SQL5820

数据手册

10 引脚 8 位
AD 型 OTP 单片机

目录

1 产品简述 1

1.1 特性 1
1.2 系统框图 2
1.3 MSOP10 引脚图 3
1.4 引脚说明 3
1.5 引脚电路 4

2 中央处理器（CPU） 5

2.1 存储器 5
2.1.1 程序存储器（ROM） 5
2.2 芯片配置选择表 10
2.2.1 芯片配置字 10
2.3 数据寄存器（RAM） 12
2.3.1 STATUS 寄存器 14
2.3.3 PC 寄存器 15
2.4 寻址模式 15
2.4.1 立即寻址 15
2.4.2 直接寻址 15
2.4.3 间接寻址 15
2.5 堆栈 16

3 复位 17

3.1 上电复位 17
3.2 外部复位 18
3.3 欠压复位 18
3.4 看门狗定时器复位 19
3.5 PCON 寄存器 19

4 系统时钟 20

4.1 概述 20
4.2 时钟框图 20
4.3 系统高频时钟 21
4.3.1 内部高频 RC 振荡器 21
4.3.2 外部高频时钟 21
4.4 系统低频时钟 22
4.4.1 低频晶体振荡器 23
4.4.2 低频 RC 振荡器 23

5 系统工作模式 24

5.1 模式切换举例 25
5.2 高低频模式切换 26
5.3 唤醒时间 27

5.4 OSCCON 寄存器 27

6 中断源 28

6.1 内核中断 29
6.2 外设中断 30
6.3 GIE 全局中断 32
6.4 中断保护 32
6.5 TIMER0 定时器中断 33
6.6 INT0 外部中断 33
6.7 PORT 电平变化中断 34
6.8 TIMER2 定时器中断 36
6.9 TIMER1 中断 36
6.10 AD 中断 37
6.11 CCP 中断 37
6.12 UART 中断 37
6.13 PWM 中断 38
6.14 多中断操作 38

7 I/O 口 40

7.1 I/O 口输入输出控制寄存器 40
7.2 I/O 口上拉控制寄存器 41
7.3 I/O 口下拉控制寄存器 41
7.4 I/O 驱动控制寄存器 41
7.5 I/O 口数据寄存器 42
7.6 I/O 口管脚配置寄存器 42

8 定时器/计数器 43

8.1 看门狗定时器 43
8.2 TIMER0 定时器/计数器 44
8.3 TIMER1 定时器/计数器 47
8.3.1 Timer1 控制寄存器 47
8.4 TIMER2 定时器 48
8.4.1 Timer2 控制寄存器 49
8.4.2 Timer2 计数寄存器 49
8.4.3 Timer2 周期寄存器 49
8.5 CCP 模块 50
8.5.1 捕捉模式 51
8.5.2 比较模式 52
8.5.3 PWM 模式 53

9 PWM 模块 58

9.1 PWM 特性 58
9.2 PWM 输出模式 58
9.2.1 互补输出模式 58
9.2.2 独立输出模式 58

9.3 PWM 相关寄存器 59
9.3.1 PWM 使能寄存器 59
9.3.2 PWM0 控制寄存器 59
9.3.3 PWM 模式寄存器 PWMM 60
9.3.4 PWM0 周期/占空比寄存器 61
9.3.5 死区时间 62
10 模数转换 (ADC) 63
10.1 A/D 引脚控制寄存器 63
10.2 A/D 控制寄存器 64
10.3 ADC 使用 68
11 串行口通信 69
11.1 串行口的控制寄存器 SCON 70
11.2 串行口数据缓冲寄存器 SBUF 70
11.3 辅助寄存器 AUXR 71
11.4 独立波特率发生器寄存器 BRT 72
11.5 自动地址识别 72
11.6 串行口工作模式 74
11.6.1 串行口工作模式 0：同步移位寄存器 74
11.6.2 串行口工作模式 1：8 位 UART，波特率可变 75
11.6.3 串行口工作模式 2：9 位 UART，波特率固定 77
11.6.4 串行口工作模式 3：9 位 UART，波特率可变 79
11.7 串口通信中波特率的设置 80
12 软件 LCD 驱动 83
12.1 相关寄存器 83
12.2 软件 LCD 操作说明 83
13 指令表 85
14 电性参数 86
14.1 极限参数 86
14.2 直流特性 86
14.3 交流特性 87
15 开发工具 88
15.1 OTP 烧录器（HC-PM18 5.0） 88
15.2 HC-IDE 88
16 封装信息 89
17 修改记录 90

1 产品简述

SQL5820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2K×16位一次性编程ROM(OTP-ROM)，256×8位的数据寄存器（RAM），2组双向I/O口，三个Timer定时器/计数器，一个PWM模块，两个CCP模块，一个AD模块，支持一路UART。一个8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
  - 2K×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
  - 2组双向I/O口：PORTA，PORTB
  - 最多8个双向I/O口
  - 所有端口4级驱动电流配置(除PORTB5)可直接驱动LED（4mA）
  - 最多8个可编程弱上拉/下拉口（PA、PB）
  - 所有端口具有唤醒功能的电平变化中断
  - 具体端口配置详见第7章
  - 所有端口支持软件1/2bias COM口功能（除VPP）
- ◆ 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
- ■ PWM模块
  - 1组可编程带死区控制的固定相位PWM 1*12Bit
- ■ BOR复位系统
  - 1.5V/1.8V/2.0V/2.2V/2.4V/2.7V/3.6V
- ■ 一个UART模块
- ◆ AD 模数转换器
  - 12位转换分辨率
  - 最多8个模拟输入通道（7个外部ADC输入，1个内部1/4VDD检测）
  - 内部参考电压(VDD、4V、3V、2V、1.2V)和外部参考电压
- ◆ 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： 32KHz
- ■ 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（MCLRB Reset）
  - 欠压复位（BOR）
  - 看门狗定时器复位（WDT Reset）
- ◆ 封装形式
  - MSOP10

■ 抗干扰能力
- ESD≥3000V
- EFT≥4000V

◆ 最低工作电压
- 1.5V~5.5V （Fsys= 4MHz）
- 4.5V~5.5V （Fsys =16MHz）
- 1.8V~5.5V （Fsys= 8MHz）

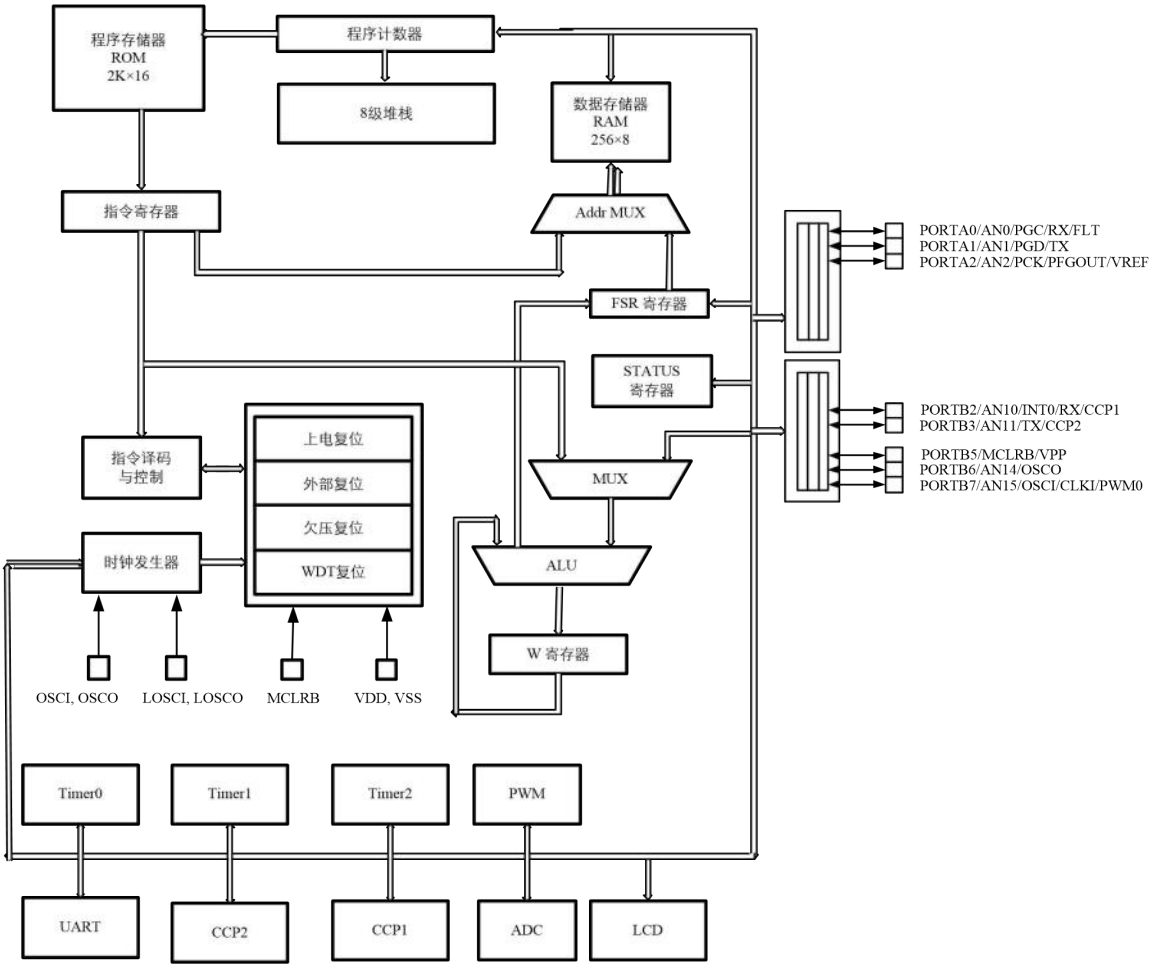
✓ 选型表

产品型号	ROM	RAM	堆栈	定时器	I/O	CCP	WEAK UP PORT	ADC	封装形式
SQL5820	2K*16	256*8	8	3	7+1	2	8	12*（7+1）	MSOP10

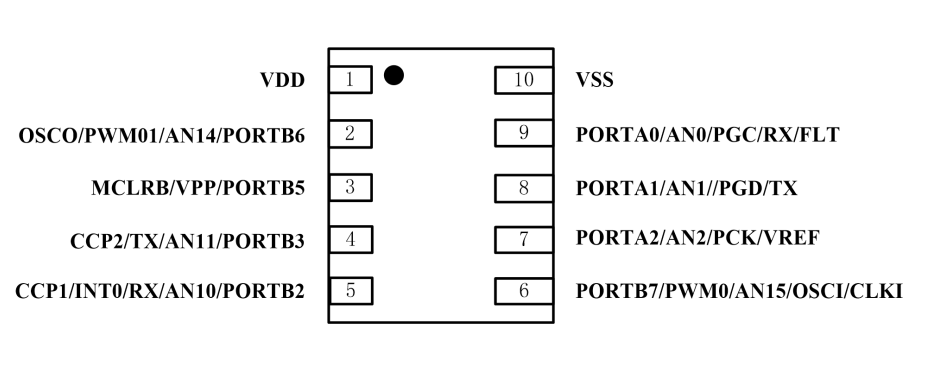
使用注意事项：

ADCON1 寄存器的低四位 Bit3~Bit0 如果全为 0，功耗会比一般情况偏大。在 sleep 模式下为保证系统的低功耗，请避免将 ADCON1 寄存器的低四位 Bit3~Bit0 全设置为 0。

1.2 系统框图



1.3 MSOP10 引脚图



1.4 引脚说明

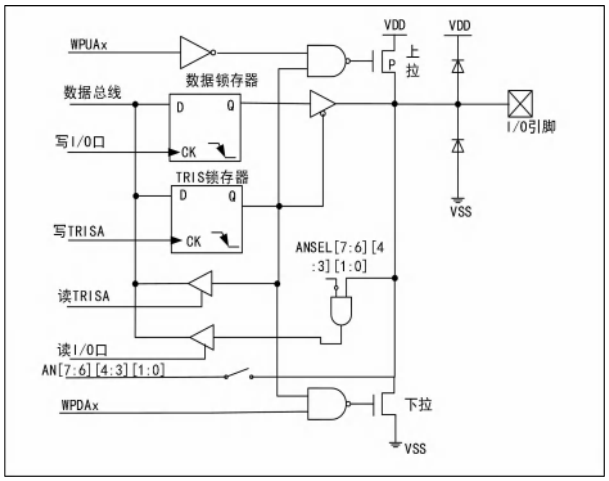
10PIN 脚位	名称	类型	说明
1	VDD	P	电源输入。
2	PORTB6 OSCO PWM01 AN14	I/O AN O AN	输入/输出口，带可编程上下拉电阻，端口电平变化中断。 高频晶体振荡器输出口。 PWM01输出口。 ADC通道14输入口。
3	PORTB5 MCLRB VPP	I I P	输入口，带可编程上下拉电阻，端口电平变化中断。 复位输入口，低电平有效。 编程高压电源输入。
4	PORTB3 AN11 CCP2 TX	I/O AN I/O O	输入/输出口，带可编程上下拉电阻，端口电平变化中断。 ADC通道11输入口。 CCP2输入/输出口。 串口数据发送端口。
5	PORTB2 CCP1 AN10 INT0 RX	I/O I/O AN I I	输入/输出口，带可编程上拉电阻，端口电平变化中断。 CCP1输入/输出口。 ADC通道10输入口。 外部中断接收端口。 串口数据接收端口。
6	PORTB7 AN15 OSCI PWM0 CLKI	I/O AN AN O I	输入/输出口，带可编程上下拉电阻，端口电平变化中断。 ADC通道15输入口。 高频晶体振荡器输入口。 PWM0输出口。 时钟输入口。
7	PORTA2 AN2 PCK	I/O AN O	输入/输出口，带可编程上拉电阻，端口电平变化中断。 ADC通道2输入口。 内部高频 RC 振荡频率输出口。

	VREF	P	外部参考电压输入口。
8	PORTA1 PGD AN1 TX	I/O I/O AN O	输入/输出口，带可编程上拉电阻，端口电平变化中断。 编程数据输入/输出口。 ADC通道1输入口。 串口数据发送端口。
9	PORTA0 AN0 PGC RX FLT	I/O I I/O I I	输入/输出口，带可编程上拉电阻，端口电平变化中断。 ADC通道1输入口。 编程数据输入/输出口。 串口数据接收端口。 PWM 外部故障检测端口。
10	VSS	P	电源地。

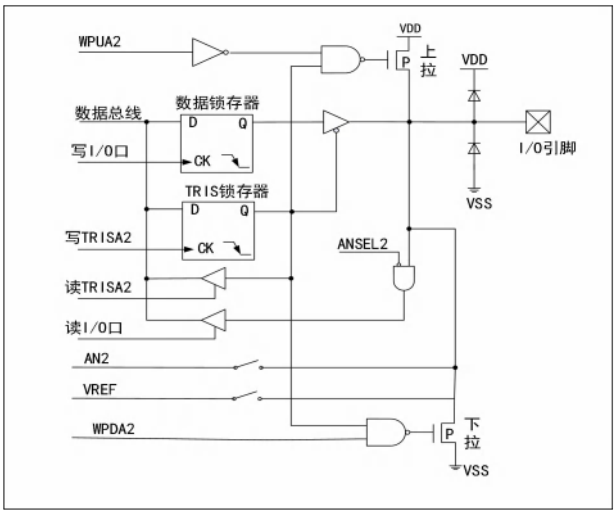
注: I = 输入 O = 输出 I/O = 输入/ 输出 P = 电源 AN = 模拟输入输出

1.5 引脚电路

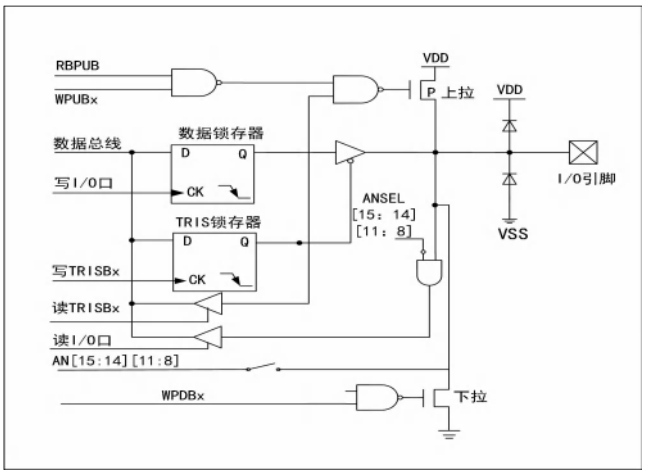
◆ PORTA [1:0]的等效电路



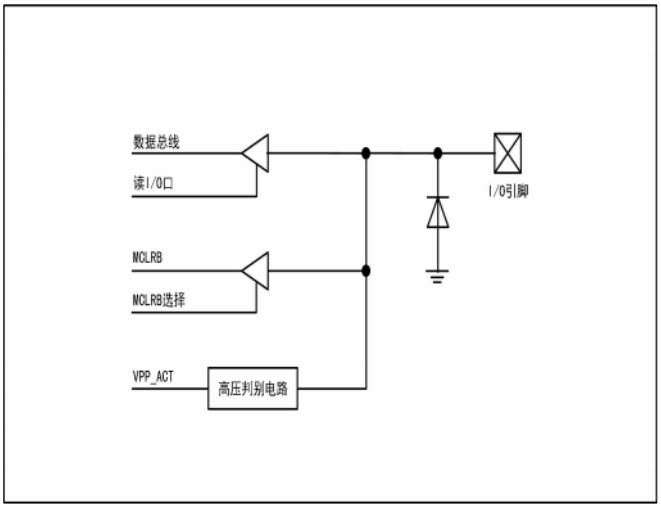
◆ PORTA2的等效电路



◆ PORTB[7:6][3:0]的等效电路



PORTB5 口的等效电路



2 中央处理器（CPU）

SQL5820 CPU内核包括：

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

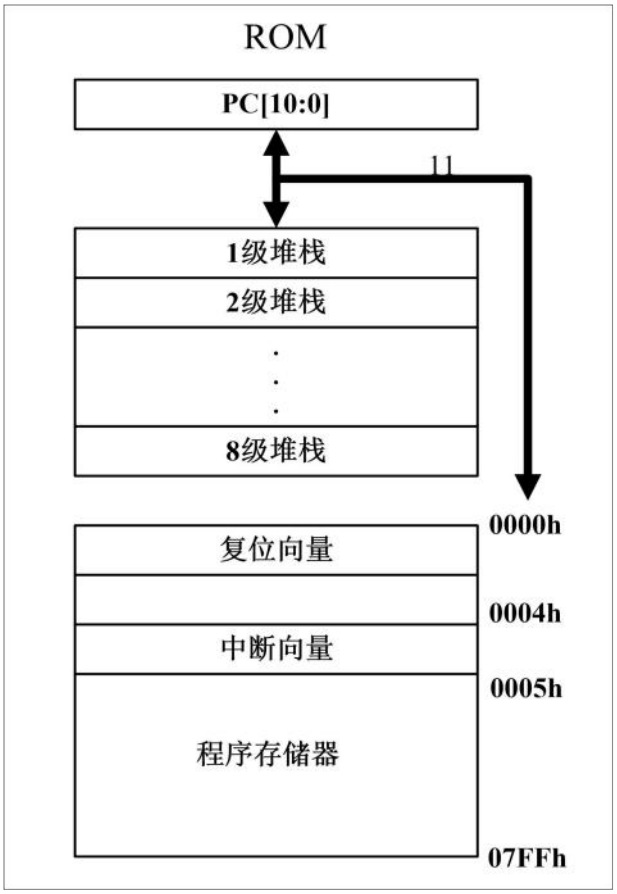
2.1 存储器

2.1.1 程序存储器（ROM）

SQL5820具有2K×16位的程序存储器，图2-1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。

图2-1 程序存储器映射和堆栈



2.1.1.1 复位向量（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:
                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
```

2.1.1.2 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根据程序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**

