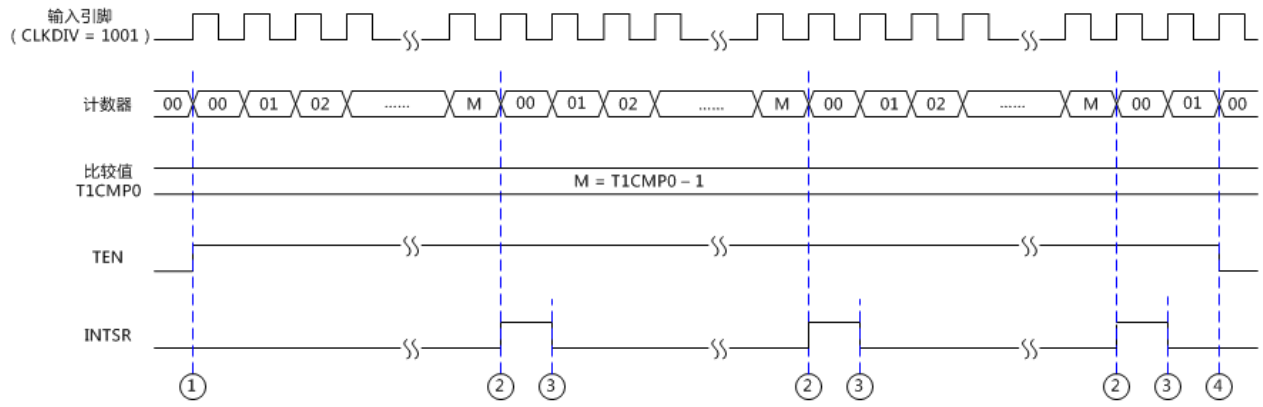


图 10-3、Timer1 外部时钟计数工作时序图

波形说明：

① 使能定时器，计数器开始计数；

② 当计数器值等于 M，产生 Timer1 中断(即 T1SR 寄存器的 INTSR 位置 1)；

③ 软件处理并清除中断；

④ 不使能定时器，计数器清 0 并停止计数。

注：1、当 T1CMP = 0 时，按 T1CMP = 256 计算，此时 M = T1CMP - 1 = 255。

2、外部输入信号的周期必须大于 Timer1 计数时钟周期的 2 倍，即外部输入信号的频率必须小于 Timer1 计数时钟频率的 1/2；

3、计数器在每个输入引脚的有效沿都会加 1；

4、T1CKCR 寄存器的 CLKDIV[3:0]位可以选择有效沿。

5、T1CMP = 1 时，不产生 Timer1 中断。

10.2、Timer1 寄存器的定义

表 10-3、Timer1 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
T1CKCR	0x78FF	XRAM	Timer1时钟控制寄存器	0000 0000
T1CR	0x79FF	XRAM	Timer1模式控制寄存器	0000 0000
T1CMP	0x7AFF	XRAM	Timer1比较寄存器	0000 0000
T1SR	0x7BFF	XRAM	Timer1状态寄存器	0000 0010
T1CNT	0x7DFF	XRAM	Timer1计数寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

T1CKCR：Timer1 时钟控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	CLKEN	CLKSEL	CLKDIV3	CLKDIV2	CLKDIV1	CLKDIV0
U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6　保留未用

bit 5　CLKEN：Timer1 计数时钟使能位

0 = 关闭 Timer1 计数时钟

1 = 打开 Timer1 计数时钟

bit 4　CLKSEL：Timer1 计数时钟选择位

0 = 选择高速时钟（系统副时钟 SCLK1）作为 Timer0 计数时钟

1 = 选择低速时钟（系统副时钟 SCLK2、内部低速 800K）作为 Timer0 计数时钟(具体是 800K 还是外部晶振需要配置 MDLCKCR_TLCKSEL 来决定，详情请参考“4.2、MDCLKCR 寄存器”描述)。

bit 3-0　CLKDIV[3:0]：Timer1 计数时钟分频或外部信号有效沿选择位

0000 = 2 分频

0001 = 4 分频

0010 = 8 分频

0011 = 16 分频

0100 = 32 分频

0101 = 64 分频

0110 = 128 分频

0111 = 256 分频

1000 = 1 分频

1001 = 外部输入信号的上升沿

1010 = 外部输入信号的下降沿

1011 = 外部输入信号的双边沿

1100~1111 = 1 分频

T1CR：Timer1 模式控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
READMODE	INVEN	OUTINIT1	OUTINIT0	INTEN	OUTSEL	MODE	TEN
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 READMODE：读 timer1 计数器模式选择，如果要读取 T1CNT 的值，则必须根据 Timer1 的时钟配置此位为正确的模式值，否则读到的 T1CNT 值与真实值之间可能存在误差

0 = Timer1 的时钟是 M 级时钟源

1 = Timer1 的时钟是 K 级时钟源（32.768K、100K、800K 或外部 M 级时钟经过 XCLKDIV 分频后的 K 级时钟）

bit 6 INVEN：Timer1 输出电平翻转使能位

0 = 当 Timer1 计数器与 T1CMP 匹配时，输出不翻转

1 = 当 Timer1 计数器与 T1CMP 匹配时，输出翻转

bit 5-4 OUTINIT[1:0]：Timer1 输出初始状态(定时器模式下才有效)

00 = 保持原状态

01 = 初始输出高电平

10 = 初始输出低电平

11 = 保持原状态

bit 3 INTEN：Timer1 中断使能位

0 = 禁止 Timer1 中断

1 = 使能 Timer1 中断

bit 2 OUTSEL：Timer1 输出初始电平选择位（PWM 模式下才有效）

0 = Timer1 输出初始电平为低电平

1 = Timer1 输出初始电平为高电平

bit 1 MODE：Timer1 工作模式选择位

0 = 定时器模式

1 = PWM 模式

bit 0 TEN：Timer1 使能位

0 = Timer1 停止工作

1 = Timer1 开始工作

T1CMP：Timer1 比较寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T1CMP[7:0]：Timer1 比较寄存器

Timer1 比较寄存器，定时器模式下，在 Timer1 工作期间，不能修改此寄存器；PWM 模式下，可以修改。

注：1、在定时器模式和 PWM 模式下，可以直接修改 T1CMP 寄存器；

2、在初始化 T1 或定时器关闭情况，设置 T1CMP 后需要延时等待>=3*$T_{所选时钟源}$的时间才能使能 T1 工作，$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T1CKCR_CLKSEL 的时钟。

例程 10-1：定时器模式下，修改 T1CMP 寄存器：

```
T1CMP = 0xA0; //修改 T1CMP 寄存器
```

T1SR：Timer1 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	-	INTSR
U-x	U-x	U-x	U-x	U-x	U-x	U-x	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1　　保留未用

bit 0　　INTSR：Timer1 中断标志位

0 = 没有产生 Timer1 中断条件

1 = 产生了 Timer1 中断条件，需软件写 0 清 0

T1CNT：Timer1 计数寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　T1CNT[7:0]：Timer1 当前计数值

读寄存器 T1CNT，可以得到 timer1 当前的计数值大小，但必须配合 T1CR 的 READMODE 位使用，如下例程。

（重启定时器可以清零计数值重新开始计时：T1CR_TEN = 0x00 ; T1CR_TEN = 0x01 ;）

例程 10-2：Timer1 的时钟是 M 级时钟时，读 T1CNT 计数值

```
T1CR_TEN = 0x00;//初始化 timer1 配置
T1CR_READMODE = 0x00;//M 级时钟
T1CR_TEN = 0x01;
……
temp = T1CNT;//读 Timer1 计数值
……
```

例程 10-3：Timer1 的时钟是 K 级时钟时，读 T1CNT 计数值

```
T1CR_TEN = 0x00;//初始化 timer1 配置
T1CR_READMODE = 0x80; //K 级时钟
T1CR_TEN = 0x01;
……
temp = T1CNT;//读 Timer1 计数值
……
```

10.3、Timer1 中断

产生 Timer1 中断有以下两种方式：

- 间隔定时/方波输出/外部事件计数模式下，当 Timer1 计数器值等于(T1CMP - 1)时；
- PWM 模式下，当 Timer1 计数器值溢出，即等于 0xFF 时。

只要有 Timer1 中断条件产生，不论是否使能 Timer1 中断，T1SR 寄存器的 INTSR 位(Timer1 中断标志位)都会置 1，INTSR 位必须在软件中清零。Timer1 的中断使能位是 T1CR 寄存器的 INTEN 位。

注：1、在休眠模式下，Timer1 中断无法唤醒处理器；

2、在空闲模式下，Timer1 中断可以唤醒处理器。

11、16 位定时器 Timer2

Timer2 模块是一个 16 位定时器，具有如下特点：

- 16 位定时器
- 3 位预分频器
- 16 位周期寄存器(T2CMP0H、T2CMP0L)
- 16 位 PWM 脉宽调制寄存器(T2CMP1H、T2CMP1L)
- Timer2 有高速时钟、低速时钟可供选择如下：高速时钟（系统副时钟 SCLK1）、低速时钟（系统副时钟 SCLK2、内部低速 800K、由内部 800k 固定 8 分频来的 100K）
- 支持实时读取计数器值

表 11-1、Timer2 时钟构成

Timer2 时钟	可选时钟	时钟源
高速时钟	SCLK1	IHCLK（48M）、EHCLK（M 级晶振）、ILCLK（800K）
低速时钟	SCLK2	EHCLK（M 级晶振）、ELCLK（K 级晶振）、
	内部低速时钟 800K	ILCLK（800K）
	内部 100K	由内部 800k 固定 8 分频来的 100K

Timer2 时钟框图详情请参考“系统总时钟框图”描述。

注：T2 定时器的时钟源选择与以下寄存器位有关：T2CKCR_CLKSEL、MDLCKCR_TLCKSEL、MISCR2_RTCCLKSEL，参考下表 11-2。

表 11-2：T2 时钟源选择寄存器关系表

T2CKCR_CLKSEL	MDLCKCR_TLCKSEL	MISCR2_RTCCLKSEL	T2 定时器所选时钟源
0	x	x	SCLK1
1	0	x	内部低速时钟 800K
1	1	0	内部 100K(由内部低速时钟 800K 固定 8 分频来)
		1	SCLK2

11.1、Timer2 支持的功能

Timer2 支持以下 3 种功能：

- 间隔定时(定时器模式)
- 方波输出（定时器模式输出）
- PWM 输出（PWM 模式输出）

Timer2 与 Timer0 的工作原理及配置相同，不同就是 Timer0 是 8 位定时器，Timer2 是 16 位定时器。

间隔定时的操作步骤

配置 Timer2 为间隔定时功能的操作步骤如下：

1、Timer2 模块时钟使能

MDLCKCR_TIMEREN = 0x04;//将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

2、关闭 Timer2

当 Timer2 在停止工作状态下才能完成 Timer2 的以下配置，T2CR_TEN = 0 即关闭 Timer2。

3、清中断标志：T2SR = 0x00 ；

4、间隔时间配置

Timer2 的间隔定时时间是通过 T2CMP0H:T2CMP0L 寄存器配置，具体的间隔时间计算如下：

间隔时间 = (T2CMP0H:T2CMP0L) * Timer2 时钟周期

= (T2CMP0H:T2CMP0L) * Timer2 计数时钟分频 * Timer2 计数时钟周期。

Timer2 计数时钟周期由选择的计数时钟决定(“第 2 步_计数时钟选择”)，共有四种时钟：

- 内部高速 OSC48M 时钟
- 内部低速 800K
- 外部 32.768K\100K 晶振(必须确保外接了 32.768K\100K 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振）

注：必须先配置 T2CMP0L,再配置 T2CMP0H，新的值将在下一个周期生效。

5、Timer2 计数时钟参数配置

- 计数时钟选择

1)、T2CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer2 的时钟；(0：高速时钟；1：低速时钟)

2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer2 的时钟。

- 计数时钟分频配置

T2CKCR 寄存器的 CLKDIV[3:0]位可以对选择的时钟源进行分频。

- 计数时钟使能

T2CKCR_CLKEN = 0x40;//将 T2CKCR 寄存器的 CLKEN 位置 1，使能 Timer2 计数时钟。

6、工作模式选择

Timer2 有定时器/方波、PWM 两种工作模式,通过 T2CR 寄存器的 MODE 位控制;这两种模式都可以实现间隔定时中断功能。

7、Timer2 模块中断控制

Timer2 只有一种中断，就是当 Timer2 计数器值等于(T2CMP0H: T2CMP0L)时，产生 Timer2 中断。在间隔定时、方波输出、PWM 输出三种模式下都可以产生 Timer2 中断。T2CR 寄存器的 INTEN 位是 Timer2 中断使能位,T2SR 寄存器的 INTSR 位是 Timer2 中断标志位。

8、使能外设模块中断

使能寄存器 IRQ4ER 的 bit1 位，IRQ4ER_TCM2INT = 0x02。

9、使能内核中断

1)、EX4 = 1; //使能内核中断 4

2)、配置内核中断 4 中断优先级 IP2_bit[1:0]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

10、Timer2 使能

置 T2CR_TEN = 1 即可使能 Timer2 开始工作(注意：在初始化定时器或 Timer2 关闭情况下，给 T2CMP0L、T2CMP0H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T2CR_TEN=1 使能 T2 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T2CKCR_CLKSEL 的时钟)。

方波输出的操作步骤

配置 Timer2 为方波输出功能的操作步骤如下：

1、端口配置

P10IOCFG = 0x26;//配置 PxyIOCFG 寄存器将 I/O 配置成定时器 0 输出功能，PxOE_y 方向寄存器会自动配置为输出

注：操作 IO 端口寄存器前应先使能 IO 模块的时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1

2-10、与以上 timer2 的“间隔定时的操作步骤”相同。(注意：在初始化定时器或 Timer2 关闭情况下，给 T2CMP0L、T2CMP0H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T2CR_TEN=1 使能 T2 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T2CKCR_CLKSEL 的时钟)。

PWM 输出的操作步骤

配置 Timer2 为方波输出功能的操作步骤如下：

1、端口配置：P10IOCFG = 0x26;//配置 PxyIOCFG 寄存器将 I/O 配置成定时器 0 输出功能，PxOE_y 方向寄存器会自动配置为

输出。**注：操作 IO 端口寄存器前应先使能 IO 模块的时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1**

2、Timer 模块时钟使能

MDLCKCR_TIMEREN = 0x04;//将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer2

当 Timer2 在停止工作状态下才能完成 Timer2 的以下配置，T2CR_TEN = 0 即关闭 Timer2。

4、清中断标志：T2SR = 0x00 ;

5、PWM 占空比配置

Timer2 的 PWM 占空比通过 T2CMP0H:T2CMP0L 与 T2CMP1H:T2CMP1L 寄存器值来配置；其中 T2CMP0H:T2CMP0L 寄存器的值决定了整个 PWM 的周期，而 T2CMP1H:T2CMP1L 寄存器的值决定了 PWM 电平输出的翻转时间，具体计算如下：

PWM 周期 = (T2CMP0H:T2CMP0L) * Timer2 时钟周期

= (T2CMP0H:T2CMP0L) * Timer2 计数时钟分频 * Timer2 计数时钟周期

PWM 占空比 = （T2CMP1H:T2CMP1L）/ (T2CMP0H:T2CMP0L) 或者等于（(T2CMP0H:T2CMP0L)-(T2CMP1H:T2CMP1L)）/(T2CMP0H:T2CMP0L))。（占空比公式取决于配置的初始电平的极性）

Timer2 计数时钟周期由选择的计数时钟决定(“第 3 步_计数时钟选择”)，共有四种时钟：

- 内部高速 OSC48M 时钟
- 内部低速 800K
- 外部 32.768K\100K 晶振(必须确保外接了 32.768K\100K 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振）

注：1)、当 T2CMP0H:T2CMP0L = 0 时，按 T2CMP0H:T2CMP0L = 0x10000 计算；

2)、当 T2CMP0H:T2CMP0L = 1 时，没有方波输出，也不产生 Timer2 中断；

3)、必须先配置 T2CMP0L,再配置 T2CMP0H，再配置 T2CMP1L,最后再配置 T2CMP1H，新的值将在下一个周期生效。

6、Timer2 计数时钟参数配置

- 计数时钟选择

  1)、T2CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer2 的时钟；(0：高速时钟；1：低速时钟)

  2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer2 的时钟。

- 计数时钟分频配置

  T2CKCR 寄存器的 CLKDIV[3:0]位可以对选择的时钟源进行分频。

- 计数时钟使能

  T2CKCR_CLKEN = 0x40;//将 T2CKCR 寄存器的 CLKEN 位置 1，使能 Timer2 计数时钟。

7、Timer2 模式配置

- 工作模式选择：T2CR_MODE = 0x00;//T2CR 寄存器的 MODE 位清 0，将工作模式选择为定时器/方波模式；
- 输出初始电平选择:T2CR 寄存器的 OUTSEL 位是 Timer2 输出初始电平选择位,对于方波输出和 PWM 输出都有效,OUTSEL 位决定了输出初始电平的极性。

8、中断控制

Timer2 只有一种中断，就是 Timer2 计数器值等于(T2CMP0H:T2CMP0L - 1)时，产生 Timer2 中断。在间隔定时、方波输出、PWM 输出三种模式下都可以产生 Timer2 中断。T2CR 寄存器的 INTEN 位是 Timer2 中断使能位，T2SR 寄存器的 INTSR 位是 Timer2 中断标志位。

9、使能外设模块中断

使能寄存器 IRQ4ER 的 bit2 位，IRQ4ER_TCM2INT = 0x04;

10、使能内核中断

1)、EX4 = 1; //使能内核中断 4

2)、配置内核中断 4 中断优先级 IP2_bit[1:0]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

11、Timer2 使能

置 T2CR_TEN = 1 即可使能 Timer2 开始工作(注意:在初始化定时器或 Timer2 关闭情况下,给 T2CMP0L、T2CMP0H、T2CMP1L、T2CMP1H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T2CR_TEN=1 使能 T2 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T2CKCR_CLKSEL 的时钟)。

11.2、Timer2 寄存器的定义

表 11-3、Timer2 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
T2CKCR	0x80FF	XRAM	Timer2时钟控制寄存器	0000 0000
T2CR	0x81FF	XRAM	Timer2模式控制寄存器	0000 0000
T2CMP0L	0x82FF	XRAM	Timer2比较寄存器0低8位	0000 0000
T2CMP0H	0x83FF	XRAM	Timer2比较寄存器0高8位	0000 0000
T2CMP1L	0x84FF	XRAM	Timer2比较寄存器1低8位	0000 0000
T2CMP1H	0x85FF	XRAM	Timer2比较寄存器1高8位	0000 0000
T2SR	0x86FF	XRAM	Timer2状态寄存器	0000 0010
T2CNTL	0xAAFF	XRAM	Timer2计数寄存器低8位	0000 0000
T2CNTH	0xABFF	XRAM	Timer2计数寄存器高8位	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

T2CKCR：Timer2 时钟控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	CLKEN	CLKSEL	-	CLKDIV2	CLKDIV1	CLKDIV0
U-x	U-x	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 保留未用

bit 5 CLKEN：Timer2 计数时钟使能位

0 = 关闭 Timer2 计数时钟

1 = 打开 Timer2 计数时钟

bit 4 CLKSEL：Timer2 计数时钟选择位

0 = 选择高速时钟（系统副时钟 SCLK1）作为 Timer0 计数时钟

1 = 选择低速时钟（系统副时钟 SCLK2、内部低速 800K）作为 Timer0 计数时钟(具体是 800K 还是外部晶振需要配置 MDLCKCR_TLCKSEL 来决定，详情请参考“4.2、MDCLKCR 寄存器”描述)。

bit 3 保留未用

bit 2-0 CLKDIV[2:0]：Timer2 计数时钟分频选择位

000 = 2 分频

001 = 4 分频

010 = 8 分频

011 = 16 分频

100 = 32 分频

101 = 64 分频

110 = 128 分频

111 = 1 分频

T2CR：Timer2 模式控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	READMODE	INTEN	OUTSEL	MODE	TEN
U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-5 保留未用

bit 4 READMODE：读 timer2 计数器模式选择，如果要读取 T2CNT 的值，则必须根据 Timer2 选择的时钟配置此位为正确的模式值，否则读到的 T2CNT 值与真实值之间可能存在误差

0 = Timer2 的时钟是 M 级时钟源

1 = Timer2 的时钟是 K 级时钟源（32.768K、100K、800K 或外部 M 级时钟经过 XCLKDIV 分频后的 K 级时钟）

bit 3 INTEN：Timer2 中断使能位

0 = 禁止计数器与 T2CMP0H、T2CMP0L 匹配中断

1 = 使能计数器与 T2CMP0H、T2CMP0L 匹配中断

bit 2 OUTSEL：Timer2 输出初始电平选择位

0 = Timer2 输出初始电平为低电平

1 = Timer2 输出初始电平为高电平

bit 1 MODE：Timer2 工作模式选择位

0 = 定时器模式

1 = PWM 模式

bit 0 TEN：Timer2 使能位

0 = Timer2 停止工作

1 = Timer2 开始工作

T2CMP0L：Timer2 比较寄存器 0 低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T2CMP0L[7:0]：Timer2 比较寄存器 0 低 8 位

T2CMP0H：Timer2 比较寄存器 0 高 8 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T2CMP0H[7:0]：Timer2 比较寄存器 0 高 8 位

Timer2 比较寄存器 0 高 8 位，在 Timer2 工作期间，可以修改此寄存器

注：1、T2CMP0H 和 T2CMP0L 组成 16 位的比较寄存器 0；

2、在 Timer2 工作期间，可以改写 T2CMP0L 和 T2CMP0H 寄存器，但必须先修改 T2CMP0L 再修改 T2CMP0H；

3、在初始化 T2 或定时器关闭情况下,设置 T2CMP0L、T2CMP0H、T2CMP1L、T2CMP1H 后需要延时等待>=3*$T_{所选时钟源}$的时间才能使能 T2 工作，$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T2CKCR_CLKSEL 的时钟。

例程 11-1：修改 T2CMP0L、T2CMP0H 寄存器：

```
T2CMP0L = 0x60;                    //修改 T2CMP0L 寄存器
T2CMP0H = 0x02;                    //修改 T2CMP0H 寄存器
```

T2CMP1L：Timer2 比较寄存器 1 低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T2CMP1L<7:0>：Timer2 比较寄存器 1 低 8 位

T2CMP1H：Timer2 比较寄存器 1 高 8 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T2CMP1H<7:0>：Timer2 比较寄存器 1 高 8 位

Timer2 比较寄存器 1 高 8 位，在 Timer2 工作期间，可以修改此寄存器。

T2CMP0H、T2CMP0L 决定输出 PWM 的周期，T2CMP1H、T2CMP1L 决定输出 PWM 的占空比，在 Timer2 工作期间，可以修改 T2CMP1H、T2CMP1L 寄存器改变输出 PWM 的占空比，**但必须先修改 T2CMP0L 再修改 T2CMP0H。**

注：1、T2CMP1H 和 T2CMP1L 组成 16 位的比较寄存器 1；

2、写 T2CMP0H 和 T2CMP0L、T2CMP1H 和 T2CMP1L 在硬件上做了保护，必须写 T2CMP0H\T2CMP1H 后配置才会有效，如果只写 T2CMP0L、T2CMP1L，配置不会生效， 占空比不变，所以必须同时更新 T2CMP0H 和 T2CMP0L、T2CMP1H 和 T2CMP1L；

3、必须先修改 T2CMP0L、T2CMP0H，再修改 T2CMP1L、T2CMP1H，否则可能出错；

4、在初始化 T2 或定时器关闭情况下，设置完 T2CMP0L、T2CMP0H、T2CMP1L、T2CMP1H 后需要延时等待>=3*T 所选时钟源的时间才能使能 T2 工作，$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T2CKCR_CLKSEL 的时钟。

T2SR：Timer2 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	-	INTSR
U-x	U-x	U-x	U-x	U-x	U-x	U-x	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 保留未用

bit 0 INTSR：Timer2 计数器与 T2CMP0L/H 匹配中断标志位

0 = Timer2 计数器与 T2CMP0L/H 不匹配

1 = 发生了 Timer2 计数器与 T2CMP0L/H 匹配，需软件写 0 清 0

T2CNTL：Timer2 计数寄存器低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T2CNTL[7:0]：Timer2 当前计数值低 8 位，与 T2CNTH 组成 16 位计数器

T2CNTH：Timer2 计数寄存器高 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T2CNTH[7:0]: Timer2 当前计数值高 8 位，与 T2CNTL 组成 16 位计数器

读寄存器 T2CNT 可以获得 Timer2 的当前计数值，读 T2CNT 前需要配置正确的 T2CR 的 READMODE 值，如下例程。

（重启定时器可以清零计数值重新开始计时：T2CR_TEN = 0x00 ; T2CR_TEN = 0x01 ;）

例程 11-2：Timer2 的时钟是 M 级时钟时，读 T2CNT 计数值

```
//初始化 timer2 配置
T2CR_TEN = 0x00;
T2CR_READMODE = 0x00;//M 级时钟
T2CR_TEN = 0x01;
……
temp L= T2CNTL;//读 Timer1 计数值低 8 位
temp H= T2CNTH;//读 Timer1 计数值高 8 位
```

例程 11-3：Timer2 的时钟是 K 级时钟时，读 T2CNT 计数值

```
//初始化 timer2 配置
T2CR_TEN = 0x00;
T2CR_READMODE = 0x10; //K 级时钟
T2CR_TEN = 0x01;
……
temp L= T2CNTL;//读 Timer1 计数值低 8 位
temp H= T2CNTH;//读 Timer1 计数值高 8 位
……
```

11.3、Timer2 中断

不论是在定时器模式还是 PWM 模式下，当 Timer2 计数器值等于(T2CMP0H:T2CMP0L - 1)时，都会产生 Timer2 中断。

只要有 Timer2 中断条件产生，不论是否使能 Timer2 中断，T2SR 寄存器的 INTSR 位(Timer2 中断标志位)都会置 1，INTSR 位必须在软件中清零。Timer2 的中断使能位是 T2CR 寄存器的 INTEN 位。

注：

1、在休眠模式下，Timer2 中断无法唤醒处理器。

2、在空闲模式下，Timer2 中断可以唤醒处理器。

12、16 位定时器 Timer3

Timer3 模块是一个 16 位定时器，具有如下特点：

- 16 位定时器
- 3 位预分频器
- 16 位周期寄存器(T3CMP0H、T3CMP0L)
- 16 位 PWM 脉宽调制寄存器(T3CMP1H、T3CMP1L)
- 16 位捕捉功能寄存器(T3CMP0H、T3CMP0L、T3CMP1H、T3CMP1L)
- 可编程外部时钟边沿选择
- Timer3 有高速时钟、低速时钟可供选择如下：高速时钟（系统副时钟 SCLK1）、低速时钟（系统副时钟 SCLK2、内部低速 800K、由内部 800k 固定 8 分频来的 100K）
- 支持实时读取计数器值

表 12-1、Timer3 时钟构成

Timer3 时钟	可选时钟	时钟源
高速时钟	SCLK1	IHCLK（48M）、EHCLK（M 级晶振）、ILCLK（800K）
低速时钟	SCLK2	EHCLK（M 级晶振）、ELCLK（K 级晶振）、
	内部低速时钟 800K	ILCLK（800K）
	内部 100K	由内部 800k 固定 8 分频来的 100K

Timer3 时钟框图详情请参考“系统总时钟框图”描述。

注：T3 定时器的时钟源选择与以下寄存器位有关：T3CKCR_CLKSEL、MDLCKCR_TLCKSEL、MISCR2_RTCCLKSEL，参考下表 12-2。

表 12-2：T3 时钟源选择寄存器关系表

T3CKCR_CLKSEL	MDLCKCR_TLCKSEL	MISCR2_RTCCLKSEL	T3 定时器所选时钟源
0	x	x	SCLK1
1	0	x	内部低速时钟 800K
1	1	0	内部 100K(由内部低速时钟 800K 固定 8 分频来)
		1	SCLK2

12.1、Timer3 的支持的功能

Timer3 支持以下 6 种功能：

- 间隔定时（定时器模式）
- 方波输出（定时器模式输出）
- PWM 输出（PWM 模式输出）
- 外部事件计数(定时器模式下对外部输入电平跳变计数)
- 可编程脉冲输出
- 外部电平宽度测量

间隔定时的操作步骤

配置 Timer3 为间隔定时功能的操作步骤如下：

1、Timer 模块时钟使能

将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

2、关闭 Timer3

当 Timer3 在停止工作状态下才能完成 Timer3 的以下配置，T3CR_MODE = 0 即关闭 Timer3。

3、清中断标志：T3SR = 0x00 ；

4、间隔时间配置

Timer3 的间隔定时时间可以通过(T3CMP0H：T3CMP0L)寄存器配置，具体的间隔时间计算如下：

间隔时间 = (T3CMP0H：T3CMP0L) * Timer3 时钟周期

= (T3CMP0H：T3CMP0L) * Timer3 计数时钟分频 * Timer3 计数时钟周期

Timer3 计数时钟周期由选择的计数时钟决定，共有三种时钟：

- 内部高速 OSC48M
- 内部低速 800K
- 外部 32.768K 晶振(必须确保外接了 32.768K 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振）

注：必须先配置 T3CMP0L，再配置 T3CMP0H，新的值将在下一个周期生效。

5、Timer3 计数时钟参数配置

- 计数时钟选择：

1)、T3CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer0 的时钟。

2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer3 的时钟。

- 计数时钟分频配置：

T3CKCR 寄存器的 CLKDIV[2:0]位可以对选择的时钟源进行分频。

- 输入引脚的有效沿设置：

将 T3CKCR 寄存器的 EDGESEL[1:0]位配置为 00，**即输入引脚的有效沿选择为禁止设置**。

- 计数时钟使能

将 T3CKCR 寄存器的 CLKEN 位置 1，使能 Timer3 计数时钟。

6、T3CMP0、T3CMP1 寄存器组的模式选择

将 T3CR 寄存器的 CMP0M 位清 0,CMP1M 位置 1,配置 T3CMP0 寄存器组作为比较寄存器,T3CMP1 寄存器组作为捕获寄存器。

7、Timer3 模块中断控制

Timer3 有两个中断，Timer3 中断 0、Timer3 中断 1，Timer3 中断 1 只有在单脉冲输出、外部电平宽度测量模式下才可以产生，Timer3 中断 0 在所有模式下都可以产生。具体请看“12.3、Timer3 中断”。

T3CR 寄存器的 INT0EN、INT1EN 位是 Timer3 中断使能位，T3SR 寄存器的 INT0SR、INT1SR 位是 Timer3 中断标志位。

8、使能外设模块中断

使能寄存器 IRQ0ER 的 bit1 位，IRQ0ER_TCM3INT0 = 0x02，使能 Timer3 中断 0；使能寄存器 IRQ1ER 的 bit1 位，IRQ1ER_TCM3INT1 = 0x02，使能 Timer3 中断 1。

9、使能内核中断

1)、EX0 = 1,EX1 = 1; //使能内核中断 0、1

2)、配置内核中断 0、1 中断优先级 IP1_bit[1:0]、IP1_bit[3:2]（0：最低优先级；1：最高优先级）

3)、使能总内核中断 EA=1。

10、工作模式选择

置 T3CR_MODE[2:0]=110/111,即可使能 Timer3 开始工作(注意：在初始化定时器或 Timer3 关闭情况下，给 T3CMP0L、T3CMP0H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T3CR_MODE[2:0] = 110/111 使能 T3 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟)。

方波输出的操作步骤

配置 Timer3 为方波输出功能的操作步骤如下：

1、端口配置

通过 PxyIOCFG 寄存器将 I/O 配置成定时器 3 输出/输入 0 功能，并且配置 PnOE 寄存器，将定时器 3 输出或输入 0 引脚方

向置为输出。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)

2-10、与以上 timer3 的“间隔定时的操作步骤”相同。(注意：在初始化定时器或 Timer3 关闭情况下，给 T3CMP0L、T3CMP0H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T3CR_MODE[2:0] = 110/111 使能 T3 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟)。

Timer3 间隔定时/方波输出工作时序图

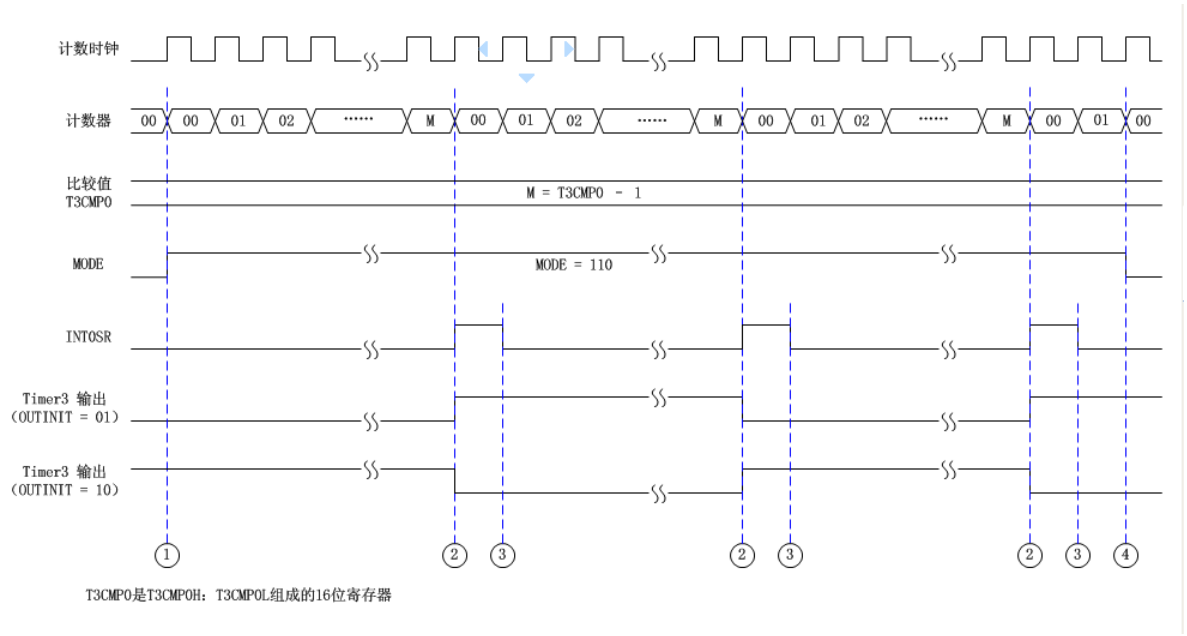


图 12-1、Timer3 间隔定时/方波输出工作时序图

波形说明：

① 设置计数器模式 MODE[2:0]位 = 110 后，计数器开始计数；

② 当计数器值等于 M，产生 Timer3 中断 0(即 T3SR 寄存器的 INT0SR 位置 1)，同时 Timer3 输出翻转；

③ 软件处理并清除中断；

④ 设置计数器模式 MODE[2:0]位 = 000 后，计数器清零并停止计数。

间隔时间 = (T3CMP0) * Timer3 时钟周期

= (T3CMP0) * Timer3 计数时钟分频 * Timer3 计数时钟周期

方波周期 = 2 *间隔时间

注：

1、当 T3CMP0 = 0 时，按 T3CMP0 = 65536 计算。

2、当 T3CMP0 = 1 时，没有方波输出，也不产生 Timer3 中断 0。

PWM 输出的操作步骤

配置 Timer3 为 PWM 输出功能的操作步骤如下：

1、端口配置：

通过 PxyIOCFG 寄存器将 I/O 配置成定时器 3 输出/输入 0 功能，并且配置 PnOE 寄存器，将定时器 3 输出或输入 0 引脚方向置为输出。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)

2、Timer 模块时钟使能：

将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer3

当 Timer3 在停止工作状态下才能完成 Timer3 的以下配置，T3CR_MODE = 0 即关闭 Timer3。

4、清中断标志：T3SR = 0x00 ；

5、PWM 占空比配置：

Timer3 的 PWM 占空比是通过 T3CMP0H：T3CMP0L 与 T3CMP1H：T3CMP1L 寄存器进行配置，具体计算如下：

PWM 周期 = (T3CMP0H:T3CMP0L) * Timer3 时钟周期

= (T3CMP0H:T3CMP0L) * Timer3 计数时钟分频 * Timer3 计数时钟周期

PWM 占空比 =(T3CMP1H:T3CMP1L)/(T3CMP0H:T3CMP0L)或者等于 ((T3CMP0H:T3CMP0L)-(T3CMP1H:T3CMP1L))/(T3CMP0H:T3CMP0L))。（占空比公式取决于配置的初始电平的极性）

Timer3 计数时钟周期由选择的计数时钟决定，共有三种时钟：

- 内部高速 OSC48M 的 3 分频
- 内部低速 800K
- 外部 32.768K 晶振(必须确保外接了 32.768K 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振）

简单地说，(T3CMP0H:T3CMP0L)是配置 PWM 输出的周期，(T3CMP1H:T3CMP1L)配置的是输出低电平的时间，所以(T3CMP1H:T3CMP1L)必须小于(T3CMP0H：T3CMP0L)。

注：1)、当(T3CMP0H ： T3CMP0L)= 0 时，按(T3CMP0H ： T3CMP0L)= 65536 计算。

2)、在 PWM 模式下，可以直接修改(T3CMP1H：T3CMP1L)寄存器来改变 PWM 的占空比，修改(T3CMP1H：T3CMP1L)时，新的占空比配置将在下个 PWM 周期生效。

3）、必须先配置 T3CMP0L，再配置 T3CMP0H，再配置 T3CMP1L，最后再配置 T3CMP1H。

6、Timer3 计数时钟参数配置：

- 计数时钟选择：

  1)、T3CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer3 的时钟。

  2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer3 的时钟。

- 计数时钟分频配置：

  T3CKCR 寄存器的 CLKDIV[2:0]位可对选择的时钟源进行分频。此时不能配置为输入引脚的有效沿作为计数时钟选项。

- 输入引脚的有效沿设置：

  将 T3CKCR 寄存器的 EDGESEL[1:0]位配置为 00，输入引脚的有效沿选择为禁止设置。

- 计数时钟使能：

  将 T3CKCR 寄存器的 CLKEN 位置 1，使能 Timer3 计数时钟。

7、Timer3 输出配置：

- 输出初始电平选择：

  T3OCR 寄存器的 OUTSEL[1:0]位是 Timer3 输出初始电平选择位，对于方波输出、PWM 输出和单脉冲输出都有效，OUTSEL 位决定了输出初始电平的极性。

- 禁止单脉冲输出模式(连续脉冲输出模式)：

  将 T3OCR 寄存器的 SPOM 位清 0，选择连续脉冲输出模式。

- 计数器与 T3CMP0、T3CMP1 寄存器组匹配输出翻转配置：

  将 T3OCR 寄存器的 INV0EN、INV1EN 位同时置 1，使能计数器与(T3CMP1H:T3CMP1L)匹配输出翻转，计数器与(T3CMP0H:T3CMP0L)匹配输出翻转。

8、T3CMP0、T3CMP1 寄存器组的模式选择：

将 T3CR 寄存器的 CMP0M、CMP1M 位清 0。配置 T3CMP0 寄存器组、T3CMP1 寄存器组都作为比较寄存器。

9、中断控制

Timer3 有两个中断，Timer3 中断 0、Timer3 中断 1：

1)、Timer3 中断 1 只有在单脉冲输出、外部电平宽度测量模式下才可以产生；

2)、Timer3 中断 0 在所有模式下都可以产生。具体请看 “12.3、Timer3 中断”。

T3CR 寄存器的 INT0EN、INT1EN 位是 Timer3 中断使能位，T3SR 寄存器的 INT0SR、INT1SR 位是 Timer3 中断标志位。

10、使能外设模块中断

使能寄存器 IRQ0ER 的 bit1 位，IRQ0ER_TCM3INT0 = 0x02，使能 Timer3 中断 0；使能寄存器 IRQ1ER 的 bit1 位，IRQ1ER_TCM3INT1 = 0x02，使能 Timer3 中断 1。

11、使能内核中断

1)、EX0 = 1,EX1 = 1; //使能内核中断 0、1

2)、配置内核中断 0、1 中断优先级 IP1_bit[1:0]、IP1_bit[3:2]（0：最低优先级；1：最高优先级）

3)、使能总内核中断 EA=1。

12、工作模式选择

置 T3CR_MODE[2:0]=110,即可使能 Timer3 开始工作(注意：在初始化定时器或 Timer3 关闭情况下，给 T3CMP0L、T3CMP0H、T3CMP1L、T3CMP1H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T3CR_MODE[2:0] = 110 使能 T3 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟)。

