PIC18(L)F25/26K83

采用CAN技术的28引脚低功耗高性能单片机

说明

PIC18(L)FXXK83是面向汽车和工业应用的全功能CAN产品系列。该产品系列提供多个通信外设（例如CAN、SPI、两个I^2C、两个UART、LIN、DMX和DALI），可以处理各种用于智能应用的有线和无线（使用外部模块）通信协议。该系列包括带计算功能的12位ADC（ADC^2）扩展模块，可自动进行信号分析以降低应用的复杂性。如果结合独立于内核的外设集成功能，该系列可实现电机控制、电源、传感器、信号和用户界面应用的相关功能。

内核特性

- C编译器优化的RISC架构
- 工作速度：
  - 时钟工作频率最高64 MHz
  - 最小指令周期为62.5 ns
- 两个直接存储器访问（Direct Memory Access，DMA）控制器：
  - 数据从闪存程序存储器、数据EEPROM或SFR/GPR空间传输到SFR/GPR空间
  - 用户可编程的源和目标大小
  - 硬件和软件触发的数据传输
- 系统总线仲裁器，用户可为扫描器和DMA1/DMA2（相对于主代码和中断执行）配置优先级
- 向量中断功能：
  - 可选高/低优先级
  - 固定中断延时
  - 可编程向量表基址
- 31级深硬件堆栈
- 低电流上电复位（Power-on Reset，POR）
- 可配置的上电延时定时器（Power-up Timer，PWRT）
- 欠压复位（Brown-Out Reset，BOR）
- 低功耗BOR（Low-Power BOR，LPBOR）选项
- 窗口看门狗定时器（Windowed Watchdog Timer，WWDT）：
  - 可变预分频比选择
  - 可变窗口大小选择
  - 可由硬件或软件配置

存储器

- 最大64 KB的闪存程序存储器
- 最大4 KB的数据SRAM存储器
- 最大1 KB的数据EEPROM
- 存储器访问分区（Memory Access Partition，MAP）：
  - 可配置的引导和应用区域大小，具有独立的写保护
- 可编程代码保护
- 器件信息区（Device Information Area，DIA）存储：
  - 惟一ID和器件ID
  - 温度传感器出厂校准数据
  - 固定参考电压校准数据
- 器件配置信息（Device Configuration Information，DCI）存储：
  - 擦除行大小
  - 每行的写锁存器数
  - 用户行数
  - 数据EEPROM存储器大小
  - 引脚数

工作特性

- 工作电压范围：
  - 1.8V至3.6V（PIC18LF25/26K83）
  - 2.3V至5.5V（PIC18F25/26K83）
- 温度范围：
  - 工业级：-40°C至85°C
  - 扩展级：-40°C至125°C

节能功能

- 打盹模式：能够使CPU内核以慢于系统时钟的速度运行
- 空闲模式：能够暂停CPU内核，而内部外设继续工作
- 休眠模式：最低功耗
- 外设模块禁止（Peripheral Module Disable，PMD）：
  - 能够禁止未使用的外设以最大限度地降低功耗

超低功耗（XLP）特性

- 休眠模式：60 nA（1.8V时，典型值）
- 窗口看门狗定时器：720 nA（1.8V时，典型值）
- 辅助振荡器：580 nA（32 kHz时）
- 工作电流：
  - 4 μA（32 kHz、1.8V时，典型值）
  - 45 μA/MHz（1.8V时，典型值）

数字外设

- 三个具有硬件限制定时器（Hardware Limit Timer，HLT）的8位定时器（TMR2/4/6）
  - 硬件监视和故障检测
- 四个16位定时器（TMR0/1/3/5）
- 四个可配置逻辑单元（Configurable Logic Cell，CLC）：
  - 集成组合和顺序逻辑
- 三个互补波形发生器（Complementary Waveform Generator，CWG）：
  - 上升沿和下降沿死区控制
  - 全桥、半桥和单通道驱动
  - 多个信号源
  - 可编程死区
  - 故障关断输入
- 四个捕捉/比较/PWM（Capture/Compare/PWM，CCP）模块
- 四个10位脉宽调制器（Pulse-Width Modulator，PWM）
- 数控振荡器（Numerically Controlled Oscillator，NCO）：
  - 产生真正的线性频率控制信号
  - 高分辨率，使用20位累加器和20位递增值
- 数据信号调制器（Data Signal Modulator，DSM）：
  - 复用两个载波时钟，具有防毛刺功能
  - 每个载波具有多个源
- 具有存储器扫描功能的可编程CRC：
  - 可靠的数据/程序存储器监视，用于故障保护操作（例如B类）
  - 计算程序存储器任何部分的CRC
- 两个UART模块：
  - 两个模块异步，兼容RS-232和RS-485。
  - 其中一个UART模块支持LIN主模式和从模式、DMX模式、DALI装置和器件协议
  - 自动或由用户定时生成间隔周期
  - 兼容DMA
  - 自动校验和
  - 可编程1、1.5和2个停止位
  - 接收到间隔字符时唤醒
- 一个SPI模块：
  - 可配置长度字节
  - 可配置长度数据包
  - 接收不发送选项
  - 发送不接收选项
  - 传输字节计数器
  - 具有2字节FIFO和DMA功能的独立发送和接收缓冲区
- CAN模块：
  - 符合CAN 2.0B Active规范
  - 三种工作模式：传统模式（与现有PIC18CXX8/FXX8 CAN模块兼容）、增强型模式和FIFO模式。
  - 最高1 Mbps的报文比特率
  - 支持DeviceNet™数据字节过滤器
  - 六个可编程接收/发送缓冲区
  - 三个专用发送缓冲区
  - 两个专用接收缓冲区
  - 16个可动态关联的29位完全验收过滤器
  - 3个29位完全验收屏蔽器
  - 自动远程帧处理
  - 高级错误管理功能
- 兼容两个I^2C模块、SMBus和PMBus™：
  - 专用地址、发送和接收缓冲区
  - 带仲裁的总线冲突检测
  - 总线超时检测和处理
  - 多主器件模式
  - 具有2字节FIFO和DMA功能的独立发送和接收缓冲区
  - I^2C、SMBus 2.0和SMBus 3.0，以及1.8V输入电平选择
- 器件I/O端口特性：
  - 25个I/O引脚（PIC18(L)F25K83）
  - 一个只用作输入的引脚
  - 可单独编程的I/O方向、漏极开路、压摆率、弱上拉控制
  - 电平变化中断
  - 三个外部中断引脚
- 外设引脚选择（Peripheral Pin Select，PPS）：
  - 支持数字I/O的引脚映射
- 两个信号测量定时器（Signal Measurement Timer，SMT）：
  - 带有预分频器的24位定时器/计数器

模拟外设

- 带计算功能的模数转换器（ADC^2）：
  - 12位，最多24个外部通道
  - 自动进行后处理
  - 自动对输入信号进行数学函数运算：平均值计算、滤波器计算、过采样和阈值比较
  - 在休眠模式下工作
  - 集成电荷泵，用于改善低电压操作
- 硬件电容分压器（Capacitive Voltage Divider，CVD）：
  - 当需要触摸或接近传感时，自动进行触摸采样并减小软件大小和降低CPU使用率
  - 可调节采样保持电容阵列
  - 两个保护环输出驱动器
- 温度传感器：
  - 内部连接至ADC
  - 可以进行校准以提高精度
- 两个比较器：
  - 低功耗/高速模式
  - 同相输入端施加固定参考电压
  - 比较器输出可从外部访问
- 5位数模转换器（Digital-to-Analog Converter，DAC）：
  - 5位分辨率，轨到轨
  - 正参考电压选择
  - 未缓冲I/O引脚输出
  - 内部连接至ADC和比较器
- 参考电压：
  - 固定参考电压：1.024V、2.048V和4.096V输出电平

灵活的振荡器结构

- 高精度内部振荡器：
  - 可选频率范围：最高64 MHz
  - 校准至±1%（标称值）
- 32 kHz低功耗内部振荡器（LFINTOSC）
- 32 kHz外部晶振（SOCS）
- 外部振荡器模块，具有：
  - x4 PLL，与外部时钟源配合使用
  - 3种晶振/谐振器模式，频率最高为20 MHz
  - 3种外部时钟模式，频率最高为20 MHz
- 故障保护时钟监视器
- 振荡器起振定时器（Oscillator Start-up Timer，OST）：
  - 确保晶振源的稳定性

表1： PIC18(L)FXXK83系列类型

器件	数据手册索引	闪存程序存储器（KB）	数据EEPROM（B）	数据SRAM（字节）	I/O引脚	12位ADC²（通道）	5位DAC	比较器	8位/（带HLT）/16位定时器	窗口看门狗定时器	信号测量定时器（SMT）	CCP/10位PWM	CWG	NCO	CLC	过零检测	直接存储器访问（DMA）	存储器访问分区	向量中断	CAN	支持各个协议的UART	I²C/SPI	外设引脚选择	外设模块禁止	调试(1)
PIC18(L)F25K83	**(A)**	32	1024	2048	25	24	1	2	3/3	Y	Y	4/4	1	1	4	Y	2	Y	Y	Y	2	1/1	Y	Y	I
PIC18(L)F26K83	**(A)**	64	1024	4096	25	24	1	2	3/3	Y	Y	4/4	1	1	4	Y	2	Y	Y	Y	2	1/1	Y	Y	I

注 1： I——芯片上集成了调试功能。

数据手册索引：

A：DS40001943 PIC18(L)F25/26K83数据手册，28引脚

注： 有关其他小型封装可用性和标识的信息，请访问**http://www.microchip.com/packaging**或联系您当地的销售办事处。

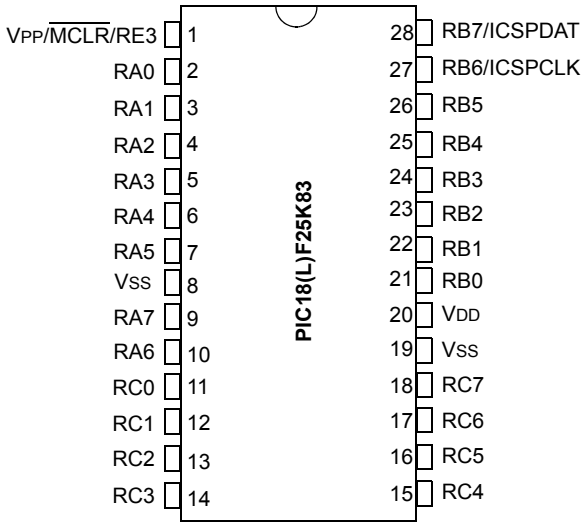
表2：　　封装

器件	SPDIP	SOIC	SSOP	UQFN	QFN
PIC18(L)F25K83	•	•	•	•	•
PIC18(L)F26K83	•	•	•	•	•

注 1： 引脚详细信息可能有所变动。

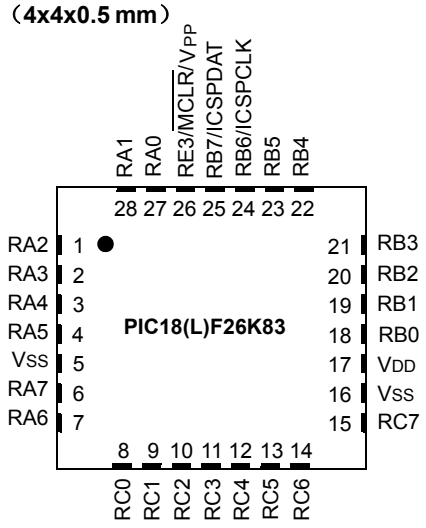
引脚图

28引脚SPDIP、SOIC和SSOP



注： 关于所有外设功能的位置，请参见表3。

28引脚QFN（6x6x0.9 mm）和UQFN（4x4x0.5 mm）



注 1： 关于所有外设功能的位置，请参见表3。

2： 建议将裸露的底部焊盘连接到VSS，但它不能是器件惟一的VSS连接。

引脚分配表

表3： 28引脚分配表（PIC18(L)F25/26K83）

I/O	28引脚SPDIP/SOIC/SSOP	28引脚(U)QFN	ADC	参考电压	DAC	比较器	过零检测	I^2C	SPI	UART	DSM	定时器/SMT	CCP和PWM	CWG	CLC	NCO	参考时钟（CLKR）	ECAN	电平变化中断	基本功能
RA0	2	27	ANA0	—	—	C1IN0- C2IN0-	—	—	—	—	—	—	—	—	CLCIN0(1)	—	—	—	IOCA0	—
RA1	3	28	ANA1	—	—	C1IN1- C2IN1-	—	—	—	—	—	—	—	—	CLCIN1(1)	—	—	—	IOCA1	—
RA2	4	1	ANA2	VREF-	DAC1OUT1	C1IN0+ C2IN0+	—	—	—	—	—	—	—	—	—	—	—	—	IOCA2	—
RA3	5	2	ANA3	VREF+	—	C1IN1+	—	—	—	—	MD1CARL(1)	—	—	—	—	—	—	—	IOCA3	—
RA4	6	3	ANA4	—	—	—	—	—	—	—	MD1CARH(1)	T0CKI(1)	—	—	—	—	—	—	IOCA4	—
RA5	7	4	ANA5	—	—	—	—	—	SS1(1,3)	—	MD1SRC(1)	—	—	—	—	—	—	—	IOCA5	—
RA6	10	7	ANA6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCA6	OSC2 CLKOUT
RA7	9	6	ANA7	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCA7	OSC1 CLKIN
RB0	21	18	ANB0	—	—	C2IN1+	ZCD	—	—	—	—	—	CCP4(1)	CWG1(1)	—	—	—	—	IOCB0 INT0(1)	—
RB1	22	19	ANB1	—	—	C1IN3- C2IN3-	—	SCL2(1,3,4)	—	—	—	—	—	CWG2(1)	—	—	—	—	IOCB1 INT1(1)	—
RB2	23	20	ANB2	—	—	—	—	SDA2(1,3,4)	—	—	—	—	—	CWG3(1)	—	—	—	—	IOCB2 INT2(1)	—
RB3	24	21	ANB3	—	—	C1IN2- C2IN2-	—	—	—	—	—	—	—	—	—	—	—	CANRX(1)	IOCB3	—
RB4	25	22	ANB4 ADACT(1)	—	—	—	—	—	—	—	—	T5G(1) SMT2WIN(1)	—	—	CLCIN2(1)	—	—	—	IOCB4	—
RB5	26	23	ANB5	—	—	—	—	—	—	—	—	T1G(1) SMT2SIG(1)	CCP3(1)	—	CLCIN3(1)	—	—	—	IOCB5	—
RB6	27	24	ANB6	—	—	—	—	—	—	CTS2(1)	—	—	—	—	—	—	—	—	IOCB6	ICSPCLK
RB7	28	25	ANB7	—	DAC1OUT2	—	—	—	—	RX2(1)	—	T6IN(1)	—	—	—	—	—	—	IOCB7	ICSPDAT
RC0	11	8	ANC0	—	—	—	—	—	—	—	—	T1CKI(1) T3CKI(1) T3G(1) SMT1WIN(1)	—	—	—	—	—	—	IOCC0	SOSCO
RC1	12	9	ANC1	—	—	—	—	—	—	—	—	SMT1SIG(1)	CCP2(1)	—	—	—	—	—	IOCC1	SOSCI
RC2	13	10	ANC2	—	—	—	—	—	—	—	—	T5CKI(1)	CCP1(1)	—	—	—	—	—	IOCC2	—
RC3	14	11	ANC3	—	—	—	—	SCL1(1)	SCK1(1,3)	—	—	T2IN(1)	—	—	—	—	—	—	IOCC3	—

注 1： 此信号为PPS可重映射输入信号。输入功能可从图示的默认位置移至其他多个PORTx引脚之一。
2： 此行显示的所有输出信号均为PPS可重映射信号。
3： 此信号为双向信号。为使模块正常工作，固件应将此信号映射到PPS输入和PPS输出寄存器中的同一引脚。
4： 这些引脚可配置为I^2C和SMBus 3.0/2.0逻辑电平；SCLx/SDAx信号可分配给RB1/RB2/RC3/RC4引脚中的任意一个。对其他引脚（如RA5）的PPS分配将起作用，但输入逻辑电平将为通过INLVL寄存器选择的标准TTL/ST，而不是I^2C特定的或SMBUS输入缓冲区阈值。

表3： 28引脚分配表（PIC18(L)F25/26K83）（续）

I/O	28引脚SPDIP/SOIC/SSOP	28引脚(U)QFN	ADC	参考电压	DAC	比较器	过零检测	I^2C	SPI	UART	DSM	定时器/SMT	CCP和PWM	CWG	CLC	NCO	参考时钟（CLKR）	ECAN	电平变化中断	基本功能
RC4	15	12	ANC4	—	—	—	—	SDA1(1)	SDI1(1)	—	—	—	—	—	—	—	—	—	IOCC4	—
RC5	16	13	ANC5	—	—	—	—	—	—	—	—	T4IN(1)	—	—	—	—	—	—	IOCC5	—
RC6	17	14	ANC6	—	—	—	—	—	—	CTS1(1)	—	—	—	—	—	—	—	—	IOCC6	—
RC7	18	15	ANC7	—	—	—	—	—	—	RX1(1)	—	—	—	—	—	—	—	—	IOCC7	—
RE3	1	26	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	IOCE3	MCLR VPP
VDD	20	17	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
VSS	8, 19	5, 16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
OUT(2)	—	—	ADGRDA ADGRDB	—	—	C1OUT C2OUT	—	SDA1 SCL1 SDA2 SCL2	SS1 SCK1 SDO1	DTR1 RTS1 TX1 DTR2 RTS2 TX2	DSM	TMR0	CCP1 CCP2 CCP3 CCP4 PWM5OUT PWM6OUT PWM7OUT PWM8OUT	CWG1A CWG1B CWG1C CWG1D CWG2A CWG2B CWG2C CWG2D CWG3A CWG3B CWG3C CWG3D	CLC1OUT CLC2OUT CLC3OUT CLC4OUT	NCO	CLKR	CANTX	—	—

注 1： 此信号为PPS可重映射输入信号。输入功能可从图示的默认位置移至其他多个PORTx引脚之一。
2： 此行显示的所有输出信号均为PPS可重映射信号。
3： 此信号为双向信号。为使模块正常工作，固件应将此信号映射到PPS输入和PPS输出寄存器中的同一引脚。
4： 这些引脚可配置为I^2C和SMBus 3.0/2.0逻辑电平； SCLx/SDAx信号可分配给RB1/RB2/RC3/RC4引脚中的任意一个。对其他引脚（如RA5）的PPS分配将起作用，但输入逻辑电平将为通过INLVL寄存器选择的标准TTL/ST，而不是I^2C特定的或SMBUS输入缓冲区阈值。

目录

致 客 户

我们旨在提供最佳文档供客户正确使用 Microchip 产品。为此，我们将不断改进出版物的内容和质量，使之更好地满足您的需求。出版物的质量将随新文档及更新版本的推出而得到提升。

如果您对本出版物有任何问题和建议，请通过电子邮件联系我公司 TRC 经理，电子邮件地址为 **CTRC@microchip.com**。我们期待您的反馈。

最新数据手册

欲获得本数据手册的最新版本，请访问我公司网站：

http://www.microchip.com

查看数据手册中任意一页下边角处的文献编号即可确定其版本。文献编号中紧跟数字串后的字母是版本号，例如：DS30000000A_CN 是文档的 A 版本。

勘误表

现有器件可能带有一份勘误表，描述了实际运行与数据手册中记载内容之间存在的细微差异以及建议的变通方法。一旦我们了解到器件 / 文档存在某些差异时，就会发布勘误表。勘误表上将注明其所适用的硅片版本和文件版本。

欲了解某一器件是否存在勘误表，请通过以下方式之一查询：

- Microchip 网站 **http://www.microchip.com**
- 当地 Microchip 销售办事处 （见最后一页）

在联络销售办事处时，请说明您所使用的器件型号、硅片版本和数据手册版本 （包括文献编号）。

客户通知系统

欲及时获知 Microchip 产品的最新信息，请到我公司网站 **www.microchip.com** 上注册。

1.0 器件概述

本文档包含以下器件的特定信息：

- PIC18F25K83
- PIC18LF25K83
- PIC18F26K83
- PIC18LF26K83

该系列具有所有PIC18单片机的优势（即能够以经济的价格实现强大的计算功能），附带高耐用性闪存程序存储器、通用异步收发器（Universal Asynchronous Receiver Transmitter，UART）、串行外设接口（Serial Peripheral Interface，SPI）、I^2C、直接存储器访问（DMA）、可配置逻辑单元（CLC）、信号测量定时器（SMT）、数控振荡器（NCO）和带计算功能的模数转换器（ADC^2）。

1.1 新特性

- **直接存储器访问控制器：**直接存储器访问（DMA）控制器旨在直接处理不同存储区域之间的数据传输，而无需CPU干预。由于无需对数据传输用中断的处理过程进行CPU密集型管理，CPU现在可以有更多时间处理其他任务。
- **向量中断控制器：**向量中断控制器模块将诸多外设中断请求信号缩减为一个送往CPU的中断请求信号。它汇集所有中断请求信号，并根据固定的自然顺序优先级和用户分配的优先级解决中断问题，从而无需扫描中断源。
- **通用异步收发器：**通用异步收发器（UART）模块是串行I/O通信外设。它包含用来完成与器件程序执行独立的输入或输出串行数据传输所需的所有时钟发生器、移位寄存器和数据缓冲区。UART也可称为串行通信接口（Serial Communication Interface，SCI），可配置为全双工异步系统或多种自动协议之一。全双工模式可通过DMA/DALI/LIN支持与外设系统通信。
- **串行外设接口：**串行外设接口（SPI）模块是以全双工模式工作的同步串行数据通信总线。器件在由主器件启动通信的主/从器件环境中进行通信。从器件通过称为从选择的片选进行控制。示例从器件包括串行EEPROM、移位寄存器、显示驱动器、A/D转换器或其他PIC。
- **I^2C 模块：**I^2C模块使用双线I^2C串行总线在单片机和其他I^2C兼容器件之间提供同步接口。器件在主/从器件环境中进行通信。I^2C总线指定两个信号连接——串行时钟（SCL）和串行数据（SDA）。SCL和SDA连接都是双向的漏极开路线，它们都需要通过上拉电阻连接到供电电压。
- **带计算功能的12位A/D转换器：**此模块具有可编程的采集时间，无需等待采样周期便可选择通道并启动转换，因此可降低代码开销。它包含一个名为ADC^2的新模块，该模块具有计算功能，并且可提供数字滤波器和阈值中断功能。

1.2 各系列成员的详细信息

PIC18(L)F25/26K83系列器件采用28引脚封装。器件框图如图3-1所示。

PIC18(L)F25/26K83系列类型表（第4页）中列出了这些器件之间的异同。表3列出了所有器件的引脚分配。

表1-1： 器件特性

特性	PIC18(L)F25K83	PIC18(L)F26K83
程序存储器（字节）	32768	65536
程序存储器（指令）	16384	32768
数据存储器（字节）	2048	4096
数据EEPROM存储器（字节）	1024	1024
封装	28引脚SPDIP 28引脚SOIC 28引脚SSOP 28引脚QFN 28引脚UQFN	28引脚SPDIP 28引脚SOIC 28引脚SSOP 28引脚QFN 28引脚UQFN
I/O端口	A、B、C、E(1)	A、B、C、E(1)
带计算加速器的12位模数转换模块（ADC2）	5个内部 24个外部	5个内部 24个外部
捕捉/比较/PWM（CCP）模块	4	
10位脉宽调制器（PWM）	4	
定时器（16位/8位）	4/3	
串行通信	2个UART（支持DMX/DALI/LIN）、2个I^2C和1个SPI	
互补波形发生器（CWG）	3	
过零检测（Zero-Cross Detect，ZCD）	1	
数据信号调制器（DSM）	1	
信号测量定时器（SMT）	2	
5位数模转换器（DAC）	1	
数控振荡器（NCO）	1	
比较器模块	2	
直接存储器访问（DMA）	2	
可配置逻辑单元（CLC）	4	
控制局域网（Control Area Network，CAN）	有	
外设模块禁止（PMD）	有	
带扫描器的16位CRC	有	
可编程高/低电压检测（HLVD）	有	
复位（和延时）	POR、可编程BOR、 RESET指令、 堆栈上溢、 堆栈下溢 （PWRT和OST）、 $\overline{MCLR}$、WDT和$\overline{MEMV}$	
指令集	81条指令； 87条指令（使能扩展指令集时）	
最大工作频率	64 MHz	

注 1： PORTE包含一个只用作输入的RE3引脚。

1.3 寄存器和位命名约定

1.3.1 寄存器名称

当器件中的同一外设有多个实例时，外设控制寄存器将以外设标识符、外设实例和控制标识符串接的形式表示。控制寄存器部分将只显示所有寄存器名称的一个实例，用“x”代替外设实例号。当器件中的某外设只有一个实例时，可能也会对该外设采用这种命名约定，以便与同系列中包含多个实例的其他器件保持兼容。

1.3.2 位名称

位名称有两种形式：

- 短名称：位功能缩写
- 长名称：外设缩写 + 短名称

1.3.2.1 短位名称

短位名称是位功能的缩写。例如，一些外设使用EN位进行使能。寄存器中显示的位名称是短名称形式。

短位名称可用于访问C程序中的位。通过短名称访问位的一般形式为*RegisterName*bits.*ShortName*。例如，T0CON0寄存器中的使能位EN可以在C程序中用指令`T0CON0bits.EN = 1`来置1。

通常不在汇编程序中使用短名称，因为不同外设可能在不同位位置使用相同名称。发生这种情况时，在包含文件生成期间，会在该短位名称的所有实例后附加一个下划线及该位所在寄存器的名称，以避免命名争用。

1.3.2.2 长位名称

长位名称的构造方法是在短名称前加上外设缩写前缀。该前缀对外设而言是惟一的，因此可使每个长位名称也惟一。Timer0使能位的长位名称为Timer0前缀T0后附加使能位短名称EN，从而得到惟一位名称T0EN。

在C程序和汇编程序中都可以使用长位名称。例如，在C程序中，可以使用`T0EN = 1`指令将T0CON0使能位置1。在汇编程序中，可以使用`BSF T0CON0,T0EN`指令将该位置1。

1.3.2.3 位字段

位字段是同一个寄存器中的两个或更多相邻位。例如，T0CON0寄存器的低四位包含输出预分频比选择位。该位字段的短名称为OUTPS，长名称为T0OUTPS。只能在C程序中进行位字段访问。以下示例给出了将Timer0输出预分频器设置为1:6后分频比的C程序指令：

```
T0CON0bits.OUTPS = 0x5;
```

此外，还可以使用长位名称和短位名称来访问位字段中的各个位。每个位的名称为位字段名称后附加位字段内的位位置编号。例如，最高有效模式位的短位名称为OUTPS3。以下两个示例给出了将Timer0输出预分频器设置为1:6后分频比的汇编程序序列：

示例1：

```
MOVLW  ~(1<<OUTPS3 | 1<<OUTPS1)
ANDWF  T0CON0,F
MOVLW  1<<OUTPS2 | 1<<OUTPS0
IORWF  T0CON0,F
```

示例2：

```
BCF    T0CON0,OUTPS3
BSF    T0CON0,OUTPS2
BCF    T0CON0,OUTPS1
BSF    T0CON0,OUTPS0
```

1.3.3 寄存器和位命名例外情况

1.3.3.1 状态、中断和镜像位

状态、中断允许、中断标志和镜像位包含在跨多个外设的寄存器中。在这些情况下，所显示的位名称是惟一的，因此没有前缀和短名称形式。

2.0 PIC18(L)F25/26K83单片机入门指南

2.1 基本连接要求

在开始使用PIC18(L)F25/26K83系列8位单片机进行开发之前，需要注意最低限度的器件引脚连接要求。

必须始终连接以下引脚：

- 所有VDD和VSS引脚（见**第2.2节“电源引脚”**）
- $\overline{\text{MCLR}}$引脚（见**第2.3节“主复位（MCLR）引脚”**）

如果最终应用中使用以下引脚，则还必须连接这些引脚：

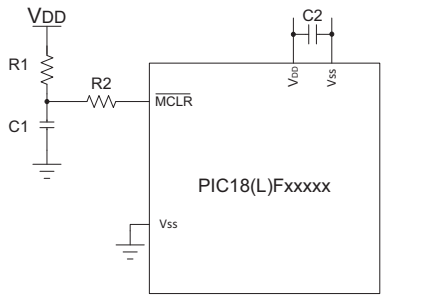
- 用于在线串行编程（In-Circuit Serial Programming™，ICSP™）和调试目的的ICSPCLK/ICSPDAT引脚（见**第2.4节“ICSP™引脚”**）
- OSCI和OSCO引脚（使用外部振荡源时）（见**第2.5节“外部振荡器引脚”**）

另外，可能需要以下引脚：

- 实现模拟模块的外部参考电压时使用VREF+/VREF-引脚。

最基本连接如图2-1所示。

图2-1： 建议的最基本连接



要点（所有值均为建议值）：
C1和C2：0.1 µF、20V陶瓷电容
R1：10 kΩ
R2：100Ω至470Ω

2.2 电源引脚

2.2.1 去耦电容

需要在每对电源引脚（VDD和VSS）上使用去耦电容。

使用去耦电容时，需要考虑以下标准：

- **电容的类型和电容值：**建议使用0.1 µF（100 nF）、10-20V的电容。该电容应为低ESR器件，谐振频率为200 MHz和更高值。建议使用陶瓷电容。
- **在印刷电路板上的放置：**去耦电容应尽可能靠近引脚。建议将电容与器件放置在电路板的同一层。如果空间有限，可使用过孔将电容放到PCB的其他层上；但是，需要确保从引脚到电容的走线长度在0.25英寸（6 mm）内。
- **高频噪声处理：**如果电路板会受到超过数十MHz的高频噪声影响，应为上述去耦电容并联一个陶瓷电容。第二个电容的值可介于0.001 µF至0.01 µF之间。请将第二个电容放在靠近主去耦电容的位置。在高速电路设计中，应在尽量靠近电源引脚和接地引脚的地方放置两个相差几个数量级的电容（例如，0.1 µF与0.001 µF的电容并联）。
- **最大程度提高性能：**对于从电源电路开始的电路板布线，需要将电源和返回走线先连接到去耦电容，然后再与器件引脚连接。这可以确保去耦电容在电源链的最前面。保持电容和电源引脚之间的走线长度尽可能短也同样重要，因为这可以减少PCB走线间的互感。

2.2.2 大容量电容

对于电源走线长度超出6英寸的电路板，建议对集成电路（包括单片机）使用大容量电容来提供本地电源。大容量电容的电容值应根据连接电源与器件的走线的电阻和应用中器件消耗的最大电流确定。也就是说，选择的大容量电容需要满足器件的可接受电压骤降要求。典型值范围为4.7 µF至47 µF。

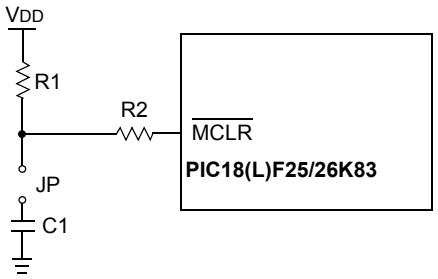
2.3 主复位（$\overline{\text{MCLR}}$）引脚

$\overline{\text{MCLR}}$引脚提供两种特定的器件功能：器件复位以及器件编程和调试。如果最终应用中无需进行编程和调试，则只需将该引脚直接连接至VDD即可。添加其他元件可能有助于应用更好地抵御因电压骤降而引起的虚假复位。典型配置如图2-1所示。可根据应用需求实现其他电路设计。

在编程和调试过程中，必须考虑到引脚上可能会增加的电阻和电容。器件编程器和调试器会驱动$\overline{\text{MCLR}}$引脚。因此，不会对特定的电压（VIH和VIL）和快速信号跳变造成不良影响。所以，需要根据应用和PCB需求来调整R1和C1的具体值。例如，在编程和调试操作期间，建议使用跳线将电容C1与$\overline{\text{MCLR}}$引脚隔离（图2-2）。正常运行操作期间，应将跳线还原。

任何与$\overline{\text{MCLR}}$引脚相关的元件均应放置在距离引脚0.25英寸（6 mm）的范围内。

图2-2： $\overline{\text{MCLR}}$引脚连接示例



注 1： 建议R1 ≤ 10 kΩ。建议的起始值为10 kΩ。请确保满足$\overline{\text{MCLR}}$引脚VIH和VIL规范。

2： R2 ≤ 470Ω用于限制由于静电放电（Electrostatic Discharge，ESD）或电过载（Electrical Overstress，EOS）导致$\overline{\text{MCLR}}$引脚损坏时从外部电容C1流入$\overline{\text{MCLR}}$的任何电流。请确保满足$\overline{\text{MCLR}}$引脚VIH和VIL规范。

2.4 ICSP™引脚

ICSPCLK和ICSPDAT引脚用于进行在线串行编程（ICSP™）和调试。建议尽可能缩短ICSP连接器与器件ICSP引脚之间的走线长度。如果ICSP连接器会遇到ESD事件，则建议添加一个串联电阻，电阻值为几十Ω，不要超出100Ω。

建议不要在ICSPCLK和ICSPDAT引脚上连接上拉电阻、串联二极管和电容，因为它们会影响编程器/调试器与器件之间的通信。如果应用需要此类分立元件，则在编程和调试期间应将它们从电路中去除。或者，请参见相应器件闪存编程规范中的交流/直流特性与时序要求信息，了解关于容性负载限制、引脚输入高电压（VIH）和输入低电压（VIL）要求的信息。

对于器件仿真，请确保编程到器件中的“通信通道选择”（即ICSPCLK/ICSPDAT引脚）符合与Microchip调试器/仿真器工具的ICSP物理连接。

关于可用Microchip开发工具连接要求的更多信息，请参见**第44.0节“开发支持”**。

2.5 外部振荡器引脚

许多单片机至少提供了两个振荡器供选用：高频主振荡器和低频辅助振荡器（详细信息见**第7.0节“振荡器模块（带故障保护时钟监视器）”**）。

振荡器电路与器件应放置在电路板的同一层。请将振荡器电路放置在靠近相应振荡器引脚的位置，电路元件和引脚之间的距离不要超出0.5英寸（12 mm）。负载电容应靠近振荡器本身，位于电路板的同一层。

请在振荡器电路周围使用接地覆铜区，以将其与周围电路隔离。接地覆铜区应与MCU地直接连接。不要在接地覆铜区内安排任何信号走线或电源走线。此外，如果使用双面电路板，请避免在电路板上的另一面（晶振所在的一面）有任何走线。

布局建议如图2-3所示。直插封装可通过完全包围振荡器引脚的单面布局进行处理。对于紧密排列封装，并不总是能够完全包围引脚和元件。合适的解决方案是将断开的保护部分连接到镜像接地层。在所有情况下，保护走线都必须返回接地端。

规划应用布线和I/O分配时，应确保相邻端口引脚及其他靠近振荡器的信号不会产生干扰（即不存在上升和下降时间较短的高频噪声以及其他类似噪声）。

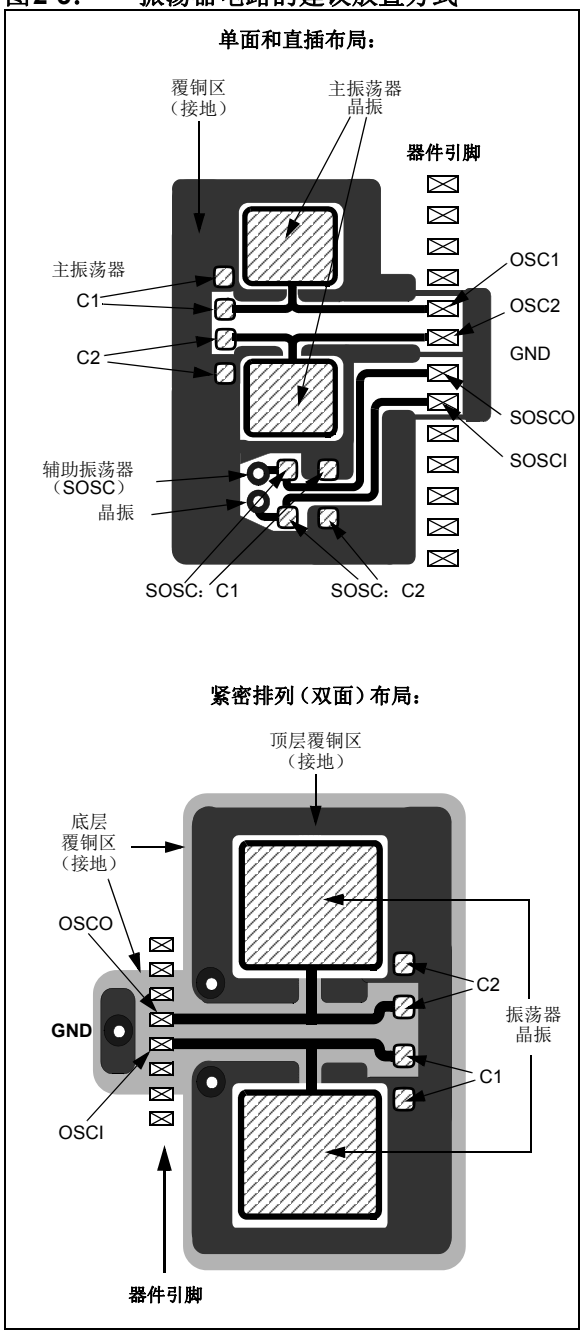
有关振荡器电路的更多信息和设计指南，请参见Microchip公司网站（www.microchip.com）上提供的以下应用笔记：

- AN826，“*Crystal Oscillator Basics and Crystal Selection for rfPIC™ and PICmicro® Devices*”
- AN849，“*Basic PICmicro® Oscillator Design*”
- AN943，“*Practical PICmicro® Oscillator Analysis and Design*”
- AN949，“*Making Your Oscillator Work*”

2.6 未用I/O

未用I/O引脚应配置为输出，并驱动为逻辑低电平状态。或者，将一个1 kΩ至10 kΩ的电阻连接到未用引脚上的Vss，并将输出驱动为逻辑低电平。

图2-3： 振荡器电路的建议放置方式

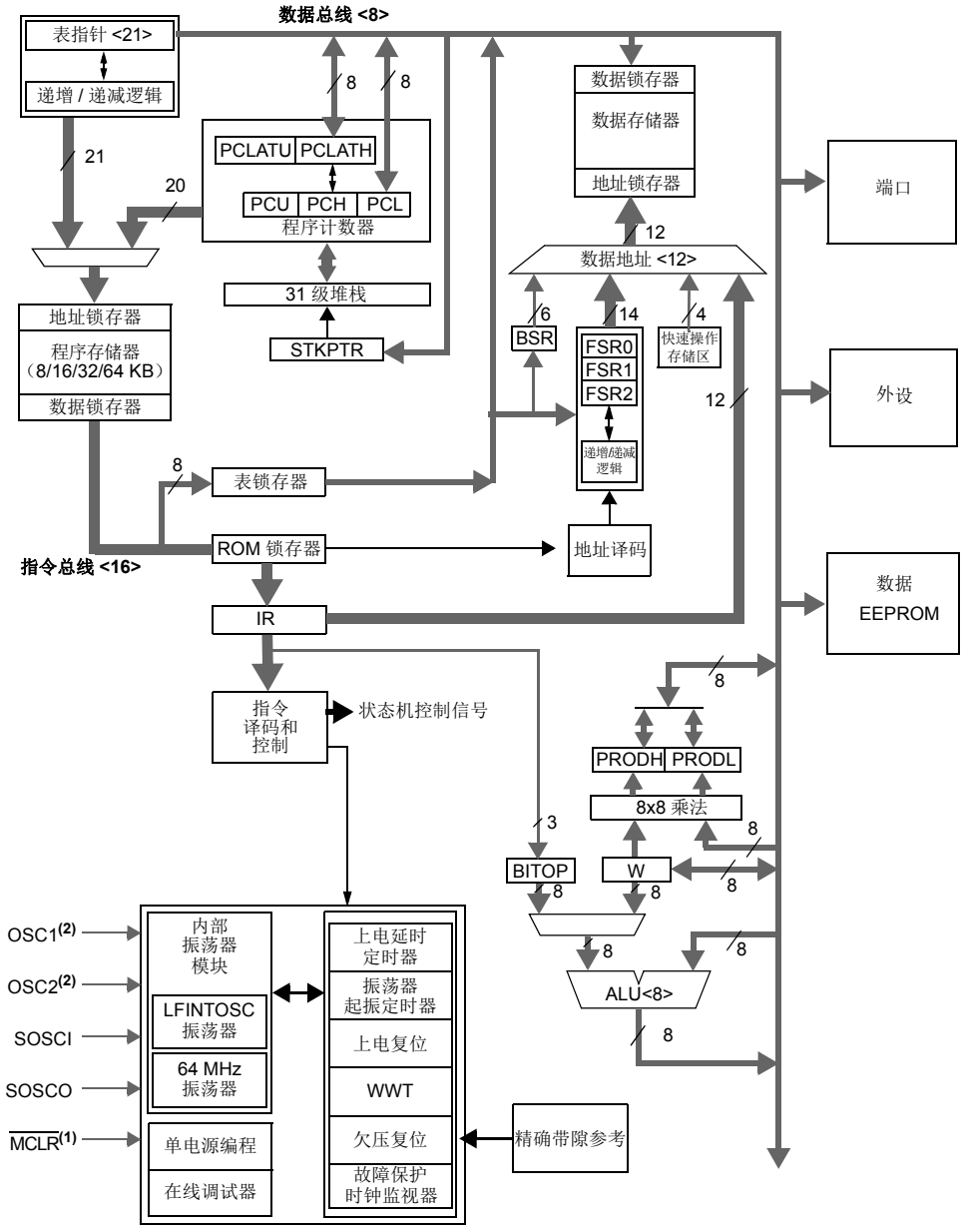


3.0 PIC18 CPU

本器件系列包含基于改进型哈佛架构的PIC18 8位CPU内核。PIC18 CPU支持：

- 系统仲裁，根据用户优先级决定存储器访问分配
- 向量中断功能，支持自动的两级深现场保护
- 31级深硬件堆栈，具有上溢和下溢复位功能
- 支持直接、间接和相对寻址模式
- 8x8硬件乘法器

图3-1： PIC18(L)F25/26K83系列框图



注 1： RE3仅在$\overline{MCLR}$功能禁止时可用。

2： OSC1/CLKIN和OSC2/CLKOUT仅在选择振荡器模式下以及这些引脚不用作数字I/O时可用。更多信息，请参见**第7.0节“振荡器模块（带故障保护时钟监视器）”**。

3.1 系统仲裁

系统仲裁器根据用户分配的优先级来解析系统级选择（即主程序和中断服务程序）和外设选择（即DMA和扫描器）之间的存储器访问优先级。每个系统级和外设选择都有自己的优先级选择寄存器。使用写入相应优先级寄存器的编号来解析存储器访问优先级，0表示最高优先级，4表示最低优先级。表3-1列出了默认优先级。

如果用户要更改优先级，请确保写入每个优先级寄存器的都是0到4之间的惟一值。

表3-1： 默认优先级

选择		优先级寄存器复位值
系统级	ISR	0
	主程序	1
外设	DMA1	2
	DMA2	3
	扫描器	4

3.1.1 优先级锁定

当PRLOCKED位（PRLOCK寄存器）置1时，系统仲裁器会将存储器访问权授予外设选择（DMAx和扫描器）。

可通过将PRLOCK寄存器的PRLOCKED位置1锁定优先级选择。置1和清零该位需要一个特殊序列作为额外的预防措施，以防止意外的更改。例3-1和例3-2给出了置1和清零PRLOCKED位的示例。

例3-1： 优先级锁定序列

```
; Disable interrupts
BCF INTCON0,GIE

; Bank to PRLOCK register
BANKSEL PRLOCK
MOVLW 55h

; Required sequence, next 4
instructions
MOVWF PRLOCK
MOVLW AAh
MOVWF PRLOCK
; Set PRLOCKED bit to grant memory
access to peripherals
BSF PRLOCK,0

; Enable Interrupts
BSF INTCON0,GIE
```

例3-2： 优先级解锁序列

```
; Disable interrupts
BCF INTCON0,GIE

; Bank to PRLOCK register
BANKSEL PRLOCK
MOVLW 55h

; Required sequence, next 4
instructions
MOVWF PRLOCK
MOVLW AAh
MOVWF PRLOCK
; Clear PRLOCKED bit to allow changing
priority settings
BCF PRLOCK,0

; Enable Interrupts
BSF INTCON0,GIE
```

3.2 存储器访问机制

用户可根据系统仲裁器将存储器访问权授予系统级选择还是外设选择来为二者分配优先级。我们来看一下ISR、主程序和外设之间的以下几种优先级情况。

注： ISR优先级始终需高于主程序优先级。

3.2.1 ISR优先级 > 主程序优先级 > 外设优先级

当外设优先级（DMAx和扫描器）低于ISR和主程序优先级时，外设需要：

1. 访问闪存程序存储器，然后外设会等待一个指令周期（例如转移指令，在此期间CPU不需要访问PFM），并在该周期内访问闪存程序存储器（正在执行PFM读/写操作的情况除外）。
2. 访问SFR/GPR，然后外设会等待一个指令周期（例如MOVLW、CALL和NOP，在此期间CPU不需要访问SFR/GPR），并在该周期内访问SFR/GPR。
3. 访问数据EEPROM，然后外设可以访问数据EEPROM（正在执行数据EEPROM读/写操作的情况除外）。

在这种情况下，外设将以最低的吞吐量访问存储器，但不会对执行时间产生任何影响。

3.2.2 外设优先级 > ISR优先级 > 主程序优先级

如果外设优先级（DMAx和扫描器）高于ISR和主程序优先级，则在外设请求访问存储器时会停止CPU操作。

在外设完成操作之前，CPU会一直保持当前状态。由于外设请求访问总线，因此在外设完成操作之前无法将其禁止。

在这种情况下，外设将以最高的吞吐量访问存储器，但代价是需要停止执行其他操作。

3.2.3 ISR优先级 > 外设优先级 > 主程序优先级

在这种情况下，中断程序和外设操作（DMAx和扫描器）会使CPU停止。中断优先于外设操作。在这种情况下，中断延时最低，外设以最高吞吐量访问存储器。

3.2.4 外设1优先级 > ISR优先级 > 主程序优先级 > 外设2优先级

在这种情况下，外设1操作会使CPU停止执行操作。但是，外设2可在外设1未使用存储器时访问存储器。

系统仲裁器的操作由以下寄存器控制：

寄存器3-1： ISRPR：中断服务程序优先级寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	ISRPR<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7-3 **未实现：**读为0

bit 2-0 **ISRPR<2:0>：**中断服务程序优先级选择位

寄存器3-2： MAINPR：主程序优先级寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-1/1
—	—	—	—	—	MAINPR<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7-3 **未实现：**读为0

bit 2-0 **MAINPR<2:0>：**主程序优先级选择位

寄存器3-3： DMA1PR：DMA1优先级寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-1/1	R/W-0/0
—	—	—	—	—	DMA1PR<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7-3 **未实现：**读为0

bit 2-0 **DMA1PR<2:0>：**DMA1优先级选择位

寄存器3-4： DMA2PR：DMA2优先级寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-1/1	R/W-1/1
—	—	—	—	—	DMA2PR<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7-3 **未实现：**读为0

bit 2-0 **DMA2PR<2:0>：**DMA2优先级选择位

寄存器3-5： SCANPR：扫描器优先级寄存器

U-0	U-0	U-0	U-0	U-0	R/W-1/1	R/W-0/0	R/W-0/0
—	—	—	—	—	SCANPR<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7-3 **未实现：**读为0

bit 2-0 **SCANPR<2:0>：**扫描器优先级选择位

寄存器3-6： PRLOCK：优先级锁定寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	PRLOCKED
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7-1 **未实现：**读为0

bit 0 **PRLOCKED：**PR寄存器锁定位**(1, 2)**
0 = 可通过写操作修改优先级寄存器；外设无法访问存储器
1 = 优先级寄存器锁定，无法写入；外设无法访问存储器

注 1： 只能在解锁序列后置1或清零PRLOCKED位。

2： 如果PR1WAY = 1，则PRLOCKED位无法在置1后清零。系统复位将清零该位，并允许该位再次置1。

表3-2： 与CPU相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
ISRPR	—	—	—	—	—	ISRPR2	ISRPR1	ISRPR0	20
MAINPR	—	—	—	—	—	MAINPR2	MAINPR1	MAINPR0	20
DMA1PR	—	—	—	—	—	DMA1PR2	DMA1PR1	DMA1PR0	20
DMA2PR	—	—	—	—	—	DMA2PR2	DMA2PR1	DMA2PR0	21
SCANPR	—	—	—	—	—	SCANPR2	SCANPR1	SCANPR0	21
PRLOCK	—	—	—	—	—	—	—	PRLOCKED	21

图注： — = 未实现位，读为0。

4.0 存储器构成

PIC18增强型单片机器件有三种类型的存储器：

- 闪存程序存储器
- 数据RAM
- 数据EEPROM

闪存程序存储器和数据RAM共用同一总线，而数据EEPROM使用单独总线。这允许并发访问存储空间。

有关闪存程序存储器和数据EEPROM存储器操作的更多详细信息，请参见**第13.0节“非易失性存储器（NVM）控制”**。

4.1 闪存程序存储器构成

PIC18单片机实现了一个21位程序计数器，能够对2 MB程序存储空间进行寻址。访问所有未实现的存储器都将返回全0（NOP指令）。

这些器件包括：

- PIC18(L)F25K83：32 KB闪存存储器，最多16,384条单字指令
- PIC18(L)F26K83：64 KB闪存存储器，最多32,768条单字指令

器件的复位向量位于地址000000h。PIC18(L)F25/26K83器件具有向量中断控制器，该控制器的专用中断向量表位于程序存储器中，请参见**第9.0节“中断控制器”**。

注：	有关本器件系列的存储器信息，请参见表4-1和表4-3。

4.2 存储器访问分区（MAP）

闪存程序存储器可分为：

- 应用程序块
- 引导块
- 存储区闪存（Storage Area Flash，SAF）块

4.2.1 应用程序块

默认情况下，应用程序块是存放用户程序的位置。配置位的默认设置（$\overline{\text{BBEN}}$ = 1且$\overline{\text{SAFEN}}$ = 1）将闪存程序存储器区域中的所有存储器分配到应用程序块。$\overline{\text{WRTAPP}}$配置位用于保护应用程序块。

4.2.2 引导块

引导块是程序存储器中的一个区域，非常适合存储自举程序代码。放置在此区域中的代码可由CPU执行。引导块可以受写保护，与主应用程序块无关。引导块通过$\overline{\text{BBEN}}$位来使能，其大小基于配置字（寄存器5-7）的BBSIZE位的值。有关引导块大小的信息，请参见表5-1。
$\overline{\text{WRTB}}$配置位用于对引导块进行写保护。

4.2.3 存储区闪存

存储区闪存（SAF）是程序存储器中可用来存储数据的区域。SAF通过配置字（寄存器5-7）的$\overline{\text{SAFEN}}$位来使能。使能后，放置在此区域中的代码无法通过CPU执行。SAF块位于存储器的末尾，大小为256个字节。
$\overline{\text{WRTSAF}}$配置位用于对存储区闪存进行写保护。

注：	如果从NVMCON寄存器写入受写保护的单元，则存储器不发生更改并且寄存器13-1中定义的WRERR位置1。

表4-1： 程序存储器和数据EEPROM存储器映射

注 1： 堆栈是一个独立于所有用户存储区的SRAM区域。
2： 00 0008h用作IVTBASE寄存器的默认复位单元，可以通过编程IVTBASE寄存器来重定位存储器中的向量表。
3： 存储区闪存以用户闪存最后128个字的形式实现（如果存在）。
4： 地址不会计满返回。该区域读为0。
5： 不受代码保护。
6： 在芯片中硬编码。
7： 该区域无法由用户写入，并且不受批量擦除的影响。

表4-2：　闪存程序存储器分区

区域	地址	分区(3)			
		$\overline{\text{BBEN}}$ = 1 SAFEN = 1	$\overline{\text{BBEN}}$ = 1 SAFEN = 0	$\overline{\text{BBEN}}$ = 0 SAFEN = 1	$\overline{\text{BBEN}}$ = 0 SAFEN = 0
闪存程序存储器	00 0000h • • • 引导块存储器的最后一个地址	应用程序块	应用程序块	引导块	引导块
	引导块存储器的最后一个地址(1) + 1 • • • 程序存储器的最后一个地址(2) – 100h			应用程序块	应用程序块
	程序存储器的最后一个地址(2) – FEh(4) • • • 程序存储器的最后一个地址(2)		存储区闪存		存储区闪存

注 1： 引导块存储器的最后一个地址基于BBSIZE<2:0>，请参见表5-1。
2： 有关程序存储器的最后一个地址，请参见表5-1。
3： 请参见寄存器5-7：配置字4L来了解$\overline{\text{BBEN}}$和$\overline{\text{SAFEN}}$的定义。
4： 存储区闪存以用户闪存最后128个字的形式实现（如果存在）。

4.2.4 程序计数器

程序计数器（Program Counter，PC）指定欲取出执行的指令的地址。PC为21位宽，保存在三个单独的8位寄存器中。低字节（即PCL寄存器）可读写。高字节（即PCH寄存器）包含PC<15:8>位；不可直接读写。通过PCLATH寄存器更新PCH寄存器。最高字节称为PCU。该寄存器包含PC<20:16>位；也不可直接读写。通过PCLATU寄存器更新PCU寄存器。

PCLATH和PCLATU的内容通过写PCL的操作传送到程序计数器。同样，程序计数器的两个高字节通过读PCL的操作传送到PCLATH和PCLATU。这对于PC计算偏移量很有用处（见**第4.3.2.1节“计算GOTO”**）。

PC按字节寻址程序存储器。为防止PC与字指令不对齐，需要将PCL的最低有效位固定取值为0。PC通过递增2来寻址程序存储器中的顺序指令。

CALL、RCALL、GOTO和程序跳转指令直接写入程序计数器。对于这些指令，PCLATH和PCLATU的内容不会传送到程序计数器。

4.2.5 返回地址堆栈

返回地址堆栈允许出现最多31个程序调用和中断的任意组合。当执行CALL或RCALL指令或应答中断时，PC值将被压入堆栈。当执行RETURN、RETLW或RETFIE指令时，PC值将弹出堆栈。PCLATU和PCLATH不受任何RETURN或CALL指令的影响。

通过21位的RAM和一个5位的堆栈指针来实现31字的堆栈操作。堆栈既不占用程序存储空间，也不占用数据存储空间。堆栈指针是可读写的，栈顶的地址可通过栈顶（Top-of-Stack，TOS）特殊文件寄存器进行读写。还可以使用这些寄存器将数据压入或弹出堆栈。

执行CALL、CALLW或RCALL指令进行压栈操作；堆栈指针先递增，随后将PC（已指向CALL后面的下一条指令）的内容写入堆栈指针所指向的地址单元。执行RETURN型指令进行出栈操作；STKPTR所指向的地址单元的内容会被传送给PC，然后堆栈指针递减。

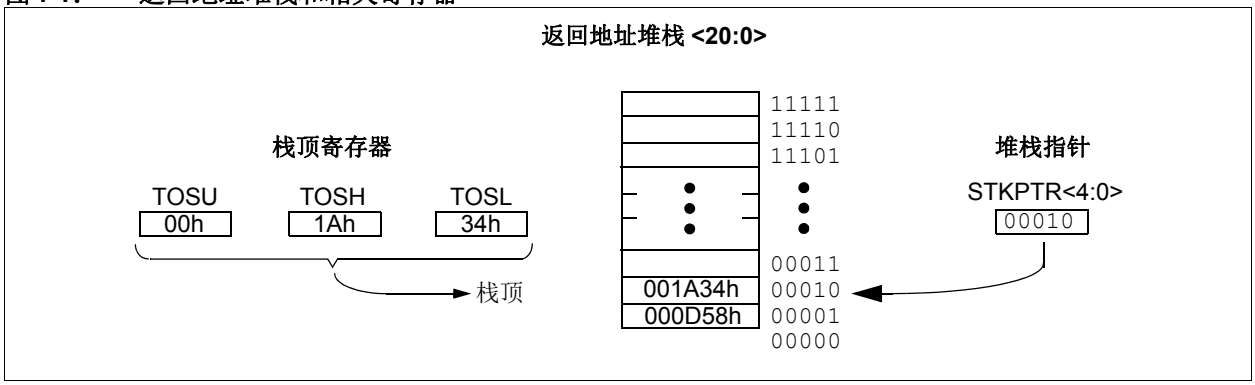
所有复位后，堆栈指针均会初始化为00000。堆栈指针值00000不指向任何RAM单元；它仅仅是一个复位值。PCON0寄存器中的状态位指示堆栈是上溢还是下溢。

4.2.5.1 栈顶访问

仅可读写返回地址的栈顶（TOS）。包含三个寄存器的寄存器组TOSU:TOSH:TOSL用于保存STKPTR寄存器指向的堆栈单元的内容（图4-1）。这可以让用户在必要时实现软件堆栈。在CALL、RCALL或中断后，软件可以通过读取TOSU:TOSH:TOSL寄存器来读取压入堆栈的值。这些值可以存放在用户定义的软件堆栈上。返回时，软件可以将这些值返回给TOSU:TOSH:TOSL并执行返回。

为防止意外损坏堆栈，访问堆栈时用户必须禁止全局中断允许（Global Interrupt Enable，GIE）位。

图4-1： 返回地址堆栈和相关寄存器



4.2.5.2 返回堆栈指针（STKPTR）

STKPTR寄存器（寄存器4-4）包含堆栈指针值。可以使用PCON0寄存器访问STKOVF（堆栈上溢）状态位和STKUNF（堆栈下溢）状态位。堆栈指针的值可以为0至31范围内的任何值。复位时，堆栈指针值将为0。用户可以读写堆栈指针值。实时操作系统（Real-Time Operating System，RTOS）可以利用该特性进行堆栈维护。PC值被压入堆栈32次（且没有值从堆栈弹出）后，STKOVF位置1。STKOVF位由软件或POR清零。由STVREN（堆栈溢出复位使能）配置位的状态决定堆栈满时将执行的操作。（有关器件配置位的说明，见**第5.1节“配置字”**。）

如果STVREN置1（默认），则第32次压栈时，将产生复位，并且STKOVF位将指示堆栈上溢。这包括CALL和CALLW指令，以及在中断响应期间的返回地址堆栈。STKOVF位将保持置1，堆栈指针将设置为0。

如果STVREN清零，第32次压栈时STKOVF位将置1，堆栈指针将保持为31，但不会发生复位。任何其他压栈操作都将覆盖第31次压栈的值，但STKPTR将保持为31。

在软件中设置STKOVF = 1将更改该位，但不会产生复位。

出栈返回值0时，STKUNF位置1。STKUNF位由软件或POR清零。由STVREN（堆栈溢出复位使能）配置位的状态决定堆栈满时将执行的操作。（有关器件配置位的说明，请参见**第5.1节“配置字”**）。

如果STVREN置1（默认）并且出栈次数足够卸空堆栈时，下一次出栈会向PC返回一个零值，并将STKUNF位置1，此时将产生复位。上述情况可以由RETURN、RETLW和RETFIE指令产生。

当STVREN = 0时，STKUNF将置1，但不会发生复位。

> **注：** 下溢时，将零值返回给PC，会使程序指向复位向量，此时可以验证堆栈状态并采取相应的操作。这与复位不同，因为下溢时SFR的内容不受影响。

4.2.5.3 PUSH和POP指令

由于栈顶是可以读写的，因此能够将值压入堆栈或从堆栈弹出而不影响程序的正常执行是一种理想的特性。PIC18指令集包括两条指令PUSH和POP，使用这两条指令可以在软件控制下对TOS执行操作。然后就可以修改TOSU、TOSH和TOSL，将数据或返回地址压入堆栈。

PUSH指令用于将当前PC值压入堆栈。执行该指令会使堆栈指针递增并将当前PC值装入堆栈。

POP指令通过递减堆栈指针来放弃当前TOS值。然后前一个入栈的值就成为了TOS值。

4.3 寄存器定义：堆栈指针

寄存器4-1： TOSU：栈顶最高字节

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	TOS<20:16>				
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现	C = 仅可清零位
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **未实现：**读为0

bit 4-0 **TOS<20:16>：**栈顶单元位

寄存器4-2： TOSH：栈顶高字节

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TOS<15:8>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现	C = 仅可清零位
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **TOS<15:8>：**栈顶单元位

寄存器4-3： TOSL：栈顶低字节

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TOS<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现	C = 仅可清零位
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **TOS<7:0>：**栈顶单元位

寄存器4-4：　　STKPTR：堆栈指针寄存器

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	STKPTR<4:0>				
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现	C = 仅可清零位
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5　　**未实现：**读为0

bit 4-0　　**STKPTR<4:0>：**堆栈指针单元位

4.3.1　　快速寄存器堆栈

有三种级别的快速堆栈寄存器可供使用——一种用于CALL型指令，两种用于中断。为STATUS、WREG和BSR寄存器提供的快速寄存器堆栈具有从中断“快速返回”的功能。当处理器转入中断向量处执行时，它装入对应寄存器的当前值。所有中断源都会将值压入堆栈寄存器。如果使用RETFIE, FAST指令从中断返回，这些寄存器中的值就会被装回相关寄存器。有关中断调用影子寄存器的信息，请参见**第4.5.6节“调用影子寄存器”**。

例4-1给出了在子程序调用和返回期间使用快速寄存器堆栈的源代码示例。

例4-1：　　快速寄存器堆栈代码示例

```
CALL SUB1, FAST    ;STATUS, WREG, BSR
                   ;SAVED IN FAST REGISTER
                   ;STACK
       •
       •

SUB1   •
       •
   RETURN, FAST    ;RESTORE VALUES SAVED
                   ;IN FAST REGISTER STACK
```

4.3.2 程序存储器中的查找表

有的编程场合可能需要在程序存储器中创建数据结构或查找表。对于PIC18器件，可以采用以下两种方式实现查找表：

- 计算GOTO
- 表读

4.3.2.1 计算GOTO

计算GOTO是通过向程序计数器添加偏移量来实现的。例4-2给出了一个示例。

可以使用ADDWF PCL指令和一组RETLW nn指令来创建查找表。在调用该表前，会先将查找表中的偏移量装入W寄存器。调用程序的第一条指令为ADDWF PCL指令。执行的下一条指令将是RETLW nn指令之一，用于将值nn返回给调用函数。

偏移量值（WREG中）指定程序计数器应增加的字节数，其值应为2的倍数（LSb = 0）。

在该方法中，每个指令单元只能存储一个数据字节，并且要求返回地址堆栈中还有空闲的单元。

例4-2： 使用偏移量值的计算GOTO

```
        MOVF    OFFSET, W
        CALL    TABLE
ORG     nn00h
TABLE   ADDWF   PCL
        RETLW   nnh
        RETLW   nnh
        RETLW   nnh
        .
        .
        .
```

4.3.2.2 表读和表写

有一种更好的方法可以将数据存储在程序存储器中，这种方法允许在每个指令单元存储2个字节的数据。

使用表读和表写方法，每个程序字可存储2个字节的查找表数据。表指针（TBLPTR）寄存器指定字节地址，而表锁存器（TABLAT）寄存器则存储从程序存储器读取或写入程序存储器的数据。

第13.1.1节“表读和表写”将进一步讨论表读和表写操作。

4.4 PIC18指令周期

4.4.1 时钟机制

单片机时钟输入（无论来自内部源还是外部源）在内部进行4分频，产生4个正交时钟（Q1、Q2、Q3和Q4）。在内部，程序计数器在每个Q1递增；在Q4期间，从程序存储器中取指令并将指令锁存到指令寄存器中。指令的译码和执行在下一个Q1至Q4周期完成。图4-2所示为时钟和指令执行流程。

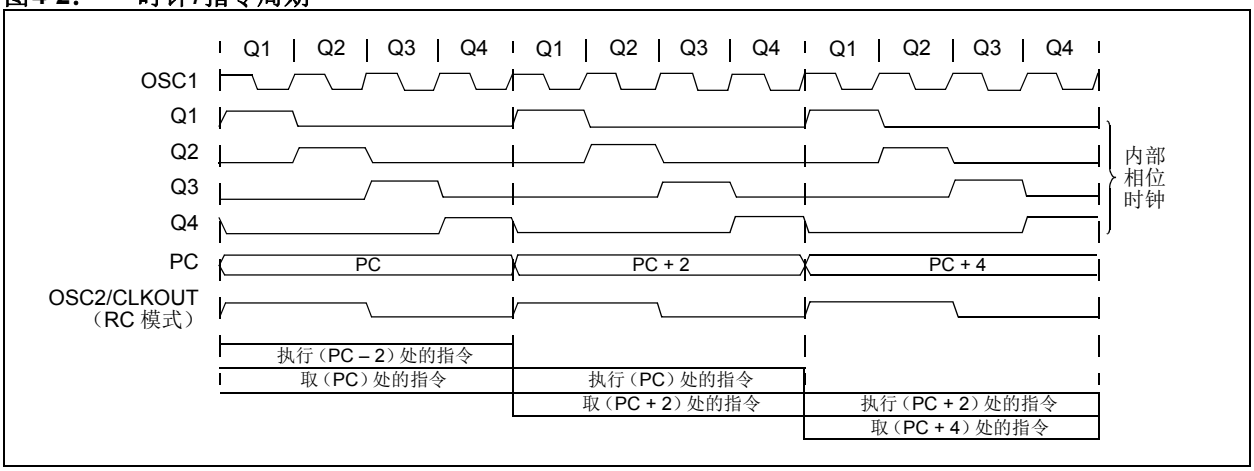
4.4.2 指令流/流水线

一个“指令周期”由Q1至Q4四个Q周期组成。取指令和执行指令是以流水线方式进行的，用一个指令周期来取指令，而用另一个指令周期译码和执行指令。但由于是流水线操作，所以每条指令的等效执行时间都是一个指令周期。如果某条指令（如GOTO）改变了程序计数器，则需要两个指令周期才能完成该指令（例4-3）。

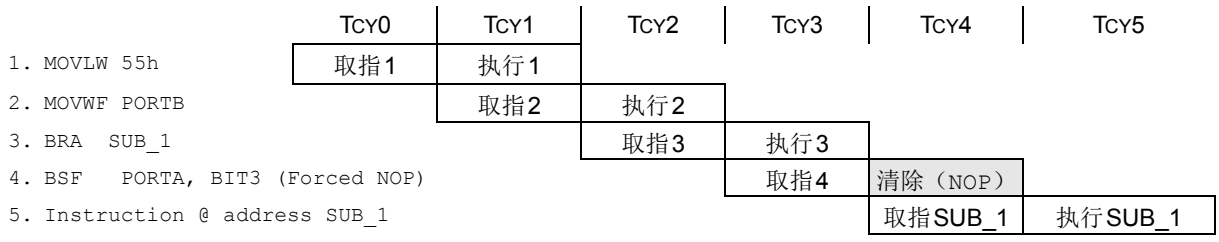
取指周期中：程序计数器（PC）在Q1周期递增，开始取指令。

指令执行周期中：在Q1周期，将所取指令锁存到指令寄存器（Instruction Register，IR）。在随后的Q2、Q3和Q4周期中译码并执行该指令。其中读数据存储器（读操作数）发生在Q2周期，写操作（写目标寄存器）发生在Q4周期。

图4-2： 时钟/指令周期



例4-3： 指令流水线流程



注： 有一些指令需要花费多个周期执行。有关详细信息，请参见**第42.0节“指令集汇总”**。

4.4.3 程序存储器中的指令

程序存储器按字节寻址。指令以2字节或4字节的形式存储在程序存储器中。指令字的最低有效字节始终存储在地址为偶数的程序存储单元中（LSb = 0）。为了保持与指令边界对齐，PC以2为增量进行递增，并且LSb始终读为0（见**第4.2.4节“程序计数器”**）。

图4-3给出了指令字存储在程序存储器中的一个示例。

CALL和GOTO指令在指令中嵌入了程序存储器的绝对地址。由于指令始终按字边界存储，因而指令所包含的数据为一个字地址。字地址会写入PC<20:1>，用于访问程序存储器中的目标字节地址。图4-3中的指令2给出了指令GOTO 0006h在程序存储器中的编码过程。程序跳转指令也采取同样的方式对相对地址偏移量进行编码。存储在跳转指令中的偏移量代表单字指令数，PC将以此作为偏移量跳转到指定的地址单元。**第42.0节“指令集汇总”**提供了指令集的更多详细信息。

4.4.4 多字指令

标准PIC18指令集有4条双字指令：CALL、MOVFF、GOTO和LFSR，以及2条三字指令：MOVFFL和MOVSFL。在所有情况下，这些指令第二个字和第三个字的高4位均为1111；其他12位是立即数数据，通常为一个数据存储器地址。

指令的高4位为1111，用于指定一条特殊形式的NOP。指令执行的正确顺序为：执行完第一个字之后立即按顺序访问并使用第二个字中的数据。如果由于某种原因跳过了第一个字而自动执行第二个字或第三个字，那么将作为一条NOP指令执行。如果多字指令跟在修改PC的条件指令后，就有必要执行此操作。例4-4给出了它的执行过程。

图4-3： 程序存储器中的指令

			LSB = 1	LSB = 0	字地址
	程序存储器 字节单元 →				
					000000h
					000002h
					000004h
					000006h
指令1：	MOVLW	055h	0Fh	55h	000008h
指令2：	GOTO	0006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
指令3：	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
指令4：	MOVFFL	123h, 456h	00h	60h	000012h
			F4h	8Ch	000014h
			F4h	56h	000016h
					000018h
					00001Ah

例4-4： 双字指令

情形1：			
目标代码	**源代码**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; Yes, skip this word
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code
情形2：			
目标代码	**源代码**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; No, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

例4-5： 三字指令

情形1：			
目标代码	**源代码**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
0000 0000 0110 0000	MOVFFL	REG1, REG2	; Yes, skip this word
1111 0100 1000 1100			; Execute this word as a NOP
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code
情形2：			
目标代码	**源代码**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
0000 0000 0110 0000	MOVFFL	REG1, REG2	; No, execute this word
1111 0100 1000 1100			; 2nd word of instruction
1111 0100 0101 0110			; 3rd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

4.5 数据存储器构成

PIC18(L)F25/26K83器件中的数据存储器以静态RAM的形式实现。数据存储器中的每个寄存器都有14位地址，允许实现最大为16384个字节的数据存储器。存储空间分为64个存储区，每个存储区包含256个字节。图4-5给出了本数据手册中PIC18(L)F25/26K83器件的数据存储器构成。

数据存储器包含特殊功能寄存器（Special Function Register，SFR）和通用寄存器（General Purpose Register，GPR）。SFR用于控制器和外设功能的控制和状态显示，而GPR用于用户应用程序中的数据存储和中间结果暂停操作。任何未实现的存储单元均读为0。

指令集和架构支持跨所有存储区的操作。可以通过直接、间接或变址寻址模式访问整个数据存储器。本章后面的部分将介绍寻址模式。

为确保可在一个周期内访问常用寄存器（选择SFR和GPR），PIC18器件实现了快速操作存储区。该存储区是一个256字节的存储空间，无需使用存储区选择寄存器（Bank Select Register，BSR）即可快速访问某些SFR以及GPR Bank 0的低地址单元。**第4.5.4节“快速操作存储区”**提供了快速操作RAM的详细说明。

4.5.1 存储区选择寄存器（BSR）

容量较大的数据存储器需要高效的寻址机制，以便对所有地址进行快速访问。理想情况下，这意味着不需要为每个读写操作提供完整地址。PIC18器件通过RAM存储区分区机制来实现快速访问。这种机制将存储空间分成连续的64个256字节的存储区。根据不同的指令，可以通过完整的14位地址，或通过8位的低字节地址和6位存储区选择寄存器直接寻址每个存储单元。

该SFR保存单元地址的高6位；指令本身包含低8位。只使用BSR的低6位（BSR<5:0>），不使用高2位，高2位总是读为0且不能被写入。可以使用MOVLB指令直接装入BSR。

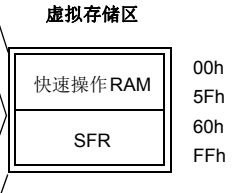
BSR的值代表数据存储器中的存储区；指令中的8位指向存储区中的存储单元，可以将其看作距离存储区下边界的偏移量。图4-5显示了BSR的值与数据存储器中的存储区之间的关系。

由于最多可有64个寄存器共享同一个低位地址，用户必须非常小心以确保在执行数据读或写操作之前选择了正确的存储区。例如，当BSR为3Fh时将程序数据写入地址为F9h的8位地址单元，将导致程序计数器被损坏。

虽然可以选择任何存储区，但只有已实现的存储区才可以读写。对未实现的存储区进行的写操作将被忽略，而读这些存储区会返回0。虽然是这样，这些操作仍然会对STATUS寄存器起作用，就好像操作成功了一样。图4-5中的数据存储器映射指出了已实现的存储区。

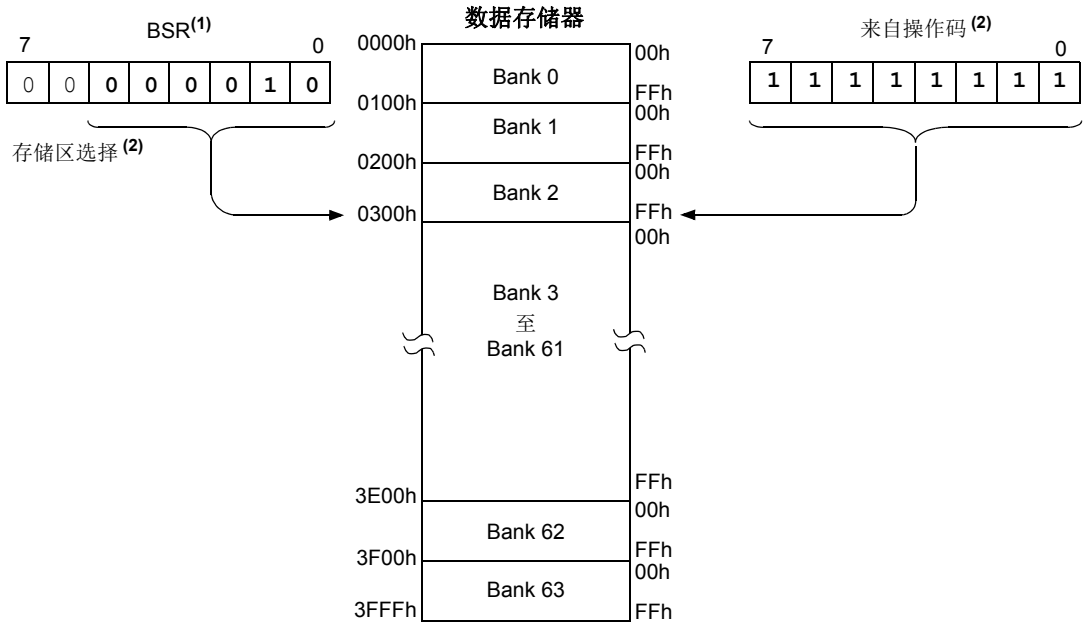
图4-4： PIC18(L)F25/26K83器件的数据存储器映射

存储区	BSR<5:0>	addr<7:0>	PIC18(L)F25K83	PIC18(L)F26K83	地址 addr<13:0>
Bank 0	00 0000	00h FFh	快速操作RAM	快速操作RAM	0000h 005Fh
			GPR	GPR	0060h 00FFh
Bank 1	00 0001	00h FFh	GPR	GPR	0100h • • • 03FFh
Bank 2	00 0010	00h FFh			
Bank 3	00 0011	00h FFh			
Bank 4至7	00 0100 — 00 0111	00h • • FFh	GPR	GPR	0400h • • 07FFh
Bank 8至15	00 1000 — 00 1111	00h • • FFh	未实现	GPR	0800h • • 0FFFh
Bank 16至31	01 0000 — 01 1111	00h • • FFh		未实现	1000h • • 1FFFh
Bank 32至53	10 0000 — 11 0111	00h • • • FFh			2000 • • • 37FF
Bank 54	d54	00h • FFh	CAN测试	CAN测试	3600h • 36FFh
Bank 55	d55	00h • FFh	CAN SFR	CAN SFR	3700h • 37FFh
Bank 56至62	11 1000 — 11 1110	00h • • FFh	SFR	SFR	3800h • • 3EFFh
Bank 63	11 1111	00h FFh	SFR	SFR	3F00h 3F5Fh 3F60h 3FFFh



注 1： 取决于SFR数。请参见后续SFR表。

图4-5： 使用存储区选择寄存器（直接寻址）



注 1： 指令的快速操作RAM位可用于强制忽略选定的存储区（BSR<5:0>），直接寻址快速操作存储区的寄存器。

4.5.2 通用寄存器文件

通用RAM从数据存储器的Bank 0开始。上电复位不会初始化GPR，并且其他复位也不会改变其内容。

4.5.3 特殊功能寄存器

特殊功能寄存器（SFR）是CPU和外设模块用来控制所需器件操作的寄存器。这些寄存器以静态RAM的形式实现。SFR从数据存储器的顶部（3FFFh）开始向下，它占用了Bank 56至Bank 63（3800h至3FFFh）。表4-3至表4-10列出了这些寄存器。有关这些寄存器的按位汇总信息，请参见**第43.0节“寄存器汇总”**。

4.5.4 快速操作存储区

为了提高访问大多数常用数据存储单元的效率，现为数据存储器配置了快速操作存储区，这样可以允许用户访问被映射的存储区而无需指定BSR。快速操作存储区由Bank 0的前96个字节（00h-5Fh）和Bank 63的后160个字节（60h-FFh）组成。地址较低的部分被称为“快速操作RAM”，由GPR组成。地址较高的部分则被映射为器件的SFR。这两个区域被连续地映射到快速操作存储区并且可以用一个8位地址进行线性寻址（图4-5）。

包括快速操作RAM位（指令中的“a”参数）的核心PIC18指令使用快速操作存储区。当a等于1时，指令使用BSR和包含在操作码中的8位地址对数据存储器寻址。但是，当a等于0时，指令使用快速操作存储区地址映射；BSR的当前值被忽略。

此“强制”寻址模式可使指令在一个周期内对数据地址进行操作，而不需要首先更新BSR。这意味着用户可以更高效地对8位地址为60h或以上的SFR进行评估和操作。地址为60h以下的快速操作RAM非常适合于存储那些用户可能需要快速访问的数据值，如直接计算结果或常用程序变量。快速操作RAM也可实现更加快速、更加高效的变量切换代码。

使能扩展指令集（XINST配置位 = 1）时的快速操作存储区的映射略有不同。在**第4.8.3节“在立即数变址寻址模式下映射快速操作存储区”**中对此进行了更详细的讨论。

表4-3： PIC18(L)F25/26K83器件BANK 63的特殊功能寄存器映射

3FFFh	TOSU	3FDFh	INDF2	3FBFh	—	3F9Fh	T4PR	3F7Fh	CCP1CAP	3F5Fh	CCPTMRS1	3F3Fh	NCO1CLK	3F1Fh	SMT1CON1
3FFEh	TOSH	3FDEh	POSTINC2	3FBEh	—	3F9Eh	T4TMR	3F7Eh	CCP1CON	3F5Eh	CCPTMRS0	3F3Eh	NCO1CON	3F1Eh	SMT1CON0
3FFDh	TOSL	3FDDh	POSTDEC2	3FBDh	—	3F9Dh	T5CLK	3F7Dh	CCPR1H	3F5Dh	—	3F3Dh	NCO1INCU	3F1Dh	SMT1PRU
3FFCh	STKPTR	3FDCh	PRECIN2	3FBCh	LATC	3F9Ch	T5GATE	3F7Ch	CCPR1L	3F5Ch	—	3F3Ch	NCO1INCH	3F1Ch	SMT1PRH
3FFBh	PCLATU	3FDBh	PLUSW2	3FBBh	LATB	3F9Bh	T5GCON	3F7Bh	CCP2CAP	3F5Bh	—	3F3Bh	NCO1INCL	3F1Bh	SMT1PRL
3FFAh	PCLATH	3FDAh	FSR2H	3FBAh	LATA	3F9Ah	T5CON	3F7Ah	CCP2CON	3F5Ah	CWG1STR	3F3Ah	NCO1ACCU	3F1Ah	SMT1CPWU
3FF9h	PCL	3FD9h	FSR2L	3FB9h	T0CON1	3F99h	TMR5H	3F79h	CCPR2H	3F59h	CWG1AS1	3F39h	NCO1ACCH	3F19h	SMT1CPWH
3FF8h	TBLPRTU	3FD8h	STATUS	3FB8h	T0CON0	3F98h	TMR5L	3F78h	CCPR2L	3F58h	CWG1AS0	3F38h	NCO1ACCL	3F18h	SMT1CPWL
3FF7h	TBLPTRH	3FD7h	IVTBASEU	3FB7h	TMR0H	3F97h	T6RST	3F77h	CCP3CAP	3F57h	CWG1CON1	3F37h	—	3F17h	SMT1CPRU
3FF6h	TBLPTRL	3FD6h	IVTBASEH	3FB6h	TMR0L	3F96h	T6CLK	3F76h	CCP3CON	3F56h	CWG1CON0	3F36h	—	3F16h	SMT1CPRH
3FF5h	TABLAT	3FD5h	IVTBASEL	3FB5h	T1CLK	3F95h	T6HLT	3F75h	CCPR3H	3F55h	CWG1DBF	3F35h	—	3F15h	SMT1CPRL
3FF4h	PRODH	3FD4h	IVTLOCK	3FB4h	T1GATE	3F94h	T6CON	3F74h	CCPR3L	3F54h	CWG1DBR	3F34h	—	3F14h	SMT1TMRU
3FF3h	PRODL	3FD3h	INTCON1	3FB3h	T1GCON	3F93h	T6PR	3F73h	CCP4CAP	3F53h	CWG1ISM	3F33h	—	3F13h	SMT1TMRH
3FF2h	—	3FD2h	INTCON0	3FB2h	T1CON	3F92h	T6TMR	3F72h	CCP4CON	3F52h	CWG1CLK	3F32h	—	3F12h	SMT1TMRL
3FF1h	PCON1	3FD1h	—	3FB1h	TMR1H	3F91h	ECANCON	3F71h	CCPR4H	3F51h	CWG2STR	3F31h	—	3F11h	SMT2WIN
3FF0h	PCON0	3FD0h	—	3FB0h	TMR1L	3F90h	COMSTAT	3F70h	CCPR4L	3F50h	CWG2AS1	3F30h	—	3F10h	SMT2SIG
3FEFh	INDF0	3FCFh	—	3FAFh	T2RST	3F8Fh	CANCON	3F6Fh	—	3F4Fh	CWG2AS0	3F2Fh	—	3F0Fh	SMT2CLK
3FEEh	POSTINC0	3FCEh	PORTE	3FAEh	T2CLK	3F8Eh	CANSTAT	3F6Eh	PWM5CON	3F4Eh	CWG2CON1	3F2Eh	—	3F0Eh	SMT2STAT
3FEDh	POSTDEC0	3FCDh	—	3FADh	T2HLT	3F8Dh	RXB0D7	3F6Dh	PWM5DCH	3F4Dh	CWG2CON0	3F2Dh	—	3F0Dh	SMT2CON1
3FECh	PRECIN0	3FCCh	PORTC	3FACh	T2CON	3F8Ch	RXB0D6	3F6Ch	PWM5DCL	3F4Ch	CWG2DBF	3F2Ch	—	3F0Ch	SMT2CON0
3FEBh	PLUSW0	3FCBh	PORTB	3FABh	T2PR	3F8Bh	RXB0D5	3F6Bh	—	3F4Bh	CWG2DBR	3F2Bh	—	3F0Bh	SMT2PRU
3FEAh	FSR0H	3FCAh	PORTA	3FAAh	T2TMR	3F8Ah	RXB0D4	3F6Ah	PWM6CON	3F4Ah	CWG2ISM	3F2Ah	—	3F0Ah	SMT2PRH
3FE9h	FSR0L	3FC9h	—	3FA9h	T3CLK	3F89h	RXB0D3	3F69h	PWM6DCH	3F49h	CWG2CLK	3F29h	—	3F09h	SMT2PRL
3FE8h	WREG	3FC8h	—	3FA8h	T3GATE	3F88h	RXB0D2	3F68h	PWM6DCL	3F48h	CWG3STR	3F28h	—	3F08h	SMT2CPWU
3FE7h	INDF1	3FC7h	—	3FA7h	T3GCON	3F87h	RXB0D1	3F67h	—	3F47h	CWG3AS1	3F27h	—	3F07h	SMT2CPWH
3FE6h	POSTINC1	3FC6h	—	3FA6h	T3CON	3F86h	RXB0D0	3F66h	PWM7CON	3F46h	CWG3AS0	3F26h	—	3F06h	SMT2CPWL
3FE5h	POSTDEC1	3FC5h	—	3FA5h	TMR3H	3F85h	RXB0DLC	3F65h	PWM7DCH	3F45h	CWG3CON1	3F25h	—	3F05h	SMT2CPRU
3FE4h	PRECIN1	3FC4h	TRISC	3FA4h	TMR3L	3F84h	RXB0EIDL	3F64h	PWM7DCL	3F44h	CWG3CON0	3F24h	—	3F04h	SMT2CPRH
3FE3h	PLUSW1	3FC3h	TRISB	3FA3h	T4RST	3F83h	RXB0EIDH	3F63h	—	3F43h	CWG3DBF	3F23h	SMT1WIN	3F03h	SMT2CPRL
3FE2h	FSR1H	3FC2h	TRISA	3FA2h	T4CLK	3F82h	RXB0SIDL	3F62h	PWM8CON	3F42h	CWG3DBR	3F22h	SMT1SIG	3F02h	SMT2TMRU
3FE1h	FSR1L	3FC1h	—	3FA1h	T4HLT	3F81h	RXB0SIDH	3F61h	PWM8DCH	3F41h	CWG3ISM	3F21h	SMT1CLK	3F01h	SMT2TMRH
3FE0h	BSR	3FC0h	—	3FA0h	T4CON	3F80h	RXB0CON	3F60h	PWM8DCL	3F40h	CWG3CLK	3F20h	SMT1STAT	3F00h	SMT2TMRL

图注： 未实现的数据存储单元和寄存器，读为0。

表4-4： PIC18(L)F25/26K83器件BANK 62的特殊功能寄存器映射

3EFFh	ADCLK	3EDFh	ADLTHH	3EBFh	CM1PCH	3E9Fh	—	3E7Fh	—	3E5Fh	—	3E3Fh	—	3E1Fh	—
3EFEh	ADACT	3EDEh	ADLTHL	3EBEh	CM1NCH	3E9Eh	DAC1CON0	3E7Eh	—	3E5Eh	—	3E3Eh	—	3E1Eh	—
3EFDh	ADREF	3EDDh	—	3EBDh	CM1CON1	3E9Dh	—	3E7Dh	—	3E5Dh	—	3E3Dh	—	3E1Dh	—
3EFCh	ADSTAT	3EDCh	—	3EBCh	CM1CON0	3E9Ch	DAC1CON1	3E7Ch	—	3E5Ch	—	3E3Ch	—	3E1Ch	—
3EFBh	ADCON3	3EDBh	—	3EBBh	CM2PCH	3E9Bh	—	3E7Bh	—	3E5Bh	—	3E3Bh	—	3E1Bh	—
3EFAh	ADCON2	3EDAh	—	3EBAh	CM2NCH	3E9Ah	—	3E7Ah	—	3E5Ah	—	3E3Ah	—	3E1Ah	—
3EF9h	ADCON1	3ED9h	—	3EB9h	CM2CON1	3E99h	—	3E79h	—	3E59h	—	3E39h	—	3E19h	—
3EF8h	ADCON0	3ED8h	—	3EB8h	CM2CON0	3E98h	—	3E78h	—	3E58h	—	3E38h	—	3E18h	—
3EF7h	ADPREH	3ED7h	ADCP	3EB7h	—	3E97h	—	3E77h	—	3E57h	—	3E37h	—	3E17h	—
3EF6h	ADPREL	3ED6h	—	3EB6h	—	3E96h	—	3E76h	—	3E56h	—	3E36h	—	3E16h	—
3EF5h	ADCAP	3ED5h	—	3EB5h	—	3E95h	—	3E75h	—	3E55h	—	3E35h	—	3E15h	—
3EF4h	ADACQH	3ED4h	—	3EB4h	—	3E94h	—	3E74h	—	3E54h	—	3E34h	—	3E14h	—
3EF3h	ADACQL	3ED3h	—	3EB3h	—	3E93h	—	3E73h	—	3E53h	—	3E33h	—	3E13h	—
2EF2h	—	3ED2h	—	3EB2h	—	3E92h	—	3E72h	—	3E52h	—	3E32h	—	3E12h	—
3EF1h	ADPCH	3ED1h	—	3EB1h	—	3E91h	—	3E71h	—	3E51h	—	3E31h	—	3E11h	—
3EF0h	ADRESH	3ED0h	—	3EB0h	—	3E90h	—	3E70h	—	3E50h	—	3E30h	—	3E10h	—
3EEFh	ADRESL	3ECFh	—	3EAFh	—	3E8Fh	—	3E6Fh	—	3E4Fh	—	3E2Fh	—	3E0Fh	—
3EEEh	ADPREVH	3ECEh	—	3EAEh	—	3E8Eh	—	3E6Eh	—	3E4Eh	—	3E2Eh	—	3E0Eh	—
3EEDh	ADPREVL	3ECDh	—	3EADh	—	3E8Dh	—	3E6Dh	—	3E4Dh	—	3E2Dh	—	3E0Dh	—
3EECh	ADRPT	3ECCh	—	3EACh	—	3E8Ch	—	3E6Ch	—	3E4Ch	—	3E2Ch	—	3E0Ch	—
3EEBh	ADCNT	3ECBh	—	3EABh	—	3E8Bh	—	3E6Bh	—	3E4Bh	—	3E2Bh	—	3E0Bh	—
3EEAh	ADACCU	3ECAh	HLVDCON1	3EAAh	—	3E8Ah	—	3E6Ah	—	3E4Ah	—	3E2Ah	—	3E0Ah	—
3EE9h	ADACCH	3EC9h	HLVDCON0	3EA9h	—	3E89h	—	3E69h	—	3E49h	—	3E29h	—	3E09h	—
3EE8h	ADACCL	3EC8h	—	3EA8h	—	3E88h	—	3E68h	—	3E48h	—	3E28h	—	3E08h	—
3EE7h	ADFLTRH	3EC7h	—	3EA7h	—	3E87h	—	3E67h	—	3E47h	—	3E27h	—	3E07h	—
3EE6h	ADFLTRL	3EC6h	—	3EA6h	—	3E86h	—	3E66h	—	3E46h	—	3E26h	—	3E06h	—
3EE5h	ADSTPTH	3EC5h	—	3EA5h	—	3E85h	—	3E65h	—	3E45h	—	3E25h	—	3E05h	—
3EE4h	ADSTPTL	3EC4h	—	3EA4h	—	3E84h	—	3E64h	—	3E44h	—	3E24h	—	3E04h	—
3EE3h	ADERRH	3EC3h	ZCDCON	3EA3h	—	3E83h	—	3E63h	—	3E43h	—	3E23h	—	3E03h	—
3EE2h	ADERRL	3EC2h	—	3EA2h	—	3E82h	—	3E62h	—	3E42h	—	3E22h	—	3E02h	—
3EE1h	ADUTHH	3EC1h	FVRCON	3EA1h	—	3E81h	—	3E61h	—	3E41h	—	3E21h	—	3E01h	—
3EE0h	ADUTHL	3EC0h	CMOUT	3EA0h	—	3E80h	—	3E60h	—	3E40h	—	3E20h	—	3E00h	—

图注： 未实现的数据存储单元和寄存器，读为0。

表4-5： PIC18(L)F25/26K83器件BANK 61的特殊功能寄存器映射

3DFFh	—	3DDFh	U2FIFO	3DBFh	—	3D9Fh	—	3D7Fh	—	3D5Fh	I2C2CON2	3D3Fh	—	3D1Fh	—
3DFEh	—	3DDEh	U2BRGH	3DBEh	—	3D9Eh	—	3D7Eh	—	3D5Eh	I2C2CON1	3D3Eh	—	3D1Eh	—
3DFDh	—	3DDDh	U2BRGL	3DBDh	—	3D9Dh	—	3D7Dh	—	3D5Dh	I2C2CON0	3D3Dh	—	3D1Dh	—
3DFCh	—	3DDCh	U2CON2	3DBCh	—	3D9Ch	—	3D7Ch	I2C1BTO	3D5Ch	I2C2ADR3	3D3Ch	—	3D1Ch	SPI1CLK
3DFBh	—	3DDBh	U2CON1	3DBBh	—	3D9Bh	—	3D7Bh	I2C1CLK	3D5Bh	I2C2ADR2	3D3Bh	—	3D1Bh	SPI1INTE
3DFAh	U1ERRIE	3DDAh	U2CON0	3DBAh	—	3D9Ah	—	3D7Ah	I2C1PIE	3D5Ah	I2C2ADR1	3D3Ah	—	3D1Ah	SPI1INTF
3DF9h	U1ERRIR	3DD9h	U2P3H	3DB9h	—	3D99h	—	3D79h	I2C1PIR	3D59h	I2C2ADR0	3D39h	—	3D19h	SPI1BAUD
3DF8h	U1UIR	3DD8h	U2P3L	3DB8h	—	3D98h	—	3D78h	I2C1STAT1	3D58h	I2C2ADB1	3D38h	—	3D18h	SPI1TWIDTH
3DF7h	U1FIFO	3DD7h	U2P2H	3DB7h	—	3D97h	—	3D77h	I2C1STAT0	3D57h	I2C2ADB0	3D37h	—	3D17h	SPI1STATUS
3DF6h	U1BRGH	3DD6h	U2P2L	3DB6h	—	3D96h	—	3D76h	I2C1ERR	3D56h	I2C2CNT	3D36h	—	3D16h	SPI1CON2
3DF5h	U1BRGL	3DD5h	U2P1H	3DB5h	—	3D95h	—	3D75h	I2C1CON2	3D55h	I2C2TXB	3D35h	—	3D15h	SPI1CON1
3DF4h	U1CON2	3DD4h	U2P1L	3DB4h	—	3D94h	—	3D74h	I2C1CON1	3D54h	I2C2RXB	3D34h	—	3D14h	SPI1CON0
3DF3h	U1CON1	3DD3h	U2TXCHK	3DB3h	—	3D93h	—	3D73h	I2C1CON0	3D53h	—	3D33h	—	3D13h	SPI1TCNTH
3DF2h	U1CON0	3DD2h	U2TXB	3DB2h	—	3D92h	—	3D72h	I2C1ADR3	3D52h	—	3D32h	—	3D12h	SPI1TCNTL
3DF1h	U1P3H	3DD1h	U2RXCHK	3DB1h	—	3D91h	—	3D71h	I2C1ADR2	3D51h	—	3D31h	—	3D11h	SPI1TXB
3DF0h	U1P3L	3DD0h	U2RXB	3DB0h	—	3D90h	—	3D70h	I2C1ADR1	3D50h	—	3D30h	—	3D10h	SPI1RXB
3DEFh	U1P2H	3DCFh	—	3DAFh	—	3D8Fh	—	3D6Fh	I2C1ADR0	3D4Fh	—	3D2Fh	—	3D0Fh	—
3DEEh	U1P2L	3DCEh	—	3DAEh	—	3D8Eh	—	3D6Eh	I2C1ADB1	3D4Eh	—	3D2Eh	—	3D0Eh	—
3DEDh	U1P1H	3DCDh	—	3DADh	—	3D8Dh	—	3D6Dh	I2C1ADB0	3D4Dh	—	3D2Dh	—	3D0Dh	—
3DECh	U1P1L	3DCCh	—	3DACh	—	3D8Ch	—	3D6Ch	I2C1CNT	3D4Ch	—	3D2Ch	—	3D0Ch	—
3DEBh	U1TXCHK	3DCBh	—	3DABh	—	3D8Bh	—	3D6Bh	I2C1TXB	3D4Bh	—	3D2Bh	—	3D0Bh	—
3DEAh	U1TXB	3DCAh	—	3DAAh	—	3D8Ah	—	3D6Ah	I2C1RXB	3D4Ah	—	3D2Ah	—	3D0Ah	—
3DE9h	U1RXCHK	3DC9h	—	3DA9h	—	3D89h	—	3D69h	—	3D49h	—	3D29h	—	3D09h	—
3DE8h	U1RXB	3DC8h	—	3DA8h	—	3D88h	—	3D68h	—	3D48h	—	3D28h	—	3D08h	—
3DE7h	—	3DC7h	—	3DA7h	—	3D87h	—	3D67h	—	3D47h	—	3D27h	—	3D07h	—
3DE6h	—	3DC6h	—	3DA6h	—	3D86h	—	3D66h	I2C2BTO	3D46h	—	3D26h	—	3D06h	—
3DE5h	—	3DC5h	—	3DA5h	—	3D85h	—	3D65h	I2C2CLK	3D45h	—	3D25h	—	3D05h	—
3DE4h	—	3DC4h	—	3DA4h	—	3D84h	—	3D64h	I2C2PIE	3D44h	—	3D24h	—	3D04h	—
3DE3h	—	3DC3h	—	3DA3h	—	3D83h	—	3D63h	I2C2PIR	3D43h	—	3D23h	—	3D03h	—
3DE2h	U2ERRIE	3DC2h	—	3DA2h	—	3D82h	—	3D62h	I2C2STAT1	3D42h	—	3D22h	—	3D02h	—
3DE1h	U2ERRIR	3DC1h	—	3DA1h	—	3D81h	—	3D61h	I2C2STAT0	3D41h	—	3D21h	—	3D01h	—
3DE0h	U2UIR	3DC0h	—	3DA0h	—	3D80h	—	3D60h	I2C2ERR	3D40h	—	3D20h	—	3D00h	—

图注： 未实现的数据存储单元和寄存器，读为0。

表4-6： PIC18(L)F25/26K83器件BANK 60的特殊功能寄存器映射

3CFFh	—	3CDFh	—	3CBFh	—	3C9Fh	—	3C7Fh	—	3C5Fh	CLC4GLS3	3C3Fh	—	3C1Fh	—
3CFEh	MD1CARH	3CDEh	—	3CBEh	—	3C9Eh	—	3C7Eh	CLCDATA0	3C5Eh	CLC4GLS2	3C3Eh	—	3C1Eh	—
3CFDh	MD1CARL	3CDDh	—	3CBDh	—	3C9Dh	—	3C7Dh	CLC1GLS3	3C5Dh	CLC4GLS1	3C3Dh	—	3C1Dh	—
3CFCh	MD1SRC	3CDCh	—	3CBCh	—	3C9Ch	—	3C7Ch	CLC1GLS2	3C5Ch	CLC4GLS0	3C3Ch	—	3C1Ch	—
3CFBh	MD1CON1	3CDBh	—	3CBBh	—	3C9Bh	—	3C7Bh	CLC1GLS1	3C5Bh	CLC4SEL3	3C3Bh	—	3C1Bh	—
3CFAh	MD1CON0	3CDAh	—	3CBAh	—	3C9Ah	—	3C7Ah	CLC1GLS0	3C5Ah	CLC4SEL2	3C3Ah	—	3C1Ah	—
3CF9h	—	3CD9h	—	3CB9h	—	3C99h	—	3C79h	CLC1SEL3	3C59h	CLC4SEL1	3C39h	—	3C19h	—
3CF8h	—	3CD8h	—	3CB8h	—	3C98h	—	3C78h	CLC1SEL2	3C58h	CLC4SEL0	3C38h	—	3C18h	—
3CF7h	—	3CD7h	—	3CB7h	—	3C97h	—	3C77h	CLC1SEL1	3C57h	CLC4POL	3C37h	—	3C17h	—
3CF6h	—	3CD6h	—	3CB6h	—	3C96h	—	3C76h	CLC1SEL0	3C56h	CLC4CON	3C36h	—	3C16h	—
3CF5h	—	3CD5h	—	3CB5h	—	3C95h	—	3C75h	CLC1POL	3C55h	—	3C35h	—	3C15h	—
3CF4h	—	3CD4h	—	3CB4h	—	3C94h	—	3C74h	CLC1CON	3C54h	—	3C34h	—	3C14h	—
3CF3h	—	3CD3h	—	3CB3h	—	3C93h	—	3C73h	CLC2GLS3	3C53h	—	3C33h	—	3C13h	—
3CF2h	—	3CD2h	—	3CB2h	—	3C92h	—	3C72h	CLC2GLS2	3C52h	—	3C32h	—	3C12h	—
3CF1h	—	3CD1h	—	3CB1h	—	3C91h	—	3C71h	CLC2GLS1	3C51h	—	3C31h	—	3C11h	—
3CF0h	—	3CD0h	—	3CB0h	—	3C90h	—	3C70h	CLC2GLS0	3C50h	—	3C30h	—	3C10h	—
3CEFh	—	3CCFh	—	3CAFh	—	3C8Fh	—	3C6Fh	CLC2SEL3	3C4Fh	—	3C2Fh	—	3C0Fh	—
3CEEh	—	3CCEh	—	3CAEh	—	3C8Eh	—	3C6Eh	CLC2SEL2	3C4Eh	—	3C2Eh	—	3C0Eh	—
3CEDh	—	3CCDh	—	3CADh	—	3C8Dh	—	3C6Dh	CLC2SEL1	3C4Dh	—	3C2Dh	—	3C0Dh	—
3CECh	—	3CCCh	—	3CACh	—	3C8Ch	—	3C6Ch	CLC2SEL0	3C4Ch	—	3C2Ch	—	3C0Ch	—
3CEBh	—	3CCBh	—	3CABh	—	3C8Bh	—	3C6Bh	CLC2POL	3C4Bh	—	3C2Bh	—	3C0Bh	—
3CEAh	—	3CCAh	—	3CAAh	—	3C8Ah	—	3C6Ah	CLC2CON	3C4Ah	—	3C2Ah	—	3C0Ah	—
3CE9h	—	3CC9h	—	3CA9h	—	3C89h	—	3C69h	CLC3GLS3	3C49h	—	3C29h	—	3C09h	—
3CE8h	—	3CC8h	—	3CA8h	—	3C88h	—	3C68h	CLC3GLS2	3C48h	—	3C28h	—	3C08h	—
3CE7h	—	3CC7h	—	3CA7h	—	3C87h	—	3C67h	CLC3GLS1	3C47h	—	3C27h	—	3C07h	—
3CE6h	CLKRCLK	3CC6h	—	3CA6h	—	3C86h	—	3C66h	CLC3GLS0	3C46h	—	3C26h	—	3C06h	—
3CE5h	CLKRCON	3CC5h	—	3CA5h	—	3C85h	—	3C65h	CLC3SEL3	3C45h	—	3C25h	—	3C05h	—
3CE4h	—	3CC4h	—	3CA4h	—	3C84h	—	3C64h	CLC3SEL2	3C44h	—	3C24h	—	3C04h	—
3CE3h	—	3CC3h	—	3CA3h	—	3C83h	—	3C63h	CLC3SEL1	3C43h	—	3C23h	—	3C03h	—
3CE2h	—	3CC2h	—	3CA2h	—	3C82h	—	3C62h	CLC3SEL0	3C42h	—	3C22h	—	3C02h	—
3CE1h	—	3CC1h	—	3CA1h	—	3C81h	—	3C61h	CLC3POL	3C41h	—	3C21h	—	3C01h	—
3CE0h	—	3CC0h	—	3CA0h	—	3C80h	—	3C60h	CLC3CON	3C40h	—	3C20h	—	3C00h	—

图注： 未实现的数据存储单元和寄存器，读为0。

表4-7： PIC18(L)F25/26K83器件BANK 59的特殊功能寄存器映射

3BFFh	DMA1SIRQ	3BDFh	DMA2SIRQ	3BBFh	—	3B9Fh	—	3B7Fh	—	3B5Fh	—	3B3Fh	—	3B1Fh	—
3BFEh	DMA1AIRQ	3BDEh	DMA2AIRQ	3BBEh	—	3B9Eh	—	3B7Eh	—	3B5Eh	—	3B3Eh	—	3B1Eh	—
3BFDh	DMA1CON1	3BDDh	DMA2CON1	3BBDh	—	3B9Dh	—	3B7Dh	—	3B5Dh	—	3B3Dh	—	3B1Dh	—
3BFCh	DMA1CON0	3BDCh	DMA2CON0	3BBCh	—	3B9Ch	—	3B7Ch	—	3B5Ch	—	3B3Ch	—	3B1Ch	—
3BFBh	DMA1SSAU	3BDBh	DMA2SSAU	3BBBh	—	3B9Bh	—	3B7Bh	—	3B5Bh	—	3B3Bh	—	3B1Bh	—
3BFAh	DMA1SSAH	3BDAh	DMA2SSAH	3BBAh	—	3B9Ah	—	3B7Ah	—	3B5Ah	—	3B3Ah	—	3B1Ah	—
3BF9h	DMA1SSAL	3BD9h	DMA2SSAL	3BB9h	—	3B99h	—	3B79h	—	3B59h	—	3B39h	—	3B19h	—
3BF8h	DMA1SSZH	3BD8h	DMA2SSZH	3BB8h	—	3B98h	—	3B78h	—	3B58h	—	3B38h	—	3B18h	—
3BF7h	DMA1SSZL	3BD7h	DMA2SSZL	3BB7h	—	3B97h	—	3B77h	—	3B57h	—	3B37h	—	3B17h	—
3BF6h	DMA1SPTRU	3BD6h	DMA2SPTRU	3BB6h	—	3B96h	—	3B76h	—	3B56h	—	3B36h	—	3B16h	—
3BF5h	DMA1SPTRH	3BD5h	DMA2SPTRH	3BB5h	—	3B95h	—	3B75h	—	3B55h	—	3B35h	—	3B15h	—
3BF4h	DMA1SPTRL	3BD4h	DMA2SPTRL	3BB4h	—	3B94h	—	3B74h	—	3B54h	—	3B34h	—	3B14h	—
3BF3h	DMA1SCNTH	3BD3h	DMA2SCNTH	3BB3h	—	3B93h	—	3B73h	—	3B53h	—	3B33h	—	3B13h	—
3BF2h	DMA1SCNTL	3BD2h	DMA2SCNTL	3BB2h	—	3B92h	—	3B72h	—	3B52h	—	3B32h	—	3B12h	—
3BF1h	DMA1DSAH	3BD1h	DMA2DSAH	3BB1h	—	3B91h	—	3B71h	—	3B51h	—	3B31h	—	3B11h	—
3BF0h	DMA1DSAL	3BD0h	DMA2DSAL	3BB0h	—	3B90h	—	3B70h	—	3B50h	—	3B30h	—	3B10h	—
3BEFh	DMA1DSZH	3BCFh	DMA2DSZH	3BAFh	—	3B8Fh	—	3B6Fh	—	3B4Fh	—	3B2Fh	—	3B0Fh	—
3BEEh	DMA1DSZL	3BCEh	DMA2DSZL	3BAEh	—	3B8Eh	—	3B6Eh	—	3B4Eh	—	3B2Eh	—	3B0Eh	—
3BEDh	DMA1DPTRH	3BCDh	DMA2DPTRH	3BADh	—	3B8Dh	—	3B6Dh	—	3B4Dh	—	3B2Dh	—	3B0Dh	—
3BECh	DMA1DPTRL	3BCCh	DMA2DPTRL	3BACh	—	3B8Ch	—	3B6Ch	—	3B4Ch	—	3B2Ch	—	3B0Ch	—
3BEBh	DMA1DCNTH	3BCBh	DMA2DCNTH	3BABh	—	3B8Bh	—	3B6Bh	—	3B4Bh	—	3B2Bh	—	3B0Bh	—
3BEAh	DMA1DCNTL	3BCAh	DMA2DCNTL	3BAAh	—	3B8Ah	—	3B6Ah	—	3B4Ah	—	3B2Ah	—	3B0Ah	—
3BE9h	DMA1BUF	3BC9h	DMA2BUF	3BA9h	—	3B89h	—	3B69h	—	3B49h	—	3B29h	—	3B09h	—
3BE8h	—	3BC8h	—	3BA8h	—	3B88h	—	3B68h	—	3B48h	—	3B28h	—	3B08h	—
3BE7h	—	3BC7h	—	3BA7h	—	3B87h	—	3B67h	—	3B47h	—	3B27h	—	3B07h	—
3BE6h	—	3BC6h	—	3BA6h	—	3B86h	—	3B66h	—	3B46h	—	3B26h	—	3B06h	—
3BE5h	—	3BC5h	—	3BA5h	—	3B85h	—	3B65h	—	3B45h	—	3B25h	—	3B05h	—
3BE4h	—	3BC4h	—	3BA4h	—	3B84h	—	3B64h	—	3B44h	—	3B24h	—	3B04h	—
3BE3h	—	3BC3h	—	3BA3h	—	3B83h	—	3B63h	—	3B43h	—	3B23h	—	3B03h	—
3BE2h	—	3BC2h	—	3BA2h	—	3B82h	—	3B62h	—	3B42h	—	3B22h	—	3B02h	—
3BE1h	—	3BC1h	—	3BA1h	—	3B81h	—	3B61h	—	3B41h	—	3B21h	—	3B01h	—
3BE0h	—	3BC0h	—	3BA0h	—	3B80h	—	3B60h	—	3B40h	—	3B20h	—	3B00h	—

图注： 未实现的数据存储单元和寄存器，读为0。

表4-8： PIC18(L)F25/26K83器件BANK 58的特殊功能寄存器映射

3AFFh	—	3ADFh	ADACTPPS	3ABFh	PPSLOCK	3A9Fh	—	3A7Fh	—	3A5Fh	—	3A3Fh	—	3A1Fh	—
3AFEh	—	3ADEh	CLCIN3PPS	3ABEh	—	3A9Eh	—	3A7Eh	—	3A5Eh	—	3A3Eh	—	3A1Eh	—
3AFDh	—	3ADDh	CLCIN2PPS	3ABDh	—	3A9Dh	—	3A7Dh	—	3A5Dh	—	3A3Dh	—	3A1Dh	—
3AFCh	—	3ADCh	CLCIN1PPS	3ABCh	—	3A9Ch	—	3A7Ch	—	3A5Ch	—	3A3Ch	—	3A1Ch	—
3AFBh	—	3ADBh	CLCIN0PPS	3ABBh	—	3A9Bh	—	3A7Bh	—	3A5Bh	RB2I2C	3A3Bh	—	3A1Bh	—
3AFAh	—	3ADAh	MD1SRCPPS	3ABAh	—	3A9Ah	—	3A7Ah	—	3A5Ah	RB1I2C	3A3Ah	—	3A1Ah	—
3AF9h	—	3AD9h	MD1CARHPPS	3AB9h	—	3A99h	—	3A79h	—	3A59h	—	3A39h	—	3A19h	—
3AF8h	—	3AD8h	MD1CARLPPS	3AB8h	—	3A98h	—	3A78h	—	3A58h	—	3A38h	—	3A18h	—
3AF7h	—	3AD7h	CWG3INPPS	3AB7h	—	3A97h	—	3A77h	—	3A57h	IOCBF	3A37h	—	3A17h	RC7PPS
3AF6h	—	3AD6h	CWG2INPPS	3AB6h	—	3A96h	—	3A76h	—	3A56h	IOCBN	3A36h	—	3A16h	RC6PPS
3AF5h	—	3AD5h	CWG1INPPS	3AB5h	—	3A95h	—	3A75h	—	3A55h	IOCBP	3A35h	—	3A15h	RC5PPS
3AF4h	—	3AD4h	SMT2SIGPPS	3AB4h	—	3A94h	—	3A74h	—	3A54h	INLVLB	3A34h	—	3A14h	RC4PPS
3AF3h	—	3AD3h	SMT2WINPPS	3AB3h	—	3A93h	—	3A73h	—	3A53h	SLRCONB	3A33h	—	3A13h	RC3PPS
3AF2h	—	3AD2h	SMT1SIGPPS	3AB2h	—	3A92h	—	3A72h	—	3A52h	ODCONB	3A32h	—	3A12h	RC2PPS
3AF1h	—	3AD1h	SMT1WINPPS	3AB1h	—	3A91h	—	3A71h	—	3A51h	WPUB	3A31h	—	3A11h	RC1PPS
3AF0h	—	3AD0h	CCP4PPS	3AB0h	—	3A90h	—	3A70h	—	3A50h	ANSELB	3A30h	—	3A10h	RC0PPS
3AEFh	—	3ACFh	CCP3PPS	3AAFh	—	3A8Fh	—	3A6Fh	—	3A4Fh	—	3A2Fh	—	3A0Fh	RB7PPS
3AEEh	—	3ACEh	CCP2PPS	3AAEh	—	3A8Eh	—	3A6Eh	—	3A4Eh	—	3A2Eh	—	3A0Eh	RB6PPS
3AEDh	CANRXPPS	3ACDh	CCP1PPS	3AADh	—	3A8Dh	—	3A6Dh	—	3A4Dh	—	3A2Dh	—	3A0Dh	RB5PPS
3AECh	—	3ACCh	T6INPPS	3AACh	—	3A8Ch	—	3A6Ch	—	3A4Ch	—	3A2Ch	—	3A0Ch	RB4PPS
3AEBh	U2CTSPPS	3ACBh	T4INPPS	3AABh	—	3A8Bh	—	3A6Bh	RC4I2C	3A4Bh	—	3A2Bh	—	3A0Bh	RB3PPS
3AEAh	U2RXPPS	3ACAh	T2INPPS	3AAAh	—	3A8Ah	—	3A6Ah	RC3I2C	3A4Ah	—	3A2Ah	—	3A0Ah	RB2PPS
3AE9h	—	3AC9h	T5GPPS	3AA9h	—	3A89h	—	3A69h	—	3A49h	—	3A29h	—	3A09h	RB1PPS
3AE8h	U1CTSPPS	3AC8h	T5CKIPPS	3AA8h	—	3A88h	—	3A68h	—	3A48h	—	3A28h	—	3A08h	RB0PPS
3AE7h	U1RXPPS	3AC7h	T3GPPS	3AA7h	—	3A87h	IOCEF	3A67h	IOCCF	3A47h	IOCAF	3A27h	—	3A07h	RA7PPS
3AE6h	I2C2SDAPPS	3AC6h	T3CKIPPS	3AA6h	—	3A86h	IOCEN	3A66h	IOCCN	3A46h	IOCAN	3A26h	—	3A06h	RA6PPS
3AE5h	I2C2SCLPPS	3AC5h	T1GPPS	3AA5h	—	3A85h	IOCEP	3A65h	IOCCP	3A45h	IOCAP	3A25h	—	3A05h	RA5PPS
3AE4h	I2C1SDAPPS	3AC4h	T1CKIPPS	3AA4h	—	3A84h	INLVLE	3A64h	INLVLC	3A44h	INLVLA	3A24h	—	3A04h	RA4PPS
3AE3h	I2C1SCLPPS	3AC3h	T0CKIPPS	3AA3h	—	3A83h	—	3A63h	SLRCONC	3A43h	SLRCONA	3A23h	—	3A03h	RA3PPS
3AE2h	SPI1SSPPS	3AC2h	INT2PPS	3AA2h	—	3A82h	—	3A62h	ODCONC	3A42h	ODCONA	3A22h	—	3A02h	RA2PPS
3AE1h	SPI1SDIPPS	3AC1h	INT1PPS	3AA1h	—	3A81h	WPUE	3A61h	WPUC	3A41h	WPUA	3A21h	—	3A01h	RA1PPS
3AE0h	SPI1SCKPPS	3AC0h	INT0PPS	3AA0h	—	3A80h	—	3A60h	ANSELC	3A40h	ANSELA	3A20h	—	3A00h	RA0PPS

图注： 未实现的数据存储单元和寄存器，读为0。

表4-9： PIC18(L)F25/26K83器件BANK 57的特殊功能寄存器映射

39FFh	—	39DFh	OSCFRQ	39BFh	—	399Fh	—	397Fh	—	395Fh	WDTU	393Fh	—	391Fh	—
39FEh	—	39DEh	OSCTUNE	39BEh	—	399Eh	—	397Eh	—	395Eh	WDTH	393Eh	—	391Eh	—
39FDh	—	39DDh	OSCEN	39BDh	—	399Dh	—	397Dh	SCANTRIG	395Dh	WDTL	393Dh	—	391Dh	—
39FCh	—	39DCh	OSCSTAT	39BCh	—	399Ch	—	397Ch	SCANCON0	395Ch	WDTCON1	393Ch	—	391Ch	—
39FBh	—	39DBh	OSCCON3	39BBh	—	399Bh	—	397Bh	SCANHADRU	395Bh	WDTCON0	393Bh	—	391Bh	—
39FAh	—	39DAh	OSCCON2	39BAh	—	399Ah	—	397Ah	SCANHADRH	395Ah	—	393Ah	—	391Ah	—
39F9h	—	39D9h	OSCCON1	39B9h	—	3999h	PIE9	3979h	SCANHADRL	3959h	—	3939h	—	3919h	—
39F8h	—	39D8h	CPUDOZE	39B8h	—	3998h	PIE8	3978h	SCANLADRU	3958h	—	3938h	—	3918h	—
39F7h	SCANPR	39D7h	—	39B7h	—	3997h	PIE7	3977h	SCANLADRH	3957h	—	3937h	—	3917h	—
39F6h	—	39D6h	—	39B6h	—	3996h	PIE6	3976h	SCANLADRL	3956h	—	3936h	—	3916h	—
39F5h	—	39D5h	—	39B5h	—	3995h	PIE5	3975h	—	3955h	—	3935h	—	3915h	—
39F4h	DMA2PR	39D4h	—	39B4h	—	3994h	PIE4	3974h	—	3954h	—	3934h	—	3914h	—
39F3h	DMA1PR	39D3h	—	39B3h	—	3993h	PIE3	3973h	—	3953h	—	3933h	—	3913h	—
39F2h	MAINPR	39D2h	—	39B2h	—	3992h	PIE2	3972h	—	3952h	—	3932h	—	3912h	—
39F1h	ISRPR	39D1h	VREGCON(1)	39B1h	—	3991h	PIE1	3971h	—	3951h	—	3931h	—	3911h	—
39F0h	—	39D0h	BORCON	39B0h	—	3990h	PIE0	3970h	—	3950h	—	3930h	—	3910h	—
39EFh	PRLOCK	39CFh	—	39AFh	—	398Fh	—	396Fh	—	394Fh	—	392Fh	—	390Fh	—
39EEh	—	39CEh	—	39AEh	—	398Eh	—	396Eh	—	394Eh	—	392Eh	—	390Eh	—
39EDh	—	39CDh	—	39ADh	—	398Dh	—	396Dh	—	394Dh	—	392Dh	—	390Dh	—
39ECh	—	39CCh	—	39ACh	—	398Ch	—	396Ch	—	394Ch	—	392Ch	—	390Ch	—
39EBh	—	39CBh	—	39ABh	—	398Bh	—	396Bh	—	394Bh	—	392Bh	—	390Bh	—
39EAh	—	39CAh	—	39AAh	—	398Ah	—	396Ah	—	394Ah	—	392Ah	—	390Ah	—
39E9h	—	39C9h	—	39A9h	PIR9	3989h	IPR9	3969h	CRCCON1	3949h	—	3929h	—	3909h	—
39E8h	—	39C8h	—	39A8h	PIR8	3988h	IPR8	3968h	CRCCON0	3948h	—	3928h	—	3908h	—
39E7h	—	39C7h	PMD7	39A7h	PIR7	3987h	IPR7	3967h	CRCXORH	3947h	—	3927h	—	3907h	—
39E6h	NVMCON2	39C6h	PMD6	39A6h	PIR6	3986h	IPR6	3966h	CRCXORL	3946h	—	3926h	—	3906h	—
39E5h	NVMCON1	39C5h	PMD5	39A5h	PIR5	3985h	IPR5	3965h	CRCSHIFTH	3945h	—	3925h	—	3905h	—
39E4h	—	39C4h	PMD4	39A4h	PIR4	3984h	IPR4	3964h	CRCSHIFTL	3944h	—	3924h	—	3904h	—
39E3h	NVMDAT	39C3h	PMD3	39A3h	PIR3	3983h	IPR3	3963h	CRCACCH	3943h	—	3923h	—	3903h	—
39E2h	—	39C2h	PMD2	39A2h	PIR2	3982h	IPR2	3962h	CRCACCL	3942h	—	3922h	—	3902h	—
39E1h	—	39C1h	PMD1	39A1h	PIR1	3981h	IPR1	3961h	CRCDATH	3941h	—	3921h	—	3901h	—
39E0h	NVMADRL	39C0h	PMD0	39A0h	PIR0	3980h	IPR0	3960h	CRCDATL	3940h	—	3920h	—	3900h	—

图注： 未实现的数据存储单元和寄存器，读为0。

注 1： 在LF器件中未实现。

表4-10： PIC18(L)F25/26K83器件BANK 56的特殊功能寄存器映射

38FFh	—	38DFh	—	38BFh	—	389Fh	IVTADU	387Fh	—	385Fh	—	383Fh	—	381Fh	—
38FEh	—	38DEh	—	38BEh	—	389Eh	IVTADH	387Eh	—	385Eh	—	383Eh	—	381Eh	—
38FDh	—	38DDh	—	38BDh	—	389Dh	IVTADL	387Dh	—	385Dh	—	383Dh	—	381Dh	—
38FCh	—	38DCh	—	38BCh	—	389Ch	—	387Ch	—	385Ch	—	383Ch	—	381Ch	—
38FBh	—	38DBh	—	38BBh	—	389Bh	—	387Bh	—	385Bh	—	383Bh	—	381Bh	—
38FAh	—	38DAh	—	38BAh	—	389Ah	—	387Ah	—	385Ah	—	383Ah	—	381Ah	—
38F9h	—	38D9h	—	38B9h	—	3899h	—	3879h	—	3859h	—	3839h	—	3819h	—
38F8h	—	38D8h	—	38B8h	—	3898h	—	3878h	—	3858h	—	3838h	—	3818h	—
38F7h	—	38D7h	—	38B7h	—	3897h	—	3877h	—	3857h	—	3837h	—	3817h	—
38F6h	—	38D6h	—	38B6h	—	3896h	—	3876h	—	3856h	—	3836h	—	3816h	—
38F5h	—	38D5h	—	38B5h	—	3895h	—	3875h	—	3855h	—	3835h	—	3815h	—
38F4h	—	38D4h	—	38B4h	—	3894h	—	3874h	—	3854h	—	3834h	—	3814h	—
38F3h	—	38D3h	—	38B3h	—	3893h	—	3873h	—	3853h	—	3833h	—	3813h	—
38F2h	—	38D2h	—	38B2h	—	3892h	—	3872h	—	3852h	—	3832h	—	3812h	—
38F1h	—	38D1h	—	38B1h	—	3891h	—	3871h	—	3851h	—	3831h	—	3811h	—
38F0h	—	38D0h	—	38B0h	—	3890h	PRODH_SHAD	3870h	—	3850h	—	3830h	—	3810h	—
38EFh	—	38CFh	—	38AFh	—	388Fh	PRODL_SHAD	386Fh	—	384Fh	—	382Fh	—	380Fh	—
38EEh	—	38CEh	—	38AEh	—	388Eh	FSR2H_SHAD	386Eh	—	384Eh	—	382Eh	—	380Eh	—
38EDh	—	38CDh	—	38ADh	—	388Dh	FSR2L_SHAD	386Dh	—	384Dh	—	382Dh	—	380Dh	—
38ECh	—	38CCh	—	38ACh	—	388Ch	FSR1H_SHAD	386Ch	—	384Ch	—	382Ch	—	380Ch	—
38EBh	—	38CBh	—	38ABh	—	388Bh	FSR1L_SHAD	386Bh	—	384Bh	—	382Bh	—	380Bh	—
38EAh	—	38CAh	—	38AAh	—	388Ah	FSR0H_SHAD	386Ah	—	384Ah	—	382Ah	—	380Ah	—
38E9h	—	38C9h	—	38A9h	—	3889h	FSR0L_SHAD	3869h	—	3849h	—	3829h	—	3809h	—
38E8h	—	38C8h	—	38A8h	—	3888h	PCLATU_SHAD	3868h	—	3848h	—	3828h	—	3808h	—
38E7h	—	38C7h	—	38A7h	—	3887h	PCLATH_SHAD	3867h	—	3847h	—	3827h	—	3807h	—
38E6h	—	38C6h	—	38A6h	—	3886h	BSR_SHAD	3866h	—	3846h	—	3826h	—	3806h	—
38E5h	—	38C5h	—	38A5h	—	3885h	WREG_SHAD	3865h	—	3845h	—	3825h	—	3805h	—
38E4h	—	38C4h	—	38A4h	—	3884h	STATUS_SHAD	3864h	—	3844h	—	3824h	—	3804h	—
38E3h	—	38C3h	—	38A3h	—	3883h	SHADCON	3863h	—	3843h	—	3823h	—	3803h	—
38E2h	—	38C2h	—	38A2h	—	3882h	BSR_CSHAD	3862h	—	3842h	—	3822h	—	3802h	—
38E1h	—	38C1h	—	38A1h	—	3881h	WREG_CSHAD	3861h	—	3841h	—	3821h	—	3801h	—
38E0h	—	38C0h	—	38A0h	—	3880h	STATUS_CSHAD	3860h	—	3840h	—	3820h	—	3800h	—

图注： 未实现的数据存储单元和寄存器，读为0。

表4-11： PIC18(L)F25/26K83器件BANK 55的特殊功能寄存器映射

37FFh	CANCON_RO0	37DFh	CANCON_RO2	37BFh	RXM1EIDL	379Fh	CANCON_RO4	377Fh	CANCON_RO6	375Fh	CANCON_RO8	373Fh	TXBIE	371Fh	RXF11EIDL
37FEh	CANSTAT_RO0	37DEh	CANSTAT_RO2	37BEh	RXM1EIDH	379Eh	CANSTAT_RO4	377Eh	CANSTAT_RO6	375Eh	CANSTAT_RO8	373Eh	BIE0	371Eh	RXF11EIDH
37FDh	RXB1D7	37DDh	TXB1D7	37BDh	RXM1SIDL	379Dh	B5D7	377Dh	B3D7	375Dh	B1D7	373Dh	BSEL0	371Dh	RXF11SIDL
37FCh	RXB1D6	37DCh	TXB1D6	37BCh	RXM1SIDH	379Ch	B5D6	377Ch	B3D6	375Ch	B1D6	373Ch	MSEL3	371Ch	RXF11SIDH
37FBh	RXB1D5	37DBh	TXB1D5	37BBh	RXM0EIDL	379Bh	B5D5	377Bh	B3D5	375Bh	B1D5	373Bh	MSEL2	371Bh	RXF10EIDL
37FAh	RXB1D4	37DAh	TXB1D4	37BAh	RXM0EIDH	379Ah	B5D4	377Ah	B3D4	375Ah	B1D4	373Ah	MSEL1	371Ah	RXF10EIDH
37F9h	RXB1D3	37D9h	TXB1D3	37B9h	RXM0SIDL	3799h	B5D3	3779h	B3D3	3759h	B1D3	3739h	MSEL0	3719h	RXF10SIDL
37F8h	RXB1D2	37D8h	TXB1D2	37B8h	RXM0SIDH	3798h	B5D2	3778h	B3D2	3758h	B1D2	3738h	RXFBCON7	3718h	RXF10SIDH
37F7h	RXB1D1	37D7h	TXB1D1	37B7h	RXF5EIDL	3797h	B5D1	3777h	B3D1	3757h	B1D1	3737h	RXFBCON6	3717h	RXF9EIDL
37F6h	RXB1D0	37D6h	TXB1D0	37B6h	RXF5EIDH	3796h	B5D0	3776h	B3D0	3756h	B1D0	3736h	RXFBCON5	3716h	RXF9EIDH
37F5h	RXB1DLC	37D5h	TXB1DLC	37B5h	RXF5SIDL	3795h	B5DLC	3775h	B3DLC	3755h	B1DLC	3735h	RXFBCON4	3715h	RXF9SIDL
37F4h	RXB1EIDL	37D4h	TXB1EIDL	37B4h	RXF5SIDH	3794h	B5EIDL	3774h	B3EIDL	3754h	B1EIDL	3734h	RXFBCON3	3714h	RXF9SIDH
37F3h	RXB1EIDH	37D3h	TXB1EIDH	37B3h	RXF4EIDL	3793h	B5EIDH	3773h	B3EIDH	3753h	B1EIDH	3733h	RXFBCON2	3713h	RXF8EIDL
37F2h	RXB1SIDL	37D2h	TXB1SIDL	37B2h	RXF4EIDH	3792h	B5SIDL	3772h	B3SIDL	3752h	B1SIDL	3732h	RXFBCON1	3712h	RXF8EIDH
37F1h	RXB1SIDH	37D1h	TXB1SIDH	37B1h	RXF4SIDL	3791h	B5SIDH	3771h	B3SIDH	3751h	B1SIDH	3731h	RXFBCON0	3711h	RXF8SIDL
37F0h	RXB1CON	37D0h	TXB1CON	37B0h	RXF4SIDH	3790h	B5CON	3770h	B3CON	3750h	B1CON	3730h	SDFLC	3710h	RXF8SIDH
37EFh	CANCON_RO1	37CFh	CANCON_RO3	37AFh	RXF3EIDL	378Fh	CANCON_RO5	376Fh	CANCON_RO7	374Fh	CANCON_RO9	372Fh	RXF15EIDL	370Fh	RXF7EIDL
37EEh	CANSTAT_RO1	37CEh	CANSTAT_RO3	37AEh	RXF3EIDH	378Eh	CANSTAT_RO5	376Eh	CANSTAT_RO7	374Eh	CANSTAT_RO9	372Eh	RXF15EIDH	370Eh	RXF7EIDH
37EDh	TXB0D7	37CDh	TXB2D7	37ADh	RXF3SIDL	378Dh	B4D7	376Dh	B2D7	374Dh	B0D7	372Dh	RXF15SIDL	370Dh	RXF7SIDL
37ECh	TXB0D6	37CCh	TXB2D6	37ACh	RXF3SIDH	378Ch	B4D6	376Ch	B2D6	374Ch	B0D6	372Ch	RXF15SIDH	370Ch	RXF7SIDH
37EBh	TXB0D5	37CBh	TXB2D5	37ABh	RXF2EIDL	378Bh	B4D5	376Bh	B2D5	374Bh	B0D5	372Bh	RXF14EIDL	370Bh	RXF6EIDL
37EAh	TXB0D4	37CAh	TXB2D4	37AAh	RXF2EIDH	378Ah	B4D4	376Ah	B2D4	374Ah	B0D4	372Ah	RXF14EIDH	370Ah	RXF6EIDH
37E9h	TXB0D3	37C9h	TXB2D3	37A9h	RXF2SIDL	3789h	B4D3	3769h	B2D3	3749h	B0D3	3729h	RXF14SIDL	3709h	RXF6SIDL
37E8h	TXB0D2	37C8h	TXB2D2	37A8h	RXF2SIDH	3788h	B4D2	3768h	B2D2	3748h	B0D2	3728h	RXF14SIDH	3708h	RXF6SIDH
37E7h	TXB0D1	37C7h	TXB2D1	37A7h	RXF1EIDL	3787h	B4D1	3767h	B2D1	3747h	B0D1	3727h	RXF13EIDL	3707h	RXFCON1
37E6h	TXB0D0	37C6h	TXB2D0	37A6h	RXF1EIDH	3786h	B4D0	3766h	B2D0	3746h	B0D0	3726h	RXF13EIDH	3706h	RXFCON0
37E5h	TXB0DLC	37C5h	TXB2DLC	37A5h	RXF1SIDL	3785h	B4DLC	3765h	B2DLC	3745h	B0DLC	3725h	RXF13SIDL	3705h	BRGCON3
37E4h	TXB0EIDL	37C4h	TXB2EIDL	37A4h	RXF1SIDH	3784h	B4EIDL	3764h	B2EIDL	3744h	B0EIDL	3724h	RXF13SIDH	3704h	BRGCON2
37E3h	TXB0EIDH	37C3h	TXB2EIDH	37A3h	RXF0EIDL	3783h	B4EIDH	3763h	B2EIDH	3743h	B0EIDH	3723h	RXF12EIDL	3703h	BRGCON1
37E2h	TXB0SIDL	37C2h	TXB2SIDL	37A2h	RXF0EIDH	3782h	B4SIDL	3762h	B2SIDL	3742h	B0SIDL	3722h	RXF12EIDH	3702h	TXERRCNT
37E1h	TXB0SIDH	37C1h	TXB2SIDH	37A1h	RXF0SIDL	3781h	B4SIDH	3761h	B2SIDH	3741h	B0SIDH	3721h	RXF12SIDL	3701h	RXERRCNT
37E0h	TXB0CON	37C0h	TXB2CON	37A0h	RXF0SIDH	3780h	B4CON	3760h	B2CON	3740h	B0CON	3720h	RXF12SIDH	3700h	CIOCON

图注： 未实现的数据存储单元和寄存器，读为0。

4.5.5 STATUS寄存器

如寄存器4-2所示，STATUS寄存器包含ALU的算术运算状态。与任何其他SFR一样，STATUS寄存器可以作为任何指令的操作数。

如果一条影响Z、DC、C、OV或N位的指令以STATUS寄存器作为目标寄存器，则不会写入指令的结果，而是根据所执行的指令更新STATUS寄存器。因此，当执行一条将STATUS寄存器作为其目标寄存器的指令时，运行结果可能会与预想的不同。例如，CLRF STATUS将Z位置1并保持其余状态位不变（0uuu u1uu）。

建议仅使用BCF、BSF、SWAPF、MOVFF、MOVWF和MOVFFL指令来改变STATUS寄存器，因为这些指令不会影响STATUS寄存器中的Z、C、DC、OV或N位。

有关其他不影响状态位的指令，请参见**第42.2节“扩展指令集”**和表42-3中的指令集汇总。

注：	在减法运算中，C位和DC位分别作为借位位和半借位位。

4.5.6 调用影子寄存器

使用CALL、CALLW和RCALL指令时，WREG、BSR和STATUS自动保存在硬件中，并可以使用WREG_CSHAD、BSR_CSHAD和STATUS_CSHAD寄存器进行访问。

4.6 寄存器定义：状态寄存器

寄存器4-2： STATUS：状态寄存器

U-0	R-1/q	R-1/q	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	$\overline{TO}$	$\overline{PD}$	N	OV	Z	DC	C
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **未实现：**读为0

bit 6 **$\overline{TO}$：**超时位
1 = 上电时置1，或者通过执行CLRWDT或SLEEP指令置1
0 = 发生了WDT超时

bit 5 **$\overline{PD}$：**掉电位
1 = 上电时置1，或者通过执行CLRWDT指令置1
0 = 通过执行SLEEP指令置1

bit 4 **N：**负标志位用于有符号的算术运算（以二进制补码方式进行）；用于表示结果是否为负，（ALU MSb = 1）。
1 = 结果为负
0 = 结果为正

bit 3 **OV：**溢出位用于有符号的算术运算（以二进制补码方式进行）；用于表示运算结果是否溢出了7位二进制数的范围，溢出导致符号位（bit 7）的状态发生改变。
1 = 当前有符号算术运算中发生溢出
0 = 未发生溢出

bit 2 **Z：**全零标志位
1 = 算术运算或逻辑运算的结果为零
0 = 算术运算或逻辑运算的结果不为零

bit 1 **DC：**半进位/$\overline{借位}$位（ADDWF、ADDLW、SUBLW和SUBWF指令）**(1)**
1 = 结果的第4个低位发生了进位
0 = 结果的第4个低位未发生进位

bit 0 **C：**进位/$\overline{借位}$位（ADDWF、ADDLW、SUBLW和SUBWF指令）**(1,2)**
1 = 结果的最高有效位发生了进位
0 = 结果的最高有效位未发生进位

注 1： 对于$\overline{借位}$，极性是相反的。减法是通过加上第二个操作数的二进制补码来执行的。
2： 对于移位（RRF和RLF）指令，此位中将装入源寄存器的最高位或最低位。

4.7 数据寻址模式

注： 当使能PIC18扩展指令集时，核心PIC18指令集中某些指令的执行方式会发生改变。更多信息，请参见**第4.8节“数据存储器和扩展指令集”**。

程序存储器只能用一种方式寻址（通过程序计数器），而数据存储空间中的信息可用多种方式寻址。大部分指令的寻址模式都是固定的。其他指令可能使用最多三种模式，根据它们所使用的操作数和是否使能了扩展指令集而定。

这些寻址模式为：

- 固有寻址
- 立即数寻址
- 直接寻址
- 间接寻址

当使能了扩展指令集（XINST配置位 = 1）时，还可使用另外一种寻址模式，即立即数变址寻址模式。关于该模式操作的详细信息，请参见**第4.8.1节“使用立即数偏移量进行变址寻址”**。

4.7.1 固有寻址和立即数寻址

很多PIC18控制指令根本不需要任何参数。执行这些指令要么对整个器件造成影响，要么仅隐式地针对一个寄存器进行操作。此寻址模式就是固有寻址。该模式的示例包括SLEEP、RESET和DAW。

其他指令的工作方式与此类似，但需要操作码中有其他显式的参数。由于指令需要一些立即数作为参数，该方法被称为立即数寻址模式。该模式的示例包括ADDLW和MOVLW，它们分别向W寄存器添加或移入立即数值。其他示例有CALL和GOTO，它们包含一个20位的程序存储器地址。

4.7.2 直接寻址

直接寻址在操作码中指定操作的全部或部分源地址和/或目标地址。这些选项由指令附带的参数指定。

在核心PIC18指令集中，针对位和针对字节的指令默认情况下使用直接寻址。所有这些指令都包含某个8位的立即数地址作为其最低有效字节。此地址指定数据RAM的某个存储区中寄存器的地址（**第4.5.2节“通用寄存器文件”**）或快速操作存储区（**第4.5.4节“快速操作存储区”**）中作为指令数据源的单元地址。

快速操作RAM位“a”决定地址的解析方式。当a为1时，BSR（**第4.5.1节“存储区选择寄存器（BSR）”**）的内容将和指令中的直接地址一起用于确定寄存器的完整14位地址。当a为0时，此直接地址将被解析为快速操作存储区中的一个寄存器。使用快速操作RAM的寻址模式有时也被称为直接强制寻址模式。

有几条指令，例如MOVFFL，在操作码中包含完整的14位地址（源地址或目标地址）。在这些情况下，BSR被完全忽略。

保存操作结果的目标寄存器由目标位d确定。当d为1时，结果被存回源寄存器并覆盖原来的内容。当d为0时，结果被存储在W寄存器中。没有d参数的指令的目标寄存器隐含在指令中，这些指令的目标寄存器是正在操作的目标寄存器或W寄存器。

4.7.3 间接寻址

间接寻址允许用户访问数据存储器中的单元而无需在指令中给出一个固定的地址。这种寻址模式是通过使用文件选择寄存器（File Select Register，FSR）作为指向要读写单元的指针实现的。由于FSR本身作为特殊文件寄存器位于RAM中，因此也可在程序控制中对其直接操作。这使得FSR对于在数据存储器中实现诸如表和数组等数据结构时非常有用。

也可以使用间接文件操作数（INDF）进行寄存器间接寻址。这种操作允许自动递增、递减或偏移指针，从而自动操作指针的值。它通过使用循环提高代码执行效率，如例4-6所示的清零整个RAM存储区的操作。

例4-6： 如何使用间接寻址清零RAM（BANK 1）

```
        LFSR   FSR0, 100h ;
NEXT    CLRF   POSTINC0   ; Clear INDF
                          ; register then
                          ; inc pointer
        BTFSS  FSR0H, 1   ; All done with
                          ; Bank1?
        BRA    NEXT       ; NO, clear next
CONTINUE                  ; YES, continue
```

4.7.3.1 FSR寄存器和INDF操作数

间接寻址的核心是三组寄存器：FSR0、FSR1和FSR2。每组寄存器都含有一对8位寄存器：FSRnH和FSRnL。FSRnH寄存器的高2位未使用，因此每对FSR只保存一个14位值，从而可以线性寻址数据存储器的整个空间。因此，FSR寄存器对被用作数据存储单元的地址指针。

间接寻址是通过一组间接文件操作数（INDF0至INDF2）完成的。这些操作数可被看作“虚拟”寄存器：它们被映射到SFR空间而不是物理实现的。对特定的INDF寄存器执行读或写操作实际上访问的是与之对应的一对FSR寄存器所寻址的数据。例如，读INDF1就是读FSR1H:FSR1L指向地址中的数据。使用INDF寄存器作为操作数的指令实际上使用相应FSR的内容作为指向指令目标地址的指针。INDF操作数只是使用指针的一种简便方法。

由于间接寻址使用完整的14位地址，因此没有必要进行数据RAM分区。所以，BSR的当前内容和快速操作RAM位对确定目标地址没有影响。

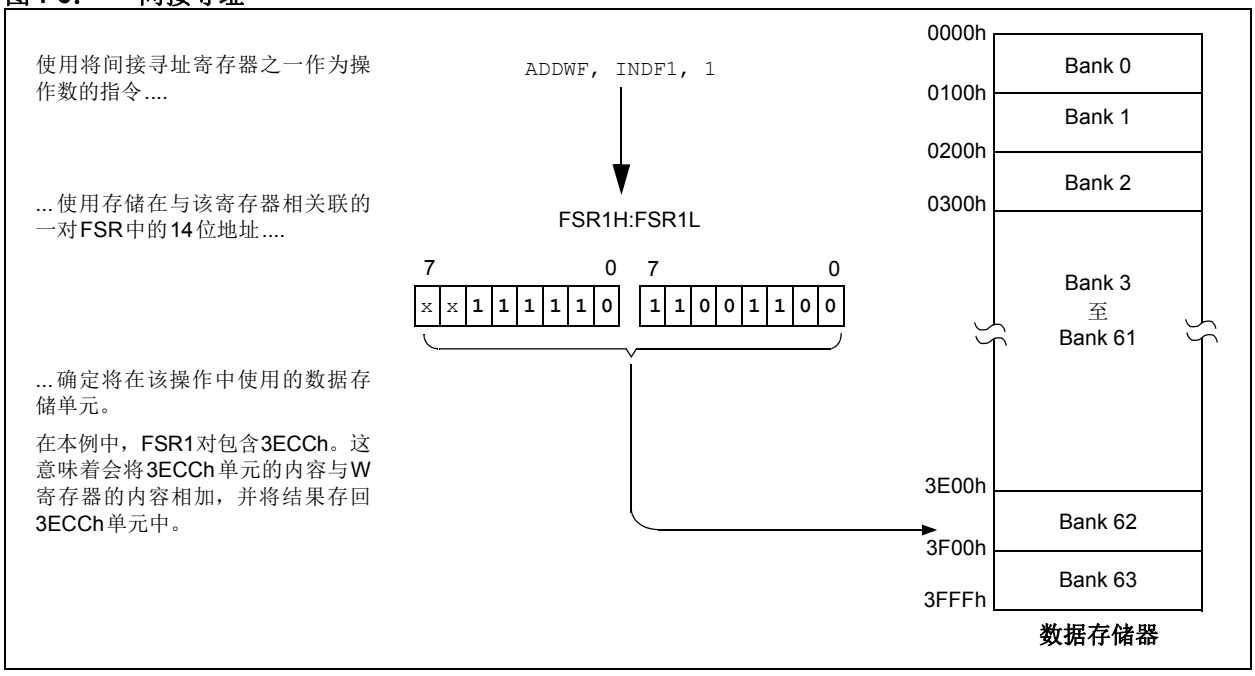
4.7.3.2 FSR寄存器和POSTINC、POSTDEC、PREINC以及PLUSW

除了INDF操作数之外，每对FSR寄存器还有4个额外的间接操作数。和INDF一样，它们也都是不能直接读写的“虚拟”寄存器。访问这些寄存器实际上访问的是与之相关的一对FSR寄存器，并对其所存储的值进行特定的操作。具体包括：

- POSTDEC：访问FSR的值，然后将其自动减1
- POSTINC：访问FSR的值，然后将其自动加1
- PREINC：将FSR的值自动加1，然后在操作中使用该值
- PLUSW：将W寄存器中的有符号值（从-127至128）与FSR中的值相加，并在操作中使用得到的新值

在本文中，访问INDF寄存器使用相关FSR寄存器中的值（不会更改此值）。同样，访问PLUSW寄存器是将W寄存器中的值作为FSR的偏移量；该操作不会改变W或FSR中的值。访问其他虚拟寄存器均会更改FSR寄存器的值。

图4-6： 间接寻址



使用POSTDEC、POSTINC和PREINC对FSR进行操作会影响整对寄存器；即FSRnL寄存器从FFh计满返回到00h并向FSRnH寄存器进位。但这些操作的结果不会更改STATUS寄存器中的任何标志位（如Z、N和OV位）的值。

PLUSW寄存器可用于在数据存储空间中实现变址寻址。通过操作W寄存器中的值，用户可以访问相对当前指针地址有固定偏移量的地址单元。在某些应用中，该功能可用于在数据存储器内部实现某些强大的程序控制结构，如软件堆栈。

4.7.3.3 通过FSR对其他FSR进行操作

在某些特殊情况下，间接寻址操作以其他FSR或虚拟寄存器作为目标。例如，使用FSR指向一个虚拟寄存器会导致操作不成功。假设如下特殊情况：FSR0H:FSR0L寄存器保存的是INDF1的地址3FE7h。尝试使用INDF0作为操作数读取INDF1的值，将返回00h。尝试使用INDF0作为操作数写入INDF1，将会导致执行一条NOP指令。

另一方面，使用虚拟寄存器对一对FSR寄存器进行写操作可能会产生与预期不同的结果。在这些情况下，会将值写入一对FSR寄存器，但FSR不会递增或递减。因此，写入INDF2或POSTDEC2寄存器时会将同样的值写入FSR2H:FSR2L。

由于FSR是映射到SFR空间中的物理寄存器，所以可以通过所有直接寻址来操作它们。用户在使用这些寄存器时应该特别小心，尤其是在代码使用间接寻址时。

同样，通常允许通过间接寻址对所有其他SFR进行操作。用户在进行此类操作时应该特别小心，以免不小心更改设置从而影响器件操作。

4.8 数据存储器和扩展指令集

使能PIC18扩展指令集（XINST配置位 = 1）显著改变了数据存储器及其寻址的某些方面。具体地说，许多核心PIC18指令以不同的方式使用快速操作存储区；这是因为引入了对数据存储空间的全新寻址模式。

同样需要了解哪些部分保持不变。数据存储空间的大小及其线性寻址模式都不会改变。SFR映射也保持不变。核心PIC18指令也仍然以直接和间接寻址模式进行操作；固有和立即数寻址指令操作照旧。FSR0和FSR1的间接寻址模式也保持不变。

4.8.1 使用立即数偏移量进行变址寻址

使能PIC18扩展指令集将更改使用FSR2寄存器对在快速操作RAM中进行间接寻址的方式。在适当的条件下，使用快速操作存储区的指令（即，大多数针对位和针对字节的指令）可以利用指令中的偏移量来执行变址寻址。这种特殊的寻址模式被称为使用立即数偏移量的变址寻址或立即数变址寻址模式。

使用扩展指令集时，这种寻址模式有如下要求：

- 强制使用快速操作存储区（a = 0）
- 文件地址参数小于或等于5Fh。

在这些条件下，指令的文件地址不会被解析为地址的低字节（在直接寻址中和BSR一起使用），或快速操作存储区中的8位地址，而是被解析为由FSR2指定的地址指针的偏移量。将该偏移量与FSR2的内容相加以获取操作的目标地址。

4.8.2 受立即数变址寻址模式影响的指令

任何使用直接寻址模式的核心PIC18指令均会受到立即数变址寻址模式的潜在影响，包括所有针对字节和针对位的指令，或标准PIC18指令集中几乎一半的指令。只使用固有寻址或立即数寻址模式的指令不受影响。

此外，如果针对字节和针对位的指令不使用快速操作存储区（快速操作RAM位为1），或者包含60h或以上的文件地址，它们也不受影响。符合这些条件的指令会像以前一样执行。图4-7给出了当使能扩展指令集时，各种寻址模式之间的对比。

那些想要在立即数变址寻址模式中使用针对位或针对字节的指令的用户，应该注意此模式下汇编语法的改变。**第42.2.1节“扩展指令语法”**中对此进行了更详细的说明。

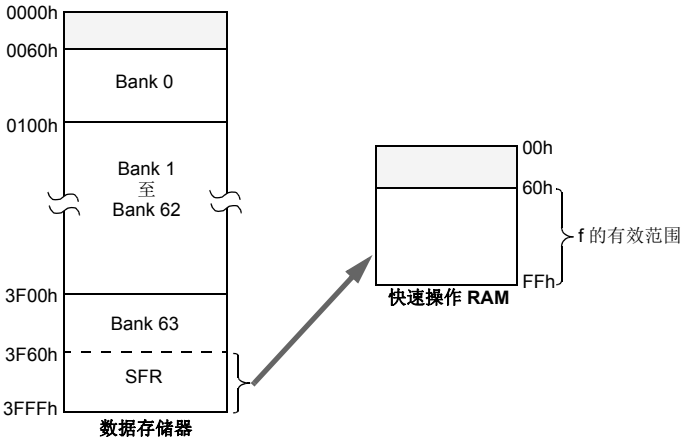
图4-7： 针对位和针对字节的指令的寻址模式对比（使能了扩展指令集）

示例指令：ADDWF, f, d, a（操作码：0010 01da ffff ffff）

当 a = 0 且 f ≥ 60h 时：

此指令以直接强制模式执行。“f”被解析为快速操作RAM中060h至0FFh之间的单元地址，该地址也是数据存储器的3F60h至3FFFh（Bank 63）。

不可用此模式寻址地址低于60h的单元。



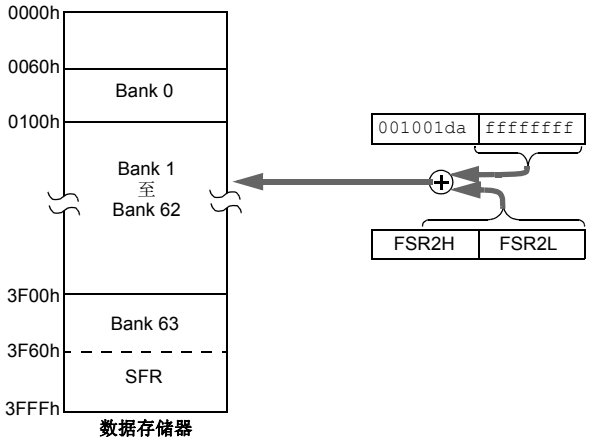
当 a = 0 且 f ≤ 5Fh 时：

此指令以立即数变址寻址模式执行。“f”被解析为FSR2中地址值的偏移量。将这两个值相加可以得到指令的目标寄存器的地址。此地址可以在数据存储空间的任何地方。

注意在此模式中，正确的语法如下：

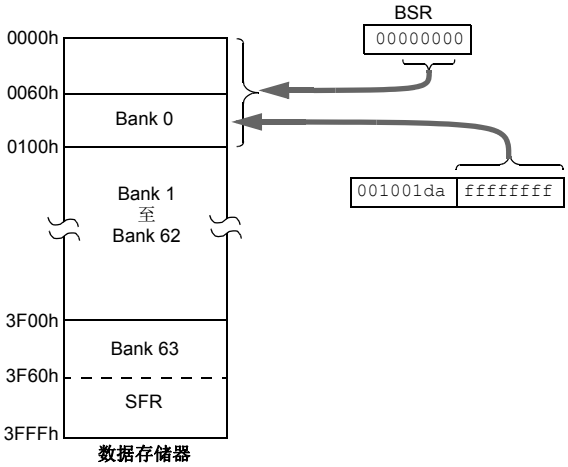
ADDWF [k], d

其中“k”就是“f”。



当 a = 1（f 可为任何值）时：

指令以直接寻址模式（也称为直接长地址寻址模式）执行。“f”被解析为数据存储空间的63个存储区之一中的一个单元地址。存储区由存储区选择寄存器（BSR）指定。此地址可以位于数据存储空间的任何已实现存储区中。



4.8.3 在立即数变址寻址模式下映射快速操作存储区

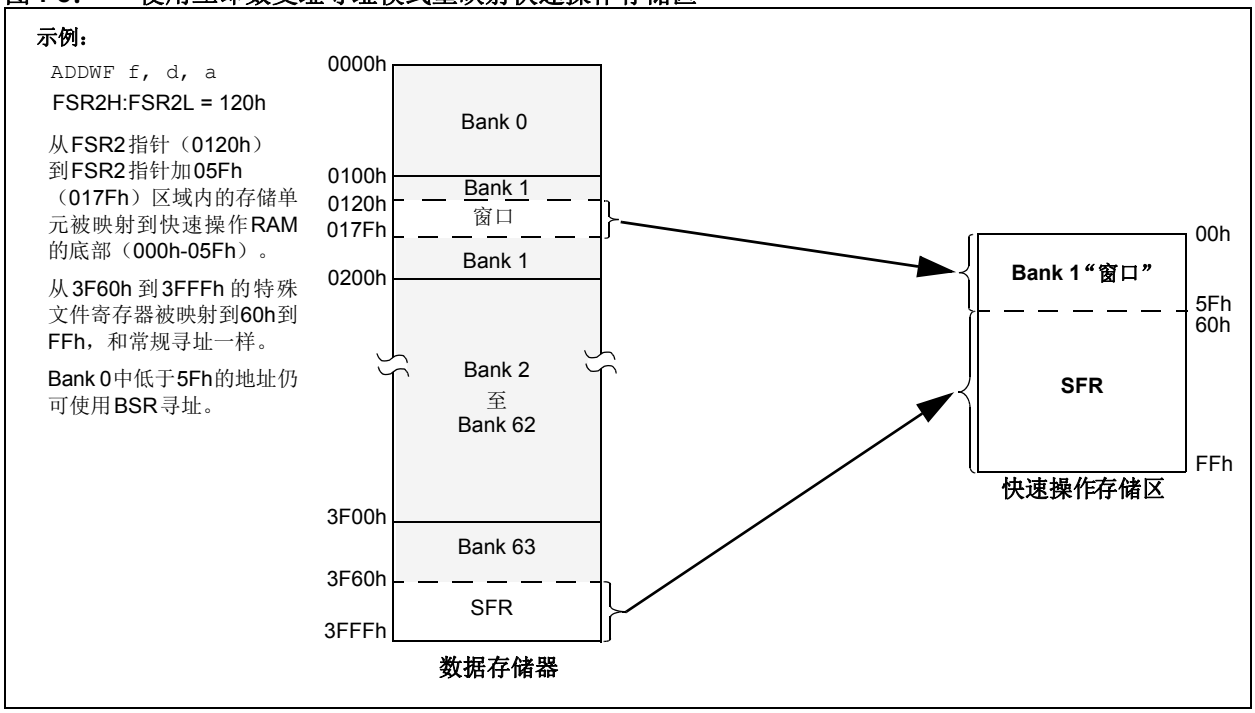
使用立即数变址寻址模式能有效改变快速操作RAM的前96个单元（00h至5Fh）的映射方式。此模式映射Bank 0的内容和由用户定义的、可位于数据存储空间中任何地方的“窗口”内容，而不仅仅映射Bank 0底部的内容。FSR2的值定义映射到窗口的地址的下边界，而上边界则由FSR2加95（5Fh）定义。地址大于5Fh的快速操作RAM的映射方法如前所述。（见**第4.5.4节“快速操作存储区”**）。图4-8给出了在此寻址模式下重映射快速操作存储区的示例。

快速操作存储区的重映射*仅*适用于使用立即数变址寻址模式的操作。使用BSR（快速操作RAM位为1）的操作和以前一样继续使用直接寻址模式。

4.9 PIC18指令执行和扩展指令集

使能扩展指令集将向现有PIC18指令集额外添加8条命令。这些指令按**第42.2节“扩展指令集”**所述执行。

图4-8： 使用立即数变址寻址模式重映射快速操作存储区



5.0 器件配置

器件配置包括配置字、用户ID、器件ID、版本ID、器件信息区（DIA）（见**第5.7节“器件信息区”**）和器件配置信息（DCI）区域（见**第5.8节“器件配置信息”**）。

5.1 配置字

配置字共有六位，允许用户通过多种选择（振荡器、复位和存储器保护选项）来设置器件。这些位实现为300000h至300008h处的配置字1至配置字6。

5.2 寄存器定义：配置字

寄存器5-1： 配置字1L（30 0000h）

U-1	R/W-1	R/W-1	R/W-1	U-1	R/W-1	R/W-1	R/W-1
—	RSTOSC<2:0>			—	FEXTOSC<2:0>		
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7 **未实现：**读为1

bit 6-4 **RSTOSC<2:0>：**COSC的上电默认值位
111 = EXTOSC根据FEXTOSC<2:0>位操作
110 = HFFRQ = 4 MHz、CDIV = 4:1的HFINTOSC
101 = LFINTOSC
100 = SOSC
011 = 保留
010 = 采用4x PLL的EXTOSC； EXTOSC根据FEXTOSC<2:0>位操作
001 = 保留
000 = HFFRQ = 64 MHz、CDIV = 1:1的HFINTOSC；将COSC/NOSC复位为3'b110

bit 3 **未实现：**读为1

bit 2-0 **FEXTOSC<2:0>：**FEXTOSC外部振荡器模式选择位
111 = EC（外部时钟）高于8 MHz； PFM设置为高功率
110 = EC（外部时钟）介于500 kHz和8 MHz之间； PFM设置为中等功率
101 = EC（外部时钟）低于500 kHz； PFM设置为低功率
100 = 不使能振荡器
011 = 保留（不要使用）
010 = HS（晶振）高于8 MHz； PFM设置为高功率
001 = XT（晶振）介于500 kHz和8 MHz之间； PFM设置为中等功率
000 = LP（晶振）优化为32.768 kHz； PFM设置为低功率

寄存器5-2：　　配置字1H（30 0001h）

U-1	U-1	R/W-1	U-1	R/W-1	U-1	R/W-1	R/W-1
—	—	FCMEN	—	CSWEN	—	PR1WAY	$\overline{\text{CLKOUTEN}}$
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7-6　**未实现：**读为1

bit 5　**FCMEN：**故障保护时钟监视器使能位
1 = 使能FSCM定时器
0 = 禁止FSCM定时器

bit 4　**未实现：**读为1

bit 3　**CSWEN：**时钟切换使能位
1 = 允许写入NOSC和NDIV
0 = 用户软件无法更改NOSC和NDIV位

bit 2　**未实现：**读为1

bit 1　**PR1WAY：**PRLOCKED一次置1使能位
1 = PRLOCKED位只可清零和置1一次；一个清零/置1周期后优先级寄存器将保持锁定
0 = PRLOCKED位可重复清零和置1（需要解锁序列）

bit 0　**$\overline{\text{CLKOUTEN}}$：**时钟输出使能位
如果FEXTOSC<2:0> = EC（高、中或低）或未使能：
1 = 禁止CLKOUT功能；OSC2为I/O或振荡器功能
0 = 使能CLKOUT功能；OSC2为FOSC/4时钟
其他情况：
忽略此位。

寄存器5-3：　　配置字2L（30 0002h）

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
BOREN<1:0>		$\overline{\text{LPBOREN}}$	IVT1WAY	MVECEN	PWRTS<1:0>		MCLRE
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **BOREN<1:0>**：欠压复位使能位
使能时，欠压复位电压（V_{BOR}）通过BORV位设置。
11 = 使能欠压复位；忽略SBOREN位
10 = 欠压复位在运行时使能，在休眠时禁止；忽略SBOREN位
01 = 根据SBOREN使能欠压复位
00 = 禁止欠压复位

bit 5 **$\overline{\text{LPBOREN}}$**：低功耗BOR使能位
1 = 禁止低功耗BOR
0 = 使能低功耗BOR

bit 4 **IVT1WAY**：IVTLOCK位一次置1使能位
1 = IVTLOCKED位只可清零和置1一次；一个清零/置1周期后IVT寄存器将保持锁定
0 = IVTLOCK ED位可重复清零和置1（需要解锁序列）

bit 3 **MVECEN**：多向量使能位
1 = 使能多向量；将向量表用于中断
0 = 传统中断行为

bit 2-1 **PWRTS<1:0>**：上电延时定时器选择位
11 = 禁止PWRT
10 = PWRT设置为64 ms
01 = PWRT设置为16 ms
00 = PWRT设置为1 ms

bit 0 **MCLRE**：主复位（$\overline{\text{MCLR}}$）使能位
如果LVP = 1：
RE3引脚功能为$\overline{\text{MCLR}}$
如果LVP = 0：
1 = $\overline{\text{MCLR}}$引脚功能为$\overline{\text{MCLR}}$
0 = $\overline{\text{MCLR}}$引脚功能为端口定义功能

寄存器5-4：　　配置字2H（30 0003h）

R/W-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{XINST}}$	—	$\overline{\text{DEBUG}}$	STVREN	PPS1WAY	$\overline{\text{ZCD}}$	BORV<1:0>(1)	
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7　**$\overline{\text{XINST}}$：**扩展指令集使能位
1 = 禁止扩展指令集和索引寻址模式（传统模式）
0 = 使能扩展指令集和索引寻址模式

bit 6　**未实现：**读为1

bit 5　**$\overline{\text{DEBUG}}$：**调试器使能位
1 = 禁止后台调试器
0 = 使能后台调试器

bit 4　**STVREN：**堆栈上溢/下溢复位使能位
1 = 堆栈上溢或下溢将导致复位
0 = 堆栈上溢或下溢不会导致复位

bit 3　**PPS1WAY：**PPSLOCKED一次置1使能位
1 = PPSLOCKED位只可清零和置1一次；一个清零/置1周期后PPS寄存器将保持锁定
0 = PPSLOCKED位可重复清零和置1（需要解锁序列）

bit 2　**$\overline{\text{ZCD}}$：**过零检测使能位
1 = 禁止ZCD；可通过将ZCDCON寄存器的SEN位置1来使能ZCD
0 = 始终使能ZCD

bit 1-0　**BORV<1:0>：**欠压复位电压选择位(1)
PIC18F25/26K83器件：
11 = 欠压复位电压（V_{BOR}）设置为2.45V
10 = 欠压复位电压（V_{BOR}）设置为2.45V
01 = 欠压复位电压（V_{BOR}）设置为2.7V
00 = 欠压复位电压（V_{BOR}）设置为2.85V
PIC18LF25/26K83器件：
11 = 欠压复位电压（V_{BOR}）设置为1.90V
10 = 欠压复位电压（V_{BOR}）设置为2.45V
01 = 欠压复位电压（V_{BOR}）设置为2.7V
00 = 欠压复位电压（V_{BOR}）设置为2.85V

注 1： 对于工作在16 MHz或高于16 MHz的情况，建议使用更高电压设置。

寄存器5-5：　　配置字3L（30 0004h）

U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	WDTE<1:0>		WDTCPS<4:0>				
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7　　**未实现：**读为1

bit 6-5　　**WDTE<1:0>：**WDT工作模式位

00 = 禁止WDT，忽略SWDTEN
01 = 通过WDTCON0中的SWDTEN位使能/禁止WDT
10 = WDT在唤醒状态下使能，在休眠状态下暂停；忽略SWDTEN
11 = 使能WDT（无论是否处于休眠状态）；忽略SWDTEN

bit 4-0　　**WDTCPS<4:0>：**WDT周期选择位

WDTCPS<4:0>	POR时的WDTPS				对WDTPS进行软件控制？
	值	分频比		典型超时（FIN=31 kHz）	
00000	00000	1:32	2^5	1 ms	否
00001	00001	1:64	2^6	2 ms	
00010	00010	1:128	2^7	4 ms	
00011	00011	1:256	2^8	8 ms	
00100	00100	1:512	2^9	16 ms	
00101	00101	1:1024	2^{10}	32 ms	
00110	00110	1:2048	2^{11}	64 ms	
00111	00111	1:4096	2^{12}	128 ms	
01000	01000	1:8192	2^{13}	256 ms	
01001	01001	1:16384	2^{14}	512 ms	
01010	01010	1:32768	2^{15}	1s	
01011	01011	1:65536	2^{16}	2s	
01100	01100	1:131072	2^{17}	4s	
01101	01101	1:262144	2^{18}	8s	
01110	01110	1:524299	2^{19}	16s	
01111	01111	1:1048576	2^{20}	32s	
10000	10000	1:2097152	2^{21}	64s	
10001	10001	1:4194304	2^{22}	128s	
10010	10010	1:8388608	2^{23}	256s	
10011 ... 11110	10011 ... 11110	1:32	2^5	1 ms	否
11111	01011	1:65536	2^{16}	2s	是

寄存器5-6：　　配置字3H（30 0005h）

U-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	WDTCCS<2:0>			WDTCWS<2:0>		
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7-6　　**未实现：**读为1

bit 5-3　　**WDTCCS<2:0>：**WDT输入时钟选择器位

如果WDTE<1:0>熔丝 = 2'b00：
忽略这些位。

其他情况：
000 = WDT参考时钟为31.0 kHz LFINTOSC
001 = WDT参考时钟为31.25 kHz MFINTOSC
010 = WDT参考时钟为SOSC
011 = 保留（默认为LFINTOSC）
•
•
110 = 保留（默认为LFINTOSC）
111 = 软件控制

bit 2-0　　**WDTCWS<2:0>：**WDT窗口选择位

WDTCWS<2:0>	POR时的窗口			对窗口进行软件控制？	需要密钥访问？
	值	窗口延时时间百分比	窗口打开时间百分比		
000	000	87.5	12.5	否	是
001	001	75	25		
010	010	62.5	37.5		
011	011	50	50		
100	100	37.5	62.5		
101	101	25	75		
110	111	n/a	100		
111	111	n/a	100	是	否

寄存器5-7：　　配置字4L（30 0006h）

R/W-1	U-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{WRTAPP}}$ (1)	—	—	$\overline{\text{SAFEN}}$ (1)	$\overline{\text{BBEN}}$ (1)	BBSIZE<2:0> (2)		
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7　　**$\overline{\text{WRTAPP}}$：**应用程序块写保护位(1)
1 = 应用程序块不受写保护
0 = 应用程序块受写保护

bit 6-5　　**未实现：**读为1

bit 4　　**$\overline{\text{SAFEN}}$：**存储区闪存使能位(1)
1 = 禁止SAF
0 = 使能SAF

bit 3　　**$\overline{\text{BBEN}}$：**引导块使能位(1)
1 = 禁止引导块
0 = 使能引导块

bit 2-0　　**BBSIZE<2:0>：**引导块大小选择位(2)
请参见表5-1。

注 1： 这些位以粘住位的形式实现。保护功能通过ICSP™或自写使能后，只能通过批量擦除操作复位。
2： BBSIZE<2:0>位只能在$\overline{\text{BBEN}}$ = 1时更改。一旦$\overline{\text{BBEN}}$=0，BBSIZE<2:0>只能通过批量擦除操作来更改。

表5-1：　　引导块大小位

$\overline{\text{BBEN}}$	BBSIZE[2:0]	引导块大小（字）	END_ADDRESS_BOOT	器件大小(1)	
				16k	32k
1	xxx	0	—	X	X
0	111	512	00 03FFh	X	X
0	110	1024	00 07FFh	X	X
0	101	2048	00 0FFFh	X	X
0	100	4096	00 1FFFh	X	X
0	011	8192	00 3FFFh	X	X
0	010	16384	00 7FFFh	—	X
0	001	32768	00 FFFFh	注2	
0	000	32768	00 FFFFh	—	—

注 1： 对于每个器件，引用的器件存储器容量规范在表4-1中列出。
2： 最大引导块大小是用户程序存储器大小的一半。如果选择的最大引导块大小超过最大值，则会默认设为PFM的一半。例如，在32 kW器件上，BBSIZE = 000至BBSIZE = 010的所有设置会将引导块大小默认设为16 kW。

寄存器5-8：　　配置字4H（30 0007h）

U-1	U-1	R/W-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	LVP(2)	—	$\overline{\text{WRTSAF}}$ (1,3)	$\overline{\text{WRTD}}$ (1,4)	$\overline{\text{WRTC}}$ (1)	$\overline{\text{WRTB}}$(1,5)
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7-6　**未实现：**读为1

bit 5　**LVP：**低电压编程使能位(2)
1 = 使能低电压编程。$\overline{\text{MCLR}}$/VPP引脚功能为$\overline{\text{MCLR}}$。忽略MCLRE（寄存器5-3）。
0 = 必须使用$\overline{\text{MCLR}}$/VPP上的HV进行编程。

bit 4　**未实现：**读为1

bit 3　**$\overline{\text{WRTSAF}}$：**存储区闪存（SAF）写保护位(1,3)
1 = SAF不受写保护
0 = SAF受写保护

bit 2　**$\overline{\text{WRTD}}$：**数据EEPROM写保护位(1,4)
1 = 数据EEPROM不受写保护
0 = 数据EEPROM受写保护

bit 1　**$\overline{\text{WRTC}}$：**配置寄存器写保护位(1)
1 = 配置寄存器不受写保护
0 = 配置寄存器受写保护

bit 0　**$\overline{\text{WRTB}}$：**引导块写保护位(1,5)
1 = 引导块不受写保护
0 = 引导块受写保护

注 1： 这些位以粘住位的形式实现。保护功能通过ICSP或自写使能后，只能通过批量擦除操作复位。
2： 通过LVP编程接口工作时，不能对LVP位进行写操作（写为0）。该规则的目的是防止用户在通过LVP模式编程时退出LVP模式，或意外地从配置状态中删除LVP模式。
3： SAF不存在时未实现，仅适用于$\overline{\text{SAFEN}}$ = 0时。
4： 数据EEPROM不存在时未实现。
5： 仅适用于$\overline{\text{BBEN}}$ = 0时。

寄存器5-9： 配置字5L（30 0008h）

U-1	U-1	U-1	U-1	U-1	U-1	U-1	R/W-1
—	—	—	—	—	—	—	$\overline{CP}$
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7-1 **未实现：**读为1

bit 0 **$\overline{CP}$：**用户闪存程序存储器和数据EEPROM代码保护位
1 = 禁止用户闪存程序存储器和数据EEPROM代码保护
0 = 使能用户闪存程序存储器和数据EEPROM代码保护

寄存器5-10： 配置字5H（30 0009h）

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为1	
-n = 空白器件的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **未实现：**读为1

表5-2： 配置字汇总

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	默认值/未编程值
30 0000h	CONFIG1L	—	RSTOSC<2:0>			—	FEXTOSC<2:0>			1111 1111
30 0001h	CONFIG1H	—	—	FCMEN	—	CSWEN	—	PR1WAY	$\overline{CLKOUTEN}$	1111 1111
30 0002h	CONFIG2L	BOREN<1:0>		$\overline{LPBOREN}$	IVT1WAY	MVECEN	PWRTS<1:0>		MCLRE	1111 1111
30 0003h	CONFIG2H	$\overline{XINST}$	—	$\overline{DEBUG}$	STVREN	PPS1WAY	$\overline{ZCD}$	BORV<1:0>		1111 1111
30 0004h	CONFIG3L	—	WDTE<1:0>		WDTCPS<4:0>					1111 1111
30 0005h	CONFIG3H	—	—	WDTCCS<2:0>			WDTCWS<2:0>			1111 1111
30 0006h	CONFIG4L	$\overline{WRTAPP}$	—	—	$\overline{SAFEN}$	$\overline{BBEN}$	BBSIZE<2:0>			1111 1111
30 0007h	CONFIG4H	—	—	LVP	—	$\overline{WRTSAF}$	$\overline{WRTD}$	$\overline{WRTC}$	$\overline{WRTB}$	1111 1111
30 0008h	CONFIG5L	—	—	—	—	—	—	—	$\overline{CP}$	1111 1111
30 0009h	CONFIG5H	—	—	—	—	—	—	—	—	1111 1111

5.3 代码保护

通过代码保护，可以防止对器件的外部访问。程序存储器保护和数据存储器通过$\overline{CP}$配置位进行控制。对程序存储器的内部访问不会受代码保护设置影响。

整个程序存储空间和数据EEPROM都通过配置字中的CP位来防止外部读写操作。当CP = 0时，将禁止对存储器的外部读写操作，读取时将返回全0。无论保护位的设置如何，CPU都可以继续读取程序存储器和数据EEPROM。对程序存储器或数据EEPROM的自写操作取决于写保护设置。

5.4 用户ID

存储空间中有8个字（200000h-200000Fh）被指定为ID存储单元，供用户存储校验和或其他代码标识号。在正常执行期间，这些存储单元是可读写的。关于访问这些存储单元的更多信息，请参见**第13.2节“器件信息区、器件配置区、用户ID、器件ID和配置字访问”**。关于校验和计算的更多信息，请参见“*PIC18(L)F25/26K83 Memory Programming Specification*”（DS40001886）。

5.5 器件ID和版本ID

16位器件ID字位于3F FFFEh，16位版本ID位于3F FFFCh。这些存储单元是只读的，不能擦除或修改。

开发工具（如器件编程器和调试器）可用于读取器件ID、版本ID和配置字。有关访问这些存储单元的更多信息，请参见**第13.0节“非易失性存储器（NVM）控制”**。

5.6 寄存器定义：器件ID和版本ID

寄存器5-11： 器件ID：器件ID寄存器

R	R	R	R	R	R	R	R
DEV<15:8>							
bit 15							bit 8

R	R	R	R	R	R	R	R
DEV<7:0>							
bit 7							bit 0

图注：			
R = 可读位	1 = 置1	0 = 清零	x = 未知

bit 15-0 **DEV<15:0>：**器件ID位

器件	器件ID
PIC18F25K83	6EE0h
PIC18F26K83	6EC0h
PIC18LF25K83	6F20h
PIC18LF26K83	6F00h

寄存器5-12：　　版本ID：版本ID寄存器

R	R	R	R	R	R	R	R
1	0	1	0	MJRREV<5:2>			
bit 15							bit 8

R	R	R	R	R	R	R	R
MJRREV<1:0>		MNRREV<5:0>					
bit 7							bit 0

图注：			
R = 可读位	1 = 置1	0 = 清零	x = 未知

bit 15-12　**读为1010**
对于该系列的所有器件，这些位的值固定为1010。

bit 11-6　**MJRREV<5:0>：**主要版本ID位
这些位用于标识主要版本。主要版本由版本中的英文字母（A、B、C等）表示。
版本A = 0b00 0000

bit 5-0　**MNRREV<5:0>：**次要版本ID位
这些位用于标识次要版本。
版本A0 = 0b00 0000

5.7 器件信息区

器件信息区（DIA）是程序存储空间中的专用区域。DIA包含内部温度指示器模块的校准数据，并存储Microchip惟一标识符字和固定参考电压读数（测量单位为mV）。

完整的DIA表如表5-3：器件信息区所示，后面还附带各区域的说明及其功能。在PIC18(L)F25/26K83系列中，数据从3F0000h映射到3F003Fh。这些存储单元是只读的，用户无法擦除或修改。数据在制造期间编程到器件中。

表5-3： 器件信息区

地址范围	区域名称	标准器件信息
3F0000h-3F000Bh	MUI0	Microchip惟一标识符（6个字）
	MUI1	
	MUI2	
	MUI3	
	MUI4	
	MUI5	
3F000Ch-3F000Fh	MUI6	未分配（2个字）
	MUI7	
3F0010h-3F0023h	EUI0	可选外部惟一标识符（10个字）
	EUI1	
	EUI2	
	EUI3	
	EUI4	
	EUI5	
	EUI6	
	EUI7	
	EUI8	
	EUI9	
3F0024h-3F0025h	TSLR1	未分配（1个字）
3F0026h-3F0027h	TSLR2	90°C时的温度指示器ADC读数（低范围设置）
3F0028h-3F0029h	TSLR3	未分配（1个字）
3F002Ah-3F002Bh	TSHR1	未分配（1个字）
3F002Ch-3F002Dh	TSHR2	90°C时的温度指示器ADC读数（高范围设置）
3F002Eh-3F002Fh	TSHR3	未分配（1个字）
3F0030h-3F0031h	FVRA1X	1x设置的ADC FVR1输出电压（单位为mV）
3F0032h-3F0033h	FVRA2X	2x设置的ADC FVR1输出电压（单位为mV）
3F0034h-3F0035h	FVRA4X	4x设置的ADC FVR1输出电压（单位为mV）
3F0036h-3F0037h	FVRC1X	1x设置的比较器FVR2输出电压（单位为mV）
3F0038h-3F0039h	FVRC2X	2x设置的比较器FVR2输出电压（单位为mV）
3F003Ah-3F003Bh	FVRC4X	4x设置的比较器FVR2输出电压（单位为mV）
3F003Ch-3F003Fh		未分配（2个字）

5.7.1 MICROCHIP惟一标识符（MUI）

PIC18(L)F25/26K83器件在最终制造期间都单独编码了Microchip惟一标识符（Microchip Unique Identifier，MUI）。用户无法擦除MUI。凭借该功能，可以在需要时跟踪应用中Microchip器件的制造信息。应用制造商也可将该功能用于多个需要未经验证的惟一标识的功能，例如：

- 跟踪器件
- 惟一序列号

MUI由6个程序字组成。这些字段一起读取构成一个惟一标识符。MUI存储在DIA空间中3F0000h和3F000Fh之间的9个只读单元中。表5-3列出了标识符字的地址。

注：	如果应用需要经过验证的惟一标识，请联系Microchip销售办事处以创建序列号快速批量编程（Serialized Quick Turn Programming℠）选项。

5.7.2 外部惟一标识符（EUI）

EUI数据存储在程序存储区中的存储单元3F0010h至3F0023h中。此区域是用于存储应用特定信息的可选空间。数据在制造期间按照客户要求进行编码。

注：	收到客户请求后，会将数据存储在该地址范围内。客户可以联系当地销售代表或现场应用工程师，并向他们提供应该存储在此区域的惟一标识符信息。

5.7.3 对温度传感器的数据进行模数转换

温度传感器模块用于提供与温度相关的电压，该电压可通过模拟模块测得，请参见**第36.0节“温度指示器模块”**。

DIA表包含温度传感器低范围和高范围在固定参考点的内部ADC测量值。数值是在测试过程中测量的，对于每个器件都是惟一的。测量数据以与ADC转换结果相对应的十六进制数形式存储在DIA存储区中。校准数据可用于绘制近似传感器输出电压V_{TSENSE}—温度曲线，而无需在应用中进行校准测量。有关温度传感器操作的更多信息，请参见**第36.0节“温度指示器模块”**。

- **TSLR1-TSLR3**：地址3F0024h至3F0029h存储V_{DD} = 3V时温度传感器低范围设置的测量值。
- **TSHR1-TSHR3**：地址3F002Ah至3F002Fh存储V_{DD} = 3V时温度传感器高范围设置的测量值。
- 存储的测量值是在内部V_{REF} = 2.048V的条件下由器件ADC测得。

5.7.4 固定参考电压数据

DIA中的程序存储单元3F0030h至3F003Bh用于存储在不同缓冲区设置（1x、2x 或 4x）下测得的该器件的FVR电压（单位为mV）。有关FVR的更多信息，请参见**第35.0节“固定参考电压（FVR）”**。

- FVRA1X存储1x设置的ADC FVR1输出电压值（单位为mV）
- FVRA2X存储2x设置的ADC FVR1输出电压值（单位为mV）
- FVRA4X存储4x设置的ADC FVR1输出电压值（单位为mV）
- FVRC1X存储2x设置的比较器FVR2输出电压值（单位为mV）
- FVRC2X存储2x设置的比较器FVR2输出电压值（单位为mV）
- FVRC4X存储4x设置的比较器FVR2输出电压值（单位为mV）

5.8 器件配置信息

器件配置信息（DCI）是程序存储空间中的专用区域，从3FFF00h映射到3FFF09h。这些存储单元中存储的数据是只读的，无法擦除。

有关完整的DCI表地址和说明，请参见表5-4：PIC18(L)F25/26K83的器件配置信息。DCI用于保存对编程和自举程序应用有帮助的器件相关信息。

擦除大小是PFM中的最小可擦除单位，以行数表示。器件闪存总容量为(行大小 * 行数)

表5-4： PIC18(L)F25/26K83的器件配置信息

地址	名称	说明	值		单位
			PIC18(L)F25K83	PIC18(L)F26K83	
3F FF00h-3F FF01h	ERSIZ	擦除行大小	64	64	字
3F FF02h-3F FF03h	WLSIZ	每行的写锁存器数	128	128	字节
3F FF04h-3F FF05h	URSIZ	用户行数	256	512	行
3F FF06h-3F FF07h	EESIZ	数据EEPROM存储器大小	1024	1024	字节
3F FF08h-3F FF09h	PCNT	引脚数	28	28	引脚

6.0 复位

该器件有多种复位方式：

- 上电复位（POR）
- 欠压复位（BOR）
- 低功耗欠压复位（Low-Power Brown-Out Reset，LPBOR）
- $\overline{\text{MCLR}}$复位
- WDT复位
- RESET指令
- 堆栈上溢
- 堆栈下溢
- 编程模式退出
- 存储器执行违例复位（$\overline{\text{MEMV}}$）

要使V_{DD}稳定，可以使能可选的上电延时定时器，以在BOR或POR事件后延长复位时间。

图6-1给出了片上复位电路的简化框图。

图6-1： 片上复位电路的简化框图

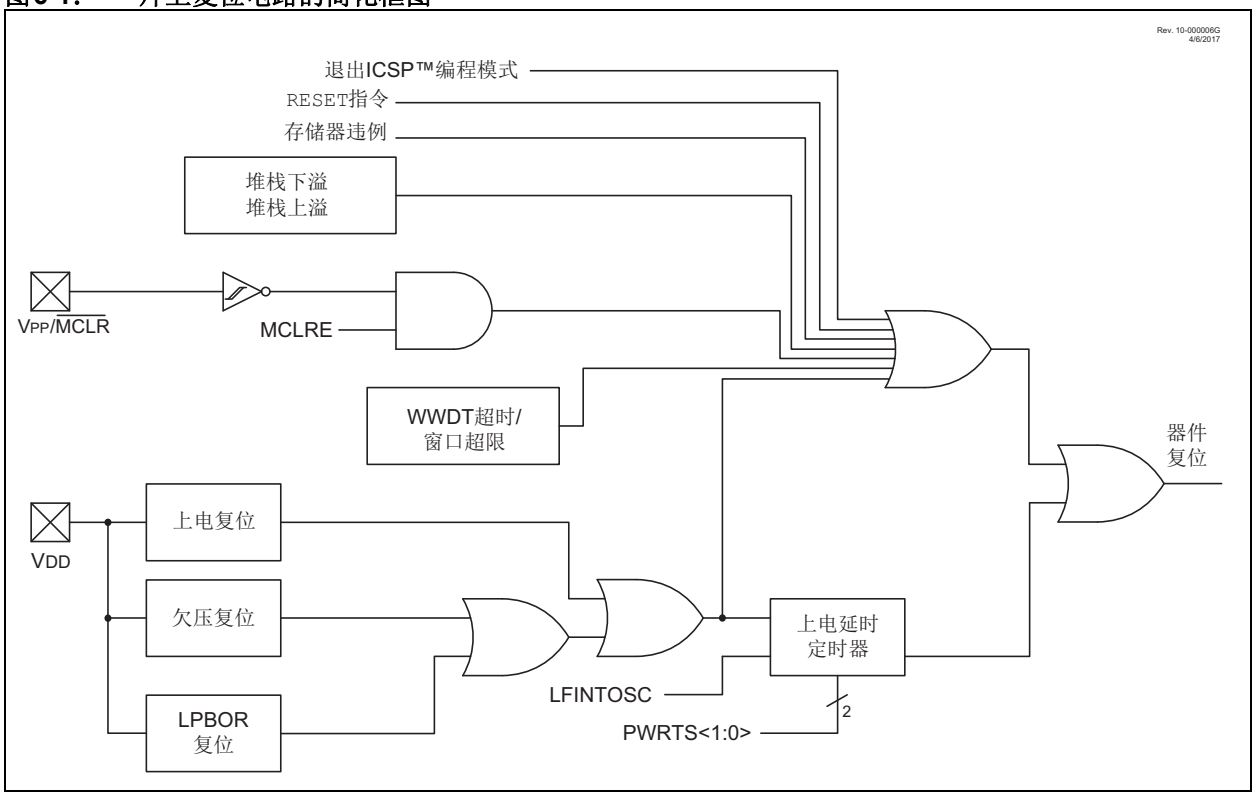
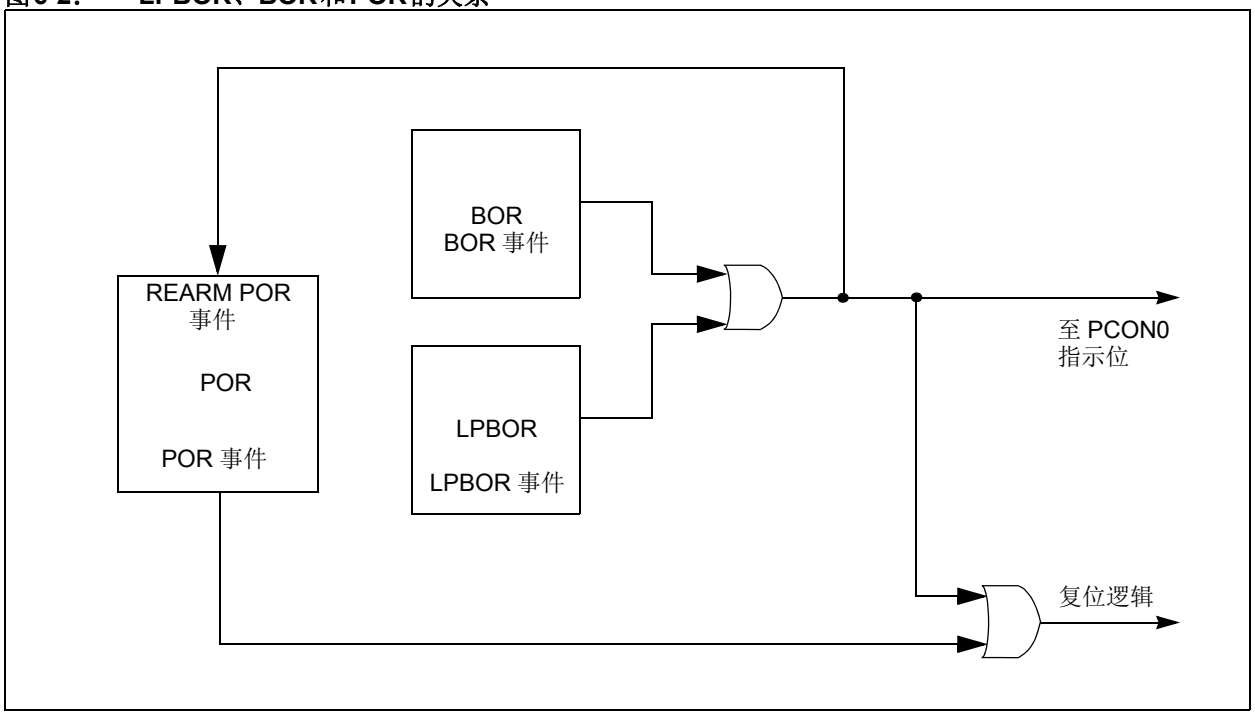


图6-2： LPBOR、BOR和POR的关系



6.1 上电复位（POR）

POR电路会将器件保持在复位状态，直到VDD达到可接受的最低工作电压为止。在VDD缓慢上升、高速运行或要求一定模拟性能时，所需的电压可能高于最低VDD。可以使用PWRT、BOR或MCLR功能来延长启动周期，直到满足所有器件工作条件为止。

6.2 欠压复位（BOR）

当VDD达到可选的最低电平时，BOR电路将器件保持在复位状态。在POR和BOR之间，可在整个电压范围内对器件的执行进行保护。

欠压复位模块具有4种工作模式，它们由配置字中的BOREN<1:0>位控制。这4种工作模式是：

- BOR总是使能
- BOR在休眠模式下禁止
- BOR通过软件进行控制
- BOR总是禁止

更多信息，请参见表6-1。

对配置字中的BORV<1:0>位进行配置来选择欠压复位电压。

VDD噪声抑制滤波器可以防止BOR在发生小事件时产生触发。如果VDD降至低于VBOR的时间大于参数TBORDC，器件将会发生复位。更多信息，请参见表45-11。

6.2.1 BOR总是使能

当配置字的BOREN位编程为11时，BOR将总是使能。器件启动会被延迟，直到BOR就绪，且VDD高于BOR阈值为止。

BOR保护在休眠期间有效。BOR不会延迟从休眠中唤醒。

6.2.2 BOR在休眠模式下禁止

当配置字的BOREN位编程为10时，除非处于休眠模式，否则BOR将使能。器件启动会被延迟，直到BOR就绪，且VDD高于BOR阈值为止。

BOR保护在休眠期间无效。器件唤醒会被延迟，直到BOR就绪为止。

6.2.3 BOR通过软件进行控制

当配置字的BOREN位编程为01时，BOR将通过BORCON寄存器的SBOREN位进行控制。器件启动不会受BOR就绪条件或VDD电压影响而延迟。

BOR保护会在BOR电路就绪时立即开始。BOR电路的状态在BORCON寄存器的BORRDY位中反映。

BOR保护在休眠期间不变。

6.2.4 BOR和批量擦除

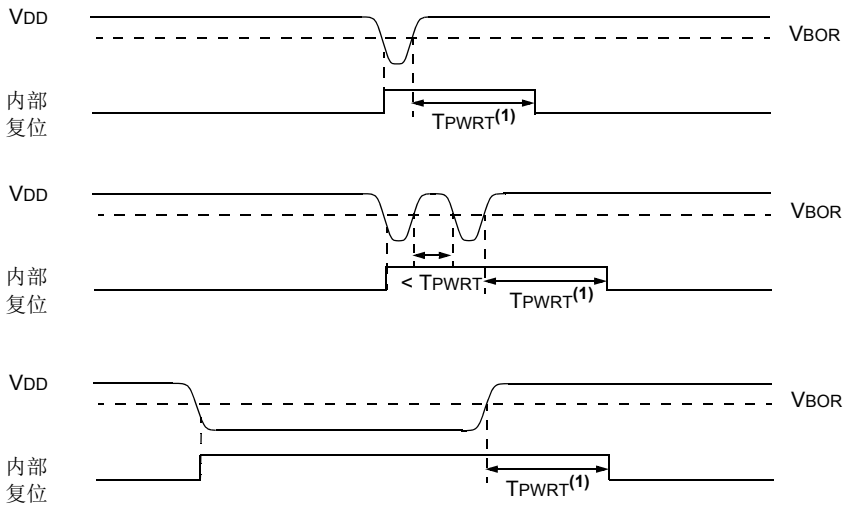
BOR在PFM批量擦除操作期间强制使能，以确保在成功擦除周期内保持安全擦除电压。

在批量擦除期间，对于F和LF器件，即使BOR配置为某个其他值，也会在电压达到2.45V时使能。如果VDD下降，则擦除周期将中止，但器件不会复位。

表6-1： BOR工作模式

BOREN<1:0>	SBOREN	器件模式	BOR模式	在以下情况下的指令执行：	
				POR释放	从休眠模式唤醒
11	X	X	有效	等待BOR释放（BORRDY = 1）	立即开始
10	X	唤醒	有效	等待BOR释放（BORRDY = 1）	N/A
		休眠	冬眠	N/A	等待BOR释放（BORRDY = 1）
01	1	X	有效	等待BOR释放（BORRDY = 1）	立即开始
	0	X	冬眠		
00	X	X	禁止	立即开始	

图6-3： 欠压情形



注 1： TPWRT延时取决于PWRTS<1:0>配置位。

6.3 寄存器定义：BOR控制

寄存器6-1： BORCON：欠压复位控制寄存器

R/W-1/u	U-0	U-0	U-0	U-0	U-0	U-0	R-q/u
SBOREN	—	—	—	—	—	—	BORRDY
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **SBOREN**：软件欠压复位使能位

如果BOREN ≠ 01：
SBOREN可读/写，但对BOR没有任何作用。
如果BOREN = 01：
1 = 使能BOR
0 = 禁止BOR

bit 6-1 **未实现**：读为0

bit 0 **BORRDY**：欠压复位电路就绪状态位

1 = 欠压复位电路有效且就绪
0 = 欠压复位电路被禁止或仍在预热

6.4 低功耗欠压复位（LPBOR）

低功耗欠压复位（LPBOR）提供了额外的BOR电路来实现低功耗操作。关于BOR如何与其他模块进行交互的信息，请参见图6-2。

LPBOR用于监视外部VDD引脚。当检测到电压太低时，器件将保持在复位状态。

6.4.1 使能LPBOR

LPBOR由配置字2L的$\overline{\text{LPBOREN}}$位控制。在器件被擦除后，LPBOR模块默认设为禁止。

6.4.1.1 LPBOR模块输出

LPBOR模块的输出是一个用于指示是否要将复位置为有效的信号。该信号与BOR模块的复位信号进行逻辑或，用以提供通用$\overline{\text{BOR}}$信号，并送至PCON0寄存器和电源控制模块。

6.5 $\overline{\text{MCLR}}$

$\overline{\text{MCLR}}$是可将器件复位的可选外部输入。$\overline{\text{MCLR}}$功能由配置字的MCLRE位和LVP位控制（表6-2）。如果发生了$\overline{\text{MCLR}}$复位，PCON0寄存器中的$\overline{\text{RMCLR}}$位将设置为0。

表6-2： $\overline{\text{MCLR}}$配置

MCLRE	LVP	$\overline{\text{MCLR}}$
x	1	使能
1	0	使能
0	0	禁止

6.5.1 $\overline{\text{MCLR}}$使能

当使能$\overline{\text{MCLR}}$并且引脚保持低电平时，器件会保持在复位状态。$\overline{\text{MCLR}}$引脚通过内部弱上拉连接到VDD。

器件在$\overline{\text{MCLR}}$复位路径中有一个噪声滤波器。该滤波器检测并滤除小脉冲。

注： 内部复位事件（RESET指令、BOR、WWDT和POR堆栈）不会将$\overline{\text{MCLR}}$引脚驱动为低电平。

6.5.2 $\overline{\text{MCLR}}$禁止

当禁止$\overline{\text{MCLR}}$时，$\overline{\text{MCLR}}$引脚仅用作输入，内部弱上拉等引脚功能由软件控制。更多信息，请参见**第16.2节“I/O优先级”**。

6.6 窗口看门狗定时器（WWDT）复位

如果固件未在设定的超时周期或窗口内发出CLRWDT指令，窗口看门狗定时器会产生复位。STATUS寄存器中的$\overline{\text{TO}}$和$\overline{\text{PD}}$位以及PCON0寄存器中的$\overline{\text{RWDT}}$位将发生变化，以指示WWDT复位。PCON0寄存器中的$\overline{\text{WDTWV}}$位指示是否因超时或窗口超限而发生WDT复位。更多信息，请参见**第11.0节“窗口看门狗定时器（WWDT）”**。

6.7 RESET指令

RESET指令会引起器件复位。PCON0寄存器中的$\overline{\text{RI}}$位将设置为0。关于发生RESET指令之后的默认条件，请参见表6-3。

6.8 堆栈上溢/下溢复位

器件可以在堆栈上溢或下溢时复位。PCON0寄存器的STKOVF或STKUNF位用于指示复位条件。这两种复位通过将配置字中的STVREN位置1来使能。更多信息，请参见**第4.2.5节“返回地址堆栈”**。

6.9 编程模式退出

在退出编程模式时，器件的行为与刚刚发生POR时的情况相同。

6.10 上电延时定时器（PWRT）

上电延时定时器在POR或欠压复位时提供一段选定的超时持续时间。

只要PWRT处于活动状态，器件就保持在复位状态。PWRT延时使VDD有额外的时间上升到可接受的电压。通过适当设置PWRTS<1:0>配置位来选择上电延时定时器。

上电延时定时器会在POR和BOR/LPBOR（如果使能）释放之后启动，如图6-1所示。

6.11 启动序列

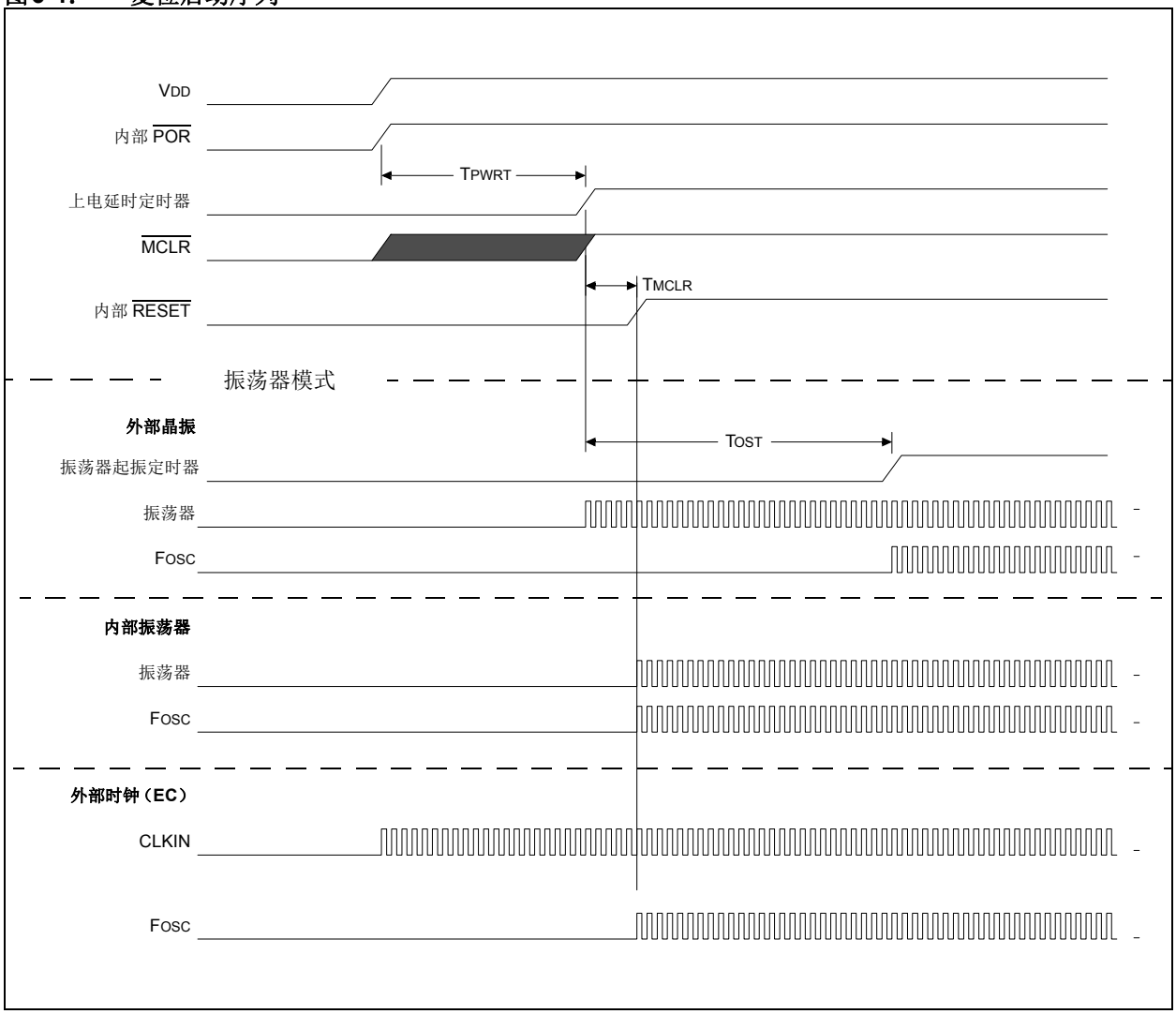
在POR或BOR释放时，只有先发生以下事件，器件才会开始执行：

1. 上电延时定时器运行完毕（如果使能）。
2. 振荡器起振定时器运行完毕（如果对于选定振荡器源需要）。
3. $\overline{\text{MCLR}}$必须被释放（如果使能）。

总超时将根据振荡器配置和上电延时定时器配置而变化。更多信息，请参见**第7.0节“振荡器模块（带故障保护时钟监视器）”**。

上电延时定时器和振荡器起振定时器的运行与$\overline{\text{MCLR}}$复位无关。如果$\overline{\text{MCLR}}$保持足够长时间的低电平，上电延时定时器和振荡器起振定时器将超时。将$\overline{\text{MCLR}}$电平拉高后，器件将在10个FOSC周期后开始执行（见图6-4）。这对于测试或同步多个并行工作的器件来说是非常有用的。

图6-4： 复位启动序列



6.11.1 存储器执行违例

如果CPU在有效执行区域之外执行，则会发生存储器执行违例复位。

无效的执行区域包括：

1. 在所实现程序存储器之外的地址（见表5-1）。
2. 程序存储器内的存储区闪存（SAF）（如果已使能）。

当产生存储器执行违例时，PCON1（寄存器6-3）中的标志$\overline{\text{MEMV}}$将清零以指示复位的原因。在发生存储器执行违例复位后，需要在用户代码中将该标志置1以检测后续的违例复位。

6.12 确定复位原因

在发生任何复位时，STATUS和PCON0寄存器中会有多个位发生更新，以指示复位的原因。表6-3列出了这些寄存器的复位条件。

表6-3： 特殊寄存器的复位条件

条件	程序计数器	STATUS寄存器(1,2)	PCON0寄存器	PCON1寄存器
上电复位	0	-110 0000	0011 110x	---- --1-
欠压复位	0	-110 0000	0011 11u0	---- --1-
正常工作期间的$\overline{\text{MCLR}}$复位	0	-uuu uuuu	uuuu 0uuu	---- --u-
休眠期间的$\overline{\text{MCLR}}$复位	0	-10u uuuu	uuuu 0uuu	---- --u-
WDT超时复位	0	-0uu uuuu	uuu0 uuuu	---- --u-
WWDT窗口超限复位	0	-uuu uuuu	uu0u uuuu	---- --u-
执行了RESET指令	0	-uuu uuuu	uuuu u0uu	---- --u-
堆栈上溢复位（STVREN = 1）	0	-uuu uuuu	1uuu uuuu	---- --u-
堆栈下溢复位（STVREN = 1）	0	-uuu uuuu	u1uu uuuu	---- --u-
存储器违例复位	0	-uuu uuuu	uuuu uuuu	---- --0-

图注： u = 不变，x = 未知，– = 未实现位，读为0。

注 1： 如果状态位未实现，该位将读为0。

2： 状态位Z、C和DC由POR/BOR复位，但并非由复位模块定义（寄存器4-2）。

6.13 电源控制（PCON0/PCON1）寄存器

电源控制（PCON0/PCON1）寄存器包含区分以下各种复位的标志位：

- 欠压复位（$\overline{\text{BOR}}$）
- 上电复位（$\overline{\text{POR}}$）
- RESET指令复位（$\overline{\text{RI}}$）
- $\overline{\text{MCLR}}$复位（$\overline{\text{RMCLR}}$）
- 看门狗定时器复位（$\overline{\text{RWDT}}$）
- 看门狗窗口超限（$\overline{\text{WDTWV}}$）
- 堆栈下溢复位（STKUNF）
- 堆栈上溢复位（STKOVF）
- 存储器违例复位（$\overline{\text{MEMV}}$）

PCON0/1寄存器位如寄存器6-2和寄存器6-3所示。硬件将在复位过程中改变相应的寄存器位；如果复位不是由相应条件引起的，对应位保持不变（表6-3）。

在重启后，软件应将相应位复位为无效状态（硬件不会复位相应位）。软件还可将PCON0位设置为有效状态，这样可测试用户代码，但不会产生任何复位操作。

6.14 寄存器定义：电源控制

寄存器6-2： PCON0：电源控制寄存器0

R/W/HS-0/q	R/W/HS-0/q	R/W/HC-1/q	R/W/HC-1/q	R/W/HC-1/q	R/W/HC-1/q	R/W/HC-0/u	R/W/HC-q/u
STKOVF	STKUNF	$\overline{\text{WDTWV}}$	$\overline{\text{RWDT}}$	$\overline{\text{MCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

图注：		
HC = 硬件清零位		HS = 硬件置1位
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-m/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **STKOVF**：堆栈上溢标志位
1 = 发生了堆栈上溢（CALL数量超出堆栈范围）
0 = 未发生堆栈上溢或该位由固件设置为0

bit 6 **STKUNF**：堆栈下溢标志位
1 = 发生了堆栈下溢（RETURN的数量多于CALL）
0 = 未发生堆栈下溢或该位由固件设置为0

bit 5 **$\overline{\text{WDTWV}}$**：看门狗窗口超限位
1 = 未发生WDT窗口超限或由固件设置为1
0 = 在WDT复位窗口关闭时发出了CLRWDT指令（在发生WDT窗口超限复位时由硬件设置为0）

bit 4 **$\overline{\text{RWDT}}$**：WDT复位标志位
1 = 未发生WDT上溢/超时复位或由固件设置为1
0 = 发生了WDT上溢/超时复位（在发生WDT复位时由硬件设置为0）

bit 3 **$\overline{\text{RMCLR}}$**：$\overline{\text{MCLR}}$复位标志位
1 = 未发生$\overline{\text{MCLR}}$复位或由固件设置为1
0 = 发生了$\overline{\text{MCLR}}$复位（发生$\overline{\text{MCLR}}$复位时由硬件设置为0)

bit 2 **$\overline{\text{RI}}$**：RESET指令标志位
1 = 未执行RESET指令或由固件设置为1
0 = 执行了RESET指令（执行RESET指令时由硬件设置为0）

bit 1 **$\overline{\text{POR}}$**：上电复位状态位
1 = 未发生上电复位或由固件设置为1
0 = 发生了上电复位（发生上电复位时由硬件设置为0）

bit 0 **$\overline{\text{BOR}}$**：欠压复位状态位
1 = 未发生欠压复位或由固件设置为1
0 = 发生了欠压复位（发生欠压复位时由硬件设置为0）

寄存器6-3： PCON1：电源控制寄存器1

U-0	U-0	U-0	U-0	U-0	U-0	R/W/HC-1/u	U-0
—	—	—	—	—	—	$\overline{\text{MEMV}}$	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-m/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-2 **未实现：**读为0

bit 1 **$\overline{\text{MEMV}}$：**存储器违例标志位
1 = 未发生存储器违例复位或由固件设置为1
0 = 发生了存储器违例复位（发生存储器违例时由硬件设置为0）

bit 0 **未实现：**读为0

表6-4： 与复位相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
BORCON	SBOREN	—	—	—	—	—	—	BORRDY	75
PCON0	STKOVF	STKUNF	$\overline{\text{WDTWV}}$	$\overline{\text{RWDT}}$	$\overline{\text{MCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	80
PCON1	—	—	—	—	—	—	$\overline{\text{MEMV}}$	—	81

图注： — = 未实现位，读为0。复位不使用阴影单元。

7.0 振荡器模块（带故障保护时钟监视器）

7.1 概述

振荡器模块具有多种时钟源和选择特性，广泛适用于各种应用，同时可最大程度地发挥性能并降低功耗。图7-1给出了振荡器模块的框图。

时钟源可由外部振荡器、石英晶体谐振器和陶瓷谐振器提供。此外，系统时钟源可由两个内部振荡器之一和PLL电路提供，并通过软件来选择速度。其他时钟特性包括：

- 可通过软件选择外部或内部时钟源作为系统时钟源。
- 故障保护时钟监视器（Fail-Safe Clock Monitor，FSCM），用来检测外部时钟源（LP、XT、HS、ECH、ECM和ECL）故障并自动切换到内部振荡器。
- 振荡器起振定时器（OST），可确保晶振源的稳定性。

配置字1（寄存器5-1）的RSTOSC位决定复位后器件运行（包括初次上电）时使用的振荡器类型。

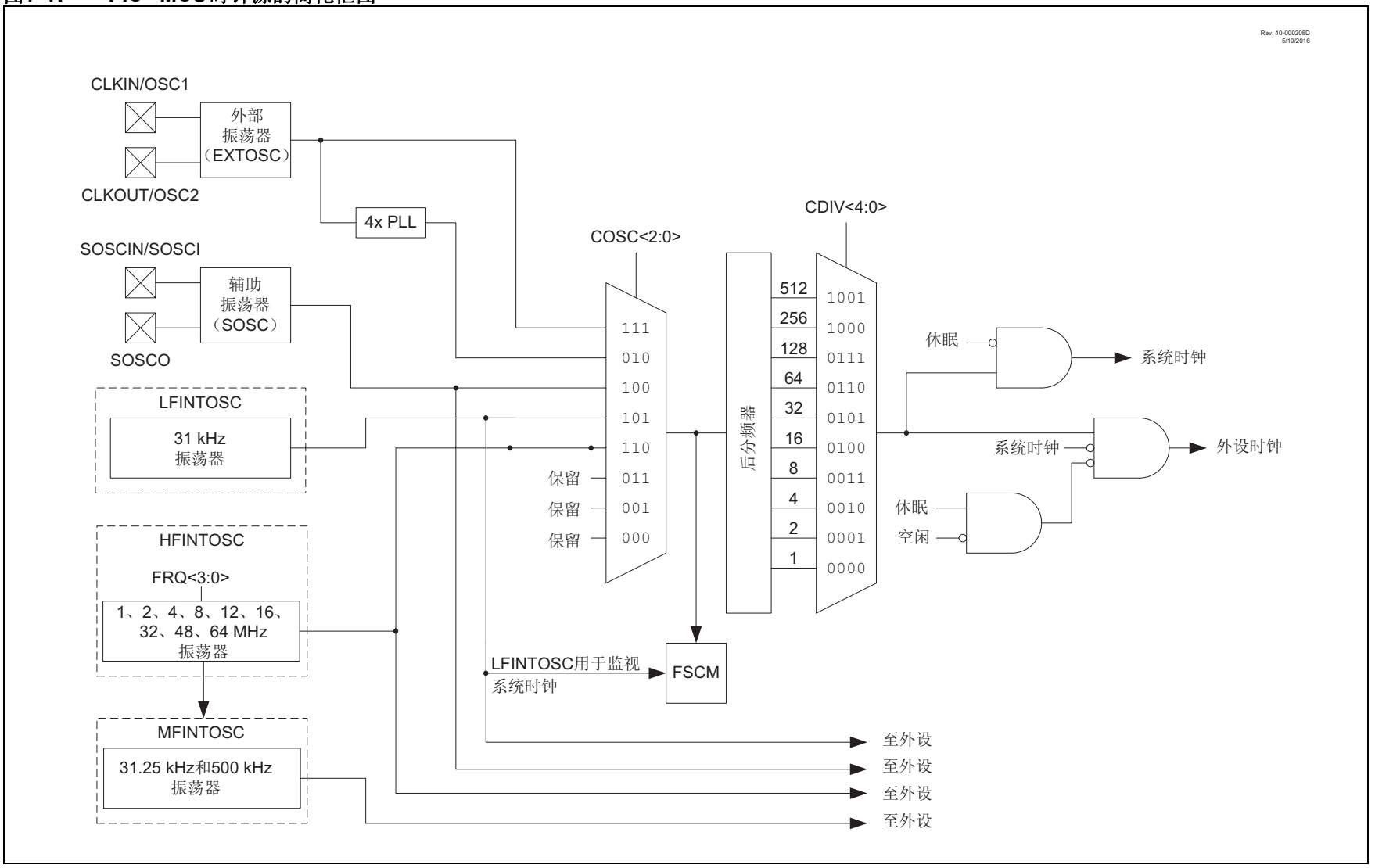
如果选择外部时钟源，必须结合使用配置字1的FEXTOSC位和RSTOSC位来选择外部时钟模式。

可通过设置FEXTOSC<2:0>配置位将外部振荡器模块配置为以下时钟模式之一：

1. ECL——外部时钟低功耗模式（低于100 kHz）
2. ECM——外部时钟中等功耗模式（100 kHz至8 MHz）
3. ECH——外部时钟高功耗模式（高于8 MHz）
4. LP——32 kHz低功耗晶振模式
5. Xt——中等增益晶振或陶瓷谐振器振荡器模式（100 kHz至8 MHz）
6. HS——高增益晶振或陶瓷谐振器模式（高于4 MHz）

ECH、ECM和ECL时钟模式依靠外部逻辑电平信号作为器件时钟源。LP、XT和HS时钟模式要求器件在外部连接一个晶振或谐振器。每种模式都针对不同频率范围而优化。内部振荡器模块可以产生低频和高频时钟源，分别用LFINTOSC和HFINTOSC表示。（见内部振荡器模块，图7-1）。这两个时钟源可产生多种器件时钟频率供选择。

图7-1： PIC® MCU时钟源的简化框图



7.2 时钟源类型

时钟源可分为外部时钟源和内部时钟源。

外部时钟源依靠外部电路工作。例如：振荡器模块（ECH、ECM和ECL模式）、石英晶振或陶瓷谐振器（LP、XT和HS模式）。

内部时钟源内置在振荡器模块中。内部振荡器模块具有两个内部振荡器，用于产生内部系统时钟源。高频内部振荡器（HFINTOSC）可产生1、2、4、8、12、16、32、48和64 MHz的时钟。可以通过OSCFRQ寄存器（寄存器7-5）控制频率。低频内部振荡器（LFINTOSC）可产生31 kHz的固定频率。

所提供的4x PLL可与外部时钟配合使用。当与EXTOSC配合使用时，4x PLL具有输入频率限制。更多详细信息，请参见**第7.2.1.4节“4x PLL”**。

可以通过OSCCON1寄存器中的NOSC位选择外部或内部时钟源作为系统时钟。更多信息，请参见**第7.3节“时钟切换”**。对于不使用OSC2引脚的任何模式，可以在OSC2/CLKOUT引脚上提供系统时钟。时钟输出功能由CONFIG1H寄存器（寄存器5-2）中的CLKOUTEN位控制。如果使能，则时钟输出信号的频率始终为FOSC/4。

7.2.1 外部时钟源

通过执行以下操作之一，可以使用外部时钟源作为器件系统时钟：

- 编程配置字中的RSTOSC<2:0>和FEXTOSC<2:0>位，选择在器件复位时用作默认系统时钟的外部时钟源。
- 写入OSCCON1寄存器中的NOSC<2:0>和NDIV<3:0>位以切换系统时钟源。

更多信息，请参见**第7.3节“时钟切换”**。

7.2.1.1 EC模式

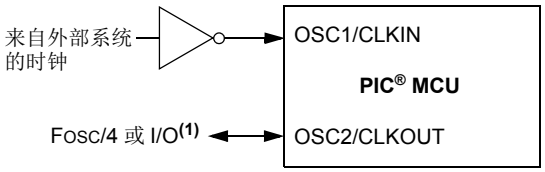
外部时钟（EC）模式允许外部产生的逻辑电平信号作为系统时钟源。工作在此模式下时，外部时钟源应连接到OSC1输入引脚。OSC2/CLKOUT可用作通用I/O或CLKOUT。图7-2给出了EC模式的引脚连接。

EC模式具有三种功耗模式，可通过配置字进行选择：

- ECH——高功耗，高于8 MHz
- ECM——中等功耗，100 kHz-8 MHz
- ECL——低功耗，低于100 kHz

当选取EC模式时，振荡器起振定时器（OST）被禁止。因此，上电复位（POR）后或者从休眠中唤醒后的操作不存在延时。因为PIC® MCU的设计是完全静态的，停止外部时钟输入将使器件暂停工作并保持所有数据完整。当再次启动外部时钟时，器件恢复工作，就好像没有停止过一样。

图7-2： 外部时钟（EC）模式的工作原理



注 1： 输出取决于配置字（CONFIG1H）的CLKOUTEN位。

7.2.1.2 LP、XT和HS模式

LP、XT和HS模式支持使用连接到OSC1和OSC2的石英晶振或陶瓷谐振器（图7-3）。这三种模式选择内部反相放大器的低、中或高增益设置，以支持各种谐振器类型及速度。

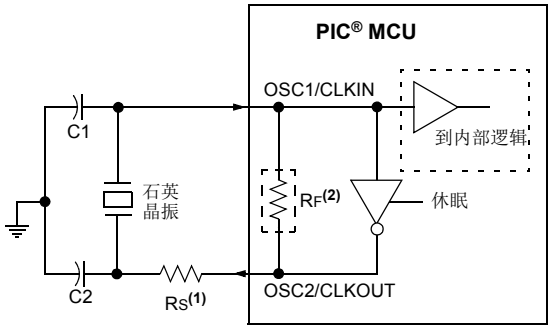
LP振荡器模式选择内部反相放大器的最低增益设置。LP模式的电流消耗在三种模式中最小。该模式设计为用来驱动仅工作在32.768 kHz的音叉（Tuning Fork）型晶振（时钟晶振）。

XT振荡器模式选择内部反相放大器的中等增益设置。XT模式的电流消耗在三种模式中居中。该模式最适合驱动具备中等驱动电流要求（100 kHz-8 MHz）的谐振器。

HS振荡器模式选择内部反相放大器的最高增益设置。HS模式的电流消耗在三种模式中最大。该模式最适合驱动需要高驱动电流（高于8 MHz）的谐振器。

图7-3和图7-4分别显示了石英晶振和陶瓷谐振器的典型电路。

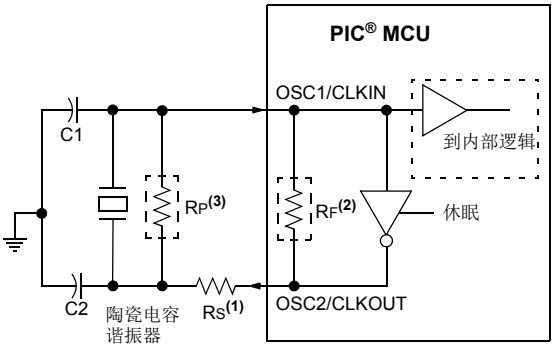
图7-3： 石英晶振的工作原理（LP、XT或HS模式）



注 1： 低驱动电平的石英晶振可能需要一个串联电阻（RS）。

2： RF的值随选定的振荡器模式而变化（通常介于2 MΩ和10 MΩ之间）。

图7-4： 陶瓷谐振器的工作原理（XT或HS模式）



注 1： 低驱动电平的陶瓷谐振器可能需要一个串联电阻（RS）。

2： RF的值随选定的振荡器模式而变化（通常介于2 MΩ和10 MΩ之间）。

3： 要让陶瓷谐振器正常工作可能还需要一个并联反馈电阻（RP）。

7.2.1.3 振荡器起振定时器（OST）

如果振荡器模块被配置为LP、XT或HS模式，则振荡器起振定时器（OST）对来自OSC1的振荡计数1024次。这发生在上电复位（POR）或从休眠中唤醒后。OST确保使用石英晶体谐振器或陶瓷谐振器的振荡器电路已经起振并为振荡器模块提供稳定的系统时钟。

7.2.1.4 4x PLL

振荡器模块包含了一个4x PLL，它可以与外部时钟源配合使用，用于提供系统时钟源。PLL的输入频率必须处于规范值范围内。请参见表45-9中的PLL时钟时序规范。

使用时，可以通过以下两种方法之一使能PLL：

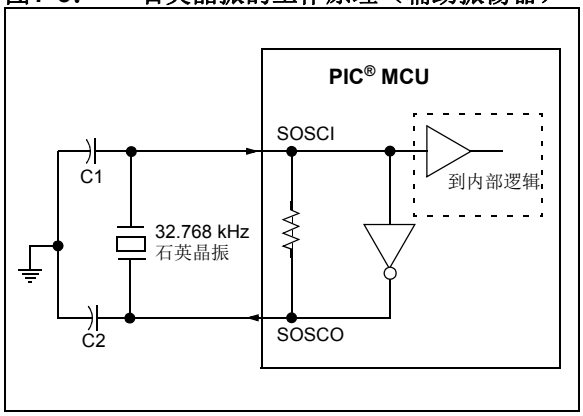
1. 将配置字1中的RSTOSC位编程为010以使能采用4x PLL的EXTOSC。
2. 向OSCCON1寄存器中的NOSC位写入010以使能采用4x PLL的EXTOSC。

7.2.1.5 辅助振荡器

辅助振荡器是一个单独的振荡器模块，可用作备用系统时钟源。辅助振荡器针对32.768 kHz进行优化，可与连接到SOSCI和SOSCO器件引脚的外部晶振或连接到SOSCIN引脚的外部时钟源配合使用。辅助振荡器可在运行期间通过时钟切换来选择。更多信息，请参见**第7.3节“时钟切换”**。

辅助振荡器有两种功耗模式。这两种模式使用SOSCPWR（OSCCON3<6>）来选择。将该位清零将选择低晶振增益模式，该模式可提供最低单片机功耗。将该位置1可实现高增益模式，从而支持更快的晶振起振速度或更高ESR的晶振。

图7-5： 石英晶振的工作原理（辅助振荡器）



注 1： 石英晶振的特性随类型、封装和制造商的不同而不同。要了解规格说明和推荐应用，应查阅制造商提供的数据手册。

2： 应始终验证振荡器在应用要求的VDD和温度范围内的性能。

3： 如需振荡器设计帮助，请参见以下Microchip应用笔记：

- AN826 “*Crystal Oscillator Basics and Crystal Selection for PIC® and PIC® Devices*”（DS00826）
- AN849，“*Basic PIC® Oscillator Design*”（DS00849）
- AN943，“*Practical PIC® Oscillator Analysis and Design*”（DS00943）
- AN949，“*Making Your Oscillator Work*”（DS00949）
- TB097，“*Interfacing a Micro Crystal MS1V-T1K 32.768 kHz Tuning Fork Crystal to a PIC16F690/SS*”（DS91097）
- AN1288，“*Design Practices for Low-Power External Oscillators*”（DS01288）

7.2.2 内部时钟源

通过执行以下操作之一，可以将器件配置为使用内部振荡器模块作为系统时钟：

- 编程配置字中的RSTOSC<2:0>位，选择在器件复位时用作默认系统时钟的INTOSC时钟源。
- 在运行时写入OSCCON1寄存器中的NOSC<2:0>位，将系统时钟源切换为内部振荡器。更多信息，请参见**第7.3节“时钟切换”**。

在INTOSC模式下，OSC1/CLKIN可用作通用I/O，前提是FEXTOSC配置为“未使能振荡器”。OSC2/CLKOUT可用作通用I/O或CLKOUT。

OSC2/CLKOUT引脚的功能由配置字中的$\overline{\text{CLKOUTEN}}$位决定。

内部振荡器模块有两个独立的振荡器，可产生两个内部系统时钟源。

1. **HFINTOSC**（高频内部振荡器）出厂时已校准，工作频率为1至64 MHz。HFINTOSC的频率可通过OSCFRQ频率选择寄存器进行选择，并通过OSCTUNE寄存器实现微调。
2. **LFINTOSC**（低频内部振荡器）出厂时已校准，工作频率为31 kHz。

7.2.2.1 HFINTOSC

高频内部振荡器（HFINTOSC）是一个高精度数字控制内部时钟源，可产生最高64 MHz的稳定时钟。HFINTOSC可通过以下方法之一来使能：

- 在器件上电或复位后，将配置字1中的RSTOSC<2:0>位编程为110（FOSC = 1 MHz）或000（FOSC = 64 MHz）以设置振荡器。
- 在运行时，写入OSCCON1寄存器的NOSC<2:0>位。更多信息，请参见**第7.3节“时钟切换”**。

可通过设置OSCFRQ寄存器的FRQ<3:0>位来选择HFINTOSC频率。

凭借OSCCON1寄存器的NDIV<3:0>位，可对HFINTOSC输出进行分频，分频比的范围为1:1至1:512。

7.2.2.2 MFINTOSC

该模块提供两个（500 kHz和31.25 kHz）恒定时钟输出。这两个时钟是对HFINTOSC进行数字分频后得到的。动态分频器逻辑用于为HFINTOSC的所有设置提供恒定的MFINTOSC时钟速率。

MFINTOSC无法驱动系统，但可以用作诸如定时器和WWDT等特定模块的时钟。

7.2.2.3 内部振荡器频率调整

内部振荡器在出厂时已校准。该内部振荡器可以通过用软件写入OSCTUNE寄存器（寄存器7-3）进行调整。

OSCTUNE寄存器的默认值为00h。该值为6位的二进制补码。值为1Fh时，将调整为最高频率。值为20h时，将调整为最低频率。

当OSCTUNE寄存器被修改时，振荡器频率将开始改变为新频率。改变期间，代码将继续执行。不会明确指示是否已发生频率改变。

OSCTUNE**不会影响**LFINTOSC频率。依赖于LFINTOSC时钟源频率的功能，如上电延时定时器（PWRT）、WWDT、故障保护时钟监视器（FSCM）以及外设等，其工作*不*受频率改变的影响。

7.2.2.4 LFINTOSC

低频内部振荡器（LFINTOSC）在出厂时已校准为31 kHz内部时钟源。

LFINTOSC的频率是上电延时定时器（PWRT）、窗口看门狗定时器（WWDT）和故障保护时钟监视器（FSCM）的时钟频率。

可以通过以下方法之一使能LFINTOSC节点：

- 编程配置字1的RSTOSC<2:0>位来使能LFINTOSC。
- 在运行时，写入OSCCON1寄存器的NOSC<2:0>位。更多信息，请参见**第7.3节，时钟切换**。

7.2.2.5 ADCRC

ADCRC是ADC^{2}模块的专用振荡器。可以使用OSCEN寄存器的ADOEN位手动使能ADCRC振荡器。ADCRC以600 kHz的固定频率运行。如果选择将ADCRC作为ADC^{2}模块的时钟源，ADCRC会自动使能。

7.2.2.6 振荡器状态和手动使能

每个振荡器（包括ADCRC振荡器）的“就绪”状态显示在OSCSTAT（寄存器7-4）中。可以通过OSCEN（寄存器7-7）明确使能振荡器（而非PLL）。

7.2.2.7 HFOR和MFOR位

HFOR和MFOR位用于指示HFINTOSC和MFINTOSC是否已就绪。这两个时钟始终有效，可随时使用，但只有在就绪后才准确。

当OSCFRQ寄存器中装载新值时，HFOR和MFOR位将清零，随后在振荡器就绪时会再次置1。在OSCFRQ待更改期间，MFINTOSC时钟将停留在高电平或低电平状态，直到HFINTOSC恢复工作。

7.3 时钟切换

可通过软件使用OSCCON1寄存器的新振荡器源（NOSC）位在外部和内部时钟源之间切换系统时钟源。可以选择以下时钟源：

- 外部振荡器
- 内部振荡器模块（INTOSC）

注：	配置字1中的时钟切换使能位可用于使能或禁止时钟切换功能。清零时，用户软件无法更改NOSC和NDIV位。置1时，允许写入NOSC和NDIV以切换时钟频率。

7.3.1 新振荡器源（NOSC）和新分频比选择请求（NDIV）位

OSCCON1寄存器的新振荡器源（NOSC）位和新分频比选择请求（NDIV）位选择用于CPU和外设的系统时钟源和频率。

当将NOSC和NDIV的新值写入OSCCON1时，当前振荡器选择将在等待新时钟源指示其已稳定并就绪时继续运行。在某些情况下，新请求的时钟源可能已在使用并立即就绪。对于仅发生分频比更改的情况，新时钟源和旧时钟源相同，因此旧时钟源将立即就绪。器件在等待切换时可能进入休眠状态，如**第7.3.2节“时钟切换和休眠”**所述。

当新振荡器就绪时，OSCCON3的新振荡器已就绪（NOSCR）位和相应PIR寄存器的时钟切换中断标志（CSWIF）位置1。如果允许时钟切换中断（CSWIE = 1），则此时将产生中断。除了中断，还可查询OSCCON3的振荡器就绪（ORDY）位来确定振荡器何时已就绪。

注：	CSWIF中断不会将系统从休眠模式唤醒。

如果OSCCON3的时钟切换保持（CSWHOLD）位清零，在新振荡器已就绪（NOSCR）位置1时将发生振荡器切换，并且将以新振荡器设置处理中断（如果允许了中断的话）。

如果CSWHOLD置1，振荡器切换将暂停，而执行将继续使用当前（旧）时钟源。当NOSCR位置1时，软件应：

- 设置CSWHOLD = 0以完成切换，或
- 将COSC复制到NOSC以放弃切换。

如果DOZE有效，那么将在下一个时钟周期发生切换，而无论CPU在该周期期间是否正在运行。

更改时钟后分频比而不更改时钟源（如，将FOSC从1 MHz更改为2 MHz）的处理方式与更改时钟源相同，如前面所述。时钟源已经在工作，因此切换相对较快。CSWHOLD必须清零（CSWHOLD = 0），切换才能完成。

当前COSC和CDIV在OSCCON2寄存器中指示，直至发生实际切换，此时OSCCON2将更新且ORDY置1。NOSCR由硬件清零以指示切换完成。

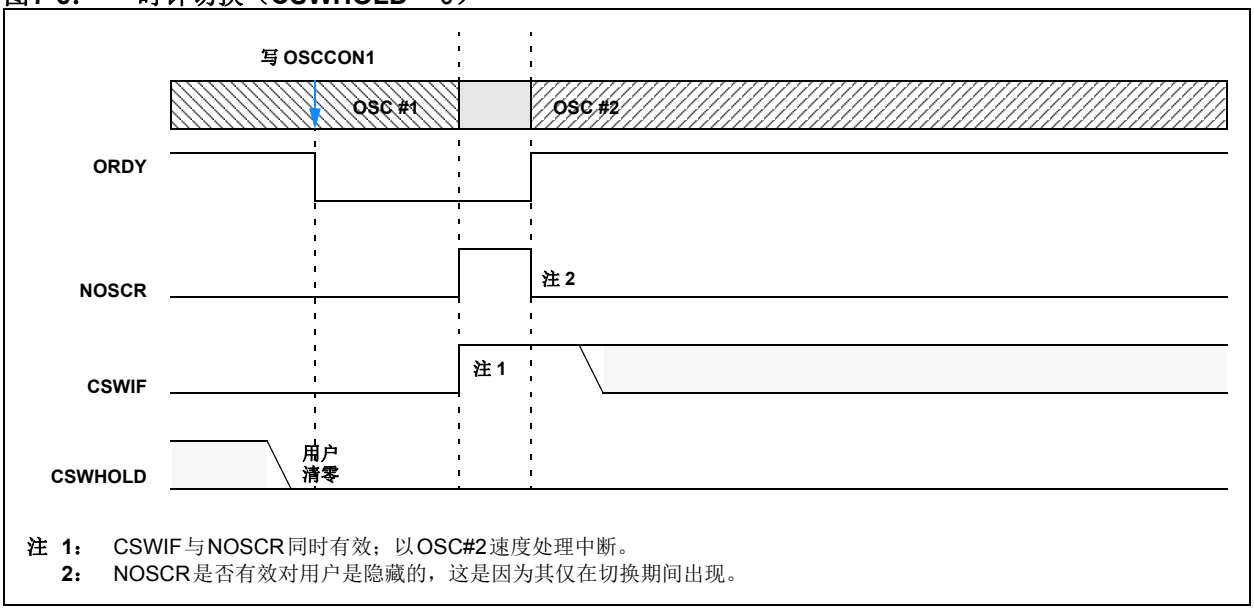
7.3.2 时钟切换和休眠

如果OSCCON1写入新值并且在切换完成之前将器件置于休眠模式，那么切换将不会发生且器件进入休眠模式。

器件从休眠模式唤醒且CSWHOLD位清零时，器件唤醒时将采用“新”时钟，时钟切换中断标志位（CSWIF）将置1。

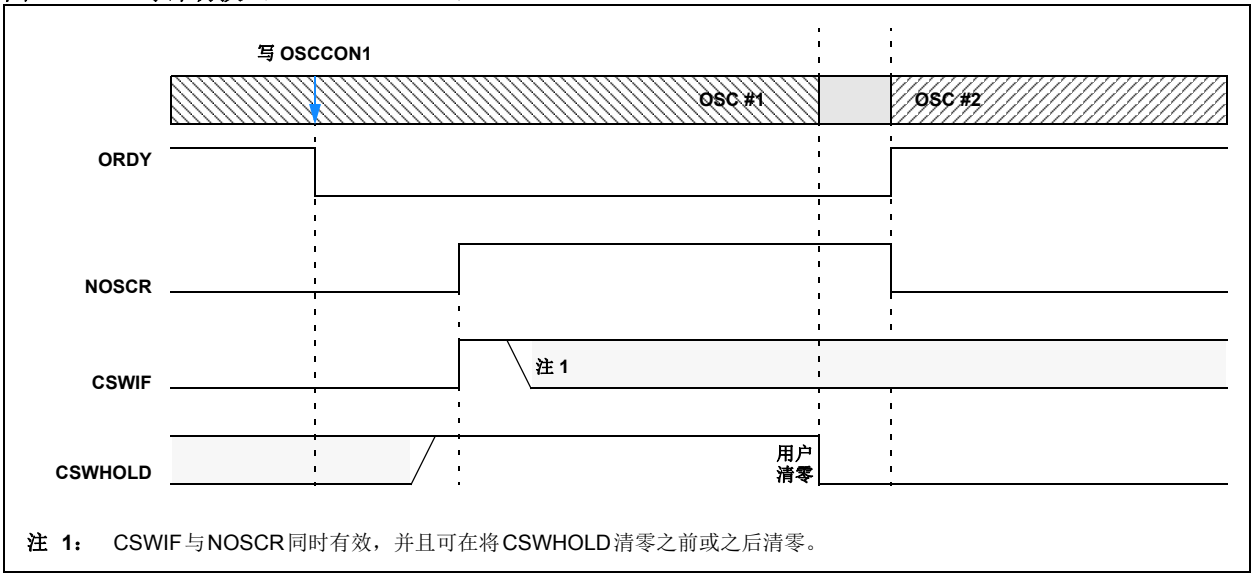
当器件从休眠模式唤醒且CSWHOLD位置1时，器件唤醒时将采用“旧”时钟并且再次请求新时钟。

图7-6： 时钟切换（CSWHOLD = 0）



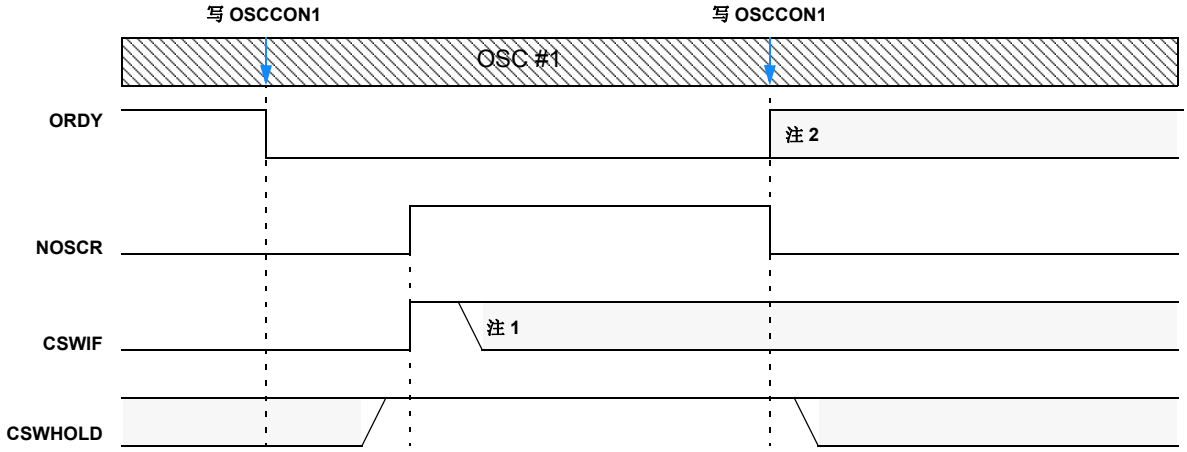
注 1： CSWIF与NOSCR同时有效；以OSC#2速度处理中断。
2： NOSCR是否有效对用户是隐藏的，这是因为其仅在切换期间出现。

图7-7： 时钟切换（CSWHOLD = 1）



注 1： CSWIF与NOSCR同时有效，并且可在将CSWHOLD清零之前或之后清零。

图7-8： 放弃时钟切换

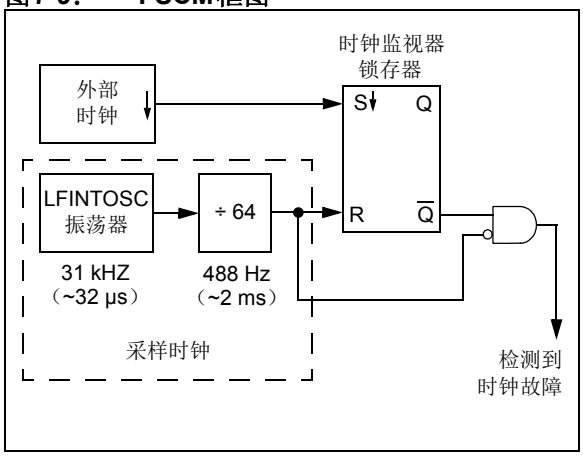


注 1： CSWIF可在重写OSCCON1之前或之后清零；不自动清零CSWIF。
2： 如果OSCCON1与OSCCON2不匹配，则ORDY = 0；将开始新的切换。

