BL08M003

数据手册

24引脚8位
ADC型MTP单片机

目 录

1 产品简介

BL08M003 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 8K×16 位 MTP-ROM，256×8 位的数据寄存器（RAM），256*16 位的 EEPROM，3 组双向 I/O 口，3 个 Timer(2 个定时器/计数器+1 个定时器)，一个 PWM01 模块、一个 PWM23 模块、一个 PWM4 模块、一个比较器(CMP1)、一个运算放大器 OP(可配置成比较器 CMP2)、一个 UART、1 个 SPI、1 个 IIC、一个 LCD 模块。一个 22+3 通道的 12 位模数转换器，多个系统时钟，四种系统工作模式(高频模式、低频模式、休眠模式、绿色模式)以及多个中断源。

1.1 功能特性

- 存储器配置
  - 程序存储器（MTP ROM）空间：8K*16 位
  - 数据存储器（RAM）空间：256*8 位
  - EEPROM 空间：256*16 位
- 强大的指令系统
  - 时钟系统可设（2T）
  - 36 条高性能精简指令
  - 大部分指令皆可在一个机器周期完成
  - 支持立即、直接和间接寻址模式
- 8 级堆栈缓存器
- I/O 引脚配置
  - 最多 22 个 IO 口均具有可编程的上下拉
  - 3 组双向 I/O 口：PORTA，PORTB ，PORTC
  - 具有唤醒功能的电平变化中断端口：PORTA，PORTB，PORTC
  - 具有唤醒功能的外部中断引脚：PORTB<0>，PORTB<1>可设置触发边沿
  - 驱动能力：PORTA/PORTB 拉灌电流：7.5mA/25mA，PORTC 拉灌电流：14mA/42.5mA
- BOR
  - 1.8V/2.0/2.4V/3V
- 中断
  - 定时器中断：Timer0、Timer1 和 Timer2
  - INT0、INT1 外部中断
  - CMP1/2 中断
  - 端口电平变化中断
  - ADC 中断
  - PWM01、PWM23、PWM4 中断
  - UART 中断
  - SPI 中断
  - IIC 中断
- 定时器
  - 看门狗计数器（WDT）
  - Timer0：带有 8 位预分频器的 8 位定时器/计数器
  - Timer1：16 位定时器/计数器
  - Timer2：带有 8 位周期寄存器的 8 位定时器
- PWM01
  - PWM0 和 PWM1 为一组，12 位精度，共周期，独立占空比，可配中心沿对齐、边沿对齐两种方式；可软件配置成互补输出；可配置成一路 LED 级联控制
- PWM23
  - PWM2 和 PWM3 为一组，12 位精度，共周期，独立占空比，可配中心沿对齐、边沿对齐两种方式；可软件配置成互补输出；可配置成一路 LED 级联控制
- PWM4
  - PWM4 单独一路，12 位精度，独立周期，独立占空比

- 系统时钟
  - 内建高精度 16MHz RC 时钟
  - 内建 40KHz 低频 RC 时钟
  - 外部低频晶体振荡器：32.768KHz
- 通讯模块
  - 1 个 UART
  - 1 个 SPI
  - 1 个 IIC
- ADC
  - 12 位转换分辨率
  - 最多 25 个模拟输入通道（22 个外部 ADC 输入，1 个内部 1/4 VDD 检测、1 个内部 GND 通道、1 个内部 OPOUT 输入）
  - 内部参考电压（VDD、1.3V、2V、3.0V）
- 支持带电烧录功能
  - PGD、PGC
- 1 路比较器
  - CMP1 正端可选：PC1/VDD 经过内部电阻分压后的电压/OPOUT
  - CMP1 负端可选：PB3/PC5/PA3/PA4/VDD 经过内部电阻分压后的电压/1.3V/OPOUT
- 1 路运算放大器
  - OP 正端：PC6
  - OP 负端可选：PC7/PB6/PA5/PB0
  - 可配置成一路比较器 CMP2
- 软件 LCD
  - 支持 1/2Bias 和 1/3Bias
- 工作模式
  - 高频模式
  - 低频模式
  - 休眠模式
  - 绿色模式
- 复位
  - 上电复位
  - BOR 欠压复位
  - WDT 溢出复位
- 封装
  - SOP16/QFN16(3*3)
  - TSSOP20
  - QFN20(3*3)
  - SSOP24
  - QFN24(4*4)
- 工作电压
  - 2.7V~5.5V（F_{CPU} =8MHZ）
  - 2.0V~5.5V（F_{CPU} =4MHZ 及以下）

1.2 应用注意说明

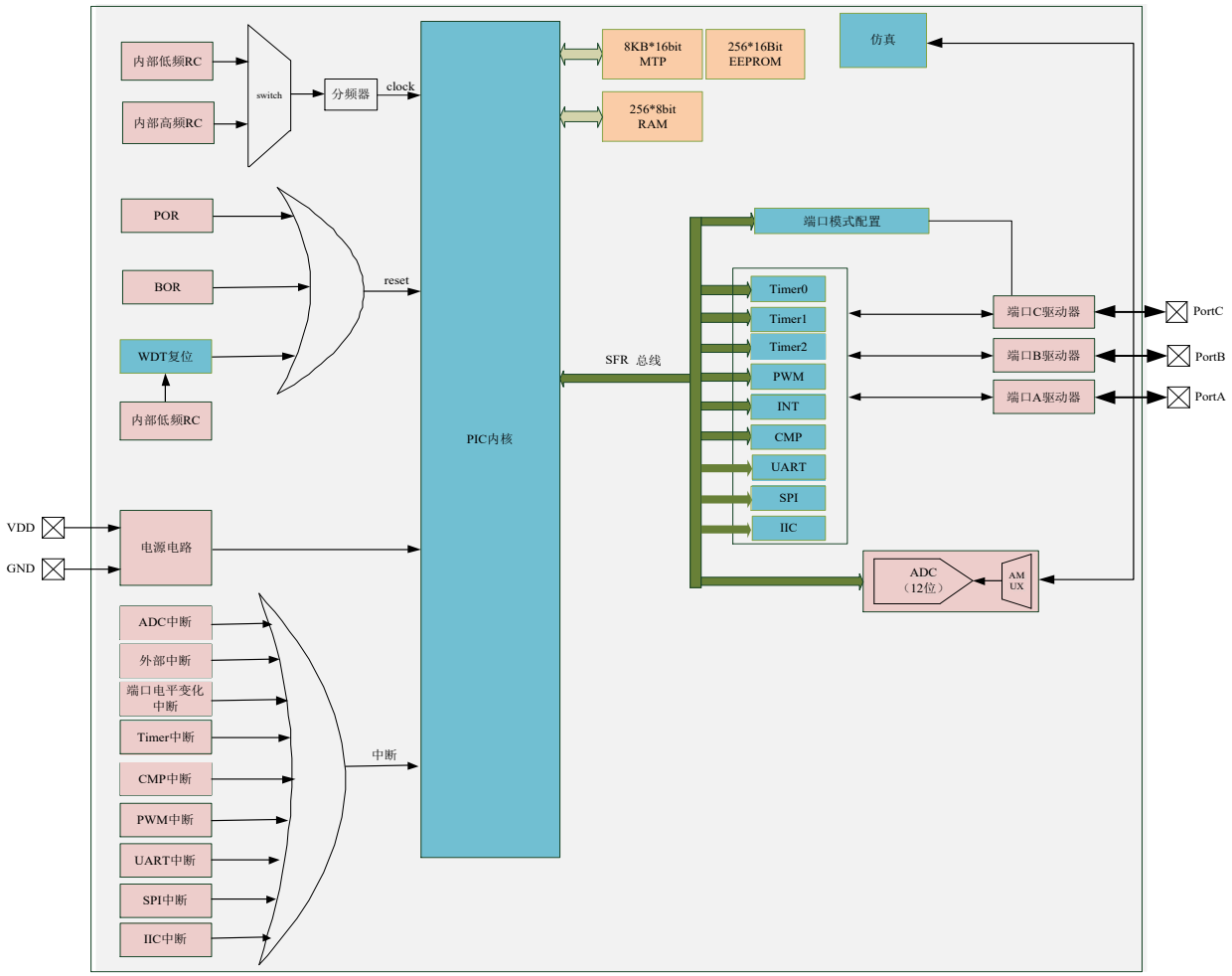
1、由于 PWM 模块时钟源为 RC 32M，故 PWM 模块只能在高频模式下才能正常使用
2、当使用外部晶振时，PC0 和 PC1 口只能当做外部晶振用，无法使用两个口上的其他资源，配置无效
3、VDD 上电需等待电源稳定后，再进行读写 EEPROM 操作
4、写 EEPROM 过程是强制打开高频时钟的
5、使能带电烧录（DBG0EN=1）后，会强制打开高频时钟，且 PGC/PGD 两个 IO 仅作为编程功能，通用功能屏蔽

6、IO 模式中，输入模式读端口，输出模式读 IO 寄存器
7、仿真时，烧录口 PA0 和 PA1 无功能，仅与仿真板通信
8、当使用 OP 时，若用内部放大电路，OP 负端输入选择的端口需要浮空，不能使用此 IO 其他功能资源
9、每次上电或复位后使用 ADC 时，需要进行 OFFSET 校准，具体校准步骤参考 9.3 ADC-OFFSET 校准章节
10、F_{CPU} = 8MHZ，VDD 电压必须大于 2.7V
11、Option 中 IP 低功耗使能仅在 CPU 时钟为低频时钟时使用

1.3 芯片选型表

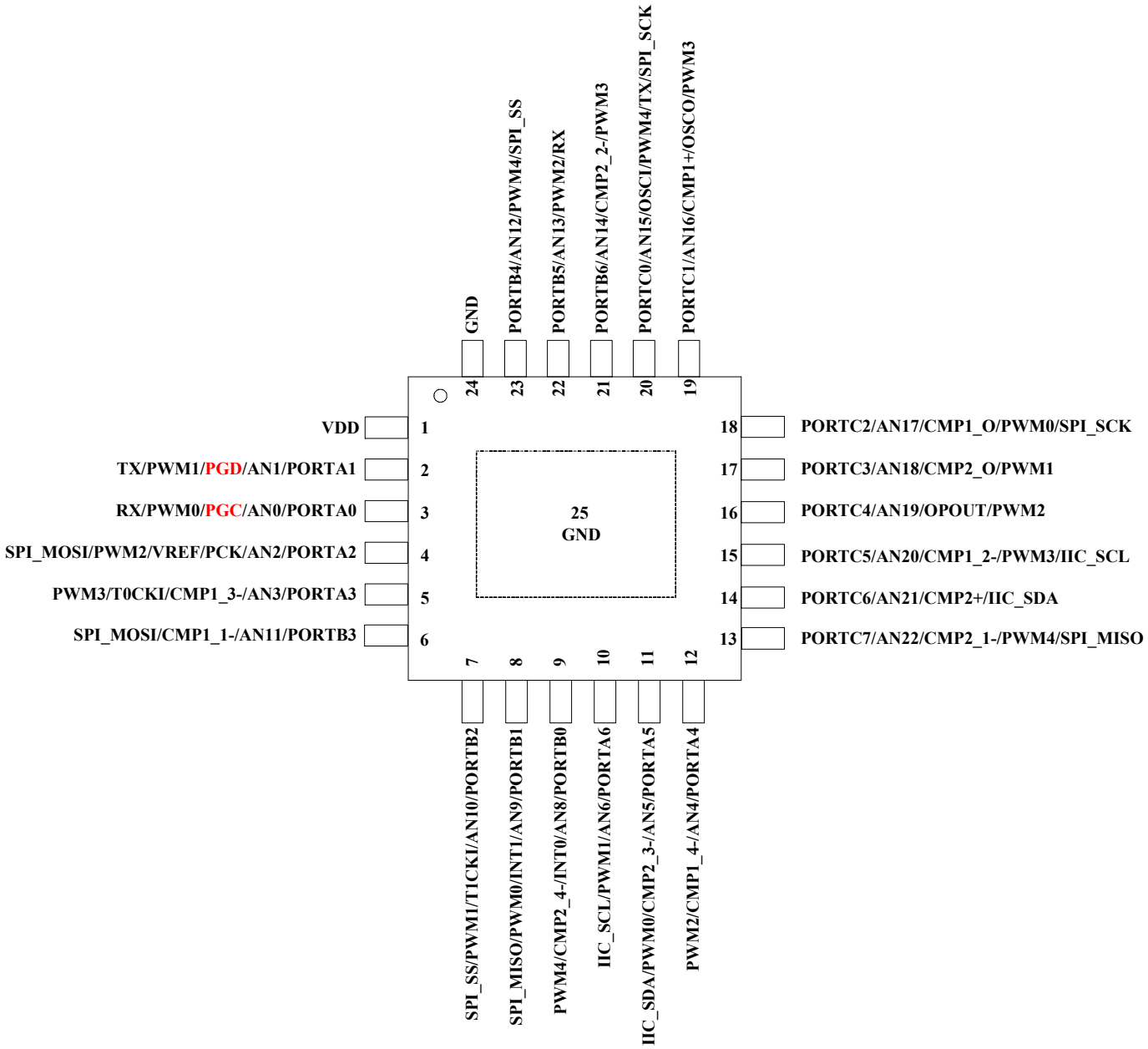
Product	ROM Bytes	RAM Bytes	EEPROM Bytes	Max Freq	I/O	A/D	Timer	PWM/LED 级联	INT	UART	IIC	SPI	WDT	LCD	OP	CMP	Package
BL08M003	8K*16bit	256*8bit	256*16bit	16M2T	22	22+3	8bit*2 16bit*1	12bit*2 组 +12bit*1 路	2	1	1	1	1	1	1	1	QFN24
BL08M003	8K*16bit	256*8bit	256*16bit	16M2T	18	18+3	8bit*2 16bit*1	12bit*2 组 +12bit*1 路	2	1	1	1	1	1	1	1	QFN20
BL08M003	8K*16bit	256*8bit	256*16bit	16M2T	14	14+3	8bit*2 16bit*1	12bit*2 组 +12bit*1 路	2	1	1	1	1	1	1	1	QFN16
BL08M003	8K*16bit	256*8bit	256*16bit	16M2T	14	14+3	8bit*2 16bit*1	12bit*2 组 +12bit*1 路	2	1	1	1	1	1	1	1	SOP16
BL08M003	8K*16bit	256*8bit	256*16bit	16M2T	18	18+3	8bit*2 16bit*1	12bit*2 组 +12bit*1 路	2	1	1	1	1	1	1	1	TSSOP20
BL08M003	8K*16bit	256*8bit	256*16bit	16M2T	22	22+3	8bit*2 16bit*1	12bit*2 组 +12bit*1 路	2	1	1	1	1	1	1	1	SSOP24

1.4 系统框图

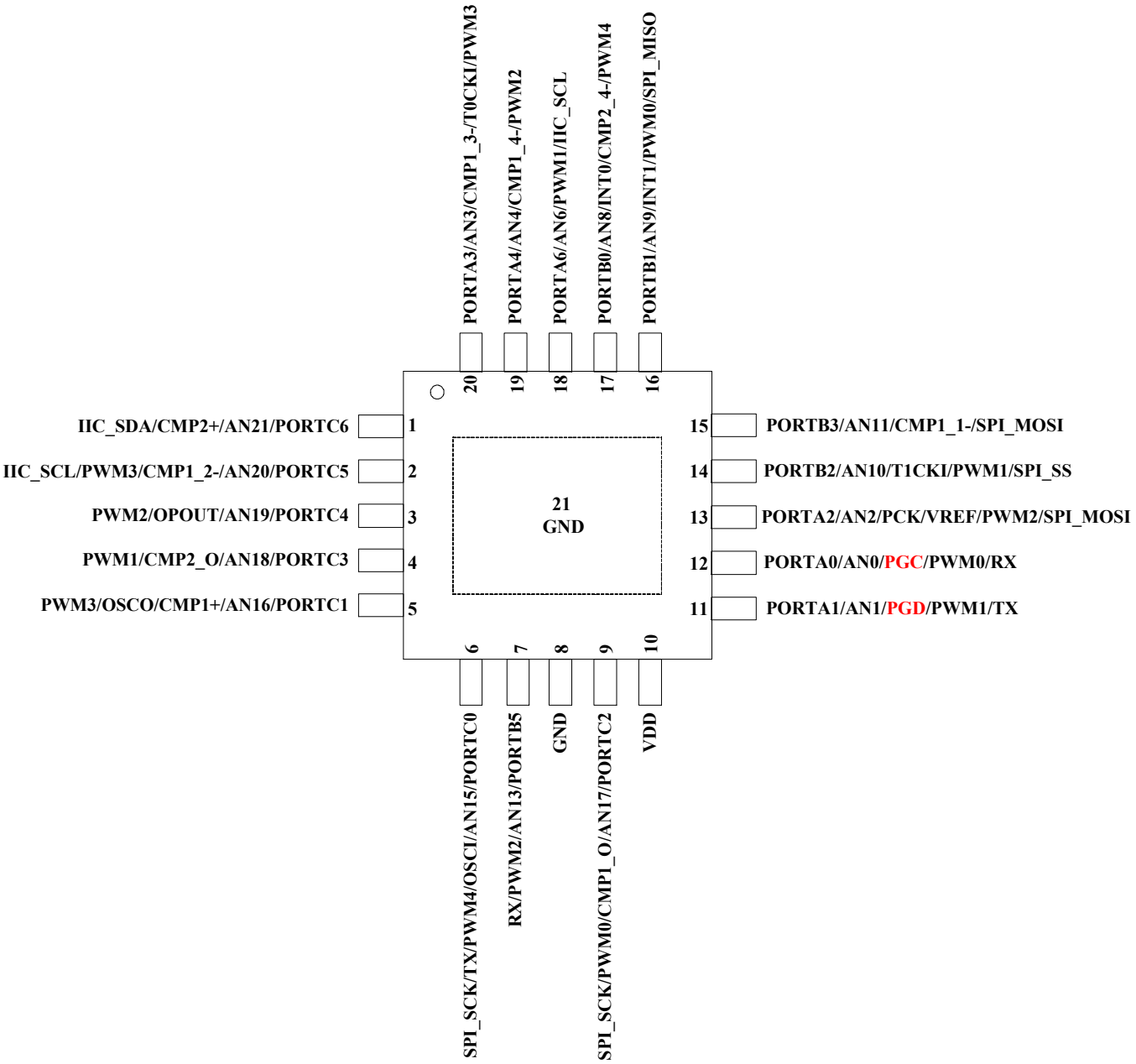

内部低频RC
内部高频RC
switch
分频器
clock
8KB*16bit MTP
256*16Bit EEPROM
256*8bit RAM
仿真
POR
BOR
WDT复位
内部低频RC
reset
端口模式配置
Timer0
Timer1
Timer2
PWM
INT
CMP
UART
SPI
IIC
SFR 总线
端口C驱动器
端口B驱动器
端口A驱动器
PortC
PortB
PortA
PIC内核
VDD
GND
电源电路
ADC（12位）
AMUX
ADC中断
外部中断
端口电平变化中断
Timer中断
CMP中断
PWM中断
UART中断
SPI中断
IIC中断
中断

1.5 引脚图

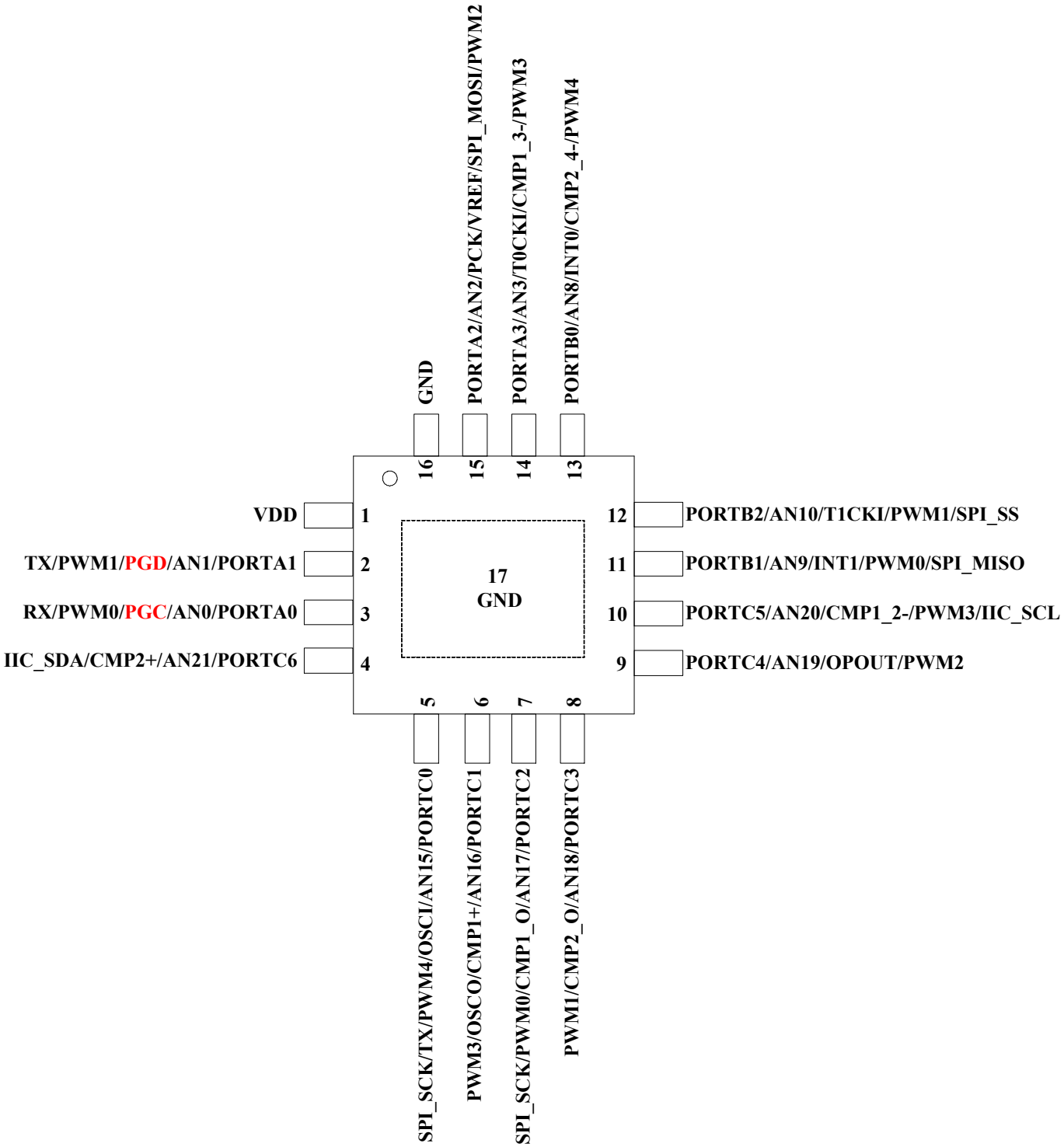
1.5.1 QFN24 引脚图



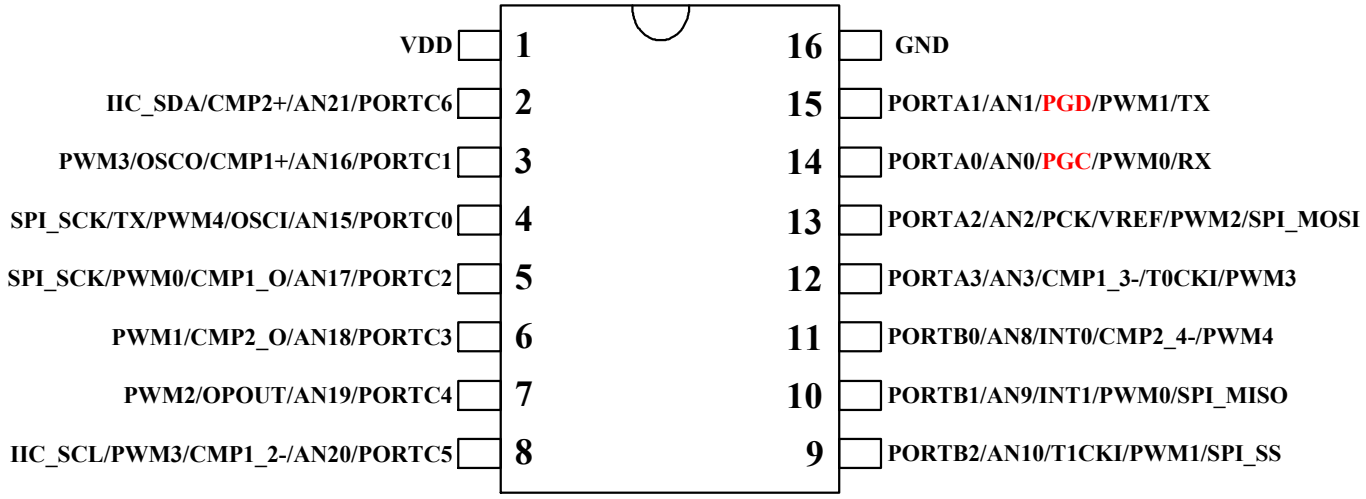
1.5.2 QFN20 引脚图



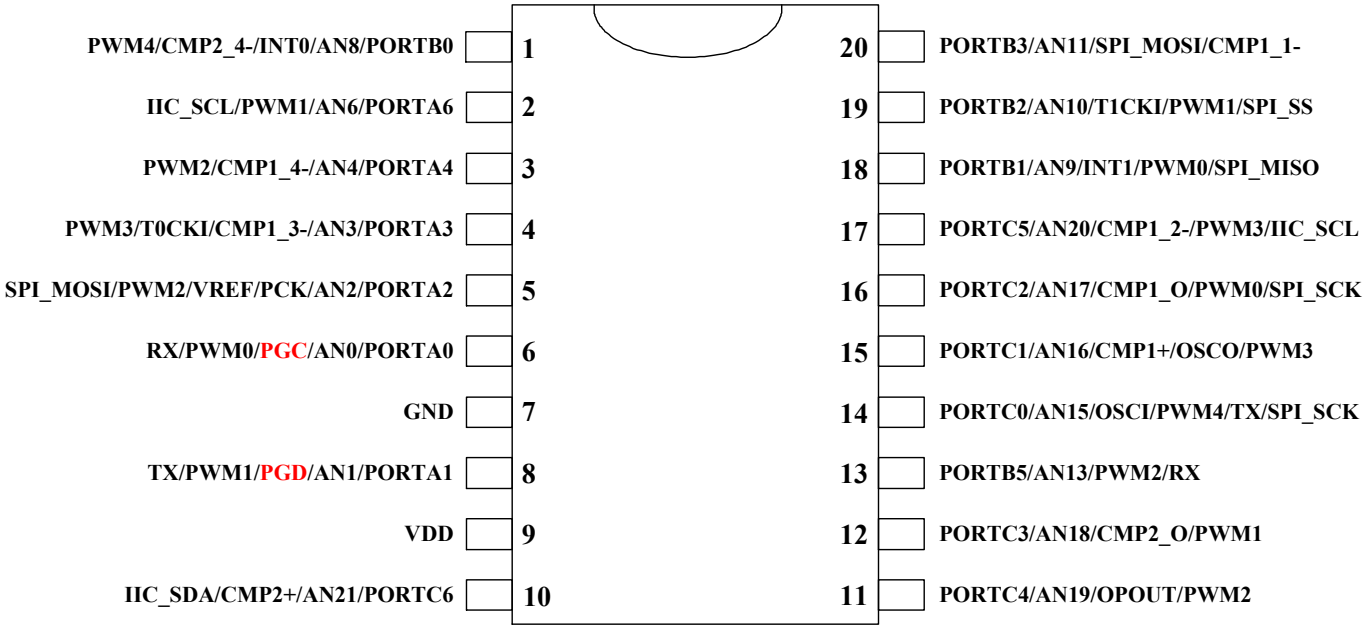
1.5.3 QFN16 引脚图



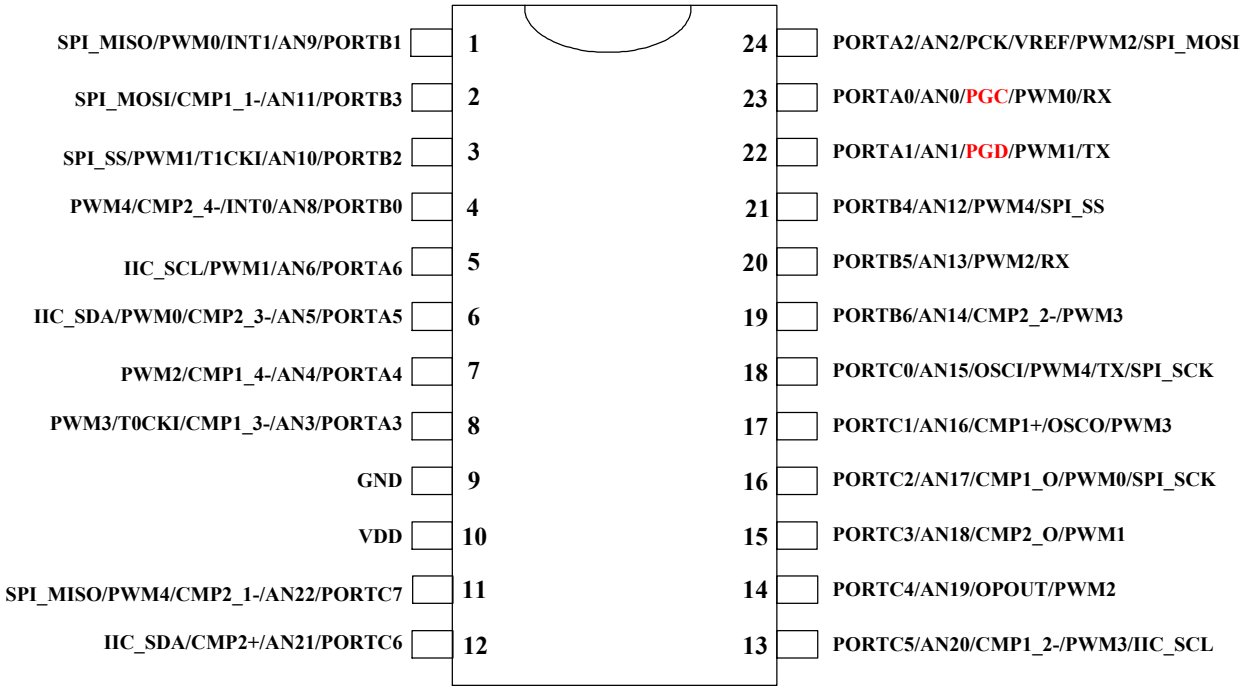
1.5.4 SOP16 引脚图



1.5.5 TSSOP20 引脚图



1.5.6 SSOP24 引脚图



1.6 引脚描述

1.6.1 QFN24 引脚描述

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

25PIN 脚位	名称	类型	说明
1	VDD	P	电源
2	PORTA1 AN1 PWM1 PGD TX	I/O AN O 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 1 输入口 PWM1 输出口 编程数据输入/输出口 PGD UART 发送口 TX
3	PORTA0 AN0 PWM0 PGC RX	I/O AN O I/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 0 输入口 PWM0 输出口 编程时钟输入口 PGC UART 接收口 RX
4	PORTA2 AN2 PWM2 PCK VREF SPI_MOSI	I/O AN O O AN I/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 2 输入口 PWM2 输出口 编程内部 RC 输出口 PCK ADC 内部参考电压校准口 SPI 通信口 MOSI
5	PORTA3 AN3 CMP1_3- PWM3 T0CKI	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 3 输入口 CMP1 负端输入口 3 PWM3 输出口 T0 定时器外部时钟输入口
6	PORTB3 AN11 CMP1_1- SPI_MOSI	I/O AN AN I/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 CMP1 负端输入口 1 SPI 数据口 MOSI
7	PORTB2 AN10 T1CKI PWM1 SPI_SS	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 10 输入口 T1 定时器外部时钟输入口 PWM1 输出口 SPI 片选信号 SS
8	PORTB1 AN9 INT1 PWM0 SPI_MISO	I/O AN I O I/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 9 输入口 外部中断口 1 PWM0 输出口 SPI 数据口 MISO
9	PORTB0 AN8 INT0 CMP2_4- PWM4	I/O AN I AN 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 8 输入口 外部中断口 0 CMP2 负端输入口 4 PWM4 输出口
10	PORTA6 AN6 PWM1 IIC_SCL	I/O AN O I/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 6 输入口 PWM1 输出口

			IIC 时钟口 SCL
11	PORTA5 AN5 CMP2_3- PWM0 IIC_SDA	I/O AN AN O I/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 5 输入口 CMP2 负端输入端口 3 PWM0 输出口 IIC 数据口 SDA
12	PORTA4 AN4 CMP1_4- PWM2	I/O AN AN 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 4 输入口 CMP1 负端输入口 4 PWM2 输出口
13	PORTC7 AN22 CMP2_1- PWM4 SPI_MISO	I/O AN AN O I/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 22 输入口 CMP2 负端输入端口 1 PWM4 输出口 SPI 数据口 MISO
14	PORTC6 AN21 CMP2+ IIC_SDA	I/O AN AN I/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 21 输入口 CMP2 正端输入端口 IIC 数据口 SDA
15	PORTC5 AN20 CMP1_2- PWM3 IIC_SCL	I/O AN AN O I/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 20 输入口 CMP1 负端输入口 2 PWM3 输出口 IIC 时钟口 SCL
16	PORTC4 AN19 OPOUT PWM2	I/O AN AN 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 19 输入口 OP 输出口 PWM2 输出口
17	PORTC3 AN18 CMP2_O PWM1	I/O AN 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 18 输入口 CMP2_O 输出口 PWM1 输出口
18	PORTC2 AN17 CMP1_O PWM0 SPI_SCK	I/O AN O 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 17 输入口 CMP1_O 输出口 PWM0 输出口 SPI 时钟口 SCK
19	PORTC1 AN16 CMP1+ OSCO PWM3	I/O AN AN 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 16 输入口 CMP1 正端输入 晶振输出口 PWM3 输出口
20	PORTC0 AN15 PWM4 SPI_SCK OSCI TX	I/O AN O O I 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 15 输入口 PWM4 输出口 SPI 时钟口 SCK 晶振输入口 UART 发送口 TX
21	PORTB6 AN14	I/O AN	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 14 输入口

	CMP2_2- PWM3	AN O	CMP2 负端输入口 2 PWM3 输出口
22	PORTB5 AN13 PWM2 RX	I/O 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 13 输入口 PWM2 输出口 UART 接收口 RX
23	PORTB4 AN12 PWM4 SPI_SS	I AN O O	输入口，带可编程上下拉电阻，端口电平变化中断 ADC 通道 12 输入口 PWM4 输出口 SPI 片选信号 SS
24	GND	P	地
25	GND	P	地

1.6.2 QFN20 引脚描述

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

21PIN 脚位	名称	类型	说明
1	PORTC6 AN21 CMP2+ IIC_SDA	I/O AN AN I/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 21 输入口 CMP2 正端输入端口 IIC 数据口 SDA
2	PORTC5 AN20 CMP1_2- PWM3 IIC_SCL	I/O AN AN O I/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 20 输入口 CMP1 负端输入口 2 PWM3 输出口 IIC 时钟口 SCL
3	PORTC4 AN19 OPOUT PWM2	I/O AN AN 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 19 输入口 OP 输出口 PWM2 输出口
4	PORTC3 AN18 CMP2_O PWM1	I/O AN 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 18 输入口 CMP2_O 输出口 PWM1 输出口
5	PORTC1 AN16 CMP1+ OSCO PWM3	I/O AN AN 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 16 输入口 CMP1 正端输入 晶振输出口 PWM3 输出口
6	PORTC0 AN15 PWM4 SPI_SCK OSCI TX	I/O AN O O I 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 15 输入口 PWM4 输出口 SPI 时钟口 SCK 晶振输入口 UART 发送口 TX
7	PORTB5 AN13 PWM2 RX	I/O 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 13 输入口 PWM2 输出口 UART 接收口 RX
8	GND	P	地
9	PORTC2 AN17 CMP1_O PWM0 SPI_SCK	I/O AN O 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 17 输入口 CMP1_O 输出口 PWM0 输出口 SPI 时钟口 SCK
10	VDD	P	电源
11	PORTA1 AN1 PWM1 PGD TX	I/O AN O 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 1 输入口 PWM1 输出口 编程数据输入/输出口 PGD UART 发送口 TX
12	PORTA0 AN0 PWM0	I/O AN 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 0 输入口

	PGC RX	I/O I	PWM0 输出口 编程时钟输入口 PGC UART 接收口 RX
13	PORTA2 AN2 PWM2 PCK VREF SPI_MOSI	I/O AN O O AN I/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 2 输入口 PWM2 输出口 编程内部 RC 输出口 PCK ADC 内部参考电压校准口 SPI 通信口 MOSI
14	PORTB2 AN10 T1CKI PWM1 SPI_SS	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 10 输入口 T1 定时器外部时钟输入口 PWM1 输出口 SPI 片选信号 SS
15	PORTB3 AN11 CMP1_1- SPI_MOSI	I/O AN AN I/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 CMP1 负端输入口 1 SPI 数据口 MOSI
16	PORTB1 AN9 INT1 PWM0 SPI_MISO	I/O AN I O I/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 9 输入口 外部中断口 1 PWM0 输出口 SPI 数据口 MISO
17	PORTB0 AN8 INT0 CMP2_4- PWM4	I/O AN I AN 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 8 输入口 外部中断口 0 CMP2 负端输入口 4 PWM4 输出口
18	PORTA6 AN6 PWM1 IIC_SCL	I/O AN 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 6 输入口 PWM1 输出口 IIC 时钟口 SCL
19	PORTA4 AN4 CMP1_4- PWM2	I/O AN AN 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 4 输入口 CMP1 负端输入口 4 PWM2 输出口
20	PORTA3 AN3 CMP1_3- PWM3 T0CKI	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 3 输入口 CMP1 负端输入口 3 PWM3 输出口 T0 定时器外部时钟输入口
21	GND	P	地

1.6.3 QFN16 引脚描述

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

17PIN 脚位	名称	类型	说明
1	VDD	P	电源
2	PORTA1 AN1 PWM1 PGD TX	I/O AN O 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 1 输入口 PWM1 输出口 编程数据输入/输出口 PGD UART 发送口 TX
3	PORTA0 AN0 PWM0 PGC RX	I/O AN O I/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 0 输入口 PWM0 输出口 编程时钟输入口 PGC UART 接收口 RX
4	PORTC6 AN21 CMP2+ IIC_SDA	I/O AN AN I/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 21 输入口 CMP2 正端输入端口 IIC 数据口 SDA
5	PORTC0 AN15 PWM4 SPI_SCK OSCI TX	I/O AN O O I 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 15 输入口 PWM4 输出口 SPI 时钟口 SCK 晶振输入口 UART 发送口 TX
6	PORTC1 AN16 CMP1+ OSCO PWM3	I/O AN AN 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 16 输入口 CMP1 正端输入 晶振输出口 PWM3 输出口
7	PORTC2 AN17 CMP1_O PWM0 SPI_SCK	I/O AN O 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 17 输入口 CMP1_O 输出口 PWM0 输出口 SPI 时钟口 SCK
8	PORTC3 AN18 CMP2_O PWM1	I/O AN 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 18 输入口 CMP2_O 输出口 PWM1 输出口
9	PORTC4 AN19 OPOUT PWM2	I/O AN AN 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 19 输入口 OP 输出口 PWM2 输出口
10	PORTC5 AN20 CMP1_2- PWM3 IIC_SCL	I/O AN AN O I/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 20 输入口 CMP1 负端输入口 2 PWM3 输出口 IIC 时钟口 SCL
11	PORTB1 AN9	I/O AN	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 9 输入口

	INT1 PWM0 SPI_MISO	I O I/O	外部中断口 1 PWM0 输出口 SPI 数据口 MISO
12	PORTB2 AN10 T1CKI PWM1 SPI_SS	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 10 输入口 T1 定时器外部时钟输入口 PWM1 输出口 SPI 片选信号 SS
13	PORTB0 AN8 INT0 CMP2_4- PWM4	I/O AN I AN 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 8 输入口 外部中断口 0 CMP2 负端输入口 4 PWM4 输出口
14	PORTA3 AN3 CMP1_3- PWM3 T0CKI	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 3 输入口 CMP1 负端输入口 3 PWM3 输出口 T0 定时器外部时钟输入口
15	PORTA2 AN2 PWM2 PCK VREF SPI_MOSI	I/O AN O O AN I/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 2 输入口 PWM2 输出口 编程内部 RC 输出口 PCK ADC 内部参考电压校准口 SPI 通信口 MOSI
16	GND	P	地
17	GND	P	地

1.6.4 SOP16 引脚描述

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

16PIN 脚位	名称	类型	说明
1	VDD	P	电源
2	PORTC6 AN21 CMP2+ IIC_SDA	I/O AN AN I/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 21 输入口 CMP2 正端输入端口 IIC 数据口 SDA
3	PORTC1 AN16 CMP1+ OSCO PWM3	I/O AN AN 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 16 输入口 CMP1 正端输入 晶振输出口 PWM3 输出口
4	PORTC0 AN15 PWM4 SPI_SCK OSCI TX	I/O AN O O I 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 15 输入口 PWM4 输出口 SPI 时钟口 SCK 晶振输入口 UART 发送口 TX
5	PORTC2 AN17 CMP1_O PWM0 SPI_SCK	I/O AN O 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 17 输入口 CMP1_O 输出口 PWM0 输出口 SPI 时钟口 SCK
6	PORTC3 AN18 CMP2_O PWM1	I/O AN 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 18 输入口 CMP2_O 输出口 PWM1 输出口
7	PORTC4 AN19 OPOUT PWM2	I/O AN AN 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 19 输入口 OP 输出口 PWM2 输出口
8	PORTC5 AN20 CMP1_2- PWM3 IIC_SCL	I/O AN AN O I/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 20 输入口 CMP1 负端输入口 2 PWM3 输出口 IIC 时钟口 SCL
9	PORTB2 AN10 T1CKI PWM1 SPI_SS	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 10 输入口 T1 定时器外部时钟输入口 PWM1 输出口 SPI 片选信号 SS
10	PORTB1 AN9 INT1 PWM0 SPI_MISO	I/O AN I O I/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 9 输入口 外部中断口 1 PWM0 输出口 SPI 数据口 MISO
11	PORTB0 AN8	I/O AN	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 8 输入口

	INT0 CMP2_4- PWM4	I AN O	外部中断口 0 CMP2 负端输入口 4 PWM4 输出口
12	PORTA3 AN3 CMP1_3- PWM3 T0CKI	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 3 输入口 CMP1 负端输入口 3 PWM3 输出口 T0 定时器外部时钟输入口
13	PORTA2 AN2 PWM2 PCK VREF SPI_MOSI	I/O AN O O AN I/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 2 输入口 PWM2 输出口 编程内部 RC 输出口 PCK ADC 内部参考电压校准口 SPI 通信口 MOSI
14	PORTA0 AN0 PWM0 PGC RX	I/O AN O I/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 0 输入口 PWM0 输出口 编程时钟输入口 PGC UART 接收口 RX
15	PORTA1 AN1 PWM1 PGD TX	I/O AN O 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 1 输入口 PWM1 输出口 编程数据输入/输出口 PGD UART 发送口 TX
16	GND	P	地

1.6.5 TSSOP20 引脚描述

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

20PIN 脚位	名称	类型	说明
1	PORTB0 AN8 INT0 CMP2_4- PWM4	I/O AN I AN 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 8 输入口 外部中断口 0 CMP2 负端输入口 4 PWM4 输出口
2	PORTA6 AN6 PWM1 IIC_SCL	I/O AN 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 6 输入口 PWM1 输出口 IIC 时钟口 SCL
3	PORTA4 AN4 CMP1_4- PWM2	I/O AN AN 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 4 输入口 CMP1 负端输入口 4 PWM2 输出口
4	PORTA3 AN3 CMP1_3- PWM3 T0CKI	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 3 输入口 CMP1 负端输入口 3 PWM3 输出口 T0 定时器外部时钟输入口
5	PORTA2 AN2 PWM2 PCK VREF SPI_MOSI	I/O AN O O AN I/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 2 输入口 PWM2 输出口 编程内部 RC 输出口 PCK ADC 内部参考电压校准口 SPI 通信口 MOSI
6	PORTA0 AN0 PWM0 PGC RX	I/O AN O I/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 0 输入口 PWM0 输出口 编程时钟输入口 PGC UART 接收口 RX
7	GND	P	地
8	PORTA1 AN1 PWM1 PGD TX	I/O AN O 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 1 输入口 PWM1 输出口 编程数据输入/输出口 PGD UART 发送口 TX
9	VDD	P	电源
10	PORTC6 AN21 CMP2+ IIC_SDA	I/O AN AN I/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 21 输入口 CMP2 正端输入端口 IIC 数据口 SDA
11	PORTC4 AN19 OPOUT PWM2	I/O AN AN 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 19 输入口 OP 输出口 PWM2 输出口
12	PORTC3 AN18	I/O AN	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 18 输入口