Timer3 PWM 输出工作时序图

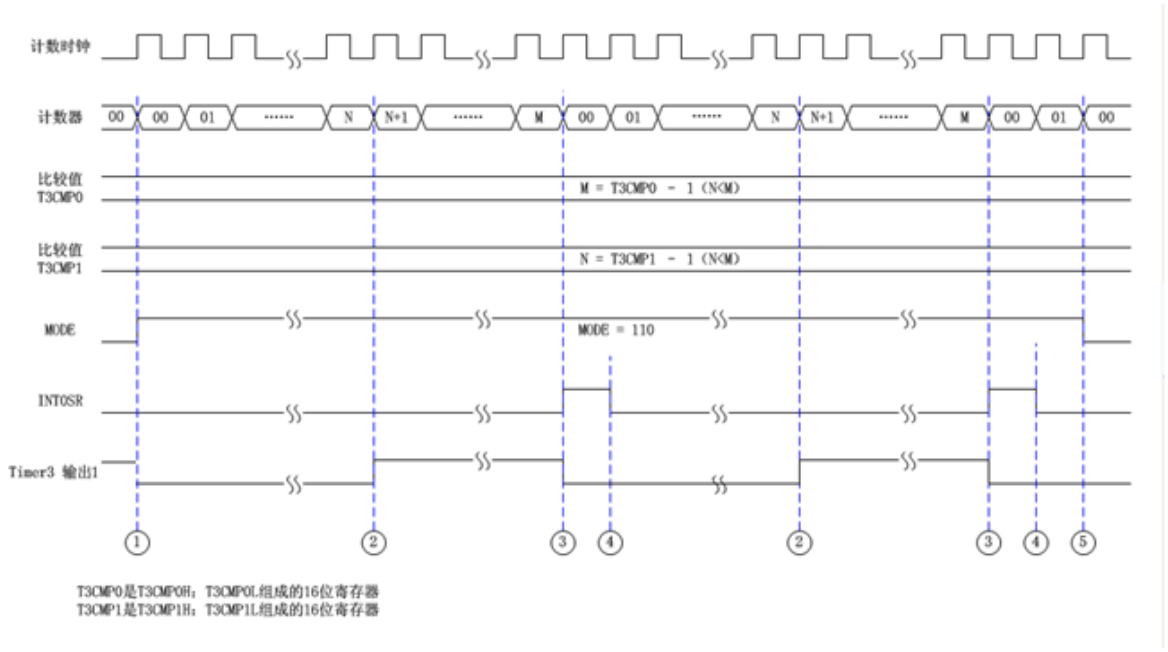


图 12-2、Timer3 PWM 输出工作时序图

波形说明：

① 设置计数器模式 MODE[2:0]位 = 110 后，计数器开始计数；

② 当计数器值等于 N，Timer3 输出翻转；

③ 当计数器值等于 M，产生 Timer3 中断 0(即 T3SR 寄存器的 INT0SR 位置 1)，同时 Timer3 输出翻转；

④ 软件处理并清除中断

⑤ 设置计数器模式 MODE[2:0]位 = 000 后，计数器清零并停止计数。

PWM 周期 = (T3CMP0) * Timer3 时钟周期

= (T3CMP0) * Timer3 计数时钟分频 * Timer3 计数时钟周期

PWM 占空比 = T3CMP1 / T3CMP0

注：T3CMP0 = 0 时，按 T3CMP0 = 65536 计算。

外部事件计数的操作步骤

配置 Timer3 为外部事件计数功能的操作步骤如下：

1、端口配置

通过 PxyIOCFG 寄存器将 I/O 配置成定时器 3 输出/输入 0 功能，并且配置 PnOE 寄存器，将定时器 3 输出或输入 0 引脚方向置为输入。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)

2、Timer 模块时钟使能

将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer3

当 Timer3 在停止工作状态下才能完成 Timer3 的以下配置，T3CR_MODE = 0 即关闭 Timer3。

4、清中断标志：T3SR = 0x00 ；

5、匹配值设置

在外部事件计数模式下，Timer3 计数器在定时器 3 输入 0 引脚的每个有效沿都会加 1，当计数器值等于(T3CMP0H:T3CMP0L - 1)时，会产生 Timer3 中断 0。T3CMP0H:T3CMP0L ≠ 1，当 T3CMP0H:T3CMP0L = 0 时，按 T3CMP0H:T3CMP0L = 65536 计算。

6、Timer3 计数时钟参数配置

- 计数时钟分频配置：

  将 T3CKCR 寄存器的 CLKDIV[2:0]位置为 1XX，计数时钟配置为输入引脚的有效沿。（此时不能配置为计数时钟分频选项。）

- 输入引脚的有效沿设置：

  T3CKCR 寄存器的 EDGESEL[1:0]位可以选择输入引脚的有效沿，有上升沿、下降沿、双边沿三种，此时不能配置为禁止设置选项。

7、禁止单脉冲输出模式(连续脉冲输出模式)

将 T3OCR 寄存器的 SPOM 位清 0，选择连续脉冲输出模式。

8、T3CMP0、T3CMP1 寄存器组的模式选择

将 T3CR 寄存器的 CMP0M 位清 0，CMP1M 位置 1，配置 T3CMP0 寄存器组作为比较寄存器，T3CMP1 寄存器组作为捕获寄存器。

9、中断控制

Timer3 有两个中断，Timer3 中断 0、Timer3 中断 1：

1)、Timer3 中断 1 只有在单脉冲输出、外部电平宽度测量模式下才可以产生；

2)、Timer3 中断 0 在所有模式下都可以产生。具体请看 “12.3、Timer3 中断”。

T3CR 寄存器的 INT0EN、INT1EN 位是 Timer3 中断使能位，T3SR 寄存器的 INT0SR、INT1SR 位是 Timer3 中断标志位。

10、使能外设模块中断

使能寄存器 IRQ0ER 的 bit1 位，IRQ0ER_TCM3INT0 = 0x02，使能 Timer3 中断 0；使能寄存器 IRQ1ER 的 bit1 位，IRQ1ER_TCM3INT1 = 0x02，使能 Timer3 中断 1。

11、使能内核中断

1)、EX0 = 1,EX1 = 1; //使能内核中断 0、1

2)、配置内核中断 0、1 中断优先级 IP1_bit[1:0]、IP1_bit[3:2]（0：最低优先级；1：最高优先级）

3)、使能总内核中断 EA=1。

12、工作模式选择

置 T3CR_MODE[2:0]=110,即可使能 Timer3 开始工作(注意：在初始化定时器或 Timer3 关闭情况下，给 T3CMP0L、T3CMP0H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T3CR_MODE[2:0] = 110 使能 T3 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟)。

Timer3 外部事件计数工作时序图

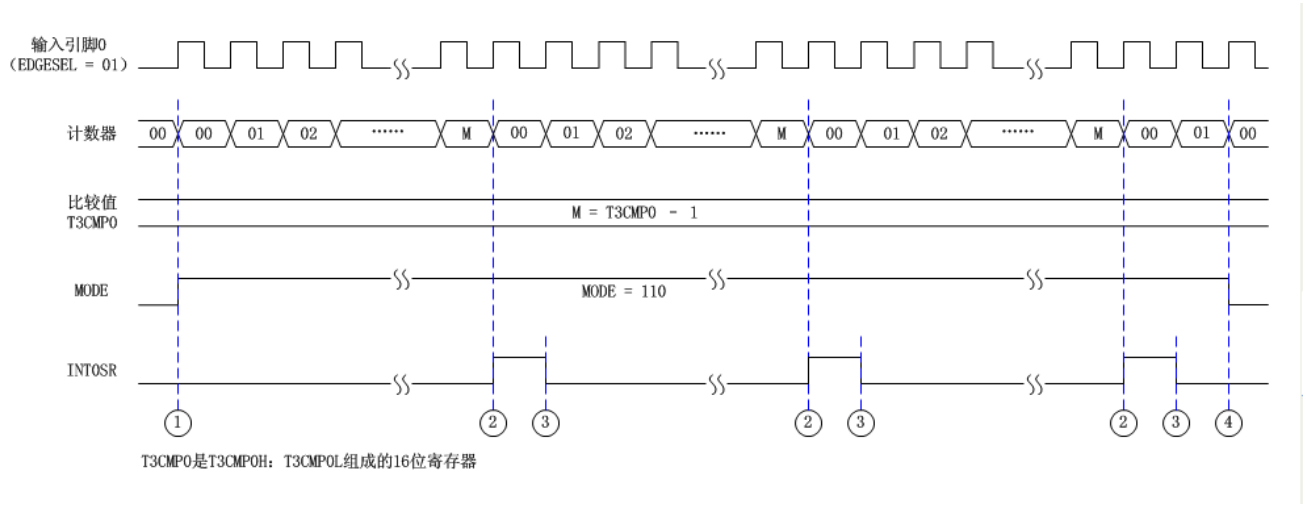


图 12-3、Timer3 外部事件计数工作时序图

波形说明：

① 设置计数器模式 MODE[2:0]位 = 110 后，计数器开始计数；

② 当计数器值等于 M 时，产生 Timer3 中断 0(即 T3SR 寄存器的 INT0SR 位置 1)；

③ 软件处理并清除中断；

④ 设置计数器模式 MODE[2:0]位 = 000 后，计数器清零并停止计数。

注：

1、当 T3CMP0 = 0 时，按 T3CMP0 = 65536 计算，此时 M = T3CMP0 - 1 = 65535。

2、计数时钟为输入引脚 0 的有效沿，只有当输入引脚脉冲宽度大于两个时钟宽度才认为是一个有效边沿，以消除干扰；

3、计数器在每个输入引脚的有效沿都会加 1；

4、T3CKCR 寄存器的 EDGESEL 位可以选择有效沿。

5、当 T3CMP0 = 1 时，不产生 Timer3 中断 0。

单脉冲输出的操作步骤

配置 Timer3 为单脉冲输出功能的操作步骤如下：

1、端口配置

通过 PxyIOCFG 寄存器将 I/O 配置成定时器 3 输出/输入 0 功能，并且配置 PnOE 寄存器，将定时器 3 输出或输入 0 引脚方向置为输出。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)

2、Timer 模块时钟使能

将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer3

当 Timer3 在停止工作状态下才能完成 Timer3 的以下配置，T3CR_MODE = 0 即关闭 Timer3。

4、清中断标志：T3SR = 0x00 ；

5、Timer3 输出配置

- 使能单脉冲输出模式：

  将 T3OCR 寄存器的 SPOM 位置 1，选择单脉冲输出模式。

- 输出初始状态选择：

  T3OCR 寄存器的 OUTINIT[1:0]位控制 Timer3 输出初始状态，只对方波输出、单脉冲输出有效，对 PWM 输出无效。

- 计数器与 T3CMP0、T3CMP1 寄存器组匹配输出翻转配置：

将 T3OCR 寄存器的 INV0EN、INV1EN 位置 1，使能计数器与(T3CMP1H：T3CMP1L)匹配输出翻转，计数器与(T3CMP0H：T3CMP0L)匹配输出翻转。

6、单脉冲宽度配置

Timer3 的单脉冲宽度是通过 T3CMP0H：T3CMP0L 与 T3CMP1H：T3CMP1L 寄存器进行配置，具体计算如下：

单脉冲宽度 = (T3CMP0H：T3CMP0L - T3CMP1H：T3CMP1L) * Timer3 时钟周期

= (T3CMP0H：T3CMP0L - T3CMP1H：T3CMP1L) * Timer3 计数时钟分频 * Timer3 计数时钟周期

Timer3 计数时钟周期由选择的计数时钟决定，共有三种时钟：

- 内部高速 OSC48M 的 3 分频
- 内部低速 800K
- 外部 32.768K 晶振(必须确保外接了 32.768K 晶振)
- 外部 M 级晶振时钟（必须接了外部 M 级晶振）

注：

1)、当(T3CMP0H ： T3CMP0L)= 0 时，按(T3CMP0H ： T3CMP0L)= 65536 计算；

2)、其中(T3CMP1H：T3CMP1L)必须小于(T3CMP0H：T3CMP0L)，当输出初始电平为低电平，单脉冲宽度是高电平宽度；当输出初始电平为高电平，单脉冲宽度是低电平宽度。

7、Timer3 计数时钟参数配置

- 计计数时钟选择：

  1)、T3CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer3 的时钟。

  2）、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer3 的时钟。

- 计数时钟分频配置：

  T3CKCR 寄存器的 CLKDIV[2:0]位可以对选择的时钟源进行分频。此时不能配置为输入引脚的有效沿作为计数时钟选项。

- 输入引脚的有效沿设置：

  将 T3CKCR 寄存器的 EDGESEL[1:0]位配置为 00，输入引脚的有效沿选择为禁止设置。

- 计数时钟使能：

  将 T3CKCR 寄存器的 CLKEN 位置 1，使能 Timer3 计数时钟。

8、T3CMP0、T3CMP1 寄存器组的模式选择

将 T3CR 寄存器的 CMP0M、CMP1M 位同时清 0，配置 T3CMP0 寄存器组、T3CMP1 寄存器组都作为比较寄存器。

9、中断控制

Timer3 有两个中断，Timer3 中断 0、Timer3 中断 1：

1)、Timer3 中断 1 只有在单脉冲输出、外部电平宽度测量模式下才可以产生；

2)、Timer3 中断 0 在所有模式下都可以产生。具体请看 “12.3、Timer3 中断”。

T3CR 寄存器的 INT0EN、INT1EN 位是 Timer3 中断使能位，T3SR 寄存器的 INT0SR、INT1SR 位是 Timer3 中断标志位。

10、使能外设模块中断

使能寄存器 IRQ0ER 的 bit1 位，IRQ0ER_TCM3INT0 = 0x02，使能 Timer3 中断 0；使能寄存器 IRQ1ER 的 bit1 位，IRQ1ER_TCM3INT1 = 0x02，使能 Timer3 中断 1。

11、使能内核中断

1)、EX0 = 1,EX1 = 1; //使能内核中断 0、1

2)、配置内核中断 0、1 中断优先级 IP1_bit[1:0]、IP1_bit[3:2]（0：最低优先级；1：最高优先级）

3)、使能总内核中断 EA=1。

12、工作模式选择

置 T3CR_MODE[2:0]=010,即可使能 Timer3 开始工作(注意:在初始化定时器或 Timer3 关闭情况下,给 T3CMP0L、T3CMP0H、T3CMP1L、T3CMP1H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T3CR_MODE[2:0]=010 使能 T3 工作,T 所选时钟源指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟)。

13、触发单脉冲输出

每次写 T3OCR 寄存器的 SPO 位为 1,即启动一次单脉冲输出,启动后应查询 T3SR 寄存器的 INT0SR 位是否置 1,判断单脉冲输出是否完成,完成一次单脉冲输出后才可以启动下一次。

Timer3 单脉冲输出工作时序图

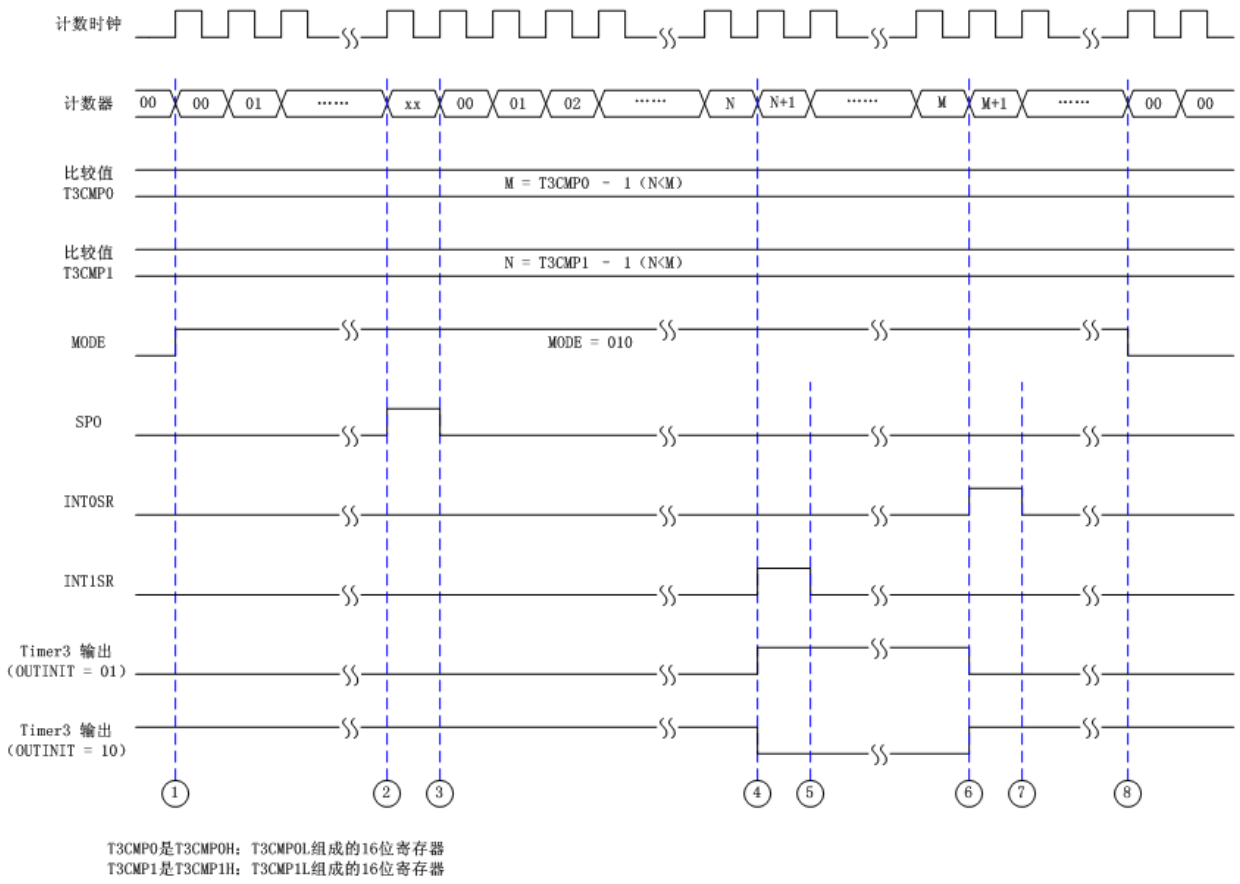


图 12-4、Timer3 单脉冲输出工作时序图

波形说明:

① 设置计数器模式 MODE[2:0]位 = 010 后,计数器开始计数;

② 软件写 SPO 为 1;

③ 延时一段时间后,计数器清零,启动单脉冲输出;

④ 当计数器值等于 N,产生 Timer3 中断 1(即 T3SR 寄存器的 INT1SR 位置 1),同时脉冲信号翻转;

⑤ 软件处理并清除中断 INT1SR;

⑥ 当计数器值等于 M,产生 Timer3 中断 0(即 T3SR 寄存器的 INT0SR 位置 1),同时脉冲信号翻转;

⑦ 软件处理并清除中断 INT0SR;

⑧ 设置计数器模式 MODE[2:0]位 = 000 后,计数器清零并停止计数。

注:写一次 SPO 为 1,只能启动 1 个脉冲输出。想发送多个脉冲必须重复写 SPO。

外部电平宽度测量

外部电平宽度有三种工作方式：

- 使用一组捕获寄存器测量一个输入电平的宽度
- 使用两组捕获寄存器测量两个输入电平的宽度
- 使用两组捕获寄存器测量一个输入电平的宽度

外部电平宽度的工作原理是当外部输入有效沿信号，会产生中断，同时计数器的值捕捉到捕捉寄存器(T3CMP0、T3CMP1两组寄存器)中；在中断中读取捕捉寄存器的值，连续两次中断读取值的差值可以得出电平宽度。

使用一组捕获寄存器测量一个输入电平的宽度

此模式下，使用的是 T3CMP1H：T3CMP1L 这组寄存器作为捕获寄存器，测量定时器 3 输入 0 的电平宽度。配置 Timer3 为使用一组捕获寄存器测量一个输入电平的宽度的操作步骤如下：

1、端口配置

通过 PxyIOCFG 寄存器将 I/O 配置成定时器 3 输出/输入 0 功能，并且配置 PnOE 寄存器，将定时器 3 输出或输入 0 引脚方向置为输入。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)

2、Timer 模块时钟使能

将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer3

当 Timer3 在停止工作状态下才能完成 Timer3 的以下配置，T3CR_MODE = 0 即关闭 Timer3。

4、清中断标志：T3SR = 0x00 ；

3、Timer3 计数时钟参数配置

- 计数时钟选择：

  1)、T3CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer3 的时钟。

  2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer3 的时钟。

- 计数时钟分频配置：

  T3CKCR 寄存器的 CLKDIV[2:0]位可以对选择的时钟源进行分频。此时不能配置为输入引脚的有效沿作为计数时钟选项。

- 输入引脚的有效沿设置：

  配置 T3CKCR 寄存器的 EDGESEL[1:0]位 = 11，选择输入引脚的有效沿为双边沿。

- 计数时钟使能：

  将 T3CKCR 寄存器的 CLKEN 位置 1，使能 Timer3 计数时钟。

4、T3CMP0、T3CMP1 寄存器组的模式选择

将 T3CR 寄存器的 CMP0M 位清 0,CMP1M 位置 1,配置 T3CMP0 寄存器组作为比较寄存器,T3CMP1 寄存器组作为捕获寄存器。

5、清 Timer3 中断标志位

使能 Timer3 中断前，先将 T3SR 寄存器的 INT0SR、INT1SR 位清 0，清 Timer3 中断标志位。

6、中断控制

Timer3 有两个中断，Timer3 中断 0、Timer3 中断 1：

1)、Timer3 中断 1 只有在单脉冲输出、外部电平宽度测量模式下才可以产生；

2)、Timer3 中断 0 在所有模式下都可以产生。具体请看 “12.3、Timer3 中断”。

T3CR 寄存器的 INT0EN、INT1EN 位是 Timer3 中断使能位，T3SR 寄存器的 INT0SR、INT1SR 位是 Timer3 中断标志位。

7、使能外设模块中断

使能寄存器 IRQ0ER 的 bit1 位，IRQ0ER_TCM3INT0 = 0x02，使能 Timer3 中断 0；使能寄存器 IRQ1ER 的 bit1 位，

IRQ1ER_TCM3INT1 = 0x02，使能 Timer3 中断 1。

8、使能内核中断

1)、EX0 = 1,EX1 = 1；//使能内核中断 0、1

2)、配置内核中断 0、1 中断优先级 IP1_bit[1:0]、IP1_bit[3:2]（0：最低优先级；1：最高优先级）

3)、使能总内核中断 EA=1。

9、工作模式选择：

1)、置 T3CR_MODE[2:0]=010,即可使能 Timer3 开始工作(注意：在初始化定时器或 Timer3 关闭情况下，给 T3CMP0L、T3CMP0H、T3CMP1L、T3CMP1H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T3CR_MODE[2:0]=010 使能 T3 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟)。

2)、配置双边沿为有效沿，如果使能了 Timer3 中断 1，在 Timer3 输入 0 引脚外部信号的上升沿、下降沿都会产生 Timer3 中断 1，连续 3 次中断就可以得出一个低电平宽度时间，一个高电平宽度时间。(具体哪个是高电平时间或低电平时间必须从实际情况分析)

Timer3 使用一组捕获寄存器测量一个输入电平的宽度工作时序图

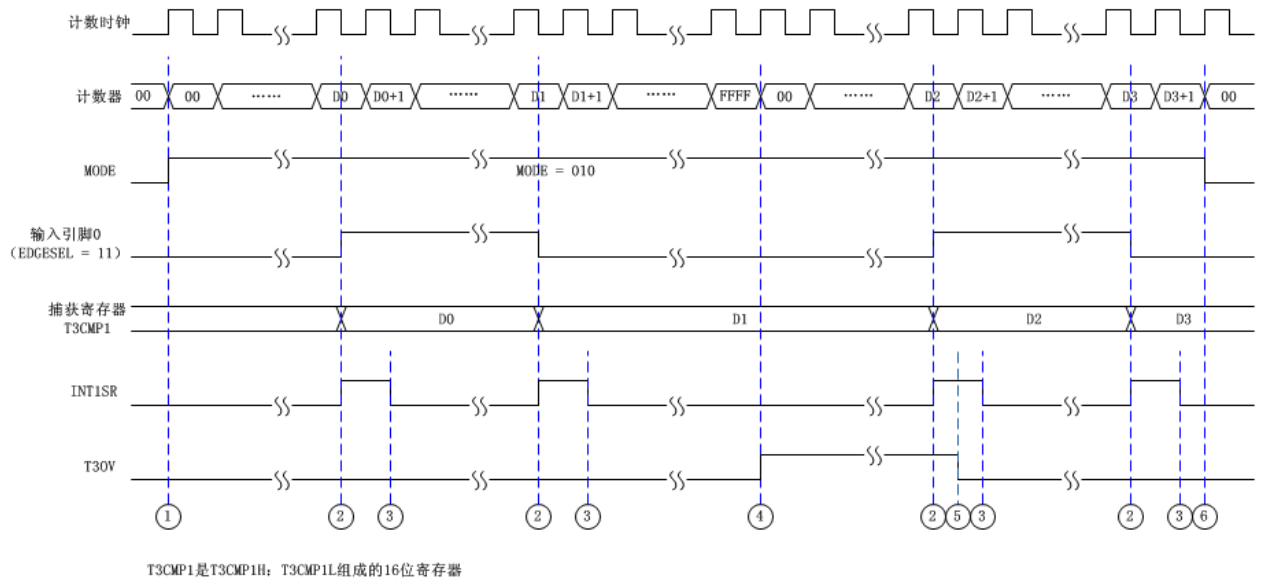


*图 12-5、Timer3 一组捕获寄存器测量一个输入电平宽度*工作时序图

波形说明：

① 设置计数器模式 MODE[2:0]位 = 010 后，计数器开始计数；

② 检测到输入引脚 0 的有效沿(这里必须设置有效沿为双边沿)，将计数器的值锁存到捕获寄存器 T3CMP1H：T3CMP1L 中，同时产生 Timer3 中断 1(即 T3SR 寄存器的 INT1SR 位置 1)；

③ 软件处理并清除中断 INT1SR；

④ 计数器溢出；软件应清除其状态。(软件在中断处理程序中读取捕获寄存器的值时，应同时关注溢出状态位，若有溢出，计算脉冲宽度时，应加上 0xFFFF)；

⑤ 软件处理后清除计数器溢出标志位；

⑥ 设置计数器模式 MODE[2:0]位 = 000 后，计数器清零并停止计数。

注：

1、输入引脚的有效沿必须设置为双边沿，这样检测到上升沿、下降沿都会产生 Timer3 中断 1，并锁存计数器的值。

2、若输入引脚是周期稳定的信号，可以根据相邻 3 次中断所捕获到的数据，得到输入引脚的信号的周期/频率。

3、理论上是无法区分出得到的是高电平宽度还是低电平宽度，必须根据实际上输入信号的特征来区分。

使用两组捕获寄存器测量两个输入电平的宽度的操作步骤

此模式下，使用的是 T3CMP0H、T3CMP0L 这组寄存器作为捕获寄存器，测量定时器 3 输入 1 的电平宽度；T3CMP1H、T3CMP1L 这组寄存器作为捕获寄存器，测量定时器 3 输入 0 的电平宽度。配置 Timer3 为使用两组捕获寄存器测量两个输入电平的宽度的操作步骤如下：

1、端口配置

1)、通过 PxyIOCFG 寄存器将 I/O 配置成定时器 3 输出/输入 0 功能，并且配置 PnOE 寄存器，将定时器 3 输出或输入 0 引脚方向置为输入。

2)、通过 PxyIOCFG 寄存器将 I/O 配置成定时器 3 输入 1 功能，此时定时器 3 输入 1 的 I/O 方向自动配置为输入。不需要配置对应的 PnOE 寄存器，选择定时器 3 输入 1 引脚方向为输入。

注：端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1。

2、Timer 模块时钟使能

将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer3

当 Timer3 在停止工作状态下才能完成 Timer3 的以下配置，T3CR_MODE = 0 即关闭 Timer3。

4、清中断标志：T3SR = 0x00 ；

3、Timer3 计数时钟参数配置

- 计数时钟选择：

  1)、T3CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer3 的时钟。

  2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer3 的时钟。

- 计数时钟分频配置：

  T3CKCR 寄存器的 CLKDIV[2:0]位可以对选择的时钟源进行分频。此时不能配置为输入引脚的有效沿作为计数时钟选项。

- 输入引脚的有效沿设置：

  配置 T3CKCR 寄存器的 EDGESEL[1:0]位 = 11，选择输入引脚的有效沿为双边沿。

- 计数时钟使能：

  将 T3CKCR 寄存器的 CLKEN 位置 1，使能 Timer3 计数时钟。

4、Timer3 模式配置

- T3CMP0、T3CMP1 寄存器组的模式选择：

  将 T3CR 寄存器的 CMP0M、CMP1M 位都置 1，配置 T3CMP0、T3CMP1 寄存器组都作为捕获寄存器。

- T3CMP0 寄存器组捕获时刻选择：

  将 T3CR 寄存器的 CMP0CAP 位清 0，选择 T3CMP0 寄存器组捕获时刻为在输入引脚 1 有效沿进行捕获。

5、清 Timer3 中断标志位

使能 Timer3 中断前，先将 T3SR 寄存器的 INT0SR、INT1SR 位清 0，清 Timer3 中断标志位。

6、中断控制

Timer3 有两个中断，Timer3 中断 0、Timer3 中断 1：

1)、Timer3 中断 1 只有在单脉冲输出、外部电平宽度测量模式下才可以产生；

2)、Timer3 中断 0 在所有模式下都可以产生。具体请看 “12.3、Timer3 中断”。

T3CR 寄存器的 INT0EN、INT1EN 位是 Timer3 中断使能位，T3SR 寄存器的 INT0SR、INT1SR 位是 Timer3 中断标志位。

7、使能外设模块中断

使能寄存器 IRQ0ER 的 bit1 位，IRQ0ER_TCM3INT0 = 0x02，使能 Timer3 中断 0；使能寄存器 IRQ1ER 的 bit1 位，IRQ1ER_TCM3INT1 = 0x02，使能 Timer3 中断 1。

8、使能内核中断

1)、EX0 = 1,EX1 = 1; //使能内核中断 0、1

2)、配置内核中断 0、1 中断优先级 IP1_bit[1:0]、IP1_bit[3:2]（0：最低优先级；1：最高优先级）

3)、使能总内核中断 EA=1。

9、工作模式选择

1)、置 T3CR_MODE[2:0]=010,即可使能 Timer3 开始工作(注意:在初始化定时器或 Timer3 关闭情况下,给 T3CMP0L、T3CMP0H、T3CMP1L、T3CMP1H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T3CR_MODE[2:0]=010 使能 T3 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟)。

2)、配置双边沿为有效沿，如果使能了 Timer3 中断 0、中断 1 中断，在 Timer3 输入 0 引脚外部信号的上升沿、下降沿产生 Timer3 中断 1，Timer3 输入 1 引脚外部信号的上升沿、下降沿产生 Timer3 中断 0，连续 3 次中断就可以得出一个低电平宽度时间，一个高电平宽度时间。(具体哪个是高电平时间或低电平时间必须从实际情况分析)

Timer3 使用两组捕获寄存器测量两个输入电平的宽度工作时序图

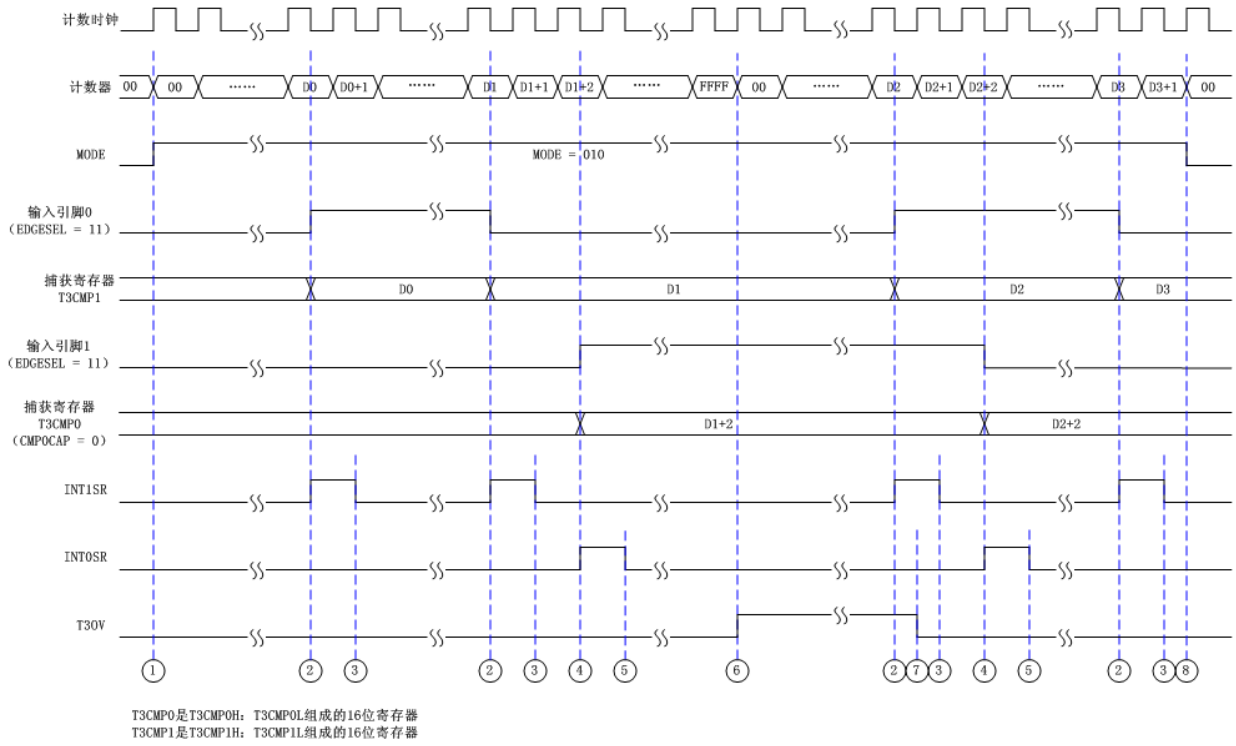


*图 12-6、Timer3 两组捕获寄存器测量两个输入电平宽度*工作时序图

波形说明：

① 设置计数器模式 MODE[2:0]位 = 010 后，计数器开始计数；

② 检测到输入引脚 0 的有效沿(这里必须设置有效沿为双边沿)，，将计数器的值锁存到捕获寄存器 T3CMP1H：T3CMP1L 中，同时产生 Timer3 中断 1(即 T3SR 寄存器的 INT1SR 位置 1)；

③ 软件处理并清除中断 INT1SR；

④ 检测到输入引脚 1 的有效沿(这里必须设置有效沿为双边沿)，将计数器的值锁存到捕获寄存器 T3CMP0H:T3CMP0L 中，同时产生 Timer3 中断 0(即 T3SR 寄存器的 INT0SR 位置 1)；

⑤ 软件处理并清除中断 INT0SR；

⑥ 计数器溢出；

⑦ 软件处理后清除计数器溢出标志位；

⑧ 设置计数器模式 MODE[2:0]位 = 000 后，计数器清零并停止计数。

注：

1、输入引脚的有效沿必须设置为双边沿，这样检测到上升沿、下降沿都会产生中断并锁存计数器的值。

2、若输入引脚是周期稳定的信号，可以根据相邻 3 次中断所捕获到的数据，得到输入引脚的信号的周期/频率。

3、理论上是无法区分出得到的是高电平宽度还是低电平宽度，必须根据实际上输入信号的特征来区分。

使用两组捕获寄存器测量一个输入电平的宽度

此模式下，使用的是 T3CMP0H：T3CMP0L、T3CMP1H：T3CMP1L 这两组寄存器作为捕获寄存器，测量定时器 3 输入 0 的电平宽度。配置 Timer3 为使用两组捕获寄存器测量一个输入电平的宽度的操作步骤如下：

1、端口配置

通过 PxyIOCFG 寄存器将 I/O 配置成定时器 3 输出/输入 0 功能，并且配置 PnOE 寄存器，将定时器 3 输出或输入 0 引脚方向置为输入。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)

2、Timer 模块时钟使能

将 MDLCKCR 寄存器的 TIMEREN 位置 1，使能 Timer 模块时钟。

3、关闭 Timer3

当 Timer3 在停止工作状态下才能完成 Timer3 的以下配置，T3CR_MODE = 0 即关闭 Timer3。

4、清中断标志：T3SR = 0x00 ；

3、Timer3 计数时钟参数配置

- 计数时钟选择：

  1)、T3CKCR 寄存器的 CLKSEL 位可以选择高速时钟（系统副时钟 SCLK1）或者低速时钟（SCLK2 或内部 800K）作为 Timer3 的时钟。

  2)、选择低速时钟时，还必须配置寄存器 MDLCKCR 的 bit7 位 MDLCKCR_TLCKSEL 来选择是内部低速 800K 或外部 M/K 级晶振作为 Timer3 的时钟。

- 计数时钟分频配置：

  T3CKCR 寄存器的 CLKDIV[2:0]位可以对选择的时钟源进行分频。此时不能配置为输入引脚的有效沿作为计数时钟选项。

- 输入引脚的有效沿设置：

  T3CKCR 寄存器的 EDGESEL[1:0]位可以选择输入引脚的有效沿，只能选择输入引脚的有效沿为上升沿或下降沿，不能配置为其他选项。

- 计数时钟使能：

  将 T3CKCR 寄存器的 CLKEN 位置 1，使能 Timer3 计数时钟。

4、Timer3 模式配置

- T3CMP0、T3CMP1 寄存器组的模式选择：

  将 T3CR 寄存器的 CMP0M、CMP1M 位都置 1，配置 T3CMP0、T3CMP1 寄存器组都作为捕获寄存器。

- T3CMP0 寄存器组捕获时刻选择：

  将 T3CR 寄存器的 CMP0CAP 位置 1，选择 T3CMP0 寄存器组捕获时刻为在输入引脚 0 有效沿的反沿进行捕获。

5、清 Timer3 中断标志位

使能 Timer3 中断前，先将 T3SR 寄存器的 INT0SR、INT1SR 位清 0，清 Timer3 中断标志位。

6、中断控制

Timer3 有两个中断，Timer3 中断 0、Timer3 中断 1：

1)、Timer3 中断 1 只有在单脉冲输出、外部电平宽度测量模式下才可以产生；

2)、Timer3 中断 0 在所有模式下都可以产生。具体请看 “12.3、Timer3 中断”。

T3CR 寄存器的 INT0EN、INT1EN 位是 Timer3 中断使能位，T3SR 寄存器的 INT0SR、INT1SR 位是 Timer3 中断标志位。

7、使能外设模块中断

使能寄存器 IRQ0ER 的 bit1 位，IRQ0ER_TCM3INT0 = 0x02，使能 Timer3 中断 0；使能寄存器 IRQ1ER 的 bit1 位，IRQ1ER_TCM3INT1 = 0x02，使能 Timer3 中断 1。

8、使能内核中断

1)、EX0 = 1,EX1 = 1；//使能内核中断 0、1

2)、配置内核中断 0、1 中断优先级 IP1_bit[1:0]、IP1_bit[3:2]（0：最低优先级；1：最高优先级）

3)、使能总内核中断 EA=1。

9、工作模式选择

1)、置 T3CR_MODE[2:0]=010,即可使能 Timer3 开始工作(注意：在初始化定时器或 Timer3 关闭情况下，给 T3CMP0L、T3CMP0H、T3CMP1L、T3CMP1H 寄存器赋值后必须等待>=3*T 所选时钟源的时间后才能 T3CR_MODE[2:0]=010 使能 T3 工作，T 所选时钟源指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟)。

2)、如果使能了 Timer3 中断 0、中断 1：

- 若配置输入引脚有效沿为上升沿，则在 Timer3 输入 0 引脚外部信号的上升沿，产生 Timer3 中断 1；在 Timer3 输入 0 引脚外部信号的下降沿，产生 Timer3 中断 0；
- 若配置输入引脚有效沿为下降沿，则在 Timer3 输入 0 引脚外部信号的下降沿，产生 Timer3 中断 1；在 Timer3 输入 0 引脚外部信号的上升沿，产生 Timer3 中断 0。
- 在 Timer3 中断 1 记录 T3CMP1H、T3CMP1L 的值，在 Timer3 中断 0 记录 T3CMP0H、T3CMP0L 的值，通过连续 3 次中断就可以得出高低电平的时间。

Timer3 使用两组捕获寄存器测量一个输入电平的宽度工作时序图

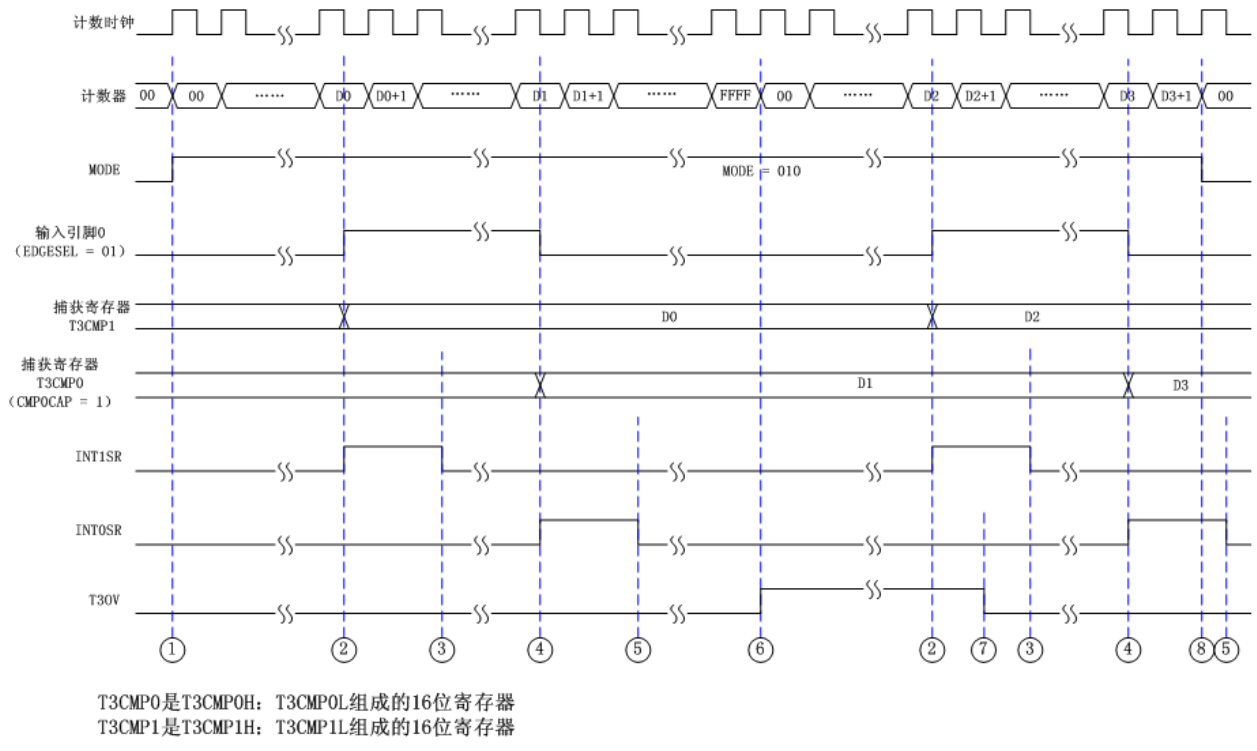


*图 12-7、Timer3 两组捕获寄存器测量两个输入电平宽度*工作时序图

波形说明：

① 设置计数器模式 MODE[2:0]位 = 010 后，计数器开始计数；

② 检测到输入引脚 0 的有效沿，将计数器的值锁存到捕获寄存器 T3CMP1H:T3CMP1L 中，同时产生 Timer3 中断 1(即 T3SR 寄存器的 INT1SR 位置 1)；