```
                ORG         0000H
                GOTO        MAIN
                ORG         0004H
                GOTO        INT_SERVICE
```

```
MAIN:
                …
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
                …
                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
```

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. 保存系统寄存器时使用到的 RAM 建议定义在所有 BANK 均映射的位置。
4. 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。
5. 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。
6. 需要软件清空对应的中断标志。
7. 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

2.1.1.3 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 3 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**

```
                ...
                MOVLW     HIGH  TAB1      ;获得数据表地址高8位（内部宏指令）
                MOVWF     PCLATH          ;表地址高位赋给PCLATH
                MOVF      TABBUF,W        ;获得表数据偏移量，调用前赋值。
                CALL      TAB1            ;调用数据表
                ...
                ORG       100H
TAB1：
                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. 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。
3. 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
        MOVLW       HIGH    TAB2    ;获得跳转表地址高位（内部宏指令）
        MOVWF       PCLATH
        MOVF        TABBUF,W
TAB2:   ADDWF       PCL,F
        GOTO        LABLE0_TAB2     ;TABBUF =0,跳转 LABLE0_TAB2
        GOTO        LABLE1_TAB2     ;以下类推
        GOTO        LABLE2_TAB2
        GOTO        LABLE3_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 PMDATLH:PMDATL 寄存器。

➢ **例：查找ROM 地址为“TABLE”的值**

```
        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 _DATA_L   ;TABLE _DATA_L= TABLE地址数据低字节
        MOVF    PMDATH, W
        MOVWF   TABLE _DATA_H   ;TABLE _DATA_H= TABLE地址数据高字节
        ...
        ...
TABLE   DW      1234H           ;定义数据表（16 位）数据。
        DW      F178H
        DW      2123H
```

PMCON

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）

PMDATL

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

PMDATH

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

Bit [15:0] **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

Bit [15:0] **PMAx**[15:0]：ROM存储器地址

2.2 芯片配置选择表

2.2.1 芯片配置字

SQL5820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。

复位时间选择	18ms
	4.5ms
施密特选择	非施密特
	施密特

时钟模式	4T
	2T
启动时钟选择	高频时钟启动
	低频时钟启动
ADC 使能位	使能 ADC
	禁止 ADC
低频系统时钟选择	内部低频 RC
	低频晶体振荡器 LOSCI/LOSCO 为低频晶体振荡器输入/输出口。
	系统仅工作于高频时钟（TMR0 振荡器选择 WDT 振荡器）
	系统仅工作于高频时钟（TMR0 振荡器选择外部 32K）
内部 RC 选择	内部 RC 振荡器频率选择 32MHz
	内部 RC 振荡器频率选择 16MHz
	内部 RC 振荡器频率选择 8MHz
	内部 RC 振荡器频率选择 4MHz
	内部 RC 振荡器频率选择 2MHz
	内部 RC 振荡器频率选择 1MHz
	内部 RC 振荡器频率选择 500KHz
	内部 RC 振荡器频率选择 62.5KHz
加密功能使能位	不加密
	加密
外部复位使能位	使能外部复位
	屏蔽外部复位做输入
WDT 使能位	使能 WDT
	禁止 WDT
高频系统时钟选择	高频晶体振荡器
	内部 RC
复位点选择位	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
	BOR1.5V 当系统电压低于 1.5V 时，系统复位
	BOR3.6V 当系统电压低于 3.6V 时，系统复位
	BOR2.7V 当系统电压低于 2.7V 时，系统复位
	BOR2.0V 当系统电压低于 2.0V 时，系统复位
	BOR2.2V 当系统电压低于 2.2V 时，系统复位
	BOR1.8V 当系统电压低于 1.8V 时，系统复位

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

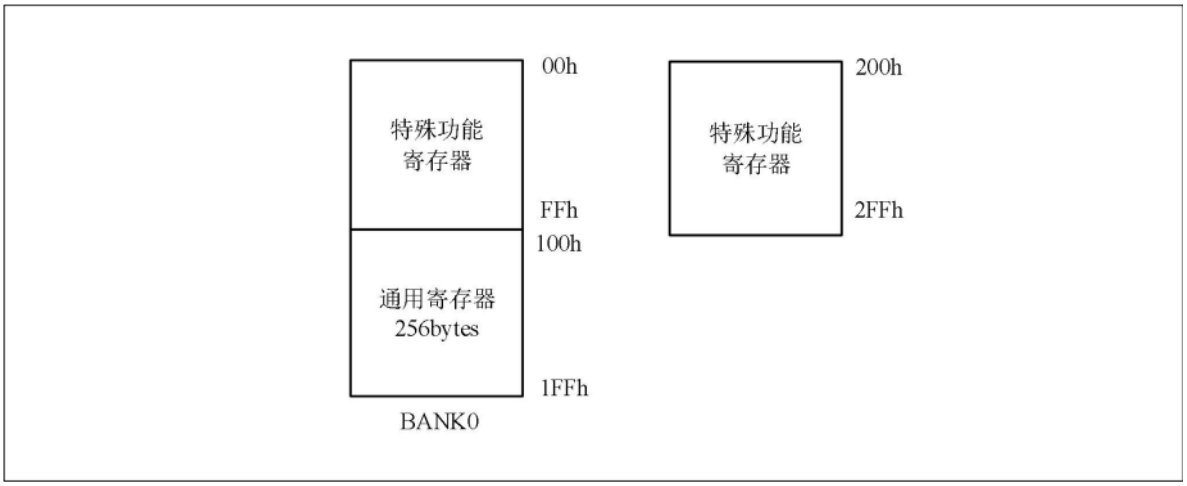
2.3 数据寄存器（RAM）

SQL5820共有256个通用寄存器（GPR）和98个特殊功能寄存器（SFR），分在2个存储区Bank0~Bank1，每个存储区的低256个地址单元保留为特殊功能寄存器，RP0是存储区的选择位。

CORE Register 00-09h&200-209h

00h&200h	INDF0	间接寻址 0 寄存器（不是实际存在的物理寄存器）							
01h&201h	INDF1	间接寻址 1 寄存器（不是实际存在的物理寄存器）							
02h&202h	PCL	程序计数器（PC）低字节							
03h&203h	STATUS			RP0	TO	PD	Z	DC	C
04h&204h	FSR0L	间接寻址 0 地址低位指针							
05h&205h	FSR0H	间接寻址 0 地址高位指针							
06h&206h	FSR1L	间接寻址 1 地址低位指针							
07h&207h	FSR1H	间接寻址 1 地址高位指针							
08h&208h	PCLATH						程序计数器高 3 位		
09h&209h	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF

数据存储器映射



特殊功能寄存器列表如下图：

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	-	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111
011h	TRISB	TRISB7	TRISB6	TRISB5	-	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111
01Ch	PORTA	PORTA7	PORTA6	-	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	0000 0000
01Dh	PORTB	PORTB7	PORTB6	PORTB5	-	PORTB3	PORTB2	PORTB1	PORTB0	0000 0000
028h	WPUA	WPUA7	WPUA6	-	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	1111 1111
029h	WPUB	WPUB7	WPUB6	WPUB5	-	WPUB3	WPUB2	WPUB1	WPUB0	1111 1111
034h	WPDA	WPDA7	WPDA6	-	WPDA4	WPDA3	WPDA2	WPDA1	WPDA0	1111 1111
035h	WPDB	WPDB7	WPDB6	WPDB5	-	WPDB3	WPDB2	WPDB1	WPDB0	1111 1111
040h	IOCA	IOCA7	IOCA6	-	IOCA4	IOCA3	IOCA2	IOCA1	IOCA0	0000 0000
041h	IOCB	IOCB7	IOCB6	IOCB5	-	IOCB3	IOCB2	IOCB1	IOCB0	0000 0000

04Ch	PORCTR	-	-	-	ONA3	CCPCT	PWMCT	UAPCT1	UAPCT0	---- 0000
04Dh	DRENAL	DRENA7L	DRENA6L	-	DRENA4L	DRENA3L	DRENA2L	DRENA1L	DRENA0L	1111 1111
04Eh	DRENBL	DRENB7L	DRENB6L	DRENB5L	-	DRENB3L	DRENB2L	DRENB1L	DRENB0L	1111 1111
054h	PIR1	-	ADIF	-	-	-	CCP1IF	T2IF	T1IF	-0-- -000
055h	PIR2	-	-	PWM0IF	-	RXIF	TXIF	-	CCP2IF	--0- 00-0
056h	PIR3	-	-	-	-	-	-	-	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	PWMPR1	PR1EN	0000 0000
070h	PIE1	-	ADIE	-	-	-	CCP1IE	T2IE	T1IE	0000 0000
071h	PIE2	-	-	PWM0IE	-	-	UARTIE	-	CCP2IE	--00 -000
072h	PIE3	-	-	-	-	-	-	-	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
08Dh	ANSELH	ANSEL15	ANSEL14	-	-	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	-	-	CHS3	CHS2	CHS1	CHS0	ADON	ADEN	00000000
095h	ADCON1	ADFM	ADCS2	ADCS1	ADCS0	VHS2	VHS0	VHS0	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

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位初值
BANK1										
23Ah	BRT									0000 0000
23Bh	AUXR	-	UARTEN	UARTM0	BRTR	BRTX12	S1BRS	SMOD	SMOD0	0000 0000
23Ch	SCON	SM0/FE	SM1	SM2	REN	TR8	RB8	RXWK	-	0000 0000
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
257h	PWM0DTL	DTL0.7	DTL0.6	DTL0.5	DTL0.4	DTL0.3	DTL0.2	DTL0.1	DTL0.0	0000 0000
258h	PWM0DTH					DTH0.3	DTH0.2	DTH0.1	DTH0.0	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			-	-	-	-	FLT0M1	FLT0M0	--00 0000
260h	PWMM				PWM0M				RELOAD0	0000 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

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

2.3.1 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CLRWDT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CLRWDT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

2.3.3 PC 寄存器

程序计数器（PC）为11位宽，低字节来自可读写的PCL寄存器，高字节（PC[10:8]）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	-	-	-	-	-	PCH10	PCH9	PCH8
R/W	-	-	-	-	-	R/W	R/W	R/W
POR的值	-	-	-	-	-	0	0	0

2.4 寻址模式

SQL5820 共有三种寻址方式：立即寻址、直接寻址和间接寻址模式

2.4.1 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
```

2.4.2 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
```

2.4.3 间接寻址

由指针 FSR 指向的寄存器参与运算的寻址方式。INDF 寄存器不是物理寄存器，对 INDF 寄存器操作可以实现间接寻址

- 例：利用间接寻址对 0X100~0X1FF 通用数据存储器进行清零

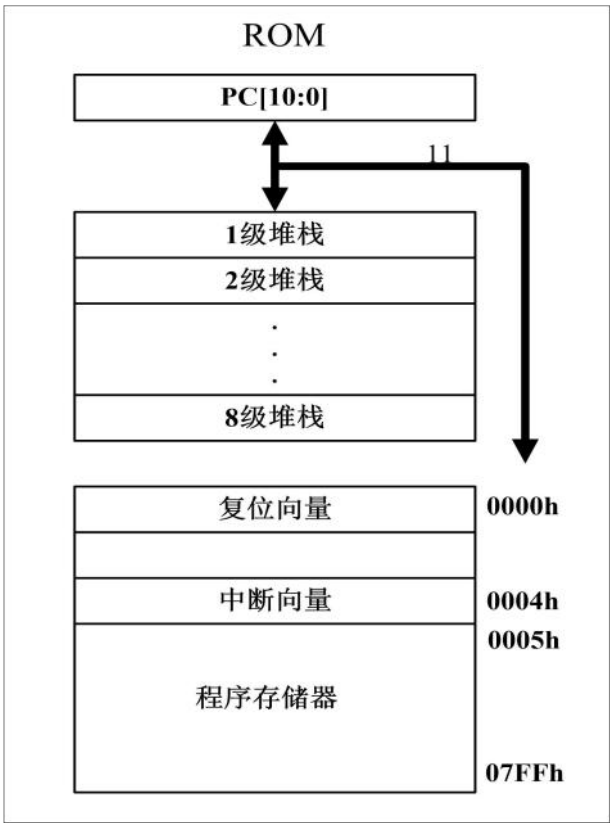
```
            MOVLW   00h       ;清零 0X100~0X1FF
            MOVWF   FSR0L
            MOVLW   0x01
            MOVWF   FSR0H     ; FSR 指向 100h 地址
NEXTBYTE:   CLRF    INDF0     ;对 FSR 指向的数据存储器清零
            INCF    FSR0L,F   ;FSR + 1,指向下一个地址
```

```
                MOVLW     00h          ;注意这里的边界值为欲操作 RAM 最大地址 + 1
                XORWF     FSR0L,W      ;利用间接寻址，注意意外指向特殊寄存器的情况
                BTFSS     STATUS,Z
                GOTO      NEXTBYTE     ;FSR 的值小于 1FFh,循环清零下一个地址

CONTINUE:       …                      ;完成清零操作
```

2.5 堆栈

SQL5820 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。

3 复位

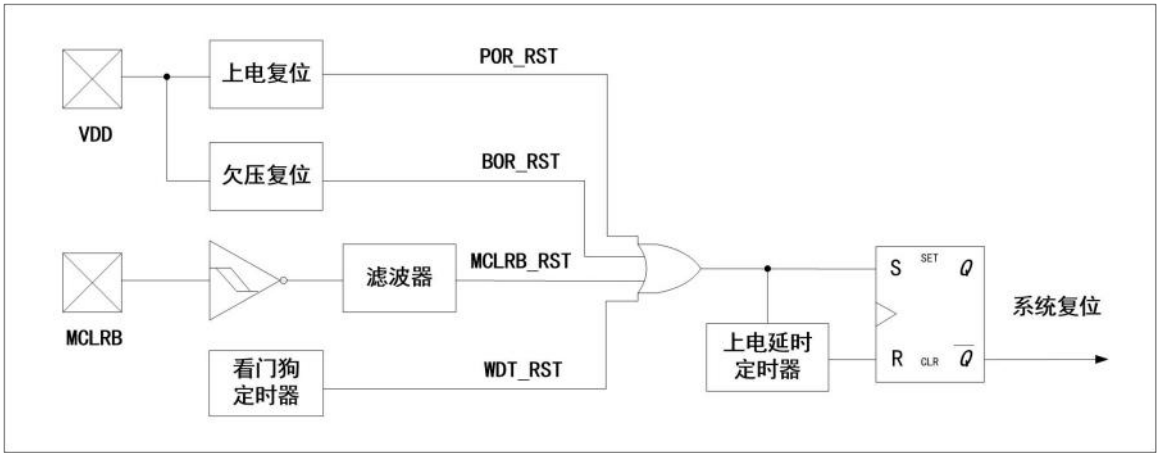
SQL5820 共有四种复位方式：

- 上电复位（POR）
- 外部复位（MCLRB 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。

上电延时定时器在复位时提供一个18 ms（典型值）的固定延时和一个振荡器起振稳定的时间延时。

图3-1 复位电路示意图



特殊功能寄存器复位状态

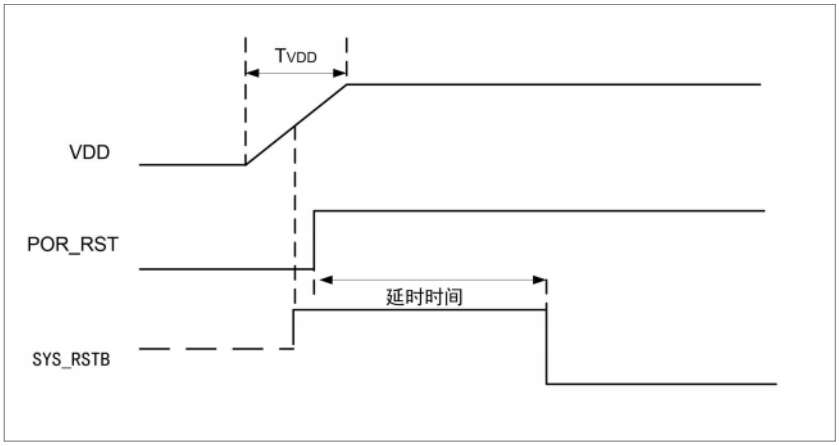
复位方式	STATUS寄存器	PCON寄存器
上电复位	0001 1xxx	00-1 qq00
正常工作模式下的外部复位	0001 1xxx	00-1 qq0u
休眠模式下的外部复位	0001 0uuu	00-1 qquu
欠压复位	0001 0uuu	00-1 qqu0
看门狗定时器复位	0000 1uuu	00-1 qquu

