

HC18P13xL

数据手册

28 引脚 8 位
AD 型 OTP 单片机

目录

1 产品简述.....	5
1.1 特性.....	5
1.2 系统框图.....	7
1.3 引脚图.....	8
1.4 引脚说明.....	10
1.5 引脚电路.....	12
2 电性参数.....	13
2.1 极限参数.....	13
2.2 直流特性.....	13
2.3 交流特性.....	14
2.4 电气特性曲线图.....	14
3 中央处理器（CPU）.....	15
3.1 存储器.....	15
3.2 寻址模式.....	27
3.3 堆栈.....	28
4 复位.....	29
4.1 概述.....	29
4.2 上电复位.....	30
4.3 看门狗定时器复位.....	30
4.4 欠压复位.....	31
4.5 外部复位.....	32
5 系统时钟.....	35
5.1 概述.....	35
5.2 时钟框图.....	35
5.3 系统高频时钟.....	36
5.4 系统低频时钟.....	37
6 系统工作模式.....	39
6.1 模式切换举例.....	40
6.2 高低频模式切换.....	41
6.3 唤醒时间.....	42
6.4 OSCCON 寄存器.....	42
7 中断源.....	43
7.1 内核中断.....	44
7.2 外设中断.....	45
7.3 GIE 全局中断.....	48
7.4 中断保护.....	48
7.5 TIMER0 定时器中断.....	49

7.6 INT0 外部中断.....	49
7.7 PORT 电平变化中断.....	50
7.8 TIMER2 定时器中断.....	51
7.9 TIMER1 中断.....	52
7.10 AD 中断.....	53
7.11 CCP 中断.....	53
7.12 UART 中断.....	53
7.13 PWM 中断.....	53
7.14 多中断操作.....	53
8 I/O 口.....	55
8.1 I/O 口输入输出控制寄存器.....	55
8.2 I/O 口上拉控制寄存器.....	56
8.3 I/O 口下拉控制寄存器.....	57
8.4 PORT 驱动控制寄存器.....	57
8.5 I/O 口数据寄存器.....	59
8.6 管脚配置寄存器.....	59
9 定时器/计数器.....	60
9.1 看门狗定时器.....	60
9.2 TIMER0 定时器/计数器.....	61
9.3 TIMER1 定时器/计数器.....	64
9.4 TIMER2 定时器.....	66
9.5 CCP 模块.....	67
10 PWM 模块.....	76
10.1 概述.....	76
10.2 PWM 相关寄存器.....	76
10.3 死区时间.....	79
11 模数转换 (ADC).....	81
11.1 ADC 概述.....	81
11.2 A/D 寄存器.....	81
11.3 A/D 控制寄存器.....	82
11.4 AD 转换时间.....	83
11.5 ADC 使用.....	85
12 串行口通信.....	87
12.1 串行口的控制寄存器 SCON.....	87
12.2 串行口数据缓冲寄存器 SBUF.....	88
12.3 辅助寄存器 AUXR.....	88
12.4 独立波特率发生器寄存器 BRT.....	89
12.5 自动地址识别.....	89
12.6 串行口工作模式.....	91
12.7.串口通信中波特率的设置.....	97

13 软件 LCD 驱动	99
13.1 相关寄存器	99
13.2 软件 LCD 操作说明	100
14 开发工具	103
14.1 OTP 烧录器 (PM18-4.0)	103
14.2 HC-IDE	103
15 封装尺寸	104
15.1 SOP16	104
15.2 SOP20	104
15.3 SOP24	105
15.4 SSOP24	106
15.5 SOP28	106
16 修改记录	107

1 产品简述

HC18P13xL是一颗采用高速低功耗CMOS工艺设计开发的8位高性能精简指令单片机，内部有4K×16位一次性编程ROM(OTP-ROM)，256×8位的数据寄存器（RAM），4组双向I/O口，三个Timer定时器/计数器，两个CCP模块，支持一路UART。一个16通道的12位模数转换器，多个系统时钟，四种系统工作模式以及多个中断源。这款单片机可以广泛应用于液晶梳子、电动车码表等产品。

1.1 特性

- CPU 特性
 - 36条高性能精简指令
 - 4K×16位的OTP程序存储器
 - 256×8位的数据存储器
 - 8级堆栈缓存器
 - 2T/4T时钟模式
 - 立即、直接和两组间接寻址模式
 - 16位RDT查表
- I/O 口
 - 4组双向I/O口：PORTA, PORTB, PORTC, PORTF
 - 最多26个双向I/O口
 - 所有端口4级驱动电流配置(除PORTB5)
 - 最多26个可编程弱上拉/下拉口(PA、PB、PC、PF)
 - 所有端口具有唤醒功能的电平变化中断
- 所有端口支持软件 1/2bias COM 口功能 (除PORTB5)
- 三个 Timer 定时器/计时器
 - Timer0: 带有8位预分频器的8位定时器/计数器
 - Timer1: 带有预分频器的16位定时器/计数器
 - Timer2: 带有8位周期寄存器的8位定时器
- 两个CCP模块
 - 16位捕捉、16位比较、最高10位PWM
- 1组可编程带死区控制的固定相位PWM 1*12bit
- BOR复位系统
 - 2.0V/2.4V/3.6V
- 一个 UART 模块
- 模数转换器
 - 12位转换分辨率
 - 最多16个模拟输入通道（15个外部ADC输入，1个内部1/4VDD检测）
 - 内部参考电压(VDD、4V、3V、2V)；外部参考电压
- 双系统时钟
 - 高频系统时钟
 - 高频晶体振荡器：最高 20MHz
 - 内部 RC 振荡器：高达 32Mhz
 - 低频系统时钟
 - 低频晶体振荡器：32.768KHz
 - 低频 RC 振荡器：32K(5V 典型值)
- 系统工作模式
 - 高频模式
 - 低频模式
 - 休眠模式
 - 绿色模式
- 中断源
 - 定时器中断：Timer0、Timer1和Timer2
 - INT0外部中断
 - 所有IO电平变化中断
 - CCP1/CCP2中断
 - ADC中断
 - UART中断
 - PWM中断
- 复位
 - 上电复位（POR）
 - 外部复位（MCLR Reset）
 - 欠压复位（BOR）
 - 看门狗定时器复位（WDT Reset）

✓ 选型表

产品型号	ROM	RAM	Freq	IO	AD	Timer	PWM	INT	WDT
HC18P132L	4K*16	256*8	8MHz	14	9+1	3	4	14	1
HC18P133L	4K*16	256*8	8MHz	18	9+1	3	4	18	1
HC18P133L	4K*16	256*8	8MHz	22	13+1	3	4	22	1
HC18P134L	4K*16	256*8	8MHz	26	15+1	3	4	26	1

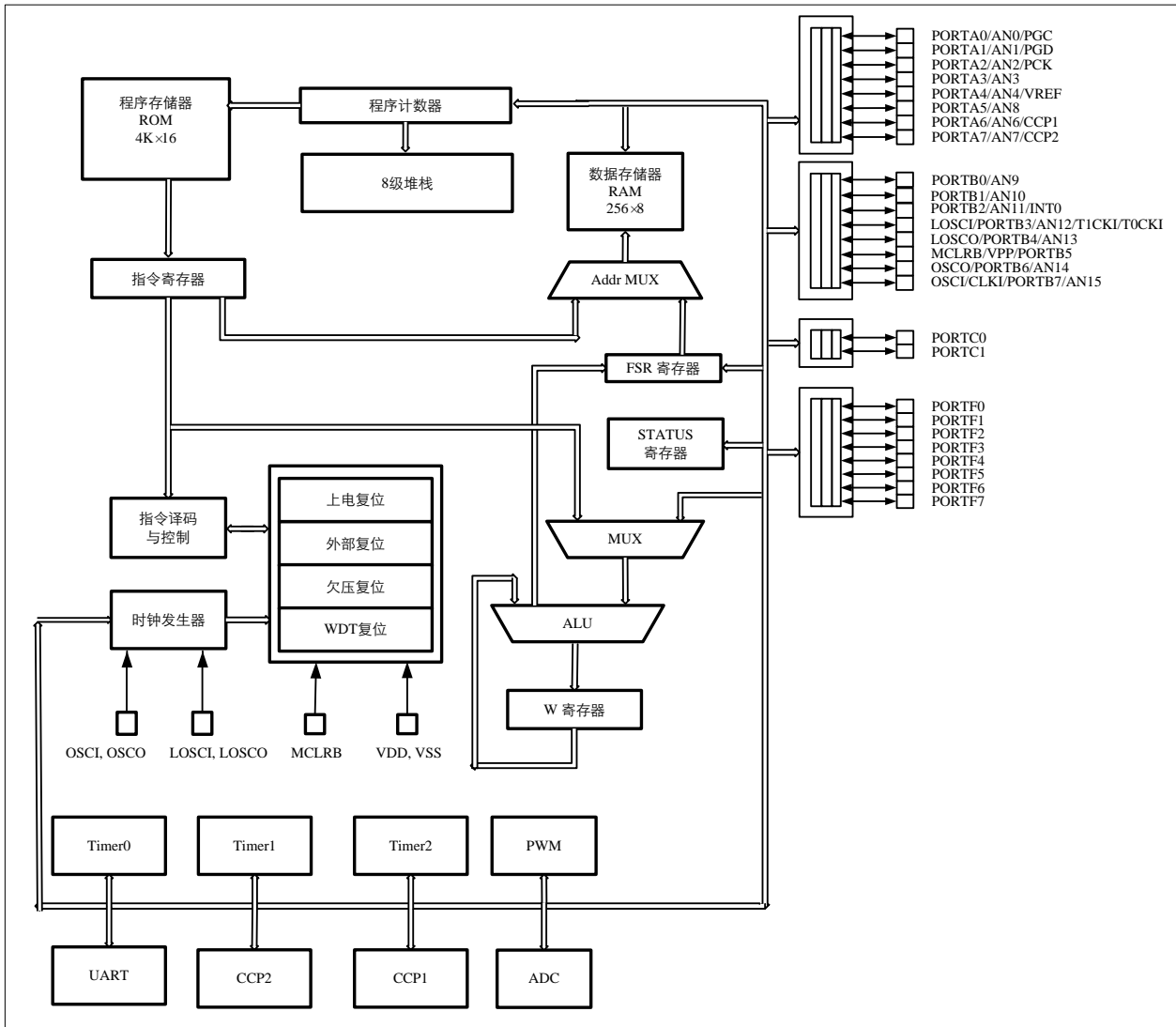
产品型号	Voltage	LCD	堆栈	EV Board	Programmer	Demo Code	PKG
HC18P132L	2.4~5.5V	Software	8	√	PM18-4.0	√	SOP16
HC18P133L	2.4~5.5V	Software	8	√	PM18-4.0	√	SOP20/QFN20
HC18P133L	2.4~5.5V	Software	8	√	PM18-4.0	√	SOP24/SSOP24
HC18P134L	2.4~5.5V	Software	8	√	PM18-4.0	√	SOP28

注:选型表中工作电压对应为 Freq 下工作电压范围, 在 Fcpu = 4MHz 时, 工作电压范围: 2.4~5.5V

使用注意事项:

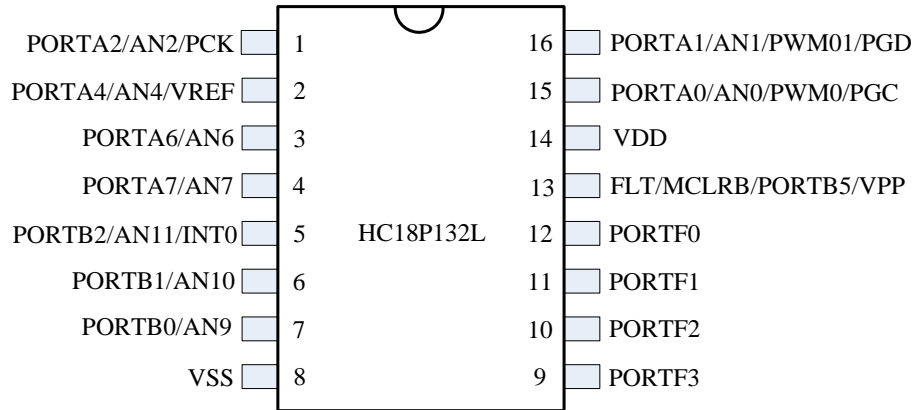
- 1、为保证系统的稳定性, 建议在 VDD 和 GND 之间接一个电容 (容值须等于或大于 0.1μF)。
- 2、在系统对功耗要求较高时 ADC 推荐使用 2M 及以下采样时钟;
- 3、为提高 ADC 检测精度, 建议在 VDD 和 GND 之间并一个电容 (104 电容即可)。
- 4、当使用 ADC 模块时, 不论参考电压选择的是什么, 系统工作电压 VDD 必须高于 2.7V。

1.2 系统框图

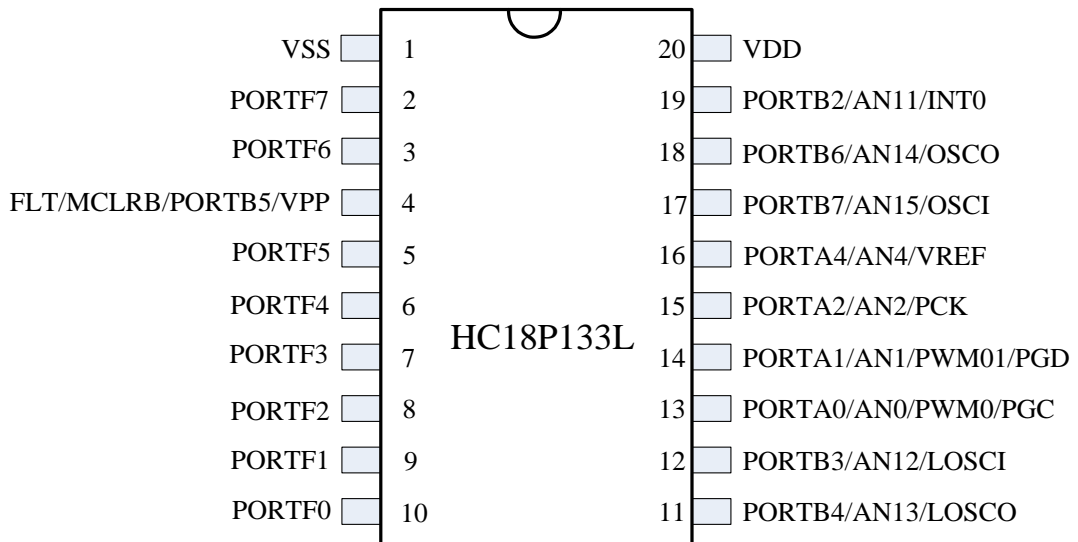


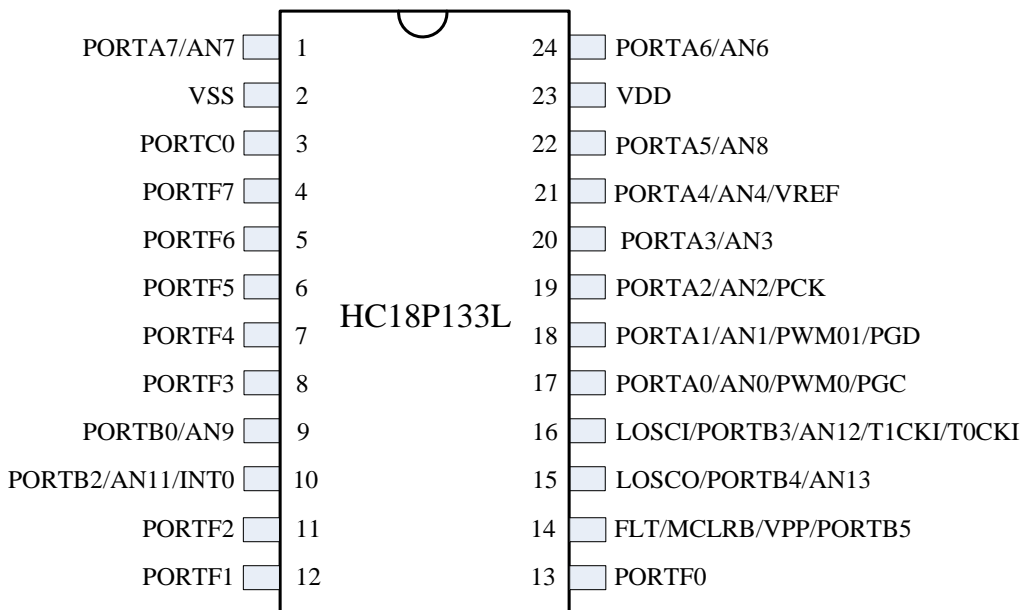
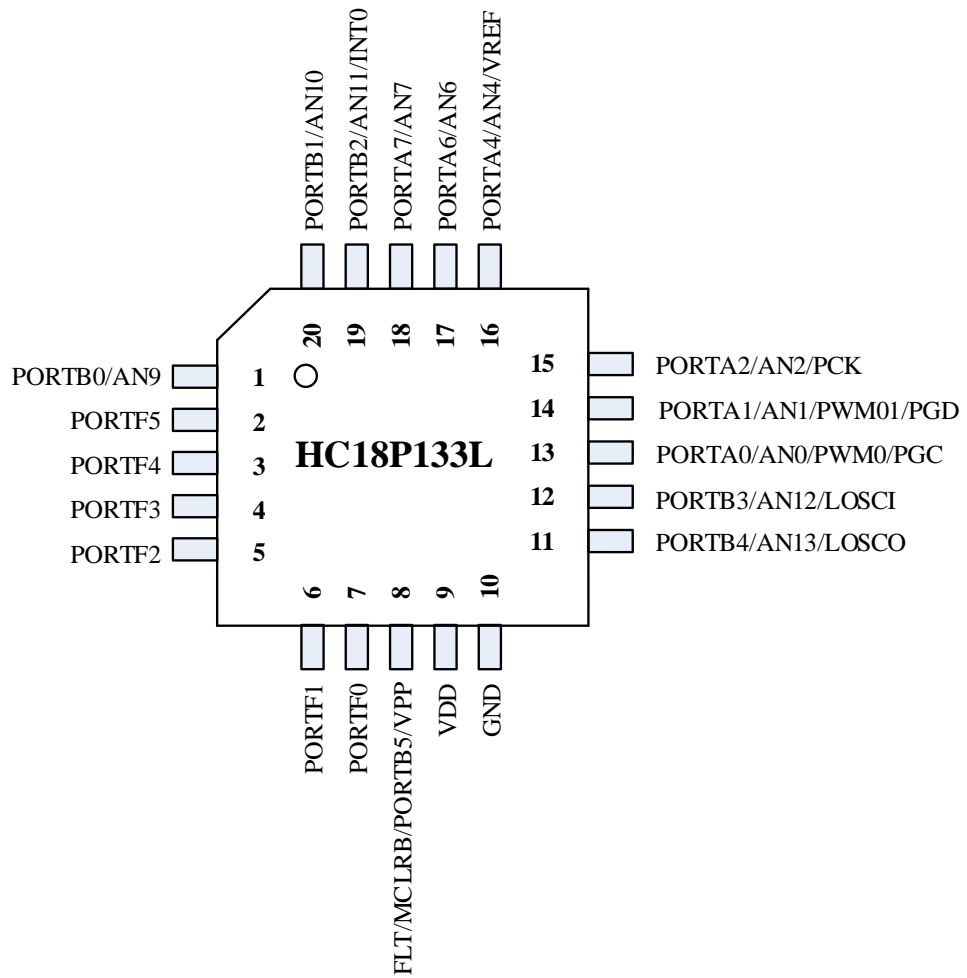
1.3 引脚图

1.3.1 HC18P132L 引脚图

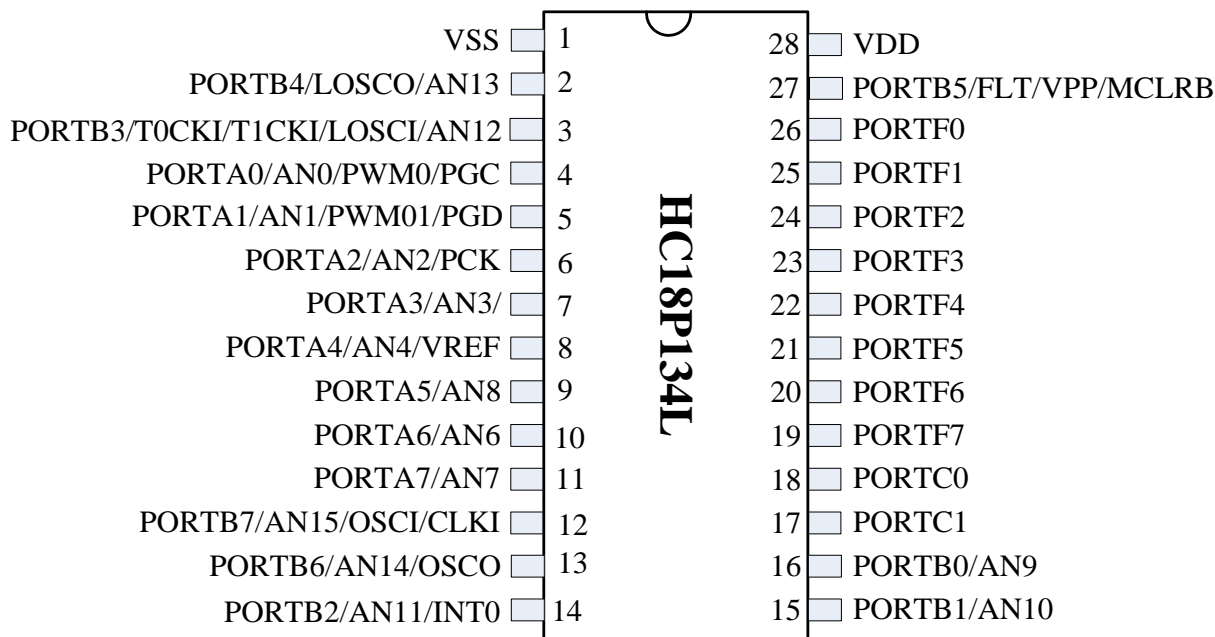


1.3.2 HC18P133L 引脚图





1.3.3 HC18P134L 引脚图



1.4 引脚说明

SOP28	名称	类型	说明
1	VSS	P	电源地
2	PORTB4 LOSCO AN13	I/O O AN	输入/输出口，带可编程上下拉，端口电平变化中断 低频晶体振荡器输出口 AD通道13输入口
3	PORTB3 T0CKI T1CKI LOSCI AN12	I/O I I I AN	输入/输出口，带可编程上下拉，端口电平变化中断 Timer0外部时钟输入口（施密特触发器） Timer1外部时钟输入口（施密特触发器） 低频晶体振荡器输入口 AD通道12输入口
4	PORTA0 AN0 PWM0 PGC	I/O AN O I	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道0输入口 12位PWM0输出口 编程时钟输入口
5	PORTA1 AN1 PWM01 PGD	I/O AN O I/O	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道1输入口 与PWM0有固定相位关系的12位PWM输出 编程数据输入/输出口
6	PORTA2 AN2 PCK	I/O AN O	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道2输入口 测试模式下内部RC输出口
7	PORTA3 AN3	I/O AN	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道3输入口
8	PORTA4	I/O	输入/输出口，带可编程上下拉，端口电平变化中断

	AN4 VREF	AN AN	AD通道4输入口 ADC参考电压输入口
9	PORTA5 AN8	I/O AN	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道8输入口
10	PORTA6 AN6	I/O AN	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道6输入口
11	PORTA7 AN7	I/O AN	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道7输入口
12	PORTB7 AN15 OSCI CLKI	I/O AN15 I I	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道15输入口 高频晶体振荡器输入口 外部时钟输入口
13	PORTB6 AN14 OSCO	I/O AN O	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道14输入口 高频晶体振荡器输出口
14	PORTB2 AN11 INT0 CCP1	I/O AN I I/O	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道11输入口 INT0外部输入口 CCP输入输出口
15	PORTB1 AN10	I/O AN	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道10输入口
16	PORTB0 AN9	I/O AN	输入/输出口，带可编程上下拉，端口电平变化中断 AD通道9输入口
17	PORTC1	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
18	PORTC0	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
19	PORTF7	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
20	PORTF6	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
21	PORTF5	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
22	PORTF4	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
23	PORTF3	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
24	PORTF2	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
25	PORTF1	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
26	PORTF0	I/O	输入/输出口，带可编程上下拉，端口电平变化中断
27	MCLRB VPP PORTB5 FLT	I P I/O I	复位输入口，低电平有效 编程高压电源输入 输入/输出口，只能输出0，端口电平变化中断 PWM故障检测输入口
28	VDD	P	电源输入

注: I = 输入 O = 输出 I/O = 输入/ 输出 P = 电源 AN = 模拟输入输出

1.5 引脚电路

图 1-1: PORTA[7:5][3:0]的等效电路

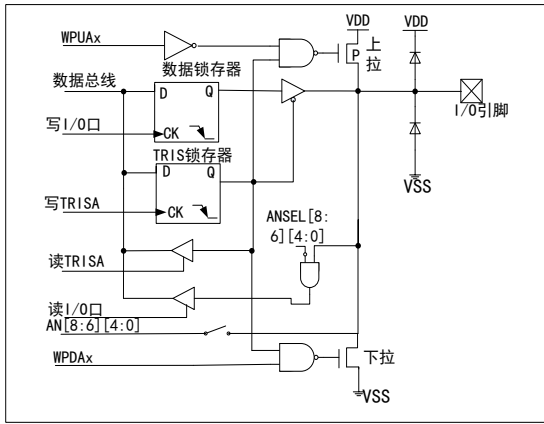


图1-3: PORTB[7:6][4:0]的等效电路

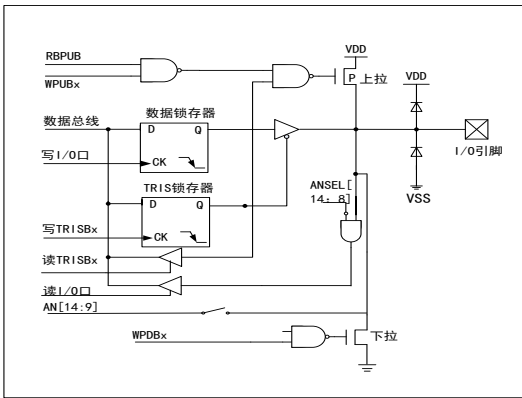


图 1-5: PORTC[1:0]口的等效电路

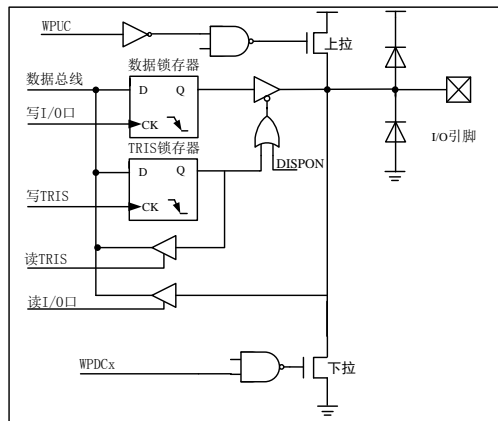


图1-2: PORTA4的等效电路

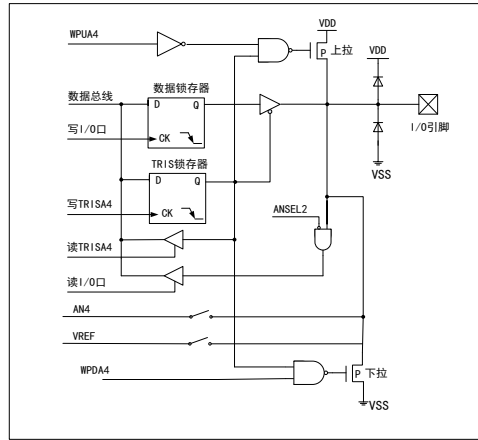


图1-4: PORTB5口的等效电路

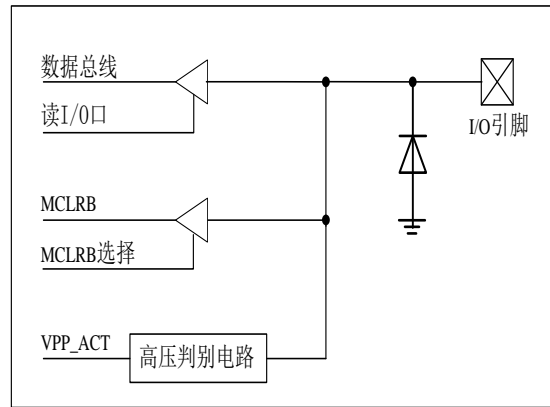
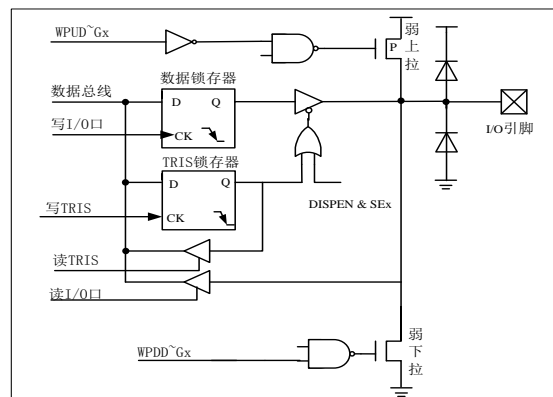


图 1-6: PORTF 口的等效电路



2 电性参数

2.1 极限参数

储存温度.....	-50℃~125℃
工作温度.....	-40℃~85℃
电源供应电压.....	VSS-0.3V~VSS+6.0V
端口输入电压.....	VSS-0.3V~VDD+0.3V

2.2 直流特性

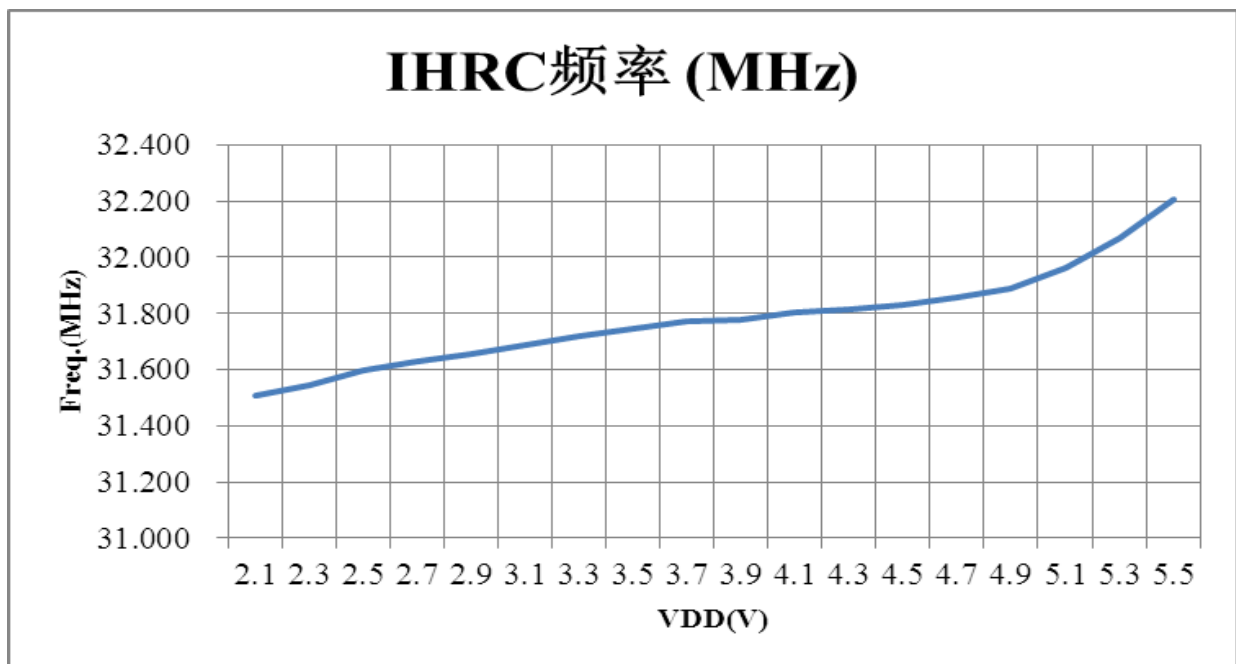
符号	参数	测试条件		最小值	典型值	最大值	单位
		VDD	条件 (常温 25℃)				
VDD	工作电压	—	Fsys = 8MHz, 2T	2.4	—	5.5	V
		—	Fsys = 16MHz, 2T	3.0	—	5.5	V
IDD1	工作电流	3V	Fsys = 16MHz, 4T, 高频模式, WDT 禁止, 无负载	—	1.70	—	mA
		5V		—	3.00	—	mA
IDD2	工作电流	3V	Fsys = 8MHz, 4T, 高频模式, WDT 禁止, 无负载	—	1.20	—	mA
		5V		—	2.20	—	mA
IDD3	工作电流	3V	Fsys = 32KHz, 4T, 低频模式, WDT 禁止, 无负载	—	5.0	—	μA
		5V		—	15.0	—	μA
IDD4	工作电流	3V	Fsys = 32KHz, 4T, 绿色模式, WDT 禁止, 无负载	—	2.0	—	μA
		5V		—	7.0	—	μA
Isb4	静态电流	3V	休眠模式, WDT 禁止, 无负载	—	—	1	μA
		5V		—	—	1	μA
VIL1	输入低电平	—	输入口	VSS	—	0.5VDD	V
VIH1	输入高电平	—	输入口	0.5VDD	—	VDD	V
VIL2	输入低电平	—	施密特输入口	VSS	—	0.3VDD	V
VIH2	输入高电平	—	施密特输入口	0.7VDD	—	VDD	V
VBOR1	低电压复位 2.0V	—	—	—	2.0	—	V
VBOR2	低电压复位 2.4V	—	—	—	2.4	—	V
VBOR3	低电压复位 3.6V	—	—	—	3.6	—	V
VLVD0	低电压标志	—	—	—	2.4	—	V
VLVD1	低电压标志	—	—	—	3.6	—	V
IOL1	(DRENxn=01)	5V	VOL=0.1VDD	—	10	—	mA
IOL2	(DRENxn=10)	5V	VOL=0.1VDD	—	25	—	mA
IOL3	(DRENxn=11)	5V	VOL=0.1VDD	—	8	—	mA
IOL4	(DRENxn=00)	5V	VOL=0.1VDD	—	50	—	mA
IOH1	(DRENxn=01)	5V	VOH= 0.9VDD	—	8	—	mA
IOH2	(DRENxn=10)	5V	VOH=0.9VDD	—	10	—	mA
IOH3	(DRENxn=11)	5V	VOH=0.9VDD	—	4	—	mA