③ 软件处理并清除中断 INT1SR，软件应在中断处理程序中将寄存器 T3CMP1H:T3CMP1L 的值保存到变量 A 中；

④ 检测到输入引脚 0 的有效沿的反沿，将计数器的值锁存到捕获寄存器 T3CMP0H:T3CMP0L 中，同时产生 Timer3 中断 0(即 T3SR 寄存器的 INT0SR 位置 1)；

⑤ 软件处理并清除中断 INT0SR，软件应在中断处理程序中将寄存器 T3CMP0H:T3CMP0L 的值保存到变量 B 中；

⑥ 计数器溢出；

⑦ 软件处理后清除计数器溢出标志位；

⑧ 设置计数器模式 MODE[2:0]位 = 000 后，计数器清零并停止计数。

注：

1、输入引脚的有效沿可以设置为上升沿/下降沿，这样有效沿的反沿就是下降沿/上升沿。在有效沿会产生 Timer3 中断 1，并锁存计数器的值到 T3CMP1H:T3CMP1L 寄存器；在反沿产 Timer3 中断 0，并锁存计数器的值到 T3CMP0H:T3CMP0L 寄存器。

2、若输入引脚是周期稳定的信号，可以根据相邻 3 次中断所捕获到的数据，得到输入引脚的信号的周期/频率。

3、本模式可以分辨出得到的是高电平宽度还是低电平宽度。

12.2、Timer3 寄存器的定义

表 12-3、Timer3 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
T3CR	0x88FF	XRAM	Timer3模式控制寄存器	0000 0000
T3CKCR	0x89FF	XRAM	Timer3时钟控制寄存器	0000 0000
T3OCR	0x8AFF	XRAM	Timer3输出控制寄存器	0000 0000
T3CMP0L	0x8BFF	XRAM	Timer3比较寄存器0低8位	0000 0000
T3CMP0H	0x8CFF	XRAM	Timer3比较寄存器0高8位	0000 0000
T3CMP1L	0x8DFF	XRAM	Timer3比较寄存器1低8位	0000 0000
T3CMP1H	0x8EFF	XRAM	Timer3比较寄存器1高8位	0000 0000
T3SR	0x8FFF	XRAM	Timer3状态寄存器	0000 1000
T3CNTL	0xADFF	XRAM	Timer3计数寄存器低8位	0000 0000
T3CNTH	0xAEFF	XRAM	Timer3计数寄存器高8位	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

T3CR：Timer3 模式控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CMP1M	CMP0M	INT1EN	INT0EN	CMP0CAP	MODE2	MODE1	MODE0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 CMP1M：T3CMP1 寄存器组的模式选择位
0 = T3CMP1 寄存器组作为比较寄存器
1 = T3CMP1 寄存器组作为捕获寄存器

bit 6 CMP0M：T3CMP0 寄存器组的模式选择位
0 = T3CMP0 寄存器组作为比较寄存器
1 = T3CMP0 寄存器组作为捕获寄存器

bit 5 INT1EN：Timer3 中断 1 使能位
0 = 禁止 Timer3 中断 1 中断
1 = 使能 Timer3 中断 1 中断

bit 4 INT0EN：Timer3 中断 0 使能位
0 = 禁止 Timer3 中断 0 中断

1 = 使能 Timer3 中断 0 中断

bit 3 CMP0CAP：T3CMP0 寄存器组捕获时刻控制位

0 = T3CMP0 寄存器组在输入引脚 1 有效沿进行捕获

1 = T3CMP0 寄存器组在输入引脚 0 有效沿的反沿进行捕获

bit 2-0 MODE[2:0]：Timer3 模式控制位

00x = Timer3 停止操作，输出无变化，无中断产生

010 = 自由运行模式(计数器溢出清零)

011 = 保留

10x = 由输入引脚的有效沿清除计数器的模式

110 = 由比较匹配清除计数器的模式，当比较匹配时，输出翻转

111 = 由比较匹配清除计数器的模式，无波形输出

T3CKCR：Timer3 时钟控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CLKEN	CLKSEL	EDGESEL1	EDGESEL0	READMODE	CLKDIV2	CLKDIV1	CLKDIV0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 CLKEN：Timer3 计数时钟使能位

0 = 关闭 Timer3 计数时钟

1 = 打开 Timer3 计数时钟

bit 6 CLKSEL：Timer3 计数时钟选择位

0 = 选择高速时钟（系统副时钟 SCLK1）作为 Timer3 计数时钟

1 = 选择低速时钟（系统副时钟 SCLK2、内部低速 800K）作为 Timer3 计数时钟(具体是 800K 还是外部晶振需要配置 MDLCKCR_TLCKSEL 来决定，详情请参考“4.2、MDCLKCR 寄存器”描述)。

bit 5-4 EDGESEL[1:0]：Timer3 输入引脚的有效沿选择位

00 = 禁止设置

01 = 上升沿

10 = 下降沿

11 = 双边沿

bit 3 READMODE：读 timer3 计数器模式选择，如果要读取 T3CNT 的值，则必须根据 Timer3 的时钟配置此位为正确的模式值，否则读到的 T3CNT 值与真实值之间可能存在误差

0 = Timer3 的时钟是 M 级时钟源

1 = Timer3 的时钟是 K 级时钟源（32.768K、100K、800K 或外部 M 级时钟经过 XCLKDIV 分频后的 K 级时钟）

bit 2-0 CLKDIV[2:0]：Timer3 计数时钟分频选择位

000 = 2 分频

001 = 4 分频

010 = 8 分频

011 = 16 分频

1xx = 输入引脚的有效沿作为计数时钟

T3OCR：Timer3 输出控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	SPO	SPOM	OUTINIT1	OUTINIT0	INV1EN	INV0EN
U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

注：R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）

bit 7-6 保留未用

bit 5 SPO：单脉冲输出触发位

写 1 后触发单脉冲输出，自动清零，该位读一直为 0。

bit 4 SPOM：单脉冲输出模式使能位

0 = 连续脉冲输出模式

1 = 单脉冲输出模式

bit 3-2 OUTINIT[1:0]：Timer3 输出电平初始状态(方波输出、PWM 输出、单脉冲输出才有效)

00 = 保持原状态

01 = 初始输出低电平

10 = 初始输出高电平

11 = 保持原状态

bit 1 INV1EN：Timer3 计数器与 T3CMP1 匹配输出翻转使能位

0 = 当 Timer3 计数器与 T3CMP1 匹配时，输出不翻转

1 = 当 Timer3 计数器与 T3CMP1 匹配时，输出翻转

bit 0 INV0EN：Timer3 计数器与 T3CMP0 匹配输出翻转使能位

0 = 当 Timer3 计数器与 T3CMP0 匹配时，输出不翻转

1 = 当 Timer3 计数器与 T3CMP0 匹配时，输出翻转

T3CMP0L：Timer3 比较寄存器 0 低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T3CMP0L[7:0]：Timer3 比较寄存器 0 低 8 位

Timer3 比较寄存器 0 低 8 位，在 Timer3 工作期间，可以修改此寄存器

T3CMP0H：Timer3 比较寄存器 0 高 8 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T3CMP0H[7:0]：Timer3 比较寄存器 0 高 8 位

Timer3 比较寄存器 0 高 8 位，在 Timer3 工作期间，可以修改此寄存器

注：1、T3CMP0H 和 T3CMP0L 组成 16 位的比较寄存器 0(T3CMP0)；

2、在比较模式下，T3CMP0 存储的是比较值，由软件写入；且在定时器工作期间，可以修改该寄存器的值；

3、在捕获模式下，T3CMP0 存储的是捕获值，由硬件写入；

4、必须先配置 T3CMP0L，再配置 T3CMP0H；

5、在初始化 T3 或定时器关闭情况下,设置完 T3CMP0L、T3CMP0H、T3CMP1L、T3CMP1H 后需要延时等待>=3*$T_{所选时钟源}$的时间才能使能 T3 工作，$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟。

例程 12-1：修改 T3CMP0H、T3CMP0L 寄存器：

```
T3CMP0L = 0x60;                    //修改 T3CMP0L 寄存器
T3CMP0H = 0x02;                    //修改 T3CMP0H 寄存器
```

T3CMP1L ：Timer3 比较寄存器 1 低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T3CMP1L[7:0]：Timer3 比较寄存器 1 低 8 位

Timer3 比较寄存器 1 低 8 位，在 Timer3 工作期间，可以修改此寄存器。

T3CMP1H：Timer3 比较寄存器 1 高 8 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T3CMP1H[7:0]：Timer3 比较寄存器 1 高 8 位

Timer3 比较寄存器 1 高 8 位，在 Timer3 工作期间，可以修改此寄存器。

T3CMP0H、T3CMP0L 决定输出 PWM 的周期，T3CMP1H、T3CMP1L 决定输出 PWM 的占空比，在 Timer3 工作期间，可以修改 T3CMP1H、T3CMP1L 寄存器改变输出 PWM 的占空比。

注：1、T3CMP1H 和 T3CMP1L 组成 16 位的比较寄存器 1(T3CMP1)；

2、在比较模式下，T3CMP1 存储的是比较值，由软件写入；

3、在捕获模式下，T3CMP1 存储的是捕获值，由硬件写入；

4、软件配置时，必须先配置 T3CMP1L，再配置 T3CMP1H；

5、在初始化 T3 或定时器关闭情况下，设置完 T3CMP0L、T3CMP0H、T3CMP1L、T3CMP1H 后需要延时等待>=3*$T_{所选时钟源}$的时间才能使能 T3 工作，$T_{所选时钟源}$指"4.1、系统总时钟框图"中到达 T3CKCR_CLKSEL 的时钟。

T3SR：Timer3 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	T3OV	INT1SR	INT0SR
U-x	U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-3 保留未用

bit 2 T3OV：Timer3 计数器溢出状态位

0 = Timer3 计数器没有溢出过

1 = Timer3 计数器已经溢出，需软件写 0 清 0

bit 1 INT1SR：Timer3 中断 1 标志位

0 = 没有产生 Timer3 中断 1 条件

1 = 产生了 Timer3 中断 1 条件，需软件写 0 清 0

bit 0 INT0SR：Timer3 中断 0 标志位

0 = 没有产生 Timer3 中断 0 条件

1 = 产生了 Timer3 中断 0 条件，需软件写 0 清 0

T3CNTL：Timer3 计数寄存器低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
注：R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T3CNTL[7:0]: Timer3 当前计数值低 8 位
与 T3CNTH 组成 16 位计数器

T3CNTH：Timer3 计数寄存器高 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 T3CNTH[7:0]: Timer3 当前计数值高 8 位，与 T3CNTL 组成 16 位计数器
读寄存器 T3CNT 可以获得 Timer3 的当前计数值，读 T3CNT 前需要配置正确的 T3CR 的 READMODE 值，如下例程。
（重启定时器可以清零计数值重新开始计时：T3CR_MODE = 0x00 ; T3CR_MODE = 0xXX ;）

例程 12-2：Timer3 的时钟是 M 级时钟时，读 T3CNT 计数值

```
//初始化 timer3 配置
T3CR_MODE = 0x00;
T3CKCR_READMODE = 0x00;//M 级时钟
T3CR_MODE = 0xXX;//配置 timer3 的工作模式，使能 Timer3
……
tempL= T3CNTL;//读 Timer3 计数值低 8 位
tempH= T3CNTH;//读 Timer3 计数值高 8 位
```

例程 12-3：Timer3 的时钟是 K 级时钟时，读 T3CNT 计数值

```
//初始化 timer3 配置
T3CR_MODE = 0x00;
T3CKCR_READMODE =  0x08; //K 级时钟
T3CR_MODE = 0xXX;//配置 timer3 的工作模式，使能 Timer3
……
temp L= T3CNTL;//读 Timer3 计数值低 8 位
temp H= T3CNTH;//读 Timer3 计数值高 8 位
```

12.3、Timer3 中断

Timer3 有两个中断，Timer3 中断 0 和 Timer3 中断 1，产生 Timer3 中断有以下几种种方式：

- 间隔定时/方波输出/PWM 输出/外部事件计数模式下，当 Timer3 计数器值等于(T3CMP0H：T3CMP0L－1)时，产生 Timer3 中断 0；
- 单脉冲输出模式下，当 Timer3 计数器值等于(T3CMP0H：T3CMP0L－1)时，产生 Timer3 中断 0，当 Timer3 计数器值等于(T3CMP1H：T3CMP1L － 1)时，产生 Timer3 中断 1。
- 外部电平宽度测量模式下，产生 Timer3 中断 0、Timer3 中断 1。具体请查看相应模式的介绍。

只要有 Timer3 中断条件产生，不论是否使能 Timer3 中断，T3SR 寄存器的 INT0SR、INT1SR 中断标志位都会置 1，INT0SR、INT1SR 位必须在软件中清零。Timer3 的中断使能位是 T3CR 寄存器的 INT0EN、INT1EN 位。

注：1、在休眠模式下，Timer3 中断无法唤醒处理器；

2、在空闲模式下，Timer3 中断可以唤醒 处理器。

13、PWM 模块

PWM 模块包括两个独立的模块：

- 16 位 PWM(PWM1)
- 8 位 PWM(PWM2)

PWM 模块的时基 = SYSCLK/(MCLKDIV+1)分频后的时钟 MCLK，时钟详情请参考“4.1、系统总时钟框图”描述。

PWM1 与 PWM2 模块的工作原理完全一样，其中 PWM1 是 16 位，而 PWM2 是 8 位。PWM1 输出的工作原理如下图 13-1 所示。

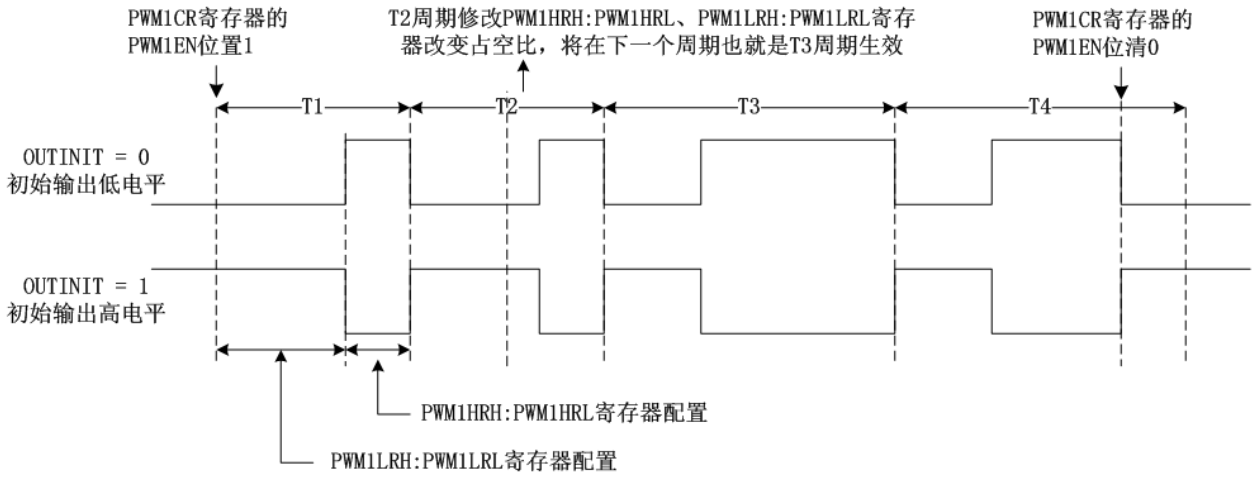


图 13-1：PWM1 输出的工作原理

注：

1、初始电平输出为低电平：

低电平宽度 = (PWM1LRH:PWM1LRL + 1) * $2^{(PWM1CR_CLKDIV + 1)}$ * (T_{SYSCLK}/(CKDIVCR_MCLKDIV[5:0]+1))

高电平宽度 = (PWM1HRH:PWM1HRL + 1) * $2^{(PWM1CR_CLKDIV + 1)}$ * (T_{SYSCLK}/(CKDIVCR_MCLKDIV[5:0]+1))

2、初始电平输出为高电平：

高电平宽度 = (PWM1LRH:PWM1LRL + 1) * $2^{(PWM1CR_CLKDIV + 1)}$ * (T_{SYSCLK}/(CKDIVCR_MCLKDIV[5:0]+1))

低电平宽度 = (PWM1HRH:PWM1HRL + 1) * $2^{(PWM1CR_CLKDIV + 1)}$ * (T_{SYSCLK}/(CKDIVCR_MCLKDIV[5:0]+1))

13.1、PWM 的操作步骤

PWM1、PWM2 的操作步骤完全一样，这里以 PWM1 为例，配置 PWM1 的操作步骤如下：

1、端口配置

通过 PxyIOCFG 寄存器将 I/O 配置成 PWM1 输出功能时，会自动将引脚方向配置为输出，不再需配置对应 PnOE 寄存器。

注：端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1。

2、PWM 模块时钟使能

将 MDLCKCR 寄存器的 PWMEN 位置 1，使能 PWM 模块时钟。

3、PWM 模块参数配置

- 输出初始状态选择：

  配置 PWM1CR 寄存器的 OUTINIT 位，选择输出初始状态。

- 计数时钟分频选择：

配置 PWM1CR 寄存器的 CLKDIV[2:0]位，选择计数时钟分频。

- 调制占空比：

配置 PWM1LRH：PWM1LRL 与 PWM1HRH：PWM1HRL 寄存器，调整 PWM1 输出的占空比。

1)、当初始电平输出为低电平时，PWM1LRH：PWM1LRL 配置的是输出**低电平**的时间，PWM1HRH：PWM1HRL 配置的是输出**高电平**的时间。

2)、当初始电平输出为高电平时，PWM1LRH：PWM1LRL 配置的是输出**高电平**的时间，PWM1HRH：PWM1HRL 配置的是输出**低电平**的时间。

注：

1、配置 PWM1 脉宽调制寄存器的顺序必须如下：先配置 PWM1LRL、再配置 PWM1LRH、再配置 PWM1HRL、最后配置 PWM1HRH。

2、配置 PWM2 脉宽调制寄存器的顺序必须如下：先配置 PWM2LR、再配置 PWM2HR。

4、使能 PWM1：

将 PWM1CR 寄存器的 PWM1EN 位置 1，PWM1 开始工作产生 PWM 信号输出。

13.2、PWM 寄存器的定义

表 13-1、PWM 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
PWM1CR	0x68FF	XRAM	PWM1控制寄存器	0000 0000
PWM1LRL	0x6AFF	XRAM	PWM1脉宽调制寄存器0低8位	0000 0000
PWM1LRH	0x6BFF	XRAM	PWM1脉宽调制寄存器0高8位	0000 0000
PWM1HRL	0x6CFF	XRAM	PWM1脉宽调制寄存器1低8位	0000 0000
PWM1HRH	0x6DFF	XRAM	PWM1脉宽调制寄存器1高8位	0000 0000
PWM2CR	0x69FF	XRAM	PWM2控制寄存器	0000 0000
PWM2LR	0x6EFF	XRAM	PWM2脉宽调制寄存器0	0000 0000
PWM2HR	0x6FFF	XRAM	PWM2脉宽调制寄存器1	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

PWM1CR：PWM1 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	PWM1EN	OUTINIT	-	CLKDIV2	CLKDIV1	CLKDIV0
U-x	U-x	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6　保留未用

bit 5　PWM1EN：PWM1 使能位

0 = 禁止 PWM1

1 = 使能 PWM1，产生 PWM 信号输出

bit 4　OUTINIT：PWM1 输出初始状态

0 = 初始输出低电平

1 = 初始输出高电平

bit 3　保留未用

bit 2-0　CLKDIV[2:0]：PWM1 计数时钟分频选择位

000 = 2 分频

001 = 4 分频

010 = 8 分频

011 = 16 分频

100 = 32 分频

101 = 64 分频

110 = 128 分频

111 = 256 分频

PWM1LRL：PWM1 脉宽调制寄存器 0 低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 PWM1LRL[7:0]：PWM1 脉宽调制寄存器 0 低 8 位，与 PWM1LRH 组成 16 位脉宽调制寄存器 0

PWM1LRH：PWM1 脉宽调制寄存器 0 高 8 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 PWM1LRH[7:0]：PWM1 脉宽调制寄存器 0 高 8 位，与 PWM1LRL 组成 16 位脉宽调制寄存器 0

PWM1HRL：PWM1 脉宽调制寄存器 1 低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 PWM1HRL[7:0]：PWM1 脉宽调制寄存器 1 低 8 位，与 PWM1HRH 组成 16 位脉宽调制寄存器 1

PWM1HRH：PWM1 脉宽调制寄存器 1 高 8 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 PWM1HRH[7:0]：PWM1 脉宽调制寄存器 1 高 8 位，与 PWM1HRL 组成 16 位脉宽调制寄存器 1

注：配置 PWM1 脉宽调制寄存器的顺序必须如下：先配置 PWM1LRL、再配置 PWM1LRH、再配置 PWM1HRL、最后配置 PWM1HRH。

PWM2CR：PWM2 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	PWM2EN	OUTINIT	CLKDIV3	CLKDIV2	CLKDIV1	CLKDIV0
U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 保留未用

bit 5 PWM2EN：PWM2 使能位

0 = 禁止 PWM2

1 = 使能 PWM2，产生 PWM 信号输出

bit 4 OUTINIT：PWM2 输出初始状态

0 = 初始输出低电平

1 = 初始输出高电平

bit 3-0 CLKDIV[3:0]：PWM2 计数时钟分频选择位

0000 = 2 分频

0001 = 4 分频

0010 = 8 分频

0011 = 16 分频

0100 = 32 分频

……

1100 = 8192 分频

1101 = 16384 分频

1110 = 32768 分频

1111 = 65536 分频

PWM2LR：PWM2 脉宽调制寄存器 0

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 PWM2LR[7:0]：PWM2 脉宽调制寄存器 0

PWM2HR：PWM2 脉宽调制寄存器 1

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 PWM2HR[7:0]：PWM2 脉宽调制寄存器 1

注：配置 PWM2 脉宽调制寄存器的顺序必须如下：先配置 PWM2LR、再配置 PWM2HR。

14、实时时钟模块(RTC)

RTC 是实时时钟模块，支持以下功能：

- 时钟功能
- 定时(闹钟)功能(以秒为最小单位)

14.1、RTC 工作原理

RTC 是以秒为最小计时单位，可以实现时、分、秒自动计时与定时闹钟功能：

RTCHDR、RTCMDR、RTCSDR 寄存器组成一组对应 24 小时制时间的时、分、秒寄存器，用于 RTC 的时钟功能。设定好当前时间(配置 RTCHDR、RTCMDR、RTCSDR)，启动 RTC(RTCCR 寄存器的 RTCEN 位置 1)，时钟功能就开始运行，之后软件读 RTCHDR、RTCMDR、RTCSDR 寄存器就得到当前的实时时间(连续两次读到的值一样才认为正确)，同时可产生对应的时、分、秒中断。

RTCAHDR、RTCAMDR、RTCASDR 寄存器组成一组设置闹钟的时、分、秒寄存器，用于 RTC 的定时闹钟功能。设定好闹钟时间(配置 RTCAHDR、RTCAMDR、RTCASDR)，启动 RTC，当 RTCHDR、RTCMDR、RTCSDR 的值与 RTCAHDR、RTCAMDR、RTCASDR 的值匹配时，就会产生闹钟中断。

14.2、RTC 时钟选择

RTC 有两种时钟源可供选择，通过配置 MISCR2 寄存器的 RTCCLKSEL 位实现。

- = 1：外部 32.768K 时钟(必须确保外部接了 32.768K 晶振，或者是外接 M 级晶振经 XCLKDIV 分频得到~32.768K 时钟)
- = 0：内部固定 100K 时钟（由内部 800K 固定分频得到 100K 时钟，分频不可配）

注：RTC 定时闹钟功能可以在休眠模式下、空闲模式下唤醒 MCU 工作

14.3、RTC 的操作步骤

配置 RTC 的操作步骤如下：

1、选择 RTC 工作时钟

RTC 可以选择内部或外部的时钟作为工作时钟，如下（时钟框图请参考“4.1 系统总时钟框图”描述）：

1)、选择内部 100K 为工作时钟，配置 MISCR2_RTCCLKSEL = 0;

2)、选择外部 32.768K 时钟为工作时钟，配置 MISCR2_RTCCLKSEL = 1（必须外接晶振）：

- 外部晶振接的是 32.768K，则配置 XCLKDIVR=0，对外部时钟 1 分频后，作为 RTC 工作时钟；
- 外部晶振接的是 M 级或其它大小晶振，可以配置寄存器 XCLKDIVR，对外部时钟进行（XCLKDIVR+1）分频后，得到~32.768K 时钟作为 RTC 的工作时钟。

2、配置当前 RTC 时间

配置 RTCHDR、RTCMDR、RTCSDR 寄存器可以设置当前时间，必须符合 24 小时制的格式(即 RTCHDR 寄存器的值不能大于 23，RTCMDR、RTCSDR 寄存器得值不能大于 59)，否则会出错。

3、配置定时闹钟时间：

配置 RTCAHDR、RTCAMDR、RTCASDR 寄存器可以设置闹钟时间，必须符合 24 小时制的格式(即 RTCAHDR 寄存器的值不能大于 23，RTCAMDR、RTCASDR 寄存器得值不能大于 59)，否则会出错。

4、清 RTC 中断标志位：

在使能中断前，先写 RTCSR 寄存器 = 0x00，清除所有 RTC 中断标志位。

5、RTC 模块中断控制：

RTCCR 寄存器的 INTSEN、INTMEN、INTHEN、INTDEN、INTCEN 位是 RTC 中断使能位，RTCSR 寄存器的 INTSSR、INTMSR、INTHSR、INTDSR、INTCSR 位是对应的 RTC 中断标志位。

6、外设模块中断控制

使能寄存器 IRQ2ER 的 bit2 位，IRQ2ER_RTCINT = 0x04，即使能 RTC 外设模块的中断。

7、内核中断控制

1)、EX2 = 1；//使能内核中断 2

2)、配置内核中断 2 中断优先级 IP1_BIT[5:4]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

8、RTC 使能：

将 RTCCR 寄存器的 RTCEN 位置 1，RTC 开始工作。

注：RTCSR 寄存器的 SETEN 位默认值为 1，只有当 SETEN = 1 时，才可以配置 RTCHDR、RTCMDR、RTCSDR、RTCAHDR、RTCAMDR、RTCASDR 寄存器中任一的一个，当配置其中一个寄存器时，硬件会先将 SETEN 位清 0，等待内部逻辑同步完成，再恢复为 1。软件在配置其中一个寄存器前，应查询到 SETEN 位等于 1 后，才可以配置 RTCHDR、RTCMDR、RTCSDR、RTCAHDR、RTCAMDR、RTCASD 这几个寄存器，否则可能会出错。

14.4、RTC 寄存器的定义

表 14-1、RTC 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
RTCCR	0xE8FF	XRAM	RTC 控制寄存器	0000 0000
RTCSR	0xE9FF	XRAM	RTC 状态寄存器	1000 0000
RTCHDR	0xEAFF	XRAM	RTC 时钟时寄存器	0000 0000
RTCMDR	0xEBFF	XRAM	RTC 时钟分寄存器	0000 0000
RTCSDR	0xECFF	XRAM	RTC 时钟秒寄存器	0000 0000
RTCAHDR	0xEDFF	XRAM	RTC 闹钟时寄存器	0001 0111
RTCAMDR	0xEEFF	XRAM	RTC 闹钟分寄存器	0011 1011
RTCASDR	0xEFFF	XRAM	RTC 闹钟秒寄存器	0011 1011

以上表中所有寄存器均支持按位操作或直接对字节操作。

RTCCR：RTC 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
RTCEN	-	-	INTCEN	INTDEN	INTHEN	INTMEN	INTSEN
R/W-0	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 RTCEN：RTC 使能位

0 = RTC 停止工作

1 = RTC 开始工作

bit 6-5 保留未用

bit 4 INTCEN：RTC 闹钟中断使能位

0 = 禁止 RTC 闹钟中断

1 = 使能 RTC 闹钟中断

bit 3 INTDEN：RTC 天中断使能位

0 = 禁止 RTC 时钟跳到新的一天中断

1 = 使能 RTC 时钟跳到新的一天中断

bit 2 INTHEN：RTC 时中断使能位

0 = 禁止 RTC 时钟跳到新的小时中断

1 = 使能 RTC 时钟跳到新的小时中断

bit 1 INTMEN：RTC 分中断使能位

0 = 禁止 RTC 时钟跳到新的分钟中断

1 = 使能 RTC 时钟跳到新的分钟中断

bit 0 INTSEN：RTC 秒中断使能位

0 = 禁止 RTC 时钟跳到新的秒钟中断

1 = 使能 RTC 时钟跳到新的秒钟中断

RTCSR：RTC 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SETEN	-	-	INTCSR	INTDSR	INTHSR	INTMSR	INTSSR
R/W-1	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 SETEN：RTC 时钟、闹钟寄存器设置状态位

0 = 不允许配置时钟、闹钟寄存器

1 = 允许配置时钟、闹钟寄存器

bit 6-5 保留未用

bit 4 INTCSR：RTC 闹钟中断标志位

0 = 产生了 RTC 闹钟中断条件

1 = 产生了 RTC 闹钟中断条件，需软件写 0 清 0

bit 3 INTDSR：RTC 天中断标志位

0 = RTC 时钟没有跳到新的一天

1 = RTC 时钟跳到了新的一天，需软件写 0 清 0

bit 2 INTHSR：RTC 时中断标志位

0 = RTC 时钟没有跳到新的一小时

1 = RTC 时钟跳到了新的一小时，需软件写 0 清 0

bit 1 INTMSR：RTC 分中断标志位

0 = RTC 时钟没有跳到新的一分钟

1 = RTC 时钟跳到了新的一分钟，需软件写 0 清 0

bit 0 INTSSR：RTC 秒中断标志位

0 = RTC 时钟没有跳到新的一秒

1 = RTC 时钟跳到了新的一秒，需软件写 0 清 0

注：RTCSR 寄存器的 SETEN 位默认值为 1，只有当 SETEN = 1 时，才可以配置 RTCHDR、RTCMDR、RTCSDR、RTCAHDR、RTCAMDR 、RTCASDR 寄存器中的一个，当配置其中一个寄存器时，硬件会先将 SETEN 位清 0，等待内部逻辑同步完成，再恢复为 1。软件在配置其中某个寄存器前，应查询到 SETEN 位等于 1 后，才可以配置 RTCHDR、RTCMDR、RTCSDR、RTCAHDR、RTCAMDR、RTCASD 这几个寄存器，否则可能会出错。

RTCHDR：RTC 时钟时寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-					
U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注: R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-5 保留未用

bit 4-0 RTCHDR<4:0>：RTC 时钟时寄存器

RTC 小时设置寄存器，不管 RTC 是否已启动，都可以配置此寄存器修改当前时间。

RTCMDR：RTC 时钟分寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-						
U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 保留未用

bit 5-0 RTCMDR[5:0]：RTC 时钟分寄存器

RTC 分钟设置寄存器，不管 RTC 是否已启动，都可以配置此寄存器修改当前时间。

RTCSDR：RTC 时钟秒寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-						
U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 保留未用

bit 5-0 RTCSDR[5:0]：RTC 时钟秒寄存器

RTC 秒钟设置寄存器，不管 RTC 是否已启动，都可以配置此寄存器修改当前时间。

RTCAHDR：RTC 闹钟时寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-					
U-x	U-x	U-x	R/W-1	R/W-0	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-5 保留未用

bit 4-0 RTCAHDR[4:0]：RTC 闹钟时寄存器

RTC 闹钟小时设置寄存器，不管 RTC 是否已启动，都可以配置此寄存器修改闹钟时间。

RTCAMDR：RTC 闹钟分寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-						
U-x	U-x	R/W-1	R/W-1	R/W-1	R/W-0	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 保留未用

bit 5-0 RTCAMDR[5:0]：RTC 闹钟分寄存器

RTC 闹钟分钟设置寄存器，不管 RTC 是否已启动，都可以配置此寄存器修改闹钟时间。

RTCASDR：RTC 闹钟秒寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-						
U-x	U-x	R/W-1	R/W-1	R/W-1	R/W-0	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6　　保留未用

bit 5-0　　RTCASDR[5:0]：RTC 闹钟秒寄存器

RTC 闹钟秒钟设置寄存器，不管 RTC 是否已启动，都可以配置此寄存器修改闹钟时间。

14.5、RTC 中断

RTC 有 5 个中断，分别是：

- RTC 秒中断(RTCSDR 每增加 1 时)
- RTC 分中断(时间从 xx：xx：59 跳到 xx：xx：00 时)
- RTC 时中断(时间从 xx：59：59 跳到 xx：00：00 时)
- RTC 天中断(时间从 23：59：59 跳到 00：00：00 时)
- RTC 定时闹钟中断(时钟寄存器与闹钟寄存器匹配时)

只要有 RTC 中断条件产生，不论是否使能 RTC 中断，RTCSR 寄存器的 INTSSR、INTMSR、INTHSR、INTDSR、INTCSR 中断标志位都会置 1，INTSSR、INTMSR、INTHSR、INTDSR、INTCSR 位必须在软件中清零。RTC 的中断使能位是 RTCCR 寄存器的 INTSEN、INTMEN、INTHEN、INTDEN、INTCEN 位。

注：

1、在休眠模式下，RTC 闹钟中断可以唤醒处理器。

2、在空闲模式下，RTC 中断可以唤醒处理器。

15、ADC 模块

CBM73xxF 系列 MCU 内置一个 10 位精度的 ADC 硬件模块，支持外部输入 8 路 ADC 采样通道。ADC 硬件模块采用逐次逼近法得到转换结果为 10 位二进制数，MCU 根据用户所配置的工作模式对转换结果作对应处理，并将最终处理后得到的值保存在 ADC 结果寄存器(ADCDRL 和 ADCDRH)中。

ADC 特性：

- 10 位数据精度
- ADC 最大采样时钟为 1.4MHZ
- 转换完成可产生中断
- 以 MCU 电压 VDD 为转换参考电压
- 低的采样温漂误差（-40℃～85℃范围内误差 <= ±1%）
- 可选的工作模式：常规模式或硬件模式（硬件模式以内部 vref1v 电压为基准，1v 误差 <= ±10mv）
- 硬件滤波功能：用户可设置滤波级数，硬件滤波处理无需用户软件参与,可节省软件处理时间

ADC 转换工作框图如下 15-1：

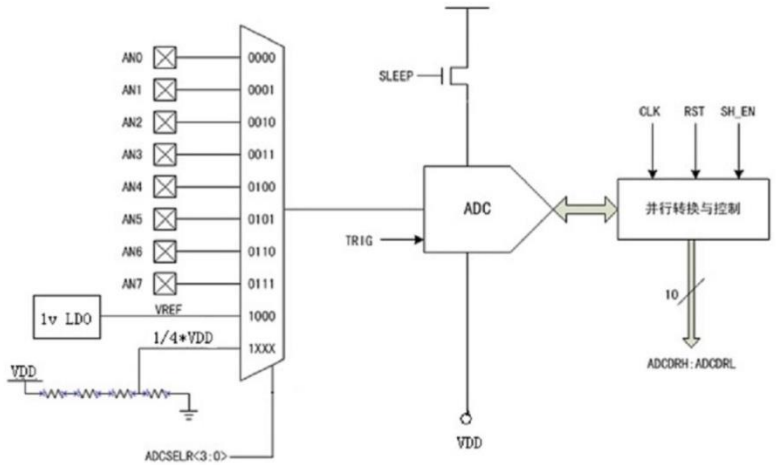


图 15-1、ADC 工作示意图

15.1、ADC 工作原理

温漂校准

当环境温度变化较大时，ADC 会存在一定的温漂，使 ADC 转换结果有误差，因此对 ADC 模块进行温漂校准是非常有必要的；建议定时地去对 ADC 进行温漂校准（根据温度变化快慢，最好温差变化<+-10℃时进行温漂校准），这样可以确保 ADC 的转换误差精度正常而不受温度影响。

启动转换

启动 ADC 转换前，必须先配置正确的工作模式(常规模式或硬件模式），然后再将 ADCCR 寄存器的 TRIG 位置 1，开始模数转换。

注：不能在配置 ADCCR 寄存器其它位的同时将 ADCCR_TRIG 位置 1 启动 ADC 转换。

完成转换

启动 ADC 转换后，当转换完成时，ADC 模块将：

1)、转换得到的结果保存到 ADCDRH：ADCDRL 寄存器中；

2)、ADCSR 寄存器的 SAMPEND 位置 1；

3)、产生中断（如果使能了转换完成中断）。

15.2、ADC 的操作步骤

配置 ADC 的操作步骤如下：

1、端口配置

P06IOCFG = 0x09;//将 P06 口配置为 ADC 通道

P00E_6 = 0x40;//将 P06 寄存器配置为输入方向

P0PU_6 = 0x00;//关闭 P06 口上拉电阻

注：1)、配置 IO 端口寄存器前，应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1（MDLCKCR_IOCEN = 0x08）；

2)、CBM73xxF 系列 MCU 只有 P06、P07、P10、P11、P12、P13、P14、P15 这 8 个 IO 可配置为 ADC 通道(对应 AN0 - AN7)。

2、ADC 模块参数配置

- 使能 ADC 模块硬件电路

  ADCCR_SLEEP = 0x00;//0x00：使能 ADC 电路； 0x40：关闭 ADC 电路

- ADC 工作模式选择：

  配置 ADCCR 寄存器的 MODE 位来选择 ADC 的工作模式：常规模式或硬件模式。

  1)、常规模式

  ADCCR_MODE = 0x00;//配置 MODE 位选择为常规模式

  LDOCR1 |= 0x40;/*在常规模式下，当采样通道 ADCCR_CHLNUM=8（VREF）时必须打开 LDOCR1.bit6 置 1；

  当 ADCCR_CHLNUM=0~7 或 9（AIN0~AIN7 或内部 1/4*VDD）时，LDOCR1.bit6 可置 0 或 1*/

  2)、硬件模式

  ADCCR_MODE = 0x40;//配置 MODE 位选择硬件模式；

  LDOCR1 |= 0X40;// 硬件模式时必须打开 LDOCR1.bit6 置 1

- 采样时钟选择：

  ADC 的时钟源是系统时钟源 SYSCLK，参考“4.1、系统总时钟框图”；通过 ADCDIVCR 寄存器可以进行分频，完成一位转换的时间定义为 TAD=采样时钟周期。一个完整的 10 位转化需要 13 个 TAD 周期，如图 15-2 所示。

- 采样保持时间配置：

  ADCCR 寄存器的 SAMPCNT[1:0]位控制采集时间。是 ADC 模块采集外部输入电压信号的时间，输入电压信号采集完成后，才会启动转换。如图 15-2 所示。(SAMPCNT 值必须大于等于 2)

- ADC 采样通道选择及硬件滤波级数的配置：

  采样通道控制位 ADCCR_CHLSEL：=0x00 表示 ADCSELR_CHLNUM 可设置为 0~8(AN0~AN7、内部 VREF)通道其中之一；

  =0x04:表示 ADCSELR_CHLNUM 只能设置为 9（即内部 1/4*VDD 通道）；

  设置采样通道 ADCSELR_CHLNUM：= 0x00~0x09（AN0~AN7、内部 VREF 1v、内部 1/4*VDD）；

  设置硬件滤波级数 ADCSELR_FLITNUM = n: 0<=n<=6，详情见下文寄存器 ADCSELR。

注：当采样通道 ADCSELR_CHLNUM=(AN0~AN7、VREF)之一时，采样通道控制位 ADCCR_CHLSEL 必须为 0；当采样通道 ADCSELR_CHLNUM=内部 1/4*VDD 时，采样通道控制位 ADCCR_CHLSEL 位必须置 1。