	CMP2_O PWM1	O O	CMP2_O 输出口 PWM1 输出口
13	PORTB5 AN13 PWM2 RX	I/O 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 13 输入口 PWM2 输出口 UART 接收口 RX
14	PORTC0 AN15 PWM4 SPI_SCK OSCI TX	I/O AN O O I 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 15 输入口 PWM4 输出口 SPI 时钟口 SCK 晶振输入口 UART 发送口 TX
15	PORTC1 AN16 CMP1+ OSCO PWM3	I/O AN AN 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 16 输入口 CMP1 正端输入 晶振输出口 PWM3 输出口
16	PORTC2 AN17 CMP1_O PWM0 SPI_SCK	I/O AN O 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 17 输入口 CMP1_O 输出口 PWM0 输出口 SPI 时钟口 SCK
17	PORTC5 AN20 CMP1_2- PWM3 IIC_SCL	I/O AN AN 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 20 输入口 CMP1 负端输入口 2 PWM3 输出口 IIC 时钟口 SCL
18	PORTB1 AN9 INT1 PWM0 SPI_MISO	I/O AN I O I/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 9 输入口 外部中断口 1 PWM0 输出口 SPI 数据口 MISO
19	PORTB2 AN10 T1CKI PWM1 SPI_SS	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 10 输入口 T1 定时器外部时钟输入口 PWM1 输出口 SPI 片选信号 SS
20	PORTB3 AN11 CMP1_1- SPI_MOSI	I/O AN AN I/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 CMP1 负端输入口 1 SPI 数据口 MOSI

1.6.6 SSOP24 引脚描述

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

24PIN 脚位	名称	类型	说明
1	PORTB1 AN9 INT1 PWM0 SPI_MISO	I/O AN I O I/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 9 输入口 外部中断口 1 PWM0 输出口 SPI 数据口 MISO
2	PORTB3 AN11 CMP1_1- SPI_MOSI	I/O AN AN I/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 CMP1 负端输入口 1 SPI 数据口 MOSI
3	PORTB2 AN10 T1CKI PWM1 SPI_SS	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 10 输入口 T1 定时器外部时钟输入口 PWM1 输出口 SPI 片选信号 SS
4	PORTB0 AN8 INT0 CMP2_4- PWM4	I/O AN I AN 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 8 输入口 外部中断口 0 CMP2 负端输入口 4 PWM4 输出口
5	PORTA6 AN6 PWM1 IIC_SCL	I/O AN 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 6 输入口 PWM1 输出口 IIC 时钟口 SCL
6	PORTA5 AN5 CMP2_3- PWM0 IIC_SDA	I/O AN AN O I/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 5 输入口 CMP2 负端输入端口 3 PWM0 输出口 IIC 数据口 SDA
7	PORTA4 AN4 CMP1_4- PWM2	I/O AN AN 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 4 输入口 CMP1 负端输入口 4 PWM2 输出口
8	PORTA3 AN3 CMP1_3- PWM3 T0CKI	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 3 输入口 CMP1 负端输入口 3 PWM3 输出口 T0 定时器外部时钟输入口
9	GND	P	地
10	VDD	P	电源
11	PORTC7 AN22 CMP2_1- PWM4 SPI_MISO	I/O AN AN O I/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 22 输入口 CMP2 负端输入端口 1 PWM4 输出口 SPI 数据口 MISO
12	PORTC6 AN21	I/O AN	输入/输出口，带可编程上下拉电阻，端口电平变化中断

	CMP2+ IIC_SDA	AN I/O	ADC 通道 21 输入口 CMP2 正端输入端口 IIC 数据口 SDA
13	PORTC5 AN20 CMP1_2- PWM3 IIC_SCL	I/O AN AN 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 20 输入口 CMP1 负端输入口 2 PWM3 输出口 IIC 时钟口 SCL
14	PORTC4 AN19 OPOUT PWM2	I/O AN AN 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 19 输入口 OP 输出口 PWM2 输出口
15	PORTC3 AN18 CMP2_O PWM1	I/O AN 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 18 输入口 CMP2_O 输出口 PWM1 输出口
16	PORTC2 AN17 CMP1_O PWM0 SPI_SCK	I/O AN O 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 17 输入口 CMP1_O 输出口 PWM0 输出口 SPI 时钟口 SCK
17	PORTC1 AN16 CMP1+ OSCO PWM3	I/O AN AN 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 16 输入口 CMP1 正端输入 晶振输出口 PWM3 输出口
18	PORTC0 AN15 PWM4 SPI_SCK OSCI TX	I/O AN O O I 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 15 输入口 PWM4 输出口 SPI 时钟口 SCK 晶振输入口 UART 发送口 TX
19	PORTB6 AN14 CMP2_2- PWM3	I/O AN AN 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 14 输入口 CMP2 负端输入口 2 PWM3 输出口
20	PORTB5 AN13 PWM2 RX	I/O 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 13 输入口 PWM2 输出口 UART 接收口 RX
21	PORTB4 AN12 PWM4 SPI_SS	I AN O O	输入口，带可编程上下拉电阻，端口电平变化中断 ADC 通道 12 输入口 PWM4 输出口 SPI 片选信号 SS
22	PORTA1 AN1 PWM1 PGD TX	I/O AN O 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 1 输入口 PWM1 输出口 编程数据输入/输出口 PGD UART 发送口 TX

23	PORTA0 AN0 PWM0 PGC RX	I/O AN O I/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 0 输入口 PWM0 输出口 编程时钟输入口 PGC UART 接收口 RX
24	PORTA2 AN2 PWM2 PCK VREF SPI_MOSI	I/O AN O O AN I/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 2 输入口 PWM2 输出口 编程内部 RC 输出口 PCK ADC 内部参考电压校准口 SPI 通信口 MOSI

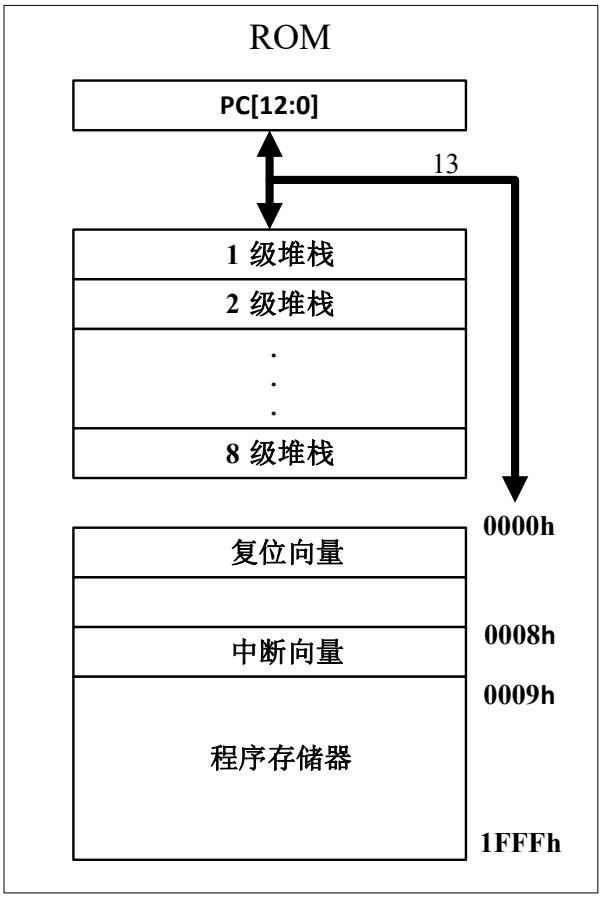
2 中央处理器（CPU）

2.1 存储器

2.1.1 程序存储器（MTP ROM）

BL08M003 具有 8K×16 位的程序存储器，下图给出了程序存储器的映射。访问超出物理地址以外的单元时，会导致返回到地址最低单元。

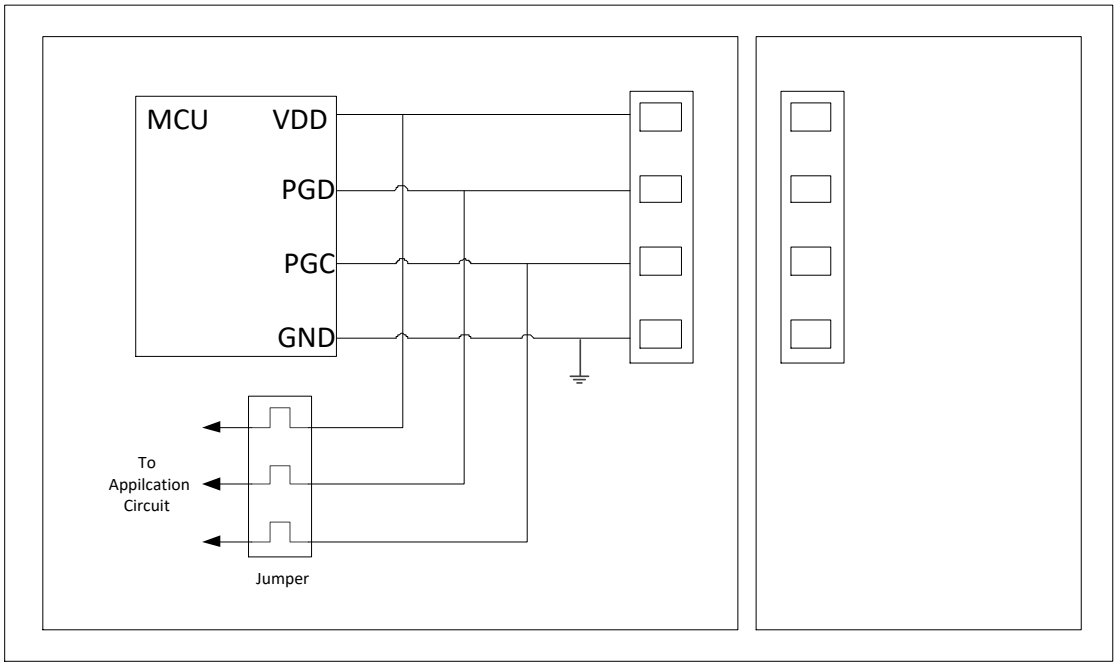
复位向量是 0000H，中断向量是 0008H。



2.1.1.1 MTP ICP 操作

用户可以通过 BL-LINK V5 仿真器的双线方式对 MCU 进行仿真和编程，当 MCU 已经焊在用户板上后，如果用户采用上电复位的方式，只需要连接四根线（VDD、GND、PGC、PGD），用户系统必须断电，由仿真器提供电源。

另外，因为编程信号非常敏感，用户需要用 3 个跳线将编程引脚（VDD、PGC、PGD）从应用电路中分离出来，如下图所示。



注意事项：

仿真时，通讯需要用到PGD、PGC端口，其中PGD、PGC两个端口不能当成普通IO口使用。仿真模式下，TIMER、PWM模块功能的仿真，推荐使用全速指令进行仿真调试。

2.1.1.2 复位向量（0000H）

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

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RSTFR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:
                BTFSC       RSTFR,POR
                GOTO        ISPOR           ;POR 标志为1，判定为上电复位
                BTFSC       RSTFR,BOR
                GOTO        ISBOR           ;POR=0,BOR=1,判定为低电压复位
                BTFSS       STATUS,TO
                GOTO        ISWDTR          ;POR=0,BOR=0,TO=0，判定为WDT复位
EXT_RST:        ...
                ...
ISPOR:          BCF         RSTFR,POR       ;上电复位处理程序
                ...
ISBOR:          BCF         RSTFR,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
```

2.1.1.3 中断向量（0008H）

中断向量地址为0008H。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0008H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         0008H
                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 0x08，响应中断后，程序自动跳转到 0x08 开始执行
2. 中断服务程序需首先对相应的寄存器进行保护。
3. 中断服务子程序返回前对保护的寄存器进行恢复，注意恢复顺序，对 W 必须使用 SWAPF。
4. 程序中使能两个以上的中断源时，程序需对发生中断的中断源进行判断，从而执行相应的服务程序。
5. 需要软件清空对应的中断标志。
6. RETFIE 指令将自动使能 GIE，请勿在中断服务子程序中用其它指令使能 GIE，以免造成中断响应混乱。

2.1.1.4 查表

方式一：

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

- **例：数据查表**

```
        ...
        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~Bit12），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
```

2.1.1.5 读ROM使能寄存器PMCON

B8h	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）

2.1.1.6 读ROM数据寄存器PMDATL、PMDATH

B6h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMDATL	PMDATL[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

Bit[7:0] PMDATL[7:0]:读ROM 数据寄存器低 8 位

B7h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMDATH	PMDATH[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit[7:0] PMDATH[15:8]:读ROM 数据寄存器高 8 位

2.1.1.7 读ROM地址寄存器PMADRL、PMADRH

B4h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMADRL	PMADRL [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

Bit[7:0] PMADRL [7:0]:读ROM 地址寄存器低 8 位

B5h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMADRH	-	-	-	PMADRH [12:8]				
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W
POR 的值	-	-	-	0	0	0	0	0

Bit[4:0] PMADRH [12:8]:读ROM 地址寄存器高 5 位

2.1.2 掉电数据存储器（EEPROM）

2.1.2.1 EEPROM特性

- 内置 256*16 位独立 EEPROM 区
- EEPROM 操作时 CPU 时钟停止

2.1.2.2 EEPROM相关寄存器

2.1.2.2.1 EEPROM 控制寄存器 EEPCON

EEPCON 寄存器

A0h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPCON	-	-	-	-	-	-	WR	RD
R/W	-	-	-	-	-	-	R/W	R/W
POR 的值	-	-	-	-	-	-	0	0

Bit[1] WR：EEPROM 写控制位

1 = 启动存储器写操作(由硬件清零 WR，用软件只能将 WR 置 1，但不能清零)

0 = 写周期完成

Bit[0] RD：EEPROM 读控制位

1 = 启动存储器读操作(由硬件清零 RD，用软件只能将 RD 置 1，但不能清零)

0 = 不启动存储器读操作

注：

程序中，执行 WR 与 RD 使能后，需至少增加两句 NOP 指令，以防时序错误

2.1.2.2.2 EEPROM 数据寄存器 EEPDAT

EEPDATH 寄存器

A1h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPDATH	EEPDATH[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit[7:0] EEPDATH[15:8]：EEPROM 数据寄存器高 8 位

EEPDATL 寄存器

A2h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPDATL	EEPDATL[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

Bit[7:0] EEPDATL[7:0]：EEPROM 数据寄存器低 8 位

2.1.2.2.3 EEPROM 页地址寄存器 EEPAGADR

EEPAGADR 寄存器

A3h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPAGADR	-	-	-	-	EEPAGADR[3:0]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit[3:0] EEPAGADR[3:0]：EEPROM 页地址寄存器

2.1.2.2.4 EEPROM 页内地址寄存器 EEONPADR

EEONPADR 寄存器

A4h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEONPADR	-	-	-	-	EEONPADR[3:0]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit[3:0] EEONPADR[3:0]：EEPROM 页内地址寄存器

2.1.2.2.5 EEPROM 擦除时间控制寄存器 EEPERAT

EEPERAT 寄存器

97h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPERAT	EEPERAT[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

Bit[7:0] EEPERAT[7:0]：EEPROM 擦除时间控制寄存器

2.1.2.2.6 EEPROM 编程时间寄存器 EEPPROT

EEPPROT 寄存器

98h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EEPPROT	EEPPROT[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

Bit[7:0] EEPPROT[7:0]：EEPROM 编程时间控制寄存器

注：

1、擦除时间和编程时间一个值为 32us，请使用 EEPROM 前，将 EEPERAT 寄存器固定为 0x4E，将 EEPPROT 寄存器固定为 0x2E

2、EEPROM 写之前会自动擦除操作，不需要软件进行控制

2.1.2.3 EEPROM读操作

1、写需要读取的地址
2、使能读操作
3、至少增加两句 NOP 指令
4、读完成
5、读数据寄存器

2.1.2.4 EEPROM写操作

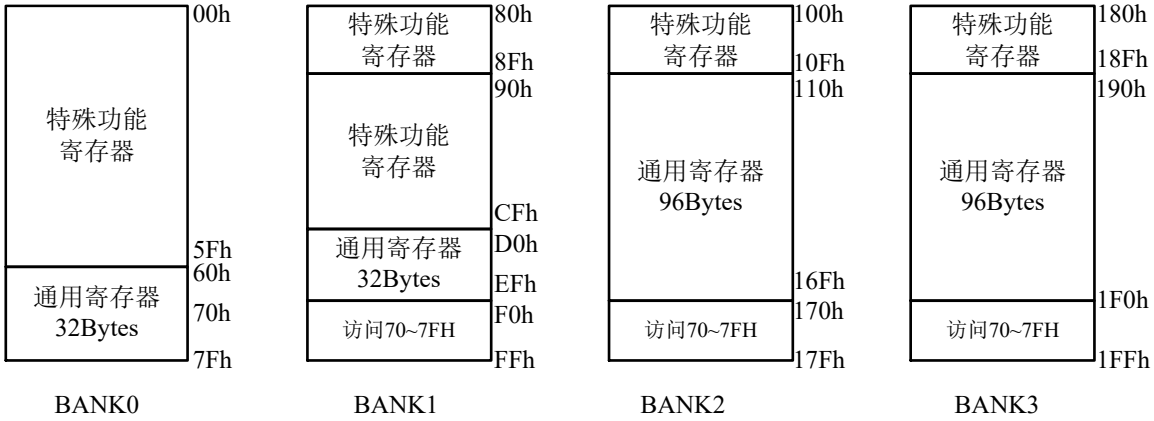
1、写数据寄存器
2、写地址
3、使能写操作
4、至少增加两句 NOP 指令
5、写完成

2.1.2.5 EEPROM写多字节操作

1、写数据寄存器
2、写地址
3、使能写操作
4、至少增加两句 NOP 指令
5、写完成
6、地址加 1
7、重复步骤 1-5，循环操作

2.1.3 通用数据存储器（RAM）

共有 256 个通用寄存器（GPR），分在 Bank0、Bank1、Bank2、Bank3 存储区。



数据存储器映射:

00h&80h&100h&180h	INDF	间接寻址寄存器（不是实际存在的物理寄存器）							
02h&82h&102h&182h	PCL	程序计数器（PC）低字节							
03h&83h&103h&183h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
04h&84h&104h&184h	FSR	间接寻址地址指针							
08h&88h&108h&188h	PCLATH	-	-	-	程序计数器高 5 位缓存器				
09h&89h&109h&189h	INTCON	GIE	-	INT0F	PBIF	T0IF	INT0E	PBIE	T0IE

2.1.4 特殊功能寄存器（SFR）

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										
10h	TRISA	-	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	_000 0000
11h	TRISB	-	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	_000 0000
12h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	0000 0000
13h	PORTA	-	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	_xxx xxxx
14h	PORTB	-	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	_xxx xxxx
15h	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	xxxx xxxx
16h	WPUA	-	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	_000 0000
17h	WPUB	-	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	_000 0000
18h	WPUC	WPUC7	WPUC6	WPUC5	WPUC4	WPUC3	WPUC2	WPUC1	WPUC0	0000 0000
19h	WPDA	-	WPDA6	WPDA5	WPDA4	WPDA3	WPDA2	WPDA1	WPDA0	_000 0000
1Ah	WPDB	-	WPDB6	WPDB5	WPDB4	WPDB3	WPDB2	WPDB1	WPDB0	_000 0000
1Bh	WPDC	WPDC7	WPDC6	WPDC5	WPDC4	WPDC3	WPDC2	WPDC1	WPDC0	0000 0000
1Ch	IOCA	-	IOCA6	IOCA5	IOCA4	IOCA3	IOCA2	IOCA1	IOCA0	_000 0000
1Dh	IOCB	-	IOCB6	IOCB5	IOCB4	IOCB3	IOCB2	IOCB1	IOCB0	_000 0000
1Eh	IOCC	IOCC7	IOCC6	IOCC5	IOCC4	IOCC3	IOCC2	IOCC1	IOCC0	0000 0000
22h	ANSELA	-	ANSELA6	ANSELA5	ANSELA4	ANSELA3	ANSELA2	ANSELA1	ANSELA0	_000 0000
23h	ANSELB	-	ANSELB6	ANSELB5	ANSELB4	ANSELB3	ANSELB2	ANSELB1	ANSELB0	_000 0000
24h	ANSELC	ANSELC7	ANSELC6	ANSELC5	ANSELC4	ANSELC3	ANSELC2	ANSELC1	ANSELC0	0000 0000

26h	PWM_MAP	PWM3_MAP[1:0]		PWM2_MAP[1:0]		PWM1_MAP[1:0]		PWM0_MAP[1:0]		0000 0000
27h	PWM4_MAP	-	-	-	-	-	-	PWM4_MAP[1:0]		____ __00
28h	UART_MAP	-	-	RX_MAP[1:0]		-	-	TX_MAP[1:0]		__00 __00
29h	SPI_MAP	SS_MAP[1:0]		SCK_MAP[1:0]		MISO_MAP[1:0]		MOSI_MAP[1:0]		0000 0000
2Ah	IIC_MAP	-	-	SCL_MAP[1:0]		-	-	SDA_MAP[1:0]		__00 __00
2Bh	PIR	T1IE	T1IF	INT1E	INT1F	ADIE	ADIF	PAIE	PAIF	0000 0000
2Ch	PIR1	CMP2IE	CMP2IF	CMP1IE	CMP1IF	PWM01IE	PWM01IF	PCIE	PCIF	0000 0000
2Dh	PIR2	RXIE	RXIF	TXIE	TXIF	PWM4IE	PWM4IF	PWM23IE	PWM23IF	0000 0000
2Eh	PIR3	-	SPIE	SPIF	MODF	T2IE	T2IF	IICIE	IICIF	_000 0000
2Fh	T0	Timer0 计数寄存器								xxxx xxxx
30h	T0CON	-	-	-	-	-	T0CS	T0SE	T0SOCEN	____ _000
31h	T1CON	T1CS1	T1CS0	T1OSCEN	T1SYNC	-	-	-	T1ON	0000 ___0
32h	T1L	Timer1 计数寄存器低八位								1111 1111
33h	T1H	Timer1 计数寄存器高八位								1111 1111
34h	PR1CON	-	-	-	-	T1CKPS3	T1CKPS2	T1CKPS1	T1CKPS0	____ 0000
35h	T2CON	SCS	-	-	T2CKPS3	T2CKPS2	T2CKPS1	T2CKPS0	T2ON	0__0 0000
36h	T2	Timer2 计数寄存器								0000 0000
37h	PR2	Timer2 周期寄存器								1111 1111
38h	OPTION	WDTEN	VBGEN	-	-	PSA	PS2	PS1	PS0	01__ 0000
39h	ADRESL	ADC 结果寄存器低字节								0000 ____
3Ah	ADRESH	ADC 结果寄存器高字节								0000 0000
3Bh	ADCON0	-	CHS[4:0]					ADON	ADEN	_000 0000
3Ch	ADCON1	-	ADCS[2:0]			-	VHS[1:0]		-	_000 _00_
3Dh	ADCJMD	-	-	ADJTRIM[5:0]						__00 0000
3Eh	CMP1CON0	CMP1EN	CMP1PS	CMP1NS[2:0]			CMP1NV	CMP1OUT	CMP1OEN	0000 0000
3Fh	CMP1CON1	CMP1IM	OPAPS	RBIAS1_H	RBIAS1_L	LVDS1[3:0]				0000 0000
40h	CMP2CON0	OPCMP2EN	CMP2IM	-	CMP2NS[1:0]		CMP2NV	CMP2OUT	CMP2OEN	00_0 0000
42h	CMP1DBC	CMP1DBCEN	-	-	-	CMP1DBC[3:0]				0___ 0000
43h	CMP2DBC	CMP2DBCEN	-	-	-	CMP2DBC[3:0]				0___ 0000
44h	CMPHYC	-	-	-	-	-	-	-	CMP1HYS	____ ___0
45h	OPA0C	-	OPASW2	OPCMPSW	OPOUTEN	-	OPASW1	OPAG[1:0]		_000 _000
46h	OPAVOS	OPAOFM	OPARSP	OPFMOUT	OPAOF[4:0]					0000 0000
47h	PWM01CON0	-	CK01[2:0]			TY01	PWM1OEN	PWM0OEN	PWM01EN	_000 0000
48h	PWM01CON1	-	-	-	-	PWM1DIR	PWM0DIR	LED0CEN	LED0EN	____ 0000
49h	PWM01TH	-	-	-	-	PWM01TH[11:8]				____ 0000
4Ah	PWM01TL	PWM01TL[7:0]								0000 0000
4Bh	PWM0DH	-	-	-	-	PWM0DH[11:8]				____ 0000
4Ch	PWM0DL	PWM0DL[7:0]								0000 0000
4Dh	PWM1DH	-	-	-	-	PWM1DH[11:8]				____ 0000
4Eh	PWM1DL	PWM1DL[7:0]								0000 0000
4Fh	LED0DATA	LED0DATA[7:0]								0000 0000

50h	PWM23CON0	-	CK23[2:0]			TY23	PWM3OEN	PWM2OEN	PWM23EN	_000 0000
51h	PWM23CON1	-	-	-	-	PWM3DIR	PWM2DIR	LED1CEN	LED1EN	____ 0000
52h	PWM23TH	-	-	-	-	PWM23TH[11:8]				____ 0000
53h	PWM23TL	PWM23TL[7:0]								0000 0000
54h	PWM2DH	-	-	-	-	PWM2DH[11:8]				____ 0000
55h	PWM2DL	PWM2DL[7:0]								0000 0000
56h	PWM3DH	-	-	-	-	PWM3DH[11:8]				____ 0000
57h	PWM3DL	PWM3DL[7:0]								0000 0000
58h	LED1DATA	LED1DATA[7:0]								0000 0000
59h	PWM4CON	-	CK4[2:0]			-	PWM4DIR	PWM4OEN	PWM4EN	_000 _000
5Ah	PWM4TH	-	-	-	-	PWM4TH[11:8]				____ 0000
5Bh	PWM4TL	PWM4TL[7:0]								0000 0000
5Ch	PWM4DH	-	-	-	-	PWM4DH[H11:8]				____ 0000
5Dh	PWM4DL	PWM4DL[7:0]								0000 0000
BANK1										
90h	LCDCON	RLCD[1:0]		BIAS	FRAME	-	-	-	LCDEN	0000 ___0
91h	SEGPAEN	-	SEGPA6EN	SEGPA5EN	SEGPA4EN	SEGPA3EN	SEGPA2EN	SEGPA1EN	SEGPA0EN	_000 0000
92h	SEGPBEN	-	SEGPB6EN	SEGPB5EN	SEGPB4EN	SEGPB3EN	SEGPB2EN	SEGPB1EN	SEGPB0EN	_000 0000
93h	SEGPCEN	SEGPC7EN	SEGPC6EN	SEGPC5EN	SEGPC4EN	SEGPC3EN	SEGPC2EN	SEGPC1EN	SEGPC0EN	0000 0000
94h	COMPAEN	-	COMPA6EN	COMPA5EN	COMPA4EN	COMPA3EN	COMPA2EN	COMPA1EN	COMPA0EN	_000 0000
95h	COMPBEN	-	COMPB6EN	COMP5EN	COMPB4EN	COMPB3EN	COMPB2EN	COMPB1EN	COMPB0EN	_000 0000
96h	COMPCEN	COMPC7EN	COMPC6EN	COMPC5EN	COMPC4EN	COMPC3EN	COMPC2EN	COMPC1EN	COMPC0EN	0000 0000
97h	EEPERAT	EEPERAT[7:0]								0000 0000
98h	EEPPROT	EEPPROT[7:0]								0000 0000
A0h	EEPCON	-	-	-	-	-	-	WR	RD	____ __00
A1h	EEPDATH	EEPDATH[15:8]								0000 0000
A2h	EEPDATL	EEPDATL[7:0]								0000 0000
A3h	EEPAGADR	-	-	-	-	EEPAGADR[3:0]				____ 0000
A4h	EEONPADR	-	-	-	-	EEONPADR[3:0]				____ 0000
A5h	S1CON	FE	RXROV	TXCOL	REN	TB8	RB8	-	-	0000 00__
A6h	S1CON2	-	-	BRTR	-	-	UARTEN	SM1	SM2	__0_ _000
A7h	SBUF	SBUF[7:0]								0000 0000
A8h	BRTL	BRTL[7:0]								0000 0000
A9h	BRTH	BRTH[7:0]								0000 0000
AAh	SPCTL	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR[1:0]		0000 0000
ABh	SPSTAT	-	WCOL	RXOV	-	-	-	-	-	_00_ ____
ACh	SPDAT	SPDAT[7:0]								0000 0000
ADh	IICCON	CR2	IICEN	STA	STO	-	AA	CR1	CR0	0000 _000
AEh	IICSTA	IICSTA[7:3]					-	-	-	0000 0___
AFh	IICDAT	IICDAT[7:0]								0000 0000
B0h	IICADR	IICADR[7:1]							GC	0000 0000
B1h	DBGCR	-	-	-	-	-	-	-	DBG0EN	____ ___0

B2h	RSTFR	-	-	-	-	-	POR	BOR	BOREN	____ _qq1
B4h	PMADRL	PMADRL [7:0]								0000 0000
B5h	PMADRH	-	-	-	PMADRH[12:8]					___0 0000
B6h	PMDATL	PMDATL[7:0]								0000 0000
B7h	PMDATH	PMDATH[15:8]								0000 0000
B8h	PMCON	-	-	-	-	-	-	-	RDON	____ ___0
B9h	INTEDG	-	-	-	INT1EDG	-	-	-	INT0EDG	___0 ___0

2.1.4.1 寄存器INDF

INDF 不是物理寄存器，对 INDF 寻址实际上是对 FSR 指向的数据存储器地址进行访问，从而实现间接寻址模式。

2.1.4.2 寄存器FSR

间接寻址指针 FSR

04h&84h& 104h&184h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FSR	间接寻址地址指针							
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

2.1.4.3 程序计数器

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

程序计数器高 5 位

08h&88h& 108h&188h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCLATH	-	-	-	程序计数器高 5 位				
R/W	-	-	-	R/W	R/W	R/W	R/W	R/W
POR 的值	-	-	-	0	0	0	0	0

程序计数器低 8 位

02h&82h& 102h&182h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCL	程序计数器低 8 位							
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

程序存储器指针（PC）的操作模式
顺序执行指令： PC+1 → PC
分支指令 GOTO/CALL：INST[12:0]（指令码低 13 位）→ PC
子程序返回指令 RETRUN/RETLW/RETFIE：TOS（堆栈栈顶）→ PC
ADDWF PCL，F
PCLATH[4:0]，ALU[7:0]（ALU 运算结果）→ PC
其它 PCL 作为目的操作数指令
PCLATH[4:0]，ALU[7:0] → PC

2.1.4.4 寄存器STATUS

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。

03h&83h& 103h&183h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W
POR 的值	0	0	0	1	1	x	x	x

Bit [7] IRP：BANK 选择位（用于间接寻址）

1 = 间接寻址 BANK2、3（100H~1FFH）

0 = 间接寻址 BANK0、1（00H~FFH）

Bit [6:5] RP[1:0]：BANK 切换位（用于直接寻址或立即寻址）

11 = 切换到 BANK3

10 = 切换到 BANK2

01 = 切换到 BANK1

00 = 切换到 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

2.1.5 芯片配置选择

BL08M003 配置表

芯片配置	配置选择	说明
BOR 欠压复位	1.8V	复位电压设置为 1.8V
	2.0V	复位电压设置为 2.0V
	2.4V	复位电压设置为 2.4V
	3.0V	复位电压设置为 3.0V
时钟模式	2T	1 个时钟周期由 2 个内部 RC 振荡器时钟组成(固定为 2T)
WDT 溢出时间	TWDT0	TWDT(no Prescaler)=14.4ms
	TWDT1	TWDT(no Prescaler)=3.6ms
	TWDT2	TWDT(no Prescaler)=230.4ms
	TWDT3	TWDT(no Prescaler)=57.6ms
WDTE	屏蔽 WDT	屏蔽芯片内嵌硬件看门狗功能
	使能 WDT	使能芯片内嵌硬件看门狗功能（仍可通过软件屏蔽）
系统启动时钟选择	高频启动	高频时钟启动
	低频启动	低频时钟启动
加密功能使能	不加密	屏蔽代码加密功能
	加密	使能代码加密功能
输入管脚施密特	使能施密特	使能输入端口施密特功能
	屏蔽施密特	屏蔽输入端口施密特功能
IP 低功耗使能	使能	启用 IP 低功耗使能（仅在低频时钟下有效）
	禁止	禁止 IP 低功耗使能
低频时钟选择	内部低频 RC 振荡器：40KHz	
	外部低频晶体振荡器：32.768KHz	
内部高频 RC 频率选择 16M	非 16M，由高频内部 RC 频率决定	
	固定选择 16M	
高频内部 RC 频率	8MHz	内部 RC 振荡器频率为 8MHz
	4MHz	内部 RC 振荡器频率为 4MHz
	2MHz	内部 RC 振荡器频率为 2MHz
	1MHz	内部 RC 振荡器频率为 1MHz
	500KHz	内部 RC 振荡器频率为 500KHz
	250KHz	内部 RC 振荡器频率为 250KHz
	125KHz	内部 RC 振荡器频率为 125KHz
	62.5KHz	内部 RC 振荡器频率为 62.5KHz

注：

Option 中 IP 低功耗使能仅在 CPU 时钟为低频时钟时使用。

2.1.6 在板烧录

芯片支持编程工具对芯片中程序存储器的在板不带电烧录编程，即在未上电的系统电路板上，借助编程工具，通过芯片的串行编程接口将用户程序代码烧录进芯片的程序存储器中。

芯片的在板烧录编程通过引脚 VDD、GND、PGD、PGC 实现，这些编程引脚的外围电路需进行针对性设计，以保证外围电路不会影响在板编程时端口上的电压、电流、时序等特性。

芯片也支持在板带电烧录编程，即可在系统电路板不掉电（芯片已正常工作）的状态下对存储器编程。

注：
1、不支持空芯片的在板带电烧录编程
2、在板带电烧录，编程器 VCC 不接入电路板，芯片由系统电路板通过 VDD 引脚供电
3、在板带电烧录，编程器 GND/PGD/PGC 接入电路板前，需注意连接端口之间电压特性是否匹配，尤其需注意编程器与系统电路板的共 GND 是否会发生浮地与市电地短路的问题
4、在板带电烧录，编程器在烧录完成后，会将芯片重新复位
5、应用中需注意，PA0(PGC)、PA1(PGD)两个 IO 为烧录口，默认下拉电阻为 120K，在烧录过程需要进行通信，且这两个 IO 为数字输入状态。

DEBUG 控制寄存器

B1h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBGCR	-	-	-	-	-	-	-	DBG0EN
R/W	-	-	-	-	-	-	-	R/W
POR 的值	-	-	-	-	-	-	-	0

Bit[0]　DBG0EN：PGC/PGD 端口编程功能使能位
0 = 端口仅在复位时用作编程端口，复位完成后用作通用端口
1 = 使能端口的编程功能，端口的通用功能被屏蔽

注：
应用程序中若不需要带电升级，可不配置此寄存器，默认为通用 IO 口；需要带电升级则需要将 DBG0EN 控制位使能，使能后 PGC/PGD 两个 IO 仅作为编程功能，通用功能屏蔽

2.2 寻址模式

BL08M003 共有三种寻址方式：立即寻址、直接寻址和间接寻址模式。

2.2.1 立即寻址

立即数参与运算的寻址方式。

- 例：立即寻址

```
        ADDLW    06h        ; W 的内容加 6，结果放入 W
```

2.2.2 直接寻址

寄存器参与运算的寻址方式。

- 例：直接寻址

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

2.2.3 间接寻址

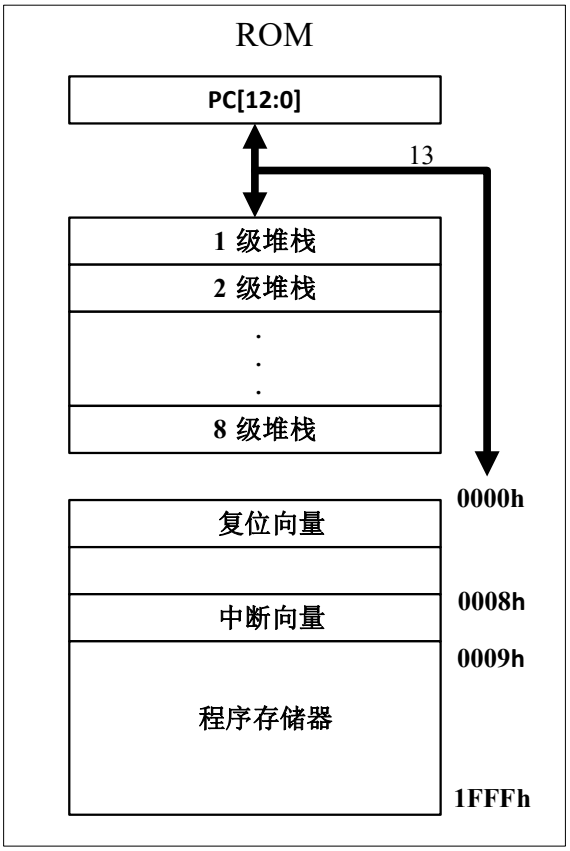
由指针 FSR 指向的寄存器参与运算的寻址方式。INDF 寄存器不是物理寄存器，对 INDF 寄存器操作可以实现间接寻址。

- 例：利用间接寻址对 0XD0~0XEF 通用数据存储器进行清零