I_{OH4}	(DREN _{xn} =00)	5V	VOH= 0.9VDD	—	16	—	mA
R_{PH}	内部上拉电阻	5V	可编程上拉电阻	—	100	—	kΩ
R_{PD}	内部下拉电阻	5V	可编程下拉电阻	—	100	—	kΩ
V_{AD}	ADC 输入电压	—	—	VSS	—	V _{REF}	V
DNL	差分非线性误差	5V	AD 时钟频率 2MHz	—	±1	—	LSB
INL	积分非线性误差	5V	AD 时钟频率 2MHz	—	±1	—	LSB
I_{ADC}	ADC 工作电流	3V	AD 时钟频率 2MHz	—	0.3	—	mA
		5V		—	0.5	—	mA

2.3 交流特性

符号	参数	测试条件		最小值	典型值	最大值	单位
		VDD	条件 (常温 25°C)				
FRCH	高频内部 RC 振荡器	5V	—	—	32	—	MHz
FRCL	低频内部 RC 振荡器	5V	—	—	32	—	KHz
F_{OSH}	外部高频晶振	—	3.0~5.5V	4	—	20	MHz
F_{OSL}	外部低频晶振	—	2.4~5.5V	—	32.768	—	KHz
TVDD	VDD 上升时间	5V	—	—	—	100	ms
TBOR	欠压复位响应时间	5V	—	100	—	—	ns
TWDT	看门狗溢出时间	5V	使用预分频 1:1	—	18	—	ms
			不使用预分频器	—	72	—	ms
TMCLRB	复位脉冲时间	5V	—	200	—	—	us

2.4 电气特性曲线图



3 中央处理器（CPU）

HC18P13xL CPU内核包括：

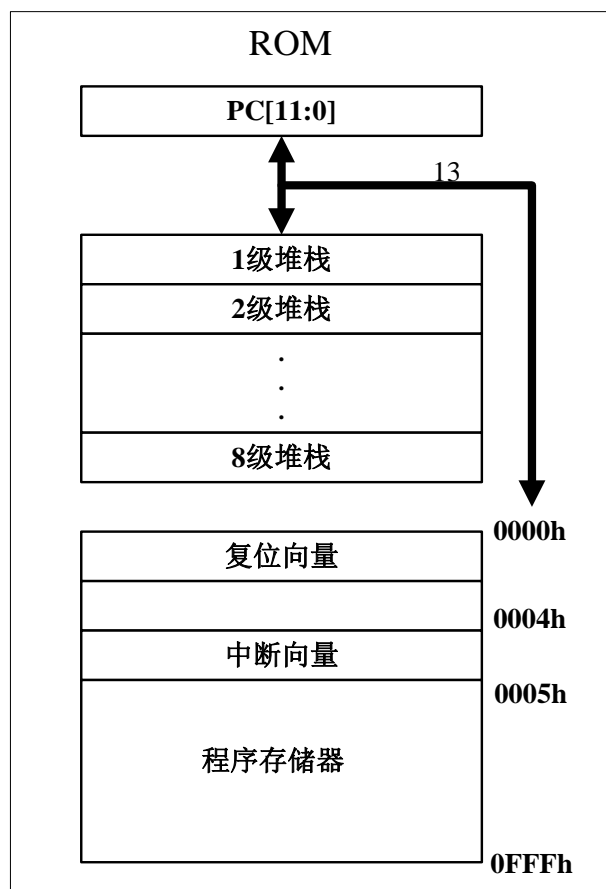
- 2T/4T 时钟模式
- 8级堆栈
- 程序存储器
- 寻址方式
- 数据存储器

3.1 存储器

3.1.1 程序存储器（OTP-ROM）

HC18P13xL具有4K×16位的程序存储器，下图给出了程序存储器的映射。访问超出物理地址以外的单元时，会导致返回到地址最低单元。

复位向量是0000h，中断向量是0004h。



3.1.1.1 复位向量（0000h）

复位向量为0000h

- 上电复位 (POR=0, BOR=X, TO=1)
- 低电压复位 (POR=1, BOR=0, TO=1)
- 看门狗复位 (POR=1, BOR=1, TO=0)
- 外部复位 (POR=1, BOR=1, TO=1)

发生上述任一种复位后，程序将从0000h处重新开始执行，系统寄存器也都将恢复为默认值。根据PCON寄存器中的POR, BOR标志及STATUS 寄存器中的TO标志位的内容可以判断系统复位方式。下面一段程序演示了如何定义ROM中的复位向量。

- 例：定义复位向量

```

                ORG          0000H      ;复位向量
                GOTO        MAIN       ;跳转到用户程序
                ...
                ORG          400H      ;用户程序起始
MAIN:
                ...
                ...
                END              ;用户程序结束
    
```

- 例：复位源判断

```

                ORG          0000H
                GOTO        RST_JUGE
                ...
RST_JUGE:
                BCF         STATUS,RP0 ;Bank0
                BTFSS      PCON,POR
                GOTO        ISPOR      ;POR标志为0, 判定为上电复位
                BTFSS      PCON,BOR
                GOTO        ISBOR      ;POR=1, BOR=0, 判定为低电压复位
                BTFSS      STATUS,TO
                GOTO        ISWDTR     ;POR=1, BOR=1, TO=0, 判定为WDT复位
EXT_RST:
                ...                 ;POR=1, BOR=1, TO=1, 判定为外部复位
                ...
ISPOR:
                BSF         PCON,POR  ;上电复位处理程序
                ...
ISBOR:
                BSF         PCON,BOR  ;低电压复位处理程序
                ...
ISWDTR:
                CLRWDT          ;TO标志置1, WDT复位处理程序
                ...            ;其他程序, 注意处理Bank
    
```

3.1.1.2 中断向量 (0004h)

中断向量地址为0004h。一旦有中断响应，程序计数器PC的当前值就会存入堆栈缓存器并跳转到0004H 开始执行中断服务程序。中断服务子程序中需要对相应状态寄存器进行适当的断点保护和恢复。下面的示例程序说明了如何编写中断服务程序。

➤ 例：中断子程序的编写

Interrupt_Sev:

```

MOVWF    W_TEMP      ;保护W寄存器
SWAPF    STATUS,W
MOVWF    STATUS_TEMP ;保护STATUS寄存器
MOVF     PCLATH,W
MOVWF    PCLATH_TEMP ;保护PCLATH寄存器
BCF      STATUS, RP0

BCF      STATUS,RP0
BTFSC    INTCON,INTF
GOTO     ISR_INT      ;发生INT0中断
BTFSC    INTCON,T0IF
GOTO     ISR_T0       ;发生T0中断
...

```

Exit_Int:

```

BCF      STATUS,RP0
MOVF     PCLATH_TEMP,W
MOVWF    PCLATH      ;恢复PCLATH
SWAPF    STATUS_TEMP,W
MOVWF    STATUS      ;恢复STATUS
SWAPF    W_TEMP,F
SWAPF    W_TEMP,W   ;恢复W
BSF      STATUS,RP0
RETFIE                      ;退出中断

```

对于编写中断服务程序，需要以下几个要点需注意：

1. 中断入口地址为 0x04，响应中断后，程序自动跳转到 0x04 开始执行；
2. 中断服务程序需首先对相应的寄存器进行保护；
3. 中断服务子程序返回前对保护的寄存器进行恢复，注意恢复顺序，对 W 必须使用 SWAPF；
4. 程序中使能两个以上的中断源时，程序需对发生中断的中断源进行判断，从而执行相应的服务程序。
5. 需要软件清零对应的中断标志；
6. RETFIE 指令将自动使能 GIE，请勿在中断服务子程序中用其它指令使能 GIE，以免造成中断响应混乱。

3.1.1.3 查表

方式一：

利用 ADDWF PCL, F 和 RETLW 指令实现数据表，因为以 PCL 为目的操作数的运算将改变程序指针(PC)值，其具体操作为 PC 的低 8 位为 ALU 的运算结果，PC 的高 5 位将从 PC 高位缓冲器 PCLATH 中获得。如下是数据表实现的一个例子。

➤ 例：数据查表

...

```

MOV LW    HIGH TAB1 ;获得数据表地址高8位（内部宏指令）
MOV WF    PCLATH    ;表地址高位赋给PCLATH
MOV F     TABBUF,W  ;获得表数据偏移量，调用前赋值。
CALL     TAB1       ;调用数据表
...

TAB1:
ORG      100H

ADDWF    PCL,F      ;表头运算
RETLW    DATA0_TAB1 ;W=0对应数据
RETLW    DATA1_TAB1 ;W=1对应数据
RETLW    DATA2_TAB1 ;W=2对应数据
...
RETLW    DATAFE_TAB1 ;W=0xFE对应数据

```

对于数据查表的编程，需注意：

1. 数据表宽度：8位；
2. 数据表无法直接跨页访问，单页可实现最大长度：255；
3. 当 PCL 与 W 的加运算有进位时，进位将被舍弃数据表溢出，将造成查表混乱；故表头尽量放在数据页前端，以免数据表溢出；
4. TABBUF 的值不得大于表长，否则将造成运行混乱。

➤ **例：跳转表**

跳转表能够实现多地址跳转功能。由于 PCL 和 W 的值相加即可得到新的 PCL，同时 PCH 从 PCLATH 中载入，因此，可以通过对 PCL 加上不同的 W 值来实现多地址跳转，可参考以下范例。

```

...
ORG      0100H
MOV LW    HIGH TAB2 ;获得跳转表地址高位（内部宏指令）
MOV WF    PCLATH
MOV F     TABBUF,W
TAB2:
ADDWF    PCL,F
GOTO     LABEL0_TAB2 ;TABBUF=0,跳转 LABEL0_TAB2
GOTO     LABEL1_TAB2 ;以下类推
GOTO     LABEL2_TAB2
GOTO     LABEL3_TAB2

```

注：

如上跳转表，有 4 个跳转分支，TABBUF 的合法范围为 0x00~0x03。

方式二：

可以通过以下5个特殊功能寄存器对ROM区中的数据进行查找。

- PMCON
- PMDATL
- PMDATH
- PMADRL

PMADRH

寄存器 PMADRH 指向 ROM 区数据地址的高字节 (bit8~bit15)，寄存器 PMADRL 指向 ROM 区数据地址的低字节 (bit0~bit7)。将 PMCON 寄存器的 RDON 位置 1 启动读操作，使用两条指令来读数据，RDON 位置 1 后的二条指令被自动忽略，建议用户 RDON 位置 1 后的两条指令为 NOP。执行完读操作后，所查找的数据保存在 PMDATH:PMDATL 寄存器。

➤ 例：查找ROM 地址为“TABLE”的值

```

BCF          STATUS,RP0      ;BANK0
MOVF        TABLE _ADDR_H, W
MOVWF       PMADRH          ;设置TABLE地址高字节
MOVF        TABLE _ADDR_L, W
MOVWF       PMADRL          ;设置TABLE地址低字节
BSF         PMCON, RDON     ;开始读
NOP
NOP                               ;等待两条指令
MOVF        PMDATL, W
MOVWF       TABLE _DATAL ;TABLE _DATAL= TABLE地址数据低字节
MOVF        PMDATH, W
MOVWF       TABLE _DATAH ;TABLE _DATAH= TABLE地址数据高字节
...
...
TABLE:      DW          1234H      ;定义数据表（16 位）数据。
            DW          F178H
            DW          2123H
    
```

9Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMCON	-	-	-	-	-	-	-	RDON
R/W	-	-	-	-	-	-	-	R/W
POR的值	-	-	-	-	-	-	-	0

bit 0 RDON: 读控制位

0 =不启动ROM存储器读操作

1 =启动ROM读操作（由硬件清零RDON；软件只能将RDON位置1，但不能清零）

9Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMDATL	PMD7	PMD6	PMD5	PMD4	PMD3	PMD2	PMD1	PMD0
R/W	R/W	R/W	- R/W	R/W	R/W	R/W	R/W	R/W
POR的值	x	x	x	x	x	x	x	x

9Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMDATH	PMD15	PMD14	PMD13	PMD12	PMD11	PMD10	PMD9	PMD8
R/W	- R/W	- R/W	- R/W	R/W	R/W	R/W	R/W	R/W
POR的值	x	x	x	x	x	x	x	x

PMDx[15:0]: ROM存储器读操作后， PMADRH:PMADRL 指向地址的数据

9Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMADRL	PMA7	PMA6	PMA5	PMA4	PMA3	PMA2	PMA1	PMA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	x	x	x	x	x	x	x	x

9Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMADRH	PMA15	PMA14	PMA13	PMA12	PMA11	PMA10	PMA9	PMA8
R/W	- R/W	- R/W	- R/W	R/W	R/W	R/W	R/W	R/W
POR的值	x	x	x	x	x	x	x	x

PMAx[15:0]: ROM存储器地址

3.1.2 芯片配置选择表

HC18P13xL提供的配置字可以对多个系统模块做配置选择，详细配置选择见下面表格。系统还提供了一个4×16bit的器件ID供用户存储校验或其他代码标识号。在程序运行过程中不能访问这些存储单元，但可在编程烧录/校验时对它们进行读写。

编译选项	内容	功能说明
高频系统时钟选择(OSCHM)	高频晶体振荡器	高频晶体振荡器 OSCI/OSCO 作为高频晶体振荡器输入/输出口。
	内部 RC 振荡器	内部 16M RC 振荡器。
低频系统时钟选择(OSCLM)	32K WDT 振荡器	内部 32K WDT 振荡器。
	定时器 1 振荡器	低频晶体振荡器，32.768KHz，LOSCI/LOSCO 为低频晶体振荡器输入/输出口。
高频内部 RC 振荡器频率选择 (ROSC)	16MHz	内部高频 RC 振荡器频率配置为 16MHz。
	8MHz	内部高频 RC 振荡器频率配置为 8MHz。
	4MHz	内部高频 RC 振荡器频率配置为 4MHz。
	2MHz	内部高频 RC 振荡器频率配置为 2MHz。
	1MHz	内部高频 RC 振荡器频率配置为 1MHz。
	500KHz	内部高频 RC 振荡器频率配置为 500KHz。
WDT 功能使能 (WDTEN)	使能 WDT 功能	使能看门狗定时器功能。
	屏蔽 WDT 功能	屏蔽看门狗定时器功能。
外部复位使能 (MCLREN)	使能外部复位	使能外部复位引脚。
	屏蔽，做输入	屏蔽外部复位引脚，外部复位引脚做输入功能。
加密使能位 (CP0)	加密	使能用户程序区 CODE 加密功能。
	不加密	屏蔽用户程序区 CODE 加密功能。
启动时钟选择 (SPDS)	高频系统时钟	系统选择高频系统时钟作为启动时钟。
	低频系统时钟	系统选择低频系统时钟作为启动时钟。
ADC 使能	打开 ADC 功能	使能 ADC 功能。
	禁止 ADC 功能	禁止 ADC 功能。
PORSEL 选择	18ms	上电复位时间选择为 18ms。
	4.5ms	上电复位时间选择为 4.5ms。
SMTENB	IO 口施密特使能	使能 IO 口施密特功能。
	IO 口施密特禁止	禁止 IO 口施密特功能。
BORSEL	BOR3.6V	当系统电压低于 3.6V 时，系统复位。
	BOR2.4V	当系统电压低于 2.4V 时，系统复位。
	BOR2.0V	当系统电压低于 2.0V 时，系统复位。
时钟模式选择 (FCPUT)	4T	4T 模式，1 个指令周期有 4 个系统时钟周期组成。
	2T	2T 模式，1 个指令周期有 2 个系统时钟周期组成。

选择芯片配置字注意事项:

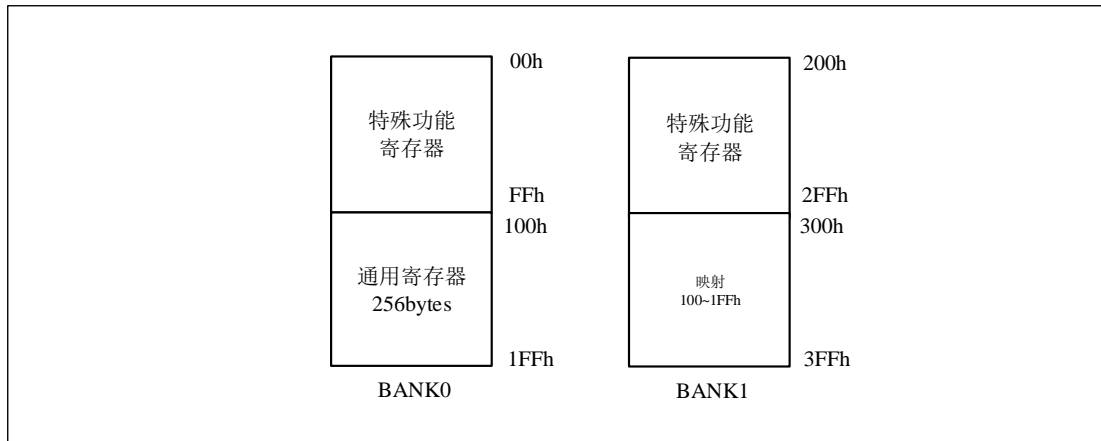
1. 在系统允许的情况下，尽量选用较低系统时钟频率，有利于降低系统功耗和提升系统电磁兼容性；
2. 时钟模式选择为 2T 时，PWM 模块的最大分辨率降低到 9 位；
3. 强干扰情况下，建议开启 WDT 功能。

3.1.3 通用数据寄存器 (RAM)

HC18P13xL共有256个通用寄存器 (GPR) 和104个特殊功能寄存器 (SFR), 分在2个存储区Bank0和Bank1, RP0是存储区的选择位。

CORE Register 00-09h&200-209h

00h&200h	INDF0	0 ^o	INDF0 ^o	间接寻址 0 寄存器 (不是实际存在的物理寄存器) ^o							
01h&201h	INDF1	1 ^o	INDF1 ^o	间接寻址 1 寄存器 (不是实际存在的物理寄存器) ^o							
02h&202h	PCL	2 ^o	PCL ^o	程序计数器 (PC) 低字节 ^o							
03h&203h	STATUS	3 ^o	STATUS ^o	^o	^o	RP0 ^o	TO ^o	PD ^o	Z ^o	DC ^o	C ^o
04h&204h	FSR0L	4 ^o	FSR0L ^o	间接寻址 0 地址低位指针 ^o							
05h&205h	FSR0H	5 ^o	FSR0H ^o	间接寻址 0 地址高位指针 ^o							
06h&206h	FSR1L	6 ^o	FSR1L ^o	间接寻址 1 地址低位指针 ^o							
07h&207h	FSR1H	7 ^o	FSR1H ^o	间接寻址 1 地址高位指针 ^o							
08h&208h	PCLATH	8 ^o	PCLATH ^o	^o	^o	^o	程序计数器高 5 位 ^o				
09h&209h	INTCON	9 ^o	INTCON ^o	GIE ^o	PEIE ^o	TOIE ^o	INTE ^o	RBIE ^o	TOIF ^o	INTF ^o	RBIF ^o



3.1.4 特殊功能寄存器 (SFR)

3.1.4.1 特殊功能寄存器列表

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位初值
BANK0										
010h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111
011h	TRISB	TRISB7	TRISB6	-	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111
012h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111
015h	TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	1111 1111
01Ch	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	0000 0000
01Dh	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	0000 0000
01Eh	PORTC	-	-	-	-	-	-	PORTC1	PORTC0	0000 0000
021h	PORTF	PORTF7	PORTF6	PORTF5	PORTF4	PORTF3	PORTF2	PORTF1	PORTF0	0000 0000
028h	WPUA	WPUA7	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	1111 1111
029h	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	1111 1111
02Ah	WPUC	-	-	-	-	-	-	WPUC1	WPUC0	1111 1111

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位初值
02Dh	WPUF	WPUF7	WPUF6	WPUF5	WPUF4	WPUF3	WPUF2	WPUF1	WPUF0	1111 1111
034h	WPDA	WPDA7	WPDA6	WPDA5	WPDA4	WPDA3	WPDA2	WPDA1	WPDA0	1111 1111
035h	WPDB	WPDB7	WPDB6	WPDB5	WPDB4	WPDB3	WPDB2	WPDB1	WPDB0	1111 1111
036h	WPDC	-	-	-	-	-	-	WPDC1	WPDC0	1111 1111
039h	WPDF	WPDF	WPDF7	WPDF6	WPDF5	WPDF4	WPDF3	WPDF2	WPDF1	1111 1111
040h	IOCA	IOCA7	IOCA6	IOCA5	IOCA4	IOCA3	IOCA2	IOCA1	IOCA0	0000 0000
041h	IOCB	IOCB7	IOCB6	IOCB5	IOCB4	IOCB3	IOCB2	IOCB1	IOCB0	0000 0000
042h	IOCC	-	-	-	-	-	-	IOCC1	IOCC0	0000 0000
045h	IOCF	IOCF7	IOCF6	IOCF5	IOCF4	IOCF3	IOCF2	IOCF1	IOCF0	0000 0000
04Ch	PORCTR	-	-	-	-	SPPCT1	SPPCT0	UAPCT1	UAPCT0	---- 0000
04Dh	DRENAL	DRENA7L	DRENA6L	DRENA5L	DRENA4L	DRENA3L	DRENA2L	DRENA1L	DRENA0L	1111 1111
04Eh	DRENB	DRENB7L	DRENB6L	DRENB5L	DRENB4L	DRENB3L	DRENB2L	DRENB1L	DRENB0L	1111 1111
04Fh	DRENCL	-	-	-	-	-	-	DRENC1L	DRENC0L	---- --11
052h	DRENFL	DRENF7L	DRENF6L	DRENF5L	DRENF4L	DRENF3L	DRENF2L	DRENF1L	DRENF0L	0000 0000
054h	PIR1	-	ADIF	-	-	-	CCP1IF	T2IF	T1IF	-0- -000
055h	PIR2	-	-	PWM0IF	-	RXIF	TXIF	-	CCP2IF	--0- 00-0
056h	PIR3	-	-	-	RFIF	-	-	RCIF	RAIF	---0 0000
058h	T1L	Timer1 计数寄存器低字节								xxxx xxxx
059h	T1H	Timer1 计数寄存器高字节								xxxx xxxx
05Ah	T1CON	T1CS1	T1CS0	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	-	T1ON	0000 00-0
05Bh	T0	Timer0 计数寄存器								xxxx xxxx
05Ch	T2	Timer2 计数寄存器								xxxx xxxx
05Dh	PR2	Timer 周期寄存器								0000 0000
05Eh	T2CON	-	T2CKPS3	T2CKPS2	T2CKPS1	T2CKPS0	T2ON	-	-	-000 00--
05Fh	PR1L	Timer1 周期寄存器低字节								xxxx xxxx
060h	PR1CON	PWM1T1	PWM1T0	PWM2T1	PWM2T0	T1CKPS3	T1CKPS2	PWMPRI	PRIEN	0000 0000
070h	PIE1	-	ADIE	-	-	-	CCP1IE	T2IE	T1IE	0000 0000
071h	PIE2	-	-	PWM0IE	-	-	UARTIE	-	CCP2IE	--00 -000
072h	PIE3	-	-	-	RFIE	-	-	RCIE	RAIE	---0 0000
078h	OPTION	RBPUB	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	0000 0000
079h	PCON	LVD2EN	LVD1EN	-	WDTENS	LVD2F	LVD1F	POR	BOR	00-1 qqqq
07Ah	OSCCON	T0OSCEN	-	-	-	-	-	HXEN	SCS	0--- --0q
080h	CCPR2L	CCP2 寄存器低字节								xxxx xxxx
081h	CCPR2H	CCP2 寄存器高字节								xxxx xxxx
082h	CCP2CON	-	-	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000
083h	CCPR1L	CCP1 寄存器低字节								0000 0000
084h	CCPR1H	CCP1 寄存器高字节								0000 0000
085h	CCP1CON	-	-	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000
08Ch	ANSELL	ANSEL7	ANSEL6	-	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0	1111 1111

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位初值
08Dh	ANSELH	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8	1111 1111
092h	ADRESL	ADC 结果寄存器低字节								0000000
093h	ADRESH	ADC 结果寄存器高字节								00000000
094h	ADCON0	VHS1	VHS0	CHS3	CHS2	CHS1	CHS0	ADON	ADEN	00000000
095h	ADCON1	ADFM	ADCS2	ADCS1	ADCS0	-	-	-	ADREF	0000---0
096h	ADCLK	-	-	-	-	-	ADCLK2	ADCLK1	ADCLK0	---- -000
09Ah	PMDATL	程序存储器读数据寄存器的低字节								0000 0000
09Bh	PMDATH	程序存储器读数据寄存器的高字节								0000 0000
09Ch	PMADRL	程序存储器读地址寄存器的低字节								0000 0000
09Dh	PMADRH	程序存储器读地址寄存器的高字节								0000 0000
09Eh	PMCON	-	-	-	-	-	-	-	RDON	---- ---0
0A0h	DRENAH	DRENA7H	DRENA6H	DRENA5H	DRENA4H	DRENA3H	DRENA2H	DRENA1H	DRENA0H	1111 1111
0A1h	DRENBH	DRENB7H	DRENB6H	DRENB5H	DRENB4H	DRENB3H	DRENB2H	DRENB1H	DRENB0H	1111 1111
0A2h	DRENCH	-	-	-	-	-	-	DRENC1H	DRENC0H	---- --11
0A5h	DRENFH	DRENF7H	DRENF6H	DRENF5H	DRENF4H	DRENF3H	DRENF2H	DRENF1H	DRENF0H	0000 0000
BANK1										
23Ah	BRT	波特率发生器寄存器								0000 0000
23Bh	AUXR	-	UARTEN	UARTM0	BRTR	BRTX12	S1BRS	SMOD	SMOD0	-000 0000
23Ch	SCON	SM0/FE	SM1	SM2	REN	TR8	RB8	RXWK	-	0000 000-
23Dh	SBUF	串行口缓冲寄存器								0000 0000
23Eh	SADEN	从机地址寄存器								0000 -000
23Fh	SADDR	从机地址掩码寄存器								00-0 0000
258h	PWM0DT	DT0.7	DT0.6	DT0.5	DT0.4	DT0.3	DT0.2	DT0.1	DT0.0	0000 0000
259h	PWM0DL	PD0.7	PD0.6	PD0.5	PD0.4	PD0.3	PD0.2	PD0.1	PD0.0	0000 0000
25Ah	PWM0DH	-	-	-	-	PD0.11	PD0.10	PD0.9	PD0.8	0000 0000
25Bh	PWM0PL	PP0.7	PP0.6	PP0.5	PP0.4	PP0.3	PP0.2	PP0.1	PP0.0	0000 0000
25Ch	PWM0PH	-	-	-	-	PP0.11	PP0.10	PP0.9	PP0.8	0000 0000
25Dh	PWM0C	-	-	FLTS	FLTC	PWM0S1	PWM0S0	CK01	CK00	0000 0000
25Eh	PWMEN	-	EFLT	-	-	EPWM01	-	-	EPWM0	0000 0000
25Fh	FLTM	-	-	-	-	-	-	FLTOM1	FLTOM0	--00 0000
2B0h	LCDCON	LCDEN	RLCD1	RLCD0	FRAME	-	-	-	-	0000 ----
2B1h	COMAEN	COMAEN7	COMAEN6	COMAEN5	COMAEN4	COMAEN3	COMAEN2	COMAEN1	COMAEN0	0000 0000
2B2h	COMBEN	COMBEN7	COMBEN6	-	COMBEN4	COMBEN3	COMBEN2	COMBEN1	COMBEN0	00-0 0000
2B3h	COMCEN	-	-	-	-	-	-	COMCEN1	COMCEN0	---- --00
2B6h	COMFEN	COMFEN7	COMFEN6	COMFEN5	COMFEN4	COMFEN3	COMFEN2	COMFEN1	COMFEN0	0000 0000

注： x = 未知， u = 不变， q = 取值视条件而定， - = 未实现

3.1.4.2 累加器

8 位数据寄存器W用来执行ALU与数据存储器之间数据的传送操作。如果操作结果为零（Z）或有进位产生（C或DC），程序状态寄存器STATUS中相应位会发生变化。

W 并不在RAM中，因此不可以用直接寻址和间接寻址模式对其进行读写。

➤ 例：W寄存器的读写操作

立即数写入W寄存器操作：

```

MOV LW    H'0FF'    ;送十六进制数
MOV LW    D'10'     ;送十进制数
MOV LW    B'11110000' ;送二进制数
    
```

将W寄存器的数据写入数据寄存器BUF中：

```
MOVWF    BUF
```

将数据寄存器BUF中的数读入W寄存器：

```
MOVF    BUF,W
```

将W寄存器的数据与BUF中的数据加法运算后，结果存入BUF中：

```
ADDWF    BUF,F
```

3.1.4.3 INDFx寄存器

INDF0、INDF1 寄存器不是实际存在的寄存器，寻址 INDF0、INDF1 将实现间接寻址。

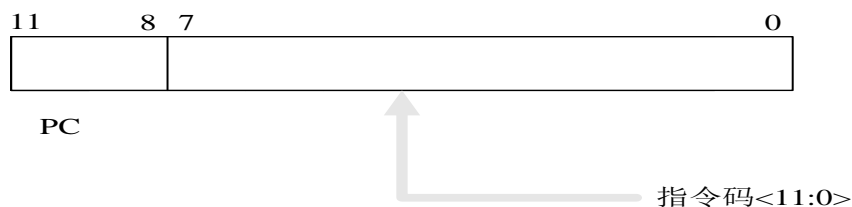
3.1.4.4 程序计数器

程序计数器（PC）为13位宽，低字节来自可读写的PCL寄存器，高字节（PC[12:8]）不可读写，可通过PCLATH 寄存器间接写入。

02h&202h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCL	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

08h&208h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCLATH	-	-	-	-	PCH11	PCH10	PCH9	PCH8
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR的值	-	-	-	-	0	0	0	0

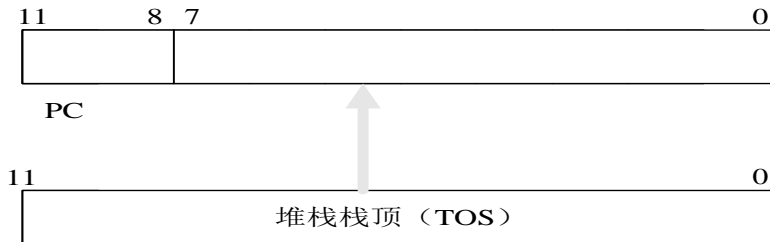
程序计数器顺序指令运行时，每个指令周期程序自动加1，以下三种情况将使程序计数器重新装载。分支指令（GOTO/CALL）：



以PCL作为目的的操作数的指令：



子程序返回指令 (RETURN/RETLW/RETFIE):