4、清 ADC 中断标志位

使能 ADC 中断前，先将 ADCSR 寄存器的 SAMPEND 位清 0，清除 ADC 采样完成标志位。

5、中断控制

ADC 模块允许在完成模数转换后产生一个中断，ADC 中断使能位是 ADCCR 寄存器的 INTEN 位；同时还必须配置相关内核中断控制。

ADCCR_INTEN = 1<<3;//使能 ADC 中断

IRQ4ER_ADCINT = 0x04;//使能 ADC 外设模块中断

EX4 = 1;//使能内核中断 4

IP1_bit[7:6] = 0;//设置内核中断 4 为最低优先级中断

EA = 1;//使能内核总中断

6、温漂校准

温漂校准分粗调和微调两部分，具体操作请参考“LDOCR0 & LDOCR1 寄存器”描述。

7、启动转换

将 ADCCR 寄存器的 TRIG 位置 1 启动转换，注意必需先配置 ADCCR 寄存器的其他位，再将 ADCCR 寄存器的 TRIG 位置 1 启动转换，两者不能同时进行。选择好转换通道后最好延时>5us 再启动转换（即 TRIG=1）

8、等待 ADC 转换结束

等待 ADC 转换结束有两种方法：

1)、查询 ADCSR 寄存器的 SAMPEND 位，等于 1 即转换结束

2)、等待 ADC 中断(使能中断)

9、读取 ADC 转换结果

转换结束，读取 ADCDRH:ADCDRL 寄存器得到 ADC 转换结果(即采样输出值)。根据工作模式、硬件滤波级数的不同配置，采样输出值的表示含义可分为以下情况：

1>、常规模式下，ADCDRH：ADCDRL 寄存器的结果表示的是采样电压转换量化值：

a、当硬件滤波级数 n=0 时，采样输出值为 10 位有效数据，则 **AD 输入电压 Vin=VDD*采样输出值/1024 v。**

b、当硬件滤波级数 1<=n<=6 时，采样输出值为 16 位有效数据，其中高 10 位 bit[15:6]为转换量化值的有效值，低 6 位 bit[5:0]无效，则 **AD 输入电压 Vin=VDD*(采样输出值>>6)/1024 v。**

2>、硬件模式下，ADCDRH：ADCDRL 寄存器的结果表示的是采样电压绝对值（以内部 Vref_1v 为参考基准转换得出）：

硬件滤波级数设为 0<=n<=6 时，ADCDRH:ADCDRL 寄存器的结果都固定为 13bit，其中高 3 位 bit[12:10]为整数部分，低 10 位 bit[9:0]为小数部分，则 **AD 输入电压 Vin =(采样输出值>>10) + (采样输出值&0x3FF)/1024 v。**

10、清 ADC 中断标志位

清 ADC 中断标志位(ADCSR 寄存器的 SAMPEND 位)，必须清除才能再次进行 ADC 转换。

例 15-1： ADC 转换配置例程

```
P06IOCFG = 0x09;//将 P06 口配置为 ADC 通道
P00E_6 = 0x40;//将 P06 寄存器配置为输入方向
P0PU_6 = 0x00;//关闭 P06 口上拉电阻
ADCCR_SLEEP = 0x00;//使能 ADC 硬件电路
ADCCR_MODE = 0x00;//常规模式
LDOCR1 |= 0x40;//常规模式采样内部 vref_1v 或硬件模式时，LDOCR1 的 bit6 必须要置 1；其它情况可以置 0 或置 1
ADCDIVCR = 0x20;//配置采样时钟
ADCCR_SAMPCNT = 0x02;//采样保持时钟
ADCCR_CHLSEL = 0x00;//0x00:采样通道可选择为 AN0~AN7、VREF 其中之一； 0x04:采样通道只能选择为内部 1/4VDD;
ADCCR_CHLNUM = 0x00;//选择 P06（AN0）通道为 ADC 采样通道
ADCSELR_FLITNUM = 0x00;//0 级滤波  0<=n<=7
ADCSR = 0x00; //清 ADC 中断标志位
i=20;
while(i--);//选择通道后延时>=5us 再启动
ADCCR_TRIG = 0x80;//启动 ADC 转换
while(0x00 == ADCSR_SAMPEND){;} //等待转换完成
ADCSR_SAMPEND = 0x00;//清中断标志位
datl = ADCDRL;//保存采样结果
dath = ADCDRH;
```

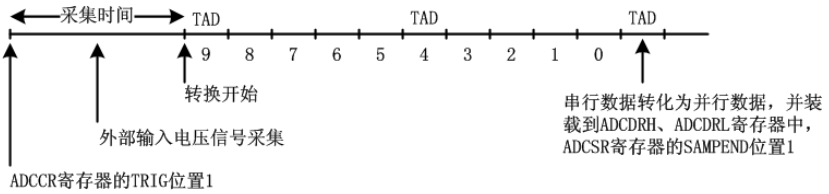


图 15-2、模数转换器 TAD 周期

15.3、ADC 寄存器的定义

表 15-1、ADC 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
ADCDIVCR	0xB8FF	XRAM	ADC 时钟分频寄存器	0001 1111
ADCSELR	0xB9FF	XRAM	ADC 转换通道控制寄存器	0000 0000
ADCCR	0xBBFF	XRAM	ADC 控制寄存器	0000 0010
ADCSR	0xBAFF	XRAM	ADC 状态寄存器	0000 0000
ADCDRL	0xBCFF	XRAM	ADC 数据寄存器低8位	0000 0000
ADCDRH	0xBDFF	XRAM	ADC 数据寄存器高8位	0000 0000
LDOCR0	0x1045	XRAM	LDO 控制寄存器0	000xxxxx
LDOCR1	0x1046	XRAM	LDO 控制寄存器1	010xxxxx

以上表中所有寄存器均支持按位操作或直接对字节操作。且配置寄存器 LDOCR0、LDOCR1 之前必须保证 IOC 模块的时钟打开。

ADCDIVCR:ADC 时钟分频寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 ADCDIVCR[7:0]：ADC 采样时钟分频寄存器，AD 充许的最大采样时钟为 1.4Mhz

ADC 采样时钟= f_{SYSCLK} /(2*(ADCDIVCR+1)) ； 当 f_{SYSCLK}=48M 时,ADCDIVCR 的值必须>=17。

ADCSELR:ADC 转换通道控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	FLITNUM2	FLITNUM1	FLITNUM0	CHLNUM3	CHLNUM2	CHLNUM1	CHLNUM0
U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 保留未用

bit 6-4 FLITNUM[2:0]：硬件滤波级数控制

000 = 1 次采样滤波结果

001 = 2 次采样滤波结果

010 = 4 次采样滤波结果

011 = 8 次采样滤波结果

100 = 16 次采样滤波结果

101 = 32 次采样滤波结果

110 = 64 次采样滤波结果

111 = 保留不可设

bit 3-0 CHLNUM[3:0]：采样通道选择

0000 = AN0

0001 = AN1

0010 = AN2

0011 = AN3

0100 = AN4

0101 = AN5

0110 = AN6

0111 = AN7

1000 = VREF（内部 1v 电压）

1001～1111 = 内部 1/4*VDD （此通道不支持硬件模式）

注：采样通道选择寄存器 ADCSELR_CHLNUM，必须与采样通道控制位寄存器 ADCCR_CHLSEL 的配置相匹配。

ADCCR：ADC 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TRIG	MODE	VREFS	SLEEP	INTEN	CHLSEL	SAMPCNT1	SAMPCNT0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 TRIG：ADC 触发转换控制位

0 = 不触发

1 = 写 1 触发一次 ADC 转换

bit 6 MODE：ADC 工作模式，不同模式输出的结果不同，详情见下文 ADC 寄存器 ADCDRL、ADCDRH 的说明

0 = 常规模式（转换结果为 VDD 的量化值）

1 = 硬件模式（转换结果为绝对电压值，即以内部 1v 电压为基准得到的绝对电压）

bit 5 VREFS：ADC 参考电压选择位（必须选择为 0）

0 = 选择内部固定参考电压

1 = 保留不可设

bit 4 SLEEP：ADC 模块休眠使能位

0 = 使能 ADC 模块硬件电路（要启动 ADC 转换前必须使 ADC 模块硬件电路）

1 = 关闭 ADC 模块硬件电路

bit 3 INTEN：转换完成中断

0 = 不使能转换完成中断

1 = 使能转换完成中断

bit 2 CHLSEL：ADC 模块采样通道控制位

0 = 选择 AN0~AN7 或 VREF 其中之一

1 = 选择 1/4*VDD 采样通道

bit 1-0 SAMPCNT[1:0]：ADC 采样保持时间 = （SAMPCNT[1:0] * 2 + 1）*Tadcclk

确保 SAMPCNT[1:0]的值>=2

注：ADCCR_CHLSEL 要与 ADCCR_CHLNUM 相匹配。

ADCSR:ADC 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	-	SAMPEND
U-x	U-x	U-x	U-x	U-x	U-x	U-x	R-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 保留未用

bit 0 SAMPEND：ADC 转换完成标志位

0 = 未完成

1 = 转换完成，需软件写 0 清 0

ADCDRL:ADC 转换输出数据寄存器低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

ADCDRH:ADC 转换输出数据寄存器高 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

根据工作模式（常规模式/硬件模式）、硬件滤波级数 ADCSELR_ FLITNUM[2:0]=n 的不同配置，分为以下情况：

1>、常规模式下，采样输出值 ADCDRH：ADCDRL 表示的是采样电压转换量化值：

a、当硬件滤波级数 n=0 时，采样输出值为 10 位有效数据，则 **AD 输入电压 Vin=VDD*采样输出值/1024 v。**

b、当硬件滤波级数 1<=n<=6 时，采样输出值为 16 位有效数据，其中高 10 位 bit[15:6]为转换量化值的有效值，低 6 位 bit[5:0]无效，则 **AD 输入电压 Vin =VDD*(采样输出值>>6)/1024 v。**

2>、硬件模式下，采样输出值 ADCDRH：ADCDRL 表示的是采样电压绝对值（以内部 Vref_1v 为参考基准转换得出）：

硬件滤波级数设为 0<=n<=6 时，ADCDRH：ADCDRL 寄存器的转换输出结果都固定为 13bit，其中高 3 位 bit[12:10]为整数部分，低 10 位 bit[9:0]为小数部分，则 **AD 输入电压 Vin =(采样输出值>>10) + (采样输出值&0x3FF)/1024 v。**

LDOCR0:LDO 控制寄存器 0

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TRIM0OUT	VDDSEL	ENTRIM0	TRIM0VAL4	TRIM0VAL3	TRIM0VAL2	TRIM0VAL1	TRIM0VAL0
R/W-0	R/W-0	R/W-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 TRIM0OUT：温漂校准**粗调**输出（请见下面粗调步骤描述）

bit 6 VDDSEL：MCU 输入电压 VDD 选择位

0 = 4.7v~5.5v

1 = 3.3v

bit 5 ENTRIM0：温漂校准粗调使能位

0 = 不使能温漂校准粗调

1 =使能温漂校准粗调

bit 4-0 TRIM0VAL[4:0]：温漂校准粗调值范围设置（0~31）

注：配置寄存器 OSC800KTRIMR、PLLTRIMR、XTALCR0、XTALCR1、XTALCR2、LDOCR0、LDOCR1 前必须打开 IOC 模块的时钟。

温漂校准粗调步骤：

1、将 ENTRIM0 位值 1，使能粗调功能；

2、修改粗调值 TRIM0VAL[4:0]（从 00000 到 11111 递增）；

3、延时约 10us，再判断粗调输出的结果 TRIM0OUT 值是否与上一次不同，如果不同则当前的 TRIM0VAL[4:0]即为温漂粗调结果；如果相同则继续第 2 步骤校准。

LDOCR1:LDO 控制寄存器 1

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TRIM1OUT	ADCBUFEN	ENTRIM1	TRIM1VAL4	TRIM1VAL3	TRIM1VAL2	TRIM1VAL1	TRIM1VAL0
R/W-0	R/W-1	R/W-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 TRIM1OUT：温漂校准**微调**输出（请见下面微调步骤描述）

bit 6 ADCBUFEN：adc buf 开关(以下情况时必须置 1 打开 adcbufen：adc 常规模式下采样 vref_1v 通道、adc 工作在硬件模式、使能了 LVD 低电压监测模块)

0 = 关闭 adc buf

1 = 打开 adc buf

bit 5 ENTRIM1：温漂校准微调使能位

0 = 不使能温漂校准微调

1 =使能温漂校准微调

bit 4-0 TRIM1VAL[4:0]：温漂校准微调值范围设置（0~31）

注：配置寄存器 OSC800KTRIMR、PLLTRIMR、XTALCR0、XTALCR1、XTALCR2、LDOCR0、LDOCR1 前必须打开 IOC 模块的时钟。

温漂校准微调步骤（先进行粗调再微调）：

1、将 ENTRIM1 位值 1，使能微调功能；

2、修改微调值 TRIM1VAL[4:0]（从 00000 到 11111 递增）；

3、延时约 10us，再判断微调输出的结果 TRIM1OUT 值是否与上一次不同，如果不同则当前的 TRIM1VAL[4:0]即为温漂微调结果；如果相同则继续第 2 步骤校准。

16、触摸按键功能(sensorADC)

CBM73xxF 系列芯片内建触摸按键功能模块，最大能连接 31 个按键。内部集成了 4 阶可选灵敏度电容，不需要外接电容，可以根据实际电路板材质以及触摸按键介质调节合适的灵敏度。电容值越小，灵敏度越高；电容值越大，灵敏度越低。

下图 16-1 是 sensorADC 的系统框图。

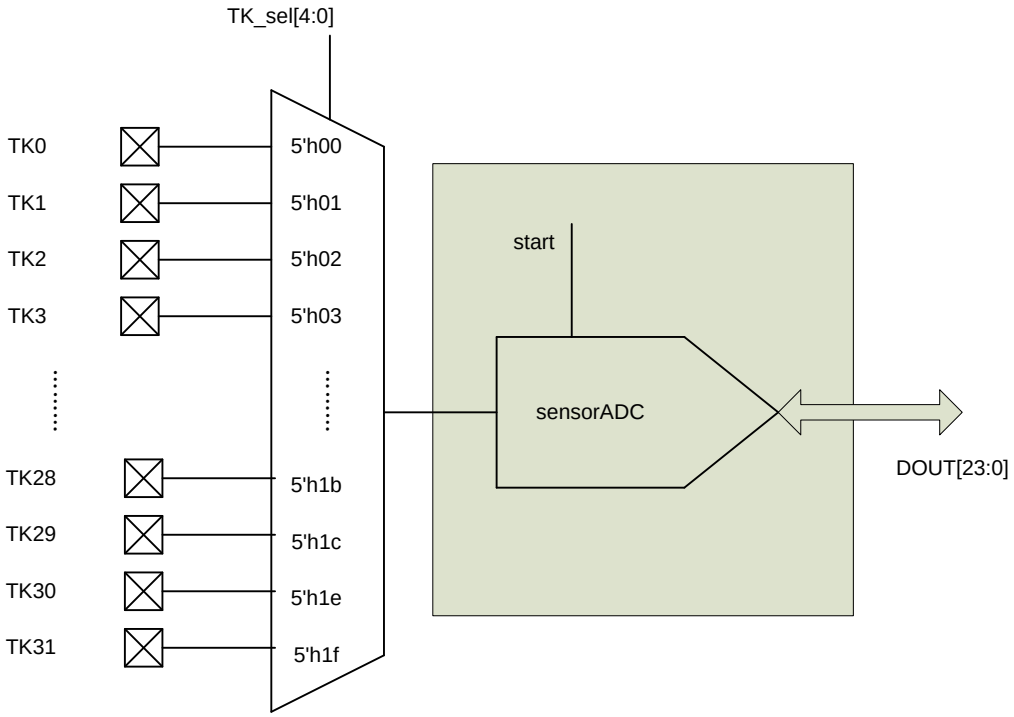


图 16-1：sensorADC 系统框图

sensorADC 工作原理

启动转换

选择软件控制方式，必须先配置好参数，使能 sensorADC 模块，然后将 SCR0 寄存器的 START 位置 1，启动 sensorADC 转换。(START 位为只写位，读为 0)

注：不能在配置 SCR0 寄存器的同时将 START 位置 1 启 动 sensorADC 采样。

完成转换

启动 sensorADC 采样后，当采样完成时，sensorADC 模块将：

1)、SSR 寄存器的 SAMPEND 位置 1

2)、转换得到的结果更新到 SDRM：SDRL 寄存器中

SensorADC 操作步骤

此处介绍正常工作软件模式下配置 sensorADC 的操作步骤，如下：

1、端口配置：

将 sensorADC 输入引脚的 I/O 功能复用配置成 sensorADC 通道(PxyIOCFG)时，会自动将引脚方向配置为输入。不需要配置对应的 PnOE 寄存器，选择 sensorADC 通道引脚方向为输入,关闭上拉电阻功能 。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)

CBM73xxF 系列芯片最多支持 32 路 sensorADC 通道，除了 P0.6、P0.7、P1.0、P1.1、P1.2、P1.3、P1.4、P1.5 这 8

个引脚外，都可以配置成 sensorADC 通道。

2、sensorADC 模块参数配置：

- 配置 sensorADC 控制方式：

  将 SCR0 寄存器的 CTRLMODE<1:0>位配置为 00，选择软件方式控制 sensorADC 的使能及通道切换。

  sensorADC 有三种控制方式：

  1) 软件方式：

  软件方式控制 sensorADC 使能以及通道的切换；

  2) 休眠模式硬件方式：

  在休眠期间，由硬件控制对选定的通道进行反复采样，直到满足唤醒条件将芯片从休眠模式下唤醒。

  3) 正常工作硬件方式：

  应用于触摸库，此处不做介绍。

- 选择硬件滤波器的工作方式：

  将 SCR0 寄存器的 FLTMODE 位清 0，选择仅使用当前 sensorADC 的采样值。

  另一种硬件滤波器的工作方式是与触摸库相关。

- 选择 sensorADC 工作方式：

  SCR1 寄存器的 WORKMODE 位用于配置 sensorADC 工作方式，置 1 为跟随模式，跟随模式可提高 sensorADC 的抗干扰能力。

- 配置灵敏度电容：

  sensorADC 模块内部集成了灵敏度电容，可通过 SCR1 寄存器的 CFSEL<1:0>位选择灵敏度电容的值，有 4 种可供选择：

  00：CI

  01：2/3*CI

  10：1/2*CI

  11：1/3*CI

- 选择工作电流：

  配置 SCR1 寄存器的 CURSEL<3:0>位选择 sensorADC 的电流。

- 配置转化稳定时间：

  启动 sensorADC 转换后，硬件上会先将模拟信号转化为数字波形，这段转化时间可以通过 SCR2 寄存器的 STABLEN<5:0>位进行配置。

  计算方法如下：

  以转化时钟为单位，实际的拍数 = STABLEN<5:0>× 4。

  转化时钟就是 OSC48M 经过 SCR3 寄存器的 TRDIV<3:0> 位分频后的时钟。

- 选择采样精度：

  配置 SCR2 寄存器的 BITLEN<1:0>位选择 sensorADC 的采样精度：

  00 = 10 位（采样时间拍数：31）

  01 = 12 位（采样时间拍数：63）

  10 = 14 位（采样时间拍数：127）

  11 = 16 位（采样时间拍数：256）

  采样精度越高，采样时间也越长。括号内是采样时间的拍数，以 OSC48M 经过 SCR3 寄存器的 TRDIV<3:0> 位分频后时钟的一个 clock 作为一拍。

- 配置当前通道抵消电容：

  由于实际电路板材质以及触摸按键介质存在差异，从而导致外部寄生电容的差异，所以需要根据实际情况配置不同的内部抵消电容。通过 SSCAPH：SSCAPL 寄存器可以配置 9bit 当前通道抵消电容。若采样通道变化，则当

前通道抵消电容也应相应的改变。

- 配置采样时钟：

  sensorADC 的时钟源是 OSC48M，采样时钟是 OSC48M 经过 SCR3 寄存器的 SHDIV<3:0>位分频后的时钟。

- 配置转化时钟：

  sensorADC 的时钟源是 OSC48M，转化时钟是 OSC48M 经过 SCR3 寄存器的 TRDIV<3:0>位分频后的时钟。

- 选择 sensorADC 采样通道：

  由 SSELR 寄存器决定将哪个通道连接到 sensorADC 采样电路。

3、禁止休眠唤醒相关配置

将 SWKCR 寄存器的 CLKSEL 位清 0，选择内部高速 OSC48M 经过 SCR3 寄存器分频后的时钟作为 sensorADC 时钟。

4、使能 sensorADC：

将 SCR1 寄存器的 ANALOGEN 位置 1，使能 sensorADC 模块，使能后 sensorADC 才能正常工作。

5、清软件方式采样完成中断标志位：

使能 sensorADC 中断前，先将 SSR 寄存器清 0，清 sensorADC 相关中断标志位。

6、中断控制：

sensorADC 模块允许在软件方式采样完成后产生一个中断。sensorADC 采样完成中断使能位是 SCR0 寄存器的 SAMPEN 位，sensorADC 中断标志位是 SSR 寄存器的 SAMPEND 位，SAMPEND 位必须软件清 0。

同时还有应用于触摸库的两个中断，中断使能位是 SCR0 寄存器的 FLTEN、CAPEN 位，中断标志位是 SSR 寄存器的 FLTEND、CAPEND 位。这里不做详细介绍，用户使用时直接禁止这两个中断。

SSR 寄存器中 BLEND 位基线更新完成标志位，不产生中断，应用于触摸库。

7、启动转换：

将 SCR0 寄存器的 START 位置 1 启动转换。注意必需先配置 SCR0 寄存器的其他位，再将 SCR0 寄存器的 START 位置 1 启动转换，两者不能同时进行。

8、等待 sensorADC 转换结束：

等待 sensorADC 转换结束有两种方法：

1)、查询 SSR 寄存器的 SAMPEND 位，等于 1 即转换结束

2)、等待 sensorADC 软件方式采样完成中断(使能中断)

9、读取 sensorADC 采样结果：

sensorADC 转换结束后，可以通过读取 SDRH：SDRL 寄存器得到 sensorADC 采样结果。

10、清软件方式采样完成中断标志位：

将 SSR 寄存器的 SAMPEND 位清 0，清 sensorADC 软件方式采样完成中断标志位。以便再次进行 sensorADC 采样。

例 16-1： sensorADC 转换

```
P00IOCFG = 0x08;     //配置 P0.0(TK0)为 sensor 通道
SSR = 0x00;          //清 sensorADC 相关中断标志位
SCR0= 0x00;      //配置软件方式控制、禁止中断
SCR1 = 0xC0;         //使能 sensorADC、配置部分参数
SCR2 = 0x43;         //配置转化稳定时间，选择精度
SCR3 = 0x82;         //配置采样时钟、转化时钟
SSCAPL = 0x00;  //配置抵消电容
SSCAPH = 0x01;
SWKCR = 0x03;        //选择 OSC48M 分频后的时钟
SSELR = 0x00;    //选择 P0.0(TK0)作为 sensorADC 输入
SCR0_START = 0x40;  //启动 ADC 转换
while(0x00 == SSR_SAMPEND){;} //等待转换完成
SSR_SAMPEND = 0x00; //清中断标志位
datl = SDRL;         //保存采样结果
dath = SDRH;
```

sensorADC 寄存器的定义

表 16-1、SensorADC 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
SCR0	90FFH	XRAM	sensorADC 控制寄存器0	0000 0000
SCR1	91FFH	XRAM	sensorADC 控制寄存器1	0011 0100
SCR2	92FFH	XRAM	sensorADC 控制寄存器2	0100 0010
SCR3	93FFH	XRAM	sensorADC 控制寄存器3	1000 0010
SSCAPL	94FFH	XRAM	sensorADC 抵消电容配置寄存器低8位	0000 0000
SSCAPH	95FFH	XRAM	sensorADC 抵消电容配置寄存器高1位	0000 0000
SSR	96FFH	XRAM	sensorADC 中断状态寄存器	0000 0000
SDRL	97FFH	XRAM	sensorADC 结果寄存器低8位	0000 0000
SDRH	98FFH	XRAM	sensorADC 结果寄存器高8位	0000 0000
SSELR	9AFFH	XRAM	sensorADC 通道选择寄存器	0000 0000
SWKCR	9EFFH	XRAM	SensorADC SLEEP 唤醒配置寄存器	0000 0011

以上表中所有寄存器均支持按位操作或直接对字节操作。

SCR0：sensorADC 控制寄存器 0

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	START	CTRLMODE1	CTRLMODE0	FLTMODE	BLEN	FLTEN	SAMPEN
U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7　　保留未用

bit 6　　START：sensorADC 转换启动位

写 1 即启动 sensorADC 转换，完成后自动清 0，应注意必须在 1 次采样完成后才能再次启动。

bit 5-4 CTRLMODE[1:0]：sensorADC 控制方式选择位

00 = 软件方式，软件控制 sensorADC 使能和通道切换。

01 = 硬件方式，用作休眠模式下唤醒。在进入休眠前，软件应选择一个通道用作唤醒通道，在休眠期间，由硬件控制对该通道进行反复采样，直到满足唤醒条件将芯片从休眠模式下唤醒。

1X = 硬件方式，由硬件算法控制 sensorADC 使能、通道切换、抵消电容。bit 3 保留。

bit3 FLTMODE：硬件滤波器工作方式选择位(触摸库相关)

0 = 仅使用当前 sensorADC 的采样值

1 = 使用当前 sensorADC 的采样值和 XRAM 中存放的值

bit2 BLEN：抵消电容自适应完成中断使能位(触摸库相关)

0 = 禁止抵消电容自适应完成中断

1 = 使能抵消电容自适应完成中断

bit1 FLTEN：硬件滤波采样完成中断使能位(触摸库相关)

0 = 禁止硬件滤波采样完成中断

1 = 使能硬件滤波采样完成中断

bit0 SAMPEN：软件方式，采样完成中断使能位

0 = 禁止采样完成中断

1 = 使能采样完成中断

SCR1：sensorADC 控制寄存器 1

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
MODE	ANALOGEN	CFSEL1	CFSEL0	CURSEL3	CURSEL2	CURSEL1	CURSEL0
R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 MODE： sensor adc 模式配置寄存器

0：固定模式

1：跟随模式

bit 6 ANALOGEN：analog ADC 模块启动使能位

0 = 不启动

1 = 启动

bit 5-4 CFSEL<1:0>：内部灵敏度电容选择(电容越小，灵敏度越高，可测试电容范围变小)

00 = CI

01 = 2/3*CI

10 = 1/2*CI

11 = 1/3*CI

bit 3-0 CURSEL<3:0>：sensorADC 电流控制位

0000 = 禁止设置

0001~1111 共 15 档配置

SCR2：sensorADC 控制寄存器 2

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
STABLEN5	STABLEN4	STABLEN3	STABLEN2	STABLEN1	STABLEN0	BITLEN1	BITLEN0

R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-2 STABLEN<5:0>：转化稳定时间选择位

用于设定 ADC 开始转换稳定时间(振荡时钟经 TRDIV 分频后的时钟为一个拍数)

实际的拍数 = STABLEN × 4。

bit 1-0 BITLEN<1:0>：sensorADC 精度选择位(振荡时钟经 TRDIV 分频后的时钟为一个拍数)

00 = 10 位（采样时间拍数：31）

01 = 12 位（采样时间拍数：63）

10 = 14 位（采样时间拍数：127）

11 = 16 位（采样时间拍数：256）

SCR3：sensorADC 控制寄存器 3

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
SHDIV3	SHDIV2	SHDIV1	SHDIV0	TRDIV3	TRDIV2	TRDIV1	TRDIV0
R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-4 SHDIV<3:0>：采样时钟分频选择位

当 SHDIV = 0，ADC 的采样时钟频率 = OSC48M；

当 SHDIV = 1~15，ADC 的采样时钟频率 = OSC48M / (2 × SHDIV)；

bit 3-0 TRDIV<3:0>：转化时钟分频选择位

当 TRDIV = 0，ADC 的转化时钟频率 = OSC48M；

当 TRDIV = 1~15，ADC 的转化时钟频率 = OSC48M / (2 × TRDIV)；

SSCAPL：sensorADC 抵消电容配置寄存器低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 SSCAPL<7:0>：sensorADC 抵消电容配置寄存器低 8 位和 SSCAPH 组成 9bit 的当前通道抵消电容配置寄存器

SSCAPH：sensorADC 抵消电容配置寄存器高 1 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
—	—	—	—	—	—	—	
U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 SSCAPH<7:0>：sensorADC 抵消电容配置寄存器高 1 位和 SSCAPL 组成 9bit 的当前通道抵消电容配置寄存器

SSR：sensorADC 中断状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	CAPEND	BLEND	FLTEND	SAMPEND
U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0

注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）

bit 7-4 保留未用

bit 3 CAPEND：抵消电容自适应完成中断标志位(触摸库相关)

0 = 抵消电容自适应进行中或未启动

1 = 完成抵消电容自适应

bit 2 BLEND：基线更新完成标志位(触摸库相关)

0 = 基线更新进行中或未启动

1 = 完成基线更新

bit 1 FLTEND：硬件方式，采样完成中断标志位

0 = 硬件方式采样进行中或未启动

1 = 完成硬件方式采样

bit 0 SAMPEND：软件方式，采样完成中断标志位

0 = 软件方式采样进行中或未启动

1 = 完成软件方式采样

SDRL: sensorADC 采样结果寄存器低 8 位（bit0~bit7）

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

SDRH: sensorADC 采样结果寄存器高 8 位（bit9~bit15）

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

SDRH: SDRL 组成 16 位 sensor ADC 采样数据结果.

SSELR: SensorADC 通道选择寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
WAIT_CENT2	WAIT_CENT1	WAIT_CENT0	SENSOR_ID4	SENSOR_ID3	SENSOR_ID2	SENSOR_ID1	SENSOR_ID0
R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

在软件模式和 PD 唤醒模式下，用于固件选择某个物理通道进行采样。物理通道与芯片引脚相对应

bit7:5: 启动信号同步时钟个数=WAIT_CENT*8 个 clk

bit4-0: 最大可以配置 32 个按键，按键值依次为 0,1,2…31.

SWKCR: SensorADC SLEEP 唤醒配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	TRDIV_BIT4	PD_TIME1	PD_TIME0	800K_EN	CMPRSET2	CMPRSET1	CMPRSET0
U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 未实现：读为 0

bit 6 转化时钟分频选择位 bit4

bit 5-4 PD 时间

bit 3 CLKSEL：sensorADC 时钟源选择位

0 = 时钟选择内部高速 OSC48M 经过 SCR3 寄存器分频后的时钟

1 = 时钟选择内部低速 OSC80OK 经过 CKDIVCR 寄存器的 SLOWDIV 位分频后，再 4 分频的时钟。

bit 2-0CMPRSET<2:0>：休眠唤醒参数配置位

在休眠模式下，要能被触摸按键唤醒，只有 sensorADC 采样值连续 N 次落入到预先设置的阈值范围内(请参考 SLPWKCR，SLPWKDR 寄存器说明)，若还不足 N 次时，出现一次不在阈值范围内的采样值，则重新计算采样值落入阈值的次数。CMPRSET 位用于设置这里的次数 N。

注：若要使用触摸按键休眠唤醒功能，在进入休眠前，必需将 CLKSEL 位置 1；

17、UART 串口模块

CBM73xxF 系列 MCU 支持两路通用异步收发器模块 UART1 和 UART2；UART1、UART2 模块是异步串行 I/O 通信外设，可与 CRT 终端、PC 机等外设通信设备进行单工、半双工或全双工异步通迅；UART1、UART2 模块在功能与应用上完全一致，只是对应的寄存器有所不同。

UART 模块包含如下功能：

- 全双工异步发送和接收（LSB 低位发送在先）
- 可将字符长度编程为 5/6/7/8 位
- 输入缓冲器溢出错误检测
- 停止位错误检测
- 奇偶校验位错误检测
- 自动波特率检测和校准
- 普通收发模式和 DMA 模式可供选择

以上功能中有两项是 CBM73xxF 系列 MCU 特有的：

1)、自动波特率检测和校准：即对方外设通过先发送 0x11 数据给 CBM73xxF，CBM73xxF 会根据 0x11 自动校准自身的波特率（外设与 CBM73xxF 设定的波特率在一定误差范围内才能有效进行自校准）。

2)、DMA 工作模式：即 UART 模块支持一次性从 XRAM 缓存区的设定地址发送出去 n（n 可编程设置，n<=128）个字节数据，n 字节发送完后可以产生发送完成中断，字节与字节之间的读取与发送无需用户软件参与；同时，还支持一次性从外设接收 n 个字节数据，并将接收到的数据存储到指定 XRAM 缓存区的设定地址中（无需用户软件参与），接收完 n 字节数据后可以产生接收完成中断，用户程序可从 XRAM 区读取接收到的数据。（UART1 与 UART2 均支持 DMA 工作模式）

17.1、UART 的操作步骤

以配置 UART1 的操作步骤为例，如下：

1、端口功能重映射配置

```
P20IOCFG = 0x27;//配置 P20 为 UART1 的输出口
P21IOCFG = 0x33;//配置 P21 为 UART1 的输入口
```

注：端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1。

2、UART1 模块时钟使能

在操作 UART1 模块相关寄存器前，必须首先使能 UART1 模块时钟，即将 MDLCKCR 寄存器的 MDLCKCR_UARTEN 位置 1。

3、UART1 模块参数配置

- 数据格式配置

  USETR 寄存器可以配置通讯数据的格式，包括校验位、数据位、停止位。

- 波特率配置

  波特率有两种配置方式(通过 UCR 寄存器的 BPRREC 位选择)：

  1)、自动波特率校准

  采用自校准方式。对方外设必须先发送 0x11 数据，CBM73xxF MCU 分析对方外设的波特率，在以后的通讯过程中，都采用和对方外设一致的波特率通讯。此种情况对 CBM73xxF MCU 所选时钟源的精准度无特别要求。

  2)、16 位波特率配置寄存器

  波特率依据 16 位波特率配置寄存器 UBPRH：UBPRL 的设置确定（在时钟源准确的情况下，可以采用这种方式）。

  CBM73xxF 选择的系统时钟源用 SYSCLK 表示（见“4.1、系统总时钟框图”详述），那么 UART1 的时钟即为 SYSCLK；UBPRL 为 UBPR 值的低 8 位，UBPRH 为 UBPR 的高 8 位。

  UBPR 寄存器的值与波特率之间的推算公式如下：

  UBPR = CLK_{UART1} / bpr - 1；(CLK_{UART1} 是 uart 的时钟，单位是 Hz，bpr 是波特率大小)

  例如：SYSCLK 选择为内部高速时钟 48M，MCU 设定波特率 bpr 是 115200，则计算 UBPR 如下：

$UBPR = CLK_{UART1} / bpr - 1 = 48 * 1000000 / 115200 - 1 = 416(0x1A0)$

所以 UBPRH = 0x01,UBPRL = 0xA0。

4、中断控制

UART1 中断根据所选工作模式分为普能模式中断、DMA 模式中断：

(1) 普通模式中断控制：

UIER 寄存器的 UIER_TXIE、UIER_RXIE 位分别是 UART1 发送完 1 个 BYTE 数据中断使能位、接收完 1 个 BYTE 数据中断使能位。对应的中断标志位是 USR 寄存器的 USR_TXDONE、USR_RXREADY 位。

不论是否使能发送完成、接收完成中断，当接收到 1 个 BYTE 数据时，USR_RXREADY 都会置 1；当发送完 1 个 BYTE 数时，USR_TXDONE 都会置 1。

USR 寄存器还包括 UART1 的奇偶校验位错误标志位、停止位错误标志位、接收溢出错误标志位，这几种情况不会产生中断，当出现相应错误时标志位就会置 1。

2)、DMA 模式中断控制：

UIER 寄存器的 UIER_TXDONEIE、UIER_RXFULLIE 位分别是 UART1 发送完 n 字节数据长度中断使能位、接收完 n 字节数据长度中断使能位。对应的中断标志位是 USR 寄存器的 USR_DMATXDONE、USR_DMARXFULL 位。

不论是否使能发送完成、接收完成中断，当接收 n 字节数据长度完成时，USR_DMARXFULL 都会置 1；当发送 n 字节数据长度完成时，USR_DMATXDONE 位置 1。

5、外设模块中断控制

使能寄存器 IRQ3ER 的 bit2 位，IRQ3ER_UARTINT = 0x04，即使能 UART1 外设模块的中断。

6、内核中断控制

1)、EX3 = 1；//使能内核中断 3

2)、配置内核中断 3 中断优先级 IP1_bit[5:4]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1。

7、接收、发送使能

当端口、数据格式、波特率都配置好后，可以将 UCR 寄存器的 UCR_RXEN 位置 1 使能数据接收，UCR_TXEN 位置 1 使能数据发送。

注：

1、数据的发送和接收功能是相互独立的，互不受影响；

2、发送、接收的工作模式也是相互独立，互不受影响。

17.2、UART1 普通模式发送数据

1、设置串口波特率，注意，设置波特率时，串口不能处于发送或接收状态，即必须在 UTRC[1:0]=00 时改变波特率；

2、配置 UTRC，设置为发送使能；

3、往 UTXDR 中写入 1byte 数据；

4、查询 USR 寄存器 USR_TXDONE 位，为高则表示发完完 1byte 数据（也可利用发送完成中断发送下一个字节数据）；

5、重复 3,4 步骤，发送多个数据。

17.3、UART1 普通模式接收数据

1、设置串口波特率，注意，设置波特率时，串口不能处于发送或接收状态，即必须在 UCR[1:0]=00 时改变波特率；

2、配置 UTRC，设置为接收使能；

3、中断程序中查询 USR 寄存器 USR_RXREADY 位，为高则表示接收到 1byte 数据；

4、软件读 URXDR 寄存器，读取接收数据；

5、重复 3,4 步骤，接收多个数据。

17.4、UART1 DMA 模式发送数据

1、设置串口波特率，注意，设置波特率时，串口不能处于发送或接收状态，即必须在 UCR[1:0]=00 时改变波特率；
2、配置 UTXDMACR,设置为 DMA 工作模式及设定 XRAM 发送缓存区空间大小；
3、配置 UTXDMAADDRL, UTXDMAADDRH,设置 XRAM 发送缓存区的起始地址；
4、配置 UTRC，设置为发送使能；
5、查询 USR 寄存器 USR_DMATXDONE 位，为高则表示发送完设定数据长度；（也可利用发送完成中断功能）
6、往 XRAM 发送缓存区里写待发送的数据；（直接写入 XRAM 缓存区地址）
7、对 USR 寄存器 USR_DMATXDONE 位写 1 清 0，UART1 将发送数据；
8、重复 5,6,7 步骤，可以发送多帧数据。

17.5、UART1 DMA 模式接收数据

1、设置串口波特率。注意，设置波特率时，串口不能处于发送或接收状态，即必须在 UCR[1:0]=00 时改变波特率；
2、配置 URXDMACR,设置为 DMA 工作模式及设定 XRAM 接收缓存区空间大小；
3、配置 URXDMAADDRL, URXDMAADDRH,设置 XRAM 接收缓存区的起始地址；
4、配置 UTRC，设置为接收使能；
5、中断中查询 USR 寄存器 USR_DMARXFULL 位，为高则表示接收到设定数据长度；
6、软件读取 XRAM 数据；（直接读取 XRAM 缓存区地址）
7、对 USR 寄存器 USR_DMARXFULL 位写 1 清 0，UART1 将重新开始接收数据；
8、重复 5,6,7 步骤，可以接收多帧数据。

17.6、UART1 寄存器的定义

表 17-1、UART1 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
UCR	0xD0FF	XRAM	UART1控制寄存器	0000 0100
USETR	0xD1FF	XRAM	UART1数据格式控制寄存器	0000 1000
UBPRL	0xD2FF	XRAM	UART1波特率配置寄存器低8位	1111 1111
UBPRH	0xD3FF	XRAM	UART1波特率配置寄存器高8位	1111 1111
UIER	0xD4FF	XRAM	UART1中断使能寄存器	0000 0000
USR	0xD5FF	XRAM	UART1状态寄存器	0101 0000
UTXDR	0xD700	XRAM	UART1发送数据寄存器	0000 0000
URXDR	0xD701	XRAM	UART1接收数据寄存器	0000 0000
UTXDMACR	0xD702	XRAM	UART1发送模式选择寄存器	0000 0000
UTXDMAADDRL	0xD703	XRAM	UART1 XRAM 模式发送数据起始地址低位	0000 0000
UTXDMAADDRH	0xD704	XRAM	UART1 XRAM 模式发送数据起始地址高位	0000 0000
URXDMACR	0xD705	XRAM	UART1接收模式选择寄存器	0000 0000
URXDMAADDRL	0xD706	XRAM	UART1 XRAM 模式接收数据起始地址低位	0000 0000
URXDMAADDRH	0xD707	XRAM	UART1 XRAM 模式接收数据起始地址高位	0000 0000

以上表中寄存器地址 0xD700~0xD707 对应的寄存器均不支持按位操作，只能对整个寄存器进行字节的操作；其它寄存器支持按位操作或直接对字节操作。

UCR：UART1 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
–	–	–	–	–	BPRREC	RXEN	TXEN