```
        BCF     IRP         ;指向 BANK0、1(00H~FFH)
        MOVLW   0XD0
        MOVWF   FSR         ;FSR 指向 A0h 地址
NEXTBYTE:
        CLRF    INDF        ;对 FSR 指向的数据存储器清零
        INCF    FSR,F       ;FSR + 1,指向下一个地址
        MOVLW   0XF0        ;注意这里的边界值为欲操作 RAM 最大地址 + 1
        XORWF   FSR,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 EFh,循环清零下一个地址
CONTINUE:   ……              ;完成清零操作
```

2.3 堆栈

BL08M003 具有一个 8 级深度的硬件堆栈，堆栈指针不能读写。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。

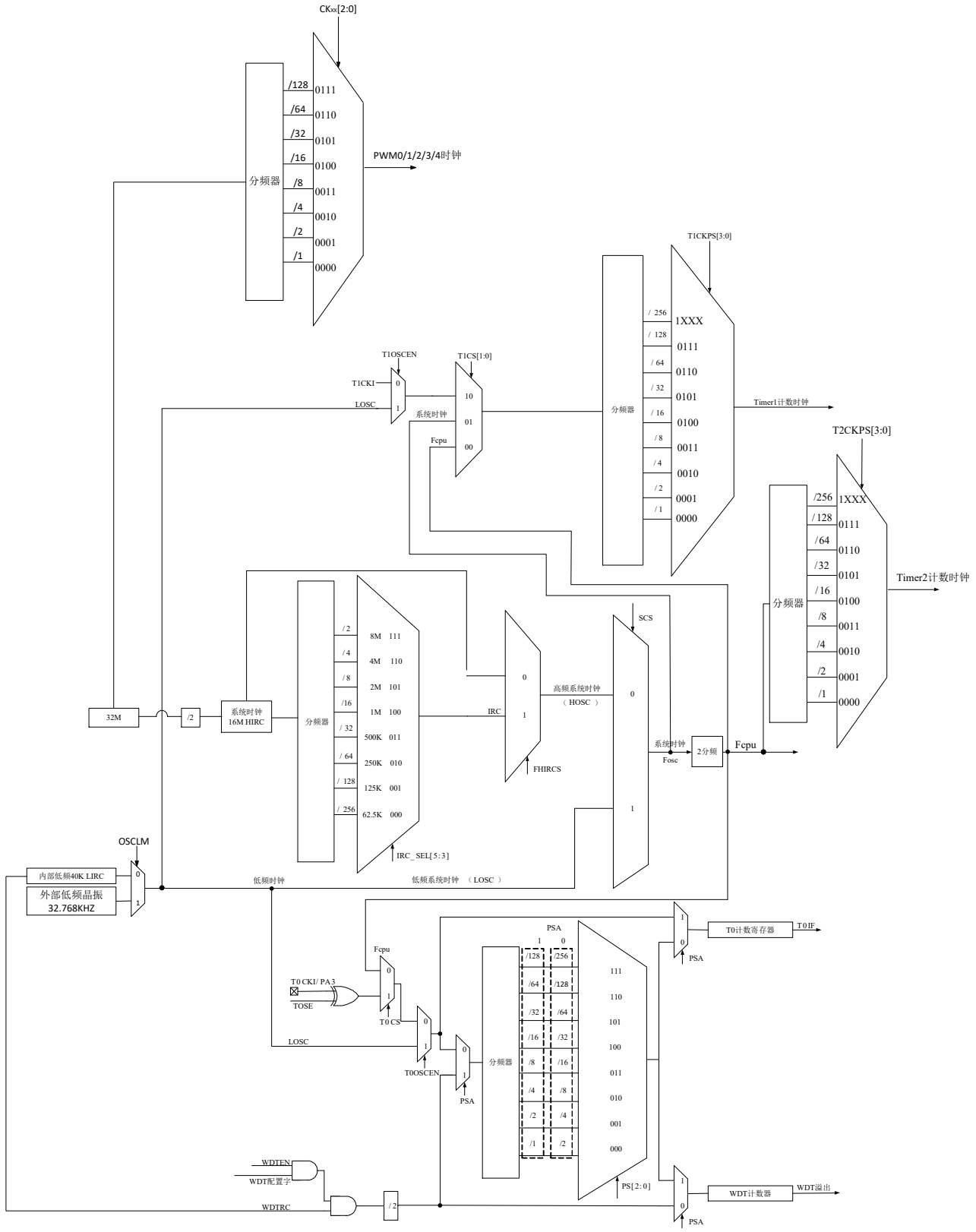


注：

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 系统时钟

3.1 概述


CKω[2:0]
分频器
/128 0111
/64 0110
/32 0101
/16 0100
/8 0011
/4 0010
/2 0001
/1 0000
PWM0/1/2/3/4时钟
T1CKPS[3:0]
T1OSCEN
T1CS[1:0]
T1CKI
LOSC
系统时钟
Fcpu
10
01
00
/ 256 1XXX
/ 128 0111
/ 64 0110
/ 32 0101
/ 16 0100
/ 8 0011
/ 4 0010
/ 2 0001
/ 1 0000
Timer1计数时钟
T2CKPS[3:0]
/256 1XXX
/ 128 0111
/ 64 0110
/ 32 0101
/ 16 0100
/8 0011
/4 0010
/2 0001
/1 0000
Timer2计数时钟
32M
/2
系统时钟
16M HIRC
/ 2 8M 111
/ 4 4M 110
/ 8 2M 101
/16 1M 100
/ 32 500K 011
/ 64 250K 010
/ 128 125K 001
/ 256 62.5K 000
IRC_ SEL[5:3]
IRC
高频系统时钟
（ HOSC ）
SCS
FHIRCS
系统时钟
Fosc
2分频
Fcpu
OSCLM
内部低频40K LIRC
外部低频晶振
32.768KHZ
低频时钟
低频系统时钟 （ LOSC ）
PSA
T0计数寄存器
T0IF
Fcpu
T0 CKI/ PA3
TOSE
T0 CS
LOSC
T0OSCEN
PSA
/128 /256 111
/64 /128 110
/32 /64 101
/16 /32 100
/8 /16 011
/4 /8 010
/2 /4 001
/1 /2 000
PS[2:0]
WDTEN
WDT配置字
WDTRC
/ 2
WDT计数器
WDT溢出
PSA

BL08M003 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内部 32MHz RC 振荡电路（RC 32MHz）2 分频后提供 16MHz 系统时钟，可通过 IRC_SEL[2:0]对 16MHz 时钟进行分频。低频时钟的时钟源则由内部低频 RC 振荡电路（RC 40KHz@5V）或外部低频晶振提供。两种时钟都可作为系统时钟源 Fosc。

高频模式：Fcpu = Fosc / 2

低频模式：Fcpu = Fosc /2

也可选用外部低频晶体振荡器作为系统时钟源，由配置字 OSCLM 控制。

低频晶体振荡器：32.768KHz

35h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T2CON	SCS	-	-	T2CKPS3	T2CKPS2	T2CKPS1	T2CKPS0	T2ON
R/W	R/W	-	-	R/W	R/W	R/W	R/W	R/W
POR 的值	0	-	-	0	0	0	0	0

Bit [7] SCS：高低频模式选择位

1 = 系统时钟切换为低频时钟

0 = 系统时钟切换为高频时钟

3.2 系统高频时钟

系统高频时钟为内部高频 RC

3.2.1 内部高频RC

内置高频 RC 振荡器有 16MHz、8MHz、4MHz、2MHz、1MHz、500KHz、250KHz、125KHz、62.5KHz 九种可选。

高频内部 RC 振荡器频率选择配置字

IRC_SEL[2:0]	说明
111	内部 RC 振荡器频率选择 8MHz
110	内部 RC 振荡器频率选择 4MHz
101	内部 RC 振荡器频率选择 2MHz
100	内部 RC 振荡器频率选择 1MHz
011	内部 RC 振荡器频率选择 500KHz
010	内部 RC 振荡器频率选择 250KHz
001	内部 RC 振荡器频率选择 125KHz
000	内部 RC 振荡器频率选择 62.5KHz

16MHz 时钟单独选择，由配置字的 FHIRCS 决定。

FHIRCS	说明
0	非 16Mhz，由 IRC_SEL[2:0]决定
1	固定选择 16Mhz

3.3 系统低频时钟

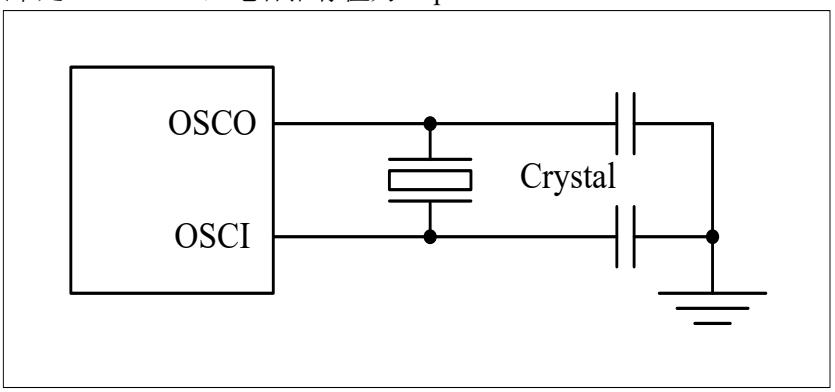
系统低频时钟为内部低频 RC 40KHz

3.3.1 内部低频RC

内部低频 RC 振荡器的频率为 40KHz，或外部低频晶体振荡器 32.768KHz，可以提供给系统使用，内部低频 RC 振荡器可供 WDT 使用。

3.3.2 外部低频晶体振荡器

低频晶体振荡器的频率是 32.768KHz，电容推荐值为 20pF。



系统工作在绿色模式下，可以使能低频晶体振荡器。

4 复位

4.1 概述

BL08M003 共有三种复位方式：

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

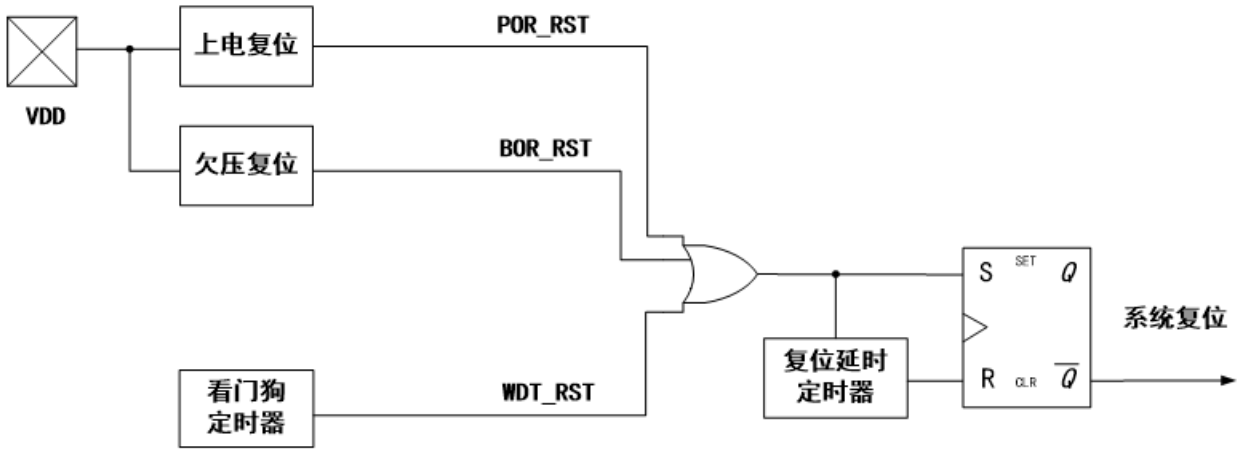
当上述任何一种复位产生时，系统进入复位状态，所有的特殊功能寄存器被初始化，程序停止运行，同时程序计数器（PC）清零。经过上电延时定时器延时后，系统结束复位状态，程序从 0000h 地址开始执行。STATUS 寄存器的 Bit4（TO 位）及 RSTFR 寄存器的 Bit3（BOR 位）、Bit4（POR 位）显示系统复位状态信息，可通过这 3 个标志位判断复位来源，从而控制系统的运行路径。

特殊功能寄存器复位状态：

TO	POR	BOR	复位方式	说明
1	1	x	上电复位	电源上电
u	u	1	欠压复位	电源电压低于 BOR 电压点
1	u	u	看门狗定时器复位	运行模式下，看门狗定时器溢出

注： u = 保持与复位前不变， x = 未知

复位电路示意图：



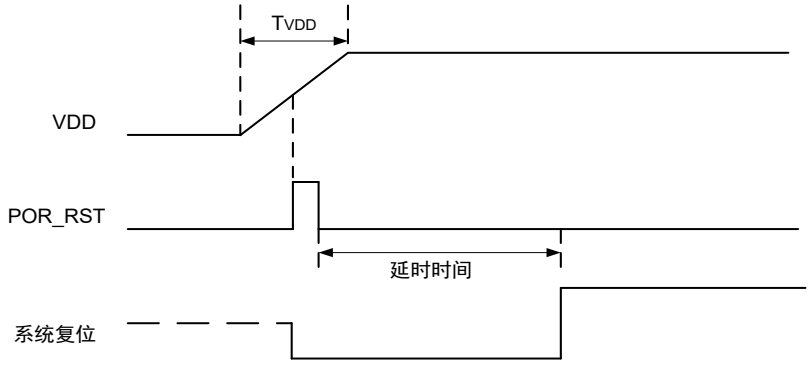
复位延时定时器在复位信号结束后，提供一定时间的延时

复位方式	复位延时定时器时间（典型值）
上电复位	2.5ms
欠压复位	2ms
看门狗定时器复位	OPTION 选择

4.2 上电复位

系统上电过程中，VDD 达到系统正常工作电压之前，上电复位电路产生内部复位信号。可通过查询 STATUS 寄存器的的 Bit4（TO 位）及 RSTFR 寄存器的 Bit3（BOR 位）、Bit4（POR 位）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 的上升速度和起振时间都不固定，内部高频 RC 振荡器的起振时间最短。在用户的使用过程中，应考虑系统对上电复位时间的要求。

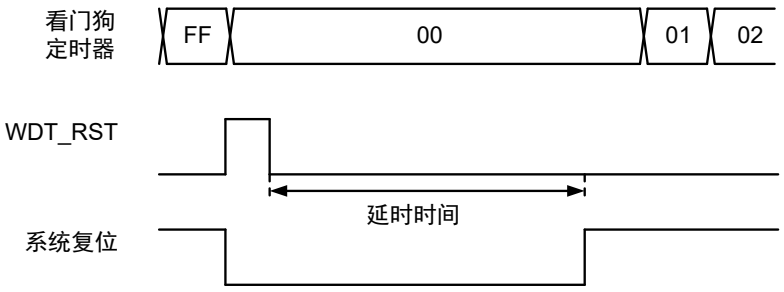
上电复位示意图：



关于上电复位，请注意以下几点：
1. VDD 上电必须从 0V 开始，若 VDD 有残留电压，POR_RST 信号无法稳定产生。
2. VDD 上电斜率必须满足大于 500mV/ms，否则 POR_RST 信号可能无法产生。

4.3 WDT 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 WDTEN 都为 1 时，才能使能看门狗定时器。
看门狗复位示意图：



关于看门狗复位使用时，请注意以下几点：

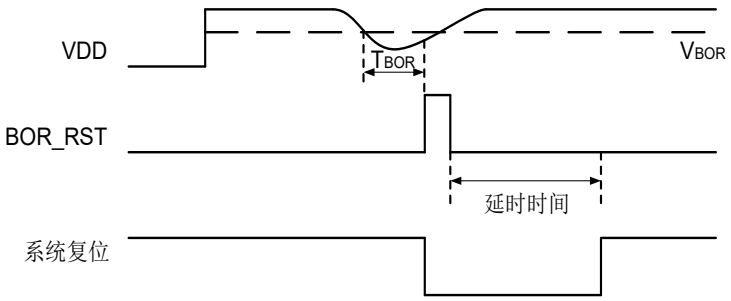
1. 主程序中有一次清看门狗的动作，这种架构能够最大限度的发挥看门狗的保护功能。看门狗的使能逻辑：看门狗使能 = 看门狗配置字使能(WDTE=1)&看门狗软件使能（WDTEN=1）。
2. 不建议在中断程序中对看门狗进行清零，否则无法监控主程序跑飞情况。

4.4 欠压复位

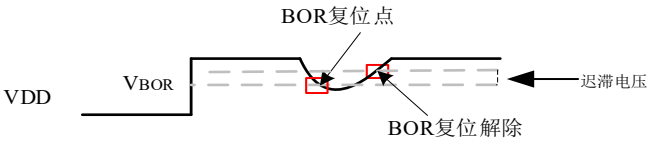
4.4.1 欠压复位的产生

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

欠压复位示意图：



注：T_{BOR} 需大于 200ns，否则电压跌落时可能不产生欠压复位信号。



特别说明：电压检测电路有一定的迟滞特性，迟滞电压为 5%V 左右。即当 VDD 电压下降到所选 BOR 电压档位时 BOR 复位有效，而 VDD 电压需要上升到(BOR 档位电压+档位电压 5%)V 时 BOR 复位才会解除。

4.5 复位相关寄存器

4.5.1 RSTFR寄存器

复位标志&BOR 相关控制寄存器 RSTFR

B2h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RSTFR	-	-	-	-	-	POR	BOR	BOREN
R/W	-	-	-	-	-	R/W	R/W	R/W
POR 的值	-	-	-	-	-	q	q	1

Bit[2]　POR：上电复位状态位

1 = 发生了上电复位（需要软件清 0）

0 = 非上电复位

Bit[1]　BOR：欠压复位状态为

1 = 发生了欠压复位（需要软件清 0）

0 = 非上电复位

Bit[0] BOREN：欠压复位使能

1 = 使能欠压复位

0 = 禁止欠压复位

5 系统工作模式

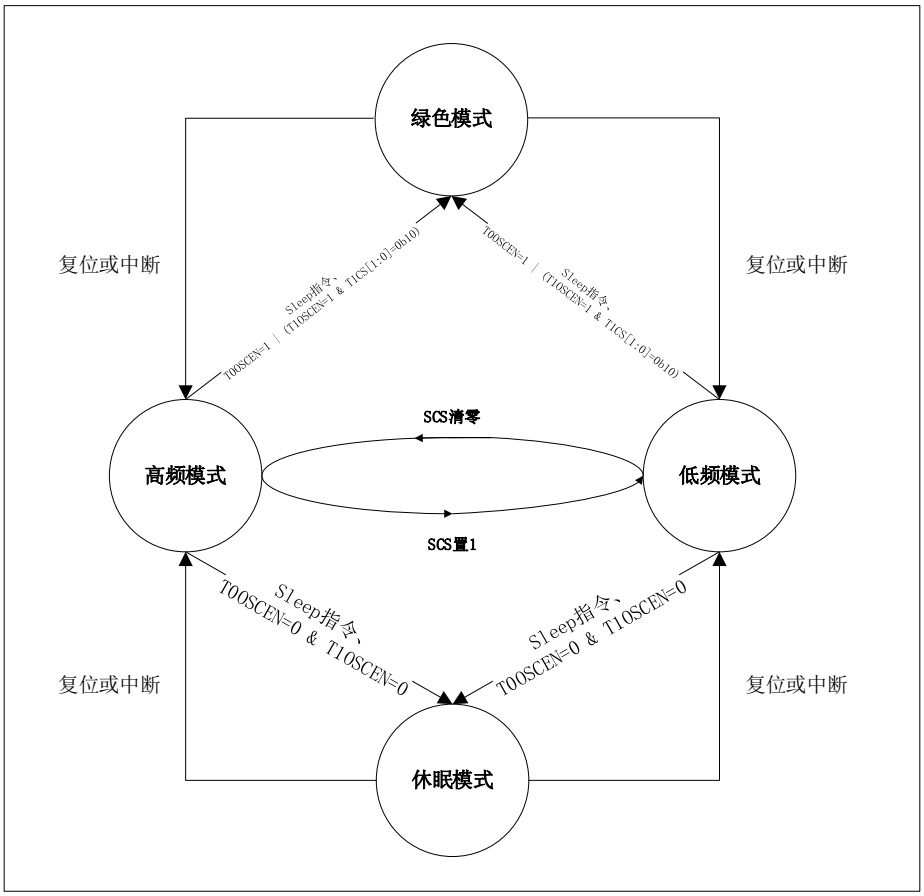
5.1 概述

BL08M003 可在如下四种工作模式之间进行切换：

- 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。

程序运行过程中，可以通过配置 T0OSCEN 或 T1OSCEN 置 1，配合 SLEEP 指令，切换绿色模式；通过配置 T0OSCEN 和 T1OSCEN 置 0，配合 SLEEP 指令，切换休眠模式。



5.2 高频模式

将 SCS 清零，系统时钟切换到高频时钟。

此时系统时钟选择为高频时钟。

5.3 低频模式

将 SCS 置 1，系统时钟切换到低频时钟。
此时系统时钟选择为低频时钟。

5.4 绿色模式

SLEEP 指令可使 MCU 进入绿色模式 T0OSCEN=1 | (T1OSCEN=1 & T1CS[1:0]=0b10)，同时对 MCU 会产生以下影响：

- 系统主时钟的振荡器停止振荡，Timer0/Timer1 保持工作（此时 Timer0/Timer1 时钟源为 LOSC）
- RAM 内容保持不变
- 所有的输入输出端口保持原态不变
- 所有的内部操作全部停止(WDT 不受影响)

以下情况使 MCU 退出休眠模式：

- 如果使能了 ADC 中断，转换完成时将唤醒 SLEEP（ADC 时钟选择 FRC）。
- 有外部中断请求发生
- 有电平变化中断请求发生
- 有 WDT 溢出发生
- 定时器 0 计数溢出发生
- 定时器 1 计数溢出发生
- 任何形式的系统复位发生

绿色模式下，系统除 Timer0 和 Timer1 工作，几乎停止了所有的操作。

注：

1、进入绿色模式并不会自动打开总中断，但只要有中断请求发生就唤醒系统，如果总中断未打开，系统继续执行下一条指令，否则响应中断服务。

2、因为 WDT 定时器的时钟源与系统主时钟无关，所以，即使系统进入绿色模式，WDT 定时器仍会工作，但在绿色模式下 WDT 只能产生唤醒信号，并不会产生复位信号。在正常工作下，当 WDT 计数溢出时，芯片复位。

5.5 休眠模式

OPTION 寄存器

38h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	WDTEN	VBGEN	-	-	PSA	PS2	PS1	PS0
R/W	R/W	R/W	-	-	R/W	R/W	R/W	R/W
POR 的值	0	1	-	-	0	0	0	0

Bit [6]　VBGEN：VBG 使能位

1 = 软件使能 VBG

0 = 软件屏蔽 VBG 功能

注：

（1）VBGEN=0，休眠唤醒后，VBG 和 BOR 同开

（2）VBGEN=1，休眠唤醒后，先开 VBG 延时开 BOR

（3）BOR 来源于 VBG，开启 VBG 时的抖动会导致 BOR 抖动，有概率误触发 BOR 复位。

SLEEP 指令可使 MCU 进入休眠模式（T0OSCEN=0&T1OSCEN=0），同时对 MCU 会产生以下影响：

- 系统主时钟的振荡器停止振荡
- RAM 内容保持不变
- 所有的输入输出端口保持原态不变
- 所有的内部操作全部停止(WDT 不受影响)

以下情况使 MCU 退出休眠模式：

- 如果使能了 ADC 中断，转换完成时将唤醒 SLEEP（ADC 时钟选择 FRC）。
- 有外部中断请求发生
- 有电平变化中断请求发生
- 有 WDT 溢出发生
- 定时器 0 外部计数溢出发生
- 定时器 1 外部计数溢出发生
- 任何形式的系统复位发生

休眠模式下，系统停止了几乎所有的操作，所以整体功耗水平非常低。

注意：

1、进入休眠模式并不会自动打开总中断，但只要有中断请求发生就唤醒系统，如果总中断未打开，系统继续执行下一条指令，否则响应中断服务。

2、因为 WDT 定时器的时钟源与系统主时钟无关，所以，即使系统进入休眠模式，WDT 定时器仍会工作，但在休眠模式下 WDT 只能产生唤醒信号，并不会产生复位信号。在正常工作下，当 WDT 计数溢出时，芯片复位。

5.6 不同工作模式下各时钟源开启情况

1、时钟源选择为：内部高频/内部低频 OSCLM = 0

	休眠模式	绿色模式	低频模式	高频模式
内部高频	关闭	关闭	关闭	打开
内部低频	关闭	打开	打开	打开

2、时钟源选择为：内部高频/外部低频 OSCLM = 1

	休眠模式	绿色模式	低频模式	高频模式
内部高频	关闭	关闭	关闭	打开
外部低频	关闭	打开	打开	打开

5.7 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。

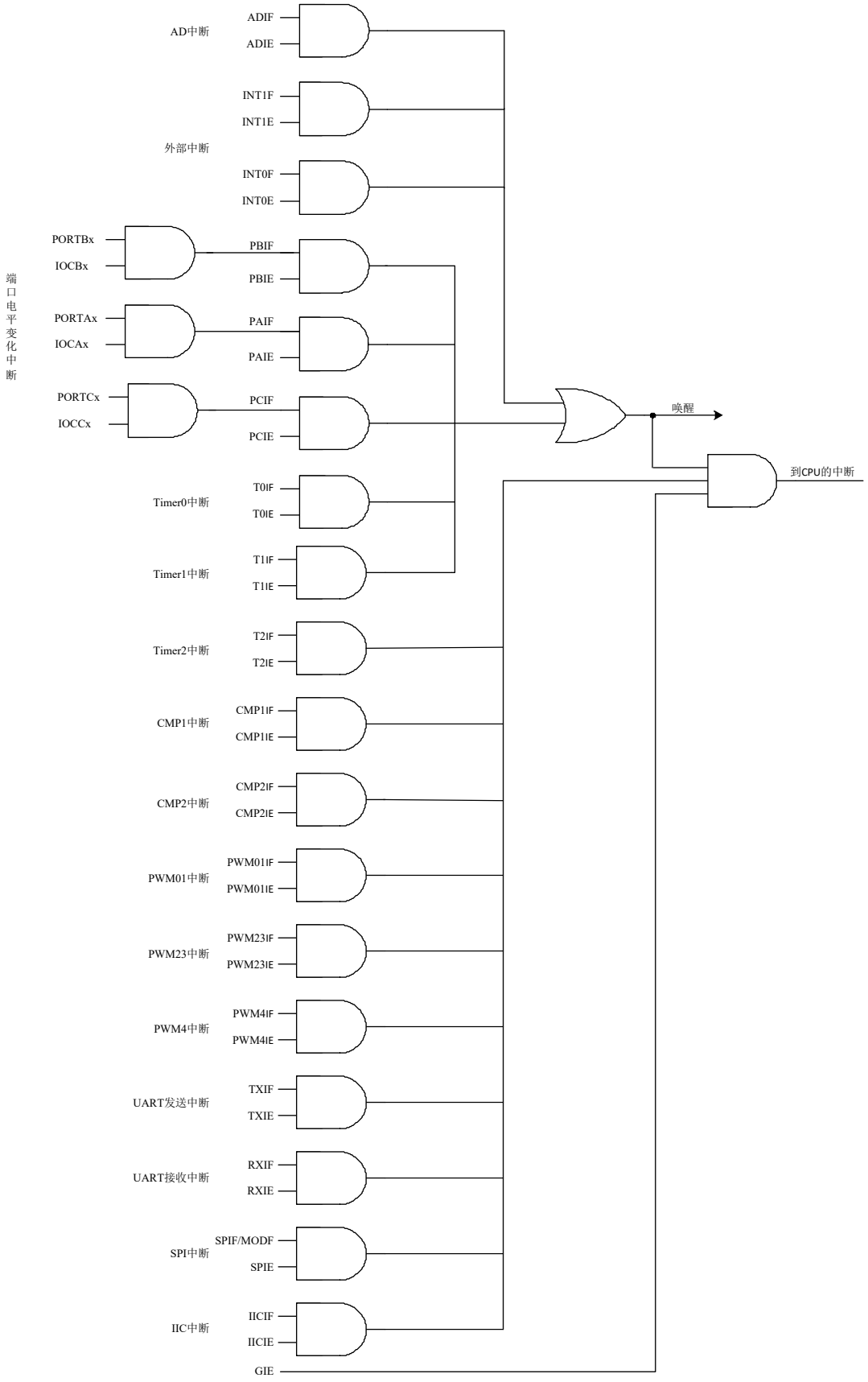
唤醒时间的计算如下：

唤醒时间 = 起振时间 + OST 定时时间

不同类型振荡器 OST 定时时间表：

振荡器类型	OST 定时时间
内部高频 RC 振荡器	176 Clock
内部低频 RC 振荡器	16 Clock
低频晶体振荡器	16Clock

6 中断


AD中断
ADIF
ADIE
外部中断
INT1F
INT1E
INT0F
INT0E
端口电平变化中断
PORTBx
IOCBx
PBIF
PBIE
PORTAx
IOCAx
PAIF
PAIE
PORTCx
IOCCx
PCIF
PCIE
唤醒
到CPU的中断
Timer0中断
T0IF
T0IE
Timer1中断
T1IF
T1IE
Timer2中断
T2IF
T2IE
CMP1中断
CMP1IF
CMP1IE
CMP2中断
CMP2IF
CMP2IE
PWM01中断
PWM01IF
PWM01IE
PWM23中断
PWM23IF
PWM23IE
PWM4中断
PWM4IF
PWM4IE
UART发送中断
TXIF
TXIE
UART接收中断
RXIF
RXIE
SPI中断
SPIF/MODF
SPIE
IIC中断
IICIF
IICIE
GIE

6.1 概述

BL08M003 提供 18 个中断源：Timer0/1/2 定时器中断、INT0/1 外部中断、PORTA/B/C 端口电平变化中断、PWM01 中断、PWM23 中断、PWM4 中断、AD 中断、CMP1、CMP2 中断、UART 发送/接收中断、SPI 中断、IIC 中断。

系统从高频或低频模式进入休眠模式时，AD 中断、INT0/1 外部中断、端口电平变化中断、以及 Timer0/1 在外部 T0CKI/T1CKI 作为计数时钟时的中断可以将单片机唤醒。

一旦程序进入中断，寄存器 INTCON 的位 GIE 将被硬件自动清零以避免再次响应其它中断。系统退出中断后，硬件自动将 GIE 置“1”，以响应下一个中断。

6.2 中断请求和标志寄存器

6.2.1 INTCON寄存器

INTCON 中存放 INT0 中断、PORTB 电平变化中断、Timer0 中断请求标志。一旦有中断请求发生，则 INTCON 中对应位将被置“1”，该请求被响应后，程序应将该标志位清零。根据 INTCON 的状态，程序判断是否有中断发生，并执行相应的中断服务。

INTCON 寄存器

09h&89h&109h&189h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE	-	INT0F	PBIF	T0IF	INT0E	PBIE	T0IE
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]　GIE：中断总使能
1 = 使能所有中断
0 = 屏蔽所有中断

Bit [5]　INT0F：INT0 中断标志位
1 = INT0 产生外部中断(必须由软件清零)
0 = INT0 未产生外部中断

Bit [4]　PBIF：PORTB 端口电平变化中断标志位
1 = PORTB 产生端口电平变化中断(必须由软件清零)
0 = PORTB 未产生端口电平变化中断

Bit [3]　T0IF：Timer0 溢出中断标志位
1 = Timer0 产生 Timer0 溢出中断(必须由软件清零)
0 = Timer0 未产生 Timer0 溢出中断

Bit [2]　INT0E：INT0 中断使能位
1 = 使能 INT0 外部中断
0 = 屏蔽 INT0 外部中断

Bit [1]　PBIE：PORTB 端口电平变化中断使能位
1 = 使能 PORTB 端口电平变化中断
0 = 屏蔽 PORTB 端口电平变化中断

Bit [0]　T0IE：Timer0 溢出中断使能位
1 = 使能 Timer0 溢出中断
0 = 屏蔽 Timer0 溢出中断

6.2.2 PIR寄存器

PIR 中存放 PORTA 端口变化中断使能位、AD 中断使能位、INT1 中断使能位、T1 溢出中断使能位，以及 INT1 中断、PORTA 电平变化中断、AD 中断请求标志、T1 溢出中断标志位。一旦有中断请求发生，则 PIR 中对应位将被置“1”，该请求被响应后，程序应将该标志位清零。根据 PIR 的状态，程序判断是否有中断发生，并执行相应的中断服务。

PIR 寄存器

2Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR	T1IE	T1IF	INT1E	INT1F	ADIE	ADIF	PAIE	PAIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7]　T1IE：Timer1 溢出中断使能位
　　1 = 使能 Timer1 溢出中断
　　0 = 屏蔽 Timer1 溢出中断
Bit [6]　T1IF：Timer1 溢出中断标志位
　　1 = Timer1 产生 Timer1 溢出中断(必须由软件清零)
　　0 = Timer1 未产生 Timer1 溢出中断
Bit [5]　INT1E：INT1 中断使能位
　　1 = 使能 INT1 外部中断
　　0 = 屏蔽 INT1 外部中断
Bit [4]　INT1F：INT1 中断标志位
　　1 = INT1 产生外部中断(必须由软件清零)
　　0 = INT1 未产生外部中断
Bit [3]　ADIE：ADC 中断使能位
　　1 = 使能 ADC 中断
　　0 = 禁止 ADC 中断
Bit [2]　ADIF：AD 中断标志位
　　1 =AD 转换已完成（必须由软件清 0）
　　0 =AD 转换未完成或尚未开始
Bit [1]　PAIE：PORTA 端口电平变化中断使能位
　　1 = 使能 PORTA 端口电平变化中断
　　0 = 屏蔽 PORTA 端口电平变化中断
Bit [0]　PAIF：PORTA 端口电平变化中断标志位
　　1 = PORTA 产生端口电平变化中断(必须由软件清零)
　　0 = PORTA 未产生端口电平变化中断

6.2.3 PIR1寄存器

PIR1 中存放 PORTC 电平变化中断使能控制位、PWM01 溢出中断允许位、CMP1、CMP2 中断使能位，以及 PORTC 电平变化中断请求标志、PWM01 中断请求标志、CMP1、CMP2 中断请求标志。一旦有中断请求发生，则 PIR1 中对应位将被置“1”，该请求被响应后，程序应将该标志位清零。根据 PIR1 的状态，程序判断是否有中断发生，并执行相应的中断服务。

PIR1 寄存器

2Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR1	CMP2IE	CMP2IF	CMP1IE	CMP1IF	PWM01IE	PWM01IF	PCIE	PCIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7]　CMP2IE：CMP2 中断使能位
1 = 使能 CMP2 中断
0 = 屏蔽 CMP2 中断

Bit [6]　CMP2IF：CMP2 中断标志位
1 = CMP2 响应中断，软件清零
0 = CMP2 未产生中断

Bit [5]　CMP1IE：CMP1 中断使能位
1 = 使能 CMP1 中断
0 = 屏蔽 CMP1 中断

Bit [4]　CMP1IF：CMP1 中断标志位
1 = CMP1 响应中断，软件清零
0 = CMP1 未产生中断

Bit [3]　PWM01IE：PWM01 中断允许位
1 =使能 PWM01 中断
0 =屏蔽 PWM01 中断

Bit [2]　PWM01IF：PWM01 中断标志位
1 = PWM01 周期计数器溢出，由硬件置 1
0 = PWM01 未溢出，软件清 0

Bit [1]　PCIE：PORTC 端口电平变化中断使能位
1 = 使能 PORTC 端口电平变化中断
0 = 屏蔽 PORTC 端口电平变化中断

Bit [0]　PCIF：PORTC 端口电平变化中断标志位
1 = PORTC 产生端口电平变化中断(必须由软件清零)
0 = PORTC 未产生端口电平变化中断

6.2.4 PIR2寄存器

PIR2 中存放 PWM23 中断允许位、PWM4 中断允许位、UART 发送和接收中断允许位，以及 PWM23、PWM4 中断请求标志、UART 发送和接收中断请求标志。一旦有中断请求发生，则 PIR2 中对应位将被置“1”，该请求被响应后，程序应将该标志位清零。根据 PIR2 的状态，程序判断是否有中断发生，并执行相应的中断服务。

PIR2 寄存器

2Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR2	RXIE	RXIF	TXIE	TXIF	PWM4IE	PWM4IF	PWM23IE	PWM23IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7]　RXIE：UART 接收中断允许位
1 = 使能 UART 接收中断
0 = 屏蔽 UART 接收中断

Bit [6]　RXIF：UART 接收中断请求标志位
1 =方式 0 时，当串行接收数据第 8 位结束时，由硬件自动置 1
方式 1 时，串行接收到停止位开始时刻由硬件置 1
0 = 软件清 0

Bit [5]　TXIE：UART 发送中断允许位
1 = 使能 UART 发送中断
0 = 屏蔽 UART 发送中断

Bit [4]　TXIF：UART 发送中断请求标志位
1 =方式 0 时，当串行发送数据第 8 位结束时，由硬件自动置 1
方式 1 时，在停止位开始发送时刻由硬件置 1
0 = 软件清 0

Bit [3]　PWM4IE：PWM4 中断允许位
1 =使能 PWM4 中断
0 =屏蔽 PWM4 中断

Bit [2]　PWM4IF：PWM4 中断标志位
1 = PWM4 周期计数器溢出，由硬件置 1
0 = PWM4 未溢出，软件清 0

Bit [1]　PWM23IE：PWM23 中断允许位
1 =使能 PWM23 中断
0 =屏蔽 PWM23 中断

Bit [0]　PWM23IF：PWM23 中断标志位
1 = PWM23 周期计数器溢出，由硬件置 1
0 = PWM23 未溢出，软件清 0

6.2.5 PIR3寄存器

PIR3 中存放 SPI 中断允许位、IIC 中断允许位、T2 溢出中断使能位，以及 SPI 传输完成中断标志位、SPI 模式故障中断标志位、IIC 中断标志位、T2 溢出中断标志位。一旦有中断请求发生，则 PIR3 中对应位将被置“1”，该请求被响应后，程序应将该标志位清零。根据 PIR3 的状态，程序判断是否有中断发生，并执行相应的中断服务。

PIR3 寄存器

2Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR3	-	SPIE	SPIF	MODF	T2IE	T2IF	IICIE	IICIF
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[6] SPIE：SPI 中断允许位
1 = 使能 SPI 中断
0 = 屏蔽 SPI 中断

Bit[5] SPIF：SPI 传输完成标志位
1 = 一次传送完成时，硬件置 1
0 = 软件写 1 清 0

Bit[4] MODF：SPI 模式故障标志位
1 = SS 引脚电平与 SPI 模式不一致时，硬件置 1（且立即切成从机模式），也做中断请求标志位
0 = 软件写 1 清 0

Bit [3] T2IE：Timer2 溢出中断使能位
1 = 使能 Timer2 溢出中断
0 = 屏蔽 Timer2 溢出中断

Bit [2] T2IF：Timer2 溢出中断标志位
1 = Timer2 产生 Timer2 溢出中断(必须由软件清零)
0 = Timer2 未产生 Timer2 溢出中断

Bit[1] IICIE：IIC 中断允许位
1 = 使能 IIC 中断
0 = 屏蔽 IIC 中断

Bit[0] IICIF：IIC 串行中断标志位
1 = 产生 IIC 通信状态码中除 0F8H 之外的状态码时置 1。必须软件清 0
0 =没有 IIC 串行中断发生

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 0008H）。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 0008H 执行中断子程序。

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

6.5 INT0/1 中断

INT0 被触发，则无论 INT0E 处于何种状态，INT0F 都会被置“1”。如果 INT0F＝1 且 INT0E＝1，系统响应该中断；如果 INT0F＝1 而 INT0E＝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。

INT1 被触发，则无论 INT1E 处于何种状态，INT1F 都会被置“1”。如果 INT1F＝1 且 INT1E＝1，系统响应该中断；如果 INT1F＝1 而 INT1E＝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。

外部中断边沿选择寄存器 INTEDG

B9h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTEDG	-	-	-	INT1EDG	-	-	-	INT0EDG
R/W	-	-	-	R/W	-	-	-	R/W
POR 的值	-	-	-	0	-	-	-	0

Bit [4] INT1EDG：INT1 中断边沿选择

1 = INT1 上升沿中断

0 = INT1 下降沿中断

Bit [0] INT0EDG：INT0 中断边沿选择

1 = INT0 上升沿中断

0 = INT0 下降沿中断

INTCON 寄存器

09h&89h& 109h&189h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE	-	INT0F	PBIF	T0IF	INT0E	PBIE	T0IE
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 [5] INT0F：INT0 中断标志位

1 = INT0 产生外部中断(必须由软件清零)

0 = INT0 未产生外部中断

Bit [2] INT0E：INT0 中断使能位

1 = 使能 INT0 外部中断

0 = 屏蔽 INT0 外部中断

PIR 寄存器

2Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR	T1IE	T1IF	INT1E	INT1F	ADIE	ADIF	PAIE	PAIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [5] INT1E：INT1 中断使能位

1 = 使能 INT1 外部中断

0 = 屏蔽 INT1 外部中断

Bit [4] INT1F：INT1 中断标志位

1 = INT1 产生外部中断(必须由软件清零)

0 = INT1 未产生外部中断

6.6 Timer0 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”，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。

INTCON 寄存器

09h&89h&109h&189h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE	-	INT0F	PBIF	T0IF	INT0E	PBIE	T0IE
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 [3]　T0IF：Timer0 溢出中断标志位

1 = Timer0 产生 Timer0 溢出中断(必须由软件清零)

0 = Timer0 未产生 Timer0 溢出中断

Bit [0]　T0IE：Timer0 溢出中断使能位

1 = 使能 Timer0 溢出中断

0 = 屏蔽 Timer0 溢出中断

6.7 Timer1 中断

当 T1 计数溢出时，Timer1 中断被触发，则无论 T1IE 处于何种状态，T1IF 都会被置“1”。如果 T1IF＝1 且 T1IE＝1，系统响应该中断；如果 T1IF＝1 而 T1IE＝0，系统并不会执行中断服务。

PIR 寄存器

2Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR	T1IE	T1IF	INT1E	INT1F	ADIE	ADIF	PAIE	PAIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7]　T1IE：Timer1 中断使能位

1 = 使能 Timer1 中断

0 = 禁止 Timer1 中断

Bit [6]　T1IF：Timer1 中断标志位

1 = Timer1 产生 Timer1 溢出中断(必须由软件清零)

0 = Timer1 未产生 Timer1 溢出中断

6.8 Timer2 中断

当 T2 计数溢出时，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，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。

PIR3 寄存器

2Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR3	-	SPIE	SPIF	MODF	T2IE	T2IF	IICIE	IICIF
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 [3]　T2IE：Timer2 中断使能位

1 = 使能 Timer2 中断

0 = 禁止 Timer2 中断

Bit [2]　T2IF：Timer2 中断标志位

1 = Timer2 产生 Timer2 溢出中断(必须由软件清零)

0 = Timer2 未产生 Timer2 溢出中断

6.9 端口电平变化中断

PORTx 电平变化中断时，则无论 PxIE 处于何种状态，相应 PxIF 都会被置“1”。如果 PxIF＝1 且 PxIE＝1，系统响应该中断；如果 PxIF＝1 而 PxIE＝0，系统并不会执行中断服务。

电平变化中断必须将 PORTx 端口设为数字输入，并将寄存器 IOCx 对应位置“1”。

注意：PORTx 端口变化中断共用中断使能控制信号 PxIE。（x=A/B/C）

IOCA 寄存器

1Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCA	-	IOCA6	IOCA5	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 [6:0] IOCAx：PORTAx 端口变化中断使能

1 = 使能 PORTAx 端口变化中断/唤醒功能

0 = 屏蔽 PORTAx 端口变化中断/唤醒功能

IOCB 寄存器

1Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCB	-	IOCB6	IOCB5	IOCB4	IOCB3	IOCB2	IOCB1	IOCB0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	-	0	0	0	0	0	0	0

Bit [6:0] IOCBx：PORTBx 端口变化中断使能

1 = 使能 PORTBx 端口变化中断/唤醒功能

0 = 屏蔽 PORTBx 端口变化中断/唤醒功能

IOCC 寄存器

1Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCC	IOCC7	IOCC6	IOCC5	IOCC4	IOCC3	IOCC2	IOCC1	IOCC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7:0] IOCCx：PORTCx 端口变化中断使能

1 = 使能 PORTCx 端口变化中断/唤醒功能

0 = 屏蔽 PORTCx 端口变化中断/唤醒功能

6.10 ADC 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 GIE 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。

PIR 寄存器

2Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR	T1IE	T1IF	INT1E	INT1F	ADIE	ADIF	PAIE	PAIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [3] ADIE：AD 中断使能位

1 = 使能 ADC 中断

0 = 禁止 ADC 中断

Bit [2] ADIF：AD 中断标志位

1 =AD 转换已完成（必须由软件清 0）

0 =AD 转换未完成或尚未开始

6.11 CMP 中断

PIR1 寄存器

2Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR1	CMP2IE	CMP2IF	CMP1IE	CMP1IF	PWM01IE	PWM01IF	PCIE	PCIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7]　CMP2IE：CMP2 中断使能位
　　1 = 使能 CMP2 中断
　　0 = 屏蔽 CMP2 中断
Bit [6]　CMP2IF：CMP2 中断标志位
　　1 = CMP2 响应中断，软件清零
　　0 = CMP2 未产生中断
Bit [5]　CMP1IE：CMP1 中断使能位
　　1 = 使能 CMP1 中断
　　0 = 屏蔽 CMP1 中断
Bit [4]　CMP1IF：CMP1 中断标志位
　　1 = CMP1 响应中断，软件清零
　　0 = CMP1 未产生中断

6.12 PWM 中断

PIR1 寄存器

2Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR1	CMP2IE	CMP2IF	CMP1IE	CMP1IF	PWM01IE	PWM01IF	PCIE	PCIF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [3]　PWM01IE：PWM01 中断允许位
　　1 =使能 PWM01 中断
　　0 =屏蔽 PWM01 中断
Bit [2]　PWM01IF：PWM01 中断标志位
　　1 = PWM01 周期计数器溢出，由硬件置 1
　　0 = PWM01 未溢出，软件清 0

PIR2 寄存器

2Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR2	RXIE	RXIF	TXIE	TXIF	PWM4IE	PWM4IF	PWM23IE	PWM23IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [3]　PWM4IE：PWM4 中断允许位
　　1 =使能 PWM4 中断
　　0 =屏蔽 PWM4 中断
Bit [2]　PWM4IF：PWM4 中断标志位
　　1 = PWM4 周期计数器溢出，由硬件置 1
　　0 = PWM4 未溢出，软件清 0
Bit [1]　PWM23IE：PWM23 中断允许位
　　1 =使能 PWM23 中断
　　0 =屏蔽 PWM23 中断
Bit [0]　PWM23IF：PWM23 中断标志位
　　1 = PWM23 周期计数器溢出，由硬件置 1

0 = PWM23 未溢出，软件清 0

6.13 IIC 中断

PIR3 寄存器

2Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR3	-	SPIE	SPIF	MODF	T2IE	T2IF	IICIE	IICIF
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[1]　IICIE：IIC 中断允许位

1 = 使能 IIC 中断

0 = 屏蔽 IIC 中断

Bit[0]　IICIF：IIC 串行中断标志位

1 = 产生 IIC 通信状态码中除 0F8H 之外的状态码时置 1。必须软件清 0

0 =没有 IIC 串行中断发生

6.14 UART 中断

PIR2 寄存器

2Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR2	RXIE	RXIF	TXIE	TXIF	PWM4IE	PWM4IF	PWM23IE	PWM23IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7]　RXIE：UART 接收中断允许位

1 = 使能 UART 接收中断

0 = 屏蔽 UART 接收中断

Bit [6]　RXIF：UART 接收中断请求标志位

1 =方式 0 时，当串行接收数据第 8 位结束时，由硬件自动置 1

方式 1 时，串行接收到停止位开始时刻由硬件置 1

0 = 软件清 0

Bit [5]　TXIE：UART 发送中断允许位

1 = 使能 UART 发送中断

0 = 屏蔽 UART 发送中断

Bit [4]　TXIF：UART 发送中断请求标志位

1 =方式 0 时，当串行发送数据第 8 位结束时，由硬件自动置 1

方式 1 时，在停止位开始发送时刻由硬件置 1

0 = 软件清 0

6.15 SPI 中断

PIR3 寄存器

2Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PIR3	-	SPIE	SPIF	MODF	T2IE	T2IF	IICIE	IICIF
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[6]　SPIE：SPI 中断允许位

1 = 使能 SPI 中断

0 = 屏蔽 SPI 中断

Bit[5]　SPIF：SPI 传输完成标志位

1 = 一次传送完成时，硬件置 1

0 = 软件写 1 清 0

Bit[4]　MODF：SPI 模式故障标志位

1 = SS 引脚电平与 SPI 模式不一致时，硬件置 1（且立即切成从机模式），也做中断请求标志位
0 = 软件写 1 清 0

7 I/O端口

7.1 I/O 端口模式

PORTA 端口方向寄存器

10h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISA	-	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
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[6:0] TRISAx：配置 PORTA 端口输入/输出模式
1 = TRISAx 对应端口为输出
0 = TRISAx 对应端口为输入
注：有关 PORTA 的相关控制寄存器。

PORTB 端口方向寄存器

11h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISB	-	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
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[6:0] TRISBx：配置 PORTB 端口输入/输出模式
1 = TRISBx 对应端口为输出
0 = TRISBx 对应端口为输入
注：有关 PORTB 的相关控制寄存器。

PORTC 端口方向寄存器

12h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit[7:0] TRISCx：配置 PORTC 端口输入/输出模式
1 = TRISCx 对应端口为输出
0 = TRISCx 对应端口为输入

注：
端口方向寄存器设为输出时，此时读端口操作为读数据寄存器的值；
端口方向寄存器设为输入时，此时读端口操作为读端口的输入电平状态。

7.2 IO 模拟模式控制寄存器

7.2.1 ANSELA寄存器

ANSELA[7:0]寄存器

22h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELA	-	ANSELA6	ANSELA5	ANSELA4	ANSELA3	ANSELA2	ANSELA1	ANSELA0
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

ANSELA[6:0]：PORTAx 端口数字模拟控制位

1 = 数字模式

0 = 模拟模式

7.2.2 ANSELB寄存器

ANSELB[7:0]寄存器

23h	Bit 7	Bit 6	Bit 5	Bit4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELB	-	ANSELB6	ANSELB5	ANSELB4	ANSELB3	ANSELB2	ANSELB1	ANSELB0
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

ANSELB[6:0]：PORTBx 端口数字模拟控制位

1 = 数字模式

0 = 模拟模式

7.2.3 ANSELC寄存器

ANSELC[7:0]寄存器

24h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELC	ANSELC7	ANSELC6	ANSELC5	ANSELC4	ANSELC3	ANSELC2	ANSELC1	ANSELC0
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

ANSELC[7:0]：PORTCx 端口数字模拟控制位

1 = 数字模式

0 = 模拟模式

注：

ANSELA 上电初始值为 B'0000 0000'，即作为模拟口。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。

ANSELA[6:0]对应 AN0~AN6（PA0~PA6），ANSELB[6:0]对应 AN8~AN14(PB0~PB6)，ANSELC[7:0]对应 AN15~AN22(PC0~PC7)

7.3 I/O 上拉模式

7.3.1 WPUA寄存器

WPUA 寄存器

16h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPUA	-	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0
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 [6:0] WPUA：PORTAx 上拉控制

1 = 使能 PORTAx 输入上拉功能

0 = 屏蔽 PORTAx 输入上拉功能

7.3.2 WPUB寄存器

WPUB 寄存器

17h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPUB	-	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0
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 [6:0] WPUB：PORTBx 上拉控制

1 = 使能 PORTBx 输入上拉功能

0 = 屏蔽 PORTBx 输入上拉功能

7.3.3 WPUC寄存器

WPUC 寄存器

18h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPUC	WPUC7	WPUC6	WPUC5	WPUC4	WPUC3	WPUC2	WPUC1	WPUC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7:0] WPUC：PORTCx 上拉控制

1 = 使能 PORTCx 输入上拉功能

0 = 屏蔽 PORTCx 输入上拉功能

7.4 I/O 下拉模式

7.4.1 WPDA寄存器

WPDA 寄存器

19h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPDA	-	WPDA6	WPDA5	WPDA4	WPDA3	WPDA2	WPDA1	WPDA0
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 [6:0]　WPDA：PORTAx 下拉控制

1 = 使能 PORTAx 输入下拉功能

0 = 屏蔽 PORTAx 输入下拉功能

7.4.2 WPDB寄存器

WPDB 寄存器

1Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPDB	-	WPDB6	WPDB5	WPDB4	WPDB3	WPDB2	WPDB1	WPDB0
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 [6:0]　WPDB：PORTBx 下拉控制

1 = 使能 PORTBx 输入下拉功能

0 = 屏蔽 PORTBx 输入下拉功能

7.4.3 WPDC寄存器

WPDC 寄存器

1Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPDC	WPDC7	WPDC6	WPDC5	WPDC4	WPDC3	WPDC2	WPDC1	WPDC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7:0]　WPDC：PORTCx 下拉控制

1 = 使能 PORTCx 输入下拉功能

0 = 屏蔽 PORTCx 输入下拉功能

7.5 I/O 端口数据寄存器

7.5.1 PORTA寄存器

PORTA 端口数据寄存器

13h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTA	-	PORTA6	PORTA5	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

7.5.2 PORTB寄存器

PORTB 端口数据寄存器

14h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTB	-	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
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

7.5.3 PORTC寄存器

PORTC 端口数据寄存器

15h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0
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

7.6 复用端口映射控制寄存器

多个输出映射到一个端口 IO 上时，只能有一个输出有效，下面是默认优先级。

优先级顺序	复用端口功能
1	PWM01
2	PWM23
3	PWM4
4	UART
5	SPI
6	IIC
7	CMP1
8	CMP2

7.6.1 PWM_MAP寄存器

PWM_MAP 寄存器

26h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM_MAP	PWM3_MAP[1:0]		PWM2_MAP[1:0]		PWM1_MAP[1:0]		PWM0_MAP[1:0]	
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

Bit[7:6] PWM3_MAP[1:0]：PWM3 输出引脚选择
- 11 = PWM3 选择在 PC1
- 10 = PWM3 选择在 PC5
- 01 = PWM3 选择在 PB6
- 00 = PWM3 选择在 PA3

Bit[5:4] PWM2_MAP[1:0]：PWM2 输出引脚选择
- 11 = PWM2 选择在 PA4
- 10 = PWM2 选择在 PC4
- 01 = PWM2 选择在 PB5
- 00 = PWM2 选择在 PA2

Bit[3:2] PWM1_MAP[1:0]：PWM1 输出引脚选择
- 11 = PWM1 选择在 PA6
- 10 = PWM1 选择在 PC3
- 01 = PWM1 选择在 PB2
- 00 = PWM1 选择在 PA1

Bit[1:0] PWM0_MAP[1:0]：PWM0 输出引脚选择
- 11 = PWM0 选择在 PA5
- 10 = PWM0 选择在 PC2
- 01 = PWM0 选择在 PB1
- 00 = PWM0 选择在 PA0

注：
1、PWM 输出端口选择需要在配置使能 PWM 输出时才有效
2、选择端口后，需配置为数字输出模式

7.6.2 PWM4_MAP寄存器

PWM4_MAP 寄存器

27h	Bit 7	Bit 6	Bit 5	Bit 4	Bit3	Bit 2	Bit 1	Bit 0
PWM4_MAP	-	-	-	-	-	-	PWM4_MAP[1:0]	
R/W	-	-	-	-	-	-	R/W	R/W
POR 的值	-	-	-	-	-	-	0	0

Bit[1:0] PWM4_MAP[1:0]：PWM4 输出引脚选择

11 = PWM4 选择在 PB4

10 = PWM4 选择在 PC7

01 = PWM4 选择在 PC0

00 = PWM4 选择在 PB0

注：

1、PWM 输出端口选择需要在配置使能 PWM 输出时才有效

2、选择端口后，需配置为数字输出模式

7.6.3 SPI_MAP寄存器

SPI_MAP 寄存器

29h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPI_MAP	SS_MAP[1:0]		SCK_MAP[1:0]		MISO_MAP[1:0]		MOSI_MAP[1:0]	
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

Bit[7:6] SS_MAP[1:0]：SPI 片选 SS 引脚选择

11 = 无映射

10 = SPI 片选 SS 引脚选择在 PB2

01 = SPI 片选 SS 引脚选择在 PB4

00 = 无映射

Bit[5:4] SCK_MAP[1:0]：SPI 时钟 SCK 引脚选择

11 = 无映射

10 = SPI 时钟 SCK 引脚选择在 PC0

01 = SPI 时钟 SCK 引脚选择在 PC2

00 = 无映射

Bit[3:2] MISO_MAP[1:0]：MISO 引脚选择

11 = 无映射

10 = MISO 引脚选择在 PC7

01 = MISO 引脚选择在 PB1

00 = 无映射

Bit[1:0] MOSI_MAP[1:0]：MOSI 引脚选择

11 = 无映射

10 = MOSI 引脚选择在 PA2

01 = MOSI 引脚选择在 PB3

00 = 无映射

注意：

主机模式下 SS、SCK、MOSI 映射端口配置为数字输出模式，MISO 映射端口配置为数字输入模式。

从机模式下 SS、MISO 映射端口配置为数字输出模式，SCK、MOSI 映射端口配置为数字输入模式。

7.6.4 UART_MAP寄存器

UART_MAP 寄存器

28h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
UART_MAP	-	-	RX_MAP[1:0]		-	-	TX_MAP[1:0]	
R/W	-	-	R/W	R/W	-	-	R/W	R/W
POR 的值	-	-	0	0	-	-	0	0

Bit[5:4] RX_MAP[1:0]：UART 接收引脚选择

11 = 无映射

10 = UART 接收引脚 RX 选择在 PB5

01 = UART 接收引脚 RX 选择在 PA0

00 = 无映射

Bit[1:0] TX_MAP[1:0]：UART 发送引脚选择

11 =无映射

10 = UART 发送引脚 TX 选择在 PC0

01 = UART 发送引脚 TX 选择在 PA1

00 = 无映射

注意：

选择端口后，用户需要手动配置 IO 模式

7.6.5 IIC_MAP寄存器

IIC_MAP 寄存器

2Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IIC_MAP	-	-	SCL_MAP[1:0]		-	-	SDA_MAP[1:0]	
R/W	-	-	R/W	R/W	-	-	R/W	R/W
POR 的值	-	-	0	0	-	-	0	0

Bit[5:4] SCL_MAP[1:0]：IIC 时钟线 SCL 引脚选择

11 = 无映射

10 = IIC 时钟线 SCL 引脚选择在 PA6

01 = IIC 时钟线 SCL 引脚选择在 PC5

00 = 无映射

Bit[1:0] SDA_MAP[1:0]：IIC 数据线 SDA 引脚选择

11 = 无映射

10 = IIC 数据线 SDA 引脚选择在 PA5

01 = IIC 数据线 SDA 引脚选择在 PC6

00 = 无映射

注意：

1、选择端口后，用户需要手动配置 IO 模式

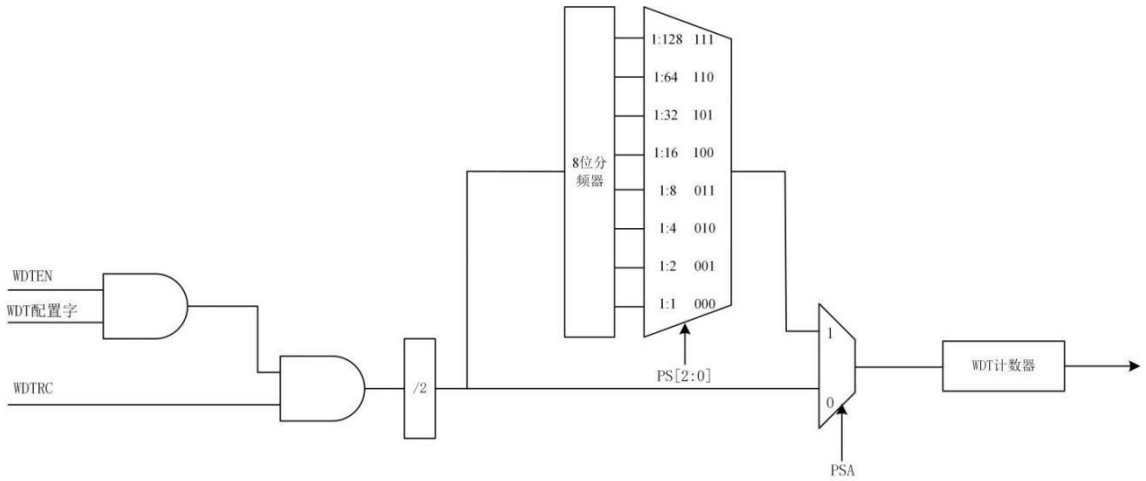
2、当使能 IIC 映射后，默认将 SDA 口设为开漏带上拉输出，但 SCL 依旧需要用户手动配置模式。

8 定时器

8.1 看门狗定时器

WDT 定时器的时钟源于内部低频 RC 振荡器，并可以选择是否经过预分频器。WDT 定时器可以用来产生 WDT 复位或唤醒休眠模式。WDT 振荡器是否开启由 OPTION 中的 WDTE 和软件的 WDTEN 位共同决定。

只有 WDTEN 为 0 时，WDT 定时器被软禁止；为 1 时软使能，若要 WDT 使能还需要 OPTION 的 WDTE 使能。



因为 WDT 定时器的时钟源与系统主时钟无关，所以，即使系统进入休眠模式，WDT 定时器仍会工作，但在休眠模式下 WDT 只能产生唤醒信号，并不会产生复位信号。在正常工作下，当 WDT 计数溢出时，芯片复位。

WDT 的基本溢出时间由 OPTION 的 WDTOV_SEL 决定，无分频的周期范围是 3.6ms－230.4ms。WDT 和 T0 共用分频器，当分频器给 T0 时，WDT 为 1 分频（无分频）；反之当分频器给 WDT 时 T0 为 1 分频（无分频），由 PSA、PS[2:0]决定。

若要更长的时间可对 WDT 进行分频，分频后 WDT 溢出时间为基本溢出时间的分频倍数。例如 OPTION 中 WDTOV_SEL 选择的基本时间为 14.4ms，软件进行 4 分频，则溢出时间为 14.4*4=57.6ms。

OPTION 寄存器

38h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	WDTEN	VBGEN	-	-	PSA	PS2	PS1	PS0
R/W	R/W	R/W	-	-	R/W	R/W	R/W	R/W
POR 的值	0	1	-	-	0	0	0	0

Bit[7]　WDTEN：看门狗使能位

1 = 软件使能 WDT

0 = 软件屏蔽 WDT 功能

看门狗定时器使能需要 WDT 定时器配置字设置使能，并且系统寄存器 WDTEN 位软件置 1。

当系统处于休眠模式，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。

8.2 T0 定时/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(时钟选择外部引脚 T0CKI 或选择低频时钟(低频时钟选择外部低频晶体振荡器))

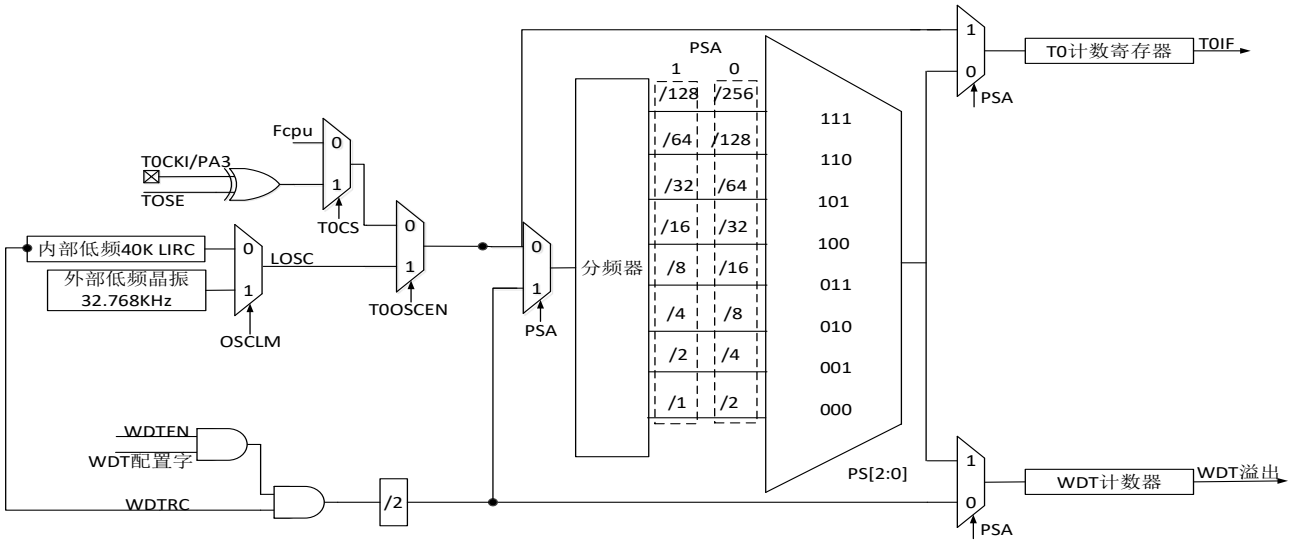
T0 的计数时钟可选择 cpu 时钟 Fcpu、外部管脚 T0CKI(PA3)、系统低频时钟 LOSC。

预分频器为定时器 T0 与 WDT 定时器共用；

T0 是一个递增计数器，它的值可以读写，当计数到从 FF 溢出到 0 时，产生 T0 溢出信号，将中断标志位 T0IF 置 1。

T0 的计数周期=（256-T0[7:0]）*（1/（所选时钟源频率/分频数））

Timer0 模块和预分频器（与 WDT 共享）框图



T0 控制寄存器 T0CON

30h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T0CON	-	-	-	-	-	T0CS	T0SE	T0OSCEN
R/W	-	-	-	-	-	R/W	R/W	R/W
POR 的值	-	-	-	-	-	0	0	0

Bit [2]　T0CS：Timer0 时钟源选择

1 = T0CKI（当 Timer0 选择 T0CKI 作为计数时钟时，T0CKI 口由硬件设为施密特端口）

0 = Fcpu

Bit [1]　T0SE：Timer0 外部 T0CKI 计数沿选择

1 = T0CKI 下降沿计数

0 = T0CKI 上升沿计数

Bit[0]　T0OSCEN：T0 时钟选择

1 = T0 计数时钟为 Losc

0 = T0 计数时钟由 T0CS 决定

注：LOSC 时钟为系统低频时钟

OPTION 寄存器

38h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	WDTEN	VBGEN	-	-	PSA	PS2	PS1	PS0
R/W	R/W	R/W	-	-	R/W	R/W	R/W	R/W
POR 的值	0	1	-	-	0	0	0	0

Bit [3]　PSA：预分频分配

1 = WDT

0 = Timer0

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
000	1 : 2	1：1
001	1 : 4	1：2
010	1：8	1：4
011	1：16	1：8
100	1：32	1：16
101	1：64	1：32
110	1：128	1：64
111	1：256	1：128

Timer0 工作模式选择

T0OSCEN	T0CS	T0SE	Timer0 工作状态
0	0	x	定时器模式，计数时钟 FCPU， 休眠下停止
0	1	0	计数器模式，计数时钟 T0CKI，上升沿计数 休眠下工作，溢出中断可唤醒 SLEEP
0	1	1	计数器模式，计数时钟 T0CKI，下降沿计数 休眠下工作，溢出中断可唤醒 SLEEP
1	x	x	定时唤醒模式，计数时钟 LOSC 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。

T0 寄存器

2Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T0	Timer0 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	X	X	X	X	X	X	X	X

Bit[7:0]　T0 的值，用于设定定时时间

注：关于 Timer0 的时钟源选择，需注意

1、Timer0 时钟源选择为外部时钟源 T0CKI 时，具有唤醒功能。

2、Timer0 时钟源选择 Losc 时，且系统低频时钟选择外部低频晶体振荡器时，具有唤醒功能

8.3 T1 定时器/计数器

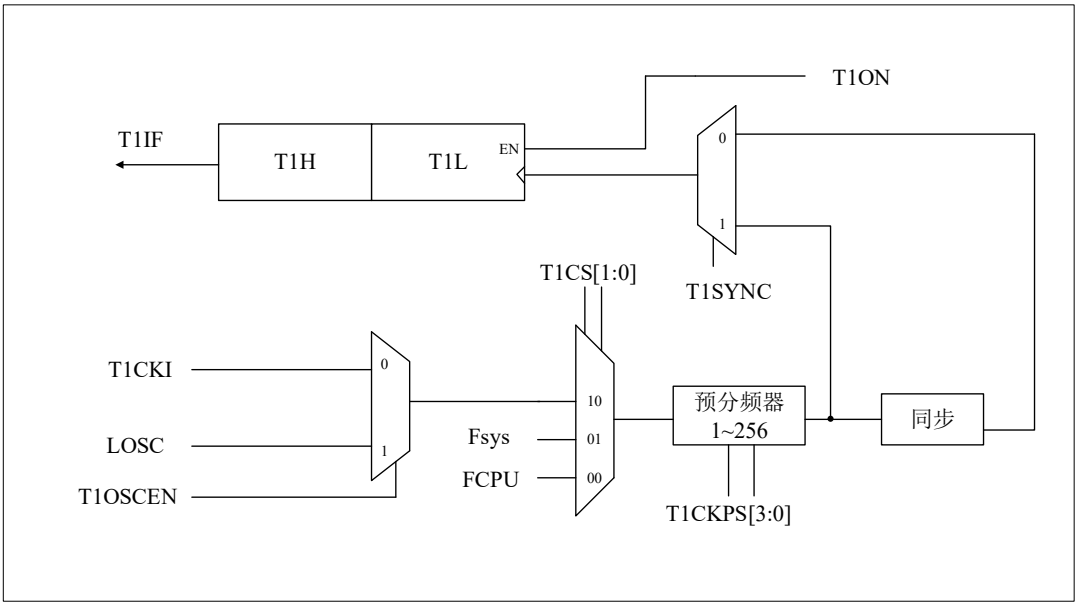
Timer1 定时器/计数器模块具有如下功能：

- 16 位可编程定时器
- 外部事件计数器
- 可通过预分频比设置 T1 计数/定时频率
- 溢出中断功能
- 溢出唤醒功能

T1 的计数时钟可选择系统时钟 Fosc、cpu 时钟 Fcpu、外部管脚 T1CKI（PB2）、系统低频时钟 LOSC。

T1 是一个递增计数器，它的值可以读写，当计数到从 FFFF 溢出到 0 时，产生 T1 溢出信号，将中断标志位 T1IF 置 1。

8.3.1 定时器/计数器T1系统框图



8.3.2 定时器/计数器T1相关寄存器

8.3.2.1 T1CON寄存器

T1 控制寄存器 T1CON

31h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T1CON	T1CS1	T1CS0	T1OSCEN	T1SYNC	-	-	-	T1ON
R/W	R/W	R/W	R/W	R/W	-	-	-	R/W
POR 的值	0	0	0	0	-	-	-	0

Bit [7:6] T1CS[1:0]：具体功能详见下表

Bit [5] T1OSCEN：具体功能详见下表

Bit [4] T1SYNC：具体功能详见下表

Bit [0] T1ON：定时器 Timer1 使能位

1 = 打开定时器 Timer1

0 = 关闭定时器 Timer1

8.3.2.2 T1L、T1H寄存器

T1 计数寄存器

32h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T1L	T1 计数寄存器低 8 位							
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

33h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T1H	T1 计数寄存器高 8 位							
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

8.3.2.3 PR1CON寄存器

T1 控制寄存器 PR1CON

34h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PR1CON	-	-	-	-	T1CKPS3	T1CKPS2	T1CKPS1	T1CKPS0
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] T1CKPS[3:0]：T1 时钟预分频

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_{osc}）
1	0	0	T1CKI 引脚上的外部时钟
1	0	1	低频系统时钟

注：
1、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。
2、当 Timer1 选择 T1CKI 作为计数时钟时，T1CKI 口由硬件设为施密特端口

Timer1 输入时钟预分频比选择

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

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

Timer1 工作模式选择

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

注：
1、T1 为 16 位计时器，在溢出中断重新赋值时应先 T1H，后 T1L，避免 T1L 在操作中的进位被覆盖；清空时则应先 T1L 后 T1H，避免 T1L 进位意外进入 T1H 造成清空失败。
2、Timer1 工作于同步计数器模式和同步定时唤醒模式时，不能唤醒 SLEEP
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

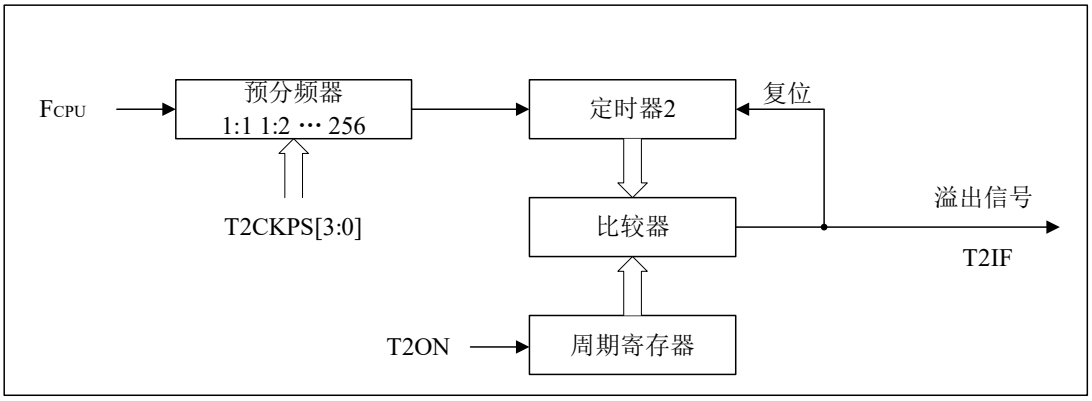
8.4T2 定时器/计数器

Timer2 定时器模块具有如下功能：

- 8 位可编程定时器
- 可通过预分频比设置 T2 定时频率
- 溢出中断功能

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 F_{CPU}，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。

8.4.1 定时器/计数器T2系统框图



8.4.2 定时器/计数器T2相关寄存器

8.4.2.1 T2CON寄存器

T2CON 寄存器

36h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T2CON	SCS	-	-	T2CKPS3	T2CKPS2	T2CKPS1	T2CKPS0	T2ON
R/W	R/W	-	-	R/W	R/W	R/W	R/W	R/W
POR 的值	0	-	-	0	0	0	0	0

Bit [4:1] T2CKPS[3:0]：T2 时钟预分频

Bit [0] T2ON： Timer2 模块使能位

0 = 禁止 Timer2 模块

1 = 使能 Timer2 模块

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

8.4.2.2 T2寄存器

Timer2 计数寄存器 T2

36h	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

Bit [7:0] T2 计数寄存器

8.4.2.3 PR2寄存器

Timer2 周期寄存器 PR2

37h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PR2	Timer2 周期寄存器							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	1	1	1	1	1	1	1	1

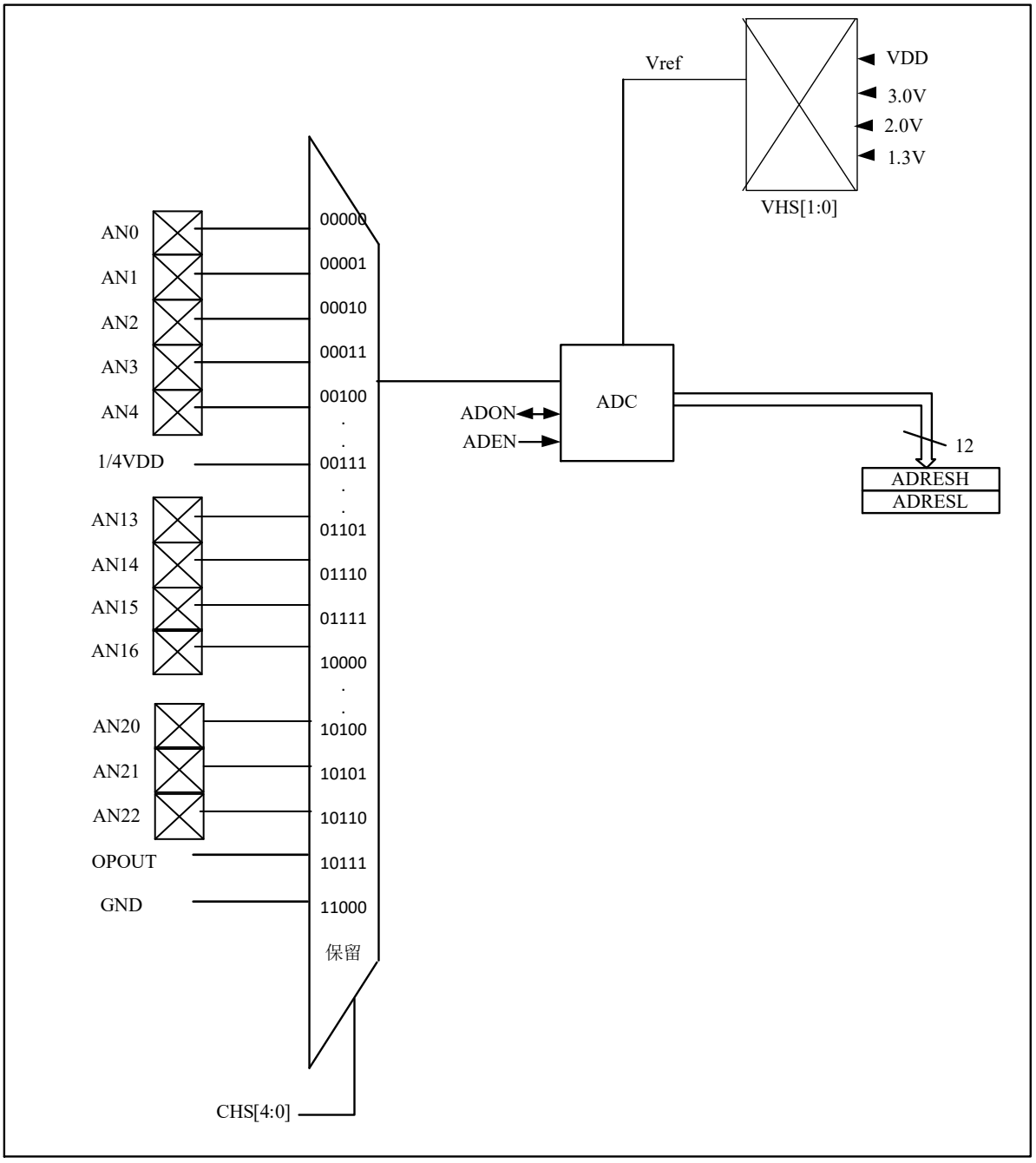
Timer2 定时器的输入时钟为指令时钟 F_{CPU}，输入时钟通过预分频器产生 Timer2 计数信号，当计数到与周期寄存器（PR2）的值相同时产生 Timer2 溢出信号。

Timer2 溢出时间 = （PR2 + 1）* 预分频比/Fcpu。

9 模数转换 (ADC)

BL08M003 具有一个 12 位转换分辨率的模数转换器，共有 22 个外部模拟输入通道，1 个内部电压检测通道、1 个内部 GND 通道、1 个内部 OPOUT 输入通道。

ADC 的等效电路：



9.1 A/D 寄存器

9.1.1 ADCON0寄存器

ADCON0 寄存器

3Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCON0	-	CHS [4:0]					ADON	ADEN
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[6:2] CHS[4:0]：AD 通道选择位

Bit[1] ADON 开始 AD 转换使能位

1 = 开始一次 AD 转换

0 = AD 转换完成后，硬件自动清零

Bit[0] ADEN： AD 使能位

1 = 使能 ADC

0 = 屏蔽 ADC

9.1.2 ADCON1寄存器

ADCON1 寄存器

3Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCON1	-	ADCS[2:0]			-	VHS[1:0]		-
R/W	-	R/W	R/W	R/W	-	R/W	R/W	-
POR 的值	-	0	0	0	-	0	0	-

Bit[6:4] ADCS[2:0]： ADC 时钟选择位

Bit[2:1] VHS[1:0]：ADC 内部参考电压选择位

ADC 模拟通道选择

CHS [4:0]	模拟通道
00000	AN0
00001	AN1
00010	AN2
00011	AN3
00100	AN4
00101	AN5
00110	AN6
00111	内部 1/4VDD(AN7)
01000	AN8
01001	AN9
01010	AN10
01011	AN11
01100	AN12
01101	AN13
01110	AN14
01111	AN15
10000	AN16
10001	AN17
10010	AN18
10011	AN19
10100	AN20
10101	AN21
10110	AN22

10111	OPOUT
11000	GND
其他	保留

ADC 参考电压选择

VHS[1:0]	参考电压
00	内部 VDD
01	内部 3.0V
10	内部 2.0V
11	内部 1.3V

9.1.3 ADRESL、ADRESH寄存器

ADRESL

39h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADRESL	ADC 结果寄存器低 4 字节				-	-	-	-
R/W	R/W	R/W	R/W	R/W	-	-	-	-
POR 的值	0	0	0	0	-	-	-	-

ADRESH

3Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADRESH	ADC 结果寄存器高 8 字节							
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

注意：
1、AN7 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 精度为 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

9.1.4 ADCJMD寄存器

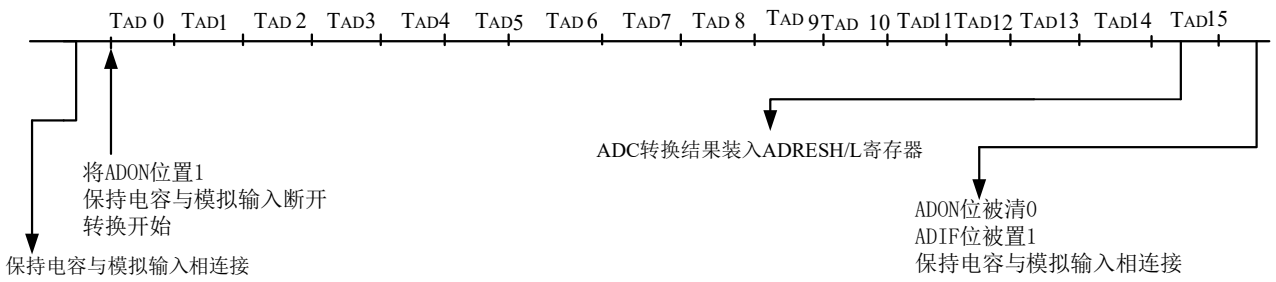
ADC-OFFSET 校准
ADCJMD 寄存器

3Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCJMD	-	-	ADJTRIM[5:0]					
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	-	-	0	0	0	0	0	0

Bit[5:0] ADJTRIM[5:0]：OFFSET 校准字

注：
ADCJMD 为 ADC 的 offset 校准寄存器，全 0 为-64 LSB，默认值 100000 为 0LSB，全 1 为+62 LSB。每次上电需要进行校准一次；在使用 ADCJMD 寄存器进行校准时，需要配置采集通道 CHS[4:0] =11000 为内部 GND 通道。

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）		cpu 频率（Fcpu）:2MHz
ADC 时钟源	ADCS[2:0]	典型值
Fcpu	000	8us
Fcpu /2	001	16us
Fcpu/4	010	32us
Fcpu /8	011	64us
Fcpu /16	100	128us
Fcpu /32	101	256us
Fcpu /64	110	512us
FRC	111	32us

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。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。

注：
1、ADC 的运行时钟最高为 2M，所以用户需根据 Fcpu 的频率来设置分频系数（ADCS[2:0]）
2、当 Fcpu 高于 1 MHz 时，仅当在休眠下进行转换时才推荐使用 FRC 时钟源。FRC 就是内部 IRC 的分频，固定是 500Khz

9.2 ADC 使用

1. 配置端口：
- 设置 TRISx 寄存器禁止引脚输出
- 设置 ANSELx 寄存器配置引脚为模拟输入（x=A/B/C）

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 VHS[1:0]
- 选择 ADC 输入通道，设置 CHS[4:0]
- 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. 等待所需稳定时间 200us
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     TRISA,0             ;设置 AD 口为输入
                MOVLW   B'01010000'         ;INNER REF Fcpu/32 ADRESH[7:0] ADRESL[7,4]
                MOVWF   ADCON1              ;配置 AD 通道
                MOVLW   B'00000000'
                MOVWF   ANSELA              ;PA0 作为模拟输入

                BCF     ADCON0,CHS4
                BCF     ADCON0,CHS3
                BCF     ADCON0,CHS2
                BCF     ADCON0,CHS1
                BCF     ADCON0,CHS0
                BSF     ADCON1,VHS1
                BCF     ADCON1,VHS0         ;参考电压为内部 2.0V
                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
                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、休眠模式下，禁止 ADC 并将 AD 参考电压设为非内部 VDD 以降低功耗。

9.3 ADC-OFFSET 校准

1. 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 VHS[1:0]
   - 选择 ADC 输入通道，设置 CHS[4:0]，选择 GND 通道
   - 配置 ADCJMD 为 0x20
   - 使能 ADC 模块，设置 ADEN
2. 等待所需稳定时间 200us
3. 设置 ADON = 1 进行 ADC 转换（设置 2 次）
4. 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）
5. 丢弃第一次转换结果，读取第二次转换结果
6. 判断转换结果，若为 0（负偏）执行步骤 7，非 0（正偏）执行步骤 12

7. ADCJMD+1，循环执行步骤 8~11
8. 设置 ADON = 1 进行 ADC 转换（设置 2 次）
9. 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）
10. 丢弃第一次转换结果，读取第二次转换结果
11. 直到 ADC 采集结果为非 0 时，进行 ADCJMD-1，保存此值，完成 ADC-OFFSET 校准（若循环次数大于 31，ADCJMD 保存为 0x3F，完成 ADC-OFFSET 校准），执行步骤 17

12. ADCJMD-1，循环执行步骤 13~16
13. 设置 ADON = 1 进行 ADC 转换（设置 2 次）
14. 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）
15. 丢弃第一次转换结果，读取第二次转换结果
16. 直到 ADC 采集结果为 0 时，保存 ADCJMD 值，完成 ADC-OFFSET 校准（若循环次数大于 32，ADCJMD 保存为 0x00，完成 ADC-OFFSET 校准
17. 完成校准后，进行 ADCJMD+5 补偿

> 注：
> 若程序需要使用 ADC 模块，则每次上电或复位均需要进行一次 OFFSET 校准，完成校准后进行 ADCJMD+5 补偿。

➢ 例：ADC_OFFSET 校准配置。