3.1.4.5 STATUS寄存器

STATUS寄存器包含ALU的算术状态、复位状态和寄存器的存储区选择位。

03&203h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	-	-	RP0	TO	PD	Z	DC	C
R/W	-	-	R/W	R	R	R/W	R/W	R/W
POR的值	-	-	0	1	1	x	x	x

bit 5 **RP0**: BANK选择位

- 1 = Bank1
- 0 = Bank0

bit 4 **TO**: 超时位

- 1 = 上电、执行了CLRWDWT指令或SLEEP指令
- 0 = 发生了WDT溢出

bit 3 **PD**: 掉电位

- 1 = 上电或执行了CLRWDWT指令
- 0 = 执行了SLEEP指令

bit 2 **Z**: 结果为零位

- 1 = 算术或逻辑运算的结果为零
- 0 = 算术或逻辑运算的结果不为零

bit 1 **DC**: 半进位/借位位

- 1 = 加法运算时低四位有进位/减法运算时没有向高四位借位
- 0 = 加法运算时低四位没有进位/减法运算时有向高四位借位

bit 0 **C**: 进位/借位位

- 1 = 加法运算时有进位/减法运算时没有借位发生/移位后移出逻辑1
- 0 = 加法运算时没有进位/减法运算时有借位发生/移位后移出逻辑0

3.1.4.6 PCON寄存器

079h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCON	LVD2EN	LVD1EN	-	WDTENS	LVD2F	LVD1F	POR	BOR
R/W	R/W	R/W	-	R/W	R	R	R/W	R/W
POR的值	0	0	-	1	q	q	q	q

注：- = 未实现，q = 取值视条件而定

bit 7 **LVD2EN**:3.6V低电压检测使能位

1 = 使能3.6V低电压检测

0 = 禁止3.6V低电压检测

bit 6 **LVD1EN**:2.4V低电压检测使能位

1 = 使能2.4V低电压检测

0 = 禁止2.4V低电压检测

bit 4 **WDTENS**:看门狗软件使能位（需要配置字中使能WDT，该位使能时才有效）

1 = 软件使能看门狗

0 = 软件禁止看门狗

bit 3 **LVD2F**:3.6V低电压检测标志位

1 = 电压低于3.6V

0 = 电压高于3.6V

bit 2 **LVD1F**:2.4V低电压检测标志位

1 = 电压低于2.4V

0 = 电压高于2.4V

bit 1 **POR**:上电复位状态位

1 = 非上电复位

0 = 上电复位（需要软件置1）

bit 0 **BOR**:欠压复位状态位

1 = 未发生欠压复位

0 = 发生了欠压复位（需要软件置1）

注：

- 1、在芯片执行 SLEEP 指令前，软件关闭看门狗定时器，可以节省休眠或绿色模式下芯片功耗；
- 2、LVD2EN 和 LVD1EN 检测电平值仅作为设计参考，不能用作芯片工作电压值得精确检测。

3.2 寻址模式

HC18P13xL 共有三种寻址方式：立即寻址、直接寻址和间接寻址模式

3.2.1 立即寻址

立即数参与运算的寻址方式

➤ 例：立即寻址

ADDLW 06h ; W 的内容加 6，结果放入 W

3.2.2 直接寻址

寄存器参与运算的寻址方式

➤ 例：直接寻址

```
MOVWF    OPTION    ; W 的内容装入 OPTION
```

3.2.3 间接寻址

由指针 FSR（FSR_{xL}/FSR_{xH}）指向的寄存器参与运算的寻址方式。INDF_x 寄存器不是物理寄存器，对 INDF_x 寄存器操作可以实现间接寻址

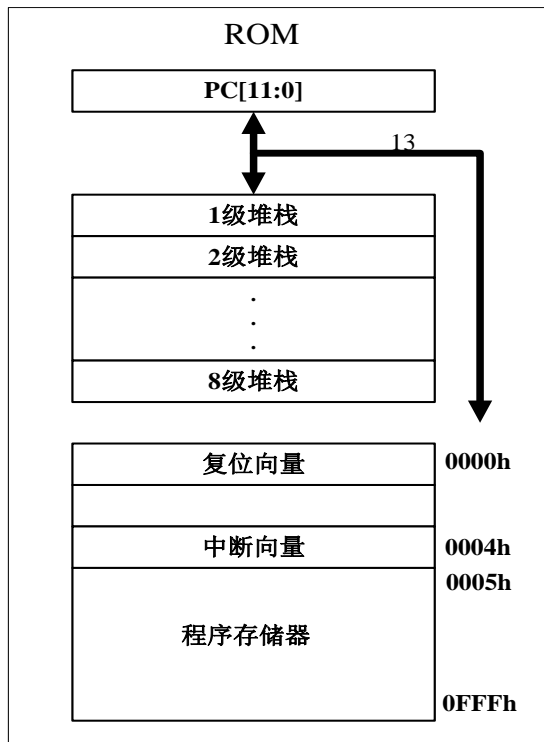
➤ 例：利用间接寻址对 100h~1FFh，300h~3FFh 通用数据存储器进行清零

```

MOV LW    00h           ;清零 0x100~0x1FF
MOV WF    FSR0L
MOV LW    0x01
MOV WF    FSR0H        ;FSR 指向 100h 地址
NEXTBYTE: CLRF         INDF0      ;对 FSR 指向的数据存储器清零
           INCFSZ      FSR,F      ;FSR0L+1, 指向下一个地址
           ;注意这里的边界值为欲操作 RAM 最大地址+ 1
           ;利用间接寻址，注意意外指向特殊寄存器的情况
           GOTO       NEXTBYTE   ;FSR0L 的值加一不溢出，循环清零下一个地址。
300h~3FFh 通用数据存储器进行清零程序可参照上述程序
    
```

3.3 堆栈

HC18P13xL 具有一个 8 级深度的硬件堆栈。当执行 CALL 指令或由于中断导致程序跳转时，PC 值会被压入堆栈；当执行 RETURN、RETLW 或 RETFIE 指令时，PC 值从堆栈弹出。



注：

压栈级数请勿超过 8 级，超过 8 级压栈将导致堆栈溢出，溢出后堆栈指针循环，新的压栈将覆盖原堆栈内容。

4 复位

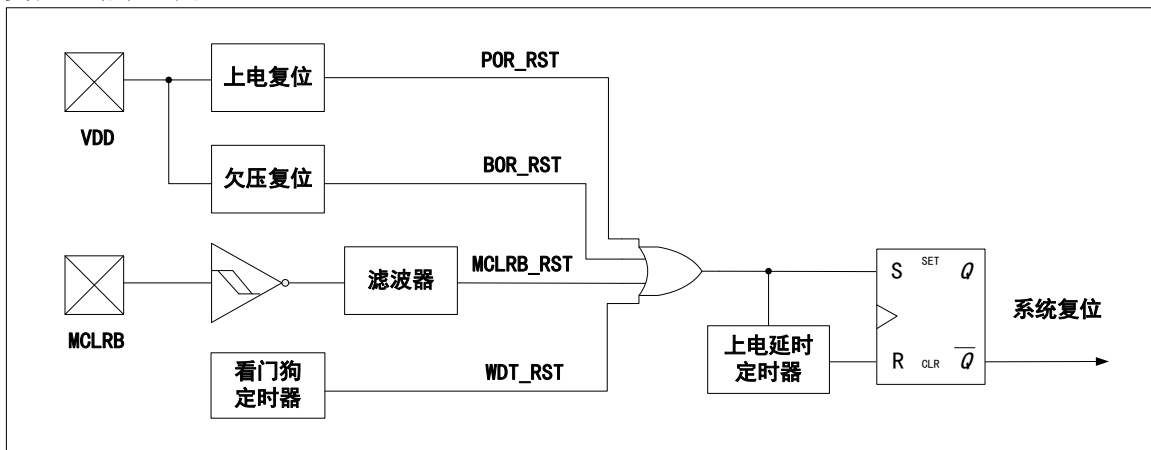
4.1 概述

HC18P13xL 共有四种复位方式：

- 上电复位 (POR)
- 外部复位 (MCLR Reset)
- 欠压复位 (BOR)
- 看门狗定时器复位 (WDT Reset)

当上述任何一种复位产生时，系统进入复位状态，所有的特殊功能寄存器被初始化，程序停止运行，同时程序计数器 (PC) 清零。经过上电延时定时器延时后，系统结束复位状态，程序从 0000h 地址开始执行。STATUS 寄存器的 bit4 (TO 位) 及 PCON 寄存器的 bit0 (BOR 位)、bit1 (POR 位) 显示系统复位状态信息，可通过 3 个标志位判断复位来源，从而控制系统的运行路径。

复位电路示意图



特殊功能寄存器复位状态

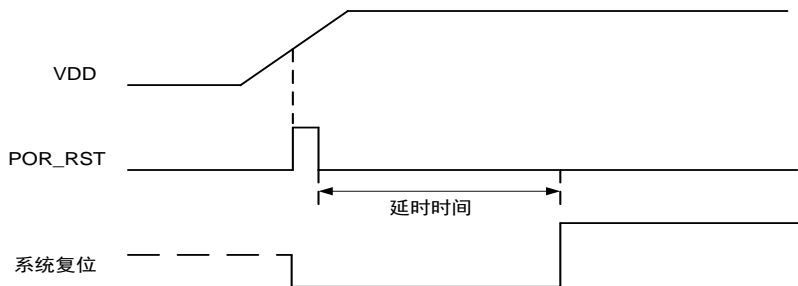
TO	POR	BOR	复位方式	说明
1	0	x	上电复位	电源上电。
u	u	0	欠压复位	电源电压低于BOR电压点。
u	u	u	外部复位	外部复位管脚低电平。
0	u	u	看门狗定时器复位	运行模式下，看门狗定时器溢出。

复位方式	STATUS寄存器	PCON寄存器
上电复位	0001 1xxx	00-1 qq00
正常工作模式下的外部复位	0001 1xxx	00-1 qq0u
休眠模式下的外部复位	0001 0uuu	00-1 qquu
欠压复位	0001 0uuu	00-1 qq00
看门狗定时器复位	0000 1uuu	00-1 qquu

注：u = 不变，x = 未知，- = 未使用，q = 取值视条件而定

4.2 上电复位

系统上电过程中，VDD 达到系统正常工作电压之前，上电复位电路产生内部复位信号，可通过查询 PCON 寄存器来判断是否发生上电复位。VDD 最大上升时间 T_{VDD} 必须满足规格要求。任何一种复位方式都需要一定的响应时间，系统提供完善的复位流程以保证复位动作的顺利进行。对于不同类型的振荡器，完成复位所需要的时间也不同。因此，VDD 的上升速度和不同晶振的起振时间都不固定。RC 振荡器的起振时间最短，晶体振荡器的起振时间则较长。在用户的使用过程中，应考虑系统对上电复位时间的要求。



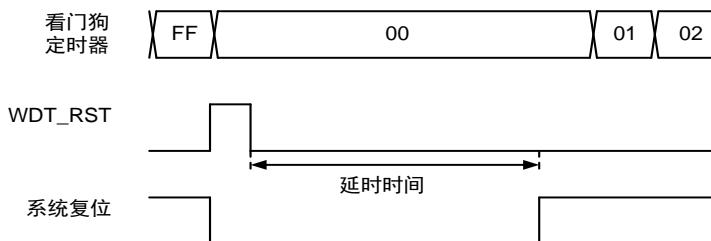
注：关于上电复位，请注意以下几点：

- 1、VDD 上电必须从 0V 开始，若 VDD 有残留电压，POR_RST 信号无法稳定产生；
- 2、VDD 上电斜率必须满足大于 500mV/ms，否则 POR_RST 信号可能无法产生；
- 3、上电复位延时的复位延时时间在配置字里选择，两个档位（18ms/4.5ms）。

4.3 看门狗定时器复位

在高频和低频模式下，看门狗定时器溢出会产生WDT复位；在绿色和休眠模式下，看门狗定时器溢出将唤醒SLEEP并使其返回高频或低频模式，程序从SLEEP指令下一条开始执行。WDT定时器配置字和WDTENS都为1时，才能使能看门狗定时器。

看门狗复位示意图：



注：关于看门狗复位使用时，请注意以下几点：

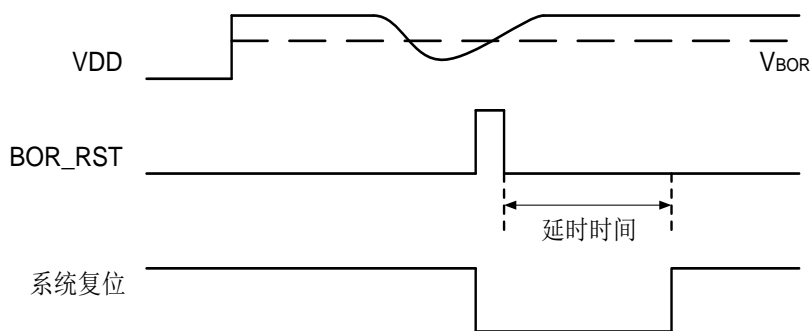
- 1、看门狗的使能逻辑：看门狗使能 = 看门狗配置字使能 & 看门狗软件使能（WDTENS=1）；
- 2、不能在中断程序中对看门狗进行清零，否则无法监控主程序跑飞情况；
- 3、程序中应该只在主程序中有一次清看门狗的动作，这种架构能够最大限度的发挥看门狗的保护功能；
- 4、看门狗复位的延时时间为 2.2ms/1.1ms；
- 5、使用时注意：不论哪种方式复位后，看门狗软件使能位（WDTENS）的值为 1。

4.4 欠压复位

4.4.1 欠压复位的产生

当VDD电压下降到 V_{BOR} 以下，且持续时间满足，系统产生欠压复位。

欠压复位示意图：



低电压复位（BOR）是单片机内置的掉电复位保护装置，当VDD 跌落并低于BOR 检测电压值时，BOR被触发，系统复位。不同的单片机有不同的BOR 检测电平。因此采用BOR 依赖于系统要求和环境状况。如果电源跌落剧烈，远低于BOR 触发点，BOR 能够起到保护作用，让系统正常复位；如果电源电压跌落不是很剧烈，仅仅是接近BOR 触发点而造成的系统错误，则BOR 就不能起到保护作用让系统复位。

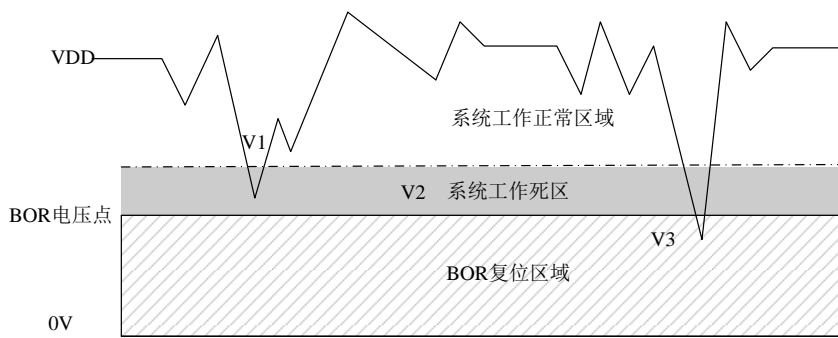
为避免电源较大的抖动，HC18P13xL采取必要的电源抖动处理电路或其他保护电路，防止电源抖动超过1.0V，导致芯片工作异常。

HC18P13xL通过配置字BOR编译选项控制选择低电压检测档位，请客户在使用时根据情况选择合适的BOR电压。

BOR档位：BOR3.6V/2.4V/2.0V

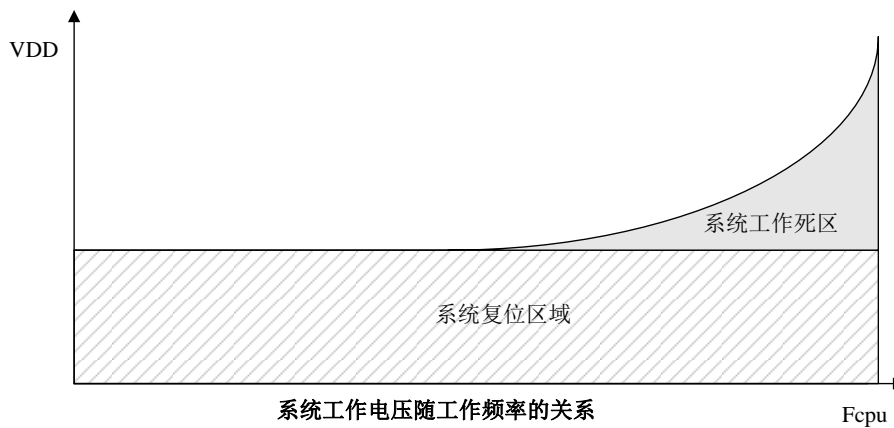
4.4.2 工作死区

电压跌落可能会进入系统死区。系统死区意味着电源不能满足系统的最小工作电压要求。下图是一个典型的掉电复位示意图。图中，VDD受到严重的干扰，电压值降得非常低。虚线以上区域系统正常工作，在虚线以下的区域内，系统进入未知的工作状态，这个区域称作死区。当VDD跌至 V_1 时，系统仍处于正常状态；当VDD跌至 V_2 时，系统进入死区，系统工作在死区时，可能导致程序的运行紊乱；当电压跌至 V_3 ，且低于BOR电压点，系统可正常复位，处于BOR电压点的时间过短，系统仍无法正常产生欠压复位信号，可能导致程序的运行紊乱。



4.4.3 工作死区与工作频率的关系

工作死区电压与工作速度相关，如下图示意了死区与工作频率的关系：



4.4.4 死区防护

对于死区防护，有以下几点建议：

- 合理使用看门狗复位电路
- 降低系统的工作频率
- 合理采用外部复位电路（电压偏移复位电路、外部 IC 复位）

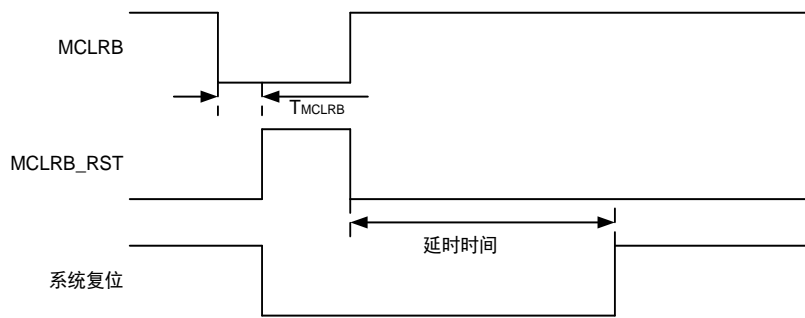
注：

二极管 RC 复位电路电压偏移复位电路、外部 IC 复位防止系统进入死区。

4.5 外部复位

当外部复位端口 MCLR_B 输入一个持续时间超过 T_{MCLR_B} 的低电平时，产生外部复位。MCLR_B 选择配置字（编译选项）为 1，MCLR_B 口为外部复位输入口。

外部复位示意图：



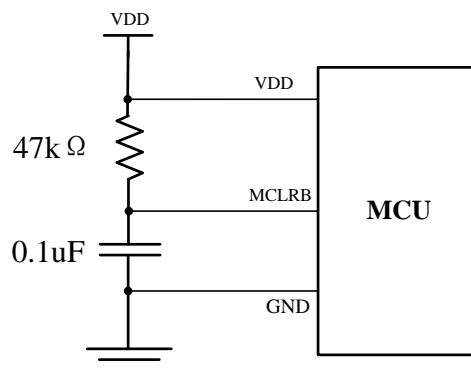
注:

T_{MCLR} 需大于 $200\mu s$ (典型值); 外部复位延时时间为 $2.2ms/1.1ms$ 。

4.5.1 外部 RC 复位电路

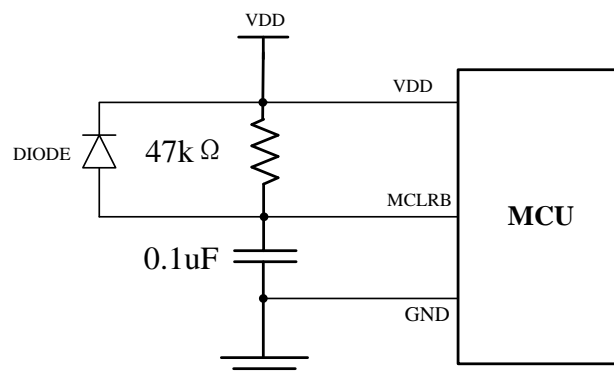
由电阻和电容组成的基本RC复位电路，它在系统上电的过程中能够为复位引脚提供一个缓慢上升的复位信号。这个复位信号的上升速度低于VDD的上电速度，为系统提供合理的复位时序，当复位引脚检测到高电平时，系统复位结束，进入正常工作状态。

如下图:



4.5.2 二极管 RC 复位电路

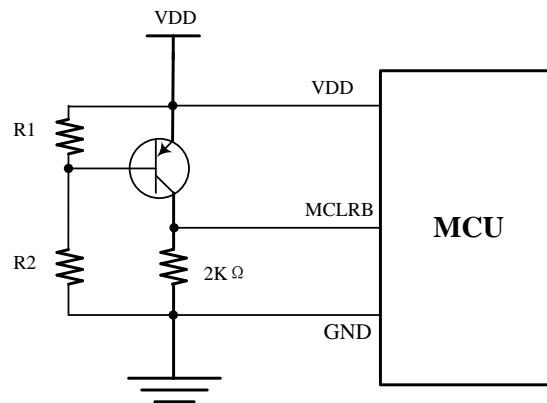
在基本RC复位电路上增加一个二极管(DIODE)，对于电源异常情况，二极管正向导通使电容快速放电并与VDD保持一致，避免复位引脚持续高电平，系统无法正常复位。



4.5.3 电压偏置复位电路

电压偏置复位电路是一种简单的电压检测复位电路，调整电压检测点，可以解决系统死区问题。电路中，R1和R2构成分压电路，当R1和R2的分压值高于三极管的开启电压时，三极管集电极输出高电平，单片机正常工作；当R1和R2的分压值低于三极管的开启电压时，集电极输出低电平，MCU复位。

对于不同应用需求，选择适当的分压电阻。分压电阻R1和R2在电路中要耗电，此处的功耗必须计入整个系统的功耗中。



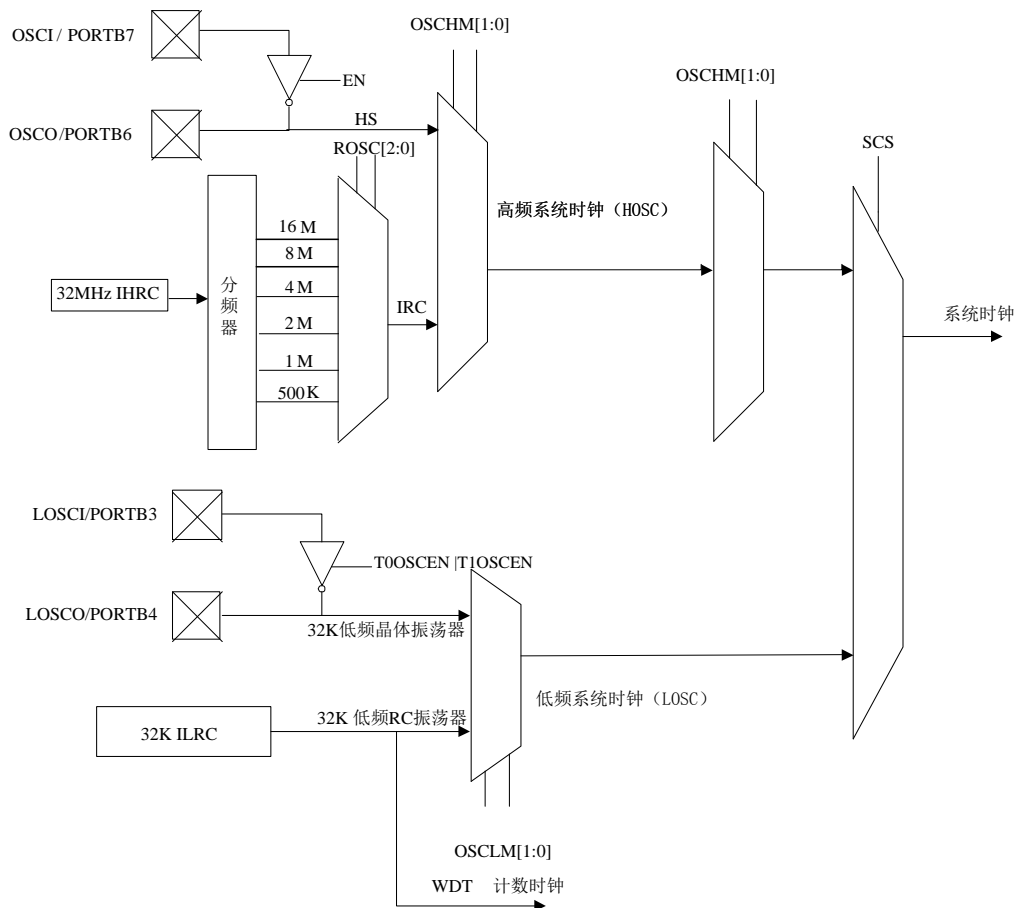
5 系统时钟

5.1 概述

HC18P13xL内带双时钟系统：高频时钟和低频时钟。高频时钟的时钟源由高频晶振或内部32MHz RC振荡电路（IRC 32MHz）提供。低频时钟的时钟源则由低频晶振或内部低速RC振荡电路（RC 32KHz@5V）提供。两种时钟都可作为系统时钟源Fosc。OSCCON寄存器的SCS位控制高频时钟和低频时钟之间切换。

- 高频模式： $F_{cpu} = F_{sys} / N$ ，N = 2或4，时钟模式选择决定N的值。
- 低频模式： $F_{cpu} = F_{sys} / N$ ，N = 2或4，时钟模式选择决定N的值。

5.2 时钟框图



- OSCHM[1:0]: 高速系统时钟选择配置字
- OSCLM[1:0]: 低速系统时钟选择配置字
- ROSC[2:0]: 高速内部RC振荡器频率选择配置字
- Fosc: 时钟源频率
- Fsys: 系统时钟频率

- Fcpu: 指令时钟频率

5.3 系统高频时钟

系统高频时钟有两种选择，通过OSCHM[1:0]高频系统时钟选择配置字来控制。

高频系统时钟选择配置字：

OSCHM[1:0]	说明
00	内部 RC 振荡器（IRC），OSCI/OSCO 作为普通 IO 口。
01	高频晶体振荡器（HS），OSCI/OSCO 作为高频晶体振荡器输入/输出口。 外部时钟输入，OSCI 作为外部时钟输入口，OSCO 作为外部时钟输出口。

5.3.1 内部高频 RC 振荡器

配置字OSCHM[1:0]和ROSC[2:0]控制单片机的内置RC高速时钟。OSCHM[1:0]若选择“00”，则内置RC振荡器作为系统时钟源，OSCI/OSCO作为通用I/O口。

内置RC高频时钟有16M/8M/4M/2M/1M /500K六种选择。

高频内部RC振荡器频率选择配置字

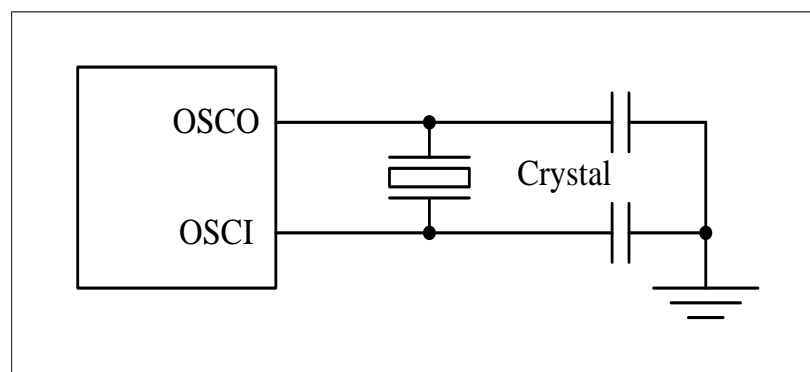
ROSC[2:0]	说明
110	内部RC振荡器频率选择16MHz
101	内部RC振荡器频率选择8MHz
100	内部RC振荡器频率选择4MHz
011	内部RC振荡器频率选择2MHz
010	内部RC振荡器频率选择1MHz
001	内部RC振荡器频率选择500KHz

5.3.2 外部高频时钟

外部高频时钟，由配置字 OSCHM 控制具体模式的选择

- 高频晶体振荡器：最高 20MHz

高频晶体振荡器的频率为1MHz~20MHz，推荐的典型值为4MHz、8MHz和16MHz，电容推荐值为20pF。



注:

OSCI 和 OSCO 引脚与振荡器和起振电容之间距离 10mm 以内。

5.4 系统低频时钟

高频时钟有两种选择，通过低频时钟选择配置字来选择。

- 低频晶体振荡器： 32.768KHz
- 低频 RC 振荡器： 32KHz（5V 典型值）

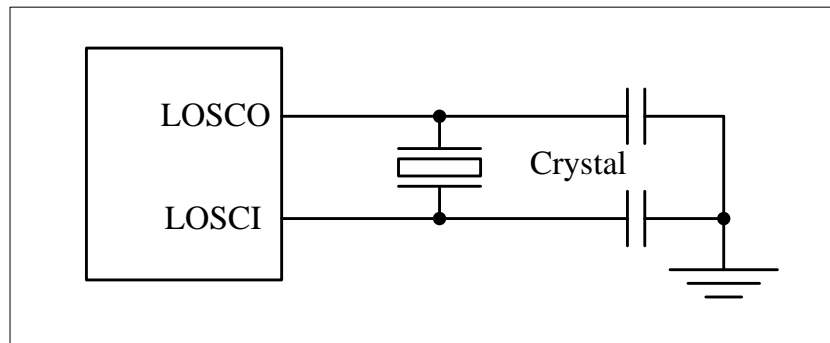
低频系统时钟选择配置字

OSCLM[1:0]	说明
00	低频 RC 振荡器，32KHz，LOSCI/LOSCO 作为输入/输出口
01	低频晶体振荡器，32.768KHz，LOSCI/LOSCO 作为低频晶体振荡器输入/输出口

5.4.1 低频晶体振荡器

低频晶体振荡器的频率为32.768KHz，电容推荐值为20pF。

低频晶体振荡器电路



系统工作在绿色模式下，可以使能低频晶体振荡器。

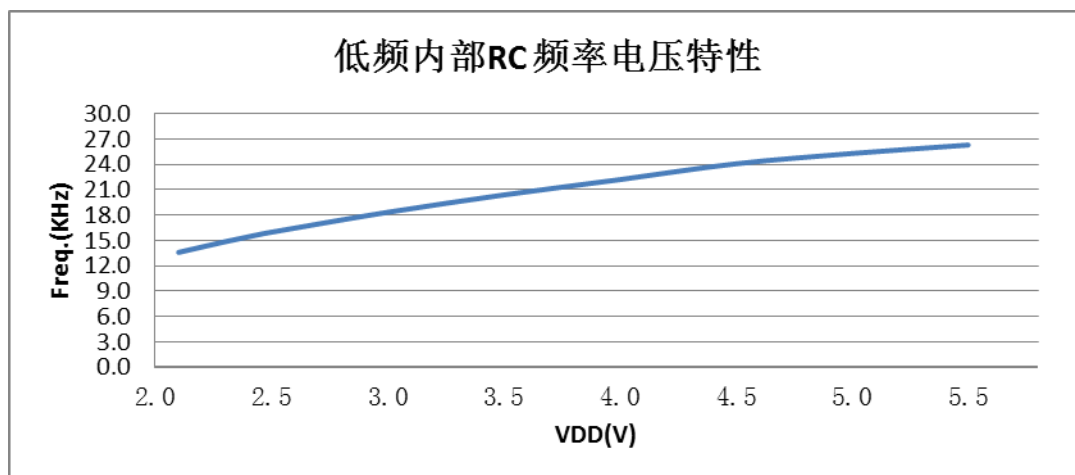
注:

外部高频晶振接 OSCO、OSCI 端口，外部低频晶振接 LOSCO、LOSCI 端口。

5.4.2 低频 RC 振荡器

系统低频时钟源也可采用RC振荡电路。低频内部RC振荡电路的输出频率受系统电压和环境温度的影响较大，通常为5V时输出32KHz（典型值）。

输出频率与工作电压之间的关系如下图所示：



注:

低频时钟也用作看门狗定时器的时钟。

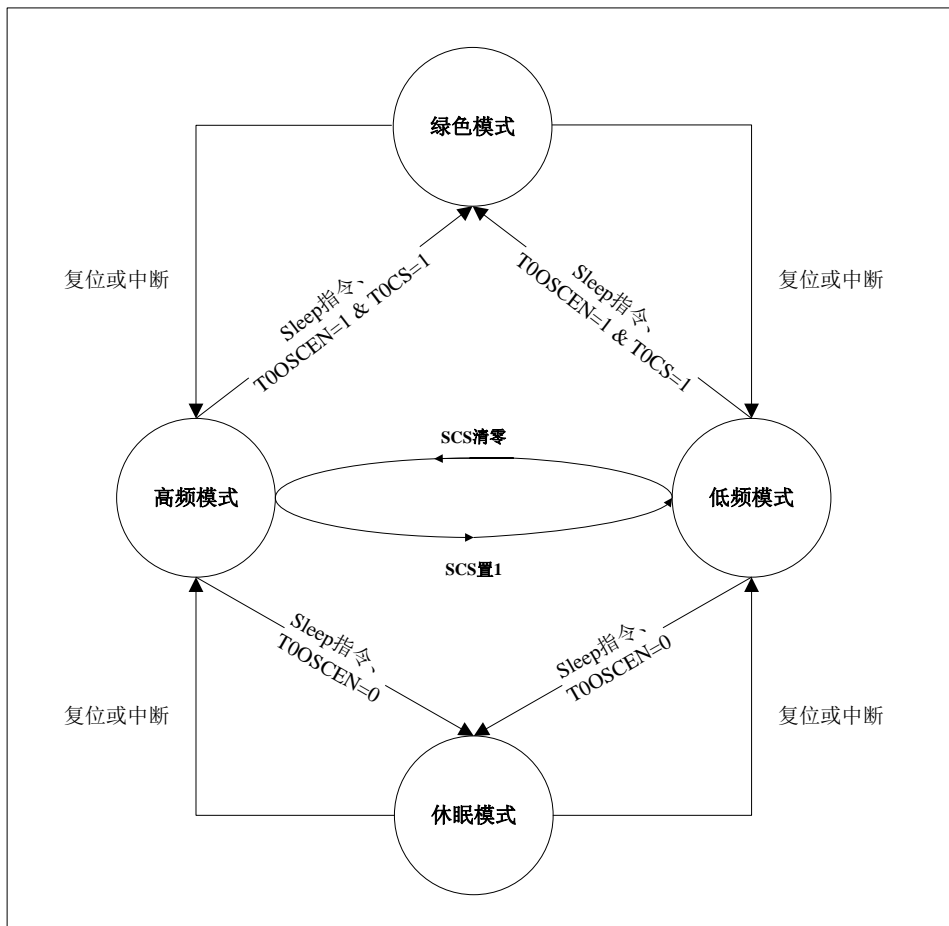
6 系统工作模式

HC18P13xL共有四种工作模式：

- 高频模式
- 低频模式
- 休眠模式
- 绿色模式

系统复位后，工作于高频模式还是低频模式，由系统配置字决定。程序运行过程中，可以通过设置SCS位使系统在高频和低频模式之间切换。

图6-1 系统工作模式转换：



注：

1. 从休眠或绿色模式唤醒，中断使能的情况则进入相应中断，否则执行下一句。
2. 外部复位和 Timer2 中断不能唤醒休眠或绿色模式。
3. 进入休眠或绿色模式前，关闭 WDT 可降低功耗。

各种模式下振荡器模块及Timer0/Timer1的工作状态表

模块	高频模式	低频模式	绿色模式	休眠模式
高频振荡器	运行	由HXEN决定	由HXEN决定	关闭
低频振荡器	运行	运行	运行	关闭
Timer0	运行	运行	定时唤醒模式下运行	计数器模式下运行
Timer1	运行	运行	异步定时唤醒模式下运行	异步计数器模式下运行

6.1 模式切换举例

- 例：高频/低频模式切换到睡眠模式。

```
BCF STATUS,RP0 ;BANK0
BCF OSCCON,T0OSCEN
SLEEP
```

- 例：高频模式切换到低频模式。

```
BCF STATUS,RP0 ;BANK0
BSF OSCCON,SCS ;SCS = 1, 系统进入低频模式
```

- 例：从低频模式切换到高频模式。

```
BCF STATUS,RP0 ;BANK0
BCF OSCCON,SCS ;SCS = 0, 系统进入高频模式
```

- 例：从高频/低频模式切换到绿色模式

;T0定时器定时唤醒

```
BCF STATUS,RP0 ;BANK0
MOVLW 0X05
MOVWF OPTION
BSF OPTION,T0CS
BSF OSCCON, T0OSCEN
BSF INTCON,T0IE ;使能T0 定时器。
BSF INTCON,GIE
CLRF T0
SLEEP
```

- 例：从高频/低频模式切换到绿色模式。

;T0定时器定时唤醒，OSCLM=01，低频晶体振荡器为32,768KHz，定时唤醒时间为0.5s。

```
BCF STATUS,RP0 ;BANK0
MOVLW 0X05
MOVWF OPTION
BSF OPTION,T0CS
BSF OSCCON,T0OSCEN
BCF OSCCON,T0IF
BSF INTCON,T0IE ;使能T0 定时器。
BSF INTCON,GIE
CLRF T0
```



```

RTC_MODE:
    SLEEP
    BCF     STATUS,RP0      ;BANK0
    BCF     INTCON,T0IF    ;0.5s时间到
    ...
    GOTO   RTC_MODE
    
```

6.2 高低频模式切换

图6-2 高低频切换时序图一

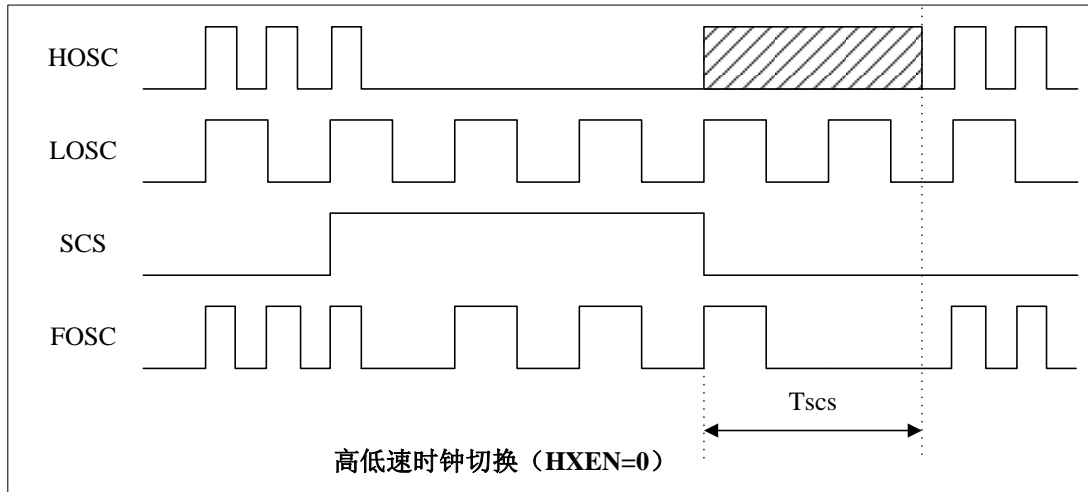
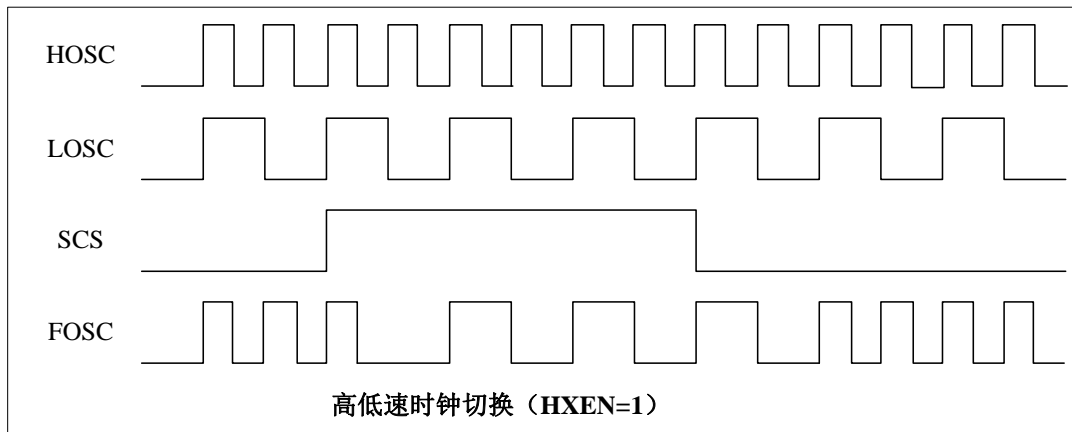


图6-3 高低频切换时序图二



时钟切换时间 (Tscs) 计算:

$$T_{scs} = \text{高频振荡器起振时间} + \text{高频振荡器稳定时间}$$

不同类型高频振荡器的稳定时间表

振荡器类型	高频振荡器稳定时间
高频晶体振荡器	1024 Clock
外部/内部 RC 振荡器	16 Clock

6.3 唤醒时间

系统进入休眠模式后，系统时钟停止运行。外部中断把系统从休眠模式下唤醒时，系统需要等待振荡器起振定时器（OST）定时结束，以使振荡电路进入稳定工作状态，等待的这一段称为唤醒时间。唤醒时间结束后，系统进入高频或低频模式。

唤醒时间的计算如下：

$$\text{唤醒时间} = \text{起振时间} + \text{OST定时时间}$$

同类型振荡器OST定时时间表

振荡器类型	OST 定时时间
高/低频晶体振荡器	1024 Clock
外/内部 RC 振荡器	16 Clock
低频 RC 振荡器	4 Clock

注：

系统进入绿色模式后，低频时钟正常运行。外部或内部中断将系统从绿色模式中唤醒不需要唤醒时间。

6.4 OSCCON 寄存器

07Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OSCCON	TOOSCEN	-	-	-	-	-	HXEN	SCS
R/W	R/W	-	-	-	-	-	R/W	R/W
POR的值	0	-	-	-	-	-	0	q

注： x = 未知， - = 未实现， q = 取值视条件而定

bit 7 TOOSCEN： 低频振荡器使能位

1 = 在低速或绿色模式下使能内部32K WDT振荡器

0 = 在低速或绿色模式下禁止内部32K WDT振荡器

bit 1 HXEN： 高频振荡器使能位

1 = 在低速或绿色模式下使能高频振荡器

0 = 在低速或绿色模式下禁止高频振荡器

bit 0 SCS： 高低频模式选择位

1 = 系统时钟选择为低频系统时钟

0 = 系统时钟选择为高频系统时钟

7 中断源

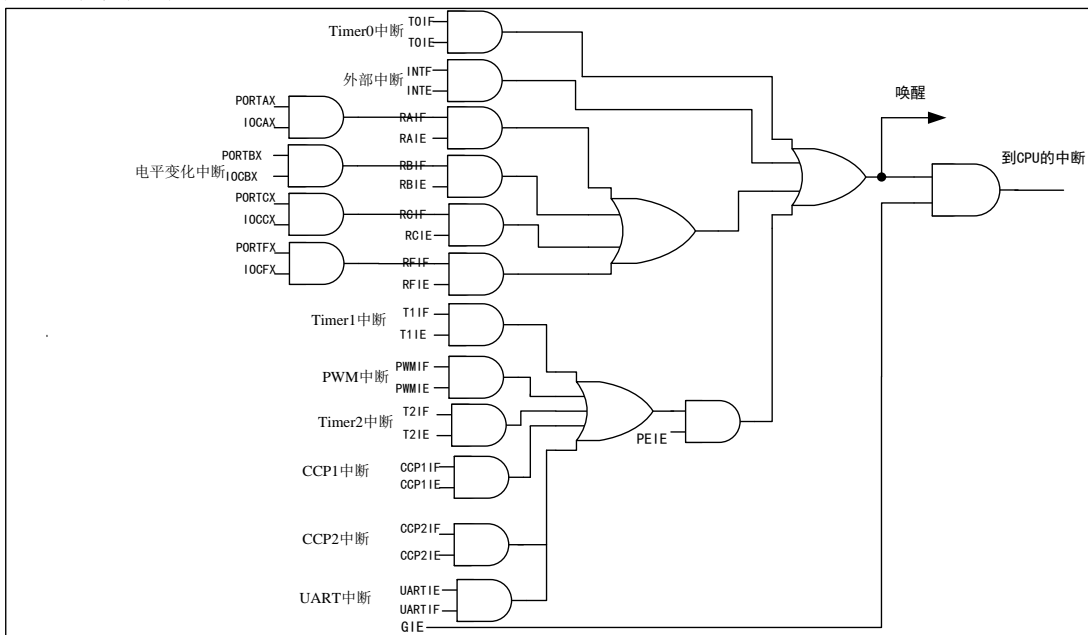
HC18P13xL中断源:

- Timer0定时器中断
- INT0外部中断
- PORT口电平变化中断
- Timer1定时器中断
- Timer2定时器中断
- CCP1中断
- CCP2中断
- AD中断
- UART中断
- PWM中断

系统产生中断时，程序计数器（PC）值压入堆栈，程序跳转至0004h，进入中断服务程序。当程序运行到RETFIE指令时，系统退出中断服务程序，程序计数器值出栈，系统执行PC+1地址对应的指令。

为避免误进入中断，在使能中断和退出中断服务程序之前，必须清除中断标志位。在中断服务程序中不需要软件使能GIE，以免造成程序紊乱。

图7-1 中断示意图



7.1 内核中断

使能内核中断必须将GIE和相应中断的使能位置1，使能PORTB电平变化中断还需要将相应端口配置为输入并且IOCB的相应位置1。INT0外部中断和PORTB电平变化中断可以唤醒SLEEP，Timer0中断在计数器模式和定时唤醒模式下可以唤醒SLEEP。

09h、209h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	x

注： x = 未知

bit 7 **GIE**: 全局中断使能位

1 = 使能所有未屏蔽的中断

0 = 禁止所有中断

bit 5 **T0IE**: Timer0溢出中断使能位

1 = 使能Timer0中断

0 = 禁止Timer0中断

bit 4 **INTE**: INT0外部中断使能位

1 = 使能INT0外部中断

0 = 禁止INT0外部中断

bit 3 **RBIE**: PORTB电平变化中断使能位

1 = 使能PORTB电平变化中断

0 = 禁止PORTB电平变化中断

bit 2 **T0IF**: Timer0溢出中断标志位，Timer0计数寄存器在FFh至00h时产生溢出信号

1 = Timer0计数寄存器溢出（必须由软件清0）

0 = Timer0计数寄存器未溢出

bit 1 **INTF**: INT0外部中断标志位

1 = 发生INT0外部中断（必须由软件清0）

0 = 未发生INT0外部中断

bit 0 **RBIF**: PORTB电平变化中断标志位

1 = PORTB[7:0]中至少有一个口的电平状态发生了改变（必须由软件清0）

0 = PORTB[7:0]电平状态没有变化

078h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	-	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	-	1	1	1	1	1	1	1

bit 6 **INTEDG**: 触发INT0外部中断的边沿选择位

1 = INT0引脚上升沿触发中断

0 = INT0引脚下降沿触发中断

7.2 外设中断

使能外设中断必须将GIE和PEIE置1，同时将相应中断的使能位置1。Timer1门控事件中断可以唤醒SLEEP，Timer1中断在异步计数器模式和异步定时唤醒模式下可以唤醒SLEEP，CCPx中断在捕捉模式下可以唤醒SLEEP。

09h、209h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	x

bit 7 **GIE**: 全局中断使能位

1 = 使能所有未屏蔽的中断

0 = 禁止所有中断

bit 6 **PEIE**: 外设中断使能位

1 = 使能所有未屏蔽的外设中断

0 = 禁止所有外设中断

070h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIE1	-	ADIE	-	-	-	CCP1IE	T2IE	T1IE
R/W	-	R/W	-	-	-	R/W	R/W	R/W
POR的值	-	0	-	-	-	0	0	0

bit 6 **ADIE**: ADC中断使能位

1 = 使能ADC中断

0 = 禁止ADC中断

bit 2 **CCP1IE**: CCP1中断使能位

1 = 使能CCP1中断

0 = 禁止CCP1中断

bit 1 **T2IE**: Timer2计数寄存器与PR2匹配中断使能位

1 = 使能Timer2匹配中断

0 = 禁止Timer2匹配中断

bit 0 **T1IE**: Timer1溢出中断使能位

1 = 使能Timer1溢出中断

0 = 禁止Timer1溢出中断

071h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIE2	-	-	PWM0IE	-	-	UARTIE	-	CCP2IE
R/W	-	-	R/W	-	-	R/W	-	R/W
POR的值	-	-	0	-	-	0	-	0

bit [5] **PWM0IE**: PWM0中断使能位

1 = 使能PWM0中断

0 = 禁止PWM0中断

bit 2 **UARTIE**: UART中断使能位

1 = 使能UART中断

0 = 禁止UART中断

bit 0 **CCP2IE**: CCP2中断使能位

1 = 使能CCP2中断

0 = 禁止CCP2中断

072h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIE3	-	-	-	RFIE -	-	-	RCIE -	RAIE
R/W	-	-	-	R/W	-	-	R/W	R/W
POR的值	-	-	-	0	-	-	0	0

Bit[4:0] **RxIE**: PORTx电平变化中断使能位 (x=A/C/F)

1 = 使能PORTx电平变化中断

0 = 禁止PORTx电平变化中断

054h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR1	-	ADIF	-	-	-	CCP1IF	T2IF	T1IF
R/W	-	R/W	-	-	-	R/W	R/W	R/W
POR的值	-	0	-	-	-	0	0	0

055h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR2	-	-	PWM0IF	-	RXIF	TXIF	-	CCP2IF
R/W	-	-	R/W	-	R/W	R/W	-	R/W
POR的值	-	-	0	-	0	0	-	0

054h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR3	-	-	-	RFIF	-	-	RCIF	RAIF
R/W	-	-	-	R/W	-	-	R/W	R/W
POR的值	-	-	-	0	-	-	0	0

bit 6 **ADIF**: AD中断标志位

1 = ADC转换已完成 (必须由软件清0)

0 = ADC转换未完成或尚未开始

bit 2 **CCP1IF**: CCP1中断标志位

捕捉模式:

1 = 发生了捕捉事件 (必须用软件清零)

0 = 未发生捕捉事件

比较模式:

1 = 发生了比较事件 (必须用软件清零)

0 = 未发生比较事件

PWM 模式:

在此模式下未使用

bit 1 **T2IF**: Timer2计数寄存器与PR2匹配中断标志位

1 = Timer2发生匹配 (必须用软件清零)

0 = Timer2未发生匹配

bit 0 **TIIF**: Timer1溢出中断标志位, Timer1计数寄存器在FFFFh至0000h时产生溢出信号

1 = Timer1计数寄存器溢出 (必须由软件清0)

0 = Timer1计数寄存器未溢出

bit 5 **PWM0IF**: PWM0中断标志位

1 = PWM0中断产生中断 (必须由软件清0)

0 = PWM0中断未产生中断

bit 3 **RXIF**: UART接收标志位

1 = UART接收中断产生中断 (必须由软件清0)

0 = UART接收中断未产生中断

bit 2 **TXIF**: UART发送中断标志位

1 = UART发送中断产生中断 (必须由软件清0)

0 = UART发送中断未产生中断

bit 0 **CCP2IF**: CCP2中断标志位

捕捉模式:

1 = 发生了捕捉事件 (必须用软件清零)

0 = 未发生捕捉事件

比较模式:

1 = 发生了比较事件 (必须用软件清零)

0 = 未发生比较事件

PWM 模式:

在此模式下未使用

bit [4:0] **RxIF**: PORTx电平变化中断标志位(x=A/C/F)

1 = PORTx[7:0]中至少有一个口的电平状态发生了改变 (必须由软件清0)

0 = PORTx[7:0]电平状态没有变化

7.3 GIE 全局中断

只有当全局中断控制位GIE置“1”的时候程序才能响应中断请求。一旦有中断发生，程序计数器入栈，程序转至中断向量地址（ORG 0004H），堆栈层数加1。

例：设置全局中断控制位（GIE）

```
BSF      INTCON,GIE      ;使能 GIE。
```

注：在所有中断中，GIE 都必须处于使能状态。

7.4 中断保护

有中断请求发生并被响应后，程序转至0004H执行中断服务程序。

中断服务程序开始执行时，需保存W寄存器、STATUS寄存器、PCLATH寄存器的内容；结束中断服务程序时，恢复PCLATH寄存器、STATUS寄存器、W寄存器的数值，注意顺序。

注：

- 1.多 BANK 的 IC 中，为了使保存系统寄存器的 RAM 可以在所有 BANK 访问，建议将这些 RAM 指定在所有 BANK 均映射的地址，该 IC 为 0X40~0X7F。
- 2.在退出中断时，由于需要先恢复 STATUS，再使用 MOVF 指令恢复 W，可能会改变 STATUS，因此必须使用 SWAPF 指令恢复 W。注意在中断中共有两句 SWAPF 指令。

➤ 例：对W、PCLATH 和STATUS 进行入栈保护。

```

ORG      0000H
GOTO    START
ORG      0004H
GOTO    INT_SERVICE
ORG      0010H

START:
...

INT_SERVICE:
MOVWF   W_TEMP           ;保存 W。
SWAPF   STATUS,W
MOVWF   STATUS_TEMP      ;保存 STATUS。
MOVF    PCLATH,W
MOVWF   PCLATH_TEMP      ;保存 PCLATH。
CLRF    STATUS           ;切换到BANK0
...
MOVF    PCLATH_TEM ,W
```



```

MOVWF PCLATH ;恢复PCLATH。
SWAPF STATUS_TEMP,W
MOVWF STATUS ;恢复STATUS。
SWAPF W_TEMP,F
SWAPF W_TEMP,W ;恢复W。
RETFIE ;退出中断。
...
END
    
```

7.5 Timer0 定时器中断

T0 溢出时，无论 TOIE 处于何种状态，TOIF 都会置“1”。若 TOIE 和 TOIF 都置“1”，且 GIE 使能，系统就会响应 TIMER0 的中断；若 TOIE = 0，则无论 TOIF 是否置“1”，系统都不会响应 TIMER0 中断。

7.6 INT0 外部中断

INT0 被触发，则无论 INTE 处于何种状态，INTF 都会被置“1”。如果 INTF=1 且 INTE=1，GIE 使能，系统响应该中断；如果 INTF=1 而 INTE=0，系统并不会执行中断服务。在处理多中断时尤其需要注意。

078h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	RBPUB	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

bit 6 **INTEDG**: 触发INT0外部中断的边沿选择位

1 = INT0引脚上升沿触发中断

0 = INT0 引脚下降沿触发中断

➤ 例：INT0 中断请求设置，电平触发。

```

BSF STATUS,RP0 ;BANK1
BSF OPTION,INTEG ;INT0 置为上升沿触发。
BCF STATUS,RP0 ;BANK0
BCF INTCON,INTF ;INT0 中断请求标志清零。
BSF INTCON,INTE ;使能INT0 中断。
BSF INTCON,GIE ;使能GIE。
    
```

➤ 例：INT0 中断。

```

ORG 0004H ;
GOTO INT_SERVICE
    
```

INT_SERVICE:

```

... ;保存STATUS、W 和PCLATH。
BCF STATUS,RP0 ;BANK0
BTFSS INTCON,INTF ;检测TOIF。
GOTO EXIT_INT ;TOIF = 0，退出中断。
BCF INTCON,INTF ;TOIF 清零。
... ;INT0 中断服务程序。
    
```

EXIT_INT:

```

...
... ;恢复STATUS、W和PCLATH。
RETFIE ;退出中断。
    
```

7.7 PORT 电平变化中断

PORT_x电平变化中断时，则无论RBIE处于何种状态，RxIF都会被置“1”。如果RxIF=1 且RxIE=1，**GIE使能**，系统响应该中断;如果RxIF=1 而RxIE=0，系统并不会执行中断服务。

电平变化中断必须将PORT_x端口设为输入，并将寄存器IOC_x对应位置“1”。

041h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCB	IOCB7	IOCB6	-	IOCB4	IOCB3	IOCB2	IOCB1	IOCB0
R/W	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	-	0	0	0	0	0

IOC_B[7:0]: PORT_B_x电平变化中断使能控制位。

0 = 该端口禁止电平变化中断;

1 = 该端口使能电平变化中断。

040h~045h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOC _x	IOC _x 7	IOC _x 6	IOC _x 5	IOC _x 4	IOC _x 3	IOC _x 2	IOC _x 1	IOC _x 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

IOC_x[7:0]: PORT_{xx}电平变化中断使能控制位。

0 = 该端口禁止电平变化中断;

1 = 该端口使能电平变化中断。

x = A、C、F

注:

- 1.如要允许 PORT_B 口电平变化中断必须将 IOCB 的对应端口的位置 1。
- 2.PORT_B 电平变化中断中，在清零 RBIF 之前必须执行 PORT_B 端口读操作。

➤ 例: PORT_B1 电平变化中断请求设置。

```

MOVLW    0X02
BSF      STATUS,RP0      ;BANK1
IORWF    TRISB,F         ;PORTB1 端口为输入。
MOVLW    0X02
BCF      STATUS,RP0      ;BANK0
IORWF    IOCB,F          ;使能PORTB1 端口为电平变化中断。
MOVF     PORTB, W        ;读PORTB 口。
BCF      INTCON,RBIF     ;PROTB 中断请求标志清零。
BSF      INTCON,RBIE     ;使能PROTB 中断。
BSF      INTCON,GIE      ;使能GIE。
    
```

➤ 例: PORT_B 中断。

```

ORG      0004H
    
```

```

GOTO      INT_SERVICE
INT_SERVICE:
...
;保存STATUS、W 和PCLATH。
BCF       STATUS,RP0      ;BANK0
BTFS     INTCON,RBIF      ;检测RBIF。
GOTO     EXIT_INT        ;RBIF = 0, 退出中断。
MOVF     PORTB,W        ;读PORTB 端口
BCF       INTCON,RBIF      ;RBIF 清零。
...
;PORTB 电平变化中断服务程序。
...

EXIT_INT:
...
;恢复STATUS、W 和PCLATH。
RETFIE    ;退出中断。

```

➤ 例：PORTB 中断唤醒SLEEP。

```

BCF       STATUS,RP0      ;BANK0
MOVLW    0X02
IORWF    TRISB,F         ;PORTB1 端口为输入。
MOVLW    0X02
BCF       STATUS,RP0      ;BANK0
IORWF    IOCB,F         ;使能PORTB1 端口为电平变化中断。
MOVF     PORTB,W         ;读PORTB 口。
BCF       INTCON,RBIF      ;PORTB 中断请求标志清零。
BSF      INTCON,RBIE      ;使能PORTB 中断。
SLEEP
BCF       INTCON,RBIE
BCF       STATUS,RP0      ;BANK0
MOVF     PORTB,W         ;读PORTB 端口。
...
;其他程序。

```

注：

1. 如要允许 PORTx 口电平变化中断必须将 IOCx 的对应端口的位置 1。
2. PORTx 电平变化中断中，在清零 RxIF 之前必须执行 PORTx 端口读操作。

7.8 Timer2 定时器中断

当T2的值和PR2的值相同时，TIMER2中断被触发，则无论T2IE 处于何种状态，T2IF 都会被置“1”。如果T2IF=1 且T2IE=1，且PEIE、GIE均使能，系统响应该中断；如果T2IF=1 而T2IE=0，系统并不会执行中断服务。

➤ 例：TIMER2 中断请求设置

```

BCF       STATUS,RP0      ;BANK0
MOVLW    0XFF

```

```

MOVWF PR2 ;设置T2 周期。
MOVLW 0X04
MOVWF T2CON ;设置分频比。
CLRF T2
BSF PIE1,T2IE ;使能TIMER2 中断。
BSF INTCON,GIE
BSF T2CON,T2ON ;使能TIMER2。
    
```

➤ 例：TIMER2 中断。

```

ORG 0004H
GOTO T2INT_SERVICE
T2INT_SERVICE:
... ;保存STATUS、W 和PCLATH。
BCF STATUS,RP0 ;BANK0
BTFSS PIR1,T2IF ;检测T2IF。
GOTO EXIT_INT ;T2IF = 0，退出中断。
BCF PIR1,T2IF ;T2IF 清零。
... ;TIMER2 中断服务程序。
...
EXIT_INT:
... ;恢复STATUS、W和PCLATH。
RETFIE ;退出中断。
    
```

7.9 Timer1 中断

T1溢出时，无论T1IE 处于何种状态，T1IF都会置“1”。若T1IE 和T1IF 都置“1”，且PEIE、GIE均使能，系统就会响应TIMER1的中断;若T1IE = 0，则无论T1IF 是否置“1”，系统都不会响应TIMER1中断。

➤ 例：TIMER1工作于异步计数模式，并中断唤醒SLEEP

```

MOVWF 0XA4
BCF STATUS,RP0 ;BANK0
MOVWF T1CON ;T1时钟源为T1CKI;分频比为1:4;异步计数器
模式
MOVWF nnH
MOVWF T1H
MOVWF nnH
MOVWF T1L ;Timer1赋初值
MOVLW 0XC0
MOVWF INTCON ;使能外设中断
BCF STATUS,RP0 ;BANK0
BSF PIE1,T1IE
BCF STATUS,RP0
BCF PIR1,T1IF
BSF T1CON,T1ON
BSF STATUS,RP0
    
```

```

BCF      OSCCON,T0OSCEN      ;禁止低频晶体振荡器
SLEEP                                ;进入SLEEP

T1INT_SERVICE:
...                                    ;保存STATUS、W 和PCLATH。
BCF      STATUS,RP0          ;BANK0
BTFSS   PIR1,T1IF           ;检测T1IF。
GOTO    EXIT_INT            ;T1IF = 0，退出中断。
BCF      PIR1,T1IF          ;T1IF 清零。
...                                    ;TIMER1 中断服务程序。
...

EXIT_INT:
...                                    ;恢复STATUS、W和PCLATH。

RETFIE                                ;退出中断。

```

7.10 AD 中断

当 ADC 完成，ADON 被硬件清零，无论 ADIE 处于何种状态，与此同时 ADIF 被置“1”。若 ADIE、ADIF 为“1”，且 PEIE、GIE 均使能，系统就会相应 ADC 中断；若 ADIE = 0，则无论 ADIF 是否置“1”，系统都不会响应 ADC 中断。

7.11 CCP 中断

当发生 CCPx 中断时，无论 CCPxIE 处于何种状态，CCPxIF 被置“1”。若 CCPxIE、CCPxIF 为“1”，且 PEIE、GIE 均使能，系统就会相应 CCPx 中断；若 CCPxIE = 0，则无论 CCPxIF 是否置“1”，系统都不会响应 CCPx 中断。

7.12 UART 中断

当 UART 接收、发送完成后，无论 UARTIE 处于何种状态，TXIF、RXIF 都被置“1”。若 UARTIE、TXIF、RXIF 为“1”，且 PEIE、GIE 均使能，系统就会相应 UART 中断。

7.13 PWM 中断

当 PWM0 周期计数器溢出，无论 PWM0IE 处于何种状态，PWM0IF 都被置“1”。若 PWM0IE 为“1”，且 PEIE、GIE 均使能，系统就会相应产生 PWM 中断。

7.14 多中断操作

在同一时刻，系统中可能出现多个中断请求。此时，用户必须根据系统的要求对各中断进行优先权的设置。中断请求标志IF由中断事件触发，当IF处于有效值“1”时，系统并不一定会响应该中断。各中断触发事件如下表所示：

中断	有效触发
T0IF	T0溢出
INTF	由INTEDG控制
RBIF	PORTB电平变化
RxIF	其他PORT口电平变化中断
T2IF	T2的值和PR2相同

UARTIF	UART发生发送接收事件
PWM0IF	PWM0周期计数溢出中断

多个中断同时发生时，需要注意的是：首先，必须预先设定好各中断的优先权；其次，利用IE和IF 控制系统是否响应该中断。在程序中，必须对中断控制位和中断请求标志进行检测。

➤ 例：多中断条件下检测中断请求。

```

        ORG    0004H ;
        GOTO  INT_SERVICE

INT_SERVICE:
        ...
INT0CHK:
        BCF   STATUS,RP0           ;保存STATUS、W和PCLATH。
        BTFSS INTCON,INTE         ;检查是否有INT0 中断请求。
        GOTO  INT0CHK             ;BANK0
        BTFSS INTCON,INTE         ;检查是否使能INT0 中断。
        GOTO  INT0CHK             ;跳到下一个中断。
        BTFSC INTCON,INTF         ;检查是否有INT0 中断请求。
        GOTO  INT0               ;进入INT0 中断。

INTT0CHK:
        BTFSS INTCON,T0IE         ;检查是否有T0 中断请求。
        GOTO  INTT2CHK           ;检查是否使能T0 中断。
        BTFSS INTCON,T0IF         ;跳到下一个中断。
        GOTO  INTT0               ;检查是否有T0 中断请求。
        GOTO  INTT0               ;进入T0 中断。

INTT2CHK:
        BCF   STATUS,RP0           ;BANK0
        BTFSS PIE1,T2IE         ;检查是否使能T2 中断。
        GOTO  INTRBCHK           ;跳到下一个中断。
        BCF   STATUS,RP0           ;BANK0
        BTFSC PIR1,T2IF         ;检查是否有T2 中断请求。
        GOTO  INTT2               ;进入T2 中断。

INTRBCHK:
        BTFSS INTCON,RBIE         ;检查是否使能PORTB 电平变化中断。
        GOTO  INT_EXIT           ;跳到中断结束。
        BTFSC INTCON,RBIF         ;检查是否有PORTB 电平变化中断请求。
        GOTO  INTRB               ;进入PORTB 电平变化中断。

INTT2:
        BCF   PIR1,T2IF
        .....
        GOTO INT_EXIT

INT_EXIT:
        ...
    RETFIE                        ;恢复STATUS、W和PCLATH。
                                ;退出中断。
    
```

8 I/O口

HC18P13xL共有七组双向端口：

- PORTA口，
- PORTB口，
- PORTC口
- PORTF口

8.1 I/O 口输入输出控制寄存器

010h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

011h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

012h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISC	-	-	-	-	-	-	TRISC1	TRISC0
R/W	-	-	-	-	-	-	R/W	R/W
POR的值	-	-	-	-	-	-	1	1