U-x	U-x	U-x	U-x	U-x	R/W-1	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-3 保留未用

bit 2 BPRREC：UART 波特率自校准使能位

0 = 波特率依据 UBPRH：UBPRL 寄存器的设置

1 = 波特率自校准使能

bit 1 RXEN：UART 接收数据使能位

0 = 禁止接收

1 = 使能接收

bit 0 TXEN：UART 发送数据使能位

0 = 禁止发送

1 = 使能发送

USETR：UART1 数据格式控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	PARSEL1	PARSEL0	STOPBIT	BITNUM3	BITNUM2	BITNUM1	BITNUM0
U-x	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 保留未用

bit 6-5 PARSEL [1:0]：UART 奇偶校验位配置位

00 = 不含奇偶校验位

01 = 奇校验

10 = 偶校验

11 = 含奇偶校验位，但是发送接收都不检测此位

bit 4 STOPBIT：停止位配置位

0 = 停止位 1bit

1 = 停止位 2bit

bit 3-0 BITNUM[3:0]：数据位配置位

1000 = 8 位

0111 = 7 位

0110 = 6 位

0101 = 5 位

其他 = 禁止设置

UIER：UART1 中断使能寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TXIE	RXIE	TXDONEIE	RXFULLIE	-	-	-	-
R/W-0	R/W-0	R/W-0	R/W-0	U-x	U-x	U-x	U-x
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 TXIE：普通模式发送数据完成中断使能位

0 = 禁止发送数据完成中断

1 = 使能发送数据完成中断

bit 6 RXIE：普通模式接收数据完成中断使能位

0 = 禁止接收数据完成中断

1 = 使能接收数据完成中断

bit 5 TXDONEIE：DMA 模式发送数据完成中断使能位

0 = 禁止发送数据完成中断

1 = 使能发送数据完成中断

bit 4 RXFULLIE：DMA 模式接收数据完成中断使能位

0 = 禁止接收数据完成中断

1 = 使能接收数据完成中断

bit 3-0 保留未用

UBPRL：UART1 波特率配置寄存器低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 UBPRL<7:0>：UART1 波特率配置寄存器低 8 位

与 UBPRH 组成 16 位波特率配置寄存器

UBPRH：UART1 波特率配置寄存器高 8 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 UBPRH[7:0]：UART1 波特率配置寄存器高 8 位

与 UBPRL 组成 16 位波特率配置寄存器

注：UART1 时钟频率 UBPR = CLK_{UART1} / bpr - 1；(CLK_{UART1} 是 uart 的时钟，单位是 Hz，bpr 是波特率大小)

例如：系统时钟源 SYSCLK 选择内部高速时钟 48M，MCU 需要设定波特率 bpr 是 115200，则计算 UBPR 如下：

UBPR = CLK_{UART1} / bpr - 1 (CLK_{UART1} 的单位是 Hz) = 1000,000 * 48 / 115200 - 1 = 416(0x1A0)

UTXDR：UART1 发送数据寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 UTXDR[7:0]：UART1 发送数据寄存器

UTXDR 为发送数据寄存器，只有当 USR_TXDONE= 1 时，才能将待发送的数据写入到该寄存器中。否则，写入的数据会丢失，即写 UTXDR 前应查询 USR_TXDONE 的标志。

URXDR：UART 接收数据寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 URXDR[7:0]：UART 接收数据寄存器

URXDR 为接收数据寄存器，只有当 USR_RXREADY= 1 时，才能读取该寄存器，获得接收的数据字节。否则，读出的数据无效。

注：根据当前 UART1 的配置，只有当停止位，奇偶校验位都正确时，才会将接收到的数据存入 URXDR 寄存器中。若配置 UART1 无奇偶校验位，或不关注奇偶校验位是否正确，则判断接收到的数据是否有效时，无须考虑奇偶校验结果。如果 USR_RXREADY= 1 时不及时读取 URXDR，则下一个接收的 Byte 数据将丢失。

USR：UART1 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	DMATXDONE	DMARXFULL	TXDONE	RXREADY	OVER	STOPB	PAR
U-x	R/W-1	R/W-0	R-1	R-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 保留未用

bit 6 DMATXDONE：Xram 缓存内设置的数据长度发送完标志

0 = 还未发送完

1 = 已发送完成，需软件写 1 清 0（USR=0x40）

注：当发送完时，此位为 1，UART1 将不在发送数据，CPU 对此位写 1 清 0，此时 UART 将发送此次数据

bit 5 DMARXFULL：Xram 缓存内设置的数据长度接收满标志

0 = 还未收满

1 = 已收满，需软件写 1 清 0（USR=0x20）

注：当收满时，MCU 将不会再往设定的 XRAM 缓存区保存新数据，后面的数据将丢失，直至 CPU 对此位写 1 清 0 后，UART1 才会重新接收后面的数据；

bit 4 TXDONE：数据发送完成标志位

0 = 还未发送出去，此时不能往 UTXDR 中写入数据

1 = 发送完成，可以往 UTXDR 中写入数据，**写 UTXDR 自动清 0 或软件写 1 清 0（USR=0x10）**

bit 3 RXREADY：接收 BYTE 数据完成标志位

0 = BYTE 数据还未接收完，不能读 URXDR

1 = BYTE 数据已接收完，可以读 URXDR 获取接收数据，**需软件读取 URXDR 才可清 0 或软件写 1 清 0（USR=0x08）**

bit 2 OVER：接收溢出错误标志位

0 = 未发生溢出错误

1 = 已发生溢出错误。溢出错误发送在接收到 1Byte 数据后，软件未及时读 URXDR，当接收到新的数据时，才会发生溢出错误，需软件写 1 清 0（USR=0x04）

bit 1 STOPB：接收停止位位错误标志位

0 = 接收时未发现停止位有错误

1 = 接收时发现停止位有错误，需软件写 1 清 0（USR=0x02）

bit 0 PAR：接收奇偶校验位错误标志位

0 = 接收时未发现奇偶校验位错误。当 PARSEL=00/11时，不检查奇偶校验位，认为奇偶校验位正确。

1 = 接收时发现奇偶校验位错误。若接收奇偶校验位出错，则当前字节丢弃，软件无法获得该字节，需软件写1清0（USR=0x01）。

UTXDMACR：UART1 发送模式配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TXLEN6	TXLEN5	TXLEN4	TXLEN3	TXLEN2	TXLEN1	TXLEN0	TXMODE
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 TXLEN[5:0]：UART1 工作在 DMA 模式下，设定发送 Xram 缓存区空间大小，大小为 TXLEN+1(最大 128byte)

bit 0 TXMODE：发送模式选择

0 = 普通模式。

1 = DMA 模式

UTXDMAADDRL：UART1DAM 工作模式下的发送数据起始地址低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 UTXDMAADDRL： UART1 在 DMA 工作模式下，发送数据 XRAM 缓存区的起始地址低位。

与 UTXDMAADDRH 组合可寻址 Xram 的范围 0x00-0x600（1536Byte）

UTXDMAADDRH：UART1DMA 工作模式下的发送数据起始地址高位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 UTXDMAADDRH：UART1 在 DMA 工作模式下，发送数据 XRAM 缓存区的起始地址高位位。

与 UTXDMAADDRL 组合可寻址 Xram 的范围 0x00-0x600（1536Byte）

注：DMA 模式下，发送数据缓存只能设在 XRAM 区域 0x00-0x600（1536Byte）。

URXDMACR：UART1 接收模式配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
RXLEN6	RXLEN5	RXLEN4	RXLEN3	RXLEN2	RXLEN1	RXLEN0	RXMODE
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 RXLEN[6:0]：UART1 工作在 DMA 模式下，设定发送 Xram 缓存区空间大小，大小为 RXLEN+1(最大 128byte)

bit 0 RXMODE：接收模式选择

0 = 普通模式。

1 = DMA 模式。

URXDMAADDRL： UART1 DMA 模式下的接收数据起始地址低位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 URXDMAADDRL：UART1 工作在 DMA 模式下，接收数据缓存 XRAM 区的起始地址低位。

与 URXDMAADDRH 组合可寻址 Xram 范围 0x00-0x600（1536Byte）

URXDMAADDRH： UART1 DMA 模式下的接收数据起始地址高位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 URXDMAADDRH：UART1 工作在 DMA 模式下，接收数据缓存 XRAM 区的起始地址高位。

与 URXDMAADDRL 组合可寻址 Xram 范围 0x00-0x600（1536Byte）

注：DMA 模式下，接收数据缓存只能设在 XRAM 区域 0x00-0x600（1536Byte）。

17.7、UART2 寄存器的定义

表 17-2、UART2 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
UCR2	0x18FF	XRAM	UART2控制寄存器	0000 0100
USETR2	0x19FF	XRAM	UART2数据格式控制寄存器	0000 1000
UBPRL2	0x1AFF	XRAM	UART2波特率配置寄存器低8位	1111 1111
UBPRH2	0x1BFF	XRAM	UART2波特率配置寄存器高8位	1111 1111
UIER2	0x1CFF	XRAM	UART2中断使能寄存器	0000 0000
USR2	0x1DFF	XRAM	UART2状态寄存器	0101 0000
UTX2DR	0x1F00	XRAM	UART2发送数据寄存器	0000 0000
URX2DR	0x1F01	XRAM	UART2接收数据寄存器	0000 0000
UTX2DMACR	0x1F02	XRAM	UART2发送模式选择寄存器	0000 0000
UTX2DMAADDRL	0x1F03	XRAM	UART2 XRAM 模式发送数据起始地址低位	0000 0000
UTX2DMAADDRH	0x1F04	XRAM	UART2 XRAM 模式发送数据起始地址高位	0000 0000
URX2DMACR	0x1F05	XRAM	UART2接收模式选择寄存器	0000 0000
URX2DMAADDRL	0x1F06	XRAM	UART2 XRAM 模式接收数据起始地址低位	0000 0000
URX2DMAADDRH	0x1F07	XRAM	UART2 XRAM 模式接收数据起始地址高位	0000 0000

以上表中寄存器地址 0x1F00~0x1F07 对应的寄存器均不支持按位操作，只能对整个寄存器进行字节的操作；其它寄存器支持按位操作或直接对字节操作。

UART2 所有寄存器的定义及应用与 UART1 对应寄存器定义完全一致，请参考对应 UART1 寄存器的定义。

18、SPI 模块

SPI 模块允许同时同步发送和接收 8 位数据。支持 SPI 的所有 4 种时序，使用以下 4 个引脚来完成通信：

- SPI 数据输出
- SPI 数据输入
- SPI 时钟信号
- SPI 片选信号

SPI 模块发送和接收都有 4 Byte 深度的数据缓冲器，在启动一组数据的传送前，必须软件写 SPICR 寄存器的 FIFOCLR 位为 1 清空数据缓冲器，防止 FIFO 中存在以前未传送完的数据。FIFOCLR 位为只写位，读为 0。

18.1、SPI 的操作步骤

配置 SPI 的操作步骤如下：

1、端口配置

SPI 有主控和从动两种工作模式，所以存在两组端口配置方式：

1)、SPI 配置为主控模式时，可通过 PxyIOCFG 寄存器将 I/O 配置成 SPIM 数据输入、SPIM 数据输出、SPIM 片选输出、SPIM 时钟输出功能，必须先选择好模式才能再配置其它寄存器，否则配置无效。

2)、SPI 配置为从动模式时，可通过 PxyIOCFG 寄存器将 I/O 配置 SPIS 数据输入、SPIS 数据输出、SPIS 片选输入、SPIS 时钟输入功能，必须先选择好模式才能再配置其它寄存器，否则配置无效。

只要配置好后，会自动将引脚方向配置为相应的方向，不需要再配置对应的 PnOE 方向寄存器。(端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1)。

2、SPI 模块时钟使能

在操作 SPI 模块相关寄存器前，必须首先使能 SPI 模块时钟，即将 MDLCKCR 寄存器的 SPIEN 位置 1。

3、SPI 模块参数配置

- 主从模式选择

SPI 有主控和从动两种工作模式。通过 SPICR 寄存器的 MSSEL 位控制。 必需注意的是 SPICR 的 MSSEL 位与其他位不能同时配置，必需先写 SPICR 的 MSSEL 位配置好模式后，才能配置其他位。

- 时钟速率(仅限主控模式)

时钟线是由主器件控制的，所以在主控模式下，时钟速率的配置才有效。当作为主控模式时，SPIBPR 寄存器可以控制 SPI 通讯的波特率。具体计算方法如下：

SPI 每个 bit 的时间 = 2 * (SPIBPR + 1) * Tspi;

其中 Tspi 为 SPI 的时钟周期，SPI 时钟是系统副时钟 SCLK1。SPIBPR 寄存器的值必须大于等于 0x06(SCLK1=16M)，spi 的时钟速度必须<=1.14Mhz。

- 片选信号产生方式选择

当 SPI 为主控模式时，可以通过 SPICR 寄存器的 CSMODE 位选择片选信号产生方式，有两种方式：

1)、硬件产生片选信号，此时通讯过程中，硬件自动产生拉高、拉低片选信号。

2)、软件产生片选信号，此时通过 SPISFMR 寄存器控制片选信号拉高、拉低。必须在时钟信号发送前将片选信号拉低，数据传送完成后再将片选信号拉高。若传送中片选拉高则会出错。

- 发送接收位顺序配置

SPICR 寄存器的 DORD 位清 0，选择 LSB 方式，发送时，先发送字节低位，接收时，先接收到的为字节低位。

SPICR 寄存器的 DORD 位置 1，选择 MSB 方式，发送时，先发送字节高位，接收时，先接收到的为字节高位。

- 时钟极性、时钟相位配置

SPI 有 4 中不同的时序，通过 SPICR 寄存器的 CPHA、CPOL 位，可配置成不同的通讯时序，CPHA 位配置时钟相位(采样的时间，上升沿或下降沿采样)，CPOL 位配置时钟极性(时钟线的空闲状态)，如下图 18-1、18-2 所示：

4、SPI 模块中断控制

SPIIER 寄存器是 SPI 中断使能寄存器，有 SPIIER[4:0]5 个位与 SPISR[4:0]一一对应，分别是发送 FIFO 空中断、发送 FIFO 满中断、接收 FIFO 非空中断、接收 FIFO 半满中断、接收 FIFO 满中断。SPISR[4:0]就是相应的中断标志位。不论中断是否使能，只要产生中断条件，相应的标志位就会置 1。

SPISR 寄存器还有 RXFOV(接收 FIFO 溢出标志位)、FWRSR(发送 FIFO 写状态位)、TRANS(SPI Master 数据传送启动位)。这些都只是标志位，不会产生中断。详细请查看寄存器介绍。

5、外设模块中断控制

使能寄存器 IRQ5ER 的 bit2 位，IRQ5SR_SPIINT = 0x04，即使能 SPI 外设模块的中断。

6、内核中断控制

1)、EX5 = 1；//使能内核中断 5

2)、配置内核中断 5 中断优先级 IP2_BIT[3:2]（0：低优先级；1：高优先级）

3)、使能总内核中断 EA=1。

7、接收发送使能

当参数都配置好以后，可以将 SPICR 寄存器的 RXEN 位置 1 使能数据接收，TXEN 置 1 使能数据发送。数据的发送和接收功能是相互独立的，不会受另一方的影响。

SPI 通讯时序图(CPHA = 0)

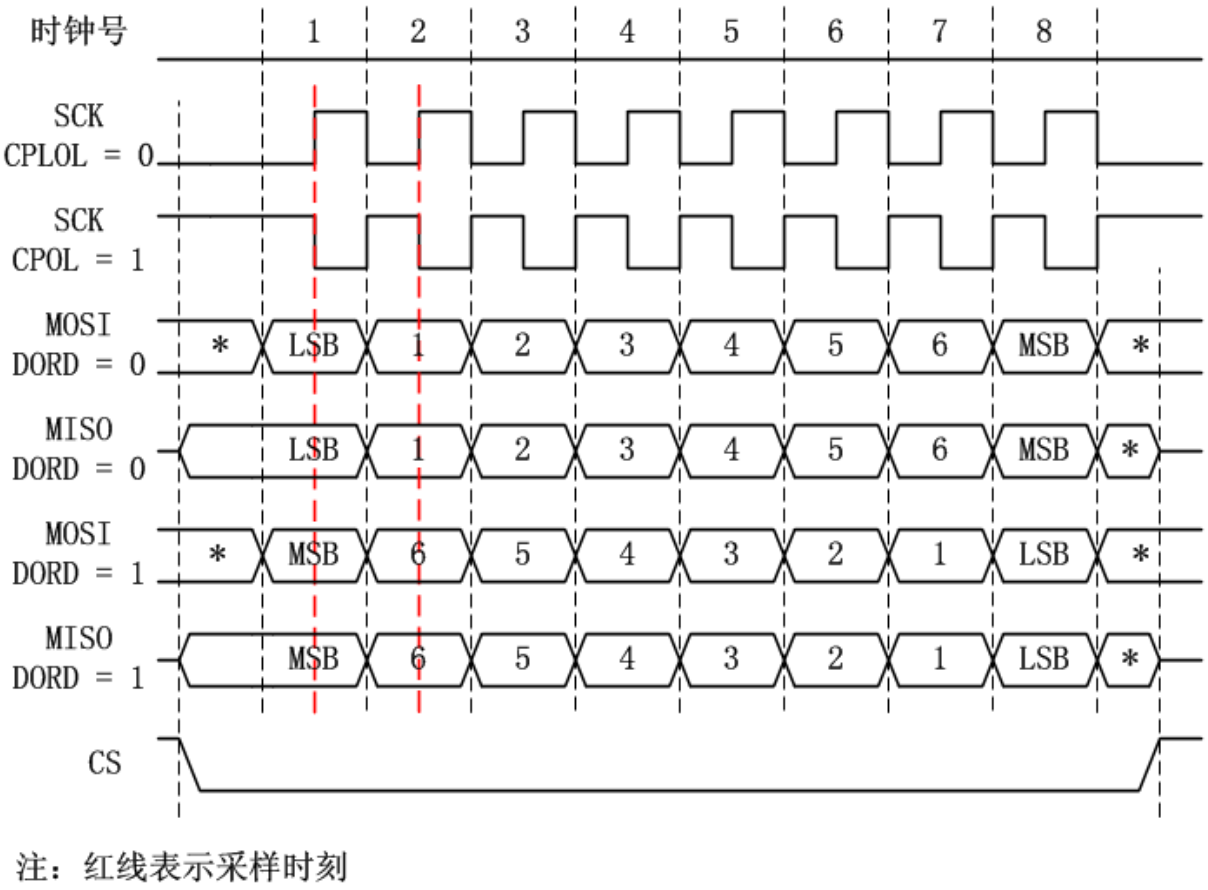


图 18-1、SPI 通讯时序图(CPHA = 0)

SPI 通讯时序(CPHA = 1)

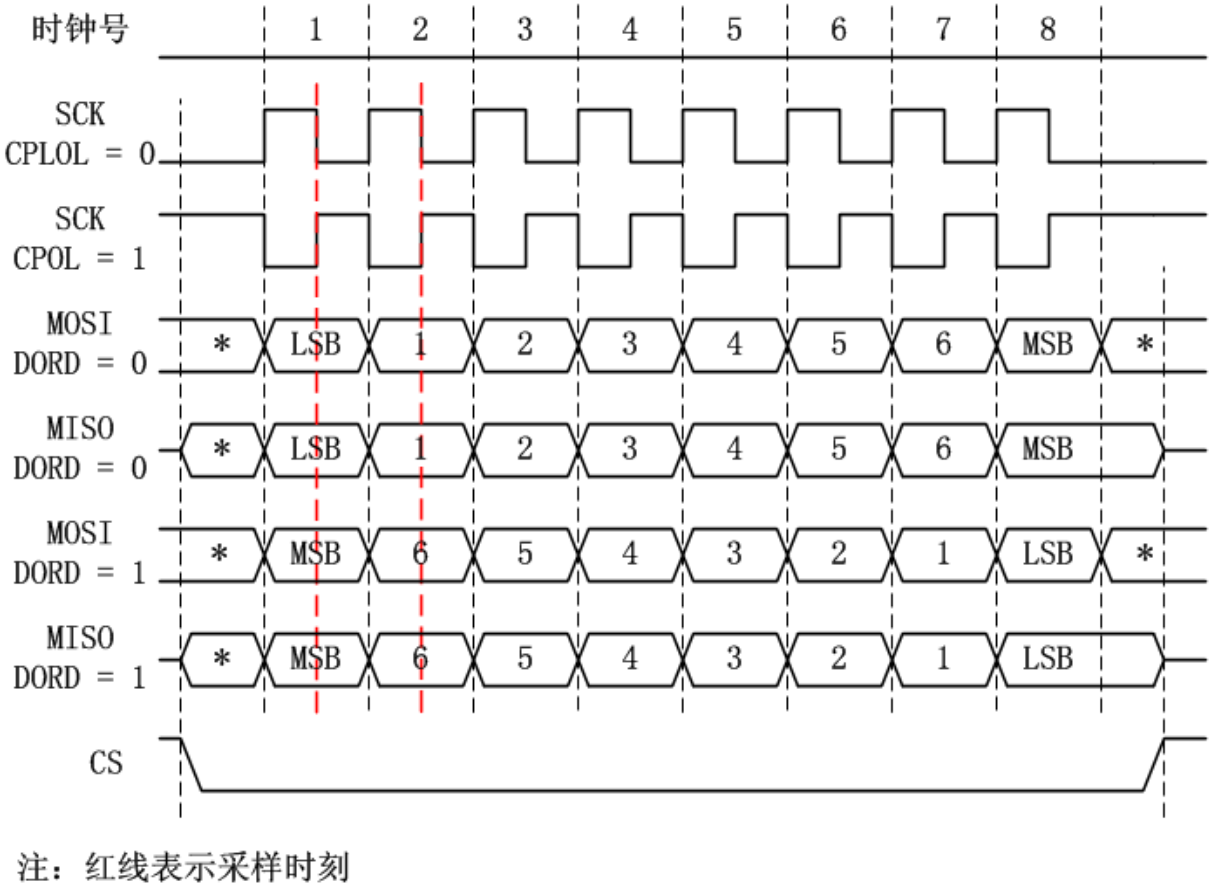


图 18-2、SPI 通讯时序图(CPHA = 0)

18.2、主控模式

主器件控制时钟线，因此可以随时启动数据传输。主器件根据软件协议确定从器件何时传输数据。

1）、数据发送：

在主控模式下，软件通过写 SPIDR 寄存器，将数据写入到发送 FIFO 中。最多可写 4 字节数据，FIFO 已满将不能再写入数据。软件应先查询发送 FIFO 满标志位为 0，再写 SPIDR。

数据写入发送 FIFO 后，将 SPISR 寄存器的 TRANS 位置 1 即启动 1Byte 的数据传送，软件查询 TRANS 位为 0 时，即表示 Byte 传输结束。必须等待 TRANS 位为 0 后，才能再次将 TRANS 位置 1 启动下次 Byte 传输。每完成 1Byte 数据传送，发送 FIFO 的数据都会减少 1Byte，SPISR 寄存器的相应标志位也会做出改变。

2）、数据接收：

在主控模式下，接收的数据先存入到接收 FIFO 中，深度也为 4Byte，软件读 SPIDR 寄存器，可将接收 FIFO 中的数据取出。

每读一次 SPIDR 寄存器，接收 FIFO 中缓存的数据都会减少 1Byte，SPISR 寄存器的相应标志位也会做出改变。软件上先查询接收 FIFO 非空标志位为 1，再读取 SPIDR。如果接收 FIFO 已满，SPI 仍继续接收数据，则会溢出。

3）、时钟速度：

理论上 SPI 的时钟速度可达到 1.14Mhz，但是需要通过相关寄存器来配置。

18.3、从动模式

在从动模式下，当时钟引脚上出现外部时钟脉冲时，即启动数据发送和接收。当最后一位数据被锁存后，SPISR 寄存器的 RXFNOTEMPTY、RXFHALF、RXFFULL 位会产生相应变化。

1）、数据发送和接收：

在从动模式下，发送接收数据与主控模式相同，区别是，从动模式不能控制何时通讯，只能被动的响应主器件的数据传送。

2）、片选信号：

从动模式下，必须接收到片选信号，才会响应数据传送。就是说只有片选引脚为低电平时，才能发送和接收数据。

注：软件仅在 SPISR 寄存器的 FWRSR 位为 0 时，才可以写 SPIDR 寄存器将待发送数据写入发送 FIFO 中。

18.4、SPI 寄存器的定义

表 18-1、SPI 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
SPICR	0xE0FF	XRAM	SPI 控制寄存器	0000 0000
SPISFMR	0xE1FF	XRAM	SPI 软件控制片选信号寄存器	0000 0001
SPIIER	0xE2FF	XRAM	SPI 中断使能寄存器	0000 0000
SPIBPR	0xE3FF	XRAM	SPI 波特率配置寄存器	0010 0000
SPISR	0xE4FF	XRAM	SPI 状态寄存器	0000 0001
SPIDR	0xE5FF	XRAM	SPI 数据接收发送寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

SPICR：SPI 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
MSSEL	CSMODE	DORD	FIFOCLR	CPHA	CPOL	TXEN	RXEN
R/W-0	R/W-0	R/W-0	W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 MSSEL：主从模式控制位

0 = 从模式

1 = 主模式

注：必须先选择好模式才能再配置其它寄存器。

bit 6 CSMODE：片选信号产生方式选择位

0 = 硬件方式产生，通讯过程中，每个字节的通讯，CS 都会自动变低再变高。

1 = 软件方式产生

此后，软件写 SPISFMR 为1即拉高 CS，写0即拉低 CS。

注：软件方式可以实现 CS 保持为低时，完成多个字节通讯

bit 5 DORD：发送接收位顺序控制位

0 = LSB 方式

1 = MSB 方式

bit 4 FIFOCLR：清 FIFO 位

只写位，软件写 1 清空 FIFO

bit 3 CPHA：时钟相位选择位

0 = 在时钟的第一个边沿采样数据

1 = 在时钟的第二个边沿采样数据

bit 2 CPOL：时钟极性选择位

0 = 时钟的空闲电平为低电平

1 = 时钟的空闲电平为高电平

bit 1 TXEN：SPI 发送数据使能位

0 = 禁止发送

1 = 使能发送

bit 0 RXEN：SPI 接收数据使能位

0 = 禁止接收

1 = 使能接收

SPISFMR：SPI 软件控制片选信号寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	-	CS
U-x	U-x	U-x	U-x	U-x	U-x	U-x	R/W-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 保留未用

bit 0 CS：SPI 软件控制片选信号寄存器

0 = 拉低片选信号

1 = 拉高片选信号

SPIIER：SPI 中断使能寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	RXFOV	RXFFULL	RXFHALF	RXFNOTEMPTY	TXFFULL	TXFEMPTY
U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-6 保留未用

bit 5 RXFOV：接收 FIFO 溢出标志位

0 = 禁止接收 FIFO 未溢出

1 = 使能 FIFO 已经溢出

bit 4 RXFFULL：接收 FIFO 满中断使能位

0 = 禁止接收 FIFO 满中断

1 = 使能接收 FIFO 满中断

bit 3 RXFHALF：接收 FIFO 半满中断使能位

0 = 禁止接收 FIFO 半满中断

1 = 使能接收 FIFO 半满中断

bit 2 RXFNOTEMPTY：接收 FIFO 非空中断使能位

0 = 禁止接收 FIFO 非空中断

1 = 使能接收 FIFO 非空中断

bit 1 TXFFULL：发送 FIFO 满中断使能位

0 = 禁止发送 FIFO 满中断

1 = 使能发送 FIFO 满中断

bit 0　　TXFEMPTY：发送 FIFO 空中断使能位

0 = 禁止发送 FIFO 空中断

1 = 使能发送 FIFO 空中断

SPIBPR：SPI 波特率配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-							
U-x	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7　　保留未用

bit 6-0　　SPIBPR[6:0]:SPI 波特率配置寄存器,SPIBPR 必须大于等于 0x06(SCLK1=16M),spi 的时钟速度必须<=1.14Mhz。。

当 SPI 作为主控模式时，SPIBPR 控制 SPI 通讯的波特率。

SPI 每个 bit 的时间 = 2 * (SPIBPR + 1) * Tspi

(Tspi 是等于系统副时钟 SCLK1)

SPIDR：SPI 数据接收发送寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　SPIDR<7:0>：SPI 数据接收发送寄存器

接口发送数据时，软件通过写此寄存器，将数据写入到发送 FIFO 中。

接口接收数据时，数据首先存入到接收 FIFO 中，软件读此寄存器，将接收 FIFO 中的数据取出。

SPISR：SPI 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TRANS	FWRSR	RXFOV	RXFFULL	RXFHALF	RXFNOTEMPTY	TXFFULL	TXFEMPTY
R/W-0	R-0	R/W-0	R-0	R-0	R-0	R-0	R-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7　　TRANS：主控模式下，Byte 数据传送启动位

0 = 数据传送已完成或不在进行中

1 = 启动数据传送，完成后自动清 0

bit 6　　FWRSR：SPI 发送 FIFO 写状态位

0 = 可写 SPIDR 寄存器将数据写入发送 FIFO。

1 = 不可写 SPIDR 寄存器将数据写入发送 FIFO。

bit 5　　RXFOV：接收 FIFO 溢出标志位

0 = 接收 FIFO 未溢出

1 = 接收 FIFO 已经溢出，需软件写 0 清 0

bit 4　　RXFFULL：接收 FIFO 满标志位

0 = 接收 FIFO 不满

1 = 接收 FIFO 已满

bit 3 RXFHALF：接收 FIFO 半满标志位

0 = 接收 FIFO 没有半满

1 = 接收 FIFO 半满

bit 2 RXFNOTEMPTY：接收 FIFO 非空标志位

0 = 接收 FIFO 空

1 = 接收 FIFO 非空

bit 1 TXFFULL：发送 FIFO 满标志位

0 = 发送 FIFO 不满

1 = 发送 FIFO 已满

bit 0 TXFEMPTY：发送 FIFO 空标志位

0 = 发送 FIFO 非空

1 = 发送 FIFO 已空

注：

1、发送或接收每个 Byte 都需要写 TRANS 位。软件查询 TRANS 位为 0 时，即表示 Byte 传输结束。

2、软件仅在 FWRSR 位为 0 时，才可以写 SPIDR 寄存器将待发送数据写入发送 FIFO 中。

3、当 RXFFULL 位为 1，SPI 继续接收到新的 Byte 数据，则会溢出，RXFOV 位会置 1。所以软件应及时将接收 FIFO 中的数据读出。

19、I2C 模块

I²C 模块包含主控和从动两个模块，这两个模块是相互独立的，可以同时工作。

I²C 模块支持 7 位寻址功能即 I2CSADDR_bit[7:0]；两根传输通迅信号线，分别是时钟线和数据线。当 IO 复用功能寄存器配置为 I²C 功能(I²C Slave 数据线、I²C Slave 时钟线；I²C Master 数据线、I²C Master 时钟线)后，会自动将引脚方向配置为相应的方向，不需要再配置对应的 PnOE 寄存器。

19.1、I2C Slave(从动模块)

I²C Slave 有 4 种工作方式：

- I²C Slave 软件快速模式(读写)《常用模式》
- I²C Slave 软件标准模式(读写)
- I²C Slave 硬件快速模式(只读)
- I²C Slave 硬件标准模式(只读)

这 4 种模式下的通讯时序要求有所不同。在软件快速、软件标准模式下，主机可以对从机进行读写。而在硬件快速、硬件标准模式下，主机只能从从机读数据而不能对从机写数据。具体的通讯时序见对应小节。

注：

1、以上所说的标准模式和快速模式是指 I2C 的工作方式（请参下考图），并非指 I2C 的传输速率级别；

2、标准模式：指必须严格按照读写 EEPROM 器件的 I2C 接口设备协议来操作，此协议中必须有从机设备地址、从机硬件地址（即从机的内存地址）；

3、快速模式：指通迅协议中不需要从机设备硬件地址，只需要从机设备地址匹配即可进行正常数据收发通迅。

I²C Slave 的操作步骤

配置 I²C Slave 的操作步骤如下：

1、端口配置

1)、配置对应 IO 复用功能寄存器，将 IO 口重映射为 I²C Slave 数据线、I²C Slave 时钟线功能（无需再配置 PnOE）。

2)、配置对应 PnPURSELRH：PnPURSELRL 寄存器可以选择对应时钟、数据线的内部上拉电阻，建议配置为 10K 上拉。

3)、配置对应 PnPU 寄存器，使能对应时钟、数据线的内部上拉电阻。

注：端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1。

2、I²C Slave 模块参数配置

1)、从机设备地址配置

主机发送开始信号后，紧接发 7 位地址+1 位读/写位，主机发的这 7 位地址数据必须与从机 I2CSADDR 寄存器的 bit[7:1]位匹配，从机才会做出应答。通过配置 I2CSADDR 寄存器为不同的值，可以实现多机通讯(一主多从)。I2CSADDR 的 bit0 位是记录主机读写命令状态位，0 表示写，1 表示读。

2)、从机硬件地址配置(在硬件标准模式下)

硬件地址配置只在硬件标准模式下有作用。只有在硬件标准模式下，当主机发送的设备地址匹配后，需要接着发送 8 位硬件地址。当这 8 位硬件地址数据与从机的 I2CSHADR 寄存器匹配时，从机才会做出应答。

3)、待发送数据配置

主机读从机数据时，从机必须先把待发送数据写入内部数据寄存器中。通过 I2CSDR 数据缓存寄存器，将待发送数据写入到内部数据寄存器中。

- 硬件模式下，内部数据缓存寄存器为 4 个字节：

  依次将 0、1、2、3 位数据字节写到 I2CSDR 寄存器；

- 软件模式下，内部数据缓存寄存器为 2 个字节：

  首次写时，依次将第 0、1 位数据字节写到 I2CSDR 寄存器；之后在每传输完一个字节产生发送中断（或查询到发送完成中断标志）时再往 I2CSDR 寄存器写入下一个字节数据。

4)、工作模式配置

I2C Slave 有 4 种工作模式，通过 I2CSCR 寄存器的 MODE<1:0>位可以配置 I2C Slave 的工作模式：

- MODE<1:0> = 00；软件标准模式（可读写）
- MODE<1:0> = 01；软件快速模式（可读写）《常用模式》
- MODE<1:0> = 10；硬件标准模式（只读）
- MODE<1:0> = 11；硬件快速模式（只读）

3、清 I²C Slave 中断标志位

使能 I²C Slave 中断前，先将 I2CSSR 寄存器清 0，清 I²C Slave 相关中断标志位。

4、中断控制

I2CSCR 寄存器的 INTEN 位是 I2C Slave 中断使能位，有 INTEN[3:0]4 个位与 I2CSSR 寄存器的[3:0]位一一对应，分别是 I2C Slave 接收到硬件地址中断、I2C Slave 接收到字节数据中断、I2C Slave 发送完字节数据中断、I2C Slave 接收到设备地址中断(只在软件快速模式下产生)。I2CSSR[3:0]就是相应的中断标志位，不论中断是否使能，只要产生中断条件，相应的标志位就会置 1。

需要注意的是，在不同模式下，产生不同的中断：

- 软件快速模式下，只产生 I2C Slave 接收设备地址中断、接收完数据字节中断、发送完数据字节中断；《常用模式》
- 软件标准模式下，只产生 I2C Slave 接收硬件地址中断（即内存地址中断）、I2C Slave 接收完数据字节中断、I2C Slave 发送完数据字节中断。
- 硬件快速模式下，不产生中断；
- 硬件标准模式下，只产生 I2C Slave 接收硬件地址中断（即内存地址中断），接收到的硬件地址保存在 I2CSRHADR 寄存器中；

5、外设模块中断控制

使能寄存器 IRQ5ER 的 bit1 位，IRQ5ER_I2CSINT = 0x02，即使能 I2C Slave 外设模块的中断。

6、内核中断控制

1)、EX5 = 1；//使能内核中断 5

2)、配置内核中断 5 中断优先级 IP2_bit[3:2]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1

7、使能响应主机读写数据

I2CSCR 寄存器的 ACKEN 位是使能从机 CBM73xxF 响应主机读写数据控制位，只有将 ACKEN 置为 1，从机才会响应主机的命令；否则就算时序正确，从机也不会回应主机。

I2C Slave 软件快速模式(可读写)《常用模式》

软件读操作步骤：

1、I2C Slave 配置成软件快速模式，使能响应主机读写；

2、从机使能接收到设备地址中断、使能发送完成中断；

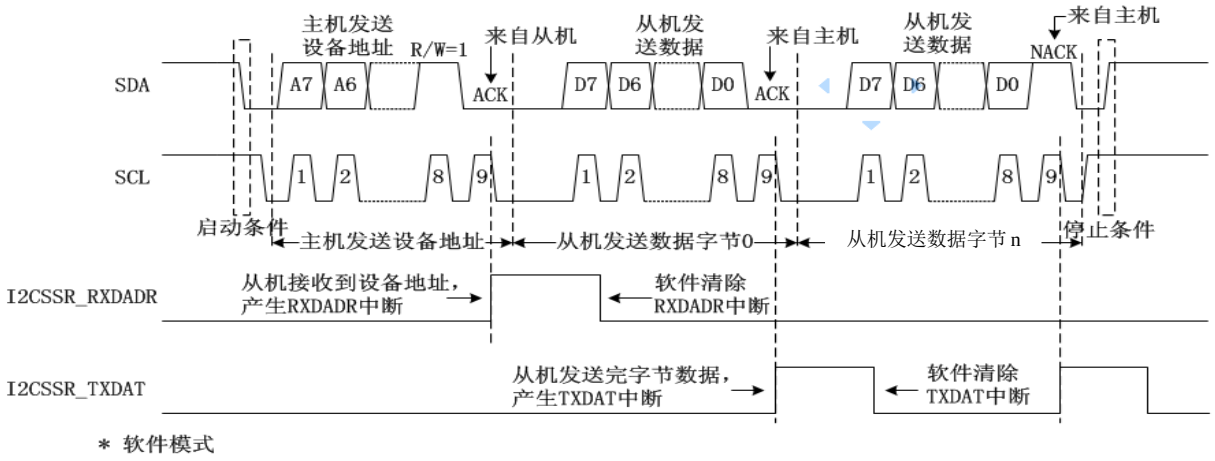
3、I2C 主机发送匹配的从机设备地址和读标志，即可从 Slave 开始读取字节数据；

4、从机产生接收到设备地址中断时，将预发送数据写入到内部数据缓存寄存器 I2CSDR 中(第一次写入 2 个待发字节数据)。

5、主机延时小段时间等待 4 步骤完成，然后开始从从机读数据；

6、从机每次产生发送数据完成中断时(通过查询到发送完数据中断标志位)，将下一个待发送数据写入到内部数据缓存寄存器 I2CSDR 中。软件快速模式读通讯时序如下图所示。

软件快速模式主机读时序图(读)



19-3、软件快速模式主机读时序图(读)

软件写操作步骤：

1、I2C Slave 配置成软件快速模式，使能响应主机读写；

2、使能接收到设备地址中断、使能接收数据完成中断；

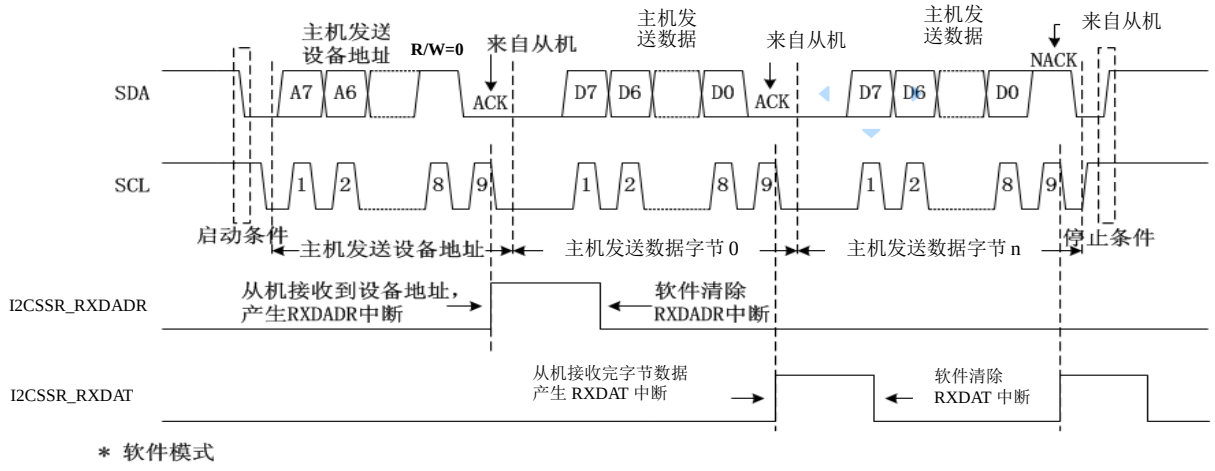
3、I2C 主机发送匹配的设备地址和读标志，即可往 Slave 开始写字节数据；

4、从机产生接收到设备地址中断时，表示主机将会发送数据给从机，从机初始化数据 Buffer 准备接收后续主机发送来的数据；

5、主机延时小段时间等待 4 步骤完成，开始继续往从机写数据；

6、从机产生接收数据完成中断时，通过查询接收完字节数据中断标志位，通过读 I2CSDR 寄存器，将主机发来的数据保存下来。软件快速模式写通讯时序如下图所示。

软件快速模式主机写时序图(写)



19-3、软件快速模式主机写时序图(写)

I2C Slave 软件标准模式(可读写)