注： u＝ 不变， x＝ 未知， -＝ 未使用，q＝ 取值视条件而定

3.1 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 震荡器的起振时间最短，晶体震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。

图3-2 上电复位示意图



3.2 外部复位

当外部复位端口 MCLRB 输入一个持续时间超过 **TMCLRB** 的低电平时，产生外部复位。MCLRB 选择配置字为 1，MCLRB 口为外部复位输入口。

图3-3 外部复位示意图

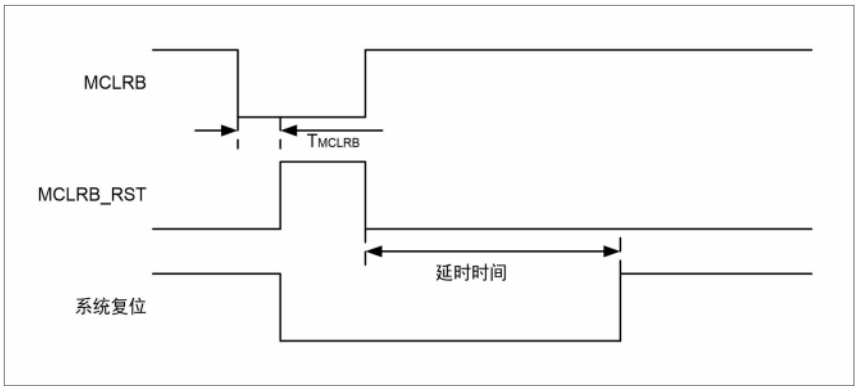
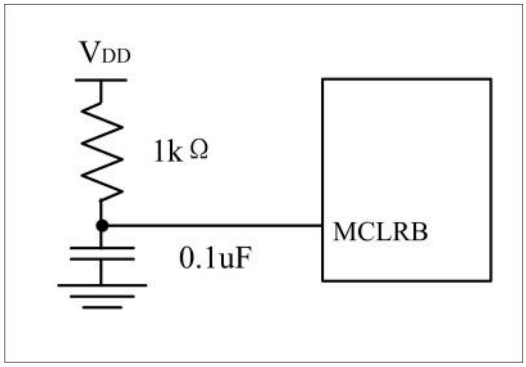


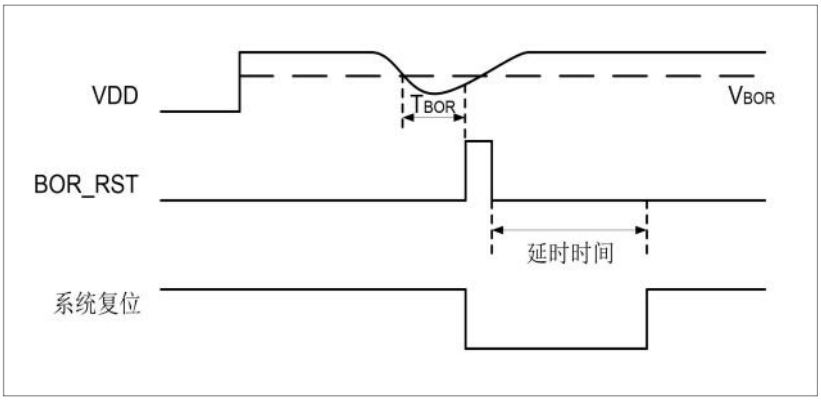
图3-4 建议外部复位电路



3.3 欠压复位

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

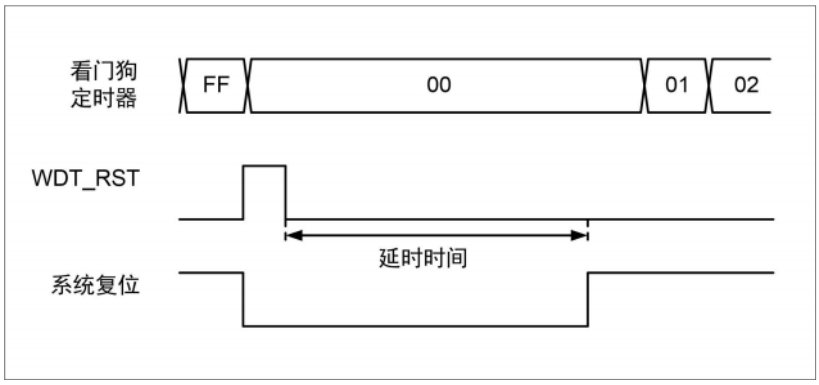
图3-5 欠压复位示意图



3.4 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。

图3-6 看门狗复位示意图



3.5 PCON 寄存器

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

注： — = 未实现，q = 取值视条件而定

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）

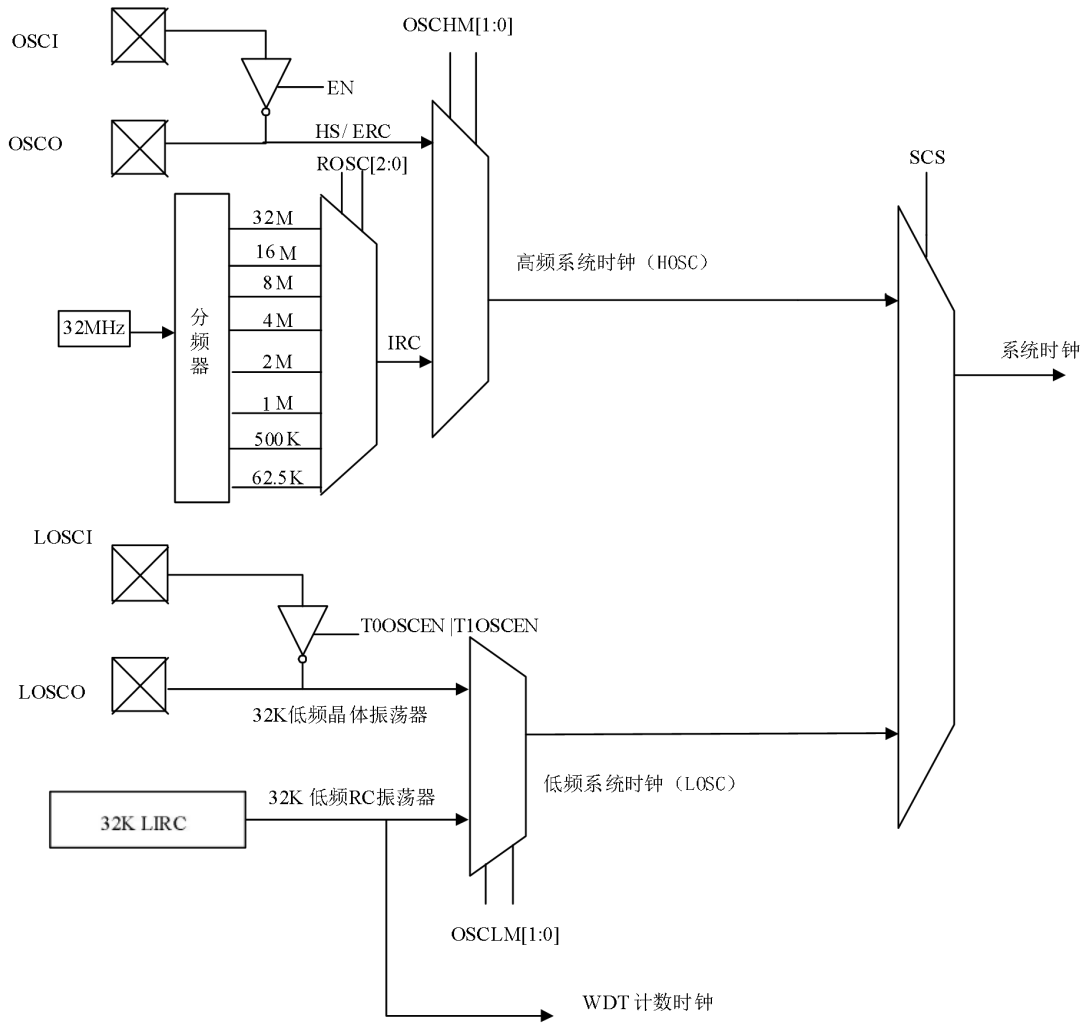
4 系统时钟

4.1 概述

SQL5820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：Fcpu = Fsys / N，N = 2或4，时钟模式选择决定N的值。
- 低频模式：Fcpu = Fsys / N，N = 2或4，时钟模式选择决定N的值。

4.2 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

4.3 系统高频时钟

系统高频时钟有三种选择，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 作为输入/输出口
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
10	保留
11	LP 模式

4.3.1 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32M/16M/8M/4M/2M/1M /500K/62.5K 八种选择。

高频内部RC振荡器频率选择配置字

ROSC[2:0]	说明
111	内部RC振荡器频率选择32MHz
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
000	内部RC振荡器频率选择62.5KHz

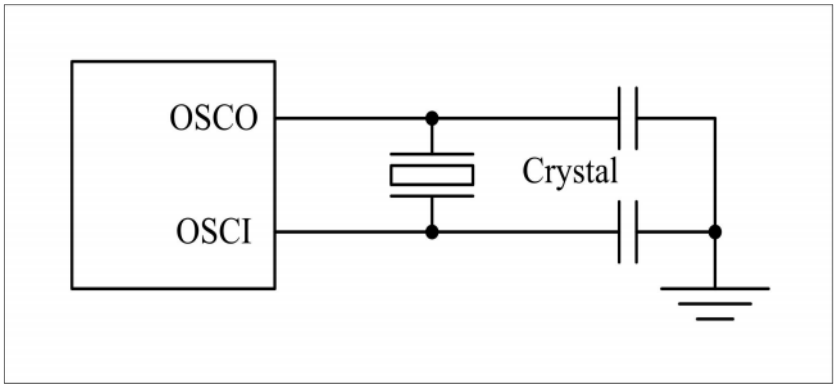
4.3.2 外部高频时钟

外部高频时钟共两种模式，由配置字 OSCHM 控制具体模式的选择

- 高频晶体振荡器： 最高 20MHz
- 外部时钟输入

4.3.2.1 高频晶体振荡器

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

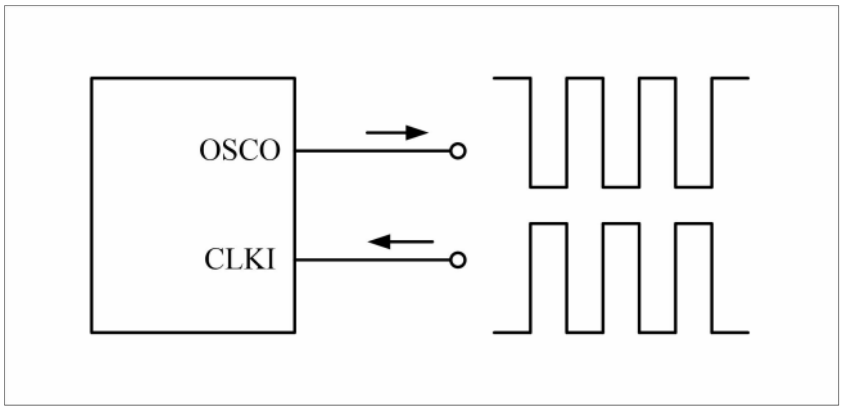


注：
OSCI 和 OSCO 引脚与振荡器和起振电容之间距离越近越好。

4.3.2.2 外部时钟源

单片机可选择外部时钟信号作为系统时钟，由配置字OSCHM控制，从OSCI 脚送入。

如果配置字OSCHM选择为高频晶体振荡器，外部时钟由CLKI引脚输入，OSCO引脚为CLKI的反向输出。



注：
外部振荡电路中的 GND 必须尽可能的接近单片机的 VSS 端口。

4.4 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： 32K（5V 典型值）

低频系统时钟选择配置字

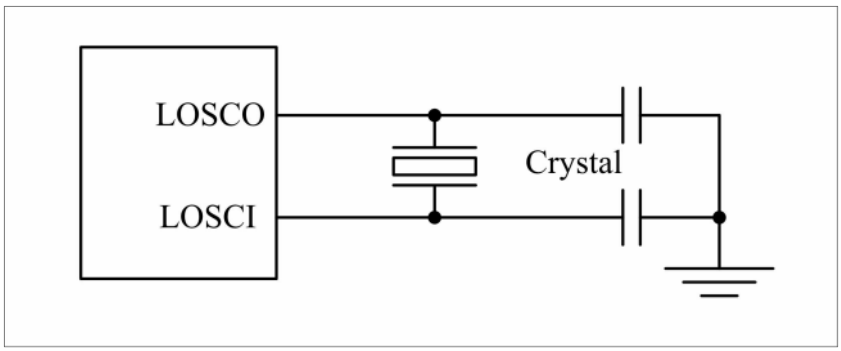
OSCLM[1:0]	说明
00	低频 RC 振荡器，32KHz，LOSCI/LOSCO 作为输入/输出口
01	低频晶体振荡器，32.768KHz，LOSCI/LOSCO 作为低频晶体振荡器输入/输出口
10	系统仅工作于高频时钟（TMR0 振荡器选择 WDT 振荡器）

11	系统仅工作于高频时钟（TMR0 振荡器选择外部 32K）

4.4.1 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。

4.4.2 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）。

注：

低频时钟也用作看门狗定时器的时钟。

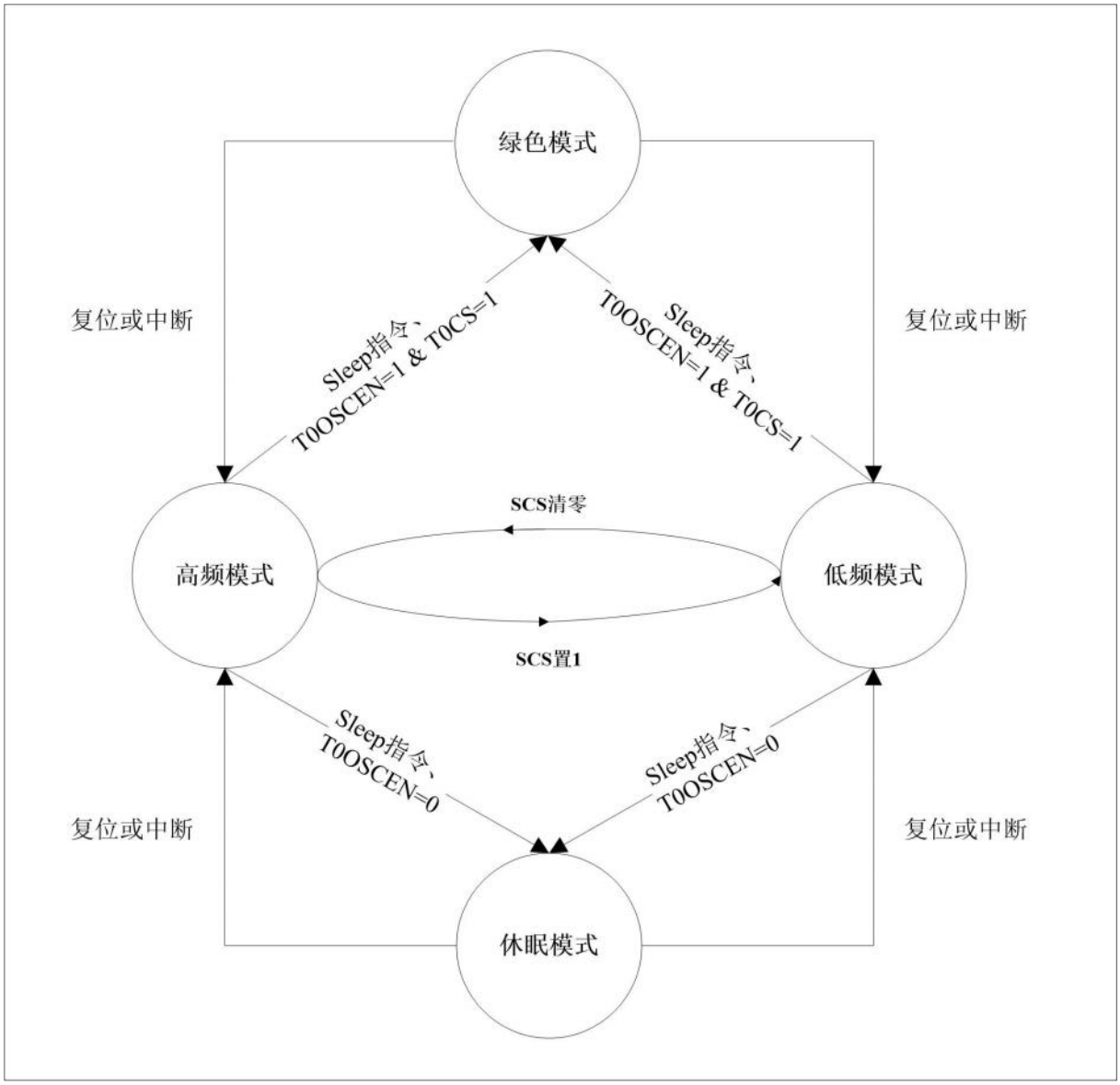
5 系统工作模式

SQL5820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5-1 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