```
int number = 0;
int ADC_NEGATIVE_FLAG = 0;        //ADC_offset 负偏标志位
int ADC_POSITIVE_FLAG = 0;        //ADC_offset 正偏标志位
int ADC_TIM_FLAG = 0;             //ADC_offset 校准标志位
typedef unsigned short  Gui_AdcValue; //可仿真在 view-watch 窗口查看采集值
```

```
void Adc_Offset1(void)
{
    ADC_TIM_FLAG = 1;
    ADCON0 = 0x60;              //内部 GND 通道
    ADCON1 = 0x44;              //内部参考电压 2.0V，ADC 时钟源 16 分频
    ADCJMD= 0x20;
    ADEN = 1;
    Delay_Us(200);

    ADON = 1;                   //启动转换
    while(ADON);
    asm("nop");
    ADON = 1;                   //启动转换
    while(ADON);
    asm("nop");

    Gui_AdcValue = ADRESH;
    Gui_AdcValue = Gui_AdcValue << 8;
    Gui_AdcValue = Gui_AdcValue | ADRESL;
    Gui_AdcValue = Gui_AdcValue >> 4;
    if(Gui_AdcValue == 0)
    {
        ADC_NEGATIVE_FLAG = 1;    //负偏
    }
    else
    {
        ADC_POSITIVE_FLAG = 1;    //正偏
    }

    while(ADC_TIM_FLAG)
    {
        if(ADC_NEGATIVE_FLAG == 1)      //负偏
        {
            ADCJMD = ADCJMD + 1;
            number++;
            ADON = 1;                   //启动转换
            while(ADON);
            asm("nop");
            ADON = 1;                   //启动转换
            while(ADON);
            asm("nop");

            Gui_AdcValue = ADRESH;
            Gui_AdcValue = Gui_AdcValue << 8;
            Gui_AdcValue = Gui_AdcValue | ADRESL;
            Gui_AdcValue = Gui_AdcValue >> 4;
            if(Gui_AdcValue != 0)
            {
                ADCJMD = ADCJMD - 1;
                ADC_NEGATIVE_FLAG = 0;
                ADC_TIM_FLAG  = 0;
            }
            if((ADCJMD == 0x3F)||(number == 31))
            {
```

```
                ADC_NEGATIVE_FLAG= 0;
                ADC_TIM_FLAG  = 0;
            }
        }

        if(ADC_POSITIVE_FLAG == 1)  //正偏
        {
            ADCJMD = ADCJMD - 1;
            number++;
            ADON = 1;                   //启动转换
            while(ADON);
            asm("nop");
            ADON = 1;                   //启动转换
            while(ADON);
            asm("nop");