软件操作步骤：

1、I2C Slave 配置成软件标准模式，使能响应主机读写；

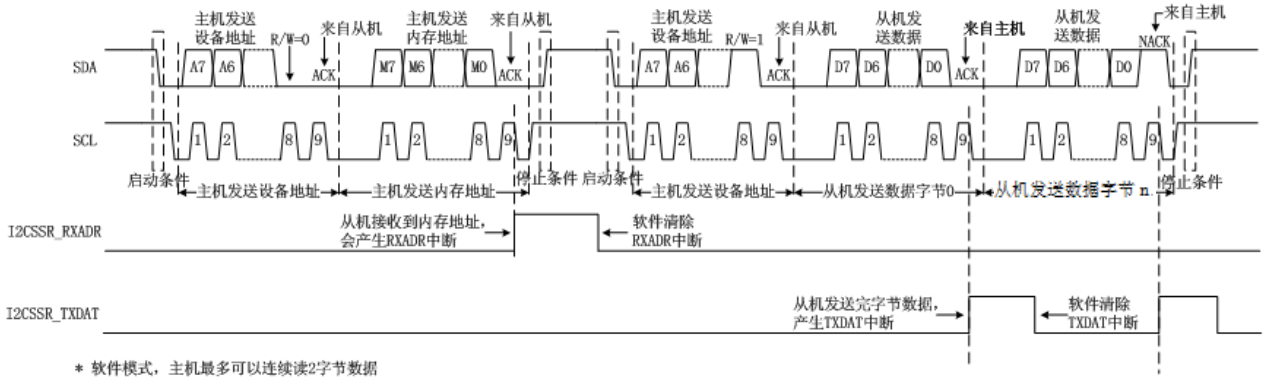
2、使能接收到地址中断；

3、通过 I2CSDR 寄存器，将预发送数据写入到内部数据寄存器中。最多可以写入 2 字节数据；

4、若主机执行写操作，I2C 主机发送匹配的设备地址和写标志，再送入内存地址，再紧跟待写入数据。从机中断触发后，根据接收到的内存地址的值以及接收数据，将数据存入到对应的地址中；

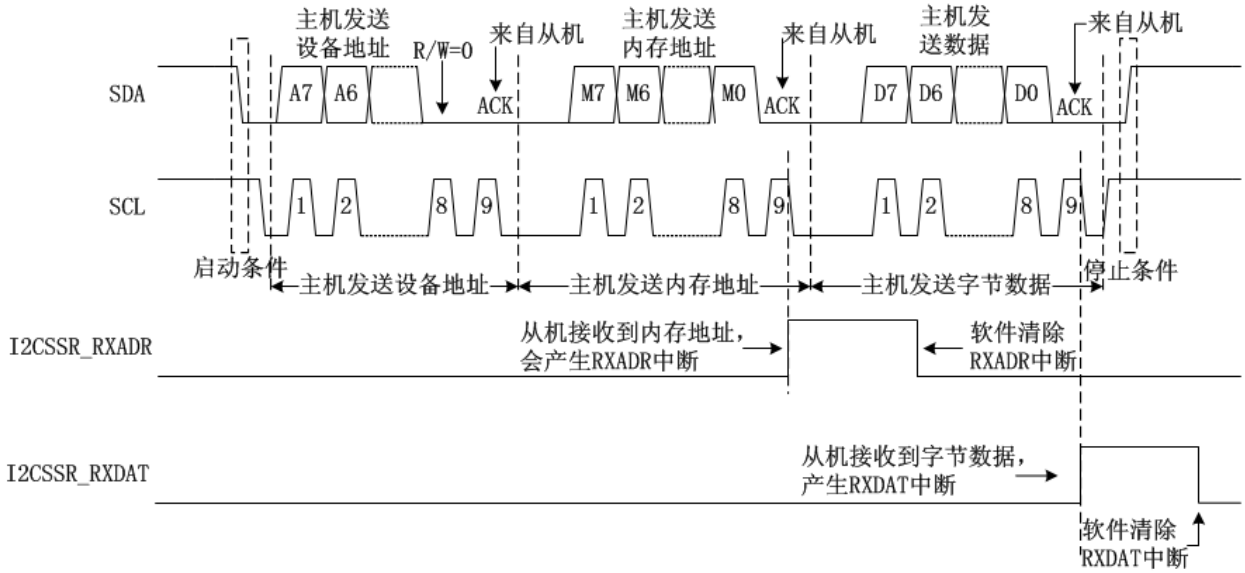
6、若主机执行读操作，I2C 主机发送匹配的设备地址和写标志，再送入内存地址。然后主机再次发送匹配的设备地址和读标志。从机中断触发后，通过查询发送完字节数据中断标志位，根据接收到的内存地址的值，调配相应的数据，通过 I2CSDR 寄存器，将后续待发送数据写入到内部数据寄存器中。

软件标准模式主机读时序图



19-4、软件标准模式主机读时序图

软件标准模式主机写时序图



19-5、软件标准模式主机写时序图

I2C Slave 硬件快速模式(只读)

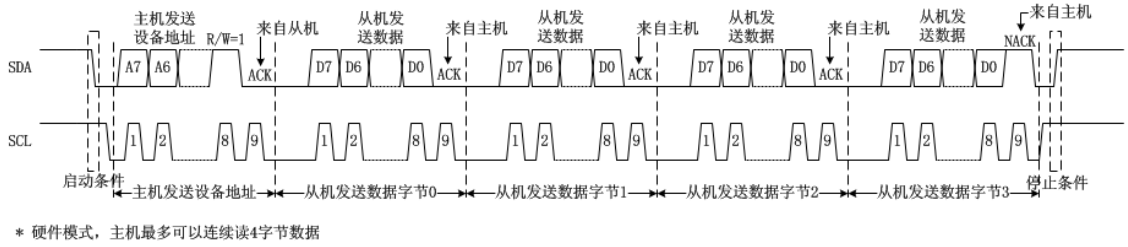
软件操作步骤：

1、I²C Slave 配置成硬件快速模式，使能响应主机读写；

2、通过 I2CSDR 寄存器，将预发送数据写入到内部数据寄存器中，最多可写入 4 字节数据；

3、I²C 主机发送匹配的设备地址和读标志，即可从 Slave 读取最多 4 个字节数据。

硬件快速模式主机读时序图(只读)



19-1、硬件快速模式主机读时序图(只读)

I2C Slave 硬件标准模式(只读)

软件操作步骤：

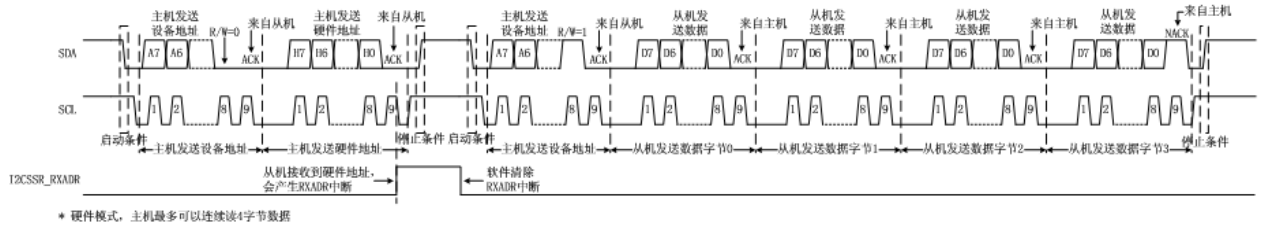
1、I2C Slave 配置成硬件标准模式，使能响应主机读写；

2、配置 I2CSHADR 寄存器；

3、通过 I2CSDR 寄存器，将预发送数据写入到内部数据寄存器中，最多可写入 4 字节数据。

4、I2C 主机发送匹配的设备地址和写标志，再送入硬件地址（主机发送的硬件地址必须要与从机软件配置的 I2CSHADR 寄存器一致），然后主机再次发送匹配的设备地址和读标志，即可从 Slave 读取最多 4 个字节数据。

硬件标准模式主机读时序图(只读)



19-2、硬件标准模式主机读时序图(只读)

I2C Slave 寄存器的定义

表 19-1、I2C Slave 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
I2CSCR	0xD8FF	XRAM	I2C Slave 控制寄存器	0000 0000
I2CSSR	0xD9FF	XRAM	I2C Slave 状态寄存器	0000 0000
I2CSDR	0xDBFF	XRAM	I2C Slave 数据寄存器	0000 0000
I2CSADDR	0xDCFF	XRAM	I2C Slave 设备地址寄存器	0010 0010
I2CSRHADR	0xDDFF	XRAM	I2C Slave 接收硬件地址寄存器	0000 0000
I2CSHADR	0xDEFF	XRAM	I2C Slave 硬件地址寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

I2CSCR：I2C Slave 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	MODE1	MODE0	ACKEN	RXIDADDRINTEN	TXDATINTEN	RXDATINTEN	RXHADDRINTEN
U-x	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7　保留未用

bit 6-5　MODE[1:0]：I²C Slave 工作模式模式选择位

00 = 软件标准模式

01 = 软件快速模式

10 = 硬件标准模式

11 = 硬件快速模式

bit 4　ACKEN：响应主机读写使能位

0 = 禁止响应主机读写

1 = 使能响应主机读写

bit 3　RXIDADDRINTEN：I²C Slave 接收到设备地址中断使能位(只在软件快速模式下产生)

0 = 禁止 I²C Slave 接收到设备地址中断

1 = 使能 I²C Slave 接收到设备地址中断

bit 2　TXDATINTEN：I²C Slave 发送完字节数据中断使能位(只在软件快速、软件标准模式下产生)

0 = 禁止 I²C Slave 发送完字节数据中断

1 = 使能 I²C Slave 发送完字节数据中断

bit 1　RXDATINTEN：I²C Slave 接收到字节数据中断使能位

0 = 禁止 I²C Slave 接收到字节数据中断

1 = 使能 I²C Slave 接收到字节数据中断

bit 0　RXHADDRINTEN：I²C Slave 接收到硬件地址(即内存地址)中断使能位

0 = 禁止 I²C Slave 接收到硬件地址中断

1 = 使能 I²C Slave 接收到硬件地址中断

I2CSSR：I2C Slave 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	RXIDADDR	TXDATA	RXDATA	RXHADDR
U-x	U-x	U-x	U-x	R/W-0	R/W-0	R/W-0	R/W-0
注：							

R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）

bit 7-5 保留未用

bit 4 SENDBUSY：I²C Slave 发送忙标志

0 = 空闲

1 = 正在发送数据

bit 3 RXIDADDR：I²C Slave 接收到设备地址中断标志位(只在软件快速模式下产生)

0 = I²C Slave 未接收到设备地址

1 = I²C Slave 接收到了设备地址，需软件写 0 清 0

bit 2 TXDATA：I²C Slave 发送完字节数据中断标志位(只在软件快速、软件标准模式下产生)

0 = I²C Slave 字节数据还未发送完

1 = I²C Slave 字节数据发送完成，需软件写 0 清 0

bit 1 RXDATA：I²C Slave 接收到字节数据中断标志位

0 = I²C Slave 未接收到字节数据

1 = I²C Slave 接收到了字节数据，需软件写 0 清 0

bit 0 RXHADDR：I²C Slave 接收到硬件地址(即内存地址)中断标志位

0 = I²C Slave 未接收到地址

1 = I²C Slave 接收到了硬件地址，需软件写 0 清 0

I2CSDR：I2C Slave 数据寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 I2CSDR[7:0]：I²C Slave 数据寄存器

将待发送数据写入到内部数据寄存器，是通过 I2CSDR 寄存器的配合来实现的。详细请查看 18.1.1.4 小节“待发送数据配置”

I2CSADDR：I2C Slave 设备地址寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0	WEN
R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-1 ADDR[6:0]：I²C Slave 设备地址位

配置 I²C 设备地址，只有当 I²C 主设备发送的设备地址与此寄存器数值匹配时，才能正常通讯。

bit 0 WEN：从机接收到 I²C 主机发送来的读写状态位

0 = I²C 主机执行写操作

1 = I²C 主机执行读操作

I2CSRHADR：I2C Slave 接收硬件地址寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 I2CSRHADR[7:0]：I²C Slave 接收硬件地址寄存器

I²C 主设备发送的硬件地址会保存在 I2CSRHADR 中，软件可以读取 I2CSRHADR 获得硬件地址值。

I2CSHADR：I2C Slave 硬件地址寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 I2CSHADR[7:0]：I²C Slave 硬件地址寄存器

只有在硬件标准模式下，I²C 主设备发送的硬件地址必须与此寄存器值匹配时才能读取数据。其它模式下配置 I2CSHADR 无效。

19.2、I2C Master(主控模块)

主控模式下所有串行时钟脉冲和启动/停止条件均由主器件 CBM73xxF 产生。

在主控发送模式下，串行数据通过数据线输出，而串行时钟由时钟线输出。发送的第一个字节由接收器件（从机）的设备地址(7 位)和 R/W 位（0：W 表示写；1：R 表示读）组成。在发送模式下，R/W 位将是逻辑 0。因此，发送的第一个字节是一个 7 位从器件设备地址，后面跟 0 表示发送。串行数据每个字节发送 8 位，每发送一个字节，主机会收到一个从机返回的应答位。主机控制启动和停止条件的输出表明串行传输的开始和结束。

在主控接收模式下，串行数据通过数据线接收，而串行时钟由时钟线输出。发送的第一个字节由发送器件（从机）的设备地址(7 位)和 R/W 位（0：W 表示写；1：R 表示读）组成。在接收模式下，R/W 位将是逻辑 1。因此，发送的第一个字节是一个 7 位从器件设备地址，后面跟 1 表示接收。每次接收 8 位串行数据。每接收到一个字节，主机都会发送一个应答位给从机。主机控制启动和停止条件的输出表明串行传输的开始和结束。

在 I²C 主控模式下，可以通过 I2CMPRERH:I2CMPRERL 寄存器配置 I²C 通讯的波特率。

I²C Master 的操作步骤

配置 I²C Master 的操作步骤如下：

1、端口配置

1)、配置 IO 复用功能寄存器，将对应 IO 口配置为 I²C Master 数据线、I²C Master 时钟线后（无需再配置 PnOE）；

2)、配置 PnPURSELRH：PnPURSELRL 寄存器可以选择对应的时钟数据线的内部上拉电阻，建议设置 10K 上拉。

注：端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1。

2、波特率配置

I²C Master 模块的时钟是由 SYSCLK/(SCLK1DIV+1)分频后的时钟（详情时钟框图请参考“4.1、系统总时钟框图”）。通过配置 I2CMPRERH: I2CMPRERL 寄存器的值，可以调整 I²C Master 的通讯波特率。

计算方法如下(SYSCLK 是系统所选择的时钟源)：

I²C Master 波特率 = SYSCLK / ((SCLK1DIV+1) * (I2CMPRERH:I2CMPRERL + 1) * 5)；

例如：SYSCLK 选择内部高速 48M 时钟，SCLK1DIV = 2，I²C Master 波特率= 100KHz；

I2CMPRERH:I2CMPRERL = 48MHz/ (3 * (5*100KHz)) - 1 = 31;

3、清 I²C Master 中断标志位

使能 I²C Master 中断前，先将 I2CMCTR 寄存器的 CLRIF 位置 1，清 I²C Master 中断标志位。

4、中断控制

I2C Master 只有一个中断，若中断使能，每完成 8 位数据传输或者当总线仲裁失败就会产生中断。中断使能位是 I2CMCR

的 IE 位，I2CMSR 寄存器的 IF 位是对应的中断标志位。不论中断是否使能，只要产生中断条件，相应的标志位就会置 1。

I2CMSR 寄存器除了 IF 中断标志位，还有许多标志位，这些只是标志位，不会产生中断。详细请查看寄存器定义。

5、外设模块中断控制

使能寄存器 IRQ5ER 的 bit3 位，IRQ5ER_I2CMINT = 0x08，即使能 I2C Master 外设模块的中断。

6、内核中断控制

1)、EX5 = 1；//使能内核中断 5

2)、配置内核中断 5 中断优先级 IP2_BIT[3:2]（0：最低优先级；3：最高优先级）

3)、使能总内核中断 EA=1

7、主控模式使能

将 I2CMCR 寄存器的 I2CMEN 位置 1，使能 I2C Master 模块功能。使能 I2C Master 后，才能进行数据传输。

6、传输控制

一旦使能主控模式，用户可以进行以下操作：

- 发出一个启动条件
- 产生一个停止条件
- 开始数据/地址的发送
- 开始接收数据
- 在接收到数据字节后产生应答条件

这些操作都是通过 I2CMCTR 寄存器来控制的，但是需要注意的是启动条件、停止条件和应答条件都不是单独控制的，必须和 8 位数据/地址的传输组合在一起。

例如：发出一个启动条件

1)、因为启动条件后面必须接一个 7 位设备地址加一位 R/W 位，所以先把传送的 7 位设备地址+1 位 R/W 位的数据写入 I2CMDR 寄存器。

2)、把 I2CMCTR 寄存器的 START 位、WR 位同时置 1。

3)、当 START、WR 位同时置 1 后，即启动传输。I²C Master 模块就会发出一个启动条件，后面接着发送 7 位设备地址+1 位 R/W 位组成的 8 位数据。传送完成后，START 和 WR 位会自动清 0。

读写例程

以下是 I²C Master 向从机的读写 1 个字节数据的两个范例程序。

例 19-1：向从机写 1 个字节数据

从机地址：0x4E

字节数据：0x65

```
I2CMTRDR = 0x9C; //设备地址+写（0x4E<<1 | 0x00）
I2CMCTR = 0x90; //START+WR
while(0x02 == I2CMSR_TIP);//等待传送完成
if(0x00 == I2CMSR_RXACK)//查询是否收到应答信号
{
    I2CMTRDR = 0x65; //待发送数据
    I2CMCTR = 0x50; //STOP+WR
    while(0x02 == I2CMSR_TIP) ;//等待传送完成
}
```

例 19-2：读取从机 1 个字节数据

从机地址：0x4E

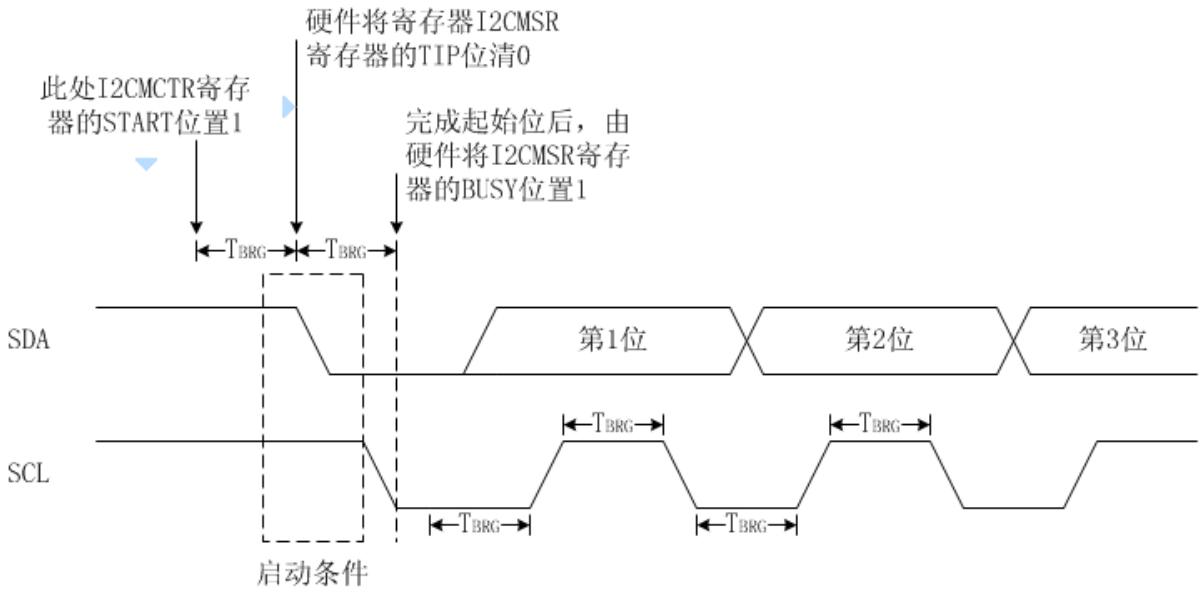
读数据地址：0x20

```
I2CMTRDR = 0x9C; //设备地址+写(0)：0x4E<<1 + 0
I2CMCTR = 0x90; //START+WR
while(0x02 == I2CMSR_TIP);//等待传送完成
if(0x00 == I2CMSR_RXACK) //查询是否收到应答信号
{
    I2CMTRDR = 0x20; //读数据地址
    I2CMCTR = 0x10; //WR
    while(0x02 == I2CMSR_TIP);//等待传送完成
    if(0x00 == I2CMSR_RXACK) //查询是否收到应答信号
    {
        I2CMTRDR = 0x9D; //设备地址+读(1)：0x4E<<1 + 1
        I2CMCTR = 0x90; //START+WR
        while(0x02 == I2CMSR_TIP);//等待传送完成
        if(0x00 == I2CMSR_RXACK) //查询是否收到应答信号
        {
            I2CMCTR = 0x62; //STOP+RD+不发送应答信号
            while(0x02 == I2CMSR_TIP);//等待传送完成
            dat = I2CMTRDR; //保存读到的数据
        }
    }
}
```

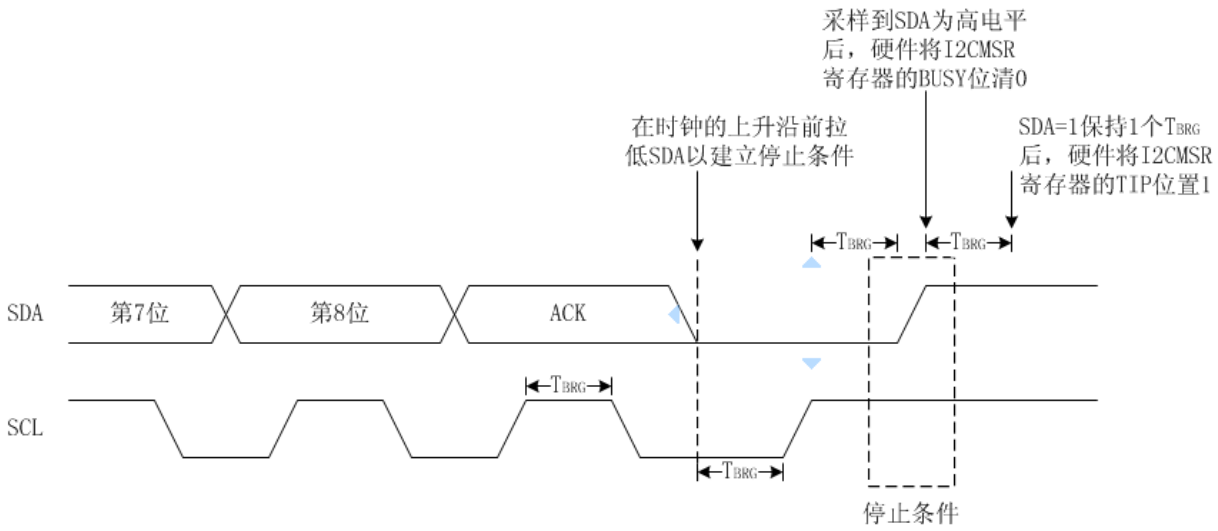
I2C Master 的时序

以下是启动、停止、应答、发送、接收的时序的时序图。

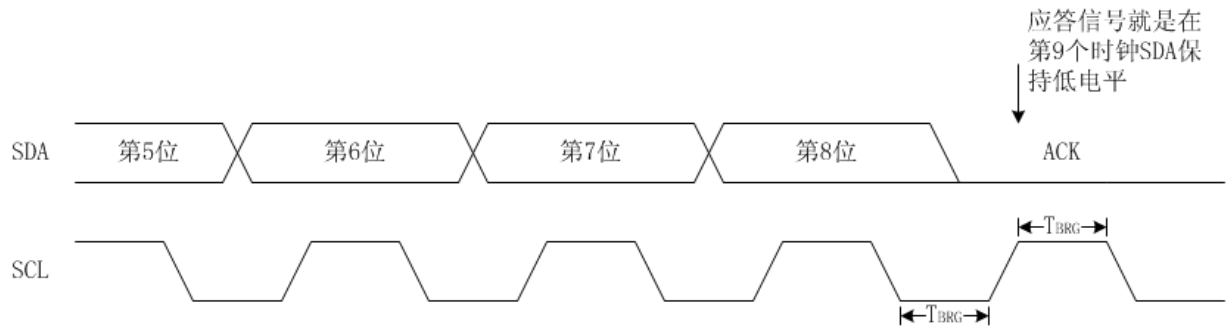
主控模式启动条件时序



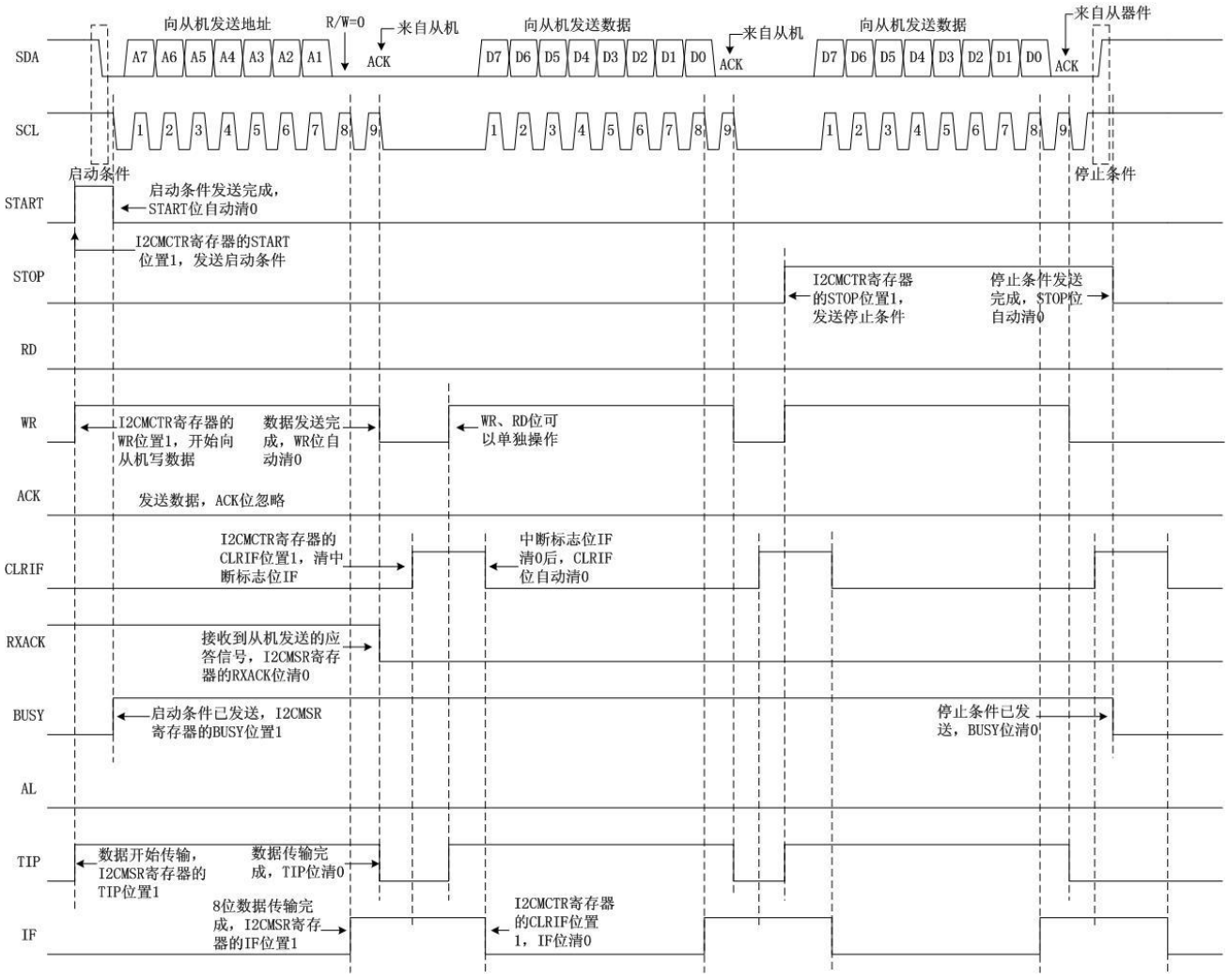
主控模式停止条件时序



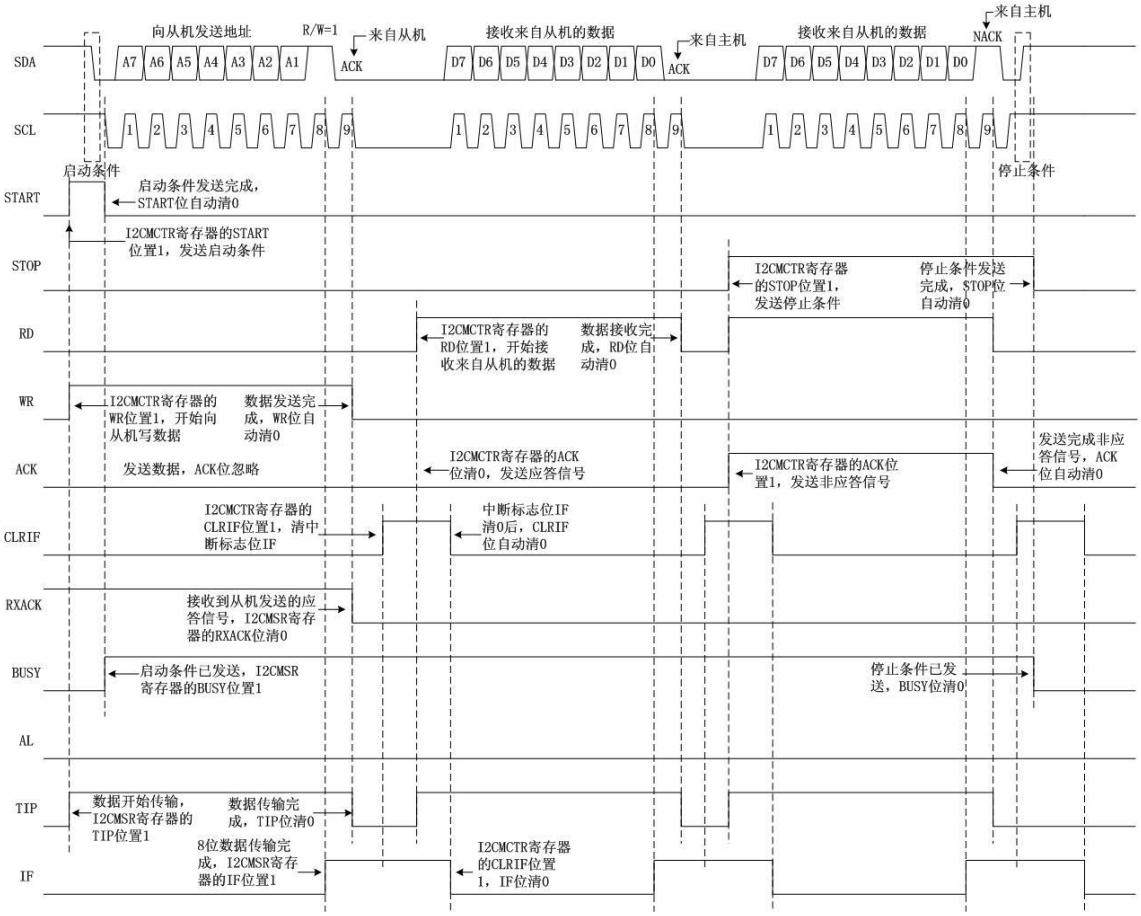
主控模式应答时序图



主控模式发送数据时序图



主控模式接收数据时序图



I2C Master 寄存器的定义

表 19-2、I2C Master 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
I2CMPRERL	0xC8FF	XRAM	I2C Master 波特率配置寄存器低 8 位	1111 1111
I2CMPRERH	0xC9FF	XRAM	I2C Master 波特率配置寄存器高 8 位	1111 1111
I2CMCR	0xCAFF	XRAM	I2C Master 控制寄存器	0000 0000
I2CMDR	0xCBFF	XRAM	I2C Master 数据寄存器	0000 0000
I2CMCTR	0xCCFF	XRAM	I2C Master 传输控制寄存器	0000 0000
I2CMSR	0xCDFF	XRAM	I2C Master 状态寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

I2CMPRERL：I2C Master 波特率配置寄存器低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 I2CMPRERL[7:0]：I^2C Master 波特率配置寄存器低 8 位

与 I2CMPRERH 寄存器组成 16 位 I^2C Master 波特率配置寄存器

I2CMPRERH：I2C Master 波特率配置寄存器高 8 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 I2CMPRERH[7:0]：I²C Master 波特率配置寄存器低 8 位

与 I2CMPRERL 寄存器组成 16 位 I²C Master 波特率配置寄存器

注：I²C Master 的波特率 = SYSCLK / （（SCLK1DIV+1） *（I2CMPRERH:I2CMPRERL + 1）* 5）；（SYSCLK 为系统所选择的时钟源）。

I2CMCR：I2C Master 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
I2CMEN	I2CMIE	-	-	-	-	-	-
R/W-0	R/W-0	U-x	U-x	U-x	U-x	U-x	U-x
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 I2CMEN：I²C 主控模式使能位

0 = 禁止 I²C 主控模式

1 = 使能 I²C 主控模式

bit 6 I2CMIE：I²C Master 中断使能位

0 = 禁止 I²C Master 中断

1 = 使能 I²C Master 中断

bit 5-0 未实现:读为 0

I2CMDR：I2C Master 数据寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 I2CMDR[7:0]：I²C Master 数据寄存器

I²C Master 数据寄存器，保存待发送数据以及接收到的数据

注：1、发送数据时，保存待发送的 8 位数据；

2、发送从机设备地址时，保存 7 位设备地址+1 位 R/W 位，其中 bit7-1 是设备地址，bit0 是 R/W 位；

3、接收数据时，保存接收到的 8 位数据。

I2CMCTR：I2C Master 传输控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
START	STOP	RD	WR	-	-	ACK	CLRIF
W-0	W-0	W-0	W-0	U-x	U-x	W-0	W-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 START：启动条件控制位

0 = 不发送启动条件

1 = 发送启动条件

bit 6 STOP：停止条件控制位

0 = 不发送停止条件

1 = 发送停止条件

bit 5 RD：从从机读数据控制位

0 = 不启动读数据

1 = 启动读，开始从从机读数据

bit 4 WR：向从机写数据控制位

0 = 不启动写数据

1 = 启动写，开始向从机写数据

bit 3-2 保留未用

bit 1 ACK：主机接收到数据时，应答条件控制位

0 =发送应答条件

1 = 不发送应答条件

bit 0 CLRIF：清除中断标志

写 0 无效；写 1 清除 I²C Master 中断标志位

注：

1、这些位写 1 后，会自动清零；

2、START、STOP、ACK 位不能单独使用，必须配合 WR、RD 位使用。但是 RD、WR 位可以单独使用，具体操作请参考以上“例程”部分描述。

I2CMSR：I2C Master 状态寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
RXACK	BUSY	AL	-	-	-	TIP	IF
R-0	R-0	R-0	U-x	U-x	U-x	R-0	R-0
注： R=可读位； W=可写位； U=保留位（读为 0）； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7 RXACK：I²C Master 接收到应答条件标志位

0 = 接收到应答条件

1 = 未接收到应答条件

bit 6 BUSY：I²C Master 总线忙标志位

0 = 发送了 STOP 信号探测到总线忙，STOP 命令失败

1 = 发送了 START 信号探测到总线忙，START 命令失败

bit 5 AL：I²C Master 总线仲裁失败标志位

0 = 总线仲裁成功

1 = 总线仲裁失败

bit 4-2 保留未用

bit 1 TIP：I²C Master 传输状态标志位

0 = 数据传输完成

1 = 数据传输中

bit 0 IF：I²C Master 中断标志位

0 = 数据正在传输或没有传输

1 = 8 位数据传输完成或仲裁失败

20、EFC 模块(类 EEPROM)

CBM73xxF 系列 MCU 的 Flash 存储器包括 Main 区和 NVR 区两部分。Main 区为芯片的程序存储器(ROM)，大小为 24K~64K；NVR 区可供用户程序访问，用于存储数据（类 EEPROM），支持掉电保存功能，大小为 1K。

Main 区和 NVR 区分为若干 Sector 区，每个 Sector 区为 512Byte，各区对应地址如下表 20-1：

表 20-1、flash sector 区地址详表

Main 区	NVR 区
Sector 0(0x0000~0x01FF)	NVR0(0xF800~0xF9FF)
Sector 1(0x0200~0x03FF)	NVR1(0xFA00~0xFBFF)
……	**NVR 区禁止越界访问**
Sector 126(0xFC00~0xFDFF)	
Sector 127(0xFE00~0xFFFF)	

Flash 存储器在正常工作条件下(整个 VDD 范围内)是可读写的。这些存储器并不直接映射到寄存器文件空间，而是通过特殊功能寄存器对其进行间接寻址。共有 8 个特殊功能寄存器用于访问这些存储器，如下：

- FDR
- FADRL
- FADRH
- FCR
- FPSR
- FDIV
- FCRCER
- FCRCSR

注：操作 EFC 模块前，必须通过配置 FDIV 时钟分频器，将 EFC 模块的时钟配置为 1MHZ，即 EFC_{CLK} = SYSCLK / FDIV（时钟源为 SYSCLK 详情见“系统总时钟框图”）。

20.1、NVR 区的操作(类 EEPROM)

NVR 区的数据具有掉电保存的功能，同时 NVR 区支持对 sector 擦除、字节编程、字节读取操作。

sector 擦除

要写 NVR 区存储器之前，必须要先进行擦除操作。EFC 模块只支持 sector 区擦除（不支持字节擦除），即每次擦除 sector 内任意地址都会将 1 个 sector 的整个区域擦除。

对于 sector 擦除操作，FADRL 和 FADRH 寄存器组成一个 16 位 Flash 存储器地址。FCR 寄存器的 NVRSEL 位必须为 1，ERASE 位用于控制启动 sector 擦除。只要设置 Flash 存储器地址落在目标 sector 内，将 ERASE 位置 1，即启动 sector 擦除，擦除完成后 ERASE 位自动清 0，所以可查询 ERASE 位的值，来判断擦除是否完成。注意，在操作 EFC 模块前，必须先按顺序写 FPSR 等于 0x55、0xAA。具体步骤可参考例 20-1：

例 20-1：NVR sector 擦除

```
void eraseSector(uint8 sel,uint16 adr)
{
    //FDIV = 0x30;//配置 EFC_CLK=SYSCLK/48=1M,SYSCLK 默认为内部 48M，FDIV 只需在配置系统时钟源时初始化一次即可
    FPSR = 0x55;
    FPSR = 0xAA;
    FCR_NVRSEL =  sel << 2 ;//1:NVR 区；0：main 区
    FADRL = adr & 0x00ff;   //adr 是编程地址
    FADRH = (adr & 0xff00) >> 8;
    FCR = 0x06;//启动擦出 NVR 区
    while(0x02 == FCR_ERASE){;}//判断是否完成
}
```

字节编程

EFC 模块支持对 NVR 区存储器字节编程，用户应该先将编程字节的地址写入 FADRL 和 FADRH 寄存器，编程数据写入 FDR 寄存器，FCR 寄存器的 NVRSEL 位必须为 1，PROG 位用于控制启动字节编程。将 PROG 位置 1，即启动字节编程，编程完成后 PROG 位自动清 0，所以可查询 ERASE 位是否为 0，来判断擦除是否完成。注意，在操作 EFC 模块前，必须先按顺序写 FPSR 等于 0x55、0xAA。具体步骤可参考例 20-2。

例 20-2：NVR 区字节编程

```
void writeByte(uint8 sel,uint16 adr,uint8 dat)
{
    //FDIV = 0x30; //配置 EFCCLK=SYSCLK/48=1M,SYSCLK 默认为内部 48M，FDIV 只需在配置系统时钟源时初始化一次即可
    FPSR = 0x55;
    FPSR = 0xAA;
    FCR_NVRSEL =  sel << 2 ;//1:NVR 区；0：main 区
    FDR = dat;//dat 是编程数据
    FADRL = adr & 0x00ff;//adr 是编程地址
    FADRH = (adr & 0xff00) >> 8;
    FCR = 0x05;//启动对 NVR 区字节编程
     while(0x01 == FCR_PORG){;}//判断是否完成
}
```

字节读

读 NVR 区存储器是通过 movc 指令实现的，对于字节读取操作，具体步骤可参考例 20-3。

例 20-3：NVR 区字节读取

```
unsigned char readByte(uint8 sel,uint16 adr)
{
    unsigned char dat;
    //FDIV = 0x30; //配置 EFCCLK=SYSCLK/48=1M,SYSCLK 默认为内部 48M，FDIV 只需在配置系统时钟源时初始化一次即可
    FCR_NVRREADSEL = sel<<3 ;//1:NVR 区；0：main 区
    dat = CBYTE[adr];
    return dat;
}
```

20.2、MAIN 区 CRC16-CCITT 校验

CBM73xxF 系列 MCU 的 MAIN 区支持 CRC16-CCITT 校验，支持：上电校验、动态校验。但 CRC 校验过程中 MCU 内核是停止工作的，不响应中断事件的发生，但会滞后一定时间（当 MCLK=16M 时，对 8 块 main 区进行 CRC 校验的总时间约等于 1.5ms）。