5.1 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
```

5.2 高低频模式切换

图5-2 高低频切换时序图一

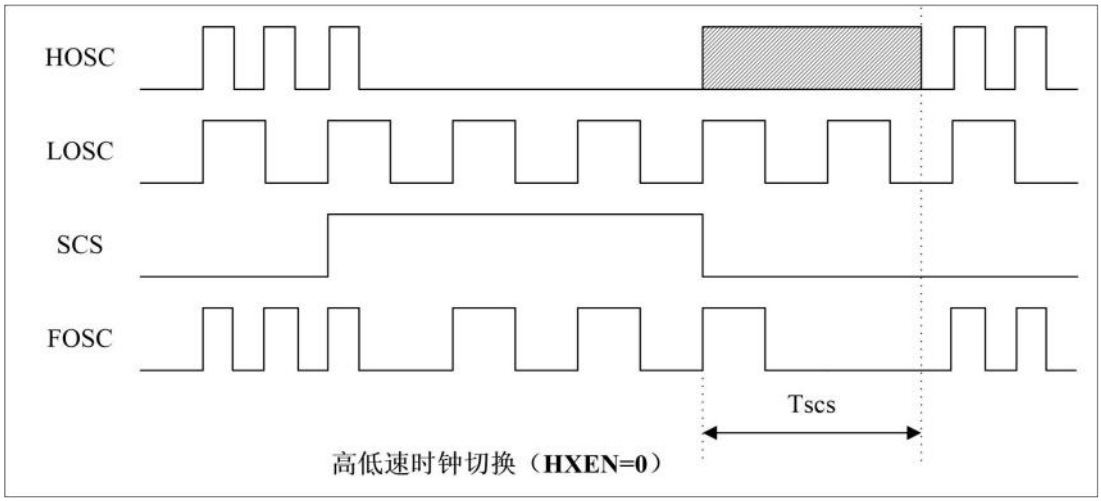
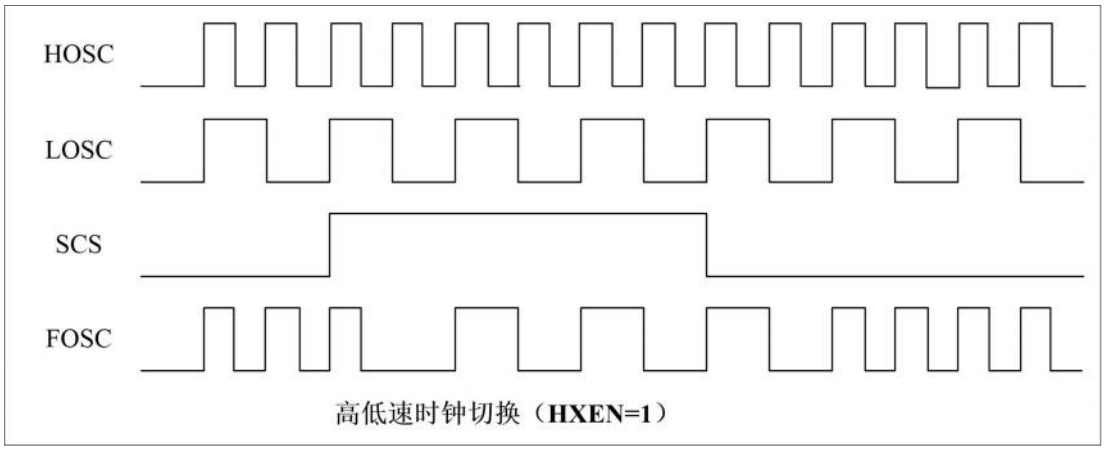


图5-3 高低频切换时序图二



时钟切换时间（Tscs）计算：

Tscs = 高频振荡器起振时间 + 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

5.3 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，等待的这一段时间称为唤醒时间。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：

唤醒时间 = 起振时间 + 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。

5.4 OSCCON 寄存器

07Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OSCCON	T0OSCEN	-	-	-	-	-	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 [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

6 中断源

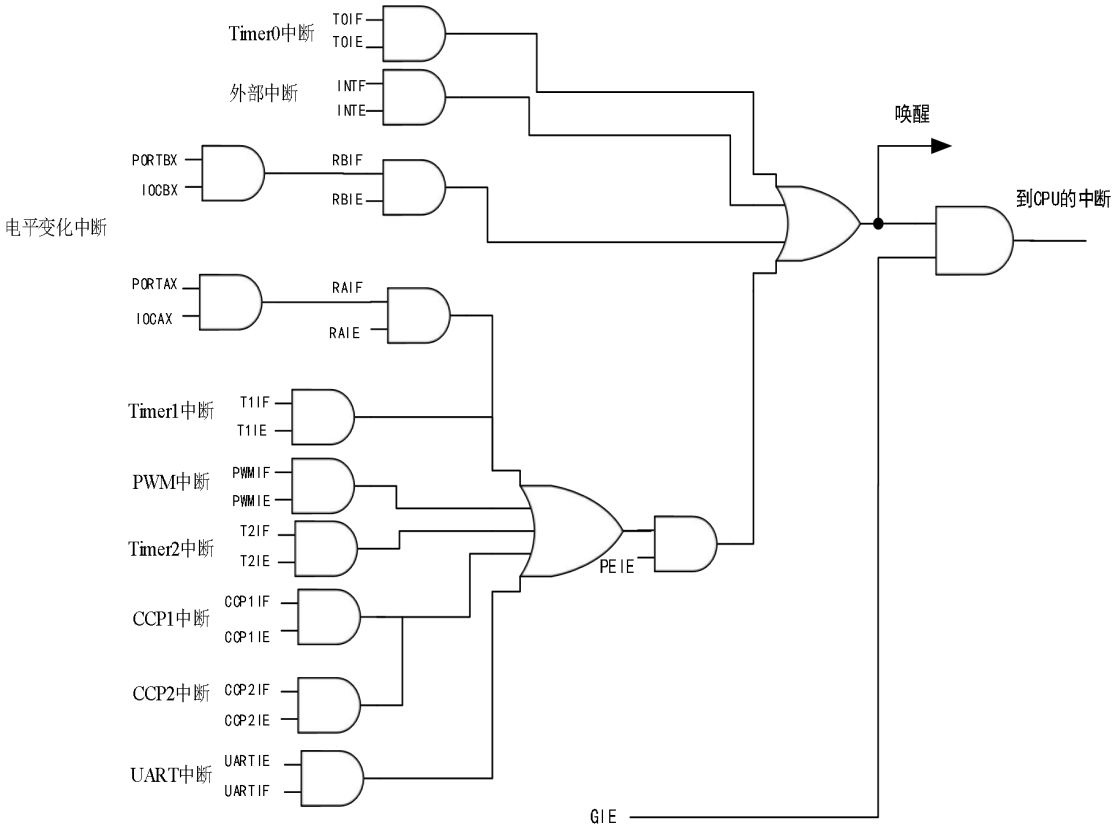
SQL5820中断源:

- 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。

图6-1 中断示意图



6.1 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

6.2 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	-	-	-	-	-	-	-	RAIE
R/W	-	-	-	-	-	-	-	R/W
POR的值	-	-	-	-	-	-	-	0

Bit [0] **RAIE**：PORTA电平变化中断使能位

1 = 使能PORTA电平变化中断

0 = 禁止PORTA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

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] **T1IF**：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

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

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

056h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR3	-	-	-	-	-	-	-	RAIF
R/W	-	-	-	-	-	-	-	R/W
POR的值	-	-	-	-	-	-	-	0

Bit [0] **RAIF**：PORTx电平变化中断标志位

1 = PORTA[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 = PORTA[7:0]电平状态没有变化

6.3 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。

6.4 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：

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
```

6.5 Timer0 定时器中断

T0 溢出时，无论 T0IE 处于何种状态，T0IF 都会置“1”。若 T0IE 和 T0IF 都置“1”，且 GIE 使能，系统就会响应 TIMER0 的中断;若 T0IE = 0，则无论 T0IF 是否置“1”，系统都不会响应 TIMER0 中断。

6.6 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T0IF。
        GOTO    EXIT_INT        ;T0IF = 0，退出中断。
        BCF     INTCON,INTF     ;T0IF 清零。
        …                       ;INT0 中断服务程序。
        …
EXIT_INT:
        …                       ;恢复STATUS、W和PCLATH。
        RETFIE                  ;退出中断。
```

6.7 PORT 电平变化中断

PORTx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PORTx端口设为输入，并将寄存器IOCx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	IOCB5	-	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

Bit [7:0] **IOCB**[7:0]：PORTBx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	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCA	IOCA7	IOCA6	-	IOCA4	IOCA3	IOCA2	IOCA1	IOCA0
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

Bit [7:0] **IOCA**[7:0]：PORTA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。

➢ 例：PORTB1 电平变化中断请求设置。

```
        BCF     STATUS,RP0      ;BANK0
        MOVLW   0X02
        IORWF   TRISB,F         ;PORTB1 端口为输入。
        MOVLW   0X02
        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：PORTB 中断。

```
        ORG     0004H
        GOTO    INT_SERVICE
INT_SERVICE:
        …                       ;保存STATUS、W 和PCLATH。
        BCF     STATUS,RP0      ;BANK0
        BTFSS   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
        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。
        SLEEP
        BCF     INTCON,RBIE
        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。

6.8 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。
```

6.9 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

```
        MOVLW   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
        MOVLW   nnH
        MOVWF   T1H
```

```
                MOVLW     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。
```

6.10 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。

6.11 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。

6.12 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。

6.13 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。

6.14 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
RAIF	PORTA电平变化
T2IF	T2的值和PR2相同
RXIF/TXIF	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:
                …                               ;保存STATUS、W和PCLATH。
INT0CHK:                                        ;检查是否有INT0 中断请求。
                BCF     STATUS,RP0              ;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:                                       ;检查是否有T0 中断请求。
                BTFSS   INTCON,T0IE             ;检查是否使能T0 中断。
                GOTO    INTT2CHK                ;跳到下一个中断。
                BTFSC   INTCON,T0IF             ;检查是否有T0 中断请求。
                GOTO    INTT0                   ;进入T0 中断。
INTT2CHK:                                       ;检查是否有T2 中断请求。
                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
        ````````                ;T2中断处理程序
        GOTO   INT_EXIT
INT_EXIT:
        …   ;恢复STATUS、W和PCLATH。
        RETFIE   ;退出中断。
```
```

7 I/O口

SQL5820共有两组双向端口：

- PORTA口
- PORTB口

7.1 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	-	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
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

011h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISB	TRISB7	TRISB6	TRISB5	-	TRISB3	TRISB2	TRISB1	TRISB0
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

注：PORTB5可配置为开漏输出

TRISx **[7:0]**：PORTx[7:0]的输入输出控制位

1 =输入状态

0 =输出状态

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	-	-	ANSEL11	ANSEL10	ANSEL9	ANSEL8
R/W	R/W	R/W	-	-	R/W	R/W	R/W	R/W
POR的值	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’ 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。

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

ANSEL[11:8]对应 AN11~AN8（PB3~PB0），ANSEL[15:14]对应 AN15、AN14（PB7、PB6）

7.2 I/O 口上拉控制寄存器

028~029h	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

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	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 [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)

7.3 I/O 口下拉控制寄存器

034h~035h	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

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。

7.4 I/O 驱动控制寄存器

04Dh~04Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DRENxL	DRENN7L	DRENN6L	DRENN5L	DRENN4L	DRENN3L	DRENN2L	DRENN1L	DRENN0L
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~0A1h	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	0	0	0	0	0	0	0	0

x=A/B

DRENxL [7:0]：DRENxL[7:0]的驱动控制位

DRENxH=0时：

0 = 源电流/灌电流（Level0）（IOH=VDD-0.6V 30mA@ 5V）（IOL=VSS+0.6V 50mA@ 5V）

1 = 源电流/灌电流（Level1）（IOH=VDD-0.6V 8mA@ 5V）（IOL=VSS+0.6V 10mA@ 5V）

DRENxH=1时：

1 = 源电流/灌电流（Level2）（IOH=VDD-0.6V 20mA@ 5V） (IOL=VSS+0.6V 25mA@ 5V）

0 = 源电流/灌电流（Level3）（IOH=VDD-0.6V 3mA@ 5V） （IOL=VSS+0.6V 5mA@ 5V）

7.5 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	-	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0
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

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

7.6 I/O 口管脚配置寄存器

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

Bit[3]: **CCPCT** CCP 管脚配置位

0 = CCP1/CCP2 管脚配置在 PORTB2/PORTB3（默认）

此芯片只能配置为 0

Bit[2]: **PWMCT** PWM 管脚配置位

0 = PWM0/PWM01 管脚配置在 PORTB7/PORTB6（默认）

此芯片只能配置为 0

Bit[1:0] **UAPCT[1:0]** UART 管脚配置位

10 = RX 配置在 PORTB2，TX 配置在 PORTB3

11 = RX 配置在 PORTA0，TX 配置在 PORTA1

此芯片默认状态下为 00.必须配置为 10 或者 11

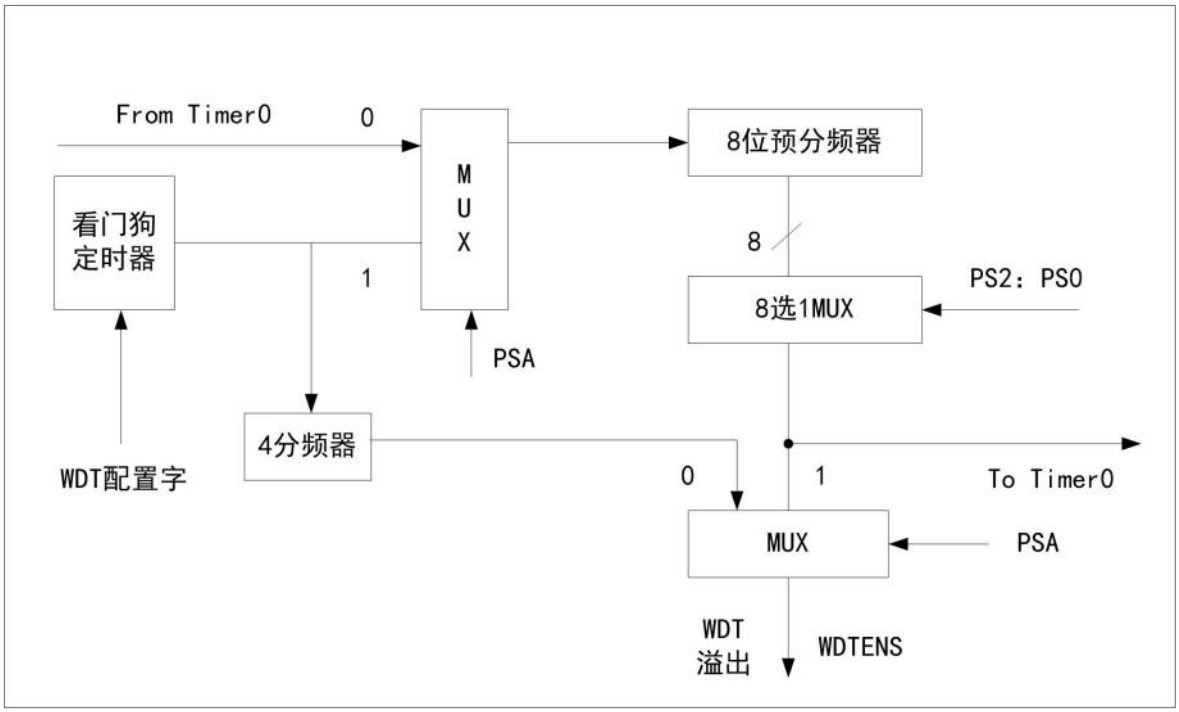
注：管脚复用功能优先级： UART > CCP > PWM > IO。

8 定时器/计数器

8.1 看门狗定时器

SQL5820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Timer0定时器/计数器。

图8-1 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

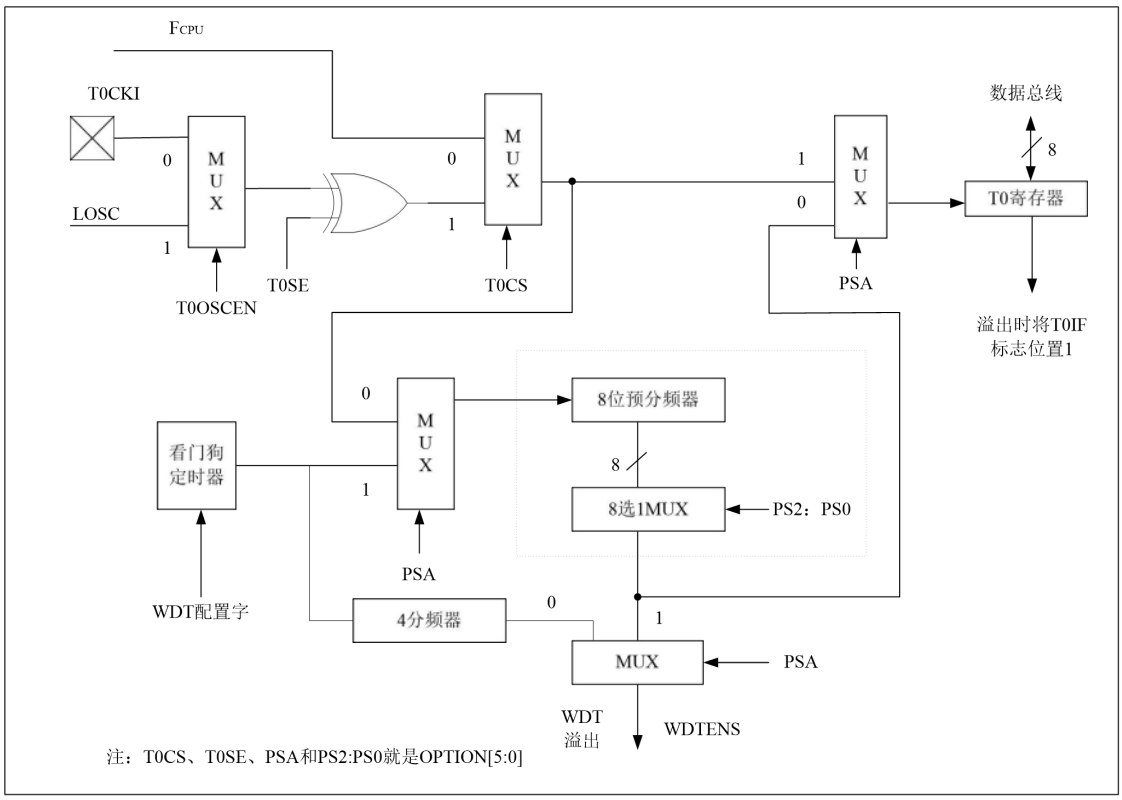
```
        BCF     STATUS,RP0      ;BANK0
        BSF     PCON,WDTENS     ;使能看门狗
        CLRWDT                  ;看门狗定时器清零
        …
```