            Gui_AdcValue = ADRESH;
            Gui_AdcValue = Gui_AdcValue << 8;
            Gui_AdcValue = Gui_AdcValue | ADRESL;
            Gui_AdcValue = Gui_AdcValue >> 4;
            if((Gui_AdcValue == 0)||(ADCJMD == 0x00)||(number == 32))
            {
                ADC_POSITIVE_FLAG = 0;
                ADC_TIM_FLAG  = 0;
            }
        }
        if(ADCJMD <= 58)
        {
            ADCJMD = ADCJMD + 5;
        }
        else
        {
            ADCJMD = 0x3F;
        }
    }
}
```

10 CMP

- 一路比较器 CMP1
- CMP1 正端输入和负端输入均可配置为外部端口输入进行比较
- CMP1 可检测 VDD 电压，负端选择 1.3V，正端选择 VDD 经过分压电阻后的电压进行比较，具体检测电压点见 10.4.3 章节的对比表
- CMP1 比较器可响应中断，可配上升沿或下降沿中断

10.1 CMP 功能框图

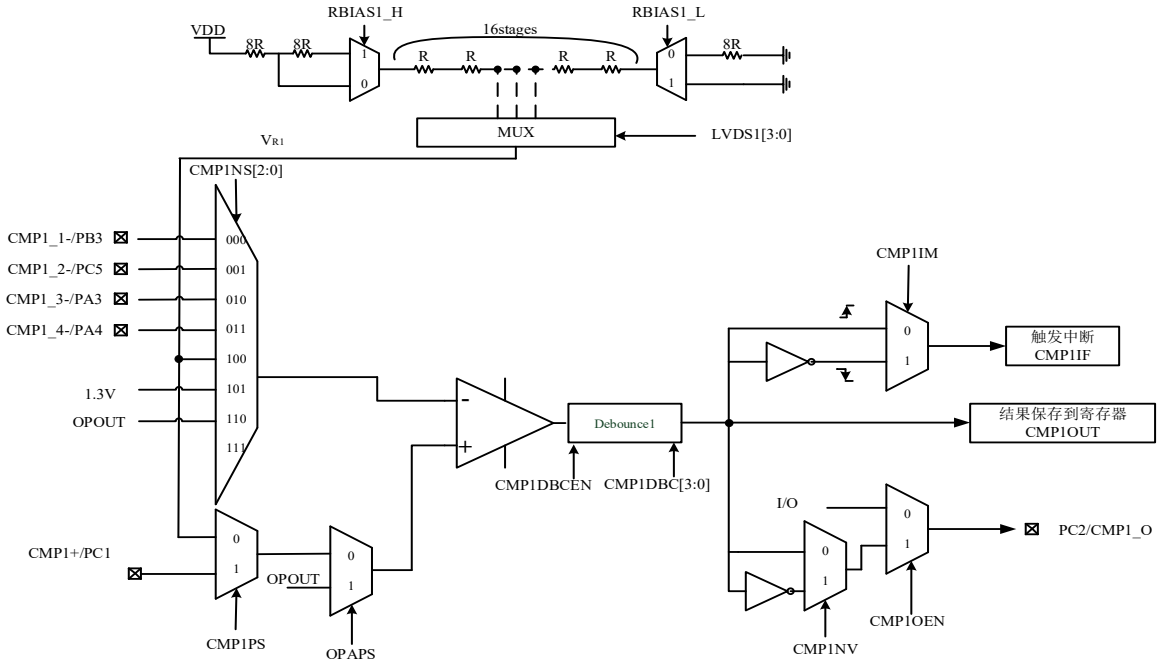


图10-1:CMP1的功能框图

10.2 CMP1 功能特性

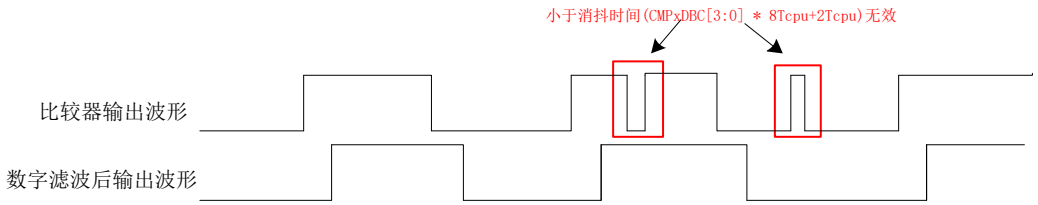
- 比较器失调电压≤±13mv;
- 输入共模电压范围：0V~(VDD-1.3V);
- 内置 1 个电阻分压模块，参考电压为 VDD;
- 比较器结果可选上升沿或下降沿触发中断；
- CMP1 结果可选择从 PC2 口输出，且支持取反输出。

10.3 CMP1 消抖

比较器 CMP1 内置的数字滤波器，用于比较器的输出信号进行数字滤波，通过配置 CMP1DBCEN 位控制是否开启数字滤波器，为“1”使能数字滤波，为“0”无滤波。

用户可使用滤波功能过滤系统噪声，比如通过端口传入噪声等，避免因噪声引起比较器 CMP 系统的误动作。

可配置 CMP1DBC 的 bit0-bit3 控制消抖时间(CMP1DBC[3:0] * 8Tcpu+2Tcpu)，使能 CMP1DBCEN = 1 数字滤波消抖后，比较器的输出结果信号持续(CMP1DBC[3:0] * 8Tcpu)以上的高电平（低电平）有效信号，才能通过数字滤波消抖信号后输出，基本示意框图如下图所示：



10.4 CMP1 相关功能

10.4.1 CMP1功能描述

图 10-1 显示了 CMP1 的功能框图。CMP1 正端输入可通过配置 CMP1CON0 寄存器的 CMP1PS 位来选择 CMP1+端口、内部电阻分压输出信号 V_{rl} 或者通过 CMP1CON1 寄存器来选择 OP 输出信号 OPOUT；负端输入可通过配置 CMP1NS[2:0]位来选择 CMP1-端口、内部电阻分压输出信号 Vrx 或者 1.3V 电压。当 CMP1 正端输入电压大于负端输入电压时，CMP1 经过数字滤波后输出 1；反之，如果 CMP1 正端输入电压小于负端输入电压，则 CMP1 经过数字滤波后输出 0。（CMP1 滤波时间可选，通过 CMP1DBC 寄存器进行配置）

10.4.2 CMP内部电阻分压输出

CMP1 内部集成了一个电阻分压模块，参考电压为 VDD。可通过配置 CMP1CON1 寄存器的 RBIAS1_H、RBIAS1_L、LVDS1[3:0]等控制位的值来获得不同的电阻分压输出 VR1,根据 RBIAS1_H、RBIAS1_L 这两个位的不同值，VR1 有如下 4 种计算公式：

RBIASm_H	RBIASm_L	VR1 计算公式
0	0	$V_R=\frac{1}{4}*VDD+\frac{n+1}{32}*VDD$
0	1	$V_R=\frac{n+1}{24}*VDD$
1	0	$V_R=\frac{1}{5}*VDD+\frac{n+1}{40}*VDD$
1	1	$V_R=\frac{n+1}{32}*VDD$

注：
n 为 LVDS1<3:0>的值，即 n=0,1,2……14,15。

10.4.3 CMP 监测电源电压

根据图 10-1 的 CMP1 功能框图和 10.4.2 里的公式可知，当 CMP1 的负端选择 1.3V，正端选择内部电阻分压输出 VR1 时，可以通过 CMP1 来检测电源电压，当电源电压低于设定值时 CMP1 输出 0,电源电压高于设定值时 CMP1 输出 1，通过配置 RBIAS1_H、RBIAS1_L、LVDS1<3:0>的值可设定不同的电压监测点，具体如下表：

RBIAS1_H	RBIAS1_L	LVDS1[3:0]	检测值(V)	RBIAS1_H	RBIAS1_L	LVDS1[3:0]	检测值(V)
1	0	0010	4.73	0	1	1010	2.84
0	0	0000	4.62	0	0	0110	2.77
0	1	0110	4.46	1	0	1010	2.74
1	0	0011	4.33	1	0	1011	2.60
0	0	0001	4.16	1	0	1100	2.48
1	0	0100	4.00	0	0	1000	2.45
0	1	0111	3.90	0	1	1100	2.40
0	0	0010	3.78	1	0	1101	2.36
1	0	0101	3.71	0	0	1001	2.31
1	0	0110	3.47	1	0	1110	2.26
1	0	0111	3.25	0	1	1101	2.23
0	0	0100	3.20	0	0	1010	2.19
0	1	1001	3.12	1	0	1111	2.17
1	0	1000	3.06	0	0	1011	2.08
0	0	0101	2.97	0	0	1100	1.98
1	0	1001	2.89				

10.4.4 CMP的中断使用

若要使用 CMP1 的中断功能则可以通过以下配置步骤来开启：

- 配置寄存器 CMP1CON0 的 CMP1PS 位选择正端输入；
- 配置寄存器 CMP1CON0 的 CMP1NS<2:0>位选择负端输入；
- 配置寄存器 CMP1CON1 的 CMP1IM 位选择上升沿或者下降沿触发中断；
- 配置寄存器 CMP1CON0 的 CMP1EN 位使能比较器；
- 延时等待 10us;
- 清零 PIR1 寄存器的 CMP1IF 位；
- PIR1 寄存器的 CMP1IE 位置 1，使能比较器中断；
- INTCON 寄存器的 GIE 位置 1，开启全局中断。

10.4.5 CMP结果输出引脚配置

CMP的输出经过数字滤波后，通过读寄存器CMP1CON0的CMP1OUT位得到当前比较的结果；还可以通过以下的配置步骤输出到CMP1_0引脚：

- 将 TRISC 控制对应的 bit 位来将 CMP1_O 引脚配置为输出口；
- 配置寄存器 CMP1CON0 的 CMP1NV 位来选择正向输出或反向输出；
- 将寄存器 CMP1CON0 的 CMP1OEN 位置 1 来使能 CMP1OUT 输出到 CMP1_O 引脚。

10.5 CMP1 相关寄存器

10.5.1 CMP1CON0寄存器

CMP1 控制寄存器 CMP1CON0

3Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMP1CON0	CMP1EN	CMP1PS	CMP1NS[2:0]			CMP1NV	CMP1OUT	CMP1OEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7] CMP1EN：CMP1 使能位
1 = 使能 CMP1
0 = 禁止 CMP1
Bit [6] CMP1PS：CMP1 正端输入选择位
1 = CMP1+端口电压
0 = VDD 经过内部电阻分压后的电压
Bit [5:3] CMP1NS[2:0]：CMP1 负端输入选择位
000 = CMP1_1-端口 PB3 电压
001 = CMP1_2-端口 PC5 电压
010 = CMP1_3-端口 PA3 电压
011 = CMP1_4-端口 PA4 电压
100 = VDD 经过内部电阻分压后的电压
101 = 1.3V
110 = OPOUT
111 = 保留
Bit [2] CMP1NV：CMP1_O 输出端口取反控制位
1 = CMP1OUT 在 CMP1_O 端口取反输出
0 = CMP1OUT 在 CMP1_O 端口正常输出
Bit [1] CMP1OUT：CMP1 结果位
Bit [0] CMP1OEN：CMP1_O 端口 PC2 输出使能位
1 = 使能 CMP1_O 端口 PC2 输出 CMP1OUT
0 = 禁止 CMP1_O 端口 PC2 输出 CMP1OUT

注：
1、在使用 CMP 时，若 CMP 的正向输入端、负向输入端以及输出端均选择 IO 进行输入输出时，需要手动配置 IO 的模式
2、在使用 CMP 检测电池电压，且同时使用 ADC 采集时，切换 ADC 的内参的时候 1.3V 会存在抖动，可能导致 CMP 标志误置起，需要打开 CMP 滤波进行消抖，至少 1us

10.5.2 CMP1CON1寄存器

CMP1 控制寄存器 CMP1CON1

3Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMP1CON1	CMP1IM	OPAPS	RBIAS1_H	RBIAS1_L	LVDS1[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	0	0	0	0

Bit [7] CMP1IM：CMP1 中断触发边沿选择
1 = CMP1 输出的下降沿触发中断
0 = CMP1 输出的上升沿触发中断
Bit [6] OPAPS：CMP1 正端输入选择
1 = CMP1 正端输入为 OPOUT
0 = CMP1 正端输入由 CMP1PS 决定
Bit [5] RBIAS1_H：具体用法参考 CMP1 的功能框图
Bit [4] RBIAS1_L：具体用法参考 CMP1 的功能框图
Bit [3:0] LVDS1[3:0]：内部分压电阻比选择位

10.5.3 CMP1DBC寄存器

CMP1 消抖时间控制 CMP1DBC

42h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMP1DBC	CMP1DBCEN	-	-	-	CMP1DBC[3:0]			
R/W	R/W	-	-	-	R/W	R/W	R/W	R/W
POR 的值	0	-	-	-	0	0	0	0

Bit [7] CMP1DBCEN：CMP1 消抖使能位
1 = 使能 CMP1 消抖
0 = 禁止 CMP1 消抖
Bit [3:0] CMP1DBC[3:0]:CMP1 消抖时间控制
消抖时间 =CMP1DBC[3:0] * $8T_{cpu}$+2Tcpu

10.5.4 CMPHYC寄存器

CMP 迟滞控制寄存器 CMPHYC

44h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMPHYC	-	-	-	-	-	-	-	CMP1HYS
R/W	-	-	-	-	-	-	-	R/W
POR 的值	-	-	-	-	-	-	-	0

Bit [0] CMP1HYS：CMP1 迟滞使能位
1 = 使能 CMP1 迟滞
0 = 禁止 CMP1 迟滞

11 OP

- 一路运算放大器 OP，可配置成比较器 CMP2
- OP 可配置成 CMP2
- OP 可选放大倍数 1 倍、5 倍、10 倍、20 倍，也可以外部搭建环境配置不同放大倍数
- 比较器 CMP2 可响应中断，可配上升沿或下降沿中断

11.1 OP 的功能框图

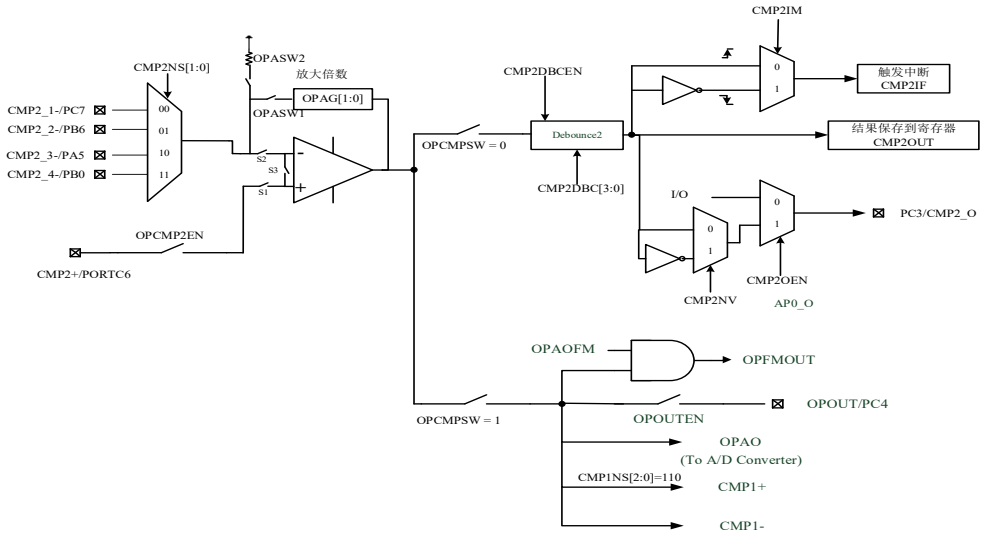


图11-1:CMP2和OP的功能框图

11.2 CMP2 功能特性

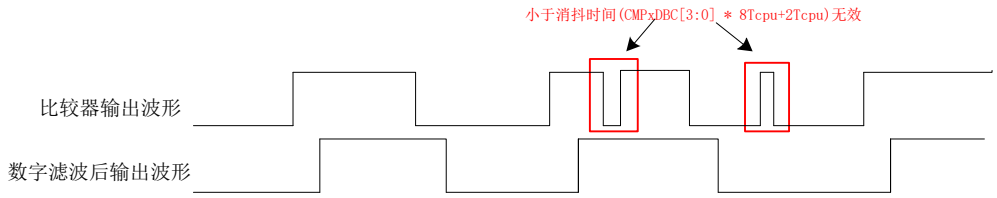
- 比较器失调电压≤±13mv;
- 输入共模电压范围：0V~VDD-1.3V;
- 比较器结果可选上升沿或下降沿触发中断;
- CMP2 结果可选择从 PC3 口输出，且支持取反输出。

11.3 CMP2 消抖

比较器 CMP2 内置的数字滤波器，用于比较器的输出信号进行数字滤波，通过配置 CMP2DBCEN 位控制是否开启数字滤波器，为“1”使能数字滤波，为“0”无滤波。

用户可使用滤波功能过滤系统噪声，比如通过端口传入噪声等，避免因噪声引起比较器 CMP2 系统的误动作。

可配置 CMP2DBC 的 bit0-bit3 控制消抖时间(CMP2DBC[3:0] * 8Tcpu+2Tcpu)，使能 CMP2DBCEN = 1 数字滤波消抖后，比较器的输出结果信号持续(CMP2DBC[3:0] * 8Tcpu)以上的高电平（低电平）有效信号，才能通过数字滤波消抖信号后输出，基本示意框图如下图所示：



11.4 CMP2 相关功能

11.4.1 CMP2功能描述

图 11-1 显示了 CMP2 的功能框图。CMP2 正端输入为 CMP2+端口 PORTC6；负端输入可通过配置 CMP2NS[1:0]位来选择 CMP2-端口。当 CMP2 正端输入电压大于负端输入电压时，CMP2 经过数字滤波后输出 1；反之，如果 CMP2 正端输入电压小于负端输入电压，则 CMP2 经过数字滤波后输出 0。（CMP2 滤波时间可选，通过 CMP2DBC 寄存器进行配置）

11.4.2 CMP2的中断使用

若要使用CMP2的中断功能则可以通过以下配置步骤来开启：

- 配置寄存器 CMP2CON0 的 CMP2NS<2:0>位选择负端输入；
- 配置寄存器 CMP2CON1 的 CMP2IM 位选择上升沿或者下降沿触发中断；
- 配置寄存器 CMP2CON0 的 CMP2EN 位使能比较器；
- 延时等待 10us；
- 清零 PIR1 寄存器的 CMP2IF 位；
- PIR1 寄存器的 CMP2IE 位置 1，使能比较器 2 中断；
- INTCON 寄存器的 GIE 位置 1，开启全局中断。

11.4.3 CMP2结果输出引脚配置

CMP的输出经过数字滤波后，通过读寄存器CMP2CON0的CMP2OUT位得到当前比较的结果；还可以通过以下的配置步骤输出到CMP2_O引脚：

- 将 TRISC 控制对应的 bit 位来将 CMP2_O 引脚配置为输出口；
- 配置寄存器 CMP2CON0 的 CMP2NV 位来选择正向输出或反向输出；
- 将寄存器 CMP2CON0 的 CMP2OEN 位置 1 来使能 CMP2OUT 输出到 CMP2_O 引脚。

11.5 OP 相关寄存器

11.5.1 OPA0C寄存器

OP 控制寄存器 OPA0C

45h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPA0C	-	OPASW2	OPCMPSW	OPOUTEN	-	OPASW1	OPAG[1:0]	
R/W	-	R/W	R/W	R/W	-	R/W	R/W	R/W
POR 的值	-	0	0	0	-	0	0	0

Bit [6] OPASW2：OP 负反馈开关 2
1 = 闭合开关
0 = 断开开关
Bit [5] OPCMPSW：OP/CMP 输出选择位
1 = OP 模式
0 = CMP 模式
Bit [4] OPOUTEN：OP 输出使能控制位
1 = OP 使能输出到 IO-PC4
0 = OP 禁止输出到 IO-PC4
Bit [2] OPASW1：OP 负反馈开关 1
1 = 闭合开关
0 = 断开开关
Bit [1:0] OPAG[1:0]：OP 放大倍数选择
00 = 放大 1 倍
01 = 放大 5 倍
10 = 放大 10 倍
11 = 放大 20 倍

注：
1、OP 模式下(OPCMPSW=1)，CMP2+(PORTC6)作为输入口，配置 OPOUTEN=1，OPOUT(PORTC4)作为放大电压输出口
2、OP 模式下(OPCMPSW=1)，通过 OPAG[1:0]配置 OP 放大倍数 1 倍时，需要配置 OPASW1=1 和 OPASW2=0；配置 OP 放大倍数 5/10/20 倍时，需要配置 OPASW1=1 和 OPASW2=1。
3、使用外部放大倍数电路搭建，配置 OPASW1=0 和 OPASW2=0，避免影响外部放大倍数。通过 CMP2NS[1:0]配置 OP-端口，OP-接电阻 R1 到 GND，配置 OPOUTEN=1，OPOUT(PORTC4)接电阻 R2 到 CMP2NS[1:0]配置 OP-端口。OP±和 OPOUT 端口均无需外接电容，加了电容会影响运放稳定性。放大倍数计算公式为（R1+R2）/R1。
4、若将 OPOUT 信号给到到 CMP1 正端输入上，需要配置 OPAPS=1，且配置 CMP1PS 无效。
5、在使用 OP 时，若 OP 的正向输入端、负向输入端以及输出端均有 IO 进行输入输出时，需要手动配置 IO 的模式选择。

11.5.2 OPAVOS寄存器

OP 失调校准寄存器 OPAVOS

46h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPAVOS	OPAOFM	OPARSP	OPFMOUT	OPAOF[4:0]				
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7] OPAOFM：OPA 工作模式选择
1 = 失调校准模式
0 = 正常工作模式

Bit [6] OPARSP：OPA 输入失调电压参考选择
1 = 输入参考电压 OP+
0 = 输入参考电压 OP-

Bit [5] OPFMOUT：OPA 输入失调校准时，OP 输出状态位（只读）
OPAOFM=1，OPFMOUT 定义 OP 输出状态 内容详见参考失调校准
OPAOFM=0 该位固定位低电平

Bit [4:0] OPAOF[4:0]：OPA 输入失调电压校准位

注：
OPFMOUT 状态位为取反输出之前的状态

失调校准说明：

OPAOFM	OPARSP	S1	S2	S3
0	x	ON	ON	OFF
1	0	OFF	ON	ON
1	1	ON	OFF	ON

输入失调校准流程说明：

注：由于运算放大器输入引脚与 I/O 引脚共用，在输入失调校准前应先将引脚配置为模拟输入模式。

步骤 1：设置 OPAOFM=1 和 OPARSP=1，使运算放大器工作于失调校准模式，S1 和 S3 都 On。为了确保校准后的 VAnOS 尽可能小，校准模式下的输入参考电压应该跟正常模式下的输入直流工作电压相同。

步骤 2：设置 OPAOF[4:0]=00000，读取 OPFMOUT 位。

步骤 3：使 OPAOF[4:0]=OPAOF[4:0]+1，读取 OPFMOUT 位。如果 OPFMOUT 位状态不变，重复步骤 3 直到 OPFMOUT 位状态改变。如果 OPFMOUT 位状态改变，记录此时的 OPAOF[4:0]值为 VAnOS1 然后转到步骤 4。

步骤 4：设置 OPAOF[4:0]=11111，读取 OPFMOUT 位。

步骤 5：使 OPAOF[4:0]=OPAOF[4:0]-1，读取 OPFMOUT 位。如果 OPFMOUT 位状态不变，重复步骤 5 直到 OPFMOUT 位状态改变。如果 OPFMOUT 位状态改变，记录此时的 OPAOF[4:0] 值为 VAnOS2 然后转到步骤 6。

步骤 6：将运算放大器输入失调校准值 VAnOS 存入 OPAOF[4:0] 位中，校准结束。

其中 VAnOS =(VAnOS1+VAnOS2)/2。如果(VAnOS1+VAnOS2)/2 不是整数，舍弃小数。

11.6 CMP2 相关寄存器

11.6.1 CMP2CON0寄存器

CMP2 控制寄存器 CMP2CON0

40h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMP2CON0	OPCMP2EN	CMP2IM	-	CMP2NS[1:0]		CMP2NV	CMP2OUT	CMP2OEN
R/W	R/W	R/W	-	R/W	R/W	R/W	R	R/W
POR 的值	0	0	-	0	0	0	0	0

Bit [7] OPCMP2EN：OPCMP2 使能位
1 = 使能 OPCMP2
0 = 禁止 OPCMP2
Bit [6] CMP2IM：CMP2 中断触发边沿选择
1 = CMP2 输出的下降沿触发中断
0 = CMP2 输出的上升沿触发中断
Bit [4:3] CMP2NS[1:0]：CMP2 负端输入选择位
00 = CMP2_1-端口 PC7 电压
01 = CMP2_2-端口 PB6 电压
10 = CMP2_3-端口 PA5 电压
11 = CMP2_4-端口 PB0 电压
Bit [2] CMP2NV：CMP2_O 输出端口取反控制位
1 = CMP2OUT 在 CMP2_O 端口取反输出
0 = CMP2OUT 在 CMP2_O 端口正常输出
Bit [1] CMP2OUT：CMP2 结果位
Bit [0] CMP2OEN：CMP2_O 端口 PC3 输出使能位
1 = 使能 CMP2_O 端口 PC3 输出 CMP2OUT
0 = 禁止 CMP2_O 端口 PC3 输出 CMP2OUT

> 注：
> 1、在使用 CMP 时，若 CMP 的正向输入端、负向输入端以及输出端均有 IO 进行输入输出时，需要手动配置 IO 的模式选择

11.6.2 CMP2DBC寄存器

CMP2 消抖时间控制 CMP2DBC

43h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMP2DBC	CMP2DBCEN	-	-	-	CMP2DBC[3:0]			
R/W	R/W	-	-	-	R/W	R/W	R/W	R/W
POR 的值	0	-	-	-	0	0	0	0

Bit [7] CMP2DBCEN：CMP2 消抖使能位
1 = 使能 CMP2 消抖
0 = 禁止 CMP2 消抖
Bit[3:0] CMP2DBC[3:0]:CMP2 消抖时间控制
消抖时间 =CMP2DBC[3:0] * $8T_{cpu}$+$2T_{cpu}$

12 PWM01

- PWM01 工作时钟源为 RC 32MHZ，可设定时钟分频比
- 提供周期相同的两路 PWM，PWM0 和 PWM1
- PWM0、PWM1 精度为 12 位
- 可软件实现一组互补带死区输出
- 每一路的 PWM 可取反输出
- 对齐方式可选中心对齐或边沿对齐
- 提供周期溢出中断
- PWM01 可配置为 LED 级联模式，由 PWM0 输出级联信号（PWM1 输出 0）

12.1 输出对齐方式

通过配置寄存器 PWM01CON0 的 TY01 位让 PWM01 处于边沿对齐的计数方式或中心对齐的计数方式下。任何输出对齐方式下，PWM0、PWM1 的输出取反均由控制寄存器 PWM01CON1 的 PWM0DIR、PWM1DIR 位控制。

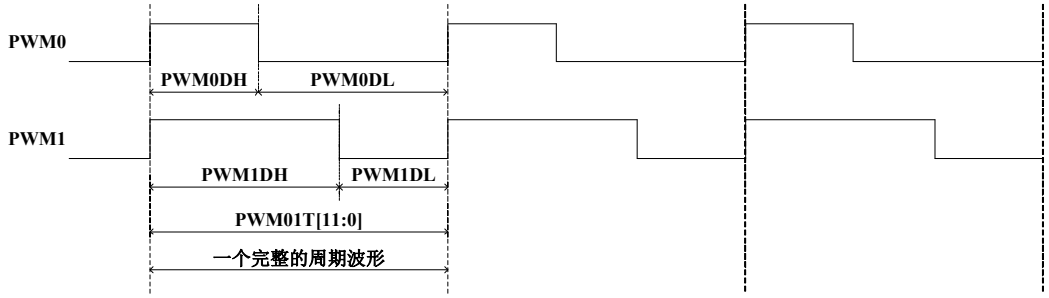
12.1.1 边沿对齐

配置 TY01=0 时，PWM0、PWM1 处于边沿对齐的计数方式下，PWM01 从 0 开始向上计数，直到计数到周期寄存器 PWM01TH 和 PWM01TL 的值，然后立即返回 0，开始下一个周期。

12.1.1.1 正常输出（PWM0DIR=1、PWM1DIR=1）

在边沿对齐模式下（TY01=0），PWM0 和 PWM1 的周期相同，其周期由周期寄存器 PWM01TH 和 PWM01TL 控制；PWM0 的占空比由 PWM0 的占空比寄存器 PWM0DH 和 PWM0DL 控制，PWM1 的占空比由 PWM1 的占空比寄存器 PWM1DH 和 PWM1DL 控制。

对应波形如下：

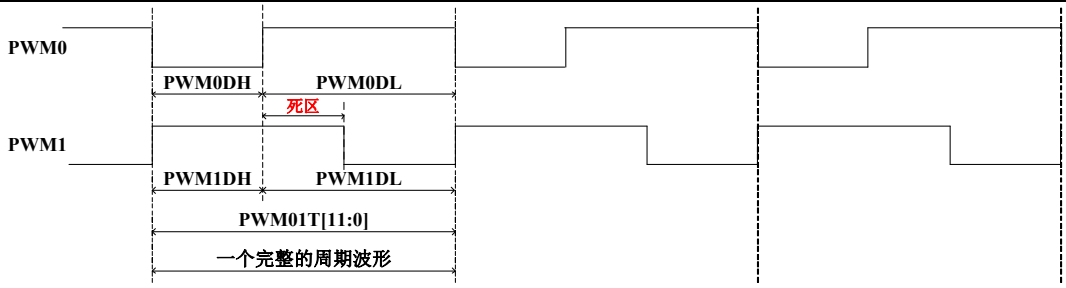


12.1.1.2 互补带死区输出（PWM0DIR=0、PWM1DIR=1）

PWM01 模块可软件配置 PWM01CON1 的 PWM0DIR、PWM1DIR 任意一位配置为 1（输出取反），PWM0 和 PWM1 变为一组互补带死区的 PWM 波形。

在边沿对齐模式下（TY01=0），PWM0 和 PWM1 的周期相同，其周期由周期寄存器 PWM01TH 和 PWM01TL 控制；PWM0 的占空比由 PWM0 的占空比寄存器 PWM0DH 和 PWM0DL 控制，PWM1 的占空比由 PWM1 的占空比寄存器 PWM1DH 和 PWM1DL 控制。先配置 PWM0 占空比，再配置 PWM1DH=PWM0DH、PWM1DL=PWM0DL，PWM0 与 PWM1 互补带死区输出。

对应波形如下：



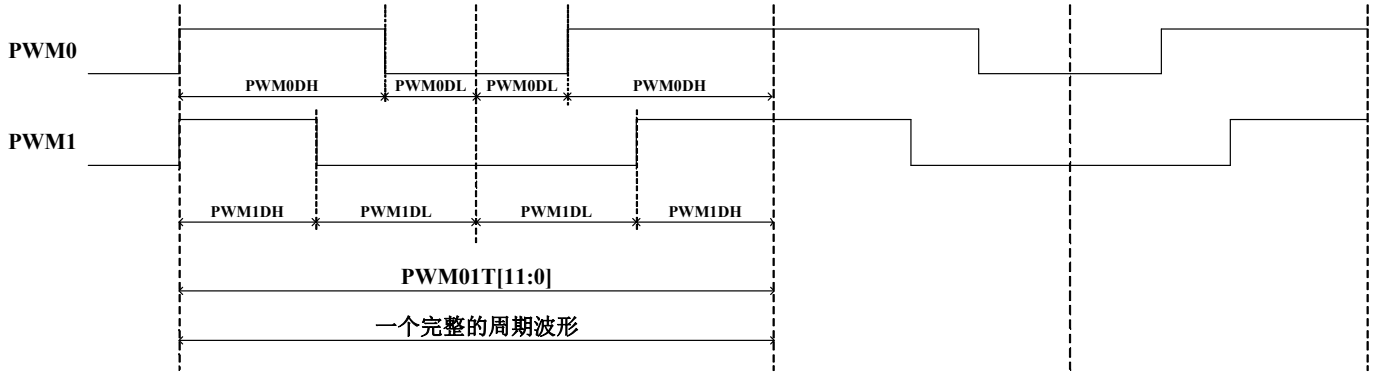
12.1.2 中心对齐

配置 TY01=1 时，PWM0、PWM1 处于中心对齐的计数方式下，PWM01 从 0 开始向上计数，直到计数到周期寄存器 PWM01TH 和 PWM01TL 的值，再开始向下计数，直到计数到 0，然后开始下一个周期。

12.1.2.1 正常输出（PWM0DIR=1、PWM1DIR=1）

在中心对齐模式下（TY01=1），PWM0 和 PWM1 的周期相同，其周期由周期寄存器 PWM01TH 和 PWM01TL 控制（{PWM01TH[11:8], PWM01TL[7:0]}x2）；PWM0 的占空比由 PWM0 的占空比寄存器 PWM0DH 和 PWM0DL 控制（{PWM0DH[11:8], PWM0DL[7:0]}x2），PWM1 的占空比由 PWM1 的占空比寄存器 PWM1DH 和 PWM1DL 控制（{PWM1DH[11:8], PWM1DL[7:0]}x2）。

对应波形如下：

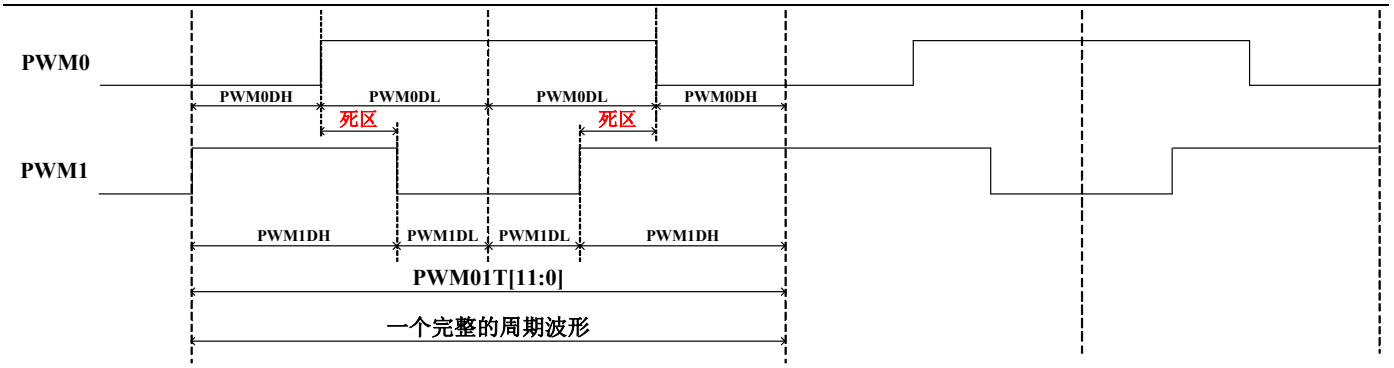


12.1.2.2 互补带死区输出（PWM0DIR=0、PWM1DIR=1）

PWM01 模块可软件配置 PWM01CON1 的 PWM0DIR、PWM1DIR 任意一位配置为 1（输出取反），PWM0 和 PWM1 变为一组互补带死区的 PWM 波形。

在中心对齐模式下（TY01=1），PWM0 和 PWM1 的周期相同，其周期由周期寄存器 PWM01TH 和 PWM01TL 控制（{PWM01TH[11:8], PWM01TL[7:0]}x2）；PWM0 的占空比由 PWM0 的占空比寄存器 PWM0DH 和 PWM0DL 控制（{PWM0DH[11:8], PWM0DL[7:0]}x2），PWM1 的占空比由 PWM1 的占空比寄存器 PWM1DH 和 PWM1DL 控制（{PWM1DH[11:8], PWM1DL[7:0]}x2）。先配置 PWM0 占空比，再配置 PWM1DH=PWM0DH、PWM1DL=PWM0DL，PWM0 与 PWM1 互补带死区输出。

对应波形如下：


PWM0
PWM0DH
PWM0DL
PWM0DL
PWM0DH
死区
死区
PWM1
PWM1DH
PWM1DL
PWM1DL
PWM1DH
PWM01T[11:0]
一个完整的周期波形

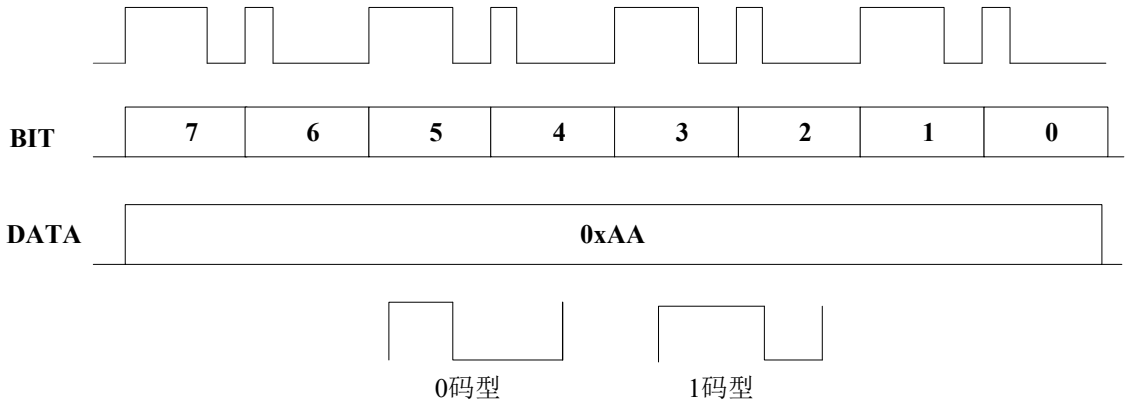
12.2 LED 级联控制

PWM01 通过寄存器 PWM01CON1 的 LED0EN 位置 1，配置成 1 路 LED 级联控制，可驱动幻彩 LED。

此 LED 级联模式下（LED0EN=1），归零码的周期由 PWM01 的周期寄存器 PWM01TH 和 PWM01TL 控制；归零码‘0’码型的高电平时间由 PWM0 的占空比寄存器 PWM0DH 和 PWM0DL 控制，归零码‘1’码型的高电平时间由 PWM1 的占空比寄存器 PWM1DH 和 PWM1DL 控制，需要发送的归零码数据存放在 LED0DATA 寄存器中。

配置 LED0CEN=1，将存放在 LED0DATA 寄存器中的数据，通过 PWM0 端口输出。数据发送结束后，LED0CEN 自动复位清零，重新配置 LED0CEN=1 开始下一个数据发送。

通过 PWM0 端口输出级联控制时序。级联 LED 驱动时序说明图如下：



注：
1、PWM01 在使用 LED 级联模式时，LED 级联数据发送可通过 PWM0 端口输出（PWM0_MAP）
2、PWM01 在使用 LED 级联模式时，PWM01 输出类型默认为边沿对齐的计数方式，选择中心沿对齐无效
3、PWM01 在使用 LED 级联模式时，PWM 输出取反控制位默认为 0，取反输出无效
4、PWM01 在使用 LED 级联模式时，PWM0 输出控制位 PWM0OEN 需要配置为 1。
5、PWM01 在使用 LED 级联模式时，PWM1 输出低电平，PWM1DIR 取反控制无效。
6、PWM01 在使用 LED 级联模式时，级联数据是高位先发

12.3 PWM01 相关寄存器说明

12.3.1 PWM01CON0寄存器

PWM01 控制寄存器 PWM01CON0

47h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM01CON0	-	CK01[2:0]			TY01	PWM1OEN	PWM0OEN	PWM01EN
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 [6:4] CK01[2:0]：PWM01 时钟分频

000 = RC32M/1
001 = RC32M/2
010 = RC32M/4
011 = RC32M/8
100 = RC32M/16
101 = RC32M/32
110 = RC32M/64
111 = RC32M/128

Bit [3] TY01:PWM0 和 PWM1 输出类型选择位

1 = PWM0 和 PWM1 中心沿对齐
0 = PWM0 和 PWM1 边沿对齐

Bit [2] PWM1OEN：PWM1 输出使能位

1 = 使能 PWM1 输出，对应端口输出 PWM1 信号
0 = 禁止 PWM1 输出，对应端口作为 I/O 口

Bit [1] PWM0OEN：PWM0 输出使能位

1 = 使能 PWM0 输出，对应端口输出 PWM0 信号
0 = 禁止 PWM0 输出，对应端口作为 I/O 口

Bit [0] PWM01EN：PWM01 使能位

1 = 使能 PWM01
0 = 禁止 PWM01

注：
PWM 模块在占空比配置为 0 时输出高电平，如需要输出低电平，需要软件配置配置取反输出，切换占空比时需要软件切换为正常输出

12.3.2 PWM01CON1寄存器

PWM01 控制寄存器 PWM01CON1

48h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM01CON1	-	-	-	-	PWM1DIR	PWM0DIR	LED0CEN	LED0EN
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3] PWM1DIR：PWM1 输出取反控制位
1 = PWM1 取反输出
0 = PWM1 正常输出
Bit [2] PWM0DIR：PWM0 输出取反控制位
1 = PWM0 取反输出
0 = PWM0 正常输出
Bit [1] LED0CEN:LED0 级联数据发送控制位
1 = 使能 LED0 级联发送数据，发送结束后自动清零
0 = 禁止发送或空闲
Bit [0] LED0EN:LED0 级联控制使能位
1 = 使能 LED0 级联功能
0 = 禁止

注：
1、PWM 禁止输出(PWMxOEN(x = 0/1) = 0)时，对应 IO 口输出 IO 数据寄存器的值
2、在使用 PWM 级联功能时，PWM 中断标志位不置 1

12.3.3 PWM01TH、PWM01TL寄存器

PWM0~PWM1 周期高 4 位寄存器 PWM01TH

49h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM01TH	-	-	-	-	PWM01TH[11:8]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] PWM01TH[11:8]：PWM0-PWM1 周期高 4 位

PWM0~PWM1 周期低 8 位寄存器 PWM01TL

4Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM01TL	PWM01TL[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

Bit [7:0] PWM01TL[7:0]：PWM0-PWM1 周期低 8 位

边沿对齐：PWM 周期=(PWM01T[11:0]+1)*T_{RC32M}*(RC32M 分频)
中心沿对齐：PWM 周期=(PWM01T[11:0]+1)*T_{RC32M}*(RC32M 分频)*2
当 PWM 周期计数器等于 PWM01T[11:0]时，在下一个递增计数周期中会发生以下事件：