CRC 校验功能把 MCU 的 MAIN flash 分成 8 大块，每块大小为 8 Kbyte（第 4 块可能为 8k 或 7.5kbytes），这样可以方便单独对某些块进行 CRC 校验操作，各块对应在 MAIN 区的地址如下表 20-2（未使能 5.9 章节的在线更新功能时） 或 表 20-3（使能 5.9 章节的在线更新功能时），芯片出厂默认为不使能在线更新功能。

*表 20-2、CRC 校验块地址详表（**未使能** 5.9 章节的在线更新功能时）*

CRC 校验块	MAIN 区 flash 地址	空间大小
第 1 块 main flash	0x0000~0x1FFF	8 Kbytes
第 2 块 main flash	0x2000~0x3FFF	8 Kbytes
第 3 块 main flash	0x4000~0x5FFF	8 Kbytes
第 4 块 main flash	0x6000~0x7FFF	8 Kbytes
第 5 块 main flash	0x8000~0x9FFF	8 Kbytes
第 6 块 main flash	0xA000~0xBFFF	8 Kbytes
第 7 块 main flash	0xC000~0xDFFF	8 Kbytes
第 8 块 main flash	0xE000~0xFFFF	8 Kbytes

*表 20-3、CRC 校验块地址详表（**使能** 5.9 章节的在线更新功能时）*

CRC 校验块	MAIN 区 flash 地址	空间大小
第 1 块 main flash	0x0000~0x1FFF	8 Kbytes
第 2 块 main flash	0x2000~0x3FFF	8 Kbytes
第 3 块 main flash	0x4000~0x5FFF	8 Kbytes
第 4 块 main flash	0x6000~0x7DFF	7.5 Kbytes

20.2.1、上电校验

是指在 MCU 上电的过程中，MCU 可以开启对 MAIN 区 flash 程序存储块的 CRC 自检功能，如果自检结果正确则上电成功，然后开始执行用户程序；如果自检失败，则可以选择停止 MCU 的上电过程，用户程序也不会运行。

注：

1、通过配置上位机编程软件，可以选择是否使能上电校验，可以选择上电校验出错时是否停止 MCU 的上电；（详情参考推广资料里的《CBM7X 系列烧录器使用手册_V1.1.pdf》）

2、用户软件也可以在上电时通过查询寄存器 FCRCSR 对应位来获取 CRC 上电校验的结果，0 表示无误，1 表示出错；如果查询到上电校验结果出错，可以执行软件复位或其它保护动作。

20.2.2、动态校验

是指 MCU 上电成功后，用户程序可以在主程序中进行软件控制，定时对 MAIN 区 Flash 程序存储块进行 CRC 自检。FCRCER 寄存器控制是否使能动态校验，FCRCSR 为动态校验结果状态寄存器。如果查询到校验结果出错，可以执行软件复位或其它保护动作。参考例程如下 20-4：

20-4：CRC 动态校验例程：

```
unsigned crc_Check(uint16secten)//当使能 remap 功能, secten 的 bit[7:4]必须配置为 0
{
    uint16 crcErrflg ;
    FCRCER = secten ;//使能对应扇区动态 CRC
    MISCR1_NVR2CRCEN=((secten&0x100)>>7) ;//使能 NVR2 区动态 crc
    FPSR = 0x55 ;
    FPSR = 0xaa ;
    FCR_CRCSEQ = 0x50 ;
    FCR_CRCSEQ = 0xa0 ;
    FCR_CRCSEQ = 0x10 ;
    crcErrflg = MISCR3_NVR2CRCSR&0x02 ;
    crcErrflg <<= 7 ;
    crcErrflg |= FCRCSR ;
    return crcErrflg ;//返回 CRC 校验结果
}
```

20.2.3、CRC16 校验公式及校验代码

CBM73xxF 系列 MCU 的 crc 校验算法使用标准的 CRC16-CCITT 进行上电校验和动态校验，其生成多项式为 x16+x12+x5+1，简记式为 1021，初始值采用 0xFFFF。

校验代码如下：

```
unsigned int CRC16_Bin(unsigned char *message, unsigned int len)
{
    unsigned int i, j;
    unsigned int crc_reg =0xffff;//初值
    for(i=0;i<len;i++)
    {
        unsigned int tmp=message[i];
        crc_reg^= (unsigned short)tmp<<8 ;
        for(j=0;j<8;j++)
        {
            if( (crc_reg&0x8000)==0x8000)
            {
                crc_reg=(crc_reg<<1)^0x1021;
            }
            else
            {
                crc_reg<<=1;
            }
        }
    }
    crc_reg^=0xffff;
    return crc_reg;
}
```

20.2.4、CRC16 校验的长度及数据内容

分两种情形：

1>、未使能 5.9 章节的在线更新功能时

此时 main flash 分为 8 块 CRC（参考上表 20-2），每块的固定大小为 8kbytes。假设.bin 烧录文件的实际数据长度为 len(len 为 16bit 变量)，则需要进行 **CRC 校验的有效块数为 n=（（（len-1）/8k）+1）**，且.bin 烧录文件的最后一块需要补数据到 8k 进行校验，补的内容分为两部分：一是.bin 烧录文件实际长度页的最后一页补 **k=(256-len%256)个数据 0x00**(k 为 8bit 变量)；二是之后的页，补 **l=(n*8k-len-k)个数据 0xFF** 直到 8k 大小。（每页为 256byte，1k=1024byte）

例如 len=10655bytes,则 n=(((len-1)/8k)+1)=((10654/8k)+1)=2 块 crc，k=(256-len%256)=(256-10655%256)=97 个数据 0x00；因为 n=2 符合<=3 的情况则 l=(n*8k-len-k)=(2*8*1024-10655-97)=5632 个数据 0xFF。

2>、使能 5.9 章节的在线更新功能时

此时 main flash 分为 4 块 CRC（参考上表 20-3），其中第 1~3 块固定大小为 8kbytes,第 4 块固定大小为 7.5k。假设.bin 烧录文件的实际数据长度为 len(len 为 16bit 变量)，则需要进行 **CRC 校验的有效块数为 n=（（（len-1）/8k）+1）**，且.bin 烧录文件的最后一块需要补数据到 8k 或 7.5k 进行校验，补的内容分为两部分：一是.bin 烧录文件实际长度页的最后一页补 **k=(256-len%256)个数据 0x00**(k 为 8bit 变量)； 二是，如果 **n<=3 时**则之后的页，补 **l=(n*8k-len-k)个数据 0xFF** 直到 8k 大小，如果 **n=4 时**则之后的页，补 **l=（(3*8k+7.5k)-len-k)个数据 0xFF** 直到 8k 大小。（每页为 256bytes，1k=1024bytes）

例如 len=10655bytes,则 n=(((len-1)/8k)+1)=((10654/8k)+1)=2 块 crc,k=(256-len%256)=(256-10655%256)=97 个 0x00；因为 n=2 符合<=3 的情况则 l=(n*8k-len-k)=(2*8*1024-10655-97)=5632 个 0xFF。

例如 len=28569bytes,则 n=(((len-1)/8k)+1)=((28568/8k)+1)=4 块 crc，k=(256-len%256)=(256-28569%256)=103 个 0x00；因为 n=4 则 l=((3*8k+7.5k)-len-k)=((3*8*1024+7.5*1024)-28569-103)=3584 个 0xFF。

20.3、EFC 寄存器的定义

表 20-3、EFC 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
FDR	0xB0FF	XRAM	EFC 编程数据寄存器	0000 0000
FADRL	0xB1FF	XRAM	EFC 编程地址寄存器低8位	0000 0000
FADRH	0xB2FF	XRAM	EFC 编程地址寄存器高8位	0000 0000
FCR	0xB3FF	XRAM	EFC 控制寄存器	0000 0000
FPSR	0xB4FF	XRAM	EFC 写保护序列寄存器	0000 0000
FDIVR	0xB5FF	XRAM	EFC 时钟分频寄存器	0011 0000
FCRCER	0xB6FF	XRAM	CRC 动态使能寄存器	0000 0000
FCRCSR	0xB7FF	XRAM	CRC 校验结果寄存器	0000 0000

以上表中所有寄存器均支持按位操作或直接对字节操作。

FDR：EFC 编程数据寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 FDR[7:0]：EFC 编程数据寄存器，用于配置编程数据

FADRL：EFC 编程地址寄存器低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 FADRL[7:0]：EFC 编程地址寄存器低 8 位，与 FADRH 组成 16 位编程地址

FADRH：EFC 编程地址寄存器高 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 FADRH[7:0]：EFC 编程地址寄存器高 8 位，与 FADRL 组成 16 位编程地址

注：对于 Sector 擦除操作，只需要设置编程地址落在目标 sector 内即可。

FCR：EFC 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CRCSEQ3	CRCSEQ2	CRCSEQ1	CRCSEQ0	NVRREADSEL	NVRSEL	ERASE	PORG
R/W-0	R/W-0	R/W-0	R/W-0	U-x	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-4　　CRCSEQ0[3:0]：CRC 校验启动命令时序寄存器位

启动 CRC 校验，需要按照顺序配置高 4bit 为 0x5,0xa,0x1 才能启动 CRC 校验，当 CRC_EN 配置全为 0 时，无法启动 CRC 校验；

bit 3　　NVRREADSEL：读数据区域选择位

0 = 选择 Main 区

1 = 选择 NVR 区

bit 2　　NVRSEL：编程区域选择位

0 = 选择 Main 区

1 = 选择 NVR 区

bit 1　　ERASE：编程区域选择位

当要启动 sector 擦除时，将此位置 1，擦除结束后会自动清 0

bit 0　　PORG：编程区域选择位

当要启动字节编程时，将此位置 1，编程结束后会自动清 0

注：写 FCR 寄存器之前，必须按顺序先写 FPSR 寄存器等于 0x55、0xAA。

FPSR：EFC 写保护序列寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　FPSR<7:0>：写保护序列寄存器。

在操作 FCR 寄存器之前，需将 FPSR 写成 0xAA；

将 FPSR 写成 0xAA 的步骤是：

1、当 FPSR 等于 0x00 时，可以将 FPSR 写成 0x55；

2、当 FPSR 等于 0x55 时，可以将 FPSR 写成 0xAA；

3、当编程结束或擦除结束时，FPSR 被强制置为 0x00；

FDIVR：EFC 时钟分频寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　FDIVR<7:0>：EFC EFC 时钟分频寄存器，必须通过配置此寄存器给 EFC 模块提供 1MHZ 的时钟，

CLK_{EFC}= SYSCLK/FDIVR，SYSCLK 为系统时钟源默认为内部 48M，时钟详情请参考“4.1、系统总时钟框图”。

FCRCER：CRC 动态校验使能寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CRC7EN	CRC6EN	CRC5EN	CRC4EN	CRC3EN	CRC2EN	CRC1EN	CRC0EN
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　CRCnEN：第 n 块 MAIN 区 flash CRC 动态校验使能位（0<= n <= 7），当使能 remap 功能,高 4bit 必须配置为 0。

0 = 不使能

0 = 使能第 n 块 MAIN 区 flash 的 crc 动态校验

对应第 n 块 MAIN 区 flash 地址请参考表“*表 23-2、CRC 校验块地址详表*”：

FCRCSR：CRC 校验结果寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CRC7SR	CRC6SR	CRC5SR	CRC4SR	CRC3SR	CRC2SR	CRC1SR	CRC0SR
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0　　CRCnSR[7-0]：CRC 校验结果状态位（0<= n <= 7）。

0 = CRC 校验结果正确

1 = CRC 校验结果错误

21、LED 驱动器

CBM73xxF 系列 MCU 支持硬件 LED 驱动扫描单元，仅需要很少的软件操作即可实现 LED 显示控制。支持多种 LED 数码管规格，配合 IO 电流驱动能力控制寄存器(PnLDRVRH:PnLDRVRL)及本模块中的灰度等级选择位(LEDSETR 寄存器的 GRAY[2:0]位)，可实现多级灰度显示。

21.1、LED 的操作步骤

配置 LED 的操作步骤如下：

1、端口配置

1)、配置 PxyIOCFG 寄存器将 IO 口复用为 LED GRID(COM)/SEG 口功能；引脚方向会自动为输出方向(无需软件再配置 PnOE)。

2)、PnLDRVRH:PnLDRVRL 寄存器用于配置 IO 的驱动电流大小，与 LEDSETR 寄存器的 GRAY[2:0]位配合，可以满足不同的显示需求。

注：端口配置前应使能 IOC 模块时钟，即将 MDLCKCR 寄存器的 IOCEN 位置 1；另外不能将 IO 口电流寄存器 PnLDRVRH/PnLDRVRL 的对应控制位配置成 00，否则可能出错。

2、LED 模块时钟使能

在操作 LED 模块相关寄存器前，必须先使能 LED 模块时钟，即将 MDLCKCR 寄存器的 LEDEN 位置 1。

3、LED 模块参数配置

3.1、数码管的类型选择：

LED 驱动器可以驱动不同类型的 LED，LEDSETR 寄存器的 TYPE 位可以选择所驱动的数码管是共阴极还是共阳极；MODE[2:0]位可以配置段位数，有 6 种类型可以选择：

- 4 位 13 段（IO 复用为 GRIDm 的引脚接数码管的位选引脚，复用为 SEGn 的引脚接数码管段引脚）
- 5 位 12 段（IO 复用为 GRIDm 的引脚接数码管的位选引脚，复用为 SEGn 的引脚接数码管段引脚）
- 6 位 11 段（IO 复用为 GRIDm 的引脚接数码管的位选引脚，复用为 SEGn 的引脚接数码管段引脚）
- 7 位 10 段（IO 复用为 GRIDm 的引脚接数码管的位选引脚，复用为 SEGn 的引脚接数码管段引脚）
- 9 位 8 段（IO 复用为 GRIDm 的引脚接数码管的段引脚，复用为 SEGn 的引脚接数码管位选引脚）
- 10 位 7 段（IO 复用为 GRIDm 的引脚接数码管的段引脚，复用为 SEGn 的引脚接数码管位选引脚）

3.2、显示的灰度等级设置：

LEDSETR 寄存器的 GRAY[2:0]位可以配置 LED 显示的灰度等级，最多支持 8 级；值越大，显示越亮。(与 IO 的驱动电流配置寄存器配合后可以满足不同的显示需求)

3.3、刷新率配置：

LED 模块的时钟是内部 OSC800K 时钟经过 CKDIVCR 寄存器的 ILCLKDIV[1:0]位分频后的时钟。详情请见“4.1 系统总时钟框图”描述；LED 显示的刷新率由 LEDDIVR 寄存器控制，LEDDIVR 寄存器可以配置 LED 时钟的分频数，分频数越大，LED 时钟越慢，LED 刷新率越低(闪烁明显)，反之，刷新率越高(闪烁不明显)。

计算刷新频率：刷新频率是指 1s 时间内，某位某段点亮的次数。

设显示模式为 M 位 N 段，但实际接的数码管为 Y(1<=Y<=M)位。LED 时钟分频数为 DIV1(LEDDIVR 寄存器配置)；LED 时钟来自于内部 OSC800K 经过 ILCLKDIV 分频后的时钟，设 ILCLKDIV 分频数为 DIV2(ILCLKDIV[1:0]配置)。

则 Y 位数码管依次被点亮的总时间为：

$T = Tosc800k \times (DIV1 + 1) \times 2^{DIV2} \times 8 \times N \times Y$;

因为控制方式为逐位扫描，所以，在 T 时间内，某位某段只被点亮了一次。那么，刷新率计算公式为：

$F = 1/T = 1/((Tosc800k \times (DIV1 + 1) \times 2^{DIV2} \times 8 \times N \times Y)$

实例：M=7，N=10，Y=5，DIV1=3，DIV2=3（7 位 10 断模式实际只驱动 5 位数码管情况），则刷新率：

$F = 1s / (1.25us \times 4 \times 2^3 \times 8 \times 10 \times 5) = 62.5Hz$。

实践表明：刷新频率为 60Hz 时闪烁明显，达到 70Hz 时可基本消除闪烁。

3.4、数码管的有效位选择：

GRIDVLDL 和 GRIDVLDH 寄存器组成 10bit 的 GRIDVLD，用于配置 LED 数码管的有效位数。LED 模块支持最多 10 位数码管，实际使用中，若不足 10 位，可配置 GRIDVLD，将使用到的位置 1，未使用的位清 0（应用中根据实际使用数码管的位数来配置，可提高显示刷新率）。

举例：若实际使用 6 位数码管，使用的是第 1，2，3，5，6，10 位，则可将 GRIDVLD 设置成 0x0237 (0010_0011_0111)。

4、显示数据模式配置：

根据 LEDSETR 寄存器 MODE[2:0]位的配置，LEDDRn 寄存器的用法有所差异。

1)、当配置成 4 位 13 段、5 位 12 段、6 位 11 段、7 位 10 段时：

LEDR1 和 LEDR0 组成 16 位数据，取低 n 位(n=10~13)作为数码管第 1 位的数据；LEDR3 和 LEDR2 组成 16 位数据，取低 n 位(n=10~13)作为数码管第 2 位的数据；以此类推。

2)、当配置成 9 位 8 段时：

LEDRn(n=0~8)作为 9 位数码管的数据。

3)、当配置成 10 位 7 段时：

LEDRn(n=0~9)作为 10 位数码管的数据，LEDRn 取低 7 位。

根据实际的硬件接法，给数码管配置合适的数据。数据的 1 个 bit 对应于 1 位数码管的 1 个段；不管共阴还是共阳的数码管，bit 位写 1 亮，写 0 灭。

5、显示使能：

LEDCR 寄存器的 ONOFF 位置 1，即使能 LED 显示，清 0，即关闭 LED 显示。需注意的是 ONOFF 位置 1 后，等到 WREN 位被清 0 时，才表示显示真正使能；ONOFF 位清 0，等到 WREN 位被置 1 时，才表示显示已经关闭。

注：当需要更改显示数据时，必须先关闭 LED 显示，查询 LEDCR 寄存器的 WREN 位为 1 后，才能配置新的显示数据，然后再使能 LED 显示。

21.2、数码管的接法

根据数码管不同的段位配置，数码管有两种接法：

1）当配置成 4 位 13 段、5 位 12 段、6 位 11 段、7 位 10 段时，IO 复用为 GRIDm 的引脚接数码管的位选引脚，复用为 SEGn 的引脚接数码管段引脚。

2）当配置成 9 位 8 段、10 位 7 段时，IO 复用为 GRIDm 的引脚接数码管的段引脚，复用为 SEGn 的引脚接数码管位选引脚。

21.3、LED 显示控制原理

共阳 LED 驱动

以 3 位 8 段共阳极数码管为例，LED 共阳显示控制时序如图 26-1 所示。

LED 模块使用的是逐位扫描的方式驱动数码管：

1）共阳极的数码管，所以 3 个 GRID 位逐个变高；

2）8 段的数码管，所以每个 GRID 位为高的时间持续 8 个时间单位(若为 7 段则持续 7 个时间单位)。

3）8 个 SEG 位分别对应于所配置数据位。在这 8 个时间单位，8 个 SEG 位变化为数据位所对应的电平，每个 SEG 位的电平持续时间为 8 个时间单位。即，在同一时刻，最多有 8 个段被点亮。

4）一个时间单位等于 8 个 LED 时钟周期，LED 时钟周期由 CKDIVCR、LEDDIVR 寄存器的配置决定。所以 LED 时钟周期改变时，刷新率会变化；当有效位数增多时，刷新率也会随之改变。

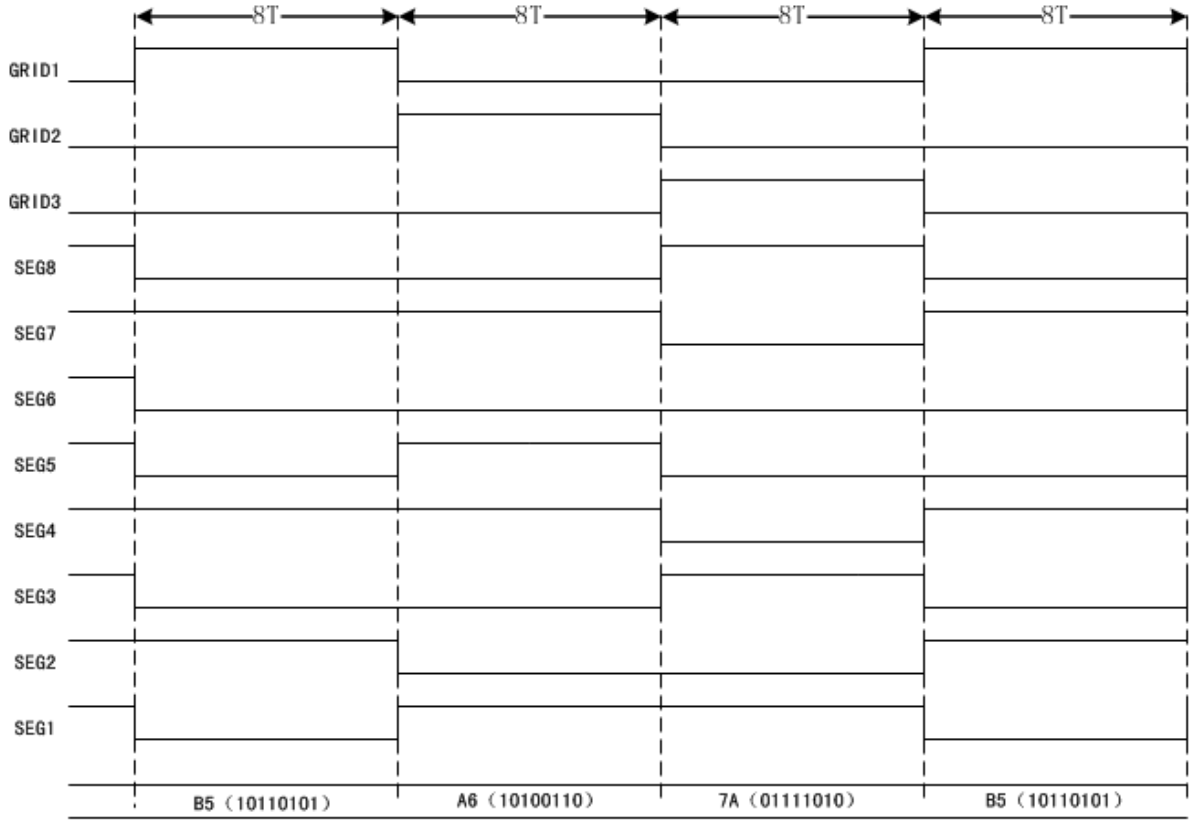


图 21-1、LED 显示控制时序图(共阳极)

注：图 21-1 中因为是共阳极，GRID 为高，SEG=0 才点亮，所以当数据位为 1 时，对应 SEG 为 0。

共阴 LED 驱动

共阴极数码管的时序则完全相反，如图 26-2 所示。

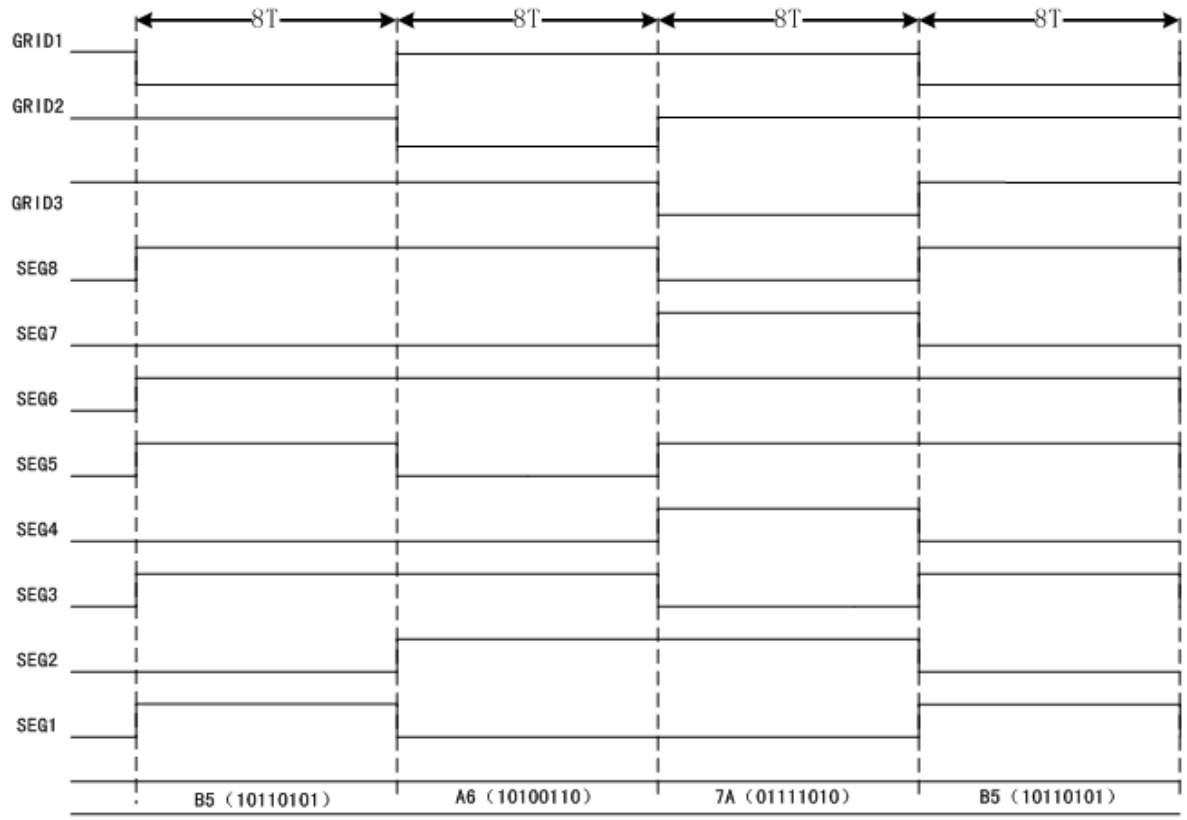


图 21-2、LED 显示控制时序图(共阴极)

注：图 21-2 中因为是共阴极，GRID 为低，SEG=1 才点亮，所以当数据位为 1 时，对应 SEG 为 1。

21.4、LED 寄存器的定义

表 21-1、LED 寄存器列表

寄存器名称	寄存器地址	地址类型	说明	POR 复位值
LEDCR	0xC0FF	XRAM	LED 控制寄存器	0000 0010
LEDSETR	0xC1FF	XRAM	LED 参数配置寄存器	0000 0111
LEDDIVR	0xC2FF	XRAM	LED 时钟分频寄存器	0000 0010
GRIDVLDL	0xC3FF	XRAM	LED 有效位配置寄存器低8位	0000 0000
GRIDVLDH	0xC4FF	XRAM	LED 有效位配置寄存器高2位	0000 0000
LEDDR0	0x1100	XRAM	LED 数据寄存器0	0000 0000
LEDDR1	0x1101	XRAM	LED 数据寄存器1	0000 0000
LEDDR2	0x1102	XRAM	LED 数据寄存器2	0000 0000
LEDDR3	0x1103	XRAM	LED 数据寄存器3	0000 0000
LEDDR4	0x1104	XRAM	LED 数据寄存器4	0000 0000
LEDDR5	0x1105	XRAM	LED 数据寄存器5	0000 0000
LEDDR6	0x1106	XRAM	LED 数据寄存器6	0000 0000
LEDDR7	0x1107	XRAM	LED 数据寄存器7	0000 0000
LEDDR8	0x1108	XRAM	LED 数据寄存器8	0000 0000
LEDDR9	0x1109	XRAM	LED 数据寄存器9	0000 0000
LEDDR10	0x110A	XRAM	LED 数据寄存器10	0000 0000
LEDDR11	0x110B	XRAM	LED 数据寄存器11	0000 0000
LEDDR12	0x110C	XRAM	LED 数据寄存器12	0000 0000
LEDDR13	0x110D	XRAM	LED 数据寄存器13	0000 0000

以上表中寄存器 LEDDRn(0~13)均不支持按位操作，只能对整个寄存器进行字节的操作；其它寄存器支持按位操作或直接对字节操作。

LEDCR：LED 控制寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-	WREN	ONOFF
U-x	U-x	U-x	U-x	U-x	U-x	R-1	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-2　　保留未用

bit 1　　WREN：可写 LED 数据寄存器标志位

只读寄存器。改变 LED 显示信息前，需先写 ONOFF 使 LED 显示灭，然后查询等待 WREN 为 1 时(当使能 LED 显示时，WREN 会变低)，才可以更改 LED 显示信息；

bit 0　　ONOFF：LED 显示使能位

0 = 关闭 LED 显示

1 = 使能 LED 显示

LEDSETR：LED 参数配置寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
TYPE	MODE2	MODE1	MODE0	-	GRAY2	GRAY1	GRAY0
R/W-0	R/W-0	R/W-0	R/W-0	U-x	R/W-1	R/W-1	R/W-1

注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）

bit 7 TYPE：LED 类型选择位

0 = 共阴极

1 = 共阳极

bit 6-4 MODE[2:0]：LED 模式选择位

000 = 4 位 13 段；SEG 脚作为段，GRID 脚作为位

001 = 5 位 12 段；SEG 脚作为段，GRID 脚作为位

010 = 6 位 11 段；SEG 脚作为段，GRID 脚作为位

011 = 7 位 10 段；SEG 脚作为段，GRID 脚作为位

10x = 9 位 8 段；GRID 脚作为段，SEG 脚作为位

11x = 10 位 7 段；GRID 脚作为段，SEG 脚作为位

bit 3 保留未用

bit 2-0 GRAY[2:0]：LED 灰度等级选择位

000 = 时间单位的 12.5%

001 = 时间单位的 25%

010 = 时间单位的 37.5%

011 = 时间单位的 50%

100 = 时间单位的 62.5%

101 = 时间单位的 75%

110 = 时间单位的 87.5%

111 = 时间单位的 100

LEDDIVR：LED 时钟分频寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-			
U-x	U-x	U-x	U-x	U-x	R/W-0	R/W-1	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-3 保留未用

bit 2-0 LEDDIVR[2:0]：LED 时钟分频寄存器。LED 扫描时钟源来自 SCLK3，详细请参考“4.1 系统总时钟框图”

000 = SCLK3 1 分频

001 = SCLK3 2 分频

010 = SCLK3 3 分频

011 = SCLK3 4 分频

100 = SCLK3 5 分频

101 = SCLK3 6 分频

110 = SCLK3 7 分频

111 = SCLK3 8 分频

GRIDVLDL：LED 有效位配置寄存器低 8 位

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）

bit 7-0 GRIDVLDL[7:0]：LED 有效位配置寄存器低 8 位

GRIDVLDH 组成 10bit 的 LED 有效位配置寄存器 GRIDVLD

GRIDVLDH：LED 有效位配置寄存器高 2 位(续)

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
-	-	-	-	-	-		
U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-2 保留未用

bit 1-0 GRIDVLDH[1:0]：LED 有效位配置寄存器高 2 位

GRIDVLDL 组成 10bit 的 LED 有效位配置寄存器 GRIDVLD

LEDDRn：LED 数据寄存器

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
注： R=可读位； W=可写位； U=保留位； -x=上电复位默认值（1：置 1； 0：清零； x：未知值）							

bit 7-0 LEDDRn[7:0]：LED 数据寄存器

LEDDRn(n = 0、1、2、3、4、5、……、10、11、12、13)是 LED 数据寄存器，根据 LEDSETR 寄存器 MODE 位的配置，LEDDRn 的用法有所区别，详情请参考“LED 的操作步骤-显示数据模式配置”。

22、电气特性

极限参数

电源供应电压.....VSS-0.3V~VSS+6.0V
端口输入电压.....VSS-0.3V~VDD+0.3V
储存温度.....-55℃~125℃
工作温度.....-40℃~85℃
VSS 引脚的最大输出电流.....120mA
VDD 引脚的最大输入电流.....120mA
任一 I/O 引脚的最大输出灌电流.....85mA
任一 I/O 引脚的最大输出拉电流.....85mA
总功耗.....600mW

注：如果芯片的工作条件超过极限参数所规定的范围，就可能对芯片造成永久性的损坏。上述值仅为运行条件极大值，我们建议不要使芯片在规定的范围以外运行。芯片长时间工作在最大值条件下，其稳定性会受到影响。

直流电气特性（T = 25℃）

符号	参数描述	测试条件		最小	典型	最大	单位
		VDD	条件				
VDD	工作电压	-	低压复位不使能，POR 使能	2.7	-	5.5	V
			低压复位使能	2.9	-	5.5	V
IDD	工作电流（正常模式）	5V	无负载，Fsys = 8MHz，WDT 使能，Touch Key off，LED off，ADC 硬件模块关闭，ADCBUFEN 关闭	-	4.5	-	mA
		5V	无负载，Fsys = 16MHz，WDT 使能，Touch Key off，LED off，ADC 硬件模块关闭，ADCBUFEN 关闭	-	5	-	mA
	工作电流（空闲模式）	5V	无负载，Fsys = 8MHz，WDT 关闭，Touch Key off，LED off，ADC 硬件模块关闭，ADCBUFEN 关闭	-	2.8	-	mA
		5V	无负载，Fsys = 16MHz，WDT 关闭，Touch Key off，LED off，ADC 硬件模块关闭，ADCBUFEN 关闭	-	3	-	mA
	工作电流（休眠模式）	5V	无负载，Fsys 关闭，WDT 关闭，Touch Key off，LED off，ADC 硬件模块关闭，ADCBUFEN 关闭，IO 高阻内部上拉	-	160	-	uA
V_{IL}	I/O 口或输入引脚的低电平输入电压	5V	-	0	-	0.65	V
		-		0	-	0.15VDD	V
V_{IH}	I/O 口或输入引脚的高电平输入电压	5V	-	3.5	-	5.0V	V
		-		0.8VDD	-	VDD	V
V_{POR}	上电复位电压	-	POR 使能，V_{POR} = 2.3v	-5%	2.3	5%	V
V_{LVD}	低压复位电压	-	低压复位使能，VLVD = 2.9V	-5%	2.9	5%	V
			低压复位使能，VLVD = 3.3V	-5%	3.3	5%	V

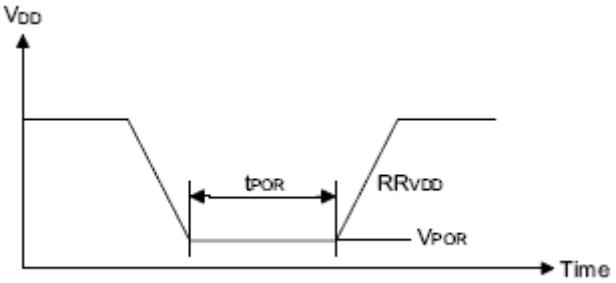
			低压复位使能，VLVD = 3.7V	-5%	3.7	5%	V
			低压复位使能，VLVD = 4.2V	-5%	4.2	5%	V
I_{OL}	灌电流	5V	V_{OL} = 0.1VDD，PnLDRVR_x = 00/11（6mA）	5	6	8	mA
			V_{OL} = 0.1VDD，PnLDRVR_x = 01（30mA）	25	30	40	mA
			V_{OL} = 0.1VDD，PnLDRVR_x = 10（60mA）	50	60	85	mA
I_{OH}	拉电流	5V	V_{OH}= 0.9VDD，PnLDRVR_x = 00/11（6mA）	5	6	8	mA
			V_{OH}= 0.9VDD，PnLDRVR_x = 01（30mA）	25	30	40	mA
			V_{OH}= 0.9VDD，PnLDRVR_x = 10（60mA）	50	60	85	mA
RPH	I/O 口上拉电阻	-	Pn_PURSEL_x = 00（88K）	-	88	-	kΩ
			Pn_PURSEL_x = 01（2.2K）	-	2.2	-	kΩ
			Pn_PURSEL_x = 10（4.5K）	-	4.5	-	kΩ
			Pn_PURSEL_x = 11（10K）	-	10	-	kΩ

交流电气特性（T = 25℃）

符号	参数描述	测试条件		最小	典型	最大	单位
		VDD	条件				
F_{IHCLK}	内部 48M 精度	3.3/5V	25℃	-1%	48M	1%	MHz
F_{IHCLKT}	内部 48M 温漂	3.3/5V	-40℃ ~ +85℃	-1%	48M	1%	MHz
T_{IHCLK}	内部 48M 启动稳定时间	3.3/5V	-40℃ ~ +85℃	-	-	0.05	ms
F_{ILCLK}	内部 800K 精度	3.3/5V	25℃	-1%	800	1%	KHz
F_{ILCLKT}	内部 800K 温漂	3.3/5V	-40℃ ~ +85℃	-1%	800	1%	KHz
T_{ILCLK}	内部 800K 启动稳定时间	3.3/5V	-40℃ ~ +85℃	-	-	0.05	ms
T_{ECLK_M}	外部 M 晶振启动稳定时间	3.3/5V	-40℃ ~ +85℃	0.05	0.1	-	s
T_{ECLK_K}	外部 K 晶振启动稳定时间	3.3/5V	-40℃ ~ +85℃	-	1	-	s
T_{EINT}	外部中断脉宽时间	-	-	16	-	-	MCLK
T_{POR}	上电复位时间	-	-	-	40	-	ms
T_{WKT}	系统唤醒时间（IDLE 模式）	-	-	-	-	2	T_{SYS}
	系统唤醒时间（SLEEP 模式）	-	-	-	-	5	T_{SYS}

上电复位特性（T = 25℃）

符号	参数	测试条件		最小	典型	最大	单位
		VDD	条件				
V_{POR}	上电复位电压	-	-40℃～ +85℃	-	-	2.3	V
RR_{VDD}	上电复位电压速率	-	-40℃～ +85℃	0.05	-	-	V/ms
T_{POR}	VDD 保持为 V_{POR} 的最小时间	-	-40℃～ +85℃	-	10	-	ms



触控按键电气特性（T = 25℃）

符号	参数	测试条件		最小	典型	最大	单位
		VDD	条件				
I_{KEY}	触摸按键工作电流	5v	使能触摸模块工作	-	3.3	-	mA
I_{REFKEY}	内部触摸按键工作电流	5v	内部 1 个 key	-	30	-	uA
C_{KEY}	触摸感应电容值	5v		5	30	60	pF
C_{REFKEY}	内建可调电容值	5v		5	-	60	pF

A/D 转换器电气特性（T = 25℃）

符号	参数	测试条件		最小	典型	最大	单位
		VDD	条件				
VDD	A/D 转换器工作电压	-	-	2.7	-	5.5	V
V_{ADI}	A/D 转换器输入电压	-	-	0	-	VDD	V
Vref1v	内部 1v 参考电压	-	-	-1%	1.00	1%	V
DNL	非线性微分误差	3.3V	-	-	-	±2	LSB
		5V					
		3.3V	-	-	-	±2	LSB
		5V					
INL	非线性积分误差	3.3V	-	-	-	±2	LSB
		5V					
		3.3V	-	-	-	±2	LSB
		5V					
I_{ADC}	A/D 硬件模块功耗	3.3V	无负载(SCLK=16M)	-	1	1.25	mA
		5V	无负载(SCLK=16M)	-	2	2.5	mA
I_{Vref1v}	使能(ADCBUFEN=1)内部 1v 参考电压增加的功耗	-	-	-	1.5	2.5	mA
T_{ADCK}	A/D 转化器时钟周期	-	-	-	-	1.4	Mhz
T_{ADC}	A/D 转换时间(包括采样和保持时间)	-	10 bit ADC	13	-	17	T_{ADCK}
T_{ADS}	A/D 转换器采样时间	-	-	2	-	6	T_{ADCK}
T_{ON2ST}	A/D 转换器 On-to-Start 时间	-	-	-	-	1	us
T_{Vref1v}	开启 Vref1v 稳定时间	-	-	-	-	500	us

flash 电气特性

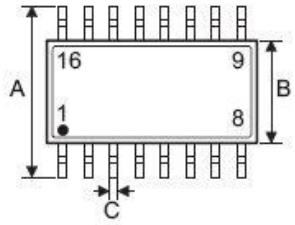
符号	参数	测试条件		最小	典型	最大	单位
		VDD	条件				
VDD	擦写电压	-	-40℃～ +85℃	2.7	5.0	5.5	v
Tcycle	擦写周期	5	-40℃～ +85℃	100,000	-	-	次
Tsave	数据保存时间	5	25℃	20	-	-	年
T_{sector}	页（512Bytes）擦除时间	5	-40℃～ +85℃	4	5	6	ms
T_{chip}	全片（64K）擦除时间	5	-40℃～ +85℃	45	53	60	ms

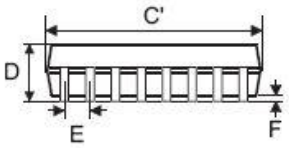
类 EEPROM 电气特性

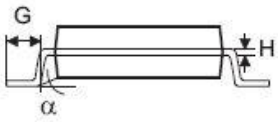
符号	参数	测试条件		最小	典型	最大	单位
		VDD	条件				
VDD	擦写电压	-	-40℃～ +85℃	2.7	5.0	5.5	v
Tcycle	擦写周期	5	-40℃～ +85℃	100,000	-	-	次
Tsave	数据保存时间	5	25℃	20	-	-	年
T_{sector}	页（512Bytes）擦除时间	5	-40℃～ +85℃	4	5	6	ms