8.2 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 8-2 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	T0OSCEN	-	-	-	-	-	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	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

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

T0CS	T0OSCEN	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

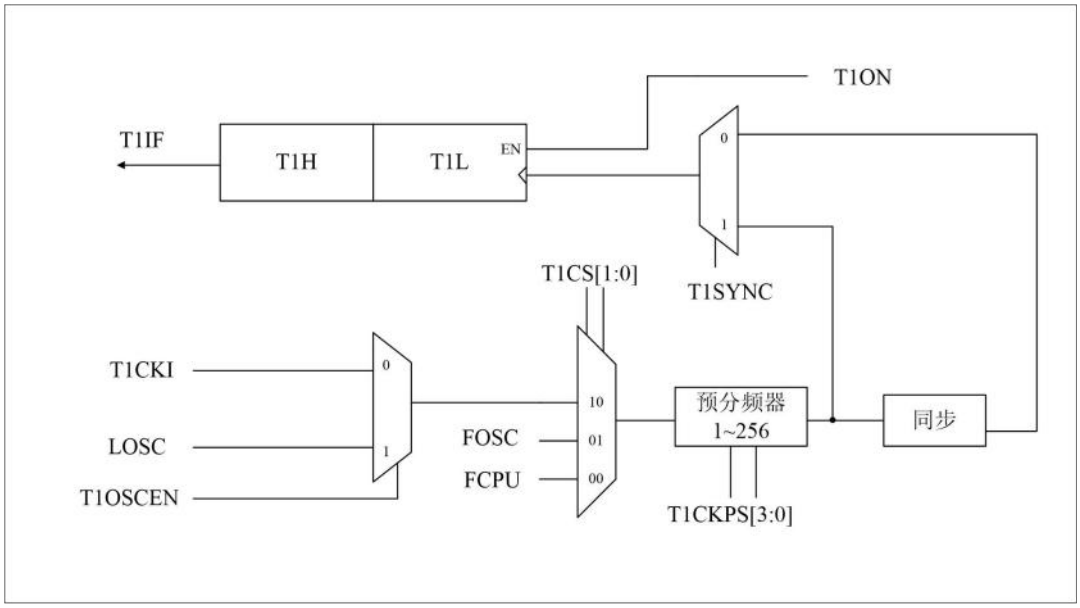
T0INT_SERVICE:
        …                           ;保存STATUS、W 和PCLATH。
        BCF     STATUS,RP0          ;BANK0
        BTFSS   INTCON,T0IF         ;检测T0IF。
        GOTO    EXIT_INT            ;T0IF = 0，退出中断。
        BCF     INTCON,T0IF         ;T0IF 清零。
        …                           ;TIMER0 中断服务程序。
        …
EXIT_INT:
        …                           ;恢复STATUS、W和PCLATH。
        RETFIE                      ;退出中断。
```

8.3 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

Timer1模块框图



8.3.1 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

注：

若使用定时器计数器功能时，配置为 1:16、1:32 及以上分频时请先将 PR1CON 寄存器的 PR1EN 置 1，PR1L 的寄存器写入 0XFF，再进行 Timer1 输入时钟分频比选择

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。

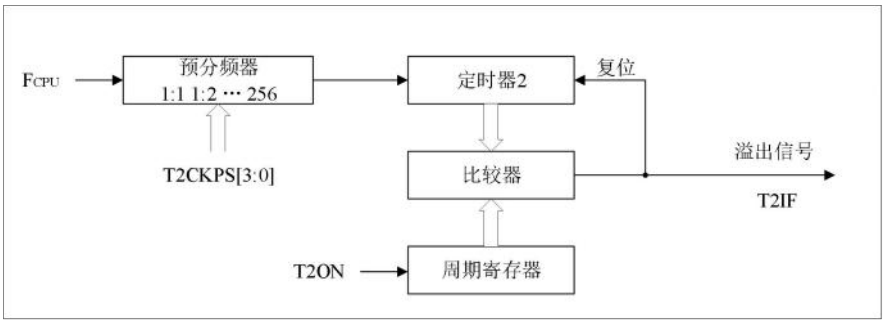
2、Timer1 工作于同步计数器模式和同步定时唤醒模式时，不能唤醒 SLEEP 或绿色模式

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

8.4 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8-4 Timer2模块框图



8.4.1 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

8.4.2 Timer2 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

8.4.3 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 FCPU，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。

Timer2 溢出时间 = （PR2 + 1）* 预分频比/Fcpu。

8.5 CCP 模块

SQL5820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

8.5.1 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。

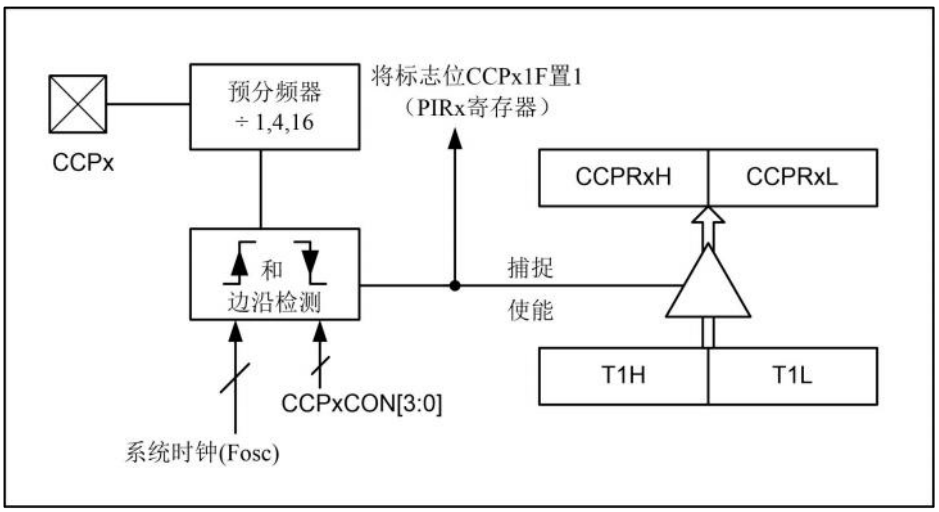
2、 CCPxCON寄存器的CCPxM[3:0]设置的是预分频器，关闭CCP模块或者CCP模块不在捕捉模式，预分频计数器将会被清零。为避免错误中断，可在改变预分频比前通过清零 CCPxCON寄存器来关闭CCP模块。如果需要中途改变预分频器的分频比，建议使用以下程序片段：

```
BCF     STATUS,RP0          ;BANK0
CLRF    CCPxCON             ;关闭CCPx模块
MOVLW   0X05/0X06/0X07      ;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。

图 8-5 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。

8.5.2 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：

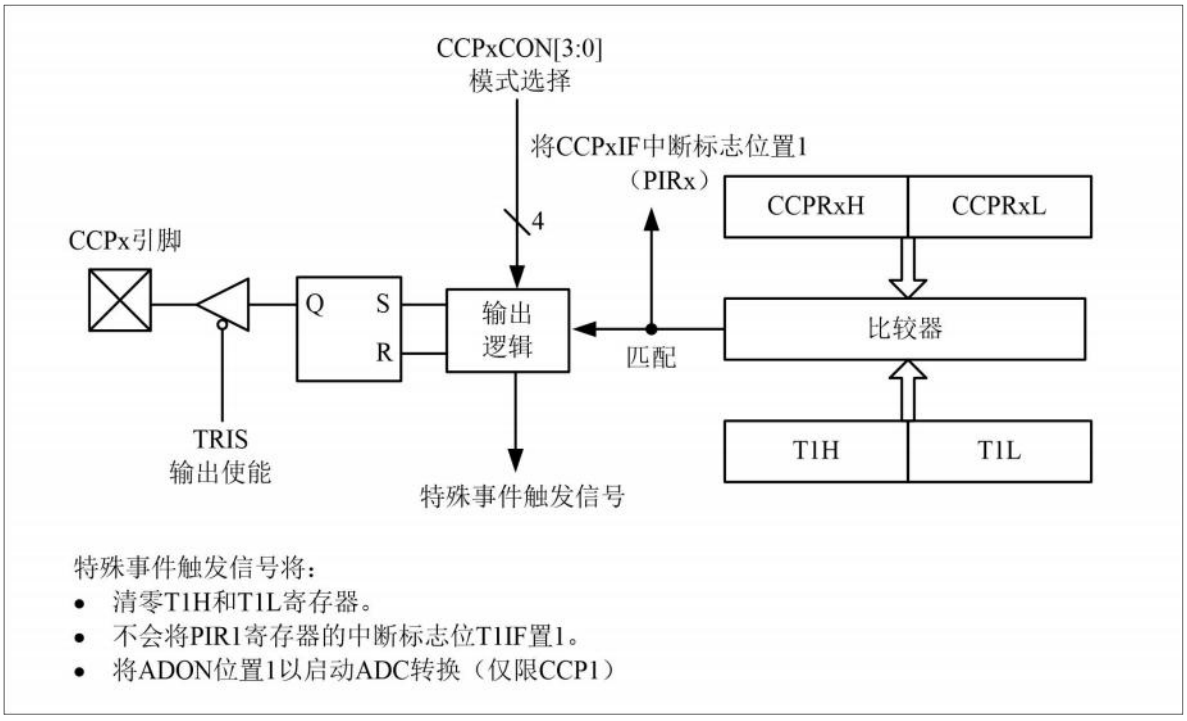
1、 当选择产生特殊事件触发信号时，如果ADC被使能，则启动一次ADC转换(仅限于CCP1)。在此模式下CCPx模块不会对CCPx引脚进行控制。

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8-6 比较模式工作原理图



特殊事件触发信号将：

- 清零T1H和T1L寄存器。
- 不会将PIR1寄存器的中断标志位T1IF置1。
- 将ADON位置1以启动ADC转换（仅限CCP1）

8.5.3 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。

8.5.3.1 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）。

图8-7　PWM模式工作原理图

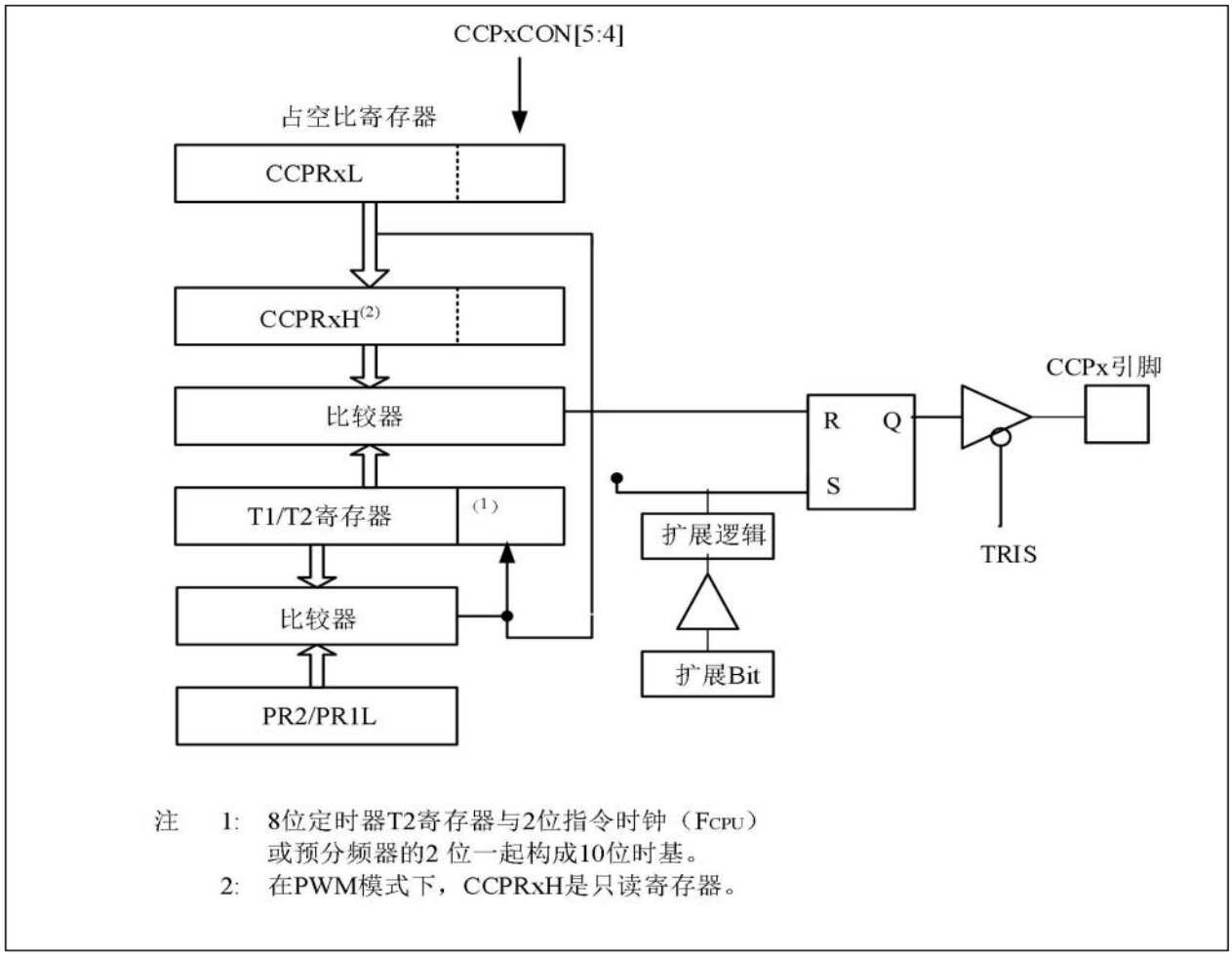
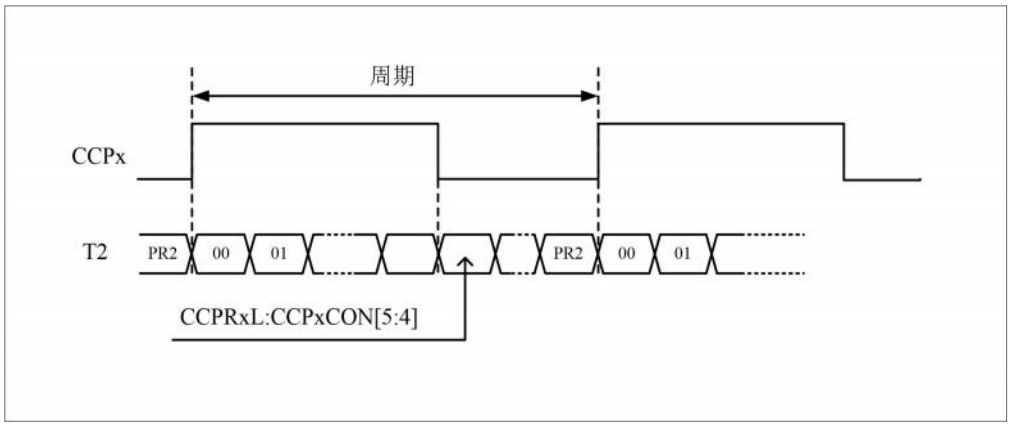


图8-8 PWM波形图



✓ **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：

$$\text{PWM 周期} = [(PR2) + 1] \bullet 4 \bullet n \bullet Tsys$$

（Timer2 预分频值）

注： $Tsys = 1/Fsys$

PWM脉冲宽度：

$$\text{脉冲宽度} = (CCPRxL:CCPxCON[5:4]) \bullet n \cdot Tsys \cdot (\text{Timer2 预分频值})$$

注： $Tsys = 1/Fsys$

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：

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

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：

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

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）。

PWM使用：

1. 设置相关TRISB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位清零，让CCPx引脚输出
7. 如何关闭PWM输出
   - 将TRISB2或TRISB3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

        BCF     STATUS,RP0      ; BANK0
        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     STATUS,RP0
        BSF     PIE1,T2IE
```

```
BTFSS   PIR1,T2IF       ;等待T2溢出
GOTO    $-1
BCF     PIR1,T2IF       ;清除中断标志
BCF     STATUS,RP0      ;BANK0
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	PWM1T1	PWM1T0	PWM2T1	PWM2T0	T1CKPS3	T1CKPS2	PWMPR1	PR1EN
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[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

Bit[1] **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[0] **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

使用注意：

1、 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，T1L必须赋值为0X00，其他使用操作可参考由Timer2提供时钟源的PWM2的使用。

2、 PWM2的时钟源选择T1的Fsys时，配置字OPTION的时钟模式选择4T，不要选择2T。否则会出现占空比异常的情况。

8.5.3.2 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	PWM1T1	PWM1T0	PWM2T1	PWM2T0	T1CKPS3	T1CKPS2	PWMPR1	PR1EN
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[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