7.4 故障保护时钟监视器

故障保护时钟监视器（FSCM）使得器件在出现外部振荡器故障时仍能继续工作。FSCM通过将配置字中的FCMEN位置1来使能。FSCM可用于所有外部振荡器模式（LP、XT、HS、ECL/M/H和辅助振荡器）。

图7-9： FSCM框图



7.4.1 故障保护检测

FSCM模块通过将外部振荡器与FSCM采样时钟比较来检测振荡器故障。通过对LFINTOSC时钟进行64分频得到采样时钟。请参见图7-9。故障检测器内部有一个锁存器。在外部时钟的每个下降沿，外部时钟将锁存器置1。在采样时钟的每个上升沿，采样时钟将锁存器清零。如果已经经过采样时钟的整个半周期，但外部时钟仍未变为低电平，则会检测到故障。

7.4.2 故障保护工作原理

当外部时钟出现故障时，FSCM会改写COSC位以选择HFINTOSC（3'b110）。HFINTOSC的频率将由FRQ位和NDIV/CDIV位先前的状态确定。相应PIR寄存器的标志位OSFIF置1，且如果相应PIR寄存器的OSFIE位也置1，将产生中断。器件固件随后会采取措施减轻可能由故障时钟所产生的问题。系统时钟将继续由内部时钟源提供，直到器件固件成功重启外部振荡器并切换回外部振荡器进行工作，这通过写OSCCON1寄存器的NOSC和NDIV位实现。

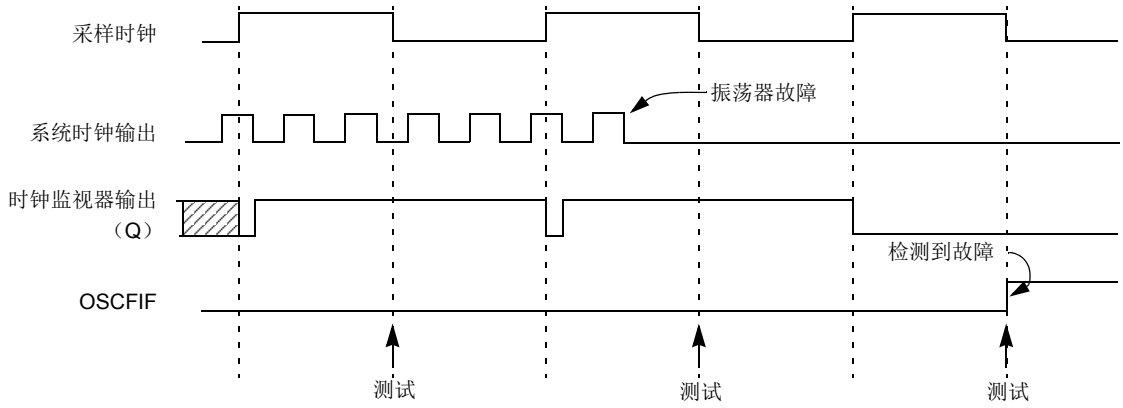
7.4.3 故障保护条件清除

在复位、执行SLEEP指令或更改OSCCON1寄存器的NOSC和NDIV位之后，故障保护条件被清除。当切换到外部振荡器或PLL时，重启OST。OST运行时，器件将依靠OSCCON1中选定的INTOSC继续工作。OST超时后，故障保护条件会在成功切换到外部时钟源之后被清除。在切换到外部时钟源之前，应先清零OSCFIF位。如果故障保护条件仍然存在，硬件会再次将OSCFIF标志置1。

7.4.4 复位或从休眠中唤醒

FSCM设计为用于在振荡器起振定时器（OST）延时结束后检测振荡器故障。从休眠模式唤醒后以及任何类型的复位后使用OST。OST不与EC时钟模式一起使用，因此FSCM将在复位或唤醒后立即生效。

图7-10： FSCM时序图



注： 系统时钟频率通常远高于采样时钟频率，本例选用的相对频率仅是为了便于说明。

表7-1： NOSC/COSC和NDIV/CDIV位设置

NOSC<2:0> COSC<2:0>	时钟源
111	EXTOSC(1)
110	HFINTOSC(2)
101	LFINTOSC
100	SOSC
011	保留
010	EXTOSC + 4x PLL(3)
001	保留
000	保留

NDIV<3:0> CDIV<3:0>	时钟分频比
1111-1010	保留
1001	512
1000	256
0111	128
0110	64
0101	32
0100	16
0011	8
0010	4
0001	2
0000	1

注 1： EXTOSC通过配置字1（寄存器5-1）的EXTOSC位配置。
2： HFINTOSC频率通过OSCFRQ寄存器（寄存器7-5）的FRQ位设置。
3： EXTOSC必须符合PLL规范（表45-9）。

7.5 寄存器定义：振荡器控制

寄存器7-1： OSCCON1：振荡器控制寄存器1

U-0	R/W-f/f	R/W-f/f	R/W-f/f	R/W-q/q	R/W-q/q	R/W-q/q	R/W-q/q
—	NOSC<2:0>			NDIV<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	f = 由配置位设置确定
		q = 复位值由硬件确定

bit 7 **未实现：**读为0

bit 6-4 **NOSC<2:0>：**新振荡器源请求位(1,2,3)
设置按照表7-1请求源振荡器和PLL组合。
POR值 = RSTOSC（寄存器5-1）。

bit 3-0 **NDIV<3:0>：**新分频比选择请求位(2,3)
设置按照表7-1确定后分频器新分频比。

注 1： 默认值（f/f）由RSTOSC配置位确定。请参见下面的表7-2。
2： 如果NOSC中写入保留值（表7-1），则该操作将被忽略，且不会对NOSC和NDIV执行写操作。
3： 当CSWEN = 0时，该寄存器为只读且不能更改为POR值以外的值。

表7-2： 默认振荡器设置

RSTOSC	SFR复位值			初始Fosc频率
	NOSC/COSC	CDIV	OSCFRQ	
111	111	1:1	4 MHz	EXTOSC（根据FEXTOSC配置）
110	110	4:1		FOSC = 1 MHz（4 MHz/4）
101	101	1:1		LFINTOSC
100	100	1:1		SOSC
011	保留			
010	010	1:1	4 MHz	EXTOSC + 4xPLL(1)
001	保留			
000	110	1:1	64 MHz	FOSC = 64 MHz

注 1： EXTOSC必须符合PLL规范（表45-9）。

寄存器7-2：　OSCCON2：振荡器控制寄存器2

U-0	R-f/f	R-f/f	R-f/f	R-f/f	R-f/f	R-f/f	R-f/f
—	COSC<2:0>			CDIV<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7　**未实现：**读为0

bit 6-4　**COSC<2:0>：**当前振荡器源选择位（只读）(1)
按照表7-1指示当前源振荡器和PLL组合。

bit 3-0　**CDIV<3:0>：**当前分频比选择位（只读）(1)
按照表7-1指示后分频器当前分频比。

注 1： POR值是开始执行用户代码时的值。

寄存器7-3：　OSCCON3：振荡器控制寄存器3

R/W/HC-0/0	R/W-0/0	U-0	R-0/0	R-0/0	U-0	U-0	U-0
CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7　**CSWHOLD：**时钟切换保持位
1 = 当NOSC选择的振荡器就绪时，时钟切换将保持（通过中断）
0 = 当NOSC选择的振荡器就绪时，时钟切换可能继续；当NOSCR变为1时，将发生切换

bit 6　**SOSCPWR：**辅助振荡器功耗模式选择位
1 = 辅助振荡器工作在高功耗模式下
0 = 辅助振荡器工作在低功耗模式下

bit 5　**未实现：**读为0

bit 4　**ORDY：**振荡器就绪位（只读）
1 = OSCCON1 = OSCCON2；当前系统时钟为NOSC所指定的时钟
0 = 正在进行时钟切换

bit 3　**NOSCR：**新振荡器就绪位（只读）(1)
1 = 正在进行时钟切换并且NOSC所选的振荡器指示“就绪”条件
0 = 未在进行时钟切换，或者NOSC选择的振荡器尚未就绪

bit 2-0　**未实现：**读为0

注 1： 如果CSWHOLD = 0，则用户可能无法看到该位置1，因为当振荡器就绪时，可能需要经过1个指令时钟的延时后该位才会置1。时钟切换发生在下一个指令周期，且该位清零。

寄存器7-4： OSCSTAT：振荡器状态寄存器1

R-q/q	R-q/q	R-q/q	R-q/q	R-q/q	R-q/q	U-0	R-q/q
EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLLR
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 复位值由硬件确定

bit 7 **EXTOR**：EXTOSC（外部）振荡器就绪位
1 = 振荡器已就绪备用
0 = 振荡器未使能，或尚未就绪备用

bit 6 **HFOR**：HFINTOSC振荡器就绪位
1 = 振荡器已就绪备用
0 = 振荡器未使能，或尚未就绪备用

bit 5 **MFOR**：MFINTOSC振荡器就绪位
1 = 振荡器已就绪备用
0 = 振荡器未使能，或尚未就绪备用

bit 4 **LFOR**：LFINTOSC振荡器就绪位
1 = 振荡器已就绪备用
0 = 振荡器未使能，或尚未就绪备用

bit 3 **SOR**：辅助（Timer1）振荡器就绪位
1 = 振荡器已就绪备用
0 = 振荡器未使能，或尚未就绪备用

bit 2 **ADOR**：ADC振荡器就绪位
1 = 振荡器已就绪备用
0 = 振荡器未使能，或尚未就绪备用

bit 1 **未实现**：读为0

bit 0 **PLLR**：PLL就绪位
1 = PLL已就绪备用
0 = PLL未使能，所需输入源未就绪，或PLL未锁定。

寄存器7-5：　OSCFRQ：HFINTOSC频率选择寄存器

U-0	U-0	U-0	U-0	R/W-q/q	R/W-q/q	R/W-q/q	R/W-q/q
—	—	—	—	FRQ<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 复位值由硬件确定

bit 7-4　**未实现：**读为0

bit 3-0　**FRQ<3:0>：**HFINTOSC频率选择位(1)

FRQ<3:0>	标称频率（MHz）
1001	保留
1010	
1111	
1110	
1101	
1100	
1011	
1000	64
0111	48
0110	32
0101	16
0100	12
0011	8
0010	4
0001	2
0000	1

注 1： 更多信息，请参见表7-2。

寄存器7-6： OSCTUNE：HFINTOSC调节寄存器

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	TUN<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 5-0 **TUN<5:0>：**HFINTOSC频率调节位

01 1111 = 最高频率

•

•

•

00 0000 = 中心频率。振荡器模块以校准的频率运行（默认值）。

•

•

•

10 0000 = 最低频率

寄存器7-7：　　OSCEN：振荡器手动使能寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0
EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7　　**EXTOEN**：外部振荡器手动请求使能位
1 =　已明确使能EXTOSC，具体操作由FEXTOSC指定
0 =　可通过请求外设使能EXTOSC

bit 6　　**HFOEN**：HFINTOSC振荡器手动请求使能位
1 =　已明确使能HFINTOSC，具体操作由OSCFRQ（寄存器7-5）指定
0 =　可通过请求外设使能HFINTOSC

bit 5　　**MFOEN**：MFINTOSC（500 kHz/31.25 kHz）振荡器手动请求使能位（源自HFINTOSC）
1 =　已明确使能MFINTOSC
0 =　可通过请求外设使能MFINTOSC

bit 4　　**LFOEN**：LFINTOSC（31 kHz）振荡器手动请求使能位
1 =　已明确使能LFINTOSC
0 =　可通过请求外设使能LFINTOSC

bit 3　　**SOSCEN**：辅助振荡器手动请求使能位
1 =　已明确使能辅助振荡器，具体操作由SOSCPWR指定
0 =　可通过请求外设使能辅助振荡器

bit 2　　**ADOEN**：ADC振荡器手动请求使能位
1 =　已明确使能ADC振荡器
0 =　可通过请求外设使能ADC振荡器

bit 1-0　　**未实现**：读为0

表7-3：　与时钟源相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
OSCCON1	—	NOSC<2:0>			NDIV<3:0>				94
OSCCON2	—	COSC<2:0>			CDIV<3:0>				95
OSCCON3	CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—	95
OSCSTAT	EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLLR	96
OSCTUNE	—	—	TUN<5:0>						98
OSCFRQ	—	—	—	—	FRQ<3:0>				97
OSCEN	EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—	99

图注： **— =** 未实现位，读为0。时钟源不使用阴影单元。

8.0 参考时钟输出模块

参考时钟输出模块能够将时钟信号发送到参考时钟输出引脚（CLKR）。参考时钟输出还可用作其他外设的信号，如数据信号调制器（DSM）、存储器扫描器和定时器模块。

参考时钟输出模块具有以下特性：

- 可通过CLKRCLK寄存器选择时钟源
- 可编程的时钟分频比
- 可选择的占空比

图8-1： 参考时钟框图

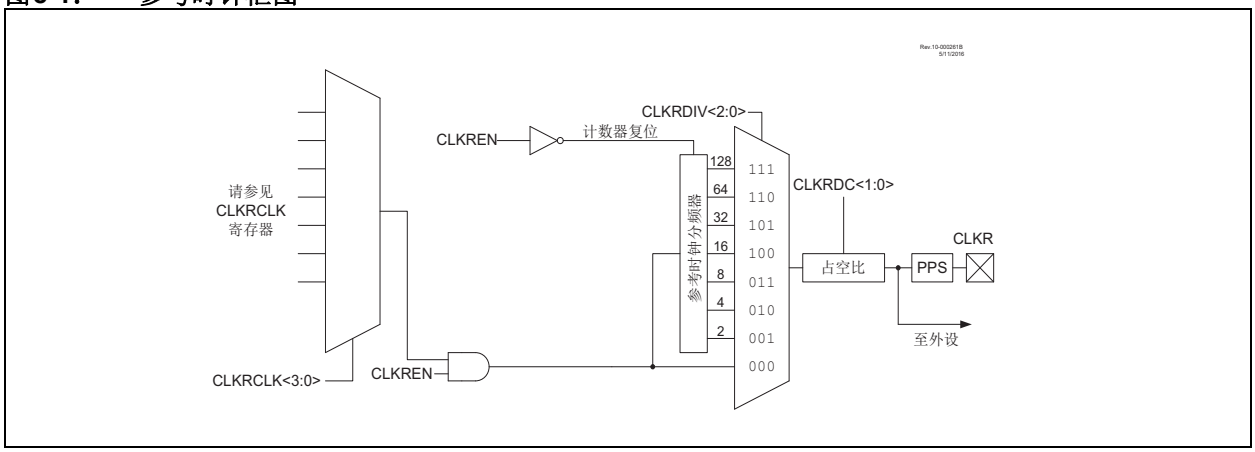
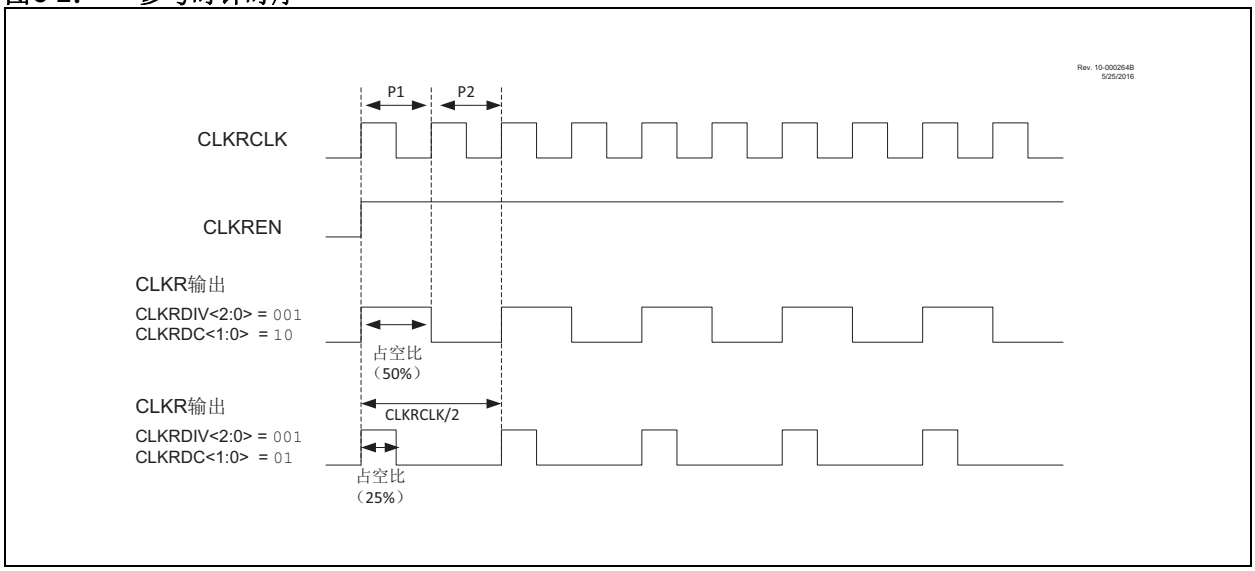


图8-2： 参考时钟时序



8.1 时钟源

可以使用CLKRCLK寄存器选择参考时钟输出的输入。

8.1.1 时钟同步

参考时钟使能（EN）位置1后，可确保模块在启动时无毛刺。

当禁止参考时钟输出时，输出信号将立即禁止。

当使能模块时，可更改时钟分频比和时钟占空比，但输出可能出现毛刺。为了避免可能的毛刺，时钟分频比和时钟占空比应仅在CLKREN清零时更改。

8.2 可编程的时钟分频比

模块采用时钟输入并根据CLKRCON寄存器（寄存器8-1）的DIV<2:0>位的值对其进行分频。

可基于DIV<2:0>位进行以下配置：

- 基本FOSC值
- FOSC 2分频
- FOSC 4分频
- FOSC 8分频
- FOSC 16分频
- FOSC 32分频
- FOSC 64分频
- FOSC 128分频

时钟分频比值可在使能模块时更改；但是，为了防止对输出产生毛刺，只应在禁止模块（EN = 0）时更改DIV<2:0>位。

8.3 可选择的占空比

CLKRCON寄存器的DC<1:0>位可以用来修改输出时钟的占空比。对于所有时钟速率，可选择25%、50%或75%占空比，但未经分频的基本FOSC值除外。

占空比可在使能模块时更改；但是，为了防止对输出产生毛刺，只应在禁止模块（EN = 0）时更改DC<1:0>位。

注：	DC1位复位为1。因此，默认占空比为50%，而非0%。

8.4 休眠模式下的操作

参考时钟输出模块时钟基于系统时钟。当器件进入休眠状态时，模块输出将保持其当前状态。这将直接影响将参考时钟输出用作输入信号的外设。进入或退出休眠状态时，模块不应发生任何变化。

8.5 寄存器定义：参考时钟

下面给出了参考时钟外设的长位名称前缀。更多信息，请参见**第1.3.2.2节“长位名称”**。

外设	位名称前缀
CLKR	CLKR

寄存器8-1： CLKRCON：参考时钟控制寄存器

R/W-0/0	U-0	U-0	R/W-1/1	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN	—	—	DC<1:0>		DIV<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN：**参考时钟模块使能位
1 = 使能参考时钟模块
0 = 禁止参考时钟模块

bit 6-5 **未实现：**读为0

bit 4-3 **DC<1:0>：**参考时钟占空比位(1)
11 = 时钟输出占空比为75%
10 = 时钟输出占空比为50%
01 = 时钟输出占空比为25%
00 = 时钟输出占空比为0%

bit 2-0 **DIV<2:0>：**参考时钟分频比位
111 = 基本时钟值进行128分频
110 = 基本时钟值进行64分频
101 = 基本时钟值进行32分频
100 = 基本时钟值进行16分频
011 = 基本时钟值进行8分频
010 = 基本时钟值进行4分频
001 = 基本时钟值进行2分频
000 = 基本时钟值

注 1： 这些位对于等于或大于2的参考时钟分频比值有效，基本时钟无法再进一步分频。

寄存器8-2：　　CLKRCLK：参考时钟选择多路开关

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	CLK<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4　　**未实现：**读为0

bit 3-0　　**CLK<3:0>：**CLKR时钟选择位

1111 = 保留
•
•
•
1011 = 保留
1010 = CLC4输出
1001 = CLC3输出
1000 = CLC2输出
0111 = CLC1输出
0110 = NCO1输出
0101 = SOSC
0100 = MFINTOSC（31.25 kHz）
0011 = MFINTOSC（500 kHz）
0010 = LFINTOSC（31 kHz）
0001 = HFINTOSC
0000 = FOSC

表8-1：　　与参考时钟输出相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
CLKRCON	EN	—	—	DC<1:0>		DIV<2:0>			103
CLKRCLK	—	—	—	—	—	CLK<2:0>			104

图注：　— = 未实现，读为0。CLKR模块不使用阴影单元。

9.0 中断控制器

向量中断控制器模块将诸多外设中断请求信号缩减为一个送往CPU的中断请求信号。该模块具有以下主要特性：

- 中断向量表（Interrupt Vector Table，IVT），其中每个中断源对应惟一的向量
- 固定且可确保的中断延时
- 中断向量表（IVT）具有可编程基址和锁定功能
- 两个用户可选择的优先级——高优先级和低优先级
- 两级现场保护
- 中断状态位，用于指示CPU的当前执行状态

中断控制器模块汇集所有中断请求信号，并根据固定的自然顺序优先级（即，由中断向量表确定）和用户分配的优先级（即，由IPRx寄存器确定）解决中断问题，从而无需扫描中断源。

9.1 中断控制和状态寄存器

该系列器件实现了以下用于中断控制器的寄存器：

- INTCON0和INTCON1控制寄存器
- PIRx——外设中断状态寄存器
- PIEx——外设中断允许寄存器
- IPRx——外设中断优先级寄存器
- IVTBASE<20:0>地址寄存器
- IVTLOCK寄存器

全局中断控制功能和外部中断由INTCON0寄存器控制。INTCON1寄存器包含中断控制器的状态标志。

PIRx寄存器包含所有中断请求标志。每个中断源都具有一个中断标志状态位，该状态位由相应的外设中断或外部中断信号置1，通过软件进行清零。

PIEx寄存器包含所有中断允许位。这些控制位用于单独允许外设中断或外部中断信号。

IPRx寄存器用于设置每个中断源的中断优先级。可以为每个用户中断源分配高优先级或低优先级。

IVTBASE寄存器可以由用户编程，用于确定中断向量表的起始地址，IVTLOCK寄存器用于防止对IVTBASE寄存器执行任何意外写操作。

此外，还有另外两个配置位，用于控制中断控制器的配置方式。

- CONFIG2L<3>——MVECEN位
- CONFIG2L<4>——IVT1WAY位

CONFIG2L中的MVECEN位用于确定是否使用向量表来确定中断优先级。

- IVT1WAY用于确定IVTLOCKED位在器件复位后可清零和置1的次数。有关详细信息，请参见**第9.2.3节“中断向量表（IVT）地址计算”**。

9.2 中断向量表（IVT）

中断控制器支持中断向量表（IVT），其中包含每个中断请求源的向量地址单元。

中断向量表（IVT）位于程序存储器中，从IVTBASE寄存器确定的地址单元开始；有关详细信息，请参见寄存器9-33至9-35。IVT包含68个向量，每个中断源对应一个。每个中断向量单元包含相关中断服务程序（Interrupt Service Routine，ISR）的起始地址。

配置字2L中的MVECEN位用于控制向量表的可用性。

9.2.1 中断向量表基址（IVTBASE）

向量表的起始地址可由用户通过IVTBASE寄存器进行编程。用户必须确保起始地址能够包含程序存储器中的整个向量表。

每个向量地址为16位字（即PIC18器件上的两个地址单元）。对于n个中断源，必须使用2n个地址单元来保存表，该表从IVTBASE（作为第一个地址单元）开始。因此，选择IVTBASE的起始地址时应确保将IVTBASE到（IVTBASE + 2n-1）的地址范围包含在闪存程序存储器中。

例如，K42器件具有最大向量编号：81。因此，选择IVTBASE时应确保（IVTBASE + 0xA1）小于闪存程序存储器中的最后一个存储单元。

在不同组向量表之间切换的情形下，可编程向量表基址非常有用，具体取决于应用程序。此外，在应用程序需要更新现有向量表（向量地址值）时，也可以使用该功能。

注： 用户需要为IVTBASE寄存器分配一个偶数地址才能实现正确操作。

9.2.2 中断向量表的内容

MVECEN = 0

当MVECEN = 0时，IVTBASE寄存器指向的地址单元有一条针对高优先级中断的GOTO指令。同样，相应的低优先级向量单元也有一条GOTO指令，该指令在低优先级中断的情况下执行。

MVECEN = 1

当MVECEN = 1时，每个中断的向量表中的值指向中断服务程序的第一条指令的地址单元。

ISR单元 = 中断向量表条目 << 2。

9.2.3 中断向量表（IVT）地址计算

MVECEN = 0

当配置字2L（寄存器5-3）中的MVECEN位清零时，使用IVTBASE寄存器所指向的地址作为高优先级中断向量地址。低优先级中断向量地址与IVTBASE寄存器中的地址偏移8个指令字。

对于PIC18器件，IVTBASE寄存器默认为00 0008h，高优先级中断向量地址将为00 0008h，低优先级中断向量地址将为00 0018h。

MVECEN = 1

每个中断具有与之相关的惟一向量编号，如表9-2中所定义。此向量编号用于计算特定中断源的中断向量单元。

中断向量地址 = IVTBASE + (2*向量编号)。

当接收到中断时，计算得到的这一中断向量地址值存储在IVTAD<20:0>寄存器中（寄存器9-36至9-38）。

用户使用IPRx寄存器分配的软件优先级不影响地址计算，仅用于解决并发中断问题。

如果由于某种原因无法从向量表中获取ISR的地址，则将导致系统复位并清零PCON1寄存器（寄存器6-3）中的存储器执行违例标志（MEMV位）。这是由于以下原因之一造成的：

- 向量表中的中断条目位于可执行PFM区域之外（当SAFEN = 1时，SAF区域是不可执行的）。
- 向量表所指向的ISR位于可执行PFM区域之外（当SAFEN = 1时，SAF区域是不可执行的）。

表9-1： IVT地址计算汇总

IVT地址计算		中断优先级INTCON0寄存器，IPEN位	
		0	1
多向量使能 CONFIG 2L寄存器 MVECEN位	0	IVTBASE	高优先级 IVTBASE
			低优先级 IVTBASE + 8个字
	1	IVTBASE + 2*(向量编号)	

9.2.4 IVTBASE寄存器的访问控制

中断控制器具有IVTLOCKED位，将该位置1可避免对IVTBASE寄存器内容作出意外更改。置1和清零该位需要一个特殊序列作为额外的预防措施，以防止意外的更改。

为了允许对IVTBASE寄存器的写操作，必须禁止中断（GIEH = 0），并且必须清零IVTLOCKED位。用户必须遵循例9-1给出的序列来清零IVTLOCKED位。

例9-1： IVT解锁序列

```
; Disable Interrupts:
        BCF             INTCON0, GIE;
; Bank to IVTLOCK register
        BANKSEL         IVTLOCK;
        MOVLW           55h;

; Required sequence, next 4 instructions
        MOVWF           IVTLOCK;
        MOVLW           AAh;
        MOVWF           IVTLOCK;

; Clear IVTLOCKED bit to enable writes
        BCF             IVTLOCK, IVTLOCKED;

; Enable Interrupts
        BSF             INTCON0, GIE;
```

用户必须遵循例9-2给出的序列来将IVTLOCKED位置1。

例9-2： IVT锁定序列

```
; Disable Interrupts:
        BCF             INTCON0, GIE;
; Bank to IVTLOCK register
        BANKSEL         IVTLOCK;
        MOVLW           55h;

; Required sequence, next 4 instructions
        MOVWF           IVTLOCK;
        MOVLW           AAh;
        MOVWF           IVTLOCK;

; Set IVTLOCKED bit to enable writes
        BSF             IVTLOCK, IVTLOCKED;

; Enable Interrupts
        BSF             INTCON0, GIE;
```

当IVT1WAY配置位置1时，IVTLOCKED位只能在器件复位之后清零和置1一次。在使用例9-2中的锁定序列将IVTLOCK置1后，例9-1中的解锁操作将不起作用。在系统复位发生之前，禁止解锁。

9.3 中断优先级

任何待处理中断源的最终优先级均先由IPRx寄存器中该中断源的用户分配优先级确定，再由IVT内的自然顺序优先级确定。下面几节详细介绍了中断优先级的工作原理。

9.3.1 用户（软件）优先级

可以通过将INTCON0寄存器（寄存器9-1）的IPEN位置1来允许用户分配中断优先级。用户可以为每个外设中断源分配高优先级或低优先级。每个中断的用户可分配中断优先级控制位位于IPRx寄存器（寄存器9-23至寄存器9-32）中。

基于下面定义的预定义中断优先级方案来处理中断。

1. 用户设置为高优先级的中断具有更高的执行优先级。在以下情况下，高优先级中断将覆盖低优先级请求：
   a) 已请求低优先级中断或其请求已处于待处理状态。
   b) 同时触发低优先级中断和高优先级中断，即在同一指令周期(1)。
   c) 请求了低优先级中断，并且当前正在执行相应的中断服务程序。在这种情况下，低优先级中断程序将在高优先级中断处理后完成执行过程(2)。
2. 用户设置为低优先级的中断具有较低的执行优先级，会被高优先级中断抢占。
3. 使用相同软件优先级定义的中断不能相互抢占或中断。使用自然顺序优先级（当MVECEN = ON时）或按ISR中轮询中断标志位的顺序（当MVECEN = OFF时）解决具有相同用户优先级的并发待处理中断问题。

注 1： 当高优先级中断抢占并发低优先级中断时，会在高优先级中断服务程序中将GIEL位清零。如果GIEL位清零，即使最初时已请求低优先级中断，也不会对其进行处理。需要在用户代码中清零相应中断标志。

2： 如果在正执行低优先级中断服务程序时请求高优先级中断，可以在高优先级中断服务程序中将GIEL位清零。即使GIEL位清零，待处理的低优先级中断也将恢复。

9.3.2 自然顺序（硬件）优先级

当请求多个具有相同用户指定优先级的中断时，将通过使用一种名为“自然顺序优先级”的方法来解决优先级冲突。自然顺序优先级是基于中断向量表的固定优先级机制。表9-2列出了自然顺序优先级和为每个中断源分配的中断向量编号。

表9-2： 中断向量优先级表

向量编号	中断源	向量编号	中断源
0	软件中断	42	TXB0IF
1	HLVD	43	TXB1IF
2	OSF	44	TXB2IF/TXBnIF
3	CSW	45	ERRIF
4	NVM	46	WAKIF
5	SCAN	47	IRXIF
6	CRC	48	C2
7	IOC	49	SMT2
8	INT0	50	SMT2PRA
9	ZCD	51	SMT2PWA
10	AD	52	DMA2SCNT
11	ADT	53	DMA2DCNT
12	C1	54	DMA2OR
13	SMT1	55	DMA2A
14	SMT1PRA	56	I2C2RX
15	SMT1PWA	57	I2C2TX
16	DMA1SCNT	58	I2C2
17	DMA1DCNT	59	I2C2E
18	DMA1OR	60	U2RX
19	DMA1A	61	U2TX
20	SPI1RX	62	U2E
21	SPI1TX	63	U2
22	SPI1	64	TMR3
23	I2C1RX	65	TMR3G
24	I2C1TX	66	TMR4
25	I2C1	67	CCP2
26	I2C1E	68	CWG2
27	U1RX	69	CLC2
28	U1TX	70	INT2
29	U1E	71	TMR5
30	U1	72	TMR5G
31	TMR0	73	TMR6
32	TMR1	74	CCP3
33	TMR1G	75	CWG3
34	TMR2	76	CLC3
35	CCP1	77	CCP4
36	NCO	78	CLC4
37	CWG1	79	—
38	CLC1	80	—
39	INT1	81	—
40	RXB0IF/FIFOIF		
41	RXB1IF/RXBnIF		

自然顺序优先级机制将向量中断0作为最高优先级，将向量中断81作为最低优先级。

例如，如果两个并行发生的中断源均使用IPRx寄存器指定为高优先级，则将使用自然顺序优先级解决（即向量编号较低的中断将优先于向量编号较高的中断）。

用户能够为每个中断源分配高优先级或低优先级意味着用户程序可以为具有低自然顺序优先级的中断分配较高的总体优先级。

9.4 中断操作

所有待处理中断均通过将PIRx寄存器中的相应标志位置1来指示。所有待处理中断均使用**第9.3节“中断优先级”**中说明的优先级机制来解决。

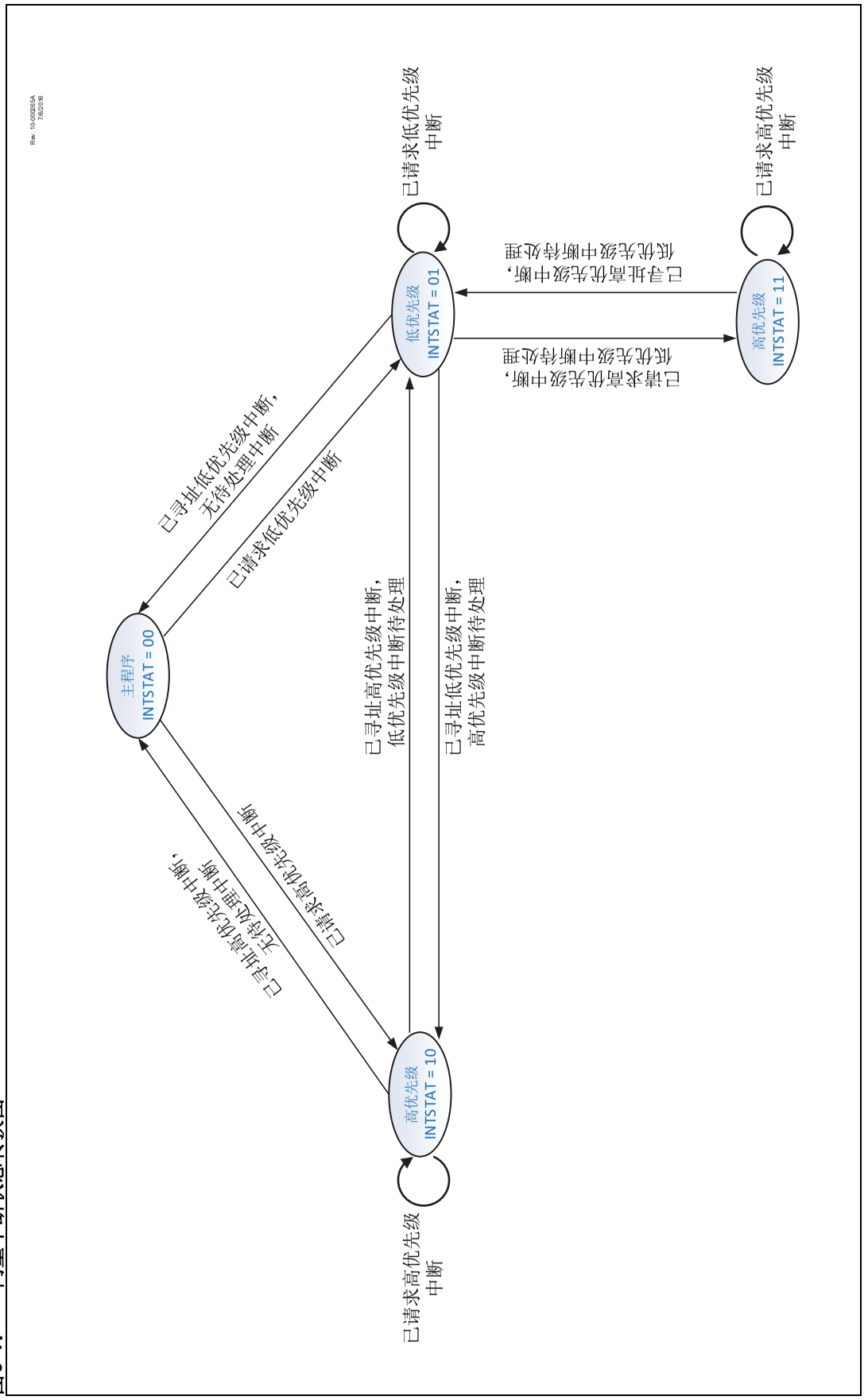
一旦待处理的中断源得到解决，程序执行便会指向已解决的中断向量地址，如**第9.2节“中断向量表（IVT）”**所述。向量编号也存储在WREG寄存器中。大多数标志位需要由应用程序软件清零，但在某些情况下，器件硬件会自动清除中断。PIRx寄存器中的一些标志位是只读位，这些标志位是中断源的汇总，中断源的相应中断标志位必须清零。

在主程序中，有效中断可以是高优先级中断或低优先级中断；在低优先级中断服务程序中，有效中断可以是高优先级中断。根据中断请求的接收顺序及其相应时序，CPU将处于INTCON1寄存器（寄存器9-2）的STAT位指示的执行状态。

图9-1所示的状态机和后续章节将详细介绍以不同顺序接收时的中断执行情况。

注：	处理中断时，硬件不会改变GIEH/L的状态。内部状态机用于跟踪执行状态。可以在用户代码中操作这些位，从而将执行转移到主程序并忽略现有中断。

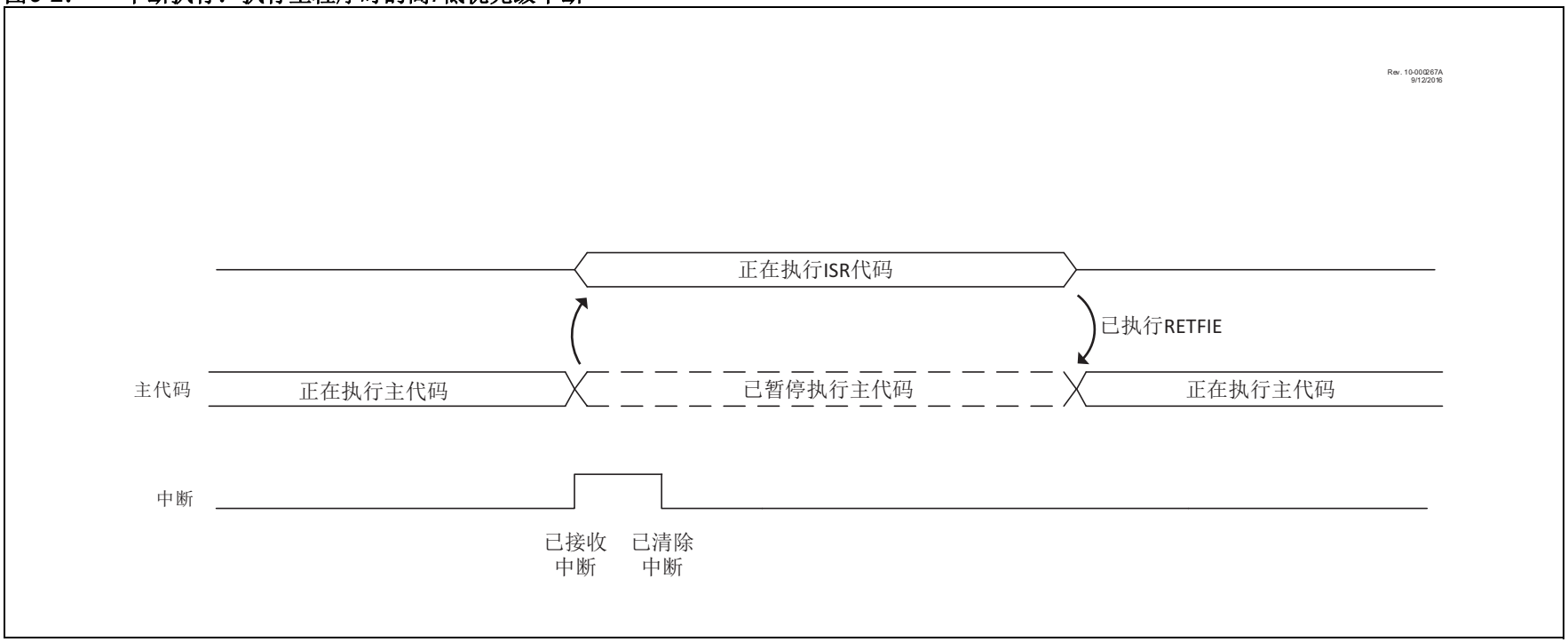
图 9-1：　向量中断状态转换图



9.4.1 在执行主程序代码时处理高优先级中断或低优先级中断

如果在执行主程序代码时请求高优先级中断或低优先级中断，则将暂停主程序执行来处理ISR，请参见图9-2。从ISR返回（通过执行RETFIE指令）后，主程序恢复执行。

图9-2： 中断执行：执行主程序时的高/低优先级中断

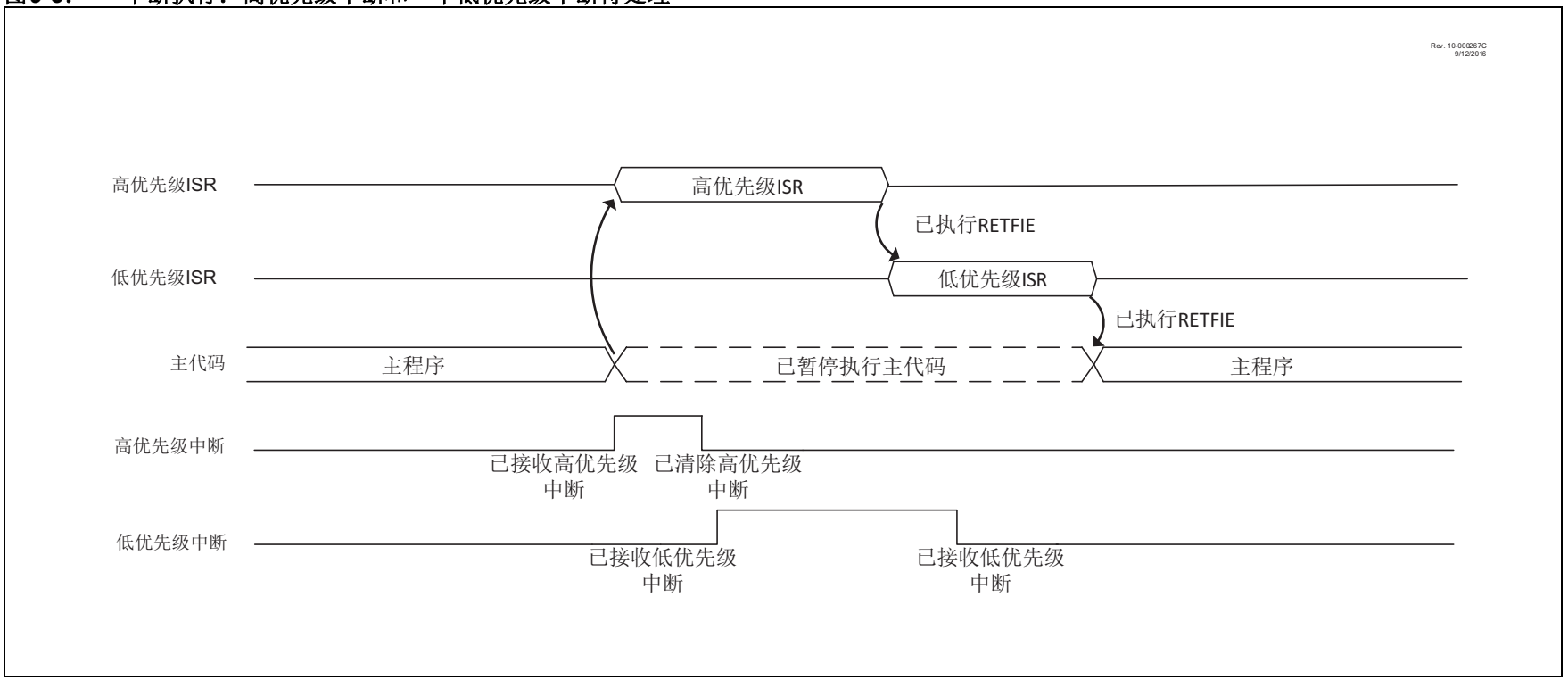


9.4.2 当低优先级中断待处理时处理高优先级中断

高优先级中断请求将始终优先于任何低优先级中断。首先应答高优先级中断，然后再应答低优先级中断。从高优先级ISR返回（通过执行RETFIE指令）后，将处理低优先级中断，请参见图9-3。

如果任何其他高优先级中断待处理且已允许，则将先处理它们，再处理待处理的低优先级中断。如果没有其他高优先级中断请求处于活动状态，则将处理低优先级中断。

图9-3： 中断执行：高优先级中断和一个低优先级中断待处理



9.4.3 抢占低优先级中断

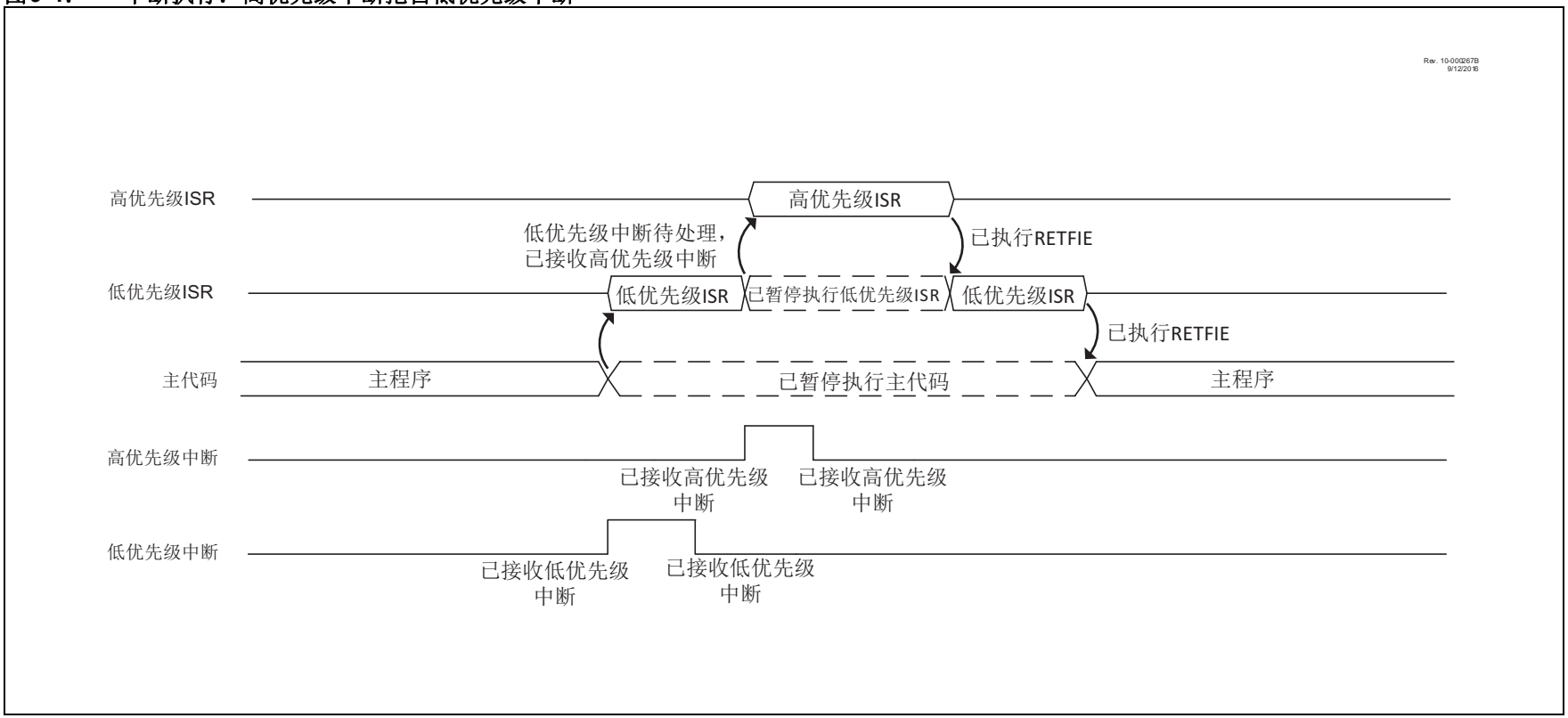
高优先级中断可抢占低优先级中断。当处于低优先级ISR中时，如果一个高优先级中断到达，将生成高优先级中断请求并暂停低优先级ISR，同时执行高优先级ISR，请参见图9-4。

在高优先级ISR完成后，如果任何其他高优先级中断请求均未处于活动状态，则将返回执行被抢占的低优先级ISR。

注 1： 必须清零高优先级中断标志以避免递归中断。

2： 如果一个低优先级ISR已在转移到高优先级ISR前处理到一半，则即使用户代码清零GIEL，也会将该低优先级ISR处理完。

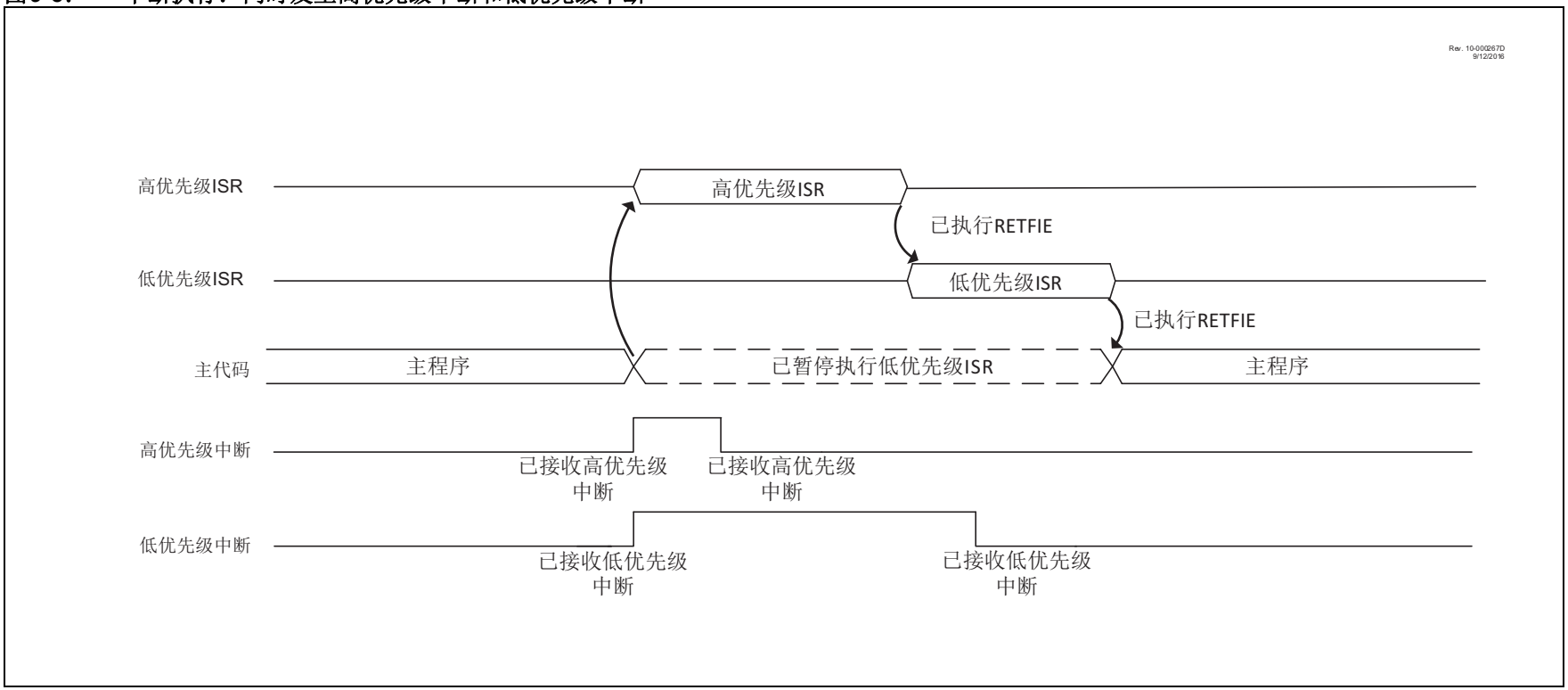
图9-4： 中断执行：高优先级中断抢占低优先级中断



9.4.4 同时发生高优先级中断和低优先级中断

当高优先级中断和低优先级中断在同一指令周期处于活动状态时（即，同时发生中断事件），则会同时生成高优先级中断请求和低优先级中断请求。先处理高优先级ISR，再处理低优先级中断，请参见图9-5。

图9-5： 中断执行：同时发生高优先级中断和低优先级中断



9.5 现场保护

中断控制器支持两级深现场保护（主程序现场和低优先级ISR现场）。有关详细信息，请参见图9-6中所示的状态机。

程序计数器（PC）保存在专用器件PC堆栈上。保存的CPU寄存器包括STATUS、WREG、BSR、FSR0/1/2、PRODL/H和PCLATH/U。

在WREG保存到现场寄存器后，待处理中断源的已解决向量编号将复制到WREG。现场保护和恢复操作通过中断控制器基于中断的当前状态及其发送到CPU的顺序完成。

现场保护/恢复操作在MVECEN的两个状态下以相同方式工作。当IPEN = 0时，只有一级中断处于活动状态。因此，接收到中断时，仅保存主现场。

9.5.1 访问影子寄存器

中断控制器将现场信息自动保存到Bank 56的影子寄存器中。所保存的两个现场值（即，主程序和低优先级ISR）可使用一组相同的影子寄存器访问。将SHADCON寄存器（寄存器9-40）中的SHADLO位清零，可访问所保存的主程序现场的CPU寄存器值；将CPU寄存器的SHADLO位置1，可访问所保存的低优先级ISR现场的值。退出高优先级ISR后，CPU寄存器将自动恢复低优先级ISR现场值。同样，退出低优先级ISR后，CPU寄存器将自动恢复主现场值。

Bank 56中的影子寄存器是可读写的，因此如果用户希望修改现场，则应修改相应的影子寄存器，退出ISR时将恢复相应值。根据用户的应用，可能还需要保存其他寄存器。

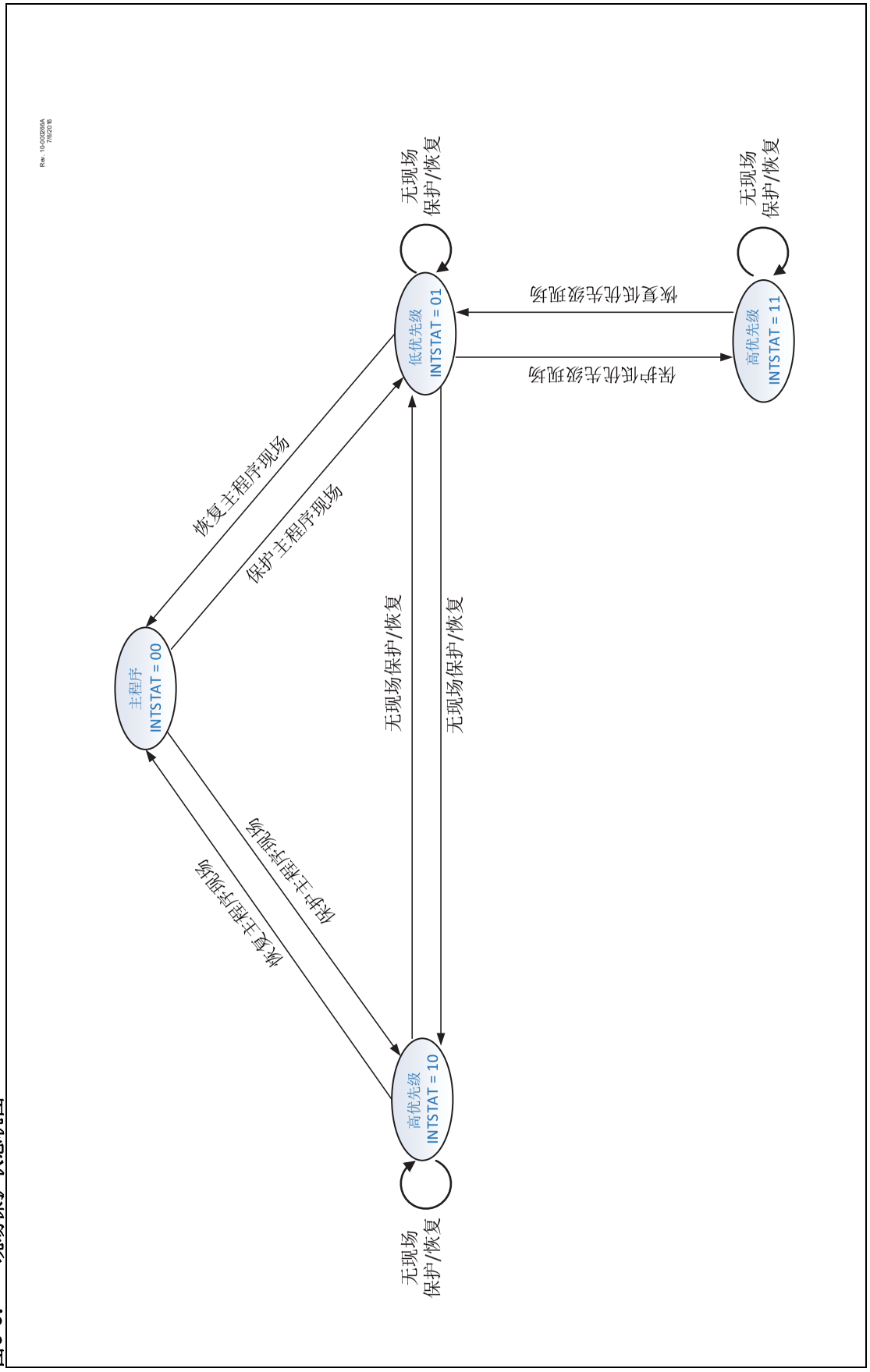


图 9-6: 现场保护状态机图

9.6 从中断服务程序（ISR）返回

“从中断返回”指令（RETFIE）用于标记ISR的结束。

执行RETFIE 1指令后，PC将从PC堆栈顶部装入保存的PC值。执行该指令还会恢复已保存的现场。因此，执行过程将返回到中断发生之前的工作状态。

执行RETFIE 0指令后，寄存器不会重新恢复保存的现场。

9.7 中断延时

为每个中断分配向量地址/数字后（MVECEN = 1），不需要扫描所有中断即可确定中断源。

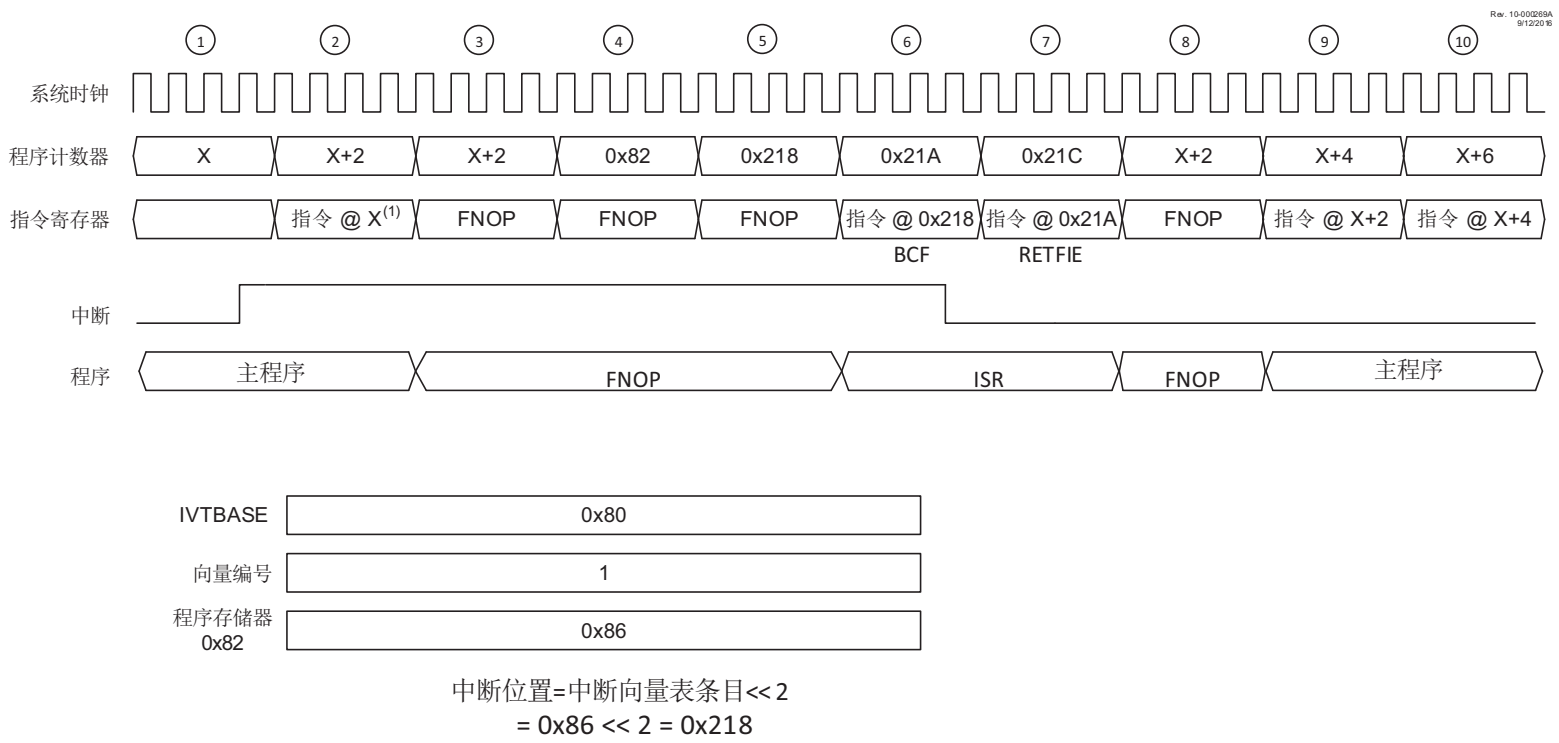
当MVECEN = 1时，向量中断控制器需要三个时钟周期从主程序指向ISR，从而消除了中断时序对编译代码的依赖性。

从中断发生时处于活动状态的指令完成到中断服务程序的第一条指令之间存在三个指令周期的固定延时。图9-7、图9-8和图9-9给出了在MVECEN = 1的情况下，如果最后执行的指令分别为单周期、双周期和三周期指令，将外设中断置为有效时的事件序列。

中断标志状态位置1后，当前指令完成执行。在第一个延时周期，PC、STATUS、WREG、BSR、FSR0/1/2、PRODL/H和PCLATH/U寄存器的内容将进行现场保护并且会计算IVTBASE+向量编号。在第二个延时周期，计算得到的中断源的向量表地址将装入PC，并且会获取ISR的起始地址。在第三个延时周期，ISR地址将装入PC。所有延时周期均作为一条FNOP指令执行。

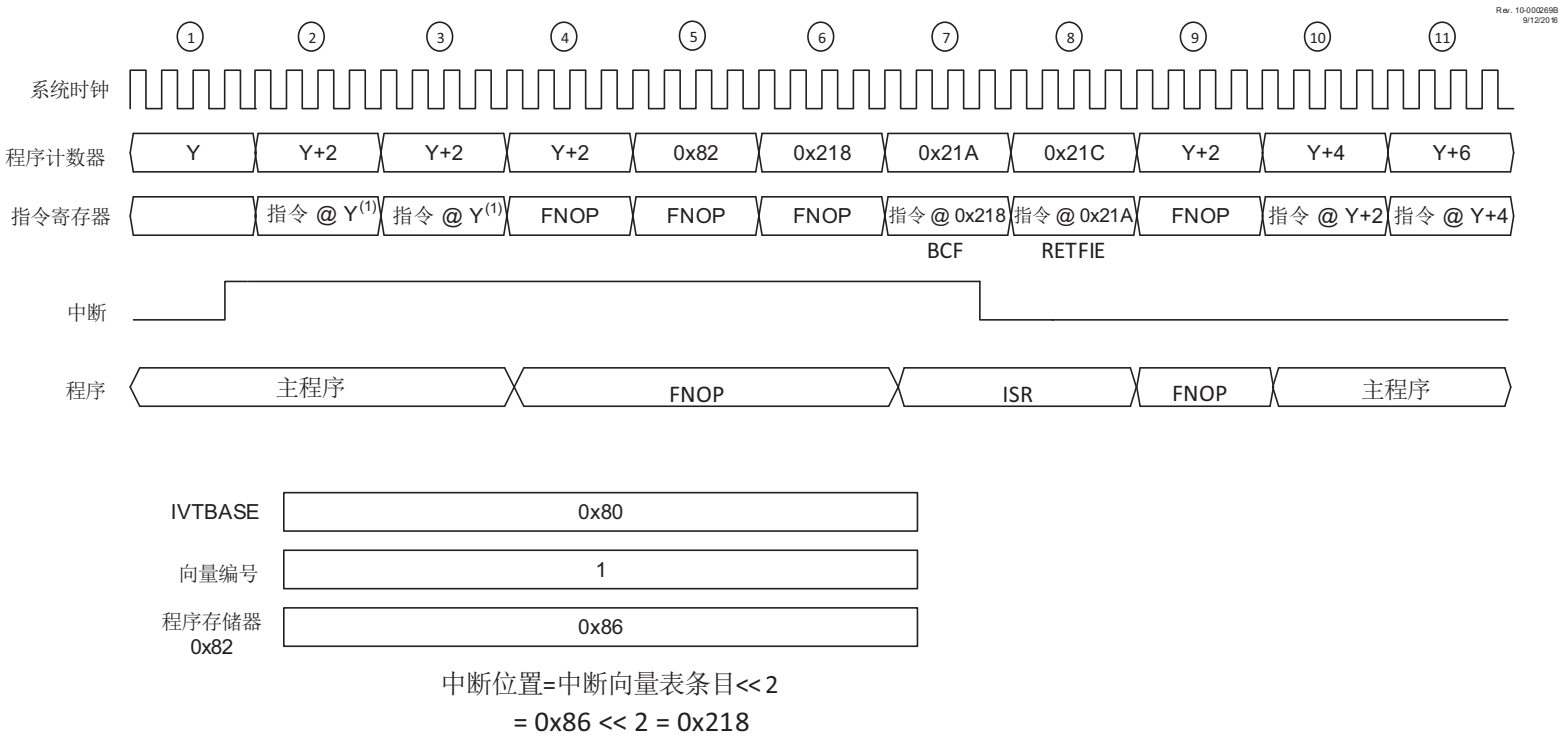
当MVECEN = 0时，向量中断控制器需要两个时钟周期从主程序指向ISR。从中断发生时处于活动状态的指令完成到中断服务程序的第一条指令之间存在两个指令周期外加软件延时的一段延时。

图9-7： 中断时序图——单周期指令



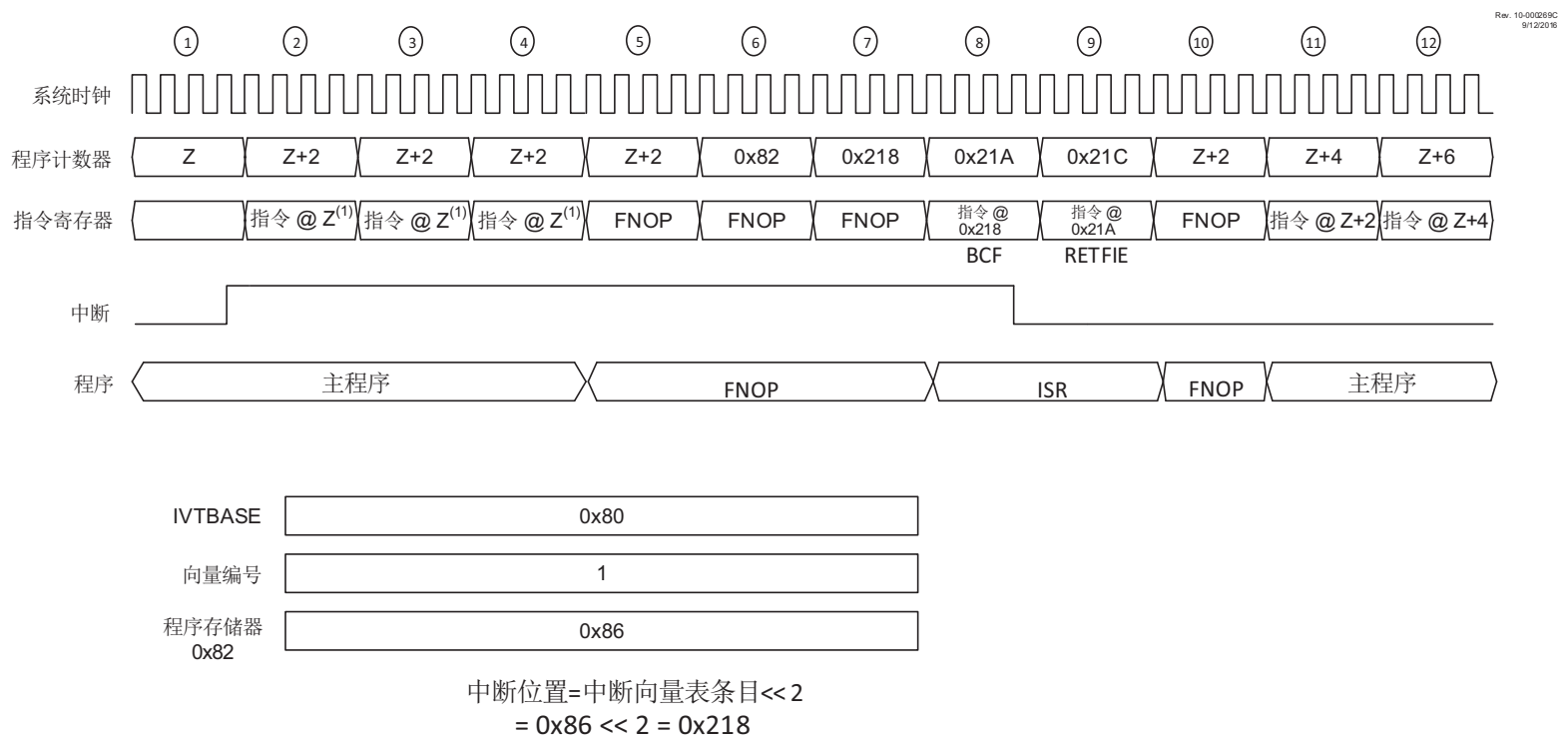
注 1： 指令@X是一条单周期指令

图9-8： 中断时序图——双字指令



注 1： 指令@Y是一条双周期指令。

图9-9： 中断时序图——三周期指令



注 1： 指令@ Z是一条三周期指令。

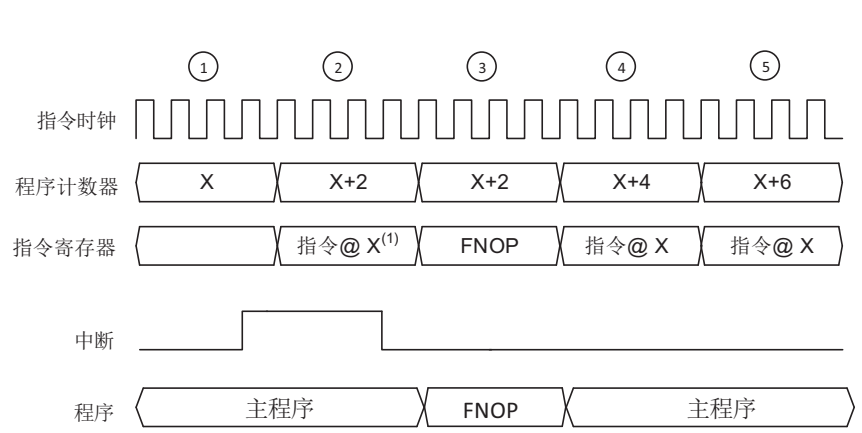
9.7.1 中止中断

如果中断控制器从主程序指向ISR之前的最后一条指令将与中断相关的GIE、PIE或PIR位清零，则控制器会在返回主程序之前执行一个强制NOP周期。

图9-10给出了最后执行的指令周期中外设中断先置为有效、再清零的事件序列。

如果与中断相关的GIE、PIE或PIR位在指向ISR之前清零，则控制器继续执行主程序。

图9-10： 中断时序图——中止中断



注 1： 指令@ X清零中断标志，例如BCF INTCON0, GIE。

9.8 中断设置过程

1. 当使用中断优先级时，需将INTCON0寄存器中的IPEN位置1，然后通过写入适当IPRx控制寄存器中的控制位为中断源选择用户分配优先级。

注：	器件复位时，IPRx寄存器将初始化，以便为所有用户中断源分配高优先级。

2. 将相关PIRx状态寄存器中与外设关联的中断标志状态位清零。
3. 通过将适当PIEx控制寄存器中与中断源关联的中断允许控制位置1来允许中断源。
4. 如果使用向量表（MVECEN = 1），则需使用IVTBASE寄存器设置中断向量表的起始地址。请参见**第9.2.2节“中断向量表的内容”**。
5. 写入IVTBASE后，将INTCON0寄存器中的中断允许位置1。
6. 例9-3和例9-4给出了使用汇编语言和C语言设置中断和ISR的例子。

9.9 外部中断引脚

PIC18(L)F26/27/45/46/47/55/56/57K42器件有三个外部中断源，可以根据PPS设置分配给不同端口上的任何引脚。有关可能的重新布线选项，请参见**第17.0节“外设引脚选择（PPS）模块”**。外部中断源是边沿触发的。如果INTCON0寄存器中的相应INTxEDG位置1（= 1），则上升沿触发中断。如果该位清零，则下降沿触发中断。

当INTx引脚上出现一个有效边沿时，PIRx寄存器中的相应标志位INTxF将置1。可通过将相应允许位INTxE清零来禁止该中断。在重新允许该中断前，必须在中断服务程序中先用软件将标志位INTxF清零。

如果INTxE位在进入空闲或休眠模式之前置1，则所有外部中断（INT0、INT1和INT2）均可将处理器从这两种模式中唤醒。如果全局中断允许位GIE/GIEH置1，则处理器将在唤醒后跳转到中断向量。中断优先级由IPRx寄存器的中断优先级位INT0IP、INT1IP和INT2IP中包含的值确定。

9.10 从休眠模式唤醒

如果允许中断事件，则每当中断事件发生时，中断控制器都会向CPU提供一个唤醒请求，与器件处于运行、空闲/打盹还是休眠模式无关。GIEH/GIEL位的状态不影响唤醒请求。唤醒请求与所有时钟异步。

9.11 中断兼容性

当配置字2L（寄存器5-3）中的MVECEN位清零时，将禁止中断向量表功能，中断与以前的高性能8位PIC18单片机器件兼容。在该模式下，中断向量表优先级没有影响。

当IPEN位也清零时，将禁止中断优先级功能，中断与PIC®16单片机中档器件兼容。由于中断优先级被禁止，所有中断均跳转到地址0008h。

例9-3： 使用MPASM设置向量中断

```
ISR_TMR0:  CODE        0x8C0                 ; ISR code at 0x08C0 in PFM
        BANKSEL        PIR0                  ; Select bank for PIR0
        BCF            PIR3, TMR0IF          ; Clear TMR0IF
        BTG            LATC, 0, ACCESS       ; Code to execute in ISR
        RETFIE         1                     ; Return from ISR

InterruptInit:
        BANKSEL        INTCON0               ; Select bank for INTCON0
        BSF            INTCON0, GIEH         ; Enable high priority interrupts
        BSF            INTCON0, GIEL         ; Enable low priority interrupts
        BSF            INTCON0, IPEN_INTCON0 ; Enable interrupt priority

        BANKSEL        PIE0                  ; Select bank for PIE0
        BSF            PIE3, TMR0IE          ; Enable TMR0 interrupt
        BSF            PIE4, TMR1IE          ; Enable TMR1 interrupt

        BCF            IPR3, TMR0IP          ; Make TMR0 interrupt low priority
        RETURN         1

VectorTableInit:
        ; Set IVTBASE (optional - default is 0x000008)
        MOVLW          0x00                  ; This is optional
        MOVWF          IVTBASEU, ACCESS      ; If not included, then the
        MOVLW          0x40                  ; hardware default value of
        MOVWF          IVTBASEH, ACCESS      ; 0x0008 will be taken.
        MOVLW          0x08
        MOVWF          IVTBASEL, ACCESS

        ; TMR0 vector at IVTBASE + 2*(TMR0 vector number i.e. 31) = 0x4046
        MOVLW          0x00                  ; Load TBLPTR with the
        MOVWF          TBLPTRU, ACCESS       ; PFM memory location to be
        MOVLW          0x40                  ; written to.
        MOVWF          TBLPTRH, ACCESS
        MOVLW          0x46
        MOVWF          TBLPTRL, ACCESS

        ; Write the contents of TMR0 vector location
        ; ISR_TMR0_ADDRESS >> 2 = 0x08C0 >> 2 = 0x0230
        MOVLW          0x30                  ; Low byte first
        MOVWF          TABLAT, ACCESS
        TBLWT*+                              ; Write to temp table latch

        MOVLW          0x02                  ; High byte next
        MOVWF          TABLAT, ACCESS
        TBLWT*+                              ; Write to temp table latch

        ; Write to PFM now using NVMCON
        BANKSEL        NVMCON1               ; Select bank for NVMCON1
        MOVLW          0x84                  ; Setting to write to PFM
        MOVWF          NVMCON1

        MOVLW          0x55                  ; Required unlock sequence
        MOVWF          NVMCON2
        MOVLW          0xAA
        MOVWF          NVMCON2
        BSF            NVMCON1, WR           ; Start writing to PFM

        BTFSC          NVMCON1, WR           ; Wait for write to complete
        GOTO           $-2

        RETURN         1
```

例9-4： 使用XC8设置向量中断

```
// NOTE 1: If IVTBASE is changed from its default value of 0x000008, then the
// "base(...)" argument must be provided in the ISR. Otherwise the vector
// table will be placed at 0x0008 by default regardless of the IVTBASE value.

// NOTE 2: When MVECEN=0 and IPEN=1, a separate argument as "high_priority"
// or "low_priority" can be used to distinguish between the two ISRs.
// If the argument is not provided, the ISR is considered high priority
// by default.

// NOTE 3: Multiple interrupts can be handled by the same ISR if they are
// specified in the "irq(...)" argument. Ex: irq(IRQ_TMR0, IRQ_CCP1)

void __interrupt(irq(IRQ_TMR0), base(0x4008)) TMR0_ISR(void)
{
        PIR3bits.TMR0IF = 0;                 // Clear the interrupt flag
        LATCbits.LC0 ^= 1;                   // ISR code goes here
}

void __interrupt(irq(default), base(0x4008)) DEFAULT_ISR(void)
{
        // Unhandled interrupts go here
}

void INTERRUPT_Initialize (void)
{
        INTCON0bits.GIEH = 1;                // Enable high priority interrupts
        INTCON0bits.GIEL = 1;                // Enable low priority interrupts
        INTCON0bits.IPEN = 1;                // Enable interrupt priority

        PIE3bits.TMR0IE = 1;                 // Enable TMR0 interrupt
        PIE4bits.TMR1IE = 1;                 // Enable TMR1 interrupt

        IPR3bits.TMR0IP = 0;                 // Make TMR0 interrupt low priority

        // Change IVTBASE if required
        IVTBASEU = 0x00;                     // Optional
        IVTBASEH = 0x40;                     // Default is 0x0008
        IVTBASEL = 0x08;
}
```

9.12 寄存器定义：中断控制

寄存器9-1： INTCON0：中断控制寄存器0

R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1
GIE/GIEH	GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **GIE/GIEH：**全局中断允许位

如果IPEN = 0：

GIE：

1 = 允许所有未被屏蔽的中断
0 = 禁止所有中断

如果IPEN = 1：

GIEH：

1 = 允许所有未被屏蔽的高优先级中断：要允许低优先级中断，还需要将相应位置1
0 = 禁止所有中断

bit 6 **GIEL：**全局低优先级中断允许位

如果IPEN = 0：

保留，读为0。

如果IPEN = 1：

GIEL：

1 = 允许所有未被屏蔽的低优先级中断，要允许低优先级中断，还需要将GIEH置1
0 = 禁止所有低优先级

bit 5 **IPEN：**中断优先级允许位

1 = 允许中断优先级
0 = 禁止中断优先级；所有中断均视为高优先级中断

bit 4-3 **未实现：**读为0

bit 2 **INT2EDG：**外部中断2边沿选择位

1 = INT2引脚的上升沿触发中断
0 = INT2引脚的下降沿触发中断

bit 1 **INT1EDG：**外部中断1边沿选择位

1 = INT1引脚的上升沿触发中断
0 = INT1引脚的下降沿触发中断

bit 0 **INT0EDG：**外部中断0边沿选择位

1 = INT0引脚的上升沿触发中断
0 = INT0引脚的下降沿触发中断

寄存器9-2： INTCON1：中断控制寄存器1

R-0/0	R-0/0	U-0	U-0	U-0	U-0	U-0	U-0
STAT<1:0>		—	—	—	—	—	—
bit 7							bit 0

图注：		
HC = 硬件清零位		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-6 **STAT<1:0>：**中断状态位

11 = 正在执行高优先级ISR，执行低优先级ISR时已收到高优先级中断
10 = 正在执行高优先级ISR，在主程序中已收到高优先级中断
01 = 正在执行低优先级ISR，在主程序中已收到低优先级中断
00 = 正在执行主程序

bit 5-0 **未实现：**读为0

寄存器9-3： PIR0：外设中断请求寄存器0

R-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
IOCIF(2)	CRCIF	SCANIF	NVMIF	CSWIF(3)	OSFIF	HLVDIF	SWIF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7 **IOCIF**：电平变化中断的中断标志位(2)
1 = 已发生中断
0 = 未发生中断事件

bit 6 **CRCIF**：CRC中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 5 **SCANIF**：存储器扫描器中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 4 **NVMIF**：NVM中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 3 **CSWIF**：时钟切换中断标志位(3)
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 2 **OSFIF**：振荡器故障中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 1 **HLVDIF**：HLVD中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 0 **SWIF**：软件中断标志位
1 = 软件中断标志使能
0 = 软件中断标志禁止

注 1： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。
2： IOCIF是只读位。要清除中断条件，必须将IOCxF寄存器中的所有位均清零。
3： CSWIF中断不会将系统从休眠模式唤醒。系统将保持休眠状态，直到另一个中断触发唤醒。

寄存器9-4：　　PIR1：外设中断请求寄存器1

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
SMT1PWAIF	SMT1PRAIF	SMT1IF	C1IF	ADTIF	ADIF	ZCDIF	INT0IF(2)
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7　　**SMT1PWAIF**：SMT1脉宽采集中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 6　　**SMT1PRAIF**：SMT1周期采集中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 5　　**SMT1IF**：SMT1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 4　　**C1IF**：CMP1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 3　　**ADTIF**：ADC阈值中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 2　　**ADIF**：ADC中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 1　　**ZCDIF**：ZCD中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 0　　**INT0IF**：外部中断0中断标志位(2)
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

注 1：当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。
2：外部中断GPIO引脚通过INTxPPS寄存器选择。

寄存器9-5： PIR2：外设中断寄存器2(1)

R-0/0	R-0/0	R-0/0	R-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
I2C1RXIF(2)	SPI1IF(3)	SPI1TXIF(4)	SPI1RXIF(4)	DMA1AIF	DMA1ORIF	DMA1DCNTIF	DMA1SCNTIF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7 **I2C1RXIF**：I^2C1接收中断标志位(2)
1 = 已发生中断
0 = 未发生中断事件

bit 6 **SPI1IF**：SPI1中断标志位(3)
1 = 已发生中断
0 = 未发生中断事件

bit 5 **SPI1TXIF**：SPI1发送中断标志位(4)
1 = 已发生中断
0 = 未发生中断事件

bit 4 **SPI1RXIF**：SPI1接收中断标志位(4)
1 = 已发生中断
0 = 未发生中断事件

bit 3 **DMA1AIF**：DMA1中止中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 2 **DMA1ORIF**：DMA1溢出中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 1 **DMA1DCNTIF**：DMA1目标计数中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 0 **DMA1SCNTIF**：DMA1源计数中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

注 1： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。

2： I2CxTXIF和I2CxRXIF是只读位。要清除中断条件，必须将I2CxSTAT1寄存器中的CLRBF位置1。

3： SPIxIF是只读位。要清除中断条件，必须将SPIxINTF寄存器中的所有位均清零。

4： SPIxTXIF和SPIxRXIF是只读位，不能由软件置1/清零。

寄存器9-6：　PIR3：外设中断寄存器3(1)

R/W/HS-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
TMR0IF	U1IF(2)	U1EIF(3)	U1TXIF(4)	U1RXIF(4)	I2C1EIF(5)	I2C1IF(6)	I2C1TXIF(7)
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7　**TMR0IF**：TMR0中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 6　**U1IF**：UART1中断标志位(2)
1 = 已发生中断
0 = 未发生中断事件

bit 5　**U1EIF**：UART1帧错误中断标志位(3)
1 = 已发生中断
0 = 未发生中断事件

bit 4　**U1TXIF**：UART1发送中断标志位(4)
1 = 已发生中断
0 = 未发生中断事件

bit 3　**U1RXIF**：UART1接收中断标志位(4)
1 = 已发生中断
0 = 未发生中断事件

bit 2　**I2C1EIF**：I^2C1错误中断标志位(5)
1 = 已发生中断
0 = 未发生中断事件

bit 1　**I2C1IF**：I^2C1中断标志位(6)
1 = 已发生中断
0 = 未发生中断事件

bit 0　**I2C1TXIF**：I^2C1发送中断标志位(7)
1 = 已发生中断
0 = 未发生中断事件

注 1： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。
2： UxIF是只读位。要清除中断条件，必须将UxUIR寄存器中的所有位均清零。
3： UxEIF是只读位。要清除中断条件，必须将UxERRIR寄存器中的所有位均清零。
4： UxTXIF和UxRXIF是只读位，不能由软件置1/清零。
5： I2CxEIF是只读位。要清除中断条件，必须将I2CxERR寄存器中的所有位均清零。
6： I2CxIF是只读位。要清除中断条件，必须将I2CxPIR寄存器中的所有位均清零。
7： I2CxTXIF和I2CxRXIF是只读位。要清除中断条件，必须将I2CxSTAT1寄存器中的CLRBF位置1。

寄存器9-7：　　PIR4：外设中断寄存器4(1)

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
INT1IF(2)	CLC1IF	CWG1IF	NCO1IF	CCP1IF	TMR2IF	TMR1GIF	TMR1IF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7　　**INT1IF**：外部中断1中断标志位(2)
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 6　　**CLC1IF**：CLC1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 5　　**CWG1IF**：CWG1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 4　　**NCO1IF**：NCO1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 3　　**CCP1IF**：CCP1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 2　　**TMR2IF**：TMR2中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 1　　**TMR1GIF**：TMR1门控中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 0　　**TMR1IF**：TMR1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

注 1：当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。
2：外部中断GPIO引脚通过INTxPPS寄存器选择。

寄存器9-8： PIR5：外设中断寄存器5(1)

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
IRXIF	WAKIF	ERRIF	TXB2IF/TXBnIF	TXB1IF	TXB0IF	RXB1IF/RXBnIF	RXB0IF/FIFOFIF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7 **IRXIF**：收到CAN无效报文中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 6 **WAKIF**：CAN总线唤醒活动中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 5 **ERRIF**：CAN错误中断标志位（COMSTAT寄存器中有多个中断源）
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 4 **TXB2IF/TXBnIF**：CAN发送缓冲区2/发送缓冲区n中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 3 **TXB1IF**：CAN发送缓冲区1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 2 **TXB0IF**：CAN发送缓冲区0中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 1 **RXB1IF/RXBnIF**：CAN接收缓冲区1/接收缓冲区n中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 0 **RXB0IF/FIFOFIF**：CAN接收缓冲区0/FIFO已满1中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

注 1： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。

寄存器9-9： PIR6：外设中断寄存器6(1)

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
DMA2AIF	DMA2ORIF	DMA2DCNTIF	DMA2SCNTIF	SMT2PWAIF	SMT2PRAIF	SMT2IF	C2IF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7 **DMA2AIF**：DMA2中止中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 6 **DMA2ORIF**：DMA2溢出中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 5 **DMA2DCNTIF**：DMA2目标计数中断标志位
1 = 已发生中断
0 = 未发生中断事件

bit 4 **DMA2SCNTIF**：DMA2源计数中断标志位
1 = 已发生中断
0 = 未发生中断事件

bit 3 **SMT2PWAIF**：SMT2脉宽采集中断标志位
1 = 已发生中断
0 = 未发生中断事件

bit 2 **SMT2PRAIF**：SMT2周期采集中断标志位
1 = 已发生中断
0 = 未发生中断事件

bit 1 **SMT2IF**：SMT2中断标志位
1 = 已发生中断
0 = 未发生中断事件

bit 0 **C2IF**：C2中断标志位
1 = 已发生中断
0 = 未发生中断事件

注 1： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。

寄存器9-10： PIR7：外设中断寄存器7(1)

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
U2IF(2)	U2EIF(3)	U2TXIF(4)	U2RXIF(4)	I2C2EIF(5)	I2C2IF(6)	I2C2TXIF(7)	I2C2RXIF(7)
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7 **U2IF**：UART2中断标志位(2)
1 = 已发生中断
0 = 未发生中断事件

bit 6 **U2EIF**：UART2帧错误中断标志位(3)
1 = 已发生中断
0 = 未发生中断事件

bit 5 **U2TXIF**：UART2发送中断标志位(4)
1 = 已发生中断
0 = 未发生中断事件

bit 4 **U2RXIF**：UART2接收中断标志位(4)
1 = 已发生中断
0 = 未发生中断事件

bit 3 **I2C2EIF**：I^2C2错误中断标志位(5)
1 = 已发生中断
0 = 未发生中断事件

bit 2 **I2C2IF**：I^2C2中断标志位(6)
1 = 已发生中断
0 = 未发生中断事件

bit 1 **I2C2TXIF**：I^2C2发送中断标志位(7)
1 = 已发生中断
0 = 未发生中断事件

bit 0 **I2C2RXIF**：I^2C2接收中断标志位(7)
1 = 已发生中断
0 = 未发生中断事件

注 1： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。

2： UxIF是只读位。要清除中断条件，必须将UxUIR寄存器中的所有位均清零。

3： UxEIF是只读位。要清除中断条件，必须将UxERRIR寄存器中的所有位均清零。

4： UxTXIF和UxRXIF是只读位，不能由软件置1/清零。

5： I2CxEIF是只读位。要清除中断条件，必须将I2CxERR寄存器中的所有位均清零。

6： I2CxIF是只读位。要清除中断条件，必须将I2CxPIR寄存器中的所有位均清零。

7： I2CxTXIF和I2CxRXIF是只读位。要清除中断条件，必须将I2CxSTAT1寄存器中的CLRBF位置1。

寄存器9-11： PIR8：外设中断寄存器8(1)

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
TMR5IF	INT2IF(2)	CLC2IF	CWG2IF	CCP2IF	TMR4IF	TMR3GIF	TMR3IF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7 **TMR5IF**：TMR5中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 6 **INT2IF**：外部中断2中断标志位(2)
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 5 **CLC2IF**：CLC2中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 4 **CWG2IF**：CWG2中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 3 **CCP2IF**：CCP2中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 2 **TMR4IF**：TMR4中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 1 **TMR3GIF**：TMR3G门控中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 0 **TMR3IF**：TMR3中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

注 1： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。

2： 外部中断GPIO引脚通过INTxPPS寄存器选择。

寄存器9-12： PIR9：外设中断寄存器9(1)

U-0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
—	CLC4IF	CCP4IF	CLC3IF	CWG3IF	CCP3IF	TMR6IF	TMR5GIF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **未实现：**读为0

bit 6 **CLC4IF：**CLC4中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 5 **CCP4IF：**CCP4中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 4 **CLC3IF：**CLC3中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 3 **CWG3IF：**CWG3中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 2 **CCP3IF：**CCP3中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 1 **TMR6IF：**TMR6中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

bit 0 **TMR5GIF：**TMR5门控中断标志位
1 = 已发生中断（必须由软件清零）
0 = 未发生中断事件

注 1： 当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置1。用户软件应确保在允许一个中断前，先将相应的中断标志位清零。

寄存器9-13： PIE0：外设中断允许寄存器0

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCIE	CRCIE	SCANIE	NVMIE	CSWIE	OSFIE	HLVDIE	SWIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **IOCIE**：电平变化中断允许位
1 = 允许
0 = 禁止

bit 6 **CRCIE**：CRC中断允许位
1 = 允许
0 = 禁止

bit 5 **SCANIE**：存储器扫描器中断允许位
1 = 允许
0 = 禁止

bit 4 **NVMIE**：NVM中断允许位
1 = 允许
0 = 禁止

bit 3 **CSWIE**：时钟切换中断允许位
1 = 允许
0 = 禁止

bit 2 **OSFIE**：振荡器故障中断允许位
1 = 允许
0 = 禁止

bit 1 **HLVDIE**：HLVD中断允许位
1 = 允许
0 = 禁止

bit 0 **SWIE**：软件中断允许位
1 = 允许
0 = 禁止

寄存器9-14： PIE1：外设中断允许寄存器1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SMT1PWAIE	SMT1PRAIE	SMT1IE	C1IE	ADTIE	ADIE	ZCDIE	INT0IE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **SMT1PWAIE**：SMT1脉宽采集中断允许位
1 = 允许
0 = 禁止

bit 6 **SMT1PRAIE**：SMT1周期采集中断允许位
1 = 允许
0 = 禁止

bit 5 **SMT1IE**：SMT1中断允许位
1 = 允许
0 = 禁止

bit 4 **C1IE**：C1中断允许位
1 = 允许
0 = 禁止

bit 3 **ADTIE**：ADC阈值中断允许位
1 = 允许
0 = 禁止

bit 2 **ADIE**：ADC中断允许位
1 = 允许
0 = 禁止

bit 1 **ZCDIE**：ZCD中断允许位
1 = 允许
0 = 禁止

bit 0 **INT0IE**：外部中断0允许位
1 = 允许
0 = 禁止

寄存器9-15： PIE2：外设中断允许寄存器2

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
I2C1RXIE	SPI1IE	SPI1TXIE	SPI1RXIE	DMA1AIE	DMA1ORIE	DMA1DCNTIE	DMA1SCNTIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **I2C1RXIE：**I^2C1接收中断允许位
1 = 允许
0 = 禁止

bit 6 **SPI1IE：**SPI1中断允许位
1 = 允许
0 = 禁止

bit 5 **SPI1TXIE：**SPI1发送中断允许位
1 = 允许
0 = 禁止

bit 4 **SPI1RXIE：**SPI1接收中断允许位
1 = 允许
0 = 禁止

bit 3 **DMA1AIE：**DMA1中止中断允许位
1 = 允许
0 = 禁止

bit 2 **DMA1ORIE：**DMA1溢出中断允许位
1 = 允许
0 = 禁止

bit 1 **DMA1DCNTIE：**DMA1目标计数中断允许位
1 = 允许
0 = 禁止

bit 0 **DMA1SCNTIE：**DMA1源计数中断允许位
1 = 允许
0 = 禁止

寄存器9-16： PIE3：外设中断允许寄存器3

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TMR0IE	U1IE	U1EIE	U1TXIE	U1RXIE	I2C1EIE	I2C1IE	I2C1TXIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **TMR0IE**：TMR0中断允许位
1 = 允许
0 = 禁止

bit 6 **U1IE**：UART1中断允许位
1 = 允许
0 = 禁止

bit 5 **U1EIE**：UART1帧错误中断允许位
1 = 允许
0 = 禁止

bit 4 **U1TXIE**：UART1发送中断允许位
1 = 允许
0 = 禁止

bit 3 **U1RXIE**：UART1接收中断允许位
1 = 允许
0 = 禁止

bit 2 **I2C1EIE**：I^2C1错误中断允许位
1 = 允许
0 = 禁止

bit 1 **I2C1IE**：I^2C1中断允许位
1 = 允许
0 = 禁止

bit 0 **I2C1TXIE**：I^2C1发送中断允许位
1 = 允许
0 = 禁止

寄存器9-17： PIE4：外设中断允许寄存器4

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
INT1IE	CLC1IE	CWG1IE	NCO1IE	CCP1IE	TMR2IE	TMR1GIE	TMR1IE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **INT1IE**：外部中断1中断允许位
1 = 允许
0 = 禁止

bit 6 **CLC1IE**：CLC1中断允许位
1 = 允许
0 = 禁止

bit 5 **CWG1IE**：CWG1中断允许位
1 = 允许
0 = 禁止

bit 4 **NCO1IE**：NCO1中断允许位
1 = 允许
0 = 禁止

bit 3 **CCP1IE**：CCP1中断允许位
1 = 允许
0 = 禁止

bit 2 **TMR2IE**：TMR2中断允许位
1 = 允许
0 = 禁止

bit 1 **TMR1GIE**：TMR1门控中断允许位
1 = 允许
0 = 禁止

bit 0 **TMR1IE**：TMR1中断允许位
1 = 允许
0 = 禁止

寄存器9-18： PIE5：外设中断允许寄存器5

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IRXIE	WAKIE	ERRIE	TXB2IE/TXBnIE	TXB1IE	TXB0IE	RXB1IE/RXBnIE	RXB0IE/FIFOFIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **IRXIE**：收到CAN无效报文中断允许位
1 = 允许
0 = 禁止

bit 6 **WAKIE**：CAN总线唤醒活动中断允许位
1 = 允许
0 = 禁止

bit 5 **ERRIE**：CAN错误中断允许位
1 = 允许
0 = 禁止

bit 4 **TXB2IE/TXBnIE**：CAN发送缓冲区2/发送缓冲区n中断允许位
1 = 允许
0 = 禁止

bit 3 **TXB1IE**：CAN发送缓冲区1中断允许位
1 = 允许
0 = 禁止

bit 2 **TXB0IE**：CAN发送缓冲区0中断允许位
1 = 允许
0 = 禁止

bit 1 **RXB1IE/RXBnIE**：CAN接收缓冲区1/接收缓冲区n中断标志中断允许位
1 = 允许
0 = 禁止

bit 0 **RXB0IE/FIFOFIE**：CAN接收缓冲区0/FIFO已满中断允许位
1 = 允许
0 = 禁止

寄存器9-19： PIE6：外设中断允许寄存器6

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
DMA2AIE	DMA2ORIE	DMA2DCNTIE	DMA2SCNTIE	SMT2PWAIE	SMT2PRAIE	SMT2IE	C2IE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **DMA2AIE**：DMA中止中断允许位
1 = 允许
0 = 禁止

bit 6 **DMA2ORIE**：DMA2溢出中断允许位
1 = 允许
0 = 禁止

bit 5 **DMA2DCNTIE**：DMA2目标计数中断允许位
1 = 允许
0 = 禁止

bit 4 **DMA2SCNTIE**：DMA2源计数中断允许位
1 = 允许
0 = 禁止

bit 3 **SMT2PWAIE**：SMT2脉宽采集中断允许位
1 = 允许
0 = 禁止

bit 2 **SMT2PRAIE**：SMT2周期采集接收中断允许位
1 = 允许
0 = 禁止

bit 1 **SMT2IE**：SMT2中断允许位
1 = 允许
0 = 禁止

bit 0 **C2IE**：C2中断允许位
1 = 允许
0 = 禁止

寄存器9-20： PIE7：外设中断允许寄存器7

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
U2IE	U2EIE	U2TXIE	U2RXIE	I2C2EIE	I2C2IE	I2C2TXIE	I2C2RXIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **U2IE**：UART2中断允许位
1 = 允许
0 = 禁止

bit 6 **U2EIE**：UART2中断允许位
1 = 允许
0 = 禁止

bit 5 **U2TXIE**：UART2发送中断允许位
1 = 允许
0 = 禁止

bit 4 **U2RXIE**：UART2接收中断允许位
1 = 允许
0 = 禁止

bit 3 **I2C2EIE**：I^2C2错误中断允许位
1 = 允许
0 = 禁止

bit 2 **I2C2IE**：I^2C2中断允许位
1 = 允许
0 = 禁止

bit 1 **I2C2TXIE**：I^2C2发送中断允许位
1 = 允许
0 = 禁止

bit 0 **I2C2RXIE**：I^2C2接收中断允许位
1 = 允许
0 = 禁止

寄存器9-21： PIE8：外设中断允许寄存器8

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TMR5IE	INT2IE	CLC2IE	CWG2IE	CCP2IE	TMR4IE	TMR3GIE	TMR3IE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **TMR5IE**：TMR5中断允许位
1 = 允许
0 = 禁止

bit 6 **INT2IE**：外部中断2中断允许位
1 = 允许
0 = 禁止

bit 5 **CLC2IE**：CLC2中断允许位
1 = 允许
0 = 禁止

bit 4 **CWG2IE**：CWG2中断允许位
1 = 允许
0 = 禁止

bit 3 **CCP2IE**：CCP2中断允许位
1 = 允许
0 = 禁止

bit 2 **TMR4IE**：TMR4中断允许位
1 = 允许
0 = 禁止

bit 1 **TMR3GIE**：TMR3门控中断允许位
1 = 允许
0 = 禁止

bit 0 **TMR3IE**：TMR3中断允许位
1 = 允许
0 = 禁止

寄存器9-22： PIE9：外设中断允许寄存器9

U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	CLC4IE	CCP4IE	CLC3IE	CWG3IE	CCP3IE	TMR6IE	TMR5GIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **未实现：**读为0

bit 6 **CLC4IE：**CLC4中断允许位
1 = 允许
0 = 禁止

bit 5 **CCP4IE：**CCP4中断允许位
1 = 允许
0 = 禁止

bit 4 **CLC3IE：**CLC3中断允许位
1 = 允许
0 = 禁止

bit 3 **CWG3IE：**CWG3中断允许位
1 = 允许
0 = 禁止

bit 2 **CCP3IE：**CCP3中断允许位
1 = 允许
0 = 禁止

bit 1 **TMR6IE：**TMR6中断允许位
1 = 允许
0 = 禁止

bit 0 **TMR5GIE：**TMR5中断允许位
1 = 允许
0 = 禁止

寄存器9-23： IPR0：外设中断优先级寄存器0

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
IOCIP	CRCIP	SCANIP	NVMIP	CSWIP	OSFIP	HLVDIP	SWIP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **IOCIP**：电平变化中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **CRCIP**：CRC中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **SCANIP**：存储器扫描器中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **NVMIP**：NVM中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **CSWIP**：时钟切换中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **OSFIP**：振荡器故障中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **HLVDIP**：HLVD中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **SWIP**：软件中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-24： IPR1：外设中断优先级寄存器1

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
SMT1PWAIP	SMT1PRAIP	SMT1IP	C1IP	ADTIP	ADIP	ZCDIP	INT0IP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **SMT1PWAIP**：SMT1脉宽采集中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **SMT1PRAIP**：SMT1周期采集中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **SMT1IP**：SMT1中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **C1IP**：C1中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **ADTIP**：ADC阈值中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **ADIP**：ADC中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **ZCDIP**：ZCD中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **INT0IP**：外部中断0中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-25： IPR2：外设中断优先级寄存器2

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
I2C1RXIP	SPI1IP	SPI1TXIP	SPI1RXIP	DMA1AIP	DMA1ORIP	DMA1DCNTIP	DMA1SCNTIP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **I2C1RXIP**：I^2C1接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **SPI1IP**：SPI1发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **SPI1TXIP**：I^2C1发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **SPI1RXIP**：SPI1接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **DMA1AIP**：DMA1中止发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **DMA1ORIP**：DMA1溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **DMA1DCNTIP**：DMA1目标计数中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **DMA1SCNTIP**：DMA1源计数中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-26： IPR3：外设中断优先级寄存器3

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TMR0IP	U1IP	U1EIP	U1TXIP	U1RXIP	I2C1EIP	I2C1IP	I2C1TXIP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **TMR0IP**：TMR0中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **U1IP**：UART1中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **U1EIP**：UART1帧错误中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **U1TXIP**：UART1发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **U1RXIP**：UART1接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **I2C1EIP**：I^2C1错误中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **I2C1IP**：I^2C1中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **I2C1TXIP**：I^2C1发送中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-27： IPR4：外设中断优先级寄存器4

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
INT1IP	CLC1IP	CWG1IP	NCO1IP	CCP1IP	TMR2IP	TMR1GIP	TMR1IP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **INT1IP**：外部中断1中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **CLC1IP**：CLC1中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **CWG1IP**：CWG1中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **NCO1IP**：NCO1中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **CCP1IP**：CCP1中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **TMR2IP**：TMR2中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **TMR1GIP**：TMR1门控中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **TMR1IP**：TMR1中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-28： IPR5：外设中断优先级寄存器5

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
IRXIP	WAKIP	ERRIP	TXB2IP/TXBnIP	TXB1IP	TXB0IP	RXB1IP/RXBnIP	RXB0IP/FIFOFIP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **IRXIP**：收到CAN无效报文中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **WAKIP**：CAN总线唤醒活动中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **ERRIP**：CAN错误中断优先级位（COMSTAT寄存器中有多个中断源）
1 = 高优先级
0 = 低优先级

bit 4 **TXB2IP/TXBnIP**：CAN发送缓冲区2/发送缓冲区n中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **TXB1IP**：CAN发送缓冲区1中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **TXB0IP**：CAN发送缓冲区0中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **RXB1IP/RXBnIP**：CAN接收缓冲区1/接收缓冲区n中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **RXB0IP/FIFOFIP**：CAN接收缓冲区0/FIFO已满中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-29： IPR6：外设中断优先级寄存器6

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
DMA2AIP	DMA2ORIP	DMA2DCNTIP	DMA2SCNTIP	SMT2PWAIP	SMT2PRAIP	SMT2IP	C2IP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **DMA2AIP**：DMA2中止中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **DMA2ORIP**：DMA2溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **DMA2DCNTIP**：DMA2目标计数中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **DMA2SCNTIP**：DMA2源计数中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **SMT2PWAIP**：SMT2脉宽采集中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **SMT2PRAIP**：SMT2周期采集中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **SMT2IP**：SMT2中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **C2IP**：C2中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-30： IPR7：外设中断优先级寄存器7

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
U2IP	U2EIP	U2TXIP	U2RXIP	I2C2EIP	I2C2IP	I2C2TXIP	I2C2RXIP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **U2IP**：UART2中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **U2EIP**：UART2帧错误中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **U2TXIP**：UART2发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **U2RXIP**：UART2接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **I2C2EIP**：I^2C2错误中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **I2C2IP**：I^2C2中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **I2C2TXIP**：I^2C2发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **I2C2RXIP**：TMR4中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-31： IPR8：外设中断优先级寄存器8

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TMR5IP	INT2IP	CLC2IP	CWG2IP	CCP2IP	TMR4IP	TMR3GIP	TMR3IP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **TMR5IP**：TMR5中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **INT2IP**：外部中断2中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **CLC2IP**：CLC2中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **CWG2IP**：CWG2中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **CCP2IP**：CCP2中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **TMR4IP**：TMR4中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **TMR3GIP**：TMR3门控中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **TMR3IP**：TMR3中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-32： IPR9：外设中断优先级寄存器9

U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
—	CLC4IP	CCP4IP	CLC3IP	CWG3IP	CCP3IP	TMR6IP	TMR5GIP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **未实现：**读为0

bit 6 **CLC4IP：**CLC4中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **CCP4IP：**CCP4中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **CLC3IP：**CLC3中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **CWG3IP：**CWG3中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **CCP3IP：**CCP3中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **TMR6IP：**TMR6中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **TMR5GIP：**TMR5中断优先级位
1 = 高优先级
0 = 低优先级

寄存器9-33： IVTBASEU：中断向量表基址最高字节寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	BASE<20:16>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现：**读为0

bit 4-0 **BASE<20:16>：**中断向量表基址位

寄存器9-34： IVTBASEH：中断向量表基址高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
BASE<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **BASE<15:8>：**中断向量表基址位

寄存器9-35： IVTBASEL：中断向量表基址低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-1/1	R/W-0/0	R/W-0/0	R/W-0/0
BASE<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **BASE<7:0>：**中断向量表基址位

寄存器9-36： IVTADU：中断向量表地址最高字节寄存器

U-0	U-0	U-0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
—	—	—	AD<20:16>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现：**读为0

bit 4-0 **AD<20:16>：**中断向量表地址位

寄存器9-37： IVTADH：中断向量表地址高字节寄存器

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
AD<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **AD<15:8>：**中断向量表地址位

寄存器9-38： IVTADL：中断向量表地址低字节寄存器

R-0/0	R-0/0	R-0/0	R-0/0	R-1/1	R-0/0	R-0/0	R-0/0
AD<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **AD<7:0>：**中断向量表地址位

寄存器9-39： IVTLOCK：中断向量表锁定寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	IVTLOCKED(1,2)
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-1 **未实现：**读为0

bit 0 **IVTLOCKED：**IVT寄存器锁定位(1,2)
1 = IVTBASE寄存器锁定，无法写入
0 = IVTBASE寄存器可通过写操作修改

注 1： 只能在例9-1中所示的解锁序列后置1或清零IVTLOCK位。
2： 如果IVT1WAY = 1，则IVTLOCK位无法在置1后清零。请参见寄存器5-3。

寄存器9-40： SHADCON：影子控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	SHADLO
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-1 **未实现：**读为0

bit 0 **SHADLO：**中断影子寄存器访问切换位
0 = 访问中断影子寄存器的主现场
1 = 访问中断影子寄存器的低优先级中断现场

表9-3：　与中断相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
INTCON0	GIE/GIEH	GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	125
INTCON1	STAT<1:0>		—	—	—	—	—	—	126
PIE0	IOCIE	CRCIE	SCANIE	NVMIE	CSWIE	OSFIE	HLVDIE	SWIE	137
PIE1	SMT1PWAIE	SMT1PRAIE	SMT1IE	C1IE	ADTIE	ADIE	ZCDIE	INT0IE	138
PIE2	I2C1RXIE	SPI1IE	SPI1TXIE	SPI1RXIE	DMA1AIE	DMA1ORIE	DMA1DCNTIE	DMA1SCNTIE	139
PIE3	TMR0IE	U1IE	U1EIE	U1TXIE	U1RXIE	I2C1EIE	I2C1IE	I2C1TXIE	140
PIE4	INT1IE	CLC1IE	CWG1IE	NCO1IE	CCP1IE	TMR2IE	TMR1GIE	TMR1IFE	141
PIE5	IRXIE	WAKIE	ERRIE	TXB2IE/TXBnIE	TXB1IE	TXB0IE	RXB1IE/RXBnIE	RXB0IE/FIFOFIE	142
PIE6	DMA2AIE	DMA2ORIE	DMA2DCNTIE	DMA2SCNTIE	SMT2PWAIE	SMT2PRAIE	SMT2IE	C2IE	143
PIE7	U2IE	U2EIE	U2TXIE	U2RXIE	I2C2EIE	I2C2IE	I2C2TXIE	I2C2RXIE	144
PIE8	TMR5IE	INT2IE	CLC2IE	CWG2IE	CCP2IE	TMR4IE	TMR3GIE	TMR3IE	145
PIE9	—	CLC4IE	CCP4IE	CLC3IE	CWG3IE	CCP3IE	TMR6IE	TMR5IE	146
PIR0	IOCIF	CRCIF	SCANIF	NVMIF	CSWIF	OSFIF	HLVDIF	SWIF	127
PIR1	SMT1PWAIF	SMT1PRAIF	SMT1IF	C1IF	ADTIF	ADIF	ZCDIF	INT0IF	128
PIR2	I2C1RXIF	SPI1IF	SPI1TXIF	SPI1RXIF	DMA1AIF	DMA1ORIF	DMA1DCNTIF	DMA1SCNTIF	129
PIR3	TMR0IF	U1IF	U1EIF	U1TXIF	U1RXIF	I2C1EIF	I2C1IF	I2C1TXIF	130
PIR4	INT1IF	CLC1IF	CWG1IF	NCO1IF	CCP1IF	TMR2IF	TMR1GIF	TMR1IF	131
PIR5	IRXIF	WAKIF	ERRIF	TXB2IF/TXBnIF	TXB1IF	TXB0IF	RXB1IF/RXBnIF	RXB0IF/FIFOFIF	132
PIR6	DMA2AIF	DMA2ORIF	DMA2DCNTIF	DMA2SCNTIF	SMT2PWAIF	SMT2PRAIF	SMT2IF	C2IF	133
PIR7	U2IF	U2EIF	U2TXIF	U2RXIF	I2C2EIF	I2C2IF	I2C2TXIF	I2C2RXIF	134
PIR8	TMR5IF	INT2IF	CLC2IF	CWG2IF	CCP2IF	TMR4IF	TMR3GIF	TMR3IF	135
PIR9	—	CLC4IF	CCP4IF	CLC3IF	CWG3IF	CCP3IF	TMR6IF	TMR5IF	136
IPR0	IOCIP	CRCIP	SCANIP	NVMIP	CSWIP	OSFIP	HLVDIP	SWIP	147
IPR1	SMT1PWAIP	SMT1PRAIP	SMT1IP	C1IP	ADTIP	ADIP	ZCDIP	INT0IP	148
IPR2	I2C1RIP	SPI1IP	SPI1TIP	SPI1RIP	DMA1AIP	DMA1ORIP	DMA1DCNTIP	DMA1SCNTIP	149
IPR3	TMR0IP	U1IP	U1EIP	U1TXIP	U1RXIP	I2C1EIP	I2C1IP	I2C1TXIP	150
IPR4	INT1IP	CLC1IP	CWG1IP	NCO1IP	CCP1IP	TMR2IP	TMR1GIP	TMR1IP	151
IPR5	IRXIP	WAKIP	ERRIP	TXB2IP/TXBnIP	TXB1IP	TXB0IP	RXB1IP/RXBnIP	RXB0IP/FIFOFIP	152
IPR6	DMA2AIP	DMA2ORIP	DMA2DCNTIP	DMA2SCNTIP	SMT2PWAIP	SMT2PRAIP	SMT2IP	C2IP	153
IPR7	U2IP	U2EIP	U2TXIP	U2RXIP	I2C2EIP	I2C2IP	I2C2TXIP	I2C2RXIP	154
IPR8	TMR5IP	INT2IP	CLC2IP	CWG2IP	CCP2IP	TMR4IP	TMR3GIP	TMR3IP	155
IPR9	—	CLC4IP	CCP4IP	CLC3IP	CWG3IP	CCP3IP	TMR6IP	TMR5IP	156
IVTBASEU	—	—	—	BASE<20:16>					157
IVTBASEH	BASE<15:8>								157
IVTBASEL	BASE<7:0>								157
IVTADU	—	—	—	AD<20:16>					158
IVTADH	AD<15:8>								158
IVTADL	AD<7:0>								158
IVTLOCK	—	—	—	—	—	—	—	IVTLOCKED	159

图注：　— = 未实现单元，读为0。中断不使用阴影位。

10.0 节能工作模式

掉电模式的目的是降低功耗。有三种掉电模式：

- 打盹模式
- 休眠模式
- 空闲模式

10.1 打盹模式

打盹模式通过减少CPU操作和程序存储器（PFM）访问而不影响外设操作来支持节能。打盹模式与休眠模式的不同之处在于，带隙和系统振荡器继续运行，仅CPU和PFM受影响。通过消除CPU和存储器中的不必要操作来减少操作执行，从而节省功耗。

当打盹使能（DOZEN）位置1（DOZEN = 1）时，CPU每N个周期仅执行一个指令周期，通过CPUDOZE寄存器的DOZE<2:0>位定义。例如，如果DOZE<2:0> = 001，则指令周期比是1:4。CPU和存储器执行一个指令周期，然后在三个指令周期内保持空闲。未使用的周期期间，外设继续以系统时钟速度运行。

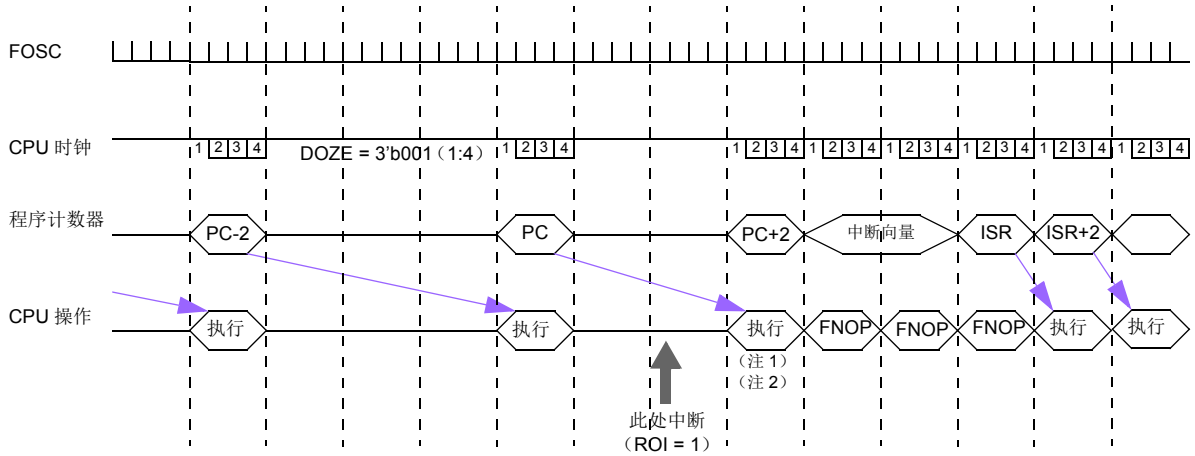
10.1.1 打盹操作

图10-1给出了打盹操作。对于本例：

- 打盹使能（DOZEN）位置1（DOZEN = 1）
- DOZE<2:0> = 001（1:4）比
- 中断恢复（Recover-on-Interrupt，ROI）位置1（ROI = 1）

与正常操作一样，PFM为下一个指令周期取指。至外设的Q时钟继续运行。

图10-1： 打盹模式操作示例（DOZE<2:0> = 001，1:4）



注 1： 在获取中断向量之前，多周期指令执行完成。
2： 如果预取指令清零中断允许位（GIEx），则ISR向量将不会发生，但将清零DOZEN，且CPU将恢复以全速执行。

10.1.2 打盹期间的中断

如果发生中断并且在中断时中断恢复位清零（ROI = 0），中断服务程序（ISR）继续以DOZE<2:0>选择的速率执行。可通过DOZE<2:0>比延长中断延时。

如果发生中断并且在中断时ROI位置1（ROI = 1），DOZEN位清零且CPU以全速执行。执行预取指令，然后执行中断向量序列。在图10-1中，在打盹周期的第2个指令周期发生中断，并立即使CPU退出打盹模式。如果在执行RETFIE操作时退出打盹（Doze-On-Exit，DOE）位置1（DOE = 1），则DOZEN置1，并且CPU基于DOZE<2:0>比降速执行。

例10-1： 打盹软件示例

```
//Mainline operation
bool somethingToDo = FALSE:
void main()
{
    initializeSystem();
            // DOZE = 64:1 (for example)
            // ROI = 1;
    GIE = 1; // enable interrupts
    while (1)
    {
        // If ADC completed, process data
        if (somethingToDo)
        {
            doSomething();
            DOZEN = 1; // resume low-power
        }
    }
}

// Data interrupt handler
void interrupt()
{
    // DOZEN = 0 because ROI = 1
    if (ADIF)
    {
        somethingToDo = TRUE;
        DOE = 0; // make main() go fast
        ADIF = 0;
    }
    // else check other interrupts...
    if (TMR0IF)
    {
        timerTick++;
        DOE = 1; // make main() go slow
        TMR0IF = 0;
    }
}
```

10.2 休眠模式

通过执行SLEEP指令进入休眠模式，而CPUDOZE寄存器的空闲使能（IDLEN）位清零（IDLEN = 0）。

在进入休眠模式时，会存在以下条件：

1. 如果在休眠期间使能WDT，则WDT会清零，但保持运行
2. STATUS寄存器的$\overline{PD}$位清零（寄存器4-2）
3. STATUS寄存器的$\overline{TO}$位置1（寄存器4-2）
4. CPU时钟被禁止
5. LFINTOSC、SOSC、HFINTOSC和ADCRC不受影响且使用这些时钟的外设可在休眠模式下继续运行。
6. I/O端口保持执行SLEEP指令之前的状态（驱动为高电平、低电平或高阻态）
7. WDT之外的复位不受休眠模式影响

关于休眠期间的外设操作的更多详细信息，请参见各个章节。

要最大程度地降低电流消耗，应考虑以下条件：

- I/O引脚不应悬空
- I/O引脚的外部电路灌电流
- I/O引脚的内部电路拉电流
- 从带内部弱上拉的引脚汲取的电流
- 使用任何振荡器的模块

为了避免输入引脚悬空而引入开关电流，应在外部将为高阻抗输入的I/O引脚拉到V_{DD}或V_{SS}。

可能拉电流的内部电路示例包括如DAC和FVR之类的模块。关于这些模块的更多信息，请参见**第38.0节“5位数模转换器（DAC）模块”**和**第35.0节“固定参考电压（FVR）”**。

10.2.1 从休眠模式唤醒

可通过以下任一事件将器件从休眠模式唤醒：

1. $\overline{\text{MCLR}}$引脚上的外部复位输入（如果使能）
2. BOR复位（如果使能）
3. 低功耗欠压复位（LPBOR）（如果使能）
4. POR复位
5. 窗口看门狗定时器（如果使能）
6. 除时钟切换中断外的所有中断源都可以唤醒器件。

前五个事件会导致器件复位。最后一个事件视为继续执行程序。要确定是发生了器件复位还是唤醒事件，请参见**第6.13节“电源控制（PCON0/PCON1）寄存器”**。

当执行SLEEP指令时，下一条指令（PC + 2）被预取出。如果希望通过中断事件唤醒器件，则必须允许相应的中断允许位。唤醒与GIE位的状态无关。如果GIE位被禁止，器件将继续执行SLEEP指令之后的指令。如果GIE位被允许，器件将执行SLEEP指令之后的指令，然后器件将调用中断服务程序。如果不想执行SLEEP指令后的指令，用户应该在SLEEP指令后面放置一条NOP指令。

器件从休眠模式唤醒时，WDT将被清零，而与唤醒源无关。

从休眠事件唤醒后，内核将在开始执行之前等待三个条件的组合。条件为：

- PFM就绪
- COSC选择的振荡器就绪
- BOR就绪（除非禁止BOR）

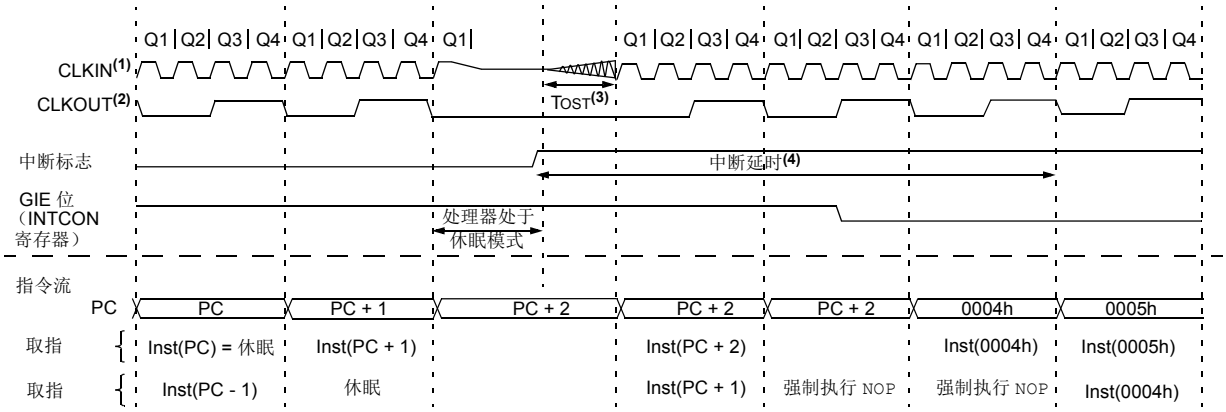
10.2.2 使用中断唤醒

当任一中断源（时钟切换中断除外）的中断允许位和中断标志位都置1时，将会发生以下事件之一：

- 如果在执行SLEEP指令**之前**发生中断
  - SLEEP指令将作为NOP指令执行
  - WDT和WDT预分频器不会被清零
  - STATUS寄存器的$\overline{\text{TO}}$位不会被置1
  - STATUS寄存器的$\overline{\text{PD}}$位不会被清零
- 如果在执行SLEEP指令**期间或之后**发生中断
  - 将完整执行SLEEP指令
  - 器件将立即从休眠模式唤醒
  - WDT和WDT预分频器将被清零
  - STATUS寄存器的$\overline{\text{TO}}$位将被置1
  - STATUS寄存器的$\overline{\text{PD}}$位将被清零

即使在执行SLEEP指令之前检查了标志位，这些标志位也有可能在SLEEP指令执行完毕之前被置1。要确定是否执行了SLEEP指令，可测试$\overline{\text{PD}}$位。如果$\overline{\text{PD}}$位置1，则说明SLEEP指令作为NOP指令执行了。

图10-2： 通过中断从休眠状态唤醒



注 1： 外部时钟。假设采用高、中或低功耗模式。
2： 此处显示的CLKOUT用于时序参考。
3： TOST = 1024 TOSC。此延时不适用于EC和INTOSC振荡器模式。
4： 假设GIE = 1。在这种情形下，处理器被唤醒后，将调用0004h处的ISR。如果GIE = 0，程序将顺序执行。

10.2.3 低功耗休眠模式

PIC18F25/26K83器件系列包含一个内部低压差（Low Dropout，LDO）稳压器，它让器件I/O引脚可以使用最高5.5V的电压工作，而内部器件逻辑可以使用较低的电压工作。在器件处于休眠模式时，LDO及其相关的参考电压电路必须保持活动状态。

PIC18F25/26K83器件允许用户根据应用需求来优化休眠模式下的工作电流。

通过将VREGCON寄存器的VREGPM位置1，可以选择低功耗休眠模式。

10.2.3.1 休眠电流与唤醒时间

在默认工作模式下，处于休眠模式时，LDO和参考电压电路会保持为正常配置。由于所有电路都保持活动状态，器件能够快速地退出休眠模式。在低功耗休眠模式下，从休眠模式中唤醒时，这些电路需要一个额外的延时，然后才会恢复为正常配置并稳定下来。

低功耗休眠模式对于需要长时间处于休眠模式的应用非常有益。正常模式对于需要快速地、频繁地从休眠模式中唤醒的应用非常有益。

10.2.3.2 休眠模式下的外设使用

选择低功耗休眠模式时，一些可以在休眠模式下工作的外设将无法正常工作。低功耗休眠模式旨在用于以下外设：

- 欠压复位（BOR）
- 窗口看门狗定时器（WWDT）
- 外部中断引脚/电平变化中断引脚
- 关闭外部辅助时钟源的外设

最终用户负责确定在进行VREGPM设置以确保休眠模式下的操作时其应用可接受的外设。

注： PIC18LF25/26K83器件不具有可配置的低功耗休眠模式。PIC18LF25/26K83是非稳压器件，它在休眠模式下总是处于最低功耗状态，并且没有唤醒时间延时。这些器件的最大VDD和I/O电压低于PIC18F25/26K83器件。更多信息，请参见**第45.0节“电气规范”**。

10.2.4 空闲模式

当IDLEN置1（IDLEN = 1）时，SLEEP指令将器件置于空闲模式。在空闲模式下，CPU和存储器操作暂停，但外设时钟继续运行。该模式与打盹模式类似，但在空闲模式下CPU和PFM关闭。

注： 如果使能$\overline{\text{CLKOUTEN}}$（$\overline{\text{CLKOUTEN}}$ = 0，配置字1H），输出将在空闲模式下继续运行。

10.2.4.1 空闲和中断

发生中断时退出空闲模式（即使GIE = 0也是如此），但IDLEN不变。器件可通过执行SLEEP指令重新进入空闲模式。

如果同时使能中断恢复（ROI = 1）和打盹模式，则发生中断时器件退出空闲模式并且CPU恢复全速执行。

10.2.4.2 空闲和WWDT

在空闲模式下，WWDT复位被禁止，但会将器件唤醒。WWDT唤醒不是中断，因此ROI不适用。

注： WDT可使器件退出空闲模式，与器件退出休眠模式的方式相同。DOZEN位不受影响。

10.3 节能模式下的外设操作

在空闲和打盹模式下，所有选定的时钟源和关闭它们的外设都有效。仅在休眠模式下，FOSC和FOSC/4时钟均不可用。如果在器件进入休眠模式之前手动或通过外设时钟选择使能，则所有其他时钟源均有效。

10.4 寄存器定义：稳压器控制

寄存器10-1： VREGCON：稳压器控制寄存器(1)

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-1/1
—	—	—	—	—	—	VREGPM	保留
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-2 **未实现：**读为0

bit 1 **VREGPM：**稳压器功耗模式选择位
1 = 休眠时使能低功耗休眠模式(2)
休眠时消耗的电流最低，唤醒速度较慢
0 = 休眠时使能正常功耗模式(2)
休眠时消耗的电流较高，唤醒速度较快

bit 0 **保留：**读为1。该位保持置1。

注 1： LF器件中不存在。
2： 请参见**第45.0节“电气规范”**。

寄存器10-2： CPUDOZE：打盹和空闲寄存器

R/W-0/u	R/W/HC/HS-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
IDLEN	DOZEN	ROI	DOE	—	DOZE<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **IDLEN**：空闲使能位
1 = SLEEP指令禁止CPU时钟，但不禁止外设时钟
0 = SLEEP指令将器件置于完全休眠模式

bit 6 **DOZEN**：打盹使能位(1,2)
1 = CPU根据DOZE设置执行指令周期
0 = CPU执行所有指令周期（最快和最高功耗操作）

bit 5 **ROI**：中断恢复位
1 = 进入中断服务程序（ISR）使DOZEN位 = 0，CPU全速运行
0 = 进入中断不改变DOZEN位状态

bit 4 **DOE**：退出打盹位
1 = 执行RETFIE使DOZEN = 1，CPU降速运行
0 = RETFIE不改变DOZEN位状态

bit 3 **未实现**：读为0

bit 2-0 **DOZE<2:0>**：CPU指令周期与外设指令周期之比
111 =1:256
110 =1:128
101 =1:64
100 =1:32
011 =1:16
010 =1:8
001 =1:4
000 =1:2

注 1： 当ROI = 1或DOE = 1时，通过进入和/或退出硬件中断改变DOZEN位状态。
2： 进入ICD将改写DOZEN，CPU返回到全速执行模式；该位不受影响。

表10-1： 与掉电模式相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
VREGCON(1)	—	—	—	—	—	—	VREGPM	保留	166
CPUDOZE	IDLEN	DOZEN	ROI	DOE	—	DOZE<2:0>			167

图注： — = 未实现位，读为0。掉电模式下不使用阴影单元。
注 1： LF器件中不存在。

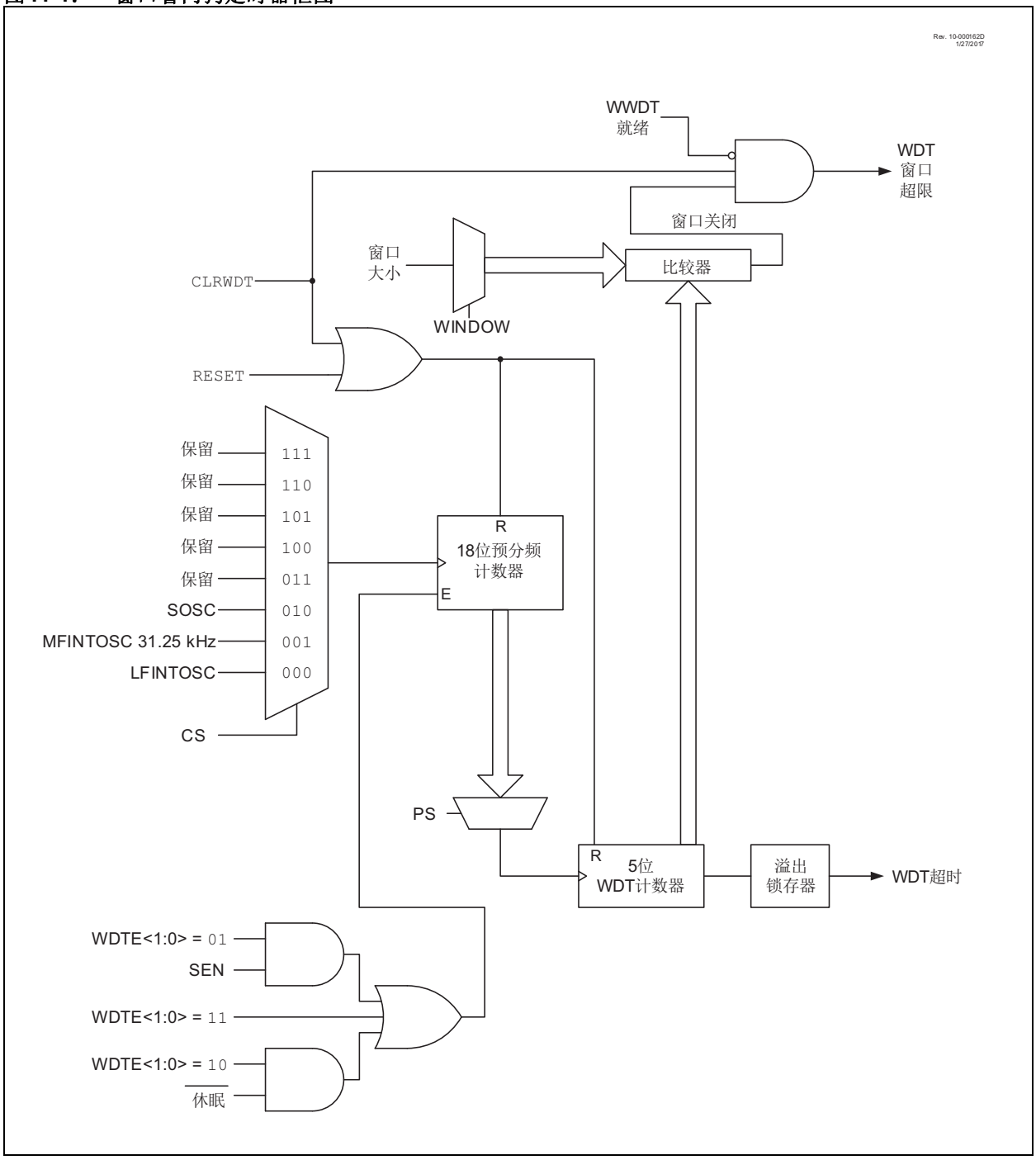
11.0 窗口看门狗定时器（WWDT）

看门狗定时器（WDT）是一个系统定时器，如果固件未在超时周期内发出CLRWDT指令，看门狗定时器会产生复位。看门狗定时器通常用于使系统从意外事件中恢复。窗口看门狗定时器（WWDT）的不同之处在于，只有在超时周期期间的特定窗口内执行CLRWDT指令时才会接受该指令。

WWDT具有以下特性：

- 可选的时钟源
- 多种工作模式
  - WWDT总是开启
  - WWDT在休眠模式下关闭
  - WWDT通过软件进行控制
  - WWDT总是关闭
- 可配置的超时周期为1 ms至256s（标称值）
- 可配置的窗口大小为超时周期的12.5%至100%
- 多种复位条件

图11-1： 窗口看门狗定时器框图



11.1 独立时钟源

WWDT以31 kHz LFINTOSC或31.25 kHz MFINTOSC的内部振荡器作为其时基，具体取决于WDTE<1:0>配置位的值。

如果WDTE = 0b1x，则将根据WDTCCS<2:0>配置位使能时钟源。

如果WDTE = 0b01，则应使用软件将SEN位置1以使能WWDT，并通过WDTCON1寄存器的CS位使能时钟源。

本章中的时间间隔均基于1 ms的最小标称时间间隔。关于LFINTOSC和MFINTOSC容差，请参见**第45.0节“电气规范”**。

11.2 WWDT工作模式

窗口看门狗定时器模块具有4种工作模式，这些工作模式由配置字中的WDTE<1:0>位控制。请参见表11-1。

11.2.1 WWDT总是开启

配置字的WDTE位设置为“11”时，WWDT总是开启。

WWDT保护功能在休眠期间有效。

11.2.2 WWDT在休眠模式下关闭

当配置字的WDTE位设置为10时，除非处于休眠模式，否则WWDT将开启。

WWDT保护在休眠期间无效。

11.2.3 WWDT通过软件进行控制

当配置字的WDTE位设置为01时，WWDT将通过WDTCON0寄存器的SEN位进行控制。

WWDT保护在休眠期间不变。更多详细信息，请参见表11-1。

表11-1： WWDT工作模式

WDTE<1:0>	SEN	器件模式	WWDT模式
11	X	X	有效
10	X	唤醒	有效
		休眠	禁止
01	1	X	有效
	0	X	禁止
00	X	X	禁止

11.3 超时周期

如果WDTCPS<4:0>配置位的默认设置为0b11111，则WDTCON0寄存器的PS位用于设置从1 ms至256s（标称值）的超时周期。如果将除默认值以外的任何值分配给WDTCPS<4:0>配置位，则定时器周期将基于CONFIG3L寄存器中的WDTCPS<4:0>位。复位后，默认的超时周期为2s。

11.4 看门狗窗口

窗口看门狗定时器具有一种可选的窗口模式，由WDTCWS<2:0>配置位和WDTCON1寄存器的WINDOW<2:0>位控制。在窗口模式下，CLRWDT指令必须在WDT周期的容许窗口内发生。在该窗口范围外发生的任何CLRWDT指令都会触发窗口违例，使WWDT复位，这与WWDT超时类似。图11-2给出了一个示例。

窗口大小由WINDOW<2:0>配置位或WDTCON1的WINDOW<2:0>位控制（如果WDTCWS<2:0> = 111）。

WDTTMR寄存器的高5位用于确定窗口是否打开，具体由WDTCON1寄存器的WINDOW<2:0>位定义。

发生窗口超限时，将产生复位，并且PCON0寄存器的WDTWV位将被清零。该位在POR时置1，也可以由固件置1。

11.5 清零WWDT

当发生以下任何条件时，WWDT被清零：

- 任何复位
- 执行了有效的CLRWDT指令
- 器件进入休眠模式
- 中断使器件退出休眠模式
- WWDT被禁止
- 振荡器起振定时器（OST）正在运行
- 对WDTCON0或WDTCON1寄存器执行任意写操作时

11.5.1 CLRWDT注意事项（窗口模式）

在窗口模式下，必须先激活WWDT，之后的CLRWDT指令才会清零定时器。这通过读取WDTCON0寄存器来执行。执行CLRWDT指令而不执行此类配置操作将触发窗口超限，而与窗口是否打开无关。

更多信息，请参见表11-2。

11.6 休眠期间的操作

当器件进入休眠模式时，WWDT会被清零。如果使能WWDT在休眠期间工作，WWDT会继续计数。当器件退出休眠模式时，WWDT会被再次清零。

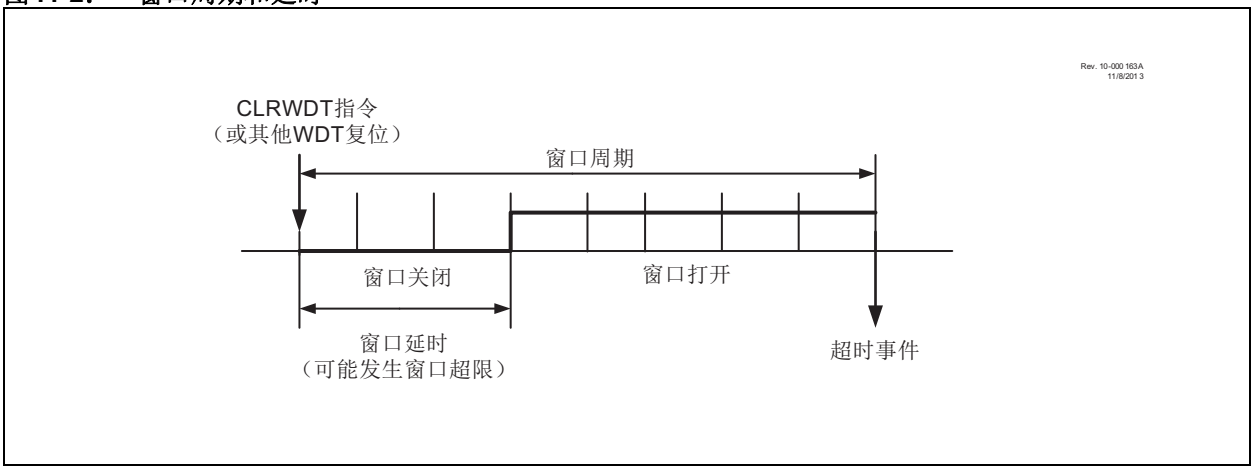
WWDT保持清零，直到振荡器起振定时器（OST）延时结束为止（如果使能）。关于OST的更多信息，请参见**第7.2.1.3节“振荡器起振定时器（OST）”**。

在器件处于休眠模式的情况下发生WWDT超时时，不会产生复位。器件将会唤醒并继续工作。STATUS寄存器中的TO和PD位会发生改变，指示发生的事件。也可以使用PCON0寄存器中的RWDT位。更多信息，请参见**第4.0节“存储器构成”**。

表11-2： WWDT清零条件

条件	WWDT
WDTE<1:0> = 00	清零
WDTE<1:0> = 01且SEN = 0	
WDTE<1:0> = 10且进入休眠模式	
CLRWDT命令	
检测到振荡器故障	
退出休眠 + 系统时钟 = SOSC、EXTRC、INTOSC或EXTCLK	
退出休眠 + 系统时钟 = XT、HS或LP	清零，直到OST延时结束
更改INTOSC分频比（IRCF位）	不受影响

图11-2： 窗口周期和延时



11.7 寄存器定义：窗口看门狗定时器控制

寄存器11-1： WDTCON0：看门狗定时器控制寄存器0

U-0	U-0	R/W(3)-q/q(2)	R/W(3)-q/q(2)	R/W(3)-q/q(2)	R/W(3)-q/q(2)	R/W(3)-q/q(2)	R/W-0/0
—	—	PS<4:0>					SEN
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-6 **未实现：**读为0

bit 5-1 **PS<4:0>：**看门狗定时器预分频比选择位(1)

位值 = 预分频比

11111 = 保留。产生最小的时间间隔（1:32）

•

•

•

10011 = 保留。产生最小的时间间隔（1:32）

10010 = 1:8388608（2^{23}）（时间间隔标称值为256s）
10001 = 1:4194304（2^{22}）（时间间隔标称值为128s）
10000 = 1:2097152（2^{21}）（时间间隔标称值为64s）
01111 = 1:1048576（2^{20}）（时间间隔标称值为32s）
01110 = 1:524288（2^{19}）（时间间隔标称值为16s）
01101 = 1:262144（2^{18}）（时间间隔标称值为8s）
01100 = 1:131072（2^{17}）（时间间隔标称值为4s）
01011 = 1:65536（时间间隔标称值为2s）（复位值）
01010 = 1:32768（时间间隔标称值为1s）
01001 = 1:16384（时间间隔标称值为512 ms）
01000 = 1:8192（时间间隔标称值为256 ms）
00111 = 1:4096（时间间隔标称值为128 ms）
00110 = 1:2048（时间间隔标称值为64 ms）
00101 = 1:1024（时间间隔标称值为32 ms）
00100 = 1:512（时间间隔标称值为16 ms）
00011 = 1:256（时间间隔标称值为8 ms）
00010 = 1:128（时间间隔标称值为4 ms）
00001 = 1:64（时间间隔标称值为2 ms）
00000 = 1:32（时间间隔标称值为1 ms）

bit 0 **SEN：**看门狗定时器软件使能/禁止位

如果WDTE<1:0> = 1x：
忽略此位。
如果WDTE<1:0> = 01：
1 = WDT开启
0 = WDT关闭
如果WDTE<1:0> = 00：
忽略此位。

注 1： 时间均为近似值。WDT时间基于31 kHz LFINTOSC。

2： 当CONFIG3L中的WDTCPS<4:0> = 11111时，PS<4:0>的复位值为01011。否则，PS<4:0>的复位值等于CONFIG3L中的WDTCPS<4:0>。

3： 当CONFIG3L中的WDTCPS<4:0> ≠ 11111时，这些位是只读的。

4： 当WWDT配置为使用SOSC作为时钟源运行，并且允许器件经历由WDT超时触发的复位时，SOSC也将经历复位。这意味着SOSC将执行其启动序列，该序列需要1024个SOSC时钟计数才能供外设使用。因此，例如，如果WDT设置为1 ms超时，并且允许器件经历WDT复位，那么实际的WDT复位周期将为：WDT_PERIOD = (1/(SOSC_FREQUENCY) * 1024) + 1 ms。

寄存器11-2： WDTCON1：看门狗定时器控制寄存器1

U-0	R/W(3)-q/q(1) R/W(3)-q/q(1) R/W(3)-q/q(1)	U-0	R/W(4)-q/q(2) R/W(4)-q/q(2) R/W(4)-q/q(2)
—	CS<2:0>	—	WINDOW<2:0>
bit 7			bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **未实现：**读为0

bit 6-4 **CS<2:0>：**看门狗定时器时钟选择位

111 = 保留
•
•
•
011 = 保留
010 = SOSC
001 = MFINTOSC 31.25 kHz
000 = LFINTOSC 31 kHz

bit 3 **未实现：**读为0

bit 2-0 **WINDOW<2:0>：**看门狗定时器窗口选择位

WINDOW<2:0>	窗口延时时间百分比	窗口打开时间百分比
111	N/A	100
110	12.5	87.5
101	25	75
100	37.5	62.5
011	50	50
010	62.5	37.5
001	75	25
000	87.5	12.5

注 1： 如果CONFIG3H中的WDTCCS<2:0> = 111，则CS<2:0>的复位值为000。
2： WINDOW<2:0>的复位值由CONFIG3H寄存器中WDTCWS<2:0>的值决定。
3： 如果CONFIG3H中的WDTCCS<2:0> ≠ 111，则这些位是只读的。
4： 如果CONFIG3H中的WDTCWS<2:0> ≠ 111，则这些位是只读的。

寄存器11-3： WDTPSL：WWDT预分频比选择低字节寄存器（只读）

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
PSCNT<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **PSCNT<7:0>**：预分频比选择低字节位(1)

注 1： 18位WDT预分频值PSCNT<17:0>包括WDTPSL、WDTPSH和WDTTMR寄存器的低位。PSCNT<17:0>用于调试操作，不应在正常操作过程中读取。

寄存器11-4： WDTPSH：WWDT预分频比选择高字节寄存器（只读）

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
PSCNT<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **PSCNT<15:8>**：预分频比选择高字节位(1)

注 1： 18位WDT预分频值PSCNT<17:0>包括WDTPSL、WDTPSH和WDTTMR寄存器的低位。PSCNT<17:0>用于调试操作，不应在正常操作过程中读取。

寄存器11-5： WDTTMR：WDT定时器寄存器（只读）

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
WDTTMR<4:0>					STATE	PSCNT<17:16>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-3 **WDTTMR<4:0>：**看门狗窗口值位

WINDOW	WDT窗口状态		打开时间百分比
	关闭	打开	
111	N/A	00000-11111	100
110	00000-00011	00100-11111	87.5
101	00000-00111	01000-11111	75
100	00000-01011	01100-11111	62.5
011	00000-01111	10000-11111	50
010	00000-10011	10100-11111	37.5
001	00000-10111	11000-11111	25
000	00000-11011	11100-11111	12.5

bit 2 **状态：**WDT就绪状态位
1 = WDT已就绪
0 = WDT未就绪

bit 1-0 **PSCNT<17:16>：**预分频比选择最高字节位(1)

注 1： 18位WDT预分频值PSCNT<17:0>包括WDTPSL、WDTPSH和WDTTMR寄存器的低位。PSCNT<17:0>用于调试操作，不应在正常操作过程中读取。

表11-3：　与窗口看门狗定时器相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
WDTCON0	—	—	PS<4:0>					SEN	172
WDTCON1	—	CS<2:0>			—	WINDOW<2:0>			173
WDTPSL	PSCNT<7:0>								174
WDTPSH	PSCNT<15:8>								174
WDTTMR	WDTTMR<4:0>					STATE	PSCNT<17:16>		175

图注： x = 未知，u = 不变，— = 未实现位，读为0。窗口看门狗定时器不使用阴影单元。

12.0 8x8硬件乘法器

12.1 简介

所有PIC18器件都包含一个8x8硬件乘法器（作为ALU的一部分）。该乘法器执行无符号运算并产生一个16位结果，该结果存储在产品寄存器对（PRODH:PRODL）中。乘法器的运算不会影响STATUS寄存器中的任何标志。

通过硬件操作执行乘法运算可在单个指令周期内完成。其优势在于可提高计算吞吐量和缩短乘法算法的代码长度，从而使PIC18器件能够用于许多之前必须采用数字信号处理器的应用。表12-1对比了各种硬件和软件乘法运算及其节省的存储空间和执行时间。

12.2 运算

例12-1给出了8x8无符号乘法的指令序列。当WREG寄存器中已装载其中一个参数时，只需要一条指令。

例12-2给出了执行8x8有符号乘法的序列。为了确定参数的符号位，将测试每个参数的最高有效位（Most Significant bit，MSb）并做适当的减法。

例12-1： 8x8无符号乘法程序

```
MOVF   ARG1, W   ;
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
```

例12-2： 8x8有符号乘法程序

```
MOVF   ARG1, W
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
BTFSC  ARG2, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG1
MOVF   ARG2, W
BTFSC  ARG1, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;         - ARG2
```

表12-1： 各种乘法运算的性能比较

程序	乘法方法	程序存储器（字）	周期（最大值）	时间			
				@ 64 MHz	@ 40 MHz	@ 10 MHz	@ 4 MHz
8x8无符号	无硬件乘法	13	69	4.3 µs	6.9 µs	27.6 µs	69 µs
	硬件乘法	1	1	62.5 ns	100 ns	400 ns	1 µs
8x8有符号	无硬件乘法	33	91	5.7 µs	9.1 µs	36.4 µs	91 µs
	硬件乘法	6	6	375 ns	600 ns	2.4 µs	6 µs
16x16无符号	无硬件乘法	21	242	15.1 µs	24.2 µs	96.8 µs	242 µs
	硬件乘法	28	28	1.8 µs	2.8 µs	11.2 µs	28 µs
16x16有符号	无硬件乘法	52	254	15.9 µs	25.4 µs	102.6 µs	254 µs
	硬件乘法	35	40	2.5 µs	4.0 µs	16.0 µs	40 µs

例12-3给出了执行16 x 16无符号乘法的序列。公式12-1给出了使用的算法。32位结果存储在四个寄存器（RES<3:0>）中。

公式12-1： 16 x 16无符号乘法算法

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

例12-3： 16 x 16无符号乘法程序

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
```

例12-4给出了执行16 x 16有符号乘法的序列。公式12-2给出了使用的算法。32位结果存储在四个寄存器（RES<3:0>）中。为了确定参数的符号位，将测试每个参数对的MSb并做适当的减法。

公式12-2： 16 x 16有符号乘法算法

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H\langle 7\rangle \bullet ARG1H:ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H\langle 7\rangle \bullet ARG2H:ARG2L \bullet 2^{16})
\end{aligned}
$$

例12-4： 16 x 16有符号乘法程序

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L ->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H ->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H ->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L ->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    BTFSS   ARG2H, 7        ; ARG2H:ARG2L neg?
    BRA     SIGN_ARG1       ; no, check ARG1
    MOVF    ARG1L, W        ;
    SUBWF   RES2            ;
    MOVF    ARG1H, W        ;
    SUBWFB  RES3
;
SIGN_ARG1
    BTFSS   ARG1H, 7        ; ARG1H:ARG1L neg?
    BRA     CONT_CODE       ; no, done
    MOVF    ARG2L, W        ;
    SUBWF   RES2            ;
    MOVF    ARG2H, W        ;
    SUBWFB  RES3
;
CONT_CODE
    :
```

13.0 非易失性存储器（NVM）控制

非易失性存储器（Nonvolatile Memory，NVM）分为两种类型：闪存程序存储器（Program Flash Memory，PFM）和数据EEPROM存储器。

PFM、数据EEPROM、用户ID和配置位均可通过NVMCON1寄存器的REG<1:0>位进行访问。

写入时间由片上定时器控制。写入/擦除电压是由片上电荷泵产生的，此电荷泵在器件的工作电压范围内工作。

NVM可通过两种方式进行保护，代码保护或写保护。代码保护（配置字5L中的CP和CPD位）会禁止通过外部器件编程器对PFM和数据EEPROM存储器进行读写访问。代码保护不会影响自写和擦除功能。代码保护只能通过器件编程器对器件执行批量擦除操作，从而清除所有非易失性存储器、配置位和用户ID而复位。

写保护会禁止对由配置字4H的WRT位所定义的部分或全部PFM进行自写和擦除操作。写保护不会影响器件编程器对器件进行读、写或擦除操作。

表13-1： NVM构成和访问信息

存储器	PC<20:0> ICSP™ Addr<21:0> TBLPTR<21:0> NVMADDR<9:0>	执行	用户访问		
		CPU执行	REG	TABLAT	NVMDAT
闪存程序存储器（PFM）	00 0000h ••• 01 FFFFh	读	10	读/写(1)	—(3)
用户ID(2)	20 0000h ••• 20 000Fh	无访问	x1	读/写	—(3)
保留	20 0010h 2F FFFFh	无访问	—(3)		
配置	30 0000h ••• 30 0009h	无访问	x1	读/写(1)	—(3)
保留	30 000Ah 30 FFFFh	无访问	—(3)		
用户数据存储器（数据EEPROM）	31 0000h ••• 31 03FFh	无访问	00	—(3)	读/写(1)
保留	31 0400h 3E FFFFh	无访问	—(3)		
器件信息区（DIA）	3F 0000h ••• 3F 003Fh	无访问	x1	读	—(3)
保留	3F 0040h 3F FF09h	无访问	—(3)		
器件配置信息（DCI）	3F FF00h ••• 3F FF09h	无访问	x1	读	—(3)
保留	3F FF0Ah 3F FFFBh	无访问	—(3)		
版本ID/器件ID	3F FFFCh ••• 3F FFFFh	无访问	x1	读	—(3)

注 1： 受存储器写保护设置的影响。
2： 用户ID仅为8个字。该区域未实现代码保护、表读保护或写保护。
3： 读为0，写操作会清零WR位且WRERR位置1。

13.1 闪存程序存储器

在整个VDD范围内的正常工作期间，闪存程序存储器都是可读写、可擦除的。

读取程序存储器时，一次读取一个字节。写入或擦除程序存储器时，一次写入或擦除*n*字节的块。有关写块和擦除块的大小，请参见表5-4。无法通过用户代码发起批量擦除操作。

写入或擦除程序存储器时将停止取指操作，直到完成写入或擦除操作。在写入或擦除期间无法访问程序存储器，因此无法执行代码。内部编程定时器会终止对程序存储器的写操作和擦除操作。

写入程序存储器的值不必为有效指令。如果执行的程序存储单元为无效指令，则会产生一个NOP指令。

对于擦除和编程操作，了解PFM存储器结构非常重要。程序存储字大小为16位宽。PFM按行排列。行是可以通过用户软件擦除的最小大小。有关这些器件的行大小，请参见表5-4。

擦除某行后，可对该行的全部或部分内容进行编程。要写入程序存储器行的数据通过6条地址线写入8位宽的数据写锁存器中。不能直接访问这些锁存器，但可以通过连续写入TABLAT寄存器来装入数据。

注：	如果只修改先前已编程行的一部分内容，则必须在擦除之前先读取整行内容，并保存到RAM中。然后，可以将新数据和已保存数据写入写锁存器，以对PFM行进行再编程。但对于任何未经过编程的存储单元，则无需先擦除行即可写入。这种情况下，不需要保存并重新写入其他先前已编程的存储单元。

表13-2： 各器件闪存构成

器件	行擦除大小（字）	写锁存器（字节）	闪存程序存储器（字）	数据存储器（字节）
PIC18(L)F25K83	64	128	16384	1024
PIC18(L)F26K83	64	128	32768	1024

13.1.1 表读和表写

为了读写程序存储器，有两种操作允许处理器在程序存储空间和数据RAM之间移动字节：

- 表读（TBLRD）
- 表写（TBLWT）

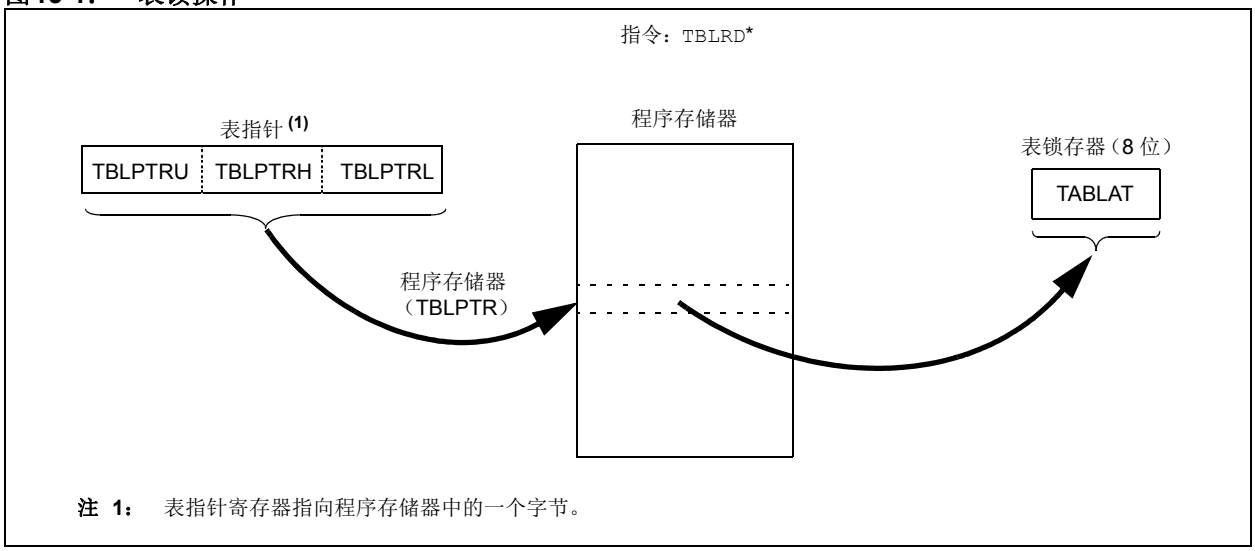
程序存储空间为16位宽，而数据RAM空间为8位宽。表读和表写通过8位寄存器（TABLAT）在这两个存储空间之间移动数据。

表读操作直接从程序存储器中获取一个数据字节，并将其放入TABLAT寄存器中。图13-1显示了表读操作。

表写操作将TABLAT寄存器中的一个数据字节存储到写块保持寄存器中。**第13.1.6节“写入闪存程序存储器”**详述了将保持寄存器内容写入程序存储器的过程。图13-2显示了关于程序存储器和数据RAM的表写操作。

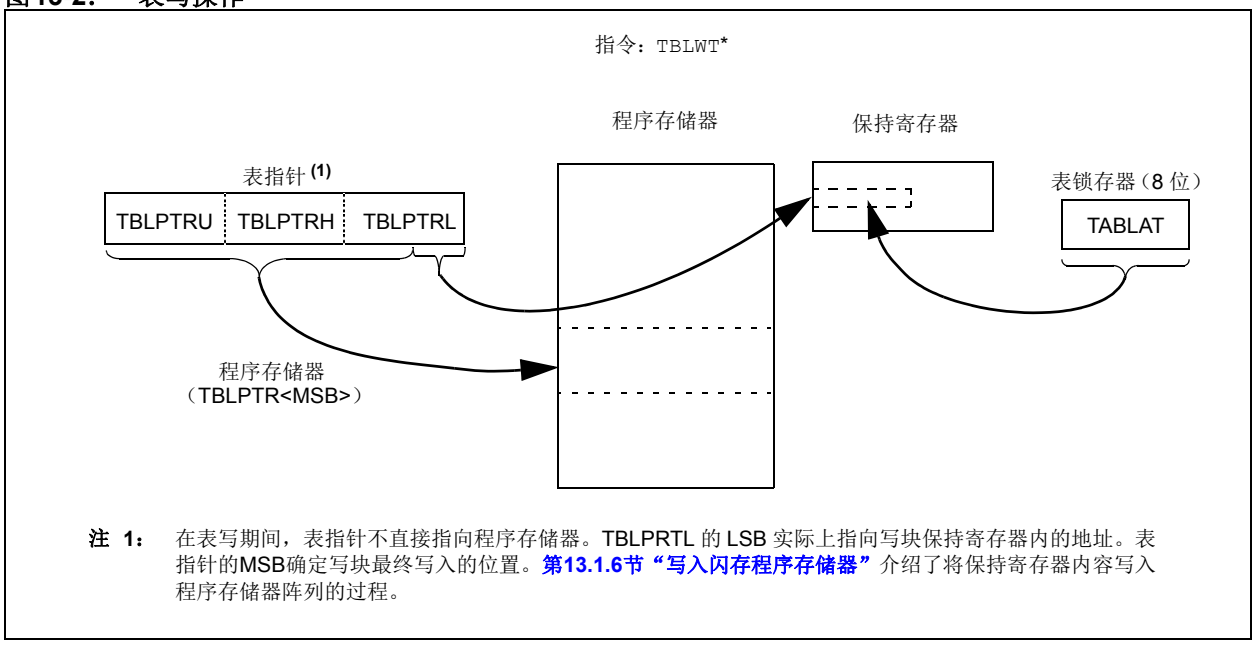
表操作的对象为字节实体。如果表中包含数据（而非程序指令），则不需要按字对齐。因此，表可以在任何字节地址处开始和结束。如果表写用于将可执行代码写入程序存储器，则程序指令需要按字对齐。

图13-1： 表读操作



注 1： 表指针寄存器指向程序存储器中的一个字节。

图13-2： 表写操作



注 1： 在表写期间，表指针不直接指向程序存储器。TBLPRTL 的 LSB 实际上指向写块保持寄存器内的地址。表指针的MSB确定写块最终写入的位置。**第13.1.6节“写入闪存程序存储器”**介绍了将保持寄存器内容写入程序存储器阵列的过程。

13.1.2 控制寄存器

TBLRD和TBLWT指令与多个控制寄存器搭配使用。这些寄存器具体如下：

- NVMCON1寄存器
- NVMCON2寄存器
- TABLAT寄存器
- TBLPTR寄存器

13.1.2.1 NVMCON1和NVMCON2寄存器

NVMCON1寄存器（寄存器13-1）是关于存储器访问的控制寄存器。NVMCON2寄存器不是物理寄存器，它专用于存储器写序列和擦除序列。读NVMCON2时得到的结果为全0。

REG<1:0>控制位确定访问的是数据EEPROM存储单元，PFM单元还是用户ID、配置位、版本ID和器件ID。当REG<1:0> = 00时，任何后续操作都将在数据EEPROM存储器上进行。当REG<1:0> = 10时，任何后续操作都将在程序存储器上进行。当REG<1:0> = x1时，任何后续操作都将在配置位、用户ID、版本ID和器件ID上进行。

FREE位用于允许对程序存储器进行擦除操作。如果FREE位置1，在收到下一条WR命令时启动擦除操作。如果FREE位清零，仅使能写操作。该位仅适用于PFM，不适用于数据EEPROM。

WREN位置1时允许编程/擦除操作。WREN位在上电时清零。

WRERR位在WR位置1时由硬件置1，而在内部编程定时器超时和写操作成功完成时清零。

WR控制位在REG<1:0>位指向数据EEPROM存储单元时启动擦除/写周期操作，而在REG<1:0>位指向PFM单元时启动写操作。WR位无法由固件清零，只能由固件置1。WR位会在写操作完成后由硬件清零。

当写操作完成时，NVMIF中断标志位置1。NVMIF标志位在由固件清零之前将保持置1。

13.1.2.2 TABLAT——表锁存器寄存器

表锁存器（TABLAT）是一个映射到SFR空间的8位寄存器。表锁存器寄存器用于在程序存储器和数据RAM之间的数据传输期间保存8位数据。

13.1.2.3 TBLPTR——表指针寄存器

表指针（TBLPTR）寄存器用于寻址程序存储器内的字节。TBLPTR由三个SFR寄存器组成：表指针最高字节、表指针高字节和表指针低字节（TBLPTRU:TBLPTRH:TBLPTRL）。这三个寄存器一起构成一个22位宽的指针。低21位允许器件寻址高达2 MB的程序存储空间。第22位允许访问器件ID、用户ID和配置位。

表指针寄存器（TBLPTR）由TBLRD和TBLWT指令使用。这些指令可根据表操作以四种方式之一更新TBLPTR。在TBLPTR上执行这些操作仅影响低21位。

13.1.2.4 表指针边界

TBLPTR用于闪存程序存储器的读写和擦除操作。

执行TBLRD时，TBLPTR的所有22位确定将程序存储器中的哪个字节直接读入TABLAT寄存器。

执行TBLWT时，TABLAT寄存器中的字节不会写入存储器，而会写入保持寄存器，为程序存储器写操作做准备。保持寄存器构成一个写块，该写块因器件而异（见表13-3）。TBLPTRL寄存器的其中3、4或5个LSb确定将写入保持寄存器块内的哪个特定地址。表指针的MSB在TBLWT操作期间不起作用。

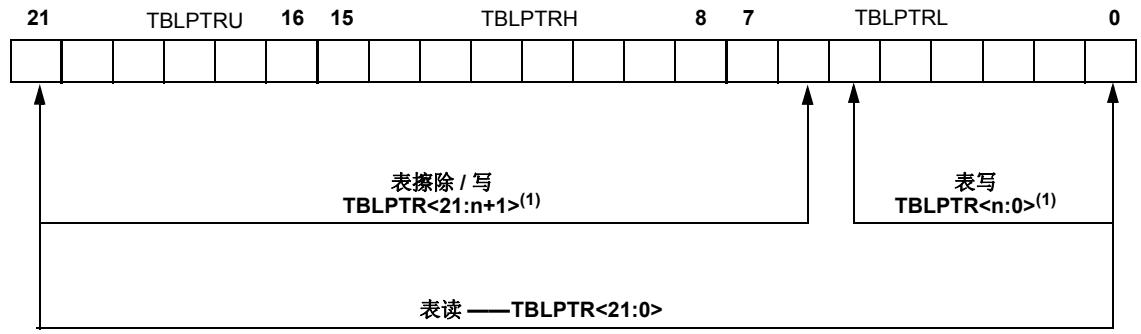
执行程序存储器写操作时，将整个保持寄存器块写入存储器中由TBLPTR的MSb确定的地址处。在存储器写操作期间，将忽略3、4或5个LSB。更多详细信息，请参见**第13.1.6节“写入闪存程序存储器”**。

图13-3给出了不同闪存程序存储器操作的TBLPTR相关边界。

表13-3： 关于TBLRD和TBLWT指令的表指针操作

示例	表指针操作
TBLRD* TBLWT*	TBLPTR未修改
TBLRD*+ TBLWT*+	TBLPTR在读/写操作之后递增
TBLRD*- TBLWT*-	TBLPTR在读/写操作之后递减
TBLRD+* TBLWT+*	TBLPTR在读/写操作之前递增

图13-3： 不同操作的表指针边界



注 1： 关于行大小值，请参见表5-4。

13.1.3 读取闪存程序存储器

TBLRD指令从程序存储器中获取数据，并将其放入数据RAM中。对程序存储器执行表读操作时，一次读取一个字节。

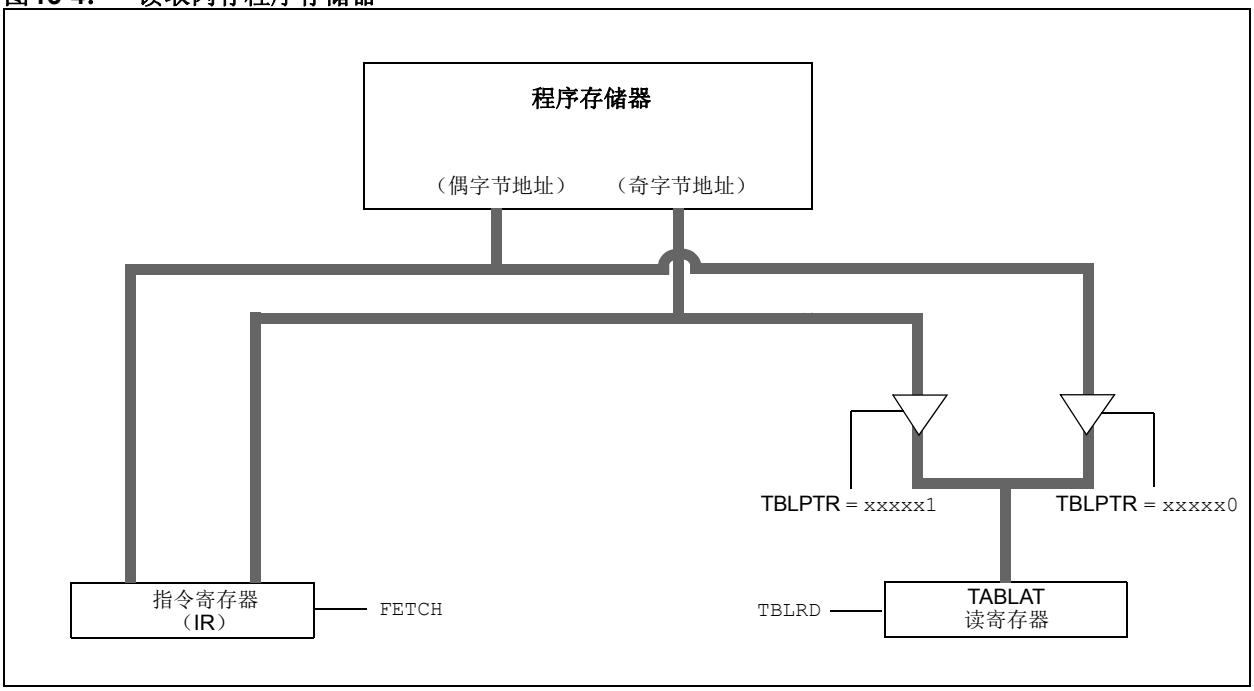
TBLPTR指向程序空间中的字节地址。执行TBLRD时会将指向的字节放入TABLAT中。此外，可以自动修改TBLPTR以进行下一次表读操作。

CPU操作在读操作期间暂停，并在完成之后立即恢复。从用户的角度来看，TABLAT在下一个指令周期生效。

内部程序存储器通常按字划分。地址的最低有效位用于选择字的高字节或低字节。

图13-4给出了内部程序存储器和TABLAT之间的接口。

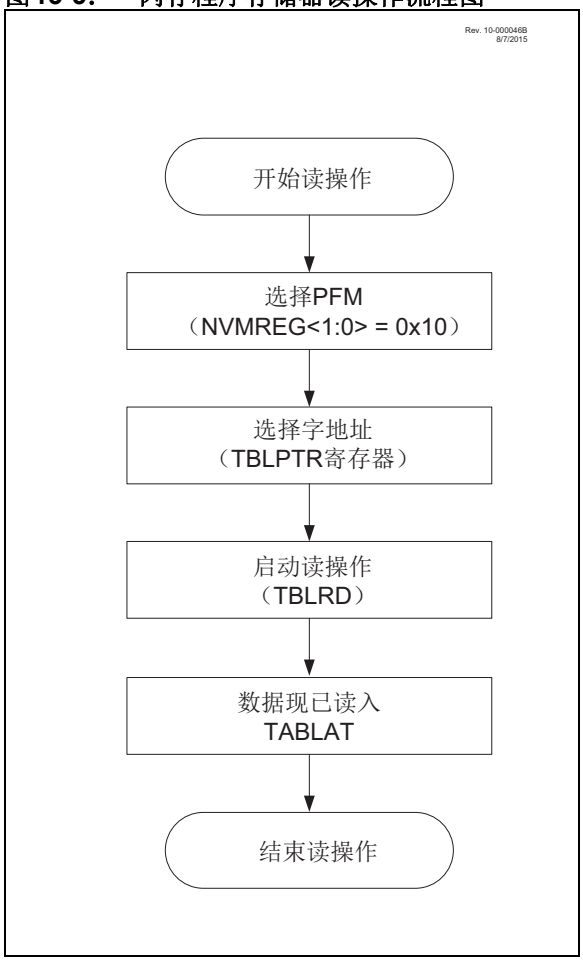
图13-4： 读取闪存程序存储器



例13-1： 读取闪存程序存储字

```
            BCF       NVMCON1, REG0          ; point to Program Flash Memory
            BSF       NVMCON1, REG1          ; access Program Flash Memory
            MOVLW     CODE_ADDR_UPPER        ; Load TBLPTR with the base
            MOVWF     TBLPTRU                ; address of the word
            MOVLW     CODE_ADDR_HIGH
            MOVWF     TBLPTRH
            MOVLW     CODE_ADDR_LOW
            MOVWF     TBLPTRL
READ_WORD
            TBLRD*+                          ; read into TABLAT and increment
            MOVF      TABLAT, W              ; get data
            MOVWF     WORD_EVEN
            TBLRD*+                          ; read into TABLAT and increment
            MOVFW     TABLAT, W              ; get data
            MOVF      WORD_ODD
```

图13-5： 闪存程序存储器读操作流程图



13.1.4 NVM解锁序列

解锁序列是一种用于保护NVM免于发生意外自写编程或擦除的机制。只有在无中断情况下执行并完成序列时，才能成功地完成以下操作之一：

- PFM行擦除
- 将PFM写锁存器内容写入PFM存储器
- 将PFM写锁存器内容写入用户ID
- 写入数据EEPROM存储器
- 写入配置字

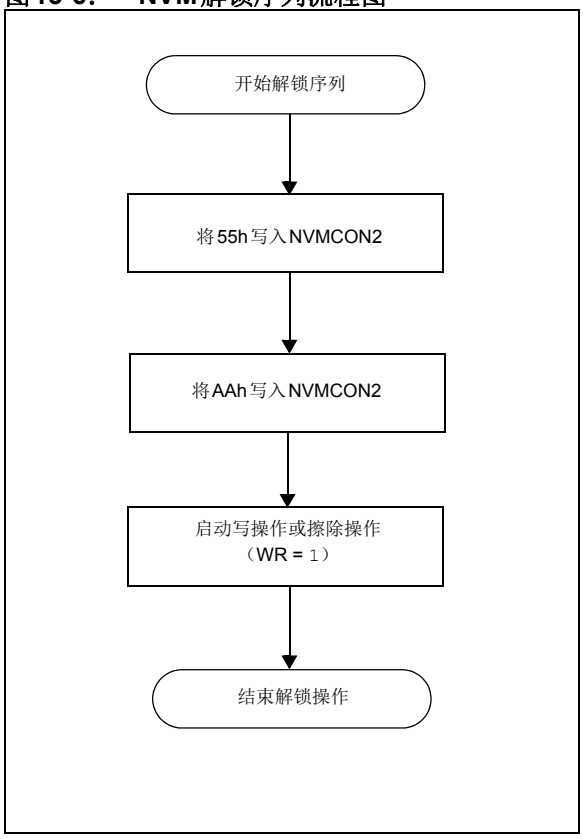
解锁序列由以下步骤组成且必须按照以下顺序完成：

- 将55h写入NVMCON2
- 将AAh写入NMVCON2
- 将NVMCON1的WR位置1

在WR位置1后，处理器会暂停内部操作，直到操作完成为止，然后再继续执行下一条指令。

由于在执行解锁序列的过程中不能发生中断，所以在执行解锁序列之前应先禁止全局中断，然后在完成解锁序列之后重新允许中断。

图13-6： NVM解锁序列流程图



例13-2： NVM解锁序列

```
BCF             INTCON0,GIE             ; Recommended so sequence is not interrupted
BANKSEL         NVMCON1
BSF             NVMCON1,WREN            ; Enable write/erase
MOVLW           55h                     ; Load 55h

MOVWF           NVMCON2                 ; Step 1: Load 55h into NVMCON2
MOVLW           AAh                     ; Step 2: Load W with AAh
MOVWF           NVMCON2                 ; Step 3: Load AAh into NVMCON2
BSF             INTCON1,WR              ; Step 4: Set WR bit to begin write/erase

BSF             INTCON0,GIE             ; Re-enable interrupts
```

注 1： 解锁序列开始于写NVMCON2；必须按所示的精确周期顺序执行步骤1-4。如果步骤1至4的时序因中断或调试器暂停而遭到破坏，则不会执行操作。

2： 操作码仅用作说明；任何具有所指示作用的指令均可使用。

13.1.5 擦除闪存程序存储器

最小擦除块为64字（见表5-4）。只有通过使用外部编程器或通过ICSP™控制，才能批量擦除更大的程序存储器块。程序存储器阵列中不支持字擦除。

例如，当从单片机中启动擦除行大小为64字的擦除序列时，将会擦除64字（128字节）的程序存储器块。TBLPTR<21:6>的高16位指向正在擦除的块。TBLPTR<5:0>位将被忽略。

NVMCON1寄存器用于发出擦除命令。REG<1:0>位必须设置为指向闪存程序存储器。WREN位必须置1才能使能写操作。FREE位置1将选择擦除操作。

应采用**第13.1.4节“NVM解锁序列”**所述的NVM解锁序列来防止发生意外写操作。这一序列有时称为长写。

擦除程序存储器时需要执行长写操作。在长写周期内，将暂停执行指令。长写操作由内部编程定时器终止。

13.1.5.1 闪存程序存储器擦除序列

用于擦除内部程序存储器块的事件序列如下：

1. NVMCON1寄存器的REG位指向PFM
2. 将NVMCON1寄存器的FREE和WREN位置1
3. 按**第13.1.4节“NVM解锁序列”**所述执行解锁序列

如果PFM地址受写保护，则清零WR位且不会执行擦除操作，这种情况下会通过WRERR指示错误状态。

该操作会擦除通过屏蔽当前TBLPTR的LSB指示的存储器行。

擦除PFM时，CPU操作暂停，并在操作完成时恢复。操作完成时，WR位由硬件清零，NVMIF置1，并且在NVMIE位也置1时将发生中断。

写锁存器数据不受擦除操作影响，且WREN将保持不变。

注 1：	如果写操作或擦除操作因意外事件而终止，则WRERR位将置1，用户可通过查看该位以决定是否需要重写存储单元。
2：	如果在TBLPTR指向写保护地址时WR被写入1，则WRERR置1。
3：	如果在TBLPTR指向无效地址单元时WR被写入1，则WRERR置1（表13-1）。

例13-3： 擦除闪存程序存储器块

```
; This sample row erase routine assumes the following:
; 1. A valid address within the erase row is loaded in variables TBLPTR register

            CLRF        NVMCON1             ; Setup PFM Access
            MOVLW       CODE_ADDR_UPPER     ; load TBLPTR with the base
            MOVWF       TBLPTRU             ; address of the memory block
            MOVLW       CODE_ADDR_HIGH
            MOVWF       TBLPTRH
            MOVLW       CODE_ADDR_LOW
            MOVWF       TBLPTRL
ERASE_BLOCK
            BCF         NVMCON1, REG0       ; point to Program Flash Memory
            BSF         NVMCON1, REG1       ; access Program Flash Memory
            BSF         NVMCON1, WREN       ; enable write to memory
            BSF         NVMCON1, FREE       ; enable block Erase operation
            BCF         INTCON0, GIE        ; disable interrupts
            MOVLW       55h
所需序列    MOVWF       NVMCON2             ; write 55h
            MOVLW       AAh
            MOVWF       NVMCON2             ; write AAh
            BSF         NVMCON1, WR         ; start erase (CPU stalls)
            BSF         INTCON0, GIE        ; re-enable interrupts
```

图13-7： PFM行擦除流程图

13.1.6 写入闪存程序存储器

表5-4介绍了编程写块大小。不支持字或字节编程。表写操作在内部用于装载编程存储器所需的保持寄存器内容。保持寄存器的数量与一个写块中的字节数量相同。关于写锁存器的大小，请参见表5-4。

由于表锁存器（TABLAT）仅有一个字节，因此每次编程操作时都需要多次执行TBLWT指令。此操作将忽略写保护状态。所有表写操作基本上都是短写操作，因为仅写入保持寄存器。写入保持寄存器时，NVMIF不受影响。

写入所有保持寄存器后，通过配置NVMCON1寄存器执行程序存储器写操作并执行长写序列来启动对该存储器块的编程操作。

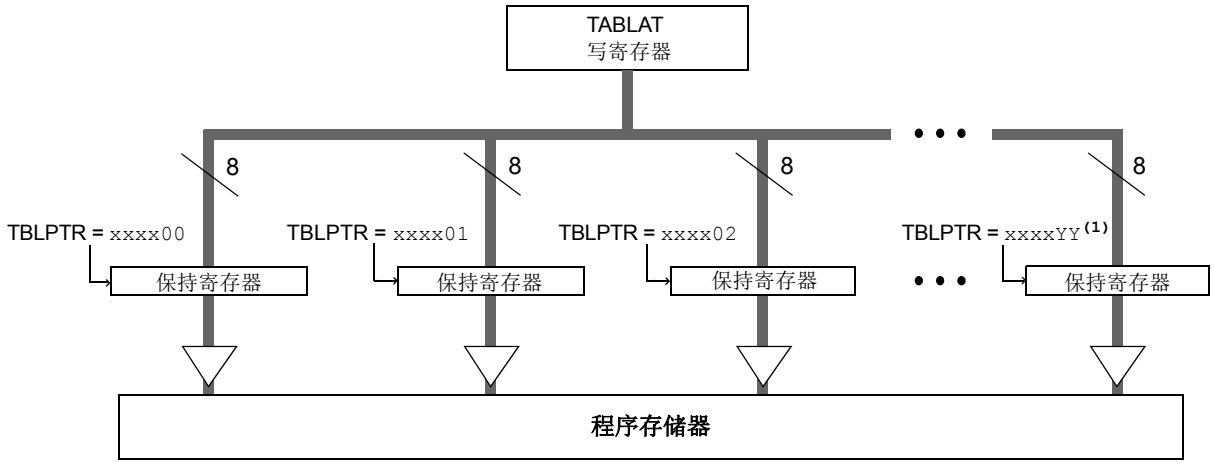
如果TBLPTR中的PFM地址受写保护或如果TBLPTR指向无效单元，则WR位清零且不产生任何影响，同时WRERR位置1。

编程程序存储器时需要执行长写操作。CPU操作在长写周期内暂停，并在操作完成时恢复。长写操作在一个指令周期内完成。操作完成时，WR由硬件清零，NVMIF置1，并且在NVMIE也置1时将发生中断。锁存数据复位为全1。WREN保持不变。

内部编程定时器控制写时间。写入/擦除电压是由片上电荷泵产生的，此电荷泵在器件的电压范围内工作。

注： 保持寄存器在器件复位和写操作后的默认值为FFh。将FFh写入保持寄存器不会修改该字节。这意味着可以修改程序存储器的各个字节，前提是修改后不会尝试将任何位从0更改为1。修改各个字节时，无需事先装载所有保持寄存器即可执行长写操作。

图13-8： 表写入闪存程序存储器



注 1： 关于保持寄存器的数量，请参见表5-4（例如，YY = 3F表示64个保持寄存器）。

13.1.6.1 闪存程序存储器写序列

编程内部程序存储单元的事件序列如下：

1. 将适当数量的字节读入RAM。关于写锁存器的大小，请参见表13-2。
2. 根据需要更新RAM中的数据值。
3. 向表指针寄存器中装载正在擦除的地址。
4. 执行块擦除过程。
5. 向表指针寄存器中装载正在写入的第一个字节的地址。
6. 以自动递增方式向保持寄存器中写入n字节块。关于写锁存器的大小，请参见表13-2。
7. 设置REG<1:0>位以指向程序存储器。
8. 将NVMCON1寄存器中的FREE位清零并将WREN位置1。
9. 禁止中断。
10. 执行解锁序列（见**第13.1.4节“NVM解锁序列”**）。
11. 将NVMCON1寄存器中的WR位置1。
12. CPU将在写操作期间内停顿（大约为2 ms，使用内部定时器计时）。
13. 重新允许中断。
14. 验证存储器（表读）。

该过程需要大约6 ms的时间来更新存储器的每个写块。例13-4给出了所需代码示例。

注：	将WR位置1之前，表指针地址需要位于保持寄存器中字节的目标地址范围内。

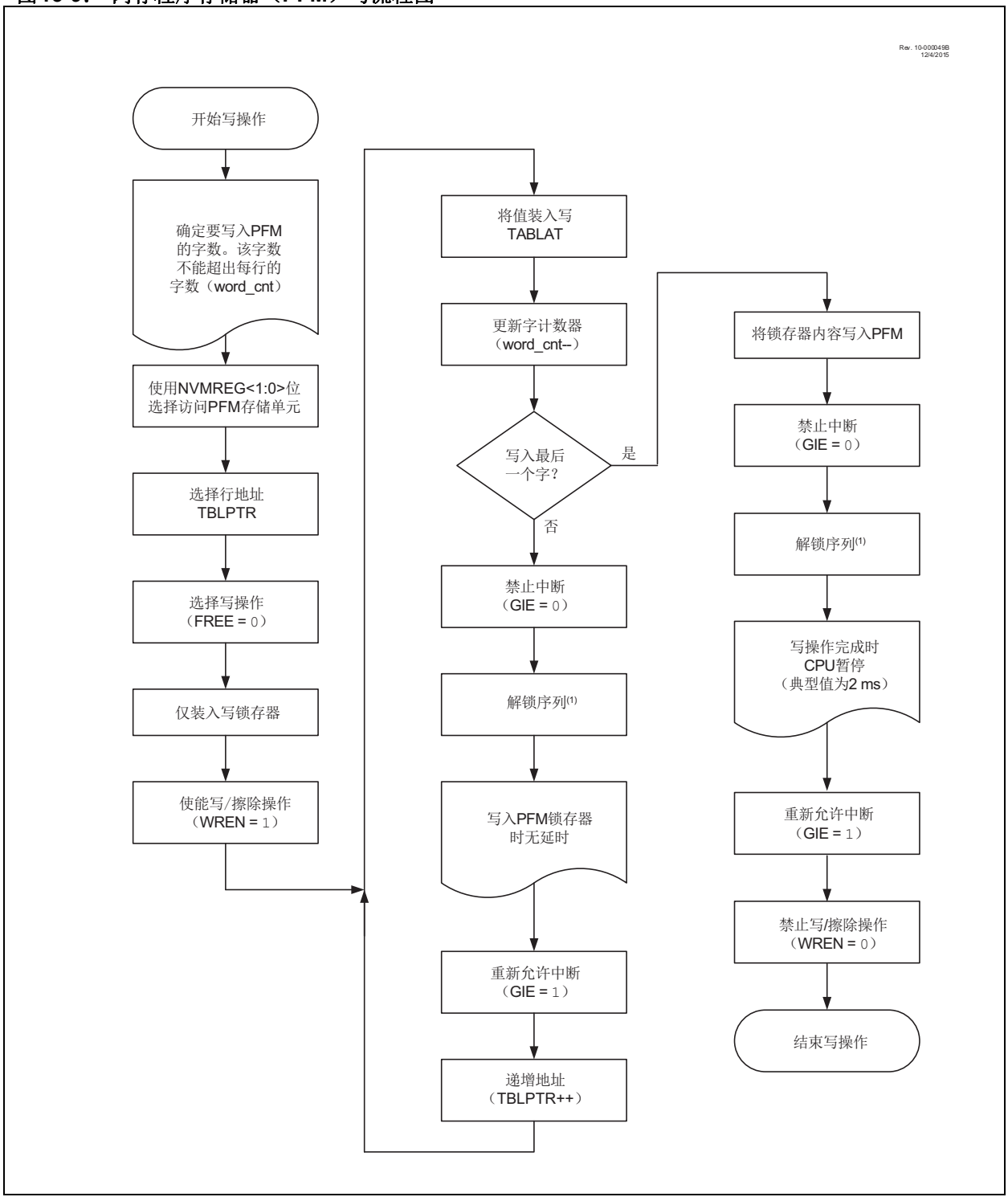
例13-4： 写入闪存程序存储器

```
                MOVLW     D'64'                 ; number of bytes in erase block
                MOVWF     COUNTER
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
                MOVLW     CODE_ADDR_UPPER       ; Load TBLPTR with the base
                MOVWF     TBLPTRU               ; address of the memory block
                MOVLW     CODE_ADDR_HIGH
                MOVWF     TBLPTRH
                MOVLW     CODE_ADDR_LOW
                MOVWF     TBLPTRL
READ_BLOCK
                TBLRD*+                         ; read into TABLAT, and inc
                MOVF      TABLAT, W             ; get data
                MOVWF     POSTINC0              ; store data
                DECFSZ    COUNTER               ; done?
                BRA       READ_BLOCK            ; repeat
MODIFY_WORD
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
                MOVLW     NEW_DATA_LOW          ; update buffer word
                MOVWF     POSTINC0
                MOVLW     NEW_DATA_HIGH
                MOVWF     INDF0
ERASE_BLOCK
                MOVLW     CODE_ADDR_UPPER       ; load TBLPTR with the base
                MOVWF     TBLPTRU               ; address of the memory block
                MOVLW     CODE_ADDR_HIGH
                MOVWF     TBLPTRH
                MOVLW     CODE_ADDR_LOW
                MOVWF     TBLPTRL
                BCF       NVMCON1, REG0         ; point to Program Flash Memory
                BSF       NVMCON1, REG1         ; point to Program Flash Memory
                BSF       NVMCON1, WREN         ; enable write to memory
                BSF       NVMCON1, FREE         ; enable Erase operation
                BCF       INTCON0, GIE          ; disable interrupts
                MOVLW     55h
所需序列        MOVWF     NVMCON2               ; write 55h
                MOVLW     AAh
                MOVWF     NVMCON2               ; write 0AAh
                BSF       NVMCON1, WR           ; start erase (CPU stall)
                BSF       INTCON0, GIE          ; re-enable interrupts
                TBLRD*-                         ; dummy read decrement
                MOVLW     BUFFER_ADDR_HIGH      ; point to buffer
                MOVWF     FSR0H
                MOVLW     BUFFER_ADDR_LOW
                MOVWF     FSR0L
WRITE_BUFFER_BACK
                MOVLW     BlockSize             ; number of bytes in holding register
                MOVWF     COUNTER
                MOVLW     D'64'/BlockSize       ; number of write blocks in 64 bytes
                MOVWF     COUNTER2
```

例13-4：　写入闪存程序存储器（续）

```
WRITE_BYTE_TO_HREGS
              MOVF      POSTINC0, W              ; get low byte of buffer data
              MOVWF     TABLAT                   ; present data to table latch
              TBLWT+*                            ; write data, perform a short write
                                                 ; to internal TBLWT holding register.
              DECFSZ    COUNTER                  ; loop until holding registers are full
              BRA       WRITE_WORD_TO_HREGS
PROGRAM_MEMORY
              BCF       NVMCON1, REG0            ; point to Program Flash Memory
              BSF       NVMCON1, REG1            ; point to Program Flash Memory
              BSF       NVMCON1, WREN            ; enable write to memory
              BCF       NVMCON1, FREE            ; enable write to memory
              BCF       INTCON0, GIE             ; disable interrupts
              MOVLW     55h
所需序列      MOVWF     NVMCON2                  ; write 55h
              MOVLW     0AAh
              MOVWF     NVMCON2                  ; write 0AAh
              BSF       NVMCON1, WR              ; start program (CPU stall)
              DCFSZ     COUNTER2                 ; repeat for remaining write blocks
              BRA       WRITE_BYTE_TO_HREGS
              BSF       INTCON0, GIE             ; re-enable interrupts
              BCF       NVMCON1, WREN            ; disable write to memory
```

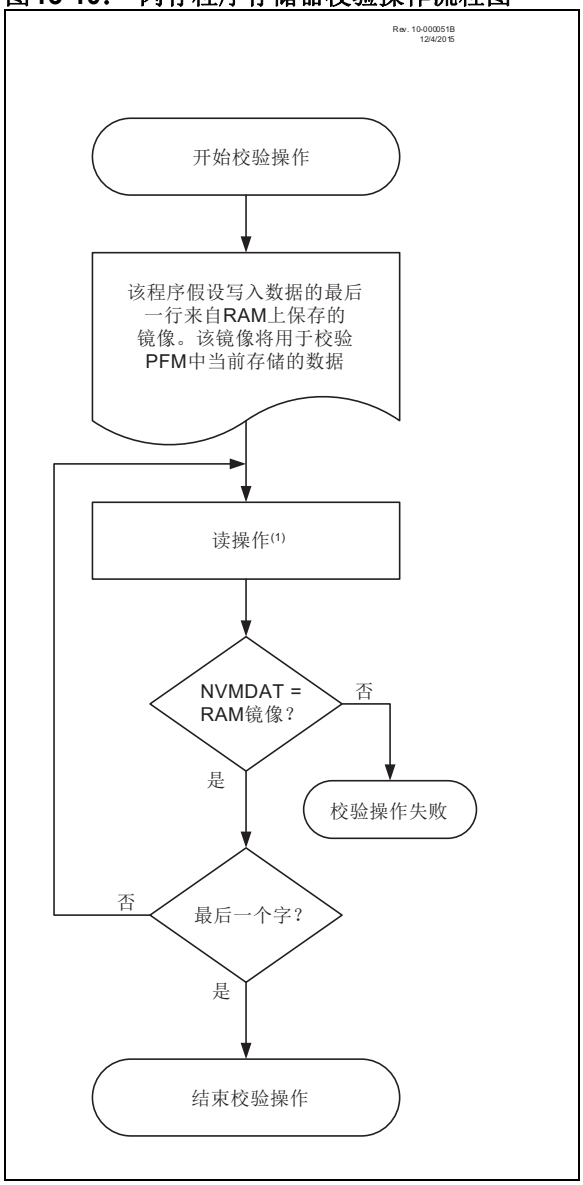
图13-9： 闪存程序存储器（PFM）写流程图



13.1.6.2 写校验

根据具体应用，好的编程做法可能要求校验要写入存储器的值与原始值是否一致。在应用中，如果某些位的写次数接近规范限值，则应使用写校验。由于程序存储器以整页形式存储，因此所存储的程序存储器内容将在最后一次写操作完成之后与RAM中存储的预期数据进行比较。

图13-10： 闪存程序存储器校验操作流程图



13.1.6.3 意外终止写操作

如果写操作因意外事件（例如断电或意外复位）而终止，则应校验刚刚编程的存储单元并在必要时重新编程。如果写操作在正常操作期间因MCLR复位或WDT超时复位而中断，则WRERR位将置1，用户可通过检查该位来决定是否需要重写存储单元。

13.1.6.4 防止误写操作的保护措施

写序列仅在同时满足以下两个条件时才有效，这样可以防止可能导致数据损坏的误写操作。

1. WR位通过WREN位进行门控。除了存储器写操作期间，建议始终清零WREN位。这样可以防止因WR位意外置1而触发存储器写操作。
2. 每次执行写操作之前必须先执行NVM解锁序列。

13.2 器件信息区、器件配置区、用户ID、器件ID和配置字访问

当NVMCON1寄存器中的REG<1:0> = 0x01或0x11时，可以访问器件信息区、器件配置区、用户ID、器件ID/版本ID和配置字。可能存在不同的读写访问权限（见表13-1）。

13.2.1 读访问

用户可通过将REG位设置为0x01或0x11来读取这些块。用户需要将地址装入TBLPTR寄存器。随后执行TBLRD会移动指向TABLAT寄存器的字节。CPU操作在读操作期间暂停，并在完成之后恢复。对表13-1中所列地址外的其他地址启动读访问时，TABLAT寄存器会被清零，读回0。

13.2.2 写访问

NVMCON1中的WREN位必须置1才能使能写操作。这样可防止因错误（意外）代码执行而对CONFIG字进行意外的写操作。除了更新CONFIG字时，WREN位应始终清零。WREN位不由硬件清零。除非WREN位置1，否则将禁止WR位置1。

在执行写命令之前，用户需要将地址和数据字节分别装入TBLPTR和TABLAT寄存器。写入用户ID/器件ID/配置字时，需要遵循解锁序列（第13.1.4节，NVM解锁序列）。如果WRTC = 0或如果TBLPTR指向无效地址单元（见表13-1），则WR位清零且不产生任何影响，同时WRERR位置1。

一次写入一个配置字字节，并且操作包括该字节的隐式擦除周期（不必将FREE置1）。CPU执行会停止，并且在写周期完成时，WR位通过硬件清零，NVM中断标志位（NVMIF）置1。当CPU恢复工作时，新的CONFIG值生效。

表13-4： DIA、DCI、用户ID、器件/版本ID和配置字访问（REG<1:0> = x1）

地址	功能	读访问	写访问
20 0000h-20 000Fh	用户ID	支持	支持
30 0000h-30 0009h	配置字	支持	支持
3F 0000h-3F 003Fh	DIA	支持	不支持
3F FF00h-3F FF09h	DCI	支持	不支持
3F FFFCh-3F FFFFh	版本ID/器件ID	支持	不支持

13.3 数据EEPROM存储器

数据EEPROM是非易失性存储器阵列，独立于数据RAM和程序存储器，用于程序数据的长期存储。它并不直接映射到寄存器文件或程序存储空间，而是通过特殊功能寄存器（SFR）来间接寻址。在整个VDD范围内，正常操作期间，EEPROM都是可读写的。

有4个SFR用于读写数据EEPROM以及程序存储器。具体包括：

- NVMCON1
- NVMCON2
- NVMDAT
- NVMADRL
- NVMADRH

数据EEPROM支持字节读写。与数据存储块接口时，NVMDAT存放要读/写的8位数据，而NVMADRH:NVMADRL寄存器对存放正在访问的EEPROM单元的地址。

EEPROM数据存储器具有高耐擦写次数。字节写操作会自动擦除目标单元并写入新数据（在写入前擦除）。写入时间由内部编程定时器控制，其值随电压和温度而变化且因器件而异。关于限值，请参见**第45.0节“电气规范”**中的数据EEPROM存储器参数。

13.3.1 NVMADRL和NVMADRH寄存器

NVMADRH:NVMADRL寄存器用于寻址数据EEPROM以进行读写操作。

13.3.2 NVMCON1和NVMCON2寄存器

访问数据EEPROM时由两个寄存器控制：NVMCON1和NVMCON2。这两个寄存器也用于控制对程序存储器的访问，使用方法与访问数据EEPROM时类似。

NVMCON1寄存器（寄存器13-1）是关于数据和程序存储器访问的控制寄存器。控制位REG<1:0>决定访问的是程序存储器，数据EEPROM存储器还是用户ID、配置位、版本ID和器件ID。

当WREN位置1时，允许进行写操作。上电时，WREN位被清零。

WRERR位在WR位置1时由硬件置1，而在内部编程定时器超时和写操作完成时清零。

WR控制位用于启动写操作。可以用软件将该位置1，但不能清零。该位只能在写操作完成时由硬件清零。

当写操作完成时，PIR0寄存器的NVMIF中断标志位置1。该位必须用软件清零。

控制位RD和WR分别用于启动读和擦除/写操作。这些位由固件置1，并在写操作完成时由硬件清零。

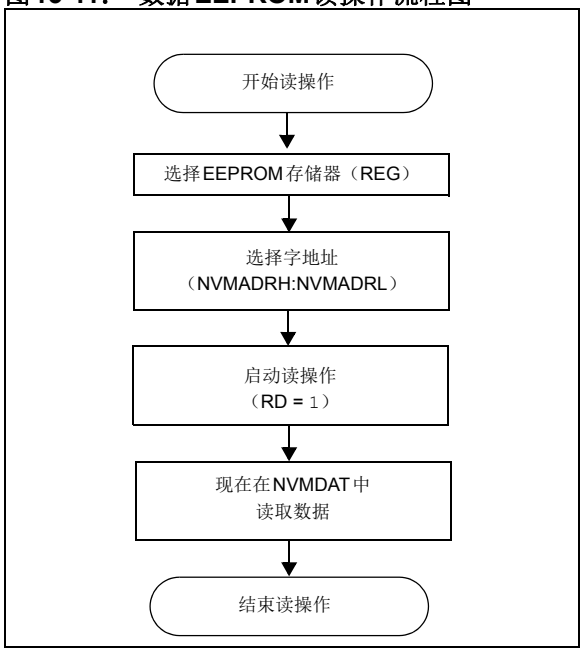
访问程序存储器（REG<1:0> = 0x10）时，RD位不能置1。程序存储器使用表读指令读取。关于表读，请参见**第13.1.1节“表读和表写”**。

13.3.3 读取数据EEPROM存储器

要读取数据存储单元，用户必须将地址写入NVMADRL和NVMADRH寄存器对，通过清零NVMCON1寄存器中的REG<1:0>控制位来访问数据EEPROM单元，然后将控制位RD置1。在紧接着的下一个指令周期数据即可使用；因此，可通过随后的指令读取NVMDAT寄存器。NVMDAT会将此值保存至下一次读操作，或保存至用户写入数据时（写操作）为止。

基本过程如例13-5所示。

图13-11： 数据EEPROM读操作流程图



13.3.4 写入数据EEPROM存储器

要写入EEPROM数据单元，必须先将地址写入NVMADRL和NVMADRH寄存器对，再将数据写入NVMDAT寄存器。必须遵循例13-6中的序列以启动写周期。

如果并非每个字节均严格遵循**第13.1.4节“NVM解锁序列”**中所述的NVM解锁序列，则不会启动写操作。强烈建议在执行此代码段时禁止中断。

此外，NVMCON1中的WREN位必须置1才能使能写操作。此机制可防止因意外代码执行（即，跑飞程序）而对数据EEPROM进行意外的写操作。除了更新EEPROM时，WREN位应始终清零。WREN位不由硬件清零。

启动写序列后，不能修改NVMCON1、NVMADRL、NVMADRH和NVMDAT。除非WREN位置1，否则将禁止WR位置1。不能使用同一指令将WR和WREN置1。

启动写序列后，清零WREN位不会影响此写周期。写入一个数据EEPROM字，并且操作包括该字的隐式擦除周期（不必将FREE置1）。CPU继续执行与之并行进行，并且在写周期完成时，WR位由硬件清零，同时NVM中断标志位（NVMIF）置1。用户可允许此中断或轮询此位。必须用软件将NVMIF清零。

13.3.5 写校验

根据具体应用，好的编程做法可能要求校验要写入存储器的值与原始值是否一致。在应用中，如果某些位的写次数接近规范限值，则应使用写校验。

例13-5： 读取数据EEPROM

```
; Data Memory Address to read
            CLRF      NVMCON1          ; Setup Data EEPROM Access
            MOVF      EE_ADDRL, W      ;
            MOVWF     NVMADRL          ; Setup Address
            BSF       NVMCON1, RD      ; Issue EE Read
            MOVF      NVMDAT, W        ; W = EE_DATA
```

例13-6： 写入数据EEPROM

```
; Data Memory Address to write
            CLRF      NVMCON1          ; Setup Data EEPROM Access
            MOVF      EE_ADDRL, W      ;
            MOVWF     NVMADRL          ; Setup Address
; Data Memory Value to write
            MOVF      EE_DATA, W       ;
            MOVWF     NVMDAT           ;
; Enable writes
            BSF       NVMCON1, WREN    ;
; Disable interrupts
            BCF       INTCON0, GIE     ;
; Required unlock sequence
            MOVLW     55h              ;
            MOVWF     NVMCON2          ;
            MOVLW     AAh              ;
            MOVWF     NVMCON2          ;
; Set WR bit to begin write
            BSF       NVMCON1, WR      ;
; Enable INT
            BSF       INTCON0, GIE     ;
; Wait for interrupt, write done
            SLEEP                      ;
; Disable writes
            BCF       NVMCON1, WREN    ;
```

13.3.6 代码保护期间的操作

数据EEPROM存储器在配置字中有自己的代码保护位。如果使能代码保护，则禁止外部读写操作。

如果数据EEPROM受写保护或如果NVMADR指向无效地址单元，则WR位清零，而不产生任何影响。在这种情况下，会通过WRERR指示错误状态。

13.3.7 防止误写操作的保护措施

有些情况下，用户可能不希望写入数据EEPROM存储器。为了防止误写EEPROM，器件实现了各种保护机制。上电时，WREN位被清零。此外，在上电延时定时器周期（TPWRT）内也会阻止对EEPROM进行写操作。

在欠压、电源故障或软件故障期间，解锁序列以及WREN位有助于防止意外写操作的发生。

13.3.8 擦除数据EEPROM存储器

通过向数据EEPROM存储器中需要擦除的所有存储单元写入0xFF，可以擦除数据EEPROM存储器。

例13-7： 数据EEPROM刷新程序

```
        CLRF    NVMADRL             ; Start at address 0
        BCF     NVMCON1, CFGS       ; Set for memory
        BCF     NVMCON1, EEPGD      ; Set for Data EEPROM
        BCF     INTCON0, GIE        ; Disable interrupts
        BSF     NVMCON1, WREN       ; Enable writes
Loop                                ; Loop to refresh array
        BSF     NVMCON1, RD         ; Read current address
        MOVLW   55h                 ;
        MOVWF   NVMCON2             ; Write 55h
        MOVLW   0AAh                ;
        MOVWF   NVMCOM2             ; Write 0AAh
        BSF     NVMCON1, WR         ; Set WR bit to begin write
        BTFSC   NVMCON1, WR         ; Wait for write to complete
        BRA     $-2
        INCFSZ  NVMADRL, F          ; Increment address
        BRA     LOOP                ; Not zero, do it again

        BCF     NVMCON1, WREN       ; Disable writes
        BSF     INTCON0, GIE        ; Enable interrupts
```

13.4 寄存器定义：非易失性存储器

寄存器13-1： NVMCON1：非易失性存储器控制1寄存器

R/W-0/0	R/W-0/0	U-0	R/S/HC-0/0	R/W/HS-x/q	R/W-0/0	R/S/HC-0/0	R/S/HC-0/0
REG<1:0>		—	FREE	WRERR	WREN	WR	RD
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	HC = 硬件清零位
x = 未知	-n = POR时的值	S = 可以用软件将位置1，但不能清零
0 = 清零	1 = 置1	U = 未实现位，读为0

bit 7-6 **REG<1:0>**：NVM区域选择位
10 = 访问PFM单元
x1 = 访问用户ID、配置位、DIA、DCI、版本ID和器件ID
00 = 访问数据EEPROM存储单元

bit 5 **未实现**：读为0

bit 4 **FREE**：闪存程序存储器擦除使能位(1)
1 = 在下一条WR命令时执行擦除操作
0 = 下一条WR命令执行写操作

bit 3 **WRERR**：写复位错误标志位(2,3,4)
1 = 写操作被复位中断（硬件置1），
*或*在访问无效地址时WR被写入1’b1（表4-1和表13-1）
*或*在REG<1:0>和地址未指向同一区域时WR被写入1’b1
*或*在访问写保护地址时WR被写入1’b1（表4-2）。
0 = 正常完成所有写操作

bit 2 **WREN**：编程/擦除使能位
1 = 允许编程/擦除和刷新周期
0 = 禁止NVM的编程/擦除和用户刷新

bit 1 **WR**：写控制位(5,6,7)
当REG指向数据EEPROM存储单元时：
1 = 启动相应数据EEPROM存储单元的擦除/编程周期
当REG指向PFM单元时：
1 = 使用保持寄存器中的数据启动PFM写操作
0 = 对NVM的编程/擦除操作已完成并且变为无效

bit 0 **RD**：读控制位(8)
1 = 启动REG和NVMADR指向的地址，并将数据装入NVMDAT
0 = NVM读操作已完成并且变为无效

注 1：该位只能与PFM配合使用。
2：当WR = 1时，该位置1，而当内部编程定时器超时或写操作成功完成时，该位清零。
3：该位必须由用户清零；硬件不能将该位清零。
4：该位可由用户写入1以实现测试序列。
5：只能紧接着**第13.1.4节“NVM解锁序列”**的解锁序列将该位置1。
6：操作是自计时的，并且当操作完成时WR位由硬件清零。
7：启动写操作之后，将该位设置为0将不起作用。
8：该位只能用软件置1。该位在操作完成时由硬件清零。

寄存器13-2： NVMCON2：非易失性存储器控制2寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
NVMCON2<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
x = 未知	0 = 清零	1 = 置1
-n = POR时的值		

bit 7-0 **NVMCON2<7:0>：**
请参见**第13.1.4节“NVM解锁序列”**。

注 1： 该寄存器始终读为0，与写入的数据无关。

寄存器13-3： NVMADRL：数据EEPROM存储器地址低字节

R/W-x/0	R/W-x/0	R/W-x/0	R/W-x/0	R/W-x/0	R/W-x/0	R/W-x/0	R/W-x/0
ADR<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
x = 未知	0 = 清零	1 = 置1
-n = POR时的值		

bit 7-0 **ADR<7:0>：**EEPROM读地址位

寄存器13-4： NVMADRH：数据EEPROM存储器地址高字节

U-0	U-0	U-0	U-0	U-0	U-0	R/W-x/u	R/W-x/u
—	—	—	—	—	—	ADR<9:8>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
x = 未知	0 = 清零	1 = 置1
-n = POR时的值		

bit 7-2 **未实现：**读为0

bit 1-0 **ADR<9:8>：**EEPROM读地址位

寄存器13-5： NVMDAT：数据EEPROM存储器数据

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
DAT<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
x = 未知	0 = 清零	1 = 置1
-n = POR时的值		

bit 7-0 **DAT<7:0>：**读命令之后从NVMADR返回的数据存储字的值，或写命令写入的数据。

表13-5： 与非易失性存储器控制相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
NVMCON1	REG<1:0>		—	FREE	WRERR	WREN	WR	RD	200
NVMCON2	解锁模式								201
NVMADRL	NVMADR<7:0>								201
NVMADRH	—	—	—	—	—	—	NVMADR<9:8>		201
NVMDAT	NVMDAT<7:0>								202

图注： — = 未实现，读为0。在EEPROM访问期间不使用阴影位。

*该页提供寄存器信息。

14.0 带存储器扫描器的循环冗余校验（CRC）模块

循环冗余校验（Cyclic Redundancy Check，CRC）模块提供了一个软件可配置的硬件实现的CRC校验和发生器。该模块具有以下特性：

- 可使用最高16位的任意标准CRC
- 可配置多项式
- 可使用最高16位的任意种子值
- 标准和反向位序可用
- 可以自动或由用户添加扩充零
- 用于对程序/数据EEPROM存储器用户数据快速计算CRC的存储器扫描器
- 用于通信CRC的软件可装入数据寄存器

14.1 CRC模块概述

CRC模块提供了一种计算程序/数据EEPROM存储器校验值的方式。CRC模块与存储器扫描器配合使用，可实现更快的CRC计算。存储器扫描器可以自动向CRC模块提供数据。CRC模块也可以在直接向SFR写入数据、无需使用扫描器的情况下工作。

14.2 CRC功能概述

CRC模块可用于使用内置存储器扫描器或通过用户输入RAM存储器来检测程序存储器中的位错误。CRC模块可接受最高16位的多项式与最高16位的种子值。然后，CRC计算的校验值（或校验和）会被生成到CRCACC<15:0>寄存器中，供用户存储。CRC模块使用异或（XOR）移位寄存器实现来执行CRC计算所需的多项式除法。

例14-1： CRC示例

Rev. 10-000206A
1/8/2014

CRC-16-ANSI

$x^{16} + x^{15} + x^2 + 1$（17位）

标准16位表示 = 0x8005

CRCXORH = 0b10000000
CRCXORL = 0b0000010- (1)

数据序列：

0x55, 0x66, 0x77, 0x88

DLEN = 0b0111
PLEN = 0b1111

输入到CRC的数据：

SHIFTM = 0：

01010101 01100110 01110111 10001000

SHIFTM = 1：

10101010 01100110 11101110 00010001

校验值（ACCM = 1）：

SHIFTM = 0：0x32D6
CRCACCH = 0b00110010
CRCACCL = 0b11010110

SHIFTM = 1：0x6BA2
CRCACCH = 0b01101011
CRCACCL = 0b10100010

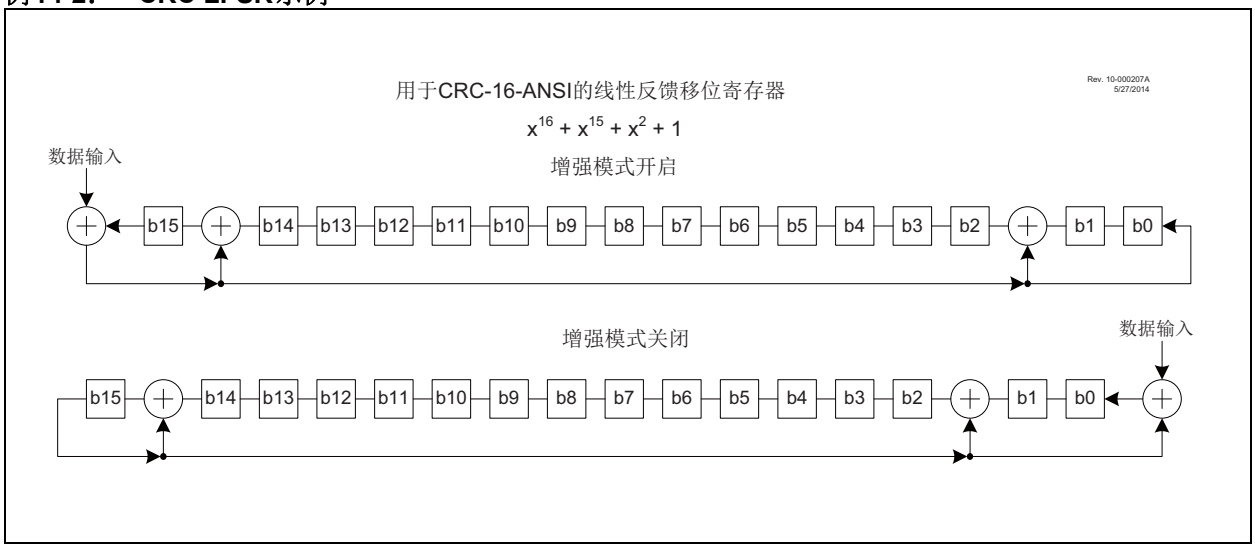
注1：Bit 0未实现。任何CRC多项式的LSb始终为1，并且CRC将始终将其视为1以计算CRC校验值。该位将在软件中读为0。

14.3 CRC多项式实现

可以使用任意多项式。多项式和累加器大小由PLEN<3:0>位确定。对于n位累加器，PLEN = n-1，相应的多项式为n+1位。因此，累加器大小任意，最多16位，相应的多项式最多17位。多项式的MSb和LSb总是由硬件强制为1。MSb和LSb之间的所有多项式位均由CRCXOR寄存器指定。例如，使用CRC-16-ANSI时，多项式定义为$X^{16}+X^{15}+X^{2}+1$。

X^{16}和X^{0} = 1项为MSb和LSb，由硬件控制。X^{15}和X^{2}项通过将相应的CRCXOR<15:0>位设置为0x8004值来指定。实际值为0x8005，因为硬件将LSb设置为1。但是，CRCXORL寄存器的LSb是未实现位，总是读为0。请参见例14-1。

例14-2： CRC LFSR示例



14.4 CRC数据源

数据可以通过两种方式输入CRC模块：

- 用户数据，使用CRCDAT寄存器（CRCDATH和CRCDATL）
- 程序存储器，使用程序存储器扫描器

要设置数据的位数（最高16位），必须相应地设置CRCCON1的DLEN位。该模块将仅使用CRCDAT寄存器中最高到DLEN的数据位，并忽略CRCDAT寄存器中的其他数据位。

数据会被移入CRCSHIFT，作为一种中间结果来计算位于CRCACC寄存器中的校验值。

SHIFTM位用于确定移入累加器的数据的位序。如果SHIFTM未置1，则先移入数据的MSb（大尾数法）。DLEN的值将决定MSb。如果SHIFTM位置1，则将按反序（LSb最先）将数据移入累加器（小尾数法）。

通过在开始CRC之前将CRCACC<15:0>寄存器设置为适当的值，可以设置CRC模块的初始种子值。

14.4.1 用户数据的CRC

要对用户的数据输入使用CRC模块，用户必须将数据写入CRCDAT寄存器。CRCDAT寄存器中的数据将会在每次写入CRCDATL寄存器时被锁存到移位寄存器中。

14.4.2 闪存的CRC

要对位于程序存储器中的数据使用CRC模块，用户可以初始化程序存储器扫描器，如**第14.8节，扫描器模块概述**所述。

14.5 CRC校验值

在CRC计算完毕后，CRC校验值将位于CRCACC寄存器。校验值将取决于CRCCON0寄存器的两种模式设置：ACCM和SHIFTM。如果ACCM位置1，CRC模块会使用一定数量（等于多项式长度）的零来扩充数据，以对齐最终的校验值。如果ACCM位未置1，CRC将在数据结束处停止。然后，可以在CRCDAT中输入一定数量（等于多项式长度）的零，以确定与扩充模式相同的校验值。或者也可以在此时输入预期的校验值，使最终结果等于0。

在SHIFTM位置1、选择先移入LSb且ACCM位置1的情况下计算得到CRC校验值时，CRCACC寄存器中的最终值会变为反序，使LSb处于MSb位置，反之亦然。它是以反向位序形式表示的预期校验值。如果要生成追加到数据流后的校验值，则必须对最终值执行倒转位序的操作，以获得正确的校验和。可以使用CRC通过以下方法来执行这种倒转：

- 将CRCACC值保存到用户RAM空间中
- 清零CRCACC寄存器
- 清零CRCXOR寄存器
- 将已保存的CRCACC值写入CRCDAT输入。

最终，CRCACC寄存器中将为方向正确的校验值。

14.6 CRC中断

BUSY位从1变为0时，CRC将会产生中断。每次BUSY位发生变化时，不论是否允许CRC中断，CRCIF中断标志位都会置1。CRCIF位只能用软件清零。

14.7 配置CRC

以下步骤说明了如何正确配置CRC。

1. 确定是使用扫描器进行自动程序存储器扫描还是通过SFR接口进行手动计算，并根据所作的决定执行**第14.4节“CRC数据源”**中指定的操作。
2. 如果需要，在CRCACCH/L寄存器中设置起始CRC值作为种子。
3. 使用所需的发生器多项式设定CRCXORH/L寄存器。
4. 使用数据字长度-1设定CRCCON1寄存器的DLEN<3:0>位（见例14-1）。这将决定对于每个数据字，移位器将移入累加器多少次。
5. 使用多项式长度-2设定CRCCON1寄存器的PLEN<3:0>位（见例14-1）。
6. 确定是否需要移入尾随零，并相应地设置CRCCON0寄存器的ACCM位。
7. 类似地，确定是先移入MSb还是LSb，并相应地写入CRCCON0寄存器的SHIFTM位。
8. 写入CRCCON0寄存器的GO位，开始移位过程。

9a. 如果使用手动SFR输入，则监视CRCCON0寄存器的FULL位。当FULL = 0时，可以向CRCDATH/L寄存器写入另一个数据字；请记住，如果数据长度大于8位，则应先写入CRCDATH，因为移位器会在写入CRCDATL寄存器时开始。

9b. 如果使用扫描器，则只要GO位置1，扫描器就会根据需要自动在CRCDATH/L寄存器中装入数据字。

10a. 如果使用手动输入，则通过监视CRCIF（和BUSY位）来确定何时可以从CRCACCH/L寄存器中读取完成的CRC计算结果。

10b. 如果使用存储器扫描器，则通过监视SCANIF（或GO位）来确定扫描器是否完成将信息压入CRCDAT寄存器。在扫描器完成之后，通过监视BUSY位来确定CRC是否已完成，之后可以从CRCACC寄存器中读取校验值。如果两个中断标志均置1，且BUSY和GO位均清零，则可以从CRCACCH/L寄存器中读取完成的CRC计算结果。

14.8 扫描器模块概述

扫描器支持将闪存程序存储器或数据EEPROM的段读出（扫描）到CRC外设。扫描器模块与CRC模块交互，一次为其提供一个数据字。数据从SCANLADR寄存器到SCANHADR寄存器定义的地址范围获取。

扫描器在SGO位置1（SCANCON0寄存器）时开始工作，在用户清零SGO或SCANLADR递增至超过SCANHADR时结束工作。也可以通过清零EN位（CRCCON0寄存器）清零SGO位。

14.9 配置扫描器

可以将扫描器模块与CRC模块配合使用，对一个程序存储器或数据EEPROM地址范围执行CRC计算。为了设置扫描器，使之与CRC配合工作，需要执行以下步骤：

1. 设置CRC模块（见**第14.7节“配置CRC”**），并通过将SCANCON0寄存器中的EN位置1来使能扫描器模块。
2. 选择扫描器模块应在哪个存储区操作，并相应地设置SCANCON0寄存器的MREG位。
3. 如果触发器用于扫描器操作，则将SCANCON0寄存器的TRIGEN位置1，并使用SCANTRIG寄存器选择触发源。使用SCANTRIG寄存器选择触发源，然后将SCANCON0寄存器的TRIGEN位置1。关于扫描器操作，请参见表14-1。
4. 如果需要突发操作模式，则将BURSTMD位（SCANCON0寄存器）置1。关于扫描器操作，请参见表14-1。
5. 使用要扫描的存储器的起始和结束位置设置SCANLADRL/H/U和SCANHADRL/H/U寄存器。
6. 为扫描器模块选择优先级（见**第3.1节“系统仲裁”**）并锁定优先级（见**第3.1.1节“优先级锁定”**）。
7. 必须使能CRCEN和CRCGO位才能使用扫描器。将SGO位置1将启动扫描器操作。

14.10 扫描器中断

当SCANLADR递增至超过SCANHADR时，扫描器将触发中断。SCANIF位只能用软件清零。

14.11 扫描模式

扫描器与系统操作的交互由系统仲裁器中的优先级选择控制（见**第3.2节“存储器访问机制”**）。此外，BURSTMD和TRIGEN还确定扫描器的操作。

14.11.1 TRIGEN = 0，BURSTMD = 0

在这种情况下，如果没有其他优先级更高的源请求访问，则将存储器访问请求授予扫描器。

所有优先级低于扫描器的源都将获得扫描器未使用的存储器访问周期。

14.11.2 TRIGEN = 1，BURSTMD = 0

在这种情况下，当CRC模块准备好接受时，生成存储器访问请求。

如果没有其他优先级更高的源请求访问，则将存储器访问请求授予扫描器。所有优先级低于扫描器的源都将获得扫描器未使用的存储器访问周期。
如果没有其他优先级更高的源请求访问，则将存储器访问请求授予扫描器。所有优先级低于扫描器的源都将获得扫描器未使用的存储器访问周期。

14.11.3 TRIGEN = x，BURSTMD = 1

在这种情况下，扫描器始终请求存储器访问。

如果没有其他优先级更高的源请求访问，则将存储器访问请求授予扫描器。在存储器完成操作（即SGO = 0（SCANCON0寄存器））之前，不会将存储器访问周期授予优先级低于扫描器的源。

注： 如果TRIGEN = 1且BURSTMD = 1，则用户应确保触发源处于活动状态，以便完成扫描器操作。

14.12 寄存器定义：CRC和扫描器控制

CRC和扫描器外设的长位名称前缀如下所示。更多信息，请参见**第1.3.2.2节“长位名称”**。

外设	位名称前缀
CRC	CRC

寄存器14-1： CRCCON0：CRC控制寄存器0

R/W-0/0	R/W-0/0	R-0	R/W-0/0	U-0	U-0	R/W-0/0	R-0
EN	GO	BUSY	ACCM	—	—	SHIFTM	FULL
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN**：CRC使能位
1 = 使能CRC模块
0 = 禁止CRC

bit 6 **GO**：CRC Go位
1 = 启动CRC串行移位器
0 = 关闭CRC串行移位器

bit 5 **BUSY**：CRC忙位
1 = 正在进行或正在等待进行移位
0 = 移位器中的所有有效位都已被移入到累加器

bit 4 **ACCM**：累加器模式位
1 = 使用零扩充数据
0 = 不使用零扩充数据

bit 3-2 **未实现**：读为0

bit 1 **SHIFTM**：移位模式位
1 = 右移（LSb）
0 = 左移（MSb）

bit 0 **FULL**：数据路径满指示位
1 = CRCDATH/L寄存器已满
0 = CRCDATH/L寄存器已将其数据移入移位器

寄存器14-2： CRCCON1：CRC控制寄存器1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
DLEN<3:0>				PLEN<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4 **DLEN<3:0>**：数据长度位
表示数据字长度 - 1（见例14-1）

bit 3-0 **PLEN<3:0>**：多项式长度位
表示多项式长度 - 1（见例14-1）

寄存器14-3： CRCDATH：CRC数据高字节寄存器

R/W-xx	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
DATA<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **DATA<15:8>**：CRC输入/输出数据位

寄存器14-4： CRCDATL：CRC数据低字节寄存器

R/W-xx	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
DATA<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **DATA<7:0>**：CRC输入/输出数据位
写入该寄存器将会填充移位器。

寄存器14-5： CRCACCH：CRC累加器高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ACC<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ACC<15:8>**：CRC累加器寄存器位

寄存器14-6： CRCACCL：CRC累加器低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ACC<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ACC<7:0>**：CRC累加器寄存器位

寄存器14-7： CRCSHIFTH：CRC移位高字节寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
SHIFT<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SHIFT<15:8>**：CRC移位器寄存器位
读取该寄存器会读取CRC移位器。

寄存器14-8： CRCSHIFTL：CRC移位低字节寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
SHIFT<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SHIFT<7:0>**：CRC移位器寄存器位
读取该寄存器会读取CRC移位器。

寄存器14-9： CRCXORH：CRC异或高字节寄存器

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
X<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **X<15:8>：**多项式项X^n的异或使能位

寄存器14-10： CRCXORL：CRC异或低字节寄存器

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	U-1
X<7:1>							—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-1 **X<7:1>：**多项式项X^n的异或使能位

bit 0 **未实现：**读为1

寄存器14-11： SCANCON0：扫描器访问控制寄存器0

R/W-0/0	R/W-0/0	R/W/HC-0/0	U-0	U-0	R/W-0/0	R/W-0/0	R-0/0
EN	TRIGEN	SGO	—	—	MREG	BURSTMD	BUSY
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **EN**：扫描器使能位(1)
1 = 使能扫描器
0 = 禁止扫描器

bit 6 **TRIGEN**：扫描器触发使能位(2)
1 = 使能扫描器触发
0 = 禁止扫描器触发
请参见表14-1。

bit 5 **SGO**：扫描器GO位(3, 4)
1 = 当CRC就绪时，将访问由MREG位设置的存储区，并将数据传递到CRC外设。
0 = 不发生扫描器操作

bit 4-3 **未实现**：读为0

bit 2 **MREG**：扫描器存储区选择位(2)
1 = 扫描器地址指向数据EEPROM
0 = 扫描器地址指向闪存程序存储器

bit 1 **BURSTMD**：扫描器突发模式位
1 = 对CPU仲裁器的存储器访问请求始终为真
0 = 对CPU仲裁器的存储器访问请求取决于CRC请求和触发器
请参见表14-1。

bit 0 **BUSY**：扫描器忙指示位
1 = 正在执行扫描器周期
0 = 扫描器周期已完成（或从未启动）

注 1：设置EN = 1（SCANCON0寄存器）不会影响任何其他寄存器内容。
2：可以使用SCANTRIG寄存器设置扫描器触发选择。
3：此位可以用软件清零。当LADR > HADR（并且不会发生数据周期）或CRCGO = 0（CRCCON0寄存器）时，此位通过硬件清零。
4：在将SGO位置1之前，必须先将CRCEN和CRCGO位（CRCCON0寄存器）置1。

表14-1： 扫描器工作模式(1)

TRIGEN	BURSTMD	扫描器操作
0	0	当CRC模块准备好接受数据时，请求存储器访问；如果没有其他优先级更高的源请求待处理，则授予该请求。
1	0	当CRC模块准备好接受数据且触发选择为真时，请求存储器访问；如果没有其他优先级更高的源请求待处理，则授予该请求。
x	1	始终请求存储器访问，如果没有其他优先级更高的源请求待处理，则授予该请求。

注 1： 关于优先级选择，请参见**第3.1节“系统仲裁”**；关于存储器访问机制，请参见**第3.2节“存储器访问机制”**。

寄存器14-12： SCANLADRU：扫描低地址高字节寄存器

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	LADR<21:16>(1,2)					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：** 读为0

bit 5-0 **LADR<21:16>：** 扫描起始/当前地址位(1,2)
要取数据的当前地址的高位，其值在每次存储器取操作时递增。

注 1： 寄存器SCANLADRU/H/L构成一个22位值，但并不保证原子或异步访问；应仅在SGO = 0（SCANCON0寄存器）时读取或写入寄存器。
2： 当SGO = 1（SCANCON0寄存器）时，对该寄存器的写操作会被忽略。

寄存器14-13： SCANLADRH：扫描低地址高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
LADR<15:8>(1, 2)							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **LADR<15:8>：** 扫描起始/当前地址位(1, 2)
要取数据的当前地址的高位，其值在每次存储器取操作时递增。

注 1： 寄存器SCANLADRU/H/L构成一个22位值，但并不保证原子或异步访问；应仅在SGO = 0（SCANCON0寄存器）时读取或写入寄存器。
2： 当SGO = 1（SCANCON0寄存器）时，对该寄存器的写操作会被忽略。

寄存器14-14： SCANLADRL：扫描低地址低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
LADR<7:0>(1, 2)							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **LADR<7:0>：**扫描起始/当前地址位(1, 2)
要取数据的当前地址的低位，其值在每次存储器取操作时递增。

注 1： 寄存器SCANLADRU/H/L构成一个22位值，但并不保证原子或异步访问；应仅在SGO = 0（SCANCON0寄存器）时读取或写入寄存器。
2： 当SGO = 1（SCANCON0寄存器）时，对该寄存器的写操作会被忽略。

寄存器14-15： SCANHADRU：扫描高地址高字节寄存器

U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
—	—	HADR<21:16>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 5-0 **HADR<21:16>：**扫描结束地址位(1, 2)
所指定扫描的结束地址的高位

注 1： 寄存器SCANHADRU/H/L构成一个22位值，但并不保证原子或异步访问；应仅在SGO = 0（SCANCON0寄存器）时读取或写入寄存器。
2： 当SGO = 1（SCANCON0寄存器）时，对该寄存器的写操作会被忽略。

寄存器14-16： SCANHADRH：扫描高地址高字节寄存器

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
HADR<15:8>(1, 2)							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **HADR<15:8>**：扫描结束地址位(1, 2)
所指定扫描的结束地址的高位

注 1： 寄存器SCANHADRU/H/L构成一个22位值，但并不保证原子或异步访问；应仅在SGO = 0（SCANCON0寄存器）时读取或写入寄存器。
2： 当SGO = 1（SCANCON0寄存器）时，对该寄存器的写操作会被忽略。

寄存器14-17： SCANHADRL：扫描高地址低字节寄存器

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
HADR<7:0>(1, 2)							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **HADR<7:0>**：扫描结束地址位(1, 2)
所指定扫描的结束地址的低位

注 1： 寄存器SCANHADRU/H/L构成一个22位值，但并不保证原子或异步访问；应仅在SGO = 0（SCANCON0寄存器）时读取或写入寄存器。
2： 当SGO = 1（SCANCON0寄存器）时，对该寄存器的写操作会被忽略。

寄存器14-18： SCANTRIG：扫描触发选择寄存器

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	TSEL<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4 **未实现：**读为0

bit 3-0 **TSEL<3:0>：**扫描器数据触发输入选择位

1111 = 保留
- •
- •
- •

1010 = 保留
1001 = SMT1_output
1000 = TMR6_postscaled
0111 = TMR5_output
0110 = TMR4_postscaled
0101 = TMR3_output
0100 = TMR2_postscaled
0011 = TMR1_output
0010 = TMR0_output
0001 = CLKREF_output
0000 = LFINTOSC

表14-2： 与CRC相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
CRCACCH	ACC<15:8>								209
CRCACCL	ACC<7:0>								210
CRCCON0	EN	GO	BUSY	ACCM	—	—	SHIFTM	FULL	208
CRCCON1	DLEN<3:0>				PLEN<3:0>				208
CRCDATH	DATA<15:8>								209
CRCDATL	DATA<7:0>								209
CRCSHIFTH	SHIFT<15:8>								210
CRCSHIFTL	SHIFT<7:0>								210
CRCXORH	X<15:8>								211
CRCXORL	X<7:1>							—	211
SCANCON0	EN	TRIGEN	SGO	—	—	MREG	BURSTMD	BUSY	212
SCANHADRU	—	—	HADR<21:16>						214
SCANHADRH	HADR<15:8>								215
SCANHADRL	HADR<7:0>								215
SCANLADRU	—	—	LADR<21:16>						213
SCANLADRH	LADR<15:8>								213
SCANLADRL	LADR<7:0>								214
SCANTRIG	—	—	—	—	TSEL<3:0>				216

图注： — = 未实现位，读为0。CRC模块不使用阴影单元。

15.0 直接存储器访问（DMA）

15.1 简介

直接存储器访问（DMA）模块旨在直接处理不同存储区之间的数据传输，而无需CPU干预。由于无需对数据传输用中断的处理过程进行CPU密集型管理，CPU现在可以有更多时间处理其他任务。

PIC18(L)F25/26K83系列有两个DMA模块，可以独立编程为在不同存储单元之间传输数据，传送不同数据大小，以及使用各种硬件触发来启动传输。这两个DMA寄存器甚至可以编程为一起工作，以便在没有CPU开销的情况下执行更复杂的数据传输。

DMA模块的主要特性包括：

- 支持访问以下存储区：
  - GPR和SFR空间（R/W）
  - 闪存程序存储器（仅限R）
  - 数据EEPROM存储器（仅限R）
- DMA和CPU操作间的优先级可编程。有关详细信息，请参见**第3.1节“系统仲裁”**。
- 可编程源地址和目标地址模式
  - 固定地址
  - 后递增地址
  - 后递减地址
- 可编程源大小和目标大小
- 源指针和目标指针寄存器，动态更新且可重载
- 源计数和目标计数寄存器，动态更新且可重载
- 基于源计数器或目标计数器的可编程自动停止
- 软件触发传输
- 用于硬件触发传输的多个用户可选择源
- 用于中止DMA传输的多个用户可选择源

15.2 DMA寄存器

DMA模块的操作涉及以下寄存器：

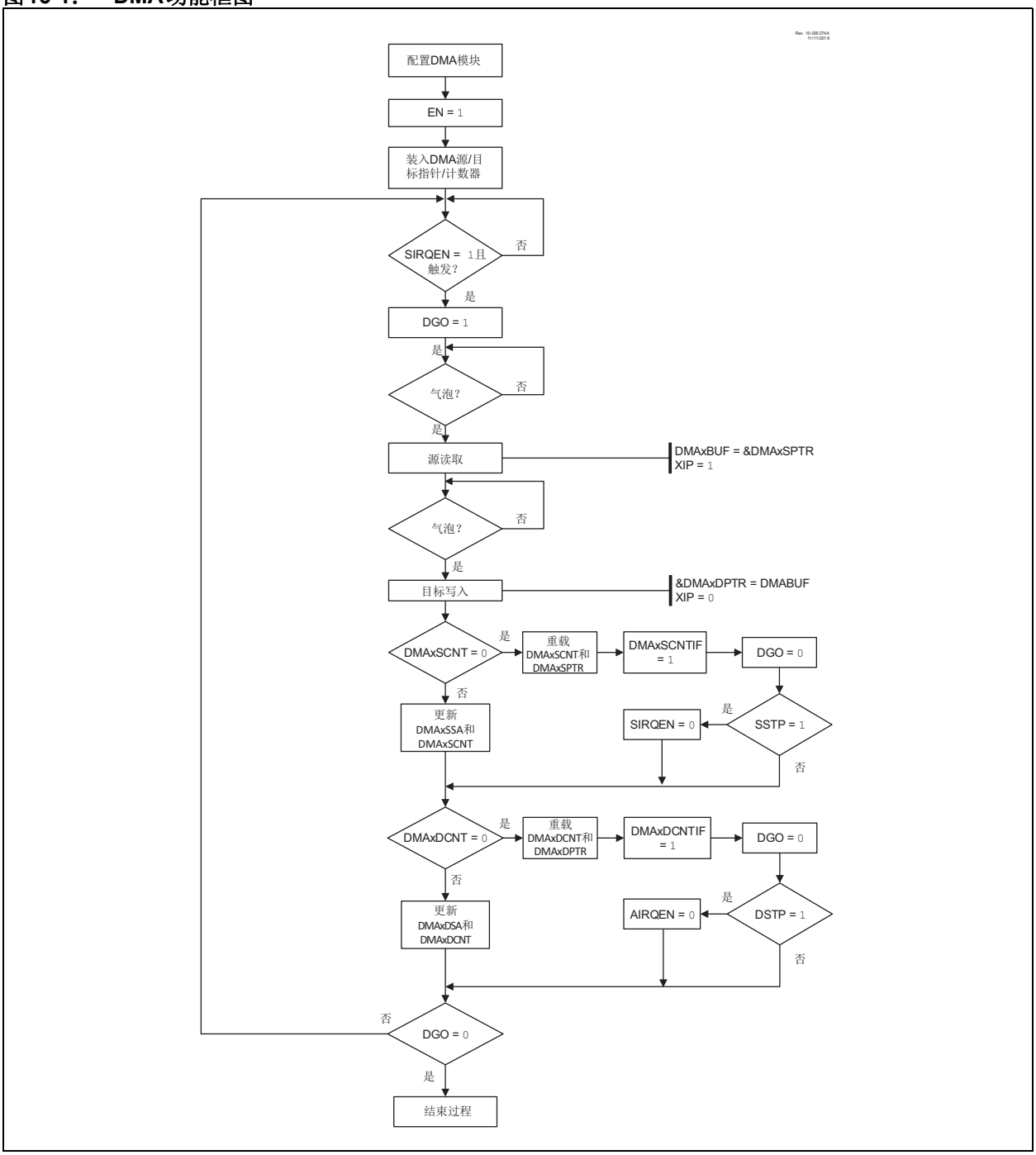
- 控制寄存器（DMAxCON0和DMAxCON1）
- 数据缓冲区寄存器（DMAxBUF）
- 源起始地址寄存器（DMAxSSAU:H:L）
- 源指针寄存器（DMAxSPTRU:H:L）
- 源报文大小寄存器（DMAxSSZH:L）
- 源计数寄存器（DMAxSCNTH:L）
- 目标起始地址寄存器（DMAxDSAH:L）
- 目标指针寄存器（DMAxDPTRH:L）
- 目标报文大小寄存器（DMAxDSZH:L）
- 目标计数寄存器（DMAxDCNTH:L）
- 启动中断请求源寄存器（DMAxSIRQ）
- 中止中断请求源寄存器（DMAxAIRQ）