015h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

TRISx[7:0]: PORTx[7:0]的输入输出控制位

1 =输入状态

0 =输出状态

注： 1. PORTB5 可设置为开漏输出。

08Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELL	ANSEL7	ANSEL6	-	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
R/W	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	-	1	1	1	1	1

08Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELH	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

ANSEL[15:0]: A/D引脚数模控制位

- 1: 模拟模式, 作为模拟信号口, 仅可作为AD通道的模拟输入。
- 0: 数字模式, 作为数字输入或输出口。

注: ANSEL 上电初始值为 B' xxx1 1111', 即作为模拟输入。无论是否应用到 AD, 均需要在上电后, 对 IO 操作之前按需配置, 否则 IO 口可能无法受控于对应的端口寄存器, 状态将不确定。

ANSEL[4:0] 对应 AN4~AN0 (PA4~PA0), ANSEL[7:6] 对应 AN7、AN6 (PA7、PA6)

ANSEL[8] 对应 AN8 (即 PA5)

ANSEL[13:9] 对应 AN13~AN9 (PB4~PB0), ANSEL[15:14] 对应 AN15、AN14 (PB7、PB6)

8.2 I/O口上拉控制寄存器

028~02Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPUx	WPUx7	WPUx6	WPUx5	WPUx4	WPUx3	WPUx2	WPUx1	WPUx0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

x = A、B、C、F

WPUx[7:0]: PORTx[7:0]的上拉使能位

- 1 = 上拉禁止
- 0 = 上拉使能

078h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	RBPUB	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

bit 7 RBPUB: PORTB上拉使能位

- 1 = PORTB上拉由WPUB决定
- 0 = 使能PORTB上拉(此时无论WPUB为何值PORTB都上拉)

注: 1. 注意此处上拉控制寄存器逻辑, 0 为使能, 1 为禁止。
2. 当端口设置为输出时, 上拉/下拉默认关闭(硬件关闭)

8.3 I/O口下拉控制寄存器

034h~039h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPDx	WPDx7	WPDx6	WPDx5	WPDx4	WPDx3	WPDx2	WPDx1	WPDx0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

x = A、B、C、F

WPDx[7:0]: PORTx[7:0]的上拉使能位

1 = 下拉禁止

0 = 下拉使能

注: 1. 注意此处下拉控制寄存器逻辑, 0 为使能, 1 为禁止;
 2. 当端口设置为输出时, 下拉无效;
 3. 当下拉打开时, 上拉根据控制寄存器使能或禁止, 即设置为输入时可同时打开上下拉。

8.4 PORT驱动控制寄存器

04Dh~04Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DRENxL	DRENx7L	DRENx6L	DRENx5L	DRENx4L	DRENx3L	DRENx2L	DRENx1L	DRENx0L
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

0A0h~0A2h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DRENxH	DRENx7H	DRENx6H	DRENx5H	DRENx4H	DRENx3H	DRENx2H	DRENx1H	DRENx0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

x=A/B/C

DRENxL[7:0]: DRENxL[7:0]的驱动控制位

DRENxH=0时:

0 = 源电流/灌电流 (Level0) (IOH=0.9VDD 22mA@ 5V) (IOL=0.1VDD 85mA@ 5V)

1 = 源电流/灌电流 (Level1) (IOH=0.9VDD 8mA@ 5V) (IOL=0.1VDD 10mA@ 5V)

DRENxH=1时:

0 = 源电流/灌电流 (Level3) (IOH=0.9VDD 10mA@ 5V) (IOL=0.1VDD 25mA@ 5V)

1 = 源电流/灌电流 (Level2) (IOH=0.9VDD 4mA@ 5V) (IOL=0.1VDD 8mA@ 5V)

052h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DRENFL	DRENF7L	DRENF6L	DRENF5L	DRENF4L	DRENF3L	DRENF2L	DRENF1L	DRENF0L
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

0A5h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DRENFH	DRENF7H	DRENF6H	DRENF5H	DRENF4H	DRENF3H	DRENF2H	DRENF1H	DRENF0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

DRENFL[7:0]: DRENxL[7:0]的驱动控制位

DRENFH=0时:

0 = 源电流/灌电流 (Level0) (IOH=0.9VDD 22mA@ 5V) (IOL=0.1VDD 85mA@ 5V)

1 =源电流/灌电流 (Level1) (IOH=0.9VDD 8mA@ 5V) (IOL=0.1VDD10mA@ 5V)

DRENFH=1时:

0 = 源电流/灌电流 (Level3) (IOH=0.9VDD10mA@ 5V) (IOL=0.1VDD 25mA@ 5V)

1 = 源电流/灌电流 (Level2) (IOH=0.9VDD 4mA@ 5V) IOL=0.1VDD 8mA@ 5V)

8.5 I/O 口数据寄存器

01Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	x	x	x	x	x	x	x	x

01Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	x	x	x	x	x	x	x	x

01Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTC	-	-	-	-	-	-	PORTC1	PORTC0
R/W	-	-	-	-	-	-	R/W	R/W
POR的值	-	-	-	-	-	-	x	x

021h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTF	PORTF7	PORTF6	PORTF5	PORTF4	PORTF3	PORTF2	PORTF1	PORTF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	x	x	x	x	x	x	x	x

8.6 管脚配置寄存器

04Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORCTR	-	-	-	CCPCT	-	-	UAPCT1	UAPCT0
R/W	-	-	-	R/W	-	-	R/W	R/W
POR的值	-	-	-	0	-	-	0	0

bit[1:0] UAPCT[1:0] UART 管脚配置位

00 = RX 配置在 PORTC0, TX 配置在 PORTC1 (默认);

01 = RX 配置在 PORTF0, TX 配置在 PORTF1;

10 = RX 配置在 PORTB0, TX 配置在 PORTB1;

11 = RX 配置在 PORTA0, TX 配置在 PORTA1;

Bit[4]:CCPCT CCP 管脚配置为

0 = CCP1/CCP2 管脚配置在 PORTB2/PORTB3 (默认)

1 = CCP1/CCP2 管脚配置在 PORTA6/PORTA7

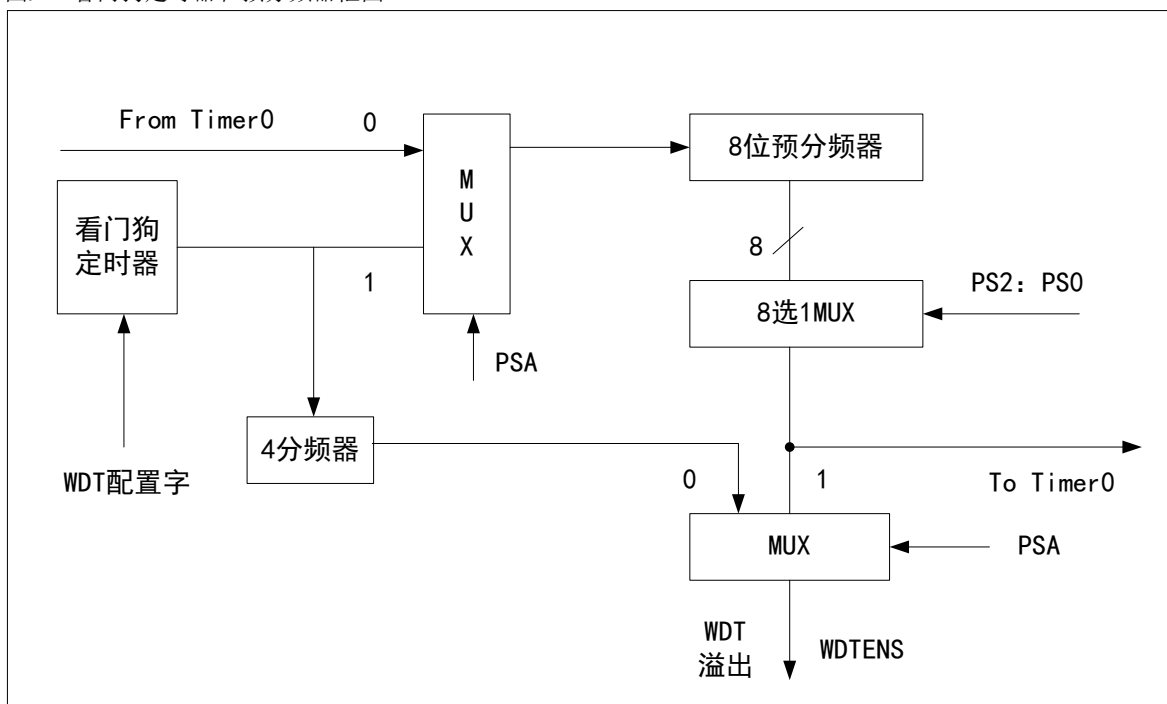
注: 管脚复用功能优先级: UART > CCP > IO。

9 定时器/计数器

9.1 看门狗定时器

HC18P13xL的看门狗定时器与Timer0定时器/计数器共用一个预分频器。当PSA为0时，看门狗定时器每72ms（典型值）产生一个溢出信号；当PSA为1时，WDT溢出时间由预分频器OPTION[2:0]设置决定，具体请参考第10章 Timer0定时器/计数器。

图9-1 看门狗定时器和预分频器框图



079h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCON	LVD2EN	LVD1EN	-	WDTENS	LVD2F	LVD1F	POR	BOR
R/W	R/W	R/W	-	R/W	R	R	R/W	R/W
POR的值	0	0	-	1	q	q	q	q

bit 4 **WDTENS**: 硬件看门狗软件使能位（需配置字使能看门狗，否则该位无效）

1 = 软件使能硬件看门狗定时器

0 = 软件屏蔽硬件看门狗定时器

注：看门狗的使能逻辑 看门狗使能 = 芯片配置字使能（WDTEN） & 软件使能（WDTENS）

当系统处于休眠或绿色模式，看门狗定时器溢出将唤醒SLEEP并使其返回高频或低频模式，程序从SLEEP指令下一条开始执行。

注:

1. 对看门狗清零之前, 检查I/O 口的状态和RAM 的内容可增强程序的可靠性;
2. 不能在中断中对看门狗清零, 否则无法侦测到主程序跑飞的状况;
3. 程序中应该只在主程序中有一次清看门狗的动作, 这种架构能够最大限度的发挥看门狗的保护功能。

➤ 例: 看门狗在主程序中的应用

MAIN:

```

BCF    STATUS,RP0      ;BANK0
BSF    PCON,WDTENS     ;软件使能WDT
...    ;检查IO状态是否正确
...    ;检查RAM是否正确
GOTO   ERR             ;检查IO/RAM出错, 进入出错处理程序
CLRWDT ;在整个程序中, 仅有一条清狗指令
...
CALL   SUB1
CALL   SUB2
...
GOTO   MAIN

```

➤ 例: 在休眠状态下, 屏蔽看门狗功能, 可以节省系统功耗

```

...
BCF    STATUS,RP0      ;BANK0
BCF    PCON,WDTENS     ;软件屏蔽看门狗功能
BCF    OSCCON,T0OSCEN ;禁止低频晶体振荡器
SLEEP  ;进入休眠模式
BSF    PCON,WDTENS     ;唤醒后,重新使能看门狗功能
...

```

➤ 例: 对看门狗定时器操作, 看门狗定时器使能和清零

```

BCF    STATUS,RP0      ;BANK0
BSF    PCON,WDTENS     ;使能看门狗
CLRWDT ;看门狗定时器清零
...

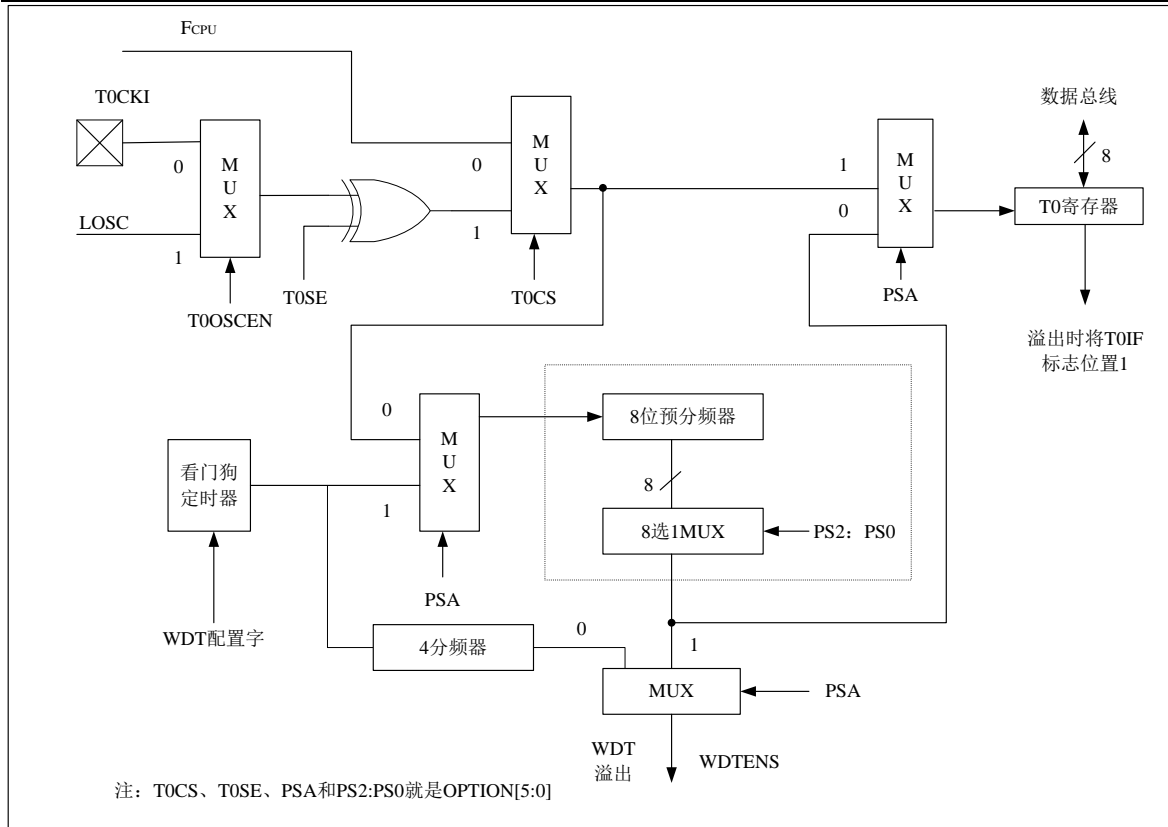
```

9.2 Timer0 定时器/计数器

Timer0 定时器/计数器模块具有如下功能:

- 8 位可编程定时器
- 外部事件计数器
- 绿色模式定时唤醒

图10-1 Timer0模块和预分频器（与WDT共享）框图



07Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OSCCON	TOOSCEN	-	-	-	-	-	HXEN	SCS
R/W	R/W	-	-	-	-	-	R/W	R/W
POR的值	0	-	-	-	-	-	0	q

078h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	-	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	-	1	1	1	1	1	1	1

看门狗定时器与Timer0定时器/计数器共用一个预分频器，当PSA=1预分频器分配给WDT时，Timer0在所选中时钟源的每个周期递增；当PSA=0预分频器分配给Timer0时，Timer0根据PS[2:0]值选择的预分频时钟递增。

Timer0的预分频器不可寻址，当预分频器分配给Timer0时，对Timer0计数寄存器的写操作可以对预分频器清0。

Timer0预分频比选择

PS[2:0]	Timer0预分频比	WDT预分频比	WDT溢出时间（典型值）
000	1 : 2	1 : 1	18ms
001	1 : 4	1 : 2	36 ms
010	1 : 8	1 : 4	72ms
011	1 : 16	1 : 8	144ms
100	1 : 32	1 : 16	288ms
101	1 : 64	1 : 32	576ms

110	1: 128	1: 64	1.152s
111	1: 256	1: 128	2.304s

Timer0 工作模式选择

TOCS	TOOSCEN	T0SE	Timer0工作状态
0	x	x	定时器模式，计数时钟 FCPU， 休眠和绿色模式下停止
1	0	0	计数器模式，计数时钟 T0CKI，上升沿计数 休眠模式下工作，溢出中断可唤醒 SLEEP
1	0	1	计数器模式，计数时钟 T0CKI，下降沿计数 休眠模式下工作，溢出中断可唤醒 SLEEP
1	1	0	定时唤醒模式，计数时钟 LOSC，上升沿计数 绿色模式下工作，溢出中断可唤醒 SLEEP
1	1	1	定时唤醒模式，计数时钟 LOSC，下降沿计数 绿色模式下工作，溢出中断可唤醒 SLEEP

注：

Timer0 工作模式的选择需符合上表描述，选择除上表以外情况可能会造成程序运行混乱，请谨慎操作

- 例：Timer0工作于定时器模式，计数时钟为Fcpu，T0计满到FF后溢出进入中断

```

MOVLW    0X01
BCF      STATUS,RP0           ;BANK0
MOVWF   OPTION               ;定时器模式，分频比为1:4
MOVLW    0X00
MOVWF   T0                   ;T0赋初值
BSF      INTCON,T0IE
BCF      INTCON,T0IF
BSF      INTCON,GIE
    
```

T0INT_SERVICE:

```

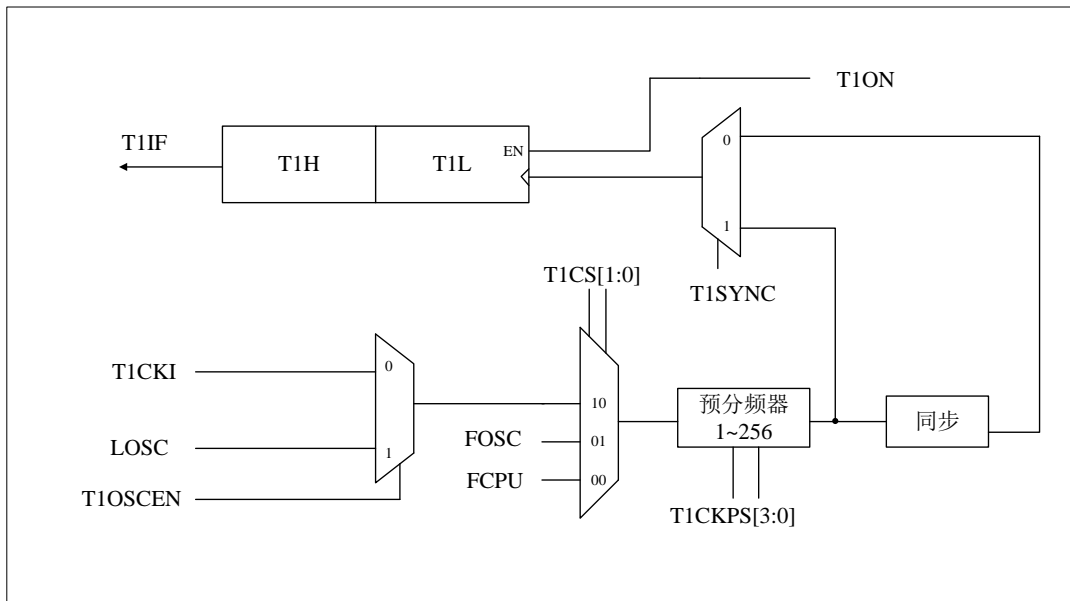
...                                     ;保存STATUS、W 和PCLATH。
BCF      STATUS,RP0           ;BANK0
BTFS    INTCON,T0IF          ;检测T0IF。
GOTO    EXIT_INT             ;T0IF = 0，退出中断。
BCF      INTCON,T0IF          ;T0IF 清零。
...                                     ;TIMER0 中断服务程序。
...
EXIT_INT:
...                                     ;恢复STATUS、W和PCLATH。
RETFIE   ;退出中断。
    
```

9.3 Timer1 定时器/计数器

Timer1 定时器/计数器模块具有如下功能：

- 16 位可编程定时器
- 外部事件计数器，可编程选择同步、异步功能
- 绿色模式定时唤醒

图11-1 Timer1 模块框图



9.3.1 Timer1 控制寄存器

05Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T1CON	T1CS1	T1CS0	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	-	T1ON
R/W	R/W	R/W	R/W	R/W	R/W	R/W	-	R/W
POR的值	0	0	0	0	0	0	-	0

Timer1 时钟源选择

T1CS1	T1CS0	T1OSCEN	时钟源
0	0	x	指令时钟 (F _{CPU})
0	1	x	系统时钟 (F _{sys})
1	0	0	T1CKI 引脚上的外部时钟
1	0	1	低频系统时钟

注：

Timer1 时钟源的选择需符合上表描述，选择除上表以外情况会造成程序运行不正常，请谨慎操作

Timer1 输入时钟预分频比选择

T1CKPS[3:0]	Timer1 预分频比
0000	1 : 1
0001	1 : 2
0010	1 : 4

0011	1: 8
0100	1: 16
0101	1: 32
0110	1: 64
0111	1: 128
1xxx	1: 256

Timer1的预分频器不可寻址，可以通过对Timer1计数寄存器写操作将预分频器清0。

Timer1工作模式选择

T1ON	T1CS[1:0]	T1OSCEN	T1SYNC	Timer1工作模式
1	00	x	x	定时器模式，休眠和绿色模式下停止
1	01	x	x	定时器模式，休眠和绿色模式下停止
1	10	0	0	同步计数器模式，休眠模式下停止
1	10	0	1	异步计数器模式，休眠模式下工作，溢出中断可唤醒SLEEP
1	10	1	0	同步定时唤醒模式，绿色模式下停止，溢出中断不能唤醒SLEEP
1	10	1	1	异步定时唤醒模式，绿色模式下工作，溢出中断可唤醒SLEEP

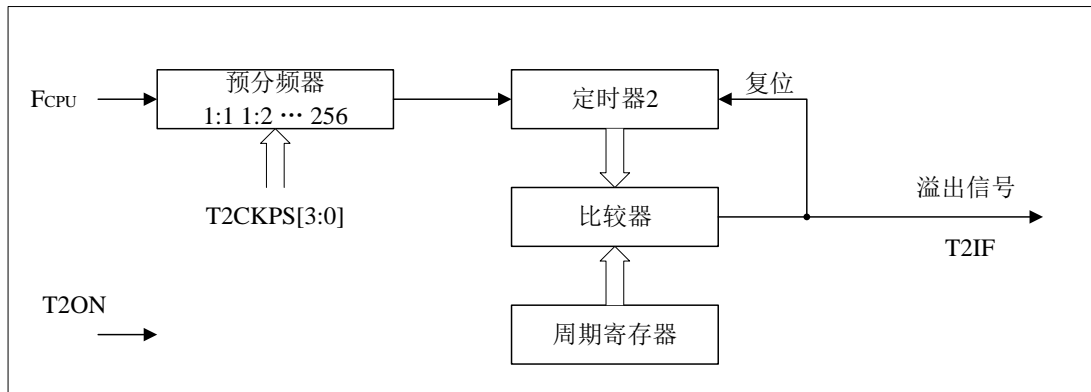
注:

- 1、T1 为 16 位计时器，在溢出中断重新赋值时应先 T1H，后 T1L，避免 T1L 在操作中的进位被覆盖；清空时则应先 T1L 后 T1H，避免 T1L 进位意外进入 T1H 造成清空失败。
- 2、Timer1 工作于同步计数器模式和同步定时唤醒模式时，不能唤醒 SLEEP 或绿色模式
- 3、Timer1 工作模式的选择需符合上表描述，选择除上表以外情况可能会造成程序运行混乱，请谨慎操作

9.4 Timer2 定时器

Timer2定时器具有8位预分频器和8位周期寄存器(PR2),Timer2定时器的输入时钟为指令时钟FCPU,输入时钟通过预分频器产生Timer2计数时钟,当计数到与周期寄存器(PR2)的值相同时,在下一指令周期产生Timer2溢出信号,可根据实际需要选择不同的预分频比及设置周期寄存器的值,产生不同溢出时间。

图12-1 Timer2模块框图



9.4.1 Timer2 控制寄存器

05Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T2CON	-	T2CKPS3	T2CKPS2	T2CKPS1	T2CKPS0	T2ON	-	-
R/W	-	R/W	R/W	R/W	R/W	R/W	-	-
POR的值	-	0	0	0	0	0	-	-

bit 2 **T2ON**: Timer2模块使能位

1 = 使能Timer2模块

0 = 禁止Timer2模块

Timer2具有一个8位可编程预分频器,关闭Timer2模块和对Timer2计数寄存器或T2CON寄存器写操作都将对预分频器清0。

T2CKPS[3:0]	Timer2 预分频比
0000	1 : 1
0001	1 : 2
0010	1 : 4
0011	1 : 8
0100	1 : 16
0101	1 : 32
0110	1 : 64
0111	1 : 128
1xxx	1 : 256

9.4.2 Timer 计数寄存器

05Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T2	Timer2计数寄存器							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

9.4.3 Timer2 周期寄存器

05Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PR2	Timer2周期寄存器							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	1	1	1	1	1	1	1	1

Timer2 定时器的输入时钟为指令时钟 F_{CPU}，输入时钟通过预分频器产生 Timer2 计数信号，当计数到与周期寄存器（PR2）的值相同时产生 Timer2 溢出信号。

$$\text{Timer2 溢出时间} = (\text{PR2} + 1) * \text{预分频比}/\text{Fcpu}。$$

9.5 CCP 模块

HC18P13xL具有2个独立的CCP模块CCP1和CCP2，每个CCP模块具有三种模式：

- 捕捉
- 比较
- PWM

CCP模块的时基由Timer1和Timer2提供。

CCP模块的时基

CCP模式	时钟源
捕捉	Timer1
比较	Timer1
PWM	Timer2/Timer1

082h、085h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CCPxCON	-	-	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	-	-	0	0	0	0	0	0

bit 5~4 **DCxB[1:0]**: PWM占空比最低有效位

捕捉模式：未使用

比较模式：未使用

PWM模式：PWM占空比的低2位，高8位是CCPRxL寄存器

bit 3~0 **CCPxM[3:0]**: CCPx模式选择位

0000 = 捕捉/比较/PWM关闭（复位CCP模块）

0001 = 未使用（保留）

0010 = 比较模式，匹配时输出翻转电平（PIRx寄存器的CCPxIF位置1）

0011 = 未使用（保留）

- 0100 = 捕捉模式，每个下降沿
- 0101 = 捕捉模式，每个上升沿
- 0110 = 捕捉模式，每4个上升沿
- 0111 = 捕捉模式，每16个上升沿
- 1000 = 比较模式，匹配时输出高电平（PIRx寄存器的CCPxIF位置1）
- 1001 = 比较模式，匹配时输出低电平（PIRx寄存器的CCPxIF位置1）
- 1010 = 比较模式，匹配时仅产生软件中断（PIRx寄存器的CCPxIF位置1，CCPx引脚不受影响）
- 1011 = 比较模式，触发特殊事件（PIRx寄存器的CCPxIF位置1，Timer1计数寄存器复位，CCPx引脚不受影响。）
- 11xx = PWM模式

9.5.1 捕捉模式

在输入捕捉模式，适合用于测量引脚输入周期性方波信号的周期、频率和占空比等，也适合用于测量引脚输入的非周期性矩形方波脉冲信号的宽度、到达时刻或消失时刻等参数。

当CCPx模块工作于捕捉模式时，一旦有下列事件在引脚CCPx上发生，CCPRx寄存器立即捕捉下这一时刻的TMR1计数值：

- 每个下降沿
- 每个上升沿
- 每 4 个上升沿
- 每 16 个上升沿

CCPxCON寄存器的CCPxM[3:0]设置的是预分频器，关闭CCP模块或者CCP模块不在捕捉模式，预分频计数器将会被清0。为避免错误中断，可在改变预分频比前通过清0 CCPxCON寄存器来关闭CCP模块。

在捕捉模式下，Timer1必须运行在定时器模式或同步计数器模式。

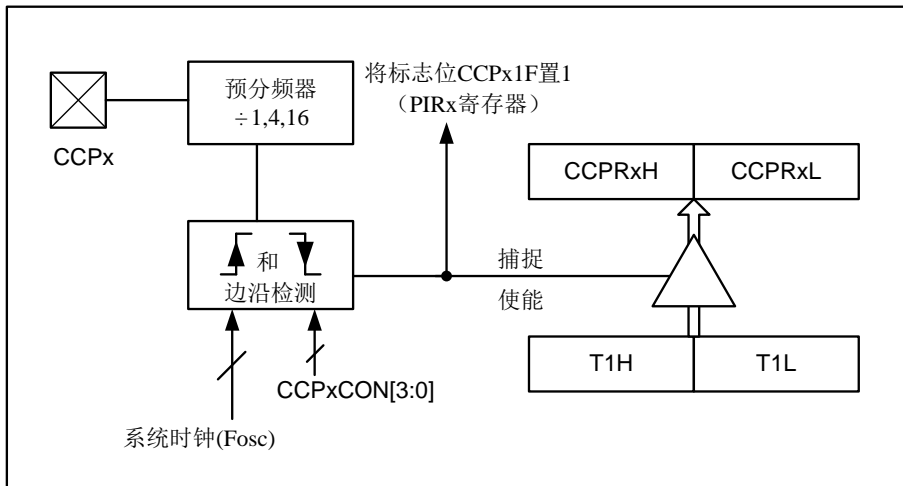
使用注意：

- 1、在捕捉模式下，CCPx引脚必须由相应的方向控制器设定为输入方式。
- 2、CCPxCON寄存器的CCPxM[3:0]设置的是预分频器，关闭CCP模块或者CCP模块不在捕捉模式，预分频计数器将会被清零。为避免错误中断，可在改变预分频比前通过清零 CCPxCON寄存器来关闭CCP模块。如果需要中途改变预分频器的分频比，建议使用以下程序片段：

```
BCF      STATUS, RP0      ;BANK0
CLRF    CCPxCON          ;关闭CCPx模块
MOVLW   NEW_CAPT_PS     ;选取新的分频比（1:1、1:4、1:16）
MOVWF   CCPxCON         ;赋予CCPxCON寄存器，并打开CCPx模块
```

- 3、当一个捕捉事件发生后，硬件自动将CCPx的中断标志位CCPxIF置1，表示产生了一次CCPx捕捉中断。CCPxIF位必须用软件重新清零。当CCPRx寄存器中的值还未被程序读取，而又发生了另一个新的捕捉事件时，原先的值将被新的值覆盖掉。
- 4、在捕捉模式下，Timer1必须运行在定时器模式或同步计数器模式。

捕捉模式工作原理图



CCPx引脚上发生变化时，CCPRxH:CCPRxL捕捉Timer1计数寄存器的16位值，PIRx寄存器的中断标志位CCPxIF被置1。如果在CCPRxH和CCPRxL寄存器的值被读出之前又发生另一次捕捉，那么原来的捕捉值会被新捕捉值覆盖。

9.5.2 比较模式

输出比较模式，适用于从引脚上输出不同宽度的矩形正脉冲、负脉冲、延时驱动信号、可控硅驱动信号、步进电机驱动信号等。

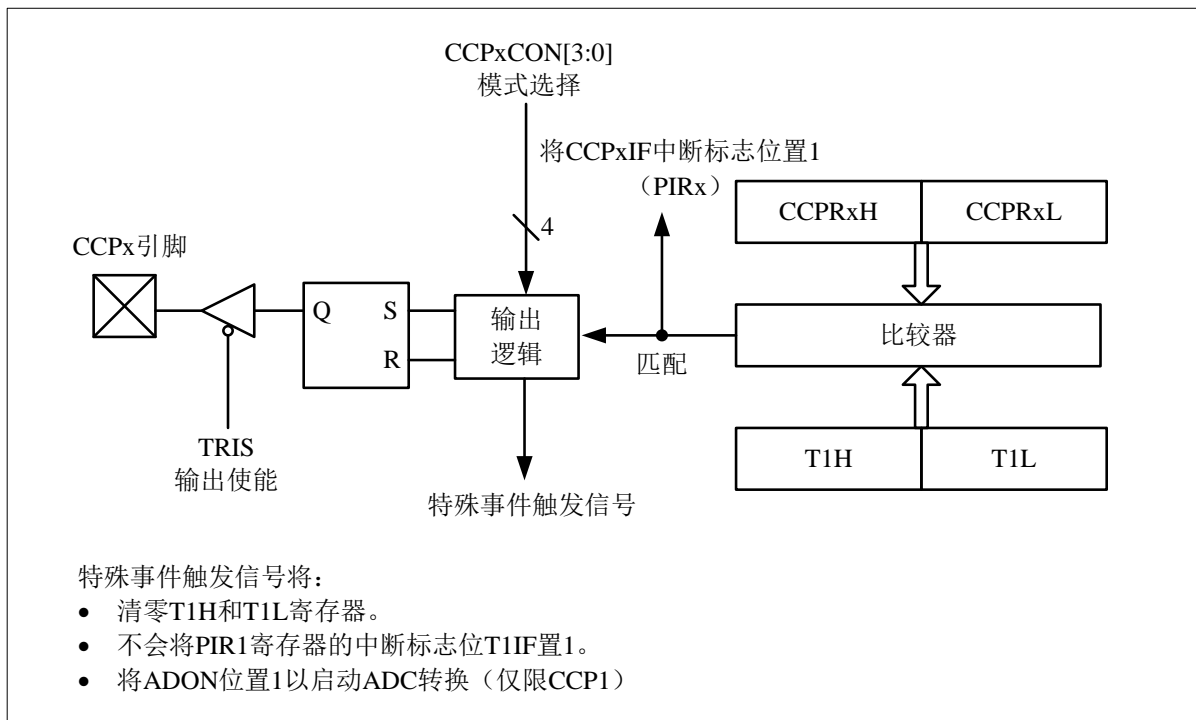
在比较模式下，CCPRxH:CCPRxL寄存器对成为了Timer1的周期寄存器，一旦Timer1计数寄存器对与CCPRxH和CCPRxL寄存器对发生匹配，Timer1计数寄存器对在Timer1时钟的下一个上升沿复位，CCPx模块根据CCPxM[3:0]控制位的配置进行相应操作：