23、封装信息

(1)SOP16 外形尺寸

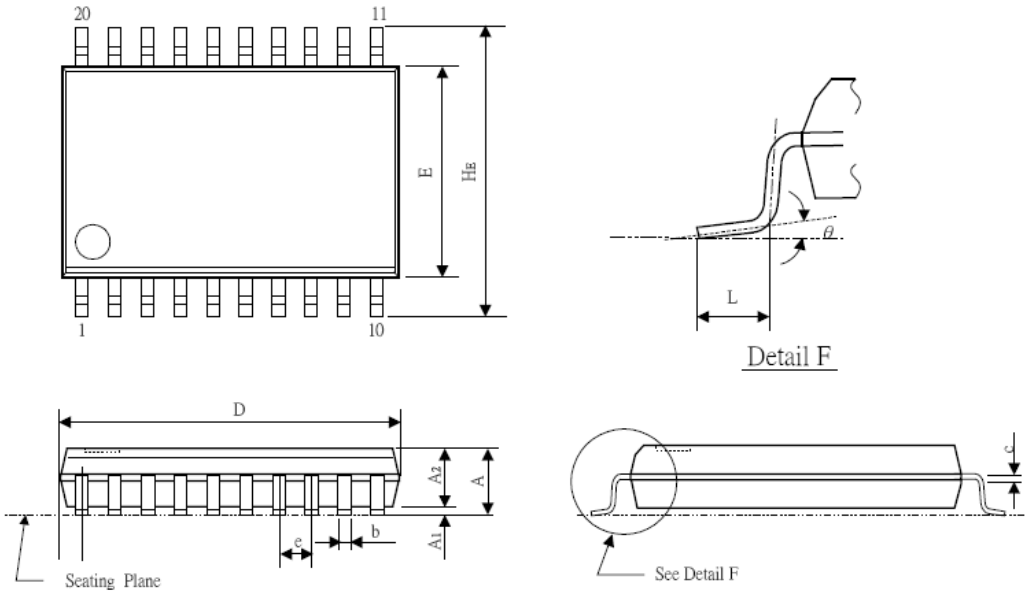






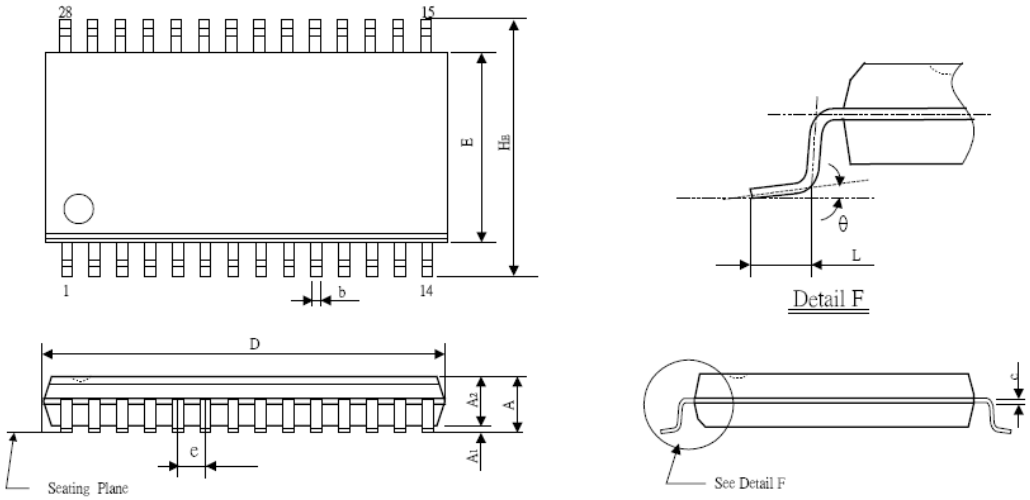
符号	说明	尺寸（mm）		
		最小	正常	最大
A	跨度	5.81	---	6.24
B	塑胶跨度	3.81	---	4.04
C	脚宽度	0.36	---	0.46
C’	塑胶体长	9.91	---	10.10
D	总高	1.40	---	1.70
E	脚间距	1.27		
F	站高	0.05	---	0.18
G	脚长	0.40	---	0.70
H	脚厚度	0.2		
a	脚角度	---	0	8

(2)SOP20 外形尺寸



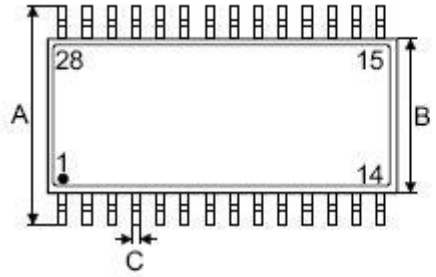
Symbol	Dimensions in inches		Dimensions in mm	
	Min	Max	Min	Max
A	0.093	0.104	2.35	2.65
A1	0.004	0.012	0.10	0.30
A2	0.083	0.098	2.10	2.50
b	0.013	0.020	0.33	0.51
c	0.008	0.013	0.20	0.33
D	0.493	0.512	12.52	13.00
E	0.291	0.299	7.40	7.60
e	0.050(BSC)		1.27(BSC)	
H_E	0.398	0.418	10.11	10.61
L	0.016	0.050	0.40	1.27
θ	0°	8°	0°	8°

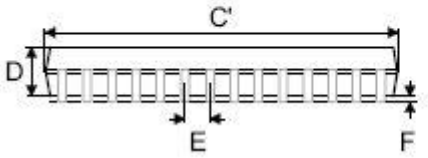
(3)SOP28 外形尺寸

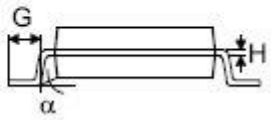


Symbol	Dimensions in inches		Dimensions in mm	
	Min	Max	Min	Max
A	0.085	0.104	2.15	2.65
A1	0.004	0.012	0.10	0.30
A2	0.081	0.098	2.05	2.50
b	0.013	0.02	0.33	0.51
c	0.008	0.014	0.20	0.36
D	0.697	0.713	17.70	18.10
E	0.291	0.3	7.40	7.62
e	0.050(BSC)		1.27(BSC)	
H_E	0.402	0.418	10.21	10.61
L	0.016	0.05	0.40	1.27
θ	0°	8°	0°	8°

(4)SSOP28 外形尺寸

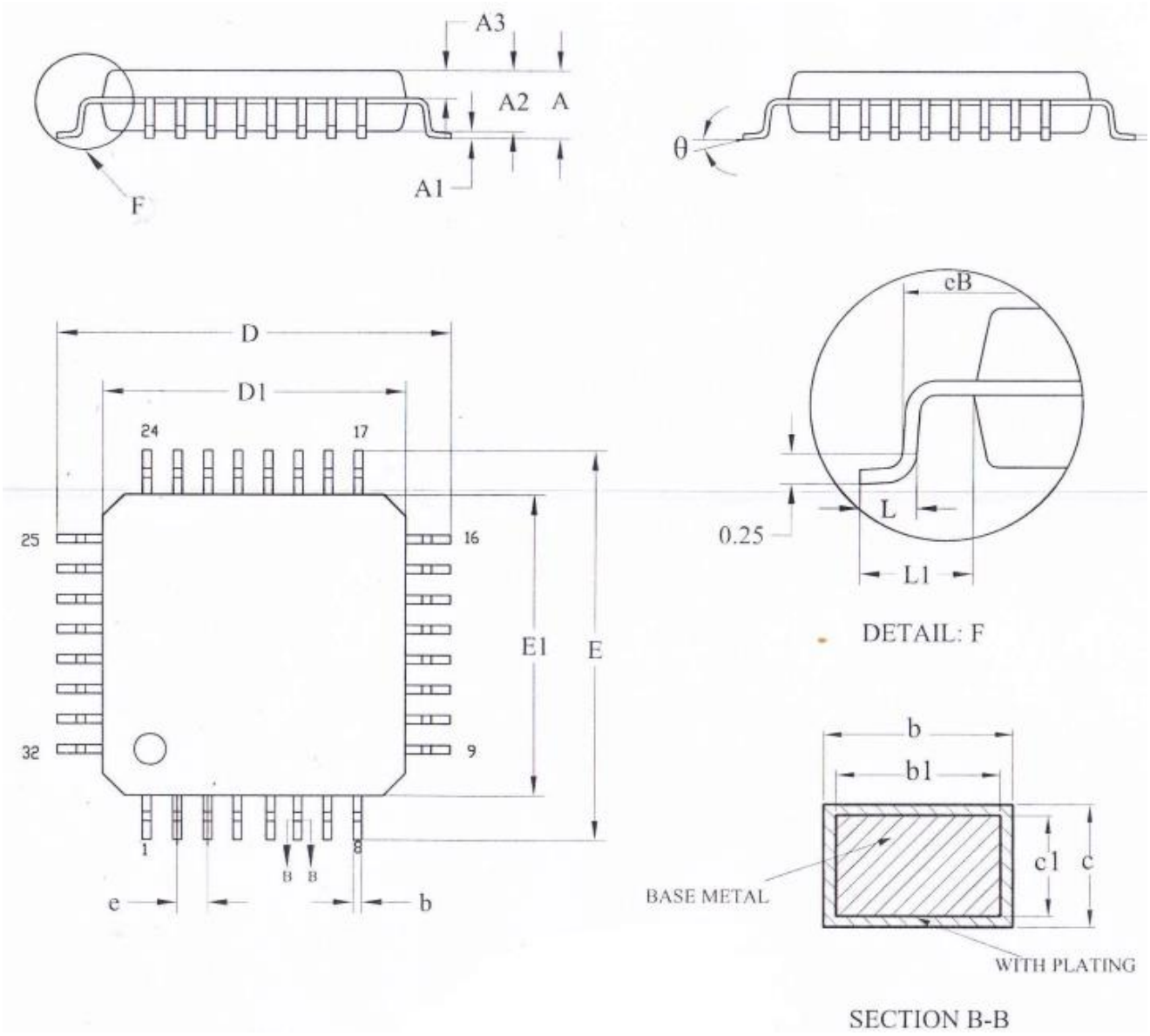






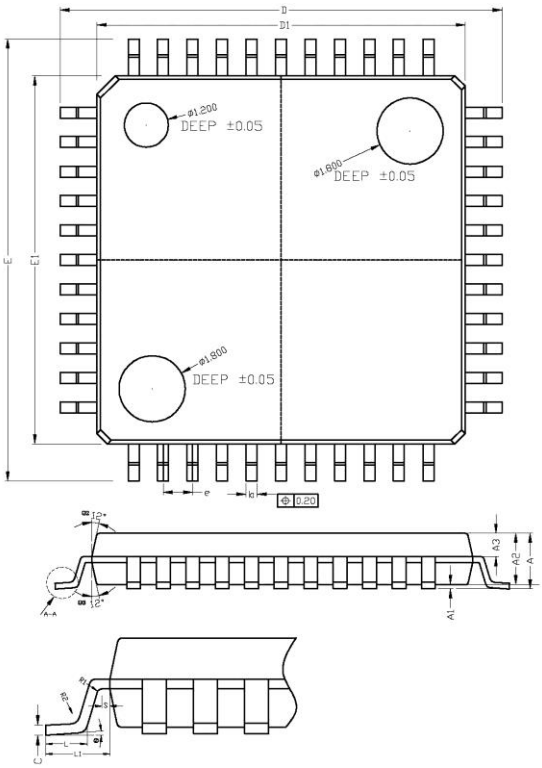
符号	说明	尺寸（mm）		
		最小	正常	最大
A	跨度	7.6	7.8	8.0
B	塑胶跨度	5.1	5.3	5.5
C	脚宽度	0.29	---	0.33
C’	塑胶体长	10	10.2	10.4
D	总高	---	---	2.0
E	脚间距	0.65		
F	站高	0.15	---	0.2
G	脚长	0.55	0.75	0.95
H	脚厚度	0.15	---	0.2
a	脚角度	---	0	8

(5)LQFP32 外形尺寸



SYMBOL	A	A1	A2	A3	b	b1	c	c1	D
MIN	-	0.05	1.35	0.59	0.33	0.32	0.13	0.12	8.80
NOM	-	-	1.40	0.64	-	0.35	-	0.13	9.00
MAX	1.6	0.15	1.45	0.69	0.41	0.38	0.17	0.14	9.20
SYMBOL	D1	E	E1	eB	e	L	L1	θ	
MIN	6.90	8.80	6.90	8.10	0.80BSC	0.45	1.00REF	0°	
NOM	7.00	9.00	7.00	-		-		-	
MAX	7.10	9.20	7.10	8.25		0.75		7°	

(6)LQFP44 外形尺寸



SYMBOL	MIN	NOM	MAX
A	–	–	1.60
A1	0.05	–	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.275	0.30	0.325
c	0.13	–	0.18
D	11.80	12.00	12.20
D1	9.90	10.00	10.10
E	11.80	12.00	12.20
E1	9.90	10.00	10.10
e	0.80BSC		
L	0.45	0.60	0.75
L1	1.00REF		
R1	0.08	–	–
R2	0.08	–	0.20
S	0.20	–	–
θ	0°	3.5°	7°
θ1	0°	–	–
θ2	11°	12°	13°
θ3	11°	12°	13°

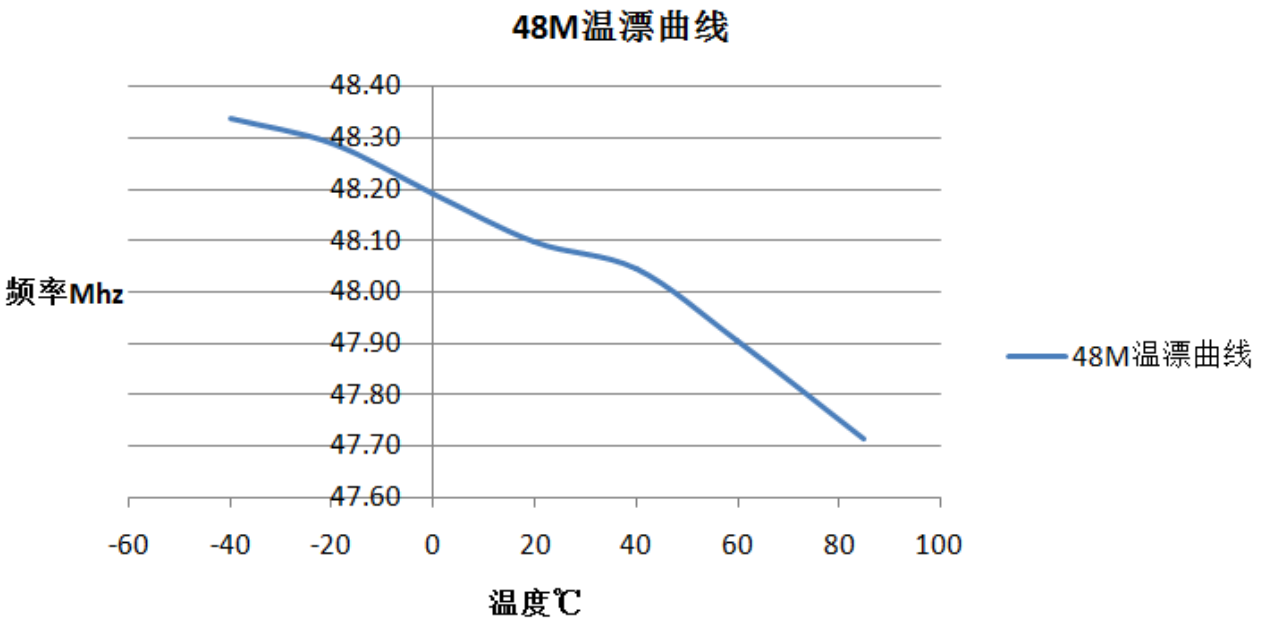
附录Ⅰ 汇编指令周期表

序号	指令	指令代码	指令长度	执行周期（取指+执行时间）
1	ACALL addr11	xxx1_0001	2	2
2	ADD A，Rn	0010_1xxx	1	2
3	ADD A,direct	0x25	2	3
4	ADD A,@Ri	0010_011x	1	2
5	ADD A,#data	0x24	2	2
6	ADDC A,Rn	0011_1xxx	1	2
7	ADDC A,direct	0x35	2	3
8	ADDC A，@Ri	0011_011x	1	2
9	ADDC A,#data	0x34	2	2
10	AJMP addr11	xxx0_0001	2	2
11	ANL A,Rn	0101_1xxx	1	2
12	ANL A,direct	0x55	2	3
13	ANL A,@Ri	0101_011x	1	2
14	ANL A,#data	0x54	2	2
15	ANL direct,A	0x52	2	3
16	ANL direct ,#data	0x53	3	3
17	ANL C,bit	0x82	2	3
18	ANL C,~bit	0xb0	2	3
19	CJNE A,direct,rel	0xb5	3	3
20	CJNE A,#data,rel	0xb4	3	3
21	CJNE Rn,#data,rel	1011_1xxx	3	3
22	CJNE @Ri,#data,rel	1011_011x	3	3
23	CLR A	0xe4	1	1
24	CLR C	0xc3	1	1
25	CLR bit	0xc2	2	2
26	CPL A	0xf4	1	1
27	CPL C	0xb3	1	1
28	CPL bit	0xb2	2	3
29	DA A	0xd4	1	1
30	DEC A	0x14	1	1
31	DEC Rn	0001_1xxx	1	2
32	DEC direct	0x15	2	3
33	DEC @Ri	0001_011x	1	2
34	DIV AB	0x84	1	3
35	DJNZ Rn,rel	1101_1xxx	2	2
36	DJNZ direct,rel	0xd5	3	3
37	INC A	0x04	1	1

38	INC Rn	0000_1xxx	1	2
39	INC direct	0x05	2	3
40	INC @Ri	0x0000_011x	1	2
41	INC DPTR	0xa3	1	4
42	JB bit,rel	0x20	3	3
43	JBC bit,rel	0x10	3	3
44	JC rel	0x40	2	2
45	JMP @A+DPRT	0x73	1	1
46	JNB bit,rel	0x30	3	3
47	JNC rel	0x50	2	2
48	JNZ rel	0x70	2	2
49	JZ rel	0x60	2	2
50	LCALL addr16	0x12	3	3
51	LJMP addr16	0x02	3	3
52	MOV A ,Rn	1110_1xxx	1	2
53	MOV A,direct	0xe5	2	3
54	MOV A,@Ri	1110_011x	1	2
55	MOV A,#data	0xe4	2	2
56	MOV Rn,A	1111_1xxx	1	1
57	MOV Rn,direct	1010_1xxx	2	3
58	MOV Rn,#data	0111_1xxx	2	1
59	MOV direct,A	0xf5	2	2
60	MOV direct,Rn	1000_1xxx	2	2
61	MOV direct,direct	0x85	3	3
62	MOV direct @Ri	1000_011x	2	2
63	MOV direct,#data	0x75	3	3
64	MOV @Ri,A	1111_011x	1	1
65	MOV @Ri,direct	1010_011x	2	3
66	MOV @Ri,#data	0111_011x	2	2
67	MOVC A,@A+DPTR	0x93	1	2
68	MOVC A,@A+PC	0x83	1	2
69	MOVX A,@Ri	1110_001x	1	2
70	MOVX A,@DPTR	0xe0	1	2
71	MOVX @RI,A	1111_001x	1	1
72	MOVX @DPTR,A	0xf0	1	1
73	MOV C,bit	0xa2	2	3
74	MOV bit,C	0x92	2	2
75	MOV DPTR,#data16	0x90	3	3
76	MUL AB	0xa4	1	3
77	NOP	0x0	1	1
78	ORL A,Rn	0100_1xxx	1	2
79	ORL A,direct	0x45	2	3

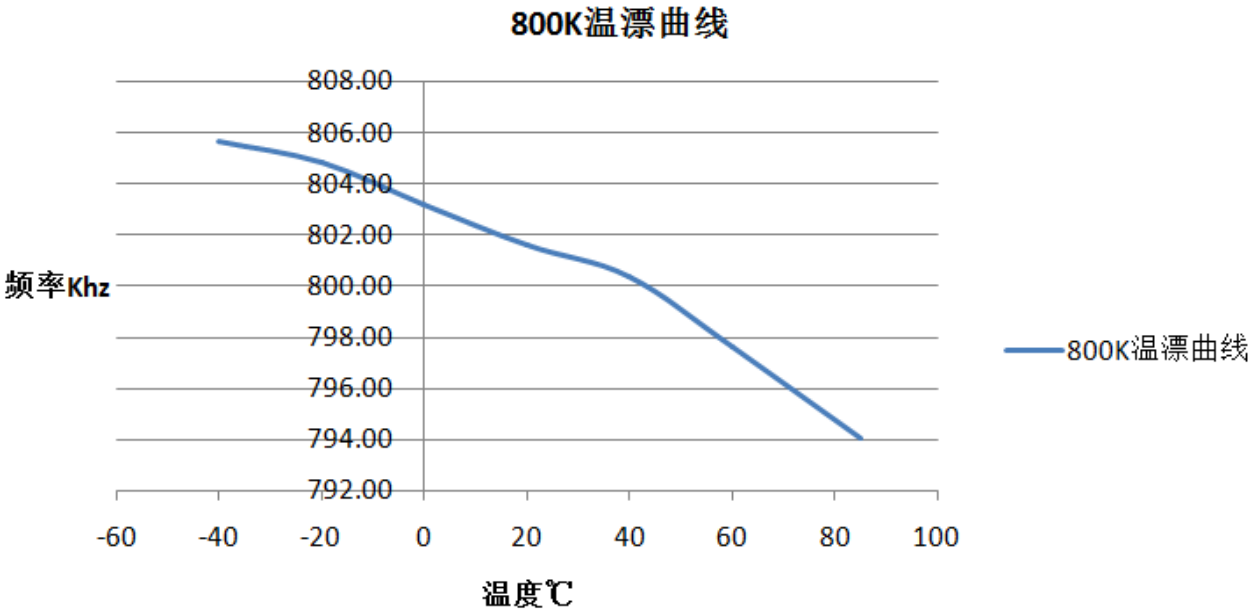
80	ORL A,@Ri	101000_011x	1	2
81	ORL A,#data	0x44	2	2
82	ORL direct,A	0x42	2	3
83	ORL direct,#data	0x43	3	3
84	ORL C,bit	0x72	2	3
85	ORL C,~bit	0xa0	2	3
86	POP direct	0xd0	2	2
87	PUSH direct	0xc0	2	3
88	RET	0x22	1	3
89	RETI	0x32	1	3
90	RL A	0x23	1	1
91	RLC A	0x33	1	1
92	RR A	0x03	1	1
93	RRC A	0x13	1	1
94	SETB C	0xd3	1	1
95	SETB bit	0xd2	2	2
96	SJMP rel	0x80	2	2
97	SUBB A,Rn	1001_1xxx	1	2
98	SUBB A,direct	0x95	2	3
99	SUBB A,@Ri	1001_011x	1	2
100	SUBB A,#data	0x94	2	2
101	SWAP A	0xC4	1	1
102	XCH A,Rn	1100_1xxx	1	3
103	XCH A,direct	0xc5	2	4
104	XCH A,@Ri	1100_011x	1	3
105	XCHD A,@Ri	1101_011x	1	3
106	XRL A,Rn	0110_1xxx	1	2
107	XRL A,direct	0x65	2	3
108	XRL A,@Ri	0110_011x	1	2
109	XRL A,#data	0x64	2	2
110	XRL direct,A	0x62	2	3
111	XRL direct,#data	0x63	3	3
备注:@Ri 为@R0,@R1;Rn 为 R0~R7				

附录Ⅱ 内部高速时钟 48M 温漂曲线



(VDD=5v,-40℃~85℃)

附录Ⅲ 内部低速时钟 800K 温漂曲线



(VDD=5v,-40℃~85℃)

附录Ⅳ ADC 采样温漂曲线（包含 Vref 全温度偏差）

1、采样数据

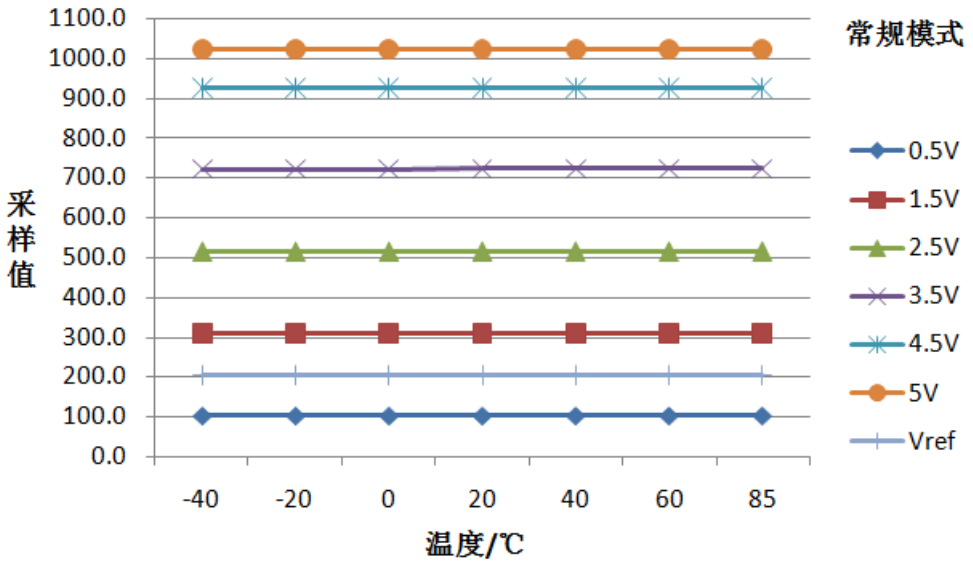
ADC 采样温漂测试数据(包含 Vref 全温度偏差)															
电压/V 温度/℃	0.5		1.5		2.5		3.5		4.5		5		Vref_1v	
	常规	硬件	常规	硬件	常规	硬件	常规	硬件	常规	硬件	常规	硬件	常规	硬件
-40	101.4	508.0	307.5	1549.4	514.6	2585.0	721.3	3625.5	926.5	4653.7	1023.0	5136.9	203.8	1024.7
-20	101.3	507.8	307.4	1548.2	514.6	2585.4	721.3	3624.0	926.4	4653.2	1023.0	5140.4	203.7	1024.1
0	101.4	508.1	307.1	1544.8	514.7	2584.3	721.5	3622.6	926.5	4650.8	1023.0	5135.6	204.0	1024.4
20	101.4	507.7	307.3	1545.3	514.9	2583.7	721.7	3621.2	926.6	4649.6	1023.0	5134.3	204.0	1024.4
40	101.5	507.7	307.6	1547.2	514.8	2584.2	721.9	3622.3	926.7	4650.7	1023.0	5134.5	204.0	1025.0
60	101.6	508.9	308.0	1548.7	515.0	2585.5	721.9	3623.8	926.9	4654.2	1023.0	5135.3	203.9	1024.1
85	101.6	510.1	308.2	1552.0	515.1	2591.3	722.0	3631.6	926.9	4662.7	1023.0	5144.6	203.6	1023.8
MAX	101.6	510.1	308.2	1552.0	515.1	2591.3	722.0	3631.6	926.9	4662.7	1023.0	5144.6	204.0	1025.0
MIN	101.3	507.7	307.1	1544.8	514.6	2583.7	721.3	3621.2	926.4	4649.6	1023.0	5134.3	203.6	1023.8
100*(MAX-20℃)/20℃	0.16	0.46	0.27	0.43	0.03	0.29	0.05	0.29	0.04	0.28	0.00	0.20	0.00	0.06
100*(MIN-20℃)/20℃	-0.10	0.00	-0.07	-0.03	-0.07	0.00	-0.05	0.00	-0.02	0.00	0.00	0.00	-0.19	-0.06
20℃对应实测电压/V	0.495	0.495	1.501	1.509	2.514	2.522	3.524	3.536	4.524	4.540	4.995	5.014	0.996	1.000
20℃实测电压误差/V	0.005	0.005	-0.001	-0.009	-0.014	-0.022	-0.024	-0.036	-0.024	-0.040	0.005	-0.014	0.004	0.000
注：100(MAX-20℃)/20℃：相对 20℃的最大正偏差采样值百分比；　100*(MIN-20℃)/20℃：相对 20℃的最大负偏差采样值百分比。*														

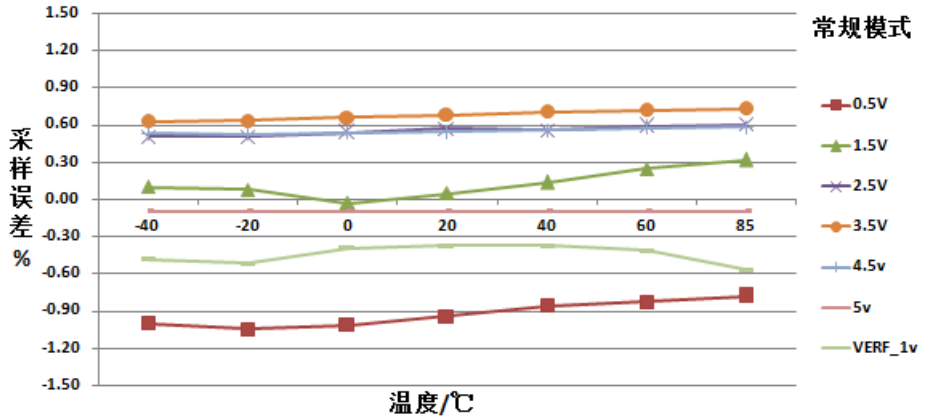
如上图中曲线所示：

1>、MCU 在-40℃～85℃工作环境下，常规模式采样值相对于 20℃工作环境下的漂移为±0.3%。

2>、MCU 在-40℃～85℃工作环境下，硬件模式采样值相对于 20℃工作环境下的漂移为±0.5%。

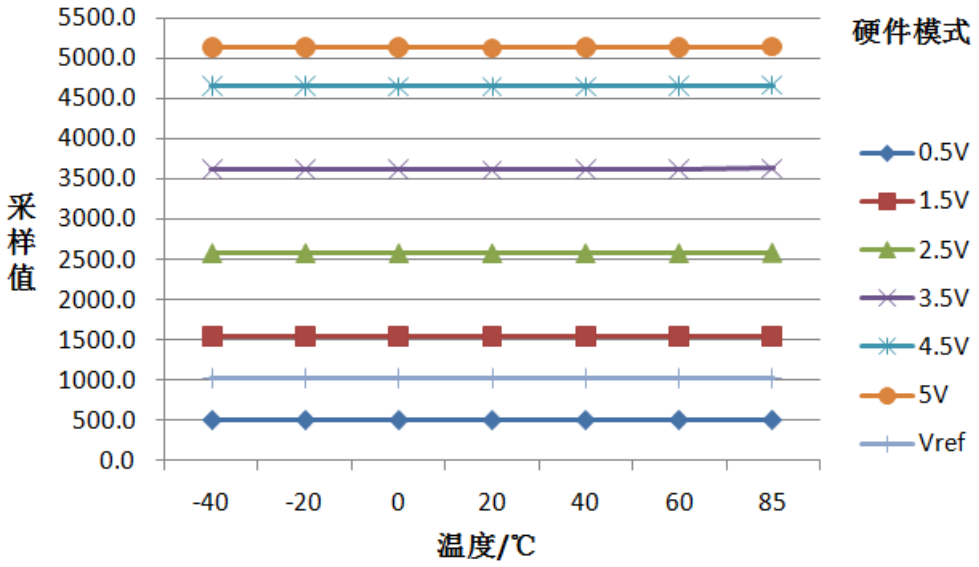
2、常规方式温漂曲线（图表对应上表中的数据）

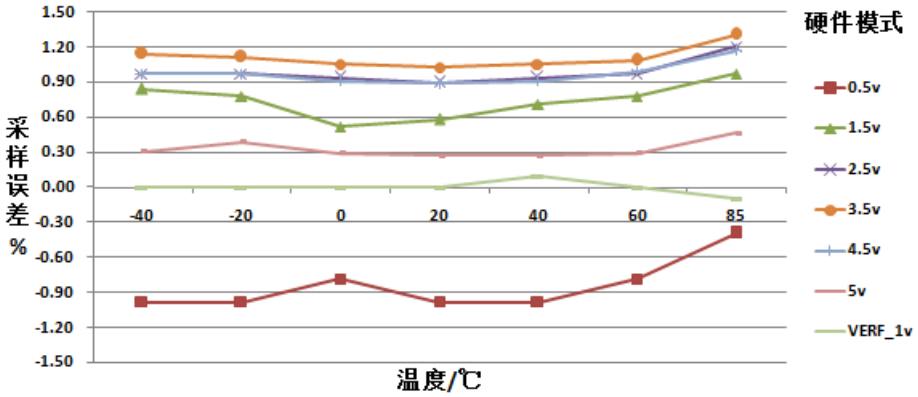




如上图中曲线所示：MCU 在-40℃～85℃工作环境下 ADC 的常规模式采样值误差为±1.5%。

3、硬件方式温漂曲线（图表对应上表中的数据）





如上图中曲线所示：MCU 在-40℃～85℃工作环境下，以 MCU 内部 vref_1v 为参考的硬件模式采样值误差为±2.5%（因为内部 vref_1v 的一致性误差为±1%）。

附录Ⅴ XBYTE 类型地址寄存器的操作说明

例如 **XBYTE** 类型地址寄存器：P0OE(P0 端口方向控制寄存器)、P0PURSELRL（P0 端口上拉电阻选择控制寄存器）

```
#define     P0OE            XBYTE[0x2AFF]
    #define     P0OE_7              XBYTE[0x2A80]
    #define     P0OE_6              XBYTE[0x2A40]
    #define     P0OE_5              XBYTE[0x2A20]
    #define     P0OE_4              XBYTE[0x2A10]
    #define     P0OE_3              XBYTE[0x2A08]
    #define     P0OE_2              XBYTE[0x2A04]
    #define     P0OE_1              XBYTE[0x2A02]
    #define     P0OE_0              XBYTE[0x2A01]

#define     P0PURSELRL      XBYTE[0x30FF]
    #define     P0PURSELRL_3        XBYTE[0x30C0]
    #define     P0PURSELRL_2        XBYTE[0x3030]
    #define     P0PURSELRL_1        XBYTE[0x300C]
    #define     P0PURSELRL_0        XBYTE[0x3003]
```

一、P0OE 寄存器

1、将 P0OE 寄存器的 P0OE_7 清 0（P0OE_7 即 P07 端口的方向寄存器）

方式一(**对字节操作**)：P0OE &= 0x7F;

方式二(**对位操作**)：P0OE_7 = 0<<7;

2、将 P0OE 寄存器的 P0OE_7 置为 1：

方式一(对字节操作)：P0OE |= 0x80;

方式二(对位操作)：P0OE_7 = 1<<7;

二、P0PURSELRL 寄存器

3、将 P0PURSELRL 寄存器的 P0PURSELRL_3 清 0（P0PURSELRL_3 即 P03 端口的电阻档位寄存器）：

方式一(**对字节操作**)：P0PURSELRL &= 0x3F;

方式二(**对位操作**)：P0PURSELRL_3 = 0x00; //(b00000000)

4、将 P0PURSELRL 寄存器的 P0PURSELRL_3 置为 1：

方式一(对字节操作)：P0PURSELRL &= 0x3F;　P0PURSELRL |= 0x40;

方式二(对位操作)：P0PURSELRL_3 = 0x40;//(b01000000)

5、将 P0PURSELRL 寄存器的 P0PURSELRL_3 置为 2：

方式一(对字节操作)：P0PURSELRL &= 0x3F;　P0PURSELRL |= 0x80;

方式二(对位操作)：P0PURSELRL_3 = 0x80; //(b10000000)

6、将 P0PURSELRL 寄存器的 P0PURSELRL_3 置为 3：

方式一(对字节操作)：P0PURSELRL &= 0x3F;　P0PURSELRL |= 0xC0;

方式二(对位操作)：P0PURSELRL_3 = 0xC0; //(b11000000)

其它的 XBYTE 类型地址寄存器的操作依次类推。

附录VI IO端口电流档位（6mA、30mA、60mA）驱动特性

说明：

1、io口具有限流能力，在一定程度上可以避免io短路异常时引起电流过大而损坏IO或芯片的情况；

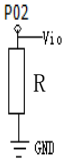
2、如下表，当IO所接的负载一定时，如果得到的驱动电流I或者驱动电压Vio达不到需求，则需调大IO电流驱动档位；

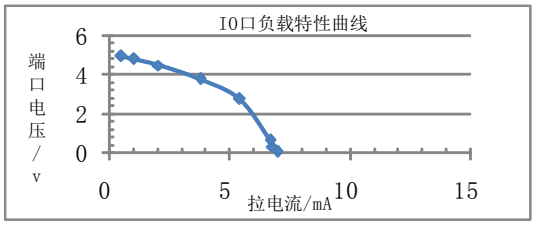
3、当测量工具不一致时，测得数据可能与表格数据存在一定误差，对应电流档位值的范围请参考“22章节的直流电气特性”。

IO口驱动特性-6mA档

拉电流（IO输出1）

Vcc/v	Vio/v	R/欧姆	I拉/mA	IO输出内阻
5.11	4.96	10000	0.5	302.4
5.11	4.81	4700	1.0	293.1
5.11	4.48	2200	2.0	309.4
5.11	3.78	1000	3.8	351.9
5.11	2.76	510	5.4	434.2
5.11	0.67	100	6.7	662.7
5.11	0.34	50	6.8	701.5
5.11	0.07	10	7.0	720.0

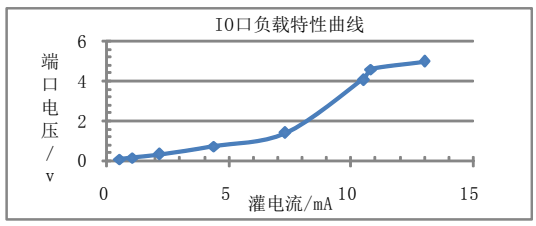




灌电流（IO输出0）

Vcc/v	Vio/v	R/欧姆	I灌/mA	IO输入内阻
5.11	0.08	10000	0.5	159.0
5.11	0.16	4700	1.1	151.9
5.11	0.34	2200	2.2	156.8
5.11	0.73	1000	4.4	166.7
5.11	1.39	510	7.3	190.6
5.11	4.06	100	10.5	386.7
5.11	4.57	50	10.8	423.1
5.11	4.98	10	13.0	383.1

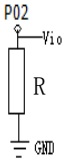


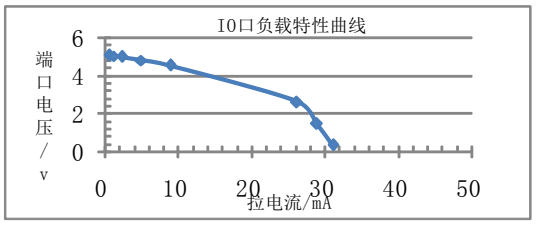


IO口驱动特性-30mA档

拉电流（IO输出1）

Vcc/v	Vio/v	R/欧姆	I拉/mA	IO输出内阻
5.11	5.08	10000	0.5	59.1
5.11	5.04	4700	1.1	65.3
5.11	4.97	2200	2.3	62.0
5.11	4.81	1000	4.8	62.4
5.11	4.53	510	8.9	65.3
5.11	2.61	100	26.1	95.8
5.11	1.44	50	28.8	127.4
5.11	0.31	10	31.0	154.8

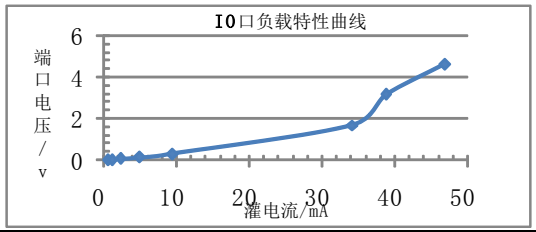




灌电流（IO输出0）

Vcc/v	Vio/v	R/欧姆	I灌/mA	IO输入内阻
5.11	0.01	10000	0.5	19.6
5.11	0.03	4700	1.1	27.8
5.11	0.07	2200	2.3	30.6
5.11	0.15	1000	5.0	30.2
5.11	0.30	510	9.4	31.8
5.11	1.69	100	34.2	49.4
5.11	3.17	50	38.8	81.7
5.11	4.64	10	47.0	98.7

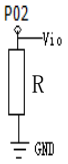


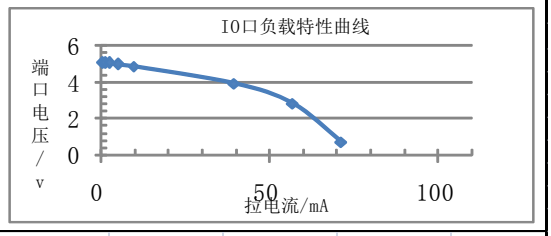


IO口驱动特性-60mA档

拉电流（IO输出1）

Vcc/v	Vio/v	R/欧姆	I拉/mA	IO输出内阻
5.11	5.09	10000	0.5	39.3
5.11	5.08	4700	1.1	27.8
5.11	5.05	2200	2.3	26.1
5.11	4.98	1000	5.0	26.1
5.11	4.86	510	9.5	26.2
5.11	3.93	100	39.3	30.0
5.11	2.83	50	56.6	40.3
5.11	0.71	10	71.0	62.0





灌电流（IO输出0）

Vcc/v	Vio/v	R/欧姆	I灌/mA	IO输入内阻
5.11	0.00	10000	0.5	0.0
5.11	0.00	4700	1.1	0.0
5.11	0.02	2200	2.3	8.6
5.11	0.06	1000	5.1	11.9
5.11	0.12	510	9.8	12.3
5.11	0.61	100	45.0	13.6
5.11	1.41	50	74.0	19.1
5.11	4.06	10	105.0	38.7