第15.13节“寄存器定义：DMA”详细介绍了这些寄存器。

15.3 DMA构成

K42系列器件上的DMA模块旨在通过使用现有指令总线<16>和数据总线<8>来移动数据，而无需存储器或外设系统具备双端口（图15-1）。DMA可在系统仲裁器授予其权限时访问所需总线。

图15-1： DMA功能框图



根据DMA相对于CPU执行的优先级（有关更多信息，见**第3.2节“存储器访问机制”**），DMA控制器可通过以下两种方法来移动数据：

- 在完成传输之前，停止CPU执行（在该工作模式下，DMA的优先级高于CPU）
- 利用未使用的CPU周期进行DMA传输（在该工作模式下，CPU的优先级高于DMA）未使用的CPU周期被称为气泡，这些气泡是可供DMA执行读写操作的指令周期。通过这种方式，可增大用于处理数据的有效带宽；同时，DMA操作可以继续进行而不会造成处理器停止。

15.4 DMA接口

DMA模块将数据从源地址传输到目标地址时，一次传输一个字节，这种最小数据移动称为DMA数据事务。DMA报文指一个或多个DMA数据事务。

每个DMA数据事务包含两个单独的操作：

- 读取源地址存储器并将值存储在DMA缓冲区寄存器中
- 将DMA缓冲区寄存器的内容写入目标地址存储器

注： DMA数据移动是一个双周期操作。

XIP位（DMAxCON0寄存器）是一个状态位，用于指示DMAxBUF寄存器中的数据是否已写入目标地址。如果该位置1，则表示数据正在等待写入目标地址。如果该位清零，则表示数据已写入目标地址或未发生源地址读取操作。

DMA具有对PFM、数据EEPROM和SFR/GPR空间的读访问权限，以及对SFR/GPR空间的写访问权限。基于这些存储器访问功能，DMA可以支持以下存储器事务：

表15-1： DMA存储器访问

读取源地址	写入目标地址
闪存程序存储器	GPR
闪存程序存储器	SFR
数据EE	GPR
数据EE	SFR
GPR	GPR
SFR	GPR
GPR	SFR
SFR	SFR

即使DMA模块可以访问CPU也可使用的所有存储器和外设，但建议DMA不要访问系统仲裁所涉及的任何寄存器。DMA作为系统仲裁客户端时，不应由其自身或其他DMA实例来读写。

以下部分将介绍DMA数据传输所需的各种控制接口。

15.4.1 DMA寻址

分别使用DMAxSSA <21:0>和DMAxDSA <15:0>寄存器来设置源读操作和目标写操作的起始地址。

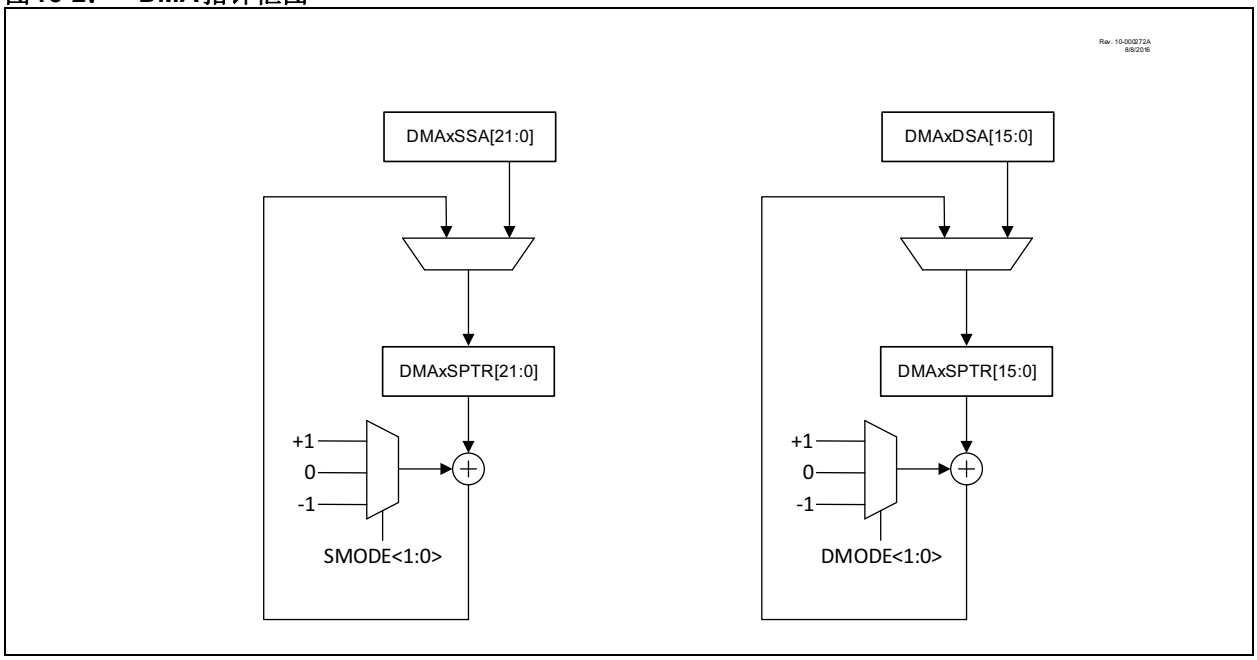
当DMA报文正在传输时，DMAxSPTR <21:0>和DMAxDPTR <15:0>寄存器包含每个源读操作和目标写操作的当前地址指针，这些寄存器将根据地址模式选择位在每个事务后进行修改。

DMAxCON1控制寄存器中的SMODE和DMODE位通过控制每个DMA数据事务组合后DMAxSPTR <21:0>和DMAxDPTR <15:0>位的更新方式来决定操作的寻址模式（图15-2）。

每个地址可单独配置为：

- 保持不变
- 递增1
- 递减1

图15-2： DMA指针框图



DMA可以启动来自PFM、数据EEPROM或SFR/GPR空间的数据传输。DMAxCON1寄存器中的SMR<1:0>位用于选择源地址指针正指向的存储器的类型。需要SMR<1.0>位，因为PFM和SFR/GPR空间具有重叠地址，这些地址不允许指定地址惟一定义要访问的存储单元。

注 1：	为了进行适当的存储器读访问，地址和空间选择的组合必须有效。
2：	目标不具有空间选择位，因为它只能写入SFR/GPR空间。

15.4.2 DMA报文大小/计数器

一个字节的传输即为事务。一条报文由一个或多个事务组成。整个DMA过程由一条或多条报文组成。大小寄存器决定一条报文中有多少个事务。DMAxSSZ寄存器决定源大小，DMAxDSZ寄存器决定目标大小。

启动DMA传输时，将大小寄存器的内容复制到控制报文持续时间的相应计数器寄存器。DMAxSCNT寄存器计数源事务，DMAxDCNT寄存器计数目标事务。这两个寄存器在每个事务后同时递减1。

报文通过将DMAxCON0寄存器的DGO位置1来启动，在两个计数器中较小的一个达到0时终止。

当其中任一计数器达到0时，DGO位清零，计数器和指针寄存器立即重载相应大小和地址数据。如果另一个计数器未达到0，则下一条报文会继续使用相应寄存器对应的计数和地址。

当源大小和目标大小寄存器不相等时，最大大小与最小大小的比值决定DMA过程中有多少条报文。例如，当目标大小为6且源大小为2时，每条报文将由两个事务组成，整个DMA过程将由三条报文组成。当较大大小不是较小大小的偶数倍时，过程中的最后一条报文将在较大计数达到0时提前终止。在这种情况下，较大的计数器将复位，较小的计数器将有一个余数，使后续报文偏移相应量。

注：	读取DMAxSCNT或DMAxDCNT寄存器始终不会返回0。当任一寄存器从1开始递减时，它将立即从相应大小寄存器重载。

图15-3： DMA计数器框图

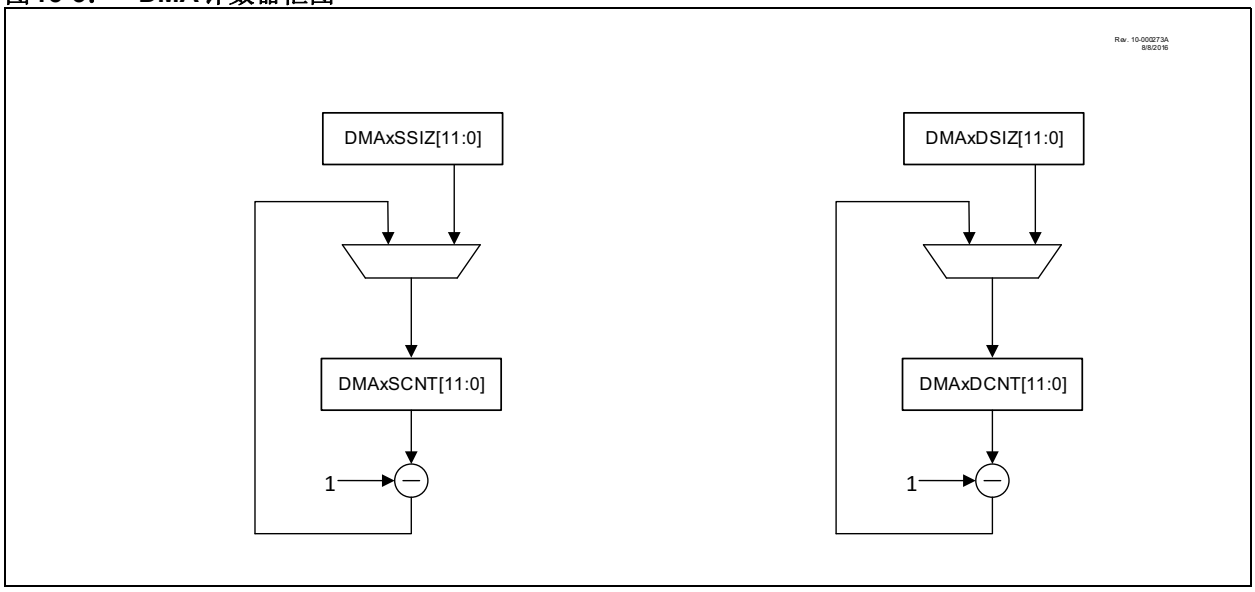


表15-2提供了配置DMA报文大小的几个示例。

表15-2： 示例报文大小表

操作	示例	SCNT	DCNT	备注
将单个SFR单元的内容读入RAM	U1RXB	1	N	N等于目标缓冲区中所需的字节数。N >= 1。
将RAM的内容写入单个SFR单元	U1TXB	N	1	N等于源缓冲区中所需的字节数。N >= 1。
读取多个SFR单元	ADRES[H:L]	2	2*N	N等于要存储在存储器中的ADC结果数。N >= 1
	TMR1[H:L]	2	2*N	N等于要存储在存储器中的TMR1采集结果数。N >= 1
	SMT1CPR[U:H:L]	3	3*N	N等于要存储在存储器中的捕捉脉宽测量结果数。N >= 1
写入多个SFR寄存器	PWMDC[H:L]	2*N	2	N等于从存储器表中装入的PWM占空比值的数量。N >= 1
	所有ADC寄存器	N*31	31	使用DMA将完整的ADC现场从RAM传输到ADC寄存器。N >= 1

15.5 DMA报文传输

通过将使能位置1来启动DMA报文传输后，源/目标指针和计数器寄存器将初始化为表15-3中所示的条件。

表15-3： DMA初始条件

寄存器	装入的值
DMAxSPTR<21:0>	DMAxSSA<21:0>
DMAxSCNT<11:0>	DMAxSSZ<11:0>
DMAxDPTR<15:0>	DMAxDSA<15:0>
DMAxDCNT<11:0>	DMAxDSZ<11:0>

表15-4和表15-5列出了在每个事务后的DMA操作期间，如何修改源/目标指针和计数器寄存器

表15-4： 操作期间的DMA源指针/计数器

寄存器	修改后的源计数器/指针值
DMAxSCNT<11:0> != 1	DMAxSCNT = DMAxSCNT -1
	SMODE = 00： DMAxSPTR = DMAxSPTR
	SMODE = 01： DMAxSPTR = DMAxSPTR + 1
	SMODE = 10： DMAxSPTR = DMAxSPTR - 1
DMAxSCNT<11:0> == 1	DMAxSCNT = DMAxSSZ
	DMAxSPTR = DMAxSSA

表15-5： 操作期间的DMA目标指针/计数器

寄存器	修改后的目标计数器/指针值
DMAxDCNT<11:0>!= 1	DMAxDCNT = DMAxDCNT -1
	DMODE = 00： DMAxDPTR = DMAxDPTR
	DMODE = 01： DMAxDPTR = DMAxDPTR + 1
	DMODE = 10： DMAxDPTR = DMAxDPTR - 1
DMAxDCNT<11:0> == 1	DMAxDCNT = DMAxDSZ
	DMAxDPTR = DMAxDSA

以下部分将介绍如何启动和终止DMA传输。

15.5.1 启动DMA报文传输

DMA可以通过以下两个条件之一来启动数据事务：

1. 用户软件控制
2. 硬件触发，SIRQ

15.5.1.1 用户软件控制

软件通过置1/清零DGO位来启动或停止DMA事务。DGO位也用于指示DMA硬件触发信号是否已被接收以及报文传输是否正在进行。

注 1： 软件启动方法仅适用于EN位（DMAxCON1）置1时。

2： 如果CPU在DGO位已置1时写入该位，则对系统没有影响，DMA将继续正常工作。

15.5.1.2 硬件触发，SIRQ

硬件触发信号是从另一个模块发送到DMA的中断请求，用于启动DMA报文。用户可以使用DMAxSIRQ寄存器来选择DMA启动触发源。

SIRQEN位（DMAxCON0寄存器）用于使能可以启动DMA传输的外部中断触发信号的采样。该位置1时，DMA将采样所选中断源，清零时，DMA将忽略所选中断源。清零SIRQEN不会停止当前正在进行的DMA事务，仅会停止接收更多的硬件请求信号。

15.5.2 停止DMA报文传输

DMA控制器可以通过以下两个条件之一来停止数据事务：

1. 清零DGO位
2. 硬件触发，AIRQ
3. 源计数重载
4. 目标计数重载
5. 清零使能位

15.5.2.1 用户软件控制

如果用户清零DGO位，报文将停止，DMA将保持当前配置。

例如，如果用户在源数据读取之后、写入目标地址之前清零DGO位，则DMAxBUF中的数据将不会到达其目标地址。

这也称为软停止，因为需要时，可通过将DGO位再次置1来恢复操作。

15.5.2.2 硬件触发，AIRQ

AIRQEN位（DMAxCON0寄存器）用于使能可以中止DMA事务的外部中断触发信号的采样。

接收到中止中断请求后，DMA将通过清零DGO位以及清零SIRQEN位来执行软停止，这样就不会发生溢出。AIRQEN位也会清零，以防止额外的中止信号触发虚假中止。

如果需要，可再次将DGO位置1，DMA将从软停止发生后操作停止的位置恢复操作，因为DMA状态信息在中止事件中未发生改变。

15.5.2.3 源计数重载

当源计数寄存器从1开始递减并重载时（即，出现了源读取或目标写入操作的最后一个字节），则视为DMA报文完成。当SSTP位置1（DMAxCON1寄存器）且重载源计数寄存器时，将停止其他报文传输。

15.5.2.4 目标计数重载

当目标计数寄存器从1开始递减并重载时（即，出现了源读取或目标写入操作的最后一个字节），则视为DMA报文完成。当DSTP位置1（DMAxCON1）且重载目标计数寄存器时，将停止其他报文传输。

注：	读取DMAxSCNT或DMAxDCNT寄存器始终不会返回0。当任一寄存器从1开始递减时，它将立即从相应大小寄存器重载。

15.5.2.5 清零使能位

如果用户清零EN位，报文将停止，DMA将恢复其默认配置。这也称为硬停止，因为DMA无法从操作停止的位置恢复操作。

注：	DMA报文传输停止后，在停止条件生效之前需要一个额外的指令周期。因此，在发生停止条件后，可能会发生源读取或目标写入操作，具体取决于源总线或目标总线的可用性。

15.5.3 完成后禁止DMA报文传输

DMA报文完成后，可能需要禁止触发源以防止数据上溢或下溢。这可通过以下方法之一完成：

1. 清零SIRQEN位
2. 将SSTP位置1
3. 将DSTP位置1

15.5.3.1 清零SIRQEN位

清零SIRQEN位（DMAxCON1寄存器）会停止外部启动中断触发信号的采样，从而阻止其他DMA报文传输。

具有电平触发中断的通信外设就是一个例子。即使没有更多的数据要移动，外设也将继续请求数据（因为其缓冲区为空）。禁止SIRQEN位将防止DMA处理这些请求

15.5.3.2 源/目标停止

SSTP和DSTP位（DMAxCON0寄存器）决定是否在DMA报文完成后禁止硬件触发（SIRQEN = 0）。

当SSTP位置1且DMAxSCNT = 0时，SIRQEN位将清零。类似地，当DSTP位置1且DMAxDCNT = 0时，SIRQEN位将清零。

注：	SSTP和DSTP位都是独立功能，互不依赖。对于一条报文而言，可通过任一计数器在报文结束时停止或通过两个计数器在报文结束时停止。

15.6 硬件触发类型

DMA有两个不同的触发输入，即源触发和中止触发。用户可以使用DMAxSIRQ和DMAxAIRQ寄存器来配置这两个触发源中的每一个。

根据为每个触发选择的触发源，有两种类型的请求可以发送到DMA。

- 边沿触发
- 电平触发

15.6.1 边沿触发请求

边沿请求仅在给定的模块中断要求为真时产生一次。

15.6.2 电平触发请求

只要导致中断的条件为真，就会产生一个电平请求。

15.7 数据传输类型

基于DMA的存储器访问功能（见表15-1），以下部分将介绍源存储区和目标存储区之间不同类型的数据移动。

- N: 1

该传输类型在通过单个接口点（如通信模块发送寄存器）发送预定义数据包（如字符串）时很常用。

- N: N

该传输类型在将信息从程序闪存或数据EEPROM移出到SRAM以供CPU或其他外设处理时非常有用。

- 1: N

该传输类型在将两个不同的模块数据流桥接在一起（通信桥）时很常用。

- 1: N

该传输类型在将信息从单个数据源移动到存储器缓冲区（通信接收寄存器）时非常有用。

15.8 DMA中断

每个DMA都有自己的一组（四个）中断标志，用于指示数据传输期间的各种情况。可以使用相应PIR寄存器访问中断标志位（见“中断”部分）。

15.8.1 DMA源计数中断

每次DMAxSCNT<11:0>达到0并重载其起始值时，DMAxSCNTIF源计数中断标志位都会置1。

15.8.2 DMA目标计数中断

每次DMAxDCNT<11:0>达到0并重载其起始值时，DMAxDCNTIF目标计数中断标志位都会置1。

DMA源计数零中断与目标计数零中断结合使用，以确定在DMA报文完成后何时向CPU发出信号。

15.8.3 中止中断

DMAxAIF中止中断标志位用于指示DMA因来自中止源之一的中止信号而暂停活动。该位用于指示事务因某种原因而暂停。

15.8.4 溢出中断

当DMA在当前报文完成之前接收到启动新报文的触发信号时，DMAxORIF溢出中断标志位将置1。

该条件表示DMA在当前事务完成之前被请求。这意味着工作的DMA可能无法满足正在处理的外设模块的要求，从而导致数据丢失。

DMAxORIF标志位置1不会导致当前DMA传输终止。

溢出中断仅适用于基于边沿的触发源，不适用于基于电平的触发源。因此，基于电平的中断源不会因系统中潜在的延时问题而触发DMA溢出错误。

定时器溢出（或周期匹配）中断就是可以使用溢出中断的一个中断示例。该事件仅在定时器每次计满返回时发生，不依赖于任何其他系统条件。

UARTTX缓冲区就是不允许溢出中断的一个中断示例。在DMA能够处理MSG之前，UART会继续将中断置为有效。由于延时问题，DMA可能无法立即处理空缓冲区，但在处理空缓冲区之前，UART会继续将其发送中断置为有效。如果在这种情况下允许溢出，则模块在每个指令周期采样中断源时，溢出几乎会立即发生。

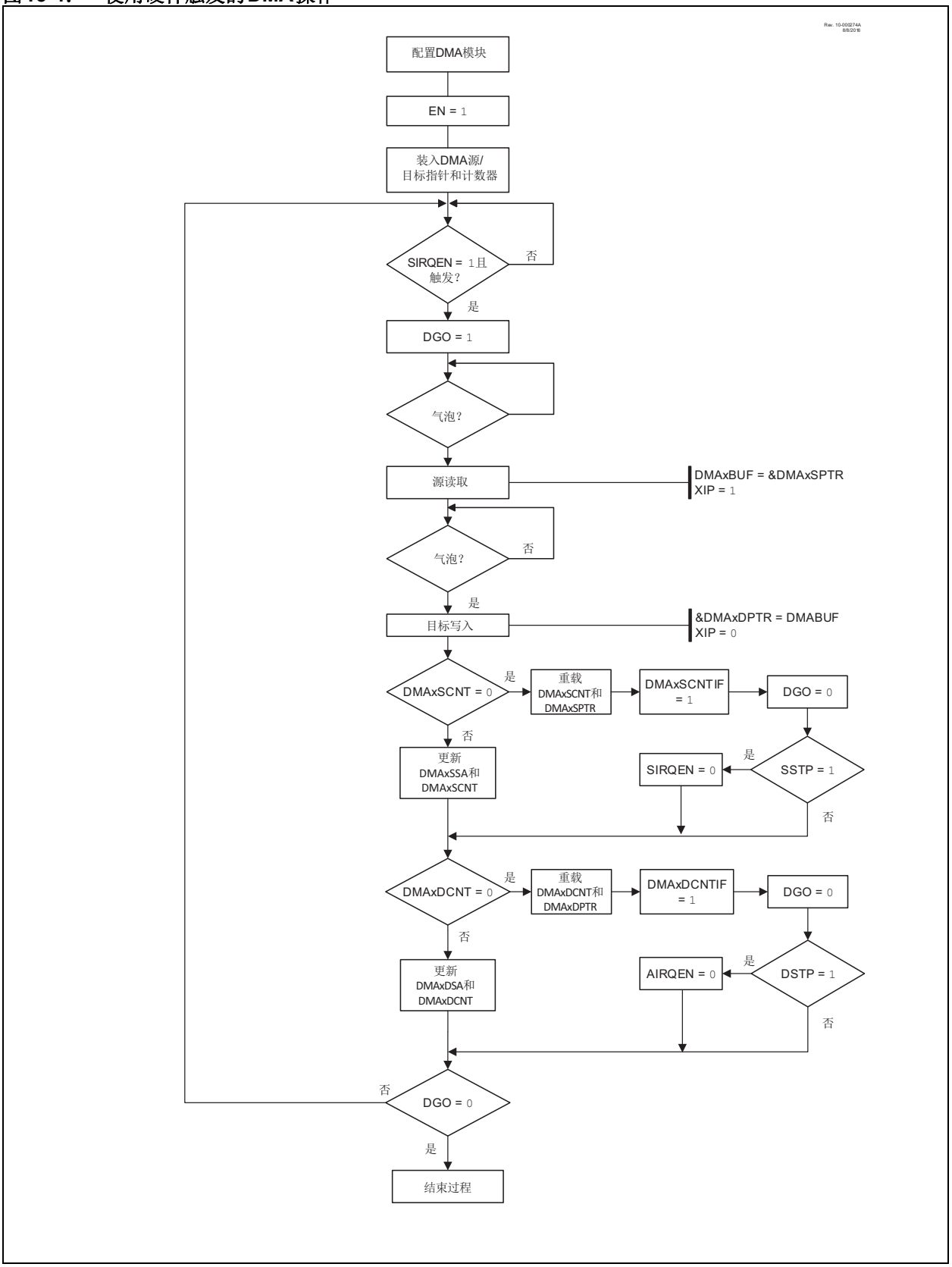
15.9 DMA设置和操作

以下步骤说明了如何配置DMA以进行数据传输：

1. 将事务的适当源地址和目标地址编程到DMAxSSA和DMAxDSA寄存器中
2. 使用SMR<1:0>位选择通过DMAxSSA寄存器寻址的源存储区。
3. 对SMODE和DMODE位进行编程，以选择寻址模式。
4. 使用要传输的字节数编程源大小DMAxSSZ和目标大小DMAxDSZ寄存器。建议采取大小寄存器互为彼此倍数的适当操作。
5. 如果用户希望在报文完成后禁止数据传输，则需要将DMAxCON0寄存器中的SSTP和DSTP位置1（见**第15.5.3.2节“源/目标停止”**）。
6. 如果使用硬件触发进行数据传输，请设置用于启动和中止DMA传输的硬件触发中断源（DMAxSIRQ和DMAxAIRQ），并将相应的中断请求使能位（SIRQEN和AIRQEN）置1。
7. 为DMA选择优先级（见**第3.1节“系统仲裁”**）并锁定优先级（见**第3.1.1节“优先级锁定”**）
8. 使能DMA（DMAxCON1bits.EN = 1）
9. 如果使用软件控制进行数据传输，则将DGO位置1，否则该位将由硬件触发置1。

设置DMA后，以下流程图将说明DMA使用硬件触发并利用未使用的CPU周期（气泡）进行DMA传输时的操作序列。

图15-4： 使用硬件触发的DMA操作

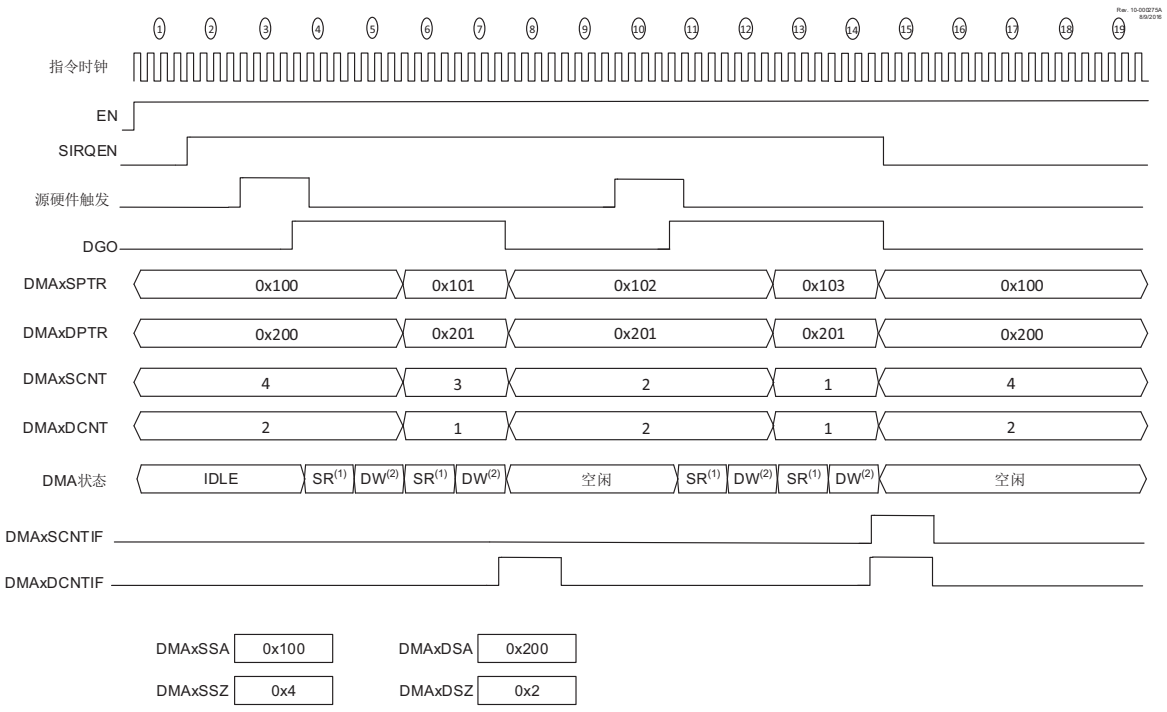


以下部分将用直观的方式对DMA模块不同配置的事件序列进行说明

15.9.1 源停止

当源停止位置1（SSTP = 1）且DMAxSCNT寄存器重载时，DMA会清零SIRQEN位以停止接收新的启动中断请求信号并将DMAxSCNTIF标志位置1。

图15-5： 使用硬件触发的GPR-GPR事务（SSTP = 1）



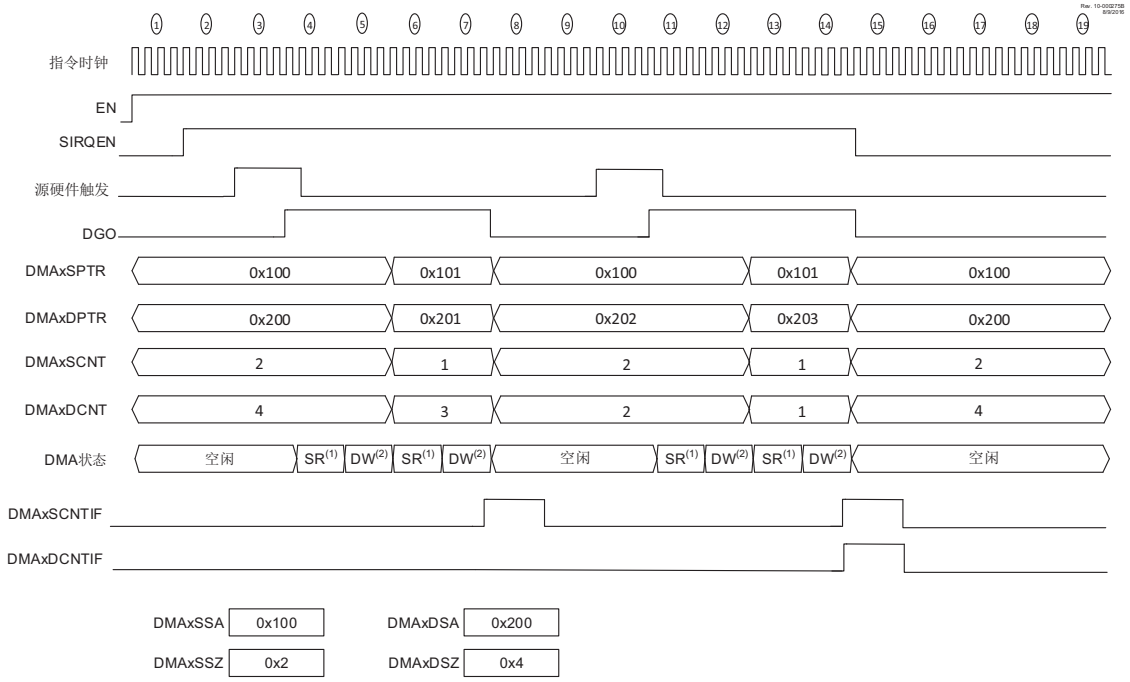
注 1： SR——源读取

2： DW——目标写入

15.9.2 目标停止

当目标停止位置1（DSTP = 1）且DMAxDCNT寄存器重载时，DMA会清零SIRQEN位以停止接收新的启动中断请求信号并将DMAxDCNTIF标志位置1。

图15-6： 使用硬件触发的GPR-GPR事务（DSTP = 1）



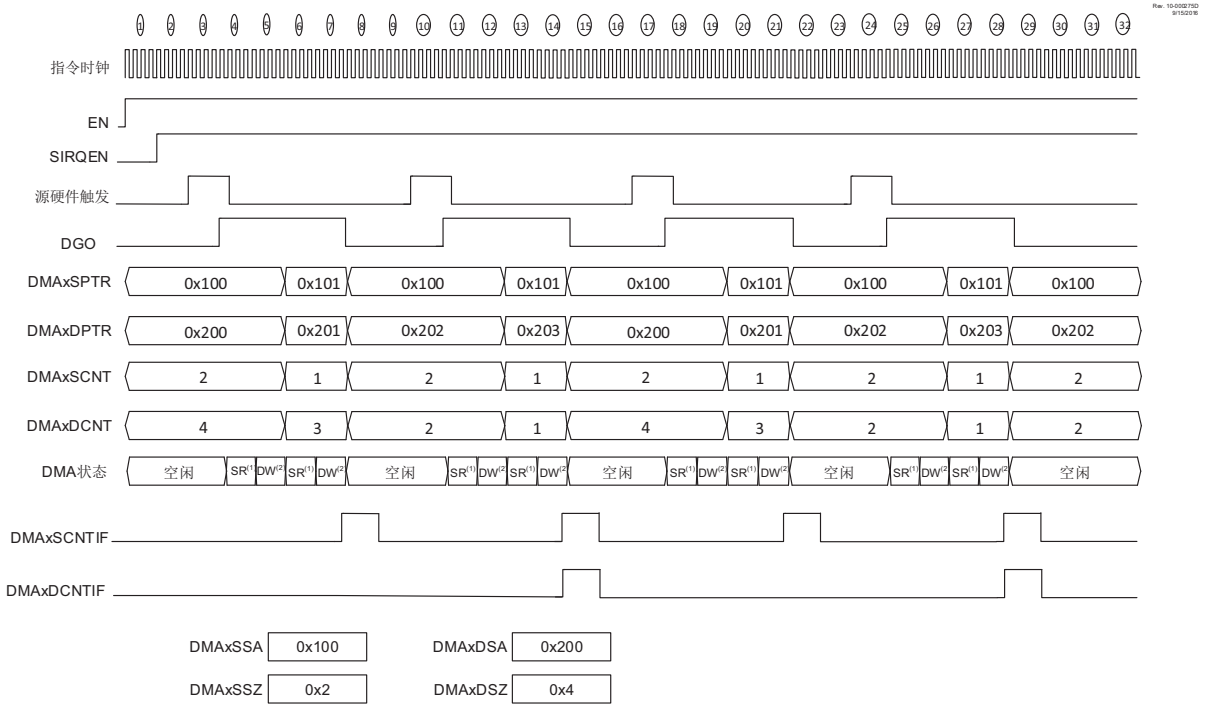
注 1： SR——源读取

2： DW——目标写入

15.9.3 连续传输

当源停止位或目标停止位清零（SSTP和DSTP = 0）时，除非用户清零，否则事务将继续传输。每当重载相应计数器寄存器时，DMAxSCNTIF和DMAxDCNTIF标志位都将置1。

图15-7： 使用硬件触发的GPR-GPR事务（SSTP和DSTP = 0）



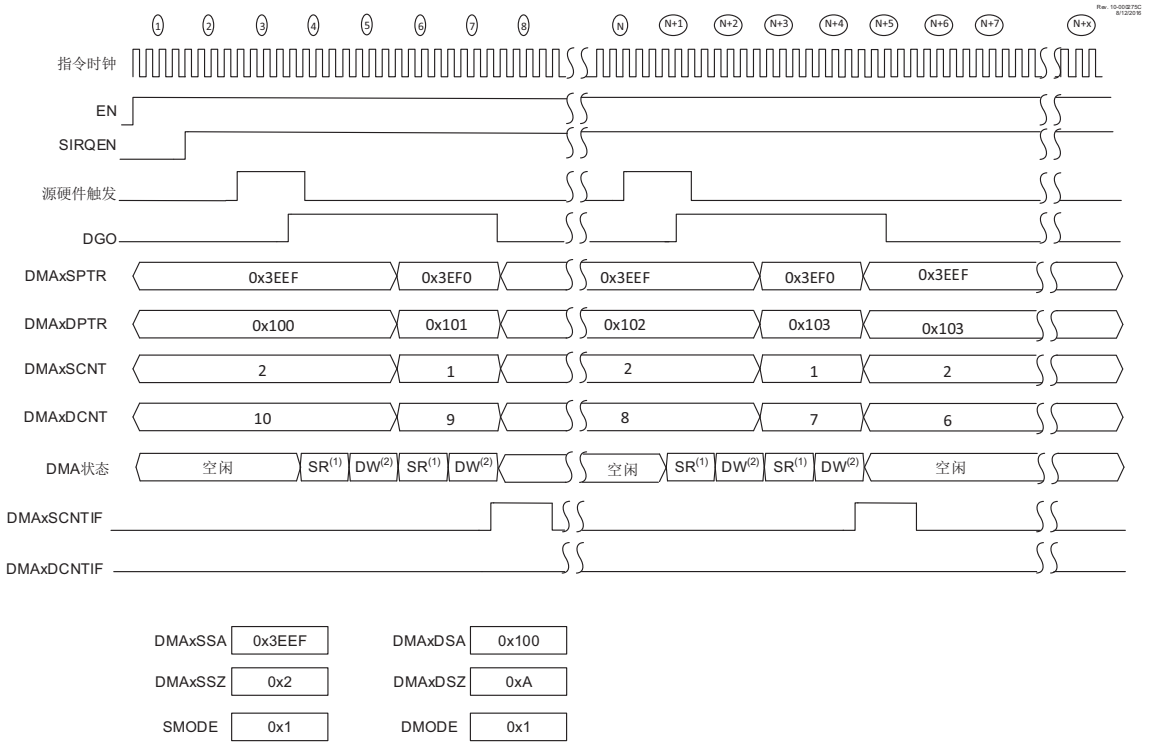
注 1： SR——源读取

2： DW——目标写入

15.9.4 从SFR传输到GPR

以下部分将用直观的方式对将ADC结果复制到GPR单元时的事件序列进行说明。ADC中断标志可选为源硬件触发，可以将源地址设置为指向3EEF处的ADC结果寄存器，将目标地址设置为指向所选的任何GPR单元（例如0x100）。

图15-8： SFR空间到GPR空间的传输



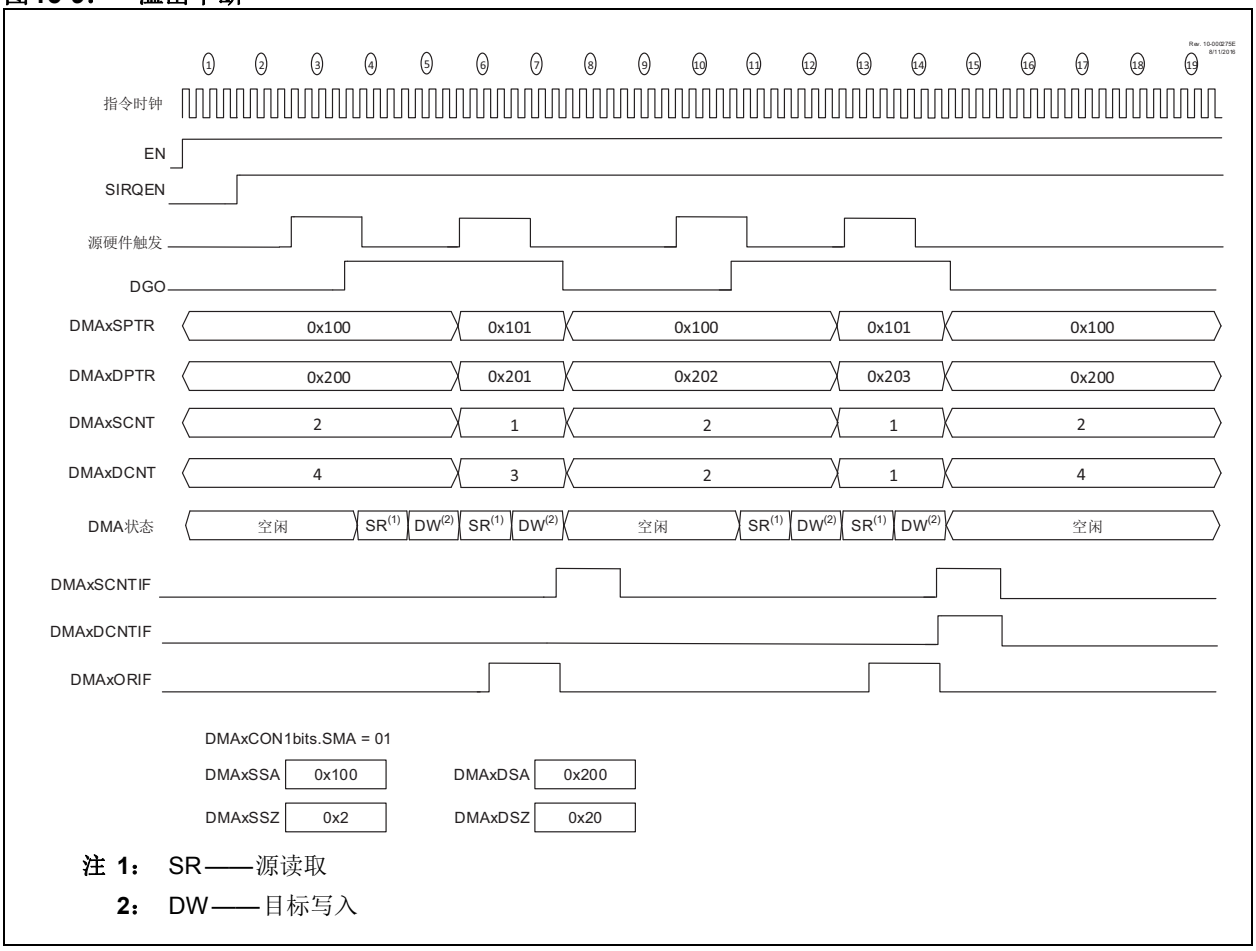
注 1： SR——源读取

2： DW——目标写入

15.9.5 溢出中断

如果DMA在当前报文完成之前接收到启动新报文的触发信号，则溢出中断标志位置1。

图15-9： 溢出中断



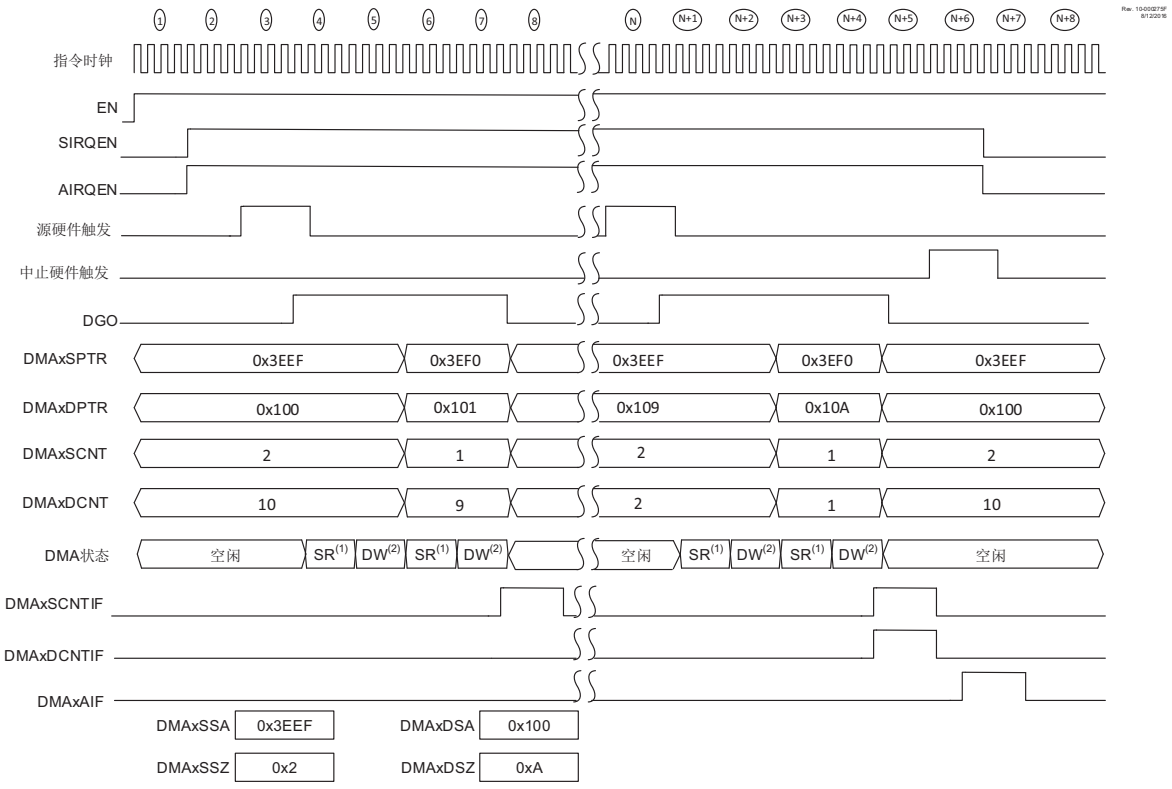
注 1： SR——源读取

2： DW——目标写入

15.9.6 中止触发，报文完成

需要将AIRQEN置1，以便DMA采样中止中断源。接收到中止中断信号后，SIRQEN位和AIRQEN位清零，以避免接收其他中止触发信号。

图15-10： 在报文结束时中止



注 1： SR——源读取

2： DW——目标写入

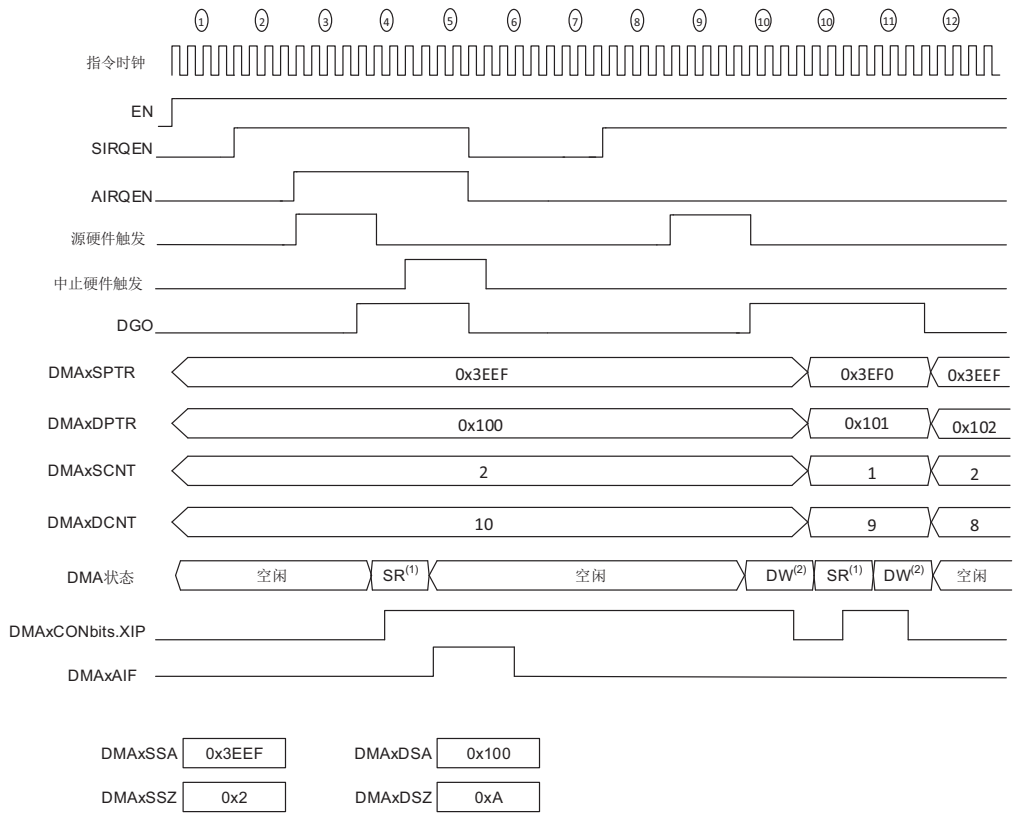
15.9.7 中止触发，报文正在进行

在DMA事务之间接收到中止中断请求后，DMA将通过清零DGO位来执行软停止（即，如果DMA正在读取源寄存器，它将完成读操作，然后清零DGO位）

SIREQEN位清零可防止发生溢出，AIRQEN位清零可防止发生虚假中止。

当DGO位再次置1时，DMA将从软停止后操作停止的位置恢复操作。

图15-11： 在报文传输期间中止



注 1： SR——源读取

2： DW——目标写入

下表包含可以配置DMA模块的一些情况。

表15-6： 示例DMA用例表

源模块	源寄存器	目标模块	目标寄存器	DCHxSIRQ	备注
信号测量定时器（SMT）	SMTxCPW[U:H:L]	GPR	GPR[x,y,z]	SMTxPWAIF	存储捕捉的脉宽值
	SMTxCPR[U:H:L]			SMTxPRAIF	存储捕捉的周期值
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x,y]	TMR0	TMR0[H:L]	TMR0IF	用作自定义16位值的Timer0重载功能
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x]	TMR0	PR0	任何	根据特定触发更新TMR0频率
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x,y]	TMR1	TMR1[H:L]	TMR1IF	用作自定义16位值的Timer1重载功能
TMR1	TMR1[H:L]	GPR	GPR[x,y]	TMR1GIF	使用TMR1门控中断标志从TMR1寄存器中读取数据
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x]	TMR2	PR2	TMR2IF	
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x,y,z]	TMR2 CCP或PWM	PR2 CCPR[H:L]或PWMDC[H:L]	任何	频率发生器（占空比为50%）查找表
CCP	CCPR[H:L]	GPR	GPR[x,y]	CCPxIF	从CCP 16b捕捉中移动数据
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x,y]	CCP	CCPR[H:L]	任何	将比较值或PWM值装入CCP
GPR/SFR/程序闪存/数据EEPROM	MEMORY [x,y,z,u,v,w]	CCPx CCPy CCPz	CCPxR[H:L] CCPyR[H:L] CCPzR[H:L]	任何	同时更新多个PWM值（例如，三相电机控制）
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x,y,z]	NCO	NCOxINC[U:H:L]	任何	频率发生器查找表
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x]	DAC	DACxCON0	任何	更新DAC值
GPR/SFR/程序闪存/数据EEPROM	MEMORY[x]	OSCTUNE	OSCTUNE	任何	自动频率抖动

15.10 复位

任何复位时，DMA寄存器都会设置为默认状态。当使能位清零（DMA1CON1bits.EN=0）时，寄存器也会复位为默认状态。

15.11 节能工作模式

DMA利用系统时钟，在进入节能工作模式时被视为外设。与其他外设类似，DMA也使用外设模块禁止位来进一步定制其在低功耗状态下的操作。

15.11.1 休眠模式

器件进入休眠模式时，模块的系统时钟将关闭，因此休眠模式下不支持任何DMA操作。禁止系统时钟后，还会禁止必要的读写时钟，没有这些时钟，DMA无法执行任何任务。

退出休眠模式后，将恢复所有正在进行的传输。寄存器内容不受器件进入或退出休眠模式的影响。建议允许DMA事务在进入休眠模式之前完成。

15.11.2 空闲模式

在空闲模式下，所有系统时钟（包括读写时钟）仍在工作，但CPU不会使用系统时钟以实现节能。

因此，每个指令周期都可用于系统仲裁器，如果将气泡授予DMA，则可以利用气泡来移动数据。

15.11.3 打盹模式

与空闲模式类似，CPU不会使用所有可用的指令周期时隙以实现节能，而仅会根据打盹设置中的相应设置执行指令。

因此，CPU未使用的每条指令都可用于系统仲裁，如果由仲裁器授予，可供DMA使用。

15.11.4 外设模块禁止

外设模块禁止（PMD）寄存器通过对所有提供给DMA的时钟源进行门控来提供一种禁止DMA的方法。需要将相应的DMAxMD位置1以禁止DMA。

15.12 DMA寄存器接口

DMA可以将数据传输到任何GPR或SFR存储单元。为优化用户可访问性，某些更常用的SFR存储空间将其镜像寄存器放置在Bank 64（0x4000-0x40FF）中，这些镜像寄存器只能通过DMA源地址和目标地址寄存器访问。

例15-1： 设置DMA1以使用硬件触发将数据从闪存程序存储器移动到UART1发送缓冲区

```
//This code example illustrates using DMA1 to transfer
//10 bytes of data from 0x1000 in PFM to U1TXB 0x3DEA

void main() {
    //System Initialize
    initializeSystem();

    //Setup UART1
    initializeUART1();

    //Setup DMA1
    //DMA1CON1 - DPTR remains, Source Memory Region PFM, SPTR increments, SSTP
    DMA1CON1 = 0x0B;

    //Source registers
    //Source size
    DMA1SSZH = 0x00;
    DMA1SSZL = 0x0A;

    //Source start address, 0x1000
    DMA1SSAU = 0x00;
    DMA1SSAH = 0x10;
    DMA1SSAL = 0x00;

    //Destination registers
    //Destination size
    DMA1DSZH = 0x00;
    DMA1DSZL = 0x01;

    //Destination start address, 0x3DEA
    DMA1DSAH = 0x3D;
    DMA1DSAL = 0xEA;

    //Start trigger source U1TX
    DMA1SIRQ = 0x1C;

    //Enable & Start DMA transfer
    DMA1CON0 = 0xC0;

    while (1) {
        doSomething();
    }
}
```

15.13 寄存器定义：DMA

表15-7列出了DMA外设的长位名称前缀。更多信息，请参见**第1.3节“寄存器和位命名约定”**。

表15-7： 寄存器和位名称

外设	位名称前缀
DMA 1	DMA1
DMA 2	DMA2

寄存器15-1： DMAxCON0：DMAx控制寄存器0

R/W-0/0	R/W/HC-0/0	R/W/HS/HC-0/0	U-0	U-0	R/W/HC-0/0	U-0	R/HS/HC-0/0
EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值		0 = 清零	x = 未知 u = 不变

bit 7 **EN**：DMA模块使能位
1 = 使能模块
0 = 禁止模块

bit 6 **SIRQEN**：启动传输中断请求使能位
1 = 允许通过硬件触发来启动DMA传输
0 = 不允许通过硬件触发来启动DMA传输

bit 5 **DGO**：DMA事务位
1 = 正在进行DMA事务
0 = 未在进行DMA事务

bit 4-3 **未实现**：读为0

bit 2 **AIRQEN**：中止传输中断请求使能位
1 = 允许通过硬件触发来中止DMA传输
0 = 不允许通过硬件触发来中止DMA传输

bit 1 **未实现**：读为0

bit 0 **XIP**：传输进行状态位
1 = DMAxBUF寄存器当前保存读操作的内容，未将数据传输到目标地址
0 = DMAxBUF寄存器为空，或已成功将数据传输到目标地址

寄存器15-2： DMAxCON1：DMAx控制寄存器1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
DMODE<1:0>		DSTP	SMR<1:0>		SMODE<1:0>		SSTP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值

bit 7-6 **DMODE<1:0>**：目标地址模式选择位

11 = 保留，不要使用
10 = 每次传输完成后，DMAxDPTR<15:0>递减
01 = 每次传输完成后，DMAxDPTR<15:0>递增
00 = 每次传输完成后，DMAxDPTR<15:0>保持不变

bit 5 **DSTP**：目标计数器重载停止位

1 = 目标计数器重载后，SIRQEN位清零
0 = 目标计数器重载后，SIRQEN位不清零

bit 4-3 **SMR[1:0]**：源存储区选择位

1x = DMAxSSA<21:0>指向数据EEPROM
01 = DMAxSSA<21:0>指向闪存程序存储器
00 = DMAxSSA<21:0>指向SFR/GPR数据空间

bit 2-1 **SMODE[1:0]**：源地址模式选择位

11 = 保留，不要使用
10 = 每次传输完成后，DMAxSPTR<21:0>递减
01 = 每次传输完成后，DMAxSPTR<21:0>递增
00 = 每次传输完成后，DMAxSPTR<21:0>保持不变

bit 0 **SSTP**：源计数器重载停止位

1 = 源计数器重载后，SIRQEN位清零
0 = 源计数器重载后，SIRQEN位不清零

寄存器15-3： DMAxBUF：DMAx数据缓冲区寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
BUF7	BUF6	BUF5	BUF4	BUF3	BUF2	BUF1	BUF0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **BUF<7:0>**：DMA内部数据缓冲区位
DMABUF<7:0>
这些位反映了内部数据缓冲区的内容，DMA外设使用内部数据缓冲区保存从源地址移动到目标地址的数据。

寄存器15-4： DMAxSSAL：DMAx源起始地址低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SSA<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **SSA<7:0>**：源起始地址位

寄存器15-5： DMAxSSAH：DMAx源起始地址高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SSA<15:8>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **SSA<15:8>**：源起始地址位

寄存器15-6： DMAxSSAU：DMAx源起始地址最高字节寄存器

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	SSA<21:16>					
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **SSA<21:16>**：源起始地址位

寄存器15-7： DMAxSPTRL：DMAx源指针低字节寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
SPTR<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 15-0 **SPTR<7:0>**：当前源地址指针

寄存器15-8： DMAxSPTRH：DMAx源指针高字节寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
SPTR<15:8>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 5-0 **SPTR<15:8>**：当前源地址指针

寄存器15-9： DMAxSPTRU：DMAx源指针最高字节寄存器

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
—	—	SPTR<21:16>					
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-6 **未实现：**读为0

bit 5-0 **SPTR<21:16>：**当前源地址指针

寄存器15-10： DMAxSSZL：DMAx源大小低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SSZ<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **SSZ<7:0>：**源报文大小位

寄存器15-11： DMAxSSZH：DMAx源大小高字节寄存器

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	SSZ<11:8>			
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-4 **未实现：**读为0

bit 3-0 **SSZ<11:8>：**源报文大小位

寄存器15-12： DMAxSCNTL：DMAx源计数低字节寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
SCNT<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **SCNT<7:0>**：当前源字节计数

寄存器15-13： DMAxSCNTH：DMAx源计数高字节寄存器

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	SCNT<11:8>			
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-4 **未实现**：读为0

bit 3-0 **SCNT<11:8>**：当前源字节计数

寄存器15-14： DMAxDSAL：DMAx目标起始地址低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
DSA<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **DSA<7:0>**：目标起始地址位

寄存器15-15： DMAxDSAH：DMAx目标起始地址高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
DSA<15:8>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **DSA<15:8>：**目标起始地址位

寄存器15-16： DMAxDPTRL：DMAx目标指针低字节寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DPTR<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **DPTR<7:0>：**当前目标地址指针

寄存器15-17： DMAxDPTRH：DMAx目标指针高字节寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DPTR<15:8>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **DPTR<15:8>：**当前目标地址指针

寄存器15-18： DMAxDSZL：DMAx目标大小低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
DSZ<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **DSZ<7:0>：**目标报文大小位

寄存器15-19： DMAxDSZH：DMAx目标大小高字节寄存器

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	DSZ<11:8>			
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-4 **未实现：**读为0

bit 3-0 **DSZ<11:8>：**目标报文大小位

寄存器15-20： DMAxDCNTL：DMAx目标计数低字节寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DCNT<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-0 **DCNT<7:0>：**当前目标字节计数

寄存器15-21： DMAxDCNTH：DMAx目标计数高字节寄存器

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	DCNT<11:8>			
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7-4 **未实现：**读为0

bit 3-0 **DCNT<11:8>：**当前目标字节计数

寄存器15-22： DMAxSIRQ：DMAx启动中断请求源选择寄存器

U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	SIRQ<6:0>						
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7 **未实现：**读为0

bit 6-0 **SIRQ<6:0>：**DMAx启动中断请求源选择位
更多信息，请参见表15-2。

寄存器15-23： DMAxAIRQ：DMAx中止中断请求源选择寄存器

U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	AIRQ<6:0>						
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n/n = POR和BOR时的值/所有其他复位时的值	1 = 置1	0 = 清零	x = 未知 u = 不变

bit 7 **未实现：**读为0

bit 6-0 **AIRQ<6:0>：**DMAx中断请求源选择位
更多信息，请参见表15-2。

表15-2： DMAxSIRQ和DMAxAIRQ中断源

DMAxSIRQ DMAxAIRQ	触发源	电平触发
0	保留	
1	LVD	否
2	OSF	否
3	CSW	否
4	NVM	否
5	SCAN	否
6	CRC	否
7	IOC	是
8	INT0	否
9	ZCD	否
10	AD	否
11	ADT	否
12	CMP1	否
13	SMT1	否
14	SMT1PRA	否
15	SMT1PWA	否
16	DMA1SCNT	否
17	DMA1DCNT	否
18	DMA1OR	否
19	DMA1A	否
20	SPI1RX	是
21	SPI1TX	是
22	SPI1	是
23	I2C1RX	是
24	I2C1TX	是
25	I2C1	是
26	I2C1E	是
27	U1RX	是
28	U1TX	是
29	U1E	是
30	U1	否
31	TMR0	否
32	TMR1	否
33	TMR1G	否
34	TMR2	否
35	CCP1	否
36	保留	
37	NCO	否
38	CWG1	否
39	CLC1	否
40	INT1	否
41	CMP2	否
42	DMA2SCNT	否
43	DMA2DCNT	否
44	DMA2OR	否
45	DMA2A	否
46	I2C2RX	是
47	I2C2TX	是
48	I2C2	是
49	I2C2E	是
50	U2RX	是
51	U2TX	是
52	U2E	是
53	U2	否
54	TMR3	否
55	TMR3G	否
56	TMR4	否
57	CCP2	否
58	保留	
59	CWG2	否
60	CLC2	否
61	INT2	否
62	保留	
63	保留	
64	保留	
65	保留	
66	保留	
67	保留	
68	保留	
69	保留	
70	TMR5	否
71	TMR5G	否
72	TMR6	否
73	CCP3	否
74	CWG3	否
75	CLC3	否
76	保留	
77	保留	
78	保留	
79	保留	
80	CCP4	否
81	CLC4	否
82 – 127	保留	

注 1： 所有非电平触发的触发源都是边沿触发的。

表15-3： 与DMA相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
DMAxCON0	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	238
DMAxCON1	DMODE<1:0>		DSTP	SMR<1:0>		SMODE<1:0>		SSTP	239
DMAxBUF	DBUF7	DBUF6	DBUF5	DBUF4	DBUF3	DBUF2	DBUF1	DBUF0	240
DMAxSSAL	SSA<7:0>								240
DMAxSSAH	SSA<15:8>								240
DMAxSSAU	—	—	SSA<21:16>						241
DMAxSPTRL	SPTR<7:0>								241
DMAxSPTRH	SPTR<15:8>								241
DMAxSPTRU	—	—	SPTR<21:16>						242
DMAxSSZL	SSZ<7:0>								242
DMAxSSZH	—	—	—	—	SSZ<11:8>				242
DMAxSCNTL	SCNT<7:0>								243
DMAxSCNTH	—	—	—	—	SCNT<11:8>				243
DMAxDSAL	DSA<7:0>								243
DMAxDSAH	DSA<15:8>								244
DMAxDPTRL	DPTR<7:0>								244
DMAxDPTRH	DPTR<15:8>								244
DMAxDSZL	DSZ<7:0>								245
DMAxDSZH	—	—	—	—	DSZ<11:8>				245
DMAxDCNTL	DCNT<7:0>								245
DMAxDCNTH	—	—	—	—	DCNT<11:8>				246
DMAxSIRQ	—	SIRQ<6:0>							246
DMAxAIRQ	—	AIRQ<6:0>							246

图注： — = 未实现位，读为0。DMA不使用阴影单元。

16.0 I/O端口

PIC18(L)F25/26K83器件有四个I/O端口，按表16-1所示分配。

表16-1： PIC18(L)F25/26K83器件的端口分配表

器件	PORTA	PORTB	PORTC	PORTE
PIC18(L)F25K83	●	●	●	●(1)
PIC18(L)F26K83	●	●	●	●(1)

注 1： 仅限引脚RE3。

每个端口都有十个控制其操作的寄存器。这些寄存器包括：

- PORTx寄存器（读取器件引脚的电平）
- LATx寄存器（输出锁存器）
- TRISx寄存器（数据方向）
- ANSELx寄存器（模拟选择）
- WPUx寄存器（弱上拉）
- INLVLx（输入电平控制）
- SLRCONx寄存器（压摆率控制）
- ODCONx寄存器（漏极开路控制）

除控制所有端口的位的寄存器外，还存在以下两个寄存器：

- RxyI2C（I^2C焊盘控制）

大多数端口引脚与器件模拟外设和数字外设共用功能。通常，当使能某个端口引脚上的外设时，该引脚将不能用作通用输出，但仍然可以对该引脚进行读操作。

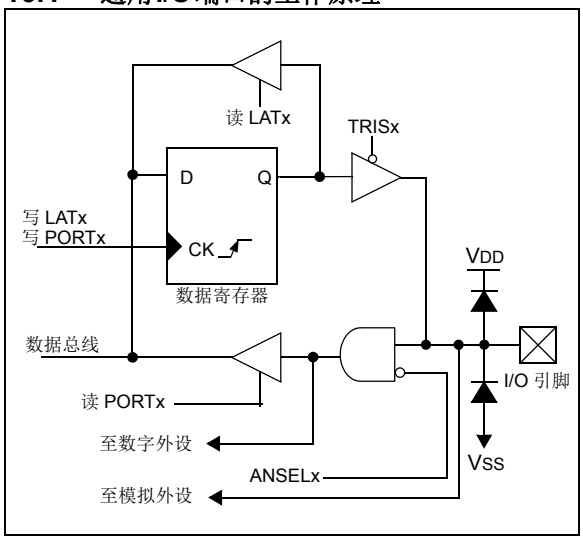
数据锁存器（LATx寄存器）在对I/O引脚驱动值进行读-修改-写操作时非常有用。

对LATx寄存器的写操作与写入相应PORTx寄存器的效果相同。读取LATx寄存器时，将会读取I/O端口锁存器中保存的值，而读取PORTx寄存器时，将会读取实际的I/O引脚值。

支持模拟输入的端口具有相关的ANSELx寄存器。当某个ANSELx位置1时，与该位相关的数字输入缓冲器会被禁止。

禁止输入缓冲器可以防止该引脚上介于逻辑高电平和低电平之间的模拟信号电平在逻辑输入电路中产生过大的电流。图16.1给出了通用I/O端口的简化模型，没有给出与其他外设的接口。

16.1 通用I/O端口的工作原理



16.2 I/O优先级

在复位之后，每个引脚均默认设为端口数据锁存器。其他功能通过外设引脚选择逻辑选择。更多信息，请参见**第17.0节“外设引脚选择（PPS）模块”**。

外设引脚选择列表中未列出模拟输入功能，例如ADC和比较器输入。这些输入在使用ANSELx寄存器将I/O引脚设置为模拟模式时有效。当引脚处于模拟模式时，数字输出功能可以继续控制引脚。

当使能模拟输出时，模拟输出优先于数字输出且强制数字输出驱动器为高阻态。

引脚功能的优先级如下所示：

1. 配置位
2. 模拟输出（禁止输入缓冲器）
3. 模拟输入
4. PPS的端口输入和输出

16.3 PORTx寄存器

在本节中，PORTx、LATx和TRISx等通用名称可与PORTA、PORTB和PORTC端口相关联。PORTE的功能与其他端口不同，将在单独的一节中介绍。

16.3.1 数据寄存器

PORTx是一个8位宽的双向端口。对应的数据方向寄存器是TRISx（寄存器16-2）。将TRISx某位置1（1）时，会将PORTA的相应引脚设为输入（即，禁止输出驱动器）。将TRISx某位清零（0）时，会将PORTx的相应引脚设为输出（即，使能输出驱动器并将输出锁存器中的内容输出到选定的引脚）。例16-1显示了如何初始化PORTx。

读PORTx寄存器（寄存器16-1）将读出相应引脚的状态，而对其进行写操作则是将数据写入端口锁存器。所有写操作都是读-修改-写操作。因此，对端口的写操作意味着先读端口引脚电平状态，然后修改这个值，最后再写入该端口的数据锁存器（LATx）。

端口数据锁存器LATx（寄存器16-3）保存输出端口数据，并包含写入LATx或PORTx的最新值。

例16-1： 初始化PORTA

```
; This code example illustrates
; initializing the PORTA register. The
; other ports are initialized in the same
; manner.

BANKSEL  PORTA        ;
CLRF     PORTA        ;Init PORTA
BANKSEL  LATA         ;Data Latch
CLRF     LATA         ;
BANKSEL  ANSELA       ;
CLRF     ANSELA       ;digital I/O
BANKSEL  TRISA        ;
MOVLW    B'11111000'  ;Set RA<7:3> as inputs
MOVWF    TRISA        ;and set RA<2:0> as
                      ;outputs
```

16.3.2 方向控制

即使在PORTx引脚用作模拟输入引脚时，TRISx寄存器（寄存器16-2）仍然控制PORTx引脚的输出驱动器。当引脚用作模拟输入引脚时，用户应确保TRISx寄存器中的相应位保持置1。配置为模拟输入的I/O引脚总是读为0。

16.3.3 模拟控制

ANSELx寄存器（寄存器16-4）用于将I/O引脚的输入模式配置为模拟。将相应的ANSELx位设置为高电平，将使引脚上的所有数字读操作都读为0，并允许引脚上的模拟功能正确工作。

ANSELx位的状态不会影响数字输出功能。TRIS清零且ANSEL置1的引脚将仍作为数字输出工作，但输入模式将变为模拟。当在受影响的端口上执行读-修改-写指令时，引脚行为可能与预期不符。

注： 在发生复位之后，ANSELx位默认设为模拟模式。要将任意引脚用作数字通用输入或外设输入，必须通过用户软件将相应的ANSEL位初始化为0。

16.3.4 漏极开路控制

ODCONx寄存器（寄存器16-6）用于控制端口的漏极开路功能。每个引脚的漏极开路操作可以独立进行选择。当ODCONx位置1时，相应的端口输出会变为只能灌入电流的漏极开路驱动器。当ODCONx位清零时，相应的端口输出引脚是能够拉出和灌入电流的标准推挽驱动器。

注： 当将引脚用于I^2C功能时，需要设置漏极开路控制。

16.3.5 压摆率控制

SLRCONx寄存器（寄存器16-7）用于控制每个端口引脚的压摆率选项。每个端口引脚的压摆率可以独立进行控制。当SLRCONx位置1时，相应端口引脚驱动器的压摆率会受到限制。当SLRCONx位清零时，相应端口引脚驱动器的压摆率将为最大可能值。

16.3.6 输入阈值控制

INLVLx寄存器（寄存器16-8）用于控制每个可用PORTx输入引脚的输入阈值电压。用户可以选择施密特触发器CMOS阈值或TTL兼容阈值。输入阈值对于确定PORTx寄存器的读取值很重要，同时它也是发生电平变化中断的电压（如果使能该功能）。关于阈值电压的更多信息，请参见表45-4。

注：	如果要更改所选择的输入阈值，则应先禁止所有外设模块再执行该操作。在模块处于工作状态时更改阈值电压，可能会意外产生与输入引脚相关联的电平变化，不论该引脚上的实际电压如何。

16.3.7 弱上拉控制

WPUx寄存器（寄存器16-5）用于控制每个端口引脚的各个弱上拉功能。

16.3.8 边沿可选的电平变化中断

可以通过检测端口引脚具有上升沿或下降沿的信号而产生中断。任意一个引脚都可以配置为产生中断。端口B、C、E的所有引脚以及RG5引脚上有电平变化中断模块。有关IOC模块的更多详细信息，请参见**第18.0节“电平变化中断”**。

16.3.9 I^2C焊盘控制

对于 PIC18(L)F25/26K83 器件，RB1、RB2、RC3、RC4、RD0(1)和RD1(1)引脚上都有I^2C特定焊盘。每个引脚的I^2C特性由RxyI2C寄存器控制（见寄存器16-9）。这些特性包括使能I^2C特定压摆率（超过标准GPIO压摆率），为I^2C引脚选择内部上拉，以及根据SMBus规范选择适当的输入阈值。

注 1：	PIC18(L)F25K83器件不提供RD0和RD1 I^2C焊盘。
2：	通过RxyI2C使能时，使用I^2C引脚的任意外设都会读取I^2C ST输入。

16.4 PORTE寄存器

16.4.1 主复位输入（$\overline{MCLR}$）

对于PIC18(L)F2xK83器件，PORTE仅在主复位功能禁止（MCLRE = 0）时可用。在这种情况下，PORTE为单位仅输入端口，仅由RE3组成。引脚工作原理与之前所述相同。PORTE寄存器的RE3是只读位，在MCLRE = 1（即使能主复位）时读为1。

16.4.2 RE3弱上拉

端口RE3引脚具有单独控制的内部弱上拉功能。置1时，WPUE3位使能RE3引脚上拉。当RE3端口引脚配置为$\overline{MCLR}$（CONFIG2L、MCLRE = 1且CONFIG4H、LVP = 0）或配置为低电压编程（MCLRE = x且LVP = 1）时，始终使能上拉，WPUE3位不产生任何影响。

16.4.3 电平变化中断

对于所有器件而言，只有PORTE的RE3引脚上提供电平变化中断功能。如果MCLRE = 1或者LVP = 1，则RE3端口功能被禁止，并且RE3的电平变化中断功能不可用。更多详细信息，请参见**第18.0节“电平变化中断”**。

16.5 寄存器定义：端口控制

寄存器16-1： PORTx：PORTx寄存器(1)

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
Rx7	Rx6	Rx5	Rx4	Rx3	Rx2	Rx1	Rx0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
1 = 置1	0 = 清零	x = 未知
-n/n = POR和BOR时的值/所有其他复位时的值		

bit 7-0 **Rx<7:0>**：Rx7:Rx0端口I/O值位
1 = 端口引脚电平 ≥ VIH
0 = 端口引脚电平 ≤ VIL

注 1： 写入PORTx时，实际上会写入相应的LATx寄存器。
读取PORTx寄存器时，将返回实际的I/O引脚值。

表16-2： 端口寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
PORTB	RB7(1)	RB6(1)	RB5	RB4	RB3	RB2	RB1	RB0
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
PORTE	—	—	—	—	RE3(2)	—	—	—

注 1： 在调试模式下，RB6和RB7位读为1。
2： PORTE3位为只读位且在MCLRE = 1（使能主复位）时读为1。

寄存器16-2： TRISx：三态控制寄存器

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISx7	TRISx6	TRISx5	TRISx4	TRISx3	TRISx2	TRISx1	TRISx0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
1 = 置1	0 = 清零	x = 未知
-n/n = POR和BOR时的值/所有其他复位时的值		

bit 7-0 **TRISx<7:0>**：TRISx端口I/O三态控制位
1 = 禁止端口输出驱动器
0 = 使能端口输出驱动器

表16-3： TRIS寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
TRISB	TRISB7(1)	TRISB6(1)	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0

注 1： 在调试模式下，RB6和RB7位读为1。

寄存器16-3： LATx：LATx寄存器(1)

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LATx7	LATx6	LATx5	LATx4	LATx3	LATx2	LATx1	LATx0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
1 = 置1	0 = 清零	x = 未知
-n/n = POR和BOR时的值/所有其他复位时的值		

bit 7-0 **LATx<7:0>**：Rx7:Rx0输出锁存器值位

注 1： 写入LATx等效于写入相应的PORTx寄存器。读取LATx寄存器将返回寄存器值，而不是I/O引脚值。

表16-4： LAT寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0

寄存器16-4： ANSELx：模拟选择寄存器

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
ANSELx7	ANSELx6	ANSELx5	ANSELx4	ANSELx3	ANSELx2	ANSELx1	ANSELx0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
1 = 置1	0 = 清零	x = 未知
-n/n = POR和BOR时的值/所有其他复位时的值		

bit 7-0 **ANSELx<7:0>**：Rx<7:0>引脚的模拟选择

1 = 禁止数字输入缓冲器。

0 = 使能ST和TTL输入器件

表16-5： 模拟选择端口寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ANSELA	ANSELA7	ANSELA6	ANSELA5	ANSELA4	ANSELA3	ANSELA2	ANSELA1	ANSELA0
ANSELB	ANSELB7	ANSELB6	ANSELB5	ANSELB4	ANSELB3	ANSELB2	ANSELB1	ANSELB0
ANSELC	ANSELC7	ANSELC6	ANSELC5	ANSELC4	ANSELC3	ANSELC2	ANSELC1	ANSELC0

寄存器16-5： WPUx：弱上拉寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
WPUx7	WPUx6	WPUx5	WPUx4	WPUx3	WPUx2	WPUx1	WPUx0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
1 = 置1	0 = 清零	x = 未知
-n/n = POR和BOR时的值/所有其他复位时的值		

bit 7-0 **WPUx<7:0>**：弱上拉PORTx控制位
1 = 使能弱上拉
0 = 禁止弱上拉

表16-6： 弱上拉端口寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WPUA	WPUA7	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0
WPUC	WPUC7	WPUC6	WPUC5	WPUC4	WPUC3	WPUC2	WPUC1	WPUC0
WPUE	—	—	—	—	WPUE3(1)	—	—	—

注 1： 如果MCLRE = 1，则RE3的弱上拉始终使能； WPUE3位不受影响。

寄存器16-6： ODCONx：漏极开路控制寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ODCx7	ODCx6	ODCx5	ODCx4	ODCx3	ODCx2	ODCx1	ODCx0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
1 = 置1	0 = 清零	x = 未知
-n/n = POR和BOR时的值/所有其他复位时的值		

bit 7-0 **ODCx<7:0>**：Rx<7:0>引脚的漏极开路配置

1 = 输出仅驱动低电平信号（仅灌电流）

0 = 输出驱动高电平和低电平信号（拉电流和灌电流）

表16-7： 漏极开路控制寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ODCONA	ODCA7	ODCA6	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0
ODCONB	ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0
ODCONC	ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0

寄存器16-7： SLRCONx：压摆率控制寄存器

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
SLRx7	SLRx6	SLRx5	SLRx4	SLRx3	SLRx2	SLRx1	SLRx0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
1 = 置1	0 = 清零	x = 未知
-n/n = POR和BOR时的值/所有其他复位时的值		

bit 7-0 **SLRx<7:0>**：各个Rx<7:0>引脚的压摆率控制

1 = 端口引脚的压摆率受到限制

0 = 端口引脚的压摆率为最大值

表16-8： 压摆率控制寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SLRCONA	SLRA7	SLRA6	SLRA5	SLRA4	SLRA3	SLRA2	SLRA1	SLRA0
SLRCONB	SLRB7	SLRB6	SLRB5	SLRB4	SLRB3	SLRB2	SLRB1	SLRB0
SLRCONC	SLRC7	SLRC6	SLRC5	SLRC4	SLRC3	SLRC2	SLRC1	SLRC0

寄存器16-8： INLVLx：输入电平控制寄存器

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
INLVLx7	INLVLx6	INLVLx5	INLVLx4	INLVLx3	INLVLx2	INLVLx1	INLVLx0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
1 = 置1	0 = 清零	x = 未知
-n/n = POR和BOR时的值/所有其他复位时的值		

bit 7-0 **INLVLx<7:0>**：各个Rx<7:0>引脚的输入电平选择
1 = 对于端口读操作和电平变化中断，使用ST输入
0 = 对于端口读操作和电平变化中断，使用TTL输入

表16-9： 输入电平端口寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INLVLA	INLVLA7	INLVLA6	INLVLA5	INLVLA4	INLVLA3	INLVLA2	INLVLA1	INLVLA0
INLVLB	INLVLB7	INLVLB6	INLVLB5	INLVLB4	INLVLB3	INLVLB2(1)	INLVLB1(1)	INLVLB0
INLVLC	INLVLC7	INLVLC6	INLVLC5	INLVLC4(1)	INLVLC3(1)	INLVLC2	INLVLC1	INLVLC0
INLVLE	—	—	—	—	INLVLE3	—	—	—

注 1： 通过RxyI2C使能时，使用I^2C引脚的任意外设都会读取I^2C ST输入。
2： PIC18(L)F25K83中未实现。

寄存器16-9： RxyI2C：I²C焊盘Rxy控制寄存器

U-0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
—	SLEW	PU<1:0>		—	—	TH<1:0>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7 **未实现**：读为0

bit 6 **SLEW**：使能I²C特定压摆率限制
1 = 使能I²C特定压摆率限制。禁止标准焊盘压摆率限制。SLRxy位被忽略。
0 = 标准GPIO压摆率；通过SLRxy位使能/禁止。

bit 5-4 **PU<1:0>**：I²C上拉选择位
11 = 保留
10 = 10倍标准弱上拉电流
01 = 2倍标准弱上拉电流
00 = 标准GPIO弱上拉，通过WPUxy位使能

bit 3-2 **未实现**：读为0

bit 1-0 **TH<1:0>**：I²C输入阈值选择位
11 = SMBus 3.0（1.35V）输入阈值
10 = SMBus 2.0（2.1V）输入阈值
01 = I²C特定输入阈值
00 = 标准GPIO输入上拉，通过INLVLxy寄存器使能

表16-10： I²C焊盘控制寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RB1I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	
RB2I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	
RC3I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	
RC4I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	

表16-11： 与I/O相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	252
PORTB	RB7(1)	RB6(1)	RB5	RB4	RB3	RB2	RB1	RB0	252
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	252
PORTE	—	—	—	—	RE3(2)	—	—	—	252
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	253
TRISB	TRISB7(3)	TRISB6(3)	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	253
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	253
LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	254
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	254
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	254
ANSELA	ANSELA7	ANSELA6	ANSELA5	ANSELA4	ANSELA3	ANSELA2	ANSELA1	ANSELA0	255
ANSELB	ANSELB7	ANSELB6	ANSELB5	ANSELB4	ANSELB3	ANSELB2	ANSELB1	ANSELB0	255
ANSELC	ANSELC7	ANSELC6	ANSELC5	ANSELC4	ANSELC3	ANSELC2	ANSELC1	ANSELC0	255
WPUA	WPUA7	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	256
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	256
WPUC	WPUC7	WPUC6	WPUC5	WPUC4	WPUC3	WPUC2	WPUC1	WPUC0	256
WPUE	—	—	—	—	WPUE3(4)	—	—	—	256
ODCONA	ODCA7	ODCA6	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0	257
ODCONB	ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0	257
ODCONC	ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0	257
SLRCONA	SLRA7	SLRA6	SLRA5	SLRA4	SLRA3	SLRA2	SLRA1	SLRA0	258
SLRCONB	SLRB7	SLRB6	SLRB5	SLRB4	SLRB3	SLRB2	SLRB1	SLRB0	258
SLRCONC	SLRC7	SLRC6	SLRC5	SLRC4	SLRC3	SLRC2	SLRC1	SLRC0	258
INLVLA	INLVLA7	INLVLA6	INLVLA5	INLVLA4	INLVLA3	INLVLA2	INLVLA1	INLVLA0	259
INLVLB	INLVLB7	INLVLB6	INLVLB5	INLVLB4	INLVLB3	INLVLB2(5)	INLVLB1(5)	INLVLB0	259
INLVLC	INLVLC7	INLVLC6	INLVLC5	INLVLC4(5)	INLVLC3(5)	INLVLC2	INLVLC1	INLVLC0	259
INLVLE	—	—	—	—	INLVLE3	—	—	—	259
RB1I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>		260
RB2I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>		260
RC3I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>		260
RC4I2C	—	SLEW	PU<1:0>		—	—	TH<1:0>		260

图注： — = 未实现位，读为0。I/O端口不使用阴影单元。

注 1： 在调试模式下，RB6和RB7位读为1。
2： PORTE3位为只读位且在MCLRE = 1（使能主复位）时读为1。
3： 在调试模式下，RB6和RB7位读为1。
4： 如果MCLRE = 1，则RE3的弱上拉始终使能；WPUE3位不受影响。
5： 通过RxyI2C使能时，使用I^2C引脚的任意外设都会读取I^2C ST输入。

17.0 外设引脚选择（PPS）模块

外设引脚选择（PPS）模块用于将外设输入和输出与器件I/O引脚连接。选择范围仅包含数字信号。所有模拟输入和输出均固定连接至它们所分配的引脚。输入和输出选择是独立的，如图17-1所示。

使用外设xxxPPS寄存器（寄存器17-1）选择外设输入，使用端口RxyPPS寄存器（寄存器17-2）选择外设输出。例如，要选择PORTC<7>作为UART1 RX输入，将U1RXPPS设置为0b1 0111，要选择PORTC<6>作为UART1 TX输出，将RC6PPS设置为0b01 0011。

17.1 PPS输入

每个外设均具有一个用于选择外设输入的PPS寄存器。输入包括器件引脚。

多个外设可以使用同一个源同时工作。端口读操作总是返回引脚电平，无论外设PPS选择如何。如果引脚还具备相关模拟功能，则必须清零该引脚的ANSEL位才会使能数字输入缓冲器。

虽然每个外设均具有自己的PPS输入选择寄存器，但每个外设的选择是相同的，如寄存器17-1所示。

注：	寄存器名称中的符号“xxx”是外设标识符的占位符。例如，INT0PPS。

17.2 PPS输出

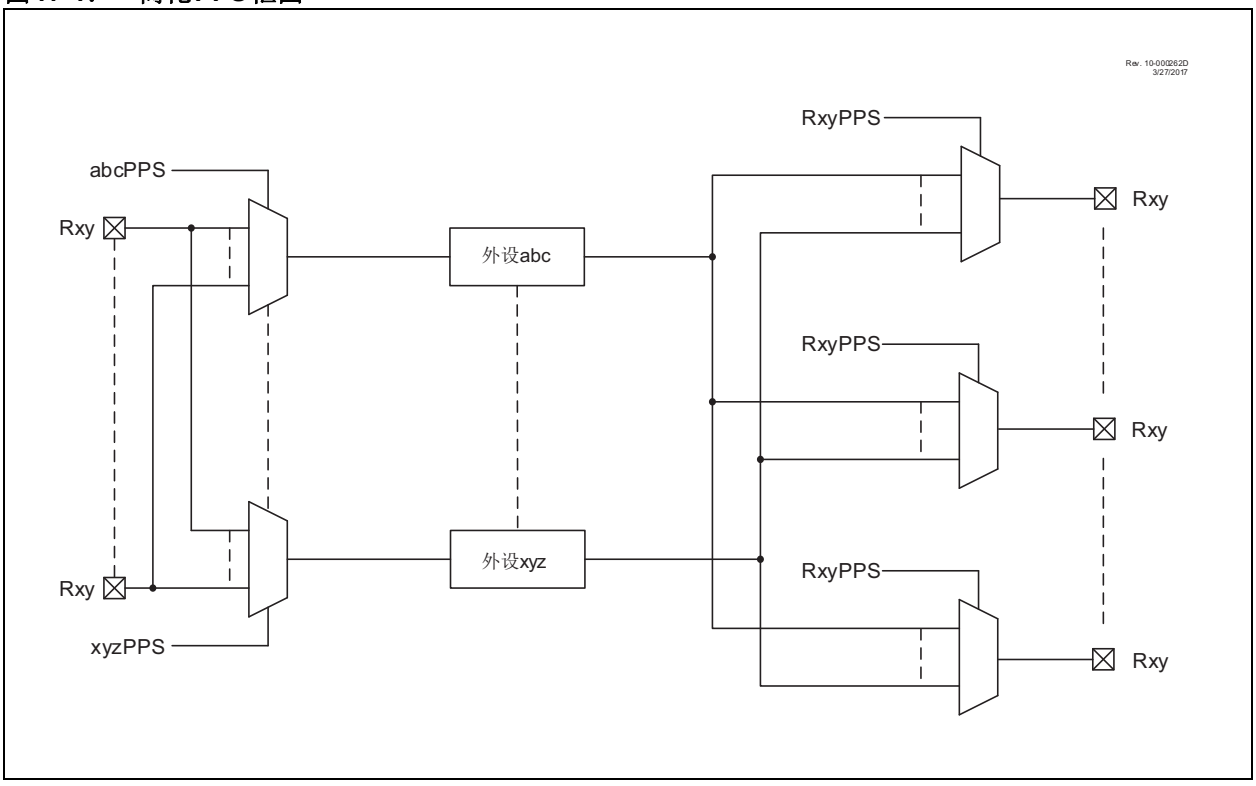
每个I/O引脚均具有一个用于选择引脚输出源的PPS寄存器。除了少数例外情况，与该引脚相关的端口TRIS控制将保持对引脚输出驱动器的控制权。作为外设操作的一部分，控制引脚输出驱动器的外设将根据需要改写TRIS控制。这些外设包括：

- UART

虽然每个引脚均具有自己的PPS外设选择寄存器，但每个引脚的选择是相同的，如寄存器17-2所示。

注：	符号“Rxy”是引脚标识符的占位符。例如，RA0PPS。

图17-1： 简化PPS框图



17.3 双向引脚

对于在单个引脚上具有双向信号的外设，在进行PPS选择时必须使PPS输入和PPS输出选择同一引脚。具有双向信号的外设包括：

- I^2C

注： 有关I^2C兼容的引脚，请参见表17-1。时钟和数据信号可输送到任何引脚，但是不具有I^2C兼容性的引脚将以标准TTL/ST逻辑电平（由INVLV寄存器选择）工作。

17.4 PPS锁定

PPS包含了一种模式，在该模式下可以锁定所有输入和输出选择，以防止意外的更改。PPS选择通过将PPSLOCK寄存器的PPSLOCKED位置1来进行锁定。置1和清零该位需要一个特殊序列作为额外的预防措施，以防止意外的更改。例17-1给出了置1和清零PPSLOCKED位的示例。

例17-1： PPS锁定序列

```
; Disable interrupts:
    BCF      INTCON0,GIE

; Bank to PPSLOCK register
    BANKSEL  PPSLOCK
    MOVLB    PPSLOCK
    MOVLW    55h

; Required sequence, next 4 instructions
    MOVWF    PPSLOCK
    MOVLW    AAh
    MOVWF    PPSLOCK

; Set PPSLOCKED bit to disable writes
; Only a BSF instruction will work
    BSF      PPSLOCK,0

; Enable Interrupts
    BSF      INTCON0,GIE
```

例17-2： PPS解锁序列

```
; Disable interrupts:
    BCF      INTCON0,GIE

; Bank to PPSLOCK register
    BANKSEL  PPSLOCK
    MOVLB    PPSLOCK
    MOVLW    55h

; Required sequence, next 4 instructions
    MOVWF    PPSLOCK
    MOVLW    AAh
    MOVWF    PPSLOCK

; Clear PPSLOCKED bit to enable writes
; Only a BCF instruction will work
    BCF      PPSLOCK,0

; Enable Interrupts
    BSF      INTCON0,GIE
```

17.5 PPS一次锁定

当该位置1时，PPSLOCKED位只能在器件复位之后清零和置1一次。这使得可以清零PPSLOCKED位，从而可以在初始化过程中进行输入和输出选择。在完成所有选择之后将PPSLOCKED位置1时，它将一直保持置1，直到下一个器件复位事件之后才能清零。

17.6 休眠期间的操作

PPS输入和输出选择不会受休眠影响。

17.7 复位的影响

器件上电复位（POR）会将所有PPS输入和输出选择清除为其默认值。所有其他复位会将选择保留不变。引脚分配表3给出了默认的输入选择。PPS一次锁定也会被移除。

17.8 寄存器定义：PPS输入选择

寄存器17-1： xxxPPS：外设xxx输入选择

U-0	U-0	R/W-m/u(1)	R/W-m/u(1)	R/W-m/u(1)	R/W-m/u(1)	R/W-m/u(1)	R/W-m/u(1)
—	—	xxxPPS<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	-n/n = POR和BOR时的值/所有其他复位时的值
u = 不变	x = 未知	q = 值取决于外设
1 = 置1	U = 未实现位，读为0	m = 值取决于该输入的默认位置
0 = 清零		

bit 7-6 **未实现：**读为0

bit 5-3 **xxxPPS<5:3>：**外设xxx输入PORTx引脚选择位
有关可用端口和默认引脚位置的列表，请参见表17-1。
101 = 保留
100 = 保留
011 = 保留
010 = PORTC
001 = PORTB
000 = PORTA

bit 2-0 **xxxPPS<2:0>：**外设xxx输入PORTx引脚选择位
111 = 外设输入来自PORTx引脚7（Rx7）
110 = 外设输入来自PORTx引脚6（Rx6）
101 = 外设输入来自PORTx引脚5（Rx5）
100 = 外设输入来自PORTx引脚4（Rx4）
011 = 外设输入来自PORTx引脚3（Rx3）
010 = 外设输入来自PORTx引脚2（Rx2）
001 = 外设输入来自PORTx引脚1（Rx1）
000 = 外设输入来自PORTx引脚0（Rx0）

注 1： 该寄存器的复位值“m”由该输入的器件默认位置确定。

表17-1： PPS输入寄存器详细信息

外设	PPS输入寄存器	POR时的默认引脚选择	POR时的寄存器复位值	所选PORTx提供的输入		
				PIC18(L)F2xK83		
中断0	INT0PPS	RB0	0b0 1000	A	B	—
中断1	INT1PPS	RB1	0b0 1001	A	B	—
中断2	INT2PPS	RB2	0b0 1010	A	B	—
Timer0时钟	T0CKIPPS	RA4	0b0 0100	A	B	—
Timer1时钟	T1CKIPPS	RC0	0b1 0000	A	—	C
Timer1门控	T1GPPS	RB5	0b0 1101	—	B	C
Timer3时钟	T3CKIPPS	RC0	0b1 0000	—	B	C
Timer3门控	T3GPPS	RC0	0b1 0000	A	—	C
Timer5时钟	T5CKIPPS	RC2	0b1 0010	A	—	C
Timer5门控	T5GPPS	RB4	0b0 1100	—	B	C
Timer2时钟	T2INPPS	RC3	0b1 0011	A	—	C
Timer4时钟	T4INPPS	RC5	0b1 0101	—	B	C
Timer6时钟	T6INPPS	RB7	0b0 1111	—	B	C
CCP1	CCP1PPS	RC2	0b1 0010	—	B	C
CCP2	CCP2PPS	RC1	0b1 0001	—	B	C
CCP3	CCP3PPS	RB5	0b0 1101	—	B	C
CCP4	CCP4PPS	RB0	0b0 1000	—	B	C
SMT1窗口	SMT1WINPPS	RC0	0b1 0000	—	B	C
SMT1信号	SMT1SIGPPS	RB4	0b0 1100	—	B	C
SMT2窗口	SMT2WINPPS	RB5	0b0 1101	—	B	C
SMT2信号	SMT2SIGPPS	RC1	0b1 0001	—	B	C
CWG1	CWG1PPS	RB0	0b0 1000	—	B	C
CWG2	CWG2PPS	RB1	0b0 1001	—	B	C
CWG3	CWG3PPS	RB2	0b0 1010	—	B	C
DSM1低载波	MD1CARLPPS	RA3	0b0 0011	A	—	C
DSM1高载波	MD1CARHPPS	RA4	0b0 0100	A	—	C
DSM1源	MD1SRCPPS	RA5	0b0 0101	A	—	C
CLCx输入1	CLCIN0PPS	RA0	0b0 0000	A	—	C
CLCx输入2	CLCIN1PPS	RA1	0b0 0001	A	—	C
CLCx输入3	CLCIN2PPS	RB6	0b0 1110	—	B	C
CLCx输入4	CLCIN3PPS	RB7	0b0 1111	—	B	C
ADC转换触发器	ADACTPPS	RB4	0b0 1100	—	B	C
SPI1时钟	SPI1SCKPPS	RC3	0b1 0011	—	B	C
SPI1数据	SPI1SDIPPS	RC4	0b1 0100	—	B	C
SPI1从选择	SPI1SSPPS	RA5	0b0 0101	A	—	C
I^2C1时钟	I2C1SCLPPS	RC3	0b1 0011	—	B	C
I^2C1数据	I2C1SDAPPS	RC4	0b1 0100	—	B	C
I^2C2时钟	I2C2SCLPPS	RB1	0b0 1001	—	B	C
I^2C2数据	I2C2SDAPPS	RB2	0b0 1010	—	B	C
UART1接收	U1RXPPS	RC7	0b1 0111	—	B	C
UART1允许发送	U1CTSPPS	RC6	0b1 0110	—	B	C
UART2接收	U2RXPPS	RB7	0b0 1111	—	B	C
UART2允许发送	U2CTSPPS	RB6	0b0 1110	—	B	C
CAN接收	CANRXPPS	RB3	0b0 1011	—	B	C

寄存器17-2：　RxyPPS：引脚Rxy输出源选择寄存器

U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	RxyPPS<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6　**未实现：**读为0

bit 5-0　**RxyPPS<5:0>：**引脚Rxy输出源选择位
有关可用端口的列表，请参见表17-2。

表17-2： PPS输出寄存器详细信息

RxyPPS<5:0>	引脚Rxy输出源	器件配置		
		PIC18(L)F2xK83		
0b11 1111 – 0b11 0101	保留			
0b11 0100	CANTX1	—	B	C
0b11 0011	CANTX0	—	B	C
0b11 0010	ADGRDB	A	—	C
0b11 0001	ADGRDA	A	—	C
0b11 0000	CWG3D	A	—	C
0b10 1111	CWG3C	A	—	C
0b10 1110	CWG3B	A	—	C
0b10 1101	CWG3A	—	B	C
0b10 1100	CWG2D	—	B	C
0b10 1011	CWG2C	—	B	C
0b10 1010	CWG2B	—	B	C
0b10 1001	CWG2A	—	B	C
0b10 1000	DSM1	A	—	C
0b10 0111	CLKR	—	B	C
0b10 0110	NCO1	A	—	C
0b10 0101	TMR0	—	B	C
0b10 0100	I^2C2（SDA）	—	B	C
0b10 0011	I^2C2（SCL）	—	B	C
0b10 0010	I^2C1（SDA）	—	B	C
0b10 0001	I^2C1（SCL）	—	B	C
0b10 0000	SPI1（$\overline{SS}$）	A	—	C
0b01 1111	SPI1（SDO）	—	B	C
0b01 1110	SPI1（SCK）	—	B	C
0b01 1101	C2OUT	A	—	C
0b01 1100	C1OUT	A	—	C
0b01 1011 – 0b01 1001	保留			
0b01 1000	UART2（$\overline{RTS}$）	—	B	C
0b01 0111	UART2（TXDE）	—	B	C
0b01 0110	UART2（TX）	—	B	C
0b01 0101	UART1（$\overline{RTS}$）	—	B	C
0b01 0100	UART1（TXDE）	—	B	C
0b01 0011	UART1（TX）	—	B	C
0b01 0010 – 0b01 0001	保留			
0b01 0000	PWM8	A	—	C
0b00 1111	PWM7	A	—	C
0b00 1110	PWM6	A	—	C
0b00 1101	PWM5	A	—	C
0b00 1100	CCP4	—	B	C
0b00 1011	CCP3	—	B	C
0b00 1010	CCP2	—	B	C
0b00 1001	CCP1	—	B	C
0b00 1000	CWG1D	—	B	C
0b00 0111	CWG1C	—	B	C
0b00 0110	CWG1B	—	B	C

表17-2： PPS输出寄存器详细信息

RxyPPS<5:0>	引脚Rxy输出源	器件配置		
		PIC18(L)F2xK83		
0b00 0101	CWG1A	—	B	C
0b00 0100	CLC4OUT	—	B	C
0b00 0011	CLC3OUT	—	B	C
0b00 0010	CLC2OUT	A	—	C
0b00 0001	CLC1OUT	A	—	C
0b00 0000	LATxy	A	B	C

寄存器17-3： PPSLOCK：PPS锁定寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	PPSLOCKED
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-1 **未实现：**读为0

bit 0 **PPSLOCKED：**PPS锁定位
1 = PPS已锁定。
0 = PPS未锁定。可以更改PPS选择。

表17-3： 与PPS模块相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
PPSLOCK	—	—	—	—	—	—	—	PPSLOCKED	268
INT0PPS	—	—	—	INT0PPS<4:0>					264
INT1PPS	—	—	—	INT1PPS<4:0>					264
INT2PPS	—	—	—	INT2PPS<4:0>					264
T0CKIPPS	—	—	—	T0CKIPPS<4:0>					264
T1CKIPPS	—	—	—	T1CKIPPS<4:0>					264
T1GPPS	—	—	—	T1GPPS<4:0>					264
T3CKIPPS	—	—	—	T3CKIPPS<4:0>					264
T3GPPS	—	—	—	T3GPPS<4:0>					264
T5CKIPPS	—	—	—	T5CKIPPS<4:0>					264
T5GPPS	—	—	—	T5GPPS<4:0>					264
T2INPPS	—	—	—	T2INPPS<4:0>					264
T4INPPS	—	—	—	T4INPPS<4:0>					264
T6INPPS	—	—	—	T6INPPS<4:0>					264
CCP1PPS	—	—	—	CCP1PPS<4:0>					264
CCP2PPS	—	—	—	CCP2PPS<4:0>					264
CCP3PPS	—	—	—	CCP3PPS<4:0>					264
CCP4PPS	—	—	—	CCP4PPS<4:0>					264
SMT1WINPPS	—	—	—	SMT1WINPPS<4:0>					264
SMT1SIGPPS	—	—	—	SMT1SIGPPS<4:0>					264
SMT2WINPPS	—	—	—	SMT2WINPPS<4:0>					264
SMT2SIGPPS	—	—	—	SMT2SIGPPS<4:0>					264
CWG1PPS	—	—	—	CWG1PPS<4:0>					264
CWG2PPS	—	—	—	CWG2PPS<4:0>					264
CWG3PPS	—	—	—	CWG3PPS<4:0>					264
MD1CARLPPS	—	—	—	MDCARLPPS<4:0>					264
MD1CARHPPS	—	—	—	MDCARHPPS<4:0>					264
MD1SRCPPS	—	—	—	MDSRCPPS<4:0>					264
CLCIN0PPS	—	—	—	CLCIN0PPS<4:0>					264
CLCIN1PPS	—	—	—	CLCIN1PPS<4:0>					264
CLCIN2PPS	—	—	—	CLCIN2PPS<4:0>					264
CLCIN3PPS	—	—	—	CLCIN3PPS<4:0>					264
ADACTPPS	—	—	—	ADACTPPS<4:0>					264
SPI1SCKPPS	—	—	—	SPI1SCKPPS<4:0>					264
SPI1SDIPPS	—	—	—	SPI1SDIPPS<4:0>					264
SPI1SSPPS	—	—	—	SPI1SSPPS<4:0>					264
I2C1SCLPPS	—	—	—	I2C1SCLPPS<4:0>					264
I2C1SDAPPS	—	—	—	I2C1SDAPPS<4:0>					264
I2C2SCLPPS	—	—	—	I2C2SCLPPS<4:0>					264
I2C2SDAPPS	—	—	—	I2C2SDAPPS<4:0>					264
U1RXPPS	—	—	—	U1RXPPS<4:0>					264
U1CTSPPS	—	—	—	U1CTSPPS<4:0>					264
U2RXPPS	—	—	—	U2RXPPS<4:0>					264
U2CTSPPS	—	—	—	U2CTPPS<4:0>					264
RxyPPS	—	—	—	RxyPPS<4:0>					264
CANRXPPS	—	—	—	CANRXPPS<4:0>					264

图注： — = 未实现，读为0。PPS模块不使用阴影单元。

18.0 电平变化中断

对于PIC18(L)F25/26K83系列器件，PORTA、PORTB、PORTC和PORTE的RE3引脚可以配置为作为电平变化中断（Interrupt-On-Change，IOC）引脚工作。可以通过检测具有上升沿或下降沿的信号而产生中断。任意一个端口引脚或端口引脚组合都可以配置为产生中断。电平变化中断模块具有以下特性：

- 电平变化中断允许（主开关）
- 独立的引脚配置
- 上升沿和下降沿检测
- 独立的引脚中断标志

图18-1给出了IOC模块的框图。

18.1 使能模块

要允许各个端口引脚产生中断，PIE0寄存器的IOCIE位必须置1。如果IOCIE位被禁止，在引脚上仍然会发生边沿检测，但不会产生中断。

18.2 独立的引脚配置

对于每个端口引脚，都提供了上升沿检测器和下降沿检测器。要允许引脚检测上升沿，需要将IOCxP寄存器的相关位置1。要允许引脚检测下降沿，需要将IOCxN寄存器的相关位置1。

通过分别将IOCxP和IOCxN寄存器中的相关位置1，一个引脚可以配置为同时检测上升沿和下降沿。

18.3 中断标志

分别位于IOCAF、IOCBF、IOCCF和IOCEF寄存器中的IOCAFx、IOCBFx、IOCCFx和IOCEF3位是对应于关联端口的电平变化中断引脚的状态标志。如果在正确使能的引脚上检测到期望的边沿，则对应于该引脚的状态标志会置1，并且如果IOCIE位也置1，则还会产生中断。PIR0寄存器的IOCIF位会反映所有IOCAFx、IOCBFx、IOCCFx和IOCEF3位的状态。

18.4 清零中断标志

各个状态标志（IOCAFx、IOCBFx、IOCCFx和IOCEF3位）可以通过将其复位为零的方式清零。如果在该清零操作期间检测到另一个边沿，则无论实际写入的值如何，相关的状态标志都会在序列结束时置1。

为了确保在清零标志时不会丢失任何已检测到的边沿，应当仅执行可屏蔽已知更改位的逻辑与操作。以下序列是一个说明应执行何种操作的示例。

例18-1： 清零中断标志（以PORTA为例）

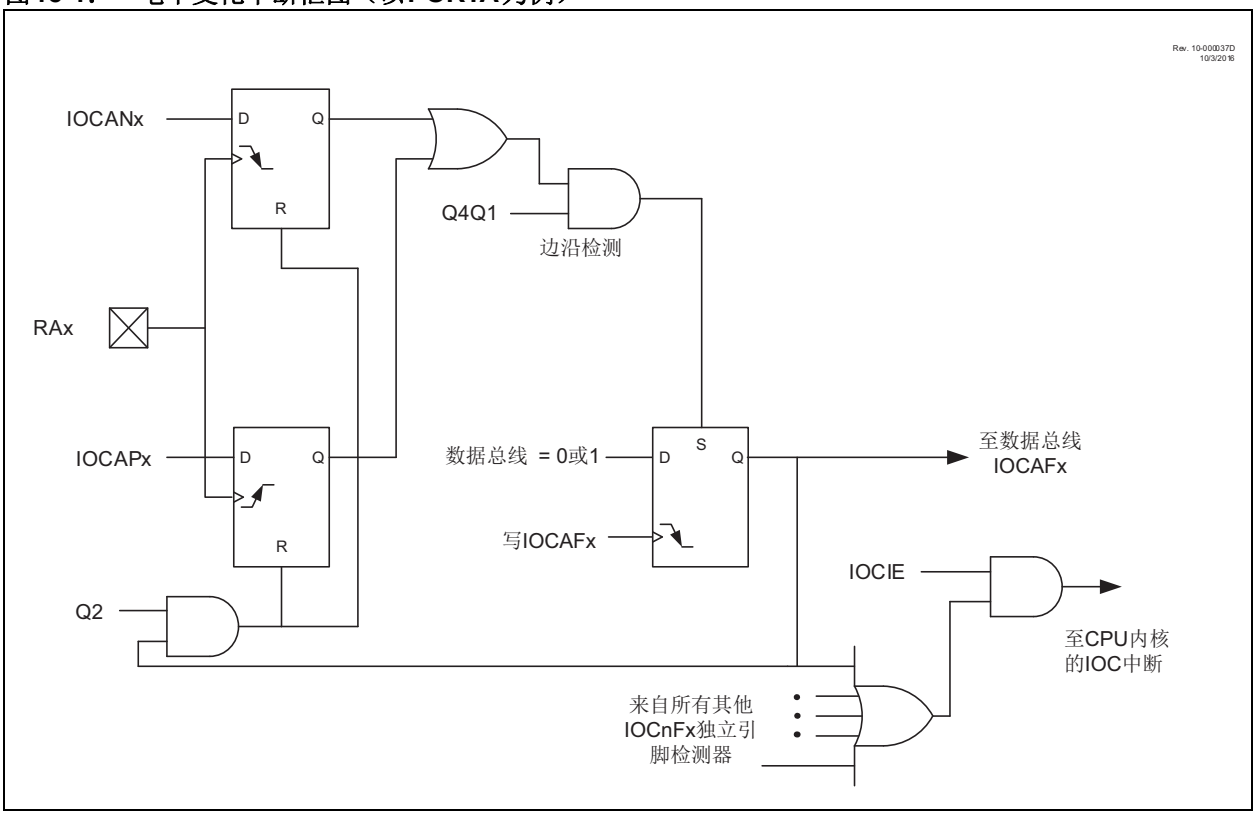
```
MOVLW   0xff
XORWF   IOCAF, W
ANDWF   IOCAF, F
```

18.5 休眠模式下的操作

如果IOCIE位置1，电平变化中断序列会将器件从休眠模式唤醒。

如果在处于休眠模式时检测到边沿，则在退出休眠模式执行第一条指令之前，会先更新IOCxF寄存器。

图18-1： 电平变化中断框图（以PORTA为例）



18.6 寄存器定义：电平变化中断控制

寄存器18-1： IOCxP：电平变化中断正边沿寄存器示例

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCxP7	IOCxP6	IOCxP5	IOCxP4	IOCxP3	IOCxP2	IOCxP1	IOCxP0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **IOCxP<7:0>：**电平变化中断正边沿允许位
1 = 在IOCx引脚上对于正边沿允许电平变化中断。检测到边沿时将相关的状态位和中断标志置1。
0 = 禁止关联引脚的电平变化中断。

寄存器18-2： IOCxN：电平变化中断负边沿寄存器示例

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
IOCxN7	IOCxN6	IOCxN5	IOCxN4	IOCxN3	IOCxN2	IOCxN1	IOCxN0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **IOCxN<7:0>：**电平变化中断负边沿允许位
1 = 在IOCx引脚上对于负边沿允许电平变化中断。检测到边沿时将相关的状态位和中断标志置1。
0 = 禁止关联引脚的电平变化中断

寄存器18-3： IOCxF：电平变化中断标志寄存器示例

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
IOCxF7	IOCxF6	IOCxF5	IOCxF4	IOCxF3	IOCxF2	IOCxF1	IOCxF0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位

bit 7-0 **IOCxF<7:0>：**电平变化中断标志位
1 = 在关联引脚上检测到使能的电平变化。当IOCP[n] = 1且在IOCn引脚上检测到正边沿时，或者当IOCN[n] = 1且在IOCn引脚上检测到负边沿时置1
0 = 未检测到电平变化，或者用户清除了检测到的电平变化

表18-1： IOC寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCAP	IOCAP7	IOCAP6	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0
IOCAN	IOCAN7	IOCAN6	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0
IOCAF	IOCAF7	IOCAF6	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0
IOCCP	IOCCP7	IOCCP6	IOCCP5	IOCCP4	IOCCP3	IOCCP2	IOCCP1	IOCCP0
IOCCN	IOCCN7	IOCCN6	IOCCN5	IOCCN4	IOCCN3	IOCCN2	IOCCN1	IOCCN0
IOCCF	IOCCF7	IOCCF6	IOCCF5	IOCCF4	IOCCF3	IOCCF2	IOCCF1	IOCCF0
IOCEP	—	—	—	—	IOCEP3(1)	—	—	—
IOCEN	—	—	—	—	IOCEN3(1)	—	—	—
IOCEF	—	—	—	—	IOCEF3(1)	—	—	—

注 1： 如果MCLRE = 1或者LVP = 1，则RE3端口功能被禁止，并且RE3上的IOC不可用。

表18-2： 与电平变化中断相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
IOCxF	IOCxF7	IOCxF6	IOCxF5	IOCxF4	IOCxF3	IOCxF2	IOCxF1	IOCxF0	272
IOCxN	IOCxN7	IOCxN6	IOCxN5	IOCxN4	IOCxN3	IOCxN2	IOCxN1	IOCxN0	272
IOCxP	IOCxP7	IOCxP6	IOCxP5	IOCxP4	IOCxP3	IOCxP2	IOCxP1	IOCxP0	272

图注： — = 未实现位，读为0。电平变化中断不使用阴影单元。

19.0 外设模块禁止（PMD）

在休眠、空闲和打盹模式下，用户可以通过减慢或停止CPU时钟来显著降低功耗。即使如此，外设模块仍然保持时钟速度，因此存在一些功耗。在某些情况下，应用可能需要这些模式无法提供的功能：在消除外设功耗的同时为CPU分配有限电源资源的能力。

PIC18(L)F25/26K83系列允许选择性地使能或禁止外设模块，以尽可能将其置于最低功耗模式，从而满足此要求。

发生任何复位后，所有模块均默认设为开启状态。

19.1 禁止模块

禁止模块有以下影响：

- 模块的所有时钟和控制输入暂停；不存在任何逻辑转换，模块无法正常工作。
- 模块保持在复位状态。
- 所有SFR变为“未实现”
  - 禁止写操作
  - 读操作返回00h
- I/O功能的优先级顺序如第16.2节“I/O优先级”所述
- 会同时禁止所有相关输入选择寄存器

19.2 使能模块

当PMD寄存器位清零时，模块重新使能并处于复位状态（上电复位）。SFR数据将反映POR复位值。

根据模块的不同，可能需要最多一个完整指令周期，模块才能变为活动状态。重新使能模块后，至少在一个指令周期内不要与模块有任何交互（例如，写寄存器）。

19.3 复位的影响

发生任何复位后，每个控制位被设置为0，同时使能所有模块。

19.4 系统时钟禁止

SYSCMD（PMD0，寄存器19-1）置1时，将禁止外设的系统时钟（FOSC）分布网络。并非所有的外设都使用SYSCLK，所以不是所有外设都会受影响。请参见具体外设说明以了解其是否会受到该位的影响。

19.5 寄存器定义：外设模块禁止

寄存器19-1： PMD0：PMD控制寄存器0

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD
7							0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **SYSCMD**：禁止外设系统时钟网络位(1)
请参见**第19.4节“系统时钟禁止”**中的说明。
1 = 禁止系统时钟网络（FOSC）
0 = 使能系统时钟网络

bit 6 **FVRMD**：禁止固定参考电压位
1 = 禁止FVR模块
0 = 使能FVR模块

bit 5 **HLVDMD**：禁止低压检测位
1 = 禁止HLVD模块
0 = 使能HLVD模块

bit 4 **CRCMD**：禁止CRC引擎位
1 = 禁止CRC模块
0 = 使能CRC模块

bit 3 SCANMD：禁止NVM存储器扫描器位(2)
1 = 禁止NVM存储器扫描模块
0 = 使能NVM存储器扫描模块

bit 2 **NVMMD**：NVM模块禁止位(3)
1 = 禁止读写所有存储器；不能写入NVMCON寄存器
0 = 使能NVM模块

bit 1 **CLKRMD**：禁止时钟参考位
1 = 禁止CLKR模块
0 = 使能CLKR模块

bit 0 **IOCMD**：禁止所有端口的电平变化中断位
1 = 禁止IOC模块
0 = 使能IOC模块

注 1： 清零SYSCMD位会禁止外设的系统时钟（FOSC），但由FOSC/4提供时钟的外设不受影响。
2： 由CONFIG4H中的SCANE位决定。
3： 使能NVM时，访问数据之前可能需要最长1 μs的延时。

寄存器19-2： PMD1：PMD控制寄存器1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
NCO1MD	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **NCO1MD**：禁止NCO1模块位
1 = 禁止NCO1模块
0 = 使能NCO1模块

bit 6 **TMR6MD**：禁止定时器TMR6位
1 = 禁止TMR6模块
0 = 使能TMR6模块

bit 5 **TMR5MD**：禁止定时器TMR5位
1 = 禁止TMR5模块
0 = 使能TMR5模块

bit 4 **TMR4MD**：禁止定时器TMR4位
1 = 禁止TMR4模块
0 = 使能TMR4模块

bit 3 **TMR3MD**：禁止定时器TMR3位
1 = 禁止TMR3模块
0 = 使能TMR3模块

bit 2 **TMR2MD**：禁止定时器TMR2位
1 = 禁止TMR2模块
0 = 使能TMR2模块

bit 1 **TMR1MD**：禁止定时器TMR1位
1 = 禁止TMR1模块
0 = 使能TMR1模块

bit 0 **TMR0MD**：禁止定时器TMR0位
1 = 禁止TMR0模块
0 = 使能TMR0模块

寄存器19-3： PMD2：PMD控制寄存器2

U-0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	DACMD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD(1)
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **未实现：** 读为0

bit 6 **DACMD：** 禁止DAC位
1 = 禁止DAC模块
0 = 使能DAC模块

bit 5 **ADCMD：** 禁止ADCC位
1 = 禁止ADCC模块
0 = 使能ADCC模块

bit 4-3 **未实现：** 读为0

bit 2 **CMP2MD：** 禁止比较器CMP2位
1 = 禁止CMP2模块
0 = 使能CMP2模块

bit 1 **CMP1MD：** 禁止比较器CMP1位
1 = 禁止CMP1模块
0 = 使能CMP1模块

bit 0 **ZCDMD：** 禁止过零检测模块位(1)
1 = 禁止ZCD模块
0 = 使能ZCD模块

注 1： 由CONFIG2H中的$\overline{\text{ZCD}}$位决定。

寄存器19-4： PMD3：PMD控制寄存器3

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
PWM8MD	PWM7MD	PWM6MD	PWM5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **PWM8MD**：禁止脉宽调制器PWM8位
1 = 禁止PWM8模块
0 = 使能PWM8模块

bit 6 **PWM7MD**：禁止脉宽调制器PWM7位
1 = 禁止PWM7模块
0 = 使能PWM7模块

bit 5 **PWM6MD**：禁止脉宽调制器PWM6位
1 = 禁止PWM6模块
0 = 使能PWM6模块

bit 4 **PWM5MD**：禁止脉宽调制器PWM5位
1 = 禁止PWM5模块
0 = 使能PWM5模块

bit 3 **CCP4MD**：禁止捕捉/比较/PWM CCP4位
1 = 禁止CCP4模块
0 = 使能CCP4模块

bit 2 **CCP3MD**：禁止捕捉/比较/PWM CCP3位
1 = 禁止CCP3模块
0 = 使能CCP3模块

bit 1 **CCP2MD**：禁止捕捉/比较/PWM CCP2位
1 = 禁止CCP2模块
0 = 使能CCP2模块

bit 0 **CCP1MD**：禁止捕捉/比较/PWM CCP1位
1 = 禁止CCP1模块
0 = 使能CCP1模块

寄存器19-5： PMD4：PMD控制寄存器4

R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	U-0
CWG3MD	CWG2MD	CWG1MD	—	—	—	—	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **CWG3MD**：禁止CWG3模块位
1 = 禁止CWG3模块
0 = 使能CWG3模块

bit 6 **CWG2MD**：禁止CWG2模块位
1 = 禁止CWG2模块
0 = 使能CWG2模块

bit 5 **CWG1MD**：禁止CWG1模块位
1 = 禁止CWG1模块
0 = 使能CWG1模块

bit 4-0 **未实现**：读为0

寄存器19-6： PMD5：PMD控制寄存器5

U-0	U-0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	U2MD	U1MD	—	SPI1MD	I2C2MD	I2C1MD
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-6 **未实现：** 读为0

bit 5 **U2MD：** 禁止UART2位
1 = 禁止UART2模块
0 = 使能UART2模块

bit 4 **U1MD：** 禁止UART1位
1 = 禁止UART1模块
0 = 使能UART1模块

bit 3 **未实现：** 读为0

bit 2 **SPI1MD：** 禁止SPI1模块位
1 = 禁止SPI1模块
0 = 使能SPI1模块

bit 1 **I2C2MD：** 禁止I^2C2模块位
1 = 禁止I^2C2模块
0 = 使能I^2C2模块

bit 0 **I2C1MD：** 禁止I^2C1模块位
1 = 禁止I^2C1模块
0 = 使能I^2C1模块

寄存器19-7： PMD6：PMD控制寄存器6

U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	SMT2MD	SMT1MD	CLC4MD	CLC3MD	CLC2MD	CLC1MD	DSMMD
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **未实现：**读为0

bit 6 **SMT2MD：**禁止SMT2模块位
1 = 禁止SMT2模块
0 = 使能SMT2模块

bit 5 **SMT1MD：**禁止SMT1模块位
1 = 禁止SMT1模块
0 = 使能SMT1模块

bit 4 **CLC1MD：**禁止CLC4模块位
1 = 禁止CLC4模块
0 = 使能CLC4模块

bit 3 **CLC3MD：**禁止CLC3模块位
1 = 禁止CLC3模块
0 = 使能CLC3模块

bit 2 **CLC2MD：**禁止CLC2模块位
1 = 禁止CLC2模块
0 = 使能CLC2模块

bit 1 **CLC1MD：**禁止CLC1模块位
1 = 禁止CLC1模块
0 = 使能CLC1模块

bit 0 **DSMMD：**禁止数据信号调制器位
1 = 禁止DSM模块
0 = 使能DSM模块

寄存器19-8： PMD7：PMD控制寄存器7

R/W-0/0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
CANMD	—	—	—	—	—	DMA2MD	DMA1MD
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **CANMD**：禁止CAN模块位
1 = 禁止CAN模块
0 = 使能CAN模块

bit 6-2 **未实现**：读为0

bit 1 **DMA2MD**：禁止DMA2模块位
1 = 禁止DMA2模块
0 = 使能DMA2模块

bit 0 **DMA1MD**：禁止DMA1模块位
1 = 禁止DMA1模块
0 = 使能DMA1模块

表19-1： 与外设模块禁止相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
PMD0	SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD	275
PMD1	NCO1MD	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	276
PMD2	—	DACMD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD	277
PMD3	PWM8MD	PWM7MD	PWM6MD	PWM5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	278
PMD4	CWG3MD	CWG2MD	CWG1MD	—	—	—	—	—	279
PMD5	—	—	U2MD	U1MD	—	SPI1MD	I2C2MD	I2C1MD	280
PMD6	—	SMT2MD	SMT1MD	CLC4MD	CLC3MD	CLC2MD	CLC1MD	DSMMD	280
PMD7	CANMD	—	—	—	—	—	DMA2MD	DMA1MD	282

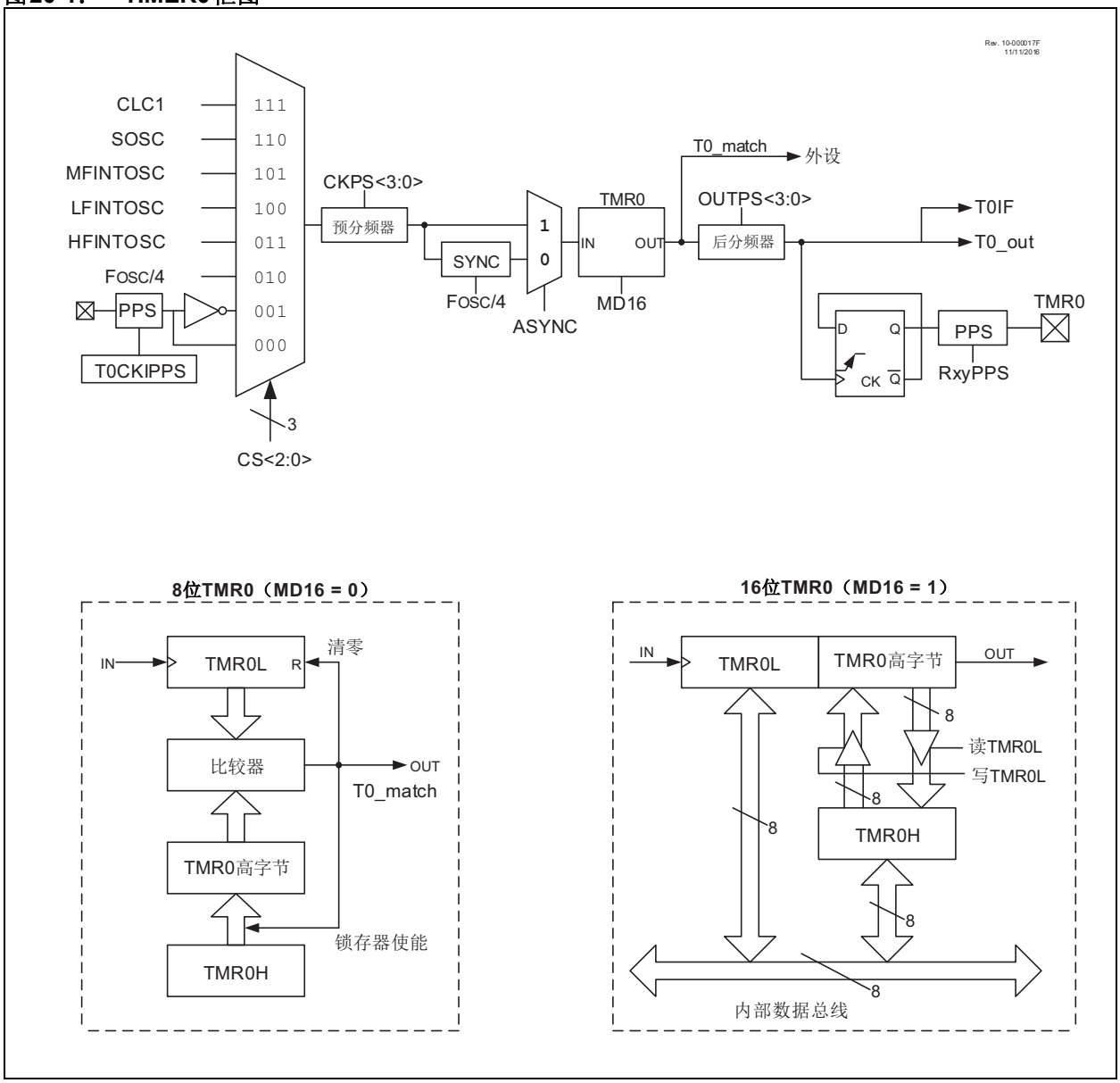
图注： — = 未实现位，读为0。外设模块禁止不使用阴影单元。

20.0 TIMER0模块

Timer0模块是8/16位定时器/计数器，具有以下特性：

- 16位定时器/计数器
- 带有可编程周期的8位定时器/计数器
- 同步或异步操作
- 可选择的时钟源
- 可编程预分频器
- 可编程后分频器
- 休眠模式期间工作
- 匹配或溢出时中断
- （通过PPS）在I/O引脚上输出或输出至其他外设

图20-1： TIMER0框图



20.1 Timer0工作原理

Timer0可用作8位定时器/计数器或16位定时器/计数器。模式通过T0CON寄存器的MD16位选择。

20.1.1 16位模式

寄存器对TMR0H:TMR0L在时钟源的上升沿递增。时钟输入上的15位预分频器提供几个预分频选项（见T0CON1寄存器中的预分频比控制位CKPS<3:0>）。

20.1.1.1 Timer0的16位读写模式

在16位模式下，为了避免读取高位字寄存器和低位字寄存器之间的计满返回，TMR0H寄存器是Timer0实际高字节的缓冲副本，不能直接读写（见图20-1）。在读TMR0L时使用Timer0高字节的内容更新TMR0H。这样可以一次读取Timer0的全部16位，而无需验证读到的高字节和低字节的有效性（在高、低字节分两次连续读取的情况下，由于可能存在计满返回，因此需要验证读到字节的有效性）。

同样，写入Timer0的高字节也是通过TMR0H缓冲寄存器来操作的。在写入TMR0L的同时，使用TMR0H的内容更新Timer0的高字节。这样一次就可以完成Timer0全部16位的更新。

20.1.2 8位模式

在8位模式下，在每个时钟周期，TMR0L的值都会与周期缓冲区（TMR0H的副本）的值进行比较。当两个值匹配时，发生以下事件：

- TMR0_out变为高电平一个预分频时钟周期
- 复位TMR0L
- TMR0H的内容复制到周期缓冲区

在8位模式下，TMR0L和TMR0H寄存器均可直接读写。在任何器件复位时，TMR0L寄存器都会清零，而TMR0H寄存器则初始化为FFh。

发生以下事件时，预分频器和后分频器计数器均会清零：

- 对TMR0L寄存器进行写操作
- 对T0CON0或T0CON1寄存器进行写操作
- 任何器件复位——上电复位（POR）、$\overline{\text{MCLR}}$复位、看门狗定时器复位（WDTR）或
- 欠压复位（BOR）

20.1.3 计数器模式

在计数器模式下，通常通过将T0CON1寄存器的CKPS位设置为0000来禁止预分频器。计数器在时钟输入（或预分频器输出，如果使用预分频器的话）的每个上升沿递增1。

20.1.4 定时器模式

在定时器模式下，只要存在有效时钟信号且T0CON1寄存器（寄存器20-2）的CKPS位设置为0000，Timer0模块将在每个指令周期递增。当增加预分频器时，定时器将以基于预分频值的速率递增。

20.1.5 异步模式

当T0CON1寄存器的ASYNC位置1（ASYNC = 1）时，计数器在输入源（或预分频器的输出，如果使用的话）的每个上升沿递增。异步模式允许计数器在休眠模式期间继续运行，前提是时钟也在休眠期间继续运行。

20.1.6 同步模式

当T0CON1寄存器的ASYNC位清零（ASYNC = 0）时，计数器时钟与系统时钟同步（FOSC/4）。当在同步模式下工作时，计数器时钟频率无法超过FOSC/4。

20.2 时钟源选择

T0CON1寄存器的CS<2:0>位用于选择Timer0的时钟源。寄存器20-2显示了时钟源选择。

20.2.1 内部时钟源

当选择内部时钟源时，Timer0作为定时器工作，并将在时钟源上升沿的整数倍（由Timer0预分频器决定）处递增。

20.2.2 外部时钟源

当选择外部时钟源时，Timer0模块可以作为定时器或计数器工作，Timer0将在外部时钟源上升沿的整数倍（由Timer0预分频器决定）处递增。

20.3 可编程预分频器

软件可编程的预分频器只能用于Timer0。Timer0有16个预分频比选项，范围从1:1到1:32768（2的次幂）。预分频值可通过T0CON1寄存器的CKPS<3:0>位进行选择。

预分频器不可直接读写。可通过写入TMR0L寄存器或T0CON0/T0CON1寄存器或任何复位来清零预分频器寄存器。

20.4 可编程后分频器

软件可编程的后分频器（输出分频器）只能用于Timer0。Timer0有16个后分频比选项，范围从1:1至1:16。后分频值可通过T0CON0寄存器的OUTPS位进行选择。

后分频器不可直接读写。可通过写入TMR0L寄存器或T0CON0/T0CON1寄存器或任何复位来清零后分频器寄存器。

20.5 休眠期间的操作

当同步工作时，Timer0将暂停。当异步工作时，Timer0将继续递增并将器件从休眠状态唤醒（如果允许Timer0中断），前提是输入时钟源处于活动状态。

20.6 Timer0中断

Timer0中断标志位（TMR0IF）在发生以下任一条件时置1：

- 8位TMR0L与TMR0H值匹配
- 16位TMR0从FFFFh计满返回

当后分频比位（OUTPS）设置为1:1操作（无分频）时，T0IF标志位将在每次发生TMR0匹配或计满返回时置1。通常，TMR0IF标志位将在每发生OUTPS +1次匹配或计满返回时置1。

如果允许Timer0中断（PIE3寄存器的TMR0IE位 = 1），CPU将发生中断且器件从休眠状态唤醒（更多详细信息，请参见**第20.2节“时钟源选择”**）。

20.7 Timer0输出

可通过RxyPPS输出选择寄存器将Timer0输出输送到任一I/O引脚（有关详细信息，请参见**第17.0节“外设引脚选择（PPS）模块”**）。Timer0输出也可供其他外设使用，例如作为模数转换器的自动转换触发信号。最后，Timer0输出可由软件通过T0CON0寄存器（寄存器20-1）的Timer0输出位（OUT）监视。

当在8位模式下TMR0L与PR0（TMR0的周期寄存器）发生匹配或在16位模式下TMR0计满返回时，TMR0_out将是一个后分频时钟周期的脉冲。Timer0输出的占空比为50%，在每个TMR0_out时钟上升沿翻转。

20.8 寄存器定义：Timer0控制

寄存器20-1： T0CON0：TIMER0控制寄存器0

R/W-0/0	U-0	R-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN	—	OUT	MD16	OUTPS<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN**：TMR0使能位
1 = 模块已使能并且正在工作
0 = 模块已禁止并且处于最低功耗模式

bit 6 **未实现**：读为0

bit 5 **OUT**：TMR0输出位（只读）
TMR0输出位

bit 4 **MD16**：TMR0用作16位定时器选择位
1 = TMR0是16位定时器
0 = TMR0是8位定时器

bit 3-0 **OUTPS<3:0>**：TMR0输出后分频比（分频比）选择位
1111 = 1:16后分频比
1110 = 1:15后分频比
1101 = 1:14后分频比
1100 = 1:13后分频比
1011 = 1:12后分频比
1010 = 1:11后分频比
1001 = 1:10后分频比
1000 = 1:9后分频比
0111 = 1:8后分频比
0110 = 1:7后分频比
0101 = 1:6后分频比
0100 = 1:5后分频比
0011 = 1:4后分频比
0010 = 1:3后分频比
0001 = 1:2后分频比
0000 = 1:1后分频比

寄存器20-2： T0CON1：TIMER0控制寄存器1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
CS<2:0>			ASYNC	CKPS<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **CS<2:0>**：Timer0时钟源选择位
111 = CLC1
110 = SOSC
101 = MFINTOSC（500 kHz）
100 = LFINTOSC
011 = HFINTOSC
010 = FOSC/4
001 = 通过T0CKIPPS选择的引脚（反相）
000 = 通过T0CKIPPS选择的引脚（同相）

bit 4 **ASYNC**：TMR0输入异步使能位
1 = TMR0计数器输入与系统时钟不同步
0 = TMR0计数器输入与FOSC/4同步

bit 3-0 **CKPS<3:0>**：预分频比选择位
1111 = 1:32768
1110 = 1:16384
1101 = 1:8192
1100 = 1:4096
1011 = 1:2048
1010 = 1:1024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

寄存器20-3： TMR0L：TIMER0计数寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TMR0L<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **TMR0L<7:0>**：TMR0计数器bit <7:0>

寄存器20-4： TMR0H：TIMER0周期寄存器

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TMR0H<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 当 MD16 = 0时
PR0<7:0>：TMR0周期寄存器bit <7:0>
当 MD16 = 1时
TMR0H<15:8>：TMR0计数器bit <15:8>

表20-1： 与TIMER0相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
T0CON0	EN	—	OUT	MD16	OUTPS<3:0>				286
T0CON1	CS<2:0>			ASYNC	CKPS<3:0>				287
TMR0L	TMR0L<7:0>								288
TMR0H	TMR0H<15:8>								288

图注： — = 未实现位，读为0。Timer0不使用阴影单元。

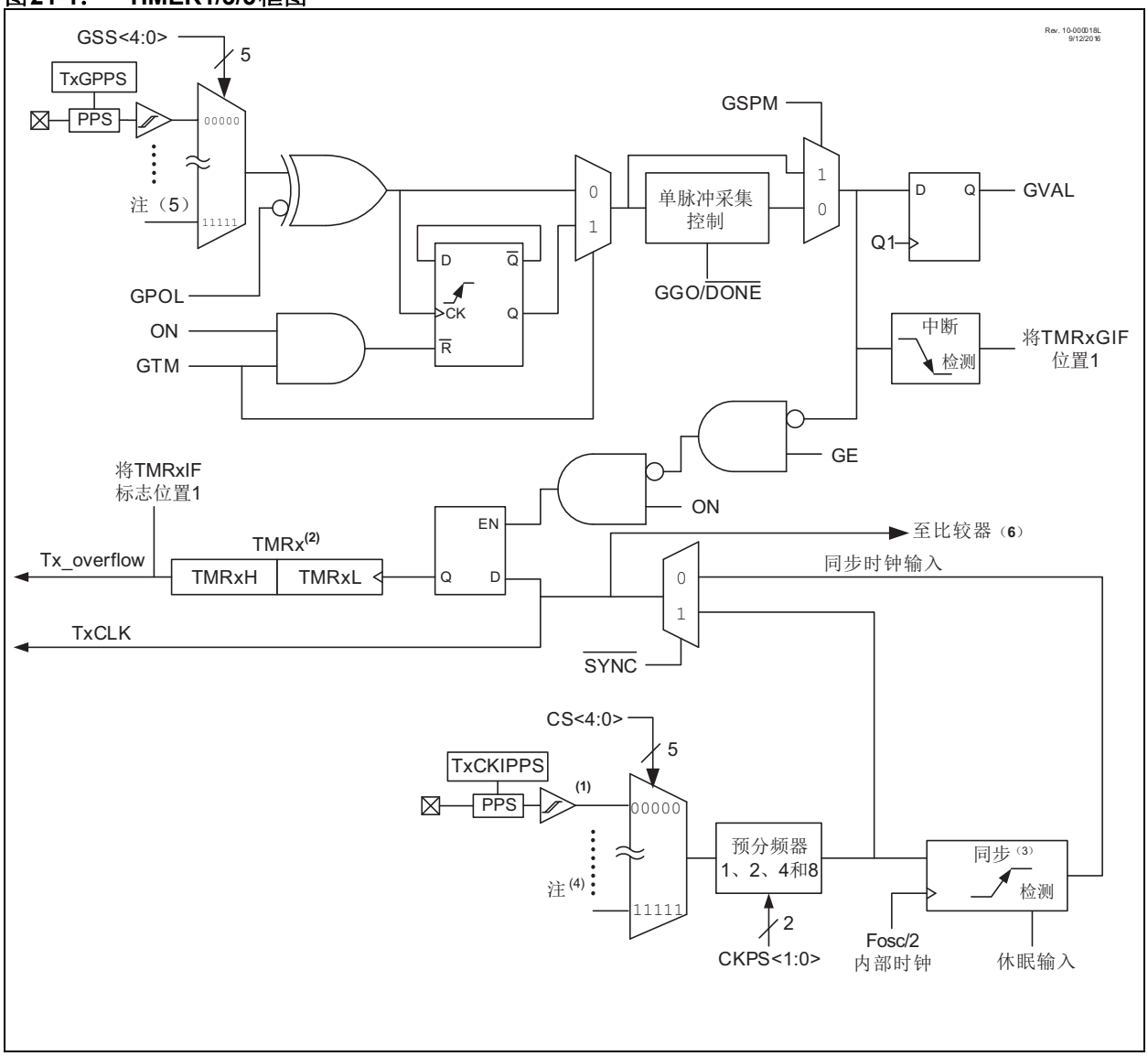
21.0 带门控控制的TIMER1/3/5模块

Timer1/3/5模块是16位定时器/计数器，具有以下特性：

- 16位定时器/计数器寄存器对（TMRxH:TMRxL）
- 可编程内部或外部时钟源
- 2位预分频器
- 专用辅助32 kHz振荡器电路
- 可选的同步比较器输出
- 多个Timer1/3/5门控（计数使能）源
- 溢出时产生中断
- 溢出触发唤醒（仅限外部时钟，异步模式）
- 16位读/写操作
- 用于CCP模块的捕捉/比较功能的时基
- 特殊事件触发器（与CCP配合工作）
- 可选择的门控源极性
- 门控翻转模式
- 门控单脉冲模式
- 门控值状态
- 门控事件中断

图21-1给出了Timer1/3/5模块的框图。

图21-1： TIMER1/3/5框图



21.1 Timer1/3/5工作原理

Timer1/3/5模块是16位递增计数器，可通过TMRxH:TMRxL寄存器对访问。写TMRxH或TMRxL会直接更新计数器。

与内部时钟源一起使用时，模块用作定时器，并在每个指令周期递增。与外部时钟源一起使用时，模块可用作定时器或计数器，在外部时钟源的每个选定边沿递增。

Timer1/3/5分别通过配置TxCON和TxGCON寄存器中的ON和GE位使能。表21-1显示了Timer1/3/5使能选择。

表21-1： TIMER1/3/5使能选择

ON	GE	Timer1/3/5工作原理
1	1	使能计数
1	0	始终开启
0	1	关闭
0	0	关闭

21.2 时钟源选择

TMRxCLK寄存器（寄存器21-3）的CS<4:0>位用于选择Timer1/3/5的时钟源。这五个TMRxCLK位可用于选择多种同步和异步时钟源。寄存器21-3显示了时钟源选择。

21.2.1 内部时钟源

当选择内部时钟源时，TMRxH:TMRxL寄存器对的递增频率将为FOSC的整数倍（取决于Timer1/3/5预分频器）。

选择FOSC内部时钟源时，Timer1/3/5寄存器的值将在每个指令时钟周期中递增4次。由于这个原因，在读取Timer1/3/5值时，分辨率将会出现2个LSB的误差。为了利用Timer1/3/5的全部分辨率，必须使用异步输入信号来对Timer1/3/5时钟输入进行门控。

可以在Timer1/3/5门控处使用以下异步源：

- TxGPPS引脚上的异步事件
- TMR0OUT
- TMR1/3/5OUT（不包括正在使用的TMR）
- TMR 2/4/6OUT（后分频）
- CMP1/2OUT
- SMT1_match
- NCO1OUT
- PWM3/4 OUT
- CCP1/2/3/4 OUT
- CLC1/2/3/4 OUT
- ZCDOUT

注： 在计数器模式下，发生以下任何一个或多个情况后，计数器在首个上升沿递增前，必须先经过一个下降沿：

- POR后使能Timer1/3/5
- 写TMRxH或TMRxL
- 禁止Timer1/3/5
- TxCKI为高电平时Timer1/3/5被禁止（TMRxON = 0），然后在TxCKI为低电平时Timer1/3/5被使能（TMRxON = 1）。

21.2.2 外部时钟源

当选择外部时钟源时，Timer1/3/5模块可以作为定时器或计数器工作。

使能Timer1/3/5计数时，在TxCKIPPS引脚的外部时钟输入的上升沿递增。该外部时钟源既可以与单片机系统时钟同步，也可以异步运行。

当用作带时钟振荡器的定时器时，可将外部32.768 kHz晶振与专用辅助内部振荡器电路结合使用。

21.3 Timer1/3/5预分频器

Timer1/3/5有4个预分频比选项，允许对时钟输入进行1、2、4或8分频。TxCON寄存器的CKPS位控制预分频器计数器。对预分频器计数器不能直接进行读写操作；但是，通过写入TMRxH或TMRxL可将预分频器计数器清零。

21.4 异步计数器模式下的Timer1/3/5操作

如果TxCON寄存器的控制位$\overline{\text{SYNC}}$置1，外部时钟输入将不同步。定时器异步于内部相位时钟进行递增计数。如果选择了外部时钟源，定时器在休眠期间将继续运行，并在溢出时产生中断以唤醒处理器。但是，用软件对定时器进行读/写操作时，要特别当心（见**第21.4.1节“在异步计数器模式下读写Timer1/3/5”**）。

注：	当从同步切换到异步操作时，可能会跳过一次递增。当从异步切换到同步操作时，可能会产生一次额外递增。

21.4.1 在异步计数器模式下读写TIMER1/3/5

当定时器采用外部异步时钟运行时，对TMRxH或TMRxL的读操作将确保为有效读操作（由硬件实现）。但是，用户应该注意的是通过读两个8位值来读取16位定时器本身就会产生某些问题，这是因为定时器可能在两次读操作之间产生溢出。对于写操作，建议用户直接停止定时器，然后写入所需的值。如果定时器寄存器正进行递增计数，对定时器寄存器进行写操作可能会导致写争用，这可能在TMRxH:TMRxL寄存器对中产生不可预测的值。

21.5 Timer1/3/5 16位读/写模式

Timer1/3/5可配置为同时对8位TMRxL和TMRxH寄存器读/写所有16位数据。通过将TxCON寄存器的RD16位置1，可以使能16位读/写操作。

要实现该功能，TMRxH寄存器值将映射到TMRxH缓冲寄存器。在16位模式下，TMRxH寄存器不能直接读写，所有读写操作都是通过使用该TMRxH缓冲寄存器来实现的。

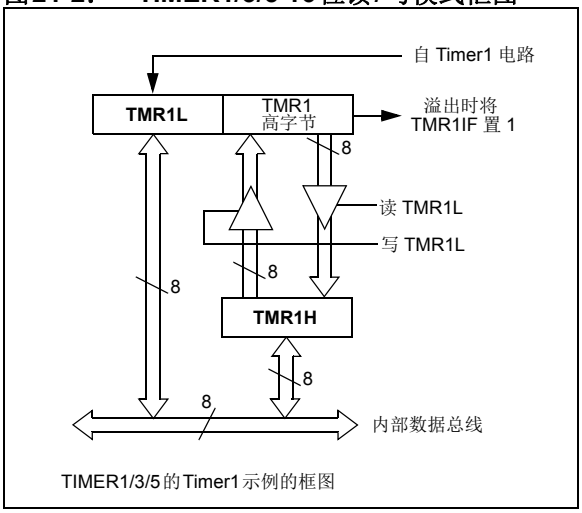
当请求读取TMRxL寄存器时，TMRxH寄存器的值会同时装入TMRxH缓冲寄存器中。当请求读取TMRxH寄存器时，该值则由TMRxH缓冲寄存器提供。这样，用户便能够精确而及时地从单个实例中读取所有16位Timer1/3/5值。更多详细信息，请参见图21-2中的框图。

相比之下，未处于16位模式时，用户必须单独读取每个寄存器，并确定这些值是否已因读操作之间可能发生的计满返回而变为无效。

当请求写入TMRxL寄存器时，会同时使用TMRxH寄存器的内容更新TMRxH缓冲寄存器。在执行TMRxL寄存器写请求之前，必须将TMRxH的值预先装入TMRxH缓冲寄存器中。这样用户就可以同时写入TMRxL:TMRxH寄存器对的全部16位。

任何直接写入TMRxH的请求都不会将Timer1/3/5预分频值清零。预分频值只能通过对TMRxL寄存器的写请求清零。

图21-2： TIMER1/3/5 16位读/写模式框图



TIMER1/3/5的Timer1示例的框图

21.6 Timer1/3/5门控

Timer1/3/5可配置为自由计数或使用Timer1/3/5门控电路使能和禁止计数。这也称为Timer1/3/5门控使能。

Timer1/3/5门控也可由多个可选择源驱动。

21.6.1 TIMER1/3/5门控使能

通过将TxGCON寄存器的TMRxGE位置1使能Timer1/3/5门控使能模式。使用TxGCON寄存器的TxGPOL位来配置Timer1/3/5门控使能模式的极性。

使能Timer1/3/5门控使能模式时，Timer1/3/5将在Timer1/3/5时钟源的上升沿递增。当Timer1/3/5门控信号无效时，定时器不会递增并且将保持当前计数。时序详细信息请参见图21-4。

表21-2： TIMER1/3/5门控使能选择

TMRxCLK	TxGPOL	TxG	Timer1/3/5 工作原理
↑	1	1	计数
↑	1	0	保持计数
↑	0	1	保持计数
↑	0	0	计数

21.6.2 TIMER1/3/5门控源选择

可以使用TMRxGATE寄存器（寄存器21-4）的GSS<4:0>位选择Timer1/3/5的门控源。门控源的极性选择由TxGCON寄存器（寄存器21-2）的TxGPOL位控制。

上述任一信号都可用于触发门控。CMPx的输出可以与Timer1/3/5时钟同步或保持异步。更多信息，请参见**第39.3.1节“比较器输出同步”**。

21.6.3 TIMER1/3/5门控翻转模式

使能Timer1/3/5门控翻转模式时，可以测量门控信号的每个上升沿和下降沿之间的持续时间。

Timer1/3/5门控源经由一个触发器输送到Timer1/3/5，该触发器在信号的每个递增边沿改变状态。时序详细信息请参见图21-5。

Timer1/3/5门控翻转模式通过将TxGCON寄存器的GTM位置1使能。GTM位清零时，将清零触发器并保持清零。这对于控制测量哪个边沿是必需的。

注：	使能翻转模式的同时更改门控信号的极性可能会导致操作不确定。

21.6.4 TIMER1/3/5门控单脉冲模式

使能Timer1/3/5门控单脉冲模式时，可能会捕捉到一个单脉冲门控事件。Timer1/3/5门控单脉冲模式首先通过将TxGCON寄存器中的GSPM位置1来使能。接下来必须将TxGCON寄存器中的GGO/$\overline{\text{DONE}}$位置1。Timer1/3/5将在门控信号的下一个递增边沿完全使能。在脉冲的下一个边沿，将自动清零GGO/$\overline{\text{DONE}}$位。不允许其他门控事件递增Timer1/3/5，直到GGO/$\overline{\text{DONE}}$位再次由软件置1。

清零TxGCON寄存器的TxGSPM位也会清零GGO/$\overline{\text{DONE}}$位。时序详细信息请参见图21-6。

同时使能翻转模式和单脉冲模式将允许两种模式协同工作。这样就可以测量Timer1/3/5门控源的周期。时序详细信息请参见图21-7。

21.6.5 TIMER1/3/5门控值状态

使用Timer1/3/5门控值状态时，可读取门控信号的最新电平。该值保存在TxGCON寄存器的GVAL位中。即使Timer1/3/5门控未使能（GE位清零），GVAL位也是有效的。

21.6.6 TIMER1/3/5门控事件中断

允许Timer1/3/5门控事件中断时，可在门控事件完成时产生一个中断。出现GVAL的下降沿时，相应PIR寄存器中的TMRxGIF标志位将置1。如果相应PIE寄存器中的TMRxGIE位置1，则会识别出一个中断。

即使Timer1/3/5门控未使能（GE位清零），TMRxGIF标志位也是有效的。

关于为Timer1/3/5门控事件中断选择高优先级状态或低优先级状态的更多信息，请参见**第9.0节“中断控制器”**。

21.7 Timer1/3/5中断

Timer1/3/5寄存器对（TMRxH:TMRxL）递增到FFFFh，然后计满返回到0000h。当Timer1/3/5计满返回时，相应PIR寄存器的Timer1/3/5中断标志位将置1。为允许计满返回时的中断，必须将以下位置1：

- TxCON寄存器的ON位
- 相应PIE寄存器的TMRxIE位
- INTCON0寄存器的GIE/GIEH位

在中断服务程序中将TMRxIF位清零清除中断。

关于为Timer1/3/5溢出中断选择高优先级状态或低优先级状态的更多信息，请参见**第9.0节“中断控制器”**。

注：	在允许中断前，应将TMRxH:TMRxL寄存器对以及TMRxIF位清零。

21.8 休眠期间的Timer1/3/5工作

只有在设置为异步计数器模式时，Timer1/3/5才能在休眠模式下工作。在该模式下，可使用外部晶振或时钟源使计数器递增计数。要设置定时器以唤醒器件：

- 必须将TxCON寄存器的ON位置1
- 必须将相应PIE寄存器的TMRxIE位置1
- 必须将TxCON寄存器的$\overline{\text{SYNC}}$位置1
- 配置TMRxCLK寄存器，将辅助振荡器用作时钟源
- 使能OSCEN寄存器（寄存器7-7）的SOSCEN位

器件将在溢出时被唤醒并执行下一条指令。如果将INTCON0寄存器的GIE/GIEH位置1，器件将调用中断服务程序。

无论$\overline{\text{SYNC}}$位的设置如何，辅助振荡器都会在休眠模式下继续工作。

21.9 CCP捕捉/比较时基

当工作在捕捉或比较模式下时，CCP模块使用TMRxH:TMRxL寄存器对作为时基。

在捕捉模式下，当发生配置的事件时，TMRxH:TMRxL寄存器对中的值被复制到CCPRxH:CCPRxL寄存器对中。

在比较模式下，当CCPRxH:CCPRxL寄存器对中的值与TMRxH:TMRxL寄存器对中的值匹配时触发事件。该件可以是特殊事件触发信号。

更多信息，请参见**第23.0节“捕捉/比较/PWM模块”**。

21.10 CCP特殊事件触发器

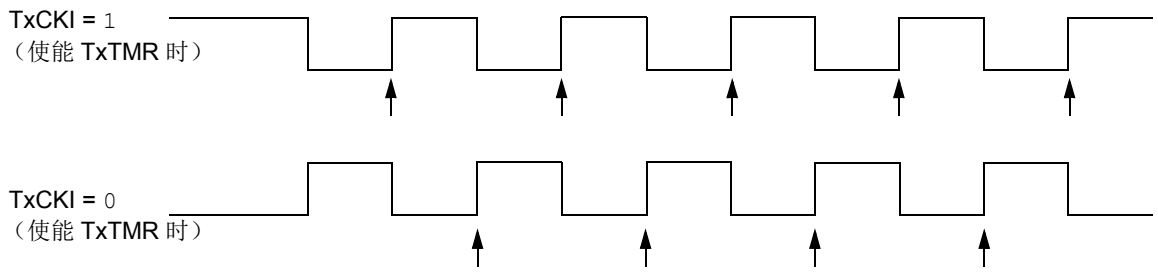
当任一CCP配置为触发特殊事件时，触发信号会将TMRxH:TMRxL寄存器对清零。该特殊事件不会引起Timer1/3/5中断。CCP模块仍可配置为产生CCP中断。

在这种工作模式下，CCPRxH:CCPRxL寄存器对变成了Timer1/3/5的周期寄存器。

Timer1/3/5应进行同步，并应选择FOSC/4作为时钟源，以利用特殊事件触发信号。Timer1/3/5的异步操作会导致错过特殊事件触发信号。

如果对TMRxH或TMRxL的写操作与来自CCP的特殊事件触发信号同时发生，则写操作优先。

图21-3： TIMER1/3/5递增边沿



注 1： 箭头指示计数器递增。

2： 在计数器模式下，计数器在时钟的首个上升沿递增之前，必须先经过一个下降沿。

图21-4： TIMER1/3/5门控使能模式

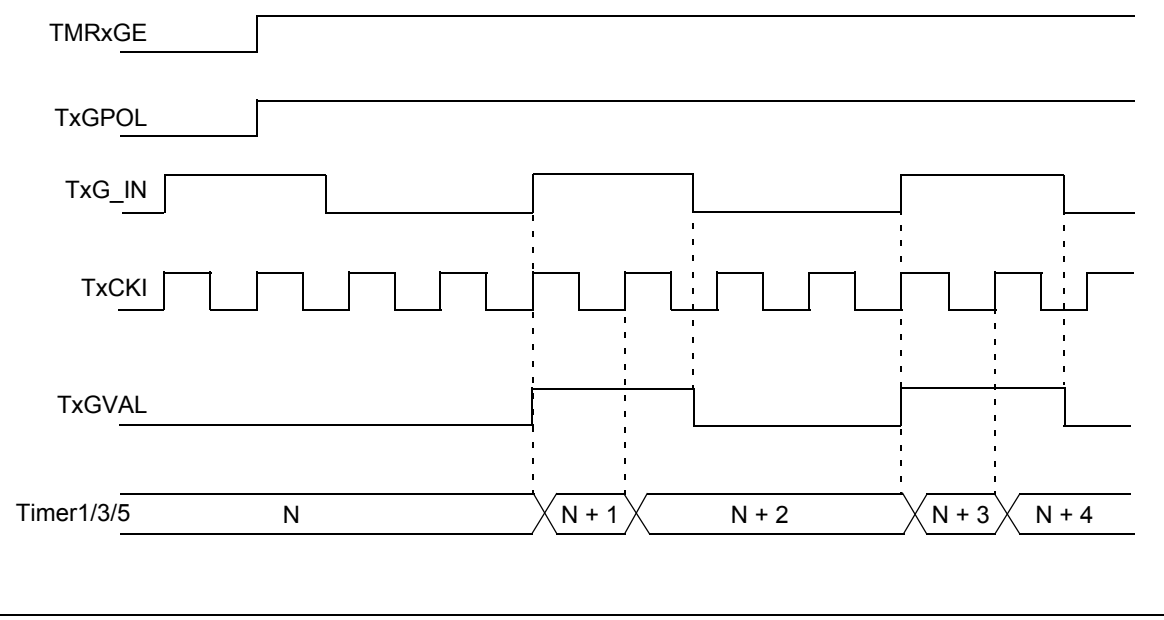


图21-5： TIMER1/3/5门控翻转模式

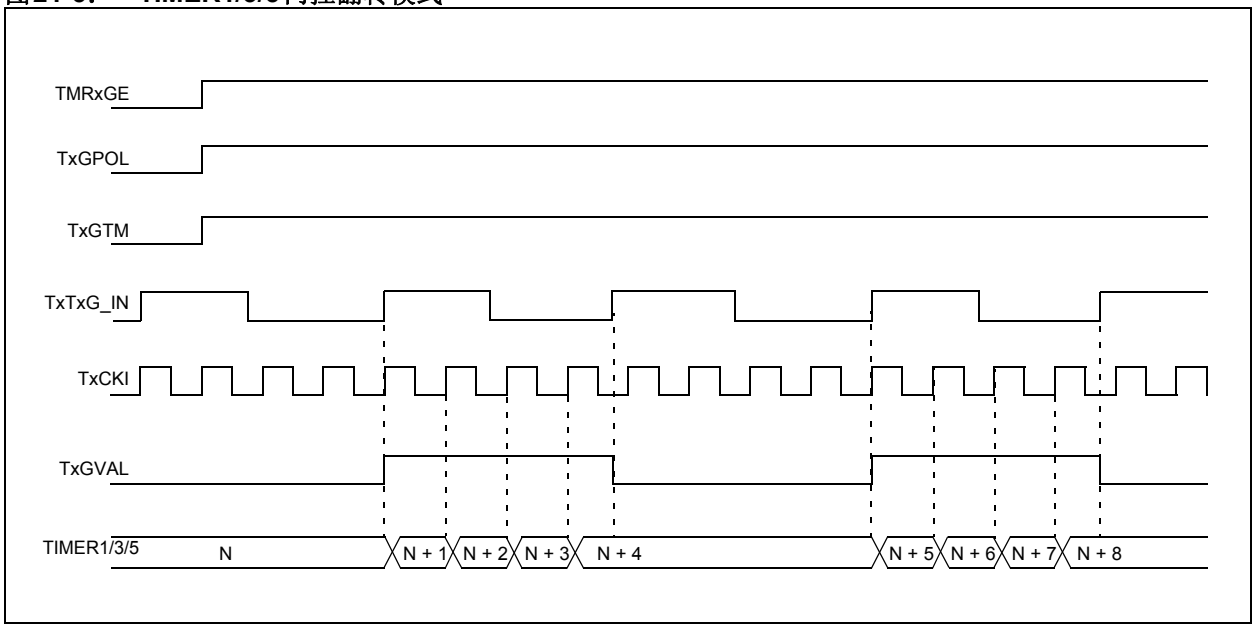


图21-6： TIMER1/3/5门控单脉冲模式

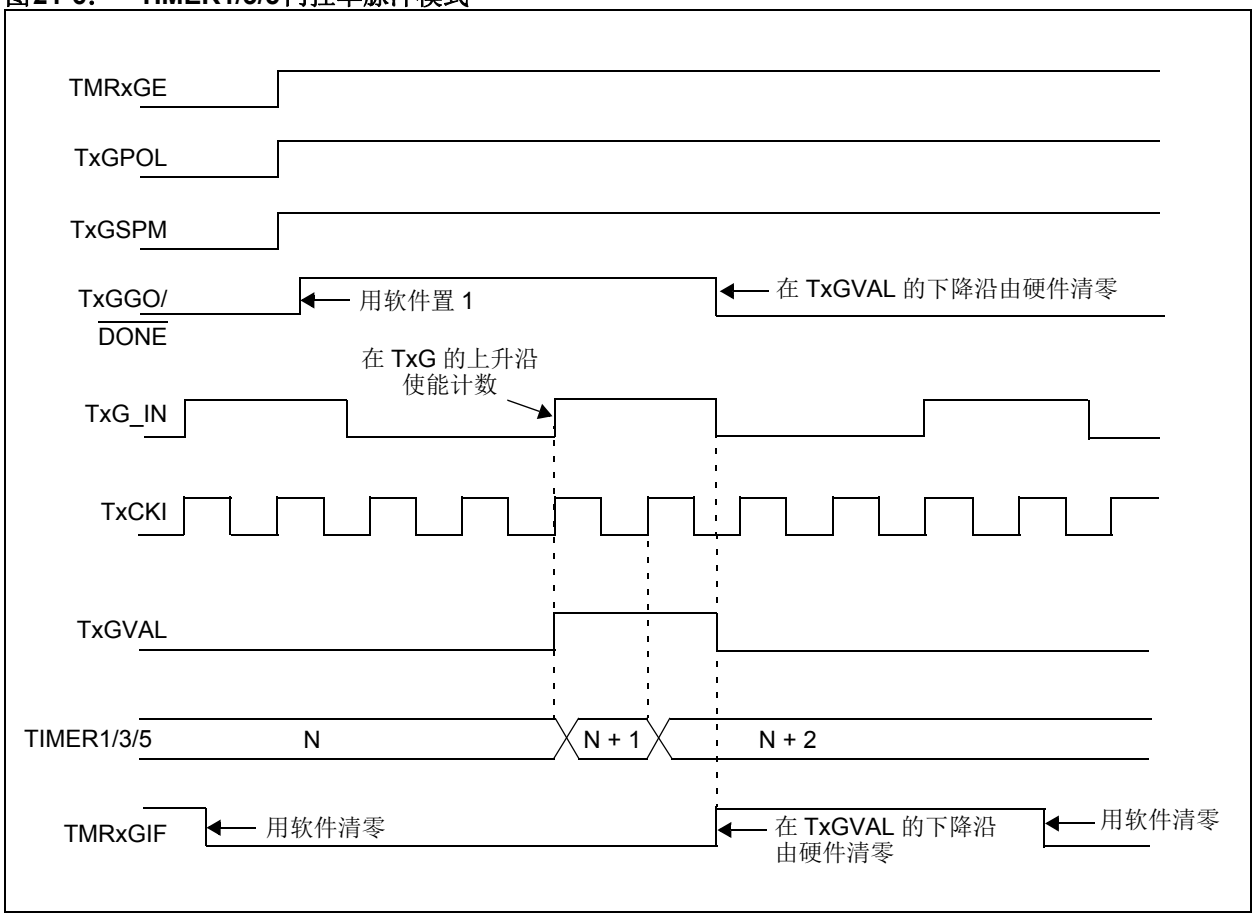
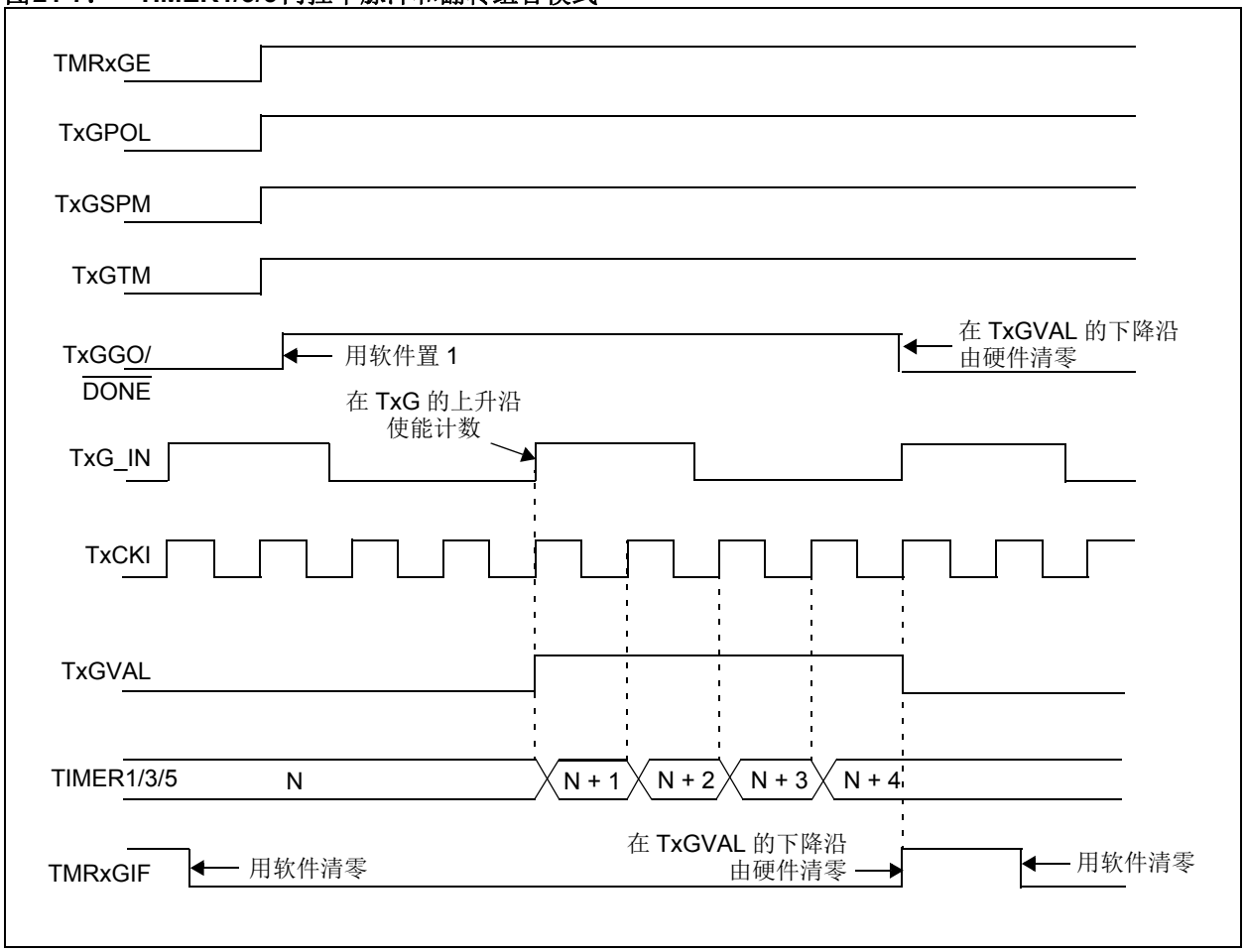


图21-7： TIMER1/3/5门控单脉冲和翻转组合模式



21.11 外设模块禁止

当外设模块未使用或无效时，可以通过将PMD寄存器中的模块禁止位置1来禁止该模块。这会将功耗降至绝对最小值。将PMD位置1可以使模块保持在复位状态，并断开模块的时钟源。Timer1、Timer3和Timer5的模块禁止位（TMR1MD、TMR3MD和TMR5MD）位于相应PMD寄存器中。更多信息，请参见**第19.0节“外设模块禁止（PMD）”**。

21.12 寄存器定义：Timer1/3/5

Timer1/3/5的长位名称前缀如下所示。更多信息，请参见**第1.3.2.2节“长位名称”**。

外设	位名称前缀
Timer1	T1
Timer3	T3
Timer5	T5

寄存器21-1： TXCON：TIMERx控制寄存器

U-0	U-0	R/W-0/u	R/W-0/u	U-0	R/W-0/u	R/W-0/0	R/W-0/u
—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	u = 不变

bit 7-6 **未实现：**读为0

bit 5-4 **CKPS<1:0>：**Timerx输入时钟预分频比选择位
11 = 1:8预分频比
10 = 1:4预分频比
01 = 1:2预分频比
00 = 1:1预分频比

bit 3 **未实现：**读为0

bit 2 **$\overline{\text{SYNC}}$：**Timerx外部时钟输入同步控制位
TMRxCLK = FOSC/4或FOSC：
忽略此位。Timer1按原样使用传入时钟。
否则：
1 = 不同步外部时钟输入
0 = 将外部时钟输入与系统时钟同步

bit 1 **RD16：**16位读/写模式使能位
1 = 使能通过一次16位操作读/写Timerx的寄存器
0 = 使能通过两次8位操作读/写Timerx的寄存器

bit 0 **ON：**Timerx使能位
1 = 使能Timerx
0 = 禁止Timerx

寄存器21-2： TxGCON：TIMERx门控寄存器

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R-x	U-0	U-0
GE	GPOL	GTM	GSPM	GGO/$\overline{\text{DONE}}$	GVAL	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **GE**：Timerx门控使能位
如果TMRxON = 1：
1 = Timerx计数由Timerx门控功能控制
0 = Timerx始终计数
如果TMRxON = 0：
该位被忽略

bit 6 **GPOL**：Timerx门控极性位
1 = Timerx门控为高电平有效（当门控信号为高电平时Timerx计数）
0 = Timerx门控为低电平有效（当门控信号为低电平时Timerx计数）

bit 5 **GTM**：Timerx门控翻转模式位
1 = 使能Timerx门控翻转模式
0 = 禁止Timerx门控翻转模式并清除翻转触发器
Timerx门控触发器在每个上升沿翻转

bit 4 **GSPM**：Timerx门控单脉冲模式位
1 = 使能Timerx门控单脉冲模式，并在使用该模式时控制Timerx门控
0 = 禁止Timerx门控单脉冲模式

bit 3 **GGO/$\overline{\text{DONE}}$**：Timerx门控单脉冲采集状态位
1 = Timerx门控单脉冲采集就绪，正在等待一个边沿
0 = Timerx门控单脉冲采集已经结束或尚未开始。
当TxGSPM位清零时，该位自动清零。

bit 2 **GVAL**：Timerx门控当前状态位
指示可提供给TMRxH:TMRxL的Timerx门控信号的当前状态
不受Timerx门控使能（TMRxGE）的影响

bit 1-0 **未实现**：读为0

寄存器21-3： TxCLK：TIMERx时钟寄存器

U-0	U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	CS<4:0>				
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	u = 不变

bit 7-5 **未实现：**读为0

bit 4-0 **CS<4:0>：**Timerx时钟源选择位

CS	Timer1	Timer3	Timer5
	时钟源	时钟源	时钟源
11111-10001	保留	保留	保留
10000	CLC4	CLC4	CLC4
01111	CLC3	CLC3	CLC3
01110	CLC2	CLC2	CLC2
01101	CLC1	CLC1	CLC1
01100	TMR5溢出	TMR5溢出	保留
01011	TMR3溢出	保留	TMR3溢出
01010	保留	TMR1溢出	TMR1溢出
01001	TMR0溢出	TMR0溢出	TMR0溢出
01000	CLKREF	CLKREF	CLKREF
00111	SOSC	SOSC	SOSC
00110	MFINTOSC（32 kHz）	MFINTOSC（32 kHz）	MFINTOSC（32 kHz）
00101	MFINTOSC（500 kHz）	MFINTOSC（500 kHz）	MFINTOSC（500 kHz）
00100	LFINTOSC	LFINTOSC	LFINTOSC
00011	HFINTOSC	HFINTOSC	HFINTOSC
00010	Fosc	Fosc	Fosc
00001	Fosc/4	Fosc/4	Fosc/4
00000	T1CKIPPS	T3CKIPPS	T5CKIPPS

寄存器21-4： TxGATE：TIMERx门控ISM寄存器

U-0	U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	GSS<4:0>				
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	u = 不变

bit 7-5 **未实现：**读为0

bit 4-0 **GSS<4:0>：**Timerx门控源选择位

GSS	Timer1	Timer3	Timer5
	门控源	门控源	门控源
11111-11011	保留	保留	保留
11010	CLC4_out	CLC4_out	CLC4_out
11001	CLC3_out	CLC3_out	CLC3_out
11000	CLC2_out	CLC2_out	CLC2_out
10111	CLC1_out	CLC1_out	CLC1_out
10110	ZCDOUT	ZCDOUT	ZCDOUT
10101	CMP2OUT	CMP2OUT	CMP2OUT
10100	CMP1OUT	CMP1OUT	CMP1OUT
10011	NCO1OUT	NCO1OUT	NCO1OUT
10010-10001	保留	保留	保留
10000	PWM8OUT	PWM8OUT	PWM8OUT
01111	PWM7OUT	PWM7OUT	PWM7OUT
01110	PWM6OUT	PWM6OUT	PWM6OUT
01101	PWM5OUT	PWM5OUT	PWM5OUT
01100	CCP4OUT	CCP4OUT	CCP4OUT
01011	CCP3OUT	CCP3OUT	CCP3OUT
01010	CCP2OUT	CCP2OUT	CCP2OUT
01001	CCP1OUT	CCP1OUT	CCP1OUT
01000	SMT1_match	SMT1_match	SMT1_match
00111	TMR6OUT（后分频）	TMR6OUT（后分频）	TMR6OUT（后分频）
00110	TMR5溢出	TMR5溢出	保留
00101	TMR4OUT（后分频）	TMR4OUT（后分频）	TMR4OUT（后分频）
00100	TMR3溢出	保留	TMR3溢出
00011	TMR2OUT（后分频）	TMR2OUT（后分频）	TMR2OUT（后分频）
00010	保留	TMR1溢出	TMR1溢出
00001	TMR0溢出	TMR0溢出	TMR0溢出
00000	通过T1GPPS选择的引脚	通过T3GPPS选择的引脚	通过T5GPPS选择的引脚

寄存器21-5： TMRxL：TIMERx低字节寄存器

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
TMRxL<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **TMRxL<7:0>**：Timerx低字节位

寄存器21-6： TMRxH：TIMERx高字节寄存器

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
TMRxH<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **TMRxH<7:0>**：Timerx高字节位

表21-3： 与TIMER1/3/5作为定时器/计数器相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TxCON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	298
TxGCON	GE	GPOL	GTM	GSPM	GO/$\overline{\text{DONE}}$	GVAL	—	—	299
TxCLK	—	—	—	CS<4:0>					300
TxGATE	—	—	—	GSS<4:0>					301
TMRxL	16位TMR3寄存器最低有效字节								302
TMRxH	16位TMR3寄存器最高有效字节的保持寄存器								302

图注： — = 未实现位，读为0。TIMER1/3/5不使用阴影单元。

22.0 TIMER2/4/6模块

Timer2/4/6模块是8位定时器，可以作为自由运行周期计数器，或者在单触发和单稳态工作模式下与控制启动、运行、暂停和复位操作的外部信号配合工作。通过将这些定时器与诸如比较器和CCP模块之类的其他内部外设配合工作，可以实现诸如脉冲密度调制之类的复杂波形控制。定时器的特性包括：

- 8位定时器寄存器
- 8位周期寄存器
- 可选的外部硬件定时器复位
- 可编程的预分频器（分频比为1:1至1:128）
- 可编程的后分频器（分频比为1:1至1:16）
- 可选的同步/异步操作
- 备用时钟源
- 发生周期匹配时中断
- 三种工作模式：
  - 自由运行周期
  - 单触发
  - 单稳态

Timer2框图请参见图22-1。时钟源框图请参见图22-2。

注： 该器件上实现了3个相同的Timer2模块。这些定时器命名为Timer2、Timer4和Timer6。对Timer2的所有引用均适用于Timer4和Timer6。对T2PR的所有引用也同样适用于T4PR和T6PR。

图22-1： TIMER2框图

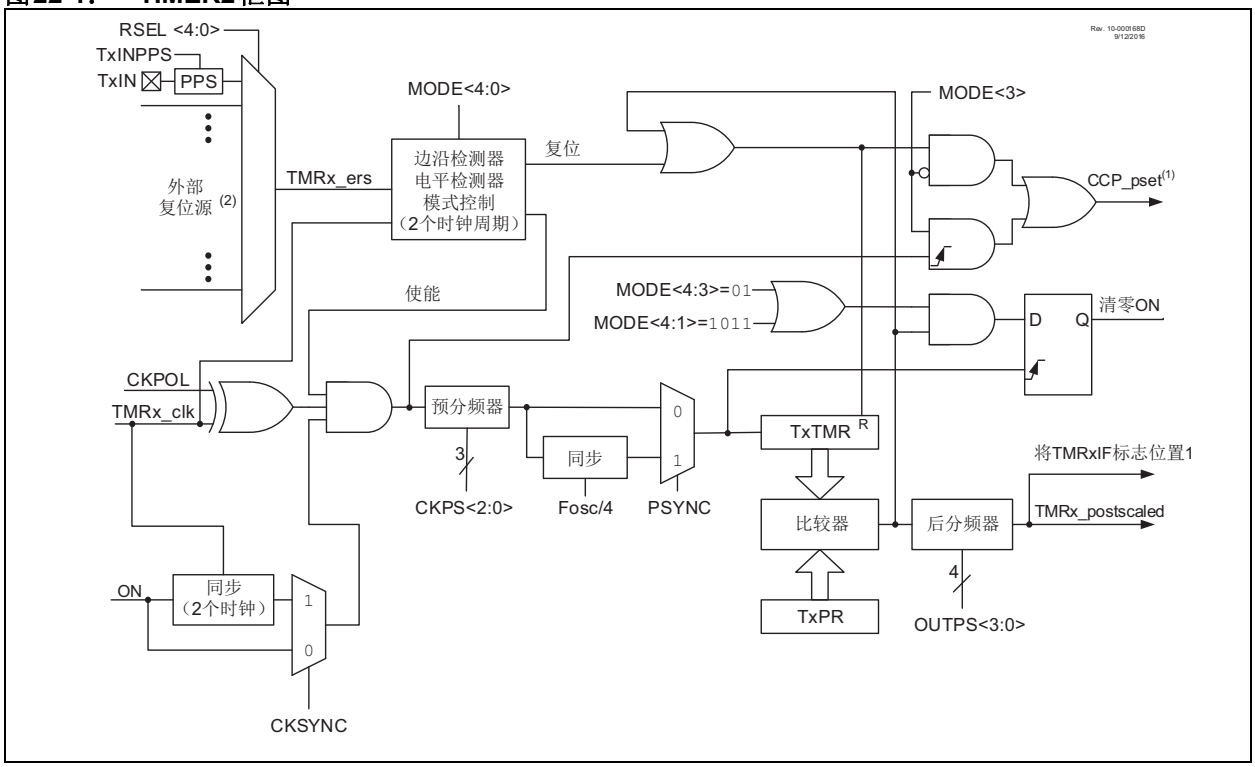
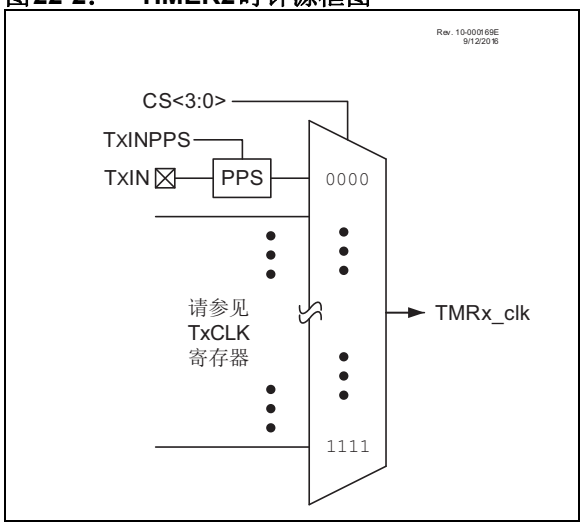


图22-2： TIMER2时钟源框图



22.1 Timer2工作模式

Timer2具有以下三种主要工作模式：

- 自由运行周期
- 单触发
- 单稳态

每种模式下都有多种启动、停止和复位选项。表22-1列出了这些选项。

在所有模式下，T2TMR计数寄存器都是在来自可编程预分频器的时钟信号的上升沿发生递增。当T2TMR等于T2PR时，会向后分频器计数器输出高电平。T2TMR在下一个时钟输入清零。

此外，还可以配置来自硬件的外部信号，对定时器操作进行门控或强制产生T2TMR计数复位。在门控模式下，计数器在门控被禁止时停止，在门控被使能时恢复。在复位模式下，T2TMR计数在外部源出现任一电平或边沿时复位。

T2TMR和T2PR寄存器均可直接读写。在任何器件复位时，T2TMR寄存器都会清零，而T2PR寄存器则初始化为FFh。发生以下事件时，预分频器和后分频器计数器均会清零：

- 对T2TMR寄存器进行写操作
- 对TxCON寄存器进行写操作
- 任何器件复位
- 复位定时器的外部复位源事件。

注：	写TxCON时T2TMR不会清零。

22.1.1 自由运行周期模式

在每个时钟周期，T2TMR的值都会与周期寄存器T2PR中的值进行比较。当两个值匹配时，比较器会在下一个周期将T2TMR的值复位为00h，并递增输出后分频器计数器。当后分频器计数等于TxCON寄存器的OUTPS位的值时，T2TMR_postscaled输出上会出现宽度为一个时钟周期的脉冲，同时后分频器计数清零。

22.1.2 单触发模式

单触发模式与自由运行周期模式基本相同，只是在T2TMR与T2PR匹配时会清零ON位并停止定时器，并且只有在关闭并开启T2ON位之后才会重新启动定时器。在该工作模式下，后分频器OUTPS位的值仅在为0时有意义，因为定时器在发生第一个周期事件时停止，并且后分频器在定时器重启时复位。

22.1.3 单稳态模式

单稳态模式与单触发模式类似，只是ON位不清零，并且定时器可以通过外部复位事件重启。

22.2 Timer2输出

Timer2模式的主输出为T2TMR_postscaled，当后分频器计数器与TxCON寄存器的OUTPS位的值匹配时，该输出将产生宽度为单个T2TMR_clk周期的脉冲。每当T2TMR值与T2PR值匹配时，T2PR后分频器都会递增。可以选择该信号作为其他几个输入模块的输入。

CCP模块在PWM模式下也会使用Timer2来进行脉冲生成。实际T2TMR值及其他内部信号都会被发送到CCP模块，以正确地控制PWM信号的周期和脉宽。关于设置Timer2来用于CCP的更多详细信息，请参见**第23.0节“捕捉/比较/PWM模块”**；关于不同Timer2模式如何影响CCP PWM输出的示例，请参见**第22.5节“操作示例”**中的时序图。

22.3 外部复位源

除了时钟源之外，Timer2还可接受外部复位源。Timer2、Timer4和Timer6的这种外部复位源分别使用T2RST、T4RST和T6RST寄存器进行选择。该复位源可以控制定时器的启动和停止，以及定时器的复位，具体取决于定时器所处的模式。定时器的模式由T2HLT寄存器中的MODE位控制。边沿触发模式要求在外部触发信号之间有6个定时器时钟周期。电平触发模式要求触发电平至少为3个定时器时钟周期长。在调试冻结模式下，外部触发信号会被忽略。

表22-1： TIMER2工作模式

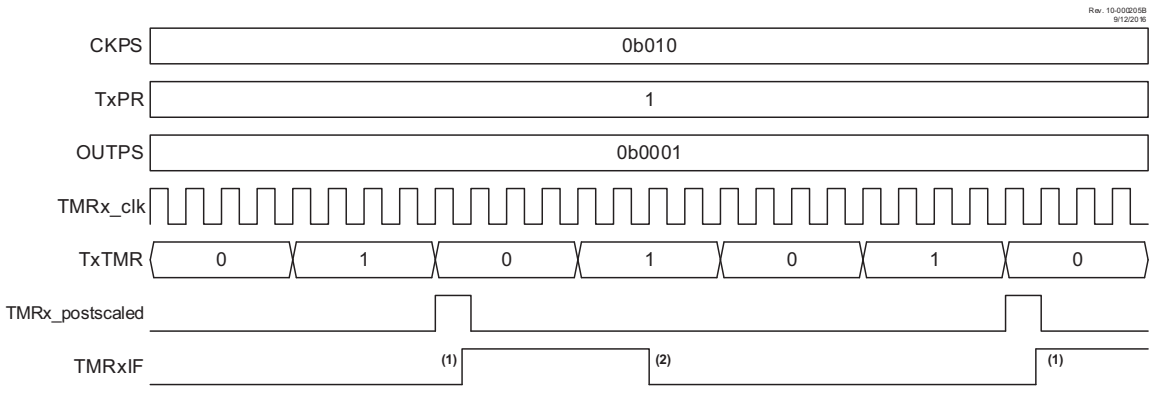
模式	MODE<4:0>		输出操作	操作	定时器控制		
	<4:3>	<2:0>			启动	复位	停止
自由运行周期	00	000	周期脉冲	软件门控（图22-6）	ON = 1	—	ON = 0
		001		硬件门控，高电平有效（图22-7）	ON = 1且 TMRx_ers = 1	—	ON = 0或 TMRx_ers = 0
		010		硬件门控，低电平有效	ON = 1且 TMRx_ers = 0	—	ON = 0或 TMRx_ers = 1
		011	周期脉冲和硬件复位	上升沿或下降沿复位	ON = 1	TMRx_ers ↕	ON = 0
		100		上升沿复位（图22-8）		TMRx_ers ↑	
		101		下降沿复位		TMRx_ers ↓	
		110		低电平复位		TMRx_ers = 0	ON = 0或 TMRx_ers = 0
		111		高电平复位（图22-9）		TMRx_ers = 1	ON = 0或 TMRx_ers = 1
单触发	01	000	单触发	软件启动（图22-10）	ON = 1	—	ON = 0 或 TMRx = PRx 后的下一个时钟 **（注2）**
		001	边沿触发启动（注1）	上升沿启动（图22-9）	ON = 1且 TMRx_ers ↑	—	
		010		下降沿启动	ON = 1且 TMRx_ers ↓	—	
		011		任意边沿启动	ON = 1且 TMRx_ers ↕	—	
		100	边沿触发启动和硬件复位（注1）	上升沿启动和上升沿复位（图22-12）	ON = 1且 TMRx_ers ↑	TMRx_ers ↑	
		101		下降沿启动和下降沿复位	ON = 1且 TMRx_ers ↓	TMRx_ers ↓	
		110		上升沿启动和低电平复位（图22-13）	ON = 1且 TMRx_ers ↑	TMRx_ers = 0	
		111		下降沿启动和高电平复位	ON = 1且 TMRx_ers ↓	TMRx_ers = 1	
单稳态	10	000	保留				
		001	边沿触发启动（注1）	上升沿启动（图22-12）	ON = 1且 TMRx_ers ↑	—	ON = 0 或 TxTMR = TxPR 后的下一个时钟 **（注3）**
		010		下降沿启动	ON = 1且 TMRx_ers ↓	—	
		011		任意边沿启动	ON = 1且 TMRx_ers ↕	—	
保留		100	保留				
保留		101	保留				
单触发		110	电平触发启动和硬件复位	高电平启动和低电平复位（图22-13）	ON = 1且 TMRx_ers = 1	TMRx_ers = 0	ON = 0或 保持在复位状态 **（注2）**
		111		低电平启动和高电平复位	ON = 1且 TMRx_ers = 0	TMRx_ers = 1	
保留	11	xxx	保留				

注 1： 如果ON = 0，则在ON = 1后需要一个边沿来重启定时器。
2： 当TxTMR = TxPR时，下一个时钟会清零ON并使TxTMR停止在00h处。
3： 当TxTMR = TxPR时，下一个时钟会使TxTMR停止在00h处，但不会清零ON。

22.4 Timer2中断

Timer2也可以产生器件中断。当后分频器计数器与16个后分频选项（1:1至1:16）之一匹配时，会产生中断；后分频选项使用T2CON寄存器的后分频控制位OUTPS进行选择。可以通过将相应PIE寄存器的T2TMR中断允许位TMR2IE置1来允许该中断。中断时序如图22-3所示。

图22-3： TIMER2预分频器、后分频器和中断时序图



注 1： 中断标志置1与指令时钟同步。
同步可能需要多达2个指令周期的时间

2： 用软件清零。

22.5 操作示例

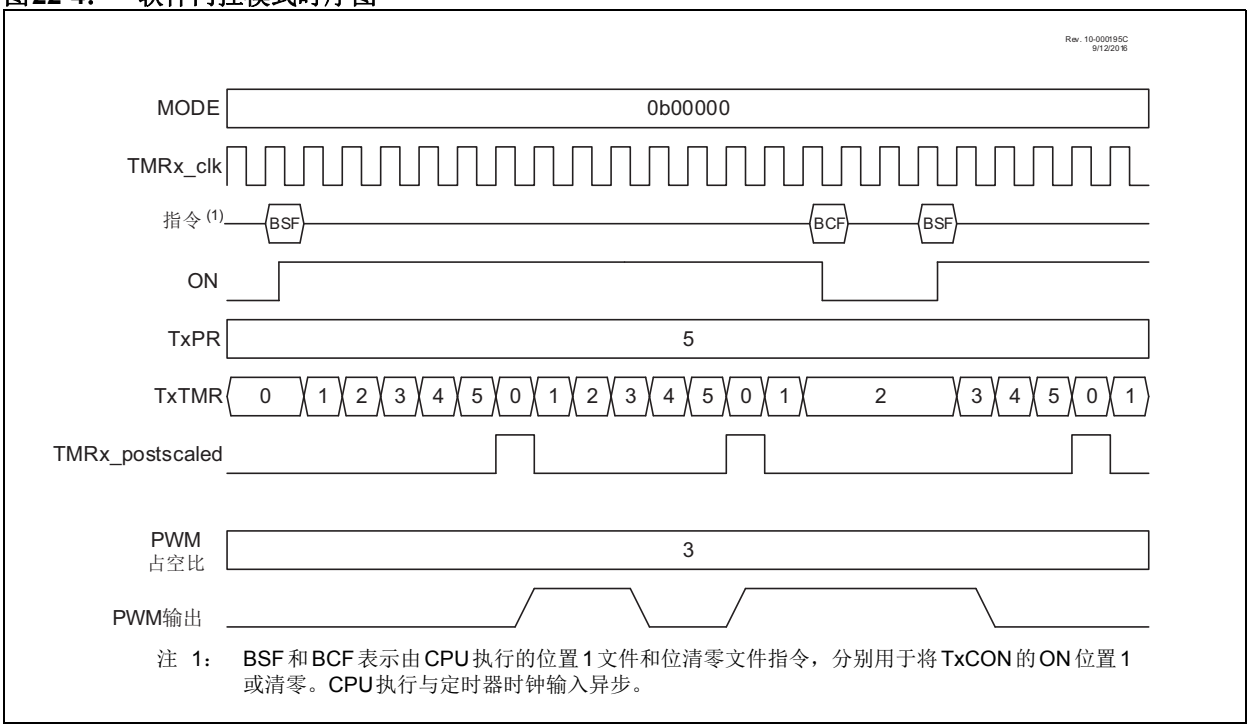
除非另外说明，否则下述内容适用于下列时序图：

- 预分频器和后分频器均设置为1:1（T2CON寄存器的CKPS和OUTPS位均清零）。
- 这些时序图说明的是使用除Fosc/4之外任何其他时钟时的操作，对于ON和T2TMR_ers显示了至少两个完整周期的时钟同步延时。使用Fosc/4时，T2TMR_ers的时钟同步延时至少为1个指令周期；ON在下一个指令周期应用。
- 仅对ON和T2TMR_ers进行了概括说明，时钟同步延时产生的结果可能与说明中略有不同。
- 图中显示了PWM占空比和PWM输出，即假定定时器用于CCP模块的PWM功能，如**第23.0节“捕捉/比较/PWM模块”**和**第24.0节“脉宽调制（PWM）”**所述。这些信号不属于T2TMR模块的一部分。

22.5.1 软件门控模式

当ON = 1时，定时器会随每个时钟输入发生递增；当ON = 0时，它不会发生递增。当T2TMR计数等于T2PR周期计数时，定时器会在下一个时钟发生复位，并继续从0开始计数。图22-4显示了ON位由软件控制时的操作。T2PR = 5时，计数器会一直递增至T2TMR = 5，并在下一个时钟变为0。

图22-4： 软件门控模式时序图



注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

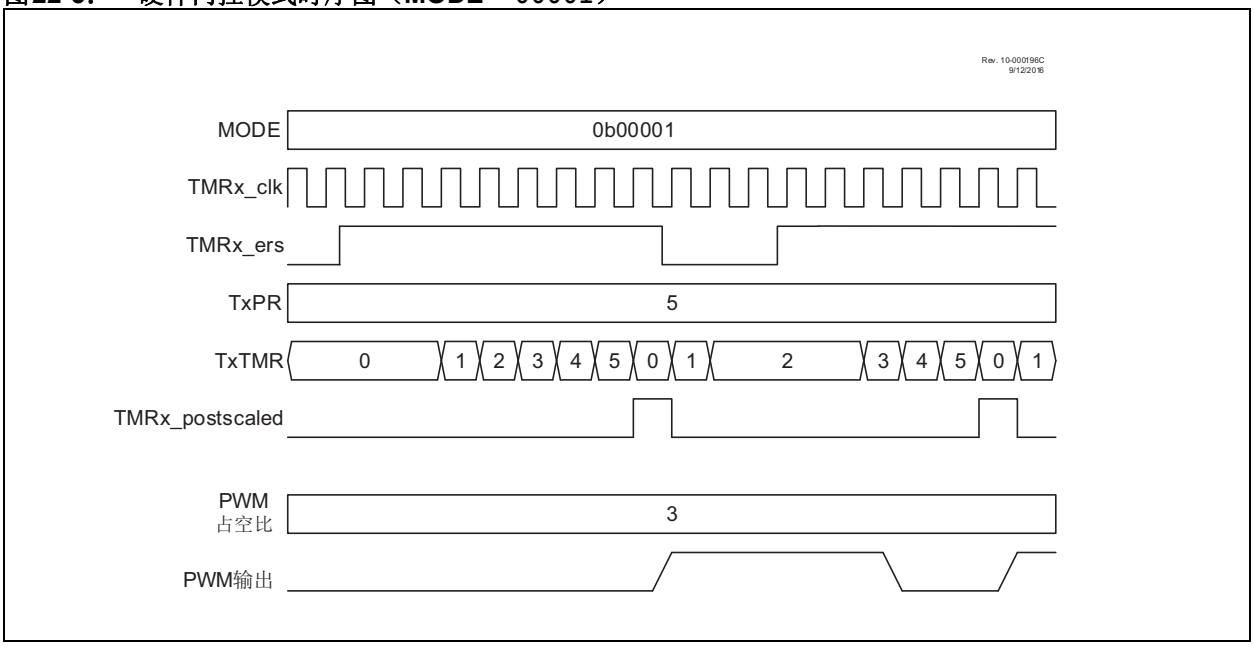
22.5.2 硬件门控模式

硬件门控模式的工作方式与软件门控模式基本相同，只是T2TMR_ers外部信号也可以对定时器进行门控。与CCP配合使用时，门控会延长PWM周期。如果定时器在PWM输出为高电平时停止，则也会增大占空比。

当MODE<4:0> = 00001时，定时器在外部信号为高电平时停止。当MODE<4:0> = 00010时，定时器在外部信号为低电平时停止。

图22-5显示了MODE<4:0> = 00001时的硬件门控模式，在该模式下输入高电平会启动计数器。

图22-5： 硬件门控模式时序图（MODE = 00001）



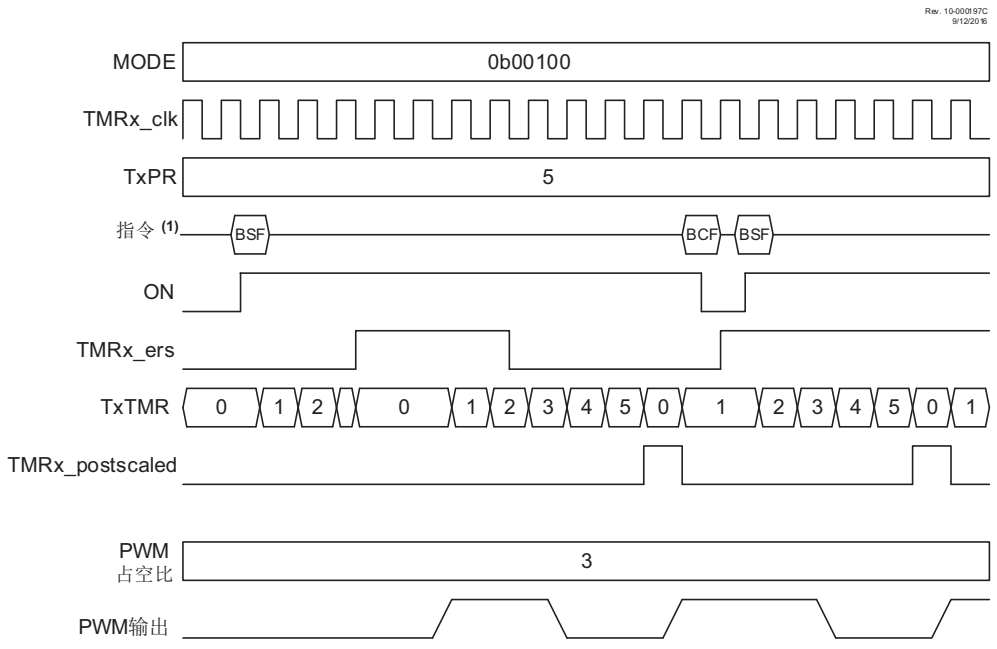
22.5.3 边沿触发硬件限制模式

在硬件限制模式下，可以在定时器达到周期计数之前通过TMRx_ers外部信号复位定时器。支持3种类型的复位：

- 在上升沿或下降沿复位（MODE<4:0> = 00011）
- 在上升沿复位（MODE<4:0> = 0010）
- 在下降沿复位（MODE<4:0> = 00101）

当定时器与PWM模式下的CCP配合使用时，提前复位会缩短周期，并在两个时钟的延时之后重新启动PWM脉冲。请参见图22-6。

图22-6： 边沿触发硬件限制模式时序图（MODE = 00100）



注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

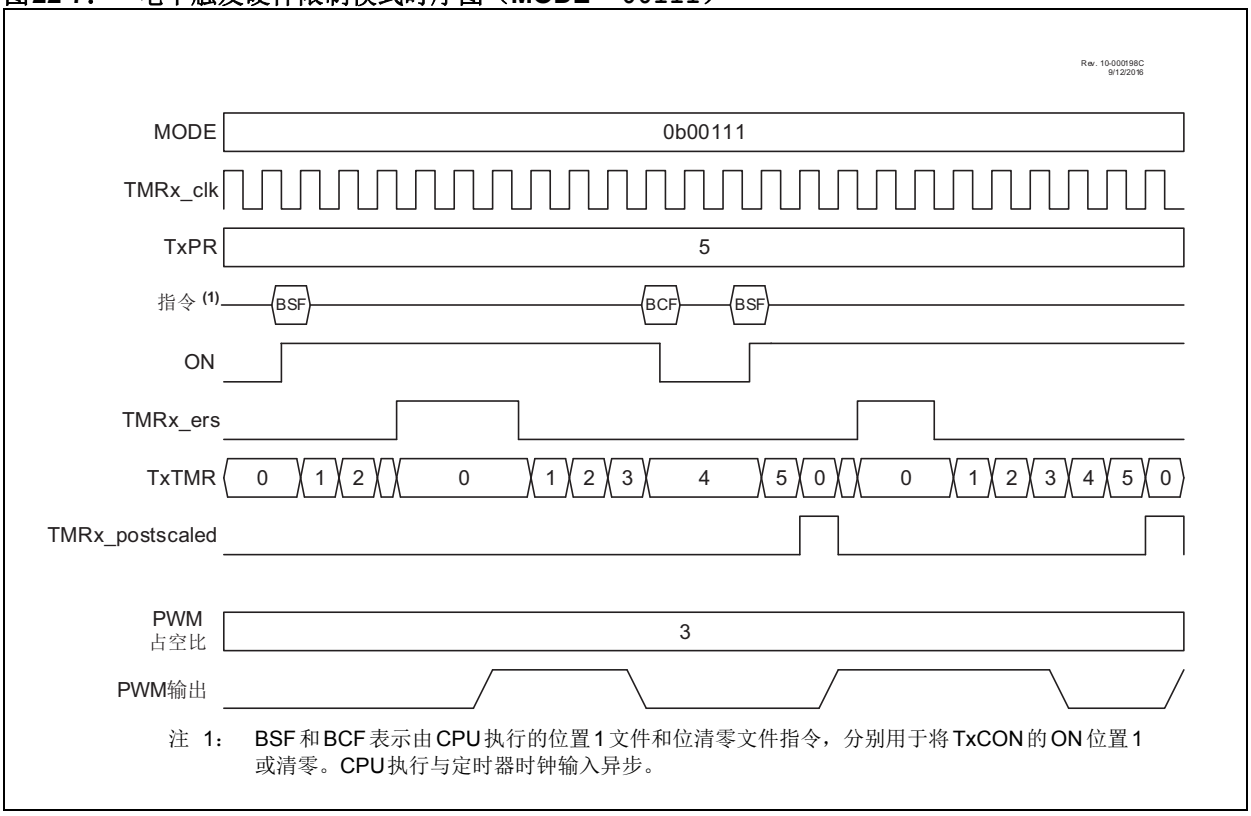
22.5.4 电平触发硬件限制模式

在电平触发硬件限制定时器模式下，计数器通过外部信号TMR2_ers的高电平或低电平进行复位，如图22-7所示。选择MODE<4:0> = 00110会导致定时器在出现低电平外部信号时复位。选择MODE<4:0> = 00111会导致定时器在出现高电平外部信号时复位。在示例中，计数器在TMR2_ers = 1时复位。ON由BSF和BCF指令控制。当ON=0时，外部信号会被忽略。

当CCP使用定时器作为PWM时基时，PWM输出将在定时器开始计数时设置为高电平，之后仅在定时器计数与CCPRx值匹配时设置为低电平。在定时器计数与T2PR值匹配时或在外部复位信号变为真并保持为真的两个时钟周期之后，定时器发生复位。

在发生T2PR匹配的下一个时钟或在外部复位信号释放复位的两个时钟后，定时器开始计数，PWM输出设置为高电平。PWM输出将一直保持高电平，直到定时器递增至与CCPRx脉宽值匹配为止。如果外部复位信号在PWM输出为高电平时变为真，则PWM输出将一直保持高电平，直到复位信号被释放，使定时器可以递增至与CCPRx值匹配为止。

图22-7： 电平触发硬件限制模式时序图（MODE = 00111）



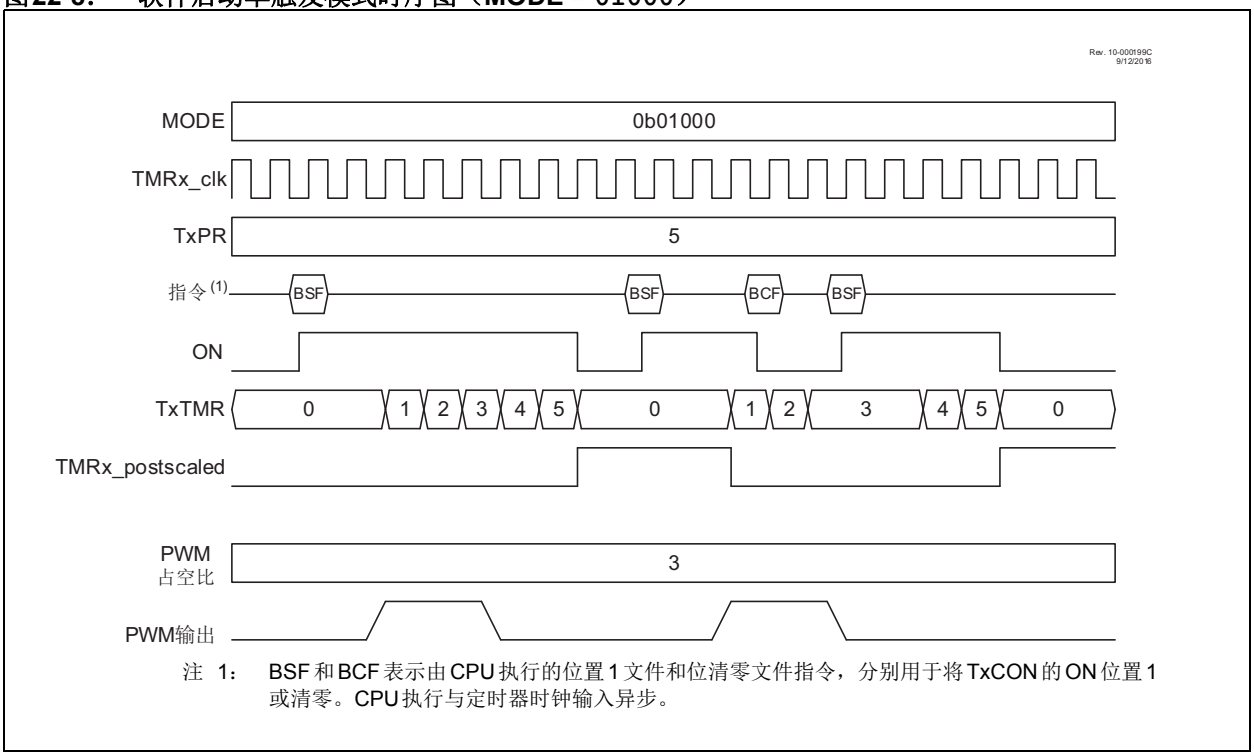
注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

22.5.5 软件启动单触发模式

在单触发模式下，当定时器值与T2PR周期值匹配时，定时器发生复位，ON位清零。要启动另一个定时器周期，必须用软件将ON位置1。设置MODE<4:0> = 01000将选择单触发模式，如图22-8所示。在示例中，ON由BSF和BCF指令控制。在第一种情况下，BSF指令将ON置1，计数器运行至完成，并清零ON。在第二种情况下，BSF指令启动计数周期，BCF/BSF指令在该周期期间关闭和开启计数器，然后它运行至完成。

当单触发模式与CCP PWM操作配合使用时，PWM脉冲驱动会随ON位置1而同时启动。在PWM驱动处于有效状态时清零ON位会延长PWM驱动。PWM驱动将在定时器值与CCPRx脉宽值匹配时终止。PWM驱动将保持关闭，直到软件通过将ON位置1而启动另一个周期为止。如果软件在CCPRx匹配之后但在T2PR匹配之前清零ON位，则PWM驱动会被延长，延长量等于ON位保持清零状态的时间长度。只有在ON位由T2PR周期计数匹配事件清零之后，才能通过将ON位置1来启动另一个计时周期。

图22-8： 软件启动单触发模式时序图（MODE = 01000）



注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

22.5.6 边沿触发单触发模式

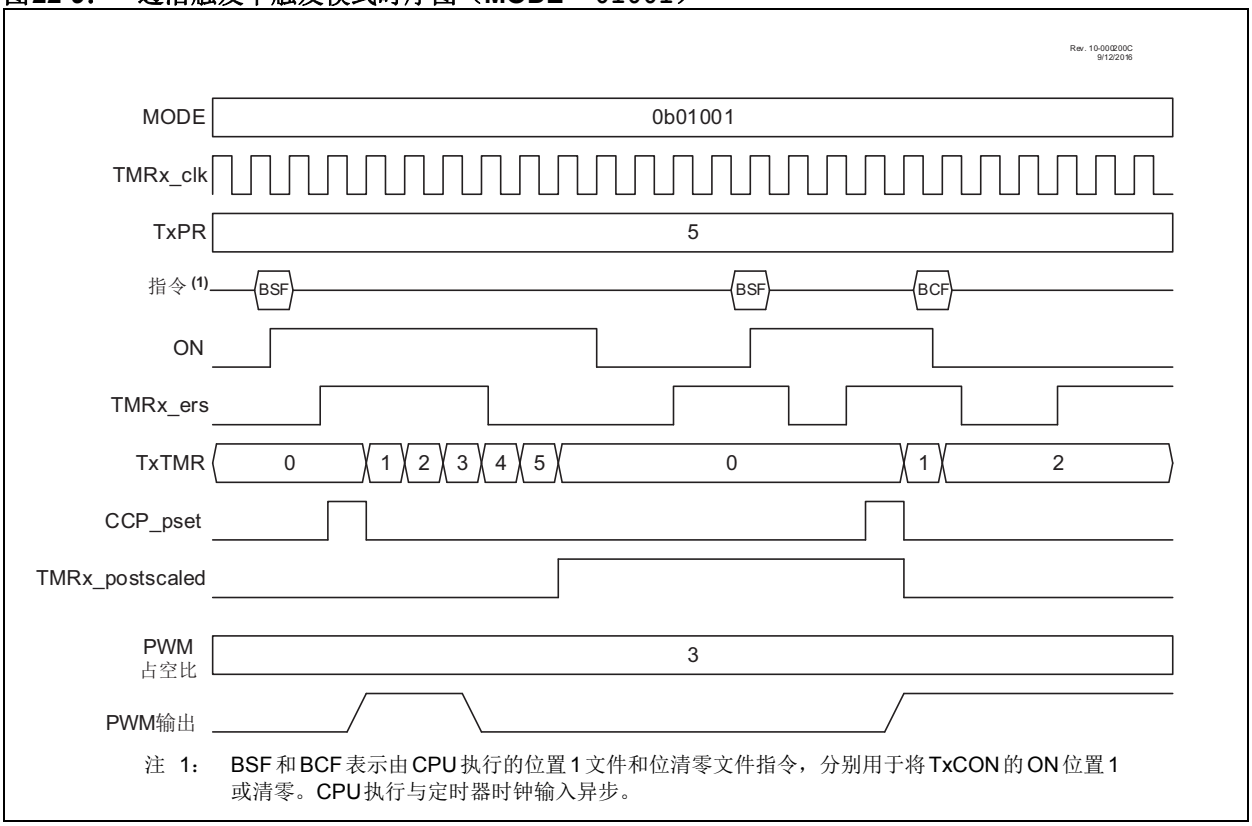
在ON位置1之后，边沿触发单触发模式会在外部信号输入出现边沿时启动定时器，在定时器与T2PR周期值匹配时清零ON位。以下边沿将启动定时器：

- 上升沿（MODE<4:0> = 01001）
- 下降沿（MODE<4:0> = 01010）
- 上升沿或下降沿（MODE<4:0> = 01011）

如果通过将ON位清零暂停了定时器，则在将ON位置1后需要另一个TMRx_ers边沿来恢复计数。图22-9显示了上升沿单触发模式下的操作。

当边沿触发单触发模式与CCP配合使用时，边沿触发信号将激活PWM驱动，并在定时器与CCPRx脉宽值匹配时，禁止PWM驱动。当定时器因发生T2PR周期计数匹配而暂定时，PWM驱动保持禁止状态。

图22-9： 边沿触发单触发模式时序图（MODE = 01001）



注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

22.5.7 边沿触发硬件限制单触发模式

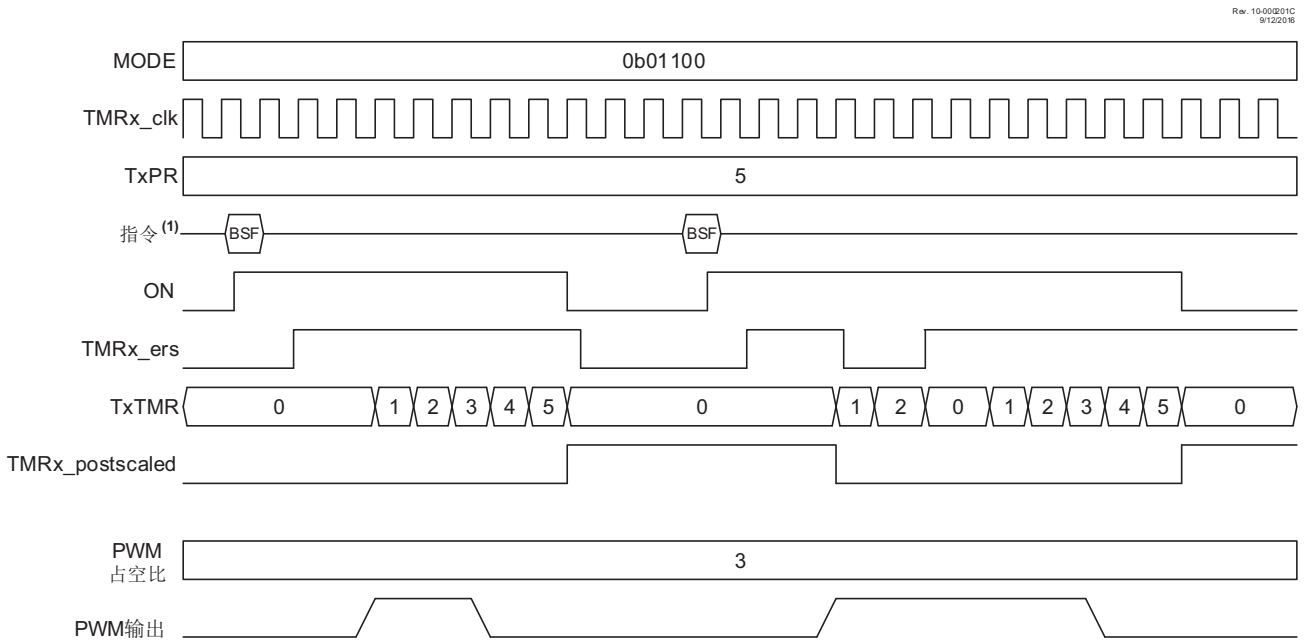
在边沿触发硬件限制单触发模式下，定时器在ON位置1后出现第一个外部信号边沿时启动，并在出现所有后续边沿时复位。在ON位置1后，只需要通过第一个边沿来启动定时器。计数器将在所有后续外部复位边沿的两个时钟后自动恢复计数。边沿触发信号如下：

- 上升沿启动和复位（MODE<4:0> = 01100）
- 下降沿启动和复位（MODE<4:0> = 01101）

当定时器值与T2PR周期值匹配时，定时器会发生复位，并清零ON位。只有在软件将ON位置1之后，外部信号边沿才会产生作用。图22-10显示了上升沿硬件限制单触发操作。

当该模式与CCP配合使用时，第一个启动边沿会触发PWM驱动，所有后续复位边沿会激活PWM驱动。除非外部信号边沿在发生匹配之前使定时器复位，否则在定时器与CCPRx脉宽值匹配时，PWM驱动会变为无效，并保持无效直到定时器由于T2PR周期匹配而暂停。

图22-10： 边沿触发硬件限制单触发模式时序图（MODE = 01100）



注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

22.5.8 电平复位、边沿触发硬件限制单触发模式

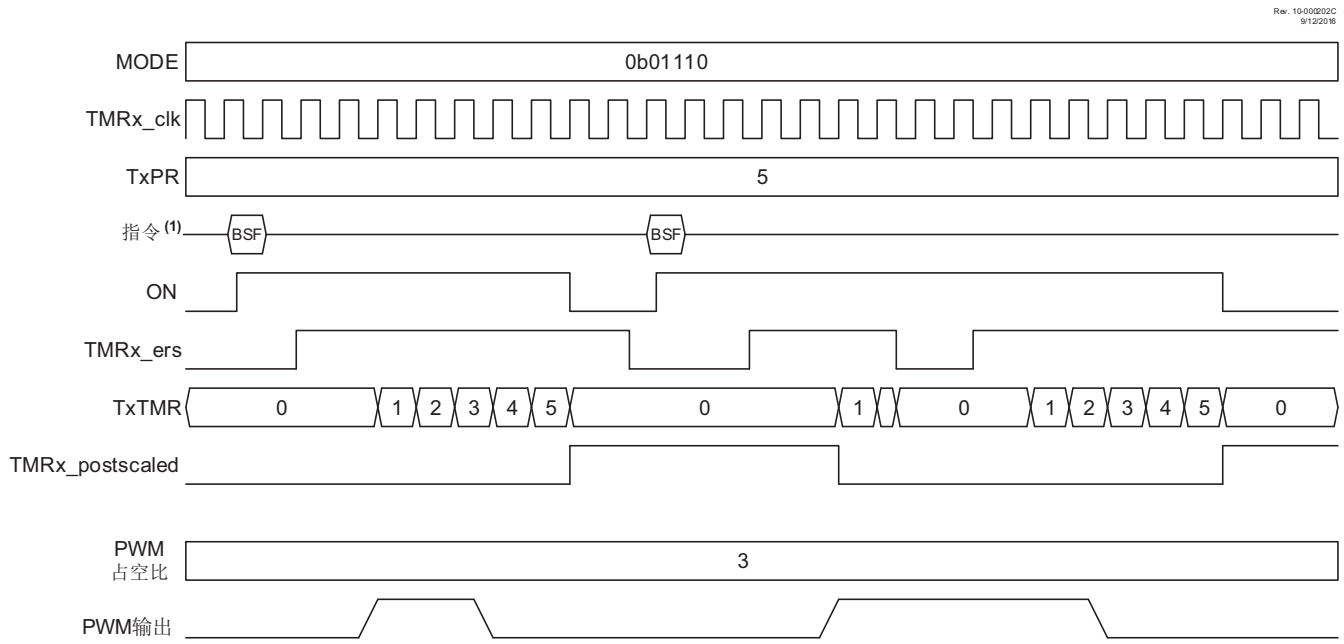
在电平复位边沿触发单触发模式下，当ON位置1时，定时器计数在出现外部信号电平时复位，在从复位电平变为有效电平的上升沿/下降沿开始计数。可选择的复位电平如下：

- 低电平复位（MODE<4:0> = 01110）
- 高电平复位（MODE<4:0> = 01111）

当定时器计数与T2PR周期计数匹配时，定时器会发生复位，ON位会被清零。当ON位由于发生T2PR匹配或在软件控制下清零时，在ON位置1之后需要一个新的外部信号边沿来启动计数器。

当电平复位边沿触发单触发模式与CCP PWM操作配合使用时，PWM驱动会随启动定时器的外部信号边沿变为有效。PWM驱动会在定时器计数等于CCPRx脉宽计数时变为无效。当定时器计数由于T2PR周期计数匹配而清零时，PWM驱动不会变为有效。

图22-11： 低电平复位、边沿触发硬件限制单触发模式时序图（MODE = 01110）



注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

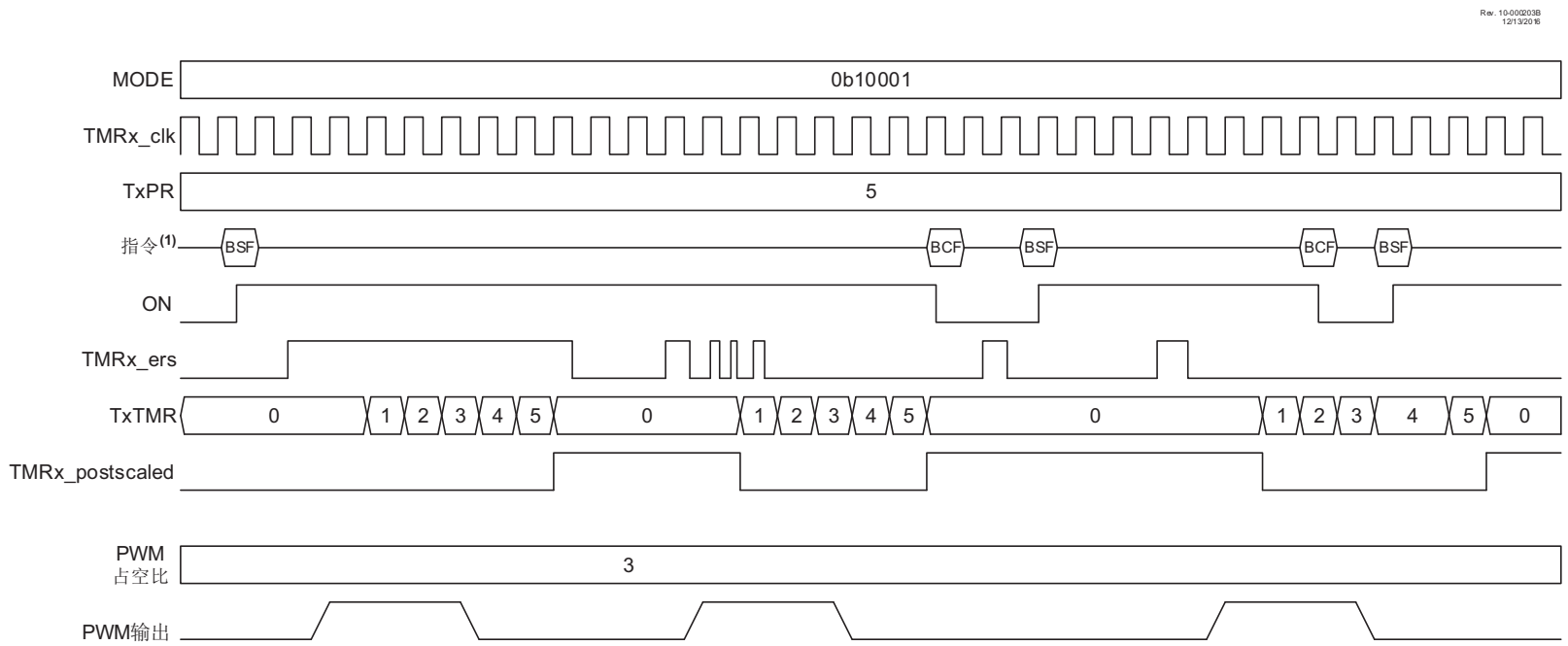
22.5.9 边沿触发单稳态模式

在ON位置1之后，边沿触发单稳态模式会在外部复位信号输入出现边沿时启动定时器，在定时器与T2PR周期值匹配时停止递增定时器。以下边沿将启动定时器：

- 上升沿（MODE<4:0> = 10001）
- 下降沿（MODE<4:0> = 10010）
- 上升沿或下降沿（MODE<4:0> = 10011）

当边沿触发单稳态模式与CCP PWM操作配合使用时，PWM驱动会随启动定时器的外部复位信号边沿而变为有效，但不会在定时器与T2PR值匹配时变为有效。当定时器在递增时，外部复位信号上的额外边沿不会影响CCP PWM。

图22-12： 上升沿触发单稳态模式时序图（MODE = 10001）



注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

22.5.10 电平触发硬件限制单触发模式

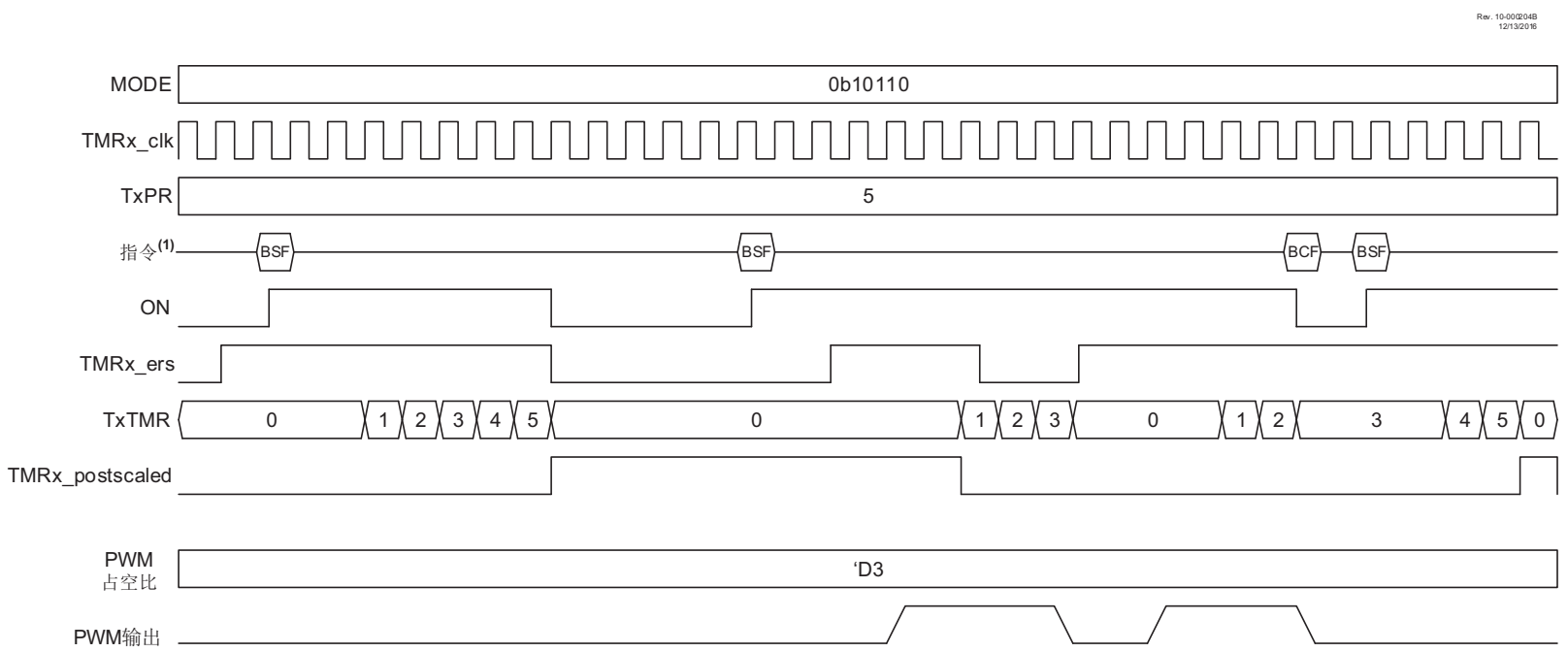
电平触发硬件限制单触发模式会在出现外部复位电平时将定时器保持在复位状态，并在ON位置1且外部信号不为复位电平时启动计数。如果存在外部信号不为复位电平或ON位置1这两者中的一种条件，则在另一信号置1/变为有效时会启动定时器。可选择的复位电平如下：

- 低电平复位（MODE<4:0> = 10110）
- 高电平复位（MODE<4:0> = 10111）

当定时器计数与T2PR周期计数匹配时，定时器会发生复位，ON位会被清零。当ON位由于发生T2PR匹配或在软件控制下清零时，定时器将保持复位状态，直到ON位置1且外部信号不为复位电平为止。

当电平触发硬件限制单触发模式与CCP PWM操作配合使用时，PWM驱动会随外部信号边沿或ON位置1（两者均会启动定时器）变为有效。

图22-13： 电平触发硬件限制单触发模式时序图（MODE = 10110）



注 1： BSF和BCF表示由CPU执行的位置1文件和位清零文件指令，分别用于将TxCON的ON位置1或清零。CPU执行与定时器时钟输入异步。

22.6 休眠期间的Timer2操作

当PSYNC = 1时，在处理器处于休眠模式时，Timer2无法工作。在处理器处于休眠模式时，T2TMR和T2PR寄存器的内容将保持不变。

当PSYNC = 0时，只要选择的时钟源也仍在运行，Timer2就会在休眠模式下工作。选择LFINTOSC、MFINTOSC或HFINTOSC振荡器作为定时器时钟源会使选定的振荡器在休眠期间保持运行。

22.7 寄存器定义：Timer2/4/6控制

表22-2给出了Timer2/4/6外设的长位名称前缀。更多信息，请参见**第1.3.2.2节“长位名称”**。

表22-2： 工作模式

外设	位名称前缀
Timer2	T2
Timer4	T4
Timer6	T6

寄存器22-1： TxCLK：TIMERx时钟选择寄存器

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	CS<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4 **未实现：**读为0

bit 3-0 **CS<3:0>：**Timerx时钟选择位

CS<3:0>	T2TMR	TMR4	TMR6
	时钟源	时钟源	时钟源
1111	保留	保留	保留
1110	CLC4_out	CLC4_out	CLC4_out
1101	CLC3_out	CLC3_out	CLC3_out
1100	CLC2_out	CLC2_out	CLC2_out
1011	CLC1_out	CLC1_out	CLC1_out
1010	ZCD_OUT	ZCD_OUT	ZCD_OUT
1001	NCO1OUT	NCO1OUT	NCO1OUT
1000	CLKREF_OUT	CLKREF_OUT	CLKREF_OUT
0111	SOSC	SOSC	SOSC
0110	MFINTOSC（32 kHz）	MFINTOSC（32 kHz）	MFINTOSC（32 kHz）
0101	MFINTOSC（500 kHz）	MFINTOSC（500 kHz）	MFINTOSC（500 kHz）
0100	LFINTOSC	LFINTOSC	LFINTOSC
0011	HFINTOSC	HFINTOSC	HFINTOSC
0010	Fosc	Fosc	Fosc
0001	Fosc/4	Fosc/4	Fosc/4
0000	通过T2INPPS选择的引脚	通过T4INPPS选择的引脚	通过T6INPPS选择的引脚

寄存器22-2： TxRST：TIMER2外部复位信号选择寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	RSEL<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现：**读为0

bit 4-0 **RSEL<4:0>：**Timer2外部复位信号源选择位

RSEL<4:0>	T2TMR	TMR4	TMR6
	复位源	复位源	复位源
11111-11001	保留	保留	保留
11000	UART2_tx_edge	UART2_tx_edge	UART2_tx_edge
10111	UART2_rx_edge	UART2_rx_edge	UART2_rx_edge
10110	UART1_tx_edge	UART1_tx_edge	UART1_tx_edge
10101	UART1_rx_edge	UART1_rx_edge	UART1_rx_edge
10100	CLC4_out	CLC4_out	CLC4_out
10011	CLC3_out	CLC3_out	CLC3_out
10010	CLC2_out	CLC2_out	CLC2_out
10001	CLC1_out	CLC1_out	CLC1_out
10000	ZCD_OUT	ZCD_OUT	ZCD_OUT
01111	CMP2OUT	CMP2OUT	CMP2OUT
01110	CMP1OUT	CMP1OUT	CMP1OUT
01101-01100	保留	保留	保留
01011	PWM8OUT	PWM8OUT	PWM8OUT
01010	PWM7OUT	PWM7OUT	PWM7OUT
01001	PWM6OUT	PWM6OUT	PWM6OUT
01000	PWM5OUT	PWM5OUT	PWM5OUT
00111	CCP4OUT	CCP4OUT	CCP4OUT
00110	CCP3OUT	CCP3OUT	CCP3OUT
00101	CCP2OUT	CCP2OUT	CCP2OUT
00100	CCP1OUT	CCP1OUT	CCP1OUT
00011	TMR6后分频	TMR6后分频	保留
00010	TMR4后分频	保留	TMR4后分频
00001	保留	T2TMR后分频	T2TMR后分频
00000	通过T2INPPS选择的引脚	通过T4INPPS选择的引脚	通过T6INPPS选择的引脚

寄存器22-3： TxTMR：TIMERx计数器寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TMRx<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **TMRx<7:0>：**定时器x计数器位

寄存器22-4： TxPR：TIMERx周期寄存器

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
PRx<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **PRx<7:0>：**Timerx周期寄存器位

寄存器22-5： TxCON：TIMERx控制寄存器

R/W/HC-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ON	CKPS<2:0>			OUTPS<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **ON**：Timerx使能位(1)
1 = 开启Timerx
0 = 关闭Timerx：复位所有计数器和状态机

bit 6-4 **CKPS<2:0>**：Timerx型时钟预分频比选择位
111 = 1:128预分频比
110 = 1:64预分频比
101 = 1:32预分频比
100 = 1:16预分频比
011 = 1:8预分频比
010 = 1:4预分频比
001 = 1:2预分频比
000 = 1:1预分频比

bit 3-0 **OUTPS<3:0>**：Timerx输出后分频比选择位
1111 = 1:16后分频比
1110 = 1:15后分频比
1101 = 1:14后分频比
1100 = 1:13后分频比
1011 = 1:12后分频比
1010 = 1:11后分频比
1001 = 1:10后分频比
1000 = 1:9后分频比
0111 = 1:8后分频比
0110 = 1:7后分频比
0101 = 1:6后分频比
0100 = 1:5后分频比
0011 = 1:4后分频比
0010 = 1:3后分频比
0001 = 1:2后分频比
0000 = 1:1后分频比

注 1： 在某些模式下，ON位将由硬件自动清零。请参见**第22.1.2节“单触发模式”**。

寄存器22-6： TxHLT：TIMERx硬件限制控制寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
PSYNC	CKPOL	CKSYNC	MODE<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **PSYNC**：Timerx预分频器同步使能位(1, 2)
1 = TxTMR预分频器输出与Fosc/4进行同步
0 = TxTMR预分频器输出不与Fosc/4进行同步

bit 6 **CKPOL**：Timerx时钟极性选择位(3)
1 = 定时器/预分频器基于输入时钟的下降沿进行计数
0 = 定时器/预分频器基于输入时钟的上升沿进行计数

bit 5 **CKSYNC**：Timerx时钟同步使能位(4, 5)
1 = ON寄存器位与T2TMR_clk输入进行同步
0 = ON寄存器位不与T2TMR_clk输入进行同步

bit 4-0 **MODE<4:0>**：Timerx控制模式选择位(6, 7)
有关所有工作模式，请参见表22-1。

注 1： 将该位置1可确保读取TxTMR将返回一个有效的数据值。
2： 当该位为1时，Timer2无法在休眠模式下工作。
3： 不应在ON = 1时更改CKPOL。
4： 置1该位可确保在使能或禁止ON位时产生无毛刺的操作。
5： 当该位置1时，定时器操作将在ON位置1后延迟两个TxTMR输入时钟。
6： 除非另有说明，否则所有模式均在ON = 1时启动，在ON = 0时停止（发生停止不会影响TxTMR的值）。
7： 当TxTMR = TxPR时，下一个时钟会清零TxTMR，无论工作模式如何。

表22-3： 与TIMER2相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
TxPR	Timer2模块周期寄存器								305*
TxTMR	8位T2TMR寄存器的保持寄存器								305*
TxCON	ON	CKPS<2:0>			OUTPS<3:0>				323
TxCLK	—	—	—	—	—	CS<2:0>			320
TxRST	—	—	—	—	RSEL<3:0>				321
TxHLT	$\overline{\text{PSYNC}}$	CPOL	$\overline{\text{CSYNC}}$	MODE<4:0>					324

图注： — = 未实现位，读为0。Timer2模块不使用阴影单元。

* 该页提供寄存器信息。

23.0 捕捉/比较/PWM模块

捕捉/比较/PWM模块是允许用户计时和控制不同事件，以及产生脉宽调制（PWM）信号的外设。在捕捉模式下，外设允许对事件的持续时间进行计时。当超过预先确定的时间时，比较模式允许用户触发一个外部事件。PWM模式可以产生不同频率和占空比的脉宽调制信号。

本系列器件包含四个标准捕捉/比较/PWM模块（CCP1、CCP2、CCP3和CCP4）。每个CCP模块可选择用于控制模块的定时器源。每个模块都可以独立选择定时器，通过CCPTMRS0寄存器（寄存器23-2）中的CxTSEL位可以访问该定时器。使用捕捉/比较模式时，默认选择定时器TMR1，而在CCPx模块中使用PWM模式时，默认选择定时器TMR2。

请注意，以下部分将针对TMR1介绍捕捉/比较模式操作，并针对TMR2介绍PWM模式操作。

所有CCP模块的捕捉和比较功能均相同。

注 1： 在具有多个CCP模块的器件中，要特别注意所使用的寄存器名称，这一点很重要。模块缩略名之后的数字用于区分不同的模块。例如，CCP1CON和CCP2CON分别控制两个完全不同CCP模块相同的工作特性。

2： 在本章中，在所有工作模式下，通常指的CCP模块都可以解释为同样适用于CCPx模块。在需要时，寄存器名称、模块信号、I/O引脚和位名称可以使用通用标识符“x”（数字）来识别某个特定模块。

23.1 CCP模块配置

每个捕捉/比较/PWM模块都与一个控制寄存器（CCPxCON）、一个捕捉输入选择寄存器（CCPxCAP）和一个数据寄存器（CCPRx）相关联。而数据寄存器由两个8位寄存器组成：CCPRxL（低字节）和CCPRxH（高字节）。

23.1.1 CCP模块和定时器源

CCP模块使用定时器1至6，具体因所选模式而异。CCP模块可以在捕捉、比较或PWM模式下使用不同的定时器，如表23-1所示。

表23-1： CCP模式——定时器资源

CCP模式	定时器资源
捕捉	Timer1、Timer3或Timer5
比较	
PWM	Timer2、Timer4或Timer6

模块的定时器分配由CCPTMRS0寄存器（见寄存器23-2）中的CCP定时器使能位决定。如果所有模块配置为同时在同一模式（捕捉/比较或PWM）下工作，则这些模块可以立即进入工作状态，并且可以共用同一定时器资源。

23.1.2 漏极开路输出选项

在输出模式（比较模式或PWM模式）下工作时，可选择将CCPx引脚的驱动器配置为漏极开路输出。此功能允许通过外部上拉电阻将引脚上的电压拉高，并允许输出与外部电路通信而无需额外的电平转换器。

23.2 捕捉模式

捕捉模式使用16位定时器Timer1资源。当捕捉源上发生事件时，16位CCPRxH:CCPRxL寄存器对会分别捕捉和存储TMRxH:TMRxL寄存器对的16位值。这些事件定义如下，可通过CCPxCON寄存器的MODE<3:0>位进行配置：

- CCPx输入的每个下降沿
- CCPx输入的每个上升沿
- CCPx输入的每4个上升沿
- CCPx输入的每16个上升沿
- CCPx输入的每个边沿（上升沿或下降沿）

进行捕捉时，相应PIR寄存器的中断请求标志位CCPxIF被置1。该中断标志必须用软件清零。如果在CCPRxH:CCPRxL寄存器对中的值被读取之前又发生另一次捕捉，那么原来的捕捉值会被新捕捉值覆盖。

注：	如果在进行2字节读取期间发生事件，则高字节和低字节数据将来自不同事件。建议在读取CCPRxH:CCPRxL寄存器对时禁止模块或读取寄存器对两次以确保数据完整性。