- CCPx 引脚输出翻转电平
- CCPx 引脚输出高电平
- CCPx 引脚输出低电平
- 仅产生软件中断
- 产生特殊事件触发信号

所有比较模式都能产生CCP中断。

- 使用注意：**
- 1、当选择产生特殊事件触发信号时，如果ADC被使能，则启动一次ADC转换(仅限于CCP1)。在此模式下CCPx模块不会对CCPx引脚进行控制。
 - 2、在比较模式下，CCPx引脚必须由相应寄存器设定为输出模式，以便作为比较器的输出端使用。
 - 3、应该注意的是，如果对控制寄存器CCPxCON进行重新赋值，将会迫使CCPx引脚输出一个默认的低电平，而这并非是正常的比较输出结果。
 - 4、在比较模式下，Timer1必须运行在定时器模式或同步计数器模式下。

图13-2 比较模式工作原理图



9.5.3 PWM 模式

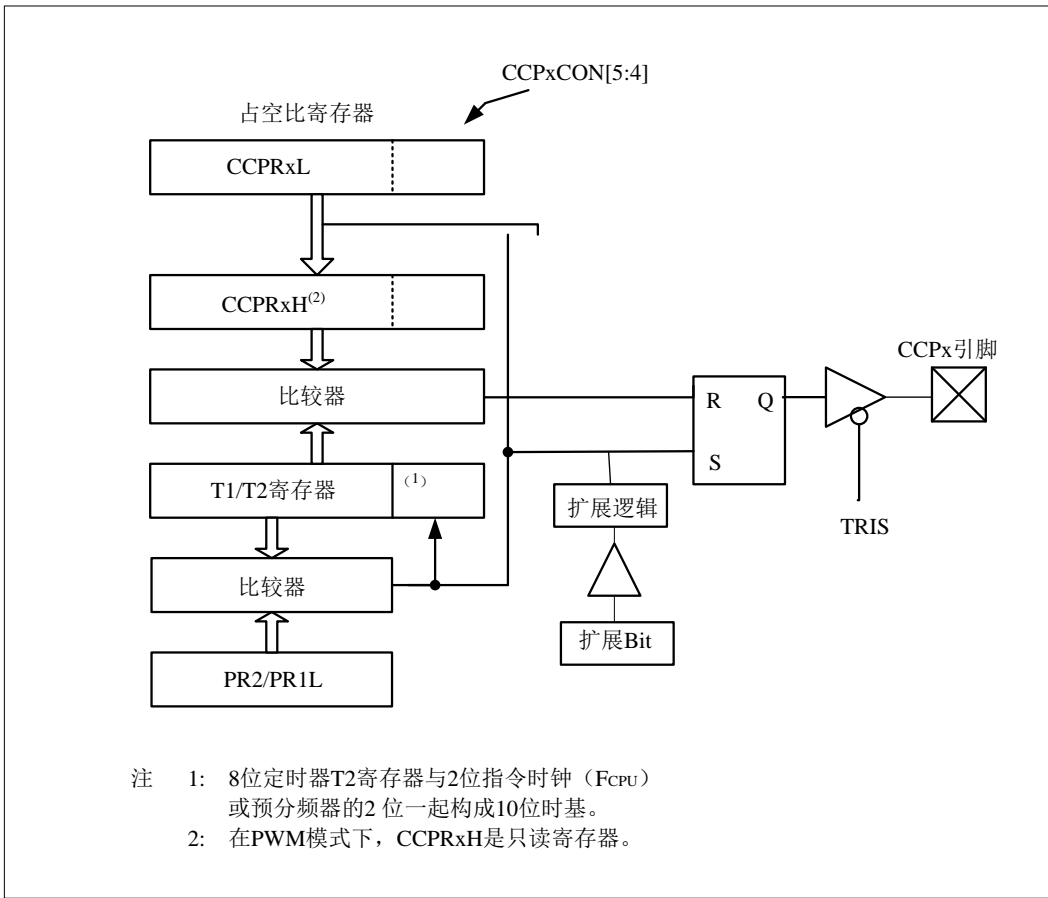
脉宽调制，PWM(pulse width modulation)输出工作模式，适用于从引脚上输出脉冲宽度随时可调的PWM信号。例如，实现直流电动机调速、简易D/A转换器、步进电机的变频控制等。

9.5.3.1 PWM不选择扩展

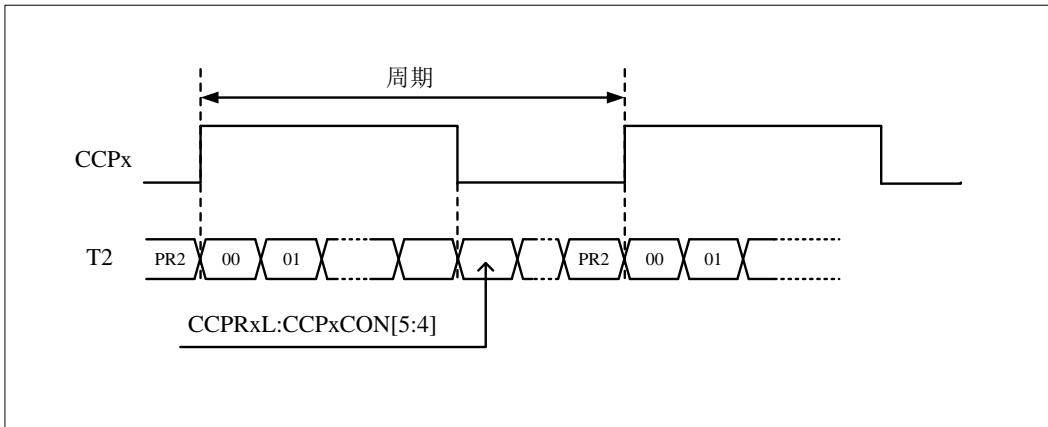
一、PWM时钟源为Timer2

在PWM模式下，当Timer2计数寄存器中的值与PR2寄存器中的值发生匹配时，在下一个计数时钟Timer2计数寄存器被清零，CCPx引脚被置1（如果PWM占空比为0%，CCPx引脚将不会被置1），PWM占空比值从CCPRxL锁存到CCPRxH（在Timer2计数寄存器中的值与PR2寄存器中的值发生匹配前，占空比值不会被锁存到CCPRxH中）。

PWM模式工作原理图



PWM波形图



✓ 以下公式中，当芯片配置为 4T 模式时， $n=1$ ；当芯片配置为 2T 模式时， $n=1/2$

PWM 周期：

$$\text{PWM 周期} = [(PR2) + 1] \cdot 4 \cdot n \cdot T_{\text{sys}}$$

(Timer2 预分频值)

注： $T_{\text{sys}} = 1/F_{\text{sys}}$

PWM脉冲宽度：

$$\text{脉冲宽度} = (\text{CCPRxL:CCPxCON}[5:4]) \cdot$$

$n \cdot T_{\text{sys}} \cdot (\text{Timer2 预分频值})$

注： $T_{\text{sys}} = 1/F_{\text{sys}}$

如果脉冲宽度值比周期长，则指定的PWM引脚将保持不变。

PWM占空比：

$$\text{占空比} = \frac{(\text{CCPRxL:CCPxCON}<5:4>)}{4(PR2 + 1)}$$

当时钟模式选择配置字选择为2T时，PWM占空比由CCPRxL寄存器和CCPxCON寄存器的DCxB[1]位决定。

PWM分辨率：

$$\text{分辨率} = \frac{\text{Log}[4(PR2 + 1)]}{\text{Log}(2)} \text{ 位}$$

分辨率是PR2寄存器值的函数，当PR2为255时PWM最大分辨率为10位（时钟模式选择配置字选择为2T时，PWM最大分辨率为9位）。

注：关于 PWM 模块的使用，需注意

1. 当 PWM:T2CON[1:0] = 0 时，PWM 输出波形恒为低电平，占空比为 0%；当 PWM:T2CON[1:0] > 4(PR2+1), PWM 输出波形恒为高电平，占空比为 100%
2. 当芯片配置为 2T 模式且 Timer2 的预分频比为 1:1 时，因需保持 PWM0 (T2CON 的 BIT0 位) = 0，所以上公式计算出来的分辨率需减 1

PWM使用：

1. 设置相关TRIS位为1，禁止CCPx引脚输出
2. 设置PWM周期，输入PR2寄存器值
3. 设置CCPxCON寄存器，将CCP模块配置为PWM模式
4. 设置PWM占空比，输入CCPRxL寄存器和CCPxCON[5:4]寄存器值
5. 配置和启动Timer2
 - 将PIR1寄存器的T2IF中断标志位清零
 - 设置T2CON寄存器的T2CKPS位，选择Timer2预分频

- 将T2CON寄存器的T2ON置1，使能Timer2

6. 设置PWM输出

- 等待Timer2溢出，PIR1寄存器的T2IF位置1
- 将TRISB2或TRISB3(TRISA6或TRISA7)位清零，让CCPx引脚输出

7. 如何关闭PWM输出

- 将TRISB2或TRISB3(TRISA6或TRISA7)位置1，设引脚输入
- 设置CCPxCON寄存器的CCPxCON[3:0]设为0000，关PWM功能，CCPx用作I/O口
- 根据需要,设PB2或PB3(PA6或PA7)输出为高电平或者低电平

➤ 例：配置PWM输出38K驱动方波，采用4M/4T模式。

```

ORG      0000H          ;复位向量
GOTO     MAIN
ORG      0004H          ;中断
.....
RETFIE

MAIN:
BCF      STATUS,RP0     ;BANK0
BCF      PCON,WDTENS    ;DISABLE WDT
CLRF     OPTION         ;使用高频时钟
BSF      TRISB,2        ;CCPx口设为输入
MOVLW   D'25'
MOVWF   PR2             ;设置PR2为26US

MOVLW   H'0D'
MOVWF   CCPR1L          ;占空比高8位
MOVLW   B'00001100'    ;选PWM模块，填写占空比低两位
MOVWF   CCP1CON         ;脉冲宽度13US

BCF      PIR1,T2IF
CLRF     T2CON          ;分频比1: 1
BSF      T2CON,T2ON     ;开T2
BSF      PIE1,T2IE

BTFSS   PIR1,T2IF      ;等待T2溢出
GOTO    $-1
BCF      PIR1,T2IF      ;清除中断标志
BCF      PIE1,T2IE
BCF      TRISB,2        ;打开PWM输出，配置完成，PB2将持续输出38K
    
```

二、PWM2时钟源为Timer1

05Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PR1L	Timer1周期寄存器低字节							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR值	0	0	0	0	0	0	0	0

060h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PR1CON	PWM1T	PWM1T	PWM2T	PWM2T	T1CKPS	T1CKPS	PWMPR	PR1EN
	1	0	1	0	3	2	1	N
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR值	0	0	0	0	0	0	0	0

Bit0 PR1EN: Timer1可设周期使能位

1= Timer1为可设周期的8位Timer

0= Timer1为16位Timer

Bit1 PWMPR1: Timer1提供PWM时钟源使能位

1=CCP2的PWM的时钟源由8位可设周期的Timer1提供（当PR1EN有效时，该位可用）

0=CCP2的PWM的时钟源由Timer2提供

Bit[3-2] T1CKPS[3-2]:8位可设周期Timer1的分频比控制位的高两位

T1CKPS[3:0]	Timer1 预分频比
0000	1 : 1
0001	1 : 2
0010	1 : 4
0011	1 : 8
0100	1 : 16
0101	1 : 32
0110	1 : 64
0111	1 : 128
1xxx	1 : 256

使用注意: 将寄存器PR1CON[bit1~bit0]置1，使PWM2的时钟源选择位可设周期的8位Timer1，其他使用操作可参考由Timer2提供时钟源的PWM2的使用。

9.5.3.2 PWM选择扩展

当PWM选择扩展周期时，4个调制周期为一个输出周期，此时PWM最高可扩展到12位。也就是说当选择扩展PWM模式时，PWM将为4个波形一个周期，此时不论你在哪一个波形输出时改变PWM的数据存储器都将在下一个周期才生效。

扩展周期控制寄存器PR1CON

060h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PR1CON	PWM1T	PWM1T	PWM2T	PWM2T	T1CKPS	T1CKPS	PWMPR	PR1EN
	1	0	1	0	3	2	1	N
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR值	0	0	0	0	0	0	0	0

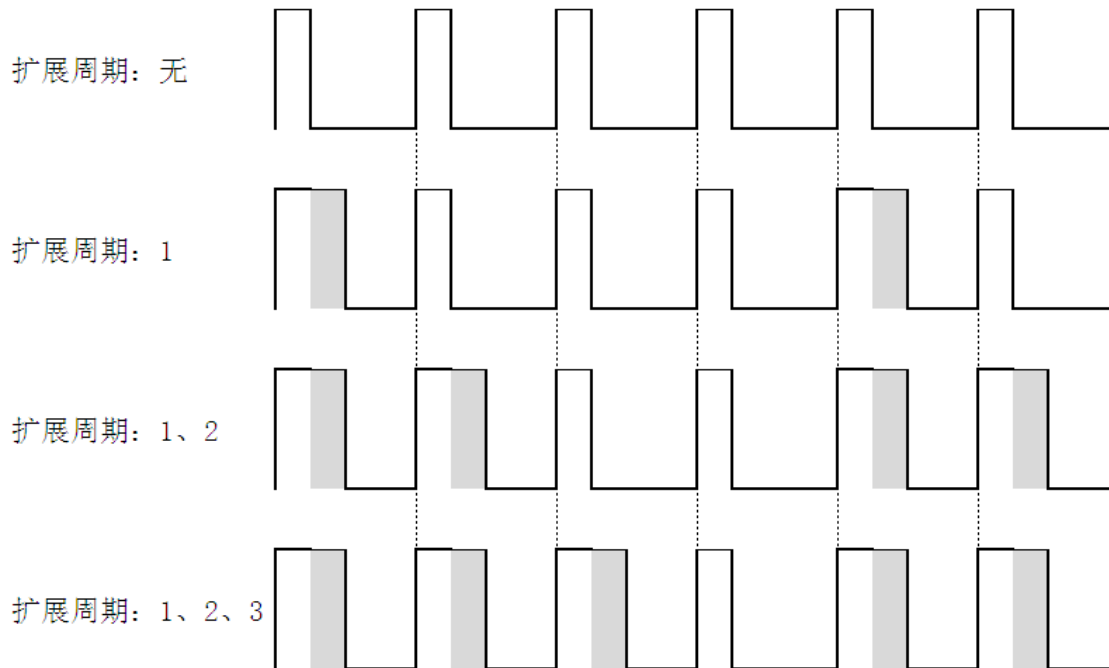
Bit[5-4] PWM2T[1-0]: PWM2的扩展周期选择位

- 00: 无扩展周期
- 01: 扩展周期为1
- 10: 扩展周期为1、2
- 11: 扩展周期为1、2、3

Bit[7-6] PWM1T[1-0]: PWM1的扩展周期选择位

- 00: 无扩展周期
- 01: 扩展周期为1
- 10: 扩展周期为1、2
- 11: 扩展周期为1、2、3

PWM扩展示意如下:



10 PWM模块

10.1 概述

- 1 组带死区互补 PWM 或 1 路独立 PWM 输出
- 提供 PWM 周期溢出中断，但中断共用同一向量入口
- 输出极性可选择
- 提供出错侦测功能可紧急关闭 PWM 输出
- PWM 工作时钟源可设定时钟分频比
- PWM 可做定时器/计数器使用

HC18P13xL 集成了 1 个 12 位 PWM 模块。支持带死区控制 PWM 输出、提供 PWM 周期溢出中断、输出极性可选择、出错侦测功能可紧急关闭 PWM 输出。通过控制相应的寄存器，PWM 模块可以产生周期和占空比分别可调整的脉宽调制波形。此外，PWM 模块还提供了有固定相位关系的 PWM 输出。

如果 EFLT 置位，PWM 输出能有 FLT 引脚的输出信号变化自动关闭。

PWM 定时器为 PWM0 提供中断源，在每个 PWM 周期都会产生中断。有不同的控制位和标志位。这样用户可以实现 PWM 周期中更改下一次循环的周期或占空比。

10.2 PWM 相关寄存器

25E	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMEN	-	EFLT	-	-	EPWM01	-	-	EPWM0
R/W	-	R/W	-	-	R/W	-	-	R/W
POR的值	-	0	-	-	0	-	-	0

bit 6 **EFLT**: FLT引脚配置位

1 = PWM故障检测输入引脚

0 = 普通I/O口或其他功能

bit 3 **EPWM01**: PWM01输出控制位

1 = PWM01输出允许

0 = PWM01 输出禁止，用作 I/O 功能

bit 0 **EPWM0**: PWM0输出控制位

1 = PWM0输出允许

0 = PWM0 输出禁止，用作 I/O 功能

当 PWMEN 清 0 后，PWM 输出立即关闭。

FLT 端口主要用于检测异常信号，快速关闭 PWM 输出，FLT 探测到故障后，由硬件执行使 PWM 输出关闭，所以当故障发生后，它可以快速响应，使得 PWM 输出无效以保护连接 PWM 的大功率器件。当使能 FLT 引脚故障检测功能后，端口禁止任何上下拉电阻功能。

如果 EFLT 位清零，则表示 FLT 端口对 PWM 定时器输出控制无效。

25Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
------	-------	-------	-------	-------	-------	-------	-------	-------

FLTM	-	-	-	-	-	-	FLT0M1	FLT0M1
R/W	-	-	-	-	-	-	R/W	R/W
POR的值	-	-	-	-	-	-	0	0

输出故障时，PWM 口输出状态：

FLT0M[1:0]：PWM0 口 FLT 故障时后，端口输出状态

00 = PWM0 输出高电平；

01 = PWM00 输出高电平，PWM01 输出低电平；

10 = PWM00 输出低电平，PWM01 输出高电平；

11 = PWM0 输出低电平；

10.2.1 PWM0 控制寄存器

HC18P13xL 包含一个 12 位 PWM 模块。PWM 模块可以产生周期和占空比分别可调的脉宽调制波形。PWMC 寄存器用于控制 PWM 模块的时钟，PWMPH/L 寄存器用于控制 PWM 输出波形的周期。PWMDH/L 寄存器用于控制 PWM 模块输出波形的占空比。

在 PWM 输出允许器件可以修改这三个寄存器，但在下一个 PWM 周期修改才会起作用。

25Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0C	-	-	FLTS	FLTC	PWM0S1	PWM0S0	CK01	CK00
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	-	-	0	0	0	0	0	0

bit[5] **FLTS**: FLT状态位

1 = PWM输出关闭，硬件置1

0 = PWM 正常状态，软件清零

bit[4] **FLTC**: FLT引脚配置位

1 = FLT为高电平时，PWM输出关闭

0 = FLT为低电平时，PWM输出关闭

bit [3:2] **PWM0S[1:0]**: PWM0和PWM01 占空比输出模式选择位

11 = PWM0和PWM01均为低有效

10 = PWM0为低有效，PWM01为高有效

01 = PWM0为高有效，PWM01为低有效

00 = PWM0和PWM01均为高有效

Bit [1:0] **CK0[1:0]**: PWM1输出控制位

11 = Fosc/16

10 = Fosc/8

01 = Fosc/4

00 = Fosc/2

注：Fosc = 32MHz

注意：

- 1、PWM0C 寄存器中的 FLTS 和 FLTC 位控制 PWM0 定时器，而寄存器中的 PWM0S, CK0[1:0] 只能影响 12 位 PWM 定时器。
- 2、PWM 输出关闭时，PWM0 和 PWM01 输出固定电平：
 - PWMxS[1:0] = 00，PWMx 和 PWMx1 均输出固定低电平；
 - PWMxS[1:0] = 01，PWMx 输出固定低电平，PWMx1 输出固定高电平；

PWMxS[1:0] =10, PWMx输出固定高电平, PWMx1输出固定低电平;

PWMxS[1:0] =11, PWMx和PWMx1均输出固定高电平;

其中x=0

3、一旦检测到FLT引脚输入有效电平, PWM输出会立即关闭, 但PWM内部计数器仍在继续运行。这样方便在FLT引脚错误去除后继续PWM输出。

4、在FLT输入信号有效期间, FLTS位无法清除。只有当FLT输入信号消失后, 才能软件清除FLTS状态位, 此时PWM恢复正常输出。

10.2.2 PWM0 周期寄存器

12 位 PWM 周期控制寄存器的高 4 位

25Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0PH	-	-	-	-	PP0.11	PP0.10	PP0.9	PP0.8
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR的值	-	-	-	-	0	0	0	0

12 位 PWM 周期控制寄存器的低 8 位

25Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0PL	PP0.7	PP0.6	PP0.5	PP0.4	PP0.3	PP0.2	PP0.1	PP0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

PWM0输出周期=[PWM0PH:PWM0PL] ×PWM时钟源

当[PWM0PH:PWM0PL]=0x00时:

如果PWM0S[1:0] =00, 不管PWM占空比为多少, PWM0输出低电平, PWM01输出低电平。

如果PWM0S[1:0] =01, 不管PWM占空比为多少, PWM0输出低电平, PWM01输出高电平。

如果PWM0S[1:0] =10, 不管PWM占空比为多少, PWM0输出高电平, PWM01输出低电平。

如果PWM0S[1:0] =11, 不管PWM占空比为多少, PWM0输出高电平, PWM01输出高电平。

10.2.3 PWM0 占空比寄存器

12 位 PWM 脉宽控制寄存器的高 4 位

25Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DH	-	-	-	-	PD0.11	PD0.10	PD0.9	PD0.8
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR的值	-	-	-	-	0	0	0	0

12 位 PWM 脉宽控制寄存器的低 8 位

259h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DL	PD0.7	PD0.6	PD0.5	PD0.4	PD0.3	PD0.2	PD0.1	PD0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

PWM输出占空比= [PWM0DH:PWM0DL] X PWM时钟源

当[PWM0PH:PWM0PL]≤[PWM0DLH:PWM0DL],

如果PWM0S[1:0]=00, PWM0输出高电平, PWM01输出低电平

如果PWM0S[1:0]=01, PWM0输出低电平, PWM01输出低电平

如果PWM0S[1:0]=10, PWM0输出高电平, PWM01输出高电平

如果PWM0S[1:0]=11, PWM0输出低电平, PWM01输出高电平

使用注意事项:

- 1、选择PWM模块时钟源。
- 2、设置PWM周期/占空比, 先设置低位, 再设置高位。注意, 即使高位数值不变, 也要重写一次, 否则, 低位的修改无效。
- 3、通过设置PWM控制寄存器(PWM0C)的PWM0Sx位选择PWM输出模式(高电平/低电平有效)。
- 4、通过设置PWM控制寄存器(PWMEN)的EPWM0或EPWM01来允许PWM上桥或下桥输出。
- 5、如果PWM周期或者占空比需要改变, 操作流程如同步骤2和步骤3说明。修改后的重载计数器的值在下一个周期开始有效。
- 6、注意不要对PWM关键寄存器进行误操作。

10.3 死区时间

一般的, 当没有插入死区时间时, PWM01 输出波形与 PWM0 输出波形成固定相位关系。当 PWM 控制寄存器中 EPWM01 位置 1 时, PWM01 的输出波形硬件自动产生。

注意:

- 1、当 PWM0 被禁止, 如果 PWM01 被允许, 则 PWM01 仍然会有信号输出。
- 2、如果 EFLT 置位, 当 FLT 端口有效时, PWM01 和 PWM0 都输出固定电平:
如果PWM0S[1:0]=00, PWM0输出高电平, PWM01输出低电平
如果PWM0S[1:0]=01, PWM0输出低电平, PWM01输出低电平
如果PWM0S[1:0]=10, PWM0输出高电平, PWM01输出高电平
如果PWM0S[1:0]=11, PWM0输出低电平, PWM01输出高电平

HC18P13xL PWM 提供死区时间控制功能。通过写 PWM01 死区时间控制寄存器, 在 PWM0 和 PWM01 之间产生死区时间。PWM0 和 PWM01 的周期相同。

注意:

- 1、修改死区时间前, 应禁止 PWM 输出。
- 2、在 PWM0 和 PWM01 为互补波形输出时, 为了产生死区时间, 请确保 (PWM0 占空比寄存器值 x 分频系数(2,4,8,16) > PWM01 的死区时间控制。否则, PWM01 当 PWM0S[1:0]=10 时输出高电平, PWM0S[1:0]=01 时输出低电平。
- 3、PWMDT 寄存器用于控制死区时间, 它的时基为振荡器时钟, 而周期和占空比的时基由 T0CK0[1:0]控制, 最小为 2 个振荡器时钟。

4、

PWM0 死区时间控制寄存器

258h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DT	DT07	DT06	DT05	DT04	DT03	DT02	DT01	DT00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

PWM0 (n=0,1,2) 的死区时间为 PWM0DT x T_{osc}

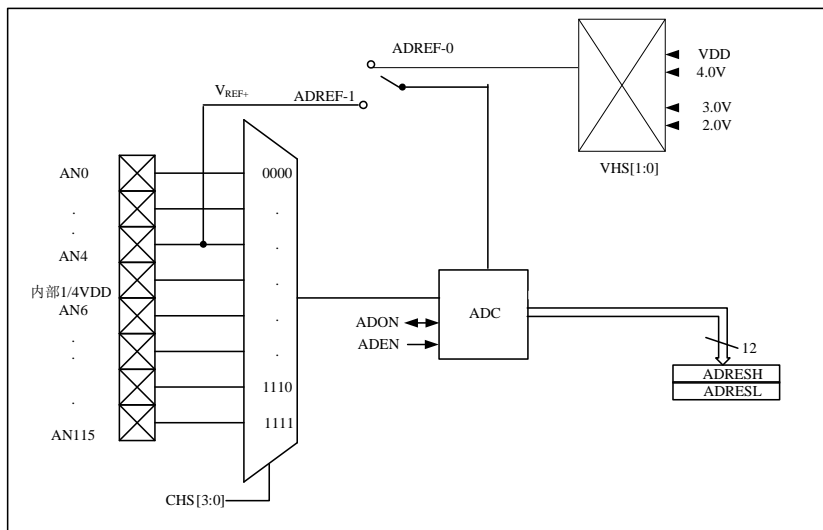
注: T_{osc} = 1/F_{osc} = 1/32MHz

11 模数转换 (ADC)

11.1 ADC 概述

HC18P13xL具有一个12位转换分辨率的模数转换器，共有15个外部模拟输入通道，1个内部电池检测通道。

ADC的等效电路：



11.2 A/D 寄存器

08Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELL	ANSEL7	ANSEL6	-	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
R/W	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W
POR值	1	1	-	1	1	1	1	1

08Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELH	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR值	1	1	1	1	1	1	1	1

ANSEL[15:0]: A/D引脚数模控制位

- 1: 模拟模式，作为模拟信号口，仅可作为AD通道的模拟输入。
- 0: 数字模式，作为数字输入或输出口。

注：1、该寄存器上电初始值为 B'1111 1111'，即作为模拟输入。无论是否应用到 AD，均需要在上电后对 IO 操作之前按需配置，否则 IO 口可能无法受控于对应的端口寄存器，状态将不确定。

2、ANSEL[4:0]对应 AN4~AN0 (PA4~PA0)，ANSEL[7:6]对应 AN7、AN6 (PA7、PA6)；

ANSEL[8]对应 AN8 (即 PA5)；

ANSEL[13:9]对应 AN13~AN11 (PB4~PB2)，ANSEL[15:14]对应 AN15、AN14 (PB7、PB6)。

11.3 A/D 控制寄存器

094h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCON0	VHS1	VHS0	CHS3	CHS2	CHS1	CHS0	ADON	ADEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

095h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCON1	ADFM	ADCS2	ADCS1	ADCS0	-	-	-	ADREF
R/W	R/W	W	W	W	-	-	-	R/W
POR的值	0	0	0	0	-	-	-	0

096h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCLK	-	-	-	-	-	ADCLK2	ADCLK1	ADCLK0
R/W	-	-	-	-	-	R/W	R/W	R/W
POR的值	-	-	-	-	-	0	0	0

ADC模拟通道选择

CHS [3:0]	模拟通道
0000	AN0
0001	AN1
0010	AN2
0011	AN3
0100	AN4
0101	内部 1/4VDD(AN5)
0110	AN6
0111	AN7
1000	AN8
1001	AN9
1010	AN10
1011	AN11
1100	AN12
1101	AN13
1110	AN14
1111	AN15

ADC参考电压选择

ADREF	VHS[1:0]	参考电压
0	00	内部2.0V
0	01	内部3.0V
0	10	内部4.0V
0	11	内部VDD
1	xx	外部参考电压

ADC数据存放格式选择

ADFM	数据格式
0	ADRESH[7:0]:ADRESL[7:4]
1	ADRESH[1:0]:ADRESL[7:0]

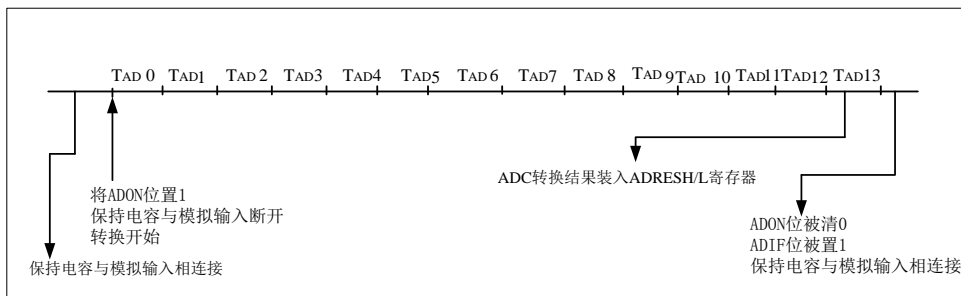
注意:

- 1、AN5为内部1/4VDD输入通道，外部没有输入引脚。可作为电池系统的电池检测器；
- 2、ADC 所采集数据，当选择存格式为左对齐时，ADC 精度只能为 12 位，高 8 位存放在 ADRESH 寄存器中，低 4 位存放在 ADRESL 寄存器高 4 位上。当选择存放格式为右对齐时，ADC 精度只能为 10 位，高 2 位存放在 ADRESH 的低 2 位上，低 8 位存放在 ADRESL 上。

11.4 AD 转换时间

ADC转换一位数据所需的时间定义为TAD，转换一次完整的12位数据需要14个TAD。为确保ADC正确转换，必须满足适当的TAD时间。

模数转换TAD 周期



ADC转换时间(TAD)与工作频率关系表

ADC 转换时间 (TAD)		系统频率 (Fsys) :4MHz
ADC 时钟源	ADCS[2:0]	典型值
Fsys	000	4us
Fsys /2	001	8us
Fsys /4	010	16us
Fsys /8	011	32us
Fsys /16	100	64us
Fsys /32	101	128us
Fsys /64	110	256us
FRC	111	视 RC 的值而定

ADCLK软件配置表

ADCLK[2:0]: AD时钟选择位 (在确定系统频率后, 选择不同的AD时钟分频对应的频率。)

000:AD转换频率在4Mhz及以上

001: AD转换频率为1M

010: AD转换频率为2M

011/1XX: AD转换频率为1M以下

系统频率/AD时钟源选择对应的ADC时钟选择为对应关系如下表所示:

Fsys (系统频率)	ADCS[2:0] AD 分频	ADCLK AD 时钟
32M	011	000
	100	010
	101	001
	11x	011/1 xx
16M	010	000
	011	010
	100	001
	101/11x	011/1 xx
8M	010	010
	011	001
	1xx	011/1 xx
4M	001	010
	010	001
	011/1xx	011/1xx
2M	000	010
	001	001
	01x/1xx	011/1 xx
1M	000	001
	001/0xx/1xx	011/1 xx
500K	xxx	011/1 xx

软件配置ADC转换时钟使用注意说明:

- 1、 Fsys为系统时钟, Fosc为RC时钟, Fcpu为指令时钟。如用户在OPTION中选择4M/2T, 对应Fosc=8M, Fsys=4M, Fcpu=2M。
- 2、 为保证ADC转换精度, 在不同ADC转换时钟下需配置不同的ADC转换时钟个数:

