PIC18F66K80 系列
数据手册

采用 nanoWatt XLP 技术、带 ECAN™ 的 28/40/44/64 引脚增强型闪存单片机

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：978-1-61341-328-9

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip 位于美国亚利桑那州 Chandler 和 Tempe 与位于俄勒冈州 Gresham 的全球总部、设计和晶圆生产厂及位于美国加利福尼亚州和印度的设计中心均通过了 ISO/TS-16949:2002 认证。公司在 PIC® MCU 与 dsPIC® DSC、KEELOQ® 跳码器件、串行 EEPROM、单片机外设、非易失性存储器和模拟产品方面的质量体系流程均符合 ISO/TS-16949:2002。此外，Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。

PIC18F66K80 系列

采用 nanoWatt XLP 技术、带 ECAN™ 的 28/40/44/64 引脚增强型闪存单片机

功耗管理模式：

- 运行：CPU 工作，外设工作
- 空闲：CPU 不工作，外设工作
- 休眠：CPU 不工作，外设不工作
- 双速振荡器启动
- 故障保护时钟监视器（Fail-Safe Clock Monitor，FSCM）
- 节能的外设模块禁止（Peripheral Module Disable，PMD）
- 超低功耗唤醒
- 快速唤醒 （1 µs，典型值）
- 低功耗 WDT （300 nA，典型值）
- 运行模式电流最低为 3.8 µA （典型值）
- 空闲模式电流最低为 880 nA （典型值）
- 休眠模式电流最低为 13 nA （典型值）

ECAN 总线模块特性：

- 符合 CAN 2.0B Active 规范
- 三种工作模式：
  - 传统模式（完全向后兼容现有 PIC18CXX8/FXX8 CAN 模块）
  - 增强型模式
  - FIFO 模式或可编程发送 / 接收缓冲区
- 报文比特率最高可达 1 Mbps
- 支持 DeviceNet™ 数据字节过滤器
- 6 个可编程接收 / 发送缓冲区
- 3 个带有优先级的专用发送缓冲区
- 2 个专用接收缓冲区

ECAN 总线模块特性 （续）：

- 16 个可动态关联的 29 位完全接收过滤器
- 3 个 29 位完全接收屏蔽器
- 自动远程帧处理
- 高级错误管理功能

单片机特性：

- 工作电压范围：1.8V 至 5.5V
- 片上 3.3V 稳压器
- 工作速度最高可达 64 MHz
- 最大 64 KB 的片上闪存程序存储器：
  - 10,000 次擦 / 写操作（典型值）
  - 最少 20 年数据保存时间（典型值）
- 1,024 字节的数据 EEPROM：
  - 可进行 100,000 次擦 / 写操作的数据 EEPROM 存储器（典型值）
- 3.6 KB 的通用寄存器 （SRAM）
- 3 个内部振荡器：LF-INTOSC（31 KHz）、MF-INTOSC（500 kHz）和 HF-INTOSC（16 MHz）
- 可在软件控制下自编程
- 中断优先级
- 8 x 8 单周期硬件乘法器
- 扩展型看门狗定时器（Watchdog Timer，WDT）：
  - 可编程周期从 4 ms 至 4,194s
- 通过两个引脚进行在线串行编程（In-Circuit Serial Programming™， ICSP™）
- 通过两个引脚进行在线调试
- 可编程 BOR
- 可编程 LVD

表 1：　　器件比较

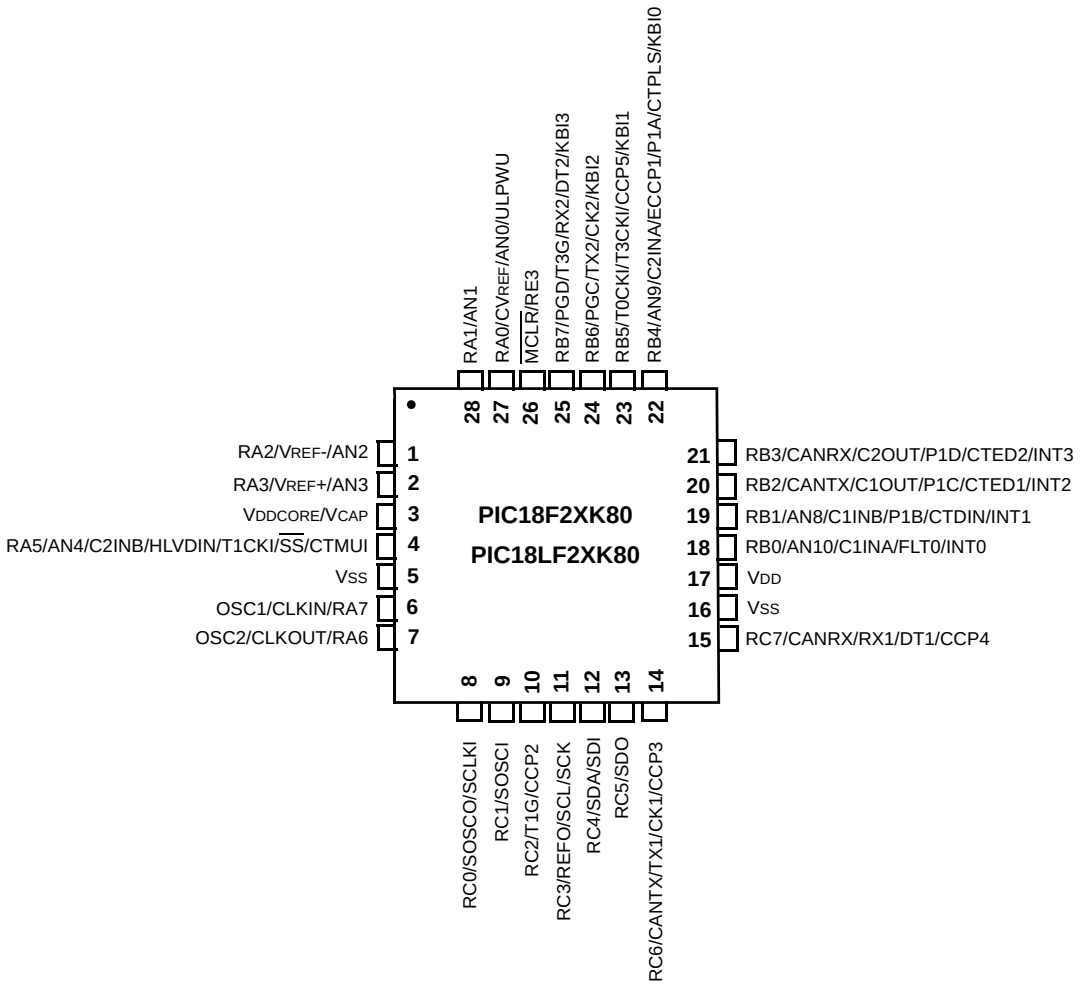
器件	程序存储器	数据存储器（字节）	数据 EE（字节）	引脚数	I/O	CTMU	12 位 A/D 通道数	CCP/ECCP	8 位 /16 位定时器	EUSART	比较器	ECAN™	MSSP	BORMV/LVD	DSM
PIC18F25K80	32 KB	3,648	1,024	28	24	1	8 路通道	4/1	2/3	2	2	1	1	有	无
PIC18LF25K80	32 KB	3,648	1,024	28	24	1	8 路通道	4/1	2/3	2	2	1	1	有	无
PIC18F26K80	64 KB	3,648	1,024	28	24	1	8 路通道	4/1	2/3	2	2	1	1	有	无
PIC18LF26K80	64 KB	3,648	1,024	28	24	1	8 路通道	4/1	2/3	2	2	1	1	有	无
PIC18F45K80	32 KB	3,648	1,024	40/44	35	1	11 路通道	4/1	2/3	2	2	1	1	有	无
PIC18LF45K80	32 KB	3,648	1,024	40/44	35	1	11 路通道	4/1	2/3	2	2	1	1	有	无
PIC18F46K80	64 KB	3,648	1,024	40/44	35	1	11 路通道	4/1	2/3	2	2	1	1	有	无
PIC18LF46K80	64 KB	3,648	1,024	40/44	35	1	11 路通道	4/1	2/3	2	2	1	1	有	无
PIC18F65K80	32 KB	3,648	1,024	64	54	1	11 路通道	4/1	2/3	2	2	1	1	有	有
PIC18LF65K80	32 KB	3,648	1,024	64	54	1	11 路通道	4/1	2/3	2	2	1	1	有	有
PIC18F66K80	64 KB	3,648	1,024	64	54	1	11 路通道	4/1	2/3	2	2	1	1	有	有
PIC18LF66K80	64 KB	3,648	1,024	64	54	1	11 路通道	4/1	2/3	2	2	1	1	有	有

外设特点：

- 5 个 CCP/ECCP 模块：
  - 4个捕捉/比较/PWM（Capture/Compare/PWM，CCP）模块
  - 1个增强型捕捉/比较/PWM（Enhanced Capture/Compare/PWM， ECCP）模块
- 5 个 8/16 位定时器 / 计数器模块：
  - Timer0：带有 8 位可编程预分频器的 8/16 位定时器 / 计数器
  - Timer1, 3：16 位定时器 / 计数器
  - Timer2, 4：8 位定时器 / 计数器
- 2 个模拟比较器
- 可配置的参考时钟输出
- 充电时间测量单元（Charge Time Measurement Unit，CTMU）：
  - 电容测量
  - 时间测量，分辨率典型值为 1 ns
  - 集成参考电压
- 高灌 / 拉电流：25 mA/25 mA（PORTB 和 PORTC）
- 最多 4 个外部中断
- 1 个主同步串行口（Master Synchronous Serial Port，MSSP）模块：
  - 3/4 线 SPI （支持所有 4 种 SPI 模式）
  - I^2C™ 主 / 从模式
- 2 个增强型可寻址 USART 模块：
  - 支持 LIN/J2602
  - 自动波特率检测 （Auto-Baud Detect， ABD）
- 最多具有 11 路通道的 12 位 A/D 转换器：
  - 自动采集和休眠操作
  - 差分输入工作模式
- 数据信号调制器模块：
  - 从各种模块输出中选择调制器和载波源
- 集成参考电压

引脚图

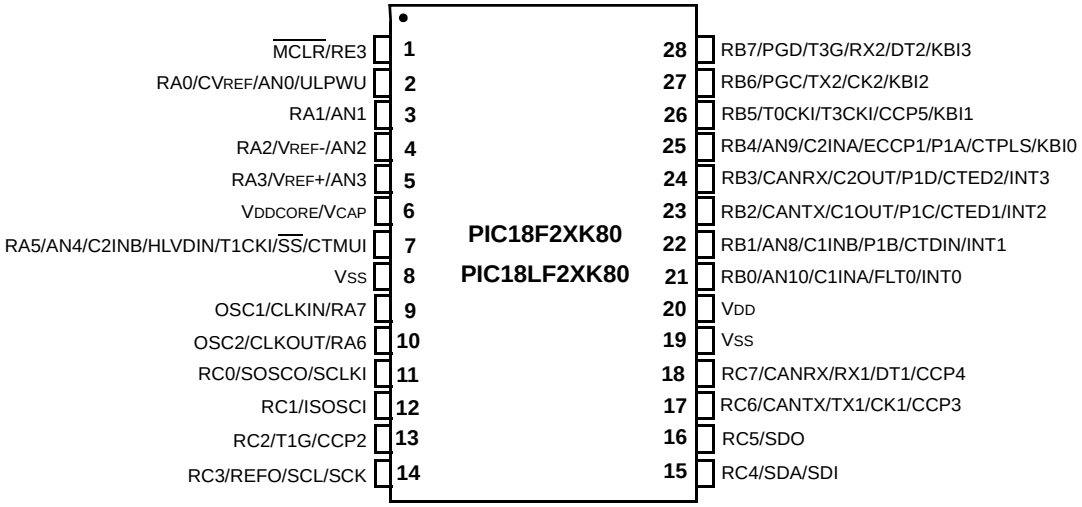
28 引脚 QFN(1)



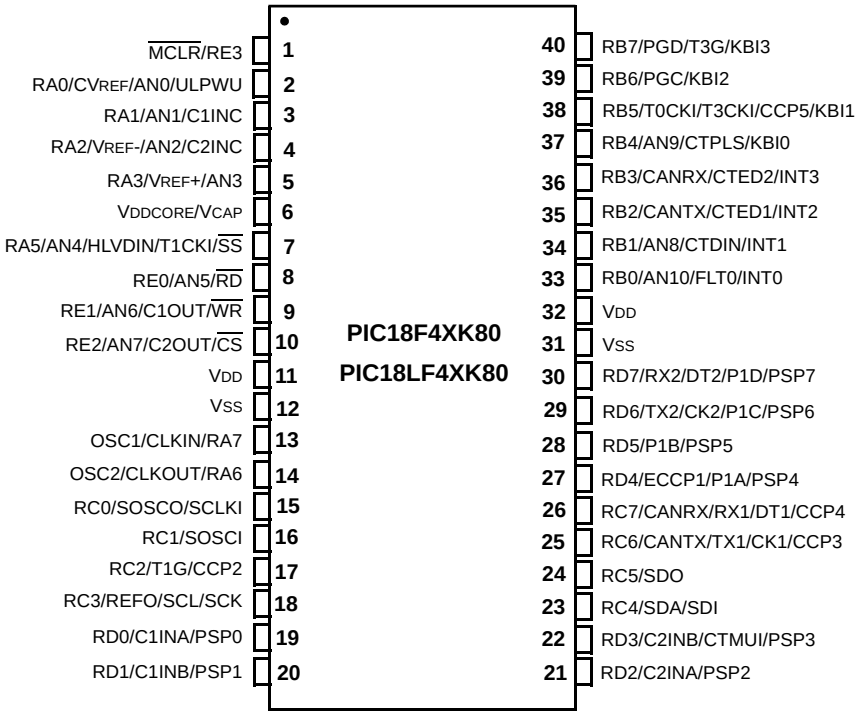
注 1： 对于 QFN 封装，建议将底部焊盘连接到 VSS。

引脚图 （续）

28 引脚 SSOP/SPDIP/SOIC

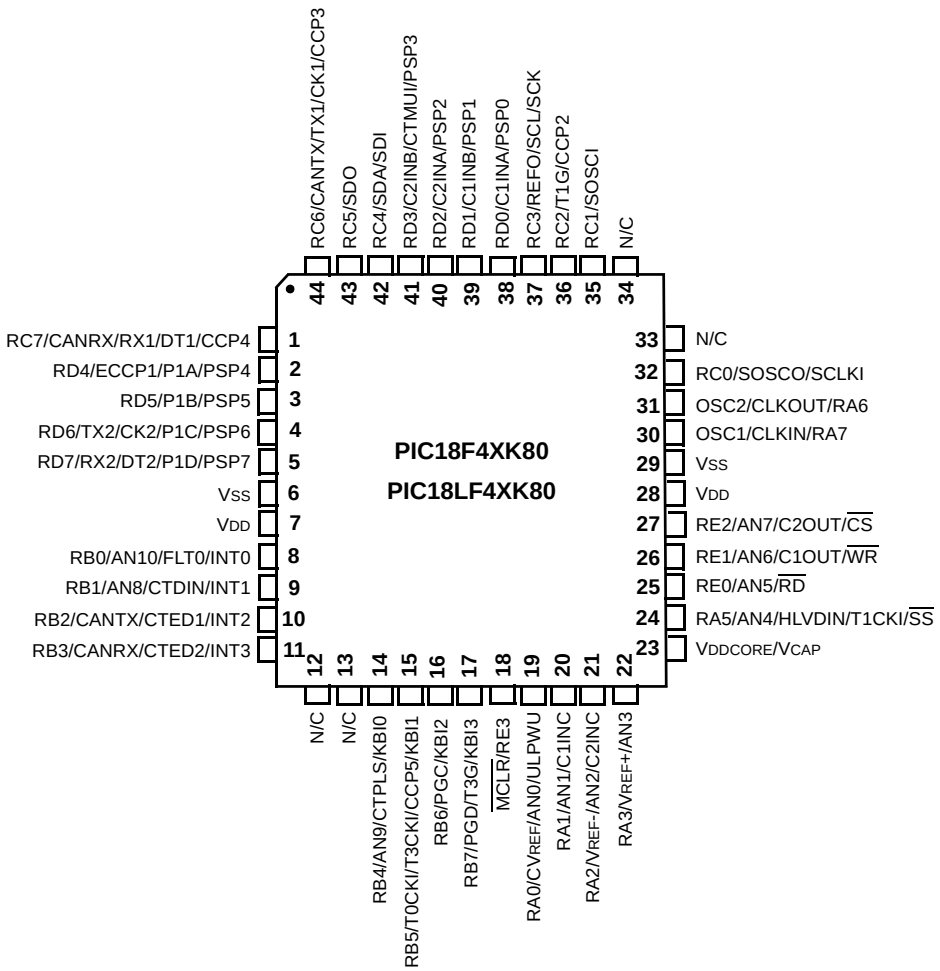


40 引脚 PDIP



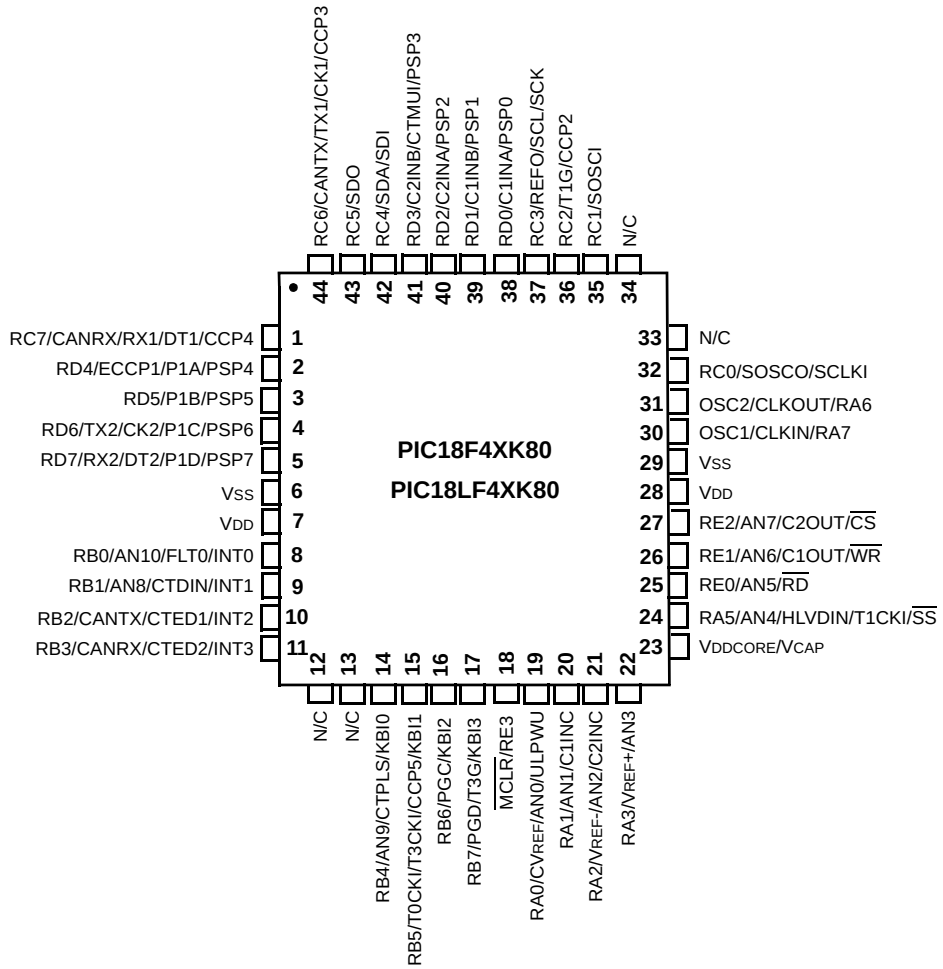
引脚图 （续）

44 引脚 TQFP



引脚图 （续）

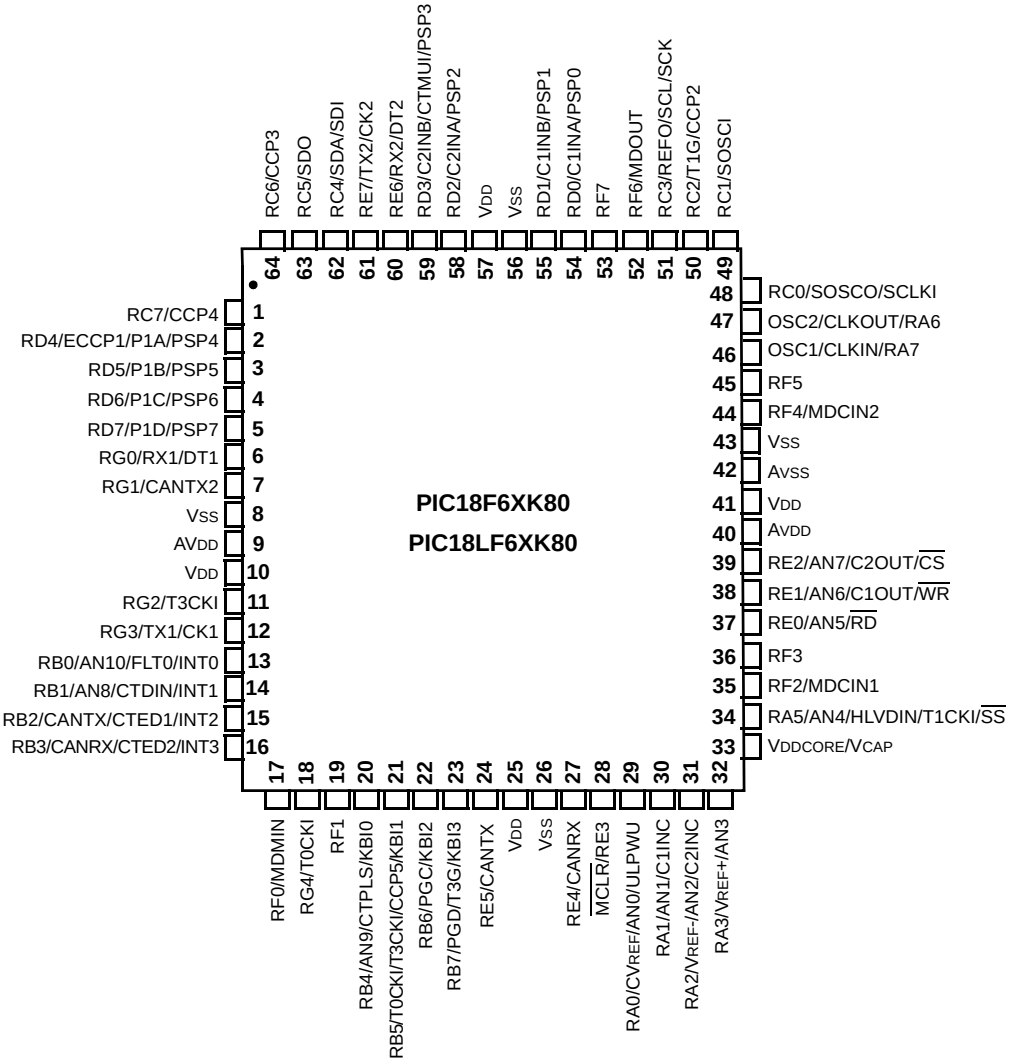
44 引脚 QFN(1)



注 1： 对于 QFN 封装，建议将底部焊盘连接到 VSS。

引脚图 （续）

64 引脚 QFN(1)/TQFP



注 1： 对于 QFN 封装，建议将底部焊盘连接到 VSS。

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要求。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**，或将本数据手册后附的**《读者反馈表》**传真到 86-21-5407 5066。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，请查询我公司的网站：

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。文献编号中数字串后的字母是版本号，例如：DS30000A 是 DS30000 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。

注：

1.0 器件概述

本文档包含以下器件的具体信息：

- PIC18F25K80
- PIC18F26K80
- PIC18F45K80
- PIC18F46K80
- PIC18F65K80
- PIC18F66K80
- PIC18LF25K80
- PIC18LF26K80
- PIC18LF45K80
- PIC18LF46K80
- PIC18LF65K80
- PIC18LF66K80

本系列具备所有 PIC18 单片机的传统优点，即出色的计算性能以及丰富的功能集，且产品价格极具竞争力。这些特性使得 PIC18F66K80 系列成为许多高性能，尤其是那些价格作为首要考虑因素的应用的理想选择。

1.1 内核特性

1.1.1 纳瓦技术

PIC18F66K80 系列的所有器件具有一系列能在工作时显著降低功耗的功能。主要包括以下几项：

- **备用运行模式：**通过将 Timer1 或内部 RC 振荡器作为控制器时钟源，可降低代码执行时的功耗。
- **多种空闲模式：**控制器还可在其 CPU 内核禁止而外设仍然工作的情况下工作。处于这些状态时，功耗可进一步降低。
- **动态模式切换：**在器件工作期间可由用户代码调用功耗管理模式，允许用户将节能的理念融入到其应用的软件设计中。
- **nanoWatt XLP：**额外的低功耗 BOR 和低功耗看门狗定时器。

1.1.2 振荡器选项和特性

PIC18F66K80 系列的所有器件可提供不同的振荡器选项，使用户在开发应用硬件时有很大的选择范围。这些选项包括：

- 外部电阻 / 电容 （RC）； RA6 可用
- 带时钟输出的外部电阻 / 电容 （RCIO）
- 三种外部时钟模式：
  - 外部时钟 （EC）； RA6 可用
  - 带时钟输出的外部时钟 （ECIO）
  - 外部晶振 （XT、 HS 和 LP）
- 一个锁相环 （Phase Lock Loop， PLL）倍频器，可在外部振荡器模式下使用，可使时钟速度最高达到 64 MHz。 PLL 还可以与内部振荡器一起使用。
- 一个内部振荡器模块，它提供一个 16 MHz 的时钟源（±2% 精度）和一个 INTOSC 时钟源（振荡频率大约为 31 kHz，温度和 V_{DD} 变化时频率保持稳定）
  - 为模块选择的频率为 16 MHz 或 500 kHz 时，将作为 HF-INTOSC 或 MF-INTOSC 工作
  - 空出两个振荡器引脚用作额外的通用 I/O 引脚

内部振荡器模块还提供了一个稳定的参考源，增加了以下功能以使器件更可靠地工作：

- **故障保护时钟监视器：**该功能的作用是持续监视主时钟源，将其与内部振荡器提供的参考信号作比较。如果时钟发生了故障，控制器会将时钟源切换到内部振荡器，使器件可继续低速工作或安全地关闭应用。
- **双速启动：**该功能允许在上电复位或从休眠模式唤醒时将内部振荡器用作时钟源，直到主时钟源可用为止。

1.1.3 存储器选项

PIC18F66K80 系列为应用程序代码提供了充足的空间，代码空间从 32 KB 至 64 KB。程序存储器的闪存单元经评测，最多可耐受 10,000 次擦 / 写。在不刷新的情况下，数据保存时间保守地估计在 20 年以上。

闪存程序存储器是可读写的。正常工作期间，PIC18F66K80 系列还为动态应用数据提供了充足的空间，最大为 3.6 KB 的数据 RAM。

1.1.4 扩展指令集

PIC18F66K80 系列在 PIC18 指令集的基础上进行了可选择的扩展，添加了 8 条新指令和一个变址寻址模式。此扩展实现了一个器件配置选项，它是为优化可重入应用程序代码而特别设计的，这些代码原来是通过高级语言 （如 C 语言）开发的。

1.1.5 移植方便

无论存储器容量如何，所有器件均共享同一组丰富外设，使得应用程序在扩展和升级时移植变得很方便。

整个系列的引脚排列设计一致也有助于向引脚更多的器件移植。在 28 引脚、40 引脚、44 引脚和 64 引脚器件之间进行移植时，乃至从较小容量存储器升级至较大容量存储器时，这一点都是真实可期的。

此外，PIC18F66K80 系列还在极大程度上与其他 PIC18 系列（例如带有 ECAN 模块的 PIC18F4580、PIC18F4680 和 PIC18F8680 单片机系列）保持引脚兼容。这为应用的更新换代拓展了空间，使开发人员能在保留类似功能集的同时在 Microchip PIC18 系列中选择不同价位的器件。

1.2 其他特殊功能

- **通信：**PIC18F66K80 系列具有一系列的串行通信外设，包括2个支持LIN/J2602的增强型USART、1 个同时支持 SPI 和 I^2C™（主 / 从）工作模式的主 SSP 模块和 1 个增强型 CAN 模块。
- **CCP 模块：**PIC18F66K80 系列器件具有 4 个捕捉 / 比较 /PWM（CCP）模块。在同一时间，最多可以使用 4 种不同时基来执行几项不同的操作。
- **ECCP 模块：**PIC18F66K80 系列具有 1 个增强型 CCP（ECCP）模块，用以最大程度提高控制应用的灵活性：
  - 最多 4 种不同时基，用于同时执行几种不同的操作
  - 最多 4 路 PWM 输出
  - 其他有用功能，如极性选择、可编程死区、自动关闭和重启，以及半桥和全桥输出模式
- **12 位 A/D 转换器：**PIC18F66K80 系列具有一个差分 ADC。该模块具备可编程采集时间，从而不必在选择通道和启动转换之间等待一个采样周期，因而减少了代码开销。
- **充电时间测量单元（CTMU）：**CTMU 是一个灵活的模拟模块，它提供脉冲源之间的精确时间差测量，以及异步脉冲生成。

  CTMU 可与其他片上模拟模块一起，用于精确测量时间、电容、电容的相对变化，或生成独立于系统时钟的输出脉冲。
- **LP 看门狗定时器（WDT）：**该增强型版本加入了一个 22 位预分频器，可以提供在不同工作电压和温度下保持稳定的扩展超时范围。超时周期请参见 **第 31.0 节 “ 电气特性 ”** 。

1.3 系列中各器件的详细说明

PIC18F66K80 系列器件提供 28 引脚、40/44 引脚和 64 引脚封装形式。图 1-1、图 1-2 和图 1-3 分别给出了每种封装的框图。

这些器件在以下方面存在差异：

- 闪存程序存储器：
  - PIC18FX5K80（PIC18F25K80、PIC18F45K80 和 PIC18F45K80） ——32 KB
  - PIC18FX6K80（PIC18F26K80、PIC18F46K80 和 PIC18F46K80） ——64 KB
- I/O 端口：
  - PIC18F2XK80 （28 引脚器件） ——3 个双向端口
  - PIC18F4XK80（40/44 引脚器件）——5 个双向端口
  - PIC18F6XK80 （64 引脚器件） ——7 个双向端口

本系列器件的所有其他功能都是相同的。表 1-1、表 1-2 和表 1-3 汇总了这些功能。

表 1-4、表 1-5 和表 1-6 列出了所有器件的引脚排列。

表 1-1： PIC18F2XK80 器件特性 （28 引脚器件）

特性	PIC18F25K80	PIC18F26K80
工作频率	DC – 64 MHz	
程序存储器 （字节）	32K	64K
程序存储器 （指令）	16,384	32,768
数据存储器 （字节）	3.6K	
中断源	31	
I/O 端口	端口 A、 B 和 C	
并行通信	并行从端口 （Parallel Slave Port， PSP）	
定时器	5 个	
比较器	2 个	
CTMU	有	
捕捉 / 比较 /PWM （CCP）模块	4 个	
增强型 CCP （ECCP）模块	1 个	
串行通信	1 个 MSSP 和 2 个增强型 USART （EUSART）	
12 位模数转换模块	8 路输入通道	
复位 （和延时）	POR、 BOR、 RESET 指令、堆栈满、堆栈下溢、 $\overline{\text{MCLR}}$ 和 WDT （PWRT 和 OST）	
指令集	75 条指令，使能扩展指令集后总共为 83 条指令	
封装	28 引脚 QFN-S、 SOIC、 SPDIP 和 SSOP	

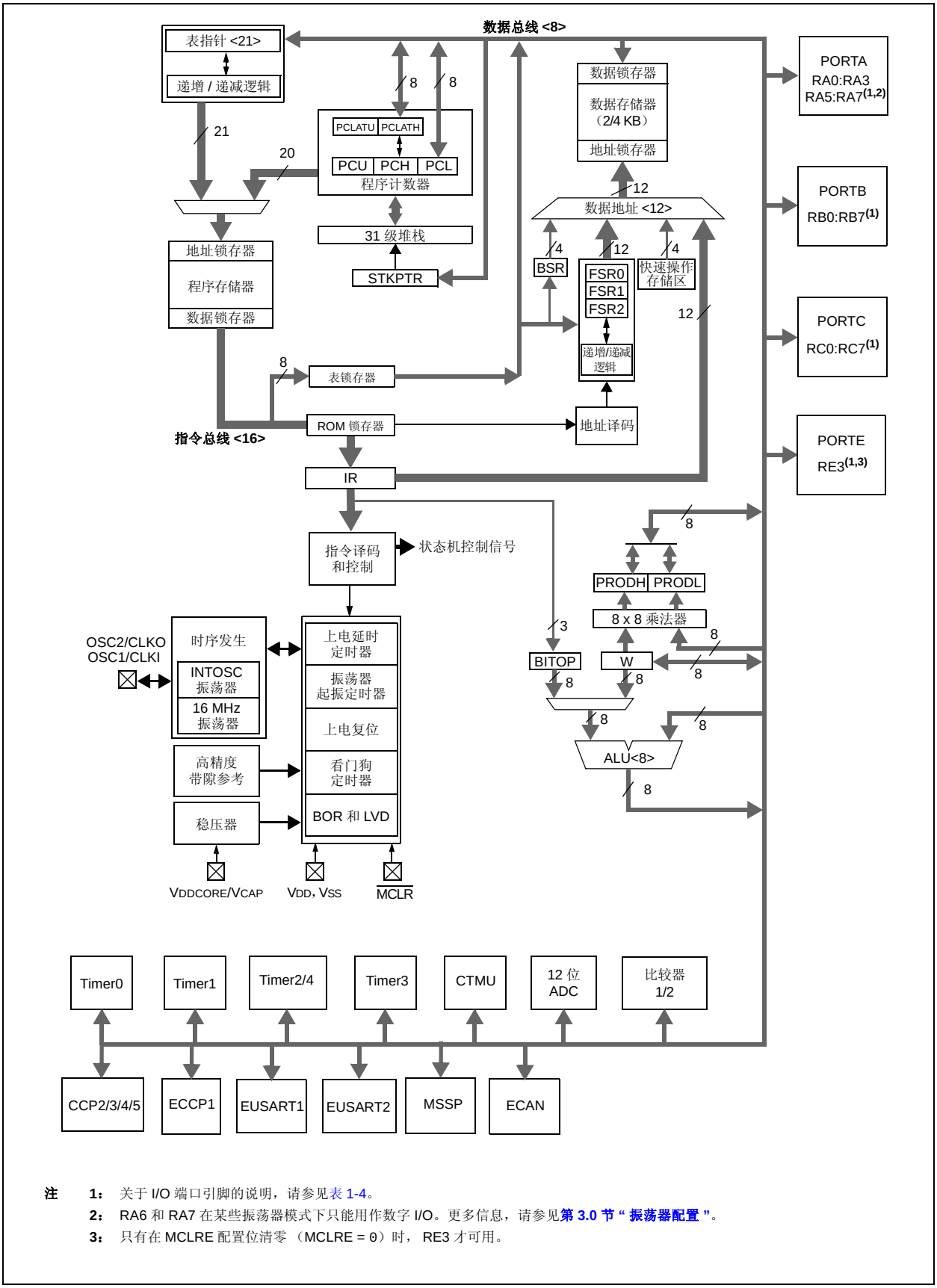
表 1-2： PIC18F4XK80 器件特性 （40/44 引脚器件）

特性	PIC18F45K80	PIC18F46K80
工作频率	DC – 64 MHz	
程序存储器 （字节）	32K	64K
程序存储器 （指令）	16,384	32,768
数据存储器 （字节）	3.6K	
中断源	32	
I/O 端口	端口 A、 B、 C、 D 和 E	
并行通信	并行从端口 （PSP）	
定时器	5 个	
比较器	2 个	
CTMU	有	
捕捉 / 比较 /PWM （CCP）模块	4 个	
增强型 CCP （ECCP）模块	1 个	
串行通信	1 个 MSSP 和 2 个增强型 USART （EUSART）	
12 位模数转换模块	11 路输入通道	
复位 （和延时）	POR、 BOR、 RESET 指令、堆栈满、堆栈下溢、 $\overline{\text{MCLR}}$ 和 WDT （PWRT 和 OST）	
指令集	75 条指令，使能扩展指令集后总共为 83 条指令	
封装	40 引脚 PDIP 以及 44 引脚 QFN 和 TQFP	

表 1-3： PIC18F6XK80 器件特性 （64 引脚器件）

特性	PIC18F65K80	PIC18F66K80
工作频率	DC – 64 MHz	
程序存储器 （字节）	32K	64K
程序存储器 （指令）	16,384	32,768
数据存储器 （字节）	3.6K	
中断源	32	
I/O 端口	端口 A、 B、 C、 D、 E、 F 和 G	
并行通信	并行从端口 （PSP）	
定时器	5 个	
比较器	2 个	
CTMU	有	
捕捉 / 比较 /PWM （CCP）模块	4 个	
增强型 CCP （ECCP）模块	1 个	
DSM	有	有
串行通信	1 个 MSSP 和 2 个增强型 USART （EUSART）	
12 位模数转换模块	11 路输入通道	
复位 （和延时）	POR、 BOR、 RESET 指令、堆栈满、堆栈下溢、 $\overline{\text{MCLR}}$ 和 WDT （PWRT 和 OST）	
指令集	75 条指令，使能扩展指令集后总共为 83 条指令	
封装	64 引脚 QFN 和 TQFP	

图 1-1：　　PIC18F2XK80 （28 引脚）框图

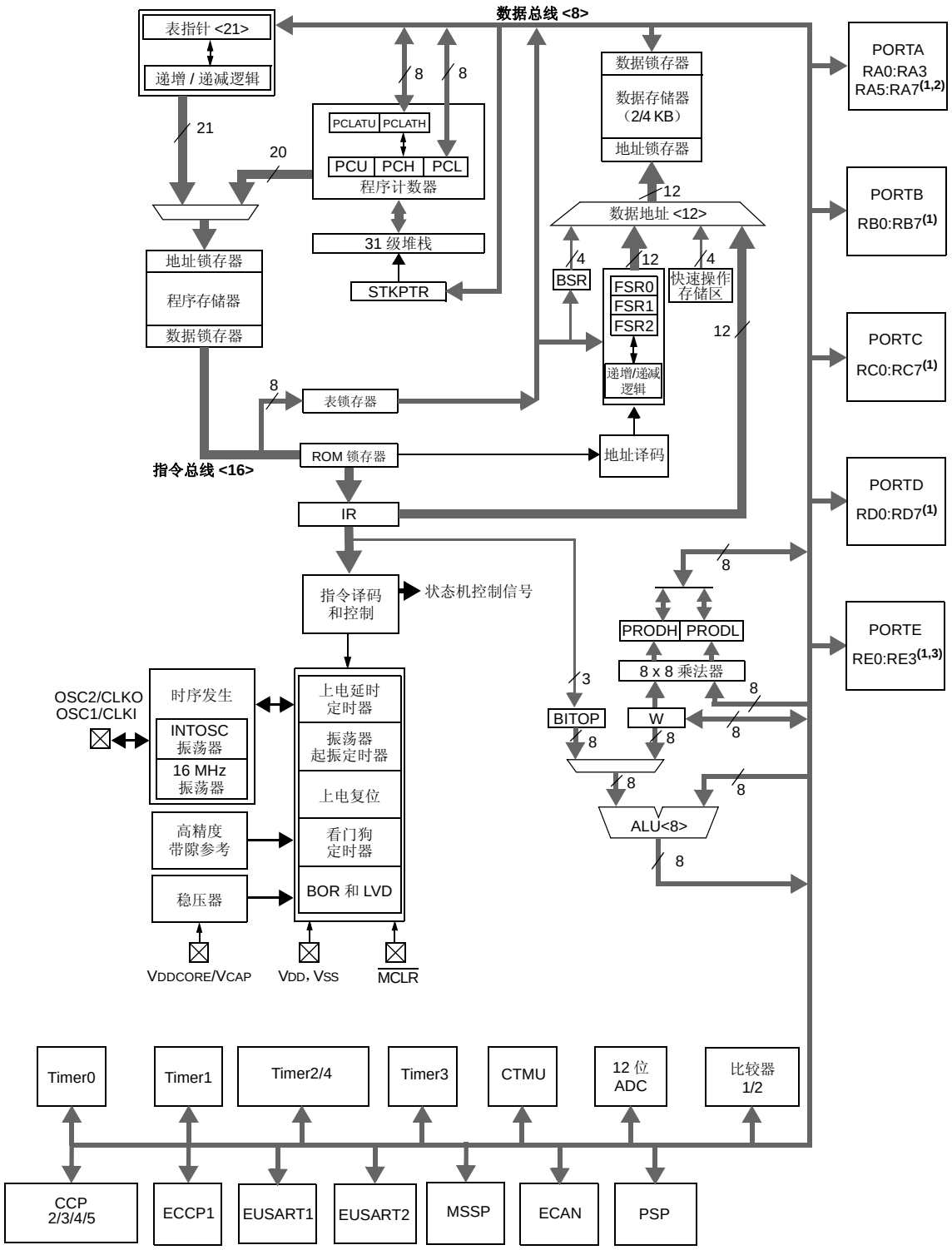


注　**1：** 关于 I/O 端口引脚的说明，请参见表 1-4。

2： RA6 和 RA7 在某些振荡器模式下只能用作数字 I/O。更多信息，请参见**第 3.0 节 “ 振荡器配置 ”**。

3： 只有在 MCLRE 配置位清零 （MCLRE = 0）时， RE3 才可用。

图 1-2：　　PIC18F4XK80 （40/44 引脚）框图

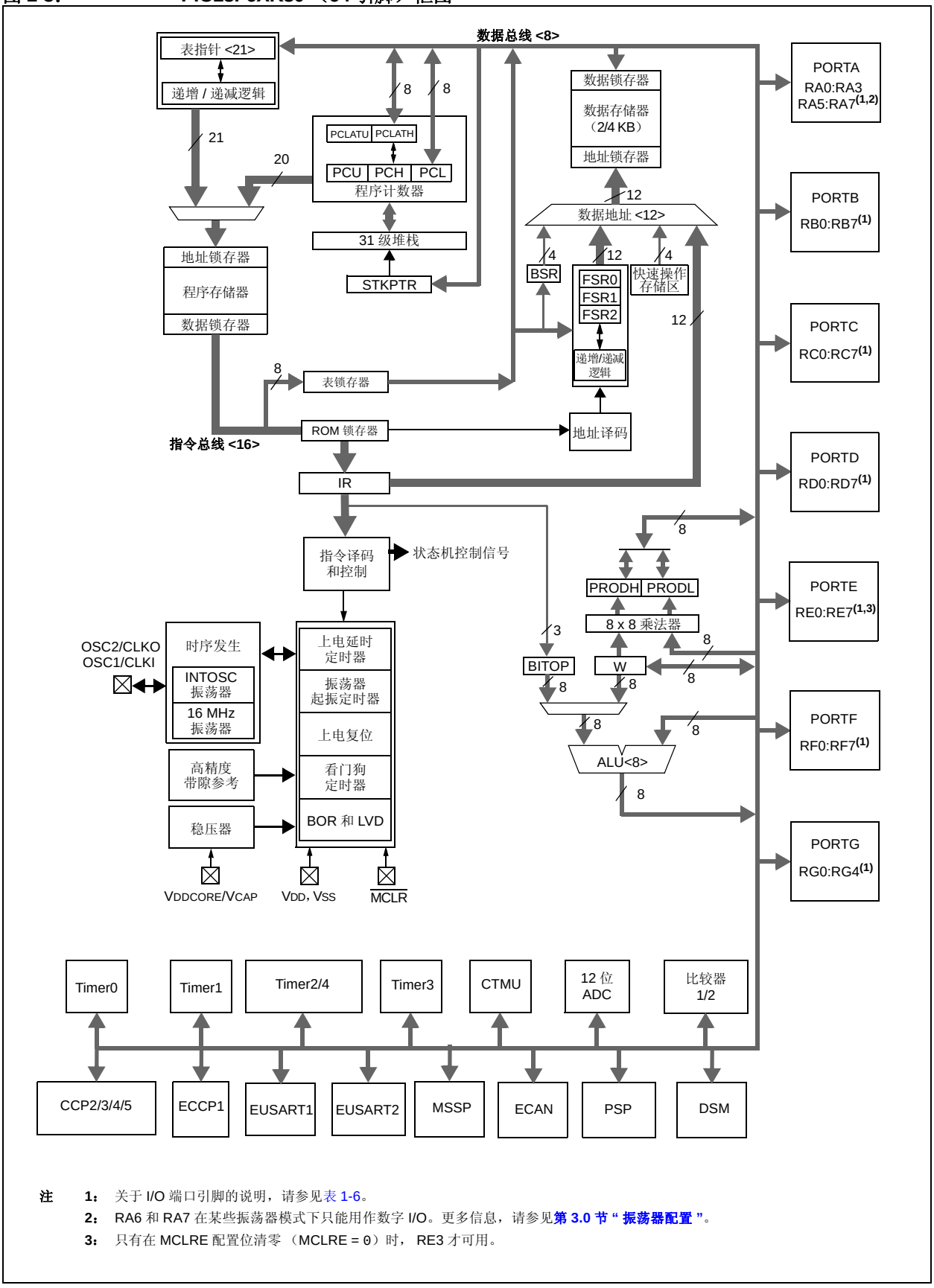


注　**1：** 关于 I/O 端口引脚的说明，请参见表 1-5。

2： RA6 和 RA7 在某些振荡器模式下只能用作数字 I/O。更多信息，请参见**第 3.0 节 “ 振荡器配置 ”**。

3： 只有在 MCLRE 配置位清零 （MCLRE = 0）时， RE3 才可用。

图 1-3：　　PIC18F6XK80 （64 引脚）框图



注　**1：** 关于 I/O 端口引脚的说明，请参见表 1-6。

2： RA6 和 RA7 在某些振荡器模式下只能用作数字 I/O。更多信息，请参见**第 3.0 节 “ 振荡器配置 ”**。

3： 只有在 MCLRE 配置位清零 （MCLRE = 0）时， RE3 才可用。

表 1-4： PIC18F2XK80 I/O 说明

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	QFN	SSOP/ SPDIP/ SOIC			
$\overline{\text{MCLR}}$/RE3	26	1			
$\overline{\text{MCLR}}$			I	ST	主复位 （输入）或编程电压 （输入）。此引脚为低电平有效的器件复位输入端。
RE3			I	ST	仅用作输入的通用引脚。
OSC1/CLKIN/RA7	6	9			
OSC1			I	ST	晶振输入。
CLKIN			I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能相关联。（见相关的 OSC1/CLKI 和 OSC2/CLKO 引脚信息。）
RA7			I/O	ST/ CMOS	通用 I/O 引脚。
OSC2/CLKOUT/RA6	7	10			
OSC2			O	—	晶振输出。 在晶振模式下，该引脚与晶振或谐振器相连。
CLKOUT			O	—	在某种振荡器模式下， OSC2 引脚输出 CLKO 信号，其频率是 OSC1 引脚上信号频率的 1/4，该频率等于指令周期的倒数。
RA6			I/O	ST/ CMOS	通用 I/O 引脚。

图注： CMOS = CMOS 兼容输入或输出　　I^2C™ = I^2C/SMBus 输入缓冲器
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-4： PIC18F2XK80 I/O 说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	QFN	SSOP/ SPDIP/ SOIC			
					PORTA 是双向 I/O 端口。
RA0/CVREF/AN0/ULPWU	27	2			
RA0			I/O	ST/ CMOS	通用 I/O 引脚。
CVREF			O	Analog	比较器参考电压输出。
AN0			I	Analog	模拟输入 0。
ULPWU			I	Analog	超低功耗唤醒输入。
RA1/AN1	28	3			
RA1			I/O	ST/ CMOS	数字 I/O。
AN1			I	Analog	模拟输入 1。
RA2/VREF-/AN2	1	4			
RA2			I/O	ST/ CMOS	数字 I/O。
VREF-			I	Analog	A/D 参考电压（低电压）输入。
AN2			I	Analog	模拟输入 2。
RA3/VREF+/AN3	2	5			
RA3			I/O	ST/ CMOS	数字 I/O。
VREF+			I	Analog	A/D 参考电压 （高电压）输入。
AN3			I	Analog	模拟输入 3。
RA5/AN4/C2INB/HLVDIN/ T1CKI/$\overline{SS}$/CTMUI	4	7			
RA5			I/O	ST/ CMOS	数字 I/O。
AN4			I	Analog	模拟输入 4。
C2INB			I	Analog	比较器 2 的输入 B。
HLVDIN			I	Analog	高 / 低压检测输入。
T1CKI			I	ST	Timer1 时钟输入。
$\overline{SS}$			I	ST	SPI 从选择输入。
CTMUI					用于 C2INB 的 CTMU 脉冲发生器充电器的输入。

图注： CMOS = CMOS 兼容输入或输出　　I^2C™ = I^2C/SMBus 输入缓冲器
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-4： PIC18F2XK80 I/O 说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	QFN	SSOP/ SPDIP/ SOIC			
RB0/AN10/C1INA/FLT0/ INT0	18	21			PORTB 是双向 I/O 端口。
RB0			I/O	ST/ CMOS	数字 I/O。
AN10			I	Analog	模拟输入 10。
C1INA			I	Analog	比较器 1 的输入 A。
FLT0			I	ST	ECCP1 的增强型 PWM 故障输入。
INT0			I	ST	外部中断 0。
RB1/AN8/C1INB/P1B/ CTDIN/INT1	19	22			
RB1			I/O	ST/ CMOS	数字 I/O。
AN8			I	Analog	模拟输入 8。
C1INB			I	Analog	比较器 1 的输入 B。
P1B			O	CMOS	增强型 PWM1 的输出 B。
CTDIN			I	ST	CTMU 脉冲延时输入。
INT1			I	ST	外部中断 1。
RB2/CANTX/C1OUT/ P1C/CTED1/INT2	20	23			
RB2			I/O	ST/ CMOS	数字 I/O。
CANTX			O	CMOS	CAN 总线发送。
C1OUT			O	CMOS	比较器 1 的输出。
P1C			O	CMOS	增强型 PWM1 的输出 C。
CTED1			I	ST	CTMU 边沿 1 输入。
INT2			I	ST	外部中断 2。
RB3/CANRX/C2OUT/ P1D/CTED2/INT3	21	24			
RB3			I/O	ST/ CMOS	数字 I/O。
CANRX			I	ST	CAN 总线接收。
C2OUT			O	CMOS	比较器 2 的输出。
P1D			O	CMOS	增强型 PWM1 的输出 D。
CTED2			I	ST	CTMU 边沿 2 输入。
INT3			I	ST	外部中断 3。

图注： CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C™ = I^2C/SMBus 输入缓冲器
Analog = 模拟输入
O = 输出

表 1-4： PIC18F2XK80 I/O 说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	QFN	SSOP/ SPDIP/ SOIC			
RB4/AN9/C2INA/ECCP1/ P1A/CTPLS/KBI0	22	25			
RB4			I/O	ST/ CMOS	数字 I/O。
AN9			I	Analog	模拟输入 9。
C2INA			I	Analog	比较器 2 的输入 A。
ECCP1			I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM1 输出。
P1A			O	CMOS	增强型 PWM1 的输出 A。
CTPLS			O	ST	CTMU 脉冲发生器输出。
KBI0			I	ST	电平变化中断引脚。
RB5/T0CKI/T3CKI/CCP5/ KBI1	23	26			
RB5			I/O	ST/ CMOS	数字 I/O。
T0CKI			I	ST	Timer0 外部时钟输入。
T3CKI			I	ST	Timer3 外部时钟输入。
CCP5			I/O	ST/ CMOS	捕捉 5 输入 / 比较 5 输出 /PWM5 输出。
KBI1			I	ST	电平变化中断引脚。
RB6/PGC/TX2/CK2/KBI2	24	27			
RB6			I/O	ST/ CMOS	数字 I/O。
PGC			I	ST	在线调试器和 ICSP™ 编程时钟输入引脚。
TX2			O	CMOS	EUSART 异步发送。
CK2			I/O	ST	EUSART 同步时钟 （见相关的 RX2/DT2 引脚信息）。
KBI2			I	ST	电平变化中断引脚。
RB7/PGD/T3G/RX2/DT2/ KBI3	25	28			
RB7			I/O	ST/ CMOS	数字 I/O。
PGD			I/O	ST	在线调试器和 ICSP 编程数据引脚。
T3G			I	ST	Timer3 外部时钟门控输入。
RX2			I	ST	EUSART 异步接收。
DT2			I/O	ST	EUSART 同步数据 （见相关的 TX2/CK2 引脚信息）。
KBI3			I	ST	电平变化中断引脚。

图注： CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C™ = I^2C/SMBus 输入缓冲器
Analog = 模拟输入
O = 输出

表 1-4： PIC18F2XK80 I/O 说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	QFN	SSOP/ SPDIP/ SOIC			
					PORTC 是双向 I/O 端口。
RC0/SOSCO/SCLKI	8	11			
RC0			I/O	ST/ CMOS	数字 I/O。
SOSCO			O	ST	SOSC 振荡器输出。
SCLKI			I	ST	数字 SOSC 输入。
RC1/SOSCI	9	12			
RC1			I/O	ST/ CMOS	数字 I/O。
SOSCI			I	CMOS	SOSC 振荡器输入。
RC2/T1G/CCP2	10	13			
RC2			I/O	ST/ CMOS	数字 I/O。
T1G			I	ST	Timer1 外部时钟门控输入。
CCP2			I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
RC3/REFO/SCL/SCK	11	14			
RC3			I/O	ST/ CMOS	数字 I/O。
REFO			O	—	参考时钟输出。
SCL			I/O	I^2C	I^2C 模式的同步串行时钟输入 / 输出。
SCK			I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
RC4/SDA/SDI	12	15			
RC4			I/O	ST/ CMOS	数字 I/O。
SDA			I/O	I^2C	I^2C 数据输入 / 输出。
SDI			I	ST	SPI 数据输入。
RC5/SDO	13	16			
RC5			I/O	ST/ CMOS	数字 I/O。
SDO			O	CMOS	SPI 数据输出。
RC6/CANTX/TX1/CK1/ CCP3	14	17			
RC6			I/O	ST/ CMOS	数字 I/O。
CANTX			O	CMOS	CAN 总线发送。
TX1			O	CMOS	EUSART 异步发送。
CK1			I/O	ST	EUSART 同步时钟。（见相关的 RX1/DT1 引脚信息。）
CCP3			I/O	ST/ CMOS	捕捉 3 输入 / 比较 3 输出 /PWM3 输出。

图注： CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C™ = I^2C/SMBus 输入缓冲器
Analog = 模拟输入
O = 输出

表 1-4： PIC18F2XK80 I/O 说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	QFN	SSOP/ SPDIP/ SOIC			
RC7/CANRX/RX1/DT1/ CCP4	15	18			
RC7			I/O	ST/ CMOS	数字 I/O。
CANRX			I	ST	CAN 总线接收。
RX1			I	ST	EUSART 异步接收。
DT1			I/O	ST	EUSART 同步数据 （见相关的 TX2/CK2 引脚信息）。
CCP4			I/O	ST CMOS	捕捉 4 输入 / 比较 4 输出 /PWM4 输出。
VSS	5	8	P		
VSS					逻辑和 I/O 引脚的参考地。
VSS	16	19			
VSS					逻辑和 I/O 引脚的参考地。
VDDCORE/VCAP	3	6	P		
VDDCORE					外部滤波电容连接。
VCAP					外部滤波电容连接。
VDD	17	20	P		
VDD					逻辑和 I/O 引脚的正电源。

图注： CMOS = CMOS 兼容输入或输出　　I^2C™ = I^2C/SMBus 输入缓冲器
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-5： PIC18F4XK80 I/O 引脚说明

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
MCLR/RE3	1	18			
MCLR			I	ST	主复位 （输入）或编程电压 （输入）。此引脚为低电平有效的器件复位输入端。
RE3			I	ST	仅用作输入的通用引脚。
OSC1/CLKIN/RA7	13	30			
OSC1			I	ST	晶振输入。
CLKIN			I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能相关联。（见相关的 OSC1/CLKI 和 OSC2/CLKO 引脚信息。）
RA7			I/O	ST/ CMOS	通用 I/O 引脚。
OSC2/CLKOUT/RA6	14	31			
OSC2			O	—	晶振输出。在晶振模式下，该引脚与晶振或谐振器相连。
CLKOUT			O	—	在某种振荡器模式下， OSC2 引脚输出 CLKO 信号，其频率是 OSC1 引脚上信号频率的 1/4，该频率等于指令周期的倒数。
RA6			I/O	ST/ CMOS	通用 I/O 引脚。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-5： PIC18F4XK80 I/O 引脚说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
					PORTA 是双向 I/O 端口。
RA0/CVREF/AN0/ULPWU	2	19			
RA0			I/O	ST/ CMOS	通用 I/O 引脚。
CVREF			O	Analog	比较器参考电压输出。
AN0			I	Analog	模拟输入 0。
ULPWU			I	Analog	超低功耗唤醒输入。
RA1/AN1/C1INC	3	20			
RA1			I/O	ST/ CMOS	数字 I/O。
AN1			I	Analog	模拟输入 1。
C1INC			I	Analog	比较器 1 的输入 C。
RA2/VREF-/AN2/C2INC	4	21			
RA2			I/O	ST/ CMOS	数字 I/O。
VREF-			I	Analog	A/D 参考电压 （低电压）输入。
AN2			I	Analog	模拟输入 2。
C2INC			I	Analog	比较器 2 的输入 C。
RA3/VREF+/AN3	5	22			
RA3			I/O	ST/ CMOS	数字 I/O。
VREF+			I	Analog	A/D 参考电压 （高电压）输入。
AN3			I	Analog	模拟输入 3。
RA5/AN4/HLVDIN/T1CKI/ $\overline{SS}$	7	24			
RA5			I/O	ST/ CMOS	数字 I/O。
AN4			I	Analog	模拟输入 4。
HLVDIN			I	Analog	高 / 低压检测输入。
T1CKI			I	ST	Timer1 时钟输入。
$\overline{SS}$			I	ST	SPI 从选择输入。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-5： PIC18F4XK80 I/O 引脚说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
					PORTB 是双向 I/O 端口。
RB0/AN10/FLT0/INT0	33	8			
RB0			I/O	ST/ CMOS	数字 I/O。
AN10			I	Analog	模拟输入 10。
FLT0			I	ST	ECCP1 的增强型 PWM 故障输入。
INT0			I	ST	外部中断 0。
RB1/AN8/CTDIN/INT1	34	9			
RB1			I/O	ST/ CMOS	数字 I/O。
AN8			I	Analog	模拟输入 8。
CTDIN			I	ST	CTMU 脉冲延时输入。
INT1			I	ST	外部中断 1。
RB2/CANTX/CTED1/ INT2	35	10			
RB2			I/O	ST/ CMOS	数字 I/O。
CANTX			O	CMOS	CAN 总线发送。
CTED1			I	ST	CTMU 边沿 1 输入。
INT2			I	ST	外部中断 2。
RB3/CANRX/CTED2/ INT3	36	11			
RB3			I/O	ST/ CMOS	数字 I/O。
CANRX			I	ST	CAN 总线接收。
CTED2			I	ST	CTMU 边沿 2 输入。
INT3			I	ST	外部中断 3。
RB4/AN9/CTPLS/KBI0	37	14			
RB4			I/O	ST/ CMOS	数字 I/O。
AN9			I	Analog	模拟输入 9。
CTPLS			O	ST	CTMU 脉冲发生器输出。
KBI0			I	ST	电平变化中断引脚。
RB5/T0CKI/T3CKI/CCP5/ KBI1	38	15			
RB5			I/O	ST/ CMOS	数字 I/O。
T0CKI			I	ST	Timer0 外部时钟输入。
T3CKI			I	ST	Timer3 外部时钟输入。
CCP5			I/O	ST	捕捉 5 输入 / 比较 5 输出 /PWM5 输出。
KBI1			I	ST	电平变化中断引脚。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-5： PIC18F4XK80 I/O 引脚说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
RB6/PGC/KBI2	39	16			
RB6			I/O	ST/ CMOS	数字 I/O。
PGC			I	ST	在线调试器和 ICSP™ 编程时钟输入引脚。
KBI2			I	ST	电平变化中断引脚。
RB7/PGD/T3G/KBI3	40	17			
RB7			I/O	ST/ CMOS	数字 I/O。
PGD			I/O	ST	在线调试器和 ICSP™ 编程数据引脚。
T3G			I	ST	Timer3 外部时钟门控输入。
KBI3			I	ST	电平变化中断引脚。

图注： I^2C™ = I^2C/SMBus 输入缓冲器
ST = 带 CMOS 电平的施密特触发器输入
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出

表 1-5： PIC18F4XK80 I/O 引脚说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
					PORTC 是双向 I/O 端口。
RC0/SOSCO/SCLKI	15	32			
RC0			I/O	ST/ CMOS	数字 I/O。
SOSCO			O	ST	SOSC 振荡器输出。
SCLKI			I	ST	数字 SOSC 输入。
RC1/SOSCI	16	35			
RC1			I/O	ST/ CMOS	数字 I/O。
SOSCI			I	CMOS	SOSC 振荡器输入。
RC2/T1G/CCP2	17	36			
RC2			I/O	ST/ CMOS	数字 I/O。
T1G			I	ST	Timer1 外部时钟门控输入。
CCP2			I/O	ST/ CMOS	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
RC3/REFO/SCL/SCK	18	37			
RC3			I/O	ST/ CMOS	数字 I/O。
REFO			O	CMOS	参考时钟输出。
SCL			I/O	I^2C	I^2C 模式的同步串行时钟输入 / 输出。
SCK			I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
RC4/SDA/SDI	23	42			
RC4			I/O	ST/ CMOS	数字 I/O。
SDA			I/O	I^2C	I^2C 数据输入 / 输出。
SDI			I	ST	SPI 数据输入。
RC5/SDO	24	43			
RC5			I/O	ST/ CMOS	数字 I/O。
SDO			O	CMOS	SPI 数据输出。
RC6/CANTX/TX1/CK1/ CCP3	25	44			
RC6			I/O	ST/ CMOS	数字 I/O。
CANTX			O	CMOS	CAN 总线发送。
TX1			O	CMOS	EUSART 同步发送。
CK1			I/O	ST	EUSART 同步时钟 （见相关的 RX2/DT2 引脚信息）。
CCP3			I/O	ST	捕捉 3 输入 / 比较 3 输出 /PWM3 输出。

图注： I^2C™ = I^2C/SMBus 输入缓冲器
ST = 带 CMOS 电平的施密特触发器输入
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出

表 1-5： PIC18F4XK80 I/O 引脚说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
RC7/CANRX/RX1/DT1/ CCP4	26	1			
RC7			I/O	ST/ CMOS	数字 I/O。
CANRX			I	ST	CAN 总线接收。
RX1			I	ST	EUSART 异步接收。
DT1			I/O	ST	EUSART 同步数据 （见相关的 TX2/CK2 引脚信息）。
CCP4			I/O	ST	捕捉 4 输入 / 比较 4 输出 /PWM4 输出。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-5： PIC18F4XK80 I/O 引脚说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
					PORTD 是双向 I/O 端口。
RD0/C1INA/PSP0	19	38			
RD0			I/O	ST/ CMOS	数字 I/O。
C1INA			I	Analog	比较器 1 的输入 A。
PSP0			I/O	ST/ CMOS	并行从端口数据。
RD1/C1INB/PSP1	20	39			
RD1			I/O	ST/ CMOS	数字 I/O。
C1INB			I	Analog	比较器 1 的输入 B。
PSP1			I/O	ST/ CMOS	并行从端口数据。
RD2/C2INA/PSP2	21	40			
RD2			I/O	ST/ CMOS	数字 I/O。
C2INA			I	Analog	比较器 2 的输入 A。
PSP2			I/O	ST/ CMOS	并行从端口数据。
RD3/C2INB/CTMUI/ PSP3	22	41			
RD3			I/O	ST/ CMOS	数字 I/O。
C2INB			I	Analog	比较器 2 的输入 B。
CTMUI					用于 C2INB 的 CTMU 脉冲发生器充电器的输入。
PSP3			I/O	ST/ CMOS	并行从端口数据。
RD4/ECCP1/P1A/PSP4	27	2			
RD4			I/O	ST/ CMOS	数字 I/O。
ECCP1			I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM1 输出。
P1A			O	CMOS	增强型 PWM1 的输出 A。
PSP4			I/O	ST/ CMOS	并行从端口数据。
RD5/P1B/PSP5	28	3			
RD5			I/O	ST/ CMOS	数字 I/O。
P1B			O	CMOS	增强型 PWM1 的输出 B。
PSP5			I/O	ST/ CMOS	并行从端口数据。

图注： I^2C™ = I^2C/SMBus 输入缓冲器
ST = 带 CMOS 电平的施密特触发器输入
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出

表 1-5： PIC18F4XK80 I/O 引脚说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
RD6/TX2/CK2/P1C/PSP6	29	4			
RD6			I/O	ST/ CMOS	数字 I/O。
TX2			I	ST	EUSART 异步发送。
CK2			I/O	ST	EUSART 同步时钟（见相关的 RX2/DT2 引脚信息）。
P1C			O	CMOS	增强型 PWM1 的输出 C。
PSP6			I/O	ST/ CMOS	并行从端口数据。
RD7/RX2/DT2/P1D/PSP7	30	5			
RD7			I/O	ST/ CMOS	数字 I/O。
RX2			I	ST	EUSART 异步接收。
DT2			I/O	ST	EUSART 同步数据（见相关的 TX2/CK2 引脚信息）。
P1D			O	CMOS	增强型 PWM1 的输出 D。
PSP7			I/O	ST/ CMOS	并行从端口数据。
RE0/AN5/$\overline{RD}$	8	25			
RE0			I/O	ST/ CMOS	数字 I/O。
AN5			I	Analog	模拟输入 5。
$\overline{RD}$			I	ST	并行从端口读选通。
RE1/AN6/C1OUT/$\overline{WR}$	9	26			
RE1			I/O	ST/ CMOS	数字 I/O。
AN6			I	Analog	模拟输入 6。
C1OUT			O	CMOS	比较器 1 的输出。
$\overline{WR}$			I	ST	并行从端口写选通。
RE2/AN7/C2OUT/$\overline{CS}$	10	27			
RE2			I/O	ST/ CMOS	数字 I/O。
AN7			I	Analog	模拟输入 7。
C2OUT			O	CMOS	比较器 2 的输出。
$\overline{CS}$			I	ST	并行从端口片选。
RE3					见 $\overline{MCLR}$/RE3 引脚信息。

图注： I^2C™ = I^2C/SMBus 输入缓冲器
ST = 带 CMOS 电平的施密特触发器输入
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出

表 1-5： PIC18F4XK80 I/O 引脚说明 （续）

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP	QFN/ TQFP			
VSS	12	29	P		
VSS					逻辑和 I/O 引脚的参考地。
VSS	31	6			
VSS					逻辑和 I/O 引脚的参考地。
VDDCORE/VCAP	6	23	P		
VDDCORE					外部滤波电容连接。
VCAP					外部滤波电容连接。
VDD	11	28	P		
VDD					逻辑和 I/O 引脚的正电源。
VDD	32	7	P		
VDD					逻辑和 I/O 引脚的正电源。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
$\overline{\text{MCLR}}$/RE3	28			
$\overline{\text{MCLR}}$		I	ST	主复位（输入）或编程电压（输入）。此引脚为低电平有效的器件复位输入端。
RE3		I	ST	仅用作输入的通用引脚。
OSC1/CLKIN/RA7	46			
OSC1		I	ST	晶振输入。
CLKIN		I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能相关联。（见相关的 OSC1/CLKI 和 OSC2/CLKO 引脚信息。）
RA7		I/O	ST/CMOS	通用 I/O 引脚。
OSC2/CLKOUT/RA6	47			
OSC2		O	—	晶振输出。在晶振模式下，该引脚与晶振或谐振器相连。
CLKOUT		O	—	在某种振荡器模式下，OSC2 引脚输出 CLKO 信号，其频率是 OSC1 引脚上信号频率的 1/4，该频率等于指令周期的倒数。
RA6		I/O	ST/CMOS	通用 I/O 引脚。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
RA0/CVREF/AN0/ ULPWU	29			PORTA 是双向 I/O 端口。
RA0		I/O	ST/ CMOS	通用 I/O 引脚。
CVREF		O	Analog	比较器参考电压输出。
AN0		I	Analog	模拟输入 0。
ULPWU		I	Analog	超低功耗唤醒输入。
RA1/AN1/C1INC	30			
RA1		I/O	ST/ CMOS	数字 I/O。
AN1		I	Analog	模拟输入 1。
C1INC		I	Analog	比较器 1 的输入 C。
RA2/VREF-/AN2/C2INC	31			
RA2		I/O	ST/ CMOS	数字 I/O。
VREF-		I	Analog	A/D 参考电压 （低电压）输入。
AN2		I	Analog	模拟输入 2。
C2INC		I	Analog	比较器 2 的输入 C。
RA3/VREF+/AN3	32			
RA3		I/O	ST/ CMOS	数字 I/O。
VREF+		I	Analog	A/D 参考电压 （高电压）输入。
AN3		I	Analog	模拟输入 3。
RA5/AN4/HLVDIN/ T1CKI/$\overline{SS}$	34			
RA5		I/O	ST/ CMOS	数字 I/O。
AN4		I	Analog	模拟输入 4。
HLVDIN		I	Analog	高 / 低压检测输入。
T1CKI		I	ST	Timer1 时钟输入。
$\overline{SS}$		I	ST	SPI 从选择输入。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
				PORTB 是双向 I/O 端口。
RB0/AN10/FLT0/INT0	13			
RB0		I/O	ST/CMOS	数字 I/O。
AN10		I	Analog	模拟输入 10。
FLT0		I	ST	ECCP1 的增强型 PWM 故障输入。
INT0		I	ST	外部中断 0。
RB1/AN8/CTDIN/INT1	14			
RB1		I/O	ST/CMOS	数字 I/O。
AN8		I	Analog	模拟输入 8。
CTDIN		I	ST	CTMU 脉冲延时输入。
INT1		I	ST	外部中断 1。
RB2/CANTX/CTED1/INT2	15			
RB2		I/O	ST/CMOS	数字 I/O。
CANTX		O	CMOS	CAN 总线发送。
CTED1		I	ST	CTMU 边沿 1 输入。
INT2		I	ST	外部中断 2。
RB3/CANRX/CTED2/INT3	16			
RB3		I/O	ST/CMOS	数字 I/O。
CANRX		I	ST	CAN 总线接收。
CTED2		I	ST	CTMU 边沿 2 输入。
INT3		I	ST	外部中断 3。
RB4/AN9/CTPLS/KBI0	20			
RB4		I/O	ST/CMOS	数字 I/O。
AN9		I	Analog	模拟输入 9。
CTPLS		O	ST	CTMU 脉冲发生器输出。
KBI0		I	ST	电平变化中断引脚。
RB5/T0CKI/T3CKI/CCP5/KBI1	21			
RB5		I/O	ST/CMOS	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟输入。
T3CKI		I	ST	Timer3 外部时钟输入。
CCP5		I/O	ST/CMOS	捕捉 5 输入 / 比较 5 输出 /PWM5 输出。
KBI1		I	ST	电平变化中断引脚。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
RB6/PGC/KBI2	22			
RB6		I/O	ST/ CMOS	数字 I/O。
PGC		I	ST	在线调试器和 ICSP™ 编程时钟输入引脚。
KBI2		I	ST	电平变化中断引脚。
RB7/PGD/T3G/KBI3	23			
RB7		I/O	ST/ CMOS	数字 I/O。
PGD		I/O	ST	在线调试器和 ICSP™ 编程数据引脚。
T3G		I	ST	Timer3 外部时钟门控输入。
KBI3		I	ST	电平变化中断引脚。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
				PORTC 是双向 I/O 端口。
RC0/SOSCO/SCLKI	48			
RC0		I/O	ST/CMOS	数字 I/O。
SOSCO		O	ST	SOSC 振荡器输出。
SCLKI		I	ST	数字 SOSC 输入。
RC1/SOSCI	49			
RC1		I/O	ST/CMOS	数字 I/O。
SOSCI		I	CMOS	SOSC 振荡器输入。
RC2/T1G/CCP2	50			
RC2		I/O	ST/CMOS	数字 I/O。
T1G		I	ST	Timer1 外部时钟门控输入。
CCP2		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
RC3/REFO/SCL/SCK	51			
RC3		I/O	ST/CMOS	数字 I/O。
REFO		O	CMOS	参考时钟输出。
SCL		I/O	I^2C	I^2C 模式的同步串行时钟输入 / 输出。
SCK		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
RC4/SDA/SDI	62			
RC4		I/O	ST/CMOS	数字 I/O。
SDA		I/O	I^2C	I^2C 数据输入 / 输出。
SDI		I	ST	SPI 数据输入。
RC5/SDO	63			
RC5		I/O	ST/CMOS	数字 I/O。
SDO		O	CMOS	SPI 数据输出。
RC6/CCP3	64			
RC6		I/O	ST/CMOS	数字 I/O。
CCP3		I/O	ST/CMOS	捕捉 3 输入 / 比较 3 输出 /PWM3 输出。
RC7/CCP4	1			
RC7		I/O	ST/CMOS	数字 I/O。
CCP4		I/O	ST/CMOS	捕捉 4 输入 / 比较 4 输出 /PWM4 输出。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
				PORTD 是双向 I/O 端口。
RD0/C1INA/PSP0	54			
RD0		I/O	ST/CMOS	数字 I/O。
C1INA		I	Analog	比较器 1 的输入 A。
PSP0		I/O	ST/CMOS	并行从端口数据。
RD1/C1INB/PSP1	55			
RD1		I/O	ST/CMOS	数字 I/O。
C1INB		I	Analog	比较器 1 的输入 B。
PSP1		I/O	ST/CMOS	并行从端口数据。
RD2/C2INA/PSP2	58			
RD2		I/O	ST/CMOS	数字 I/O。
C2INA		I	Analog	比较器 2 的输入 A。
PSP2		I/O	ST/CMOS	并行从端口数据。
RD3/C2INB/CTMUI/PSP3	59			
RD3		I/O	ST/CMOS	数字 I/O。
C2INB		I	Analog	比较器 2 的输入 B。
CTMUI		I	CMOS	用于 C2INB 的 CTMU 脉冲发生器充电器的输入。
PSP3		I/O	ST/CMOS	并行从端口数据。
RD4/ECCP1/P1A/PSP4	2			
RD4		I/O	ST/CMOS	数字 I/O。
ECCP1		I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM1 输出。
P1A		O	CMOS	增强型 PWM1 的输出 A。
PSP4		I/O	ST/CMOS	并行从端口数据。
RD5/P1B/PSP5	3			
RD5		I/O	ST/CMOS	数字 I/O。
P1B		O	CMOS	增强型 PWM1 的输出 B。
PSP5		I/O	ST/CMOS	并行从端口数据。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　Analog = 模拟输入
I = 输入　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
RD6/P1C/PSP6	4			
RD6		I/O	ST/CMOS	数字 I/O。
P1C		O	CMOS	增强型 PWM1 的输出 C。
PSP6		I/O	ST/CMOS	并行从端口数据。
RD7/P1D/PSP7	5			
RD7		I/O	ST/CMOS	数字 I/O。
P1D		O	CMOS	增强型 PWM1 的输出 D。
PSP7		I/O	ST/CMOS	并行从端口数据。

图注： I^2C™ = I^2C/SMBus 输入缓冲器
ST = 带 CMOS 电平的施密特触发器输入
I = 输入
P = 电源
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
				PORTE 是双向 I/O 端口。
RE0/AN5/$\overline{RD}$	37			
RE0		I/O	ST/CMOS	数字 I/O。
AN5		I	Analog	模拟输入 5。
$\overline{RD}$		I	ST	并行从端口读选通。
RE1/AN6/C1OUT/$\overline{WR}$	38			
RE1		I/O	ST/CMOS	数字 I/O。
AN6		I	Analog	模拟输入 6。
C1OUT		O	CMOS	比较器 1 的输出。
$\overline{WR}$		I	ST	并行从端口写选通。
RE2/AN7/C2OUT/$\overline{CS}$	39			
RE2		I/O	ST/CMOS	数字 I/O。
AN7		I	Analog	模拟输入 7。
C2OUT		O	CMOS	比较器 2 的输出。
$\overline{CS}$		I	ST	并行从端口片选。
RE3				见 $\overline{MCLR}$/RE3 引脚信息。
RE4/CANRX	27			
RE4		I/O	ST/CMOS	数字 I/O。
CANRX		I	ST	CAN 总线接收。
RE5/CANTX	24			
RE5		I/O	ST/CMOS	数字 I/O。
CANTX		O	CMOS	CAN 总线发送。
RE6/RX2/DT2	60			
RE6		I/O	ST/CMOS	数字 I/O。
RX2		I	ST	EUSART 异步接收。
DT2		I/O	ST	EUSART 同步数据 （见相关的 TX2/CK2 引脚信息）。
RE7/TX2/CK2	61			
RE7		I/O	ST/CMOS	数字 I/O。
TX2		O	CMOS	EUSART 异步发送。
CK2		I/O	ST	EUSART 同步时钟 （见相关的 RX2/DT2 引脚信息）。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
				PORTF 是双向 I/O 端口。
RF0/MDMIN	17			
RF0		I/O	ST/CMOS	数字 I/O。
MDMIN		I	CMOS	调制器源输入。
RF1	19			
RF1		I/O	ST/CMOS	数字 I/O。
RF2/MDCIN1	35			
RF2		I/O	ST/CMOS	数字 I/O。
MDCIN1		I	ST	调制器载波输入 1。
RF3	36			
RF3		I/O	ST/CMOS	数字 I/O。
RF4/MDCIN2	44			
RF4		I/O	ST/CMOS	数字 I/O。
MDCIN2		I	ST	调制器载波输入 2。
RF5	45			
RF5		I/O	ST/CMOS	数字 I/O。
RF6/MDOUT	52			
RF6		I/O	ST/CMOS	数字 I/O。
MDOUT		O	CMOS	调制器输出。
RF7	53			
RF7		I/O	ST/CMOS	数字 I/O。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
				PORTG 是双向 I/O 端口。
RG0/RX1/DT1	6			
RG0		I/O	ST/ CMOS	数字 I/O。
RX1		I	ST	EUSART 异步接收。
DT1		I/O	ST	EUSART 同步数据 （见相关的 TX2/CK2 引脚信息）。
RG1/CANTX2	7			
RG1		I/O	ST/ CMOS	数字 I/O。
CANTX2		O	CMOS	CAN 总线互补发送输出或 CAN 总线时间时钟。
RG2/T3CKI	11			
RG2		I/O	ST/ CMOS	数字 I/O。
T3CKI		I	ST	Timer3 时钟输入。
RG3/TX1/CK1	12			
RG3		I/O	ST/ CMOS	数字 I/O。
TX1		O	CMOS	EUSART 异步发送。
CK1		I/O	ST	EUSART 同步时钟 （见相关的 RX2/DT2 引脚信息）。
RG4/T0CKI	18			
RG4		I/O	ST/ CMOS	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟输入。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

表 1-6： PIC18F6XK80 I/O 引脚说明 （续）

引脚名称	引脚编号	引脚类型	缓冲器类型	说明
VSS	8		P	
VSS				逻辑和 I/O 引脚的参考地。
VSS	26		P	
VSS				逻辑和 I/O 引脚的参考地。
AVSS	42		P	
AVSS				模拟模块的参考地。
VSS	43		P	
VSS				逻辑和 I/O 引脚的参考地。
VSS	56		P	
VSS				逻辑和 I/O 引脚的参考地。
AVDD	9		P	
AVDD				模拟模块的正电源。
VDD	10		P	
VDD				逻辑和 I/O 引脚的正电源。
VDD	25		P	
VDD				逻辑和 I/O 引脚的正电源。
VDDCORE/VCAP	33		P	
VDDCORE				外部滤波电容连接。
VCAP				外部滤波电容连接。
AVDD	40		P	
AVDD				模拟模块的正电源。
VDD	41		P	
VDD				逻辑和 I/O 引脚的正电源。
VDD	57		P	
VDD				逻辑和 I/O 引脚的正电源。

图注： I^2C™ = I^2C/SMBus 输入缓冲器　　CMOS = CMOS 兼容输入或输出
ST = 带 CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源

注：

2.0 PIC18FXXKXX 单片机入门指南

2.1 基本连接要求

在开始使用 PIC18F66K80 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 ”**）
- 所有 AVDD 和 AVSS 引脚（不论是否使用模拟器件功能）
  （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 ”**）

如果在最终应用中使用了以下引脚，则也必须连接它们：

- PGC/PGD 引脚，用于进行在线串行编程（ICSP™）和调试 （见**第 2.5 节 “ICSP 引脚 ”**）
- OSCI 和 OSCO 引脚 （使用外部振荡器源时）
  （见**第 2.6 节 “ 外部振荡器引脚 ”**）

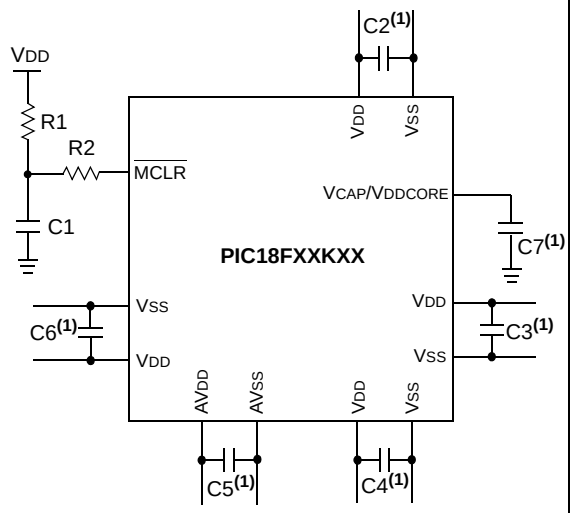
此外，可能还需要连接以下引脚：

- VREF+/VREF- 引脚 （在实现模拟模块的外部参考电压时使用）

注：	不论是否使用任何模拟模块，都必须始终连接 AVDD 和 AVSS 引脚。

图 2-1 中显示了最低限度的连接要求。

图 2-1： 建议的最低限度连接



关键 （所有值均为建议值）：

C1 至 C6：0.1 µF， 20V 陶瓷电容

R1：10 kΩ

R2：100Ω 至 470Ω

注 1： 所给出的示例针对的是具有 5 个 VDD/VSS 和 AVDD/AVSS 引脚对的 PIC18F 器件。其他器件的引脚对可能增多或减少，请相应地调整去耦电容的数量。

2.2 电源引脚

2.2.1 去耦电容

需要在每对电源引脚（例如，VDD/VSS 和 AVDD/AVSS）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：**建议使用参数为0.1 µF（100 nF）、10-20V的电容。该电容应具有低ESR，谐振频率为 200 MHz 或更高。建议使用陶瓷电容。
- **在印制电路板上的放置：**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。如果空间受到限制，可以使用过孔将电容放置在 PCB 的另一层，但请确保从引脚到电容的走线长度不超出 0.25 英寸 （6 毫米）。
- **高频噪声处理：**如果电路板遇到高频噪声（频率高于数十 MHz），则另外添加一个陶瓷电容，与上述去耦电容并联。第二个电容的电容值可以介于 0.001 µF 至 0.01 µF 之间。请将第二个电容放置在靠近每个主去耦电容的位置。在高速电路设计中，需要考虑尽可能靠近电源和接地引脚放置十对这样的电容（例如， 0.1 µF 电容与 0.001 µF 电容并联构成一对）。
- **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是电源链中的第一个元件。同等重要的是尽可能减小电容和电源引脚之间的走线长度，从而降低 PCB 走线电感。

2.2.2 槽路电容

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）使用槽路电容来提供本地电源。槽路电容的电容值应根据连接电源与器件的走线电阻和应用中的器件的最大电流确定。也就是说，选择的槽路电容需要满足器件的可接受电压骤降要求。典型值的范围为 4.7 µF 至 47 µF。

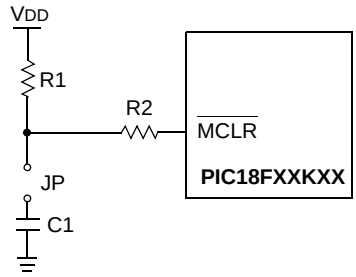
2.3 主复位（$\overline{\text{MCLR}}$）引脚

$\overline{\text{MCLR}}$ 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。如果最终应用中不需要进行编程和调试，则只需直接连接 VDD 即可。添加其他元件有助于提高应用抵抗由于电压骤降导致意外复位的能力。图 2-1 给出了一种典型配置。根据应用的需求，还可以实现其他电路设计。

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，特定电平 （VIH 和 VIL）和快速信号跳变一定不能受到不利影响。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，建议通过使用跳线将电容 C1 与 $\overline{\text{MCLR}}$ 引脚隔离 （图 2-2）。对于正常的运行时操作，可以将跳线放回原处。

与 $\overline{\text{MCLR}}$ 引脚关联的所有元件都应放置在距离该引脚 0.25 英寸 （6 毫米）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将限制任何电流从外部电容C流入 $\overline{\text{MCLR}}$，以避免由于静电放电 （Electrostatic Discharge， ESD）或电过载（Electrical Overstress，EOS）导致 $\overline{\text{MCLR}}$ 引脚损坏。请确保满足 $\overline{\text{MCLR}}$ 引脚 VIH 和 VIL 规范。

2.4 稳压器引脚（VCAP/VDDCORE）

在 PIC18F66K80 系列器件上，使能稳压器时，需要在 VCAP/VDDCORE 引脚上使用低 ESR （< 5Ω）电容，以稳定稳压器的输出电压。VCAP/VDDCORE 引脚一定不能与 VDD 连接，并且必须使用 10 µF 的电容接地。可以使用陶瓷电容或钽电容。表 2-1 列出了一些适用电容的示例。同等规格的电容都可以使用。设计人员可以根据图 2-3 来评估候选器件的 ESR 等效值。

建议走线长度不要超出 0.25 英寸 （6 毫米）。更多信息，请参见**第 31.0 节 “ 电气特性 ”**。

当禁止稳压器时，应将 VCAP/VDDCORE 引脚通过一个 0.1 µF 的电容接地。该电容的特性必须与**第 2.2.1 节**中介绍的 “ 去耦 ” 电容的特性相类似。关于禁止稳压器时的 VDD 要求的详细信息，请参见**第 31.0 节 “ 电气特性 ”** 中的参数 D001。

一些PIC18FXXKXX 系列或某个系列中的一些器件不提供使能或禁止片上稳压器的选项：

- PIC18**LF**XXKXX 器件会永久性地禁止稳压器。

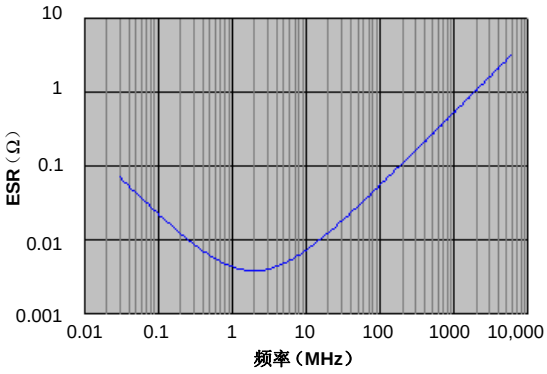
  这些器件需要在 VCAP/VDDCORE 引脚上连接一个 0.1 µF 的电容。这些器件的 VDD 电压必须符合**第 31.0 节 “ 电气特性 ”** 中参数 D001 的 “ 禁止稳压器 ” 规范。

- PIC18FXXKXX 器件会永久性地使能稳压器。

  这些器件需要在 VCAP/VDDCORE 引脚上连接一个 10 µF 的电容。

关于 PIC18F66K80 系列所有成员的详细信息，请参见**第 28.3 节 “ 片上稳压器 ”**。

图 2-3： 建议的 VCAP 的频率与 ESR 性能



注： 在 25°C、 0V 直流偏置条件下测量的典型数据。

2.5 ICSP 引脚

PGC 和 PGD 引脚用于进行在线串行编程 （ICSP™）和调试。建议尽可能减小 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，电阻值为几十欧姆，不要超出 100Ω。

建议不要在 PGC 和 PGD 引脚上连接上拉电阻、串联二极管和电容，因为它们会影响与器件的编程器 / 调试器通信。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，请确保给器件编程的 “ 通信通道选择 ”（即，PGCx/PGDx 引脚）符合 ICSP 到 Microchip 调试器 / 仿真器工具的物理连接。

关于可用的 Microchip 开发工具连接要求的更多信息，请参见**第 30.0 节 “ 开发支持 ”**。

表 2-1：　　适用的等效电容

制造商	部件编号	标称电容	基本容差	额定电压	温度范围
TDK	C3216X7R1C106K	10 µF	±10%	16V	-55 至 125ºC
TDK	C3216X5R1C106K	10 µF	±10%	16V	-55 至 85ºC
Panasonic	ECJ-3YX1C106K	10 µF	±10%	16V	-55 至 125ºC
Panasonic	ECJ-4YB1C106K	10 µF	±10%	16V	-55 至 85ºC
Murata	GRM32DR71C106KA01L	10 µF	±10%	16V	-55 至 125ºC
Murata	GRM31CR61C106KC31L	10 µF	±10%	16V	-55 至 85ºC

2.6　外部振荡器引脚

许多单片机都有至少两个振荡器可供选择：高频主振荡器和低频辅助振荡器（详情请参见**第 3.0 节 “ 振荡器配置 ”**）。

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，电路元件与引脚之间的距离不要超出 0.5 英寸（12 毫米）。负载电容应靠近振荡器自身，位于电路板的同一层。

请在振荡器电路周围使用接地灌铜区，以将其与周围电路隔离。接地灌铜区应与 MCU 地直接连接。不要在接地灌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，请避免在电路板上晶振所在位置的背面有任何走线。

图 2-4 给出了一些布线建议。直插式封装可以采用可完全容纳振荡器引脚的单面布线来处理。对于引脚排列紧密的器件，单面布局则可能无法始终完全地容纳所有引脚和元件。一种适合的解决方案是将含有保护走线的部分连接到反面的接地层。在所有情形中，保护走线都必须回接到地。

在规划应用的走线和 I/O 分配时，需要确保相邻端口引脚和其他邻近振荡器的信号是无害的（即，无高频、无短暂上升和下降时间，以及无其他类似噪声）。

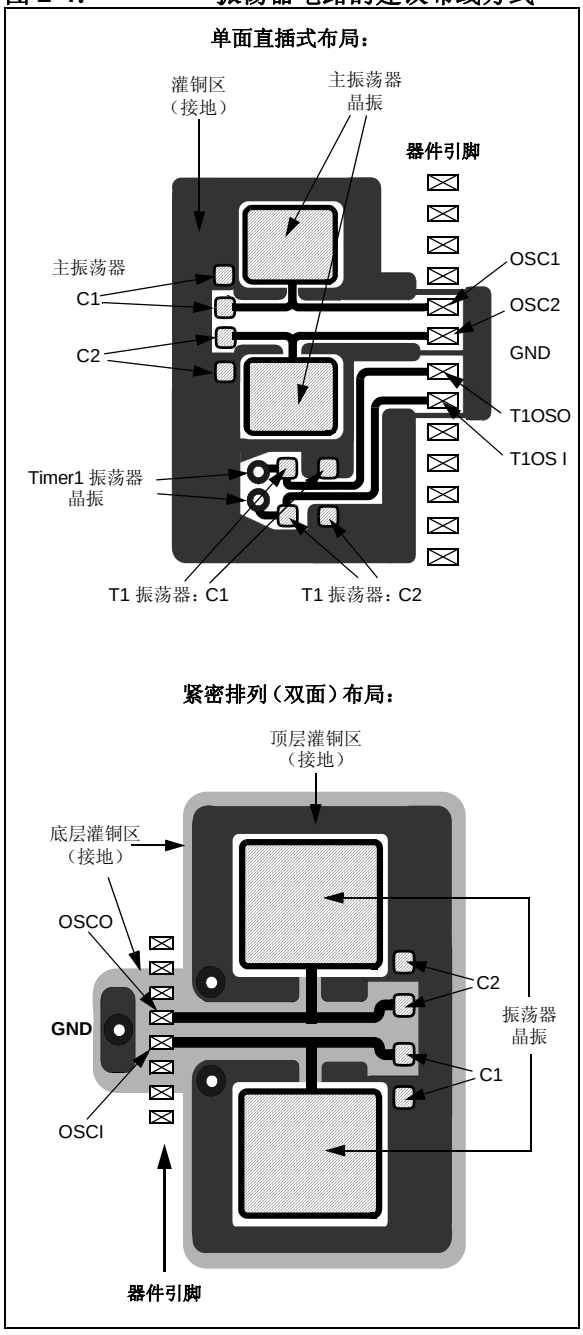
关于振荡器电路的其他信息和设计指南，请参见 Microchip 公司网站（www.microchip.com）上提供的以下应用笔记：