图23-1给出了捕捉操作的简化框图。

23.2.1 捕捉源

在捕捉模式下，应该通过将相应的TRIS控制位置1把CCPx引脚配置为输入引脚。

注：	如果将CCPx引脚配置为输出引脚，对该端口的写操作可能引发一次捕捉事件。

捕捉源通过配置CCPxCAP寄存器的CTS<2:0>位来选择。有关可以选择的源的列表，请参见CCPxCAP寄存器（寄存器23-4）。

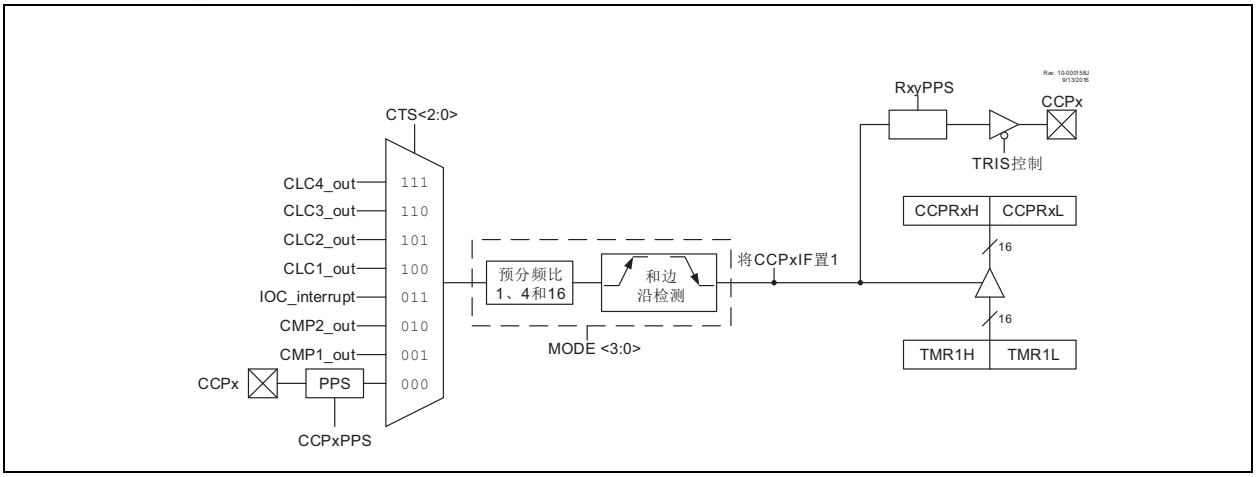
23.2.2 TIMER1模式资源

为使CCP模块使用捕捉特性，Timer1必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，可能无法进行捕捉操作。

- 关于配置Timer1的更多信息，请参见**第21.0节“带门控控制的Timer1/3/5模块”**。

注：	在捕捉模式下，Timer1时钟源不能由系统时钟（FOSC）提供。要使捕捉模式能够识别CCPx引脚上的触发事件，Timer1的时钟必须来自指令时钟（FOSC/4）或外部时钟源。

图23-1： 捕捉模式工作原理框图



23.2.3 软件中断模式

当捕捉模式改变时，可能会产生错误的捕捉中断。用户应保持相应PIE寄存器的CCPxIE中断优先级位清零以避免错误中断。此外，用户应在工作模式的任何改变之后清零相应PIR寄存器的CCPxIF中断标志位。

23.2.4 休眠期间的捕捉操作

捕捉模式依靠Timer1模块才能正确工作。可选用两种方式在捕捉模式下驱动 Timer1 模块：它可由指令时钟（FOSC/4）驱动，或由外部时钟源驱动。

Timer1由FOSC/4提供时钟时，Timer1在休眠期间不进行递增操作。当器件从休眠模式唤醒时，Timer1将从其之前状态继续工作。

只要Timer1的时钟源在休眠模式下有效，捕捉模式就会在休眠期间继续工作。

23.3 比较模式

比较模式使用16位定时器Timer1资源。CCPRxH:CCPRxL寄存器对的16位值会不断与TMRxH:TMRxL寄存器对的16位值进行比较。当发生匹配时，将发生以下事件之一：

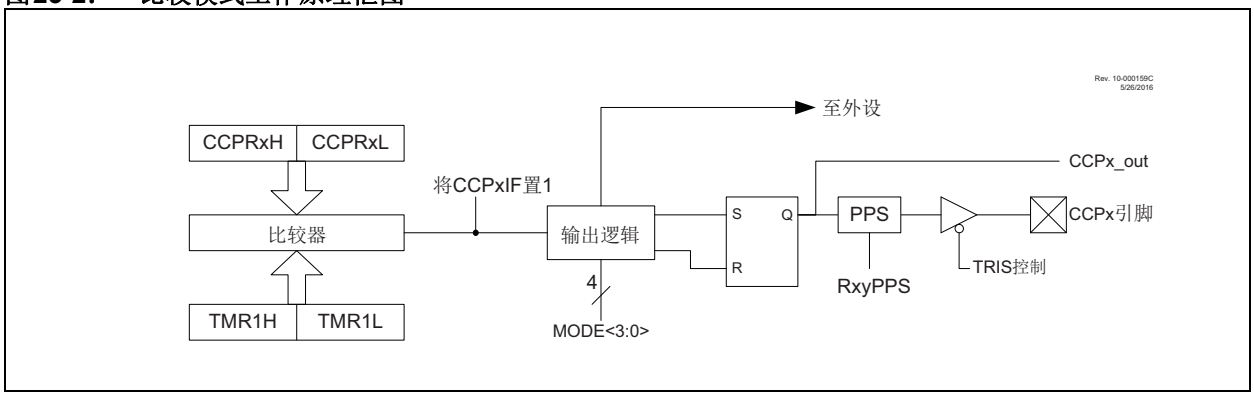
- 翻转CCPx输出，清零TMRx
- 翻转CCPx输出
- 将CCPx输出置1
- 将CCPx输出清零
- 脉冲输出
- 脉冲输出，清零TMRx

引脚的动作由CCPxCON寄存器的MODE<3:0>控制位的值决定。同时，中断标志位CCPxIF置1，并且触发ADC转换（如果选择的话）。

所有比较模式都能产生中断和触发ADC转换。当MODE = 0b0001或0b1011时，CCP复位TMR寄存器对。

图23-2给出了比较操作的简化框图。

图23-2： 比较模式工作原理框图



23.3.1 CCPx引脚配置

软件必须通过将相关的TRIS位清零并通过RxyPPS寄存器定义适当的输出引脚，将CCPx引脚配置为输出。更多详细信息，请参见**第17.0节“外设引脚选择（PPS）模块”**。

注： 清零CCPxCON寄存器会将CCPx比较输出锁存器强制设为默认的低电平。这不是端口I/O数据锁存器。

23.3.2 TIMER1模式资源

在比较模式下，Timer1必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，可能无法进行比较操作。

关于配置Timer1的更多信息，请参见**第21.0节“带门控控制的Timer1/3/5模块”**。

注： 在比较模式下，Timer1时钟源不能由系统时钟（Fosc）提供。欲使比较模式能够识别CCPx引脚上的触发事件，Timer1的时钟必须来自指令时钟（Fosc/4）或外部时钟源。

23.3.3 自动转换触发器

所有CCPx模式将CCP中断标志位（CCPxIF）置1。当此标志置1且发生匹配时，如果选择CCP模块作为转换触发源，将会触发自动转换。

更多信息，请参见**第37.2.5节“自动转换触发器”**。

注： 在产生自动转换触发信号的时钟边沿和产生Timer1复位的时钟边沿之间改变CCPRxH和CCPRxL寄存器对的内容可清除匹配条件，从而阻止复位发生

23.3.4 休眠期间的比较操作

由于FOSC在休眠模式下关闭，比较模式在休眠模式下将不能正常工作，除非定时器正在运行。器件将在发生中断（如果允许）时唤醒。

23.4 PWM概述

脉宽调制（PWM）是一种通过在完全开启和完全关闭状态之间进行快速切换而为负载供电的方案。PWM信号类似于方波，信号的高电平部分视为开启状态，信号的低电平部分视为关闭状态。高电平部分（也称为脉宽）可以随时间而变，并以步为单位进行定义。施加的步数越多（这会增大脉宽），为负载提供的功率就越大。施加的步数降低时（这会缩短脉宽），提供的功率就越小。PWM周期定义为一个完整周期的持续时间，或者开启和关闭时间相加的总时间。

PWM分辨率定义为可以在单个PWM周期中出现的最大步数。分辨率越高，就可以越精确地控制脉宽时间，从而更精确地控制施加在负载上的功率。

占空比这一术语描述开启时间与关闭时间之间以百分比形式表示的比例，0%代表完全关闭，100%代表完全开启。占空比越低，施加的功率就越低；占空比越高，施加的功率就越高。

图23-3给出了PWM信号的典型波形。

23.4.1 标准PWM操作

标准PWM模式可以在CCPx引脚上产生最高可达10位分辨率的脉宽调制（PWM）信号。周期、占空比和分辨率由以下寄存器控制：

- T2PR寄存器
- T2CON寄存器
- CCPRxL和CCPRxH寄存器
- CCPxCON寄存器

为确保PWM正常工作，需要选择Fosc/4作为TMR2/4/6的时钟输入。图23-4给出了PWM操作的简化框图。

注： 要使能CCPx引脚上的PWM输出，必须清零相应的TRIS位。

图23-3： CCP PWM输出信号

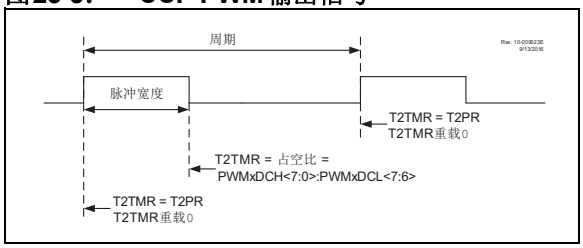
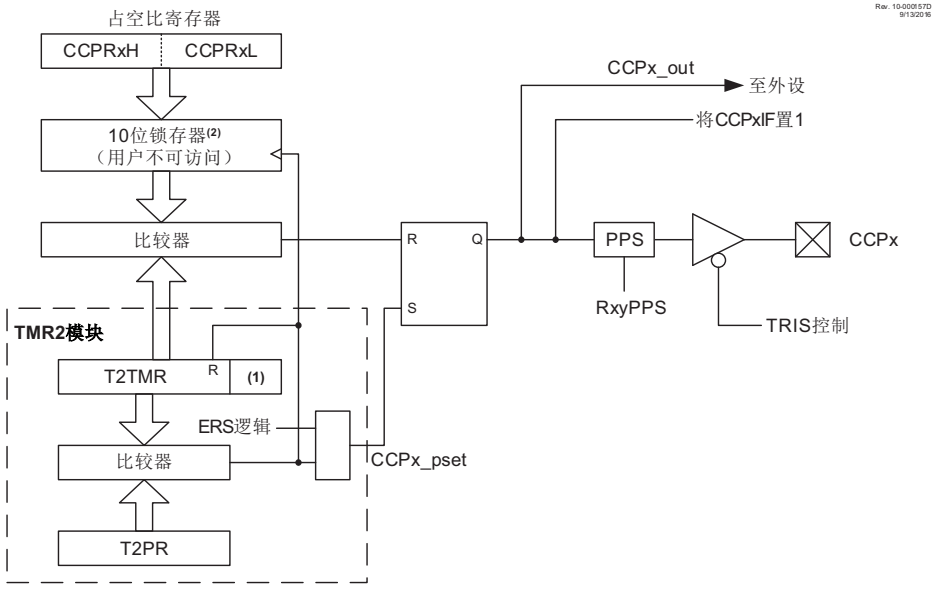


图23-4： 简化的PWM框图



注： 1. 8位定时器与Fosc生成的2位或内部预分频器的2位连接，构成10位时基。
2. CCPR寄存器中10位值的对齐方式由CCPxFMT位决定。

23.4.2 设置PWM操作

当将CCP模块配置为标准PWM操作时，应采用以下步骤：

1. 使用所需输出引脚RxyPPS控制来选择CCPx作为源，并通过将关联的TRIS位置1而禁止CCPx引脚输出驱动器。
2. 将PWM周期值装入T2PR寄存器。
3. 通过将相应值装入CCPxCON寄存器，将CCP模块配置为PWM模式。
4. 将PWM占空比值装入CCPRxL寄存器和CCPRxH寄存器并配置CCPxCON寄存器的FMT位来设置适当的寄存器对齐方式。
5. 配置并启动Timer2：
   - 清零相应PIR寄存器的TMR2IF中断标志位。请参见下面的“注”。
   - 使用T2CLK寄存器选择FOSC/4作为定时器时钟源。必须这样做才能确保PWM模块正常工作。
   - 用定时器预分频值配置T2CON寄存器的CKPS位。
   - 通过将T2CON寄存器的ON位置1使能定时器。
6. 使能PWM输出引脚：
   - 等待定时器溢出并且PIR4寄存器的TMR2IF位置1。请参见下面的“注”。
   - 通过将相关的TRIS位清零，使能CCPx引脚输出驱动器。

注： 为在第一个PWM输出时发送完整的占空比和周期，设置过程必须包含上述步骤。如果在第一个输出时以完整的PWM信号起始并非至关重要，那么可以忽略步骤6。

23.4.3 TIMER2定时器资源

PWM标准模式使用8位Timer2定时器资源指定PWM周期。

23.4.4 PWM周期

PWM周期可通过Timer2的T2PR寄存器来指定。PWM周期可由公式23-1计算。

公式23-1： PWM周期

$$PWM\,周期 = [(T2PR) + 1] \bullet 4 \bullet TOSC \bullet (TMR2\,预分频值)$$

注 1： TOSC = 1/FOSC

当T2TMR中的值与T2PR中的值相等时，在下一个递增周期将发生以下3个事件：

- T2TMR被清零
- CCPx引脚被置1。（例外情况：如果PWM占空比 = 0%，引脚将不会被置1。）
- PWM占空比从CCPRxL/H寄存器对传送到10位缓冲区中。

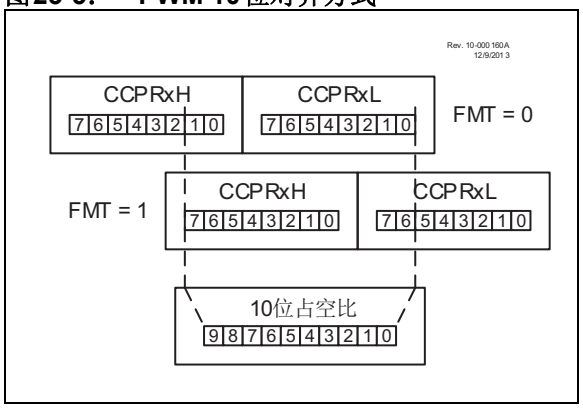
注： 在确定PWM频率时不会用到定时器后分频比（见**第22.3节“外部复位源”**）。

23.4.5 PWM 占空比

通过将10位值写入CCPRxH:CCPRxL寄存器对来指定PWM占空比。10位值的对齐方式由CCPxCON寄存器的FMT位决定（见图23-5）。CCPRxH:CCPRxL寄存器对可以在任意时间写入，但是在T2PR和T2TMR发生匹配之前，占空比值不会被锁存到10位缓冲区中。

公式23-2用于计算PWM脉冲宽度。公式23-3用于计算PWM占空比。

图23-5： PWM 10位对齐方式



公式23-2： 脉冲宽度

$$脉冲宽度 = (CCPRxH:CCPRxL\ 寄存器对) \bullet T_{OSC} \bullet (TMR2\ 预分频值)$$

公式23-3： 占空比

$$占空比 = \frac{(CCPRxH:CCPRxL\ 寄存器对)}{4(T2PR + 1)}$$

CCPRxH:CCPRxL寄存器对用于为PWM占空比提供双重缓冲。这种双重缓冲可以避免在PWM工作过程中产生毛刺。

8位定时器T2TMR寄存器与2位内部系统时钟（FOSC）或预分频器的2位一起构成10位时基。如果Timer2预分频比设置为1:1，则使用系统时钟。

当10位时基与CCPRxH:CCPRxL寄存器对匹配时，CCPx引脚被清零（见图23-4）。

23.4.6 PWM 分辨率

分辨率决定给定周期的可用占空比数。例如，10位分辨率将可得到1024个离散的占空比，而8位分辨率将可得到256个离散的占空比。

当T2PR为255时，最大PWM分辨率为10位。分辨率是T2PR寄存器值的函数，如公式23-4所示。

公式23-4： PWM 分辨率

$$分辨率 = \frac{\log[4(T2PR + 1)]}{\log(2)}\ 位$$

注： 如果脉宽值大于周期值，则指定的 PWM 引脚将保持不变。

表23-2： PWM频率和分辨率示例（FOSC = 20 MHz）

PWM频率	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
定时器预分频比	16	4	1	1	1	1
T2PR值	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
最高分辨率（位）	10	10	10	8	7	6.6

表23-3： PWM频率和分辨率示例（FOSC = 8 MHz）

PWM频率	1.22 kHz	4.90 kHz	19.61 kHz	76.92 kHz	153.85 kHz	200.0 kHz
定时器预分频比	16	4	1	1	1	1
T2PR值	0x65	0x65	0x65	0x19	0x0C	0x09
最高分辨率（位）	8	8	8	6	5	5

23.4.7 休眠模式下的操作

在休眠模式下，T2TMR寄存器将不会递增，模块状态也不会改变。如果CCPx引脚正在驱动一个值，则会继续驱动该值。当器件被唤醒时，T2TMR将从先前状态继续。

23.4.8 改变系统时钟频率

PWM频率是由系统时钟频率得到的。系统时钟频率的任何改变将导致PWM频率的改变。更多详细信息，请参见**第7.0节“振荡器模块（带故障保护时钟监视器）”**。

23.4.9 复位的影响

任何复位都将强制所有端口为输入模式，并强制CCP寄存器为其复位状态。

23.5 寄存器定义：CCP控制

CCP外设的长位名称前缀如下所示。更多信息，请参见**第1.3.2.2节“长位名称”**。

外设	位名称前缀
CCP1	CCP1
CCP2	CCP2
CCP3	CCP3
CCP4	CCP4

寄存器23-1：CCPxCON：CCPx控制寄存器

R/W-0/0	U-0	R-x	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN	—	OUT	FMT	MODE<3:0>			
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **EN**：CCP模块使能位
1 = 使能CCP
0 = 禁止CCP

bit 6 **未实现**：读为0

bit 5 **OUT**：CCPx输出数据位（只读）

bit 4 **FMT**：CCPW（脉宽）对齐方式位
MODE = 捕捉模式：
未使用
MODE = 比较模式：
未使用
MODE = PWM模式：
1 = 左对齐格式
0 = 右对齐格式

bit 3-0 **MODE<3:0>**：CCPx模式选择位

MODE	工作模式	操作	将CCPxIF置1
11xx	PWM	PWM操作	有
1011	比较	脉冲输出；清零TMR1(2)	有
1010		脉冲输出	有
1001		清零输出(1)	有
1000		置1输出(1)	有
0111	捕捉	CCPx输入的每16个上升沿	有
0110		CCPx输入的每4个上升沿	有
0101		CCPx输入的每个上升沿	有
0100		CCPx输入的每个下降沿	有
0011		CCPx输入的每个边沿	有
0010	比较	翻转输出	有
0001		翻转输出；清零TMR1(2)	有
0000	禁止		—

注 1： 比较模式的置1和清零操作通过设置MODE = 4'b0000或EN = 0来复位。
2： 当MODE = 0001或1011时，与CCP模块相关的定时器清零。TMR1是CCP模块的默认选择，因此仅用于说明目的。

寄存器23-2：CCPTMRS0：CCP定时器控制寄存器0

R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1
C4TSEL<1:0>		C3TSEL<1:0>		C2TSEL<1:0>		C1TSEL<1:0>	
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **C4TSEL<1:0>：**CCP4定时器选择位
11 = 在捕捉/比较模式下，CCP4以Timer5作为定时器；在PWM模式下，CCP4以Timer6作为定时器
10 = 在捕捉/比较模式下，CCP4以Timer3作为定时器；在PWM模式下，CCP4以Timer4作为定时器
01 = 在捕捉/比较模式下，CCP4以Timer1作为定时器；在PWM模式下，CCP4以Timer2作为定时器
00 = 保留

bit 5-4 **C3TSEL<1:0>：**CCP3定时器选择位
11 = 在捕捉/比较模式下，CCP3以Timer5作为定时器；在PWM模式下，CCP3以Timer6作为定时器
10 = 在捕捉/比较模式下，CCP3以Timer3作为定时器；在PWM模式下，CCP3以Timer4作为定时器
01 = 在捕捉/比较模式下，CCP3以Timer1作为定时器；在PWM模式下，CCP3以Timer2作为定时器
00 = 保留

bit 3-2 **C2TSEL<1:0>：**CCP2定时器选择位
11 =在捕捉/比较模式下，CCP2以Timer5作为定时器；在PWM模式下，CCP2以Timer6作为定时器
10 =在捕捉/比较模式下，CCP2以Timer3作为定时器；在PWM模式下，CCP2以Timer4作为定时器
01 =在捕捉/比较模式下，CCP2以Timer1作为定时器；在PWM模式下，CCP2以Timer2作为定时器
00 = 保留

bit 1-0 **C1TSEL<1:0>：**CCP1定时器选择位
11 =在捕捉/比较模式下，CCP1以Timer5作为定时器；在PWM模式下，CCP1以Timer6作为定时器
10 =在捕捉/比较模式下，CCP1以Timer3作为定时器；在PWM模式下，CCP1以Timer4作为定时器
01 =在捕捉/比较模式下，CCP1以Timer1作为定时器；在PWM模式下，CCP1以Timer2作为定时器
00 = 保留

寄存器23-3：CCPTMRS1：CCP定时器控制寄存器1

R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1
P8TSEL<1:0>		P7TSEL<1:0>		P6TSEL<1:0>		P5TSEL<1:0>	
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **P8TSEL<1:0>：**PWM8定时器选择位
11 = PWM8基于TMR8
10 = PWM8基于TMR6
01 = PWM8基于TMR4
00 = PWM8基于TMR2

bit 5-4 **P7TSEL<1:0>：**PWM7定时器选择位
11 = PWM7基于TMR8
10 = PWM7基于TMR6
01 = PWM7基于TMR4
00 = PWM7基于TMR2

bit 3-2 **P6TSEL<1:0>：**PWM6定时器选择位
11 = PWM6基于TMR8
10 = PWM6基于TMR6
01 = PWM6基于TMR4
00 = PWM6基于TMR2

bit 1-0 **P5TSEL<1:0>：**PWM5定时器选择位
11 = PWM5基于TMR8
10 = PWM5基于TMR6
01 = PWM5基于TMR4
00 = PWM5基于TMR2

寄存器23-4：CCPxCAP：捕捉输入选择多路开关寄存器

U-0	U-0	U-0	U-0	R/W-0/x	R/W-0/x	R/W-0/x	R/W-0/x
—	—	—	—	CTS<3:0>			
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-3 **未实现：**读为0

bit 2-0 **CTS<2:0>：**捕捉触发输入选择位

CTS<3:0>	连接			
	CCP1	CCP2	CCP3	CCP4
1111-1001	保留			
1000	CAN_rx_timestamp			
0111	CLC4_out			
0110	CLC3_out			
0101	CLC2_out			
0100	CLC1_out			
0011	IOC_Interrupt			
0010	CMP2_output			
0001	CMP1_output			
0000	通过CCP1PPS选择引脚	通过CCP2PPS选择引脚	通过CCP3PPS选择引脚	通过CCP4PPS选择引脚

寄存器23-5：CCPRxL：CCPx寄存器低字节

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
RL<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0

MODE = 捕捉模式：

RL<7:0>：捕捉的TMR1值的LSB

MODE = 比较模式：

RL<7:0>：LSB与TMR1值进行比较

MODE = PWM模式且FMT = 0：

RL<7:0>：CCPW<7:0>——脉冲宽度LS 8位

MODE = PWM模式且FMT = 1：

RL<7:6>：CCPW<1:0>——脉冲宽度LS 2位

RL<5:0>：未使用

寄存器23-6：CCPRxH：CCPx寄存器高字节

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
RH<7:0>							
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0

MODE = 捕捉模式：
RH<7:0>：捕捉的TMR1值的MSB
MODE = 比较模式：
RH<7:0>：MSB与TMR1值进行比较
MODE = PWM模式且FMT = 0：
RH<7:2>：未使用
RH<1:0>：CCPW<9:8>——脉冲宽度MS 2位
MODE = PWM模式且FMT = 1：
RH<7:0>：CCPW<9:2>——脉冲宽度MS 8位

表23-4： 与CCPx相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
CCPxCON	EN	—	OUT	FMT	MODE<3:0>				335
CCPxCAP	—	—	—	—	—	—	CTS<1:0>		338
CCPRxL	CCPRx<7:0>								338
CCPRxH	CCPRx<15:8>								339
CCPTMRS0	C4TSEL<1:0>		C3TSEL<1:0>		C2TSEL<1:0>		C1TSEL<1:0>		336

图注： — = 未实现位，读为0。CCP模块不使用阴影单元。

24.0 脉宽调制（PWM）

PWM模块可产生由占空比、周期和分辨率决定的脉宽调制信号，占空比、周期和分辨率则通过以下寄存器进行配置：

- TxPR
- TxCON
- PWMxDCH
- PWMxDCL
- PWMxCON

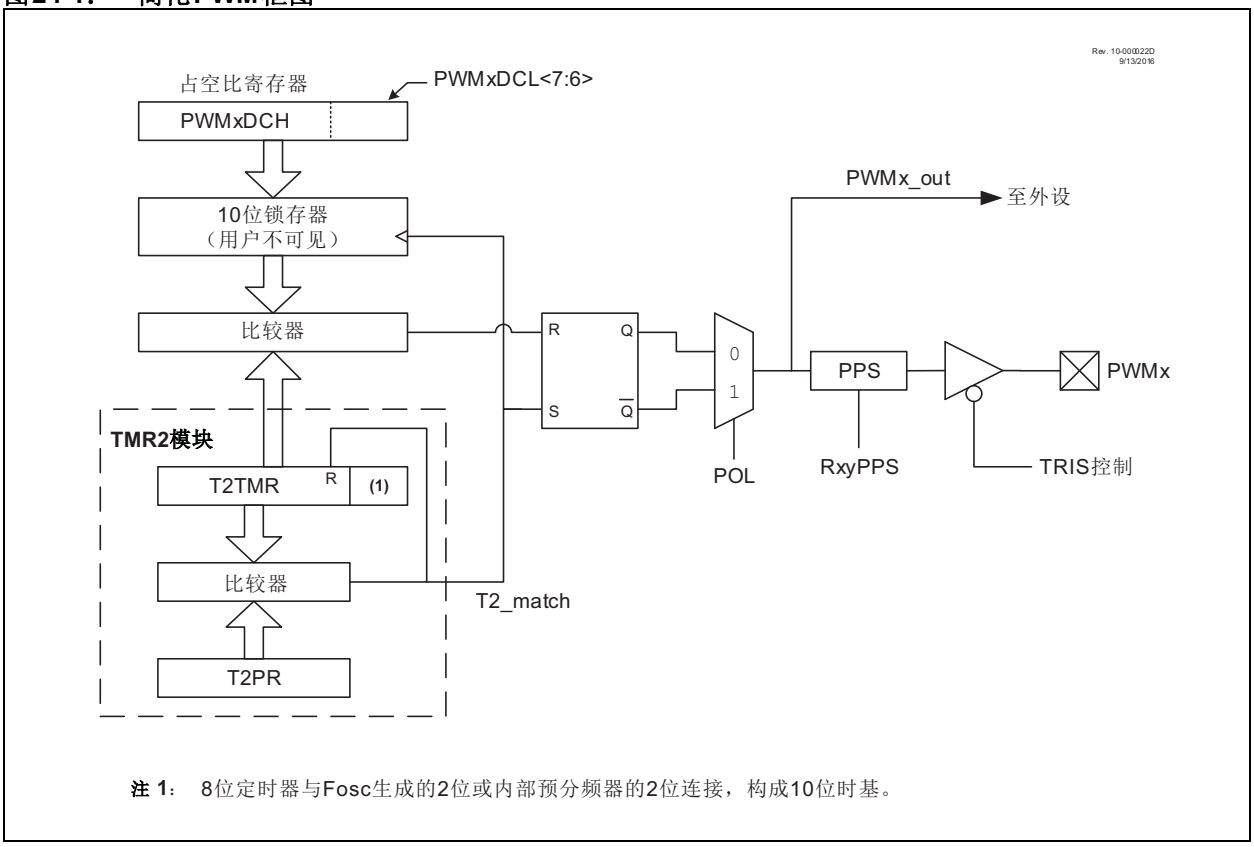
注： 必须将PWMx引脚对应的TRIS位清零，以使能该引脚上的PWM输出。

每个PWM模块可选择用于控制模块的定时器源。每个模块都可以独立选择定时器，通过CCPTMRS1寄存器（寄存器23-2）可以访问该定时器。请注意，以下部分针对T2TMR介绍了PWM模式操作。

图24-1给出了PWM操作的简化框图。

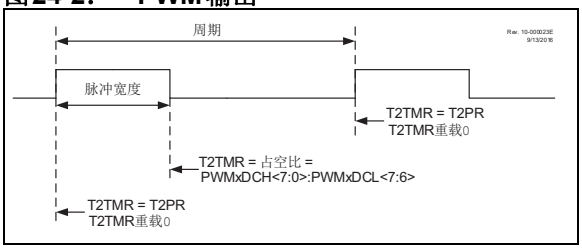
图24-2给出了PWM信号的典型波形。

图24-1： 简化PWM框图



注 1： 8位定时器与Fosc生成的2位或内部预分频器的2位连接，构成10位时基。

图24-2： PWM输出



关于如何设置该模块使之工作于PWM模式的详细步骤，请参见**第24.1.9节“使用PWMx引脚设置PWM操作”**。

24.1 PWMx引脚配置

所有PWM输出都与端口数据锁存器复用。用户必须通过清零相关的TRIS位将引脚配置为输出。

24.1.1 基本操作

PWM 模块可产生一个 10 位分辨率的输出。可使用CCPTMRS1寄存器中的PxTSEL位选择PWM定时器。PWMx的默认选择为T2TMR。请注意，以下部分针对T2TMR介绍了PWM模块操作。Timer2和T2PR用于设置PWM的周期。PWMxDCL和PWMxDCH寄存器用于配置占空比。周期由所有PWM模块共用，而占空比则独立进行控制。

注： 在确定 PWM 频率时不会用到 Timer2 后分频器。后分频器可用不同于PWM输出频率的频率进行伺服数据更新。

当T2TMR清零时，与Timer2相关的所有PWM输出都会置1。当T2TMR等于相应PWMxDCH（8 MSb）和PWMxDCL<7:6>（2 LSb）寄存器指定的值时，每个PWMx都会清零。当值大于等于T2PR时，PWM输出永远不会清零（占空比为100%）。

注： PWMxDCH和PWMxDCL寄存器是双重缓冲的。当Timer2与T2PR匹配时，缓冲区会发生更新。在定时器匹配发生之前更新两个寄存器时需要非常小心。

24.1.2 PWM输出极性

输出极性通过将PWMxCON寄存器的PWMxPOL位置1来进行反相。

24.1.3 PWM周期

PWM周期可通过Timer2的T2PR寄存器来指定。PWM周期可由公式24-1计算。为确保PWM正常工作，需要选择Fosc/4作为定时器2/4/6的时钟输入。

公式24-1： PWM周期

$$PWM\,周期 = [(T2PR) + 1] \bullet 4 \bullet T_{OSX} \bullet (TMR2\,预分频值)$$

注： Tosc = 1/Fosc

当T2TMR中的值与T2PR中的值相等时，在下一个递增周期将发生以下3个事件：

- T2TMR被清零
- PWM输出有效。（例外情况：当PWM占空比 = 0%时，PWM输出将保持无效。）
- PWMxDCH 和 PWMxDCL 寄存器的值被锁存到缓冲区中。

注： Timer2后分频器对PWM操作没有任何作用。

24.1.4 PWM占空比

PWM 占空比通过将一个 10 位值写入 PWMxDCH 和PWMxDCL 寄存器对来指定。PWMxDCH 寄存器包含高8位，而PWMxDCL<7:6>包含低2位。PWMxDCH和PWMxDCL寄存器可以在任意时刻写入。

公式24-2用于计算PWM脉冲宽度。

公式24-3用于计算PWM占空比。

公式24-2： 脉冲宽度

$$脉冲宽度 = (PWMxDCH:PWMxDCL<7:6>) \bullet T_{OSC} \bullet (TMR2\,预分频值)$$

注： Tosc = 1/Fosc

公式24-3： 占空比

$$占空比 = \frac{(PWMxDCH:PWMxDCL<7:6>)}{4(T2PR + 1)}$$

8位定时器T2TMR寄存器与1/Fosc的低2位连接，通过Timer2预分频器进行调节，构成10位时基。如果Timer2预分频比设置为1:1，则使用系统时钟。

24.1.5 PWM分辨率

分辨率决定给定周期的可用占空比数。例如，10位分辨率将可得到1024个离散的占空比，而8位分辨率将可得到256个离散的占空比。

当T2PR为255时，最大PWM分辨率为10位。分辨率是T2PR寄存器值的函数，如公式24-4所示。

公式24-4： PWM分辨率

$$分辨率 = \frac{log[4(T2PR + 1)]}{log(2)} 位$$

注： 如果脉宽值大于周期值，则指定的PWM引脚将保持不变。

表24-1： PWM频率和分辨率示例（Fosc = 20 MHz）

PWM频率	0.31 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
定时器预分频比	64	4	1	1	1	1
T2PR值	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
最高分辨率（位）	10	10	10	8	7	6.6

表24-2： PWM频率和分辨率示例（Fosc = 8 MHz）

PWM频率	0.31 kHz	4.90 kHz	19.61 kHz	76.92 kHz	153.85 kHz	200.0 kHz
定时器预分频比	64	4	1	1	1	1
T2PR值	0x65	0x65	0x65	0x19	0x0C	0x09
最高分辨率（位）	8	8	8	6	5	5

24.1.6 休眠模式下的操作

在休眠模式下，T2TMR寄存器将不会递增，模块状态也不会改变。如果PWMx引脚正在驱动一个值，则会继续驱动该值。当器件被唤醒时，T2TMR将从先前状态继续。

24.1.7 改变系统时钟频率

PWM频率是由系统时钟频率（Fosc）产生的。系统时钟频率的任何改变将导致PWM频率的改变。更多详细信息，请参见**第7.0节“振荡器模块（带故障保护时钟监视器）”**。

24.1.8 复位的影响

任何复位都将强制所有端口为输入模式，并强制PWM寄存器为其复位状态。

24.1.9 使用PWMx引脚设置PWM操作

当使用PWMx引脚将模块配置为PWM操作时，可采用以下步骤：

1. 通过将相关的TRIS位置1，禁止PWMx引脚输出驱动器。
2. 清零PWMxCON寄存器。
3. 将PWM周期值装入T2PR寄存器。
4. 将PWM占空比值装入PWMxDCH寄存器和PWMxDCL寄存器的bit <7:6>。
5. 配置并启动Timer2：
   - 清零相应PIR寄存器的TMR2IF中断标志位。请参见下面的“注1”。
   - 使用TxCLK寄存器选择Fosc/4作为定时器时钟源。必须这样做才能确保PWM模块正常工作。
   - 用Timer2预分频值配置T2CON寄存器的CKPS位。
   - 通过将T2CON寄存器的ON位置1来使能Timer2。
6. 使能PWM输出引脚并等待直到Timer2溢出，相应PIR寄存器的TMR2IF位置1。请参见下面的“注”。
7. 通过将关联的TRIS位清零并将所需的引脚PPS控制位置1，使能PWMx引脚输出驱动器。
8. 通过将相应值装入PWMxCON寄存器来配置PWM模块。

注 1： 为在第一个PWM输出时发送完整的占空比和周期，必须按给出的顺序执行上述步骤。如果以一个完整PWM信号开始并非至关重要，则用步骤8来代替步骤4。

2： 对于仅针对其他外设的操作，请禁止PWMx引脚输出。

24.1.10 为其他器件外设设置PWM操作

当将模块配置为PWM操作以供其他器件外设使用时，应执行以下步骤：

1. 通过将相关的TRIS位置1，禁止PWMx引脚输出驱动器。
2. 清零PWMxCON寄存器。
3. 将PWM周期值装入T2PR寄存器。
4. 将PWM占空比值装入PWMxDCH寄存器和PWMxDCL寄存器的bit <7:6>。
5. 配置并启动Timer2：
   - 清零相应PIR寄存器的TMR2IF中断标志位。请参见下面的“注1”。
   - 使用TxCLK寄存器选择Fosc/4作为定时器时钟源。必须这样做才能确保PWM模块正常工作。
   - 用Timer2预分频值配置T2CON寄存器的CKPS位。
   - 通过将T2CON寄存器的ON位置1来使能Timer2。
6. 使能PWM输出引脚：
   - 等待直到Timer2溢出，相应PIR寄存器的TMR2IF位置1。请参见下面的“注1”。
7. 通过将相应值装入PWMxCON寄存器来配置PWM模块。

注 1： 为在第一个PWM输出时发送完整的占空比和周期，设置过程必须包含上述步骤。如果在第一个输出时以完整的PWM信号起始并非至关重要，那么可以忽略步骤6。

24.2 寄存器定义：PWM控制

PWM外设的长位名称前缀如下所示。更多信息，请参见第1.3.2.2节“长位名称”。

外设	位名称前缀
PWM3	PWM3
PWM4	PWM4

寄存器24-1： PWMxCON：PWM控制寄存器

R/W-0/0	U-0	R-0/0	R/W-0/0	U-0	U-0	U-0	U-0
EN	—	OUT	POL	—	—	—	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN**：PWM模块使能位
1 = 使能PWM模块
0 = 禁止PWM模块

bit 6 **未实现**：读为0

bit 5 **OUT**：读取该位时的PWM模块输出电平

bit 4 **POL**：PWM输出极性选择位
1 = PWM输出反相
0 = PWM输出正常

bit 3-0 **未实现**：读为0

寄存器24-2： CCPTMRS1：CCP定时器控制寄存器1

R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1
P8TSEL<1:0>		P7TSEL<1:0>		P6TSEL<1:0>		P5TSEL<1:0>	
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **P8TSEL<1:0>**：PWM8定时器选择位
11 = PWM8基于TMR6
10 = PWM8基于TMR4
01 = PWM8基于TMR2
00 = 保留

bit 5-4 **P7TSEL<1:0>**：PWM7定时器选择位
11 = PWM7基于TMR6
10 = PWM7基于TMR4
01 = PWM7基于TMR2
00 = 保留

bit 3-2 **P6TSEL<1:0>**：PWM6定时器选择位
11 = PWM6基于TMR6
10 = PWM6基于TMR4
01 = PWM6基于TMR2
00 = 保留

bit 1-0 **P5TSEL<1:0>**：PWM5定时器选择位
11 = PWM5基于TMR6
10 = PWM5基于TMR4
01 = PWM5基于TMR2
00 = 保留

寄存器24-3： PWMxDCH：PWM占空比高位

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
DC<9:2>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **DC<9:2>**：PWM占空比高8位
这些位是PWM占空比的高8位。低2位位于PWMxDCL寄存器中。

寄存器24-4： PWMxDCL：PWM占空比低位

R/W-x/u	R/W-x/u	U-0	U-0	U-0	U-0	U-0	U-0
DC<1:0>		—	—	—	—	—	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **DC<1:0>**：PWM占空比低2位
这两位是PWM占空比的低2位。高8位位于PWMxDCH寄存器中。

bit 5-0 **未实现**：读为0

表24-3： 与PWM相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
PWMxCON	EN	—	OUT	POL	—	—	—	—	344
PWMxDCH	DC<9:2>								346
PWMxDCL	DC<1:0>		—	—	—	—	—	—	346
CCPTMRS1	P8TSEL<1:0>		P7TSEL<1:0>		P6TSEL<1:0>		P5TSEL<1:0>		345

图注： – = 未实现单元，读为0，u = 不变，x = 未知。PWM不使用阴影单元。

25.0 信号测量定时器（SMTx）

SMT是一个具有高级时钟和门控逻辑的24位计数器，它可以配置为测量各种数字信号参数，例如脉冲宽度、频率和占空比，以及两个信号上边沿之间的时间差。该器件仅实现了一个SMT模块。

SMT的特性包括：

- 24位定时器/计数器
  - 3个8位寄存器（SMTxL/H/U）
  - 可读写
  - 可选的16位工作模式
- 2个24位测量捕捉寄存器
- 1个24位周期匹配寄存器
- 多模式工作，包括相对时序测量
- 发生周期匹配时中断
- 多个时钟、门控和信号源
- 采集完成时产生中断
- 可读取当前输入值

图25-1： SMT框图

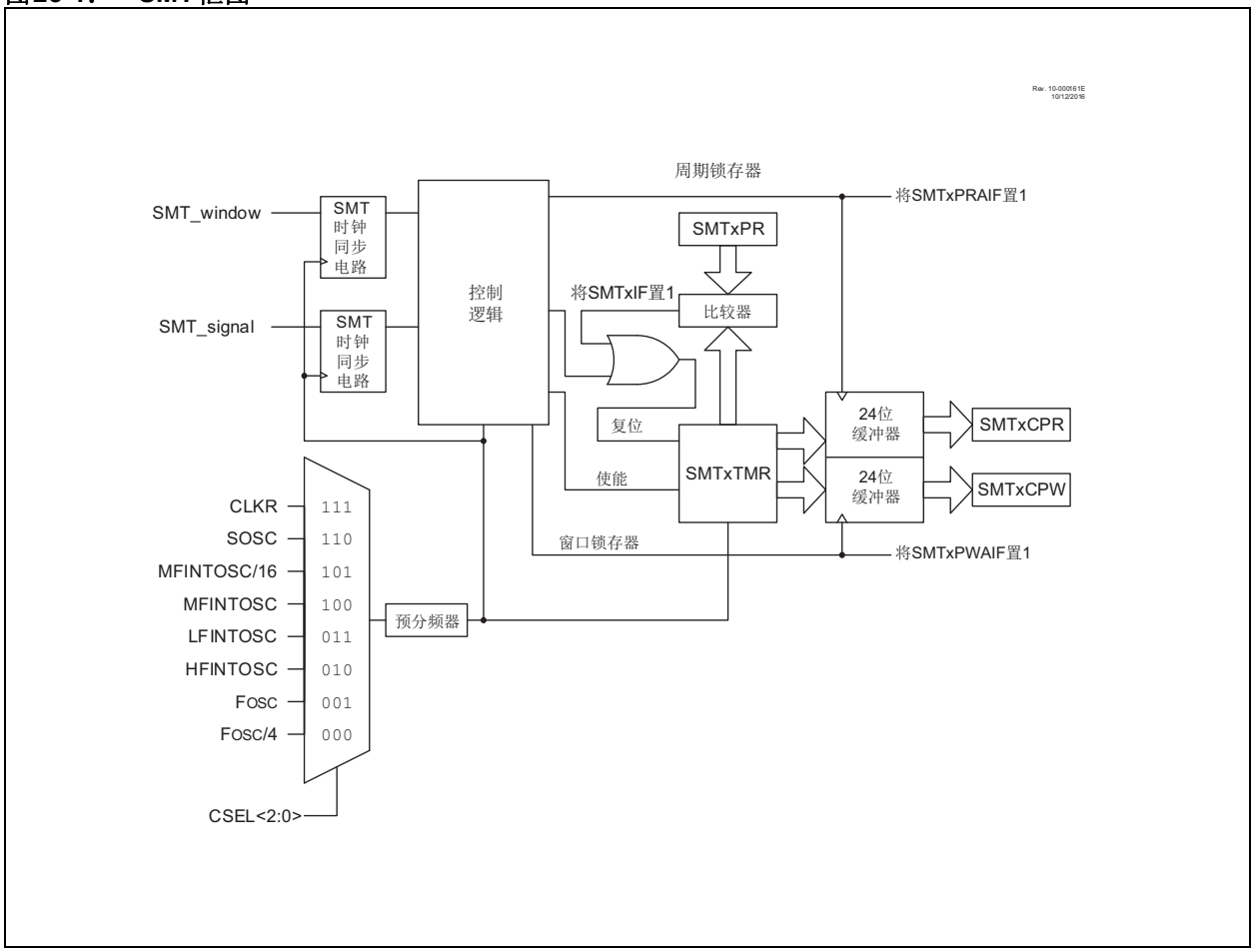
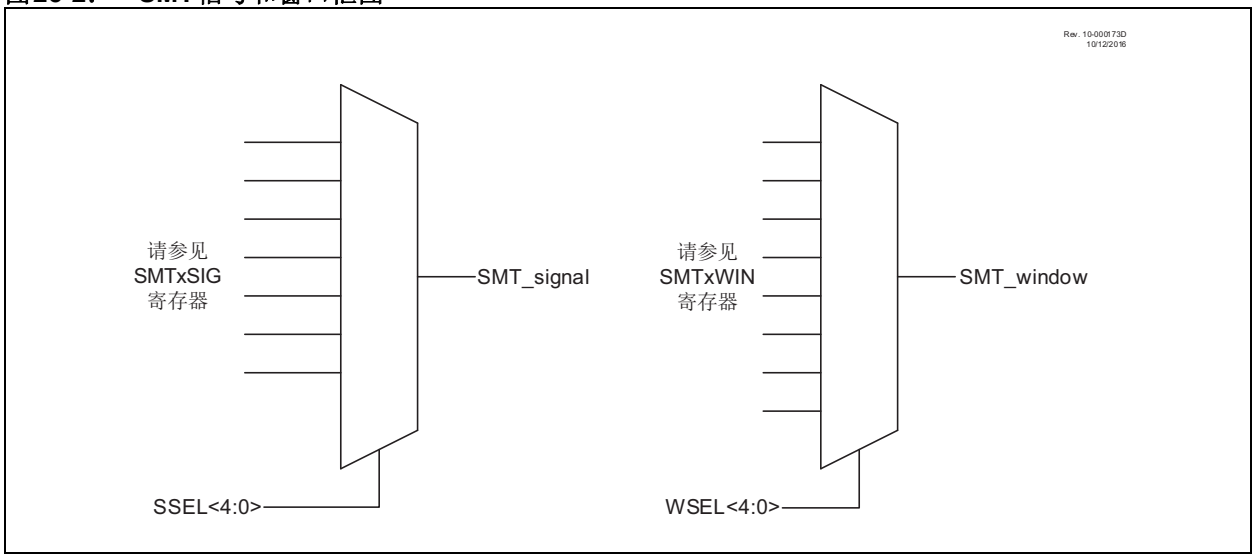


图25-2： SMT信号和窗口框图



25.1 SMT工作原理

该模块的核心是一个与复杂数据采集前端相结合的24位计数器SMTxTMR。根据所选的工作模式，SMT可以执行表25-1汇总的各种测量。

25.1.1 时钟源

SMT可用的时钟源包括：

- FOSC
- FOSC/4
- HFINTOSC 16 MHz
- LFINTOSC
- MFINTOSC 31.25 kHz

SMT时钟源通过配置SMTxCLK寄存器的CSEL<2:0>位进行选择。时钟源还可以使用SMTxCON0寄存器的PS<1:0>位进行预分频。预分频后的时钟源用于为计数器和该模块使用的任何同步逻辑提供时钟。

25.1.2 周期匹配中断

类似于其他定时器，SMT会在SMTxTMR计满返回到0时触发中断。它在SMTxTMR = SMTxPR时发生，无论模式如何。因此，在任何依赖于外部信号或窗口来复位定时器的模式下，需要将SMTxPR设置为大于预期信号或窗口的周期，这样才能正确工作。

25.2 基本定时器功能寄存器

SMTxTMR时基和SMTxCPW/SMTxPR/SMTxCPR缓冲寄存器提供了几种功能，可以使用软件手动进行更新。

25.2.1 时基

SMTxTMR是一个24位计数器，它是SMT的核心。在SMT的每种模式下，它用作进行测量的基本计数器/定时器。通过将SMTxSTAT寄存器的RST位置1，可以将它复位为值24'h00_0000。可以用软件读写它，但并不保证原子访问，因此应当仅在GO = 0时读写SMTxTMR，或软件应采取其他措施来以确保SMTxTMR读/写的完整性。

25.2.2 脉宽锁存器寄存器

SMTxCPW寄存器为24位SMT脉宽锁存器。它们用于在被各种信号触发时锁存SMTxTMR的值，这些信号由SMT当前所处的模式决定。通过将SMTxSTAT寄存器的CPWUP位置1，也可以使用SMTxTMR的当前值来更新SMTxCPW寄存器。

25.2.3 周期锁存器寄存器

SMTxCPR寄存器为24位SMT周期锁存器。它们用于在被各种其他信号触发时锁存SMTxTMR的其他值，这些其他信号由SMT当前所处的模式决定。

通过将SMTxSTAT寄存器的CPRUP位置1，也可以使用SMTxTMR的当前值来更新SMTxCPR寄存器。

25.3 暂停操作

使用SMTxCON0寄存器中的STP位，可以阻止计数器发生计满返回。当使能暂停时，周期匹配中断会持续发生，直到复位SMTxTMR为止（通过手动复位（**第25.2.1节“时基”**）或通过用软件清零SMTxCON1寄存器的GO位并写入SMTxTMR值）。

25.4 极性控制

SMT的3个输入信号具有极性控制功能，决定它们是高电平有效/正边沿还是低电平有效/负边沿信号。

以下位适用于极性控制：

- WSEL位（窗口极性）
- SSEL位（信号极性）
- CSEL位（时钟极性）

这些位位于SMTxCON0寄存器中。

25.5 状态信息

SMT可以为用户提供输入状态信息，而无需处理传入信号的极性。

25.5.1 窗口状态

窗口状态通过SMTxSTAT寄存器的WS位确定。该位仅在窗口测量、门控计数器和门控窗口测量模式下使用，并且仅在TS = 1时有效，在非计数器模式下会被延迟，被延迟的时间量等于同步器延时。

25.5.2 信号状态

信号状态通过SMTxSTAT寄存器的AS位确定。该位在除窗口测量、行程时间和捕捉测量之外的所有其他模式下使用，并且仅在TS = 1时有效，在非计数器模式下会被延迟，被延迟的时间量等于同步器延时。

25.5.3 运行状态

定时器运行状态通过SMTxSTAT寄存器的TS位确定，在非计数器模式下会被延迟，被延迟的时间量等于同步器延时。

25.6 工作模式

表25-1对工作模式进行了汇总。后面几节提供了关于如何使用这些模式的详细说明和示例。请注意，所有波形均假定WPOL/SPOL/CPOL = 0。当WPOL/SPOL/CPOL = 1时，所有SMTSIGx、SMTWINx和SMT时钟信号的极性将与所示极性相反。对于所有模式，REPEAT 位控制是进行重复采集还是单次采集。当REPEAT = 0（单次采集模式）时，则在完成采集时，定时器将停止递增，GO位将发生复位。否则，定时器将继续工作，允许后续采集覆盖先前的采集，直到用软件停止定时器为止。

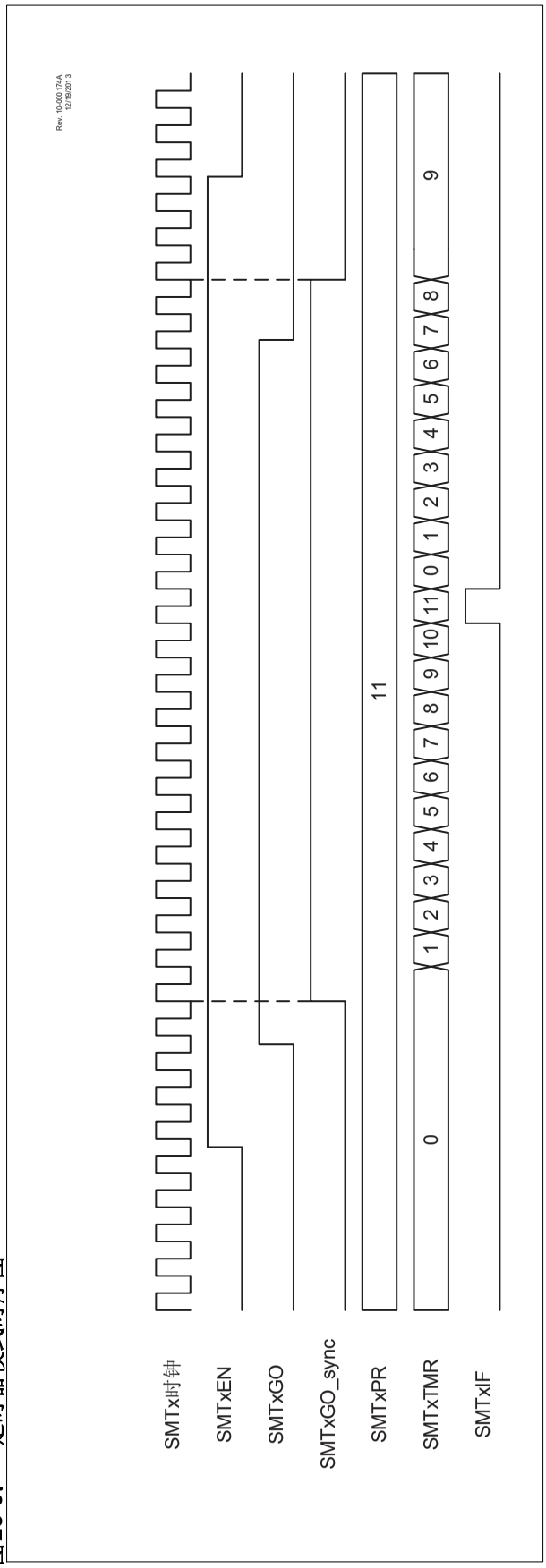
25.6.1 定时器模式

定时器模式是最简单的工作模式，SMTxTMR在该模式下用作16/24位定时器。该模式下不会发生数据采集。只要用软件将GO位置1，定时器就会一直递增。没有SMT窗口或SMT信号事件会影响GO位。所有时序都与SMT时钟源进行同步。当定时器发生周期匹配（SMTxTMR = SMTxPR）时，SMTxTMR 会发生复位，并触发周期匹配中断。请参见图25-3。

表25-1： 工作模式

模式	工作模式	同步操作	参考
0000	定时器	是	**第25.6.1节“定时器模式”**
0001	门控定时器	是	**第25.6.2节“门控定时器模式”**
0010	周期和占空比采集	是	**第25.6.3节“周期和占空比模式”**
0011	高电平和低电平时间测量	是	**第25.6.4节“高电平和低电平测量模式”**
0100	窗口测量	是	**第25.6.5节“窗口测量模式”**
0101	门控窗口测量	是	**第25.6.6节“门控窗口测量模式”**
0110	行程时间	是	**第25.6.7节“行程时间测量模式”**
0111	捕捉	是	**第25.6.8节“捕捉模式”**
1000	计数器	否	**第25.6.9节“计数器模式”**
1001	门控计数器	否	**第25.6.10节“门控计数器模式”**
1010	窗口计数器	否	**第25.6.11节“窗口计数器模式”**
1011-1111	保留	—	—

图 25-3：　定时器模式时序图



25.6.2 门控定时器模式

门控定时器模式使用SMTSIGx输入来控制SMTxTMR是否进行递增。在外部信号的下降沿，SMTxCPW寄存器会被更新为SMTxTMR的当前值。图25-4和图25-5给出了进行重复采集和单次采集的示例波形。

图25-4： 门控定时器模式重复采集时序图

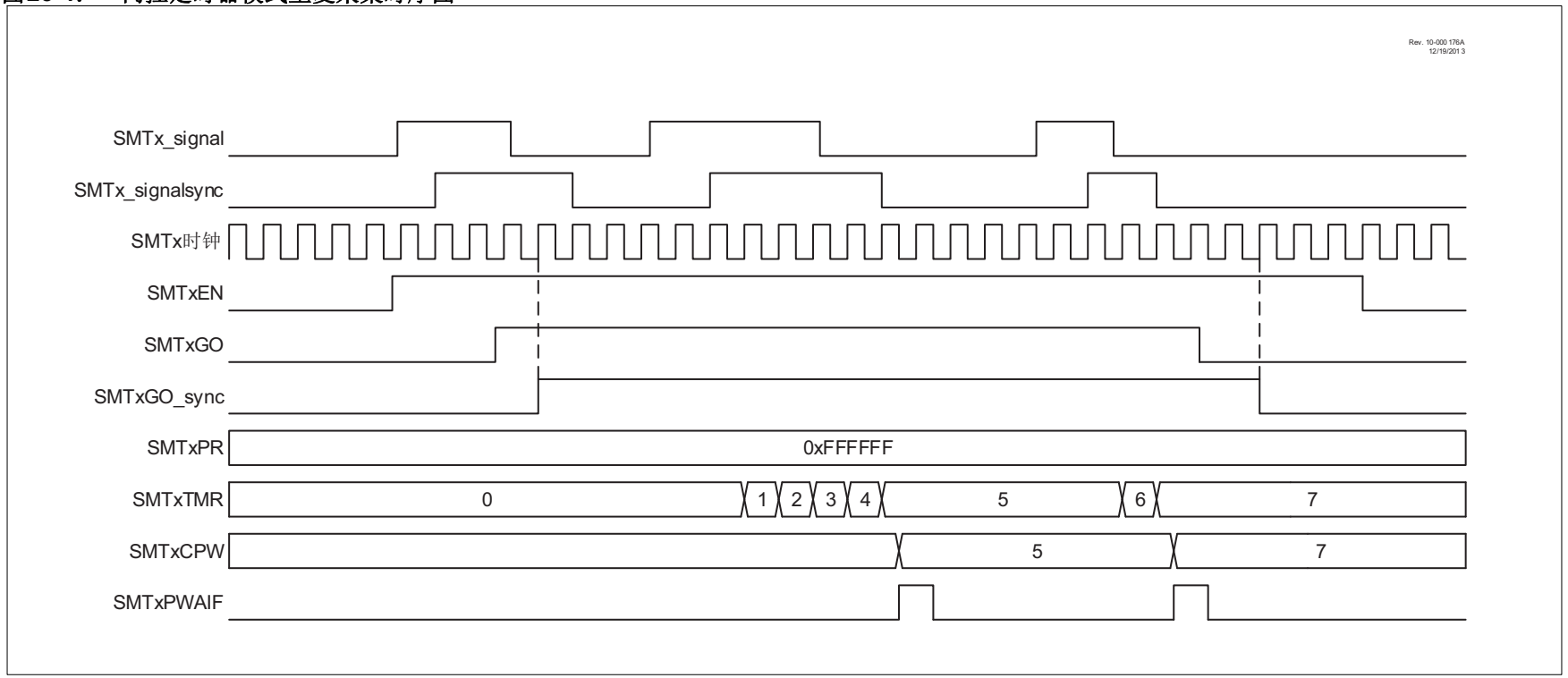
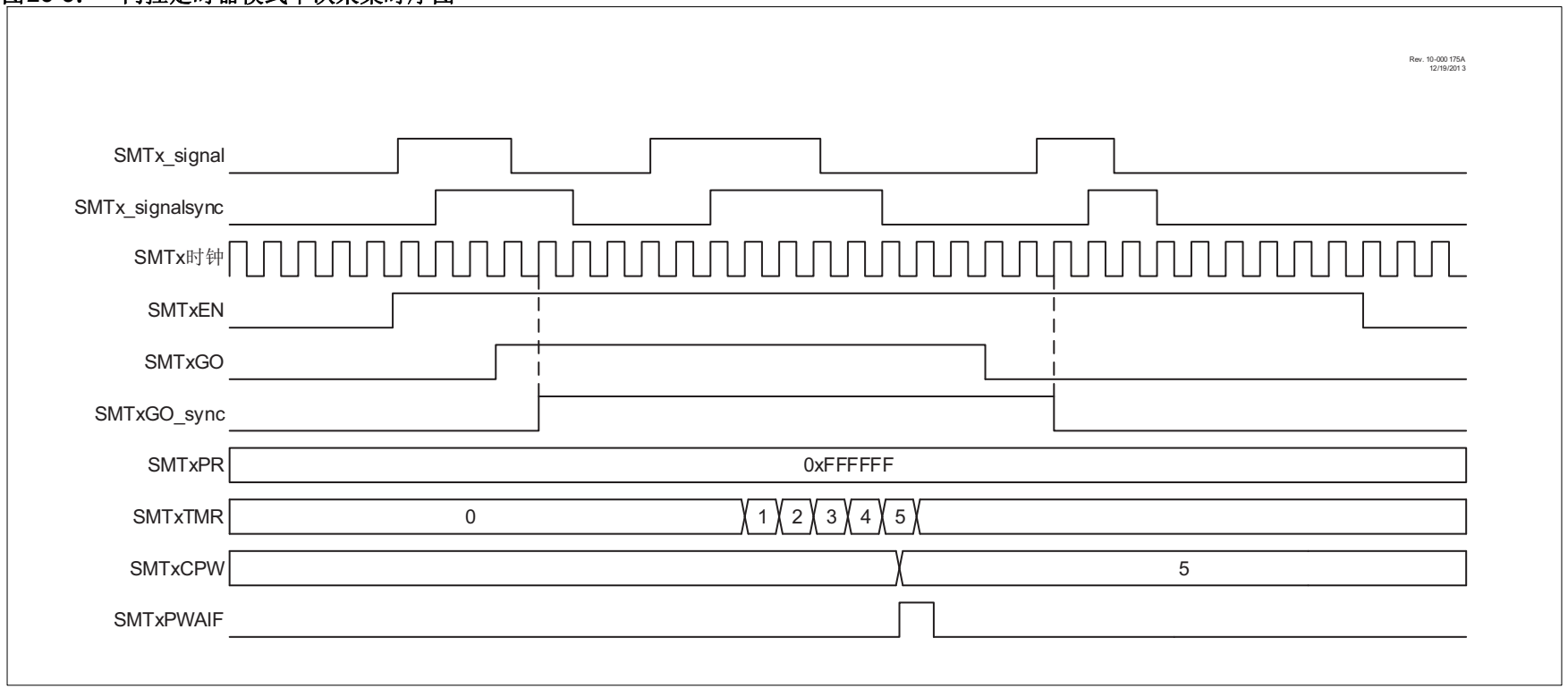


图25-5： 门控定时器模式单次采集时序图



25.6.3 周期和占空比模式

在占空比模式下，SMTx_signal的占空比或周期（取决于极性）都可基于SMT时钟进行采集。CPW寄存器在信号的下降沿更新，CPR寄存器在信号的上升沿更新，并且SMTxTMR复位为0x0001。此外，在SMT处于单次采集模式时，GO位是在上升沿发生复位。请参见图25-6和图25-7。

图25-6： 周期和占空比重复采集模式时序图

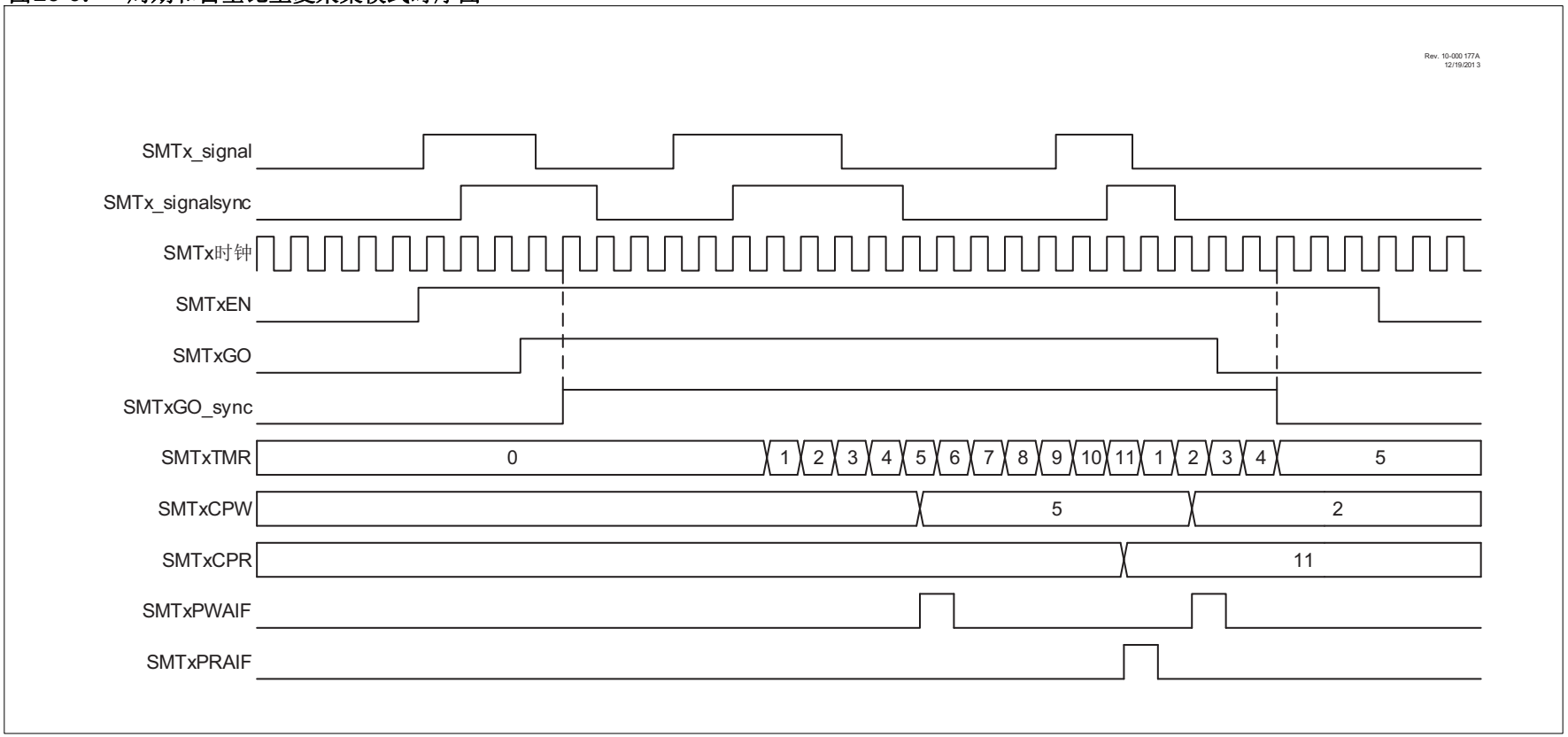
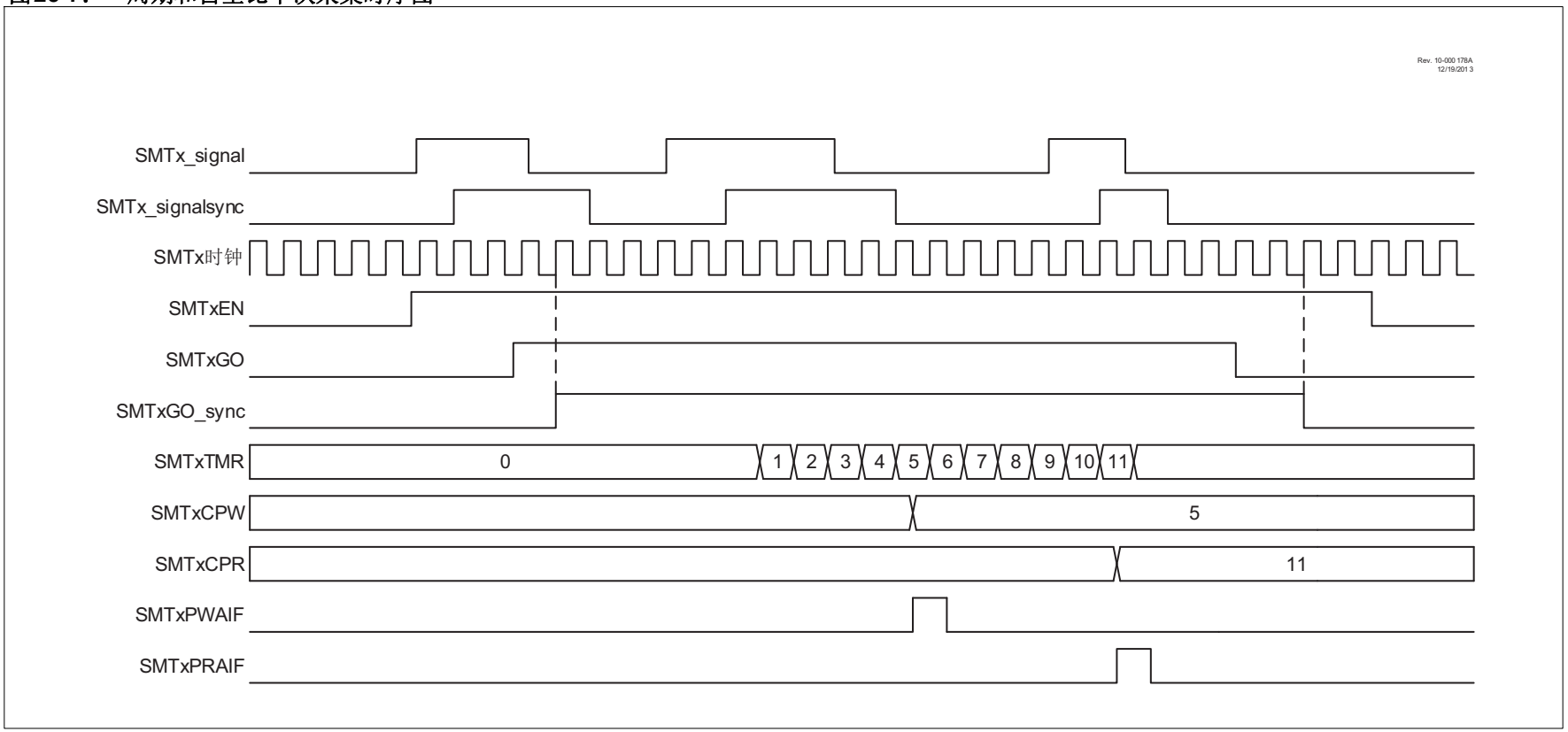


图25-7： 周期和占空比单次采集时序图



25.6.4 高电平和低电平测量模式

该模式相对于SMT时钟测量SMTSIGx的高电平和低电平脉冲时间。它在SMTSIGx输入的上升沿开始递增SMTxTMR，在下降沿使用值更新SMTxCPW寄存器并复位SMTxTMR，然后重新开始递增。在检测到另一个上升沿时，它会使用其当前值更新SMTxCPR寄存器并再次复位SMTxTMR，然后重新开始递增。请参见图25-8和图25-9。

图25-8： 高电平和低电平测量模式重复采集时序图

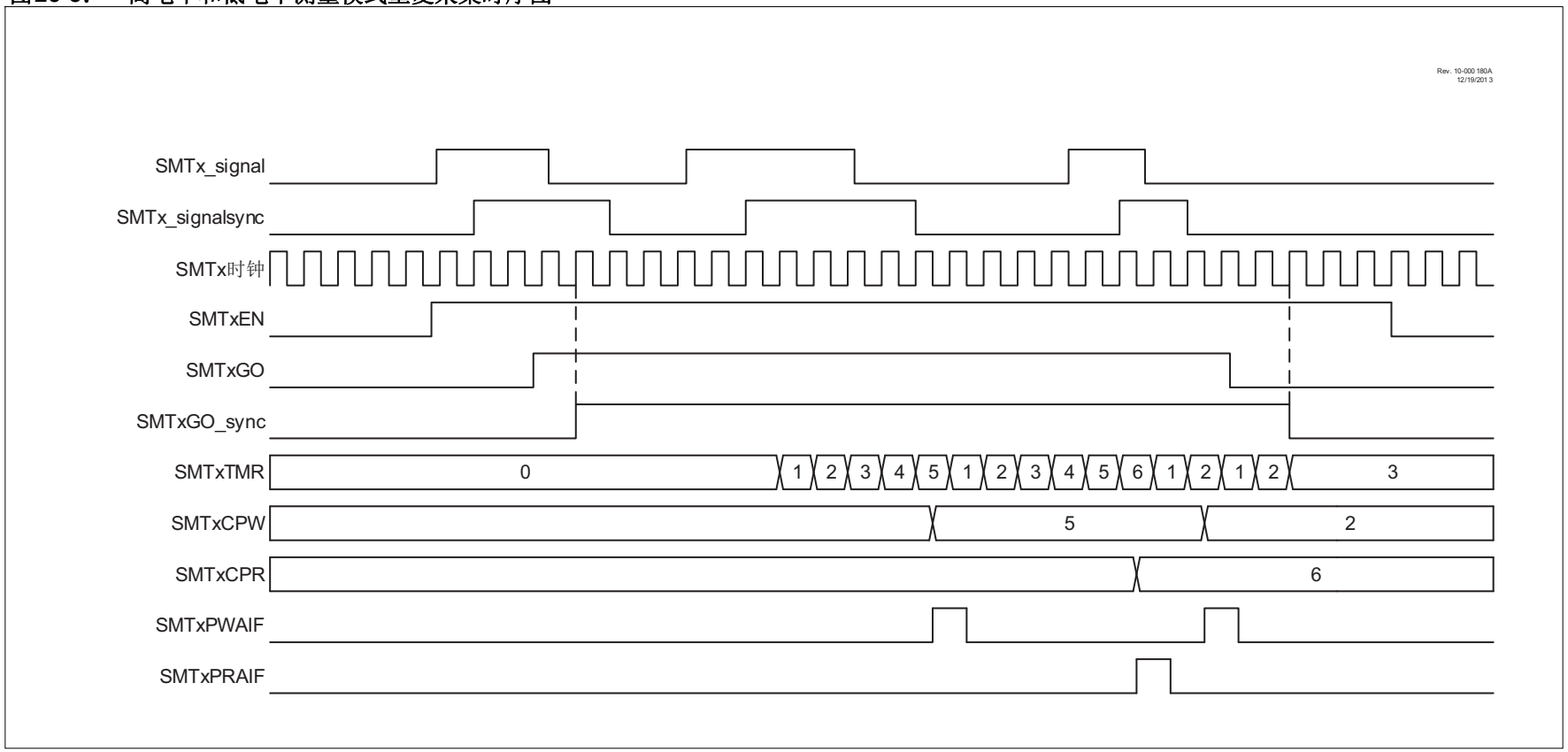
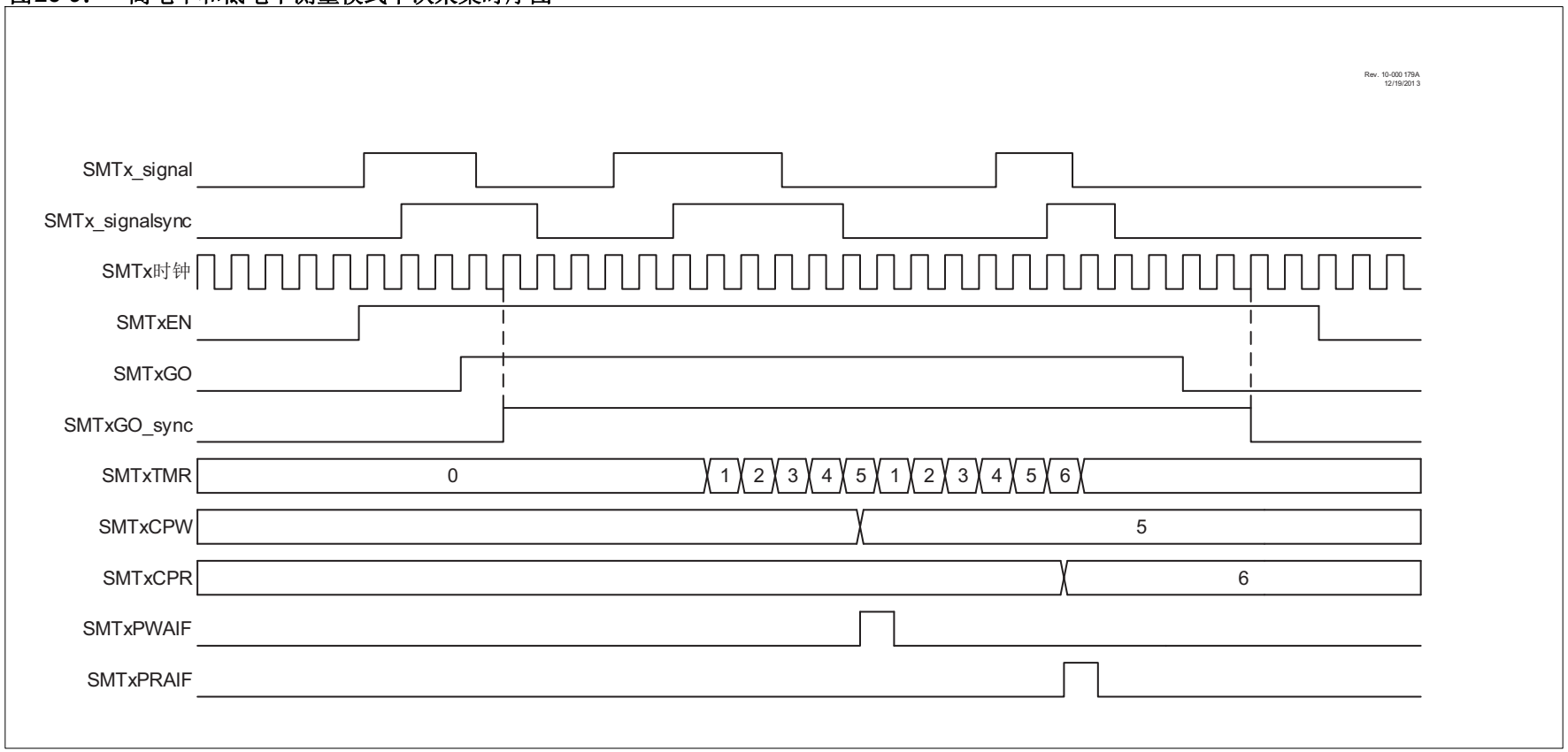


图25-9： 高电平和低电平测量模式单次采集时序图



25.6.5 窗口测量模式

该模式测量SMT的SMTWINx输入的窗口持续时间。它在SMTWINx输入的上升沿开始递增定时器，在第二个上升沿使用定时器的值更新SMTxCPR寄存器并复位定时器。请参见图25-10和图25-11。

图 25-10： 窗口测量模式重复采集时序图

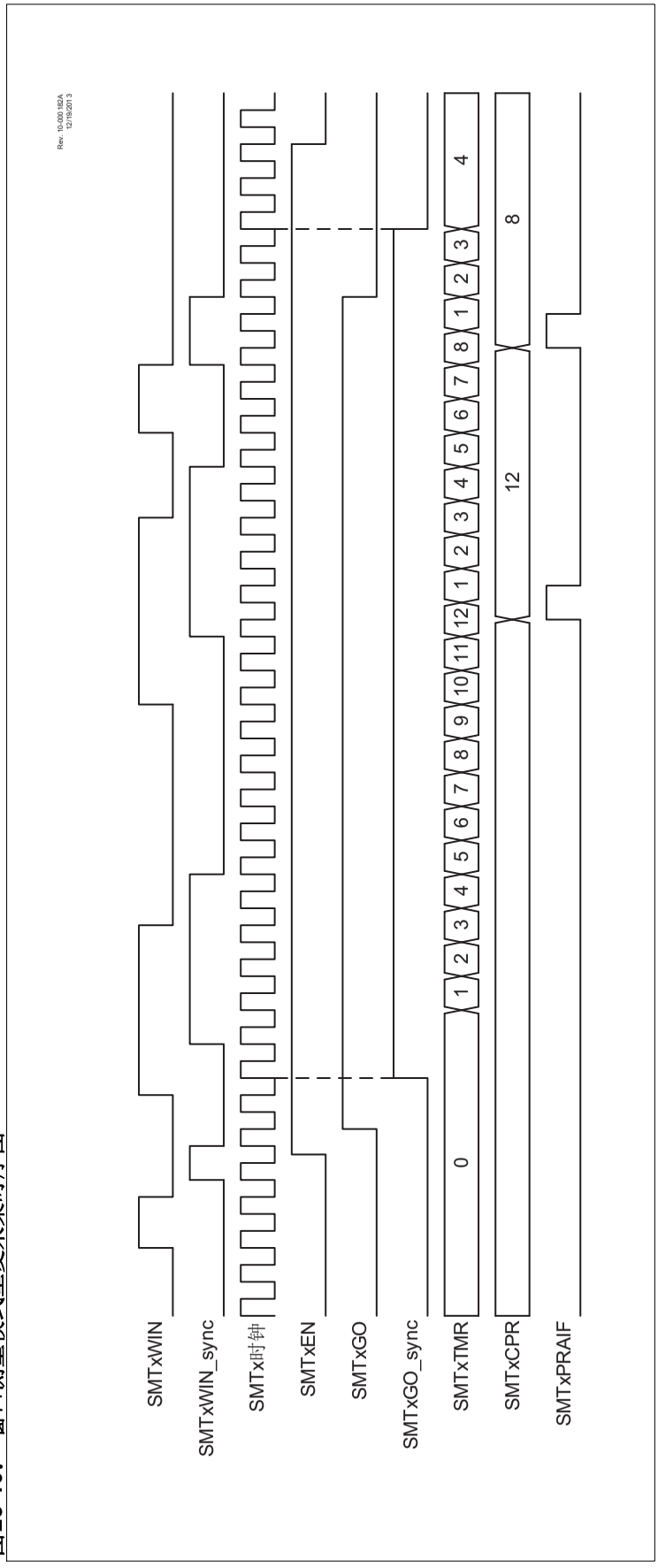
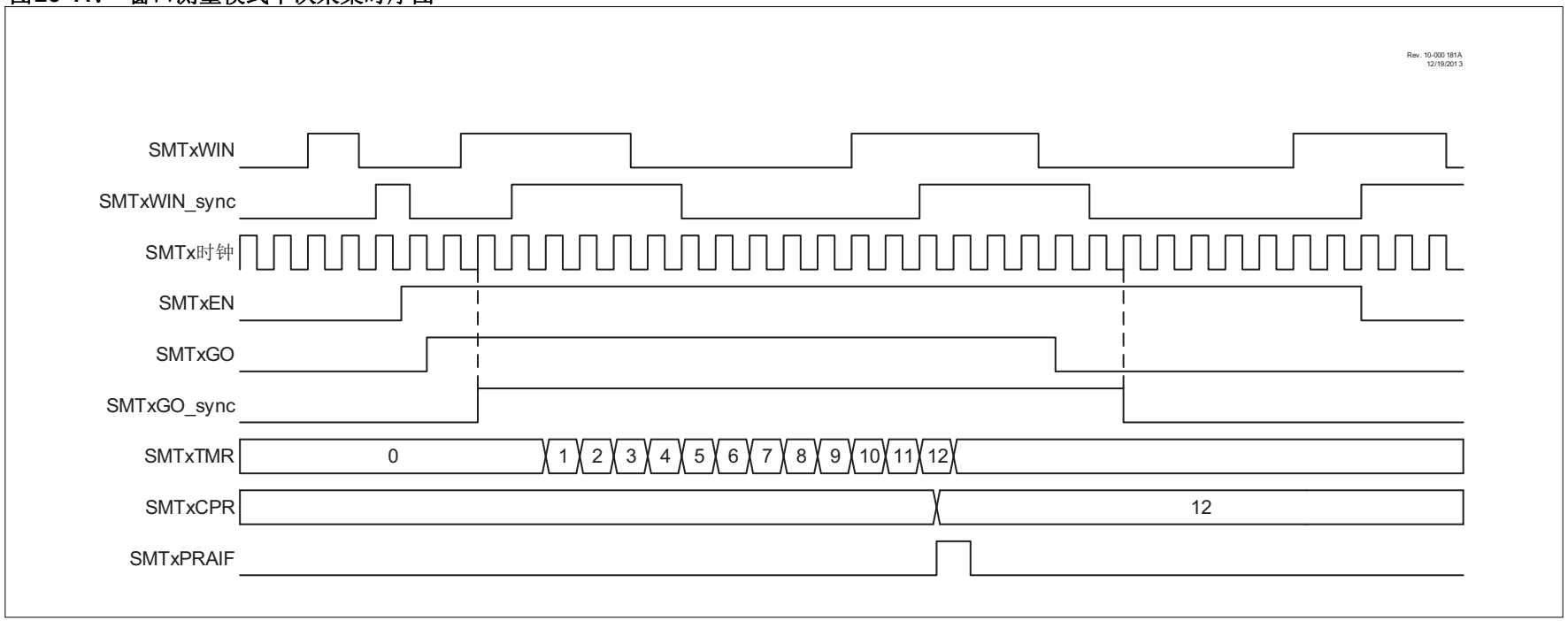


图25-11： 窗口测量模式单次采集时序图



25.6.6 门控窗口测量模式

该模式测量SMTx_signal输入在一个已知输入窗口中的占空比。它的实现方式是，在SMTx_signal输入为高电平时，对于时钟信号的每个脉冲递增定时器，并在SMTWINx输入第一个上升沿之后的每个上升沿更新SMTxCPR寄存器并复位定时器。请参见图25-12和图25-13。

图25-12： 门控窗口测量模式重复采集时序图

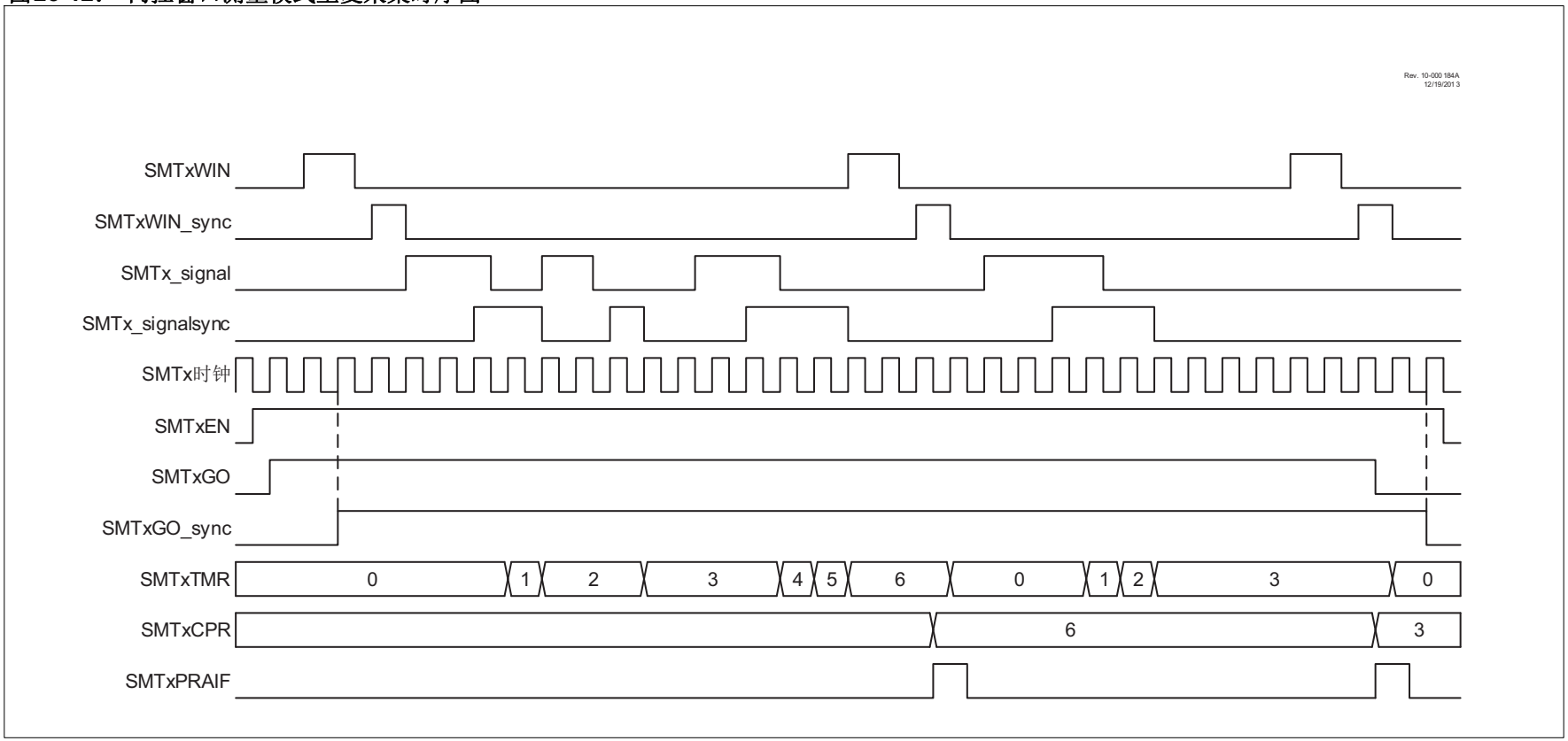
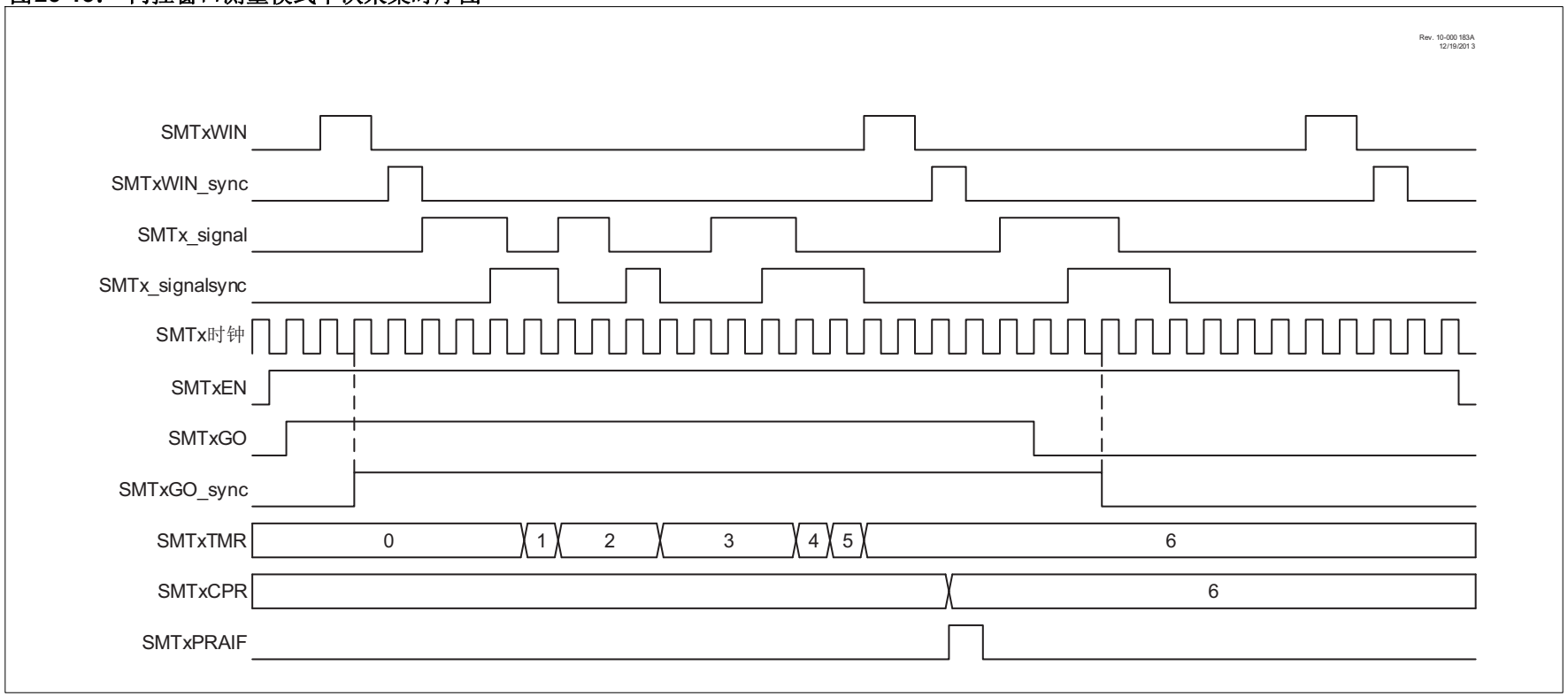


图25-13： 门控窗口测量模式单次采集时序图



25.6.7 行程时间测量模式

该模式测量SMTWINx输入上升沿和SMTx_signal输入上升沿之间的时间间隔，即在SMTWINx输入上检测到上升沿时开始递增定时器，在SMTx_signal输入上检测到上升沿时更新SMTxCPR寄存器并复位定时器。在出现两个SMTWINx上升沿而未出现SMTx_signal上升沿的情况下，它会使用定时器的当前值更新SMTxCPW寄存器并复位定时器值。请参见图25-14和图25-15。

图25-14： 行程时间模式重复采集时序图

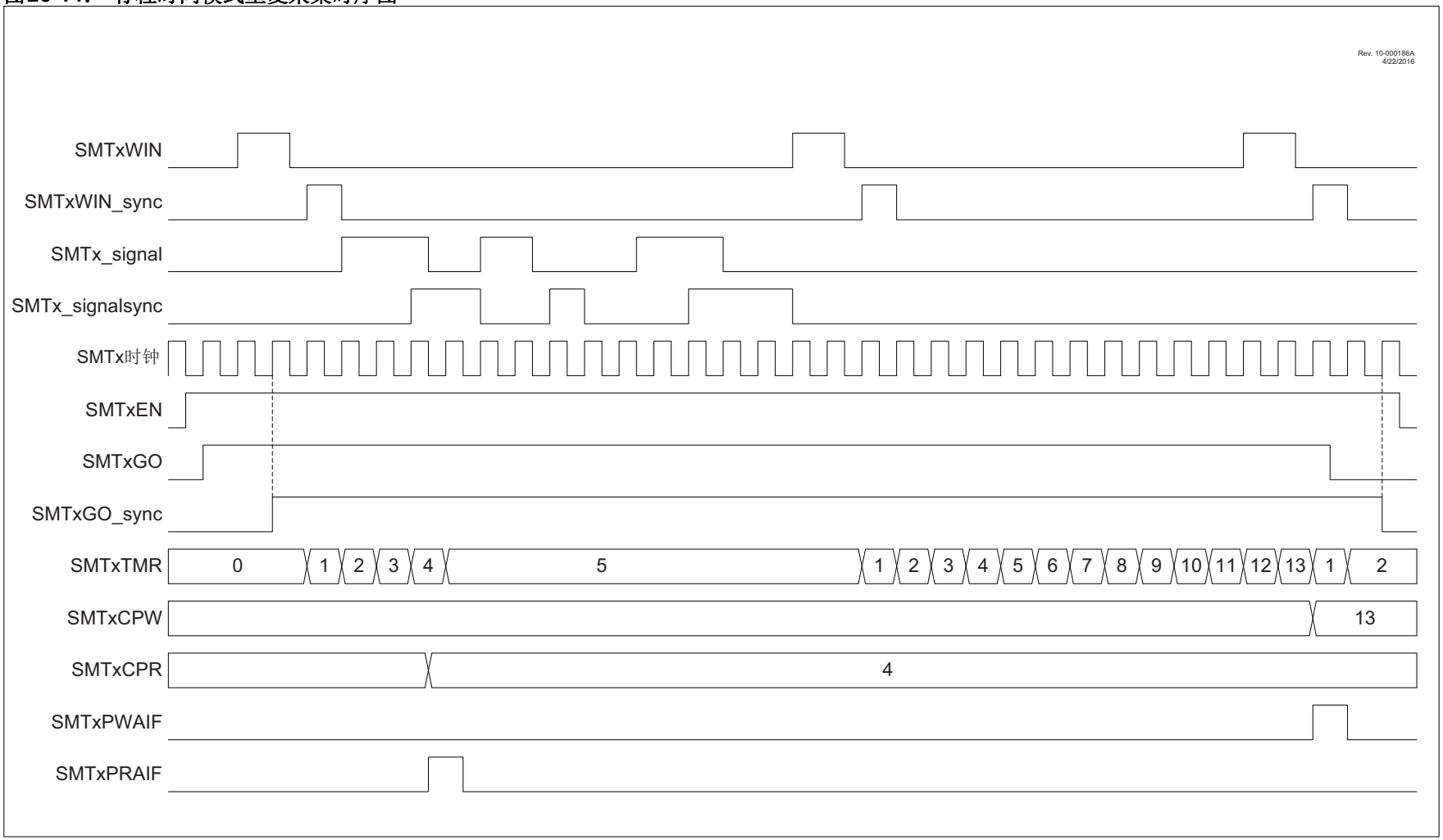
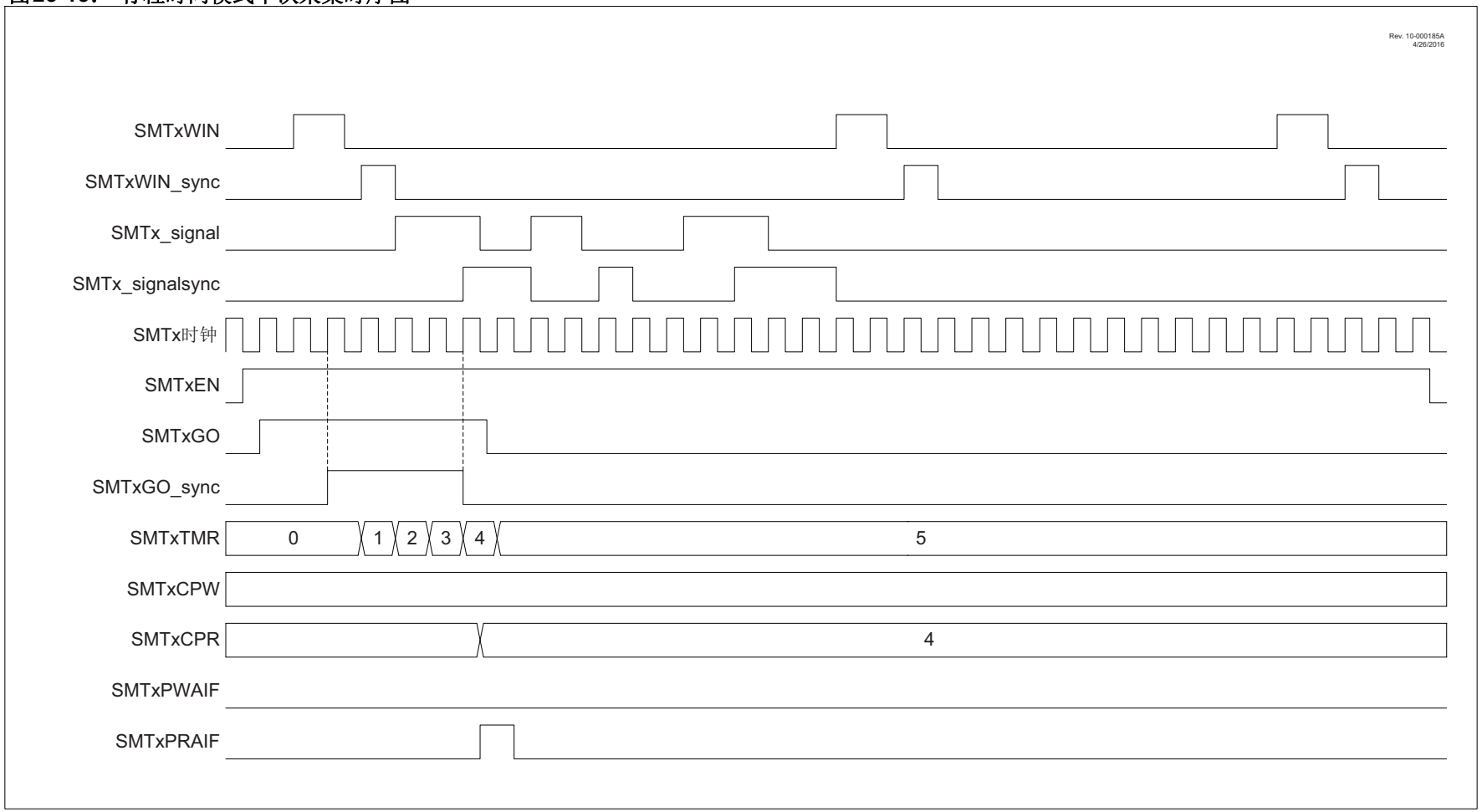


图25-15： 行程时间模式单次采集时序图



25.6.8 捕捉模式

该模式基于SMTWINx输入上的上升沿或下降沿捕捉定时器值并触发中断。它模仿了CCP模块的捕捉功能。定时器在GO位置1时开始递增，并在SMTWINx的每个上升沿更新SMTxCPR寄存器的值，在SMTWINx的每个下降沿更新CPW寄存器的值。在该模式下，任何硬件条件都不会复位定时器，如果需要，必须用软件复位。请参见图25-16和图25-17。

图25-16： 捕捉模式重复采集时序图

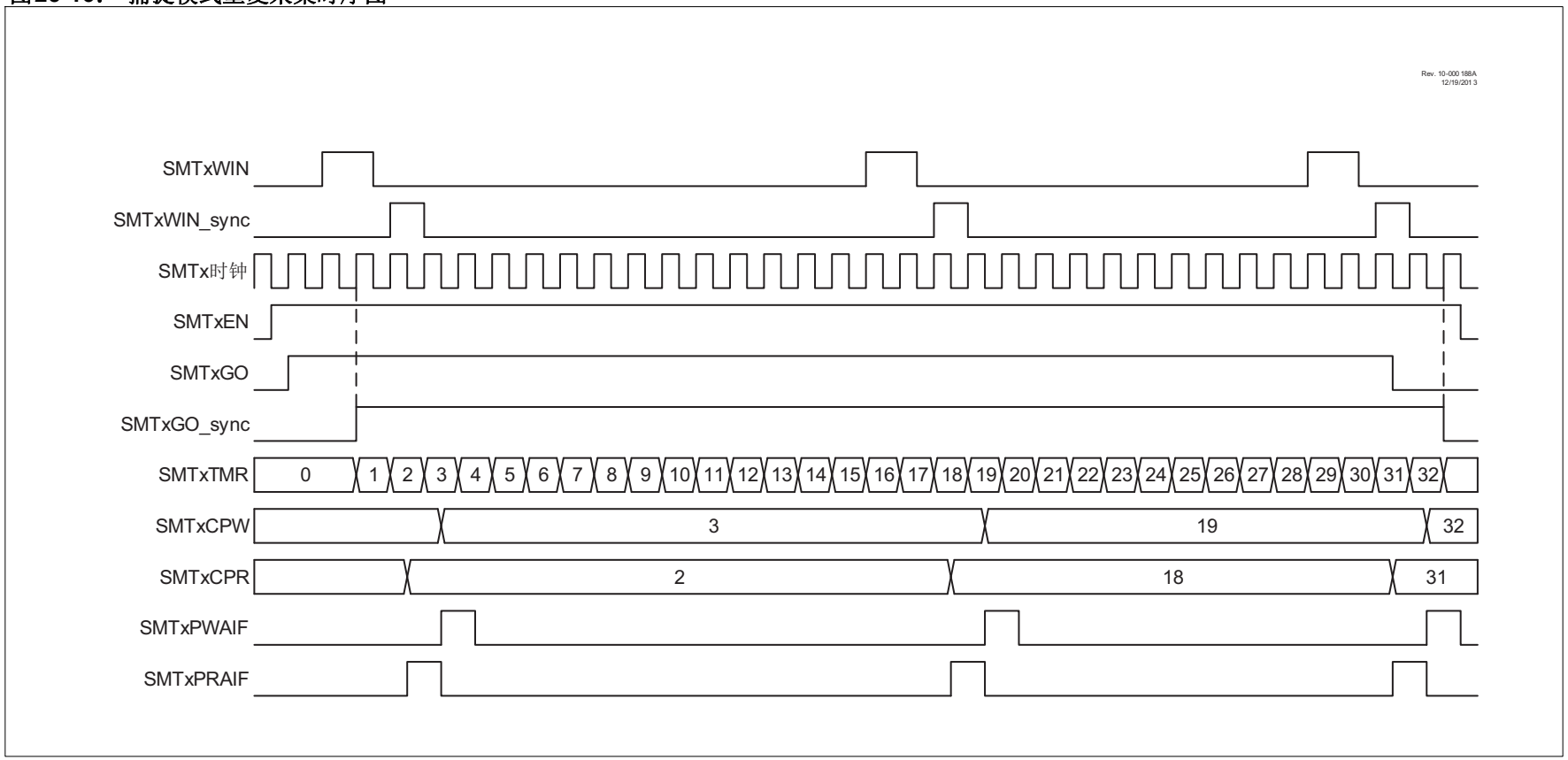
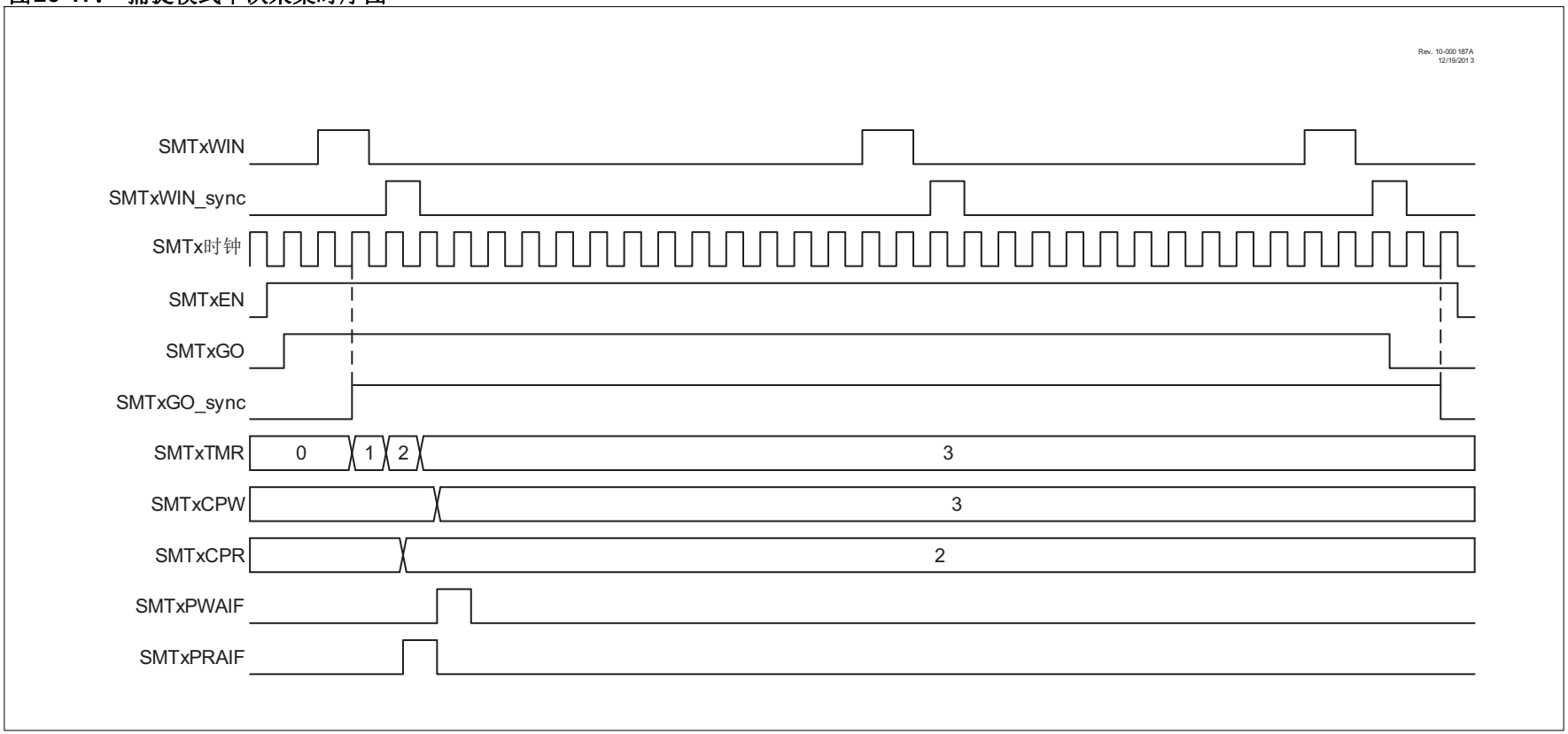


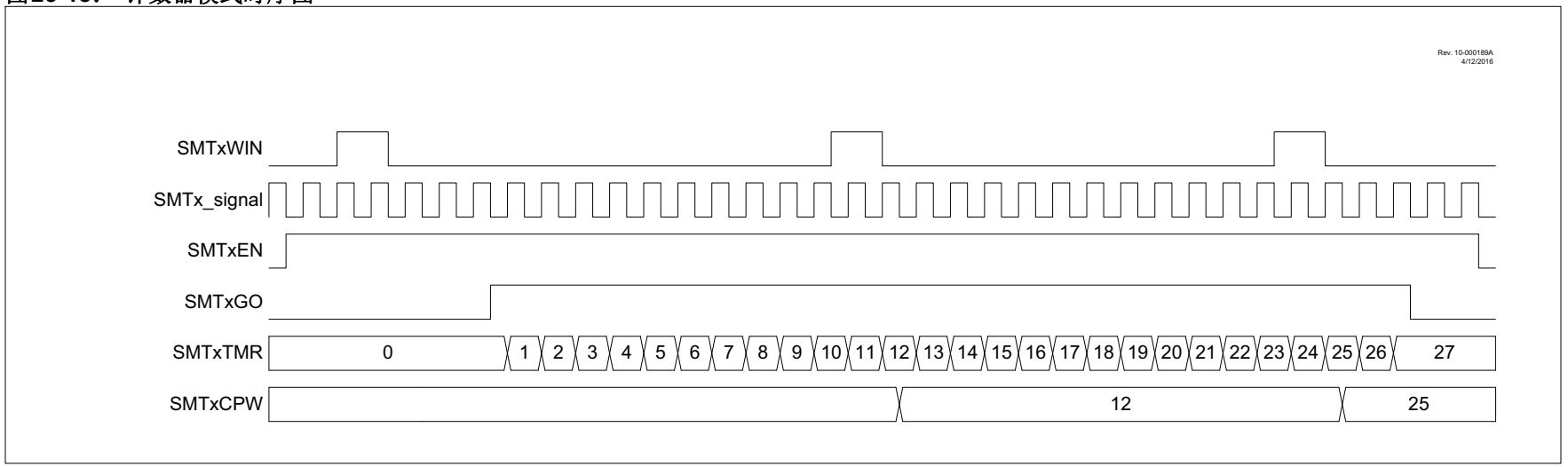
图25-17： 捕捉模式单次采集时序图



25.6.9 计数器模式

该模式对于SMTx_signal输入的每个脉冲递增定时器。该模式是与SMT时钟异步的，它使用SMTx_signal作为时间源。可以在SMTxWIN输入的上升沿将SMTxCPW寄存器更新为当前SMTxTMR值。请参见图25-18。

图25-18： 计数器模式时序图



25.6.10 门控计数器模式

该模式对SMTx_signal输入上的脉冲进行计数，并通过SMTxWIN输入进行门控。它在检测到SMTxWIN输入的上升沿时开始递增定时器，并在SMTxWIN输入的下降沿更新SMTxCPW寄存器。请参见图25-19和图25-20。

图25-19： 门控计数器模式重复采集时序图

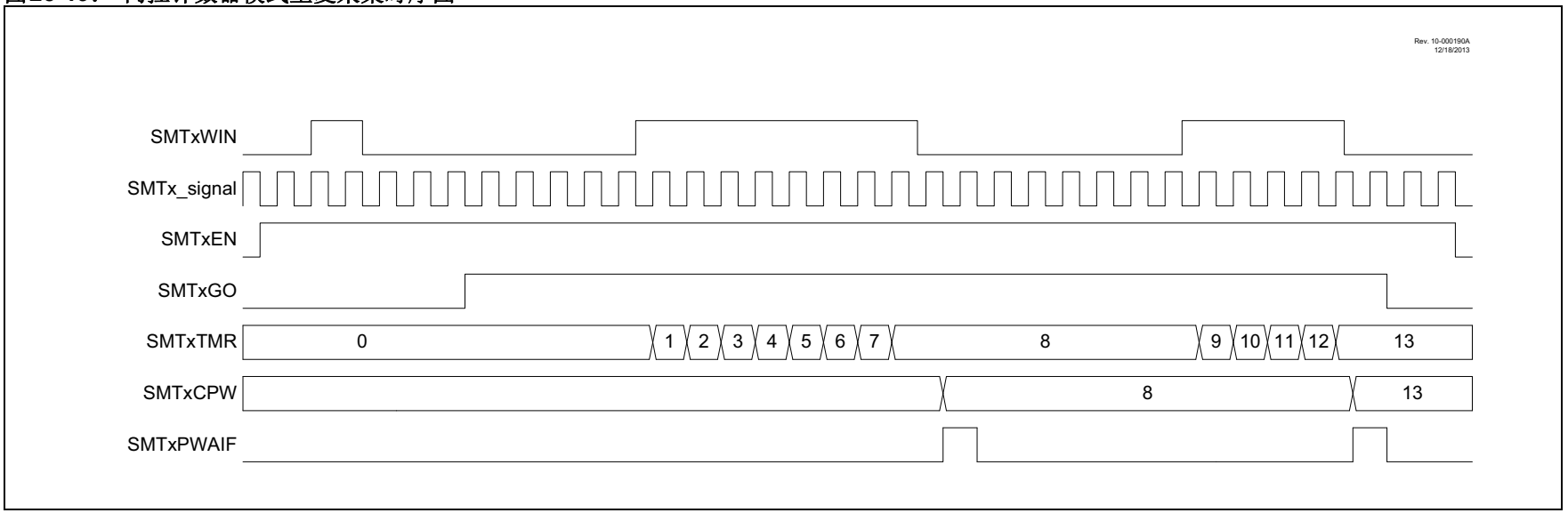
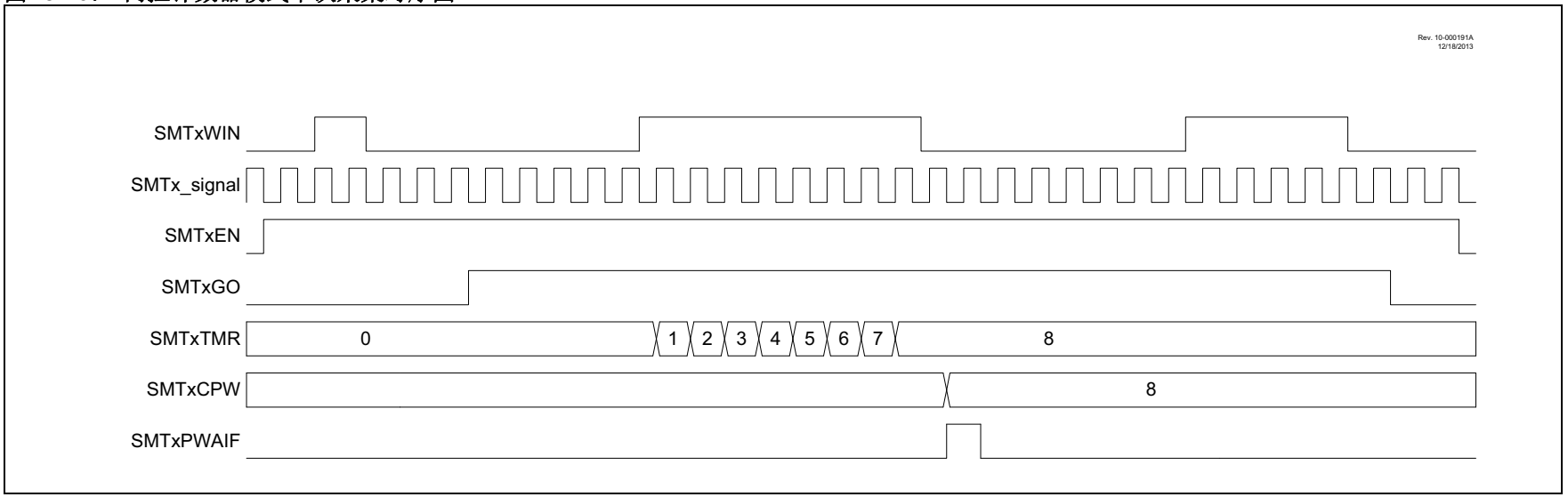


图25-20： 门控计数器模式单次采集时序图



25.6.11 窗口计数器模式

该模式在由SMTxWIN输入指定的窗口中，对SMTx_signal 输入上的脉冲进行计数。它在检测到SMTxWIN输入的上升沿时开始计数，在SMTxWIN输入的下降沿更新SMTxCPW寄存器，并在SMTxWIN输入第一个上升沿之外的每个上升沿更新SMTxCPR寄存器。请参见图25-21和图25-22。

图25-21： 窗口计数器模式重复采集时序图

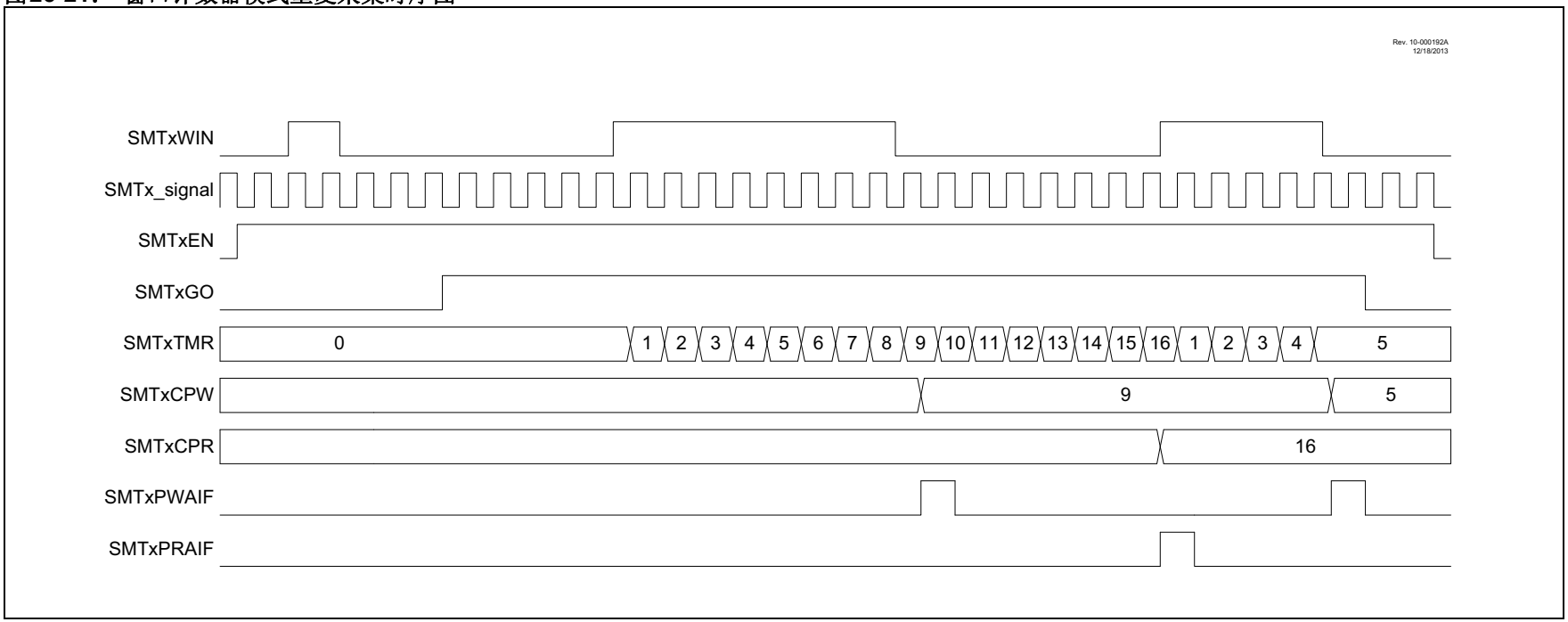
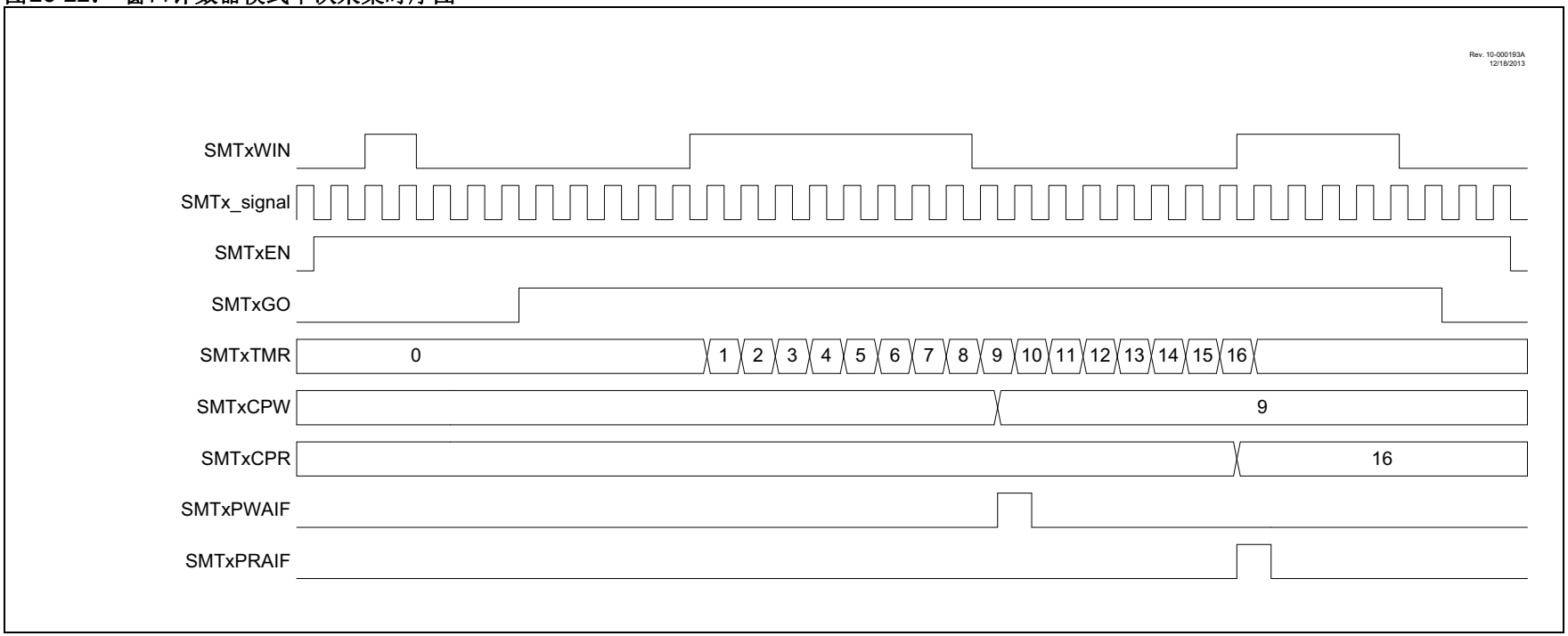


图25-22： 窗口计数器模式单次采集时序图



25.7 中断

SMT可以在3种不同条件下触发中断：

- PW采集完成
- PR采集完成
- 计数器周期匹配

中断由器件的PIR和PIE寄存器控制。

25.7.1 PW和PR采集中断

SMT可以在它每次更新SMTxCPW和SMTxCPR寄存器时触发中断，进行这些更新的条件取决于SMT模式，每种模式的具体章节中对此进行了介绍。SMTxCPW中断由SMTxPWAIF和SMTxPWAIE位控制，这两个位分别位于PIR和PIE寄存器中。SMTxCPR中断由SMTxPRAIF和SMTxPRAIE位控制，这两个位分别位于PIR和PIE寄存器中。

在同步SMT模式下，中断触发与SMTxCLK进行同步。在异步模式下，中断触发是异步的。在两种模式下，在触发之后，中断都会与CPU时钟进行同步。

25.7.2 计数器周期匹配中断

如**第25.1.2节“周期匹配中断”**所述，SMT还会在SMTxTMR与SMTxPR匹配时产生中断，其周期匹配限制功能如**第25.3节“暂停操作”**所述。周期匹配中断由SMTxIF和SMTxIE控制，这两个位分别位于PIR和PIE寄存器中。

25.8 寄存器定义：SMT控制

第1.3节“寄存器和位命名约定”给出了信号测量定时器外设的长位名称前缀。

表25-2： SMT外设的长位名称前缀

外设	位名称前缀
SMT1	SMT1
SMT2	SMT2

寄存器25-1：SMTxCON0：SMT控制寄存器0

R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN(1)	—	STP	WPOL	SPOL	CPOL	PS<1:0>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN**：SMT使能位(1)
1 = 使能SMT
0 = 禁止SMT；复位内部状态，禁止时钟请求

bit 6 **未实现**：读为0

bit 5 **STP**：SMT计数器暂停使能位
当SMTxTMR = SMTxPR时：
1 = 计数器保持为SMTxPR；在输入时钟时产生周期匹配中断
0 = 计数器复位为24’h000000；在输入时钟时产生周期匹配中断

bit 4 **WPOL**：SMTxWIN输入极性控制位
1 = SMTxWIN信号低电平有效/使能下降沿
0 = SMTxWIN信号高电平有效/使能上升沿

bit 3 **SPOL**：SMTxSIG输入极性控制位
1 = SMTx_signal低电平有效/使能下降沿
0 = SMTx_signal高电平有效/使能上升沿

bit 2 **CPOL**：SMT时钟输入极性控制位
1 = SMTxTMR在选定时钟信号的下降沿递增
0 = SMTxTMR在选定时钟信号的上升沿递增

bit 1-0 **PS<1:0>**：SMT预分频比选择位
11 = 预分频比 = 1:8
10 = 预分频比 = 1:4
01 = 预分频比 = 1:2
00 = 预分频比 = 1:1

注 1： 将EN设置为0不会影响寄存器的内容。

寄存器25-2：SMTxCON1：SMT控制寄存器1

R/W/HC-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
GO	REPEAT	—	—	MODE<3:0>			
bit 7							bit 0

图注：		
HC = 硬件清零位		HS = 硬件置1位
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **GO**：运行数据采集位
1 = 使能递增、数据采集
0 = 禁止递增、数据采集

bit 6 **REPEAT**：SMT重复采集使能位
1 = 使能重复数据采集模式
0 = 使能单次采集模式

bit 5-4 **未实现**：读为0

bit 3-0 **MODE<3:0>**：SMT工作模式选择位
1111 = 保留
•
•
•
1011 = 保留
1010 = 窗口计数器
1001 = 门控计数器
1000 = 计数器
0111 = 捕捉
0110 = 行程时间
0101 = 门控窗口测量
0100 = 窗口测量
0011 = 高电平和低电平时间测量
0010 = 周期和占空比采集
0001 = 门控定时器
0000 = 定时器

寄存器25-3：SMTxSTAT：SMT状态寄存器

R/W/HC-0/0	R/W/HC-0/0	R/W/HC-0/0	U-0	U-0	R-0/0	R-0/0	R-0/0
CPRUP	CPWUP	RST	—	—	TS	WS	AS
bit 7							bit 0

图注：		
HC = 硬件清零位		HS = 硬件置1位
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **CPRUP**：SMT手动周期缓冲区更新位
1 = 请求更新SMTxPRx寄存器
0 = SMTxPRx寄存器更新已完成

bit 6 **CPWUP**：SMT手动脉冲宽度缓冲区更新位
1 = 请求更新SMTxCPW寄存器
0 = SMTxCPW寄存器更新已完成

bit 5 **RST**：SMT手动定时器复位位
1 = 请求复位SMTxTMR寄存器
0 = SMTxTMR寄存器更新已完成

bit 4-3 **未实现**：读为0

bit 2 **TS**：运行值状态位
1 = SMT定时器正在递增
0 = SMT定时器不在递增

bit 1 **WS**：SMTxWIN值状态位
1 = SMT窗口已打开
0 = SMT窗口已关闭

bit 0 **AS**：SMT_signal值状态位
1 = 正在进行SMT采集
0 = 未进行SMT采集

寄存器25-4：SMTxCLK：SMT时钟选择寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	CSEL<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-3 **未实现**：读为0

bit 2-0 **CSEL<2:0>**：SMT时钟选择位

111 = 参考时钟输出
110 = SOSC
101 = FINTOSC/16（32 kHz）
100 = MFINTOSC（500 kHz）
011 = LFINTOSC
010 = HFINTOSC 16 MHz
001 = FOSC
000 = FOSC/4

寄存器25-5：SMTxWIN：SMTx窗口输入选择寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	WSEL<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-5 **未实现**：读为0

bit 4-0 **WSEL<4:0>**：SMTx窗口选择位

11111 = 保留
•
•
•
11011 = 保留
11010 = CLC4_out
11001 = CLC3_out
11000 = CLC2_out
10111 = CLC1_out
10110 = ZCD1_out
10101 = CMP2_out
10100 = CMP1_out
10011 = NCO1_out
10010 = 保留
10001 = 保留
10000 = PWM8_out
01111 = PWM7_out
01110 = PWM6_out
01101 = PWM5_out
01100 = CCP4_out
01011 = CCP3_out
01010 = CCP2_out
01001 = CCP1_out
01000 = TMR6_postscaled
00111 = TMR4_postscaled
00110 = TMR2_postscaled
00101 = TMR0_overflow
00100 = CLKREF
00011 = SOSC
00010 = MFINTOSC/16（32 kHz）
00001 = LFINTOSC
00000 = SMTxWINPPS

寄存器25-6：SMTxSIG：SMTx信号输入选择寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	SSEL<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-5 **未实现**：读为0

bit 4-0 **SSEL<4:0>**：SMTx信号选择位

11111 = 保留
•
•
•
11010 = CAN_rx_timestamp
11001 = CLC4_out
11000 = CLC3_out
10111 = CLC2_out
10110 = CLC1_out
10101 = ZCD1_out
10100 = CMP2_out
10011 = CMP1_out
10010 = NCO1_out
10001 = 保留
10000 = 保留
01111 = PWM8_out
01110 = PWM7_out
01101 = PWM6_out
01100 = PWM5_out
01011 = CCP4_out
01010 = CCP3_out
01001 = CCP2_out
01000 = CCP1_out
00111 = TMR6_postscaled
00110 = TMR5_overflow
00101 = TMR4_postscaled
00100 = TMR3_overflow
00011 = TMR2_postscaled
00010 = TMR1_overflow
00001 = TMR0_overflow
00000 = SMTxSIGPPS

寄存器25-7：SMTxTMRL：SMT定时器寄存器——低字节

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SMTxTMR<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxTMR<7:0>**：SMT计数器的有效位——低字节

寄存器25-8：SMTxTMRH：SMT定时器寄存器——高字节

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SMTxTMR<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxTMR<15:8>**：SMT计数器的有效位——高字节

寄存器25-9：SMTxTMRU：SMT定时器寄存器——高字节

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
SMTxTMR<23:16>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxTMR<23:16>**：SMT计数器的有效位——高字节

寄存器25-10：SMTxCPRL：SMT捕捉周期寄存器——低字节

R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x
SMTxCPR<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxCPR<7:0>**：SMT周期锁存器的有效位——低字节

寄存器25-11：SMTxCPRH：SMT捕捉周期寄存器——高字节

R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x
SMTxCPR<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxCPR<15:8>**：SMT周期锁存器的有效位——高字节

寄存器25-12：SMTxCPRU：SMT捕捉周期寄存器——高字节

R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x
SMTxCPR<23:16>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxCPR<23:16>**：SMT周期锁存器的有效位——高字节

寄存器25-13：SMTxCPWL：SMT捕捉脉冲宽度寄存器——低字节

R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x
SMTxCPW<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxCPW<7:0>**：SMT PW锁存器的有效位——低字节

寄存器25-14：SMTxCPWH：SMT捕捉脉冲宽度寄存器——高字节

R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x
SMTxCPW<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxCPW<15:8>**：SMT PW锁存器的有效位——高字节

寄存器25-15：SMTxCPWU：SMT捕捉脉冲宽度寄存器——高字节

R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x	R-x/x
SMTxCPW<23:16>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxCPW<23:16>**：SMT PW锁存器的有效位——高字节

寄存器25-16：SMTxPRL：SMT周期寄存器——低字节

R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1
SMTxPR<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxPR<7:0>**：周期匹配时的SMT定时器值的有效位——低字节

寄存器25-17：SMTxPRH：SMT周期寄存器——高字节

R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1
SMTxPR<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxPR<15:8>**：周期匹配时的SMT定时器值的有效位——高字节

寄存器25-18：SMTxPRU：SMT周期寄存器——高字节

R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1	R/W-x/1
SMTxPR<23:16>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **SMTxPR<23:16>**：周期匹配时的SMT定时器值的有效位——高字节

表25-3： 与SMTx相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
SMT1CON0	EN	—	STP	WPOL	SPOL	CPOL	SMT1PS<1:0>		380
SMT1CON1	GO	REPEAT	—	—	MODE<3:0>				381
SMT1STAT	CPRUP	CPWUP	RST	—	—	TS	WS	AS	382
SMT1CLK	—	—	—	—	—	CSEL<2:0>			383
SMT1SIG	—	—	—	SSEL<4:0>					385
SMT1WIN	—	—	—	WSEL<4:0>					384
SMT1TMRL	TMR<7:0>								386
SMT1TMRH	TMR<15:8>								386
SMT1TMRU	TMR<23:16>								386
SMT1CPRL	CPR<7:0>								387
SMT1CPRH	CPR<15:8>								387
SMT1CPRU	CPR<23:16>								387
SMT1CPWL	CPW<7:0>								388
SMT1CPWH	CPW<15:8>								388
SMT1CPWU	CPW<23:16>								388
SMT1PRL	PR<7:0>								389
SMT1PRH	PR<15:8>								389
SMT1PRU	PR<23:16>								389
SMT2CON0	EN	—	STP	WPOL	SPOL	CPOL	SMT2PS<1:0>		380
SMT2CON1	GO	REPEAT	—	—	MODE<3:0>				381
SMT2STAT	CPRUP	CPWUP	RST	—	—	TS	WS	AS	382
SMT2CLK	—	—	—	—	—	CSEL<2:0>			383
SMT2SIG	—	—	—	SSEL<4:0>					385
SMT2WIN	—	—	—	WSEL<4:0>					384
SMT2TMRL	TMR<7:0>								386
SMT2TMRH	TMR<15:8>								386
SMT2TMRU	TMR<23:16>								386
SMT2CPRL	CPR<7:0>								387
SMT2CPRH	CPR<15:8>								387
SMT2CPRU	CPR<23:16>								387
SMT2CPWL	CPW<7:0>								388
SMT2CPWH	CPW<15:8>								388
SMT2CPWU	CPW<23:16>								388
SMT2PRL	PR<7:0>								389
SMT2PRH	PR<15:8>								389
SMT2PRU	PR<23:16>								389

图注： – = 未实现，读为0。SMTx模块不使用阴影单元。

26.0 互补波形发生器（CWG）模块

互补波形发生器（CWG）可产生半桥、全桥和转向PWM波形。它与以前的CCP功能保持向后兼容。PIC18(L)F25/26K83系列具有3个CWG模块实例。

每个CWG模块都具有以下特性：

- 6种工作模式：
  - 同步转向模式
  - 异步转向模式
  - 全桥模式，正向
  - 全桥模式，反向
  - 半桥模式
  - 推挽模式
- 输出极性控制
- 输出转向
- 独立的6位上升沿和下降沿事件死区定时器
  - 时钟驱动死区
  - 独立的上升沿和下降沿死区使能
- 自动关断控制：
  - 可选关断源
  - 自动重启选项
  - 自动关断引脚改写控制

26.1 基本操作

CWG通过所选输入源生成两个输出波形。

每路输出由关到开的转变可能会因其他输出由开到关的转变而受到延时，因而在未驱动任何输出前会立即产生延时。这被称为死区，**第26.6节“死区控制”**对此进行了介绍。

可能需要防止电路发生故障、反馈事件太晚送达或根本不送达的可能性。在这种情况下，必须在故障条件造成损坏之前终止有效驱动。这称为自动关断，**第26.10节“自动关断”**将对此进行介绍。

26.2 工作模式

CWG模块可以在6种不同模式下工作，这些模式由CWGxCON0寄存器的MODE<2:0>位指定：

- 半桥模式
- 推挽模式
- 异步转向模式
- 同步转向模式
- 全桥模式，正向
- 全桥模式，反向

所有模式均接受单脉冲数据输入，并且提供最多4个输出，如以下小节所述。

所有模式均包含自动关断控制，如**第26.10节“自动关断”**所述。

注：	除全桥模式（如**第26.2.3节“全桥模式”**中所述）外，应仅在EN = 0（寄存器26-1）时进行模式更改。

26.2.1 半桥模式

在半桥模式下，将以输入的真值和反相形式生成两个输出信号，如图26-2所示。在两个输出之间插入不重叠（死区）时间，如**第26.6节“死区控制”**所述。该模式下无法使用输出转向功能。图26-1给出了该模式的基本框图。

未用输出CWGxC和CWGxD驱动与CWGxA和CWGxB类似的信号，它们的极性分别由CWGxCON1寄存器的POLC和POLD位独立控制。

图26-1： CWG简化框图（半桥模式，MODE<2:0> = 100）

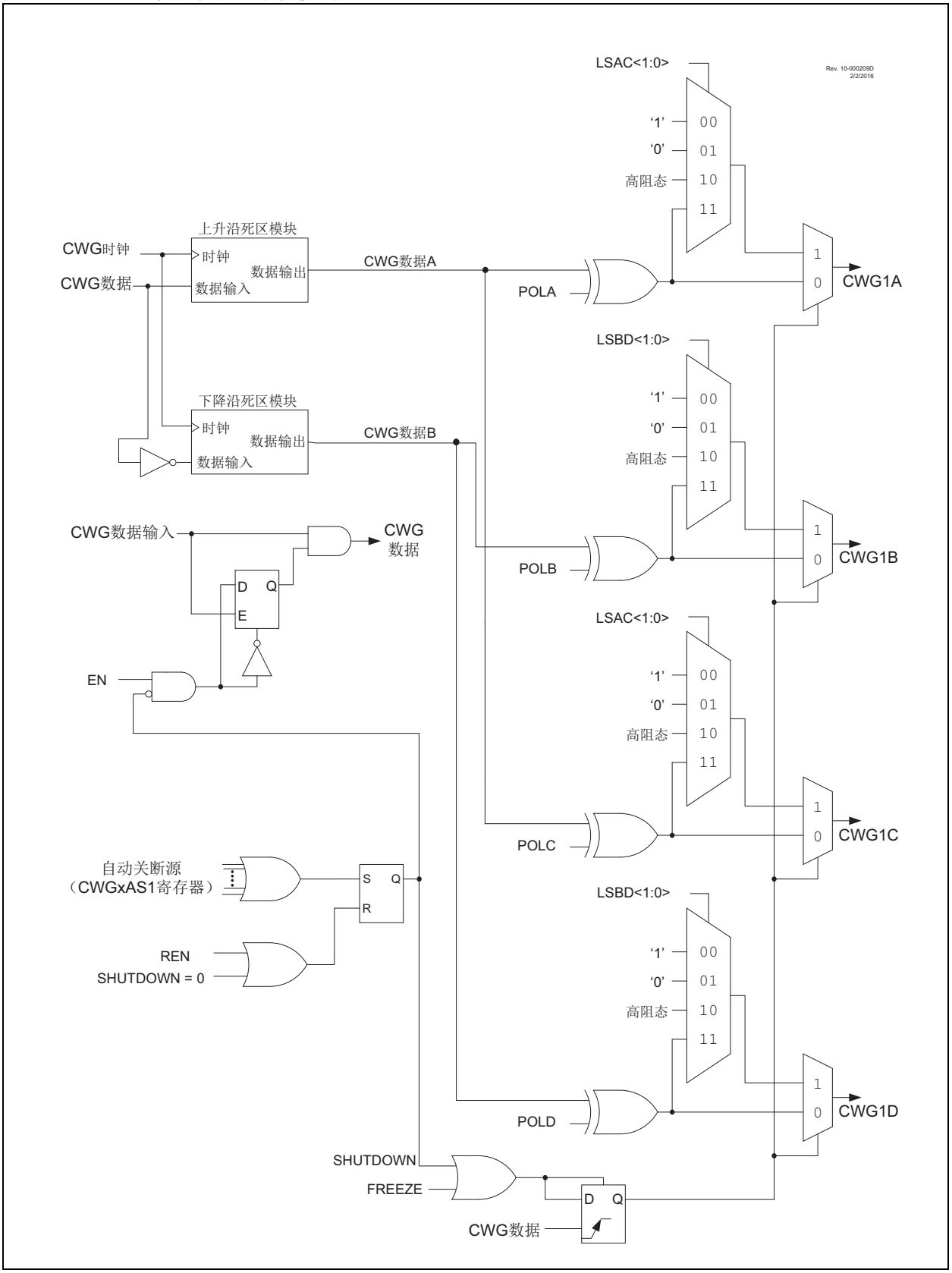
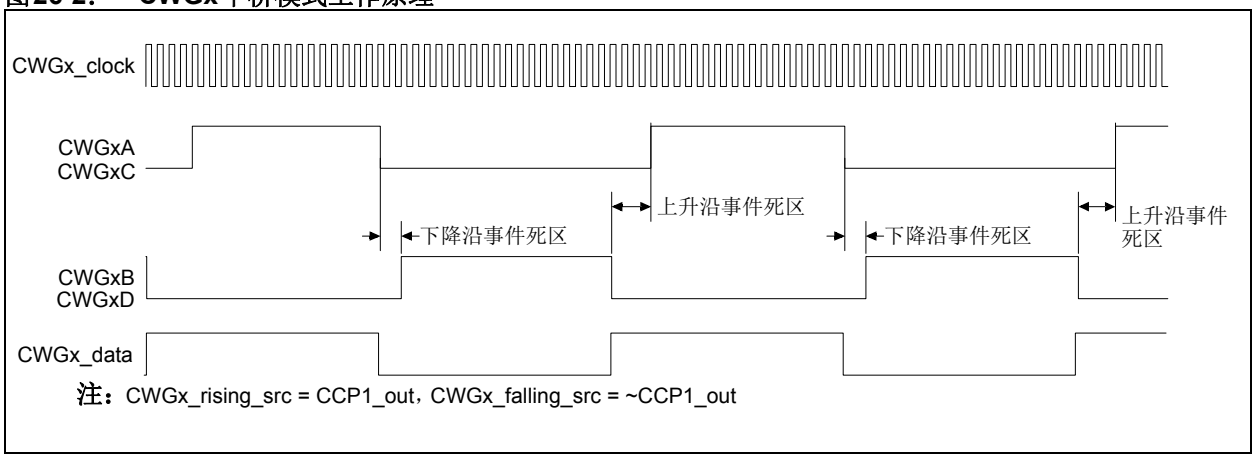


图26-2： CWGx半桥模式工作原理



26.2.2 推挽模式

在推挽模式下，将生成两个输出信号，它们为输入的交替副本，如图26-4所示。这种交替可以产生驱动一些基于变压器的电源设计所需的推挽效应。转向模式不用于推挽模式。图26-3给出了推挽模式下的基本框图。

推挽排序器在每次EN = 0或发生自动关断事件时复位。该排序器由第一个输入脉冲提供时钟，第一个输出出现在CWGxA上。

未用输出CWGxC和CWGxD分别驱动CWGxA和CWGxB的副本，但极性分别由CWGxCON1寄存器的POLC和POLD位控制。

图26-3： CWG简化框图（推挽模式，MODE<2:0> = 101）

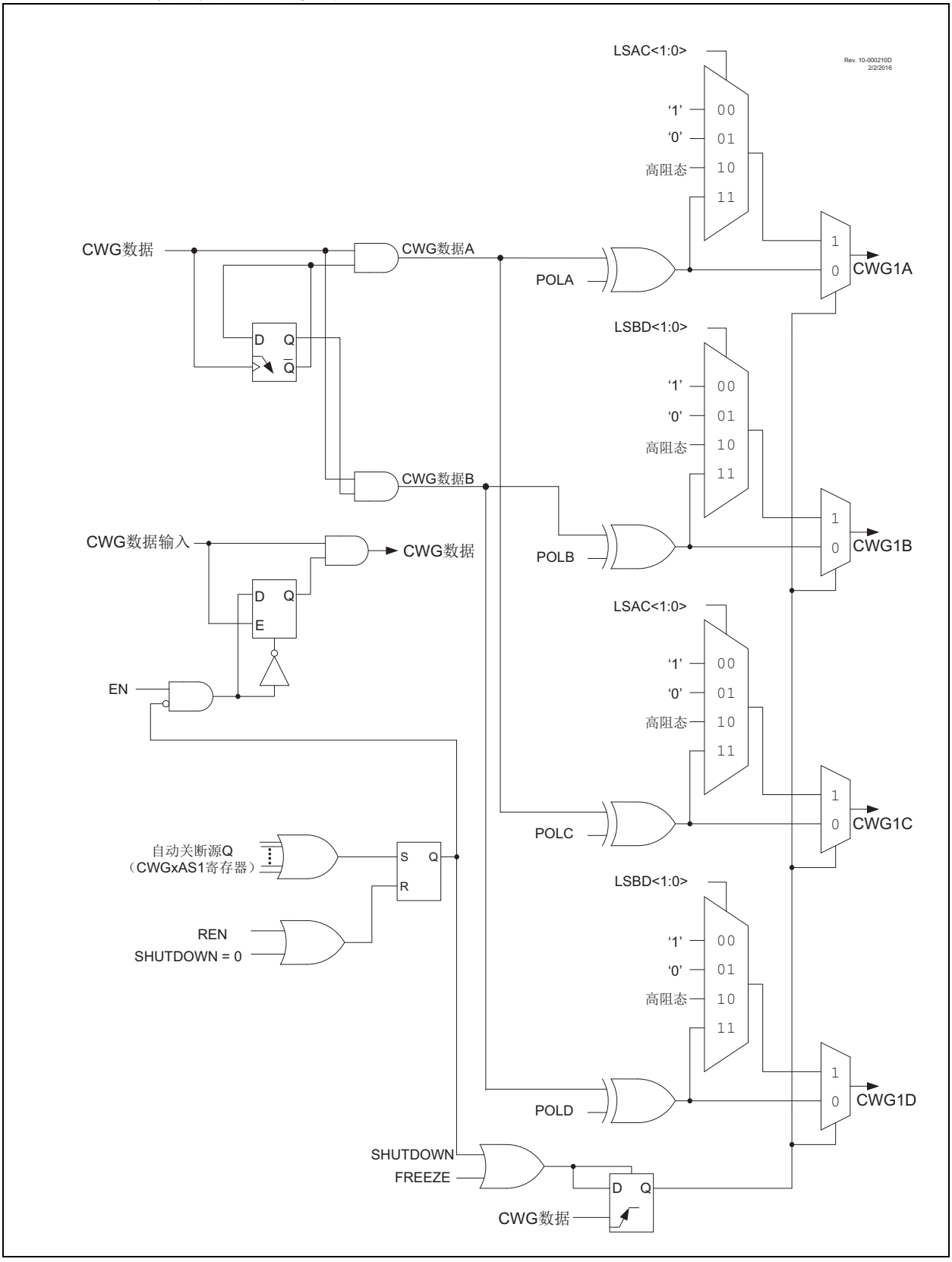
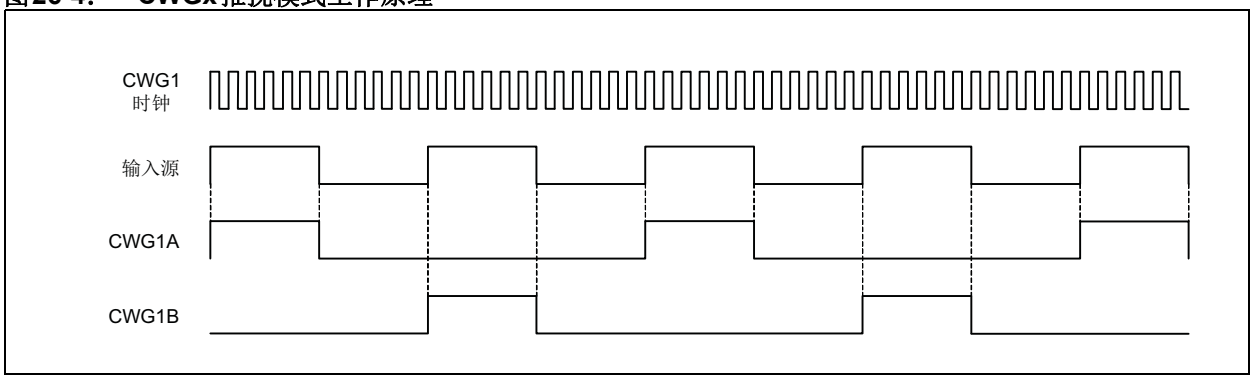


图26-4： CWGx推挽模式工作原理



26.2.3 全桥模式

在正向和反向全桥模式下，3个输出驱动静态值，第4个输出则通过输入数据信号进行调制。模式选择可以在正向和反向之间切换，方法是翻转CWGxCON0的MODE<0>位，同时将MODE<2:1>保持静态，而无需禁止CWG模块。当按图26-5所示连接时，输出适合全桥电机驱动器。每个CWG输出信号具有独立极性控制，因此该电路适合高电平有效和低电平有效驱动器。图26-6给出了全桥模式下的简化框图。

图26-5： 全桥应用示例

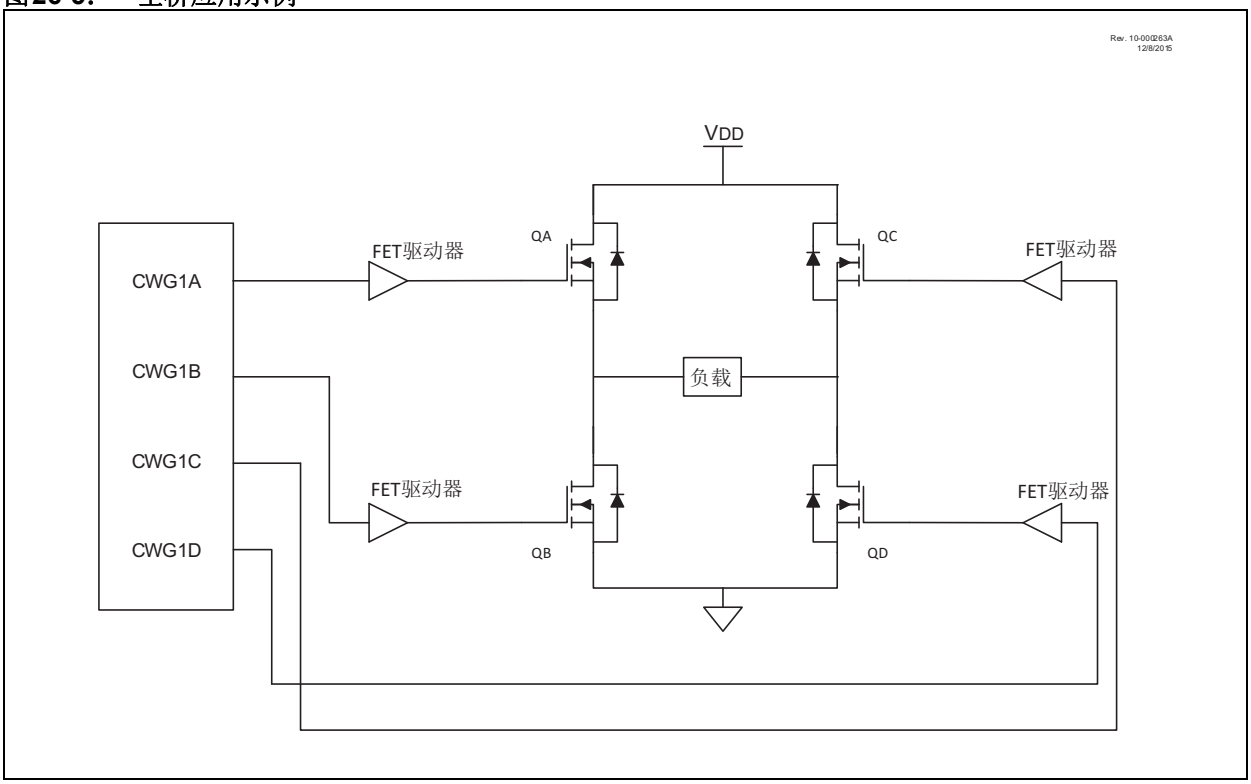
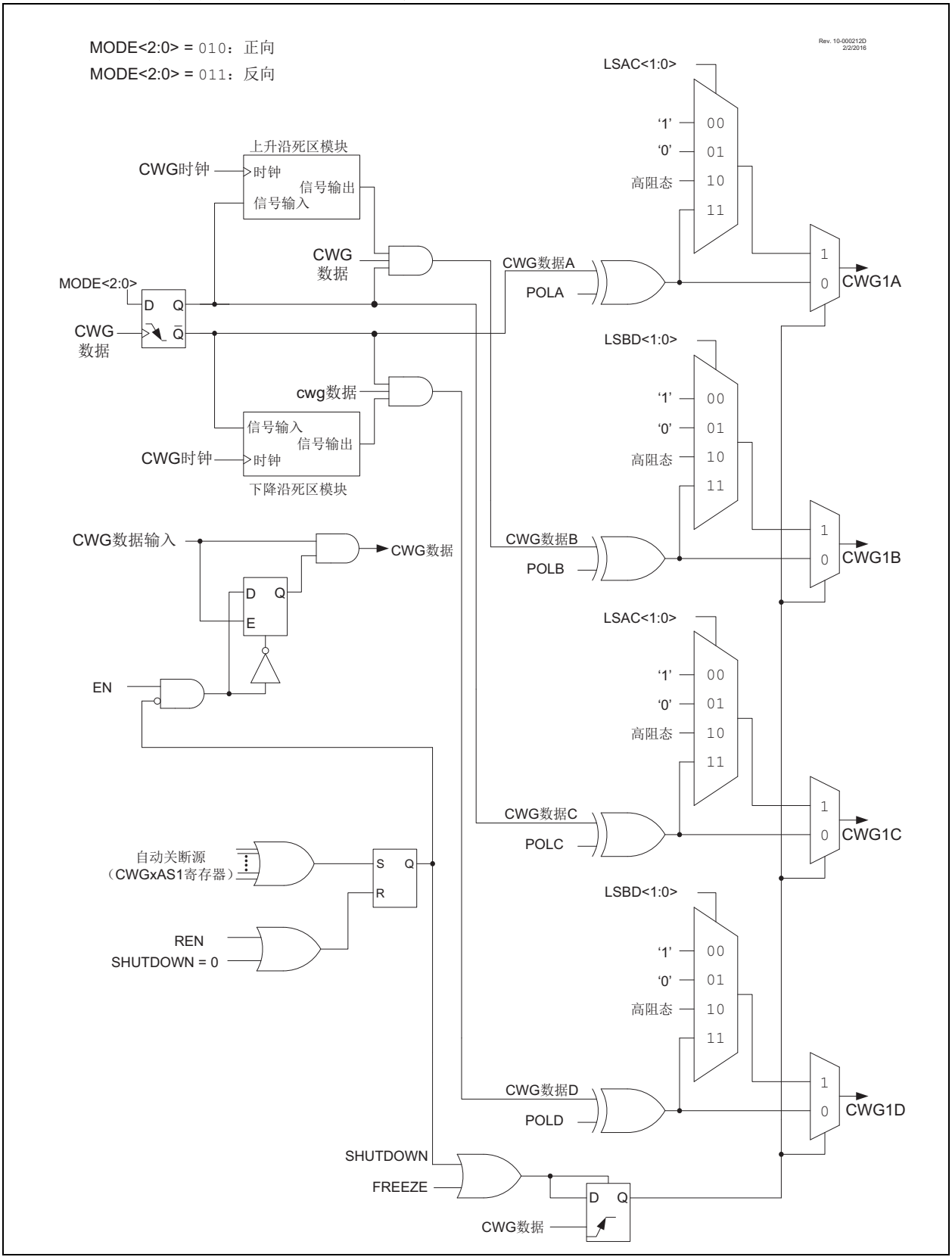


图26-6： CWG简化框图（正向和反向全桥模式）

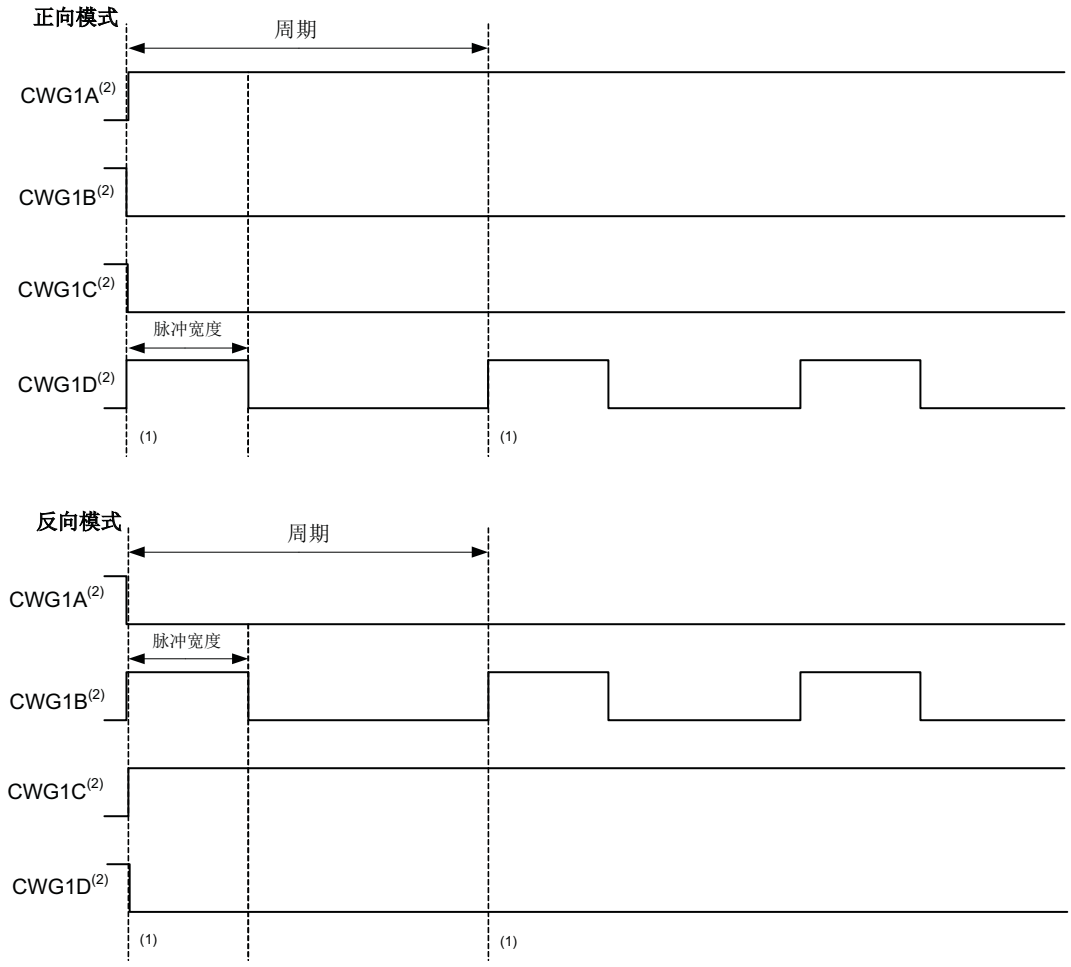


在正向全桥模式（MODE<2:0> = 010）下，CWGxA驱动为有效状态，CWGxB和CWGxC驱动为无效状态，CWGxD由输入信号进行调制，如图26-7所示。

在反向全桥模式（MODE<2:0> = 011）下，CWGxC驱动为有效状态，CWGxA和CWGxD驱动为无效状态，CWGxB由输入信号进行调制，如图26-7所示。

在全桥模式下，在从正向切换为反向（或反之）时会应用死区周期。**第26.6节“死区控制”**对这种死区控制进行了介绍，**第26.7节“上升沿和反向死区”**和**第26.8节“下降沿和正向死区”**提供了更多详细信息。转向模式不与任一全桥模式配合使用。模式选择可以在正向和反向之间切换，方法是翻转CWGxCON0的MODE<0>位，同时将MODE<2:1>保持静态，而无需禁止CWG模块。

图26-7： 全桥输出示例



注 1： 上升的CWG数据输入会在调制输出上创建上升沿事件。
2： 所示的输出信号为高电平有效；所有POLy位均清零。

26.2.3.1 全桥模式下的方向更改

在全桥模式下，更改MODE<2:0>控制正向/反向方向。对MODE<2:0>做出更改时，会在调制输入的下一个上升沿更改为新方向。

使用软件通过更改CWGxCON0寄存器的MODE<2:0>位的方式来启动方向更改。具体序列如图26-8所示。

- 相关的有效输出CWGxA和无效输出CWGxC切换为以相反的方向驱动。
- 先前调制输出CWGxD切换为无效状态，而先前无效输出CWGxB开始调制。
- 在经过方向切换死区后，恢复CWG调制。

26.2.3.2 全桥模式下的死区延时

当满足以下任一条件时，死区延时很重要：

1. 当数据输入的占空比接近或等于100%时，CWG输出的方向发生更改，或者
2. 功率开关（包括功率器件和驱动电路）的关断时间大于导通时间。

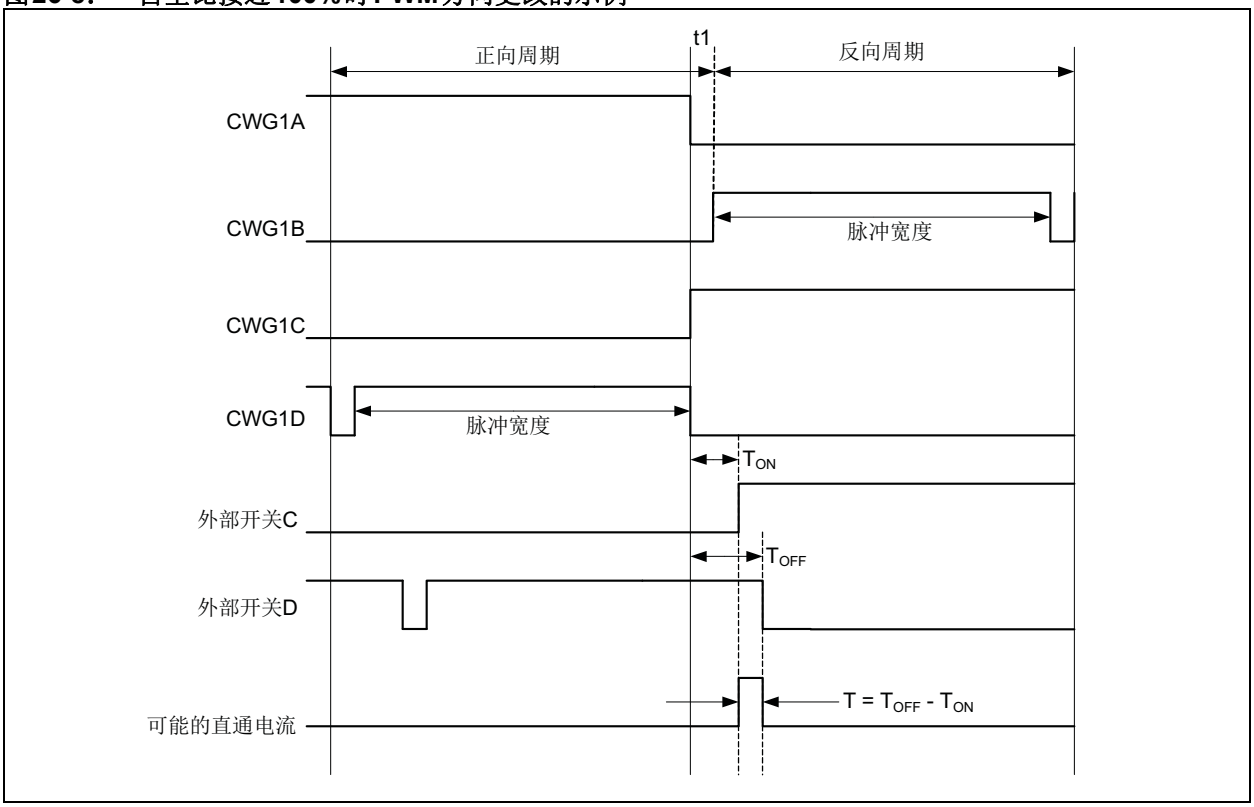
仅可在方向发生更改时插入死区延时，并且仅调制输出受到影响。静态配置输出（CWGxA和CWGxC）不提供死区，并且基本上同时开关。

图26-8给出了占空比接近100%时，CWG输出从正向变为反向的示例。在此示例中，在t1时刻CWGxA和CWGxD输出变为无效，而CWGxC输出变为有效。由于功率器件的关断时间比导通时间长，直通电流可能在时间段“t”内流过功率器件QC和QD。当CWG方向从反向变为正向时，功率器件QA和QB上也会发生同样的现象。

如果应用需要在高占空比时更改CWG方向，则有两种方法可以避免出现直通电流：

1. 在更改方向前的一个周期减小CWG的占空比。
2. 使用可使开关元件的关断速度比导通速度更快的开关驱动器。

图26-8： 占空比接近100%时PWM方向更改的示例



26.2.4 转向模式

在同步和异步转向模式下，可以将调制输入信号转向到4个CWG输出任意组合并在未用于PWM输出的所有输出上送出固定值。每个输出具有独立的极性、转向和关断选项。死区控制不用于任一转向模式。

当STRx = 0（寄存器26-5）时，相应引脚保持在OVRx（寄存器26-5）定义的电平。当STRx = 1时，引脚由调制输入信号驱动。

POLx位（寄存器26-2）仅在STRx = 1时控制信号极性。CWG自动关断工作原理也适用于转向模式，如**第26.14节“寄存器定义：CWG控制”**所述。

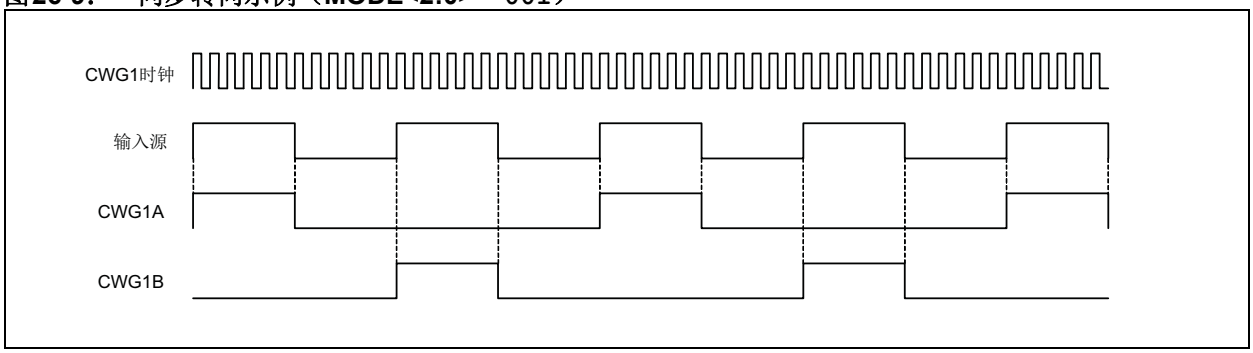
注：	仅STRx位同步；SDATx（数据）位不同步。

CWG自动关断操作也适用于转向模式，如**第26.10节“自动关断”**所述。自动关断事件只对STRx = 1的引脚有影响。

26.2.4.1 同步转向模式

在同步转向模式下（MODE<2:0>位 = 001，寄存器26-1），对转向选择寄存器的更改将在调制数据输入的下一个上升沿生效（图26-9）。在同步转向模式下，输出将始终产生完整波形。

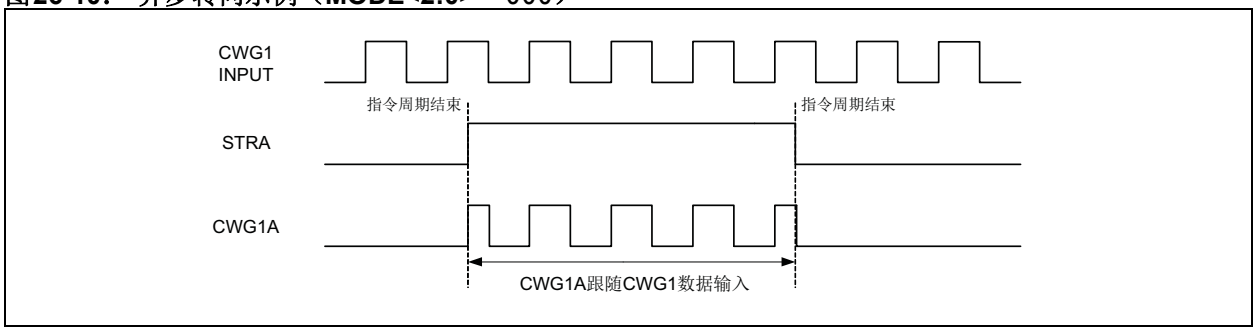
图26-9： 同步转向示例（MODE<2:0> = 001）



26.2.4.2 异步转向模式

在异步模式下（MODE<2:0>位 = 000，寄存器26-1），转向在写STR的指令周期结束时生效。在异步转向模式下，输出信号可能是不完整波形（图26-10）。此操作在用户固件需要立即除去该输出引脚的信号时非常有用。

图26-10： 异步转向示例（MODE<2:0> = 000）

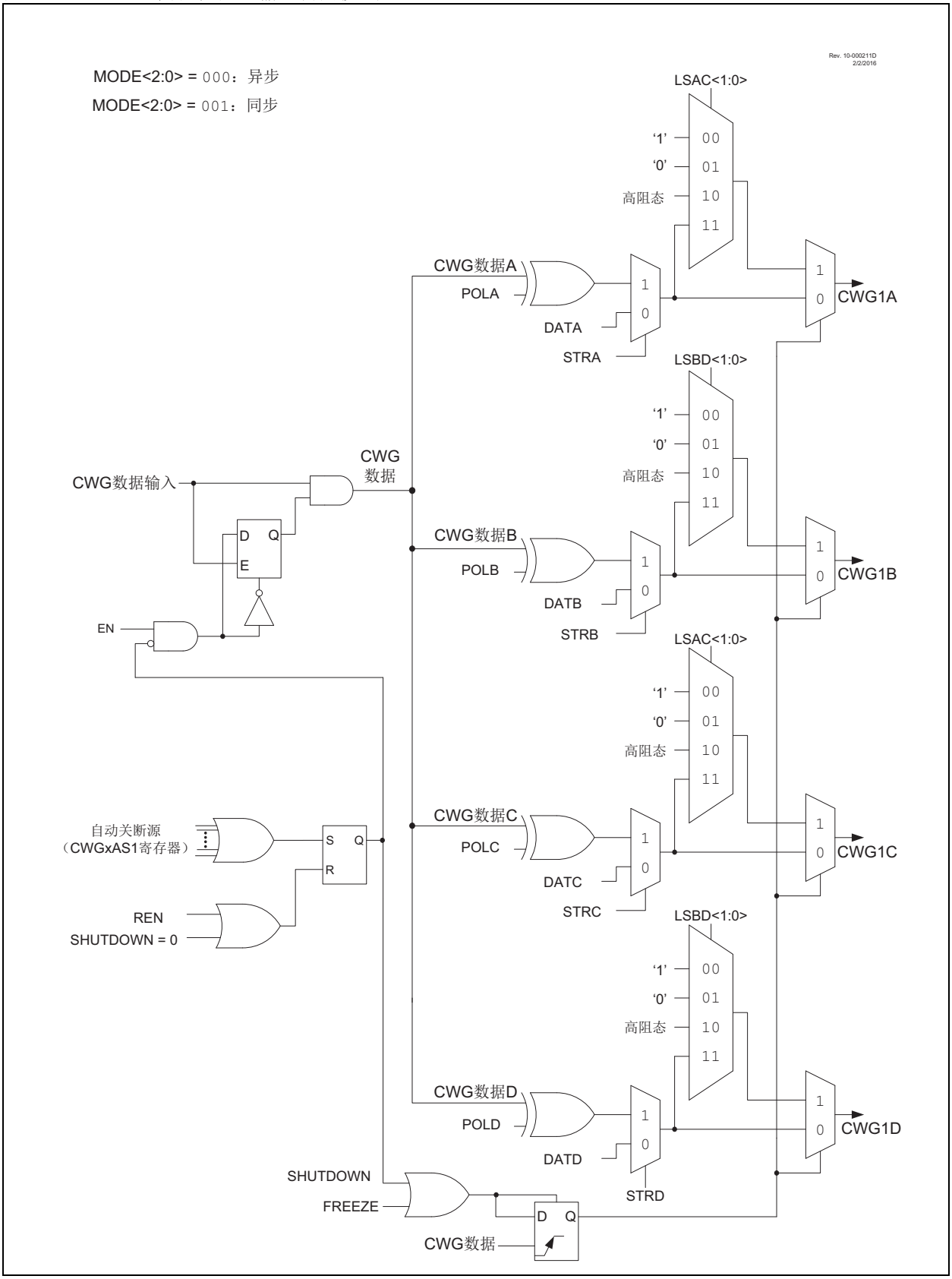


26.2.4.3 启动注意事项

应用硬件必须在CWG输出引脚上使用适当的外部上拉和 / 或下拉电阻。这是必需的，因为所有I/O引脚在复位时强制为高阻抗。

POLy位（寄存器26-2）允许用户选择输出信号是高电平有效还是低电平有效。

图26-11： CWG简化框图（输出转向模式）



26.3 时钟源

时钟源用于驱动死区时序电路。CWG模块允许选择以下时钟源：

- Fosc（系统时钟）
- HFINTOSC

选择HFINTOSC时，HFINTOSC将在休眠期间保持运行。因此，需要死区的CWG模式可在休眠模式下运行，前提是CWG数据输入也在休眠期间有效。时钟源使用CWGxCLKCON寄存器（寄存器26-3）的CS位进行选择。系统时钟Fosc在休眠状态下被禁止，因此无法使用死区控制。

26.4 可选输入源

CWG通过以下输入源生成输出波形：

表26-1： 可选输入源

源外设	信号名称	ISM<2:0>
CWGxPPS	通过CWGxPPS选择的引脚	000
CCP1	CCP1输出	001
CCP2	CCP2输出	010
PWM3	PWM3输出	011
PWM4	PWM4输出	100
CMP1	比较器1输出	101
CMP2	比较器2输出	110
DSM	数据信号调制器输出	111

使用CWGxISM寄存器（寄存器26-4）的IS<4:0>位选择输入源。

26.5 输出控制

26.5.1 CWG输出

每个CWG输出可通过RxyPPS寄存器输送到外设引脚选择（PPS）输出（见**第17.0节“外设引脚选择（PPS）模块”**）。

26.5.2 极性控制

每个CWG输出的极性可以单独进行选择。当输出极性位置1时，相应的输出为高电平有效。清零输出极性位时，相应输出将配置为低电平有效。但是，极性不会影响改写电平。输出极性使用CWGxCON1的POLy位进行选择。自动关断和转向选项不会受极性影响。

26.6 死区控制

死区控制用于提供不重叠的PWM信号，以防止PWM开关中产生直通电流。半桥和全桥模式会采用死区操作。CWG包含两个6位死区计数器。一个用于半桥模式下的输入源控制的上升沿，或用于全桥模式下的反向死区。另一个用于半桥模式下的输入源控制的下降沿，或用于全桥模式下的正向死区。

通过对CWG时钟周期计数来对死区进行计时，计数范围为零到上升沿或下降沿死区计数器寄存器中的值。请分别参见CWGxDBR和CWGxDBF寄存器。

26.6.1 半桥模式下的死区功能

在半桥模式下，死区计数器决定正常输出下降沿和反相输出上升沿之间的延时。图26-2中可以看到这一点。

26.6.2 全桥模式下的死区功能

在全桥模式下，死区计数器在方向改变时使用。CWGxCON0寄存器的MODE<0>位可以在CWG运行时置1或清零，从而可以从正向变为反向模式。CWGxA和CWGxC信号会在方向改变后出现第一个上升输入沿时立即改变，但调制信号（CWGxB或CWGxD，取决于改变的方向）会遇到由死区计数器决定的延时。

26.7 上升沿和反向死区

在半桥模式下，上升沿死区在CWG数据输入的上升沿之后延迟CWGxA输出的导通时间。在全桥模式下，仅在方向从正向模式变化为反向模式时才插入反向死区延时，并且仅调制输出CWGxB受到影响。

CWGxDBR寄存器用于确定输入源信号上升沿死区时间间隔的持续时间。该持续时间为0至64个CWG时钟周期。

死区总是在输入源信号的边沿启动。计数为0表示不存在死区。

如果输入源信号在死区计数完成之前翻转极性，则相应输出上不会出现任何信号。

CWGxDBR寄存器值是双重缓冲的。当EN = 0（寄存器26-1）时，会在写入CWGxDBR时装入缓冲区。当EN = 1时，在LD位（寄存器26-1）置1后将在数据输入的第一个下降沿之后的上升沿装入缓冲区。相关示例请参见图26-12。

26.8 下降沿和正向死区

在半桥模式下，下降沿死区在CWG数据输入的下降沿延迟CWGxB输出的导通时间。在全桥模式下，仅在方向从反向模式变化为正向模式时才插入正向死区延时，并且仅调制输出CWGxD受到影响。

CWGxDBF寄存器用于确定输入源信号下降沿死区时间间隔的持续时间。该持续时间为0至64个CWG时钟周期。

死区延时总是在输入源信号的边沿启动。计数为0表示不存在死区。

如果输入源信号在死区计数完成之前翻转极性，则相应输出上不会出现任何信号。

CWGxDBF寄存器值是双重缓冲的。当EN = 0（寄存器26-1）时，会在写入CWGxDBF时装入缓冲区。当EN = 1时，在LD位（寄存器26-1）置1后将在数据输入的第一个下降沿之后的上升沿装入缓冲区。相关示例请参见图26-13。

图26-12： 死区操作，CWGxDBR = 0x01，CWGxDBF = 0x02

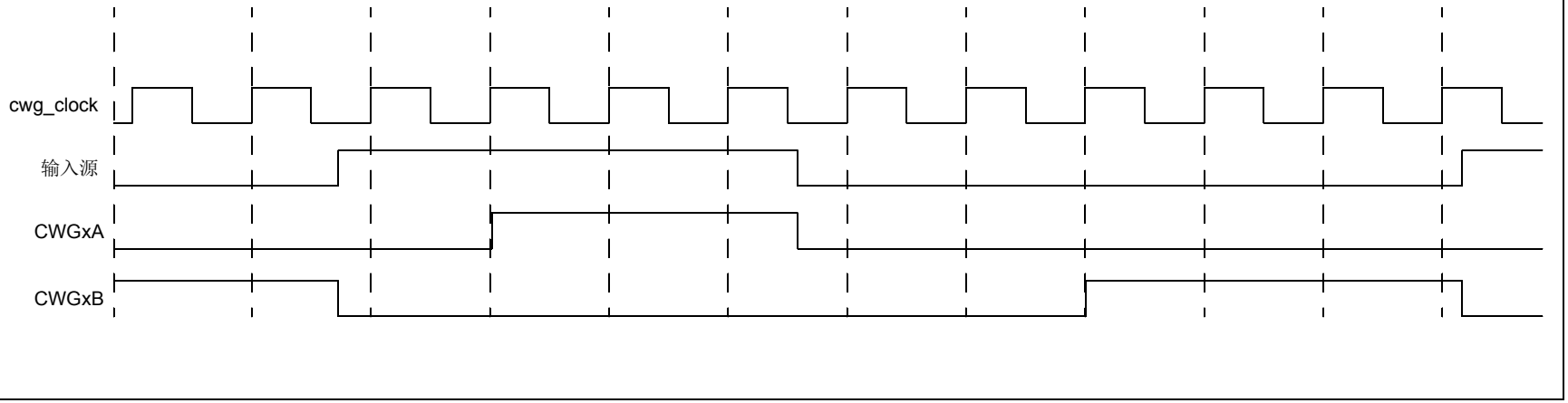
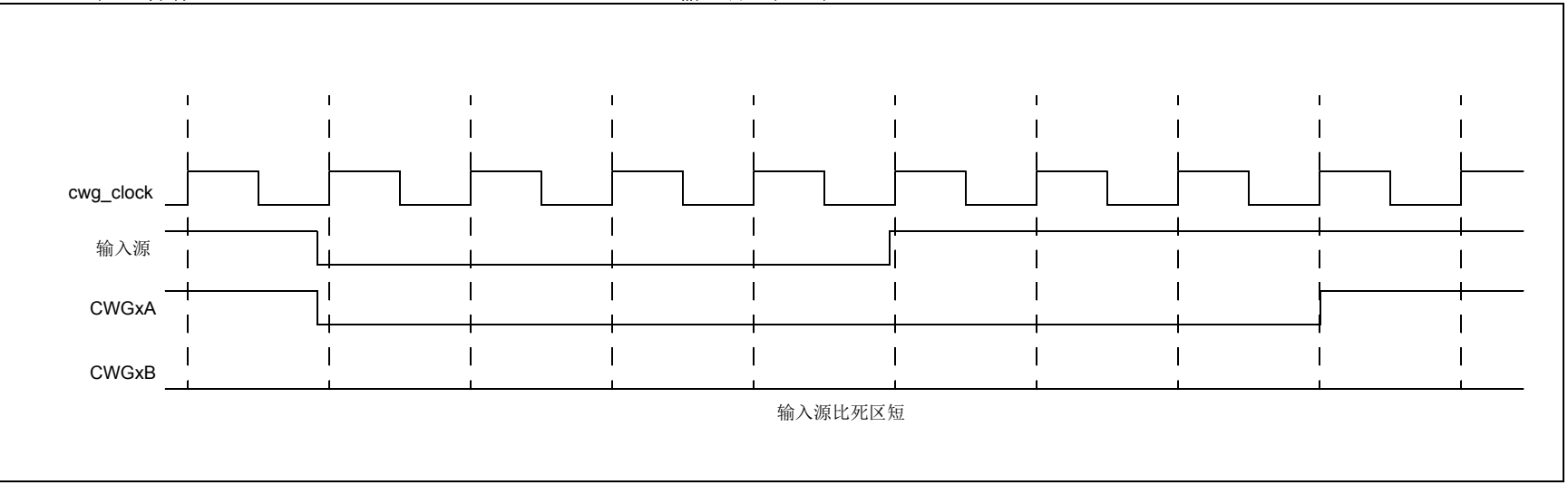


图26-13： 死区操作，CWGxDBR = 0x03，CWGxDBF = 0x06，输入源比死区短



26.9 死区抖动

当输入源的上升沿和下降沿与CWG时钟异步时，会在死区延时期间产生抖动。最大抖动等于1个CWG时钟周期。更多详细信息，请参见公式26-1。

公式26-1： 死区延时计算

$$T_{DEAD-BAND\_MIN} = \frac{1}{F_{CWG\ CLOCK}} \bullet DBx<4:0>$$

$$T_{DEAD-BANDMAX} = \frac{1}{F_{CWG\ CLOCK}} \bullet DBx<4:0>+1$$

$$T_{JITTER} = T_{DEAD-BAND\_MAX} - T_{DEAD-BAND\_MIN}$$

$$T_{JITTER} = \frac{1}{F_{CWG\_CLOCK}}$$

$$T_{DEAD-BAND\_MAX} = T_{DEAD-BAND\_MIN} + T_{JITTER}$$

示例

$$DBR<4:0> = 0x0A = 10$$

$$F_{CWG\_CLOCK} = 8\ MHz$$

$$T_{JITTER} = \frac{1}{8MHz} = 125\ ns$$

$$T_{DEAD-BAND\_MIN} = 125\ ns * 10 = 125\ \mu s$$

$$T_{DEAD-BAND\_MAX} = 1.25\ \mu s + 0.125 \mu s = 1.37 \mu s$$

26.10 自动关断

自动关断是一种使用特定改写信号立即改写CWG输出电平，从而安全关断电路的方法。关断状态可以自动清除，也可以一直保持，直到用软件清除。图26-14给出了自动关断电路的图示。

26.10.1 关断

关断状态可以通过以下两种方法之一进入：

- 软件生成
- 外部输入

26.10.1.1 软件生成的关断

将CWGxAS0寄存器的SHUTDOWN位置1可以强制CWG进入关断状态。

禁止自动重启后，只要SHUTDOWN位置1，关断状态就将持续。

在使能自动重启时，SHUTDOWN位会自动清零，并在发生下一个上升沿事件时继续工作。SHUTDOWN位指示何时存在关断条件。该位可由软件或硬件置1或清零。

26.10.1.2 外部输入源

外部关断输入提供了在出现故障条件时安全地暂停CWG工作的最快办法。当选定的任意关断输入变为有效时，CWG输出会立即变为指定的改写电平，无任何软件延时。改写电平通过CWGxAS0寄存器（寄存器26-6）的LSBD<1:0>和LSAC<1:0>位选择。可以选择几个输入源来产生关断条件。所有输入源均为低电平有效。这些输入源是：

- 通过CWGxPPS选择的引脚
- Timer2后分频输出
- Timer4后分频输出
- Timer6后分频输出
- 比较器1输出
- 比较器2输出
- CLC2输出

关断输入源通过CWGxAS1寄存器（寄存器26-7）的ASxE位单独使能。

注：	关断输入是电平敏感的，而不是边沿敏感的。只要关断输入电平仍然存在，除非禁止自动关断，否则无法清除关断状态。

26.10.1.3 引脚改写电平

在发生自动关断事件期间驱动到CWG输出的电平通过CWGxAS0寄存器（寄存器26-6）的LSBD<1:0>和LSAC<1:0>位进行控制。LSBD<1:0>位控制CWGxB/D输出电平，而LSAC<1:0>位控制CWGxA/C输出电平。

26.10.1.4 自动关断中断

当发生自动关断事件时，可通过软件或硬件将SHUTDOWN置1，以便将PIR寄存器的CWGxIF标志位置1。

26.11 自动关断重启

在发生自动关断事件之后，可以使用两种方法来恢复工作：

- 软件控制
- 自动重启

在任一情况下，在进行重启之前必须清除关断源。即，必须除去关断条件，或者必须清零相应的ASxE位。

26.11.1 软件控制重启

当CWGxAS0寄存器的REN位清零（REN = 0）时，CWG模块必须在自动关断事件后通过软件重启。

一旦除去所有自动关断源后，软件必须清零SHUTDOWN。SHUTDOWN清零后，CWG模块将在CWG数据输入的第一个上升沿继续工作。

注：	如果自动关断条件仍然存在，则无法用软件清零SHUTDOWN位。

26.11.2 自动重启

当CWGxAS0寄存器的REN位置1（REN = 1）时，CWG模块将从关断状态自动重启。

一旦除去所有自动关断条件后，硬件将自动清零SHUTDOWN。SHUTDOWN清零后，CWG模块将在CWG数据输入的第一个上升沿继续工作。

注：	如果自动关断条件仍然存在，则无法用软件清零SHUTDOWN位。

26.12 休眠期间的操作

CWG模块独立于系统时钟工作，只要选定的时钟源和输入源保持活动状态，它就会继续在休眠期间运行。

满足以下所有条件时，HFINTOSC会在休眠期间保持活动状态：

- 使能CWG模块
- 输入源为活动状态
- 无论选择的系统时钟源如何，都会选择HFINTOSC作为时钟源。

即，如果在CWG使能且输入源为活动状态时，同时选择HFINTOSC作为系统时钟和CWG时钟，则在休眠期间CPU会进入空闲状态，而HFINTOSC将保持活动状态，并且CWG会继续工作。这会直接影响休眠模式的电流。

26.13 配置CWG

1. 确保对应于CWG输出的TRIS控制位置1，从而将所有输出配置为输入，确保输出在设置期间无效。外部硬件应确保引脚电平保持为安全电平。
2. 清零EN位（如果尚未清零）。
3. 配置CWGxCON0寄存器的MODE<2:0>位来设置输出工作模式。
4. 配置CWGxCON1寄存器的POLy位来设置输出极性。
5. 配置CWGxISM寄存器的ISM<4:0>位来选择数据输入源。
6. 如果选择转向模式，则配置STRx位，以在CWG输出上选择所需输出。
7. 配置CWGxASD0寄存器的LSBD<1:0>和LSAC<1:0>位，以选择自动关断输出改写状态（即使未使用自动关断，这也是必需的，因为启动将从关断状态开始）。
8. 如果需要自动重启，则将CWGxAS0的REN位置1。
9. 如果需要自动关断，配置CWGxAS1寄存器的ASxE位来选择关断源。
10. 通过CWGxDBR和CWGxDBF寄存器设置所需上升和下降死区时间。
11. 在CWGxCLKCON寄存器中选择时钟源。
12. 将EN位置1，以使能模块。
13. 将对应于CWG输出的TRIS位清零，以将其设为输出。

如果要使用自动重启，则将REN位置1，SHUTDOWN位将会自动清零。否则，由软件清零SHUTDOWN位来启动CWG。

图26-14： CWG关断框图

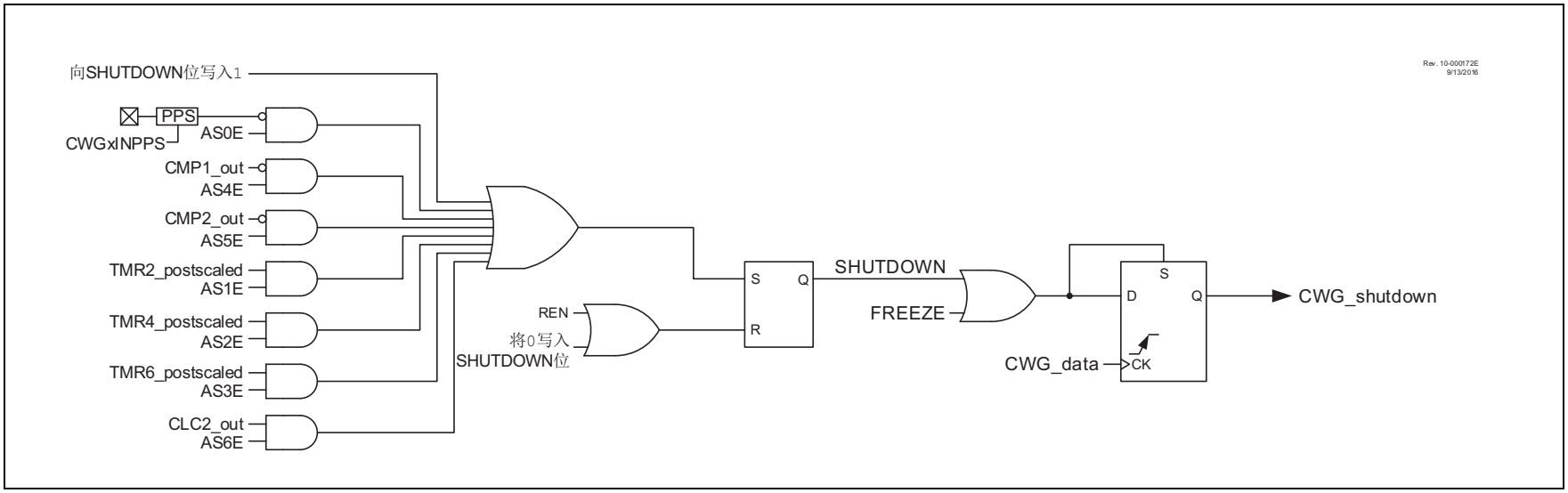
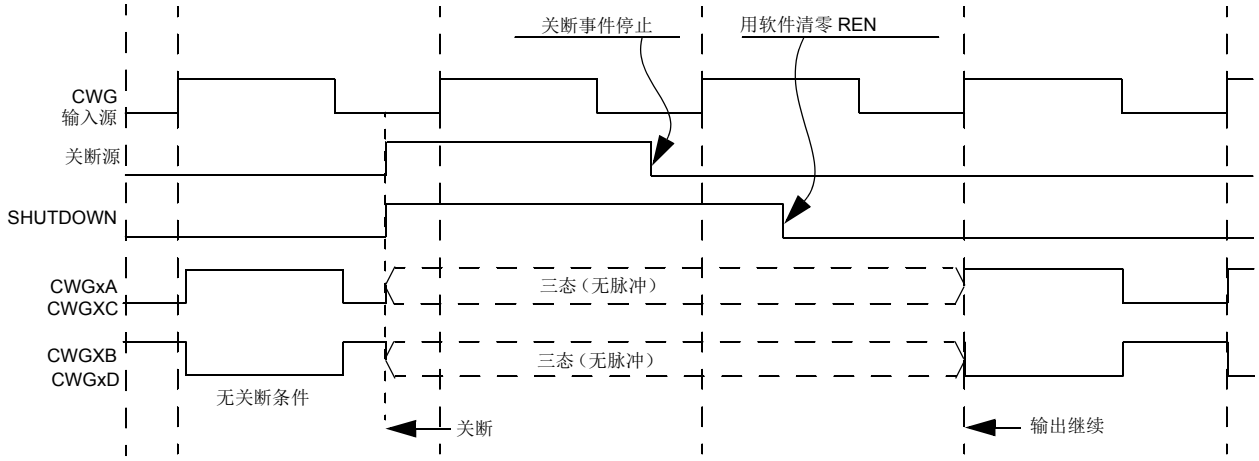


图26-15： 关断功能，禁止自动重启（REN = 0，LSAC = 01，LSBD = 01）



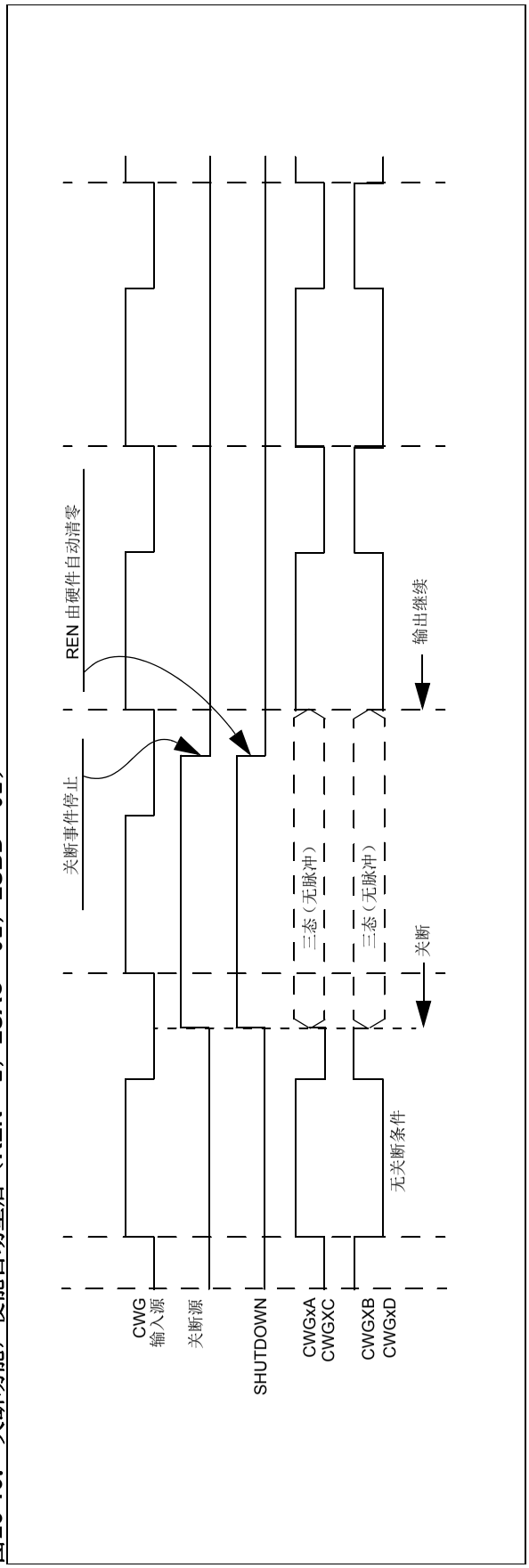


图 26-16：关断功能，使能自动重启（REN = 1，LSAC = 01，LSBD = 01）

26.14 寄存器定义：CWG控制

CWG外设的长位名称前缀如下所示。更多信息，请参见第1.3.2.2节“长位名称”。

外设	位名称前缀
CWG1	CWG1
CWG2	CWG2
CWG3	CWG3

寄存器26-1：CWGxCON0：CWG控制寄存器0

R/W-0/0	R/W/HC-0/0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
EN	LD(1)	—	—	—	MODE<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **EN**：CWGx使能位
1 = 使能模块
0 = 禁止模块

bit 6 **LD**：CWGx装入缓冲区位(1)
1 = 在紧接着该位置1后第一个下降沿的CWG数据上升沿装入死区计数缓冲区
0 = 缓冲区保持不变

bit 5-3 **未实现**：读为0

bit 2-0 **MODE<2:0>**：CWGx模式位
111 = 保留
110 = 保留
101 = CWG输出在推挽模式下工作
100 = CWG输出在半桥模式下工作
011 = CWG输出在反向全桥模式下工作
010 = CWG输出在正向全桥模式下工作
001 = CWG输出在同步转向模式下工作
000 = CWG输出在异步转向模式下工作

注 1： 该位只能在EN = 1后置1，不能在EN置1的同一周期中置1。

寄存器26-2：CWGxCON1：CWG控制寄存器1

U-0	U-0	R-x	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	IN	—	POLD	POLC	POLB	POLA
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-6 **未实现：**读为0

bit 5 **IN：**CWG输入值位（只读）

bit 4 **未实现：**读为0

bit 3 **POLD：**CWGxD输出极性位
1 = 信号输出极性翻转
0 = 信号输出为正常极性

bit 2 **POLC：**CWGxC输出极性位
1 = 信号输出极性翻转
0 = 信号输出为正常极性

bit 1 **POLB：**CWGxB输出极性位
1 = 信号输出极性翻转
0 = 信号输出为正常极性

bit 0 **POLA：**CWGxA输出极性位
1 = 信号输出极性翻转
0 = 信号输出为正常极性

寄存器26-3：CWGxCLK：CWGx时钟输入选择寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	CS
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-1 **未实现：**读为0

bit 0 **CS：**CWG时钟源选择位

CS	CWG1	CWG2	CWG3
1	HFINTOSC (1)	HFINTOSC (1)	HFINTOSC (1)
0	FOSC	FOSC	FOSC

注 1： HFINTOSC在休眠期间继续工作。

寄存器26-4：CWGxISM：CWGx输入选择寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	IS<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-5 **未实现**，读为0

bit 4-0 **IS<4:0>**：CWG数据输入选择多路开关选择位

IS<4:0>	CWG1	CWG2	CWG3
	输入选择	输入选择	输入选择
11111-10011	保留	保留	保留
10010	CLC4_out	CLC4_out	CLC4_out
10001	CLC3_out	CLC3_out	CLC3_out
10000	CLC2_out	CLC2_out	CLC2_out
01111	CLC1_out	CLC1_out	CLC1_out
01110	DSM_out	DSM_out	DSM_out
01101	CMP2OUT	CMP2OUT	CMP2OUT
01100	CMP1OUT	CMP1OUT	CMP1OUT
01011	NCO1OUT	NCO1OUT	NCO1OUT
01010-01001	保留	保留	保留
01000	PWM8OUT	PWM8OUT	PWM8OUT
00111	PWM7OUT	PWM7OUT	PWM7OUT
00110	PWM6OUT	PWM6OUT	PWM6OUT
00101	PWM5OUT	PWM5OUT	PWM5OUT
00100	CCP4_out	CCP4_out	CCP4_out
00011	CCP3_out	CCP3_out	CCP3_out
00010	CCP2_out	CCP2_out	CCP2_out
00001	CCP1_out	CCP1_out	CCP1_out
00000	通过CWG1PPS选择引脚	通过CWG2PPS选择引脚	通过CWG3PPS选择引脚

寄存器26-5：CWGxSTR(1)：CWG转向控制寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
OVRD	OVRC	OVRB	OVRA	STRD(2)	STRC(2)	STRB(2)	STRA(2)
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **OVRD**：转向数据D位

bit 6 **OVRC**：转向数据C位

bit 5 **OVRB**：转向数据B位

bit 4 **OVRA**：转向数据A位

bit 3 **STRD**：转向使能位D(2)
1 = CWGxD输出具有CWG数据输入波形，其极性由POLD位控制
0 = CWGxD输出指定为OVRD位的值

bit 2 **STRC**：转向使能位C(2)
1 = CWGxC输出具有CWG数据输入波形，其极性由POLC位控制
0 = CWGxC输出指定为OVRC位的值

bit 1 **STRB**：转向使能位B(2)
1 = CWGxB输出具有CWG数据输入波形，其极性由POLB位控制
0 = CWGxB输出指定为OVRB位的值

bit 0 **STRA**：转向使能位A(2)
1 = CWGxA输出具有CWG数据输入波形，其极性由POLA位控制
0 = CWGxA输出指定为OVRA位的值

注 1： 该寄存器中的位仅在MODE<2:0> = 00x（寄存器26-1，转向模式）时适用。
2： MODE<2:0> = 001时，该位是双重缓冲的。

寄存器26-6：CWGxAS0：CWG自动关断控制寄存器0

R/W/HS/HC-0/0	R/W-0/0	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	U-0	U-0
SHUTDOWN	REN	LSBD<1:0>		LSAC<1:0>		—	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS/HC = 由硬件置1/清零位
q = 值取决于具体条件		

bit 7 **SHUTDOWN**：自动关断事件状态位(1,2)
1 = 自动关断状态有效
0 = 未发生自动关断事件

bit 6 **REN**：自动重启使能位
1 = 使能自动重启
0 = 禁止自动重启

bit 5-4 **LSBD<1:0>**：CWGxB和CWGxD自动关断状态控制位
11 = 发生自动关断事件时，将逻辑1放置在CWGxB/D上。
10 = 发生自动关断事件时，将逻辑0放置在CWGxB/D上。
01 = 发生自动关断事件时，CWGxB/D上的引脚处于三态。
00 = 发生自动关断事件时，在所需的死区时间间隔之后，将引脚的无效状态（包括极性）放置在CWGxB/D上。

bit 3-2 **LSAC<1:0>**：CWGxA和CWGxC自动关断状态控制位
11 = 发生自动关断事件时，将逻辑1放置在CWGxA/C上。
10 = 发生自动关断事件时，将逻辑0放置在CWGxA/C上。
01 = 发生自动关断事件时，CWGxA/C上的引脚处于三态。
00 = 发生自动关断事件时，在所需的死区时间间隔之后，将引脚的无效状态（包括极性）放置在CWGxA/C上。

bit 1-0 **未实现**：读为0

注 1：在EN = 0（寄存器26-1）时，可以写入该位，将输出置为关断配置。
2：输出将一直保持在自动关断状态，直到该位清零后出现CWG数据输入的下一个上升沿为止。

寄存器26-7：CWGxAS1：CWG自动关断控制寄存器1

U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **未实现**，读为0

bit 6 **AS6E**：CWG自动关断源6使能位
1 = 使能源6的自动关断

CWG模块	CWG1	CWG2	CWG3
自动关断源6	CLC2 OUT	CLC3 OUT	CLC4 OUT

0 = 禁止源6的自动关断

bit 5 **AS5E**：CWG自动关断源5（CMP2 OUT）使能位
1 = 使能CMP2 OUT的自动关断
0 = 禁止CMP2 OUT的自动关断

bit 4 **AS4E**：CWG自动关断源4（CMP1 OUT）使能位
1 = 使能CMP1 OUT的自动关断
0 = 禁止CMP1 OUT的自动关断

bit 3 **AS3E**：CWG自动关断源3（TMR6_Postscaled）使能位
1 = 使能TMR6_Postscaled的自动关断
0 = 禁止TMR6_Postscaled的自动关断

bit 2 **AS2E**：CWG自动关断源2（TMR4_Postscaled）使能位
1 = 使能TMR4_Postscaled的自动关断
0 = 禁止TMR4_Postscaled的自动关断

bit 1 **AS1E**：CWG自动关断源1（TMR2_Postscaled）使能位
1 = 使能TMR2_Postscaled的自动关断
0 = 禁止TMR2_Postscaled的自动关断

bit 0 **AS0E**：CWG自动关断源0（通过CWGxPPS选择引脚）使能位
1 = 使能CWGxPPS引脚的自动关断
0 = 禁止CWGxPPS引脚的自动关断

寄存器26-8：CWGxDBR：CWG上升沿死区计数寄存器

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	DBR<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-6 **未实现：**读为0

bit 5-0 **DBR<5:0>：**CWG上升沿触发死区计数位

11 1111 = 63-64个CWG时钟周期
11 1110 = 62-63个CWG时钟周期
.
.
.
00 0010 = 2-3个CWG时钟周期
00 0001 = 1-2个CWG时钟周期
00 0000 = 0个CWG时钟周期。死区生成被旁路

寄存器26-9：CWGxDBF：CWG下降沿死区计数寄存器

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	DBF<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7-6 **未实现：**读为0

bit 5-0 **DBF<5:0>：**CWG下降沿触发死区计数位

11 1111 = 63-64个CWG时钟周期
11 1110 = 62-63个CWG时钟周期
.
.
.
00 0010 = 2-3个CWG时钟周期
00 0001 = 1-2个CWG时钟周期
00 0000 = 0个CWG时钟周期。死区生成被旁路。

表26-2： 与CWG相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
CWGxCON0	EN	LD	—	—	—	MODE<2:0>			410
CWGxCON1	—	—	IN	—	POLD	POLC	POLB	POLA	411
CWGxCLK	—	—	—	—	—	—	—	CS	412
CWGxISM	—	—	—	—	—	ISM<2:0>			413
CWGxSTR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	414
CWGxAS0	SHUTDOWN	REN	LSBD<1:0>		LSAC<1:0>		—	—	415
CWGxAS1	—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	416
CWGxDBR	—	—	DBR<5:0>						417
CWGxDBF	—	—	DBF<5:0>						417

图注： – = 未实现单元，读为0。CWG不使用阴影单元。

27.0 可配置逻辑单元（CLC）

可配置逻辑单元（CLCx）模块提供可超越软件执行速度限制而工作的可编程逻辑。该逻辑单元最多可接收32个输入信号，并通过使用可配置门将32个输入缩减为4条驱动8种可选单输出逻辑功能之一的逻辑线。

输入源是以下信号源的组合：

- I/O引脚
- 内部时钟
- 外设
- 寄存器位

可将输出内部连接到外设和输出引脚。

该器件提供了4个CLC模块，即CLC1、CLC2、CLC3和CLC4。

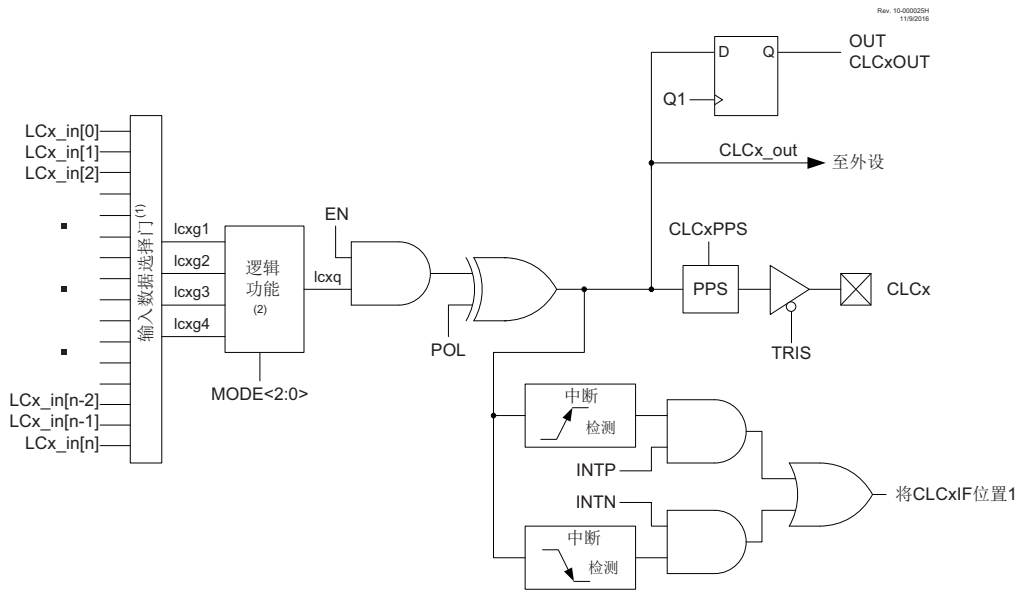
> **注：** CLC1、CLC2、CLC3和CLC4是相同CLC模块设计的4个独立模块实例。在本章中，寄存器名称中的小写字母“x”泛指CLC实例编号（在代码开发期间应用1、2、3或4来替换）。例如，控制寄存器在本章中统称为CLCxCON，但实际器件寄存器为CLC1CON、CLC2CON、CLC3CON和CLC4CON。

关于说明通过CLCx的信号流的简化框图，请参见图27-1。

可能的配置包括：

- 组合逻辑
  - AND
  - NAND
  - AND-OR
  - AND-OR-INVERT
  - OR-XOR
  - OR-XNOR
- 锁存器
  - S-R
  - 带置1和复位功能的时钟控制D型锁存器
  - 带置1和复位功能的透明D型锁存器

图27-1： CLCx简化框图



注 1： 请参见图27-2：输入数据选择和门。

2： 请参见图27-3：可编程逻辑功能。

27.1 CLCx设置

CLCx模块的编程通过配置逻辑信号流中的4级来实现。这4级为：

- 数据选择
- 数据门
- 逻辑功能选择
- 输出极性

每级都可在运行时通过写入相应的CLCx特殊功能寄存器来进行设置。这具有支持在程序执行期间运行时执行逻辑重新配置的额外优点。

27.1.1 数据选择

有32个信号可用作可配置逻辑的输入。使用4个32输入多路开关来选择要传递到下一级的输入。

数据选择通过图27-2左侧所示的4个多路开关来进行。图中的数据输入使用通用编号的输入名称来表示。

表27-1列出了每个CLC模块中的通用编号的输入名称与实际信号的关联。标有“DyS<4:0>值”的列给出了选定数据输入的多路开关选择代码。DyS是MUX选择输入代码的缩写：D1S<4:0>至D4S<4:0>。

数据输入使用CLCxSEL0至CLCxSEL3寄存器（寄存器27-3至寄存器27-6）进行选择。

注：	数据选择在上电时是未定义的。

表27-1： CLCx数据输入选择

DyS<5:0>值	CLCx输入源
111111 [63]	保留
•	
•	
•	
110110 [55]	
110110 [54]	CAN_tx1
110101 [53]	CAN_tx0
110100 [52]	CWG3B_out
110011 [51]	CWG3A_out
110010 [50]	CWG2B_out
110001 [49]	CWG2A_out
110000 [48]	CWG1B_out
101111 [47]	CWG1A_out
101110 [46]	SS1
101101 [45]	SCK1
101100 [44]	SDO1
101011 [43]	保留
101010 [42]	UART2_tx_out
101001 [41]	UART1_tx_out
101000 [40]	CLC4_out
100111 [39]	CLC3_out
100110 [38]	CLC2_out
100101 [37]	CLC1_out
100100 [36]	DSM1_out
100011 [35]	IOC_flag
100010 [34]	ZCD_out
100001 [33]	CMP2_out
100000 [32]	CMP1_out
011111 [31]	NCO1_out
011110 [30]	保留
011101 [29]	保留
011100 [28]	PWM8_out
011011 [27]	PWM7_out
011010 [26]	PWM6_out
011001 [25]	PWM5_out
011000 [24]	CCP4_out
010111 [23]	CCP3_out
010110 [22]	CCP2_out
010101 [21]	CCP1 _out
010100 [20]	SMT2_out
010011 [19]	SMT1_out
010010 [18]	TMR6_out

表27-1： CLCx数据输入选择（续）

DyS<5:0>值	CLCx输入源
010001 [17]	TMR5 _overflow
010000 [16]	TMR4 _out
001111 [15]	TMR3 _overflow
001110 [14]	TMR2 _out
001101 [13]	TMR1 _overflow
001100 [12]	TMR0 _overflow
001011 [11]	CLKR _out
001010 [10]	ADCRC
001001 [9]	SOSC
001000 [8]	MFINTOSC（32 kHz）
000111 [7]	MFINTOSC（500 kHz）
000110 [6]	LFINTOSC
000101 [5]	HFINTOSC
000100 [4]	FOSC
000011 [3]	CLCIN3PPS
000010 [2]	CLCIN2PPS
000001 [1]	CLCIN1PPS
000000 [0]	CLCIN0PPS

27.1.2 数据门

来自输入多路开关的输出将通过数据门级转送到所需的逻辑功能输入。每个数据门可以转送由4个选定输入组成的任意组合。

注： 在上电时，数据门是未定义的。

门级不仅仅是信号方向。可将门配置为将每个输入信号转换为反相或同相数据。在每个门中，将转换后的信号进行逻辑与。每个门的输出可以先进行反相，然后再进入逻辑功能级。

门控实际上是一个1至4的输入AND/NAND/OR/NOR门。如果将每个输入和输出进行反相，则该门的作用是对所有已使能数据输入进行逻辑或。如果输入和输出不进行反相，则该门的作用是对所有已使能输入进行逻辑与。

表27-2总结了可以通过使用门逻辑选择位在门1中获得的基本逻辑。该表列出了具有4个输入变量的逻辑，但每个门可以配置为使用少于4个输入。如果未选择任何输入，则输出将为0或1，具体取决于门输出极性位。

表27-2： 数据门逻辑

CLCxGLSy	GyPOL	门逻辑
0x55	1	AND
0x55	0	NAND
0xAA	1	NOR
0xAA	0	OR
0x00	0	逻辑0
0x00	1	逻辑1

用户可以（但建议不要）同时选择同一输入的正负值。如果这么做，则无论其他输入如何，门的输出都将为0，但可能会出现逻辑毛刺（瞬变引起的脉冲）。如果通道的输出必须为0或1，则建议的方法是将所有门位设置为0，并使用门极性位来设置所需的电平。

数据门使用如下逻辑门选择寄存器进行配置：

- 门1：CLCxGLS0（寄存器27-7）
- 门2：CLCxGLS1（寄存器27-8）
- 门3：CLCxGLS2（寄存器27-9）
- 门4：CLCxGLS3（寄存器27-10）

寄存器编号后缀不同于门编号，这是因为该模块的其他形式在同一寄存器中具有多种门选择。

图27-2右侧给出了数据门的图示。其中仅详细说明了一个门。其余三个门使用相同的配置，只是数据使能对应于该门的使能信号。

27.1.3 逻辑功能

有8种可用的逻辑功能，包括：

- AND-OR
- OR-XOR
- AND
- S-R锁存器
- 带置1和复位功能的D型触发器
- 带复位功能的D型触发器
- 带复位功能的J-K型触发器
- 带置1和复位功能的透明锁存器

这些逻辑功能如图27-2所示。每种逻辑功能具有4个输入和1个输出。4个输入是上一级的4个数据门输出。输出送到反相级，接着送到其他外设和输出引脚，然后回到CLCx。

27.1.4 输出极性

可配置逻辑单元中的最后一级是输出极性。将CLCxPOL寄存器的POL位置1时，来自逻辑级的输出信号会进行反相。如果在允许中断时改变极性，则会导致输出结果的变化发生中断。

27.2 CLCx中断

如果相应的中断允许位置1，则在CLCx的输出值改变时，将会产生中断。因此，每个CLC中都具有一个上升沿检测器和一个下降沿检测器。

触发其中一个边沿检测器，且其相关的中断允许位置1时，相关PIR5寄存器的CLCxIF位会置1。INTP位用于允许上升沿中断，INTN位用于允许下降沿中断。它们都位于CLCxCON寄存器中。

要完全允许中断，需要将以下位置1：

- 相应PIE寄存器的CLCxIE位
- CLCxCON寄存器的INTP位（对于上升沿检测）
- CLCxCON寄存器的INTN位（用于下降沿检测）
- INTCON0寄存器的GIE位

作为中断服务的一部分，必须用软件将相应PIR寄存器的CLCxIF位清零。如果在清零该标志时检测到另一个边沿，则标志仍然会在序列结束时置1。

27.3 输出镜像副本

所有CON输出位的镜像副本包含在CLCxDATA寄存器中。读取该寄存器将同时读取所有CLC的输出。这可以防止由于测试或读取各个CLCxCON寄存器中的OUT位而导致读取差错。

27.4 复位的影响

发生复位后，CLCxCON寄存器会清零。所有其他选择和门值保持不变。

27.5 休眠期间的操作

CLC模块独立于系统时钟工作，只要选定的输入源保持活动状态，它就会继续在休眠期间运行。

如果使能了CLC模块，并且选择HFINTOSC作为输入源，则无论所选择的系统时钟源如何，HFINTOSC都会在休眠期间保持活动状态。

即，如果在CLC使能时，同时选择HFINTOSC作为系统时钟和CLC输入源，则在休眠期间CPU会进入空闲状态，而CLC会继续工作，并且HFINTOSC将保持活动状态。

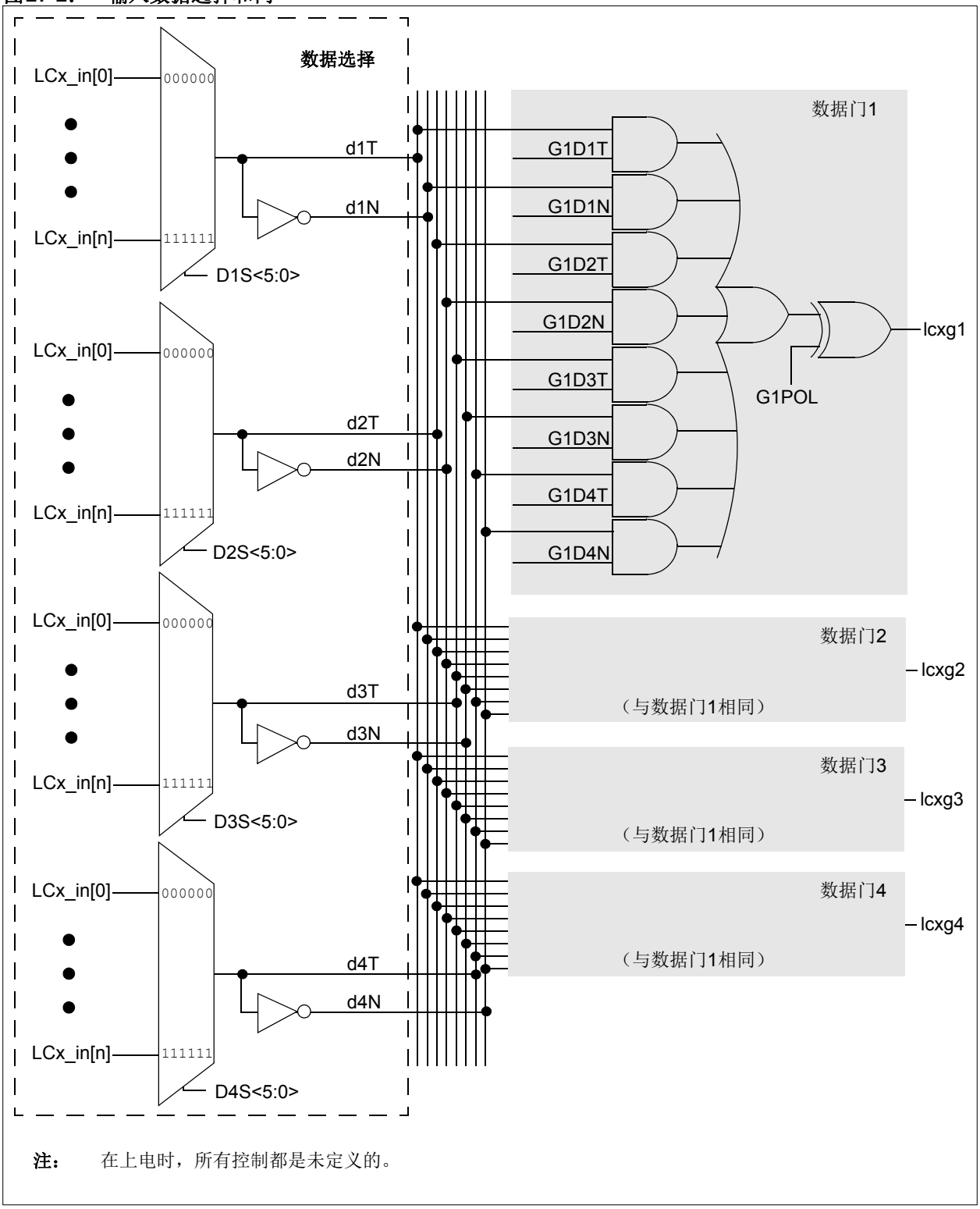
这会直接影响休眠模式的电流。

27.6 CLCx设置步骤

在设置CLCx时，应遵循以下步骤：

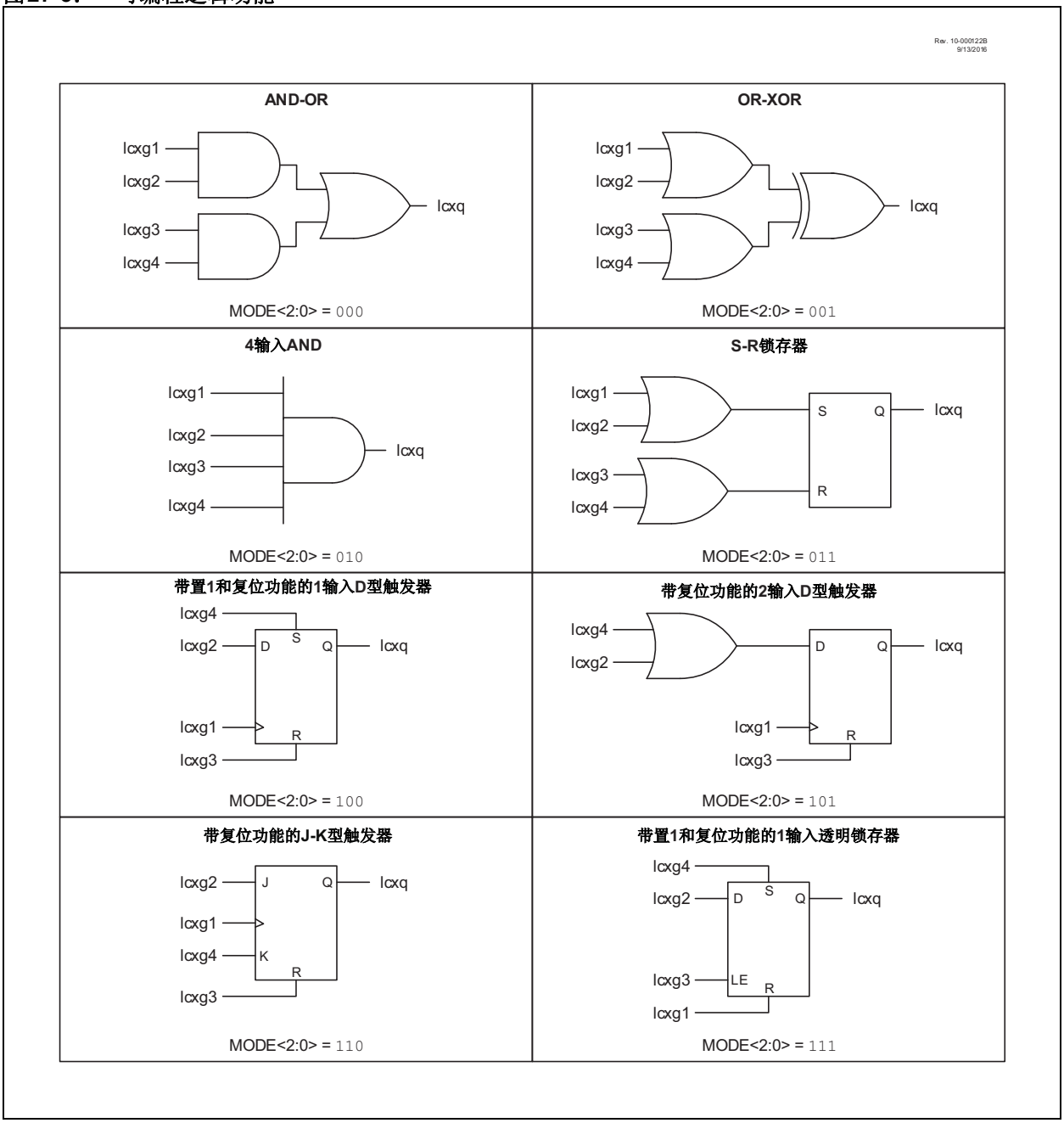
- 通过清零EN位来禁止CLCx。
- 使用CLCxSEL0至CLCxSEL3寄存器选择所需的输入（见表27-1）。
- 清零所有相关的ANSEL位。
- 将与输入相关的所有TRIS位置1。
- 将与输出相关的所有TRIS位清零。
- 使用CLCxGLS0、CLCxGLS1、CLCxGLS2和CLCxGLS3寄存器使能通过4个门的可选输入。
- 使用CLCxPOL寄存器的GyPOL位选择门输出极性。
- 使用CLCxCON寄存器的MODE<2:0>位选择所需的逻辑功能。
- 使用CLCxPOL寄存器的POL位选择所需的逻辑输出极性。（该步骤可与前面的门输出极性步骤结合）。
- 如果驱动某个器件引脚，则设置所需的引脚PPS控制寄存器，并另外清零对应于该输出的TRIS位。
- 如果需要中断，则配置以下位：
  - 上升沿事件时，将CLCxCON寄存器中的INTP位置1。
  - 下降沿事件时，将CLCxCON寄存器中的INTN位置1。
  - 将相应PIE寄存器的CLCxIE位置1。
  - 将INTCON0寄存器的GIE位置1。
- 通过将CLCxCON寄存器的EN位置1来使能CLCx。

图27-2： 输入数据选择和门



注： 在上电时，所有控制都是未定义的。

图27-3： 可编程逻辑功能



27.7 寄存器定义：CLC控制

寄存器27-1： CLCxCON：可配置逻辑单元控制寄存器

R/W-0/0	U-0	R-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN	—	OUT	INTP	INTN	MODE<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN**：可配置逻辑单元使能位
1 = 使能可配置逻辑单元，并混合输入信号
0 = 禁止可配置逻辑单元，并输出逻辑0

bit 6 **未实现**：读为0

bit 5 **OUT**：可配置逻辑单元数据输出位
只读：经过LCPOL之后的逻辑单元输出数据；从CLCxOUT采样

bit 4 **INTP**：可配置逻辑单元上升边沿中断允许位
1 = CLCxIF将在CLCxOUT上出现上升沿时置1
0 = CLCxIF不会置1

bit 3 **INTN**：可配置逻辑单元下降边沿中断允许位
1 = CLCxIF将在CLCxOUT上出现下降沿时置1
0 = CLCxIF不会置1

bit 2-0 **MODE<2:0>**：可配置逻辑单元功能模式位
111 = 单元是带置1和复位功能的1输入透明锁存器
110 = 单元是带复位功能的J-K型触发器
101 = 单元是带复位功能的2输入D型触发器
100 = 单元是带置1和复位功能的1输入D型触发器
011 = 单元是S-R型锁存器
010 = 单元是4输入AND逻辑
001 = 单元是OR-XOR逻辑
000 = 单元是AND-OR逻辑

寄存器27-2： CLCxPOL：信号极性控制寄存器

R/W-0/0	U-0	U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
POL	—	—	—	G4POL	G3POL	G2POL	G1POL
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **POL**：CLCxOUT输出极性控制位
1 = 逻辑单元的输出反相
0 = 逻辑单元的输出同相

bit 6-4 **未实现**：读为0

bit 3 **G4POL**：门3输出极性控制位
1 = 门3的输出在施加到逻辑单元时反相
0 = 门3的输出同相

bit 2 **G3POL**：门2输出极性控制位
1 = 门2的输出在施加到逻辑单元时反相
0 = 门2的输出同相

bit 1 **G2POL**：门1输出极性控制位
1 = 门1的输出在施加到逻辑单元时反相
0 = 门1的输出同相

bit 0 **G1POL**：门0输出极性控制位
1 = 门0的输出在施加到逻辑单元时反相
0 = 门0的输出同相

寄存器27-3： CLCxSEL0：通用CLCx数据0选择寄存器

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	D1S<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 5-0 **D1S<5:0>：**CLCx数据1输入选择位
请参见表27-1。

寄存器27-4： CLCxSEL1：通用CLCx数据1选择寄存器

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	D2S<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 5-0 **D2S<5:0>：**CLCx数据2输入选择位
请参见表27-1。

寄存器27-5： CLCxSEL2：通用CLCx数据2选择寄存器

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	D3S<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 5-0 **D3S<5:0>：**CLCx数据3输入选择位
请参见表27-1。

寄存器27-6： CLCxSEL3：通用CLCx数据3选择寄存器

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	D4S<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 5-0 **D4S<5:0>：**CLCx数据4输入选择位
请参见表27-1。

寄存器27-7： CLCxGLS0：门0逻辑选择寄存器

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **G1D4T：**门0数据4真值（同相）位
1 = CLCIN3（真值）通过门输入到CLCx门0
0 = CLCIN3（真值）不通过门输入到CLCx门0

bit 6 **G1D4N：**门0数据4取反（反相）位
1 = CLCIN3（反相）通过门输入到CLCx门0
0 = CLCIN3（反相）不通过门输入到CLCx门0

bit 5 **G1D3T：**门0数据3真值（同相）位
1 = CLCIN2（真值）通过门输入到CLCx门0
0 = CLCIN2（真值）不通过门输入到CLCx门0

bit 4 **G1D3N：**门0数据3取反（反相）位
1 = CLCIN2（反相）通过门输入到CLCx门0
0 = CLCIN2（反相）不通过门输入到CLCx门0

bit 3 **G1D2T：**门0数据2真值（同相）位
1 = CLCIN1（真值）通过门输入到CLCx门0
0 = CLCIN1（真值）不通过门输入到CLCx门0

bit 2 **G1D2N：**门0数据2取反（反相）位
1 = CLCIN1（反相）通过门输入到CLCx门0
0 = CLCIN1（反相）不通过门输入到CLCx门0

bit 1 **G1D1T：**门0数据1真值（同相）位
1 = CLCIN0（真值）通过门输入到CLCx门0
0 = CLCIN0（真值）不通过门输入到CLCx门0

bit 0 **G1D1N：**门0数据1取反（反相）位
1 = CLCIN0（反相）通过门输入到CLCx门0
0 = CLCIN0（反相）不通过门输入到CLCx门0

寄存器27-8： CLCxGLS1：门1逻辑选择寄存器

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **G2D4T：**门1数据4真值（同相）位
1 = CLCIN3（真值）通过门输入到CLCx门1
0 = CLCIN3（真值）不通过门输入到CLCx门1

bit 6 **G2D4N：**门1数据4取反（反相）位
1 = CLCIN3（反相）通过门输入到CLCx门1
0 = CLCIN3（反相）不通过门输入到CLCx门1

bit 5 **G2D3T：**门1数据3真值（同相）位
1 = CLCIN2（真值）通过门输入到CLCx门1
0 = CLCIN2（真值）不通过门输入到CLCx门1

bit 4 **G2D3N：**门1数据3取反（反相）位
1 = CLCIN2（反相）通过门输入到CLCx门1
0 = CLCIN2（反相）不通过门输入到CLCx门1

bit 3 **G2D2T：**门1数据2真值（同相）位
1 = CLCIN1（真值）通过门输入到CLCx门1
0 = CLCIN1（真值）不通过门输入到CLCx门1

bit 2 **G2D2N：**门1数据2取反（反相）位
1 = CLCIN1（反相）通过门输入到CLCx门1
0 = CLCIN1（反相）不通过门输入到CLCx门1

bit 1 **G2D1T：**门1数据1真值（同相）位
1 = CLCIN0（真值）通过门输入到CLCx门1
0 = CLCIN0（真值）不通过门输入到CLCx门1

bit 0 **G2D1N：**门1数据1取反（反相）位
1 = CLCIN0（反相）通过门输入到CLCx门1
0 = CLCIN0（反相）不通过门输入到CLCx门1

寄存器27-9： CLCxGLS2：门2逻辑选择寄存器

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **G3D4T：**门2数据4真值（同相）位
1 = CLCIN3（真值）通过门输入到CLCx门2
0 = CLCIN3（真值）不通过门输入到CLCx门2

bit 6 **G3D4N：**门2数据4取反（反相）位
1 = CLCIN3（反相）通过门输入到CLCx门2
0 = CLCIN3（反相）不通过门输入到CLCx门2

bit 5 **G3D3T：**门2数据3真值（同相）位
1 = CLCIN2（真值）通过门输入到CLCx门2
0 = CLCIN2（真值）不通过门输入到CLCx门2

bit 4 **G3D3N：**门2数据3取反（反相）位
1 = CLCIN2（反相）通过门输入到CLCx门2
0 = CLCIN2（反相）不通过门输入到CLCx门2

bit 3 **G3D2T：**门2数据2真值（同相）位
1 = CLCIN1（真值）通过门输入到CLCx门2
0 = CLCIN1（真值）不通过门输入到CLCx门2

bit 2 **G3D2N：**门2数据2取反（反相）位
1 = CLCIN1（反相）通过门输入到CLCx门2
0 = CLCIN1（反相）不通过门输入到CLCx门2

bit 1 **G3D1T：**门2数据1真值（同相）位
1 = CLCIN0（真值）通过门输入到CLCx门2
0 = CLCIN0（真值）不通过门输入到CLCx门2

bit 0 **G3D1N：**门2数据1取反（反相）位
1 = CLCIN0（反相）通过门输入到CLCx门2
0 = CLCIN0（反相）不通过门输入到CLCx门2

寄存器27-10： CLCxGLS3：门3逻辑选择寄存器

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **G4D4T：**门3数据4真值（同相）位
1 = CLCIN3（真值）通过门输入到CLCx门3
0 = CLCIN3（真值）不通过门输入到CLCx门3

bit 6 **G4D4N：**门3数据4取反（反相）位
1 = CLCIN3（反相）通过门输入到CLCx门3
0 = CLCIN3（反相）不通过门输入到CLCx门3

bit 5 **G4D3T：**门3数据3真值（同相）位
1 = CLCIN2（真值）通过门输入到CLCx门3
0 = CLCIN2（真值）不通过门输入到CLCx门3

bit 4 **G4D3N：**门3数据3取反（反相）位
1 = CLCIN2（反相）通过门输入到CLCx门3
0 = CLCIN2（反相）不通过门输入到CLCx门3

bit 3 **G4D2T：**门3数据2真值（同相）位
1 = CLCIN1（真值）通过门输入到CLCx门3
0 = CLCIN1（真值）不通过门输入到CLCx门3

bit 2 **G4D2N：**门3数据2取反（反相）位
1 = CLCIN1（反相）通过门输入到CLCx门3
0 = CLCIN1（反相）不通过门输入到CLCx门3

bit 1 **G4D1T：**门3数据1真值（同相）位
1 = CLCIN0（真值）通过门输入到CLCx门3
0 = CLCIN0（真值）不通过门输入到CLCx门3

bit 0 **G4D1N：**门3数据1取反（反相）位
1 = CLCIN0（反相）通过门输入到CLCx门3
0 = CLCIN0（反相）不通过门输入到CLCx门3

寄存器27-11： CLCDATA：CLC数据输出

U-0	U-0	U-0	U-0	R-0	R-0	R-0	R-0
—	—	—	—	CLC4OUT	CLC3OUT	CLC2OUT	CLC1OUT
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4 **未实现：**读为0

bit 3 **CLC4OUT：**CLC4CON寄存器的OUT镜像副本位

bit 2 **CLC3OUT：**CLC3CON寄存器的OUT镜像副本位

bit 1 **CLC2OUT：**CLC2CON寄存器的OUT镜像副本位

bit 0 **CLC1OUT：**CLC1CON寄存器的OUT镜像副本位

表27-3： 与CLCx相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
CLCxCON	EN	—	OUT	INTP	INTN	MODE<2:0>			426
CLCxPOL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	427
CLCxSEL0	—	—	D1S<5:0>						428
CLCxSEL1	—	—	D2S<5:0>						428
CLCxSEL2	—	—	D3S<5:0>						428
CLCxSEL3	—	—	D4S<5:0>						428
CLCxGLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	429
CLCxGLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	430
CLCxGLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	431
CLCxGLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	432
CLCDATA	—	—	—	—	CLC4OUT	CLC3OUT	CLC2OUT	CLC1OUT	433