ADC转换时钟	ADCLK(ADCLK [2:0])	说明
2 Mhz	010	当ADC转换时钟频率为2Mhz时, 需将ADCLK[2:0]配置成010
1 Mhz	001	当ADC转换时钟频率为1Mhz时, 需将ADCLK[2:0]配置成001
<1 Mhz	011/1xx	当ADC转换时钟频率小于1 Mhz时, 需将ADCLK[2:0]配置成011/1xx(ADCLK[2:0]上电默认为111)

3、 为了加快ADC转换速度, 建议选用较快的时钟源 (ADC转换时钟不能超过4MHz)。

11.5 ADC 使用

1. 配置端口:
 - 设置 TRISA 寄存器禁止引脚输出
 - 设置 ANSEL 寄存器配置引脚为模拟输入
2. 配置ADC模块:
 - 选择 ADC 转换时钟, 设置 ADCS[2:0]、ADCLK[2:0]
 - 选择 ADC 参考电压, 设置 ADREF、ADCON0
 - 选择 ADC 输入通道, 设置 CHS[3:0]
 - 使能 ADC 模块, 设置 ADEN
3. 配置ADC中断 (可选):
 - 清零 ADC 中断标志
 - 使能 ADC 中断
 - 使能外设中断
 - 使能全局中断
4. 等待所需采集时间
5. 设置ADON为1 启动一次ADC转换
6. 通过以下方式之一等待ADC转换完成:
 - 查询 ADON 位
 - 等待 ADC 中断 (已使能中断)
7. 读取ADC结果
8. 清零ADC中断标志 (如果已使能中断则需要)

➤ 例: 配置AD, 结果保留在Bank0的NTCADHIGH、NTCADLOW中。

```

... ;其他程序
AD_TEST:
    BCF     STATUS,RP0    ;Bank0
    BSF     TRISA,0       ;设置AD口为输入
    BCF     STATUS,RP0    ;Bank0
    MOVLW  B'01010000'   ;INNER REF Fsys/32 ADRESH[7:0]
    ADRESL[7,6]
    
```

```

MOVWF    ADCON1
MOVLW    B'00000011'
MOVWF    ADCLK                                ;配置AD通道

MOVLW    B'00000001'
MOVWF    ANSELL                                ;PA0作为模拟输入
CLRF     ANSELH
BCF      ADCON0,CHS2
BCF      ADCON0,CHS1
BCF      ADCON0,CHS0
BCF      ADCON0,VHS1
BCF      ADCON0,VHS0                        ;参考电压为内部VDD
NOP
NOP
BSF      ADCON0,ADEN                        ;使能ADC
CALL     DELAY_1                            ;延时，用户可自行完成
BSF      ADCON0,ADON                        ;开始一次转换
AD_TEST_WAIT:
BTFSC    ADCON0,ADON                        ;等待转换完成
GOTO     AD_TEST_WAIT

;转换完成，保存结果
MOVF     ADRESH,W                            ;LOAD THE AD HIGH 8 BITS TO W
MOVWF    NTCADHIGH                          ;客户应用时注意BANK
MOVF     ADRESL,W                            ;LOAD THE AD LOW 8 BITS TO W
MOVWF    NTCADLOW
    
```

注意：

- 1、使能ADEN后（不是使能ADON），系统必须延迟一定的时间（视外部输入信号而定）等待ADC电路稳定；
- 2、睡眠或绿色模式下，禁止ADC并将AD参考电压设为非内部VDD以降低功耗；
- 3、为保证ADC转换精度，芯片VDD电压应高于所选ADC内部参考电压（4V/3V/2V）0.7V以上。

12 串行口通信

HC18P13xL 具有 1 个采用 UART(Universal Asynchronous Receiver/Transmitter)工作方式的全双工串行通信接口。串行口由 2 个数据缓冲器、一个移位寄存器、一个串行控制寄存器和一个波特率发生器等组成。串行口的数据缓冲器由 2 个互相独立的接收、发送缓冲器构成，可以同时发送和接收数据。发送缓冲器只能写入而不能读出，接收缓冲器只能读出而不能写入，因而两个缓冲器可以共用一个地址码。串行口的两个缓冲器共用的地址码是 23Dh。两个缓冲器统称串行通信特殊功能寄存器 SBUF。

HC18P13xL 的串行口都有 4 种工作方式，其中两种方式的波特率是可变的，另两种是固定的，以供不同应用场合选用。用户可用软件设置不同的波特率和选择不同的工作方式。主机可通过查询或中断方式对接收/发送进行程序处理，使用十分灵活。

串行口对应的硬件部分是 TxD 和 Rx D 引脚。

12.1 串行口的控制寄存器 SCON

SCON

23Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SCON	SM0/FE	SM1	SM2	REN	TB8	RB8	RXWK	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
POR 的值	0	0	0	0	0	x	0	-

- SM0/FE:

当 AUXR 寄存器的 SMOD0/AUXR.1 位为 1 时，该位用于帧错误检测。当检测到一个无效停止位时，通过 UART 接收器设置该位。它必须由软件清零。

当 AUXR 寄存器的 SMOD0/AUXR.1 位为 0 时，该位和 SM1 一起指定串行通信的工作方式，如下表所示

SM0	SM1	工作方式	功能说明	波特率
0	0	方式 0	同步移位串行方式：移位寄存器	当 UARTM0=0 时，波特率是 $F_{cpu}/12$ 当 UARTM0=1 时，波特率是 $F_{cpu}/2$
0	1	方式 1	8 位 UART，波特率可变	$(2SMOD / 32) \times (BRT \text{ 独立波特率发生器的溢出率})$
1	0	方式 2	9 位 UART	$(2SMOD / 64) \times F_{cpu}$ 系统工作时钟频率
1	1	方式 3	9 位 UART，波特率可变	$(2SMOD / 32) \times (BRT \text{ 独立波特率发生器的溢出率})$

当 BRTX12=0 时，BRT 独立波特率发生器的溢出率 = $F_{cpu}/12/(256-BRT)$

当 BRTX12=1 时，BRT 独立波特率发生器的溢出率 = $F_{cpu}/(256-BRT)$

- SM2: 允许方式 2 或方式 3 多机通信控制位
SM2=1: 且 REN 位为 1，则允许接收机处于地址筛选状态
SM2=0: 且 REN 位为 1，则禁止接收机处于地址筛选状态
- REN: 允许/禁止串行接收控制位，由软件置位
REN=1: 允许串行接收状态
REN=0: 禁止串行接收状态
- TB8: 在方式 2 或者方式 3，它为要发送的第 9 位数据，由软件置位，在方式 0、方式 1，该位无用。

- **RB8**: 在方式2或者方式3, 是接收到的第9位数据, 作为奇偶校验位或地址帧/数据帧的标志位。
- **RXWK**: 允许/禁止接收中断唤醒使能位
 RXWK = 1: 允许RXD下降沿置RXIF, 允许RXD唤醒SLEEP;
 RXWK = 0: 禁止RXD下降沿置RXIF, 禁止RXD唤醒SLEEP。

12.2 串行口数据缓冲寄存器 SBUF

SBUF

23Dh	Bit7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SBUF								

串行口缓冲寄存器(SBUF)的地址是23Dh, 实际是2个缓冲器, 写SBUF的操作完成待发送数据的□加载, 读SBUF的操作可获得已接收到的数据。两个操作分别对应两个不同的寄存器, 1个是只写寄存器, 1个是只读寄存器。

串行通道内设有数据寄存器。在所有的串行通信方式中, 在写入SBUF信号的控制下, 把数据装入相同的9位移位寄存器, 前面8位为数据字节, 其最低位为移位寄存器的输出位。根据不同的工作方式会自动将“1”或TB8的值装入移位寄存器的第9位, 并进行发送。

串行通道的接收寄存器是一个输入移位寄存器。在方式0时它的字长为8位, 其他方式时为9位。当一帧接收完毕, 移位寄存器中的数据字节装入串行数据缓冲器SBUF中, 其第9位则装入SCON寄存器中的RB8位。如果由于SM2使得已接收到的数据无效时, RB8和SBUF中内容不变。

由于接收通道内设有输入移位寄存器和SBUF缓冲器, 从而能使一帧接收完将数据由移位寄存器装入SBUF后, 可立即开始接收下一帧信息, 主机应在该帧接收结束前从SBUF缓冲器中将数据取走, 否则前一帧数据将丢失。SBUF以并行方式送往内部数据总线。

12.3 辅助寄存器 AUXR

AUXR

23Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AUXR	-	UARTEN	UARTM0	BRTR	BRTX12	S1BRS	SMOD	SMOD0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	-	0	0	0	0	x	0	0

bit 6 **UARTEN**: UART管脚控制位

- 1 = 使能UART管脚功能
- 0 = 禁止UART管脚功能, 做普通IO口

bit 5 **UARTM0**: 串口模式0的通信速度设置位

- 1 = UART串口模式0 2分频
- 0 = UART 串口模式0 12分频

bit 4 **BRTR**: 独立波特率发生器运行控制位

- 1 = 允许独立波特率发生器运行
- 0 = 不允许独立波特率发生器运行

bit 3 **BRTX12**: 独立波特率发生器计数控制位

1 = 独立波特率发生器每1个时钟计数一次

0 = 独立波特率发生器每 12 个时钟计数一次

bit 2 **S1BRS**: 串口 (UART) 的波特率发生器控制位

1 = 选择独立波特率发生器作为串口 (UART) 波特率发生器

0 = 选择定时器1作为串口 (UART) 波特率发生器

bit 1 **SMOD**: 波特率选择位

1 = 串行通信方式1、2、3的波特率加倍

0 = 串行通信方式1、2、3的波特率不加倍

bit 0 **SMOD0**: 帧错误检测有效控制位

1 = SCON寄存器中的 SM0/FE应用于FE (帧错误检测) 功能

0 = SCON寄存器中的 SM0/FE应用于SM0功能, 和SM1一起指定串行口的工作方式

12.4 独立波特率发生器寄存器 BRT

23Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BRT								
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	x	0	0

独立波特率发生器寄存器BRT用于保存重装时间常数。

12.5 自动地址识别

从机地址寄存器 SADEN

23Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SADEN								
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	x	0	0

从机地址掩码寄存器 SADDR

23Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SADDR								
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	x	0	0

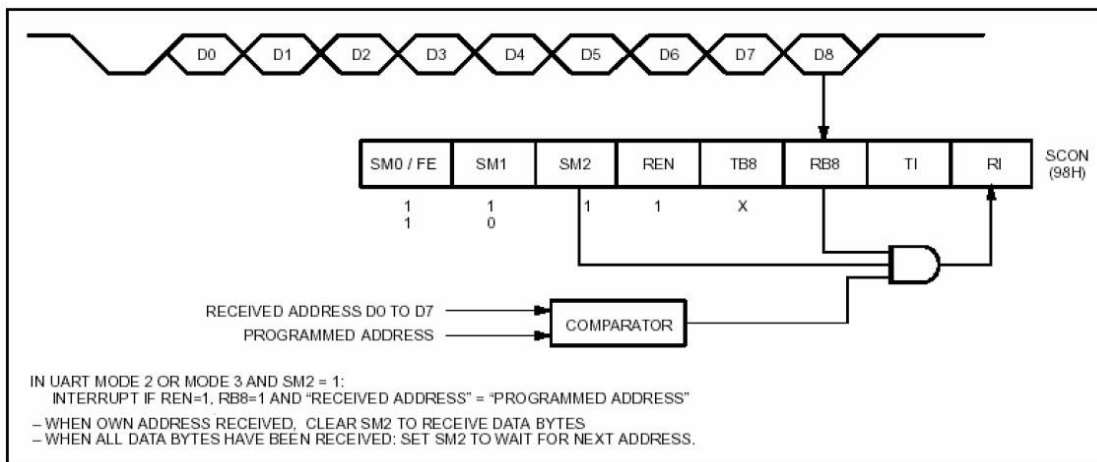
自动地址识别通过硬件比较可以让UART识别串行码流中的地址部分, 该功能免去了使用软件识别时需要大量代码的麻烦。该功能通过设定SCON的SM2位来开启。

在9位数据UART模式下, 即模式2和模式3, 收到特定地址或广播地址时自动置位接收中断(RXIF)标志, 9位模式的第9位信息为1表明接收的是一个地址而不是数据。自动地址识别功能请参考下图。在8位模式, 即模式1下, 如果SM2置位并且在8位地址与给定地址或广播地址核对一致后收到有效停止位则RXIF置位。模式0是移位寄存器模式, SM2被忽略。

使用自动地址识别功能可以让一个主机选择性的同一个或多个从机进行通讯, 所有从机可以使用广

播地址接收信息。增加了SADDR从机地址寄存器和SADEN地址掩码寄存器。

SADEN用来定义SADDR中的那些位是“无关紧要”的，SADEN掩码和SADDR寄存器进行逻辑与来定义供主机寻址从机的“给定”地址，该地址让多个从机进行排他性的识别。



下面的实例帮助理解这个方案的通用性:

从机 0

SADDR = 1100 0000

SADEN = 1111 1101

地址 = 1100 00X0

从机 1

SADDR = 1100 0000

SADEN = 1111 1110

地址 = 1100 000X

上面的例子中SADDR是相同的值，而使用SADEN数据来区分两个从机。从机0要求第0位必须为0，并忽略第1位的值；从机1要求第1位必须为0，并忽略第0位的值。从机0的唯一地址是11000010，而从机1的唯一地址是1100 0001，地址1100 0000是可以同时寻找到从机0和从机1的

下面一个更为复杂的系统可以寻址到从机1和从机2，而不会寻址到从机0:

从机 0

SADDR = 1100 0000

SADEN = 1111 1001

地址 = 1100 0XX0

从机 1

SADDR = 1110 0000

SADEN = 1111 1010

地址 = 1110 0X0X

从机 2

SADDR = 1110 0000
SADEN = 1111 1100
地址 = 1110 00XX

上面的例子中，3个从机的低3位地址不一样，从机0要求第0位必须为0，1110 0110可以唯一寻址从机0；从机1要求第1位必须为0，1110 0101可以唯一寻址从机1；从机2要求第2位必须为0，它的唯一地址是1110 0011。为了寻址到从机0和从机1而不会寻址到从机2，可以使用地址1110 0100，因为这个地址第2位是1。

每个从机的广播地址SADDR和SADEN的逻辑或，0按不需关心处理。大部分情况下，使用FF作为广播地址。

复位后，SADDR（SFR地址23Fh）和SADEN（SFR地址23Eh）值均为0，这样可以接收所有地址的信息，也就有效的禁用了自动地址识别模式，从而使该处理器运行于标准80C51的UART下。

12.6 串行口工作模式

串行通信接口有4种工作模式，可通过软件编程对SCON中的SM0、SM1的设置进行选择。其中模式1、模式2、模式3为异步通信，每个发送和接收的字符都带有1个启动位和1个停止位。在模式0中，串行口被作为1个简单的移位寄存器应用。

12.6.1 串行口工作模式0：同步移位寄存器

在模式0状态，串行通信接口工作在同步移位寄存器模式，当串行口模式0的通信速度设置位UARTM0/AUXR.5 = 0时，其波特率固定为Fcpu/12。当串行口模式0的通信速度设置位UARTM0/AUXR.5 = 1时，其波特率固定为Fcpu/2。串行口数据由RxD端输入，同步移位脉冲（SHIFTCLOCK）由TxD输出，发送、接收的是8位数据，低位在先。

模式0的发送过程：当主机执行将数据写入发送缓冲器SBUF指令时启动发送，串行口即将8位数据以Fcpu/12或Fcpu/2(由UARTM0/AUXR.5确定是12分频还是2分频)的波特率从RxD管脚输出(从低位到高位)，发送完中断标志TXIF置"1"，TxD管脚输出同步移位脉冲（SHIFTCLOCK）。波形如图8-1中“发送”所示。

当写信号有效后，相隔一个时钟，发送控制端SEND有效(高电平)，允许RxD发送数据，同时允许TxD输出同步移位脉冲。一帧(8位)数据发送完毕时，各控制端均恢复原状态，只有TXIF保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将TXIF清0。

模式0接收过程：模式0接收时，复位接收中断请求标志RXIF，即RXIF=0，置位允许接收控制位REN=1时启动串行模式0接收过程。启动接收过程后，RxD为串行输入端，TxD为同步脉冲输出端。串行接收的波特率为Fcpu/12或Fcpu/2(由UARTM0/AUXR.5确定是12分频还是2分频)。其时序图如图8-1中“接收”所示。

当接收完成一帧数据(8位)后，控制信号复位，中断标志RXIF被置"1"，呈中断申请状态。当再次接收时，必须通过软件将RXIF清0工作于模式0时，必须清0多机通信控制位SM2，使不影响TB8位和RB8位。由于波特率固定为Fcpu/12或Fcpu/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串行口工作模式0的示意图如图8-1所示

由示意图中可见，由TX和RX控制单元分别产生中断请求信号并置位TXIF=1或RXIF=1，经“或门”送主机请求中断，所以主机响应中断后必须软件判别是TXIF还是RXIF请求中断，必须软件清0中断请求标志位TXIF或RXIF。

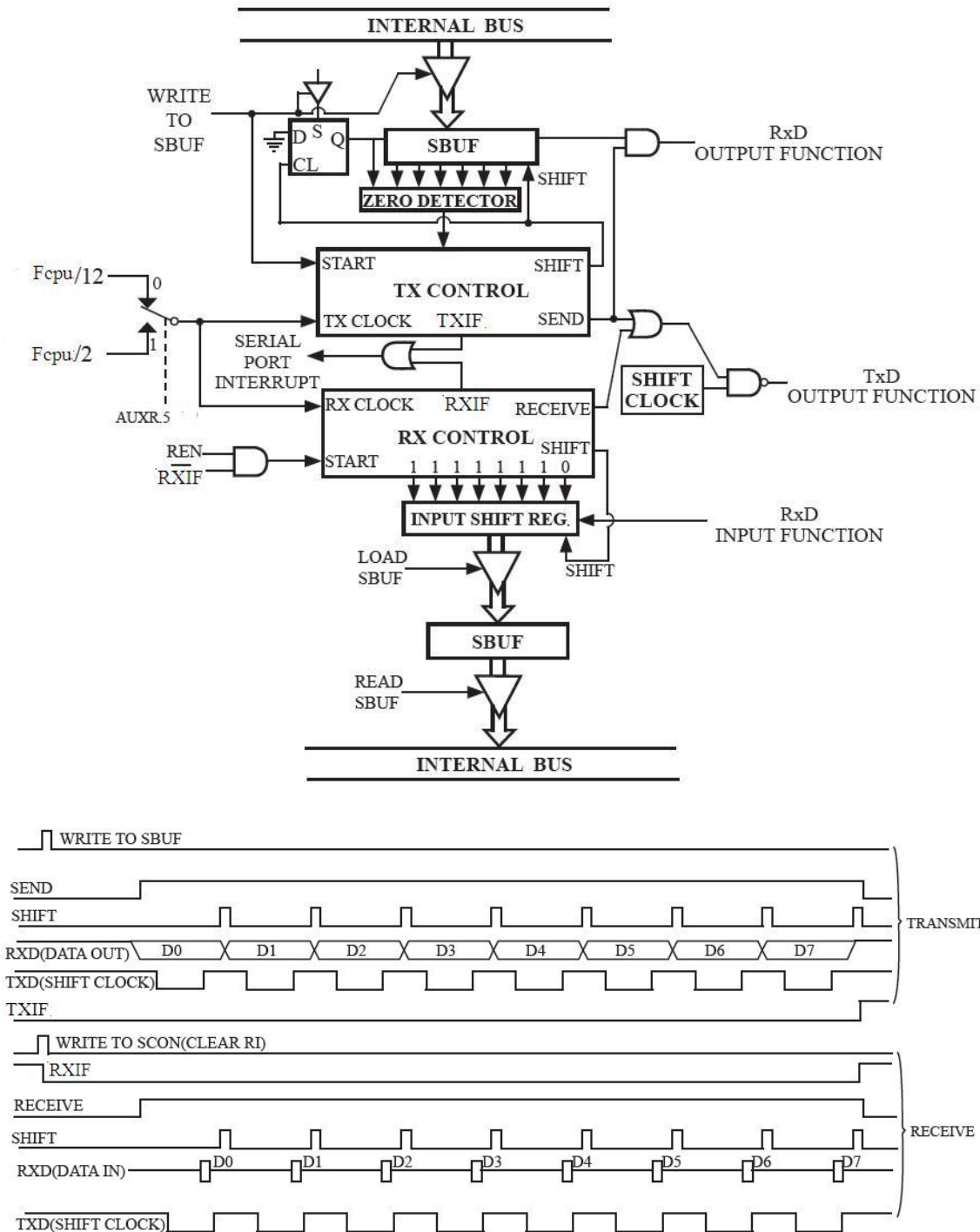


图8-1 串行口1模式0功能结构及时序示意图

12.6.2 串行口工作模式1: 8位UART, 波特率可变

当软件设置SCON的SM0、SM1为“01”时, 串行口则以模式1工作。此模式为8位UART格式, 一帧信息为10位: 1位起始位, 8位数据位(低位在先)和1位停止位。波特率可变, 即可根据需要进行设置。TxD/P3.1为发送信息, RxD/P3.0为接收端接收信息, 串行口为全双工接受/发送串行口。

图8-2为串行模式1的功能结构示意图及接收/发送时序图

模式1的发送过程: 串行通信模式发送时, 数据由串行发送端TxD输出。当主机执行一条写“SBUF”的指令就启动串行通信的发送, 写“SBUF”信号还把“1”装入发送移位寄存器的第9位, 并通知TX控制单

元开始发送。发送各位的定时是由16分频计数器同步。

移位寄存器将数据不断右移送TxD端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第9位“1”，在它的左边各位全为“0”，这个状态条件，使TX控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位TXIF，即TXIF=1，向主机请求中断处理。

模式1的接收过程：当软件置位接收允许标志位REN，即REN=1时，接收器便以选定波特率的16分频的速率采样串行接收端口RxD，当检测到RxD端口从“1”→“0”的负跳变时就启动接收器准备接收数据，并立即复位16分频计数器，将1FFH值装入移位寄存器。复位16分频计数器是使它与输入位时间同步。

16分频计数器的16个状态是将1波特率（每位接收时间）均为16等份，在每位时间的7、8、9状态由检测器对RxD端口进行采样，所接收的值是这次采样值经“三中取二”的值，即3次采样至少2次相同的值，以此消除干扰影响，提高可靠性。在起始位，如果接收到的值不为“0”（低电平），则起始位无效，复位接收电路，并重新检测“1”→“0”的跳变。如果接收到的起始位有效，则将它输入移位寄存器，并接收本帧的其余信息。

接收的数据从接收移位寄存器的右边移入，已装入的1FFH向左边移出，当起始位“0”移到移位寄存器的最左边时，使RX控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

RXIF=0;

SM2=0或接收到的停止位为1。

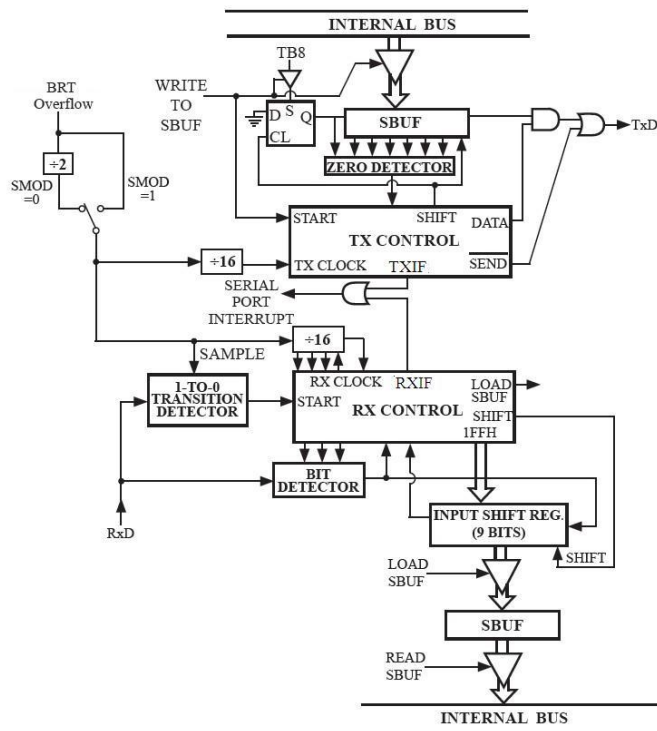
则接收到的数据有效，实现装载入SBUF，停止位进入RB8，置位RXIF，即RXIF=1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测RxD端口上的“1”→“0”的跳变，继续下一帧的接收。接收有效，在响应中断后，必须由软件清0，即RXIF=0。通常情况下，串行通信工作于模式1时，SM2设置为“0”。

串行通信模式1的波特率是可变的，可变的波特由定时器/计数器1或独立波特率发生器产生。

串行通信模式1的波特率=2SMOD/32×(定时器/计数器1溢出率或BRT独立波特率发生器溢出率)

当BRTx12 = 0时， BRT独立波特率发生器的溢出率 = Fcpu/12/(256 - BRT);

当BRTx12 = 1时， BRT独立波特率发生器的溢出率 = Fcpu/(256 - BRT)



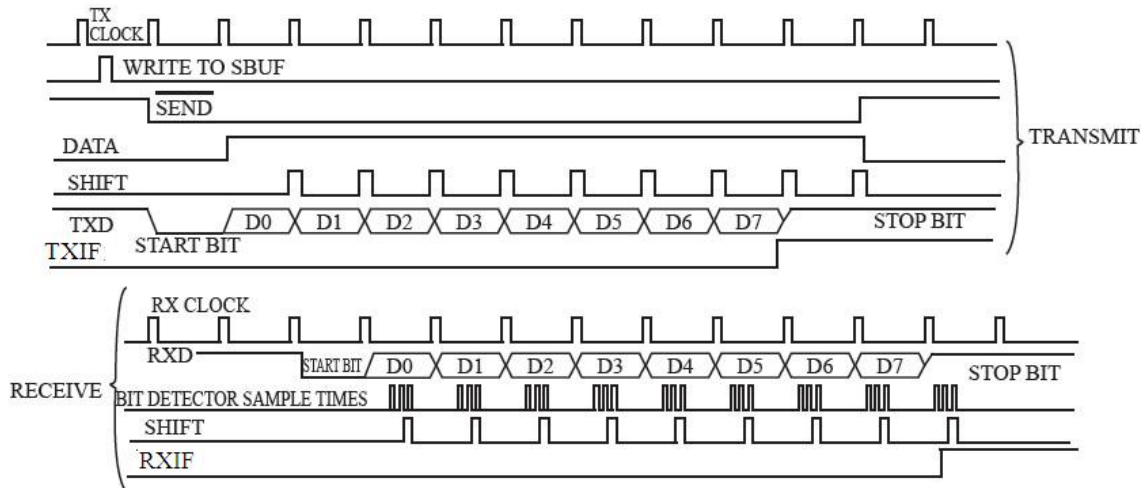


图8-2 串行口模式1功能结构示意图及接收/发送时序图

12.6.3 串行口工作模式2：9位UART，波特率固定

当SM0、SM1两位为10时，串行口工作在模式2。串行口工作模式2为9位数据异步通信UART模式，其帧的信息由11位组成：1位起始位，8位数据位(低位在先)，1位可编程位(第9位数据)和1位停止位。发送时可编程位(第9位数据)由SCON中的TB8提供，可软件设置为1或0，或者将奇/偶校验值装入TB8(TB8既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位)。接收时第9位数据装入SCON的RB8。TxD为发送端口，RxD为接收端口，以全双工模式进行接收/发送。

模式2的波特率为：

$$\text{串行通信模式2波特率} = 2\text{SMOD}/64 \times (\text{Fcpu})$$

上述波特率可通过软件对AUXR中的SMOD位进行设置，当SMOD=1时，选择1/32(Fcpu)；当SMOD=0时，选择1/64(Fcpu)，故而称SMOD为波特率加倍位。可见，模式2的波特率基本上是固定的。

图8-3为串行通信模式2的功能结构示意图及其接收/发送时序图。

由图8-3可知，模式2和模式1相比，除波特率发生源略有不同，发送时由TB8提供给移位寄存器第9数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

RXIF=0

SM2=0或者SM2=1，并且接收到的第9数据位RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入SBUF和RB8中，并置位RXIF=1，向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位RXIF。无论上述条件满足与否，接收器又重新开始检测RxD输入端口的跳变信息，接收下一帧的输入信息。

在模式2中，接收到的停止位与SBUF、RB8和RXIF无关。

通过软件对SCON中的SM2、TB8的设置以及通信协议的约定，为多机通信提供了方便。

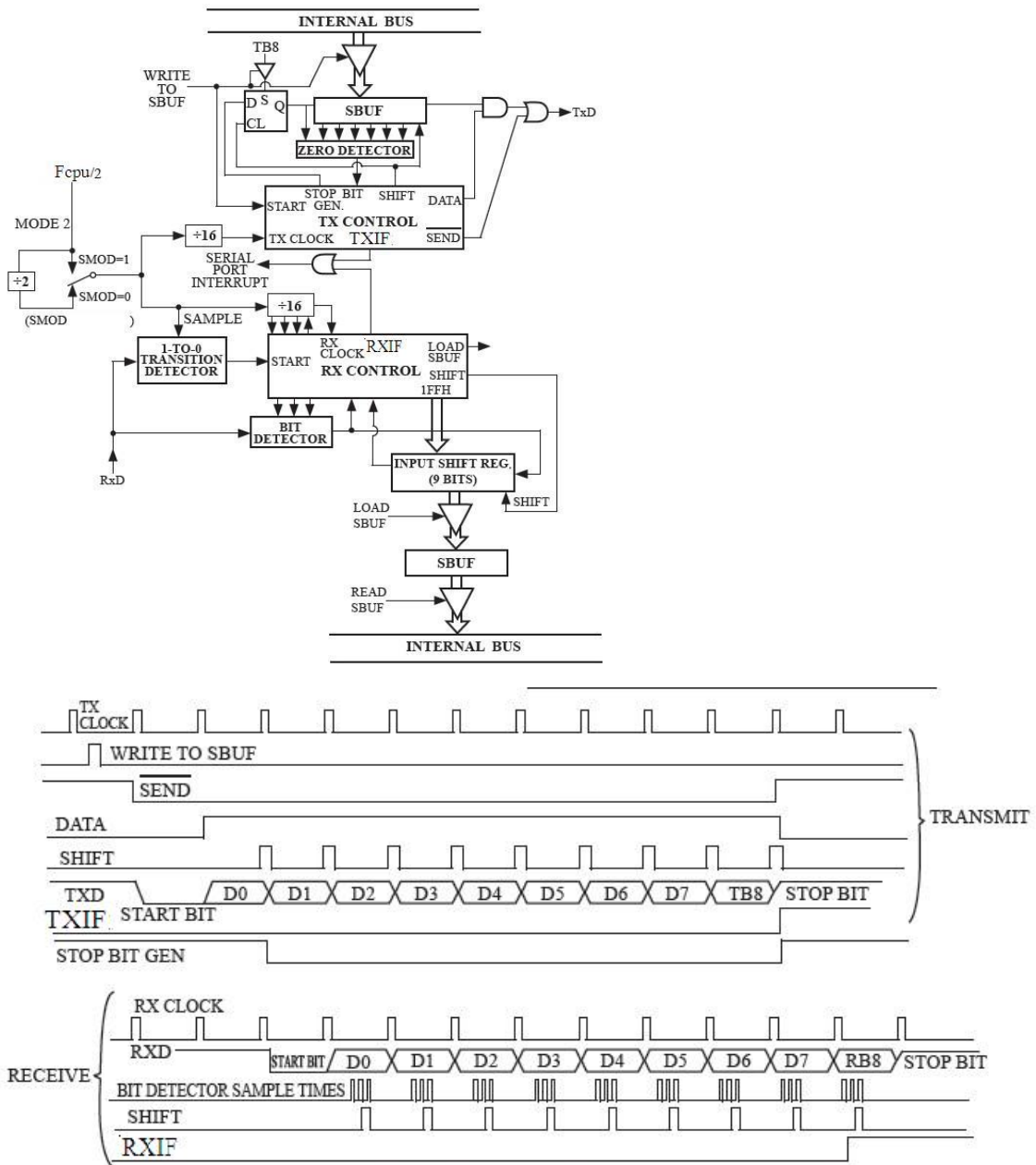


图8-3 串行口模式2功能结构示意图及接收/发送时序图

12.6.4 串行口工作模式3：9位UART，波特率可变

当SM0、SM1两位为“11”时，串行口工作在模式3。串行通信模式3为9位数据异步通信UART模式，其一帧的信息由11位组成：1位起始位，8位数据位(低位在先)，1位可编程位(第9位数据)和1位停止位。发送时可编程位(第9位数据)由SCON中的TB8提供，可软件设置为1或0，装入TB8(TB8既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位)。接收时第9位数据装入SCON的RB8。TxD为发送端口，RxD为接收端口，以全双工模式进行接收/发送。

模式3的波特率为：

串行通信模式3波特率=2SMOD/32×(BRT独立波特率发生器的溢出率)