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

2.7 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，将未用引脚通过一个1 kΩ至10 kΩ的电阻与Vss连接，并将输出驱动为逻辑低电平。

图 2-4： 振荡器电路的建议布线方式



注：

3.0 振荡器配置

3.1 振荡器类型

PIC18F66K80 系列器件可以在以下振荡器模式下工作：

- EC 外部时钟， RA6 可用
- ECIO 外部时钟，时钟输出 RA6 （RA6 上输出 FOSC/4 信号）
- HS 高速晶振 / 谐振器
- XT 晶振 / 谐振器
- LP 低功耗晶振
- RC 外部电阻 / 电容， RA6 可用
- RCIO 外部电阻 / 电容，时钟输出 RA6（RA6 上输出 FOSC/4 信号）
- INTIO2 内部振荡器， RA6 和 RA7 均用作 I/O 引脚
- INTIO1 内部振荡器，通过 RA6 引脚输出 FOSC/4 信号， RA7 用作 I/O 引脚

此外，器件还提供了对输入频率处于 4 至 16 MHz 范围内的任意时钟源运行 4xPLL 的选项。

可以通过将 PLLCFG 位（CONFIG1H<4>）或 PLLEN 位（OSCTUNE<6>）置 1 来使能 PLL。

对于 EC 和 HS 模式，可以使用 PLLEN（软件）或 PLLCFG （CONFIG1H<4>）位来使能 PLL。

对于 INTIOx 模式 （HF-INTOSC）：

- 只有 PLLEN 可以使能 PLL（PLLCFG 会被忽略）。
- 当振荡器配置为内部振荡器（FOSC<3:0> = 100x）时，只有在 HF-INTOSC 频率为 4、8 或 16 MHz 时才能使能 PLL。

当 RA6 和 RA7 引脚不用于振荡器功能或 CLKOUT 功能时，它们可用作通用 I/O。

要在使用 EC/HS/XT/LP/RC 作为主振荡器时优化功耗，可以通过配置频率输入范围来产生优化的功耗偏置：

- 低功耗偏置 —— 外部频率小于 160 kHz
- 中等功耗偏置 —— 外部频率介于 160 kHz 和 16 MHz 之间
- 高功耗偏置 —— 外部频率大于 16 MHz

用户可以通过编程 FOSC<3:0> 配置位（CONFIG1H<3:0>）来选择所有这些模式。此外，PIC18F66K80 系列器件还可以在软件控制下或在某些条件下自动在不同时钟源之间进行切换。通过实时管理器件时钟速度而无需复位应用，进一步节省了功耗。图 3-1 显示了 PIC18F66K80 系列器件的时钟源。

对于 HS 和 EC 模式，另外还有一些功耗工作模式，具体取决于工作频率。

HS1 是中等功耗模式，其频率范围为 4 MHz 至 16 MHz。HS2 是高功耗模式，此时的振荡器频率可以为 16 MHz 至 25 MHz。 HS1 和 HS2 通过正确设置 CONFIG1H<3:0> 位来实现。（详情请参见第 464 页上的寄存器 28-2。）

EC 模式具有以下工作模式：

- EC1—— 低功耗，频率范围最高为 160 kHz
- EC2——中等功耗，频率范围为160 kHz至16 MHz
- EC3—— 高功耗，频率范围为 16 MHz 至 64 MHz

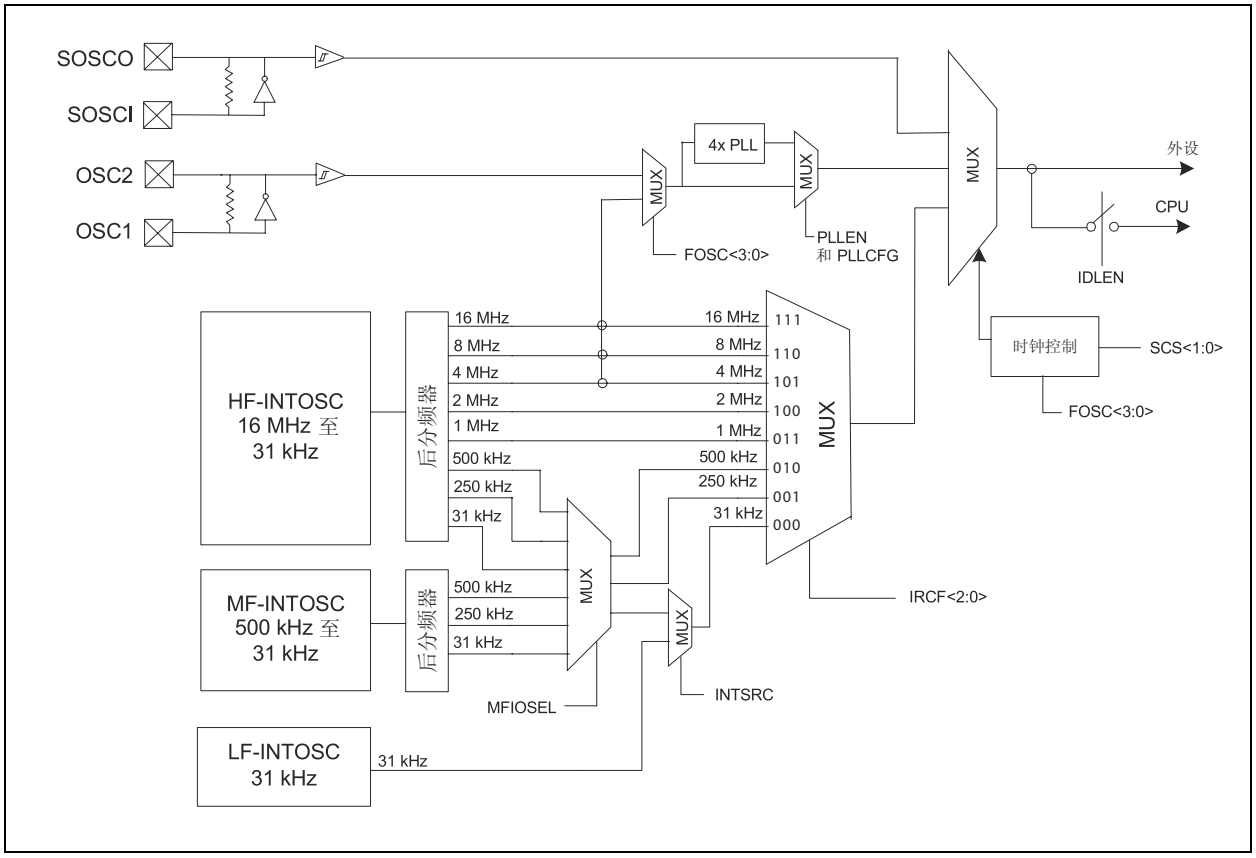
EC1、 EC2 和 EC3 通过正确设置 CONFIG1H<3:0> 来实现。（详情请参见第 464 页上的寄存器 28-2。）

表 3-1 列出了 HS 和 EC 模式的频率范围和 FOSC<3:0> 设置。

表 3-1： HS、 EC、 XT、 LP 和 RC 模式：范围和设置

模式	频率范围	FOSC<3:0> 设置
EC1 （低功耗）	DC-160 kHz	1101
（EC1 和 EC1IO）		1100
EC2 （中等功耗）	160 kHz-16 MHz	1011
（EC2 和 EC2IO）		1010
EC3 （高功耗）	16 MHz-64 MHz	0101
（EC3 和 EC3IO）		0100
HS1 （中等功耗）	4 MHz-16 MHz	0011
HS2 （高功耗）	16 MHz-25 MHz	0010
XT	100 kHz-4 MHz	0001
LP	31.25 kHz	0000
RC （外部）	0-4 MHz	001x
INTIO	32 kHz-16 MHz	100x （以及 OSCCON 和 OSCCON2）

图 3-1： PIC18F66K80 系列时钟框图



3.2 控制寄存器

OSCCON 寄存器（寄存器 3-1）控制器件时钟操作的主要方面。它选择要使用的振荡器类型、要调用的功耗管理模式以及 INTOSC 时钟源的输出频率。它还提供振荡器的状态。

OSCTUNE 寄存器（寄存器 3-3）控制内部振荡器模块的调节和操作。它还实现了 PLLEN 位，该位用于控制锁相环（Phase Locked Loop， PLL）的操作（见**第 3.5.3 节 “PLL 倍频器 ”**）。

寄存器 3-1： OSCCON：振荡器控制寄存器

R/W-0	R/W-1	R/W-0	R/W-0	R(1)	R-0	R/W-0	R/W-0
IDLEN	IRCF2(2)	IRCF1(2)	IRCF0(2)	OSTS	HFIOFS	SCS1(4)	SCS0(4)
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 **IDLEN：**空闲使能位
1 = 执行 SLEEP 指令后器件进入空闲模式
0 = 执行 SLEEP 指令后器件进入休眠模式

bit 6-4 **IRCF<2:0>：**内部振荡器频率选择位 (2)
111 = 使用 HF-INTOSC 输出频率 （16 MHz）
110 = 使用 HF-INTOSC/2 输出频率 （8 MHz，默认）
101 = 使用 HF-INTOSC/4 输出频率 （4 MHz）
100 = 使用 HF-INTOSC/8 输出频率 （2 MHz）
011 = 使用 HF-INTOSC/16 输出频率 （1 MHz）
如果 INTSRC = 0 且 MFIOSEL = 0： (3,5)
010 = 使用 HF-INTOSC/32 输出频率 （500 kHz）
001 = 使用 HF-INTOSC/64 输出频率 （250 kHz）
000 = 使用 LF-INTOSC 输出频率 （31.25 kHz） (6)
如果 INTSRC = 1 且 MFIOSEL = 0： (3,5)
010 = 使用 HF-INTOSC/32 输出频率 （500 kHz）
001 = 使用 HF-INTOSC/64 输出频率 （250 kHz）
000 = 使用 HF-INTOSC/512 输出频率 （31.25 kHz）
如果 INTSRC = 0 且 MFIOSEL = 1： (3,5)
010 = 使用 MF-INTOSC 输出频率 （500 kHz）
001 = 使用 MF-INTOSC/2 输出频率 （250 kHz）
000 = 使用 LF-INTOSC 输出频率 （31.25 kHz） (6)
如果 INTSRC = 1 且 MFIOSEL = 1： (3,5)
010 = 使用 MF-INTOSC 输出频率 （500 kHz）
001 = 使用 MF-INTOSC/2 输出频率 （250 kHz）
000 = 使用 MF-INTOSC/16 输出频率 （31.25 kHz）

bit 3 **OSTS：**振荡器起振定时器延时状态位 (1)
1 = 振荡器起振定时器（Oscillator Start-up Timer，OST）延时已结束；由 FOSC<3:0> 定义的主振荡器正在运行
0 = 振荡器起振定时器（OST）正在延时；主振荡器未就绪——器件正在使用内部振荡器（HF-INTOSC、MF-INTOSC 或 LF-INTOSC）运行

注 1： 复位状态取决于 IESO 配置位 （CONFIG1H<7>）的状态。
2： 如果由内部振荡器提供器件时钟，修改这些位将导致立即进行时钟频率切换。
3： 由 INTSRC 位 （OSCTUNE<7>）选择的时钟源。
4： 修改这些位将导致立即进行时钟源切换。
5： INTSRC = OSCTUNE<7> 且 MFIOSEL = OSCCON2<0>。
6： 内部时钟源的最低功耗选项。

寄存器 3-1： OSCCON：振荡器控制寄存器 （续）

bit 2 **HFIOFS**：HF-INTOSC 频率稳定位
1 = HF-INTOSC 振荡器频率稳定
0 = HF-INTOSC 振荡器频率不稳定

bit 1-0 **SCS<1:0>**：系统时钟选择位 **(4)**
1x = 内部振荡器模块（LF-INTOSC、 MF-INTOSC 或 HF-INTOSC）
01 = SOSC 振荡器
00 = 默认主振荡器 （OSC1/OSC2 或 HF-INTOSC，带或不带 PLL。由 FOSC<3:0> 配置位 （CONFIG1H<3:0>）定义。）

注 1： 复位状态取决于 IESO 配置位 （CONFIG1H<7>）的状态。
2： 如果由内部振荡器提供器件时钟，修改这些位将导致立即进行时钟频率切换。
3： 由 INTSRC 位 （OSCTUNE<7>）选择的时钟源。
4： 修改这些位将导致立即进行时钟源切换。
5： INTSRC = OSCTUNE<7> 且 MFIOSEL = OSCCON2<0>。
6： 内部时钟源的最低功耗选项。

寄存器 3-2： OSCCON2：振荡器控制寄存器 2

U-0	R-0	U-0	R/W-0	R/W-0	U-0	R-x	R/W-0
—	SOSCRUN	—	SOSCDRV(1)	SOSCGO	—	MFIOFS	MFIOSEL
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 **SOSCRUN**：SOSC 运行状态位
1 = 系统时钟来自辅助 SOSC
0 = 系统时钟来自除 SOSC 外的振荡器

bit 5 **未实现**：读为 0

bit 4 **SOSCDRV**：辅助振荡器驱动控制位 **(1)**
1 = 选择高功耗 SOSC 电路
0 = 通过 SOSCSEL<1:0> 配置位选择低 / 高功耗

bit 3 **SOSCGO**：振荡器启动控制位
1 = 即使没有任何其他时钟源请求使用振荡器，振荡器仍保持运行
0 = 如果没有任何其他时钟源请求使用振荡器，则关闭振荡器 （选择通过数字时钟输入而不是外部晶振来运行 SOSC 时，该位不起作用。）

bit 2 **未实现**：读为 0

bit 1 **MFIOFS**：MF-INTOSC 频率稳定位
1 = MF-INTOSC 稳定
0 = MF-INTOSC 不稳定

bit 0 **MFIOSEL**：MF-INTOSC 选择位
1 = 使用 MF-INTOSC 来替代 HF-INTOSC 频率 500 kHz、 250 kHz 和 31.25 kHz
0 = 不使用 MF-INTOSC

注 1： 当选择从数字时钟输入而不是外部晶振来运行 SOSC 时，该位不起作用。

寄存器 3-3： OSCTUNE：振荡器调节寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTSRC	PLLEN	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
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 **INTSRC：**内部振荡器低频时钟源选择位
1 = 来自 16 MHz INTOSC 时钟源的 31.25 kHz 器件时钟 （使能 512 分频， HF-INTOSC）
0 = 来自 INTOSC 31 kHz 振荡器的 31 kHz 器件时钟 （LF-INTOSC）

bit 6 **PLLEN：**倍频器 PLL 使能位
1 = 使能 PLL
0 = 禁止 PLL

bit 5-0 **TUN<5:0>：**快速 RC 振荡器 （INTOSC）频率调节位
011111 = 最高频率
• •
• •
000001
000000 = 中心频率；快速 RC 振荡器运行在经过校准的频率下
111111
• •
• •
100000 = 最低频率

3.3 时钟源与振荡器切换

基本上，PIC18F66K80 系列器件都具有以下独立时钟源：

- 主振荡器
- 辅助振荡器
- 内部振荡器

主振荡器可认为是主要的器件振荡器。这些是指与 OSC1 和 OSC2 引脚连接的任何外部振荡器，包括外部晶振和谐振器模式以及外部时钟模式。通过FOSC<3:0>配置位（CONFIG1H<3:0>）选定时，可以将内部振荡器模块视为主振荡器。内部振荡器模块可以是以下之一：

- 31 kHz LF-INTOSC 时钟源
- 31 kHz 至 500 kHz MF-INTOSC 时钟源
- 31 kHz 至 16 MHz HF-INTOSC 时钟源

特定的模式由 FOSC 配置位定义。这些模式的详细信息将在**第 3.5 节 “ 外部振荡器模式 ”** 中进行介绍。

辅助振荡器是指那些不与 OSC1 或 OSC2 引脚连接的外部时钟源。即使在控制器处于功耗管理模式时，这些时钟源仍可继续工作。 PIC18F66K80 系列器件将 SOSC（Timer1/3/5/7）振荡器作为辅助振荡器源。

SOSC 可以通过请求使用它的任意外设使能。SOSC 可以通过几种方法来使能，即执行以下操作之一：

- 通过任意一个奇编号定时器选择 SOSC 作为时钟源，这需要通过定时器的相应 SOSCEN 位（TxCON<3>）来设置
- 通过 SCS 位 （OSCCON<1:0>）选择 SOSC 作为 CPU 时钟源
- SOSCGO 位 （OSCCON2<3>）置 1

SOSCGO 位用于预热 SOSC，使之在任意外设发出请求之前就绪。

辅助振荡器具有三种运行模式。SOSCSEL<1:0> 位（CONFIG1L<4:3>）决定 SOSC 的工作模式：

- 11 = 高功耗 SOSC 电路
- 10 = 数字 （SCLKI）模式
- 11 = 低功耗 SOSC 电路

如果不需要辅助振荡器，而是需要端口引脚 RC0 和 RC1 上的数字 I/O，则必须将 SOSCSEL 位设置为数字模式。

除了在某些情况下作为主时钟源之外，**内部振荡器**还可以作为功耗管理模式的时钟源。 LF-INTOSC 时钟源也可作为几种特殊功能部件 （如 WDT 和故障保护时钟监视器）的时钟源。在**第 3.6 节 “ 内部振荡器模块 ”** 中对内部振荡器模块进行了更详细的讨论。

PIC18F66K80 系列包含了允许器件时钟源从主振荡器（由器件配置选择）切换到备用时钟源之一的功能。当使能备用时钟源时，可以使用多种功耗管理工作模式。

3.3.1 OSC1/OSC2 振荡器

OSC1/OSC2 振荡器模块用于提供振荡器模式和频率范围：

模式	设计工作频率
LP	31.25-100 kHz
XT	100 kHz 至 4 MHz
HS	4 MHz 至 25 MHz
EC	0 至 64 MHz （外部时钟）
EXTRC	0 至 4 MHz （外部 RC）

基于晶振的振荡器（XT、HS 和 LP）存在一个内置的起振时间。 EC 和 EXTRC 时钟则可以即时开始工作。

3.3.2 时钟源选择

系统时钟选择位 SCS<1:0>（OSCCON<1:0>）用于选择时钟源。可用的时钟源包括主时钟 （由 FOSC<3:0>配置位定义）、辅助时钟 （SOSC 振荡器）和内部振荡器。当写入一个或多个位之后，接着是一段很短的时钟转换间隔，然后时钟源会改变。

OSTS（OSCCON<3>）和SOSCRUN（OSCCON2<6>）位用于指示当前提供器件时钟的是哪个时钟源。OSTS 位用于指示振荡器起振定时器（OST）是否已超时，以及是否由主时钟在主时钟模式下提供器件时钟。SOSCRUN 位用于指示 SOSC 振荡器 （来自 Timer1/3/5/7）何时在辅助时钟模式下提供器件时钟。在功耗管理模式下，任意时刻这些位中只有一位会置 1。如果这些位都没有置 1，则表明当前时钟源是 INTOSC，或内部振荡器刚刚起振且尚未稳定。

IDLEN 位 （OSCCON<7>）决定当执行 SLEEP 指令时器件是进入休眠模式还是某种空闲模式。

第 4.0 节 “ 功耗管理模式 ” 更详细地讨论了 OSCCON 寄存器中标志位和控制位的使用。

注 1： 要选择辅助时钟源，必须使能 Timer1/3/5/7 振荡器。 Timerx 振荡器通过将 Timerx 控制寄存器中的 SOSCEN 位（TxCON<3>）置 1 来使能。如果未使能 Timerx 振荡器，则在执行 SLEEP 指令时选择辅助时钟源的任何尝试都会被忽略。

2： 建议在Timerx振荡器稳定工作之后再执行 SLEEP 指令，否则在 Timerx 振荡器起振时可能会发生很长的延时。

3.3.2.1 系统时钟选择和器件复位

发生所有形式的复位时 SCS 位都会被清零，这意味着 FOSC<3:0> 配置位定义的主振荡器用作器件复位时的主时钟源。它可以是内部振荡器模块自身，也可以是其他主时钟源（HS、EC、XT、LP、外部 RC 和使能 PLL 的模式）之一。

在内部振荡器模块（不带 PLL）作为复位时默认时钟的情况下，快速 RC 振荡器 （INTOSC）将被用作器件时钟源。它起振后的频率为 8 MHz，对应于 IRCF<2:0> 位复位值 （110）的后分频比选择。

不管选择了哪个主振荡器，INTOSC 总是会在器件上电时被使能。它将作为时钟源直到器件已从存储器中装入了其配置值。此时 FOSC 配置位被读取并选择了振荡器的工作模式。

请注意，主时钟源或内部振荡器在任意给定时间都将有两种可能的 SCS<1:0> 位设置选项。

3.3.3 振荡器转换

PIC18F66K80 系列器件包含在时钟源切换时防止时钟产生 “ 毛刺 ” 的电路。在时钟切换时，系统时钟会有短暂的停顿。该停顿的时间长度是旧时钟源的两个周期与新时钟源的三到四个周期的和。此公式假设新时钟源是稳定的。

第 4.1.2 节 “ 进入功耗管理模式 ” 详细讨论了时钟转换。

3.4 RC 振荡器

对于对时序要求不高的应用，选择 RC 和 RCIO 振荡器模式能更好地节约成本。实际的振荡器频率是以下几个因素的函数：

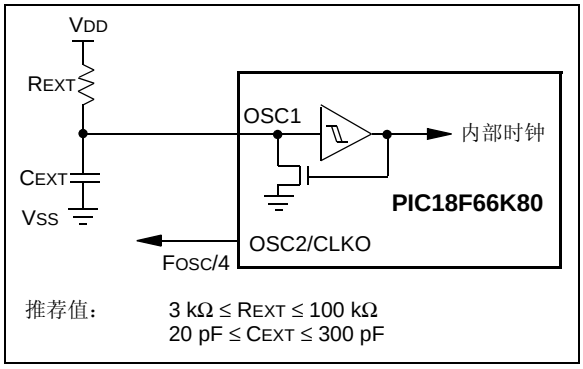
- 供电电压
- 外部电阻 （REXT）和电容 （CEXT）的值
- 工作温度

给定同样的器件、工作电压和温度以及元件值，振荡频率仍然会各不相同。这些频率上的差异是由诸如以下因素引起的：

- 正常制造工艺的差异
- 不同封装类型引线电容的差异 （尤其当 CEXT 值较小时）
- REXT 和 CEXT 在容差范围内的数值波动

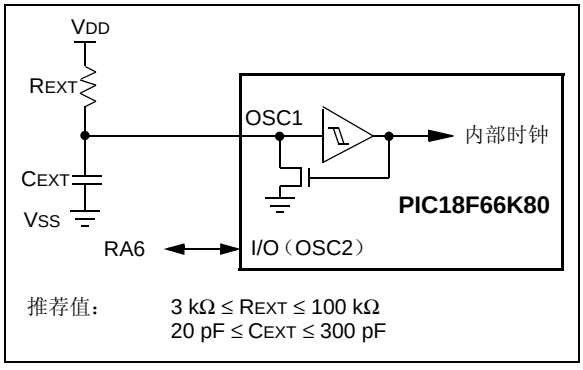
在 RC 振荡器模式下，由 OSC2 引脚输出振荡器频率的 4 分频信号。此信号可用于测试或同步其他逻辑。图 3-2 显示了外接 R/C 组合电路的连接方式。

图 3-2： RC 振荡器模式



RCIO 振荡器模式（图 3-3）的工作方式类似于 RC 模式，不同之处在于 OSC2 引脚变成了一个额外的通用 I/O 引脚。该 I/O 引脚成为 PORTA 的 bit 6 （RA6）。

图 3-3： RCIO 振荡器模式



3.5 外部振荡器模式

3.5.1 晶振 / 陶瓷谐振器 （HS 模式）

在 HS 或 HSPLL 振荡器模式下，将晶振或陶瓷谐振器连接在 OSC1 和 OSC2 引脚之间来产生振荡信号。图 3-4 显示了引脚连接方式。

振荡器的设计要求使用平行切割的晶体。

注：	使用顺序切割的晶体，可能会使振荡器产生的频率超出晶体制造商所给的规范。

表 3-2： 陶瓷谐振器的电容选择

使用的典型电容值：			
模式	**频率**	**OSC1**	**OSC2**
HS	8.0 MHz 16.0 MHz	27 pF 22 pF	27 pF 22 pF

上述电容值仅供设计参考。

要达到理想的振荡器工作状况，可能需要不同的电容值。用户应在应用要求的 VDD 和温度范围内测试振荡器的性能。请参见以下应用笔记以获取振荡器的相关信息：

- *AN588， "PIC® Microcontroller Oscillator Design Guide"*
- *AN826， "Crystal Oscillator Basics and Crystal Selection for rfPIC® and PIC® Devices"*
- *AN849， "Basic PIC® Oscillator Design"*
- *AN943， "Practical PIC® Oscillator Analysis and Design"*
- *AN949， "Making Your Oscillator Work"*

更多信息，请参见表 3-3 下方的 " 注 "。

表 3-3： 晶振的电容选择

振荡器类型	晶振频率	已测试的典型电容值：	
		C1	**C2**
HS	4 MHz	27 pF	27 pF
	8 MHz	22 pF	22 pF
	20 MHz	15 pF	15 pF

上述电容值仅供设计参考。

要达到理想的振荡器工作状况，可能需要不同的电容值。用户应在应用要求的 VDD 和温度范围内测试振荡器的性能。

请参见表 3-2 中列出的 Microchip 应用笔记以获取振荡器的相关信息。更多信息，请参见本表下方的 " 注 "。

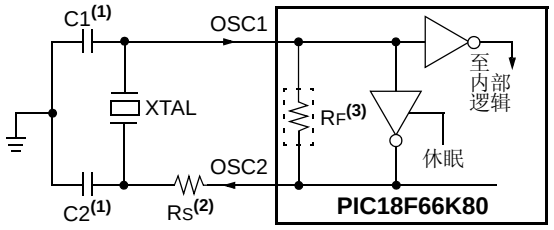
注 1： 电容值越大，振荡器的稳定性越高，但同时起振时间也越长。

2： 因为每种谐振器 / 晶振都有其自身特性，用户应当向谐振器 / 晶振制造商咨询外部元件的适当值。

3： 可能需要使用电阻 RS 以避免对低驱动规格的晶体造成过驱动。

4： 应始终验证振荡器在应用要求的 VDD 和温度范围内的性能。

图 3-4： 晶振 / 陶瓷谐振器工作原理（HS 或 HSPLL 配置）



注 1： 关于C1和C2的初始值，请参见表3-2和表3-3。

2： 对于 AT 条形切割的晶体，可能需要串联一个电阻 （RS）。

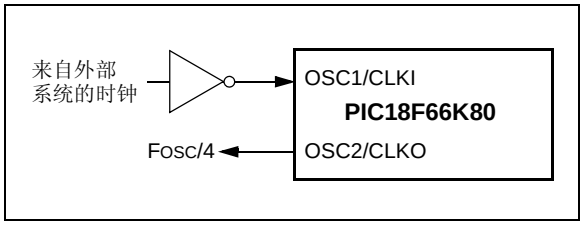
3： RF 的值根据所选的振荡器模式而有所不同。

3.5.2 外部时钟输入 （EC 模式）

EC 和 ECPLL 振荡器模式要求 OSC1 引脚与一个外部时钟源相连。在上电复位后或从休眠模式退出后，不需要振荡器起振时间。

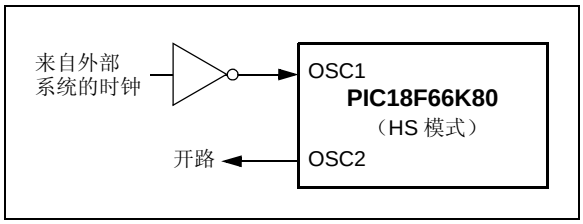
在 EC 振荡器模式下，由 OSC2 引脚输出振荡器频率的 4 分频信号。此信号可用于测试或同步其他逻辑。图 3-5 显示了 EC 振荡器模式的引脚连接方式。

图 3-5： 外部时钟输入工作原理（EC 配置）



如图 3-6 所示，在 HS 模式下， OSC1 引脚也可以连接外部时钟源。在此配置中，OSC2 上无法得到 4 分频输出信号。此配置中的电流消耗稍高于 EC 模式，因为内部振荡器的反馈电路将被使能（在 EC 模式下，反馈电路被禁止）。

图 3-6： 外部时钟输入工作原理（HS 振荡器配置）



3.5.3 PLL 倍频器

如果用户希望使用较低频率振荡器电路或通过晶振将器件时钟频率调节至其最高额定频率，可以选择使用锁相环 （PLL）电路。对于担心高频晶振引起 EMI 或需要内部振荡器提供高速时钟的用户而言，这样做可能有用。

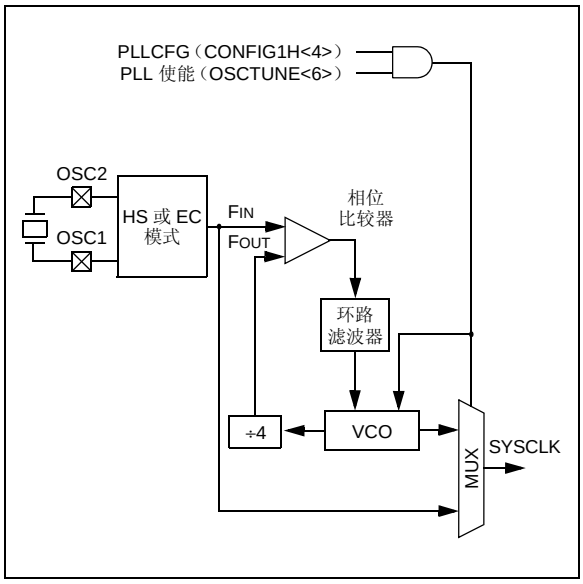
3.5.3.1 HSPLL 和 ECPLL 模式

HSPLL 和 ECPLL 模式可以为器件提供频率为外部振荡源 4 倍的时钟，从而最高频率可达 64 MHz。

可以通过将 PLLEN 位（SCTUNE<6>）或 PLLCFG 位（CONFIG1H<4>）置 1 来使能 PLL。 HF-INTOSC 作为主振荡器时，PLL 必须使用 PLLEN 来使能。这提供了一种通过软件控制 PLL 的方法 （即使 PLLCFG 设置为 1 时也支持），从而只有在 HF-INTOSC 频率介于 4 MHz 至 16 MHz 的输入范围时才使能 PLL。

它还允许用软件更灵活地控制应用的时钟速度。在 HF-INTOSC 模式下，只有在输入频率介于 4 MHz-16 MHz 范围时，才应使能 PLLEN。

图 3-7： PLL 框图



3.5.3.2 PLL 和 HF-INTOSC

当内部振荡器模块配置为主时钟源时，内部振荡器模块也可以使用 PLL。在此配置下，用软件使能 PLL 并产生最高为 64 MHz 的时钟输出。

第 3.6.2 节 “INTPLL 模式 ” 介绍了带 PLL 的 INTOSC 的工作原理。需要注意，只有在 HF-INTOSC 后分频比配置为产生 4 MHz、8 MHz 或 16 MHz 的频率时，才使能 PLL。

3.6 内部振荡器模块

PIC18F66K80 系列器件包含一个内部振荡器模块，它可以产生两种不同的时钟信号。其中任意一个时钟都可以用作单片机的时钟源，这使得可以无需在 OSC1 和 / 或 OSC2 引脚上安装外部振荡器电路。

内部振荡器由 3 个模块组成，具体取决于工作频率。它们是 HF-INTOSC、 MF-INTOSC 和 LF-INTOSC。

在 HF-INTOSC 模式下，内部振荡器可以提供 31 kHz 至 16 MHz 的频率，后分频比将决定所选择的频率（IRCF<2:0>）。

INTSRC 位（OSCTUNE<7>）和 MFIOSEL 位（OSCCON2<0>）还决定由哪一个 INTOSC 提供较低的频率（31 kHz 至 500 kHz）。如果要使用 HF-INTOSC 提供这些频率，则需要 INTSRC = 1 且 MFIOSEL = 0。

在 HF-INTOSC 模式下，后分频比 （IRCF<2:0>）提供 31 kHz 至 16 MHz 的频率范围。如果 HF-INTOSC 与 PLL 配合使用，则送至 PLL 的输入频率应为 4 MHz 至 16 MHz （IRCF<2:0> = 111、 110 或 101）。

要使用 MF-INTOSC 模式来提供 31 kHz 至 500 kHz 的频率范围，则要求 INTSRC = 1 且 MFIOSEL = 1。在该模式下，后分频比（IRCF<2:0>）提供31 kHz至500 kHz 的频率范围。

如果 INTSRC = 0，则 LF-INTOSC 只能提供 31 kHz 的频率。

如果选择 LF-INTOSC 作为器件时钟源，则它会提供 31 kHz 的频率并使能。当使能以下任一功能时，会自动使能该模式：

- 上电延时定时器
- 故障保护时钟监视器
- 看门狗定时器
- 双速启动

第 28.0 节 “CPU 的特殊功能 ” 将详细讨论以上功能。

时钟源频率（HF-INTOSC、MF-INTOSC 或 LF-INTOSC 未经分频的频率）通过配置 OSCCON 寄存器的 IRCF 位以及 INTSRC 和 MFIOSEL 位来进行选择。器件复位时的默认频率为 8 MHz。

3.6.1 INTIO 模式

使用内部振荡器作为时钟源最多可以节约两个外部振荡器引脚，从而将它们用作数字 I/O。有两种不同的振荡器配置 （由 FOSC 配置位决定）可用：

- 在 INTIO1 模式下，OSC2 引脚（RA6）输出 FOSC/4，而 OSC1 用作 RA7 （见图 3-8），用于数字输入和输出。
- 在 INTIO2 模式下， OSC1 用作 RA7， OSC2 用作 RA6 （见图 3-9）。两者都可用作数字输入和输出端口。

图 3-8： INTIO1 振荡器模式

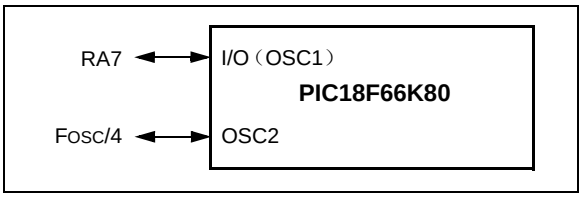
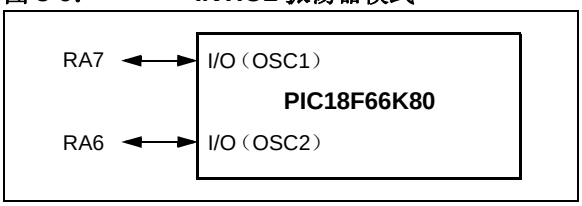


图 3-9： INTIO2 振荡器模式



3.6.2 INTPLL 模式

HF-INTOSC 可以通过使用 4 倍频锁相环（PLL）来产生比内部振荡器源通常所能产生的时钟速度更快的器件时钟速度。当使能时，PLL 可产生 16 MHz 或 64 MHz 的时钟速度。

PLL操作通过软件控制。控制位PLLEN（OSCTUNE<6>）和 PLLCFG （CONFIG1H<4>）用于使能或禁止其操作。 PLL 仅适用于 HF-INTOSC。其他振荡器使用 HS 和EC模式来设置。此外，仅当选定的输出频率为4 MHz 或 16 MHz（OSCCON<6:4> = 111、110 或 101）时，PLL 才会工作。

与 INTIO 模式一样，有两种不同的 INTPLL 模式可用：

- 在 INTPLL1 模式下， OSC2 引脚输出 FOSC/4，而 OSC1 用作 RA7，用于数字输入和输出。从外部看，这与 INTIO1 完全相同 （图 3-8）。
- 在 INTPLL2 模式下， OSC1 用作 RA7， OSC2 用作 RA6，两者都用于数字输入和输出。从外部看，这与 INTIO2 完全相同 （图 3-9）。

3.6.3 内部振荡器输出频率和调节

出厂时已校准了内部振荡器模块使之能够产生 16 MHz 的 INTOSC 输出频率。可以通过写 OSCTUNE 寄存器（寄存器 3-3）中的 TUN<5:0>（OSCTUNE<5:0>），在用户应用程序中进行调整。

在修改了 OSCTUNE 寄存器后，INTOSC（HF-INTOSC 和 MF-INTOSC）的频率将开始改变为新的频率。振荡器需要一定时间才能稳定下来。在此转变期间，代码会继续执行，是否已发生频率转变并无明确的指示。

LF-INTOSC 振荡器独立于 HF-INTOSC 或 MF-INTOSC 时钟源工作。在不同电压和温度下，HF-INTOSC 或 MF-INTOSC 时钟源中的任何变化不一定会反映为 LF-INTOSC 中的变化，反之亦然。LF-INTOSC 的频率不受 OSCTUNE 的影响。

3.6.4 INTOSC 频率漂移

INTOSC 频率可能会随着 VDD 或温度的改变而发生漂移，这一点可能会以各种方式影响控制器的操作。通过修改 OSCTUNE 寄存器中的值可以调节 INTOSC 的频率。根据不同器件，这可能不会对 LF-INTOSC 时钟源的频率造成影响。

调节 INTOSC 需要了解何时调节、调节的方向以及在某些情况下的调整量。这里给出了三种补偿技术。

3.6.4.1 用 EUSART 进行补偿

当 EUSART 开始产生帧错误，或者在异步模式下接收数据有错误时可能需要进行调节。帧错误表示器件时钟的频率太高。要对此进行调整，可以减小 OSCTUNE 中的值来降低时钟频率。另一方面，数据中有错误可能表明时钟速度太低。要对此进行补偿，可以增大 OSCTUNE 中的值来提高时钟频率。

3.6.4.2 用定时器进行补偿

此技术是将器件时钟的速度与某一个参考时钟进行比较。可能要用到两个定时器；一个由外设时钟提供时钟源，而另一个由一个固定的参考源（如 SOSC 振荡器）提供时钟源。

两个定时器都被清零，但由参考源提供时钟信号的定时器产生中断。当中断发生时，使用内部时钟源的定时器值被读取且两个定时器均被清零。如果使用内部时钟源的定时器的值比期望值大很多，则表示内部振荡器模块运行过快。要对此进行调整，需减小 OSCTUNE 寄存器中的值。

3.6.4.3 在捕捉模式下用 CCP 模块进行补偿

CCP 模块可以使用由内部振荡器模块提供时钟信号的自由运行的 Timer1（或 Timer3）和已知周期的外部事件（即交流电源频率）。在 CCPRxH:CCPRxL 寄存器中捕捉并记录第一个事件的时间以备以后使用。当第二个事件导致捕捉时，要用第二个事件的时间减去第一个事件的时间。由于外部事件的周期是已知的，因此可以计算两个事件之间的时间差。

如果测得的时间比计算得到的时间大很多，则表示内部振荡器模块运行过快。要对此进行补偿，需减小 OSCTUNE 寄存器中的值。如果测得的时间比计算得到的时间小很多，则表示内部振荡器模块运行过慢。要对此进行补偿，需增大 OSCTUNE 寄存器中的值。

3.7 参考时钟输出

除了某些振荡器模式中的 FOSC/4 时钟输出外，PIC18F66K80 系列中的器件时钟也可以配置为向端口引脚提供参考时钟输出信号。该功能在所有振荡器配置中都可用，允许用户选择更大范围的时钟分频比来驱动应用中的外部器件。

该参考时钟输出由 REFOCON 寄存器（寄存器 3-4）控制。将 ROON 位（REFOCON<7>）置 1 将使 REFO（RC3）引脚上输出时钟信号。RODIV<3:0> 位允许选择 16 个不同的时钟分频比选项。

ROSSLP 和 ROSEL 位（REFOCON<5:4>）控制休眠模式下参考时钟输出的可用性。ROSEL 位决定用 OSC1 和 OSC2 上的振荡器还是当前系统时钟源作为参考时钟输出。ROSSLP 位决定器件处于休眠模式时 RE3 上的参考时钟源是否可用。

要在休眠模式下使用参考时钟输出，ROSSLP 和 ROSEL 位都必须置 1。器件时钟也必须配置为 EC 或 HS 模式。否则，在器件进入休眠模式时，OSC1 和 OSC2 上的振荡器将会掉电。清零 ROSEL 位允许参考输出频率在任何时钟切换期间随着系统时钟的改变而改变。

寄存器 3-4：　　REFOCON：参考振荡器控制寄存器

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ROON	—	ROSSLP	ROSEL(1)	RODIV3	RODIV2	RODIV1	RODIV0
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　**ROON：**参考振荡器输出使能位
1 = 在 REFO 引脚上使能参考振荡器输出
0 = 禁止参考振荡器输出

bit 6　**未实现：**读为 0

bit 5　**ROSSLP：**参考振荡器输出在休眠模式下停止的位
1 = 参考振荡器在休眠模式下继续运行
0 = 参考振荡器在休眠模式下被禁止

bit 4　**ROSEL：**参考振荡器源选择位 (1)
1 = 主振荡器 （EC 或 HS）用作基本时钟
0 = 系统时钟用作基本时钟；基本时钟反映器件的任何时钟切换

bit 3-0　**RODIV<3:0>：**参考振荡器分频比选择位
1111 = 基本时钟值被 32,768 分频
1110 = 基本时钟值被 16,384 分频
1101 = 基本时钟值被 8,192 分频
1100 = 基本时钟值被 4,096 分频
1011 = 基本时钟值被 2,048 分频
1010 = 基本时钟值被 1,024 分频
1001 = 基本时钟值被 512 分频
1000 = 基本时钟值被 256 分频
0111 = 基本时钟值被 128 分频
0110 = 基本时钟值被 64 分频
0101 = 基本时钟值被 32 分频
0100 = 基本时钟值被 16 分频
0011 = 基本时钟值被 8 分频
0010 = 基本时钟值被 4 分频
0001 = 基本时钟值被 2 分频
0000 = 基本时钟值

注　1：对于 ROSEL（REFOCON<4>），只有在通过 FOSC 设置配置为默认时，主振荡器才可用。这与器件是否处于休眠模式无关。

3.8 功耗管理模式对各种时钟源的影响

当选取 PRI_IDLE 模式时，指定的主振荡器会继续运行而不中断。对于所有其他功耗管理模式，使用 OSC1 引脚的振荡器会被禁止。OSC1引脚（若振荡器使用OSC2引脚，则也包括 OSC2 引脚）将会停止振荡。

在辅助时钟模式（SEC_RUN 和 SEC_IDLE）下，SOSC 振荡器作为器件时钟源工作。如果需要，SOSC 振荡器也可以运行在所有功耗管理模式下为 SOSC 提供时钟。

在 RC_RUN 和 RC_IDLE 模式下，由内部振荡器提供器件时钟源。无论是哪种功耗管理模式，31 kHz 的 LF-INTOSC 输出均可直接用来提供时钟并且可使能来支持多种特殊的功能部件（关于 WDT、故障保护时钟监视器和双速启动的更多信息，请参见**第 28.2 节“ 看门狗定时器（WDT）”**至**第 28.5 节“ 故障保护时钟监视器 ”**）。

如果选择了休眠模式，所有的时钟源都会停止。因为休眠模式切断了所有晶体管的开关电流，休眠模式能实现最低的器件电流消耗（仅泄漏电流）。

在休眠期间使能任何片上功能都将增加休眠时的电流消耗。要支持 WDT 工作，需要使能 INTOSC。 SOSC 振荡器可用来为 Timer1 或 Timer3 提供时钟源。其他功能部件无需器件时钟源也可以工作（即， MSSP 从器件、INTx 引脚和其他等）。在**第 31.2 节“ 直流特性：掉电电流和供电电流 PIC18F66K80 系列（工业级/扩展级）”**中列出了可能显著增加电流消耗的外设。

3.9 上电延时

由两个定时器控制上电延时，这样大多数应用都无需外接复位电路。上电延时可以确保在器件电源稳定（常规环境下）和主时钟稳定工作之前器件保持在复位状态。关于上电延时的更多信息，请参见**第 5.6.1 节“ 上电延时定时器 （PWRT） ”**。

第一个定时器是上电延时定时器 （Power-up Timer，PWRT），它在上电时提供固定的延时 （表 31-11 中的参数 33）；它总是使能的。

第二个定时器是振荡器起振定时器（OST），用于在晶振（HS、XT 或 LP 模式）稳定前使芯片保持在复位状态。OST 在计数 1,024 个振荡周期后允许振荡器为器件提供时钟。

POR 之后有一个 TCSD（表 31-11 中的参数 38）时间间隔的延时，在延时期间控制器为执行指令做准备。

表 3-4： 休眠模式下 OSC1 和 OSC2 引脚的状态

振荡器模式	OSC1 引脚	OSC2 引脚
EC 和 ECPLL	悬空，由外部时钟驱动	处于逻辑低电平 （时钟 4 分频输出）
HS 和 HSPLL	反馈反相器被静态休眠电压禁止	反馈反相器被静态休眠电压禁止
INTOSC 和 INTPLL1/2	I/O 引脚 RA6，方向由 TRISA<6> 控制	I/O 引脚 RA6，方向由 TRISA<7> 控制

注： 关于因休眠和 $\overline{\text{MCLR}}$ 复位而引起的延时，请参见**第 5.0 节“ 复位 ”**。

注：

4.0 功耗管理模式

PIC18F66K80 系列器件总共提供 7 种工作模式，可以更有效地进行功耗管理。这些工作模式提供了多种选择，可在资源受限的应用（如电池供电的设备）中节省功耗。

功耗管理模式有三种类别：

- 运行模式
- 空闲模式
- 休眠模式

器件具有超低功耗唤醒（Ultra Low-Power Wake-up，ULPWU）功能，用于从休眠模式中唤醒器件。

这些类别定义了需要为器件的哪些部分提供时钟，有时还需要定义时钟的速度。运行和空闲模式可以使用三种时钟源（主时钟源、辅助时钟源或内部振荡器模块）中的任意一种；休眠模式则不使用时钟源。

RA0 引脚上的 ULPWU 模式允许缓慢下降的电压能够产生唤醒（即使器件处于休眠模式），同时不会消耗很大的电流。（请参见**第 4.7 节 “ 超低功耗唤醒 ”**。）

功耗管理模式包括几个由早期的 PIC® 器件提供的节省功耗的功能。其中之一就是在其他 PIC18 器件中提供的时钟切换功能。该功能允许控制器使用 SOSC 振荡器代替主振荡器。另一种节省功耗的功能是所有 PIC 器件都提供的休眠模式，在该模式下，器件所有的时钟都停止。

4.1 选择功耗管理模式

选择功耗管理模式之前需要先做出两个决定：

- 是否为 CPU 提供时钟源
- 选择何种时钟源

IDLEN 位（OSCCON<7>）控制是否为 CPU 提供时钟源，而 SCS<1:0> 位（OSCCON<1:0>）用于选择时钟源。表 4-1 总结了各个模式下的位设置、时钟源和受影响的模块。

4.1.1 时钟源

SCS<1:0> 位用于为功耗管理模式在三个时钟源中任选其一。这些时钟源是：

- 主时钟，由 FOSC<3:0> 配置位定义
- 辅助时钟（SOSC 振荡器）
- 内部振荡器模块（用于 LF-INTOSC 模式）

4.1.2 进入功耗管理模式

可以通过装载 OSCCON 寄存器从一种功耗管理模式切换到另一种功耗管理模式。SCS<1:0> 位用于选择时钟源并确定使用运行模式还是空闲模式。更改这些位会导致立即切换到一个新的时钟源（假定新时钟源正在运行）。此切换可能会引起时钟转换延时。**第 4.1.3 节 “ 时钟转换和状态指示 ”** 及其后续章节将会讨论这些注意事项。

执行 SLEEP 指令可以触发进入功耗管理空闲模式或休眠模式。最后实际进入哪个模式由 IDLEN 位的状态决定。

更改功耗管理模式并不总是要求设置之前讨论的所有位，而是取决于当前的模式和将要切换到的模式。通过在发出 SLEEP 指令之前更改振荡器选择位或更改 IDLEN 位可完成多种模式转换。如果已经根据需要配置了 IDLEN 位，可能只需执行 SLEEP 指令就可切换到所需的模式。

表 4-1： 功耗管理模式

模式	OSCCON 位		模块时钟		可用时钟和振荡器源
	IDLEN<7>(1)	SCS<1:0>	CPU	外设	
休眠	0	N/A	关闭	关闭	无 —— 所有时钟均被禁止
PRI_RUN	N/A	00	提供时钟	提供时钟	主时钟——XT、LP、HS、EC、RC 和 PLL 模式。这是正常的全功耗执行模式。
SEC_RUN	N/A	01	提供时钟	提供时钟	辅助时钟 ——SOSC 振荡器
RC_RUN	N/A	1x	提供时钟	提供时钟	内部振荡器模块 (2)
PRI_IDLE	1	00	关闭	提供时钟	主时钟 ——LP、 XT、 HS、 RC 和 EC
SEC_IDLE	1	01	关闭	提供时钟	辅助时钟 ——SOSC 振荡器
RC_IDLE	1	1x	关闭	提供时钟	内部振荡器模块 (2)

注 1： 仅在执行 SLEEP 指令时， IDLEN 位的值才有意义。

2： 包括 INTOSC（HF-INTOSC 和 MF-INTOSC）和 INTOSC 后分频器以及 LF-INTOSC 时钟源。

4.1.3 时钟转换和状态指示

在两个时钟源之间进行转换所需的时间长度是旧时钟源的两个周期与新时钟源的三到四个周期的和。此公式假设新时钟源是稳定的。本章将 HF-INTOSC 和 MF-INTOSC 合称为 INTOSC。

以下三位用于指示当前的时钟源及其状态，如表 4-2 中所示。这三位是：

- OSTS （OSCCON<3>）
- HFIOFS （OSCCON<2>）
- SOSCRUN （OSCCON2<6>）

表 4-2： 系统时钟指示

主时钟源	OSTS	HFIOFS 或 MFIOFS	SOSCRUN
主振荡器	1	0	0
INTOSC （HF-INTOSC 或 MF-INTOSC）	0	1	0
辅助振荡器	0	0	1
MF-INTOSC 或 HF-INTOSC 作为主时钟源	1	1	0
LF-INTOSC 正在运行或 INTOSC 尚未稳定	0	0	0

当 OSTS 位置 1 时，表明由主时钟提供器件时钟。当 HFIOFS 或 MFIOFS 位置 1 时，表明由 INTOSC 输出提供稳定的时钟源到分频器，实际上由分频器驱动器件时钟。当 SOSCRUN 位置 1 时，表明由 SOSC 振荡器提供时钟。如果这些位均不置 1，则表明要么由 LF-INTOSC 时钟源提供器件时钟，要么 INTOSC 时钟源尚未稳定。

如果用 FOSC<3:0> 配置位（CONFIG1H<3:0>）将内部振荡器模块配置为主时钟源，则在处于 PRI_RUN 或 PRI_IDLE 模式时，OSTS 和 HFIOFS 或 MFIOFS 位可能同时置 1。这指示主时钟 （INTOSC 输出）正在产生稳定的输出。进入频率相同的另一种 INTOSC 功耗管理模式将清零 OSTS 位。

注 1： 在仅修改 IRCF 位时应该特别小心。在 VDD 较低时，可以选择一个比该 VDD 所支持时钟速度更高的时钟速度。违反 VDD/FOSC 规范可能导致器件不能正常工作。

2： 执行 SLEEP 指令并不一定会将器件置于休眠模式。它只是作为触发条件，让器件进入休眠模式或一种空闲模式，具体何种模式由 IDLEN 位的设置决定。

4.1.4 多条 SLEEP 命令

使用 SLEEP 指令调用功耗管理模式时，具体进入何种模式在该指令执行那一刻由IDLEN位的设置决定。如果执行了另一条 SLEEP 指令，器件将进入由此时 IDLEN 位指定的功耗管理模式。如果 IDLEN 位已更改，器件将进入由新的设置指定的新的功耗管理模式。

4.2 运行模式

在运行模式下，内核和外设的时钟均有效。这些运行模式之间的区别就在于时钟源不同。

4.2.1 PRI_RUN 模式

PRI_RUN 模式是单片机的正常全功耗执行模式。除非使能了双速启动，否则该模式也是器件复位后的默认模式。（详情请参见**第 28.4 节 “ 双速启动 ”**。）在该模式下， OSTS 位置 1。如果内部振荡器模块为主时钟源，则 HFIOFS 或 MFIOFS 位也可能置 1。（请参见**第 3.2 节 “ 控制寄存器 ”**。）

4.2.2 SEC_RUN 模式

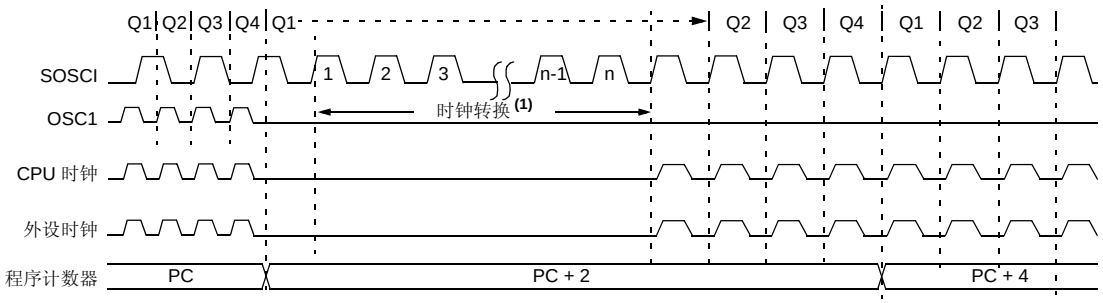
SEC_RUN 模式与其他 PIC18 器件提供的 “ 时钟切换 ” 功能兼容。在该模式下，CPU 和外设将 SOSC 振荡器作为时钟源。这使得在保留高精度时钟源的情况下仍可获得较低的功耗。

通过将 SCS<1:0> 位设置为 01 可以进入 SEC_RUN 模式。器件时钟源被切换到 SOSC 振荡器 （见图 4-1），主振荡器被关闭， SOSCRUN 位（OSCCON2<6>）被置 1 并且 OSTS 位被清零。

注： 可以通过将SOSCGO位（OSCCON2<3>）置 1 使能 SOSC 振荡器。如果该位置 1，那么在 SCS<1:0> 设置为 01 之后时钟会立即切换到SEC_RUN模式下的时钟源。

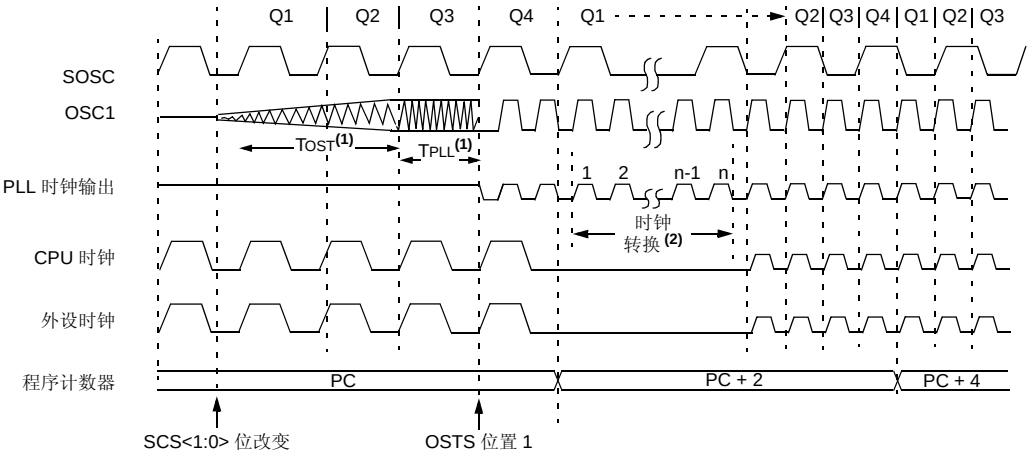
在从 SEC_RUN 模式转换到 PRI_RUN 模式时，外设和 CPU 继续使用 SOSC 振荡器作为时钟源，直到主时钟启动。当主时钟就绪时，时钟切换回主时钟 （见图 4-2）。当时钟切换完成时，SOSCRUN 位被清零，OSTS 位被置 1 并且由主时钟提供器件时钟。唤醒不会影响 IDLEN 和 SCS 位， SOSC 振荡器继续运行。

图 4-1：　　进入 SEC_RUN 模式的转换时序



注 1：时钟转换通常发生在 2-4 个 TOSC 内。

图 4-2：　　从 SEC_RUN 模式切换到 PRI_RUN 模式的转换时序 （HSPLL）



注　1：TOST = 1024 TOSC；TPLL = 2 ms（近似值）。这些时间间隔未按比例绘制。
2：时钟转换通常发生在 2-4 个 TOSC 内。

4.2.3　RC_RUN 模式

在 RC_RUN 模式下，使用 INTOSC 多路开关将内部振荡器模块作为 CPU 和外设的时钟源。在该模式下，主时钟关闭。当使用 LF-INTOSC 时钟源时，该模式是在代码执行期间所有运行模式中最节省功耗的模式。它非常适用于对时序不是很敏感或者不是一直需要高速时钟的应用。

如果主时钟源为内部振荡器模块 LF-INTOSC 或 INTOSC（MF-INTOSC 或 HF-INTOSC），在代码执行期间，PRI_RUN 和 RC_RUN 这两种模式区别不大。但是在进入或退出 RC_RUN 模式时会发生时钟切换延时。因此，如果主时钟源为内部振荡器模块，建议不要使用 RC_RUN 模式。

通过将 SCS1 位设置为 1 可以进入该模式。为保持与未来器件的软件兼容性，即使 SCS0 位被忽略，还是建议将该位清零。当时钟源切换到 INTOSC 多路开关时（见图 4-3），主振荡器被关闭，OSTS 位被清零。在任意时刻修改 IRCF 位可以立即改变时钟速度。

注：	在只修改一个 IRCF 位时应该特别小心。在 VDD 较低时，您很有可能会选择一个比该 VDD 所支持的时钟速度更高的时钟速度。违反 VDD/FOSC 规范可能导致器件不能正常工作。

如果 IRCF 位和 INTSRC 位均清零，则 INTOSC 输出（HF-INTOSC/MF-INTOSC）不会使能，HFIOFS 和 MFIOFS 位将保持清零状态。此时将不会有任何关于当前时钟源的指示。由 LF-INTOSC 时钟源提供器件时钟。

如果 IRCF 位从全清零状态改变（因而使能 INTOSC 输出），或者如果 INTSRC 或 MFIOSEL 被置 1，则在 INTOSC 输出稳定后 HFIOFS 或 MFIOFS 位将被置 1。详情请参见表 4-3。

表 4-3：　　内部振荡器频率稳定位

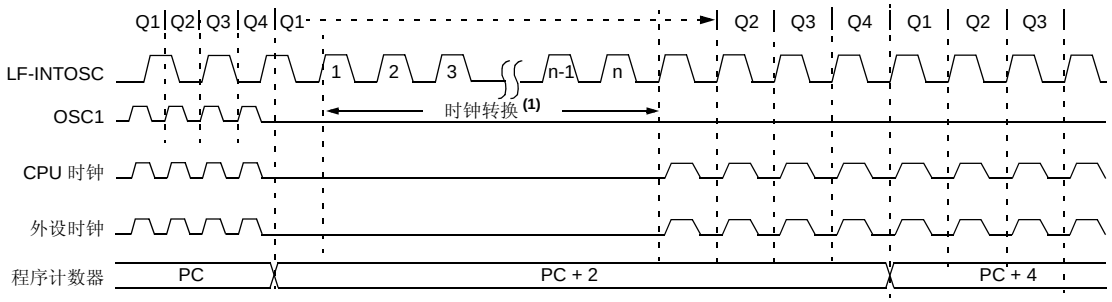
IRCF<2:0>	INTSRC	MFIOSEL	INTOSC 稳定时 MFIOFS 或 HFIOFS 的状态
000	0	x	MFIOFS = 0，HFIOFS = 0 且时钟源为 LF-INTOSC
000	1	0	MFIOFS = 0，HFIOFS = 1 且时钟源为 HF-INTOSC
000	1	1	MFIOFS = 1，HFIOFS = 0 且时钟源为 MF-INTOSC
非零	x	0	MFIOFS = 0，HFIOFS = 1 且时钟源为 HF-INTOSC
非零	x	1	MFIOFS = 1，HFIOFS = 0 且时钟源为 MF-INTOSC

在一个 TIOBST（表 31-11 中的参数 39）时间间隔之后，INTOSC 时钟源趋于稳定，此时器件时钟继续运行。

如果之前 IRCF 位为一个非零值，或者如果在将 SCS1 置 1 之前 INTSRC 已经置 1 并且 INTOSC 时钟源已达到稳定，那么 HFIOFS 或 MFIOFS 位将保持置 1 状态。

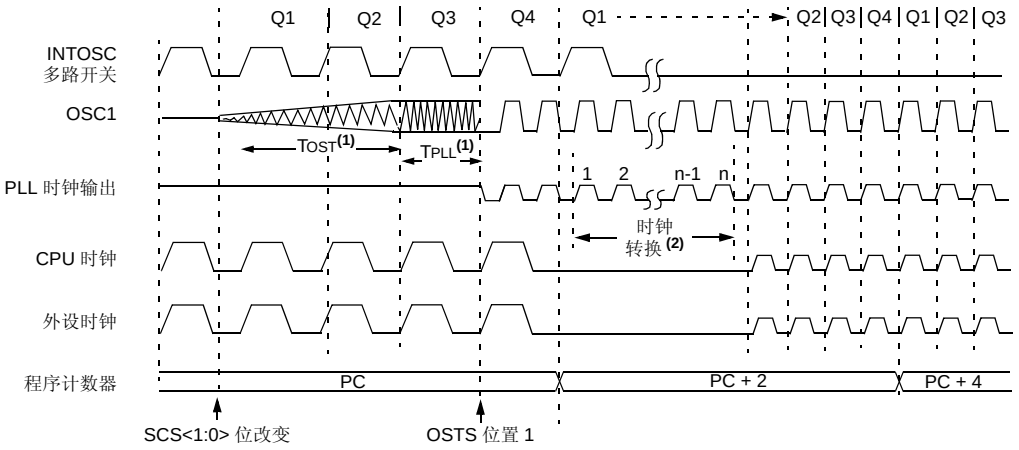
在从 RC_RUN 模式转换到 PRI_RUN 模式时，在主时钟处于启动状态时，器件将继续使用 INTOSC 多路开关作为时钟源。当主时钟就绪时，时钟切换到主时钟（见图 4-4）。当时钟切换完成时，HFIOFS 或 MFIOFS 位被清零，OSTS 位被置 1 并且由主时钟提供器件时钟。切换不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器（FSCM），则 LF-INTOSC 时钟源将继续运行。

图 4-3：　　到 RC_RUN 模式的转换时序



注 1：时钟转换通常发生在 2-4 个 TOSC 内。

图 4-4：　　从 RC_RUN 模式切换到 PRI_RUN 模式的转换时序



注　1：TOST = 1024 TOSC； TPLL = 2 ms （近似值）。这些时间间隔未按比例绘制。

2：时钟转换通常发生在 2-4 个 TOSC 内。

4.3 休眠模式

PIC18F66K80 系列器件的功耗管理休眠模式和所有其他 PIC 器件提供的传统休眠模式相同。通过清零 IDLEN 位（器件复位时的默认状态）并执行 SLEEP 指令即可进入该模式。这将关闭选定的振荡器（图 4-5），并将所有的时钟源状态位清零。

从任何其他模式进入休眠模式不需要切换时钟。这是因为控制器一旦进入休眠模式就不需要时钟了。如果选择了 WDT，LF-INTOSC 时钟源将继续工作。如果使能了 SOSC 振荡器，它也将继续运行。

当在休眠模式下发生唤醒事件（通过中断、复位或 WDT 超时）时，在时钟源（通过 SCS<1:0> 位选择）就绪之前器件将没有时钟源（见图 4-6）。或者，如果使能了双速启动或故障保护时钟监视器，器件将使用内部振荡器模块作为时钟源（见**第 28.0 节 “CPU 的特殊功能 ”**）。在这两种情况下，当由主时钟提供器件时钟时，OSTS 位将置 1。唤醒不会影响 IDLEN 和 SCS 位。

4.4 空闲模式

空闲模式允许在外设继续工作的同时有选择地关闭控制器的 CPU。选择特定的空闲模式允许用户进一步管理功耗。

如果在执行 SLEEP 指令时， IDLEN 位被设置为 1，外设将使用由 SCS<1:0> 位选择的时钟源，而 CPU 没有时钟源。时钟源状态位不受影响。这种方法可以用于从给定运行模式快速地切换到其相应的空闲模式。

如果选择了 WDT，LF-INTOSC 时钟源将继续工作。如果使能了 SOSC 振荡器，它也将继续运行。

由于 CPU 没有执行指令，器件只能通过中断、WDT 超时或复位从空闲模式退出。当发生唤醒事件时，会有一段长度为 TCSD（表 31-11 中的参数 38）的延时，CPU 开始执行代码。当 CPU 开始执行代码时，它将沿用当前空闲模式所使用的时钟源。例如，当从 RC_IDLE 模式唤醒时，将使用内部振荡器模块为 CPU 和外设提供时钟（即 RC_RUN 模式）。唤醒不会影响 IDLEN 和 SCS 位。

当处于任何空闲模式或休眠模式下时，WDT 超时会导致 WDT 唤醒并进入当前由 SCS<1:0> 位指定的运行模式。

图 4-5： 进入休眠模式的转换时序

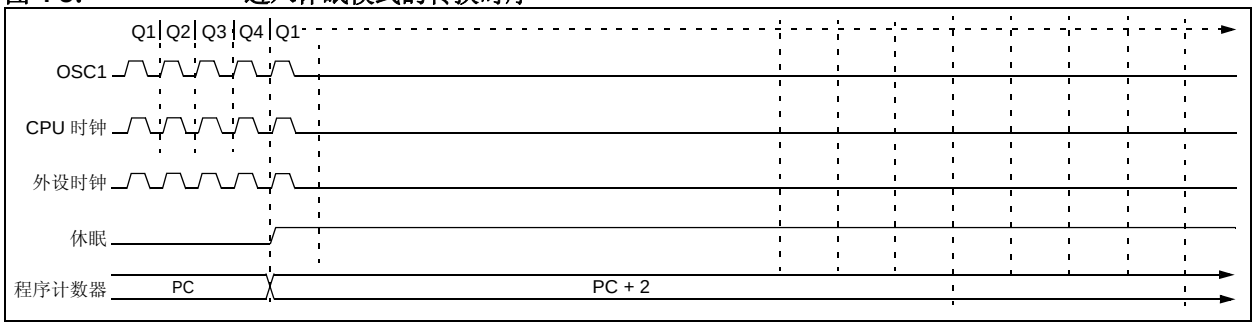
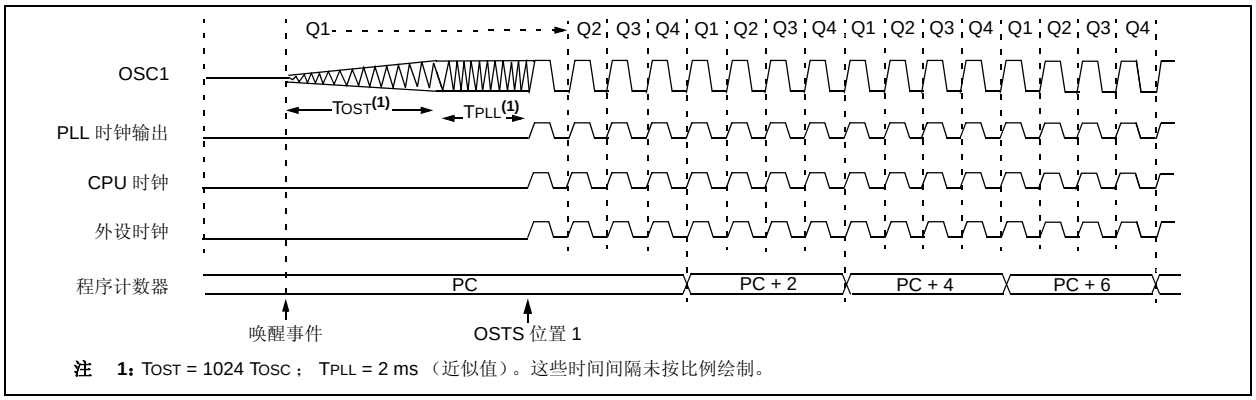


图 4-6： 从休眠模式唤醒的转换时序（HSPLL）



注 1： TOST = 1024 TOSC； TPLL = 2 ms（近似值）。这些时间间隔未按比例绘制。

4.4.1 PRI_IDLE 模式

在三种低功耗空闲模式中，只有该模式不会禁止主器件时钟。由于时钟源不需要“ 预热 ”或是从其他振荡器转换过来，对于时序敏感的应用，选用该模式可以使用较精确的主时钟源以最快的速度恢复器件运行。

可以通过将 IDLEN 位置 1 并执行 SLEEP 指令以实现从 PRI_RUN 模式进入 PRI_IDLE 模式。如果器件处于另一种运行模式，则应首先将 IDLEN 位置 1，然后将 SCS 位清零并执行 SLEEP 指令。虽然 CPU 已被禁止，但外设仍可继续使用由 FOSC<3:0> 配置位指定的主时钟源为其提供时钟信号。 OSTS 位保持置 1 （见图 4-7）。

当发生唤醒事件时，由主时钟源为 CPU 提供时钟。在唤醒事件和代码执行开始之间需要一个 TCSD（表 31-11 中的参数 38）时间间隔的延时。该延时用来让 CPU 做好执行指令的准备。在唤醒之后，OSTS 位保持置 1 状态。唤醒不会影响 IDLEN 和 SCS 位 （见图 4-8）。

4.4.2 SEC_IDLE 模式

在 SEC_IDLE 模式下，CPU 被禁止，但外设继续将 SOSC 振荡器作为时钟源。可以通过将 IDLEN 位置 1 并执行 SLEEP 指令从 SEC_RUN 模式进入该模式。如果器件处于另一种运行模式，则应首先将 IDLEN 位置 1，然后将 SCS<1:0> 位设置为 01 并执行 SLEEP 指令。当时钟源切换到 SOSC 振荡器时，主振荡器被关闭，OSTS 位被清零并且 SOSCRUN 位被置 1。

当唤醒事件发生时，外设继续将 SOSC 振荡器作为时钟源。唤醒事件发生后经过一个 TCSD 时间间隔，CPU 开始执行代码并使用 SOSC 振荡器作为其时钟源。唤醒不会影响 IDLEN 和 SCS 位，SOSC 振荡器继续运行（见图 4-8）。

图 4-7： 进入空闲模式的转换时序

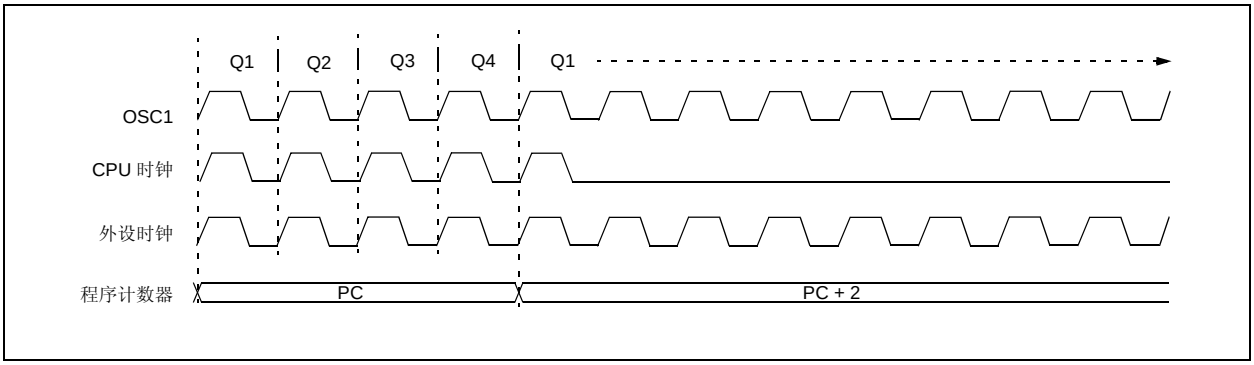
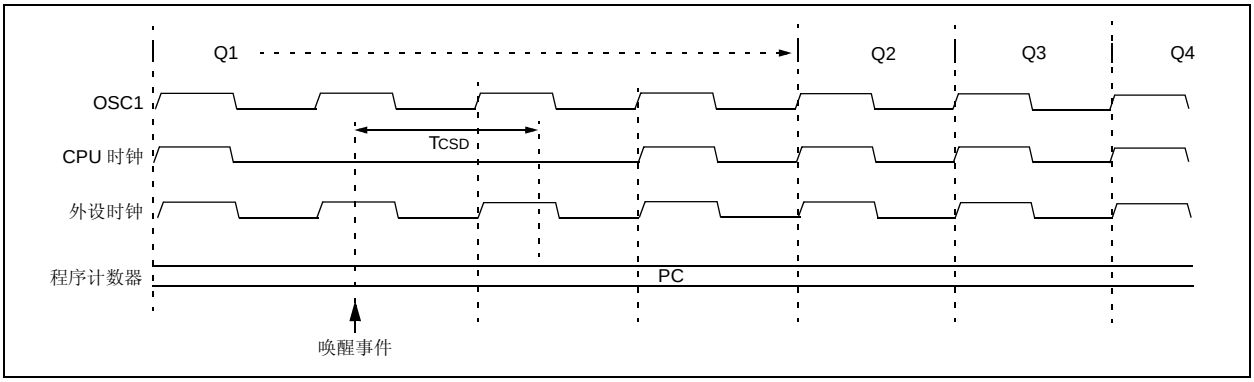


图 4-8： 从空闲模式唤醒进入运行模式的转换时序



4.4.3 RC_IDLE 模式

在 RC_IDLE 模式下，CPU 被禁止，但仍继续由使用 INTOSC 多路开关的内部振荡器模块为外设提供时钟。该模式允许在空闲期间对功耗管理进行控制。

可以通过将IDLEN位置1并执行SLEEP指令从RC_RUN模式进入该模式。如果器件处于另一种运行模式，可以先将IDLEN位置1，然后再将SCS1位置1并执行SLEEP指令。为保持与未来器件的软件兼容性，尽管 SCS0 的值被忽略，还是建议将其清零。通过在执行 SLEEP 指令之前修改 IRCF 位，可以使用 INTOSC 多路开关来选择更高的时钟频率。当时钟源切换到 INTOSC 多路开关时，主振荡器被关闭并且 OSTS 位被清零。

如果 IRCF 位被设置为任何非零值，或者 INTSRC/MFIOSEL 位被置 1，就会使能 INTOSC 输出。在一个 TIOBST（表 31-11 中的参数 39）时间间隔之后 INTOSC 输出趋于稳定，随后 HFIOFS/MFIOFS 位被置 1。关于 HFIOFS/MFIOFS 位的信息，请参见表 4-3。

在 INTOSC 时钟源稳定期间继续为外设提供时钟。如果之前 IRCF 位为一个非零值，或者如果在执行 SLEEP 指令之前 INTSRC 已经置 1 并且 INTOSC 时钟源已达到稳定，那么 HFIOFS/MFIOFS 位将保持置 1 状态。如果 IRCF 位和 INTSRC 位均清零，则 INTOSC 输出不会使能，HFIOFS/MFIOFS 位将保持清零状态，此时将不会有任何关于当前时钟源的指示。

当唤醒事件发生时，外设继续将 INTOSC 多路开关作为时钟源。在唤醒事件后的一个 TCSD （表 31-11 中的参数 38）延时之后， CPU 开始执行代码并使用 INTOSC 多路开关作为时钟源。唤醒不会影响IDLEN和SCS位。如果使能了 WDT 或故障保护时钟监视器，则 INTOSC 时钟源将继续运行。

4.5 选择性外设模块控制

利用空闲模式，用户可以通过停止 CPU 时钟来充分地降低功耗。然而，外设模块的时钟仍然保持运行，因此会有功耗产生。有些应用可能存在此模式无法提供的需求：为 CPU 处理供电，而外设的功耗达到最低。

PIC18F66K80 系列器件支持选择性地禁止外设模块，从而降低或消除外设功耗，以此来满足这一需求。这可以通过两个控制位来实现：

- 外设使能位（通常命名为 XXXEN）—— 位于相应模块的主控制寄存器中
- 外设模块禁止（Peripheral Module Disable，PMD）位（ 通常命名为 XXXMD）—— 位于 PMDx 控制寄存器 （PMD0、 PMD1 或 PMD2）之一中

通过清零模块的 XXXEN 位以禁止其功能来禁止该模块，但会保留其寄存器的读写功能。这也会降低功耗，但不如第二种方法降低得多。

大多数外设模块都有一个使能位。

相反，将模块的 PMD 位置 1 会禁止该模块的所有时钟源，从而将其功耗降至绝对最低。在该状态下，与外设相关的控制和状态寄存器也会被禁止，因此写入这些寄存器不起作用，且读取值无效。许多外设模块都有一个对应的 PMD 位。

在 PIC18F66K80 系列器件中有 3 个 PMD 寄存器：PMD0、 PMD1 和 PMD2。这些寄存器具有一些与每个模块关联的位，用于禁止或使能特定的外设。

寄存器 4-1：　　PMD2：外设模块禁止寄存器 2

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	MODMD	ECANMD	CMP2MD	CMP1MD
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 7-4　**未实现：**读为 0

bit 3　**MODMD：**调制器输出模块禁止位 **(1)**
1 = 禁止调制器输出模块。所有调制器输出寄存器都保持在复位状态且不可写。
0 = 使能调制器输出模块

bit 2　**ECANMD：**增强型 CAN 模块禁止位
1 = 禁止增强型 CAN 模块。所有增强型 CAN 寄存器都保持在复位状态且不可写。
0 = 使能增强型 CAN 模块

bit 1　**CMP2MD：**比较器 2 模块禁止位
1 = 禁止比较器 2 模块。所有比较器 2 寄存器都保持在复位状态且不可写。
0 = 使能比较器 2 模块

bit 0　**CMP1MD：**比较器 1 模块禁止位
1 = 禁止比较器 1 模块。所有比较器 1 寄存器都保持在复位状态且不可写。
0 = 使能比较器 1 模块

注　1： 仅在 64 引脚器件 （PIC18F6XK80 和 PIC18LF6XK80）上实现。

寄存器 4-2：　　PMD1：外设模块禁止寄存器 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
PSPMD(1)	CTMUMD	ADCMD	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	
-n = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 7　**PSPMD**：外设模块禁止位 (1)
1 = 禁止 PSP 模块。所有 PSP 寄存器都保持在复位状态且不可写。
0 = 使能 PSP 模块

bit 6　**CTMUMD**：PMD CTMU 禁止位
1 = 禁止 CTMU 模块。所有 CTMU 寄存器都保持在复位状态且不可写。
0 = 使能 CTMU 模块

bit 5　**ADCMD**：ADC 模块禁止位
1 = 禁止 ADC 模块。所有 ADC 寄存器都保持在复位状态且不可写。
0 = 使能 ADC 模块

bit 4　**TMR4MD**：TMR4MD 禁止位
1 = 禁止 Timer4 模块。所有 Timer4 寄存器都保持在复位状态且不可写。
0 = 使能 Timer4 模块

bit 3　**TMR3MD**：TMR3MD 禁止位
1 = 禁止 Timer3 模块。所有 Timer3 寄存器都保持在复位状态且不可写。
0 = 使能 Timer3 模块

bit 2　**TMR2MD**：TMR2MD 禁止位
1 = 禁止 Timer2 模块。所有 Timer2 寄存器都保持在复位状态且不可写。
0 = 使能 Timer2 模块

bit 1　**TMR1MD**：TMR1MD 禁止位
1 = 禁止 Timer1 模块。所有 Timer1 寄存器都保持在复位状态且不可写。
0 = 使能 Timer1 模块

bit 0　**TMR0MD**：Timer0 模块禁止位
1 = 禁止 Timer0 模块。所有 Timer0 寄存器都保持在复位状态且不可写。
0 = 使能 Timer0 模块

注　1： 在 28 引脚器件 （PIC18F2XK80 和 PIC18LF2XK80）上未实现。

寄存器 4-3：　　PMD0：外设模块禁止寄存器 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
CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
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　**CCP5MD**：CCP5 模块禁止位
1 = 禁止 CCP5 模块。所有 CCP5 寄存器都保持在复位状态且不可写。
0 = 使能 CCP5 模块

bit 6　**CCP4MD**：CCP4 模块禁止位
1 = 禁止 CCP4 模块。所有 CCP4 寄存器都保持在复位状态且不可写。
0 = 使能 CCP4 模块

bit 5　**CCP3MD**：CCP3 模块禁止位
1 = 禁止 CCP3 模块。所有 CCP3 寄存器都保持在复位状态且不可写。
0 = 使能 CCP3 模块

bit 4　**CCP2MD**：CCP2 模块禁止位
1 = 禁止 CCP2 模块。所有 CCP2 寄存器都保持在复位状态且不可写。
0 = 使能 CCP2 模块

bit 3　**CCP1MD**：ECCP1 模块禁止位
1 = 禁止 ECCP1 模块。所有 ECCP1 寄存器都保持在复位状态且不可写。
0 = 使能 ECCP1 模块

bit 2　**UART2MD**：EUSART2 模块禁止位
1 = 禁止 USART2 模块。所有 USART2 寄存器都保持在复位状态且不可写。
0 = 使能 USART2 模块

bit 1　**UART1MD**：EUSART1 模块禁止位
1 = 禁止 USART1 模块。所有 USART1 寄存器都保持在复位状态且不可写。
0 = 使能 USART1 模块

bit 0　**SSPMD**：MSSP 模块禁止位
1 = 禁止 MSSP 模块。所有 SSP 寄存器都保持在复位状态且不可写。
0 = 使能 MSSP 模块

4.6 退出空闲和休眠模式

由中断、复位或 WDT 超时触发从休眠模式或任何空闲模式的退出。本节将讨论从功耗管理模式退出的触发方式。在每种功耗管理模式章节中我们已经讨论过其时钟源子系统的操作（见**第 4.2 节 “ 运行模式 ”**、**第 4.3 节 “ 休眠模式 ”** 和**第 4.4 节 “ 空闲模式 ”**）。

4.6.1 通过中断退出

任何可用的中断源都可导致器件从空闲模式或休眠模式退出回到运行模式。要使能此功能，必须通过将对应 INTCONx 或 PIEx 寄存器中的中断允许位置 1 来允许中断源。当相应的中断标志位置 1 时，触发退出操作。

在通过中断从空闲或休眠模式退出时，如果 GIE/GIEH 位（INTCON<7>）置 1，程序将跳转到中断向量处执行代码。否则，代码将继续执行，不进行跳转（见**第 10.0 节 “ 中断 ”**）。

4.6.2 通过 WDT 超时退出

根据 WDT 超时发生时器件所处的不同功耗管理模式会引发不同的操作。

如果器件不在执行代码（所有空闲模式和休眠模式），超时将导致从功耗管理模式退出（见**第 4.2 节 “ 运行模式 ”** 和**第 4.3 节 “ 休眠模式 ”**）。如果器件正在执行代码（所有运行模式），超时将导致 WDT 复位（见**第 28.2 节 “ 看门狗定时器（WDT） ”**）。

执行 SLEEP 或 CLRWDT 指令、当前选定的时钟源失效（如果使能了故障保护监视器）以及修改 OSCCON 寄存器中的 IRCF 位（如果内部振荡器模块为器件时钟源），均将清零 WDT 定时器和后分频器。

4.6.3 通过复位退出

通常，器件通过振荡器起振定时器（OST）保持在复位状态，直到主时钟就绪。主时钟就绪后，OSTS 位置 1，器件开始执行代码。如果内部振荡器模块是新时钟源，则 HFIOFS/MFIOFS 位将置 1。

从复位状态退出到开始执行代码期间的延迟时间由唤醒前后的时钟源以及振荡器的类型（如果新时钟源为主时钟）决定。表 4-4 中总结了退出延时。

可以在主时钟就绪之前开始执行代码。如果使能了双速启动（见**第 28.4 节 “ 双速启动 ”**）或故障保护时钟监视器（见**第 28.5 节 “ 故障保护时钟监视器 ”**），器件可以在复位源被清除之后立即开始执行代码。由内部振荡器模块驱动的 INTOSC 多路开关作为代码执行的时钟源。执行代码时，由内部振荡器模块提供时钟源直到主时钟就绪，或者在主时钟就绪前进入功耗管理模式，随后将关闭主时钟。

4.6.4 在没有振荡器起振延时的情况下退出

从某些功耗管理模式退出完全不需要 OST 延时。有以下两种情形：

- 处于主时钟源不停止的 PRI_IDLE 模式时
- 主时钟源不是 LP、XT、HS 或 HSPLL 中的任意一种模式时

在这些情况下，主时钟源不需要振荡器起振延时，因为它已经在运行（PRI_IDLE），或者它本来就不需要振荡器起振延时（RC、EC 和 INTIO 振荡器模式）。但是，当器件退出休眠和空闲模式时，在唤醒事件之后仍然需要一个固定的 T_{CSD} 时间间隔的延时，以便让 CPU 准备好执行代码。在延时后的第一个时钟周期重新开始执行指令。

4.7 超低功耗唤醒

RA0 引脚上的超低功耗唤醒（ULPWU）功能允许缓慢下降的电压能够产生中断，同时不消耗很大的电流。

要使用该功能：

1. 通过将 RA0 引脚配置为输出并将其设置为 1，对 RA0 上的电容充电。
2. 通过将 RA0 配置为输入来停止对电容充电。
3. 将 WDTCON 寄存器中的 ULPEN 和 ULPSINK 位置 1，使电容放电。
4. 配置休眠模式。
5. 进入休眠模式。

当 RA0 上的电压降到低于 VIL 时，将唤醒器件并执行下一条指令。

该功能提供了一种定期将器件从休眠模式唤醒的低功耗技术。

超时时间取决于 RA0 上 RC 电路的放电时间。

当 ULPWU 模块将器件从休眠模式唤醒时，ULPLVL 位（WDTCON<5>）会置 1。软件可以在唤醒时检查该位，以确定唤醒源。

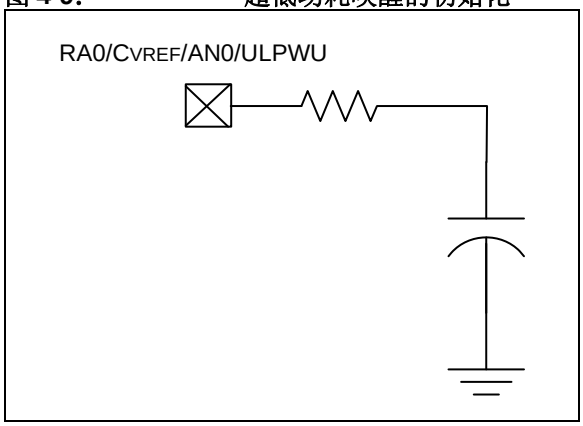
关于初始化 ULPWU 模块的信息，请参见例 4-1。

例 4-1：　　超低功耗唤醒的初始化

```
            //***************************
            //Charge the capacitor on RA0
            //***************************
TRISAbits.TRISA0 = 0;
PORTAbits.RA0 = 1;
for(i = 0; i < 10000; i++) Nop();
            //****************************
            //Stop Charging the capacitor
            //on RA0
            //****************************
TRISAbits.TRISA0 = 1;
            //****************************
            //Enable the Ultra Low Power
            //Wakeup module and allow
            //capacitor discharge
            //****************************
WDTCONbits.ULPEN = 1;
WDTCONbits.ULPSINK = 1;
            //For Sleep
OSCCONbits.IDLEN = 0;
            //Enter Sleep Mode
            //
Sleep();
            //for sleep, execution will
            //resume here
```

RA0 和外部电容之间的串联电阻为 RA0/CVREF/AN0/ULPWU 引脚提供过流保护，同时允许使用软件校准超时时间（见图 4-9）。

图 4-9：　　超低功耗唤醒的初始化



可以使用一个定时器来测量电容的充放电时间。然后调节充电时间，以提供休眠模式下所需的延时。该技术将对温度、电压和元件精度的影响进行补偿。外设还可以配置为简单的可编程低压检测（Low-Voltage Detect，LVD）或温度传感器。

注： 更多信息，请参见 *AN879，“Using the Microchip Ultra Low-Power Wake-up Module”*（DS00879）。

表 4-4： 通过复位从休眠模式或任何空闲模式唤醒的退出延时 （按时钟源分类）

功耗管理模式	时钟源 (5)	退出延时	时钟就绪状态位
PRI_IDLE 模式	LP、 XT 或 HS	T_{CSD}(1)	OSTS
	HSPLL		
	EC 或 RC		
	HF-INTOSC(2)		HFIOFS
	MF-INTOSC(2)		MFIOFS
	LF-INTOSC		无
SEC_IDLE 模式	SOSC	T_{CSD}(1)	SOSCRUN
RC_IDLE 模式	HF-INTOSC(2)	T_{CSD}(1)	HFIOFS
	MF-INTOSC(2)		MFIOFS
	LF-INTOSC		无
休眠模式	LP、 XT 或 HS	T_{OST}(3)	OSTS
	HSPLL	$T_{OST} + t_{rc}$(3)	
	EC 或 RC	T_{CSD}(1)	
	HF-INTOSC(2)	T_{IOBST}(4)	HFIOFS
	MF-INTOSC(2)		MFIOFS
	LF-INTOSC		无

注 1： 当从休眠模式和所有空闲模式唤醒时都需要 T_{CSD} （表 31-11 中的参数 38）延时，该延时与所需的其他延时同时发生 （见**第 4.4 节 “ 空闲模式 ”**）。

2： 包括后分频器产生的频率。复位时， INTOSC 默认设为 8 MHz 的 HF-INTOSC。

3： T_{OST} 是振荡器起振定时器的延迟时间 （表 31-11 中的参数 32）。 T_{RC} 是 PLL 锁定延时定时器的延迟时间（表 31-7 中的参数 F12）； 它还可指定为 T_{PLL}。

4： 在 INTOSC 稳定周期 T_{IOBST} （表 31-11 中的参数 39）延时期间，代码继续执行。

5： 时钟源取决于 SCS（OSCCON<1:0>）、IRCF（OSCCON<6:4>）和 FOSC（CONFIG1H<3:0>）位的设置。

5.0 复位

PIC18F66K80 系列器件有以下几种不同类型的复位：

a) 上电复位 （Power-on Reset， POR）
b) 正常工作期间的 MCLR 复位
c) 功耗管理模式下的 MCLR 复位
d) 看门狗定时器 （WDT）复位 （执行程序期间）
e) 配置不匹配（Configuration Mismatch，CM）复位
f) 可编程欠压复位 （Brown-out Reset， BOR）
g) RESET 指令
h) 堆栈满复位
i) 堆栈下溢复位

本节讨论了由 MCLR、POR 和 BOR 产生的复位，并涉及各种起振定时器的工作方式。堆栈复位事件将在**第 6.1.3.4 节 “ 堆栈满和下溢复位 ”** 中讨论。WDT 复位将在**第 28.2 节 “ 看门狗定时器 （WDT） ”** 中讨论。