- PWM 周期计数器被清零；
- PWMx 引脚被置 1；
- PWM 新周期值被锁存；
- PWM 新占空比值被锁存；
- 产生 PWM 中断标志位；

12.3.4 PWM0DH、PWM0DL寄存器

PWM0 占空比高 4 位寄存器 PWM0DH

4Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DH	-	-	-	-	PWM0DH[11:8]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] PWM0DH[11:8]：PWM0 占空比高 4 位

PWM0 占空比低 8 位寄存器 PWM0DL

4Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DL	PWM0DL[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

Bit [7:0] PWM0DL[7:0]：PWM0 占空比低 8 位

12.3.5 PWM1DH、PWM1DL寄存器

PWM1 占空比高 4 位寄存器 PWM1DH

4Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1DH	-	-	-	-	PWM1DH[11:8]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] PWM1DH[11:8]：PWM1 占空比高 4 位

PWM1 占空比低 8 位寄存器 PWM1DL

4Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1DL	PWM1DL[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

Bit [7:0] PWM1DL[7:0]：PWM1 占空比低 8 位

脉冲宽度计算公式：x = 0/1

边沿对齐：脉冲宽度=(PWMxD[11:0]+1)*T_{RC32M}*(RC32M 分频)

中心沿对齐：脉冲宽度=(PWMxD[11:0]+1)*T_{RC32M}*(RC32M 分频)*2

PWM 占空比计算公式：

$$占空比=\frac{PWM01D[11:0]+1}{PWM01T[11:0]+1}$$

注：

写入 PWM0DH[11:8]/PWM1DH[11:8]并不能立即生效，需有写入 PWM0DL/PWM1DL 操作后才能生效。

12.3.6 LED0DATA寄存器

LED 级联数据寄存器 LED0DATA

4Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LED0DATA	LED0DATA[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

Bit [7:0] LED0DATA[7:0]：LED0 级联数据

13 PWM23

- PWM01 工作时钟源为 RC32MHZ，可设定时钟分频比
- 提供周期相同的两路 PWM，PWM2 和 PWM3
- PWM2、PWM3 精度为 12 位
- 可软件实现一组互补带死区输出
- 每一路的 PWM 可取反输出
- 对齐方式可选中心对齐或边沿对齐
- 提供周期溢出中断
- PWM23 可配置为 LED 级联模式，由 PWM2 输出级联信号（PWM3 输出 0）

13.1 输出对齐方式

通过配置寄存器 PWM23CON0 的 TY23 位让 PWM23 处于边沿对齐的计数方式或中心对齐的计数方式下。任何输出对齐方式下，PWM2、PWM3 的输出取反均由控制寄存器 PWM23CON1 的 PWM2DIR、PWM3DIR 位控制。

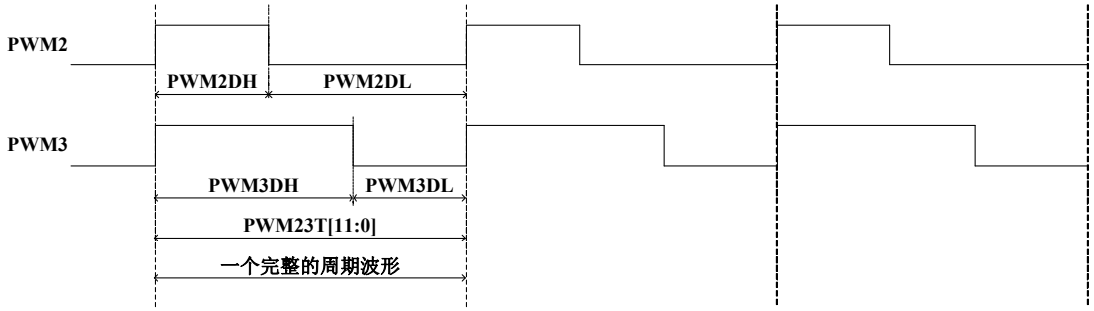
13.1.1 边沿对齐

配置 TY23=0 时，PWM2、PWM3 处于边沿对齐的计数方式下，PWM23 从 0 开始向上计数，直到计数到周期寄存器 PWM23TH 和 PWM23TL 的值，然后立即返回 0，开始下一个周期。

13.1.1.1 正常输出（PWM2DIR=1、PWM3DIR=1）

在边沿对齐模式下（TY23=0），PWM2 和 PWM3 的周期相同，其周期由周期寄存器 PWM23TH 和 PWM23TL 控制；PWM2 的占空比由 PWM2 的占空比寄存器 PWM2DH 和 PWM2DL 控制，PWM3 的占空比由 PWM3 的占空比寄存器 PWM3DH 和 PWM3DL 控制。

对应波形如下：

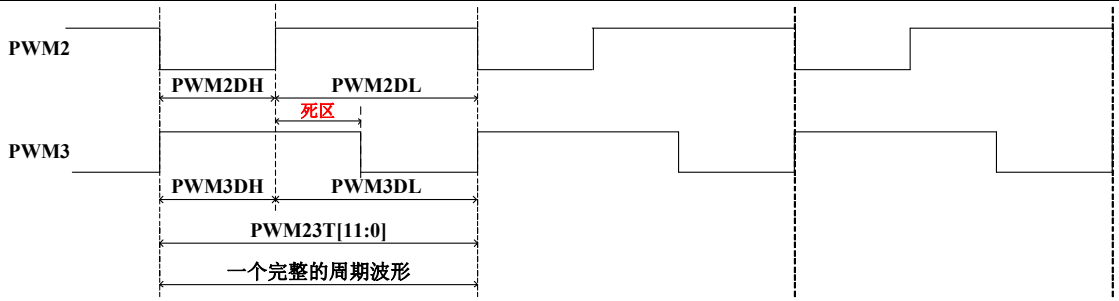


13.1.1.2 互补带死区输出（PWM2DIR=0，PWM3DIR=1）

PWM23 模块可软件配置 PWM23CON1 中的 PWM2DIR、PWM3DIR 任意一位配置为 1（输出取反），PWM2 和 PWM3 变为一组互补带死区的 PWM 波形。

在边沿对齐模式下（TY23=0），PWM2 和 PWM3 的周期相同，其周期由周期寄存器 PWM23TH 和 PWM23TL 控制；PWM2 的占空比由 PWM2 的占空比寄存器 PWM2DH 和 PWM2DL 控制，PWM3 的占空比由 PWM3 的占空比寄存器 PWM3DH 和 PWM3DL 控制。先配置 PWM2 占空比，再配置 PWM3DH=PWM2DH、PWM3DL=PWM2DL，PWM2 与 PWM3 互补带死区输出。

对应波形如下：



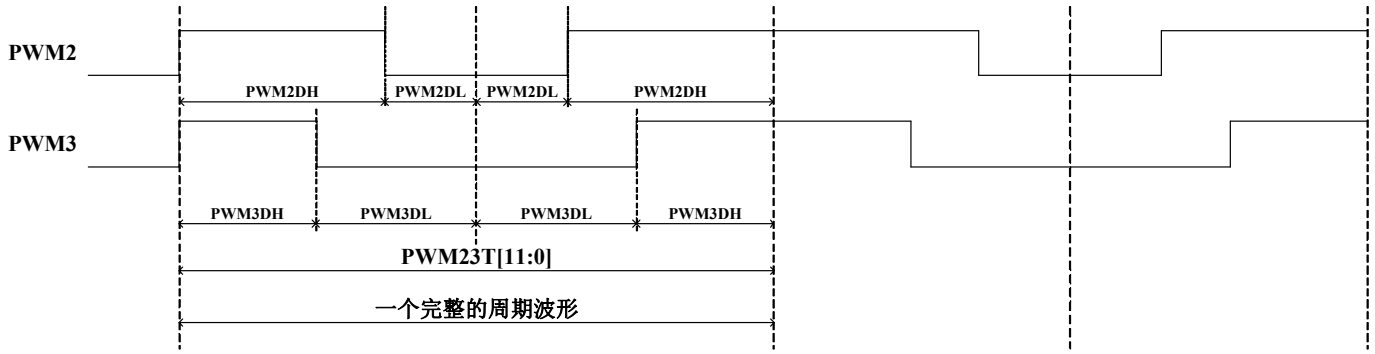
13.1.2 中心对齐

配置 TY23=1 时，PWM2、PWM3 处于中心对齐的计数方式下，PWM23 从 0 开始向上计数，直到计数到周期寄存器 PWM23TH 和 PWM23TL 的值，再开始向下计数，直到计数到 0，然后开始下一个周期。

13.1.2.1 正常输出（PWM2DIR = 1、PWM3DIR = 1）

在中心对齐模式下（TY23=1），PWM2 和 PWM3 的周期相同，其周期由周期寄存器 PWM23TH 和 PWM23TL 控制（{PWM23TH[11:8], PWM23TL[7:0]}x2）；PWM2 的占空比由 PWM2 的占空比寄存器 PWM2DH 和 PWM2DL 控制（{PWM2DH[11:8], PWM2DL[7:0]}x2），PWM3 的占空比由 PWM3 的占空比寄存器 PWM3DH 和 PWM3DL 控制（{PWM3DH[11:8], PWM3DL[7:0]}x2）。

对应波形如下：

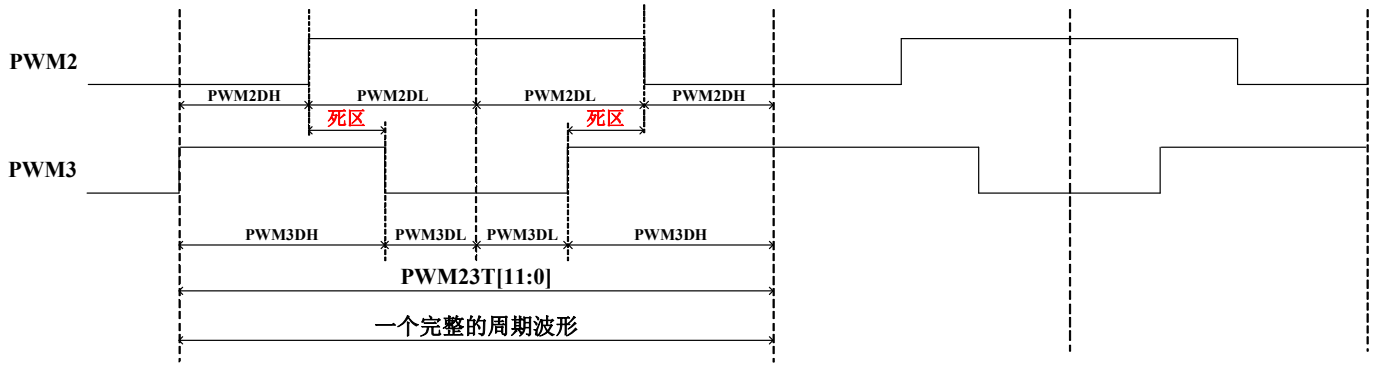


13.1.2.2 互补带死区输出（PWM2DIR=0，PWM3DIR=1）

PWM23 模块可软件配置 PWM23CON1 的 PWM2DIR、PWM3DIR 任意一位配置为 1（输出取反），PWM2 和 PWM3 变为一组互补带死区的 PWM 波形。

在中心对齐模式下（TY23=1），PWM2 和 PWM3 的周期相同，其周期由周期寄存器 PWM23TH 和 PWM23TL 控制（{PWM23TH[11:8], PWM23TL[7:0]}x2）；PWM2 的占空比由 PWM2 的占空比寄存器 PWM2DH 和 PWM2DL 控制（{PWM2DH[11:8], PWM2DL[7:0]}x2），PWM3 的占空比由 PWM3 的占空比寄存器 PWM3DH 和 PWM3DL 控制（{PWM3DH[11:8], PWM3DL[7:0]}x2）。先配置 PWM2 占空比，再配置 PWM3DH=PWM2DH、PWM3DL=PWM2DL，PWM2 与 PWM3 互补带死区输出。

对应波形如下：



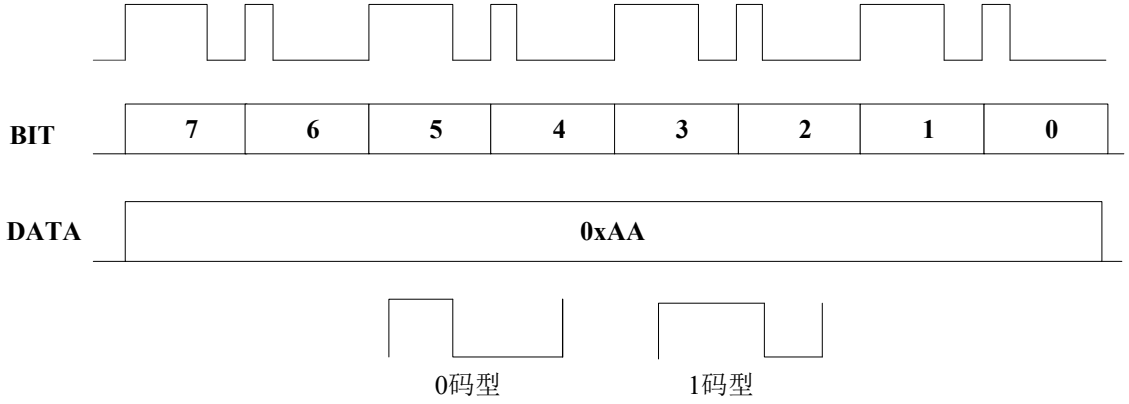
13.2LED 级联控制

PWM23 通过寄存器 PWM23CON1 的 LED1EN 位置 1，配置成 1 路 LED 级联控制，可驱动幻彩 LED。

此 LED 级联模式下（LED1EN=1），归零码的周期由 PWM23 的周期寄存器 PWM23TH 和 PWM23TL 控制；归零码‘0’码型的高电平时间由 PWM2 的占空比寄存器 PWM2DH 和 PWM2DL 控制，归零码‘1’码型的高电平时间由 PWM3 的占空比寄存器 PWM3DH 和 PWM3DL 控制，需要发送的归零码数据存放在 LED1DATA 寄存器中。

配置 LED1CEN=1，将存放在 LED1DATA 寄存器中的数据，通过 PWM2 端口输出。数据发送结束后，LED1CEN 自动复位清零，重新置位 LED1CEN 开始下一个数据发送。

通过 PWM2 端口输出级联控制时序。级联 LED 驱动时序说明图如下：



注：
1、PWM23 在使用 LED 级联模式时，LED 级联数据发送可通过 PWM2 端口输出（PWM2_MAP）
2、PWM23 在使用 LED 级联模式时，PWM23 输出类型默认为边沿对齐的计数方式，选择中心沿对齐无效
3、PWM23 在使用 LED 级联模式时，PWM 输出取反控制位默认为 0，取反输出无效
4、PWM23 在使用 LED 级联模式时，PWM2 输出控制位 PWM2OEN 需要配置为 1。
5、PWM23 在使用 LED 级联模式时，PWM3 输出低电平，PWM3DIR 取反控制位无效。
6、PWM23 在使用 LED 级联模式时，级联数据是高位先发

13.3 PWM23 相关寄存器说明

13.3.1 PWM23CON0寄存器

PWM23 控制寄存器 PWM23CON

50h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM23CON0	-	CK23[2:0]			TY23	PWM3OEN	PWM2OEN	PWM23EN
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 [6:4] CK23[2:0]：PWM23 时钟分频

000 = RC32M/1
001 = RC32M/2
010 = RC32M/4
011 = RC32M/8
100 = RC32M/16
101 = RC32M/32
110 = RC32M/64
111 = RC32M/128

Bit [3] TY23：PWM2 和 PWM3 输出类型选择位

1 = PWM2 和 PWM3 中心沿对齐
0 = PWM2 和 PWM3 边沿对齐

Bit [2] PWM3OEN：PWM3 输出使能位

1 = 使能 PWM3 输出，对应端口输出 PWM3 信号
0 = 禁止 PWM3 输出，对应端口作为 I/O 口

Bit [1] PWM2OEN：PWM2 输出使能位

1 = 使能 PWM2 输出，对应端口输出 PWM2 信号
0 = 禁止 PWM2 输出，对应端口作为 I/O 口

Bit [0] PWM23EN：PWM23 使能位

1 = 使能 PWM23
0 = 禁止 PWM23

注：
PWM 模块在占空比配置为 0 时输出高电平，如需要输出低电平，需要软件配置配置取反输出，切换占空比时需要软件切换为正常输出

13.3.2 PWM23CON1寄存器

PWM23 控制寄存器 PWM23CON1

51h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM23CON1	-	-	-	-	PWM3DIR	PWM2DIR	LED1CEN	LED1EN
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3] PWM3DIR：PWM3 输出取反控制位
1 = PWM3 取反输出
0 = PWM3 正常输出
Bit [2] PWM2DIR：PWM2 输出取反控制位
1 = PWM2 取反输出
0 = PWM2 正常输出
Bit [1] LED1CEN:LED1 级联数据发送控制位
1 = 使能 LED1 级联发送数据，发送结束后自动清零
0 = 禁止发送或空闲
Bit [0] LED1EN:LED1 级联控制使能位
1 = 使能 LED1 级联功能
0 = 禁止

注：
1、PWM 禁止输出(PWMxOEN(x = 0/1) = 0)时，对应 IO 口输出 IO 数据寄存器的值
2、在使用 PWM 级联功能时，PWM 中断标志位不置 1

13.3.3 PWM23TH、PWM23TL寄存器

PWM2~PWM3 周期高 4 位寄存器 PWM23TH

52h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM23TH	-	-	-	-	PWM23TH[11:8]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] PWM23TH[11:8]：PWM2-PWM3 周期高 4 位

PWM2~PWM3 周期低 8 位寄存器 PWM23TL

53h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM23TL	PWM23TL[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

Bit [7:0] PWM23TL[7:0]：PWM2-PWM3 周期低 8 位

边沿对齐：PWM 周期=(PWM23T[11:0]+1)*T_{RC32M}*(RC32M 分频)
中心沿对齐：PWM 周期=(PWM23T[11:0]+1)*T_{RC32M}*(RC32M 分频)*2
当 PWM 周期计数器等于 PWM23T[11:0]时，在下一个递增计数周期中会发生以下事件：
- PWM 周期计数器被清零；
- PWMx 引脚被置 1；
- PWM 新周期值被锁存；
- PWM 新占空比值被锁存；
- 产生 PWM 中断标志位；

13.3.4 PWM2DH、PWM2DL寄存器

PWM2 占空比高 4 位寄存器 PWM2DH

54h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2DH	-	-	-	-	PWM2DH[11:8]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] PWM2DH[11:8]：PWM2 占空比高 4 位

PWM2 占空比低 8 位寄存器 PWM2DL

55h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2DL	PWM2DL[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

Bit [7:0] PWM2DL[7:0]：PWM2 占空比低 8 位

13.3.5 PWM3DH、PWM3DL寄存器

PWM3 占空比高 4 位寄存器 PWM3DH

56h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM3DH	-	-	-	-	PWM3DH[11:8]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] PWM3DH[11:8]：PWM3 占空比高 4 位

PWM3 占空比低 8 位寄存器 PWM3DL

57h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM3DL	PWM3DL[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

Bit [7:0] PWM3DL[7:0]：PWM3 占空比低 8 位

脉冲宽度计算公式：x = 2/3

边沿对齐：脉冲宽度=(PWMxD[11:0]+1)*T_{RC32M}*(RC32M 分频)

中心沿对齐：脉冲宽度=(PWMxD[11:0]+1)*T_{RC32M}*(RC32M 分频)*2

PWM 占空比计算公式：

$$占空比=\frac{PWM23D[11:0]+1}{PWM23T[11:0]+1}$$

注：

写入 PWM2DH[11:8]/PWM3DH[11:8]并不能立即生效，需有写入 PWM2DL/PWM3DL 操作后才能生效。

13.3.6 LED1DATA寄存器

LED 级联数据寄存器 LED1DATA

58h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LED1DATA	LED1DATA[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

Bit [7:0] LED1DATA[7:0]：LED1 级联数据

14 PWM4

- PWM4 精度为 12 位
- PWM4 工作时钟源为 RC 32MHZ，可设定时钟分频比
- 提供每个 PWM 周期溢出中断
- 输出可选正常输出和取反输出

14.1 PWM4 相关寄存器说明

14.1.1 PWM4控制寄存器PWM4CON

PWM4 控制寄存器 PWM4CON

59h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4CON	-	CK4[2:0]			-	PWM4DIR	PWM4OEN	PWM4EN
R/W	-	R/W	R/W	R/W	-	R/W	R/W	R/W
POR 的值	-	0	0	0	-	0	0	0

Bit [6:4] CK4[2:0]：PWM4 时钟分频

000 = RC32M/1
001 = RC32M/2
010 = RC32M/4
011 = RC32M/8
100 = RC32M/16
101 = RC32M/32
110 = RC32M/64
111 = RC32M/128

Bit [2] PWM4DIR：PWM4 输出取反控制位

1 = PWM4 取反输出
0 = PWM4 正常输出

Bit [1] PWM4OEN：PWM4 输出使能位

1 = 使能 PWM4 输出，对应端口输出 PWM0 信号
0 = 禁止 PWM4 输出，对应端口作为 I/O 口

Bit [0] PWM4EN：PWM4 使能位

1 = 使能 PWM4
0 = 禁止 PWM4

注：
1、PWM4OEN 输出使能位需要在 PWM4EN 使能位置 1 时，对应 IO 才能输出 PWM 波形
2、PWM 禁止输出(PWM4OEN = 0)时，对应 IO 口输出 IO 数据寄存器的值
3、PWM 模块在占空比配置为 0 时输出高电平，如需要输出低电平，需要软件配置配置取反输出，切换占空比时需要软件切换为正常输出

14.1.2 PWM4TH、PWM4TL周期寄存器

PWM4 周期高 4 位寄存器 PWM4TH

5Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4TH	-	-	-	-	PWM4TH[11:8]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] PWM4TH[11:8]：PWM4 周期高 4 位

PWM4 周期低位寄存器 PWM4TL

5Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4TL	PWM4TL[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

Bit [7:0] PWM4TL[7:0]：PWM4 周期低 8 位

PWM4 周期=(PWM4T[11:0]+1)*T_{RC32M}*(RC32M 分频)

当 PWM 周期计数器等于 PWM4T[11:0]时，在下一个递增计数周期中会发生以下事件：

- PWM 周期计数器被清零；
- PWM4 引脚被置 1；
- PWM 新周期值被锁存；
- PWM 新占空比值被锁存；
- 产生 PWM 中断标志位；

14.1.3 PWM4DH、PWM4DL占空比寄存器

PWM4 占空比高 4 位寄存器 PWM4DH

5Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4DH	-	-	-	-	PWM4DH[11:8]			
R/W	-	-	-	-	R/W	R/W	R/W	R/W
POR 的值	-	-	-	-	0	0	0	0

Bit [3:0] PWM4DH[11:8]：PWM4 占空比高 4 位

PWM4 占空比低位寄存器 PWM4DL

5Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4DL	PWM4DL[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

Bit [7:0] PWM4DL[7:0]：PWM4 占空比低 8 位

脉冲宽度计算公式：

脉冲宽度=(PWM4D[11:0]+1)*T_{RC32M}*(RC32M 分频)

PWM 占空比计算公式：

$$占空比=\frac{PWM4D[11:0]+1}{PWM4T[11:0]+1}$$

15 通用串行总线

- 自带波特率发生器
- 波特率发生器为一个 16 位向上的计数器
- 两种工作方式可选
- 增加帧错误，接收溢出及写冲突检测

15.1 UART 工作方式

15.1.1 方式0：8位UART，可变波特率，异步全双工

方式 0 提供 10 位全双工异步通信，10 位由一个起始位（逻辑 0），8 个数据位（低位在前）和一个停止位（逻辑 1）组成。在接收时，这 8 个数据位存储在 SBUF 中而停止位储存在 RB8 中。方式 0 中的波特率来源于波特率发生器。功能块框图如下图所示。

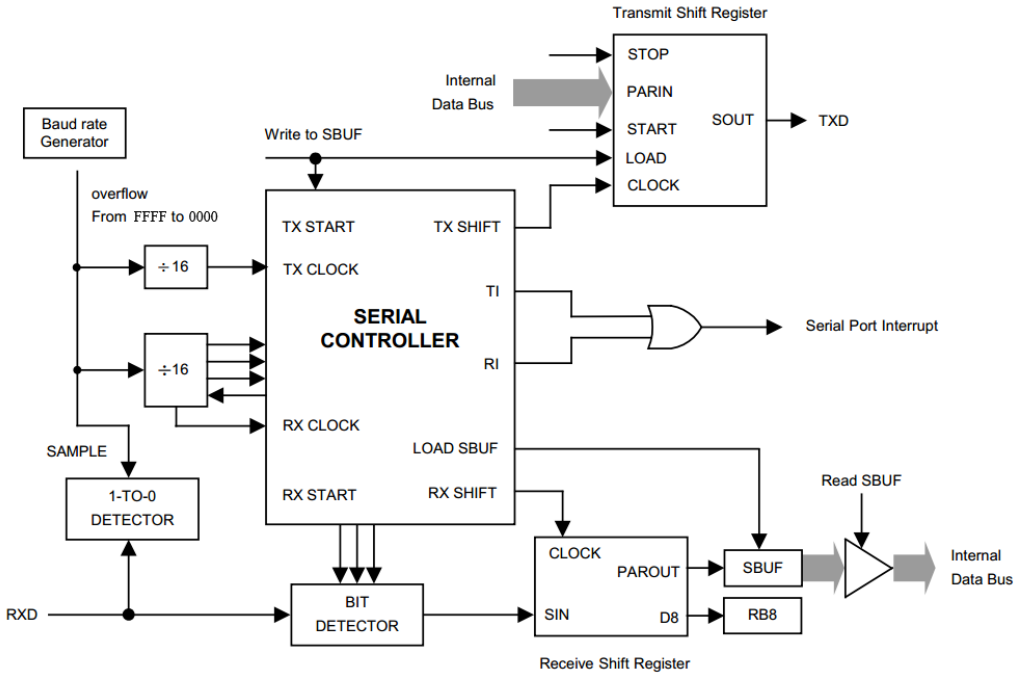


Figure 15-1 Receive Shift Register

任何将 SBUF 作为目标寄存器的写操作都会启动发送，实际上发送是从 16 分频计数器中的下一次跳变之后的系统时钟开始的，因此位时间与 16 分频计数器是同步的，与对 SBUF 的写操作不同步。起始位首先在 TXD 引脚上移出，然后是 8 位数据位。在发送移位寄存器中的所有 8 位数据都发送完后，停止位在 TXD 引脚上移出，在停止位发出的同时 TI 标志置位。

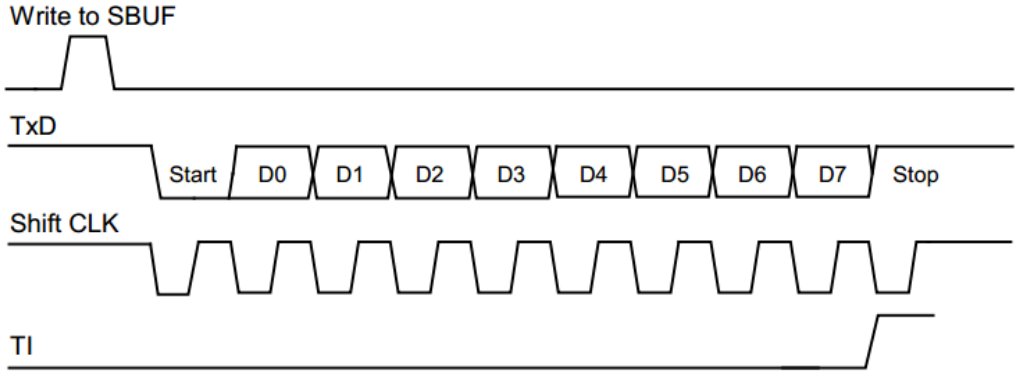


Figure 15-2 Send Timing of Mode 1

只有 REN 置 1 时才允许接收。当 RXD 引脚检测到下降沿时串行口开始接收串行数据。为此，CPU 对 RXD 不断采样，采样速率为波特率的 16 倍。当检测下降沿时，16 分频计数器立即复位，这有助于 16 分频计数器与 RXD 引脚上的串行数据位同步。16 分频计数器把每一位的时间分为 16 个状态，在第 7、8、9 状态时，位检测器对 RXD 端的电平进行采样。为抑制噪声，在这 3 个状态采样中至少有 2 次采样值一致数据才被接收。如果所接收的第一位不是 0，说明这位不是一帧数据的起始位，该位被忽略，接收电路被复位，等待 RXD 引脚上另一个下降沿的到来。若起始位有效，则移入移位寄存器，并接着移入其它位到移位寄存器。8 个数据位和 1 个停止位（包含错误的停止位，详细见寄存器 SM2 位说明）移入之后，移位寄存器的内容和停止位(包含错误的停止位)被分别装入 SBUF 和 RB8 中，RI 置 1，但必须满足下列条件：

(1) RI = 0

(2) SM2 = 0 或者接收的停止位= 1

如果这些条件被满足，那么停止位（包含错误的停止位）装入 RB8，8 个数据位装入 SBUF，RI 被置位。否则接收的帧会丢失。这时，接收器将重新去探测 RXD 端是否另一个下降沿。用户必须用软件清零 RI，然后才能再次接收。

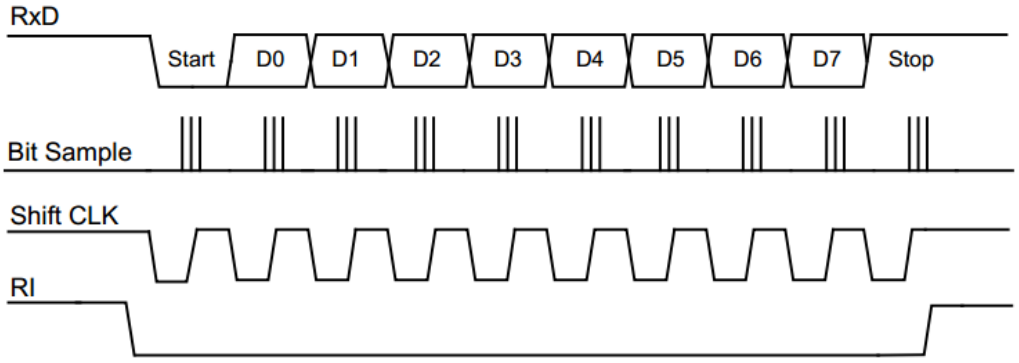


Figure 15-3 Receive Timing of Mode 0

15.1.2 方式1：9位UART，可变波特率，异步全双工

这个方式使用异步全双工通信中的 11 位。一帧由一个起始位（逻辑 0），8 个数据位（低位在前），一个可编程的第 9 数据位和一个停止位（逻辑 1）组成。方式 1 支持多机通信，在数据传送时，第 9 数据位（TB8 位）可以写 0 或 1，例如，可写入 PSW 中的奇偶位 P，或用作多机通信中的数据/地址标志位。当接收到数据时，第 9 数据位移入 RB8 而停止位不保存。

任何将 SBUF 作为目标寄存器的写操作都会启动发送，同时也将 TB8 载入到发送移位寄存器的第 9 位中。实际上发送是从 16 分频计数器中的下一次跳变之后的系统时钟开始的，因此位时间与 16 分频计数器是同步的，与对 SBUF 的写操作不同步。起始位首先在 TXD 引脚上移出，然后是 9 位数据。在发送转换寄存器中的所有 9 位数据都发送完后，停止位在 TXD 引脚上移出，在停止位开始发送时 TI 标志置位。

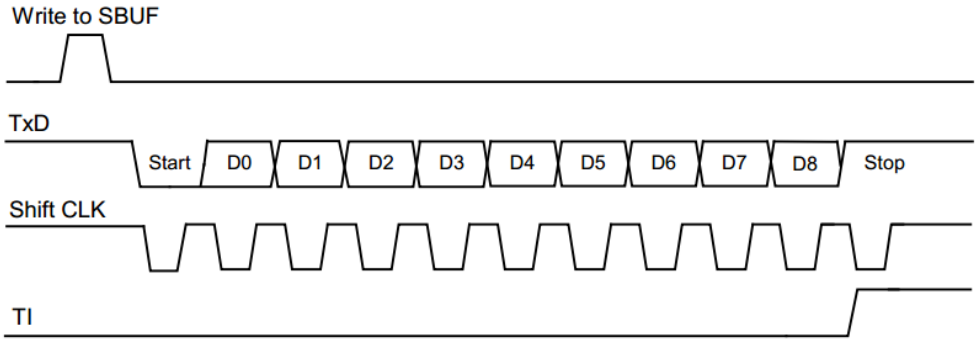


Figure 15-4 Send Timing of Mode 2

只有 REN 置位时才允许接收。当 RXD 引脚检测到下降沿时串行口开始接收串行数据。为此，CPU 对 RXD 不断采样，采样速率为波特率的 16 倍。当检测下降沿时，16 分频计数器立即复位。这有助于 16 分频计数器与 RXD 引脚上的串行数据位同步。16 分频计数器把每一位的时间分为 16 个状态，在第 7、8、9 状态时，位检测器对 RXD 端的电平进行采样。为抑制噪声，在这 3 个状态采样中至少有 2 次采样值一致数据才被接收。如果所接收的第一位不是 0，说明这位不是一帧数据的起始位，该位被忽略，接收电路被复位，等待 RXD 引脚上另一个下降沿的到来。若起始位有效，则移入移位寄存器，并接着移入其它位到移位寄存器。9 个数据位移入之后，移位寄存器的内容被分别装入 SBUF 和 RB8 中，但必须满足下列条件：

(1) RI = 0

(2) SM2 = 0 或者接收的第 9 位= 1

如果这些条件被满足，那么第 9 位移入 RB8，8 位数据移入 SBUF。但还需要检测停止位，只有停止位为 1，才能置位 RI，如果停止位为 0，则 RI 不会置位。

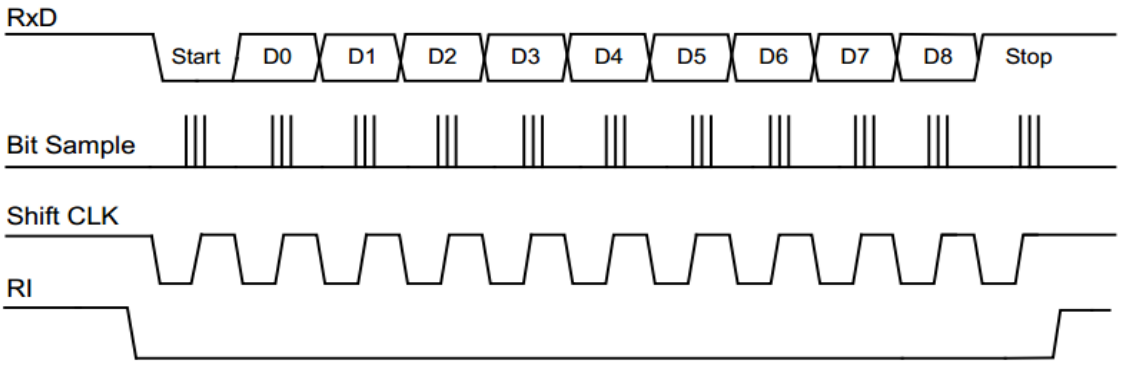


Figure 15-5 Receive Timing of Mode 2

15.2 波特率

UART 自带一个波特率发生器，它实质上就是一个 16 位递增计数器。

UART 的时钟由 UARTCLKSEL 决定，在方式 0 和方式 1 中，波特率公式如下：

$$Baud = \frac{F_{cpu}}{16 \times (65536\text{-}SBRT)}，SBRT = [SBRTH : SBRTL]$$

15.3 帧出错检测

3 个错误标志位被置位后，只能通过软件清零，尽管后续接收的帧没有任何错误也不会自动清零。

15.3.1 发送冲突

如果在一个发送正在进行时，用户软件写数据到 SBUF 寄存器时，发送冲突位（TXCOL 位）置 1。如果发生了冲突，新数据会被忽略，不能被写入发送缓冲器（即不影响传送）。

15.3.2 接收溢出

RI 置 1，接收缓冲器中的数据未被读取，RI 被清 0，又开始新的数据接收，若在新的数据接收完成前（RI 置 1）还未读取之前接收缓冲区中的数据，在那么接收溢出位（RXROV 位）置位。如果发生了接收溢出，接收缓冲器中原来的数据不影响，后面的数据则丢失。

15.3.3 帧出错

如果检测到一个无效（低）停止位，那么帧出错位（FE 位）置 1。

15.4 UART 相关寄存器

15.4.1 S1CON寄存器

S1CON 寄存器

A5h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
S1CON	FE	RXROV	TXCOL	REN	TB8	RB8	-	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	-	-
POR 的值	0	0	0	0	0	0	-	-

Bit [7]　FE：帧错误检测位
　　1 = 有帧错误，硬件置 1
　　0 = 无帧错误或软件清 0
Bit [6]　RXROV：接收溢出标志位
　　1 = 接收溢出，硬件置 1
　　0 = 无接收溢出或软件清 0
Bit [5]　TXCOL：发送冲突标志位
　　1 = 有发送冲突，硬件置 1
　　0 = 无发送冲突或软件清 0
Bit [4]　REN：串行接收使能控制位
　　1 = 使能串行接收
　　0 = 禁止串行接收
Bit [3]　TB8：为要发送的第 9 位数据，由软件置 1 或清 0
Bit [2]　RB8：为接收到的第 9 位数据，做奇偶校验位或地址帧/数据帧的标志位

15.4.2 S1CON2寄存器

S1CON2 寄存器

A6h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
S1CON2	-	-	BRTR	-	-	UARTEN	SM1	SM2
R/W	-	-	R/W	-	-	R/W	R/W	R/W
POR 的值	-	-	0	-	-	0	0	0

Bit [5]　BRTR：独立波特率发生器 BRT 运行控制位
　　1 = 启动独立波特率发生器 BRT 工作
　　0 = 停止独立波特率发生器 BRT 工作
Bit [2]　UARTEN：UART 使能位
　　1 = 使能 UART
　　0 = 禁止 UART
Bit [1]　SM1：串口工作方式选择
　　1 = 9 位 UART
　　0 = 8 位 UART
Bit [0]　SM2：第 9 位检测使能位
　　1 = 在方式 0 下，只有停止位为 1 才能置位 RI
　　在方式 1 下，只有第 9 位为 1 才能置位 RI
　　0 = 在方式 0 下，不检测停止位，停止位无论是 0 还是 1 都会置位 RI
　　在方式 1 下，不检测第 9 位，第 9 位无论是 0 还是 1 都会置位 RI

15.4.3 SBUF寄存器

SBUF 寄存器

A7h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SBUF	SBUF[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

Bit[7:0]　SBUF[7:0]：串口缓冲数据寄存器

注：写 SBUF 为需要发送的数据，读 SBUF 为接收到的数据

15.4.4 BRTL、BRTH寄存器

BRTL 寄存器

A8h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BRTL	BRTL[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

Bit[7:0]　BRTL[7:0]：波特率发生器寄存器 BRT 低 8 位，用于保存重装时间常数

BRTH 寄存器

A9h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BRTH	BRTH[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

Bit[7:0]　BRTH[7:0]：波特率发生器寄存器 BRT 高 8 位，用于保存重装时间常数

波特率计算公式：

$$Baud = \frac{Fcpu}{16*(65536-BRT)}，\ BRT = [BRTH:BRTL]$$

16 串行外部设备接口SPI

16.1 SPI 特性

- 全双工，三/四线同步传输
- 主从机操作
- 4 级可编程主时钟频率
- 极性相位可编程的串行时钟
- 可选择数据传输方向
- 写冲突及接收溢出标志
- 带 MCU 中断的主模式故障出错标志
- 带 MCU 中断的传输结束标志
- 主模式支持高达 0.5Mbps 通信速率（Fcpu=8MHz），从模式下通信速率须在 Fcpu/16 及 Fcpu/16 以下

16.2 SPI 信号描述

主输出从输入（MOSI）：该信号连接主设备和一个从设备，数据通过 MOSI 从主设备串行传送到从设备，主设备输出，从设备输入。

主输入从输出（MISO）：该信号连接主设备和一个从设备。数据通过 MISO 从从设备串行传入到主设备，从设备输出，主设备输入。若该设备为从设备且未被选时，从设备的 MISO 引脚处于高阻状态。

串行时钟（SCK）：该信号用作控制 MOSI 和 MISO 线上输入输出数据的同步移动，每 8 个时钟周期 MOSI 和 MISO 线上传送一个字节，如果从设备未被选中，SCK 信号将被此设备忽略。注意：只有主设备才能产生 SCK 信号。

从设备选择引脚（$\overline{SS}$）：每个从属外围设备由一个从选择引脚$\overline{SS}$选择，当引脚信号为低电平时，表明该从设备被选中。主设备可以通过软件控制连接于从设备$\overline{SS}$引脚的端口电平选择每个从设备，很明显，只有一个主设备可以驱动通讯网络。为了防止 MISO 总线冲突，同一时间只允许一个从设备与主设备通讯。在主设备模式中，$\overline{SS}$引脚状态关联 SPI 状态寄存器 SPSTAT 中 MODF 标志位以防止多个主设备驱动 MOSI 和 SCK。