Bit[5:4] **PWM2T[1:0]**：PWM2的扩展周期选择位

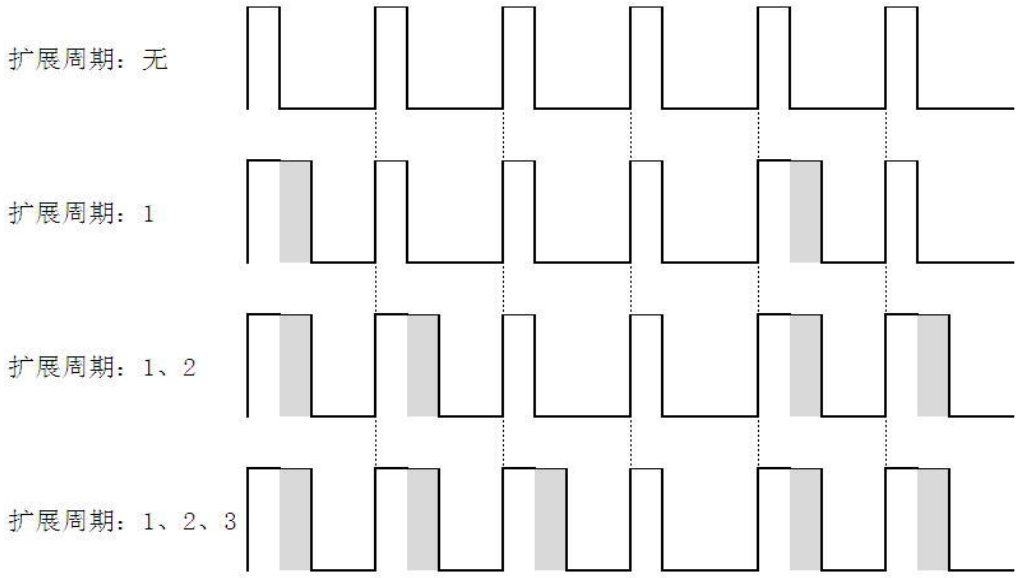
00： 无扩展周期

01： 扩展周期为1

10： 扩展周期为1、2

11： 扩展周期为1、2、3

PWM扩展示意如下：



9 PWM模块

9.1 PWM 特性

- 1 组带死区互补 PWM 或 2 路独立 PWM 输出
- 提供每个 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

SQL5820 集成了一个 12 位 PWM 模块 PWM0，PWM0 有一个计数器，PWM0 的计数器由 PWM0_EN 和 PWM01_EN 来控制，只要使能 PWM0_EN 和 PWM01_EN 中的任何一个，计数器就会启动，计数器的时钟源通过 PWM0C 控制寄存器里的 CK0 来选择。

当需要从芯片管脚输出 PWM 波形时，还需要将端口设置为输出模式。如果不希望从芯片管脚上输出 PWM 波形，可以不用设置端口模式，这时候 PWM 的计数器可以当一个定时器来使用，当计数器溢出时，如果中断允许也会产生 PWM 中断。

如果 EFLT0 置 1，PWM0 输出和其互补输出可由 FLT0 引脚输入信号变化自动关闭。一旦检测到 FLT0 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 FLT0 引脚错误去除后继续 PWM 输出。在 FLT0 输入信号有效期间，FLT0S 位无法清除。只有当 FLT0 输入信号消失后，才能软件清除 FLT0S 状态位，此时 PWM 恢复正常输出。

PWM 模块有相应的控制位及标志位，方便用户定时修改 PWM 模块下一次循环的周期或占空比。

9.2 PWM 输出模式

PWM 模块包含独立的波形发生模块，对应的 PWM 输出为 PWM0/PWM01，通过控制相关寄存器可使 PWM 输出配置成互补输出模式或独立输出模式。

9.2.1 互补输出模式

当 PWM0M 置 0：PWM 将工作在互补输出模式，互补输出模式时，可以控制对应的周期寄存器、占空比寄存器及死区时间寄存器，从而控制互补波形的输出。互补输出时可选择 PWM0&PWM01 输出性， 方便用户各种电平驱动需求

9.2.2 独立输出模式

当 PWM0M 置 1：PWM 将工作在独立输出模式，独立输出模式时，可以控制相关寄存器使能对应 PWM 端口单一输出或同时输出， 同时让 PWM0&PWM01 输出时，其周期相同但占空比可单独设置。此时互补输出模式时占空比寄存器将控制 PWM0 的占空比，死区时间控制寄存器将控制 PWM01 的占空比，独立输出时也可控制 PWMx&PWMx1 输出极性，方便用户各种电平驱动需求。

9.3 PWM 相关寄存器

9.3.1 PWM 使能寄存器

25Eh	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

0 = PWM01输出禁止，用作I/O功能

1 = PWM01 输出允许

Bit [0] **EPWM0**：PWM0输出控制位

0 = PWM0输出禁止，用作I/O功能

1 = PWM0 输出允许

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：

Bit [1:0] **FLT0M[1:0]**：PWM0 口 FLT 故障时后，端口输出状态

00 = PWM0 输出低电平

01 = PWM00 输出低电平，PWM01 输出高电平

10 = PWM00 输出高电平，PWM01 输出低电平

11 = PWM0 输出高电平

9.3.2 PWM0 控制寄存器

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 0

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/128

10 = Fosc/32

01 = Fosc/8

00 = Fosc/1

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，PWM0/1/2和PWM01/11/21输出固定电平：

PWM0S[1:0] =00，PWM0和PWM01均输出固定低电平；

PWM0S[1:0] =01，PWM0输出固定低电平，PWM01输出固定高电平；

PWM0S[1:0] =10，PWM0输出固定高电平，PWM01输出固定低电平；

PWM0S[1:0] =11，PWM0和PWM01均输出固定高电平；

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。

9.3.3 PWM 模式寄存器 PWMM

260h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMM	-	-	-	PWM0M	-	-	-	RELOAD0
R/W	-	-	-	R/W	-	-	-	R/W
POR的值	-	-	-	0	-	-	-	0

Bit [4] **PWM0M**：PWM0工作模式选择位

1 = PWM1&PWM11工作于独立输出模式

0 = PWM0&PWM01 工作于互补输出模式

Bit [0] **RELOAD0**：PWM0自动重载使能位

1 =使能自动重载

0 = 禁止自动重载

注：默认值为 1，默认状态下修改周期、占空比、死区等参数后，参数会自动重载，并在下一个 PWM 周期使用这些参数。在修改周期、占空比、死区等参数前设置此位为 0 将禁止自动重载，PWM 不会使用任何新的参数，输出状态保持不变，在软件修改参数完成，再设置此位为 1，新修改的参数将在下一个 PWM 周期统一使用，这样可实现多组 PWM 间的同步。

9.3.4 PWM0 周期/占空比寄存器

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，占空比 [PWM0DH:PWM0DL] =0x00，独立模式：

如果PWM0S[1:0] =00，PWM0输出低电平，PWM01输出低电平

如果PWM0S[1:0] =01，PWM0输出低电平，PWM01输出高电平

如果PWM0S[1:0] =10，PWM0输出高电平，PWM01输出低电平

如果PWM0S[1:0] =11，PWM0输出高电平，PWM01输出高电平

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] * PWM时钟源

当[PWM0PH:PWM0PL]<[PWM0DLH:PWM0DL]，并且[PWM0PH:PWM0PL] ≠0x00，独立模式：

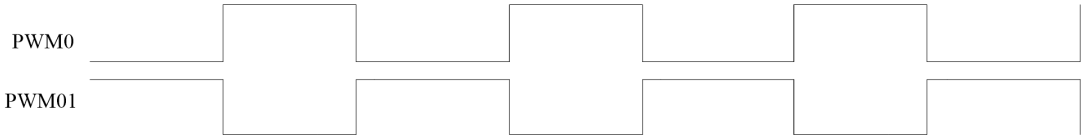
如果PWM0S[1:0] =00，PWM0输出高电平，PWM01输出高电平

如果PWM0S[1:0] =01，PWM0输出高电平，PWM01输出低电平

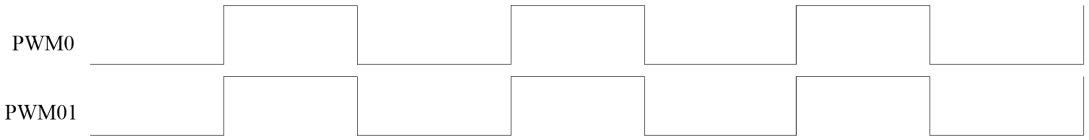
如果PWM0S[1:0] =10，PWM0输出低电平，PWM01输出高电平

如果PWM0S[1:0] =11，PWM0输出低电平，PWM01输出低电平

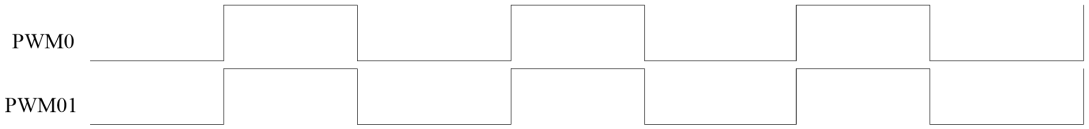
PWM0S=00& PWM0M=0：PWM0和PWM01工作于互补模式且均为高有效



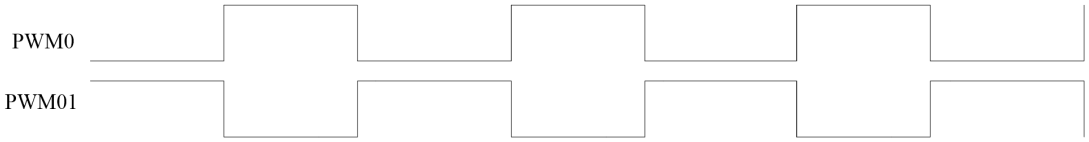
PWM0S=00& PWM0M=1：PWM0和PWM01工作于独立模式且均为高有效



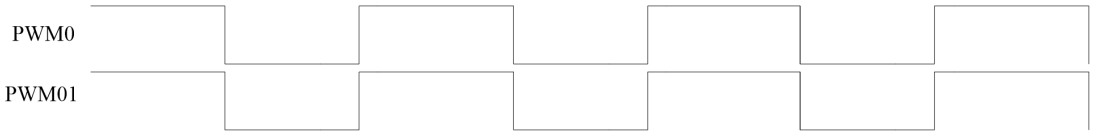
PWM0S=01& PWM0M=0：PWM0和PWM01工作于互补模式且PWM0为高有效、PWM01为低有效



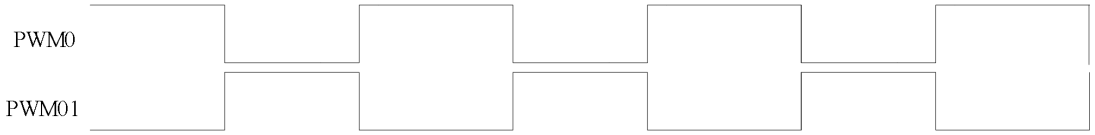
PWM0S=01& PWM0M=1：PWM0和PWM01工作于独立模式且PWM0为高有效、PWM01为低有效



PWM0S=10& PWM0M=0：PWM0和PWM01工作于互补模式且PWM0为低有效、PWM01为高有效



PWM0S=10& PWM0M=1：PWM0和PWM01工作于独立模式且PWM0为低有效、PWM01为高有效



9.3.5 死区时间

PWM0 死区时间控制寄存器

257h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DTL	PWM0DTL[7:0]							
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

258h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DTH					PWM0DTH[3:0]			
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

当 PWM0_M=1 时，PWM0 工作在 2 路独立模式，此时的死区时间寄存器被用来当做 PWM01 的占空比寄存器，即独立模式的 PWM0 可以产生 2 路周期相同，但占空比可以不同的 PWM 波形。

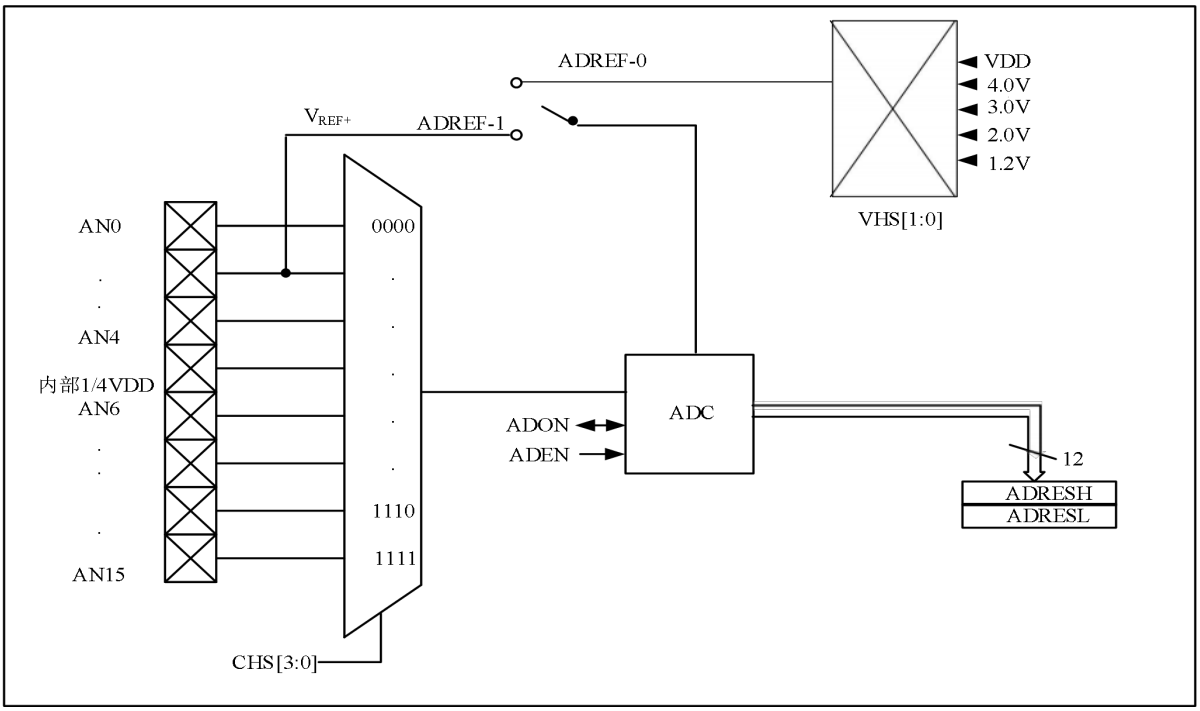
互补模式下：PWM0 死区时间 =[PWM0DTH : PWM0DTL] * PWM0 工作时钟周期；
死区时间必须小于占空比时间，死区时间与占空比时间的和必须小于 PWM0 周期。

独立模式下：PWM01占空比时间 =[PWM0DTH : PWM0DTL] * PWM0工作时钟周期。

10 模数转换（ADC)

SQL5820具有一个12位转换分辨率的模数转换器，共有7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：



10.1 A/D 引脚控制寄存器

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

08Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELH	ANSEL15	ANSEL14	-	-	ANSEL11	ANSEL10	-	-
R/W	R/W	R/W	-	-	R/W	R/W	-	-
POR值	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’ 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。

ANSEL[2:0]对应 AN2~AN0（PA2~PA0），

ANSEL[11:10]对应 AN11~AN10（PB3~PB2），ANSEL[15:14]对应 AN15~AN14（PB7、PB6）

10.2 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	-	-	CHS3	CHS2	CHS1	CHS0	ADON	ADEN
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	-	-	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	VHS2	VHS1	VHS0	ADREF
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

注：

ADCON1 寄存器的低四位 Bit3~Bit0 如果全为 0，功耗会比一般情况偏大。在 sleep 模式下为保证系统的低功耗，请避免将 ADCON1 寄存器的低四位 Bit3~Bit0 全设置为 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[2:0]	参考电压
0	000	内部VDD
0	001	内部4.0V
0	010	内部3.0V
0	011	内部2.0V
0	100	内部1.2V
1	xxx	外部参考电压

ADC数据存放格式选择

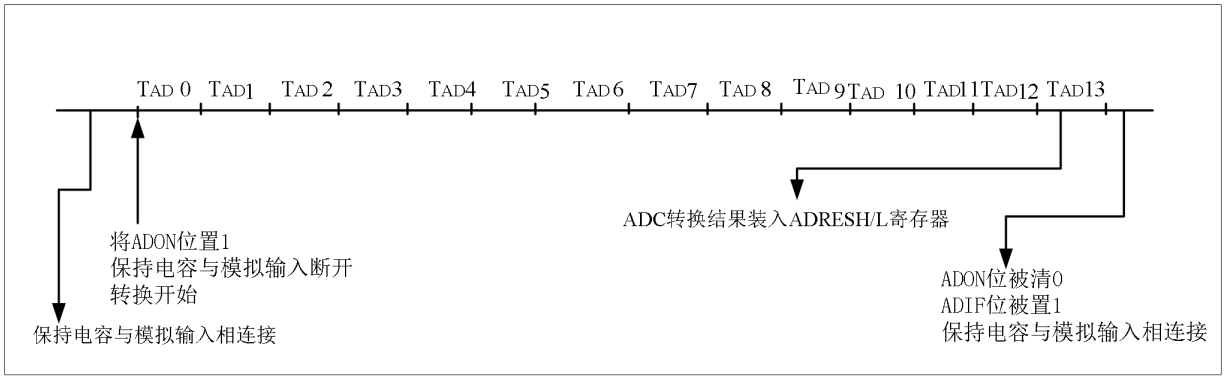
ADFM	数据格式
0	ADRESH[7:0]:ADRESL[7:4]
1	ADRESH[1:0]:ADRESL[7:0]

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。

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	500khz

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：

ADCLK软件配置表

Fsys（系统频率）	ADCS[2:0] AD 分频	ADCLK[2:0] AD 时钟
32M	000	000
	001	000
	010	000
	011	000
	100	001/010
	101	001/010
	110	011/1XX
	111	011/1XX
16M	000	000
	001	000
	010	000
	011	001/010
	100	001/010
	101	011/1XX
	110	011/1XX
	111	011/1XX
8M	000	000
	001	000
	010	001/010
	011	001/010
	100	011/1XX
	101	011/1XX
	110	011/1XX
	111	011/1XX
4M	000	000
	001	001/010
	010	001/010
	011	011/1XX
	100	011/1XX
	101	011/1XX