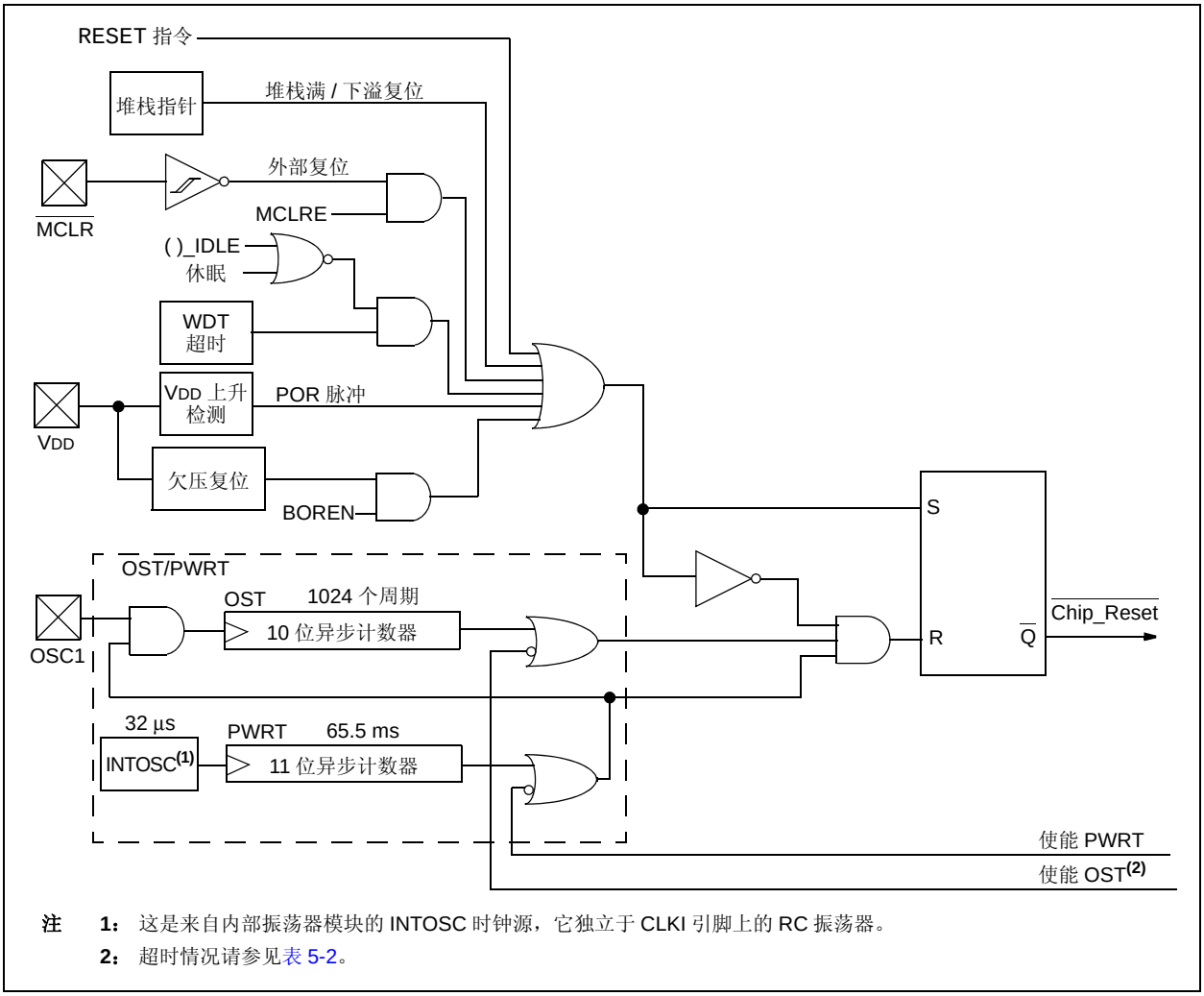
图 5-1 给出了片上复位电路的简化框图。

5.1 RCON 寄存器

通过 RCON 寄存器 （寄存器 5-1）跟踪器件复位事件。该寄存器的低 5 位表明是否已经发生了特定的复位事件。在大多数情况下，只能通过事件将这些位清零，而且必须在事件发生后由应用程序将它们置1。需要读取所有这些标志位来确定刚发生的复位的类型。在**第 5.7 节 “ 寄存器的复位状态 ”** 中对此进行了更详细的说明。

RCON 寄存器还有设置中断优先级的控制位（IPEN）和对 BOR 进行软件控制的控制位 （SBOREN）。将在**第 10.0 节 “ 中断 ”** 中讨论中断优先级。在**第 5.4 节 “ 欠压复位 （BOR） ”** 中讨论 BOR。

图 5-1： 片上复位电路的简化框图



注 1： 这是来自内部振荡器模块的 INTOSC 时钟源，它独立于 CLKI 引脚上的 RC 振荡器。
2： 超时情况请参见表 5-2。

寄存器 5-1：　　RCON：复位控制寄存器

R/W-0	R/W-1(1)	R/W-1	R/W-1	R-1	R-1	R/W-0(2)	R/W-0
IPEN	SBOREN	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
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　**IPEN**：中断优先级使能位
1 = 使能中断优先级
0 = 禁止中断优先级 （PIC16CXXX 兼容模式）

bit 6　**SBOREN**：BOR 软件使能位 (1)
如果 BOREN<1:0> = 01：
1 = 使能 BOR
0 = 禁止 BOR
如果 BOREN<1:0> = 00、 10 或 11：
该位被禁止并读为 0。

bit 5　**$\overline{CM}$**：配置不匹配标志位
1 = 未发生配置不匹配复位
0 = 发生了配置不匹配复位。发生复位后必须用软件置 1。

bit 4　**$\overline{RI}$**：RESET 指令标志位
1 = 未执行 RESET 指令 （只能由固件置 1）
0 = 执行了 RESET 指令，导致器件复位 （发生欠压复位后必须用软件置 1）

bit 3　**$\overline{TO}$**：看门狗超时标志位
1 = 通过上电、 CLRWDT 指令或 SLEEP 指令置 1
0 = 发生了 WDT 超时

bit 2　**$\overline{PD}$**：掉电检测标志位
1 = 通过上电或 CLRWDT 指令置 1
0 = 通过执行 SLEEP 指令置 1

bit 1　**$\overline{POR}$**：上电复位状态位 (2)
1 = 未发生上电复位 （只能由固件置 1）
0 = 发生了上电复位 （发生上电复位后必须用软件置 1）

bit 0　**$\overline{BOR}$**：欠压复位状态位
1 = 未发生欠压复位 （只能由固件置 1）
0 = 发生了欠压复位 （发生欠压复位后必须用软件置 1）

注　1： 如果使能了 SBOREN，其复位状态为 1；否则为 0。
2： $\overline{POR}$ 的实际复位值由器件复位的类型决定。更多信息，请参见该寄存器下方的 “ 注 ” 和**第 5.7 节 “ 寄存器的复位状态 ”**。

注　1： 建议在检测到上电复位后，将 $\overline{POR}$ 位置 1，以便继续检测后续的上电复位。
2： 当 $\overline{POR}$ 为 0 并且 $\overline{POR}$ 为 1 时 （假定在上电复位之后立即用软件将 $\overline{POR}$ 设置为 1），可以说已发生了欠压复位。

5.2 主复位（MCLR）

MCLR 引脚提供了触发器件外部复位的方法。将该引脚拉低可以产生复位信号。该系列器件在 MCLR 复位路径上有一个噪声滤波器，该滤波器可以检测并滤除小的干扰脉冲。

任何内部复位，包括 WDT 复位，均不能将 MCLR 引脚驱动为低电平。

在 PIC18F66K80 系列器件中，可以用 MCLRE 配置位禁止 MCLR 输入。当禁止 MCLR 时，该引脚将成为一个数字输入引脚。更多信息，请参见**第 11.6 节 “PORTE、TRISE 和 LATE 寄存器 ”**。

5.3 上电复位（POR）

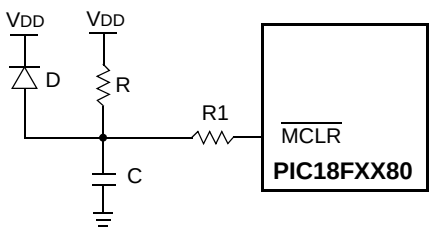
只要当 VDD 上升到高于某个门限时，就会在片上产生上电复位脉冲。这使得 VDD 达到满足器件正常工作的数值时，器件会以初始化状态启动。

为了利用 POR 电路，需要将 MCLR 引脚通过一个电阻（阻值范围为 1 kΩ 至 10 kΩ）连接到 VDD。这样可以省去产生上电复位延时通常所需的外部 RC 元件。VDD 的最小上升速率已指定（参数 D004）。上升速率缓慢的情况，请参见图 5-2。

当器件开始正常工作（即，退出复位状态）时，器件的工作参数（电压、频率和温度等）必须得到满足，以确保其正常工作。如果不满足这些条件，那么器件必须保持在复位状态，直到满足工作条件为止。

POR 事件由 POR 位（RCON<1>）捕捉。每当发生上电复位时，该位的状态就会被设置为 0；任何其他复位事件均不能改变它。任何硬件事件均不能将 POR 复位为 1。要捕捉多个事件，用户必须在上电复位之后用软件手动将该位复位为 1。

图 5-2： 外部上电复位电路（VDD 缓慢上电的情况）



注 1： 仅当 VDD 上电速率过慢时才需要外部上电复位电路。二极管 D 有助于在 VDD 掉电时使电容迅速放电。

2： 建议 R < 40 kΩ，确保电阻 R 两端压降符合器件的电气规范。

3： R1 ≥ 1 kΩ 将限制任何电流从外部电容 C 流入 MCLR，以避免由于静电放电（ESD）或电过载（EOS）导致 MCLR/VPP 引脚损坏。

5.4 欠压复位 （BOR）

PIC18F66K80 系列具有 4 种 BOR 功耗模式：

- 高功耗 BOR
- 中等功耗 BOR
- 低功耗 BOR
- 零功耗 BOR

每种功耗模式通过BORPWR<1:0>设置（CONFIG2L<6:5>）进行选择。对于低功耗、中等功耗和高功耗 BOR，模块会根据 BORV<1:0> 设置 （CONFIG1L<3:2>）来监视 V_{DD}。零功耗、低功耗和中等功耗 BOR 的典型消耗电流（ΔIBOR）分别为 200 nA、750 nA 和 3 µA。BOR 事件会重新激活上电复位。它还会引起复位，具体取决于所设置的跳变电压：1.8V、 2V、 2.7V 或 3V。

可以通过BOREN<1:0>（CONFIG2L<2:1>）和SBOREN位（RCON<6>）来使能 BOR。表 5-1 中总结了 4 种 BOR 配置。

在零功耗 BOR（ZPBORMV）下，模块会监视 V_{DD} 电压，并在电压约为 2V 时重新激活 POR。ZPBORMV 不会导致复位，但会重新激活 POR。

BOR 的精度会随其功耗级别而变。功耗设置越低，BOR 跳变电压的精度就越低。因此，高功耗 BOR 的精度最高，低功耗 BOR 的精度最低。**第 31.0 节 “ 电气特性 ”** 中对跳变电压（B_{VDD}，参数 D005）、电流消耗（**第 31.2 节 “ 直流特性：掉电电流和供电电流 PIC18F66K80 系列（工业级 / 扩展级）”**）和需要低于 B_{VDD} 的时间（T_{BOR}，参数 35）进行了说明。

5.4.1 用软件使能 BOR

当 BOREN<1:0> = 01 时，用户可以用软件使能或禁止 BOR。这通过控制位 SBOREN（RCON<6>）实现。如前所述，将 SBOREN 置 1 可使能 BOR。清零 SBOREN 将完全禁止 BOR。 SBOREN 位只在该模式下工作；否则读为 0。

用软件控制 BOR 位可使用户能更灵活地定制应用程序以使其适应环境，而无需通过对器件再编程来更改BOR配置。它还允许用户通过软件消除 BOR 额外消耗的电流，从而调节器件的功耗。虽然 BOR 的电流通常很小，但是它可能对低功耗应用有一些影响。

注：	即使当 BOR 受软件控制时，欠压复位电压仍将由 BORV<1:0> 配置位设置；该值不能用软件更改。

5.4.2 检测 BOR

使能欠压复位后，在发生任何欠压复位或上电复位时，$\overline{BOR}$ 位总是复位为 0。因此只通过读 $\overline{BOR}$ 的状态很难确定是否发生了欠压复位事件。更可靠的方法是同时检查 $\overline{POR}$ 和 $\overline{BOR}$ 的状态。采用这种方法的前提是在发生任何上电复位事件后， $\overline{POR}$ 位被立即用软件复位为 1。如果 $\overline{BOR}$ 为 0 同时 $\overline{POR}$ 为 1，那么就可以断定已经发生了欠压复位事件。

5.4.3 在休眠模式下禁止 BOR

当 BOREN<1:0> = 10 时，BOR 受硬件控制并且像前面描述的那样工作。每当器件进入休眠模式时，就会自动禁止 BOR。当器件返回到任何其他工作模式时，又将自动重新使能 BOR。

该模式使应用能在有效执行代码的同时从欠压状态恢复，这也是器件最需要 BOR 保护的状况。同时，通过消除少量额外的 BOR 电流，可以节省休眠模式下的额外功耗。

表 5-1： BOR 配置

BOR 配置		SBOREN 的状态（RCON<6>）	BOR 操作
BOREN1	BOREN0		
0	0	不可用	禁止 BOR ； 必须通过对配置位再编程来使能 BOR。
0	1	可用	用软件使能 BOR ； 工作模式由 SBOREN 控制。
1	0	不可用	用硬件在运行和空闲模式下使能 BOR ； 在休眠模式下禁止 BOR。
1	1	不可用	用硬件使能 BOR ； 必须通过对配置位再编程来禁止 BOR。

5.5 配置不匹配 （CM）

配置不匹配 （CM）复位旨在用于检测随机存储器损坏事件，并尝试从事件中恢复。这些事件包括静电放电（ESD）事件，该事件会导致整个器件中大范围的单个位单元内容改变，并导致灾难性的故障。

在 PIC18FXXKXX 闪存器件中，在工作期间会持续监视器件配置寄存器 （位于配置存储空间中），即将它们的值和与之配对的影子寄存器进行比较。如果在两组寄存器之间检测到不匹配，则会自动发生 CM 复位。这些事件通过 $\overline{\text{CM}}$ 位 （RCON<5>）设置为 0 来捕捉。

任何其他复位事件均不能改变该位。 CM 复位的行为类似于主复位、RESET 指令、WDT 超时或堆栈事件复位。类似于所有硬复位和电源复位事件，在器件重启时，会从程序存储器中的闪存配置字重新装入器件配置字。

5.6 器件复位定时器

PIC18F66K80 系列器件包含了三个独立的片上定时器，有助于调节上电复位过程。它们的主要功能是确保在代码执行之前器件时钟稳定。这些定时器是：

- 上电延时定时器 （PWRT）
- 振荡器起振定时器 （OST）
- PLL 锁定延时定时器

5.6.1 上电延时定时器 （PWRT）

PIC18F66K80 系列器件的上电延时定时器 （PWRT）是一个 11 位计数器，它使用 INTOSC 时钟源作为时钟输入。该定时器可产生大约 2048 x 32 µs = 65.6 ms 的时间间隔。 PWRT 计数期间，器件保持在复位状态。

上电延时时间取决于 INTOSC 时钟，并且由于温度和工艺的不同，不同器件的延迟时间也将各不相同。详情请参见直流参数 33。

通过清零 $\overline{\text{PWRTEN}}$ 配置位可使能 PWRT。

5.6.2 振荡器起振定时器 （OST）

在 PWRT 延时 （参数 33）结束以后，由振荡器起振定时器（OST）提供一个 1024 振荡周期（来自 OSC1 输入）的延时，从而确保晶振或谐振器的起振和稳定工作。

只有在 XT、LP、HS 和 HSPLL 模式下，并且仅当发生上电复位或从大多数功耗管理模式退出时，才启动 OST 延时。

5.6.3 PLL 锁定延时定时器

当在 PLL 模式下使能 PLL 时，上电复位后的延时时序与其他振荡器模式略有不同。在 PLL 模式下需要使用一个独立的定时器来提供一段足够让 PLL 锁定主振荡器频率的固定延时。 PLL 锁定延时 （TPLL）通常为 2 ms，且在振荡器起振延时后发生。

5.6.4 延时时序

上电延时时序如下：

1. POR 脉冲清零后，启动 PWRT 延时（如果使能）。
2. 然后， OST 被激活。

总延迟时间取决于振荡器配置和PWRT的状态。图5-3、图 5-4、图 5-5、图 5-6 和图 5-7 各自描述了不同的上电延时时序，其中上电延时定时器被使能，并且器件工作在HS振荡器模式下。图 5-3 至图 5-6 也适用于在XT或LP 模式下工作的器件。对于工作在 RC 模式下且禁止了 PWRT 的器件，将根本没有延时。

由于延时是由 POR 脉冲触发的，因此如果 $\overline{\text{MCLR}}$ 保持足够长时间的低电平，所有延时都将结束。将 $\overline{\text{MCLR}}$ 电平拉高后器件将立即开始执行程序 （图 5-5）。这对于测试或同步多个并行工作的 PIC18FXXXX 器件来说是非常有用的。

表 5-2： 各种情形下的延时

振荡器配置	上电复位和欠压复位		从功耗管理模式退出
	$\overline{\text{PWRTEN}}$ = 0	$\overline{\text{PWRTEN}}$ = 1	
HSPLL	66 ms(1) + 1024 TOSC + 2 ms(2)	1024 TOSC + 2 ms(2)	1024 TOSC + 2 ms(2)
HS、 XT 和 LP	66 ms(1) + 1024 TOSC	1024 TOSC	1024 TOSC
EC 和 ECIO	66 ms(1)	—	—
RC 和 RCIO	66 ms(1)	—	—
INTIO1 和 INTIO2	66 ms(1)	—	—

注 1： 66 ms （65.5 ms）是上电延时定时器 （PWRT）延迟时间的标称值。

2： 2 ms 是 PLL 锁定所需的标称时间。

图 5-3： 上电延时时序 （$\overline{\text{MCLR}}$ 连接到 VDD， VDD 电压上升时间 < TPWRT）

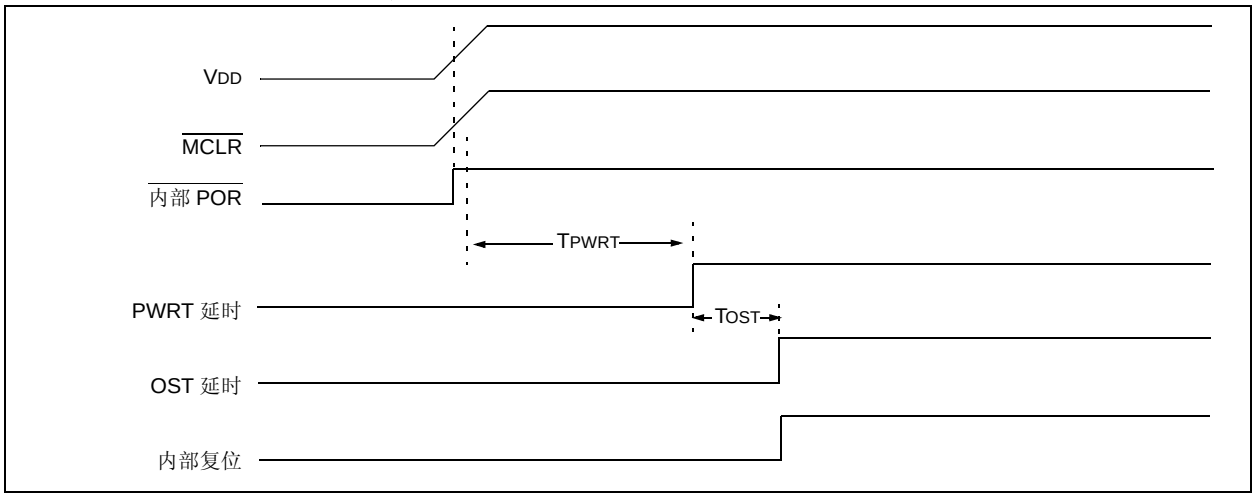


图 5-4：　　上电延时时序（$\overline{\text{MCLR}}$ 未连接到 VDD）：情形 1

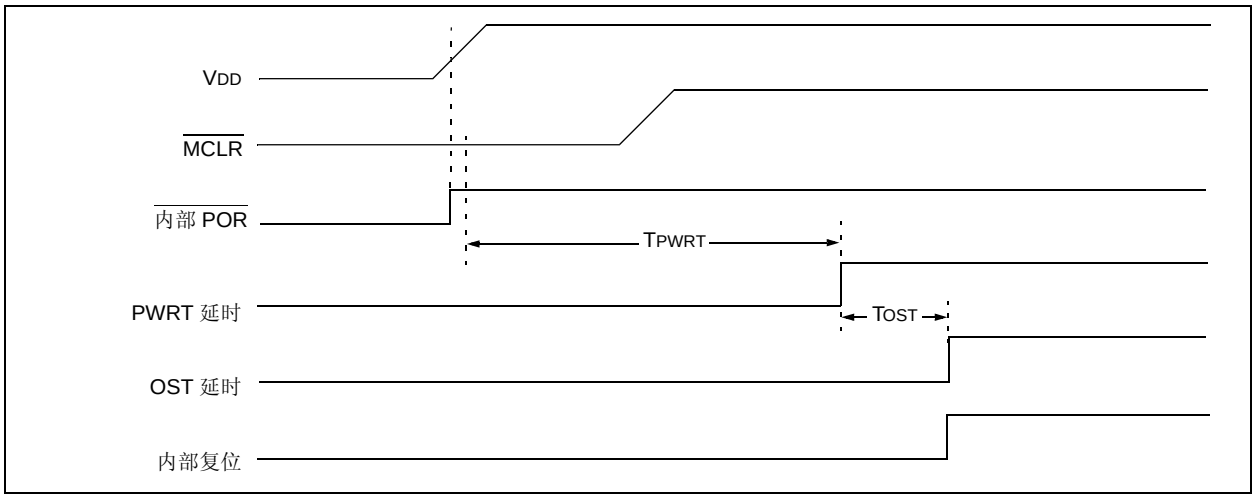


图 5-5：　　上电延时时序（$\overline{\text{MCLR}}$ 未连接到 VDD）：情形 2

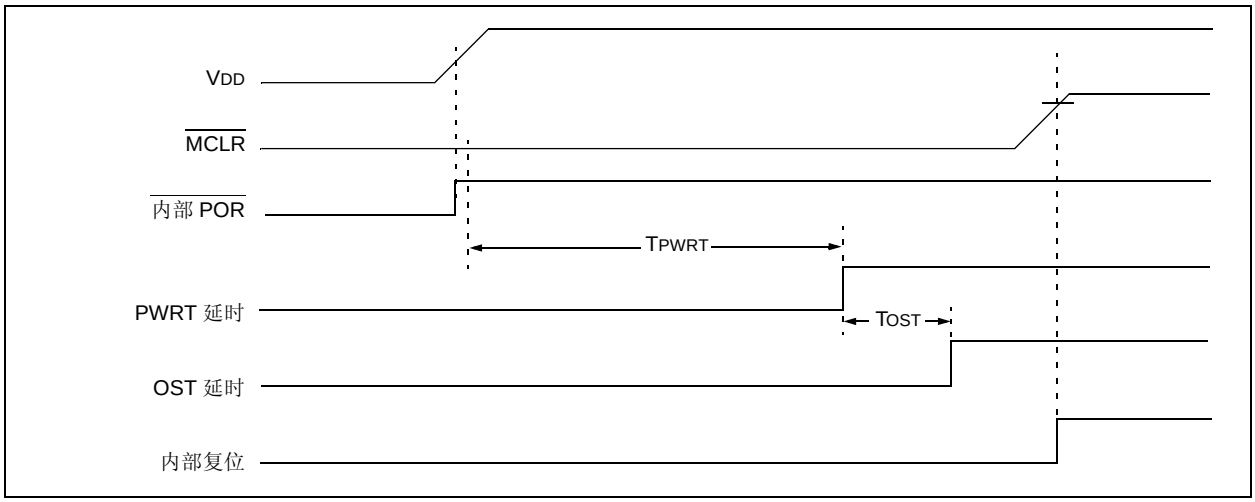


图 5-6：　　缓慢上升时间（$\overline{\text{MCLR}}$ 连接到 VDD，VDD 电压上升时间 > TPWRT）

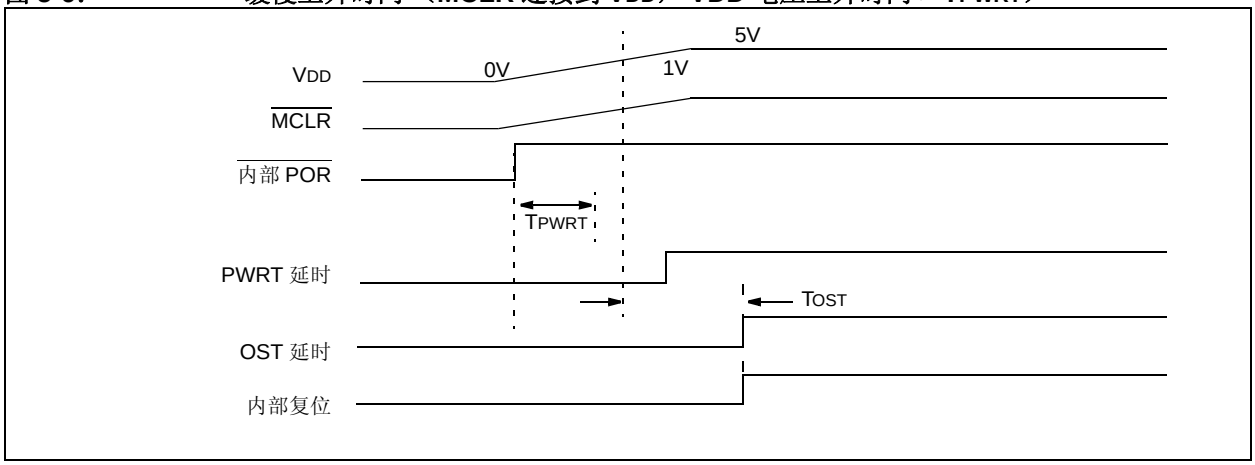
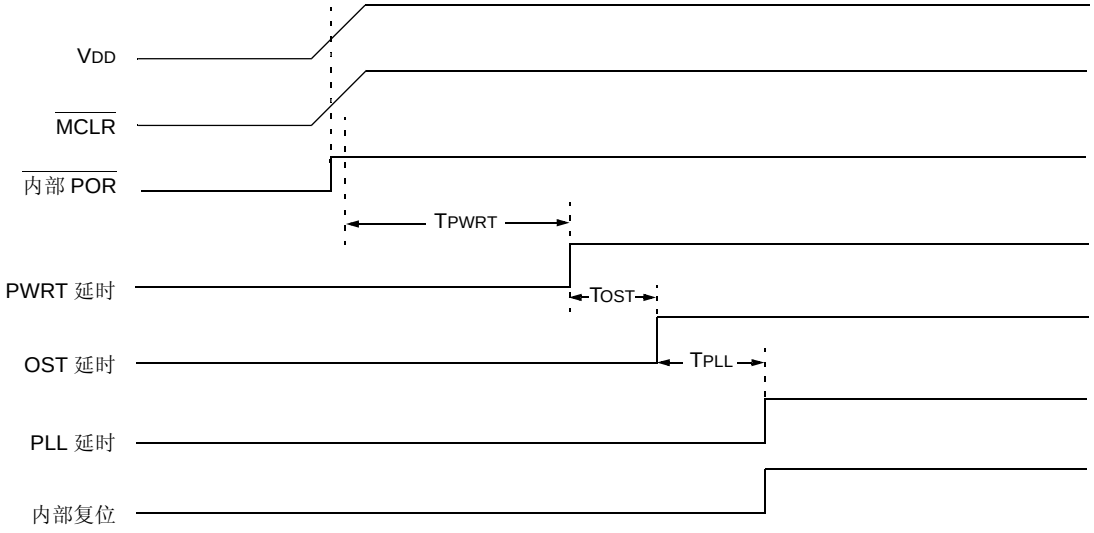


图 5-7：　　在 PLL 使能时 POR 的延时时序 （$\overline{\text{MCLR}}$ 连接到 VDD）



注： TOST = 1024 个时钟周期。
TPLL » 2 ms （最大）， PWRT 定时器延时的前三个阶段。

5.7 寄存器的复位状态

大多数寄存器不受复位的影响。在上电复位时这些寄存器的状态不确定，而在其他复位时它们的状态不变。而其他寄存器则根据不同的复位类型被强制为“复位状态”。

大多数寄存器不受 WDT 唤醒的影响，因为这被视为恢复正常工作。如表 5-3 所示，RCON 寄存器中的状态位（$\overline{RI}$、$\overline{TO}$、$\overline{PD}$、$\overline{CM}$、$\overline{POR}$ 和 $\overline{BOR}$）在不同的复位情形下会分别被置 1 或清零。可在软件中使用这些位判断复位的性质。

表 5-4 描述了所有特殊功能寄存器的复位状态。可以将这些复位状态分类为上电和欠压复位、主复位、WDT 复位以及 WDT 唤醒。

表 5-3： RCON 寄存器的状态位、含义以及初始化状态

条件	程序计数器 (1)	RCON 寄存器							STKPTR 寄存器	
		SBOREN	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
上电复位	0000h	1	1	1	1	1	0	0	0	0
RESET 指令	0000h	u(2)	u	0	u	u	u	u	u	u
欠压复位	0000h	u(2)	1	1	1	1	u	0	u	u
功耗管理运行模式下的 $\overline{MCLR}$ 复位	0000h	u(2)	u	u	1	u	u	u	u	u
功耗管理空闲模式和休眠模式下的 $\overline{MCLR}$ 复位	0000h	u(2)	u	u	1	0	u	u	u	u
全功耗或功耗管理运行模式下的 WDT 超时	0000h	u(2)	u	u	0	u	u	u	u	u
全功耗执行期间的 $\overline{MCLR}$ 复位	0000h	u(2)	u	u	u	u	u	u	u	u
堆栈满复位 （STVREN = 1）	0000h	u(2)	u	u	u	u	u	u	1	u
堆栈下溢复位 （STVREN = 1）	0000h	u(2)	u	u	u	u	u	u	u	1
堆栈下溢错误 （不是真正的复位， STVREN = 0）	0000h	u(2)	u	u	u	u	u	u	u	1
功耗管理空闲或休眠模式下的 WDT 超时	PC + 2	u(2)	u	u	0	0	u	u	u	u
通过中断从功耗管理模式退出	PC + 2	u(2)	u	u	u	0	u	u	u	u

图注： u = 不变

注 1： 当器件被中断唤醒且 GIEH 或 GIEL 置 1 时，PC 装入中断向量 （008h 或 0018h）。

2： 当软件使能 BOR（BOREN<1:0> 配置位 = 01 且 SBOREN = 1）时， POR 的复位状态为 1 且所有其他复位不能改变该状态；否则，其复位状态为 0。

表 5-4：　　所有寄存器的初始化状态

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
TOSU	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---0 0000	---0 0000	---0 uuuu(3)
TOSH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu(3)
TOSL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu(3)
STKPTR	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	00-0 0000	uu-0 0000	uu-u uuuu(3)
PCLATU	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 000x	0000 000u	uuuu uuuu(1)
INTCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 -1-1	1111 -1-1	uuuu -u-u(1)
INTCON3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	11x0 0x00	11x0 0x00	uuuu uuuu(1)
INDF0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
POSTINC0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
POSTDEC0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
PREINC0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
PLUSW0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
FSR0H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---- xxxx	---- uuuu	---- uuuu
FSR0L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
POSTINC1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
POSTDEC1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
PREINC1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
PLUSW1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
FSR1H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---- xxxx	---- uuuu	---- uuuu
FSR1L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---- 0000	---- 0000	---- uuuu
INDF2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响（引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量（0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4：　所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
POSTINC2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
POSTDEC2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
PREINC2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
PLUSW2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	N/A	N/A	N/A
FSR2H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---- xxxx	---- uuuu	---- uuuu
FSR2L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	uuuu uuuu	uuuu uuuu
TMR0L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0100 q000	0100 00q0	uuuu uuqu
OSCCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-x-x x-xx	-0-0 0-01	-u-u u-uu
WDTCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0-x0 -xx0	0-x0 -xx0	u-u0 -uu0
RCON(4)	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0111 11q0	0111 qquu	uuuu qquu
TMR1H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
T2CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-000 0000	-uuu uuuu
SSPBUF	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	uuuu uuuu	uuuu uuuu
SSPADD	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SSPCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SSPCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
ADRESH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-000 0000	-uuu uuuu
ADCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0qqq	0000 0qqq	uuuu uuuu
ADCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0-00 0000	0-00 0000	u-uu uuuu
ECCP1AS	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	xxxx xxxx
CCPR1H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
TXSTA2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0010	0000 0010	uuuu uuuu
BAUDCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	01x0 0-00	01x0 0-00	uuuu u-uu
IPR4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 -111	1111 -111	uuuu -uuu
PIR4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 -000	0000 -000	uuuu -uuu
PIE4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 -000	0000 -000	uuuu -uuu
CVRCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CMSTAT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	11-- ----	11-- ----	uu-- ----
TMR3H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
T3GCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0x00	0000 0x00	uuuu u-uu
SPBRG1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	0000 0000	uuuu uuuu
TXSTA1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0010	0000 0010	uuuu uuuu
RCSTA1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 000x	0000 000x	uuuu uuuu
T1GCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0x00	0000 0x00	uuuu u-uu
PR4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
HLVDCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	01x0 0-00	01x0 0-00	uuuu u-uu
RCSTA2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 000x	0000 000x	uuuu uuuu
IPR3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--11 111-	--11 111-	--uu uuu-
PIR3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--x0 xxx-	--x0 xxx-	--uu uuu-
PIE3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
IPR2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1--- 111x	1--- 111x	u--- uuuu
PIR2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0--- 000x	0--- 000x	u--- uuuu(1)
PIE2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0--- 000x	0--- 0000	u--- uuuu
IPR1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-111 1111	-111 1111	-uuu uuuu
PIR1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu(1)
	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-000 0000	-uuu uuuu
PIE1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-000 0000	-uuu uuuu
PSTR1CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	00-0 0001	xx-x xxxx	—

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4：　所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
OSCTUNE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
REFOCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0-00 0000	0-00 0000	u-uu uuuu
CCPTMRS	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---1 1111	---1 1111	---u uuuu
TRISG	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---1 111	---1 1111	---u uuuu
TRISF	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
TRISE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 -111	1111 -111	uuuu -uuu
TRISD	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
TRISA(5)	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	111- 1111(5)	111- 1111(5)	uuu- uuuu(5)
ODCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SLRCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-111 1111	-111 1111	-111 1111
LATG	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---x xxxx	---x xxxx	---u uuuu
LATF	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx -xxx	xxxx -xxx	uuuu -uuu
LATE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
LATD	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
LATC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
LATB	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
LATA(5)	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- xxxx(5)	xxx- xxxx(5)	uuu- uuuu(5)
T4CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-000 0000	-000 0000	-uuu uuuu
TMR4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
PORTG	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---x xxxx	---x xxxx	---u uuuu
PORTF	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTD	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTB	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTA(5)	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- xxxx(5)	xxx- xxxx(5)	uuu- uuuu(5)
EECON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xx-0 x000	uu-0 u000	uu-u uuuu
EECON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SPBRGH1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SPBRGH2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SPBRG2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
TXREG2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
IPR5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
PIR5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
PIE5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
EEADRH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---- --00	---- --00	---- --00
EEADR	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
EEDATA	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
ECANCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0001 0000	0001 0000	uuuu uuuu
COMSTAT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CIOCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 ---0	0000 ---0	uuuu ---u
CANCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
CANSTAT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
RXB0D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0xxx xxxx	0uuu uuuu	uuuu uuuu
RXB0EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x-xx	uuuu u-uu	uuuu u-uu
RXB0SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB0CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXB0CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CM1CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0001 1111	0001 1111	uuuu uuuu
CM2CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0001 1111	0001 1111	uuuu uuuu
ANCON0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
ANCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-111 1111	-111 1111	-uuu uuuu
WPUB	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 1111	1111 1111	uuuu uuuu
IOCB	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1111 ----	1111 ----	uuuu ----
PMD0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4：　所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
PMD1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
PMD2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---- 0000	---- 0000	---- uuuu
PADCFG1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 ---0	0000 ---0	uuuu ---u
CTMUCONH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0-00 0000	0-00 0000	u-uu uuuu
CTMUCONL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CTMUICONH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CCPR2H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR2L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP2CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--00 0000	--00 0000	--uu uuuu
CCPR3H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR3L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP3CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--00 0000	--00 0000	--uu uuuu
CCPR4H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR4L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP4CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--00 0000	--00 0000	--uu uuuu
CCPR5H	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR5L	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP5CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	--00 0000	--00 0000	--uu uuuu
PSPCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 ----	0000 ----	uuuu ----
MDCON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0010 0--0	0010 0--0	uuuu u--u
MDSRC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0--- xxxx	0--- xxxx	u--- uuuu
MDCARH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0xx- xxxx	0xx- xxxx	uuu- uuuu
MDCARL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0xx- xxxx	0xx- xxxx	uuu- uuuu
CANCON_RO0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
CANSTAT_RO0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
RXB1D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0xxx xxxx	0uuu uuuu	xxxx xxxx

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
RXB1EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x0xx	uuuu u0uu	uuuu uuuu
RXB1SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXB1CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CANCON_RO1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
CANSTAT_RO1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
TXB0D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB0D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB0D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB0D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB0D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB0D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB0D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB0D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB0DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXB0EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXB0EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXB0SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	xxx- x-xx	uuuu uuuu
TXB0SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXB0CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu u-uu
CANCON_RO2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
CANSTAT_RO2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
TXB1D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-x-- xxxx	-u-- uuuu	-u-- uuuu
TXB1EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4：　　所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
TXB1SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x-xx	uuuu u-uu	uuuu u-uu
TXB1SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB1CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CANCON_RO3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
CANSTAT_RO3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
TXB2D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-x-- xxxx	-u-- uuuu	-u-- uuuu
TXB2EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
TXB2SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXB2CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0-00	0000 0-00	uuuu u-uu
RXM1EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXM1EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXM1SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXM1SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXM0EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXM0EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXM0SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXM0SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF5EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF5EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF5SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF5SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF4EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF4EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF4SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
RXF4SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF3EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF3EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF3SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF3SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF2EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF2EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF2SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF2SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF1EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF1EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF1SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF1SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF0EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF0EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF0SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF0SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
CANCON_RO4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
CANSTAT_RO4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
B5D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-xxx xxxx	-uuu uuuu	uuuu uuuu
B5EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x-xx	uuuu u-uu	uuuu uuuu
B5SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B5CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CANCON_RO5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4：　所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
CANSTAT_RO5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
B4D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-xxx xxxx	-uuu uuuu	-uuu uuuu
B4EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B4SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x-xx	uuuu u-uu	uuuu u-uu
B4SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	xxxx xxxx
B4CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CANCON_RO6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
CANSTAT_RO6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
B3D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-xxx xxxx	-uuu uuuu	-uuu uuuu
B3EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x-xx	uuuu u-uu	uuuu u-uu
B3SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B3CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CANCON_RO7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
B2D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。
2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。
3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。
4： 具体条件下的复位值，请参见表 5-3。
5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。
6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
B2D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-xxx xxxx	-uuu uuuu	-uuu uuuu
B2EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x-xx	uuuu u-uu	uuuu u-uu
B2SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B2CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CANCON_RO8	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
B1D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-xxx xxxx	-uuu uuuu	-uuu uuuu
B1EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x-xx	uuuu u-uu	uuuu u-uu
B1SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B1CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
CANCON_RO9	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
CANSTAT_RO9	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	1000 0000	1000 0000	uuuu uuuu
B0D7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0D6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0D5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0D4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0D3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4：　所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	$\overline{\text{MCLR}}$ 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
B0D2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0D1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0D0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0DLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	-xxx xxxx	-uuu uuuu	-uuu uuuu
B0EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx x-xx	uuuu u-uu	uuuu uuuu
B0SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
B0CON	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TXBIE	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---0 00--	---u uu--	---u uu--
BIE0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
BSEL0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 00--	0000 00--	uuuu uu--
MSEL3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
MSEL2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
MSEL1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
MSEL0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0101 0000	0101 0000	uuuu uuuu
RXFBCON7	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXFBCON6	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXFBCON5	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXFBCON4	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXFBCON3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXFBCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0001 0001	0001 0001	uuuu uuuu
RXFBCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0001 0001	0001 0001	uuuu uuuu
RXFBCON0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
SDFLC	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	---0 0000	---0 0000	---u uuuu
RXF15EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF15EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF15SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF15SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF14EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF14EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF14SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF14SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF13EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4： 所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
RXF13EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF13SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF13SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF12EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF12EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF12SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF12SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF11EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF11EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF11SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF11SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF10EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF10EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF10SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF10SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF9EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF9EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF9SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF9SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF8EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF8EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF8SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF8SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF7EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF7EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF7SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF7SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF6EIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF6EIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXF6SIDL	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxx- x-xx	uuu- u-uu	uuu- u-uu
RXF6SIDH	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	xxxx xxxx	uuuu uuuu	uuuu uuuu
RXFCON0	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXFCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
BRGCON3	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	00-- -000	00-- -000	—

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

表 5-4：　　所有寄存器的初始化状态 （续）

寄存器	适用器件			上电复位，欠压复位	MCLR 复位，WDT 复位，RESET 指令，堆栈复位	通过 WDT 或中断唤醒
BRGCON2	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
BRGCON1	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
TXERRCNT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu
RXERRCNT	PIC18F2XK80	PIC18F4XK80	PIC18F6XK80	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

注 1： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （引起唤醒）。

2： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

3： 当器件被中断唤醒且 GIEL 或 GIEH 位置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4： 具体条件下的复位值，请参见表 5-3。

5： 根据所选的振荡器模式使能 PORTA、 LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 0。

6： 将 ECAN™ 技术配置为在模式 1 或模式 2 下时，该寄存器将读为全 0。

注：

6.0 存储器构成

PIC18F66K80 系列器件有三种类型的存储器：

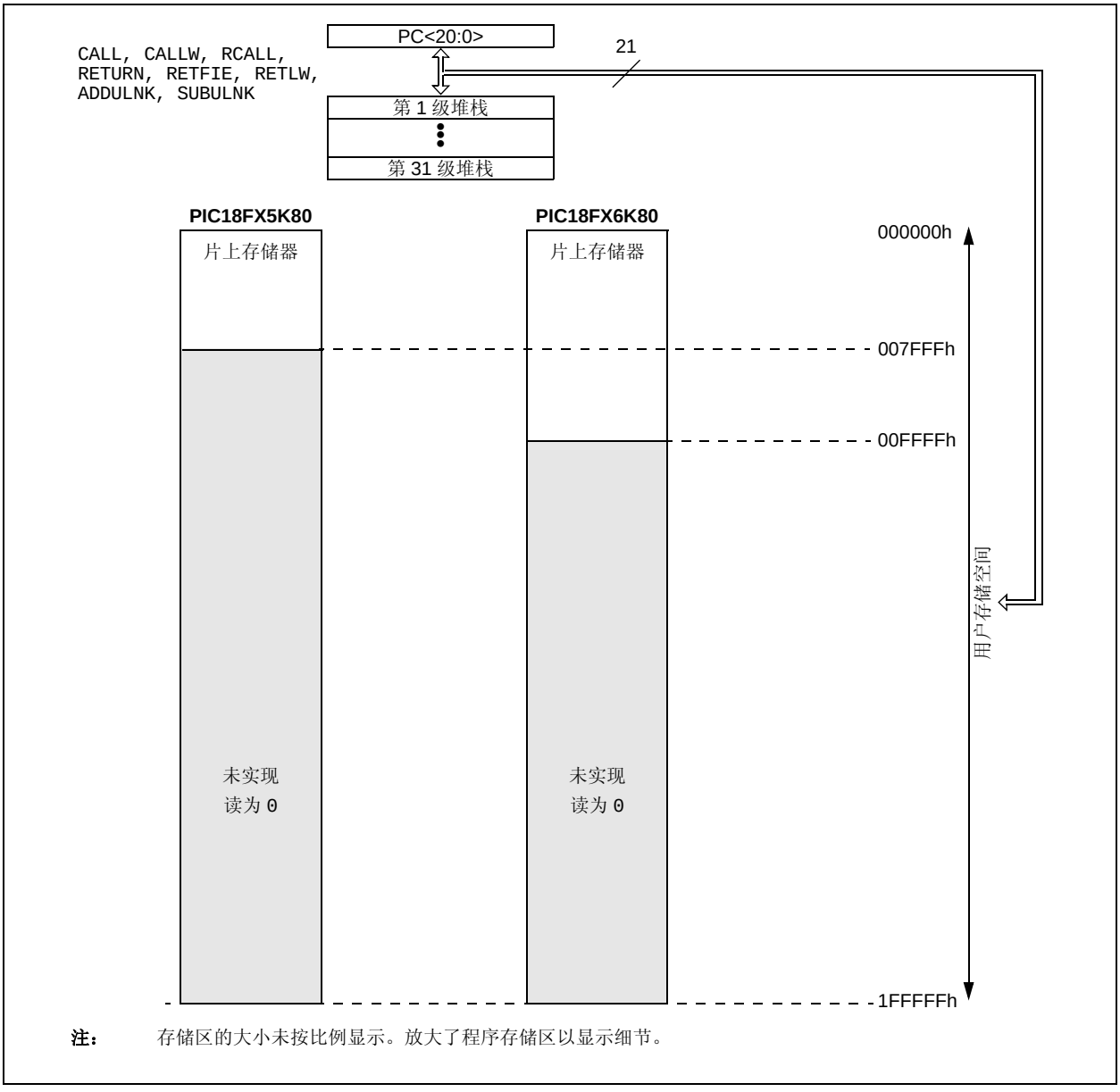
- 程序存储器
- 数据 RAM
- 数据 EEPROM

由于是哈佛架构的器件，数据和程序存储器使用不同的总线，因而可同时访问这两种存储空间。

实际使用时，可将数据 EEPROM 当作外设，因为它可以通过一组控制寄存器进行寻址和访问。

第 7.0 节 “ 闪存程序存储器 ” 提供了关于闪存程序存储器操作的更多详细信息。数据 EEPROM 将单独在**第 8.0 节 “ 数据 EEPROM 存储器 ”** 中讨论。

图 6-1： PIC18F66K80 系列器件的存储器映射



注： 存储区的大小未按比例显示。放大了程序存储区以显示细节。

6.1 程序存储器构成

PIC18 单片机具有一个 21 位程序计数器，可以对 2 MB 的程序存储空间进行寻址。访问物理实现存储器的上边界和 2 MB 地址之间的存储单元会返回全 0（NOP 指令）。

整个 PIC18F66K80 系列提供了各种容量的片上闪存程序存储器，从 32 KB（16,384 条单字指令）到 64 KB（32,768 条单字指令）。

- PIC18F25K80、PIC18F45K80 和 PIC18F65K80——32 KB 闪存，最多存储 16,384 条单字指令
- PIC18F26K80、PIC18F46K80 和 PIC18F66K80——64 KB 闪存，最多存储 32,768 条单字指令

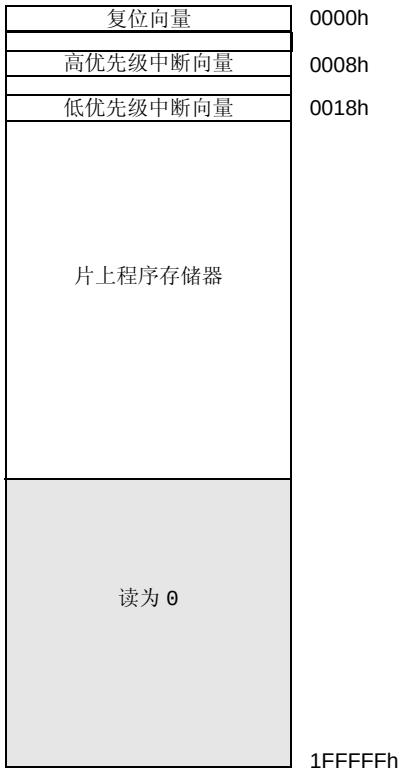
图 6-1 给出了本系列中各器件的程序存储器映射。

6.1.1 存储器硬编码向量

所有的 PIC18 器件在它们的程序存储空间内共有 3 个硬编码的返回向量。复位向量地址是在器件发生任何复位时程序计数器返回到的默认值。它位于 0000h。

PIC18 器件还有两个中断向量地址，用于处理高优先级和低优先级中断。高优先级中断向量位于 0008h，低优先级中断向量位于 0018h。图 6-2 给出了这些向量单元在程序存储器映射中的相对位置。

图 6-2： PIC18F66K80 系列器件的硬编码向量



图注： （存储器的顶部）代表片上程序存储空间的上边界 （关于特定器件的具体值，请参见图 6-1）。阴影区域表示未实现的存储区。这些区域未按比例显示。

6.1.2 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，保存在三个不同的 8 位寄存器中。

存储低字节的寄存器称为 PCL 寄存器，该寄存器可读写。存储高字节的寄存器，即 PCH 寄存器，存储 PC<15:8> 位；该寄存器不可直接读写。更新 PCH 寄存器的操作是通过 PCLATH 寄存器实现的。存储最高字节的寄存器称为 PCU。该寄存器存储 PC<20:16> 位；它也不能直接读写。更新 PCU 寄存器的操作是通过 PCLATU 寄存器实现的。

PCLATH 和 PCLATU 的内容通过执行写 PCL 的任何操作被传送到程序计数器。同样，程序计数器的两个高字节通过读 PCL 的操作被传送到 PCLATH 和 PCLATU。这对于 PC 计算偏移量很有用处（见**第 6.1.5.1 节 “ 计算 GOTO”**）。

PC 是按字节寻址程序存储器的。为了防止 PC 不能与字指令对齐，需要将 PCL 的最低有效位（Least Significant bit，LSb）固定取值为 0。PC 每次加 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。

6.1.3 返回地址堆栈

返回地址堆栈允许执行最多 31 个程序调用和中断的任意组合。当执行 CALL、RCALL 指令或响应中断时，PC 值会被压入该堆栈。而在执行 RETURN、RETLW 或 RETFIE 指令时，PC 值会从堆栈弹出。如果使能了扩展指令集，则在执行 ADDULNK 和 SUBULNK 指令时，PC 值也会从堆栈弹出。PCLATU 和 PCLATH 不受 RETURN 或 CALL 指令的影响。

通过 21 位的 RAM 和一个 5 位的堆栈指针 STKPTR 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，并且通过栈顶（Top-of-Stack，TOS）特殊功能寄存器可以读写栈顶地址。也可以使用这些寄存器将数据压入堆栈或者从堆栈弹出。

执行 CALL 类型指令进行压栈操作：首先堆栈指针加 1，并且将 PC（PC 已经指向 CALL 之后的下一条指令）的内容写入堆栈指针所指向的地址单元。执行 RETURN 类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，然后堆栈指针减 1。

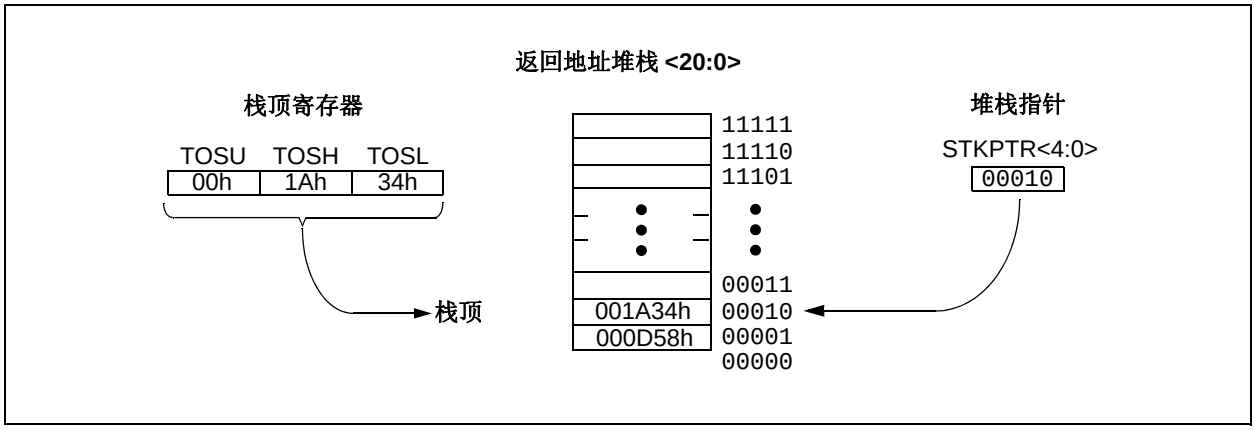
所有复位后，堆栈指针均会初始化为 00000。堆栈指针值 00000 不指向任何 RAM 单元；它仅仅是一个复位值。状态位指示堆栈是已满、上溢还是下溢。

6.1.3.1 访问栈顶

只可读写返回地址堆栈的栈顶。有三个寄存器 TOSU:TOSH:TOSL 用于保存由 STKPTR 寄存器所指向的堆栈单元的内容（图 6-3）。这可以让用户在必要时实现软件堆栈。在 CALL、RCALL 或中断（如果使能了扩展指令集，则还包括 ADDULNK 和 SUBULNK 指令）后，软件可以通过读取 TOSU:TOSH:TOSL 寄存器来读取压入堆栈的值。这些值可以被存放到由用户定义的软件堆栈。返回时，软件将这些值存回 TOSU:TOSH:TOSL 并执行返回。

为防止意外的堆栈操作，访问堆栈时用户必须禁止全局中断允许位。

图 6-3： 返回地址堆栈和相关的寄存器



6.1.3.2 返回堆栈指针 （STKPTR）

STKPTR寄存器（寄存器6-1）包含堆栈指针值、STKFUL（堆栈满）状态位和 STKUNF （堆栈下溢）状态位。堆栈指针值可为 0 至 31 范围内的任意值。向堆栈压入值前，堆栈指针加 1；而从堆栈弹出值后，堆栈指针减 1。复位时，堆栈指针值为零。

用户可以读写堆栈指针的值。实时操作系统（Real-Time Operating System， RTOS）可以利用此特性对返回堆栈进行维护。

向堆栈压入 PC 值 31 次（且没有值从堆栈弹出）后，STKFUL 位置 1。通过软件或 POR 将 STKFUL 位清零。

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。（关于器件配置位的说明，请参见**第 28.1 节 “ 配置位 ”**。）如果 STVREN 置 1（默认），第31次压栈会将（PC + 2）值压入堆栈，从而将STKFUL位置 1 并复位器件。 STKFUL 位将保持置 1，而堆栈指针将被清零。

如果 STVREN 清零，第 31 次压栈时 STKFUL 位将会置 1，堆栈指针值递增到 31。任何其他压栈操作都不会覆盖第 31 次进栈的值，并且 STKPTR 将保持为 31。

当堆栈弹出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，而堆栈指针则保持为零。STKUNF 位将保持置 1，直到由软件清零或发生 POR 为止。

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。

6.1.3.3 PUSH 和 POP 指令

由于栈顶是可以读写的，因此将值压入堆栈或从堆栈弹出而不影响程序的正常执行是非常理想的。PIC18 指令集包含两条指令 PUSH 和 POP，使用这两条指令可在软件控制下对 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 指令将当前的 PC 值压入堆栈。执行该指令会使堆栈指针加 1 并将当前的 PC 值装入堆栈。

POP 指令通过将堆栈指针减 1 来放弃当前的 TOS 值。然后前一个入栈的值就成为了 TOS 值。

寄存器 6-1： STKPTR：堆栈指针寄存器

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL(1)	STKUNF(1)	—	SP4	SP3	SP2	SP1	SP0
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 **STKFUL：**堆栈满标志位 (1)
1 = 堆栈已满或已溢出
0 = 堆栈未满也未溢出

bit 6 **STKUNF：**堆栈下溢标志位 (1)
1 = 发生了堆栈下溢
0 = 未发生堆栈下溢

bit 5 **未实现：**读为 0

bit 4-0 **SP<4:0>：**堆栈指针地址位

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。

6.1.3.4 堆栈满和下溢复位

通过将 STVREN 位（CONFIG4L<0>）置 1，来使能在堆栈溢出或下溢时的器件复位。当 STVREN 置 1 时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，然后使器件复位。当 STVREN 清零时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，但不会使器件复位。只能通过用户软件或上电复位将 STKFUL 或 STKUNF 位清零。

6.1.4 快速寄存器堆栈

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。此堆栈只有一层深且不可读写。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工作寄存器。

如果同时允许低优先级中断和高优先级中断，从低优先级中断返回时，无法可靠地使用堆栈寄存器。如果在处理低优先级中断时，发生了高优先级中断，则低优先级中断存储在堆栈寄存器中的值将被覆盖。在这种情况下，在处理低优先级中断时用户必须用软件保存关键寄存器的值。

如果未使用中断优先级，所有中断都可以使用快速寄存器堆栈从中断返回。如果没有使用中断，快速寄存器堆栈可以用于在子程序调用结束后恢复 STATUS、WREG 和 BSR 寄存器。要在子程序调用中使用快速寄存器堆栈，必须执行 CALL label, FAST 指令，将 STATUS、WREG 和 BSR 寄存器的内容存入快速寄存器堆栈。然后执行 RETURN, FAST 指令，从快速寄存器堆栈恢复这些寄存器。

例 6-1 给出了一个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 6-1： 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
```

6.1.5 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，可以用两种方式实现查找表：

- 计算 GOTO
- 表读

6.1.5.1 计算 GOTO

计算 GOTO 是通过向程序计数器加一个偏移量来实现的。例 6-2 中给出了一个示例。

可以使用 ADDWF PCL 指令和一组 RETLW nn 指令创建一个查找表。在调用该表前，会先将查找表中的偏移量装入 W 寄存器。被调用子程序的第一条指令应该是 ADDWF PCL 指令。接下去执行的一条是 RETLW nn 指令，它将值 nn 返回给调用函数。

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

在这种方式中，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 6-2： 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
        .
        .
        .
```

6.1.5.2 表读

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。

编程时，每个程序字可以存储 2 个字节的查找表数据。表指针（TBLPTR）指定字节地址，而表锁存器（TABLAT）则存储从程序存储器读取的数据。一次只能从程序存储器读取一个字节。

第 7.1 节“表读与表写”将进一步讨论表读操作。

6.2 PIC18 指令周期

6.2.1 时钟机制

来自内部或外部时钟源的单片机时钟输入都将在内部被四分频以产生四个互不重叠的正交时钟信号（Q1、Q2、Q3 和 Q4）。在单片机内部，程序计数器在每个 Q1 递增；在 Q4 期间，从程序存储器取指令并将指令锁存到指令寄存器 （Instruction Register， IR）中。

指令的译码和执行在下一个 Q1 至 Q4 周期完成。图 6-4 所示为时钟和指令执行流程。

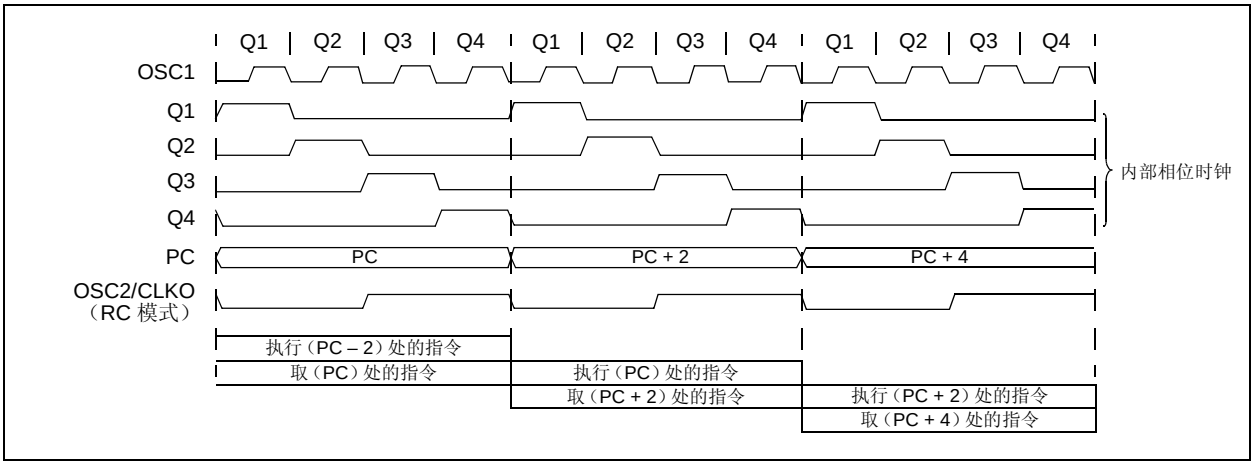
6.2.2 指令流 / 流水线

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 （见例 6-3）。

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（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 6-4： 时钟 / 指令周期



例 6-3： 指令流水线流程

	TCY0	TCY1	TCY2	TCY3	TCY4	TCY5
1. MOVLW 55h	取指 1	执行 1				
2. MOVWF PORTB		取指 2	执行 2			
3. BRA SUB_1			取指 3	执行 3		
4. BSF PORTA, BIT3 (Forced NOP)				取指 4	清除 （NOP）	
5. Instruction @ address SUB_1					取指 SUB_1	执行 SUB_1

除程序跳转指令外，所有的指令都是单周期指令。由于程序跳转指令将导致一条已取指令从流水线清除，与此同时取新指令，然后执行指令，所以程序跳转指令需要两个周期。

6.2.3 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 （Least Significant Byte，LSB）始终存储在地址为偶数的程序存储单元中（LSB = 0）。要保证与指令边界对齐，PC 必须以 2 为单位递增，并且 LSB 总是读为 0（见**第 6.1.2 节 “ 程序计数器 ”**）。

图 6-5 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。由于指令总是按字边界存储，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 6-5 中的指令 2 给出了指令 GOTO 0006h 在程序存储器中的译码过程。程序跳转指令也采取同样的方式对相对地址偏移量进行译码。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。关于指令集的更多详细信息，请参见**第 29.0 节 “ 指令集汇总 ”**。

图 6-5： 程序存储器中的指令

程序存储器
字节单元 →

			LSB = 1	LSB = 0	字地址 ↓
					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
					000012h
					000014h

6.2.4 双字指令

标准的 PIC18 指令集有 4 条双字指令：CALL、MOVFF、GOTO 和 LSFR。这些指令第二个字的高 4 位 （Most Significant bit，MSb）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。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。如果由于某些原因跳过了第一个字而自动执行指令的第二个字，那么将作为一条 NOP 指令执行。如果双字指令跟在修改 PC 的条件指令后，就有必要执行此操作。例 6-4 给出了它的执行过程。

注： 关于扩展指令集中的双字指令信息，请参见**第 6.5 节 “ 程序存储器和扩展指令集 ”**。

例 6-4： 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	; No, 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	; Yes, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

6.3 数据存储器构成

注： 当使能了 PIC18 扩展指令集时，数据存储器某些方面的操作会有所改变。更多信息，请参见**第 6.6 节 “ 数据存储器和扩展指令集 ”**。

PIC18 器件中的数据存储器是用静态 RAM 实现的。在数据存储器中，每个寄存器都有 12 位地址，允许实现最大为 4,096 个字节的数据存储器。存储空间被分为 16 个存储区，每个存储区包含 256 个字节。

图 6-6 和图 6-7 给出了器件的数据存储器构成。

数据存储器由特殊功能寄存器（Special Function Register，SFR）和通用寄存器（General Purpose Register，GPR）组成。SFR 用于单片机和外设功能模块的控制和状态指示，GPR 则用于用户应用程序的数据存储和中间结果暂存。任何未实现的存储单元均读为 0。

这样的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。本章后面的部分将讨论寻址模式。

为确保能在一个周期中访问常用寄存器（某些 SFR 和 GPR），PIC18 器件实现了一个快速操作存储区。该存储区是一个 256 字节的存储空间，它可实现对 SFR 和 GPR Bank 0 的低地址单元的快速访问，而无需使用存储区选择寄存器。关于快速操作 RAM 的详细信息，请参见**第 6.3.2 节 “ 快速操作存储区 ”**。

6.3.1 存储区选择寄存器

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。理想状况下，这意味着不必为每次读写操作提供完整地址。PIC18 器件是使用 RAM 存储区分区机制实现快速访问的。这种机制将存储空间分成连续的 16 个 256 字节的存储区。根据不同的指令，可以通过完整的 12 位地址，或通过 8 位的低字节地址和 4 位存储区指针直接寻址每个存储单元。

PIC18 指令集中的大部分指令都使用存储区指针，也就是存储区选择寄存器（Bank Select Register，BSR）。这个 SFR 保存单元地址的高 4 位，而指令本身则包括低 8 位。只使用 BSR 的低 4 位（BSR<3:0>），不使用高 4 位，高 4 位总是读为 0 且不能被写入。可以通过使用 MOVLB 指令直接装入 BSR。

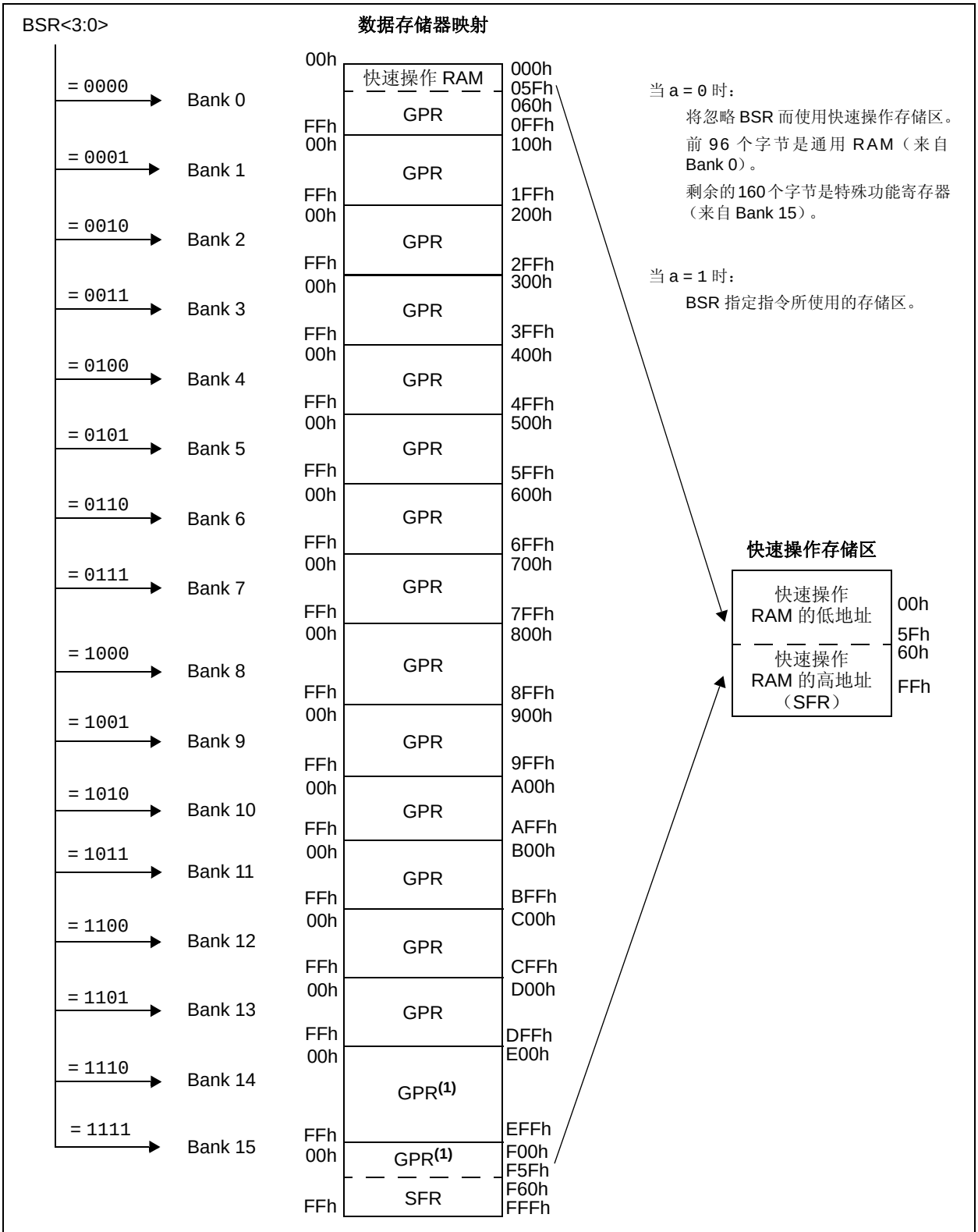
BSR 的值代表数据存储器中的存储区，指令中的 8 位指向存储区中的存储单元，可以将它看作距离存储区下边界的偏移量。图 6-7 显示了 BSR 的值与数据存储器中的存储区之间的关系。

由于最多可有 16 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之前选择了正确的存储区。例如，当 BSR 为 0Fh 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复位。

当选择存储区时，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 6-6 中的数据存储器映射指出了已实现的存储区。

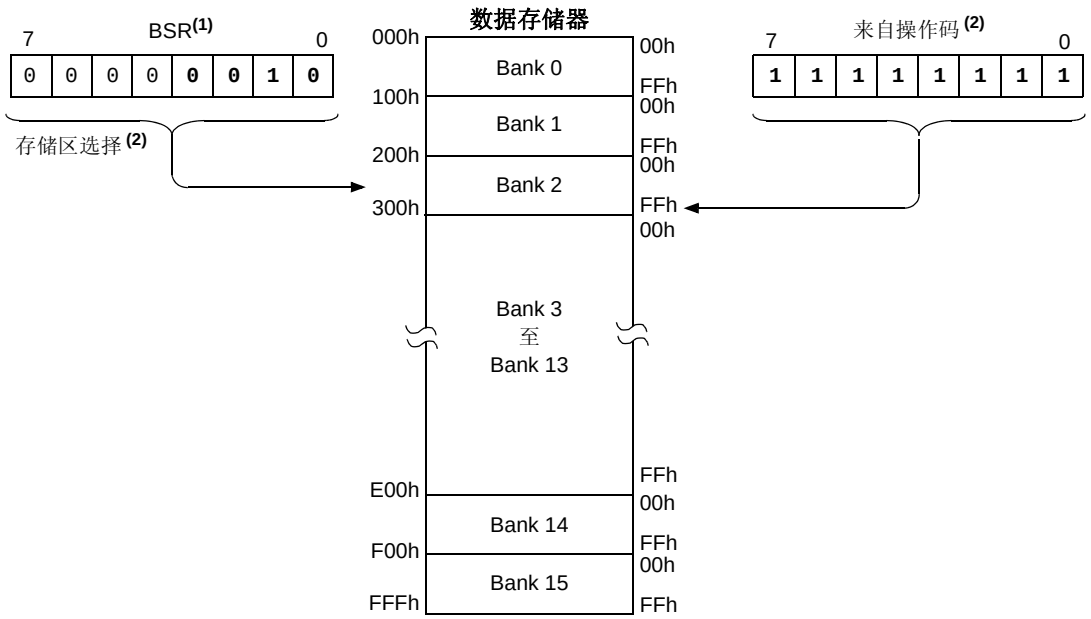
在核心 PIC18 指令集中，只有 MOVFF 指令指定源寄存器和目标寄存器的完整 12 位地址。该指令在执行时完全忽略 BSR。所有其他指令仅包含作为操作数的低位地址，而且必须使用 BSR 或快速操作存储区来寻址目标寄存器。

图 6-6：　　PIC18FX5K80 和 PIC18FX6K80 器件的数据存储器映射



注　　1： SFR 还使用 E41h 至 F5Fh 的地址，但它们不是快速操作 RAM 的一部分。用户必须总是使用完整的地址或装入正确的 BSR 值，以访问这些寄存器。

图 6-7：　　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<3:0>），直接寻址快速操作存储区的寄存器。

2： MOVFF 指令在指令中嵌入了完整的 12 位地址。

6.3.2　　快速操作存储区

使用 BSR 和嵌入的 8 位地址，用户可以寻址数据存储器的整个空间，但这同时也意味着用户必须确保选择了正确的存储区。否则，可能会从错误的单元读取数据或将数据写入错误的单元。如果本来是向 GPR 进行写操作，却将结果写入了 SFR，后果是非常严重的。但是在每次对数据存储器进行读或写操作时验证和 / 或更改 BSR 会严重影响工作效率。

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 15 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 6-6）。

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。当 “a” 为 0 时，强制指令使用快速操作存储区地址映射，此时完全忽略 BSR 的当前值。

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 进行取值和操作。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 也可实现更加快速和高效的现场保护和变量切换代码。

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 6.6.3 节 “ 在立即数变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。

6.3.3　　通用寄存器文件

PIC18 器件在 GPR 区中划分了一部分存储区。这部分存储区为数据 RAM，所有指令均可访问它。 GPR 区从 Bank 0 的底部（地址 000h）开始向上延伸直到 SFR 区的底部。上电复位不会初始化 GPR，并且其他复位也不会改变其内容。

6.3.4 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 （FFFh）开始向下，它占用了 Bank 15 的全部空间 （F00h 至 FFFh）和 Bank 14 的顶部空间 （EF4h 至 EFFh）。

表 6-1 和表 6-2 列出了这些寄存器。

可以将 SFR 归类为两组：与器件 “ 内核 ” 功能 （ALU、复位和中断）相关的寄存器和与外设功能相关的寄存器。复位和中断寄存器在相应章节中进行讨论，本章后面的部分将对 ALU 的 STATUS 寄存器进行说明。与外设操作相关的寄存器将在相应外设的章节中进行说明。

SFR 通常分布在功能受其控制的外设中。未使用的 SFR 单元是未实现的，读为 0。

表 6-1： PIC18F66K80 系列的特殊功能寄存器映射

地址	名称	地址	名称	地址	名称	地址	名称	地址	名称	地址	名称
FFFh	TOSU	FDFh	INDF2(1)	FBFh	ECCP1AS	F9Fh	IPR1	F7Fh	EECON1	F5Fh	CM1CON(5)
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	ECCP1DEL	F9Eh	PIR1	F7Eh	EECON2	F5Eh	CM2CON(5)
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCPR1H	F9Dh	PIE1	F7Dh	SPBRGH1	F5Dh	ANCON0(5)
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR1L	F9Ch	PSTR1CON	F7Ch	SPBRGH2	F5Ch	ANCON1(5)
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCP1CON	F9Bh	OSCTUNE	F7Bh	SPBRG2	F5Bh	WPUB(5)
FFAh	PCLATH	FDAh	FSR2H	FBAh	TXSTA2	F9Ah	REFOCON	F7Ah	RCREG2	F5Ah	IOCB(5)
FF9h	PCL	FD9h	FSR2L	FB9h	BAUDCON2	F99h	CCPTMRS	F79h	TXREG2	F59h	PMD0(5)
FF8h	TBLPTRU	FD8h	STATUS	FB8h	IPR4	F98h	TRISG(3)	F78h	IPR5	F58h	PMD1(5)
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	PIR4	F97h	TRISF(3)	F77h	PIR5	F57h	PMD2(5)
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	PIE4	F96h	TRISE(4)	F76h	PIE5	F56h	PADCFG1(5)
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD(4)	F75h	EEADRH	F55h	CTMUCONH(5)
FF4h	PRODH	FD4h	—(2)	FB4h	CMSTAT	F94h	TRISC	F74h	EEADR	F54h	CTMUCONL(5)
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	EEDATA	F53h	CTMUICONH(5)
FF2h	INTCON	FD2h	OSCCON2	FB2h	TMR3L	F92h	TRISA	F72h	ECANCON	F52h	CCPR2H(5)
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	ODCON	F71h	COMSTAT	F51h	CCPR2L(5)
FF0h	INTCON3	FD0h	RCON	FB0h	T3GCON	F90h	SLRCON	F70h	CIOCON	F50h	CCP2CON(4,5)
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG(3)	F6Fh	CANCON	F4Fh	CCPR3H(4,5)
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF(3)	F6Eh	CANSTAT	F4Eh	CCPR3L(4,5)
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE(4)	F6Dh	RXB0D7	F4Dh	CCP3CON(5)
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD(4)	F6Ch	RXB0D6	F4Ch	CCPR4H(5)
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC	F6Bh	RXB0D5	F4Bh	CCPR4L(5)
FEAh	FSR0H	FCAh	T2CON	FAAh	T1GCON	F8Ah	LATB	F6Ah	RXB0D4	F4Ah	CCP4CON(5)
FE9h	FSR0L	FC9h	SSPBUF	FA9h	PR4	F89h	LATA	F69h	RXB0D3	F49h	CCPR5H(5)
FE8h	WREG	FC8h	SSPADD	FA8h	HLVDCON	F88h	T4CON	F68h	RXB0D2	F48h	CCPR5L(5)
FE7h	INDF1(1)	FC8h	SSPMSK	FA7h	BAUDCON1	F87h	TMR4	F67h	RXB0D1	F47h	CCP5CON(5)
FE6h	POSTINC1(1)	FC7h	SSPSTAT	FA6h	RCSTA2	F86h	PORTG(3)	F66h	RXB0D0	F46h	PSPCON(4,5)
FE5h	POSTDEC1(1)	FC6h	SSPCON1	FA5h	IPR3	F85h	PORTF(3)	F65h	RXB0DLC	F45h	MDCON(3,5)
FE4h	PREINC1(1)	FC5h	SSPCON2	FA4h	PIR3	F84h	PORTE	F64h	RXB0EIDL	F44h	MDSRC(3,5)
FE3h	PLUSW1(1)	FC4h	ADRESH	FA3h	PIE3	F83h	PORTD(4)	F63h	RXB0EIDH	F43h	MDCARH(3,5)
FE2h	FSR1H	FC3h	ADRESL	FA2h	IPR2	F82h	PORTC	F62h	RXB0SIDL	F42h	MDCARL(3,5)
FE1h	FSR1L	FC2h	ADCON0	FA1h	PIR2	F81h	PORTB	F61h	RXB0SIDH	F41h	—(2)
FE0h	BSR	FC1h	ADCON1	FA0h	PIE2	F80h	PORTA	F60h	RXB0CON	F40h	—(2)
		FC0h	ADCON2								

注
1. 这不是实际存在的寄存器。
2. 未实现的寄存器，读为 0。
3. 该寄存器仅在 64 引脚器件上可用。
4. 该寄存器在 28 引脚器件上不可用。
5. SFR 还使用 E41h 至 F5Fh 的地址，但它们不是快速操作 RAM 的一部分。要访问这些寄存器，用户必须总是装入正确的 BSR 值。

表 6-1： PIC18F66K80 系列的特殊功能寄存器映射 （续）

地址	名称	地址	名称	地址	名称	地址	名称	地址	名称	地址	名称
F3Fh	CANCON_RO0(5)	F0Fh	CANCON_RO3(5)	EDFh	CANCON_RO4(5)	EAFh	CANCON_RO7(5)	E7Fh	TXBIE(5)	E4Fh	RXF7EIDL(5)
F3Eh	CANSTAT_RO0(5)	F0Eh	CANSTAT_RO3(5)	EDEh	CANSTAT_RO4(5)	EAEh	CANSTAT_RO7(5)	E7Eh	BIE0(5)	E4Eh	RXF7EIDH(5)
F3Dh	RXB1D7(5)	F0Dh	TXB2D7(5)	EDDh	B5D7(5)	EADh	B2D7(5)	E7Dh	BSEL0(5)	E4Dh	RXF7SIDL(5)
F3Ch	RXB1D6(5)	F0Ch	TXB2D6(5)	EDCh	B5D6(5)	EACh	B2D6(5)	E7Ch	MSEL3(5)	E4Ch	RXF7SIDH(5)
F3Bh	RXB1D5(5)	F0Bh	TXB2D5(5)	EDBh	B5D5(5)	EABh	B2D5(5)	E7Bh	MSEL2(5)	E4Bh	RXF6EIDL(5)
F3Ah	RXB1D4(5)	F0Ah	TXB2D4(5)	EDAh	B5D4(5)	EAAh	B2D4(5)	E7Ah	MSEL1(5)	E4Ah	RXF6EIDH(5)
F39h	RXB1D3(5)	F09h	TXB2D3(5)	ED9h	B5D3(5)	EA9h	B2D3(5)	E79h	MSEL0(5)	E49h	RXF6SIDL(5)
F38h	RXB1D2(5)	F08h	TXB2D2(5)	ED8h	B5D2(5)	EA8h	B2D2(5)	E78h	RXFBCON7(5)	E48h	RXF6SIDH(5)
F37h	RXB1D1(5)	F07h	TXB2D1(5)	ED7h	B5D1(5)	EA7h	B2D1(5)	E77h	RXFBCON6(5)	E47h	RXFCON0(5)
F36h	RXB1D0(5)	F06h	TXB2D0(5)	ED6h	B5D0(5)	EA6h	B2D0(5)	E76h	RXFBCON5(5)	E46h	RXFCON1(5)
F35h	RXB1DLC(5)	F05h	TXB2DLC(5)	ED5h	B5DLC(5)	EA5h	B2DLC(5)	E75h	RXFBCON4(5)	E45h	BRGCON3(5)
F34h	RXB1EIDL(5)	F04h	TXB2EIDL(5)	ED4h	B5EIDL(5)	EA4h	B2EIDL(5)	E74h	RXFBCON3(5)	E44h	BRGCON2(5)
F33h	RXB1EIDH(5)	F03h	TXB2EIDH(5)	ED3h	B5EIDH(5)	EA3h	B2EIDH(5)	E73h	RXFBCON2(5)	E43h	BRGCON1(5)
F32h	RXB1SIDL(5)	F02h	TXB2SIDL(5)	ED2h	B5SIDL(5)	EA2h	B2SIDL(5)	E72h	RXFBCON1(5)	E42h	TXERRCNT(5)
F31h	RXB1SIDH(5)	F01h	TXB2SIDH(5)	ED1h	B5SIDH(5)	EA1h	B2SIDH(5)	E71h	RXFBCON0(5)	E41h	RXERRCNT(5)
F30h	RXB1CON(5)	F00h	TXB2CON(5)	ED0h	B5CON(5)	EA0h	B2CON(5)	E70h	SDFLC(5)		
F30h	RXB1CON(5)	EFFh	RXM1EIDL(5)	ECFh	CANCON_RO5(5)	E9Fh	CANCON_RO8(5)	E6Fh	RXF15EIDL(5)		
F2Fh	CANCON_RO1(5)	EFEh	RXM1EIDH(5)	ECEh	CANSTAT_RO5(5)	E9Eh	CANSTAT_RO8(5)	E6Eh	RXF15EIDH(5)		
F2Eh	CANSTAT_RO1(5)	EFDh	RXM1SIDL(5)	ECDh	B4D7(5)	E9Dh	B1D7(5)	E6Dh	RXF15SIDL(5)		
F2Dh	TXB0D7(5)	EFCh	RXM1SIDH(5)	ECCh	B4D6(5)	E9Ch	B1D6(5)	E6Ch	RXF15SIDH(5)		
F2Ch	TXB0D6(5)	EFBh	RXM0EIDL(5)	ECBh	B4D5(5)	E9Bh	B1D5(5)	E6Bh	RXF14EIDL(5)		
F2Bh	TXB0D5(5)	EFAh	RXM0EIDH(5)	ECAh	B4D4(5)	E9Ah	B1D4(5)	E6Ah	RXF14EIDH(5)		
F2Ah	TXB0D4(5)	EF9h	RXM0SIDL(5)	EC9h	B4D3(5)	E99h	B1D3(5)	E69h	RXF14SIDL(5)		
F29h	TXB0D3(5)	EF8h	RXM0SIDH(5)	EC8h	B4D2(5)	E98h	B1D2(5)	E68h	RXF14SIDH(5)		
F28h	TXB0D2(5)	EF7h	RXF5EIDL(5)	EC7h	B4D1(5)	E97h	B1D1(5)	E67h	RXF13EIDL(5)		
F27h	TXB0D1(5)	EF6h	RXF5EIDH(5)	EC6h	B4D0(5)	E96h	B1D0(5)	E66h	RXF13EIDH(5)		
F26h	TXB0D0(5)	EF5h	RXF5SIDL(5)	EC5h	B4DLC(5)	E95h	B1DLC(5)	E65h	RXF13SIDL(5)		
F25h	TXB0DLC(5)	EF4h	RXF5SIDH(5)	EC4h	B4EIDL(5)	E94h	B1EIDL(5)	E64h	RXF13SIDH(5)		
F24h	TXB0EIDL(5)	EF3h	RXF4EIDL(5)	EC3h	B4EIDH(5)	E93h	B1EIDH(5)	E63h	RXF12EIDL(5)		
F23h	TXB0EIDH(5)	EF2h	RXF4EIDH(5)	EC2h	B4SIDL(5)	E92h	B1SIDL(5)	E62h	RXF12EIDH(5)		
F22h	TXB0SIDL(5)	EF1h	RXF4SIDL(5)	EC1h	B4SIDH(5)	E91h	B1SIDH(5)	E61h	RXF12SIDL(5)		
F21h	TXB0SIDH(5)	EF0h	RXF4SIDH(5)	EC0h	B4CON(5)	E90h	B1CON(5)	E60h	RXF12SIDH(5)		
F20h	TXB0CON(5)	EEFh	RXF3EIDL(5)	EBFh	CANCON_RO6(5)	E90h	B1CON(5)	E5Fh	RXF11EIDL(5)		
F1Fh	CANCON_RO2(5)	EEEh	RXF3EIDH(5)	EBEh	CANSTAT_RO6(5)	E8Fh	CANCON_RO9(5)	E5Eh	RXF11EIDH(5)		
F1Eh	CANSTAT_RO2(5)	EEDh	RXF3SIDL(5)	EBDh	B3D7(5)	E8Eh	CANSTAT_RO9(5)	E5Dh	RXF11SIDL(5)		
F1Dh	TXB1D7(5)	EECh	RXF3SIDH(5)	EBCh	B3D6(5)	E8Dh	B0D7(5)	E5Ch	RXF11SIDH(5)		
F1Ch	TXB1D6(5)	EEBh	RXF2EIDL(5)	EBBh	B3D5(5)	E8Ch	B0D6(5)	E5Bh	RXF10EIDL(5)		
F1Bh	TXB1D5(5)	EEAh	RXF2EIDH(5)	EBAh	B3D4(5)	E8Bh	B0D5(5)	E5Ah	RXF10EIDH(5)		
F1Ah	TXB1D4(5)	EE9h	RXF2SIDL(5)	EB9h	B3D3(5)	E8Ah	B0D4(5)	E59h	RXF10SIDL(5)		
F19h	TXB1D3(5)	EE8h	RXF2SIDH(5)	EB8h	B3D2(5)	E89h	B0D3(5)	E58h	RXF10SIDH(5)		
F18h	TXB1D2(5)	EE7h	RXF1EIDL(5)	EB7h	B3D1(5)	E88h	B0D2(5)	E57h	RXF9EIDL(5)		
F17h	TXB1D1(5)	EE6h	RXF1EIDH(5)	EB6h	B3D0(5)	E87h	B0D1(5)	E56h	RXF9EIDH(5)		
F16h	TXB1D0(5)	EE5h	RXF1SIDL(5)	EB5h	B3DLC(5)	E86h	B0D0(5)	E55h	RXF9SIDL(5)		
F15h	TXB1DLC(5)	EE4h	RXF1SIDH(5)	EB4h	B3EIDL(5)	E85h	B0DLC(5)	E54h	RXF9SIDH(5)		
F14h	TXB1EIDL(5)	EE3h	RXF0EIDL(5)	EB3h	B3EIDH(5)	E84h	B0EIDL(5)	E53h	RXF8EIDL(5)		
F13h	TXB1EIDH(5)	EE2h	RXF0EIDH(5)	EB2h	B3SIDL(5)	E83h	B0EIDH(5)	E52h	RXF8EIDH(5)		
F12h	TXB1SIDL(5)	EE1h	RXF0SIDL(5)	EB1h	B3SIDH(5)	E82h	B0SIDL(5)	E51h	RXF8SIDL(5)		
F11h	TXB1SIDH(5)	EE0h	RXF0SIDH(5)	EB0h	B3CON(5)	E81h	B0SIDH(5)	E50h	RXF8SIDH(5)		
F10h	TXB1CON(5)					E80h	B0CON(5)				

注 **1：** 这不是实际存在的寄存器。
2： 未实现的寄存器，读为 0。
3： 该寄存器仅在 64 引脚器件上可用。
4： 该寄存器在 28 引脚器件上不可用。
5： SFR 还使用 E41h 至 F5Fh 的地址，但它们不是快速操作 RAM 的一部分。要访问这些寄存器，用户必须总是装入正确的 BSR 值。

表 6-2： PIC18F66K80 系列的寄存器文件汇总

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
FFFh	TOSU	—	—	—	栈顶最高字节 （TOS<20:16>）					90
FFEh	TOSH	栈顶高字节 （TOS<15:8>）								90
FFDh	TOSL	栈顶低字节 （TOS<7:0>）								90
FFCh	STKPTR	STKFUL	STKUNF	—	SP4	SP3	SP2	SP1	SP0	90
FFBh	PCLATU	—	—	Bit 21	PC<20:16> 的保持寄存器					90
FFAh	PCLATH	PC<15:8> 的保持寄存器								90
FF9h	PCL	PC 低字节 （PC<7:0>）								90
FF8h	TBLPTRU	—	—	Bit 21	程序存储器表指针最高字节 （TBLPTR<20:16>）					90
FF7h	TBLPTRH	程序存储器表指针高字节 （TBLPTR<15:8>）								90
FF6h	TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								90
FF5h	TABLAT	程序存储器表锁存器								90
FF4h	PRODH	乘积寄存器的高字节								90
FF3h	PRODL	乘积寄存器的低字节								90
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	90
FF1h	INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	90
FF0h	INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	90
FEFh	INDF0	使用 FSR0 的内容寻址数据存储器 ——FSR0 的值不变 （不是实际存在的寄存器）								90
FEEh	POSTINC0	使用 FSR0 的内容寻址数据存储器 ——FSR0 的值后递增 （不是实际存在的寄存器）								90
FEDh	POSTDEC0	使用 FSR0 的内容寻址数据存储器 ——FSR0 的值后递减 （不是实际存在的寄存器）								90
FECh	PREINC0	使用 FSR0 的内容寻址数据存储器 ——FSR0 的值预递增 （不是实际存在的寄存器）								90
FEBh	PLUSW0	使用 FSR0 的内容寻址数据存储器 ——FSR0 的值预递增 （不是实际存在的寄存器）， FSR0 偏移量的值由 W 寄存器提供								90
FEAh	FSR0H	—	—	—	—	间接数据存储器地址指针 0 的高字节				90
FE9h	FSR0L	间接数据存储器地址指针 0 的低字节								90
FE8h	WREG	工作寄存器								90
FE7h	INDF1	使用 FSR1 的内容寻址数据存储器 ——FSR1 的值不变 （不是实际存在的寄存器）								90
FE6h	POSTINC1	使用 FSR1 的内容寻址数据存储器 ——FSR1 的值后递增 （不是实际存在的寄存器）								90
FE5h	POSTDEC1	使用 FSR1 的内容寻址数据存储器 ——FSR1 的值后递减 （不是实际存在的寄存器）								90
FE4h	PREINC1	使用 FSR1 的内容寻址数据存储器 ——FSR1 的值预递增 （不是实际存在的寄存器）								90
FE3h	PLUSW1	使用 FSR1 的内容寻址数据存储器 ——FSR1 的值预递增 （不是实际存在的寄存器）， FSR1 偏移量的值由 W 寄存器提供								90
FE2h	FSR1H	—	—	—	—	间接数据存储器地址指针 1 的高字节				90
FE1h	FSR1L	间接数据存储器地址指针 1 的低字节								90
FE0h	BSR	—	—	—	—	存储区选择寄存器				90
FDFh	INDF2	使用 FSR2 的内容寻址数据存储器 ——FSR2 的值不变 （不是实际存在的寄存器）								90
FDEh	POSTINC2	使用 FSR2 的内容寻址数据存储器 ——FSR2 的值后递增 （不是实际存在的寄存器）								91
FDDh	POSTDEC2	使用 FSR2 的内容寻址数据存储器 ——FSR2 的值后递减 （不是实际存在的寄存器）								91
FDCh	PREINC2	使用 FSR2 的内容寻址数据存储器 ——FSR2 的值预递增 （不是实际存在的寄存器）								91
FDBh	PLUSW2	使用 FSR2 的内容寻址数据存储器 ——FSR2 的值预递增 （不是实际存在的寄存器）， FSR2 偏移量的值由 W 寄存器提供								91
FDAh	FSR2H	—	—	—	—	间接数据存储器地址指针 2 的高字节				91
FD9h	FSR2L	间接数据存储器地址指针 2 的低字节								91
FD8h	STATUS	—	—	—	N	OV	Z	DC	C	91
FD7h	TMR0H	Timer0 寄存器的高字节								91
FD6h	TMR0L	Timer0 寄存器的低字节								91
FD5h	T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	91
FD4h	未实现									—
FD3h	OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	HFIOFS	SCS1	SCS0	91
FD2h	OSCCON2	—	SOSCRUN	—	SOSCDRV	SOSCGO	—	MFIOFS	MFIOSEL	91
FD1h	WDTCON	REGSLP	—	ULPLVL	SRETEN	—	ULPEN	ULPSINK	SWDTEN	91
FD0h	RCON	IPEN	SBOREN	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	91