当BRTx12 = 0时， BRT独立波特率发生器的溢出率 = Fcpu/12/(256 - BRT)；

当BRTx12 = 1时， BRT独立波特率发生器的溢出率 = Fcpu/(256 - BRT)

可见，模式3和模式1一样，其波特率可通过软件对定时器/计数器1或独立波特率发生器的设置进行波特率的选择，是可变的。

图8-4为串行口工作模式3的功能结构示意图及其接收/发送时序图

由图8-4可知，模式3和模式1相比，除发送时由TB8提供给移位寄存器第9数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

RXIF=0

SM2=0或者**SM2=1**，并且接收到的第9数据位**RB8=1**。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入SBUF和RB8中，并置位RXIF=1，向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位RXIF。无论上述条件满足与否，接收器又重新开始检测RxD输入端口的跳变信息，接收下一帧的输入信息。在模式3中，接收到的停止位与SBUF、RB8和RXIF无关。通过软件对SCON中的SM2、TB8的设置以及通信协议的约定，为多机通信提供了方便。

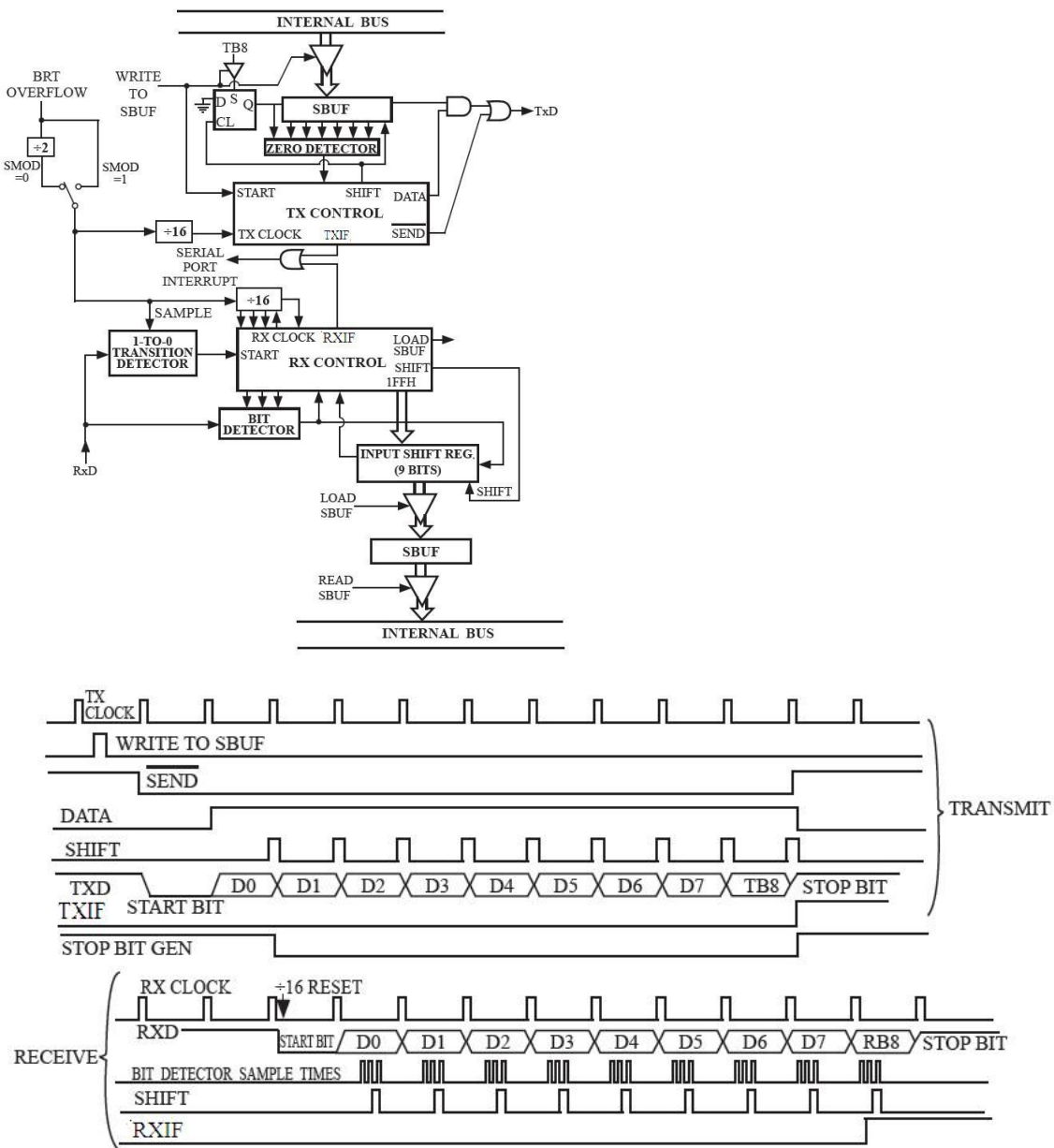


图8-4 串行口模式3功能结构示意图及接收/发送时序图

12.7.串口通信中波特率的设置

此系列单片机串行通信的波特率随所选工作模式的不同而异，对于工作模式0和模式2，其波特率与系统时钟频率Fcpu和AUXR中的波特率选择位SMOD有关，而模式1和模式3的波特率除与Fcpu和AUXR位有关外，还与定时器/计数器1或BRT独立波特率发生器设置有关。通过对定时器/计数器1或BR独立波特率发生器的设置，可选择不同的波特率，所以这种波特率是可变的。

串行通信模式0，其波特率与系统时钟频率SYSclk有关。

当模式0的通信速度设置位UARTM0/AUXR.5 = 0时，其波特率 = Fcpu/12。

当模式0的通信速度设置位UARTM0/AUXR.5 = 1时，其波特率 = Fcpu/2。

一旦Fcpu选定且UART_M0x6/AUXR.5设置好，则串行通信工作模式0的波特率固定不变。

串行通信工作模式2，其波特率除与SYSclk有关外，还与SMOD位有关。

其基本表达式为：串行通信模式2波特率=2SMOD/64×(Fcpu系统工作时钟频率)

当SMOD=1时，波特率=2/64(Fcpu)=1/32(Fcpu)；

当SMOD=0时，波特率=1/64(Fcpu)。

当Fcpu选定后，通过软件设置AUXR中的SMOD位，可选择两种波特率。所以，这种模式的波特率基本固定。

串行通信模式1和3，其波特率是可变的：

模式1、3波特率=2SMOD/32×(BRT独立波特率发生器的溢出率)

当BRTx12 = 0时， BRT独立波特率发生器的溢出率 = Fcpu/12/(256 - BRT)；

当BRTx12 = 1时， BRT独立波特率发生器的溢出率 = Fcpu/ (256 - BRT)

通过对定时器/计数器1和BRT独立波特率发生器的设置，可灵活地选择不同的波特率。在实际应用中多半选用串行模式1或串行模式3。显然，为选择波特率，关键在于定时器/计数器1和BRT独立波特率发生器的溢出率的计算。SMOD的选择，只需根据需要执行下列指令就可实现SMOD=0或1；

BCF AUXR, 1 ; 使SMOD=0

BSF AUXR, 1; 使SMOD=1

SMOD只占用电源控制寄存器AUXR的BIT2，其他各位的具体设置应根据实际情况而定。

当用户选择BRT独立波特率发生器作波特率发生器时，为选择波特率，关键在于独立波特率发生器的溢出率。当用户选择BRT独立波特率发生器作波特率发生器时，定时器/计数器1可以释放出来作为定时器/计数器/时钟输出使用。

用户在程序中如何具体使用串口1 和独立波特率发生器BRT

1.设置串口的工作模式，SCON 寄存器中的SM0 和SM1 两位决定了串口1 的4 种工作模式。

2.设置串口的波特率,使用独立波特率发生器寄存器和相应的位：

BRT 独立波特率发生器寄存器,BRTx12 位, SMOD 位

3.启动独立波特率发生器，让BRTR 位为1， BRT 独立波特率发生器寄存器就立即开始计数。

4.设置串口的中断，及打开中断相应的控制位是:PS,PSH,ES,EA

5.如要串口接收，将REN 置1 即可

如要串口发送，将数据送入SBUF 即可，

接收完成标志RXIF,发送完成标志TXIF,要由软件清0。

当串口工作在模式1 和模式3 时,计算相应的波特率需要设置的重装载数,结果送入BRT 寄存器计算自

动重装数 RELOAD (SMOD = 0, SMOD 是 AUXR 特殊功能寄存器的最高位):

1. 计算 RELOAD (以下是 SMOD = 0 时的计算公式)

a) 12T 模式的计算公式: $RELOAD = 256 - INT(F_{cpu}/Baud0/32/12 + 0.5)$

b) 1T 模式的计算公式: $RELOAD = 256 - INT(F_{cpu}/Baud0/32 + 0.5)$

计算出的 RELOAD 数直接送 BRT 寄存器

式中: INT() 表示取整运算即舍去小数, 在式中加 0.5 可以达到四舍五入的目的

SYSClk = 晶振频率

Baud0 = 标准波特率

2. 计算用 RELOAD 产生的波特率:

a) $Baud = F_{cpu}/(256 - RELOAD)/32/12$ 12T 模式

b) $Baud = F_{cpu}/(256 - RELOAD)/32$ 1T 模式

3. 计算误差

$error = (Baud - Baud0)/Baud0 * 100\%$

4. 如果误差绝对值 > 3% 要更换波特率或者更换晶体频率, 重复步骤 1-4

例: $F_{cpu} = 22.1184MHz$, $Baud0 = 57600$ (12T 模式)

1. $RELOAD = 256 - INT(22118400/57600/32/12 + 0.5)$

$= 256 - INT(1.5)$

$= 256 - 1$

$= 255$

$= 0FFH$

2. $Baud = 22118400/(256-255)/32/12 = 57600$

3. 误差等于零

13 软件LCD驱动

HC18P13xL 中内置软件 LCD 驱动，所有端口可通过 COMxEN[7:0]使能或关闭 COM 口功能。

13.1 相关寄存器

LCDCON

2B0h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LCDCON	LCDEN	RLCD1	RLCD0	FRAM	-	-	-	-
R/W	R/W	R/W	R/W	R/W	-	-	-	-
POR的值	0	0	0	0	-	-	-	-

bit[7]:LCDEN：软件 LCD 模块使能控制位

0 = 禁止

1 = 使能

Bit[6:5]:RLCD[1:0]：软件 LCD 电阻选择位

00 = 600K

01 = 300K

10 = 100K

11 = 50K

Bit4: Frame0 或 Frame1 输出使能位

0 = Frame0

1 = Frame1

COM 口使能位

2B1h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COMAEN	COMAEN7	COMAEN6	COMAEN5	COMAEN4	COMAEN3	COMAEN2	COMAEN1	COMAEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

2B2h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COMBEN	COMBEN7	COMBEN6	-	COMBEN4	COMBEN3	COMBEN2	COMBEN1	COMBEN0
R/W	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	-	0	0	0	0	0

2B3h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COMCEN	-	-	-	-	-	-	COMCEN1	COMCEN0
R/W	-	-	-	-	-	-	R/W	R/W
POR的值	-	-	-	-	-	-	0	0

2B6h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COMFEN	COMAEN7	COMAEN6	COMAEN5	COMAEN4	COMAEN3	COMAEN2	COMAEN1	COMAEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	0	0	0

Bit[7:0] COMxEN[7:0]COM 功能使能位

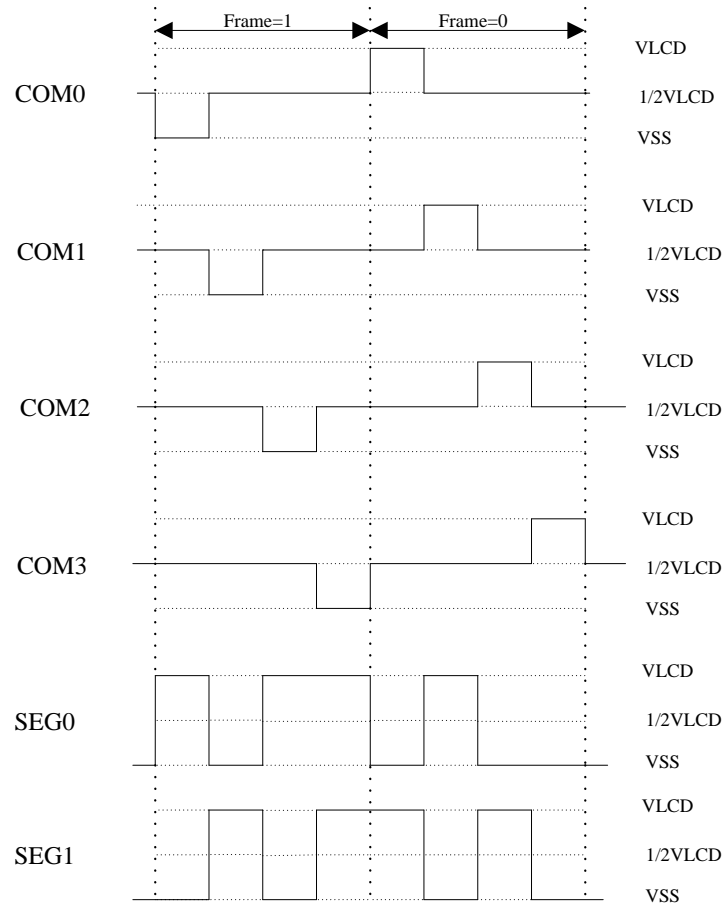
0 = 禁止对应 COMxENy 的 COM 功能，当 IO 使用

1 = 使能对应 COMxENy 的 COM 功能

说明：请根据实际需要选择适当个数 COM 功能。

13.2 软件 LCD 操作说明

- 1 设置LCD总使能，LCDEN=1；打开电阻分压电路；
 - 2 设置驱动能力，选择不同的电阻分压ISEL[1:0]；
 - 3 对IO口使能COM，COMxEN=1，这是具体设置某个IO的状态，使能LCD的模拟通道；
 - 4 SEG口设置为输出模式；
 - 5 设定frame0=0，用于确定点亮和非点亮电平（如下图中后半周期）
 - 6 设置定时开始，设置COM[3:0]=0001，设置SEG[19:0]=XXX，等待定时结束；
 - 7 设置定时开始，设置COM[3:0]=0010，设置SEG[19:0]=XXX，等待定时结束；
 - 8 设置定时开始，设置COM[3:0]=0100，设置SEG[19:0]=XXX，等待定时结束；
 - 9 设置定时开始，设置COM[3:0]=1000，设置SEG[19:0]=XXX，等待定时结束；
 - 10 设定frame0=1，用于确定点亮和非点亮电平（如下图中前半周期）
 - 11 设置定时开始，设置COM[3:0]=0001，设置SEG[19:0]=XXX，等待定时结束；
 - 12 设置定时开始，设置COM[3:0]=0010，设置SEG[19:0]=XXX，等待定时结束；
 - 13 设置定时开始，设置COM[3:0]=0100，设置SEG[19:0]=XXX，等待定时结束；
 - 14 设置定时开始，设置COM[3:0]=1000，设置SEG[19:0]=XXX，等待定时结束；
- 说明：COM[3:0] = PORTF[3:0]设定的相应数据寄存器为PORTF；SEG[19:0]同理。



16 指令表

Field	指令格式	描述	C	DC	Z	周期
移动	MOVWF F	$F \leftarrow W$	-	-	-	1
	MOVF F, D	$D \leftarrow F$ (D=0 时为 W, D=1 时为 F)	-	-	√	1
	MOVLW k	$W \leftarrow k$	-	-	-	1
算术	ADDWF F, D	$D \leftarrow W+F$ (D=0 时为 W, D=1 时为 F)	√	√	√	1
	ADDLW k	$W \leftarrow W+k$	√	√	√	1
	SUBWF F, D	$D \leftarrow F-W$ (D=0 时为 W, D=1 时为 F)	√	√	√	1
	SUBLW k	$W \leftarrow k - W$ (D=0 时为 W, D=1 时为 F)	√	√	√	1
	DAW	W 寄存器值进行 BCD 调整	√	√	-	1
	INCF F, D	$D \leftarrow F+1$ (D=0 时为 W, D=1 时为 F)	-	-	√	1
	DECF F, D	$D \leftarrow F-1$ (D=0 时为 W, D=1 时为 F)	-	-	√	1
逻辑	ANDWF F, D	$D \leftarrow W$ 与 F (D=0 时为 W, D=1 时为 F)	-	-	√	1
	ANDLW k	$W \leftarrow W$ 与 k	-	-	√	1
	IORWF F, D	$D \leftarrow W$ 或 F (D=0 时为 W, D=1 时为 F)	-	-	√	1
	IORLW k	$W \leftarrow W$ 或 k	-	-	√	1
	XORWF F, D	$D \leftarrow W$ 异或 F (D=0 时为 W, D=1 时为 F)	-	-	√	1
	XORLW k	$W \leftarrow W$ 异或 k	-	-	√	1
	COMF F, D	$D \leftarrow F$ 取反 (D=0 时为 W, D=1 时为 F)	-	-	√	1
处理	SWAPF F, D	$D[7:4,3:0] \leftarrow F[3:0,7:4]$ (D=0 时为 W, D=1 时为 F)	-	-	-	1
	RRF F, D	$D \leftarrow F$ 带进位右移 (D=0 时为 W, D=1 时为 F)	√	-	-	1
	RLF F, D	$D \leftarrow F$ 带进位左移 (D=0 时为 W, D=1 时为 F)	√	-	-	1
	CLRW	$W \leftarrow 0$	-	-	√	1
	CLRF F	$F \leftarrow 0$	-	-	√	1
	CLRWD	清零看门狗定时器, 影响 TO, PD 位	-	-	-	1
	BCF F, d	$F[d] \leftarrow 0$ (0≤d≤7)	-	-	-	1
	BSF F, d	$F[d] \leftarrow 1$ (0≤d≤7)	-	-	-	1
分支	INCFSZ F, D	$D \leftarrow F+1$ (D=0 时为 W, D=1 时为 F), 如果 D=0 则跳过下一句	-	-	-	1(2)
	DECFSZ F, D	$D \leftarrow F-1$ (D=0 时为 W, D=1 时为 F), 如果 D=0 则跳过下一句	-	-	-	1(2)
	BTFSC F, d	如果 $F[d]=0$ (0≤d≤7) 则跳过下一句	-	-	-	1(2)
	BTFSS F, d	如果 $F[d]=1$ (0≤d≤7) 则跳过下一句	-	-	-	1(2)
	GOTO k	无条件跳转	-	-	-	2
	CALL k	调用子程序	-	-	-	2
其他	RETURN	从子程序返回	-	-	-	2
	RETFIE	从中断返回, 并置位 GIE	-	-	-	2
	RETLW k	$W \leftarrow k$, 带参数返回	-	-	-	2
	NOP	空操作	-	-	-	1
	SLEEP	进入待机模式, 影响 TO, PD 位	-	-	-	1

14 开发工具

14.1 OTP 烧录器（PM18-4.0）

- HC-PM18-4.0: 支持HC18系列MCU大批量的脱机烧录

注:

详情请参考 HC-PM18 用户手册。如有更新请到网站下载最新资料。

<http://www.holychip.cn>

14.2 HC-IDE

Holychip 8位单片机的集成开发环境HC-IDE包括编译器。

- HC-IDE: HC-IDE V3.0.x.0(支持汇编/C语言)

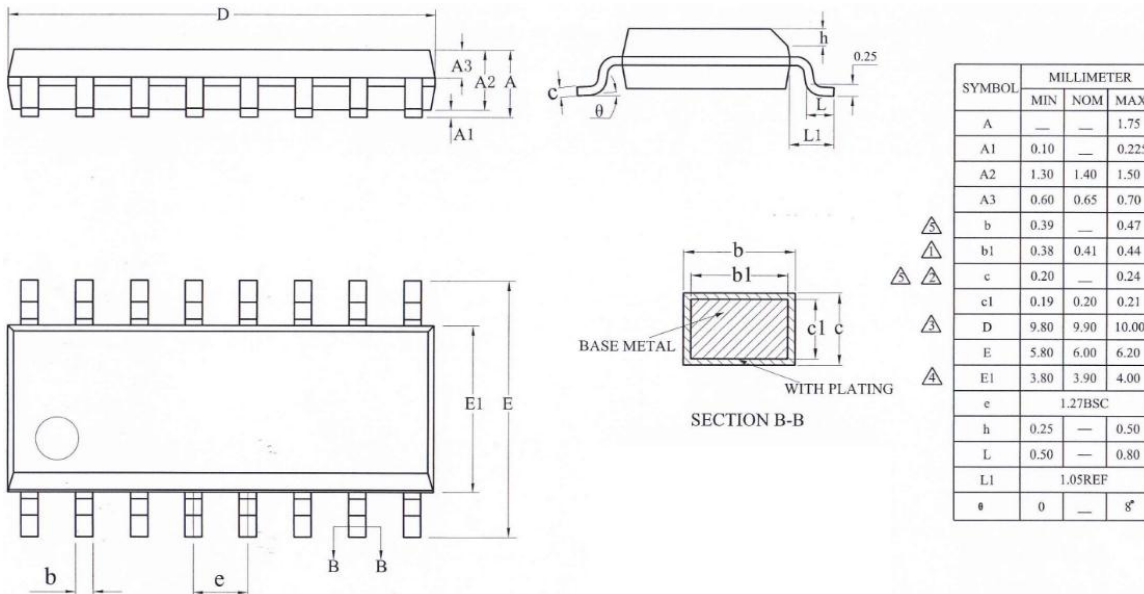
注:

1、详情请参考 HC-IDE 用户手册。

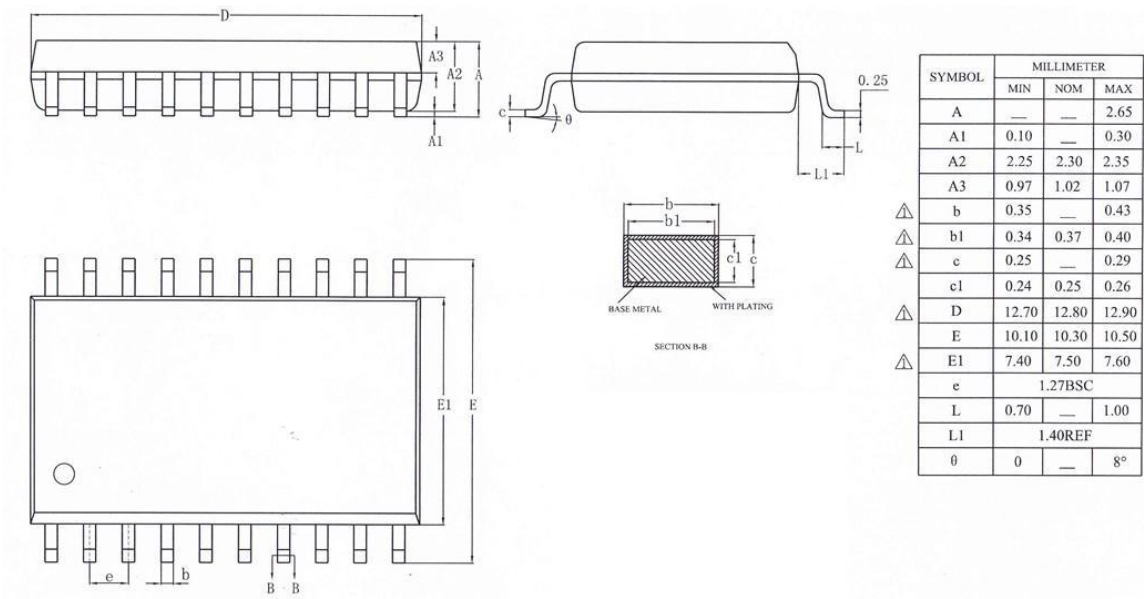
2、IDE 版本请关注芯圣官网: <http://www.holychip.cn/>。

15 封装尺寸

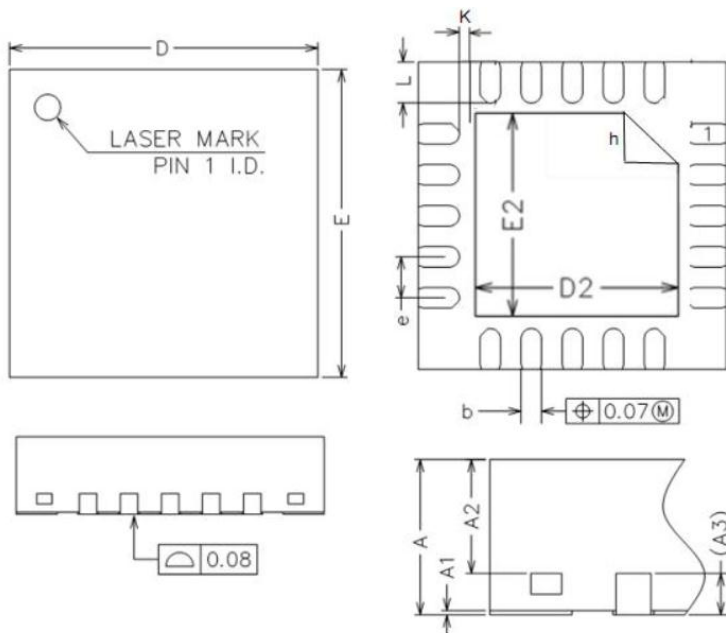
15.1 SOP16



15.2 SOP20

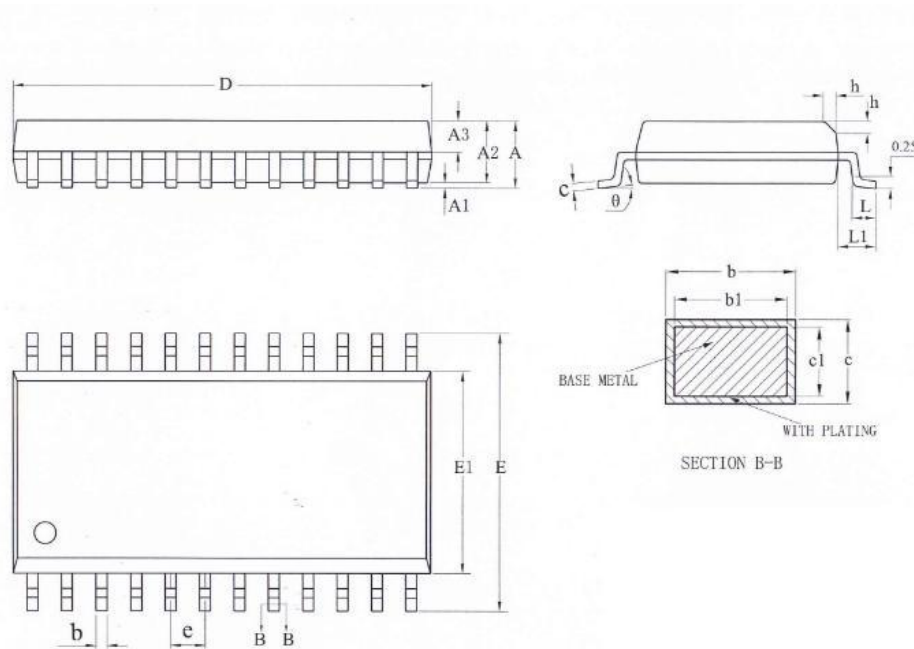


15.3 QFN20



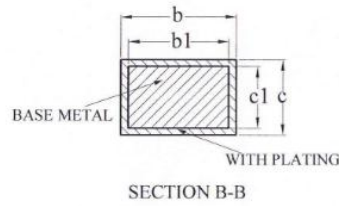
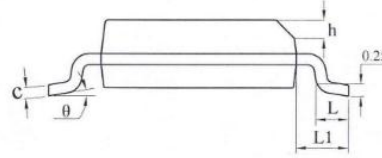
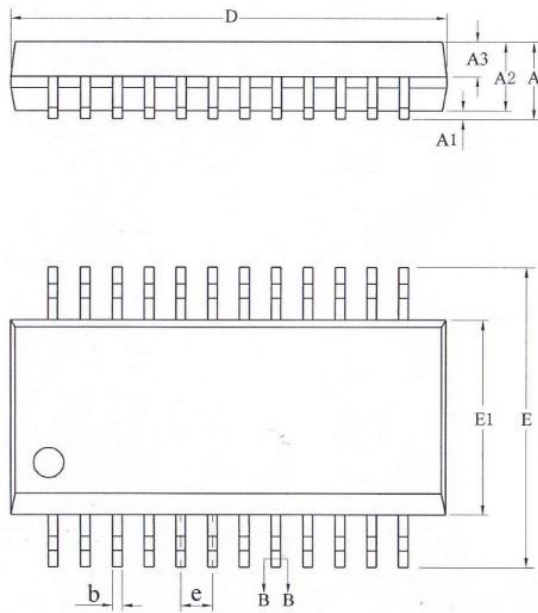
SYMBOL	MIN	MON	MAX
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A2	0.50	0.55	0.60
A3	0.20REF		
b	0.15	0.20	0.25
D	2.90	3.00	3.10
E	2.90	3.00	3.10
D2	1.60	1.65	1.70
E2	1.60	1.65	1.70
e	0.30	0.40	0.50
K	0.175REF		
L	0.35		0.45
h	0.25		

15.4 SOP24



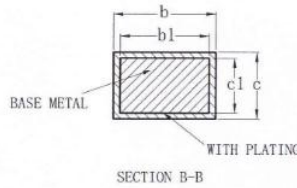
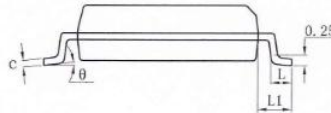
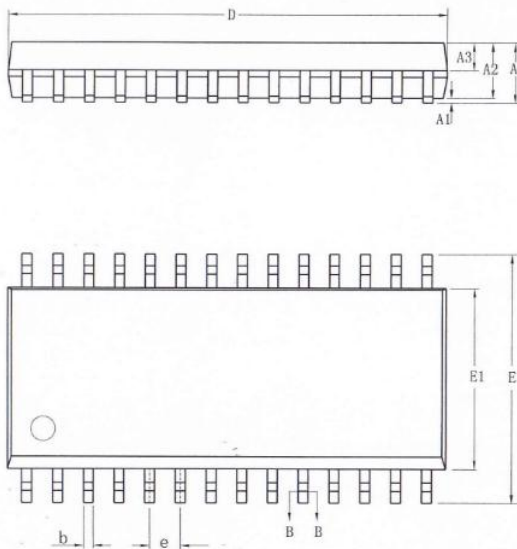
SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	2.36	2.54	2.64
A1	0.10	0.20	0.30
A2	2.26	2.30	2.35
A3	0.97	1.02	1.07
b	0.39	—	0.47
b1	0.38	0.41	0.44
c	0.25	—	0.29
c1	0.24	0.25	0.26
D	15.30	15.40	15.50
E	10.10	10.30	10.50
E1	7.40	7.50	7.60
c	1.27BSC		
L	0.70	—	1.00
L1	1.40REF		
h	0.25	—	0.75
θ	0	—	8°

15.5 SSOP24



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	—	—	1.75
A1	0.10	0.15	0.25
A2	1.30	1.40	1.50
A3	0.60	0.65	0.70
b	0.23	—	0.31
b1	0.22	0.25	0.28
c	0.20	—	0.24
c1	0.19	0.20	0.21
D	8.55	8.65	8.75
E	5.80	6.00	6.20
E1	3.80	3.90	4.00
e	0.635BSC		
h	0.30	—	0.50
L	0.50	—	0.80
L1	1.05REF		
θ	0	—	8°

15.6 SOP28



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	—	—	2.65
A1	0.10	—	0.30
A2	2.25	2.30	2.35
A3	0.97	1.02	1.07
b	0.39	—	0.47
b1	0.38	0.41	0.44
c	0.25	—	0.29
c1	0.24	0.25	0.26
D	17.90	18.00	18.10
E	10.10	10.30	10.50
E1	7.40	7.50	7.60
e	1.27BSC		
L	0.70	—	1.00
L1	1.40REF		
θ	0	—	8°

16 修改记录

版本	日期	描述
Ver1.00	2018-04	第一版
Ver1.01	2019-09	修改 PWM 中描述错误，修改 ADC 中寄存器描述； RAM 分布中映射修改； IDE 描述修改。 修改低频内部 RC 频率电压特性 OSCCON 寄存器添加 TOOSCEN 位的介绍
Ver1.02	2019-11	增加 QFN20 封装及选型

HOLYCHIP 公司保留对以下所有产品在可靠性、功能和设计方面的改进作进一步说明的权利。

HOLYCHIP 不承担由本手册所涉及的产品或电路的运用和使用所引起的任何责任，HOLYCHIP 的产品不是专门设计来应用于外科植入、生命维持和任何 HOLYCHIP 产品产生的故障会对个体造成伤害甚至死亡的领域。如果将 HOLYCHIP 的产品用于上述领域，即使这些是由 HOLYCHIP 在产品设计和制造上的疏忽引起的，用户应赔偿所有费用、损失、合理的人身伤害或死亡所直接或间接所产生的律师费用，并且用户保证 HOLYCHIP 及其雇员、子公司、分支机构和销售商与上述事宜无关。

芯圣电子

2019 年 11 月