	110	011/1XX
	111	011/1XX
2M	000	001/010
	001	001/010
	010	011/1XX
	011	011/1XX
	100	011/1XX
	101	011/1XX
	110	011/1XX
	111	011/1XX
1M	000	001/010
	001	011/1XX
	010	011/1XX
	011	011/1XX
	100	011/1XX
	101	011/1XX
	110	011/1XX
	111	011/1XX
500K	000	011/1XX
	001	011/1XX
	010	011/1XX
	011	011/1XX
	100	011/1XX
	101	011/1XX
	110	011/1XX
	111	011/1XX
250K	000	011/1XX
	001	011/1XX
	010	011/1XX
	011	011/1XX
	100	011/1XX
	101	011/1XX
	110	011/1XX
	111	011/1XX

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

001/010：AD转换频率为1M或2MHz

011/1XX：AD转换频率为1M以下

注：

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

2、当系统频率高于1 MHz 时，仅当在休眠和绿色模式下进行转换时才推荐使用FRC时钟源。

选择FRC时钟源后，ADC需等待一个指令周期后才能启动转换操作，这使得可以执行SLEEP 指令，以降低转换期间的系统噪声。如果使能了ADC中断，转换完成时将唤醒SLEEP。如果禁止了ADC中断，尽管ADEN位仍保持为1，转换完成后ADC模块将关闭。ADC时钟源不是FRC时，尽管ADEN位仍保持为1，SLEEP指令会导致当前转换中止，ADC模块关闭。

除了选择FRC时钟源，改变系统时钟频率均会改变ADC的时钟频率，从而影响ADC转换时间。

ADEN位置1将使能ADC模块，ADON位置1将启动一次ADC转换。ADC转换完成，ADON位硬件清零，ADIF中断标志位置1，ADRESH/ADRESL寄存器值被更新。如果必须在转换完成前终止转换，可用软件将ADON位清零，ADRESH/ ADRESL寄存器将保持前次ADC转换的结果。

10.3 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]
   - 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
                                        ;配置AD通道
        BCF     STATUS,RP0              ;BANK0
```