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
FCFh	TMR1H	Timer1 寄存器的高字节								91
FCEh	TMR1L	Timer1 寄存器的低字节								91
FCDh	T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON	91
FCCh	TMR2	Timer2 寄存器								91
FCBh	PR2	Timer2 周期寄存器								91
FCAh	T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	91
FC9h	SSPBUF	MSSP 接收缓冲 / 发送寄存器								91
FC8h	SSPADD	MSSP 地址寄存器 （I²C™ 从模式）， MSSP 波特率重载寄存器 （I²C 主模式）								91
FC8h	SSPMSK	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	91
FC7h	SSPSTAT	SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF	91
FC6h	SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	91
FC5h	SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	91
FC4h	ADRESH	A/D 结果寄存器的高字节								91
FC3h	ADRESL	A/D 结果寄存器的低字节								91
FC2h	ADCON0	—	CHS4	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON	91
FC1h	ADCON1	TRIGSEL1	TRIGSEL0	VCFG1	VCFG0	VNCFG	CHSN2	CHSN1	CHSN0	91
FC0h	ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	91
FBFh	ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0	91
FBEh	ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0	91
FBDh	CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								91
FBCh	CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								91
FBBh	CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	91
FBAh	TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	92
FB9h	BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	92
FB8h	IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP	92
FB7h	PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF	92
FB6h	PIE4	TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE	92
FB5h	CVRCON	CVREN	CVROE	CVRSS	CVR4	CVR3	CVR2	CVR1	CVR0	92
FB4h	CMSTAT	CMP2OUT	CMP1OUT	—	—	—	—	—	—	92
FB3h	TMR3H	Timer3 寄存器的高字节								92
FB2h	TMR3L	Timer3 寄存器的低字节								92
FB1h	T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON	92
FB0h	T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/ $\overline{\text{T3DONE}}$	T3GVAL	T3GSS1	T3GSS0	92
FAFh	SPBRG1	EUSART1 波特率发生器寄存器的低字节								92
FAEh	RCREG1	EUSART1 接收寄存器								92
FADh	TXREG1	EUSART1 发送寄存器								92
FACh	TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	92
FABh	RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	92
FAAh	T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/ $\overline{\text{T1DONE}}$	T1GVAL	T1GSS1	T1GSS0	92
FA9h	PR4	Timer4 周期寄存器								92
FA8h	HLVDCON	VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	92
FA7h	BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	92
FA6h	RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	92
FA5h	IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—	92
FA4h	PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—	92
FA3h	PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—	92
FA2h	IPR2	OSCFIP	—	—	—	BCLIP	HLVDIP	TMR3IP	TMR3GIP	92
FA1h	PIR2	OSCFIF	—	—	—	BCLIF	HLVDIF	TMR3IF	TMR3GIF	92
FA0h	PIE2	OSCFIE	—	—	—	BCLIE	HLVDIE	TMR3IE	TMR3GIE	92

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
F9Fh	IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP	92
F9Eh	PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF	92
F9Dh	PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE	92
F9Ch	PSTR1CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA	92
F9Bh	OSCTUNE	INTSRC	PLLEN	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	93
F9Ah	REFOCON	ROON	—	ROSSLP	ROSEL	RODIV3	RODIV2	RODIV1	RODIV0	93
F99h	CCPTMRS	—	—	—	C5TSEL	C4TSEL	C3TSEL	C2TSEL	C1TSEL	93
F98h	TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	93
F97h	TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	93
F96h	TRISE	TRISE7	TRISE6	TRISE5	TRISE4	—	TRISE2	TRISE1	TRISE0	93
F95h	TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	93
F94h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	93
F93h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	93
F92h	TRISA	TRISA7	TRISA6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0	93
F91h	ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD	93
F90h	SLRCON	—	SLRG	SLRF	SLRE	SLRD	SLRC	SLRB	SLRA	93
F8Fh	LATG	—	—	—	LATG4	LATG3	LATG2	LATG1	LATG0	93
F8Eh	LATF	LATF7	LATF6	LATF5	LATF4	—	LATF2	LATF1	LATF0	93
F8Dh	LATE	LATE7	LATE6	LATE5	LATE4	—	LATE2	LATE1	LATE0	93
F8Ch	LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	93
F8Bh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	93
F8Ah	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	93
F89h	LATA	LATA7	LATA6	LATA5	—	LATA3	LATA2	LATA1	LATA0	93
F88h	T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	93
F87h	TMR4	Timer4 寄存器								93
F86h	PORTG	—	—	—	RG4	RG3	RG2	RG1	RG0	93
F85h	PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0	93
F84h	PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	93
F83h	PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	93
F82h	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	93
F81h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	93
F80h	PORTA	RA7	RA6	RA5	—	RA3	RA2	RA1	RA0	93
F7Fh	EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	93
F7Eh	EECON2	闪存自编程控制寄存器 （不是实际存在的寄存器）								93
F7Dh	SPBRGH1	EUSART1 波特率发生器寄存器的高字节								93
F7Ch	SPBRGH2	EUSART2 波特率发生器寄存器的高字节								93
F7Bh	SPBRG2	EUSART2 波特率发生器寄存器的低字节								93
F7Ah	RCREG2	EUSART2 接收寄存器								93
F79h	TXREG2	EUSART2 发送寄存器								94
F78h	IPR5	IRXIP	WAKIP	ERRIP	TX2BIP	TXB1IP	TXB0IP	RXB1IP	RXB0IP	94
F77h	PIR5	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF	TXB0IF	RXB1IF	RXB0IF	94
F76H	PIE5	IRXIE	WAKIE	ERRIE	TX2BIE	TXB1IE	TXB0IE	RXB1IE	RXB0IE	94
F75h	EEADRH	数据 EE 地址寄存器的高字节								94
F74h	EEADR	数据 EE 地址寄存器的低字节								94
F73h	EEDATA	数据 EE 数据寄存器								94
F72h	ECANCON	MDSEL1	MDSEL0	FIFOWM	EWIN4	EWIN3	EWIN2	EWIN1	EWIN0	94
F71h	COMSTAT	RXB0OVFL	RXB1OVFL	TXBO	TXBP	RXBP	TXWARN	RXWARN	EWARN	94
F70h	CIOCON	TX2SRC	TX2EN	ENDRHI	CANCAP	—	—	—	CLKSEL	94
F6Fh	CANCON	REQOP2	REQOP1	REQOP0	ABAT	WIN2/FP3	WIN1/FP2	WIN0/FP1	FP0	94
F6Eh	CANSTAT	OPMODE2	OPMODE1	OPMODE0	—/ EICOD4	ICODE2/ EICODE3	ICODE1/ EICODE2	ICODE0/ EICODE1	—/ EICODE0	94

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
F6Dh	RXB0D7	RXB0D77	RXB0D76	RXB0D75	RXB0D74	RXB0D73	RXB0D72	RXB0D71	RXB0D70	94
F6Ch	RXB0D6	RXB0D67	RXB0D66	RXB0D65	RXB0D64	RXB0D63	RXB0D62	RXB0D61	RXB0D60	94
F6Bh	RXB0D5	RXB0D57	RXB0D56	RXB0D55	RXB0D54	RXB0D53	RXB0D52	RXB0D51	RXB0D50	94
F6Ah	RXB0D4	RXB0D47	RXB0D46	RXB0D45	RXB0D44	RXB0D43	RXB0D42	RXB0D41	RXB0D40	94
F69h	RXB0D3	RXB0D37	RXB0D36	RXB0D35	RXB0D34	RXB0D33	RXB0D32	RXB0D31	RXB0D30	94
F68h	RXB0D2	RXB0D27	RXB0D26	RXB0D25	RXB0D24	RXB0D23	RXB0D22	RXB0D21	RXB0D20	94
F67h	RXB0D1	RXB0D17	RXB0D16	RXB0D15	RXB0D14	RXB0D13	RXB0D12	RXB0D11	RXB0D10	94
F66h	RXB0D0	RXB0D07	RXB0D06	RXB0D05	RXB0D04	RXB0D03	RXB0D02	RXB0D01	RXB0D00	94
F65h	RXB0DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	94
F64h	RXB0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	94
F63h	RXB0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	94
F62h	RXB0SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	94
F61h	RXB0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	94
F60h	RXB0CON	RXFUL	RXM1	RXM0	—	RXRTRRO	RXB0DBEN	JTOFF	FILHIT0	94
F60h	RXB0CON	RXFUL	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	94
F5Fh	CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	94
F5Eh	CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	94
F5Dh	ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0	94
F5Ch	ANCON1	—	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8	94
F5Bh	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	94
F5Ah	IOCB	IOCB7	IOCB6	IOCB5	IOCB4	—	—	—	—	94
F59h	PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD	94
F58h	PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	95
F57h	PMD2	—	—	—	—	MODMD	ECANMD	CMP2MD	CMP1MD	95
F56h	PADCFG1	RDPU	REPU	RFPU	RGPU	—	—	—	CTMUDS	95
F55h	CTMUCONH	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG	95
F54h	CTMUCONL	EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT	95
F53h	CTMUICON	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	95
F52h	CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								95
F51h	CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								95
F50h	CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	95
F4Fh	CCPR3H	捕捉 / 比较 /PWM 寄存器 3 的高字节								95
F4Eh	CCPR3L	捕捉 / 比较 /PWM 寄存器 3 的低字节								95
F4Dh	CCP3CON	—	—	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0	95
F4Ch	CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节								95
F4Bh	CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节								95
F4Ah	CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	95
F49H	CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节								95
F48h	CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节								95
F47h	CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	95
F46h	PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	95
F45h	MDCON	MDEN	MDOE	MDSLR	MDOPOL	MDO	—	—	MDBIT	95
F44h	MDSRC	MDSODIS	—	—	—	MDSRC3	MDSRC2	MDSRC1	MDSRC0	95
F43h	MDCARH	MDCHODIS	MDCHPOL	MDCHSYNC	—	MDCH3	MDCH2	MDCH1	MDCH0	95
F42h	MDCARL	MDCLODIS	MDCLPOL	MDCLSYNC	—	MDCL3	MDCL2	MDCL1	MDCL0	95
F41h	未实现									—
F40h	未实现									—
F3Fh	CANCON_RO0	CANCON_RO0								95
F3Eh	CANSTAT_RO0	CANSTAT_RO0								95
F3Dh	RXB1D7	RXB1D77	RXB1D76	RXB1D75	RXB1D74	RXB1D73	RXB1D72	RXB1D71	RXB1D70	95
F3Ch	RXB1D6	RXB1D67	RXB1D66	RXB1D65	RXB1D64	RXB1D63	RXB1D62	RXB1D61	RXB1D60	95

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
F3Bh	RXB1D5	RXB1D57	RXB1D56	RXB1D55	RXB1D54	RXB1D53	RXB1D52	RXB1D51	RXB1D50	95
F3Ah	RXB1D4	RXB1D47	RXB1D46	RXB1D45	RXB1D44	RXB1D43	RXB1D42	RXB1D41	RXB1D40	95
F39h	RXB1D3	RXB1D37	RXB1D36	RXB1D35	RXB1D34	RXB1D33	RXB1D32	RXB1D31	RXB1D30	95
F38h	RXB1D2	RXB1D27	RXB1D26	RXB1D25	RXB1D24	RXB1D23	RXB1D22	RXB1D21	RXB1D20	95
F37h	RXB1D1	RXB1D17	RXB1D16	RXB1D15	RXB1D14	RXB1D13	RXB1D12	RXB1D11	RXB1D10	95
F36h	RXB1D0	RXB1D07	RXB1D06	RXB1D05	RXB1D04	RXB1D03	RXB1D02	RXB1D01	RXB1D00	95
F35h	RXB1DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	95
F34h	RXB1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	96
F33h	RXB1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	96
F32h	RXB1SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	96
F31h	RXB1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	96
F30h	RXB1CON	RXFUL	RXM1	RXM0	—	RXRTRRO	RXBODBEN	JTOFF	FILHIT0	96
F30h	RXB1CON	RXFUL	RXM1	RTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	96
F2Fh	CANCON_RO1	CANCON_RO1								96
F2Eh	CANSTAT_RO1	CANSTAT_RO1								96
F2Dh	TXB0D7	TXB0D77	TXB0D76	TXB0D75	TXB0D74	TXB0D73	TXB0D72	TXB0D71	TXB0D70	96
F2Ch	TXB0D6	TXB0D67	TXB0D66	TXB0D65	TXB0D64	TXB0D63	TXB0D62	TXB0D61	TXB0D60	96
F2Bh	TXB0D5	TXB0D57	TXB0D56	TXB0D55	TXB0D54	TXB0D53	TXB0D52	TXB0D51	TXB0D50	96
F2Ah	TXB0D4	TXB0D47	TXB0D46	TXB0D45	TXB0D44	TXB0D43	TXB0D42	TXB0D41	TXB0D40	96
F29h	TXB0D3	TXB0D37	TXB0D36	TXB0D35	TXB0D34	TXB0D33	TXB0D32	TXB0D31	TXB0D30	96
F28h	TXB0D2	TXB0D27	TXB0D26	TXB0D25	TXB0D24	TXB0D23	TXB0D22	TXB0D21	TXB0D20	96
F27h	TXB0D1	TXB0D17	TXB0D16	TXB0D15	TXB0D14	TXB0D13	TXB0D12	TXB0D11	TXB0D10	96
F26h	TXB0D0	TXB0D07	TXB0D06	TXB0D05	TXB0D04	TXB0D03	TXB0D02	TXB0D01	TXB0D00	96
F25h	TXB0DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	96
F24h	TXB0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	96
F23h	TXB0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	96
F22h	TXB0SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	96
F21h	TXB0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	96
F20h	TXB0CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	96
F1Fh	CANCON_RO2	CANCON_RO2								96
F1Eh	CANSTAT_RO2	CANSTAT_RO2								96
F1Dh	TXB1D7	TXB1D77	TXB1D76	TXB1D75	TXB1D74	TXB1D73	TXB1D72	TXB1D71	TXB1D70	96
F1Ch	TXB1D6	TXB1D67	TXB1D66	TXB1D65	TXB1D64	TXB1D63	TXB1D62	TXB1D61	TXB1D60	96
F1Bh	TXB1D5	TXB1D57	TXB1D56	TXB1D55	TXB1D54	TXB1D53	TXB1D52	TXB1D51	TXB1D50	96
F1Ah	TXB1D4	TXB1D47	TXB1D46	TXB1D45	TXB1D44	TXB1D43	TXB1D42	TXB1D41	TXB1D40	96
F19h	TXB1D3	TXB1D37	TXB1D36	TXB1D35	TXB1D34	TXB1D33	TXB1D32	TXB1D31	TXB1D30	96
F18h	TXB1D2	TXB1D27	TXB1D26	TXB1D25	TXB1D24	TXB1D23	TXB1D22	TXB1D21	TXB1D20	96
F17h	TXB1D1	TXB1D17	TXB1D16	TXB1D15	TXB1D14	TXB1D13	TXB1D12	TXB1D11	TXB1D10	96
F16h	TXB1D0	TXB1D07	TXB1D06	TXB1D05	TXB1D04	TXB1D03	TXB1D02	TXB1D01	TXB1D00	96
F15h	TXB1DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	96
F14h	TXB1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	96
F13h	TXB1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	96
F12h	TXB1SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	96
F11h	TXB1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	96
F10h	TXB1CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	96
F0Fh	CANCON_RO3	CANCON_RO3								96
F0Eh	CANSTAT_RO3	CANSTAT_RO3								96
F0Dh	TXB2D7	TXB2D77	TXB2D76	TXB2D75	TXB2D74	TXB2D73	TXB2D72	TXB2D71	TXB2D70	96
F0Ch	TXB2D6	TXB2D67	TXB2D66	TXB2D65	TXB2D64	TXB2D63	TXB2D62	TXB2D61	TXB2D60	97
F0Bh	TXB2D5	TXB2D57	TXB2D56	TXB2D55	TXB2D54	TXB2D53	TXB2D52	TXB2D51	TXB2D50	97
F0Ah	TXB2D4	TXB2D47	TXB2D46	TXB2D45	TXB2D44	TXB2D43	TXB2D42	TXB2D41	TXB2D40	—

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
F09h	TXB2D3	TXB2D37	TXB2D36	TXB2D35	TXB2D34	TXB2D33	TXB2D32	TXB2D31	TXB2D30	97
F08h	TXB2D2	TXB2D27	TXB2D26	TXB2D25	TXB2D24	TXB2D23	TXB2D22	TXB2D21	TXB2D20	97
F07h	TXB2D1	TXB2D17	TXB2D16	TXB2D15	TXB2D14	TXB2D13	TXB2D12	TXB2D11	TXB2D10	97
F06h	TXB2D0	TXB2D07	TXB2D06	TXB2D05	TXB2D04	TXB2D03	TXB2D02	TXB2D01	TXB2D00	97
F05h	TXB2DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	97
F04h	TXB2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	97
F03h	TXB2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	97
F02h	TXB2SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	97
F01h	TXB2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	97
F00h	TXB2CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	97
EFFh	RXM1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	97
EFEh	RXM1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	97
EFDh	RXM1SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	97
EFCh	RXM1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	97
EFBh	RXM0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	97
EFAh	RXM0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	97
EF9h	RXM0SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	97
EF8h	RXM0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	97
EF7h	RXF5EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	97
EF6h	RXF5EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	97
EF5h	RXF5SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	97
EF4h	RXF5SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	97
EF3h	RXF4EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	97
EF2h	RXF4EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	97
EF1h	RXF4SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	97
EF0h	RXF4SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	98
EEFh	RXF3EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	98
EEEh	RXF3EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	98
EEDh	RXF3SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	98
EECh	RXF3SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	98
EEBh	RXF2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	98
EEAh	RXF2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	98
EE9h	RXF2SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	98
EE8h	RXF2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	98
EE7h	RXF1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	98
EE6h	RXF1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	98
EE5h	RXF1SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	98
EE4h	RXF1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	98
EE3h	RXF0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	98
EE2h	RXF0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	98
EE1h	RXF0SIDL	SID2	SID1	SID0	—	EXIDEN	—	EID17	EID16	98
EE0h	RXF0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	98
EDFh	CANCON_RO4	CANCON_RO4								98
EDEh	CANSTAT_RO4	CANSTAT_RO4								98
EDDh	B5D7	B5D77	B5D76	B5D75	B5D74	B5D73	B5D72	B5D71	B5D70	98
EDCh	B5D6	B5D67	B5D66	B5D65	B5D64	B5D63	B5D62	B5D61	B5D60	98
EDBh	B5D5	B5D57	B5D56	B5D55	B5D54	B5D53	B5D52	B5D51	B5D50	98
EDAh	B5D4	B5D47	B5D46	B5D45	B5D44	B5D43	B5D42	B5D41	B5D40	98
ED9h	B5D3	B5D37	B5D36	B5D35	B5D34	B5D33	B5D32	B5D31	B5D30	98
ED8h	B5D2	B5D27	B5D26	B5D25	B5D24	B5D23	B5D22	B5D21	B5D20	98
ED7h	B5D1	B5D17	B5D16	B5D15	B5D14	B5D13	B5D12	B5D11	B5D10	98

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
ED6h	B5D0	B5D07	B5D06	B5D05	B5D04	B5D03	B5D02	B5D01	B5D00	98
ED5h	B5DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	98
ED4h	B5EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	98
ED3h	B5EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	98
ED2h	B5SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	98
ED1h	B5SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	98
ED0h	B5CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	98
ECFh	CANCON_RO5	CANCON_RO5								98
ECEh	CANSTAT_RO5	CANSTAT_RO5								99
ECDh	B4D7	B4D77	B4D76	B4D75	B4D74	B4D73	B4D72	B4D71	B4D70	99
ECCh	B4D6	B4D67	B4D66	B4D65	B4D64	B4D63	B4D62	B4D61	B4D60	99
ECBh	B4D5	B4D57	B4D56	B4D55	B4D54	B4D53	B4D52	B4D51	B4D50	99
ECAh	B4D4	B4D47	B4D46	B4D45	B4D44	B4D43	B4D42	B4D41	B4D40	99
EC9h	B4D3	B4D37	B4D36	B4D35	B4D34	B4D33	B4D32	B4D31	B4D30	99
EC8h	B4D2	B4D27	B4D26	B4D25	B4D24	B4D23	B4D22	B4D21	B4D20	99
EC7h	B4D1	B4D17	B4D16	B4D15	B4D14	B4D13	B4D12	B4D11	B4D10	99
EC6h	B4D0	B4D07	B4D06	B4D05	B4D04	B4D03	B4D02	B4D01	B4D00	99
EC5h	B4DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	99
EC4h	B4EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	99
EC3h	B4EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	99
EC2h	B4SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	99
EC1h	B4SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	99
EC0h	B4CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	99
EBFh	CANCON_RO6	CANCON_RO6								99
EBEh	CANSTAT_RO6	CANSTAT_RO6								99
EBDh	B3D7	B3D77	B3D76	B3D75	B3D74	B3D73	B3D72	B3D71	B3D70	99
EBCh	B3D6	B3D67	B3D66	B3D65	B3D64	B3D63	B3D62	B3D61	B3D60	99
EBBh	B3D5	B3D57	B3D56	B3D55	B3D54	B3D53	B3D52	B3D51	B3D50	99
EBAh	B3D4	B3D47	B3D46	B3D45	B3D44	B3D43	B3D42	B3D41	B3D40	99
EB9h	B3D3	B3D37	B3D36	B3D35	B3D34	B3D33	B3D32	B3D31	B3D30	99
EB8h	B3D2	B3D27	B3D26	B3D25	B3D24	B3D23	B3D22	B3D21	B3D20	99
EB7h	B3D1	B3D17	B3D16	B3D15	B3D14	B3D13	B3D12	B3D11	B3D10	99
EB6h	B3D0	B3D07	B3D06	B3D05	B3D04	B3D03	B3D02	B3D01	B3D00	99
EB5h	B3DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	99
EB4h	B3EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	99
EB3h	B3EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	99
EB2h	B3SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	99
EB1h	B3SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	99
EB0h	B3CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	99
EAFh	CANCON_RO7	CANCON_RO7								99
EAEh	CANSTAT_RO7	CANSTAT_RO7								99
EADh	B2D7	B2D77	B2D76	B2D75	B2D74	B2D73	B2D72	B2D71	B2D70	99
EACh	B2D6	B2D67	B2D66	B2D65	B2D64	B2D63	B2D62	B2D61	B2D60	99
EABh	B2D5	B2D57	B2D56	B2D55	B2D54	B2D53	B2D52	B2D51	B2D50	100
EAAh	B2D4	B2D47	B2D46	B2D45	B2D44	B2D43	B2D42	B2D41	B2D40	100
EA9h	B2D3	B2D37	B2D36	B2D35	B2D34	B2D33	B2D32	B2D31	B2D30	100
EA8h	B2D2	B2D27	B2D26	B2D25	B2D24	B2D23	B2D22	B2D21	B2D20	100
EA7h	B2D1	B2D17	B2D16	B2D15	B2D14	B2D13	B2D12	B2D11	B2D10	100
EA6h	B2D0	B2D07	B2D06	B2D05	B2D04	B2D03	B2D02	B2D01	B2D00	100
EA5h	B2DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	100
EA4h	B2EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	100

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
EA3h	B2EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	100
EA2h	B2SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	100
EA1h	B2SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	100
EA0h	B2CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	—	TXPRI1	TXPRI0	100
E9Fh	CANCON_RO8	CANCON_RO8								100
E9Eh	CANSTAT_RO8	CANSTAT_RO8								100
E9Dh	B1D7	B1D77	B1D76	B1D75	B1D74	B1D73	B1D72	B1D71	B1D70	100
E9Ch	B1D6	B1D67	B1D66	B1D65	B1D64	B1D63	B1D62	B1D61	B1D60	100
E9Bh	B1D5	B1D57	B1D56	B1D55	B1D54	B1D53	B1D52	B1D51	B1D50	100
E9Ah	B1D4	B1D47	B1D46	B1D45	B1D44	B1D43	B1D42	B1D41	B1D40	100
E99h	B1D3	B1D37	B1D36	B1D35	B1D34	B1D33	B1D32	B1D31	B1D30	100
E98h	B1D2	B1D27	B1D26	B1D25	B1D24	B1D23	B1D22	B1D21	B1D20	100
E97h	B1D1	B1D17	B1D16	B1D15	B1D14	B1D13	B1D12	B1D11	B1D10	100
E96h	B1D0	B1D07	B1D06	B1D05	B1D04	B1D03	B1D02	B1D01	B1D00	100
E95h	B1DLC	—	TXRTR	—	—	DLC3	DLC2	DLC1	DLC0	100
E94h	B1EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	100
E93h	B1EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	100
E92h	B1SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	100
E91h	B1SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	100
E90h	B1CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0	100
E90h	B1CON	RXFUL	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	100
E8Fh	CANCON_RO9	CANCON_RO9								100
E8Eh	CANSTAT_RO9	CANSTAT_RO9								100
E8Dh	B0D7	B0D77	B0D76	B0D75	B0D74	B0D73	B0D72	B0D71	B0D70	100
E8Ch	B0D6	B0D67	B0D66	B0D65	B0D64	B0D63	B0D62	B0D61	B0D60	100
E8Bh	B0D5	B0D57	B0D56	B0D55	B0D54	B0D53	B0D52	B0D51	B0D50	100
E8Ah	B0D4	B0D47	B0D46	B0D45	B0D44	B0D43	B0D42	B0D41	B0D40	100
E89h	B0D3	B0D37	B0D36	B0D35	B0D34	B0D33	B0D32	B0D31	B0D30	100
E88h	B0D2	B0D27	B0D26	B0D25	B0D24	B0D23	B0D22	B0D21	B0D20	101
E87h	B0D1	B0D17	B0D16	B0D15	B0D14	B0D13	B0D12	B0D11	B0D10	101
E86h	B0D0	B0D07	B0D06	B0D05	B0D04	B0D03	B0D02	B0D01	B0D00	101
E85h	B0DLC	—	RXRTR	RB1	RB0	DLC3	DLC2	DLC1	DLC0	101
E84h	B0EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	101
E83h	B0EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	101
E82h	B0SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	101
E81h	B0SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	101
E80h	B0CON	TXBIF	TXABT	TXLARB	TXERR	TXREQ	RTREN	TXPRI1	TXPRI0	101
E80h	B0CON	RTXFUL	RXM1	RXRTRRO	FILHIT4	FILHIT3	FILHIT2	FILHIT1	FILHIT0	101
E7Fh	TXBIE	—	—	—	CAN TX 缓冲区中断允许			—	—	101
E7Eh	BIE0	CAN 缓冲区中断允许								101
E7Dh	BSEL0	模式选择寄存器 0						—	—	101
E7Ch	MSEL3	CAN 屏蔽器选择寄存器 3								101
E7Bh	MSEL2	CAN 屏蔽器选择寄存器 2								101
E7Ah	MSEL1	CAN 屏蔽器选择寄存器 1								101
E79h	MSEL0	CAN 屏蔽器选择寄存器 0								101
E78h	RXFBCON7	CAN 缓冲区 15/14 指针寄存器								101
E77h	RXFBCON6	CAN 缓冲区 13/12 指针寄存器								101
E76h	RXFBCON5	CAN 缓冲区 11/10 指针寄存器								101
E75h	RXFBCON4	CAN 缓冲区 9/8 指针寄存器								101
E74h	RXFBCON3	CAN 缓冲区 7/6 指针寄存器								101
E73h	RXFBCON2	CAN 缓冲区 5/4 指针寄存器								101

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
E72h	RXFBCON1	CAN 缓冲区 3/2 指针寄存器								101

表 6-2： PIC18F66K80 系列的寄存器文件汇总 （续）

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值 所在页
E71h	RXFBCON0	CAN 缓冲区 1/0 指针寄存器								101
E70h	SDFLC	—	—	—	CAN DeviceNet 计数寄存器					101
E6Fh	RXF15EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	101
E6Eh	RXF15EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	101
E6Dh	RXF15SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	101
E6Ch	RXF15SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	101
E6Bh	RXF14EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	101
E6Ah	RXF14EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	101
E69h	RXF14SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	101
E68h	RXF14SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	101
E67h	RXF13EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	101
E66h	RXF13EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	102
E65h	RXF13SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	102
E64h	RXF13SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	102
E63h	RXF12EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	102
E62h	RXF12EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	102
E61h	RXF12SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	102
E60h	RXF12SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	102
E5Fh	RXF11EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	102
E5Eh	RXF11EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	102
E5Dh	RXF11SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	102
E5Ch	RXF11SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	102
E5Bh	RXF10EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	102
E5Ah	RXF10EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	102
E59h	RXF10SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	102
E58h	RXF10SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	102
E57h	RXF9EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	102
E56h	RXF9EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	102
E55h	RXF9SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	102
E54h	RXF9SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	102
E53h	RXF8EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	102
E52h	RXF8EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	102
E51h	RXF8SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	102
E50h	RXF8SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	102
E4Fh	RXF7EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	102
E4Eh	RXF7EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	102
E4Dh	RXF7SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	102
E4Ch	RXF7SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	102
E4Bh	RXF6EIDL	EID7	EID6	EID5	EID4	EID3	EID2	EID1	EID0	102
E4Ah	RXF6EIDH	EID15	EID14	EID13	EID12	EID11	EID10	EID9	EID8	102
E49h	RXF6SIDL	SID2	SID1	SID0	SRR	EXID	—	EID17	EID16	102
E48h	RXF6SIDH	SID10	SID9	SID8	SID7	SID6	SID5	SID4	SID3	102
E47h	RXFCON0	CAN 接收过滤器控制寄存器 0								102
E46h	RXFCON1	CAN 接收过滤器控制寄存器 1								102
E45h	BRGCON3	WAKDIS	WAKFIL	—	—	—	SEG2PH2	SEG2PH1	SEG2PH0	102
E44h	BRGCON2	SEG2PHTS	SAM	SEG1PH2	SEG1PH1	SEG1PH0	PRSEG2	PRSEG1	PRSEG0	103
E43h	BRGCON1	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	103
E42h	TXERRCNT	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	103
E41h	RXERRCNT	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	103

6.3.5 STATUS 寄存器

如寄存器 6-2 所示，STATUS 寄存器包含 ALU 的算术运算状态。和任何其他寄存器一样，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，则会禁止对这 5 位进行写操作。

这些位根据器件逻辑被置 1 或清零。因此，在执行一条将 STATUS 寄存器作为目标寄存器的指令后，运行结果可能与预想的不同。例如， CLRF STATUS 将 Z 位置 1 并保持其余位不变。这样，读取 STATUS 寄存器将得到“000u u1uu”。

因此，建议仅使用 BCF、BSF、SWAPF、MOVFF 和 MOVWF 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。

关于其他不会影响状态位的指令，请参见表 29-2 和表 29-3 中的指令集汇总。

注： 在减法运算中，C 和 DC 位分别作为借位位和半借位位。

寄存器 6-2： STATUS 寄存器

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC(1)	C(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 **N：** 负标志位
该位用于有符号的算术运算 （以二进制补码方式进行）。它可以表示结果是否为负 （ALU MSB = 1）。
1 = 结果为负
0 = 结果为正

bit 3 **OV：** 溢出标志位
该位用于有符号的算术运算 （以二进制补码方式进行）。它表明运算结果溢出了 7 位二进制数的范围，溢出导致符号位 （bit 7）发生改变。
1 = 有符号算术运算中发生溢出 （本次算术运算）
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 / 借位位 (1)
对于 ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令：
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 / 借位位 (2)
对于 ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令：
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。对于移位指令（RRF 和 RLF），此位来自源寄存器的 bit 4 或 bit 3。

2： 对于借位，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。对于移位指令（RRF 和 RLF），此位来自源寄存器的最高位或最低位。

6.4 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 6.6 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 6.6.1 节 “ 使用立即数偏移量进行变址寻址 ”**。

6.4.1 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。该模式的示例包括指令 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。该模式的示例包括指令 ADDLW 和 MOVLW，它们分别向 W 寄存器加或移入立即数值。其他立即数寻址指令，例如 CALL 和 GOTO，它们包括一个 20 位的程序存储器地址。

6.4.2 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 6.3.3 节 “ 通用寄存器文件 ”**）或快速操作存储区（**第 6.3.2 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 6.3.1 节 “ 存储区选择寄存器 ”**）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 12 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 MOVFF，在操作码中包含完整的 12 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。

6.4.3 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本身作为特殊功能寄存器位于 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（Indirect File Operand，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 6-5 所示的清零整个 RAM 存储区的操作。它还允许用户在数据存储器中执行变址寻址和其他针对程序存储器的堆栈指针操作。

例 6-5： 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
```

6.4.3.1 FSR 寄存器和 INDF 操作数

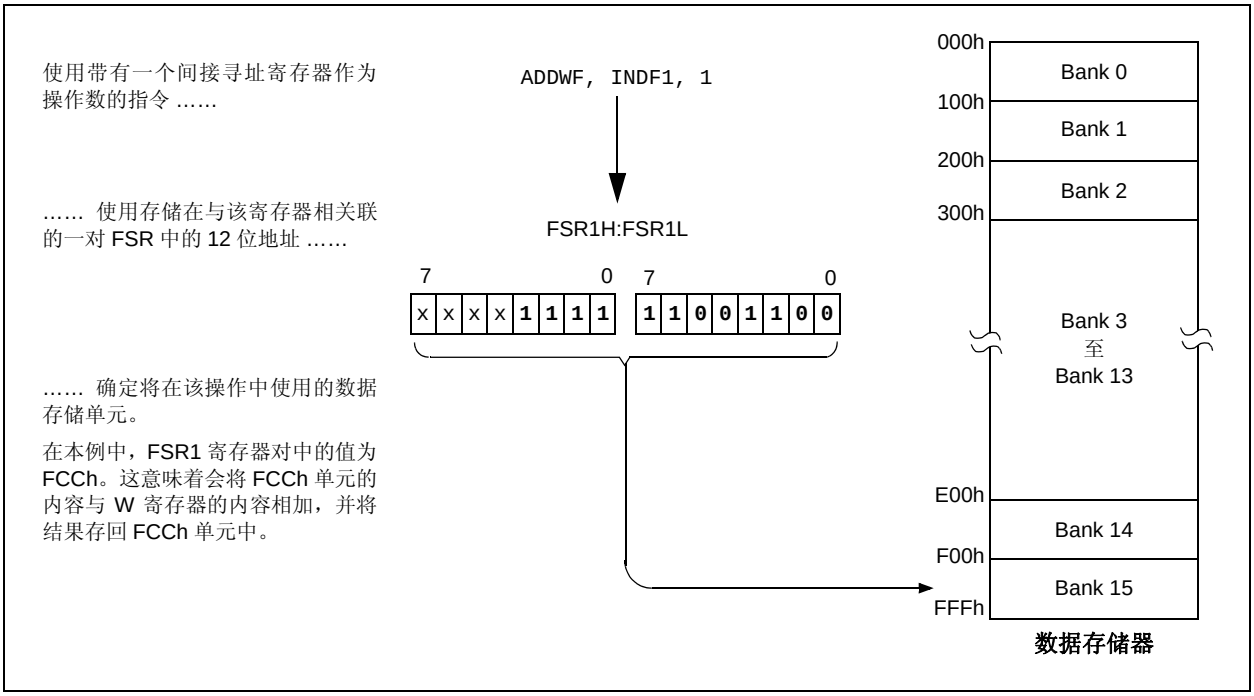
间接寻址的核心是三组寄存器：FSR0、FSR1 和 FSR2。每组寄存器都含有一对 8 位寄存器：FSRnH 和 FSRnL。FSRnH 寄存器的高 4 位未使用，因此每对 FSR 只保存一个 12 位值，从而可以线性寻址数据存储器的整个空间。因此，FSR寄存器对被用作数据存储器的地址指针。

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。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 12 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 位对于确定目标地址没有影响。

图 6-8：　　间接寻址



6.4.3.2 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。

这些操作数是：

- 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。

使用 POSTDEC、POSTINC 和 PREINC 对 FSR 进行操作会影响整对寄存器。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）。

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。

6.4.3.3 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 FE7h。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 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。

6.5 程序存储器和扩展指令集

程序存储器的操作不受扩展指令集的影响。

使能扩展指令集会将 5 条额外的双字命令添加到现有的 PIC18 指令集中：即 ADDFSR、CALLW、MOVSF、MOVSS 和 SUBFSR。这些指令如**第 6.2.4 节 “ 双字指令 ”** 中所述执行。

6.6 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。许多核心 PIC18 指令使用快速操作存储区，引入了对数据存储空间的新的寻址模式。该模式还会改变使用 FSR2 及其相关操作数进行间接寻址的方式。

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。

6.6.1 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寄存器对及其相关文件操作数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（即绝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：

- 使用快速操作存储区 （a = 0）
- 文件地址参数要小于或等于 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。

6.6.2 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，即核心 PIC18 指令集中几乎一半的指令。只有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使用快速操作存储区 （快速操作 RAM 位 = 1）或包含 60h 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 6-9 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。在**第 29.2.1 节 “ 扩展指令的语法 ”** 中对此进行了更详细的说明。

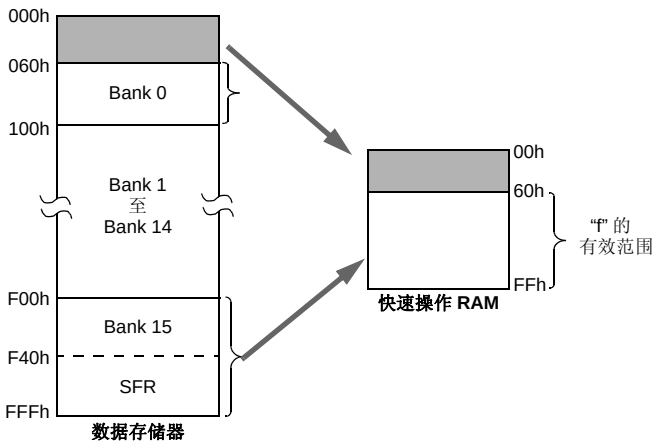
图 6-9：　　针对位和针对字节的指令的寻址模式对比 （使能了扩展指令集）

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 FFFh 之间的单元地址，该地址也是数据存储器的 F60h 至 FFFh （Bank 15）。

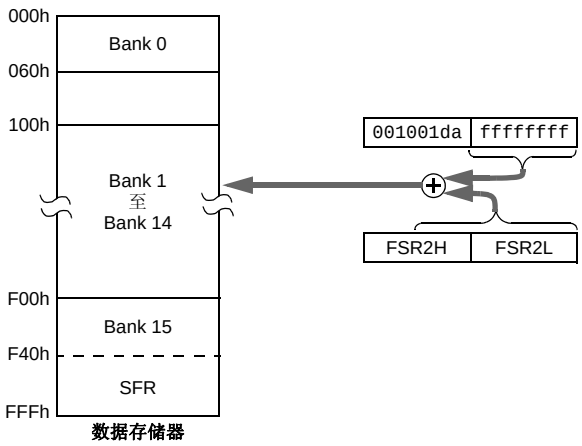
不可用此模式寻址地址低于 060h 的单元。



当 a = 0 且 f ≤ 5Fh 时：

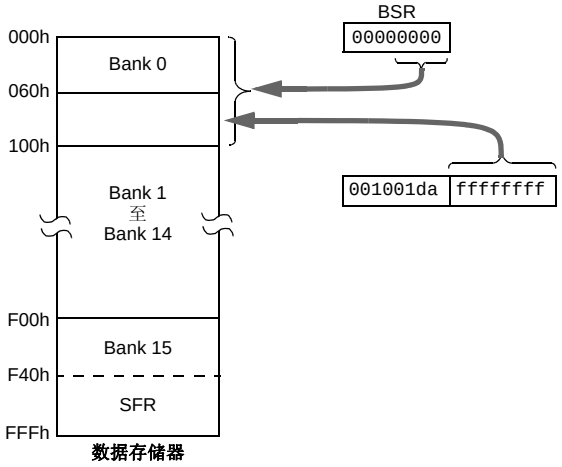
此指令以立即数变址寻址模式执行。 “f” 被解析为 FSR2 中地址值的偏移量。将这两个值相加可以得到指令的目标寄存器的地址。此地址可以在数据存储空间的任何地方。

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16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。



6.6.3 在立即数变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 低地址单元（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。

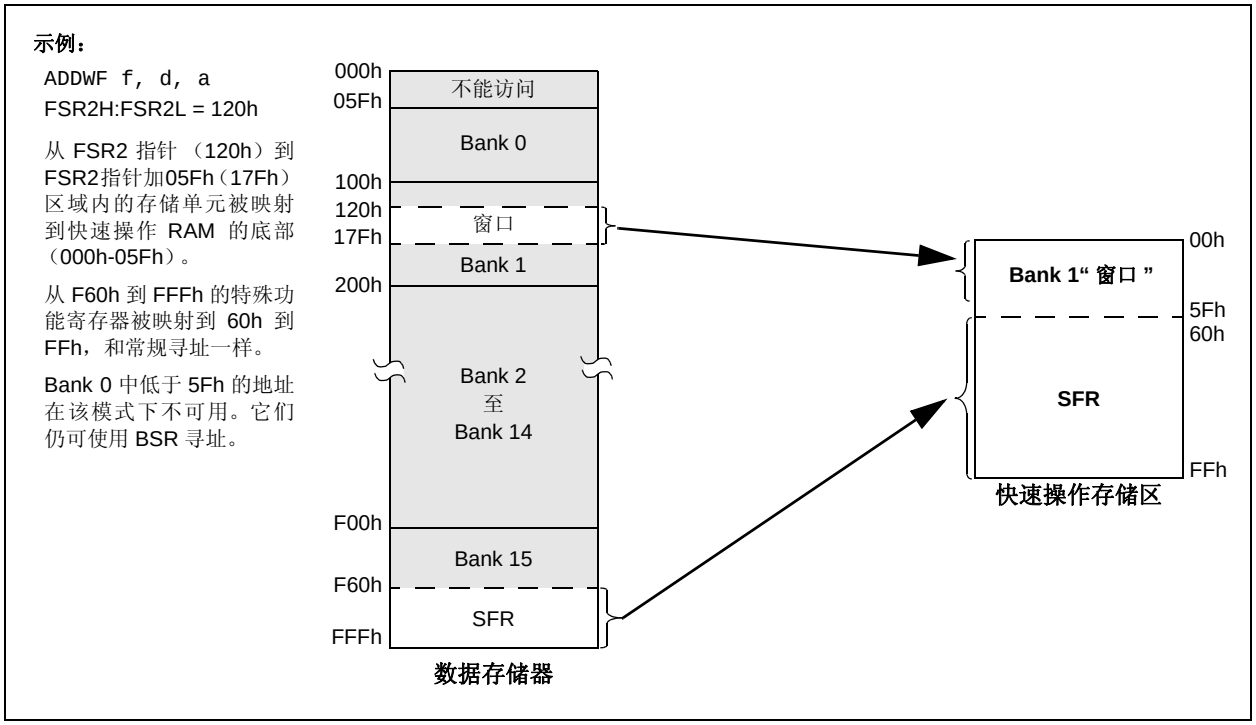
FSR2 的值定义映射到窗口的地址的下边界，而上边界则由 FSR2 加 95（5Fh）决定。地址大于 5Fh 的快速操作 RAM 的映射方法如前所述（见**第 6.3.2 节“快速操作存储区”**）。图 6-10 给出了在此寻址模式下重映射快速操作存储区的示例。

快速操作存储区的重映射仅适用于立即数变址寻址模式。使用 BSR（快速操作 RAM 位 = 1）的操作和以前一样继续使用直接寻址模式。任何明确使用间接文件操作数（包括 FSR2）的间接或变址寻址操作都将像标准间接寻址一样操作。任何使用快速操作存储区（另外包括大于 05Fh 的寄存器地址）的指令都将使用直接寻址和常规的快速操作存储区映射。

6.6.4 立即数变址模式中的 BSR

尽管使能扩展指令集时会重映射快速操作存储区，但 BSR 的操作保持不变。使用 BSR 选择数据存储区的直接寻址操作方式和以前描述的相同。

图 6-10：　使用立即数变址寻址模式重映射快速操作存储区



7.0 闪存程序存储器

在整个 VDD 范围内，正常操作期间，闪存程序存储器都是可读写、可擦除的。

读程序存储器时，每次读取一个字节。写程序存储器时，每次写入一个 64 字节的块。擦除程序存储器时，每次擦除一个 64 字节的块。用户代码不能执行批量擦除操作。

在擦写程序存储器时，系统会停止取指令直到操作完成。擦写期间不能访问程序存储器，因此也就无法执行代码。由内部编程定时器来终止程序存储器的擦写操作。

写入程序存储器的值不一定非要是有效指令。执行存储无效指令的程序存储单元会导致执行 NOP。

7.1 表读与表写

为了读写程序存储器，有两个操作可供处理器在程序存储空间和数据 RAM 之间传送字节：

- 表读 （TBLRD）
- 表写 （TBLWT）

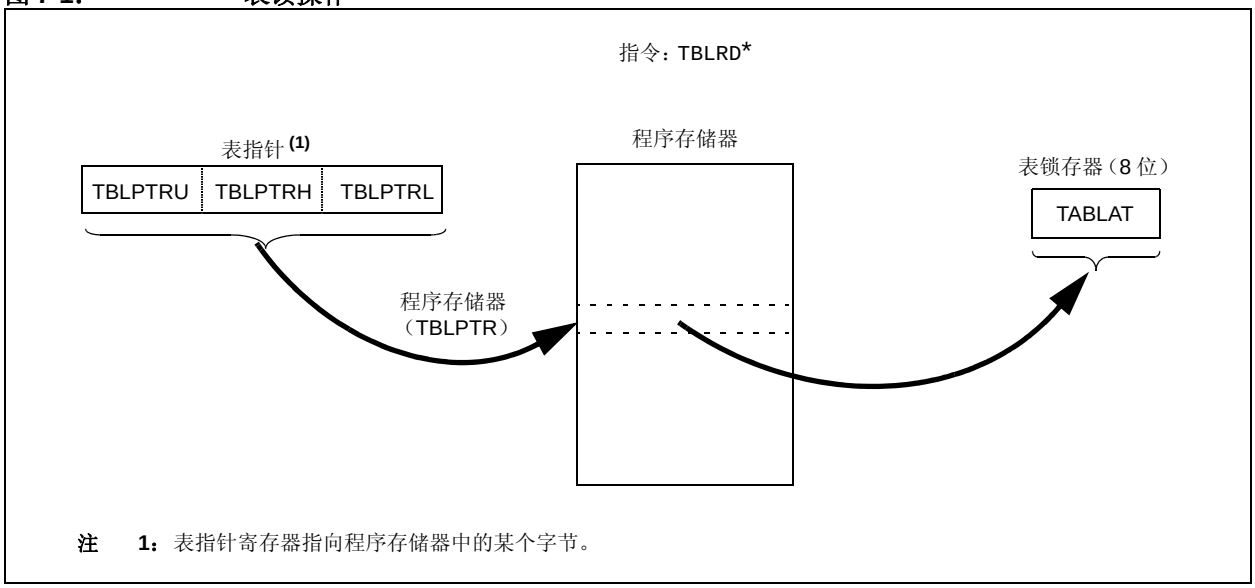
程序存储空间为 16 位宽，而数据 RAM 空间为 8 位宽。表读和表写操作通过一个 8 位寄存器 （TABLAT）在这两个存储空间之间传送数据。

表读操作从程序存储器获取数据并将其放入数据 RAM 空间。图 7-1 显示了程序存储器和数据 RAM 之间的一次表读操作。

表写操作将数据存储空间中的数据存储到程序存储器的保持寄存器中。**第 7.5 节 “ 写闪存程序存储器 ”** 详细介绍了将保持寄存器的内容写入程序存储器的过程。图 7-2 显示了程序存储器和数据 RAM 之间的一次表写操作。

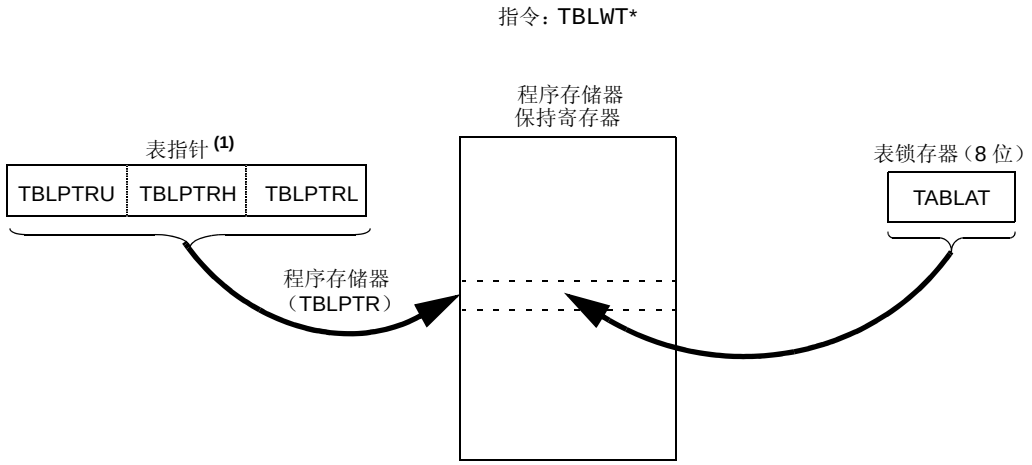
表操作以字节为单位。包含数据而非程序指令的表块不需要按字对齐。因此，表块可以在任何字节地址开始和结束。如果使用表写操作向程序存储器写入可执行代码，程序指令必须按字对齐。

图 7-1： 表读操作



注 1： 表指针寄存器指向程序存储器中的某个字节。

图 7-2：　　表写操作



注　1： 表指针实际指向 64 个保持寄存器之一，其地址由 TBLPTRL<5:0> 决定。**第 7.5 节 “ 写闪存程序存储器 ”** 中将讨论向程序存储器阵列写入数据的物理过程。

7.2　控制寄存器

TBLRD 和 TBLWT 指令要用到几个控制寄存器。这些寄存器包括：

- EECON1 寄存器
- EECON2 寄存器
- TABLAT 寄存器
- TBLPTR 寄存器

7.2.1　EECON1 和 EECON2 寄存器

EECON1 寄存器（寄存器 7-1）是存储器访问的控制寄存器。EECON2 寄存器不是实际存在的寄存器，专用于存储器的擦写操作。读 EECON2 将得到全 0。

EEPGD 控制位决定访问的是程序存储器还是数据 EEPROM 存储器。当 EEPGD 清零时，任何后续操作都将针对数据 EEPROM 存储器进行。当 EEPGD 置 1 时，任何后续操作都将针对程序存储器进行。

CFGS 控制位决定访问的是配置寄存器还是程序存储器 / 数据 EEPROM 存储器。当 CFGS 置 1 时，不管 EEPGD的值如何，后续操作将针对配置寄存器进行（见 **第 28.0 节 “CPU 的特殊功能 ”**）。当 CFGS 清零时，则由 EEPGD 来选择访问的存储器。

当 FREE 位置 1 时，允许对程序存储器进行擦除操作，擦除操作由下一条 WR 命令启动。当 FREE 清零时，则仅使能写操作。

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，在内部编程定时器超时、写操作结束时被清零。

注：	在正常操作期间，WRERR 读为 1。这表明写操作被复位提早终止或进行了不合法的写操作。

WR 控制位用于启动写操作。用软件只能将该位置 1 而无法清零。在写操作完成时，由硬件将其清零。

注：	当写操作完成时，EEIF 中断标志位（PIR4<6>）被置 1。它必须用软件清零。

寄存器 7-1：　　EECON1：EEPROM 控制寄存器 1

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR(1)	WREN	WR	RD
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　**EEPGD：**闪存程序存储器或数据 EEPROM 存储器选择位
1 = 访问闪存程序存储器
0 = 访问数据 EEPROM 存储器

bit 6　**CFGS：**闪存程序存储器 / 数据 EEPROM 存储器或配置寄存器选择位
1 = 访问配置寄存器
0 = 访问闪存程序存储器或数据 EEPROM 存储器

bit 5　**未实现：**读为 0

bit 4　**FREE：**闪存行擦除使能位
1 = 在下一条 WR 命令时擦除 TBLPTR 指定的程序存储器行 （擦除操作完成后清零）
0 = 仅执行写操作

bit 3　**WRERR：**闪存程序存储器 / 数据 EEPROM 存储器错误标志位 (1)
1 = 写操作提早终止（由于正常操作中自定时编程期间的任何复位，或不合法的写操作）
0 = 写操作完成

bit 2　**WREN：**闪存程序存储器 / 数据 EEPROM 存储器写使能位
1 = 允许对闪存程序存储器 / 数据 EEPROM 存储器的写周期
0 = 禁止对闪存程序存储器 / 数据 EEPROM 存储器的写周期

bit 1　**WR：**写控制位
1 = 启动数据 EEPROM 擦除 / 写周期或者程序存储器的擦除周期或写周期。
（操作是自定时的，一旦写操作完成，该位即由硬件清零。
用软件只能将 WR 位置 1，但不能清零。）
0 = EEPROM 写周期完成

bit 0　**RD：**读控制位
1 = 启动 EEPROM 读操作（读操作需要一个指令周期。RD 由硬件清零。用软件只能将 RD 位置 1，但不能清零。 EEPGD = 1 或 CFGS = 1 时， RD 位无法置 1。）
0 = 不启动 EEPROM 读操作

注　1： 当发生 WRERR 时， EEPGD 和 CFGS 位不会被清零。这样可以跟踪错误情况。

7.2.2 TABLAT—— 表锁存寄存器

表锁存器（TABLAT）是映射到 SFR 空间的一个 8 位寄存器。表锁存器用于在程序存储器和数据 RAM 之间传输数据时保存 8 位数据。

7.2.3 TBLPTR—— 表指针寄存器

表指针（TBLPTR）寄存器在程序存储器中以字节为单位进行寻址。TBLPTR 由 3 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 3 个寄存器合起来组成一个 22 位宽的指针。其中低 21 位允许器件寻址最大 2 MB 的程序存储空间。第 22 位则允许访问器件 ID、用户 ID 和配置位。

TBLRD 和 TBLWT 指令要使用表指针寄存器 TBLPTR。这些指令可以基于表操作以 4 种方法之一更新 TBLPTR。表 7-1 列出了这些操作，这些操作只会影响 TBLPTR 的低 21 位。

7.2.4 表指针边界

TBLPTR 用于读、写和擦除闪存程序存储器。

当执行 TBLRD 时，TBLPTR 的所有 22 位决定将程序存储器的哪个字节读入 TABLAT。

当执行 TBLWT 时，表指针寄存器的低 6 位（TBLPTR<5:0>）决定要写入程序存储器的哪个保持寄存器（共有 64 个）。当程序存储器的定时写入（通过 WR 位）开始时，TBLPTR 的高 16 位（TBLPTR<21:6>）将决定要写入哪个程序存储器块（每块 64 字节）。更多详细信息，请参见**第 7.5 节 “ 写闪存程序存储器 ”**。

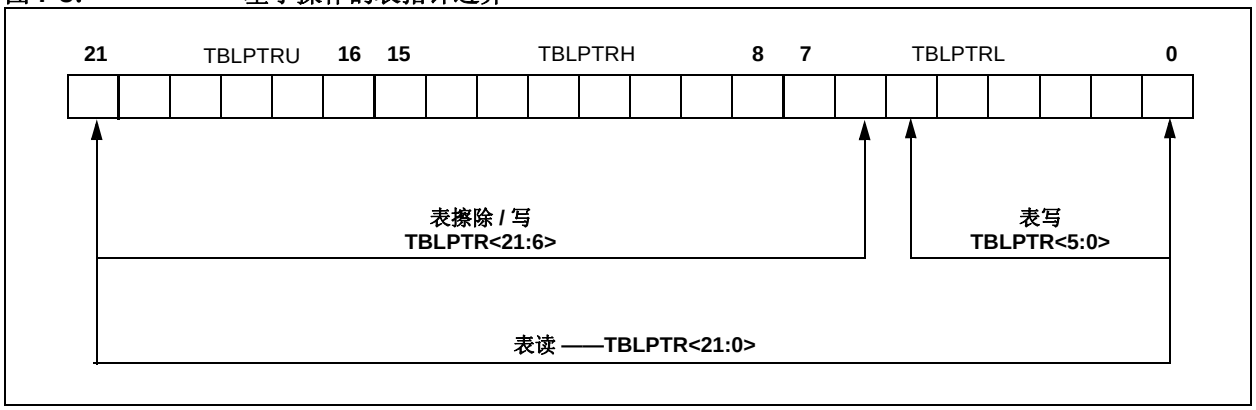
当执行擦除程序存储器时，表指针寄存器的高 16 位（TBLPTR<21:6>）指向将要擦除的 64 字节块。最低有效位（TBLPTR<5:0>）被忽略。

图 7-3 说明了基于闪存程序存储器操作的 TBLPTR 相关边界。

表 7-1： 执行 TBLRD 和 TBLWT 指令的表指针操作

示例	表指针操作
TBLRD* TBLWT*	不修改 TBLPTR
TBLRD*+ TBLWT*+	TBLPTR 在读 / 写后递增
TBLRD*- TBLWT*-	TBLPTR 在读 / 写后递减
TBLRD+* TBLWT+*	TBLPTR 在读 / 写前递增

图 7-3： 基于操作的表指针边界



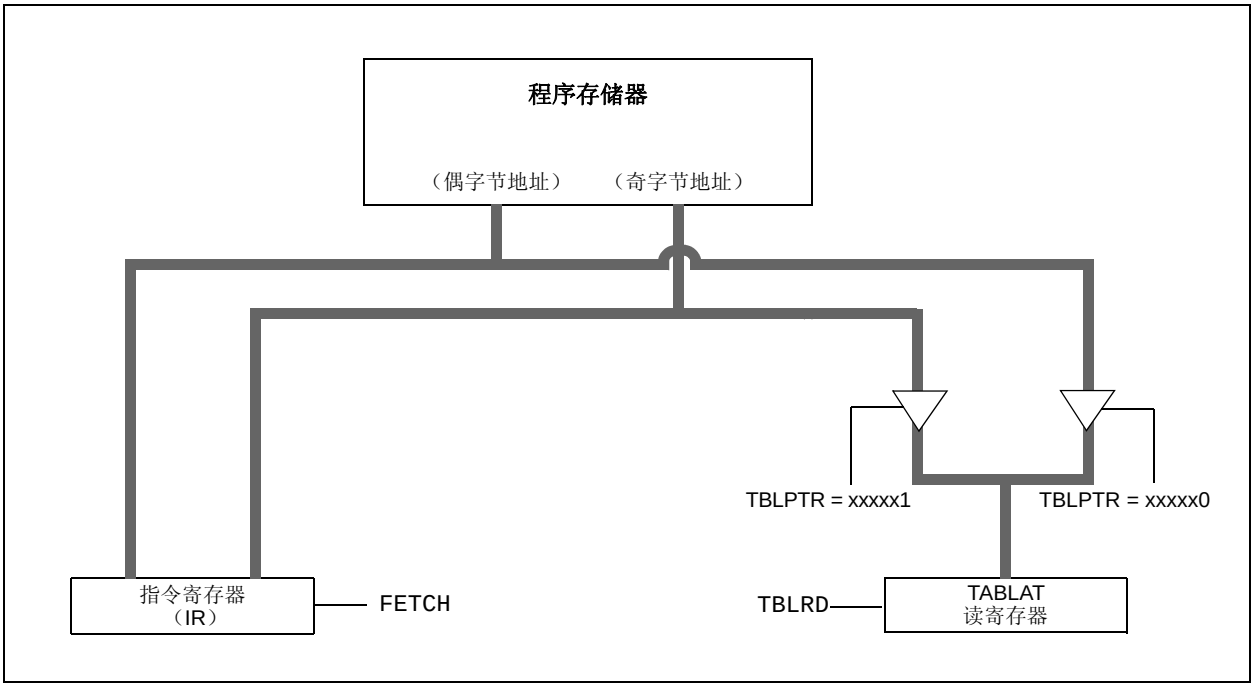
7.3 读闪存程序存储器

TBLRD 指令用于从程序存储器读取数据并放入数据 RAM。表读操作每次从程序存储器读取一个字节。

TBLPTR 指向程序存储空间的某个字节地址。执行 TBLRD 指令会将指向的字节装入 TABLAT。此外，还可以自动修改 TBLPTR 以进行下一次表读操作。

内部程序存储器通常以字为单位进行组织。由地址的最低有效位来选择字的高字节或低字节。图 7-4 显示了内部程序存储器和 TABLAT 之间的接口。

图 7-4：　　读闪存程序存储器



例 7-1：　　读闪存程序存储器的一个字

```
            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
            MOVF    TABLAT, W             ; get data
            MOVF    WORD_ODD
```

7.4 擦除闪存程序存储器

擦除块大小为 32 个字或 64 字节。

闪存阵列不支持字擦除。

当单片机自身启动一个擦除序列时，会擦除一个 64 字节的程序存储器块。高 16 位 TBLPTR<21:6> 指向要擦除的块。 TBLPTR<5:0> 被忽略。

擦除操作由 EECON1 寄存器控制。必须将 EEPGD 位置 1 以指向闪存程序存储器。WREN 位必须被置 1 以使能写操作。 FREE 位被置 1 以选择擦除操作。

为了安全起见，必须使用 EECON2 的写启动序列。

擦除内部闪存必须执行长写操作。在长写周期中，指令暂停执行。由内部编程定时器终止长写操作。

7.4.1 闪存程序存储器擦除序列

擦除内部程序存储器块的过程如下：

1. 将要擦除的行地址装入表指针寄存器。
2. 设置 EECON1 寄存器来执行擦除操作：
   - 将 EEPGD 位置 1 以指向程序存储器
   - 将 CFGS 位清零以访问程序存储器
   - 将 WREN 位置 1 以使能写操作
   - 将 FREE 位置 1 以使能擦除操作
3. 禁止中断。
4. 将 55h 写入 EECON2。
5. 将 0AAh 写入 EECON2。
6. 将 WR 位置 1。
   这将开始行擦除周期。
   CPU 在擦除期间 （Tiw）将会停止工作 （见参数 D133A）。
7. 重新允许中断。

例 7-2： 擦除闪存程序存储器的一行

```
                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_ROW
                BSF     EECON1, EEPGD           ; point to Flash program memory
                BCF     EECON1, CFGS            ; access Flash program memory
                BSF     EECON1, WREN            ; enable write to memory
                BSF     EECON1, FREE            ; enable Row Erase operation
                BCF     INTCON, GIE             ; disable interrupts
                MOVLW   55h
                MOVWF   EECON2                  ; write 55h
必需的序列      MOVLW   0AAh
                MOVWF   EECON2                  ; write 0AAh
                BSF     EECON1, WR              ; start erase (CPU stall)
                BSF     INTCON, GIE             ; re-enable interrupts
```

7.5 写闪存程序存储器

编程块大小为 32 个字或 64 字节。

不支持字或字节编程。

在内部使用表写指令将需要写入闪存的内容装入保持寄存器中。表写操作使用 64 个保持寄存器进行编程。

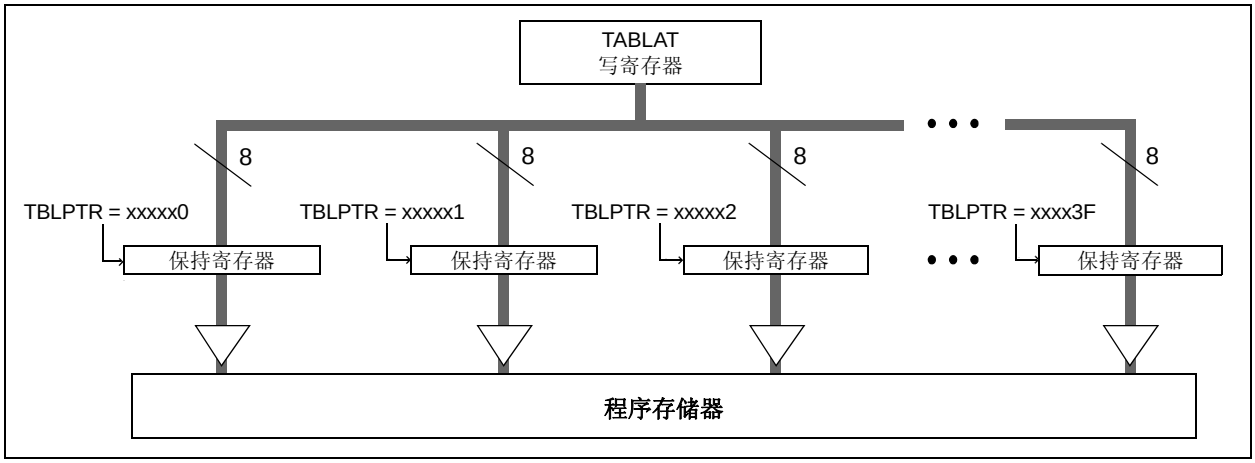
由于表锁存器（TABLAT）只是单字节寄存器，所以每次编程操作 TBLWT 指令可能需要执行 64 次。因为只写保持寄存器，所以所有的表写操作实际上都是短写。更新 64 个或 128 个保持寄存器后，必须写 EECON1 寄存器，以便启动长写周期开始编程操作。

对内部闪存编程要求使用长写操作。在长写周期中，指令暂停执行。由内部编程定时器终止长写操作。

由 EEPROM 片上定时器控制写入的时间。写入 / 擦除电压由片上的电荷泵产生，该电荷泵可以在器件的电压范围内工作。

注： 器件复位和写操作完成后保持寄存器的默认值为 FFh。将 FFh 写入保持寄存器不会修改其中的数值。这意味着可以修改程序存储器的各字节（假如不是想将任何位从 0 更改为 1）。当修改各字节时，无需在执行写操作前装入所有 64 个保持寄存器。

图 7-5： 对闪存程序存储器的表写操作



7.5.1 闪存程序存储器写序列

对内部程序存储单元编程的过程如下：

1. 将 64 字节读入 RAM。
2. 必要时更新 RAM 中的数据值。
3. 将要擦除的地址装入表指针寄存器。
4. 执行行擦除过程。
5. 将要写入的第一个字节的地址装入表指针寄存器。
6. 通过自动递增将 64 字节写入保持寄存器。
7. 设置 EECON1 寄存器来执行写操作：
   - 将 EEPGD 位置 1 以指向程序存储器
   - 将 CFGS 位清零以访问程序存储器
   - 将 WREN 置 1 以使能字节写操作
8. 禁止中断。
9. 将 55h 写入 EECON2。
10. 将 0AAh 写入 EECON2。
11. 将 WR 位置 1。这将开始写周期。CPU 在写入期间（Tiw，见参数 D133A）将会停止工作。
12. 重新允许中断。
13. 校验存储器（表读）。

下页中的例 7-3 给出了所需代码的示例。

注： 在将 WR 位置 1 前，表指针地址必须处于保持寄存器中的 64 字节预期地址范围内。

例 7-3：　　写闪存程序存储器

```
                MOVLW   SIZE_OF_BLOCK          ; 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   DATA_ADDR_HIGH         ; point to buffer
                MOVWF   FSR0H
                MOVLW   DATA_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
                BSF     EECON1, EEPGD          ; point to Flash program memory
                BCF     EECON1, CFGS           ; access Flash program memory
                BSF     EECON1, WREN           ; enable write to memory
                BSF     EECON1, FREE           ; enable Row Erase operation
                BCF     INTCON, GIE            ; disable interrupts
                MOVLW   55h
                MOVWF   EECON2                 ; write 55h
   必须的序列   MOVLW   0AAh
                MOVWF   EECON2                 ; write 0AAh
                BSF     EECON1, WR             ; start erase (CPU stall)
                BSF     INTCON, 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   SIZE_OF_BLOCK          ; number of bytes in holding register
                MOVWF   COUNTER
WRITE_BYTE_TO_HREGS
                MOVFF   POSTINC0, WREG         ; 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 buffers are full
                BRA     WRITE_BYTE_TO_HREGS
```

例 7-3：　写闪存程序存储器 （续）

```
PROGRAM_MEMORY
                BSF     EECON1, EEPGD       ; point to Flash program memory
                BCF     EECON1, CFGS        ; access Flash program memory
                BSF     EECON1, WREN        ; enable write to memory
                BCF     INTCON, GIE         ; disable interrupts
                MOVLW   55h
                MOVWF   EECON2              ; write 55h
必需的序列      MOVLW   0AAh
                MOVWF   EECON2              ; write 0AAh
                BSF     EECON1, WR          ; start program (CPU stall)
                BSF     INTCON, GIE         ; re-enable interrupts
                BCF     EECON1, WREN        ; disable write to memory
```

7.5.2　写校验

根据具体应用，将写入存储器的值对照原始值进行校验是一个很好的编程习惯。在应用中，如果某些位的写次数接近规定极限值，就应该进行写校验。

7.5.3　意外终止写操作

如果由于意外事件 （如掉电或意外复位）终止了写操作，应该对刚刚编程的存储单元进行校验，如有必要，还要重新进行编程。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，用户可以检查 WRERR 位以确定是否需要重写该单元。

7.5.4　防止误写操作的保护措施

为防止对闪存程序存储器的误写操作，必须遵循写操作的启动顺序。更多详细信息，请参见**第 28.0 节 “CPU 的特殊功能 ”**。

7.6　代码保护期间闪存程序存储器的操作

关于闪存程序存储器代码保护的详细信息，请参见**第 28.6 节 “ 程序校验和代码保护 ”**。

表 7-2：　与闪存程序存储器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TBLPTRU	—	—	bit 21(1)	程序存储器表指针最高字节（TBLPTR<20:16>）				
TBPLTRH	程序存储器表指针高字节 （TBLPTR<15:8>）							
TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）							
TABLAT	程序存储器表锁存器							
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
EECON2	EEPROM 控制寄存器 2 （不是实际存在的寄存器）							
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIE4	TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE

图注：　— = 未实现，读为 0。闪存 /EEPROM 访问期间不使用阴影单元。

注　1：　TBLPTRU 的 bit 21 允许访问器件配置位。

注：

8.0 数据 EEPROM 存储器

数据 EEPROM 是非易失性的存储器阵列，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 范围内的正常运行期间， EEPROM 是可读写的。

有 5 个 SFR 用于读写数据 EEPROM 以及程序存储器。它们是：

- EECON1
- EECON2
- EEDATA
- EEADR
- EEADRH

数据 EEPROM 允许以字节为单位读写。当与数据存储器模块接口时， EEDATA 存放 8 位读写数据，而 EEADRH:EEADR 寄存器对存放被访问的 EEPROM 存储单元的地址。

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存储单元并写入新数据（在写入前擦除）。写入时间由片上定时器控制，其值根据电压、温度和不同的芯片而不同。具体的限制值，请参见参数 D122（**第 31.0 节 “ 电气特性 ”** 中的表 31-1）。

8.1 EEADR 和 EEADRH 寄存器

EEADRH:EEADR 寄存器对用于寻址数据 EEPROM 以进行读写操作。 EEADRH 保存地址的高 2 位；高 6 位被忽略。 10 位的寄存器对可寻址 1024 字节 （00h 至 3FFh）的存储器范围。

8.2 EECON1 和 EECON2 寄存器

对数据 EEPROM 的访问由 EECON1 和 EECON2 两个寄存器控制。它们也用来控制对程序存储器的访问，使用方法与访问数据 EEPROM 类似。

EECON1 寄存器（寄存器 8-1）是数据和程序存储器访问的控制寄存器。控制位 EEPGD 决定访问的是程序存储器还是数据 EEPROM 存储器。当 EEPGD 清零时，操作将访问数据 EEPROM 存储器。当 EEPGD 置 1 时，将访问程序存储器。

控制位 CFGS 决定访问的是配置寄存器还是程序存储器 / 数据 EEPROM 存储器。当 CFGS 置 1 时，后续操作将访问配置寄存器。当 CFGS 清零时，则由 EEPGD 位来选择具体访问闪存程序存储器还是数据 EEPROM 存储器。

当 WREN 位置 1 时，允许进行写操作。上电时，WREN 位被清零。WRERR 位在 WREN 位置 1 时由硬件置 1，在内部编程定时器超时、写操作结束时被清零。

注：	在正常操作期间， WRERR 读为 1。这表明写操作被复位提早终止或进行了不合法的写操作。

WR 控制位用于启动写操作。用软件只能将该位置 1 而无法清零。在写操作完成后，由硬件将其清零。

注：	当写操作完成时， EEIF 中断标志位（PIR4<6>）被置 1。它必须用软件清零。

控制位 RD 和 WR 分别启动读和擦写操作。这两位由固件置 1，并在操作完成时由硬件清零。

当访问程序存储器（EEPGD = 1）时，RD 位无法置 1。程序存储器是通过表读指令读取的。关于表读的信息，请参见**第 7.1 节 “ 表读与表写 ”**。

EECON2 寄存器不是实际存在的寄存器。它专用于存储器写和擦除过程。读 EECON2 将得到全 0。

寄存器 8-1：　　EECON1：数据 EEPROM 控制寄存器 1

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR(1)	WREN	WR	RD
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　**EEPGD**：闪存程序存储器或数据 EEPROM 存储器选择位
1 = 访问闪存程序存储器
0 = 访问数据 EEPROM 存储器

bit 6　**CFGS**：闪存程序存储器 / 数据 EEPROM 存储器或配置寄存器选择位
1 = 访问配置寄存器
0 = 访问闪存程序存储器或数据 EEPROM 存储器

bit 5　**未实现**：读为 0

bit 4　**FREE**：闪存行擦除使能位
1 = 在下一条 WR 命令时擦除 TBLPTR 指定的程序存储器行 （擦除操作完成后清零）
0 = 仅执行写操作

bit 3　**WRERR**：闪存程序存储器 / 数据 EEPROM 存储器错误标志位 (1)
1 = 写操作提早终止 （由于正常操作中自定时编程期间的任何复位，或不合法的写操作）
0 = 写操作完成

bit 2　**WREN**：闪存程序存储器 / 数据 EEPROM 存储器写使能位
1 = 允许对闪存程序存储器 / 数据 EEPROM 存储器的写周期
0 = 禁止对闪存程序存储器 / 数据 EEPROM 存储器的写周期

bit 1　**WR**：写控制位
1 = 启动数据 EEPROM 擦除 / 写周期或者程序存储器的擦除周期或写周期。
（操作是自定时的，一旦写操作完成，该位即由硬件清零。
用软件只能将 WR 位置 1，但不能清零。）
0 = EEPROM 写周期完成

bit 0　**RD**：读控制位
1 = 启动 EEPROM 读操作
（读操作需要一个周期。RD 由硬件清零。用软件只能将 RD 位置 1，但不能清零。EEPGD = 1 或 CFGS = 1 时， RD 位无法置 1。）
0 = 不启动 EEPROM 读操作

注　1： 当发生 WRERR 时， EEPGD 和 CFGS 位不会被清零。这样可以跟踪错误情况。

8.3 读数据 EEPROM 存储器

要读取数据存储单元，用户必须将地址写入 EEADRH:EEADR 寄存器对，清零 EEPGD 控制位（EECON1<7>），然后将控制位 RD （EECON1<0>）置 1。可在一个指令周期后访问 EEDATA 寄存器中的数据。可在一条 NOP 指令后读取。EEDATA 会将此值保存至下一次读取或用户向该单元写入数据时（写操作）为止。

基本过程如例 8-1 中所示。

8.4 写数据 EEPROM 存储器

要向 EEPROM 数据存储单元写入数据，必须首先将地址写入 EEADRH:EEADR 寄存器对，并将数据写入 EEDATA 寄存器。必须遵循例 8-2 中的序列启动写周期。

如果没有完全遵循该指令序列（即首先将 55h 写入 EECON2，随后将 0AAh 写入 EECON2，最后将 WR 位置 1）写每个字节，将不会启动写操作。强烈建议在这段代码执行期间禁止中断。

此外，必须将 EECON1 中的 WREN 位置 1 以使能写操作。这种机制可防止由于意外执行代码 （即程序失控）导致误写数据 EEPROM。除了更新 EEPROM 时以外，WREN 位应始终保持清零。WREN 位不会被硬件清零。

一个写序列启动后，EECON1、EEADRH:EEADR 和 EEDATA 不会被修改。除非 WREN 位置 1，否则禁止将 WR 位置 1。WREN 位必须由前一条指令置 1。WR 和 WREN 不能被同一条指令置 1。

写周期完成后， WR 位由硬件清零并且 EEPROM 中断标志位（EEIF）被置 1。用户可以允许此中断或查询此位； EEIF 必须用软件清零。

8.5 写校验

根据具体应用，将写入存储器的值对照原始值进行校验是一个很好的编程习惯。在应用中，如果某些位的写次数接近规定极限值，就应该进行写校验。

注：	在 LP 振荡器 （低功耗）模式下运行时，不能执行对闪存和 EEPROM 存储器的自写操作。执行自写操作会将器件置于高功耗模式。

例 8-1：　　读数据 EEPROM

```
        MOVLW   DATA_EE_ADDRH      ;
        MOVWF   EEADRH             ; Upper bits of Data Memory Address to read
        MOVLW   DATA_EE_ADDR       ;
        MOVWF   EEADR              ; Lower bits of Data Memory Address to read
        BCF     EECON1, EEPGD      ; Point to DATA memory
        BCF     EECON1, CFGS       ; Access EEPROM
        BSF     EECON1, RD         ; EEPROM Read
        NOP
        MOVF    EEDATA, W          ; W = EEDATA
```

例 8-2：　　写数据 EEPROM

```
                MOVLW   DATA_EE_ADDRH      ;
                MOVWF   EEADRH             ; Upper bits of Data Memory Address to write
                MOVLW   DATA_EE_ADDR       ;
                MOVWF   EEADR              ; Lower bits of Data Memory Address to write
                MOVLW   DATA_EE_DATA       ;
                MOVWF   EEDATA             ; Data Memory Value to write
                BCF     EECON1, EEPGD      ; Point to DATA memory
                BCF     EECON1, CFGS       ; Access EEPROM
                BSF     EECON1, WREN       ; Enable writes

                BCF     INTCON, GIE        ; Disable Interrupts
                MOVLW   55h                ;
                MOVWF   EECON2             ; Write 55h
必需的序列      MOVLW   0AAh               ;
                MOVWF   EECON2             ; Write 0AAh
                BSF     EECON1, WR         ; Set WR bit to begin write
                BTFSC   EECON1, WR         ; Wait for write to complete GOTO $-2
                BSF     INTCON, GIE        ; Enable Interrupts

                                           ; User code execution
                BCF     EECON1, WREN       ; Disable writes on write complete (EEIF set)
```

8.6 代码保护期间的操作

数据 EEPROM 存储器在配置字中有它自己的代码保护位。如果代码保护机制被使能，外部读写操作就被禁止。

单片机本身可以读写内部数据 EEPROM，与代码保护配置位的状态无关。更多信息，请参见**第 28.0 节 “CPU 的特殊功能 ”**。

8.7 防止误写操作的保护措施

有些情况下，器件并不希望写入数据EEPROM存储器。为了防止 EEPROM 误写操作，器件实现了各种保护机制。上电时， WREN 位被清零。而且，上电延时期间（TPWRT，参数 33）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，写操作的启动序列以及 WREN 位可共同防止意外写操作的发生。

8.8 使用数据 EEPROM

数据 EEPROM 是高耐用性可字节寻址的阵列，已将其优化以便存储频繁变动的信息（例如，程序变量或其他经常更新的数据）。频繁变动值的更新频率通常高于规范 D124 中的规定。如果情况并非如此，必须执行阵列刷新。出于此原因，不经常更新的变量（如常量、ID 和校准值等）应存储在闪存程序存储器中。

简单的数据 EEPROM 刷新程序如例 8-3 中所示。

注：	如果数据 EEPROM 仅用于存储常量和 / 或很少改变的数据，没有必要执行阵列刷新。请参见规范 D124。

例 8-3： 数据 EEPROM 刷新程序

```
        CLRF    EEADR               ; Start at address 0
        CLRF    EEADRH              ;
        BCF     EECON1, CFGS        ; Set for memory
        BCF     EECON1, EEPGD       ; Set for Data EEPROM
        BCF     INTCON, GIE         ; Disable interrupts
        BSF     EECON1, WREN        ; Enable writes
LOOP                                ; Loop to refresh array
        BSF     EECON1, RD          ; Read current address
        MOVLW   55h                 ;
        MOVWF   EECON2              ; Write 55h
        MOVLW   0AAh                ;
        MOVWF   EECON2              ; Write 0AAh
        BSF     EECON1, WR          ; Set WR bit to begin write
        BTFSC   EECON1, WR          ; Wait for write to complete
        BRA     $-2
        INCFSZ  EEADR, F            ; Increment address
        BRA     LOOP                ; Not zero, do it again
        INCFSZ  EEADRH, F           ; Increment the high address
        BRA     LOOP                ; Not zero, do it again

        BCF     EECON1, WREN        ; Disable writes
        BSF     INTCON, GIE         ; Enable interrupts
```

表 8-1：　　与数据 EEPROM 存储器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
EEADRH	EEPROM 地址寄存器的高字节							
EEADR	EEPROM 地址寄存器的低字节							
EEDATA	EEPROM 数据寄存器							
EECON2	EEPROM 控制寄存器 2（不是实际存在的寄存器）							
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIE4	TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE

图注：　　— = 未实现，读为 0。闪存 /EEPROM 访问期间不使用阴影单元。

9.0 8 x 8 硬件乘法器

9.1 简介

所有 PIC18 器件均包含一个 8 x 8 硬件乘法器（是 ALU的一部分）。该乘法器可执行无符号运算并产生一个 16 位运算结果，该结果存储在一对乘积寄存器 PRODH:PRODL 中。该乘法器执行的运算不会影响 STATUS 寄存器中的任何标志。

通过硬件执行乘法运算只需要一个指令周期。硬件乘法器具有更高的计算吞吐量并缩短了乘法算法的代码长度，从而可在许多先前仅能使用数字信号处理器的应用中使用 PIC18 器件。表 9-1 给出了硬件和软件乘法运算的比较，包括所需存储空间和执行时间。

9.2 工作原理

例 9-1 给出了一个 8 x 8 无符号乘法运算的指令序列。当已在 WREG 寄存器中装入了一个参数时，实现该运算仅需一条指令。

例 9-2 给出了一个 8 x 8 有符号乘法运算的指令序列。要弄清参数的符号位，必须检查每个参数的最高有效位（Most Significant bit， MSb），并做相应的减法。

例 9-1： 8 x 8 无符号乘法程序

```
MOVF    ARG1, W     ;
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
```

例 9-2： 8 x 8 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 9-1： 各种乘法运算的性能比较

程序	乘法实现方法	程序存储器（字）	周期数（最多）	时间			
				64 MHz 时	48 MHz 时	10 MHz 时	4 MHz 时
8 x 8 无符号	软件乘法	13	69	4.3 µs	5.7 µs	27.6 µs	69 µs
	硬件乘法	1	1	62.5 ns	83.3 ns	400 ns	1 µs
8 x 8 有符号	软件乘法	33	91	5.6 µs	7.5 µs	36.4 µs	91 µs
	硬件乘法	6	6	375 ns	500 ns	2.4 µs	6 µs
16 x 16 无符号	软件乘法	21	242	15.1 µs	20.1 µs	96.8 µs	242 µs
	硬件乘法	28	28	1.7 µs	2.3 µs	11.2 µs	28 µs
16 x 16 有符号	软件乘法	52	254	15.8 µs	21.2 µs	101.6 µs	254 µs
	硬件乘法	35	40	2.5 µs	3.3 µs	16.0 µs	40 µs

例 9-3 给出了一个 16 x 16 无符号乘法运算的指令序列。公式 9-1 为所使用的算法。32 位结果存储在 4 个寄存器（RES3:RES0）中。

公式 9-1： 16 x 16 无符号乘法算法

RES3:RES0 = ARG1H:ARG1L • ARG2H:ARG2L
= (ARG1H • ARG2H • 2^{16}) +
(ARG1H • ARG2L • 2^{8}) +
(ARG1L • ARG2H • 2^{8}) +
(ARG1L • ARG2L)

例 9-3： 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 9-4 给出了一个 16 x 16 有符号乘法运算的指令序列。公式 9-2 为所使用的算法。32 位结果存储在 4 个寄存器（RES3:RES0）中。要弄清参数的符号位，必须检查每个参数对的 MSb，并做相应的减法。

公式 9-2： 16 x 16 有符号乘法算法

RES3:RES0 = ARG1H:ARG1L • ARG2H:ARG2L
= (ARG1H • ARG2H • 2^{16}) +
(ARG1H • ARG2L • 2^{8}) +
(ARG1L • ARG2H • 2^{8}) +
(ARG1L • ARG2L) +
(-1 • ARG2H<7> • ARG1H:ARG1L • 2^{16}) +
(-1 • ARG1H<7> • ARG2H:ARG2L • 2^{16})

例 9-4： 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
   :
```

10.0 中断

PIC18F66K80 系列器件具有多个中断源及中断优先级功能，该功能可以给大多数中断源分配高优先级或者低优先级。高优先级中断向量位于 0008h，低优先级中断向量位于 0018h。高优先级中断事件可以中断正在处理的低优先级中断。

用于控制中断操作的寄存器是：

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1、 PIR2、 PIR3、 PIR4 和 PIR5
- PIE1、 PIE2、 PIE3、 PIE4 和 PIE5
- IPR1、 IPR2、 IPR3、 IPR4 和 IPR5

建议使用 MPLAB® IDE 提供的 Microchip 头文件命名这些寄存器中的位。这使得汇编器 / 编译器能够自动识别指定寄存器中的这些位。

通常，中断源有 3 个位用于控制其操作。这些位分别是：

- **标志位** —— 指示发生了中断事件
- **允许位** —— 允许程序跳转到中断向量地址处执行（当标志位置 1 时）
- **优先级位** —— 指定高优先级还是低优先级

通过将 IPEN 位（RCON<7>）置 1，可使能中断优先级功能。当使能中断优先级时，有 2 个全局中断允许位。将 GIEH 位 （INTCON<7>）置 1，可允许所有优先级位置 1（高优先级）的中断。将 GIEL 位（INTCON<6>）置 1，可允许所有优先级位清零（低优先级）的中断。当中断标志位、允许位及相应的全局中断允许位均被置 1 时，中断将根据优先级位的设置立即跳转到地址 0008h 或 0018h。也可以通过设置相应的中断允许位来禁止个别中断。

当 IPEN 位清零（默认状态）时，便会禁止中断优先级功能，此时中断是与 PIC® 中档器件兼容的。在兼容模式下，各个中断源的中断优先级位不起作用。INTCON<6> 是 PEIE 位，用于允许 / 禁止所有外设中断源。INTCON<7> 是 GIE 位，用于允许 / 禁止所有中断源。在兼容模式下，所有中断均跳转到地址 0008h。

当响应中断时，全局中断允许位被清零以禁止后续中断。清零后的 IPEN 位就是 GIE 位。如果使用了中断优先级，这个位就是 GIEH 或 GIEL 位。高优先级中断源会中断低优先级中断。在处理高优先级中断时，低优先级中断将不被处理。

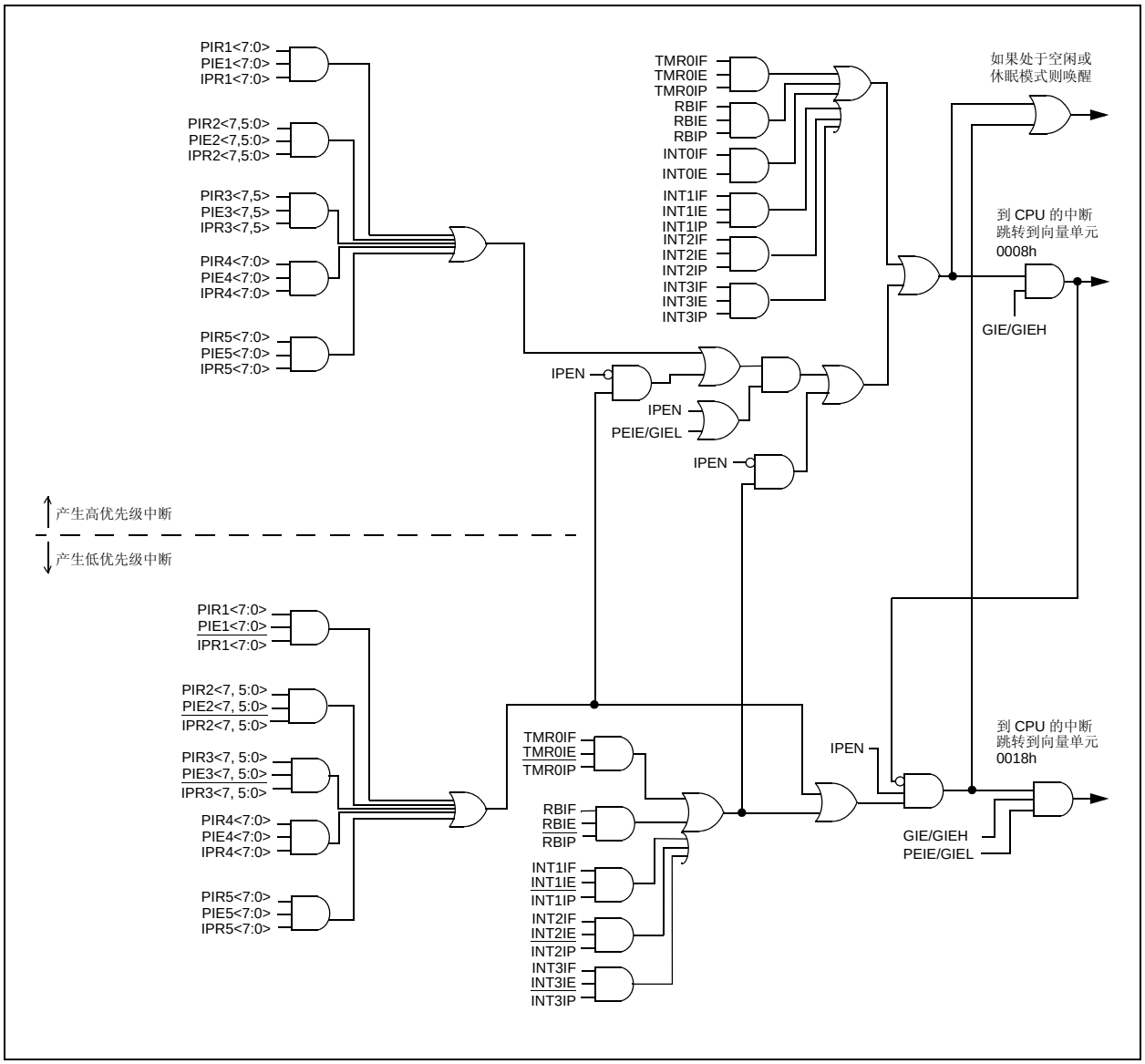
返回地址被压入堆栈，中断向量地址（0008h 或 0018h）被装入 PC。可以在中断服务程序 （Interrupt Service Routine， ISR）中，通过查询中断标志位来确定中断源。在重新允许中断前，必须用软件将中断标志位清零，以避免重复响应同一中断。