图注： — = 未实现，读为0。CLCx模块不使用阴影单元。

28.0 数控振荡器（NCO）模块

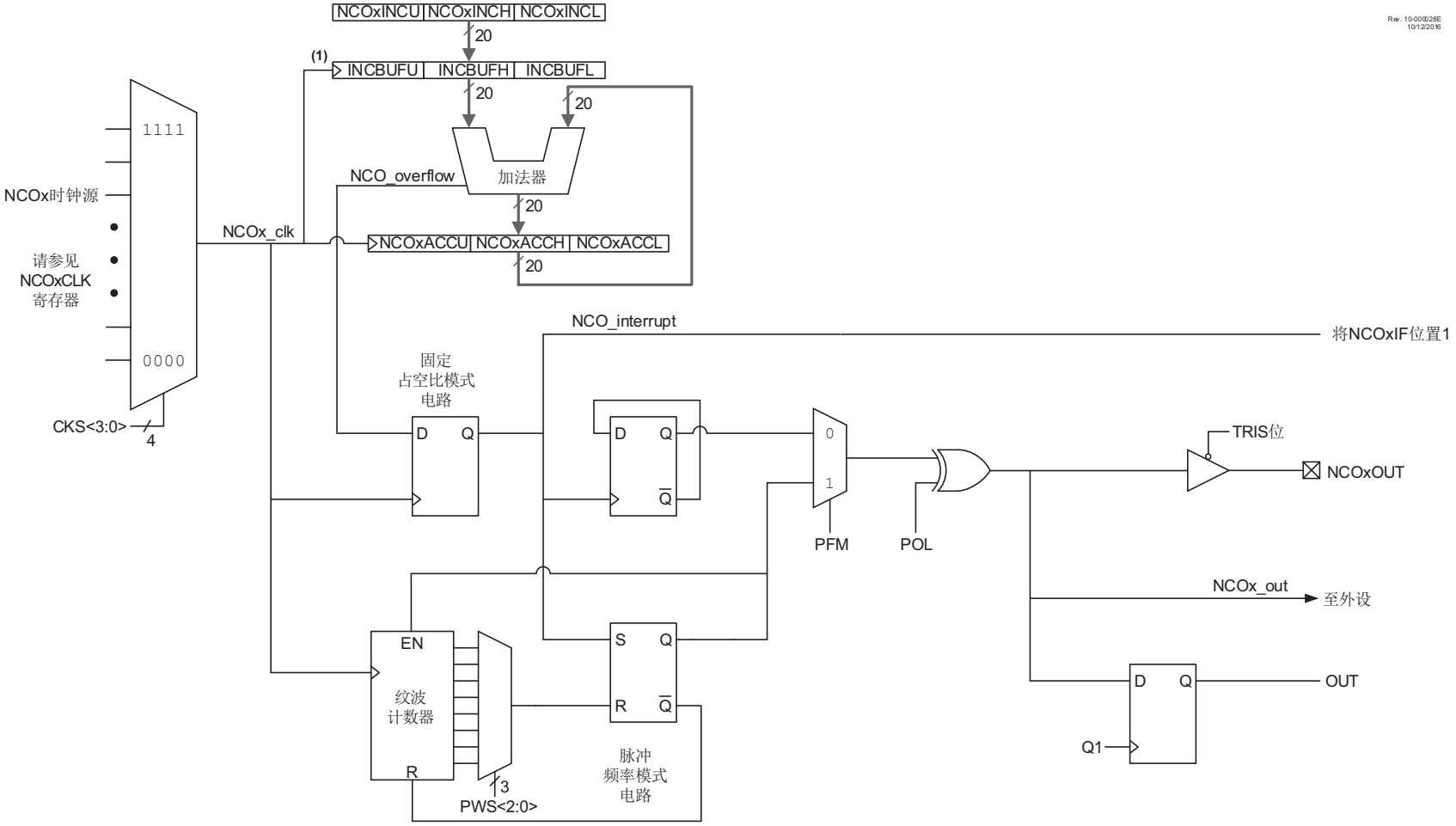
数控振荡器（NCO）模块是一个定时器，它使用由于加上递增值而引起的溢出来对输入频率进行分频。加法运算相比于简单计数器驱动定时器的优势在于，输出频率分辨率不会随着分频器值的变化而变化。NCO对于要求在固定占空比条件下确保频率精度和精细分辨率的应用最为有用。

NCO的特性包括：

- 20位递增功能
- 固定占空比（Fixed Duty Cycle，FDC）模式
- 脉冲频率（Pulse Frequency，PF）模式
- 输出脉冲宽度控制
- 多个时钟输入源
- 输出极性控制
- 中断能力

图28-1给出了NCO模块的简化框图。

图28-1： 直接数字合成模块简化框图



注1： 递增寄存器是双重缓冲的，以允许在没有先禁止NCO模块的情况下更改值。写入NCOxINCL寄存器后会立即产生NCOx_clk信号，完整递增值将在该信号的第二个上升沿装入缓冲区寄存器。用户不能访问该缓冲区，此处仅将该缓冲区作为参考。

28.1 NCO工作原理

NCO的工作方式是重复向累加器增加一个固定值。达到特定输入时钟速率时会进行加法运算。累加器会定期发生进位溢出，该进位为原始的NCO输出（NCO_overflow）。这会有效降低输入时钟，降低的倍数为增加值与最大累加器值的比率。请参见公式28-1。

NCO输出可以通过延长脉冲或翻转触发器进一步进行修正。修正后的NCO输出随后会内部分配至其他外设，也可选择性地输出到引脚。累加器溢出也会产生中断（NCO_overflow）。

NCO周期以离散步阶进行变化，从而产生一个平均频率。该输出依靠接收电路（即，CWG或外部谐振转换器电路）平均NCO输出的能力来降低不确定性。

公式28-1： NCO溢出频率

$$F_{OVERFLOW} = \frac{NCO时钟频率 \times 递增值}{2^{20}}$$

28.1.1 NCO时钟源

可用于NCO的时钟源包括：

- FOSC
- HFINTOSC
- LFINTOSC
- MFINTOSC/4（32 kHz）
- MFINTOSC（500 kHz）
- CLC1/2/3/4_out
- CLKREF
- SOSC

NCO时钟源通过配置NCO1CLK寄存器的N1CKS<2:0>位进行选择。

28.1.2 累加器

累加器是一个20位寄存器。可通过以下三个寄存器对累加器进行读写访问：

- NCO1ACCL
- NCO1ACCH
- NCO1ACCU

28.1.3 加法器

NCO加法器是一个全加器，它独立于时钟源工作。先前结果与递增值的加法运算结果将在每个输入时钟的上升沿替换累加器值。

28.1.4 递增寄存器

递增值存储在三个寄存器中，以构成一个20位递增寄存器。寄存器按从低字节到高字节的顺序排列为：

- NCO1INCL
- NCO1INCH
- NCO1INCU

当使能NCO模块时，应首先写NCO1INCU和NCO1INCH寄存器，然后写NCO1INCL寄存器。NCO1INCL寄存器写操作会在NCO_clk信号的第二个上升沿同时启动装入递增缓冲寄存器。

这些寄存器是可读写的。递增寄存器是双重缓冲的，以允许在没有先禁止NCO模块的情况下更改值。

当禁止NCO模块时，在对递增寄存器执行写操作后，会立即装入递增缓冲区。

注： 用户不能访问递增缓冲寄存器。

28.2 固定占空比模式

在固定占空比（FDC）模式下，每次累加器发生溢出（NCO_overflow）时，输出都会发生翻转。在递增值保持不变的情况下，这会提供50%的占空比。更多信息，请参见图28-2。

28.3 脉冲频率模式

在脉冲频率（PF）模式下，每次累加器溢出时，输出会在一个或多个时钟周期内有效。时钟周期结束时，输出返回到无效状态。这就产生了一个脉冲控制的输出。输出在紧跟溢出事件之后的时钟上升沿变为有效。更多信息，请参见图28-2。

有效状态和无效状态的值取决于NCO1CON寄存器中的极性位POL。

可通过将NCO1CON寄存器中的PFM位置1来选择PF模式。

28.3.1 输出脉冲宽度控制

在PF模式下工作时，输出有效状态的宽度会随多个时钟周期而变化。可使用NCO1CLK寄存器中的PWS<2:0>位来选择各种脉冲宽度。

当所选脉冲宽度大于累加器溢出时间帧时，DDS操作不确定。

28.4 输出极性控制

NCO模块中的最后一级是输出极性。NCO1CON寄存器中的POL位用于选择输出极性。如果在允许中断时改变极性，则会导致输出结果的变化发生中断。NCO输出信号可用于该器件上提供的大多数其他外设。

28.5 中断

当累加器溢出（NCO_overflow）时，PIR4寄存器的NCO中断标志位NCO1IF置1。要允许此中断事件（NCO_interrupt），必须将以下各位置1：

- NCO1CON寄存器的EN位
- PIE4寄存器的NCO1IE位
- INTCON0寄存器的GIE/GIEH位

中断必须由软件通过在中断服务程序中将NCO1IF位清零来清除。

28.6 复位的影响

发生复位后，所有NCO寄存器都会清零。

28.7 休眠模式下的操作

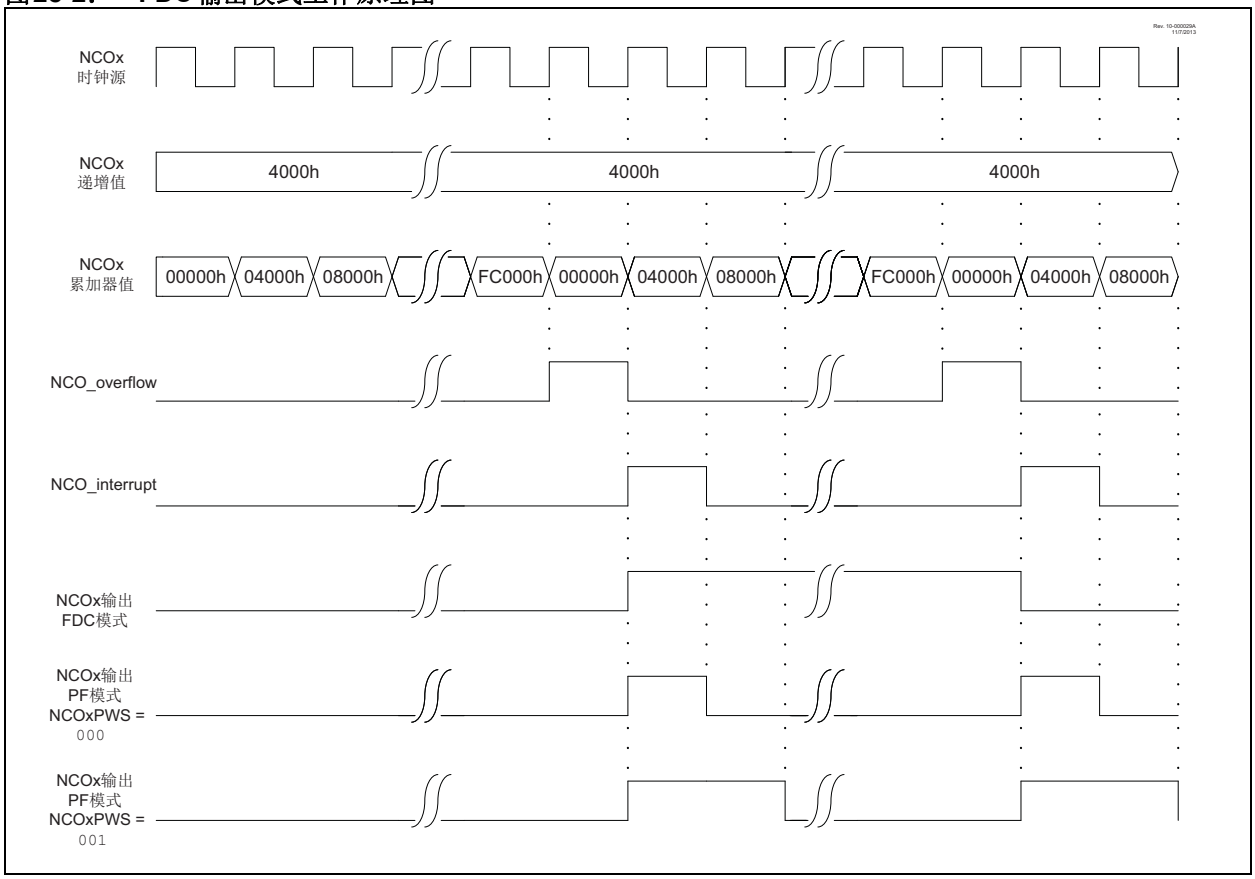
NCO模块独立于系统时钟工作，只要选定的时钟源保持活动状态，它就会继续在休眠期间运行。

如果使能了NCO模块，并且选择HFINTOSC作为时钟源，则无论所选择的系统时钟源如何，HFINTOSC都会在休眠期间保持活动状态。

即，如果在NCO使能时，同时选择HFINTOSC作为系统时钟和NCO时钟源，则在休眠期间CPU会进入空闲状态，而NCO会继续工作，并且HFINTOSC将保持活动状态。

这会直接影响休眠模式的电流。

图28-2： FDC输出模式工作原理图



28.8 NCO控制寄存器

寄存器28-1： NCO1CON：NCO控制寄存器

R/W-0/0	U-0	R-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
EN	—	OUT	POL	—	—	—	PFM
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN**：NCO1使能位
1 = 使能NCO1模块
0 = 禁止NCO1模块

bit 6 **未实现**：读为0

bit 5 **OUT**：NCO1输出位
显示NCO1模块的当前输出值。

bit 4 **POL**：NCO1极性
1 = NCO1输出信号反相
0 = NCO1输出信号同相

bit 3-1 **未实现**：读为0

bit 0 **PFM**：NCO1脉冲频率模式位
1 = NCO1在脉冲频率模式下工作
0 = NCO1在固定占空比模式下工作，2分频

寄存器28-2： NCO1CLK：NCO1输入时钟控制寄存器

R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
PWS<2:0>(1,2)			—	CKS<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **PWS<2:0>**：NCO1输出脉冲宽度选择位(1,2)
111 = NCO1输出在128个输入时钟周期内有效
110 = NCO1输出在64个输入时钟周期内有效
101 = NCO1输出在32个输入时钟周期内有效
100 = NCO1输出在16个输入时钟周期内有效
011 = NCO1输出在8个输入时钟周期内有效
010 = NCO1输出在4个输入时钟周期内有效
001 = NCO1输出在2个输入时钟周期内有效
000 = NCO1输出在1个输入时钟周期内有效

bit 4 **未实现**：读为0

bit 3-0 **CKS<3:0>**：NCO1时钟源选择位
1111 = 保留
•
•
•
1011 = 保留
1010 = CLC4_out
1001 = CLC3_out
1000 = CLC2_out
0111 = CLC1_out
0110 = CLKREF_out
0101 = SOSC
0100 = MFINTOSC/4（32 kHz）
0011 = MFINTOSC（500 kHz）
0010 = LFINTOSC
0001 = HFINTOSC
0000 = FOSC

注 1： 仅在脉冲频率模式下工作时N1PWS才适用。
2： 如果NCO1脉冲宽度大于NCO1溢出周期，则操作将不确定。

寄存器28-3： NCO1ACCL：NCO1累加器寄存器——低字节

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ACC<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ACC<7:0>：**NCO1累加器低字节

寄存器28-4： NCO1ACCH：NCO1累加器寄存器——高字节

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ACC<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ACC<15:8>：**NCO1累加器高字节

寄存器28-5： NCO1ACCU：NCO1累加器寄存器——最高字节(1)

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	ACC<19:16>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4 **未实现：**读为0

bit 3-0 **ACC<19:16>：**NCO1累加器最高字节

注 1： 累加器跨越寄存器NCO1ACCU:NCO1ACCH:NCO1ACCL。24位保留，但并未全部使用。该寄存器实时更新，与CPU异步；使用8位总线无法保证原子访问该24位空间。在模块正在工作时写该寄存器将产生不确定结果。

寄存器28-6： NCO1INCL：NCO1递增寄存器——低字节(1,2)

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-1/1
INC<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **INC<7:0>：**NCO1递增低字节

注 1： 逻辑递增跨越NCO1INCU:NCO1INCH:NCO1INCL。

2： NCO1INC双重缓冲为INCBUF；写入NCO1INCL后，INCBUF在写入NCO1INCL后的NCOCLK下一个下降沿更新；应先写NCO1INCU和NCO1INCH，然后再写NCO1INCL。

寄存器28-7： NCO1INCH：NCO1递增寄存器——高字节(1)

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
INC<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **INC<15:8>：**NCO1递增高字节

注 1： 逻辑递增跨越NCO1INCU:NCO1INCH:NCO1INCL。

寄存器28-8： NCO1INCU：NCO1递增寄存器——最高字节(1)

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
INC<19:6>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4 **未实现：**读为0

bit 3-0 **INC<19:16>：**NCO1递增最高字节

注 1： 逻辑递增跨越NCO1INCU:NCO1INCH:NCO1INCL。

表28-1： 与NCO相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
NCO1CON	N1EN	—	N1OUT	N1POL	—	—	—	N1PFM	439
NCO1CLK	N1PWS<2:0>			—	—	N1CKS<2:0>			440
NCO1ACCL	NCO1ACC<7:0>								441
NCO1ACCH	NCO1ACC<15:8>								441
NCO1ACCU	—	—	—	—	NCO1ACC<19:16>				442
NCO1INCL	NCO1INC<7:0>								442
NCO1INCH	NCO1INC<15:8>								442
NCO1INCU	—	—	—	—	NCO1INC<19:16>				443

图注： – = 未实现，读为0。NCO模块不使用阴影单元。

29.0 过零检测（ZCD）模块

ZCD模块用于检测交流信号越过地电位的时刻。实际的过零阈值为过零参考电压VCPINV，它通常比地电位高0.75V。

要检测的信号通过一个串联限流电阻进行连接。该模块会在ZCD引脚上施加拉电流或灌电流，以便在引脚上维持恒定的电压，从而防止引脚电压使ESD保护二极管发生正向偏置。当施加的电压大于参考电压时，该模块会灌电流。当施加的电压小于参考电压时，该模块会拉电流。拉电流和灌电流操作会使引脚电压在所施加电压的完整范围内保持恒定。图29-2给出了ZCD模块的简化框图。

ZCD模块可用于为以下用途（但不限于以下用途）监视交流波形：

- 交流周期测量
- 精确的长期时间测量
- 调光器相位延时驱动
- 低EMI周期切换

29.1 外部电阻选择

ZCD模块要求将一个限流电阻与外部电压源串联。该电阻的阻抗和参数取决于外部源峰值电压。在通过电阻的电流标称值为300 µA时，选择的阻值产生的压降需要等于峰值电压。请参见公式29-1和图29-1。请确保禁止ZCD I/O引脚内部弱上拉，使它不会干扰拉电流和灌电流。

公式29-1： 外部电阻

$$RSERIES = \frac{VPEAK}{3\times10^{-4}}$$

图29-1： 外部电压

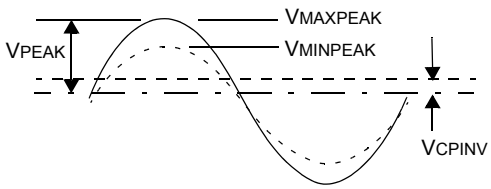
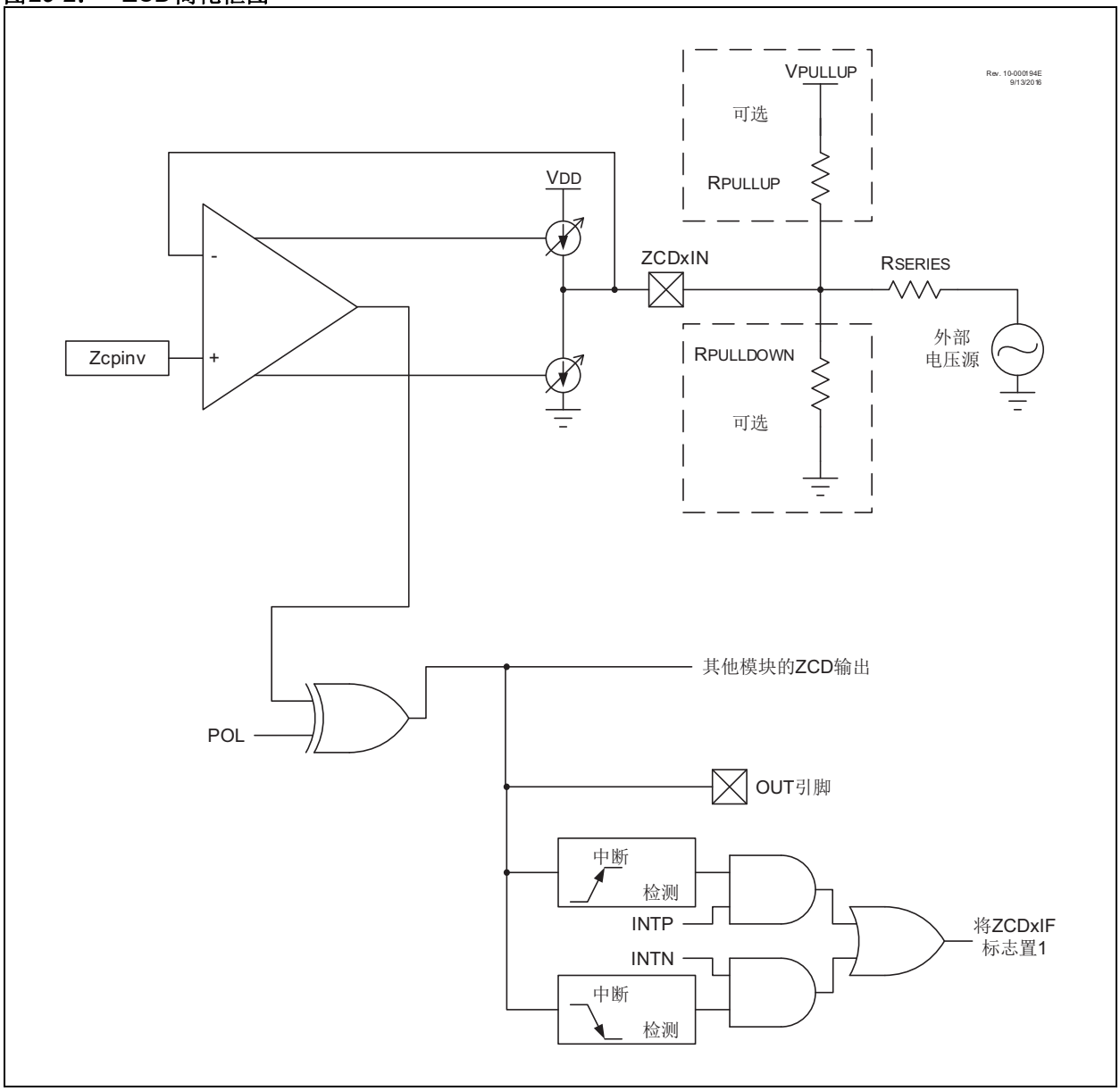


图29-2： ZCD简化框图



29.2 ZCD逻辑输出

ZCD模块包含了一个状态位，通过读取它可以确定处于工作状态的是拉电流还是灌电流。当处于工作状态的是灌电流时，ZCDCON寄存器的OUT位会置1，当处于工作状态的是拉电流时，该位会清零。即使禁止模块，OUT位也会受极性位影响。

OUT信号也可用作其他模块的输入。这由相应模块的寄存器来控制。OUT可用作以下源：

- TMR1/3/5的门控源
- TMR2/4/6的时钟源
- TMR2/4/6的复位源

29.3 ZCD逻辑极性

ZCDCON寄存器的POL位用于将OUT位相对于拉电流和灌电流输出进行反相。当POL位置1时，OUT为高电平表示处于工作状态的是拉电流，低电平输出表示处于工作状态的是灌电流。

POL位会影响ZCD中断。

29.4 ZCD中断

如果相应的中断允许位置1，则在ZCD逻辑输出发生改变时，将会产生中断。因此，ZCD中具有一个上升沿检测器和一个下降沿检测器。

触发其中一个边沿检测器，且其相关的中断允许位置1时，相应PIR寄存器的ZCDIF位会置1。INTP位用于允许上升沿中断，INTN位用于允许下降沿中断。它们都位于ZCDCON寄存器中。如果INTCON寄存器的IPEN位置1，则可以更改中断的优先级。可以通过将相应IPR寄存器的ZCDIP位置1或清零，将ZCD中断设置为高优先级或低优先级。

要完全允许中断，必须将以下位置1：

- 相应PIE寄存器的ZCDIE位
- ZCDCON寄存器的INTP位（对于上升沿检测）
- ZCDCON寄存器的INTN位（用于下降沿检测）
- INTCON0寄存器的GIE位

更改POL位将导致中断，无论SEN位的电平如何。

作为中断服务的一部分，必须用软件将相应PIR寄存器的ZCDIF位清零。如果在清零该标志时检测到另一个边沿，则标志仍然会在序列结束时置1。

29.5 修正VCPINV偏移

ZCD发生切换时的实际电压是ZCD运放同相输入上的参考电压。对于除方波之外的其他外部电压源波形，该电压相对于零电压发生偏移会导致过早或过迟地发生过零事件。当波形相对VSS发生变化时，若波形下降，则会过早检测到过零事件；若波形上升，则会过晚检测到过零事件。当波形相对VDD发生变化时，若波形上升，则会过晚检测到过零事件；若波形下降，则会过早检测到过零事件。使用公式29-2所示的相应公式，可以确定正弦波形的实际偏移时间。

公式29-2： ZCD事件偏移

当外部电压源相对VSS发生变化时：

$$T_{OFFSET} = \frac{asin\left(\frac{V_{CPINV}}{V_{PEAK}}\right)}{2\pi \bullet Freq}$$

当外部电压源相对VDD发生变化时：

$$T_{OFFSET} = \frac{asin\left(\frac{V_{DD}-V_{CPINV}}{V_{PEAK}}\right)}{2\pi \bullet Freq}$$

这种偏移时间可以通过在ZCD引脚添加上拉或下拉偏置电阻进行补偿。当外部电压源相对VSS发生变化时，使用上拉电阻。当电压相对VDD发生变化时，使用下拉电阻。该电阻会增大ZCD引脚的偏置电压，使得只有目标外部电压源达到零电压时，才会将引脚电压拉至VCPINV切换电压。可以使用公式29-3或公式29-4确定上拉或下拉值。

公式29-3： ZCD上拉/下拉

当外部信号相对Vss发生变化时：

$$R_{PULLUP} = \frac{R_{SERIES}(V_{PULLUP} - V_{CPINV})}{V_{CPINV}}$$

当外部信号相对VDD发生变化时：

$$R_{PULLDOWN} = \frac{R_{SERIES}(V_{CPINV})}{(V_{DD} - V_{CPINV})}$$

测量VCPINV是很困难的，特别是在波形是相对于VDD时。但是，通过结合公式29-2和公式29-3，可以根据ZCD_output高电平和低电平周期之间的时间差来确定电阻值。请注意，时间差ΔT为4*TOFFSET。可使用公式29-4根据高电平和低电平ZCD_output周期确定上拉和下拉电阻值。

公式29-4：　上拉/下拉电阻值

$$R = R_{SERIES}\left(\frac{V_{BIAS}}{V_{PEAK}\left(\sin\left(\pi Freq\frac{(\Delta T)}{2}\right)\right)} - 1\right)$$

R为上拉或下拉电阻。

VBIAS是VPULLUP（当R为上拉电阻时）或VDD（当R为下拉电阻时）。

ΔT是ZCDOUT高电平周期与低电平周期的时间差。

29.6　处理VPEAK变化

如果预期外部电压的峰值大小会发生变化，则选择的串联电阻必须使ZCD拉电流和灌电流保持低于±600 μA的最大设计范围，并高于合理的最小范围。一般的经验是，最大峰值电压不能超出最小峰值电压的6倍。为了确保最大电流不超出±600 μA，最小值至少为±100 μA，需要按照公式29-5所示的方式计算串联电阻。该串联电阻的补偿上拉电阻可以使用公式29-3确定，因为上拉值与峰值电压无关。

公式29-5：　对应于V范围的串联电阻R

$$R_{SERIES} = \frac{V_{MAXPEAK} + V_{MINPEAK}}{7\times10^{-4}}$$

29.7　休眠期间的操作

ZCD电流源和中断不受休眠影响。

29.8　复位的影响

ZCD电路可以配置为在发生上电复位（POR）时默认设为工作或非工作状态。当$\overline{\text{ZCD}}$配置位清零时，ZCD电路将在POR时处于工作状态。当$\overline{\text{ZCD}}$配置位置1时，必须通过将ZCDCON寄存器的SEN位置1来使能ZCD模块。

29.9　禁止ZCD模块

可以采用以下两种方式禁止ZCD模块：

1. 配置字2H具有$\overline{\text{ZCD}}$位，该位在置1时会禁止ZCD模块，但可以使用ZCDCON寄存器（寄存器29-1）的SEN位来使能该位。如果$\overline{\text{ZCD}}$位清零，则ZCD始终使能。
2. 也可以使用PMD2寄存器（寄存器19-3）的ZCDMD位禁止ZCD。这会受到$\overline{\text{ZCD}}$位状态的制约。

29.10 寄存器定义：ZCD控制

寄存器29-1： ZCDCON：过零检测控制寄存器

R/W-0/0	U-0	R-x	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
SEN	—	OUT	POL	—	—	INTP	INTN
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **SEN：**过零检测软件使能位
当ZCDSEN配置位置1时忽略该位。
1 = 使能过零检测。
0 = 禁止过零检测。ZCD引脚根据PPS和TRIS控制工作。

bit 6 **未实现：**读为0

bit 5 **OUT：**过零检测数据输出位
ZCDPOL位 = 0：
1 = ZCD引脚产生灌电流
0 = ZCD引脚产生拉电流
ZCDPOL位 = 1：
1 = ZCD引脚产生拉电流
0 = ZCD引脚产生灌电流

bit 4 **POL：**过零检测极性位
1 = 对ZCD逻辑输出进行反相
0 = 不对ZCD逻辑输出进行反相

bit 3-2 **未实现：**读为0

bit 1 **INTP：**过零检测正向边沿中断允许位
1 = 当ZCD_output从低电平跳变为高电平时，ZCDIF位置1
0 = 当ZCD_output从低电平跳变为高电平时，ZCDIF位不受影响

bit 0 **INTN：**过零检测负向边沿中断允许位
1 = 当ZCD_output从高电平跳变为低电平时，ZCDIF位置1
0 = 当ZCD_output从高电平跳变为低电平时，ZCDIF位不受影响

表29-1： 与ZCD模块相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
ZCDCON	SEN	—	OUT	POL	—	—	INTP	INTN	448

图注： — = 未实现，读为0。ZCD模块不使用阴影单元。

30.0 数据信号调制器（DSM）模块

数据信号调制器（DSM）是一种外设，用户可以通过它将数据流（也称为调制器信号）与载波信号进行混合来产生调制输出。

载波和调制器信号均送到DSM模块，信号可以来自内部、某个外设的输出，也可以通过某个输入引脚从外部提供。

调制输出信号通过对载波信号和调制器信号执行逻辑“与”运算生成，然后提供给MDOUT引脚。

载波信号由两个不同的独立信号组成：高载波（CARH）信号和低载波（CARL）信号。在调制器（MOD）信号处于逻辑高电平状态期间，DSM会将高载波信号与调制器信号进行混合。在调制器信号处于逻辑低电平状态时，DSM会将低载波信号与调制器信号进行混合。

通过这种方法，DSM可以产生以下几种键控调制方案：

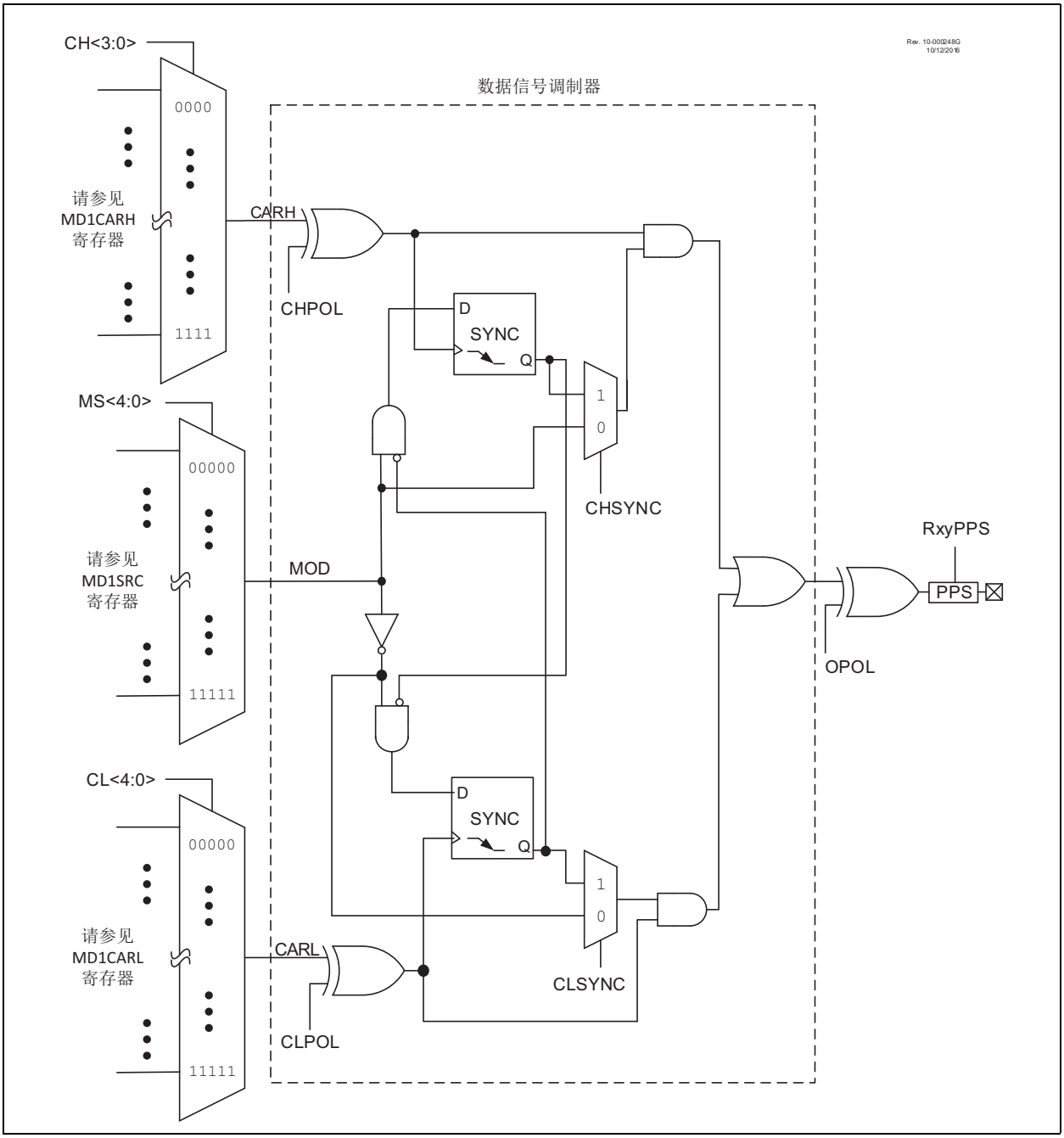
- 频移键控（Frequency-Shift Keying，FSK）
- 相移键控（Phase-Shift Keying，PSK）
- 开关键控（On-Off Keying，OOK）

此外，DSM模块还提供了以下特性：

- 载波同步
- 载波源极性选择
- 可编程调制器数据
- 调制输出极性选择
- 外设模块禁止，可使DSM模块进入最低功耗模式

图30-1给出了数据信号调制器外设的简化框图。

图30-1： 数据信号调制器的简化框图



30.1 DSM工作原理

DSM模块通过将MD1CON0寄存器的EN位置1来使能。清零MD1CON0寄存器中的EN位会禁止DSM模块输出并将高载波信号和低载波信号分别切换为MD1CARHPPS和MD1CARLPPS的默认选项。调制器信号源也会被切换至MD1CON0寄存器中的BIT位。

当EN位清零且DSM模块被禁止时，由调制源、调制高载波信号和调制低载波信号控制寄存器保存的、用于选择高载波信号、低载波信号和调制器信号源的值不会受影响。在DSM不工作时，这些寄存器中的值会保持不变。当EN位置1且DSM模块再次使能并处于工作状态时，将会再次选择高载波信号、低载波信号和调制器信号的信号源。

30.2 调制器信号源

调制器信号可以从表30-3中指定的源提供：

调制器信号通过配置MD1SRC寄存器的MS<4:0>位来选择。

30.3 载波信号源

高载波信号和低载波信号可以从表30-1中指定的源提供。

高载波信号通过配置MD1CARH寄存器的CH<4:0>位来选择。低载波信号通过配置MD1CARL寄存器的CL<4:0>位来选择。

30.4 载波同步

当DSM在高载波信号源和低载波信号源之间切换时，调制输出信号中的载波数据可能会被截短。为了防止这种情况，可以将载波信号与调制器信号进行同步。当使能同步时，DSM将允许在切换时进行混合的载波脉冲先变为低电平，然后再切换为另一个载波源。

对于高载波信号源和低载波信号源，同步功能单独进行使能。高载波信号的同步通过将MD1CON1寄存器的CHSYNC位置1来使能。低载波信号的同步通过将MD1CON1寄存器的CLSYNC位置1来使能。

图30-2至图30-6给出了使用各种同步方法时的时序图。

图30-2： 开关键控（OOK）同步

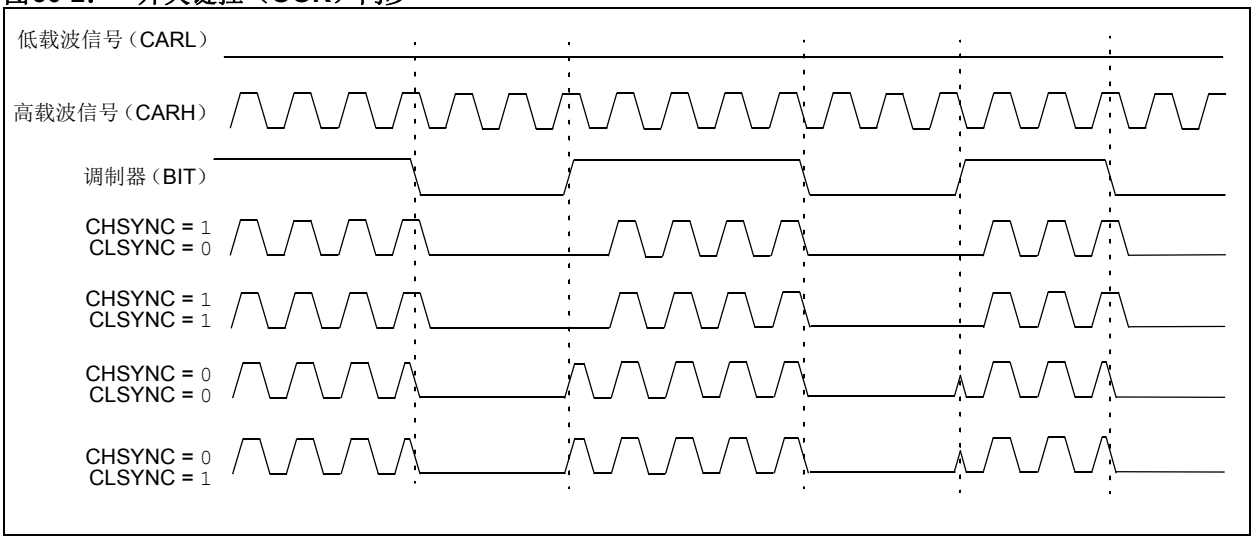


图30-3： 无同步（CHSYNC = 0，CLSYNC = 0）

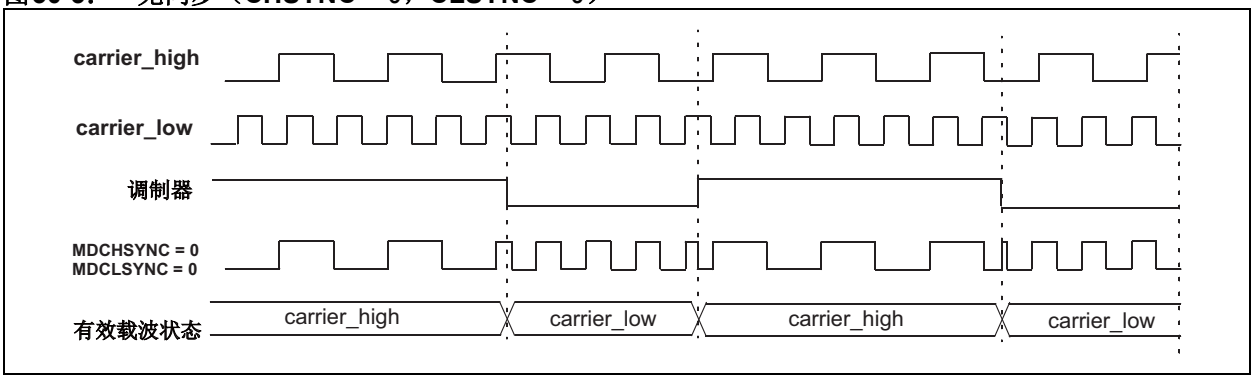


图30-4： 高载波信号同步（CHSYNC = 1，CLSYNC = 0）

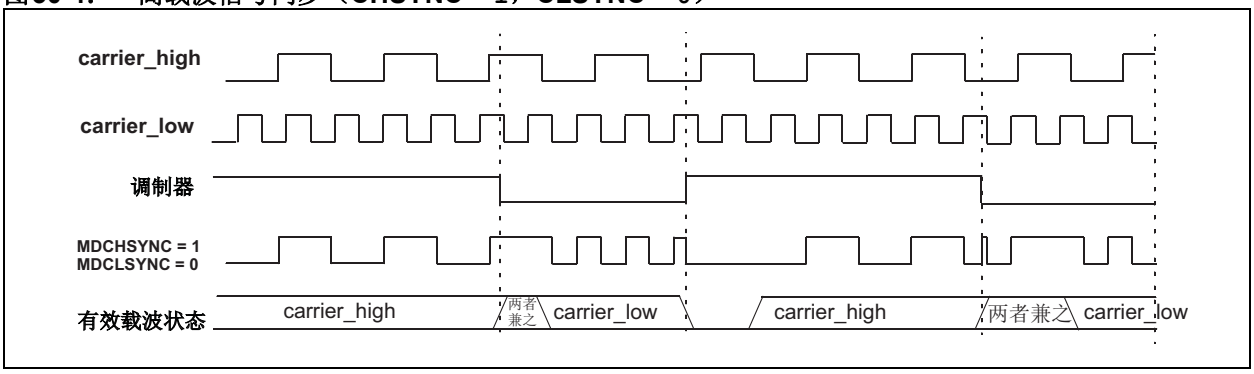


图30-5：　低载波信号同步（CHSYNC = 0，CLSYNC = 1）

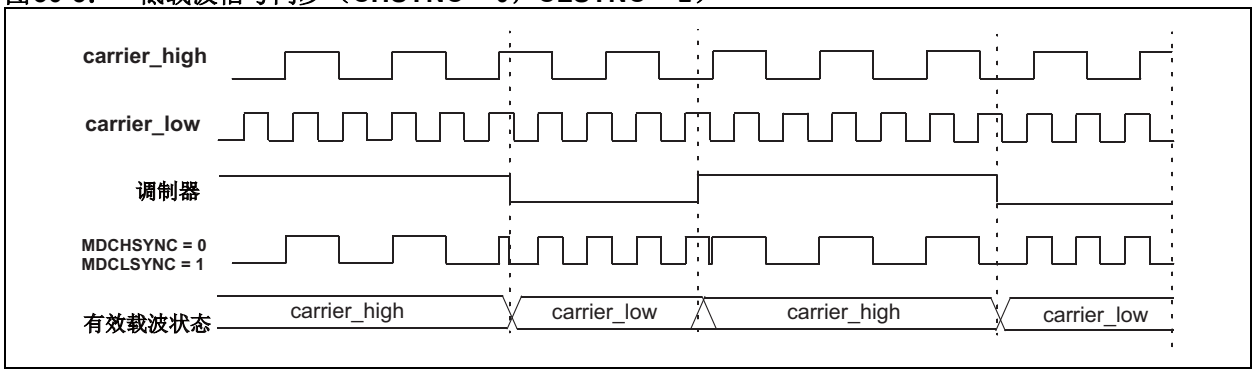
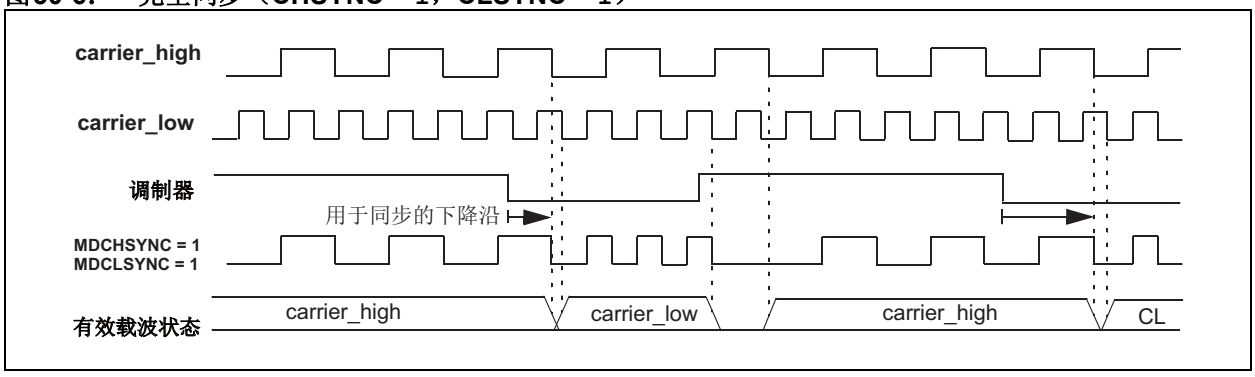


图30-6：　完全同步（CHSYNC = 1，CLSYNC = 1）



30.5 载波源极性选择

高载波信号和低载波信号的任意选定输入源提供的信号都可以进行反相。对高载波信号源信号反相通过将MD1CON1寄存器的CHPOL位置1来使能。对低载波信号源信号反相通过将MD1CON1寄存器的CLPOL位置1来使能。

30.6 可编程调制器数据

用户可以选择MD1CON0寄存器的BIT位作为调制器信号的信号源。这使得用户能够设置用于调制的值。

30.7 调制输出极性

送到DSM引脚上的调制输出信号也可以进行反相。调制输出信号反相通过将MD1CON0寄存器的OPOL位置1来使能。

30.8 休眠模式下的操作

DSM模块不受休眠模式的影响。如果载波输入源和调制器输入源仍在休眠期间工作，则DSM在休眠期间仍然可以工作。更多详细信息，请参见**第10.0节“节能工作模式”**。

30.9 复位的影响

在发生任何器件复位时，都将禁止DSM模块。用户的固件负责在使能输出之前初始化该模块。寄存器会复位为其默认值。

30.10 外设模块禁止

可以使用PMD模块完全禁止DSM模块，从而最大限度地降低功耗。当PMD6（寄存器19-7）的DSMMD位置1时，将完全禁止DSM模块。重新使能时，DSM模块的所有寄存器都默认为POR状态。

30.11 寄存器定义：调制控制

调制外设的长位名称前缀如下所示。更多信息，请参见**第1.3.2.2节“长位名称”**。

外设	位名称前缀
MD1	MD1

寄存器30-1： MD1CON0：调制控制寄存器0

R/W-0/0	U-0	R-0/0	R/W-0/0	U-0	U-0	U-0	R/W-0/0
EN	—	OUT	OPOL	—	—	—	BIT
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN：**调制器模块使能位
1 = 使能调制器模块，且该模块混合输入信号
0 = 禁止调制器模块，且该模块没有输出

bit 6 **未实现：**读为0

bit 5 **OUT：**调制器输出位
指示调制器模块的当前输出值。(1)

bit 4 **OPOL：**调制器输出极性选择位
1 = 调制器输出信号反相；空闲时输出高电平
0 = 调制器输出信号同相；空闲时输出低电平。

bit 3-1 **未实现：**读为0

bit 0 **BIT：**允许软件手动设置模块的调制源输入(2)
1 = 调制器选择高载波信号
0 = 调制器选择低载波信号

注 1： 调制输出频率可能会高于更新该寄存器位的时钟，与该时钟异步；位值对于速度较高的调制器信号或载波信号可能无效。

2： 对于该操作，必须在MD1SRC寄存器中选择BIT作为调制源。

寄存器30-2： MD1CON1：调制控制寄存器1

U-0	U-0	R/W-0/0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	CHPOL	CHSYNC	—	—	CLPOL	CLSYNC
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 5 **CHPOL：**调制器高载波信号极性选择位
1 = 选定的高载波信号反相
0 = 选定的高载波信号同相

bit 4 **CHSYNC：**调制器高载波信号同步使能位
1 = 调制器先等待高载波信号上出现下降沿，然后再切换为低载波信号
0 = 调制器输出不与高载波信号进行同步(1)

bit 3-2 **未实现：**读为0

bit 1 **CLPOL：**调制器低载波信号极性选择位
1 = 选定的低载波信号反相
0 = 选定的低载波信号同相

bit 0 **CLSYNC：**调制器低载波信号同步使能位
1 = 调制器先等待低载波信号上出现下降沿，然后再切换为高载波信号
0 = 调制器输出不与低载波信号进行同步(1)

注 1： 如果载波未进行同步，则信号流中的载波脉宽可能会变窄，或者可能出现杂散信号。

寄存器30-3： MD1CARH：调制高载波信号控制寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	CH<4:0>(1)				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现：**读为0

bit 4-0 **CH<4:0>：**调制器高载波信号选择位(1)
有关信号列表，请参见表30-1

注 1： 未使用的选择提供输入值。

寄存器30-4： MD1CARL：调制低载波信号控制寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	CL<4:0>(1)				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现：**读为0

bit 4-0 **CL<4:0>：**调制器低载波信号输入选择位(1)
有关信号列表，请参见表30-1

注 1： 未使用的选择提供0作为输入值。

表30-1： MD1CARH/MD1CARL选择多路开关连接

MD1CARH			MD1CARL		
CH<4:0>		连接	CL<4:0>		连接
11111-10011	31-19	保留	11111-10011	31-19	保留
10010	18	CLC4OUT	10010	18	CLC4OUT
10001	17	CLC3OUT	10001	17	CLC3OUT
10000	16	CLC2OUT	10000	16	CLC2OUT
01111	15	CLC1OUT	01111	15	CLC1OUT
01110	14	NCO1OUT	01110	14	NCO1OUT
01101-01100	13-12	保留	01101-01100	13-12	保留
01011	11	PWM8 OUT	01011	11	PWM8 OUT
01010	10	PWM7 OUT	01010	10	PWM7 OUT
01001	9	PWM6 OUT	01001	9	PWM6 OUT
01000	8	PWM5 OUT	01000	8	PWM5 OUT
00111	7	CCP4 OUT	00111	7	CCP4 OUT
00110	6	CCP3 OUT	00110	6	CCP3 OUT
00101	5	CCP2 OUT	00101	5	CCP2 OUT
00100	4	CCP1 OUT	00100	4	CCP1 OUT
00011	3	CLKREF输出	00011	3	CLKREF输出
00010	2	HFINTOSC	00010	2	HFINTOSC
00001	1	FOSC（系统时钟）	00001	1	FOSC（系统时钟）
00000	0	通过MD1CARHPPS选择引脚	00000	0	通过MD1CARLPPS选择引脚

寄存器30-5： MD1SRC：调制源控制寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	MS<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现：**读为0

bit 4-0 **MS<4:0>：**调制器源选择位(1)
有关信号列表，请参见表30-2

注 1： 未使用的选择提供0作为输入值。

表30-2： MD1SRC选择多路开关连接

MS<4:0>		连接
1 1111 – 1 1000	31 - 24	保留
1 0111	23	CAN_tx0
1 0110	22	SPI1 SDO
1 0101	21	保留
1 0100	20	UART2 TX
1 0011	19	UART1 TX
1 0010	18	CLC4 OUT
1 0001	17	CLC3 OUT
1 0000	16	CLC2 OUT
0 1111	15	CLC1 OUT
0 1110	14	CMP2 OUT
0 1101	13	CMP1 OUT
0 1100	12	NCO1 OUT
0 1011	11	保留
0 1010	10	保留
0 1001	9	PWM8 OUT
0 1000	8	PWM7 OUT
0 0111	7	PWM6 OUT
0 0110	6	PWM5 OUT
0 0101	5	CCP4 OUT
0 0100	4	CCP3 OUT
0 0011	3	CCP2 OUT
0 0010	2	CCP1 OUT
0 0001	1	DSM1 BIT
0 0000	0	通过MDSRCPPS选择的引脚

表30-3： 与数据信号调制器模式相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
MD1CON0	EN	—	OUT	OPOL	—	—	—	BIT	455
MD1CON1	—	—	CHPOL	CHSYNC	—	—	CLPOL	CLSYNC	456
MD1CARH	—	—	—	—	—	CHS<2:0>			457
MD1CARL	—	—	—	—	—	CLS<2:0>			457
MDSRC	—	—	—	—	SRCS<3:0>				458

图注： — = 未实现，读为0。数据信号调制器模式不使用阴影单元。

31.0 具有协议支持的通用异步收发器（UART）

通用异步收发器（UART）模块是串行I/O通信外设。它包含用来完成与器件程序执行独立的输入或输出串行数据传输所需的所有时钟发生器、移位寄存器和数据缓冲区。UART也可称为串行通信接口（SCI），可配置为全双工异步系统或多种自动协议之一。全双工模式可用来与外设系统通信，如CRT终端和个人计算机。

支持的协议包括：

- LIN主模式和从模式
- DMX模式
- DALI控制装置和控制器件

UART模块具备以下功能：

- 全双工异步收发
- 双字符输入缓冲区
- 单字符输出缓冲区
- 可编程7位或8位字符长度
- 第9位地址检测
- 第9位偶校验或奇校验
- 输入缓冲区溢出错误检测
- 接收字符帧错误检测
- 硬件和软件流控制
- 自动校验和
- 可编程1、1.5和2个停止位
- 可编程数据极性
- 曼彻斯特编码器/解码器
- 休眠模式下的操作
- 波特率的自动检测和校准
- 接收到间隔字符时唤醒
- 自动和用户定时的间隔周期生成
- RX和TX无活动超时（使用Timer2）

UART发送器和接收器的框图如图31-1和图31-2所示。

UART发送输出（TX_out）可送至TX引脚和在内部送至各种外设。

图31-1： UART发送框图

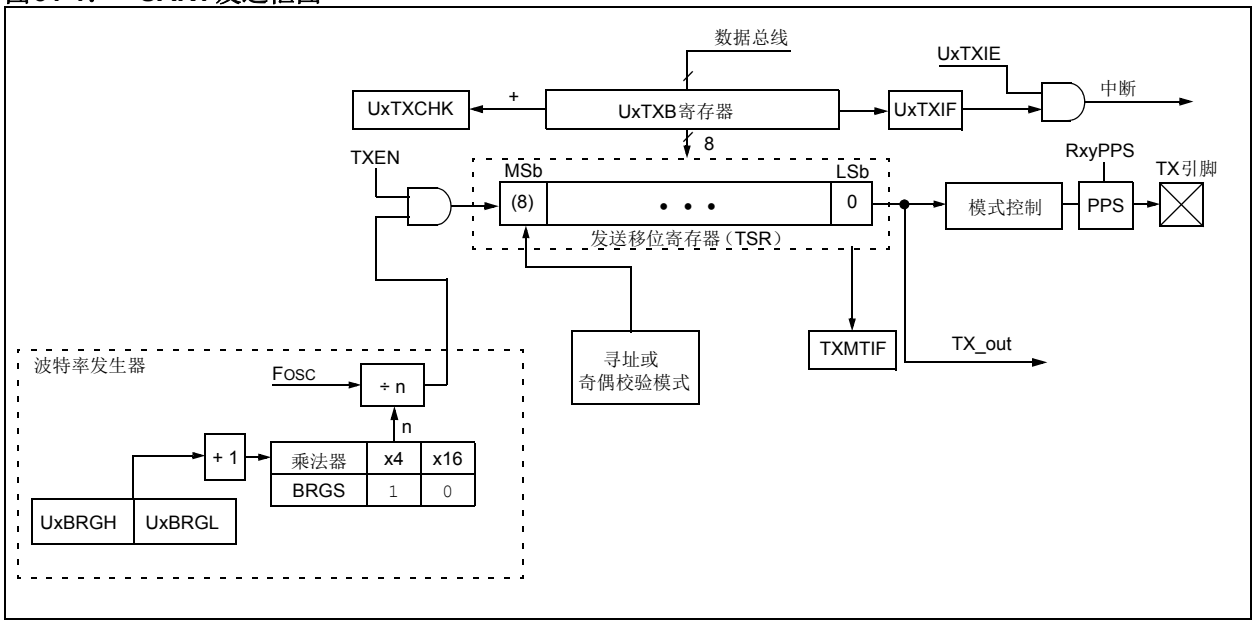
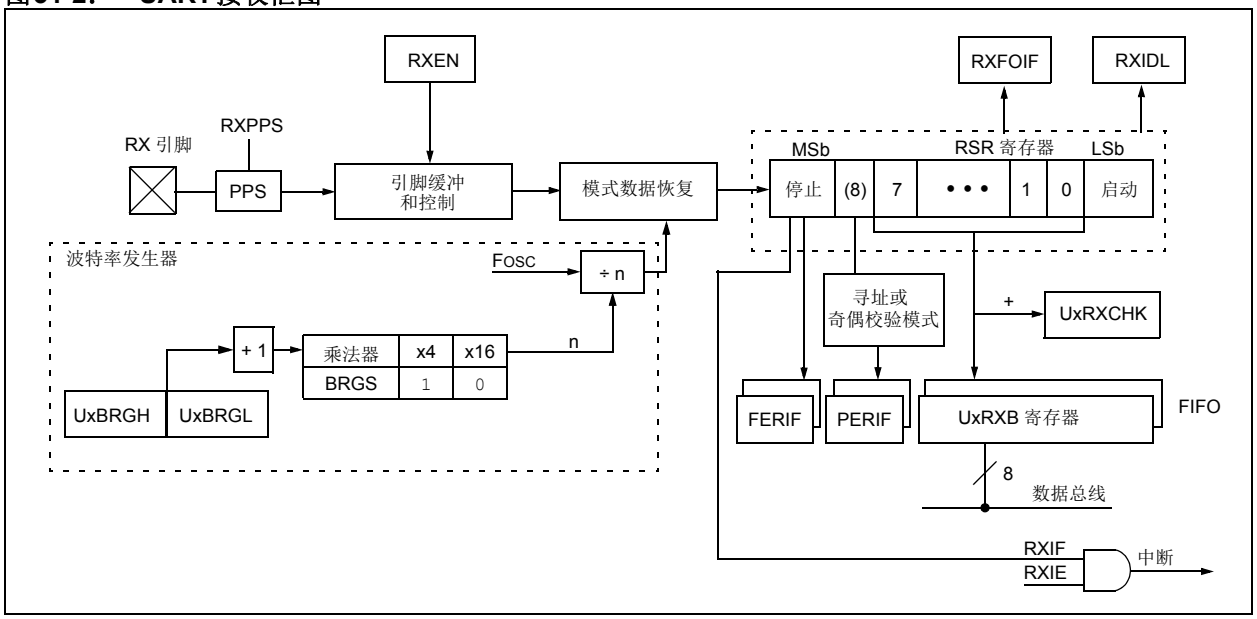


图31-2： UART接收框图



UART模块的操作由以下19个寄存器控制：

- 三个控制寄存器（UxCON0-UxCON2）
- 错误使能和状态（UxERRIE、UxERRIR和UxUIR）
- UART缓冲区状态和控制（UxFIFO）
- 三个9位协议参数（UxP1-UxP3）
- 16位波特率发生器（UxBRGH:L）
- 发送缓冲区写（UxTXB）
- 接收缓冲区读（UxRXB）
- 接收校验和（UxRXCHK）
- 发送校验和（UxTXCHK）

第31.21节“寄存器定义：UART控制”详细介绍了这些寄存器。

31.1 UART I/O引脚配置

RX输入引脚通过UxRPPS寄存器进行选择。TX输出引脚使用每个引脚的RxyPPS寄存器进行选择。当对应于TX输出的引脚的TRIS控制清零时，UART将保持TX引脚上的控制和逻辑电平。无论EN或TXEN的值如何，更改UxCON2中的TXPOL位都会立即改变TX引脚的逻辑电平。

31.2 UART异步模式

UART有五种异步模式：

- 7位
- 8位
- 8位，第9位为偶校验
- 8位，第9位为奇校验
- 8位，第9位为地址指示符

UART采用标准不归零（Non-Return-to-Zero，NRZ）格式发送和接收数据。NRZ实现为两种电平：VOH标记状态（Mark state）代表“1”数据位，而VOL空格状态（Space state）代表“0”数据位。NRZ指的是连续发送的具有相同值的数据位保持在相应位的输出电平，而不会在发送完每个位之后回到中间电平。NRZ发送端口在标记状态下为空闲。每个字符发送包含1个启动位及随后的7个或8个数据位，1个可选的奇偶校验或地址位，并始终由1个或多个停止位终止。启动位始终是一个空格，停止位始终是标记。最常用的数据格式为8位，无奇偶校验位。每个发送的位保持时间为1/(波特率)。使用片上专用16位波特率发生器从系统振荡器产生标准波特率频率。更多信息，请参见**第31.17节“UART波特率发生器（BRG）”**。

在所有异步模式中，UART首先发送和接收LSb。UART的发送器和接收器在功能上是相互独立的，但它们的数据格式和波特率相同。偶校验模式和奇校验模式通过硬件支持奇偶校验。

31.2.1 UART异步发送器

图31-1给出了UART发送器框图。发送器的核心是串行发送移位寄存器（Transmit Shift Register，TSR），该寄存器不可用软件直接访问。TSR从发送缓冲区（即UxTXB寄存器）获取数据。

31.2.1.1 使能发送器

UART发送器可通过配置以下控制位使能为异步操作：

- TXEN = 1
- MODE<3:0> = 0h至3h
- UxBRGH:L = 所需的波特率
- UxBRGS = 所需的波特率倍频值
- RxyPPS = 所需输出引脚的代码
- ON = 1

假定所有其他UART控制位均处于其默认状态。

将UxCON0寄存器中的TXEN位置1来使能UART的发送器电路。UxCON0寄存器中的MODE<3:0>位选择所需的模式。将UxCON1寄存器中的ON位置1可使能UART。TXEN置1且发送器未处于空闲状态时，TX引脚自动配置为输出。发送器空闲时，TX引脚驱动被释放到端口TRIS控制。如果TX引脚与模拟外设共用，则应通过清零相应的ANSEL位禁止模拟I/O功能。

注： TXEN使能位置1且UxTXB寄存器可以接受数据时，UxTXIF发送器中断标志位也将置1。

31.2.1.2 发送数据

向UxTXB寄存器写入一个字符时启动发送。如果这是第一个字符，或前一个字符被完全从TSR中发送，UxTXB中的数据就立即被传送到TSR寄存器。如果TSR仍包含前一个字符的全部或部分，则新字符数据保存在UxTXB中，直到前一个字符的发送完毕。然后，在前一个字符的停止位发送开始时，立即将UxTXB中待处理的字符传送到TSR。在前一个字符的全部停止位发送完毕后，立即开始启动位、数据位和停止位序列的发送。

31.2.1.3 发送数据极性

可通过UxCON2寄存器中的TXPOL位来控制发送数据的极性。该位的默认状态为0，选择高电平有效发送空闲和数据位。将TXPOL位设置为1会将发送数据的极性取反，从而选择低电平有效空闲和数据位。TXPOL位在所有模式下控制发送数据的极性。

31.2.1.4 发送中断标志

只要UART发送器被使能且UxTXB中没有等待发送的字符，PIR寄存器中的UxTXIF中断标志位就被置1。换句话说，只有在TSR正在处理字符且UxTXB中还有一个排队等待发送的新字符时，UxTXIF位才被清零。

将PIE寄存器中的UxTXIE中断允许位置1可允许UxTXIF中断。但是，只要UxTXB为空，无论UxTXIE允许位的状态如何，UxTXIF标志位都会置1。UxTXIF位是只读位，不能用软件置1或清零。

要在发送数据时使用中断，应仅在仍有数据要发送时将UxTXIE位置1。在将发送的最后一个字符写入UxTXB时清零UxTXIE中断允许位。

31.2.1.5 TSR状态

UxERRIR寄存器中的TXMTIF位指示TSR的状态。该位是只读位。TSR为空且空闲时，TXMTIF位置1。当一个字符从UxTXB传送到TSR中时，TXMTIF位清零。TXMTIF位将保持清零，直到所有位（包括停止位）移出TSR并且UxTXB寄存器中没有等待的字节。

UxERRIE寄存器中的TXMTIE位置1时，TXMTIF将会产生中断。

注： TSR不映射到数据存储器中，因此用户无法使用。

31.2.1.6 发送器7位模式

当MODE<3:0>位设置为0001时，选择7位模式。在7位模式下，仅发送写入UxTXB的数据的七个最低有效位。最高有效位被忽略。

31.2.1.7 发送器奇偶校验模式

当选择奇校验模式或偶校验模式时，所有数据都以9位发送。前8位是数据位，第9位是奇偶校验位。当MODE<3:0>位设置为0011和0010时，分别选择偶校验和奇校验。奇偶校验位由模块自动确定并插入串行数据流中。

31.2.1.8 异步发送设置

1. 初始化 UxBRGH 和 UxBRGL 寄存器对以及 BRGS 位，以获得所需的波特率（见**第 31.17 节“UART 波特率发生器（BRG）”**）。
2. 将 MODE<3:0> 位设置为所需的异步模式。
3. 如果需要将 TX 输出反相，将 TXPOL 位置 1。
4. 将 ON 位置 1，使能异步串口。
5. 将 TXEN 控制位置 1，使能发送器。这将导致 UxTXIF 中断标志位置 1。
6. 如果器件有 PPS，使用 TX 输出的代码配置所需的 I/O 引脚 RxyPPS 寄存器。
7. 如果需要中断，将相应 PIE 寄存器中的 UxTXIE 中断允许位置 1。如果 INTCON0 寄存器中的 GIE 位也置 1，则立即产生中断。
8. 向 UxTXB 寄存器写入一个字节的数据。这将启动发送。
9. 当 UxTXIF 位为 1 时，可能会写入后续字节。

图 31-3： 异步发送

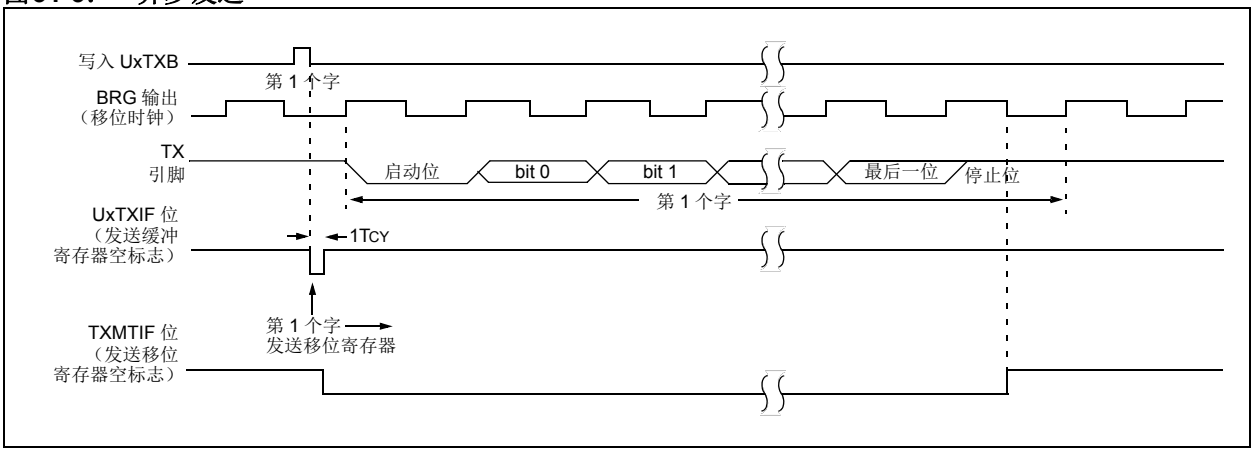
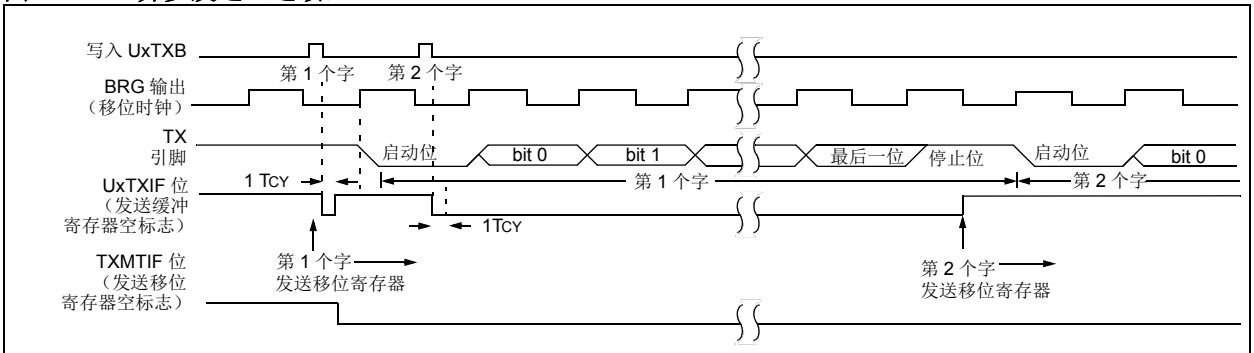


图 31-4： 异步发送（连续）



注： 这张时序图显示了第一个发送和第二个连续发送的开始。

31.2.2 UART异步接收器

异步模式通常用于RS-232系统中。图31-2给出了接收器框图。数据通过RX引脚接收并驱动数据恢复模块。数据恢复模块实际上是一个高速移位器，工作频率为4或16倍波特率，而串行接收移位寄存器（Receive Shift Register，RSR）的工作频率为比特率。字符的所有位移入后被立即传送到双字符的先进先出（First-In-First-Out，FIFO）存储区中。软件开始处理UART接收器前，FIFO缓冲区允许先接收两个完整字符和第三个字符的开始部分。FIFO寄存器和RSR不能直接用软件访问。通过UxRXB寄存器访问接收数据。

31.2.2.1 使能接收器

UART接收器可通过配置以下控制位使能为异步操作：

- RXEN = 1
- MODE<3:0> = 0h至3h
- UxBRGH:L = 所需的波特率
- RXPPS = 所需输入引脚的代码
- 输入引脚ANSEL位 = 0
- ON = 1

假定所有其他UART控制位均处于其默认状态。

将UxCON0寄存器的RXEN位置1来使能UART的接收器电路。通过将UxCON0寄存器中的MODE<3:0>位置1，可将UART配置为所需的异步模式。将UxCON1寄存器中的ON位置1可使能UART。必须将所选RX I/O引脚对应的TRIS位置1，以将该引脚配置为输入。

注：	如果RX功能位于模拟引脚上，则必须清零相应的ANSEL位使接收器工作。

31.2.2.2 接收数据

通过定时到位中心并对输入电平进行采样，从比特流中恢复数据。在高速模式下，每位有4个BRG时钟，每位只进行一次采样。在正常速度模式下，每位有16个BRG时钟，每位进行三次采样。

接收器的数据恢复电路在启动位的下降沿启动字符接收。启动位，始终为0。启动位限定在位的中间位置。仅在正常速度模式中时，启动位也限定在位的上升沿。以下段落介绍了正常速度模式的大多数检测采样。

下降沿启动波特率发生器（Baud Rate Generator，BRG）时钟。在第一个和第二个BRG时钟处对输入进行采样。

如果两次采样均为高电平，则下降沿被视为毛刺，UART返回到启动位检测状态而不产生错误。

如果任一次采样为低电平，则数据恢复电路继续对BRG时钟进行计数，并在时钟计数7、8和9处进行采样。当少于两次采样为低电平时，启动位被视为无效，数据恢复电路中止字符接收，不产生错误，并恢复寻找启动位的下降沿。

当两次或多次采样为低电平时，启动位被视为有效，数据恢复继续。检测到有效启动位后，BRG时钟计数器继续并在计数16处复位。这是第一个数据位的开始。

数据恢复电路从该位开始时对BRG时钟进行计数，并在时钟7、8和9处进行采样。位值由大多数采样确定。产生的0或1被移入RSR中。BRG时钟计数器继续并在计数16处复位。该序列重复此过程直到所有数据位均被采样并移入RSR。

移入所有数据位后，对第一个停止位进行采样。停止位，始终为1。如果位采样确定停止位处为0，则置1此字符的帧错误标志位。否则，清零此字符的帧错误标志位。关于帧错误的更多信息，请参见**第31.2.2.4节“接收帧错误”**。

31.2.2.3 接收中断

所有数据位和停止位被接收后，RSR中的字符就被立即传送到UART接收FIFO。如果相应PIR寄存器中的UxRXIF中断标志位未受到抑制，则将在此时置1。

UxRXIF由以下任意一位抑制：

- FERIF，如果FERIE置1
- PERIF，如果PERIE置1

这将暂停DMA数据传输，直到软件处理错误并读取UxRXB以使FIFO越过错误。

将以下所有位置1可允许UxRXIF中断：

- PIE寄存器中的中断允许位UxRXIE
- INTCON0寄存器中的全局中断允许位GIE

当UxRXIF中断标志位未受到抑制且FIFO中存在未被读取的字符时，无论中断允许位的状态如何，均会被置1。读取UxRXB寄存器时，会将顶部字符送出FIFO，使FIFO内容减少一个字符。UxRXIF中断标志位是只读位，不能用软件置1或清零。

31.2.2.4 接收帧错误

接收FIFO缓冲区中的每个字符都有相应的帧错误标志位。帧错误表明在预期时间内未见到停止位。通过UxERRIR寄存器中的FERIF位可访问帧错误标志。FERIF位表示接收FIFO顶部未读取字符的帧状态。因此，在读UxRXB之前必须先读FERIF位。

FERIF位是只读位，只适用于接收FIFO的顶部未读取字符。帧错误（FERIF = 1）并不会禁止接收更多字符。既不必要也不可能将FERIF位直接清零。从FIFO缓冲区读出下一个字符将使FIFO转到下一个字符和下一个相应的帧错误。

当FIFO顶部的字符不具有帧错误或接收FIFO中的所有字节都被读取时，FERIF位清零。清零ON位将复位接收FIFO，进而会清零FERIF位。

UxERRIE寄存器中的FERIE位置1时，帧错误将产生摘要UxERR中断。当FIFO顶部的FERIF位为0或检索到所有FIFO字符时，将复位摘要错误。

如果FERIE置1，则当FERIF为1时，UxRXIF中断被抑制。

31.2.2.5 接收器奇偶校验模式

当MODE<3:0>位设置为0011和0010时，分别自动检测偶校验和奇校验。奇偶校验模式的每个字符接收8个数据位和1个奇偶校验位，总共9位。UxERRIR寄存器中的PERIF位表示接收FIFO的顶部未读取字符的奇偶校验错误，而非奇偶校验位本身。在读取UxRXB寄存器使FIFO递增之前，必须读取奇偶校验错误。

UxERRIE寄存器中的PERIE位置1时，奇偶检验错误将产生摘要UxERR中断。当FIFO顶部的PERIF位为0或检索到所有FIFO字符时，将复位摘要错误。

如果PERIE置1，则当PERIF为1时，UxRXIF中断被抑制。

31.2.2.6 接收FIFO溢出

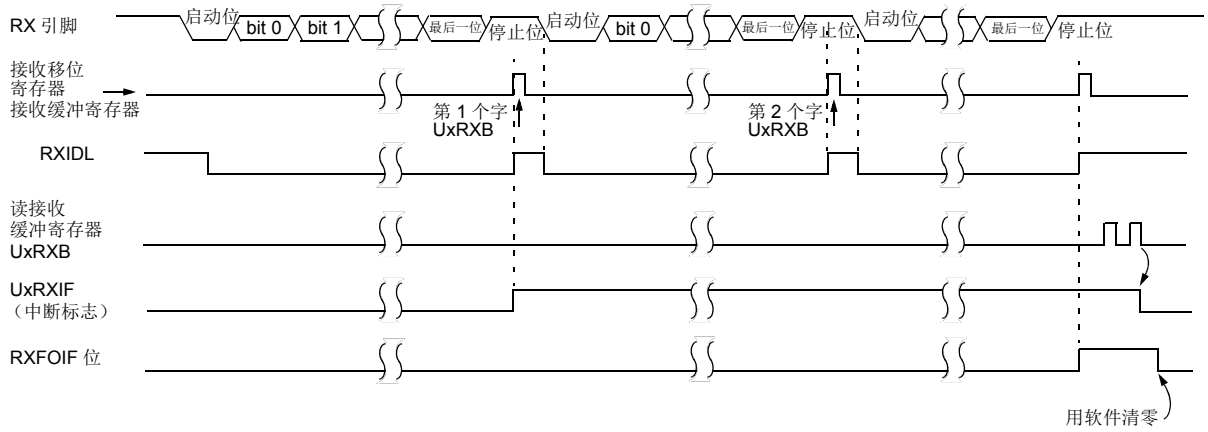
当接收到的字符超过接收FIFO可容纳的字符时，UxERRIR寄存器中的RXFOIF位置1。引起溢出条件的字符会被丢弃。UxCON2寄存器中的RUNOVF位决定溢出条件持续时接收电路如何对字符作出响应。当RUNOVF置1时，接收移位器通过响应启动位、数据位和停止位与输入数据流保持同步。但是，将丢弃FIFO中尚不存在的所有接收字节。当RUNOVF清零时，接收移位器停止操作，并忽略启动位、数据位和停止位。通过读取UxRXB寄存器并清零RXFOIF位来清除接收溢出条件。如果未读取UxRXB寄存器以在FIFO中打开空间，则接收的下一个字符将被丢弃并导致另一个溢出条件。

UxERRIE寄存器中的RXFOIE位置1时，接收溢出错误将产生摘要UxEIF中断。

31.2.2.7 异步接收设置

1. 初始化UxBRGH和UxBRGL寄存器对以及BRGS位，以获得所需的波特率（见**第31.17节“UART波特率发生器（BRG）”**）。
2. 为所需的RX引脚配置RXPPS寄存器
3. 清零RX引脚的ANSEL位（如适用）。
4. 将MODE<3:0>位设置为所需的异步模式。
5. 如果数据流反相，则将RXPOL位置1。
6. 将ON位置1来使能串口。
7. 如果需要中断，将PIEx寄存器中的UxRXIE位和INTCON0寄存器中的GIE位置1。
8. 将RXEN位置1来使能接收。
9. 当字符从RSR被移入接收缓冲区时，UxRXIF中断标志位将被置1。如果UxRXIE中断允许位也置1，则产生中断。
10. 读取UxERRIR寄存器，取得错误标志。
11. 读取UxRXB寄存器，取得接收的字节。
12. 如果发生溢出，则清零RXFOIF位。

图31-5： 异步接收



注： 此时序图显示了在RX输入引脚上连续接收的3个字。在接收第3个字后读UxRXB（接收缓冲区）会使RXFOIF（FIFO溢出）位置1。STPMD = 0，STP<1:0> = 00。

31.3 异步寻址模式

当多个接收器共用同一条传输线时（如在RS-485系统中），有一个特殊的地址检测模式可供使用。

当使能异步寻址模式时，所有数据都以9位字符的形式发送和接收。第9个位确定字符是地址还是数据。当第9个位置1时，8个最低有效位为地址。当第9个位清零时，最低有效位为数据。在任何一种情况下，当字节写入接收FIFO时，第9个位存储在PERIF中。当PERIE也置1时，RXIF将被抑制，从而暂停DMA传输，允许软件处理接收到的地址。

地址字符将使能与地址匹配的所有接收器并禁止所有其他接收器。使能接收器后，将接收所有非地址字符，直至接收到不匹配的地址字符。

31.3.1 地址模式发送

UART发送器可通过配置以下控制位使能为异步地址操作：

- TXEN = 1
- MODE<3:0> = 0100
- UxBRGH:L = 所需的波特率
- RxyPPS = 所需输出引脚的代码
- ON = 1

通过写入UxP1L寄存器发送地址。第9个位置1，表示该字节是地址时，将发送写入字节。

通过写入UxTXB寄存器发送数据。第9个位清零，表示该字节是数据时，将发送写入字节。

要将数据发送到传输总线上的特定器件，首先发送目标器件的地址。所有后续数据仅由该器件接受，直到发送另一器件的地址。

写入UxP1L优先于写入UxTXB。如果在TSR忙时写入UxP1L和UxTXB寄存器，要发送的下一个字节将来自UxP1L。

为确保在更改地址之前发送用于某一器件的所有数据，请等待 TXMTIF 位为高电平，然后再将新地址写入UxP1L。

31.3.2 地址模式接收

UART接收器可通过配置以下控制位使能为异步地址操作：

- RXEN = 1
- MODE<3:0> = 0100
- UxBRGH:L = 所需的波特率
- RXPPS = 所需输入引脚的代码
- 输入引脚ANSEL位 = 0
- UxP2L = 接收器地址
- UxP3L = 地址掩码
- ON = 1

在地址模式下，在接收到有效地址之前，不会将数据传输到输入FIFO。这是默认状态。以下任何条件都将导致UART恢复为默认状态：

- ON = 0
- RXEN = 0
- 接收到的地址不匹配

当接收到第9个位置1的字符时，该字符的低8位将由UxP2L和UxP3L寄存器中的值限定。

该字节先与UxP2L进行异或运算，然后与UxP3L进行与运算。当结果为0h时发生匹配，在这种情况下，未更改的接收字符存储在接收FIFO中，从而将UxRXIF中断位置1。第9个位存储在相应的PERIF位中，将该字节标识为地址。

地址匹配还会使能所有数据的接收器，这样，第9个位未置1的所有后续字符都将存储在接收FIFO中。

当第9个位置1且未发生匹配时，该字符不会存储在接收FIFO中，并且所有后续数据都将被忽略。

UxP3L寄存器掩码支持接受某个地址范围。然后，软件可以通过处理接收的地址字符来确定范围的子地址。

31.4 DMX模式

DMX是舞台和演出设备使用的协议。它包括照明、烟雾机和电机等。该协议包括一个发出命令的控制器以及接收这些命令的接收器（如剧场灯）。DMX协议通常是单向的，但在半双工或全双工模式下可变为双向协议。半双工模式的一个示例是位于DMX512A上的RDM（远程设备管理）协议。控制器发送命令，接收器接收命令。此外，没有错误条件或重复发送机制。

DMX，又称DMX512A，包括由512个通道组成的“世界”。这意味着一个控制器可以在单个DMX链路上输出最多512个字节。经过编程，线路上的每个设备可以监听这些字节中的一个或多个组成的连续序列。

例如，连接到其中一个“世界”的烟雾机可编程为接收从字节编号10开始的一个字节，照明单元可编程为接收从字节编号22开始的四个字节。

31.4.1 DMX控制器

通过下列设置来配置DMX控制器模式：

- MODE<3:0> = 1010
- TXEN = 1
- RXEN = 0
- TXPOL = 1
- UxP1 = 比要发送的字节数小1（不包括起始码）
- UxBRGH:L = 实现250K波特率的值
- STP<1:0> = 10（2个停止位）
- RxyPPS = TX引脚输出编码
- ON = 1

每个DMX发送都以一个间隔开始，后跟一个称为“起始码”的字节。间隔的宽度固定为25个位时间。间隔之后是“间隔后标记”（Mark After Break，MAB）空闲周期。在此空闲周期之后，发送第1个到第n个字节，其中“n-1”是UxP1中的值。请参见图31-6。

软件通过将要按所需顺序发送的每个字节写入UxTXB寄存器来发送起始码和“n”个数据字节。UxTXIF值为1表示UxTXB已准备好接受下一个字节。

软件无法访问内部字节计数器。软件需要跟踪写入UxTXB的字节数，以确保发送的字节数等于“n”字节，因为DMX状态机将在写入“n”个字节后自动插入间隔并复位其内部计数器。确保硬件和软件之间同步的一种方法是在“世界”的最后一个字节完全移出发送移位寄存器之后翻转TXEN，如TXMTIF位所示。

31.4.2 DMX接收器

通过下列设置来配置DMX接收器模式：

- MODE<3:0> = 1010
- TXEN = 0
- RXEN = 1
- RXPOL = 1
- UxP2 = 要接收的第一个字节的数量
- UxP3 = 要接收的最后一个字节的数量
- UxBRGH:L = 实现250K波特率的值
- STP<1:0> = 10（2个停止位）
- ON = 1
- UxRXPPS = 所需输入引脚的代码
- 输入引脚ANSEL位 = 0

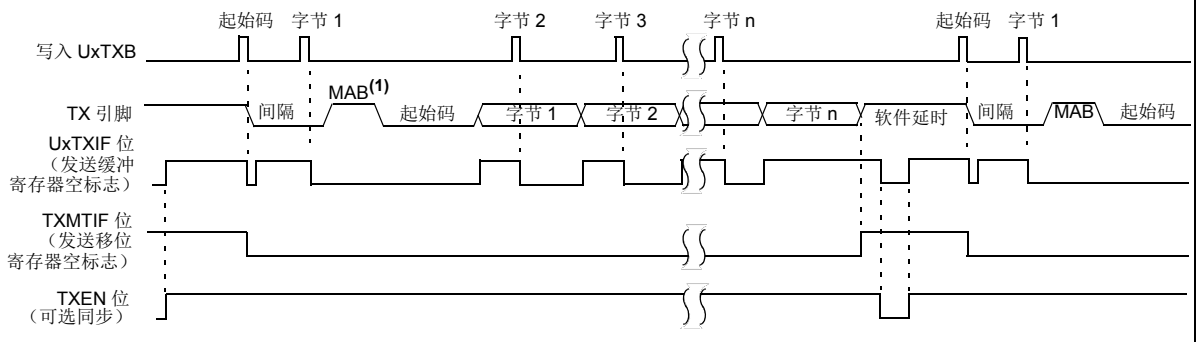
配置为DMX接收器时，UART监听至少23个位周期宽的间隔字符。如果间隔短于23个位时间，该间隔将被忽略，DMX 状态机保持空闲模式。接收到间隔后，DMX计数器将复位以与输入数据流一致。间隔后，UART将立即看到“间隔后标记”（MAB）。UART会忽略此空间。起始码遵循MAB，并始终存储在接收FIFO中。

在起始码之后，将接收到第1个至第512个字节，但并非所有字节都存储在接收FIFO中。UART会忽略接收的所有字节，直到接收到所需的字节。这可通过UxP2和UxP3寄存器来实现。UxP2寄存器保存启动接收过程的字节编号值，UxP3寄存器保存结束接收过程的字节编号值。字节计数器从0开始（起始码后的第一个字节）。例如，要从起始码后的第10个字节开始接收4个字节，将009h（十进制为9）写入UxP2H:L，并将00Ch（十进制为12）写入UxP3H:L。接收FIFO只有2个字节深，因此必须通过读取UxRXB来检索字节，因为它们可以避免接收FIFO溢出条件。

通常在字节之间插入两个停止位。如果任一停止位被检测为0，则该字节的帧错误将被置1。

由于DMX序列始终以间隔开始，因此软件可以通过监视RXBKIF标志来验证它是否与序列同步，以确保将在RXBKIF后接收的下一个字节作为起始码，并将后续字节作为预期数据。

图31-6： DMX发送序列



注1： MAB周期固定为3位周期。

31.5 LIN模式

LIN是一种主要用于汽车应用的协议。LIN网络由两类软件进程组成：主进程和从进程。每个网络只有一个主进程和一个或多个从进程。

从物理层的角度来看，一个处理器上的UART可以由主进程和从进程驱动，只要网络上只存在一个主进程即可。

LIN事务由主进程和从进程组成。从进程可能包括多个从器件，其中一个正在发送而其余正在接收。事务由以下主进程发送序列开始：

1. 间隔
2. 分隔符位
3. 同步字段
4. PID字节

PID确定哪些从进程将响应主器件。当PID字节完成时，TX输出保持在空闲状态。可能有一个或多个从进程响应主进程。如果在字节间周期内没有从进程响应，则主进程可以自由地开始另一次发送。字节间周期由软件使用UART以外的方式定时。

从进程遵循主进程。从软件识别出PID后，该从进程通过发送所需的响应或通过接收发送的数据来响应。只有从进程发送数据。因此，接收数据的从进程接收的是另一个从进程的数据。

当从进程发送数据时，从UART会在字节发送后自动计算这些字节的校验和，并将取反后的校验和字节附加到从响应中。

当从器件接收数据时，将在接收到每个字节时对相应字节的校验和进行累加，校验和的算法与发送过程相同。最后一个字节（即通过发送过程计算的校验和取反后的结果）将与通过UART本地计算的校验和相加。当结果全部为1时，校验通过，否则校验失败，此时CERIF位置1。

有两种方法可用于计算校验和：传统方法和增强型方法。传统校验和仅包括数据字节。增强型校验和包括PID和数据。UxCON2寄存器中的C0EN控制位决定校验和方法。将C0EN设置为1将选择增强型方法。软件必须在接收到校验和字节的启动位之前选择适当的方法。

31.5.1 LIN主/从模式

LIN主模式包括生成从进程的功能。主进程在PID发送时停止。在主/从模式下发送的任何数据都是作为从进程完成的。通过下列设置来配置LIN主/从模式：

- MODE<3:0> = 1100
- TXEN = 1
- RXEN = 1
- UxBRGH:L = 实现所需波特率的值
- TXPOL = 0（用于高空闲状态）
- STP = 所需的停止位选择
- C0EN = 所需的校验和模式
- RxyPPS = TX引脚选择代码
- TX引脚TRIS控制 = 0
- ON = 1

注： TXEN位必须在接收到主进程之前置1，并在LIN模式下保持置1，无论从进程是否为发送器均如此。

当UxP2为0且UART空闲时，通过将PID写入UxP1L寄存器来启动主进程。在这种情况下，UxTXIF不会置1。在PID发送中仅使用UxP1L的六个最低有效位。

发送PID的两个最高有效位是PID奇偶校验位。PID<6>是PID位0、1、2和4的异或结果。PID<7>是PID位1、3、4和5的异或结果的取反值。

UART计算并在串行流中插入这些位。

写入UxP1L会自动清零UxTXCHK和UxRXCHK寄存器，并生成事务的间隔、分隔符位、同步字符（55h）和PID发送部分。随后事务的数据部分（如果存在）是从进程。关于该进程的更多详细信息，请参见**第31.5.2节“LIN从模式”**。当RXEN置1时，主进程接收自己的PID。软件执行与发送和接收的PID相对应的从进程。在活动主进程完成之前尝试写入UxP1L将不会成功。TXWRE位将置1。

31.5.2 LIN从模式

通过下列设置来配置LIN从模式：

- MODE<3:0> = 1011
- TXEN = 1
- RXEN = 1
- UxP2 = 要发送的数据字节数
- UxP3 = 要接收的数据字节数
- UxBRGH:L = 实现默认波特率的值
- TXPOL = 0（用于高空闲状态）
- STP = 所需的停止位选择
- C0EN = 所需的校验和模式
- RxyPPS = TX引脚选择代码
- TX引脚TRIS控制 = 0
- ON = 1

从进程在检测到RX引脚上的间隔时启动。间隔清零UxTXCHK、UxRXCHK、UxP2和UxP3寄存器。在间隔结束时，使用间隔之后的同步字符激活自动波特率电路，并自动设置波特率。同步字符后面的字符作为PID代码接收并保存在接收FIFO中。UART根据PID的六个最低有效位计算两个PID奇偶校验位。如果任一奇偶校验位与接收到的PID代码的相应位不匹配，则PERIF标志置1并保存在与PID代码相同的FIFO位置。UxRXIF位置1，表示PID可用。

软件通过读取UxRXB寄存器来检索PID，并确定要从中执行的从进程。校验和方法、数据字节数以及是否发送或接收数据均由软件根据PID代码定义。

31.5.2.1 LIN从接收器

当从进程是接收器时，软件执行以下任务：

- 将大小等于要接收的数据字节数的值写入UxP3寄存器。
- C0EN位置1或清零以选择适当的校验和。这必须在接收到校验和字节的启动位之前完成。
- 当UxRXIF置1时，从UxRXB读取进程响应的每个字节。

UART更新每个接收字节的校验和。当接收到最后一个数据字节时，计算出的校验和总数存储在UxRXCHK寄存器中。下一个接收的字节保存在接收FIFO中，并与UxRXCHK中的值相加。相加的结果不可访问。但是，如果结果不全为1，则将UxERRIR中的CERIF位置1。CERIF标志持续存在，直到用软件清零。软件需要读取UxRXB以从FIFO中删除校验和字节，但如果不需要用于任何其他目的，则可以丢弃该字节。

接收到校验和后，UART忽略RX引脚上的所有活动，直到间隔开始下一个事务。

31.5.2.2 LIN从发送器

当从进程是发送器时，软件按所示顺序执行以下任务：

- 将大小等于要发送的字节数的值写入UxP2寄存器。这将使能TXIF标志，当UxP2为0时，将禁止该标志。
- C0EN位置1或清零以选择适当的校验和
- 执行字节间延时
- 当UxTXIF置1时，进程响应的每个字节都写入UxTXB

当向UxTXB写入每个字节时，UART累加校验和。写入最后一个字节后，UART将计算出的校验和存储在UxTXCHK寄存器中，并将相反的结果作为响应中的最后一个字节发送。

写入UxP2字节时，禁止TXIF标志。写入UxTXB时，超出UxP2计数的任何写操作都将被忽略，并且会将UxFIFO寄存器中的TXWRE标志置1。

31.6 DALI模式

DALI是一种用于楼宇自动化智能照明控制的协议。该协议由“控制器件”和“控制装置”组成。控制器件是向照明装置发出命令的应用控制器。照明装置本身被称为控制装置。使用UART硬件执行的曼彻斯特编码完成通信。

曼彻斯特编码由单个比特流中的时钟和数据组成。高电平与低电平之间的跳变始终发生在位周期的中间，无法保证发生在位周期的边界。

当比特流中的连续位具有相同的值（即，连续的1或连续的0）时，在位边界处发生跳变。但是，当位值改变时，在位边界处没有跳变。根据该标准，半位时间通常为416.7 µs。双半位或单个位的时间通常为833.3 µs。

该协议本质上是半双工协议。总线上的通信以前向和后向帧的形式发生。帧之间的等待时间在标准中定义，旨在防止帧之间发生冲突。

控制器件发送被称为“前向帧”。在DALI 2.0标准中，前向帧的长度可以为两个或三个字节。两个字节的前向帧用于控制器件和控制装置之间的通信，而三个字节的前向帧用于总线上控制器件之间的通信。前向帧中的第一个字节是控制字节，后跟一个或两个数据字节。当控制器件开始发送时，事务开始。与其他协议不同，帧中的每个字节首先发送MSB。典型帧时序如图31-8中所示。

在两个控制器件之间的通信期间，需要发送三个字节。在这种情况下，在UxTXIF变为真后、输出移位器变为空前，软件必须立即将第三个字节写入UxTXB。这确保连续发送前向帧的三个字节，而没有任何中断。

总线上的所有控制装置都接收前向帧。如果前向帧要求发送回复，则其中一个控制装置可以用单个字节（称为“后向帧”）响应。2.0标准要求控制装置在接收前向帧之后的5.5 ms到10.5 ms（约14到22个半位时间）之间开始发送后向帧。控制器件接收到后向帧后，则需要等待至少2.4 ms（约半位时间的6倍）。在此等待时间之后，控制器件可以自由发送另一个前向帧（见图31-9）。

启动位用于指示前向帧和后向帧的开始。当ABDEN = 0时，接收器比特率由BRG寄存器确定。当ABDEN = 1时，第一个位将接收器与发射器同步并设置接收器比特率。测量启动位的低电平周期，并将其用作前向帧和后向帧中所有数据位的时序参考。如果启动位的低电平周期导致测量计数器溢出，则ABDOVF位置1。启动位之后的所有位都是数据位。当在位周期的中间没有检测到跳变时，比特流终止（见图31-7）。

前向帧和后向帧由两个空闲位周期或停止位终止。通常，它们从一个字节的第一个位周期开始。如果两个停止位均有效，则终止字节接收。

如果任一停止位无效，则通过将其保存为空字节并将接收FIFO中的帧错误置1来将帧标记为无效。

当字节接收未完成时，如果在位周期的中间没有检测到总线上的跳变，也会发生帧错误。在这种情况下，字节将与FERIF位一起保存。

31.6.1 控制器件

通过下列设置来配置控制器件模式：

- MODE<3:0> = 1000
- TXEN = 1
- RXEN = 1
- UxP1 = 在前向帧或后向帧完成后，前向帧等待发送的半位周期数。
- UxP2 = 前向/后向帧阈值定界符。在前向帧或后向帧完成后开始该半位周期数的任何接收都会被检测为前向帧，并且会将相应接收字节的PERIF标志置1。
- UxBRGH:L = 实现1200波特率的值
- TXPOL = 接口电路的适当极性
- STP<1:0> = 10（两个停止位）
- RxyPPS = TX引脚选择代码
- TX引脚TRIS控制 = 0
- ON = 1

向UxTXB寄存器写入控制字节时启动前向帧。UxTXIF变为真后，必须将控制字节后的每个数据字节立即写入UxTXB寄存器。在UxTXIF变为真之后，必须执行每次写操作，以确保发送缓冲区已准备好接受该字节。此外，每次写操作还必须在TXMTIF位变为真之前发生，以确保在不中断的情况下生成前向帧的比特流。

当TXMTIF变为真时，表示发送移位寄存器已完成发送帧中的最后一个字节，TX输出将保持在空闲状态，持续的半位周期数由UxCON2寄存器中的STP位选择。

在最后一个停止位之后，TX输出将在由UxP1寄存器中的半位周期数确定的额外等待时间内保持空闲状态。例如，2450 µs的延时（约为6个半位时间）要求UxP1L中的值为6。

在TXMTIF变为真，但在UxP1等待时间之前发生的对UxTXB寄存器的任何写操作将被保持，然后在等待时间后立即发送。如果在等待时间内接收到后向帧，则在完成后向帧接收并经过后向帧加UxP1等待时间后，将发送可能已写入UxTXB的任何字节。

等待定时器由后向帧复位，并在后向帧的停止位之后立即重新开始。经过等待时间后，将发送发送移位寄存器中的待处理数据。

要替换或删除任何待处理的前向帧数据，需要将TXBE位置1以刷新移位寄存器和发送缓冲区，然后将新的控制字节写入UxTXB寄存器。新的控制字节将保存在缓冲区中，并在UxP1等待时间后作为下一个前向帧的开始发送。

在控制器件模式下，PERIF在接收到前向帧时置1。这可帮助软件区分接收到的字节是来自控制器件（来自指定的控制器件或来自总线上的另一个控制器件）的前向帧还是来自控制装置的后向帧。

31.6.2 控制装置

使用下列设置来配置控制装置模式：

- MODE<3:0> = 1001
- TXEN = 1
- RXEN = 1
- UxP1 = 在前向帧完成后，后向帧等待发送的半位周期数。
- UxP2 = 前向/后向帧阈值定界符。超过该半位周期数的空闲周期被检测为前向帧。
- UxBRGH:L = 实现1200波特率的值
- TXPOL = 接口电路的适当极性
- RXPOL = 与TXPOL相同
- STP = 10（两个停止位）
- RxyPPS = TX引脚输出编码
- TX引脚TRIS控制 = 0
- RXPPS = RX引脚选择代码
- RX引脚TRIS控制 = 1
- 输入引脚ANSEL位 = 0
- ON = 1

进入控制装置模式后，UART开始监听前向帧。只有在经过长于UxP2个半位周期的空闲周期后，帧才会被检测为前向帧。来自其他控制装置的后向帧将被忽略。只有前向帧会存储在UxRXB中。必须这样操作，因为后向帧只能作为对前向帧的响应发送。

正向帧可在接收FIFO中一次接收一个字节，并通过读取UxRXB寄存器来获取。前向帧结束时，会启动定时器来使后向帧响应延迟一段等待时间，时长等于存储在UxP1中的半位周期数。在前向帧中接收的数据由应用程序软件处理。如果应用程序决定发送后向帧作为对前向帧的响应，则将后向帧的值写入UxTXB。该值在发送移位寄存器中保持待发送状态，直到等待时间到期，然后立即发送。

如果在等待时间到期后将后向帧数据写入UxTXB，则在经过下一个正向帧后的等待时间前，数据将一直保存在UxTXB寄存器中。当后向帧数据保存在发送移位寄存器中时，TXMTIF位为假。在TXMTIF变为真之前接收UxRXIF中断表示后向帧写入太晚，在发送后向帧之前接收到另一个前向帧。必须通过将TXBE位置1来刷新待处理的后向帧，以避免在下一个前向帧之后发送该后向帧。

图31-7： 曼彻斯特时序

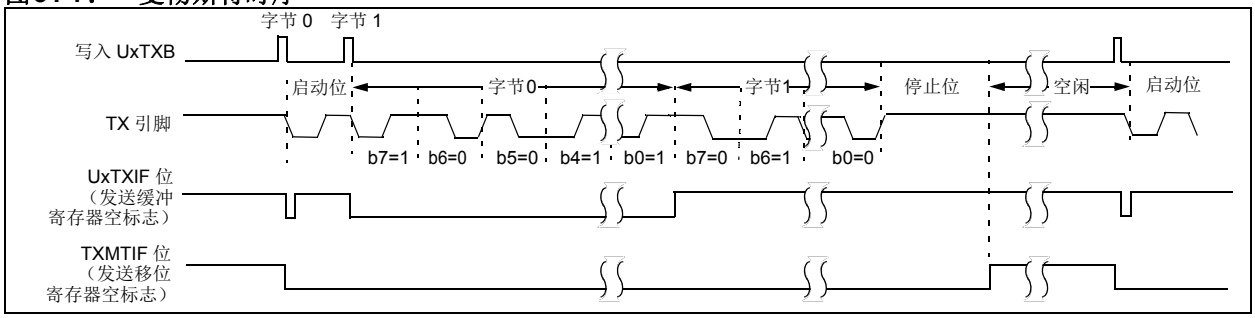


图31-8： DALI帧时序

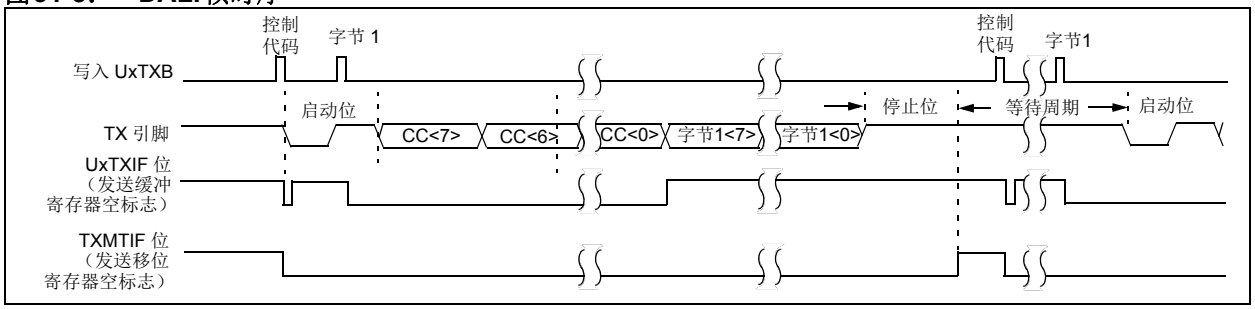
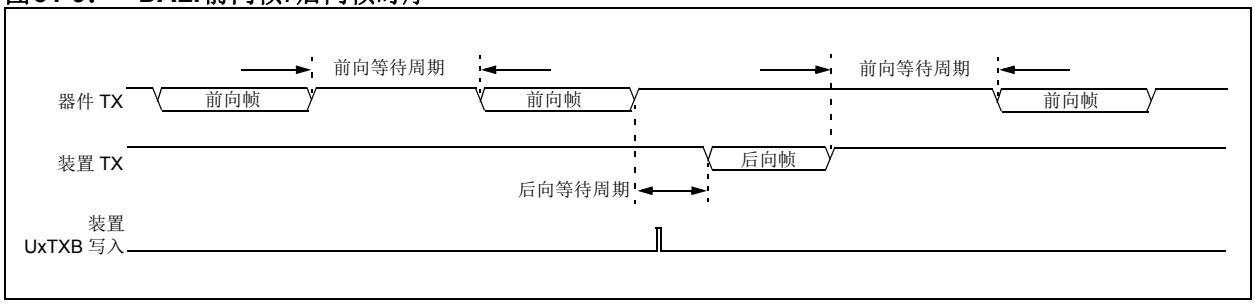


图31-9： DALI前向帧/后向帧时序



31.7 通用曼彻斯特

通用曼彻斯特是DALI模式的一个子集。当UxP1L寄存器清零时，帧之间没有最小等待时间。这可以实现全双工和半双工操作，因为写入UxTXB不会暂停来等待接收操作完成。

通用曼彻斯特操作维护DALI模式的所有其他方面，例如：

- 单脉冲启动位
- 最高有效位优先
- 连续字节之间无停止周期

通过下列设置来配置通用曼彻斯特模式：

- MODE<3:0> = 1000
- TXEN = 1
- RXEN = 1
- UxP1 = 0h
- UxBRGH:L = 所需的波特率
- TXPOL和RXPOL = 所需的空闲状态
- STP = 所需的停止周期数
- RxyPPS = TX引脚选择代码
- TX引脚TRIS控制 = 0
- RXPPS = RX引脚选择代码
- RX引脚TRIS控制 = 1
- 输入引脚ANSEL位 = 0
- ON = 1

图31-7给出了曼彻斯特比特流时序。

31.8 极性

用户可选择接收和发送极性，极性会影响所有操作模式。

空闲电平可通过UxCON2寄存器中的极性控制位进行编程。控制位默认为0，选择高空闲电平。通过将控制位设置为1来选择低电平空闲状态。TXPOL控制TX空闲电平。RXPOL控制RX空闲电平。

31.9 停止位

用户可通过UxCON2寄存器中的STP位选择停止位的数量。STP位会影响所有操作模式。

停止位选择包括：

- 发送1个停止位，对第1个进行接收验证
- 发送1.5个停止位，对第1个进行接收验证
- 发送2个停止位，对全部2个进行接收验证
- 发送2个停止位，仅对第1个进行接收验证

在除DALI以外的所有模式中，发送器在连续发送的各个字之间处于空闲状态，持续时间为停止位周期数。在DALI中，停止位在所发送数据流中的最后一位之后生成。

在第一个停止位的中间（选择对第1个停止位进行接收验证时），以及第二个停止位的中间（选择对全部2个停止位进行接收验证时），检查输入的空闲电平。如果任何停止位验证指示非空闲电平，则将接收字帧错误FERIF位置1。

31.9.1 延迟UXRXIF

当工作在半双工模式下时，单片机需要在接收后反转收发器方向，在停止位结束之前延迟UxRXIF中断来避免线路争用可能更方便。用户通过UxFIFO寄存器中的STPMD位选择何时发生UxRXIF中断。当STPMD为1时，在最后一个停止位结束时发生UxRXIF。当STPMD为0时，在接收字节存储在接收FIFO中时发生UxRXIF。当STP<1:0> = 10时，在第二个停止位的中间执行存储操作，否则，在第一个停止位的中间执行存储操作。STPMD不会延迟FERIF和PERIF中断。当STPMD置1时，只会延迟UxRXIF，并且应作为反转收发器方向的惟一指示。

31.10 FIFO溢出后的操作

接收移位寄存器（RSR）可配置为在接收FIFO溢出条件下停止或继续运行。停止操作是传统模式。

当RSR在溢出条件下继续运行时，清除溢出后接收到的第一个字将始终有效。

当RSR在溢出条件下停止时，与启动位的同步将丢失。因此，清除溢出之后接收到的第一个字可以从一个字的中间开始。

溢出期间的操作使用UxCON2寄存器中的RUNOVF位进行选择。将RUNOVF位置1来选择溢出期间的运行方法。

31.11 接收和发送缓冲区

UART使用小缓冲区来发送和接收数据。它们有时被称为FIFO。

接收器有一个接收移位寄存器（RSR）和两个缓冲寄存器。通过读取UxRXB寄存器来检索FIFO顶部的缓冲区（最早进入FIFO的字节）。

发送器有一个发送移位寄存器（TSR）和一个缓冲寄存器。写入UxTXB将转到发送缓冲区，然后立即转到TSR（如果它为空）。当TSR不为空时，将保持对UxTXB的写操作，然后在其可用时传送到TSR。

31.11.1 FIFO状态

UxFIFO寄存器包含几个状态位，用于确定接收和发送缓冲区的状态。

RXBE位表示接收FIFO为空。该位基本上与UxRXIF相反。RXBF位表示接收FIFO已满。

发送器只有一个缓冲寄存器，因此状态位基本上是一个副本，与UxTXIF位相反。TXBE位表示缓冲区为空（与UxTXIF相同），TXBF位表示缓冲区已满（与UxTXIF相反）。每当TXBF位置1时执行UxTXB写操作，就会将第三个发送器状态位TXWRE（发送写错误）置1。这表明写入不成功。

31.11.2 FIFO复位

所有模式都支持复位接收和发送缓冲区。

当UxFIFO寄存器中的RXBE位写入1时，刷新接收缓冲区并丢弃所有未读数据。应该使用TXBE位清零的MOVWF指令，以避免在UxTXB为空时意外清零TSR中待处理的字节。

TXEN为低电平时写入UxTXB的数据将保存在发送移位寄存器（TSR）中，然后在TXEN置1时发送。通过将UxFIFO寄存器中的TXBE位置1来刷新发送缓冲区和非活动TSR。如果在字符从TSR主动发送时将TXBE置1，则将完成发送而不刷新。

清零ON位将丢弃所有接收到的数据，并发送TSR和UxTXB中待处理的数据。

31.12 流控制

本节不适用于LIN、DALI或DMX模式。

流控制是通过接收UART暂停发送UART数据流的方法。流控制可以防止输入缓冲区在没有软件干预的情况下溢出。UART支持硬件和XON/XOFF流控制方法。

流控制方法通过UxCON2寄存器中的FLO<1:0>位进行选择。当两个位都被清零时，流控制被禁止。

31.12.1 硬件流控制

硬件流控制可通过将FLO<1:0>位设置为10进行选择。

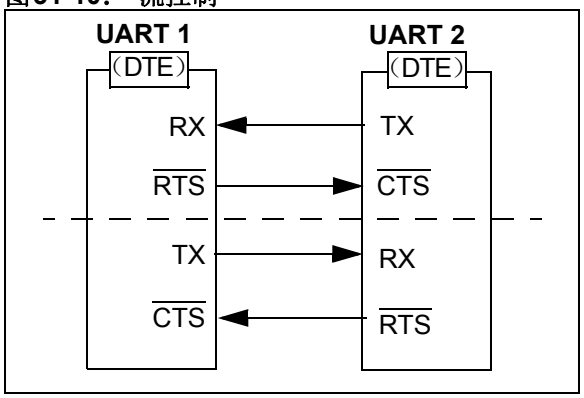
硬件流控制由三根线组成。其中两根的RS-232信号名称为$\overline{\text{RTS}}$和$\overline{\text{CTS}}$。两者都是低电平有效。第三根线可用于控制RS-485收发器。它的信号名称为TXDE，用于发送驱动器使能。该输出在TX输出主动发送一个字符时为高电平，在所有其他时间为低电平。UART配置为DTE（计算机）设备，这意味着$\overline{\text{RTS}}$是输出，$\overline{\text{CTS}}$是输入。

$\overline{\text{RTS}}$和$\overline{\text{CTS}}$信号成对使用以控制发送流。DTE到DTE的配置将接收UART的$\overline{\text{RTS}}$输出连接到发送UART的$\overline{\text{CTS}}$输入。请参见图31-10。

当输入FIFO为空时，接收数据的UART会将$\overline{\text{RTS}}$输出置为低电平。当接收到一个字符时，$\overline{\text{RTS}}$输出变为高电平，直到读取UxRXB以释放两个FIFO单元。

如果$\overline{\text{CTS}}$输入在字节开始发送后变为高电平，发送将正常完成。即使$\overline{\text{CTS}}$输入为高电平，接收器也可通过接受第二个FIFO单元中的字符来实现此目的。

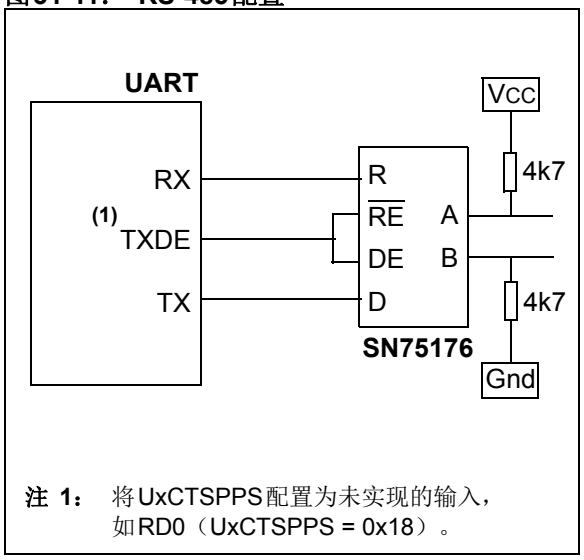
图31-10： 流控制



31.12.2 RS-485收发器控制

硬件流控制可用于控制RS-485收发器的方向，如图31-11所示。通过将UxCTSPPS选择设置为未实现的端口引脚（如RD0），将CTS输入配置为始终使能。当信号线和控制线按照图31-11所示进行配置时，UART将不会接收到自己的发送。要验证RS-485线上没有冲突，可以将收发器RE控制从TXDE断开并连接到低电平，从而实现所有发送的环回接收。更多信息，请参见**第31.14节“冲突检测”**。

图31-11： RS-485配置



注 1： 将UxCTSPPS配置为未实现的输入，如RD0（UxCTSPPS = 0x18）。

31.12.3 XON/XOFF流控制

XON/XOFF流控制可通过将FLO<1:0>位设置为01进行选择。

XON/XOFF是一种基于数据的流控制方法。暂停和恢复发送的信号是接收器发送到发送器的特殊字符。优点是不需要额外的硬件线。

XON/XOFF流控制需要全双工操作，因为发送器必须能够接收信号以在进行发送时暂停发送。虽然在ASCII代码中没有定义XON和XOFF，但XOFF普遍接受的值为13h，XON为11h。UART使用这些代码。

发送器默认为XON，即使能发送器。该状态也由UxFIFO寄存器中的只读XON位指示。

当接收到XOFF字符时，发送器在完成主动发送的字符后停止发送。发送器将保持禁止状态，直到接收到XON字符。

当软件切换TXEN位时，XON将被强制打开。

当UxCON2寄存器中的RUNOVF位置1时，将继续接收和处理XON和XOFF字符，无需通过读取UxRXB来清零输入FIFO。但是，如果RUNOVF位清零，则必须读取UxRXB以避免接收溢出，否则将在接收缓冲区溢出时暂停流控制。

31.13 校验和

本节不适用于LIN模式，LIN模式下会自动处理校验和。

当UxCON2寄存器中的C0EN位置1时，使能发送和接收校验和加法器。使能时，加法器会累加发送或接收的每个字节。累加的和包括加法的进位。软件负责在事务之前清零校验和寄存器，并在事务结束时执行校验。

以下是如何在异步模式下使用校验和寄存器的示例。

31.13.1 发送校验和方法

1. 清零UxTXCHK寄存器。
2. 将C0EN位置1。
3. 发送事务输出的所有字节。
4. 翻转UxTXCHK并将结果作为事务的最后一个字节发送。

31.13.2 接收校验和方法

1. 清零UxRXCHK寄存器。
2. 将C0EN位置1。
3. 接收事务中的所有字节，包括校验和字节。
4. 如果选择了7位模式，则将UxRXCHK的MSb置1。
5. 将UxRXCHK加上1。
6. 如果结果为0，校验和通过，否则失败。

在除LIN以外的任何模式下，CERIF校验和中断标志都无效。

31.14 冲突检测

通过冲突检测对所有操作模式下干扰发送线路的外力进行检测。当RXEN和TXEN都置1时，冲突检测始终有效。

当接收输入通过相同的I/O引脚或外部电路连接到发送输出时，将为发送的每个字符接收一个字符。当接收的字与发送的字不匹配时，冲突检测电路将提供警告。

UxERRIR寄存器中的TXCIF标志用于发出冲突信号。仅当TX输出环回到RX输入并且预计将接收到发送的所有内容时，该信号才有用。如果多个发送器同时有效，则可以假定TX字与RX字不匹配。TXCIF会检测到这种不匹配并标记中断。当接收到的位丢失预期的中间位转换时，TXCIF位也将在DALI模式发送中置1。

无论RX输入是否连接到TX输出，冲突检测始终有效。当不需要冲突中断时，用户可以禁止TXCIE位。

通过将UxCON2中的RUNOVF位置1、忽略接收中断并使接收缓冲区溢出，可以避免卸载发送数据的接收缓冲区的软件开销。发送完成后，通过刷新接收缓冲区来准备接收数据（见第31.11.2节“FIFO复位”）并清零UxERRIR寄存器中的RXFOIF溢出标志。

31.15 RX/TX活动超时

UART与HLT定时器配合使用来监视RX和TX线上的活动。使用此功能可确定在用户指定的时间段内接收或发送线路上没有活动的时间。

要使用此功能，请通过HLT时钟源、定时器预分频值和定时器周期寄存器的组合将HLT设置为所需的超时周期。将HLT配置为在UART TX或RX线上复位，并在UART启动的同时启动HLT。UART活动将持续复位HLT，以防止整个HLT周期结束。当选定TX或RX线上无活动的时间超过HLT周期时，会发生HLT中断，并发出超时事件信号。

例如，以下寄存器设置将为HLT2配置5 ms超时（U1RX上无活动时）：

- T2PR = 0x9C（156个预分频周期）
- T2CLKCON = 0x05（500 kHz内部振荡器）
- T2HLT = 0x04（自由运行，上升沿复位）
- T2RST = 0x15（U1RX复位）
- T2CON = 0xC0（Timer2打开，预分频比为1:16）

31.16 异步操作的时钟精度

内部振荡器模块输出（INTOSC）在出厂时做了校准。但是，VDD或温度变化时，INTOSC频率有可能漂移，这将直接影响异步波特率。有两种方法可用来调整波特率时钟，但它们都需要某种参考时钟源。

第一种（首选）方法使用OSCTUNE寄存器调整INTOSC输出。调整OSCTUNE寄存器的值可对系统时钟源的分辨率进行微调。更多信息，请参见**第7.2.2.3节“内部振荡器频率调整”**。

另一种方法调整波特率发生器的值。自动波特率检测可自动完成这种调整（见**第31.17.1节“自动波特率检测”**）。通过调整波特率发生器来补偿外设时钟频率的逐渐变化时，可能无法足够精细地调节分辨率。

31.17 UART波特率发生器（BRG）

波特率发生器（BRG）是16位定时器，专用于支持UART操作。

UxBRGH和UxBRGL寄存器对决定自由运行波特率定时器的周期。波特率周期的倍频值由UxCON0寄存器中的BRGS位决定。

表31-1提供了确定波特率的公式。例31-1提供了确定波特率和波特率误差的计算示例。

高波特率范围（BRGS = 1）用于在无法达到所需波特率时将波特率范围扩展到更快的速率。使用任一范围均可实现所需波特率时，建议使用正常波特率范围（BRGS = 0）。

将新值写入UxBRGH和UxBRGL寄存器对将导致BRG定时器复位（或清零）。这可以确保BRG无需等待定时器溢出就可以输出新的波特率。

如果系统时钟在有效的接收操作过程中被更改，可能会导致接收错误或数据丢失。为避免此问题，应检查RXIDL位的状态，以确保在改变系统时钟前接收操作处于空闲状态。

例31-1： 计算波特率误差

针对工作在异步模式下，FOSC为16 MHz，目标波特率为9600，BRGS = 0的器件：

$$所需波特率 = \frac{FOSC}{16([UxBRG]+1)}$$

$$X = \frac{\frac{FOSC}{所需波特率}}{16} - 1$$

$$= \frac{\frac{16000000}{9600}}{16} - 1$$

$$= [103.17] = 103$$

$$计算得到的波特率 = \frac{16000000}{16(103+1)}$$

$$= 9615$$

$$误差 = \frac{计算得到的波特率 - 所需波特率}{所需波特率}$$

$$= \frac{(9615-9600)}{9600} = 0.16\%$$

表31-1： 波特率公式

BRGS	BRG/UART模式	波特率公式
1	高波特率	FOSC/[4 (n+1)]
0	正常波特率	FOSC/[16(n+1)]

图注： n = UxBRGH和UxBRGL寄存器对的值。

31.17.1 自动波特率检测

UART模块支持8位异步和LIN模式下的波特率自动检测和校准。但是，既不必要也不可能在LIN模式下将ABDEN置1以启动自动波特率检测，因为该模式支持在每个数据包开始时自动进行自动波特率检测。通过ABDEN位使能自动波特率检测仅适用于异步模式。

当自动波特率检测（Auto-Baud Detect，ABD）有效时，提供给BRG的时钟是反向的。BRG并不为传入的RX信号提供时钟信号，而是由RX信号为BRG定时。波特率发生器用来测量接收55h字符（ASCII“U”，也是LIN总线的同步字符）的周期。该字符的惟一特性是它具有5个下降沿（包括启动位边沿在内）和5个上升沿（包括停止位边沿在内）。

在8位异步模式下，通过将UxCON0寄存器中的ABDEN位置1，可以使能自动波特率校准序列。ABDEN置1后，RX输入的第一个下降沿将启动自动波特率校准序列。当发生ABD序列时，UART状态机保持在空闲状态。UxBRG使用BRG计数器时钟在接收信号的第一个下降沿开始计数（如图31-12所示）。在bit 7周期的开始将在RX引脚上出现第5个下降沿。此时，正确的BRG周期总数累计值被保存在UxBRGH和UxBRGL寄存器对中，ABDEN位被自动清零而ABDIF中断标志被置1。必须用软件将ABDIF清零。

RXIDL表示同步输入有效。RXIDL将在第一个下降沿变为低电平，在第五个上升沿变为高电平。

BRG自动波特率时钟由BRGS位决定，如表31-2所示。在ABD期间，内部BRG寄存器用作16位计数器。但是，UxBRGH和UxBRGL寄存器保留先前的BRG值，直到自动波特率过程成功完成。在校准波特率周期时，内部BRG寄存器的时钟频率为BRG基本时钟频率的1/8。得到的字节测量结果为全速时的平均位时间，并在完成时传输到UxBRGH和UxBRGL寄存器。

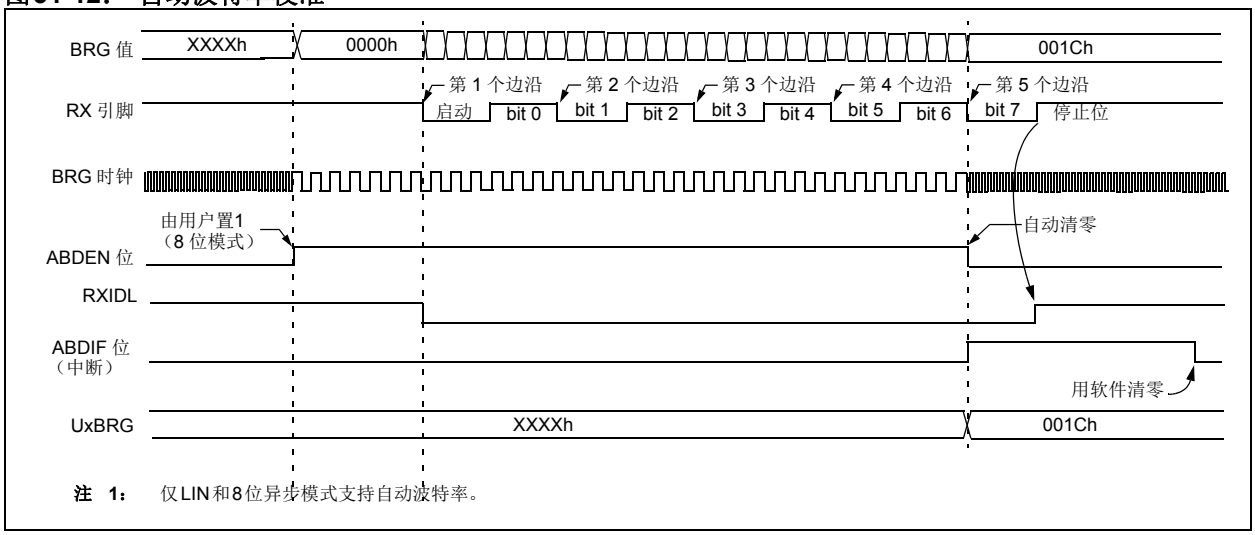
注 1： 如果WUE位和ABDEN位都置1，自动波特率检测将从间隔字符之后的字节开始（见**第31.17.3节“接收到间隔字符时自动唤醒”**）。

2： 需要由用户来判断输入字符的波特率是否处于所选BRG时钟源范围内。可能无法实现某些振荡器频率和UART波特率组合。

表31-2： BRG计数器时钟速率

BRGS	BRG基本时钟	BRG ABD时钟
1	FOSC/4	FOSC/32
0	FOSC/16	FOSC/128

图31-12： 自动波特率校准



31.17.2 自动波特率溢出

在自动波特率检测过程中，如果在RX引脚上检测到第5个下降沿之前波特率计数器溢出，则UxERRIR寄存器中的ABDOVF位将被置1。ABDOVF位指示计数器已超出UxBRGH:UxBRGL寄存器对的16位所能允许的最大计数值。在ABDOVF位置1后，状态机将一直搜索，直到在RX引脚上检测到第5个下降沿为止。检测到第5个RX下降沿时，硬件会将ABDIF中断标志位置1并将UxCON0寄存器中的ABDEN位清零。UxBRGH和UxBRGL寄存器值保留其先前的值。UxUIR寄存器中的ABDIF标志和UxERRIR寄存器中的ABDOVF标志可直接由软件清零。要在自动波特率溢出条件下产生中断，必须将以下所有位置1：

- UxERRIE寄存器中的ABDOVE位
- PIEx寄存器中的UxEIE位
- INTCON寄存器中的PIE和GIE位

要在ABDIF标志位置1之前终止自动波特率检测过程，可依次将UxERRIR寄存器中的ABDEN位和ABDOVF位清零。

31.17.3 接收到间隔字符时自动唤醒

在休眠模式下，UART的所有时钟都会暂停。因此，波特率发生器处于不工作状态，不能正常进行字符接收。自动唤醒功能使控制器可被RX线上的活动唤醒。

自动唤醒功能可通过将UxCON1寄存器中的WUE位和PIEx寄存器中的UxIE位置1来使能。一旦置1，RX上的正常接收序列就被禁止，UART保持在空闲状态，监视与CPU模式无关的唤醒事件。唤醒事件包含RX线上退出空闲状态。（这与间隔字符或LIN协议的唤醒信号字符的启动条件一致。）

UART模块产生的WUIF中断与唤醒事件同步。在正常CPU工作模式下，与Q时钟同步产生中断（图31-13）；在休眠模式下，与Q时钟异步产生中断（图31-14）。可通过清零UxUIR寄存器中的WUIF位来清除中断条件。要在唤醒事件下产生中断，必须将以下所有位置1：

- PIEx寄存器中的UxIE位
- INTCON寄存器中的PIE和GIE位

RX线在间隔字符末尾转换到空闲状态将自动清零WUE位。这向用户表明间隔事件结束。此时，UART模块处于空闲模式，等待接收下一个字符。

31.17.3.1 特殊注意事项

间隔字符

在发生唤醒事件期间为了避免字符错误或字符碎片，唤醒字符必须为全零。

唤醒被使能时，其工作状况与数据流的低电平时间无关。如果WUE位置1并接收到了有效的非零字符，则从启动位至第一个上升沿的低电平时间将被解读为唤醒事件。字符的其余位将作为碎片字符接收，后续字符有可能产生帧错误或溢出错误。

因此，发送的首字符必须为全0。这必须持续11个或更长的位时间，对于LIN总线，建议持续13个位时间，而对于标准RS-232器件，可为任意个位时间。

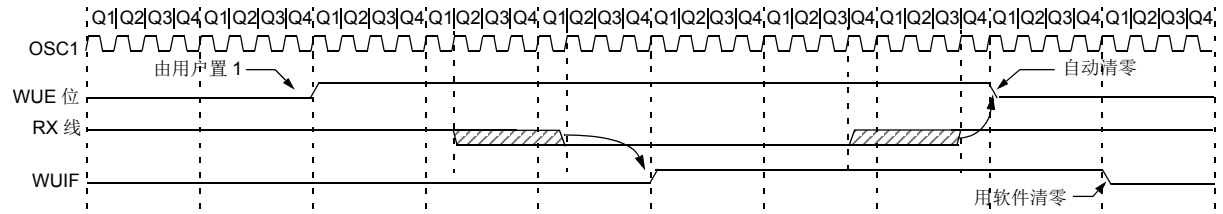
振荡器起振时间

必须考虑振荡器起振时间，特别在使用起振时间较长的振荡器（即，LP、XT或HS/PLL模式）的应用中。同步间隔（或唤醒信号）字符必须足够长，且随后有一个足够长的间隔时间，以使所选的振荡器有足够的时间起振并在这段时间对UART进行正确初始化。

WUE位

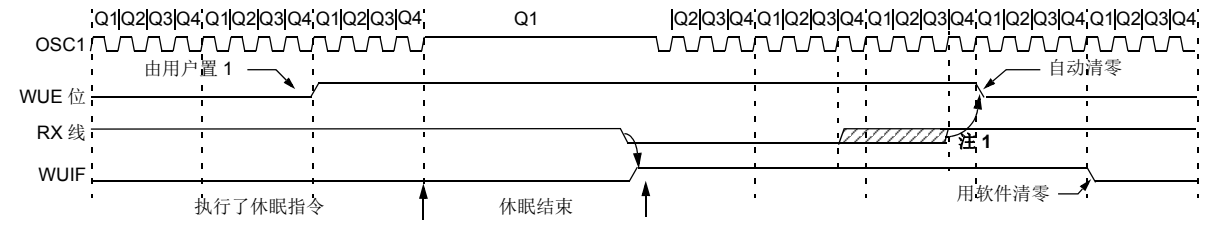
要确保不丢失实际数据，应在将WUE位置1前检查RXIDL位，验证没有接收操作在进行。如果未发生接收操作，可在进入休眠模式前将WUE位置1。

图31-13： 正常工作时的自动唤醒位（WUE）时序



注 1： WUE位置1时，**UART**保持在空闲状态。

图31-14： 休眠时的自动唤醒位（WUE）时序



注 1： 如果唤醒事件需要较长的振荡器预热时间，在stposc信号仍然有效时可能发生WUE位自动清零。该序列与Q时钟出现与否无关。

2： WUE位置1时，**UART**保持在空闲状态。

31.18 发送间隔

UART模块能够发送固定长度的间隔周期或软件定时的间隔周期。固定长度的间隔包含1个启动位以及随后的12 个 0 位和 1 个停止位。软件定时的间隔可通过将UxCON1寄存器中的BRKOVR位置1和清零产生。

要发送固定长度的间隔，应将 UxCON0 寄存器中的SENDB和TXEN位置1。随后对UxTXB执行写操作可启动间隔序列。定时的间隔将首先发生，然后是启动间隔的写入UxTXB的字符。启动字符通常是LIN规范的同步字符。

在LIN和DMX模式下禁止SENB，因为这些模式会自动生成间隔序列。

在间隔停止位完成后，硬件会自动将SENDB位复位。

UxERRIR寄存器中的TXMTIF位表明发送操作何时处于有效或空闲状态，这与正常发送时相同。图31-15给出了间隔序列的时序。

31.19 接收间隔

UART具有计数器，用于检测RX输入何时长时间保持在空格状态。此时，UxERRIR寄存器中的RXBKIF位置1。

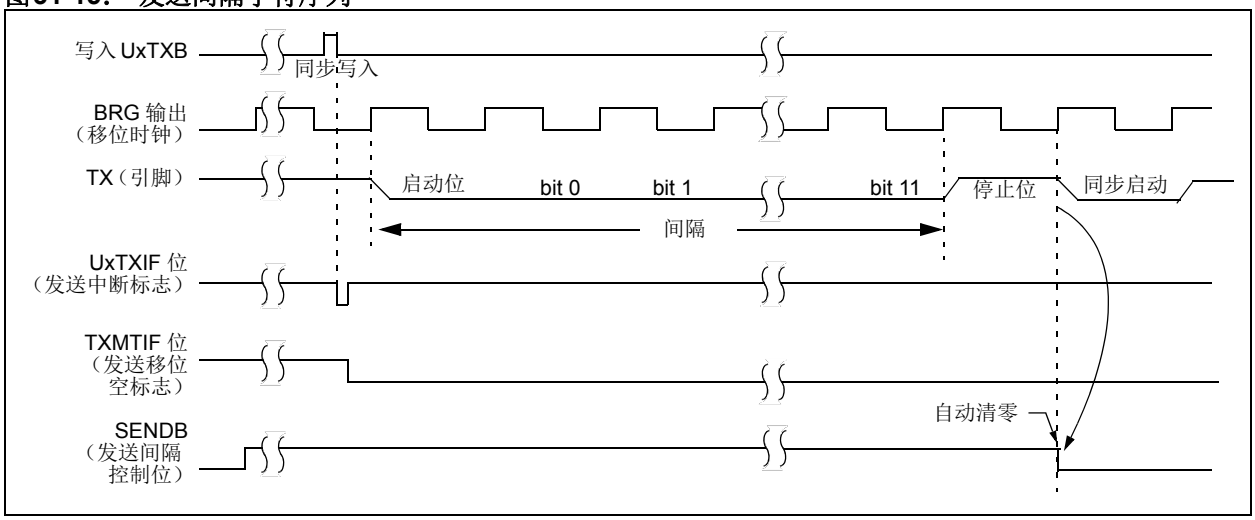
当RX输入在异步和LIN模式下保持11位周期的空格状态，在DMX模式下保持23位周期的空格状态时，将检测到间隔。

当RX输入返回空闲状态时，用户可以选择在检测到间隔后或者在间隔结束时接收间隔中断。当UxCON1中的RXBIMD位为1时，RXBKIF在检测到间隔时立即置1。当RXBIMD为0时，RXBKIF在RX输入返回空闲状态时置1。

31.20 休眠期间的UART操作

UART在休眠期间停止工作。通过串行操作将器件从休眠模式唤醒的安全方法是使用UART的接收到间隔字符时唤醒功能。请参见第31.17.3节“接收到间隔字符时自动唤醒”

图31-15： 发送间隔字符序列



31.21 寄存器定义：UART控制

UART外设的长位名称前缀如下所示。更多信息，请参见**第1.3节“寄存器和位命名约定”**。

外设	位名称前缀
UART 1	U1
UART 2	U2

寄存器31-1： UxCON0：UART控制寄存器 0

R/W-0/0	R/W/HS/HC-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
BRGS	ABDEN	TXEN	RXEN	MODE<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **BRGS**：波特率发生器速度选择位
1 = 波特率发生器为高速，每个位4个波特率时钟
0 = 波特率发生器为正常速度，每个位16个波特率时钟

bit 6 **ABDEN**：自动波特率检测使能位(3)
1 = 使能自动波特率。接收器正在等待同步字符（0x55）
0 = 未使能自动波特率或已完成自动波特率

bit 5 **TXEN**：发送使能控制位(2)
1 = 使能发送。TX输出引脚驱动在发送有效时强制开启，在发送空闲时由端口TRIS控制进行控制。
0 = 禁止发送。TX输出引脚驱动由端口TRIS控制进行控制

bit 4 **RXEN**：接收使能控制位(2)
1 = 使能接收器
0 = 禁止接收器

bit 3-0 **MODE<3:0>**：UART模式选择位(1)
1111 = 保留
1110 = 保留
1101 = 保留
1100 = LIN主/从模式
1011 = 仅LIN从模式
1010 = DMX模式
1001 = DALI控制装置模式
1000 = DALI控制器件模式
0111 = 保留
0110 = 保留
0101 = 保留
0100 = 异步9位UART地址模式。第9位：1 = 地址；0 = 数据
0011 = 异步8位UART模式，第9位为偶校验位
0010 = 异步8位UART模式，第9位为奇校验位
0001 = 异步7位UART模式
0000 = 异步8位UART模式

注 1： ON = 1时更改UART模式可能会导致意外的结果。
2： 清零TXEN或RXEN不会清零相应的缓冲区。使用TXBE或RXBE清零缓冲区。
3： 当MODE = 1001时，ABDEN为只读位。当MODE = 100x且ABDEN = 1时，自动波特率由启动位决定。

寄存器31-2： UxCON1：UART控制寄存器 1

R/W-0/0	U-0	U-0	R/W/HC-0/0	R/W-0/0	U-0	R/W-0/0	R/W/HC-0/0
ON	—	—	WUE	RXBIMD	—	BRKOVR	SENDB
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **ON：**串口使能位
1 = 使能串口
0 = 禁止串口（保持在复位状态）

bit 6-5 **未实现：**读为0

bit 4 **WUE：**唤醒使能位
1 = 接收器正在等待RX输入下降沿，这会将UxIF位置1。发生唤醒事件时由硬件清零。还需要PIEx的UxIE位才能使能唤醒。
0 = 接收器正常工作

bit 3 **RXBIMD：**接收间隔字符中断模式选择位
1 = 当RX输入持续低电平达最小间隔时间时，立即将RXBKIF置1
0 = 在RX输入持续低电平达最小间隔时间后，在RX输入上升沿将RXBKIF置1

bit 2 **未实现：**读为0

bit 1 **BRKOVR：**发送间隔软件改写位
1 = TX输出被强制为非空闲状态
0 = TX输出由发送移位寄存器驱动

bit 0 **SENDB：**发送间隔控制位**(1)**
1 = 在UxTXB写入时输出间隔。写入字节之后是间隔。该位由硬件清零。
0 = 已完成或禁止间隔字符的发送

注 1： 该位在LIN、DMX和DALI模式下为只读位。

寄存器31-3： UxCON2：UART控制寄存器 2

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
RUNOVF	RXPOL	STP<1:0>		C0EN	TXPOL	FLO<1:0>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **RUNOVF**：溢出期间运行控制位
1 = RX输入移位器在溢出条件后继续与启动位同步
0 = RX输入移位器在接收器溢出条件下停止所有活动

bit 6 **RXPOL**：接收极性控制位
1 = RX极性翻转，空闲状态为低电平
0 = RX极性不翻转，空闲状态为高电平

bit 5-4 **STP<1:0>**：停止位模式控制位(1)
11 = 发送2个停止位，接收器验证第一个停止位
10 = 发送2个停止位，接收器验证第一个和第二个停止位
01 = 发送1.5个停止位，接收器验证第一个停止位
00 = 发送1个停止位，接收器验证第一个停止位

bit 3 **C0EN**：校验和模式选择位
LIN模式：
1 = 校验和模式1，增强型LIN校验和，包括PID总和
0 = 校验和模式0，传统LIN校验和，不包括PID总和
其他模式：
1 = 添加所有TX和RX字符
0 = 禁止校验和

bit 2 **TXPOL**：发送极性控制位
1 = 输出数据翻转，TX输出在空闲状态下为低电平
0 = 输出数据不翻转，TX输出在空闲状态下为高电平

bit 1-0 **FLO<1:0>**：握手流控制位
11 = 保留
10 = RTS/CTS和TXDE硬件流控制
01 = XON/XOFF软件流控制
00 = 流控制关闭

注 1： 所有模式都发送选定数量的停止位。只有DMX和DALI接收器验证选定数量的停止位，而所有其他接收器仅验证第一个停止位。

寄存器31-4： UxERRIR：UART错误中断标志寄存器

R/S/C-1/1	R/S/C-0/0	R/W/S-0/0	R/W/S-0/0	R/S/C-0/0	R/W/S-0/0	R/W/S-0/0	R/W/S-0/0
TXMTIF	PERIF	ABDOVF	CERIF	FERIF	RXBKIF	RXFOIF	TXCIF
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值	
1 = 置1	0 = 清零	S = 硬件置1	C = 硬件清零

bit 7 **TXMTIF**：发送移位寄存器空中断标志位
1 = 发送移位寄存器为空（在停止位结束时置1）
0 = 发送移位寄存器正在主动移位数据

bit 6 **PERIF**：奇偶校验错误中断标志位
LIN和奇偶校验模式：
1 = 输入FIFO顶部的未读字节存在奇偶校验错误
0 = 输入FIFO顶部的未读字节没有奇偶校验错误
DALI器件模式：
1 = 输入FIFO顶部的未读字节作为前向帧接收
0 = 输入FIFO顶部的未读字节作为后向帧接收
地址模式：
1 = 输入FIFO顶部的未读字节作为地址接收
0 = 输入FIFO顶部的未读字节作为数据接收
其他模式：
未使用

bit 5 **ABDOVF**：自动波特率检测溢出中断标志位
DALI模式：
1 = 启动位测量溢出计数器
0 = 启动位测量期间无溢出
其他模式：
1 = 自动检测序列期间波特率发生器溢出
0 = 波特率发生器未溢出

bit 4 **CERIF**：校验和错误中断标志位（仅LIN模式）
1 = 校验和错误
0 = 无校验和错误

bit 3 **FERIF**：帧错误中断标志位
1 = 输入FIFO顶部的未读字节存在帧错误
0 = 输入FIFO顶部的未读字节没有帧错误

bit 2 **RXBKIF**：间隔接收中断标志位
1 = 检测到间隔
0 = 未检测到间隔

bit 1 **RXFOIF**：接收FIFO溢出中断标志位
1 = 接收FIFO已溢出
0 = 接收FIFO未溢出

bit 0 **TXCIF**：发送冲突中断标志位
1 = 发送的字与发送期间接收的字不相等
0 = 发送的字与发送期间接收的字相等

寄存器31-5： UxERRIE：UART错误中断允许寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TXMTIE	PERIE	ABDOVE	CERIE	FERIE	RXBKIE	RXFOIE	TXCIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **TXMTIE**：发送移位寄存器空中断允许位
1 = 允许中断
0 = 禁止中断

bit 6 **PERIE**：奇偶校验错误中断允许位
1 = 允许中断
0 = 禁止中断

bit 5 **ABDOVE**：自动波特率检测溢出中断允许位
1 = 允许中断
0 = 禁止中断

bit 4 **CERIE**：校验和错误中断允许位
1 = 允许中断
0 = 禁止中断

bit 3 **FERIE**：帧错误中断允许位
1 = 允许中断
0 = 禁止中断

bit 2 **RXBKIE**：间隔接收中断允许位
1 = 允许中断
0 = 禁止中断

bit 1 **RXFOIE**：接收FIFO溢出中断允许位
1 = 允许中断
0 = 禁止中断

bit 0 **TXCIE**：发送冲突中断允许位
1 = 允许中断
0 = 禁止中断

寄存器31-6： UxUIR：UART通用中断寄存器

R/S/W-0/0	R/S/W-0/0	U-0	U-0	U-0	R/W-0/0	U-0	U-0
WUIF	ABDIF	—	—	—	ABDIE	—	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	S = 硬件置1

bit 7 **WUIF**：唤醒中断位
1 = 当WUE置1时，检测到RX线上空闲到非空闲的转换。还会将UxIF置1。（必须用软件将WUIF清零以清零UxIF）
0 = 软件未使能WUE或未检测到转换

bit 6 **ABDIF**：自动波特率检测中断位
1 = 自动波特率检测已完成。ABDIE置1时，将在UxIF中显示状态。（必须用软件清零）
0 = 自动波特率未使能；或自动波特率已使能，但未完成自动波特率检测

bit 5-3 **未实现**：读为0

bit 2 **ABDIE**：自动波特率检测中断允许位
1 = ABDIF将PIRx寄存器中的UxIF位置1
0 = ABDIF不会将UxIF置1

bit 1-0 **未实现**：读为0

寄存器31-7： UxFIFO：UART FIFO状态寄存器

R/W/S-0/0	R/W-0/0	R/W/S/C-1/1	R/S/C-0/0	R/S/C-1/1	S/C-1/1	R/W/S/C-1/1	R/S/C-0/0
TXWRE	STPMD	TXBE	TXBF	RXIDL	XON	RXBE	RXBF
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值	
1 = 置1	0 = 清零	S = 硬件置1	C = 硬件清零

bit 7 **TXWRE**：发送写错误状态位（必须由软件清零）

LIN主模式：
1 = 在主进程处于活动状态时写入UxP1L

LIN从模式：
1 = 在UxP2 = 0时或自上次间隔以来已经向UxTXB写入了超过UxP2个字节时写入UxTXB

地址检测模式：
1 = 在将UxP1L中的先前数据传输到TX移位器之前写入UxP1L

所有模式：
1 = 输出FIFO已满时向UxTXB写入一个新字节
0 = 无错误

bit 6 **STPMD**：停止位检测模式位
1 = 如果STP = 11，在最后一个停止位结束时或在第一个停止位结束时将UxRXIF置为有效
0 = 在第一个停止位的中间将UxRXIF置为有效

bit 5 **TXBE**：发送缓冲区空状态位
1 = 发送缓冲区为空。将此位置1将清零发送缓冲区和输出移位寄存器。
0 = 发送缓冲区不为空。软件无法清零此位。

bit 4 **TXBF**：发送缓冲区满状态位
1 = 发送缓冲区已满
0 = 发送缓冲区未满

bit 3 **RXIDL**：接收引脚空闲状态位
1 = 接收引脚处于空闲状态
0 = UART正在接收启动、停止、数据、自动波特率或间隔

bit 2 **XON**：软件流控制发送使能状态位
1 = 使能发送器
0 = 禁止发送器

bit 1 **RXBE**：接收缓冲区空状态位
1 = 接收缓冲区为空。置1该位会清零RX缓冲区(1)
0 = 接收缓冲区不为空。软件无法清零此位。

bit 0 **RXBF**：接收缓冲区满状态位
1 = 接收缓冲区已满
0 = 接收缓冲区未满

注 1： 不应使用BSF指令将RXBE置1，因为这样做会在UxTXB寄存器为空时清零发送移位寄存器中的待处理字节。而是应在TXBE位的位置使用值为0的MOVWF指令。

寄存器31-8： UxBRGL：UART波特率发生器低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
BRG<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **BRG<7:0>：**波特率发生器的最低有效字节

寄存器31-9： UxBRGH：UART波特率发生器高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
BRG<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **BRG<15:8>：**波特率发生器的最高有效字节

注 1： 只有在ON = 0时才能写入UxBRG寄存器。

2： 当MODE = 100x且BRGS = 1时，最大BRG值为0x7FFE。

3： 当MODE = 100x且BRGS = 0时，最大BRG值为0x1FFE。

寄存器31-10： UxRXB：UART接收寄存器

R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
RXB<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **RXB<7:0>**：接收缓冲区顶部

寄存器31-11： UxTXB：UART发送寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TXB<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **TXB<7:0>**：发送缓冲区底部

寄存器31-12： UxP1H：UART参数1高字节寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	P1<8>
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现**：读为0

bit 0 **P1<8>**：参数1的最高有效位

DMX模式：
要在起始码和自动间隔生成之间发送的字节数的最高有效位

DALI控制器件模式：
发送前向帧之前的空闲时间延时的最高有效位。以半位周期为单位测量

DALI控制装置模式：
前向帧结束与后向帧开始之间的延时的最高有效位。以半位周期为单位测量

其他模式：
未使用

寄存器31-13： UxP1L：UART参数1低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
P1<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **P1<7:0>**：参数1的最低有效位

DMX模式：
要在起始码和自动间隔生成之间发送的字节数的最低有效字节

DALI控制器件模式：
发送前向帧之前的空闲时间延时的最低有效字节。以半位周期为单位测量

DALI控制装置模式：
前向帧结束与后向帧开始之间的延时的最低有效字节。
以半位周期为单位测量

LIN模式：
待发送的PID（仅使用低6位）

异步寻址模式：
待发送的地址（第9个发送位自动设置为1）

其他模式：
未使用

寄存器31-14： UxP2H：UART参数2高字节寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	P2<8>
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 0 **P2<8>：**参数2的最高有效位

DMX模式：
接收块的第一个地址的最高有效位

DALI模式：
前向帧检测阈值中空闲时间的半位周期数的最高有效位

其他模式：
未使用

寄存器31-15： UxP2L：UART参数2低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
P2<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **P2<7:0>：**参数2的最低有效位

DMX模式：
接收块的第一个地址的最低有效字节

LIN从模式：
要发送的数据字节数

DALI模式：
前向帧检测阈值中空闲时间的半位周期数的最低有效字节

异步寻址模式：
接收器地址

其他模式：
未使用

寄存器31-16： UxP3H：UART参数3高字节寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	P3<8>
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现：**读为0

bit 0 **P3<8>：**参数3的最高有效位

DMX模式：
接收块的最后一个地址的最高有效位

其他模式：
未使用

寄存器31-17： UxP3L：UART参数3低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
P3<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **P3<7:0>：**参数3的最低有效位

DMX模式：
接收块的最后一个地址的最低有效字节

LIN从模式：
要接收的数据字节数

异步寻址模式：
接收器地址掩码。接收的地址先与UxP2L进行异或运算，然后与UxP3L进行与运算
当结果为零时发生匹配

其他模式：
未使用

寄存器31-18： UxTXCHK：UART发送校验和结果寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TXCHK<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **TXCHK<7:0>**：根据TX字节计算的校验和

LIN模式且C0EN = 1：
发送的所有字节之和，包括PID

LIN模式且C0EN = 0：
发送的所有字节之和，不包括PID

所有其他模式且C0EN = 1：
自上次清零以来发送的所有字节之和

所有其他模式且C0EN = 0：
未使用

寄存器31-19： UxRXCHK：UART接收校验和结果寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
RXCHK<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **RXCHK<7:0>**：根据RX字节计算的校验和

LIN模式且C0EN = 1：
接收的所有字节之和，包括PID

LIN模式且C0EN = 0：
接收的所有字节之和，不包括PID

所有其他模式且C0EN = 1：
自上次清零以来接收的所有字节之和

所有其他模式且C0EN = 0：
未使用

表31-3： 与UART相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
UxCON0	BRGS	ABDEN	TXEN	RXEN	MODE<3:0>				483
UxCON1	ON	—	—	WUE	RXBIMD	—	BRKOVR	SENDB	484
UxCON2	RUNOVF	RXPOL	STP<1:0>		C0EN	TXPOL	FLO<1:0>		485
UxERRIR	TXMTIF	PERIF	ABDOVF	CERIF	FERIF	RXBKIF	RXFOIF	TXCIF	486
UxERRIE	TXMTIE	PERIE	ABDOVE	CERIE	FERIE；	RXBKIE	RXFOIE	TXCIE	487
UxUIR	WUIF	ABDIF	—	—	—	ABDIE	—	—	488
UxFIFO	TXWRE	STPMD	TXBE	TXBF	RXIDL	XON	RXBE	RXBF	489
UxBRGL	BRG<7:0>								490
UxBRGH	BRG<15:8>								490
UxRXB	RXB<7:0>								491
UxTXB	TXB<7:0>								491
UxP1H	—	—	—	—	—	—	—	P1<8>	492
UxP1L	P1<7:0>								492
UxP2H	—	—	—	—	—	—	—	P2<8>	493
UxP2L	P2<7:0>								493
UxP3H	—	—	—	—	—	—	—	P3<8>	494
UxP3L	P3<7:0>								494
UxTXCHK	TXCHK<7:0>								495
UxRXCHK	RXCHK<7:0>								495

图注： — = 未实现，读为0。UART模块不使用阴影单元。

32.0 串行外设接口（SPI）模块

32.1 SPI模块概述

SPI（串行外设接口）模块是以全双工模式工作的同步串行数据通信总线。器件在由主器件启动通信的主/从器件环境中进行通信。从器件通过称为从选择的片选进行控制。示例从器件包括串行 EEPROM、移位寄存器、显示驱动器、模数转换器或其他PIC®器件。

SPI总线规定了4种信号连接：

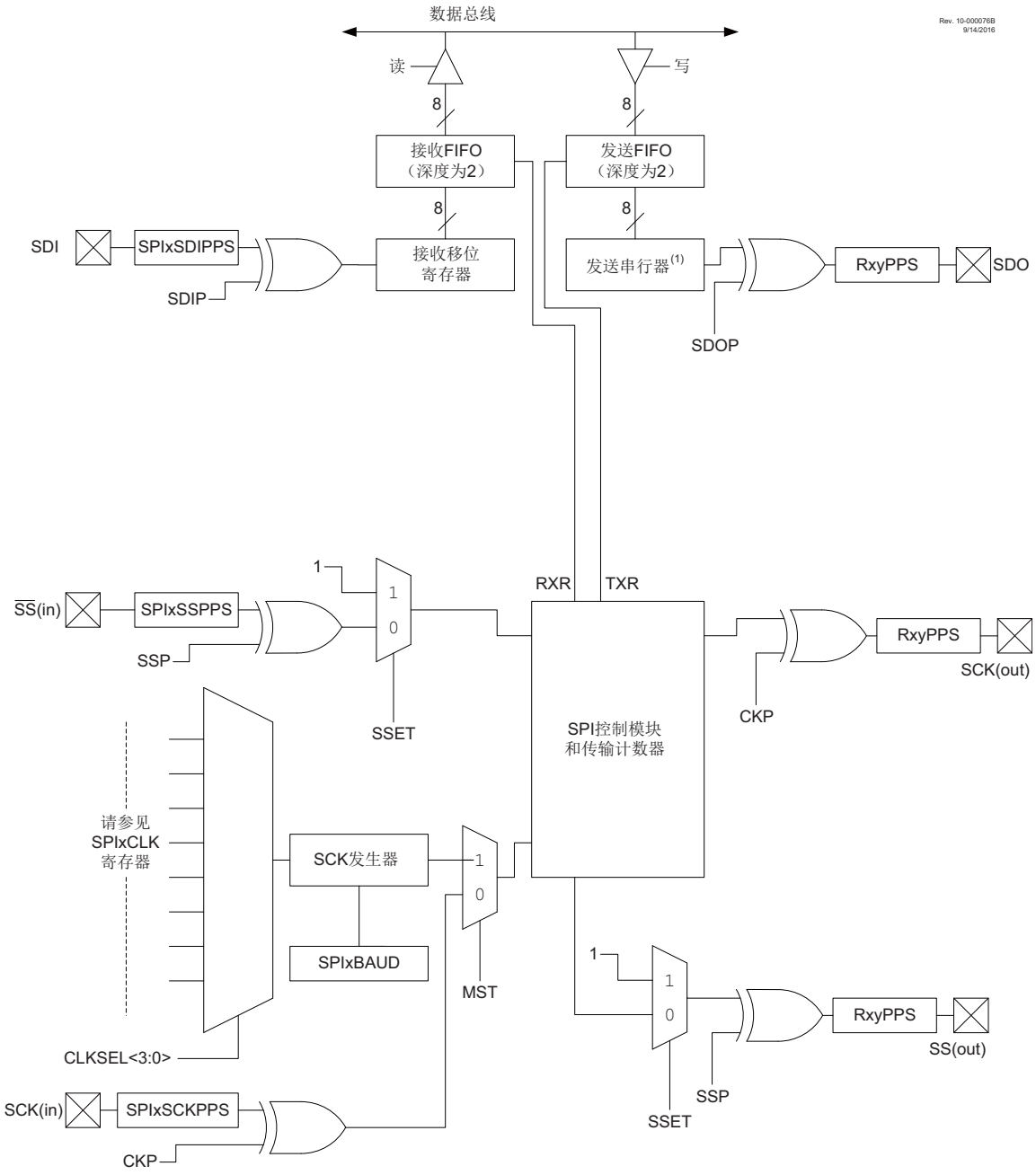
- 串行时钟（Serial Clock，SCK）
- 串行数据输出（Serial Data Out，SDO）
- 串行数据输入（Serial Data IN，SDI）
- 从选择（Slave Select，$\overline{SS}$）

SPI接口支持以下模式和特性：

- 主模式
- 从模式
- 时钟极性和边沿选择
- SDI、SDO和SS极性控制
- 独立的发送和接收使能
- 从选择同步
- 从器件的菊花链连接
- 具有2字节FIFO和DMA功能的独立发送和接收缓冲区

图32-1给出了SPI模块的框图。

图32-1： SPI模块的简化框图



注1：如果TXR = 1且发送FIFO为空，则接收移位寄存器的前一个值将被发送到发送串行器。

SPI发送输出（SDO_out）可送至可重映射PPS SDO引脚和在内部送至以下外设：

- 可配置逻辑单元（CLC）
- 数据信号调制器（DSM）

SPI总线工作时通常使用单个主器件和一个或多个从器件。使用多个从器件时，从主器件到每个从器件都需要独立的从选择连接。

主器件每次仅选择一个从器件。大多数从器件都具有三态输出，所以在未选择它们时，它们的输出信号与总线断开。

数据发送通常涉及到移位寄存器，它们大小都为8位，一个在主器件中，一个在从器件中。不论是对于主器件还是从器件，数据总是每次移出一位，最高有效位（MSb）先移出。与此同时，新的位会被移入器件中。与旧版 Microchip 器件不同，PIC18(L)F2X/4X/5XK42上的SPI包含两个独立的寄存器，分别用于存储传入数据和传出数据。这两个寄存器还有2字节FIFO缓冲区，允许DMA总线连接。

图32-2给出了分别配置为主器件和从器件的两个PIC18F2X/4XK42器件之间的典型连接。

数据在所设定的时钟边沿从发送FIFO移出，并在相反的时钟边沿移入接收移位寄存器。

主器件通过它的SDO输出引脚发送信息，并由该引脚所连接的从器件SDI输入引脚接收。从器件通过它的SDO输出引脚发送信息，并由该引脚所连接的主器件SDI输入引脚接收。

主器件发送时钟信号。主器件和从器件应配置为相同的时钟极性。

在每个SPI时钟周期中，会发生全双工数据发送。这意味着，在主器件从其输出寄存器中发送出MSb（在其SDO引脚上），从器件读取该位并将它保存为其输入寄存器的LSb的同时，从器件也会从其移位寄存器中发送出MSb（在其SDO引脚上），而主器件也会读取该位并将它保存为其输入寄存器的LSb。

在移出8位之后，主器件和从器件便交换了寄存器值，并将传入的数据存储到接收FIFO中。

如果需要交换更多数据，寄存器中会装入新数据，并重复该过程。

数据有意义还是无意义（无效数据），取决于应用软件。这就导致以下三种数据传输情形：

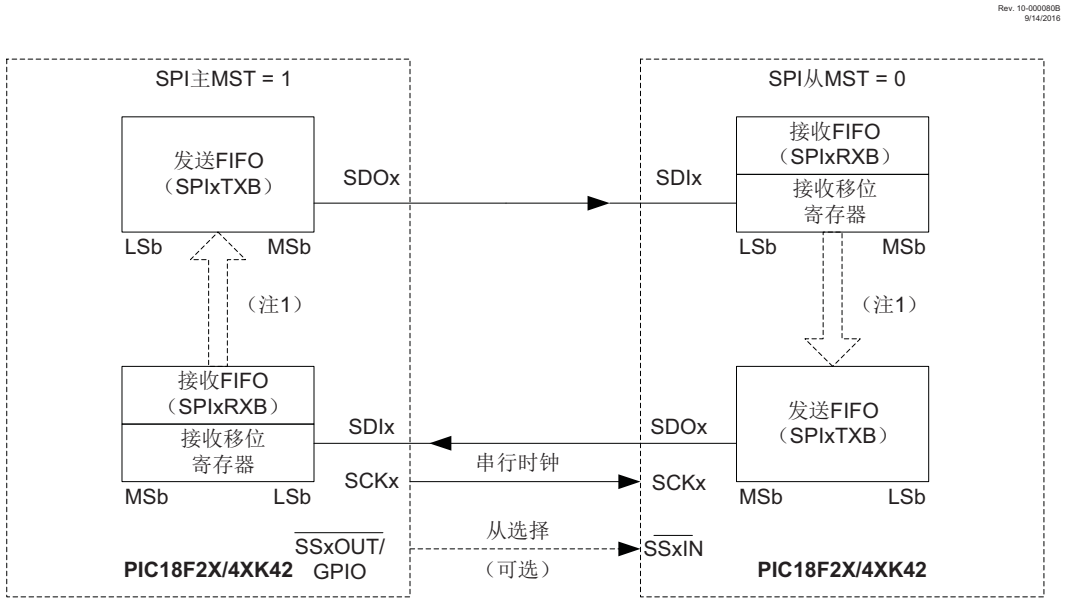
- 主器件发送有用数据，从器件发送无效数据
- 主器件发送有用数据，从器件发送有用数据
- 主器件发送无效数据，从器件发送有用数据

在该特定SPI模块中，可以通过清零RXR位（用于接收无效数据）或TXR位（用于发送无效数据）（有关TXR/RXR设置的更多详细信息，请参见表32-1以及**第32.5节“主模式”**和**第32.6节“从模式”**）发送无效数据，而无需使用软件。该SPI模块可以发送任意数量的位，并且可以不同大小的段（宽度为1-8位）发送信息。因此，数据发送可能会需要不定数量的时钟周期，具体取决于要发送的数据量。

在没有更多数据需要发送时，主器件会停止发送时钟信号，并取消选择从器件。

每个与总线连接、但未通过其从选择线选择的从器件，都忽略时钟和数据发送信号，并且不发送自己的任何数据。

图32-2： SPI主/从器件连接与FIFO



注1：在某些模式下，如果发送FIFO为空，则将发送最近接收的数据字节
2：该图假设LSBF位清零（采用MSb优先的通信方式）。
如果LSBF置1，则将采用LSb优先的通信方式。

32.2 SPI寄存器

- SPI中断标志寄存器（SPIxINTF）
- SPI中断允许寄存器（SPIxINTE）
- SPI字节计数高字节和低字节寄存器（SPIxTCTH/L）
- SPI位计数寄存器（SPIxTWIDTH）
- SPI波特率寄存器（SPIxBAUD）
- SPI控制寄存器0（SPIxCON0）
- SPI控制寄存器1（SPIxCON1）
- SPI控制寄存器2（SPIxCON2）
- SPI FIFO状态寄存器（SPIxSTATUS）
- SPI接收缓冲区寄存器（SPIxRB）
- SPI发送缓冲区寄存器（SPIxTB）
- SPI时钟选择寄存器（SPIxCLKSEL）

SPIxCON0、SPIxCON1和SPIxCON2是SPI模块的控制寄存器。

SPIxSTATUS包含几个状态位，用于指示SPI模块以及接收和发送FIFO的状态。

SPIxBAUD和SPIxCLKSEL控制主模式下SPI模块的波特率发生器。SPIxCLKSEL选择使用的时钟源。SPIxBAUD配置该时钟使用的时钟分频比。关于波特率发生器的更多信息，请参见**第32.5.6节“主模式SPI时钟配置”**。

SPIxTxB和SPIxRxB分别是用于在SPI总线上发送和接收数据的发送和接收缓冲区寄存器。通过这两个缓冲区寄存器，可间接访问用于移入和移出数据的移位寄存器。这两个移位寄存器均访问两个字节的FIFO，允许在软件传输数据的间隔时间存储多个发送/接收数据。

SPIxTCTH:L寄存器对可对数据传输中的位数或字节数进行计数或控制。当BMODE = 1时，SPIxTCT值表示字节，SPIxTWIDTH值表示一个字节的位数。当BMODE = 0时，SPIxTCT值将与SPIxTWIDTH寄存器相结合来表示位。在只接收主模式（TXR = 0且RXR = 1）下，通过向SPIxTCT写入所需传输的位或字节值来启动数据传输。在发送主模式（TXR = 1）下，通过写入SPIxTxB寄存器来启动数据传输，在这种情况下，SPIxTCT是用于对传输的位和字节进行计数的递减计数器。

SPIxINTF和SPIxINTE分别是SPI特定中断的标志位和允许位。它们分别与PIR和PIE寄存器中的SPIxIF标志位和SPIxIE允许位相关联，当触发SPIxINTF/SPIxINTE寄存器中包含的任何中断时，将触发标志位。PIR/PIE寄存器还包含SPIxTXIF/SPIxTXIE位（它们是SPI发送中断的中断标志位和中断允许位）以及SPIxRXIF/SPIxRXIE位（它们是SPI接收中断的中断标志位和中断允许位）。

32.3 SPI工作模式

初始化SPI时需要指定几个选项。可以通过编程相应的控制位（SPIxCON0<2:0>、SPIxCON1<7:4>、SPIxCON1<2:0> 和 SPIxCON2<2:0>）来指定这些选项。这些控制位用于指定以下选项：

- 主模式（SCK作为时钟输出）
- 从模式（SCK作为时钟输入）
- 时钟极性（SCK的空闲状态）
- 输入、输出和从选择极性
- 数据输入采样阶段（数据输出时间的中间或末尾）
- 时钟边沿（在SCK的第一个/第二个边沿输出数据）
- 时钟速率（仅限主模式）
- 从选择模式（主/从模式）
- MSB优先或LSB优先
- 接收/发送模式
  - 全双工
  - 接收不发送
  - 发送不接收
- 传输计数器模式（发送不接收模式）

32.3.1 使能和禁止SPI模块

要使能串行外设，SPI使能位（SPIxCON0中的EN）必须置1。要复位或重新配置SPI模式，先将EN位清零，重新初始化SSPxCONx寄存器，然后再将EN位置1。将EN位置1可以使能SPI输入和输出：SDI、SDO、SCK(out)、SCK(in)、SS(out)和SS(in)。以上所有输入和输出均通过PPS选择引脚，因此必须将其功能正确映射到器件引脚才能正常工作（见**第17.0节“外设引脚选择（PPS）模块”**）。此外，SS(out)和SCK(out)所选择的引脚必须设置为输出（TRIS位必须为0）才能正确输出。清零SDO引脚的TRIS位将导致SPI模块始终控制该引脚，但对于SDO功能来说并不是必要的。（见**第32.3.5节“输入和输出极性位”**）。当EN位置1时，不得更改由下列寄存器选择的配置：

- SPIxBAUD
- SPIxCON1
- SPIxCON0（EN位清零除外）

清零EN位将中止正在进行的传输、禁止硬件将中断标志置1并复位FIFO占用率（有关FIFO的更多详细信息，请参见**第32.3.3节“发送和接收FIFO”**）。

32.3.2 BUSY位

当正在进行数据传输时，SPI模块会将SPIxCON2的BUSY位置1。用户可通过轮询此位来确定SPI模块的当前状态，以及了解通信何时完成。当BUSY位置1时，不得通过软件写入以下寄存器/位：

- SPIxTCNTH/L
- SPIxTWIDTH
- SPIxCON2
- SPIxSTATUS的CLRBF位

注： 此外，建议不要在BUSY位置1时读取SPIxTCNTH/L，因为这些寄存器中的值不能作为传输计数器的可靠指示。使用传输计数为零中断标志（SPIxINTF的TCZIF位）准确地确定传输计数器已达到0。

32.3.3 发送和接收FIFO

SPI模块的数据发送和接收由两个FIFO处理，一个用于接收，另一个用于发送（分别由SFR SPIxRXB和SPIxTXB寻址）。TXFIFO由软件写入，并由SPI模块读取以将数据移入SDO引脚。RXFIFO从SDI引脚移入数据时由SPI模块写入，并由软件读取。将SPIxSTATUS的CLRBF位置1会复位两个FIFO的占用率，清空两个缓冲区。禁止SPI模块也会复位FIFO。

注： TXFIFO占用率和RXFIFO占用率仅表示当前存储在每个FIFO中的字节数。本章使用这些值是为了说明这些FIFO的功能，实际上无法通过软件直接访问这些值。

SPIxRXB寄存器用于寻址接收FIFO，并且是只读的。读取该寄存器时将读取硬件所写入的第一个FIFO单元并减少RXFIFO占用率。如果FIFO为空，则读取该寄存器时将返回零值，并将SPIxSTATUS寄存器的RXRE（接收缓冲区读取错误）位置1。之后必须用软件清零RXRE位，以便正确反映读取错误的状态。如果RXFIFO已满，SPIxSTATUS寄存器的RXBF位将置1。当器件在SDI引脚上接收数据时，可通过硬件写入接收FIFO，此时占用率会增加，具体取决于模式和接收器设置（如表32-1中所述）。

SPIxTXB寄存器用于寻址发送FIFO，并且是只写的。写入该寄存器时将写入第一个空FIFO单元并增加占用率。如果FIFO已满，写入该寄存器不会影响数据，但会将SPIxSTATUS寄存器的TXWE位置1。如果TXFIFO为空，SPIxSTATUS的TXBE位将置1。当发生数据传输时，可以从写入的第一个FIFO单元读取数据，此时占用率会减少，具体取决于模式和发送器设置（如表32-1和**第32.6.1节“从模式发送选项”**所述）。

32.3.4 LSB优先和MSB优先操作

通常，SPI通信先输出最高有效位，但某些器件/总线可能不遵循此标准。在这种情况下，可使用LSBF位改变数据交换期间位移出的顺序。在主模式和从模式下，SPIxCON0的LSBF位控制数据先移出MSb还是LSb。清零该位（默认）会将数据配置为先传输MSb（此为传统SPI操作），而将该位置1会将数据配置为先传输LSb。

32.3.5 输入和输出极性位

SPIxCON1 有 3 个位可控制 SPI 输入和输出的极性。SDIP 位控制 SDI 输入的极性，SDOP 位控制 SDO 输出的极性，SSP 位控制从 $\overline{SS}$ 输入和主 SS 输出的极性。对于全部 3 个位而言，当该位清零时，输入或输出为高电平有效；当该位置 1 时，输入或输出为低电平有效。当 SPIxCON0 的 EN 位清零时，SS(out)和 SCK(out)都将恢复为无效状态（由其极性位指示）。SPIxCON0 的 EN 位清零时的 SDO 输出状态由多个因素决定。

- 当 SDO 引脚的相关 TRIS 位清零且 SPI 在发送后进入空闲状态时，SDO 输出将保持最后一位的电平。如果 EN 清零，SDO 引脚将恢复为空闲状态。
- 当 SDO 引脚的相关 TRIS 位置 1 时，从模式和主模式下的行为会有所不同。
  - 在从模式下，SDO 引脚在以下情况下处于三态：
  - 从选择无效，
  - SPIxCON0 的 EN 位清零，或者
  - SPIxCON2 的 TXR 位清零。
  - 在主模式下，SDO 引脚在 TXR = 0 时处于三态。当 TXR = 1 且 SPI 在发送后进入空闲状态时，SDO 输出将保持最后一位的电平。如果 EN 清零，SDO 引脚将恢复为空闲状态。

32.4 传输计数器

在所有主模式下，传输计数器可用于确定 SPI 将进行的数据传输（发送/接收）次数。传输计数器由 SPIxTCTH/L 寄存器组构成，并由 SPIxTWIDTH 寄存器进行部分控制。传输计数器主要有两种模式，具体由 SPIxCON0 寄存器的 BMODE 位决定。每种模式都使用 SPIxTCTH/L 和 SPIxTWIDTH 寄存器来确定传输的次数和大小。在这两种模式下，当传输计数器达到零时，TCZIF 中断标志置 1。

注： 在所有主模式下，如果 BMODE = 1，则传输计数器在发生传输时仍将递减计数，并可用于计数发送/接收的报文数，以及控制 SS(out) 并触发 TCZIF（在从模式下始终如此）。此外，当 BMODE = 1 时，主模式和从模式下可使用 SPIxTWIDTH 寄存器来确定 SPI 发送和接收的报文大小，即使传输计数器并非主动用于控制 SPI 模块发送/接收的报文数时也是如此。

32.4.1 总位计数模式（BMODE = 0）

在该模式下，将结合 SPIxTCTH/L 和 SPIxTWIDTH 来确定要传输的总位数。这些位将以 8 位递增的方式从发送 FIFO 装载/向接收 FIFO 装入，并且传输计数器将递减 8，直到剩余位的总数小于 8。如果有任何剩余的位（SPIxTWIDTH ≠ 0），发送 FIFO 将发出最后一条报文，其中超过剩余位的任何位都将被忽略。SPIxTWIDTH 是剩余的位数，但该值不会像 SPIxTCT 值那样改变。类似地，接收器将最后一个字节装入接收器 FIFO 中，并用零填充额外的位。SPIxCON0 的 LSBF 位决定忽略/填充该最后一个字节的高位还是低位。例如，当 LSBF = 0 且最后一次传输仅包含两位时，如果发送的最后一个字节是 5Fh，则接收器的 RXB 将包含 40h，即最后一个字节的高两位，其余低位用零填充。

在该模式下，无论 TXR 和 RXR 设置如何，SPI 主器件都仅在 SPIxTCT 值大于零时发送报文。在发送主模式下，传输将在数据写入 SPIxTXB 寄存器后或计数值写入 SPIxTCTL 寄存器后（以后发生者为准）启动。在仅接收主模式下，传输时钟在写入 SPIxTCTL 值后启动。传输时钟在接收 FIFO 已满时暂停，在 FIFO 被读取时恢复。

32.4.2 可变传输大小模式（BMODE = 1）

在该模式下，SPIxTWIDTH指定每个数据传输块的宽度（以位为单位）。SPIxTCTH/SPIxTCTL指定该位长度的传输的次数。如果SPIxTWIDTH = 0，每一块都是一个完整的数据字节。如果SPIxTWIDTH ≠ 0，则仅会从发送FIFO移出指定的位数，未使用的位将被忽略。接收到的数据传输到接收FIFO后，将用零填充未使用的位区域。SPIxCON0的LSBF位决定忽略/填充传输的高位还是低位。在该模式下，传输计数器为零仅会在“仅接收”模式下停止发送/接收报文。

注：	当BMODE = 1时，传输计数器（SPIxTCTH/L）可能会递减到零以下，但在“仅接收”主模式下，传输时钟将在传输计数器达到零时停止。

32.4.3 从模式下的传输计数器

在从模式下，当数据移入和移出SPI模块时，传输计数器仍会递减，但它不会控制数据传输。此外，在从模式下，BMODE位和传输计数器一起用于确定器件何时应查找从选择故障。当BMODE = 0时，如果从选择在数据字节传输期间从有效状态转换为无效状态，或者如果在发送最后一个字节的最后一位之前发生如上状态转换（SPIxTWIDTH ≠ 0时），则SSFLT位将置1。当BMODE = 1时，如果从选择在完成每次单独传输的最后一位之前从有效状态转换为无效状态，则SSFLT位将置1。请注意，SSFLT没有关联的中断，因此应在软件中确认状态转换。最好是在触发从选择结束中断（EOSIF）时执行此操作（见**第32.8.3.3节“从选择开始中断和从选择结束中断”**）。

32.5 主模式

在主模式下，器件控制SCK线，因此可启动数据传输并确定何时由从器件将数据广播到SPI总线上。

该器件的主模式可由TXR和RXR位配置为四种不同模式：

- 全双工模式
- 仅接收模式
- 仅发送模式
- 传输关闭模式

下面的表32-1对这些模式进行了说明。

表32-1： 主模式TXR/RXR设置

	TXR = 1	TXR = 0
RXR = 1	全双工模式 如果BMODE = 1，则在RxFIFO未满且TxFIFO非空时传输。 如果BMODE = 0，则在RXFIFO未满、TXFIFO非空且传输计数器非零时传输。	仅接收模式 在RxFIFO未满且传输计数器非零时传输 发送的数据为FIFO顶部的数据或最新接收的数据
RXR = 0	仅发送模式 如果BMODE = 1，则在TxFIFO非空时传输 如果BMODE = 0，则在TXFIFO非空且传输计数器非零时传输 不会存储接收到的数据	无传输

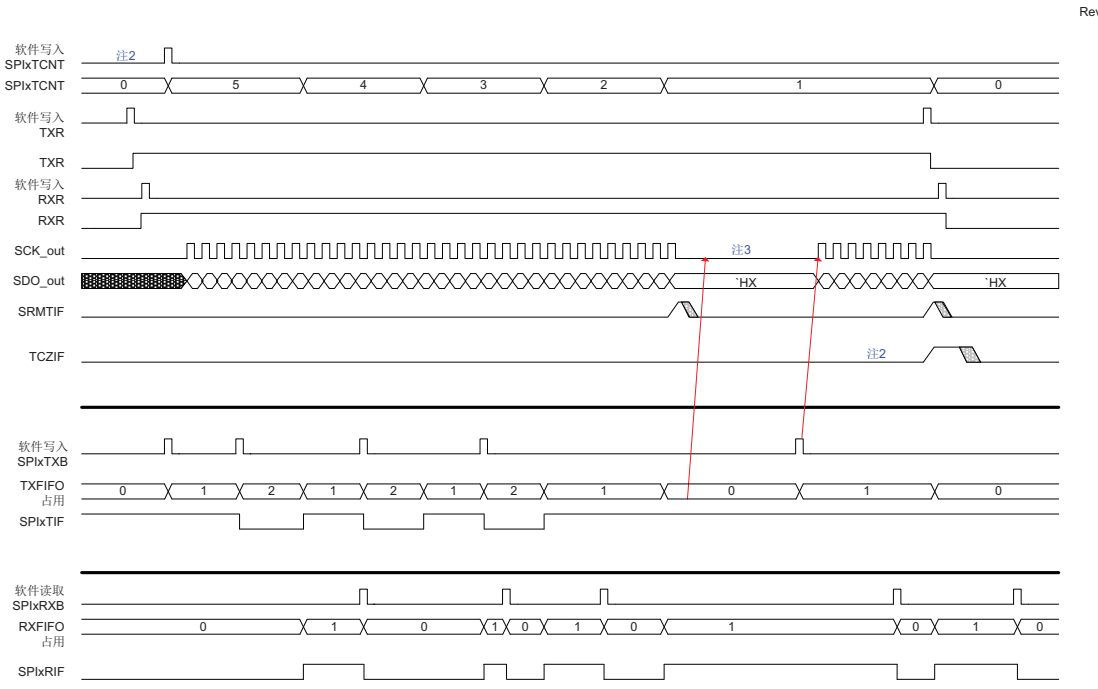
32.5.1 全双工模式

当TXR和RXR都置1时，SPI主器件处于全双工模式。在该模式下，数据传输触发受SPIxCON0的BMODE位影响。

当BMODE = 1时，只要RXFIFO未满并且TXFIFO中存在数据，就会发生数据传输。实际上，只要RXFIFO未满，就会在写入SPIxTxB寄存器时立即发送/接收数据，这与旧版8位Microchip器件上的SPI（MSSP）模块的功能相匹配。每次传输时，SPIxTCNT都会递减。不过，当SPIxTCNT为零时，不会禁止下一次传输，相应的SPIxTCNT递减将会导致计满返回至最大值。图32-3给出了使用该模式的通信示例。

当BMODE = 0时，传输计数器（SPIxTCNTH/SPIxTCNTL）还必须在传输之前写入，当传输计数器达到0时，传输将停止。例如，如果SPIxTXB写入两次并且SPIxTCTL写入“3”，则传输将以SPIxTCTL写入操作开始。将发送TXFIFO中的两个字节，之后传输将暂停，直到第三个字节和最后一个字节写入SPIxTXB。

图32-3： SPI主器件操作——数据交换，TXR/RXR = 1/1



注：1. 此图中未显示SS(out)
2. SPIxTCNT写入是可选的，TXR/RXR = 1/1，BMODE = 1。
如果BMODE = 0，则需要写入SPIxTCNT才能开始发送；TCZIF用于指示发送是否结束。
3. 等待发送器数据时出现发送间隙。

32.5.2 仅发送模式

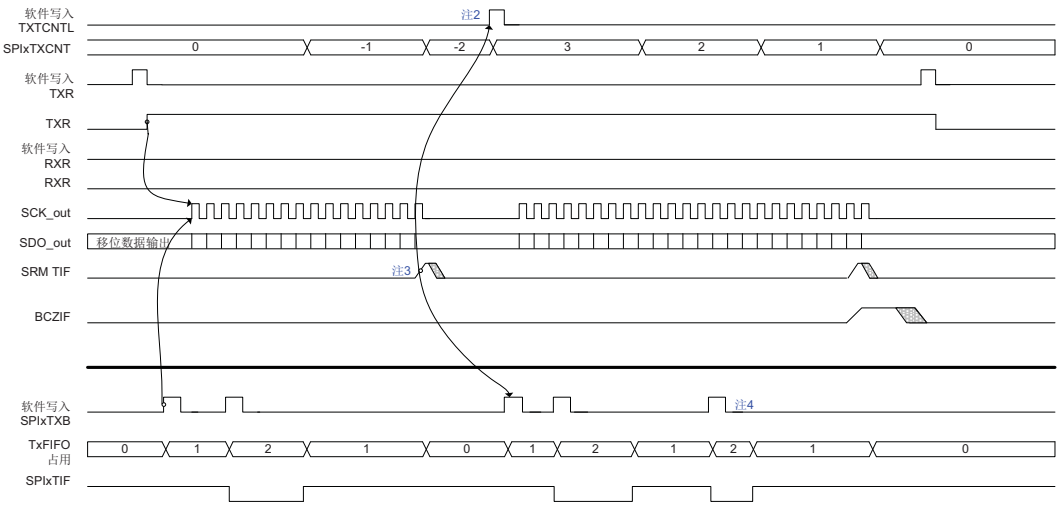
当TXR置1且RXR清零时，SPI主器件处于仅发送模式。在该模式下，数据传输触发受SPIxCON0的BMODE位影响。

当BMODE = 1时，只要TXFIFO非空，就会发生数据传输。写入TXFIFO寄存器时立即发送数据，这与旧版8位Microchip器件上的SPI（MSSP）模块的功能相匹配。每次传输时，SPIxTCNT都会递减。不过，当SPIxTCNT为零时，不会禁止下一次传输，相应的SPIxTCNT递减将会导致计满返回至最大值。该模式下接收的任何数据均不会存储在RXFIFO中。图32-4给出了使用该模式先后发送命令和数据字节的示例。

当BMODE = 0时，传输计数器（SPIxTCNTH/L）还必须在传输之前写入，当传输计数器达到0时，传输将停止。

例如，如果SPIxTXB写入两次并且SPIxTCTL写入“3”，则传输将以SPIxTCTL写入操作开始。将发送TXFIFO中的两个字节，之后传输将暂停，直到第三个字节和最后一个字节写入SPIxTXB。

图32-4： SPI主器件操作，命令+写数据，TXR/RXR = 1/0

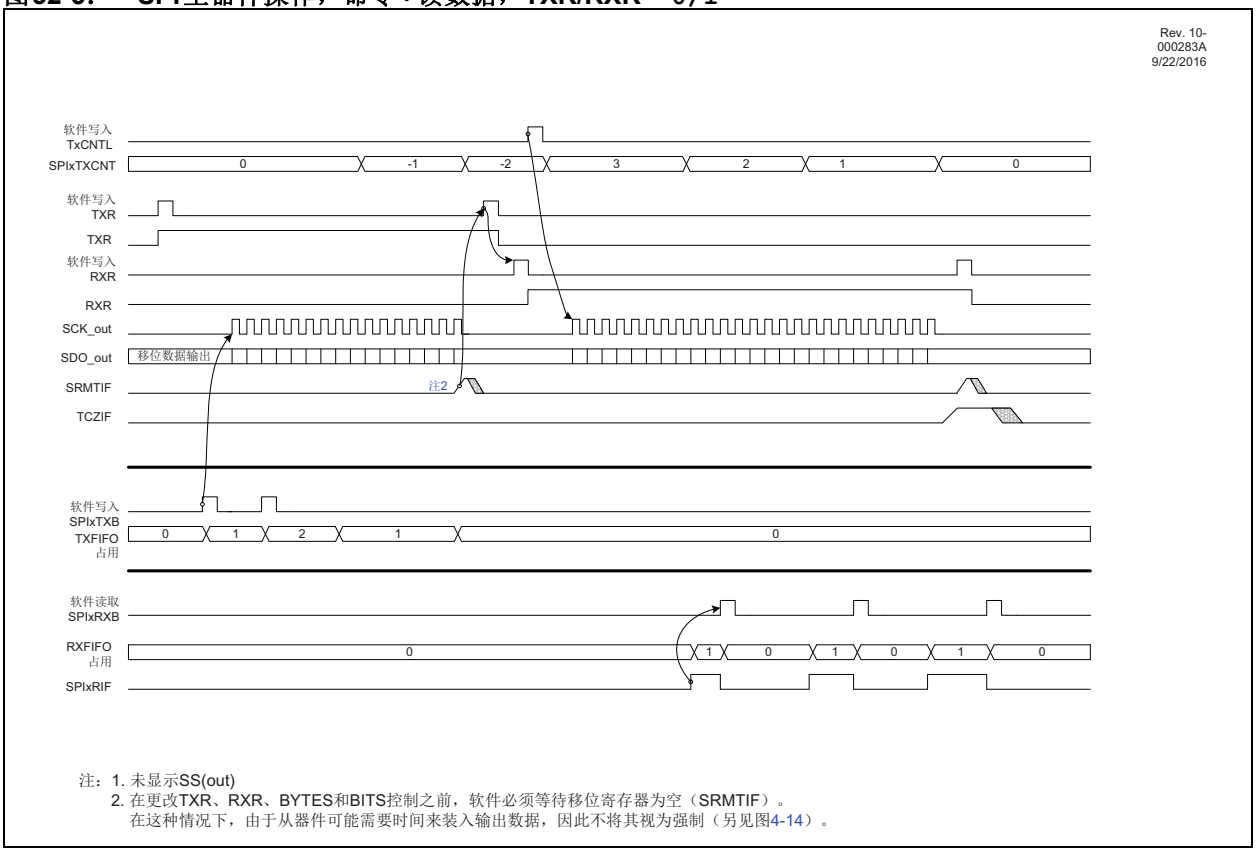


注：1. 未显示SS(out)
2. 当TXR/RXR = 1/0时，字节计数器是可选的；
3. 在命令字节之后，在装入SPIxBYTESL之前等待SRMTIF，否则命令字节将递减BYTES。
或者，装入BC = 5并计数命令字节；同样，TCZIF用于指示发送是否结束。
4. 发送数据中断处理程序（或DMA）必须只写入所需的字节；字节计数器不可用作指示器。
5. 由于RXR = 0，不需要读取RXFIFO。

32.5.3 仅接收模式

当RXR置1且TXR清零时，SPI主器件处于仅接收模式。在该模式下，如果RXFIFO未满并且传输计数器非零，则将发生数据传输。在该模式下，将值写入SPIxTCNTL将启动时钟传输。时钟将在RXFIFO已满时暂停，在SPIxTCNT达到0时停止（见**第32.4节“传输计数器”**）。如果TXFIFO中有数据，则每次数据交换时都将发送第一个写入TXFIFO的数据，但TXFIFO占用率不会改变，这意味着每次传输都会发送相同的报文。如果TXFIFO中没有数据，则会发送最新接收到的数据。图32-5给出了使用**第32.5.2节“仅发送模式”**发送命令然后使用该模式接收数据字节的示例。

图32-5： SPI主器件操作，命令+读数据，TXR/RXR = 0/1



注：1. 未显示SS(out)
2. 在更改TXR、RXR、BYTES和BITS控制之前，软件必须等待移位寄存器为空（SRMTIF）。
在这种情况下，由于从器件可能需要时间来装入输出数据，因此不将其视为强制（另见图4-14）。

32.5.4 传输关闭模式

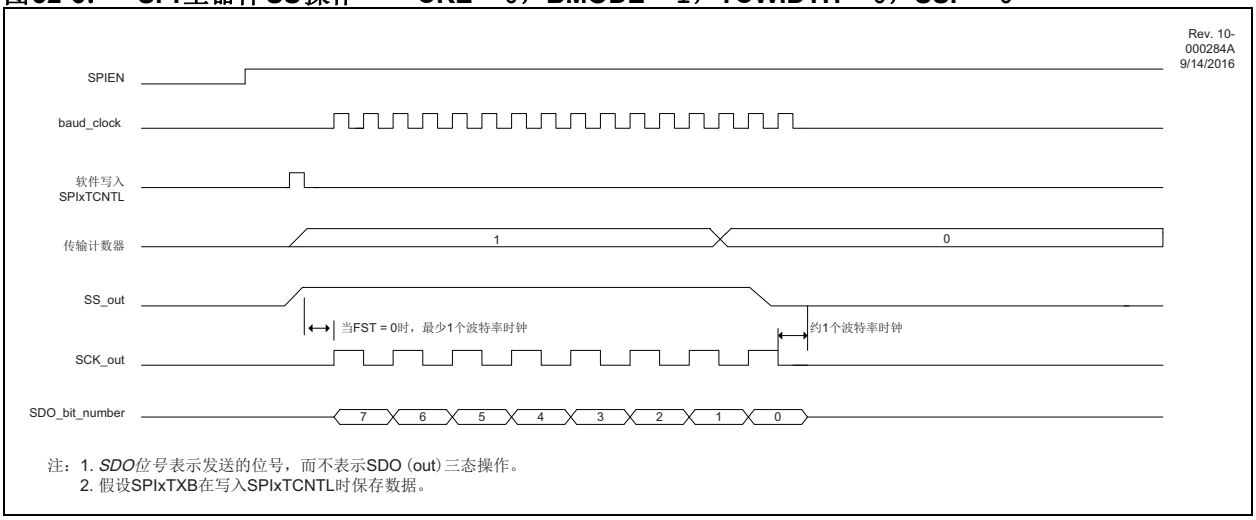
当TXR和RXR都清零时，SPI主器件处于传输关闭模式。在该模式下，SCK 将不会翻转，也不会交换数据。但是，写入SPIxTXB的内容将传输到TXFIFO，当TXR位置1时，将发送这些内容。

32.5.5 主模式从选择控制

32.5.5.1 硬件从选择控制

该SPI模块允许直接对从选择输出进行硬件控制。当传输计数器非零时，从器件选择输出SS(out)既可以通过SPIxCON2的SSET位直接控制，也可以通过硬件间接控制（见**第32.4节“传输计数器”**）。SS(out)通过PPS寄存器选择引脚（见**第17.2节“PPS输出”**），其极性由SPIxCON1的SSP位控制。将SSET位置1还会将SS(out)置为有效。清零SSET位将使SS(out)由传输计数器控制。装入传输计数器后，SPI模块会自动将SS置为有效。当传输计数器递减到零时，SPI模块将在最后一次传输的最后一个SCK脉冲之后再经过一个波特率周期（如果CKE/SMP = 0/1）或半个波特率周期（其他情况）将SS置为无效（见图32-6）。

图32-6： SPI主器件SS操作——CKE = 0，BMODE = 1，TCWIDTH = 0，SSP = 0



注：1. *SDO位号*表示发送的位号，而不表示SDO (out)三态操作。
2. 假设SPIxTXB在写入SPIxTCNTL时保存数据。

32.5.5.2 软件从选择控制

从选择还可由软件通过通用I/O引脚控制。在这种情况下，确保通过PPS将相关引脚配置为GPIO（见**第17.2节“PPS输出”**），并确保将该引脚设置为输出（将相应TRIS寄存器中的适当位清零）。在这种情况下，SSET不会影响从选择，传输计数器不会自动控制从选择输出，从选择输出线的所有置1和清零操作都必须由软件直接控制。

32.5.6 主模式SPI时钟配置

32.5.6.1 SPI时钟选择

SPI主模式的时钟源由SPIxCLK寄存器选择。时钟源选项包括：

- FOSC
- HFINTOSC
- CLKREF
- Timer0_overflow
- Timer2_Postscaled
- Timer4_Postscaled
- Timer6_Postscaled
- SMT_match

SPIxBAUD寄存器允许对该时钟进行分频。SCK输出的频率由公式32-1定义：

公式32-1： SCK输出信号的频率

$$F_{BAUD} = \frac{F_{CSEL}}{(2 \cdot (BAUD + 1))}$$

其中，FBAUD是SCK引脚上的波特率频率输出，FCSEL是SPIxCLK寄存器选择的输入时钟的频率，BAUD是SPIxBAUD寄存器中包含的值。

32.5.6.2 CKE、CKP和SMP

CKP、CKE和SMP位控制SCK时钟输出、SDO输出数据变化和SDI输入数据采样之间的关系。位功能如下：

- CKP——SCK输出极性
- CKE——SDO输出相对于SCK时钟的变化
- SMP——SDI输入相对于时钟边沿的采样情况

CKE位置1时，会将SCK输出的低电平空闲状态翻转为高电平空闲状态。

图32-7至图32-10给出了CKP、CKE和SMP位选择的八种可能组合。

当CKE位置1时，SDO数据在SCK上出现时钟边沿前一直有效。当CKE位清零时，SDO数据在第一个SCK边沿之前是不确定的。

注： 所有时序图均假设SPIxCON0的LSBF位清零。

图32-7： 时钟详细信息——主模式，CKE/SMP = 0/0

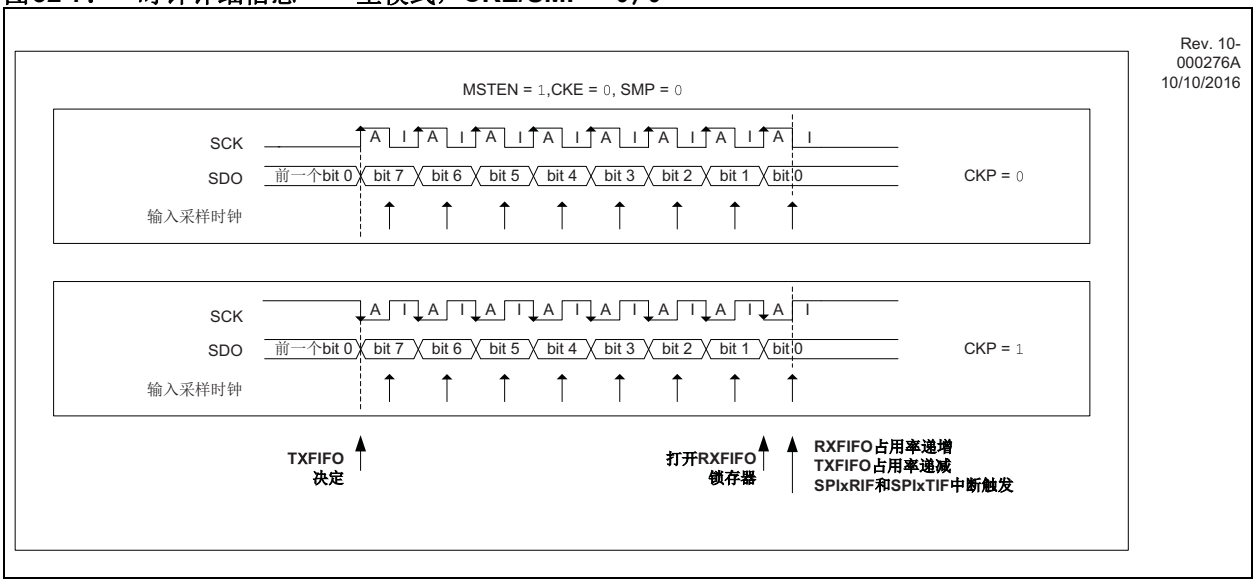


图32-8： 时钟详细信息——主模式，CKE/SMP = 1/1

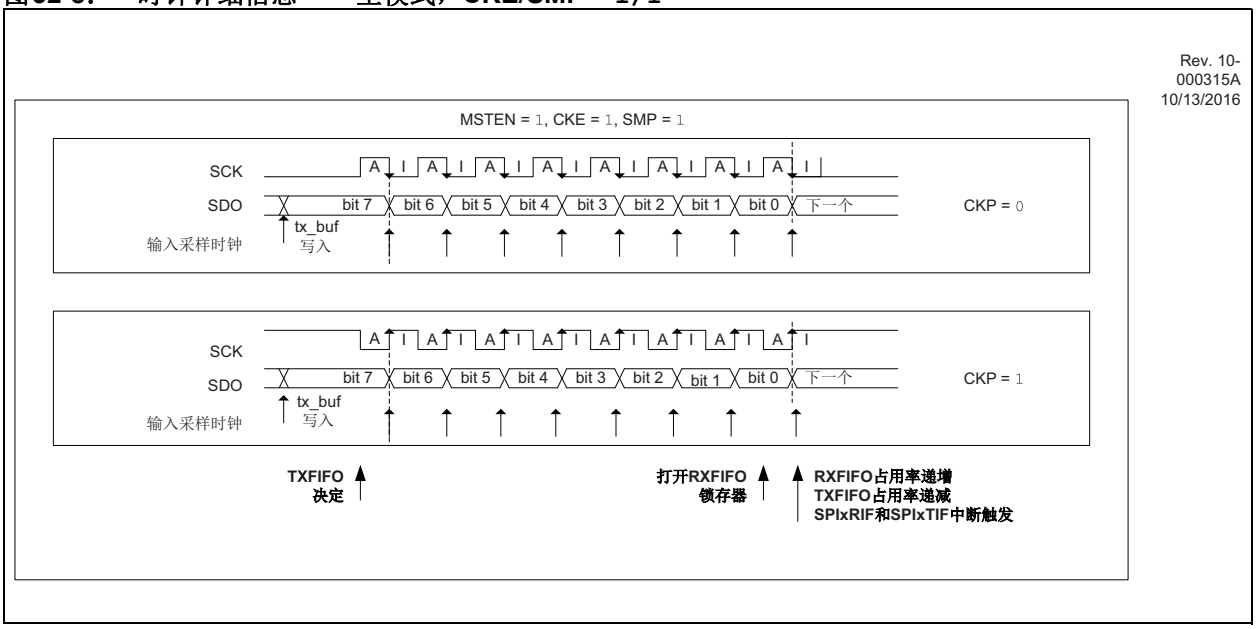


图32-9： 时钟详细信息——主模式，CKE = 0，SMP = 1

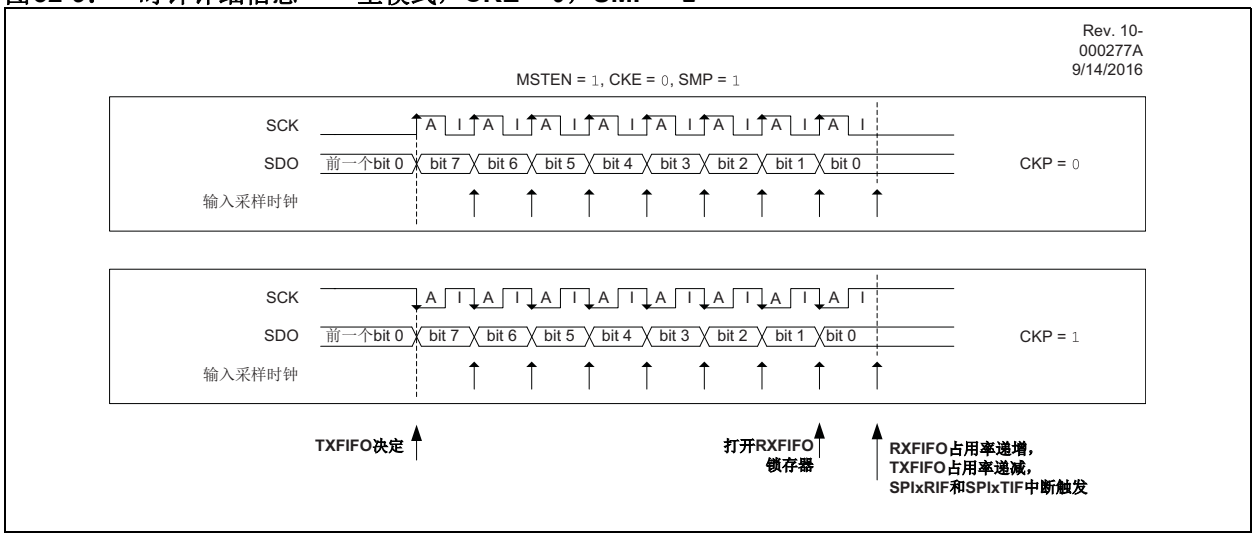
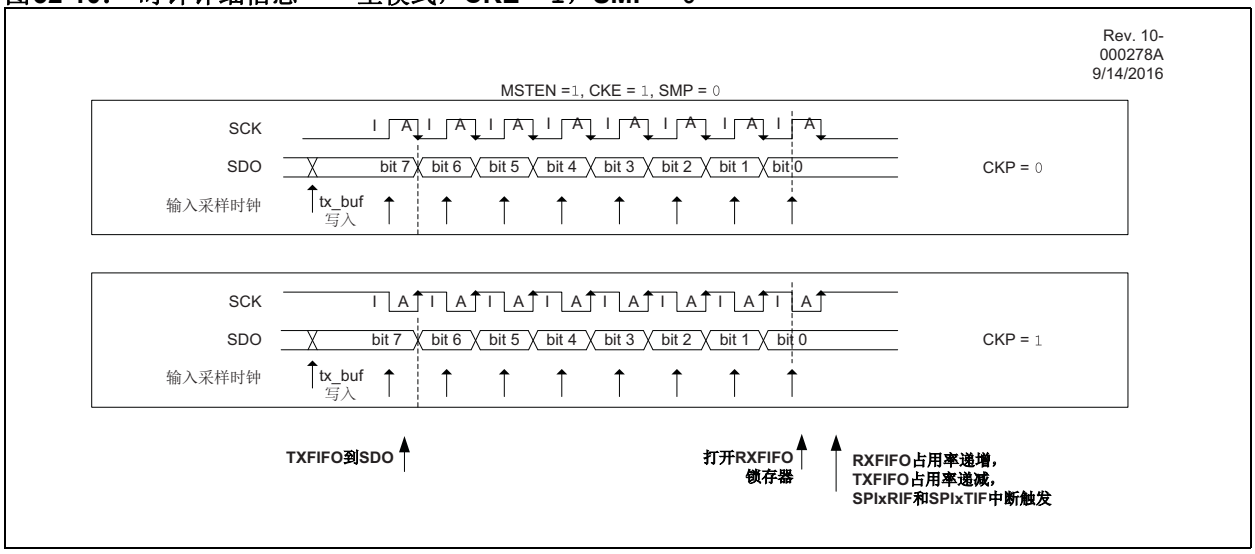


图32-10： 时钟详细信息——主模式，CKE = 1，SMP = 0



32.5.6.3 SCK启动延时

当启动SPI数据交换时，主器件会将SS输出置1（通过硬件或软件），然后触发模块发送数据。在第一个SCK脉冲出现之前，这些数据触发信号与SPIxCLK寄存器选择的时钟同步，通常需要所选时钟的一个或两个时钟。

SPI模块在SCK生成过程中有一段同步延时，专门用于确保从选择输出的时序正确无误，而无需使用精确的软件时序循环。

当SPIxBAUD寄存器的值较小（表示SCK频率较高）时，SS置1到第一个SCK之间的同步延时可能相对较长。当SPIxBAUD的值较大（表示SCK频率较低）时，此延时要短得多，并且在SS置1后，第一个SCK可以相对快速地出现。

默认情况下，SPI模块在第一个SCK脉冲之前插入½波特率延时（由SPIxCLK寄存器选择的时钟周期的一半）。这样可在第一个时钟之前为SPIxBAUD值较高的系统留出额外的设置时间。将SPIxCON1中的FST位置1会消除此额外延时，从而为SPIxBAUD值较低（即同步延时较长）的系统取消这种不必要的额外延时。

32.6 从模式

32.6.1 从模式发送选项

从模式下SPI模块的SDO输出由SPIxCON2的TXR位、与SDO引脚相关的TRIS位、从选择输入以及TXFIFO的当前状态控制。表32-2中对此控制进行了总结。在该表中，TRISxn指的是TRIS寄存器中与通过PPS为SDO分配的引脚相对应的位，TXR是SPIxCON2的需要发送数据控制位，SS是从选择输入的状态，TXBE是SPIxSTATUS的TXFIFO缓冲区空位。

32.6.1.1 SDO驱动/三态

与SDO引脚相关的TRIS位控制SDO引脚是否为三态。当该TRIS位清零时，即使SPI模块不工作，该引脚也将始终驱动至某个电平。当SPI模块不工作时（由于主模块不为SCK线路提供时钟或SS为假），SDO引脚将被驱动为与SDO引脚相关的LAT位的值。当SPI模块工作时，其输出将取决于TXR位设置以及TXFIFO中是否有数据。

如果与SDO引脚相关的TRIS位置1，则当TXR = 1且从选择输入为真时，该引脚仅驱动输出电平。在所有其他情况下，该引脚均处于三态。

32.6.1.2 SDO输出数据

TXR位控制从模式下发送的数据的性质。当TXR置1时，发送的数据从TXFIFO获取。如果FIFO为空，则发送最近接收的数据，并将TXUIF标志置1，以指示发生了发送FIFO下溢。

当TXR清零时，数据将从TXFIFO中获取，TXFIFO占用率不会降低。如果TXFIFO为空，则将发送最新接收到的数据，TXUIF位将不会置1。但是，如果与SDO引脚相关的TRIS位置1，则清零TXR位将导致SPI模块不向SDO引脚输出任何数据。

表32-2： 从模式发送

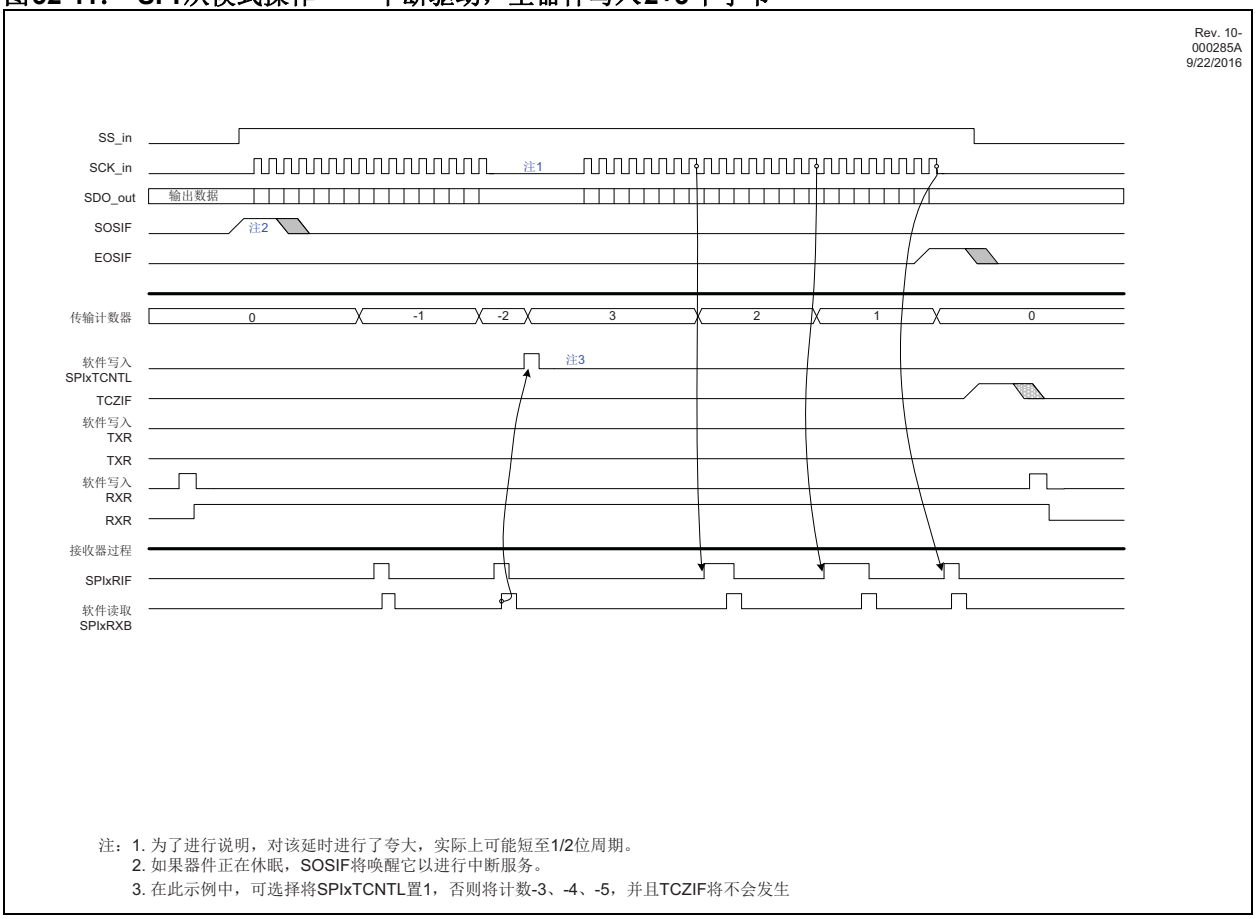
TRISxn(1)	TXR	SS	TXBE	SDO状态
0	0	假	0	驱动为由LATxn(2)确定的状态
0	0	假	1	驱动为由LATxn(2)确定的状态
0	0	真	0	输出TXFIFO中最早的字节 不删除TXFIFO中的数据
0	0	真	1	输出最新接收的字节
0	1	假	0	驱动为由LATxn(2)确定的状态
0	1	假	1	驱动为由LATxn(2)确定的状态
0	1	真	0	输出TXFIFO中最早的字节 删除从TXFIFO发送的字节 减少TXFIFO的占用率
0	1	真	1	输出最新接收的字节 将SPIxINTF的TXUIF位置1
1	0	假	0	三态
1	0	假	1	三态
1	0	真	0	三态
1	0	真	1	三态
1	1	假	0	三态
1	1	假	1	三态
1	1	真	0	输出TXFIFO中最早的字节 删除从TXFIFO发送的字节 减少TXFIFO的占用率
1	1	真	1	输出最新接收的字节 将SPIxINTF的TXUIF位置1

注 1： TRISxn是TRISx寄存器中与通过PPS为SDO分配的引脚相对应的位。
2： LATxn是LATx寄存器中与通过PPS为SDO分配的引脚相对应的位。

32.6.2 从模式接收选项

RXR位控制从模式下接收的性质。当RXR置1时，如果RXFIFO未满，则SDI输入数据将存储在其中。如果RXFIFO已满，则RXOIF位将置1以指示RXFIFO溢出错误，且数据会被丢弃。当RXR清零时，接收的所有数据都将被忽略并且不会存储在RXFIFO中（尽管TXFIFO为空时，它仍可用于传输）。图32-11所示为典型从模式通信，其中显示了主器件先写入2个字节、再写入3个字节的情况，还显示了中断以及从模式下传输计数器的行为（有关从模式下传输计数器的更多详细信息，请参见**第32.4.3节“从模式下的传输计数器”**；有关中断的更多信息，请参见**第32.8节“SPI中断”**）。

图32-11： SPI从模式操作——中断驱动，主器件写入2+3个字节



注：1. 为了进行说明，对该延时进行了夸大，实际上可能短至1/2位周期。
2. 如果器件正在休眠，SOSIF将唤醒它以进行中断服务。
3. 在此示例中，可选择将SPIxTCNTL置1，否则将计数-3、-4、-5，并且TCZIF将不会发生

32.6.3 从模式从选择

在从模式下，可以使用外部从选择信号来同步与主器件的通信。从选择线保持无效状态（默认为高电平），直到主器件准备好进行通信。当从选择转换为有效状态时，从器件就知道新的数据发送正在启动。

当从选择在数据发送结束时变为假时，所选SPI从器件的接收功能将恢复为无效状态。然后，从器件会在从选择再次变为真时准备好接收新的发送数据。

从选择信号在$\overline{SS}$输入引脚上接收。该引脚可通过SPIxSSPPS寄存器重映射（见**第17.1节“PPS输入”**）。当该引脚上的输入为真时，使能数据的发送和接收，同时驱动SDO引脚。当该引脚上的输入为假时，SDO引脚处于三态（如果与SDO引脚相关的TRIS位置1）或驱动为与SDO引脚相关的LAT位的值（如果与SDO引脚相关的TRIS位清零）。此外，SCK输入将被忽略。

如果SS输入变为假，而数据传输仍在进行中，则视为从选择故障。SPIxCON2的SSFLT位指示是否发生了此类事件。传输计数器值确定有效数据传输中的位数（更多详细信息，请参见**第32.4节“传输计数器”**）。

从选择极性由SPIxCON1的SSP位控制。当SSP置1（其默认状态）时，从选择输入为低电平有效；当SSP清零时，从选择输入为高电平有效。

SPI模块的从选择由SPIxCON2的SSET位控制。当该位清零（其默认状态）时，从选择将按上述方式执行操作。当该位置1时，SPI模块将按SS输入始终处于有效状态执行操作。

注：	当SSET置1时，有效SS(in)信号始终处于有效状态。因此，可以忽略SSFLT位。

32.6.4 从模式时钟配置

在从模式下，SCK为输入，必须配置为与主器件相同的极性和时钟边沿。与主模式一样，时钟输入的极性由SPIxCON1的CKP位控制，用于发送数据的时钟边沿由SPIxCON1的CKE位控制。

32.6.5 菊花链配置

SPI总线可采用菊花链配置进行连接。第一个从器件的输出与第二个从器件的输入连接，第二个从器件的输出与第三个从器件的输入连接，依此类推。最后一个从器件的输出与主器件的输入连接。在第二组时钟脉冲期间，每个从器件会送出在第一组时钟脉冲期间所接收数据的精确副本。整个链充当一个大的通信移位寄存器。菊花链功能只需将主器件的一条从选择线连接到所有从器件（也可以通过将SSET位置1将从器件配置为忽略从选择线）。在典型的菊花链配置中，来自主器件的SCK信号连接到每个从器件SCK输入。不过，SCK输入和输出是由PPS控制选择的独立信号。当通过PPS选择在单独的引脚上配置SCK输入和SCK输出时，SCK输出将跟随SCK输入，允许SCK信号像SDO/SDI信号一样采用菊花链配置。

图32-12所示为典型菊花链连接的框图，图32-13所示为使用此SPI模块可实现的菊花链连接的框图。

图32-12： 典型SPI菊花链连接

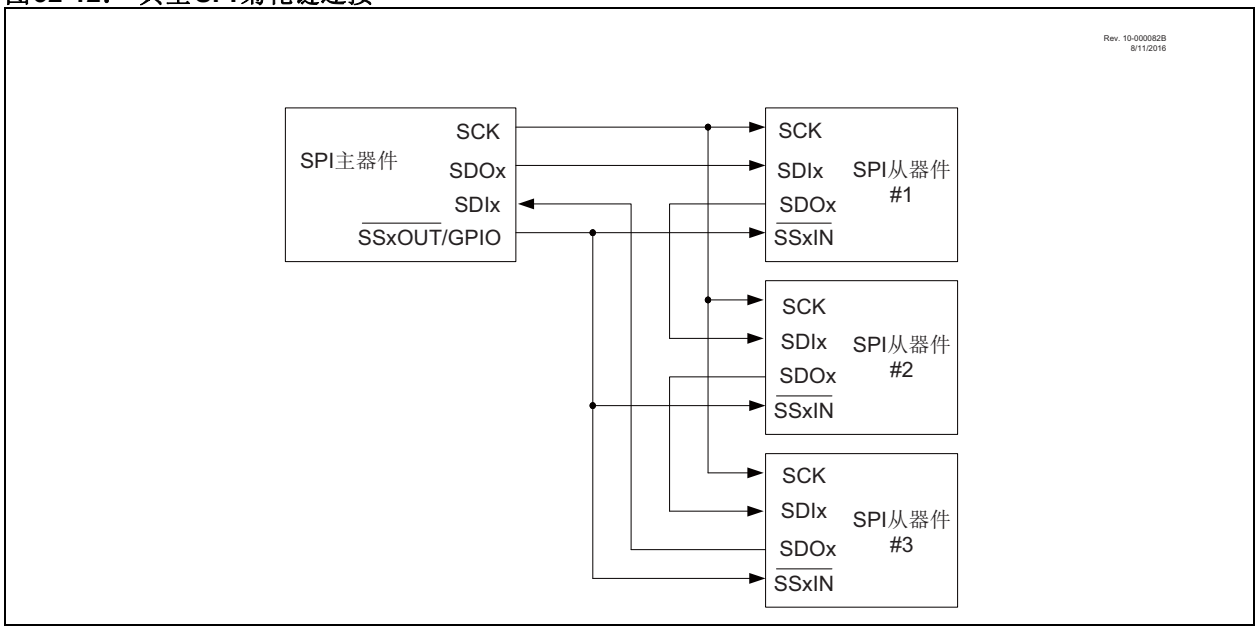
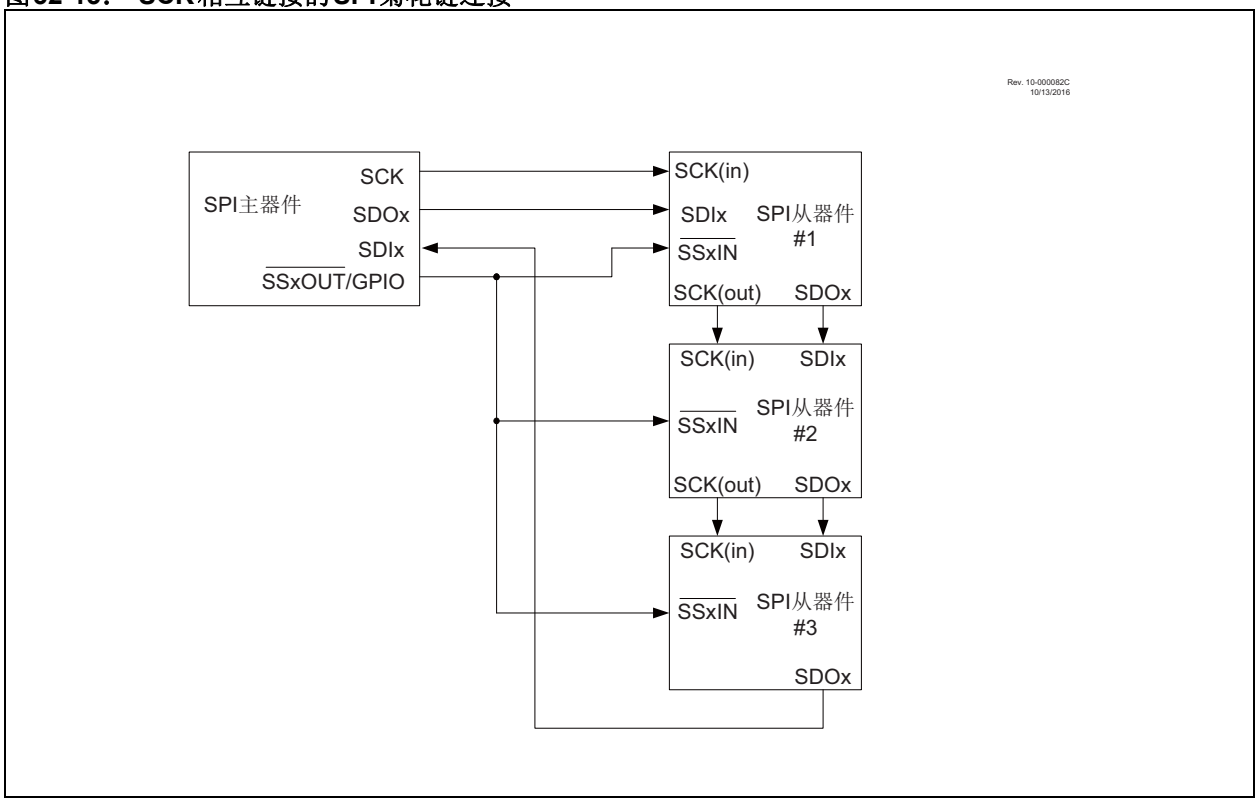


图32-13： SCK相互链接的SPI菊花链连接



32.7 休眠模式下的SPI操作

如果SPIxCLK选择的时钟源在休眠模式下有效，则SPI主模式将在休眠模式下工作。当器件唤醒时，FIFO将按预期工作。当TXR = 1时，TXFIFO需要包含数据才能在休眠模式下进行传输。所有中断在休眠状态下仍会将中断标志置1，但只有允许的中断才能将器件从休眠状态唤醒。

SPI从模式在休眠状态下仍可工作，因为时钟由外部主器件提供。FIFO仍将工作，中断会将中断标志置1，允许的中断会将器件从休眠状态唤醒。

32.8 SPI中断

PIRx寄存器中有三个顶级SPI中断：

- SPI发送
- SPI接收
- SPI模块状态

在SPIxINTE寄存器的模块级允许状态中断。只有允许的状态中断会将单个顶级SPIxIF标志置1。

32.8.1 SPI接收器数据中断

SPI接收器数据中断在RXFIFO包含数据时置1，在RXFIFO为空时清零。中断标志SPI1RXIF位于PIRx中，中断允许SPI1RXIE位于PIEx中。该中断标志是只读的。

32.8.2 SPI发送器数据中断

SPI发送器数据中断在TXFIFO未满时置1，在TXFIFO已满时清零。中断标志SPI1TXIF位于PIRx中，中断允许SPI1TXIE位于PIEx中。该中断标志是只读的。

32.8.3 SPI模块状态中断

当SPIxINTF中的任何单独状态标志及其各自的SPIxINTE位置1时，相应PIR寄存器中的SPIxIF标志置1。为了通过将任何特定的中断标志置1来中断正常的程序流程，必须将SPIxIE位以及SPIxINTE中与该中断相关的特定位置1。

状态中断包括：

- 移位寄存器空中断
- 传输计数器为零中断
- 从选择开始中断
- 从选择结束中断
- 接收器上溢中断
- 发送器下溢中断

32.8.3.1 移位寄存器空中断

移位寄存器空中断标志位和允许位分别是SRMTIF和SRMTIE位。该中断仅在主模式下可用，在数据传输完成且没有启动新传输的条件时（由TXR和RXR位指示）触发（有关不同TXR/RXR设置下启动新主模式数据传输的条件的信息，请参见表32-1）。该中断将在最后一个完整的位周期结束时（在SCK保持1/2个波特率周期的低电平状态后）触发。有关此中断以及其他中断的时序的更多详细信息，请参见图32-14。当启动新传输的条件发生时，该位不会自行清零，必须用软件清零。

32.8.3.2 传输计数器为零中断

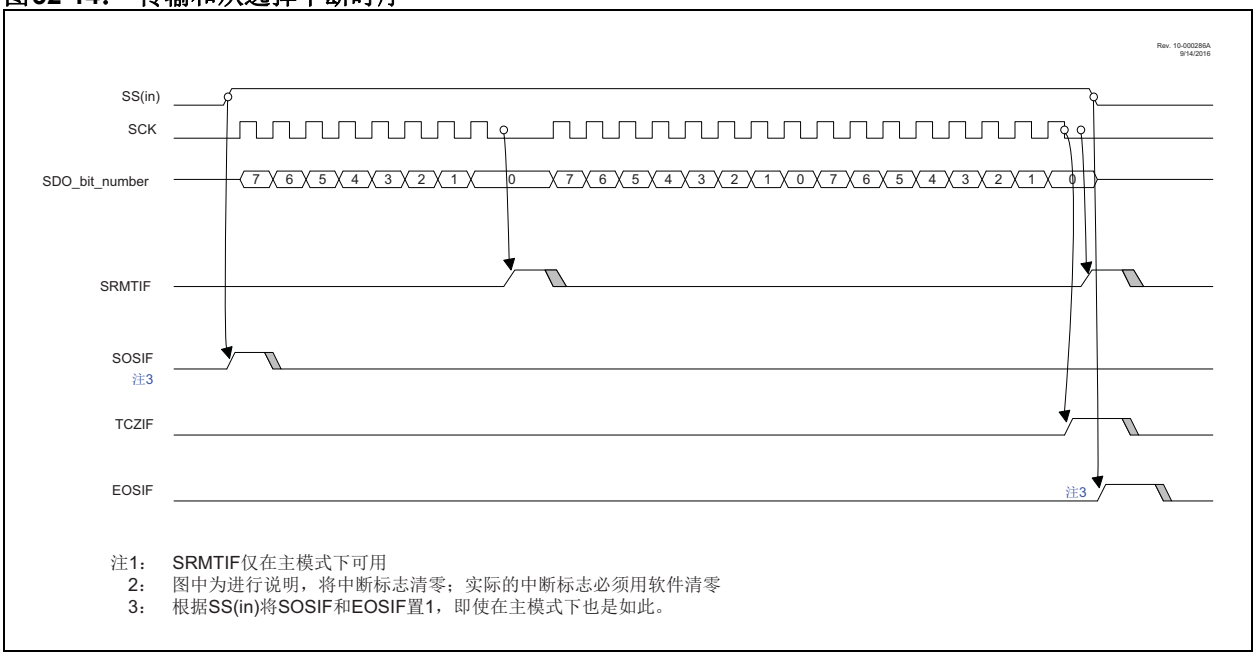
传输计数器为零中断标志位和允许位分别是TCZIF和TCZIE位。当传输计数器（由BMODE、SPIxTCTH/L和SPIxTWIDTH定义）从1递减到0时，将触发该中断。有关此中断以及其他中断的时序的更多详细信息，请参见图32-14。此位必须用软件清零。

注： TCZIF标志仅指示传输计数器从1递减到0，不能指示整个数据传输过程已完成。可通过轮询SPIxCON2的BUSY位并等待它被清零或者使用移位寄存器空中断（SRMTIF）来确定数据传输是否全部完成。

32.8.3.3 从选择开始中断和从选择结束中断

从选择开始中断标志位和允许位分别是SOSIF和SOSIE位；相似地，从选择结束中断标志位和允许位分别是EOSIF和EOSIE位。这些中断在从选择输入的前沿和后沿触发。请注意，中断在主模式和从模式下均有效，无论SPI处于哪种模式，从选择输入发生转换时都会触发中断。在主模式下，应使用PPS将从选择输入路由到与从选择输出相同的引脚，从而允许在从选择输出发生变化时触发这些中断。另请注意，在从模式下，更改SSET位时会更改从选择的有效输入值，并因此触发这些中断。SOSIF和EOSIF均必须用软件清零。

图32-14： 传输和从选择中断时序



注1： SRMTIF仅在主模式下可用
2： 图中为进行说明，将中断标志清零；实际的中断标志必须用软件清零
3： 根据SS(in)将SOSIF和EOSIF置1，即使在主模式下也是如此。

32.8.3.4 接收器上溢中断和发送器下溢中断

如果在RXFIFO已满且RXR = 1时接收到数据，则会触发接收器上溢中断。在这种情况下，数据将被丢弃，RXOIF位将置1。接收器上溢中断标志位是SPIxINTF的RXOIF位。接收器上溢中断允许位是SPIxINTE的RXOIE位。

如果在TXFIFO为空且TXR = 1时开始数据传输，则会触发发送器下溢中断标志位。在这种情况下，最新接收的数据将被发送，TXUIF位将置1。发送器下溢中断标志位是SPIxINTF的TXUIF位。发送器下溢中断允许位是SPIxINTE的TXUIE位。

这两个中断仅在从模式下发生，因为主模式不允许RXFIFO上溢或TXFIFO下溢。

32.9 寄存器定义：SPI

寄存器32-1： SPIxINTF：SPI中断标志寄存器

R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	U-0	R/W/HS-0/0	R/W/HS-0/0	U-0
SRMTIF	TCZIF	SOSIF	EOSIF	—	RXOIF	TXUIF	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0 HS = 硬件置1位

bit 7 **SRMTIF**：移位寄存器空中断标志位

从模式：

该位被忽略

主模式：

1 = 数据传输完成

0 = 未发生数据传输或正在进行数据传输

bit 6 **TCZIF**：传输计数器为零中断标志位

1 = 传输计数器（由寄存器32-7中的BMODE、TCNTH/L和TWIDTH定义）递减为0。

0 = 没有待处理中断

bit 5 **SOSIF**：从选择开始中断标志位

1 = SS(in)从假转变为真

0 = 没有待处理中断

bit 4 **EOSIF**：从选择结束中断标志位

1 = SS(in)从真转变为假

0 = 没有待处理中断

bit 3 **未实现**：读为0

bit 2 **RXOIF**：接收器上溢中断标志位

1 = RXBF = 1（边沿触发）且RXR = 1时，数据传输完成

0 = 没有待处理中断

bit 1 **TXUIF**：发送器下溢中断标志位

1 = TXBE = 1且TXR = 1时，从器件数据传输开始

0 = 没有待处理中断

bit 0 **未实现**：读为0

寄存器32-2： SPIxINTE：SPI中断允许寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0
SRMTIE	TCZIE	SOSIE	EOSIE	—	RXOIE	TXUIE	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7 **SRMTIE**：移位寄存器空中断允许位
1 = 允许移位寄存器空中断
0 = 禁止移位寄存器空中断

bit 6 **TCZIE**：传输计数器为零中断允许位
1 = 允许传输计数器为零中断
0 = 禁止传输计数器为零中断

bit 5 **SOSIE**：从选择开始中断允许位
1 = 允许从选择开始中断
0 = 禁止从选择开始中断

bit 4 **EOSIE**：从选择结束中断允许位
1 = 允许从选择结束中断
0 = 禁止从选择结束中断

bit 3 **未实现**：读为0

bit 2 **RXOIE**：接收器上溢中断允许位
1 = 允许接收器上溢中断
0 = 禁止接收器上溢中断

bit 1 **TXUIE**：发送器下溢中断允许位
1 = 允许发送器下溢中断
0 = 禁止发送器下溢中断

bit 0 **未实现**：读为0

寄存器32-3： SPIxTCNTL——SPI传输计数器LSB寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7-0 **TCNT<7:0>**：
BMODE = 0
传输计数器的bit 10-3，统计要传输的总位数
BMODE = 1
传输计数器的bit 7-0，统计要传输的总字节数

注： 正在进行传输时（SPIxCON2的BUSY位置1），不应写入该寄存器。

寄存器32-4： SPIxTCNTH：SPI传输计数器MSB寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	TCNT10	TCNT9	TCNT8
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7-3 **未实现**：读为0

bit 2-0 **TCNT<10:8>**：

BMODE = 0

传输计数器的bit 13-11，统计要传输的总位数

BMODE = 1

传输计数器的bit 10-8，统计要传输的总字节数

注： 正在进行传输时（SPIxCON2的BUSY位置1），不应写入该寄存器。

寄存器32-5： SPIxTWIDTH：SPI传输宽度寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	TWIDTH2	TWIDTH1	TWIDTH0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7-3 **未实现**：读为0

bit 2-0 **TWIDTH<2:0>**：

BMODE = 0

传输计数器的bit 2-0，统计要传输的总位数

BMODE = 1

传输计数器计数的每次传输的大小（以位为单位）

111 = 7位

110 = 6位

101 = 5位

100 = 4位

011 = 3位

010 = 2位

001 = 1位

000 = 8位

注： 正在进行传输时（SPIxCON2的BUSY位置1），不应写入该寄存器。

寄存器32-6： SPIxBAUD：SPI波特率寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
BAUD7	BAUD6	BAUD5	BAUD4	BAUD3	BAUD2	BAUD1	BAUD0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7-0 **BAUD<7:0>**：波特率时钟预分频比选择位

SCK高电平或低电平时间：TSC = SPI时钟周期*(波特率+1)

SCK翻转频率：FSCK = FBAUD = SPI时钟频率/(2*(波特率+1))

注： SPI使能时（SPIxCON0的EN位 = 1），不应写入该寄存器

寄存器32-7： SPIxCON0：SPI配置寄存器0

R/W-0/0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
EN	—	—	—	—	LSBF	MST	BMODE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7 **EN**：SPI模块使能控制位

1 = 使能SPI

0 = 禁止SPI

bit 6-3 **未实现**：读为0

bit 2 **LSBF**：以LSB在前方式交换数据位

1 = 以LSb在前的方式交换数据

0 = 以MSb在前的方式交换数据（传统SPI操作）

bit 1 **MST**：SPI工作模式主选择位

1 = SPI模块作为总线主器件工作

0 = SPI模块作为总线从器件工作

bit 0 **BMODE**：位长度模式选择位

1 = SPIxTWIDTH设置适用于每个字节：发送的总位数为SPIxTWIDTH*SPIxTCNT，当SPIxTCNT = 0时数据包结束

0 = SPIxTWIDTH设置仅适用于最后交换的字节；发送的总位数是SPIxTWIDTH + (SPIxTCNT*8)

注： 该寄存器只能在EN位清零时或即将清零时写入。

寄存器32-8： SPIxCON1：SPI配置寄存器1

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-1/1	R/W-0/0	R/W-0/0
SMP	CKE	CKP	FST	—	SSP	SDIP	SDOP
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7 **SMP**：SPI输入采样相位控制位

从模式：

1 = 保留

0 = 在数据输出时间的中间采样SDI输入

主模式：

1 = 在数据输出时间的末尾采样SDI输入

0 = 在数据输出时间的中间采样SDI输入

bit 6 **CKE**：时钟边沿选择位

1 = 时钟状态从有效转换到空闲时，输出数据发生变化

0 = 时钟状态从空闲转换到有效时，输出数据发生变化

bit 5 **CKP**：时钟极性选择位

1 = SCK的空闲状态为高电平

0 = SCK的空闲状态为低电平

bit 4 **FST**：快速启动使能位

从模式：

该位被忽略

主模式：

1 = 到第一个SCK的延时可能小于½个波特率周期

0 = 到第一个SCK的延时将至少为½个波特率周期

bit 3 **未实现**：读为0

bit 2 **SSP**：SS输入/输出极性控制位

1 = SS为低电平有效

0 = SS为高电平有效

bit 1 **SDIP**：SDI输入极性控制位

1 = SDI输入为低电平有效

0 = SDI输入为高电平有效

bit 0 **SDOP**：SDI输出极性控制位

1 = SDO输出为低电平有效

0 = SDO输出为高电平有效

寄存器32-9：　SPIxCON2：SPI配置寄存器2

R-0/0	R-0/0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
BUSY	SSFLT	—	—	—	SSET	TXR(1)	RXR(1)
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7　**BUSY**：SPI模块繁忙状态位

1 = 数据交换繁忙

0 = 未发生数据交换

bit 6　**SSFLT**：SS(in)故障状态位

如果SSET = 0

1 = SS(in)意外结束了事务，并且丢失了正在接收的数据字节

0 = SS(in)正常结束

如果SSET = 1

该位不变。

bit 5-3　**未实现**：读为0

bit 2　**SSET**：从选择使能位

主模式：

1 = SS(out)连续驱动为有效状态

0 = 发送计数器不为0时，SS(out)驱动为有效状态

从模式：

1 = SS(in)被忽略，数据的时钟驱动由所有SCK(in)提供（就像一直处于“SS = 真”状态一样）

0 = SS(in)使能/禁止数据输入，并在与SDO引脚相关的TRIS位置1时将SDO置于三态（有关详细信息，请参见表32-2）

bit 1　**TXR**：需要发送数据控制位(1)

1 = 传输需要TxFIFO数据

0 = 传输不需要TxFIFO数据

bit 0　**RXR**：需要接收FIFO空间控制位(1)

1 = 如果RxFIFO已满，则数据传输暂停

0 = 不将接收的数据存储到FIFO中

注 1： 有关TXR和RXR功能的更多详细信息，请参见表32-1以及**第32.5节“主模式”**和**第32.6节“从模式”**。

2： 正在进行传输时（SPIxCON2的BUSY位置1），不应写入该寄存器。

寄存器32-10： SPIxSTATUS：SPI状态寄存器

R/C/HS-0/0	U-0	R-1/1	U-0	R/C/HS-0/0	S-0/0	U-0	R-0/0
TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0 C = 可清零位 S = 可置1位 HS = 硬件置1位

bit 7 **TXWE**：发送缓冲区写错误位
1 = 当TxFIFO已满时，写入SPIxTxB
0 = 未发生错误

bit 6 **未实现**：读为0

bit 5 **TXBE**：发送缓冲区空位（只读）
1 = 发送缓冲区TxFIFO为空
0 = 发送缓冲区不为空

bit 4 **未实现**：读为0

bit 3 **RXRE**：接收缓冲区读错误位
1 = 当RxFIFO为空时，读取SPIxRB
0 = 未发生错误

bit 2 **CLRBF**：清除缓冲区控制位（只写）
1 = 复位接收和发送缓冲区，使两个缓冲区都为空
0 = 不采取任何操作

bit 1 **未实现**：读为0

bit 0 **RXBF**：接收缓冲区满位（只读）
1 = 接收缓冲区已满
0 = 接收缓冲区未满

寄存器32-11： SPIxRxB：SPI读缓冲区寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7-0 **RXB<7:0>**：接收缓冲区位（只读）
如果RX缓冲区不为空：
包含RXFIFO的最顶层字节，读取该寄存器将删除最顶层字节RXFIFO，并减少RXFIFO的占用率
如果RX缓冲区为空：
读取该寄存器时的结果为0，占用率保持不变，并将SPIxSTATUS的RXRE位置1

寄存器32-12： SPIxTxB：SPI发送缓冲区寄存器

W-0	W-0	W-0	W-0	W-0	W-0	W-0	W-0
TXB7	TXB6	TXB5	TXB4	TXB3	TXB2	TXB1	TXB0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7-0 **TXB<7:0>**：发送缓冲区位（只写）

如果TXFIFO未满：

写入该寄存器会将数据添加到TXFIFO的顶部，并增加TXFIFO写指针的占用率

如果TXFIFO已满：

写入该寄存器不会影响TXFIFO中的数据或写指针，SPIxSTATUS的TXWE位将置1

寄存器32-13： SPIxCLK：SPI时钟选择寄存器

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	CLKSEL3	CLKSEL2	CLKSEL1	CLKSEL0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0

bit 7-4 **未实现**：读为0

bit 3-0 **CLKSEL<3:0>**：SPI时钟源选择位

1111-1001 = 保留
1000 = SMT_match
0111 = TMR6_Postscaled
0110 = TMR4_Postscaled
0101 = TMR2_Postscaled
0100 = TMR0_overflow
0011 = CLKREF
0010 = MFINTOSC
0001 = HFINTOSC
0000 = FOSC