下列情况，$\overline{SS}$引脚可以作为普通端口或其它功能使用：

（1）设备作为主设备，SPI 控制寄存器 SPCTL 寄存器的 SSIG 位置 1。这种配置仅仅存在于通讯网络中只有一个主设备的情况，因此，SPI 状态寄存器 SPSTA 中 MODF 标志位不会被置 1。

（2）设备配置为从设备，SPI 控制寄存器 SPCTL 的 CPHA 位和 SSIG 位置 1。这种配置情况存在于只有一个主设备一个从设备的通讯网络中，因此，设备总是被选中的，主设备也不需要控制从设备的$\overline{SS}$引脚选择其作为通讯目标。

从设备的$\overline{SS}$引脚被使能时，其它主设备可通过使该引脚维持低电平，从而选中该从设备。为防止 MISO 总线冲突，原则上不允许两个及以上的从设备被选中。

主设备的$\overline{SS}$引脚被使能时，若$\overline{SS}$被拉低将置模式错误标志 MODF（可中断），且 MSTR 位也将被清 0，从而使该设备强制切换成从设备。

当 MSTR = 0（从模式）及 CPHA = 0 时，SSIG 必须为 0，因为此时数据传送需要$\overline{SS}$引脚配合，才能完成多数据传送。

16.3 SPI 时钟速率

在主模式下，SPI 的速率有 4 级选择，分别是 CPU 指令时钟的 2、4、8 或 16 分频，可通过 SPCTL 寄存器的 SPR[1:0]位进行选择。

16.4 SPI 功能框图

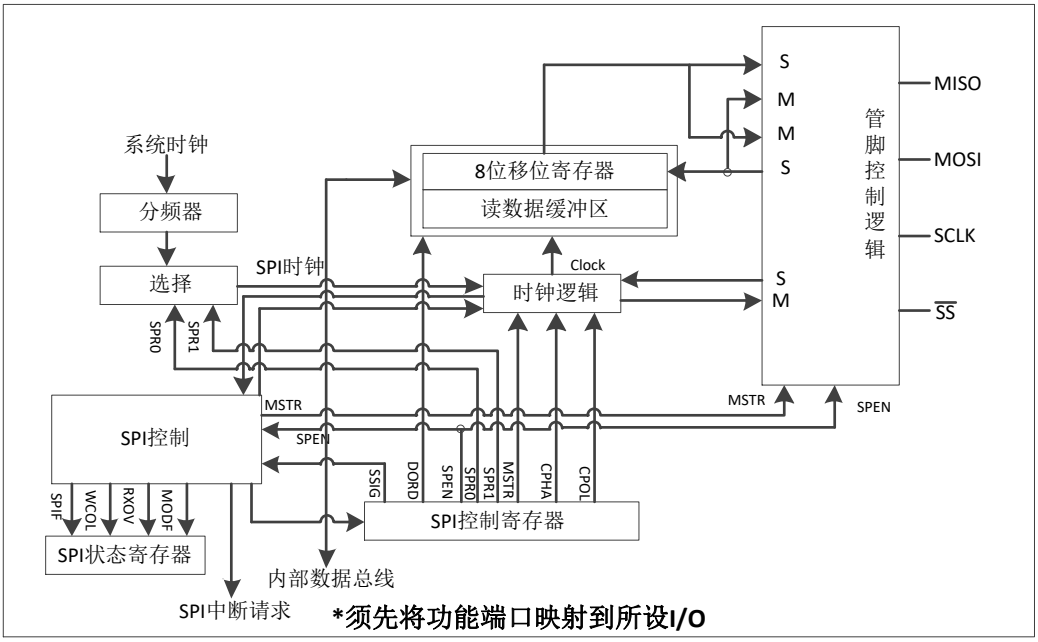


Figure 16-1 SPI 功能方框图

16.5 SPI 工作模式

SPI 可配置为主模式或从模式中的一种。SPI 模块的配置和初始化通过设置相关寄存器来完成。进一步设置相关寄存器即可完成数据传送。

在 SPI 通讯期间，数据同步地被串行的移进移出，串行时钟线（SCK）使两条串行数据线（MOSI&MISO）上数据的移动和采样保持同步。从设备选择线（$\overline{SS}$）可以独立地选择从属设备；如果从设备没有被选中，则不能参与 SPI 总线上的活动。

当 SPI 主设备通过 MOSI 线传送数据到从设备时，从设备通过 MISO 线发送数据到主设备作为相应，从而实现在同一时钟下数据发送与接收的同步全双工传输。发送移位寄存器和接收寄存器使用相同的 SFR 地址，对 SPI 数据寄存器 SPDAT 进行写操作将写入发送移位寄存器，对 SPDAT 寄存器进行读操作将获得接收移位寄存器的数据。注：写入的数据不会影响到需要读出的数据。

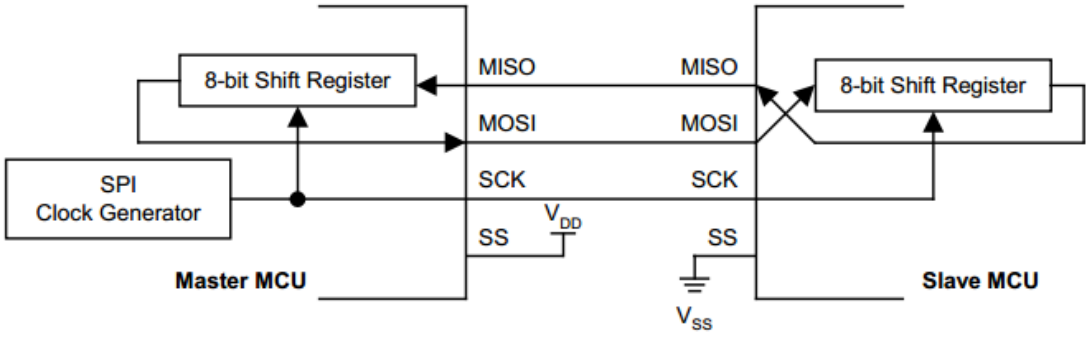


Figure 16-2 全双工主从互联图

主模式

(1)模式启动

SPI 主设备控制控制 SPI 总线上的所有数据传送的启动。一个 SPI 总线中只允许一个主设备可以启动传送。

(2)发送

在 SPI 主模式下，写一个字节数据到 SPI 数据寄存器 SPDAT，数据将会写入发送移位缓冲器。如果发送移位寄存器中已经存在一个数据或正在传送一个数据，那么主 SPI 将产生一个 WCOL 信号以表明写入太快。但是发送移

位寄存器中的数据不会受到影响，发送也不会中断。

(3)接收

当主设备通过 MOSI 线传送数据到从设备时，同时对应的从设备也可以通过 MISO 线将其发送移位寄存器的数据传送给主设备的接收移位寄存器，实现全双工操作。故 SPIF 标志置 1 即表示数据发送完成也表示数据接收完成。本 SPI 模块接收为双缓冲器，即数据可以在 SPIF 置 1 后读出，但必须在下一字节数据接收完成前读出，否则将置接收溢出标志 RXOV，如果发生接收溢出，则后面的数据将不会被移入接收寄存器，接收溢出时，SPIF 可正常置 1。

从模式

(1)模式启动

将 MSTR 置 0（若$\overline{SS}$被使能则必须拉低）时，设备处于从模式下运行，数据传送过程中设备模式不能改变（$\overline{SS}$引脚必须维持低电平），否则数据传送将失败（SPIF 不会被置 1）。

(2)发送

SPI 从设备下不能启动数据传送，所以 SPI 从设备必须在主设备开始一次新的数据传送之前将要传送给主设备的数据写入发送移位寄存器。若发送前未写入数据到发送移位寄存器，从设备将传送数据“0x00”给主设备。若写入数据时发送移位寄存器已经存在数据（或发生在传送过程中），那么 SPI 从设备的 WCOL 标志位将置 1，表示发生写 SPDAT 冲突。但是移位寄存器的数据不受影响，传送也不会被中断，传送完成 SPIF 将被置 1。

(3)接收

从模式下，按照主设备控制的 SCK 信号，数据通过 MOSI 引进移入，当计数器计数 SCK 边缘数到 8 时，表示一个字节数据接收完毕，SPIF 将置 1，数据可以通过此时读取 SPDAT 寄存器获得，但必须在下一数据接收完成前被读出，否则将置接收溢出标志 RXOV，如果发生接收溢出，则后面的数据将不会被移入接收寄存器，接收溢出时，SPIF 可正常置 1。

16.6 SPI 传送形式

通过软件设置寄存器的 CPOL 位和 CPHA 位，用户可以选择 SPI 时钟极性和相位的四种组合方式。CPOL 位定义时钟的极性，即空闲时的电平状态。CPHA 位定义时钟相位，即定义允许数据移位采样的时钟边沿。在通信的两个主从设备中，时钟极性相位设置应当保持一致。

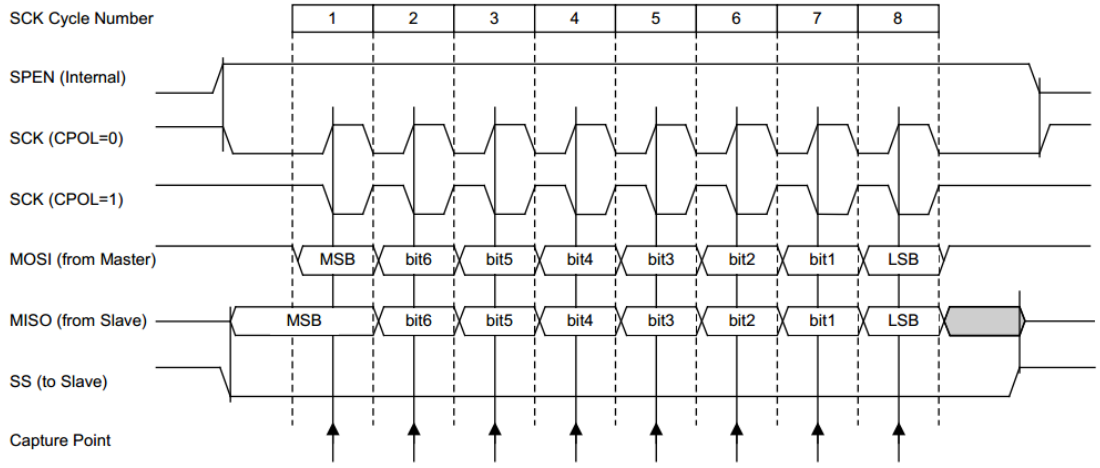


Figure 16-3 数据传送形式（CPHA=0）

如果 CPHA = 0；数据在 SCK 的第一沿就被捕获，所以从设备必须在 SCK 的第一个沿之前就准备好数据，因此，$\overline{SS}$引脚的下降沿从设备就开始数据。$\overline{SS}$引脚在每次传送完一个字节后必须拉高，在发送下一字节之前重新又被拉低，故 CPHA = 0 时，SSIG 位无效，即$\overline{SS}$脚被强制使能。

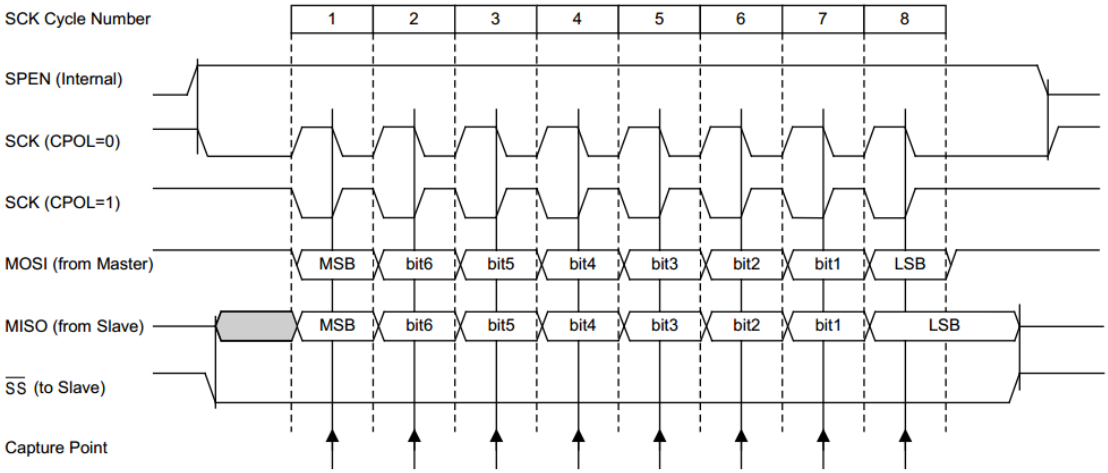


Figure 16-4 数据发送形式（CPHA=1）

如果 CPHA = 1，主设备在 SCK 的第一个沿将数据输出到 MOSI 线上，从设备吧 SCK 的第一个沿作为开始发送信号。用户必须在第一个 SCK 的前 2 个沿内完成对 SPDAT 完成写操作。传送过程中彼此模式不能改变，否则数据发送接收将失败，模式被改变的寄存器数据（发送数据）及状态（接收为空）不变。这种数据传送形式为单一主从设备间通信的首先形式。

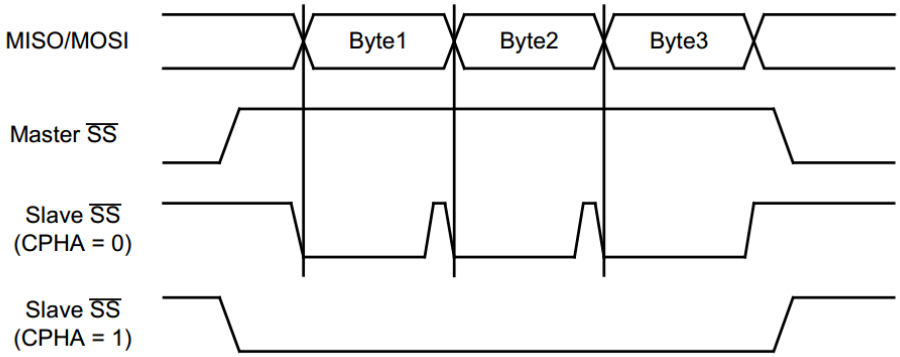


Figure 16-5 CPHA/SS 时序

16.7 SPI 出错检测

SPSTA 寄存器中的一些标志位表示 SPI 通信中的通信错误情况：

(1)模式故障（MODF）

SPI 主模式下的模式故障出错表明$\overline{SS}$引脚上的电平状态与实际设备模式不一致，MODF 标志位将被置 1（可产生中断），以来表明 SPI 控制系统中存在多主设备的冲突情况，此时硬件将自动清除 SPEN 位，即先关闭 SPI 模块；同时硬件也将自动清除 MSTR 位。需要重启 SPI 模块时，MODF 必须先软件写 1 清 0，再置 SPEN 位。

(2)写冲突（WCOL）

在数据未发送或发送期间继续对 SPDAT 做写入操作会引起写冲突，WCOL 位会被置 1，但发送不会终止。需软件写 1 清 0

(3)接收溢出（RXOV）

在接收第二数据完成前仍未清除之前接收数据产生的 SPIF 标志，将置接收溢出标志 RXOV，SPIF 被置 1 时，后面的数据将不会被传入接收寄存器，故接收的数据存入 SPDAT 前必须清除 SPIF，RXOV 位需软件写 1 清 0。

16.8 SPI 中断

两种 SPI 状态标志 SPIF&MODF 都能产生一个 CPU 中断请求。

串行数据传输完成标志 SPIF：完成一个字节数据发送/接收后由硬件置 1。

故障模式标志 MODF：该位被置 1 是指设备模式（主机）与$\overline{SS}$引脚电平不一致，SSIG 位为 1（$\overline{SS}$未被使能）时，无 MODF 中断请求。

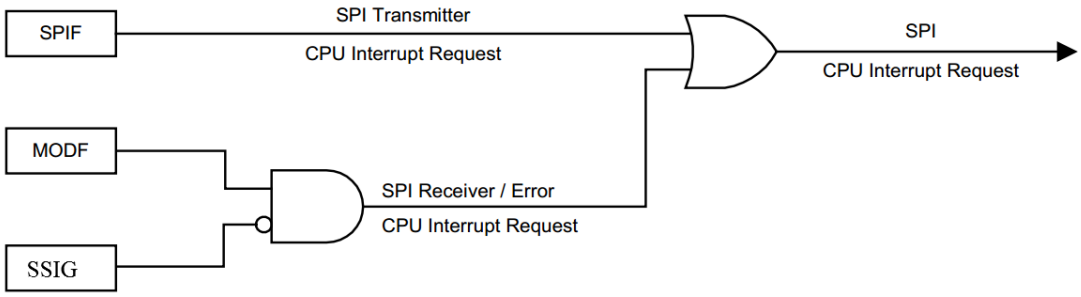


Figure 16-6 SPI 中断请求的产生

16.9 SPI 配置对照

SPEN	SSIG	$\overline{SS}$	MSTR	主或从模式	MISO	MOSI	SCK	备注
0	x	I/O	x	SPI 功能禁止	I/O	I/O	I/O	SPI 禁止
1	0	0	0	从机模式	输出	输入	输入	选择从机
1	0	1	0	从机模式未被选中	高阻	输入	输入	未被选中。MISO 为高阻，以避免总线冲突
1→0	0	0	1→0	关闭 SPI	输出	输入	输入	SS 配置为输入，SSIG 为 0。如果 SS 被驱动为低电平。则被选择作为从机。此时 MSTR 将清零，并置模式错误标志 MODF，可用于请求中断。
1	0	1	1	主（空闲）	输入	高阻	高阻	当主机空闲时 MOSI 和 SCK 为高阻态以避免总线冲突。用户必须将 SCK 上拉或下拉（根据 CPOL 的取值）以避免 SCK 出现悬浮状态。
				主（激活）		输出	输出	作为主机激活时，MOSI 和 SCK 为推挽输出。
1	1	I/O	0	从	输出	输入	输入	CPHA 不能为 0
1	1	I/O	1	主	输入	输出	输出	-

16.10 SPI 相关寄存器

16.10.1 SPCTL寄存器

SPI 控制寄存器 SPCTL

AAh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPCTL	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR[1:0]	
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

Bit [7]　SSIG：$\overline{SS}$引脚使能位
　　1 =MSTR 确认器件时主机还是从机，$\overline{SS}$脚作为普通 I/O 使用
　　0 =SS 脚被使能用于确认器件位主机还是从机
Bit [6]　SPEN：SPI 使能位
　　1 =使能 SPI 模块，相关管脚为 SPI 通信管脚
　　0 =禁止 SPI 模块，相关管脚为普通 I/O
Bit [5]　DORD：主/从机数据发送模式选择位
　　1 = LSB 先发送
　　0 = MSB 先发送
Bit [4]　MSTR：传送方向选择
　　1 = 主机模式
　　0 = 从机模式
Bit [3]　CPOL：SPI 时钟极性选择位
　　1 = SCK 空闲时为高电平
　　0 = SCK 空闲时为低电平
Bit [2]　CPHA：SPI 时钟相位选择位
　　1 = 数据在 SPI 时钟的第二边沿采样

0 = 数据在 SPI 时钟的第一边沿采样

注：SSIG = 0&CPHA = 0 时，数据在$\overline{SS}$为底被驱动；CPHA = 1 时，数据在 SCK 的前时钟沿驱动

Bit [1:0] SPR[1:0]：SPI 时钟速率选择控制位

11 = Fcpu/128
10 = Fcpu/64
01 = Fcpu/16
00 = Fcpu/4

16.10.2 SPSTAT寄存器

SPI 状态寄存器 SPSTAT

ABh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPSTAT	-	WCOL	RXOV	-	-	-	-	-
R/W	-	R/W	R/W	-	-	-	-	-
POR 的值	-	0	0	-	-	-	-	-

Bit [6] WCOL：SPI 写冲突标志位

1 =传送过程中对 SPDAT 执行写操作硬件置 1（正在传送的数据不受影响）
0 =软件写 1 清 0

注：
当 OSC_CLK 的频率 Fosc 和 CPU 的频率 Fcpu 不一致时，此标志位会错误置位，但不影响正常的 SPI 通信。

Bit [5] RXOV：SPI 接收溢出标志位

1 = 发生接收溢出，硬件置 1，LSB 先发送
0 = 软件写 1 清 0

注：
接收为双 BUFF，接收溢出发生在第二个数据传送完成前仍未清除之前接收数据产生的 SPIF 标志，故每次准备接收下一个数据前必须先清除 SPIF，否则 RXOV 将置 1，RXOV 置 1 不会影响 SPI 接收。

16.10.3 SPDAT寄存器

SPI 数据寄存器 SPDAT

ACh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPDAT	SPDAT[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

Bit[7:0] SPDAT[7:0]：SPI 数据寄存器

17 IIC总线

17.1 IIC 特性

- 双线通信
- 支持主机模式及从机模式
- 支持多主机通信时仲裁功能
- 支持地址可编程

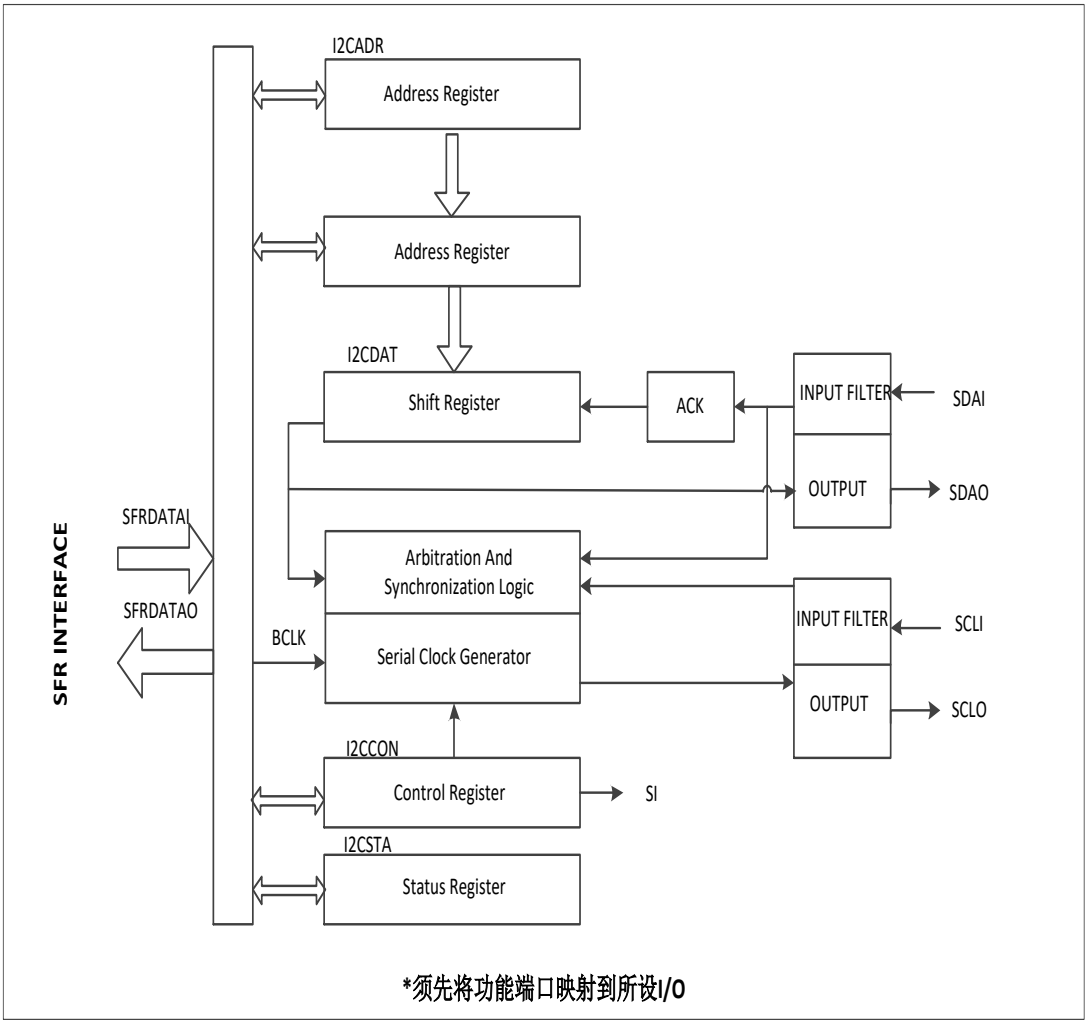


Figure 17-1 IIC 功能框图

17.2 IIC 总线工作原理

物理结构上，IIC 系统由一条串行数据线 SDA 和一条串行时钟线 SCL 组成。主机按一定的通信协议向从机寻址和进行信息传输，在数据传输时，由主机初始化一次数据传输，主机使数据在 SDA 线上传输的同时还通过 SCL 线传输时钟。信息传输的对象和方向以及信息传输的开始和终止均由主机决定。

每个器件都有一个唯一的地址，而且可以是单接收的器件或者可以接收也可以发送的器件。发送器或接收器可以在主模式或从模式下操作，这取决于芯片是否必须启动数据的传输还是仅仅被寻址。

下图是最常用、最典型的 IIC 总线连接方式。

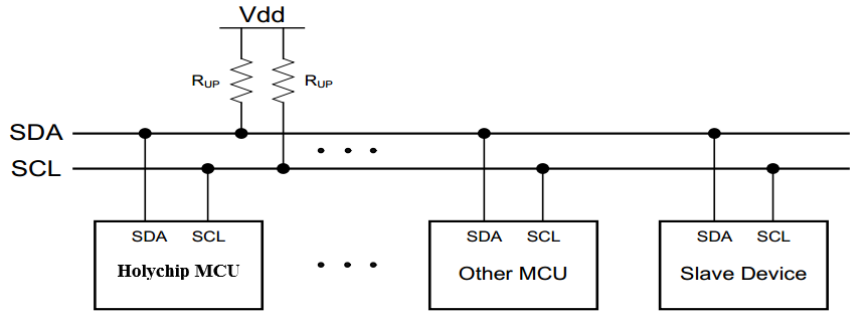


Figure 17-2 IIC 总线连接图

17.3 总线上数据的有效性

IIC 总线是以串行方式传输数据，从数据字节的最高位开始传送，每一个数据位在 SCL 上都有一个时钟脉冲相对应。在时钟线高电平期间数据线上必须保持稳定的逻辑电平状态，高电平为数据 1，低电平为数据 0。只有在时钟线为低电平时，才允许数据线上的电平状态变化，如 Figure17-3 所示。

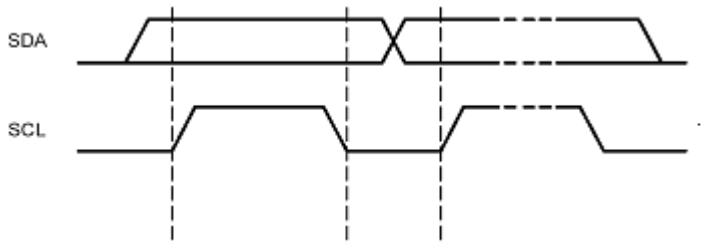


Figure 17-3 IIC 总线上数据的有效性

17.4 总线上的信号

IIC 总线在传送数据过程中共有四种类型信号，它们分别是：开始信号、停止信号、重新开始信号和应答信号。

开始信号（START）：如 Figure 16-4 所示，当 SCL 为高电平时，SDA 由高电平向低电平跳变，产生开始信号。当总线空闲的时候，例如，没有主动设备在使用总线（SDA 和 SCL 都处于高电平），主机通过发送开始（START）信号建立通信。

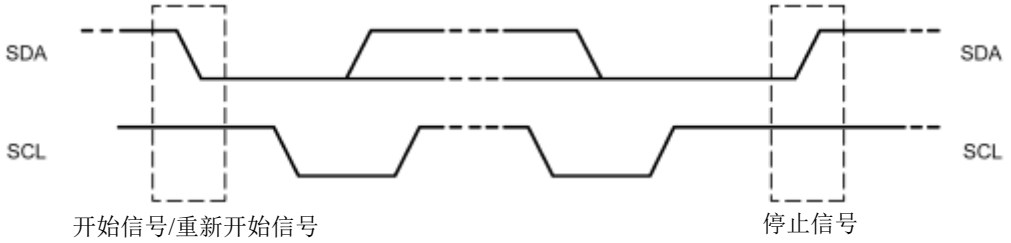


Figure 17-4 开始、重新开始、停止信号

停止信号（STOP）：如 Figure177-4 所示，当 SCL 为高电平时，SDA 由低电平向高电平跳变，产生停止信号。主机通过发送停止信号，结束数据通信。

重新开始信号（Repeated START）：在 IIC 总线上，由主机发送一个开始信号启动一次通信后，在首次发送停止信号之前，主机通过发送重新开始信号，可以转换与当前从机的通信模式，或是切换到与另一个从机通信。如 Figure177-5 所示，当 SCL 为高电平时，SDA 由高电平向低电平跳变，产生重新开始信号，它的本质就是一个开始信号。

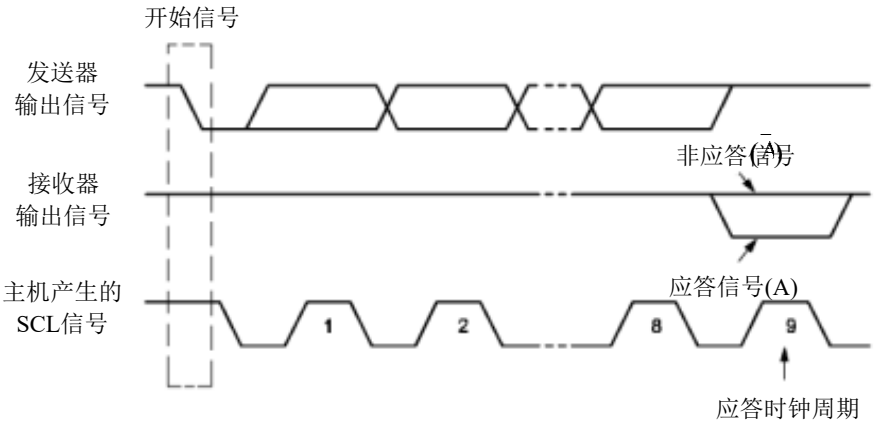


Figure 177-5 IIC 总线的应答信号

应答信号（A）：接收数据的 IIC 在接收到 8 位数据后，向发送数据的 IC 发出的特定的低电平脉冲。每一个数据字节后面都要跟一位应答信号，表示已收到数据。应答信号在第 9 个时钟周期出现，这时发送器必须在这一时钟位上释放数据线，由接收设备拉低 SDA 电平来产生应答信号，由接收设备保持 SDA 的高电平来产生非应答信号（$\bar{A}$），如 Figure17-5 所示。所以，一个完整的字节数据传输需要 9 个时钟脉冲。如果从机作为接收方向主机发送非应答信号，这样，主机方就认为此次数据传输失败；如果是主机作为接收方，在从机发送器发送完一个字节数据后，发送了非应答信号，从机就认为数据传输结束，并释放 SDA 线。不论是以上哪种情况都会终止数据传输，这时，主机或是产生停止信号释放总线，或是产生重新开始信号，开始一次新的通信。开始信号、重新开始信号和停止信号都是由主控制器产生，应答信号由接收器产生，总线上带有 IIC 总线接口的器件很容易检测到这些信号。

17.5 总线上数据初始格式

一般情况下，一个标准的 IIC 通信由四部分组成：开始信号、从机地址传输、数据传输、停止信号。

由主机发送一个开始信号，启动一次 IIC 通信；在主机对从机寻址后，再在总线上传输数据。IIC 总线上传送的每一个字节均为 8 位，首先发送的数据位为最高位，每传送一个字节后都必须跟随一个应答位，每次通信的数据字节数是没有限制的；在全部数据传送结束后，由主机发送停止信号，结束通信。

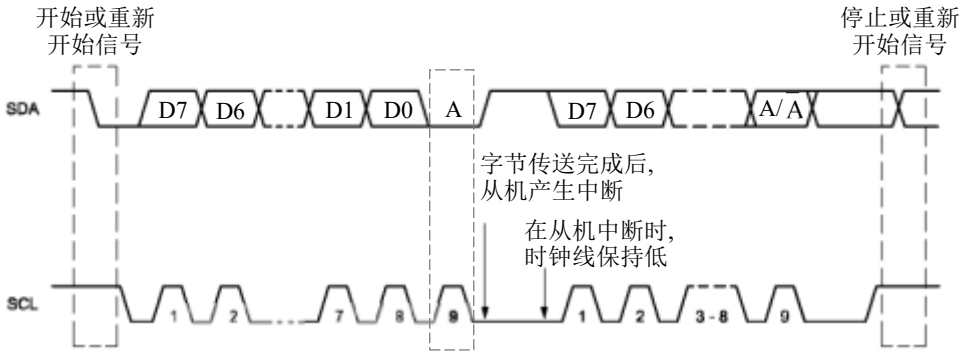


Figure 17-7 IIC 总线的数据传输格式

如 Figure 17-6 所示，时钟线为低电平时数据传送将停止进行。这种情况可以用于当接收器接收到一个字节数据后要进行一些其它工作而无法立即接收下一个数据时，迫使总线进入等待状态，直到接收器准备好接收新数据时，接收器再释放时钟线使数据传送得以继续正常进行。例如，当接收器接收完主控制器的一个字节数据后，产生中断信号并进行中断处理，中断处理完毕才能接收下一个字节数据，这时接收器在中断处理时将钳住 SCL 为低电平，直到中断处理完毕才释放 SCL。

17.6 IIC 总线寻址约定

IIC 总线系统中挂接的所有外围器件，一般均拥有一个专用的 7 位从器件地址码。由于 7 位从器件地址码，其编码空间最多只有 128 个，后来在原有的 7 位地址码格式基础上，又发展了 10 位地址码格式。10 位地址格式仍然符合总线协议。

“广播呼叫”是个例外，它可以通过将第一个字节的数据全部赋值为 0 来寻址所有器件。广播呼叫用于当主机希望发送相同信息到几个从机时情况。当该地址在使用时，其他器件根据软件配置可能响应应答或忽略。如果器件响应广播呼叫，其操作就像从机接收器模式。

17.7 主机向从机读写 1 个字节数据的过程

如 Figure 177-7 所示，主机要向从机写 1 个字节数据时，主机首先产生 START 信号，然后紧跟着发送一个从机地址，这个地址共有 7 位，紧接着的第 8 位是数据方向位（R/W），0 表示主机发送数据（写），1 表示主机接收数据（读），这时候主机等待从机的应答信号（A），当主机收到应答信号时，发送要访问的地址，继续等待从机的应答信号，当主机收到应答信号时，发送 1 个字节的数据，继续等待从机的应答信号，当主机收到应答信号时，产生停止信号，结束传送过程。

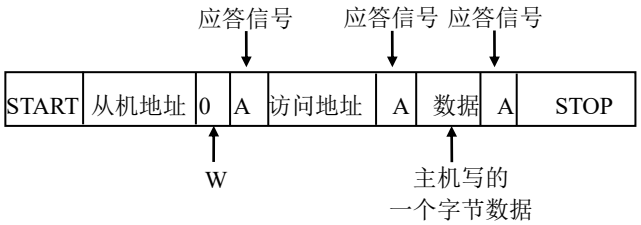


Figure 177-8 主机向从机写数据

如 Figure 177-8 所示，主机要从从机读 1 个字节数据时，主机首先产生 START 信号，然后紧跟着发送一个从机地址，注意此时该地址的第 8 位为 0，表明是向从机写命令，这时候主机等待从机的应答信号（A），当主机收到应答信号时，发送要访问的地址，继续等待从机的应答信号，当主机收到应答信号后，主机要改变通信模式（主机将由发送变为接收，从机将由接收变为发送）所以主机发送重新开始信号，然后紧跟着发送一个从机地址，注意此时该地址的第 8 位为 1，表明将主机设置成接收模式开始读取数据，这时候主机等待从机的应答信号，当主机收到应答信号时，就可以接收 1 个字节的数据，当接收完成后，主机发送非应答信号，表示不在接收数据，主机进而产生停止信号，结束传送过程。

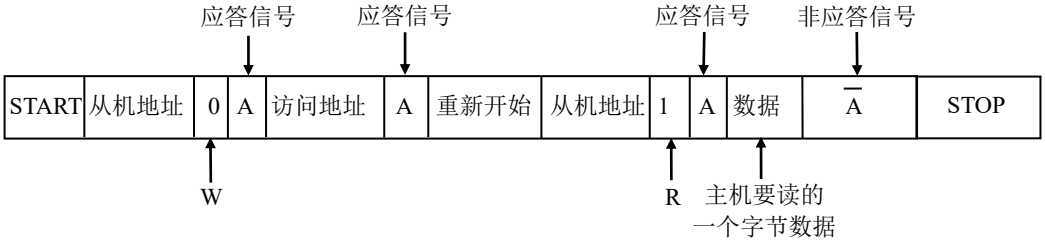


Figure 17-9 主机向从机读写数据 1 个字节数据

17.8 IIC 工作模式

17.8.1 主机发送模式

在主机发送模式下，向从机接收器发送几个数据字节。主机通过 CR[2:0]设置期望时钟速率并向 IICEN 位写 1 使能 IIC 总线，设置 STA 位为 1 进入主机发送模式，只要总线空闲，硬件将测试总线并产生起始信号，成功产生起始信号后，SI 标志位将置位且 IICSTA 的状态码为 08H，之后就是给 IICDAT 载入目标从机地址和数据方向位“写”（SLA+W），SLA+W 开始传输时 SI 位必须清零。

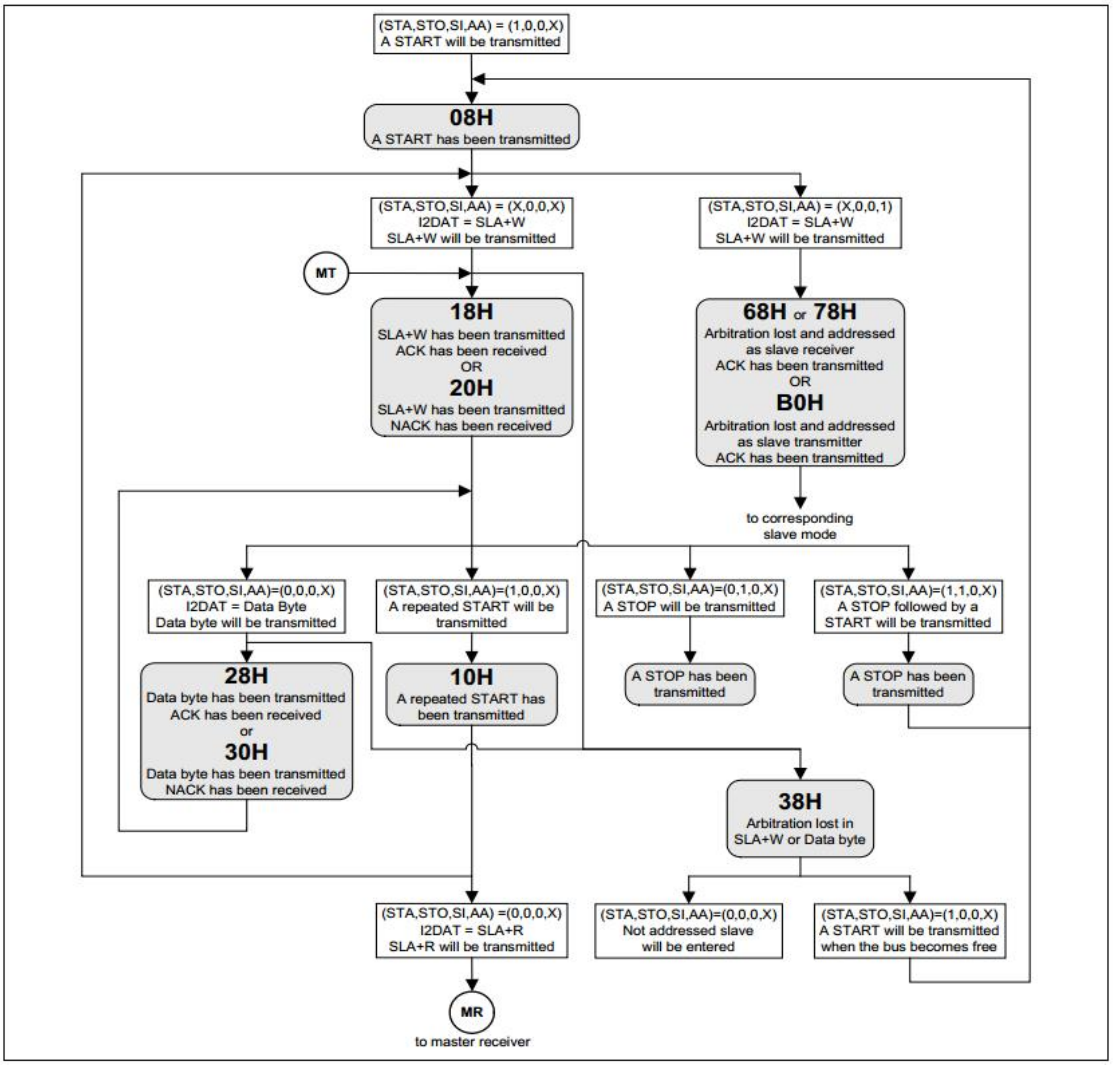


Figure 177-10 主机发送模式流程与状态

17.8.2 主机接收模式

在主机接收模式下，从从机发送器接收几个字节的数据。传输开始与主机发送模式相似，在起始信号之后，IICDAT 应该加载目标从机地址和数据方向位“读”（SLA+R），SLA+R 字节发送后，且返回应答位，重新置位 SI 标志且 IICSTA 读出为 40H，SI 标志应该被清零以便接收从机发送过来的数据，如果 AA 标志位置位，主机接收器将应答从机发送器，如果清零 AA，主机接收器将不会应答从机，并释放从机发送器为不被寻址的从机，然后主机产生停止信号或重复起始信号中止传输或开始另一次传输。

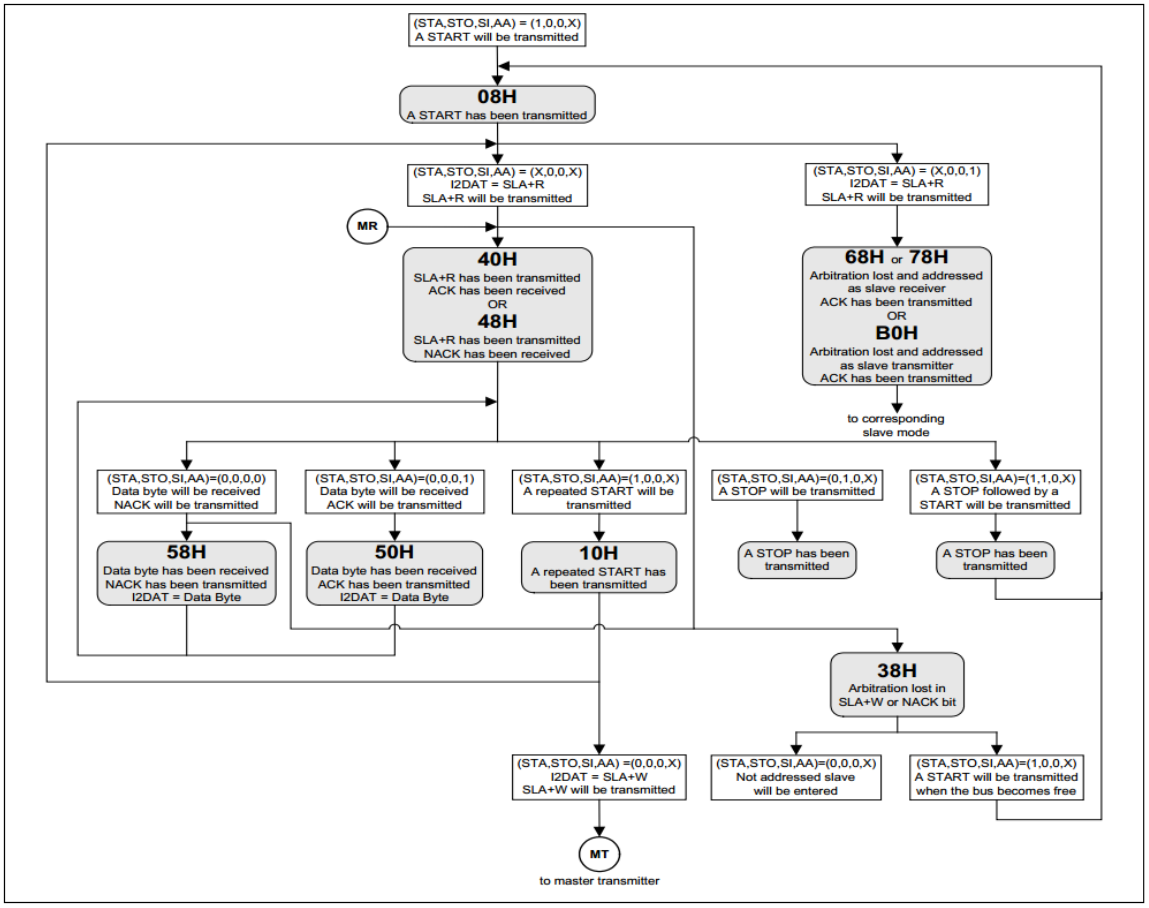


Figure 177-11 主机接收模式流程与状态

17.8.3 从机发送模式

在从机发送模式下，发送几个字节数据到主机接收器。确定 IICADR 和 IICCON 的值之后，IIC 等待自己的地址被寻址“读”（SLA+R）。如果仲裁失败后，也可以进入从机发送模式。

在从机被 SLA+W 寻址后，应该清 SI 标志以便传输数据到主机发送器，通常主机接收器将在从机发送每个字节数据之后返回应答，如果没有接收到应答，如果继续传输将发送全“1”，就成为不被寻址的从机，如果在传输中清了 AA 标志，从机发送最后一个字节数据，下一次传输数据全为“1”，从机成为不被寻址。

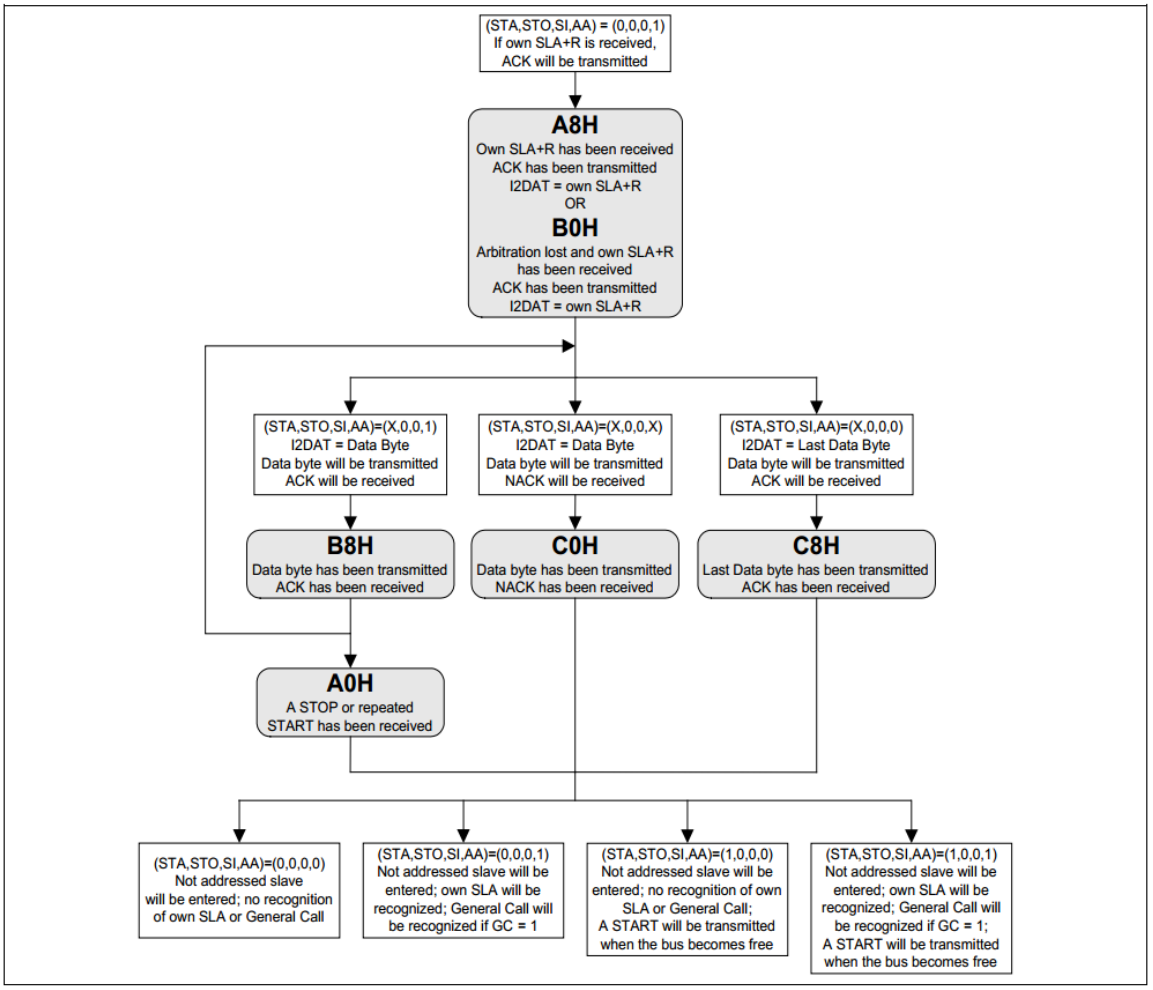


Figure 177-12 从机发送模式流程与状态

17.8.4 从机接收模式

在从机接收模式下，从主机发送器接收几个字节数据。发送开始之前，IICADR 必须装载响应器件的地址，以让主机寻址，AA 位必须设置使能应答自身从机地址或广播呼叫，完成以上初始过程后，IIC 等待自身地址被寻址与数据方向位“写”（SLA+W）或被广播呼叫寻址。如果在仲裁失败时，也可以进入从机接收模式。

在从机被 SLA+W 寻址后，应该清 SI 标志以便接收主机发送过来的数据，传输期间，如果 AA 位为 0，从机将在下一次接收到的数据字节之后返回无应答（non-acknowledge），从机也不被寻址并与主机分离，不能接收 IICDAT 的任何字节，而保持当前接收到的数据字节。

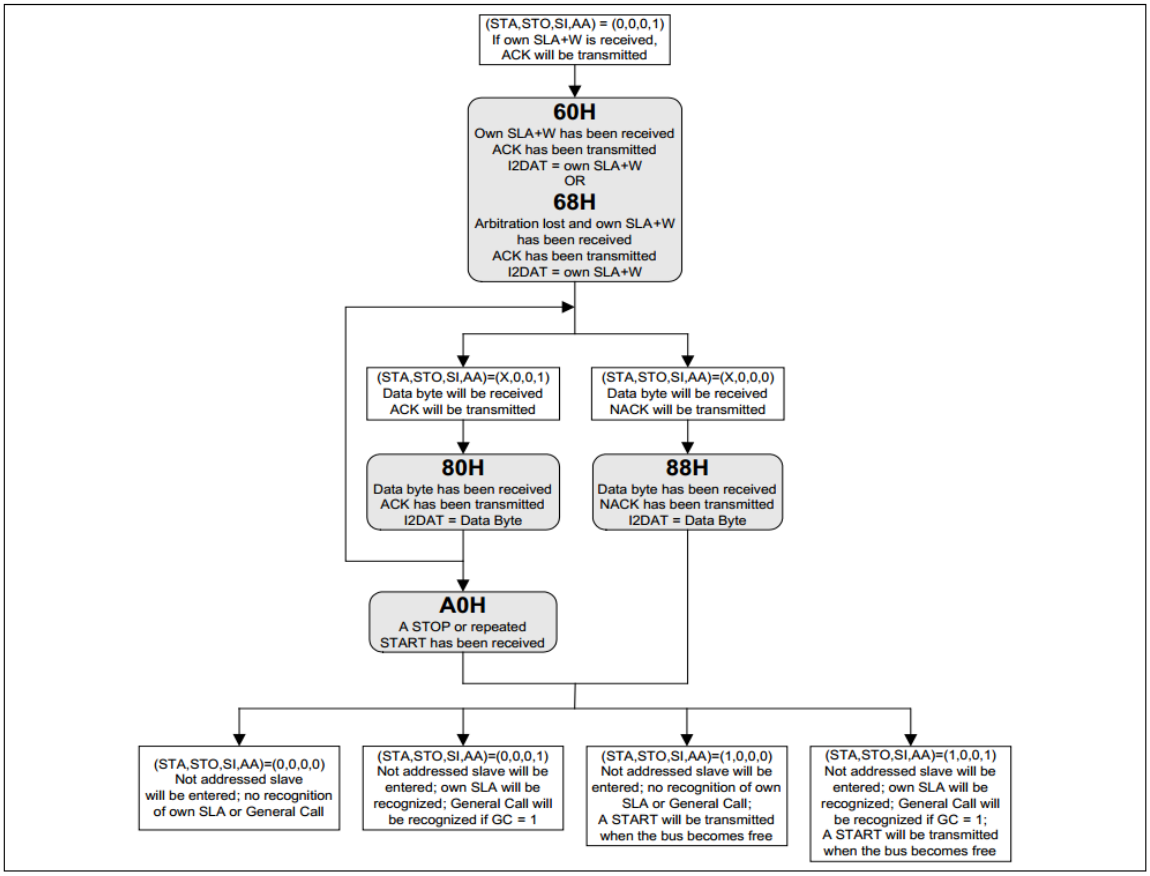


Figure 177-13 从机接收模式流程与状态

17.8.5 广播呼叫

广播呼叫是从机接收模式的一种特殊情况，即从机地址和数据方向位全为 0，被广播呼叫寻址的从机在正常从机接收模式的 IICSTA 里有不同状态码，如果仲裁失败，也可以产生广播呼叫。

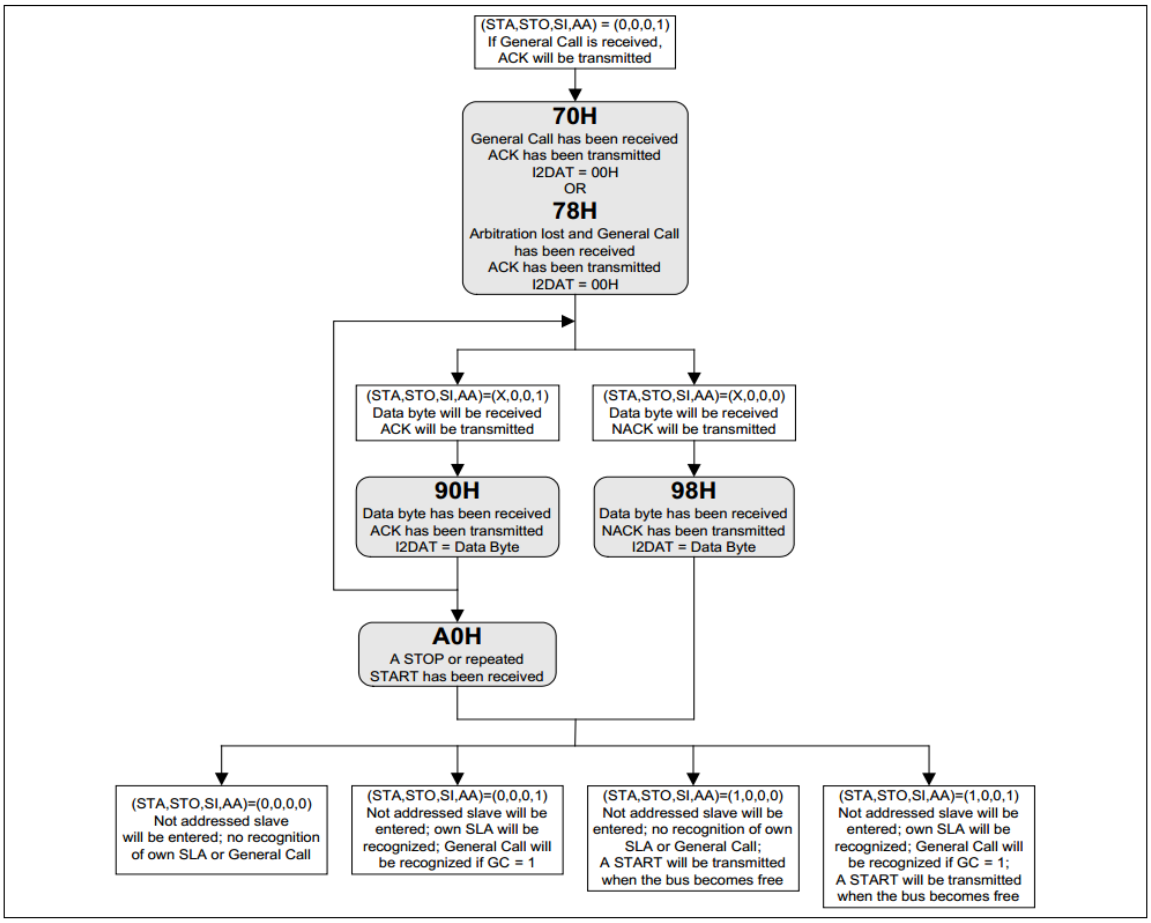


Figure 177-14 广播呼叫模式流程与状态

17.8.6 其他状态

有两个 IICSTA 状态码与 24 个定义状态不一致，即前面提到的 0F8H 和 00H 状态。

第一个状态码 0F8H 表示在每次传输期间没有得到相关信息，同时，SI 标志为 0 且没有 IIC 中断请求。

另一个标志码 00H 意味在传输过程中发生错误，总线错误是由 START 或停止信号暂时出现在一个非法的位置，如地地址字节里第 2 位换到第 8 位，或数据字节包括应答位，当出现总线错误时，SI 标志立即置位，当在 IIC 总线上检测到总线错误，工作器件立即切换到不被寻址从机模式，释放 SDA 和 SCL 总，置位 SI 标志，将 00H 载入 IICSTA。要从总线错误恢复，STO 位必须设置为逻辑 1 且 SI 必须清零，然后，STO 由硬件清零且在没有停止信号就释放 IIC 总线。

特例：如果没有成功产生 START 或重复起始信号，IIC 总线被 SDA 的低电平阻挡，如一个从 CPU 时钟件没有位同步，可以通过在 SCL 总线上发送额外时钟脉冲解决这个问题。当 STA 位置位时，IIC 硬件发送额外时钟脉冲，但是由于 SDA 被拉低，不能产生起始信号，当 SDA 总线最终被释放，发送一个普通的 START 条件，进入状态 08H，继续进行串行传输。当 SDA 为低，如果发送重复起始信号，IIC 硬件也执行以上相同的动作。此情况下，在成功发送起始信号后，进入状态 08H，而不是进入 10H。

注：软件不能解决这类总线问题。

17.9 IIC 总线相关寄存器

17.9.1 IICCON寄存器

IIC 控制寄存器 IICCON

ADh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IICCON	CR2	IICEN	STA	STO	-	AA	CR1	CR0
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]　CR2：IIC 通信时钟选择位 2

Bit [6]　IICEN：IIC 模块使能位

1 = 启动 IIC 模块

0 = 禁止 IIC 模块

Bit [5]　STA：起始位

1 = 总线空闲时产生起始信号。忙时，等待停止信号后产生一个起始信号。主机模式下，IIC 准备好发送或接收一个或多个字节时，置 1 产生一个重复的起始信号

0 = 不发送起始信号

Bit [4]　STO：停止位

1 = 主机模式时产生停止信号，当检测到总线上出现停止信号。IIC 硬件清除 STO 标志。STO 标示的设置也用于将 IIC 设备从错误状态（IICSTA 为 00H）恢复，此条件下，没有停止信号发送 IIC 总线上。若 STA 和 STO 都置 1，且在主机模式下设备为原始的，IIC 总线将产生停止信号并立即伴随着起始信号。如果设备为从机模式，置 STO 恢复到非寻址从机，STO 将会硬件清 0。

0 = 不发送停止信号

Bit [2]　AA：为接收到的第 9 位数据，做奇偶校验位或地址帧/数据帧的标志位

1 = 回复 ACK（SDA 上为低电平）

0 = 回复 NACK（SDA 上为高电平）

Bit [1]　CR1：IIC 通信时钟选择位 1

Bit [0]　CR0：IIC 通信时钟选择位 0

CR[2:0] IIC 通信时钟选择位：

CR2	CR1	CR0	Fcpu		分频系数
			4 MHz	8 MHz	
0	0	0	15.625KHz	31.2KHz	256
0	0	1	17.85KHz	35.7KHz	224
0	1	0	20.8KHz	41.6KHz	192
0	1	1	25KHz	50KHz	160
1	0	0	4.16KHz	8.3KHz	960
1	0	1	33.3KHz	66KHz	120
1	1	0	66.6KHz	133KHz	60
1	1	1	无效位		

17.9.2 IICSTA寄存器

IIC 状态寄存器 IICSTA

AEh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IICSTA	IICSTA[7:3]					-	-	-
R/W	R	R	R	R	R	-	-	-
POR 的值	1	1	1	1	1	-	-	-

Bit[7:3] IICSTA[7:3]：IIC 状态码，共 26 个可能得状态码，状态码中除 0F8H 外都可置 SI 标志

17.9.3 IICDAT寄存器

IIC 数据寄存器 IICDAT

AFh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IICDAT	IICDAT[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

Bit[7:0] IICDAT[7:0]：IIC 数据

IICDAT 包含一个字节的将被发送或刚接收到的 IIC 数据。只要 SI 为逻辑 1，IICDAT 中的数据保持不变，在 IIC 发送接收过程中，读或写 IICDAT 的结果都是不确定的。

当 IICDAT 的数据被移出，总线上的数据同步被移入以更新 IICDAT。IICDAT 常显示当前 IIC 总线上的最后字节。因此失去仲裁，在传输之后的 IICDAT 原始值被改变。

17.9.4 IICADR寄存器

IIC 地址寄存器 IICADR

B0h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IICADR	IICADR[7:1]							GC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit[7:1] IICADR[7:1]：

从机模式：IIC 设备自身从机地址

主机模式：该数据无影响

Bit[0] GC：广播呼叫位

1 = 如果 AA 标志为 1，广播呼叫被识别；如果 AA 为 0，忽略广播呼叫

0 = 广播呼叫常被忽略

注：

该位只在从机模式有效，主机模式无影响。做从机时，置 AA 标志，在空闲模式下，若总线其它主机寻址地址与本从机地址匹配，则将唤醒本从机。

18 软件LCD

LCD 模块具有如下功能：

- 支持 1/2Bias 和 1/3Bias 的 LCD 点阵
- 驱动能力可配置
- LCD 控制信号（COM 和 SEG）由软件程序实现

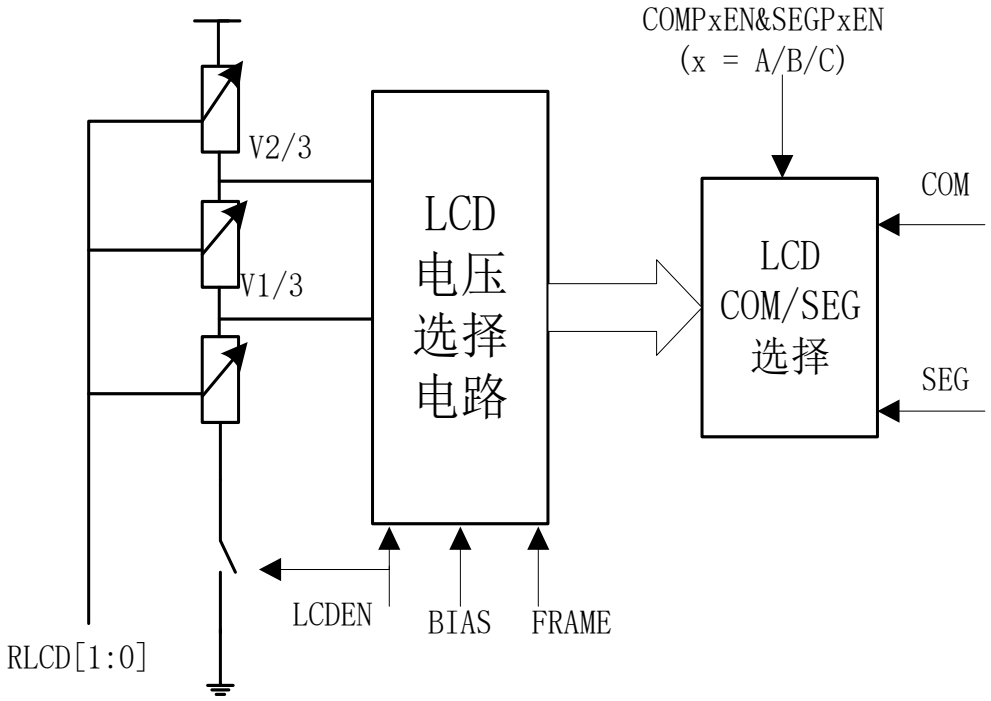


Figure 18-1 LCD 系统框图

18.1 LCD 软件流程说明

下面以 COM[3:0] = COMPBEN[3:0]，SEG[7:0] = SEGPCEN[7:0]为例：

(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) 设置 COM 口使能控制寄存器或 SEG 口使能控制寄存器，COMPBEN=0x0F，SEGPCEN=0xFF，这是具体设置某个 IO 的状态，使能 LCD 的模拟通道，需配置 IO 口为模拟输出模式；
(4) 设定 Frame0（FRAME=0），用于确定点亮和非点亮电平；
(5) 设置定时开始，设置 COM 端口数据寄存器 PORTB=0x01，设置 SEG 数据寄存器 PORTC=0xXX，等待定时结束；
(6) 设置定时开始，设置 COM 端口数据寄存器 PORTB=0x02，设置 SEG 数据寄存器 PORTC=0xXX，等待定时结束；
(7) 设置定时开始，设置 COM 端口数据寄存器 PORTB=0x04，设置 SEG 数据寄存器 PORTC=0xXX，等待定时结束；
(8) 设置定时开始，设置 COM 端口数据寄存器 PORTB=0x08，设置 SEG 数据寄存器 PORTC=0xXX，等待定时结束；
(9) 设定 Frame1（FRAME=1），用于确定点亮和非点亮电平；
(10) 设置定时开始，设置 COM 端口数据寄存器 PORTB=0x01，设置 SEG 数据寄存器 PORTC=0xXX，等待定时结束；
(11) 设置定时开始，设置 COM 端口数据寄存器 PORTB=0x02，设置 SEG 数据寄存器 PORTC=0xXX，等待定时结束；
(12) 设置定时开始，设置 COM 端口数据寄存器 PORTB=0x04，设置 SEG 数据寄存器 PORTC=0xXX，等待定时结束；
(13) 设置定时开始，设置 COM 端口数据寄存器 PORTB=0x08，设置 SEG 数据寄存器 PORTC=0xXX，等待定时结束；
(14) 循环 4-13。

18.2 LCD 帧

一个完整的LCD波形周期包含两个Frame，即Frame0和Frame1。

Frame 0

当要输出Frame0的波形，需将LCDCON寄存器中FRAME设为0。

在Frame0中，COM信号输出可以是VDD，或是VBIAS=1/3VDD（1/2VDD）；

在Frame0中，SEG信号输出可以是GND，或是VBIAS=2/3VDD（VDD）。

Frame 1

当要输出Frame1的波形，需将LCDCON寄存器中FRAME设为1。

在Frame1中，COM信号输出可以是GND，或是VBIAS=2/3VDD（1/2VDD）；

在Frame1中，SEG信号输出可以是VDD，或是VBIAS=1/3VDD（GND）。

例：COMPCEN = 0x03;(使能PORTC0、PORTC1当做COM1、COM2口)，SEGPAEN = 0x03;(使能PORTA0、PORTA1当做SEG1、SEG2口)