执行 “ 从中断返回 ” 指令 RETFIE 将退出中断程序，同时将 GIE 位（若使用中断优先级则为 GIEH 或 GIEL 位）置 1，从而重新允许中断。

对于外部中断事件，例如 INTx 引脚中断或者 PORTB 输入电平变化中断，中断响应延时将会是 3 至 4 个指令周期。对于单周期或双周期指令，中断响应延时完全相同。各中断标志位的置 1 不受对应的中断允许位和 GIE 位状态的影响。

注：	当允许**任何**中断时，不要使用 MOVFF 指令修改中断控制寄存器。否则可能导致单片机操作出错。

图 10-1：　　PIC18F66K80 系列中断逻辑



10.1 INTCON 寄存器

INTCON 寄存器是可读写的寄存器，包含各个中断允许位、优先级位和标志位。

注： 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。用户软件应在允许一个中断前，先将相应的中断标志位清零。这样做便可用软件查询中断标志位。

寄存器 10-1： INTCON：中断控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE(2)	TMR0IF	INT0IF	RBIF(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 **GIE/GIEH**：全局中断允许位
当 IPEN = 0 时：
1 = 允许所有未被屏蔽的中断
0 = 禁止所有中断
当 IPEN = 1 时：
1 = 允许所有高优先级中断
0 = 禁止所有中断

bit 6 **PEIE/GIEL**：外设中断允许位
当 IPEN = 0 时：
1 = 允许所有未被屏蔽的外设中断
0 = 禁止所有外设中断
当 IPEN = 1 时：
1 = 允许所有低优先级的外设中断
0 = 禁止所有低优先级的外设中断

bit 5 **TMR0IE**：TMR0 溢出中断允许位
1 = 允许 TMR0 溢出中断
0 = 禁止 TMR0 溢出中断

bit 4 **INT0IE**：INT0 外部中断允许位
1 = 允许 INT0 外部中断
0 = 禁止 INT0 外部中断

bit 3 **RBIE**：RB 端口电平变化中断允许位 (2)
1 = 允许 RB 端口电平变化中断
0 = 禁止 RB 端口电平变化中断

bit 2 **TMR0IF**：TMR0 溢出中断标志位
1 = TMR0 寄存器已溢出 （必须用软件清零）
0 = TMR0 寄存器未溢出

bit 1 **INT0IF**：INT0 外部中断标志位
1 = 发生了 INT0 外部中断 （必须用软件清零）
0 = 未发生 INT0 外部中断

bit 0 **RBIF**：RB 端口电平变化中断标志位 (1)
1 = RB<7:4> 引脚中至少有一个引脚的电平状态发生了改变 （必须用软件清零）
0 = 所有 RB<7:4> 引脚的电平状态都没有改变

注 1： 不匹配条件会继续将该位置 1。读取 PORTB 并等待一个额外的指令周期将结束不匹配条件，并允许将该位清零。
2： 每个 PORTB 电平变化中断引脚在 IOCB 寄存器中单独使能和禁止。默认情况下，所有引脚都是使能的。

寄存器 10-2：　　INTCON2：中断控制寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-x	R/W-1	R/W-x	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP
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　**$\overline{\text{RBPU}}$**：PORTB 上拉使能位
1 = 禁止所有 PORTB 上拉
0 = 根据各端口锁存值使能 PORTB 上拉

bit 6　**INTEDG0**：外部中断 0 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 5　**INTEDG1**：外部中断 1 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 4　**INTEDG2**：外部中断 2 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 3　**INTEDG3**：外部中断 3 边沿选择位
1 = 上升沿触发中断
0 = 下降沿触发中断

bit 2　**TMR0IP**：TMR0 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 1　**INT3IP**：INT3 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit 0　**RBIP**：RB 端口电平变化中断优先级位
1 = 高优先级
0 = 低优先级

注：　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。用户软件应在允许一个中断前，先将相应的中断标志位清零。这样做便可用软件查询中断标志位。

寄存器 10-3： INTCON3：中断控制寄存器 3

R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-x	R/W-0	R/W-0
INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
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 **INT2IP**：INT2 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **INT1IP**：INT1 外部中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **INT3IE**：INT3 外部中断允许位
1 = 允许 INT3 外部中断
0 = 禁止 INT3 外部中断

bit 4 **INT2IE**：INT2 外部中断允许位
1 = 允许 INT2 外部中断
0 = 禁止 INT2 外部中断

bit 3 **INT1IE**：INT1 外部中断允许位
1 = 允许 INT1 外部中断
0 = 禁止 INT1 外部中断

bit 2 **INT3IF**：INT3 外部中断标志位
1 = 发生了 INT3 外部中断 （必须用软件清零）
0 = 未发生 INT3 外部中断

bit 1 **INT2IF**：INT2 外部中断标志位
1 = 发生了 INT2 外部中断 （必须用软件清零）
0 = 未发生 INT2 外部中断

bit 0 **INT1IF**：INT1 外部中断标志位
1 = 发生了 INT1 外部中断 （必须用软件清零）
0 = 未发生 INT1 外部中断

注： 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。用户软件应在允许一个中断前，先将相应的中断标志位清零。这样做便可用软件查询中断标志位。

10.2 PIR 寄存器

PIR 寄存器包含各外设中断的标志位。根据外设中断源的数量，有 6 个外设中断请求 （标志）寄存器 （PIR1 至 PIR5）。

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 GIE（INTCON<7>）的状态如何，中断标志位都将置 1。

2： 用户软件应在允许中断前和处理完中断后，将相应的中断标志位清零。

寄存器 10-4： PIR1：外设中断请求 （标志）寄存器 1

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
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 **PSPIF：**并行从端口读 / 写中断标志位
1 = 发生了读或写操作 （必须用软件清零）
0 = 未发生读或写操作

bit 6 **ADIF：**A/D 转换器中断标志位
1 = A/D 转换已完成 （必须用软件清零）
0 = A/D 转换未完成

bit 5 **RC1IF：**EUSART 接收中断标志位
1 = EUSART 接收缓冲区 RCREG1 已满 （读取 RCREG1 时清零）
0 = EUSART 接收缓冲区为空

bit 4 **TX1IF：**EUSART 发送中断标志位
1 = EUSART 发送缓冲区 TXREG1 为空 （写入 TXREG1 时清零）
0 = EUSART 发送缓冲区已满

bit 3 **SSPIF：**主同步串口中断标志位
1 = 发送 / 接收已完成 （必须用软件清零）
0 = 等待发送 / 接收

bit 2 **TMR1GIF：**Timer1 门控中断标志位
1 = 发生了定时器门控中断 （必须用软件清零）
0 = 未发生定时器门控中断

bit 1 **TMR2IF：**TMR2 与 PR2 匹配中断标志位
1 = TMR2 与 PR2 发生匹配 （必须用软件清零）
0 = TMR2 与 PR2 未发生匹配

bit 0 **TMR1IF：**TMR1 溢出中断标志位
1 = TMR1 寄存器已溢出 （必须用软件清零）
0 = TMR1 寄存器未溢出

寄存器 10-5： PIR2：外设中断请求 （标志）寄存器 2

R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIF	—	—	—	BCLIF	HLVDIF	TMR3IF	TMR3GIF
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 **OSCFIF：**振荡器故障中断标志位
1 = 器件振荡器发生故障，改由 INTOSC 作为时钟输入 （必须用软件清零）
0 = 器件时钟正常工作

bit 6-4 **未实现：**读为 0

bit 3 **BCLIF：**总线冲突中断标志位
1 = 发生了总线冲突 （必须用软件清零）
0 = 未发生总线冲突

bit 2 **HLVDIF：**高 / 低压检测中断标志位
1 = 发生了低压条件 （必须用软件清零）
0 = 器件电压高于稳压器的低压跳变点

bit 1 **TMR3IF：**TMR3 溢出中断标志位
1 = TMR3 寄存器已溢出 （必须用软件清零）
0 = TMR3 寄存器未溢出

bit 0 **TMR3GIF：**TMR3 门控中断标志位
1 = 发生了定时器门控中断 （必须用软件清零）
0 = 未发生定时器门控中断

寄存器 10-6：　　PIR3：外设中断请求 （标志）寄存器 3

U-0	U-0	R-0	R-0	R/W-0	R/W-0	R/W-0	U-0
—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
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　**未实现：**读为 0

bit 5　**RC2IF：**EUSART 接收中断标志位
1 = EUSART 接收缓冲区 RCREG2 已满 （读取 RCREG2 时清零）
0 = EUSART 接收缓冲区为空

bit 4　**TX2IF：**EUSART 发送中断标志位
1 = EUSART 发送缓冲区 TXREG2 为空 （写入 TXREG2 时清零）
0 = EUSART 发送缓冲区已满

bit 3　**CTMUIF：**CTMU 中断标志位
1 = 发生了 CTMU 中断 （必须用软件清零）
0 = 未发生 CTMU 中断

bit 2　**CCP2IF：**CCP2 中断标志位
捕捉模式：
1 = 发生了 TMR1/TMR3 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕捉
比较模式：
1 = 发生了 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

bit 1　**CCP1IF：**ECCP1 中断标志位
捕捉模式：
1 = 发生了 TMR1/TMR3 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕捉
比较模式：
1 = 发生了 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

bit 0　**未实现：**读为 0

寄存器 10-7： PIR4：外设中断请求 （标志）寄存器 4

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
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 **TMR4IF**：TMR4 溢出中断标志位
1 = TMR4 寄存器已溢出 （必须用软件清零）
0 = TMR4 寄存器未溢出

bit 6 **EEIF**：数据 EEDATA/ 闪存写操作中断标志位
1 = 写操作完成 （必须用软件清零）
0 = 写操作未完成或尚未开始

bit 5 **CMP2IF**：CMP2 中断标志位
1 = 发生了 CMP2 中断 （必须用软件清零）
0 = 未发生 CMP2 中断

bit 4 **CMP1IF**：CMP1 中断标志位
1 = 发生了 CMP1 中断 （必须用软件清零）
0 = 未发生 CMP1 中断

bit 3 **未实现**：读为 0

bit 2 **CCP5IF**：CCP5 中断标志位
捕捉模式
1 = 发生了 TMR 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR 寄存器捕捉
比较模式
1 = 发生了 TMR 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR 寄存器的比较匹配
PWM 模式
在 PWM 模式下未使用。

bit 1 **CCP4IF**：CCP4 中断标志位
捕捉模式
1 = 发生了 TMR 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR 寄存器捕捉
比较模式
1 = 发生了 TMR 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR 寄存器的比较匹配
PWM 模式
在 PWM 模式下未使用。

bit 0 **CCP3IF**：CCP3 中断标志位
捕捉模式
1 = 发生了 TMR 寄存器捕捉 （必须用软件清零）
0 = 未发生 TMR 寄存器捕捉
比较模式
1 = 发生了 TMR 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TMR 寄存器的比较匹配
PWM 模式
在 PWM 模式下未使用。

寄存器 10-8： PIR5：外设中断请求 （标志）寄存器 5

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	TXB0IF	RXB1IF	RXB0IF/ FIFOFIF
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**：收到无效报文中断标志位
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**：总线唤醒活动中断标志位
1 = CAN 总线上发生了活动
0 = CAN 总线上未发生任何活动

bit 5 **ERRIF**：错误中断标志位 （COMSTAT 寄存器中的多个来源）
1 = CAN 模块中发生了错误 （多个来源）
0 = 未发生 CAN 模块错误

bit 4 **TXB2IF**：发送缓冲区 2 中断标志位
1 = 发送缓冲区 2 已完成报文发送，可以向其中重新装入数据
0 = 发送缓冲区 2 尚未完成报文发送

bit 3 **TXB1IF**：发送缓冲区 1 中断标志位
1 = 发送缓冲区 1 已完成报文发送，可以向其中重新装入数据
0 = 发送缓冲区 1 尚未完成报文发送

bit 2 **TXB0IF**：发送缓冲区 0 中断标志位
1 = 发送缓冲区 0 已完成报文发送，可以向其中重新装入数据
0 = 发送缓冲区 0 尚未完成报文发送

bit 1 **RXB1IF**：接收缓冲区 1 中断标志位
模式 0：
1 = CAN 接收缓冲区 1 已接收到新报文
0 = CAN 接收缓冲区 1 未接收到新报文
模式 1 和 2：
1 = CAN 接收缓冲区 /FIFO 已接收到新报文
0 = CAN 接收缓冲区 /FIFO 未接收到新报文

bit 0 位操作取决于选定的模式：
模式 0：
RXB0IF：接收缓冲区 0 中断标志位
1 = CAN 接收缓冲区 0 已接收到新报文
0 = CAN 接收缓冲区 0 未接收到新报文
模式 1：
未实现：读为 0
模式 2：
FIFOFIF：FIFO 满中断标志位
1 = FIFO 已达到 FIFO_HF 位所定义的满状态
0 = FIFO 未达到 FIFO_HF 位所定义的满状态

10.3 PIE 寄存器

PIE 寄存器包含各外设中断的允许位。根据外设中断源的数量，有 6 个外设中断允许寄存器（PIE1 至 PIE6）。当 IPEN（RCON<7>）= 0 时，要允许任一外设中断，必须将 PEIE 位置 1。

寄存器 10-9： PIE1：外设中断允许寄存器 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
PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
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 **PSPIE**：并行从端口读 / 写中断允许位
1 = 允许 PSP 读 / 写中断
0 = 禁止 PSP 读 / 写中断

bit 6 **ADIE**：A/D 转换器中断允许位
1 = 允许 A/D 中断
0 = 禁止 A/D 中断

bit 5 **RC1IE**：EUSART 接收中断允许位
1 = 允许 EUSART 接收中断
0 = 禁止 EUSART 接收中断

bit 4 **TX1IE**：EUSART 发送中断允许位
1 = 允许 EUSART 发送中断
0 = 禁止 EUSART 发送中断

bit 3 **SSPIE**：主同步串口中断允许位
1 = 允许 MSSP 中断
0 = 禁止 MSSP 中断

bit 2 **TMR1GIE**：TMR1 门控中断允许位
1 = 允许门控中断
0 = 禁止门控中断

bit 1 **TMR2IE**：TMR2 与 PR2 匹配中断允许位
1 = 允许 TMR2 与 PR2 匹配中断
0 = 禁止 TMR2 与 PR2 匹配中断

bit 0 **TMR1IE**：TMR1 溢出中断允许位
1 = 允许 TMR1 溢出中断
0 = 禁止 TMR1 溢出中断

寄存器 10-10：　　PIE2：外设中断允许寄存器 2

R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	—	—	—	BCLIE	HLVDIE	TMR3IE	TMR3GIE
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　**OSCFIE**：振荡器故障中断允许位
1 = 允许
0 = 禁止

bit 6-4　**未实现**：读为 0

bit 3　**BCLIE**：总线冲突中断允许位
1 = 允许
0 = 禁止

bit 2　**HLVDIE**：高 / 低压检测中断允许位
1 = 允许
0 = 禁止

bit 1　**TMR3IE**：TMR3 溢出中断允许位
1 = 允许
0 = 禁止

bit 0　**TMR3GIE**：Timer3 门控中断允许位
1 = 允许
0 = 禁止

寄存器 10-11：　　PIE3：外设中断允许寄存器 3

U-0	U-0	R-0	R-0	R/W-0	R/W-0	R/W-0	U-0
—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
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　**未实现：**读为 0

bit 5　**RC2IE：**EUSART 接收中断允许位
1 = 允许
0 = 禁止

bit 4　**TX2IE：**EUSART 发送中断允许位
1 = 允许
0 = 禁止

bit 3　**CTMUIE：**CTMU 中断允许位
1 = 允许
0 = 禁止

bit 2　**CCP2IE：**CCP2 中断允许位
1 = 允许
0 = 禁止

bit 1　**CCP1IE：**ECCP1 中断允许位
1 = 允许
0 = 禁止

bit 0　**未实现：**读为 0

寄存器 10-12：　　PIE4：外设中断允许寄存器 4

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE
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　**TMR4IE：**TMR4 溢出中断允许位
1 = 允许中断
0 = 禁止中断

bit 6　**EEIE：**数据 EEDATA/ 闪存写操作中断允许位
1 = 允许中断
0 = 禁止中断

bit 5　**CMP2IE：**CMP2 中断允许位
1 = 允许中断
0 = 禁止中断

bit 4　**CMP1IE：**CMP1 中断允许位
1 = 允许中断
0 = 禁止中断

bit 3　**未实现：**读为 0

bit 2　**CCP5IE：**CCP5 中断允许位
1 = 允许中断
0 = 禁止中断

bit 1　**CCP4IE：**CCP4 中断允许位
1 = 允许中断
0 = 禁止中断

bit 0　**CCP3IE：**CCP3 中断允许位
1 = 允许中断
0 = 禁止中断

寄存器 10-13： PIE5：外设中断允许寄存器 5

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	TXB0IE	RXB1IE	RXB0IE/ FIFOFIE
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**：收到无效报文中断允许位
1 = 允许中断
0 = 禁止中断

bit 6 **WAKIE**：总线唤醒活动中断允许位
1 = 允许中断
0 = 禁止中断

bit 5 **ERRIE**：错误中断允许位 （COMSTAT 寄存器中的多个中断源）
1 = 允许中断
0 = 禁止中断

bit 4 **TXB2IE**：发送缓冲区 2 中断允许位
1 = 允许中断
0 = 禁止中断

bit 3 **TXB1IE**：发送缓冲区 1 中断允许位
1 = 允许中断
0 = 禁止中断

bit 2 **TXB0IE**：发送缓冲区 0 中断允许位
1 = 允许中断
0 = 禁止中断

bit 1 **RXB1IE**：接收缓冲区 1 中断允许位
1 = 允许中断
0 = 禁止中断

bit 0 位操作取决于选定的模式：
模式 0：
RXB0IE：接收缓冲区 0 中断允许位
1 = 允许中断
0 = 禁止中断
模式 1：
未实现：读为 0
模式 2：
FIFOFIE：FIFO 满中断允许位
1 = 允许中断
0 = 禁止中断

10.4 IPR 寄存器

IPR 寄存器包含各外设中断的优先级位。根据外设中断源的数量，有 6 个外设中断优先级寄存器（IPR1 至 IPR6）。使用优先级位时，要求将中断优先级使能（IPEN）位 （RCON<7>）置 1。

寄存器 10-14： IPR1：外设中断优先级寄存器 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
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 **PSPIP**：并行从端口读 / 写中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **ADIP**：A/D 转换器中断优先级位
1 = 高优先级
0 = 低优先级

bit 5 **RC1IP**：EUSART 接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **TX1IP**：EUSART 发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **SSPIP**：主同步串口中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **TMR1GIP**：Timer1 门控中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **TMR2IP**：TMR2 与 PR2 匹配中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **TMR1IP**：TMR1 溢出中断优先级位
1 = 高优先级
0 = 低优先级

寄存器 10-15：　　IPR2：外设中断优先级寄存器 2

R/W-1	U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1
OSCFIP	—	—	—	BCLIP	HLVDIP	TMR3IP	TMR3GIP
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　**OSCFIP：**振荡器故障中断优先级位
1 = 高优先级
0 = 低优先级

bit 6-4　**未实现：**读为 0

bit 3　**BCLIP：**总线冲突中断优先级位
1 = 高优先级
0 = 低优先级

bit 2　**HLVDIP：**高 / 低压检测中断优先级位
1 = 高优先级
0 = 低优先级

bit 1　**TMR3IP：**TMR3 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 0　**TMR3GIP：**TMR3 门控中断优先级位
1 = 高优先级
0 = 低优先级

寄存器 10-16：　　IPR3：外设中断优先级寄存器 3

U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	U-0
—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
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　**未实现：**读为 0

bit 5　**RC2IP：**EUSART 接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 4　**TX2IP：**EUSART 发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 3　**CTMUIP：**CTMU 中断优先级位
1 = 高优先级
0 = 低优先级

bit 2　**CCP2IP：**CCP2 中断优先级位
1 = 高优先级
0 = 低优先级

bit 1　**CCP1IP：**ECCP1 中断优先级位
1 = 高优先级
0 = 低优先级

bit 0　**未实现：**读为 0

寄存器 10-17：　　IPR4：外设中断优先级寄存器 4

R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	R/W-1	R/W-1
TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
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　**TMR4IP**：TMR4 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 6　**EEIP**：EE 中断优先级位
1 = 高优先级
0 = 低优先级

bit 5　**CMP2IP**：CMP2 中断优先级位
1 = 高优先级
0 = 低优先级

bit 4　**CMP1IP**：CMP1 中断优先级位
1 = 高优先级
0 = 低优先级

bit 3　**未实现**：读为 0

bit 2　**CCP5IP**：CCP5 中断优先级位
1 = 高优先级
0 = 低优先级

bit 1　**CCP4IP**：CCP4 中断优先级位
1 = 高优先级
0 = 低优先级

bit 0　**CCP3IP**：CCP3 中断优先级位
1 = 高优先级
0 = 低优先级

寄存器 10-18： IPR5：外设中断优先级寄存器 5

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	TXB0IP	RXB1IP	RXB0IP/ FIFOFIE
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**：收到无效报文中断优先级位
1 = 高优先级
0 = 低优先级

bit 6 **WAKIP**：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 总线错误中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **TXB2IP**：发送缓冲区 2 中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **TXB1IP**：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**：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**：接收缓冲区 1 中断优先级位
模式 0：
1 = 接收缓冲区 1 具有高优先级
0 = 接收缓冲区 1 具有低优先级
模式 1 和 2：
1 = 接收到的报文具有高优先级
0 = 接收到的报文具有低优先级

bit 0 **RXB0IP/FIFOFIP**：接收缓冲区 0 中断优先级位
模式 0：
1 = 接收缓冲区 0 具有高优先级
0 = 接收缓冲区 0 具有低优先级
模式 1：
未实现：读为 0
模式 2：
FIFOFIE：FIFO 满中断优先级位
1 = 高优先级
0 = 低优先级

10.5 RCON 寄存器

RCON寄存器中包含的标志位可用来确定器件上次复位或者从空闲模式或休眠模式唤醒的原因。RCON 还包含一个可使能中断优先级的 IPEN 位。

寄存器 10-19： RCON：复位控制寄存器

R/W-0	R/W-1	R/W-1	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
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 **IPEN**：中断优先级使能位
1 = 使能中断优先级
0 = 禁止中断优先级 （PIC16CXXX 兼容模式）

bit 6 **SBOREN**：软件 BOR 使能位
位操作的详细信息，请参见寄存器 5-1。

bit 5 **$\overline{CM}$**：配置不匹配标志位
1 = 未发生配置不匹配复位
0 = 发生了配置不匹配复位 （随后必须用软件置 1）

bit 4 **$\overline{RI}$**：RESET 指令标志位
位操作的详细信息，请参见寄存器 5-1。

bit 3 **$\overline{TO}$**：看门狗定时器超时标志位
位操作的详细信息，请参见寄存器 5-1。

bit 2 **$\overline{PD}$**：掉电检测标志位
位操作的详细信息，请参见寄存器 5-1。

bit 1 **$\overline{POR}$**：上电复位状态位
位操作的详细信息，请参见寄存器 5-1。

bit 0 **$\overline{BOR}$**：欠压复位状态位
位操作的详细信息，请参见寄存器 5-1。

10.6 INTx 引脚中断

RB0/INT0、RB1/INT1、RB2/INT2 和 RB3/INT3 引脚上的外部中断都是边沿触发的。如果 INTCON2 寄存器中相应的 INTEDGx 位被置 1（= 1），则由上升沿触发中断。如果该位被清零，则由下降沿触发中断。

当 RBx/INTx 引脚上出现一个有效边沿时，相应的标志位 INTxIF 被置 1。通过清零相应的中断允许位 INTxIE，可禁止该中断。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 （INTxIF）清零。

如果 INTxIE 位在进入功耗管理模式前被置 1，则所有的外部中断（INT0、INT1、INT2 和 INT3）均能将处理器从功耗管理模式唤醒。如果全局中断允许位（GIE）被置 1，则处理器将在被唤醒之后跳转到中断向量处执行程序。

INT1、 INT2 和 INT3 的中断优先级由中断优先级位 INT1IP（INTCON3<6>）、INT2IP（INTCON3<7>）和 INT3IP （INTCON2<1>）的值决定。

没有与 INT0 相关的优先级位。INT0 始终是一个高优先级的中断源。

10.7 TMR0 中断

在 8 位模式（默认模式）下，TMR0 寄存器的溢出（FFh → 00h）会将 TMR0IF 标志位置 1。在 16 位模式下，TMR0H:TMR0L 寄存器对的溢出（FFFFh → 0000h）会将 TMR0IF 标志位置 1。

可以通过置 1/ 清零中断允许位 TMR0IE（INTCON<5>）来允许 / 禁止该中断。Timer0 的中断优先级由中断优先级位 TMR0IP （INTCON2<2>）的值决定。关于 Timer0 模块的更多详细信息，请参见**第 13.0 节 “Timer0 模块 ”**。

10.8 PORTB 电平变化中断

PORTB<7:4> 上的输入电平变化会将标志位 RBIF（INTCON<0>）置 1。可以通过置 1/ 清零中断允许位 RBIE（INTCON<3>）来允许 / 禁止该中断，可以通过 IOCB 寄存器中的相应位来使能 / 禁止各个引脚。

PORTB 电平变化中断的优先级由中断优先级位 RBIP（INTCON2<0>）的值决定。

寄存器 10-20： IOCB：电平变化中断 PORTB 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
IOCB7(1)	IOCB6(1)	IOCB5(1)	IOCB4(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 7-4 **IOCB<7:4>：**电平变化中断 PORTB 控制位 (1)
1 = 允许电平变化中断
0 = 禁止电平变化中断

bit 3-0 **未实现：**读为 0

注 1： 电平变化中断还要求将 INTCON 寄存器的 RBIE 位置 1。

10.9 中断现场保护

在中断期间， PC 的返回地址被保存在堆栈中。另外，WREG、 STATUS 和 BSR 寄存器的值被压入快速返回堆栈。

如果未使用从中断快速返回功能（见**第 6.3 节 “ 数据存储器构成 ”**），那么用户可能需要在进入中断服务程序（ISR）前，保存 WREG、 STATUS 和 BSR 寄存器的值。根据用户的具体应用，还可能需要保存其他寄存器的值。

例 10-1 在执行中断服务程序期间，保存并恢复 WREG、STATUS 和 BSR 寄存器的值。

例 10-1： 将 STATUS、 WREG 和 BSR 寄存器的值保存在 RAM 中

```
MOVWF   W_TEMP                  ; W_TEMP is in virtual bank
MOVFF   STATUS, STATUS_TEMP     ; STATUS_TEMP located anywhere
MOVFF   BSR, BSR_TEMP           ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF   BSR_TEMP, BSR           ; Restore BSR
MOVF    W_TEMP, W               ; Restore WREG
MOVFF   STATUS_TEMP, STATUS     ; Restore STATUS
```

表 10-1： 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
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
PIR1	PSPIP	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIR2	OSCFIF	—	—	—	BCLIF	HLVDIF	TMR3IF	TMR3GIF
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIR5	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF	TXB0IF	RXB1IF	RXB0IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
PIE2	OSCFIE	—	—	—	BCLIE	HLVDIE	TMR3IE	TMR3GIE
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
PIE4	TMR4IE	EEIE	CCP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE
PIE5	IRXIE	WAKIE	ERRIE	TXB2IE	TXB1IE	TXB0IE	RXB1IE	RXB0IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
IPR2	OSCFIP	—	—	—	BCLIP	HLVDIP	TMR3IP	TMR3GIP
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
IPR5	IRXIP	WAKIP	ERRIP	TXB2IP	TXB1IP	TXB0IP	RXB1IP	RXB0IP
RCON	IPEN	SBOREN	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$

图注： 中断不使用阴影单元。

注：

11.0 I/O 端口

根据选定的器件和使能的功能，最多有 7 个端口可供使用。I/O 端口的一些引脚与器件上外设功能的一个备用功能复用。通常而言，当某个外设使能时，其相关引脚可能不能用作通用 I/O 引脚。

每个端口都有 3 个存储器映射的寄存器与其操作相关：

- TRIS 寄存器 （数据方向寄存器）
- PORT 寄存器 （读取器件引脚的电平）
- LAT 寄存器 （输出锁存寄存器）

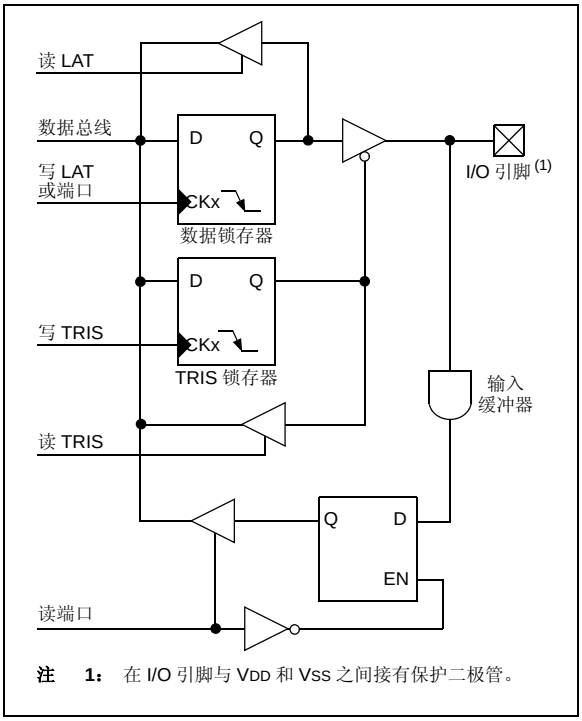
读 PORT 寄存器将读出引脚的当前状态，而写 PORT 寄存器则是将数据写入输出锁存 （LAT）寄存器。

将 TRIS 某位置 1（= 1）时，会将相应的端口引脚设为输入（使相应的输出驱动器呈高阻态）。将 TRIS 某位清零（= 0）时，会将相应的端口引脚设为输出 （即，将相应 LAT 位的内容输出到选中引脚）。

输出锁存器 （LAT 寄存器）对 I/O 引脚驱动值进行读 - 修改 - 写操作时非常有用。对 LAT 寄存器执行读 - 修改 - 写操作将读写 PORT 寄存器的锁存输出值。

图 11-1 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。

图 11-1： 通用 I/O 端口的工作原理



注 1： 在 I/O 引脚与 VDD 和 VSS 之间接有保护二极管。

11.1 I/O 端口引脚功能

在开发应用程序时，必须考虑到端口引脚的能力。某些引脚上的输出驱动能力比其他引脚要高。同样，某些引脚可以承受高于 VDD 的输入电压。

所有数字端口均可承受 5.5V 的输入电压。模拟端口在内部具有钳位二极管，可承受相同的电压。

11.1.1 引脚输出驱动能力

用作数字 I/O 时，满足各种应用需求的引脚组的输出引脚驱动能力是不同的。通常，按驱动能力可划分为两类输出引脚。

- 设计用于驱动较高电流负载 （如 LED）：
  - PORTA – PORTB
  - PORTC
- 驱动电压较低、但能够驱动具有高输入阻抗的一般数字电路负载。能够驱动 LED，但仅仅是电流要求较低的那些 LED：
  - PORTD(1) – PORTE(1)
  - PORTF(2) – PORTG(2)

注 1： 这些端口在 28 引脚器件上不存在。
2： 这些端口在 28 引脚或 40/44 引脚器件上不存在。

更多详细信息，请参见**第 31.0 节 “ 电气特性 ”** 中的 “ 绝对最大值 ”。

11.1.2 上拉配置

5 个 I/O 端口（PORTB、PORTD、PORTE、PORTF 和 PORTG）的所有引脚均实现有可配置弱上拉。这些内部上拉可使悬空数字输入信号上拉至一固定的电平而无需使用外部电阻。

可通过各个特定位使能每个端口的上拉功能：PORTB 的 RBPU（INTCON2<7>），以及其他端口的 RDPU、REPU、 RFPU 和 RGPU （PADCFG1<7:4>）。

此外，PORTB 上拉电阻可以使用 WPUB 寄存器单独使能。寄存器中的每个位对应于 PORTB 上的一个位。

寄存器 11-1： PADCFG1：焊盘配置寄存器

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0
RDPU(1)	REPU(1)	RFPU(2)	RGPU(2)	—	—	—	CTMUDS
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 **RDPU**：PORTD 上拉使能位 (1)
1 = 根据各端口锁存值使能 PORTD 上拉电阻
0 = 禁止所有 PORTD 上拉电阻

bit 6 **REPU**：PORTE 上拉使能位 (1)
1 = 根据各端口锁存值使能 PORTE 上拉电阻
0 = 禁止所有 PORTE 上拉电阻

bit 5 **RFPU**：PORTF 上拉使能位 (2)
1 = 根据各端口锁存值使能 PORTF 上拉电阻
0 = 禁止所有 PORTF 上拉电阻

bit 4 **RGPU**：PORTG 上拉使能位 (2)
1 = 根据各端口锁存值使能 PORTG 上拉电阻
0 = 禁止所有 PORTG 上拉电阻

bit 3-1 **未实现**：读为 0

bit 0 **CTMUDS**：CTMU 比较器数据选择位
1 = 使用外部比较器 （以及引脚 CTDIN 上的输出）来进行 CTMU 比较
0 = 使用内部比较器 （CMP2）来进行 CTMU 比较

注 1： 在 28 引脚器件上未实现。
2： 在 40 引脚器件上未实现。

寄存器 11-2： WPUB：弱上拉 PORTB 使能寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0
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 **WPUB<7:0>**：弱上拉使能寄存器位
1 = 当 $\overline{\text{RBPU}}$ = 0 且相应 PORTB 引脚为输入时，使能引脚上的上拉功能
0 = 禁止相应 PORTB 引脚上的上拉功能

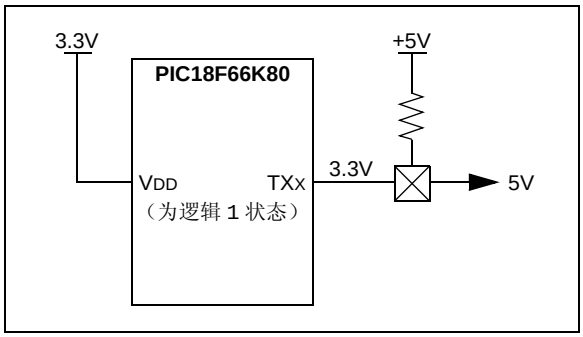
11.1.3 漏极开路输出

几个外设的输出引脚还配备了一个可配置的漏极开路输出选项。这使外设可以与工作在较高电压下的外部数字逻辑通信，而无需使用电平转换器。

漏极开路选项在与 USART、MSSP 模块（在 SPI 模式下）和 CCP 模块的数据和时钟输出相关的端口引脚上实现。通过将 ODCON 寄存器中的漏极开路控制位置 1，可以有选择地使能该选项。

当需要漏极开路选项时，输出引脚也必须通过用户提供的外部上拉电阻连接到较高电压，最高为 5V（图 11-2）。当输出数字逻辑高电平信号时，它被上拉至较高电压。

图 11-2：　使用漏极开路输出（以 USART 为例）



寄存器 11-3：　ODCON：外设漏极开路控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD
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　**SSPOD**：SPI 漏极开路输出使能位
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 6　**CCP5OD**：CCP5 漏极开路输出使能位
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 5　**CCP4OD**：CCP4 漏极开路输出使能位
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 4　**CCP3OD**：CCP3 漏极开路输出使能位
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 3　**CCP2OD**：CCP2 漏极开路输出使能位
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 2　**CCP1OD**：CCP1 漏极开路输出使能位
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 1　**U2OD**：UART2 漏极开路输出使能位
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 0　**U1OD**：UART1 漏极开路输出使能位
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

11.1.4 模拟和数字端口

许多端口都支持模拟和数字功能复用，为硬件设计人员带来了极大的灵活性。PIC18F66K80 系列器件可以将任意模拟引脚设置为模拟或数字模式，具体取决于应用的需求。端口的模拟 / 数字功能由两个寄存器控制：ANCON0 和 ANCON1。

将这些寄存器置 1 可以使相应引脚成为模拟引脚，将寄存器清零会使端口成为数字端口。关于这些寄存器的详细信息，请参见**第 23.0 节 “12 位模数转换器（A/D）模块 ”**。

11.1.5 端口压摆率

可对每个端口的输出压摆率进行编程，以选择标准变化速率或降低的变化速率（标准变化速率的 10%，最大程度降低 EMI）。对于所有端口，默认压摆率是降低的变化速率。

寄存器 11-4： SLRCON：压摆率控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	SLRG(1)	SLRF(1)	SLRE(2)	SLRD(2)	SLRC(2)	SLRB	SLRA
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 **SLRG：**PORTG 压摆率控制位 (1)
1 = PORTG 上所有输出引脚的压摆率为标准速率的 0.1 倍
0 = PORTG 上所有输出引脚的压摆率为标准速率

bit 5 **SLRF：**PORTF 压摆率控制位 (1)
1 = PORTF 上所有输出引脚的压摆率为标准速率的 0.1 倍
0 = PORTF 上所有输出引脚的压摆率为标准速率

bit 4 **SLRE：**PORTE 压摆率控制位 (2)
1 = PORTE 上所有输出引脚的压摆率为标准速率的 0.1 倍
0 = PORTE 上所有输出引脚的压摆率为标准速率

bit 3 **SLRD：**PORTD 压摆率控制位 (2)
1 = PORTD 上所有输出引脚的压摆率为标准速率的 0.1 倍
0 = PORTD 上所有输出引脚的压摆率为标准速率

bit 2 **SLRC：**PORTC 压摆率控制位 (2)
1 = PORTC 上所有输出引脚的压摆率为标准速率的 0.1 倍
0 = PORTC 上所有输出引脚的压摆率为标准速率

bit 1 **SLRB：**PORTB 压摆率控制位
1 = PORTB 上所有输出引脚的压摆率为标准速率的 0.1 倍
0 = PORTB 上所有输出引脚的压摆率为标准速率

bit 0 **SLRA：**PORTA 压摆率控制位
1 = PORTA 上所有输出引脚的压摆率为标准速率的 0.1 倍
0 = PORTA 上所有输出引脚的压摆率为标准速率

注 1： 在 28 引脚和 40/44 引脚器件上未实现，读为 0。
2： 在 28 引脚器件上未实现，读为 0。

11.2 PORTA、 TRISA 和 LATA 寄存器

PORTA 是一个 7 位宽的双向端口。对应的数据方向和输出锁存寄存器是 TRISA 和 LATA。

RA5 和 RA<3:0> 与 A/D 转换器的模拟输入复用。

通过将 ANCON1 寄存器中的 ANSEL 控制位清零或置1，可将模拟输入选作A/D转换器输入。相应的TRISA位控制这些引脚的方向，即使它们被用作模拟输入。当引脚用作模拟输入时，用户必须确保 TRISA 寄存器中的相应位保持置 1。

注：	RA5 和 RA<3:0> 在任何复位时被配置为模拟输入并读为 0。

OSC2/CLKO/RA6 和 OSC1/CLKI/RA7 通常用作外部（主）振荡器电路（HS 振荡器模式），或外部时钟输入和输出 （EC 振荡器模式）的外部电路连接引脚。在这些情况下， RA6 和 RA7 不能用作数字 I/O，并且其相应的 TRIS 和 LAT 位读为 0。当器件被配置为使用 HF-INTOSC、MF-INTOSC 或 LF-INTOSC 作为默认振荡器模式时， RA6 和 RA7 会被自动配置为数字 I/O ；振荡器和时钟输入 / 时钟输出功能被禁止。

RA5 还另外具有用于 Timer1 和 Timer3 的功能。它可以配置为 Timer1 时钟输入或 Timer3 外部时钟门控输入。

例 11-1： 初始化 PORTA

```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output latches
CLRF    LATA    ; Alternate method to
                ; clear output data latches
MOVLW   00h     ; Configure A/D
MOVWF   ANCON1  ; for digital inputs
MOVLW   0BFh    ; Value used to initialize
                ; data direction
MOVWF   TRISA   ; Set RA<7, 5:0> as inputs,
                ; RA<6> as output
```

表 11-1： PORTA 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RA0/CVREF/AN0/ ULPWU	RA0	0	O	DIG	LATA<0> 数据输出；不受模拟输入影响。
		1	I	ST	PORTA<0> 数据输入；当使能模拟输入时被禁止。
	CVREF	x	O	ANA	比较器参考电压输出。使能该功能将禁止数字 I/O。
	AN0	1	I	ANA	A/D 输入通道 0。 POR 时的默认输入配置；不影响数字输出。
	ULPWU	1	I	DIG	超低功耗唤醒输入。
RA1/AN1/C1INC	RA1	0	O	DIG	LATA<1> 数据输出；不受模拟输入影响。
		1	I	ST	PORTA<1> 数据输入；当使能模拟输入时被禁止。
	AN1	1	I	ANA	A/D 输入通道 1。 POR 时的默认输入配置；不影响数字输出。
	C1INC(1)	x	I	ANA	比较器 1 的输入 C。
RA2/VREF-/AN2/ C2INC	RA2	0	O	DIG	LATA<2> 数据输出；不受模拟输入影响。
		1	I	ST	PORTA<2> 数据输入；当使能模拟功能时被禁止。
	VREF-	1	I	ANA	A/D 和比较器低参考电压输入。
	AN2	1	I	ANA	A/D 输入通道 2。 POR 时的默认输入配置。
	C2INC(1)	x	I	ANA	比较器 2 的输入 C。
RA3/VREF+/AN3	RA3	0	O	DIG	LATA<3> 数据输出；不受模拟输入影响。
		1	I	ST	PORTA<3> 数据输入；当使能模拟输入时被禁止。
	VREF+	1	I	ANA	A/D 输入通道 3。 POR 时的默认输入配置。
	AN3	1	I	ANA	A/D 和比较器高参考电压输入。
RA5/AN4/C2INB/ HLVDIN/T1CKI/$\overline{SS}$/ CTMUI	RA5	0	O	DIG	LATA<5> 数据输出；不受模拟输入影响。
		1	I	ST	PORTA<5> 数据输入；当使能模拟输入时被禁止。
	AN4	1	I	ANA	A/D 输入通道 4。 POR 时的默认配置。
	C2INB(2)	1	I	ANA	比较器 2 的输入 B。
	HLVDIN	1	I	ANA	高 / 低压检测外部跳变点输入。
	T1CKI	x	I	ST	Timer1 时钟输入。
	$\overline{SS}$	1	I	ST	MSSP 模块的从选择输入。
	CTMUI(2)	x	I	—	用于 C2INB 比较器的 CTMU 脉冲发生器充电器的输入。
RA6/OSC2/ CLKOUT	RA6	0	O	DIG	LATA<6> 数据输出；当 FOSC2 配置位置 1 时被禁止。
		1	I	ST	PORTA<6> 数据输入；当 FOSC2 配置位置 1 时被禁止。
	OSC2	x	O	ANA	主振荡器反馈输出连接 （HS、 XT 和 LP 模式）。
	CLKOUT	x	O	DIG	系统周期时钟输出 （FOSC/4）（EC 和 INTOSC 模式）。
RA7/OSC1/CLKIN	RA7	0	O	DIG	LATA<7> 数据输出；当 FOSC2 配置位置 1 时被禁止。
		1	I	ST	PORTA<7> 数据输入；当 FOSC2 配置位置 1 时被禁止。
	OSC1	x	I	ANA	主振荡器输入连接 （HS、 XT 和 LP 模式）。
	CLKIN	x	I	ANA	主外部时钟源输入 （EC 模式）。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

注 1： 引脚分配不适用于 28 引脚器件 （PIC18F2XK80）。

2： 引脚分配仅适用于 28 引脚器件 （PIC18F2XK80）。

表 11-2：　　与 PORTA 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
PORTA	RA7(1)	RA6(1)	RA5	—	RA3	RA2	RA1	RA0
LATA	LATA7(1)	LATA6(1)	LATA5	—	LATA3	LATA2	LATA1	LATA0
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0

图注：　— = 未实现，读为 0。 PORTA 不使用阴影单元。

注　1： 这些位根据所选的振荡器模式被使能。如果未被使能为 PORTA 引脚，则它们将被禁止并读为 x。

11.3 PORTB、 TRISB 和 LATB 寄存器

PORTB 是一个 8 位宽的双向端口。对应的数据方向和输出锁存寄存器是 TRISB 和 LATB。PORTB 上的所有引脚只能用作数字功能。

例 11-2： 初始化 PORTB

```
CLRF    PORTB   ; Initialize PORTB by
                ; clearing output
                ; data latches
CLRF    LATB    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISB   ; Set RB<3:0> as inputs
                ; RB<5:4> as outputs
                ; RB<7:6> as inputs
```

PORTB 的每个引脚都具有内部弱上拉功能。通过一个控制位即可接通所有上拉。这可以通过清零 $\overline{\text{RBPU}}$ 位（INTCON2<7>）来实现。当端口引脚被配置为输出时，其弱上拉功能会自动关闭。发生上电复位时，上拉功能会被禁止。

PORTB 的 4 个引脚（RB<7:4>）具有电平变化中断功能。仅当将这些引脚配置为输入时，才可使用此中断功能。RB<7:4> 中的任何一个引脚被配置为输出时，该引脚将不再具有电平变化中断功能。

将输入引脚（RB<7:4>）上的电平与 PORTB 上次读入锁存器的旧值进行比较。对 RB<7:4> 上的 “ 不匹配 ” 输出进行逻辑或运算，产生 RB 端口电平变化中断，并将标志位 RBIF （INTCON<0<）置 1。

该中断可将器件从功耗管理模式唤醒。要在中断服务程序中清除该中断：

1. 读或写 PORTB （MOVFF (ANY),PORTB 指令除外）。
2. 等待一个指令周期 （例如执行 NOP 指令）。

   这会结束不匹配条件。
3. 清零标志位 RBIF。

不匹配条件会继续将标志位 RBIF 置 1。读 PORTB 将结束不匹配条件，并允许在一个 TCY 延时后将标志位 RBIF 清零。

建议使用电平变化中断功能实现按键唤醒操作，以及那些仅将 PORTB 用于电平变化中断功能的操作。在使用电平变化中断功能时，建议不要查询 PORTB 的状态。

RB<3:2> 引脚与 CTMU 边沿输入复用。RB5 还另外具有用于 Timer3 和 Timer1 的功能。它可以配置为 Timer3 时钟输入或 Timer1 外部时钟门控输入。

表 11-3： PORTB 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RB0/AN10/C1INA FLT0/INT0	RB0	0	O	DIG	LATB<0> 数据输出。
		1	I	ST	PORTB<0> 数据输入；当 RBPU 位清零时使能弱上拉。
	AN10	1	I	ANA	A/D 输入通道 10 和比较器 C1+ 的输入。 POR 时的默认输入配置。
	C1INA(1)	1	I	ANA	比较器 1 的输入 A。
	FLT0	x	I	ST	ECCPx 的增强型 PWM 故障输入。
	INT0	1	I	ST	外部中断 0 输入。
RB1/AN8/C1INB/ P1B/CTDIN/INT1	RB1	0	O	DIG	LATB<1> 数据输出。
		1	I	ST	PORTB<1> 数据输入；当 RBPU 位清零时使能弱上拉。
	AN8	1	I	ANA	A/D 输入通道 8 和比较器 C2+ 的输入。 POR 时的默认输入配置；不受模拟输出影响。
	C1INB(1)	1	I	ANA	比较器 1 的输入 B。
	P1B(1)	0	O	DIG	ECCP1 PWM 的输出 B。可以在增强型 PWM 关闭事件期间被配置为三态。
	CTDIN	1	I	ST	CTMU 脉冲延时输入。
	INT1	1	I	ST	外部中断 1 输入。
RB2/CANTX/C1OUT/ P1C/CTED1/INT2	RB2	0	O	DIG	LATB<2> 数据输出。
		1	I	ST	PORTB<2> 数据输入；当 RBPU 位清零时使能弱上拉。
	CANTX(2)	0	O	DIG	CAN 总线发送。
	C1OUT(1)	0	O	DIG	比较器 1 的输出；优先于端口数据。
	P1C(1)	0	O	DIG	ECCP1 PWM 的输出 C。可以在增强型 PWM 期间被配置为三态。
	CTED1	x	I	ST	CTMU 边沿 1 输入。
	INT2	1	I	ST	外部中断 2。
RB3/CANRX/ C2OUT/P1D/ CTED2/INT3	RB3	0	O	DIG	LATB<3> 数据输出。
		1	I	ST	PORTB<3> 数据输入；当 RBPU 位清零时使能弱上拉。
	CANRX(2)	1	I	ST	CAN 总线接收。
	C2OUT(1)	x	I	ST	CTMU 边沿 2 输入。
	P1D(1)	0	O	DIG	ECCP1 PWM 的输出 D。可以在增强型 PWM 期间被配置为三态。
	CTED2	x	I	ST	CTMU 边沿 2 输入。
	INT3	1	I	ST	外部中断 3 输入。
RB4/AN9/C2INA/ ECCP1/P1A/CTPLS/ KBI0	RB4	0	O	DIG	LATB<4> 数据输出。
		1	I	ST	PORTB<4> 数据输入；当 RBPU 位清零时使能弱上拉。
	AN9	1	I	ANA	A/D 输入通道 9 和比较器 C2+ 的输入。 POR 时的默认输入配置；不受模拟输出影响。
	C2INA(1)	2	I	ANA	比较器 2 的输入 A。
	ECCP1(1)	0	O	DIG	ECCP1 比较输出和 ECCP1 PWM 输出。优先于端口数据。
		1	I	ST	ECCP1 捕捉输入。
	P1A(1)	0	O	DIG	ECCP1 增强型 PWM 输出，通道 A。可以在增强型 PWM 关闭事件期间被配置为三态。优先于端口数据。
	CTPLS	x	O	DIG	CTMU 脉冲发生器输出。
	KBI0	1	I	ST	引脚电平变化中断。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

注 1： 引脚分配仅适用于 28 引脚器件 （PIC18F2XK80）。

2： 当 CANMX 配置位置 1 时， CANRX 和 CANTX 的默认引脚分配。

3： 当 T0CKMX 配置位置 1 时， T0CKI 的默认引脚分配。

4： T3CKI 的默认引脚分配（对于 28、40 和 44 引脚器件）。当 T3CKMX 清零时，T3CKI 的备用引脚分配（对于 64 引脚器件）。

表 11-3： PORTB 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RB5/T0CKI/T3CKI/CCP5/KBI1	RB5	0	O	DIG	LATB<5> 数据输出。
		1	I	ST	PORTB<5> 数据输入；当 RBPU 位清零时使能弱上拉。
	T0CKI(3)	x	I	ST	Timer0 时钟输入。
	T3CKI(4)	x	I	ST	Timer3 时钟输入。
	CCP5	0	O	DIG	CCP5 比较 /PWM 输出。优先于端口数据。
		1	I	ST	CCP5 捕捉输入。
	KBI1	1	I	ST	引脚电平变化中断。
RB6/PGC/TX2/CK2/KBI2	RB6	0	O	DIG	LATB<6> 数据输出。
		1	I	ST	PORTB<6> 数据输入；当 RBPU 位清零时使能弱上拉。
	PGC	x	I	ST	供 ICSP 和 ICD 操作使用的串行执行 （ICSP™）时钟输入。
	TX2(1)	0	O	DIG	异步串行数据输出 （EUSART 模块）；优先于端口数据。
	CK2(1)	0	O	DIG	同步串行时钟输出 （EUSART 模块）；用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入 （EUSART 模块）；用户必须将其配置为输入。
	KBI2	1	I	ST	引脚电平变化中断。
RB7/PGD/T3G/RX2/DT2/KBI3	RB7	0	O	DIG	LATB<7> 数据输出。
		1	I	ST	PORTB<7> 数据输入；当 RBPU 位清零时使能弱上拉。
	PGD	x	O	DIG	供 ICSP 和 ICD 操作使用的串行执行数据输出。
		x	I	ST	供 ICSP 和 ICD 操作使用的串行执行数据输入。
	T3G	x	I	ST	Timer3 外部时钟门控输入。
	RX2(1)	1	I	ST	异步串行接收数据输入 （EUSART 模块）。
	DT2(1)	1	O	DIG	同步串行数据输出 （AUSART 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入 （AUSART 模块）；用户必须将其配置为输入。
	KBI3	1	I	ST	引脚电平变化中断。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

注 1： 引脚分配仅适用于 28 引脚器件 （PIC18F2XK80）。
2： 当 CANMX 配置位置 1 时， CANRX 和 CANTX 的默认引脚分配。
3： 当 T0CKMX 配置位置 1 时， T0CKI 的默认引脚分配。
4： T3CKI 的默认引脚分配（对于 28、40 和 44 引脚器件）。当 T3CKMX 清零时，T3CKI 的备用引脚分配（对于 64 引脚器件）。

表 11-4： 与 PORTB 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
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD
ANCON1	—	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8

图注： PORTB 不使用阴影单元。

11.4 PORTC、 TRISC 和 LATC 寄存器

PORTC 是一个 8 位宽的双向端口。对应的数据方向和输出锁存寄存器是 TRISC 和 LATC。PORTC 引脚中只有 RC2 至 RC7 是仅用作数字功能的引脚。

PORTC 与 CCP、MSSP 和 EUSART 外设功能复用（表 11-5）。这些引脚配有施密特触发器输入缓冲器。无论何时，只要这些功能有效，CCP、SPI 和 EUSART 的引脚就可配置为漏极开路输出。通过将 ODCON 寄存器中的 SSPOD、CCPxOD 和 U1OD 控制位置 1 选择漏极开路配置。

当 RC1 引脚上的 CCP2 有效时，该引脚还可配置为漏极开路输出。通过将 CCP2OD 控制位（ODCON<3>）置 1 选择漏极开路配置。

当使能外设功能时，应小心定义每个 PORTC 引脚的 TRIS 位。有些外设会改写 TRIS 位的设置，将引脚重新定义为输出引脚或输入引脚。请查阅相应的外设章节来正确设置 TRIS 位。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

外设对引脚的改写会影响 TRISC 寄存器的内容。尽管外设可能会改写一个或多个引脚，读 TRISC 总是会返回其当前的内容。

例 11-3： 初始化 PORTC

```
CLRF    PORTC   ; Initialize PORTC by
                ; clearing output
                ; data latches
CLRF    LATC    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISC   ; Set RC<3:0> as inputs
                ; RC<5:4> as outputs
                ; RC<7:6> as inputs
```

表 11-5： PORTC 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RC0/SOSCO/ SCLKI	RC0	0	O	DIG	LATC<0> 数据输出。
		1	I	ST	PORTC<0> 数据输入。
	SOSCO	1	O	ST	SOSC 振荡器输出。
	SCLKI	1	I	ST	数字时钟输入；当禁止 SOSC 振荡器时被使能。
RC1/SOSCI	RC1	0	O	DIG	LATC<1> 数据输出。
		1	I	ST	PORTC<1> 数据输入。
	SOSCI	x	I	ANA	SOSC 振荡器输入。
RC2/T1G/ CCP2	RC2	0	O	DIG	LATC<2> 数据输出。
		1	I	ST	PORTC<2> 数据输入。
	T1G	x	I	ST	Timer1 外部时钟门控输入。
	CCP2	0	O	DIG	CCP2 比较 /PWM 输出。优先于端口数据。
		1	I	ST	CCP2 捕捉输入。
RC3/REFO/ SCL/SCK	RC3	0	O	DIG	LATC<3> 数据输出。
		1	I	ST	PORTC<3> 数据输入。
	REFO	x	O	DIG	参考输出时钟。
	SCL	0	O	DIG	I^2C™ 时钟输出 （MSSP 模块）；优先于端口数据。
		1	I	I^2C	I^2C 时钟输入 （MSSP 模块）；输入类型取决于模块设置。
	SCK	0	O	DIG	SPI 时钟输出 （MSSP 模块）；优先于端口数据。
		1	I	ST	SPI 时钟输入 （MSSP 模块）。
RC4/SDA/SDI	RC4	0	O	DIG	LATC<4> 数据输出。
		1	I	ST	PORTC<4> 数据输入。
	SDA	1	O	DIG	I^2C 数据输出 （MSSP 模块）；优先于端口数据。
		1	I	I^2C	I^2C 数据输入 （MSSP 模块）；输入类型取决于模块设置。
	SDI	1	I	ST	SPI 数据输入 （MSSP 模块）。
RC5/SDO	RC5	0	O	DIG	LATC<5> 数据输出。
		1	I	ST	PORTC<5> 数据输入。
	SDO	0	O	DIG	SPI 数据输出 （MSSP 模块）。
RC6/CANTX/ TX1/CK1/ CCP3	RC6	0	O	DIG	LATC<6> 数据输出。
		1	I	ST	PORTC<6> 数据输入。
	CANTX(1)	0	O	DIG	CAN 总线发送。
	TX1(1)	0	O	DIG	异步串行数据输出 （EUSART 模块）；优先于端口数据。
	CK1(1)	0	O	DIG	同步串行时钟输出 （EUSART 模块）；用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入 （EUSART 模块）；用户必须将其配置为输入。
	CCP3	0	O	DIG	CCP3 比较 /PWM 输出。优先于端口数据。
		1	I	ST	CCP3 捕捉输入。

图注： O = 输出，I = 输入，I^2C = I^2C/SMBus， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

注 1： 针对 28、 40 和 44 引脚器件 （PIC18F2XK80 和 PIC18F4XK80）的引脚分配。

2： 当 CANMX 配置位置 1 时， 28、 40 和 44 引脚器件 （PIC18F4XK80）上 CANRX 和 CANTX 的备用引脚分配。

表 11-5： PORTC 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RC7/CANRX/ RX1/DT1/ CCP4	RC7	0	O	DIG	LATC<7> 数据输出。
		1	I	ST	PORTC<7> 数据输入。
	CANRX(1)	1	I	ST	CAN 总线接收。
	RX1(1)	1	I	ST	异步串行接收数据输入 （EUSART 模块）。
	DT1(1)	1	O	DIG	同步串行数据输出 （EUSART 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入 （EUSART 模块）；用户必须将其配置为输入。
	CCP4	0	O	DIG	CCP4 比较 /PWM 输出。优先于端口数据。
		1	I	ST	CCP4 捕捉输入。

图注： O = 输出，I = 输入，I^2C = I^2C/SMBus， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

注 1： 针对 28、 40 和 44 引脚器件 （PIC18F2XK80 和 PIC18F4XK80）的引脚分配。

2： 当 CANMX 配置位置 1 时， 28、 40 和 44 引脚器件 （PIC18F4XK80）上 CANRX 和 CANTX 的备用引脚分配。

表 11-6： 与 PORTC 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
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
LATC	LATC7	LATBC6	LATC5	LATCB4	LATC3	LATC2	LATC1	LATC0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

图注： PORTC 不使用阴影单元。

11.5 PORTD、 TRISD 和 LATD 寄存器

PORTD 是一个 8 位宽的双向端口。对应的数据方向和输出锁存寄存器是 TRISD 和 LATD。

注：	PORTD 在 28 引脚器件上不可用。

PORTD 上的所有引脚都配有施密特触发器输入缓冲器。每个引脚都可被单独配置为输入或输出。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

PORTD 的每个引脚都具有内部弱上拉功能。可通过单个控制位关闭所有上拉。这可以通过清零 RDPU 位（PADCFG1<7>）来实现。当端口引脚被配置为输出时，其弱上拉功能会自动关闭。发生所有器件复位时，上拉功能会被禁止。

还可通过将控制位 PSPMODE（PSPCON<4>）置 1，将 PORTD 配置为 8 位宽的微处理器端口（并行从端口）。在该模式下，输入缓冲器是 ST。更多信息，请参见**第 11.9 节 “ 并行从端口 ”**。

RD3 具有 CTMU 功能。

例 11-4： 初始化 PORTD

```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

表 11-7： PORTD 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RD0/C1INA/ PSP0	RD0	0	O	DIG	LATD<0> 数据输出。
		1	I	ST	PORTD<0> 数据输入。
	C1INA	1	I	ANA	比较器 1 的输入 A。
	PSP0	x	I/O	ST	并行从端口数据。
RD1/C1INB/ PSP1	RD1(1)	0	O	DIG	LATD<1> 数据输出。
		1	I	ST	PORTD<1> 数据输入。
	C1INB(1)	1	I	ANA	比较器 1 的输入 B。
	PSP1(1)	x	I/O	ST	并行从端口数据。
RD2/C2INA/ PSP2	RD2	0	O	DIG	LATD<2> 数据输出。
		1	I	ST	PORTD<2> 数据输入。
	C2INA	1	I	ANA	比较器 2 的输入 A。
	PSP2	x	I/O	ST	并行从端口数据。
RD3/C2INB/ CTMUI/PSP3	RD3	0	O	DIG	LATD<3> 数据输出。
		1	I	ST	PORTD<3> 数据输入。
	C2INB	1	I	ANA	比较器 2 的输入 B。
	CTMUI	x	I	—	用于 C2INB 比较器的 CTMU 脉冲发生器充电器的输入。
	PSP3	x	I/O	ST	并行从端口数据。
RD4/ECCP1/ P1A/PSP4	RD4	0	O	DIG	LATD<4> 数据输出。
		1	I	ST	PORTD<4> 数据输入。
	ECCP1	0	O	DIG	ECCP1 比较输出和 ECCP1 PWM 输出。优先于端口数据。
		1	I	ST	ECCP1 捕捉输入。
	P1A	0	O	DIG	ECCP1 增强型 PWM 输出，通道 A。可以在增强型 PWM 关闭事件期间被配置为三态。优先于端口数据。
	PSP4	x	I/O	ST	并行从端口数据。
RD5/P1B/PSP5	RD5	0	O	DIG	LATD<5> 数据输出。
		1	I	ST	PORTD<5> 数据输入。
	P1B	0	O	DIG	ECCP1 增强型 PWM 输出，通道 B。可以在增强型 PWM 关闭事件期间被配置为三态。优先于端口数据。
	PSP5	x	I/O	ST	并行从端口数据。
RD6/TX2/CK2 P1C/PSP6	RD6	0	O	DIG	LATD<6> 数据输出。
		1	I	ST	PORTD<6> 数据输入。
	TX2(1)	0	O	DIG	异步串行数据输出（EUSART 模块）；优先于端口数据。
	CK2(1)	0	O	DIG	同步串行时钟输出（EUSART 模块）；用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入（EUSART 模块）；用户必须将其配置为输入。
	P1C	0	O	DIG	ECCP1 增强型 PWM 输出，通道 C。可以在增强型 PWM 期间被配置为三态。
	PSP6	x	I/O	ST	并行从端口数据。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位（TRIS 位不影响端口方向或在此可忽略）

注 1： 针对 40 和 44 引脚器件（PIC18F4XK80）的引脚分配。

表 11-7： PORTD 功能（续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RD7/RX2/DT2/ P1D/PSP7	RD7	0	O	DIG	LATD<7> 数据输出。
		1	I	ST	PORTD<7> 数据输入。
	RX2(1)	1	I	ST	异步串行接收数据输入（EUSART 模块）。
	DT2(1)	1	O	DIG	同步串行数据输出（EUSART 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入（EUSART 模块）；用户必须将其配置为输入。
	P1D	0	O	DIG	ECCP1 增强型 PWM 输出，通道 D。可以在增强型 PWM 期间被配置为三态。
	PSP7	x	I/O	ST	并行从端口数据。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位（TRIS 位不影响端口方向或在此可忽略）

注 1： 针对 40 和 44 引脚器件（PIC18F4XK80）的引脚分配。

表 11-8： 与 PORTD 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
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0
PADCFG1	RDPU(1)	REPU(1)	RFPU(2)	RGPU(2)	—	—	—	CTMUDS
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD
ANCON1	—	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8

图注： PORTD 不使用阴影单元。

注 1： 在 28 引脚器件上未实现，读为 0。

2： 在 28/40/44 引脚器件上未实现，读为 0。

11.6 PORTE、 TRISE 和 LATE 寄存器

PORTE 是一个 7 位宽的双向端口。对应的数据方向和输出锁存寄存器是 TRISE 和 LATE。

注：	PORTE 在 28 引脚器件上不可用。

PORTE 上的所有引脚都配有施密特触发器输入缓冲器。每个引脚都可被单独配置为输入或输出。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

PORTE的每个引脚都具有内部弱上拉功能。可通过单个控制位关闭所有上拉。这可以通过清零REPU位（PADCFG1<6>）来实现。当端口引脚被配置为输出时，其弱上拉功能会自动关闭。发生任何器件复位时，上拉功能会被禁止。

PORTE 还能与并行从端口地址线复用。RE1 和 RE0 与并行从端口 （PSP）控制信号 $\overline{WR}$ 和 $\overline{RD}$ 复用。

例 11-5： 初始化 PORTE

```
CLRF    PORTE   ; Initialize PORTE by
                ; clearing output
                ; data latches
CLRF    LATE    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   03h     ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISE   ; Set RE<1:0> as inputs
                ; RE<7:2> as outputs
```

表 11-9： PORTE 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE0/AN5/$\overline{RD}$	RE0	0	O	DIG	LATE<0> 数据输出。
		1	I	ST	PORTE<0> 数据输入。
	AN5	1	I	ANA	A/D 输入通道 5。 POR 时的默认输入配置；不影响数字输出。
	$\overline{RD}$	x	O	DIG	并行从端口读选通引脚。
		x	I	ST	并行从端口读操作引脚。
RE1/AN6/ C1OUT/$\overline{WR}$	RE1	0	O	DIG	LATE<1> 数据输出。
		1	I	ST	PORTE<1> 数据输入。
	AN6	1	I	ANA	A/D 输入通道 5。 POR 时的默认输入配置；不影响数字输出。
	C1OUT	0	O	DIG	比较器 1 的输出；优先于端口数据。
	$\overline{WR}$	x	O	DIG	并行从端口写选通引脚。
		x	I	ST	并行从端口写操作引脚。
RE2/AN7/ C2OUT/$\overline{CS}$	RE2	0	O	DIG	LATE<2> 数据输出。
		1	I	ST	PORTE<2> 数据输入。
	AN7	1	I	ANA	A/D 输入通道 7。 POR 时的默认输入配置；不影响数字输出。
	C2OUT	0	O	DIG	比较器 2 的输出；优先于端口数据。
	$\overline{CS}$	x	I	ST	并行从端口片选。
RE3	RE3	1	I	ST	PORT<3> 数据输入。
RE4/CANRX	RE4(1)	0	O	DIG	LATE<4> 数据输出。
		1	I	ST	PORTE<4> 数据输入。
	CANRX(1,2)	1	I	ST	CAN 总线接收。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

注 1： 不适用于 40 和 44 引脚器件 （PIC18F4XK80）。

2： 当 CANMX 配置位清零时， 64 引脚器件 （PIC18F6XK80）上 CANRX 和 CANTX 的备用引脚分配。

表 11-9： PORTE 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE5/CANTX	RE5(1)	0	O	DIG	LATE<5> 数据输出。
		1	I	ST	PORTE<5> 数据输入。
	CANTX(1,2)	0	O	DIG	CAN 总线发送。
RE6/RX2/DT2	RE6(1)	0	O	DIG	LATE<6> 数据输出。
		1	I	ST	PORTE<6> 数据输入。
	RX2(1)	1	I	ST	异步串行接收数据输入 （EUSART 模块）。
	DT2(1)	1	O	DIG	同步串行数据输出 （EUSART 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入 （EUSART 模块）；用户必须将其配置为输入。
RE7/TX2/CK2	RE7(1)	0	O	DIG	LATE<7> 数据输出。
		1	I	ST	PORTE<7> 数据输入。
	TX2(1)	0	O	DIG	异步串行数据输出 （EUSART 模块）；优先于端口数据。
	CK2(1)	0	O	DIG	同步串行时钟输出 （EUSART 模块）；用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入 （EUSART 模块）；用户必须将其配置为输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

注 1： 不适用于 40 和 44 引脚器件 （PIC18F4XK80）。

2： 当 CANMX 配置位清零时， 64 引脚器件 （PIC18F6XK80）上 CANRX 和 CANTX 的备用引脚分配。

表 11-10： 与 PORTE 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
PORTE	RE7(1)	RE6(1)	RE5(1)	RE4(1)	RE3	RE2	RE1	RE0
LATE	LATE7	LATE6	LATE5	LATE4	—	LATE2	LATE1	LATE0
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	—	TRISE2	TRISE1	TRISE0
PADCFG1	RDPU	REPU	RFPU(1)	RGPU(1)	—	—	—	CTMUDS
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0

图注： PORTE 不使用阴影单元。

注 1： 在 44 引脚器件上未实现，读为 0。

11.7 PORTF、 LATF 和 TRISF 寄存器

PORTF 是一个 8 位宽的双向端口。对应的数据方向和输出锁存寄存器是 TRISF 和 LATF。 PORTF 上的所有引脚都配有施密特触发器输入缓冲器。每个引脚都可被单独配置为输入或输出。

注： PORTF 仅在 64 引脚器件上可用。

PORTF 的每个引脚都具有内部弱上拉功能。可通过单个控制位关闭所有上拉。这可以通过清零 RFPU 位（PADCFG1<5>）来实现。当端口引脚被配置为输出时，其弱上拉功能会自动关闭。发生任何器件复位时，上拉功能会被禁止。

注： 发生器件复位时，RF<7:1>引脚被配置为模拟输入并读为 0。

例 11-6： 初始化 PORTF

```
CLRF    PORTF   ; Initialize PORTF by
                ; clearing output
                ; data latches
CLRF    LATF    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CEh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISF   ; Set RF3:RF1 as inputs
                ; RF5:RF4 as outputs
                ; RF7:RF6 as inputs
```

表 11-11： PORTF 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RF0/MDMIN	RF0	0	O	DIG	LATF<0> 数据输出。
		1	I	ST	PORTF<0> 数据输入。
	MDMIN	1	I	ST	调制器源输入。
RF1	RF1	0	O	DIG	LATF<1> 数据输出。
		1	I	ST	PORTF<1> 数据输入。
RF2/MDCIN1	RF2	0	O	DIG	LATF<2> 数据输出。
		1	I	ST	PORTF<2> 数据输入。
	MDCIN1	1	I	ST	调制器载波输入 1。
RF3	RF3	0	O	DIG	LATF<3> 数据输出。
		1	I	ST	PORTF<3> 数据输入。
RF4/MDCIN2	RF4	0	O	DIG	LATF<4> 数据输出。
		1	I	ST	PORTF<4> 数据输入。
	MDCIN2	1	I	ST	调制器载波输入 2。
RF5	RF5	0	O	DIG	LATF<5> 数据输出。
		1	I	ST	PORTF<5> 数据输入。
RF6/MDOUT	RF6	0	O	DIG	LATF<6> 数据输出。
		1	I	ST	PORTF<6> 数据输入。
	MDOUT	0	O	DIG	调制器输出。
RF7	RF7	0	O	DIG	LATF<7> 数据输出。
		1	I	ST	PORTF<7> 数据输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

表 11-12： 与 PORTF 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
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0
LATF	LATF7	LATF6	LATF5	LATF4	—	LATF2	LATF1	LATF0
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0
PADCFG1	RDPU	REPU	RFPU(1)	RGPU(1)	—	—	—	CTMUDS

图注： — = 未实现，读为 0。 PORTF 不使用阴影单元。

注 1： 在 28 引脚器件上未实现；读为 0。

11.8 PORTG、 TRISG 和 LATG 寄存器

PORTG 是一个 5 位宽的双向端口。对应的数据方向和输出锁存寄存器是 TRISG 和 LATG。

注：	PORTG 仅在 64 引脚器件上可用。

PORTG 与 EUSART 和 CCP、ECCP、模拟、比较器以及定时器输入功能复用（表 11-13）。当用作 I/O 时，所有 PORTG 引脚都配有施密特触发器输入缓冲器。可以使用 ODCON 配置 UART 的漏极开路功能。

PORTG 的每个引脚都具有内部弱上拉功能。可通过单个控制位关闭所有上拉。这可以通过清零 RGPU 位（PADCFG1<4>）来实现。当端口引脚被配置为输出时，其弱上拉功能会自动关闭。发生任何器件复位时，上拉功能会被禁止。

当使能外设功能时，应小心定义每个 PORTG 引脚的 TRIS 位。有些外设会改写 TRIS 位的设置，将引脚重新定义为输出引脚或输入引脚。用户应该查阅相应的外设章节来正确设置 TRIS 位。引脚改写值不装入 TRIS 寄存器中。这将允许对 TRIS 寄存器执行读 - 修改 - 写操作而无需担心外设的改写。

例 11-7： 初始化 PORTG