```
        MOVLW   B'00000001'
        MOVWF   ANSELL          ;PA0作为模拟输入
        CLRF    ANSELH
        BCF     STATUS,RP0      ;BANK0
        BCF     ADCON0,CHS2
        BCF     ADCON0,CHS1
        BCF     ADCON0,CHS0
        BCF     ADCON0,VHS1
        BCF     ADCON0,VHS0     ;参考电压为内部VDD
        NOP                     ;延时
        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
        BCF     STATUS,RP0      ;BANK0
        MOVWF   NTCADLOW        ;客户应用时注意BANK
```

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。

11 串行口通信

SQL5820 具有 1 个采用 UART(Universal Asychronous 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。

SQL5820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RxD引脚。

11.1 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Fcpu/12 当UARTM0=1时，波特率是Fcpu/2
0	1	方式1	8位UART，波特率可变	（2SMOD /32)×BRT独立波特率发生器的溢出率
1	0	方式2	9位UART	（2SMOD /64)×Fcpu系统工作时钟频率
1	1	方式3	9位UART，波特率可变	（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)

- 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/禁止接收唤醒使能位

  RXWK = 1：允许RXD唤醒SLEEP

  RXWK = 0：禁止RXD唤醒SLEEP

11.2 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								
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

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。

11.3 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

11.4 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。

11.5 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

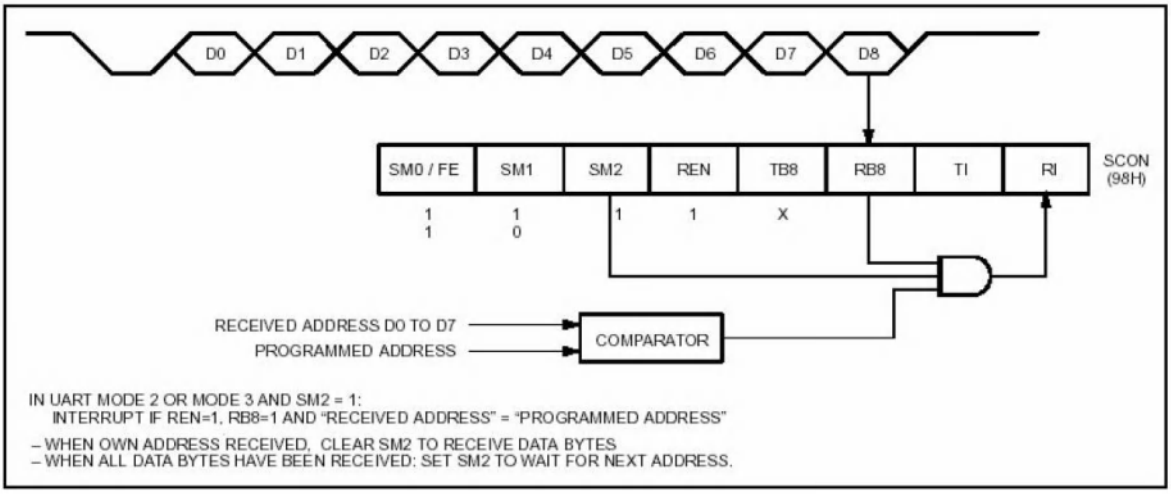
23Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SADDR								
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR的值	0	0	0	0	0	x	0	0

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 = 1110 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。

11.6 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。

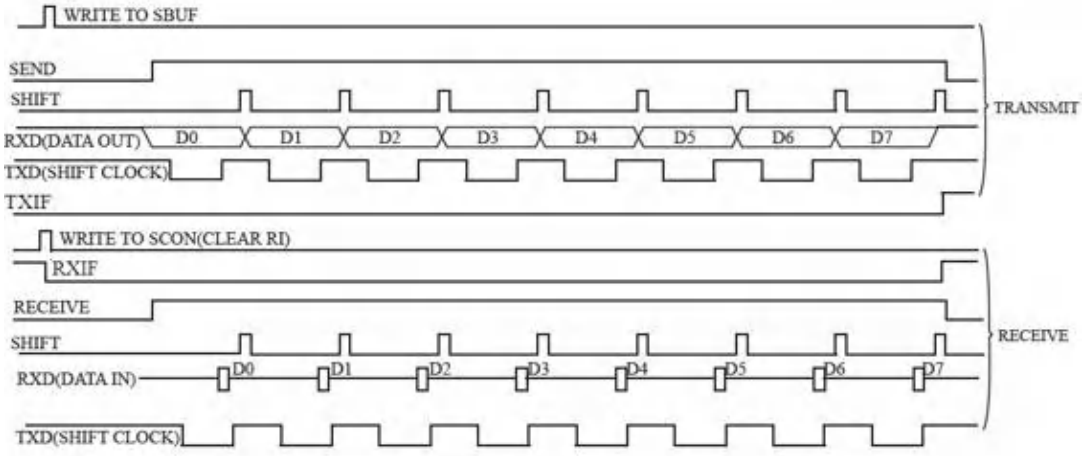
11.6.1 串行口工作模式 0：同步移位寄存器

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）。波形如下图中“发送”所示。

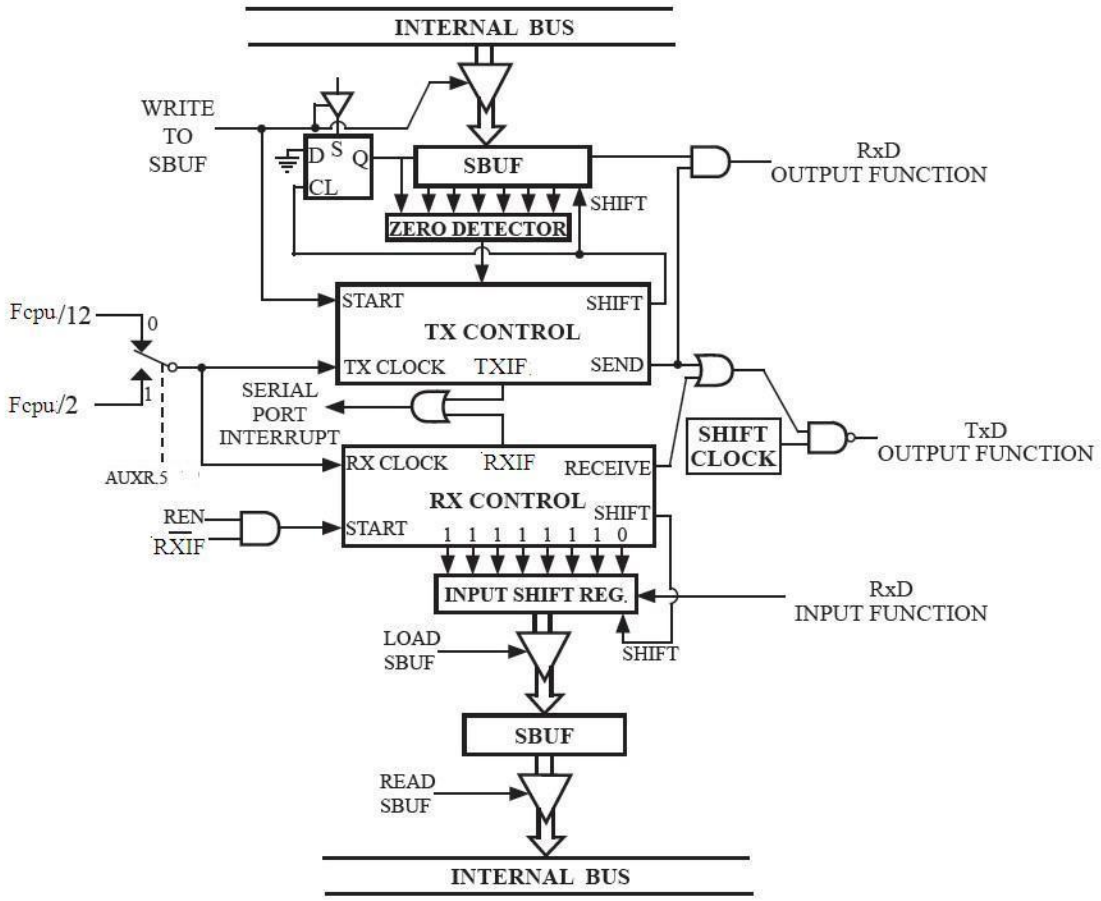
当写信号有效后，相隔一个时钟，发送控制端SEND有效(高电平)，允许RxD发送数据，同时允许TxD输出同步移位脉冲。一帧(8位)数据发送完毕时，各控制端均恢复原状态，只有TXIF保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将TXIF清0。

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)。其时序图如下图中所示：



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的示意图如下图所示：

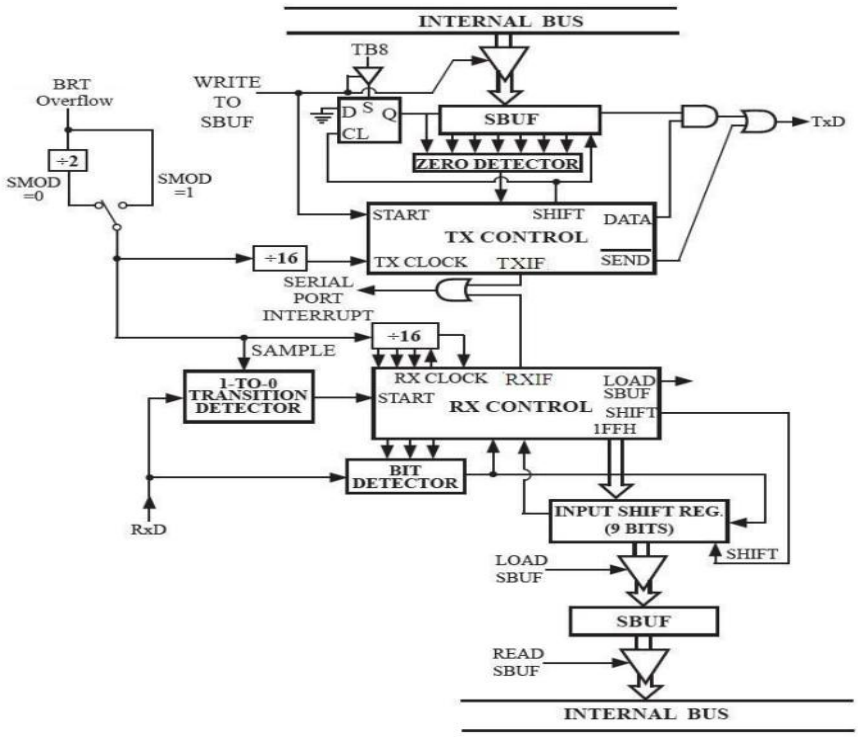


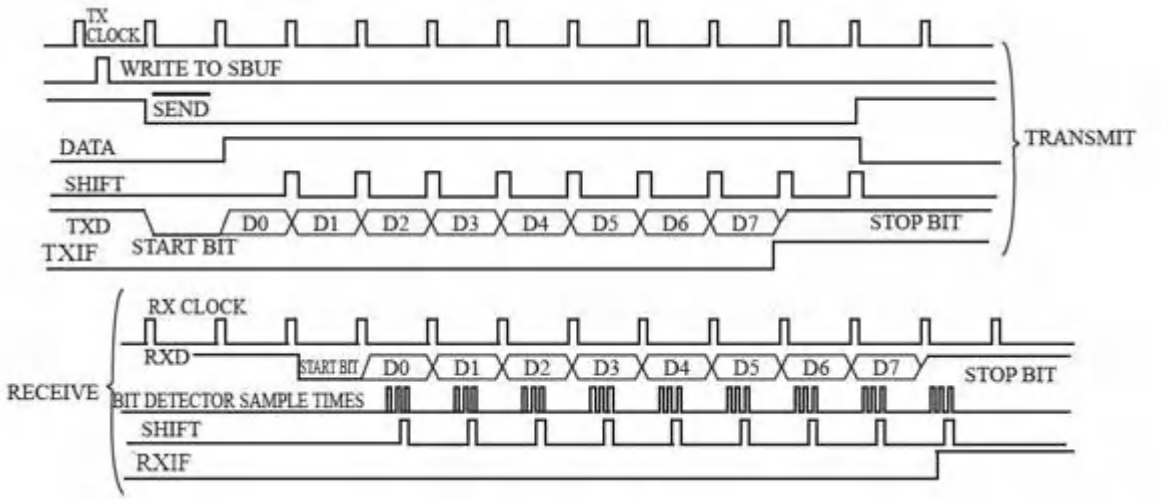
由示意图中可见，由TX和RX控制单元分别产生中断请求信号并置位TXIF=1或RXIF =1，经“或门“送主机请求中断，所以主机响应中断后必须软件判别是TXIF还是RXIF请求中断，必须软件清0中断请求标志位TXIF或RXIF。

11.6.2 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。

下图为串行模式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植装入移位寄存器。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。

模式1的波特率为：

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)。

11.6.3 串行口工作模式 2：9 位 UART，波特率固定

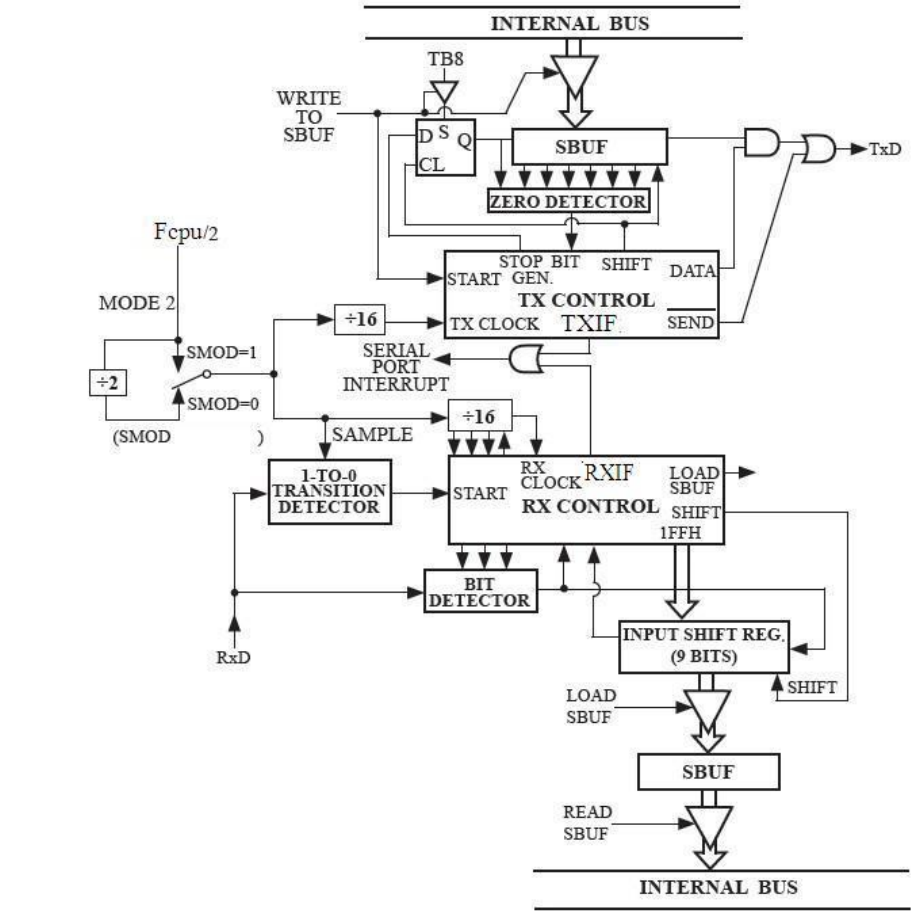
当SM0、SM1两位为10时，串行口工作在模式2。串行口工作模式2为9位数据异步通信UART模式，其一帧的信息由11 位组成：1位起始位，8位数据位(低位在先)，1位可编程位(第9位数据)和1位停止位。发送时可编程位(第9位数据)由SCON中的TB8提供，可软件设置为1或0，或者将奇/偶校验值装入TB8（TB8既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第9位数据装入SCON的RB8。TxD为发送端口，RxD为接收端口，以全双工模式进行接收/发送。

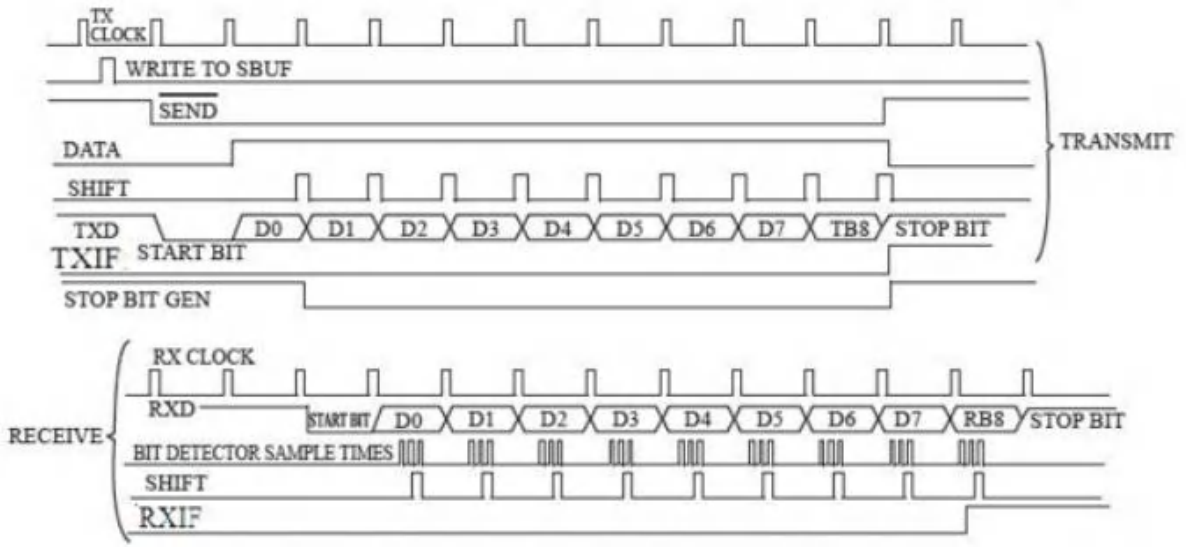
模式2的波特率为：

串行通信模式2波特率=2SMOD/64×(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。

下图为串行通信模式2的功能结构示意图及其接收/发送时序图：





由上图可知，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。

11.6.4 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：

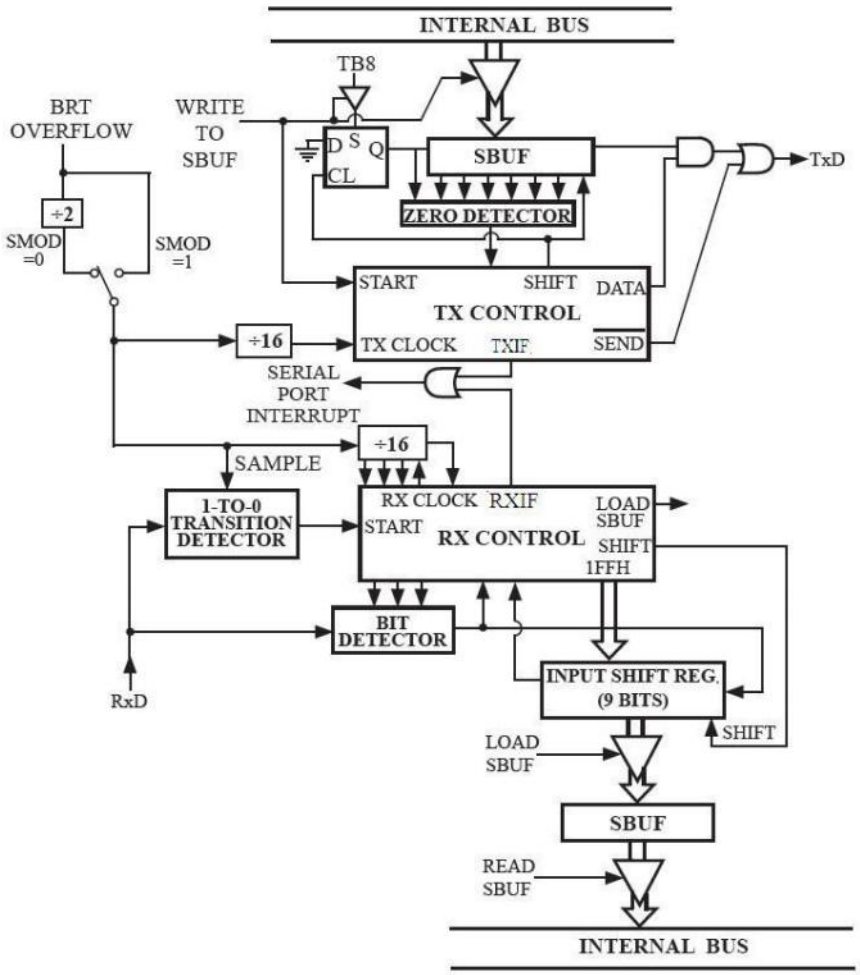
串行通信模式3波特率=2SMOD/32×(BRT独立波特率发生器的溢出率)。

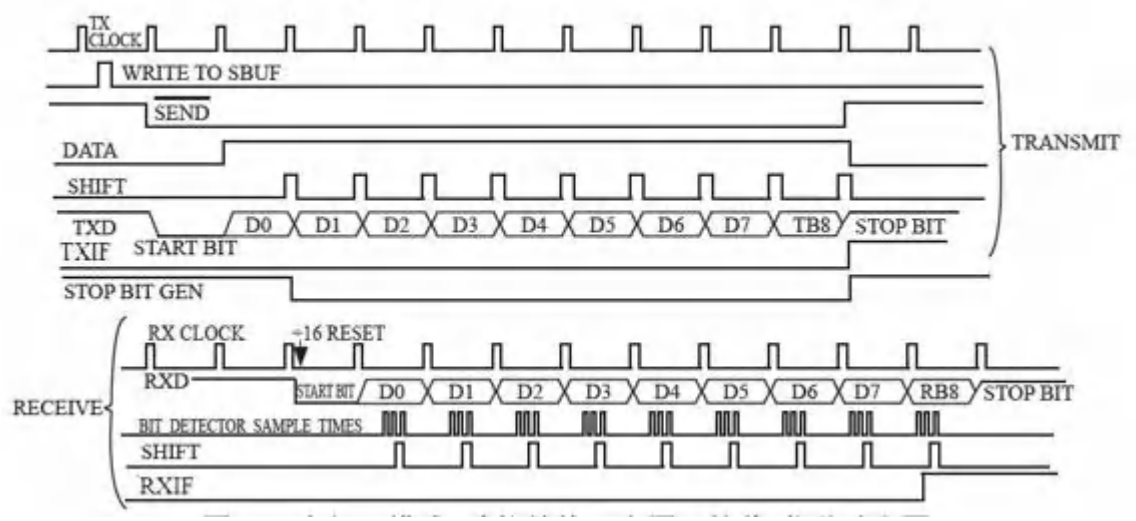
当BRTx12 = 0时， BRT独立波特率发生器的溢出率 = Fcpu/12/(256 - BRT)；

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。

下图为串行口工作模式3的功能结构示意图及其接收/发送时序图：





由下图可知，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。

11.7 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BRT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(Fcpu/Baud0/32/12 + 0.5)

b) 1T 模式的计算公式： RELOAD = 256 - INT(Fcpu/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 = Fcpu/(256 - RELOAD)/32/12 12T 模式

b) Baud = Fcpu/(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: Fcpu= 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

12 软件LCD驱动

SQL5820 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。

12.1 相关寄存器

LCDCON 寄存器

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

Bit[4] **FRAM**：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

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。

12.2 软件 LCD 操作说明

1 设置LCD总使能，LCDEN=1；这是总的使能，打开电阻分压电路；

2 设置驱动能力，选择不同的电阻分压RLCD[1:0]；

3 对IO口使能COM或SEG，COMXEN=1,SEGXEN=1;这是具体设置某个IO的状态，使能LCD的模拟通道；

4 设定frame0，用于确定点亮和非点亮电平；

5 设置定时开始，设置COM[3:0]=0001，设置SEG[19:0]=XXX，等待定时结束；

6 设置定时开始，设置COM[3:0]=0010，设置SEG[19:0]=XXX，等待定时结束；

7 设置定时开始，设置COM[3:0]=0100，设置SEG[19:0]=XXX，等待定时结束；

8 设置定时开始，设置COM[3:0]=1000，设置SEG[19:0]=XXX，等待定时结束；

9 设定frame0，用于确定点亮和非点亮电平；

10设置定时开始，设置COM[3:0]=0001，设置SEG[19:0]=XXX，等待定时结束；

11设置定时开始，设置COM[3:0]=0010，设置SEG[19:0]=XXX，等待定时结束；

12设置定时开始，设置COM[3:0]=0100，设置SEG[19:0]=XXX，等待定时结束；

13设置定时开始，设置COM[3:0]=1000，设置SEG[19:0]=XXX，等待定时结束。

注意：

1 驱动加速：在每次切换COM时，设置最大的驱动能力RLCD [1:0]=11，等一段时间如50us，重新设置最小的驱动能力ISEL[1:0]=00；

2 不用的SEG口，可以不设置，SEG数不限定于20个；

3 不用的 COM 口，软件直接跳过，不需要延时等待，COM 数不限定于 4 个。

13 指令表

助记符	操作数	说明	周期数	受影响状态位
ADDWF	f,d	W 和 f 相加	1	C，DC，Z
ADDLW	k	将立即数和 W 相加	1	C，DC，Z
SUBWF	f,d	f 减去 W	1	C，DC，Z
SUBLW	k	立即数减去 W	1	C，DC，Z
DAW	-	W 寄存器值进行 BCD 调整	1	C，DC
ANDWF	f,d	W 和 f 作逻辑与运算	1	Z
ANDLW	k	立即数和 W 作逻辑与运算	1	Z
IORWF	f,d	W 和 f 作逻辑或运算	1	Z
IORLW	k	立即数和 W 作逻辑或运算	1	Z
XORWF	f,d	W 和 f 作逻辑异或运算	1	Z
XORLW	k	立即数和 W 作逻辑异或运算	1	Z
COMF	f,d	f 取反	1	Z
CLRW	-	将 W 清零	1	Z
CLRF	f	将 f 清零	1	Z
INCF	f,d	f 加 1	1	Z
INCFSZ	f,d	f 加 1，为 0 则跳过	1(2)	-
DECF	f,d	f 减 1	1	Z
DECFSZ	f,d	f 减 1，为 0 则跳过	1(2)	-
BCF	f,d	将 f 中的 d 位清 0	1	-
BSF	f,d	将 f 中的 d 位置 1	1	-
BTFSC	f,d	检测 f 中的 d 位，为 0 则跳过	1(2)	-
BTFSS	f,d	检测 f 中的 d 位，为 1 则跳过	1(2)	-
MOVWF	f	将 W 的内容传送到 f	1	-
MOVF	f,d	将 f 的内容送到目标寄存器	1	Z
MOVLW	k	将立即数 k 传送到 W	1	-
RLF	f,d	对 f 执行带进位的循环左移	1	C
RRF	f,d	对 f 执行带进位的循环右移	1	C
SWAPF	f,d	将 f 的两个半字节进行交换	1	-
CALL	k	调用子程序	2	-
GOTO	k	无条件跳转	2	-
RETFIE	-	从中断返回	2	GIE
RETURN	-	从子程序返回	2	-
RETLW	k	返回时将立即数传送到 W	2	-
CLRWDT	-	清零看门狗定时器	1	TO，PD
SLEEP	-	进入待机模式	1	TO，PD
NOP	-	空操作	1	-

14 电性参数

14.1 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

14.2 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	1.8	—	5.5	V
		—	Fsys = 16MHz，2T	2.6	—	5.5	V
I_{DD1}	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
I_{DD2}	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
I_{DD3}	工作电流	3V	Fsys = 32KHz，4T，低频模式，WDT 禁止，无负载	—	3.0	—	μA
		5V		—	10.0	—	μA
I_{DD4}	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
I_{SB4}	静态电流	3V	休眠模式，WDT 禁止，无负载	—	—	1	μA
		5V		—	—	1	μA
V_{IL1}	输入低电平	—	输入口	VSS	—	0.3VDD	V
V_{IH1}	输入高电平	—	输入口	0.7VDD	—	VDD	V
V_{IL2}	输入低电平	—	施密特输入口	VSS	—	0.2VDD	V
V_{IH2}	输入高电平	—	施密特输入口	0.8VDD	—	VDD	V
V_{BOR1}	低电压复位 1.5V	—	—	1.4	1.5	1.6	V
V_{BOR2}	低电压复位 1.8V	—	—	1.7	1.8	1.9	V
V_{BOR3}	低电压复位 2.0V	—	—	1.9	2.0	2.1	V
V_{BOR4}	低电压复位 2.2V	—	—	2.1	2.2	2.3	V
V_{BOR5}	低电压复位 2.4V	—	—	2.3	2.4	2.5	V
V_{BOR6}	低电压复位	—	—	2.6	2.7	2.8	V

	2.7V						
V_{BOR7}	低电压复位 3.6V	—	—	3.5	3.6	3.7	V
V_{LVD0}	低电压标志	—	—	2.3	2.4	2.5	V
V_{LVD1}	低电压标志	—	—	3.5	3.6	3.7	V
I_{OL1}	(DRENxn=00)	3V	VOL=VSS+0.6V	—	—	—	mA
		5V		—	50	—	mA
I_{OL2}	(DRENxn=01)	3V	VOL=VSS+0.6V	—	—	—	mA
		5V		—	10	—	mA
I_{OL3}	(DRENxn=10)	3V	VOL=VSS+0.6V	—	—	—	mA
		5V		—	25	—	mA
I_{OL4}	(DRENxn=11)	3V	VOL=VSS+0.6V	—	—	—	
		5V		—	5	—	
I_{OH1}	(DRENxn=00)	3V	VOH= VDD-0.6V			—	mA
		5V		—	30	—	mA
I_{OH2}	(DRENxn=01)	3V	VOH= VDD-0.6V			—	mA
		5V		—	8	—	mA
I_{OH3}	(DRENxn=10)	3V	VOH=VDD-0.6V	—		—	mA
		5V		—	20	—	mA
I_{OH4}	(DRENxn=11)	3V	VOH=VDD-0.6V	—		—	mA
		5V		—	3	—	mA
R_{PH1}	内部上拉电阻	5V	可编程上拉电阻(PA2 除外)	5	10	15	kΩ
R_{PD1}	内部下拉电阻	5V	可编程下拉电阻（PA2，PB5 除外）	5	10	15	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

14.3 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）				
F_{RCH}	高频内部 RC 振荡器	5V	—	31.68	32	32.32	MHz
F_{RCH}	高频内部 RC 振荡器		2.0-5.5V	31.36	32	32.64	MHz
F_{RCL}	低频内部 RC 振荡器	5V	—	16	32	48	KHz
F_{OSH}	外部高频晶振	—	2.0~5.5V	4	—	20	MHz

FOSL	外部低频晶振	—	2.0~5.5V	—	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

15 开发工具

15.1 OTP 烧录器（HC-PM18 5.0）

- PM18 5.0：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。

15.2 HC-IDE

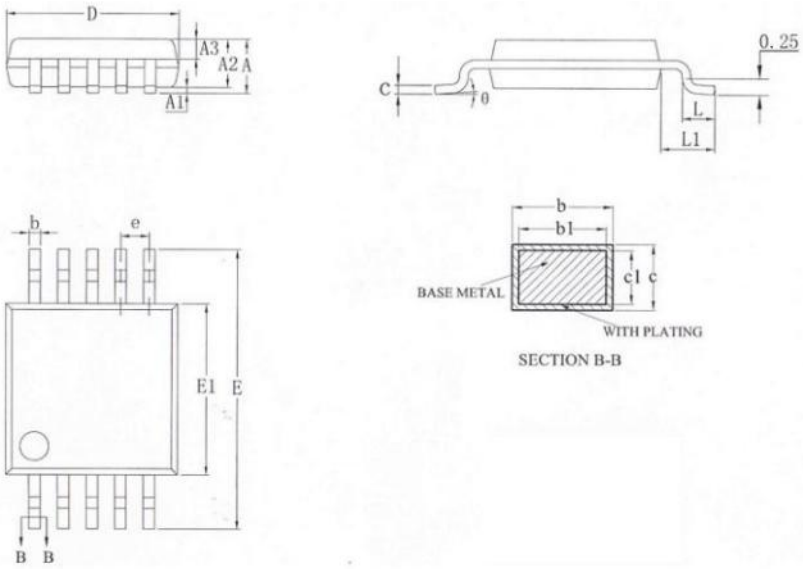
Holychip 8 位单片机的集成开发环境 HC-IDE 包括编译器、HC-PM18 下载烧录软件。

- HC-PM18：V5.0.50
- HC-IDE：V3.0.50 (支持汇编/C 语言)

注：
1、 详情请参考 HC-IDE 用户手册。
2、 IDE 更新请关注芯圣官网：http://www.holychip.cn/

16 封装信息

MSOP10



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	—	—	1.10
A1	0.05	—	0.15
A2	0.75	0.85	0.95
A3	0.30	0.35	0.40
b	0.18	—	0.26
b1	0.17	0.20	0.23
c	0.15	—	0.19
c1	0.14	0.15	0.16
D	2.90	3.00	3.10
E	4.70	4.90	5.10
E1	2.90	3.00	3.10
e	0.50BSC		
L	0.40	—	0.70
L1	0.95REF		
θ	0	—	8°

17 修改记录

版本	日期	描述
Ver1.00	2021-09	第一版
Ver1.01	2023-05	修改 MSOP10 引脚图中 6 脚的 AN15

芯圣电子

2021年02月