表32-3： 与SPI相关的寄存器汇总

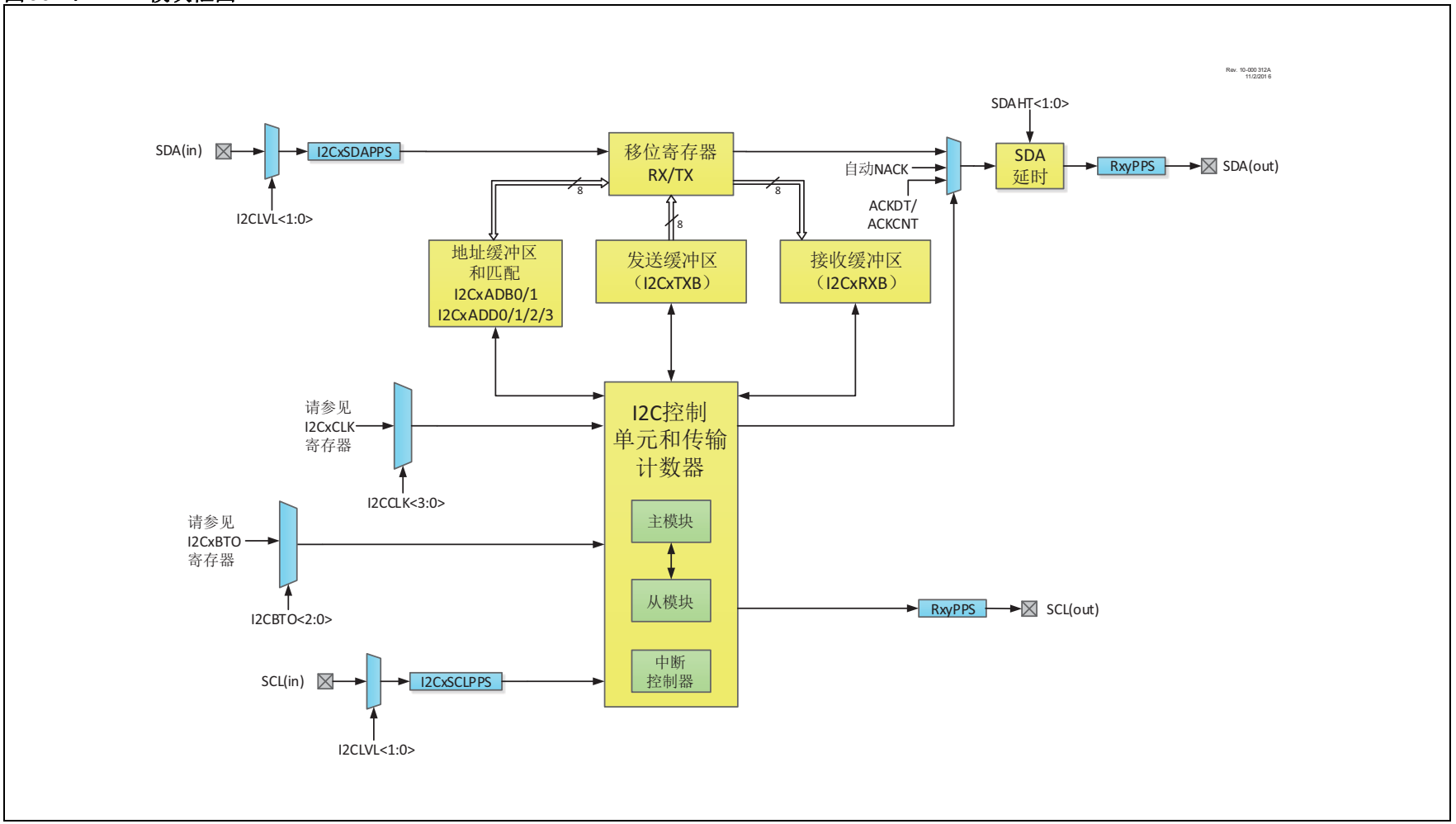
名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
SPIxINTF	SRMTIF	TCZIF	SOSIF	EOSIF	—	RXOIF	TXUIF	—	520
SPIxINTE	SRMTIE	TCZIE	SOSIE	EOSIE	—	RXOIE	TXUIE	—	521
SPIxTCNTH	—	—	—	—	—	TCNT10	TCNT9	TCNT8	522
SPIxTCNTL	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0	521
SPIxTWIDTH	—	—	—	—	—	TWIDTH2	TWIDTH1	TWITDH0	522
SPIxBAUD	BAUD7	BAUD6	BAUD5	BAUD4	BAUD3	BAUD2	BAUD1	BAUD0	523
SPIxCON0	EN	—	—	—	—	LSBF	MST	BMODE	523
SPIxCON1	SMP	CKE	CKP	FST	—	SSP	SDIP	SDOP	524
SPIxCON2	BUSY	SSFLT	—	—	—	SSET	TXR	RXR	525
SPIxSTATUS	TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF	526
SPIxRXB	RXB7	RXB6	RXB5	RXB4	RXB3	RXB2	RXB1	RXB0	526
SPIxTXB	TXB7	TXB6	TXB5	TXB4	TXB3	TXB2	TXB1	TXB0	527
SPIxCLK	—	—	—	—	CLKSEL3	CLKSEL2	CLKSEL1	CLKSEL0	527

图注： — = 未实现，读为0。SPI模块不使用阴影单元。

33.0 I²C模块

该器件有两个专用的独立I²C模块。图33-1给出了I²C接口模块的框图。该图显示了主模式和从模式。

图33-1： I²C模块框图



33.1 I²C特性

- I²C接口的硬件支持以下模式：
  - 主模式
  - 从模式（支持字节无应答）
  - 多主器件模式
- 专用地址、接收和发送缓冲区
- 最多四个从地址匹配
- 广播呼叫地址匹配
- 带掩码的7位和10位寻址模式
- 启动、重复启动、停止、寻址、写入和应答中断
- 时钟延长硬件，适用于：
  - RX缓冲区已满
  - TX缓冲区为空
  - 寻址、写入和应答之后
- 带仲裁的总线冲突检测
- 总线超时检测
- SDA保持时间选择
- I²C、SMBus 2.0和SMBus 3.0输入电平选择

33.2 I²C模块概述

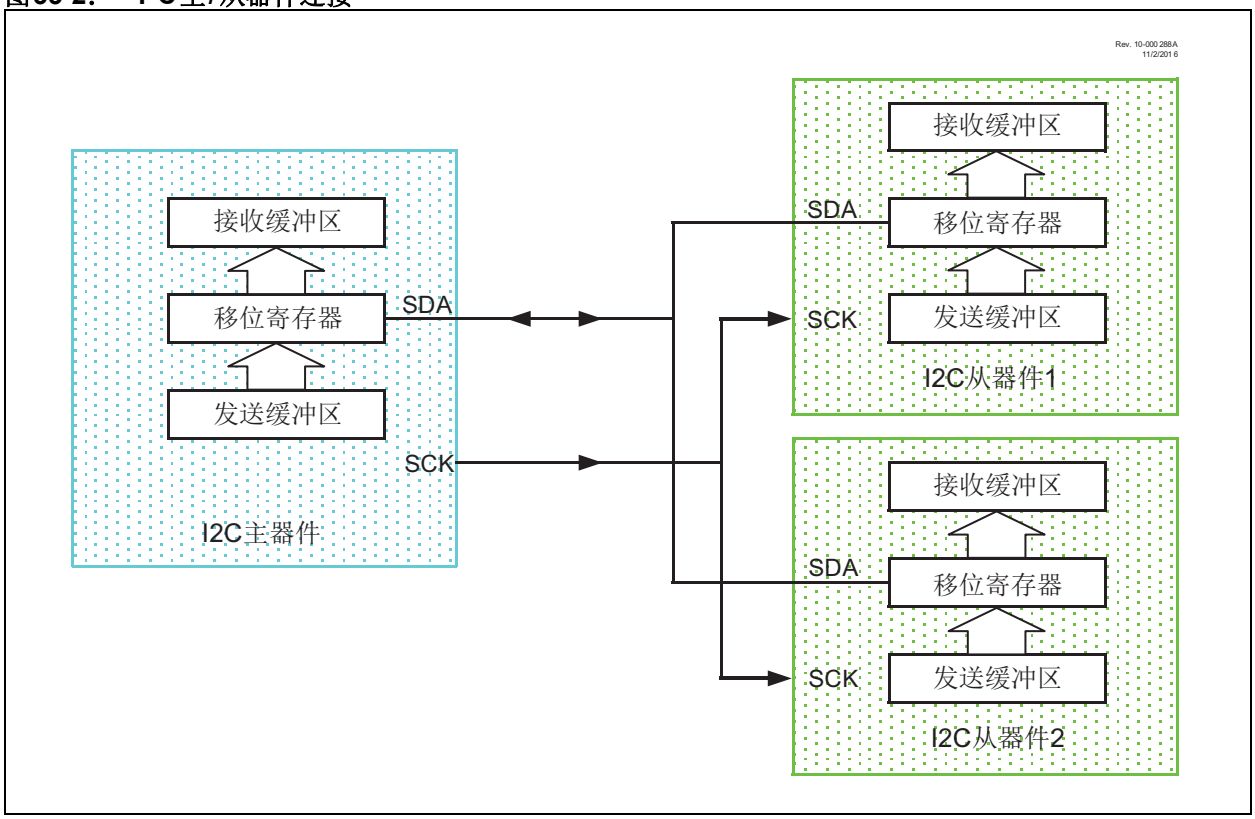
I²C模块使用双线I²C串行总线在单片机和其他I²C兼容器件之间提供同步接口。器件在主/从器件环境中进行通信。I²C总线规定了两种信号连接：

- 串行时钟（SCL）
- 串行数据（SDA）

SCL和SDA连接都是双向的漏极开路线，它们都需要使用连接到供电电压的上拉电阻。线路下拉到地时，信号视为逻辑0；线路保持悬空时，信号视为逻辑1。I²C总线上的每个事务都必须由主器件启动。

图33-2给出了主器件和多个从器件之间的典型连接。

图33-2： I²C主/从器件连接



根据I²C通信期间共用数据的方向，可分为以下四种主要工作模式：

- 主发送模式（主器件向从器件发送数据）
- 主接收模式（主器件从从器件接收数据）
- 从发送模式（从器件向主器件发送数据）
- 从接收模式（从器件从主器件接收数据）

要启动任何I²C通信，主器件需发送出启动位，后面跟随它希望进行通信的从器件的地址字节。后面再跟随单个读/写位，该位决定主器件是向从器件发送数据还是从从器件接收数据。

如果总线上存在所请求的从器件，从器件会使用应答位（也称为ACK）进行响应。然后，主器件会继续将数据移入或移出从器件，直到其通过停止条件终止报文。

以下章节将更加详细地介绍I²C模块。

33.3 I²C工作模式

所有I²C通信都采用8位数据加1位应答的形式，并且先移出MSb。用户可以使用多个控制寄存器和中断标志来控制软件和模块之间的交互。模块通过两个引脚SDA和SCL来与其他外部I²C器件进行通信。

33.3.1 I²C术语的定义

下面的表33-1中定义了I²C通信协议术语以供参考。本文档中使用了这些术语。表33-1根据Phillips I²C规范改写。

表33-1： I²C总线术语

术语	说明
发射器	将数据移出到总线上的器件
接收器	从总线上移入数据的器件
主机	启动数据传输、产生时钟信号和终止数据传输的器件
从机	主器件寻址到的器件
多主器件	有多个器件可以启动数据传输的总线
仲裁	用于确保每次只有一个主器件控制总线的过程。仲裁获胜可以确保报文不会被损坏
同步	用于将总线上两个或更多器件的时钟进行同步的过程。
空闲	没有任何主器件在控制总线，并且SDA和SCL线均为高电平
有效	每当有一个或多个主器件在控制总线时
寻址到的从器件	已接收到匹配地址并且正在由主器件提供时钟的从器件
匹配地址	从器件接收到的地址字节与存储在I2CxADR中的值相匹配
写请求	从器件接收到R/W位清零的匹配地址，并已准备好随时钟移入数据
读请求	主器件发送R/W位置1的地址字节，表示要求从器件随时钟移出数据。从器件在接收到该地址字节后会立即移出所有数据字节，直到发生重复启动或停止条件。
时钟延长	总线上的器件通过将SCL保持为低电平来暂停通信的时间
总线冲突	每当模块进行输出并期望SDA线为高电平，却采样到SDA线为低电平时。
总线超时	每当I2CBTOISM输入跳变为高电平时，I²C模块就会复位且模块变为空闲。

33.3.2 字节格式

I²C中的所有通信都采用9位形式。从主器件向从器件（或者反之）发送一个字节之后，接收器将会送回一个应答位。在SCL线的第8个下降沿之后，器件将数据发送到SDA线，将该引脚控制权交由输入引脚，然后在下一个时钟脉冲时读取应答值。时钟信号由主器件提供。在SCL线为低电平时，数据可以有效地更改，并且在时钟上升沿进行采样。在SCL线为高电平时，SDA线上的电平变化定义总线上的启动和停止条件，以下章节会对此进行详细说明。

33.3.3 SDA和SCL引脚

用户必须将这些引脚配置为漏极开路输出。可以通过清零相应的TRIS位和置1适当的ODCON位来完成此操作。用户还可以使用RxyI2C控制寄存器（寄存器16-9）选择输入阈值、压摆率和内部上拉设置。

33.3.4 SDA保持时间

SDA引脚的保持时间通过I2CxCON2寄存器的SDAHT<1:0>位进行选择。保持时间是SDA在SCL的下降沿之后保持有效的时间。较长的保持时间设置可能有益于具有大电容的总线。

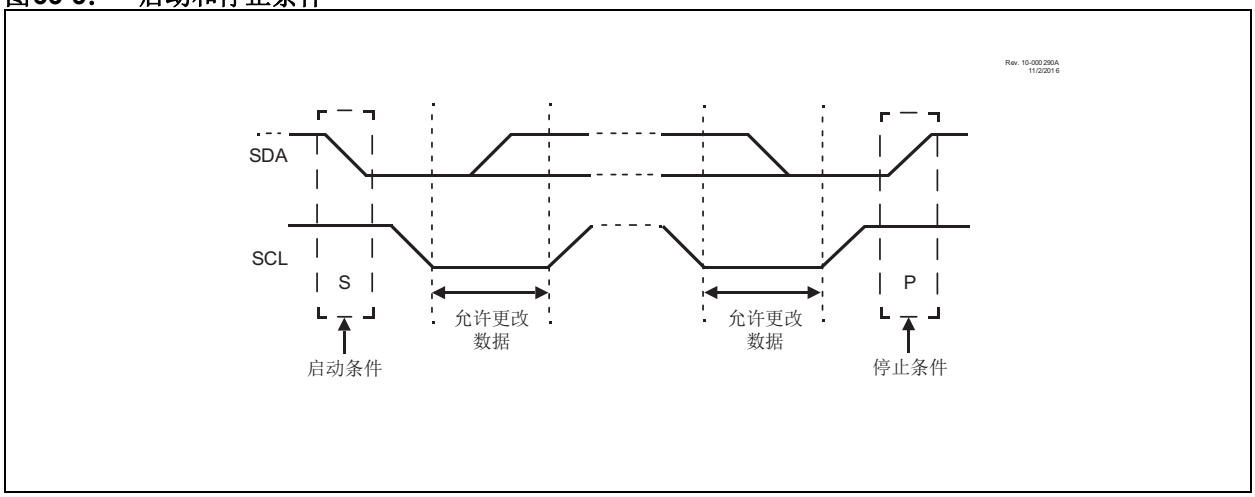
33.3.5 启动条件

I²C规范将启动条件定义为在SCL线为高电平时，SDA线从高电平变为低电平状态。启动条件总是由主器件产生，指示总线从空闲状态变为有效状态。图33-3给出了启动条件的波形图。主器件硬件等待I2CxSTAT0的BFRE位置1，之后在SCL和SDA线上发出启动条件。如果两个主器件同时发出启动条件，则寻址阶段将发生冲突。

33.3.6 停止条件

停止条件定义为在SCL线为高电平时，SDA线从低电平变为高电平状态。图33-3给出了停止条件的波形图。

图33-3： 启动和停止条件



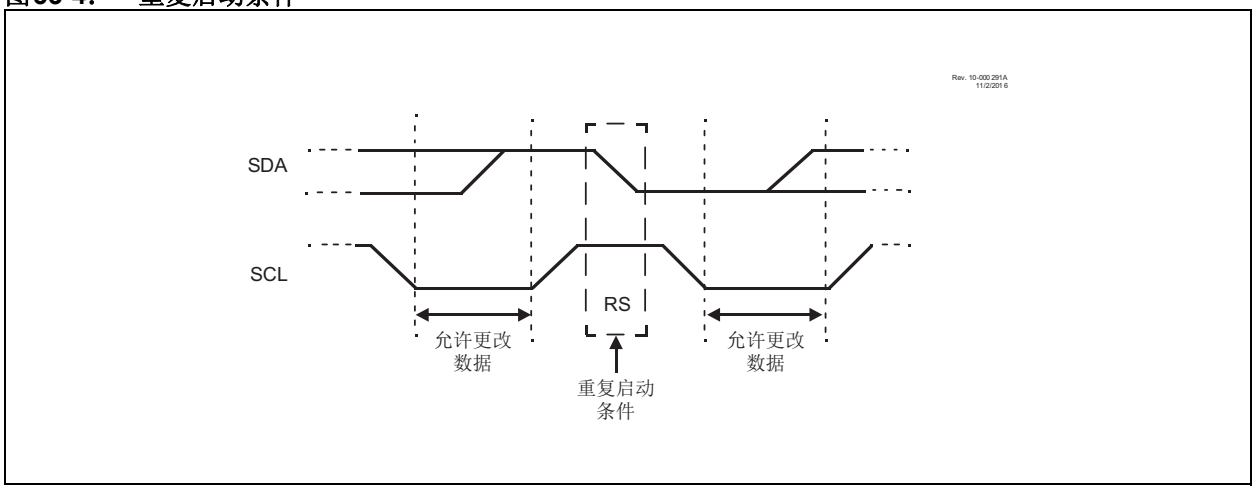
注：	必须至少出现一个SCL低电平信号，停止条件才有效。因此，如果SDA线先变为低电平，然后又变为高电平，而SCL线始终保持高电平，那么只能检测到启动条件。

33.3.7 重复启动条件

重复启动条件在每次停止条件有效的时候有效。如果主器件希望在终止当前传输之后保持对总线的控制，主器件可以发出重复启动条件。重复启动对从器件产生的影响与启动条件相同，即复位所有从器件逻辑并使之准备接收一个地址。主器件可以寻址同一个或另一个从器件。图33-4给出了重复启动条件的波形图。

在10位寻址从模式下，要从寻址到的从器件中移出数据，主器件需要产生重复启动条件。从器件完全寻址（高地址字节和低地址字节均匹配，SMA = 1）之后，主器件可以发出重复启动条件和R/W位置1的高地址字节。然后，从器件逻辑会保持时钟，并准备送出数据。

图33-4： 重复启动条件



33.3.8 应答序列

在I^2C中，所有传输字节的第9个SCL脉冲都专门用作应答信号。它使接收器件可以通过将SDA线下拉为低电平来响应发送器。发送器在该时间内必须释放对线路的控制，以移入响应信号。应答（ACK）是低电平有效信号，它会将SDA线下拉为低电平，用于指示发送器器件已接收到发送数据并已准备好接收更多数据。

ACK的结果会被放入I2CxCON1寄存器的ACKSTAT位中。ACKSTAT位在接收器件发送应答时清零；在接收器件不应答时置1。从器件将在识别出相应地址时发送应答。当处于接收数据的模式时，发送到发送器的ACK数据取决于I2CxCNT寄存器的值。ACKDT是I2CxCNT! = 0时发送的值。当I2CxCNT = 0时，则使用ACKCNT值。

在从模式下，如果ADRIE或WRIE位置1，则在发生地址匹配或尝试写入从器件时启动时钟延长。这允许用户设置发送回发送器的ACK值。用户可以通过置1/清零I2CxCON1寄存器的ACKDT位来决定响应。如果ADRIE或WRIE位清零，则从器件硬件会产生ACK响应。

某些条件将导致自动发送无应答（NACK）响应。如果RXRE、TXRE、RXO或TXU位中的任何一位置1，则硬件响应被强制为NACK。器件用于地址匹配或数据的所有后续响应将为NACK响应。

33.3.9 总线超时

I2CxBTO寄存器可用于选择模块的超时源。当所选总线超时信号变为高电平时，I^2C模块复位。此特性对实现SMBus和PMBus™兼容性十分有用。

例如，可以选择Timer2作为总线超时源，并将其配置为在SCL引脚为低电平时进行计数。如果定时器在SCL引脚转换为高电平之前运行，则定时器输出脉冲将复位模块。

注： 总线超时源应产生上升沿。

如果模块配置为从器件，并且从器件处于工作状态（即SMA位置1）时发生BTO事件，则模块立即复位。SMA和CSTR位也会清零，BTOIF位置1。

如果在模块配置为主器件且处于工作状态（即MMA位置1）时发生BTO事件，则模块立即尝试发出停止条件并将BTOIF位置1。如果某从器件对总线执行时钟延长，则实际生成停止条件的时间可能有所延迟。只有在生成停止条件后，MMA位才会清零。

33.3.10 地址缓冲区

I²C模块有两个地址缓冲区寄存器，即I2CxADB0和I2CxADB1。这些寄存器可用作接收地址缓冲区寄存器或发送地址缓冲区寄存器，具体取决于模式。有关这些寄存器中的数据流方向，请参见表33-2。在从模式下，这些寄存器仅在发生地址匹配时更新。I2CxCON2寄存器中的ADB位用于使能/禁止地址缓冲区功能。禁止时，地址数据来自发送缓冲区并存储在接收缓冲区中。

表33-2： 不同I²C模式下的地址缓冲区方向

模式	MODE<2:0>	I2CxADB0	I2CxADB1
从模式（7位）	000	RX	—
	001	RX	—
从模式（10位）	010	RX	RX
	011	RX	RX
主模式（7位）	100	—	TX
主模式（10位）	101	TX	TX
多主器件模式（7位）	110	RX	TX
	111	RX	TX

33.3.10.1 从模式（7位）

在7位从模式下，I2CxADB0装入接收到的匹配地址和R/W数据。在此模式下，I2CxADB1寄存器将被忽略。

33.3.10.2 从模式（10位）

在10位从模式下，I2CxADB0装入接收到的匹配地址的低8位。I2CxADB1装入高地址字节的完整8位，包括R/W位。

33.3.10.3 主模式（7位）

在此模式下，I2CxADB0寄存器将被忽略。在7位主模式下，I2CxADB1寄存器用于将地址数据字节（包括R/W值）复制到移位寄存器。

33.3.10.4 主模式（10位）

在10位主模式下，I2CxADB0寄存器存储低地址数据字节值，该值将在高地址字节移出后复制到移位寄存器。I2CxADB1寄存器存储将要复制到移位寄存器的高地址字节值。即使I²C规范将高5位定义为常量，也由用户指定全部8位。

33.3.10.5 多主器件模式（仅限7位）

在多主器件模式下，器件可以作为主器件和从器件，具体取决于总线上事件的序列。如果作为从器件寻址，则I2CxADB0寄存器存储接收到的匹配从器件地址字节。如果器件尝试在总线上作为主器件进行通信，则I2CxADB1寄存器的内容将复制到移位寄存器以寻址从器件。

33.3.11 接收和发送缓冲区

接收缓冲区在一个数据字节移入SDA引脚时保存另一个数据字节。用户可借助软件（或DMA）的方式通过I2CxRXB寄存器访问缓冲区。当新数据装入I2CxRXB寄存器时，接收缓冲区满状态位（RXBF）置1，读I2CxRXB寄存器会清零该位。

如果用户试图在I2CxRXB为空（即RXBF = 0）时对其进行读取，则接收读错误位（RXRE）置1并产生NACK。用户必须清零错误位才能恢复正常工作。

发送缓冲区在一个数据字节通过SDA引脚移出时保存另一个数据字节。用户可借助软件（或DMA）的方式通过I2CxTXB寄存器访问缓冲区。当I2CxTXB不包含任何发送数据时，发送缓冲区空状态位(TXBE)置1。此时，用户可以将另一个字节装入缓冲区中。

如果用户试图在I2CxTXB非空（即TXBE = 0）时对其进行写入，则发送写错误标志位（TXRE）置1并丢弃新数据。当TXRE置1时，用户必须清除此错误条件才能恢复正常工作。

用户可通过将I2CxSTAT1寄存器中的CLRBF位置1来清空接收和发送缓冲区。CLRBF也将清零I2CxRXIF和I2CxTXIF位。

33.3.12 时钟延长

在从器件尚未完成数据处理时，它可以通过时钟延长这一过程来延迟更多数据的传输。寻址到的从器件可以在接收或发送一位数据之后将SCL时钟线保持为低电平，指示它尚未准备好继续通信。主器件将会尝试将SCL线拉至高电平，以传输下一位数据，但它会检测到时钟线尚未被释放。由于SCL连接是漏极开路，所以从器件可以将线路保持为低电平，直到它准备好继续通信为止。通过时钟延长，无法与发送器保持同速的接收器可以控制传入数据流。

通过将I2CxCON2寄存器中的CSTRDIS（时钟延长禁止）位清零或置1，可以使能或禁止时钟延长。该位仅在多主器件模式或从模式下有效。

33.3.12.1 缓冲区操作的时钟延长

使能时，在缓冲区读/写操作期间强制进行时钟延长。例如，在从模式下，如果RXBF = 1（接收缓冲区已满），时钟将在SCL的第7个下降沿后延长。只有在用户从接收缓冲区读取数据后才释放SCL线。这可确保始终不会出现接收数据上溢。在这种情况下，如果禁止时钟延长，则I2CxCON1中的RXO位置1，指示接收上溢。置1时，模块将始终以NACK响应。

同样，当TXBE = 1（发送缓冲区为空）且I2CCNT! = 0时，时钟将在SCL的第8个下降沿后延长。只有在用户向发送缓冲区装入新数据后才释放SCL线。这可确保始终不会出现发送下溢。在这种情况下，如果禁止时钟延长，则I2CxCON1中的TXUF位置1，指示发送下溢。置1时，模块将始终以NACK响应。

33.3.12.2 其他从器件操作的时钟延长

共有三个中断和保持位，用于在从模式下提供时钟延长。这些位还可与PIRx寄存器中的I2CxIE位一起使用来生成系统级中断。

- 传入地址匹配中断
  - 通过I2CxPIE寄存器的地址中断保持（ADRIE）位使能传入匹配地址字节后的时钟延长。当ADRIE = 1时，CSTR位置1，SCL线在所接收匹配地址的SCL的第8个下降沿之后延长。这允许用户从I2CADB0/1寄存器读取接收到的地址，并根据接收到的地址选择性地发送ACK/NACK。通过ADRIE实现的时钟延长可通过用软件清零CSTR位的方式来解除。
- 数据写中断
  - 数据写中断和保持使能（WRIE）位用于使能在接收到的数据字节之后进行时钟延长。当WRIE = 1时，CSTR位置1，SCL线在传入从数据的第8个SCL下降沿后延长。该位允许用户软件在接收到的每个数据字节后选择性地发送ACK/NACK。通过WRIE实现的时钟延长可通过用软件清零CSTR位的方式来解除。
- 应答状态
  - 应答状态时间中断和保持使能（ACKTIE）位用于使能在发送过程的ACK阶段之后进行时钟延长。此位可为所有地址/数据事务（寻址、写入或读取）使能时钟延长。在ACK之后，从器件硬件会将CSTR置1。通过ACKTIE实现的时钟延长可通过用软件清零CSTR位的方式来解除。

33.3.13 数据字节计数

I2CxCNT寄存器用于指定完整I^2C数据包中的字节数。每当通过I^2C模块接收或发送数据字节时，该寄存器中的值就会递减。I2CxCNT寄存器不会递减至零以下。

如果字节传输导致I2CxCNT寄存器递减至零，则在I2CxPIR中的计数中断标志位（CNTIF）将置1。该标志位在发送操作的SCL的第9个下降沿置1，如果指定了边沿，则可以提供更多信息。

如果I2CxCON2寄存器中的ACNT位置1，则可自动装入I2CxCNT寄存器。当ACNT位置1时，地址字节后的数据字节将装入I2CxCNT寄存器。

注 1： I2CxCNT在SCL的第8个（接收）或第9个（发送）下降沿递减；在此位时间内写入会破坏值。

2： 如果报文的块大小大于255，则I2CxCNT寄存器可以在报文中间更新，以防止递减至零。

33.4 I^2C从模式

通过I2CxCON0中的模式位，可在4种I^2C从模式中选择一种作为工作模式。这些模式可以分为7位和10位寻址模式。10位寻址模式的工作方式与7位寻址模式相同，只是在处理较大地址时需要一些额外的开销。

33.4.1 从寻址模式

I2CxADR/1/2/3寄存器包含从模式地址。在启动或重复启动条件之后接收到的第一个字节将与这些寄存器中的存储值进行比较。如果字节与值匹配，则将其装入I2CxADB0/1寄存器中。如果值不匹配，则模块不会发出任何响应。I^2C模块可采用以下从配置。

33.4.1.1 7位寻址模式

在该模式下，在确定地址是否匹配时，所接收数据字节的LSb会被忽略。全部四个I2CxADR寄存器均单独与接收的地址字节进行比较。

33.4.1.2 带掩码的7位寻址模式

在该模式下，I2CxADR0中的值被I2CxADR1中的值掩码，以确定是否发生了地址匹配。还会比较I2CxADR2/3中的第二组地址和掩码。当Mode<2:0> = 001或111时，I2CxADR1/3寄存器的内容用作I2CxADR0/2的掩码值。可以对全部7位地址进行掩码

33.4.1.3 10位寻址模式

在该模式下，I2CxADR0和I2CxADR1寄存器中的存储值用于创建10位地址。第二个10位比较地址由I2CxADR2和I2CxADR3形成。

33.4.1.4 带掩码的10位寻址模式

在该模式下，I2CxADR0/1寄存器用于形成10位地址，I2CxADR2/3寄存器用于形成该地址的10位掩码。当MODE<2:0> = 011时，I2CxADR2/3寄存器的内容用作I2CxADR0/1中存储的10位地址的掩码值。

注： 尽管10位寻址仅调出地址比较中使用的10位，但在这些模式下，I2CxADR0/1中的全部15个地址位都会进行比较。

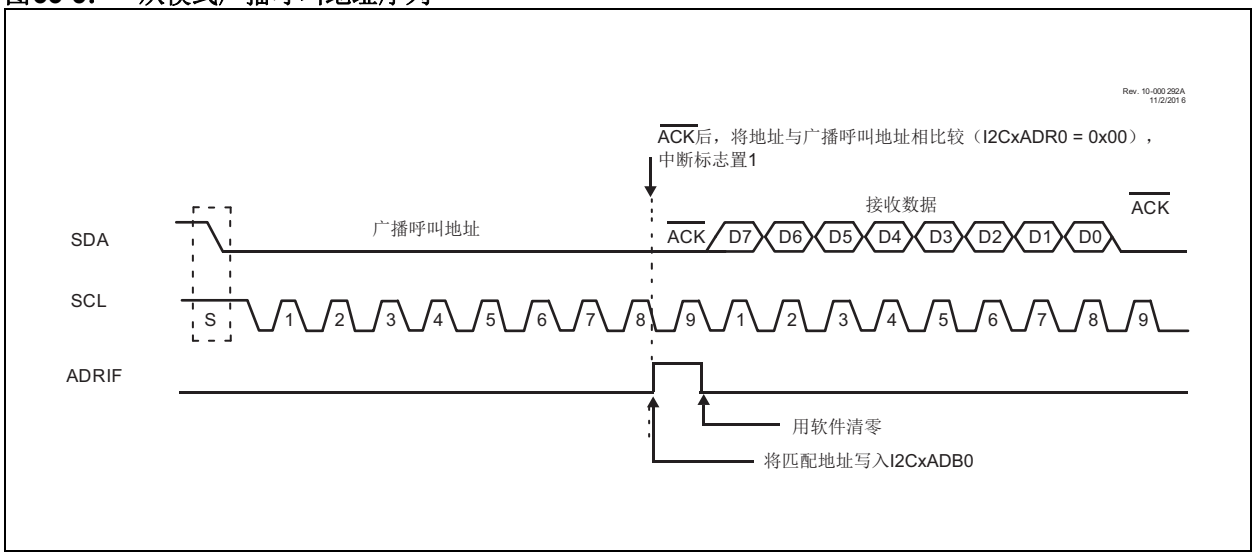
33.4.2 广播呼叫地址支持

在I^2C总线的寻址过程中，通常由启动条件后的第一个字节决定主器件将寻址哪个从器件。但广播呼叫地址例外，它能寻址所有器件。当使用这个地址时，理论上所有器件都应该发送一个ACK信号来响应。广播呼叫地址是I^2C协议中的保留地址，定义为地址0x00。为了使从器件硬件能够应答该地址，必须通过将I2CxCON2寄存器中的GCEN位置1来使能。不需要将I2CxADR0/1/2/3寄存器之一设置为0x00。图33-5显示了广播呼叫接收序列。

如果ADRIE位置1，则模块将在第8个SCL脉冲之后进行时钟延长，方式与任何其他地址匹配一样。

注： 仅7位寻址模式支持广播呼叫寻址

图33-5： 从模式广播呼叫地址序列



33.4.3 7位寻址模式下的从操作

主器件发送的地址字节中的第8位用于确定主器件要读取还是写入从器件。置1时，表示主器件要读取从器件，清零时，表示主器件要写入从器件。如果发生地址匹配，则将R/W位复制到I2CxSTAT0寄存器的R/W位。

33.4.3.1 从接收（7位寻址模式）

本节介绍在7位寻址模式下，配置为I^2C从器件且正在接收数据的I^2C模块的事件序列。图33-6、图33-7和图33-8用直观的方式对此作了说明。

1. 主器件在总线上发出启动条件（也可以是重复启动条件）。I2CxPIR寄存器中的启动条件中断标志（SCIF）置1。
2. 如果允许了启动条件中断（SCIE位置1），则通用中断I2CxIF置1。
3. 主器件发送8个位，7位地址和R/W（R/W = 0）。
4. 接收的地址与I2CxADR0/I2CxADR1/I2CxADR2/I2CxADR3寄存器中的值进行比较。有关从寻址模式的信息，请参见**第33.4.1节“从寻址模式”**。
5. 如果地址匹配；I2CxSTAT0寄存器中的SMA置1，R/W复位到R/W位，D/A位清零。如果地址不匹配；模块变为空闲状态。
6. 匹配的地址数据装入I2CxADB0，I2CxPIR寄存器中的ADRIF置1。
7. 如果允许了地址保持中断（ADRIE = 1），则CSTR置1。I2CxIF置1。从软件可在释放SCL之前从I2CxADB0读取地址以及置1/清零ACKDT。
8. 如果存在任何先前的错误条件（例如，接收缓冲区上溢或发送缓冲区下溢错误），则从器件将强制NACK，模块变为空闲状态。
9. ACKDT值复制到SDA，以便主器件在第9个SCL脉冲读取ACK脉冲。
10. 如果使能了应答中断和保持（ACKTIE = 1），CSTR置1，I2CxIF置1，则从软件可在通过清零CSTR释放SCL之前从I2CxADB0寄存器读取地址以及更改ACKDT的值。
11. 主器件发送数据字节的前7个SCL脉冲或停止条件（在NACK的情况下）。
12. 如果出现停止条件，I2CxPIR寄存器中的PCIF置1，模块变为空闲状态。
13. 如果接收缓冲区在前一个事务后已满（即RXBF = 1（I2CxRXIF = 1）），则CSTR置1。从软件必须从I2CxRXB读取数据才能恢复通信。
14. 主器件发送数据字节的第8个SCL脉冲。D/A位置1，WRIF置1。
15. I2CxRXB装入新数据，RXBF位置1，I2CxRXIF置1。
16. 如果使能了数据写入中断和保持（WRIE = 1），则CSTR置1，I2CxIF置1。从软件可在通过清零CSTR释放SCL之前从I2CxRXB读取数据以及置1/清零ACKDT。
17. 如果I2CxCNT = 0，则ACKCNT值输出到SDA；否则，如果I2CxCNT!= 0，则使用ACKDT值且I2CxCNT的值递减。
18. ACK值复制到SDA，以便主器件在第9个SCL脉冲进行读取。
19. 如果I2CxCNT = 0，CNTIF置1。
20. 如果发送了NACK，则NACKIF置1，模块变为空闲状态。
21. 如果ACKTIE = 1，则CSTR置1，I2CxIF置1。从软件可在通过清零CSTR释放SCL之前从I2CxRXB读取数据，清零RXBF。
22. 转到步骤11。

图33-6： I²C从模式接收时序（ACKTIE = 0，ADRIE = 0，WRIE = 0，7位地址）

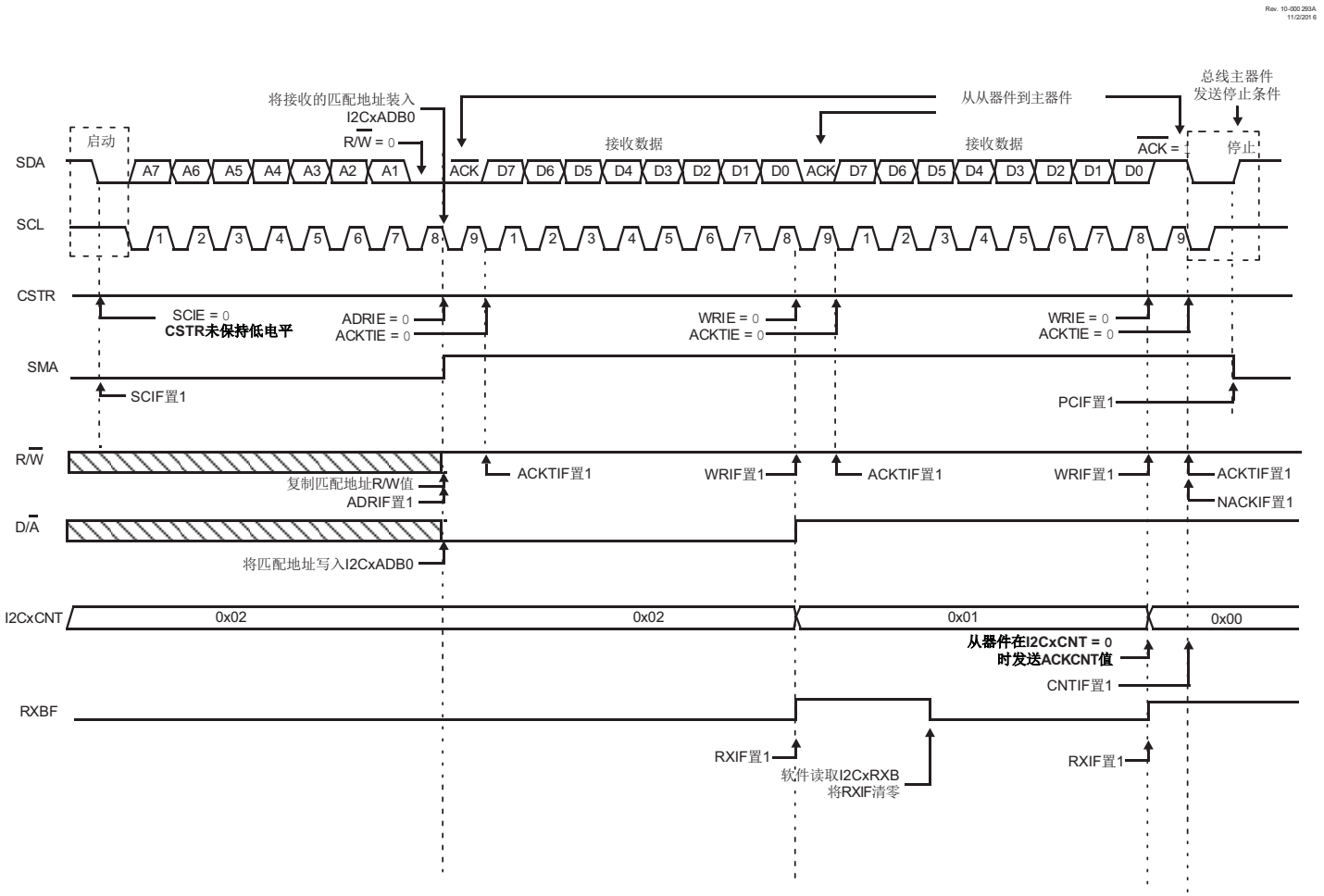


图33-7： I²C从模式接收时序（ACKTIE = 1，ADRIE = 0，WRIE = 0，7位地址，使用I2CxCNT）

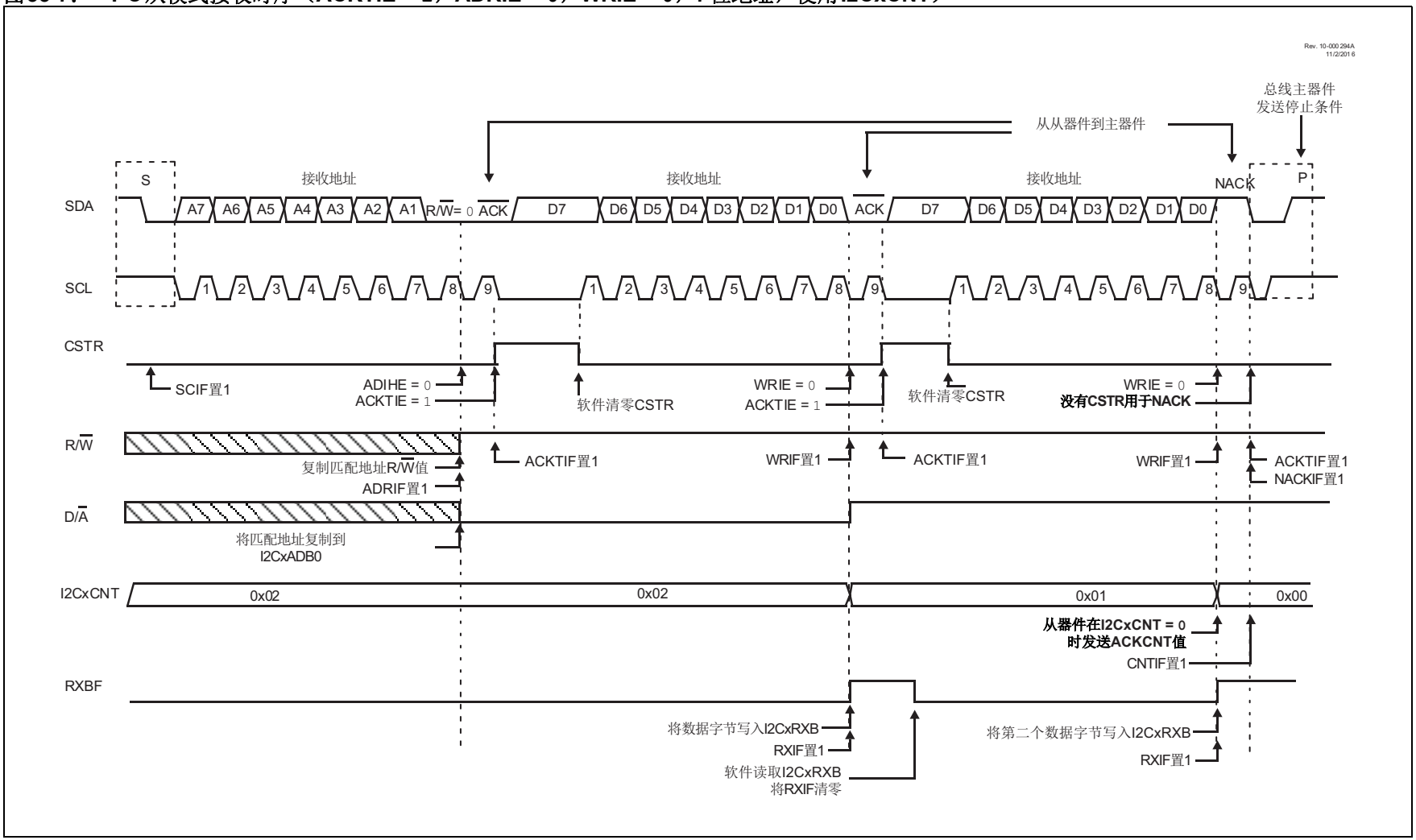
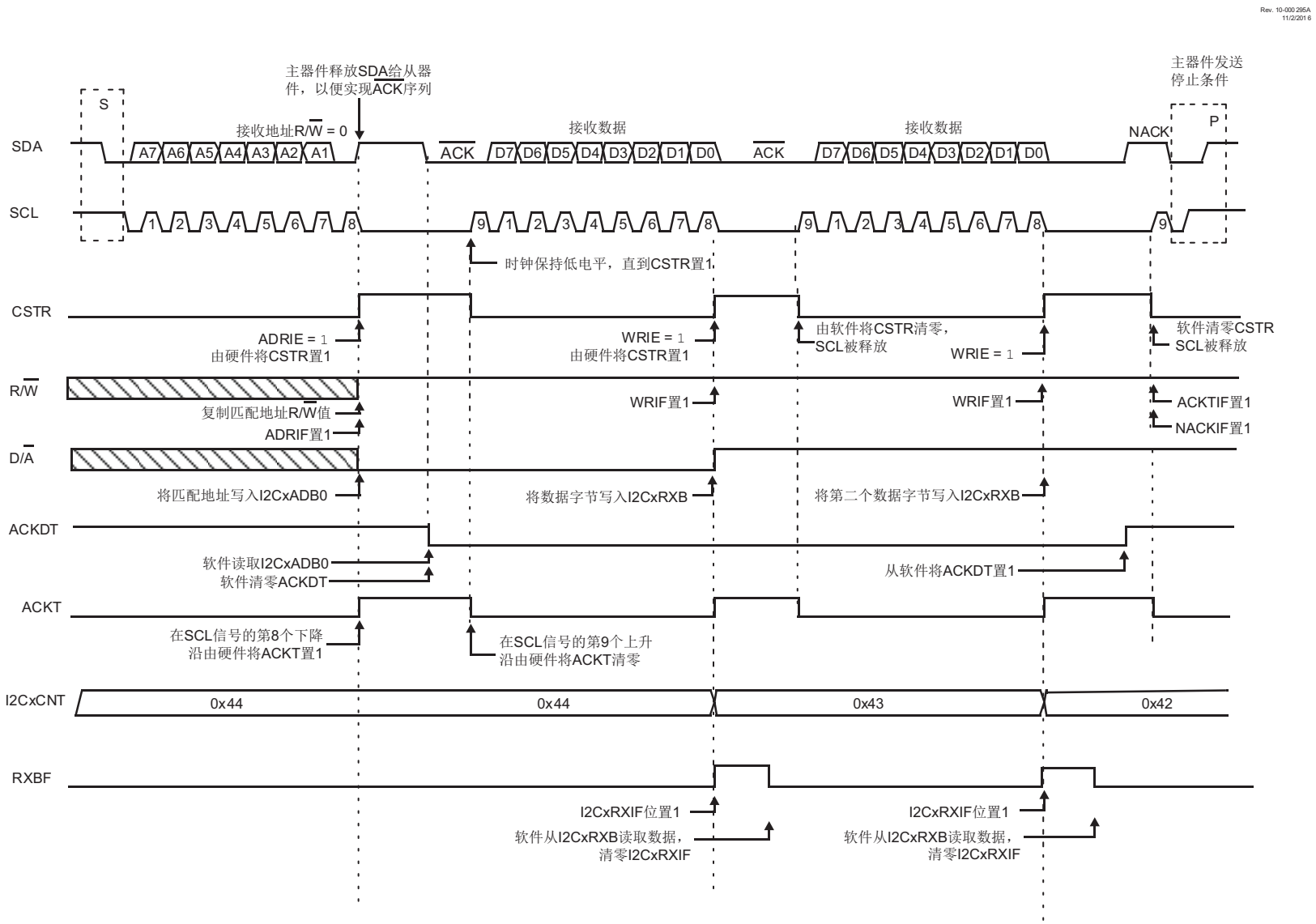


图33-8： I²C从模式接收时序（ACKTIE = 0，ADRIE = 1，WRIE = 1，7位地址，不使用I2CxCNT）



33.4.3.2 从发送（7位寻址模式）

本节介绍在7位寻址模式下，配置为I²C从器件且正在发送数据的I²C模块的事件序列。图33-9和图33-10用直观的方式对此作了说明。

1. 主器件在总线上发出启动条件（也可以是重复启动条件）。I2CxPIR寄存器中的启动条件中断标志（SCIF）置1。
2. 如果允许了启动条件中断（SCIE位置1），则通用中断I2CxIF置1。
3. 主器件发送8个位，7位地址和R/W（R/W = 1）。
4. 接收的地址与I2CxADR0/I2CxADR1/I2CxADR2/I2CxADR3寄存器中的值进行比较。有关从寻址模式的信息，请参见**第33.4.1节“从寻址模式”**。
5. 如果地址匹配；I2CxSTAT0寄存器中的SMA置1，R/W复位到R/W位，D/A位清零。如果地址不匹配；模块变为空闲状态。
6. 匹配的地址数据装入 I2CxADB0，I2CxPIR 寄存器中的ADRIF置1。
7. 如果允许了地址保持中断（ADRIE = 1），则CSTR置1。I2CxIF置1。从软件可在释放SCL之前从I2CxADB0读取地址以及置1/清零ACKDT。可通过清零CSTR来释放SCL线。
8. 如果发送缓冲区在前一个事务后为空，即TXBE = 1且I2CxCNT! = 0（I2CxTXIF = 1），则CSTR置1。从软件必须将数据装入I2CxTXB才能释放 SCL。将字节装入移位寄存器后，I2CxCNT递减。
9. 从器件硬件等待主器件在第 9 个 SCL 脉冲发出ACK数据。
10. 如果I2CxCNT = 0，CNTIF置1。
11. 如果使能了应答中断和保持（ACKTIE = 1），则CSTR置1，I2CxIF置1。
12. 从软件可在通过清零 CSTR 释放 SCL 之前更改ACKDT的值。
13. 主器件可通过发送8个SCL脉冲来随时钟输出数据，也可通过发出停止条件来结束事务。
14. 转到步骤8。

图33-9： I²C从模式发送时序（7位地址）

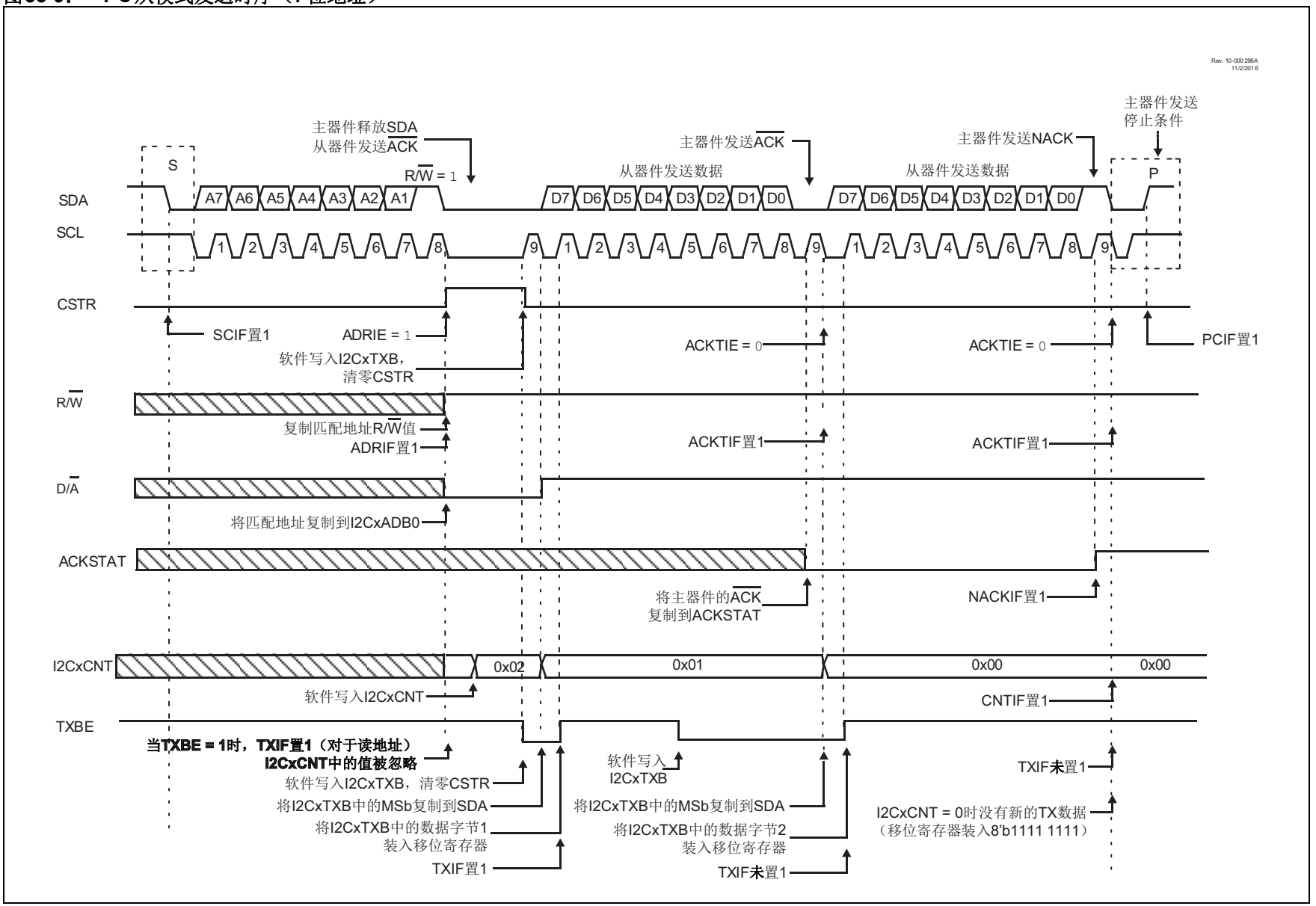
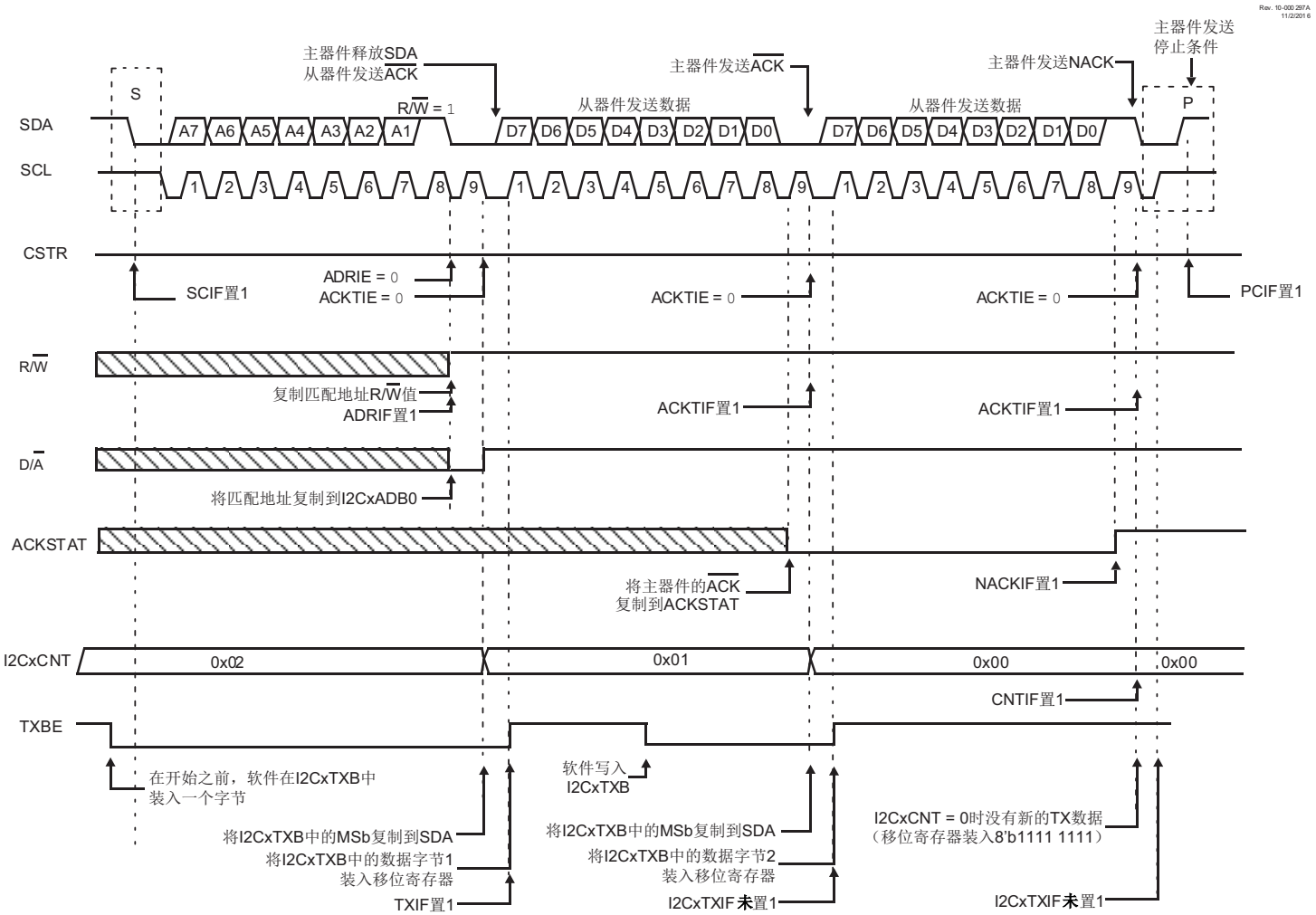


图33-10： I²C从模式发送时序（7位地址，无时钟延长）



33.4.3.3 10位寻址模式下的从操作

在10位寻址模式下，接收到的第一个字节将与二进制值11110A9A80进行比较。A9和A8是10位地址的高2位。第一个字节将与I2CxADR1和I2CxADR3寄存器中的值进行比较。在应答高字节后，低地址字节随时钟移入，且全部8位将与I2CxADR0和I2CxADR2寄存器中的低地址值进行比较。在所有10位寻址通信开始时，都需要以写请求方式进行高地址和低地址匹配。要启动读操作，主器件需要在寻址到从器件后发出重复启动条件，并随着时钟移入R/W位置1的高地址。然后，从器件硬件将会应答读请求，并准备好随着时钟移出数据。仅当高地址字节和低地址字节均匹配时，SMA（从器件工作）位才会置1。

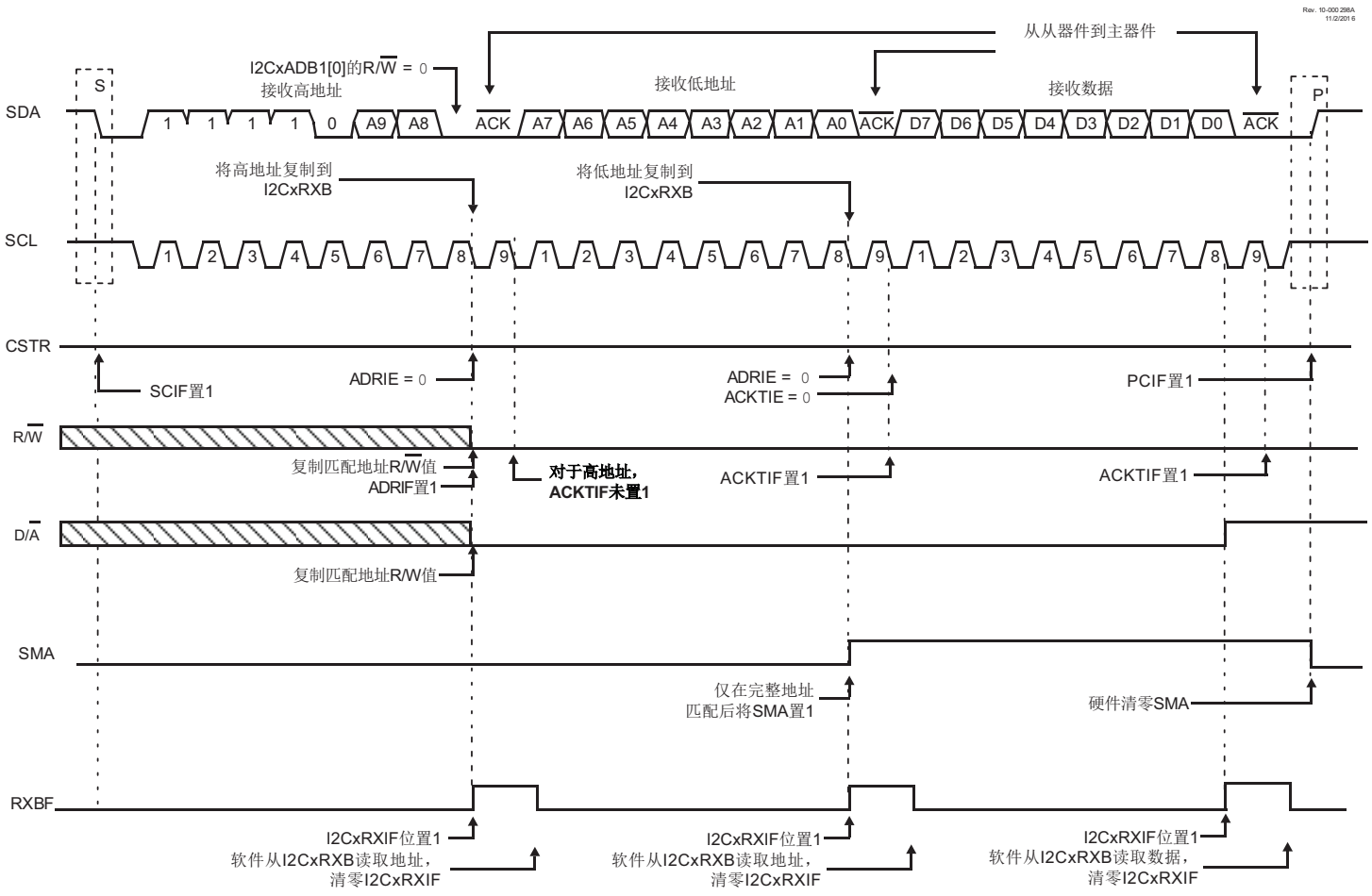
注：	接收到的高地址的全部7位都将与I2CxADR1和I2CxADR3寄存器中的值进行比较。模块硬件不强制使用“11110”5 位高地址格式。用户负责正确配置这些位。

33.4.3.4 从接收（10位寻址模式）

本节介绍在10位寻址模式下，配置为I^2C从器件且正在接收数据的I^2C模块的事件序列。图33-11用直观的方式对此作了说明。

1. 主器件在总线上发出启动条件（也可以是重复启动条件）。I2CxPIR寄存器中的启动条件中断标志（SCIF）置1。如果允许了启动条件中断（SCIE位置1），则通用中断I2CxIF置1。
2. 主器件发送R/W = 0的高地址字节。
3. 接收到的高地址将与I2CxADR1和I2CxADR3寄存器中的值进行比较。
4. 如果高地址匹配；R/W复制到R/W位，D/A位清零，高地址数据复制到I2CxADB1。如果地址不匹配；模块变为空闲状态。
5. 如果允许了地址保持中断（ADRIE = 1），则CSTR置1。I2CxIF置1。
6. 从软件可在释放SCL之前从I2CxADB1读取高地址以及置1/清零ACKDT。
7. ACKDT值复制到SDA以获得ACK脉冲。可通过清零CSTR来释放SCL线。
8. 主器件发送第9个SCL脉冲以获得ACK信号。
9. 由于先前的错误（例如，接收缓冲区上溢错误或发送缓冲区下溢错误）未清除，从器件此时可强制发出NACK信号。在这些情况下，从器件硬件强制发出NACK信号，模块变为空闲状态。
10. 主器件发送低地址数据字节。
11. 如果低地址匹配；则SMA置1，ADRIF置1，R/W复制到R/W位，D/A位清零，低地址数据复制到I2CxADB0，ACTDT 复制到SDA。如果地址不匹配；模块变为空闲状态。
12. 如果允许了地址保持中断，则CSTR位将按步骤6所述置1。从软件可在释放SCL之前从I2CxADB0寄存器读取低地址字节以及更改ACKDT值。
13. 主器件发送第9个SCL脉冲以便获得ACK信号。
14. 如果使能了应答中断和保持（ACKTIE = 1），则CSTR置1，I2CxIF置1。
15. 从软件可在通过清零 CSTR 释放 SCL 之前从I2CxADB0和I2CxADB1寄存器读取地址以及更改ACKDT的值。
16. 主器件发送数据字节的前7个SCL脉冲或停止条件（在NACK的情况下）。
17. 如果出现停止条件，I2CxPIR寄存器中的PCIF置1，模块变为空闲状态。
18. 如果接收缓冲区在前一个事务后已满，即RXBF = 1（I2CxRXIF = 1），则CSTR置1。从软件必须从I2CxRXB读取数据才能恢复通信。
19. 主器件发送数据字节的第8个SCL脉冲。D/A位置1，WRIF置1。I2CxRXB装入新数据，RXBF位置1。
20. 如果使能了数据写入中断和保持（WRIE = 1），则CSTR置1，I2CxIF置1。从软件可在通过清零CSTR释放SCL之前从I2CxRXB读取数据以及置1/清零ACKDT。
21. 如果I2CxCNT = 0，则ACKCNT值输出到SDA；否则使用ACKDT值且I2CxCNT的值递减。
22. 主器件发送SCL脉冲以便获得ACK信号。
23. 如果I2CxCNT = 0，CNTIF置1。
24. 如果响应为NACK，则NACKIF置1，模块变为空闲状态。
25. 如果ACKTIE = 1，则CSTR置1，I2CxIF置1。从软件可在通过清零 CSTR 释放 SCL 之前从I2CxRXB读取数据，清零RXBF。
26. 转到步骤16。

图33-11： I²C从模式接收时序（ADB = 1，10位地址，使用停止条件）

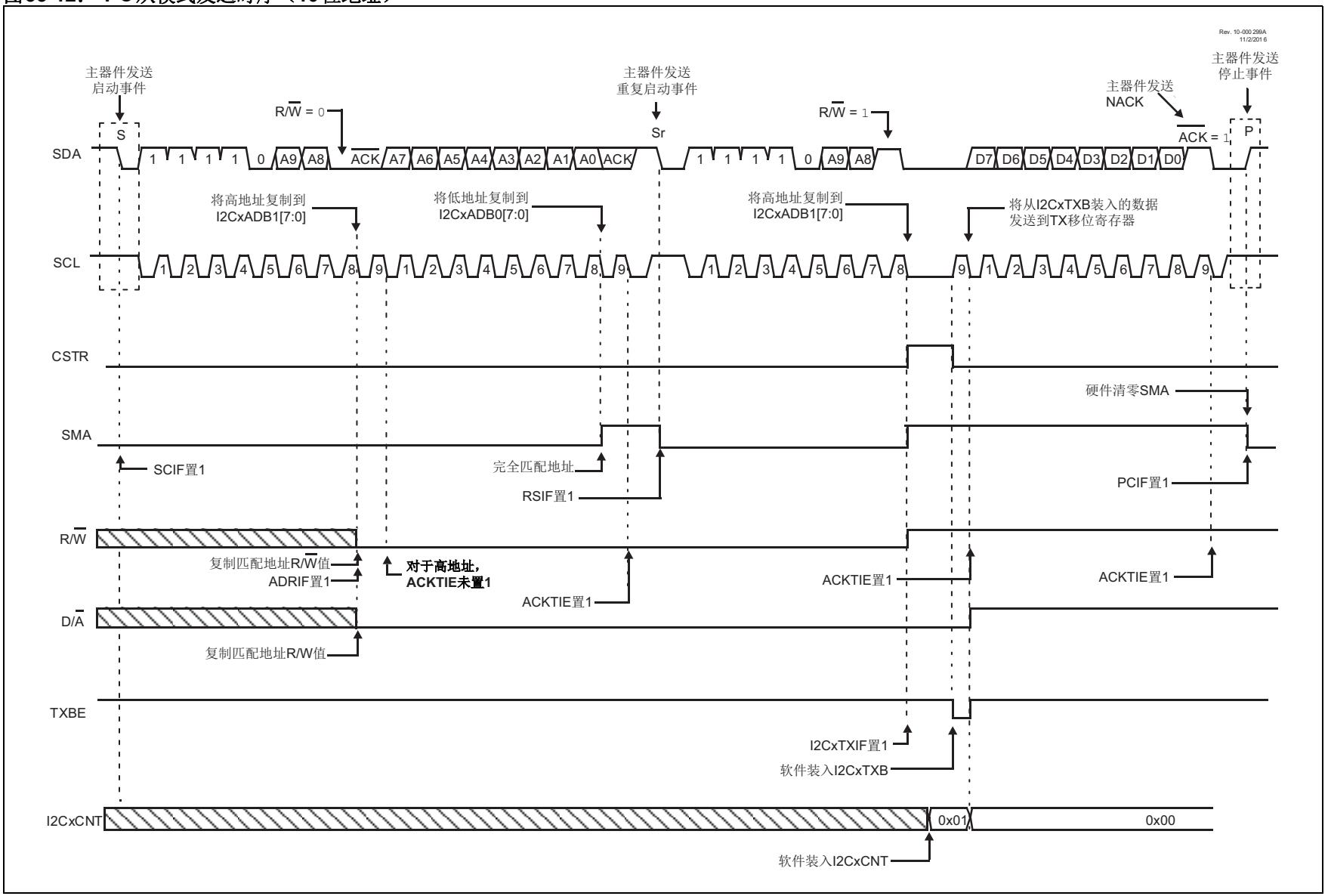


33.4.3.5 从发送（10位寻址模式）

本节介绍在10位寻址模式下，配置为I^2C从器件且正在发送数据的I^2C模块的事件序列。图33-12用直观的方式对此作了说明。

1. 主器件在总线上发出启动条件（也可以是重复启动条件）。I2CxPIR寄存器中的启动条件中断标志（SCIF）置1。如果允许了启动条件中断（SCIE位置1），则通用中断I2CxIF置1。
2. 主器件发送R/W = 0的高地址字节。
3. 接收到的高地址将与I2CxADR1和I2CxADR3寄存器中的值进行比较。
4. 如果高地址匹配；R/W复制到R/W位，D/A位清零，高地址数据复制到I2CxADB1。如果地址不匹配；模块变为空闲状态。
5. 如果允许了地址保持中断（ADRIE = 1），则CSTR置1。I2CxIF置1。
6. 从软件可在释放SCL之前从I2CxADB1读取高地址以及置1/清零ACKDT。
7. ACKDT值复制到SDA以获得ACK脉冲。可通过清零CSTR来释放SCL线。
8. 主器件发送第9个SCL脉冲以便获得ACK信号。
9. 由于先前的错误（例如，接收缓冲区上溢错误或发送缓冲区下溢错误）未清除，从器件此时可强制发出NACK信号。在这些情况下，从器件硬件强制发出NACK信号，模块变为空闲状态。
10. 主器件发送低地址数据字节。
11. 如果低地址匹配；则SMA置1，ADRIF置1，R/W复制到R/W位，D/A位清零，低地址数据复制到I2CxADB0，ACTDT复制到SDA。如果地址不匹配；模块变为空闲状态。
12. 如果允许了地址保持中断，则CSTR位将按步骤6所述置1。从软件可在释放SCL之前从I2CxADB0寄存器读取低地址字节以及更改ACKDT值。
13. 主器件发送第9个SCL脉冲以便获得ACK信号。
14. 如果使能了应答中断和保持（ACKTIE = 1），则CSTR置1，I2CxIF置1。
15. 从软件可在通过清零CSTR释放SCL之前从I2CxADB0和I2CxADB1寄存器读取地址以及更改ACKDT的值。
16. 主器件在总线上发出重复启动条件（不可以是启动条件）。重复启动条件中断标志（RSCIF）置1。如果允许了重复启动条件中断，则通用中断I2CxIF置1。
17. 主器件发送R/W = 1的高地址字节。
18. 如果SMA = 1且高地址匹配，R/W复制到R/W位，D/A位清零，高地址数据复制到I2CxADB1且ACTDT输出到SDA。如果地址不匹配或SMA = 0，模块变为空闲状态。
19. 如果ADRIE = 1，CSTR置1。I2CIF置1。从软件可从I2CxADB0/1读取地址以及置1/清零ACKDT。ACKDT值复制到SDA。可通过清零CSTR位来释放SCL。
20. 如果TXBE = 1且I2CCNT! = 0（I2CTXIF = 1），则CSTR置1。从软件必须将数据装入I2CxTXB才能释放SCL。
21. 主器件发送SCL脉冲以便获得ACK信号。如果I2CCNT = 0，CNTIF置1。
22. 如果发送NACK；NACKIF置1，从器件变为空闲状态。
23. 如果ACKTIE = 1，则CSTR置1，I2CIF置1。从软件可在通过清零CSTR释放SCL之前从I2CxADB0/1读取地址。
24. 主器件通过发送8个SCL脉冲来随时钟移出数据。
25. 转到步骤20。

图33-12： I²C从模式发送时序（10位地址）



33.5 I²C 主模式

通过将I2CxCON中的相应Mode<2:0>位置1和清零，同时将I2CEN位置1，可以使能主模式。通过在缓冲区已满（RXIF）、缓冲区为空（TXIF）以及检测到启动、重复启动和停止条件时产生中断来支持主操作模式。停止（P）、重复启动（RS）和启动（S）位在复位时或禁止I²C 模块时被清零。当I2CSTAT0 的BFRE 位置1时，将控制I²C总线。

33.5.1 I²C 主模式操作

主器件产生所有的串行时钟脉冲、启动条件、重复启动条件和停止条件。以停止条件或重复启动条件结束传输过程。因为重复启动条件也是下一次串行传输的开始，因此I²C总线不会被释放，MMA位将保持置1，表示主模块仍工作。

启动事务的步骤取决于I2CxCON2寄存器的地址缓冲区禁止位（ABD）的设置。

- ABD = 0（使能地址缓冲区）

在这种情况下，主器件模块将使用存储在地址缓冲区寄存器（I2CxADB0/1）中的地址来启动与从器件的通信。用户软件需要将I2CxCON0寄存器中的启动位（S）置1才能启动通信。这对于7位寻址模式和10位寻址模式均有效。

- ABD = 1（禁止地址缓冲区）

在这种情况下，从地址通过发送缓冲区发送，地址缓冲区的内容将被忽略。用户软件需要将从地址写入发送缓冲区（I2CxTXB）才能启动通信。在此模式下，对启动位执行的写操作将被忽略。这对于7位寻址模式和10位寻址模式均有效。

33.5.1.1 主发送器

在主发送器模式下，发送的第一个字节包括接收器件的从地址（7位）和读/写（R/W）位。在主发送器的情况下，R/W位将为逻辑0。一次发送8位串行数据。每发送一个字节，都会接收到一个应答位。输出启动和停止条件指示串行传输的开始和结束。

33.5.1.2 主接收器

在主接收模式下，发送的第一个字节包括发送器件的从地址（7位）和R/W位。在这种情况下，R/W位将为逻辑1。因此，发送的第一个字节是一个7位从地址，后面跟随一个1来指示接收位。串行数据通过SDA接收，而串行时钟由SCL输出。一次接收8位串行数据。每接收到一个字节，都会发送一个应答位。启动条件和停止条件指示发送的开始和结束。

33.5.2 主器件时钟源与仲裁

I²C模块时钟源通过I2CxCLK寄存器选择。I²C时钟为主模式提供SCL输出时钟，供总线空闲定时器使用。I²C时钟可以来自多个外设。

33.5.3 总线空闲时间

在主模式下，I2CxSTAT0寄存器的BFRE位用于指示总线空闲状态。在该位由硬件置1后，主器件硬件才能发出启动条件。这可以防止主器件与可能已经在总线上通信的其他主器件发生冲突。使用I2CxCON1的BFRET<1:0>位，可在将BFRE位置1之前选择8至64个I²C时钟输入脉冲。BFRET位用于确保I²C模块始终遵循最小停止保持时间。电气规范一章中列出了I²C时序要求。

注： 不要求I²C时钟具有50%占空比。

33.5.4 主器件时钟时序

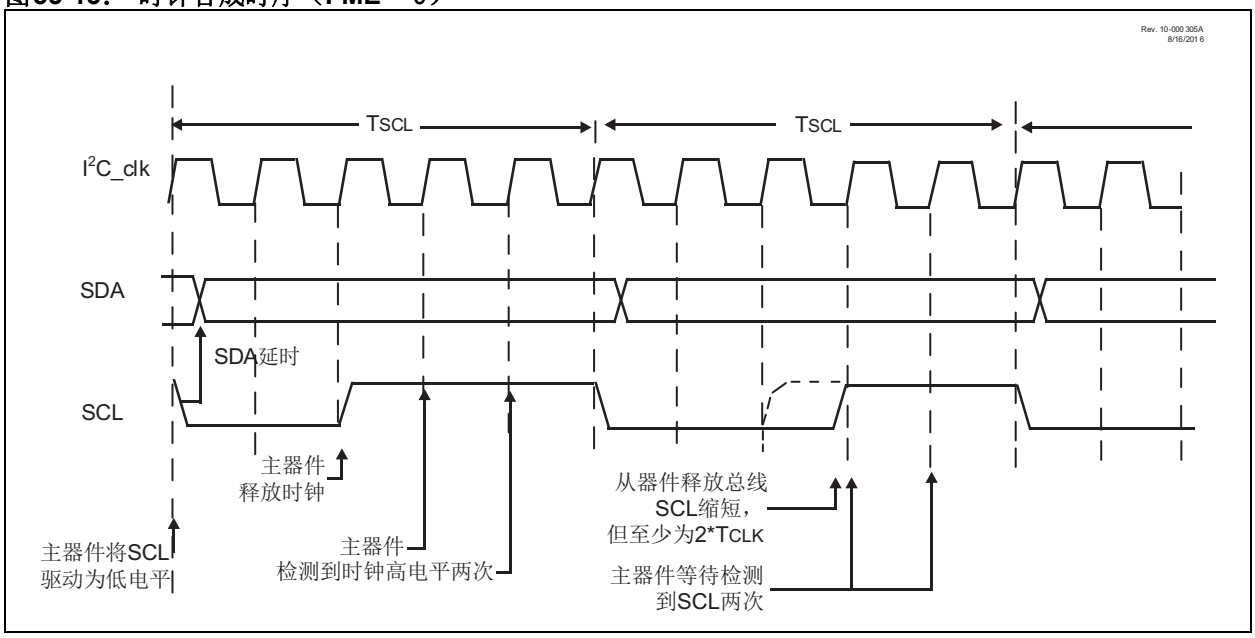
I²C模块中的时钟生成可使用I2CxCON2寄存器的快速模式使能（Fast Mode Enable，FME）位配置。该位控制SCL引脚在受主器件硬件驱动之前被采样的次数。

33.5.4.1 FME = 0时的时钟时序

一个Tscl由I²C时钟输入的五个时钟组成。第一个时钟用于将SCL驱动为低电平，第三个时钟用于将SCL释放为高电平。第四个时钟和第五个时钟用于检测SCL引脚实际上是高电平还是由从器件延长。

如果从器件进行时钟延长，则硬件等待；在每个连续的I²C时钟检查SCL，仅在检测到SCL为高电平后才继续操作。图33-13给出了FME = 0时的时钟合成时序。

图33-13： 时钟合成时序（FME = 0）

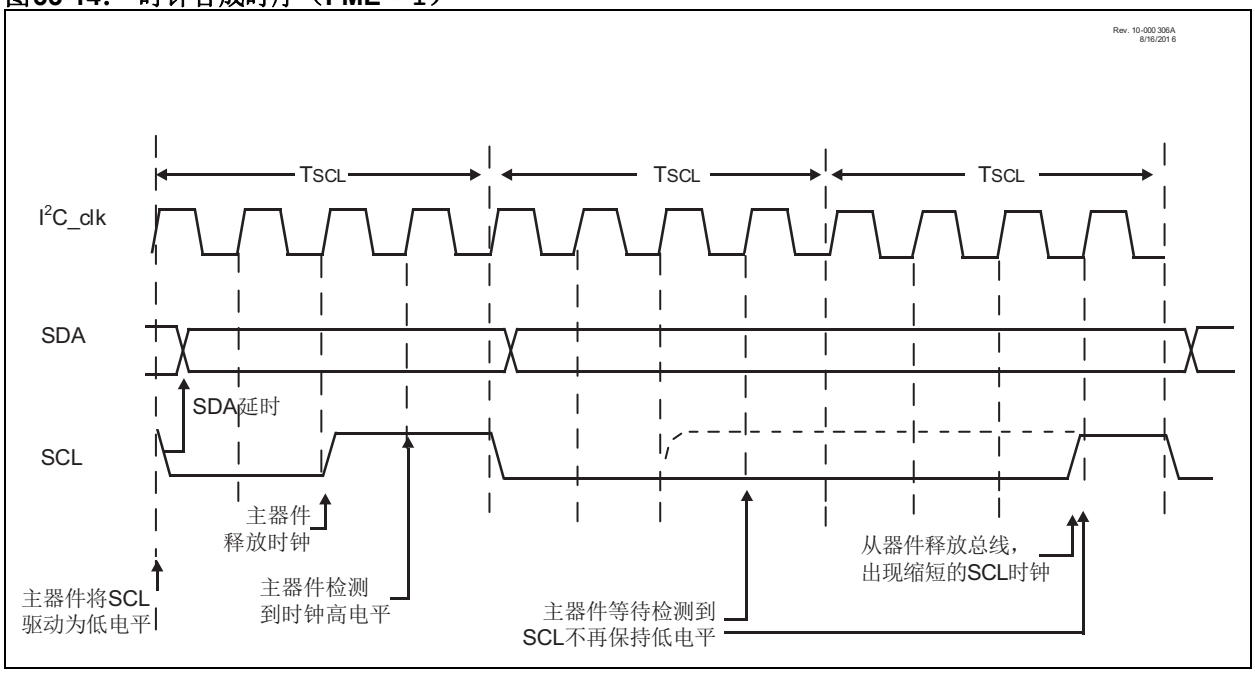


33.5.4.2 FME = 1时的时钟时序

一个TSCL由I²C时钟输入的四个时钟组成。第一个时钟用于将SCL驱动为低电平，第三个时钟用于将SCL释放为高电平，第四个时钟用于检测时钟实际上为高电平还是由从器件延长。

如果从器件进行时钟延长，则硬件等待；在每个连续的I²C时钟检查SCL，仅在检测到SCL为高电平后才继续操作。图33-14给出了FME = 1时的时钟合成时序。

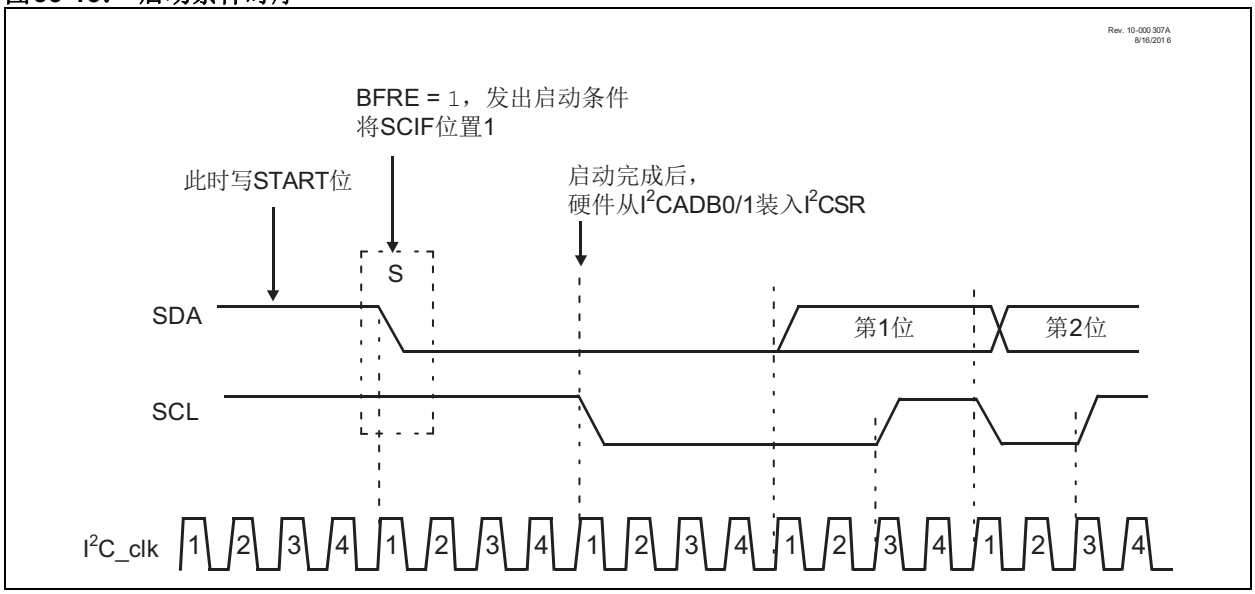
图33-14： 时钟合成时序（FME = 1）



33.5.5 I²C主模式启动条件时序

用户可通过以下方式来发出启动条件：写入I2CxCON0寄存器的启动位（S）或根据ABD位设置写入I2CxTXB寄存器。主器件硬件等待BFRE = 1，然后发出启动条件。当SCL为高电平时，将SDA驱动为低电平会产生启动条件，并使SCIF位置1。一个TSCL之后，SCL被置为低电平，结束启动序列。图33-15给出了启动条件时序。

图33-15： 启动条件时序

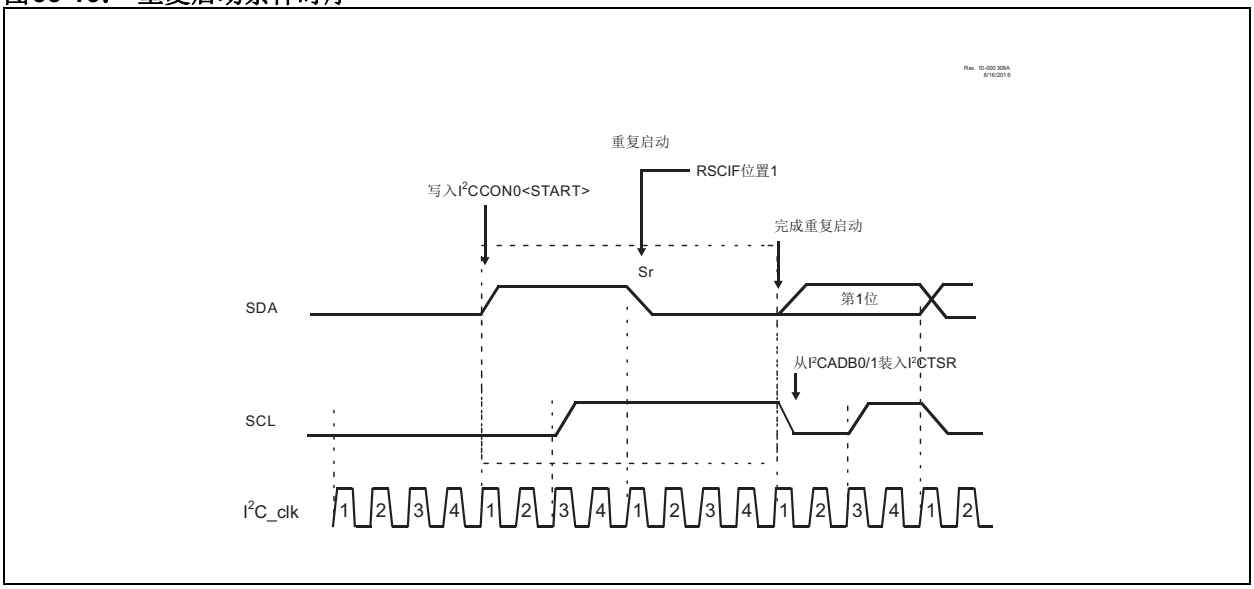


33.5.6 I²C主模式重复启动条件时序

当I2CxCON0寄存器的启动位置1且主模块正在等待重复启动时钟延长事件时（RSEN = 1且I2CxCNT = 0），会产生重复启动条件。

当启动位置1时，SDA引脚释放为高电平并持续TSCL/2的时间。SCL引脚释放（悬空）为高电平并持续TSCL/2的时间。如果检测到SDA引脚为低电平，则总线冲突标志（BCLIF）置1且主器件变为空闲状态。如果检测到SDA为高电平，则SDA引脚将被拉为低电平（启动条件）并持续TSCL的时间。最后，SCL置为低电平，I2CxADB0/1装入移位寄存器。一旦在SDA和SCL引脚上检测到重复启动条件，RSCIF位将置1。图33-16给出了重复启动条件的时序。

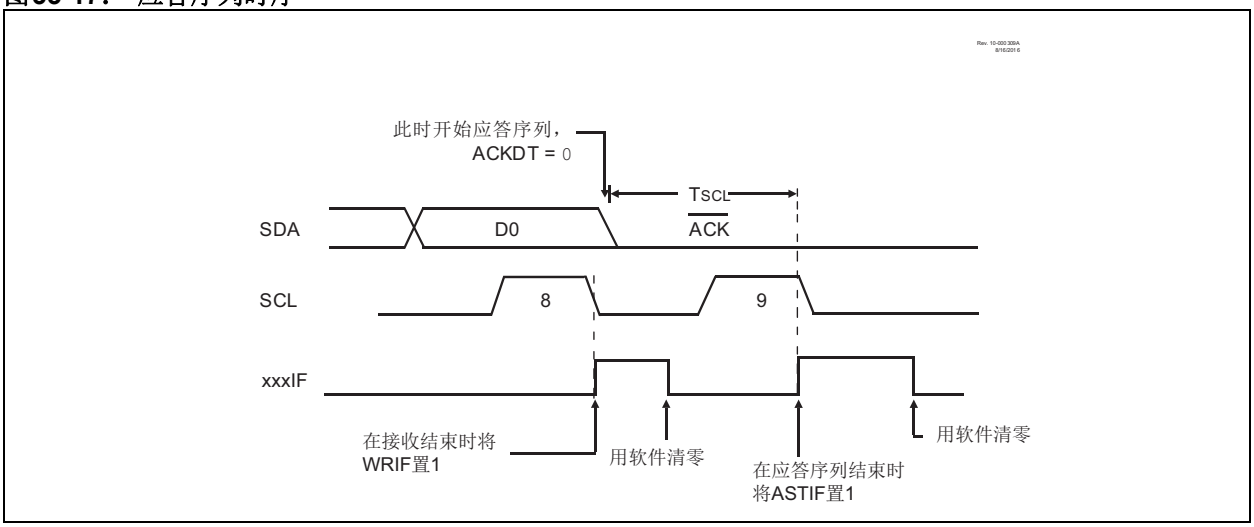
图33-16： 重复启动条件时序



33.5.7 应答序列时序

应答序列在地址/数据字节发送之后自动使能。SCL引脚被拉为低电平，应答数据位（ACKDT/ACKCNT）的内容输出到SDA引脚上。如果用户希望产生应答，则应将ACKDT位清零。否则，用户应在应答序列开始前将ACKDT位置1。然后，主器件等待一个时钟周期（TSCL），SCL引脚释放为高电平。当采样到SCL引脚为高电平（时钟仲裁）时，主器件再进行一个TSCL周期的计数。然后SCL引脚被拉为低电平。图33-17给出了应答序列的时序。

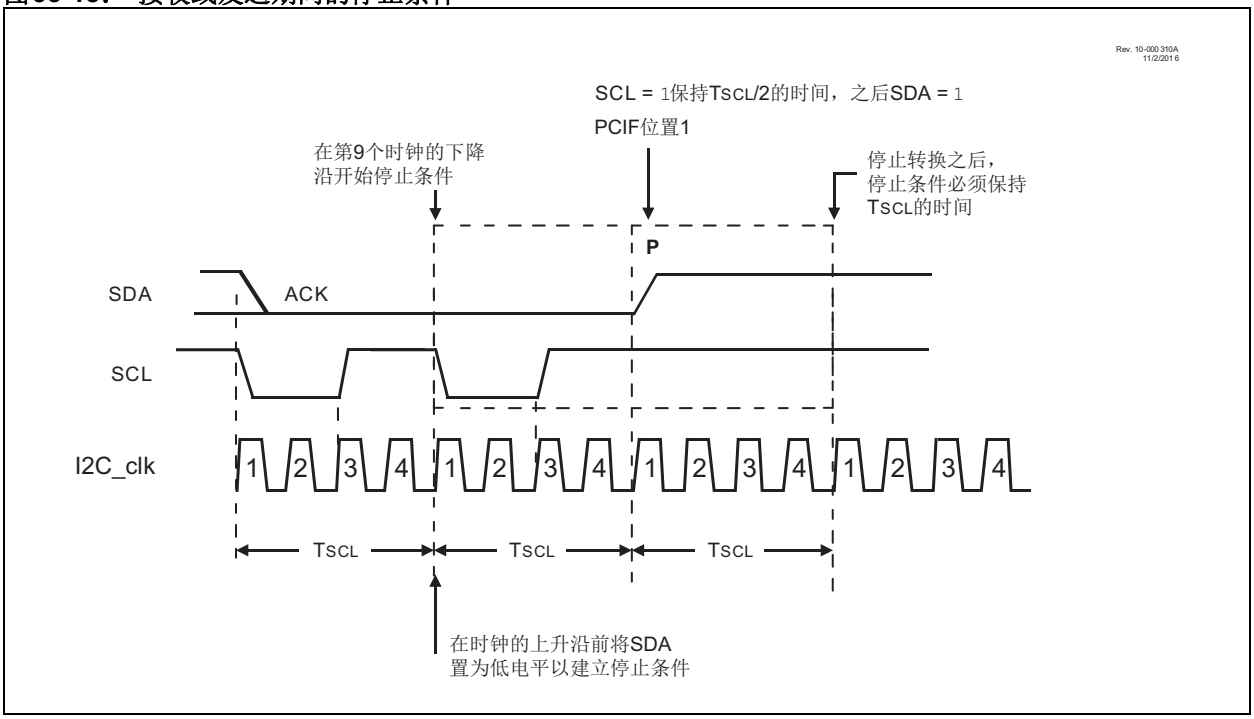
图33-17： 应答序列时序



33.5.8 停止条件时序

当I2CxCNT = 0时，在接收/发送结束时，SDA引脚上将发出停止位。在接收/发送序列的上一个字节之后，SCL线保持低电平。主器件将SDA线置为低电平。然后，SCL引脚会在TSCL/2后释放为高电平并被检测到高电平。之后，SDA引脚被释放。当SDA引脚转换为高电平且SCL也是高电平时，I2CxIF寄存器的PCIF位置1。图33-18给出了停止条件的时序。

图33-18： 接收或发送期间的停止条件



33.5.9 7位寻址模式下的主发送

本节介绍在7位寻址模式下，配置为I^2C主器件且正在发送数据的I^2C模块的事件序列。图33-19用直观的方式对此作了说明。

1. 如果ABD = 0；即，使能地址缓冲区

主软件将在一个序列中发送的字节数装入I2CxCNT，将R/W = 0的从地址装入I2CxADB1，将第一个数据字节装入I2CxTXB。主软件必须将启动（S）位置1才能启动通信。

如果ABD = 1；即，禁止地址缓冲区

主软件将要在一个序列中发送的字节数装入I2CxCNT，将R/W = 0的从地址装入I2CxTXB寄存器。写入I2CxTXB将在总线上发出启动条件并将S位置1。在这种情况下，软件对S位执行的写操作将被忽略。

2. 主器件硬件等待 BFRE 位置 1，然后移出启动条件和地址。
3. 如果发送缓冲区为空（即 TXBE = 1）且I2CxCNT! = 0，则I2CxTXIF和MDR位置1，时钟在第8个SCL下降沿延长。可通过将下一个数据字节装入I2CxTXB寄存器来启动时钟。
4. 主器件发送第9个SCL脉冲以便获得ACK信号。
5. 如果主器件硬件接收到来自从器件的 ACK 信号，它会将发送缓冲区（I2CxTXB）中的下一个字节装入移位寄存器，并且I2CxCNT寄存器的值会递减。
6. 如果接收到NACK信号，主器件硬件将发出停止或重复启动条件
7. 如果ABD = 0；即，使能地址缓冲区

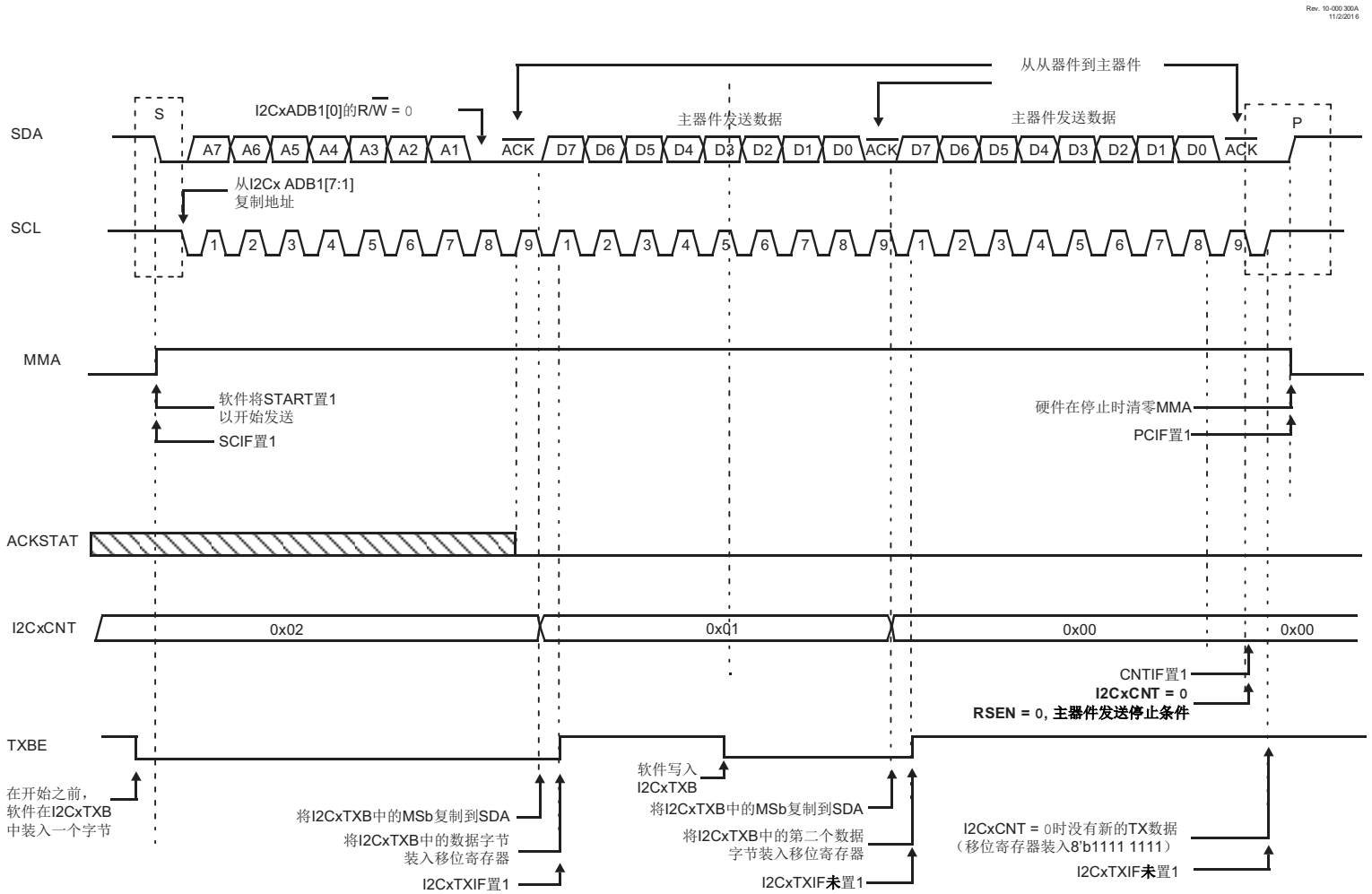
如果I2CxCNT = 0，主器件硬件将发出停止条件或在RSEN = 1时将MDR置1，并等待软件将启动位再次置1以发出重复启动条件。

如果ABD = 1；即，禁止地址缓冲区

如果I2CxCNT = 0，主器件硬件将发出停止条件或在RSEN = 1时将MDR置1，并等待软件将新地址写入I2CxTXB寄存器。在这种情况下，软件对S位执行的写操作将被忽略。

8. 主器件硬件将数据输出到SDA。
9. 如果TXBE = 1且I2CxCNT! = 0，则I2CxTXIF和MDR位置1，时钟在第8个SCL下降沿延长。用户可通过将下一个数据字节写入I2CxTXB寄存器来释放时钟。
10. 主器件硬件随时钟移入来自从器件的 ACK 信号，并将I2CTXB的下一个数据字节装入移位寄存器。I2CxCNT的值递减。
11. 转到步骤7。

图33-19： I²C主模式发送时序（7位地址，使用停止条件）



33.5.10 7位寻址模式下的主接收

本节介绍在7位寻址模式下，配置为I^2C主器件且正在接收数据的I^2C模块的事件序列。图33-20用直观的方式对此作了说明。

1. 主软件将R/W位 = d的从地址装入I2CxADB1，将要在一个序列中接收的字节数装入I2CxCNT寄存器。
2. 主器件硬件等待BFRE位置1；然后移出启动条件和R/W = 1的地址。
3. 主器件发送第9个SCL脉冲以获得ACK信号，主器件硬件随时钟移入来自从器件的ACK信号
4. 如果ABD = 0；即，使能地址缓冲区

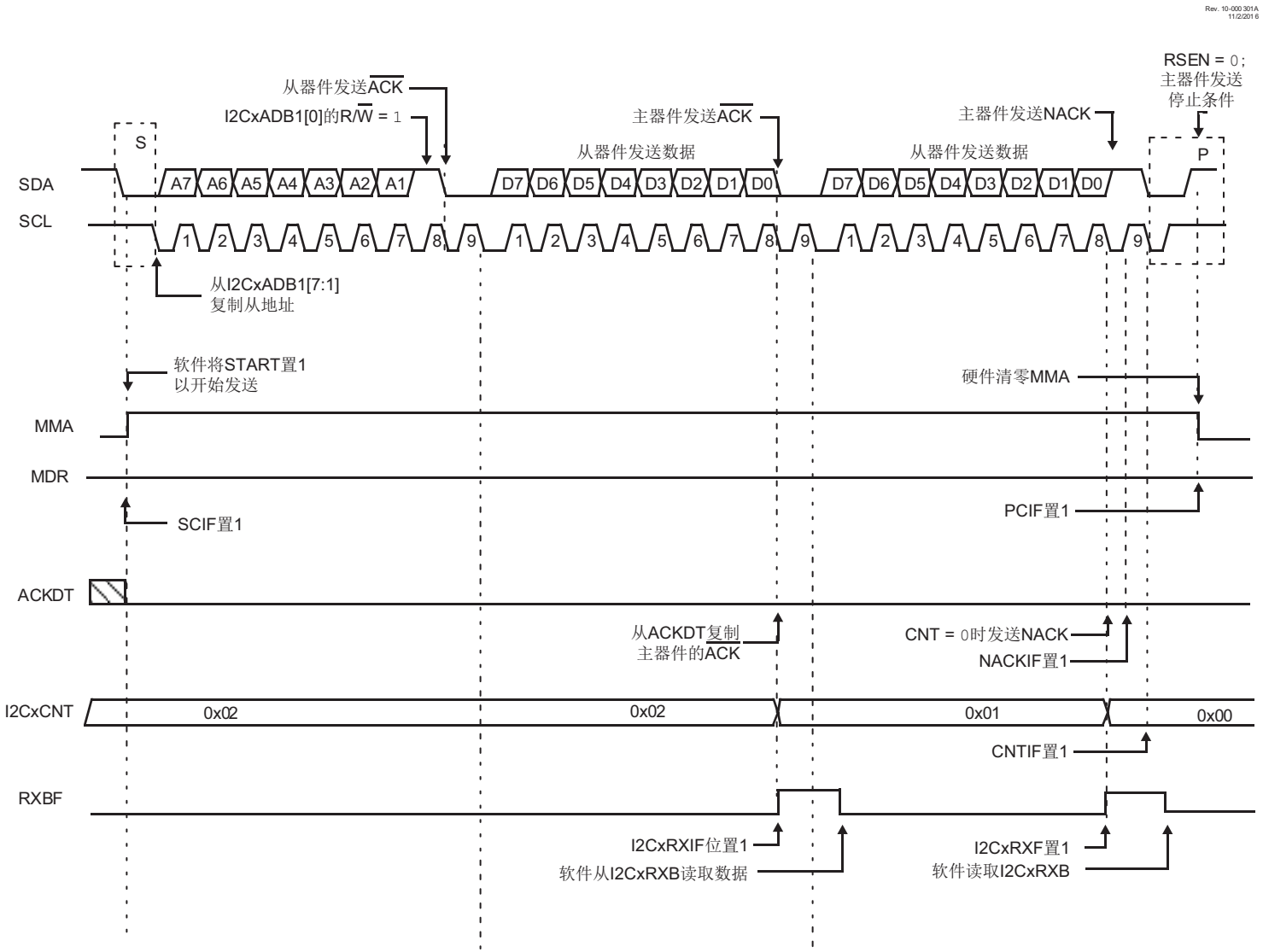
如果发出NACK信号，主器件硬件将发送停止条件或将MDR置1（如果RSEN = 1），并等待用户软件写入S位以便重复启动。

如果ABD = 1；即，禁止地址缓冲区

如果发出NACK信号，主器件硬件将发送停止条件或将MDR置1（如果RSEN = 1），并等待用户软件将新地址装入I2CxTXB。在这种情况下，软件对S位执行的写操作将被忽略。

5. 如果发出ACK信号，主器件硬件会在移位寄存器中接收7位数据。
6. 如果接收缓冲区已满（即RXBF = 1），则时钟在第7个SCL下降沿延长。
7. 主软件必须从 I2CxRXB 读取先前的数据才能清零RXBF。
8. 主器件硬件会在移位寄存器中接收 8 位数据并将其装入I2CxRXB，将I2CxRXIF和RXBF位置1。I2CxCNT的值递减。
9. 如果I2CxCNT! = 0，则主器件硬件随时钟移出ACKDT作为从器件的ACK值。如果I2CxCNT = 0，则主器件硬件随时钟移出ACKCNT作为从器件的ACK值。用户负责正确设置ACKDT和ACKCNT的值。如果用户不将ACKCNT设置为1，则当I2CxCNT变为0时，主器件硬件始终不会发送NACK信号。由于总线上没有NACK信号，主器件硬件也不会发出停止条件。
10. 转到步骤4。

图33-20： I²C主模式接收时序（7位地址）



33.5.11 10位寻址模式下的主发送

本节介绍在10位寻址模式下，配置为I^2C主器件且正在发送数据的I^2C模块的事件序列。图33-21用直观的方式对此作了说明。

1. 如果ABD = 0；即，使能地址缓冲区

主软件将一个序列中要发送的字节的数量装入I2CxCNT，将R/W = 0的从地址的高地址字节装入I2CxADB1，将低地址字节装入I2CxADB0，将第一个数据字节装入I2CxTXB。主软件必须将启动（S）位置1才能启动通信。

如果ABD = 1；即，禁止地址缓冲区

主软件将一个序列中要发送的字节的数量装入I2CxCNT，将R/W = 0的从地址的高地址字节装入I2CxTXB寄存器。写入I2CxTXB将在总线上发出启动条件并将S位置1。在这种情况下，软件对S位执行的写操作将被忽略。

2. 主器件硬件等待BFRE位置1；然后移出启动条件和高地址并等待应答。
3. 如果发出NACK信号，则主器件硬件发送停止条件。
4. 如果ABD = 0；即使能地址缓冲区

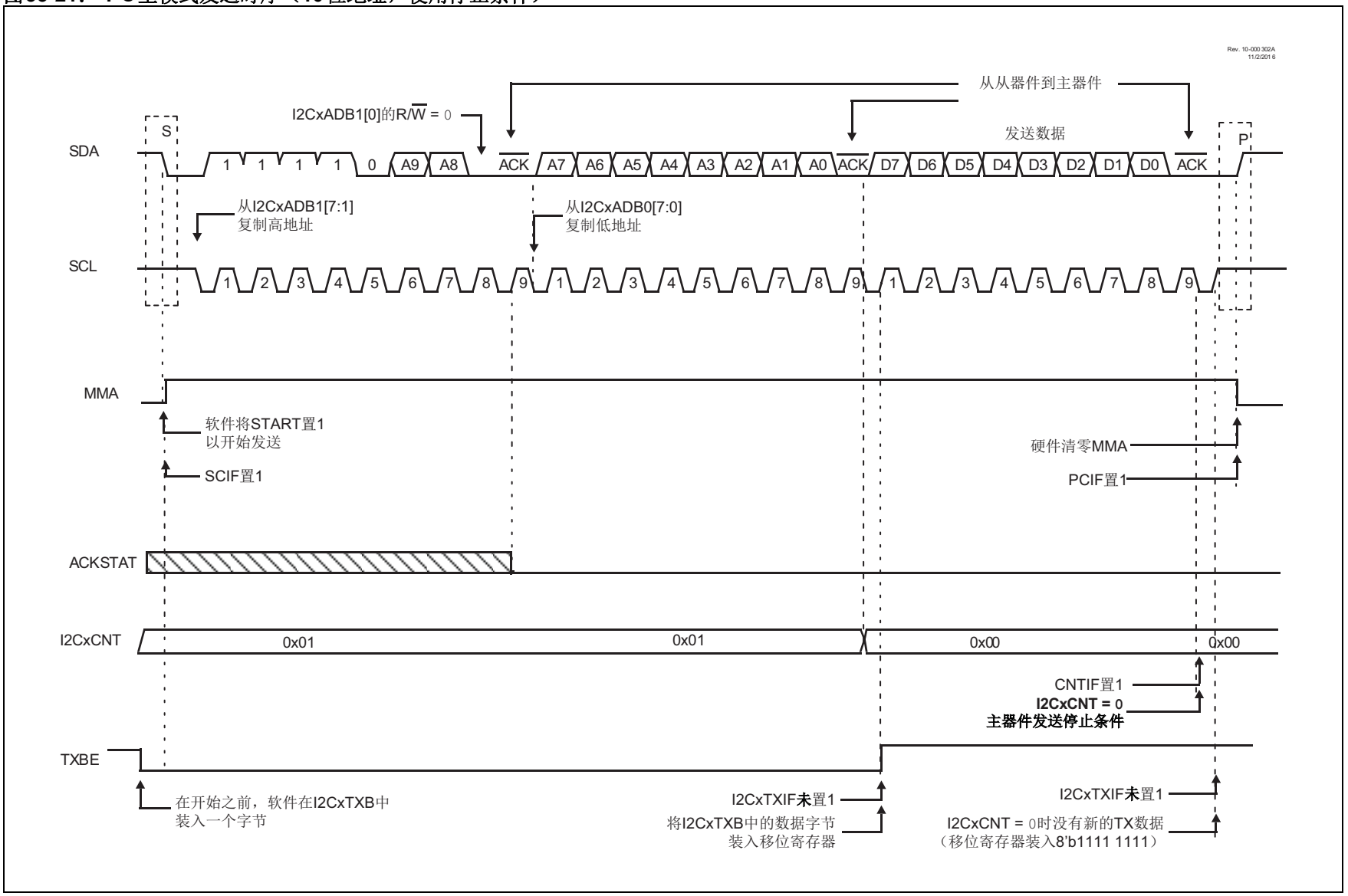
如果发出ACK信号，则主器件硬件发送来自I2CxADB0的低地址字节。

如果ABD = 1；即禁止地址缓冲区

如果发出ACK信号，则主器件硬件将TXIF和MDR位置1，软件必须将低地址字节写入I2CxTXB。写入I2CxTXB将在总线上发送低地址。

5. 如果TXBE = 1且I2CxCNT!= 0，则I2CxTXIF和MDR位置1。时钟在第8个SCL下降沿延长，直到主软件将下一个数据字节写入I2CxTXB。
6. 主器件硬件发送第9个SCL脉冲以便获得来自从器件的ACK信号，并将I2CxTXB的内容装入移位寄存器。I2CxCNT的值递减。
7. 如果从器件发送NACK信号，则主器件硬件将发送停止条件结束发送。
8. 如果从器件发送ACK信号，则主器件硬件会将移位寄存器中的数据输出到SDA。在第8个SCL下降沿检查I2CxCNT值。如果I2CxCNT = 0；则主器件硬件发送第9个SCL脉冲以便获得ACK信号且CNTIF置1。
9. 如果I2CxCNT ! = 0，请转到步骤5。

图33-21： I²C主模式发送时序（10位地址，使用停止条件）

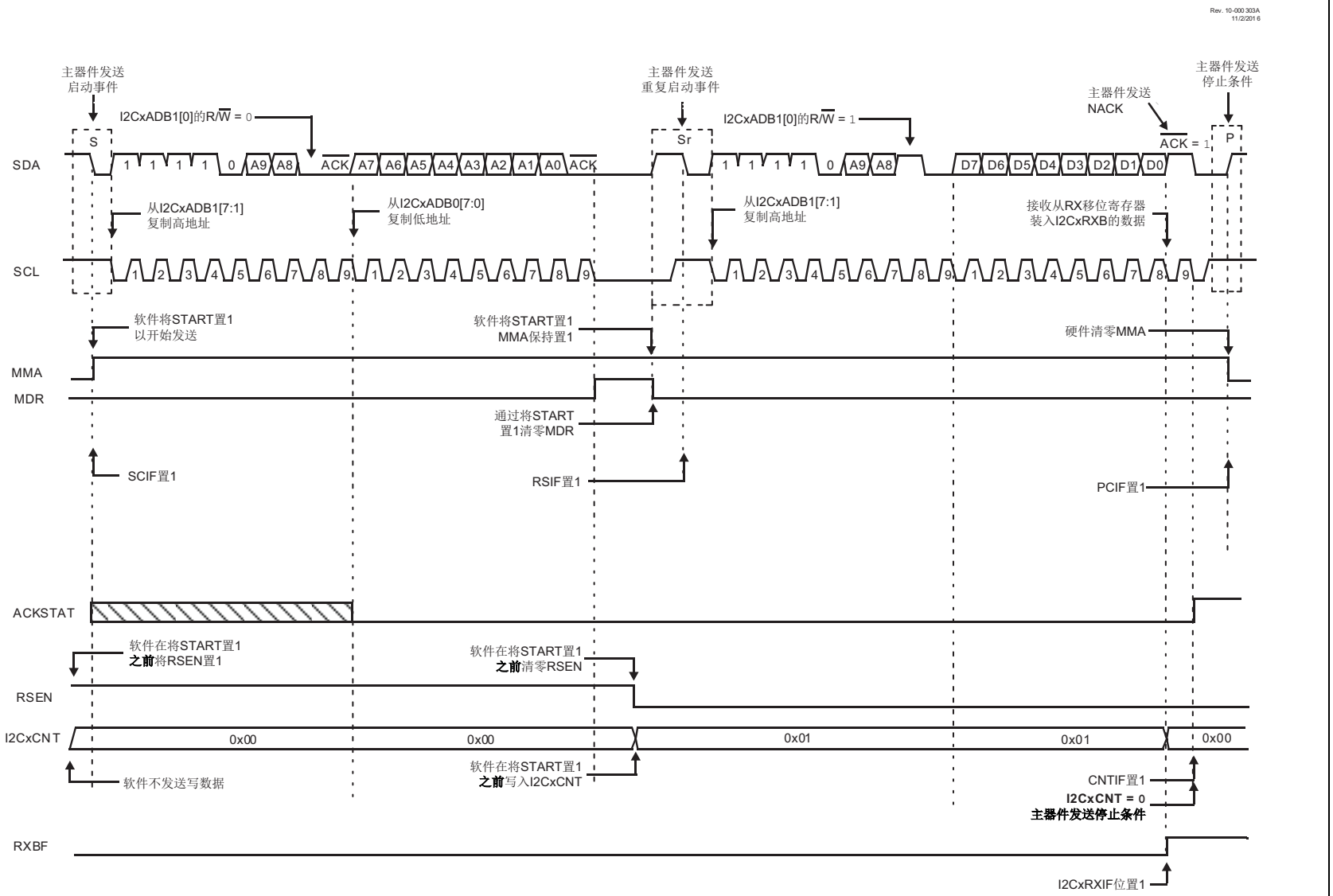


33.5.12 10位寻址模式下的主接收

本节介绍在10位寻址模式下，配置为I^2C主器件且正在接收数据的I^2C模块的事件序列。图33-22用直观的方式对此作了说明。

1. 主软件将高地址字节装入 I2CxADB1，将低地址字节装入I2CxADB0以实现写操作，并将重复启动使能（RSTEN）位置1。
2. 主软件将START位置1。
3. 主器件硬件等待 BFRE 位置 1；然后移出启动条件和高地址并等待应答。
4. 如果从器件以NACK进行响应，则主器件硬件将发送停止条件结束通信。
5. 如果从器件以 ACK 进行响应，则主器件硬件将移出低地址。
6. 如果发送缓冲区空标志（TXBE）置 1 且 I2CxCNT! = 0，则时钟在第8个SCL下降沿延长。这将允许主软件将下一个数据写入I2CxTXB。
7. 主器件硬件发送第9个SCL脉冲以便获得来自从器件的ACK信号，并将I2CxTXB的内容装入移位寄存器。
8. 如果从器件以NACK进行响应，则主器件硬件将发送停止条件结束通信。
9. 如果从器件以ACK进行响应且I2CxCNT = 0，则主器件硬件会将MDR位置1，请转到步骤11。
10. 如果从器件以 ACK 进行响应且 I2CxCNT! = 0，则主器件硬件会将移位寄存器中的数据输出到SDA，并等待来自从器件的ACK信号。转到步骤4。
11. 主软件装入I2CxADB0以供读取，并将要在当前事务中接收的字节数装入I2CCNT。
12. 软件将启动位置1。
13. 主器件硬件移出重复启动条件和R/W = 1的高地址。
14. 主器件发送第9个SCL脉冲以便获得来自从器件的ACK信号。
15. 如果从器件以NACK进行响应，则主器件硬件将发送停止条件或将MDR（RSEN位）置1。
16. 如果从器件以 ACK 进行响应，则主器件硬件会将从器件的7位数据移入移位寄存器。
17. 如果接收缓冲区满标志（RXBF）置1，则时钟在第7个SCL下降沿延长。
18. 主软件可通过读取接收缓冲区中先前的数据来清除时钟延长。
19. 主器件硬件将从器件中的 8 位数据移入移位寄存器并将其装入I2CxRXB。
20. 主软件从I2CxRXB寄存器读取数据。
21. 如果I2CxCNT!= 0，则主器件硬件随时钟移出ACKDT作为从器件的ACK值。
22. 如果I2CxCNT = 0，则主器件硬件随时钟移出ACKCNT作为从器件的ACK值。
23. 转到步骤4。

图33-22： I²C主模式接收时序（10位地址，使用RSTEN位）



33.6 I²C多主器件模式

在多主器件模式下，主器件可通过总线空闲（BFRE）位确定总线何时空闲。当I2CxSTAT0寄存器的BFRE位置1时，将控制I²C总线。在检测到从地址匹配时（ADRIE）将产生中断，这将导致时钟延长并允许用户软件将所寻址的主器件作为从器件来响应。接收到匹配从地址时，从器件工作（SMA）位置1。

如果在任何接收、发送或重复启动/停止条件期间，主器件释放了SCL引脚（允许SCL悬空为高电平），就会发生时钟仲裁。当允许SCL引脚悬空为高电平时，将通过监视SCL线来确定是否实际采样到该引脚为高电平。

注： 在此模式下，从器件硬件优先于主器件硬件。只有在SMA = 0时，才能发起主模式通信。

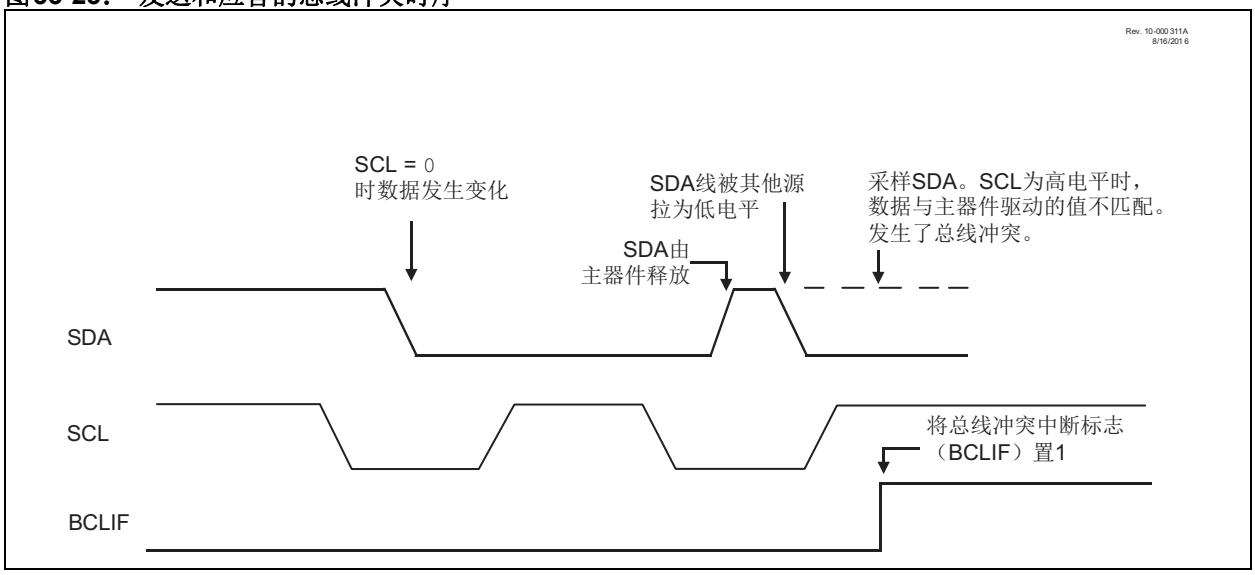
在主器件操作中，必须监视SDA线来进行仲裁，以确认信号电平是否为期望的输出电平。此检查由硬件执行，并将结果保存在BCLIF位中。BCLIF置1时清零MSTACK。可能导致仲裁失败的情况是：

- 地址传输
- 数据传输（主器件写）
- 重复启动条件
- 应答条件

33.6.1 多主器件模式总线冲突

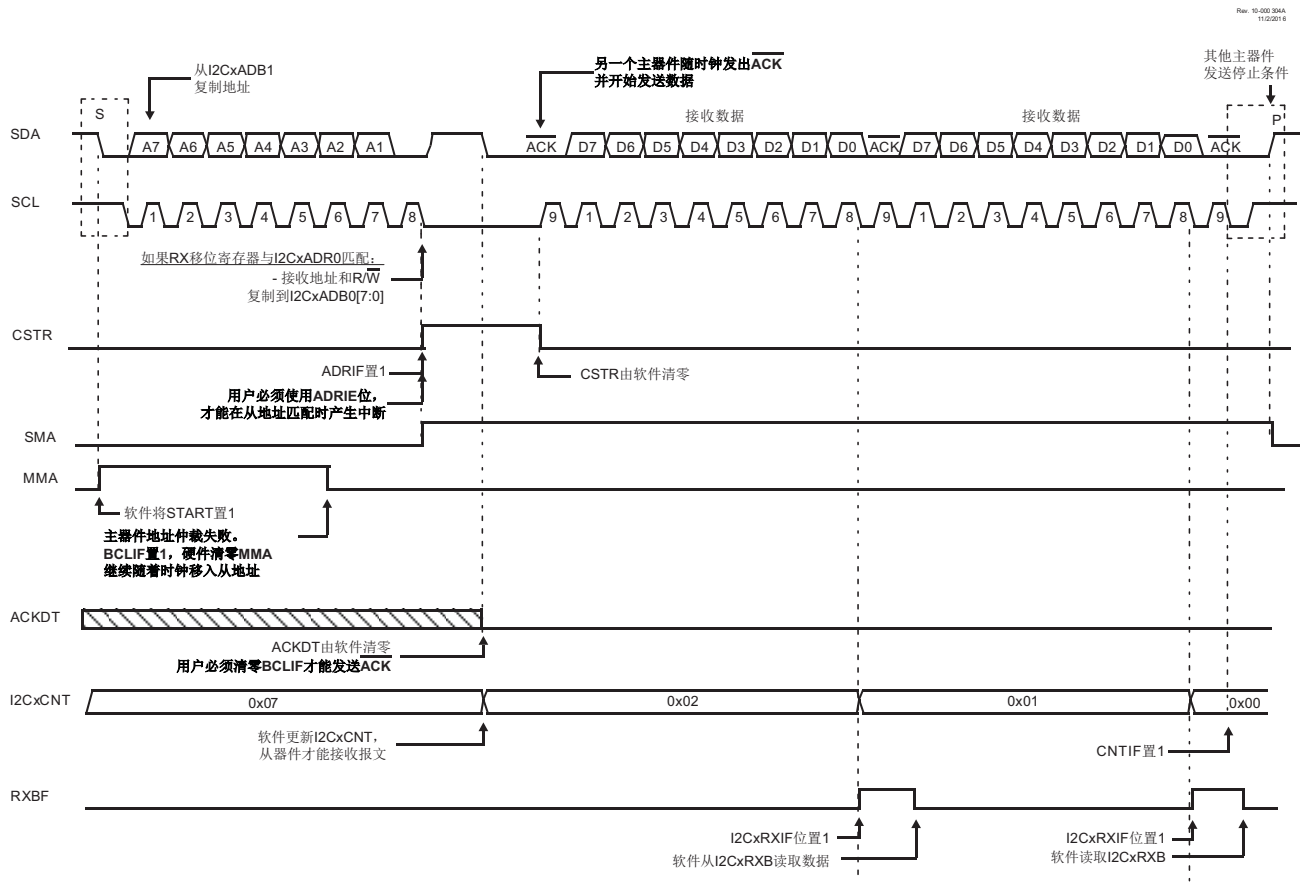
多主器件模式是通过总线仲裁来支持的。当主器件将地址/数据位输出到SDA引脚时，如果一个主器件在SDA引脚上输出1（将SDA引脚悬空为高电平），而另一个主器件输出0，就会发生总线仲裁。当SCL引脚悬空为高电平时，数据是稳定的。如在SDA引脚上期望的数据是1，而实际采样到的数据是0，则发生了总线冲突。主器件会将总线冲突中断标志位BCLIF置1，并将I²C总线复位为空闲状态。有关详细时序图，请参见图33-23。

图33-23： 发送和应答的总线冲突时序



如果发生总线冲突时正在进行发送，则会释放SDA和SCL线。如果发生总线冲突时正在进行重复启动、停止或应答，则会中止操作，并且会释放SDA和SCL线。必须由软件将BCLIF条件清除才能将ACK再次移出到总线，在此之前，模块将始终以NACK进行响应。有关多主器件模式下事务的详细时序图，请参见图33-24。

图33-24： I²C多主器件模式写时序（ADRIE = 1，WRIE = 0，7位地址）



33.7 寄存器定义：I²C控制

本节定义了与I²C总线的控制和状态相关的所有寄存器。

寄存器33-1： I2CxCON0：I²C控制寄存器0

R/W-0	R/W-0	R/W/HC/HS-0	R/C/HS/HC-0	R-0	R/W-0	R/W-0	R/W-0
EN(1,2)	RSEN	S	CSTR(3)	MDR	MODE <2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7 **EN**：I²C模块使能位
1 = 使能I²C模块(1,2)
0 = 禁止I²C模块

bit 6 **RSEN**：重复启动使能位（仅限Mode<2:0> = 1xx时）
1 = 当I2CCNT = 0或ACKSTAT = 1时，在第9个SCL下降沿，将MDR置1
0 = 当I2CCNT = 0或ACKSTAT = 1时，在第9个SCL下降沿，主器件移出停止条件

bit 5 **S**：主器件启动/重复启动位（仅限Mode<2:0> = 1xx时）
当MMA = 0时
1 = START位由用户置1或写入I2CTXB，等待BFRE = 1以通过启动条件开始
0 = 发送启动条件后由硬件清零
当MMA = 1、MDR = 1且暂停以重复启动时
1 = START位由用户置1或写入I2CTXB，通过重复启动条件恢复通信
0 = 发送重复启动条件后由硬件清零
其他情况——写入I2CTXB或置1对启动位没有影响

bit 4 **CSTR**：从器件时钟延长位(3)
1 = 时钟保持低电平（时钟延长）
0 = 使能时钟，释放SCL控制

SMA = 1且RXBF = 1(6)
- 在第7个SCL下降沿由硬件置1
- 用户必须从I2CRXB读取字节才能释放SCL
SMA = 1、TXBE = 1且I2CCNT!= 0
- 在第8个SCL下降沿由硬件置1
- 用户必须向I2CTXB写入字节才能释放SCL
当ADRIE置1时(4)
- 在所接收匹配地址的第8个SCL下降沿由硬件置1
- 用户必须清零CSTR才能释放SCL
SMA = 1且WRIE = 1
- 在所接收数据字节的第8个SCL下降沿由硬件置1
- 用户必须清零CSTR才能释放SCL
SMA = 1且ACKTIE = 1
- 在第9个SCL下降沿由硬件置1
- 用户必须清零CSTR才能释放SCL

bit 3 **MDR：**主器件数据请求*（主器件暂停）*

1 = 主器件状态机暂停，直到读取/写入数据才继续运行（SCL输出保持低电平）

0 = 使能数据的主器件时钟

MMA = 1且RXBF = 1

暂停以接收 - 在第7个SCL下降沿由硬件置1

- 用户必须读取I2CRXB才能释放SCL

MMA = 1、TXBE = 1且I2CCNT!= 0

暂停以发送 - 在第8个SCL下降沿由硬件置1

- 用户必须写入I2CTXB才能释放SCL

ADB = 1

- 10位模式下的高地址和低地址忽略I2CCNT

暂停以重复启动 - 在第9个SCL下降沿由硬件置1

RSEN = 1、MMA = 1且I2CCNT = 0 || ACKSTAT = 1

- 用户必须将START置1或写入I2CTXB才能释放SCL并将重复启动条件移入总线

bit 2-0 **MODE<2:0>：**I^2C模式选择位

111 = I^2C多主器件模式（SMBus 2.0主机），**(5)**
同时以Mode<2:0> = 001和Mode<2:0> = 100方式工作

110 = I^2C多主器件模式（SMBus 2.0主机），**(5)**
同时以Mode<2:0> = 000和Mode<2:0> = 100方式工作

101 = I^2C主模式，10位地址

100 = I^2C主模式，7位地址

011 = I^2C从模式，1个带掩码的10位地址

010 = I^2C从模式，2个10位地址

001 = I^2C从模式，2个带掩码的7位地址

000 = I^2C从模式，4个7位地址

注 1： 必须将SDA和SCL引脚配置为漏极开路且具有内部或外部上拉电阻

2： 必须在PPS中选择SDA和SCL引脚作为输入和输出

3： CSTR可以由多个硬件源置1，所有硬件源必须在SCL线释放之前由用户软件寻址。CSTR是模块状态位，不表示真正的总线状态。

4： 在接收到匹配地址时，SMA在与CSTR相同的SCL边沿置1

5： 在此模式下，ADRIE应置1，这允许中断清除BCLIF条件并允许地址匹配时发出ACK。

6： 在10位从模式下，当ADB = 1时，如果未在移入低地址前从I2CxRXB读取高地址，则CSTR将置1。

寄存器33-2： I2CxCON1：I²C控制寄存器1

R/W-0	R/W-0	R-0	R-0	U-0	R/W/HS-0	R/W/HS-0	R/W-0
ACKCNT(2)	ACKDT(1,2)	ACKSTAT	ACKT	—	RXO	TXU	CSD
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7 **ACKCNT**：应答计数结束位(2)
当I2CCNT = 0时，应答接收数据后发送的值
1 = 无应答（复制到SDA输出）
0 = 应答（复制到SDA输出）

bit 6 **ACKDT**：应答数据位(1,2)
应答匹配地址后发送的值
当I2CCNT! = 0时，应答接收数据后发送的值
1 = 无应答（复制到SDA输出）
0 = 应答（复制到SDA输出）

bit 5 **ACKSTAT**：应答状态位（仅限发送）
1 = 最近的发送未接收到应答
0 = 最近的发送接收到应答

bit 4 **ACKT**：应答时间状态位
1 = 表示I²C总线上有应答序列，在SCL时钟的第8个下降沿置1
0 = 无应答序列，在SCL时钟的第9个上升沿清零

bit 3 **未实现**：读为1'b0

bit 2 **RXO**：接收上溢状态位（MODE<2:0> = 0xx和11x）
仅当CSD= 1时才能将此位置1。
1 = 在SMA = 1时置1，主器件在RXBF = 1时随时钟移入数据
0 = 无从器件上溢条件

bit 1 **TXU**：发送下溢状态位（MODE<2:0> = 0xx和11x）
仅当CSTRDIS = 1时才能将此位置1。
1 = 在SMA = 1时置1，主器件在TXBE = 1时随时钟移出数据
0 = 无从器件下溢条件

bit 0 **CSD**：时钟延长禁止位（MODE<2:0> = 0xx和11x）
1 = 当SMA = 1时，CSTR位始终不会置1
0 = 从器件正常进行时钟延长

注 1： 必须保证从软件写入ACKDT位到清零CSTR之间的SDA数据建立时间最短。
2： 当I2CxSTAT1或I2CxERR寄存器中指示总线错误时，I²C硬件仍可生成NACK。

寄存器33-3： I2CxCON2：I²C控制寄存器2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ACNT	GCEN	FME	ADB	SDAHT<1:0>		BFRET<1:0>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7 **ACNT**：自动装入I²C计数寄存器使能位
1 = 地址后接收或发送的第一个字节自动装入I2CCNT寄存器中。在值移入/移出移位寄存器的同时装入I2CCNT寄存器。
ACKDT用于确定接收报文的地址字节和第一个数据字节的ACK/NACK值。这可防止针对会更新I2CCNT寄存器的字节发送I2CCNT<NACK>。
0 = 禁止I2CCNT的自动装入

bit 6 **GCEN**：广播呼叫地址使能位（MODE<2:0> = 00x和11x）
1 = 广播呼叫地址0x00会导致地址匹配事件
0 = 禁止广播呼叫地址

bit 5 **FME**：快速模式使能位
1 = 在将SCL驱动为低电平之前，仅一次采样到SCL为高电平。（FSCL = FCLK/4）
0 = 在将SCL驱动为低电平之前，两次采样到SCL为高电平。（FSCL = FCLK/5）

bit 4 **ADB**：地址数据缓冲区禁止位
1 = 接收的地址数据装入I2CADB和I2CRXB
发送的地址数据从I2CTXB装入
0 = 接收的地址数据仅装入I2CADB
发送的地址数据从I2CADB0/1寄存器装入

bit 3-2 **SDAHT<1:0>**：SDA保持时间选择位
11 = 保留
10 = 在SCL的下降沿之后，在SDA上最少有30 ns的保持时间
01 = 在SCL的下降沿之后，在SDA上最少有100 ns的保持时间
00 = 在SCL的下降沿之后，在SDA上最少有300 ns的保持时间

bit 1-0 **BFRET<1:0>**：总线空闲时间选择位
11 = 64个I²C时钟脉冲
10 = 32个I²C时钟脉冲
01 = 16个I²C时钟脉冲
00 = 8个I²C时钟脉冲

寄存器33-4：　I2CxCLK：I²C时钟选择寄存器

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	CLK<3:0>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位　HC = 硬件清零位

bit 7-4　**未实现：**读为0

bit 3-0　**CLK<3:0>：**I²C时钟选择位

CLK<3:0>	I²Cx时钟选择
1010-1111	保留
1001	SMT1溢出
1000	TMR6后分频输出
0111	TMR4后分频输出
0110	TMR2后分频输出
0101	TMR0溢出
0100	时钟参考输出
0011	MFINTOSC（500 kHz）
0010	HFINTOSC
0001	Fosc
0000	Fosc/4

寄存器33-5： I2CxBTO：I²C总线超时选择寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	BTO<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位　HC = 硬件清零位

bit 7-3 **未实现：**读为0

bit 2-0 **BTO<2:0>：**I²C总线超时选择位

BTO<2:0>	**I²Cx总线超时选择**
111	CLC4OUT
110	CLC3OUT
101	CLC2OUT
100	CLC1OUT
011	TMR6后分频输出
010	TMR4后分频输出
001	TMR2后分频输出
000	保留

寄存器33-6： I2CxSTAT0：I²C状态寄存器0

R-0	R-0	R-0	R-0	R-0	U-0	U-0	U-0
BFRE(3)	SMA	MMA	R(1, 2)	D	—	—	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7 **BFRE：**总线空闲状态位(3)
1 = 表示I²C总线空闲
SCL和SDA保持高电平的时间与I2CCON2<BFRET<1:0>>位选择的超时时间相等。
I2CCLK必须选择有效时钟源，该位才能正常工作。
0 = 总线不空闲（当没有选择I2CCLK时，该位保持清零）

bit 6 **SMA：**从模块工作状态位
1 = 在所接收7位匹配从地址的第8个SCL下降沿后置1
在所接收10位匹配从地址的低地址的第8个SCL下降沿后置1
在所接收10位匹配从地址的高地址（带读地址）的第8个SCL下降沿后置1（仅在前一个匹配高地址和低地址（带写地址）后）
0 = 通过在总线上检测到的任何重复启动/停止条件清零
通过BTOIF和BCLIF条件清零

bit 5 **MMA：**主模块工作状态位
1 = 主模式状态机工作
主模式状态机在总线上发出启动条件时置1
0 = 主模式状态机空闲
BCLIF置1时清零
主器件移出停止条件时清零
对于BTOIF条件，在主器件成功移出停止条件时清零

bit 4 **R：**读信息位(1, 2)
1 = 表示上一个接收的匹配（高）地址为读请求
0 = 表示上一个接收的匹配（高）地址为写请求

bit 3 **D：**数据位
1 = 表示上一个接收或发送的字节是数据
0 = 表示上一个接收或发送的字节是地址

bit 2-0 **未实现：**读为1'b0

注 1： 该位保存上一次接收地址匹配后的R位信息。主器件发送的地址或出现在总线上但不匹配的地址不影响该位。
2： 从模式下禁止I2CxCLK寄存器中的时钟请求和输入。
3： 软件必须使用EN位强制主器件硬件或从器件硬件进入空闲状态。

寄存器33-7： I2CxSTAT1：I²C状态寄存器1

R/W/HS-0	U-0	R-1	U-0	R/W/HS-0	R/S-0/0	U-0	R-0
TXWE(2)	—	TXBE(1, 3)	—	RXRE(2)	CLRBF	—	RXBF(1,3)
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7 **TXWE**：发送写错误状态位(2)
1 = 在I2CTXB已满时向其写入一个字节的新数据（必须用软件清零）
0 = 无发送写错误

bit 6 **未实现**：读为0

bit 5 **TXBE**：发送缓冲区空状态位
1 = I2CTXB为空（通过写入I2CTXB寄存器清零）
0 = I2CTXB已满

bit 4 **未实现**：读为0

bit 3 **RXRE**：接收读错误状态位
1 = 在I2CxRXB为空时从中读取一个数据字节（必须由软件清零）
0 = 无接收上溢

bit 2 **CLRBF**：清零缓冲区位
将该位置1可清空接收和发送缓冲区，导致RXBF和TXBE复位。
将该位置1可清零RXIF和TXIF中断标志。
该位为仅置1特殊功能位，始终读为0

bit 1 **未实现**：读为0

bit 0 **RXBF**：接收缓冲区满状态位
1 = I2CRXB已接收新数据（通过读取I2CRXB寄存器清零）
0 = I2CRXB为空

注 1：当I2CEN = 0时，位保持复位状态。
2：将针对从地址和主/从数据读字节发送NACK。
3：用于DMA操作的触发信号。

寄存器33-8： I2CxERR：I²C错误寄存器

U-0	R/W/HS-0	R/W/HS-0	R/W/HS-0	U-0	R/W-0	R/W-0	R/W-0
—	BTOIF(1,2)	BCLIF(1)	NACKIF(1)	—	BTOIE	BCLIE	NACKIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7 **未实现：**读为0

bit 6 **BTOIF：**总线超时中断标志位(1,2)
1 = 发生总线超时
0 = 无总线超时

bit 5 **BCLIF：**总线冲突检测中断标志位(1)
1 = 检测到总线冲突（在SCL输入的上升沿，SDA输出为高电平且采样到输入为低电平）
从模式和主模式下，模块立即进入空闲状态
多主器件模式尝试匹配从地址，并且/或者进入空闲状态
0 = 未检测到总线冲突

bit 4 **NACKIF：**NACK检测中断标志位(1)
1 = 当SMA = 1 || MMA = 1时，在总线上检测到NACK
当任何TXWRE、RXRDE、TXUF和RXOVR位置1时，NACKIF也会置1。
0 = 未检测到NACK/错误
在针对不匹配的从地址发送NACK时，NACKIF不置1

bit 3 **未实现：**读为0

bit 2 **BTOIE：**总线超时中断允许位
1 = 允许总线超时中断
0 = 不允许总线超时

bit 1 **BCLIE：**总线冲突检测中断允许位
1 = 允许总线冲突中断
0 = 禁止总线冲突中断

bit 0 **NACKIE：**NACK检测中断允许位
1 = 允许NACKIF中断
0 = 禁止NACKIF中断

注 1： 所允许错误中断标志进行逻辑或运算可形成PIRx<I2CEIF>位。
2： 用户软件必须在I2CBTO寄存器中选择总线超时源。

寄存器33-9： I2CxCNT：I²C字节计数寄存器

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
CNT<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7-0 **CNT<7:0>**：I²C字节计数寄存器位

接收数据时，

如果一个新的数据字节装入I2CxRXB，则在第8个SCL边沿递减

发送数据时，

如果一个新的数据字节移出I2CxTXB，则在第9个SCL边沿递减

当I2CxCNT = 0时，CNTIF标志在第9个SCL下降沿置1。（字节计数不能递减至0以下）

注 1： 建议仅当模块空闲（MMA = 0，SMA = 0）或进行时钟延长（CSTR = 1 || MDR = 1）时写入该寄存器。

寄存器33-10： I2CxPIR：I2CxIF中断标志寄存器

R/W/HS-0	R/W/HS-0	U-0	R/W/HS-0	R/W/HS-0	R/W/HS-0	R/W/HS-0	R/W/HS-0
CNTIF	ACKTIF	—	WRIF	ADRIF	PCIF	RSCIF	SCIF
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7 **CNTIF**：字节计数中断标志位
1 = 当I2CCNT = 0时，通过SCL的第9个下降沿置1。
0 = 未出现I2CCNT条件。

bit 6 **ACKTIF**：应答状态时间中断标志位(2)（MODE<2:0> = 0xx或11x）
1 = 当作为从器件寻址时，通过任何字节的SCL的第9个下降沿置1
0 = 未检测到应答条件

bit 5 **未实现**：读为0

bit 4 **WRIF**：数据写中断标志位（MODE<2:0> = 0xx或11x）
1 = 在所接收数据字节的SCL的第8个下降沿置1
0 = 未检测到数据写条件

bit 3 **ADRIF**：地址中断标志位（MODE<2:0> = 0xx或11x）
1 = 在所接收匹配（高/低）地址字节的SCL的第8个下降沿置1
0 = 未检测到地址条件

bit 2 **PCIF**：停止条件中断标志
1 = 在检测到停止条件时置1
0 = 未检测到停止条件

bit 1 **RSCIF**：重复启动条件中断标志
1 = 在检测到重复启动条件时置1
0 = 未检测到重复启动条件

bit 0 **SCIF**：启动条件中断标志
1 = 在检测到启动条件时置1
0 = 未检测到启动条件

注 1： 所允许中断标志进行逻辑或运算可形成PIRx<I2CxIF>位。
2： R/W位清零的10位匹配高地址字节不会将ACKTIF置1。只有在移入匹配低地址字节后，才会将该位置1。

寄存器33-11： I2CxPIE：I2CxIE中断和保持使能寄存器

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CNTIE	ACKTIE	—	WRIE	ADRIE	PCIE	RSCIE	SCIE
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位 HC = 硬件清零位

bit 7 **CNTIE**：字节计数中断允许位
1 = 当CNTIF置1时
0 = 禁止字节计数中断

bit 6 **ACKTIE**：应答中断和保持使能位
1 = 当ACKTIF置1时
如果生成ACK，则CSTR也会置1。
如果生成NACK，则CSTR不变。
0 = 禁止应答保持和中断

bit 5 **未实现**：读为0

bit 4 **WRIE**：数据写中断和保持使能位
1 = 当WRIF置1时；CSTR置1
0 = 禁止数据写保持和中断

bit 3 **ADRIE**：地址中断和保持使能位
1 = 当ADRIF置1时；CSTR置1
0 = 禁止地址保持和中断

bit 2 **PCIE**：停止条件中断允许位
1 = 允许在检测到停止条件时产生中断
0 = 禁止在检测到停止条件时产生中断

bit 1 **RSCIE**：重复启动条件中断允许位
1 = 允许在检测到重复启动条件时产生中断
0 = 禁止在检测到启动条件时产生中断

bit 0 **SCIE**：启动条件中断允许位
1 = 允许在检测到启动条件时产生中断
0 = 禁止在检测到启动条件时产生中断

注 1： 所允许中断标志进行逻辑或运算可形成PIRx<I2CxIF>位。

寄存器33-12： I2CxADR0：I²C地址0寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位　HC = 硬件清零位

bit 7-0 **ADR<7-0>**：地址1的各个位

MODE<2:0> = 00x | 11x - 7位从模式/多主器件模式

ADR0<7:1>：7位从地址

ADR0<0>：此模式下未使用；位状态为“无关”

MODE<2:0> = 01x - 10位从模式

ADR0<7:0>：10位地址0的低8位

寄存器33-13： I2CxADR1：I²C地址1寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0
ADR14	ADR13	ADR12	ADR11	ADR10	ADR9	ADR8	—
bit 7							bit 0
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位　HC = 硬件清零位

bit 7-1 **ADR[7-1]：**地址或分频比位

MODE<2:0> = 000 | 110 - 7位从模式/多主器件模式

ADR<7:1>：7位从地址

ADR<0>：此模式下未使用；位状态为“无关”

MODE<2:0> = 001 | 111 - 带掩码的7位从模式/多主器件模式

MSK0<7:1>：7位从地址

MSK0<0>：此模式下未使用；位状态为“无关”

MODE<2:0> = 01x - 10位从模式

ADR<14-10>：主器件发送的位模式由 I²C 规范确定，必须等于 11110。但这些位值将通过硬件与接收到的数据进行比较以确定是否匹配。用户负责将这些位设置为11110。

ADR<9-8>：10位地址的高2位

bit 0 **未实现：**读为0。

寄存器33-14： I2CxADR2：I²C地址2寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位　HC = 硬件清零位

bit 7-0 **ADR<7-0>**：地址2的各个位

MODE<2:0> = 000 | 110 - 7位从模式/多主器件模式
ADR<7:1>：7位从地址

MODE<2:0> = 001 | 111 - 带掩码的7位从模式/多主器件模式
ADR<7:1>：7位从地址

MODE<2:0> = 010 - 10位从模式
ADR<7:0>：第二个10位地址的低8位

MODE<2:0> = 011 - 带掩码的10位从模式
MSK0<7-0>：对所接收的地址字节进行掩码，然后将其与I2CxADR0进行比较

寄存器33-15： I2CXADR3：I²C地址3寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0
ADR14	ADR13	ADR12	ADR11	ADR10	ADR9	ADR8	—
bit 15							bit 8
R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0
ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位　HC = 硬件清零位

bit 7-0 **ADR<7-0>**：地址3的各个位

MODE<2:0> = 000 | 110 - 7位从模式/多主器件模式

ADR<7:1>：7位从地址

ADR<0>：此模式下未使用；位状态为“无关”

MODE<2:0> = 001 | 111 - 带掩码的7位从模式/多主器件模式

MSK1<7:1>：7位从地址

MSK1<0>：此模式下未使用；位状态为“无关”

MODE<2:0> = 010 - 10位从模式

ADR<14-10>：主器件发送的位模式由 I²C 规范确定，必须等于 11110。但这些位值将通过硬件与接收到的数据进行比较以确定是否匹配。用户负责将这些位设置为11110。

ADR<9-8>：10位地址的高2位

MODE<2:0> = 011 - 带掩码的10位从模式

MSK0<14-8>：接收到的地址字节的bit *n*与I2CxADR0相比较来检测I²C模式下地址是否匹配

寄存器33-16： I2CxADB0：I²C地址数据缓冲区0寄存器(1)

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADB7	ADB6	ADB5	ADB4	ADB3	ADB2	ADB1	ADB0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位　HC = 硬件清零位

bit 7-0

MODE<2:0> = 00x

ADB<7:1>：地址数据字节

接收到的7位匹配从地址数据

R/$\overline{W}$：读/非写数据位

从7位地址字节接收到的读/写值

MODE<2:0> = 01x

ADB<7:0>：地址数据字节

接收到的10位从地址数据的低8位匹配数据

MODE<2:0> = 100

此模式下未使用；位状态为“无关”

MODE<2:0> = 101

ADB<7:0>：低地址数据字节

复制到发送移位寄存器的低10位地址值

MODE<2:0> = 11x

ADB<7:1>：地址数据字节

接收到的7位匹配从地址

R/$\overline{W}$：读/非写数据位

从7位从地址字节接收到的读/写值

注 1： 该寄存器是只读寄存器，但10位主寻址模式（MODE<2:0> = 101）除外。

寄存器33-17： I2CxADB1：I²C地址数据缓冲区1寄存器(1)

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADB7	ADB6	ADB5	ADB4	ADB3	ADB2	ADB1	ADB0
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1位　HC = 硬件清零位

bit 7-0

MODE<2:0> = 00x

此模式下未使用；位状态为“无关”

MODE<2:0> = 01x

ADB<7:1>：10位地址高字节

接收到的10位匹配高地址数据

R/$\overline{W}$：读/非写数据位

从10位匹配高地址接收到的读/写值

MODE<2:0> = 100

ADB<7:1>：地址数据字节

复制到发送移位寄存器的7位地址值

R/$\overline{W}$：读/非写数据位

复制到发送移位寄存器的读/写值

主器件硬件使用该位生成读操作和写操作。

MODE<2:0> = 101

ADB<7:1>：10位地址高数据字节

复制到发送移位寄存器的10位高地址值

R/$\overline{W}$：读/非写数据位

复制到发送移位寄存器的读/写值

主器件硬件使用该位生成读操作和写操作。

MODE<2:0> = 11x

ADB<7:1>：地址数据字节

复制到发送移位寄存器的7位地址值

R/$\overline{W}$：读/非写数据位

复制到发送移位寄存器的读/写值

主器件硬件使用该位生成读操作和写操作。

注 1： 该寄存器在7位从寻址模式（MODE<2:0> = 0xx）下是只读寄存器。

表33-18：　I²C 8位宏的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
I2CxBTO	—	—	—	—	—	BTO<2:0>			567
I2CxCLK	—	—	—	—	—	CLK<2:0>			566
I2CxPIE	CNTIE	ACKTIE	—	WRIE	ADRIE	PCIE	RSCIE	SCIE	573
I2CxPIR	CNTIF	ACKTIF	—	WRIF	ADRIF	PCIF	RSCIF	SCIF	572
I2CxERR	—	BTOIF	BCLIF	NACKIF	—	BTOIE	BCLIE	NACKIE	570
I2CxSTAT0	BFRE	SMA	MMA	R	D	—	—	—	568
I2CxSTAT1	TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF	569
I2CxCON0	EN	RSEN	S	CSTR	MDR	MODE<2:0>			562
I2CxCON1	ACKCNT	ACKDT	ACKSTAT	ACKT	—	RXOV	TXU	CSD	564
I2CxCON2	ACNT	GCEN	FME	ADB	SDAHT<3:2>		BFRET<1:0>		565
I2CxADR0	ADR<7:0>								574
I2CxADR1	ADR<7:1>							—	575
I2CxADR2	ADR<7:0>								576
I2CxADR3	ADR<7:1>							—	577
I2CxADB0	ADB<7:0>								578
I2CxADB1	ADB<7:0>								579
I2CxCNT	CNT<7:0>								571
I2CxRXB	RXB<7:0>								—
I2CxTXB	TXB<7:0>								—

图注：　— = 未实现，读为0。I²C模块不使用阴影单元。

34.0 CAN模块

该系列器件包含一个控制器局域网（CAN）模块。CAN模块完全向后兼容旧版PIC18器件中的CAN和ECAN模块。

控制器局域网（CAN）模块是用于同其他外设或单片机器件进行通信的串行接口。该接口（或协议）设计旨在实现在嘈杂环境下通信。

CAN模块是通信控制器，用于实现BOSCH规范中定义的CAN 2.0A或CAN 2.0B协议。该模块将支持CAN 1.2、CAN 2.0A、CAN 2.0B Passive和CAN 2.0B Active版本的协议。该模块实现是一个完整的CAN系统；但本数据手册不包含CAN规范。更多详细信息，请参见BOSCH CAN规范。

该模块具有以下特性：

- 实现CAN协议CAN 1.2、CAN 2.0A和CAN 2.0B
- 支持DeviceNet™数据字节过滤器
- 标准和扩展数据帧
- 0-8字节数据长度
- 最高1 Mbps的可编程比特率
- 完全向后兼容旧版PIC18器件上的CAN模块
- 三种工作模式：
  - 模式0——传统模式
  - 模式1——支持DeviceNet的增强型传统模式
  - 模式2——支持DeviceNet的FIFO模式
- 支持远程帧自动处理
- 双缓冲接收器，带有2个优先的接收报文存储缓冲区
- 6个缓冲区，可编程为RX和TX报文缓冲区
- 16个完全（标准/扩展标识符）验收过滤器，可以链接到四个屏蔽器之一
- 2个完全验收过滤屏蔽器，可以分配给任何过滤器
- 1个完全验收过滤器，可用作验收过滤器或验收过滤屏蔽器
- 3个专用发送缓冲区，具有应用特定的优先级和中止功能
- 可编程唤醒功能，集成低通滤波器
- 可编程环回模式支持自检操作
- 通过中断功能针对所有CAN接收器和发送器错误状态发送信号
- 可编程时钟源
- 定时器模块可编程链接，用于实现时间戳功能和网络同步
- 低功耗休眠模式

34.1 模块概述

CAN总线模块由协议引擎以及报文缓冲和控制模块组成。CAN协议引擎自动处理在CAN总线上接收和发送报文的所有功能。通过首先装载相应的数据寄存器发送报文。可通过读取相应的寄存器检查状态和错误。将检查CAN总线上检测到的所有报文是否存在错误，然后与过滤器进行匹配，以确定是否应接收相应报文并将其存入两个接收寄存器之一。

CAN模块支持以下帧类型：

- 标准数据帧
- 扩展数据帧
- 远程帧
- 错误帧
- 过载帧接收

CANRX输入引脚通过CANRXPPS寄存器进行选择。CANTX输出引脚通过每个引脚的RxyPPS寄存器进行选择。

注： CANRX引脚默认为引脚RB3，但CANTX没有默认位置，必须在CAN发送之前分配给某个引脚。

在正常模式下，用户必须确保CANRX的相应TRIS位置1，CANRX的相应TRIS位清零。此外，CANRX的相应ANSEL位必须清零才能禁止模拟输入缓冲区。

注： 与具有CAN功能的旧版Microchip器件不同，CAN引脚可以映射到具有模拟功能的引脚。请确保禁止CANRX引脚上的模拟功能，否则CAN模块将无法正常工作。

34.1.1 模块功能

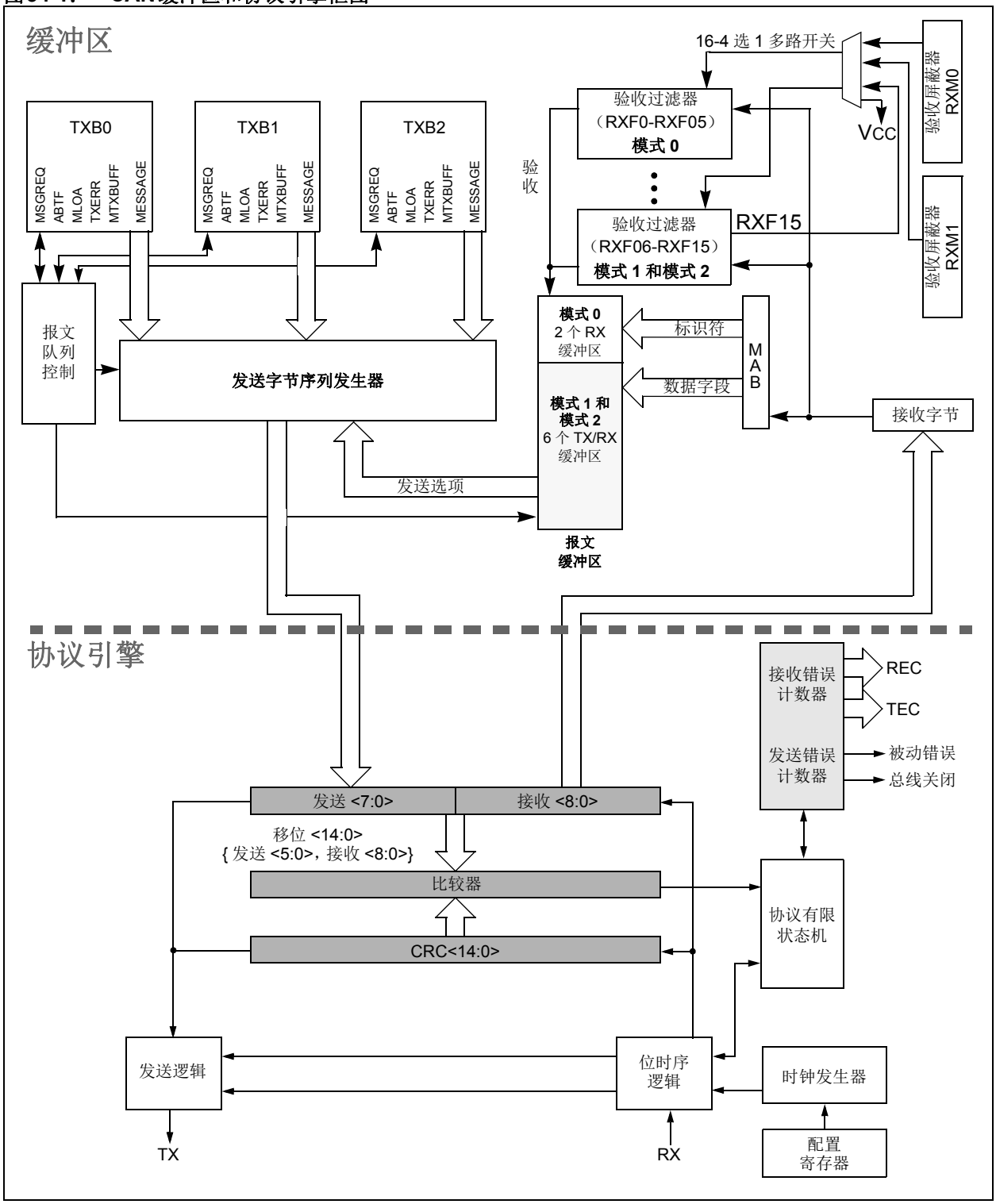
CAN总线模块由协议引擎以及报文缓冲和控制模块组成（见图34-1）。定义模块发送和接收的数据帧类型是理解协议引擎的最佳途径。

以下是在使用CAN模块发送或接收报文之前执行的几个必要初始化步骤，可以根据应用的具体要求适当增加或删减。

1. 使用CANRXPPS和适当的RxyPPS寄存器将CANRX和CANTX功能映射到所需的器件引脚。
2. 为所选的CANRX和CANTX引脚初始化LAT、TRIS和ANSEL位。
3. 确保CAN模块处于配置模式。
4. 选择CAN功能模式。

5. 设置波特率寄存器。
6. 设置过滤器和屏蔽器寄存器。
7. 将 CAN 模块设置为正常模式或应用逻辑所需的任何其他模式。

图34-1： CAN缓冲区和协议引擎框图



34.2 CAN工作模式

CAN模块有6种主要工作模式：

- 配置模式
- 禁止/休眠模式
- 正常工作模式
- 仅监听模式
- 环回模式
- 错误识别模式

除了错误识别模式外，所有模式均通过设置REQOP位（CANCON<7:5>）来请求。错误识别模式通过接收缓冲区寄存器的RXM位请求。通过监视OPMODE位，可以确认进入哪种模式。

切换模式时，在所有等待发送的报文发送完成前，模式不会发生实际变化。因此，用户必须事先验证器件实际上已切换为请求的模式，然后才能执行进一步操作。

注： 如果CANRX和CANTX引脚没有从外部连接到CAN收发器，则模块可能无法从配置模式切换为其他模式。如果在特定的用例或应用中不需要与收发器连接（例如，切换为环回模式进行开发/调试），则CANRX引脚必须通过10k上拉电阻从外部连接到VDD。

34.2.1 配置模式

CAN模块激活前必须先初始化。该操作只能在模块处于配置模式下时进行。通过将REQOP<2:0>位设置为0b100来请求配置模式。只有当状态位OPMODE<2:0>等于0b100时，才能执行初始化。之后，可以写入配置寄存器、验收屏蔽器寄存器和验收过滤器寄存器。

配置模式通过编程错误来防止用户意外违反CAN协议，因为在模块处于在线状态时，所有用于控制模块配置的寄存器均无法修改。发送或接收时，CAN模块不会进入配置模式。以下寄存器只能在配置模式下修改：

- 配置寄存器
- 功能模式选择寄存器
- 位时序寄存器
- 标识符验收过滤器寄存器
- 标识符验收屏蔽器寄存器
- 过滤器和屏蔽器控制寄存器
- 屏蔽器选择寄存器

在配置模式下，模块不会进行发送或接收操作。错误计数器将清零，中断标志保持不变。编程器可以访问在其他模式下访问受限的配置寄存器。I/O引脚将恢复为正常的I/O功能。

34.2.2 禁止/休眠模式

当REQOP<2:0>位设置为001时，模块将进入禁止/休眠模式。该模式类似于通过关闭模块使能功能来禁止其他外设模块。除非模块处于工作状态（即正在接收或发送报文），否则这会导致模块内部时钟停止。如果模块处于工作状态，则会先等待CAN总线上出现11个隐性位，检测到该条件即可判定总线处于空闲状态，之后才能接受模块禁止/休眠命令。OPMODE<2:0> = 001指示模块成功进入模块禁止/休眠模式。在禁止/休眠模式下，模块不会进行发送或接收操作。模块支持因总线活动而将WAKIF位置1。但是，任何待处理的中断都将保留，错误计数器将保留其值。

WAKIF中断是惟一在禁止/休眠模式下仍有效的模块中断。如果WAKDIS清零且WAKIE置1，则只要模块检测到隐性到显性的转换，处理器就会收到中断。唤醒时，模块将自动设置为先前的工作模式。例如，如果在总线活动唤醒时模块从正常模式切换到禁止/休眠模式，模块将自动进入正常模式，而导致模块唤醒的第一条报文丢失。模块将不会生成任何错误帧。固件逻辑必须检测此条件并确保请求重新发送。如果处理器在休眠时收到唤醒中断，则可能会丢失多条报文。丢失报文的实际数量取决于处理器振荡器启动时间和传入报文比特率。

当模块处于禁止/休眠模式下时，CANTX引脚将保持在隐性状态。

34.2.3 正常模式

这是CAN模块的标准工作模式。在此模式下，器件将主动监视所有总线报文并生成应答位和错误帧等。只有在正常模式下，CAN模块才能在CAN总线上发送报文。将CANCON寄存器中的模式请求位清零可激活正常模式。

34.2.4 仅监听模式

仅监听模式为CAN模块提供了一种接收所有报文（包括有错误的报文）的方法。此模式可用于总线监视应用或检测“热插拔”情况下的波特率。进行自动波特率检测时，至少需要有其他两个正在相互通信的节点。可通过测试不同的值直至接收到有效报文，从经验上检测波特率。仅监听模式是一种安静模式，即器件在该状态下不会发送任何报文（包括错误标志或应答信号）。在仅监听模式下，无论RXMn位的设置如何，都会同时接收有效报文和无效报文。此外，仍可使用过滤器和屏蔽器只将特定的有效报文装入接收寄存器中，也可将过滤屏蔽器设置为全零以允许带有任何标识符的报文通过。所有无效报文在此模式下都将被接收，与过滤器和屏蔽器或RXMn接收缓冲区模式位无关。错误计数器在此状态下将被复位和禁止。将CANCON寄存器中的模式请求位设置为0b011可激活仅监听模式。

34.2.5 环回模式

此模式允许在内部将报文从发送缓冲区发送到接收缓冲区，无需通过CAN总线实际发送报文。此模式可用于系统开发和测试。在此模式下，ACK位将被忽略，器件将像接收来自其他节点的报文一样接收自身发送的报文。环回模式是一种安静模式，即器件在该状态下不会发送任何报文（包括错误标志或应答信号）。当器件处于此模式时，TXCAN引脚将恢复为端口I/O。可使用过滤器和屏蔽器仅将特定的报文装入接收寄存器。可将屏蔽器全部设置为零来进入能接收所有报文的模式。将CANCON寄存器中的模式请求位设置为0b010可激活环回模式。

34.2.6 错误识别模式

可以将模块设置为忽略所有错误并接收所有报文。在功能模式0下，将RXBnCON寄存器中的RXM<1:0>位设置为11可激活错误识别模式。在此模式下，发生错误前报文组合缓冲区中的数据被复制到接收缓冲区中，可通过CPU接口读取。

34.3 CAN模块功能模式

除了CAN工作模式之外，CAN模块还提供了三种功能模式。这些模式分别标识为模式0、模式1和模式2。

34.3.1 模式0——传统模式

模式0设计为与PIC18CXX8和PIC18FXX8器件中使用的CAN模块完全兼容。这是所有复位条件下的默认工作模式。因此，只需极少量的代码更改，便可在此CAN模块上使用为PIC18XX8 CAN模块编写的模块代码。

下面列出了模式0下可用的资源：

- 3个发送缓冲区：TXB0、TXB1和TXB2
- 2个接收缓冲区：RXB0和RXB1
- 2 个验收屏蔽器，每个接收缓冲区 1 个：RXM0 和 RXM1
- 6 个验收过滤器，RXB0 有 2 个，RXB1 有 4 个：RXF0、RXF1、RXF2、RXF3、RXF4和RXF5

34.3.2 模式1——增强型传统模式

模式1与模式0类似，只是模式1下有更多的资源可用。共有16个验收过滤器和2个验收屏蔽器寄存器。验收过滤器15可用作验收过滤器或验收屏蔽器寄存器。除了3个发送缓冲区和2个接收缓冲区，还有6个报文缓冲区。可将这些附加缓冲区中的一个或多个编程为发送或接收缓冲区。也可将这些附加缓冲区编程为自动处理RTR报文。

16个验收过滤器寄存器中有14个可动态关联任何接收缓冲区和验收屏蔽器寄存器。此功能可用于将多个过滤器与任何一个缓冲区相关联。

当接收缓冲区被编程为使用标准标识符报文时，完全验收过滤器寄存器的一部分可以用作数据字节过滤器。数据字节过滤器的长度可编程为0位到18位。此功能简化了高级协议（如DeviceNet™协议）的实现。

下面列出了模式1下可用的资源：

- 3个发送缓冲区：TXB0、TXB1和TXB2
- 2个接收缓冲区：RXB0和RXB1
- 6个可编程为TX或RX的缓冲区：B0-B5
- B0-B5的自动RTR处理
- 16个动态分配的验收过滤器：RXF0-RXF15
- 2个专用的验收屏蔽器寄存器；RXF15可编程为第三个屏蔽器：RXM0-RXM1和RXF15
- 针对标准标识符报文的可编程数据过滤器：SDFLC

34.3.3 模式2——增强型FIFO模式

在模式2下，使用2个或多个接收缓冲区来构成接收FIFO（先入先出）缓冲区。接收缓冲区和验收过滤器寄存器之间没有一一对应的关系。任何已使能并链接到任何FIFO接收缓冲区的过滤器都可以生成验收条件并使FIFO更新。

FIFO长度可由用户编程为2-8个缓冲区深。FIFO长度由第一个配置为发送缓冲区的可编程缓冲区决定。例如，如果缓冲区2（B2）被编程为发送缓冲区，则FIFO由RXB0、RXB1、B0和B1组成，因此FIFO的长度为4。如果所有可编程缓冲区均配置为接收缓冲区，则FIFO的最大长度为8。

下面列出了模式2下可用的资源：

- 3个发送缓冲区：TXB0、TXB1和TXB2
- 2个接收缓冲区：RXB0和RXB1
- 6个可编程为TX或RX的缓冲区；接收缓冲区构成FIFO：B0-B5
- B0-B5的自动RTR处理
- 16个验收过滤器：RXF0-RXF15
- 2个专用的验收屏蔽器寄存器； RXF15可编程为第三个屏蔽器：RXM0-RXM1和RXF15
- 针对标准标识符报文的可编程数据过滤器：SDFLC，用于DeviceNet协议

34.4 CAN报文缓冲区

34.4.1 专用发送缓冲区

CAN模块实现了3个专用的发送缓冲区——TXB0、TXB1和TXB2。每个缓冲区都占用14个字节的SRAM并映射到SFR存储器映射中。这些是惟一可在模式0下使用的发送缓冲区。模式1和模式2可使用这些缓冲区，也可使用其他额外的缓冲区。

每个发送缓冲区包含1个控制寄存器（TXBnCON）、4个标识符寄存器（TXBnSIDL、TXBnSIDH、TXBnEIDL和TXBnEIDH）、1个数据长度计数寄存器（TXBnDLC）和8个数据字节寄存器（TXBnDm）。

34.4.2 专用接收缓冲区

CAN模块实现了2个专用的接收缓冲区：RXB0和RXB1。每个缓冲区都占用14个字节的SRAM并映射到SFR存储器映射中。这些是惟一可在模式0下使用的接收缓冲区。模式1和模式2可使用这些缓冲区，也可使用其他额外的缓冲区。

每个接收缓冲区包含1个控制寄存器（RXBnCON）、4个标识符寄存器（RXBnSIDL、RXBnSIDH、RXBnEIDL和RXBnEIDH）、1个数据长度计数寄存器（RXBnDLC）和8个数据字节寄存器（RXBnDm）。

还有一个单独的报文组合缓冲区（MAB），可作为额外的接收缓冲区。MAB始终用于接收来自总线的下一条报文，无法直接供用户固件访问。MAB会将传入的所有报文逐个组合在一起。只有满足相应的验收过滤器条件，报文才会传送到适当的接收缓冲区。

34.4.3 可编程发送/接收缓冲区

CAN模块实现了6个非专用的缓冲区：B0-B5。这些缓冲区可单独编程为发送或接收缓冲区，仅适用于模式1和模式2。与专用的发送和接收缓冲区一样，每个可编程缓冲区都占用14个字节的SRAM并映射到SFR存储器映射中。

每个缓冲区包含1个控制寄存器（BnCON）、4个标识符寄存器（BnSIDL、BnSIDH、BnEIDL和BnEIDH）、1个数据长度计数寄存器（BnDLC）和8个数据字节寄存器（BnDm）。每个寄存器包含两组控制位。用户可根据缓冲区配置为发送缓冲区还是接收缓冲区选择使用相应的控制位组。默认情况下，所有缓冲区均配置为接收缓冲区。将BSEL0寄存器中相应的TXENn位置1可将每个缓冲区单独配置为发送或接收缓冲区。

配置为发送缓冲区时，用户固件可按照类似于访问专用发送缓冲区的顺序访问发送缓冲区。在使能模式1的接收配置中，用户固件还可按所需顺序访问接收缓冲区。但在模式2下，所有接收缓冲区组合在一起构成一个FIFO。实际FIFO长度可由用户固件编程。访问FIFO时必须通过CANCON寄存器中的FIFO指针位（FP<4:0>）来完成。必须注意的是，没有针对FIFO读取乱序问题的硬件保护功能。

34.4.4 可编程自动RTR缓冲区

在模式1和模式2下，6个可编程发送/接收缓冲区中的任何一个均可编程为自动响应预定义的RTR报文，而无需用户固件干预。将BSEL0寄存器中的TX2EN位以及BnCON寄存器中的RTREN位置1可使能自动RTR处理。在此设置下，当接收到RTR请求时，TXREQ位将自动置1，当前缓冲区内容将作为RTR响应自动排队发

送出去。与所有发送缓冲区一样，TXREQ位置1后，缓冲区寄存器立即变成只读状态，对它们进行的任何写操作都将被忽略。

下面概述了自动处理RTR报文所需的步骤：

1. 将BSEL0寄存器中的TXnEN位置1可将缓冲区设置为发送模式。
2. 至少有1个验收过滤器必须与此缓冲区关联并预先装入预期的RTR标识符。
3. BnCON寄存器中的RTREN位必须置1。
4. 缓冲区必须预先装入要作为RTR响应发送的数据。

通常，用户固件会使缓冲区数据寄存器保持最新。如果固件在自动RTR响应的发送过程中试图更新缓冲区，则对缓冲区执行的所有写操作都将被忽略。

34.5 CAN报文发送

34.5.1 启动发送

为使MCU能够对报文缓冲区进行写访问，TXREQ位必须清零，以表明报文缓冲区清除了所有待发送的报文。至少要装入SIDH、SIDL和DLC寄存器。如果报文中存在数据字节，则还要装入数据寄存器。如果报文要使用扩展标识符，则还要装入EIDH:EIDL寄存器，并将EXIDE位置1。

为启动报文发送，必须为每个要发送的缓冲区将TXREQ位置1。TXREQ置1时，将清零TXABT、TXLARB和TXERR位。要成功完成发送，网络上必须至少有一个波特率匹配的节点。

将TXREQ位置1不会启动报文发送；只会将报文缓冲区标记为准备发送。发送将在器件检测到总线可用后开始。之后，器件将开始发送已准备就绪的最高优先级报文。

报文成功发送后，TXREQ位将清零，TXBnIF位将置1，如果将TXBnIE位置1，则会产生中断。

如果报文发送失败，TXREQ位将保持置1，表明该报文仍在等待发送并且以下条件标志之一将被置1。如果报文开始发送但出现一个错误条件，TXERR和IRXIF位将置1，从而会产生中断。如果报文仲裁失败，TXLARB位将置1。

34.5.2 中止发送

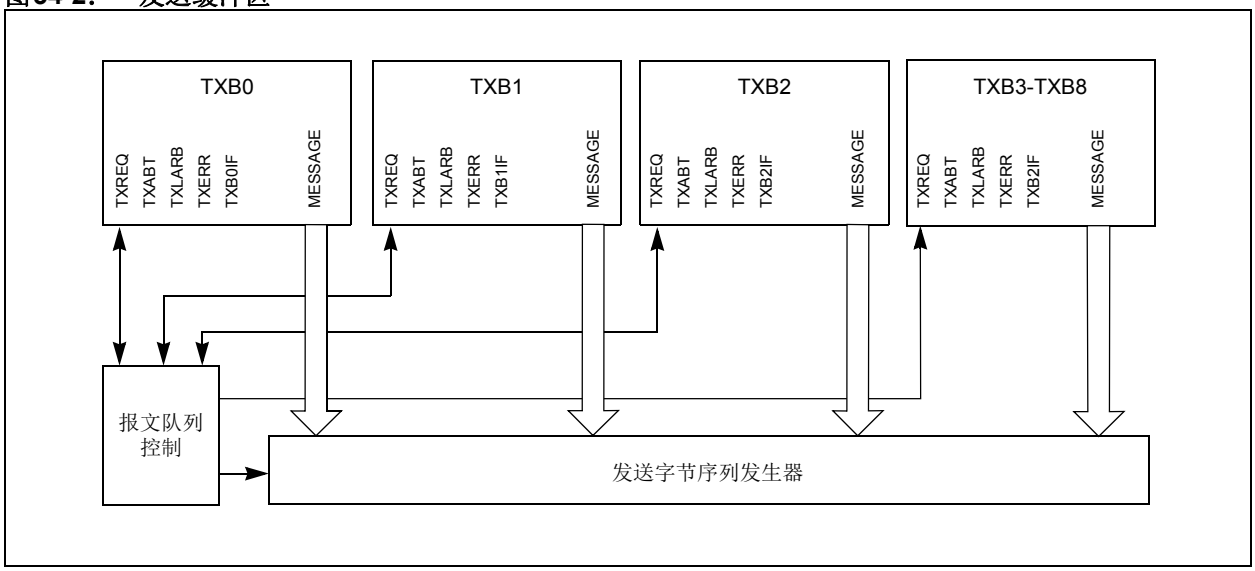
MCU可通过将与相应报文缓冲区关联的TXREQ位（TXBnCON<3>或BnCON<3>）清零来请求中止报文。将ABAT位（CANCON<4>）置1会请求中止所有待处理的报文。如果报文尚未启动发送，或者报文已启动但由于仲裁失败或错误而中断，则将处理中止请求。当模块将相应缓冲区的TXABT位（TXBnCON<6>或BnCON<6>）置1时，即表明中止。如果报文已启动发送，它将尝试完全发送当前报文。如果当前报文已完全发送且仲裁未失败也未出现错误，则TXABT位将不会置1，因为报文已成功发送。同样，如果在中止请求期间正在发送报文，而报文仲裁失败或出现错误，则将不会重新发送报文，TXABT位将置1，表明报文已成功中止。

通过将ABAT或TXABT位置1请求中止后，无法通过清零这些位来取消中止请求。只有CAN模块硬件或POR条件能够清零这些位。

34.5.3 发送优先级

发送优先级是ECAN模块中等待发送的报文的优先级。它与CAN协议中固有的报文仲裁方案隐含的任何优先级没有关系。发送帧起始（Start-of-Frame，SOF）之前，将比较列队等待发送的所有缓冲区的优先级。优先级最高的发送缓冲区将首先发送。如果两个缓冲区的优先级设置相同，则缓冲区编号较大的缓冲区将优先发送。有四个发送优先级。如果特定报文缓冲区的TXP位设置为11，则该缓冲区具有最高优先级。如果特定报文缓冲区的TXP位设置为00，则该缓冲区具有最低优先级。

图34-2： 发送缓冲区



34.6 报文接收

34.6.1 接收报文

在所有接收缓冲区中，MAB始终用于接收来自总线的下一条报文。当有缓冲区用于接收报文或用于保存之前接收的报文时，MCU可访问其他缓冲区。

注：	报文一经验收，MAB 的全部内容将立即移入接收缓冲区。这意味着，无论标识符是何种类型（标准标识符或扩展标识符）或接收到的数据字节数是多少，整个接收缓冲区都将被MAB内容覆盖。因此，接收到任何报文时都必须假定缓冲区内所有寄存器的内容都已被修改。

将报文移入接收缓冲区后，相应的RXFUL位将置1。MCU在完成缓冲区中报文的处理后必须将该位清零，以便使新报文接收到缓冲区中。该位具有正向锁定功能，可确保固件完成报文处理后，模块再次尝试向接收缓冲区中装入新报文。如果允许了接收中断，将生成一个中断，表明已接收到有效报文。

报文装入任何匹配的缓冲区后，用户固件即可通过检查RXBnCON或BnCON寄存器中的过滤器命中位确定究竟是哪个过滤器引起此接收操作。在模式0下，RXBnCON的FILHIT<2:0>位作为过滤器命中位。在模式1和模式2下，BnCON的FILHIT<4:0>位作为过滤器命中位。这两个寄存器还指示当前报文是否为RTR帧。如果RXBnSIDL或BnSIDL寄存器中的EXID/EXIDE位清零，则接收到的报文被视为标准标识符报文。相反，如果EXID位置1，则接收到的报文被视为扩展标识符报文。如果接收到的报文是标准标识符报文，则用户固件需要读取SIDL和SIDH寄存器。如果接收到的报文是扩展标识符报文，固件应读取SIDL、SIDH、EIDL和EIDH寄存器。如果RXBnDLC或BnDLC寄存器包含非零数据计数，则用户固件还应通过访问RXBnDm或BnDm寄存器来读取相应数量的数据字节。当收到的报文是RTR时，如果当前缓冲区未配置为进行自动RTR处理，则用户固件必须手动采取适当的措施并作出响应。

每个接收缓冲区均包含RXM位，用于设置特殊的接收模式。在模式0下，RXBnCON中的RXM<1:0>位共定义了4种接收模式。在模式1和模式2下，RXM1位与EXID屏蔽位和过滤位一起定义了相同的4种接收模式。通常，这些位会设置为00，以接收相应验收过滤器确定的所有有效报文。在这种情况下，是否接收标准报文或扩展报文由验收过滤器寄存器中的EXIDE位确定。在模式0下，如果RXM位设置为01或10，则接收器将分别仅接收带标准标识符或扩展标识符的报文。如果验收过滤器中的EXIDE位的设置与RXM模式不对应，则验收过滤器将不起作用。在模式1和模式2下，将SIDL屏蔽器寄存器中的EXID置1将确保仅接收标准标识符或扩展标识符。上述两种RXM位的模式可用于已知总线上仅传输标准报文或扩展报文的系统。如果将RXM位设置为11（模式1和模式2下的RXM1 = 1），则无论验收过滤器值如何，缓冲区都将接收所有报文。此外，如果报文在帧结束前出错，则错误帧出现之前在MAB中所组合的那部分报文将装入缓冲区。该模式可作为给定

CAN网络的重要调试工具，不应在实际的系统环境中使用，因为实际系统中总是会有一些总线错误，而总线上的所有节点往往都会忽略这些错误。

在模式1和模式2下，当某个可编程缓冲区配置为发送缓冲区并且有一个或多个验收过滤器与其关联时，将丢弃与该验收过滤条件匹配的所有传入报文。为了避免这种情况，用户固件必须确保没有验收过滤器与配置为发送缓冲区的缓冲区关联。

34.6.2 接收优先级

当处于模式0时，RXB0是具有较高优先级的缓冲区，与两个报文验收过滤器相关联。RXB1是具有较低优先级的缓冲区，与四个验收过滤器相关联。由于验收过滤器数量较少，因此RXB0上的匹配条件更为严格，表明RXB0具有较高的优先级。此外，可对RXB0CON寄存器进行配置，以便在RXB0已包含一条有效报文并且接收到另一条有效报文时，不会发生溢出错误，并且无论RXB1是何种验收条件，新的报文都会移入RXB1。此外，还有两个可编程的验收过滤屏蔽器可用，每个接收缓冲区一个（见**第34.4节“CAN报文缓冲区”**）。

在模式1和模式2下，共有16个验收过滤器可用，每个验收过滤器均可动态分配给任何接收缓冲区。缓冲区的编号越小，优先级越高。鉴于此，如果传入报文与2个或多个接收缓冲区验收标准匹配，则编号较小的缓冲区将装入该报文。

34.6.3 增强型FIFO模式

当配置为模式2时，2个专用接收缓冲区与1个或多个可编程发送/接收缓冲区组合，用于创建最多8个缓冲区深的FIFO缓冲区。在此模式下，过滤器和接收缓冲区寄存器之间没有直接关联。任何已使能的过滤器都可以生成验收条件。报文被接受后，将存储在下一个可用的接收缓冲区寄存器中，内部写指针递增。FIFO最多可为8个缓冲区深。整个FIFO必须由连续的接收缓冲区组成。FIFO头部从RXB0缓冲区开始，尾部在B5结束。FIFO的最大长度受限于从B0开始的第一个发送缓冲区是否存在。如果某个缓冲区配置为发送缓冲区，则FIFO长度会相应缩短。例如，如果B3配置为发送缓冲区，则实际FIFO将由RXB0、RXB1、B0、B1和B2共5个缓冲区组成。如果B0配置为发送缓冲区，则FIFO长度将为2。如果没有任何可编程缓冲区配置为发送缓冲区，则FIFO将为8个缓冲区深。如果系统需要更多的发送缓冲区，则应尝试在B0-B5缓冲区的最末端放置发送缓冲区，以最大程度延长可用的FIFO长度。

在FIFO模式下接收到报文时，CANSTAT寄存器中的中断标志代码位（EICODE<4:0>）的值为10000，表示FIFO已收到报文。CANCON寄存器中的FIFO指针位FP<3:0>指向包含未读取数据的缓冲区。这意味着，FIFO指针位用作FIFO读指针。用户应使用FP位并读取相应的缓冲区数据。当不再需要接收数据时，必须将当前缓冲区中的RXFUL位清零，让模块更新FP<3:0>。

要确定FIFO是否为空，用户可以使用FP<3:0>位来访问当前缓冲区中的RXFUL位。如果RXFUL清零，则认为FIFO为空。如果置1，则FIFO可能包含一条或多条报文。在模式2下，模块还在ECANCON寄存器中提供称为FIFO高水印（FIFOWM）的位。只要FIFO只包含1个或4个空缓冲区，该位就可用于产生中断。FIFO高水印中断可作为完整FIFO条件的早期警告。

34.6.4 时间戳

可以将CAN模块编程为针对接收到的每条报文生成时间戳。使能时，模块会为CCP模块生成一个捕捉信号，进而捕捉Timer1、Timer3或Timer5的值。该值可用作报文时间戳。

要使用时间戳功能，请将相应CCPxCAP寄存器的CTS<3:0>位设置为1000，以将CCP模块捕捉输入配置为CAN_rx_timestamp信号。

此外，还可以将CAN_rx_timestamp选作信号测量定时器的信号输入，从而可用于各种其他定时应用。

34.7 报文验收过滤器和屏蔽器

报文验收过滤器和屏蔽器用于确定报文组合缓冲区中的报文是否应装入任何接收缓冲区。MAB中接收到有效报文后，会立即将报文的标识符字段与过滤器值进行比较。如果两者匹配，则该报文将装入相应的接收缓冲区。过滤屏蔽器用于确定过滤器需要检查标识符中的哪些位。下文的表34-1中所示的真值表列出了如何通过将标识符中的各个位与屏蔽器和过滤器进行比较来确定报文是否应装入接收缓冲区。屏蔽器主要用于确定要对哪些位进行验收过滤。如果任意屏蔽位设置为零，则该位将被自动接受，这与过滤位无关。

表34-1： 过滤器/屏蔽器真值表

屏蔽位n	过滤位n	报文标识符位n001	接受或拒绝位n
0	x	x	接受
1	0	0	接受
1	0	1	拒绝
1	1	0	拒绝
1	1	1	接受

图注： x = 无关

在模式0下，验收过滤器RXF0和RXF1以及过滤屏蔽器RXM0与RXB0相关联。过滤器RXF2、RXF3、RXF4和RXF5以及屏蔽器RXM1都与RXB1相关联。

在模式1和模式2下，还有另外10个验收过滤器RXF6-RXF15，因此共有16个可用过滤器。RXF15可用作验收过滤器或验收屏蔽器寄存器。通过将RXFCONn寄存器中的RXFENn位置1或清零，可以单独使能或禁止其中每个验收过滤器。这16个验收过滤器中的任何一个都可以与任何接收缓冲区动态关联。将RXFBCONn寄存器中的相应位置1可实现实际关联。每个RXFBCONn寄存器包含每个过滤器的半字节。此半字节可用于将特定过滤器与任何可用的接收缓冲区相关联。用户固件可以将多个过滤器与任何一个特定的接收缓冲区相关联。

除了过滤器与缓冲区的动态关联之外，在模式1和模式2下，每个过滤器还可与可用的验收屏蔽器寄存器动态关联。MSELn寄存器中的FILn_m位可用于将特定的验收过滤器与验收屏蔽器寄存器相关联。与过滤器到缓冲区的关联一样，也可以将多个屏蔽器与特定的验收过滤器相关联。

当过滤器匹配且报文装入接收缓冲区时，使能报文接收的过滤器编号将被装入FILHIT位中。在RXB1的模式0下，RXB1CON寄存器包含FILHIT<2:0>位。其编码如下：

- 101 = 验收过滤器5（RXF5）
- 100 = 验收过滤器4（RXF4）
- 011 = 验收过滤器3（RXF3）
- 010 = 验收过滤器2（RXF2）
- 001 = 验收过滤器1（RXF1）
- 000 = 验收过滤器0（RXF0）

> **注：** 000和001仅适用于RXB0CON寄存器中的RXB0DBEN位置1的情况，允许RXB0报文滚存到RXB1中。

RXB0DBEN位的编码方式使得这三个位的用法与FILHIT位类似，并且可区分在RXB0中或滚存到RXB1中时过滤器RXF0和RXF1接收报文的情况。

- 111 = 验收过滤器1（RXF1）
- 110 = 验收过滤器0（RXF0）
- 001 = 验收过滤器1（RXF1）
- 000 = 验收过滤器0（RXF0）

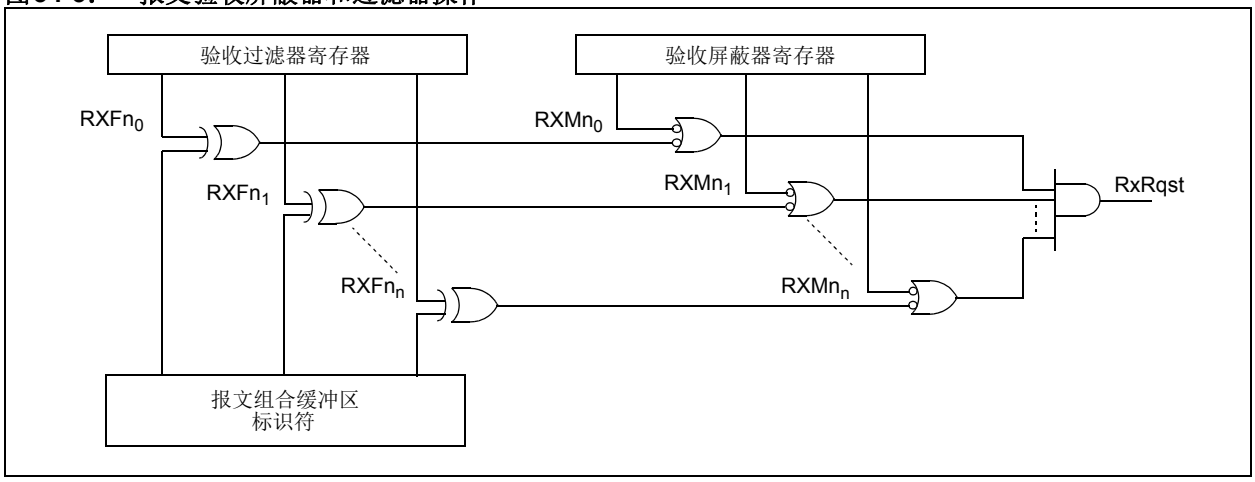
如果RXB0DBEN位清零，则有六种编码对应于六个过滤器。如果将RXB0DBEN位置1，则除了这六种编码对应于六个过滤器外，还有两种编码对应于滚存到RXB1中的RXF0和RXF1过滤器。

在模式1和模式2下，每个缓冲区控制寄存器包含5个过滤器命中位（FILHIT<4:0>）。二进制值0表示RXF0命中，15表示RXF15命中。

如果与多个验收过滤器匹配，则FILHIT位将对所匹配的过滤器中编号最小的过滤器的二进制值进行编码。换言之，如果过滤器RXF2和过滤器RXF4满足匹配条件，FILHIT位将装入RXF2的值。这实际上是赋予了编号较小的验收过滤器较高的优先级。报文按过滤器编号升序与过滤器进行比较。

只有CAN模块处于配置模式时才能修改屏蔽器和过滤器寄存器。

图34-3： 报文验收屏蔽器和过滤器操作



34.8 波特率设置

给定CAN总线上的所有节点必须具有相同的标称比特率。CAN协议使用不归零（NRZ）编码，这种编码方式不对数据流内的时钟进行编码。因此，接收时钟必须通过接收节点恢复并与发送器时钟同步。

由于振荡器和传输时间可能因节点而异，因此接收器必须将某种类型的锁相环（Phase Lock Loop，PLL）与时钟传输边沿同步，以便同步和维持接收器时钟。由于数据采用NRZ编码，因此必须包括位填充以确保至少每六个位时间出现一个边沿，以便维持数字锁相环（Digital Phase Lock Loop，DPLL）同步。

CAN模块的位时序使用DPLL实现，DPLL配置为与传入数据同步并且可为发送的数据提供标称时序。DPLL将每个位时间分成多个时间段，每个时间段由称为*时间份额*（TQ）的最小时间段组成。

在位时间帧内执行的总线时序功能（例如与本地振荡器的同步、网络传输延时补偿和采样点定位）由DPLL的可编程位时序逻辑定义。

CAN总线上的所有器件必须使用相同的比特率。但是，不要求所有器件具有相同的主振荡器时钟频率。如果各个器件的时钟频率有所不同，必须通过适当设置波特率预分频比和每个时间段中的时间份额数来调整比特率。

“标称比特率”是指理想情况下（发送器和振荡器均为理想状态且未发生重新同步）每秒发送的比特数。标称比特率定义为最高1 Mbps。

“标称位时间”定义为：

公式34-1： 标称位时间

$$T_{BIT} = 1/\text{标称比特率}$$

标称位时间可被视为分成多个互不重叠的单独时间段。这些时间段（图34-4）包括：

- 同步段（Sync_Seg）
- 传播时间段（Prop_Seg）
- 相位缓冲段1（Phase_Seg1）
- 相位缓冲段2（Phase_Seg2）

这些时间段（以及标称位时间）又由整数个称为时间份额（即 TQ）的时间单位组成（见图 34-4）。根据定义，标称位时间可编程为8 TQ（最小值）到25 TQ（最大值）范围内的值。同样根据定义，最小标称位时间为1 µs，对应于最大速率1 Mbps。实际持续时间由以下关系给出：

公式34-2： 标称位时间持续时间

$$\text{标称位时间} = T_Q * (\text{Sync\_Seg} + \text{Prop\_Seg} + \text{Phase\_Seg1} + \text{Phase\_Seg2})$$

时间份额是由振荡器周期得出的固定时间单位。它还可通过可编程波特率预分频比（从1到64的整数值）和用于生成时钟的固定分频比2来定义。数学公式为：

公式34-3： 时间份额

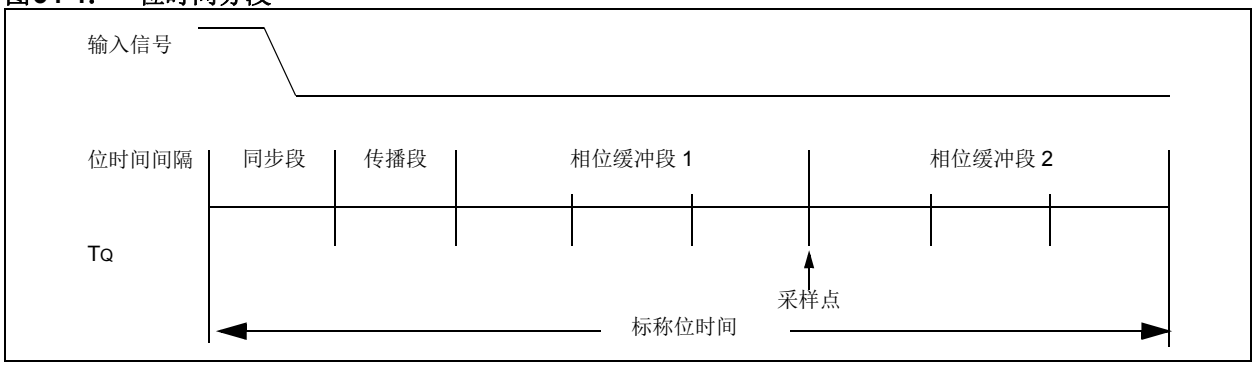
$$T_Q（\mu s）= (2 * (BRP + 1))/F_{OSC}（MHz）$$

或

$$T_Q（\mu s）= (2 * (BRP + 1)) * T_{OSC}（\mu s）$$

其中，FOSC是时钟频率，TOSC是对应的振荡器周期，BRP是0至63的整数（用BRGCON1<5:0>的二进制值表示）。以上公式是指单片机使用的有效时钟频率。例如，如果在HS模式下使用10 MHz晶振，则FOSC = 10 MHz，TOSC = 100 ns。如果在HS-PLL模式下也使用10 MHz晶振，则有效频率FOSC = 40 MHz，TOSC = 25 ns。

图34-4： 位时间分段



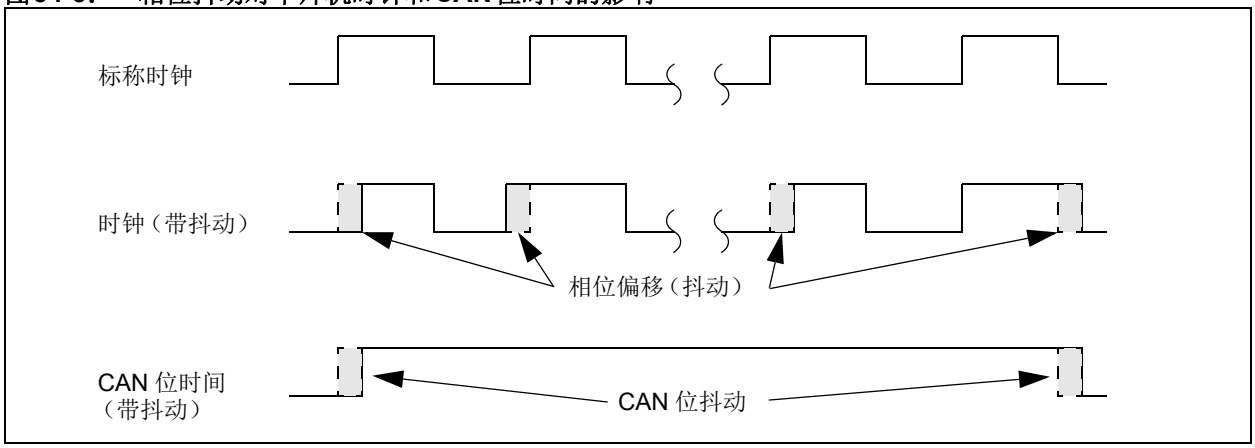
34.8.1 基于HS-PLL的振荡器中的外部时钟、内部时钟和可测量抖动

从PLL电路产生的单片机时钟频率受抖动（也定义为相位抖动或相位偏移）的影响。对于PIC18增强型单片机，Microchip将相位抖动（p_{jitter}）指定为2%（高斯分布，在三个标准差内，请参见表45-9中的参数PLL04），总抖动（t_{jitter}）为2 * p_{jitter}。

CAN协议使用位填充技术，在给定极性的5个位之后插入1个极性相反的位。这样，便可在未进行重新同步（补偿抖动或相位误差）的情况下发送10个位。

鉴于所增加抖动误差的随机性，最终由抖动引起的总误差往往会随时间而自行抵消。对于10位的周期，必须仅添加两个抖动间隔来校正抖动引起的误差：一个间隔添加到10位周期开始处，另一个间隔添加到结束处。总体影响如图34-5所示。

图34-5： 相位抖动对单片机时钟和CAN位时间的影响



考虑上述因素后，可以将抖动和总频率误差定义为如下关系：

公式34-4： 抖动和总频率误差

$$\Delta f = \frac{T_{抖动}}{10 \times \mathrm{NBT}} = \frac{2 \times P_{\mathrm{jitter}}}{10 \times \mathrm{NBT}}$$

其中，抖动用时间项表示，NBT是标称位时间。

例如，假设CAN比特率为125 kbps，可得出NBT为8 µs。对于由4x PLL生成的16 MHz时钟，此时钟频率下的抖动为：

公式34-5： 4x PLL生成的16 MHz时钟下的抖动：

$$2\% \times \frac{1}{16\ \mathrm{MHz}} = \frac{0.02}{16 \times 10^{6}} = 1.25\ \mathrm{ns}$$

得到的频率误差为：

公式34-6： 得到的频率误差：

$$\frac{2 \times (1.25 \times 10^{-9})}{10 \times (8 \times 10^{-6})} = 3.125 \times 10^{-5} = 0.0031\%$$

表34-2列出了PLL生成的时钟与抖动引起的频率误差（测量的抖动引起的误差为2%，高斯分布，在三个标准差内）之间的关系，频率误差以标称时钟频率的百分比表示。

这明显小于晶振的预期漂移（通常规定为100 ppm，即0.01%）。如果我们为振荡器漂移添加抖动，则总频率漂移为0.0132%。表34-3列出了常见时钟频率和比特率下的总振荡器频率误差（包括漂移和抖动）。

表34-2： PLL生成的各种时钟速度下由抖动引起的频率误差

PLL输出	P_{jitter}	T_{jitter}	各种标称位时间（比特率）下的频率误差			
			8 µs（125 kbps）	4 µs（250 kbps）	2 µs（500 kbps）	1 µs（1 Mbps）
40 MHz	0.5 ns	1 ns	0.00125%	0.00250%	0.005%	0.01%
24 MHz	0.83 ns	1.67 ns	0.00209%	0.00418%	0.008%	0.017%
16 MHz	1.25 ns	2.5 ns	0.00313%	0.00625%	0.013%	0.025%

表34-3： PLL生成的各种时钟速度下的总频率误差（100 PPM振荡器漂移，包括抖动引起的误差）

标称PLL输出	各种标称位时间（比特率）下的频率误差			
	8 µs（125 kbps）	4 µs（250 kbps）	2 µs（500 kbps）	1 µs（1 Mbps）
40 MHz	0.01125%	0.01250%	0.015%	0.02%
24 MHz	0.01209%	0.01418%	0.018%	0.027%
16 MHz	0.01313%	0.01625%	0.023%	0.035%

34.8.2 时间份额

如上所述，时间份额是由振荡器周期和波特率预分频比得出的一个固定时间单位。例34-1给出了它与TBIT和标称比特率之间的关系。

例34-1： 计算TQ、标称比特率和标称位时间

TQ（µs）= (2 * (BRP + 1))/FOSC（MHz）

TBIT（µs）= TQ（µs）* 每个位间隔的TQ数

标称比特率（bps）= 1/TBIT

该频率（FOSC）指的是使用的有效频率。例如，如果将10 MHz外部信号与PLL一起使用，则有效频率为4 x 10 MHz，即40 MHz。

情形1：

FOSC = 16 MHz，BRP<5:0> = 00h且
标称位时间 = 8 TQ：

TQ = (2 * 1)/16 = 0.125 µs（125 ns）

TBIT = 8 * 0.125 = 1 µs（10^{-6}s）

标称比特率 = $1/10^{-6} = 10^6$ bps（1 Mbps）

情形2：

FOSC = 20 MHz，BRP<5:0> = 01h且
标称位时间 = 8 TQ：

TQ = (2 * 2)/20 = 0.2 µs（200 ns）

TBIT = 8 * 0.2 = 1.6 µs（$1.6 * 10^{-6}$s）

标称比特率 = $1/1.6 * 10^{-6}$s = 625,000 bps（625 kbps）

情形3：

FOSC = 25 MHz，BRP<5:0> = 3Fh且
标称位时间 = 25 TQ：

TQ = (2 * 64)/25 = 5.12 µs

TBIT = 25 * 5.12 = 128 µs（$1.28 * 10^{-4}$s）

标称比特率 = $1/1.28 * 10^{-4}$ = 7813 bps（7.8 kbps）

对于不同节点中的振荡器频率，必须适当协调以便在整个系统范围内提供指定的标称位时间。这意味着所有振荡器的TOSC必须为TQ的整数倍。另应注意的是，尽管TQ的数量可编程为4到25之间的值，但可用的最小值是8 TQ。如果位时间长度小于8 TQ，则无法保证正常工作。

34.8.3 同步段

这部分位时间用于同步总线上的各个CAN节点。预计输入信号的边沿会在同步段期间出现。持续时间为1 TQ。

34.8.4 传播段

这部分位时间用于补偿网络中的物理延时。这些延时包括总线上的信号传播时间和节点的内部延时。通过设置PRSEG<2:0>位可将传播段的长度编程为1 TQ到8 TQ。

34.8.5 相位缓冲段

相位缓冲段用于将接收位的采样点置于标称位时间内的最佳位置。采样点位于相位缓冲段1和相位缓冲段2之间。通过重新同步过程可延长或缩短这些相位缓冲段。相位缓冲段1的末端决定了采样点在位时间内的位置。相位缓冲段1的持续时间可编程为1 TQ到8 TQ。相位缓冲段2在下一次发送数据转换之前提供了一段延时，其持续时间也可编程为1 TQ到8 TQ。不过，受IPT要求限制，相位缓冲段2的实际最小长度为2 TQ，或者可以定义为等于相位缓冲段1或信息处理时间（Information Processing Time，IPT）中的较大者。采样点应尽可能晚出现（即大约在位时间的80%处出现）。

34.8.6 采样点

采样点是读取总线电平并确定接收位值的时间点。采样点出现在相位缓冲段1的末端。如果位时间很长，包含多个TQ，则可以指定在采样点多次采样总线。通过择多判定从三个值中确定接收位的值。三次采样中有一次发生在采样点，有两次发生在采样点之前，各次采样的间隔时间为TQ/2。

34.8.7 信息处理时间

信息处理时间（IPT）是从采样点开始的时间段，保留用于计算后续位的电平。CAN规范将此时间定义为小于或等于2 TQ。ECAN模块将此时间定义为2 TQ。因此，相位缓冲段2的长度必须至少为2 TQ。

34.8.8 时钟选择

CIOCON寄存器的CLKSEL位允许在两个CAN输入时钟之间进行选择。当CLKSEL = 0（默认值）时，CAN时钟（上面公式中的FOSC）将与系统时钟相同。当CLKSEL = 1时，CAN时钟将是由FEXTOSC配置位选择的时钟，与系统时钟无关。这样，便可由PLL在64 MHz（16 MHz HS晶振+4xPLL）下为器件内核提供时钟，同时仍由基础16 MHz HS晶振（不使用PLL）为CAN提供时钟。

注：	如果CLKSEL = 1，系统时钟必须大于或等于FEXTOSC选择的时钟。如果系统时钟慢于CAN时钟，则将导致意外行为。信息处理时间（IPT）是从采样点开始的时间段，保留用于计算后续位的电平。CAN规范将此时间定义为小于或等于2 TQ。CAN模块将此时间定义为2 TQ。因此，相位缓冲段2的长度必须至少为2 TQ。

34.9 同步

为补偿总线上各节点振荡器频率之间的相移，每个CAN控制器都必须能够与传入信号的相关边沿同步。当检测到传输数据的边沿时，逻辑会将边沿的位置与预期时间（Sync_Seg）进行比较。之后，电路将根据需要调整相位缓冲段1和相位缓冲段2的值。有两种同步机制。

34.9.1 硬同步

只有在总线空闲条件下出现从隐性到显性的边沿转换时（表示开始传输报文），才会进行硬同步。硬同步后，位时间计数器将以Sync_Seg重新启动。硬同步会将出现的边沿强制置于重新启动位时间的同步段内。受同步规则限制，如果出现硬同步，则在该位时间内不会重新同步。

34.9.2 重新同步

重新同步后，相位缓冲段1会延长，相位缓冲段2会缩短。相位缓冲段延长或缩短的量有上限，具体由同步跳转宽度（Synchronization Jump Width，SJW）决定。相位缓冲段1将加上SJW的值（见图34-6），相位缓冲段2将减去SJW的值（见图34-7）。SJW可编程为1 TQ到4 TQ。

时钟信息只能从隐性到显性的转换中获取。同值连续位的数量有一个固定的最大值，这一属性可确保与帧内的比特流重新同步。

边沿的相位误差由边沿相对于Sync_Seg的位置给出，测量单位为TQ。相位误差以TQ为单位进行定义，具体如下：

- 如果边沿位于Sync_Seg内，则e = 0。
- 如果边沿位于采样点之前，则e > 0。
- 如果边沿位于前一位的采样点之后，则e < 0。

如果相位误差小于或等于同步跳转宽度的编程值，则重新同步的效果与硬同步的效果相同。

如果相位误差大于同步跳转宽度并且相位误差为正，则相位缓冲段1延长的量等于同步跳转宽度。

如果相位误差大于重新同步跳转宽度并且相位误差为负，则相位缓冲段2缩短的量等于同步跳转宽度。

34.9.3 同步规则

- 一个位时间内仅允许进行一次同步。
- 只有在前一个采样点检测到的值（先前读取的总线值）与紧接在边沿之后的总线值不同时，才会使用边沿进行同步。
- 满足规则1和规则2的所有其他隐性到显性边沿将用于重新同步，但对于具有正相位误差的隐性到显性边沿，传输显性位的节点不会执行重新同步。

图34-6： 延长位周期（相位缓冲段1加上SJW）

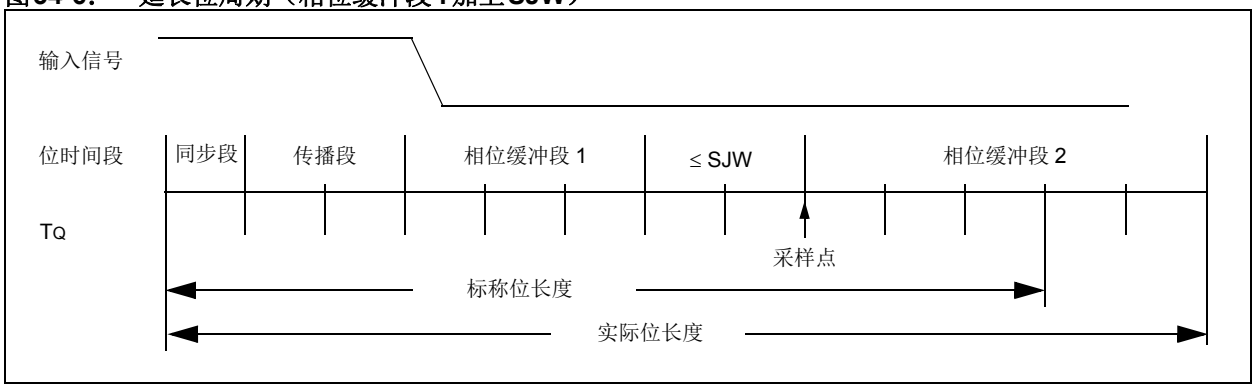
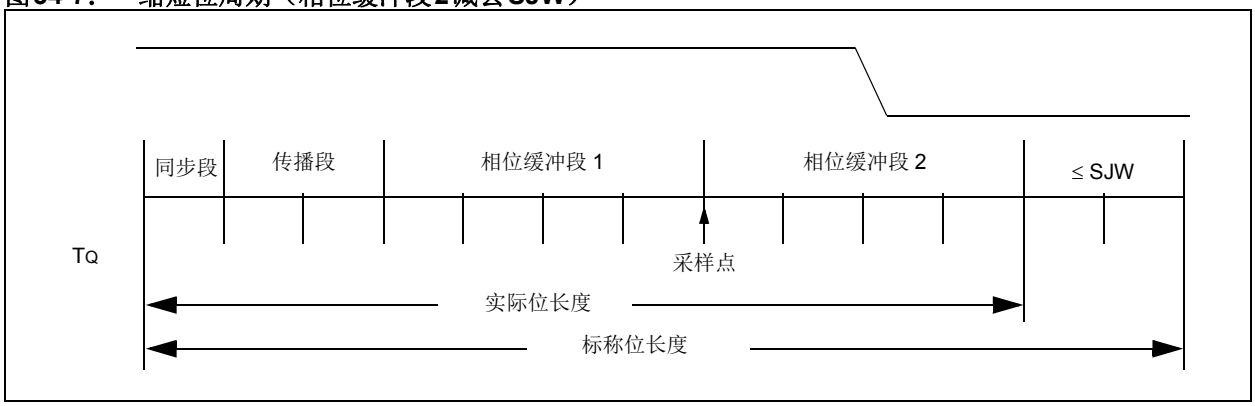


图34-7： 缩短位周期（相位缓冲段2减去SJW）



34.10 编程时间段

关于编程时间段的两点要求：

- Prop_Seg + Phase_Seg 1 ≥ Phase_Seg 2
- Phase_Seg 2 ≥ 同步跳转宽度

例如，假设需要125 kHz CAN波特率，使用20 MHz的FOSC。当TOSC为50 ns时，如果波特率预分频值为04h，可得出TQ为500 ns。要获得125 kHz的标称比特率，标称位时间必须为8 μs或16 TQ。

如果Sync_Seg使用1 TQ，Prop_Seg使用2 TQ，相位缓冲段1使用7 TQ，则采样点位于转换后的10 TQ处。这将使相位缓冲段2为6 TQ。

根据上述规则，同步跳转宽度最大为4 TQ。但是，通常只有当不同节点的时钟生成不准确或不稳定时（例如使用陶瓷谐振器），才需要较大的SJW。通常，SJW为1便足够。

34.11 振荡器容差

一般来说，位时序要求允许陶瓷谐振器用于传输速率最高为125 kbps的应用。要满足CAN协议的整个总线速度范围，则需要使用石英振荡器。关于振荡器容差要求，请参见ISO11898-1。

34.12 位时序配置寄存器

波特率控制寄存器（BRGCON1、BRGCON2和BRGCON3）控制CAN总线接口的位时序。只有CAN模块处于配置模式时才能修改这些寄存器。

34.12.1 BRGCON1

BRP位控制波特率预分频比。SJW<1:0>位选择同步跳转宽度（以TQ为单位表示）。

34.12.2 BRGCON2

PRSEG位设置传播段的长度（以TQ为单位表示）。SEG1PH位设置相位缓冲段1的长度（以TQ为单位表示）。SAM位控制RXCAN引脚被采样的次数。将该位设置为1会导致总线被采样三次：两次在采样点前的TQ/2处，一次在正常采样点处（即相位缓冲段1末端）。总线的值至少通过两次采样读取的值来确定。如果SAM位设置为0，则RXCAN引脚仅在采样点采样一次。SEG2PHTS位控制相位缓冲段2长度的确定方式。如果该位设置为1，则相位缓冲段2的长度由BRGCON3的SEG2PH位确定。如果SEG2PHTS位设置为0，则相位缓冲段2的长度是相位缓冲段1和信息处理时间（ECAN模块固定为2 TQ）中的较大者。

34.12.3 BRGCON3

如果SEG2PHTS位设置为1，则PHSEG2<2:0>位设置相位缓冲段2的长度（以TQ为单位表示）。如果SEG2PHTS位设置为0，则PHSEG2<2:0>位不起作用。

34.13 错误检测

CAN协议提供了精密的错误检测机制。可检测出以下错误。

34.13.1 CRC错误

发送器通过循环冗余校验（CRC）计算特殊校验位以确定从帧开始到数据字段结束的位序列。CRC序列在CRC字段中发送。接收节点也采用相同的公式计算CRC序列，并与接收的序列进行比较。如果检测到不匹配，即发生了CRC错误并将生成错误帧。报文将重新发送。

34.13.2 应答错误

在报文的应答字段中，发送器将检查应答间隙位（已作为隐性位发送）是否含有显性位。如果没有，则表明没有任何其他节点正确接收到此帧。此时会发生应答错误并生成错误帧，而报文则必须重新发送。

34.13.3 格式错误

如果节点检测到四个段（包括帧结束（End-of-Frame，EOF）、帧间空间、应答分隔符或CRC分隔符）中的一个包含显性位，则会发生格式错误并生成错误帧。报文将重新发送。

34.13.4 位错误

在监视实际总线电平并将其与刚发送的位进行比较时，如果发送器发送的是显性位但检测到隐性位，或者发送器发送的是隐性位但检测到显性位，则发生位错误。如果发送的是隐性位，而在仲裁字段和应答间隙期间检测到的是显性位，由于正在进行正常仲裁，因此不会生成位错误。

34.13.5 填充位错误

如果在帧开始（SOF）和CRC分隔符之间检测到六个极性相同的连续位，则表明违反了位填充规则。此时会发生填充位错误并生成错误帧。报文将重新发送。

34.13.6 错误状态

检测到的错误将通过错误帧通知给所有其他节点。应尽快中止出错报文的发送并重新进行发送。此外，根据内部错误计数器的值，每个CAN节点都处于三种错误状态之一：“主动错误”、“被动错误”或“总线关闭”。主动错误状态为常态，该状态下总线节点可不受任何限制地发送报文和激活错误帧（由显性位组成）。被动错误状态下，报文和被动错误帧（由隐性位组成）的发送可能受限。总线关闭状态可使节点暂时无法参与总线通信。该状态下无法收发报文。

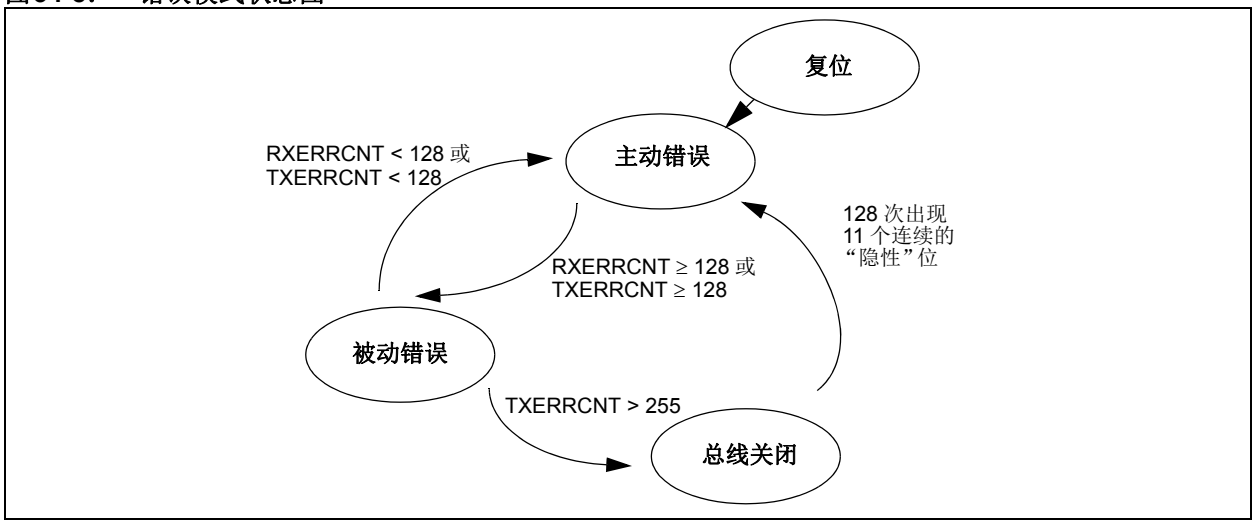
34.13.7 错误模式和错误计数器

CAN模块包含两个错误计数器：接收错误计数器（RXERRCNT）和发送错误计数器（TXERRCNT）。可通过MCU读取两个计数器的值。这些计数器将根据CAN总线规范增减计数值。

当两个错误计数器的计数值都小于被动错误状态限值128时，CAN模块处于主动错误状态。只要有一个错误计数器的计数值大于或等于128，就会进入被动错误状态。如果发送错误计数器的计数值大于或等于总线关闭限值256，则会进入总线关闭状态。在总线关闭恢复序列完成之前，器件将一直保持该状态。总线关闭恢复序列由重复出现128次的11个连续隐性位组成（见图34-8）。请注意，进入总线关闭状态后，如果总线空闲时间持续128 x 11位的时间，则CAN模块将恢复为主动错误状态，无需MCU进行任何干预。如果不希望自动恢复，可通过错误中断服务程序解决。MCU可通过COMSTAT寄存器读取CAN模块的当前错误模式。

此外，只要有一个错误计数器的计数值大于或等于错误警告限值96，错误状态警告标志位就会置1。如果两个错误计数器的计数值都小于错误警告限值，则EWARN复位。

图34-8： 错误模式状态图



34.14 CAN中断

该模块有多个中断源。其中每个中断均可以单独允许或禁止。PIR5寄存器包含中断标志。PIE5寄存器包含8个主中断的允许功能。CANSTAT寄存器中有一组特殊只读位（ICODE位）可与跳转表结合使用，从而高效处理中断。

除模式1和模式2下的错误中断和缓冲区中断外，所有中断均有一个中断源。任何错误中断源都可将错误中断标志置1。可通过读取通信状态寄存器COMSTAT来确定错误中断源。在模式1和模式2下，有两个中断允许/禁止位和标志位：一个用于所有发送缓冲区，另一个用于所有接收缓冲区。

中断可以分为两类：接收中断和发送中断。

与接收相关的中断包括：

- 接收中断
- 唤醒中断
- 接收器上溢中断
- 接收器警告中断
- 接收器被动错误中断

与发送相关的中断包括：

- 发送中断
- 发送器警告中断
- 发送器被动错误中断
- 总线关闭中断

34.14.1 中断代码位

为简化用户固件中的中断处理过程，ECAN模块需对一组特殊位进行编码。在模式0下，这些位是CANSTAT寄存器中的ICODE<3:1>。在模式1和模式2下，这些位是CANSTAT寄存器中的EICODE<4:0>。中断存在内部优先级，中断优先级越高，所分配的值越小。当优先级最高的中断条件被清除后，ICODE位将反映下一个优先级最高的中断（如果存在）的代码（见表34-4）。请注意，ICODE位仅反映那些相关的中断允许位已置1的中断源。

在模式2下，当发生接收报文中断时，EICODE位将始终由10000组成。用户固件可以使用FIFO指针位来实际访问下一个可用的缓冲区。

34.14.2 发送中断

允许发送中断时，如果相关的发送缓冲区为空并且随时可装入新报文，则将产生中断。在模式0下，三个专用发送缓冲区中的每一个都有单独的中断允许/禁止位和标志位。TXBnIF位将置1，以指示中断源。中断由MCU清除，同时TXBnIF位会复位为0。在模式1和模式2下，所有发送缓冲区共用一个中断允许/禁止位和一个标志位。在模式1和模式2下，PIE5中的TXBnIE和PIR5中的TXBnIF指示发送缓冲区何时完成其报文的发送。在模式1和模式2下，不使用2PIR5中的TXBnIF、PIE5中的TXBnIE和IPR5中的TXBnIP。将TXBnIE和B0IE寄存器位置1或清零可允许或禁止各发送缓冲区中断。发生共用中断时，用户固件必须轮询所有发送缓冲区的TXREQ位以检测中断源。

34.14.3 接收中断

允许接收中断时，如果报文已成功接收并装入相关接收缓冲区，则将产生中断。接收到帧结束（EOF）字段后会立即激活该中断。

在模式0下，RXBnIF位置1可指示中断源。中断由MCU清除，同时RXBnIF位会复位为0。

在模式1和模式2下，所有接收缓冲区共用PIE5中的RXBnIE、PIR5中的RXBnIF和IPR5中的RXBnIP。各接收缓冲区中断可由TXBnIE和BIE0寄存器控制。在模式1下，发生共用接收中断时，用户固件必须轮询每个接收缓冲区的RXFUL位以检测中断源。在模式2下，接收中断指示新报文装入FIFO中。可以使用FIFO指针位FP读取FIFO。

表34-4： ICODE<2:0>的值

ICODE <2:0>	中断	布尔表达式
000	无	$\overline{ERR}$•$\overline{WAK}$•$\overline{TX0}$•$\overline{TX1}$•$\overline{TX2}$•$\overline{RX0}$•$\overline{RX1}$
001	错误	ERR
010	TXB2	$\overline{ERR}$•$\overline{TX0}$•$\overline{TX1}$•TX2
011	TXB1	$\overline{ERR}$•$\overline{TX0}$•TX1
100	TXB0	$\overline{ERR}$•TX0
101	RXB1	$\overline{ERR}$•$\overline{TX0}$•$\overline{TX1}$•$\overline{TX2}$•$\overline{RX0}$•RX1
110	RXB0	$\overline{ERR}$•$\overline{TX0}$•$\overline{TX1}$•$\overline{TX2}$•RX0
111	唤醒中断	$\overline{ERR}$•$\overline{TX0}$•$\overline{TX1}$•$\overline{TX2}$•$\overline{RX0}$•$\overline{RX1}$•WAK

图注：

ERR = ERRIF * ERRIE　　RX0 = RXB0IF * RXB0IE
TX0 = TXB0IF * TXB0IE　　RX1 = RXB1IF * RXB1IE
TX1 = TXB1IF * TXB1IE　　WAK = WAKIF * WAKIE
TX2 = TXB2IF * TXB2IE

34.14.4 报文错误中断

收发报文期间发生错误时，报文错误标志IRXIF将置1，如果此时IRXIE位置1，则将产生中断。该中断配合仅监听模式使用时用于加快波特率确定的速度。

34.14.5 总线活动唤醒中断

当ECAN模块处于休眠模式并且允许总线活动唤醒中断时，如果在CAN总线上检测到活动，则将产生中断并且WAKIF位将置1。该中断会使MCU退出休眠模式。中断由MCU复位，同时WAKIF位会清零。

34.14.6 错误中断

允许CAN模块错误中断时（PIE5中的ERRIE），如果发生溢出或发送器/接收器的错误状态已改变，则将产生中断。COMSTAT中的错误标志将指示以下状况之一。

34.14.6.1 接收器溢出

当MAB组合好收到的有效报文（该报文符合验收过滤器条件），而与该过滤器相关的接收缓冲区尚无法装入新报文时，将发生溢出。COMSTAT寄存器中相关的RXBnOVFL位将置1，以指示溢出条件。此位必须由MCU清零。在模式0下，RXB0和RXB1具有单独的溢出位。在模式1和模式2下，这两个接收缓冲区共用一个位来指示是否溢出，而具体是哪个缓冲区溢出则需要逐个确认。

34.14.6.2 接收器警告

接收错误计数器的计数值已达MCU警告限值96。这将由COMSTAT寄存器的RXWARN位指示。

34.14.6.3 发送器警告

发送错误计数器的计数值已达MCU警告限值96。这将由COMSTAT寄存器的TXWARN位指示。

34.14.6.4 接收器总线被动

当器件已进入被动错误状态时会发生这种情况，因为接收错误计数器的计数值大于或等于128。这将由COMSTAT寄存器的RXBP位指示。

34.14.6.5 发送器总线被动

当器件已进入被动错误状态时会发生这种情况，因为发送错误计数器的计数值大于或等于128。这将由COMSTAT寄存器的TXBP位指示。

34.14.6.6 总线关闭

发送错误计数器的计数值超出了255，器件已进入总线关闭状态。这将由COMSTAT寄存器的TXBO位指示。

34.15 CAN模块寄存器

注： 并非所有CAN寄存器均可在快速操作存储区中访问。

CAN模块有许多相关的控制和数据寄存器。为方便起见，分以下几个部分对相关寄存器进行说明：

- 控制和状态寄存器
- 专用发送缓冲区寄存器
- 专用接收缓冲区寄存器
- 可编程TX/RX和自动RTR缓冲区
- 波特率控制寄存器
- I/O控制寄存器
- 中断状态和控制寄存器

以下各节详细介绍了各个寄存器及其用法。

34.15.1 CAN控制和状态寄存器

本节所述的寄存器控制CAN模块的整体操作并显示其工作状态。

寄存器34-1： CANCON：CAN控制寄存器

模式0	R/W-1	R/W-0	R/W-0	R/S-0	R/W-0	R/W-0	R/W-0	U-0
	REQOP2	REQOP1	REQOP0	ABAT	WIN2	WIN1	WIN0	—

模式1	R/W-1	R/W-0	R/W-0	R/S-0	U0	U-0	U-0	U-0
	REQOP2	REQOP1	REQOP0	ABAT	—	—	—	—

模式2	R/W-1	R/W-0	R/W-0	R/S-0	R-0	R-0	R-0	R-0
	REQOP2	REQOP1	REQOP0	ABAT	FP3	FP2	FP1	FP0
	bit 7							bit 0

图注：	S = 可置1位		
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **REQOP<2:0>：**请求CAN工作模式位
1xx = 请求配置模式
011 = 请求仅监听模式
010 = 请求环回模式
001 = 禁止/休眠模式
000 = 请求正常模式

bit 4 **ABAT：**中止所有待处理发送位
1 = 中止所有待处理发送（所有发送缓冲区中）(1)
0 = 正常发送

bit 3-1 模式0：
WIN<2:0>：窗口地址位
这些位选择将哪个CAN缓冲区切换到快速操作存储区。这允许从任何数据存储区访问缓冲区寄存器。当某个帧导致中断后，可以将ICODE<3:0>位复制到WIN<2:0>位以选择正确的缓冲区。有关代码示例，请参见例34-3。
111 = 接收缓冲区0
110 = 接收缓冲区0
101 = 接收缓冲区1
100 = 发送缓冲区0
011 = 发送缓冲区1
010 = 发送缓冲区2
001 = 接收缓冲区0
000 = 接收缓冲区0

bit 0 模式0：
未实现：读为0

bit 4-0 模式1：
未实现：读为0
模式2：
FP<3:0>：FIFO读指针位
这些位指向要读取的报文缓冲区。
0000 = 接收报文缓冲区0
0001 = 接收报文缓冲区1
0010 = 接收报文缓冲区2
0011 = 接收报文缓冲区3
0100 = 接收报文缓冲区4
0101 = 接收报文缓冲区5
0110 = 接收报文缓冲区6
0111 = 接收报文缓冲区7
1000:1111 保留

注 1： 该位将在所有发送中止时清零。

寄存器34-2： CANSTAT：CAN状态寄存器

模式0	R-1	R-0	R-0	R-0	R-0	R-0	R-0	U-0
	OPMODE2(1)	OPMODE1(1)	OPMODE0(1)	—	ICODE2	ICODE1	ICODE0	—

模式1和模式2	R-1	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	OPMODE2(1)	OPMODE1(1)	OPMODE0(1)	EICODE4	EICODE3	EICODE2	EICODE1	EICODE0
	bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **OPMODE<2:0>：**工作模式状态位(1)
111 = 保留
110 = 保留
101 = 保留
100 = 配置模式
011 = 仅监听模式
010 = 环回模式
001 = 禁止/休眠模式
000 = 正常模式

bit 4 模式0：
未实现：读为0

bit 3-1和4-0 模式0：
ICODE<2:0>：中断代码位
发生中断时，优先编码的中断值将出现在这些位中。此代码指示中断源。通过将ICODE<3:1>复制到ICODE<3:1>（模式0）或将EICODE<4:0>复制到EWIN<4:0>（模式1和模式2），可以选择正确的缓冲区映射到快速操作存储区。有关代码示例，请参见例34-3。为方便说明，下表列出了全部5个位。

	模式0	模式1	模式2
不产生中断	00000	00000	00000
CAN总线错误中断	00010	00010	00010
TXB2中断	00100	00100	00100
TXB1中断	00110	00110	00110
TXB0中断	01000	01000	01000
RXB1中断	01010	10001	-----
RXB0中断	01100	10000	10000
唤醒中断	01110	01110	01110
RXB0中断	-----	10000	10000
RXB1中断	-----	10001	10000
RX/TX B0中断	-----	10010	10010(2)
RX/TX B1中断	-----	10011	10011(2)
RX/TX B2中断	-----	10100	10100(2)
RX/TX B3中断	-----	10101	10101(2)
RX/TX B4中断	-----	10110	10110(2)
RX/TX B5中断	-----	10111	10111(2)

bit 0 模式0：
未实现：读为0

bit 4-0 模式1和模式2：
EICODE<4:0>：中断代码位
请参见上面的ICODE<3:1>。

注 1： 为了最大程度降低功耗和/或能够在出现CAN总线活动时唤醒，请在将器件置于休眠模式之前将CAN模块切换到禁止/休眠模式。
2： 如果缓冲区配置为接收器，则中断时EICODE位将包含10000。

例34-2： 切换到配置模式

```
    ; Request Configuration mode.
    MOVLW   B'10000000'                 ; Set to Configuration Mode.
    MOVWF   CANCON
    ; A request to switch to Configuration mode may not be immediately honored.
    ; Module will wait for CAN bus to be idle before switching to Configuration Mode.
    ; Request for other modes such as Loopback, Disable etc. may be honored immediately.
    ; It is always good practice to wait and verify before continuing.
ConfigWait:
    MOVF    CANSTAT, W                  ; Read current mode state.
    ANDLW   B'10000000'                 ; Interested in OPMODE bits only.
    TSTFSZ  WREG                        ; Is it Configuration mode yet?
    BRA     ConfigWait                  ; No. Continue to wait...
    ; Module is in Configuration mode now.
    ; Modify configuration registers as required.
    ; Switch back to Normal mode to be able to communicate.
```

例34-3： 访问TX/RX缓冲区时中断服务程序中WIN和ICODE位的应用

```
    ; Save application required context.
    ; Poll interrupt flags and determine source of interrupt
    ; This was found to be CAN interrupt
    ; TempCANCON and TempCANSTAT are variables defined in Access Bank low
    MOVFF   CANCON, TempCANCON          ; Save CANCON.WIN bits
                                        ; This is required to prevent CANCON
                                        ; from corrupting CAN buffer access
                                        ; in-progress while this interrupt
                                        ; occurred
    MOVFF   CANSTAT, TempCANSTAT        ; Save CANSTAT register
                                        ; This is required to make sure that
                                        ; we use same CANSTAT value rather
                                        ; than one changed by another CAN
                                        ; interrupt.
    MOVF    TempCANSTAT, W              ; Retrieve ICODE bits
    ANDLW   B'00001110'
    ADDWF   PCL, F                      ; Perform computed GOTO
                                        ; to corresponding interrupt cause
    BRA     NoInterrupt                 ; 000 = No interrupt
    BRA     ErrorInterrupt              ; 001 = Error interrupt
    BRA     TXB2Interrupt               ; 010 = TXB2 interrupt
    BRA     TXB1Interrupt               ; 011 = TXB1 interrupt
    BRA     TXB0Interrupt               ; 100 = TXB0 interrupt
    BRA     RXB1Interrupt               ; 101 = RXB1 interrupt
    BRA     RXB0Interrupt               ; 110 = RXB0 interrupt
                                        ; 111 = Wake-up on interrupt
WakeupInterrupt
    BCF     PIR3, WAKIF                 ; Clear the interrupt flag
    ;
    ; User code to handle wake-up procedure
    ;
    ;
    ; Continue checking for other interrupt source or return from here
    …
NoInterrupt
    …                                   ; PC should never vector here. User may
                                        ; place a trap such as infinite loop or pin/port
                                        ; indication to catch this error.
```

例34-3： 访问TX/RX缓冲区时中断服务程序中WIN和ICODE位的应用（续）

```
ErrorInterrupt
    BCF     PIR3, ERRIF                 ; Clear the interrupt flag
    …                                   ; Handle error.
    RETFIE
TXB2Interrupt
    BCF     PIR3, TXB2IF                ; Clear the interrupt flag
    GOTO    AccessBuffer
TXB1Interrupt
    BCF     PIR3, TXB1IF                ; Clear the interrupt flag
    GOTO    AccessBuffer
TXB0Interrupt
    BCF     PIR3, TXB0IF                ; Clear the interrupt flag
    GOTO    AccessBuffer
RXB1Interrupt
    BCF     PIR3, RXB1IF                ; Clear the interrupt flag
    GOTO    Accessbuffer
RXB0Interrupt
    BCF     PIR3, RXB0IF                ; Clear the interrupt flag
    GOTO    AccessBuffer
AccessBuffer                            ; This is either TX or RX interrupt
    ; Copy CANSTAT.ICODE bits to CANCON.WIN bits
    MOVF    TempCANCON, W               ; Clear CANCON.WIN bits before copying
                                        ; new ones.
    ANDLW   B’11110001’                 ; Use previously saved CANCON value to
                                        ; make sure same value.
    MOVWF   TempCANCON                  ; Copy masked value back to TempCANCON
    MOVF    TempCANSTAT, W              ; Retrieve ICODE bits
    ANDLW   B’00001110’                 ; Use previously saved CANSTAT value
                                        ; to make sure same value.
    IORWF   TempCANCON                  ; Copy ICODE bits to WIN bits.
    MOVFF   TempCANCON, CANCON          ; Copy the result to actual CANCON
    ; Access current buffer…
    ; User code
    ; Restore CANCON.WIN bits
    MOVF    CANCON, W                   ; Preserve current non WIN bits
    ANDLW   B’11110001’
    IORWF   TempCANCON                  ; Restore original WIN bits
    ; Do not need to restore CANSTAT - it is read-only register.
    ; Return from interrupt or check for another module interrupt source
```

寄存器34-3： ECANCON：增强型CAN控制寄存器

R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
MDSEL1(1)	MDSEL0(1)	FIFOWM(2)	EWIN4	EWIN3	EWIN2	EWIN1	EWIN0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **MDSEL<1:0>：**模式选择位(1)
00 = 传统模式（默认为模式0）
01 = 增强型传统模式（模式1）
10 = 增强型FIFO模式（模式2）
11 = 保留

bit 5 **FIFOWM：**FIFO高水印位(2)
1 = 当有一个接收缓冲区仍为空时，将导致FIFO中断
0 = 当有四个接收缓冲区仍为空时，将导致FIFO中断(3)

bit 4-0 **EWIN<4:0>：**增强型窗口地址位
这些位将一组16个存储区的CAN SFR映射到快速操作存取区地址0F60-0F6Dh。要映射的确切寄存器组由这些位的二进制值确定。
模式0：
未实现：读为0
模式1和模式2：
00000 = 验收过滤器0、1和2以及BRGCON2和3
00001 = 验收过滤器3、4和5以及BRGCON1和CIOCON
00010 = 验收过滤器屏蔽、错误和中断控制
00011 = 发送缓冲区0
00100 = 发送缓冲区1
00101 = 发送缓冲区2
00110 = 验收过滤器6、7和8
00111 = 验收过滤器9、10和11
01000 = 验收过滤器12、13和14
01001 = 验收过滤器15
01010-01110 = 保留
01111 = RXINT0和RXINT1
10000 = 接收缓冲区0
10001 = 接收缓冲区1
10010 = TX/RX缓冲区0
10011 = TX/RX缓冲区1
10100 = TX/RX缓冲区2
10101 = TX/RX缓冲区3
10110 = TX/RX缓冲区4
10111 = TX/RX缓冲区5
11000-11111 = 保留

注 1： 只能在配置模式下更改这些位。有关如何切换到配置模式的信息，请参见寄存器34-1。
2： 该位仅在模式2下使用。
3： 如果FIFO配置为包含四个或以下的缓冲区，则将触发FIFO中断。

寄存器34-4： COMSTAT：通信状态寄存器

模式0	R/C-0	R/C-0	R-0	R-0	R-0	R-0	R-0	R-0
	RXB0OVFL	RXB1OVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN

模式1	R/C-0	R/C-0	R-0	R-0	R-0	R-0	R-0	R-0
	—	RXBnOVFL	TXB0	TXBP	RXBP	TXWARN	RXWARN	EWARN

模式2	R/C-0	R/C-0	R-0	R-0	R-0	R-0	R-0	R-0
	$\overline{\text{FIFOEMPTY}}$	RXBnOVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN
	bit 7							bit 0

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 模式0：
RXB0OVFL：接收缓冲区0溢出位
1 = 接收缓冲区0溢出
0 = 接收缓冲区0未溢出
模式1：
未实现：读为0
模式2：
FIFOEMPTY：FIFO非空位
1 = 接收FIFO非空
0 = 接收FIFO为空

bit 6 模式0：
RXB1OVFL：接收缓冲区1溢出位
1 = 接收缓冲区1溢出
0 = 接收缓冲区1未溢出
模式1和模式2：
RXBnOVFL：接收缓冲区n溢出位
1 = 接收缓冲区n溢出
0 = 接收缓冲区n未溢出

bit 5 **TXBO**：发送器总线关闭位
1 = 发送错误计数器 > 255
0 = 发送错误计数器 ≤ 255

bit 4 **TXBP**：发送器总线被动位
1 = 发送错误计数器 > 127
0 = 发送错误计数器 ≤ 127

bit 3 **RXBP**：接收器总线被动位
1 = 接收错误计数器 > 127
0 = 接收错误计数器 ≤ 127

bit 2 **TXWARN**：发送器警告位
1 = 发送错误计数器 > 95
0 = 发送错误计数器 ≤ 95

bit 1 **RXWARN**：接收器警告位
1 = 127 ≥ 接收错误计数器 > 95
0 = 接收错误计数器 ≤ 95

bit 0 **EWARN**：错误警告位
该位是RXWARN和TXWARN位的标志。
1 = RXWARN或TXWARN位置1
0 = RXWARN或TXWARN位均未置1

34.15.2 专用CAN发送缓冲区寄存器

本节介绍专用CAN发送缓冲区寄存器及其相关的控制寄存器。

寄存器34-5： TXBnCON：发送缓冲区n控制寄存器[0 ≤ n ≤ 2]

RC-0	R-0	R-0	R-0	R/W-0	U-0	R/W-0	R/W-0
TXBIF	TXABT(1)	TXLARB(1)	TXERR(1)	TXREQ(2)	—	TXPRI1(3)	TXPRI0(3)
bit 7							bit 0

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **TXBIF：**发送缓冲区中断标志位
1 = 发送缓冲区已完成报文的发送，可以重载
0 = 发送缓冲区尚未完成报文的发送

bit 6 **TXABT：**发送中止状态位(1)
1 = 报文中止
0 = 报文未中止

bit 5 **TXLARB：**发送仲裁失败状态位(1)
1 = 报文在发送过程中仲裁失败
0 = 报文在发送过程中没有仲裁失败

bit 4 **TXERR：**发送错误检测状态位(1)
1 = 发送报文时发生总线错误
0 = 发送报文时未发生总线错误

bit 3 **TXREQ：**发送请求状态位(2)
1 = 请求发送报文；清零TXABT、TXLARB和TXERR位
0 = 成功发送报文时自动清零

bit 2 **未实现：**读为0

bit 1-0 **TXPRI<1:0>：**发送极性位(3)
11 = 优先级3（最高优先级）
10 = 优先级2
01 = 优先级1
00 = 优先级0（最低优先级）

注 1： 当TXREQ置1时自动清零该位。
2： 当TXREQ置1时，发送缓冲区寄存器保持只读状态。在该位置1时用软件清零该位将请求中止报文。
3： 这些位定义了发送缓冲区的发送顺序。它们不会改变CAN报文标识符。

寄存器34-6： TXBnSIDH：发送缓冲区n标准标识符寄存器高字节[0 ≤ n ≤ 2]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **SID<10:3>：**标准标识符位（EXIDE（TXBnSIDL<3>）= 0时）
扩展标识符位EID<28:21>（EXIDE = 1时）

寄存器34-7： TXBnSIDL：发送缓冲区n标准标识符寄存器低字节[0 ≤ n ≤ 2]

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **SID<2:0>：**标准标识符位（EXIDE（TXBnSIDL<3>）= 0时）
扩展标识符位EID<20:18>（EXIDE = 1时）
bit 4 **未实现：**读为0
bit 3 **EXIDE：**扩展标识符使能位
1 = 报文将发送扩展ID，SID<10:0>变为EID<28:18>
0 = 报文将发送标准ID，EID<17:0>将被忽略
bit 2 **未实现：**读为0
bit 1-0 **EID<17:16>：**扩展标识符位

寄存器34-8： TXBnEIDH：发送缓冲区n扩展标识符寄存器高字节[0 ≤ n ≤ 2]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<15:8>：**扩展标识符位（发送标准标识符报文时不使用）

寄存器34-9： TXBnEIDL：发送缓冲区n扩展标识符寄存器低字节[0 ≤ n ≤ 2]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<7:0>**：扩展标识符位（发送标准标识符报文时不使用）

寄存器34-10： TXBnDm：发送缓冲区n数据字段字节m寄存器[0 ≤ n ≤ 2，0 ≤ m ≤ 7]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
TXBnDm7	TXBnDm6	TXBnDm5	TXBnDm4	TXBnDm3	TXBnDm2	TXBnDm1	TXBnDm0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **TXBnDm<7:0>**：发送缓冲区n数据字段字节m位（其中0 ≤ n < 3且0 ≤ m < 8）
每个发送缓冲区都有一组寄存器。例如，发送缓冲区0有7个寄存器：TXB0D0至TXB0D7。

寄存器34-11： TXBnDLC：发送缓冲区n数据长度码寄存器[0 ≤ n ≤ 2]

U-0	R/W-x	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **未实现：**读为0

bit 6 **TXRTR：**发送远程帧发送请求位
1 = 发送的报文将使TXRTR位置1
0 = 发送的报文将使TXRTR位清零

bit 5-4 **未实现：**读为0

bit 3-0 **DLC<3:0>：**数据长度码位
1111 = 保留
1110 = 保留
1101 = 保留
1100 = 保留
1011 = 保留
1010 = 保留
1001 = 保留
1000 = 数据长度 = 8个字节
0111 = 数据长度 = 7个字节
0110 = 数据长度 = 6个字节
0101 = 数据长度 = 5个字节
0100 = 数据长度 = 4个字节
0011 = 数据长度 = 3个字节
0010 = 数据长度 = 2个字节
0001 = 数据长度 = 1个字节
0000 = 数据长度 = 0个字节

寄存器34-12： TXERRCNT：发送错误计数寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **TEC<7:0>：**发送错误计数器位
该寄存器包含一个由错误发生率导出的值。当错误计数溢出时，将出现总线关闭状态。当总线出现128次11个连续的隐性位时，计数器值被清零。

例34-4： 使用分区方法发送CAN报文

```
; Need to transmit Standard Identifier message 123h using TXB0 buffer.
; To successfully transmit, CAN module must be either in Normal or Loopback mode.
; TXB0 buffer is not in access bank. And since we want banked method, we need to make sure
; that correct bank is selected.
BANKSEL TXB0CON                  ; One BANKSEL in beginning will make sure that we are
                                 ; in correct bank for rest of the buffer access.
; Now load transmit data into TXB0 buffer.
MOVLW   MY_DATA_BYTE1            ; Load first data byte into buffer
MOVWF   TXB0D0                   ; Compiler will automatically set "BANKED" bit
; Load rest of data bytes - up to 8 bytes into TXB0 buffer.
...
; Load message identifier
MOVLW   60H                      ; Load SID2:SID0, EXIDE = 0
MOVWF   TXB0SIDL
MOVLW   24H                      ; Load SID10:SID3
MOVWF   TXB0SIDH
; No need to load TXB0EIDL:TXB0EIDH, as we are transmitting Standard Identifier Message only.

; Now that all data bytes are loaded, mark it for transmission.
MOVLW   B'00001000'              ; Normal priority; Request transmission
MOVWF   TXB0CON

; If required, wait for message to get transmitted
BTFSC   TXB0CON, TXREQ           ; Is it transmitted?
BRA     $-2                      ; No. Continue to wait...

; Message is transmitted.
```

例34-5： 使用WIN位发送CAN报文

```
; Need to transmit Standard Identifier message 123h using TXB0 buffer.
; To successfully transmit, CAN module must be either in Normal or Loopback mode.
; TXB0 buffer is not in access bank. Use WIN bits to map it to RXB0 area.
MOVF    CANCON, W                    ; WIN bits are in lower 4 bits only. Read CANCON
                                     ; register to preserve all other bits. If operation
                                     ; mode is already known, there is no need to preserve
                                     ; other bits.
ANDLW   B'11110000'                  ; Clear WIN bits.
IORLW   B'00001000'                  ; Select Transmit Buffer 0
MOVWF   CANCON                       ; Apply the changes.
; Now TXB0 is mapped in place of RXB0. All future access to RXB0 registers will actually
; yield TXB0 register values.

; Load transmit data into TXB0 buffer.
MOVLW   MY_DATA_BYTE1                ; Load first data byte into buffer
MOVWF   RXB0D0                       ; Access TXB0D0 via RXB0D0 address.
; Load rest of the data bytes - up to 8 bytes into "TXB0" buffer using RXB0 registers.
...
; Load message identifier
MOVLW   60H                          ; Load SID2:SID0, EXIDE = 0
MOVWF   RXB0SIDL
MOVLW   24H                          ; Load SID10:SID3
MOVWF   RXB0SIDH
; No need to load RXB0EIDL:RXB0EIDH, as we are transmitting Standard Identifier Message only.

; Now that all data bytes are loaded, mark it for transmission.
MOVLW   B'00001000'                  ; Normal priority; Request transmission
MOVWF   RXB0CON

; If required, wait for message to get transmitted
BTFSC   RXB0CON, TXREQ               ; Is it transmitted?
BRA     $-2                          ; No. Continue to wait...

; Message is transmitted.
; If required, reset the WIN bits to default state.
```

34.15.3 专用CAN接收缓冲区寄存器

本节介绍专用CAN接收缓冲区寄存器及其相关的控制寄存器。

寄存器34-13： RXB0CON：接收缓冲区0控制寄存器

模式0	R/C-0	R/W-0	R/W-0	U-0	R-0	R/W-0	R-0	R-0
	RXFUL(1)	RXM1	RXM0	—	RXRTRRO	RXB0DBEN	JTOFF(2)	FILHIT0

模式1和模式2	R/C-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
	RXFUL(1)	RXM1	RTRRO	FILHITF4	FILHIT3	FILHIT2	FILHIT1	FILHIT0
	bit 7							bit 0

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **RXFUL：**接收满状态位(1)
1 = 接收缓冲区已被收到的报文装满
0 = 接收缓冲区仍可继续接收新报文

bit 6,6-5 模式0：
RXM<1:0>：接收缓冲区模式位1（与RXM0组合形成RXM<1:0>位，请参见bit 5）
11 = 接收所有报文（包括有错误的报文）；过滤条件被忽略
10 = 仅接收具有扩展标识符的有效报文；RXFnSIDL中的EXIDEN必须为1
01 = 仅接收具有标准标识符的有效报文；RXFnSIDL中的EXIDEN必须为0
00 = 根据RXFnSIDL寄存器中的EXIDEN位接收所有有效报文

模式1和模式2：
RXM1：接收缓冲区模式位1
1 = 接收所有报文（包括有错误的报文）；验收过滤器被忽略
0 = 根据验收过滤器接收所有有效报文

bit 5 模式0：
RXM0：接收缓冲区模式位0（与RXM1组合形成RXM<1:0>位，请参见bit 6）

模式1和模式2：
RTRRO：所接收报文的远程发送请求位（只读）
1 = 接收到远程发送请求
0 = 未接收到远程发送请求

bit 4 模式0：
未实现：读为0

模式1和模式2：
FILHIT<4:0>：过滤器命中位4
该位与其他位组合形成过滤器验收位<4:0>。

bit 3 模式0：
RXRTRRO：所接收报文的远程发送请求位（只读）
1 = 接收到远程发送请求
0 = 未接收到远程发送请求

模式1和模式2：
FILHIT<4:0>：过滤器命中位3
该位与其他位组合形成过滤器验收位<4:0>。

注 1： 该位在接收到报文时由CAN模块置1，读取缓冲区后必须由软件清零。只要RXFUL置1，就不会装入新报文，缓冲区将被视为已满。当RXFUL标志清零后，可以清零PIR5位RXB0IF。如果RXB0IF清零，但RXFUL未清零，则RXB0IF再次置1。

2： 该位允许RXB0CON和RXB1CON使用相同的过滤器跳转表。

寄存器34-13： RXB0CON：接收缓冲区0控制寄存器（续）

bit 2

模式0：

RB0DBEN：接收缓冲区0双缓冲区使能位

1 = 接收缓冲区0溢出会写入接收缓冲区1

0 = 接收缓冲区0溢出不会写入接收缓冲区1

模式1和模式2：

FILHIT<4:0>：过滤器命中位2

该位与其他位组合形成过滤器验收位<4:0>。

bit 1

模式0：

JTOFF：跳转表偏移量位（RXB0DBEN的只读副本）(2)

1 = 允许跳转表偏移量介于6和7之间

0 = 允许跳转表偏移量介于1和0之间

模式1和模式2：

FILHIT<4:0>：过滤器命中位1

该位与其他位组合形成过滤器验收位<4:0>。

bit 0

模式0：

FILHIT0：过滤器命中位0

该位指示哪个验收过滤器允许接收的报文存储到接收缓冲区0中。

1 = 验收过滤器1（RXF1）

0 = 验收过滤器0（RXF0）

模式1和模式2：

FILHIT<4:0>：过滤器命中位0

该位与FILHIT<4:1>组合来指示哪个验收过滤器允许接收的报文存储到此接收缓冲区。

01111 = 验收过滤器15（RXF15）

01110 = 验收过滤器14（RXF14）

...