```
CLRF    PORTG           ; Initialize PORTG by
                        ; clearing output
                        ; data latches
CLRF    LATG            ; Alternate method
                        ; to clear output
                        ; data latches
MOVLW   04h             ; Value used to
                        ; initialize data
                        ; direction
MOVWF   TRISG           ; Set RG1:RG0 as
                        ; outputs
                        ; RG2 as input
                        ; RG4:RG3 as inputs
```

表 11-13： PORTG 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RG0/RX1/DT1	RG0	0	O	DIG	LATG<0> 数据输出。
		1	I	ST	PORTG<0> 数据输入。
	RX1	1	I	ST	异步串行接收数据输入 （EUSART 模块）。
	DT1	0	O	DIG	同步串行数据输出 （EUSART 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入 （EUSART 模块）；用户必须将其配置为输入。
RG1/CANTX	RG1	0	O	DIG	LATG<1> 数据输出。
		1	I	ST	PORTG<1> 数据输入。
	CANTX	0	O	DIG	CAN 总线发送。
RG2/T3CKI	RG2	0	O	DIG	LATG<2> 数据输出。
		1	I	ST	PORTG<2> 数据输入。
	T3CKI(2)	x	I	ST	Timer3 时钟输入。
RG3/TX1/CK1	RG3	0	O	DIG	LATG<3> 数据输出。
		1	I	ST	PORTG<3> 数据输入。
	TX1	0	O	DIG	异步串行数据输出 （EUSART 模块）；优先于端口数据。
	CK1	0	O	DIG	同步串行时钟输出 （EUSART 模块）；用户必须将其配置为输入。
		1	I	ST	同步串行时钟输入 （EUSART 模块）；用户必须将其配置为输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位 （TRIS 位不影响端口方向或在此可忽略）

注 1： 当 T0CKMX 配置位清零时， 64 引脚器件上 T0CKI 的备用引脚分配。

2： 当 T3CKMX 配置位置 1 时， 64 引脚器件上 T3CKI 的默认引脚分配。

表 11-13： PORTG 功能（续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RG4/T0CKI	RG4	0	O	DIG	LATG<4> 数据输出。
		1	I	ST	PORTG<4> 数据输入。
	T0CKI(1)	x	I	ST	Timer0 时钟输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = CMOS 输出， ST = 施密特触发器缓冲器输入，
x = 无关位（TRIS 位不影响端口方向或在此可忽略）

注 1： 当 T0CKMX 配置位清零时， 64 引脚器件上 T0CKI 的备用引脚分配。

2： 当 T3CKMX 配置位置 1 时， 64 引脚器件上 T3CKI 的默认引脚分配。

表 11-14： 与 PORTG 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
PORTG	—	—	—	RG4	RG3	RG2	RG1	RG0
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0
PADCFG1	RDPU	REPU	RFPU(1)	RGPU(1)	—	—	—	CTMUDS

图注： — = 未实现，读为 0。 PORTG 不使用阴影单元。

注 1： 在 28 引脚器件上未实现。读为 0。

11.9 并行从端口

当控制位 PSPMODE（PSPCON<4>）置 1 时，PORTD 也可用作 8 位宽的并行从端口（PSP）或微处理器端口。可从外部通过 $\overline{RD}$ 控制输入引脚（RE0/AN5/$\overline{RD}$）和 $\overline{WR}$ 控制输入引脚（RE1/AN6/C1OUT/$\overline{WR}$）对端口进行异步读写。

注：	并行从端口仅在 40/44 引脚和 64 引脚器件上可用。

PSP 可以直接与 8 位微处理器数据总线连接。外部微处理器可以读或写 PORTD 的 8 位锁存值。

将 PSPMODE 位置 1 会将端口引脚 RE0/AN5/$\overline{RD}$ 使能为 $\overline{RD}$ 输入，将 RE1/AN6/C1OUT/$\overline{WR}$ 使能为 $\overline{WR}$ 输入，并将 RE2/AN7/C2OUT/$\overline{CS}$ 使能为 $\overline{CS}$（片选）输入。要实现此功能，TRISE 寄存器中的对应数据方向位（TRISE<2:0>）必须配置为输入（= 111）。

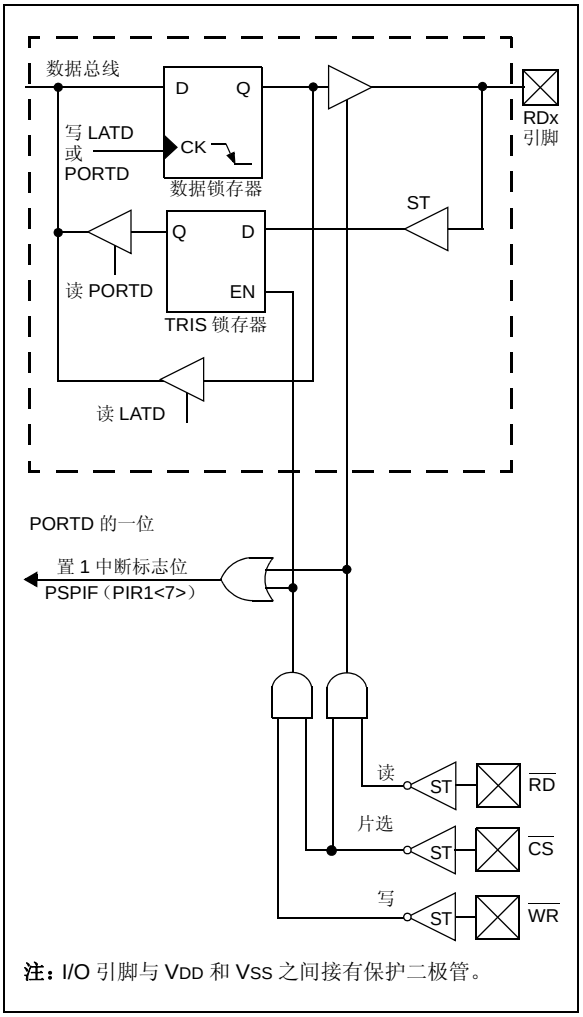
当第一次检测到 $\overline{CS}$ 和 $\overline{WR}$ 线均为低电平时发生对 PSP 的写操作，当检测到任何一根线为高电平时结束操作。当写操作结束时，PSPIF和IBF标志位（分别为PIR1<7>和 PSPCON<7>）会置 1。

当第一次检测到 $\overline{CS}$ 和 $\overline{RD}$ 线均为低电平时发生对 PSP 的读操作。PORTD 中的数据被读出且 OBF 位（PSPCON<6>）置 1。如果用户通过将新数据写入 PORTD 而试图将 OBF 置 1，该数据会立即被读出，但 OBF 位不会被置 1。

当 $\overline{CS}$ 或 $\overline{RD}$ 线被检测到高电平时，PORTD 引脚返回到输入状态且PSPIF 位被置1。用户应用程序在为 PSP 提供服务之前应该等待 PSPIF 被置 1。发生这种情况时，可以查询 IBF 和 OBF 位并进行相应的操作。

写和读模式下控制信号的时序分别如图 11-4 和图 11-5 所示。

图 11-3：　PORTD 和 PORTE 框图（并行从端口）



注： I/O 引脚与 VDD 和 VSS 之间接有保护二极管。

寄存器 11-5：　　PSPCON：并行从端口控制寄存器

R-0	R-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
IBF	OBF	IBOV	PSPMODE	—	—	—	—
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　**IBF：**输入缓冲区满状态位
1 = 已接收一个字，等待 CPU 读取
0 = 未接收到任何字

bit 6　**OBF：**输出缓冲区满状态位
1 = 输出缓冲区仍保存之前写入的字
0 = 输出缓冲区已被读取

bit 5　**IBOV：**输入缓冲区溢出检测位
1 = 之前输入的字尚未被读取时发生写操作 （必须用软件清零）
0 = 未发生溢出

bit 4　**PSPMODE：**并行从端口模式选择位
1 = 并行从端口模式
0 = 通用 I/O 模式

bit 3-0　**未实现：**读为 0

图 11-4：　　并行从端口写波形

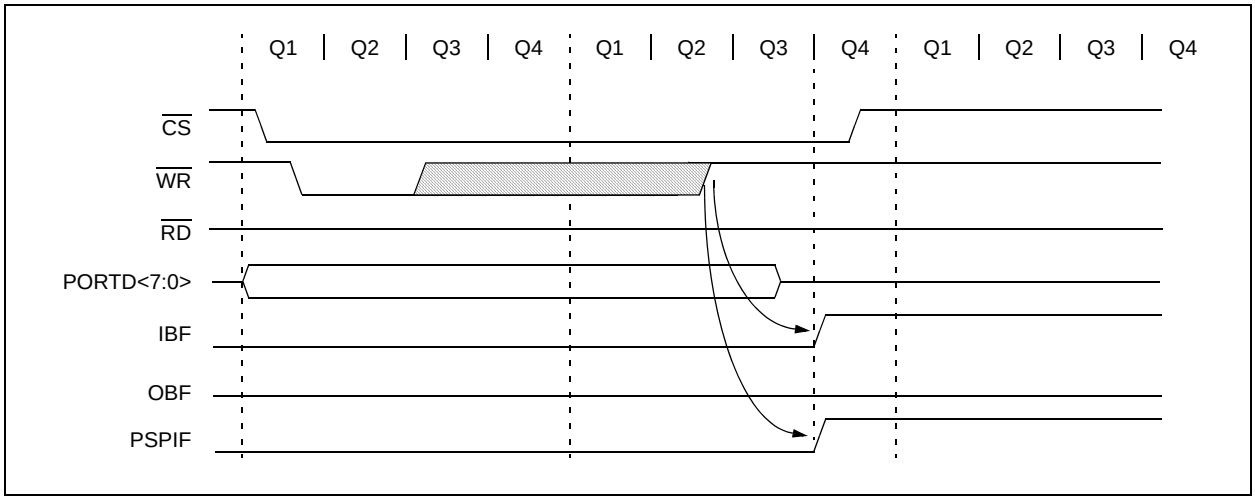


图 11-5：　　并行从端口读波形

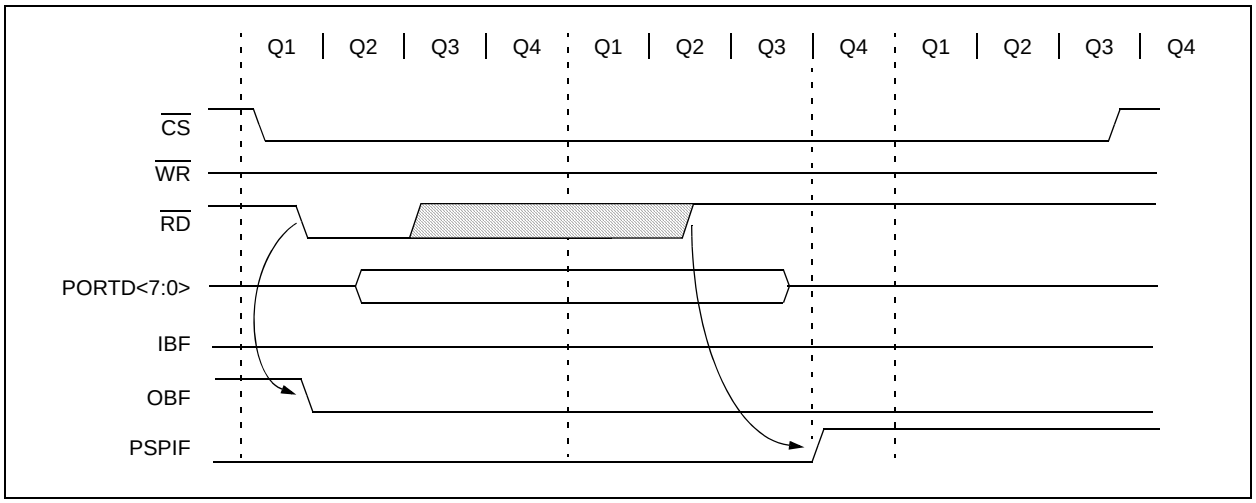


表 11-15：　　与并行从端口相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0
PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0
LATE	LATE7	LATE6	LATE5	LATE4	—	LATE2	LATE1	LATE0
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	—	TRISE2	TRISE1	TRISE0
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD

图注：　— = 未实现，读为 0。并行从端口不使用阴影单元。

12.0 数据信号调制器

注： 数据信号调制器仅在 64 引脚器件（PIC18F6XK80）上可用。

数据信号调制器（Data Signal Modulator，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，信号可以来自某个片内外设的输出，也可以通过某个输入引脚从外部提供。

调制输出信号的产生方式是：对载波和调制器信号执行逻辑与操作，然后送到 MDOUT 引脚上。

载波信号由两个不同的独立信号组成：载波高（CARH）信号和载波低（CARL）信号。在调制器（MOD）信号处于逻辑高电平状态期间，DSM 会将载波高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 会将载波低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：

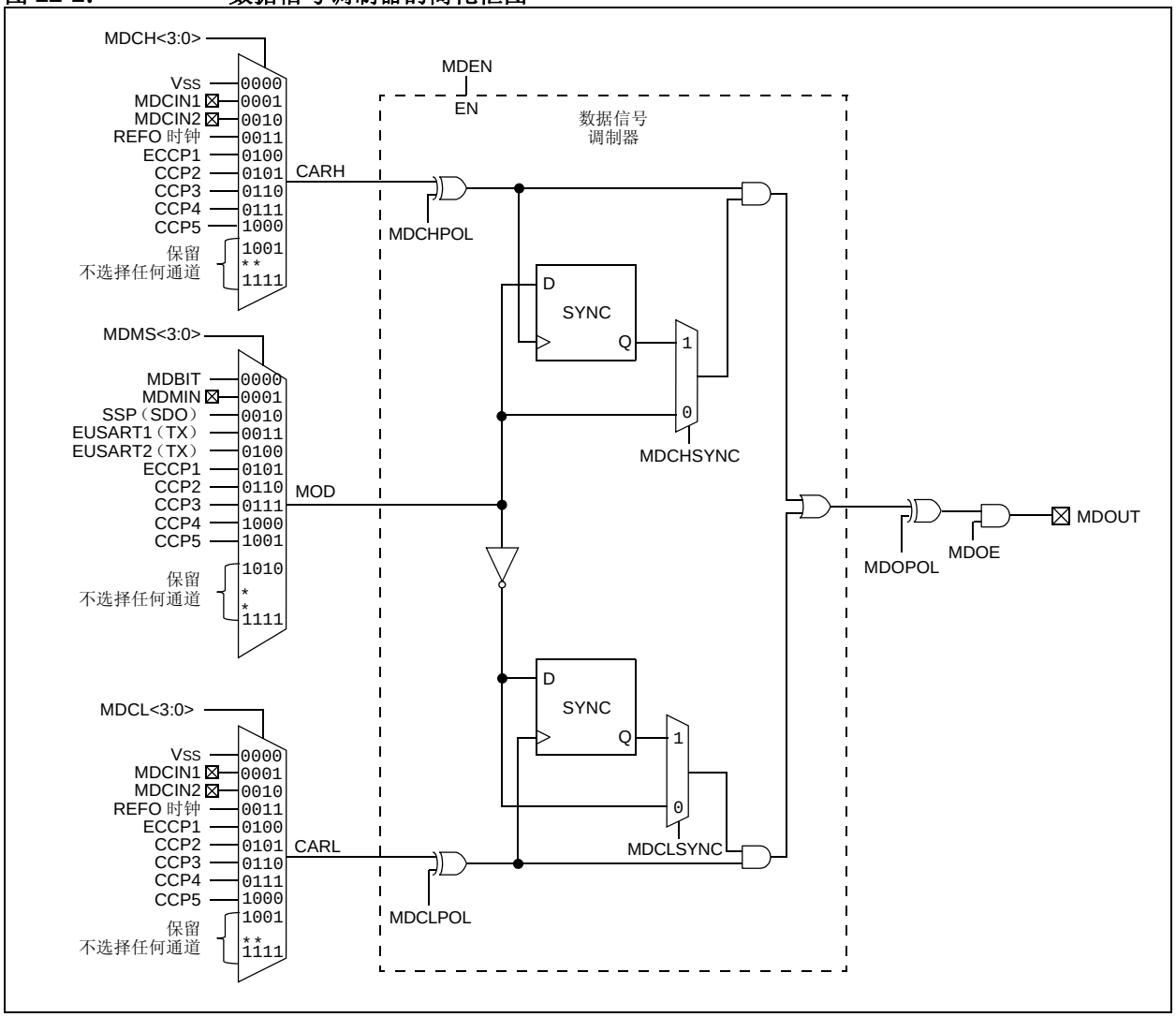
- 频移键控（Frequency-Shift Keying， FSK）
- 相移键控（Phase-Shift Keying， PSK）
- 开关键控（On-Off Keying， OOK）

此外， DSM 模块还提供了以下特性：

- 载波同步
- 载波源极性选择
- 载波源引脚禁止
- 可编程调制器数据
- 调制器源引脚禁止
- 调制输出极性选择
- 压摆率控制

图 12-1 给出了数据信号调制器外设的简化框图。

图 12-1：　　数据信号调制器的简化框图



12.1 DSM 工作原理

DSM 模块可以通过将 MDCON 寄存器中的 MDEN 位置 1 来使能。清零 MDCON 寄存器中的 MDEN 位时，将会通过自动将载波高信号和载波低信号切换至 Vss 信号源来禁止 DSM 模块。调制器信号源也会被切换至 MDCON 寄存器中的 MDBIT。这不仅可以确保 DSM 模块不活动，而且会使电流消耗降至最低。

当 MDEN 位清零且 DSM 模块被禁止时，由调制源、调制载波高信号和调制载波低信号控制寄存器保存的、用于选择载波高信号、载波低信号和调制器源的值不会受影响。在 DSM 不活动时，这些寄存器中的值会保持不变。当 MDEN 位置 1 且 DSM 模块再次使能并处于活动状态时，将会再次选择载波高信号、载波低信号和调制器信号的信号源。

用户可以在无需关闭 DSM 模块的情况下禁止调制输出信号。 DSM 模块将保持活动状态，继续对信号进行混合，但输出值不会发送到 MDOUT 引脚上。在禁止输出期间， MDOUT 引脚将保持低电平。调制输出可以通过清零 MDCON 寄存器中的 MDOE 位来禁止。

12.2 调制器信号源

调制器信号可以通过以下信号源提供：

- ECCP1 信号
- CCP2 信号
- CCP3 信号
- CCP4 信号
- CCP5 信号
- MSSP SDO 信号 （仅限 SPI 模式）
- EUSART1 TX1 信号
- EUSART2 TX2 信号
- MDMIN 引脚上的外部信号 （RF0/MDMIN）
- MDCON 寄存器中的 MDBIT 位

调制器信号通过配置MDSRC寄存器中的MDSRC<3:0>位来进行选择。

12.3 载波信号源

载波高信号和载波低信号可以通过以下信号源提供：

- CCP1 信号
- CCP2 信号
- CCP3 信号
- CCP4 信号
- 参考时钟模块信号
- MDCIN1 引脚上的外部信号 （RF2/MDCIN1）
- MDCIN2 引脚上的外部信号 （RF4/MDCIN2）
- Vss

载波高信号通过配置MDCARH寄存器中的MDCH<3:0>位来进行选择。载波低信号通过配置 MDCARL 寄存器中的 MDCL<3:0> 位来进行选择。

12.4 载波同步

当 DSM 在载波高信号源和载波低信号源之间切换时，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。

对于载波高信号源和载波低信号源，同步功能单独进行使能。载波高信号的同步可以通过将 MDCARH 寄存器中的 MDCHSYNC 位置 1 来使能。载波低信号的同步可以通过将 MDCARL 寄存器中的 MDCLSYNC 位置 1 来使能。

图 12-2 至图 12-6 给出了使用各种同步方法的时序图。

图 12-2：　　开关键控 （OOK）同步

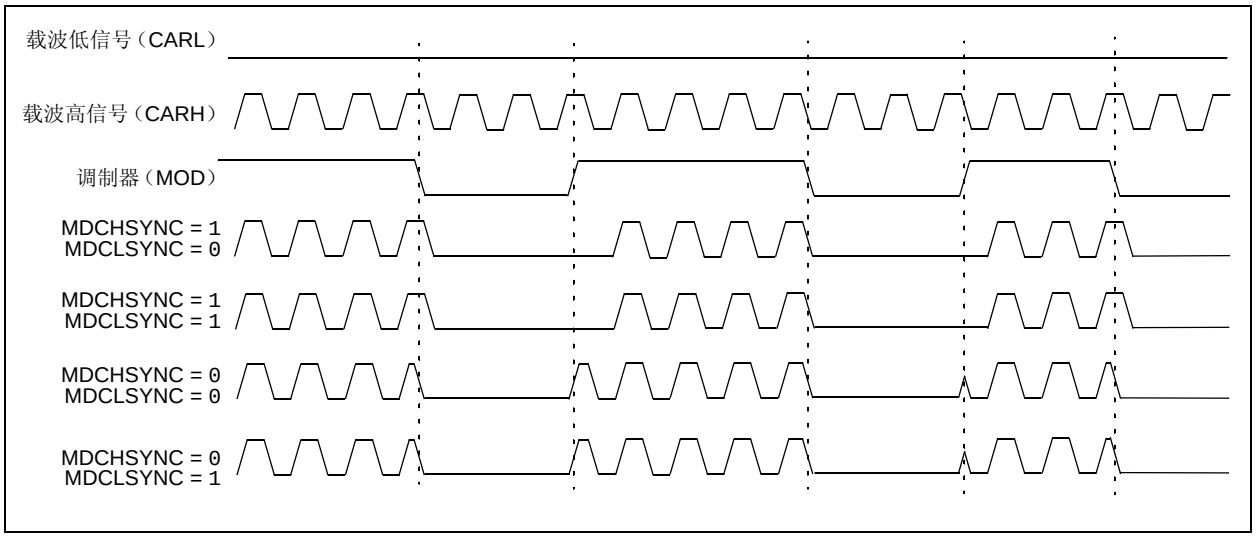


图 12-3：　　无同步 （MDCHSYNC = 0， MDCLSYNC = 0）

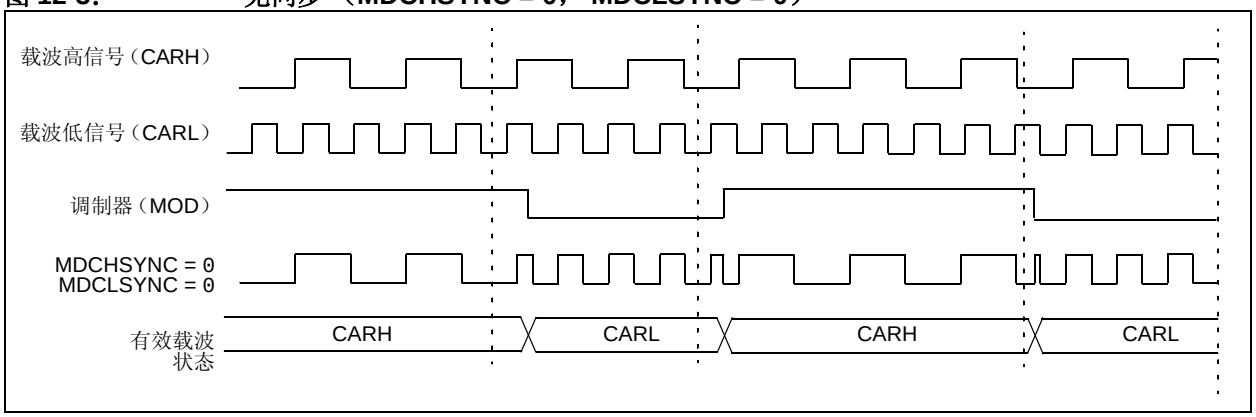


图 12-4：　　载波高信号同步 （MDCHSYNC = 1， MDCLSYNC = 0）

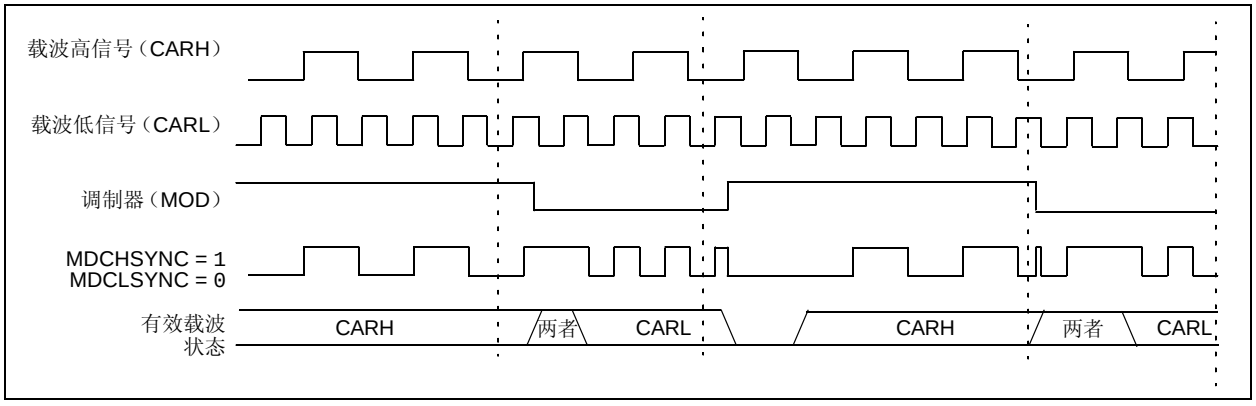


图 12-5：　　载波低信号同步 （MDCHSYNC = 0， MDCLSYNC = 1）

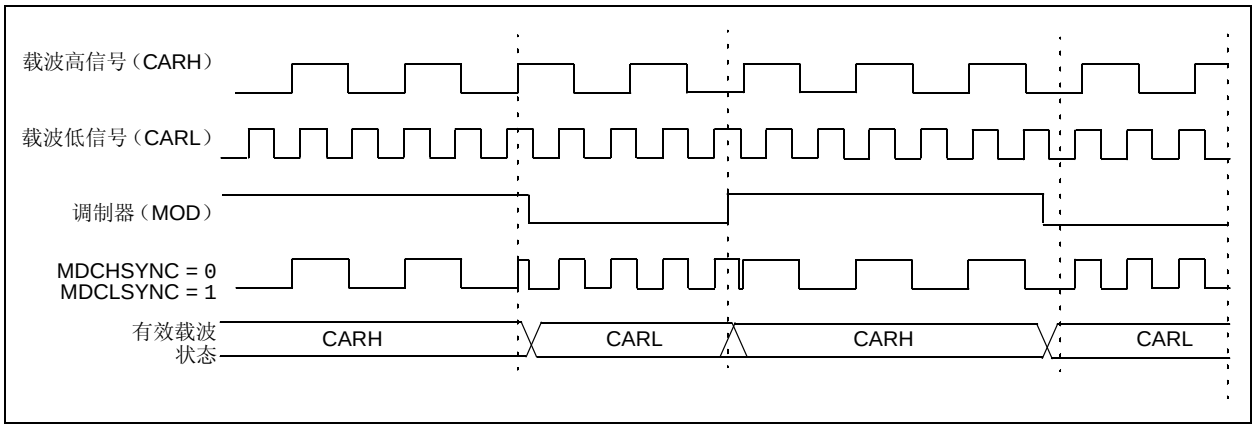
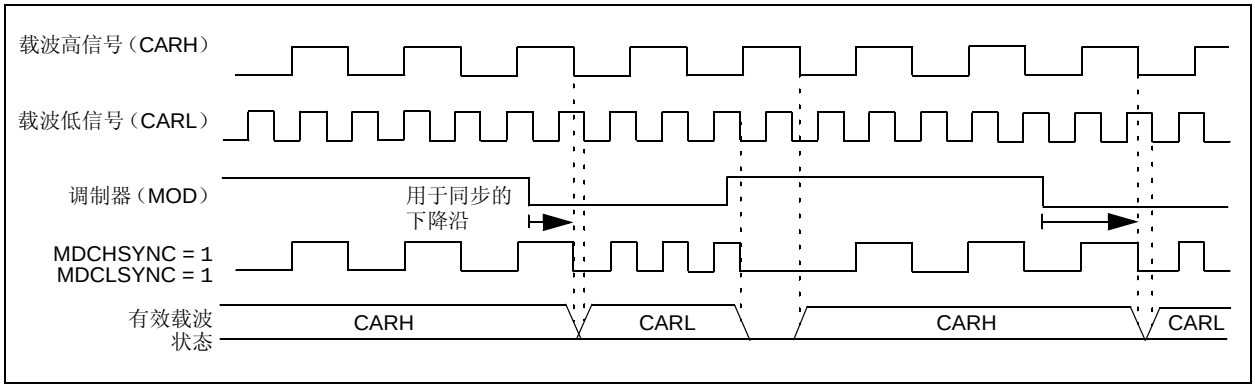


图 12-6：　　完全同步 （MDCHSYNC = 1， MDCLSYNC = 1）



12.5 载波源极性选择

任意选定输入源提供的载波高信号和载波低信号都可以反相。对载波高信号源信号反相通过将 MDCARH 寄存器的 MDCHPOL 位置 1 来使能。对载波低信号源信号反相通过将 MDCARL 寄存器的 MDCLPOL 位置 1 来使能。

12.6 载波源引脚禁止

使能某些外设时，这些外设具有其相应输出引脚的控制权。例如，当使能 CCP1 模块时， CCP1 的输出将与 CCP1 引脚连接。

引脚的默认连接可以通过将 MDCARH 寄存器中的 MDCHODIS 位（对于载波高信号源）和 MDCARL 寄存器中的 MDCLODIS 位（对于载波低信号源）置 1 来禁止。

12.7 可编程调制器数据

用户可以选择 MDCON 寄存器的 MDBIT 作为调制器信号的信号源。这使用户可以设定用于调制的值。

12.8 调制器源引脚禁止

引脚的调制器源默认连接可以通过将 MDSRC 寄存器中的 MDSODIS 位置 1 来禁止。

12.9 调制输出极性

也可以对送到 MDOUT 引脚上的调制输出信号进行反相。调制输出信号反相通过将 MDCON 寄存器的 MDOPOL 位置 1 来使能。

12.10 压摆率控制

当要求调制数据流频率为 20 MHz 或更高时，可以禁止对于输出端口引脚的压摆率限制。压摆率限制可以通过将 MDCON 寄存器中的 MDSLR 位清零来取消。

12.11 休眠模式下的操作

DSM模块不受休眠模式的影响。如果载波和调制器输入源可在休眠期间继续工作，则 DSM 也可以在休眠期间继续工作。

12.12 复位的影响

在发生任何器件复位时，数据信号调制器模块都会被禁止。用户的固件负责在使能输出之前初始化模块。寄存器会复位为其默认值。

寄存器 12-1： MDCON：调制控制寄存器

R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	U-0	U-0	R/W-0
MDEN	MDOE	MDSLR	MDOPOL	MDO	—	—	MDBIT
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 **MDEN：**调制器模块使能位
1 = 使能调制器模块，并对输入信号进行混合
0 = 禁止调制器模块，不产生任何输出

bit 6 **MDOE：**调制器模块引脚输出使能位
1 = 使能调制器引脚输出
0 = 禁止调制器引脚输出

bit 5 **MDSLR：**MDOUT 引脚压摆率限制位
1 = 使能 MDOUT 引脚压摆率限制
0 = 禁止 MDOUT 引脚压摆率限制

bit 4 **MDOPOL：**调制器输出极性选择位
1 = 调制器输出信号反相
0 = 调制器输出信号不反相

bit 3 **MDO：**调制器输出位
指示调制器模块的当前输出值。**(2)**

bit 2-1 **未实现：**读为 0

bit 0 **MDBIT：**调制器源输入位
供软件用于手动设置模块的调制源输入。**(1)**

注 1： 对于该操作，必须在 MDSRC 寄存器中选择 MDBIT 作为调制源。

2： 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或载波信号可能无效。

寄存器 12-2：　　MDSRC：调制源控制寄存器

R/W-x	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
MDSODIS	—	—	—	MDSRC3	MDSRC2	MDSRC1	MDSRC0
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　**MDSODIS：**调制源输出禁止位
1 = 禁止驱动外设输出引脚 （通过 MDMS<3:0> 选择）的输出信号
0 = 使能驱动外设输出引脚 （通过 MDMS<3:0> 选择）的输出信号

bit 6-4　**未实现：**读为 0

bit 3-0　**MDSRC<3:0>：**调制源选择位
1111-1010 = 保留；不连接任何通道
1001 = CCP5 输出 （仅限 PWM 输出模式）
1000 = CCP4 输出 （仅限 PWM 输出模式）
0111 = CCP3 输出 （仅限 PWM 输出模式）
0110 = CCP2 输出 （仅限 PWM 输出模式）
0101 = ECCP1 输出 （仅限 PWM 输出模式）
0100 = EUSART2 TX 输出
0011 = EUSART1 TX 输出
0010 = MSSP SDO 输出
0001 = MDMIN 端口引脚
0000 = MDCON 寄存器的 MDBIT 位是调制源

寄存器 12-3： MDCARH：调制载波高信号控制寄存器

R/W-x	R/W-x	R/W-x	U-0	R/W-x	R/W-x	R/W-x	R/W-x
MDCHODIS	MDCHPOL	MDCHSYNC	—	MDCH3(1)	MDCH2(1)	MDCH1(1)	MDCH0(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 **MDCHODIS：**调制器载波高信号输出禁止位
1 = 禁止驱动外设输出引脚 （通过 MDCH<3:0> 选择）的输出信号
0 = 使能驱动外设输出引脚 （通过 MDCH<3:0> 选择）的输出信号

bit 6 **MDCHPOL：**调制器载波高信号极性选择位
1 = 选定的载波高信号反相
0 = 选定的载波高信号不反相

bit 5 **MDCHSYNC：**调制器载波高信号同步使能位
1 = 调制器先等待载波高信号上出现下降沿，然后再切换为载波低信号
0 = 调制器输出不与载波高信号进行同步 (1)

bit 4 **未实现：**读为 0

bit 3-0 **MDCH<3:0>：**调制器数据载波高信号选择位 (1)
1111-1001 = 保留
1000 = CCP5 输出 （仅限 PWM 输出模式）
0111 = CCP4 输出 （仅限 PWM 输出模式）
0110 = CCP3 输出 （仅限 PWM 输出模式）
0101 = CCP2 输出 （仅限 PWM 输出模式）
0100 = ECCP1 输出 （仅限 PWM 输出模式）
0011 = 参考时钟模块信号
0010 = MDCIN2 端口引脚
0001 = MDCIN1 端口引脚
0000 = VSS

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，或者可能出现尖峰。

寄存器 12-4： MDCARL：调制载波低信号控制寄存器

R/W-0	R/W-x	R/W-x	U-0	R/W-x	R/W-x	R/W-x	R/W-x
MDCLODIS	MDCLPOL	MDCLSYNC	—	MDCL3(1)	MDCL2(1)	MDCL1(1)	MDCL0(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 **MDCLODIS：**调制器载波低信号输出禁止位
1 = 禁止驱动外设输出引脚 （通过 MDCARL 寄存器的 MDCL<3:0> 选择）的输出信号
0 = 使能驱动外设输出引脚 （通过 MDCARL 寄存器的 MDCL<3:0> 选择）的输出信号

bit 6 **MDCLPOL：**调制器载波低信号极性选择位
1 = 选定的载波低信号反相
0 = 选定的载波低信号不反相

bit 5 **MDCLSYNC：**调制器载波低信号同步使能位
1 = 调制器先等待载波低信号上出现下降沿，然后再切换为载波高信号
0 = 调制器输出不与载波低信号进行同步 (1)

bit 4 **未实现：**读为 0

bit 3-0 **MDCL<3:0>：**调制器数据载波低信号选择位 (1)
1111-1001 = 保留
1000 = CCP5 输出 （仅限 PWM 输出模式）
0111 = CCP4 输出 （仅限 PWM 输出模式）
0110 = CCP3 输出 （仅限 PWM 输出模式）
0101 = CCP2 输出 （仅限 PWM 输出模式）
0100 = ECCP1 输出 （仅限 PWM 输出模式）
0011 = 参考时钟模块信号
0010 = MDCIN2 端口引脚
0001 = MDCIN1 端口引脚
0000 = VSS

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，或者可能出现尖峰。

表 12-1： 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
MDCARH	MDCHODIS	MDCHPOL	MDCHSYNC	—	MDCH3	MDCH2	MDCH1	MDCH0
MDCARL	MDCLODIS	MDCLPOL	MDCLSYNC	—	MDCL3	MDCL2	MDCL1	MDCL0
MDCON	MDEN	MDOE	MDSLR	MDOPOL	MDO	—	—	MDBIT
MDSRC	MDSODIS	—	—	—	MDSRC3	MDSRC2	MDSRC1	MDSRC0
PMD2	—	—	—	—	MODMD	ECANMD	CMP2MD	CMP1MD

图注： — = 未实现，读为 0。数据信号调制器模式下不使用阴影单元。

13.0 TIMER0 模块

Timer0 模块具有以下特性：

- 可由软件选择作为 8 位或 16 位定时器 / 计数器
- 可读写寄存器
- 专用的 8 位软件可编程预分频器
- 可选的时钟源 （内部或外部）
- 外部时钟的边沿选择
- 溢出时产生中断

T0CON 寄存器（寄存器 13-1）控制该模块操作的所有方面，包括预分频比的选择。它是可读写的。

图 13-1 给出了 8 位模式下 Timer0 模块的简化框图。
图 13-2 给出了 16 位模式下 Timer0 模块的简化框图。

寄存器 13-1： T0CON：TIMER0 控制寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
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 **TMR0ON**：Timer0 开 / 关控制位
1 = 使能 Timer0
0 = 停止 Timer0

bit 6 **T08BIT**：Timer0 8 位 /16 位控制位
1 = Timer0 被配置为 8 位定时器 / 计数器
0 = Timer0 被配置为 16 位定时器 / 计数器

bit 5 **T0CS**：Timer0 时钟源选择位
1 = T0CKI 引脚上的电平跳变
0 = 内部指令周期时钟 （CLKO）

bit 4 **T0SE**：Timer0 时钟源边沿选择位
1 = 在 T0CKI 引脚信号从高至低跳变时，递增计数
0 = 在 T0CKI 引脚信号从低至高跳变时，递增计数

bit 3 **PSA**：Timer0 预分频器分配位
1 = 未分配 Timer0 预分频器。 Timer0 时钟输入不经预分频器分频
0 = 已分配 Timer0 预分频器。 Timer0 时钟输入来自预分频器的输出

bit 2-0 **T0PS<2:0>**：Timer0 预分频比选择位
111 = 1:256 预分频比
110 = 1:128 预分频比
101 = 1:64 预分频比
100 = 1:32 预分频比
011 = 1:16 预分频比
010 = 1:8 预分频比
001 = 1:4 预分频比
000 = 1:2 预分频比

13.1 Timer0 工作原理

Timer0 既可用作定时器也可用作计数器。可以通过 T0CS 位（T0CON<5>）来选择模式。在定时器模式（T0CS = 0）下，该模块在每个时钟周期都会递增（默认情况下），除非选择了其他预分频值（见**第 13.3 节 “预分频器 ”**）。在对 TMR0 寄存器执行写操作之后的两个指令周期内禁止 Timer0 递增。用户可通过将调整值写入 TMR0 寄存器来解决这一问题。

通过将 T0CS 位置 1（= 1）选择计数器模式。在该模式下， Timer0 可在 T0CKI 引脚信号的每个上升沿或下降沿递增。递增边沿由 Timer0 时钟源边沿选择位 T0SE（T0CON<4>）决定；清零该位即选择上升沿。下面讨论外部时钟输入的限制条件。

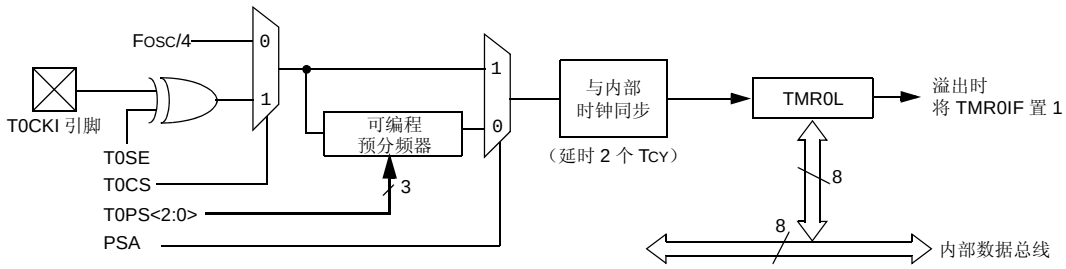
可以使用外部时钟源来驱动 Timer0 ；但是，必须满足一定要求，以确保外部时钟和内部相位时钟（Tosc）保持同步。在同步之后，定时器 / 计数器需要一定的延时才开始递增。

13.2 16 位模式下 Timer0 的读写操作

TMR0H 并不是 16 位模式下 Timer0 的实际高字节，而是 Timer0 实际高字节的缓存形式，不可以被直接读写（见图 13-2）。在读 TMR0L 时使用 Timer0 高字节的内容更新 TMR0H。这种方式使用户可以读取 Timer0 的全部 16 位，而不需要验证高字节和低字节读取的有效性。若采用高字节和低字节连续读取的方式，由于两次读取之间可能发生低字节向高字节的进位，因而导致读取值无效。

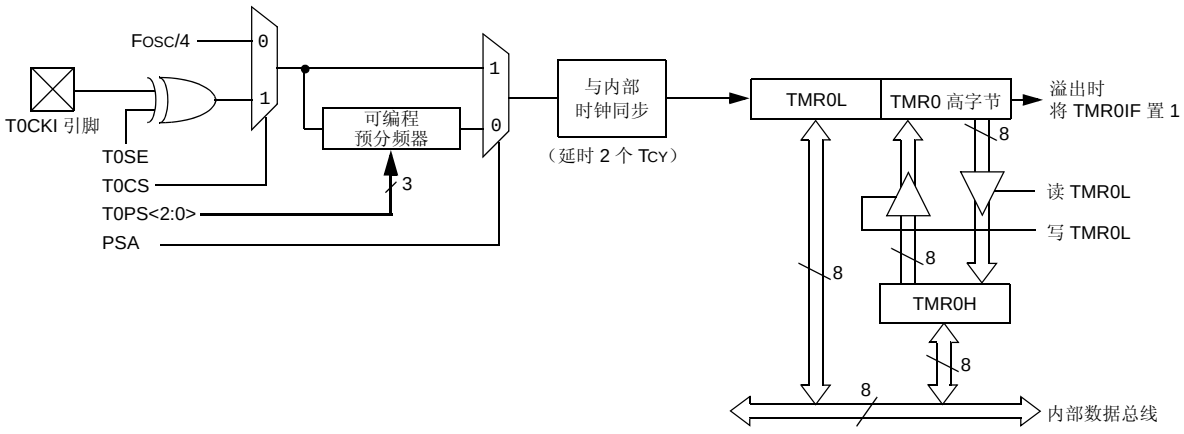
同样，写入 Timer0 的高字节也必须通过 TMR0H 缓冲寄存器来操作。在写入 TMR0L 的同时，使用 TMR0H 的内容更新 Timer0 的高字节。这样一次就可以完成 Timer0 全部 16 位的更新。

图 13-1： TIMER0 框图 （8 位模式）



注： 复位时，Timer0 被使能为在 8 位模式下工作，其时钟输入来自 T0CKI 引脚的最大预分频信号。

图 13-2： TIMER0 框图 （16 位模式）



注： 复位时，Timer0 被使能为在 8 位模式下工作，其时钟输入来自 T0CKI 引脚的最大预分频信号。

13.3 预分频器

Timer0 模块的预分频器为一个 8 位计数器。该预分频器不可直接读写。通过PSA和T0PS<2:0>位（T0CON<3:0>）进行预分频器的分配和设定预分频比。

将 PSA 位清零可将预分频器分配给 Timer0 模块。如果已经分配了预分频器，预分频值可在 1:2 至 1:256 之间进行选择，以 2 的整数次幂递增。

如果将预分频器分配给 Timer0 模块，所有写入 TMR0 寄存器的指令（例如，CLRF TMR0、MOVWF TMR0 和 BSF TMR0 等）都会将预分频器的计数值清零。

注：	如果将预分频器分配给Timer0，写入TMR0会将预分频器的计数值清零，但不会改变预分频器的分配。

13.3.1 切换预分频器的分配

预分频器的分配完全由软件控制，并且在程序执行期间可以随时更改。

13.4 Timer0 中断

8 位模式下 TMR0 寄存器从 FFh 溢出到 00h，或 16 位模式下 TMR0 从 FFFFh 溢出到 0000h 时，将产生 TMR0 中断。这种溢出会将 TMR0IF 标志位置 1。可以通过清零 TMR0IE 位（INTCON<5>）来屏蔽该中断。在重新允许该中断前，必须在中断服务程序（ISR）中用软件清零 TMR0IF 位。

由于 Timer0 在休眠模式下是关闭的，所以 TMR0 中断无法将处理器从休眠状态唤醒。

表 13-1： 与 TIMER0 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TMR0L	Timer0 寄存器的低字节							
TMR0H	Timer0 寄存器的高字节							
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD

图注： — = 未实现，读为 0。 Timer0 不使用阴影单元。

注：

14.0 TIMER1 模块

Timer1 定时器 / 计数器模块具有以下特性：

- 可由软件选择作为 16 位定时器或计数器
- 可读写的 8 位寄存器 （TMR1H 和 TMR1L）
- 可选择器件时钟或 SOSC 内部振荡器作为时钟源（内部或外部）
- 溢出时产生中断
- 在 ECCP 特殊事件触发时复位
- 带门控的定时器

图 14-1 给出了 Timer1 模块的简化框图。

模块基于辅助振荡器或外部数字时钟源来产生其时钟源。如果使用辅助振荡器，则还有另外一些选项，可用于选择低功耗、高功耗和外部数字时钟源。

Timer1 由 T1CON 控制寄存器（寄存器 14-1）控制。该寄存器还包含 Timer1 振荡器使能位 （SOSCEN）。可以通过将控制位 TMR1ON （T1CON<0>）置 1 或清零来使能或禁止 Timer1。

Fosc 时钟源不应与 ECCP 捕捉 / 比较功能一起使用。如果捕捉或比较功能需要使用定时器，则应选择其他定时器时钟源选项。

寄存器 14-1： T1CON：TIMER1 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON
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 **TMR1CS<1:0>：**Timer1 时钟源选择位
10 = Timer1 时钟源来自引脚或振荡器，具体取决于 SOSCEN 位的设置：
SOSCEN = 0：
外部时钟来自 T1CKI 引脚 （上升沿计数）。
SOSCEN = 1：
根据 SOSCSEL 配置位的设置，时钟源可以是 SOSCI/SOSCO 上的晶振或来自 SCLKI 引脚的内部数字时钟。
01 = Timer1 时钟源为系统时钟 （Fosc） **(1)**
00 = Timer1 时钟源为指令时钟 （Fosc/4）

bit 5-4 **T1CKPS<1:0>：**Timer1 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 **SOSCEN：**SOSC 振荡器使能位
1 = 使能 SOSC，它可用于 Timer1
0 = 禁止 SOSC，它不可用于 Timer1
关闭振荡器的反相器和反馈电阻以减少功耗。

bit 2 **$\overline{\text{T1SYNC}}$：**Timer1 外部时钟输入同步选择位
TMR1CS<1:0> = 10：
1 = 不同步外部时钟输入
0 = 同步外部时钟输入
TMR1CS<1:0> = 0x：
该位被忽略。当 TMR1CS<1:0> = 1x 时， Timer1 使用内部时钟。

注 1： 如果定时器要与 ECCP 捕捉 / 比较功能一起使用，则不应选择 Fosc 时钟源。

寄存器 14-1：　　T1CON：TIMER1 控制寄存器（续）

bit 1　　**RD16**：16 位读 / 写模式使能位
1 = 使能 Timer1 通过一次 16 位操作进行寄存器读 / 写
0 = 使能 Timer1 通过两次 8 位操作进行寄存器读 / 写

bit 0　　**TMR1ON**：Timer1 使能位
1 = 使能 Timer1
0 = 停止 Timer1

注　1： 如果定时器要与 ECCP 捕捉 / 比较功能一起使用，则不应选择 FOSC 时钟源。

14.1 Timer1 门控控制寄存器

Timer1 门控控制寄存器（T1GCON）如寄存器 14-2 所示，用于控制 Timer1 门控。

寄存器 14-2： T1GCON：TIMER1 门控控制寄存器 (1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-x	R/W-0	R/W-0
TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/$\overline{\text{T1DONE}}$	T1GVAL	T1GSS1	T1GSS0
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 **TMR1GE**：Timer1 门控使能位
如果 TMR1ON = 0：
该位被忽略。
如果 TMR1ON = 1：
1 = Timer1 计数由 Timer1 门控功能控制
0 = Timer1 计数与 Timer1 门控功能无关

bit 6 **T1GPOL**：Timer1 门控极性位
1 = Timer1 门控高电平有效 （当门控为高电平时 Timer1 计数）
0 = Timer1 门控低电平有效 （当门控为低电平时 Timer1 计数）

bit 5 **T1GTM**：Timer1 门控翻转模式位
1 = 使能 Timer1 门控翻转模式
0 = 禁止 Timer1 门控翻转模式并清除单稳态触发器的输出
Timer1 门控单稳态触发器在每个上升沿翻转。

bit 4 **T1GSPM**：Timer1 门控单脉冲模式位
1 = 使能 Timer1 门控单脉冲模式，控制 Timer1 门控
0 = 禁止 Timer1 门控单脉冲模式

bit 3 **T1GGO/$\overline{\text{T1DONE}}$**：Timer1 门控单脉冲采集状态位
1 = Timer1 门控单脉冲采集就绪，正在等待一个边沿
0 = Timer1 门控单脉冲采集已经结束或尚未开始
当 T1GSPM 清零时，该位会自动清零。

bit 2 **T1GVAL**：Timer1 门控当前状态位
指示可提供给 TMR1H:TMR1L 的 Timer1 门控信号的当前状态；不受 Timer1 门控使能（TMR1GE）位的影响。

bit 1-0 **T1GSS<1:0>**：Timer1 门控源选择位
11 = 比较器 2 的输出
10 = 比较器 1 的输出
01 = TMR2 匹配 PR2 输出
00 = Timer1 门控引脚

注 1： 建议在设定 T1CON 之前先设定 T1GCON。

14.2 Timer1 工作原理

Timer1 模块是一个 8 位或 16 位递增计数器，可通过 TMR1H:TMR1L 寄存器对访问。

Timer1 与内部时钟源一起使用时，模块为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 通过分别配置 T1CON 和 T1GCON 寄存器中的 TMR1ON 和 TMR1GE 位使能。

将 SOSC 选择为晶振模式（通过 SOSCSEL）时，RC1/SOSCI 和 RC0/SOSCO/SCLKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

14.3 时钟源选择

T1CON 寄存器的 TMR1CS<1:0> 和 SOSCEN 位用于选择 Timer1 的时钟源。表 14-1 显示了时钟源选择。

14.3.1 内部时钟源

当选择内部时钟源时， TMR1H:TMR1L 寄存器对将在 FOSC 的整数倍 （由 Timer1 预分频比决定）处递增。

14.3.2 外部时钟源

当选择外部时钟源时，Timer1 模块可以作为定时器或计数器工作。

使能计数时，Timer1 在外部时钟输入 T1CKI 的上升沿递增。这些外部时钟源可以与单片机系统时钟同步，也可以异步运行。

依靠时钟振荡器作为定时器工作时，可以将外部 32.768 kHz 晶振与专用内部振荡器电路一起使用。

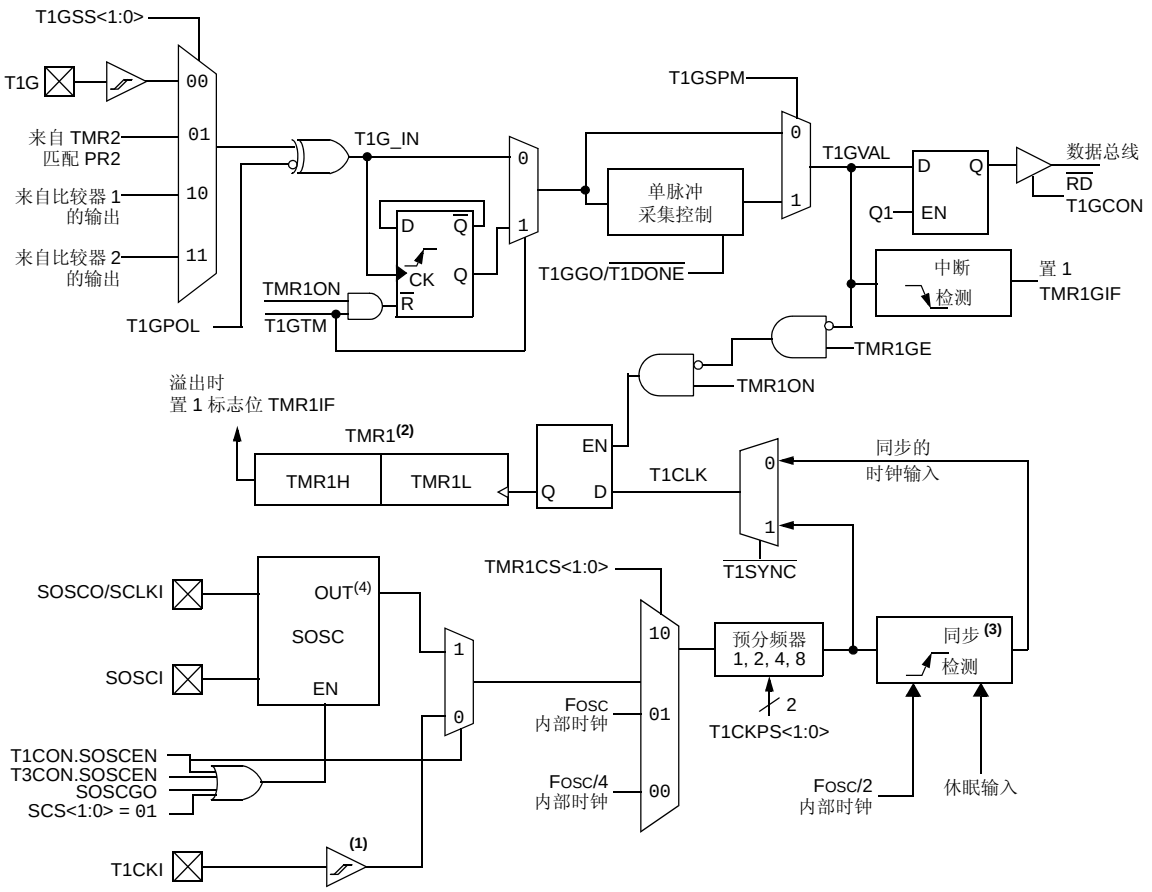
注： 在计数器模式下，发生以下任何一个或多个情况后，计数器在首个上升沿递增前，必须先经过一个下降沿：

- Timer1 在 POR 复位后被使能
- 写入 TMR1H 或 TMR1L
- Timer1 被禁止
- T1CKI 为高电平时 Timer1 被禁止（TMR1ON = 0）， T1CKI 为低电平时 Timer1 被使能 （TMR1ON = 1）。

表 14-1： TIMER1 时钟源选择

TMR1CS1	TMR1CS0	SOSCEN	时钟源
0	1	x	时钟源 （FOSC）
0	0	x	指令时钟 （FOSC/4）
1	0	0	T1CKI 引脚上的外部时钟
1	0	1	SOSCI/SOSCO 引脚上的振荡器电路

图 14-1： TIMER1 框图



注 1： 当使用 T1CKI 时， ST 缓冲器为高速型。
2： Timer1 寄存器在上升沿递增。
3： 处于休眠模式时同步不工作。
4： SOSC 的输出由 SOSCSEL<1:0> 配置位决定。

14.4 Timer1 的 16 位读 / 写模式

可将 Timer1 配置为 16 位读写模式。当 RD16 控制位（T1CON<1>）置 1 时，TMR1H 的地址被映射到 Timer1 的高字节缓冲寄存器。读 TMR1L 会将 Timer1 的高字节的内容装入 Timer1 高字节缓冲寄存器。这种方式使用户可以精确地读取 Timer1 的全部 16 位，而不需要像先读高字节再读低字节那样，由于两次读取之间可能存在进位，而不得不验证读取的有效性。

写 Timer1 的高字节也必须通过 TMR1H 缓冲寄存器进行。在写入 TMR1L 的同时，使用 TMR1H 的内容更新 Timer1 的高字节。这样允许用户将所有 16 位一次写入 Timer1 的高字节和低字节。

在该模式下不能直接读写 Timer1 的高字节。所有读写都必须通过 Timer1 高字节缓冲寄存器来进行。写入 TMR1H 不会清零 Timer1 预分频器。只有在写 TMR1L 时才会清零该预分频器。

14.5 SOSC 振荡器

片上晶振电路连接在 SOSCI（输入）引脚和 SOSCO（放大器输出）引脚之间。它可以使用以下方法之一使能：

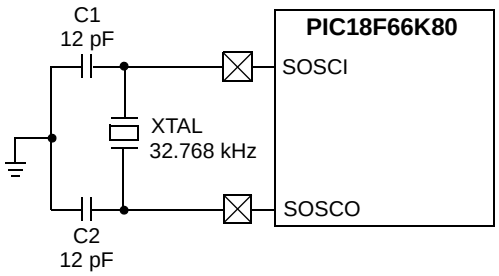
- 将 T1CON 或 T3CON 寄存器中的 SOSCEN 位（TxCON<3>）置 1
- 将 OSCCON2 寄存器中的 SOSCGO 位（OSCCON2<3>）置 1
- 将 OSCCON 寄存器中的 SCS 位设置为辅助时钟源（OSCCON<1:0> = 01）

SOSCGO 位用于预热 SOSC，使之在任意外设发出请求之前就绪。

该振荡器电路是一种低功耗电路，它采用了额定振荡频率为 32 kHz 的晶振。在所有功耗管理模式下都可继续运行。图 14-2 所示为典型的低功耗振荡器电路。表 14-2 给出了供 SOSC 振荡器选择的电容值。

用户必须提供软件延时来确保 SOSC 振荡器的正常起振。

图 14-2： SOSC 振荡器的外部元件



注： 关于电容选择的更多信息，请参见表 14-2 下方的“注”。

表 14-2： 定时器振荡器的电容选择 (2,3,4,5)

振荡器类型	频率	C1	C2
LP	32 kHz	12 pF(1)	12 pF(1)

注 1： Microchip 建议仅将这些值作为验证振荡器电路的起始点。

2： 电容越大，振荡器越稳定，但起振时间越长。

3： 因为每种谐振器 / 晶振都有其自身特性，用户应当向谐振器 / 晶振制造商咨询外部元件的适当值。

4： 上述电容值仅供设计参考。列出的值是 SOSCSEL = 11 时，额定负载电容为 CL = 10 pF 的晶振的典型值。

5： 如果电容值不正确，则可能导致频率不满足晶振制造商的容差规范。

SOSC晶振驱动电平根据SOSCSEL（CONFIG1L<4:3>）配置位确定。高驱动电平模式（SOSCSEL<1:0> = 11）用于驱动具有多种负载电容（CL）额定值的一系列广泛的 32.768 kHz 晶振。

低驱动电平模式则是针对超低功耗而高度优化。它不用于驱动任何类型的 32.768 kHz 晶振。在低驱动电平模式下，如果 SOSCO 和 SOSCI 引脚上放置了极大的分立电容，则晶振电路可能不能正确工作。该模式仅设计为在以下情况下工作：每个引脚上的分立电容约为3 pF-10 pF。

晶振制造商通常会为他们的晶振指定一个 CL（负载电容）额定值。图 14-2 中的 C1 和 C2 就是要用于晶振的电容，指定的值可以与图示相同，也可以不同。

关于如何为给定晶振选择最佳 C1 和 C2 的更多详细信息，请参见晶振制造商的应用信息。最佳值部分依赖于电路中的寄生电容，该寄生电容通常是未知的。因此，选择了电容值之后，强烈建议您对振荡器执行全面的测试和验证。

14.5.1 使用 SOSC 作为时钟源

在功耗管理模式下也可以将SOSC振荡器用作时钟源。通过将时钟选择位 SCS<1:0>（OSCCON<1:0>）设置为 01，器件可以切换到 SEC_RUN 模式； CPU 和外设都可以用 SOSC 振荡器作为时钟源。如果 IDLEN 位（OSCCON<7>）被清零并且执行了 SLEEP 指令，器件将进入 SEC_IDLE 模式。更多详细信息，请参见**第 4.0 节 “功耗管理模式 ”**。

无论何时将 SOSC 振荡器用作时钟源，SOSC 系统时钟状态标志 SOSCRUN （OSCCON2<6>）均会置 1。这可用于确定控制器的当前时钟模式。该位也可指示故障保护时钟监视器当前正使用的时钟源。

如果使能了时钟监视器并且SOSC振荡器在提供时钟信号时发生了故障，查询SOCSRUN位可以确定时钟源是 SOSC 振荡器还是其他时钟源。

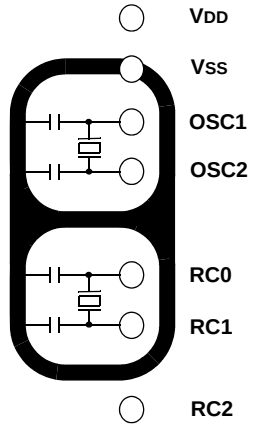
14.5.2 SOSC 振荡器布线注意事项

SOSC振荡器电路在工作期间仅消耗极小的电流。鉴于此振荡器的低功耗特性，它对附近变化较快的信号比较敏感。在振荡器被配置为超低功耗模式 （SOSCSEL<1:0>（CONFIG1L<4:3>） = 01）时尤其如此。

如图 14-2 所示，振荡器电路应该尽可能靠近单片机。除了 VSS 或 VDD 外，在该振荡器电路边界内不应有其他电路通过。

如果必须要在该振荡器附近布高速电路，那么在该振荡器电路周围布接地保护环可能会有帮助。可以在单面 PCB 上使用接地保护环 （如图 14-3 所示），或再加一个地平面。（高速电路示例包括输出比较或PWM 模式下的 ECCP1 引脚，或使用 OSC2 引脚的主振荡器。）

图 14-3： 带有接地保护环的振荡器电路



注：未按比例绘制。

在低驱动电平模式 （SOSCSEL<1:0> = 01）下，有一点非常关键，即使 RC2 I/O 引脚信号避开振荡器电路。如果将 RC2 配置为数字输出并不断翻转它的输出信号，则可能会影响振荡器电路，即使是对于相对较好的 PCB 布线设计也是如此。如果可能，将 RC2 保留为不用，或者将它用作输入引脚，但对其信号源的压摆率有一定限制。如果必须将 RC2 用作数字输出，则对于许多 PCB 布线设计，可能必须使用高驱动电平振荡器模式 （SOSCSEL<1:0> = 11）。

即使是在高驱动电平模式下，设计振荡器电路时，仍然应当谨慎布线。

除了要考虑 dV/dt 感应噪声之外，确保电路板洁净也非常重要。即使是极少量的导电焊剂残留物也会导致 PCB 泄漏电流，对振荡器电路产生很大的影响。

14.6 Timer1 中断

TMR1 寄存器对 （TMR1H:TMR1L）从 0000h 递增到 FFFFh，然后计满返回到 0000h 重新开始计数。如果允许了 Timer1 中断，则溢出时会产生 Timer1 中断，并反映到中断标志位 TMR1IF （PIR1<0>）中。可以通过将 Timer1 中断允许位 TMR1IE（PIE1<0>）置 1 或清零来允许或禁止该中断。

14.7 使用 ECCP 特殊事件触发信号复位 Timer1

如果 ECCP 模块配置为使用 Timer1 并在比较模式（CCP1M<3:0> = 1011）下产生特殊事件触发信号，该信号将复位 Timer1。如果使能了 A/D 模块，来自 ECCP 的触发信号还将启动 A/D 转换。（更多信息，请参见**第 20.3.4 节 “ 特殊事件触发器 ”**。）

要使用这一功能，必须将模块配置为定时器或同步计数器。在这种情况下， CCPR1H:CCPR1L 寄存器对实际上变成了 Timer1 的周期寄存器。

如果 Timer1 在异步计数器模式下运行，复位操作可能不起作用。

如果对 Timer1 的写操作和特殊事件触发信号同时发生，则写操作优先。

注：	ECCP 模块产生的特殊事件触发信号只会清除 TMR1 寄存器的内容，而不会将 TMR1IF 中断标志位 （PIR1<0>）置 1。

14.8 Timer1 门控

Timer1 可配置为自由计数或用 Timer1 门控电路使能和禁止计数。这也称为 Timer1 门控计数使能。

Timer1 门控也可由多个可选择源驱动。

14.8.1 TIMER1 门控计数使能

通过将 T1GCON 寄存器的 TMR1GE 位置 1 使能 Timer1 门控使能模式。使用 T1GPOL 位 （T1GCON<6>）来配置 Timer1 门控使能模式的极性。

使能 Timer1 门控使能模式时，Timer1 将在 Timer1 时钟源的上升沿递增。禁止 Timer1 门控使能模式时，不会发生递增，Timer1 将保持当前计数。时序详细信息请参见图 14-4。

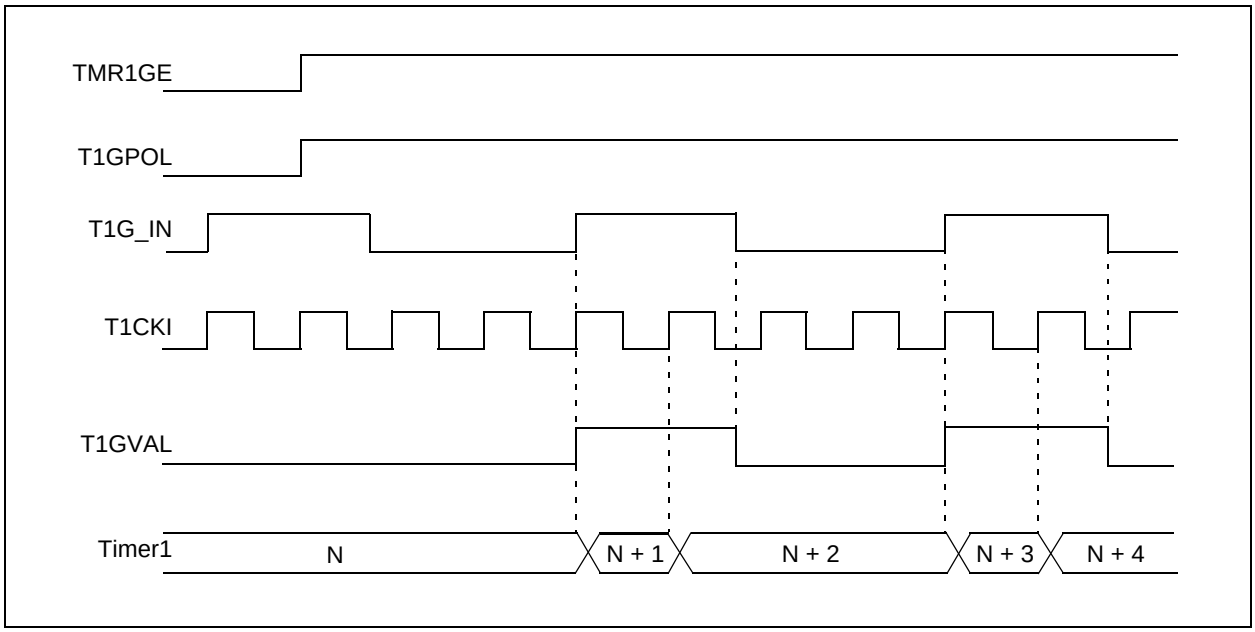
表 14-3： TIMER1 门控使能选择

T1CLK(†)	T1GPOL （T1GCON<6>）	T1G 引脚	Timer1 工作状态
↑	0	0	计数
↑	0	1	保持计数
↑	1	0	保持计数
↑	1	1	计数

† TMR1 运行所使用的时钟。更多信息，请参见图 14-1。

注：	对于某些模式，CCP 和 ECCP 模块会使用 Timer1 至 Timer4。要将哪个特定的定时器分配给 CCP/ECCP 模块由 CCPTMRS 寄存器中的 “CCP 的定时器 ” 使能位决定。更多详细信息，请参见寄存器 20-2 和寄存器 19-2。

图 14-4：　　TIMER1 门控计数使能模式



14.8.2　TIMER1 门控源选择

Timer1 门控源可从四种不同源之中选择。门控源的选择由 T1GSSx（T1GCON<1:0>）位控制（见表 14-4）。

表 14-4：　　TIMER1 门控源

T1GSS<1:0>	Timer1 门控源
00	Timer1 门控引脚
01	TMR2 匹配 PR2 （TMR2 递增以匹配 PR2）
10	比较器 1 的输出 （比较器逻辑高电平输出）
11	比较器 2 的输出 （比较器逻辑高电平输出）

每个可用源的极性也是可选择的，由 T1GPOL 位（T1GCON<6>）控制。

14.8.2.1　T1G 引脚门控操作

T1G引脚是Timer1门控控制源之一。它可用于向Timer1门控电路提供外部源。

14.8.2.2　Timer2 匹配门控操作

TMR2 寄存器将递增到与 PR2 寄存器中的值匹配为止。在紧接着的下一个递增周期，TMR2 将复位为 00h。发生该复位时，将自动产生由低至高脉冲并在内部提供给 Timer1 门控电路。脉冲将保持一个指令周期的高电平，然后恢复为低电平状态，直到再一次发生匹配。

根据 T1GPOL 的设置，Timer1 在 TMR2 与 PR2 发生匹配时的递增形式会不同。当 T1GPOL = 1 时，Timer1 在 TMR2 与 PR2 发生匹配之后的单个指令周期中递增。当 T1GPOL = 0 时，发生匹配后的一个周期以后，Timer1 会在门控信号从低电平变为高电平时不断递增。

14.8.2.3　比较器 1 输出门控操作

比较器 1 的输出可以在内部送至 Timer1 门控电路。使用 CM1CON 寄存器设置比较器 1 之后，Timer1 会根据 CMP1OUT（CMSTAT<6>）位的电平变化进行递增。

14.8.2.4　比较器 2 输出门控操作

比较器 2 的输出可以在内部送至 Timer1 门控电路。使用 CM2CON 寄存器设置比较器 2 之后，Timer1 会根据 CMP2OUT（CMSTAT<7>）位的电平变化进行递增。

14.8.3 TIMER1 门控翻转模式

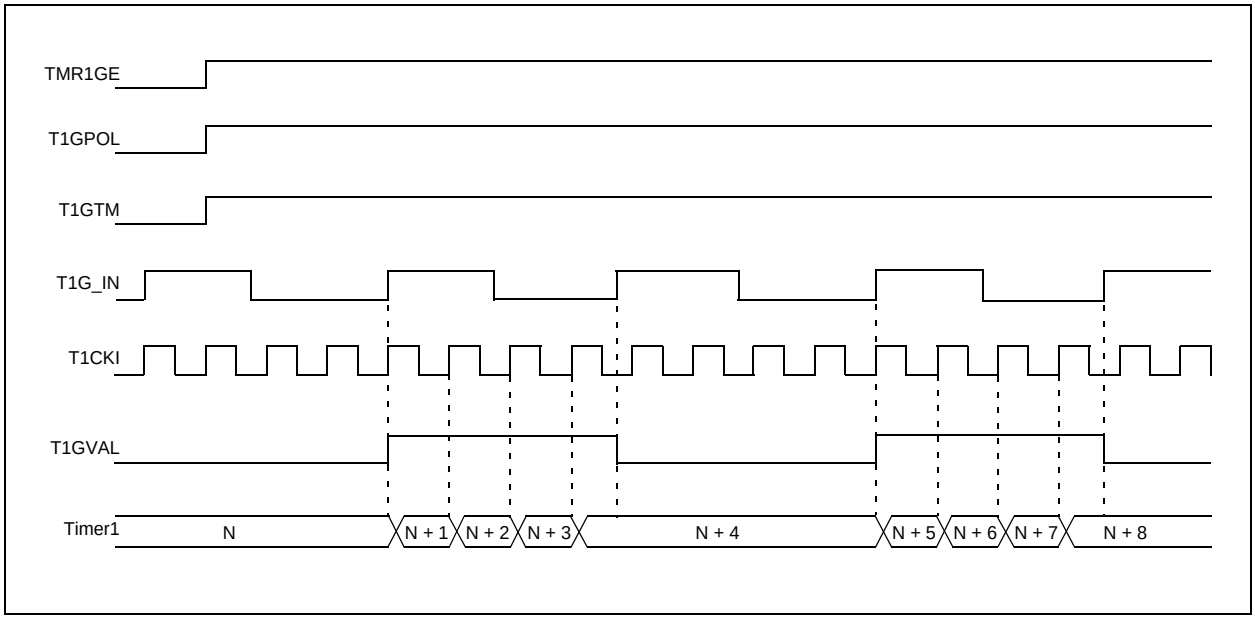
使能 Timer1 门控翻转模式时，可测量 Timer1 门控信号整个周期的长度，而不是单电平脉冲的持续时间。

Timer1 门控源通过单稳态触发器连接，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 14-5。）

T1GVAL 位（T1GCON<2>）指示翻转模式何时工作以及定时器何时计数。

通过将 T1GTM 位 （T1GCON<5>）置 1 使能 Timer1 门控翻转模式。当 T1GTM 清零时，单稳态触发器将清零并保持。这对于控制测量哪个边沿是必需的。

图 14-5： TIMER1 门控翻转模式



14.8.4 TIMER1 门控单脉冲模式

使能 Timer1 门控单脉冲模式时，可能会捕捉到一个单脉冲门控事件。通过将 T1GSPM 位（T1GCON<4>）和 T1GGO/T1DONE 位（T1GCON<3>）置 1 使能 Timer1 门控单脉冲模式。Timer1 将在下一个递增边沿完全使能。

在脉冲的下个后边沿，将自动清零 T1GGO/T1DONE 位。不允许其他门控事件递增 Timer1，直到 T1GGO/T1DONE 位再次用软件置 1。

清零 T1GCON 寄存器的 T1GSPM 位也会清零 T1GGO/T1DONE 位。（时序详细信息请参见图 14-6。）

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 门控源的周期时间。（时序详细信息请参见图 14-7。）

14.8.5 TIMER1 门控值状态

使用 Timer1 门控值状态时，可读取门控控制值的最新电平。该值保存在 T1GVAL 位（T1GCON<2>）中。即使 Timer1 门控未使能（TMR1GE 位清零），该位也是有效的。

图 14-6： TIMER1 门控单脉冲模式

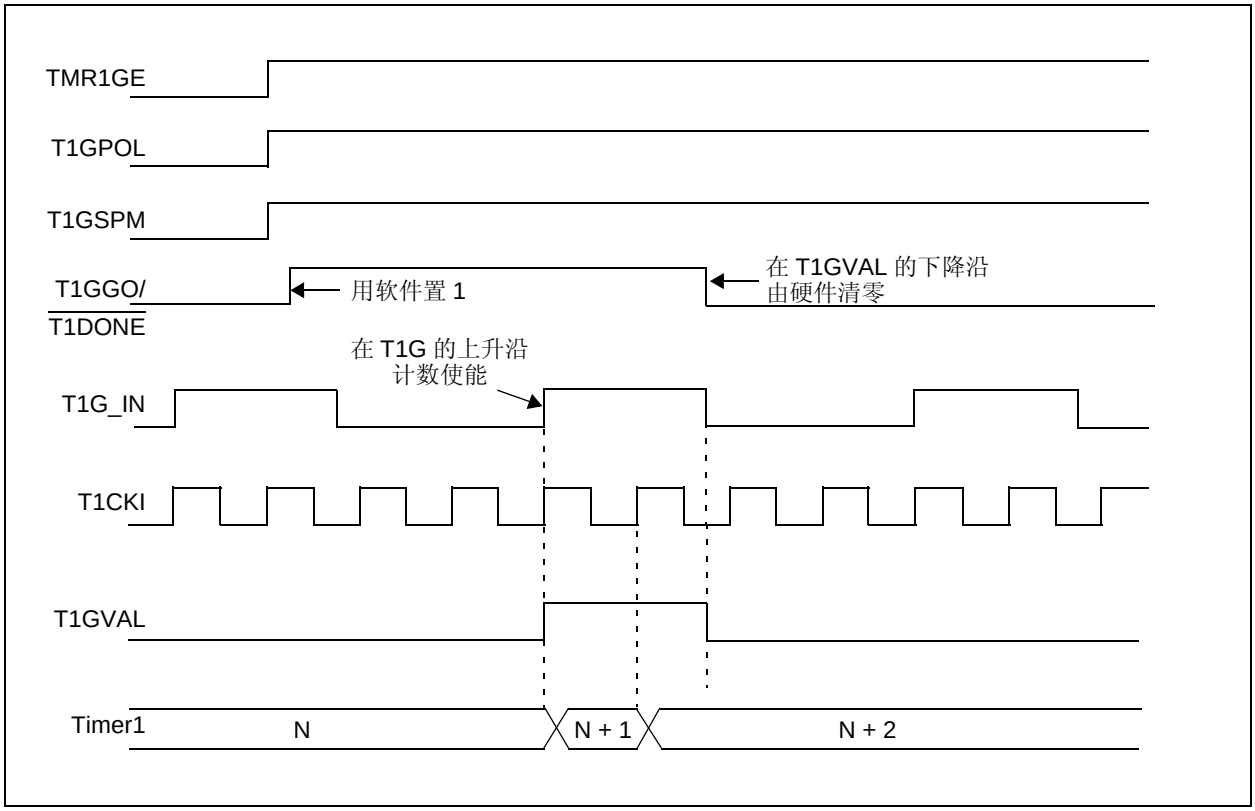


图 14-7：　　TIMER1 门控单脉冲和翻转组合模式

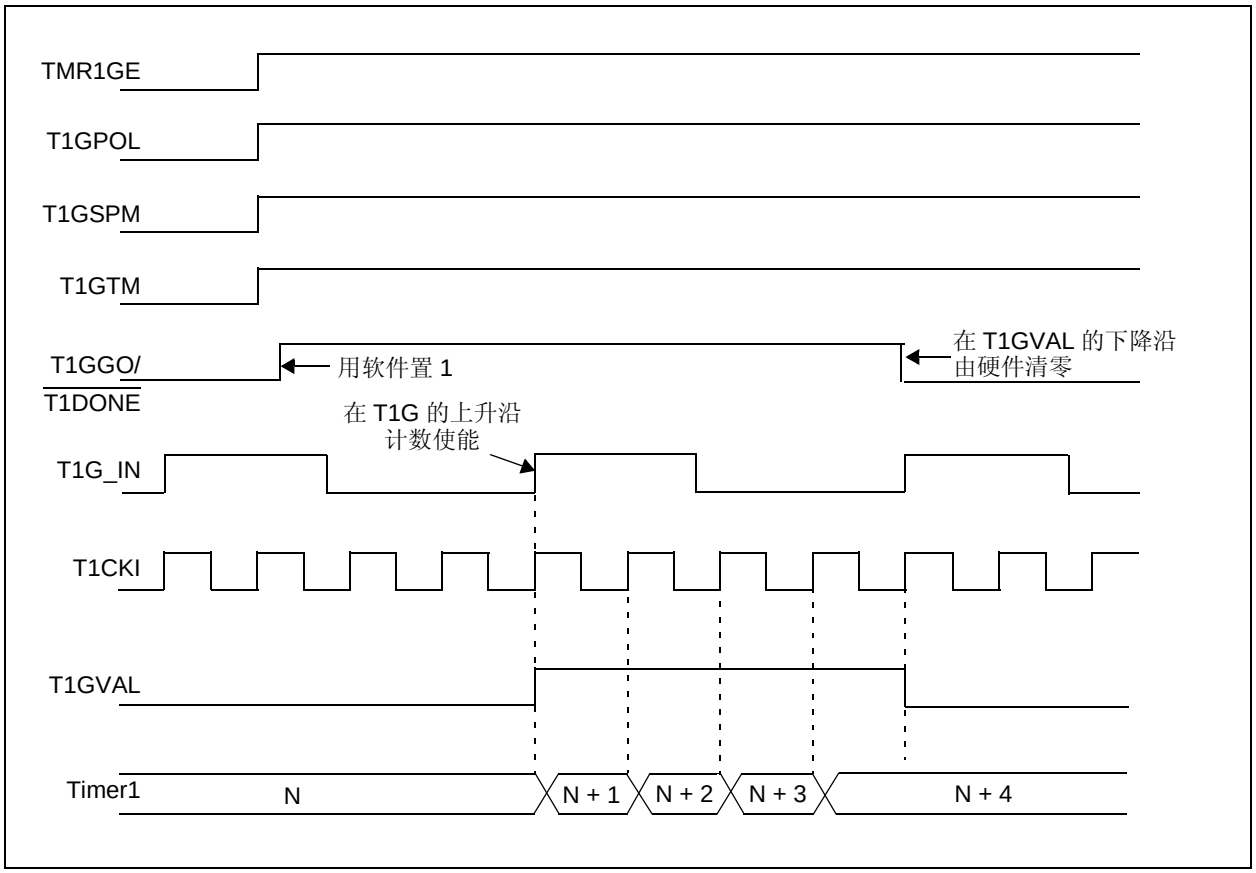


表 14-5：　　与 TIMER1 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
TMR1L	Timer1 寄存器的低字节							
TMR1H	Timer1 寄存器的高字节							
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/ $\overline{\text{T1DONE}}$	T1GVAL	T1GSS1	T1GSS0
OSCCON2	—	SOSCRUN	—	SOSCDRV	SOSCGO	—	MFIOFS	MFIOSEL
PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD

图注：　Timer1 模块不使用阴影单元。

15.0 TIMER2 模块

Timer2 模块具有以下特性：

- 8 位定时器和周期寄存器（分别为 TMR2 和 PR2）
- 两个寄存器均可读写
- 可软件编程的预分频器（分频比为1:1、1:4和1:16）
- 可软件编程的后分频器 （分频比为 1:1 至 1:16）
- TMR2 与 PR2 匹配时产生中断
- 可选择用作 MSSP 模块的移位时钟

该模块由 T2CON 寄存器（寄存器 15-1）控制，该寄存器使能或禁止定时器并配置预分频比和后分频比。可以通过清零控制位 TMR2ON（T2CON<2>）关闭 Timer2，以实现功耗最小。

图 15-1 给出了该模块的简化框图。

15.1 Timer2 工作原理

在正常工作模式下，TMR2 从 00h 开始，每个时钟周期（Fosc/4）递增 1。4 位计数器 / 预分频器提供了直接时钟输入、对时钟输入 4 分频和 16 分频三个预分频选项。这些选项通过预分频比控制位T2CKPS<1:0>（T2CON<1:0>）进行选择。

在每个时钟周期，TMR2 的值都会与周期寄存器 PR2 中的值进行比较。当两个值匹配时，由比较器产生匹配信号作为定时器的输出。该信号也会将 TMR2 的值在下一个周期复位为 00h，并驱动输出计数器 / 后分频器 （见 **第 15.2 节 “Timer2 中断 ”**）。

TMR2 和 PR2 寄存器均可直接读写。在任何器件复位时，TMR2 寄存器都会清零，而 PR2 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 TMR2 寄存器进行写操作
- 对 T2CON 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）

写 T2CON 时 TMR2 不会清零。

> **注：** 对于某些模式，CCP 和 ECCP 模块会使用 Timer1 至 Timer4。要将哪个特定的定时器分配给 CCP/ECCP 模块由 CCPTMRS 寄存器中的 “CCP 的定时器 ” 使能位决定。更多详细信息，请参见寄存器 20-2 和寄存器 19-2。

寄存器 15-1： T2CON：TIMER2 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
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 6-3 **T2OUTPS<3:0>：** Timer2 输出后分频比选择位
0000 = 1:1 后分频比
0001 = 1:2 后分频比
•
•
•
1111 = 1:16 后分频比

bit 2 **TMR2ON：** Timer2 使能位
1 = 使能 Timer2
0 = 关闭 Timer2

bit 1-0 **T2CKPS<1:0>：** Timer2 时钟预分频比选择位
00 = 预分频比为 1
01 = 预分频比为 4
1x = 预分频比为 16

15.2 Timer2 中断

Timer2 也可以产生可选的器件中断。 Timer2 输出信号（TMR2 与 PR2 匹配时）为 4 位输出计数器 / 后分频器提供输入。该计数器产生 TMR2 匹配中断，对应的中断标志位为 TMR2IF （PIR1<1>）。可以通过将 TMR2 匹配中断允许位 TMR2IE（PIE1<1>）置 1 来允许该中断。

可以通过后分频比控制位T2OUTPS<3:0>（T2CON<6:3>）在 16 个后分频比选项 （从 1:1 至 1:16）中选择其一。

15.3 Timer2 输出

TMR2 的未经分频的输出主要用于 ECCP 模块，它用作 ECCP 模块在 PWM 模式下工作时的时基。

还可选择将 Timer2 用作 MSSP 模块在 SPI 模式下工作时的移位时钟源。**第 21.0 节 “ 主同步串行口（MSSP）模块 ”** 中提供了更多信息。

图 15-1： TIMER2 框图

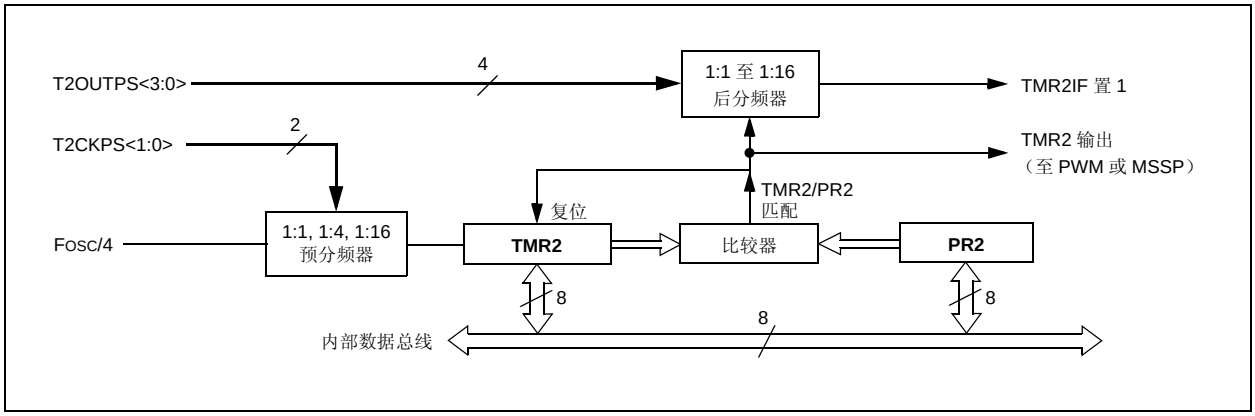


表 15-1： 与 TIMER2 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
TMR2	Timer2 寄存器							
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
PR2	Timer2 周期寄存器							
PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD

图注： — = 未实现，读为 0。 Timer2 模块不使用阴影单元。

16.0 TIMER3 模块

Timer3 定时器 / 计数器模块具有以下特性：

- 可由软件选择作为 16 位定时器或计数器
- 可读写的 8 位寄存器 （TMR3H 和 TMR3L）
- 可选择器件时钟或 SOSC 内部振荡器作为时钟源（内部或外部）
- 溢出时产生中断
- CCP 特殊事件触发模块复位

图 16-1 给出了 Timer3 模块的简化框图。

Timer3 模块是通过 T3CON 寄存器 （寄存器 16-1）来控制的。它还可以为 ECCP 模块选择时钟源。（更多信息，请参见**第 20.1.1 节 “ECCP 模块和定时器资源 ”**。）

Fosc 时钟源不应与 ECCP 捕捉 / 比较功能一起使用。如果捕捉或比较功能需要使用定时器，则应选择其他定时器时钟源选项。

寄存器 16-1： T3CON：TIMER3 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON
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 **TMR3CS<1:0>**：Timer3 时钟源选择位

10 = Timer3 时钟源来自引脚或振荡器，具体取决于 SOSCEN 位的设置：

SOSCEN = 0：

外部时钟来自 T3CKI 引脚 （上升沿计数）。

SOSCEN = 1：

根据 SOSCSEL 配置位的设置，时钟源可以是 SOSCI/SOSCO 上的晶振或来自 SCLKI 引脚的内部数字时钟。

01 = Timer3 时钟源为系统时钟 （Fosc） **(1)**

00 = Timer3 时钟源为指令时钟 （Fosc/4）

bit 5-4 **T3CKPS<1:0>**：Timer3 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 **SOSCEN**：SOSC 振荡器使能位

1 = 使能 SOSC，它可用于 Timer3

0 = 禁止 SOSC，它不可用于 Timer3

bit 2 **$\overline{\text{T3SYNC}}$**：Timer3 外部时钟输入同步控制位

（不适用于器件时钟来自 Timer1/Timer3 的场合。）

当 TMR3CS<1:0> = 10 时：

1 = 不同步外部时钟输入

0 = 同步外部时钟输入

当 TMR3CS<1:0> = 0x 时：

该位被忽略； Timer3 使用内部时钟。

bit 1 **RD16**：16 位读 / 写模式使能位

1 = 使能 Timer3 通过一次 16 位操作进行寄存器读 / 写

0 = 使能 Timer3 通过两次 8 位操作进行寄存器读 / 写

bit 0 **TMR3ON**：Timer3 使能位

1 = 使能 Timer3

0 = 停止 Timer3

注 1： 如果定时器要与 ECCP 捕捉 / 比较功能一起使用，则不应选择 Fosc 时钟源。

16.1 Timer3 门控控制寄存器

Timer3 门控控制寄存器（T3GCON）如寄存器 16-2 所示，用于控制 Timer3 门控。

寄存器 16-2： T3GCON：TIMER3 门控控制寄存器 (1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-x	R/W-0	R/W-0
TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/$\overline{\text{T3DONE}}$	T3GVAL	T3GSS1	T3GSS0
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 **TMR3GE**：Timer3 门控使能位

如果 TMR3ON = 0：
该位被忽略。

如果 TMR3ON = 1：
1 = Timer3 计数由 Timer3 门控功能控制
0 = Timer3 计数与 Timer3 门控功能无关

bit 6 **T3GPOL**：Timer3 门控极性位

1 = Timer3 门控高电平有效 （当门控为高电平时 Timer3 计数）
0 = Timer3 门控低电平有效 （当门控为低电平时 Timer3 计数）

bit 5 **T3GTM**：Timer3 门控翻转模式位

1 = 使能 Timer3 门控翻转模式
0 = 禁止 Timer3 门控翻转模式并清除单稳态触发器的输出
Timer3 门控单稳态触发器在每个上升沿翻转。

bit 4 **T3GSPM**：Timer3 门控单脉冲模式位

1 = 使能 Timer3 门控单脉冲模式，控制 Timer3 门控
0 = 禁止 Timer3 门控单脉冲模式

bit 3 **T3GGO/$\overline{\text{T3DONE}}$**：Timer3 门控单脉冲采集状态位

1 = Timer3 门控单脉冲采集就绪，正在等待一个边沿
0 = Timer3 门控单脉冲采集已经结束或尚未开始
当 T3GSPM 清零时，该位会自动清零。

bit 2 **T3GVAL**：Timer3 门控当前状态位

指示可提供给 TMR3H:TMR3L 的 Timer3 门控信号的当前状态。不受 Timer3 门控使能 （TMR3GE）位的影响。

bit 1-0 **T3GSS<1:0>**：Timer3 门控源选择位

11 = 比较器 2 的输出
10 = 比较器 1 的输出
01 = TMR4 匹配 PR4 输出
00 = Timer3 门控引脚
如果 TMR3GE = 1，则看门狗定时器振荡器开启，与 TMR3ON 的状态无关。

注 1： 建议在设定 T3CON 之前先设定 T3GCON。

寄存器 16-3： OSCCON2：振荡器控制寄存器 2

U-0	R-0	U-0	RW-0	R/W-0	U-0	R-0	R/W-0
—	SOSCRUN	—	SOSCDRV(1)	SOSCGO	—	MFIOFS	MFIOSEL
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 **SOSCRUN：**SOSC 运行状态位
1 = 系统时钟来自辅助 SOSC
0 = 系统时钟来自除 SOSC 外的振荡器

bit 5 **未实现：**读为 0

bit 4 **SOSCDRV：**辅助振荡器驱动控制位 **(1)**
1 = 选择高功耗 SOSC 电路
0 = 通过 SOSCSEL<1:0> 配置位选择低 / 高功耗

bit 3 **SOSCGO：**振荡器启动控制位
1 = 即使没有任何其他时钟源请求使用振荡器，振荡器仍保持运行
0 = 如果没有任何其他时钟源请求使用振荡器，则关闭振荡器（选择通过数字时钟输入而不是外部晶振来运行 SOSC 时，该位不起作用。）

bit 2 **未实现：**读为 0

bit 1 **MFIOFS：**MF-INTOSC 频率稳定位
1 = MF-INTOSC 稳定
0 = MF-INTOSC 不稳定

bit 0 **MFIOSEL：**MF-INTOSC 选择位
1 = 使用 MF-INTOSC 来替代 HF-INTOSC 频率 500 kHz、 250 kHz 和 31.25 kHz
0 = 不使用 MF-INTOSC

注 1： 当选择从数字时钟输入而不是外部晶振来运行 SOSC 时，该位不起作用。

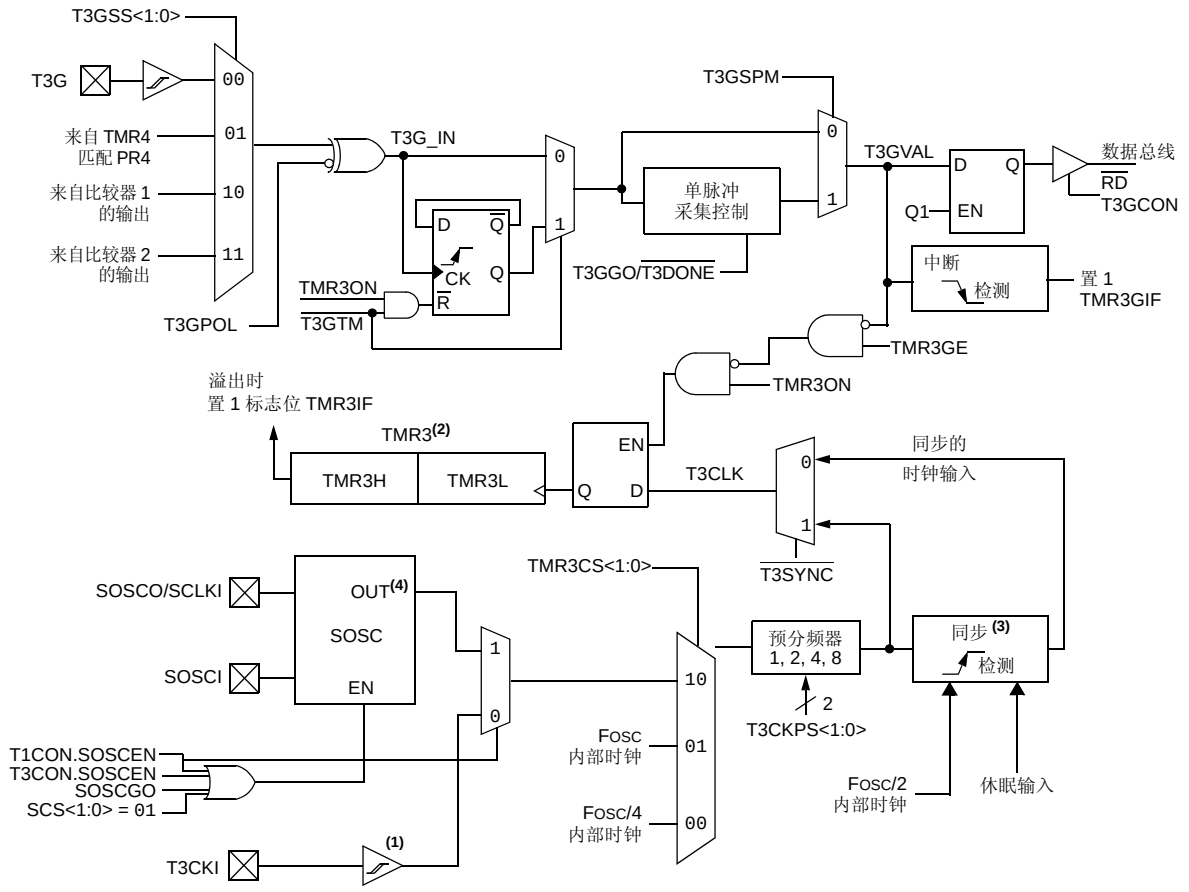
16.2 Timer3 工作原理

Timer3 可在以下模式下工作：

- 定时器
- 同步计数器
- 异步计数器
- 带门控的定时器

工作模式由时钟选择位 TMR3CSx（T3CON<7:6>）决定。当 TMR3CSx 位清零（= 00）时，Timer3 在每个内部指令周期（FOSC/4）递增。当 TMR3CSx = 01 时，Timer3 时钟源为系统时钟（FOSC）；当 TMR3CSx = 10 时， Timer3 用作计数器，采用来自 T3CKI 引脚 （在第一个下降沿之后的上升沿）或 SOSC 振荡器的外部时钟。

图 16-1： TIMER3 框图



注 1： 当使用 T3CKI 时， ST 缓冲器为高速型。
2： Timer3 寄存器在上升沿递增。
3： 处于休眠模式时同步不工作。
4： SOSC 的输出由 SOSCSEL<1:0> 配置位决定。

16.3 Timer3 的 16 位读 / 写模式

可将 Timer3 配置为 16 位读写模式。当 RD16 控制位（T3CON<1>）置1时，TMR3H的地址被映射到Timer3的高字节缓冲寄存器。读 TMR3L 会将 Timer3 的高字节的内容装入 Timer3 高字节缓冲寄存器。这种方式使用户可以精确地读取 Timer3 的全部 16 位，而不需要像先读高字节再读低字节那样，由于两次读取之间可能存在进位，而不得不验证读取的有效性。

写 Timer3 的高字节也必须通过 TMR3H 缓冲寄存器进行。在写入 TMR3L 的同时，使用 TMR3H 的内容更新 Timer3 的高字节。这样允许用户将所有 16 位一次写入 Timer3 的高字节和低字节。

在该模式下不能直接读写 Timer3 的高字节。所有读写都必须通过 Timer3 高字节缓冲寄存器来进行。

写入 TMR3H 不会清零 Timer3 预分频器。只有在写 TMR3L 时才会清零该预分频器。

16.4 使用 SOSC 振荡器作为 Timer3 的时钟源

SOSC 内部振荡器可用作 Timer3 的时钟源。它可以使用以下方法之一使能：

- 将 T1CON 或 T3CON 寄存器中的 SOSCEN 位（TxCON<3>）置 1
- 将 OSCCON2 寄存器中的 SOSCGO 位（OSCCON2<3>）置 1
- 将 OSCCON 寄存器中的 SCS 位设置为辅助时钟源（OSCCON<1:0> = 01）

SOSCGO 位用于预热 SOSC，使之在任意外设发出请求之前就绪。

要将它用作 Timer3 的时钟源，还必须将 TMR3CSx 位置 1。如前文所述，这样做也会将 Timer3 配置为在振荡器源的每个上升沿递增。

在**第 14.5 节 “SOSC 振荡器 ”** 中对 SOSC 振荡器进行了说明。

16.5 Timer3 门控

Timer3 可配置为自由计数或用 Timer3 门控电路使能和禁止计数。这也称为 Timer3 门控计数使能。

Timer3 门控也可由多个可选择源驱动。

16.5.1 TIMER3 门控计数使能

通过将 TMR3GE 位（TxGCON<7>）置 1 使能 Timer3 门控使能模式。使用 T3GPOL 位（T3GCON<6>）来配置 Timer3 门控使能模式的极性。

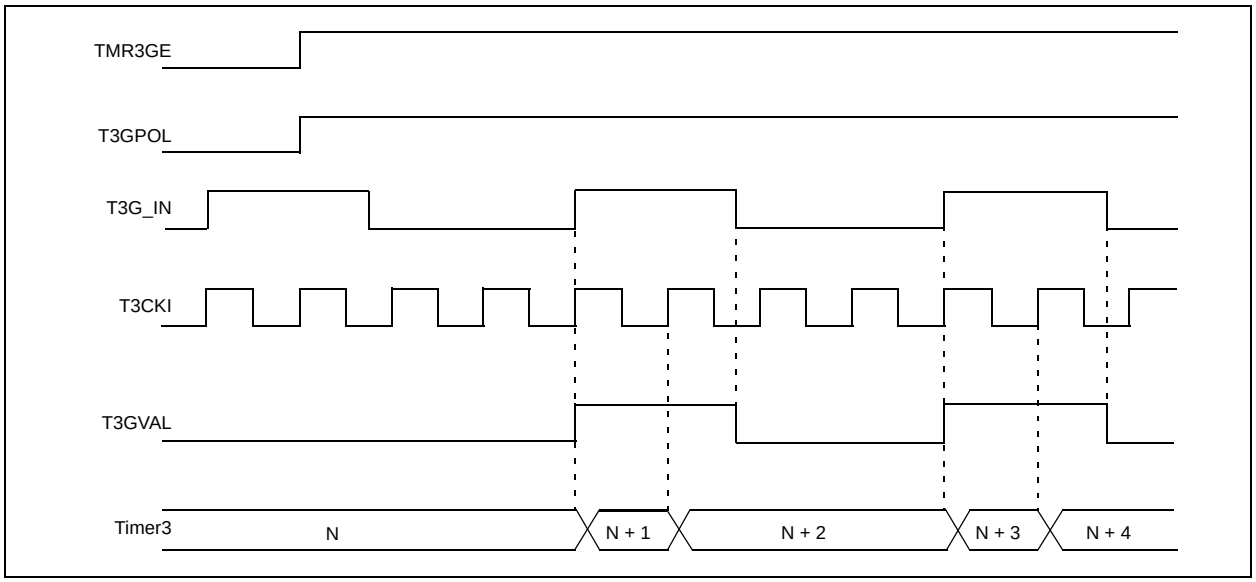
使能 Timer3 门控使能模式时，Timer3 将在 Timer3 时钟源的上升沿递增。禁止 Timer3 门控使能模式时，不会发生递增， Timer3 将保持当前计数。时序详细信息请参见图 16-2。

表 16-1： TIMER3 门控使能选择

T3CLK(†)	T3GPOL（T3GCON<6>）	T3G 引脚	Timer3 工作状态
↑	0	0	计数
↑	0	1	保持计数
↑	1	0	保持计数
↑	1	1	计数

† TMR3 运行所使用的时钟。更多信息，请参见图 16-1 中的 T3CLK。

图 16-2： TIMER3 门控计数使能模式



16.5.2 TIMER3 门控源选择

Timer3 门控源可从四种不同源之中选择。门控源的选择由 T3GSS<1:0> 位 （T3GCON<1:0>）控制。每个可用源的极性也是可选择的，由T3GPOL位（T3GCON<6>）控制。

表 16-2： TIMER3 门控源

T3GSS<1:0>	Timer3 门控源
00	Timer3 门控引脚
01	TMR4 匹配 PR4 （TMR4 递增以匹配 PR4）
10	比较器 1 的输出 （比较器逻辑高电平输出）
11	比较器 2 的输出 （比较器逻辑高电平输出）

16.5.2.1 T3G 引脚门控操作

T3G引脚是Timer3门控控制源之一。它可用于向Timer3门控电路提供外部源。

16.5.2.2 Timer4 匹配门控操作

TMR4 寄存器将递增到与 PR4 寄存器中的值匹配为止。在紧接着的下一个递增周期， TMR4 将复位为 00h。发生该复位时，将自动产生由低至高脉冲并在内部提供给 Timer3 门控电路。脉冲将保持一个指令周期的高电平，然后恢复为低电平状态，直到再一次发生匹配。

根据 T3GPOL 的设置，Timer3 在 TMR4 与 PR4 发生匹配时的递增形式会不同。当 T3GPOL = 1 时，Timer3 在 TMR4 与 PR4 发生匹配之后的单个指令周期中递增。当 T3GPOL = 0 时，发生匹配后的一个周期以后， Timer3 会在门控信号从低电平变为高电平时不断递增。

16.5.2.3 比较器 1 输出门控操作

比较器 1 的输出可以在内部送至 Timer3 门控电路。使用 CM1CON 寄存器设置比较器 1 之后，Timer3 会根据 CMP1OUT （CMSTAT<6>）位的电平变化进行递增。

16.5.2.4 比较器 2 输出门控操作

比较器 2 的输出可以在内部送至 Timer3 门控电路。使用 CM2CON 寄存器设置比较器 2 之后，Timer3 会根据 CMP2OUT （CMSTAT<7>）位的电平变化进行递增。

16.5.3 TIMER3 门控翻转模式

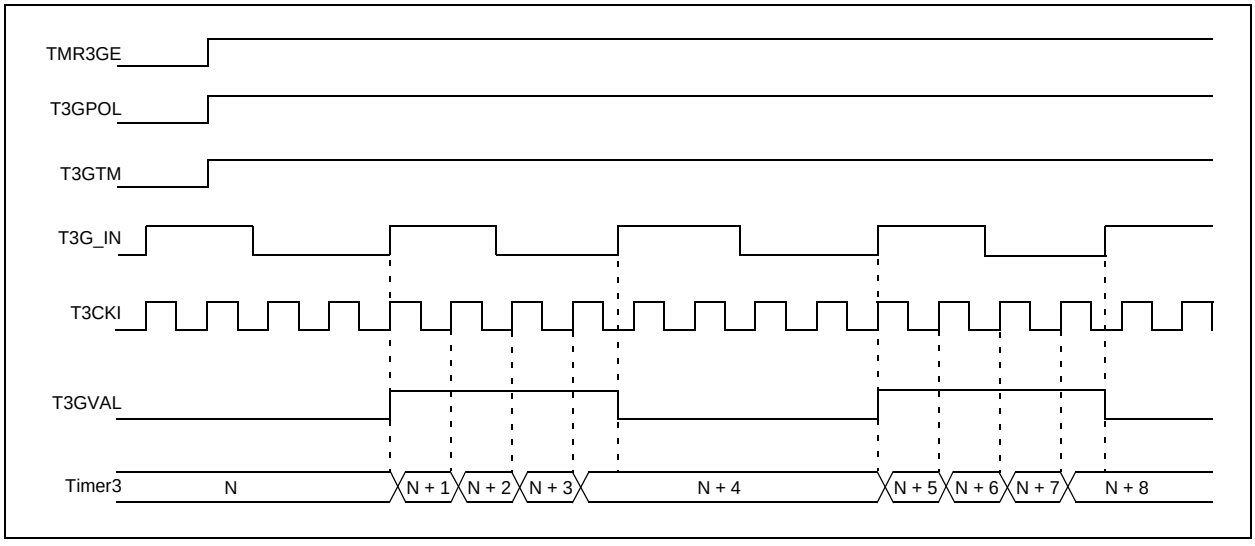
使能 Timer3 门控翻转模式时，可测量 Timer3 门控信号整个周期的长度，而不是单电平脉冲的持续时间。

Timer3 门控源通过单稳态触发器连接，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 16-3。）

T3GVAL 位将指示翻转模式何时工作以及定时器何时计数。

通过将 T3GTM 位 （T3GCON<5>）置 1 使能 Timer3 门控翻转模式。当 T3GTM 位清零时，单稳态触发器将清零并保持。这对于控制测量哪个边沿是必需的。

图 16-3： TIMER3 门控翻转模式



16.5.4 TIMER3 门控单脉冲模式

使能 Timer3 门控单脉冲模式时，可能会捕捉到一个单脉冲门控事件。首先通过将 T3GSPM 位（T3GCON<4>）置 1 使能 Timer3 门控单脉冲模式。接下来必须将 T3GGO/$\overline{\text{T3DONE}}$ 位 （T3GCON<3>）置 1。

Timer3 将在下一个递增边沿完全使能。在脉冲的下个后边沿，将自动清零 T3GGO/$\overline{\text{T3DONE}}$ 位。不允许其他门控事件递增 Timer3，直到 T3GGO/$\overline{\text{T3DONE}}$ 位再次用软件置 1。

清零 T3GSPM 位也将清零 T3GGO/$\overline{\text{T3DONE}}$ 位。（时序详细信息请参见图 16-4。）

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 Timer3 门控源的周期时间。（时序详细信息请参见图 16-5。）

图 16-4： TIMER3 门控单脉冲模式

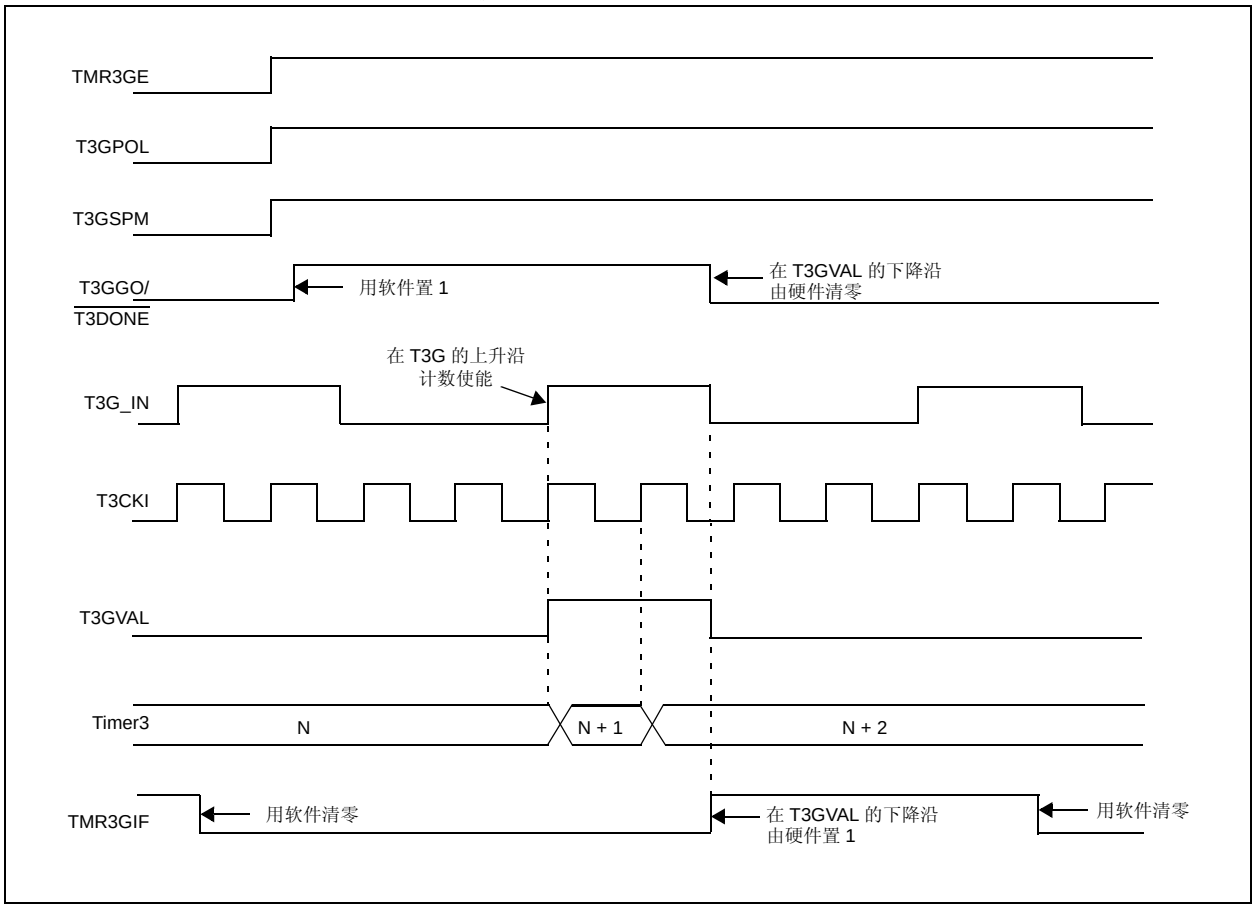
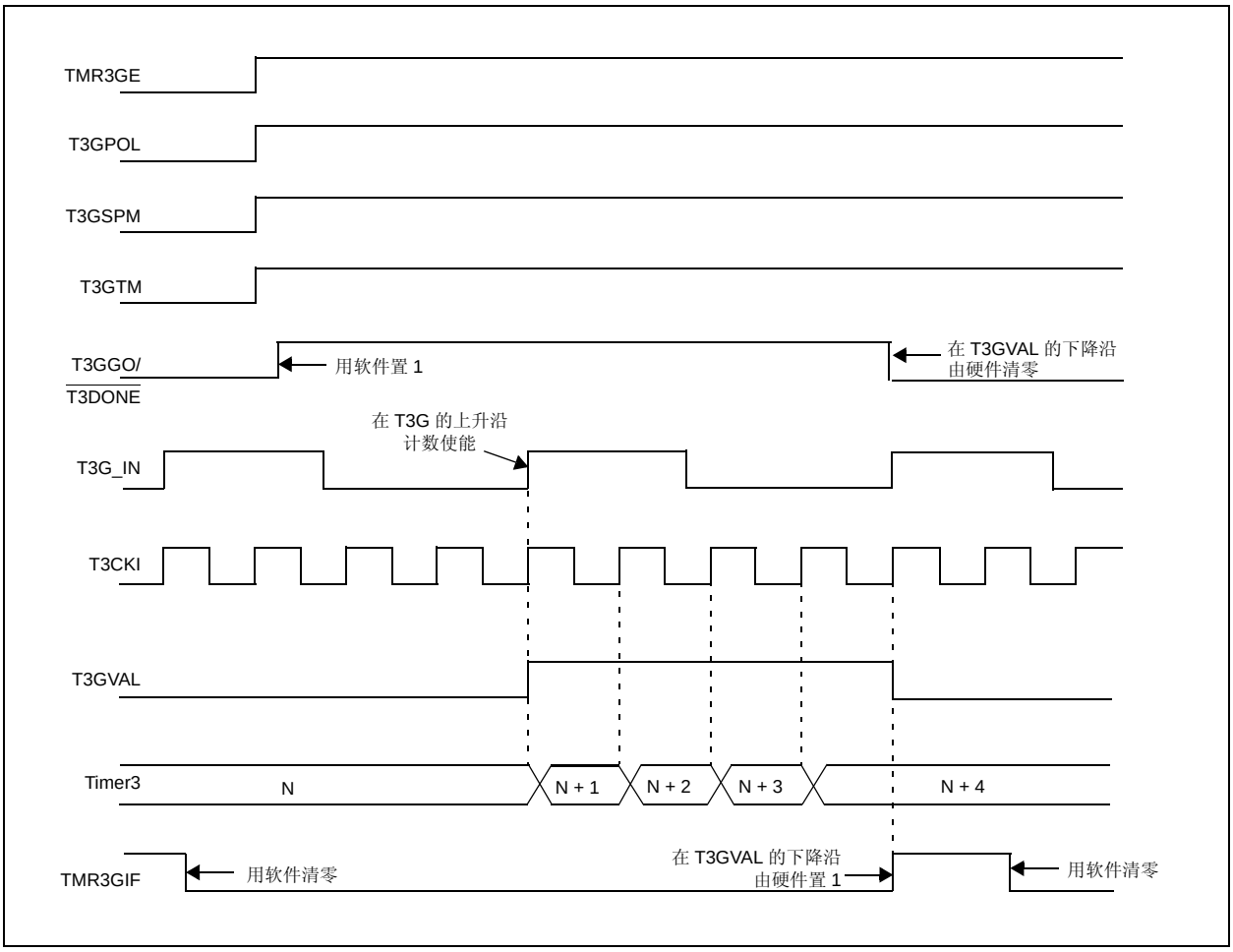


图 16-5：　　TIMER3 门控单脉冲和翻转组合模式



16.5.5　　TIMER3 门控值状态

使用 Timer3 门控值状态时，可读取门控控制值的最新电平。该值保存在 T3GVAL 位（T3GCON<2>）中。即使 Timer3 门控未使能（TMR3GE 位清零），T3GVAL 位也是有效的。

16.5.6　　TIMER3 门控事件中断

允许 Timer3 门控事件中断时，可在门控事件完成时产生一个中断。出现 T3GVAL 的下降沿时， PIR3 寄存器中的 TMR3GIF 标志位将置 1。如果 PIE3 寄存器中的 TMR3GIE 位置 1，则会响应中断。

即使 Timer3 门控未使能（TMR3GE 位清零），TMR3GIF 标志位也能工作。

16.6 Timer3 中断

TMR3 寄存器对（TMR3H:TMR3L）从 0000h 递增到 FFFFh，然后计满返回到 0000h 重新开始计数。如果允许了 Timer3 中断，则溢出时会产生 Timer3 中断，并反映到中断标志位 TMR3IF 中。表 16-3 列出了每个模块的中断标志位。

可以通过将 TMR3IE 位置 1 或清零来允许或禁止该中断。表 16-3 列出了每个模块的中断允许位。

16.7 使用 ECCP 特殊事件触发信号复位 Timer3

如果 ECCP 模块配置为使用 Timer3 并在比较模式（CCP3M<3:0> = 1011）下产生特殊事件触发信号，该信号将复位 Timer3。如果使能了 A/D 模块，来自 ECCP 的触发信号还将启动 A/D 转换（更多信息，请参见**第 20.3.4 节 “ 特殊事件触发器 ”**）。

要使用这一功能，必须将模块配置为定时器或同步计数器。在这种情况下， CCPR3H:CCPR3L 寄存器对实际上变成了 Timer3 的周期寄存器。

如果 Timer3 在异步计数器模式下运行，复位操作可能不起作用。

如果 Timer3 的写操作和来自 ECCP 模块的特殊事件触发同时发生，则写操作优先。

注：	ECCPx模块产生的特殊事件触发信号只会清除 TMR3 寄存器的内容，而不会将 TMR3IF 中断标志位（PIR3<1>）置 1。

注：	对于某些模式，CCP 和 ECCP 模块会使用 Timer1 至 Timer4。要将哪个特定的定时器分配给 CCP/ECCP 模块由 CCPTMRS 寄存器中的 “CCP 的定时器 ” 使能位决定。更多详细信息，请参见寄存器 20-2 和寄存器 19-2。

表 16-3： 与 TIMER3 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR5	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF	TXB0IF	RXB1IF	RXB0IF
PIE5	IRXIE	WAKIE	ERRIE	TX2BIE	TXB1IE	TXB0IE	RXB1IE	RXB0IE
PIR2	OSCFIF	—	—	—	BCLIF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	—	—	BCLIE	HLVDIE	TMR3IE	TMR3GIE
TMR3H	Timer3 寄存器的高字节							
TMR3L	Timer3 寄存器的低字节							
T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/$\overline{\text{T3DONE}}$	T3GVAL	T3GSS1	T3GSS0
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON
OSCCON2	—	SOSCRUN	—	SOSCDRV	SOSCGO	—	MFIOFS	MFIOSEL
PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD

图注： — = 未实现，读为 0。 Timer3 模块不使用阴影单元。

17.0 TIMER4 模块

Timer4 定时器模块具有以下特性：

- 8 位定时器寄存器 （TMR4）
- 8 位周期寄存器 （PR4）
- 可读写 （以上两个寄存器）
- 可软件编程的预分频器（分频比为1:1、1:4和1:16）
- 可软件编程的后分频器 （分频比为 1:1 至 1:16）
- TMR4 与 PR4 匹配时产生中断

Timer4 模块具有如寄存器 17-1 所示的控制寄存器。可以通过清零控制位 TMR4ON（T4CON<2>）关闭 Timer4，以实现功耗最小。该寄存器还控制对 Timer4 的预分频比和后分频比的选择。图 17-1 给出了 Timer4 模块的简化框图。

17.1 Timer4 工作原理

Timer4 可以用作 ECCP 模块在 PWM 模式下的 PWM 时基。TMR4 寄存器是可读写的，在任何器件复位时都会被清零。输入时钟 （Fosc/4）具有一个由控制位 T4CKPS<1:0>（T4CON<1:0>）选择的预分频选项（1:1、1:4 或 1:16）。TMR4 的匹配输出通过一个 4 位后分频器 （提供 1:1 至 1:16 的分频比）产生 TMR4 中断，并反映到标志位 TMR4IF 中。表 17-1 列出了每个模块的中断标志位。

可以通过将 Timer4 中断允许位（TMR4IE）置 1 或清零来允许或禁止该中断。

发生以下事件时，预分频器和后分频器计数器均会清零：

- 对 TMR4 寄存器进行写操作
- 对 T4CON 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）

写 T4CON 时 TMR4 不会清零。

注： 对于某些模式，CCP 和 ECCP 模块会使用 Timer1 至 Timer4。要将哪个特定的定时器分配给 CCP/ECCP 模块由 CCPTMRS 寄存器中的 “CCP 的定时器 ” 使能位决定。更多详细信息，请参见寄存器 20-2 和寄存器 19-2。

寄存器 17-1： T4CON：TIMER4 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
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 6-3 **T4OUTPS<3:0>：** Timer4 输出后分频比选择位
0000 = 1:1 后分频比
0001 = 1:2 后分频比
•
•
•
1111 = 1:16 后分频比

bit 2 **TMR4ON：** Timer4 使能位
1 = 使能 Timer4
0 = 关闭 Timer4

bit 1-0 **T4CKPS<1:0>：** Timer4 时钟预分频比选择位
00 = 预分频比为 1
01 = 预分频比为 4
1x = 预分频比为 16

17.2 Timer4 中断

Timer4 模块具有一个 8 位周期寄存器 PR4，该寄存器是可读写的。Timer4 从 00h 开始递增直到与 PR4 匹配为止，然后在下一个递增周期复位为 00h。在复位时，PR4 寄存器初始化为 FFh。

17.3 TMR4 的输出

TMR4 的输出（在通过后分频器之前）只能用作 ECCP 模块的 PWM 时基。它们不能像 Timer2 输出一样用作 MSSP 模块的波特率时钟。

图 17-1： TIMER4 框图

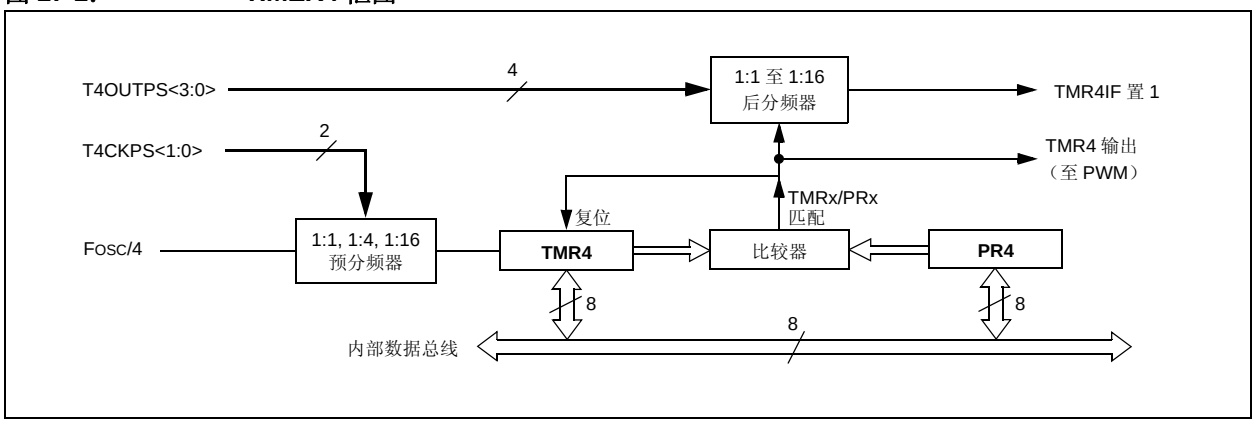


表 17-1： 与 TIMER4 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIE4	TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE
TMR4	Timer4 寄存器							
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
PR4	Timer4 周期寄存器							
PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD

图注： — = 未实现，读为 0。 Timer4 模块不使用阴影单元。

18.0 充电时间测量单元（CTMU）

充电时间测量单元（CTMU）是一个灵活的模拟模块，它提供脉冲源之间的精确时间差测量，以及异步脉冲生成功能。 CTMU 可与其他片上模拟模块配合使用，精确测量时间、电容、电容的相对变化，或生成具有特定延时的输出脉冲。 CTMU 是连接电容式传感器的理想选择。

该模块具有以下主要特性：

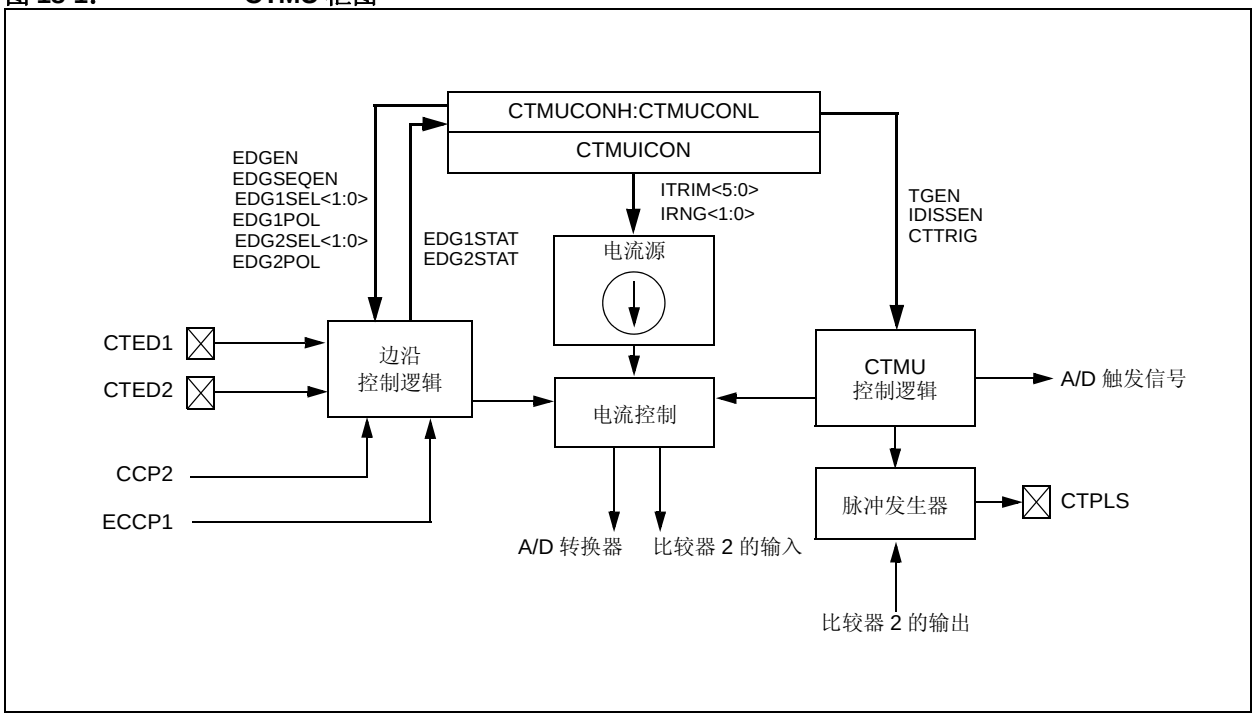
- 最多 11 路通道，可用于电容或时间测量输入
- 使用片上二极管通道进行低成本温度测量
- 片上精确电流源
- 4 个边沿输入触发源
- 每个边沿源的极性控制
- 边沿顺序控制
- 控制对边沿的响应
- 时间测量分辨率为 1 纳秒
- 高精度时间测量
- 与系统时钟异步的外部或内部信号延时
- 适合电容测量的精确电流源

CTMU 与 A/D 转换器配合工作，根据具体器件和可用的 A/D 通道数，最多可提供 11 路通道用于时间或电荷测量。如果配置为产生延时，那么 CTMU 连接到其中一个模拟比较器。电平敏感输入边沿源有 4 种可供选择：两个外部输入或 ECCP1/CCP2 特殊事件触发器。

CTMU 特殊事件能触发模数转换器模块。

图 18-1 给出了 CTMU 的框图。

图 18-1： CTMU 框图



18.1 CTMU 寄存器

CTMU 的控制寄存器有：

- CTMUCONH
- CTMUCONL
- CTMUICON

CTMUCONH 和 CTMUCONL 寄存器（寄存器 18-1 和寄存器 18-2）包含一些控制位，这些控制位用于配置 CTMU 模块边沿源选择、边沿源极性选择、边沿顺序、A/D 触发、模拟电路电容放电和使能。 CTMUICON 寄存器（寄存器 18-3）包含一些用于选择电流源范围和电流源微调的位。

寄存器 18-1： CTMUCONH：CTMU 控制高字节寄存器

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG
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 **CTMUEN：**CTMU 使能位
1 = 使能模块
0 = 禁止模块

bit 6 **未实现：**读为 0

bit 5 **CTMUSIDL：**空闲模式停止位
1 = 当器件进入空闲模式时，模块停止工作
0 = 在空闲模式下模块继续工作

bit 4 **TGEN：**延时产生使能位
1 = 使能边沿延时产生
0 = 禁止边沿延时产生

bit 3 **EDGEN：**边沿使能位
1 = 未阻止边沿
0 = 阻止边沿

bit 2 **ESGSEQEN：**边沿顺序使能位
1 = 边沿 1 事件必须在边沿 2 事件发生前发生
0 = 无需边沿顺序

bit 1 **IDISSEN：**模拟电流源控制位
1 = 模拟电流源输出接地
0 = 模拟电流源输出未接地

bit 0 **CTTRIG：**CTMU 特殊事件触发位
1 = 使能 CTMU 特殊事件触发器
0 = 禁止 CTMU 特殊事件触发器

寄存器 18-2： CTMUCONL：CTMU 控制低字节寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT
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 **EDG2POL：**边沿 2 极性选择位
1 = 边沿 2 设定为正边沿响应
0 = 边沿 2 设定为负边沿响应

bit 6-5 **EDG2SEL<1:0>：**边沿 2 源选择位
11 = CTED1 引脚
10 = CTED2 引脚
01 = ECCP1 特殊事件触发信号
00 = CCP2 特殊事件触发信号

bit 4 **EDG1POL：**边沿 1 极性选择位
1 = 边沿 1 设定为正边沿响应
0 = 边沿 1 设定为负边沿响应

bit 3-2 **EDG1SEL<1:0>：**边沿 1 源选择位
11 = CTED1 引脚
10 = CTED2 引脚
01 = ECCP1 特殊事件触发信号
00 = CCP2 特殊事件触发信号

bit 1 **EDG2STAT：**边沿 2 状态位
1 = 已发生边沿 2 事件
0 = 未发生边沿 2 事件

bit 0 **EDG1STAT：**边沿 1 状态位
1 = 已发生边沿 1 事件
0 = 未发生边沿 1 事件

寄存器 18-3： CTMUICON：CTMU 电流控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0
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 **ITRIM<5:0>：**电流源微调位

011111 = 对标称电流的最大正向调整 （典型值为 +62%）
011110
.
.
.
000001 = 对标称电流的最小正向调整 （典型值为 +2%）
000000 = IRNG<1:0> 指定的标称电流输出
111111 = 对标称电流的最小负向调整 （典型值为 -2%）
.
.
.
100010
100001 = 对标称电流的最大负向调整 （典型值为 -62%）

bit 1-0 **IRNG<1:0>：**电流源范围选择位

11 = 100 x 基本电流
10 = 10 x 基本电流
01 = 基本电流 （标称值为 0.55 μA）
00 = 禁止电流源

18.2 CTMU 工作原理

CTMU 的工作方式是使用固定电流源来对电路进行充电。电路的类型取决于要进行的测量的类型。

在进行电荷测量的情况下，电流是固定的，向电路施加电流的时间也是固定的。只要通过 A/D 测得电压就可以测得电路的电容。

在进行时间测量的情况下，电流和电路的电容都是固定的。这种情况下，由 A/D 读取的电压可以代表从电流源开始对电路进行充电到停止充电经过的时间。

如果 CTMU 用于产生延时，那么电容和电流源，以及向比较器电路提供的电压都是固定的。信号的延时由将电压充电到比较器门限电压所需的时间决定。

18.2.1 工作原理

CTMU 的工作原理基于以下电荷公式：

$$I = C \bullet \frac{dV}{dT}$$

简单来说，在电路中测量的电荷（以库仑为单位）定义为：以安培为单位的电流（I）乘以以秒为单位的电流流动时间（t）。电荷也可以定义为：以法拉为单位的电容（C）乘以电路的电压（V）。可得：

$$I \bullet t = C \bullet V$$

CTMU 模块提供了恒定、已知的电流源。A/D 转换器用于测量公式中的电压（V），剩下两个未知量：电容（C）和时间（t）。以上公式可用于计算电容或时间，根据以下关系，使用电路的已知固定电容：

$$t = (C \bullet V)/I$$

或根据：

$$C = (I \bullet t)/V$$

使用电流源施加于电路的固定时间。

18.2.2 电流源

CTMU 的核心是精确电流源，旨在提供用于测量的恒定基准。用户可以从三个范围或总共两个数量级的电流中选择电流等级，并可以按 ±2% 的增量（标称值）对输出进行微调。电流范围通过 IRNG<1:0> 位（CTMUICON<1:0>）进行选择，值 01 代表最低范围。电流微调通过 ITRIM<5:0> 位（CTMUICON<7:2>）进行。这 6 个位使得可以按大约每步 2% 的步阶微调电流源。其中一半的范围用于正向调整电流源，另一半用于负向调整电流源。值 000000 是中性位置（无变化）。值 100001 代表最大负调整（大约 -62%），011111 代表最大正调整（大约 +62%）。

18.2.3 边沿选择和控制

CTMU 测量通过在模块的两路输入通道中发生的边沿事件进行控制。每路通道（称为边沿 1 和边沿 2）可以配置为接收来自一个边沿输入引脚（CTED1 和 CTED2）或 CCPx 特殊事件触发器（ECCP1 和 CCP2）的输入脉冲。输入通道是电平敏感的，响应通道中的瞬时电平，而不是电平跳变。输入使用 EDG1SEL 和 EDG2SEL 位（CTMUCONL<3:2> 和 6:5>）选择。

除了电流源之外，还可以使用 EDGE2POL 和 EDGE1POL 位（CTMUCONL<7,4>）配置每路通道的事件极性。还可以对输入通道进行过滤以选择边沿事件顺序（边沿 1 在边沿 2 之前发生），方法是将 EDGSEQEN 位（CTMUCONH<2>）置 1。

18.2.4 边沿状态

CTMUCONL 寄存器还包含两个状态位 EDG2STAT 和 EDG1STAT（CTMUCONL<1:0>）。它们的主要功能是显示在相应的通道中是否发生了边沿响应。当在通道中检测到边沿响应时，CTMU 会自动将特定的位置 1。输入通道的电平敏感特性也意味着，如果通道的配置发生改变并且与其电流状态匹配，那么状态位会立即置 1。

模块使用边沿状态位来控制到外部模拟模块（如 A/D 转换器）的电流源输出。只有其中一个状态位置 1 而不是两个状态位同时置 1 时，才会向外部模块提供电流。如果两个位同时置 1 或同时清零，则会切断电流。这使 CTMU 可以仅测量两个边沿事件之间的电流。在两个状态位都置 1 后，必须先将它们清零，然后才能进行另一次测量。两个位应同时清零（如果可能），以避免重新使能 CTMU 电流源。

除了可以由 CTMU 硬件置 1 之外，边沿状态位也可以用软件置 1。这允许在用户应用程序中手动使能或禁止电流源。将其中任意一位置 1（但不是同时置 1）即可使能电流源。将两位同时置1或清零即可立即禁止电流源。

18.2.5 中断

每当电流源先使能，然后禁止时，CTMU 就会将其中断标志位（PIR3<3>）置 1。只有相应的中断允许位（PIE3<3>）也置 1 时，才会产生中断。如果未使能边沿顺序（即，边沿 1 必须在边沿 2 之前发生），则需要监视边沿状态位，确定上次发生并导致中断的是哪一个边沿事件。

18.3 CTMU 模块初始化

以下过程是用于初始化 CTMU 模块的一般原则：

1. 使用 IRNGx 位（CTMUICON<1:0>）选择电流源范围。
2. 使用 ITRIMx 位（CTMUICON<7:2>）微调电流源。
3. 通过设置 EDG1SEL 和 EDG2SEL 位（分别为 CTMUCONL<3:2> 和 <6:5>）配置边沿 1 和边沿 2 的边沿输入源。
4. 使用EDG1POL和EDG2POL位（CTMUCONL<4,7>）配置边沿输入的输入极性。

   默认配置是使用负边沿极性 （从高至低跳变）。
5. 使用 EDGSEQEN 位（CTMUCONH<2>）使能边沿顺序。

   默认情况下，将禁止边沿顺序。
6. 使用 TGEN 位（CTMUCONH<4>）选择工作模式 （测量或产生延时）。

   默认模式是时间 / 电容测量。
7. 使用 CTTRIG 位（CTMUCONH<0>）将模块配置为在发生第二个边沿事件时自动触发 A/D 转换。

   默认情况下，会禁止转换触发。
8. 通过将 IDISSEN 位（CTMUCONH<1>）置 1，对所连接电路放电。
9. 在等待足够时间，让电路完成放电之后，清零 IDISSEN 位。
10. 通过清零 CTMUEN 位（CTMUCONH<7>）禁止该模块。
11. 清零边沿状态位 EDG2STAT 和 EDG1STAT（CTMUCONL<1:0>）。

    两个位应同时清零（如果可能），以避免重新使能 CTMU 电流源。
12. 通过将 EDGEN 位（CTMUCONH<3>）置 1 使能两个边沿输入。
13. 通过将 CTMUEN 位置 1 使能该模块。

根据要执行的测量或脉冲生成的类型，可能还需要再初始化和配置一个或更多其他模块，与 CTMU 模块配合使用：

- 边沿源生成：除了外部边沿输入引脚之外，ECCP1/CCP2 特殊事件触发器也可以用作 CTMU 的边沿源。
- 电容或时间测量：CTMU 模块使用 A/D 转换器来测量连接到一路模拟输入通道的电容两端的电压。
- 脉冲生成：在生成独立于系统时钟的输出脉冲时，CTMU 模块使用比较器 2 和关联的比较器参考电压。

18.4 校准 CTMU 模块

要精确测量电容和时间，以及产生精确延时，需要对 CTMU 进行校准。如果应用只需要测量电容或时间的相对变化，则通常不需要校准。精度较低的应用的示例包括电容式触摸开关，在这种应用中，触摸电路具有基本电容，所增加的人体电容会改变电路的总电容。

如果需要测量实际的电容或时间，则必须进行两项硬件校准：

- 电流源需要进行校准，以使其提供精确的电流。
- 要测量的电路也需要进行校准，以测量或抵消要测量电容之外的所有其他电容。

18.4.1 电流源校准

CTMU 模块随附的电流源具有三种电流范围，其中每种范围都可以在其标称值 ±62% 的范围内进行调节。要进行精确测量，可以通过在未用模拟通道上放置一个高精度电阻 RCAL，测量并调整该电流源。图 18-2 给出了示例电路。

要测量电流源：

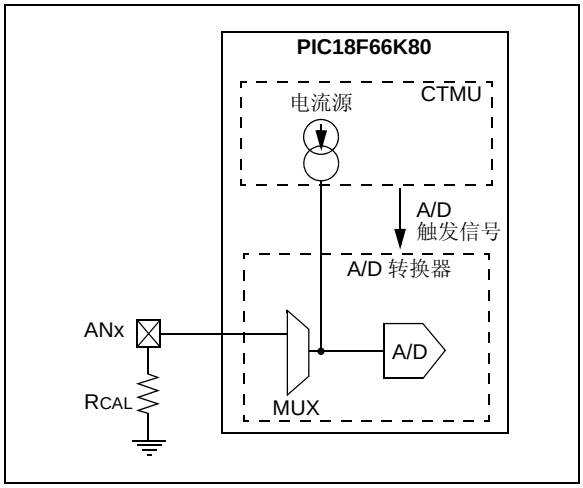
1. 初始化 A/D 转换器。
2. 初始化 CTMU。
3. 通过将 EDG1STAT（CTMUCONL<0>）置 1 使能电流源。
4. 产生延时使 RCAL 两端的电压达到稳定并对 ADC 采样 / 保持电容进行充电。
5. 执行 A/D 转换。
6. 使用 I = V / RCAL 计算电流源电流；其中，RCAL 是高精度电阻， V 通过执行 A/D 转换来测量。

CTMU 电流源可以使用 CTMUICON 中的微调位进行微调，通过迭代过程来获取所需的精确电流。或者，也可以使用未经调整的标称值。可以由软件存储调整后的电流值，用于所有后续的电容或时间测量。

要计算 RCAL 的最佳值，必须选择标称电流。

例如，如果 A/D 转换器参考电压为 3.3V，使用满量程的 70%（即 2.31V）作为要由 A/D 转换器读取的所需近似电压。如果 CTMU 电流源的范围选择为 0.55 µA，则所需的电阻值使用 RCAL = 2.31V/0.55 µA 计算，得到值为 4.2 MÙ。类似地，如果电流源选择为 5.5 µA，则 RCAL 将为 420,000Ù；如果电流源设置为 55 µA，则为 42,000Ù。

图 18-2：　　CTMU 电流源校准电路



选择满量程电压 70% 的值，以确保 A/D 转换器处于充分高于基底噪声的范围。如果选择了某个需要结合使用 CTMUICON 的微调位的精确电流，则可能需要对 RCAL 的电阻值进行相应调整。可能还需要再次调整 RCAL，以允许选择可用的电阻值。考虑将使用 CTMU 进行测量的电路所需的精度， RCAL 应选择可用的最高精度。建议最小精度是允许 0.1% 的误差。

以下示例给出了执行 CTMU 电流校准的一种典型方法。

- 例 18-1 演示了如何初始化 A/D 转换器和 CTMU。

  该程序是同时使用两个模块的应用的典型程序。

- 例 18-2 演示了实际校准程序的一种方法。

  该方法需要手动触发 A/D 转换器，以逐步演示整个过程。也可以通过将 CTMU 的 CTTRIG 位（CTMUCONH<0>）置 1 来自动触发转换。

例 18-1：　　CTMU 校准设置程序