```
ANSELA = 0x00;
TRISA = 0x03;          //配置PA0/1为模拟输出模式
ANSELC = 0x00;
TRISC = 0x03;          //配置PC0/1为模拟输出模式
SEGPAEN = 0x03;
COMPCEN = 0x03;

Frame = 0;
PORTC0 = 1;
PORTC1 = 0;
PORTA0 = 1;
PORTA1 = 0;
COM1输出VDD、COM2输出1/3VDD(1/2VDD)
SEG1输出GND、SEG2输出2/3VDD(VDD)

Frame = 1;
PORTC0 = 1;
PORTC1 = 0;
PORTA0 = 1;
PORTA1 = 0;
COM1输出GND、COM2输出2/3VDD(1/2VDD)
SEG1输出VDD、SEG2输出1/3VDD(GND)
```

通过软件设定FRAME位及相应的I/O数据寄存器来决定COM口目前输出的是VDD，GND或VBIAS。通过软件设定FRAME位及相应的I/O数据寄存器来决定SEG口目前输出是VDD，GND或VBIAS（在1/2bias时，SEG只输出VDD或GND）。

下面的波形图显示了一个利用应用程序产生的典型 1/2Bias LCD 波形。其中写“1”代表点亮 LCD。COMn 和 SEGm 引脚上所产生的 COM 和 SEG 信号极性（0 或 1）通过相应的端口数据寄存器位来产生。

1/2Bias, 1/4Duty

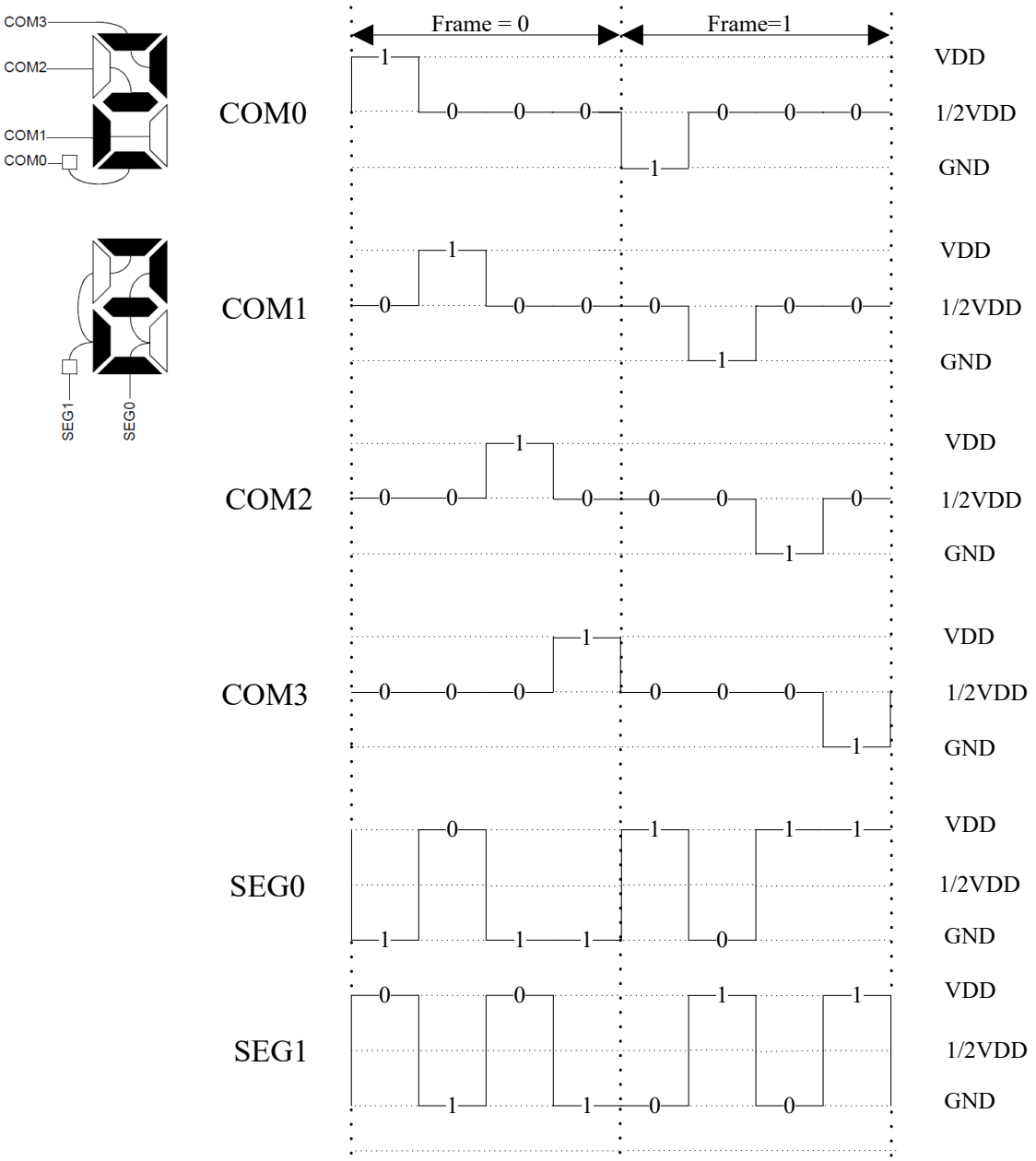


Figure 18-1 1/2bias LCD 波形图

注：
图形中的逻辑值为 COM 或 SEG 对应端口数据寄存器的位值。

下面的波形图显示了一个利用应用程序产生的典型 1/3Bias LCD 波形。其中写“1”代表点亮 LCD。COMn 和 SEGm 引脚上所产生的 COM 和 SEG 信号极性（0 或 1）通过相应的端口数据寄存器位来产生。

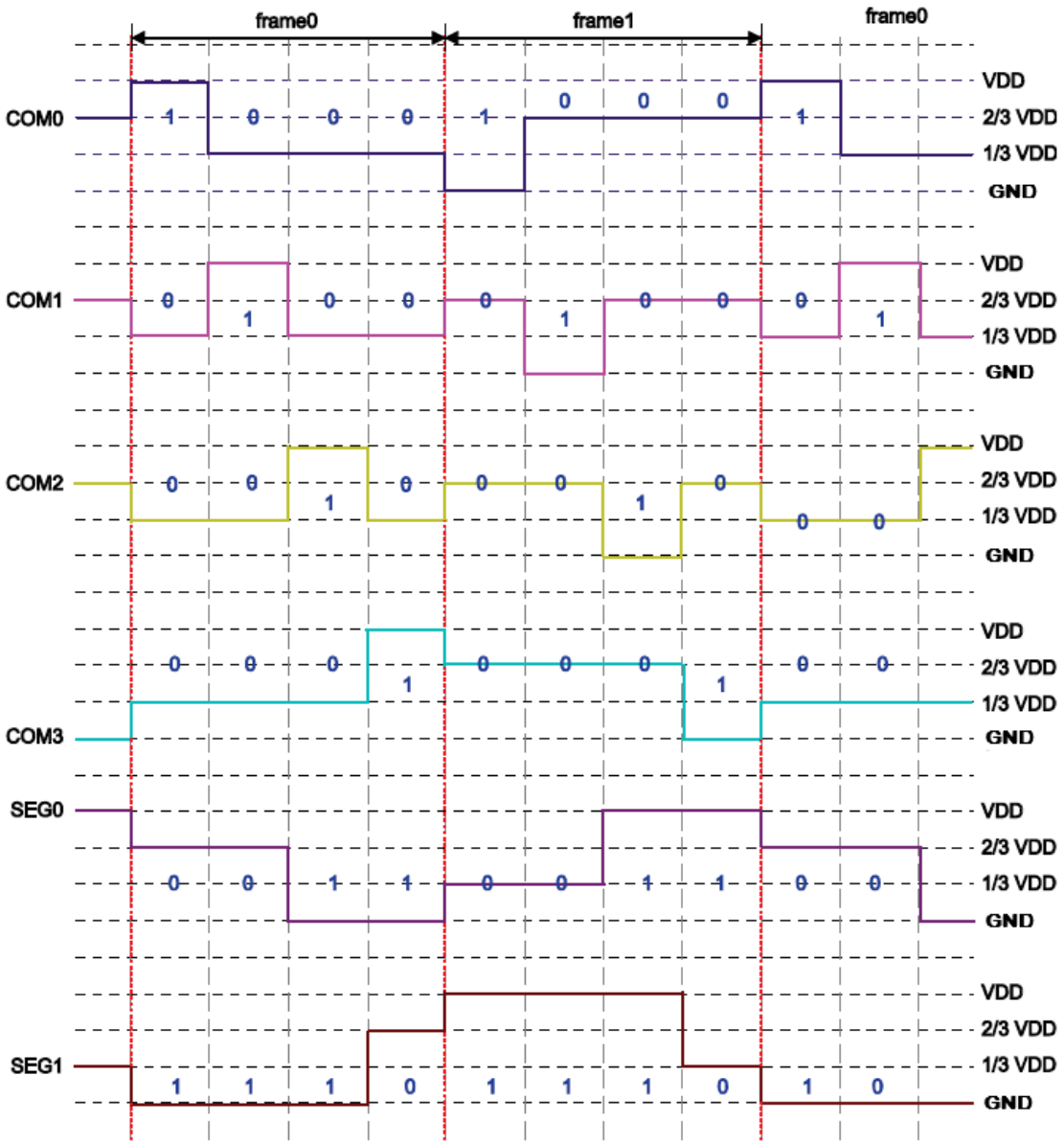


Figure 18-2 1/3bias LCD 波形图

注：

图形中的逻辑值为 COM 或 SEG 对应端口数据寄存器的位值。

18.3 LCD 相关寄存器

18.3.1 SEGPAEN寄存器

SEGPAEN 寄存器

91h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SEGPAEN	-	SEGPA6EN	SEGPA5EN	SEGPA4EN	SEGPA3EN	SEGPA2EN	SEGPA1EN	SEGPA0EN
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 [6:0] SEGPAEN：SEG 口使能选择

1 = 使能 SEG 口

0 = 禁止，普通 IO 口

18.3.2 SEGPBEN寄存器

SEGPBEN 寄存器

92h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SEGPBEN	-	SEGPB6EN	SEGPB5EN	SEGPB4EN	SEGPB3EN	SEGPB2EN	SEGPB1EN	SEGPB0EN
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 [6:0] SEGPBEN：SEG 口使能选择

1 = 使能 SEG 口

0 = 禁止，普通 IO 口

18.3.3 SEGPCEN寄存器

SEGPCEN 寄存器

93h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SEGPCEN	SEGPC7EN	SEGPC6EN	SEGPC5EN	SEGPC4EN	SEGPC3EN	SEGPC2EN	SEGPC1EN	SEGPC0EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7:0] SEGPCEN：SEG 口使能选择

1 = 使能 SEG 口

0 = 禁止，普通 IO 口

18.3.4 COMPAEN寄存器

COMPAEN 寄存器

94h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COMPAEN	-	COMPA6EN	COMPA5EN	COMPA4EN	COMPA3EN	COMPA2EN	COMPA1EN	COMPA0EN
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 [6:0] COMPAEN：COM 口使能选择

1 = 使能 COM 口

0 = 禁止，普通 IO 口

18.3.5 COMPBEN寄存器

COMPBEN 寄存器

95h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COMPBEN	-	COMPB6 EN	COMPB5 EN	COMPB4 EN	COMPB3 EN	COMPB2 EN	COMPB1 EN	COMPB0 EN
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 [6:0] COMPBEN：COM 口使能选择

1 = 使能 COM 口

0 = 禁止，普通 IO 口

18.3.6 COMPCEN寄存器

COMPCEN 寄存器

96h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
COMPCEN	COMPC7 EN	COMPC6 EN	COMPC5 EN	COMPC4 EN	COMPC3 EN	COMPC2 EN	COMPC1 EN	COMPC0 EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR 的值	0	0	0	0	0	0	0	0

Bit [7:0] COMPCEN：COM 口使能选择

1 = 使能 COM 口

0 = 禁止，普通 IO 口

18.3.7 LCDCON寄存器

LCDCON 寄存器

90h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LCDCON	RLCD[1:0]		BIAS	FRAME	-	-	-	LCDEN
R/W	R/W	R/W	R/W	R/W	-	-	-	R/W
POR 的值	0	0	0	0	-	-	-	0

Bit [7:6] RLCD[1:0]：偏置电阻选择

00 =偏置电阻总和 600K

01 =偏置电阻总和 300K

10 =偏置电阻总和 100K

11 =偏置电阻总和 50K

Bit [5] BIAS：LCD 偏压选择位

1 =1/3 BIAS

0 =1/2 BIAS

Bit [4] FRAME：FRAME0 和 FRAME1 输出使能位

1 = Frame1

0 = Frame0

Bit [0] LCDEN：LCD 模块使能位

1 = 使能 LCD 模块

0 = 禁止 LCD 模块

19 指令表

助记符操作数		说明	周期数	16 位操作码				受影响的状态位
ADDWF	f,d	W 和 f 相加	1	0111	10df	ffff	ffff	C，DC，Z
ADDLW	k	将立即数和 W 相加	1	0000	1000	kkkk	kkkk	C，DC，Z
SUBWF	f,d	f 减去 W	1	0110	11df	ffff	ffff	C，DC，Z
SUBLW	k	立即数减去 W	1	0000	1010	kkkk	kkkk	C，DC，Z
DAW	-	W 寄存器值进行 BCD 调整	1	0000	0000	0000	0101	C，DC
ANDWF	f,d	W 和 f 作逻辑与运算	1	0111	00df	ffff	ffff	Z
ANDLW	k	立即数和 W 作逻辑与运算	1	0000	1110	kkkk	kkkk	Z
IORWF	f,d	W 和 f 作逻辑或运算	1	0111	01df	ffff	ffff	Z
IORLW	k	立即数和 W 作逻辑或运算	1	0000	1111	kkkk	kkkk	Z
XORWF	f,d	W 和 f 作逻辑异或运算	1	0111	11df	ffff	ffff	Z
XORLW	k	立即数和 W 作逻辑异或运算	1	0000	1101	kkkk	kkkk	Z
COMF	f,d	f 取反	1	0100	00df	ffff	ffff	Z
CLRW	-	将 W 清零	1	0001	1000	0000	0000	Z
CLRF	f	将 f 清零	1	0110	001f	ffff	ffff	Z
INCF	f,d	f 加 1	1	0100	11df	ffff	ffff	Z
INCFSZ	f,d	f 加 1，为 0 则跳过	1(2)	0101	10df	ffff	ffff	-
DECF	f,d	f 减 1	1	0110	10df	ffff	ffff	Z
DECFSZ	f,d	f 减 1，为 0 则跳过	1(2)	0100	10df	ffff	ffff	-
BCF	f,d	将 f 中的 d 位清 0	1	1111	bbbf	ffff	ffff	-
BSF	f,d	将 f 中的 d 位置 1	1	1110	bbbf	ffff	ffff	-
BTFSC	f,d	检测 f 中的 d 位，为 0 则跳过	1(2)	1101	bbbf	ffff	ffff	-
BTFSS	f,d	检测 f 中的 d 位，为 1 则跳过	1(2)	1100	bbbf	ffff	ffff	-
MOVWF	f	将 W 的内容传送到 f	1	0110	011f	ffff	ffff	-
MOVF	f,d	将 f 的内容送到目标寄存器	1	0100	01df	ffff	ffff	Z
MOVLW	k	将立即数 k 传送到 W	1	0000	0100	kkkk	kkkk	-
RLF	f,d	对 f 执行带进位的循环左移	1	0101	00df	ffff	ffff	C
RRF	f,d	对 f 执行带进位的循环右移	1	0101	01df	ffff	ffff	C
SWAPF	f,d	将 f 的两个半字节进行交换	1	0101	11df	ffff	ffff	-
CALL	k	调用子程序	2	101k	kkkk	kkkk	kkkk	-
GOTO	k	无条件跳转	2	100k	kkkk	kkkk	kkkk	-
RETFIE	-	从中断返回	2	0000	0000	0000	0001	GIE
RETURN	-	从子程序返回	2	0000	0000	0000	0010	-
RETLW	k	返回时将立即数传送到 W	2	0000	0001	kkkk	kkkk	-
CLRWDT	-	清零看门狗定时器	1	0000	0000	0000	0100	TO，PD
SLEEP	-	进入待机模式	1	0000	0000	0000	0011	TO，PD
NOP	-	空操作	1	0000	0000	0000	0000	-

20 电气特性

20.1 极限参数

储存温度……-50°C ~125°C
工作温度……-40°C ~105°C
电源供应电压……VSS-0.3V~VSS+6.0V
端口输入电压……VSS-0.3V~VDD+0.3V
流过 VDD 最大电流……100mA
流过 GND 最大电流……150mA

20.2 直流特性

注意：如无另外说明，以上数据测试条件均为 VDD=5V、常温 25°C。

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）				
VDD1	工作电压	—	FCPU = 8MHz	2.7	-	5.5	V
VDD2	工作电压	—	FCPU = 0~4MHz	2.0	-	5.5	V
IDD1	工作电流	3V	FCPU = 8MHz，高频模式	-	1.4	-	mA
		5V	WDT 禁止，无负载	-	2.7	-	mA
IDD2	工作电流	3V	FCPU = 4MHz，高频模式	-	1.1	-	mA
		5V	WDT 禁止，无负载	-	2	-	mA
IDD3	工作电流	3V	FCPU = 2MHz，高频模式	-	0.9	-	mA
		5V	WDT 禁止，无负载	-	1.6	-	mA
IDD4	工作电流	3V	FCPU = 20KHz，低频模式，WDT 禁止，无负载	-	7	-	μA
		5V		-	10	-	μA
Isb1	静态电流	3V	休眠模式，WDT 使能，无负载，BOR 禁止	-	8	-	μA
		5V		-	13	-	μA
Isb2	静态电流	3V	休眠模式，WDT 使能，无负载，BOR 使能	-	22	-	μA
		5V		-	30.5	-	μA
Isb3	静态电流	3V	休眠模式，WDT 禁止，无负载，BOR 禁止	-	-	1	μA
		5V		-	-	1	μA
Isb4	静态电流	3V	休眠模式，WDT 禁止，无负载，BOR 使能	-	16	-	μA
		5V		-	19	-	μA
ILC	端口输入漏电流	3V	端口输入模式，VIN=VDD 或 GND	-1	0	1	μA
		5V	端口输入模式，VIN=VDD 或 GND	-1	0	1	μA
VIL1	输入低电平	5V	非施密特输入口	-	0.5VDD	-	V
VIH1	输入高电平	5V		-	0.5VDD	-	V
VIL2	输入低电平	5V	施密特输入口	-	0.3VDD	-	V
VIH2	输入高电平	5V		-	0.7VDD	-	V
IOL1	输出灌电流	5V	输出口，Vout=VSS+0.5V(PORTA/PORTB)	17.5	25	32.5	mA
IOL3	输出灌电流	5V	输出口，Vout=VSS+0.5V(PORTC)	31.5	45	58.5	mA
IOH1	输出拉电流	5V	输出口，Vout=VDD-0.5V(PORTA/PORTB)	5	7.5	9.5	mA
IOH2	输出拉电流	5V	输出口，Vout=VDD-0.5V(PORTC)	8	13	17	mA
RPH	内部上拉电阻	5V	上拉电阻	84	120	156	kΩ
RPD	内部下拉电阻	5V	下拉电阻	84	120	156	kΩ
V_{BOR}	低电压复位	—	-	-6%	档位电压	+6%	V
V_{POR}	上电复位电压	—	-	-	1.35	-	V

V_{BG}	V_{BG} 电压精度	—	VDD = 5V/3V	1.19	1.233	1.25	V

20.3 AC 特性

参数	符号	条件	最小值	典型值	最大值	单位
内部 RC16M 启动时间	Tset1	常温，VDD=5V	-	-	10	μs
内部 RC40K 启动时间	Tset2	常温，VDD=5V	-	-	150	μs
内部高频 RC 频率精度	F_{IRC1}	VDD=1.8V~5.5V， 25°C	16(1-1%)	16	16(1+1 %)	MHz
内部高频 RC 频率精度	F_{IRC2}	VDD=5.0V，-40°C ~+85°C	16(1-2.5%)	16	16(1+2.5%)	MHz
内部低频 RC 频率精度	F_{WRC1}	VDD=1.8V~5.5V， 25°C	-50%	40	+50%	KHz
内部低频 RC 频率精度	F_{WRC2}	VDD=5.0V，-40°C ~+85°C	-50%	40	+50%	KHz
外部低频晶振	F_{OSL}	2.0~5.5V	-	32.768	-	KHz

20.4OP 特性

参数	符号	条件	最小值	典型值	最大值	单位
工作电压	Vop	-	2.5	-	5.5	V
工作电流	I_1	VDD=5.0V	-	200	-	μA
输入失调电压	Vos	VDD=5.0V，校准前	-	12	-	mV
		VDD=5.0V，校准后	-	2.5	-	
共模抑制比	CMRR	VDD=5.0V	50	80	-	dB
电源电压抑制比	PSRR		50	70	-	dB
运放增益精度	Ga	VDD=5.0V，输出符合 V_{OR}	-5	-	5	%
		VDD=3.0V，输出符合 V_{OR}	-5	-	5	
输出电压摆幅	V_{OR}	VDD=3.0V/5.0V	GND+0.1	-	VDD-0.1	V
单位增益带宽	GSW	R=1MΩ，Cload=100pF	-	3.64	-	MHz
输入共模电压	VICM		VSS	-	VDD-1.4	V
SR 转换速率		空载，VDD=5V	-	1.4		V/us

20.5 CMP1 特性

参数	符号	条件	最小值	典型值	最大值	单位
工作电压	Vop	Fosc = 8MHz，Fcpu=2MHz	1.9	-	5.5	V
输入失调电压	Vos	VDD=5.0V	-	14	-	mV
响应时间	T_{RS}	正常模式：输出低->高	-	55	-	ns
		正常模式：输出高->低	-	40	-	ns
比较器使能的额外电流	I_{CMP}	25°C，VDD=5V	-	105	-	uA
比较器共模电压范围	V_{CM}	-40°C~85°C，VDD=2.0~5.5V	V_{SS}	-	V_{DD}-1.0	V
共模抑制比	CMRR	-40°C~85°C，VDD=2.0~5.5V	55	-	-	db

20.6 CMP2 特性

参数	符号	条件	最小值	典型值	最大值	单位
工作电压	Vop	Fosc = 8MHz，Fcpu=2MHz	2.2	-	5.5	V
响应时间	T_{RS}	正常模式：输出低->高	-	1	-	us
		正常模式：输出高->低	-	1.5	-	us
比较器使能的额外电流	I_{CMP}	25℃，VDD=5V	-	60	-	uA
比较器共模电压范围	V_{CM}	-40℃~85℃，VDD=2.0~5.5V	V_{SS}	-	V_{DD}-1.0	V
共模抑制比	CMRR	-40℃~85℃，VDD=2.0~5.5V	55	-	-	db

20.7 MTP 特性

参数	符号	条件	最小值	典型值	最大值	单位
读写测试	MN_{ENDUR}	-	10000	-	-	Cycle
数据保存时间	MT_{RET}	T=25℃	10	-	-	year

20.8 EEPROM 特性

参数	符号	条件	最小值	典型值	最大值	单位
读写测试	EN_{ENDUR}	-	10000	-	-	Cycle
数据保存时间	ET_{RET}	T=25℃	10	-	-	year
字节写入电压	EV_{WRITE}	关闭 BOR	1.6	-	5.5	V
字节读取电压	EV_{READ}	关闭 BOR	1.6	-	5.5	V
字节写入时间	ET_{PROG}	1 个字节	-	-	-	us
字节读取时间	ET_{READ}	1 个字节	-	-	-	us

20.9 LCD 电气特性

参数	符号	条件	最小值	典型值	最大值	单位
LCD 偏置电压	VBIAS	1/2 Bias，无负载	-	1/2VDD	-	V
		1/3 Bias，无负载	-	1/3VDD	-	V
LCD 偏置电阻	RLCD	RLCD[1:0]=00	480	600	720	KΩ
		RLCD[1:0]=01	240	300	360	KΩ
		RLCD[1:0]=10	80	100	120	KΩ
		RLCD[1:0]=11	50	60	70	KΩ

20.10 ADC 特性

参数	符号	条件	最小值	典型值	最大值	单位
供电电压	VAD	-	2.3	5.0	5.5	V
精度	NR	GND≤VAIN≤Vref	-	-	12	bit
ADC 输入电压	VAIN	-	GND	-	Vref	V
ADC 输入电阻	RAIN	VAIN=5V	2	-	-	MΩ
模拟电压源推荐阻抗	ZAIN	-	-	-	10	kΩ
ADC 转换电流	IAD	ADC 模块打开，VDD=5.0V	-	0.6	1	mA
ADC 输入电流	IADIN	VDD=5.0V	-	-	10	μA
积分非线性误差（1MHz 转换频率）	ILE	VDD=5.0V，Vref=1.3V，25℃	-4	-	+4	LSB
		VDD=5.0V，Vref =2V，25℃	-3	-	+2	
		VDD=5.0V，Vref =3V，25℃	-3	-	+2	
		VDD=5.0V，Vref=VDD，25℃	-3	-	+3	
微分非线性误差（1MHz 转换频率）	DLE	VDD=5.0V，Vref=1.3V，25℃	-2	-	+3	LSB
		VDD=5.0V，Vref =2V，25℃	-2	-	+3	
		VDD=5.0V，Vref =3V，25℃	-2	-	+3	
		VDD=5.0V，Vref=VDD，25℃	-2	-	+3	
满刻度误差	EF	VDD=5.0V	-5	-	+5	LSB
偏移量误差	EZ	VDD=5.0V	-3	-	+5	LSB
总绝对误差	EAD	VDD=5.0V	-5	-	+5	LSB

20.11 其他特性

1、ESD（HBM）： CLASS 3A（≥4000V）
2、ESD（MM）：CLASS 2（≥200V）
3、Latch_up：CLASS I

21 工具

21.1 MTP 烧录器（BL-PM18 PRO 5.0）

- PM18 PRO ：支持 BL18 系列 MCU 大批量的脱机烧录。

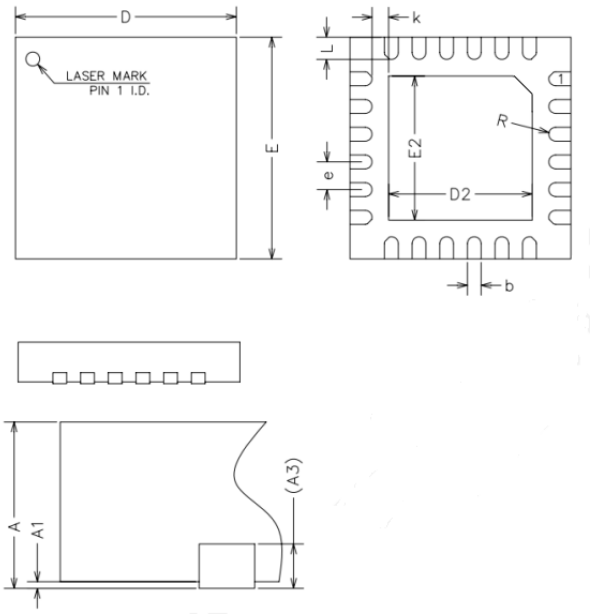
21.2 BL-IDE

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

- BL-IDE：BL-IDE V3.0.x.x(支持汇编/C 语言)

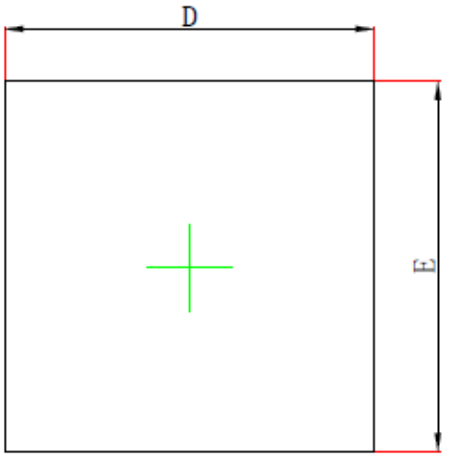
22 封装信息

22.1 QFN24(4*4)

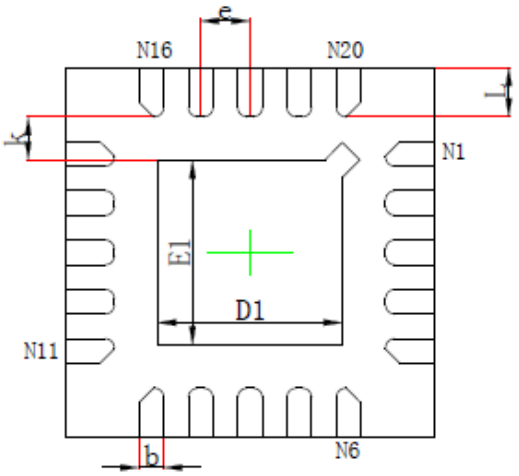


SYMBOL	MIN	NOM	MAX
A	0.70	0.75	0.80
A1	0	0.02	0.05
A3		0.20REF	
b	0.20	0.25	0.30
D	3.90	4.00	4.10
E	3.90	4.00	4.10
D2	2.50	2.60	2.70
E2	2.50	2.60	2.70
e	0.40	0.50	0.60
K	0.20	–	–
L	0.35	0.40	0.45
R	0.09	–	–

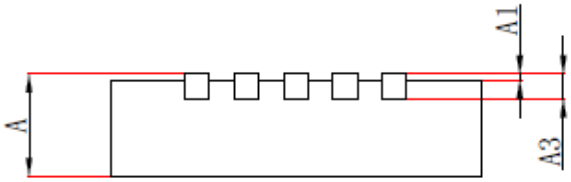
22.2 QFN20



Top View



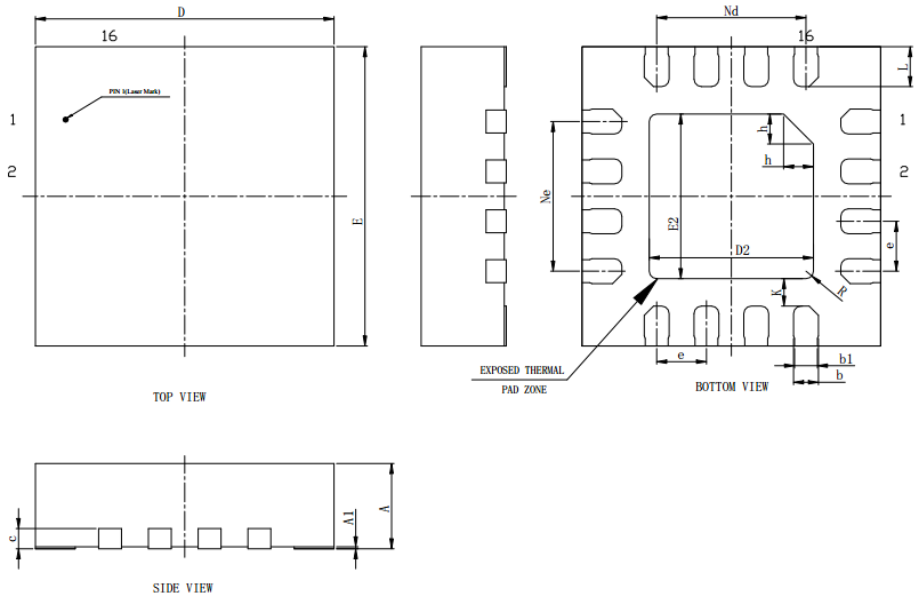
Bottom View



Side View

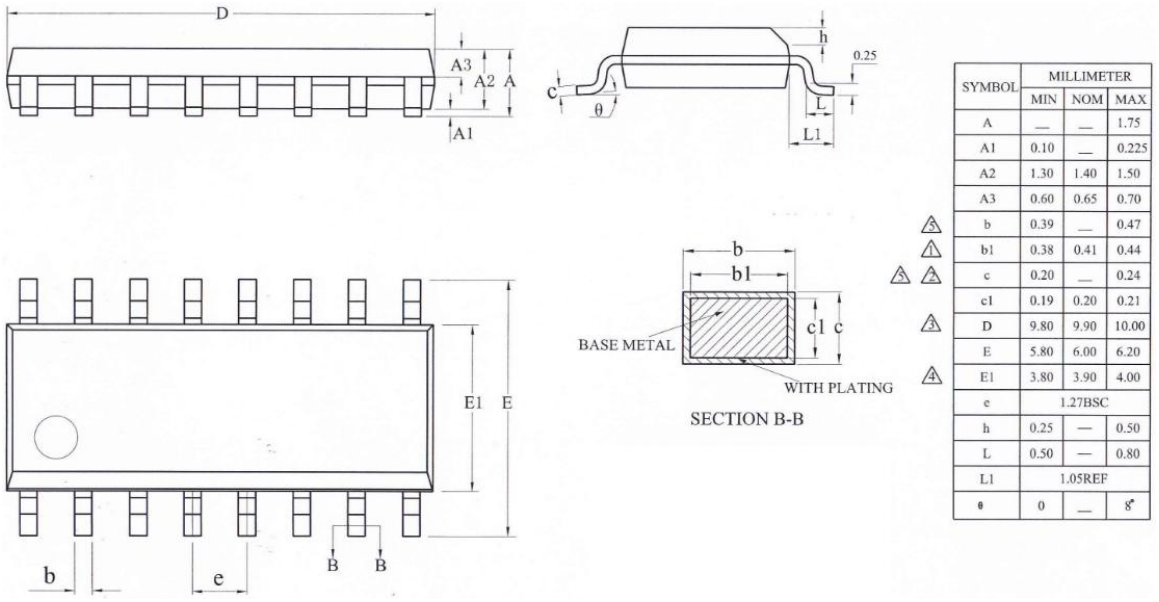
Symbol	Dimensions In Millimeters		Dimensions In Inches	
	Min.	Max.	Min.	Max.
A	0.700/0.800	0.800/0.900	0.028/0.031	0.031/0.035
A1	0.000	0.050	0.000	0.002
A3	0.203REF.		0.008REF.	
D	2.924	3.076	0.115	0.121
E	2.924	3.076	0.115	0.121
D1	1.400	1.600	0.055	0.063
E1	1.400	1.600	0.055	0.063
k	0.200MIN.		0.008MIN.	
b	0.150	0.250	0.006	0.010
e	0.400TYP.		0.016TYP.	
L	0.324	0.476	0.013	0.019

22.3 QFN16



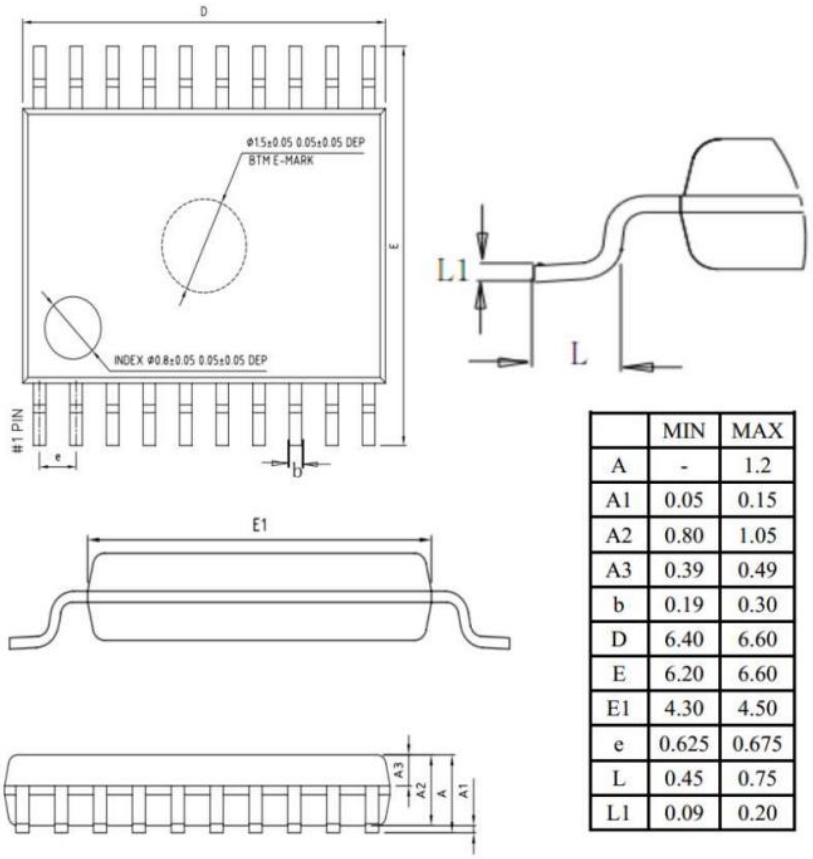
SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	0.70	0.75	0.80
A1	0	0.02	0.05
b	0.20	0.25	0.30
b1		0.23REF	
c		0.203REF	
D	2.90	3.00	3.10
D2	1.60	1.65	1.70
e		0.50BSC	
Nd		1.50BSC	
Ne		1.50BSC	
E	2.90	3.00	3.10
E2	1.60	1.65	1.70
L	0.35	0.40	0.45
h	0.25	0.30	0.35
K	0.225	0.275	0.325
R		0.075REF	

22.4 SOP16



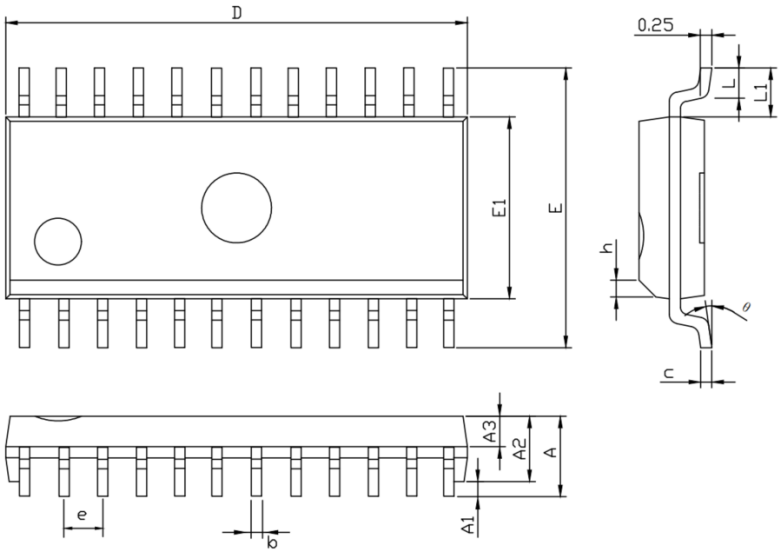
SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	—	—	1.75
A1	0.10	—	0.225
A2	1.30	1.40	1.50
A3	0.60	0.65	0.70
b	0.39	—	0.47
b1	0.38	0.41	0.44
c	0.20	—	0.24
c1	0.19	0.20	0.21
D	9.80	9.90	10.00
E	5.80	6.00	6.20
E1	3.80	3.90	4.00
e	1.27BSC		
h	0.25	—	0.50
L	0.50	—	0.80
L1	1.05REF		
θ	0	—	8°

22.5 TSSOP20



	MIN	MAX
A	-	1.2
A1	0.05	0.15
A2	0.80	1.05
A3	0.39	0.49
b	0.19	0.30
D	6.40	6.60
E	6.20	6.60
E1	4.30	4.50
e	0.625	0.675
L	0.45	0.75
L1	0.09	0.20

22.6 SSOP24



COMMON DIMENSIONS
(UNITS OF MEASURE=MILLIMETER)

SYMBOL	MIN	NOM	MAX
A	–	–	1.75
*A1	0.05	–	0.25
A2	1.35	1.45	1.55
A3	0.65	0.70	0.75
b	0.203	–	0.305
c	0.17	–	0.25
D	8.45	8.60	8.85
*E	5.80	6.00	6.20
*E1	3.80	3.90	4.00
*e	0.615	0.635	0.655
h	0.30	–	0.50
L	0.40	–	0.80
*L1	1.00	1.05	1.10
θ	0°	–	8°

备注：1.标注“*”项为测量尺寸

23 版本修正记录

版本	日期	描述
Ver1.00	2025-01-06	初版
Ver1.01	2025-06-17	1、PWM时钟源改为32MHZ 2、ADC-OFFSET校准ADJTRIM从5bit改为6bit，校准范围-64LSB~+62LSB 3、增加F_{CPU}=8MHZ，最低工作电压2.5V描述 4、修改部分描述错误
Ver1.02	2025-07-17	1、增加ADC-offset校准完成后进行+5补偿 2、更新OP模块中OP放大倍数配置

波领科技

2025 年 06 月