00000 = 验收过滤器0（RXF0）

注 1： 该位在接收到报文时由CAN模块置1，读取缓冲区后必须由软件清零。只要RXFUL置1，就不会装入新报文，缓冲区将被视为已满。当RXFUL标志清零后，可以清零PIR5位RXB0IF。如果RXB0IF清零，但RXFUL未清零，则RXB0IF再次置1。

2： 该位允许RXB0CON和RXB1CON使用相同的过滤器跳转表。

寄存器34-14： RXB1CON：接收缓冲区1控制寄存器

模式0	R/C-0	R/W-0	R/W-0	U-0	R-0	R/W-0	R-0	R-0
	RXFUL(1)	RXM1	RXM0	—	RXRTRRO	FILHIT2	FILHIT1	FILHIT0

模式1和模式2	R/C-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
	RXFUL(1)	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0
	bit 7							bit 0

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **RXFUL：**接收满状态位(1)
1 = 接收缓冲区已被收到的报文装满
0 = 接收缓冲区仍可继续接收新报文

bit 6-5, 6 模式0：
RXM<1:0>：接收缓冲区模式位1（与RXM0组合形成RXM<1:0>位，请参见bit 5）
11 = 接收所有报文（包括有错误的报文）；过滤条件被忽略
10 = 仅接收具有扩展标识符的有效报文；RXFnSIDL中的EXIDEN必须为1
01 = 仅接收具有标准标识符的有效报文；RXFnSIDL中的EXIDEN必须为0
00 = 根据RXFnSIDL寄存器中的EXIDEN位接收所有有效报文
模式1和模式2：
RXM1：接收缓冲区模式位
1 = 接收所有报文（包括有错误的报文）；验收过滤器被忽略
0 = 根据验收过滤器接收所有有效报文

bit 5 模式0：
RXM<1:0>：接收缓冲区模式位0（与RXM1组合形成RXM<1:0>位，请参见bit 6）
模式1和模式2：
RTRRO：所接收报文的远程发送请求位（只读）
1 = 接收到远程发送请求
0 = 未接收到远程发送请求

bit 4 模式0：
FILHIT**4：**过滤器命中位4
模式1和模式2：
FILHIT<4:0>：过滤器命中位4
该位与其他位组合形成过滤器验收位<4:0>。

bit 3 模式0：
RXRTRRO：所接收报文的远程发送请求位（只读）
1 = 接收到远程发送请求
0 = 未接收到远程发送请求
模式1和模式2：
FILHIT<4:0>：过滤器命中位3
该位与其他位组合形成过滤器验收位<4:0>。

注 1： 该位在接收到报文时由CAN模块置1，读取缓冲区后必须由软件清零。只要RXFUL置1，就不会装入新报文，缓冲区将被视为已满。

寄存器34-14： RXB1CON：接收缓冲区1控制寄存器（续）

bit 2-0

模式0：

FILHIT<2:0>：过滤器命中位

这些位指示哪个验收过滤器允许接收的最后一条报文存储到接收缓冲区1中。

111 = 保留
110 = 保留
101 = 验收过滤器5（RXF5）
100 = 验收过滤器4（RXF4）
011 = 验收过滤器3（RXF3）
010 = 验收过滤器2（RXF2）
001 = 验收过滤器1（RXF1），仅适用于RXB0DBEN位置1时
000 = 验收过滤器0（RXF0），仅适用于RXB0DBEN位置1时

模式1和模式2：

FILHIT<4:0>：过滤器命中位<2:0>

这些位与FILHIT<4:3>组合来指示哪个验收过滤器允许接收的报文存储到此接收缓冲区。

01111 = 验收过滤器15（RXF15）
01110 = 验收过滤器14（RXF14）
...
00000 = 验收过滤器0（RXF0）

注 1： 该位在接收到报文时由CAN模块置1，读取缓冲区后必须由软件清零。只要RXFUL置1，就不会装入新报文，缓冲区将被视为已满。

寄存器34-15： RXBnSIDH：接收缓冲区n标准标识符寄存器高字节[0 ≤ n ≤ 1]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0

SID<10:3>：标准标识符位（EXID（RXBnSIDL<3>）= 0时）

扩展标识符位EID<28:21>（EXID = 1时）

寄存器34-16： RXBnSIDL：接收缓冲区n标准标识符寄存器低字节[0 ≤ n ≤ 1]

R-x	R-x	R-x	R-x	R-x	U-0	R-x	R-x
SID2	SID1	SID0	SRR	EXID	—	EID17	EID16
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **SID<2:0>：**标准标识符位（EXID = 0时）
扩展标识符位EID<20:18>（EXID = 1时）

bit 4 **SRR：**替代远程请求位

bit 3 **EXID：**扩展标识符位
1 = 接收到的报文是扩展数据帧，SID<10:0>是EID<28:18>
0 = 接收到的报文是标准数据帧

bit 2 **未实现：**读为0

bit 1-0 **EID<17:16>：**扩展标识符位

寄存器34-17： RXBnEIDH：接收缓冲区n扩展标识符寄存器高字节[0 ≤ n ≤ 1]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<15:8>：**扩展标识符位

寄存器34-18： RXBnEIDL：接收缓冲区n扩展标识符寄存器低字节[0 ≤ n ≤ 1]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<7:0>：**扩展标识符位

寄存器34-19： RXBnDLC：接收缓冲区n数据长度码寄存器[0 ≤ n ≤ 1]

U-0	R-x	R-x	R-x	R-x	R-x	R-x	R-x
—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **未实现：**读为0

bit 6 **RXRTR：**接收器远程发送请求位
1 = 远程发送请求
0 = 无远程发送请求

bit 5 **RB1：**保留位1
根据CAN规范保留，读为0。

bit 4 **RB0：**保留位0
根据CAN规范保留，读为0。

bit 3-0 **DLC<3:0>：**数据长度码位
1111 = 无效
1110 = 无效
1101 = 无效
1100 = 无效
1011 = 无效
1010 = 无效
1001 = 无效
1000 = 数据长度 = 8个字节
0111 = 数据长度 = 7个字节
0110 = 数据长度 = 6个字节
0101 = 数据长度 = 5个字节
0100 = 数据长度 = 4个字节
0011 = 数据长度 = 3个字节
0010 = 数据长度 = 2个字节
0001 = 数据长度 = 1个字节
0000 = 数据长度 = 0个字节

寄存器34-20： RXBnDm：接收缓冲区n数据字段字节m寄存器[0 ≤ n ≤ 1, 0 ≤ m ≤ 7]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
RXBnDm7	RXBnDm6	RXBnDm5	RXBnDm4	RXBnDm3	RXBnDm2	RXBnDm1	RXBnDm0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **RXBnDm<7:0>：**接收缓冲区n数据字段字节m位（其中0 ≤ n < 1且0 < m < 7）
每个接收缓冲区都有一组寄存器。例如，接收缓冲区0有8个寄存器：RXB0D0至RXB0D7。

寄存器34-21： RXERRCNT：接收错误计数寄存器

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **REC<7:0>**：接收错误计数器位

该寄存器包含CAN规范定义的接收错误值。当RXERRCNT > 127时，模块将进入被动错误状态。RXERRCNT无法将模块置于“总线关闭”状态。

例34-6： 读取CAN报文

```
    ; Need to read a pending message from RXB0 buffer.
    ; To receive any message, filter, mask and RXM1:RXM0 bits in RXB0CON registers must be
    ; programmed correctly.
    ;
    ; Make sure that there is a message pending in RXB0.
    BTFSS   RXB0CON, RXFUL                ; Does RXB0 contain a message?
    BRA     NoMessage                     ; No. Handle this situation...
    ; We have verified that a message is pending in RXB0 buffer.
    ; If this buffer can receive both Standard or Extended Identifier messages,
    ; identify type of message received.
    BTFSS   RXB0SIDL, EXID                ; Is this Extended Identifier?
    BRA     StandardMessage               ; No. This is Standard Identifier message.
                                          ; Yes. This is Extended Identifier message.
    ; Read all 29-bits of Extended Identifier message.
    ...
    ; Now read all data bytes
    MOVFF   RXB0DO, MY_DATA_BYTE1
    ...
    ; Once entire message is read, mark the RXB0 that it is read and no longer FULL.
    BCF     RXB0CON, RXFUL                ; This will allow CAN Module to load new messages
                                          ; into this buffer.
    ...
```

34.15.3.1 可编程TX/RX和自动RTR缓冲区

ECAN模块包含6个可编程为发送或接收缓冲区的报文缓冲区。其中任一缓冲区也可编程为自动处理RTR报文。

注： 这些寄存器在模式0下不使用。

寄存器34-22： BnCON：接收模式下的TX/RX缓冲区n控制寄存器 [0 ≤ n ≤ 5，TXnEN（BSEL0<n>）= 0](1)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
RXFUL(2)	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **RXFUL：**接收满状态位(2)
1 = 接收缓冲区已被收到的报文装满
0 = 接收缓冲区仍可继续接收新报文

bit 6 **RXM1：**接收缓冲区模式位
1 = 接收所有报文（包括部分报文和无效报文）；验收过滤器被忽略
0 = 根据验收过滤器接收所有有效报文

bit 5 **RXRTRRO：**所接收报文的远程发送请求位（只读）
1 = 所接收报文是远程发送请求
0 = 所接收报文不是远程发送请求

bit 4-0 **FILHIT<4:0>：**过滤器命中位
这些位指示哪个验收过滤器允许接收的最后一条报文存储到此缓冲区中。
01111 = 验收过滤器15（RXF15）
01110 = 验收过滤器14（RXF14）
...
00001 = 验收过滤器1（RXF1）
00000 = 验收过滤器0（RXF0）

注 1： 这些寄存器仅适用于模式1和模式2。
2： 该位在接收到报文时由CAN模块置1，读取缓冲区后必须由软件清零。只要RXFUL置1，就不会装入新报文，缓冲区将被视为已满。

寄存器34-23：　BnCON：发送模式下的TX/RX缓冲区n控制寄存器 [0 ≤ n ≤ 5，TXnEN（BSEL0<n>）= 1](1)

R/W-0	R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TXBIF(3)	TXABT(3)	TXLARB(3)	TXERR(3)	TXREQ(2,4)	RTREN	TXPRI1(5)	TXPRI0(5)
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7　**TXBIF**：发送缓冲区中断标志位(3)
1 = 已成功发送报文
0 = 未发送报文

bit 6　**TXABT**：发送中止状态位(3)
1 = 报文中止
0 = 报文未中止

bit 5　**TXLARB**：发送仲裁失败状态位(3)
1 = 报文在发送过程中仲裁失败
0 = 报文在发送过程中没有仲裁失败

bit 4　**TXERR**：发送错误检测状态位(3)
1 = 发送报文时发生总线错误
0 = 发送报文时未发生总线错误

bit 3　**TXREQ**：发送请求状态位(2,4)
1 = 请求发送报文；清零TXABT、TXLARB和TXERR位
0 = 成功发送报文时自动清零

bit 2　**RTREN**：自动远程发送请求使能位
1 = 接收到远程发送请求时，TXREQ将自动置1
0 = 接收到远程发送请求时，TXREQ不受影响

bit 1-0　**TXPRI<1:0>**：发送极性位(5)
11 = 优先级3（最高优先级）
10 = 优先级2
01 = 优先级1
00 = 优先级0（最低优先级）

注 1：这些寄存器仅适用于模式1和模式2。
2：在该位置1时用软件清零该位将请求中止报文。
3：当TXREQ置1时自动清零该位。
4：当TXREQ置1或正在进行发送时，发送缓冲区寄存器保持只读状态。
5：这些位设置发送缓冲区寄存器的发送顺序。它们不会改变CAN报文标识符。

寄存器34-24： BnSIDH：接收模式下的TX/RX缓冲区n标准标识符寄存器高字节 [0 ≤ n ≤ 5，TXnEN（BSEL0<n>）= 0][(1)]

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **SID<10:3>**：标准标识符位（EXIDE（BnSIDL<3>）= 0时）
扩展标识符位EID<28:21>（EXIDE = 1时）

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-25： BnSIDH：发送模式下的TX/RX缓冲区n标准标识符寄存器高字节 [0 ≤ n ≤ 5，TXnEN（BSEL0<n>）= 1][(1)]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **SID<10:3>**：标准标识符位（EXIDE（BnSIDL<3>）= 0时）
扩展标识符位EID<28:21>（EXIDE = 1时）

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-26： BnSIDL：接收模式下的TX/RX缓冲区n标准标识符寄存器低字节 [0 ≤ n ≤ 5，TXnEN（BSEL0<n>）= 0](1)

R-x	R-x	R-x	R-x	R-x	U-0	R-x	R-x
SID2	SID1	SID0	SRR	EXIDE	—	EID17	EID16
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **SID<2:0>**：标准标识符位（EXID = 0时）
扩展标识符位EID<20:18>（EXID = 1时）

bit 4 **SRR**：替代远程发送请求位
该位在EXID = 1时始终为1，在EXID = 0时等于RXRTRRO（BnCON<5>）的值。

bit 3 **EXIDE**：扩展标识符使能位
1 = 接收到的报文是扩展标识符帧（SID<10:0>是EID<28:18>）
0 = 接收到的报文是标准标识符帧

bit 2 **未实现**：读为0

bit 1-0 **EID<17:16>**：扩展标识符位

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-27： BnSIDL：发送模式下的TX/RX缓冲区n标准标识符寄存器低字节 [0 ≤ n ≤ 5，TXnEN（BSEL0<n>）= 1](1)

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDE	—	EID17	EID16
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **SID<2:0>**：标准标识符位（EXIDE（TXBnSIDL<3>）= 0时）
扩展标识符位EID<20:18>（EXIDE = 1时）

bit 4 **未实现**：读为0

bit 3 **EXIDE**：扩展标识符使能位
1 = 报文将发送扩展ID，SID<10:0>位变为EID<28:18>
0 = 报文将发送标准ID，EID<17:0>将被忽略

bit 2 **未实现**：读为0

bit 1-0 **EID<17:16>**：扩展标识符位

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-28： BnEIDH：接收模式下的TX/RX缓冲区n扩展标识符寄存器高字节 [0 ≤ n ≤ 5，TXnEN（BSEL0<n>）= 0](1)

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<15:8>：**扩展标识符位

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-29： BnEIDH：发送模式下的TX/RX缓冲区n扩展标识符寄存器高字节 [0 ≤ n ≤ 5，TXnEN（BSEL0<n>）= 1](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<15:8>：**扩展标识符位

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-30： BnEIDL：接收模式下的TX/RX缓冲区n扩展标识符寄存器低字节 [0 ≤ n ≤ 5，TXnEN（BSEL<n>）= 0](1)

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<7:0>：**扩展标识符位

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-31： BnEIDL：接收模式下的TX/RX缓冲区n扩展标识符寄存器低字节 [0 ≤ n ≤ 5，TXnEN（BSEL<n>）= 1](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	FEID4	EID3	EID2	EID1	EID0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<7:0>：**扩展标识符位

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-32： BnDm：接收模式下的TX/RX缓冲区n数据字段字节m寄存器 [0 ≤ n ≤ 5，0 ≤ m ≤ 7，TXnEN（BSEL<n>）= 0](1)

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
BnDm7	BnDm6	BnDm5	BnDm4	BnDm3	BnDm2	BnDm1	BnDm0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **BnDm<7:0>：**接收缓冲区n数据字段字节m位（其中0 ≤ n < 3且0 < m < 8）
每个接收缓冲区都有一组寄存器。例如，接收缓冲区0有7个寄存器：B0D0至B0D7。

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-33： BnDm：发送模式下的TX/RX缓冲区n数据字段字节m寄存器 [0 ≤ n ≤ 5，0 ≤ m ≤ 7，TXnEN（BSEL<n>）= 1](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
BnDm7	BnDm6	BnDm5	BnDm4	BnDm3	BnDm2	BnDm1	BnDm0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **BnDm<7:0>：**发送缓冲区n数据字段字节m位（其中0 ≤ n < 3且0 < m < 8）
每个发送缓冲区都有一组寄存器。例如，发送缓冲区0有7个寄存器：TXB0D0至TXB0D7。

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-34： BnDLC：接收模式下的TX/RX缓冲区n数据长度码寄存器 [0 ≤ n ≤ 5，TXnEN（BSEL<n>）= 0][(1)]

U-0	R-x	R-x	R-x	R-x	R-x	R-x	R-x
—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **未实现：**读为0

bit 6 **RXRTR：**接收器远程发送请求位
1 = 这是远程发送请求
0 = 这不是远程发送请求

bit 5 **RB1：**保留位1
根据CAN规范保留，读为0。

bit 4 **RB0：**保留位0
根据CAN规范保留，读为0。

bit 3-0 **DLC<3:0>：**数据长度码位
1111 = 保留
1110 = 保留
1101 = 保留
1100 = 保留
1011 = 保留
1010 = 保留
1001 = 保留
1000 = 数据长度 = 8个字节
0111 = 数据长度 = 7个字节
0110 = 数据长度 = 6个字节
0101 = 数据长度 = 5个字节
0100 = 数据长度 = 4个字节
0011 = 数据长度 = 3个字节
0010 = 数据长度 = 2个字节
0001 = 数据长度 = 1个字节
0000 = 数据长度 = 0个字节

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-35： BnDLC：发送模式下的TX/RX缓冲区n数据长度码寄存器 [0 ≤ n ≤ 5，TXnEN（BSEL<n>）= 1][(1)]

U-0	R/W-x	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **未实现：**读为0

bit 6 **TXRTR：**发送器远程发送请求位
1 = 发送的报文将使RTR位置1
0 = 发送的报文将使RTR位清零

bit 5-4 **未实现：**读为0

bit 3-0 **DLC<3:0>：**数据长度码位
1111-1001 = 保留
1000 = 数据长度 = 8个字节
0111 = 数据长度 = 7个字节
0110 = 数据长度 = 6个字节
0101 = 数据长度 = 5个字节
0100 = 数据长度 = 4个字节
0011 = 数据长度 = 3个字节
0010 = 数据长度 = 2个字节
0001 = 数据长度 = 1个字节
0000 = 数据长度 = 0个字节

注 1： 这些寄存器仅适用于模式1和模式2。

寄存器34-36： BSEL0：缓冲区选择寄存器0[(1)]

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
B5TXEN	B4TXEN	B3TXEN	B2TXEN	B1TXEN	B0TXEN	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-2 **B<5:0>TXEN：**缓冲区5至缓冲区0发送使能位
1 = 缓冲区配置为发送模式
0 = 缓冲区配置为接收模式

bit 1-0 **未实现：**读为0

注 1： 这些寄存器仅适用于模式1和模式2。

34.15.3.2 报文验收过滤器和屏蔽器

本节介绍CAN接收缓冲区的报文验收过滤器和屏蔽器。

寄存器34-37： RXFnSIDH：接收验收过滤器n标准标识符过滤器寄存器高字节[0 ≤ n ≤ 15](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **SID<10:3>：**标准标识符过滤位（EXIDEN = 0时）
扩展标识符过滤位EID<28:21>（EXIDEN = 1时）

注 1： 寄存器RXF6SIDH:RXF15SIDH仅适用于模式1和模式2。

寄存器34-38： RXFnSIDL：接收验收过滤器n标准标识符过滤器寄存器低字节[0 ≤ n ≤ 15](1)

R/W-x	R/W-x	R/W-x	U-0	R/W-x	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDEN(2)	—	EID17	EID16
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **SID<2:0>：**标准标识符过滤位（EXIDEN = 0时）
扩展标识符过滤位EID<20:18>（EXIDEN = 1时）

bit 4 **未实现：**读为0

bit 3 **EXIDEN：**扩展标识符过滤器使能位(2)
1 = 过滤器只接受扩展ID报文
0 = 过滤器只接受标准ID报文

bit 2 **未实现：**读为0

bit 1-0 **EID<17:16>：**扩展标识符过滤位

注 1： 寄存器RXF6SIDL:RXF15SIDL仅适用于模式1和模式2。
2： 在模式0中，必须根据需要将该位置1/清零，与相应屏蔽器寄存器的值无关。

寄存器34-39： RXFnEIDH：接收验收过滤器n扩展标识符，寄存器高字节[0 ≤ n ≤ 15](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<15:8>**：扩展标识符过滤位

注 1： 寄存器RXF6EIDH:RXF15EIDH仅适用于模式1和模式2。

寄存器34-40： RXFnEIDL：接收验收过滤器n扩展标识符，寄存器低字节[0 ≤ n ≤ 15](1)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<7:0>**：扩展标识符过滤位

注 1： 寄存器RXF6EIDL:RXF15EIDL仅适用于模式1和模式2。

寄存器34-41： RXMnSIDH：接收验收屏蔽器n标准标识符屏蔽器，寄存器高字节[0 ≤ n ≤ 1]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **SID<10:3>**：标准标识符屏蔽位或扩展标识符屏蔽位（EID<28:21>）

寄存器34-42： RXMnSIDL：接收验收屏蔽器n标准标识符屏蔽器，寄存器低字节[0 ≤ n ≤ 1]

R/W-x	R/W-x	R/W-x	U-0	R/W-0	U-0	R/W-x	R/W-x
SID2	SID1	SID0	—	EXIDEN(1)	—	EID17	EID16
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **SID<2:0>**：标准标识符屏蔽位或扩展标识符屏蔽位（EID<20:18>）

bit 4 **未实现**：读为0

bit 3 模式0：
未实现：读为0
模式1和模式2：
EXIDEN：扩展标识符过滤器使能屏蔽位(1)
1 = 将接受RXFnSIDL中EXIDEN位选择的报文
0 = 将接受标准和扩展标识符报文

bit 2 **未实现**：读为0

bit 1-0 **EID<17:16>**：扩展标识符屏蔽位

注 1： 该位仅适用于模式1和模式2。

寄存器34-43： RXMnEIDH：接收验收屏蔽器n扩展标识符屏蔽器，寄存器高字节[0 ≤ n ≤ 1]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<15:8>**：扩展标识符屏蔽位

寄存器34-44： RXMnEIDL：接收验收屏蔽器n扩展标识符屏蔽器，寄存器低字节[0 ≤ n ≤ 1]

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **EID<7:0>：**扩展标识符屏蔽位

寄存器34-45： RXFCONn：接收过滤器控制寄存器n [0 ≤ n ≤ 1](1)

RXFCON0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	RXF7EN	RXF6EN	RXF5EN	RXF4EN	RXF3EN	RXF2EN	RXF1EN	RXF0EN

RXFCON1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	RXF15EN	RXF14EN	RXF13EN	RXF12EN	RXF11EN	RXF10EN	RXF9EN	RXF8EN
	bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **RXF<7:0>EN：**接收过滤器n使能位
0 = 禁止过滤器
1 = 使能过滤器

注 1： 该寄存器仅适用于模式1和模式2。

注： 寄存器34-46至寄存器34-51只能在配置模式下写入。

寄存器34-46： SDFLC：标准数据字节过滤器长度计数寄存器(1)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	FLC4	FLC3	FLC2	FLC1	FLC0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **未实现：**读为0

bit 4-0 **FLC<4:0>：**过滤器长度计数位

模式0：
未使用；强制为00000。

模式1和模式2：

00000-10010 = 0 标准数据字节过滤器可使用18位。实际使用的位数取决于正在接收的报文的DLC<3:0>位（如果配置为RX缓冲区，则为RXBnDLC<3:0>或BnDLC<3:0>）。

如果DLC<3:0>= 0000 没有位将与传入的数据位进行比较。

如果DLC<3:0>= 0001 最多8个数据位（即RXFnEID<7:0>，具体由FLC<2:0>确定）将与所传入报文的相应数量的数据位进行比较。

如果DLC<3:0>= 0010 最多16个数据位（即RXFnEID<15:0>，具体由FLC<3:0>确定）将与所传入报文的相应数量的数据位进行比较。

如果DLC<3:0>= 0011 最多18个数据位（即RXFnEID<17:0>，具体由FLC<4:0>确定）将与所传入报文的相应数量的数据位进行比较。

注 1： 该寄存器仅适用于模式1和模式2。

寄存器34-47： RXFBCONn：接收过滤器缓冲区控制寄存器n(1)

RXFBCON0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F1BP_3	F1BP_2	F1BP_1	F1BP_0	F0BP_3	F0BP_2	F0BP_1	F0BP_0
RXFBCON1	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1
	F3BP_3	F3BP_2	F3BP_1	F3BP_0	F2BP_3	F2BP_2	F2BP_1	F2BP_0
RXFBCON2	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-1
	F5BP_3	F5BP_2	F5BP_1	F5BP_0	F4BP_3	F4BP_2	F4BP_1	F4BP_0
RXFBCON3	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F7BP_3	F7BP_2	F7BP_1	F7BP_0	F6BP_3	F6BP_2	F6BP_1	F6BP_0
RXFBCON4	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F9BP_3	F9BP_2	F9BP_1	F9BP_0	F8BP_3	F8BP_2	F8BP_1	F8BP_0
RXFBCON5	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F11BP_3	F11BP_2	F11BP_1	F11BP_0	F10BP_3	F10BP_2	F10BP_1	F10BP_0
RXFBCON6	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F13BP_3	F13BP_2	F13BP_1	F13BP_0	F12BP_3	F12BP_2	F12BP_1	F12BP_0
RXFBCON7	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	F15BP_3	F15BP_2	F15BP_1	F15BP_0	F14BP_3	F14BP_2	F14BP_1	F14BP_0
	bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-0 **F<15:2>BP_<3:0>：** 过滤器n缓冲区指针半字节位

0000 = 过滤器n与RXB0相关联
0001 = 过滤器n与RXB1相关联
0010 = 过滤器n与B0相关联
0011 = 过滤器n与B1相关联
...
0111 = 过滤器n与B5相关联
1111-1000 = 保留

注 1： 该寄存器仅适用于模式1和模式2。

寄存器34-48： MSEL0：屏蔽器选择寄存器0(1)

R/W-0	R/W-1	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0
FIL3_1	FIL3_0	FIL2_1	FIL2_0	FIL1_1	FIL1_0	FIL0_1	FIL0_0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **FIL3_<1:0>：**过滤器3选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 5-4 **FIL2_<1:0>：**过滤器2选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 3-2 **FIL1_<1:0>：**过滤器1选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 1-0 **FIL0_<1:0>：**过滤器0选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

注 1： 该寄存器仅适用于模式1和模式2。

寄存器34-49： MSEL1：屏蔽器选择寄存器1(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
FIL7_1	FIL7_0	FIL6_1	FIL6_0	FIL5_1	FIL5_0	FIL4_1	FIL4_0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **FIL7_<1:0>：**过滤器7选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 5-4 **FIL6_<1:0>：**过滤器6选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 3-2 **FIL5_<1:0>：**过滤器5选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 1-0 **FIL4_<1:0>：**过滤器4选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

注 1： 该寄存器仅适用于模式1和模式2。

寄存器34-50： MSEL2：屏蔽器选择寄存器2(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FIL11_1	FIL11_0	FIL10_1	FIL10_0	FIL9_1	FIL9_0	FIL8_1	FIL8_0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **FIL11_<1:0>**：过滤器11选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 5-4 **FIL10_<1:0>**：过滤器10选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 3-2 **FIL9_<1:0>**：过滤器9选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 1-0 **FIL8_<1:0>**：过滤器8选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

注 1： 该寄存器仅适用于模式1和模式2。

寄存器34-51： MSEL3：屏蔽器选择寄存器3(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FIL15_1	FIL15_0	FIL14_1	FIL14_0	FIL13_1	FIL13_0	FIL12_1	FIL12_0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **FIL15_<1:0>：**过滤器15选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 5-4 **FIL14_<1:0>：**过滤器14选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 3-2 **FIL13_<1:0>：**过滤器13选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

bit 1-0 **FIL12_<1:0>：**过滤器12选择位1和0
11 = 无屏蔽器
10 = 过滤器15
01 = 验收屏蔽器1
00 = 验收屏蔽器0

注 1： 该寄存器仅适用于模式1和模式2。

34.15.4 CAN波特率寄存器

本节介绍CAN波特率寄存器。

注：	这些寄存器只能在配置模式下写入。

寄存器34-52： BRGCON1：波特率控制寄存器1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-6 **SJW<1:0>：**同步跳转宽度位
11 = 同步跳转宽度时间 = 4 x TQ
10 = 同步跳转宽度时间 = 3 x TQ
01 = 同步跳转宽度时间 = 2 x TQ
00 = 同步跳转宽度时间 = 1 x TQ

bit 5-0 **BRP<5:0>：**波特率预分频比位
111111 = TQ = (2 x 64)/FOSC
111110 = TQ = (2 x 63)/FOSC
:
:
000001 = TQ = (2 x 2)/FOSC
000000 = TQ = (2 x 1)/FOSC

寄存器34-53： BRGCON2：波特率控制寄存器2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SEG2PHTS	SAM	SEG1PH2	SEG1PH1	SEG1PH0	PRSEG2	PRSEG1	PRSEG0
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **SEG2PHTS**：相位缓冲段2时间选择位
1 = 可自由编程
0 = PHEG1的最大值与信息处理时间（IPT）中的较大值

bit 6 **SAM**：CAN总线线路采样位
1 = 在采样点之前对总线线路采样三次
0 = 在采样点对总线线路采样一次

bit 5-3 **SEG1PH<2:0>**：相位缓冲段1位
111 = 相位缓冲段1时间 = 8 x TQ
110 = 相位缓冲段1时间 = 7 x TQ
101 = 相位缓冲段1时间 = 6 x TQ
100 = 相位缓冲段1时间 = 5 x TQ
011 = 相位缓冲段1时间 = 4 x TQ
010 = 相位缓冲段1时间 = 3 x TQ
001 = 相位缓冲段1时间 = 2 x TQ
000 = 相位缓冲段1时间 = 1 x TQ

bit 2-0 **PRSEG<2:0>**：传播时间选择位
111 = 传播时间 = 8 x TQ
110 = 传播时间 = 7 x TQ
101 = 传播时间 = 6 x TQ
100 = 传播时间 = 5 x TQ
011 = 传播时间 = 4 x TQ
010 = 传播时间 = 3 x TQ
001 = 传播时间 = 2 x TQ
000 = 传播时间 = 1 x TQ

寄存器34-54： BRGCON3：波特率控制寄存器3

R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
WAKDIS	WAKFIL	—	—	—	SEG2PH2(1)	SEG2PH1(1)	SEG2PH0(1)
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **WAKDIS：**唤醒禁止位
1 = 禁止CAN总线活动唤醒功能
0 = 使能CAN总线活动唤醒功能

bit 6 **WAKFIL：**选择CAN总线线路过滤器来唤醒位
1 = 使用CAN总线线路过滤器来唤醒
0 = 不使用CAN总线线路过滤器来唤醒

bit 5-3 **未实现：**读为0

bit 2-0 **SEG2PH<2:0>：**相位缓冲段2时间选择位(1)
111 = 相位缓冲段2时间 = 8 x TQ
110 = 相位缓冲段2时间 = 7 x TQ
101 = 相位缓冲段2时间 = 6 x TQ
100 = 相位缓冲段2时间 = 5 x TQ
011 = 相位缓冲段2时间 = 4 x TQ
010 = 相位缓冲段2时间 = 3 x TQ
001 = 相位缓冲段2时间 = 2 x TQ
000 = 相位缓冲段2时间 = 1 x TQ

注 1： SEG2PHTS位（BRGCON2<7>）为0时忽略这些位。

34.15.5 CAN模块I/O控制寄存器

该寄存器控制与单片机其余部分相关的CAN模块I/O引脚的操作。

寄存器34-55： CIOCON：CAN I/O控制寄存器

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
TX1SRC	—	—	—	—	—	—	CLKSEL
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **TX1SRC**：CAN_tx1信号数据源位
1 = CAN_tx1信号将输出CAN时钟
0 = CAN_tx1信号将输出CANTX

bit 6-1 **未实现**：读为0

bit 0 **CLKSEL**：CAN时钟源选择位
1 = 无论系统时钟如何，CAN时钟都由FEXTOSC配置位字段选择的时钟提供(1)
0 = CAN时钟由系统时钟提供

注 1： 当CLKSEL = 1时，FEXTOSC提供的时钟必须小于或等于系统时钟。如果CAN时钟大于系统时钟，则会发生意外行为。

34.15.6 CAN中断寄存器

本节中的寄存器与**第9.0节“中断控制器”**所述的寄存器相同。为方便起见，这里将再次列出。

寄存器34-56： PIR5：外设中断请求（标志）寄存器5

模式0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF(1)	TXB0IF(1)	RXB1IF	RXB0IF

模式1和模式2	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	IRXIF	WAKIF	ERRIF	TXBnIF	TXB1IF(1)	TXB0IF(1)	RXBnIF	FIFOWMIF
	bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **IRXIF**：收到CAN总线错误报文中断标志位
1 = CAN总线上出现了无效报文
0 = CAN总线上未出现无效报文

bit 6 **WAKIF**：CAN总线活动唤醒中断标志位
1 = CAN总线上发生了活动
0 = CAN总线上未发生活动

bit 5 **ERRIF**：CAN模块错误中断标志位
1 = CAN模块发生了错误（有关多个错误源的信息，请参见**第34.14.6节“错误中断”**）
0 = CAN模块未发生错误

bit 4 当CAN处于模式0时：
TXB2IF：CAN发送缓冲区2中断标志位
1 = 发送缓冲区2已完成报文的发送，可以重载
0 = 发送缓冲区2未完成报文的发送
当CAN处于模式1或模式2时：
TXBnIF：任何发送缓冲区中断标志位
1 = 一个或多个发送缓冲区已完成报文的发送，可以重载
0 = 没有发送缓冲区准备重载

bit 3 **TXB1IF**：CAN发送缓冲区1中断标志位(1)
1 = 发送缓冲区1已完成报文的发送，可以重载
0 = 发送缓冲区1未完成报文的发送

bit 2 **TXB0IF**：CAN发送缓冲区0中断标志位(1)
1 = 发送缓冲区0已完成报文的发送，可以重载
0 = 发送缓冲区0未完成报文的发送

bit 1 当CAN处于模式0时：
RXB1IF：CAN接收缓冲区1中断标志位
1 = 接收缓冲区1已接收新报文
0 = 接收缓冲区1未接收新报文
当CAN处于模式1或模式2时：
RXBnIF：任何接收缓冲区中断标志位
1 = 一个或多个接收缓冲区已接收新报文
0 = 没有接收缓冲区已接收新报文

bit 0 当CAN处于模式0时：
RXB0IF：CAN接收缓冲区0中断标志位
1 = 接收缓冲区0已接收新报文
0 = 接收缓冲区0未接收新报文
当CAN处于模式1时：
未实现：读为0
当CAN处于模式2时：
FIFOWMIF：FIFO水印中断标志位
1 = 达到FIFO高水印
0 = 未达到FIFO高水印

注 1： 在CAN模式1和模式2下，这些位被强制为0。

寄存器34-57： PIE5：外设中断允许寄存器5

模式0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	IRXIE	WAKIE	ERRIE	TXB2IE	TXB1IE(1)	TXB0IE(1)	RXB1IE	RXB0IE

模式1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	IRXIE	WAKIE	ERRIE	TXBnIE	TXB1IE(1)	TXB0IE(1)	RXBnIE	FIFOWMIE
	bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **IRXIE**：收到CAN总线错误报文中断允许位
1 = 允许收到无效报文中断
0 = 禁止收到无效报文中断

bit 6 **WAKIE**：CAN总线活动唤醒中断允许位
1 = 允许总线活动唤醒中断
0 = 禁止总线活动唤醒中断

bit 5 **ERRIE**：CAN总线错误中断允许位
1 = 允许CAN模块错误中断
0 = 禁止CAN模块错误中断

bit 4 当CAN处于模式0时：
TXB2IE：CAN发送缓冲区2中断允许位
1 = 允许发送缓冲区2中断
0 = 禁止发送缓冲区2中断
当CAN处于模式1或模式2时：
TXBnIE：CAN发送缓冲区中断允许位
1 = 允许发送缓冲区中断；通过TXBIE和BIE0允许单独中断
0 = 禁止所有发送缓冲区中断

bit 3 **TXB1IE**：CAN发送缓冲区1中断允许位(1)
1 = 允许发送缓冲区1中断
0 = 禁止发送缓冲区1中断

bit 2 **TXB0IE**：CAN发送缓冲区0中断允许位(1)
1 = 允许发送缓冲区0中断
0 = 禁止发送缓冲区0中断

bit 1 当CAN处于模式0时：
RXB1IE：CAN接收缓冲区1中断允许位
1 = 允许接收缓冲区1中断
0 = 禁止接收缓冲区1中断
当CAN处于模式1或模式2时：
RXBnIE：CAN接收缓冲区中断允许位
1 = 允许接收缓冲区中断；通过BIE0允许单独中断
0 = 禁止所有接收缓冲区中断

bit 0 当CAN处于模式0时：
RXB0IE：CAN接收缓冲区0中断允许位
1 = 允许接收缓冲区0中断
0 = 禁止接收缓冲区0中断
当CAN处于模式1时：
未实现：读为0
当CAN处于模式2时：
FIFOWMIE：FIFO水印中断允许位
1 = 允许FIFO水印中断
0 = 禁止FIFO水印中断

注 1： 在CAN模式1和模式2下，这些位被强制为0。

寄存器34-58： IPR5：外设中断优先级寄存器5

模式0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
	IRXIP	WAKIP	ERRIP	TXB2IP	TXB1IP(1)	TXB0IP(1)	RXB1IP	RXB0IP

模式1和模式2	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
	IRXIP	WAKIP	ERRIP	TXBnIP	TXB1IP(1)	TXB0IP(1)	RXBnIP	FIFOWMIP
	bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **IRXIP**：收到CAN总线错误报文中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **WAKIP**：CAN总线活动唤醒中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **ERRIP**：CAN模块错误中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 当CAN处于模式0时：
TXB2IP：CAN发送缓冲区2中断优先级位
1 = 高优先级
0 = 低优先级
当CAN处于模式1或模式2时：
TXBnIP：CAN发送缓冲区中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **TXB1IP**：CAN发送缓冲区1中断优先级位(1)
1 = 高优先级
0 = 低优先级

bit 2 **TXB0IP**：CAN发送缓冲区0中断优先级位(1)
1 = 高优先级
0 = 低优先级

bit 1 当CAN处于模式0时：
RXB1IP：CAN接收缓冲区1中断优先级位
1 = 高优先级
0 = 低优先级
当CAN处于模式1或模式2时：
RXBnIP：CAN接收缓冲区中断优先级位
1 = 高优先级
0 = 低优先级

注 1： 在CAN模式1和模式2下，这些位被强制为0。

寄存器34-58： IPR5：外设中断优先级寄存器5（续）

bit 0

当CAN处于模式0时：
RXB0IP：CAN接收缓冲区0中断优先级位
1 = 高优先级
0 = 低优先级
当CAN处于模式1时：
未实现：读为0
当CAN处于模式2时：
FIFOWMIP：FIFO水印中断优先级位
1 = 高优先级
0 = 低优先级

注 1： 在CAN模式1和模式2下，这些位被强制为0。

寄存器34-59： TXBIE：发送缓冲区中断允许寄存器(1)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0	U-0
—	—	—	TXB2IE(2)	TXB1IE(2)	TXB0IE(2)	—	—
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-5 **未实现**：读为0

bit 4-2 **TXB2IE:TXB0IE**：发送缓冲区2-0中断允许位(2)
1 = 允许发送缓冲区中断
0 = 禁止发送缓冲区中断

bit 1-0 **未实现**：读为0

注 1： 该寄存器仅适用于模式1和模式2。
2： 必须将PIE5寄存器中的TXBnIE置1才能获取中断。

寄存器34-60： BIE0：缓冲区中断允许寄存器0(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
B5IE(2)	B4IE(2)	B3IE(2)	B2IE(2)	B1IE(2)	B0IE(2)	RXB1IE(2)	RXB0IE(2)
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-2 **B<5:0>IE：** 可编程发送/接收缓冲区5-0中断允许位(2)
1 = 允许中断
0 = 禁止中断

bit 1-0 **RXB<1:0>IE：** 专用接收缓冲区1-0中断允许位(2)
1 = 允许中断
0 = 禁止中断

注 1： 该寄存器仅适用于模式1和模式2。
2： 必须将PIE5寄存器中的TXBnIE或RXBnIE置1才能获取中断。

35.0 固定参考电压（FVR）

固定参考电压（Fixed Voltage Reference，FVR）是独立于VDD的稳定参考电压，可选的输出电压有1.024V、2.048V或4.096V。FVR的输出可以配置为向以下对象提供参考电压：

- ADC输入通道
- ADC正参考电压
- 比较器输入
- 数模转换器（DAC）

FVR可以通过将FVRCON寄存器的EN位置1来使能。

注：	固定参考电压输出不能超过VDD。

35.1 独立的增益放大器

连接至ADC、比较器和DAC的FVR输出会经过两个独立的可编程增益放大器。每个放大器的增益都可以设定为1x、2x或4x，从而产生三种可能电压。

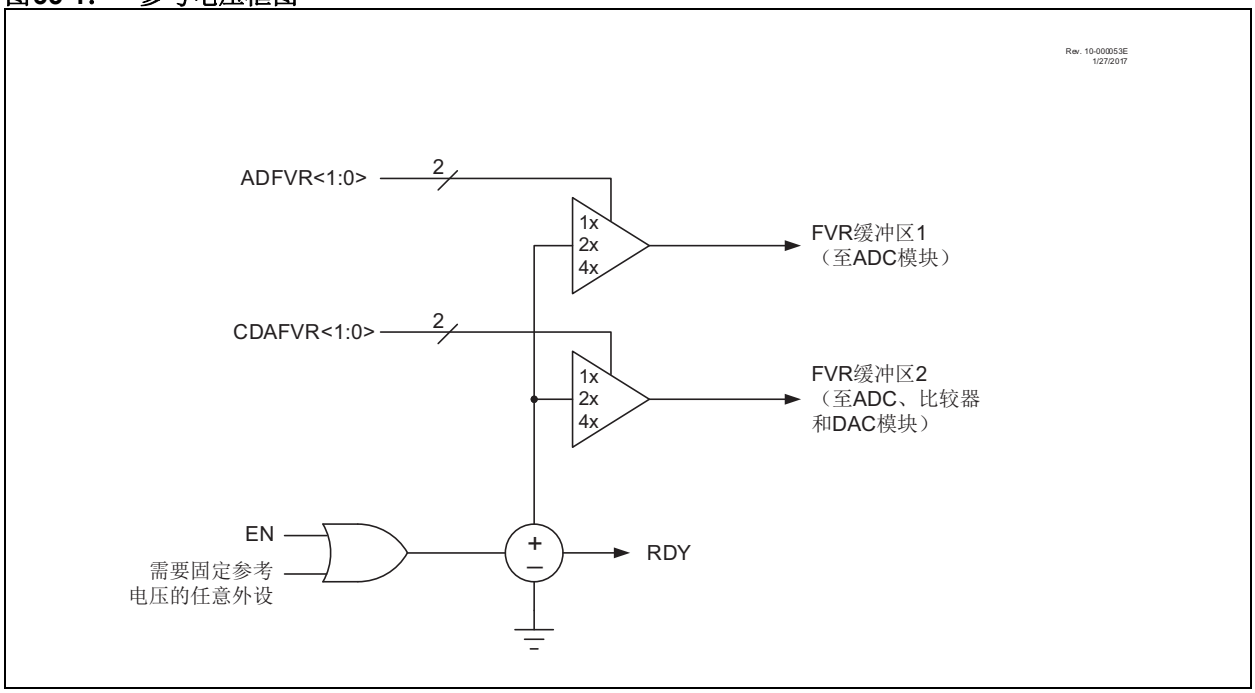
FVRCON寄存器的ADFVR<1:0>位用于使能和配置送到ADC模块的参考电压的增益放大器设置。更多信息，请参见**第37.0节“带计算功能的模数转换器（ADC2）模块”**。

可使用FVRCON寄存器的CDAFVR<1:0>位来为提供给DAC和比较器模块的参考电压使能和配置增益放大器设置。更多信息，请参见**第38.0节“5位数模转换器（DAC）模块”**和**第39.0节“比较器模块”**。

35.2 FVR稳定周期

当固定参考电压模块使能时，参考电压和放大器电路需要一段时间才能达到稳定。在电路稳定下来、可供使用时，FVRCON寄存器的RDY位将会置1。

图35-1： 参考电压框图



35.3 寄存器定义：FVR控制

寄存器35-1： FVRCON：固定参考电压控制寄存器

R/W-0/0	R-q/q	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN	RDY(1)	TSEN(3)	TSRNG(3)	CDAFVR<1:0>		ADFVR<1:0>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	q = 值取决于具体条件

bit 7 **EN**：固定参考电压使能位
1 = 使能固定参考电压
0 = 禁止固定参考电压

bit 6 **RDY**：固定参考电压就绪标志位(1)
1 = 固定参考电压输出就绪备用
0 = 固定参考电压输出未就绪或未使能

bit 5 **TSEN**：温度指示器使能位(3)
1 = 使能温度指示器
0 = 禁止温度指示器

bit 4 **TSRNG**：温度指示器范围选择位(3)
1 = VOUT = 3VT（高电压范围）
0 = VOUT = 2VT（低电压范围）

bit 3-2 **CDAFVR<1:0>**：比较器FVR缓冲区增益选择位
11 = FVR缓冲区1增益为4x（4.096V）(2)
10 = FVR缓冲区1增益为2x（2.048V）(2)
01 = FVR缓冲区1增益为1x（1.024V）
00 = FVR缓冲区1关闭

bit 1-0 **ADFVR<1:0>**：ADC FVR缓冲区增益选择位
11 = FVR缓冲区2增益为4x（4.096V）(2)
10 = FVR缓冲区2增益为2x（2.048V）(2)
01 = FVR缓冲区2增益为1x（1.024V）
00 = FVR缓冲区2关闭

注 1： FVRRDY始终为1。
2： 固定参考电压输出不能超过VDD。
3： 更多信息，请参见**第36.0节“温度指示器模块”**。

表35-1： 与固定参考电压相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
FVRCON	EN	RDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		650

图注： — = 未实现位，读为0。固定参考电压不使用阴影单元。

36.0 温度指示器模块

本系列器件配备了用于测量硅芯片工作温度的温度电路。温度指示器模块主要用于提供与温度相关的电压，该电压可通过模数转换器测得。

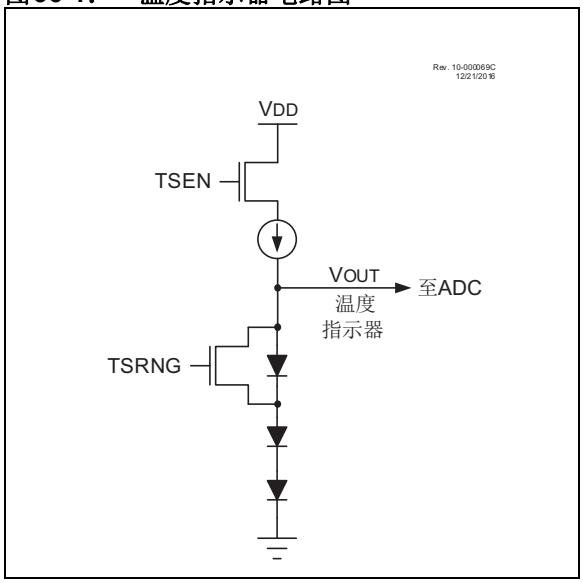
电路的工作温度范围介于-40°C和+125°C之间。电路可以用作温度阈值检测器，也可以用作更精确的温度指示器，具体取决于所执行校准的级别。执行单点校准时，电路可以指示邻近该点的温度。执行双点校准时，电路可以更精确地检测整个温度范围。

36.1 模块工作原理

温度指示器模块由为器件ADC提供电压的温度检测电路组成。模拟电压输出VTSENSE与器件温度成反比。温度指示器的输出称为VOUT。

图36-1给出了温度指示器模块的简化框图。

图36-1： 温度指示器电路图



电路的输出使用内部模数转换器测量。保留一路通道用于温度电路输出。详细信息，请参见**第37.0节“带计算功能的模数转换器（ADC2）模块”**。

模块的ON/OFF位位于FVRCON寄存器中。更多信息，请参见**第35.0节“固定参考电压（FVR）”**。将FVRCON寄存器的TSEN位置1可以使能该电路。禁止模块时，该电路不消耗电流。

电路可以工作于高电压范围或低电压范围。有关范围设置的更多详细信息，请参见**第36.5节“温度指示器范围”**。

36.2 温度估算

本节介绍如何使用传感器电压来估算模块的温度。要使用传感器，需测量输出电压VTSENSE并确定相应的温度。可使用公式36-1基于VTSENSE值估算芯片温度。

公式36-1： 传感器温度

$$T_{SENSE} = V_{TSENSE} \times (-Mt) + T_{OFFSET}$$

其中：

Mt = 1/Mv，其中Mv = 传感器电压灵敏度（V/°C）。

TOFFSET是理论温度和实际温度之间的温差。

36.2.1 校准

36.2.1.1 单点校准

单点校准由应用软件使用公式36-1和假定的Mt来执行。首先获取某个已知温度下的VTSENSE的读数，并通过临时将TOFFSET设置为0来计算理论温度。然后，计算实际温度与计算得到的温度的差值TOFFSET。最后，将TOFFSET存储在器件内的非易失性存储器中，并应用于后续读数以获得更准确的测量结果。

36.2.1.2 高阶校准

如果应用需要更精确的温度测量，则需要执行额外的校准步骤。对于这类应用，建议进行双点或三点校准。

注 1： TOFFSET值可由用户通过温度测试确定。

2： 虽然测量范围为-40°C至+125°C，但由于失调误差的变化，在应用校准失调值之前，未经单点校准的TSENSE计算值可能指示-140°C至+225°C范围内的温度。

3： 用户必须考虑器件在不同时钟频率和输出引脚负载条件下的自发热现象。有关封装的热特性信息，请参见表45-6。

36.2.2 温度分辨率

ADC读数的分辨率Ma（°C/计数）取决于ADC分辨率N与用于转换的参考电压，如公式36-2所示。建议使用最小的VREF值，例如2.048 FVR参考电压，而不是VDD。

注： 关于FVR参考电压精度，请参见表45-17。

公式36-2： 温度分辨率（°C/LSb）

$$Ma = \frac{V_{REF}}{2^N} \times Mt$$

$$Ma = \frac{\frac{V_{REF}}{2^N}}{Mv}$$

其中：

Mv = 传感器电压灵敏度（V/°C）

VREF = ADC模块的参考电压（V）

N = ADC的分辨率

单个二极管的典型Mv值约为-1.267至-1.32 mV/C。

两个二极管堆叠使用时的典型Mv值（低电压范围设置）约为-2.533 mV/C。

三个二极管堆叠使用时的典型Mv值（高电压范围设置）约为-3.8 mV/C。

36.3 ADC采集时间

为了确保精确的温度测量，用户必须在ADC输入多路开关连接到温度指示器输出之后等待一定的时间使ADC值稳定，然后再执行转换。

36.4 最小工作电压VDD

当温度电路工作于低电压范围时，器件可以在规范范围内的任意工作电压下工作。当温度电路工作于高电压范围时，器件工作电压VDD必须足够高，以确保正确地偏置温度电路。

表36-1给出了建议的最小VDD与范围设置。

表36-1： 建议的VDD与范围

最小VDD，TSRNG = 1（高电压范围）	最小VDD，TSRNG = 0（低电压范围）
≥ 2.5	≥ 1.8

36.5 温度指示器范围

温度指示器电路可以工作于高电压范围或低电压范围。高电压范围的选择方式是将FVRCON寄存器的TSRNG位置1，从而可提供较宽的输出电压。这可以在整个温度范围中提供更高的分辨率。高电压范围需要较高的偏置电压才能工作，所以需要较高的VDD。低电压范围的选择方式是将FVRCON寄存器的TSRNG位清零。低电压范围产生的传感器电压较低，所以只需较低的VDD电压就可以让电路工作。

传感器的输出电压在-40°C时达到最大值，在+125°C时达到最小值。

- **高电压范围：**高电压范围用于ADC参考电压VREF = 2.048V的应用。该电压范围可能并不适用于电池供电的应用。
- **低电压范围：**该模式用于VDD过低而无法进行高电压范围操作的应用。该模式下的VDD可以低至1.8V。但是，VDD必须至少比最大传感器电压高0.5V，具体取决于预期的低电压范围工作温度。

36.6 DIA信息

DIA数据提供某一个工作温度下的ADC读数。DIA数据在工厂测试期间获取并存储在器件内。单独的90°C读数可实现第36.2.1节“校准”所述的单点校准，通过求解公式36-1得到TOFFSET。

有关DIA中存储的数据及其访问方式的更多信息，请参见**第5.7节“器件信息区”**。

注：	在较低的温度范围（例如，-40°C）下，精度将受到影响，因为温度转换必须推至参考点以下进行，而这样会放大测量误差。

表36-2： 与温度指示器相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
FVRCON	EN	RDY	TSEN	TSRNG	CDAFVR<1:0>		ADFVR<1:0>		650

图注： — = 未实现位，读为0。温度指示器模块不使用阴影单元。

37.0 带计算功能的模数转换器（ADC^2）模块

带计算功能的模数转换器（ADC^2）可将模拟输入信号转换为信号的12位二进制表示。该模块使用模拟输入，这些输入通过多路开关连接到同一个采样保持电路。采样保持电路的输出与转换器的输入相连接。转换器通过逐次逼近法产生12位二进制结果，并将转换结果存储在ADC结果寄存器（ADRESH:ADRESL寄存器对）中。

此外，ADC模块还提供了以下特性：

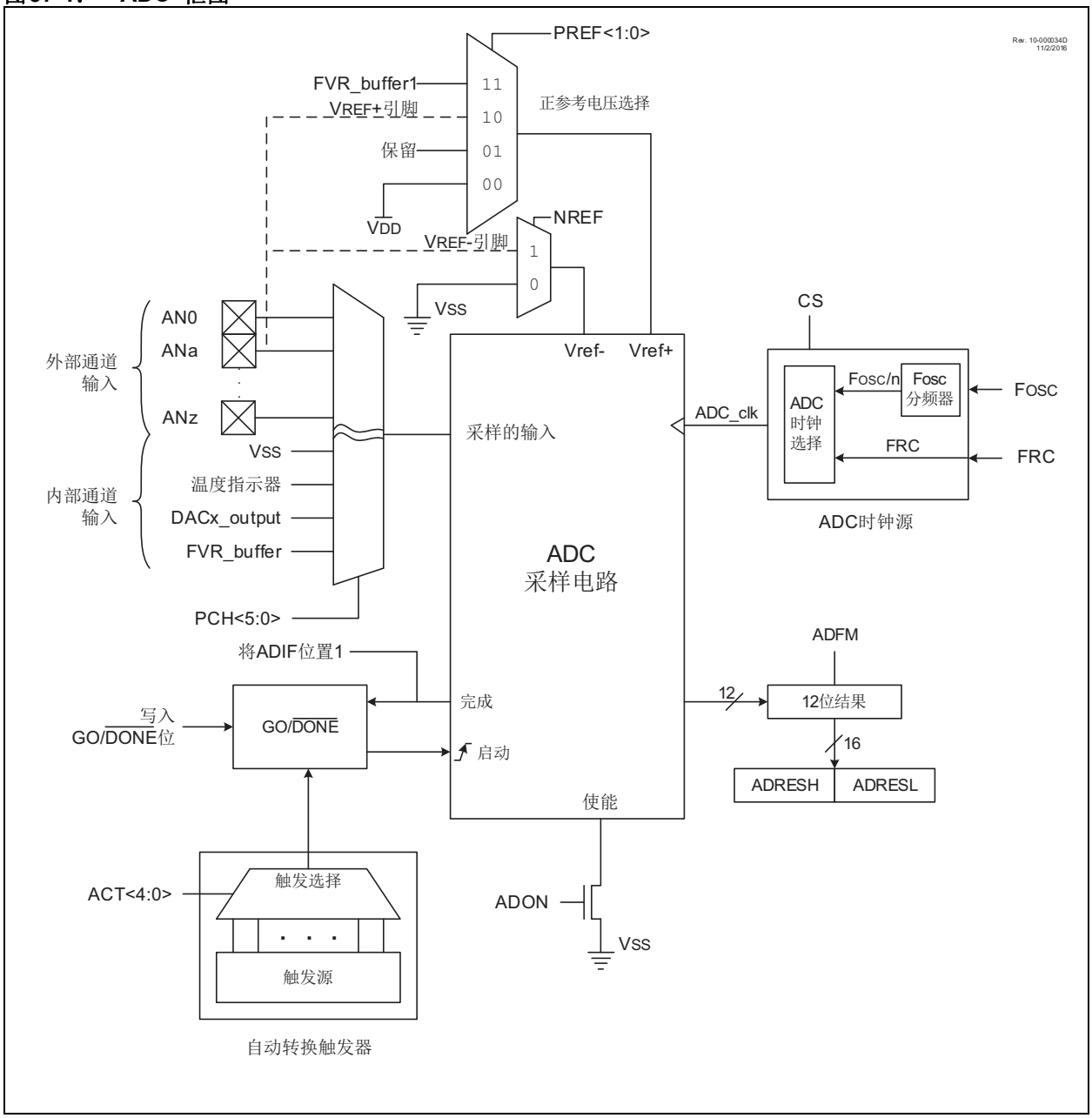
- 13位采集定时器
- 支持硬件电容分压器（CVD）：
  - 13位预充电定时器
  - 可调节采样保持电容阵列
  - 保护环数字输出驱动器
- 自动重复和排序：
  - CVD的自动双采样转换
  - 两组结果寄存器（结果和前一个结果）
  - 自动转换触发器
  - 内部重新触发
- 计算特性：
  - 求平均值和低通滤波功能
  - 参考比较
  - 2级阈值比较
  - 可选中断

图37-1给出了ADC的框图。

可通过软件方式选择内部产生的电压或外部提供的电压作为ADC参考电压。

ADC可在转换完成和阈值比较时产生中断。这些中断可用于将器件从休眠模式唤醒。

图37-1： ADC^{2}框图


PREF<1:0>
FVR_buffer1
VREF+引脚
保留
VDD
11
10
01
00
正参考电压选择
NREF
VREF-引脚
1
0
VSS
AN0
ANa
ANz
外部通道输入
VSS
温度指示器
DACx_output
FVR_buffer
内部通道输入
PCH<5:0>
Vref-
Vref+
采样的输入
ADC
采样电路
CS
ADC_clk
ADC时钟选择
Fosc/n
Fosc分频器
FOSC
FRC
FRC
ADC时钟源
ADFM
12
12位结果
16
ADRESH
ADRESL
将ADIF位置1
完成
写入GO/DONE位
GO/DONE
启动
使能
ACT<4:0>
触发选择
触发源
ADON
VSS
自动转换触发器

37.1 ADC配置

配置和使用ADC时必须考虑以下功能：

- 端口配置
- 通道选择
- ADC参考电压选择
- ADC转换时钟源
- 中断控制
- 结果格式
- 转换触发选择
- ADC采集时间
- ADC预充电时间
- 附加采样保持电容
- 单/双采样转换
- 保护环输出

37.1.1 端口配置

ADC可用于转换模拟信号和数字信号。转换模拟信号时，应通过设置相关的TRIS和ANSEL位将I/O引脚配置为模拟。更多信息，请参见**第16.0节“I/O端口”**。

注：	在任何定义为数字输入的引脚上施加模拟电压可能导致输入缓冲器消耗的电流过大。

37.1.2 通道选择

有多个通道选择可供使用：

- 8个PORTA引脚（RA<7:0>）
- 8个PORTB引脚（RB<7:0>）
- 8个PORTC引脚（RC<7:0>）
- 温度指示器
- DAC输出
- 固定参考电压（FVR）
- VSS（地）

ADPCH寄存器决定与采样保持电路相连接的通道。

当改变通道时，在开始下一次转换前需要一段延时。

更多信息，请参见**第37.2节“ADC工作原理”**。

37.1.3 ADC参考电压

ADREF寄存器的ADPREF<1:0>位控制正参考电压。正参考电压可以是：

- VREF+引脚
- VDD
- FVR输出

ADREF寄存器的ADNREF位控制负参考电压。负参考电压可以是：

- VREF-引脚
- VSS

关于固定参考电压的更多详细信息，请参见**第35.0节“固定参考电压（FVR）”**。

37.1.4 转换时钟

可通过软件设置ADCLK寄存器和ADCON0寄存器的CS位来选择转换时钟源。如果选择FOSC作为ADC时钟，则可通过一个预分频比进行时钟分频，使相应时钟满足ADC时钟周期规范。ADC时钟源选项如下：

- FOSC/(2*n)（其中，n介于1和128之间）
- FRC（专用RC振荡器）

完成一个位的转换所需的时间定义为TAD。有关ADC转换的完整时序详细信息，请参见图37-2。

为正确转换，必须满足相应的TAD规范。更多信息，请参见表45-14。表37-1给出了适当的ADC时钟选择的示例。

注 1：	除非使用FRC，否则系统时钟频率的任何变化都会改变ADC时钟频率，从而对ADC结果产生不利影响。
2：	ADC的内部控制逻辑基于ADCON0的CS位选择的时钟运行。这可能意味着如果ADCON0的CS位置1（ADC基于FRC运行），则将ADC控制位置1时，可能会出现意外的工作延时。

表37-1： ADC时钟周期（TAD）与器件工作频率的关系(1,4)

ADC时钟周期（TAD）		器件频率（FOSC）						
ADC 时钟源	CS<5:0>	64 MHz	32 MHz	20 MHz	16 MHz	8 MHz	4 MHz	1 MHz
FOSC/2	000000	31.25 ns(2)	62.5 ns(2)	100 ns(2)	125 ns(2)	250 ns(2)	500 ns(2)	2.0 μs
FOSC/4	000001	62.5 ns(2)	125 ns(2)	200 ns(2)	250 ns(2)	500 ns(2)	1.0 μs	4.0 μs
FOSC/6	000010	125 ns(2)	187.5 ns(2)	300 ns(2)	375 ns(2)	750 ns(2)	1.5 μs	6.0 μs
FOSC/8	000011	187.5 ns(2)	250 ns(2)	400 ns(2)	500 ns(2)	1.0 μs	2.0 μs	8.0 μs
...	...	...	...	...	...	...	...	...
FOSC/16	000111	250 ns(2)	500 ns(2)	800 ns(2)	1.0 μs	2.0 μs	4.0 μs	16.0 μs(3)
...	...	...	...	...	...	...	...	...
FOSC/128	111111	2.0 μs	4.0 μs	6.4 μs	8.0 μs	16.0 μs(3)	32.0 μs(2)	128.0 μs(2)
FRC	CS（ADCON0<4>）= 1	1.0-6.0 μs	1.0-6.0 μs	1.0-6.0 μs	1.0-6.0 μs	1.0-6.0 μs	1.0-6.0 μs	1.0-6.0 μs

图注： 阴影单元表示超出了建议范围。

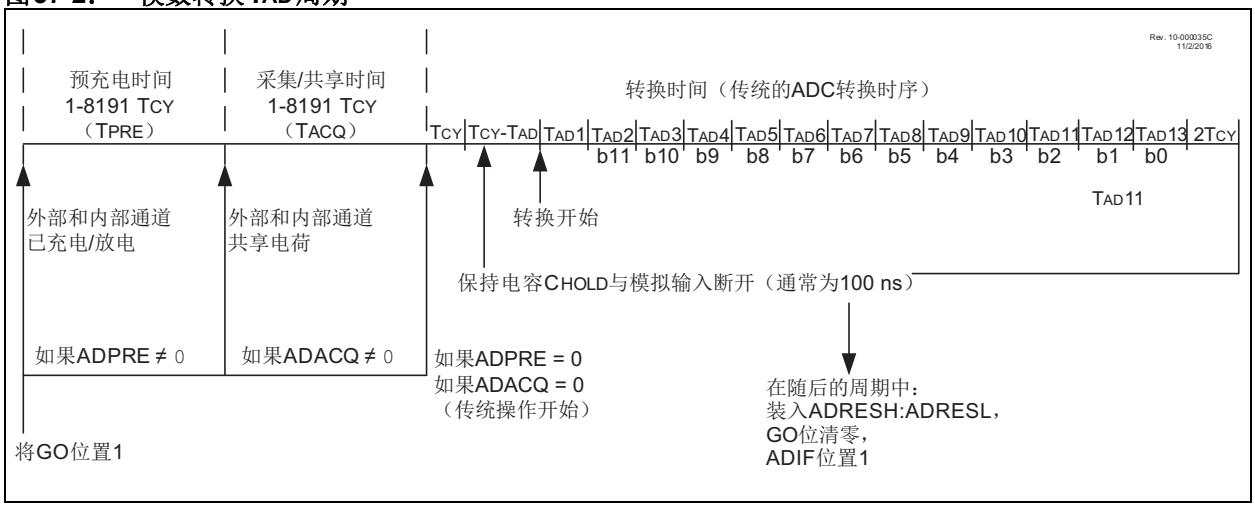
注 1： 有关FRC源的典型TAD值，请参见TAD参数。

2： 这些值违反了所需的TAD时间。

3： 超出了建议的TAD时间范围。

4： 通过系统时钟FOSC来产生ADC时钟时，可以最大程度缩短ADC时钟周期（TAD）和ADC总转换时间。但是，如果要在器件处于休眠模式时执行转换，则必须使用FRC振荡器源。

图37-2： 模数转换TAD周期



37.1.5 中断

ADC模块可在模数转换完成时产生中断。ADC中断标志位是PIR1寄存器中的ADIF位。ADC中断允许位是PIE1寄存器中的ADIE位。ADIF位必须用软件清零。

注 1： ADIF位在每次转换完成时置1，与是否允许ADC中断无关。

2： 仅当在选择了FRC振荡器时，ADC才能在休眠期间工作。

器件工作或休眠时都可产生该中断。如果器件处于休眠模式，该中断会唤醒器件。从休眠模式唤醒时，总是执行紧跟SLEEP指令后的下一条指令。如果用户试图从休眠状态唤醒器件并恢复主代码执行，必须将PIEx寄存器的ADIE位和INTCON0寄存器的GIE位均置1。如果这些位全部置1，则执行将切换到中断服务程序。

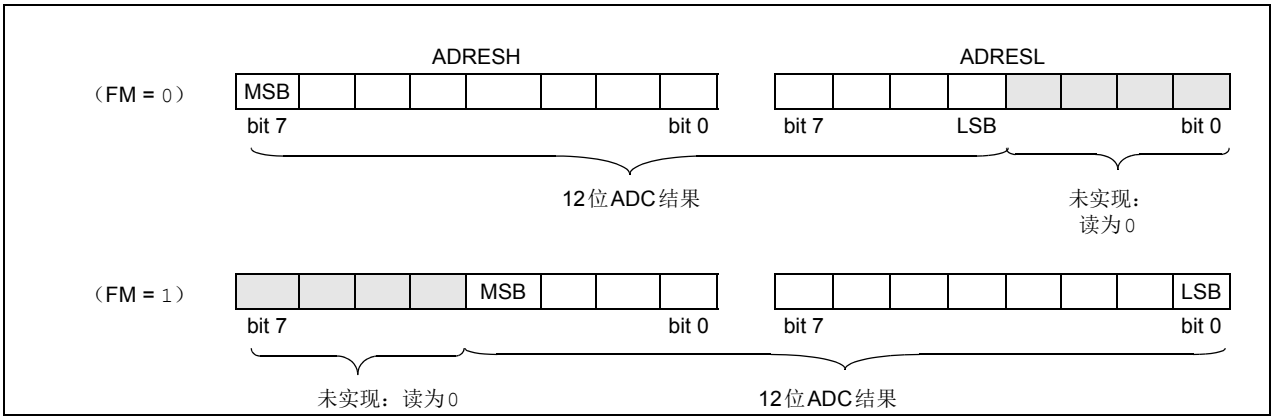
37.1.6 结果格式

12位ADC转换结果可以两种格式提供：左对齐或右对齐。ADCON0寄存器的FM位控制输出格式。

图37-3给出了两种输出格式。

写入ADRES寄存器对时始终采用右对齐，与所选的格式模式无关。因此，如果ADFRM0 = 0，则在写入ADRES后读取的数据将左移四位。

图37-3： 12位ADC转换结果格式



37.2 ADC工作原理

37.2.1 启动转换

要使能ADC模块，必须将ADCON0寄存器的ON位设置为1。可通过以下任一方式启动转换：

- 使用软件将ADCON0的GO位置1
- 外部触发（通过寄存器37-3选择）
- 连续模式重新触发（见**第37.6.8节“连续采样模式”**）

> **注：** 不应在启动ADC的同一指令中将GO位置1。请参见**第37.2.6节“ADC转换步骤（基本模式）”**。

37.2.2 转换完成

当任何单个转换完成时，ADRES中已有的值将写入PREV（如果ADPSIS = 1），新的转换结果将出现在ADRES中。转换完成时，ADC模块将执行以下操作：

- 将GO位清零（除非ADCON0的CONT位置1）
- 将ADIF中断标志位置1
- 将ADMATH位置1
- 更新ACC

在ADDSEN = 0时的每次转换后或者在ADDSEN = 1时的每隔一次转换后，将发生以下事件：

- 计算ERR
- 如果ERR计算满足阈值比较，则ADTIF将置1

重要的是，转换本身完成后会发生滤波和阈值计算。因此，响应ADIF的中断处理程序应在读取滤波和阈值结果之前检测ADTIF。

37.2.3 休眠期间的ADC操作

ADC模块可以在休眠模式下工作。这需要将ADC时钟源设置为FRC选项。当选择FRC振荡器源时，ADC需等待一个额外的指令周期后才能启动转换。这使得可以执行SLEEP指令，这将降低转换期间的系统噪声。如果允许了ADC中断，转换完成时器件将从休眠状态唤醒。如果禁止了ADC中断，尽管ON位仍保持置1，但转换完成后ADC模块将关闭。

37.2.4 休眠期间的外部触发

在休眠期间，如果在ADC时钟源设置为FRC时接收到外部触发信号，则ADC模块将执行转换并在完成后将ADIF位置1。

如果在ADC时钟源不是FRC时接收到外部触发信号，则会记录触发信号，但直到器件退出休眠模式后才会开始转换。

37.2.5 自动转换触发器

自动转换触发器允许定期进行ADC测量而无需软件干预。当出现选定触发源的上升沿时，GO位由硬件置1。

自动转换触发源由ADACT寄存器选择。

使用自动转换触发器不能确保正确的ADC时序。用户需负责确保满足ADC时序要求。关于自动转换源，请参见寄存器37-33。

37.2.6 ADC转换步骤（基本模式）

以下是使用ADC执行模数转换的示例步骤：

1. 配置端口：
   - 禁止引脚输出驱动器（见TRISx寄存器）
   - 将引脚配置为模拟引脚（见ANSELx寄存器）
2. 配置ADC模块：
   - 选择ADC转换时钟
   - 选择参考电压
   - 选择ADC输入通道
   - 预充电和采集
   - 开启ADC模块
3. 配置ADC中断（可选）：
   - 清零ADC中断标志
   - 允许ADC中断
   - 允许全局中断（GIEL位）(1)
4. 如果ADACQ = 0，软件必须等待所需的采集时间(2)。
5. 通过将GO位置1来启动转换。
6. 通过以下方式之一等待ADC转换完成：
   - 轮询GO位
   - 轮询ADIF位
   - 等待ADC中断（已允许中断）
7. 读取ADC结果。
8. 清零ADC中断标志（如果已允许中断则需要此操作）。

注 1： 如果用户试图从休眠模式唤醒器件并恢复主代码执行，必须禁止全局中断。

2： 请参见**第37.3节“ADC采集要求”**。

例37-1： ADC转换

```
/*This code block configures the ADC
for polling, VDD and VSS references, FRC
oscillator and AN0 input.
Conversion start & polling for completion
are included.
 */
void main() {
    //System Initialize
    initializeSystem();

    //Setup ADC
    ADCON0bits.FM = 1; //right justify
    ADCON0bits.CS = 1; //FRC Clock
    ADPCH = 0x00; //RA0 is Analog channel
    TRISAbits.TRISA0 = 1; //Set RA0 to input
    ANSELAbits.ANSELA0 = 1; //Set RA0 to analog
    ADCON0bits.ON = 1; //Turn ADC On

    while (1) {
        ADCON0bits.GO = 1; //Start conversion
        while (ADCON0bits.GO); //Wait for conversion done
        resultHigh = ADRESH; //Read result
        resultLow = ADRESL; //Read result
    }
}
```

37.3 ADC采集要求

为了使ADC达到规定的精度，必须使充电保持电容（CHOLD）完全充电至输入通道的电压。模拟输入模型如图37-4所示。模拟信号源阻抗（RS）和内部采样开关阻抗（RSS）直接影响电容CHOLD的充电时间。采样开关阻抗（RSS）随器件电压（VDD）的变化而变化，请参见图37-4。**模拟信号源的最大阻抗推荐值为10 kΩ。**

如果源阻抗降低，采集时间可能会缩短。在选择（或改变）模拟输入通道后，必须在启动转换前完成ADC采集。可以使用公式37-1来计算最小采集时间。该公式假设误差为1/2 LSb（ADC转换需要4,096步）。1/2 LSb误差是ADC达到规定分辨率所能允许的最大误差。

公式37-1： 采集时间示例

假设： *温度 = 50°C，外部阻抗为10 kΩ，VDD为5.0V*

$$\begin{aligned} TACQ &= \text{放大器稳定时间} + \text{保持电容充电时间} + \text{温度系数} \\ &= TAMP + TC + TCOFF \\ &= 2\mu s + TC + [(\text{温度} - 25°C)(0.05\mu s/°C)] \end{aligned}$$

TC 值可以用以下公式近似计算：

$$VAPPLIED\left(1 - \frac{1}{(2^{n+1}) - 1}\right) = VCHOLD \quad ; \text{ [1] 充电到 VCHOLD（1/2 LSb 误差范围）}$$

$$VAPPLIED\left(1 - e^{\frac{-TC}{RC}}\right) = VCHOLD \quad ; \text{ [2] 响应 VAPPLIED 充电到 VCHOLD}$$

$$VAPPLIED\left(1 - e^{\frac{-Tc}{RC}}\right) = VAPPLIED\left(1 - \frac{1}{(2^{n+1}) - 1}\right) \quad ; \text{ 合并 [1] 和 [2]}$$

注：其中 n = ADC 的位数。

求解 TC：

$$\begin{aligned} TC &= -CHOLD(RIC + RSS + RS)\ ln(1/8191) \\ &= -28\ pF(1\ k\Omega + 7\ k\Omega + 10\ k\Omega)\ ln(0.0001221) \\ &= 4.54\ \mu s \end{aligned}$$

因此：

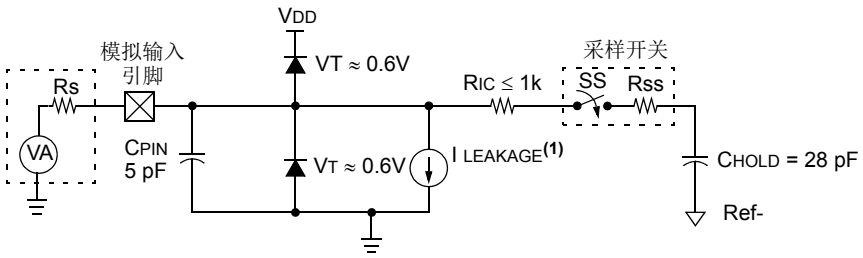
$$\begin{aligned} TACQ &= 2\ \mu s + 4.54\ \mu s + [(50°C - 25°C)(0.05\ \mu s/°C)] \\ &= 7.79\ \mu s \end{aligned}$$

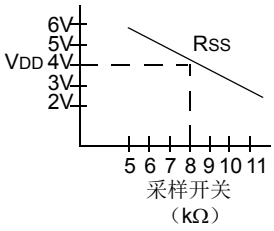
注 1： 因为参考电压（VREF）自行抵消，因此它对该公式没有影响。

2： 充电保持电容（CHOLD）在每次转换后不会放电。

3： 模拟信号源的最大阻抗推荐值为10 kΩ。要求符合引脚泄漏电流的规范。

图37-4： 模拟输入模型



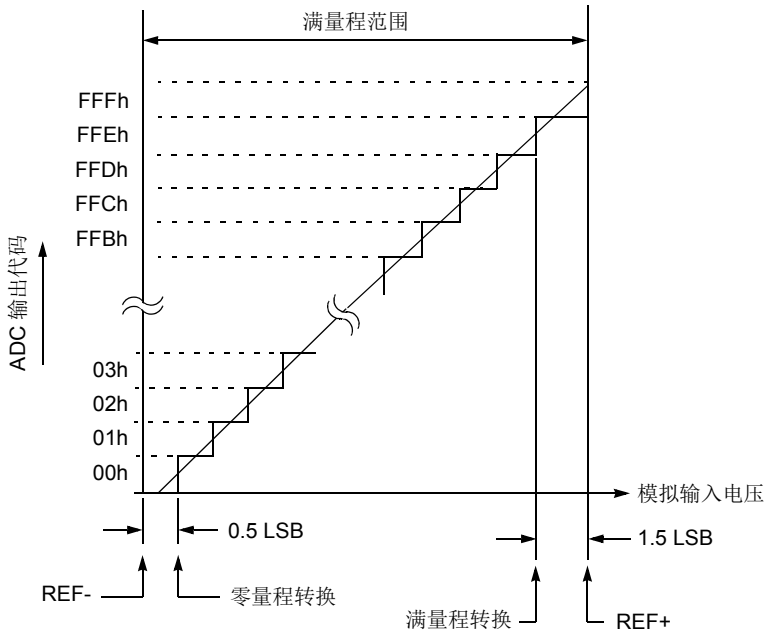


图注：

CHOLD	=	采样/保持电容
CPIN	=	输入电容
I LEAKAGE	=	各个连接点在引脚上产生的泄漏电流
RIC	=	片内走线等效电阻
RSS	=	采样开关的电阻
SS	=	采样开关
VT	=	阈值电压

注 1： 请参见表45-4（参数D340和D341）。

图37-5： ADC传递函数



37.4 ADC 电荷泵

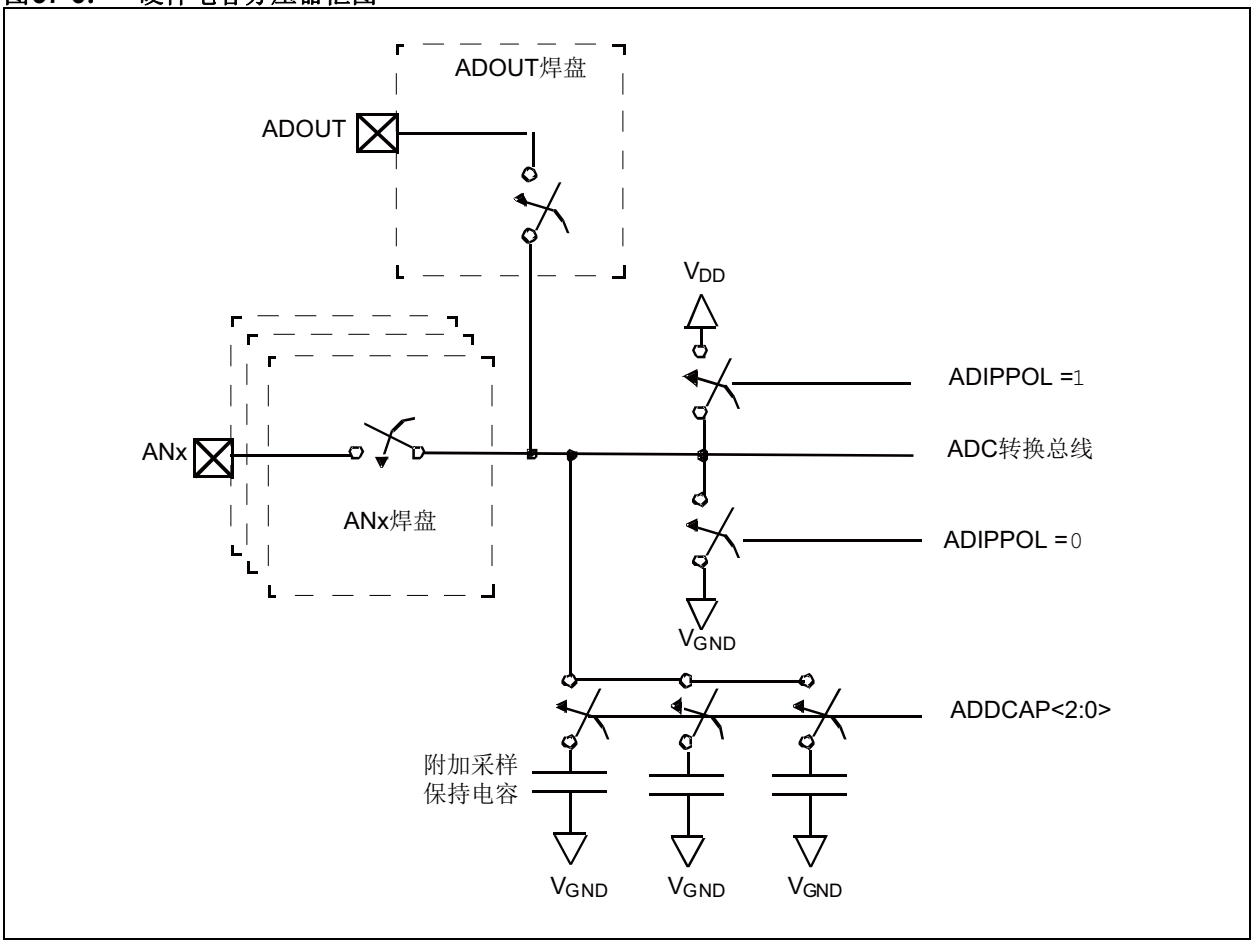
ADC 模块具有一个专用电荷泵，可通过ADCP 寄存器（寄存器37-36）进行控制。电荷泵的主要用途是为A/D 转换器中晶体管器件的栅极、信号以及参考输入传输门提供恒定电压，以防止晶体管在低工作电压下的性能下降。

电荷泵可通过将ADC 寄存器的CPON 位置1 来使能。使能后，电荷泵将经历一段启动时间来使电荷泵输出达到稳定。输出稳定下来并可供使用后，ADCP 寄存器的CPRDY 位将置1。

37.5 电容分压器（CVD）特性

ADC 模块包含多个特性，允许用户以内部ADC 采样保持电容为参考在任何ADC 通道上执行相对电容测量。此相对电容测量可用于实现电容式触摸或接近传感应用。图37-6 所示为ADC 模块的CVD 部分的基本框图。

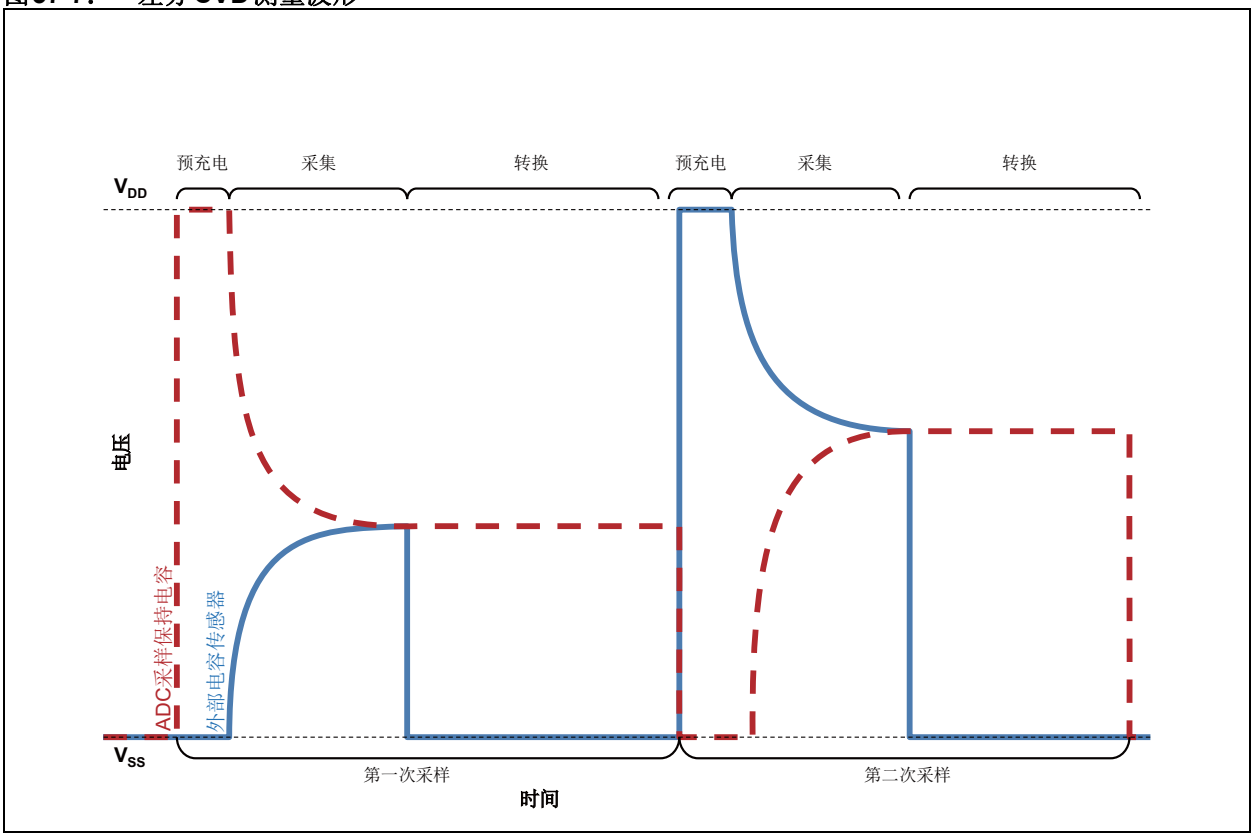
图37-6： 硬件电容分压器框图



37.5.1 CVD操作

ADC的内部采样保持电容（C_{HOLD}）从连接其与外部电容传感器节点的路径断开时，CVD开始工作。断开后，C_{HOLD}预充电至V_{DD}或V_{SS}，而连至传感器节点的路径预充电至与C_{HOLD}相反的电平。预充电阶段完成时，两个节点的V_{DD}/V_{SS}预充电路径关闭，C_{HOLD}与连至外部传感器节点的路径重新连接，此时CVD开始采集阶段。采集期间，预充电C_{HOLD}与传感器节点之间构成一个电容分压器，最终在C_{HOLD}上获得一个稳定的电压，电压值由电容值和两个节点的预充电电压决定。采集结束后，ADC将转换C_{HOLD}上的电压。该过程将在之后重复执行，但会针对C_{HOLD}和外部传感器节点反相预充电电平。图37-7给出了两次反相CVD测量（称为差分CVD测量）的波形。

图37-7： 差分CVD测量波形



37.5.2 预充电控制

预充电阶段是一个可选时间段，可将外部通道和内部采样保持电容置于已知电压。通过向PRE寄存器写入非零值来使能预充电。当通过将GO位置1、特殊事件触发信号或由计算功能实现的转换重启开始ADC转换时，将启动此阶段。如果ADC转换开始时PRE寄存器清零，则跳过此阶段。

在预充电期间，CHOLD从外部与连接外部电容传感器的采样路径断开并连接到VDD或VSS，具体取决于ADCON1寄存器的ADPPOL位的值。同时，所选模拟通道的端口引脚逻辑被改写以驱动数字高/低输出，以便对ADC外部采样路径（包括外部传感器）进行预充电。此改写的输出极性也由ADCON1寄存器的ADPPOL位决定。该充电时间由PRE寄存器控制。

注 1：	外部充电将改写相应I/O引脚的TRIS设置。
2：	如果有一个器件连接到该引脚，则不应使用预充电。

37.5.3 采集控制

采集阶段为一段可选时间，可针对内部采样保持电容的电压通过所选模拟通道充电或放电。该采集时间由ADACQ寄存器控制。如果PRE = 0，则采集阶段在转换开始时开始。如果PRE = 1，则采集阶段在预充电结束时开始。

采集阶段开始时，所选模拟通道的端口引脚逻辑被改写以关断数字高/低输出驱动器，从而使这些驱动器不影响电荷平均的最终结果。并且，所选ADC通道连接到CHOLD。这样便可在预充电通道和CHOLD电容之间进行电荷平均。

注：	当PRE! = 0时，采集时间不能为0。在这种情况下，将ADACQ设置为0将设置最大采集时间（8191个ADC时钟周期）。禁止预充电时，将ADACQ设置为0将禁止硬件采集时间控制。

37.5.4 保护环输出

图37-8所示为典型的保护环电路。CGUARD表示PCB板上保护环走线的电容。用户选择RA和RB的值，使CGUARD的电压曲线与所选采集通道匹配。

保护环的作用是产生一个与CVD传感信号同相的信号，以最大程度降低寄生电容对传感电极的影响。它还可用作互驱动器以实现电容性互感。有关主动保护和互驱动器的更多信息，请参见应用笔记AN1478《mTouch® 触摸传感解决方案采集方法电容分压器》（DS01478B_CN）。

ADC有两个保护环驱动输出，即ADGRDA和ADGRDB。这些输出可以通过PPS控制连接到I/O引脚（有关详细信息，见**第17.0节“外设引脚选择（PPS）模块”**），这些输出的极性由ADCON1的ADGPOL和ADIPEN位控制。

在第一个预充电阶段开始时，两个输出都设置为匹配ADCON1的ADGPOL位。采集阶段开始后，ADGRDA将改变极性，而ADGRDB保持不变。当执行双采样转换时，将ADCON1的ADIPEN位置1会导致两个保护环输出在第二个预充电阶段开始时转换为ADGPOL的相反极性，ADGRDA将再次翻转以进行第二次采集。有关保护环输出时序的更多信息，请参见图37-8和图37-9。

图37-8： 保护环电路

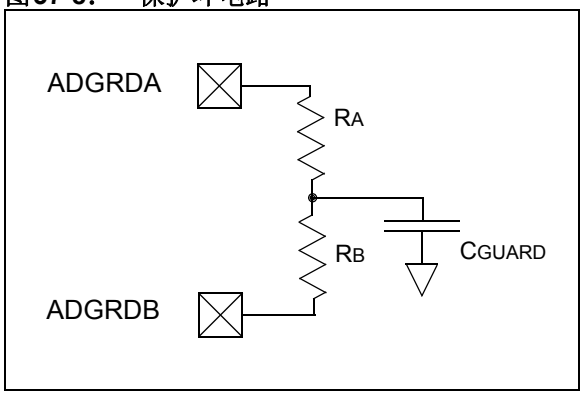
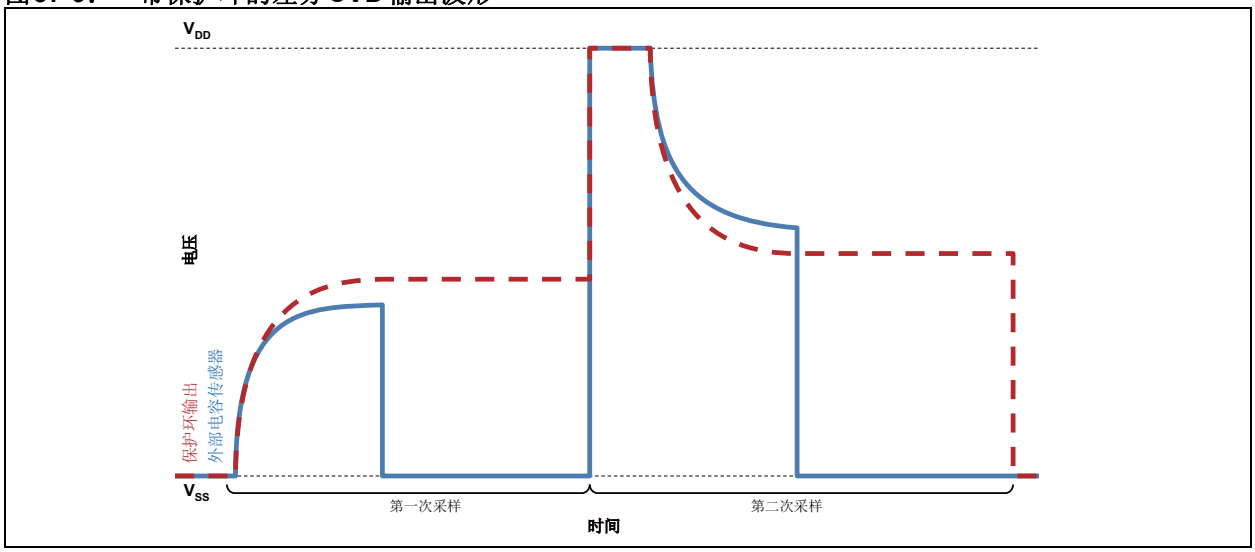


图37-9： 带保护环的差分CVD输出波形



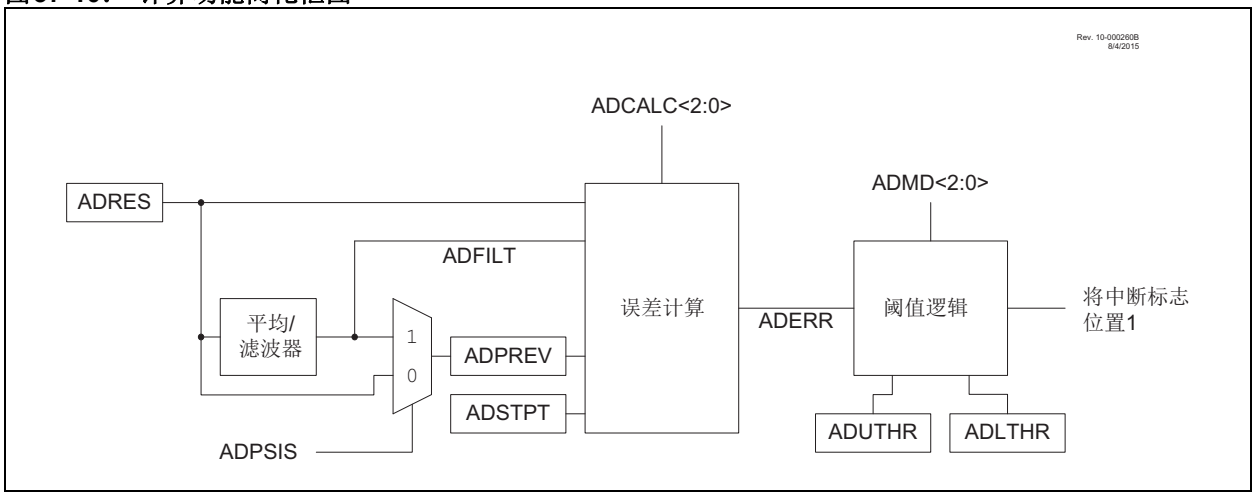
37.5.5 附加采样保持电容

可使用ADCAP寄存器为内部采样保持电容（CHOLD）并联一个附加电容。此寄存器用于选择可数字编程的电容（添加到ADC转换总线），从而增加ADC模块中采样保持电容的有效内部电容。这用于提高内部电容和外部电容的匹配度，从而实现更好的传感性能。由于转换期间未连接附加电容，因此不会影响ADC的模拟性能。请参见图37-10。

37.6 计算操作

ADC模块硬件配有转换后计算功能。这些功能提供可作用于ADC转换结果的数据后处理功能，包括数字滤波/平均值计算和阈值比较功能。

图37-10： 计算功能简化框图



ADC计算功能的具体操作由ADCON2寄存器的ADMD<2:0>位控制。

该模块可在以下五种模式之一下工作：

• **基本模式**：在该模式下，ADC转换基于单（ADDSEN = 0）/ 双（ADDSEN = 1）采样进行。ADIF在所有转换完成后置1。

• **累加模式**：每次触发时，ADC转换结果都将加到累加器中，CNT递增。ADIF在每次转换后置1。ADTIF根据计算模式置1。

• **平均模式**：每次触发时，ADC转换结果都加到累加器中。当RPT采样数累加完成后，将执行阈值测试。下一次触发时，累加器将清零。对于后续测试，则需要累加额外的RPT采样。

• **突发平均模式**：触发时，累加器将清零。随后将重复收集ADC转换结果，直至完成RPT采样累加，最终会测试阈值。

• **低通滤波器（LPF）模式**：每次触发时，ADC转换结果均会通过滤波器发送。完成RPT采样后，将执行阈值测试。此后每次触发时，都将通过滤波器发送ADC转换结果并再次执行阈值测试。

下面的表37-2总结了上述五种模式。

表37-2： 计算模式

		位清零条件	触发完成后的值		阈值操作			ADTIF中断时的值		
模式	ADMD	ACC和CNT	ACC	CNT	重新触发	阈值测试	中断	ADAOV	FLTR	CNT
基本	0	ADACLR = 1	不变	不变	否	每次采样	如果阈值 = 真	N/A	N/A	计数
累加	1	ADACLR = 1	S + ACC 或 (S2-S1) + ACC	如果CNT=0xFF：CNT， 否则：CNT+1	否	每次采样	如果阈值 = 真	ACC溢出	$ACC/2^{ADCRS}$	计数
平均	2	ADACLR = 1、CNT>=RPT（GO时）或重新触发	S + ACC 或 (S2-S1) + ACC	如果CNT=0xFF：CNT， 否则：CNT+1	否	如果 CNT>=RPT	如果阈值 = 真	ACC溢出	$ACC/2^{ADCRS}$	计数
突发平均	3	ADACLR = 1、GO置1或重新触发	每次重复：与平均模式相同以将所有采样相加结束	每次重复：与平均模式相同以CNT=RPT结束	在CNT<RPT时重复	如果 CNT>=RPT	如果阈值 = 真	ACC溢出	$ACC/2^{ADCRS}$	RPT
低通滤波器	4	ADACLR = 1	$S+ACC-ACC/2^{ADCRS}$ 或 $(S2-S1)+ACC-ACC/2^{ADCRS}$	递增计数， CNT = 0xFF时停止计数	否	如果 CNT>=RPT	如果阈值 = 真	ACC溢出	滤波后的值	计数

注： S1和S2分别为采样1和采样2的缩写。当ADDSEN = 0时，S1 = ADRES；当ADDSEN = 1时，S1 = PREV且S2 = ADRES。

37.6.1 数字滤波器/平均值计算

数字滤波器/平均值计算模块由具有数据反馈选项的累加器以及用于确定何时需要应用阈值测试的控制逻辑组成。该累加器是一个可以通过ADACCH:ADACCL寄存器对访问的16位宽寄存器。

每次发生触发事件（GO位置1或外部事件触发）时，ADC转换结果都加到累加器中。如果累加结果超过$2^{(accumulator_width)}$-1（即262143，其中accumulator_width = 18），则ADSTAT寄存器的溢出位ADAOV置1。

要累加的采样数由RPT（A/D重复设置）寄存器决定。每次将采样加到累加器中，ADCNT寄存器都会递增。累加RPT个采样（CNT = RPT）后，可以通过用软件将ADCON2寄存器的ADACLR位置1的方式来发出累加器清零命令。将ADACLR位置1还将清零ADSTAT寄存器的ADAOV（累加器溢出）位以及ADCNT寄存器。累加器清零操作完成时，ADACLR位将由硬件清零。

注：	当ADC通过FRC工作时，需要五个FRC时钟周期来执行ACC清零操作。

ADCON2寄存器的ADCRS <2:0>位用于控制累加器结果的数据移位，可以有效地对累加器（ADACCU:ADACCH:ADACCL）寄存器对中的值进行除法运算。对于数字滤波器的累加模式，移位提供了一种简单的换算方法。对于平均模式/突发平均模式，移位用于确定要对累加结果执行的逻辑右移的位数。对于低通滤波器模式，移位是滤波器不可或缺的一部分，决定了滤波器的截止频率。表37-3列出了用于计算-3 dB增益处截止频率的相应ωT（弧度）和奈奎斯特频率（ωT = π）下该滤波器的最大信号衰减。

表37-3： -3 dB增益处低通滤波器的截止频率

ADCRS	用于计算-3 dB增益处频率的相应ωT（弧度）	$F_{nyquist}$ = 1/(2T)时的增益（dB）
1	0.72	-9.5
2	0.284	-16.9
3	0.134	-23.5
4	0.065	-29.8
5	0.032	-36.0
6	0.016	-42.0
7	0.0078	-48.1

37.6.2 基本模式

基本模式（ADMD = 000）禁止所有附加计算功能。在该模式下，不发生累加，但执行阈值误差比较。双采样、连续模式和所有CVD功能仍然可用，但不会使用涉及数字滤波器/平均值计算的功能。

37.6.3 累加模式

在累加模式（ADMD = 001）下，每次转换后，ADC结果都会加到ADACC寄存器中。ADACC寄存器按照ADCON2寄存器的ADCRS位的值右移。该右移值会复制到ADFLT寄存器。格式模式不会影响ACC值的右对齐。每次采样时，CNT也会递增，因此累加的采样数会递增。每次采样和累加后，都会对ACC值执行阈值比较（见**第37.6.7节“阈值比较”**），并且可能触发ADTIF中断。

37.6.4 平均模式

在平均模式（ADMD = 010）下，ADACC寄存器随每次ADC采样累加，这与累加模式几乎相同，ADCNT寄存器随每次采样递增。ADFLT寄存器也将用ADACC寄存器的右移值进行更新。ADCRS位的值管理右移位数。但是，在平均模式下，当CNT大于或等于用户定义的RPT值时会执行阈值比较。在该模式下，当RPT = 2^CNT时，最终的累加值将除以采样数，以便对收集的所有采样的平均值执行阈值比较操作。

37.6.5 突发平均模式

突发平均模式（ADMD = 011）的作用在大多数方面与平均模式相同。其中一个不同之处在于，即使未使能连续采样模式（见**第37.6.8节“连续采样模式”**），该模式也会连续重新触发ADC采样，直到CNT值大于或等于RPT。这样可以对ADC采样的短突发平均值执行阈值比较。

37.6.6 低通滤波器模式

低通滤波器模式（ADMD = 100）在如何处理采样方面的作用与平均模式相似（累加样本直到CNT值大于或等于RPT，然后触发阈值比较），但是该模式会对所有采样执行低通滤波操作而不是进行简单的平均值计算，从而降低了高频噪声对平均值的影响，然后再对结果进行阈值比较。（有关数学运算的更多详细说明，请参见表37-2）。在该模式下，ADCRS位决定了低通滤波器的截止频率（如表37-3所示）。

37.6.7 阈值比较

在每次计算结束时：

- 转换结果在转换结束时锁存并保持稳定。
- 误差基于ADCON3寄存器的ADCALC<2:0>位选择的差值计算来计算。该值可以是以下计算结果之一（有关更多详细信息，见寄存器37-4）：
  - 单次测量结果的一阶导数
  - CVD模式下的CVD结果
  - 当前结果与设定值之差
  - 当前结果与滤波后的结果/平均值计算结果之差
  - 滤波后的值/平均值的一阶导数
  - 滤波后的值/平均值与设定值之差
- 计算结果（ERR）将与上限阈值和下限阈值（UTH<ADUTHH:ADUTHL>和LTH<ADLTHH:ADLTHL>寄存器）进行比较，以设置ADUTHR和ADLTHR标志位。阈值逻辑通过ADCON3寄存器的ADTMD<2:0>位选择。阈值触发选项可以是以下各项之一：
  - 永不中断
  - 误差小于下限阈值
  - 误差大于或等于下限阈值
  - 误差介于上限阈值和下限阈值之间（包含上限和下限）
  - 误差超出阈值
  - 误差小于或等于上限阈值
  - 误差大于上限阈值
  - 无论阈值测试结果如何，始终中断
  - 如果满足阈值条件，则阈值中断标志 ADTIF置1。

注 1： 阈值测试为有符号运算。

2： 如果ADAOV置1，则发出阈值中断信号。

37.6.8 连续采样模式

将ADCON0寄存器的CONT位置1会在更新ADACC寄存器后自动重新触发新的转换周期。GO位保持置1状态，并且自动进行重新触发。

如果ADSOI = 1，阈值中断条件将清零GO，转换将停止。

37.6.9 双采样转换

可通过将ADCON1寄存器的ADDSEN位置1使能双采样。该位置1时，在模块计算阈值误差之前需要进行两次转换（每次转换仍须单独触发）。第一次转换将ADSTAT寄存器的ADMATH位置1并更新ADACC，但不会计算ERR或触发ADTIF。当第二次转换完成时，第一个值会传输到PREV（取决于ADPSIS的设置），而第二次转换的值会放置在ADRES中。只有在第二次转换完成时，才会计算ERR和触发ADTIF（取决于ADCALC的值）。

37.7 寄存器定义：ADC控制

寄存器37-1： ADCON0：ADC控制寄存器0

R/W-0/0	R/W-0/0	U-0	R/W-0/0	U-0	R/W-0/0	U-0	R/W/HC-0
ON	CONT	—	CS	—	FM	—	GO
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **ON**：ADC使能位
1 = 使能ADC
0 = 禁止ADC

bit 6 **CONT**：ADC连续操作使能位
1 = 每次转换触发完成时都会重新触发GO，直到ADTIF置1（如果ADSOI置1）或GO清零（无论ADSOI的值如何）
0 = 每次转换触发完成时都会清零ADC

bit 5 **未实现**：读为0

bit 4 **CS**：ADC时钟选择位
1 = 由FRC专用振荡器提供时钟
0 = 由Fosc提供时钟（根据ADCLK寄存器分频）

bit 3 **未实现**：读为0

bit 2 **FM**：ADC结果格式/对齐选择
1 = ADRES和PREV数据是右对齐的
0 = ADRES和PREV数据是左对齐的，并经过零填充

bit 1 **未实现**：读为0

bit 0 **GO**：ADC转换状态位(1)
1 = ADC转换正在进行。将该位置1可启动ADC转换周期。该位由硬件清零（取决于CONT位）
0 = ADC转换已完成/未进行

注 1： 该位需要将ON位置1。
2： 如果正在进行转换时使用软件清零，则在此之前的转换结果将传送到ADRES并且状态机将复位，但ADIF中断标志位不会置1；滤波和阈值操作将不会执行。

寄存器37-2： ADCON1：ADC控制寄存器1

R/W-0/0	R/W-0/0	R/W-0/0	U-0	U-0	U-0	U-0	R/W-0/0
PPOL	IPEN	GPOL	—	—	—	—	DSEN
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **PPOL：**预充电极性位
如果PRE>0x00：

PPOL	第1个预充电阶段中的操作	
	外部（选定的模拟I/O引脚）	内部（AD采样电容）
1	连接至V_{DD}	C_{HOLD}连接至V_{SS}
0	连接至V_{SS}	C_{HOLD}连接至V_{DD}

其他情况：
忽略该位

bit 6 **IPEN：**A/D反相预充电使能位
如果DSEN = 1
1 = 第二个转换周期中的预充电和保护信号的极性与第一个周期中的相反
0 = 两个转换周期均使用ADPPOL和ADGPOL指定的预充电和保护信号
其他情况：
忽略该位

bit 5 **GPOL：**保护环极性选择位
1 = 在预充电阶段，ADC保护环输出以数字高电平开始
0 = 在预充电阶段，ADC保护环输出以数字低电平开始

bit 4-1 **未实现：**读为0

bit 0 **DSEN：**双采样使能位
1 = 每次触发都执行两次转换。第一次转换的数据将出现在PREV中
0 = 每次触发执行一次转换

寄存器37-3： ADCON2：ADC控制寄存器2

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W/HC-0	R/W-0/0	R/W-0/0	R/W-0/0
PSIS	CRS<2:0>			ACLR	MD<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **PSIS**：ADC前一次采样输入选择位
1 = 开始转换时PREV为FLTR的值
0 = 开始转换时PREV为RES的值

bit 6-4 **CRS<2:0>**：ADC累加计算右移选择位
如果ADMD = 100：
低通滤波器时间常数为2^{ADCRS}，滤波器增益为1:1
如果ADMD = 001、010或011：
累加值按CRS右移（除以2^{ADCRS}）**(1,2)**
其他情况：
忽略这些位

bit 3 **ACLR**：A/D累加器清零命令位**(3)**
1 = ACC、AOV和CNT寄存器清零
0 = 清零操作已完成（或未开始）

bit 2-0 **MD<2:0>**：ADC工作模式选择位**(4)**
111-101 = 保留
100 = 低通滤波器模式
011 = 突发平均模式
010 = 平均模式
001 = 累加模式
000 = 基本模式

注 1： 要正确计算平均值，采样数（在RPT中设置）必须是2^{ADCRS}。
2： ADCRS = 3'b111为保留选项。
3： 累加器操作完成时该位由硬件清零；延时可能为多条指令的时间，具体取决于振荡器选择。
4： 有关完整模式说明，请参见表37-2。

寄存器37-4： ADCON3：ADC控制寄存器3

U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W/HC-0	R/W-0/0	R/W-0/0	R/W-0/0
—	CALC<2:0>			SOI	TMD<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HC = 硬件清零位

bit 7 **未实现：**读为0

bit 6-4 **CALC<2:0>：**ADC误差计算模式选择位

CALC	DSEN = 0单采样模式	DSEN = 1 CVD双采样模式(1)	应用
111	保留	保留	保留
110	保留	保留	保留
101	FLTR-STPT	FLTR-STPT	平均值/滤波后的值与设定值之差
100	PREV-FLTR	PREV-FLTR	滤波后的值的一阶导数(3)（负）
011	保留	保留	保留
010	RES-FLTR	(RES-PREV)-FLTR	实际结果与平均值/滤波后的值之差
001	RES-STPT	(RES-PREV)-STPT	实际结果与设定值之差
000	RES-PREV	RES-PREV	单次测量结果的一阶导数(2)
			CVD模式下的实际CVD结果(2)

bit 3 **SOI：**ADC中断时停止位

如果CONT = 1：

1 = 满足阈值条件时清零GO，否则重新触发转换

0 = 不通过硬件清零GO，必须使用软件清零GO以停止重新触发

bit 2-0 **TMD<2:0>：**阈值中断模式选择位

111 = 无论阈值测试结果如何，始终中断

110 = ERR > UTH时中断

101 = ERR ≤ UTH时中断

100 = ERR < LTH或ERR>UTH时中断

011 = ERR > LTH且ERR < UTH时中断

010 = ERR ≥ LTH时中断

001 = ERR < LTH时中断

000 = 永不中断

注 1： PSIS = 0时，(RES - PREV)的值是表37-2的(S2 - S1)的值。

2： PSIS = 0时。

3： PSIS = 1时。

寄存器37-5： ADSTAT：ADC状态寄存器

R-0/0	R-0/0	R-0/0	R/HS/HC-0/0	U-0	R-0/0	R-0/0	R-0/0
AOV	UTHR	LTHR	MATH	—	STAT<2:0>		
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS/HC = 由硬件置1/清零位

bit 7 **AOV**：ADC累加器溢出位
1 = ADC累加器或ERR计算溢出
0 = ADC累加器和ERR计算未溢出

bit 6 **UTHR**：ADC模块大于上限阈值标志位
1 = ERR > UTH
0 = ERR ≤ UTH

bit 5 **LTHR**：ADC模块小于下限阈值标志位
1 = ERR < LTH
0 = ERR ≥ LTH

bit 4 **MATH**：ADC模块计算状态位
1 = 寄存器ACC、FLTR、UTH、LTH和AOV位正在更新或已更新
0 = 自该位上次清零以来，相关寄存器/位没有发生变化

bit 3 **未实现**：读为0

bit 2-0 **STAT<2:0>**：ADC模块周期多阶段状态位(1)
111 = ADC模块处于第二个转换阶段
110 = ADC模块处于第二个采集阶段
101 = ADC模块处于第二个预充电阶段
100 = 不使用
011 = ADC模块处于第一个转换阶段
010 = ADC模块处于第一个采集阶段
001 = ADC模块处于第一个预充电阶段
000 = ADC模块未在转换

注 1： 如果CS = 1，并且FOSC<FRC，这些位可能无效。

寄存器37-6： ADCLK：ADC时钟选择寄存器

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	CS<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现**：读为0

bit 5-0 **CS<5:0>**：ADC转换时钟选择位
111111 = FOSC/128
111110 = FOSC/126
111101 = FOSC/124
•
•
•
000000 = FOSC/2

寄存器37-7： ADREF：ADC参考选择寄存器

U-0	U-0	U-0	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	NREF	—	—	PREF<1:0>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现**：读为0

bit 4 **NREF**：ADC负参考电压选择位
1 = VREF-连接到外部VREF-
0 = VREF-连接到VSS

bit 3-2 **未实现**：读为0

bit 1-0 **PREF**：ADC正参考电压选择位
11 = VREF+连接到内部固定参考电压（FVR）模块
10 = VREF+连接到外部VREF+
01 = 保留
00 = VREF+连接到VDD

寄存器37-8： ADPCH：ADC正通道选择寄存器

U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	ADPCH<5:0>					
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-6 **未实现**：读为0

bit 5-0 **ADPCH<5:0>**：ADC正输入通道选择位

111111 = 固定参考电压（FVR）(2)
111110 = DAC1输出(1)
111101 = 温度指示器(3)
111100 = AVss（模拟地）
111011 = 保留。不连接任何通道
•
•
•
010111 = ANC7
010110 = ANC6
010101 = ANC5
010100 = ANC4
010011 = ANC3
010010 = ANC2
010001 = ANC1
010000 = ANC0
001111 = ANB7
001110 = ANB6
001101 = ANB5
001100 = ANB4
001011 = ANB3
001010 = ANB2
001001 = ANB1
001000 = ANB0
000111 = ANA7
000110 = ANA6
000101 = ANA5
000100 = ANA4
000011 = ANA3
000010 = ANA2
000001 = ANA1
000000 = ANA0

注 1： 更多信息，请参见**第38.0节“5位数模转换器（DAC）模块”**。
2： 更多信息，请参见**第35.0节“固定参考电压（FVR）”**。
3： 更多信息，请参见**第36.0节“温度指示器模块”**。

寄存器37-9： ADPREL：ADC预充电时间控制寄存器（低字节）

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
PRE<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **PRE<7:0>**：预充电时间选择位
请参见表37-4。

寄存器37-10： ADPREH：ADC预充电时间控制寄存器（高字节）

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	PRE<12:8>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现**：读为0
bit 4-0 PRE<12:8>：预充电时间选择位
请参见表37-4。

注： 如果PRE不等于0，则ADACQ = b'00000000意味着采集时间是256个所选ADC时钟。

表37-4： 预充电时间

ADPRE	预充电时间
1 1111 1111 1111	8191个所选ADC时钟
1 1111 1111 1110	8190个所选ADC时钟
1 1111 1111 1101	8189个所选ADC时钟
...	...
0 0000 0000 0010	2个所选ADC时钟
0 0000 0000 0001	1个所选ADC时钟
0 0000 0000 0000	不包含在数据转换周期中

寄存器37-11： ADACQL：ADC采集时间控制寄存器（低字节）

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
ACQ<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 ACQ<7:0>：采集（电荷共享时间）选择位
请参见表37-5。

寄存器37-12： ADACQH：ADC采集时间控制寄存器（高字节）

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	ACQ<12:8>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现**：读为0
bit 4-0 ACQ<12:8>：采集（电荷共享时间）选择位
请参见表37-5。

表37-5： 采集时间

ADACQ	采集时间
1 1111 1111 1111	8191个所选ADC时钟
1 1111 1111 1110	8190个所选ADC时钟
1 1111 1111 1101	8189个所选ADC时钟
...	...
0 0000 0000 0010	2个所选ADC时钟
0 0000 0000 0001	1个所选ADC时钟
0 0000 0000 0000	不包含在数据转换周期中(1)

注 1： 如果ADPRE不等于0，则ADACQ = 0b0_0000_0000_0000意味着采集时间是8192个所选ADC时钟。

寄存器37-13： ADCAP：ADC附加采样电容选择寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	ADCAP<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现**：读为0

bit 4-0 **ADCAP<4:0>**：ADC附加采样电容选择位
11111 = 31 pF
11110 = 30 pF
11101 = 29 pF
•
•
•
00011 = 3 pF
00010 = 2 pF
00001 = 1 pF
00000 = 无附加电容

寄存器37-14： ADRPT：ADC重复设置寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
RPT<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **RPT<7:0>**：ADC重复阈值位
计数ADC被触发的次数，并与CNT一起用于确定在计算处于低通滤波器模式、突发平均模式或平均模式下时何时检测误差阈值。更多详细信息，请参见表37-2。

寄存器37-15： ADCNT：ADC重复计数器寄存器

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
CNT<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **CNT<7:0>**：ADC重复计数位
确定在计算处于低通滤波器模式、突发平均模式或平均模式下时检测阈值之前ADC被触发的次数。更多详细信息，请参见表37-2。

寄存器37-16： ADFLTRH：ADC滤波器高字节寄存器

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
FLTR<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **FLTR<15:8>**：ADC滤波器输出最高有效位
在累加模式、平均模式和突发平均模式下，等于按ADCON2的ADCRS位右移的ACC。在LPF模式下，则为低通滤波器的输出。

寄存器37-17： ADFLTRL：ADC滤波器低字节寄存器

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
FLTR<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **FLTR<7:0>**：ADC滤波器输出最低有效位
在累加模式、平均模式和突发平均模式下，等于按ADCON2的ADCRS位右移的ACC。在LPF模式下，则为低通滤波器的输出。

寄存器37-18： ADRESH：ADC结果寄存器的高字节（FM = 0）

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADRES<11:4>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ADRES<11:4>**：ADC结果寄存器位
12位转换结果的高8位。

寄存器37-19： ADRESL：ADC结果寄存器的低字节（FM = 0）

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	U-0	U-0	U-0	U-0
ADRES<3:0>				—	—	—	—
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4 **ADRES<3:0>**：ADC结果寄存器位。12位转换结果的低4位。
bit 3-0 **保留**

寄存器37-20： ADRESH：ADC结果寄存器的高字节（FM = 1）

U-0	U-0	U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	—	—	ADRES<11:8>			
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-4 **保留**

bit 3-0 **ADRES<11:8>**：ADC采样结果位。12位转换结果的高4位。

寄存器37-21： ADRESL：ADC结果寄存器的低字节（FM = 1）

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
ADRES<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ADRES<7:0>**：ADC结果寄存器位。12位转换结果的低8位。

寄存器37-22： ADPREVH：ADC前一个结果寄存器

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
PREV<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **PREV<15:8>**：前一个ADC结果位
如果ADPSIS = 1：
当前的ADC转换开始时FLTR的高字节
如果ADPSIS = 0：
当前的ADC转换开始时ADRES的高位(1)

注 1： 如果ADPSIS = 0，则ADPREVH和ADPREVL采用与ADRES相同的格式，具体取决于FM位。

寄存器37-23： ADPREVL：ADC前一个结果寄存器

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
PREV<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **PREV<7:0>**：前一个ADC结果位
如果ADPSIS = 1：
当前的ADC转换开始时FLTR的低字节
如果ADPSIS = 0：
当前的ADC转换开始时ADRES的低位(1)

注 1： 如果ADPSIS = 0，则ADPREVH和ADPREVL采用与ADRES相同的格式，具体取决于FM位。

寄存器37-24： ADACCU：ADC累加器最高字节寄存器

U-0	U-0	U-0	U-0	U-0	U-0	R/W-x/x	R/W-x/x
—	—	—	—	—	—	ACC<17:16>	
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-2 **未实现**：读为0

bit 1-0 **ACC<17:16>**：ADC累加器MSB。累加器值的最高两位。更多详细信息，请参见表37-2。

寄存器37-25： ADACCH：ADC累加器高字节寄存器

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
ACC<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ACC<15:8>**：ADC累加器中间位。累加器值的中间8位。更多详细信息，请参见表37-2。

寄存器37-26： ADACCL：ADC累加器低字节寄存器

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
ACC<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ACC<7:0>**：ADC累加器LSB。累加器值的低8位。更多详细信息，请参见表37-2。

寄存器37-27： ADSTPTH：ADC阈值设定值高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
STPT<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **STPT<15:8>**：ADC阈值设定值MSB。ADC阈值设定值的高字节，可根据ADCALC用于确定ERR。更多详细信息，请参见寄存器37-29。

寄存器37-28： ADSTPTL：ADC阈值设定值低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
STPT<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **STPT<7:0>**：ADC阈值设定值LSB。ADC阈值设定值的低字节，可根据ADCALC用于确定ERR。更多详细信息，请参见寄存器37-30。

寄存器37-29： ADERRH：ADC设定值误差高字节寄存器

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
ERR<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ERR<15:8>**：ADC设定值误差MSB。ADC设定值误差的高字节。设定值误差计算结果由ADCON3的CALC位确定。更多详细信息，请参见寄存器37-4。

寄存器37-30： ADERRL：ADC设定值误差低字节寄存器

R-x	R-x	R-x	R-x	R-x	R-x	R-x	R-x
ERR<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **ERR<7:0>**：ADC设定值误差LSB。ADC设定值误差计算结果的低字节由ADCON3的CALC位确定。更多详细信息，请参见寄存器37-4。

寄存器37-31： ADLTHH：ADC下限阈值高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
LTH<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **LTH<15:8>**：ADC下限阈值MSB。将LTH和UTH与ERR进行比较以设置ADSTAT的ADUTHR和ADLTHR位。该比较的结果可能会触发中断，具体取决于ADTMD的设置。

寄存器37-32： ADLTHL：ADC下限阈值低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
LTH<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **LTH<7:0>**：ADC 下限阈值LSB。将LTH 和UTH 与ERR 进行比较以设置ADSTAT 的ADUTHR 和ADLTHR位。该比较的结果可能会触发中断，具体取决于ADTMD的设置。

寄存器37-33： ADUTHH：ADC上限阈值高字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
UTH<15:8>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **UTH<15:8>**：ADC 上限阈值MSB。将LTH 和UTH 与ERR 进行比较以设置ADSTAT 的ADUTHR 和ADLTHR位。该比较的结果可能会触发中断，具体取决于ADTMD的设置。

寄存器37-34： ADUTHL：ADC上限阈值低字节寄存器

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
UTH<7:0>							
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-0 **UTH<7:0>**：ADC 上限阈值LSB。将LTH 和UTH 与ERR 进行比较以设置ADSTAT 的ADUTHR 和ADLTHR位。该比较的结果可能会触发中断，具体取决于ADTMD的设置。

寄存器37-35： ADACT：ADC自动转换触发控制寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	ACT<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现**：读为0

bit 4-0 **ADACT<4:0>**：自动转换触发选择位

11111 = 保留，不要使用
•
•
•
11110 = 保留，不要使用
11101 = 软件写入ADPCH
11100 = 保留，不要使用
11011 = 软件读取ADRESH
11010 = 软件读取ADERRH
11001 = CLC4_out
11000 = CLC3_out
10111 = CLC2_out
10110 = CLC1_out
10101 = 所有电平变化中断标志的逻辑或运算结果
10100 = CMP2_out
10011 = CMP1_out
10010 = NCO1_out
10001 = PWM8_out
10000 = PWM7_out
01111 = PWM6_out
01110 = PWM5_out
01101 = CCP4_trigger
01100 = CCP3_trigger
01011 = CCP2_trigger
01010 = CCP1_trigger
01001 = SMT1_trigger
01000 = TMR6_postscaled
00111 = TMR5_overflow
00110 = TMR4_postscaled
00101 = TMR3_overflow
00100 = TMR2_postscaled
00011 = TMR1_overflow
00010 = TMR0_overflow
00001 = 通过ADACTPPS选择的引脚
00000 = 禁止外部触发

寄存器37-36： ADCP：ADC电荷泵控制寄存器

R/W-0/0	U-0	U-0	U-0	U-0	U-0	U-0	R-0/0
CPON	—	—	—	—	—	—	CPRDY
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	HS = 硬件置1

bit 7 **CPON**：电荷泵开启控制位
1 = ADC请求时电荷泵开启
0 = 电荷泵关闭

bit 6-1 **未实现**：读为0

bit 0 **CPRDY**：电荷泵就绪状态位
1 = 电荷泵已就绪
0 = 电荷泵未就绪（或从未启动）

表37-6： 与ADC相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
ADCON0	ON	CONT	—	CS	—	FM	—	GO	671
ADCON1	ADPPOL	ADIPEN	ADGPOL	—	—	—	—	ADDSEN	672
ADCON2	ADPSIS	ADCRS<2:0>			ADACLR	MD<2:0>			673
ADCON3	—	ADCALC<2:0>			ADSOI	ADTMD<2:0>			674
ADSTAT	ADAOV	ADUTHR	ADLTHR	ADMATH	ADSTAT<3:0>				675
ADCLK	—	—	CS<5:0>						676
ADREF	—	—	—	ADNREF	—	—	ADPREF<1:0>		676
ADPCH	—	—	ADPCH<5:0>						677
ADPREL	PRE<7:0>								678
ADPREH	—	—	—	PRE<12:8>					678
ADACQL	ACQ<7:0>								679
ADCAP	—	—	—	ADCAP<4:0>					680
ADRPT	RPT<7:0>								680
ADCNT	CNT<7:0>								681
ADFLTRL	FLTR<7:0>								681
ADFLTRH	FLTR<15:8>								681
ADRESL	ADRESL<7:0>								682, 683
ADRESH	ADRESH<7:0>								682, 683
ADPREVH	PREV<15:8>								684
ADPREVL	PREV<7:0>								684
ADACCH	ACC<15:8>								685
ADACCL	ACC<7:0>								685
ADACCU	—	—	—	—	—	—	ACC<17:16>		685
ADSTPTL	STPT<7:0>								686
ADSTPTH	STPT<15:8>								686
ADERRL	ERR<7:0>								687
ADERRH	ERR<15:8>								687
ADLTHH	LTH<15:8>								687
ADLTHL	LTH<7:0>								688
ADUTHH	UTH<15:8>								688
ADUTHL	UTH<7:0>								688
ADERRL	ERR<15:8>								687
ADACT	—	—	—	—	ADACT<4:0>				673
ADCP	CPON	—	—	—	—	—	—	CPRDY	690

图注： – = 未实现，读为0。ADC模块不使用阴影单元。

38.0 5位数模转换器（DAC）模块

数模转换器提供了一个可变参考电压，它与输入源成比例，具有32个可选输出电压。

DAC的正输入源（VSOURCE+）可以连接到：

- FVR缓冲区
- 外部VREF+引脚
- VDD供电电压

DAC的负输入源（VSOURCE-）可以连接到：

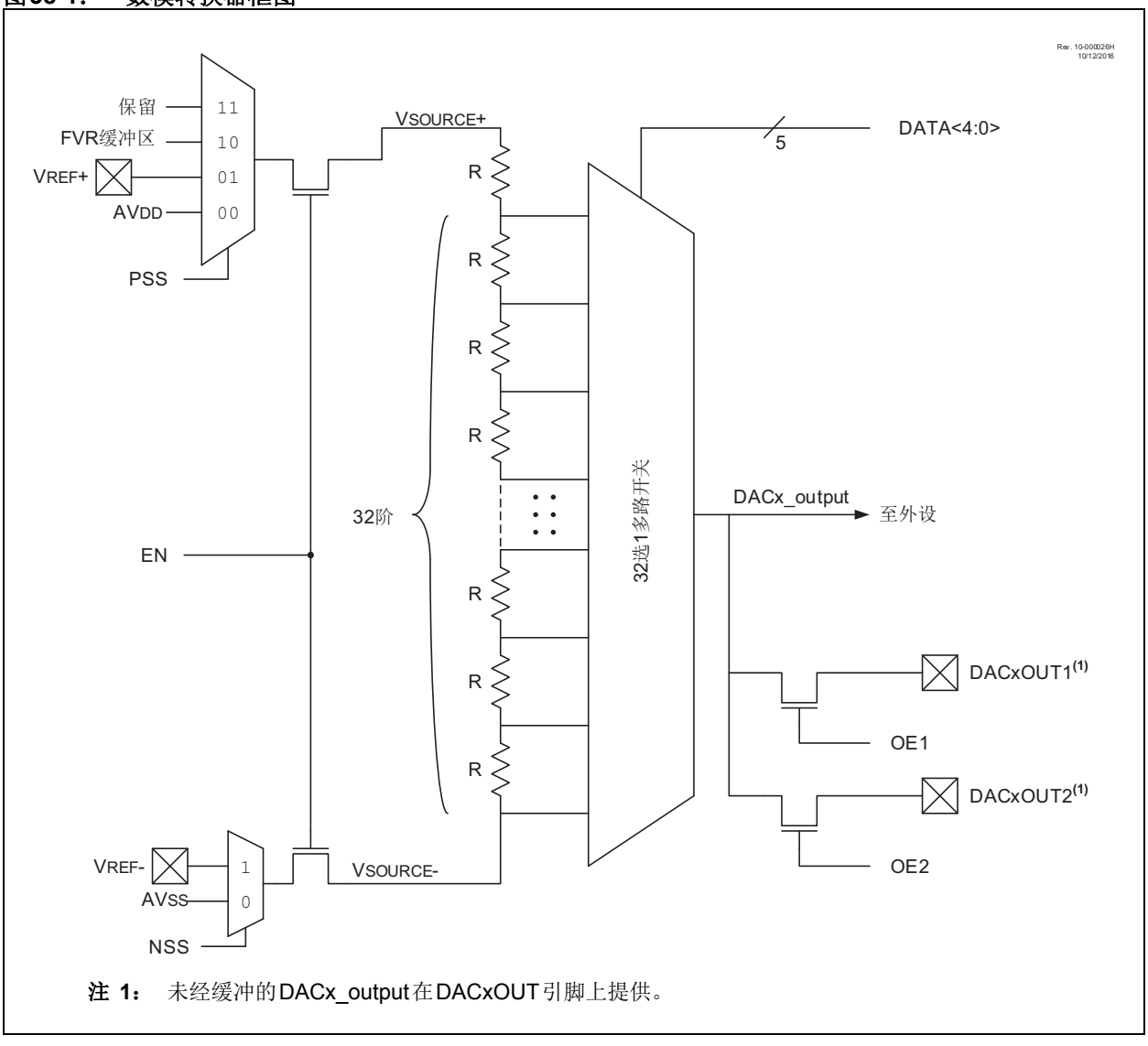
- 外部VREF-引脚
- Vss

DAC的输出（DAC1_output）可选作以下各项的参考电压：

- 比较器正输入
- ADC输入通道
- DAC1OUT1引脚
- DAC1OUT2引脚

可通过将DAC1CON0寄存器的EN位置1来使能数模转换器（DAC）。

图38-1： 数模转换器框图



注 1： 未经缓冲的DACx_output在DACxOUT引脚上提供。

38.1 输出电压选择

DAC 具有32 个输出电压可供选择。这32 个电压通过DAC1CON1 寄存器的DATA<4:0>位进行设置。

可以使用公式38-1 来确定DAC输出电压。

38.2 比例输出电压

DAC输出值通过使用一个梯形电阻网络产生，梯形电阻网络的每一端分别与正参考电压和负参考电压输入源连接。如果任一输入源的电压发生波动，DAC输出值中会产生类似的波动。

表45-16中给出了梯形电阻网络中各个电阻的阻值。

38.3 DAC参考电压输出

通过将DAC1CON0寄存器中的DACOEn位置1，可以将未经缓冲的DAC 电压输出到相应的DAC1OUTn 引脚。选择将DAC参考电压输出到DAC1OUTn引脚会自动改写数字输出缓冲器，以及该引脚的弱上拉和数字输入阈值检测器功能。

当DAC1OUTn引脚已被配置为DAC参考电压输出时，读取该引脚将始终返回0。

注： 不应将未经缓冲的DAC输出（DAC1OUTn）用于驱动外部负载。

38.4 休眠期间的操作

如果因中断或窗口看门狗定时器超时将器件从休眠模式唤醒，DAC1CON0寄存器的内容将不受影响。要降低休眠模式下的电流消耗，应禁止参考电压。

38.5 复位的影响

器件复位会产生以下影响：

- DAC1被禁止。
- DAC1输出电压从DAC1OUTn引脚上被移除。
- DAC1R<4:0>范围选择位被清零。

公式38-1： DAC输出电压

如果 DACEN = 1

$$DACx\_output = \left((V_{REF+} - V_{REF-}) \times \frac{DATA[4:0]}{2^5} \right) + V_{REF-}$$

注： 有关可用$V_{SOURCE+}$和$V_{SOURCE-}$选择的信息，请参见DAC1CON0寄存器。

38.6 寄存器定义：DAC控制

DAC外设的长位名称前缀如下所示。更多信息，请参见第1.3.2.2节“长位名称”。

外设	位名称前缀
DAC1	DAC1

寄存器38-1： DAC1CON0：DAC控制寄存器

R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0
EN	—	OE1	OE2	PSS<1:0>		—	NSS
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7 **EN**：DAC使能位
1 = 使能DAC
0 = 禁止DAC(1)

bit 6 **未实现**：读为0

bit 5 **OE1**：DAC电压输出使能位
1 = DAC电压从DAC1OUT1引脚输出
0 = DAC电压从DAC1OUT1引脚断开

bit 4 **OE2**：DAC电压输出使能位
1 = DAC电压从DAC1OUT2引脚输出
0 = DAC电压从DAC1OUT2引脚断开

bit 3-2 **PSS<1:0>**：DAC正电压源选择位
11 = 保留
10 = FVR缓冲区2
01 = V_{REF+}
00 = V_{DD}

bit 1 **未实现**：读为0

bit 0 **NSS**：DAC负电压源选择位
1 = V_{REF-}
0 = V_{SS}

注 1： DAC1OUTx输出引脚仍有效。

寄存器38-2： DAC1CON1：DAC数据寄存器

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	DATA<4:0>				
bit 7							bit 0

图注：		
R = 可读位	W = 可写位	U = 未实现位，读为0
u = 不变	x = 未知	-n/n = POR和BOR时的值/所有其他复位时的值
1 = 置1	0 = 清零	

bit 7-5 **未实现：**读为0

bit 4-0 **DATA<4:0>：**DAC数据输入寄存器位

表38-1： 与DAC模块相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
DAC1CON0	EN	—	OE1	OE2	PSS<1:0>		—	NSS	694
DAC1CON1	—	—	—	DATA<4:0>					695

图注： — = 未实现位，读为0。DAC模块不使用阴影单元。

39.0 比较器模块

注： PIC18(L)F25/26K83器件有两个比较器。因此，本章中的所有信息同时适用于C1和C2。

比较器通过比较两个模拟电压并提供其相对幅值的数字表示，用于建立模拟电路与数字电路的接口。比较器是非常有用的混合信号模块，因为它们提供了与程序执行相独立的模拟功能。

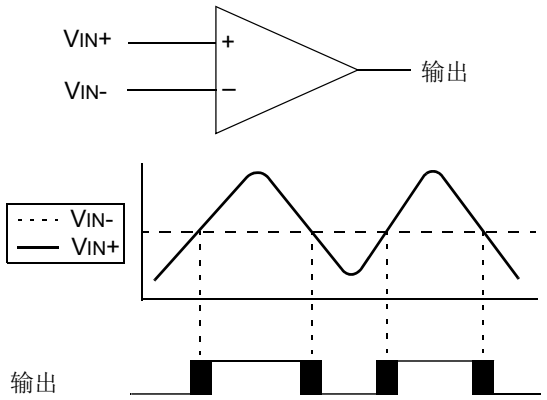
模拟比较器模块具有以下特性：

- 可编程输入选择
- 可编程输出极性
- 输出上升沿/下降沿中断

39.1 比较器概述

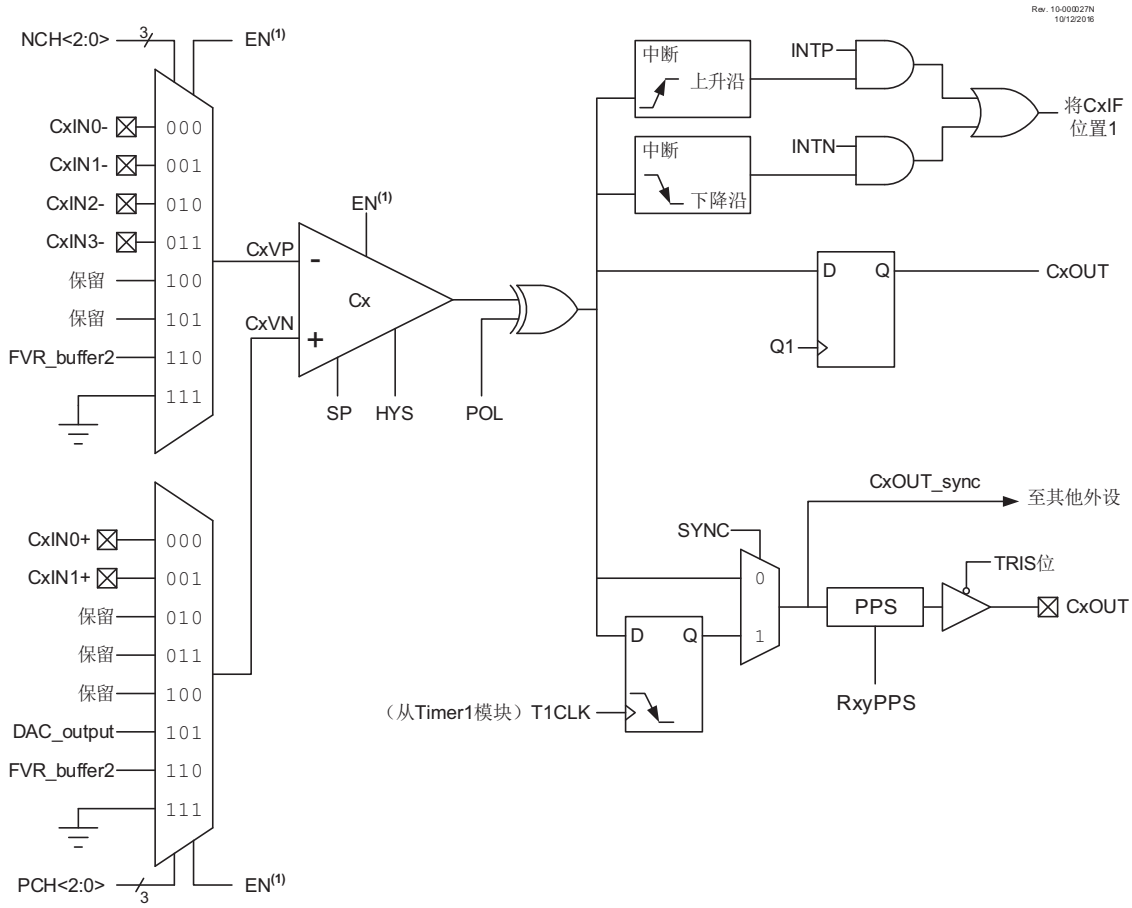
图39-1所示为单比较器以及模拟输入电压与数字输出之间的关系。当VIN+上的模拟电压小于VIN-上的模拟电压时，比较器输出为数字低电平。当VIN+上的模拟电压大于VIN-上的模拟电压时，比较器输出为数字高电平。

图39-1： 单个比较器



注： 比较器输出的黑色区域表示因输入失调电压和响应时间所造成的输出不确定区域。

图39-2： 比较器模块简化框图



注1： 当CxON = 0时，断开所有多路开关输入，比较器输出为0。

39.2 比较器控制

每个比较器都具有2个控制寄存器：CMxCON0和CMxCON1。

CMxCON0寄存器（见寄存器39-1）包含以下控制和状态位：

- 使能
- 输出
- 输出极性
- 滞后使能
- Timer1输出同步

CMxCON1寄存器（见寄存器39-2）包含以下控制位：

- 正/负边沿中断允许

CMxPCH和CMxNCH寄存器分别用于选择正、负输入通道。

39.2.1 比较器使能

将CMxCON0寄存器的EN位置1可以使能比较器操作。清零EN位可以禁止比较器，以使电流消耗降至最低。

39.2.2 比较器输出

可以通过读CMxCON0寄存器的CxOUT位或CMOUT寄存器的CxOUT位监视比较器的输出。

比较器输出还可以通过RxyPPS寄存器（寄存器17-2）输送到外部引脚。要使能引脚作为输出，必须清零相应的TRIS位。

注 1：	比较器的内部输出在每个指令周期被锁存。除非另外指定，否则不锁存外部输出。

39.2.3 比较器输出极性

将比较器的输出反相在功能上等效于交换比较器输入。可以通过将CMxCON0寄存器的POL位置1来使比较器输出的极性反相。清零POL位时输出同相。

表39-1给出了输出状态与输入条件的关系（包括极性控制）。

表39-1： 比较器输出状态与输入条件

输入条件	POL	CxOUT
CxVN > CxVP	0	0
CxVN < CxVP	0	1
CxVN > CxVP	1	1
CxVN < CxVP	1	0

39.3 比较器滞后

通过在每个比较器的输入引脚上施加不同的正向和负向增长阈值电压，可以为整体操作提供滞后功能。滞后功能通过将CMxCON0寄存器的HYS位置1来使能。

更多信息，请参见表45-15的比较器规范。

39.3.1 比较器输出同步

通过将CMxCON0寄存器的SYNC位置1，可以将比较器的输出与Timer1同步。

使能比较器的输出时，比较器的输出在Timer1时钟源的下降沿被锁存。如果使用预分频器，则CxOUT位与定时器同步，这样软件就不会遇到因定时而造成的模糊问题。更多信息，请参见比较器框图（图39-2）和Timer1框图（图21-1）。

39.4 比较器中断

可针对比较器输出的每个上升沿或下降沿生成中断。

当触发任一边沿检测器时，如果它相关的允许位已置1（CMxCON1寄存器的INTP和/或INTN位），则相应的中断标志位（相应PIR寄存器的CxIF位）会置1。

要允许中断，必须将以下位置1：

- CMxCON0寄存器的EN位
- 相应PIE寄存器的CxIE位
- CMxCON1寄存器的INTP位（用于上升沿检测）
- CMxCON1寄存器的INTN位（用于下降沿检测）
- INTCON0寄存器的GIE位

相关的中断标志位（相应PIR寄存器的CxIF位）必须用软件清零。如果在清零该标志时检测到另一个边沿，则标志仍然会在序列结束时置1。

注：	即使比较器被禁止，还是可以通过使用CMxCON0寄存器的POL位更改输出极性来产生中断，或者通过使用CMxCON0寄存器的EN位开启或关闭比较器来产生中断。

39.5 比较器同相输入选择

配置CMxPCH寄存器的PCH<2:0>位可将内部参考电压或模拟引脚连接到比较器的同相输入：

- CxIN0+和CxIN1+模拟引脚
- DAC输出
- 固定参考电压（FVR）
- Vss（地）

关于固定参考电压模块的更多信息，请参见**第35.0节“固定参考电压（FVR）”**。

关于DAC输入信号的更多信息，请参见**第38.0节“5位数模转换器（DAC）模块”**。

每当禁止比较器（EN = 0）时，所有比较器输入都会被禁止。

39.6 比较器反相输入选择

CMxNCH寄存器的NCH<2:0>位可将模拟输入引脚和内部参考电压或模拟地连接到比较器的反相输入：

- CxIN0-、CxIN1-、CxIN2-和CxIN3-模拟引脚
- 固定参考电压（FVR）
- 模拟地

注：	要将CxINy+和CxINy-引脚用作模拟输入，必须将ANSEL寄存器中的相应位置1，同时也必须将相应的TRIS位置1来禁止输出驱动器。

39.7 比较器响应时间

在改变输入源或选择新的参考电压后的一段时间内，比较器的输出状态都是不确定的。这段时间被称为响应时间。比较器的响应时间不同于参考电压的稳定时间。因此，在确定比较器输入改变的总响应时间时，必须考虑这两个时间。更多详细信息，请参见表45-15和表45-17中的比较器和参考电压规范。

39.8 模拟输入连接注意事项

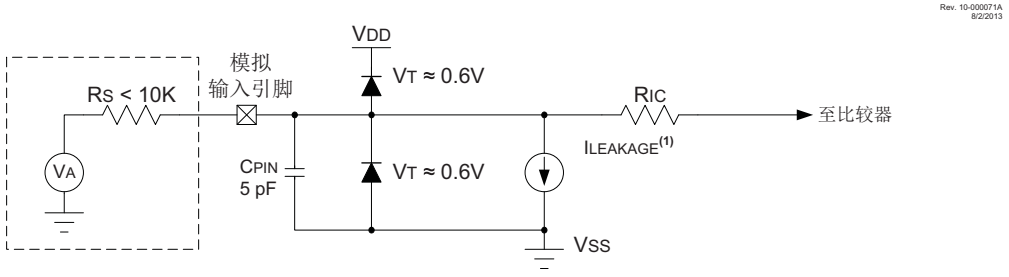
模拟输入的简化电路如图39-3所示。由于模拟输入引脚与数字输入共用连接，它们与VDD和VSS之间连有反向偏置的ESD保护二极管。因此，模拟输入的值必须在VSS和VDD之间。如果输入电压与这一范围偏离的绝对值超过0.6V，就可能发生一个二极管正向偏置，从而可能导致锁死发生。

模拟信号源的最大阻抗推荐值为10 kΩ。任何连接到模拟输入引脚的外部元件（如电容或齐纳二极管），应保证其泄漏电流极小以使引入的误差降至最小。

注 1： 当读端口寄存器时，所有配置为模拟输入的引脚都读为0。配置为数字输入的引脚将按照输入规范转换为模拟输入。

2： 定义为数字输入的引脚上的模拟电压可能会使输入缓冲器的电流消耗超过规定值。

图39-3： 模拟输入模型



图注：

CPIN	=	输入电容
ILEAKAGE	=	各个连接点在引脚上产生的泄漏电流
RIC	=	内部走线等效电阻
RS	=	信号源阻抗
VA	=	模拟电压
VT	=	阈值电压

注 1： 请参见**第45.0节“电气规范”**。

39.9 CWG1 自动关断源

比较器模块的输出可用作CWG1模块的自动关断源。当比较器的输出有效且相应的WGASxE使能时，CWG操作立即暂停（见**第26.10.1.2节“外部输入源”**）。

39.10 ADC 自动触发源

比较器模块的输出可用于触发ADC转换。如果将ADACT寄存器设置为通过比较器输出触发，则比较器输出变为高电平时将触发ADC转换。

39.11 TMR2/4/6复位

比较器模块的输出可用于复位Timer2。如果对TxERS寄存器进行了相应设置，则当比较器输出变为高电平时，定时器将复位。

39.12 休眠模式下的操作

比较器模块可以在休眠模式下工作。比较器时钟源基于Timer1时钟源。如果Timer1时钟源为系统时钟（FOSC）或指令时钟（FOSC/4），Timer1将不会在休眠期间工作，并且同步比较器输出将无法运行。

比较器中断可以将器件从休眠模式唤醒。相应PIE寄存器的CxIE位必须置1才能允许比较器中断。

39.13 寄存器定义：比较器控制

表39-2给出了比较器的长位名称前缀。更多信息，请参见**第1.3.2.2节“长位名称”**。

表39-2：

外设	位名称前缀
C1	C1
C2	C2

寄存器39-1： CMxCON0：比较器x控制寄存器0

R/W-0/0	R-0/0	U-0	R/W-0/0	U-0	U-1	R/W-0/0	R/W-0/0
EN	OUT	—	POL	—	—	HYS	SYNC
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **EN：**比较器使能位
1 = 使能比较器
0 = 禁止比较器，不消耗有功功率

bit 6 **OUT：**比较器输出位
如果POL = 0（极性同相）：
1 = CxVP > CxVN
0 = CxVP < CxVN
如果POL = 1（极性反相）：
1 = CxVP < CxVN
0 = CxVP > CxVN

bit 5 **未实现：**读为0

bit 4 **POL：**比较器输出极性选择位
1 = 比较器输出反相
0 = 比较器输出同相

bit 3 **未实现：**读为0

bit 2 **未实现：**读为1

bit 1 **HYS：**比较器滞后使能位
1 = 使能比较器滞后
0 = 禁止比较器滞后

bit 0 **SYNC：**比较器输出同步模式位
1 = 送到Timer1/3/5和I/O引脚的比较器输出与Timer1时钟源的变化同步。
0 = 送到Timer1/3/5和I/O引脚的比较器输出是异步的
输出在Timer1/3/5时钟源的下降沿进行更新。

寄存器39-2： CMxCON1：比较器x控制寄存器1

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0
—	—	—	—	—	—	INTP	INTN
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-2 未实现：读为0

bit 1 **INTP**：比较器正向边沿中断允许位
1 = 在CxOUT位的正向边沿，CxIF中断标志将置1
0 = 在CxOUT位的正向边沿，中断标志不会置1

bit 0 **INTN**：比较器负向边沿中断允许位
1 = 在CxOUT位的负向边沿，CxIF中断标志将置1
0 = 在CxOUT位的负向边沿，中断标志不会置1

寄存器39-3： CMxNCH：比较器x反相通道选择寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	NCH<2:0>		
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-3 **未实现**：读为0

bit 2-0 **NCH<2:0>**：比较器反相输入通道选择位
111 = Vss
110 = FVR_Buffer2
101 = NCH未连接
100 = NCH未连接
011 = CxIN3-
010 = CxIN2-
001 = CxIN1-
000 = CxIN0-

寄存器39-4： CMxPCH：比较器x同相通道选择寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	PCH<2:0>		
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-3 **未实现：**读为0

bit 2-0 **PCH<2:0>：**比较器同相输入通道选择位
111 = Vss
110 = FVR_Buffer2
101 = DAC_Output
100 = PCH未连接
011 = PCH未连接
010 = PCH未连接
001 = CxIN1+
000 = CxIN0+

寄存器39-5： CMOUT：比较器输出寄存器

U-0	U-0	U-0	U-0	U-0	U-0	R-0/0	R-0/0
—	—	—	—	—	—	C2OUT	C1OUT
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7-2 **未实现：**读为0

bit 1 **C2OUT：**C2OUT的镜像副本位

bit 0 **C1OUT：**C1OUT的镜像副本位

表39-3： 与比较器模块相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
CMxCON0	EN	OUT	—	POL	—	—	HYS	SYNC	702
CMxCON1	—	—	—	—	—	—	INTP	INTN	703
CMxNCH	—	—	—	—	—	NCH<2:0>			703
CMxPCH	—	—	—	—	—	PCH<2:0>			704
CMOUT	—	—	—	—	—	—	C2OUT	C1OUT	704

图注： — = 未实现，读为0。比较器模块不使用阴影单元。

40.0 高/低电压检测（HLVD）

PIC18(L)F25/26K83 系列器件配有一个高/低电压检测（HLVD）模块。该模块是一个可编程的电路，用于设置器件电压跳变点和从该点开始的电压变化方向（正向、负向或两个方向）。如果器件电压按照指定的方向相对于该跳变点发生了偏离，就会将中断标志置1。如果允许了中断，程序将跳转到中断向量地址处执行，由软件响应该中断。

HLVD 模块完全由 HLVDCON0 和 HLVDCON1 寄存器控制。这样，用户便可通过软件控制“关闭”电路，从而将器件的电流消耗降至最低。

图40-1给出了该模块的框图。

由于 HLVD 可由软件通过 EN 位使能，因此置 1 和清零使能位不会导致虚假的 HLVD 事件毛刺。每次使能 HLVD 模块时，电路都需要一段时间达到稳定。RDY 位（HLVDCON0<4>）是一个只读位，用于指示带隙参考电压何时达到稳定。

该模块开启后需等待带隙参考电压准备就绪才能生成中断。

INTH 和 INTL 位决定模块的整体操作。当 INTH 置 1 时，模块会监视 V_{DD} 是否上升到 HLVDCON1 寄存器设置的跳变点之上。当 INTL 置 1 时，模块会监视 V_{DD} 是否下降到 HLVDCON1 寄存器设置的跳变点之下。当 INTH 和 INTL 位均置 1 时，可以同时监视电压是否上升/下降到 HLVDCON1 寄存器设置的跳变点之上/之下。

通过读取 OUT 位，可以确定电压是高于还是低于 HLVDCON1 寄存器选择的电压。

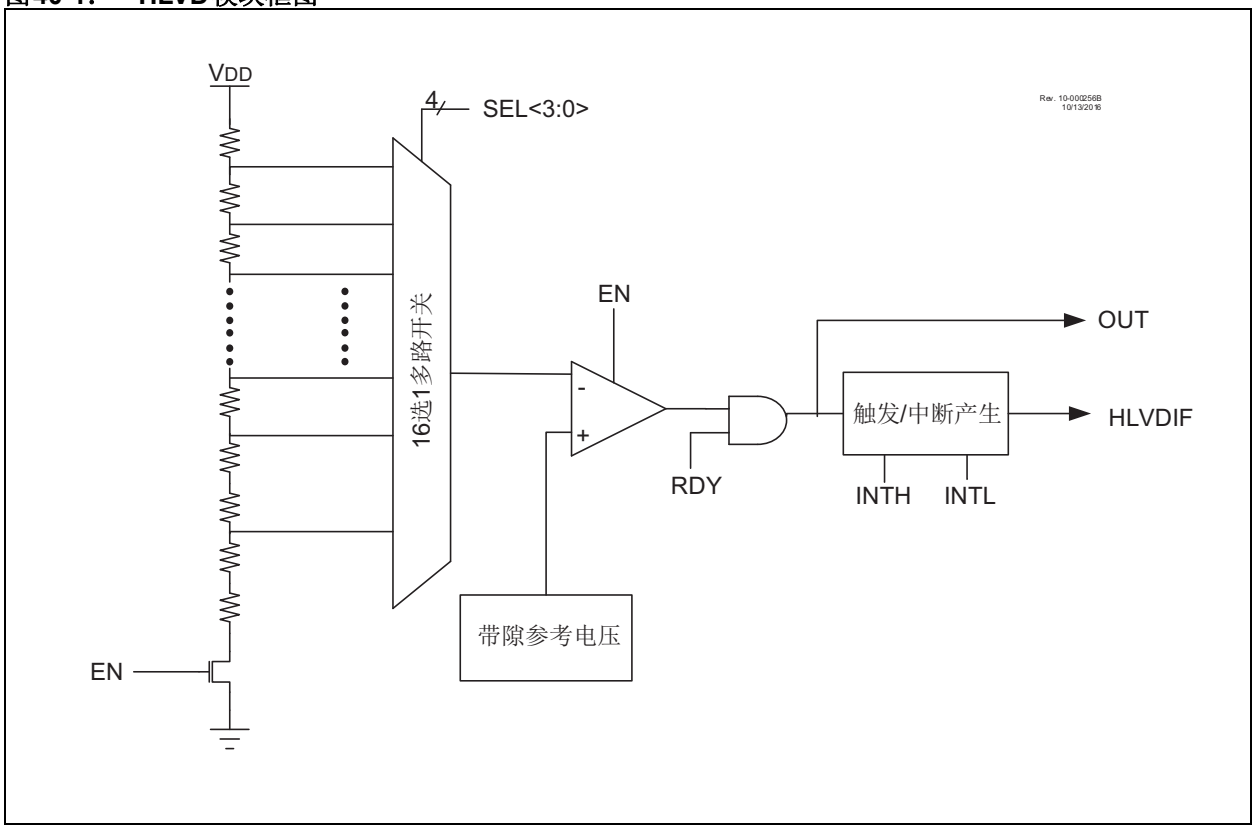
40.1 工作原理

使能HLVD模块后，比较器使用内部产生的参考电压作为设定点。将设定点的电压与跳变点的电压进行比较，其中电阻分压器中的每个节点均代表一个电压跳变点。“跳变点”电压是指器件检测到高电压或低电压事件时的电压，具体取决于模块的配置。

当供电电压等于跳变点电压时，电阻阵列的节点电压输出值等于由参考电压模块产生的内部参考电压。然后比较器通过将HLVDIF位置1产生一个中断信号。

跳变点电压可由软件编程为SEL<3:0>位（HLVDCON1<3:0>）中的任何一个。

图40-1： HLVD模块框图



40.2 HLVD设置

要设置HLVD模块，需按以下步骤操作：

1. 通过将值写入HLVDCON1寄存器的SEL<3:0>位选择所需的HLVD跳变点。
2. 根据应用需求（即检测高电压峰值、低电压谷值还是两者兼之），相应地将INTH或INTL位置1。
3. 通过将EN位置1，使能HLVD模块。
4. 清零HLVD中断标志（PIR2寄存器），该标志可能被先前的中断置1。
5. 如果需要中断，可通过将PIE2寄存器中的HLVDIE和GIE位置1来允许HLVD中断。

   直到RDY位也置1时才会产生中断。

注： 在更改任何模块设置（INTH、INTL或SEL<3:0>）之前，需先禁止模块（EN = 0），然后再进行更改并重新使能模块。这样可以防止产生虚假HLVD事件。

40.3 电流消耗

使能了该模块就使能了HLVD比较器和分压器，并将消耗静态电流。电气规范参数**D206**（表45-3）给出了使能该模块时的总电流消耗。

HLVD模块无需一直工作，具体取决于应用。要降低电流消耗，只需要在检测电压时短时间使能HLVD电路，在检测完成之后可以禁止该模块。

40.4 HLVD启动时间

电气规范（表45-17）规定了HLVD模块的内部参考电压，该参考电压也可供其他内部电路（如可编程欠压复位电路）使用。如果禁止了HLVD或其他使用参考电压的电路以降低器件的电流消耗，则参考电压电路将需要一段时间稳定下来之后才能可靠地检测低电压或高电压条件。该启动时间TFVRST与器件时钟速度无关，由电气规范（表45-17）规定。

直到TFVRST结束且参考电压达到稳定后才允许HLVD中断标志。因此，在此时间间隔内，可能无法检测到超出设定点的短暂偏离（见图40-2或图40-3）。

图40-2： 低电压检测工作原理（INTL = 1）

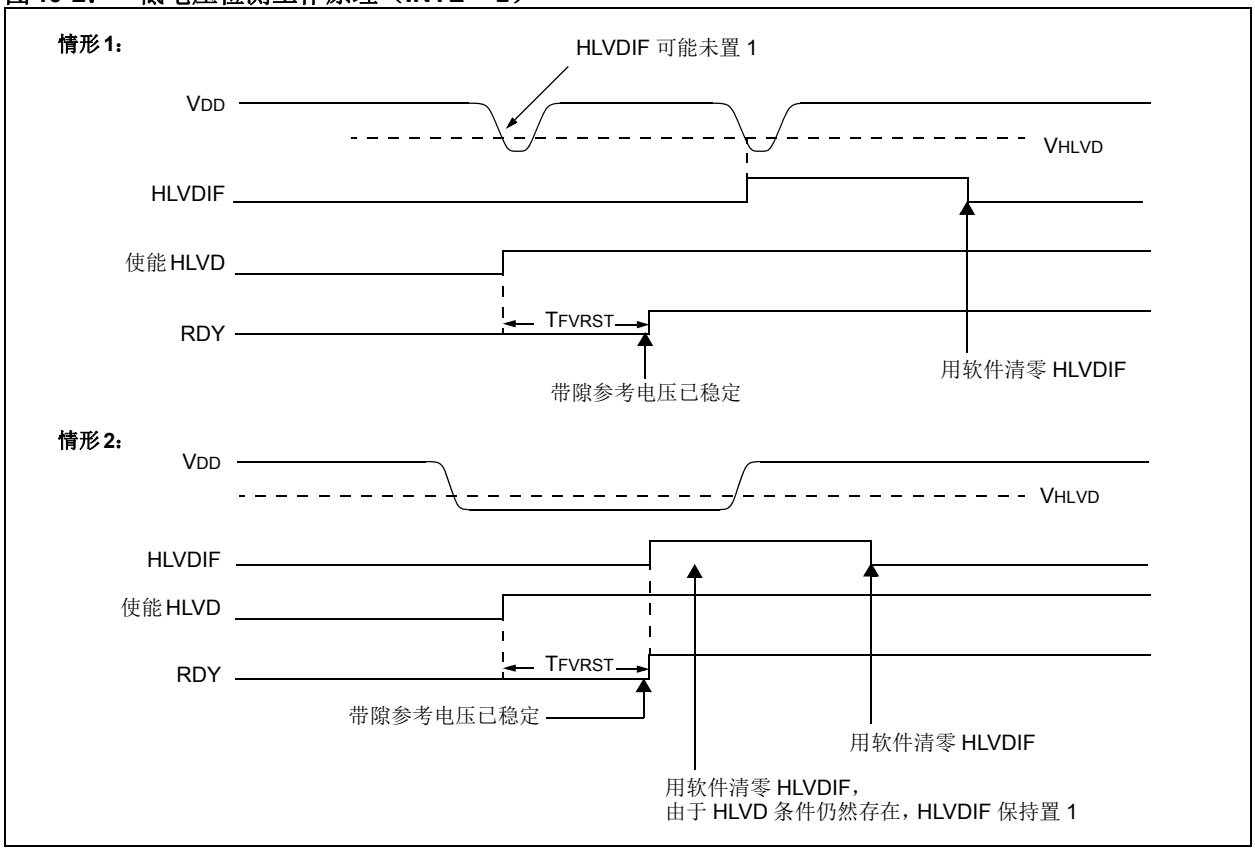
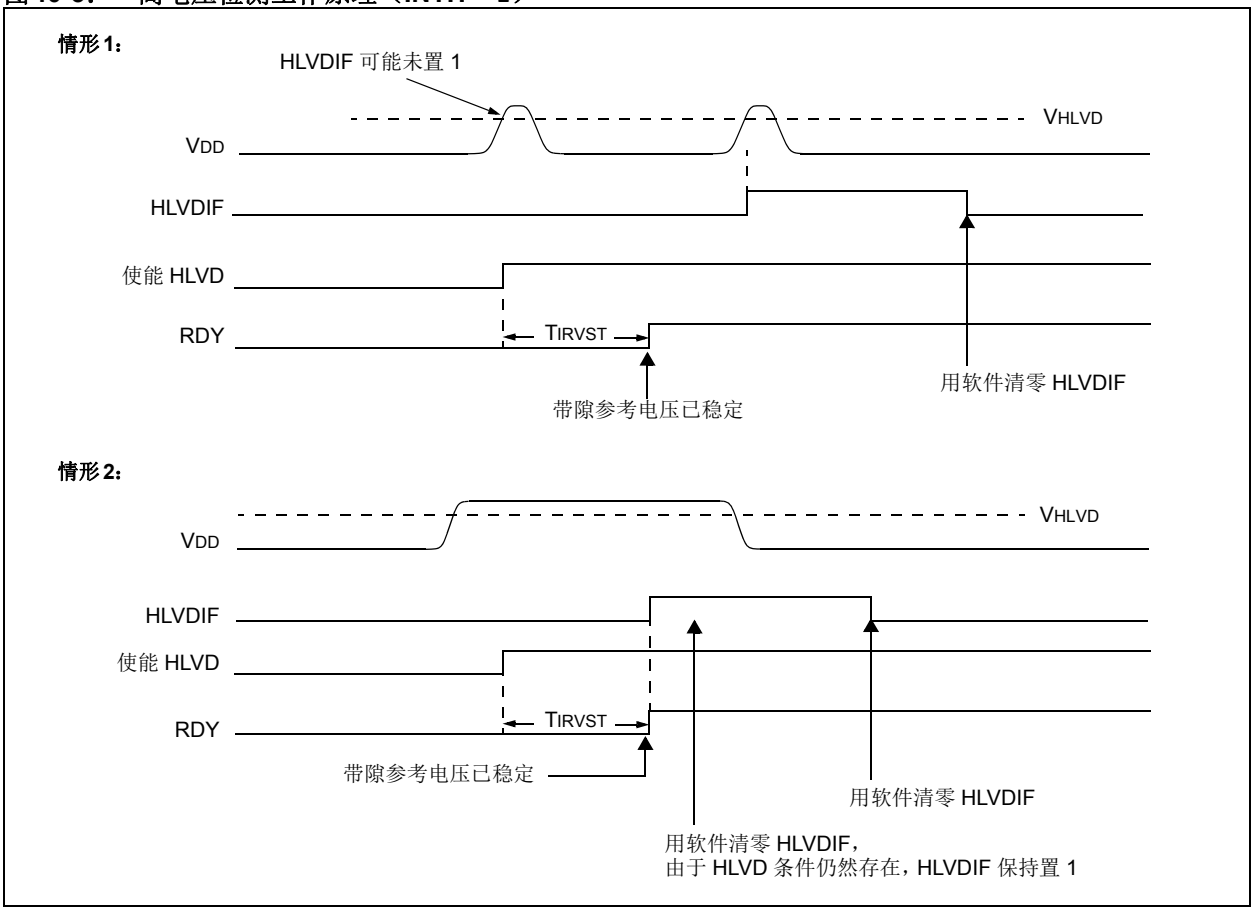


图40-3： 高电压检测工作原理（INTH = 1）

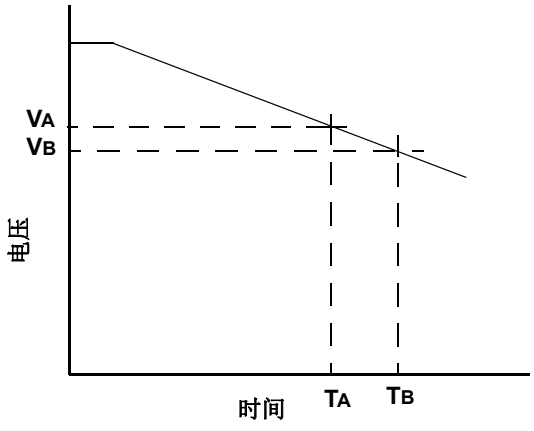


40.5 应用

在许多应用中，当电压低于或高于某个电压阈值时，系统希望可以检测到该事件。例如，可以定期使能HLVD模块来检测是否连接通用串行总线（Universal Serial Bus，USB）。这里假定断开连接时器件的供电电压低于USB电压。如果连接了USB，将检测到3.3V到5V的高电压（USB上的电压）；如果断开连接，情况正好相反。凭借此功能，可以在设计中省去一些额外的元件和连接信号（输入引脚）。

对于一般的电池应用，图40-4给出了一条可能的电压曲线。器件电压会随时间逐渐下降。当器件电压达到电压VA时，HLVD逻辑电路会在时间TA产生中断。该中断将导致执行中断服务程序（ISR），从而使应用程序能在器件电压退出有效工作范围（对应的时间为TB）之前执行“日常任务”，并执行受控关闭。这会提供一个时间窗（表示为TA和TB的时间差）使应用程序能安全地退出。

图40-4： 典型低电压检测应用



图注： VA = HLVD 跳变点
VB = 器件的最低有效工作电压

40.6 休眠期间的操作

如果使能了HLVD电路，则其在休眠期间将继续工作。如果器件电压越过了跳变点，HLVDIF位将会被置1并且器件将从休眠状态中被唤醒。如果已经允许了全局中断，程序将跳转到中断向量地址处继续执行。

40.7 空闲模式和打盹模式期间的操作

在空闲模式和打盹模式下，模块处于工作状态，并且如果使能外设，则会生成相关事件。

40.8 冻结期间的操作

在冻结模式下，不会生成新事件或中断。LRDY位的状态将被冻结。

允许通过CPU接口读写寄存器。

40.9 复位的影响

器件复位将强制所有寄存器进入复位状态。这会强制关闭HLVD模块。

40.10 寄存器定义：HLVD控制

表40-1列出了HLVD外设的长位名称前缀。更多信息，请参见**第1.3.2.2节“长位名称”**。

表40-1：

外设	位名称前缀
HLVD	HLVD

寄存器40-1： HLVDCON0：高/低电压检测控制寄存器0

R/W-0/0	U-0	R-x	R-x	U-0	U-0	R/W-0/0	R/W-0/0
EN	—	OUT	RDY	—	—	INTH	INTL
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	x = 未知

bit 7 **EN**：高/低电压检测电源使能位
1 = 使能HLVD，为HLVD电路上电并支持参考电路
0 = 禁止HLVD，为HLVD电路掉电并支持参考电路

bit 6 **未实现**：读为0

bit 5 **OUT**：HLVD比较器输出位
1 = 电压 ≤ 选定的检测限值（HLVDL<3:0>）
0 = 电压 ≥ 选定的检测限值（HLVDL<3:0>）

bit 4 **RDY**：带隙参考电压稳定状态标志位
1 = 指示HLVD模块已就绪且输出稳定
0 = 指示HLVD模块未就绪

bit 3-2 **未实现**：读为0

bit 1 **INTH**：HLVD正向（高电压）中断允许
1 = HLVDIF将在电压 ≥ 选定的检测限值（SEL<3:0>）时置1
0 = HLVDIF不会置1

bit 0 **INTL**：HLVD负向（低电压）中断允许
1 = HLVDIF将在电压 ≤ 选定的检测限值（SEL<3:0>）时置1
0 = HLVDIF不会置1

寄存器40-2： HLVDCON1：低电压检测控制寄存器1

U-0	U-0	U-0	U-0	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	—	SEL<3:0>			
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为0	
-n = POR时的值	1 = 置1	0 = 清零	u = 不变

bit 7-4 **未实现：**读为0

bit 3-0 **SEL<3:0>：**高/低电压检测限值选择位

SEL<3:0>	典型电压
1111	保留
1110	4.65V
1101	4.35V
1100	4.20V
1011	4.00V
1010	3.75V
1001	3.60V
1000	3.35V
0111	3.15V
0110	2.90V
0101	2.75V
0100	2.60V
0011	2.50V
0010	2.25V
0001	2.10V
0000	1.90V

表40-2： 与高/低电压检测模块相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
HLVDCON0	EN	—	OUT	RDY	—	—	INTH	INTL	711
HLVDCON1	—	—	—	—	SEL<3:0>				712

图注： — = 未实现，读为0。HLVD模块不使用阴影单元。

41.0 在线串行编程（ICSP）

ICSP编程允许用户在生产电路板时使用未编程器件。编程可以在组装流程之后完成，从而可以使用最新版本的固件或者定制固件对器件编程。ICSP编程需要5个引脚：

- ICSPCLK
- ICSPDAT
- $\overline{\text{MCLR}}$/VPP
- VDD
- VSS

在编程/校验模式下，通过串行通信对程序存储器、用户ID和配置字进行编程。ICSPDAT引脚是用于传输串行数据的双向I/O引脚，ICSPCLK引脚是时钟输入引脚。关于ICSP的更多信息，请参见“*PIC18(L)F25/26K82 Memory Programming Specification*”（DS40000000）。

41.1 高电压编程模式

通过将ICSPCLK和ICSPDAT引脚保持为低电平，然后将$\overline{\text{MCLR}}$/VPP上的电压升至VIHH，将器件置于高电压编程模式。

41.2 低电压编程模式

通过低电压编程模式，只需使用VDD就可以对PIC®闪存MCU进行编程，而无需使用高电压。当配置字的LVP位设置为1时，将会使能低电压ICSP编程模式。要禁止低电压ICSP模式，LVP位必须编程为0。

进入低电压编程模式需要执行以下步骤：

1. $\overline{\text{MCLR}}$电压设置为VIL。
2. 在ICSPDAT引脚上输出32位密钥序列，而在ICSPCLK引脚上输出时钟。

输出完密钥序列后，在需要维持编程/校验模式的时间内，必须将$\overline{\text{MCLR}}$保持为VIL。

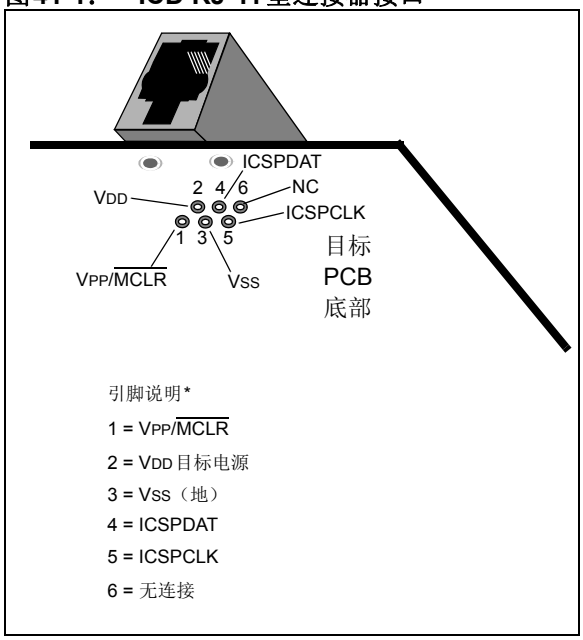
如果使能了低电压编程（LVP = 1），则$\overline{\text{MCLR}}$复位功能会被自动使能，无法禁止。更多信息，请参见**第6.5节“MCLR”**。

LVP位只能通过使用高电压编程模式重新设定为0。

41.3 常用编程接口

与目标器件的连接通常通过ICSP连接头来实现。开发工具中常见的连接器是采用6P6C（6引脚，6连接器）配置的RJ-11。请参见图41-1。

图41-1： ICD RJ-11型连接器接口

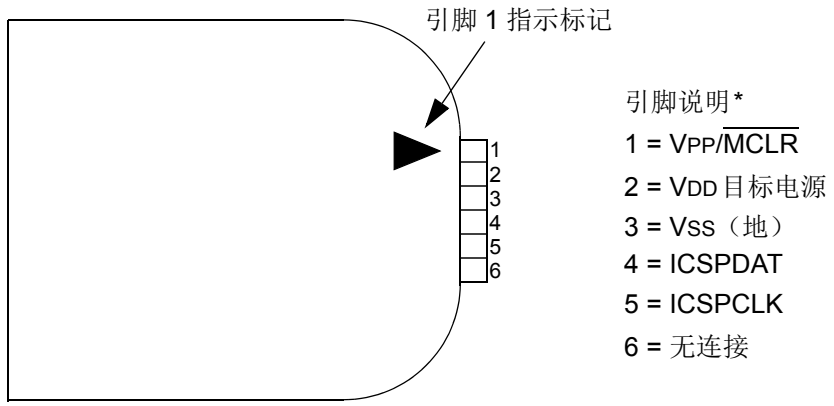


另一种常用于PICkit™编程器的连接器是间距为0.1英寸的标准6引脚连接头。请参见图41-2。

关于其他接口建议，请在进行PCB设计之前参考具体的器件编程器手册。

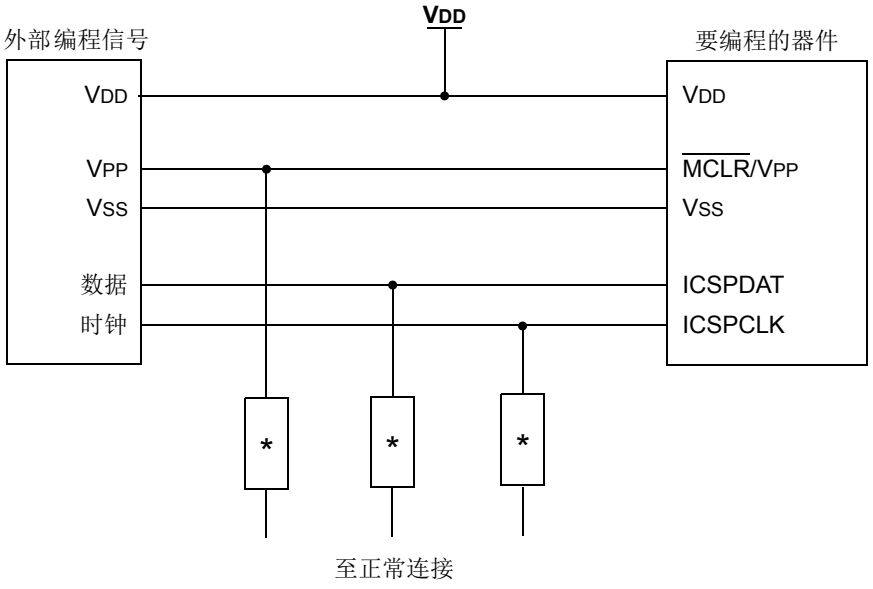
建议使用隔离器件来隔离编程引脚与其他电路。隔离类型高度依赖于具体应用，可能会包含诸如电阻、二极管甚至跳线之类的元件。更多信息，请参见图41-3。

图41-2： PICkit™ 编程器型连接器接口



* 6引脚连接头（0.100"间距）可连接0.025"方形引脚。

图41-3： ICSP™ 编程的典型连接



* 隔离器件（根据需要而定）。

42.0 指令集汇总

PIC18(L)F25/26K83器件包含标准PIC18核心指令集和扩展指令集，用于优化递归或利用软件堆栈的代码。本节将讨论扩展指令集。

42.1 标准指令集

标准PIC18指令集相较于以前的PIC®MCU指令集增加了许多增强功能，同时可以从这些PIC®MCU指令集轻松移植。大部分指令的长度为一个程序存储字（16位），但有四条指令需要两个程序存储单元，两条指令需要三个程序存储单元。

每一条单字指令长16位，分为一个指定指令类型的操作码和进一步指定指令操作的一个或多个操作数。

指令集是高度正交的，分为4个基本类别：

- **针对字节的**操作
- **针对位的**操作
- **立即数**操作
- **控制**操作

表42-3是PIC18指令集的汇总，列出了**针对字节的**、**针对位的**、**立即数**和**控制**操作。表42-1给出了操作码字段说明。

大多数**针对字节的**指令有三个操作数：

1. 文件寄存器（由f指定）
2. 保存结果的目标寄存器（由d指定）
3. 访问的存储器（由a指定）

文件寄存器标识符f指定指令将会使用哪个文件寄存器。目标寄存器标识符d指定操作结果的存放位置。如果d为0，则将操作结果存入WREG寄存器中。如果d为1，则将操作结果存入指令指定的文件寄存器中。

所有**针对位的**指令有三个操作数：

1. 文件寄存器（由f指定）
2. 文件寄存器中的位（由b指定）
3. 访问的存储器（由a指定）

位域标识符b用于选择操作所影响的位，文件寄存器标识符f代表位所在文件寄存器的编号。

立即数指令可以使用下列操作数：

- 要被装入到文件寄存器中的立即数（由k指定）
- 要装入立即数的所需FSR寄存器（由f指定）
- 不需要操作数（由—指定）

控制指令可以使用下列操作数：

- 程序存储器地址（由n指定）
- CALL或RETURN指令的模式（由s指定）
- 表读和表写指令的模式（由m指定）
- 不需要操作数（由—指定）

除了四个双字指令外，所有指令都是单字指令。这些指令是双字指令，以包含32位的所需信息。在第二个字中，高四位全为1。如果指令自身将第二个字当作一条指令来执行的话，它将作为一条NOP指令来执行。

所有单字长指令都在一个指令周期内执行，除非条件测试为真、指令执行结果改变了程序计数器。在这些情况下，指令执行需要两个指令周期，在额外的指令周期中执行NOP指令。

双字指令执行需要两个指令周期。

一个指令周期包含四个振荡器周期。因此，如果振荡器频率为4 MHz，则正常的指令执行时间为1 µs。如果条件测试为真或者指令执行改变了程序计数器的值，则指令执行时间为2 µs。双字跳转指令（如果为真）的执行时间为3 µs。

图42-1显示了指令的一般格式。所有示例均使用约定的“nnh”表示一个十六进制数。

指令集汇总（如表42-3所示）列出了可被Microchip汇编器（MPASM™）识别的标准指令。

第42.1.1节“标准指令集”给出了每条指令的说明。

表42-1： 操作码字段说明

字段	说明
a	RAM访问位 a = 0：访问RAM中的RAM存储单元（忽略BSR寄存器） a = 1：RAM存储区由BSR寄存器指定
ACCESS	ACCESS = 0：RAM访问位码
BANKED	BANKED = 1：RAM访问位码
bbb	8位文件寄存器内的位地址（0至7）
BSR	存储区选择寄存器。用于选择当前RAM存储区。
d	目标选择位； d = 0：结果存入WREG， d = 1：结果存入文件寄存器f。
dest	目标为WREG寄存器或指定的寄存器文件存储单元
f	8位文件寄存器地址（00h至FFh）
f_n	FSR编号（0至2）
f_s	12位文件寄存器地址（000h至FFFh）。这是源地址。
f_d	12位文件寄存器地址（000h至FFFh）。这是目标地址。
z_s	用作文件寄存器地址的FSR2的7位立即数偏移量（000h至FFFh）。这是源地址。
z_d	用作文件寄存器地址的FSR2的7位立即数偏移量（000h至FFFh）。这是目标地址。
k	立即数字段、常量数据或标号（可以为6位、8位、12位或20位值）
label	标号名称
mm	TBLPTR寄存器表读和表写指令的模式 仅用于表读和表写指令：
*	无需更改寄存器（例如带表读和表写的TBLPTR）
*+	后递增寄存器（例如带表读和表写的TBLPTR）
*-	后递减寄存器（例如带表读和表写的TBLPTR）
+*	预递增寄存器（例如带表读和表写的TBLPTR）
n	相对转移指令的相对地址（二进制补码），或Call/Branch和Return指令的直接地址
PRODH	乘以高字节的乘积
PRODL	乘以低字节的乘积
s	快速Call/Return模式选择位。 s = 0：不要更新到影子寄存器/从影子寄存器更新 s = 1：某些寄存器加载到影子寄存器/从影子寄存器装载（快速模式）
u	未使用或未更改
W	W = 0：目标选择位码
WREG	工作寄存器（累加器）
x	忽略（0或1） 汇编器将生成x = 0的代码。为了与所有的Microchip软件工具兼容，建议使用这种格式。
TBLPTR	21位表指针（指向程序存储单元）
TABLAT	8位表锁存
TOS	栈顶
PC	程序计数器
PCL	程序计数器低字节
PCH	程序计数器高字节
PCLATH	程序计数器高字节锁存
PCLATU	程序计数器高字节锁存
GIE	全局中断允许位
WDT	看门狗定时器
TO	超时位
PD	掉电位
C、DC、Z、OV和N	进位、半进位、零、溢出和负ALU状态位
[ ]	可选
( )	内容
→	分配至

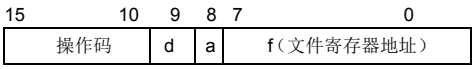
表42-1： 操作码字段说明（续）

字段	说明
< >	寄存器位域
∈	属于
斜体	用户定义的术语（字体为courier）

图42-1： 指令的一般格式（1/2）

针对字节的文件寄存器操作类指令

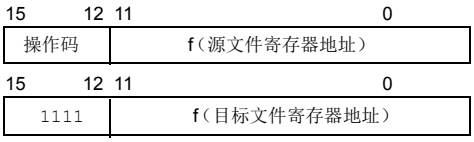
指令示例




```
ADDWF MYREG, W, B
```

d = 0，结果目标为 WREG 寄存器
d = 1，结果目标为文件寄存器（f）
a = 0，强制为快速操作存储区
a = 1，BSR 用于选择存储区
f = 8 位文件寄存器地址

字节到字节移动操作（2 字）



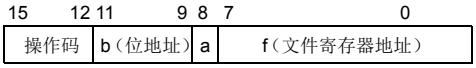

```
MOVFF MYREG1, MYREG2
```

f = 12 位文件寄存器地址

字节到字节移动操作（3 字）


```
MOVFFL MYREG1, MYREG2
```

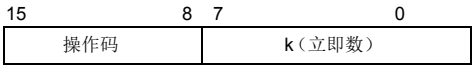
针对位的文件寄存器操作类指令




```
BSF MYREG, bit, B
```

b = 文件寄存器中位的 3 位位置（f）
a = 0，强制为快速操作存储区
a = 1，BSR 用于选择存储区
f = 8 位文件寄存器地址

立即数操作




```
MOVLW 7Fh
```

k = 8 位立即数值

图42-2： 指令的一般格式（2/2）

控制操作

CALL、GOTO 和 Branch 操作

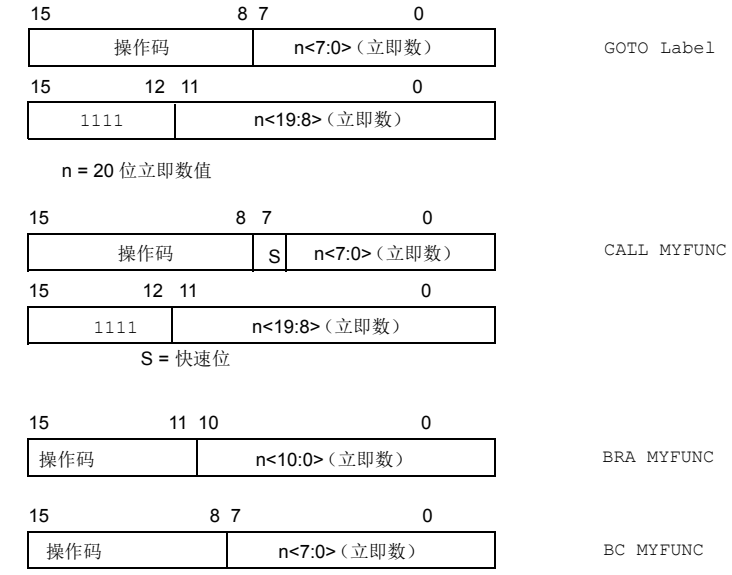


表42-2： 指令集

助记符，操作数		说明	周期数	16位指令字 MSb	LSb	受影响的状态位	注
针对字节的文件寄存器指令							
ADDWF	f, d ,a	WREG和f相加	1	0010 01da	ffff ffff	C、DC、Z、OV和N	
ADDWFC	f, d, a	WREG和f进行带进位的相加	1	0010 00da	ffff ffff	C、DC、Z、OV和N	
ANDWF	f, d, a	WREG和f作逻辑与运算	1	0001 01da	ffff ffff	Z和N	
CLRF	f, a	将f清零	1	0110 101a	ffff ffff	Z	
COMF	f, d, a	f取反	1	0001 11da	ffff ffff	Z和N	
DECF	f, d, a	f递减1	1	0000 01da	ffff ffff	C、DC、Z、OV和N	
INCF	f, d, a	f递增1	1	0010 10da	ffff ffff	C、DC、Z、OV和N	
IORWF	f, d, a	WREG与f作逻辑或运算	1	0001 00da	ffff ffff	Z和N	
MOVF	f, d, a	将f中的内容送入WREG或f	1	0101 00da	ffff ffff	Z和N	
MOVFF	f_s, f_d	将f_s（源）中的内容送入 第1个字 f_d（目标） 第2个字	2	1100 ffff 1111 ffff	ffff ffff ffff ffff	无	2和3
MOVFFL	f_s, f_d	将f_s（源）中的内容送入 g（满目标） f_d（满目标） 第3个字	3	0000 0000 1111 ffff 1111 gggg	0110 ffff ffff ffgg gggg gggg	无	2
MOVWF	f, a	将WREG中的内容送入f	1	0110 111a	ffff ffff	无	
MULWF	f, a	WREG和f作乘法运算	1	0000 001a	ffff ffff	无	
NEGF	f, a	将f取反	1	0110 110a	ffff ffff	C、DC、Z、OV和N	
RLCF	f, d, a	f带进位循环左移	1	0011 01da	ffff ffff	C、Z和N	
RLNCF	f, d, a	f循环左移（不带进位）	1	0100 01da	ffff ffff	Z和N	
RRCF	f, d, a	f带进位循环右移	1	0011 00da	ffff ffff	C、Z和N	
RRNCF	f, d, a	f循环右移（不带进位）	1	0100 00da	ffff ffff	Z和N	
SETF	f, a	将f置1	1	0110 100a	ffff ffff	无	
SUBFWB	f, d, a	WREG减去f（带借位）	1	0101 01da	ffff ffff	C、DC、Z、OV和N	
SUBWF	f, d, a	f减去WREG	1	0101 11da	ffff ffff	C、DC、Z、OV和N	
SUBWFB	f, d, a	f减去WREG（带借位）	1	0101 10da	ffff ffff	C、DC、Z、OV和N	
SWAPF	f, d, a	将f中的两个半字节进行交换	1	0011 10da	ffff ffff	无	
XORWF	f, d, a	WREG与f作逻辑异或运算	1	0001 10da	ffff ffff	Z和N	
针对字节的跳过指令							
CPFSEQ	f, a	比较f和WREG，如果相等则跳过	1（2或3）	0110 001a	ffff ffff	无	1
CPFSGT	f, a	比较f和WREG，如果大于则跳过	1（2或3）	0110 010a	ffff ffff	无	1
CPFSLT	f, a	比较f和WREG，如果小于则跳过	1（2或3）	0110 000a	ffff ffff	无	1
DECFSZ	f, d, a	f递减1，为0则跳过	1（2或3）	0010 11da	ffff ffff	无	1
DCFSNZ	f, d, a	f递减1，不为0则跳过	1（2或3）	0100 11da	ffff ffff	无	1
INCFSZ	f, d, a	f递增1，为0则跳过	1（2或3）	0011 11da	ffff ffff	无	1
INFSNZ	f, d, a	f递增1，不为0则跳过	1（2或3）	0100 10da	ffff ffff	无	1
TSTFSZ	f, a	测试f，为0则跳过	1（2或3）	0110 011a	ffff ffff	无	1
针对位的文件寄存器指令							
BCF	f, b, a	将f中的某位清零	1	1001 bbba	ffff ffff	无	
BSF	f, b, a	将f中的某位置1	1	1000 bbba	ffff ffff	无	
BTG	f, d, a	将f中的指定位取反	1	0111 bbba	ffff ffff	无	
针对位的跳过指令							
BTFSC	f, b, a	测试f中的某位，为0则跳过	1（2或3）	1011 bbba	ffff ffff	无	1
BTFSS	f, b, a	测试f中的某位，为1则跳过	1（2或3）	1010 bbba	ffff ffff	无	1

注 1： 如果程序计数器（PC）被修改或条件测试结果为真，则该指令需要一个额外的周期。额外的周期执行一条NOP指令。

2： 有些指令是多字指令。这些指令的第二个/第三个字将被解码为NOP，除非指令的第一个字检索嵌入在这16位中的信息。这确保了所有程序存储单元都有一个有效的指令。

3： f_s和f_d不覆盖整个存储器范围。存储区选择的高2位被强制为'b00以限制这些指令的范围为低4k地址空间。

表42-2： 指令集（续）

助记符，操作数		说明	周期数	16位指令字 MSb LSb	受影响的状态位	注
控制指令						
BC	n	如果有进位则跳转	1（2）	1110 0010 nnnn nnnn	无	1
BN	n	如果为负则跳转	1（2）	1110 0110 nnnn nnnn	无	1
BNC	n	如果没有进位则跳转	1（2）	1110 0011 nnnn nnnn	无	1
BNN	n	如果不为负则跳转	1（2）	1110 0111 nnnn nnnn	无	1
BNOV	n	如果未溢出则跳转	1（2）	1110 0101 nnnn nnnn	无	1
BNZ	n	如果不为零则跳转	2	1110 0001 nnnn nnnn	无	1
BOV	n	如果溢出则跳转	1（2）	1110 0100 nnnn nnnn	无	1
BRA	n	无条件跳转	1（2）	1101 0nnn nnnn nnnn	无	
BZ	n	如果为零则跳转	1（2）	1110 0000 nnnn nnnn	无	1
CALL	n, s	调用子程序 第1个字	2	1110 110s nnnn nnnn	无	2
		第2个字		1111 nnnn nnnn nnnn		
GOTO	n	跳转到地址 第1个字	2	1110 1111 nnnn nnnn	无	2
	—	第2个字		1111 nnnn nnnn nnnn		
CALLW	—	W -> PCL和调用子程序	2	0000 0000 0001 0100	无	1
RCALL	n	相对调用	2	1101 1nnn nnnn nnnn	无	1
RETFIE	s	使能从中断返回	2	0000 0000 0001 000s	无	1
RETLW	k	返回并将立即数传送到WREG	2	0000 1100 kkkk kkkk	无	1
RETURN	s	从子程序返回	2	0000 0000 0001 001s	无	1
固有指令						
CLRWDT	—	将看门狗定时器清零	1	0000 0000 0000 0100	无	
DAW	—	十进制调整WREG	1	0000 0000 0000 0111	C	
NOP	—	空操作	1	0000 0000 0000 0000	无	
NOP	—	空操作	1	1111 xxxx xxxx xxxx	无	2
POP	—	将返回栈顶（TOS）的内容弹出	1	0000 0000 0000 0110	无	
PUSH	—	将返回栈顶（TOS）的内容压入	1	0000 0000 0000 0101	无	
RESET		软件器件复位	1	0000 0000 1111 1111	全部	
SLEEP	—	进入待机模式	1	0000 0000 0000 0011	无	

注 1： 如果程序计数器（PC）被修改或条件测试结果为真，则该指令需要一个额外的周期。额外的周期执行一条NOP指令。

2： 有些指令是多字指令。这些指令的第二个/第三个字将被解码为NOP，除非指令的第一个字检索嵌入在这16位中的信息。这确保了所有程序存储单元都有一个有效的指令。

3： f_s和f_d不覆盖整个存储器范围。存储区选择的高2位被强制为'b00以限制这些指令的范围为低4k地址空间。

表42-2： 指令集（续）

助记符，操作数		说明	周期数	16位指令字 MSb LSb	受影响的状态位	注
立即数指令						
ADDLW	k	立即数和WREG相加	1	0000 1111 kkkk kkkk	C、DC、Z、OV和N	
ANDLW	k	立即数与WREG作逻辑与运算	1	0000 1011 kkkk kkkk	Z和N	
IORLW	k	立即数与WREG作逻辑或运算	1	0000 1001 kkkk kkkk	Z和N	
LFSR	f_n, k	用14位立即数（k）装载FSR（f_n）	2	1110 1110 00ff kkkk 1111 00kk kkkk kkkk	无	
ADDFSR	f_n, k	将FSR（f_n）与（k）相加	1	1110 1000 ffkk kkkk	无	
SUBFSR	f_n, k	从FSR（f_n）中减去（k）	1	1110 1001 ffkk kkkk	无	
MOVLB	k	将立即数送入BSR<5:0>	1	0000 0001 00kk kkkk	无	
MOVLW	k	将立即数送入WREG	1	0000 1110 kkkk kkkk	无	
MULLW	k	立即数与WREG作乘法运算	1	0000 1101 kkkk kkkk	无	
RETLW	k	返回并将立即数传送到WREG	2	0000 1100 kkkk kkkk	无	
SUBLW	k	立即数减去WREG	1	0000 1000 kkkk kkkk	C、DC、Z、OV和N	
XORLW	k	立即数和WREG作逻辑异或运算	1	0000 1010 kkkk kkkk	Z和N	
数据存储器–程序存储器指令						
TBLRD*		表读	2 - 5	0000 0000 0000 1000	无	
TBLRD*+		表读（后递增）		0000 0000 0000 1001	无	
TBLRD*-		表读（后递减）		0000 0000 0000 1010	无	
TBLRD+*		表读（预递增）		0000 0000 0000 1011	无	
TBLWT*		表写	2 - 5	0000 0000 0000 1100	无	
TBLWT*+		表写（后递增）		0000 0000 0000 1101	无	
TBLWT*-		表写（后递减）		0000 0000 0000 1110	无	
TBLWT+*		表写（预递增）		0000 0000 0000 1111	无	

注 1： 如果程序计数器（PC）被修改或条件测试结果为真，则该指令需要一个额外的周期。额外的周期执行一条NOP指令。

2： 有些指令是多字指令。这些指令的第二个/第三个字将被解码为NOP，除非指令的第一个字检索嵌入在这16位中的信息。这确保了所有程序存储单元都有一个有效的指令。

3： f_s和f_d不覆盖整个存储器范围。存储区选择的高2位被强制为'b00以限制这些指令的范围为低4k地址空间。

42.1.1 标准指令集

ADDFSR **将立即数加到FSR**

语法： ADDFSR f, k

操作数： $0 \le k \le 63$
$f \in [0, 1, 2]$

操作： FSR(f) + k → FSR(f)

受影响的状态位： 无

编码：

1110	1000	ffkk	kkkk

说明： 将6位立即数k加到f指定的FSR的内容。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入FSR
译码	读立即数k	处理数据	写入FSR

示例：
```
ADDFSR 2, 23h
```
执行指令前
FSR2 = 03FFh
执行指令后
FSR2 = 0422h

ADDLW **立即数与W相加**

语法： ADDLW k

操作数： $0 \le k \le 255$

操作： (W) + k → W

受影响的状态位： N、OV、C、DC和Z

编码：

0000	1111	kkkk	kkkk

说明： 将W寄存器的内容与8位立即数k相加，结果存入W寄存器。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入W

示例：
```
ADDLW 15h
```
执行指令前
W = 10h
执行指令后
W = 25h

ADDWF **W与f相加**

语法： ADDWF f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

操作： (W) + (f) → 目标寄存器

受影响的状态位： N、OV、C、DC和Z

编码：

0010	01da	ffff	ffff

说明： W与寄存器f相加。如果d为0，结果存入W。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要$f \le 95$（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：
```
ADDWF REG, 0, 0
```
执行指令前
W = 17h
REG = 0C2h
执行指令后
W = 0D9h
REG = 0C2h

注： 所有PIC18指令都可以在指令助记符之前采用可选的标号参数，以用于符号寻址。如果使用标号，则指令格式变为：{标号} 指令参数。

ADDWFC **W与f相加（带进位）**

语法： ADDWFC f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) + (f) + (C) → 目标寄存器

受影响的状态位： N、OV、C、DC和Z

编码：

0010	00da	ffff	ffff

说明： 将W的内容、进位标志位与数据存储单元f的内容相加。如果d为0，结果放入W寄存器。如果d为1，结果放入数据存储单元f。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例： ADDWFC REG, 0和1

执行指令前
进位位 = 1
REG = 02h
W = 4Dh
执行指令后
进位位 = 0
REG = 02h
W = 50h

ANDLW **立即数和W作逻辑与运算**

语法： ANDLW k

操作数： 0 ≤ k ≤ 255

操作： (W) .AND. k → W

受影响的状态位： N和Z

编码：

0000	1011	kkkk	kkkk

说明： 将W寄存器的内容与8位立即数k进行逻辑与运算。结果存入W寄存器。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入W

示例： ANDLW 05Fh

执行指令前
W = A3h
执行指令后
W = 03h

ANDWF　W与f作逻辑与运算

语法：　ANDWF　f {,d {,a}}

操作数：　0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作：　(W) .AND. (f) → 目标寄存器

受影响的状态位：　N和Z

编码：

0001	01da	ffff	ffff

说明：　将W寄存器的内容与寄存器f进行逻辑与运算。
如果d为0，结果存入W。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数：　1

指令周期数：　1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：

```
ANDWF    REG,  0,  0
```

执行指令前
W = 17h
REG = C2h
执行指令后
W = 02h
REG = C2h

BC　如果有进位则跳转

语法：　BC　n

操作数：　-128 ≤ n ≤ 127

操作：　如果进位位为1
(PC) + 2 + 2n → PC

受影响的状态位：　无

编码：

1110	0010	nnnn	nnnn

说明：　如果进位位为1，程序将跳转。
将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。
然后，该指令为一条双周期指令。

指令字数：　1

指令周期数：　1（2）

Q周期活动：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	空操作

示例：

```
HERE      BC   5
```

执行指令前
PC = 地址（HERE）
执行指令后
如果进位位 = 1；
PC = 地址（HERE + 12）
如果进位位 = 0；
PC = 地址（HERE + 2）

BCF **将f中的某位清零**

语法： BCF f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

操作： 0 → f<b>

受影响的状态位： 无

编码：

1001	bbba	ffff	ffff

说明： 将寄存器f中的bit b清零。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入寄存器f

示例： BCF FLAG_REG, 7, 0

执行指令前
FLAG_REG = C7h
执行指令后
FLAG_REG = 47h

BN **如果为负则跳转**

语法： BN n

操作数： -128 ≤ n ≤ 127

操作： 如果负标志位为1
(PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1110	0110	nnnn	nnnn

说明： 如果负标志位为1，程序将跳转。
将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。然后，该指令为一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q周期活动：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	空操作

示例： HERE BN Jump

执行指令前
PC = 地址（HERE）
执行指令后
如果负标志位 = 1；
PC = 地址（Jump）
如果负标志位 = 0；
PC = 地址（HERE + 2）

BNC **如果没有进位则跳转**

语法： BNC n

操作数： -128 ≤ n ≤ 127

操作： 如果进位位为0
(PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1110	0011	nnnn	nnnn

说明： 如果进位位为0，程序将跳转。
将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。然后，该指令为一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q周期活动：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	空操作

示例： HERE BNC Jump

执行指令前
PC = 地址（HERE）
执行指令后
如果进位位 = 0；
PC = 地址（Jump）
如果进位位 = 1；
PC = 地址（HERE + 2）

BNN **如果不为负则跳转**

语法： BNN n

操作数： -128 ≤ n ≤ 127

操作： 如果负标志位为0
(PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1110	0111	nnnn	nnnn

说明： 如果负标志位为0，程序将跳转。
将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。然后，该指令为一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q周期活动：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	空操作

示例： HERE BNN Jump

执行指令前
PC = 地址（HERE）
执行指令后
如果负标志位 = 0；
PC = 地址（Jump）
如果负标志位 = 1；
PC = 地址（HERE + 2）

BNOV **如果未溢出则跳转**

语法： BNOV n

操作数： -128 ≤ n ≤ 127

操作： 如果溢出位为0
(PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1110	0101	nnnn	nnnn

说明： 如果溢出位为0，程序将跳转。
将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。然后，该指令为一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q周期活动：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	空操作

示例：
```
HERE    BNOV  Jump
```
执行指令前
PC = 地址（HERE）
执行指令后
如果溢出位 = 0；
PC = 地址（Jump）
如果溢出位 = 1；
PC = 地址（HERE + 2）

BNZ **如果不为零则跳转**

语法： BNZ n

操作数： -128 ≤ n ≤ 127

操作： 如果零位为0
(PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1110	0001	nnnn	nnnn

说明： 如果零位为0，程序将跳转。
将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。然后，该指令为一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q周期活动：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	空操作

示例：
```
HERE    BNZ  Jump
```
执行指令前
PC = 地址（HERE）
执行指令后
如果零位 = 0；
PC = 地址（Jump）
如果零位 = 1；
PC = 地址（HERE + 2）

BRA 无条件跳转

语法： BRA n

操作数： -1024 ≤ n ≤ 1023

操作： (PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1101	0nnn	nnnn	nnnn

说明： 将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。该指令为一条双周期指令。

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

示例： HERE BRA Jump

执行指令前
PC = 地址（HERE）
执行指令后
PC = 地址（Jump）

BSF 将f中的某位置1

语法： BSF f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

操作： 1 → f<b>

受影响的状态位： 无

编码：

1000	bbba	ffff	ffff

说明： 将寄存器f中的bit b置1。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入寄存器f

示例： BSF FLAG_REG, 7, 1

执行指令前
FLAG_REG = 0Ah
执行指令后
FLAG_REG = 8Ah

BTFSC **测试文件中的某位，为0则跳过**

语法： BTFSC f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

操作： 如果(f<b>) = 0，则跳过

受影响的状态位： 无

编码：

1011	bbba	ffff	ffff

说明： 如果寄存器f的位b为0，则跳过下一条指令。如果位b为0，则丢弃执行当前指令的过程中取指的下一条指令，代之执行一条NOP指令，使之成为一条双周期指令。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```
执行指令前
PC = 地址（HERE）
执行指令后
如果FLAG<1> = 0；
PC = 地址（TRUE）
如果FLAG<1> = 1；
PC = 地址（FALSE）

BTFSS **测试文件中的某位，为1则跳过**

语法： BTFSS f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b < 7
a ∈ [0,1]

操作： 如果(f<b>) = 1，则跳过

受影响的状态位： 无

编码：

1010	bbba	ffff	ffff

说明： 如果寄存器f的位b为1，则跳过下一条指令。如果位b为1，则丢弃执行当前指令的过程中取指的下一条指令，代之执行一条NOP指令，使之成为一条双周期指令。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```
执行指令前
PC = 地址（HERE）
执行指令后
如果FLAG<1> = 0；
PC = 地址（FALSE）
如果FLAG<1> = 1；
PC = 地址（TRUE）

BTG **将f中的指定位取反**

语法： BTG f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b < 7
a ∈ [0,1]

操作： （$\overline{f<b>}$） → f<b>

受影响的状态位： 无

编码：

0111	bbba	ffff	ffff

说明： 将数据存储单元f中的bit b翻转。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入寄存器f

示例：
```
BTG     PORTC,  4, 0
```
执行指令前：
PORTC = 0111 0101 [75h]
执行指令后：
PORTC = 0110 0101 [65h]

BOV **如果溢出则跳转**

语法： BOV n

操作数： -128 ≤ n ≤ 127

操作： 如果溢出位为1
(PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1110	0100	nnnn	nnnn

说明： 如果溢出位为1，程序将跳转。
将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。
然后，该指令为一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q周期活动：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	空操作

示例：
```
HERE    BOV   Jump
```
执行指令前
PC = 地址（HERE）
执行指令后
如果溢出位 = 1；
PC = 地址（Jump）
如果溢出位 = 0；
PC = 地址（HERE + 2）

BZ **如果为零则跳转**

语法： BZ n

操作数： -128 ≤ n ≤ 127

操作： 如果零位为1
(PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1110	0000	nnnn	nnnn

说明： 如果零位为1，程序将跳转。
将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。
然后，该指令为一条双周期指令。

指令字数： 1

指令周期数： 1（2）

Q周期活动：
如果跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	写入PC
空操作	空操作	空操作	空操作

如果不跳转：

Q1	Q2	Q3	Q4
译码	读立即数n	处理数据	空操作

示例： HERE BZ Jump

执行指令前
PC = 地址（HERE）
执行指令后
如果零位 = 1；
PC = 地址（Jump）
如果零位 = 0；
PC = 地址（HERE + 2）

CALL **调用子程序**

语法： CALL k {,s}

操作数： 0 ≤ k ≤ 1048575
s ∈ [0,1]

操作： (PC) + 4 → TOS，
k → PC<20:1>，
如果s = 1
（W） → WS，
（Status） → STATUSS，
（BSR） → BSRS

受影响的状态位： 无

编码：

第1个字（k<7:0>）	1110	110s	$k_7$kkk	kkkk$_0$
第2个字（k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明： 在整个2 MB存储器范围内调用子程序。首先，将返回地址（PC + 4）压入返回堆栈。如果s = 1，则将W、Status和BSR寄存器压入其相应的影子寄存器WS、STATUSS和BSRS。
如果s = 0，则不发生更新（默认）。然后，20位值k被装入PC<20:1>。
CALL是一条双周期指令。

指令字数： 2

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k<7:0>	将PC压入堆栈	读立即数k<19:8>，写入PC
空操作	空操作	空操作	空操作

示例： HERE CALL THERE, 1

执行指令前
PC = 地址（HERE）
执行指令后
PC = 地址（THERE）
TOS = 地址（HERE + 4）
WS = W
BSRS = BSR
STATUSS = Status

CALLW **使用WREG调用子程序**

语法： CALLW

操作数： 无

操作： （PC + 2） → TOS，
（W） → PCL，
（PCLATH） → PCH，
（PCLATU） → PCU

受影响的状态位： 无

编码：

0000	0000	0001	0100

说明 首先，将返回地址（PC + 2）压入返回堆栈。接下来，将W的内容写入PCL并丢弃现有值。然后，PCLATH和PCLATU的内容分别被锁存到PCH和PCU中。
取出新的下一条指令时，第二个周期执行一条NOP指令。

与CALL不同，没有更新W、Status或BSR的选项。

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读WREG	将PC压入堆栈	空操作
空操作	空操作	空操作	空操作

示例： HERE CALLW

执行指令前
PC = 地址（HERE）
PCLATH = 10h
PCLATU = 00h
W = 06h

执行指令后
PC = 001006h
TOS = 地址（HERE + 2）
PCLATH = 10h
PCLATU = 00h
W = 06h

CLRF **将f清零**

语法： CLRF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： 000h → f
1 → Z

受影响的状态位： Z

编码：

0110	101a	ffff	ffff

说明： 清除指定寄存器的内容。

如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入寄存器f

示例： CLRF FLAG_REG, 1

执行指令前
FLAG_REG = 5Ah
执行指令后
FLAG_REG = 00h

CLRWDT **将看门狗定时器清零**

语法： CLRWDT

操作数： 无

操作： 000h → WDT，
000h → WDT后分频器，
1 → $\overline{TO}$，
1 → PD

受影响的状态位： $\overline{TO}$和$\overline{PD}$

编码：

0000	0000	0000	0100

说明： CLRWDT指令复位看门狗定时器及其后分频器。状态位$\overline{TO}$和$\overline{PD}$均被置1。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	空操作

示例： CLRWDT

执行指令前
WDT计数器 = ?
执行指令后
WDT计数器 = 00h
WDT后分频器 = 0
$\overline{TO}$ = 1
PD = 1

COMF **f取反**

语法： COMF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： ($\bar{f}$) → 目标寄存器

受影响的状态位： N和Z

编码：

0001	11da	ffff	ffff

说明： 将寄存器f的内容取反。
如果d为0，结果存入W。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例： COMF REG, 0, 0

执行指令前
REG = 13h
执行指令后
REG = 13h
W = ECh

CPFSEQ 比较f和W，如果f = W则跳过

语法： CPFSEQ f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (f) – (W)，
如果(f) = (W)则跳过
（无符号比较）

受影响的状态位： 无

编码：

0110	001a	ffff	ffff

说明： 通过执行无符号减法将数据存储单元f的内容与W的内容进行比较。
如果f = W，则丢弃取出的指令，转而执行一条NOP指令，从而使该指令成为双周期指令。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     CPFSEQ REG, 0
NEQUAL   :
EQUAL    :
```

执行指令前
PC地址 = HERE
W = ?
REG = ?
执行指令后
如果REG = W；
PC = 地址（EQUAL）
如果REG ≠ W；
PC = 地址（NEQUAL）

CPFSGT 比较f和W，如果f > W则跳过

语法： CPFSGT f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (f) – (W)，
如果(f) > (W)则跳过
（无符号比较）

受影响的状态位： 无

编码：

0110	010a	ffff	ffff

说明： 通过执行无符号减法将数据存储单元f的内容与W的内容进行比较。
如果f的内容大于WREG的内容， 则丢弃取出的指令，转而执行一条NOP指令，从而使该指令成为双周期指令。

如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     CPFSGT REG, 0
NGREATER :
GREATER  :
```

执行指令前
PC = 地址（HERE）
W = ?
执行指令后
如果REG > W；
PC = 地址（GREATER）
如果REG ≤ W；
PC = 地址（NGREATER）

CPFSLT **比较f和W，如果f < W则跳过**

语法： CPFSLT f {,a}

操作数： $0 \le f \le 255$
$a \in [0,1]$

操作： (f) – (W)，
如果(f) < (W)则跳过
（无符号比较）

受影响的状态位： 无

编码：

0110	000a	ffff	ffff

说明： 通过执行无符号减法将数据存储单元f的内容与W的内容进行比较。
如果f的内容小于W的内容，则丢弃取出的指令，转而执行一条NOP指令，从而使该指令成为双周期指令。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```

执行指令前
PC = 地址（HERE）
W = ?
执行指令后
如果REG < W；
PC = 地址（LESS）
如果REG ≥ W；
PC = 地址（NLESS）

DAW **十进制调整W寄存器**

语法： DAW

操作数： 无

操作： 如果[W<3:0> > 9]或[DC = 1]，则
(W<3:0>) + 6 → W<3:0>；
否则
(W<3:0>) → W<3:0>；

如果[W<7:4> + DC > 9]或[C = 1]，则
(W<7:4>) + 6 + DC → W<7:4>；
否则
(W<7:4>) + DC → W<7:4>

受影响的状态位： C

编码：

0000	0000	0000	0111

说明： DAW调整W中的8位值（由两个变量先前的加法运算得出，每个都打包成BCD格式）并产生正确的打包BCD结果。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器W	处理数据	写W

示例1：
```
DAW
```
执行指令前
W = A5h
C = 0
DC = 0
执行指令后
W = 05h
C = 1
DC = 0

示例2：
执行指令前
W = CEh
C = 0
DC = 0
执行指令后
W = 34h
C = 1
DC = 0

DECF **f递减1**

语法： DECF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f) – 1 → 目标寄存器

受影响的状态位： C、DC、N、OV和Z

编码：

0000	01da	ffff	ffff

说明： 将寄存器f的内容递减1。如果d为0，结果存入W。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：
```
DECF    CNT,  1, 0
```
执行指令前
CNT = 01h
Z = 0
执行指令后
CNT = 00h
Z = 1

DECFSZ **f递减1，为0则跳过**

语法： DECFSZ f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f) – 1 → 目标寄存器，
如果结果 = 0则跳过

受影响的状态位： 无

编码：

0010	11da	ffff	ffff

说明： 将寄存器f的内容递减1。
如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果结果为0，则丢弃已经取指的下一条指令，代之执行一条NOP指令，使之成为一条双周期指令。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    DECFSZ  CNT, 1, 1
        GOTO    LOOP
CONTINUE
```
执行指令前
PC = 地址（HERE）
执行指令后
CNT = CNT - 1
如果CNT = 0；
PC = 地址（CONTINUE）
如果CNT ≠ 0；
PC = 地址（HERE + 2）

DCFSNZ **f递减1，不为0则跳过**

语法： DCFSNZ f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f) – 1 → 目标寄存器，
如果结果 ≠ 0 则跳过

受影响的状态位： 无

编码：

0100	11da	ffff	ffff

说明： 将寄存器f的内容递减1。
如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果结果不为0，则丢弃已经取指的下一条指令，代之执行一条NOP指令，使之成为一条双周期指令。

如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    DCFSNZ  TEMP, 1, 0
ZERO    :
NZERO   :
```
执行指令前
TEMP = ?
执行指令后
TEMP = TEMP – 1，
如果TEMP = 0；
PC = 地址（ZERO）
如果TEMP ≠ 0；
PC = 地址（NZERO）

GOTO **无条件跳转**

语法： GOTO k

操作数： 0 ≤ k ≤ 1048575

操作： k → PC<20:1>

受影响的状态位： 无

编码：

第1个字（k<7:0>）	1110	1111	$k_7$kkk	kkkk_0
第2个字（k<19:8>）	1111	k_{19}kkk	kkkk	kkkk_8

说明： GOTO允许无条件跳转到整个2 MB存储器范围内的任何位置。
20位值k被装入PC<20:1>。GOTO始终是一条双周期指令。

指令字数： 2

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数 k<7:0>	空操作	读立即数 k<19:8>，写入PC
空操作	空操作	空操作	空操作

示例：
```
GOTO THERE
```
执行指令后
PC = 地址（THERE）

INCF **f递增1**

语法： INCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f) + 1 → 目标寄存器

受影响的状态位： C、DC、N、OV和Z

编码：

0010	10da	ffff	ffff

说明： 将寄存器f的内容递增1。
如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：
```
INCF    CNT, 1, 0
```
执行指令前
CNT = FFh
Z = 0
C = ?
DC = ?
执行指令后
CNT = 00h
Z = 1
C = 1
DC = 1

INCFSZ **f递增1，为0则跳过**

语法： INCFSZ f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f) + 1 → 目标寄存器，
如果结果 = 0则跳过

受影响的状态位： 无

编码：

0011	11da	ffff	ffff

说明： 将寄存器f的内容递增1。
如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果结果为0，则丢弃已经取指的下一条指令，代之执行一条NOP指令，使之成为一条双周期指令。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    INCFSZ  CNT, 1, 0
NZERO   :
ZERO    :
```
执行指令前
PC = 地址（HERE）
执行指令后
CNT = CNT + 1
如果CNT = 0：
PC = 地址（ZERO）
如果CNT ≠ 0：
PC = 地址（NZERO）

INFSNZ **f递增1，不为0则跳过**

语法： INFSNZ f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f) + 1 → 目标寄存器，
如果结果 ≠ 0 则跳过

受影响的状态位： 无

编码：

0100	10da	ffff	ffff

说明： 将寄存器f的内容递增1。
如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果结果不为0，则丢弃已经取指的下一条指令，代之执行一条NOP指令，使之成为一条双周期指令。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    INFSNZ  REG, 1, 0
ZERO
NZERO
```
执行指令前
PC = 地址（HERE）
执行指令后
REG = REG + 1
如果REG ≠ 0；
PC = 地址（NZERO）
如果REG = 0；
PC = 地址（ZERO）

IORLW **立即数和W作逻辑或运算**

语法： IORLW k

操作数： 0 ≤ k ≤ 255

操作： (W) .OR. k → W

受影响的状态位： N和Z

编码：

0000	1001	kkkk	kkkk

说明： 将W寄存器的内容与8位立即数k进行逻辑或运算。结果存入W寄存器。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入W

示例：
```
IORLW   35h
```
执行指令前
W = 9Ah
执行指令后
W = BFh

IORWF **W与f作逻辑或运算**

语法： IORWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) .OR. (f) → 目标寄存器

受影响的状态位： N和Z

编码：

0001	00da	ffff	ffff

说明： W与寄存器f作逻辑或运算。如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：
```
IORWF  RESULT, 0, 1
```
执行指令前
RESULT = 13h
W = 91h
执行指令后
RESULT = 13h
W = 93h

LFSR **装载FSR**

语法： LFSR f, k

操作数： 0 ≤ f ≤ 2
0 ≤ k ≤ 16383

操作： k → FSRf

受影响的状态位： 无

编码：

1110 1111	1110 0000	$00k_{13}k$ k_7kkk	kkkk kkkk

说明： 将14位立即数k装入f指向的文件选择寄存器。

指令字数： 2

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k的MSB	处理数据	将立即数k的MSB写入FSRfH
译码	读立即数k的LSB	处理数据	将立即数k写入FSRfL

示例：
```
LFSR 2, 3ABh
```
执行指令后
FSR2H = 03h
FSR2L = ABh

MOVF **传送f**

语法： MOVF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： f → 目标寄存器

受影响的状态位： N和Z

编码：

0101	00da	ffff	ffff

说明： 根据d的状态，将寄存器f的内容传送到目标寄存器。
如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。存储单元f可以位于256字节存储区中的任何位置。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写W

示例： MOVF REG, 0, 0

执行指令前
REG = 22h
W = FFh
执行指令后
REG = 22h
W = 22h

MOVFF **将f中的内容送入f**

语法： MOVFF f_s,f_d

操作数： $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

操作： $(f_s) \rightarrow f_d$

受影响的状态位： 无

编码：

第1个字（源）	1100	ffff	ffff	$ffff_s$
第2个字（目标）	1111	ffff	ffff	$ffff_d$

说明： 将源寄存器f_s的内容传送到目标寄存器f_d。
源f_s的位置可以是4096字节数据空间（000h至FFFh）中的任何位置，目标f_d的位置也可以是000h至FFFh的任何位置。
MOVFF已经将源地址和目标地址范围缩小到存储器的低4 KB空间（存储区1至15）。对于其他范围，使用MOVFFL。

指令字数： 2

指令周期数： 2（3）

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f（源）	处理数据	空操作
译码	空操作 无空读	空操作	写入寄存器f（目标寄存器）

示例： MOVFF REG1, REG2

执行指令前
REG1 = 33h
REG2 = 11h
执行指令后
REG1 = 33h
REG2 = 33h

MOVFFL **将f中的内容送入f（长距离）**

语法： MOVFFL f_s,f_d

操作数： $0 \le f_s \le 16383$
$0 \le f_d \le 16383$

操作： $(f_s) \rightarrow f_d$

受影响的状态位： 无

编码：

第1个字	0000	0000	0110	$f_sf_sf_sf_s$
第2个字	1111	$f_sf_sf_sf_s$	$f_sf_sf_sf_s$	$f_sf_sf_df_d$
第3个字	1111	$f_df_df_df_d$	$f_df_df_df_d$	$f_df_df_df_d$

说明： 将源寄存器f_s的内容传送到目标寄存器f_d。源f_s的位置可以是16 KB数据空间（0000h至3FFFh）中的任何位置。
源或目标都可以是W（一个有用的特殊情况）。
MOVFFL对于将数据存储单元传输到外设寄存器（例如发送缓冲区或I/O端口）特别有用。
MOVFFL指令不能使用PCL、TOSU、TOSH或TOSL作为目标寄存器。

指令字数： 3

指令周期数： 3

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
译码	读取寄存器f_s（源）	处理数据	空操作
译码	空操作 无空读	空操作	写入寄存器f_d（目标）

示例： MOVFFL 2000h, 200Ah

执行指令前
2000h的内容 = 33h
200Ah的内容 = 11h
执行指令后
2000h的内容 = 33h
200Ah的内容 = 33h

MOVLB **将立即数送入BSR**

语法： MOVLW k

操作数： $0 \le k \le 63$

操作： $k \rightarrow BSR$

受影响的状态位： 无

编码：

0000	0001	00kk	kkkk

说明： 将6位立即数k装入存储区选择寄存器（BSR<5:0>）。BSR<7:6>的值始终保持为0。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	将立即数k写入BSR

示例： MOVLB 5

执行指令前
BSR寄存器 = 02h
执行指令后
BSR寄存器 = 05h

MOVLW **将立即数传送到W**

语法： MOVLW k

操作数： 0 ≤ k ≤ 255

操作： k → W

受影响的状态位： 无

编码：

0000	1110	kkkk	kkkk

说明： 将8位立即数k装入W。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入W

示例：
```
MOVLW   5Ah
```
执行指令后

W = 5Ah

MOVWF **将W的内容送入f**

语法： MOVWF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (W) → f

受影响的状态位： 无

编码：

0110	111a	ffff	ffff

说明： 将W的数据传送到寄存器f。
存储单元f可以位于256字节存储区中的任何位置。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入寄存器f

示例：
```
MOVWF   REG, 0
```
执行指令前

W = 4Fh
REG = FFh

执行指令后

W = 4Fh
REG = 4Fh

MULLW **立即数与W作乘法运算**

语法： MULLW k

操作数： 0 ≤ k ≤ 255

操作： (W) x k → PRODH:PRODL

受影响的状态位： 无

编码：

0000	1101	kkkk	kkkk

说明： 对W的内容和8位立即数k执行无符号乘法。16位结果存入PRODH:PRODL寄存器对。PRODH包含高字节。
W不变。
状态标志均不受影响。
请注意，此操作中既不可能溢出，也不可能进位。结果可能为零，但不会被检测到。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入寄存器PRODH:PRODL

示例： MULLW 0C4h

执行指令前

W = E2h
PRODH = ?
PRODL = ?

执行指令后

W = E2h
PRODH = ADh
PRODL = 08h

MULWF **W和f作乘法运算**

语法： MULWF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (W) x (f) → PRODH:PRODL

受影响的状态位： 无

编码：

0000	001a	ffff	ffff

说明： 对W的内容和寄存器文件存储单元k执行无符号乘法。16位结果存入PRODH:PRODL 寄存器对。PRODH包含高字节。W和f均不变。
状态标志均不受影响。
请注意，此操作中既不可能溢出，也不可能进位。结果可能为零，但不会被检测到。
如果a为0，则选择快速操作存储区。
如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入寄存器PRODH:PRODL

示例： MULWF REG, 1

执行指令前

W = C4h
REG = B5h
PRODH = ?
PRODL = ?