```
#include "p18cxxx.h"
/*************************************************************************/
/*Setup CTMU ************************************************************/
/*************************************************************************/
void setup(void)

{ //CTMUCON - CTMU Control register

    CTMUCONH = 0x00;              //make sure CTMU is disabled
    CTMUCONL = 0x90;
    //CTMU continues to run when emulator is stopped,CTMU continues
    //to run in idle mode,Time Generation mode disabled, Edges are blocked
    //No edge sequence order, Analog current source not grounded, trigger
    //output disabled, Edge2 polarity = positive level, Edge2 source =
    //source 0, Edge1 polarity = positive level, Edge1 source = source 0,
    // Set Edge status bits to zero


    //CTMUICON - CTMU Current Control Register
    CTMUICON = 0x01;              //0.55uA, Nominal - No Adjustment

/*************************************************************************/
//Setup AD converter;
/*************************************************************************/

    TRISA=0x04;                   //set channel 2 as an input

    // Configured AN2 as an analog channel
    // ANCON1
    ANCON1 = 0x04;

    // ADCON1
    ADCON2bits.ADFM=1;            // Result format 1= Right justified
    ADCON2bits.ACQT=1;            // Acquisition time 7 = 20TAD 2 = 4TAD 1=2TAD
    ADCON2bits.ADCS=2;            // Clock conversion bits 6= FOSC/64 2=FOSC/32

    // ADCON1
    ADCON1bits.VCFG0 =0;          // Vref+ = AVdd
    ADCON1bits.VCFG1 =0;          // Vref+ = AVdd
    ADCON1bits.VNCFG = 0;         // Vref- = AVss
    ADCON1bits.CHS=2;             // Select ADC channel

    ADCON0bits.ADON=1;            // Turn on ADC

}
```

例 18-2：　　电流校准程序

```
#include "p18cxxx.h"

#define COUNT 500                       //@ 8MHz = 125uS.
#define DELAY for(i=0;i<COUNT;i++)
#define RCAL .027                       //R value is 4200000 (4.2M)
                                        //scaled so that result is in
                                        //1/100th of uA
#define ADSCALE 1023                    //for unsigned conversion 10 sig bits
#define ADREF 3.3                       //Vdd connected to A/D Vr+

int main(void)
{
    int i;
    int j = 0;                          //index for loop
    unsigned int Vread = 0;
    double VTot = 0;
    float Vavg=0, Vcal=0, CTMUISrc = 0;  //float values stored for calcs

//assume CTMU and A/D have been setup correctly
//see Example 25-1 for CTMU & A/D setup
setup();

CTMUCONHbits.CTMUEN = 1;                //Enable the CTMU
    for(j=0;j<10;j++)
    {
        CTMUCONHbits.IDISSEN = 1;       //drain charge on the circuit
        DELAY;                          //wait 125us
        CTMUCONHbits.IDISSEN = 0;       //end drain of circuit

        CTMUCONLbits.EDG1STAT = 1;      //Begin charging the circuit
                                        //using CTMU current source
        DELAY;                          //wait for 125us
        CTMUCONLbits.EDG1STAT = 0;      //Stop charging circuit

        PIR1bits.ADIF = 0;              //make sure A/D Int not set
        ADCON0bits.GO=1;                //and begin A/D conv.
        while(!PIR1bits.ADIF);          //Wait for A/D convert complete

        Vread = ADRES;                  //Get the value from the A/D
        PIR1bits.ADIF = 0;              //Clear A/D Interrupt Flag
        VTot += Vread;                  //Add the reading to the total
    }

    Vavg = (float)(VTot/10.000);        //Average of 10 readings
    Vcal = (float)(Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL;               //CTMUISrc is in 1/100ths of uA

}
```

18.4.2 电容校准

内部 A/D 转换器采样电容和电路板走线与焊盘的杂散电容虽然容值较小，但仍会影响电容测量的精度。在确保移除期望测量的电容的情况下，可以对杂散电容进行测量。

移除要测量的电容之后：

1. 初始化 A/D 转换器和 CTMU。
2. 将 EDG1STAT 置 1 （= 1）。
3. 等待固定延时 t。
4. 清零 EDG1STAT。
5. 执行 A/D 转换。
6. 计算杂散电容和 A/D 采样电容：

$$C_{OFFSET} = C_{STRAY} + C_{AD} = (I \bullet t)/V$$

其中：

- I 从电流源测量步骤获知
- t 是固定延时
- V 通过执行 A/D 转换来测量

然后，可以存储该测量值，用于时间测量时的计算，或在电容测量时减去该值。要进行校准，需要大致了解 $C_{STRAY} + C_{AD}$ 的电容值； C_{AD} 约为 4 pF。

可能需要使用一个迭代过程来调整时间 t，该时间是对电路进行充电，以从 A/D 转换器获得合理电压读数的时间。 t 的值可以通过将 C_{OFFSET} 设置为理论值，然后求解 t 来确定。例如，如果 C_{STRAY} 的理论计算值为 11 pF，V 预期为 V_{DD} 的 70% （即 2.31V），那么 t 为：

$$(4\ pF + 11\ pF) \bullet 2.31V/0.55\ \mu A$$

或 63 µs。

请参见例 18-3 了解 CTMU 电容校准的典型程序。

例 18-3：　　电容校准程序

```
#include "p18cxxx.h"

#define COUNT 25                        //@ 8MHz INTFRC = 62.5 us.
#define ETIME COUNT*2.5                 //time in uS
#define DELAY for(i=0;i<COUNT;i++)
#define ADSCALE 1023                    //for unsigned conversion 10 sig bits
#define ADREF 3.3                       //Vdd connected to A/D Vr+
#define RCAL .027                       //R value is 4200000 (4.2M)
                                        //scaled so that result is in
                                        //1/100th of uA

int main(void)
{
    int i;
    int j = 0;                          //index for loop
    unsigned int Vread = 0;
    float CTMUISrc, CTMUCap, Vavg, VTot, Vcal;

//assume CTMU and A/D have been setup correctly
//see Example 25-1 for CTMU & A/D setup
setup();

CTMUCONHbits.CTMUEN = 1;                //Enable the CTMU
    for(j=0;j<10;j++)
    {
        CTMUCONHbits.IDISSEN = 1;       //drain charge on the circuit
DELAY;                                  //wait 125us
        CTMUCONHbits.IDISSEN = 0;       //end drain of circuit

        CTMUCONLbits.EDG1STAT = 1;      //Begin charging the circuit
                                        //using CTMU current source
DELAY;                                  //wait for 125us
        CTMUCONLbits.EDG1STAT = 0;      //Stop charging circuit

        PIR1bits.ADIF = 0;              //make sure A/D Int not set
        ADCON0bits.GO=1;                //and begin A/D conv.
        while(!PIR1bits.ADIF);          //Wait for A/D convert complete


        Vread = ADRES;                  //Get the value from the A/D
        PIR1bits.ADIF = 0;              //Clear A/D Interrupt Flag
VTot += Vread;                          //Add the reading to the total
    }

    Vavg = (float)(VTot/10.000);        //Average of 10 readings
    Vcal = (float)(Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL;               //CTMUISrc is in 1/100ths of uA
    CTMUCap = (CTMUISrc*ETIME/Vcal)/100;
}
```

18.5 使用 CTMU 测量电容

使用 CTMU 测量电容有两种方法。绝对方法需要测量实际电容值。相对方法只需要测量电容的变化量。

18.5.1 绝对电容测量

对于绝对电容测量，应遵循**第 18.4 节 “ 校准 CTMU 模块 ”** 中的电流和电容校准步骤。

要执行这些测量：

1. 初始化 A/D 转换器。
2. 初始化 CTMU。
3. 将 EDG1STAT 置 1。
4. 等待固定延时 T。
5. 清零 EDG1STAT。
6. 执行 A/D 转换。
7. 计算总电容 CTOTAL = (I * T)/V，其中：
   - I 从电流源测量步骤（**第 18.4.1 节 “ 电流源校准 ”**）获知
   - T 是固定延时
   - V 通过执行 A/D 转换来测量
8. 从 CTOTAL 中减去杂散电容和 A/D 采样电容（COFFSET 来自**第 18.4.2 节 “ 电容校准 ”**），确定被测电容的值。

18.5.2 使用相对电荷测量实现电容式触摸传感

并不是所有应用都需要精确的电容测量。在检测电容式开关的有效按压时，只需要检测电容的相对变化。

在此类应用中，当开关打开（未被触摸）时，总电容是电路板走线、A/D 转换器和其他元件的组合电容。此时 A/D 转换器将会测量到较大的电压。当开关关闭（被触摸）时，由于以上所列电容中增加了人体的电容，总电容增大， A/D 转换器将测量到较小的电压。

要检测电容变化，只需：

1. 初始化 A/D 转换器和 CTMU。
2. 将 EDG1STAT 置 1。
3. 等待固定延时。
4. 清零 EDG1STAT。
5. 执行 A/D 转换。

通过执行 A/D 转换测量的电压可以指示相对电容。在这种情况下，不需要对电流源或电路电容测量进行校准。（关于电容式触摸开关的软件程序示例，请参见例 18-4。）

例 18-4：　　用于电容式触摸开关的程序

```
#include "p18cxxx.h"

#define COUNT 500                       //@ 8MHz = 125uS.
#define DELAY for(i=0;i<COUNT;i++)
#define OPENSW 1000                     //Un-pressed switch value
#define TRIP 300                        //Difference between pressed
                                        //and un-pressed switch
#define HYST 65                         //amount to change
                                        //from pressed to un-pressed
#define PRESSED 1
#define UNPRESSED 0

int main(void)
{
unsigned int Vread;                     //storage for reading
    unsigned int switchState;
    int i;

    //assume CTMU and A/D have been setup correctly
    //see Example 25-1 for CTMU & A/D setup
    setup();

CTMUCONHbits.CTMUEN = 1;                //Enable the CTMU

        CTMUCONHbits.IDISSEN = 1;       //drain charge on the circuit
DELAY;                                  //wait 125us
        CTMUCONHbits.IDISSEN = 0;       //end drain of circuit

        CTMUCONLbits.EDG1STAT = 1;      //Begin charging the circuit
                                        //using CTMU current source
DELAY;                                  //wait for 125us
        CTMUCONLbits.EDG1STAT = 0;      //Stop charging circuit

        PIR1bits.ADIF = 0;              //make sure A/D Int not set
        ADCON0bits.GO=1;                //and begin A/D conv.
        while(!PIR1bits.ADIF);          //Wait for A/D convert complete

        Vread = ADRES;                  //Get the value from the A/D

    if(Vread < OPENSW - TRIP)
    {
        switchState = PRESSED;
    }
    else if(Vread > OPENSW - TRIP + HYST)
    {
        switchState = UNPRESSED;
    }
}
```

18.6 使用 CTMU 模块测量时间

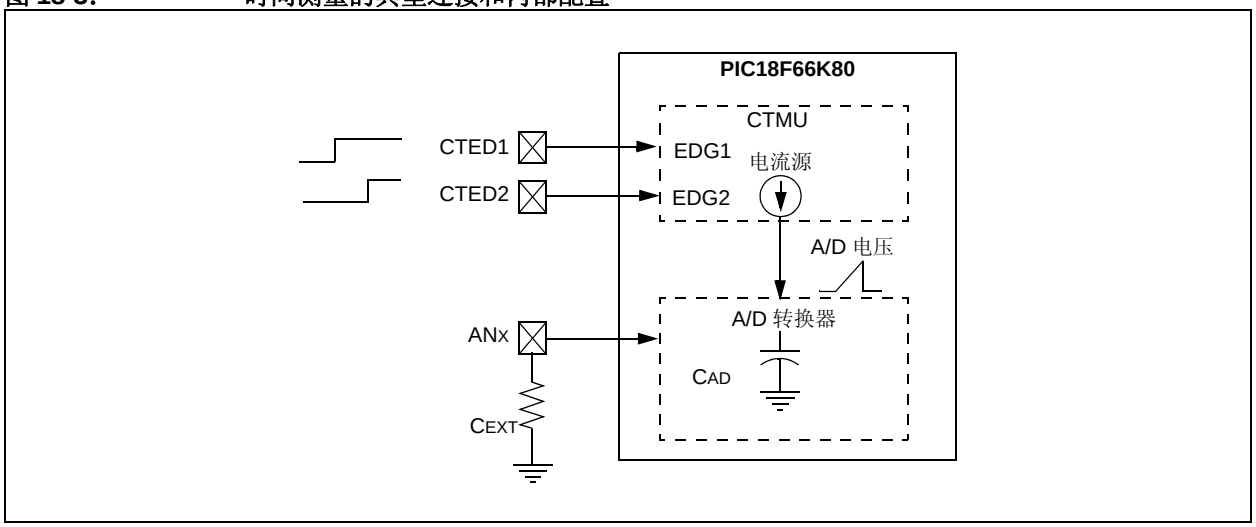
通过电流和电容校准步骤测量比率（C/I）之后，可以精确测量时间。请执行以下步骤：

1. 初始化 A/D 转换器和 CTMU。
2. 将 EDG1STAT 置 1。
3. 将 EDG2STAT 置 1。
4. 执行 A/D 转换。
5. 根据 T = (C/I) * V 计算边沿之间的时间，其中：
   - I 在电流校准步骤（**第 18.4.1 节 “ 电流源校准 ”**）中计算
   - C 在电容校准步骤（**第 18.4.2 节 “ 电容校准 ”**）中计算
   - V 通过执行 A/D 转换来测量

假定所测量的时间足够小，电容 CAD + CEXT 可以向 A/D 转换器提供有效的电压。要进行最小的时间测量，请始终将 A/D 通道选择位 CHS<4:0>（ADCON0<6:2>）设置为未用的 A/D 通道，该通道的相应引脚不连接到任何电路板走线。这可以最大程度减小所增加的杂散电容，保持总电路电容接近于 A/D 转换器自身的电容（25 pF）。

要测量较长的时间间隔，可以将一个外部电容连接到 A/D 通道，并在进行时间测量时选择该通道。

图 18-3： 时间测量的典型连接和内部配置



18.7 使用 CTMU 测量温度

通过利用普通低价二极管的基本特性，可以将 CTMU 模块提供的恒流源用于低成本温度测量。ADC 通道 29 上提供了一个片上温度检测二极管，用以进一步简化设计和降低成本。

18.7.1 基本原则

可以证明，P-N 结（如二极管）的正向电压（V_F）可由结点热电压公式计算：

$$V_F = \frac{kT}{q} \ln\left(1 - \frac{I_F}{I_S}\right)$$

其中，k 是波尔茨曼常数（1.38×10^{-23} J K^{-1}），T 是以开尔文为单位的绝对结温，q 是电子电荷（1.6×10^{-19} C），I_F 是对二极管施加的正向电流，I_S 是二极管的特性饱和电流，它会随器件而变化。

由于 k 和 q 是物理常数，I_S 是器件的一个常量，所以只剩下 T 和 I_F 是独立变量。如果 I_F 保持恒定，根据公式可以推得 V_F 将为 T 的函数。由于公式的自然对数项总是为负，所以温度会与 V_F 成反比。即，当温度升高时，V_F 会下降。

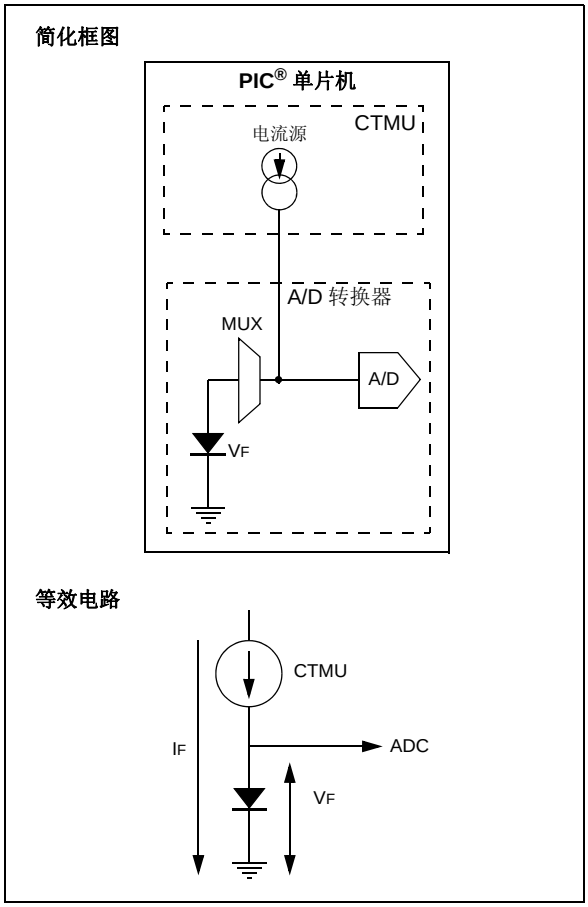
通过使用 CTMU 的电流源来提供恒定的 I_F，可以通过测量二极管上的 V_F 来计算温度。

18.7.2 实现方案

要将这一理论付诸实践，所需要的就是将一个普通的结型二极管与单片机的一个 A/D 引脚连接（图 18-2）。A/D 通道多路开关由 CTMU 和 ADC 共用。

要执行测量，需要对多路开关进行配置，以选择与二极管连接的引脚。然后，开启 CTMU 电流源，并对通道执行 A/D 转换。如等效电路图中所示，二极管由 CTMU 以 I_F 驱动。在二极管上产生的 V_F 通过 ADC 来测量。例 18-5 给出了代码片断。

图 18-4： CTMU 温度测量电路



例 18-5： 使用内部二极管进行温度测量的程序

```
// Initialize CTMU
CTMUICON = 0x03;
CTMUCONHbits.CTMUEN = 1;
CTMUCONLbits.EDG1STAT = 1;

// Initialize ADC
ADCON0 = 0xE5;                          // Enable ADC and connect to Internal diode
ADCON1 = 0x00;
ADCON2 = 0xBE;                          //Right Justified

ADCON0bits.GO = 1;                      // Start conversion
while(ADCON0bits.G0);
Temp = ADRES;                           // Read ADC results (inversely proportional to temperature)
```

注： 温度二极管未经过校准或标准化；用户必须根据应用对二极管进行校准。

18.8 使用 CTMU 模块产生延时

CTMU 模块具有一种独特功能，即它可以根据外部电压或外部电容值产生独立于系统时钟的输出脉冲。使用外部电压时，这通过使用 CTDIN 输入引脚作为脉冲延时的触发信号来实现。使用外部电容值时，这通过使用内部比较器参考电压模块和比较器 2 输入引脚来实现。脉冲输出到 CTPLS 引脚上。要使能该模式，需将 TGEN 位置 1。

示例电路请参见图18-5。当CTMUDS（PADCFG1<0>）清零时，脉冲延时由比较器 2 的输出决定，而当它置 1 时，脉冲延时由 CTDIN 的输入决定。CDELAY 由用户选择，用于确定 CTPLS 上的输出脉冲宽度。脉冲宽度根据 T = (CDELAY/I)*V 计算；其中，I 从电流源测量步骤（**第 18.4.1 节 “ 电流源校准 ”**）获知，V 是内部参考电压 （CVREF）。

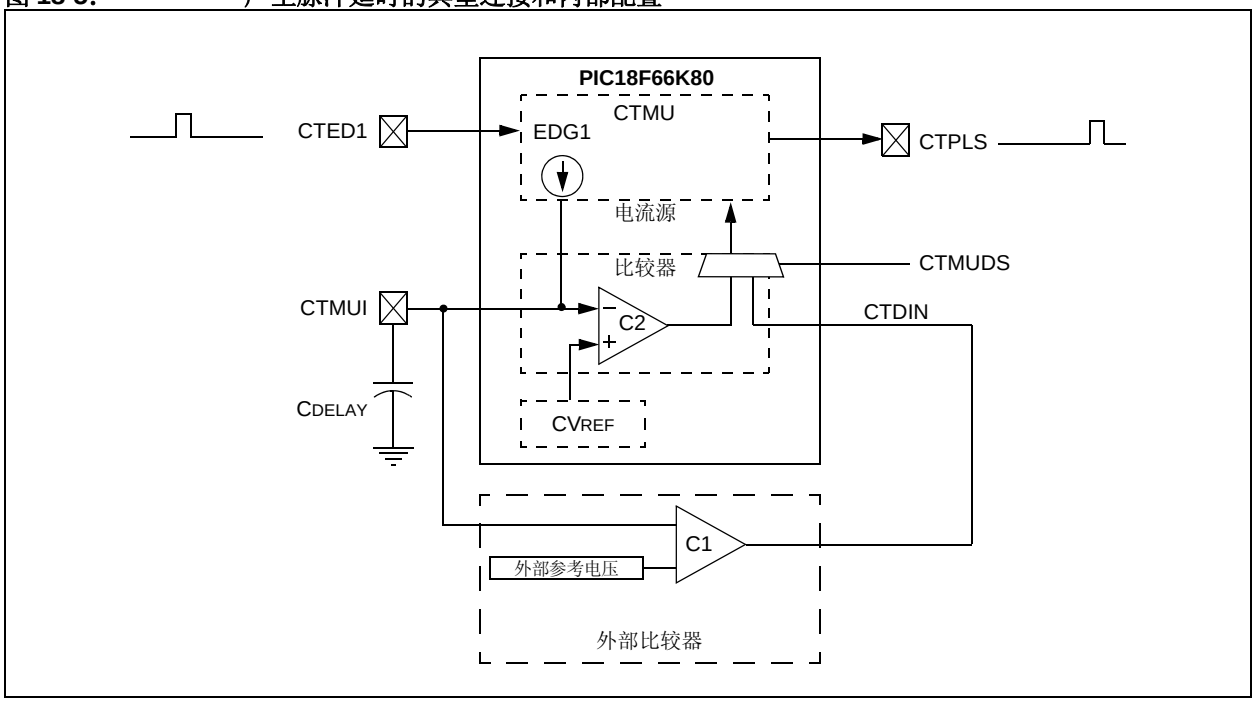
外部电容功能的使用示例是与基于可变电容的传感器进行连接，例如湿度传感器。当湿度发生变化时，CTPLS 上的脉宽输出也会变化。CTDIN 功能的使用示例是与数字传感器连接。 CTPLS 输出引脚可以连接到输入捕捉引脚，通过测量变化的脉冲宽度来确定应用中传感器的输出。

要使用该功能：

1. 如果 CTMUDS 清零，初始化比较器 2。
2. 如果 CTMUDS 清零，初始化比较器参考电压。
3. 初始化 CTMU，并通过将 TGEN 位置 1 来使能延时生成。
4. 将 EDG1STAT 置 1。

当 CTMUDS 清零时，只要 CDELAY 充电到参考电压跳变点的值时，就会在 CTPLS 上产生输出脉冲。当 CTMUDS 置 1 时，只要 CTDIN 置 1，就会在 CTPLS 上产生输出脉冲。

图 18-5： 产生脉冲延时的典型连接和内部配置



18.9 使用 CTMU 模块测量温度

CTMU 与内部二极管一起，可以用于测量温度。ADC 可以连接到内部二极管，CTMU 模块可以向二极管提供电流。ADC 读数将会反映温度。温度升高时，ADC 读数会降低。这可以用于低成本的温度测量应用。

例 18-6： 使用内部二极管进行温度测量的程序

```
// Initialize CTMU
CTMUICON = 0x03;
CTMUCONHbits.CTMUEN = 1;
CTMUCONLbits.EDG1STAT = 1;

// Initialize ADC
ADCON0 = 0xE5;                          // Enable ADC and connect to Internal diode
ADCON1 = 0x00;
ADCON2 = 0xBE;                          //Right Justified

ADCON0bits.GO = 1;                      // Start conversion
while(ADCON0bits.G0);
Temp = ADRES;                           // Read ADC results (inversely proportional to temperature)
```

注： 温度二极管未经过校准或标准化；用户必须根据应用对二极管进行校准。

18.10 休眠 / 空闲模式期间的操作

18.10.1 休眠模式

当器件进入休眠模式时，CTMU 模块电流源将始终禁止。如果调用休眠模式时，CTMU 正在执行依赖于电流源的操作，则操作可能不会正确终止。电容和时间测量可能会返回错误值。

18.10.2 空闲模式

CTMU 在空闲模式下的行为由 CTMUSIDL 位（CTMUCONH<5>）决定。如果 CTMUSIDL 清零，在空闲模式下，模块将继续工作。如果 CTMUSIDL 置 1，则在器件进入空闲模式时，模块的电流源会被禁止。这种情况下，如果调用空闲模式时，模块正在执行操作，结果将类似于休眠模式下的结果。

18.11 复位对 CTMU 的影响

在复位时，CTMU 的所有寄存器都会被清零。这会禁止 CTMU 模块、关闭其电流源并将所有配置选项恢复为它们的默认设置。在任意复位之后，模块都需要重新初始化。

如果发生复位时，CTMU 正在进行测量，测量结果将丢失。正在测量的电路可能会存在部分充电的情况，在随后 CTMU 尝试进行测量之前，应正确进行放电。电路放电方法是，在 A/D 转换器连接到相应通道的同时，先将 IDISSEN 位（CTMUCONH<1>）置 1，然后再将其清零。

表 18-1： 与 CTMU 模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CTMUCONH	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG
CTMUCONL	EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT
CTMUICON	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
PADCFG1	RDPU	REPU	RFPU	RGPU	—	—	—	CTMUDS

图注： — = 未实现，读为 0。 CTMU 操作期间不使用阴影单元。

19.0 捕捉 / 比较 /PWM （CCP）模块

PIC18F66K80 系列器件具有 4 个 CCP （捕捉 / 比较 / PWM）模块，指定为 CCP2 至 CCP5。所有模块均可实现标准的捕捉、比较和脉宽调制（Pulse-Width Modulation， PWM）模式。

注： 在本章中，使用通用表示形式来表示除 “x” 变量外其余都相同的寄存器和位名称，“x”变量指示与特定 CCP 模块的项关联。例如，控制寄存器被命名为 CCPxCON，指 CCP2CON 至 CCP5CON。

每个 CCP 模块都包含一个 16 位寄存器，可用作 16 位捕捉寄存器、16 位比较寄存器或 PWM 主 / 从占空比寄存器。为清晰起见，以下小节中所有的 CCP 模块操作说明均针对 CCP2，但同样适用于 CCP3 至 CCP5。

寄存器 19-1： CCPxCON：CCPx 控制寄存器 （CCP2-CCP5 模块）(1)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3(1)	CCPxM2(1)	CCPxM1(1)	CCPxM0(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 7-6 **未实现：**读为 0

bit 5-4 **DCxB<1:0>：**CCPx 模块的 PWM 占空比 bit 1 和 bit 0

捕捉模式：
未使用。

比较模式：
未使用。

PWM 模式：
这两位是 10 位 PWM 占空比的低 2 位（bit 1 和 bit 0）。占空比的高 8 位（DCx<9:2>）在 CCPRxL 中。

bit 3-0 **CCPxM<3:0>：**CCPx 模块模式选择位 (1)

0000 = 禁止捕捉 / 比较 /PWM （复位 CCPx 模块）
0001 = 保留
0010 = 比较模式：匹配时输出电平翻转 （CCPxIF 位置 1）
0011 = 保留
0100 = 捕捉模式：每个下降沿或接收到 CAN 报文 （时间标记） (2)
0101 = 捕捉模式：每个上升沿或接收到 CAN 报文 （时间标记） (2)
0110 = 捕捉模式：每 4 个上升沿或每接收到 4 个 CAN 报文 （时间标记） (2)
0111 = 捕捉模式：每 16 个上升沿或每接收到 16 个 CAN 报文 （时间标记） (2)
1000 = 比较模式：初始化 CCPx 引脚为低电平；比较匹配时强制 CCPx 引脚为高电平（CCPxIF 位置 1）
1001 = 比较模式：初始化 CCPx 引脚为高电平；比较匹配时强制 CCPx 引脚为低电平（CCPxIF 位置 1）
1010 = 比较模式：比较匹配时产生软件中断 （CCPxIF 位置 1， CCPx 引脚反映 I/O 状态）
1011 = 比较模式：特殊事件触发； CCPx 匹配时复位定时器 （CCPxIF 位置 1）
11xx = PWM 模式

注 1： 如果 CCPxM<3:0> = 1011，在 CCPx 发生匹配时将只复位定时器而不启动 A/D 转换。
2： 仅在 CCP2 上可用。通过 CANCAP （CIOCON<4>）位选择。改写 CCP2 输入引脚源。

寄存器 19-2：　　CCPTMRS：CCP 定时器选择寄存器

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	C5TSEL	C4TSEL	C3TSEL	C2TSEL	C1TSEL
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　**C5TSEL：**CCP5 定时器选择位
0 = CCP5 基于 TMR1/TMR2 产生
1 = CCP5 基于 TMR3/TMR4 产生

bit 3　**C4TSEL：**CCP4 定时器选择位
0 = CCP4 基于 TMR1/TMR2 产生
1 = CCP4 基于 TMR3/TMR4 产生

bit 2　**C3TSEL：**CCP3 定时器选择位
0 = CCP3 基于 TMR1/TMR2 产生
1 = CCP3 基于 TMR3/TMR4 产生

bit 1　**C2TSEL：**CCP2 定时器选择位
0 = CCP2 基于 TMR1/TMR2 产生
1 = CCP2 基于 TMR3/TMR4 产生

bit 0　**C1TSEL：**CCP1 定时器选择位
0 = ECCP1 基于 TMR1/TMR2 产生
1 = ECCP1 基于 TMR3/TMR4 产生

寄存器 19-3： CCPRxL：CCPx 周期低字节寄存器

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
CCPRxL7	CCPRxL6	CCPRxL5	CCPRxL4	CCPRxL3	CCPRxL2	CCPRxL1	CCPRxL0
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 **CCPRxL<7:0>：**CCPx 周期寄存器低字节位
捕捉模式：捕捉寄存器的低字节
比较模式：比较寄存器的低字节
PWM 模式：占空比寄存器

寄存器 19-4： CCPRxH：CCPx 周期高字节寄存器

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
CCPRxH7	CCPRxH6	CCPRxH5	CCPRxH4	CCPRxH3	CCPRxH2	CCPRxH1	CCPRxH0
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 **CCPRxH<7:0>：**CCPx 周期寄存器高字节位
捕捉模式：捕捉寄存器的高字节
比较模式：比较寄存器的高字节
PWM 模式：占空比缓冲寄存器

19.1 CCP 模块配置

每个捕捉 / 比较 /PWM 模块都与一个控制寄存器（通常为 CCPxCON）和一个数据寄存器（CCPRx）相关联。数据寄存器由两个 8 位寄存器组成：CCPRxL（低字节）和CCPRxH（高字节）。所有寄存器都是可读写的。

19.1.1 CCP 模块和定时器资源

CCP 模块根据选定的模式使用 Timer1 至 Timer4。在捕捉、比较或 PWM 模式下，有各种定时器可供 CCP 模块使用，如表 19-1 所示。

表 19-1： CCP 模式 —— 定时器资源

CCP 模式	定时器资源
捕捉	Timer1 或 Timer3
比较	
PWM	Timer2 或 Timer4

要将哪个特定的定时器分配给 CCP 模块由 CCPTMRS 寄存器（见寄存器 19-2）中的 “CCP 的定时器 ” 使能位决定。如果将所有 CCP 模块配置为同时工作在相同的模式（捕捉 / 比较或 PWM）下，那么它们可立刻被激活并可共用相同的定时器资源。

CCPTMRS 寄存器选择 CCP 模块 2、3、4 和 5 的定时器。表 19-2 列出了可能的配置。

表 19-2： CCP 模块 2、 3、 4 和 5 的定时器分配

CCPTMRS 寄存器											
CCP2			CCP3			CCP4			CCP5		
C2TSEL	捕捉 / 比较模式	PWM 模式	C3TSEL	捕捉 / 比较模式	PWM 模式	C4TSEL	捕捉 / 比较模式	PWM 模式	C5TSEL	捕捉 / 比较模式	PWM 模式
0	TMR1	TMR2	0	TMR1	TMR2	0	TMR1	TMR2	0 0	TMR1	TMR2
1	TMR3	TMR4	1	TMR3	TMR4	1	TMR3	TMR4	0 1	TMR3	TMR4

19.1.2 漏极开路输出选项

当在输出模式（比较或 PWM 模式）下工作时，可以选择将 CCPx 引脚的驱动器配置为漏极开路输出。该功能使引脚上的电压可通过外部上拉电阻上拉至较高的电压，并且无需额外的电平转换器就可使输出与外部电路进行通信。

漏极开路输出选项由 CCPxOD 位（ODCON<6:2>）控制。通过将相应位置 1 可将对应模块的引脚配置为漏极开路操作。

19.2 捕捉模式

在捕捉模式下，当相应的 CCPx 引脚发生以下事件时，CCPRxH:CCPRxL 寄存器对捕捉 CCPTMRS 中选择的定时器寄存器的 16 位值。事件定义为以下情况之一：

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

> **注：** 仅对于 CCP2，捕捉模式可以使用 CCP2 输入引脚作为 CCP2 的捕捉触发信号，或者该输入也可以通过 CAN 模块用于时间标记功能。CAN 模块提供了必需的控制和触发信号。

事件由模式选择位 CCPxM<3:0>（CCPxCON<3:0>）选择。当完成一次捕捉时，中断请求标志位 CCPxIF（PIR4<x>）置 1；它必须用软件清零。如果在读取 CCPRx 值之前发生了另一次捕捉，那么原来的捕捉值会被新的捕捉值覆盖。

图 19-1 给出了捕捉模块的框图。

19.2.1 CCP 引脚配置

在捕捉模式下，应通过将相应的 TRIS 方向位置 1，将 CCPx 引脚配置为输入。

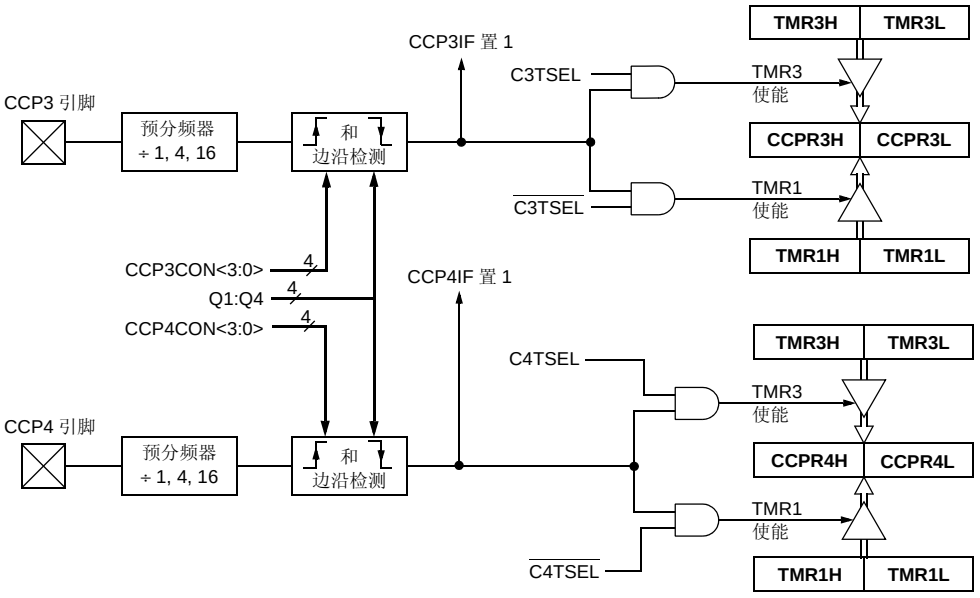
19.2.2 TIMER1/3 模式选择

对于可用于捕捉功能的定时器（Timer1/3），所用的定时器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。

可在 CCPTMRS 寄存器中选择用于每个 CCP 模块的定时器（见**第 19.1.1 节 “CCP 模块和定时器资源”**）。

表 19-2 给出了 CCP 模块的定时器分配的详细信息。

图 19-1：　　捕捉模式工作原理框图



> **注：** 该框图以 CCP3 和 CCP4 及其相应的定时器为例。关于所有 CCP 模块及其定时器分配的详细信息，请参见表 19-2。

19.2.3 软件中断

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 CCPxIE 位（PIE4<x>）清零以避免错误中断，并且还应在工作模式改变后清零标志位 CCPxIF。

19.2.4 CCP 预分频器

在捕捉模式下有 4 种预分频比设置。它们作为工作模式的一部分由模式选择位（CCPxM<3:0>）指定。每当关闭 CCP 模块，或者 CCP 模块不在捕捉模式下时，预分频器计数器就会被清零。这意味着任何复位都会将预分频器计数器清零。

在两个捕捉预分频比之间切换可能会产生中断。而且，不会清零预分频器计数器；这意味着第一次捕捉可能来自于一个非零的预分频器。

例 19-1 给出了切换捕捉预分频比时建议采用的方法。这个示例使预分频器计数器清零且不会产生错误中断。

例 19-1： 改变捕捉预分频比

```
CLRF   CCPxCON      ; Turn CCP module off
MOVLW  NEW_CAPT_PS  ; Load WREG with the
                    ; new prescaler mode
                    ; value and CCP ON
MOVWF  CCPxCON      ; Load CCPxCON with
                    ; this value
```

19.2.5 CAN 报文时间标记 （仅限 CCP2）

对于 CCP2，只有在任意接收缓冲区中接收到报文时，才会发生 CAN 捕捉事件。在配置后，CAN 模块可以向 CCP2 模块提供可导致捕捉事件的触发信号。该功能用于标记 CAN 报文的接收时间。

该功能通过将 CAN I/O 控制寄存器的 CANCAP 位（CIOCON<4>）置 1 来使能。然后，来自 CAN 模块的报文接收信号会取代 RC2/CCP2 上的事件。

如果选择了该功能，则 CCP2M<3:0> 可提供 4 种不同的捕捉选项：

- 0100—— 每次接收到 CAN 报文时
- 0101—— 每次接收到 CAN 报文时
- 0110—— 每接收到 4 个 CAN 报文时
- 0111—— 捕捉模式，每接收到 16 个 CAN 报文时

19.3 比较模式

在比较模式下，16 位 CCPRx 寄存器的值不断地与 CCPTMR 寄存器中选择的定时器寄存器对的值作比较。当两者匹配时， CCPx 引脚将会：

- 驱动为高电平
- 驱动为低电平
- 翻转 （高电平变为低电平或低电平变为高电平）
- 不变 （即反映 I/O 锁存器的状态）

引脚动作取决于模式选择位（CCPxM<3:0>）的值。同时，中断标志位 CCPxIF 置 1。

图 19-2 给出了比较模式的框图。

19.3.1 CCP 引脚配置

用户必须通过将相应的 TRIS 位清零，将 CCPx 引脚配置为输出。

注：	清零 CCPxCON 寄存器会将相应的 CCPx 比较输出锁存器（取决于器件配置）强制为默认的低电平。这不是 PORTx 数据锁存器。

19.3.2 TIMER1/3 模式选择

如果 CCPx 模块与任一 Timer1/3 定时器配合使用比较功能，定时器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。

注：	表 19-2 给出了 CCPx 模块的定时器分配的详细信息。

19.3.3 软件中断模式

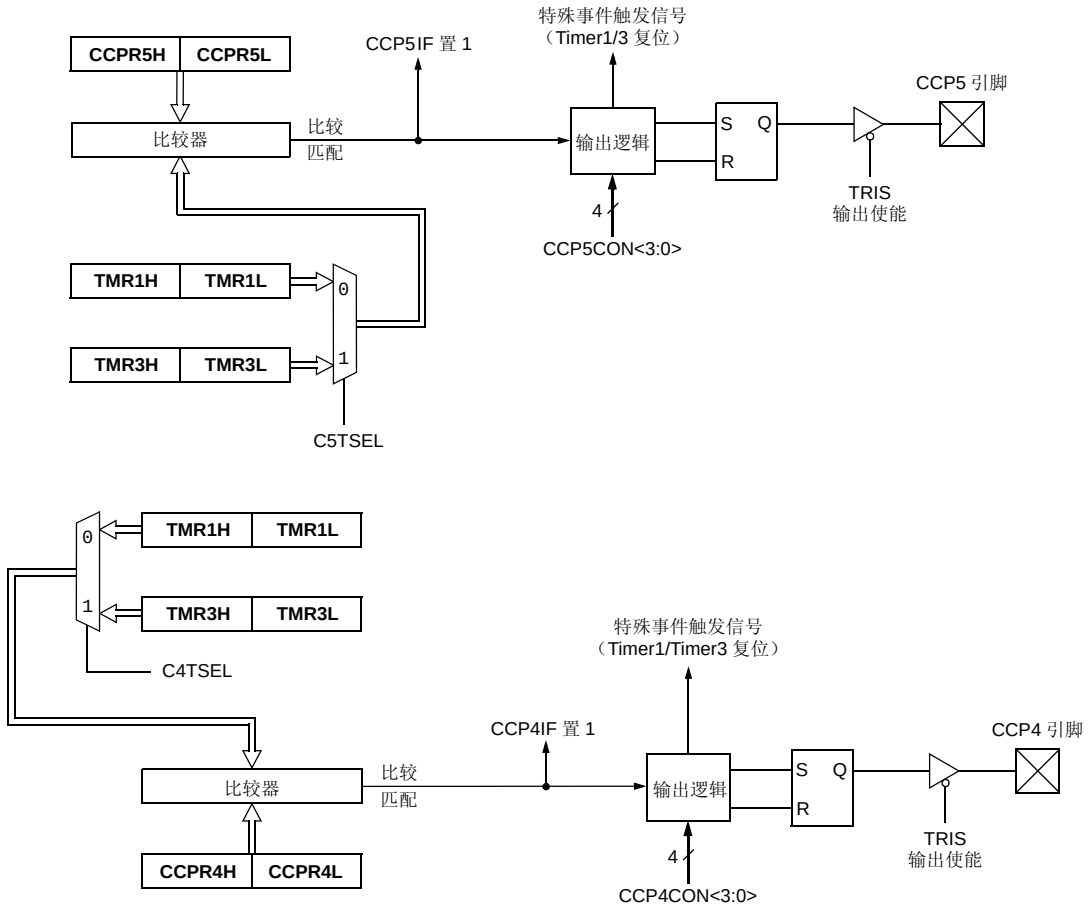
当选择了 “ 产生软件中断 ” 模式（CCPxM<3:0> = 1010）时， CCPx 引脚不受影响。如果中断被允许，将仅产生 CCP 中断并将 CCPxIF 位置 1。

19.3.4 特殊事件触发器

所有 CCP 模块都配备了一个特殊事件触发器。在比较模式下可产生内部硬件信号以触发其他模块动作。通过选择 “ 比较特殊事件触发 ” 模式（CCPxM<3:0> = 1011），使能特殊事件触发器。

对于任何一个 CCPx 模块，无论当前使用哪个定时器资源作为模块的时基，特殊事件触发信号会将对应的定时器寄存器对复位。这样 CCPRx 寄存器可用作两个定时器中任一定时器的可编程周期寄存器。

图 19-2：　　比较模式工作原理框图



注：　该框图以 CCP4 和 CCP5 及其相应的定时器为例。关于所有 CCP 模块及其定时器分配的详细信息，请参见表 19-2。

表 19-3：　　与捕捉、比较和 TIMER1/3 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
RCON	IPEN	SBOREN	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIE4	TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
TMR1L	Timer1 寄存器的低字节							
TMR1H	Timer1 寄存器的高字节							
TMR3L	Timer3 寄存器的低字节							
TMR3H	Timer3 寄存器的高字节							
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{T1SYNC}$	RD16	TMR1ON
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{T3SYNC}$	RD16	TMR3ON
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节							
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节							
CCPR3L	捕捉 / 比较 /PWM 寄存器 3 的低字节							
CCPR3H	捕捉 / 比较 /PWM 寄存器 3 的高字节							
CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节							
CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节							
CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节							
CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节							
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0
CCP3CON	—	—	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0
CCPTMRS	—	—	—	C5TSEL	C4TSEL	C3TSEL	C2TSEL	C1TSEL
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD

图注：　　— = 未实现，读为 0。捕捉 / 比较或 Timer1/3 不使用阴影单元。

19.4 PWM 模式

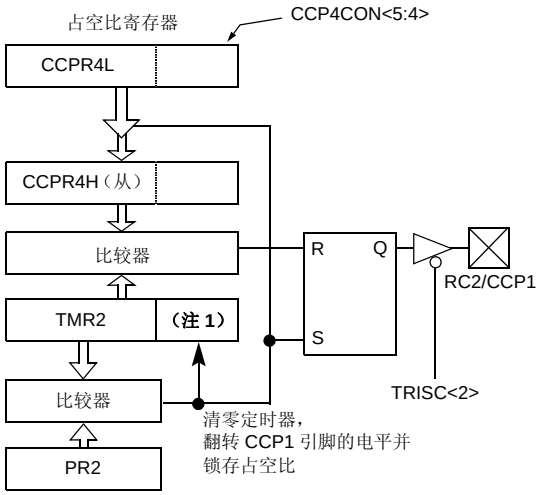
在脉宽调制（PWM）模式下，CCPx 引脚会产生最高 10 位分辨率的 PWM 输出信号。由于 CCPx 引脚与 PORTC 或 PORTB 数据锁存器复用，必须清零相应的 TRIS 位才能使 CCPx 引脚成为输出引脚。

注： 清零CCPxCON寄存器会将相应的CCPx输出锁存器（取决于器件配置）强制为默认的低电平。这不是 PORTx I/O 数据锁存器。

图 19-3 给出了 PWM 模式下 CCPx 模块的简化框图。

关于如何设置 CCP 模块使之工作于 PWM 模式的详细步骤，请参见**第 19.4.3 节 “ 设置 PWM 操作 ”**。

图 19-3： 简化的 PWM 框图

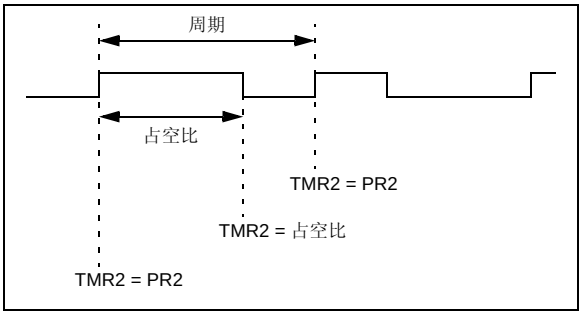


注 1： 8 位 TMR2 值与内部 Q 时钟或预分频器中的 2 位一起构成 10 位时基。

2： 以 CCP4 及其相应的定时器为例。关于所有 CCP 模块及其定时器分配的详细信息，请参见表 19-2 。

PWM 输出（图 19-4）有一个时基（周期）和一段输出保持为高电平的时间 （占空比）。 PWM 的频率是周期的倒数 （1/ 周期）。

图 19-4： PWM 输出



19.4.1 PWM 周期

可通过写 PR2 寄存器指定 PWM 周期。 PWM 周期可由以下公式计算：

公式 19-1：

$$\text{PWM 周期} = [(\text{PR2}) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

PWM 频率定义为 1/[PWM 周期]。

当 TMR2 中的值与 PR2 中的值相等时，在下一个递增周期将发生以下 3 个事件：

- TMR2 被清零
- CCP4 引脚被置 1
  （例外情况：如果 PWM 占空比 = 0%， CCP4 引脚将不会被置 1）
- PWM 占空比从 CCPR4L 锁存到 CCPR4H

注： 在确定 PWM 频率时不会用到 Timer2 后分频比 （见**第 15.0 节 “Timer2 模块 ”**）。后分频器可用不同于 PWM 输出频率的频率进行数据更新。

19.4.2 PWM 占空比

以CCP4为例，通过写入CCPR4L寄存器和CCP4CON<5:4>位来指定PWM占空比。分辨率最高可达10位。CCPR4L包含高 8 位而 CCP4CON<5:4> 包含低 2 位。这 10 位值由 CCPR4L:CCP4CON<5:4> 表示。以下公式用于计算 PWM 的占空比 （用时间来表示）：

公式 19-2：

PWM 占空比 = (CCPR4L:CCP4CON<5:4>) •
TOSC • (TMR2 预分频值)

可以在任意时刻写入 CCPR4L 和 CCP4CON<5:4>，但是在 PR2 和 TMR2 发生匹配（即周期结束）前占空比值不会被锁存到 CCPR4H 中。在 PWM 模式下，CCPR4H 是只读寄存器。

CCPR4H 寄存器和一个 2 位的内部锁存器用于给 PWM 占空比提供双重缓冲。这种双重缓冲结构非常重要，它可以避免在 PWM 操作中产生毛刺。

当 CCPR4H 和 2 位锁存值与 TMR2 （以及内部 2 位 Q 时钟或 TMR2 预分频值的 2 位）匹配时，CCP4 引脚被清零。

在给定 PWM 频率的情况下，最大的 PWM 分辨率（位）由以下公式给出：

公式 19-3：

$$\text{PWM 分辨率 （最大）} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{位}$$

注： 如果 PWM 占空比的值比 PWM 周期长，则 CCP4 引脚将不会被清零。

表 19-4： 40 MHz 时的 PWM 频率和分辨率示例

PWM 频率	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
定时器预分频值（1、4 和 16）	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最大分辨率 （位）	10	10	10	8	7	6.58

19.4.3 设置 PWM 操作

以 CCP4 为例，配置 CCP 模块使之工作于 PWM 模式：

1. 通过写 PR2 寄存器设置 PWM 周期。
2. 通过写 CCPR4L 寄存器和 CCP4CON<5:4> 位设置 PWM 占空比。
3. 通过清零相应的 TRIS 位将 CCP4 引脚设为输出引脚。
4. 通过写 T2CON 设置 TMR2 预分频值并随后使能 Timer2。
5. 配置 CCP4 模块使之工作于 PWM 模式。

表 19-5：　　与 PWM 和定时器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
RCON	IPEN	SBOREN	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIE4	TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
TMR2	Timer2 寄存器							
TMR4	Timer4 寄存器							
PR2	Timer2 周期寄存器							
PR4	Timer4 周期寄存器							
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节							
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节							
CCPR3L	捕捉 / 比较 /PWM 寄存器 3 的低字节							
CCPR3H	捕捉 / 比较 /PWM 寄存器 3 的高字节							
CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节							
CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节							
CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节							
CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节							
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0
CCP3CON	—	—	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0
CCPTMRS	—	—	—	C5TSEL	C4TSEL	C3TSEL	C2TSEL	C1TSEL
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD

图注：　　— = 未实现，读为 0。 PWM 或 Timer2/4 不使用阴影单元。

20.0 增强型捕捉 / 比较 /PWM （ECCP）模块

PIC18F66K80 系列器件具有一个增强型捕捉 / 比较 / PWM （ECCP）模块：ECCP1。这些模块包含一个 16 位寄存器，可用作 16 位捕捉寄存器、16 位比较寄存器或 PWM 主 / 从占空比寄存器。这些 ECCP 模块与 CCP 模块兼容。

ECCP1 实现为具有增强型 PWM 功能的标准 CCP 模块。这些功能包括：

- 提供 2 路或 4 路输出通道
- 输出转向模式
- 可编程极性
- 可编程死区控制
- 自动关闭和重启

第 20.4 节 “PWM（增强型模式）” 将详细讨论增强功能。

ECCP1 模块使用控制寄存器 CCP1CON。控制寄存器 CCP2CON 至 CCP5CON 用于模块 CCP2 至 CCP5。

寄存器 20-1：　　CCP1CON：增强型捕捉 / 比较 /PWM1 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
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　**P1M<1:0>**：增强型 PWM 输出配置位

如果 CCP1M<3:2> = 00、 01 和 10：

xx = P1A 配置为捕捉 / 比较输入 / 输出； P1B、 P1C 和 P1D 配置为端口引脚

如果 CCP1M<3:2> = 11：

00 = 单输出：P1A、 P1B、 P1C 和 P1D 通过转向控制 （见**第 20.4.7 节 “ 脉冲转向模式 ”**）
01 = 全桥正向输出：P1D 被调制； P1A 有效； P1B 和 P1C 无效
10 = 半桥输出：P1A 和 P1B 被调制，带有死区控制； P1C 和 P1D 配置为端口引脚
11 = 全桥反向输出：P1B 被调制； P1C 有效； P1A 和 P1D 无效

bit 5-4　**DC1B<1:0>**：PWM 占空比 bit 1 和 bit 0

捕捉模式：

未使用。

比较模式：

未使用。

PWM 模式：

这两位是 10 位 PWM 占空比的低 2 位。占空比的高 8 位在 CCPR1L 中。

bit 3-0　**CCP1M<3:0>**：ECCP1 模式选择位

0000 = 捕捉 / 比较 /PWM 关闭（复位 ECCP1 模块）
0001 = 保留
0010 = 比较模式，匹配时输出电平翻转
0011 = 捕捉模式
0100 = 捕捉模式，每个下降沿
0101 = 捕捉模式，每个上升沿
0110 = 捕捉模式，每 4 个上升沿
0111 = 捕捉模式，每 16 个上升沿
1000 = 比较模式，初始化 ECCP1 引脚为低电平，比较匹配时输出置 1（CCP1IF 置 1）
1001 = 比较模式，初始化 ECCP1 引脚为高电平，比较匹配时输出清零（CCP1IF 置 1）
1010 = 比较模式，仅产生软件中断， ECCP1 引脚恢复到 I/O 状态
1011 = 比较模式，触发特殊事件 （ECCP1 复位 TMR1 或 TMR3，启动 A/D 转换， CCP1IF 位置 1）
1100 = PWM 模式； P1A 和 P1C 高电平有效； P1B 和 P1D 高电平有效
1101 = PWM 模式； P1A 和 P1C 高电平有效； P1B 和 P1D 低电平有效
1110 = PWM 模式； P1A 和 P1C 低电平有效； P1B 和 P1D 高电平有效
1111 = PWM 模式； P1A 和 P1C 低电平有效； P1B 和 P1D 低电平有效

寄存器 20-2：　　CCPTMRS：CCP 定时器选择寄存器

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	C5TSEL	C4TSEL	C3TSEL	C2TSEL	C1TSEL
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　**C5TSEL：**CCP5 定时器选择位
0 = CCP5 基于 TMR1/TMR2 产生
1 = CCP5 基于 TMR3/TMR4 产生

bit 3　**C4TSEL：**CCP4 定时器选择位
0 = CCP4 基于 TMR1/TMR2 产生
1 = CCP4 基于 TMR3/TMR4 产生

bit 2　**C3TSEL：**CCP3 定时器选择位
0 = CCP3 基于 TMR1/TMR2 产生
1 = CCP3 基于 TMR3/TMR4 产生

bit 1　**C2TSEL：**CCP2 定时器选择位
0 = CCP2 基于 TMR1/TMR2 产生
1 = CCP2 基于 TMR3/TMR4 产生

bit 0　**C1TSEL：**CCP1 定时器选择位
0 = ECCP1 基于 TMR1/TMR2 产生
1 = ECCP1 基于 TMR3/TMR4 产生

除了可使用 CCP1CON 和 ECCP1AS 寄存器提供的扩展模式外，ECCP 模块还有两个与增强型 PWM 操作和自动关闭功能相关的寄存器。它们是：

- ECCP1DEL—— 增强型 PWM 控制
- PSTR1CON—— 脉冲转向控制

20.1 ECCP 输出和配置

取决于所选定的工作模式，增强型 CCP 模块最多有 4 路 PWM 输出。CCP1CON 寄存器改为支持控制 4 个 PWM 输出：ECCP1/P1A、P1B、P1C 和 P1D。应用可以使用其中的 1 路、 2 路或 4 路输出。

输出根据选定的 ECCP 工作模式被激活。表 20-2 中总结了引脚分配。

若要将 I/O 引脚配置为 PWM 输出，必须通过设置 P1M<1:0> 和 CCP1M<3:0> 位来选择适当的 PWM 模式。端口引脚的相应 TRIS 方向位也必须设置为输出。

20.1.1 ECCP 模块和定时器资源

ECCP模块根据选定的模式使用Timer1、Timer2、Timer3 和 Timer4。在捕捉、比较或 PWM 模式下，这些定时器可供 CCP 模块使用，如表 20-1 所示。

表 20-1： ECCP 模式 —— 定时器资源

ECCP 模式	定时器资源
捕捉	Timer1 或 Timer3
比较	Timer1 或 Timer3
PWM	Timer2 或 Timer4

要将哪个特定的定时器分配给ECCP模块由CCPTMRS寄存器 （寄存器 20-2）中的 “ECCP 的定时器 ” 使能位决定。图 20-1 描述了这两种模块间的相互关系。捕捉操作设计为在定时器配置为同步计数器模式时使用。如果关联的定时器配置为异步计数器模式，捕捉操作可能不会按预期工作。

20.2 捕捉模式

在捕捉模式下，当相应的ECCP1引脚发生以下事件时，CCPR1H:CCPR1L 寄存器对捕捉 TMR1 或 TMR3 寄存器的 16 位值。事件定义为以下情况之一：

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

事件由模式选择位CCP1M<3:0>（CCP1CON<3:0>）选择。当完成一次捕捉时，中断请求标志位 CCP1IF（PIR3<1>）置 1。该标志必须用软件清零。如果在读取 CCPR1H/L 寄存器中的值之前发生了另一次捕捉，那么原来的捕捉值会被新的捕捉值覆盖。

20.2.1 ECCP 引脚配置

在捕捉模式下，应通过将相应的 TRIS 方向位置 1，将ECCP1 引脚配置为输入。

注：	如果ECCP1引脚被配置为输出，则写端口将产生一次捕捉条件。

20.2.2 TIMER1/2/3/4 模式选择

用于捕捉功能的定时器（Timer1、Timer2、Timer3 或 Timer4）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。可在 CCPTMRS 寄存器（寄存器 20-2）中选择用于每个 ECCP 模块的定时器。

20.2.3 软件中断

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 CCP1IE 中断允许位清零以避免错误中断。应在工作模式改变后清零中断标志位 CCP1IF。

20.2.4 ECCP 预分频器

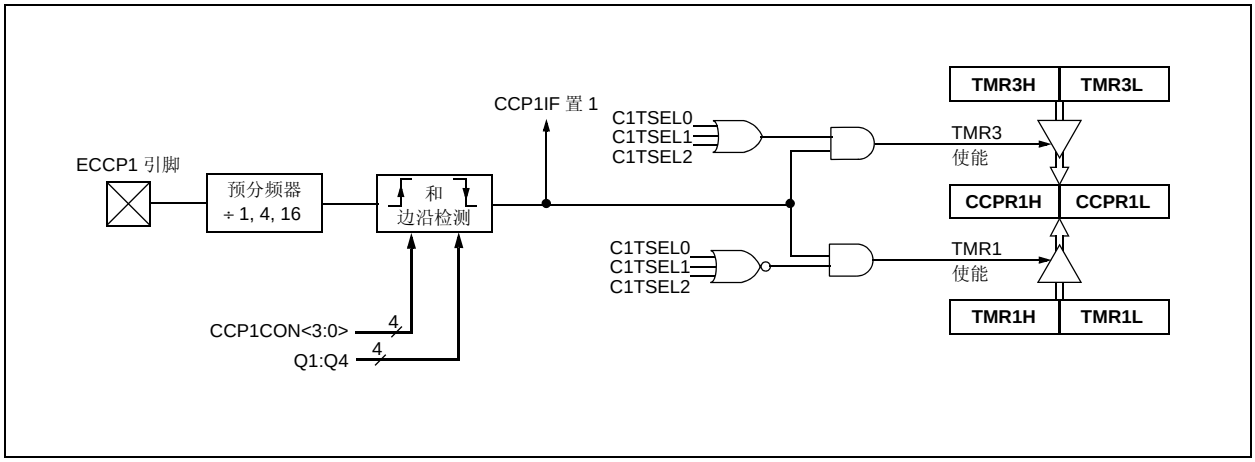
在捕捉模式下有 4 种预分频比设置；它们作为工作模式的一部分由模式选择位（CCP1M<3:0>）指定。只要关闭 ECCP 模块或禁止捕捉模式，预分频器计数器就会被清零。这意味着任何复位都会将预分频器计数器清零。

在两个捕捉预分频比之间切换可能会产生中断。而且，预分频器计数器不会被清零；因此，第一次捕捉可能来自于一个非零的预分频器。例 20-1 给出了切换捕捉预分频比时建议采用的方法。这个示例使预分频器计数器清零且不会产生错误中断。

例 20-1： 改变捕捉预分频比