执行指令后

W = C4h
REG = B5h
PRODH = 8Ah
PRODL = 94h

NEGF **将f取反**

语法： NEGF f {,a}

操作数： $0 \le f \le 255$
$a \in [0,1]$

操作： $(\overline{f}) + 1 \rightarrow f$

受影响的状态位： N、OV、C、DC和Z

编码：

0110	110a	ffff	ffff

说明： 使用二进制补码对存储单元f进行取反。结果放入数据存储单元f。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入寄存器f

示例：
```
NEGF    REG, 1
```
执行指令前
REG = 0011 1010 [3Ah]
执行指令后
REG = 1100 0110 [C6h]

NOP **空操作**

语法： NOP

操作数： 无

操作： 空操作

受影响的状态位： 无

编码：

0000 1111	0000 xxxx	0000 xxxx	0000 xxxx

说明： 不执行任何操作。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作

示例：
无。

POP **将返回栈顶的内容弹出**

语法： POP

操作数： 无

操作： （TOS） → 位桶

受影响的状态位： 无

编码：

0000	0000	0000	0110

说明： 将TOS值从返回堆栈中取出并丢弃。然后，TOS值就变成了压入返回堆栈的前一个值。
提供此指令是为了使用户能够正确地管理返回堆栈以集成软件堆栈。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	将TOS值弹出	空操作

示例：
```
POP
GOTO      新状态
```
执行指令前
TOS = 0031A2h
堆栈（1级） = 014332h

执行指令后
TOS = 014332h
PC = NEW

PUSH **将返回栈顶的内容压入**

语法： PUSH

操作数： 无

操作： （PC + 2） → TOS

受影响的状态位： 无

编码：

0000	0000	0000	0101

说明： 将PC + 2压入返回栈顶。将前一个TOS值压入堆栈。
此指令允许通过修改TOS并将其压入返回堆栈来实现软件堆栈。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	将PC + 2压入返回堆栈	空操作	空操作

示例：
```
PUSH
```
执行指令前
TOS = 345Ah
PC = 0124h

执行指令后
PC = 0126h
TOS = 0126h
堆栈（1级） = 345Ah

RCALL **相对调用**

语法： RCALL n

操作数： -1024 ≤ n ≤ 1023

操作： (PC) + 2 → TOS，
(PC) + 2 + 2n → PC

受影响的状态位： 无

编码：

1101	1nnn	nnnn	nnnn

说明： 子程序调用可以从当前位置跳转到最远1K的位置。首先，将返回地址（PC + 2）压入堆栈。然后，将二进制补码2n加到PC。由于PC将递增以便取出下一条指令，所以新地址将为PC + 2 + 2n。该指令为一条双周期指令。

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数n 将PC压入堆栈	处理数据	写入PC
空操作	空操作	空操作	空操作

示例： HERE RCALL Jump

执行指令前
PC = 地址（HERE）
执行指令后
PC = 地址（Jump）
TOS = 地址（HERE + 2）

RESET **复位**

语法： RESET

操作数： 无

操作： 复位受$\overline{\text{MCLR}}$复位影响的所有寄存器和标志。

受影响的状态位： 全部

编码：

0000	0000	1111	1111

说明： 此指令可实现用软件执行$\overline{\text{MCLR}}$复位。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	开始复位	空操作	空操作

示例： RESET

执行指令后
寄存器 = 复位值
标志* = 复位值

RETFIE **从中断返回**

语法： RETFIE {s}

操作数： s ∈ [0,1]

操作： (TOS) → PC，
如果s = 1，将内容从相应的影子寄存器存储到WREG、STATUS、BSR、FSR0H、FSR0L、FSR1H、FSR1L、FSR2H、FSR2L、PRODH、PRODL、PCLATH和PCLATU寄存器中。

如果s = 0，所有寄存器的状态均没有变化。

受影响的状态位： INTCON1寄存器中的STAT<1:0>

编码：

0000	0000	0001	000s

说明： 从中断返回。执行出栈操作，将栈顶（TOS）的内容装入PC。通过将高优先级或低优先级全局中断允许位置1来允许中断。
如果s = 1，影子寄存器WREG、STATUS、BSR、FSR0H、FSR0L、FSR1H、FSR1L、FSR2H、FSR2L、PRODH、PRODL、PCLATH和PCLATU的内容装入相应的寄存器中。有主程序现场和低优先级现场两组影子寄存器。执行RETFIE指令时检索的影子寄存器组取决于执行RETFIE时CPU的工作状态。如果s = 0，则不会更新这些寄存器（默认）。

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	空操作	将堆栈的内容弹出到PC 将GIEH或GIEL置1
空操作	空操作	空操作	空操作

示例： RETFIE 1

中断后

PC	=	TOS
WREG	=	WREG_SHAD
BSR	=	BSR_SHAD
STATUS	=	STATUS_SHAD
FSR0L/H	=	FSR0L/H_SHAD
FSR1L/H	=	FSR1L/H_SHAD
FSR2L/H	=	FSR2L/H_SHAD
PROD/H	=	PROD/H_SHAD
PCLATH/U	=	PCLATH/U_SHAD

RETLW **将立即数返回W**

语法： RETLW k

操作数： 0 ≤ k ≤ 255

操作： k → W，
(TOS) → PC，
PCLATU和PCLATH不变

受影响的状态位： 无

编码：

0000	1100	kkkk	kkkk

说明： 将8位立即数k装入W。将栈顶内容（返回地址）装入程序计数器。高地址锁存（PCLATH）保持不变。

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	将堆栈的内容弹出到PC，写入W
空操作	空操作	空操作	空操作

示例：

```
    CALL TABLE ; W contains table
               ; offset value
               ; W now has
               ; table value
    :
TABLE
    ADDWF PCL  ; W = offset
    RETLW k0   ; Begin table
    RETLW k1   ;
    :
    :
    RETLW kn   ; End of table
```

执行指令前
W = 07h
执行指令后
W = kn值

RETURN 从子程序返回

语法： RETURN {s}

操作数： s ∈ [0,1]

操作： (TOS) → PC，
如果s = 1
(WS) → W，
(STATUSS) → Status，
(BSRS) → BSR，
PCLATU和PCLATH不变

受影响的状态位： 无

编码：

0000	0000	0001	001s

说明： 从子程序返回。执行出栈操作，将栈顶（TOS）内容装入程序计数器。如果s = 1，影子寄存器WS、STATUSS和BSRS的内容装入它们相应的W、Status和BSR寄存器中。
如果s = 0，则不会更新这些寄存器（默认）。

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	将堆栈的内容弹出到PC
空操作	空操作	空操作	空操作

示例： RETURN

执行指令后：
PC = TOS

RLCF f带进位循环左移

语法： RLCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

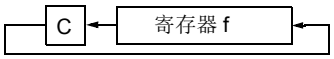
操作： (f<n>) → dest<n + 1>，
(f<7>) → C，
(C) → dest<0>

受影响的状态位： C、N和Z

编码：

0011	01da	ffff	ffff

说明： 将寄存器f的内容连同进位标志位一起循环左移1位。如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。
如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。



指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：
```
RLCF    REG, 0, 0
```
执行指令前
REG = 1110 0110
C = 0
执行指令后
REG = 1110 0110
W = 1100 1100
C = 1

RLNCF **f循环左移（不带进位）**

语法： RLNCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

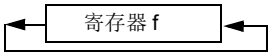
操作： (f<n>) → dest<n + 1>,
(f<7>) → dest<0>

受影响的状态位： N和Z

编码：

0100	01da	ffff	ffff

说明： 将寄存器f的内容循环左移1位。如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。



指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：
```
RLNCF   REG, 1, 0
执行指令前
  REG  =  1010 1011
执行指令后
  REG  =  0101 0111
```

RRCF **f带进位循环右移**

语法： RRCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

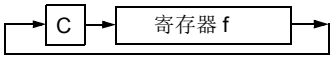
操作： (f<n>) → dest<n – 1>,
(f<0>) → C,
(C) → dest<7>

受影响的状态位： C、N和Z

编码：

0011	00da	ffff	ffff

说明： 将寄存器f的内容连同进位标志位一起循环右移1位。如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。



指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：
```
RRCF    REG, 0, 0
执行指令前
  REG  =  1110 0110
  C    =  0
执行指令后
  REG  =  1110 0110
  W    =  0111 0011
  C    =  0
```

RRNCF **f循环右移（不带进位）**

语法： RRNCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

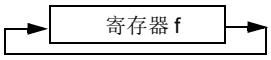
操作： (f<n>) → dest<n – 1>,
(f<0>) → dest<7>

受影响的状态位： N和Z

编码：

0100	00da	ffff	ffff

说明： 将寄存器f的内容循环右移1位。如果d为0，结果放入W寄存器。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区（默认），同时改写BSR值。如果a为1，则根据BSR值选择存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。



指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例1：
```
RRNCF   REG, 1, 0
```
执行指令前
REG = 1101 0111
执行指令后
REG = 1110 1011

示例2：
```
RRNCF   REG, 0, 0
```
执行指令前
W = ?
REG = 1101 0111
执行指令后
W = 1110 1011
REG = 1101 0111

SETF **将f置1**

语法： SETF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： FFh → f

受影响的状态位： 无

编码：

0110	100a	ffff	ffff

说明： 指定寄存器的内容会设置为FFh。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入寄存器f

示例：
```
SETF    REG, 1
```
执行指令前
REG = 5Ah
执行指令后
REG = FFh

SLEEP **进入休眠模式**

语法： SLEEP

操作数： 无

操作： 00h → WDT，
0 → WDT后分频器，
1 → $\overline{TO}$，
0 → PD

受影响的状态位： $\overline{TO}$和$\overline{PD}$

编码：

0000	0000	0000	0011

说明： 掉电状态位（$\overline{PD}$）被清零。超时状态位（$\overline{TO}$）被置1。看门狗定时器及其后分频器被清零。
振荡器停振，处理器进入休眠模式。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	进入休眠状态

示例： SLEEP

执行指令前
$\overline{TO}$ = ?
$\overline{PD}$ = ?

执行指令后
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† 如果WDT导致唤醒，则该位将清零。

SUBFSR **从FSR中减去立即数**

语法： SUBFSR f, k

操作数： 0 ≤ k ≤ 63
f ∈ [0, 1, 2]

操作： FSR(f) – k → FSRf

受影响的状态位： 无

编码：

1110	1001	ffkk	kkkk

说明： 从f指定的FSR的内容中减去6位立即数k。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例： SUBFSR 2, 23h

执行指令前
FSR2 = 03FFh

执行指令后
FSR2 = 03DCh

SUBFWB **W减去f（带借位）**

语法： SUBFWB f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

操作： (W) – (f) – ($\overline{C}$) → 目标寄存器

受影响的状态位： N、OV、C、DC和Z

编码：

0101	01da	ffff	ffff

说明： 用W的内容减去寄存器f的内容和进位标志（借位）（通过二进制补码方式进行运算）。如果d为0，结果存入W。如果d为1，结果存入寄存器f（默认）。
如果a为0，则选择快速操作存储区。
如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例1： SUBFWB REG, 1, 0

执行指令前
REG = 3
W = 2
C = 1

执行指令后
REG = FF
W = 2
C = 0
Z = 0
N = 1 ；结果为负

示例2： SUBFWB REG, 0, 0

执行指令前
REG = 2
W = 5
C = 1

执行指令后
REG = 2
W = 3
C = 1
Z = 0
N = 0 ；结果为正

示例3： SUBFWB REG, 1, 0

执行指令前
REG = 1
W = 2
C = 0

执行指令后
REG = 0
W = 2
C = 1
Z = 1 ；结果为零
N = 0

SUBLW **立即数减去W**

语法： SUBLW k

操作数： $0 \le k \le 255$

操作： k – (W) → W

受影响的状态位： N、OV、C、DC和Z

编码：

0000	1000	kkkk	kkkk

说明 用8位立即数k减去W的内容。
结果存入W寄存器。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入W

示例1： SUBLW 02h

执行指令前
W = 01h
C = ?

执行指令后
W = 01h
C = 1 ；结果为正
Z = 0
N = 0

示例2： SUBLW 02h

执行指令前
W = 02h
C = ?

执行指令后
W = 00h
C = 1 ；结果为零
Z = 1
N = 0

示例3： SUBLW 02h

执行指令前
W = 03h
C = ?

执行指令后
W = FFh ；（以二进制补码方式进行）
C = 0 ；结果为负
Z = 0
N = 1

SUBWF　　f减去W

语法：　SUBWF　f {,d {,a}}

操作数：　0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作：　(f) – (W) → 目标寄存器

受影响的状态位：　N、OV、C、DC和Z

编码：

0101	11da	ffff	ffff

说明：　用寄存器f的内容减去W的内容（通过二进制补码方式进行运算）。
如果d为0，结果存入W。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。
如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数：　1

指令周期数：　1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例1：
```
SUBWF   REG, 1, 0
执行指令前
    REG =  3
    W   =  2
    C   =  ?
执行指令后
    REG =  1
    W   =  2
    C   =  1   ; 结果为正
    Z   =  0
    N   =  0
```

示例2：
```
SUBWF   REG, 0, 0
执行指令前
    REG =  2
    W   =  2
    C   =  ?
执行指令后
    REG =  2
    W   =  0
    C   =  1   ; 结果为零
    Z   =  1
    N   =  0
```

示例3：
```
SUBWF   REG, 1, 0
执行指令前
    REG =  1
    W   =  2
    C   =  ?
执行指令后
    REG =  FFh ; （以二进制补码方式进行）
    W   =  2
    C   =  0   ; 结果为负
    Z   =  0
    N   =  1
```

SUBWFB　　f减去W（带借位）

语法：　SUBWFB　f {,d {,a}}

操作数：　0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作：　(f) – (W) – $(\overline{C})$ → 目标寄存器

受影响的状态位：　N、OV、C、DC和Z

编码：

0101	10da	ffff	ffff

说明：　用寄存器f的内容减去W的内容和进位标志（借位）（通过二进制补码方式进行运算）。如果d为0，结果存入W。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数：　1

指令周期数：　1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例1：
```
SUBWFB  REG, 1, 0
执行指令前
    REG =  19h    (0001 1001)
    W   =  0Dh    (0000 1101)
    C   =  1
执行指令后
    REG =  0Ch    (0000 1100)
    W   =  0Dh    (0000 1101)
    C   =  1
    Z   =  0
    N   =  0      ; 结果为正
```

示例2：
```
SUBWFB  REG, 0, 0
执行指令前
    REG =  1Bh    (0001 1011)
    W   =  1Ah    (0001 1010)
    C   =  0
执行指令后
    REG =  1Bh    (0001 1011)
    W   =  00h
    C   =  1
    Z   =  1      ; 结果为零
    N   =  0
```

示例3：
```
SUBWFB  REG, 1, 0
执行指令前
    REG =  03h    (0000 0011)
    W   =  0Eh    (0000 1110)
    C   =  1
执行指令后
    REG =  F5h    (1111 0101)
                  ; [以二进制补码方式进行]
    W   =  0Eh    (0000 1110)
    C   =  0
    Z   =  0
    N   =  1      ; 结果为负
```

SWAPF **交换f**

语法： SWAPF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f<3:0>) → dest<7:4>，
(f<7:4>) → dest<3:0>

受影响的状态位： 无

编码：

0011	10da	ffff	ffff

说明： 寄存器f的高半字节和低半字节相互交换。如果d为0，结果放入W寄存器。如果d为1，结果存入寄存器f（默认）。如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：

```
SWAPF   REG, 1, 0
```

执行指令前
REG = 53h
执行指令后
REG = 35h

TBLRD **表读**

语法： TBLRD (*; *+; *-; +*)

操作数： 无

操作： 如果TBLRD *,
(Prog Mem (TBLPTR)) → TABLAT；
TBLPTR – 不改变；
如果TBLRD *+,
(Prog Mem (TBLPTR)) → TABLAT；
(TBLPTR) + 1 → TBLPTR；
如果TBLRD *-,
(Prog Mem (TBLPTR)) → TABLAT；
(TBLPTR) – 1 → TBLPTR；
如果TBLRD +*,
(TBLPTR) + 1 → TBLPTR；
(Prog Mem (TBLPTR)) → TABLAT；

受影响的状态位： 无

编码：

0000	0000	0000	10nn nn=0 * =1 *+ =2 *– =3 +*

说明： 该指令用于读取程序存储器（P.M.）的内容。若要寻址程序存储器，使用称为表指针（TBLPTR）的指针。
TBLPTR（21位指针）指向程序存储器中的每个字节。TBLPTR具有2 MB地址范围。

TBLPTR[0] = 0： 程序存储字的最低有效字节
TBLPTR[0] = 1： 程序存储字的最高有效字节

TBLRD指令可以按如下方式修改TBLPTR的值：

- 不改变
- 后递增
- 后递减
- 预递增

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作（读取程序存储器）	空操作	空操作（写入TABLAT）

TBLRD **表读（续）**

示例1：
```
TBLRD  *+ ;
```
执行指令前
TABLAT = 55h
TBLPTR = 00A356h
存储器（00A356h） = 34h
执行指令后
TABLAT = 34h
TBLPTR = 00A357h

示例2：
```
TBLRD  +* ;
```
执行指令前
TABLAT = AAh
TBLPTR = 01A357h
存储器（01A357h） = 12h
存储器（01A358h） = 34h
执行指令后
TABLAT = 34h
TBLPTR = 01A358h

TBLWT **表写**

语法： TBLWT (*; *+; *-; +*)

操作数： 无

操作： 如果TBLWT*，
(TABLAT) → 保持寄存器；
TBLPTR – 不改变；
如果TBLWT*+，
(TABLAT) → 保持寄存器；
(TBLPTR) + 1 → TBLPTR；
如果TBLWT*-，
(TABLAT) → 保持寄存器；
(TBLPTR) – 1 → TBLPTR；
如果TBLWT+*，
(TBLPTR) + 1 → TBLPTR；
(TABLAT) → 保持寄存器；

受影响的状态位： 无

编码：

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

说明： 该指令使用 TBLPTR 的低三位来确定将TABLAT写入八个保持寄存器中的哪一个。保持寄存器用于编程程序存储器（P.M.）的内容。（有关编程闪存的更多详细信息，请参见**第13.1节“闪存程序存储器”**。）
TBLPTR（21位指针）指向程序存储器中的每个字节。TBLPTR具有2 MB地址范围。TBLPTR的LSb选择要访问的程序存储单元的字节。

TBLPTR[0] = 0： 程序存储字的最低有效字节
TBLPTR[0] = 1： 程序存储字的最高有效字节

TBLWT指令可以按如下方式修改TBLPTR的值：

- 不改变
- 后递增
- 后递减
- 预递增

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作（读取TABLAT）	空操作	空操作（写入保持寄存器）

TBLWT **表写（续）**

示例1： `TBLWT *+;`

执行指令前
TABLAT = 55h
TBLPTR = 00A356h
保持寄存器（00A356h） = FFh
执行指令后（表写完成）
TABLAT = 55h
TBLPTR = 00A357h
保持寄存器（00A356h） = 55h

示例2： `TBLWT +*;`

执行指令前
TABLAT = 34h
TBLPTR = 01389Ah
保持寄存器（01389Ah） = FFh
保持寄存器（01389Bh） = FFh
执行指令后（表写完成）
TABLAT = 34h
TBLPTR = 01389Bh
保持寄存器（01389Ah） = FFh
保持寄存器（01389Bh） = 34h

TSTFSZ **测试f，为0则跳过**

语法： TSTFSZ f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： 如果f = 0，则跳过

受影响的状态位： 无

编码：

0110	011a	ffff	ffff

说明： 如果f = 0，则丢弃执行当前指令的过程中取指的下一条指令，执行一条NOP指令，使之成为一条双周期指令。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过且后跟双字指令，则为3个周期。

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	空操作

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过且后跟双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```

执行指令前
PC = 地址（HERE）
执行指令后
如果CNT = 00h，
PC = 地址（ZERO）
如果CNT ≠ 00h，
PC = 地址（NZERO）

XORLW **立即数和W作逻辑异或运算**

语法： XORLW k

操作数： 0 ≤ k ≤ 255

操作： (W) .XOR. k → W

受影响的状态位： N和Z

编码：

0000	1010	kkkk	kkkk

说明： 将W的内容与8位立即数k进行逻辑异或运算。结果存入W寄存器。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入W

示例：

```
XORLW   0AFh
```

执行指令前
W = B5h
执行指令后
W = 1Ah

XORWF **W和f作逻辑异或运算**

语法： XORWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) .XOR. (f) → 目标寄存器

受影响的状态位： N和Z

编码：

0001	10da	ffff	ffff

说明： 将W的内容与寄存器f的内容进行逻辑异或运算。如果d为0，结果存入W。如果d为1，结果存回寄存器f（默认）。
如果a为0，则选择快速操作存储区。如果a为1，BSR用于选择GPR存储区。
如果a为0且使能了扩展指令集，则只要f ≤ 95（5Fh），该指令就会在立即数变址寻址模式下工作。有关详细信息，请参见**第42.2.3节“立即数变址模式下针对字节和针对位的指令”**。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例：
```
XORWF   REG, 1, 0
```
执行指令前
REG = AFh
W = B5h
执行指令后
REG = 1Ah
W = B5h

42.2 扩展指令集

除PIC18指令集的标准75指令外，PIC18(L)F25/26K83器件还提供核心CPU功能的可选扩展。增加的功能包括八条附加指令，用于增强间接和变址寻址操作以及许多标准PIC18指令的立即数变址寻址模式的实现。

默认情况下，禁用扩展指令集的附加功能。要使能这些功能，用户必须将XINST配置位置1。

扩展集中的指令都可以归类为立即数操作，可以操作文件选择寄存器，也可以将它们用于变址寻址。其中的ADDFSR和SUBFSR两条指令，各有一个使用FSR2的附加特殊实例。这些版本（ADDULNK和SUBULNK）允许执行后自动返回。

扩展指令专门用于优化以高级语言（特别是C语言）编写的可重入程序代码（即递归或使用软件堆栈的代码）。此外，它们支持使用高级语言工作的用户更有效地对数据结构执行某些操作。这些模块包括：

- 进入和离开子程序时动态分配和释放软件堆栈空间
- 函数指针调用
- 软件堆栈指针操作
- 处理位于软件堆栈中的变量

表42-3中提供了扩展指令集中的指令汇总。**第42.2.2节“扩展指令集”**中给出了详细说明。表42-1中的操作码字段说明适用于标准和扩展PIC18指令集。

注：	指令集扩展和立即数变址寻址模式专为优化以C语言编写的应用程序而设计。用户可能永远不会直接在汇编器中使用这些指令。对于可能正在查看由编译器生成的代码的用户，提供这些命令的语法作为参考。

42.2.1 扩展指令语法

大多数扩展指令使用变址参数，使用一个文件选择寄存器和一些偏移量来指定源或目标寄存器。当指令的参数作为变址寻址的一部分时，参数将包含在方括号（“[]”）中。这样做是为了指示该参数用作变址或偏移量。如果MPASM汇编器确定变址或偏移值未包含在方括号中，则会标记错误。

使能扩展指令集时，括号也用于指示针对字节和针对位的指令中的变址参数。这是对其语法的其他更改的补充。更多详细信息，请参见**第42.2.3.1节“标准PIC18命令的扩展指令语法”**。

注：	过去，方括号曾用于表示PIC18和早期指令集中的可选参数。在本文和之后，可选参数由花括号（“{}”）表示。

表42-3： PIC18指令集的扩展

助记符，操作数		说明	周期数	16位指令字				受影响的状态位
				MSb			LSb	
ADDULNK	k	将FSR2与（k）相加并返回	2	1110	1000	11kk	kkkk	无
MOVSF	z_s和f_d	将z_s（源）中的内容送入 第1个字	2	1110	1011	0zzz	zzzz	无
		f_d（目标） 第2个字	2	1111	ffff	ffff	ffff	
MOVSFL	z_s和f_d	操作码 第1个字		0000	0000	0000	0010	无
		将z_s（源）中的内容送入 第2个字	3	1111	xxxz	zzzz	zzff	
		f_d（满目标） 第3个字		1111	ffff	ffff	ffff	
MOVSS	z_s和z_d	将z_s（源）中的内容送入 第1个字		1110	1011	1zzz	zzzz	无
		z_d（目标） 第2个字	2	1111	xxxx	xzzz	zzzz	
PUSHL	k	将立即数压入POSTDEC2	1	1110	1010	kkkk	kkkk	无
SUBULNK	k	从FSR2中减去（k）并返回	2	1110	1001	11kk	kkkk	无

注 1： 如果程序计数器（PC）被修改或条件测试结果为真，则该指令需要一个额外的周期。额外的周期执行一条NOP指令。

2： 有些指令是多字指令。这些指令的第二个/第三个字将被解码为NOP，除非指令的第一个字检索嵌入在这16位中的信息。这确保了所有程序存储单元都有一个有效的指令。

3： 仅在使能扩展指令集时可用。

4： f_s和f_d不覆盖整个存储器范围。存储区选择的高2位被强制为b00以限制这些指令的范围为低4k地址空间。

42.2.2 扩展指令集

ADDULNK **将立即数加到FSR2并返回**

语法： ADDULNK k

操作数： $0 \le k \le 63$

操作： FSR2 + k → FSR2，
(TOS) → PC

受影响的状态位： 无

编码：

1110	1000	11kk	kkkk

说明： 将6位立即数k加到FSR2的内容。然后通过向PC装入TOS来执行RETURN。
该指令需要两个周期来执行；在第二个周期执行NOP。
可认为该指令是ADDFSR指令的特例，其中f = 3（二进制为11）；它仅在FSR2上工作。

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读立即数k	处理数据	写入FSR
空操作	空操作	空操作	空操作

示例： ADDULNK 23h

执行指令前
FSR2 = 03FFh
PC = 0100h

执行指令后
FSR2 = 0422h
PC = (TOS)

注： 所有PIC18指令都可以在指令助记符之前采用可选的标号参数，以用于符号寻址。如果使用标号，则指令语法变为：{标号} 指令参数。

MOVSF **将变址的内容送入f**

语法： MOVSF [z_s], f_d

操作数： $0 \le z_s \le 127$
$0 \le f_d \le 4095$

操作： ((FSR2) + z_s) → f_d

受影响的状态位： 无

编码：

第1个字（源）	1110	1011	0zzz	$zzzz_s$
第2个字（目标）	1111	ffff	ffff	$ffff_d$

说明： 将源寄存器的内容传送到目标寄存器f_d。通过将第一个字中的7位立即数偏移z_s与FSR2的值相加来确定源寄存器的实际地址。目标寄存器的地址由第二个字中的12位立即数f_d指定。两个地址可以是4096字节数据空间（000h至FFFh）中的任何位置。
MOVSF已经将目标范围缩小到存储器的低4 KB空间（存储区1至15）。对于其他范围，使用MOVSFL。

指令字数： 2

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读取源寄存器
译码	空操作 无空读	空操作	写入寄存器f（目标寄存器）

示例： MOVSF [05h], REG2

执行指令前
FSR2 = 80h
85h的内容 = 33h
REG2 = 11h
执行指令后
FSR2 = 80h
85h的内容 = 33h
REG2 = 33h

MOVSFL **将变址的内容送入f（长距离）**

语法： MOVSFL [z_s], f_d

操作数： $0 \le z_s \le 127$
$0 \le f_d \le 16383$

操作： ((FSR2) + z_s) → f_d

受影响的状态位： 无

编码：

第1个字（操作码）	0000	0000	0110	0010
第2个字（源）	1111	xxxz	zzzz	zz_sff
第3个字（完整目标）	1111	ffff	ffff	$ffff_d$

说明： 将源寄存器的内容传送到目标寄存器f_d。通过将第一个字中的7位立即数偏移z_s与FSR2（14位）的值相加来确定源寄存器的实际地址。目标寄存器的地址由第二个字中的14位立即数f_d指定。两个地址可以是16 KB数据空间（0000h至3FFFh）中的任何位置。
MOVSFL 指令不能使用 PCL、TOSU、TOSH或TOSL作为目标寄存器。如果结果源地址指向间接寻址寄存器，则返回的值将为00h。

指令字数： 3

指令周期数： 3

Q周期活动：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
译码	读取寄存器z（源）	处理数据	空操作
译码	空操作 无空读	空操作	写入寄存器f（目标）

示例： MOVSFL [05h], REG2

执行指令前
FSR2 = 80h
85h的内容 = 33h
REG2 = 11h
执行指令后
FSR2 = 80h
85h的内容 = 33h

MOVSS **通过变址寻址传递数据**

语法： MOVSS [z_s], [z_d]

操作数： $0 \le z_s \le 127$
$0 \le z_d \le 127$

操作： ((FSR2) + z_s) → ((FSR2) + z_d)

受影响的状态位： 无

编码：

第1个字（源）	1110	1011	1zzz	zzzz$_s$
第2个字（目标）	1111	xxxx	xzzz	zzzz$_d$

说明 将源寄存器的内容传送到目标寄存器。通过将7位立即数偏移z_s或z_d分别与FSR2的值相加来确定源寄存器和目标寄存器的地址。
两个寄存器可以位于4096字节数据存储器空间（000h至FFFh）中的任何位置。
MOVSS 指令不能使用 PCL、TOSU、TOSH或TOSL作为目标寄存器。
如果结果源地址指向间接寻址寄存器，则返回的值将为00h。如果结果目标地址指向间接寻址寄存器，则指令将作为NOP指令执行。

指令字数： 2

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读取源寄存器
译码	确定目标地址	确定目标地址	写入目标寄存器

示例：
```
MOVSS [05h], [06h]
```
执行指令前
FSR2内容 = 80h
85h的内容 = 33h
86h内容 = 11h
执行指令后
FSR2内容 = 80h
85h的内容 = 33h
86h内容 = 33h

PUSHL **将立即数存入FSR2，FSR2递减1**

语法： PUSHL k

操作数： $0 \le k \le 255$

操作： k → (FSR2),
FSR2 – 1 → FSR2

受影响的状态位： 无

编码：

1111	1010	kkkk	kkkk

说明： 将8位立即数k写入FSR2指定的数据存储器地址。在该操作后，将FSR2递减1。
该指令允许用户将值压入软件堆栈。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取k	处理数据	写入目标

示例：
```
PUSHL 08h
```
执行指令前
FSR2H:FSR2L = 01ECh
存储器（01ECh） = 00h

执行指令后
FSR2H:FSR2L = 01EBh
存储器（01ECh） = 08h

SUBULNK **从FSR2中减去立即数并返回**

语法： SUBULNK k

操作数： $0 \le k \le 63$

操作： FSR2 - k → FSR2

(TOS) → PC

受影响的状态位： 无

编码：

1110	1001	11kk	kkkk

说明： 从FSR2的内容中减去6位立即数k。然后通过向PC装入TOS来执行RETURN。该指令需要两个周期来执行；在第二个周期执行NOP。

可认为该指令是SUBFSR指令的特例，其中f = 3（二进制为11）；它仅在FSR2上工作。

指令字数： 1

指令周期数： 2

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标
空操作	空操作	空操作	空操作

示例：

```
SUBULNK 23h
```

执行指令前

FSR2 = 03FFh

PC = 0100h

执行指令后

FSR2 = 03DCh

PC = (TOS)

42.2.3 立即数变址模式下针对字节和针对位的指令

注： 使能PIC18指令集扩展可能会导致传统应用程序表现异常或完全失效。

除扩展集中的八个新命令外，使能扩展指令集还将使能立即数变址寻址模式（**第4.8.1节“使用立即数偏移量进行变址寻址”**）。这会显著影响解释标准PIC18指令集的许多命令的方式。

禁用扩展集时，嵌入操作码中的地址被视为立即数存储单元：快速操作存储区中的存储单元（a = 0），或者由BSR指定的GPR存储区（a = 1）。但是，当使能扩展指令集且a = 0时，5Fh或更小的文件寄存器参数被解释为FSR2中指针值的偏移量，而不是立即数地址。在实际应用中，这意味着当使能扩展指令集时，所有使用访问RAM位作为参数的指令（即所有针对字节和针对位的指令，或几乎一半的核心PIC18指令）可能具有不同的行为。

当FSR2的内容为00h时，访问RAM的边界基本上会重新映射到其原始值。这在创建向后兼容代码时可能很有用。如果使用此技术，可能需要保存FSR2的值，并在C和汇编程序之间来回移动时将其恢复，以便保留堆栈指针。用户还必须牢记扩展指令集的语法要求（见**第42.2.3.1节“标准PIC18命令的扩展指令语法”**）。

尽管立即数变址寻址模式对于动态堆栈和指针操作非常有用，但如果在错误的寄存器上执行简单的算术运算，它也会非常麻烦。习惯PIC18编程的用户必须记住，当使能扩展指令集时，5Fh或更小的寄存器地址用于立即数变址寻址。

下页提供了立即数变址寻址模式中典型的针对字节和针对位的指令的代表示例，以说明如何对执行产生影响。示例中显示的操作数条件适用于这些类型的所有指令。

42.2.3.1 标准PIC18命令的扩展指令语法

当使能扩展指令集时，标准的针对字节和针对位的命令中的文件寄存器参数f将替换为立即数偏移值k。如前所述，这仅在f小于或等于5Fh时发生。使用偏移值时，必须用方括号（“[ ]”）表示。与扩展指令一样，括号的使用向编译器指示了该值将被解释为变址或偏移量。省略括号或在括号内使用大于5Fh的值将在MPASM汇编器中生成错误。

如果变址参数被正确地括起来用于立即数变址寻址，则无需指定访问RAM参数，会自动将其假设为0。基于目标地址将a置1时，这与标准操作（禁用扩展指令集）不同。在此模式下声明访问RAM位也会在MPASM汇编器中生成错误。

目标参数d的功能与之前相同。

在最新版本的MPASM汇编器中，必须显式调用扩展指令集的语言支持。这可以通过命令行选项/y或源列表中的PE指令来完成。

42.2.4 使能扩展指令集时的注意事项

需要注意的是，对指令集的扩展可能并不是对所有用户都有利。特别是，不编写使用软件堆栈的代码的用户可能无法从使用指令集的扩展中获益。

此外，立即数变址寻址模式可能会产生与写入PIC18汇编器的传统应用程序相关的问题。原因在于，传统代码中的指令可能试图在5Fh以下的快速操作存储区中寻址寄存器。由于在使能指令集扩展时这些地址被解释为FSR2的立即数偏移，因此应用程序可能会读取或写入错误的数据地址。

将应用程序移植到PIC18(L)F25/26K83时，考虑代码的类型非常重要。使用指令集扩展时，使用C编写并将受益于高效编译的大型可重入应用程序将顺利运行。大量使用快速操作存储区的传统应用程序很可能无法从使用扩展指令集中获益。

ADDWF **将W与变址相加（立即数变址模式）**

语法： ADDWF [k] {,d}

操作数： $0 \le k \le 95$
$d \in [0,1]$

操作： (W) + ((FSR2) + k) → 目标寄存器

受影响的状态位： N、OV、C、DC和Z

编码：

0010	01d0	kkkk	kkkk

说明： 将W的内容与FSR2指示的偏移值k的寄存器的内容相加。
如果d为0，结果存入W。如果d为1，结果存回寄存器f（默认）。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取k	处理数据	写入目标

示例： ADDWF [OFST], 0

执行指令前
W = 17h
OFST = 2Ch
FSR2 = 0A00h
0A2Ch的内容 = 20h

执行指令后
W = 37h
0A2Ch的内容 = 20h

BSF **位置1变址（立即数变址模式）**

语法： BSF [k], b

操作数： $0 \le f \le 95$
$0 \le b \le 7$

操作： 1 → ((FSR2) + k)<b>

受影响的状态位： 无

编码：

1000	bbb0	kkkk	kkkk

说明： 将FSR2指示的寄存器（偏移值为k）的位b置1。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取寄存器f	处理数据	写入目标

示例： BSF [FLAG_OFST], 7

执行指令前
FLAG_OFST = 0Ah
FSR2 = 0A00h
0A0Ah的内容 = 55h

执行指令后
0A0Ah的内容 = D5h

SETF **置1变址（立即数变址模式）**

语法： SETF [k]

操作数： $0 \le k \le 95$

操作： FFh → ((FSR2) + k)

受影响的状态位： 无

编码：

0110	1000	kkkk	kkkk

说明： 将FSR2指示的寄存器（偏移量为k）的内容设置为FFh。

指令字数： 1

指令周期数： 1

Q周期活动：

Q1	Q2	Q3	Q4
译码	读取k	处理数据	写入寄存器

示例： SETF [OFST]

执行指令前
OFST = 2Ch
FSR2 = 0A00h
0A2Ch的内容 = 00h

执行指令后
0A2Ch的内容 = FFh

42.2.5 MICROCHIP MPLAB® IDE工具的特殊注意事项

Microchip软件工具的最新版本已设计为完全支持PIC18(L)F25/26K83系列器件的扩展指令集。包括MPLAB C18 C编译器、MPASM汇编语言和MPLAB集成开发环境（IDE）。

选择用于软件开发的目标器件时，MPLAB IDE将自动为该器件设置默认配置位。XINST配置位的默认设置为0，禁用扩展指令集和立即数变址寻址模式。为了正确执行为利用扩展指令集而开发的应用程序，必须在编程期间将XINST置1。

要为扩展指令集开发软件，用户必须在其语言工具中使能对指令和变址寻址模式的支持。根据所使用的环境，可以通过以下几种方式完成：

- 允许用户为项目配置语言工具及其设置的环境中的菜单选项或对话框
- 命令行选项
- 源代码中的指令

这些选项因编译器、汇编器和开发环境而异。建议用户查看其开发系统随附的文档，以获取相应的信息。

43.0 寄存器汇总

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3FFFh	TOSU	—	—	—	栈顶最高字节					28
3FFEh	TOSH	栈顶高字节								28
3FFDh	TOSL	栈顶低字节								28
3FFCh	STKPTR	—	—	—	堆栈指针					29
3FFBh	PCLATU	—	—	—	PC最高字节的保持寄存器					29
3FFAh	PCLATH	PC高字节的保持寄存器								29
3FF9h	PCL	PC低字节								29
3FF8h	TBLPTRU	—	—	程序存储器表指针最高字节						29
3FF7h	TBLPTRH	程序存储器表指针高字节								182
3FF6h	TBLPTRL	程序存储器表指针低字节								182
3FF5h	TABLAT	表锁存器								182
3FF4h	PRODH	产品寄存器高字节								177
3FF3h	PRODL	产品寄存器低字节								177
3FF2h	—	未实现								—
3FF1h	PCON1	—	—	—	—	—	—	$\overline{\text{MEMV}}$	—	81
3FF0h	PCON0	STKOVF	STKUNF	$\overline{\text{WDTWV}}$	$\overline{\text{RWDT}}$	$\overline{\text{MCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	80
3FEFh	INDF0	使用FSR0的内容寻址数据存储器——FSR0的值保持不变								50
3FEEh	POSTINC0	使用FSR0的内容寻址数据存储器——FSR0的值后递增								51
3FEDh	POSTDEC0	使用FSR0的内容寻址数据存储器——FSR0的值后递减								51
3FECh	PREINC0	使用FSR0的内容寻址数据存储器——FSR0的值预递增								51
3FEBh	PLUSW0	使用FSR0的内容寻址数据存储器——FSR0的值预递增——FSR0偏移量的值由W寄存器指定								51
3FEAh	FSR0H	—	—	间接数据存储器地址指针0高字节						51
3FE9h	FSR0L	间接数据存储器地址指针0低字节								51
3FE8h	WREG	工作寄存器								—
3FE7h	INDF1	使用FSR1的内容寻址数据存储器——FSR1的值保持不变								51
3FE6h	POSTINC1	使用FSR1的内容寻址数据存储器——FSR1的值后递增								51
3FE5h	POSTDEC1	使用FSR1的内容寻址数据存储器——FSR1的值后递减								51
3FE4h	PREINC1	使用FSR1的内容寻址数据存储器——FSR1的值预递增								51
3FE3h	PLUSW1	使用FSR1的内容寻址数据存储器——FSR1的值预递增——FSR1偏移量的值由W寄存器指定								51
3FE2h	FSR1H	—	—	间接数据存储器地址指针1高字节						51
3FE1h	FSR1L	间接数据存储器地址指针1低字节								51
3FE0h	BSR	—	—	存储区选择寄存器						34
3FDFh	INDF2	使用FSR2的内容寻址数据存储器——FSR2的值保持不变								51
3FDEh	POSTINC2	使用FSR2的内容寻址数据存储器——FSR2的值后递增								51
3FDDh	POSTDEC2	使用FSR2的内容寻址数据存储器——FSR2的值后递减								51
3FDCh	PREINC2	使用FSR2的内容寻址数据存储器——FSR2的值预递增								51
3FDBh	PLUSW2	使用FSR2的内容寻址数据存储器——FSR2的值预递增——FSR2偏移量的值由W寄存器指定								51
3FDAh	FSR2H	—	—	间接数据存储器地址指针2高字节						51
3FD9h	FSR2L	间接数据存储器地址指针2低字节								51
3FD8h	STATUS	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	N	OV	Z	DC	C	48
3FD7h	IVTBASEU	—	—	—	BASE20	BASE19	BASE18	BASE17	BASE16	157
3FD6h	IVTBASEH	BASE15	BASE14	BASE13	BASE12	BASE11	BASE10	BASE9	BASE8	157
3FD5h	IVTBASEL	BASE7	BASE6	BASE5	BASE4	BASE3	BASE2	BASE1	BASE0	157
3FD4h	IVTLOCK	—	—	—	—	—	—	—	IVTLOCKED	159
3FD3h	INTCON1	STAT		—	—	—	—	—	—	126
3FD2h	INTCON0	GIE	GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	125
3FD1h - 3FD0h	—	未实现								—

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3FCEh	PORTE	—	—	—	—	RE3	—	—	—	252
3FCDh	—	未实现								—
3FCCh	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	252
3FCBh	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	252
3FCAh	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	252
3FC9h - 3FC5h	—	未实现								—
3FC4h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	253
3FC3h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	253
3FC2h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	253
3FC1h - 3FBDh	—	未实现								—
3FBCh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	254
3FBBh	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	254
3FBAh	LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	254
3FB9h	T0CON1	CS<2:0>			ASYNC	CKPS<3:0>				287
3FB8h	T0CON0	EN	—	OUT	MD16	OUTPS				286
3FB7h	TMR0H	TMR0H								288
3FB6h	TMR0L	TMR0L								288
3FB5h	T1CLK	CS								300
3FB4h	T1GATE	GSS								301
3FB3h	T1GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	299
3FB2h	T1CON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	323
3FB1h	TMR1H	TMR1H								302
3FB0h	TMR1L	TMR1L								302
3FAFh	T2RST	—	—	—	RSEL					321
3FAEh	T2CLK	—	—	—	—	CS				300
3FADh	T2HLT	PSYNC	CKPOL	CKSYNC	MODE					324
3FACh	T2CON	ON	CKPS		OUTPS					298
3FABh	T2PR	PR2								322
3FAAh	T2TMR	TMR2								322
3FA9h	T3CLK	CS								300
3FA8h	T3GATE	GSS								301
3FA7h	T3GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	299
3FA6h	T3CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	323
3FA5h	TMR3H	TMR3H								302
3FA4h	TMR3L	TMR3L								302
3FA3h	T4RST	—	—	—	RSEL					321
3FA2h	T4CLK	—	—	—	—	CS				320
3FA1h	T4HLT	PSYNC	CKPOL	CKSYNC	MODE					324
3FA0h	T4CON	ON	CKPS		OUTPS					323
3F9Fh	T4PR	PR4								322
3F9Eh	T4TMR	TMR4								322
3F9Dh	T5CLK	CS								320
3F9Ch	T5GATE	GSS								301
3F9Bh	T5GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	299
3F9Ah	T5CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	323
3F99h	TMR5H	TMR5H								302
3F98h	TMR5L	TMR5L								302
3F97h	T6RST	—	—	—	RSEL					321

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3F96h	T6CLK	—	—	—	—	CS				300
3F95h	T6HLT	PSYNC	CKPOL	CKSYNC	MODE					324
3F94h	T6CON	ON	CKPS			OUTPS				323
3F93h	T6PR	PR6								322
3F92h	T6TMR	TMR6								322
3F91h	ECANCON	MDSEL1	MDSEL0	FIFOWM	EWIN4	EWIN3	EWIN2	EWIN1	EWIN0	607
3F90h	COMSTAT	RXB0OVFL	RXB1OVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN	608
3F90h	COMSTAT	—	RXBnOVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN	608
3F90h	COMSTAT	$\overline{\text{FIFOEMPTY}}$	RXBnOVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN	608
3F8Fh	CANCON	REQOP2	REQOP1	REQOP0	ABAT	WIN2	WIN1	WIN0	—	603
3F8Fh	CANCON	REQOP2	REQOP1	REQOP0	ABAT	—	—	—	—	603
3F8Fh	CANCON	REQOP2	REQOP1	REQOP0	ABAT	FP3	FP2	FP1	FP0	603
3F8Eh	CANSTAT	OPMODE2	OPMODE1	OPMODE0	—	ICODE2	ICODE1	ICODE0	—	604
3F8Eh	CANSTAT	OPMODE2	OPMODE1	OPMODE0	EICODE4	EICODE3	EICODE2	EICODE1	EICODE0	604
3F8Dh	RXB0D7	RXB0Dm7	RXB0Dm6	RXB0Dm5	RXB0Dm4	RXB0Dm3	RXB0Dm2	RXB0Dm1	RXB0Dm0	620
3F8Ch	RXB0D6	RXB0Dm7	RXB0Dm6	RXB0Dm5	RXB0Dm4	RXB0Dm3	RXB0Dm2	RXB0Dm1	RXB0Dm0	620
3F8Bh	RXB0D5	RXB0Dm7	RXB0Dm6	RXB0Dm5	RXB0Dm4	RXB0Dm3	RXB0Dm2	RXB0Dm1	RXB0Dm0	620
3F8Ah	RXB0D4	RXB0Dm7	RXB0Dm6	RXB0Dm5	RXB0Dm4	RXB0Dm3	RXB0Dm2	RXB0Dm1	RXB0Dm0	620
3F89h	RXB0D3	RXB0Dm7	RXB0Dm6	RXB0Dm5	RXB0Dm4	RXB0Dm3	RXB0Dm2	RXB0Dm1	RXB0Dm0	620
3F88h	RXB0D2	RXB0Dm7	RXB0Dm6	RXB0Dm5	RXB0Dm4	RXB0Dm3	RXB0Dm2	RXB0Dm1	RXB0Dm0	620
3F87h	RXB0D1	RXB0Dm7	RXB0Dm6	RXB0Dm5	RXB0Dm4	RXB0Dm3	RXB0Dm2	RXB0Dm1	RXB0Dm0	620
3F86h	RXB0D0	RXB0Dm7	RXB0Dm6	RXB0Dm5	RXB0Dm4	RXB0Dm3	RXB0Dm2	RXB0Dm1	RXB0Dm0	620
3F85h	RXB0DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	620
3F84h	RXB0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	620
3F83h	RXB0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	620
3F82h	RXB0SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	620
3F81h	RXB0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	620
3F80h	RXB0CON	RXFUL	RXM1	RXM0	—	RXRTRRO	RXB0DBEN	JTOFF	FILHIT0	620
3F80h	RXB0CON	RXFUL	RXM1	RTRRO	FILHITF4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	620
3F7Fh	CCP1CAP	—	—	—	—	CTS<3:0>				338
3F7Eh	CCP1CON	EN	—	OUT	FMT	MODE				335
3F7Dh	CCPR1H	RH								339
3F7Ch	CCPR1L	RL								338
3F7Bh	CCP2CAP	—	—	—	—	CTS<3:0>				338
3F7Ah	CCP2CON	EN	—	OUT	FMT	MODE				335
3F79h	CCPR2H	RH								339
3F78h	CCPR2L	RL								338
3F77h	CCP3CAP	—	—	—	—	CTS<3:0>				338
3F76h	CCP3CON	EN	—	OUT	FMT	MODE				335
3F75h	CCPR3H	RH								339
3F74h	CCPR3L	RL								338
3F73h	CCP4CAP	—	—	—	—	CTS<3:0>				338
3F72h	CCP4CON	EN	—	OUT	FMT	MODE				335
3F71h	CCPR4H	RH								339
3F70h	CCPR4L	RL								338
3F6Fh	—	未实现								—
3F6Eh	PWM5CON	EN	—	OUT	POL	—	—	—	—	344
3F6Dh	PWM5DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	346
3F6Ch	PWM5DCL	DC1	DC0	—	—	—	—	—	—	346
3F6Bh	—	未实现								—

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件

注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3F6Ah	PWM6CON	EN	—	OUT	POL	—	—	—	—	344
3F69h	PWM6DCH	DC9		DC7	DC6	DC5	DC4	DC3	DC2	346
3F68h	PWM6DCL	DC1	DC0	—	—	—	—	—	—	346
3F67h	—	未实现								—
3F66h	PWM7CON	EN	—	OUT	POL	—	—	—	—	344
3F65h	PWM7DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	346
3F64h	PWM7DCL	DC1	DC0	—	—	—	—	—	—	346
3F63h	—	未实现								—
3F62h	PWM8CON	EN	—	OUT	POL	—	—	—	—	344
3F61h	PWM8DCH	DC9	DC8	DC7	DC6	DC5	DC4	DC3	DC2	346
3F60h	PWM8DCL	DC1	DC0	—	—	—	—	—	—	346
3F5Fh	CCPTMRS1	P8TSEL		P7TSEL		P6TSEL		P5TSEL		345
3F5Eh	CCPTMRS0	C4TSEL		C3TSEL		C2TSEL		C1TSEL		345
3F5Dh - 3F5Bh	—	未实现								—
3F5Ah	CWG1STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	414
3F59h	CWG1AS1	—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	416
3F58h	CWG1AS0	SHUTDOWN	REN	LSBD		LSAC		—	—	415
3F57h	CWG1CON1	—	—	IN	—	POLD	POLC	POLB	POLA	411
3F56h	CWG1CON0	EN	LD	—	—	—	MODE			410
3F55h	CWG1DBF	—	—	DBF						417
3F54h	CWG1DBR	—	—	DBR						417
3F53h	CWG1ISM	—	—	—	—	IS				413
3F52h	CWG1CLK	—	—	—	—	—	—	—	CS	412
3F51h	CWG2STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	414
3F50h	CWG2AS1	—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	416
3F4Fh	CWG2AS0	SHUTDOWN	REN	LSBD		LSAC		—	—	415
3F4Eh	CWG2CON1	—	—	IN	—	POLD	POLC	POLB	POLA	411
3F4Dh	CWG2CON0	EN	LD	—	—	—	MODE			410
3F4Ch	CWG2DBF	—	—	DBF						417
3F4Bh	CWG2DBR	—	—	DBR						417
3F4Ah	CWG2ISM	—	—	—	—	IS				413
3F49h	CWG2CLK	—	—	—	—	—	—	—	CS	412
3F48h	CWG3STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	414
3F47h	CWG3AS1	—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	416
3F46h	CWG3AS0	SHUTDOWN	REN	LSBD		LSAC		—	—	415
3F45h	CWG3CON1	—	—	IN	—	POLD	POLC	POLB	POLA	411
3F44h	CWG3CON0	EN	LD	—	—	—	MODE			410
3F43h	CWG3DBF	—	—	DBF						417
3F42h	CWG3DBR	—	—	DBR						417
3F41h	CWG3ISM	—	—	—	—	IS				413
3F40h	CWG3CLK	—	—	—	—	—	—	—	CS	412
3F3Fh	NCO1CLK	PWS			—	CKS				440
3F3Eh	NCO1CON	EN	—	OUT	POL	—	—	—	PFM	439
3F3Dh	NCO1INCU	INC								443
3F3Ch	NCO1INCH	INC								442
3F3Bh	NCO1INCL	INC								442
3F3Ah	NCO1ACCU	ACC								442
3F39h	NCO1ACCH	ACC								441
3F38h	NCO1ACCL	ACC								441

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3F37h - 3F24h	—	未实现								—
3F23h	SMT1WIN	—	—	—	WSEL					384
3F22h	SMT1SIG	—	—	—	SSEL					385
3F21h	SMT1CLK	—	—	—	—	—	CSEL			383
3F20h	SMT1STAT	CPRUP	CPWUP	RST	—	—	TS	WS	AS	382
3F1Fh	SMT1CON1	GO	REPEAT	—	—	MODE				381
3F1Eh	SMT1CON0	EN	—	STP	WPOL	SPOL	CPOL	PS		380
3F1Dh	SMT1PRU	PR								387
3F1Ch	SMT1PRH	PR								389
3F1Bh	SMT1PRL	PR								389
3F1Ah	SMT1CPWU	CPW								388
3F19h	SMT1CPWH	CPW								388
3F18h	SMT1CPWL	CPW								388
3F17h	SMT1CPRU	CPR								387
3F16h	SMT1CPRH	CPR								387
3F15h	SMT1CPRL	CPR								387
3F14h	SMT1TMRU	TMR								386
3F13h	SMT1TMRH	TMR								386
3F12h	SMT1TMRL	TMR								386
3F11h	SMT2WIN	—	—	—	WSEL					384
3F10h	SMT2SIG	—	—	—	SSEL					385
3F0Fh	SMT2CLK	—	—	—	—	—	CSEL			383
3F0Eh	SMT2STAT	CPRUP	CPWUP	RST	—	—	TS	WS	AS	382
3F0Dh	SMT2CON1	GO	REPEAT	—	—	MODE				381
3F0Ch	SMT2CON0	EN	—	STP	WPOL	SPOL	CPOL	PS		380
3F0Bh	SMT2PRU	PR								387
3F0Ah	SMT2PRH	PR								389
3F09h	SMT2PRL	PR								389
3F08h	SMT2CPWU	CPW								388
3F07h	SMT2CPWH	CPW								388
3F06h	SMT2CPWL	CPW								388
3F05h	SMT2CPRU	CPR								387
3F04h	SMT2CPRH	CPR								387
3F03h	SMT2CPRL	CPR								387
3F02h	SMT2TMRU	TMR								386
3F01h	SMT2TMRH	TMR								386
3F00h	SMT2TMRL	TMR								386
3EFFh	ADCLK	—	—	CS						676
3EFEh	ADACT	—	—	—	ACT					676
3EFDh	ADREF	NREF				PREF				676
3EFCh	ADSTAT	OV	UTHR	LTHR	MATH	—	STAT			675
3EFBh	ADCON3	—	CALC			SOI	TMD			674
3EFAh	ADCON2	PSIS	CRS			ACLR	MODE			673
3EF9h	ADCON1	PPOL	IPEN	GPOL	—	—	—	—	DSEN	672
3EF8h	ADCON0	ON	CONT	—	CS	FM		—	GO	671
3EF7h	ADPREH	—	—	—	PRE					678
3EF6h	ADPREL	PRE								678
3EF5h	ADCAP	—	—	—	ADCAP					680
3EF4h	ADACQH	—	—	—	ACQ					679

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3EF3h	ADACQL	ACQ								679
3EF2h	—	未实现								—
3EF1h	ADPCH	—	—	ADPCH						677
3EF0h	ADRESH	RES								682
3EEFh	ADRESL	RES								682
3EEEh	ADPREVH	PREV								684
3EEDh	ADPREVL	PREV								684
3EECh	ADRPT	RPT								680
3EEBh	ADCNT	CNT								681
3EEAh	ADACCU	ACC								685
3EE9h	ADACCH	ACC								685
3EE8h	ADACCL	ACC								685
3EE7h	ADFLTRH	FLTR								681
3EE6h	ADFLTRL	FLTR								681
3EE5h	ADSTPTH	STPT								686
3EE4h	ADSTPTL	STPT								686
3EE3h	ADERRH	ERR								687
3EE2h	ADERRL	ERR								687
3EE1h	ADUTHH	UTH								688
3EE0h	ADUTHL	UTH								688
3EDFh	ADLTHH	LTH								687
3EDEh	ADLTHL	LTH								688
3EDDh - 3ED8h	—	未实现								—
3ED7h	ADCP	ON	—	—	—	—	—	—	CPRDY	690
3ED6h - 3ECBh	—	未实现								—
3ECAh	HLVDCON1	—	—	—	—	SEL				712
3EC9h	HLVDCON0	EN	—	OUT	RDY	—	—	INTH	INTL	711
3EC8h - 3EC4h	—	未实现								—
3EC3h	ZCDCON	SEN	—	OUT	POL	—	—	INTP	INTN	448
3EC2h	—	未实现								—
3EC1h	FVRCON	EN	RDY	TSEN	TSRNG	CDAFVR		ADFVR		650
3EC0h	CMOUT	—	—	—	—	—	—	C2OUT	C1OUT	704
3EBFh	CM1PCH	—	—	—	—	—	PCH			704
3EBEh	CM1NCH	—	—	—	—	—	NCH			703
3EBDh	CM1CON1	—	—	—	—	—	—	INTP	INTN	703
3EBCh	CM1CON0	EN	OUT	—	POL	—	—	HYS	SYNC	702
3EBBh	CM2PCH	—	—	—	—	—	PCH			704
3EBAh	CM2NCH	—	—	—	—	—	NCH			703
3EB9h	CM2CON1	—	—	—	—	—	—	INTP	INTN	703
3EB8h	CM2CON0	EN	OUT	—	POL	—	—	HYS	SYNC	702
3EB7h - 3E9Fh	—	未实现								—
3E9Eh	DAC1CON0	EN	—	OE1	OE2	PSS		—	NSS	694
3E9Dh	—	未实现								—
3E9Ch	DAC1CON1	—	—	—	DATA					695
3E9Bh - 3DFBh	—	未实现								
3DFAh	U1ERRIE	TXMTIE	PERIE	ABDOVE	CERIE	FERIE	RXBKIE	RXFOIE	TXCIE	487

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3DF9h	U1ERRIR	TXMTIF	PERIF	ABDOVF	CERIF	FERIF	RXBKIF	RXFOIF	TXCIF	486
3DF8h	U1UIR	WUIF	ABDIF	—	—	—	ABDIE	—	—	488
3DF7h	U1FIFO	TXWRE	STPMD	TXBE	TXBF	RXIDL	XON	RXBE	RXBF	489
3DF6h	U1BRGH	BRGH								490
3DF5h	U1BRGL	BRGL								490
3DF4h	U1CON2	RUNOVF	RXPOL	STP		C0EN	TXPOL	FLO		485
3DF3h	U1CON1	ON	—	—	WUE	RXBIMD	—	BRKOVR	SENDB	484
3DF2h	U1CON0	BRGS	ABDEN	TXEN	RXEN	MODE				483
3DF1h	U1P3H	—	—	—	—	—	—	—	P3H	494
3DF0h	U1P3L	P3L								494
3DEFh	U1P2H	—	—	—	—	—	—	—	P2H	493
3DEEh	U1P2L	P2L								493
3DEDh	U1P1H	—	—	—	—	—	—	—	P1H	492
3DECh	U1P1L	P1L								492
3DEBh	U1TXCHK	TXCHK								495
3DEAh	U1TXB	TXB								491
3DE9h	U1RXCHK	RXCHK								495
3DE8h	U1RXB	RXB								491
3DE7h - 3DE3h	—	未实现								—
3DE2h	U2ERRIE	TXMTIE	PERIE	ABDOVE	CERIE	FERIE	RXBKIE	RXFOIE	TXCIE	487
3DE1h	U2ERRIR	TXMTIF	PERIF	ABDOVF	CERIF	FERIF	RXBKIF	RXFOIF	TXCIF	486
3DE0h	U2UIR	WUIF	ABDIF	—	—	—	ABDIE	—	—	488
3DDFh	U2FIFO	TXWRE	STPMD	TXBE	TXBF	RXIDL	XON	RXBE	RXBF	489
3DDEh	U2BRGH	BRGH								490
3DDDh	U2BRGL	BRGL								490
3DDCh	U2CON2	RUNOVF	RXPOL	STP		—	TXPOL	FLO		485
3DDBh	U2CON1	ON	—	—	WUE	RXBIMD	—	BRKOVR	SENDB	484
3DDAh	U2CON0	BRGS	ABDEN	TXEN	RXEN	MODE				483
3DD9h	U2P3H	—	—	—	—	—	—	—	P3H	494
3DD8h	U2P3L	P3L								494
3DD7h	U2P2H	—	—	—	—	—	—	—	P2H	493
3DD6h	U2P2L	P2L								493
3DD5h	U2P1H	—	—	—	—	—	—	—	P1H	492
3DD4h	U2P1L	P1L								492
3DD3h	U2TXCHK	TXCHK								495
3DD2h	U2TXB	TXB								491
3DD1h	U2RXCHK	RXCHK								495
3DD0h	U2RXB	RXB								491
3DCFh - 3D7Dh	—	未实现								—
3D7Ch	I2C1BTO	BTO								567
3D7Bh	I2C1CLK	CLK								566
3D7Ah	I2C1PIE	CNTIE	ACKTIE	—	WRIE	ADRIE	PCIE	RSCIE	SCIE	573
3D79h	I2C1PIR	CNTIF	ACKTIF	—	WRIF	ADRIF	PCIF	RSCIF	SCIF	572
3D78h	I2C1STAT1	TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF	569
3D77h	I2C1STAT0	BFRE	SMA	MMA	R	D	—	—	—	568
3D76h	I2C1ERR	—	BTOIF	BCLIF	NACKIF	—	BTOIE	BCLIE	NACKIE	570
3D75h	I2C1CON2	ACNT	GCEN	FME	ABD	SDAHT		BFRET		565
3D74h	I2C1CON1	ACKCNT	ACKDT	ACKSTAT	ACKT	—	RXO	TXU	CSD	564

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3D73h	I2C1CON0	EN	RSEN	S	CSTR	MDR	MODE			562
3D72h	I2C1ADR3	ADR							—	577
3D71h	I2C1ADR2	ADR								576
3D70h	I2C1ADR1	ADR							—	575
3D6Fh	I2C1ADR0	ADR								574
3D6Eh	I2C1ADB1	ADB								579
3D6Dh	I2C1ADB0	ADB								578
3D6Ch	I2C1CNT	CNT								571
3D6Bh	I2C1TXB	TXB								—
3D6Ah	I2C1RXB	RXB								—
3D69h - 3D67h	—	未实现								—
3D66h	I2C2BTO	BTO								567
3D65h	I2C2CLK	CLK								566
3D64h	I2C2PIE	CNTIE	ACKTIE	—	WRIE	ADRIE	PCIE	RSCIE	SCIE	573
3D63h	I2C2PIR	CNTIF	ACKTIF	—	WRIF	ADRIF	PCIF	RSCIF	SCIF	572
3D62h	I2C2STAT1	TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF	569
3D61h	I2C2STAT0	BFRE	SMA	MMA	R	D	—	—	—	568
3D60h	I2C2ERR	—	BTOIF	BCLIF	NACKIF	—	BTOIE	BCLIE	NACKIE	570
3D5Fh	I2C2CON2	ACNT	GCEN	FME	ABD	SDAHT		BFRET		565
3D5Eh	I2C2CON1	ACKCNT	ACKDT	ACKSTAT	ACKT	—	RXO	TXU	CSD	564
3D5Dh	I2C2CON0	EN	RSEN	S	CSTR	MDR	MODE			562
3D5Ch	I2C2ADR3	ADR							—	577
3D5Bh	I2C2ADR2	ADR								576
3D5Ah	I2C2ADR1	ADR							—	575
3D59h	I2C2ADR0	ADR								574
3D58h	I2C2ADB1	ADB								579
3D57h	I2C2ADB0	ADB								578
3D56h	I2C2CNT	CNT								571
3D55h	I2C2TXB	TXB								—
3D54h	I2C2RXB	RXB								—
3D53h - 3D1Dh	—	未实现								—
3D1Ch	SPI1CLK	CLKSEL								527
3D1Bh	SPI1INTE	SRMTIE	TCZIE	SOSIE	EOSIE	—	RXOIE	TXUIE	—	521
3D1Ah	SPI1INTF	SRMTIF	TCZIF	SOSIF	EOSIF	—	RXOIF	TXUIF	—	520
3D19h	SPI1BAUD	BAUD								523
3D18h	SPI1TWIDTH	—	—	—	—	—	TWIDTH			522
3D17h	SPI1STATUS	TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF	526
3D16h	SPI1CON2	BUSY	SSFLT	—	—	—	SSET	TXR	RXR	525
3D15h	SPI1CON1	SMP	CKE	CKP	FST	—	SSP	SDIP	SDOP	524
3D14h	SPI1CON0	EN	—	—	—	—	LSBF	MST	BMODE	523
3D13h	SPI1TCNTH	—	—	—	—	—	TCNTH			522
3D12h	SPI1TCNTL	TCNTL								521
3D11h	SPI1TXB	TXB								527
3D10h	SPI1RXB	RXB								526
3D0Fh - 3CFFh	—	未实现								—
3CFEh	MD1CARH	—	—	—	CH					457
3CFDh	MD1CARL	—	—	—	CL					457

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件

注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3CFCh	MD1SRC	—	—	—	MS					458
3CFBh	MD1CON1	—	—	CHPOL	CHSYNC	—	—	CLPOL	CLSYNC	456
3CFAh	MD1CON0	EN	—	OUT	OPOL	—	—	—	BIT	455
3CF9h - 3CE7h	—	未实现								—
3CE6h	CLKRCLK	—	—	—	—	CLK				104
3CE5h	CLKRCON	EN	—	—	DC		DIV			103
3CE4h - 3C7Fh	—	未实现								—
3C7Eh	CLCDATA0	—	—	—	—	CLC4OUT	CLC3OUT	CLC2OUT	CLC1OUT	433
3C7Dh	CLC1GLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	432
3C7Ch	CLC1GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	431
3C7Bh	CLC1GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	430
3C7Ah	CLC1GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	429
3C79h	CLC1SEL3	D4S								428
3C78h	CLC1SEL2	D3S								428
3C77h	CLC1SEL1	D2S								428
3C76h	CLC1SEL0	D1S								428
3C75h	CLC1POL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	427
3C74h	CLC1CON	EN	OE	OUT	INTP	INTN	MODE			426
3C73h	CLC2GLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	432
3C72h	CLC2GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	431
3C71h	CLC2GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	430
3C70h	CLC2GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	429
3C6Fh	CLC2SEL3	D4S								428
3C6Eh	CLC2SEL2	D3S								428
3C6Dh	CLC2SEL1	D2S								428
3C6Ch	CLC2SEL0	D1S								428
3C6Bh	CLC2POL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	427
3C6Ah	CLC2CON	EN	OE	OUT	INTP	INTN	MODE			426
3C69h	CLC3GLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	432
3C68h	CLC3GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	431
3C67h	CLC3GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	430
3C66h	CLC3GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	429
3C65h	CLC3SEL3	D4S								428
3C64h	CLC3SEL2	D3S								428
3C63h	CLC3SEL1	D2S								428
3C62h	CLC3SEL0	D1S								429
3C61h	CLC3POL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	427
3C60h	CLC3CON	EN	OE	OUT	INTP	INTN	MODE			426
3C5Fh	CLC4GLS3	G4D4T	G4D4N	G4D3T	G4D3N	G4D2T	G4D2N	G4D1T	G4D1N	432
3C5Eh	CLC4GLS2	G3D4T	G3D4N	G3D3T	G3D3N	G3D2T	G3D2N	G3D1T	G3D1N	431
3C5Dh	CLC4GLS1	G2D4T	G2D4N	G2D3T	G2D3N	G2D2T	G2D2N	G2D1T	G2D1N	430
3C5Ch	CLC4GLS0	G1D4T	G1D4N	G1D3T	G1D3N	G1D2T	G1D2N	G1D1T	G1D1N	429
3C5Bh	CLC4SEL3	D4S								428
3C5Ah	CLC4SEL2	D3S								428
3C59h	CLC4SEL1	D2S								428
3C58h	CLC4SEL0	D1S								429
3C57h	CLC4POL	POL	—	—	—	G4POL	G3POL	G2POL	G1POL	427
3C56h	CLC4CON	EN	OE	OUT	INTP	INTN	MODE			426

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3C55h - 3C00h	—	未实现								—
3BFFh	DMA1SIRQ	SIRQ								246
3BFEh	DMA1AIRQ	AIRQ								246
3BFDh	DMA1CON1	DMODE		DSTP	SMR		SMODE		SSTP	239
3BFCh	DMA1CON0	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	238
3BFBh	DMA1SSAU	—	—	SSA						241
3BFAh	DMA1SSAH	SSA								240
3BF9h	DMA1SSAL	SSA								240
3BF8h	DMA1SSZH	—	—	—	—	SSZ				242
3BF7h	DMA1SSZL	SSZ								242
3BF6h	DMA1SPTRU	—	—	SPTR						242
3BF5h	DMA1SPTRH	SPTR								241
3BF4h	DMA1SPTRL	SPTR								241
3BF3h	DMA1SCNTH	—	—	—	—	SCNT				243
3BF2h	DMA1SCNTL	SCNT								243
3BF1h	DMA1DSAH	DSA								244
3BF0h	DMA1DSAL	SSA								243
3BEFh	DMA1DSZH	—	—	—	—	DSZ				245
3BEEh	DMA1DSZL	DSZ								245
3BEDh	DMA1DPTRH	DPTR								244
3BECh	DMA1DPTRL	DPTR								244
3BEBh	DMA1DCNTH	—	—	—	—	DCNT				243
3BEAh	DMA1DCNTL	DCNT								245
3BE9h	DMA1BUF	BUF								240
3BE8h - 3BE0h	—	未实现								—
3BDFh	DMA2SIRQ	—	SIRQ							246
3BDEh	DMA2AIRQ	—	AIRQ							246
3BDDh	DMA2CON1	DMODE		DSTP	SMR		SMODE		SSTP	239
3BDCh	DMA2CON0	EN	SIRQEN	DGO	—	—	AIRQEN	—	XIP	238
3BDBh	DMA2SSAU	—	—	SSA						241
3BDAh	DMA2SSAH	SSA								240
3BD9h	DMA2SSAL	SSA								240
3BD8h	DMA2SSZH	—	—	—	—	SSZ				242
3BD7h	DMA2SSZL	SSZ								242
3BD6h	DMA2SPTRU	—	—	SPTR						242
3BD5h	DMA2SPTRH	SPTR								241
3BD4h	DMA2SPTRL	SPTR								241
3BD3h	DMA2SCNTH	—	—	—	—	SCNT				243
3BD2h	DMA2SCNTL	SCNT								243
3BD1h	DMA2DSAH	DSA								244
3BD0h	DMA2DSAL	SSA								243
3BCFh	DMA2DSZH	—	—	—	—	DSZ				245
3BCEh	DMA2DSZL	DSZ								245
3BCDh	DMA2DPTRH	DPTR								244
3BCCh	DMA2DPTRL	DPTR								244
3BCBh	DMA2DCNTH	—	—	—	—	DCNT				243
3BCAh	DMA2DCNTL	DCNT								245
3BC9h	DMA2BUF	BUF								240

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3BC8h - 3AEEh	—	未实现								—
3AEDh	CANRXPPS	—	—	—	CANRXPPS					264
3AECh	—	未实现								—
3AEBh	U2CTSPPS	—	—	—	U2CTSPPS					264
3AEAh	U2RXPPS	—	—	—	U2RXPPS					264
3AE9h	—	未实现								—
3AE8h	U1CTSPPS	—	—	—	U1CTSPPS					264
3AE7h	U1RXPPS	—	—	—	U1RXPPS					264
3AE6h	I2C2SDAPPS	—	—	—	I2C2SDAPPS					264
3AE5h	I2C2SCLPPS	—	—	—	I2C2SCLPPS					264
3AE4h	I2C1SDAPPS	—	—	—	I2C1SDAPPS					264
3AE3h	I2C1SCLPPS	—	—	—	I2C1SCLPPS					264
3AE2h	SPI1SSPPS	—	—	—	SPI1SSPPS					264
3AE1h	SPI1SDIPPS	—	—	—	SPI1SDIPPS					264
3AE0h	SPI1SCKPPS	—	—	—	SPI1SCKPPS					264
3ADFh	ADACTPPS	—	—	—	ADACTPPS					264
3ADEh	CLCIN3PPS	—	—	—	CLCIN3PPS					264
3ADDh	CLCIN2PPS	—	—	—	CLCIN2PPS					264
3ADCh	CLCIN1PPS	—	—	—	CLCIN1PPS					264
3ADBh	CLCIN0PPS	—	—	—	CLCIN0PPS					264
3ADAh	MD1SRCPPS	—	—	—	MD1SRCPPS					264
3AD9h	MD1CARHPPS	—	—	—	MD1CARHPPS					264
3AD8h	MD1CARLPPS	—	—	—	MD1CARLPPS					264
3AD7h	CWG3INPPS	—	—	—	CWG3INPPS					264
3AD6h	CWG2INPPS	—	—	—	CWG2INPPS					264
3AD5h	CWG1INPPS	—	—	—	CWG1INPPS					264
3AD4h	SMT2SIGPPS	—	—	—	SMT2SIGPPS					264
3AD3h	SMT2WINPPS	—	—	—	SMT2WINPPS					264
3AD2h	SMT1SIGPPS	—	—	—	SMT1SIGPPS					264
3AD1h	SMT1WINPPS	—	—	—	SMT1WINPPS					264
3AD0h	CCP4PPS	—	—	—	CCP4PPS					264
3ACFh	CCP3PPS	—	—	—	CCP3PPS					264
3ACEh	CCP2PPS	—	—	—	CCP2PPS					264
3ACDh	CCP1PPS	—	—	—	CCP1PPS					264
3ACCh	T6INPPS	—	—	—	T6INPPS					264
3ACBh	T4INPPS	—	—	—	T4INPPS					264
3ACAh	T2INPPS	—	—	—	T2INPPS					264
3AC9h	T5GPPS	—	—	—	T5GPPS					264
3AC8h	T5CLKIPPS	—	—	—	T5CLKIPPS					264
3AC7h	T3GPPS	—	—	—	T3GPPS					264
3AC6h	T3CLKIPPS	—	—	—	T3CLKIPPS					264
3AC5h	T1GPPS	—	—	—	T1GPPS					264
3AC4h	T1CKIPPS	—	—	—	T1CKIPPS					264
3AC3h	T0CKIPPS	—	—	—	T0CKIPPS					264
3AC2h	INT2PPS	—	—	—	INT2PPS					264
3AC1h	INT1PPS	—	—	—	INT1PPS					264
3AC0h	INT0PPS	—	—	—	INT0PPS					264
3ABFh	PPSLOCK	—	—	—	—	—	—	—	PPSLOCKED	268

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3ABEh - 3A88h	—	未实现								—
3A87h	IOCEF	—	—	—	—	IOCEF3	—	—	—	272
3A86h	IOCEN	—	—	—	—	IOCEN3	—	—	—	272
3A85h	IOCEP	—	—	—	—	IOCEP3	—	—	—	272
3A84h	INLVLE	—	—	—	—	INLVLE3	—	—	—	259
3A83h	—	未实现								—
3A82h	—	未实现								—
3A81h	WPUE	—	—	—	—	WPUE3	—	—	—	256
3A80h- 3A6Ch	—	未实现								—
3A6Bh	RC4I2C	—	SLEW	PU		—	—	TH		252
3A6Ah	RC3I2C	—	SLEW	PU		—	—	TH		252
3A69h	—	未实现								—
3A68h	—	未实现								—
3A67h	IOCCF	IOCCF7	IOCCF6	IOCCF5	IOCCF4	IOCCF3	IOCCF2	IOCCF1	IOCCF0	272
3A66h	IOCCN	IOCCN7	IOCCN6	IOCCN5	IOCCN4	IOCCN3	IOCCN2	IOCCN1	IOCCN0	272
3A65h	IOCCP	IOCCP7	IOCCP6	IOCCP5	IOCCP4	IOCCP3	IOCCP2	IOCCP1	IOCCP0	272
3A64h	INLVLC	INLVLC7	INLVLC6	INLVLC5	INLVLC4	INLVLC3	INLVLC2	INLVLC1	INLVLC0	259
3A63h	SLRCONC	SLRC7	SLRC6	SLRC5	SLRC4	SLRC3	SLRC2	SLRC1	SLRC0	258
3A62h	ODCONC	ODCC7	ODCC6	ODCC5	ODCC4	ODCC3	ODCC2	ODCC1	ODCC0	257
3A61h	WPUC	WPUC7	WPUC6	WPUC5	WPUC4	WPUC3	WPUC2	WPUC1	WPUC0	256
3A60h	ANSELC	ANSELC7	ANSELC6	ANSELC5	ANSELC4	ANSELC3	ANSELC2	ANSELC1	ANSELC0	255
3A5Fh - 3A5Ch	—	未实现								—
3A5Bh	RB2I2C	—	SLEW	PU		—	—	TH		252
3A5Ah	RB1I2C	—	SLEW	PU		—	—	TH		252
3A59h	—	未实现								—
3A58h	—	未实现								—
3A57h	IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0	272
3A56h	IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0	272
3A55h	IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0	272
3A54h	INLVLB	INLVLB7	INLVLB6	INLVLB5	INLVLB4	INLVLB3	INLVLB2	INLVLB1	INLVLB0	259
3A53h	SLRCONB	SLRB7	SLRB6	SLRB5	SLRB4	SLRB3	SLRB2	SLRB1	SLRB0	258
3A52h	ODCONB	ODCB7	ODCB6	ODCB5	ODCB4	ODCB3	ODCB2	ODCB1	ODCB0	257
3A51h	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	256
3A50h	ANSELB	ANSELB7	ANSELB6	ANSELB5	ANSELB4	ANSELB3	ANSELB2	ANSELB1	ANSELB0	255
3A4Fh - 3A48h	—	未实现								—
3A47h	IOCAF	IOCAF7	IOCAF6	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0	272
3A46h	IOCAN	IOCAN7	IOCAN6	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0	272
3A45h	IOCAP	IOCAP7	IOCAP6	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0	272
3A44h	INLVLA	INLVLA7	INLVLA6	INLVLA5	INLVLA4	INLVLA3	INLVLA2	INLVLA1	INLVLA0	259
3A43h	SLRCONA	SLRA7	SLRA6	SLRA5	SLRA4	SLRA3	SLRA2	SLRA1	SLRA0	258
3A42h	ODCONA	ODCA7	ODCA6	ODCA5	ODCA4	ODCA3	ODCA2	ODCA1	ODCA0	257
3A41h	WPUA	WPUA7	WPUA6	WPUA5	WPUA4	WPUA3	WPUA2	WPUA1	WPUA0	256
3A40h	ANSELA	ANSELA7	ANSELA6	ANSELA5	ANSELA4	ANSELA3	ANSELA2	ANSELA1	ANSELA0	255
3A3Fh - 3A18h	—	未实现								—
3A17h	RC7PPS	—	—	—	RC7PPS4	RC7PPS3	RC7PPS2	RC7PPS1	RC7PPS0	266
3A16h	RC6PPS	—	—	—	RC6PPS4	RC6PPS3	RC6PPS2	RC6PPS1	RC6PPS0	266

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3A15h	RC5PPS	—	—	—	RC5PPS4	RC5PPS3	RC5PPS2	RC5PPS1	RC5PPS0	266
3A14h	RC4PPS	—	—	—	RC4PPS4	RC4PPS3	RC4PPS2	RC4PPS1	RC4PPS0	266
3A13h	RC3PPS	—	—	—	RC3PPS4	RC3PPS3	RC3PPS2	RC3PPS1	RC3PPS0	266
3A12h	RC2PPS	—	—	—	RC2PPS4	RC2PPS3	RC2PPS2	RC2PPS1	RC2PPS0	266
3A11h	RC1PPS	—	—	—	RC1PPS4	RC1PPS3	RC1PPS2	RC1PPS1	RC1PPS0	266
3A10h	RC0PPS	—	—	—	RC0PPS4	RC0PPS3	RC0PPS2	RC0PPS1	RC0PPS0	266
3A0Fh	RB7PPS	—	—	—	RB7PPS4	RB7PPS3	RB7PPS2	RB7PPS1	RB7PPS0	266
3A0Eh	RB6PPS	—	—	—	RB6PPS4	RB6PPS3	RB6PPS2	RB6PPS1	RB6PPS0	266
3A0Dh	RB5PPS	—	—	—	RB5PPS4	RB5PPS3	RB5PPS2	RB5PPS1	RB5PPS0	266
3A0Ch	RB4PPS	—	—	—	RB4PPS4	RB4PPS3	RB4PPS2	RB4PPS1	RB4PPS0	266
3A0Bh	RB3PPS	—	—	—	RB3PPS4	RB3PPS3	RB3PPS2	RB3PPS1	RB3PPS0	266
3A0Ah	RB2PPS	—	—	—	RB2PPS4	RB2PPS3	RB2PPS2	RB2PPS1	RB2PPS0	266
3A09h	RB1PPS	—	—	—	RB1PPS4	RB1PPS3	RB1PPS2	RB1PPS1	RB1PPS0	266
3A08h	RB0PPS	—	—	—	RB0PPS4	RB0PPS3	RB0PPS2	RB0PPS1	RB0PPS0	266
3A07h	RA7PPS	—	—	—	RA7PPS4	RA7PPS3	RA7PPS2	RA7PPS1	RA7PPS0	266
3A06h	RA6PPS	—	—	—	RA6PPS4	RA6PPS3	RA6PPS2	RA6PPS1	RA6PPS0	266
3A05h	RA5PPS	—	—	—	RA5PPS4	RA5PPS3	RA5PPS2	RA5PPS1	RA5PPS0	266
3A04h	RA4PPS	—	—	—	RA4PPS4	RA4PPS3	RA4PPS2	RA4PPS1	RA4PPS0	266
3A03h	RA3PPS	—	—	—	RA3PPS4	RA3PPS3	RA3PPS2	RA3PPS1	RA3PPS0	266
3A02h	RA2PPS	—	—	—	RA2PPS4	RA2PPS3	RA2PPS2	RA2PPS1	RA2PPS0	266
3A01h	RA1PPS	—	—	—	RA1PPS4	RA1PPS3	RA1PPS2	RA1PPS1	RA1PPS0	266
3A00h	RA0PPS	—	—	—	RA0PPS4	RA0PPS3	RA0PPS2	RA0PPS1	RA0PPS0	266
39FFh - 39F8h	—	未实现								—
39F7h	SCANPR	—	—	—	—	—	PR			21
39F6h - 39F5h	—	未实现								—
39F4h	DMA2PR	—	—	—	—	—	PR			21
39F3h	DMA1PR	—	—	—	—	—	PR			20
39F2h	MAINPR	—	—	—	—	—	PR			20
39F1h	ISRPR	—	—	—	—	—	PR			20
39F0h	—	未实现								—
39EFh	PRLOCK	—	—	—	—	—	—	—	PRLOCKED	21
39EEh - 39E7h	—	未实现								—
39E6h	NVMCON2	NVMCON2								201
39E5h	NVMCON1	REG		—	FREE	WRERR	WREN	WR	RD	200
39E4h	—	未实现								—
39E3h	NVMDAT	DAT								202
39E2h	—	未实现								—
39E1h	—	未实现								—
39E0h	NVMADRL	ADR								201
39DFh	OSCFRQ	—	—	—	—	FRQ				97
39DEh	OSCTUNE	—	—	TUN						98
39DDh	OSCEN	EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—	99
39DCh	OSCSTAT	EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLLR	96
39DBh	OSCCON3	CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—	95
39DAh	OSCCON2	—	COSC			CDIV				95
39D9h	写	—	NOSC			NDIV				94
39D8h	CPUDOZE	IDLEN	DOZEN	ROI	DOE	—	DOZE			167

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
39D7h - 39D2h	—	未实现								—
39D1h	VREGCON(1)	—	—	—	—	—	—	VREGPM	—	166
39D0h	BORCON	SBOREN	—	—	—	—	—	—	BORRDY	75
39CFh - 39C8h	—	未实现								—
39C7h	PMD7	CANMD	—	—	—	—	—	DMA2MD	DMA1MD	282
39C6h	PMD6	—	SMT2MD	SMT1MD	CLC4MD	CLC3MD	CLC2MD	CLC1MD	DSMMD	281
39C5h	PMD5	—	—	U2MD	U1MD	—	SPI1MD	I2C2MD	I2C1MD	280
39C4h	PMD4	CWG3MD	CWG2MD	CWG1MD	—	—	—	—	—	279
39C3h	PMD3	PWM8MD	PWM7MD	PWM6MD	PWM5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	278
39C2h	PMD2	—	DACMD	ADCMD	—	—	CMP2MD	CMP1MD	ZCDMD	277
39C1h	PMD1	NCO1MD	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	276
39C0h	PMD0	SYSCMD	FVRMD	HLVDMD	CRCMD	SCANMD	NVMMD	CLKRMD	IOCMD	275
39BFh - 39AAh	—	未实现								—
39A9h	PIR9	—	CLC4IF	CCP4IF	CLC3IF	CWG3IF	CCP3IF	TMR6IF	TMR5IF	136
39A8h	PIR8	TMR5IF	INT2IF	CLC2IF	CWG2IF	CCP2IF	TMR4IF	TMR3GIF	TMR3IF	135
39A7h	PIR7	U2IF	U2EIF	U2TXIF	U2RXIF	I2C2EIF	I2C2IF	I2C2TXIF	I2C2RXIF	134
39A6h	PIR6	DMA2AIF	DMA2ORIF	DMA2DCNTIF	DMA2SCNTIF	SMT2PWAIF	SMT2PRAIF	SMT2IF	C2IF	133
39A5h	PIR5	IRXIF	WAKIF	ERRIF	TXB2IF/ TXBnIF	TXB1IF	TXB0IF	RXB1IF/ RXBnIF	RXB0IF/ FIFOFIF	132
39A4h	PIR4	INT1IF	CLC1IF	CWG1IF	NCO1IF	CCP1IF	TMR2IF	TMR1GIF	TMR1IF	131
39A3h	PIR3	TMR0IF	U1IF	U1EIF	U1TXIF	U1RXIF	I2C1EIF	I2C1IF	I2C1TXIF	130
39A2h	PIR2	I2C1RXIF	SPI1IF	SPI1TXIF	SPI1RXIF	DMA1AIF	DMA1ORIF	DMA1DCNTIF	DMA1SCNTIF	128
39A1h	PIR1	SMT1PWAIF	SMT1PRAIF	SMT1IF	C1IF	ADTIF	ADIF	ZCDIF	INT0IF	128
39A0h	PIR0	IOCIF	CRCIF	SCANIF	NVMIF	CSWIF	OSFIF	HLVDIF	SWIF	127
399Fh - 399Ah	—	未实现								—
3999h	PIE9	—	CLC4IE	CCP4IE	CLC3IE	CWG3IE	CCP3IE	TMR6IE	TMR5IE	146
3998h	PIE8	TMR5IE	INT2IE	CLC2IE	CWG2IE	CCP2IE	TMR4IE	TMR3GIE	TMR3IE	145
3997h	PIE7	U2IE	U2EIE	U2TXIE	U2RXIE	I2C2EIE	I2C2IE	I2C2TXIE	I2C2RXIE	144
3996h	PIE6	DMA2AIE	DMA2ORIE	DMA2DCNTIE	DMA2SCNTIE	SMT2PWAIE	SMT2PRAIE	SMT2IE	C2IE	143
3995h	PIE5	IRXIE	WAKIE	ERRIE	TXB2IE/ TXBnIE	TXB1IE	TXB0IE	RXB1IE/ RXBnIE	RXB0IF/ FIFOFIF	142
3994h	PIE4	INT1IE	CLC1IE	CWG1IE	NCO1IE	CCP1IE	TMR2IE	TMR1GIE	TMR1IE	141
3993h	PIE3	TMR0IE	U1IE	U1EIE	U1TXIE	U1RXIE	I2C1EIE	I2C1IE	I2C1TXIE	140
3992h	PIE2	I2C1RXIE	SPI1IE	SPI1TXIE	SPI1RXIE	DMA1AIE	DMA1ORIE	DMA1DCNTIE	DMA1SCNTIE	139
3991h	PIE1	SMT1PWAIE	SMT1PRAIE	SMT1IE	C1IE	ADTIE	ADIE	ZCDIE	INT0IE	138
3990h	PIE0	IOCIE	CRCIE	SCANIE	NVMIE	CSWIE	OSFIE	HLVDIE	SWIE	137
398Fh - 398Ah	—	未实现								—
3989h	IPR9	—	CLC4IP	CCP4IP	CLC3IP	CWG3IP	CCP3IP	TMR6IP	TMR5IP	156
3988h	IPR8	TMR5IP	INT2IP	CLC2IP	CWG2IP	CCP2IP	TMR4IP	TMR3GIP	TMR3IP	155
3987h	IPR7	U2IP	U2EIP	U2TXIP	U2RXIP	I2C2EIP	I2C2IP	I2C2TXIP	I2C2RXIP	154
3986h	IPR6	DMA2AIP	DMA2ORIP	DMA2DCNTIP	DMA2SCNTIP	SMT2PWAIP	SMT2PRAIP	SMT2IP	C2IP	153
3985h	IPR5	IRXIP	WAKIP	ERRIP	TXB2IP/ TXBnIP	TXB1IP	TXB0IP	RXB1IP/ RXBnIP	RXB0IP/ FIFOFIP	152
3984h	IPR4	INT1IP	CLC1IP	CWG1IP	NCO1IP	CCP1IP	TMR2IP	TMR1GIP	TMR1IP	151
3983h	IPR3	TMR0IP	U1IP	U1EIP	U1TXIP	U1RXIP	I2C1EIP	I2C1IP	I2C1TXIP	150
3982h	IPR2	I2C1RXIP	SPI1IP	SPI1TXIP	SPI1RXIP	DMA1AIP	DMA1ORIP	DMA1DCNTIP	DMA1SCNTIP	149
3981h	IPR1	SMT1PWAIP	SMT1PRAIP	SMT1IP	C1IP	ADTIP	ADIP	ZCDIP	INT0IP	148

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3980h	IPR0	IOCIP	CRCIP	SCANIP	NVMIP	CSWIP	OSFIP	HLVDIP	SWIP	147
397Fh - 397Eh	—	未实现								—
397Dh	SCANTRIG	—	—	—	—	TSEL				216
397Ch	SCANCON0	EN	TRIGEN	SGO	—	—	MREG	BURSTMD	BUSY	212
397Bh	SCANHADRU	—	—	HADR						214
397Ah	SCANHADRH	HADR								215
3979h	SCANHADRL	HADR								215
3978h	SCANLADRU	—	—	LADR						213
3977h	SCANLADRH	LADR								213
3976h	SCANLADRL	LADR								214
3975h - 396Ah	—	未实现								—
3969h	CRCCON1	DLEN				PLEN				208
3968h	CRCCON0	EN	CRCGO	BUSY	ACCM	—	—	SHIFTM	FULL	208
3967h	CRCXORH	X15	X14	X13	X12	X11	X10	X9	X8	211
3966h	CRCXORL	X7	X6	X5	X4	X3	X2	X1	—	211
3965h	CRCSHIFTH	SHFT15	SHFT14	SHFT13	SHFT12	SHFT11	SHFT10	SHFT9	SHFT8	210
3964h	CRCSHIFTL	SHFT7	SHFT6	SHFT5	SHFT4	SHFT3	SHFT2	SHFT1	SHFT0	210
3963h	CRCACCH	ACC15	ACC14	ACC13	ACC12	ACC11	ACC10	ACC9	ACC8	209
3962h	CRCACCL	ACC7	ACC6	ACC5	ACC4	ACC3	ACC2	ACC1	ACC0	210
3961h	CRCDATH	DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8	209
3960h	CRCDATL	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0	209
395Fh	WDTTMR	WDTTMR					STATE	PSCNT		175
395Eh	WDTPSH	PSCNT								174
395Dh	WDTPSL	PSCNT								174
395Ch	WDTCON1	—	WDTCS			—	WINDOW			173
395Bh	WDTCON0	—	—	WDTPS					SEN	172
395Ah - 38A0h	—	未实现								—
389Fh	IVTADU	AD								158
389Eh	IVTADH	AD								158
389Dh	IVTADL	AD								158
389Ch - 3891h	—	未实现								—
3890h	PRODH_SHAD	PRODH								115
388Fh	PRODL_SHAD	PRODL								115
388Eh	FSR2H_SHAD	—	—	FSR2H						115
388Dh	FSR2L_SHAD	FSR2L								115
388Ch	FSR1H_SHAD	—	—	FSR1H						115
388Bh	FSR1L_SHAD	FSR1L								115
388Ah	FSR0H_SHAD	—	—	FSR0H						115
3889h	FSR0L_SHAD	FSR0L								115
3888h	PCLATU_SHAD	—	—	—	PCU					115
3887h	PCLATH_SHAD	PCH								115
3886h	BSR_SHAD	—	—	BSR						115
3885h	WREG_SHAD	WREG								115
3884h	STATUS_SHAD	—	$\overline{TO}$	$\overline{PD}$	N	OV	Z	DC	C	115
3883h	SHADCON	—	—	—	—	—	—	—	SHADLO	159
3882h	BSR_CSHAD	—	—	BSR						47
3881h	WREG_CSHAD	WREG								47

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3880h	STATUS_CSHAD	—	$\overline{TO}$	$\overline{PD}$	N	OV	Z	DC	C	47
387Fh - 3800h	—	未实现								—
37FFh	CANCON_RO0	CANCON_RO0								603
37FEh	CANSTAT_RO0	CANSTAT_RO0								604
37FDh	RXB1D7	RXB1D7								620
37FCh	RXB1D6	RXB1D6								620
37FBh	RXB1D5	RXB1D5								620
37FAh	RXB1D4	RXB1D4								620
37F9h	RXB1D3	RXB1D3								620
37F8h	RXB1D2	RXB1D2								620
37F7h	RXB1D1	RXB1D1								620
37F6h	RXB1D0	RXB1D0								620
37F5h	RXB1DLC	—	RXRTR	RB1	R0	DLC3	DLC2	DLC1	DLC0	620
37F4h	RXB1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	619
37F3h	RXB1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	619
37F2h	RXB1SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	619
37F1h	RXB1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	618
37F0h	RXB1CON	RXFUL	RXM1	RXM0	—	RXRTRRO	FILHIT2	FILHIT1	FILHIT0	617
37F0h	RXB1CON	RXFUL	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	617
37EFh	CANCON_RO1	CANCON_RO1								603
37EEh	CANSTAT_RO1	CANSTAT_RO1								604
37EDh	TXB0D7	TXB0D7								611
37ECh	TXB0D6	TXB0D6								611
37EBh	TXB0D5	TXB0D5								611
37EAh	TXB0D4	TXB0D4								611
37E9h	TXB0D3	TXB0D3								611
37E8h	TXB0D2	TXB0D2								611
37E7h	TXB0D1	TXB0D1								611
37E6h	TXB0D0	TXB0D0								611
37E5h	TXB0DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	612
37E4h	TXB0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	611
37E3h	TXB0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	610
37E2h	TXB0SIDL	SID2	SID1	SID0	—	EXIDE	—	EID17	EID16	610
37E1h	TXB0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	610
37E0h	TXB0CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	609
37DFh	CANCON_RO2	CANCON_RO2								603
37DEh	CANSTAT_RO2	CANSTAT_RO2								604
37DDh	TXB1D7	TXB1D7								611
37DCh	TXB1D6	TXB1D6								611
37DBh	TXB1D5	TXB1D5								611
37DAh	TXB1D4	TXB1D4								611
37D9h	TXB1D3	TXB1D3								611
37D8h	TXB1D2	TXB1D2								611
37D7h	TXB1D1	TXB1D1								611
37D6h	TXB1D0	TXB1D0								611
37D5h	TXB1DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	612
37D4h	TXB1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	611
37D3h	TXB1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	610

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
37D2h	TXB1SIDL	SID2	SID1	SID0	—	EXIDE	—	EID17	EID16	610
37D1h	TXB1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	610
37D0h	TXB1CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	609
37CFh	CANCON_R03	CANCON_RO3								603
37CEh	CANSTAT_R03	CANSTAT_RO3								604
37CDh	TXB2D7	TXB2D7								611
37CCh	TXB2D6	TXB2D6								611
37CBh	TXB2D5	TXB2D5								611
37CAh	TXB2D4	TXB2D4								611
37C9h	TXB2D3	TXB2D3								611
37C8h	TXB2D2	TXB2D2								611
37C7h	TXB2D1	TXB2D1								611
37C6h	TXB2D0	TXB2D0								611
37C5h	TXB2DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	612
37C4h	TXB2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	611
37C3h	TXB2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	610
37C2h	TXB2SIDL	SID2	SID1	SID0	—	EXIDE	—	EID17	EID16	610
37C1h	TXB2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	610
37C0h	TXB2CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	609
37BFh	RXM1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	633
37BEh	RXM1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	632
37BDh	RXM1SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	632
37BCh	RXM1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	631
37BBh	RXM0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	633
37BAh	RXM0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	632
37B9h	RXM0SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	631
37B8h	RXM0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	631
37B7h	RXF5EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
37B6h	RXF5EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
37B5h	RXF5SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
37B4h	RXF5SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
37B3h	RXF4EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
37B2h	RXF4EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
37B1h	RXF4SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
37B0h	RXF4SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
37AFh	RXF3EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
37AEh	RXF3EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
37ADh	RXF3SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
37ACh	RXF3SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
37ABh	RXF2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
37AAh	RXF2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
37A9h	RXF2SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
37A8h	RXF2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
37A7h	RXF1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
37A6h	RXF1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
37A5h	RXF1SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
37A4h	RXF1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
37A3h	RXF0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
37A2h	RXF0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
37A1h	RXF0SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
37A0h	RXF0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
379Fh	CANCON_RO4	CANCON_RO4								603
379Eh	CANSTAT_RO4	CANSTAT_RO4								604
379Dh	B5D7	B5D7								627
379Ch	B5D6	B5D6								627
379Bh	B5D5	B5D5								627
379Ah	B5D4	B5D4								627
3799h	B5D3	B5D3								627
3798h	B5D2	B5D2								627
3797h	B5D1	B5D1								627
3796h	B5D0	B5D0								627
3795h	B5DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	628
3795h	B5DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	629
3794h	B5EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	626
3793h	B5EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	626
3792h	B5SIDL	SID2	SID1	SID0	SRR	EXIDE	—	EID17	EID16	625
3791h	B5SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	624
3790h	B5CON	RXFUL	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	622
3790h	B5CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0	623
378Fh	CANCON_RO5	CANCON_RO5								603
378Eh	CANSTAT_RO5	CANSTAT_RO5								604
378Dh	B4D7	B4D7								627
378Ch	B4D6	B4D6								627
378Bh	B4D5	B4D5								627
378Ah	B4D4	B4D4								627
3789h	B4D3	B4D3								627
3788h	B4D2	B4D2								627
3787h	B4D1	B4D1								627
3786h	B4D0	B4D0								627
3785h	B4DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	628
3785h	B4DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	629
3784h	B4EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	626
3783h	B4EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	626
3782h	B4SIDL	SID2	SID1	SID0	SRR	EXIDE	—	EID17	EID16	625
3781h	B4SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	624
3780h	B4CON	RXFUL	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	622
3780h	B4CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0	623
377Fh	CANCON_RO6	CANCON_RO6								603
377Eh	CANSTAT_RO6	CANSTAT_RO6								604
377Dh	B3D7	B3D7								627
377Ch	B3D6	B3D6								627
377Bh	B3D5	B3D5								627
377Ah	B3D4	B3D4								627
3779h	B3D3	B3D3								627
3778h	B3D2	B3D2								627
3777h	B3D1	B3D1								627
3776h	B3D0	B3D0								627
3775h	B3DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	628
3775h	B3DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	629
3774h	B3EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	626

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3773h	B3EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	626
3772h	B3SIDL	SID2	SID1	SID0	SRR	EXIDE	—	EID17	EID16	625
3771h	B3SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	624
3770h	B3CON	RXFUL	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	622
3770h	B3CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0	623
376Fh	CANCON_RO7	CANCON_RO7								603
376Eh	CANSTAT_RO7	CANSTAT_RO7								604
376Dh	B2D7	B2D7								627
376Ch	B2D6	B2D6								627
376Bh	B2D5	B2D5								627
376Ah	B2D4	B2D4								627
3769h	B2D3	B2D3								627
3768h	B2D2	B2D2								627
3767h	B2D1	B2D1								627
3766h	B2D0	B2D0								627
3765h	B2DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	628
3765h	B2DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	629
3764h	B2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	626
3763h	B2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	626
3762h	B2SIDL	SID2	SID1	SID0	SRR	EXIDE	—	EID17	EID16	625
3761h	B2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	624
3760h	B2CON	RXFUL	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	622
3760h	B2CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0	623
375Fh	CANCON_RO8	CANCON_RO8								603
375Eh	CANSTAT_RO8	CANSTAT_RO8								604
375Dh	B1D7	B1D7								627
375Ch	B1D6	B1D6								627
375Bh	B1D5	B1D5								627
375Ah	B1D4	B1D4								627
3759h	B1D3	B1D3								627
3758h	B1D2	B1D2								627
3757h	B1D1	B1D1								627
3756h	B1D0	B1D0								627
3755h	B1DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	628
3755h	B1DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	629
3754h	B1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	626
3753h	B1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	626
3752h	B1SIDL	SID2	SID1	SID0	SRR	EXIDE	—	EID17	EID16	625
3751h	B1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	624
3750h	B1CON	RXFUL	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	622
3750h	B1CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0	623
374Fh	CANCON_RO9	CANCON_RO9								603
374Eh	CANSTAT_RO9	CANSTAT_RO9								604
374Dh	B0D7	B0D7								627
374Ch	B0D6	B0D6								627
374Bh	B0D5	B0D5								627
374Ah	B0D4	B0D4								627
3749h	B0D3	B0D3								627
3748h	B0D2	B0D2								627
3747h	B0D1	B0D1								627

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3746h	B0D0	B0D0								627
3745h	B0DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	628
3745h	B0DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	629
3744h	B0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	626
3743h	B0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	626
3742h	B0SIDL	SID2	SID1	SID0	SRR	EXIDE	—	EID17	EID16	625
3741h	B0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	624
3740h	B0CON	RXFUL	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	622
3740h	B0CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0	623
373Fh	TXBIE	—	—	—	TXB2IE	TXB1IE	TXB0IE	—	—	647
373Eh	BIE0	B5IE	B4IE	B3IE	B2IE	B1IE	B0IE	RXB1IE	RXB0IE	648
373Dh	BSEL0	B5TXEN	B4TXEN	B3TXEN	B2TXEN	B1TXEN	B0TXEN	—	—	629
373Ch	MSEL3	FIL15_1	FIL15_0	FIL14_1	FIL14_0	FIL13_1	FIL13_0	FIL12_1	FIL12_0	639
373Bh	MSEL2	FIL11_1	FIL11_0	FIL10_1	FIL10_0	FIL9_1	FIL9_0	FIL8_1	FIL8_0	638
373Ah	MSEL1	FIL7_1	FIL7_0	FIL6_1	FIL6_0	FIL5_1	FIL5_0	FIL4_1	FIL4_0	637
3739h	MSEL0	FIL3_1	FIL3_0	FIL2_1	FIL2_0	FIL1_1	FIL1_0	FIL0_1	FIL0_0	639
3738h	RXFBCON7	F15BP_3	F15BP_2	F15BP_1	F15BP_0	F14BP_3	F14BP_2	F14BP_1	F14BP_0	635
3737h	RXFBCON6	F13BP_3	F13BP_2	F13BP_1	F13BP_0	F12BP_3	F12BP_2	F12BP_1	F12BP_0	635
3736h	RXFBCON5	F11BP_3	F11BP_2	F11BP_1	F11BP_0	F10BP_3	F10BP_2	F10BP_1	F10BP_0	635
3735h	RXFBCON4	F9BP_3	F9BP_2	F9BP_1	F9BP_0	F8BP_3	F8BP_2	F8BP_1	F8BP_0	635
3734h	RXFBCON3	F7BP_3	F7BP_2	F7BP_1	F7BP_0	F6BP_3	F6BP_2	F6BP_1	F6BP_0	635
3733h	RXFBCON2	F5BP_3	F5BP_2	F5BP_1	F5BP_0	F4BP_3	F4BP_2	F4BP_1	F4BP_0	635
3732h	RXFBCON1	F3BP_3	F3BP_2	F3BP_1	F3BP_0	F2BP_3	F2BP_2	F2BP_1	F2BP_0	635
3731h	RXFBCON0	F1BP_3	F1BP_2	F1BP_1	F1BP_0	F0BP_3	F0BP_2	F0BP_1	F0BP_0	635
3730h	SDFLC	—	—	—	FLC4	FLC3	FLC2	FLC1	FLC0	634
372Fh	RXF15EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
372Eh	RXF15EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
372Dh	RXF15SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
372Ch	RXF15SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
372Bh	RXF14EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
372Ah	RXF14EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
3729h	RXF14SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
3728h	RXF14SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
3727h	RXF13EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
3726h	RXF13EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
3725h	RXF13SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
3724h	RXF13SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
3723h	RXF12EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
3722h	RXF12EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
3721h	RXF12SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
3720h	RXF12SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
371Fh	RXF11EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
371Eh	RXF11EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
371Dh	RXF11SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
371Ch	RXF11SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
371Bh	RXF10EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
371Ah	RXF10EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
3719h	RXF10SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
3718h	RXF10SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
3717h	RXF9EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

表43-1： PIC18(L)F25/26K83器件的寄存器文件汇总（续）

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	寄存器所在页
3716h	RXF9EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
3715h	RXF9SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
3714h	RXF9SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
3713h	RXF8EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
3712h	RXF8EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
3711h	RXF8SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
3710h	RXF8SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
370Fh	RXF7EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
370Eh	RXF7EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
370Dh	RXF7SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
370Ch	RXF7SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
370Bh	RXF6EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	631
370Ah	RXF6EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	631
3709h	RXF6SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	630
3708h	RXF6SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	630
3707h	RXFCON1	RXF15EN	RXF14EN	RXF13EN	RXF12EN	RXF11EN	RXF10EN	RXF9EN	RXF8EN	633
3706h	RXFCON0	RXF7EN	RXF6EN	RXF5EN	RXF4EN	RXF3EN	RXF2EN	RXF1EN	RXF0EN	633
3705h	BRGCON3	WAKDIS	WAKFIL	—	—	—	SEG2PH2	SEG2PH1	SEG2PH0	642
3704h	BRGCON2	SEG2PHTS	SAM	SEG1PH2	SEG1PH1	SEG1PH0	PRSEG2	PRSEG1	PRSEG0	641
3703h	BRGCON1	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	640
3702h	TXERRCNT	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	612
3701h	RXERRCNT	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	621
3700h	CIOCON	TX1SRC	—	—	—	—	—	—	CLKSEL	643

图注： x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件
注 1： LF器件中不存在。

44.0 开发支持

一系列软件及硬件开发工具对 PIC® 单片机（MCU）和 dsPIC® 数字信号控制器（DSC）提供支持：

- 集成开发环境
  - MPLAB® X IDE 软件
- 编译器 / 汇编器 / 链接器
  - MPLAB XC 编译器
  - MPASM™ 汇编器
  - MPLINK™ 目标链接器 / MPLIB™ 目标库管理器
  - 适用于各种器件系列的 MPLAB 汇编器 / 链接器 / 库管理器
- 模拟器
  - MPLAB X SIM 软件模拟器
- 仿真器
  - MPLAB REAL ICE™ 在线仿真器
- 在线调试器 / 编程器
  - MPLAB ICD 3
  - PICkit™ 3
- 器件编程器
  - MPLAB PM3 器件编程器
- 低成本演示 / 开发板、评估工具包及入门工具包
- 第三方开发工具

44.1 MPLAB X 集成开发环境软件

MPLAB X IDE 是适用于 Microchip 和第三方软硬件开发工具统一的通用图形用户界面，可以在 Windows®、Linux 和 Mac OS® X 上运行。MPLAB X IDE 是一款全新的 IDE，它基于 NetBeans IDE，包含许多免费的软件组件和插件，适用于高性能的应用程序开发和调试。通过这一无缝交互的用户界面，在不同工具之间的迁移以及从软件模拟器到硬件调试和编程工具的升级都变得极为简便。

MPLAB X IDE 具有完善的项目管理、可视化的调用图、可配置的观察窗口以及包含代码补全功能和上下文菜单的功能丰富编辑器，因此对于新用户来说非常灵活和友好。 MPLAB X IDE 支持对多个项目使用多个工具和同时调试，因此也完全可以满足经验丰富用户的需求。

功能丰富的编辑器：

- 彩色高亮显示语法
- 智能代码补全功能，在输入代码时提供建议和提示
- 基于用户定义规则，代码自动格式化
- 即时解析

用户友好的可定制界面：

- 完全可定制界面：工具栏、工具栏图标、窗口和窗口放置等
- 调用图窗口

基于项目的工作空间：

- 多个项目
- 多个工具
- 多种配置
- 同时调试会话

文件历史和错误跟踪：

- 本地文件历史功能
- 内建对 Bugzilla 缺陷跟踪系统的支持

44.2 MPLAB XC 编译器

MPLAB XC编译器是适用于Microchip所有8位、16位和32位MCU以及DSC器件的完全ANSI C编译器。这些编译器提供强大的集成功能以及出色的代码优化功能，且易于使用。MPLAB XC 编译器可在 Windows、Linux 或 Mac OS X 上运行。

为方便进行源代码级调试，编译器提供了已针对 MPLAB X IDE 优化的调试信息。

MPLAB XC 编译器的免费版支持所有器件和命令，没有时间或存储容量限制，且为大多数应用程序提供了充分的代码优化。

MPLAB XC 编译器包含汇编器、链接器和实用程序。汇编器生成可重定位目标文件，然后通过链接器将生成的可重定位目标文件与其他可重定位目标文件或归档文件归档或链接在一起，进而生成可执行文件。MPLAB XC 编译器使用汇编器来生成目标文件。汇编器具有如下突出特性：

- 支持全部器件指令集
- 支持定点和浮点数据
- 命令行接口
- 丰富的伪指令集
- 灵活的宏语言
- 与 MPLAB X IDE 兼容

44.3 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于 PIC10/12/16/18 MCU。

MPASM 汇编器可生成用于 MPLINK 目标链接器的可重定位目标文件、Intel® 标准 HEX 文件、详细描述存储器使用状况和符号参考的 MAP 文件、包含源代码行及生成机器码的绝对 LST 文件以及用于调试的 COFF 文件。

MPASM 汇编器具有如下特性：

- 集成在 MPLAB X IDE 项目中
- 用户定义的宏可简化汇编代码
- 对多用途源文件进行条件汇编
- 允许完全控制汇编过程的指令

44.4 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器组合由 MPASM 汇编器生成的可重定位目标文件。通过使用链接器脚本中的伪指令，它还可链接预编译库中的可重定位目标文件。

MPLIB目标库管理器管理预编译代码库文件的创建和修改。当从源文件调用库中的一段子程序时，只有包含此子程序的模块被链接到应用程序。这样可使大型库在许多不同应用中被高效地利用。

目标链接器 / 库管理器具有如下特性：

- 高效地连接单个的库而不是许多小文件
- 通过将相关的模块组合在一起来增强代码的可维护性
- 只要列出、替换、删除和抽取模块，便可灵活地创建库

44.5 适用于各种器件系列的 MPLAB 汇编器、链接器和库管理器

MPLAB汇编器为PIC24和PIC32 MCU以及dsPIC DSC器件从符号汇编语言生成可重定位机器码。MPLAB XC 编译器使用该汇编器生成目标文件。汇编器产生可重定位目标文件之后，可将这些目标文件存档，或与其他可重定位目标文件和存档链接以生成可执行文件。该汇编器有如下显著特性：

- 支持整个器件指令集
- 支持定点和浮点数据
- 命令行接口
- 丰富的指令集
- 与 MPLAB X IDE 兼容

44.6 MPLAB X SIM 软件模拟器

MPLAB X SIM 软件模拟器通过在指令级对 PIC MCU 和 dsPIC DSC 进行模拟，可在 PC 主机环境下进行代码开发。对于任何给定的指令，都可以对数据区进行检查或修改，并通过一个全面的激励控制器来施加激励。可以将各寄存器记录在文件中，以便进行进一步的运行时分析。跟踪缓冲区和逻辑分析器的显示使软件模拟器还能记录和跟踪程序的执行、 I/O 的动作、大部分的外设及内部寄存器。

MPLAB X SIM 软件模拟器完全支持使用 MPLAB XC 编译器以及 MPASM 和 MPLAB 汇编器的符号调试。该软件模拟器可用于在硬件实验室环境外灵活地开发和调试代码，是一款完美且经济的软件开发工具。

44.7 MPLAB REAL ICE 在线仿真器系统

MPLAB REAL ICE 在线仿真器系统是 Microchip 针对其闪存 DSC 和 MCU 器件推出的新一代高速仿真器。结合 MPLAB X IDE 易于使用且功能强大的图形用户界面，该仿真器可对所有 8 位、16 位和 32 位 MCU 及 DSC 器件进行调试和编程。

该仿真器通过高速 USB 2.0 接口与设计工程师的 PC 相连，并利用与在线调试器系统兼容的连接器（RJ-11）或新型抗噪声、高速低压差分信号 （LVDS）互连电缆（CAT5）与目标板相连。

可通过 MPLAB X IDE 下载将来版本的固件，对该仿真器进行现场升级。在同类仿真器中，MPLAB REAL ICE 的优势十分明显：全速仿真、运行时变量观察、跟踪分析、复杂断点、逻辑探针、耐用的探针接口及较长 （长达 3 米）的互连电缆。

44.8 MPLAB ICD 3 在线调试器系统

MPLAB ICD 3 在线调试器系统是 Microchip 成本效益最高的高速硬件调试器 / 编程器，适用于 Microchip 的闪存 DSC 和 MCU 器件。结合 MPLAB X IDE 功能强大但易于使用的图形用户界面，该调试器可对 PIC 闪存单片机和 dsPIC DSC 进行调试和编程。

MPLAB ICD 3 在线调试器通过高速 USB 2.0 接口与设计工程师的PC相连，并利用与MPLAB ICD 2或MPLAB REAL ICE 系统兼容的连接器（RJ-11）与目标板相连。MPLAB ICD 3 支持所有 MPLAB ICD 2 连接器。

44.9 PICkit 3 在线调试器 / 编程器

结合 MPLAB X IDE 功能强大的图形用户界面，MPLAB PICkit 3 可对 PIC 闪存单片机和 dsPIC 数字信号控制器进行调试和编程，且价位较低。MPLAB PICkit 3 通过全速USB接口与设计工程师的PC相连，并利用Microchip 调试连接器 （RJ-11）（与 MPLAB ICD 3 和 MPLAB REAL ICE 兼容）与目标板相连。连接器使用两个器件 I/O 引脚和复位线来实现在线调试和在线串行编程 （In-Circuit Serial Programming™， ICSP™）。

PICkit 3 Debug Express 包括 PICkit 3、演示板和单片机、连接电缆和光盘（内含用户指南、课程、教程、编译器和 MPLAB IDE 软件）。

44.10 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款符合 CE 规范的通用器件编程器，在 V_{DDMIN} 和 V_{DDMAX} 点对其可编程电压进行校验以确保可靠性最高。它有一个用来显示菜单和错误消息的大 LCD 显示器（128 x 64），以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 ICSP 电缆。在单机模式下， MPLAB PM3 器件编程器不必与 PC 相连即可对 PIC MCU 和 dsPIC DSC 器件进行读取、校验和编程。在该模式下它还可设置代码保护。MPLAB PM3 通过 RS-232 或 USB 电缆连接到 PC 主机上。 MPLAB PM3 具备高速通信能力以及优化算法，可对具有大存储器的器件进行快速编程。它还包含了 MMC 卡，用于文件存储及数据应用。

44.11 演示 / 开发板、评估工具包及入门工具包

有许多演示、开发和评估板可用于各种 PIC MCU 和 dsPIC DSC，实现对全功能系统的快速应用开发。大多数的演示、开发和评估板都有实验布线区，供用户添加定制电路；还有应用固件和源代码，用于检查和修改。

这些板支持多种功能部件，包括 LED、温度传感器、开关、扬声器、 RS-232 接口、 LCD 显示器、电位计和附加 EEPROM 存储器。

演示和开发板可用于教学环境，在实验布线区设计定制电路，从而掌握各种单片机应用。

除了 PICDEM™ 和 dsPICDEM™ 演示 / 开发板系列电路外，Microchip 还有一系列评估工具包和演示软件，适用于模拟滤波器设计、KEELOQ® 数据安全产品 IC、CAN、IrDA®、 PowerSmart 电池管理、 SEEVAL® 评估系统、Σ–Δ ADC、流速传感器，等等。

同时还提供入门工具包，其中包含体验指定器件功能所需的所有软硬件。通常提供单个应用以及调试功能，都包含在一块电路板上。

有关演示、开发和评估工具包的完整列表，请访问 Microchip 网站 （www.microchip.com）。

44.12 第三方开发工具

Microchip 还提供一些来自第三方供应商的优秀开发工具。这些工具均经过精心挑选，功能独特，物有所值。

- SoftLog 和 CCS 等公司提供的器件编程器和量产编程器
- Gimpel 和 Trace Systems 等公司提供的软件工具
- Saleae 和 Total Phase 等公司提供的协议分析器
- MikroElektronika、Digilent® 和 Olimex 等公司提供的演示板
- EZ Web Lynx、WIZnet 和 IPLogika® 等公司提供的嵌入式以太网解决方案

45.0 电气规范

45.1 绝对最大值(†)

偏置时的环境温度 ..... -40°C至+125°C
储存温度..... -65°C至+150°C
引脚相对于VSS的电压
VDD引脚
PIC18F25/26K83 ..... -0.3V至+6.5V
PIC18LF25/26K83 ..... -0.3V至+4.0V
$\overline{MCLR}$引脚 ..... -0.3V至+9.0V
所有其他引脚 ..... -0.3V至（VDD + 0.3V）
最大电流
VSS引脚(1)
–40°C ≤ TA ≤ +85°C ..... 350 mA
85°C < TA ≤ +125°C ..... 120 mA
VDD引脚（28引脚器件）(1)
–40°C ≤ TA ≤ +85°C ..... 250 mA
+85°C < TA ≤ +125°C ..... 85 mA
任意标准I/O引脚 ..... ±50 mA
钳位电流，IK（VPIN < 0或VPIN > VDD） ..... ±20 mA
总功耗(2)..... 800 mW

注 1： 最大电流值要求I/O引脚上具有均匀的负载分布。最大电流可以通过器件封装功率耗散特性进行限制，请参见表45-6来计算器件规范值。

2： 功耗按如下公式计算：

$$P_{DIS} = V_{DD} \times \{I_{DD} - \Sigma I_{OH}\} + \Sigma \{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma (V_{OI} \times I_{OL})$$

†注：如果器件工作条件超过上述“绝对最大值”，可能对器件造成永久性损坏。上述值仅代表本规范规定的极限工作条件，不代表器件在上述极限值或超出极限值的情况下仍可正常工作。器件长时间工作在最大值条件下，其可靠性会受到影响。

45.2 标准工作条件

所有器件的标准工作条件定义如下：

工作电压： $V_{DDMIN} \le V_{DD} \le V_{DDMAX}$
工作温度： $T_{A\_MIN} \le T_A \le T_{A\_MAX}$

V_{DD}——工作电源电压范围(1)

PIC18LF25/26K83

V_{DDMIN}（Fosc ≤ 16 MHz） ..... +1.8V
V_{DDMIN}（Fosc ≤ 32 MHz） ..... +2.5V
V_{DDMIN}（Fosc ≤ 64 MHz） ..... +2.7V
V_{DDMAX} ..... +3.6V

PIC18F25/26K83

V_{DDMIN} (Fosc ≤ 16 MHz) ..... +2.3V
V_{DDMIN}（Fosc ≤ 32 MHz） ..... +2.5V
V_{DDMIN}（Fosc ≤ 64 MHz） ..... +3.0V
V_{DDMAX} ..... +5.5V

T_A——工作环境温度范围

工业级温度

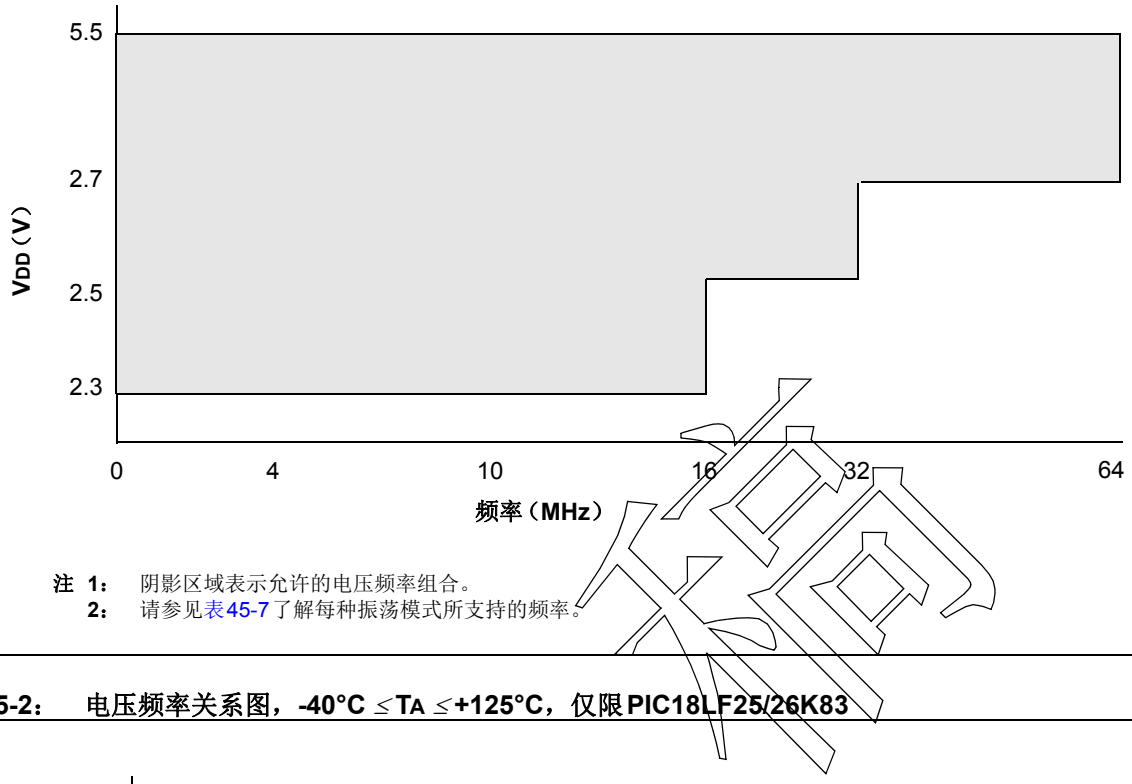
T_{A_MIN} ..... –40°C
T_{A_MAX} ..... +85°C

扩展级温度

T_{A_MIN} ..... –40°C
T_{A_MAX} ..... +125°C

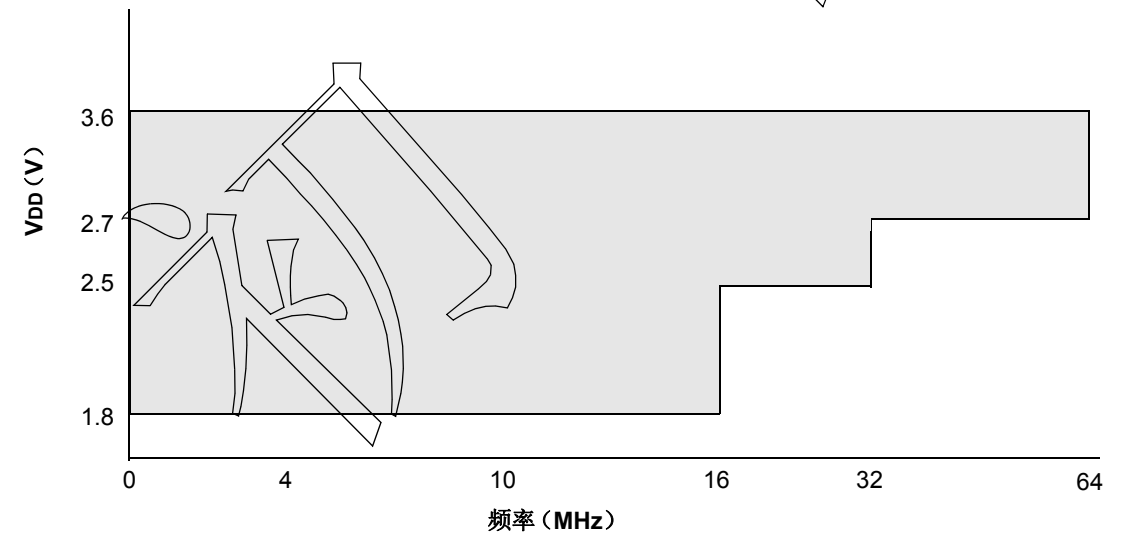
注 1： 请参见参数电源电压，DS特性：电源电压。

图45-1： 电压频率关系图，-40°C ≤TA ≤+125°C，仅限PIC18F25/26K83



注 1： 阴影区域表示允许的电压频率组合。
2： 请参见表45-7了解每种振荡模式所支持的频率。

图45-2： 电压频率关系图，-40°C ≤TA ≤+125°C，仅限PIC18LF25/26K83



注 1： 阴影区域表示允许的电压频率组合。
2： 请参见表45-7了解每种振荡模式所支持的频率。

45.3 直流特性

表45-1： 电源电压

PIC18LF25/26K83			标准工作条件（除非另外声明）				
PIC18F25/26K83							
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
电源电压							
D002	VDD		1.8 2.5 2.7	— — —	3.6 3.6 3.6	V V V	FOSC ≤ 16 MHz FOSC > 16 MHz FOSC > 32 MHz
D002	VDD		2.3 2.5 2.7	— — —	5.5 5.5 5.5	V V V	FOSC ≤ 16 MHz FOSC > 16 MHz FOSC > 32 MHz
RAM数据保持电压(1)							
D003	VDR		1.5	—	—	V	器件处于休眠模式
D003	VDR		1.7	—	—	V	器件处于休眠模式
上电复位释放电压(2)							
D004	VPOR		—	1.6	—	V	禁止BOR或LPBOR(3)
D004	VPOR		—	1.6	—	V	禁止BOR或LPBOR(3)
上电复位重新激活电压(2)							
D005	VPORR		—	0.8	—	V	禁止BOR或LPBOR(3)
D005	VPORR		—	1.5	—	V	禁止BOR或LPBOR(3)
确保内部上电复位信号的VDD上升速率(2)							
D006	SVDD		0.05	—	—	V/ms	禁止BOR或LPBOR(3)
D006	SVDD		0.05	—	—	V/ms	禁止BOR或LPBOR(3)

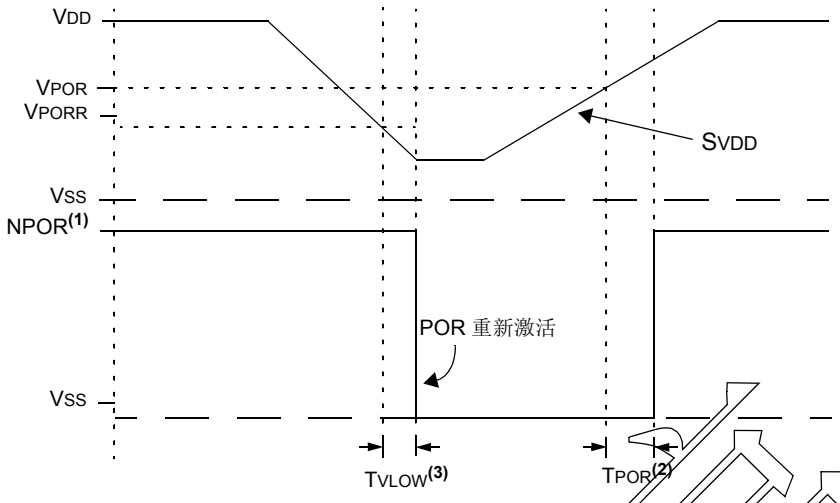
† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 这是在不丢失RAM数据的前提下，休眠模式下VDD的下限值。

2： 请参见图45-3，可以看到VDD缓慢上升时，POR和POR重新激活。

3： 有关BOR和LPBOR跳变点的信息，请参见表45-11。

图45-3： VDD缓慢上升时，POR和POR重新激活



注 1： 当NPOR为低电平时，器件保持在复位状态。
2： TPOR典型值为1 µs。
3： TVLOW典型值为2.7 µs。

表45-2： 供电电流（IDD）(1,2,4)

PIC18LF25/26K83			标准工作条件（除非另外声明）					
PIC18F25/26K83								
参数编号	**符号**	**器件特性**	**最小值**	**典型值†**	**最大值**	**单位**	**条件**	
							VDD	**注**
D100	$I_{DD_{XT4}}$	XT = 4 MHz	—	590	1200	μA	3.0V	
D100	$I_{DD_{XT4}}$	XT = 4 MHz	—	770	1300	μA	3.0V	
D100A	$I_{DD_{XT4}}$	XT = 4 MHz	—	390	850	μA	3.0V	PMD为全1
D100A	$I_{DD_{XT4}}$	XT = 4 MHz	—	620	950	μA	3.0V	PMD为全1
D101	$I_{DD_{HFO16}}$	HFINTOSC = 16 MHz	—	2.3	5.0	mA	3.0V	
D101	$I_{DD_{HFO16}}$	HFINTOSC = 16 MHz	—	2.4	5.1	mA	3.0V	
D101A	$I_{DD_{HFO16}}$	HFINTOSC = 16 MHz	—	1.5	3.2	mA	3.0V	PMD为全1
D101A	$I_{DD_{HFO16}}$	HFINTOSC = 16 MHz	—	1.5	3.5	mA	3.0V	PMD为全1
D102	$I_{DD_{HFOPLL}}$	HFINTOSC = 64 MHz	—	8.4	18.5	mA	3.0V	
D102	$I_{DD_{HFOPLL}}$	HFINTOSC = 64 MHz	—	8.4	19	mA	3.0V	
D102A	$I_{DD_{HFOPLL}}$	HFINTOSC = 64 MHz	—	5	11	mA	3.0V	PMD为全1
D102A	$I_{DD_{HFOPLL}}$	HFINTOSC = 64 MHz	—	5	11.5	mA	3.0V	PMD为全1
D103	$I_{DD_{HSPLL64}}$	HS+PLL = 64 MHz	—	8.4	19	mA	3.0V	
D103	$I_{DD_{HSPLL64}}$	HS+PLL = 64 MHz	—	8.4	20	mA	3.0V	
D103A	$I_{DD_{HSPLL64}}$	HS+PLL = 64 MHz	—	5	10	mA	3.0V	PMD为全1
D103A	$I_{DD_{HSPLL64}}$	HS+PLL = 64 MHz	—	5	10	mA	3.0V	PMD为全1
D104	$I_{DD_{IDLE}}$	空闲模式，HFINTOSC = 16 MHz	—	1.9	4.5	mA	3.0V	
D104	$I_{DD_{IDLE}}$	空闲模式，HFINTOSC = 16 MHz	—	1.9	4.5	mA	3.0V	
D105	$I_{DD_{DOZE}}$(3)	打盹模式，HFINTOSC = 16 MHz， 打盹模式时钟分频比 = 16	—	1.9	—	mA	3.0V	
D105	$I_{DD_{DOZE}}$(3)	打盹模式，HFINTOSC = 16 MHz， 打盹模式时钟分频比 = 16	—	1.9	—	mA	3.0V	

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 有效工作模式下，所有IDD测量值的测试条件为：OSC1 = 外部方波，轨到轨满幅；所有I/O引脚均为驱动为低电平的输出；$\overline{MCLR}$ = VDD；禁止WDT。

2： 供电电流主要受工作电压和频率的影响。其他因素，如I/O引脚负载和开关速率、振荡器类型、内部代码执行模式和温度也会对电流消耗产生影响。

3： $I_{DD_{DOZE}} = [I_{DD_{IDLE}}*(N-1)/N] + I_{DD_{HFO}}16/N$，其中N = 打盹模式时钟分频比（寄存器10-2）。

4： PMD位均处于默认状态，不禁止任何模块。

表45-3： 掉电电流（IPD）(1,2)

PIC18LF25/26K83				标准工作条件（除非另外声明）					
PIC18F25/26K83				标准工作条件（除非另外声明） VREGPM = 1					
参数编号	符号	器件特性	最小值	典型值†	最大值 +85°C	最大值 +125°C	单位	条件	
								VDD	注
D200	IPD	基本IPD电流	—	0.07	2	6	μA	3.0V	
D200 D200A	IPD	基本IPD电流	—	0.4	2.5	8	μA	3.0V	
			—	20	37	45	μA	3.0V	VREGPM = 0
D201	IPD_WDT	低频内部振荡器/WDT	—	0.9	2.9	9	μA	3.0V	
D201	IPD_WDT	低频内部振荡器/WDT	—	1.1	3.3	9	μA	3.0V	
D202	IPD_SOSC	辅助振荡器（SOSC）	—	0.6	2.8	13	μA	3.0V	LP模式
D202	IPD_SOSC	辅助振荡器（SOSC）	—	0.8	3.2	15	μA	3.0V	LP模式
D203	IPD_FVR	FVR	—	37	70	75	μA	3.0V	FVRCON = 0x81或0x84
D203	IPD_FVR	FVR	—	30	70	76	μA	3.0V	FVRCON = 0x81或0x84
D204	IPD_BOR	欠压复位（BOR）	—	9.4	16	18	μA	3.0V	
D204	IPD_BOR	欠压复位（BOR）	—	9.4	17	19	μA	3.0V	
D205	IPD_LPBOR	低功耗欠压复位（LPBOR）	—	0.2	3	6	μA	3.0V	
D205	IPD_LPBOR	低功耗欠压复位（LPBOR）	—	0.5	3	5	μA	3.0V	
D206	IPD_HLVD	高/低电压检测（HLVD）	—	9.5	16	19	μA	3.0V	
D206	IPD_HLVD	高/低电压检测（HLVD）	—	9.7	17	20	μA	3.0V	
D207	IPD_ADCA	ADC——工作	—	400	—	—	μA	3.0V	ADC正在进行转换(4)
D207	IPD_ADCA	ADC——工作	—	400	—	—	μA	3.0V	ADC正在进行转换(4)
D208	IPD_CMP	比较器	—	33	50	55	μA	3.0V	
D208	IPD_CMP	比较器	—	30	50	60	μA	3.0V	

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 外设电流为基本IDD与该外设使能时所额外消耗的电流之和。可通过从该参数值中减去基本IDD或IPD电流，以确定外设Δ电流。在计算总电流消耗时应使用最大值。

2： 在休眠模式下，掉电电流与振荡器类型无关。掉电电流是在器件处于休眠模式、所有I/O引脚处于高阻态并且连接到VSS时测得的。

3： 如果多个外设可用，那么列出的所有外设电流均是基于每个外设的。

4： ADC时钟源为FRC。

表45-4： I/O端口

标准工作条件（除非另外声明）

参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
	VIL	**输入低电压**					
		I/O端口：					
D300		带TTL缓冲器	—	—	0.8	V	4.5V ≤ VDD ≤ 5.5V
D301			—	—	0.15 VDD	V	1.8V ≤ VDD ≤ 4.5V
D302		带施密特触发器缓冲器	—	—	0.2 VDD	V	2.0V ≤ VDD ≤ 5.5V
D303		I^2C电平	—	—	0.3 VDD	V	
D304		SMBus 2.0	—	—	0.8	V	2.7V ≤ VDD ≤ 5.5V
D305		SMBus 3.0	—	—	0.8	V	1.8V ≤ VDD ≤ 5.5V
D306		$\overline{MCLR}$	—	—	0.2 VDD	V	
	VIH	**输入高电压**					
		I/O端口：					
D320		带TTL缓冲器	2.0	—	—	V	4.5V ≤ VDD ≤ 5.5V
D321			0.25 VDD + 0.8	—	—	V	1.8V ≤ VDD ≤ 4.5V
D322		带施密特触发器缓冲器	0.8 VDD	—	—	V	2.0V ≤ VDD ≤ 5.5V
D323		I^2C电平	0.7 VDD	—	—	V	
D324		SMBus 2.0	2.1	—	—	V	2.7V ≤ VDD ≤ 5.5V
D325		SMBus 3.0	1.35	—	—	V	1.8V ≤ VDD ≤ 5.5V
D326		$\overline{MCLR}$	0.7 VDD	—	—	V	
	IIL	**输入泄漏电流(1)**					
D340		I/O端口	—	± 5	± 125	nA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态，85°C
D341			—	± 5	± 1000	nA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态，125°C
D342		$\overline{MCLR}$(2)	—	± 50	± 200	nA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态，85°C
	IPUR	**弱上拉电流**					
D350			25	120	200	µA	VDD = 3.0V，VPIN = VSS
	VOL	**输出低电压**					
D360		I/O端口	—	—	0.6	V	IOL = 10.0 mA，VDD = 3.0V
	VOH	**输出高电压**					
D370		I/O端口	VDD - 0.7	—	—	V	IOH = 6.0 mA，VDD = 3.0V
D380	CIO	**所有I/O引脚**	—	5	50	pF	

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 负电流定义为引脚的拉电流。

2： $\overline{MCLR}$ 引脚上的泄漏电流主要取决于所施加的电压。规定电压为正常工作条件下的电压。在不同的输入电压下可测得更高的泄漏电流。

表45-5： 存储器编程规范

标准工作条件（除非另外声明）							
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
数据EEPROM存储器规范							
MEM20	E_D	数据EEPROM字节耐擦写能力	100k	—	—	E/W	−40°C ≤ TA ≤ +85°C
MEM21	T_{D_RET}	特性保持时间	—	40	—	年	假设没有违反其他规范
MEM22	N_{D_REF}	刷新前的总擦除/写周期	1M 500k	10M —	— —	E/W	−40°C ≤ TA ≤ +60°C −40°C ≤ TA ≤ +85°C
MEM23	V_{D_RW}	用于读或擦除/写操作的VDD	VDDMIN	—	VDDMAX	V	
MEM24	T_{D_BEW}	字节擦除和写周期时间	—	4.0	5.0	ms	
闪存程序存储器规范							
MEM30	E_P	存储器单元耐擦写能力	10k	—	—	E/W	−40°C ≤ TA ≤ +85°C（**注1**）
MEM32	T_{P_RET}	特性保持时间	—	40	—	年	假设没有违反其他规范
MEM33	V_{P_RD}	用于读操作的VDD	VDDMIN	—	VDDMAX	V	
MEM34	V_{P_REW}	用于行擦除/写操作的VDD	VDDMIN	—	VDDMAX	V	
MEM35	T_{P_REW}	自定时行擦除或自定时写周期	—	2.0	2.5	ms	

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 程序存储器单元耐擦写能力定义为：一个行擦除操作和一个自定时写操作。

表45-6： 温度特性

标准工作条件（除非另外声明）

参数编号	符号	特性	典型值	单位	条件
TH01	θJA	热阻（结到环境）	60	°C/W	28引脚SPDIP封装
			80	°C/W	28引脚SOIC封装
			90	°C/W	28引脚SSOP封装
			27.5	°C/W	28引脚UQFN 4x4 mm封装
			27.5	°C/W	28引脚QFN 6x6 mm封装
TH02	θJC	热阻（结到管壳）	31.4	°C/W	28引脚SPDIP封装
			24	°C/W	28引脚SOIC封装
			24	°C/W	28引脚SSOP封装
			24	°C/W	28引脚UQFN 4x4mm封装
			24	°C/W	28引脚QFN 6x6 mm封装
TH03	TJMAX	最高结温	150	°C	
TH04	PD	功耗	—	W	PD = PINTERNAL + PI/O(3)
TH05	PINTERNAL	内部功耗	—	W	PINTERNAL = IDD x VDD(1)
TH06	PI/O	I/O功耗	—	W	PI/O = Σ (IOL * VOL) + Σ (IOH * (VDD - VOH))
TH07	PDER	降额功耗	—	W	PDER = PDMAX (TJ - TA)/θJA(2)

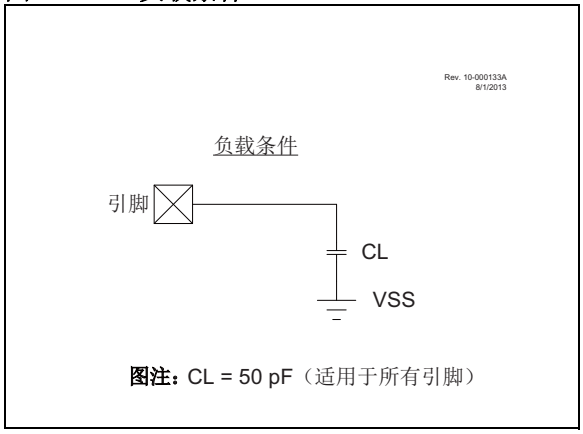
注 1： IDD为输出引脚上不驱动任何负载时使芯片独立运行的电流。

2： TA= 环境温度，TJ = 结温

3： 有关总功耗，请参见绝对最大值。

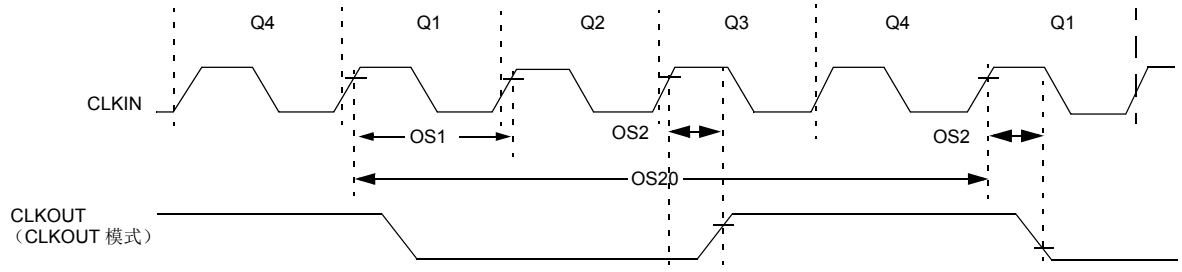
45.4 交流特性

图45-4： 负载条件



图注：CL = 50 pF（适用于所有引脚）

图45-5： 时钟时序



注 1： 请参见表45-7。

表45-7： 外部时钟/振荡器时序要求

标准工作条件（除非另外声明）

参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
ECL振荡器							
OS1	F_{ECL}	时钟频率	—	—	500	kHz	
OS2	T_{ECL_DC}	时钟占空比	40	—	60	%	
ECM振荡器							
OS3	F_{ECM}	时钟频率	—	—	4	MHz	
OS4	T_{ECM_DC}	时钟占空比	40	—	60	%	
ECH振荡器							
OS5	F_{ECH}	时钟频率	—	—	32	MHz	
OS6	T_{ECH_DC}	时钟占空比	40	—	60	%	
LP振荡器							
OS7	F_{LP}	时钟频率	—	—	100	kHz	**注4**
XT振荡器							
OS8	F_{XT}	时钟频率	—	—	4	MHz	**注4**
HS振荡器							
OS9	F_{HS}	时钟频率	—	—	20	MHz	**注4**
辅助振荡器							
OS10	F_{SEC}	时钟频率	32.4	32.768	33.1	kHz	
系统振荡器							
OS20	F_{OSC}	系统时钟频率	—	—	64	MHz	**（注2和注3）**

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 指令周期（TCY）等于输入振荡器时基周期的四倍。所有规定值均为基于针对特定振荡器类型，器件在标准工作条件下执行代码时的特性数据。超出这些规定的限定值，可能导致振荡器运行不稳定和/或导致电流消耗超出预期值。所有器件在测试“最小”值时，都在OSC1引脚连接了外部时钟。当使用了外部时钟输入时，所有器件的“最大”周期时间限制为“DC”（无时钟）。

2： 系统时钟频率（FOSC）通过“主时钟切换控制”选择，如**第10.0节“节能工作模式”**所述。

3： 系统时钟频率（FOSC）必须满足**第45.2节“标准工作条件”**中定义的电压要求。

4： LP、XT 和 HS 振荡器模式要求将一个适当的晶振或谐振器连接到器件。当通过外部方波为器件提供时钟时，必须使用其中一个EC模式选择。

表45-7： 外部时钟/振荡器时序要求 （续）

标准工作条件（除非另外声明）							
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
OS21	F_{CY}	指令频率	—	FOSC/4	—	MHz	
OS22	T_{CY}	指令周期	62.5	$1/F_{CY}$	—	ns	

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 指令周期（TCY）等于输入振荡器时基周期的四倍。所有规定值均为基于针对特定振荡器类型，器件在标准工作条件下执行代码时的特性数据。超出这些规定的限定值，可能导致振荡器运行不稳定和/或导致电流消耗超出预期值。所有器件在测试“最小”值时，都在OSC1引脚连接了外部时钟。当使用了外部时钟输入时，所有器件的“最大”周期时间限制为“DC”（无时钟）。

2： 系统时钟频率（FOSC）通过“主时钟切换控制”选择，如**第10.0节“节能工作模式”**所述。

3： 系统时钟频率（FOSC）必须满足**第45.2节“标准工作条件”**中定义的电压要求。

4： LP、XT 和 HS 振荡器模式要求将一个适当的晶振或谐振器连接到器件。当通过外部方波为器件提供时钟时，必须使用其中一个EC模式选择。

表45-8： 内部振荡器参数(1)

标准工作条件（除非另外声明）							
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
OS50	FHFOSC	已校准精度的HFINTOSC频率	—	4 8 12 16 48 64	—	MHz	（注2）
OS51	FHFOSCLP	针对低功耗优化的HFINTOSC频率	0.93 1.86	1 2	1.07 2.14	MHz MHz	
OS53*	FLFOSC	内部LFINTOSC频率	—	31	—	kHz	
OS54*	THFOSCST	HFINTOSC从休眠模式唤醒的起振时间	— —	11 50	20 —	μs μs	VREGPM = 0 VREGPM = 1
OS56	TLFOSCST	LFINTOSC从休眠模式唤醒的起振时间	—	0.2	—	ms	

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 为了确保振荡器频率容差，必须尽可能靠近器件、在VDD和VSS之间接去耦电容。建议并联0.1 μF和0.01 μF的电容。

2： 请参见图45-6：器件VDD和温度范围内的已校准精度的HFINTOSC频率的精度。

图45-6： 器件VDD和温度范围内的已校准精度的HFINTOSC频率的精度

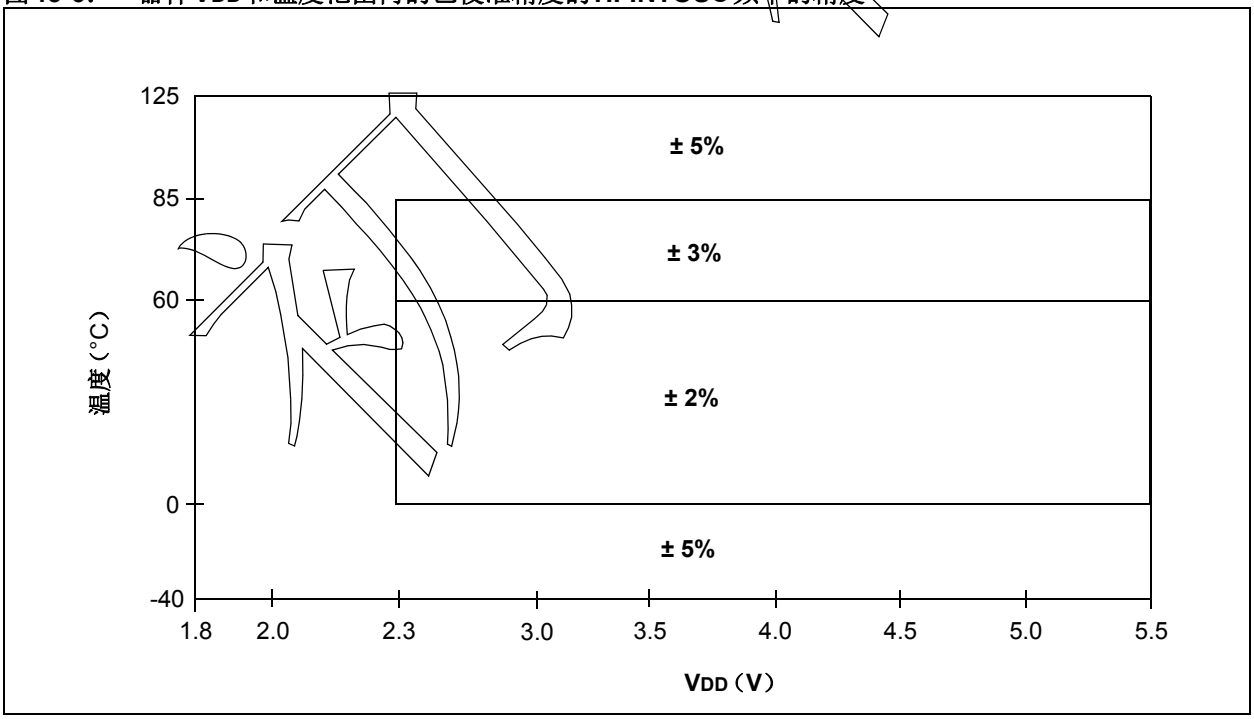


表45-9： PLL规范

标准工作条件（除非另外声明）$V_{DD} \geq 2.5V$							
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
PLL01	FPLLIN	PLL输入频率范围	4	—	16	MHz	
PLL02	FPLLOUT	PLL输出频率范围	16	—	64	MHz	注1
PLL03	TPLLST	PLL自起振的锁定时间	—	200	—	μs	
PLL04	FPLLJIT	PLL输出频率稳定性（抗抖动性）	-0.25	—	0.25	%	

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为5V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： PLL的输出频率必须满足参数D002中列出的FOSC要求。

图45-7： CLKOUT和I/O时序

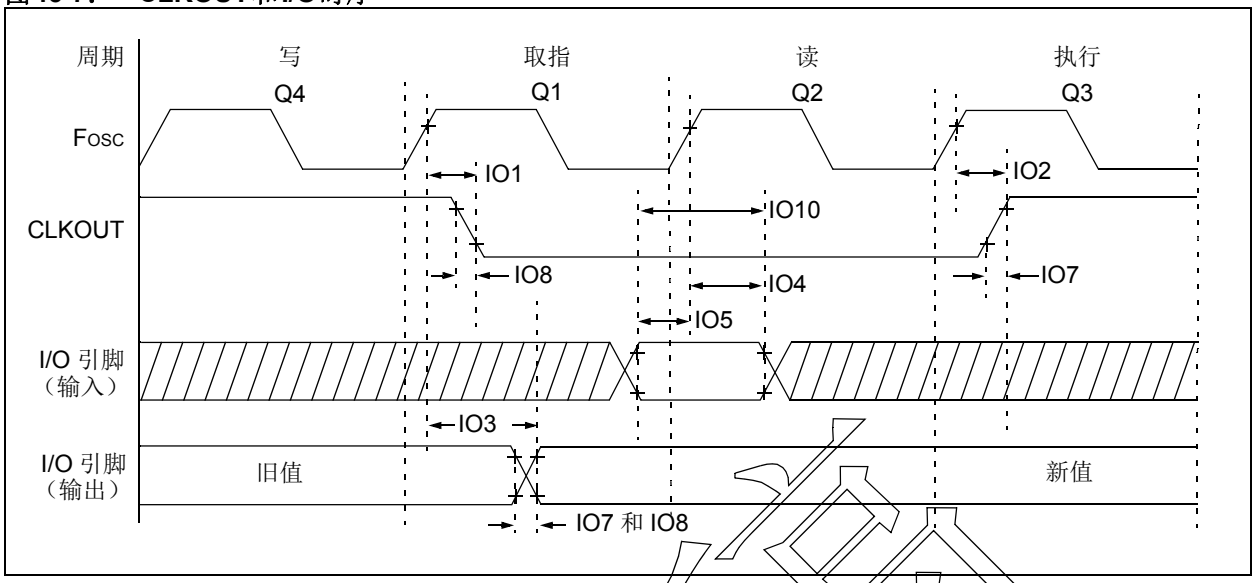


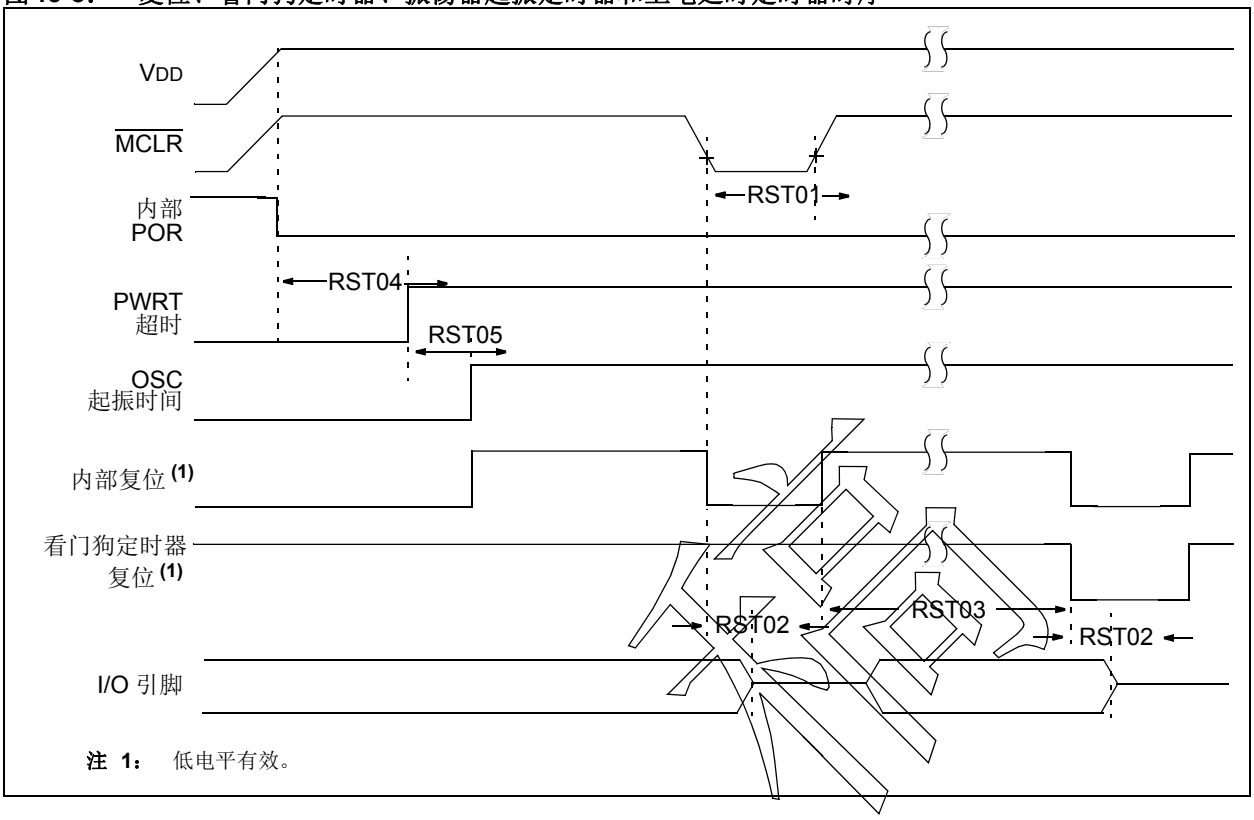
表45-10： I/O和CLKOUT时序规范

标准工作条件 （除非另外声明）

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
IO1*	$T_{CLKOUTH}$	CLKOUT上升沿延时（Fosc上升沿（Q1周期）到CLKOUT下降沿的时间）	—	—	70	ns	
IO2*	$T_{CLKOUTL}$	CLKOUT下降沿延时（Fosc上升沿（Q3周期）到CLKOUT上升沿的时间）	—	—	72	ns	
IO3*	T_{IO_VALID}	端口输出有效时间（Fosc上升沿（Q1周期）到端口有效的时间）	—	50	70	ns	
IO4*	T_{IO_SETUP}	端口输入建立时间（Fosc上升沿（Q2周期）之前的建立时间）	20	—	—	ns	
IO5*	T_{IO_HOLD}	端口输入保持时间（Fosc上升沿（Q2周期）之后的保持时间）	50	—	—	ns	
IO6*	T_{IOR_SLREN}	端口I/O上升时间，使能压摆率	—	25	—	ns	V_{DD} = 3.0V
IO7*	T_{IOR_SLRDIS}	端口I/O上升时间，禁止压摆率	—	5	—	ns	V_{DD} = 3.0V
IO8*	T_{IOF_SLREN}	端口I/O下降时间，使能压摆率	—	25	—	ns	V_{DD} = 3.0V
IO9*	T_{IOF_SLRDIS}	端口I/O下降时间，禁止压摆率	—	5	—	ns	V_{DD} = 3.0V
IO10*	T_{INT}	INT引脚触发中断的高电平时间或低电平时间	25	—	—	ns	
IO11*	T_{IOC}	电平变化中断触发中断的最短高电平或低电平时间	25	—	—	ns	

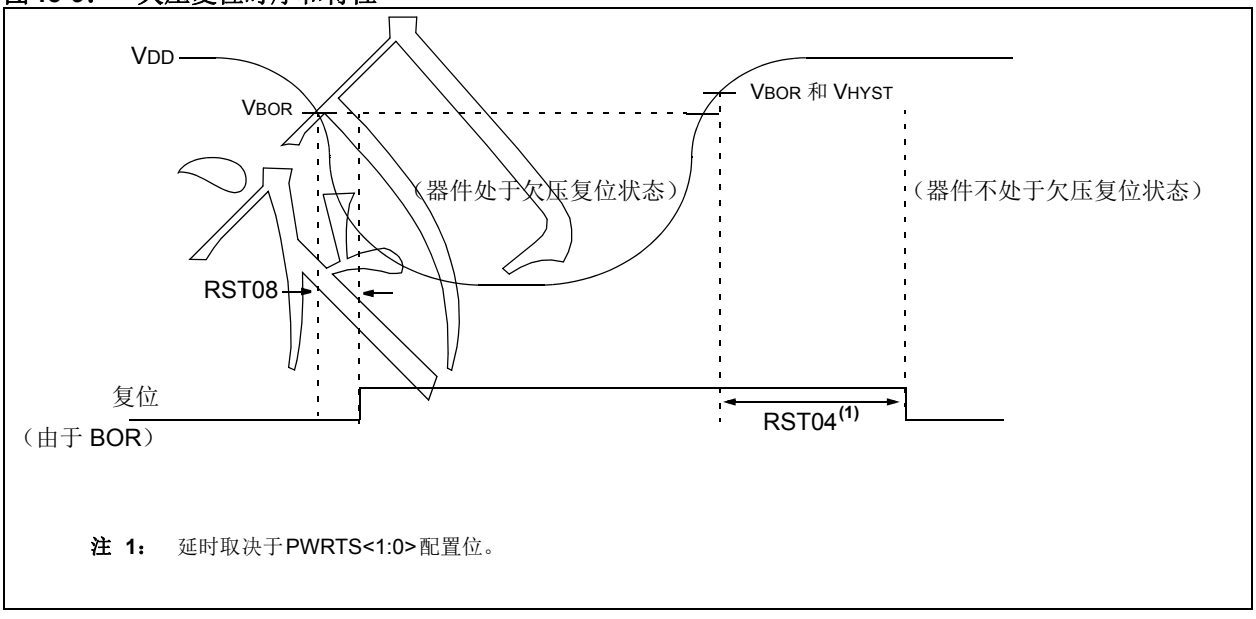
* 这些参数为特性值，但未经测试。

图45-8： 复位、看门狗定时器、振荡器起振定时器和上电延时定时器时序



注 1： 低电平有效。

图45-9： 欠压复位时序和特性



注 1： 延时取决于PWRTS<1:0>配置位。

表45-11： 复位、WDT、振荡器起振定时器、上电定时器、欠压复位和低功耗欠压复位规范

标准工作条件（除非另外声明）							
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
RST01*	TMCLR	确保复位的$\overline{\text{MCLR}}$脉冲宽度（低电平）	2	—	—	µs	
RST02*	TIOZ	自检测到复位起I/O处于高阻态的时间	—	—	2	µs	
RST03	TWDT	看门狗定时器超时周期	—	16	—	ms	1:512预分频比
RST04*	TPWRT	上电延时定时器周期	— — —	1 16 64	— — —	ms ms ms	PWRTS = 00 PWRTS = 01 PWRTS = 10
RST05	TOST	振荡器起振定时器周期(1,2)	—	1024	—	TOSC	
RST06	VBOR	欠压复位电压(4)	2.7 2.55 2.3 2.3 1.8	2.85 2.7 2.45 2.45 1.9	3.0 2.85 2.6 2.6 2.05	V V V V V	BORV = 00 BORV = 01 BORV = 10 BORV = 11（PIC18Fxxx） BORV = 11（PIC18LFxxx）
RST07	VBORHYS	欠压复位滞后电压	—	40	—	mV	
RST08	TBORDC	欠压复位响应时间	—	3	—	µs	
RST09	VLPBOR	低功耗欠压复位电压	1.8	1.9	2.2	V	

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 根据设计，振荡器起振定时器（OST）在提供系统频率之前首先计数1024个周期。

2： 为了确保这些电压容差，必须尽可能靠近器件、在VDD和VSS之间接去耦电容。建议并联0.1 µF和0.01 µF的电容。

表45-12： 高/低压检测特性

标准工作条件（除非另外声明）							
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
HLVD01	V_{DET}	电压检测	—	1.90	—	V	HLVDSEL<3:0>=0000
			—	2.10	—	V	HLVDSEL<3:0>=0001
			—	2.25	—	V	HLVDSEL<3:0>=0010
			—	2.50	—	V	HLVDSEL<3:0>=0011
			—	2.60	—	V	HLVDSEL<3:0>=0100
			—	2.75	—	V	HLVDSEL<3:0>=0101
			—	2.90	—	V	HLVDSEL<3:0>=0110
			—	3.15	—	V	HLVDSEL<3:0>=0111
			—	3.35	—	V	HLVDSEL<3:0>=1000
			—	3.60	—	V	HLVDSEL<3:0>=1001
			—	3.75	—	V	HLVDSEL<3:0>=1010
			—	4.00	—	V	HLVDSEL<3:0>=1011
			—	4.20	—	V	HLVDSEL<3:0>=1100
			—	4.35	—	V	HLVDSEL<3:0>=1101
			—	4.65	—	V	HLVDSEL<3:0>=1110

表45-13： 模数转换器（ADC）精度规范(1,2)：

工作条件（**除非另外声明**） VDD = 3.0V，TA = 25°C，TAD = 1 µs							
参数编号	**符号**	**特性**	**最小值**	**典型值†**	**最大值**	**单位**	**条件**
AD01	NR	分辨率	—	—	12	bit	
AD02	EIL	积分误差	—	±0.1	±2.0	LSb	ADCREF+ = 3.0V，ADCREF-= 0V
AD03	EDL	微分误差	—	±0.1	±1.0	LSb	ADCREF+ = 3.0V，ADCREF-= 0V
AD04	EOFF	失调误差	—	0.5	6.0	LSb	ADCREF+ = 3.0V，ADCREF-= 0V
AD05	EGN	增益误差	—	±0.2	±6.0	LSb	ADCREF+ = 3.0V，ADCREF-= 0V
AD06	VADREF	ADC参考电压（ADREF+ - ADREF-）	1.8	—	VDD	V	
AD07	VAIN	满量程范围	ADREF-	—	ADREF+	V	
AD08	ZAIN	模拟信号源的推荐阻抗	—	10	—	kΩ	
AD09	RVREF	ADC参考电压梯形阻抗	—	50	—	kΩ	**注3**

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 总绝对误差是失调误差、增益误差和积分非线性（INL）误差的总和。

2： ADC转换结果不会因输入的增加而减小，并且不会丢失编码。

3： 这是选择外部参考焊盘时VREF焊盘的阻抗。

表45-14： 模数转换器（ADC）转换时序规范

标准工作条件（除非另外声明）							
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
AD20	TAD	ADC时钟周期	1	—	9	μs	使用FOSC作为ADC时钟源，ADOCS = 0
AD21			—	2	—	μs	使用FRC作为ADC时钟源，ADOCS = 1
AD22	TCNV	转换时间(1)	—	13/+3 TAD	—	TAD	GO/$\overline{\text{DONE}}$位置1到GO/$\overline{\text{DONE}}$位清零的时间
AD23	TACQ	采集时间	—	2	—	μs	
AD24	THCD	采样保持电容断开时间	3	—	—	TAD	基于FOSC的时钟源 基于FRC的时钟源

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 不适用于ADCRC振荡器。

图45-10： ADC转换时序（ADC时钟基于Fosc）

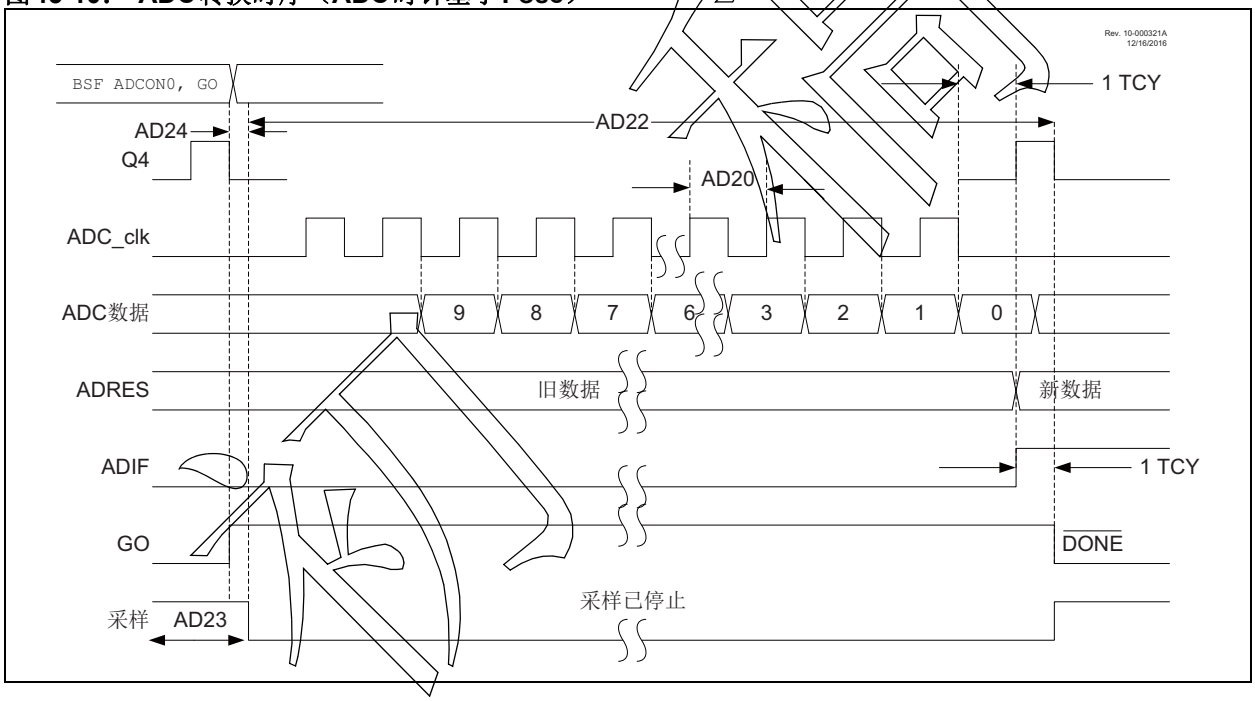
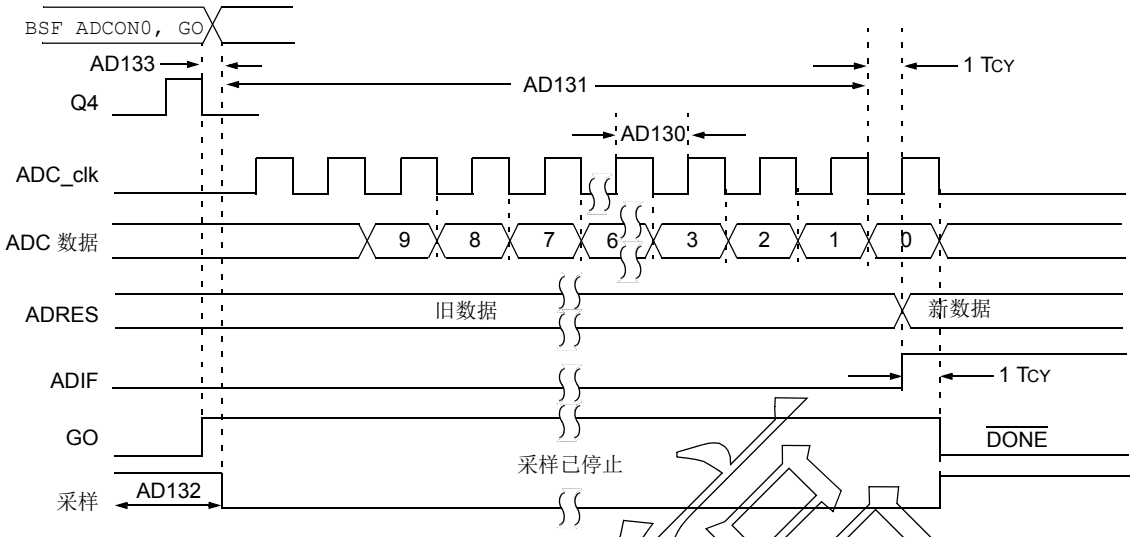


图45-11： ADC转换时序（ADC时钟来自ADCRC）



注 1： 如果选择ADCRC作为ADC时钟源，在ADC时钟启动前要加上一个TCY时间，用以执行SLEEP指令。

表45-15： 比较器规范

工作条件（除非另外声明） VDD = 3.0V，TA = 25°C							
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**备注**
CM01	VIOFF	输入失调电压	—	—	±40	mV	VICM = VDD/2
CM02	VICM	输入共模范围	GND	—	VDD	V	
CM03	CMRR	共模输入抑制比	—	50	—	dB	
CM04	VHYST	比较器滞后	10	25	40	mV	
CM05	TRESP(1)	响应时间上升沿	—	300	600	ns	
		响应时间下降沿	—	220	500	ns	

* 这些参数为特性值，但未经测试。

注 1： 响应时间是在比较器的一个输入端电压为VDD/2，而另一个输入端从VSS跳变到VDD时测得的。

2： 模式改变包括改变任一控制寄存器值（包括模块使能）。

表45-16： 5位DAC规范

标准工作条件（除非另外声明） VDD = 3.0V，TA = 25°C							
参数编号	**符号**	**特性**	**最小值**	**典型值†**	**最大值**	**单位**	**备注**
DSB01	VLSB	步长	—	(VDACREF+ - VDACREF-) / 32	—	V	
DSB01	VACC	绝对精度	—	—	± 0.5	LSb	
DSB03*	RUNIT	单位电阻值	—	5000	—	Ω	
DSB04*	TST	稳定时间(1)	—	—	10	µs	

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 稳定时间是在DACR<4:0>从00000跳变到01111时测得的。

表45-17： 固定参考电压（FVR）规范

标准工作条件（除非另外声明）							
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**条件**
FVR01	VFVR1	1x增益（1.024V）	-4	—	+4	%	VDD ≥ 2.5V，-40°C至85°C
FVR02	VFVR2	2x增益（2.048V）	-4	—	+4	%	VDD ≥ 2.5V，-40°C至85°C
FVR03	VFVR4	4x增益（4.096V）	-5	—	+5	%	VDD ≥ 4.75V，-40°C至85°C
FVR04	TFVRST	FVR启动时间	—	25	—	µs	

表45-18： 过零检测（ZCD）规范

标准工作条件（除非另外声明） VDD = 3.0V，TA = 25°C							
参数编号	**符号**	**特性**	**最小值**	**典型值†**	**最大值**	**单位**	**备注**
ZC01	VPINZC	过零引脚上的电压	—	0.75	—	V	
ZC02	IZCD_MAX	最大拉电流或灌电流	—	—	600	µA	
ZC03	TRESPH	响应时间上升沿	—	1	—	µs	
	TRESPL	响应时间下降沿	—	1	—	µs	

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

图45-12： TIMER0和TIMER1外部时钟时序

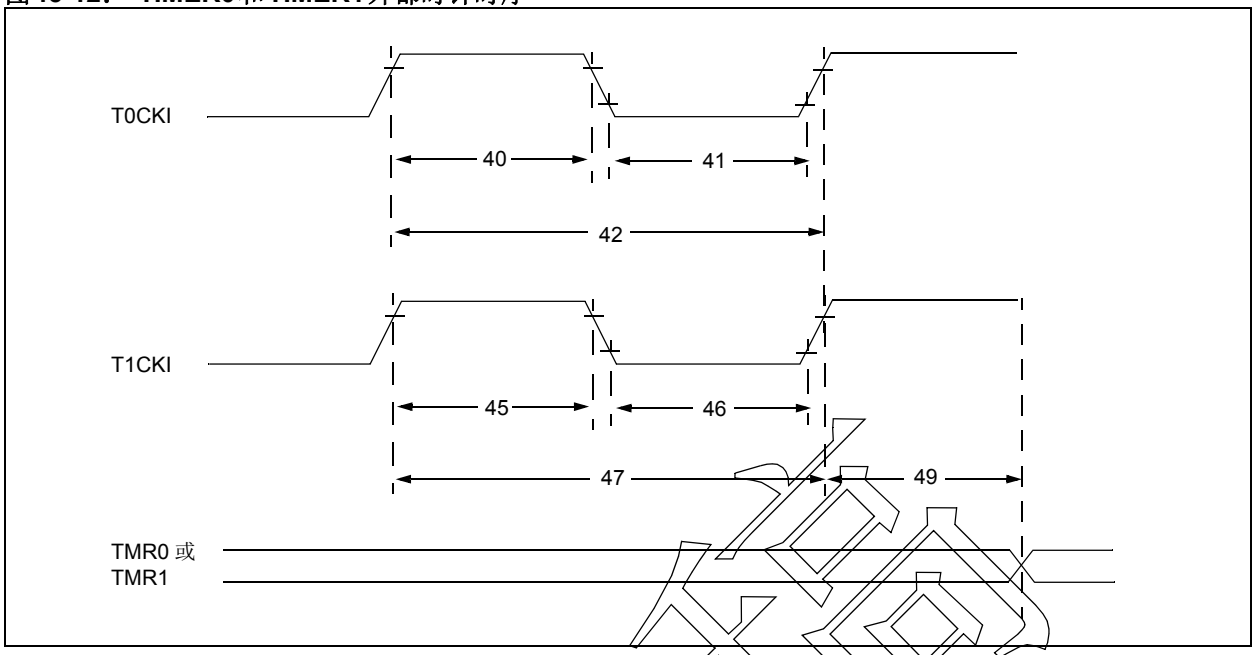


表45-19： TIMER0和TIMER1外部时钟要求

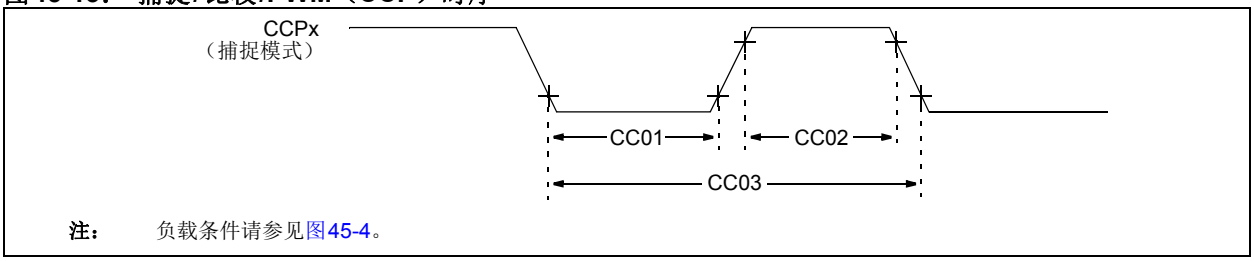
标准工作条件（除非另外声明）
工作温度 -40°C ≤ TA ≤ +125°C

参数编号	符号	特性		最小值	典型值†	最大值	单位	条件
40*	TT0H	T0CKI高电平脉冲宽度	无预分频器	0.5 TCY + 20	—	—	ns	
			带预分频器	10	—	—	ns	
41*	TT0L	T0CKI低电平脉冲宽度	无预分频器	0.5 TCY + 20	—	—	ns	
			带预分频器	10	—	—	ns	
42*	TT0P	T0CKI周期		取如下二者中较大值： 20或$\frac{T_{CY}+40}{N}$	—	—	ns	N = 预分频值
45*	TT1H	T1CKI高电平时间	同步，无预分频器	0.5 TCY + 20	—	—	ns	
			同步，带预分频器	15	—	—	ns	
			异步	30	—	—	ns	
46*	TT1L	T1CKI低电平时间	同步，无预分频器	0.5 TCY + 20	—	—	ns	
			同步，带预分频器	15	—	—	ns	
			异步	30	—	—	ns	
47*	TT1P	T1CKI输入周期	同步	取如下二者中较大值： 30或$\frac{T_{CY}+40}{N}$	—	—	ns	N = 预分频值
			异步	60	—	—	ns	
49*	TCKEZTMR1	从外部时钟边沿到定时器递增的延时		2 TOSC	—	7 TOSC	—	同步模式下的定时器

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

图45-13： 捕捉/比较/PWM（CCP）时序



注： 负载条件请参见图45-4。

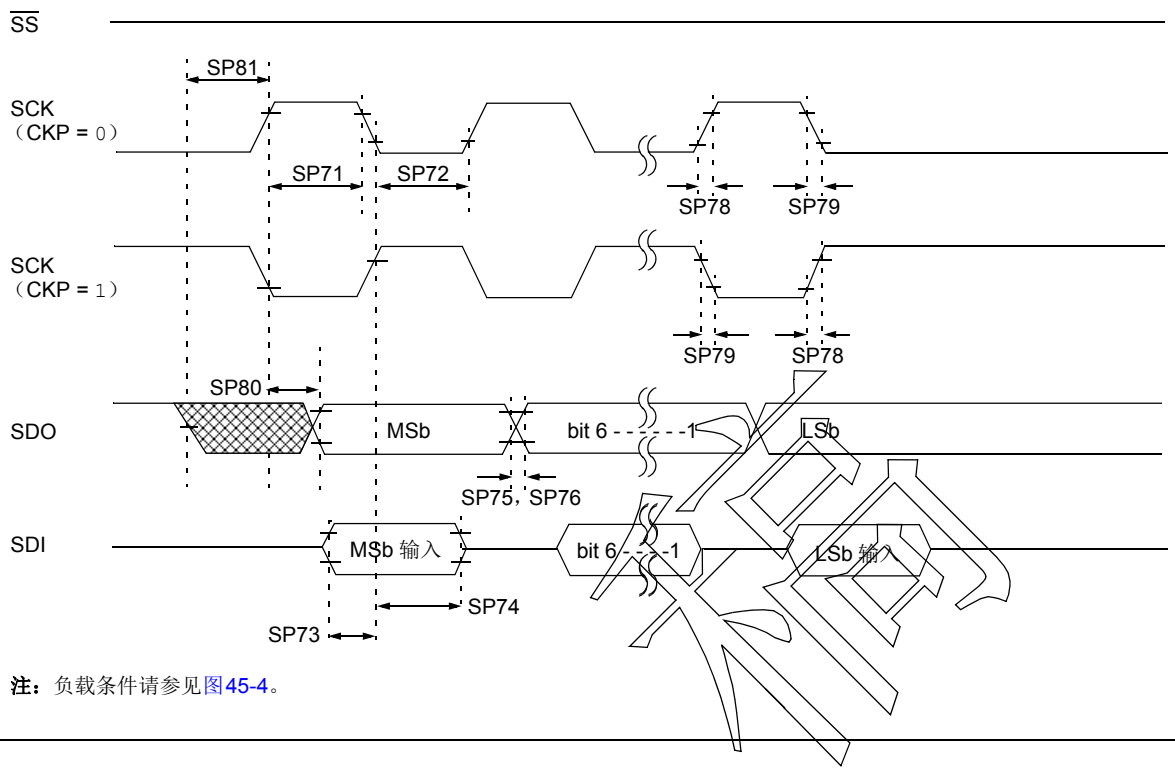
表45-20： 捕捉/比较/PWM要求（CCP）

标准工作条件（除非另外声明） 工作温度 -40°C ≤ TA ≤ +125°C								
参数编号	**符号**	**特性**		**最小值**	**典型值†**	**最大值**	**单位**	**条件**
CC01*	TccL	CCPx输入低电平时间	无预分频器	0.5TCY + 20	—	—	ns	
			带预分频器	20	—	—	ns	
CC02*	TccH	CCPx输入高电平时间	无预分频器	0.5TCY + 20	—	—	ns	
			带预分频器	20	—	—	ns	
CC03*	TccP	CCPx输入周期		$\frac{3T_{CY} + 40}{N}$	—	—	ns	N = 预分频值

* 这些参数为特性值，但未经测试。

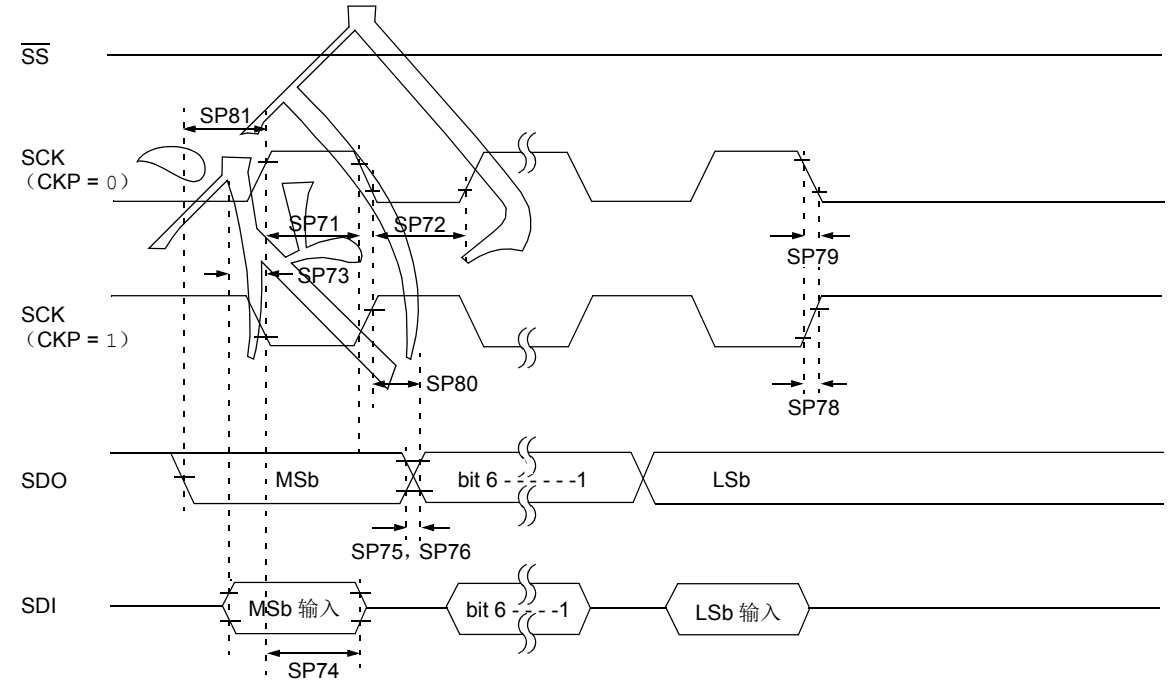
† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

图45-14： SPI主模式时序（CKE = 0，SMP = 0）



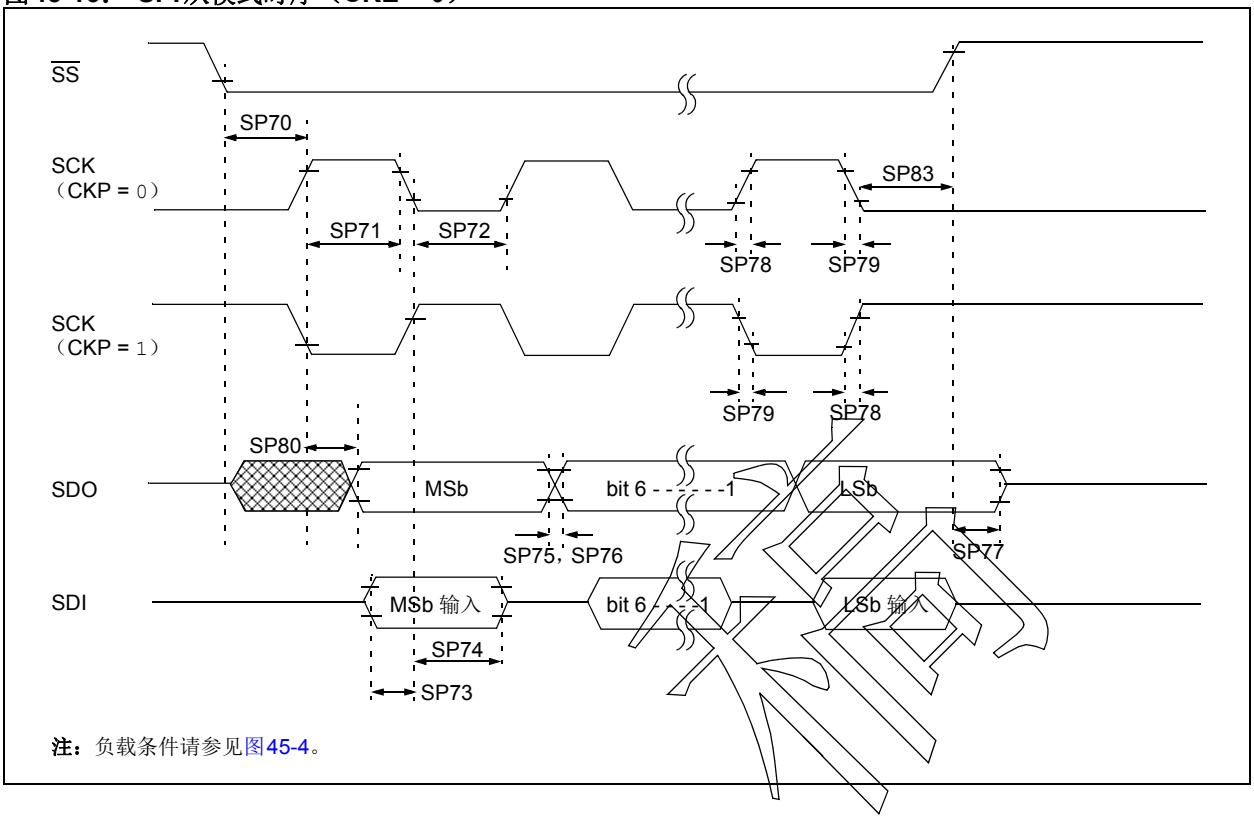
注： 负载条件请参见图45-4。

图45-15： SPI主模式时序（CKE = 1，SMP = 1）



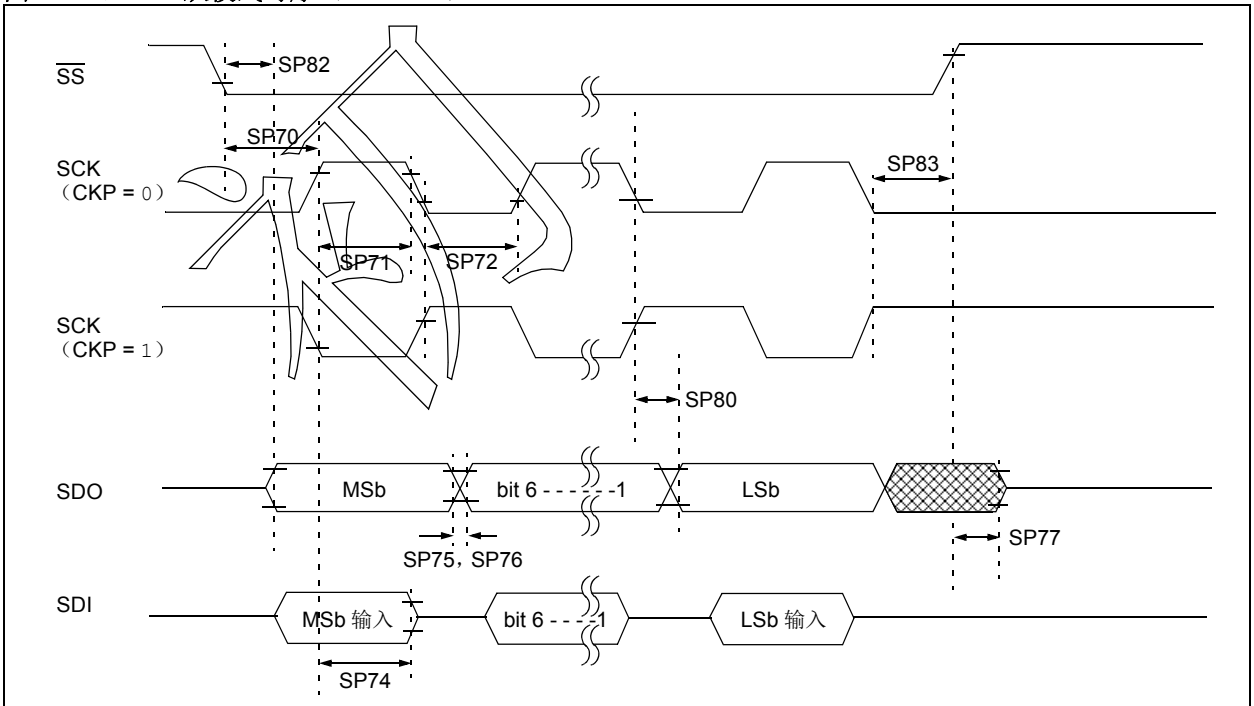
注： 负载条件请参见图45-4。

图45-16： SPI从模式时序（CKE = 0）



注： 负载条件请参见图45-4。

图45-17： SPI从模式时序（CKE = 1）



注： 负载条件请参见图45-4。

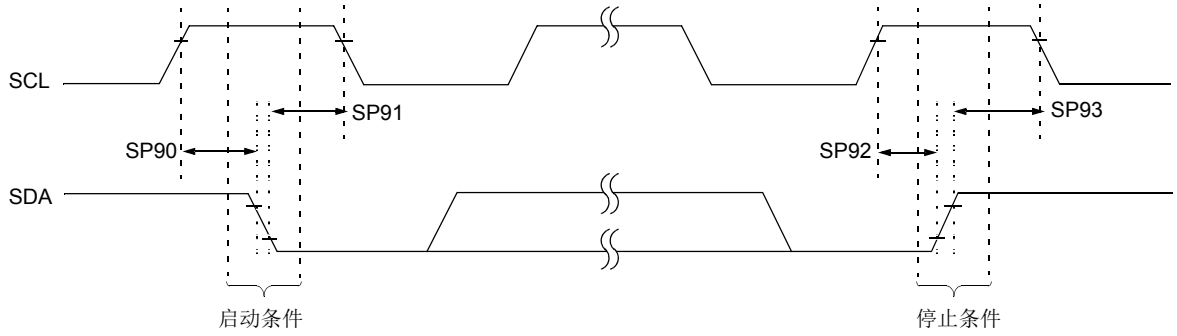
表45-21： SPI模式要求

标准工作条件（除非另外声明）							
参数编号	**符号**	**特性**	**最小值**	**典型值†**	**最大值**	**单位**	**条件**
SP70*	TssL2scH、TssL2scL	$\overline{SS}$↓到SCK↓或SCK↑输入的时间	2.25*Tcy	—	—	ns	
SP71*	TscH	SCK输入高电平时间（从模式）	Tcy + 20	—	—	ns	
SP72*	TscL	SCK输入低电平时间（从模式）	Tcy + 20	—	—	ns	
SP73*	TdiV2scH、TdiV2scL	SDI数据输入到SCK边沿的建立时间	100	—	—	ns	
SP74*	TscH2diL、TscL2diL	SDI数据输入到SCK边沿的保持时间	100	—	—	ns	
SP75*	TdoR	SDO数据输出上升时间	—	10	25	ns	3.0V ≤ VDD ≤ 5.5V
			—	25	50	ns	1.8V ≤ VDD ≤ 5.5V
SP76*	TdoF	SDO数据输出下降时间	—	10	25	ns	
SP77*	TssH2doZ	$\overline{SS}$↑到SDO输出高阻态的时间	10	—	50	ns	
SP78*	TscR	SCK输出上升时间（主模式）	—	10	25	ns	3.0V ≤ VDD ≤ 5.5V
			—	25	50	ns	1.8V ≤ VDD ≤ 5.5V
SP79*	TscF	SCK输出下降时间（主模式）	—	10	25	ns	
SP80*	TscH2doV，TscL2doV	SCK边沿之后SDO数据输出有效的时间	—	—	50	ns	3.0V ≤ VDD ≤ 5.5V
			—	—	145	ns	1.8V ≤ VDD ≤ 5.5V
SP81*	TdoV2scH，TdoV2scL	SDO数据输出建立到SCK边沿的时间	1 Tcy	—	—	ns	
SP82*	TssL2doV	$\overline{SS}$↓边沿之后SDO数据输出有效的时间	—	—	50	ns	
SP83*	TscH2ssH、TscL2ssH	SCK边沿之后$\overline{SS}$↑的时间	1.5 Tcy + 40	—	—	ns	

* 这些参数为特性值，但未经测试。

† 除非另外声明，否则“典型值”栏中的数据均为3.0V和25°C条件下的值。这些参数仅供设计参考，未经测试。

图45-18： I²C总线启动位/停止位时序



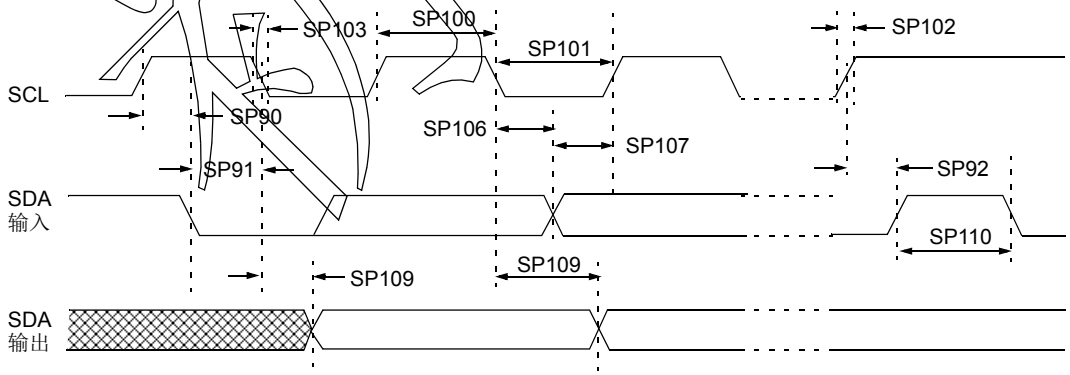
注： 负载条件请参见图45-4。

表45-22： I²C总线启动位/停止位要求

标准工作条件（除非另外声明）								
参数编号	**符号**	**特性**		**最小值**	**典型值**	**最大值**	**单位**	**条件**
SP90*	TSU:STA	启动条件建立时间	100 kHz模式	4700	—	—	ns	仅与重复启动条件相关
			400 kHz模式	600	—	—		
SP91*	THD:STA	启动条件保持时间	100 kHz模式	4000	—	—	ns	此周期后产生第一个时钟脉冲
			400 kHz模式	600	—	—		
SP92*	TSU:STO	停止条件建立时间	100 kHz模式	4700	—	—	ns	
			400 kHz模式	600	—	—		
SP93	THD:STO	停止条件保持时间	100 kHz模式	4000	—	—	ns	
			400 kHz模式	600	—	—		

* 这些参数为特性值，但未经测试。

图45-19： I²C总线数据时序



注： 负载条件请参见图45-4。

表45-23： I^2C总线数据要求

标准工作条件（除非另外声明）							
参数编号	**符号**	**特性**		**最小值**	**最大值**	**单位**	**条件**
SP100*	THIGH	时钟高电平时间	100 kHz模式	4.0	—	μs	器件工作频率不得低于1.5 MHz
			400 kHz模式	0.6	—	μs	器件工作频率不得低于10 MHz
			SSP模块	1.5TCY	—		
SP101*	TLOW	时钟低电平时间	100 kHz模式	4.7	—	μs	器件工作频率不得低于1.5 MHz
			400 kHz模式	1.3	—	μs	器件工作频率不得低于10 MHz
			SSP模块	1.5TCY	—		
SP102*	TR	SDA和SCL上升时间	100 kHz模式	—	1000	ns	
			400 kHz模式	20 + 0.1CB	300	ns	CB值规定为10-400 pF
SP103*	TF	SDA和SCL下降时间	100 kHz模式	—	250	ns	
			400 kHz模式	20 + 0.1CB	250	ns	CB值规定为10-400 pF
SP106*	THD:DAT	数据输入保持时间	100 kHz模式	0	—	ns	
			400 kHz模式	0	0.9	μs	
SP107*	TSU:DAT	数据输入建立时间	100 kHz模式	250	—	ns	（注2）
			400 kHz模式	100	—	ns	
SP109*	TAA	时钟有效到输出有效的时间	100 kHz模式	—	3500	ns	（注1）
			400 kHz模式	—	—	ns	
SP110*	TBUF	总线空闲时间	100 kHz模式	4.7	—	μs	在启动一个新的传输前总线必须保持空闲的时间
			400 kHz模式	1.3	—	μs	
SP111	CB	总线容性负载		—	400	pF	

* 这些参数为特性值，但未经测试。

注 1： 为避免意外产生启动或停止条件，作为发送器的器件必须提供这个内部最小延时（最小值300 ns）以补偿SCL下降沿的未定义区域。

2： 快速模式（400 kHz）的I^2C总线器件也可在标准模式（100 kHz）的I^2C总线系统中使用，但必须满足TSU:DAT ≥ 250 ns的要求。如果器件没有延长SCL信号的低电平周期，则必然满足此条件。如果该器件延长了SCL信号的低电平时间，其下一个数据位必须输出到SDA线。在SCL线被释放前，根据标准模式I^2C总线规范，TR max. + TSU:DAT = 1000 + 250 = 1250 ns。

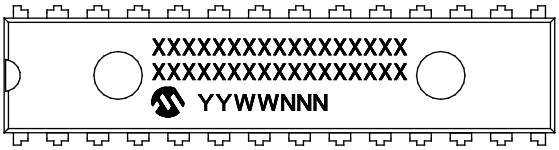
46.0 直流和交流特性图表

当前没有可用图表。

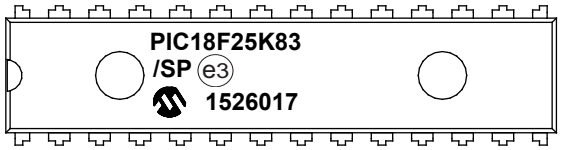
47.0 封装信息

封装标识信息

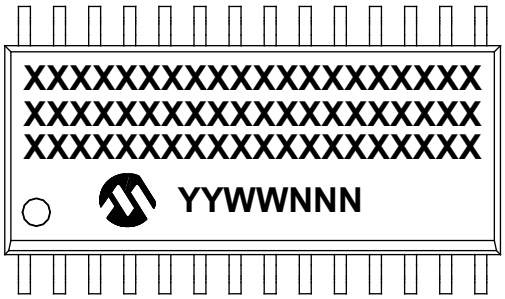
28 引脚 SPDIP（0.300”）



示例

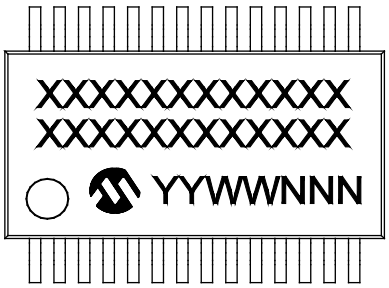


28 引脚 SOIC（7.50 mm）

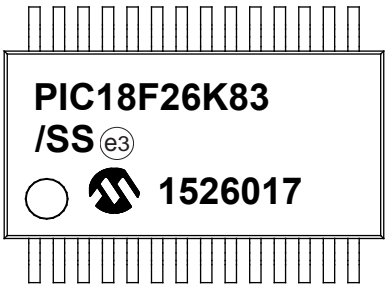


示例

28 引脚 SSOP（5.30 mm）



示例

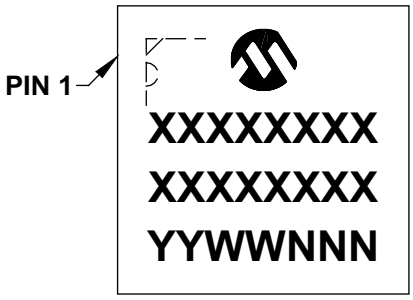


图注：	XX...X	客户指定信息或Microchip部件编号
	Y	年份代码（日历年的最后一位数字）
	YY	年份代码（日历年的最后两位数字）
	WW	星期代码（一月一日的星期代码为“01”）
	NNN	由字母数字组成的追踪代码
	(e3)	雾锡（Matte Tin，Sn）的JEDEC®无铅标志
	*	表示无铅封装。JEDEC无铅标志（(e3)）标示于此种封装的外包装上。

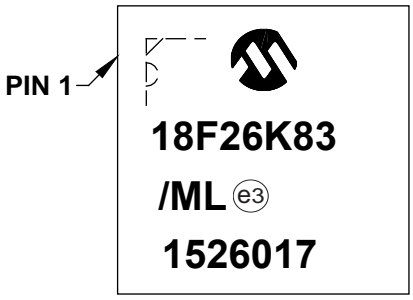
注： Microchip 部件编号如果无法在同一行内完整标注，将换行标出，因此会限制表示客户指定信息的字符数。

封装标识信息（续）

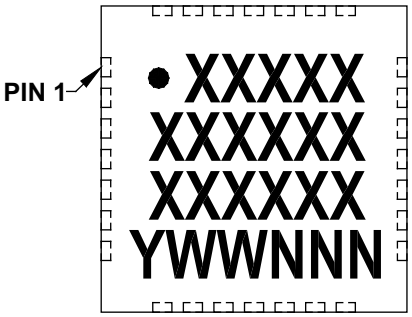
28 引脚 QFN（6x6 mm）



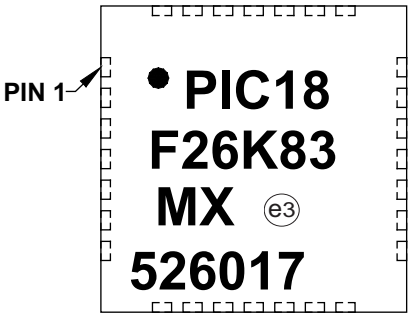
示例



28 引脚 UQFN（6x6x0.5 mm）



示例



图注：	XX...X	客户指定信息或Microchip部件编号
	Y	年份代码（日历年的最后一位数字）
	YY	年份代码（日历年的最后两位数字）
	WW	星期代码（一月一日的星期代码为“01”）
	NNN	由字母数字组成的追踪代码
	(e3)	雾锡（Matte Tin，Sn）的JEDEC®无铅标志
	*	表示无铅封装。JEDEC无铅标志（(e3)）标示于此种封装的外包装上。

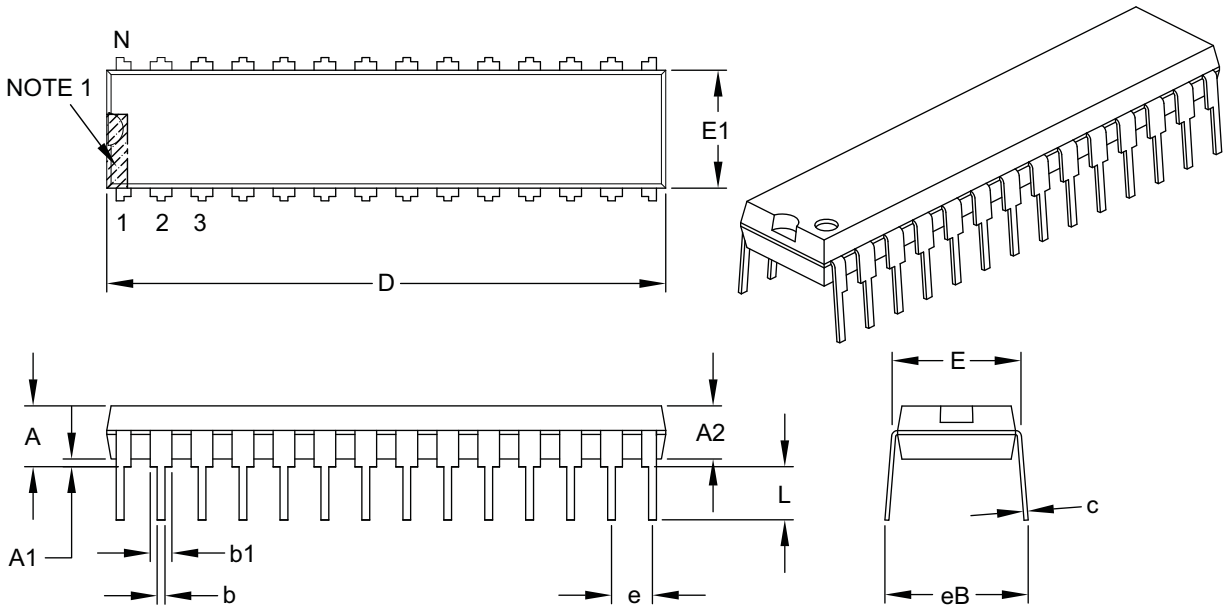
注： Microchip 部件编号如果无法在同一行内完整标注，将换行标出，因此会限制表示客户指定信息的字符数。

47.1 封装详细信息

以下部分将介绍各种封装的技术细节。

28引脚窄型塑封双列直插式封装（SP）——主体300 mil [SPDIP]

注： 最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

Notes:

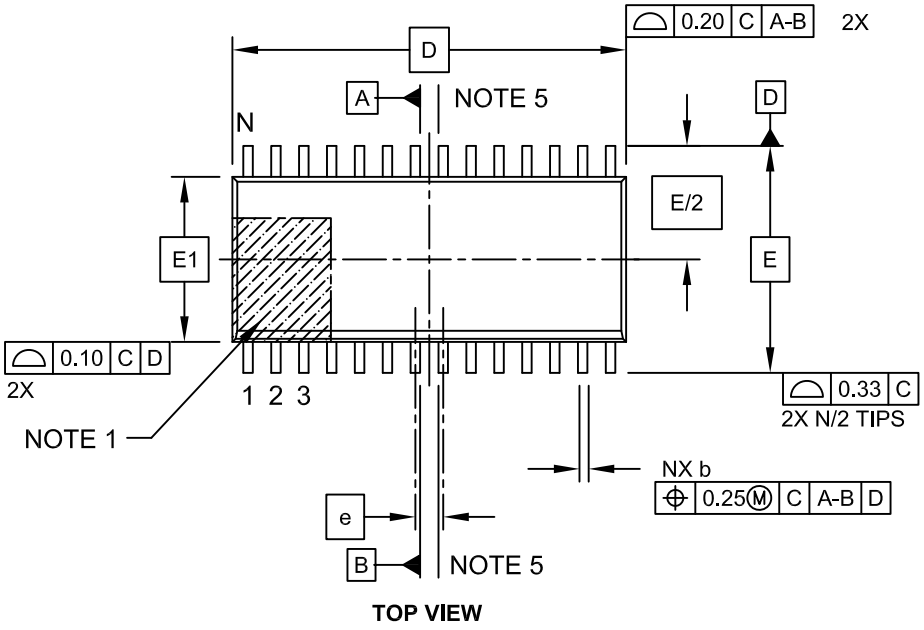
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

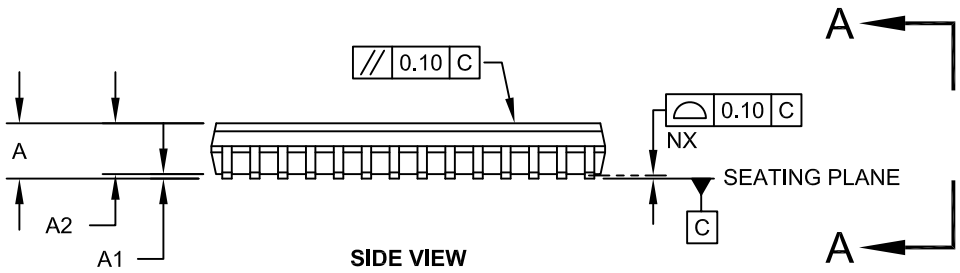
Microchip Technology Drawing C04-070B

28引脚塑封宽条小外形封装（SO）——主体7.50 mm [SOIC]

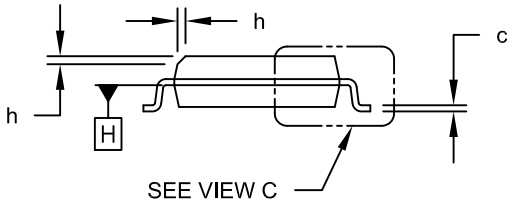
注：	最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



TOP VIEW



SIDE VIEW

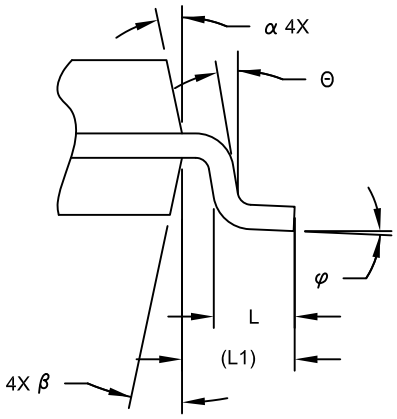


VIEW A-A

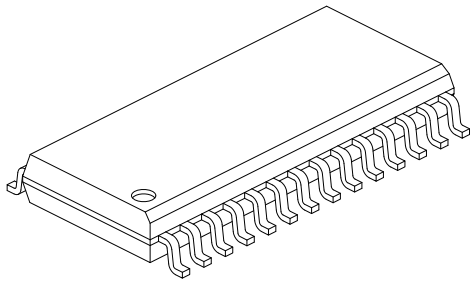
Microchip Technology Drawing C04-052C Sheet 1 of 2

28引脚塑封宽条小外形封装（SO）——主体7.50 mm [SOIC]

注： 最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



VIEW C



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		1.27 BSC	
Overall Height	A	-	-	2.65
Molded Package Thickness	A2	2.05	-	-
Standoff §	A1	0.10	-	0.30
Overall Width	E		10.30 BSC	
Molded Package Width	E1		7.50 BSC	
Overall Length	D		17.90 BSC	
Chamfer (Optional)	h	0.25	-	0.75
Foot Length	L	0.40	-	1.27
Footprint	L1		1.40 REF	
Lead Angle	Θ	0°	-	-
Foot Angle	φ	0°	-	8°
Lead Thickness	c	0.18	-	0.33
Lead Width	b	0.31	-	0.51
Mold Draft Angle Top	α	5°	-	15°
Mold Draft Angle Bottom	β	5°	-	15°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic
3. Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M

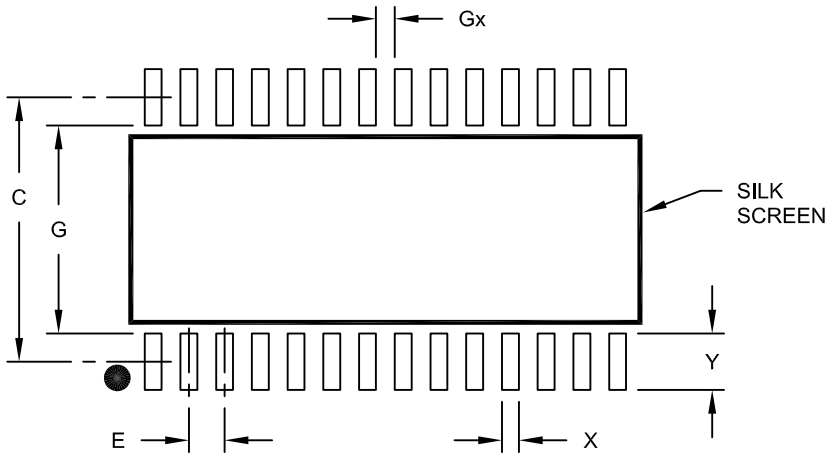
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.
5. Datums A & B to be determined at Datum H.

Microchip Technology Drawing C04-052C Sheet 2 of 2

28引脚塑封宽条小外形封装（SO）——主体7.50 mm [SOIC]

注： 最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		1.27 BSC	
Contact Pad Spacing	C		9.40	
Contact Pad Width (X28)	X			0.60
Contact Pad Length (X28)	Y			2.00
Distance Between Pads	Gx	0.67		
Distance Between Pads	G	7.40		

Notes:

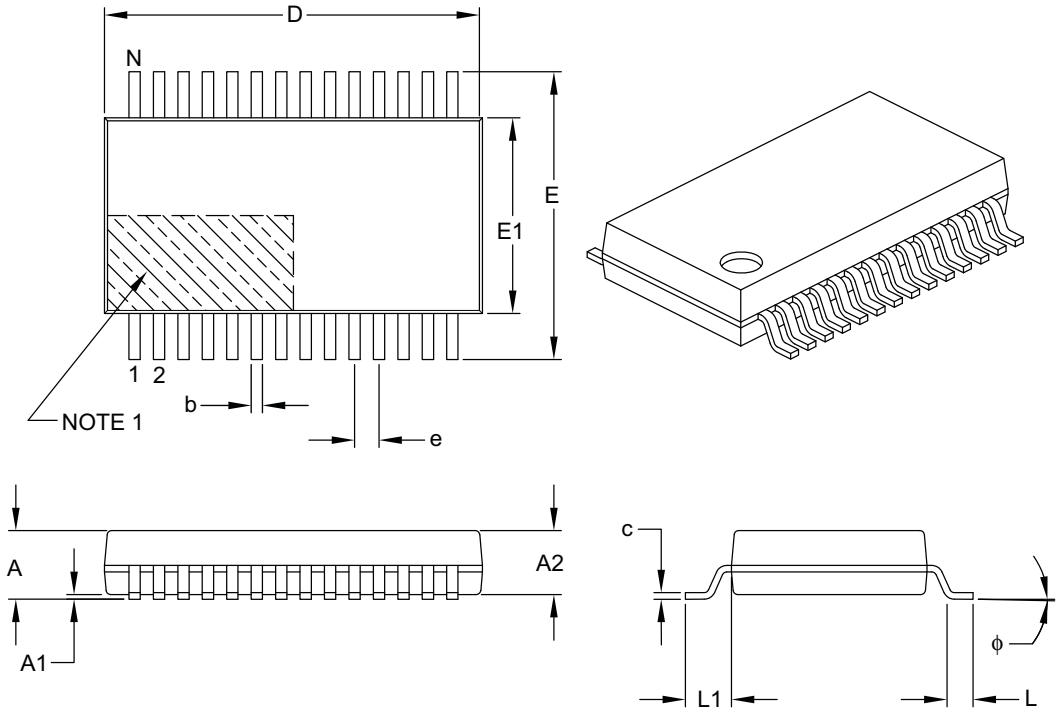
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2052A

28引脚塑封紧缩小外形封装（SS）——主体5.30 mm [SSOP]

注： 最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	–	–	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	–	–
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	9.90	10.20	10.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1		1.25 REF	
Lead Thickness	c	0.09	–	0.25
Foot Angle	ϕ	0°	4°	8°
Lead Width	b	0.22	–	0.38

Notes:

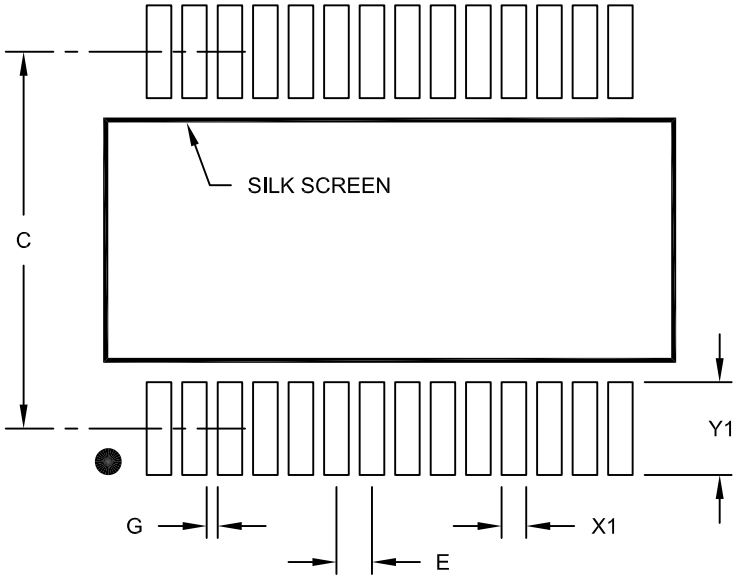
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

28引脚塑封紧缩小外形封装（SS）——主体5.30 mm [SSOP]

注： 最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Contact Pad Spacing	C		7.20	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.75
Distance Between Pads	G	0.20		

Notes:

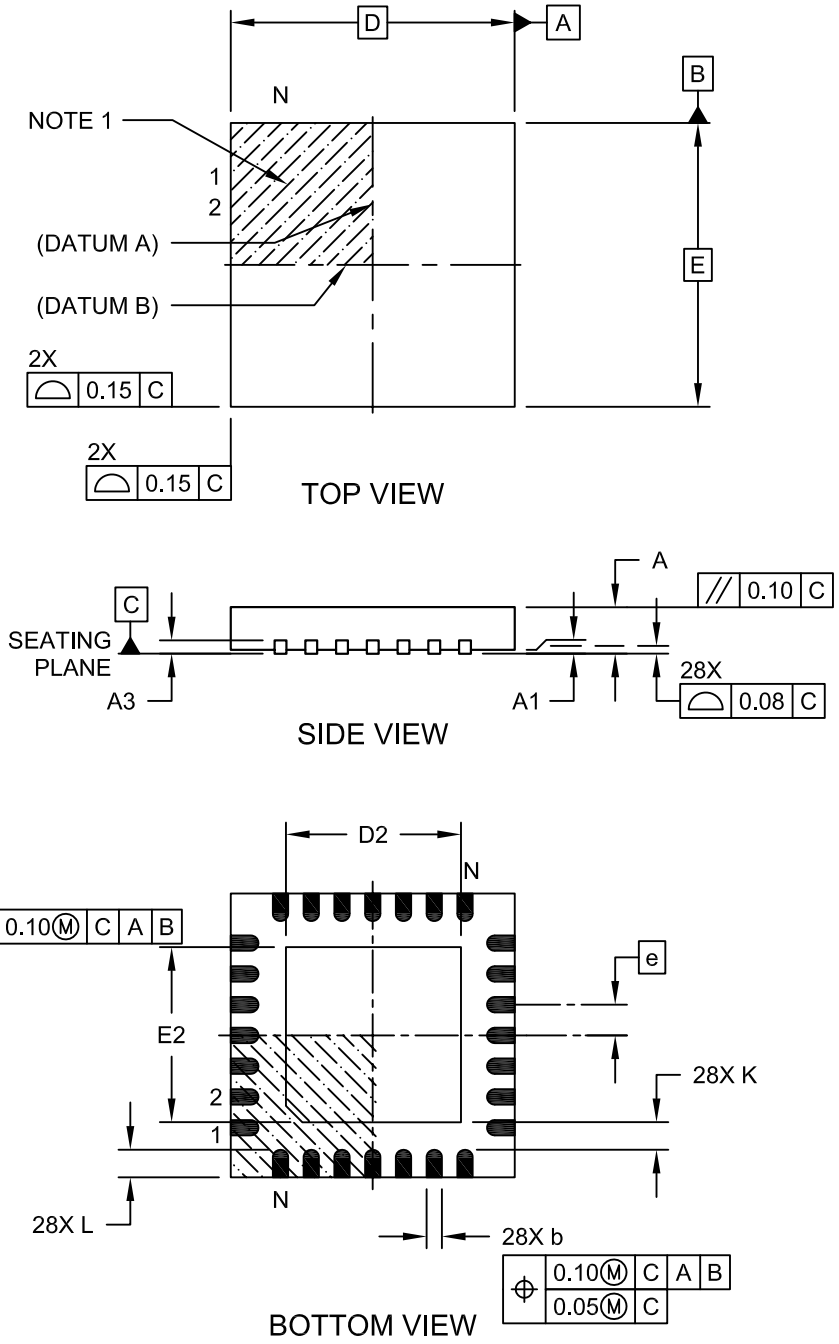
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2073A

28引脚塑封正方扁平无引脚封装（ML）——主体6x6 mm [QFN]
端子长度为0.55 mm

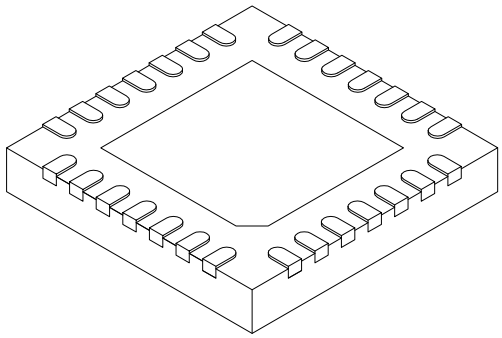
注：	最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



Microchip Technology Drawing C04-105C Sheet 1 of 2

28引脚塑封正方扁平无引脚封装（ML）——主体6x6 mm [QFN]
端子长度为0.55 mm

注： 最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



	Units	MILLIMETERS		
Dimension	Limits	MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.20 REF	
Overall Width	E		6.00 BSC	
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D		6.00 BSC	
Exposed Pad Length	D2	3.65	3.70	4.20
Terminal Width	b	0.23	0.30	0.35
Terminal Length	L	0.50	0.55	0.70
Terminal-to-Exposed Pad	K	0.20	-	-

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M.

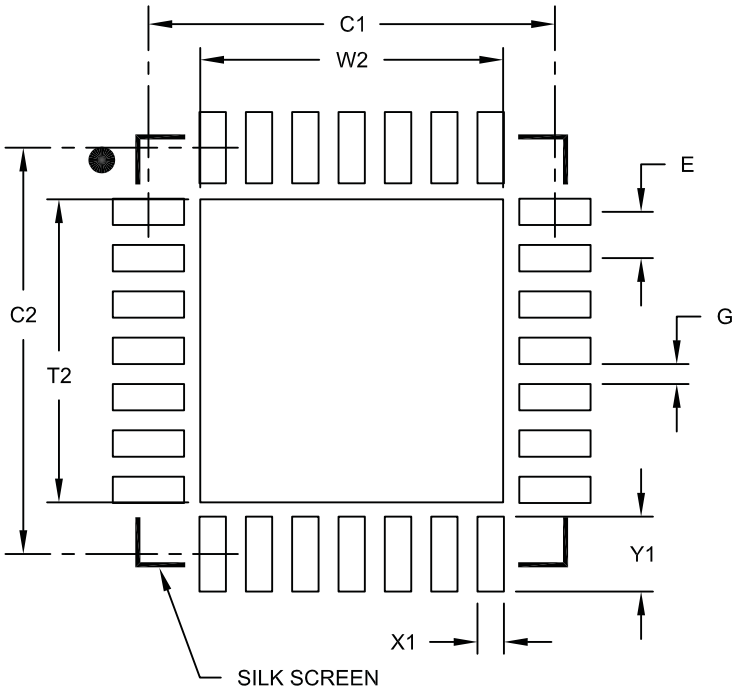
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-105C Sheet 2 of 2

28引脚塑封正方扁平无引脚封装（ML）——主体6x6 mm [QFN]
触点长度为0.55 mm

注：	最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
Distance Between Pads	G	0.20		

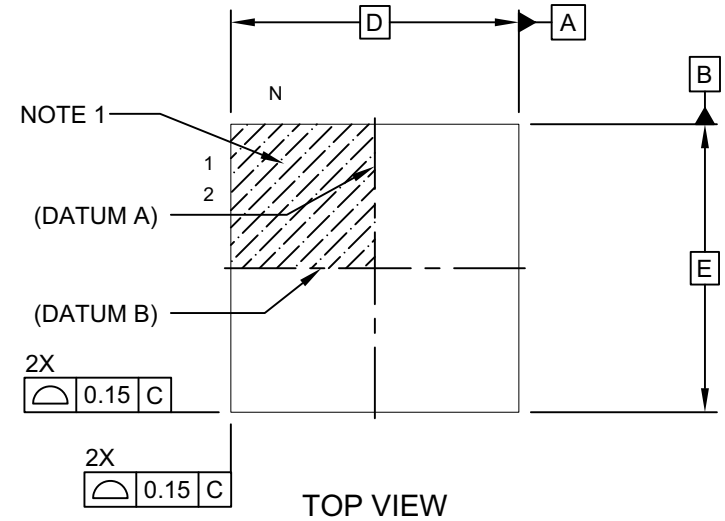
Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

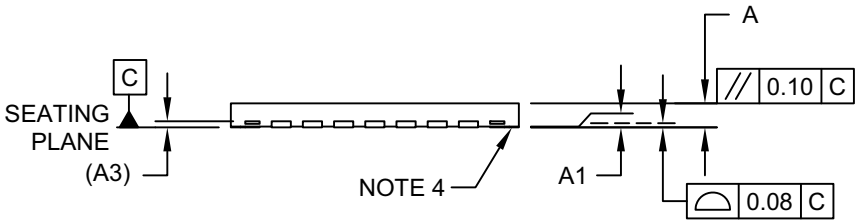
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2105A

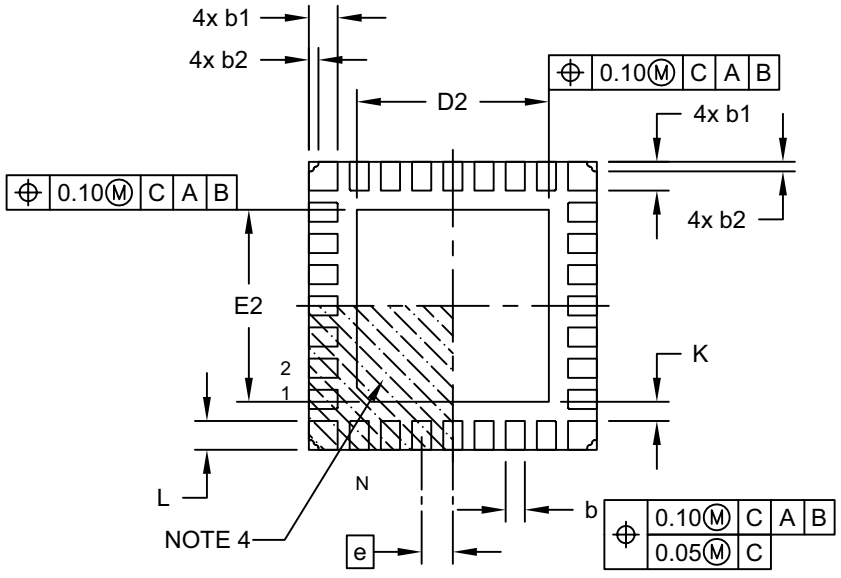
注：	最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



TOP VIEW



SIDE VIEW

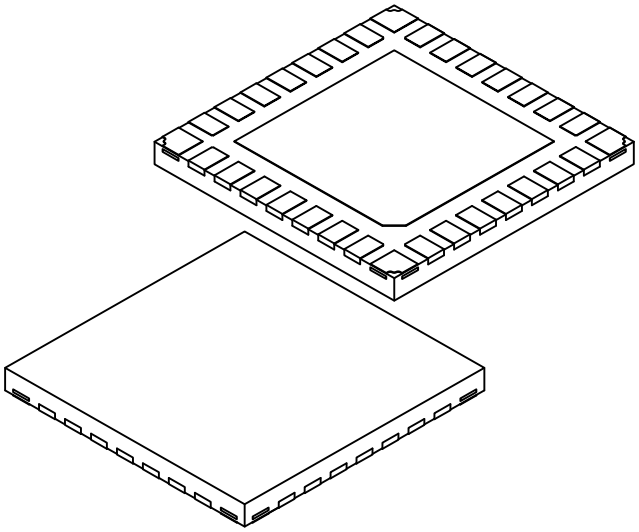


BOTTOM VIEW

Microchip Technology Drawing C04-0209 Rev C Sheet 1 of 2

28引脚塑封正方扁平无引脚封装（MX）——主体6x6x0.5 mm [UQFN]
超薄型，端子宽度/长度为0.40 x 0.60 mm，带角锚点

注： 最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	0.40	0.50	0.60
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	(A3)		0.127 REF	
Overall Width	E		6.00 BSC	
Exposed Pad Width	E2		4.00	
Overall Length	D		6.00 BSC	
Exposed Pad Length	D2		4.00	
Terminal Width	b	0.35	0.40	0.45
Corner Pad	b1	0.55	0.60	0.65
Corner Pad, Metal Free Zone	b2	0.15	0.20	0.25
Terminal Length	L	0.55	0.60	0.65
Terminal-to-Exposed Pad	K	0.20	-	-

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M

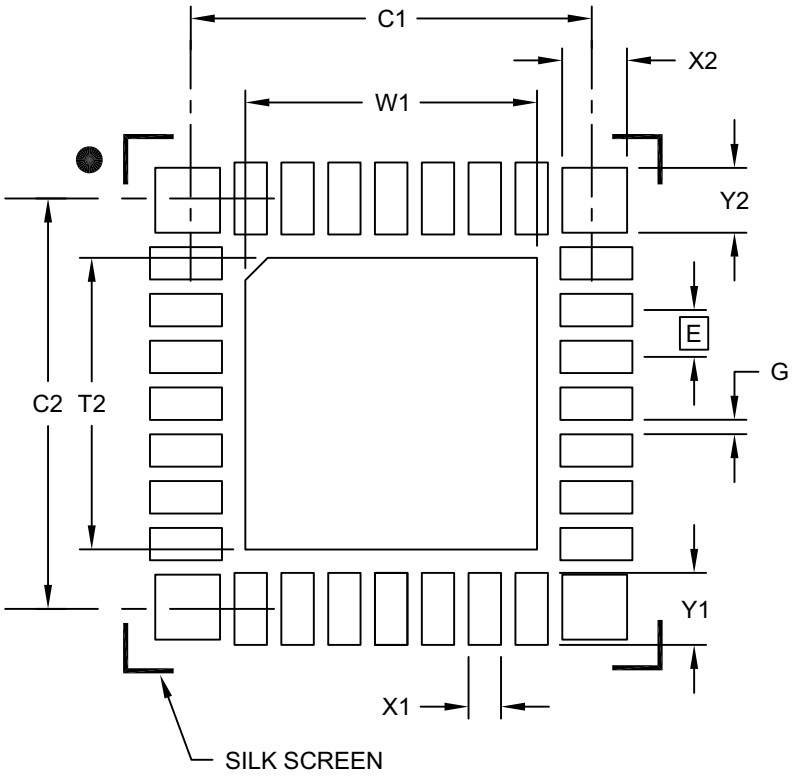
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.
4. Outermost portions of corner structures may vary slightly.

Microchip Technology Drawing C04-0209 Rev C Sheet 2 of 2

28引脚塑封正方扁平无引脚封装（MX）——主体6x6 mm [UQFN]
触点长度为0.60 mm，带角锚点

注：	最新封装图请至http://www.microchip.com/packaging查看Microchip封装规范。



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W1			4.05
Optional Center Pad Length	T2			4.05
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.00
Corner Pad Width (X4)	X2			0.90
Corner Pad Length (X4)	Y2			0.90
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2209B

版本历史

版本A（2017年8月）

本文档的初始版本。

MICROCHIP网站

Microchip 网站（www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的互联网浏览器即可访问。网站提供以下信息：

- **产品支持** —— 数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及归档软件
- **一般技术支持** —— 常见问题解答 （FAQ）、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务** —— 产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

变更通知客户服务

Microchip的变更通知客户服务有助于客户了解Microchip产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 www.microchip.com。在“支持”（Support）下，点击“变更通知客户”（Customer Change Notification）服务后按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师 （FAE）
- 技术支持

客户应联系其代理商、代表或应用工程师 （FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过 http://microchip.com/support 获得网上技术支持。

产品标识体系

欲订货或获取价格、交货等信息，请与我公司生产厂或各销售办事处联系。

部件编号	[X](2)	-	X	/XX	XXX
器件	卷带式选项		温度范围	封装	定制编号

器件：	PIC18F25K83，PIC18LF25K83 PIC18F26K83，PIC18LF26K83	
卷带式选项：	空白 = 标准包装（料管或托盘包装） T = 卷带式(1), (2)	
温度范围：	E = -40°C至+125°C（扩展级） I = -40°C至+85°C（工业级）	
封装：	ML = 28引脚QFN 6x6 mm MV = 28引脚UQFN 4x4x0.5 mm NG = 28引脚SOIC SP = 28引脚窄型塑封DIP SS = 28引脚SSOP	
定制编号：	QTP、SQTP、编码或特殊要求（其他情况均为空白）	

示例：

a) PIC18F25K83-E/P 301 = 扩展级温度，SPDIP封装，QTP模式#301。

b) PIC18F26K83-E/SO = 扩展级温度，SOIC封装。

c) PIC18F25K83T-I/ML = 卷带式，工业级温度，QFN封装。

注 1： 卷带式选项仅适用于具有工业级温度范围的ML、MV、SO和SS封装。

2： 卷带式标识符仅出现在产品目录的部件编号描述中。该标识符用于订货目的，不会印刷在器件封装上。

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下， Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿与那些注重代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是 “牢不可破”的。

代码保护功能处于持续发展中。Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了《数字器件千年版权法案 （Digital Millennium Copyright Act）》。如果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

ISBN：978-1-5224-4484-8

全球销售及服务网点

美洲

公司总部 Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://www.microchip.com/support
网址：www.microchip.com

亚特兰大 Atlanta
Duluth, GA
Tel: 1-678-957-9614
Fax: 1-678-957-1455

奥斯汀 Austin, TX
Tel: 1-512-257-3370

波士顿 Boston
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Novi, MI
Tel: 1-248-848-4000

休斯敦 Houston, TX
Tel: 1-281-894-5983

印第安纳波利斯 Indianapolis
Noblesville, IN
Tel: 1-317-773-8323
Fax: 1-317-773-5453
Tel: 1-317-536-2380

洛杉矶 Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608
Tel: 1-951-273-7800

罗利 Raleigh, NC
Tel: 1-919-844-7510

纽约 New York, NY
Tel: 1-631-435-6000

圣何塞 San Jose, CA
Tel: 1-408-735-9110
Tel: 1-408-436-4270

加拿大多伦多 Toronto
Tel: 1-905-695-1980
Fax: 1-905-695-2078

亚太地区

中国 - 北京
Tel: 86-10-8569-7000

中国 - 成都
Tel: 86-28-8665-5511

中国 - 重庆
Tel: 86-23-8980-9588

中国 - 东莞
Tel: 86-769-8702-9880

中国 - 广州
Tel: 86-20-8755-8029

中国 - 杭州
Tel: 86-571-8792-8115

中国 - 南京
Tel: 86-25-8473-2460

中国 - 青岛
Tel: 86-532-8502-7355

中国 - 上海
Tel: 86-21-3326-8000

中国 - 沈阳
Tel: 86-24-2334-2829

中国 - 深圳
Tel: 86-755-8864-2200

中国 - 苏州
Tel: 86-186-6233-1526

中国 - 武汉
Tel: 86-27-5980-5300

中国 - 西安
Tel: 86-29-8833-7252

中国 - 厦门
Tel: 86-592-238-8138

中国 - 香港特别行政区
Tel: 852-2943-5100

中国 - 珠海
Tel: 86-756-321-0040

台湾地区 - 高雄
Tel: 886-7-213-7830

台湾地区 - 台北
Tel: 886-2-2508-8600

台湾地区 - 新竹
Tel: 886-3-577-8366

亚太地区

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733

印度 India - Bangalore
Tel: 91-80-3090-4444

印度 India - New Delhi
Tel: 91-11-4160-8631

印度 India - Pune
Tel: 91-20-4121-0141

日本 Japan - Osaka
Tel: 81-6-6152-7160

日本 Japan - Tokyo
Tel: 81-3-6880-3770

韩国 Korea - Daegu
Tel: 82-53-744-4301

韩国 Korea - Seoul
Tel: 82-2-554-7200

马来西亚 Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

马来西亚 Malaysia - Penang
Tel: 60-4-227-8870

菲律宾 Philippines - Manila
Tel: 63-2-634-9065

新加坡 Singapore
Tel: 65-6334-8870

泰国 Thailand - Bangkok
Tel: 66-2-694-1351

越南 Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

欧洲

奥地利 Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

丹麦 Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

芬兰 Finland - Espoo
Tel: 358-9-4520-820

法国 France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 Germany - Garching
Tel: 49-8931-9700

德国 Germany - Haan
Tel: 49-2129-3766400

德国 Germany - Heilbronn
Tel: 49-7131-72400

德国 Germany - Karlsruhe
Tel: 49-721-625370

德国 Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

德国 Germany - Rosenheim
Tel: 49-8031-354-560

以色列 Israel - Ra'anana
Tel: 972-9-744-7705

意大利 Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

意大利 Italy - Padova
Tel: 39-049-7625286

荷兰 Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

挪威 Norway - Trondheim
Tel: 47-7288-4388

波兰 Poland - Warsaw
Tel: 48-22-3325737

罗马尼亚 Romania - Bucharest
Tel: 40-21-407-87-50

西班牙 Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

瑞典 Sweden - Gothenberg
Tel: 46-31-704-60-40

瑞典 Sweden - Stockholm
Tel: 46-8-5090-4654

英国 UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820