```
CLRF   CCP1CON     ; Turn ECCP module off
MOVLW  NEW_CAPT_PS ; Load WREG with the
                   ; new prescaler mode
                   ; value and ECCP ON
MOVWF  CCP1CON     ; Load ECCP1CON with
                   ; this value
```

图 20-1： 捕捉模式工作原理框图



20.3 比较模式

在比较模式下，16 位 CCPR1 寄存器的值不断地与 CCPTMR1 寄存器中选择的定时器寄存器对的值作比较。当两者匹配时， ECCP1 引脚将会：

- 驱动为高电平
- 驱动为低电平
- 翻转 （高电平变为低电平或低电平变为高电平）
- 不变 （即反映 I/O 锁存器的状态）

引脚动作取决于模式选择位（CCP1M<3:0>）的值。同时，中断标志位 CCP1IF 置 1。

20.3.1 ECCP 引脚配置

用户必须通过将相应的 TRIS 位清零，将 ECCP1 引脚配置为输出。

注：	清零 CCP1CON 寄存器会将 ECCP1 比较输出锁存器 （取决于器件配置）强制为默认的低电平。这不是端口 I/O 数据锁存器。

20.3.2 TIMER1/2/3/4 模式选择

如果 ECCP 模块使用比较功能，则 Timer1、 Timer2、Timer3 或 Timer4 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。

20.3.3 软件中断模式

当选择了“产生软件中断”模式（CCP1M<3:0> = 1010）时，ECCP1引脚不受影响；只会影响CCP1IF中断标志。

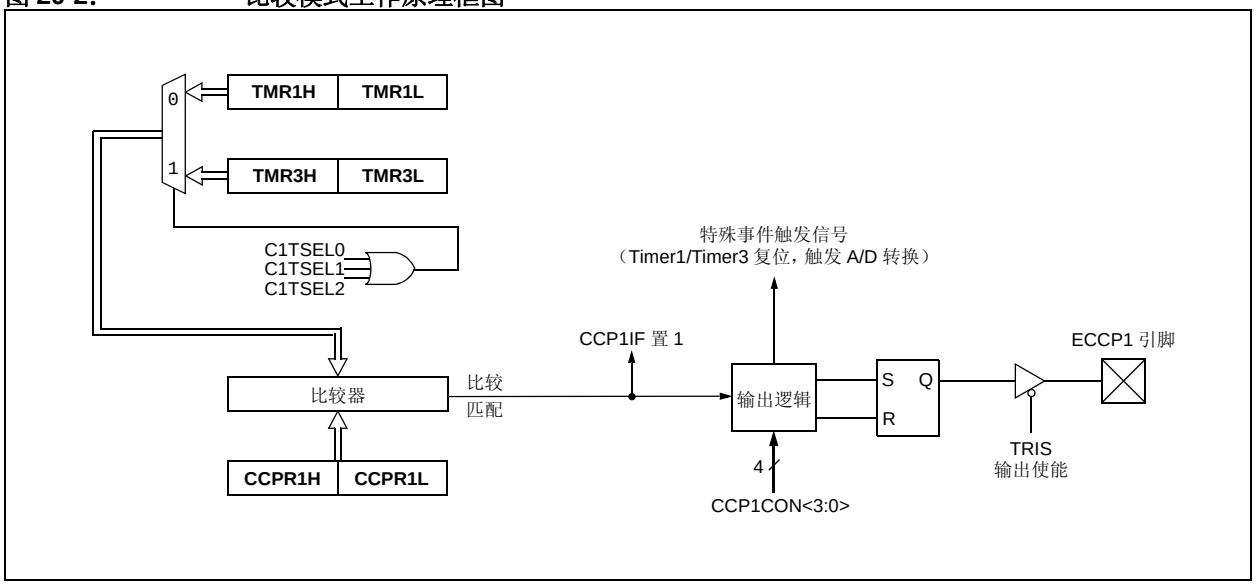
20.3.4 特殊事件触发器

ECCP 模块配备了一个特殊事件触发器。在比较模式下可产生内部硬件信号以触发其他模块动作。通过选择“比较特殊事件触发”模式 （CCP1M<3:0> = 1011），使能特殊事件触发器。

无论当前使用哪个定时器资源作为模块的时基，特殊事件触发信号会将对应的定时器寄存器对复位。这样 CCPR1 寄存器可用作任一定时器的可编程周期寄存器。

特殊事件触发信号还能启动 A/D 转换。要实现此功能，必须首先使能 A/D 转换器。

图 20-2： 比较模式工作原理框图



20.4 PWM （增强型模式）

增强型 PWM 模式可以在最多 4 个输出引脚上产生 PWM 信号，最高可达 10 位分辨率。可通过 4 种 PWM 输出模式实现：

- 单 PWM
- 半桥 PWM
- 全桥 PWM，正向模式
- 全桥 PWM，反向模式

要选择增强型 PWM 模式，必须正确设置 CCP1CON 寄存器的 P1M 位。

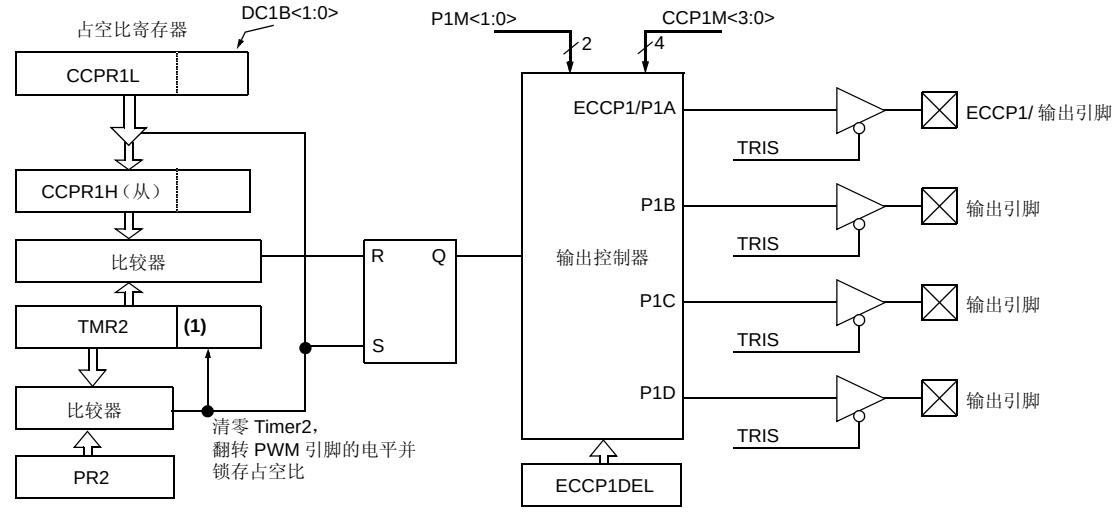
PWM 输出与 I/O 引脚复用，指定为 P1A、P1B、P1C 和 P1D。 PWM 引脚的极性是可配置的，通过恰当设置 CCP1CON 寄存器中的 CCP1M 位来选择。

表 20-1 给出了每种增强型 PWM 模式的引脚分配。

图 20-3 给出了增强型 PWM 模块的简化框图的示例。

注：	为防止在最开始使能 PWM 时产生不完整的波形， ECCP 模块在产生 PWM 信号前会等待直到新的 PWM 周期开始。

图 20-3： 增强型 PWM 模式的简化框图示例



注 1： 8 位 TMR2 寄存器的值与 2 位内部 Q 时钟或预分频器中的 2 位一起构成 10 位时基。

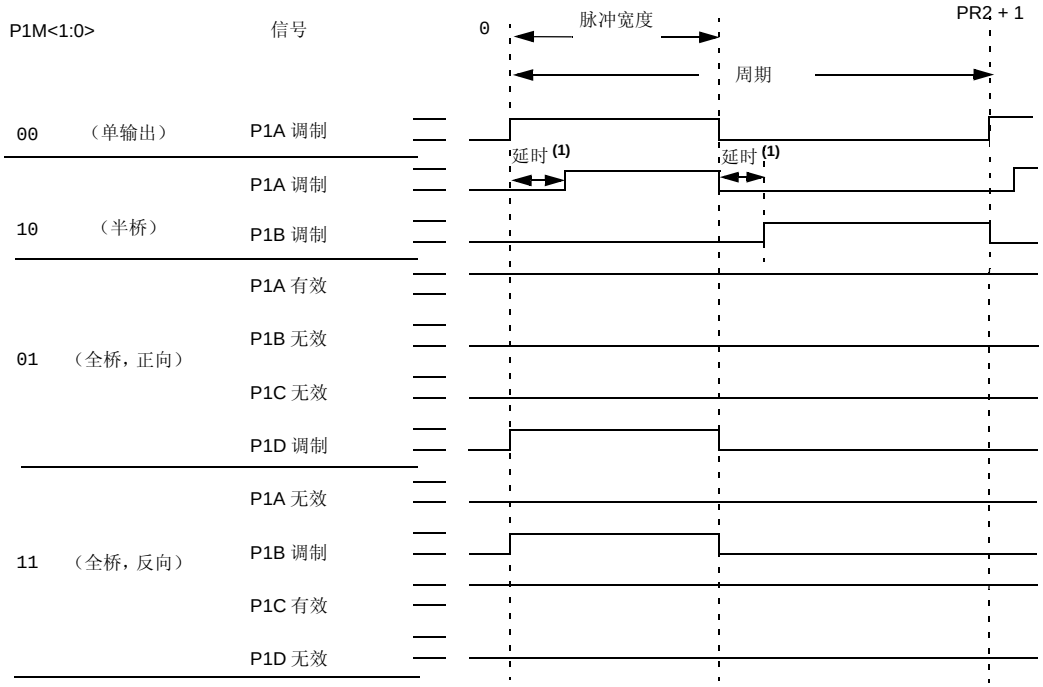
注 1：	每个 PWM 输出的 TRIS 寄存器值必须进行适当配置。
2：	增强型 PWM 模式没有使用的任何引脚均可用于备用引脚功能。

表 20-2：　　各种 PWM 增强型模式的引脚分配示例

ECCP 模式	P1M<1:0>	P1A	P1B	P1C	P1D
单 PWM	00	使用 (1)	使用 (1)	使用 (1)	使用 (1)
半桥	10	使用	使用	不使用	不使用
全桥，正向	01	使用	使用	使用	使用
全桥，反向	11	使用	使用	使用	使用

注　1： 在单输出模式下，输出通过脉冲转向使能（见寄存器 20-5）。

图 20-4：　　PWM（增强型模式）输出关系示例（高电平有效状态）

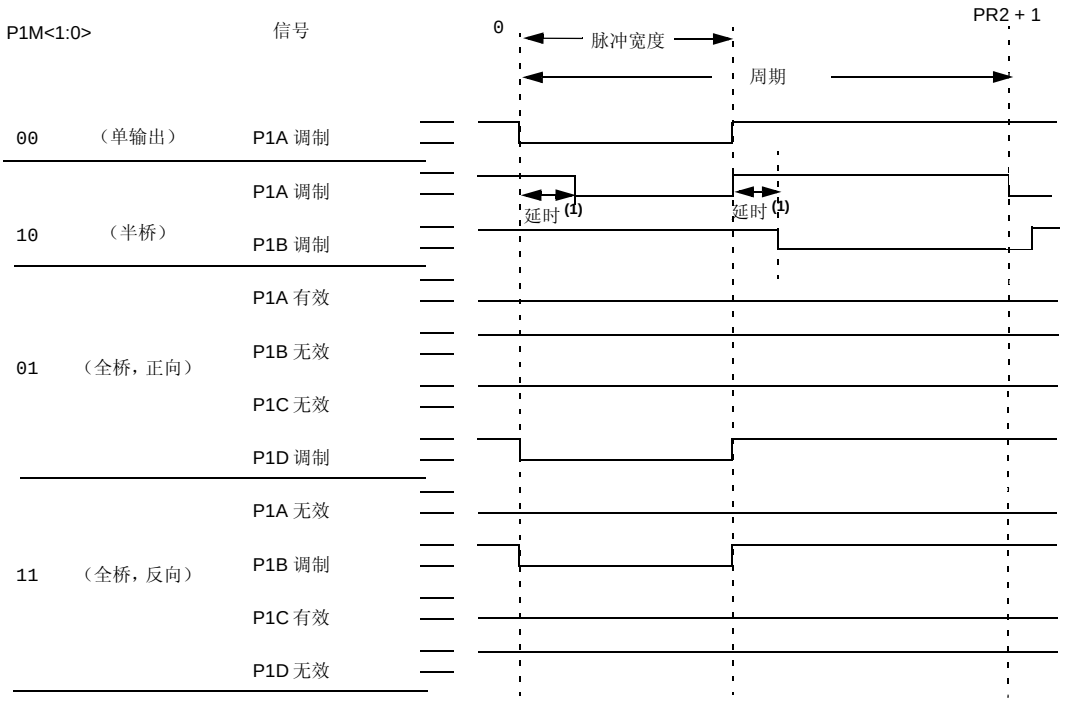


关系：

- 周期 = 4 * Tosc * (PR2 + 1) * (TMR2 预分频值)
- 脉冲宽度 = Tosc * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 预分频值)
- 延时 = 4 * Tosc * (ECCP1DEL<6:0>)

注　1： 使用 ECCP1DEL 寄存器对死区延时编程（**第 20.4.6 节 " 可编程死区延时模式 "**）。

图 20-5：　　增强型 PWM 输出关系示例 （低电平有效状态）



关系：

- 周期 = 4 * Tosc * (PR2 + 1) * (TMR2 预分频值)
- 脉冲宽度 = Tosc * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 预分频值)
- 延时 = 4 * Tosc * (ECCP1DEL<6:0>)

注　　1：　使用 ECCP1DEL 寄存器对死区延时编程（**第 20.4.6 节 " 可编程死区延时模式 "**）。

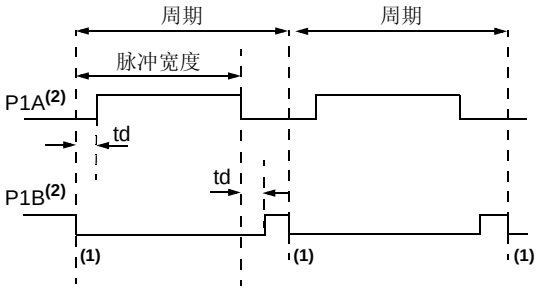
20.4.1 半桥模式

在半桥模式下，有两个引脚用作输出驱动推挽式负载。P1A 引脚输出 PWM 输出信号，P1B 引脚输出互补的 PWM 输出信号（见图 20-6）。这种模式可用于半桥应用（如图 20-7 所示）；或者用于全桥应用，在全桥应用中使用两个 PWM 信号调制 4 个功率开关。

在半桥模式下，可编程死区延时可用来防止半桥功率器件中流过直通电流。ECCP1DEL 寄存器的 P1DC<6:0> 位的值设置在输出被驱动为有效之前的指令周期数。如果这个值比占空比大，则在整个周期中相应的输出保持为无效。关于死区延时操作的更多详细信息，请参见**第 20.4.6 节 “ 可编程死区延时模式 ”**。

由于 P1A 和 P1B 输出与端口数据锁存器是复用的，相关的TRIS位必须清零，从而将P1A和P1B配置为输出。

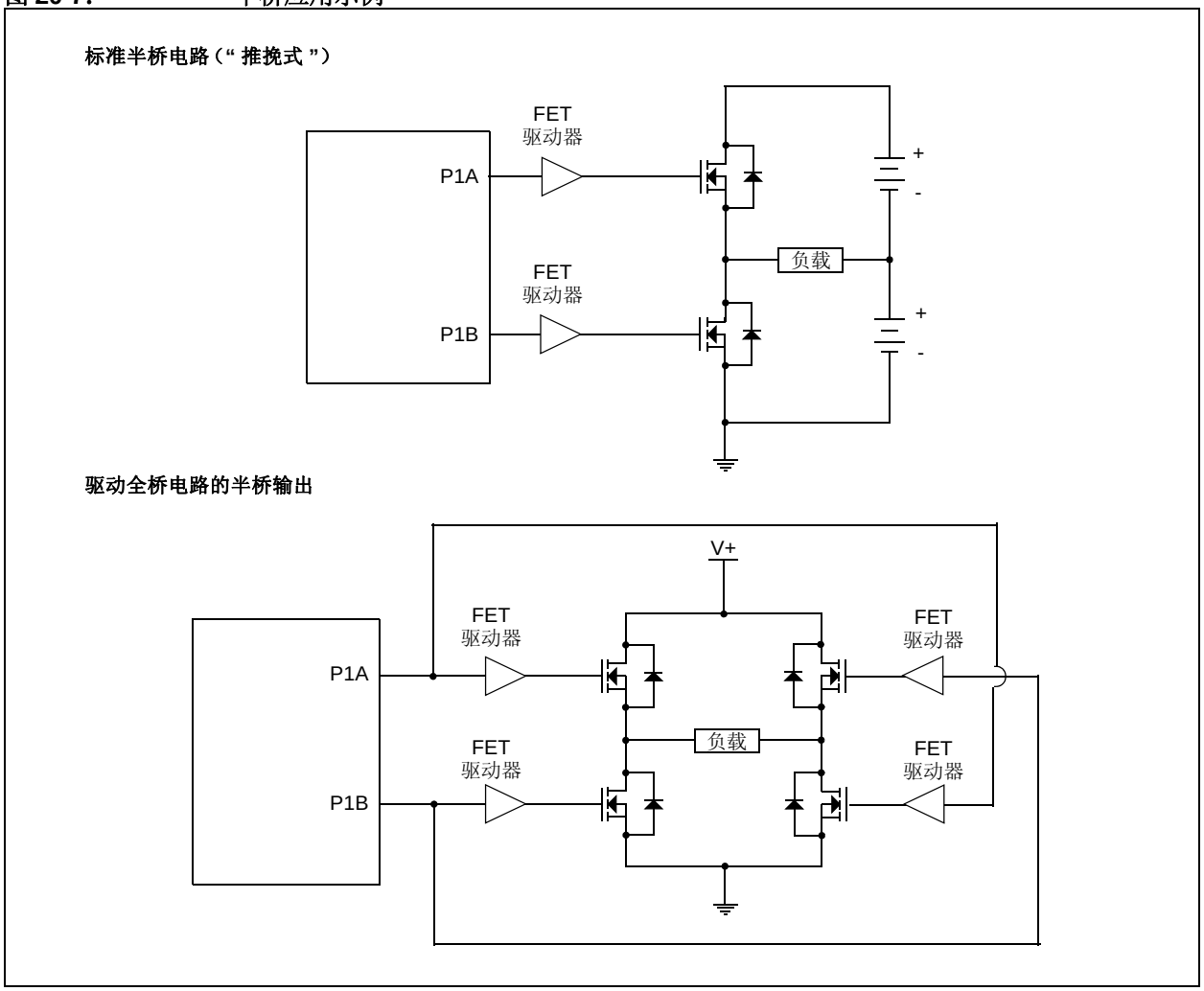
图 20-6： 半桥 PWM 输出示例



注 1： 此时 TMR2 寄存器和 PR2 寄存器的值相等。

2： 图中输出信号为高电平有效。

图 20-7： 半桥应用示例



20.4.2 全桥模式

在全桥模式下， 4 个引脚都用作输出。全桥应用的示例如图 20-8 所示。

在正向模式下，P1A 引脚被驱动为有效状态，P1D 引脚被调制，而 P1B 和 P1C 引脚将被驱动为无效状态，如图 20-9 所示。

在反向模式下，P1C 引脚被驱动为有效状态，P1B 引脚被调制，而 P1A 和 P1D 引脚将被驱动为无效状态，如图 20-9 所示。

P1A、P1B、P1C 和 P1D 输出与端口数据锁存器复用。相关的 TRIS 位必须清零，从而将 P1A、P1B、P1C 和 P1D 引脚配置为输出。

图 20-8： 全桥应用示例

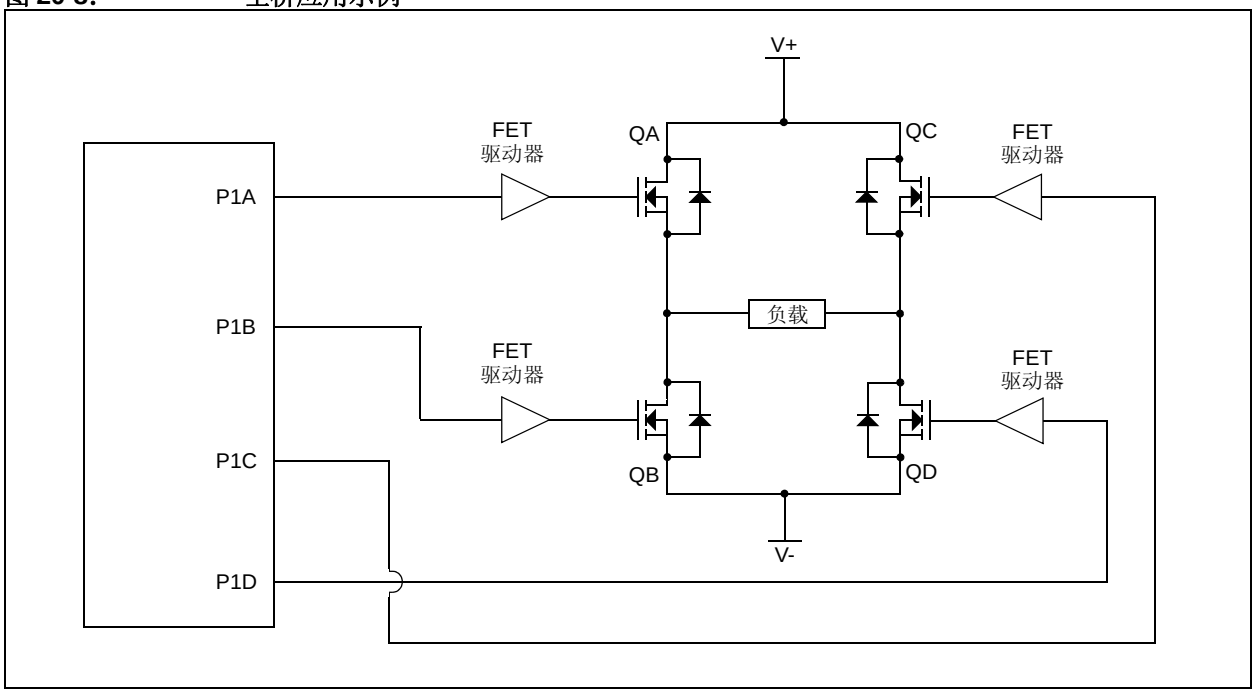
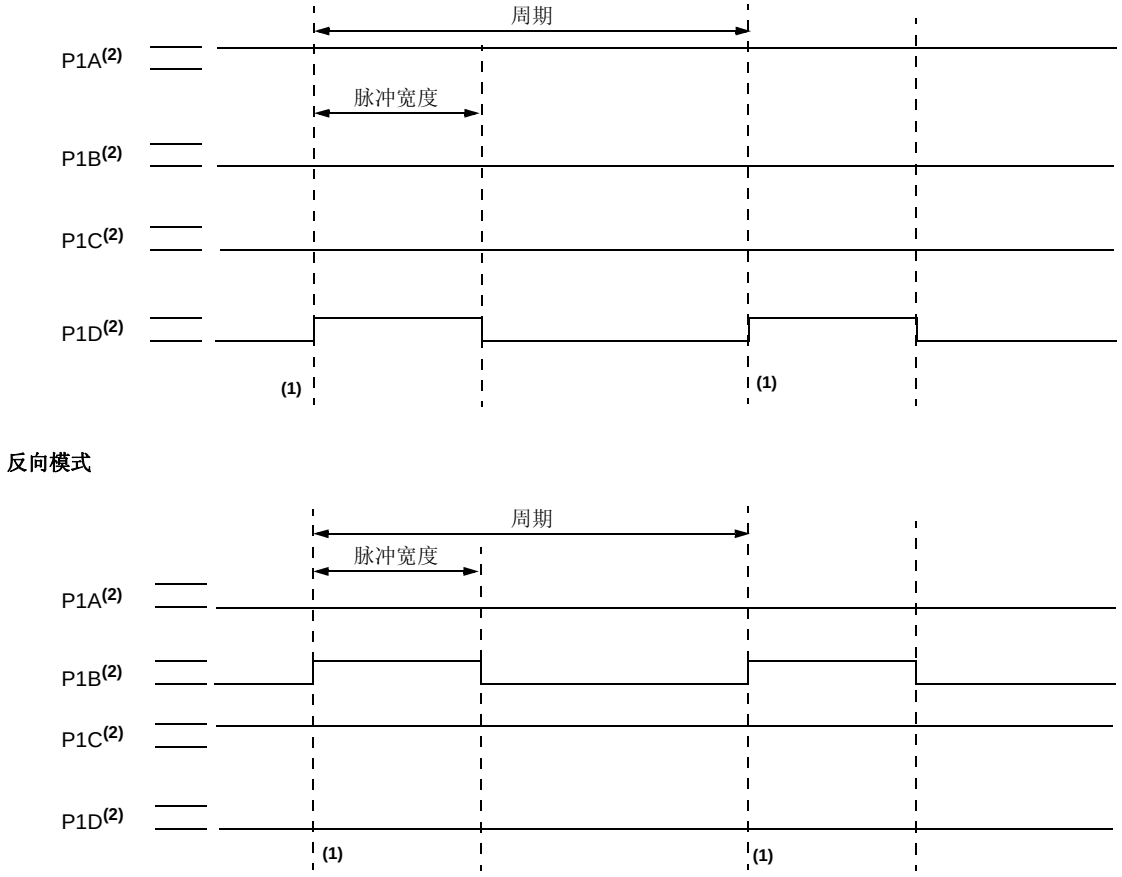


图 20-9：　　全桥 PWM 输出示例



注　1： 此时 TMR2 寄存器和 PR2 寄存器的值相等。

2： 图中输出信号为高电平有效。

20.4.2.1 全桥模式中的方向改变

在全桥模式下，CCP1CON 寄存器中的 P1M1 位允许用户控制正/反方向。当应用软件改变这个方向控制位时，模块将在下一个 PWM 周期改用新的方向。

通过改变 CCP1CON 寄存器的 P1M1 位，可以用软件启动方向改变。以下序列在当前 PWM 周期结束前发生：

- 调制输出 （P1B 和 P1D）进入无效状态。
- 相关的未调制输出 （P1A 和 P1C）被切换到以相反的方向驱动。
- PWM 调制在下一个周期开始继续。

关于该序列的说明，请参见图 20-10。

全桥模式不提供死区延时。因为一次只有一个输出被调制，所以一般不需要死区延时。有一种情况需要死区延时，这一情况发生在以下两个条件同时满足时：

- 当输出的占空比达到或者接近 100%，PWM 输出方向改变。
- 功率开关（包括功率器件和驱动电路）的关断时间比导通时间要长。

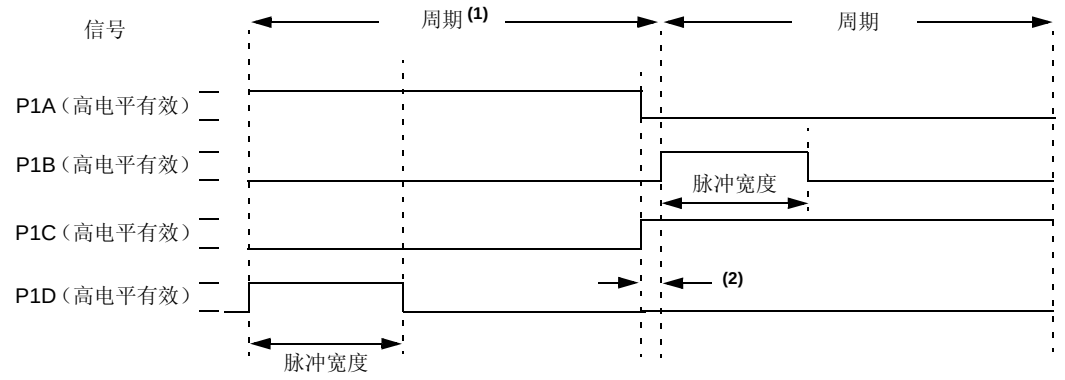
在图 20-11 所示的示例中，在占空比接近 100% 时，PWM 方向从正向改变到反向。在这个示例中，在时间 t1，P1A 和 P1D 输出变为无效，而 P1C 输出变为有效。因为功率器件的关断时间比导通时间要长，在 “t” 时间内，功率器件 QC 和 QD 中可能流过直通电流 （见图 20-8）。当 PWM 方向从反向改变到正向时，功率器件 QA 和 QB 也将出现相同的现象。

如果应用中需要在高占空比时改变 PWM 方向，避免直通电流可采用以下两种方法：

- 在改变方向之前的一个 PWM 周期降低 PWM 占空比。
- 使用开关驱动电路，使开关的关断时间比导通时间短。

也可能存在其他避免直通电流的方案。

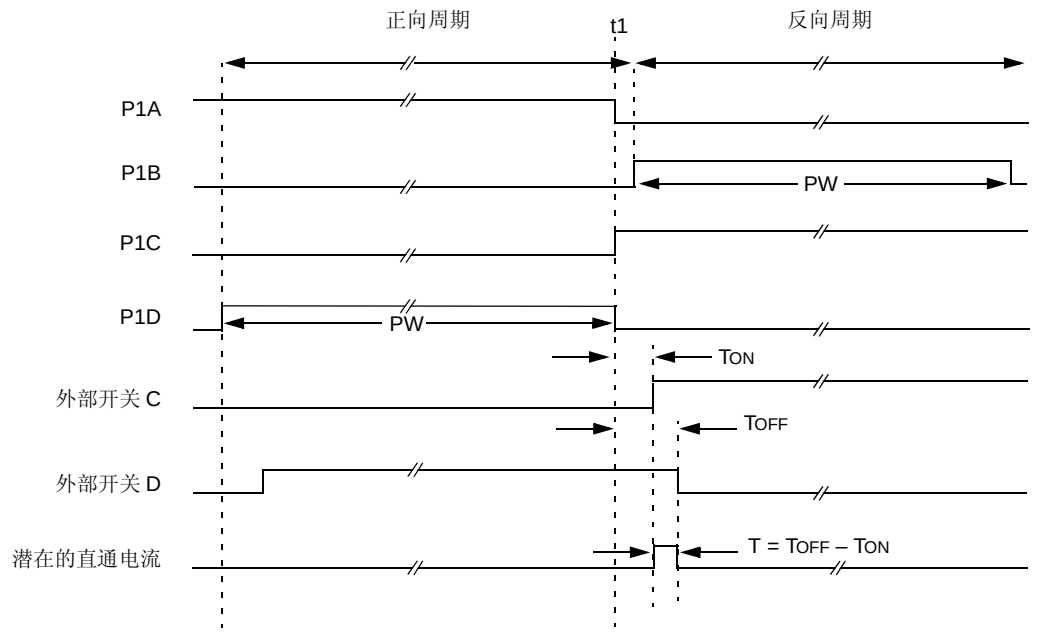
图 20-10： PWM 方向改变的示例



注 1： 可在 PWM 周期的任意时刻，写 CCP1CON 寄存器的方向位 P1M1。

2： 当改变方向时， P1A 和 P1C 信号在当前 PWM 周期结束之前切换。此时被调制的 P1B 和 P1D 信号是无效的。该时间长度为 (1/FOSC) • TMR2 预分频值。

图 20-11：　　在占空比接近 100% 时改变 PWM 方向的示例



注　1： 图中所有信号均为高电平有效。

2： TON 是功率开关 QC 及其驱动电路的导通延时。

3： TOFF 是功率开关 QD 及其驱动电路的关断延时。

20.4.3　启动注意事项

当使用任何 PWM 模式时，应用硬件必须在 PWM 输出引脚上外接适当的上拉和 / 或下拉电阻。

注：	当单片机退出复位状态时，所有 I/O 引脚呈高阻态。外部电路必须保持功率开关器件处于截止状态，直到单片机将 I/O 引脚驱动为适当的信号电平，或者激活 PWM 输出为止。

CCP1CON 寄存器的 CCP1M<1:0> 位允许用户为每一对 PWM 输出引脚（P1A/P1C 和 P1B/P1D）选择 PWM 输出信号是高电平有效还是低电平有效。PWM 输出极性必须在使能 PWM 引脚输出驱动器之前选择。由于可能导致应用电路的损坏，因此不推荐在使能 PWM 引脚输出驱动器的同时改变极性配置。

当 PWM 模块初始化时，P1A、P1B、P1C 和 P1D 输出锁存器可能不在正确的状态。这样在使能增强型 PWM 模式的同时使能 PWM 引脚输出驱动器，可能损坏应用电路。应首先将增强型 PWM 模式配置为正确的输出模式并经过一个完整的 PWM 周期之后，再使能 PWM 引脚输出驱动器。当第二个 PWM 周期开始时， PIR1 或 PIR4 寄存器的 TMR2IF 或 TMR4IF 位置 1 表明一个完整的 PWM 周期结束了。

20.4.4　增强型 PWM 自动关闭模式

PWM 模式支持自动关闭模式，当外部关闭事件发生时将禁止 PWM 输出。自动关闭模式将 PWM 输出引脚置于预先确定的状态。该模式用于防止 PWM 损坏应用。

通过使用 ECCP1AS<2:0> 位（ECCP1AS<6:4>）来选择自动关闭源。关闭事件由以下条件产生：

- 分配为 FLT0 输入功能的引脚上出现逻辑 0
- 比较器 C1
- 比较器 C2
- 用固件将 ECCP1ASE 位置 1

关闭条件由 ECCP1ASE（自动关闭事件状态）位（ECCP1AS<7>）指示。如果该位为 0，PWM 引脚正常工作。如果该位为 1，PWM 输出处于关闭状态。

当关闭事件发生时，会发生以下两个事件：

- ECCP1ASE 位被设置为 1。ECCP1ASE 将保持置 1 直到由固件清零或发生自动重启（见**第 20.4.5 节 “自动重启模式”**）。
- 使能的 PWM 引脚被陆续置为其关闭状态。PWM 输出引脚被分组为（P1A/P1C）和（P1B/P1D）对。每对引脚的状态由 PSS1AC 和 PSS1BD 位（分别为 ECCP1AS<3:2> 和 <1:0>）决定。

每对引脚可以设置为以下 3 种状态之一：

- 驱动逻辑 1
- 驱动逻辑 0
- 三态 （高阻态）

寄存器 20-3： ECCP1AS：ECCP1 自动关闭控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0
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 **ECCP1ASE**：ECCP 自动关闭事件状态位
1 = 发生了关闭事件； ECCP 输出为关闭状态
0 = ECCP 输出正常工作

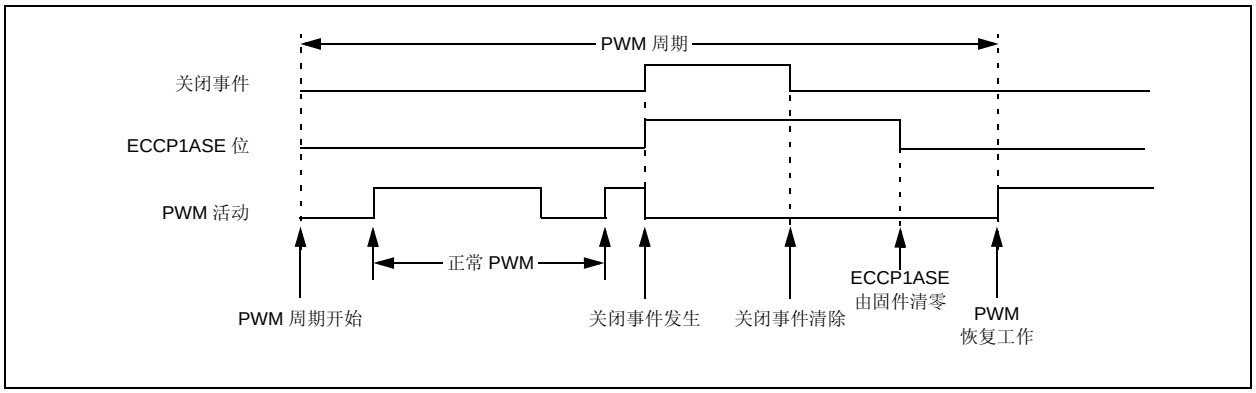
bit 6-4 **ECCP1AS<2:0>**：ECCP 自动关闭源选择位
000 = 禁止自动关闭
001 = 比较器 C1OUT 输出为高电平
010 = 比较器 C2OUT 输出为高电平
011 = 比较器 C1OUT 或 C2OUT 为高电平
100 = FLT0 引脚电压为 VIL
101 = FLT0 引脚电压为 VIL 或比较器 C1OUT 输出为高电平
110 = FLT0 引脚电压为 VIL 或比较器 C2OUT 输出为高电平
111 = FLT0 引脚电压为 VIL，或者比较器 C1OUT 或比较器 C2OUT 为高电平

bit 3-2 **PSS1AC<1:0>**：P1A 和 P1C 引脚关闭状态控制位
00 = 驱动引脚 P1A 和 P1C 为 0
01 = 驱动引脚 P1A 和 P1C 为 1
1x = 引脚 P1A 和 P1C 为三态

bit 1-0 **PSS1BD<1:0>**：P1B 和 P1D 引脚关闭状态控制位
00 = 驱动引脚 P1B 和 P1D 为 0
01 = 驱动引脚 P1B 和 P1D 为 1
1x = 引脚 P1B 和 P1D 为三态

注 1：自动关闭条件是基于电平的信号，而不是基于边沿的信号。只要电平存在，自动关闭就将持续。
2：当自动关闭条件存在时，禁止写 ECCP1ASE 位。
3：一旦自动关闭条件被移除并且 PWM 重启 （通过固件或自动重启）， PWM 信号将总是在下一个 PWM 周期重启。

图 20-12： PWM 自动关闭，可用固件重启 （P1RSEN = 0）



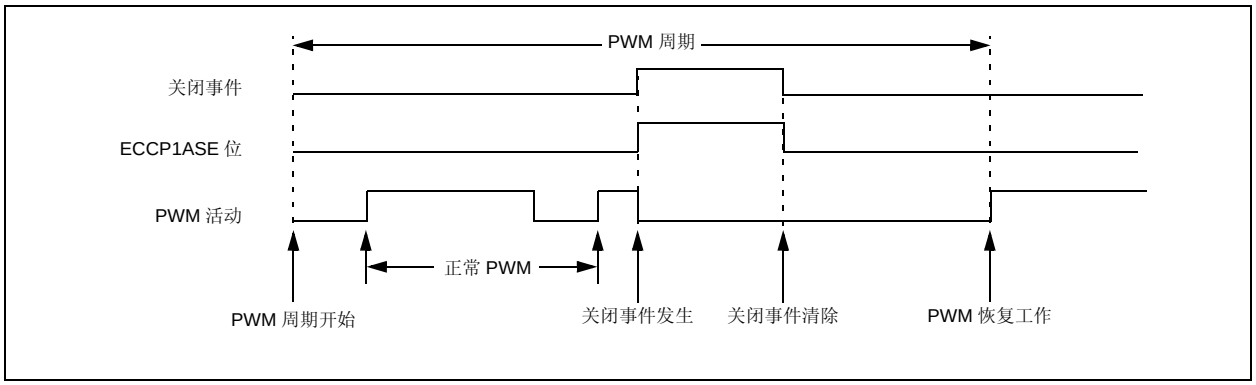
20.4.5 自动重启模式

一旦自动关闭条件被移除，增强型 PWM 可被配置为自动重启PWM信号。通过将P1RSEN位(ECCP1DEL<7>)置 1 使能自动重启。

如果使能了自动重启，只要自动关闭条件有效，ECCP1ASE 位将保持置 1。当自动关闭条件被移除时，ECCP1ASE 位将由硬件清零并恢复正常工作。

但模块将一直等到下一个 PWM 周期开始，然后再重新使能输出引脚。这种行为使得可以在基于当前模式 PWM 控制的应用中使用自动关闭与自动重启功能。

图 20-13： PWM 自动关闭，可使能自动重启 （P1RSEN = 1）

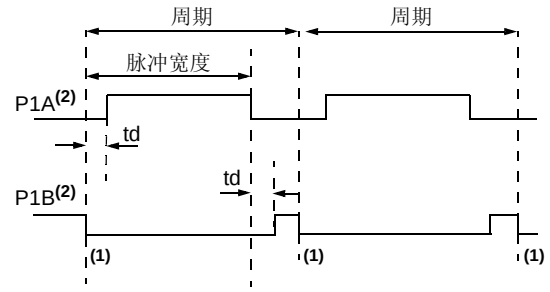


20.4.6 可编程死区延时模式

在所有功率开关都以 PWM 频率调制的半桥应用中，功率开关关断通常比导通需要更多的时间。如果上下两个功率开关同时开关（一个导通，另一个关断），那么在一段很短的时间里，两个开关可能同时导通，直到其中一个开关完全关断为止。在这短暂的时间间隔中，两个功率开关中将流过很高的电流（直通电流），致使桥式供电电路短路。为避免开关过程中可能会出现的破坏性直通电流，通常需要延迟功率开关的导通，保证在另一个开关完全关断之后，再导通相应的功率开关。

在半桥模式下，可采用数字可编程死区延时来避免出现破坏桥式功率开关的直通电流。在信号从无效状态切换到有效状态时发生延时。说明请参见图 20-14。相关的 ECCP1DEL 寄存器（寄存器 20-4）的低 7 位以单片机指令周期 （TCY 或 4 TOSC）为单位设置延时。

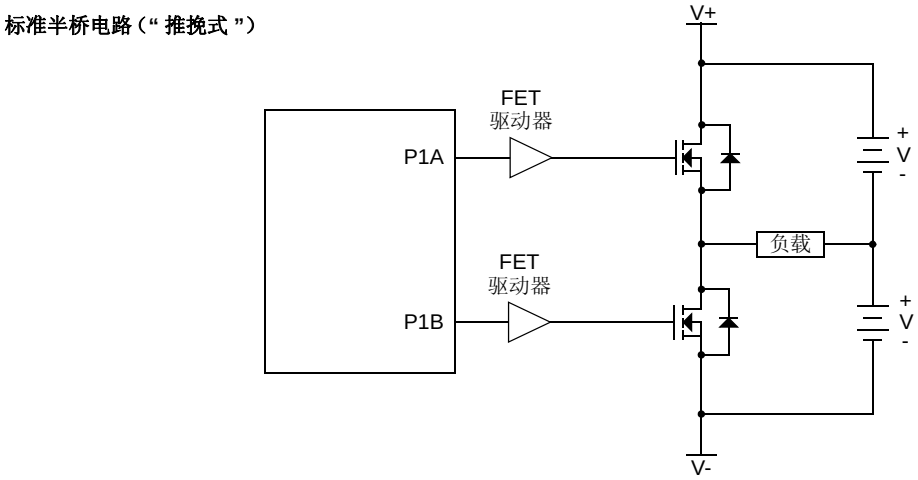
图 20-14： 半桥 PWM 输出示例



td = 死区延时

注 1： 此时 TMR2 寄存器和 PR2 寄存器的值相等。
2： 图中输出信号为高电平有效。

图 20-15： 半桥应用示例



寄存器 20-4：　　ECCP1DEL：增强型 PWM 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0
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　　**P1RSEN：**PWM 重启使能位

1 = 自动关闭时，一旦关闭事件消失， ECCP1ASE 位自动清零； PWM 自动重启

0 = 自动关闭时， ECCP1ASE 必须用软件清零以重启 PWM

bit 6-0　　**P1DC<6:0>：**PWM 延时计数位

P1DCn = 在 PWM 信号**应该**转换为有效的预定时间和转换为有效的**实际**时间之间的 FOSC/4（4 * TOSC）周期数。

20.4.7　脉冲转向模式

在单输出模式下，脉冲转向允许任何 PWM 引脚为调制信号。此外，多个引脚上可以同时使用同一PWM信号。

一旦选择了单输出模式（CCP1CON 寄存器的 CCP1M<3:2> = 11 且 P1M<1:0> = 00），通过设置相应的 STR<D:A> 位（PSTR1CON<3:0>），用户固件可将同一 PWM 信号加到 1、2、3 或 4 个输出引脚，如表 20-2 所示。

注：　必须将相关的 TRIS 位设为输出（0）以使能引脚输出驱动器，从而在引脚上看到 PWM 信号。

当 PWM 转向模式有效时，CCP1M<1:0> 位（CCP1CON<1:0>）将为 P1<D:A> 引脚选择 PWM 输出极性。

PWM 自动关闭操作也适用于 PWM 转向模式，如**第 20.4.4 节 “ 增强型 PWM 自动关闭模式 ”** 中所述。自动关闭事件只对使能 PWM 输出的引脚有影响。

寄存器 20-5： PSTR1CON：脉冲转向控制寄存器 (1)

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA
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 **CMPL<1:0>：**互补模式输出分配转向同步位
00 = 请参见 STR<D:A>
01 = 选择 PA 和 PB 作为互补输出对
10 = 选择 PA 和 PC 作为互补输出对
11 = 选择 PA 和 PD 作为互补输出对

bit 5 **未实现：**读为 0

bit 4 **STRSYNC：**转向同步位
1 = 在下一个 PWM 周期发生输出转向更新
0 = 在指令周期边界的开始发生输出转向更新

bit 3 **STRD：**转向使能位 D
1 = P1D 引脚的 PWM 波形极性受 CCP1M<1:0> 控制
0 = P1D 引脚被分配为端口引脚

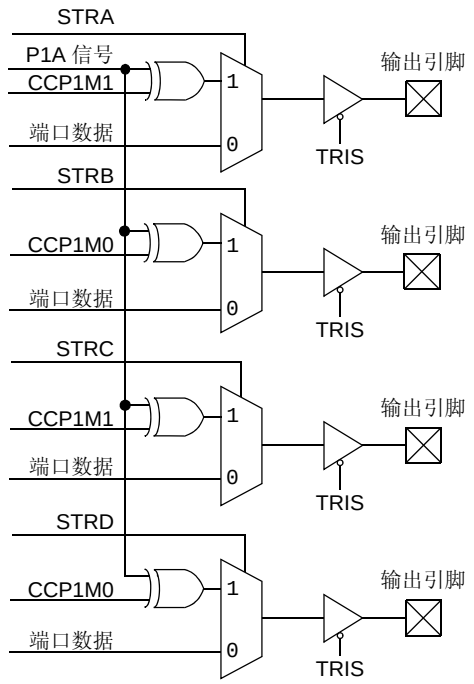
bit 2 **STRC：**转向使能位 C
1 = P1C 引脚的 PWM 波形极性受 CCP1M<1:0> 控制
0 = P1C 引脚被分配为端口引脚

bit 1 **STRB：**转向使能位 B
1 = P1B 引脚的 PWM 波形极性受 CCP1M<1:0> 控制
0 = P1B 引脚被分配为端口引脚

bit 0 **STRA：**转向使能位 A
1 = P1A 引脚的 PWM 波形极性受 CCP1M<1:0> 控制
0 = P1A 引脚被分配为端口引脚

注 1： PWM 转向模式仅在 CCP1CON 寄存器位 CCP1M<3:2> = 11 且 P1M<1:0> = 00 时可用。

图 20-16：　　转向简化框图 (1,2)



注　**1：** 当 CCP1CON 寄存器位 P1M<1:0> = 00 且 CCP1M<3:2> = 11 时，端口输出如图所示配置。

2： 单 PWM 输出需要将至少一个 STR<D:A> 位置 1。

20.4.7.1　转向同步

当转向事件发生时，PSTR1CON 寄存器的 STRSYNC 位向用户提供两种选择。当 STRSYNC 位为 0 时，转向事件将发生在写 PSTR1CON 寄存器指令结束时。在这种情况下，P1<D:A> 引脚的输出信号可能是一个不完整的 PWM 波形。用户固件需要立即从引脚移除 PWM 信号时，该操作非常有用。

当 STRSYNC 位为 1 时，在下一个 PWM 周期的开始将发生有效的转向更新。此时，转向开 / 关 PWM 输出将始终产生一个完整的 PWM 波形。

图 20-17 和图 20-18 是根据 STRSYNC 设置的 PWM 转向时序图。

图 20-17：　　指令结束时发生的转向事件的示例 （STRSYNC = 0）

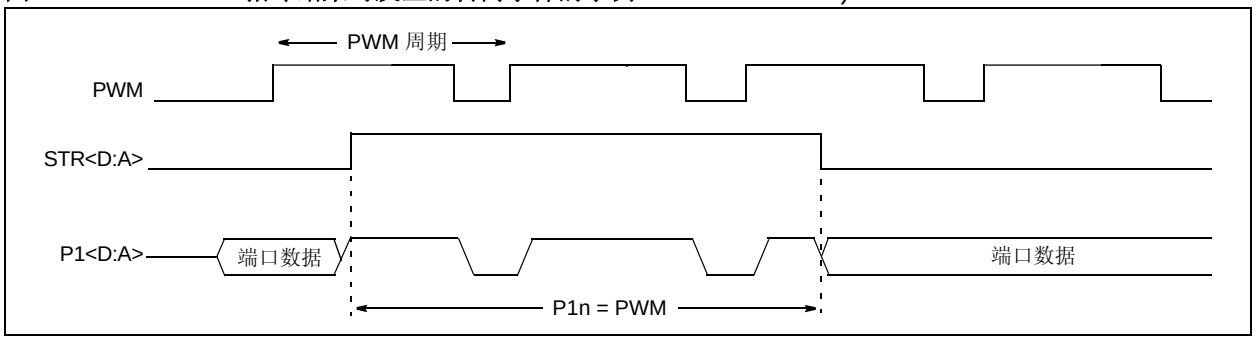
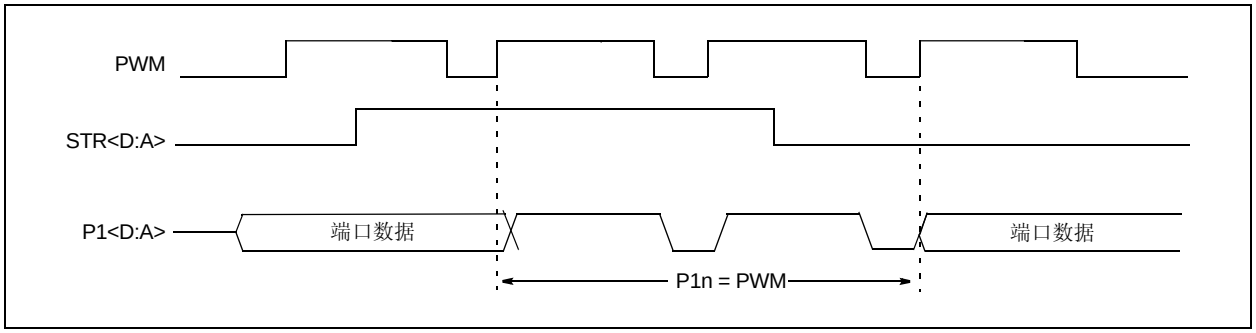


图 20-18：　　指令开始时发生的转向事件的示例 （STRSYNC = 1）



20.4.8 在功耗管理模式下的操作

在休眠模式下，所有时钟源都被禁止。Timer2/4 不再递增，模块的状态也不会改变。如果 ECCP1 引脚正在驱动一个值，则会继续驱动该值。当器件被唤醒时，将从该状态继续。如果使能了双速启动，来自 HF-INTOSC 和后分频器的初始启动频率可能不会立即稳定。

在 PRI_IDLE 模式下，主时钟将继续作为 ECCP1 模块的时钟源，保持不变。

20.4.8.1 故障保护时钟监视器 （FSCM）相关操作

如果使能了故障保护时钟监视器（FSCM），时钟故障将强制器件进入 RC_RUN 功耗管理模式，并将 PIR2 寄存器的 OSCFIF 位置 1。ECCP1 将从内部振荡器时钟源获取时钟信号，该时钟信号的频率可能与主时钟的时钟频率不同。

20.4.9 复位的影响

上电复位及后续的复位都将强制所有端口为输入模式，并强制 ECCP 寄存器为复位状态。

这将强制 ECCP 模块复位为这种状态：与其他 PIC18 和 PIC16 器件中使用的早期版本非增强型 CCP 模块兼容。

表 20-3：　　与 ECCP1 模块和 TIMER1/2/3/4 相关的寄存器

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
RCON	IPEN	SBOREN	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIE4	TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	—	TRISE2	TRISE1	TRISE0
TMR1H	Timer1 寄存器的高字节							
TMR1L	Timer1 寄存器的低字节							
TMR2	Timer2 寄存器							
TMR3H	Timer3 寄存器的高字节							
TMR3L	Timer3 寄存器的低字节							
TMR4	Timer4 寄存器							
PR2	Timer2 周期寄存器							
PR4	Timer4 周期寄存器							
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{T1SYNC}$	RD16	TMR1ON
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{T3SYNC}$	RD16	TMR3ON
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节							
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节							
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节							
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节							
CCPR3H	捕捉 / 比较 /PWM 寄存器 3 的高字节							
CCPR3L	捕捉 / 比较 /PWM 寄存器 3 的低字节							
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0
CCP3CON	—	—	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0
CCPTMRS	—	—	—	C5TSEL	C4TSEL	C3TSEL	C2TSEL	C1TSEL
ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0
ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD

注　　1： 在具有 32 KB 程序存储器的器件（PIC18F25K80 和 PIC18F46K80）上未实现。

21.0 主同步串行口 （MSSP）模块

21.1 主 SSP （MSSP）模块概述

主同步串行口 （MSSP）模块是用于同其他外设或单片机进行通信的串行接口。这些外设可以是串行EEPROM、移位寄存器、显示驱动器和 A/D 转换器等。MSSP 模块有以下两种工作模式：

- 串行外设接口 （Serial Peripheral Interface， SPI）
- I^2C™
  - 全功能主模式
  - 从模式 （支持广播地址呼叫）

I^2C 接口硬件上支持以下模式：

- 主模式
- 多主器件模式
- 带 5 位和 7 位地址掩码的从模式 （具有用于 10 位和 7 位寻址的地址掩码）

21.2 控制寄存器

MSSP 模块有三个相关的控制寄存器，包括一个状态寄存器 （SSPSTAT）和两个控制寄存器 （SSPCON1 和 SSPCON2）。根据 MSSP 模块是在 SPI 模式还是 I^2C 模式下工作，这些寄存器及其各自的配置位的使用将有很大不同。

下面各节会提供更多详细信息。

21.3 SPI 模式

SPI 模式允许同时同步发送和接收 8 位数据。器件支持SPI的所有四种模式。通常使用以下3个引脚来实现通信:

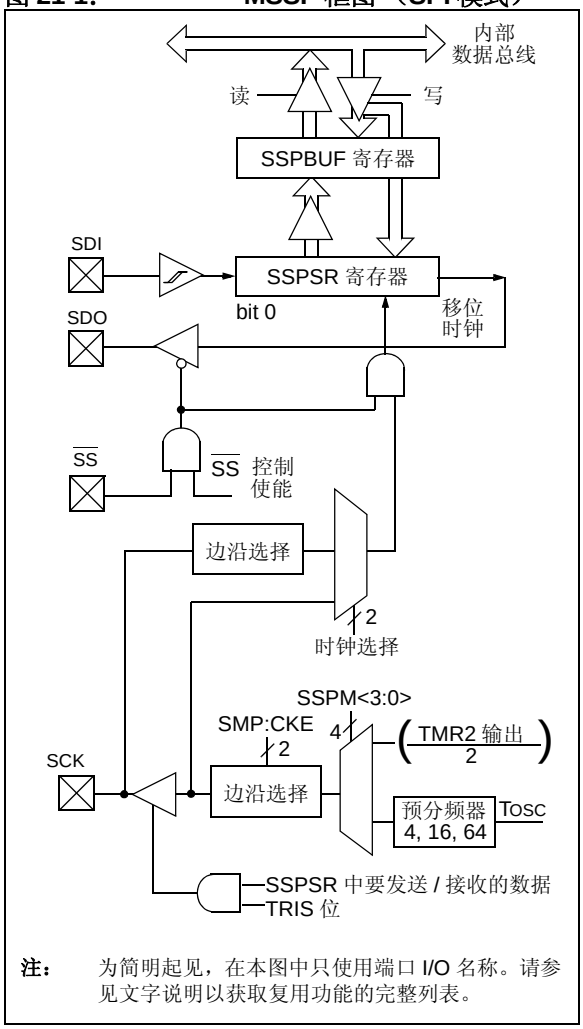
- 串行数据输出 （SDO） ——RC5/SDO
- 串行数据输入 （SDI） ——RC4/SDA/SDI
- 串行时钟 （SCK） ——RC3/REF0/SCL/SCK

此外，当处于从工作模式时要使用第 4 个引脚：

- 从选择 （$\overline{SS}$） ——RA5/AN4/C2INB/HLVDIN/T1CKI/$\overline{SS}$/CTMU1

图21-1给出了MSSP模块在SPI模式下的工作原理框图。

图 21-1： MSSP 框图 （SPI 模式）



注： 为简明起见，在本图中只使用端口 I/O 名称。请参见文字说明以获取复用功能的完整列表。

21.3.1 寄存器

MSSP 模块有四个寄存器用于 SPI 工作模式。这些寄存器包括：

- MSSP 控制寄存器 1 （SSPCON1）
- MSSP 状态寄存器 （SSPSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPBUF）
- MSSP 移位寄存器（SSPSR） —— 不可直接访问

SSPCON1 和 SSPSTAT 是 SPI 工作模式下的控制寄存器和状态寄存器。 SSPCON1 寄存器是可读写的。SSPSTAT 的低 6 位是只读的，而高 2 位是可读写的。

SSPSR是用来将数据移入或移出的移位寄存器。SSPBUF是缓冲寄存器，可用于数据字节的写入或读出。

在接收操作中，SSPSR 和 SSPBUF 共同构成一个双重缓冲接收器。 SSPSR 接收到一个完整的字节之后，该字节会被送入SSPBUF，同时将中断标志位SSPIF置1。

在数据发送过程中， SSPBUF 不是双重缓冲的，对 SSPBUF 的写操作将同时写入 SSPBUF 和 SSPSR。

寄存器 21-1： SSPSTAT：MSSP 状态寄存器 （SPI 模式）

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE(1)	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
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 **SMP**：采样位
SPI 主模式：
1 = 在数据输出时间的末尾采样输入数据
0 = 在数据输出时间的中间采样输入数据
SPI 从模式：
当 SPI 工作在从模式时，必须将 SMP 清零。

bit 6 **CKE**：SPI 时钟选择位 (1)
1 = 时钟状态从有效转换到空闲时发送
0 = 时钟状态从空闲转换到有效时发送

bit 5 **D/$\overline{A}$**：数据 / $\overline{地址}$位
仅在 I^2C™ 模式下使用。

bit 4 **P**：停止位
仅在 I^2C 模式下使用。当禁止 MSSP 模块时，该位被清零； SSPEN 被清零。

bit 3 **S**：启动位
仅在 I^2C 模式下使用。

bit 2 **R/$\overline{W}$**：读 / $\overline{写}$信息位
仅在 I^2C 模式下使用。

bit 1 **UA**：更新地址位
仅在 I^2C 模式下使用。

bit 0 **BF**：缓冲区满状态位 （仅限接收模式）
1 = 接收完成， SSPBUF 已满
0 = 接收未完成， SSPBUF 为空

注 1： 时钟状态的极性由 CKP 位 （SSPCON1<4>）设置。

寄存器 21-2： SSPCON1：MSSP 控制寄存器 1（SPI 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV(1)	SSPEN(2)	CKP	SSPM3(3)	SSPM2(3)	SSPM1(3)	SSPM0(3)
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 **WCOL**：写冲突检测位
1 = 正在发送前一个字时，又有数据写入 SSPBUF 寄存器 （必须用软件清零）
0 = 未发生冲突

bit 6 **SSPOV**：接收溢出指示位 (1)
SPI 从模式：
1 = SSPBUF 寄存器中仍保存前一数据时，又接收到一个新的字节。如果发生溢出， SSPSR 中的数据会丢失。溢出只会在从模式下发生。即使只是发送数据，用户也必须读 SSPBUF，以避免将溢出标志位置 1 （该位必须用软件清零）。
0 = 无溢出

bit 5 **SSPEN**：主同步串口使能位 (2)
1 = 使能串口并将 SCK、 SDO、 SDI 和 $\overline{SS}$ 配置为串口引脚
0 = 禁止串口并将上述引脚配置为 I/O 端口引脚

bit 4 **CKP**：时钟极性选择位
1 = 空闲状态时，时钟为高电平
0 = 空闲状态时，时钟为低电平

bit 3-0 **SSPM<3:0>**：主同步串口模式选择位 (3)
1010 = SPI 主模式：时钟 = Fosc/8
0101 = SPI 从模式：时钟 = SCK 引脚；禁止 $\overline{SS}$ 引脚控制；可将 $\overline{SS}$ 用作 I/O 引脚
0100 = SPI 从模式：时钟 = SCK 引脚；使能 $\overline{SS}$ 引脚控制
0011 = SPI 主模式：时钟 = TMR2 输出 /2
0010 = SPI 主模式：时钟 = Fosc/64
0001 = SPI 主模式：时钟 = Fosc/16
0000 = SPI 主模式：时钟 = Fosc/4

注 1： 在主模式下，溢出位不会被置 1，因为每次接收 （和发送）新数据都是通过写入 SSPBUF 寄存器启动的。
2： 当使能时，必须将这些引脚正确地配置为输入或输出。
3： 在此未列出的位组合被保留或仅在 I^2C 模式下实现。

21.3.2 工作原理

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 （SSPCON1<5:0> 和 SSPSTAT<7:6>）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）
- 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 的上升沿 / 下降沿输出数据）
- 时钟速率 （仅限主模式）
- 从选择模式 （仅限从模式）

MSSP 模块由一个发送 / 接收移位寄存器 （SSPSR）和一个缓冲寄存器（SSPBUF）组成。SSPSR 将数据移入 / 移出器件，先移位 MSb。在新数据接收完毕前，SSPBUF 保存上次写入 SSPSR 的数据。一旦 8 位数据接收完毕，该字节就被移入 SSPBUF 寄存器。然后，缓冲区满检测位 BF （SSPSTAT<0>）和中断标志位 SSPIF 被置 1。这种双重缓冲数据接收方式 （SSPBUF），允许在 CPU 读取刚接收的数据之前，就开始接收下一个字节。在数据发送 / 接收期间，任何对 SSPBUF 寄存器的写操作都将被忽略，并且写冲突检测位 WCOL （SSPCON1<7>）将被置 1。用户必须用软件将 WCOL 位清零才能判断以后对 SSPBUF 寄存器的写入是否成功。

为确保应用软件能接收有效数据，在下一个要发送的数据字节写入 SSPBUF 之前，读取 SSPBUF 中现有的数据。缓冲区满位 BF （SSPSTAT<0>）用于表示何时 SSPBUF 装入了接收到的数据（发送完成）。SSPBUF 中的数据被读取后， BF 位即被清零。如果 SPI 仅作为一个发送器，则不必理会该数据。通常，可用 MSSP 中断来判断发送 / 接收是否已完成。如果不打算使用中断，用软件查询的方法同样可确保不会发生写冲突。例 21-1 说明了如何为 SSPBUF （SSPSR）装入数据，以进行数据发送。

不能直接读写 SSPSR 寄存器，只能通过寻址 SSPBUF 寄存器来访问。此外， SSPSTAT 寄存器用于指示各种状态条件。

21.3.3 漏极开路输出选项

用于 SDO 输出和 SCK 时钟引脚的驱动器可以有选择地配置为漏极开路输出。此功能使得可以通过外部上拉电阻将引脚上的电平上拉至较高电平，这使输出无需额外的电平转换器就可以与外部电路进行通信。更多信息，请参见**第 11.1.3 节 “ 漏极开路输出 ”**。

漏极开路输出选项由 SSPOD 位（ODCON<7>）控制。SSPOD 位置 1 时，可以将 SDO 和 SCK 引脚配置为相应的漏极开路操作。

例 21-1： 装载 SSPBUF （SSPSR）寄存器

```
LOOP    BTFSS   SSPSTAT, BF     ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSPBUF, W       ;WREG reg = contents of SSPBUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSPBUF          ;New data to xmit
```

21.3.4 使能 SPI I/O

要使能串口，MSSP 使能位 SSPEN（SSPCON1<5>）必须置 1。要复位或重新配置 SPI 模式，先将 SSPEN 位清零，重新初始化 SSPCON 寄存器，然后再将 SSPEN 位置 1。这会将 SDI、SDO、SCK 和 $\overline{SS}$ 引脚配置为串口引脚。要将引脚用作串口功能，必须正确设置其中一些引脚的数据方向位（在 TRIS 寄存器中）：

- SDI 由 SPI 模块自动控制
- SDO 必须将 TRISC<5> 位清零
- SCK（主模式）必须将 TRISC<3> 位清零
- SCK（从模式）必须将 TRISC<3> 位置 1
- $\overline{SS}$ 必须将 TRISA<5> 位置 1

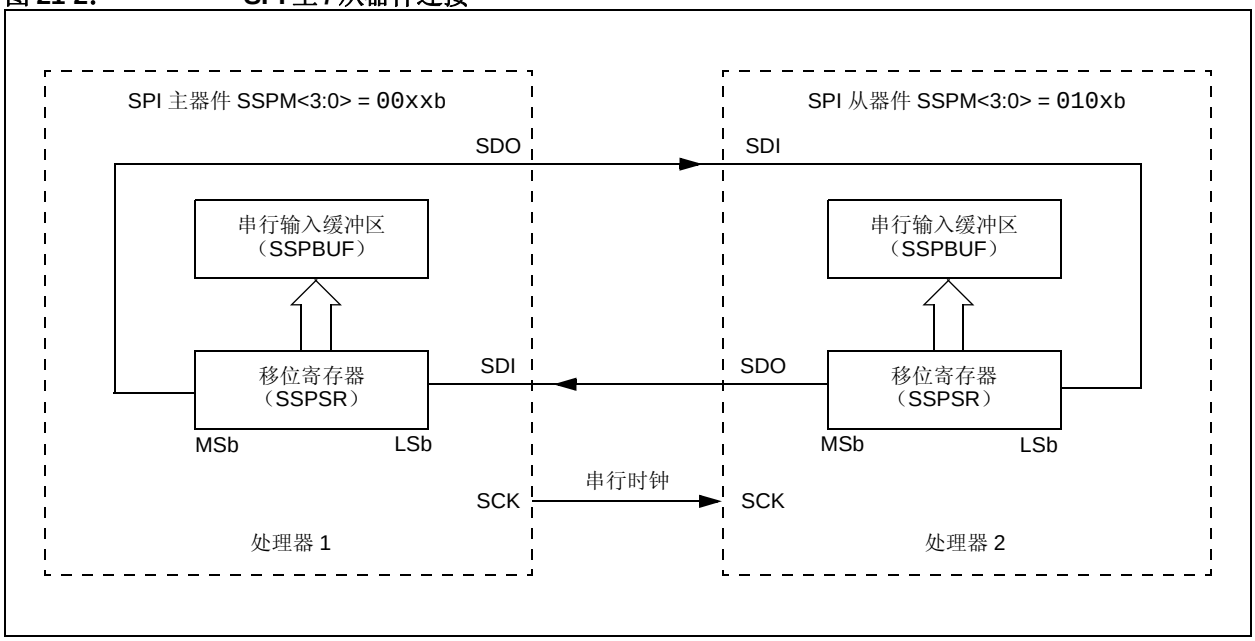
对于不需要的串口功能，可通过将对应的数据方向寄存器（TRIS）设置为相反值来改写。

21.3.5 典型连接

图 21-2 给出了两个单片机之间的典型连接。主控制器（处理器 1）通过发送 SCK 信号来启动数据传输。数据在编程设定的时钟边沿从两个移位寄存器移出，并在相反的时钟边沿锁存。必须将两个处理器的时钟极性（CKP）设置为相同，这样就可以同时收发数据。数据是否有意义（或无效数据），取决于应用软件。这就导致以下三种数据传输情形：

- 主器件发送数据 —— 从器件发送无效（Dummy）数据
- 主器件发送数据 —— 从器件发送数据
- 主器件发送无效数据 —— 从器件发送数据

图 21-2： SPI 主 / 从器件连接



21.3.6 主模式

因为由主器件控制 SCK 信号，所以它可以在任意时刻启动数据传输。主器件根据软件协议确定从器件（图 21-2 中的处理器 1）在何时广播数据。

在主模式下，一写入 SSPBUF 寄存器就发送或接收数据。如果只打算将 SPI 作为接收器，则可以禁止 SDO 输出 （将其编程设置为输入）。 SSPSR 寄存器按设置的时钟速率，连续移入 SDI 引脚上的信号。每接收到一个字节，就将其装入 SSPBUF 寄存器，就像接收到普通字节一样 （中断和状态位相应置 1）。这在以 “ 线路活动监控 ”（Line Activity Monitor）方式工作的接收器应用中很有用。

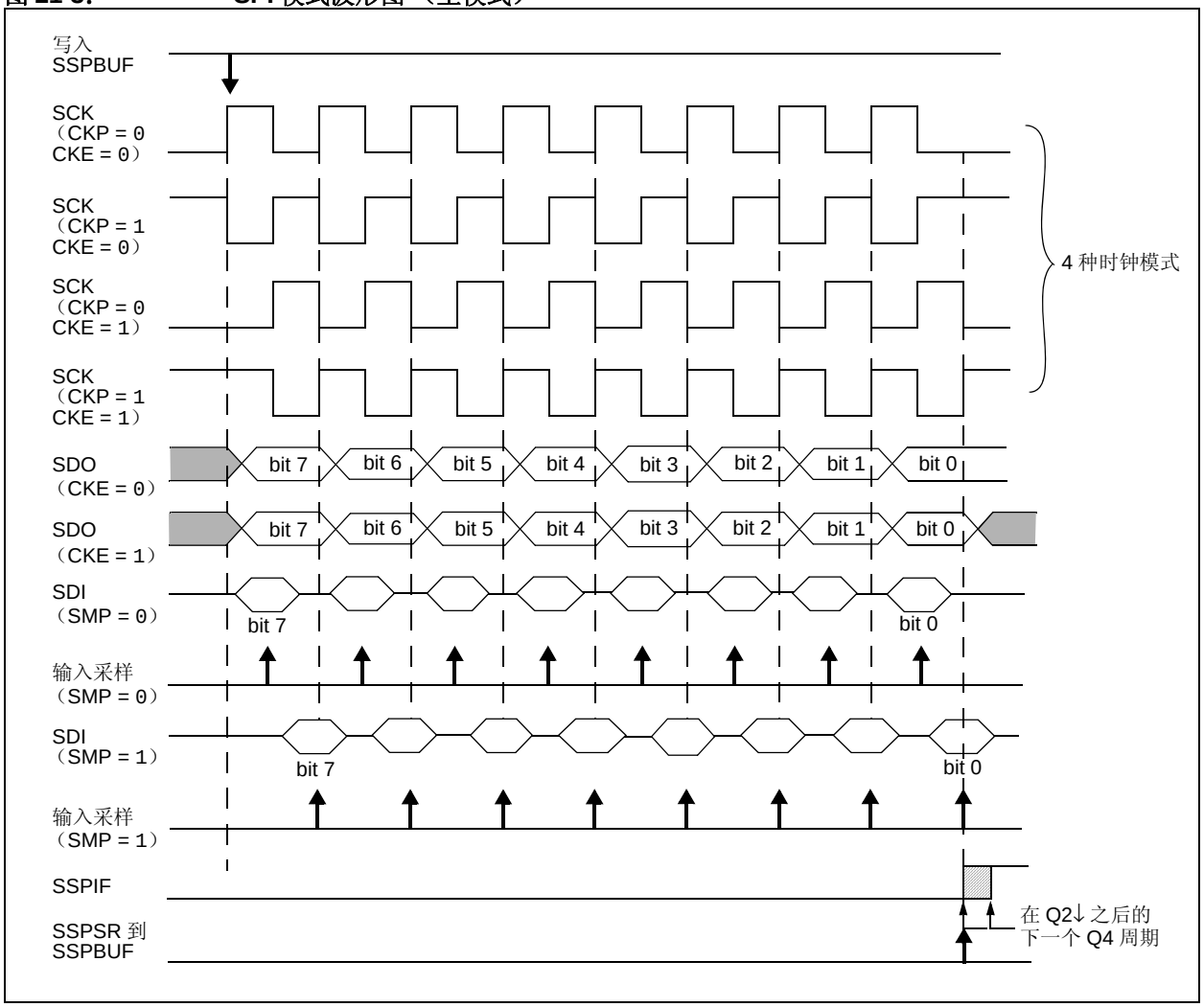
可通过对 CKP 位（SSPCON1<4>）进行适当的编程来选择时钟极性。图 21-3、图 21-5 和图 21-6 给出了 SPI 通信的波形图，其中 MSB 先发送。在主模式下，SPI 时钟速率 （比特率）可由用户编程设定为以下几种之一：

- Fosc/4 （或 Tcy）
- Fosc/16 （或 4 • Tcy）
- Fosc/64 （或 16 • Tcy）
- Timer2 输出 /2

这样可使数据速率最高达到 16 Mbps （时钟频率为 64 MHz）。

图 21-3 给出了主模式的波形图。当 CKE 位置 1 时，SDO 数据在 SCK 上出现时钟边沿前一直有效。图中所示的输入采样的变化由 SMP 位的状态反映。图中给出了将接收到的数据装入 SSPBUF 的时刻。

图 21-3： SPI 模式波形图 （主模式）



21.3.7 从模式

在从模式下，当 SCK 引脚上出现外部时钟脉冲时发送和接收数据。最后一位数据被锁存后，中断标志位 SSPIF 置 1。

在从模式下，外部时钟由 SCK 引脚上的外部时钟源提供。外部时钟必须满足电气规范中规定的高电平和低电平的最短时间要求。

在休眠模式下，从器件仍可发送 / 接收数据。当接收到一个字节时，器件可配置为从休眠状态唤醒。

21.3.8 从选择同步

$\overline{SS}$ 引脚允许器件工作于同步从模式。SPI 必须处于从模式，并使能 $\overline{SS}$ 引脚控制 （SSPCON1<3:0> = 04h）。当 $\overline{SS}$ 引脚为低电平时，使能数据的发送和接收，同时 SDO 引脚被驱动。当 $\overline{SS}$ 引脚变为高电平时，即使是在字节的发送过程中，也不再驱动 SDO 引脚，而是变成悬空输出状态。可根据应用需要，在 SDOx 引脚上外接上拉 / 下拉电阻。

注	**1：** 当 SPI 处于从模式且 $\overline{SS}$ 引脚控制使能（SSPCON1<3:0> = 0100）时，如果 $\overline{SS}$ 引脚设置为 V_{DD}， SPI 模块将会复位。 **2：** 如果 SPI 工作在从模式下并且 CKE 置 1，则必须使能 $\overline{SS}$ 引脚控制。

SPI 模块复位后，位计数器被强制为 0。这是通过强制将 $\overline{SS}$ 引脚拉为高电平或将 SSPEN 位清零来实现的。

可将 SDO 引脚和 SDI 引脚相连来仿真二线制通信。当 SPI 需要作为接收器工作时， SDO 引脚可被配置为输入端。这样就禁止了从 SDO 发送数据。因为 SDI 不会引起总线冲突，所以可以一直将其保留为输入（SDI 功能）。

图 21-4： 从同步波形图

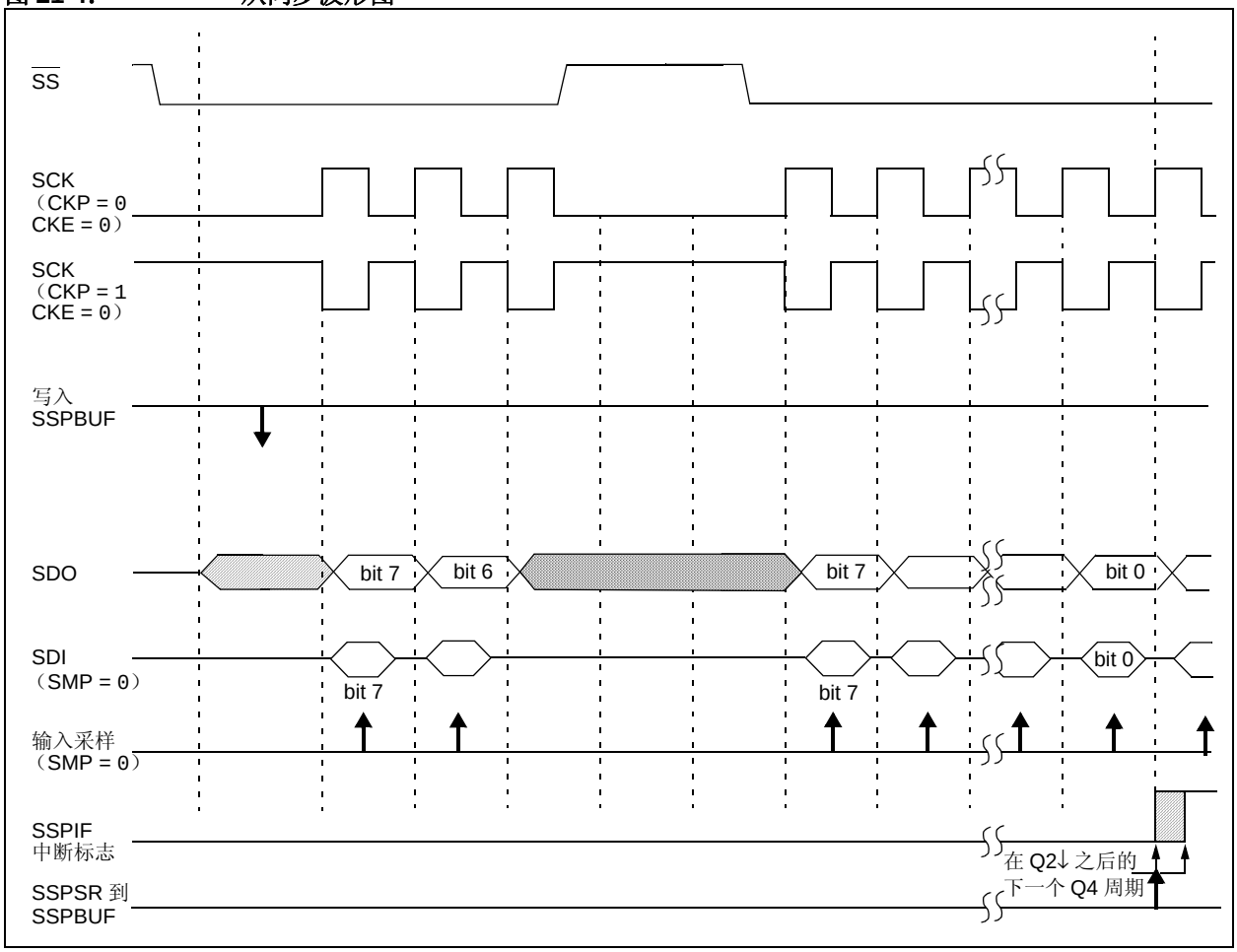


图 21-5：　　SPI 模式波形图 （从模式， CKE = 0）

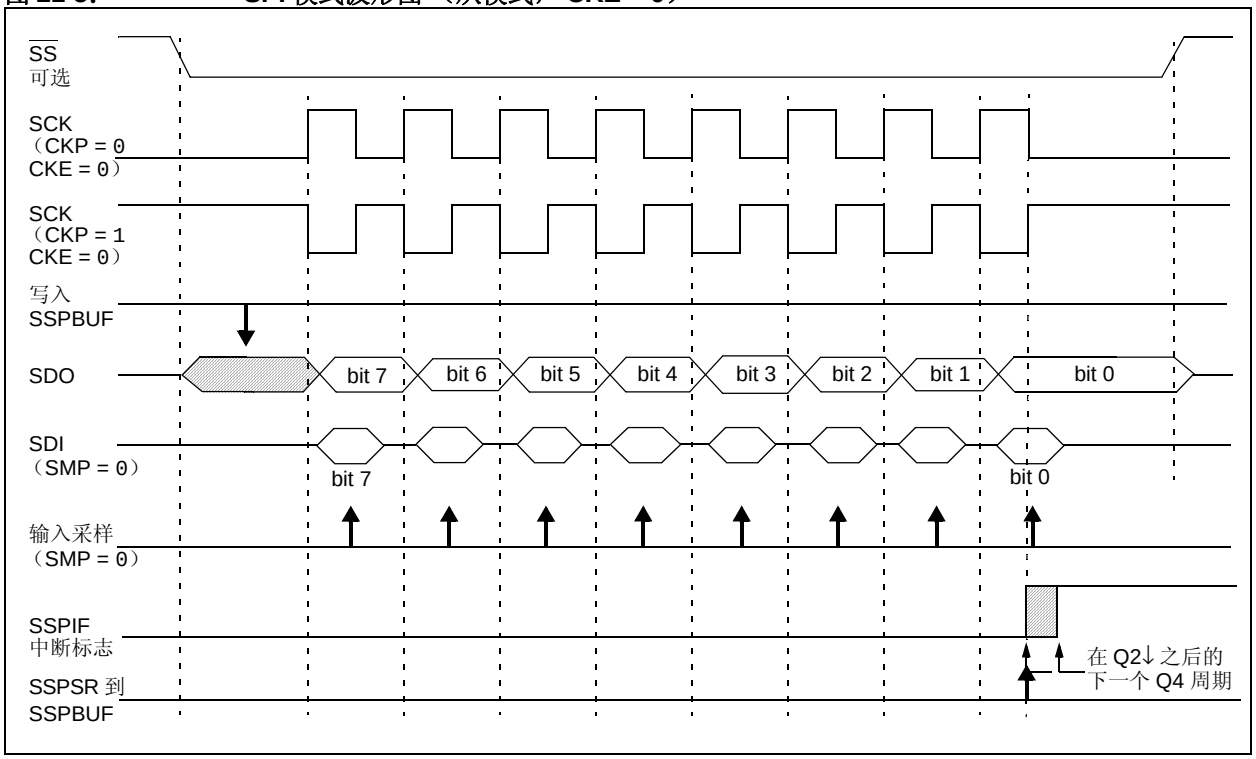
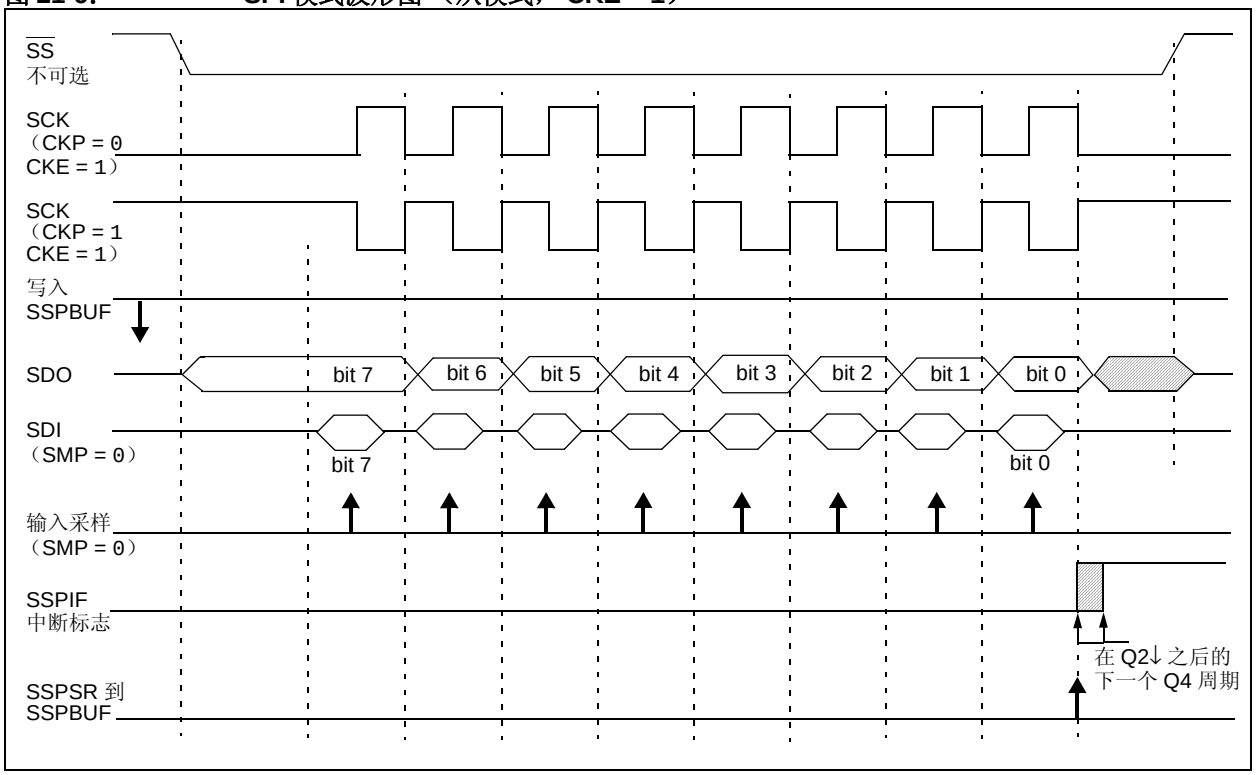


图 21-6：　　SPI 模式波形图 （从模式， CKE = 1）



21.3.9 在功耗管理模式下的操作

在 SPI 主模式下，模块时钟速度与全功耗模式下的不同；处于休眠模式时，所有时钟都停止。

在空闲模式下，需要为外设提供一个时钟。该时钟可能来自主时钟源、辅助时钟源（SOSC振荡器）或INTOSC时钟源。更多信息，请参见**第 3.3 节 “ 时钟源与振荡器切换 ”**。

在大多数情况下，主器件为 SPI 数据提供的时钟速度并不重要；但是，每个系统都应该评估此因素。

如果允许了 MSSP 中断，那么当主器件发送完数据时，这些中断可以将控制器从休眠模式或某种空闲模式唤醒。如果不想从休眠或空闲模式退出，应该禁止 MSSP 中断。

如果选择了休眠模式，所有模块的时钟都将停止，并且在器件被唤醒前，发送 / 接收将保持此停止状态。器件返回到运行模式后，MSSP 模块将重新开始发送和接收数据。

在 SPI 从模式下， SPI 发送 / 接收移位寄存器与器件异步工作。这可使器件处于任何功耗管理模式下，而且数据仍可被移入 SPI 发送 / 接收移位寄存器。 8 位数据全部接收到后， MSSP 中断标志位将置 1，并且如果允许中断的话，器件被唤醒。

21.3.10 复位的影响

复位会禁止 MSSP 模块并终止当前的数据传输。

21.3.11 总线模式兼容性

表 21-1 显示了标准 SPI 模式与 CKP 和 CKE 控制位状态之间的兼容性。

表 21-1： SPI 总线模式

标准 SPI 模式术语	控制位状态	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

还有一个 SMP 位用来控制数据何时被采样。

表 21-2： 与 SPI 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
TRISA	TRISA7	TRISA6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
SSPBUF	MSSP 接收缓冲 / 发送寄存器							
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD

图注： SPI 模式下的 MSSP 模块不使用阴影单元。

21.4 I²C 模式

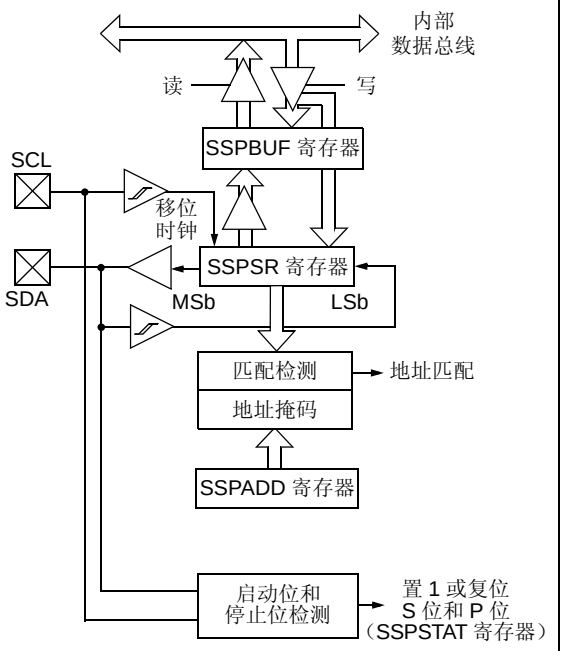
MSSP 模块工作在 I²C 模式时，可以实现所有的主和从功能（包括广播呼叫支持），并且硬件上提供启动位和停止位的中断来判断总线何时空闲（多主器件功能）。MSSP 模块实现了标准模式规范以及 7 位和 10 位寻址。

有两个引脚用于数据传输：

- 串行时钟（Serial Clock，SCL）——RC3/REFO/SCL/SCK
- 串行数据（Serial Data，SDA）——RC4/SDA/SDI

用户必须通过设置相关的 TRIS 位将这些引脚配置为输入引脚。

图 21-7：　　MSSP 框图（I²C™ 模式）



注： 为简明起见，在本图中只使用端口 I/O 名称。请参见文字说明以获取复用功能的完整列表。

21.4.1 寄存器

MSSP模块有7个寄存器用于I²C操作。这些寄存器包括：

- MSSP 控制寄存器 1（SSPCON1）
- MSSP 控制寄存器 2（SSPCON2）
- MSSP 状态寄存器（SSPSTAT）
- 串行接收 / 发送缓冲寄存器（SSPBUF）
- MSSP 移位寄存器（SSPSR）—— 不可直接访问
- MSSP 地址寄存器（SSPADD）
- I²C 从地址掩码寄存器（SSPMSK）

SSPCON1、SSPCON2 和 SSPSTAT 是在 I²C 工作模式下的控制寄存器和状态寄存器。SSPCON1 和 SSPCON2 寄存器是可读写的。SSPSTAT 的低 6 位是只读的，而高 2 位是可读写的。

SSPSR 是用来将数据移入或移出的移位寄存器。SSPBUF 是缓冲寄存器，可用于数据字节的写入或读出。

当 MSSP 被配置为工作在 I²C 从模式下时，SSPADD 保存从器件的地址。当 MSSP 被配置为工作在主模式下时，SSPADD 的低 7 位用作波特率发生器的重载值。

当模块配置为 7 位地址掩码模式时，SSPMSK 保存从地址掩码值。虽然它是独立的寄存器，它与 SSPADD 共用相同的 SFR 地址；只有 SSPM<3:0> 位特别设置为允许访问时，才能访问它。更多详细信息，请参见**第 21.4.3.4 节 “7 位地址掩码模式 ”**。

在接收操作中，SSPSR 和 SSPBUF 共同构成一个双重缓冲接收器。SSPSR 接收到一个完整的字节之后，该字节会被送入 SSPBUF，同时将中断标志位 SSPIF 置 1。

在数据发送过程中，SSPBUF 不是双重缓冲的，对 SSPBUF 的写操作将同时写入 SSPBUF 和 SSPSR。

寄存器 21-3： SSPSTAT：MSSP 状态寄存器 （I²C™ 模式）

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P(1)	S(1)	R/$\overline{W}$(2,3)	UA	BF
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 **SMP**：压摆率控制位
在主模式或从模式下：
1 = 标准速度模式下禁止压摆率控制 （100 kHz 和 1 MHz）
0 = 高速模式下使能压摆率控制 （400 kHz）

bit 6 **CKE**：SMBus 选择位
在主模式或从模式下：
1 = 使能 SMBus 特定输入
0 = 禁止 SMBus 特定输入

bit 5 **D/$\overline{A}$**：数据 / $\overline{地址}$位
在主模式下：
保留。
在从模式下：
1 = 指示上一个接收或发送的字节是数据
0 = 指示上一个接收或发送的字节是地址

bit 4 **P**：停止位 (1)
1 = 指示上次检测到停止位
0 = 上次未检测到停止位

bit 3 **S**：启动位 (1)
1 = 指示上次检测到启动位
0 = 上次未检测到启动位

bit 2 **R/$\overline{W}$**：读 / $\overline{写}$信息位 (2,3)
在从模式下：
1 = 读
0 = 写
在主模式下：
1 = 正在进行发送
0 = 不在进行发送

bit 1 **UA**：更新地址位 （仅限 10 位从模式）
1 = 指示用户需要更新 SSPADD 寄存器中的地址
0 = 不需要更新地址

bit 0 **BF**：缓冲区满状态位
在发送模式下：
1 = SSPBUF 已满
0 = SSPBUF 为空
在接收模式下：
1 = SSPBUF 已满 （不包括 $\overline{ACK}$ 位和停止位）
0 = SSPBUF 为空 （不包括 ACK 位和停止位）

注 1： 该位在复位及 SSPEN 清零时被清零。
2： 该位保存上一次地址匹配后的 R/$\overline{W}$ 位信息。该位仅在从地址匹配到出现下一个启动位、停止位或非 $\overline{ACK}$ 位之间有效。
3： 将该位与 SEN、 RSEN、 PEN、 RCEN 或 ACKEN 进行逻辑或运算将指示 MSSP 是否处于有效模式。

寄存器 21-4： SSPCON1：MSSP 控制寄存器 1 （I²C™ 模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN(1)	CKP	SSPM3(2)	SSPM2(2)	SSPM1(2)	SSPM0(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 **WCOL**：写冲突检测位
在主发送模式下：
1 = 当 I²C 不满足启动发送数据的条件时，试图向 SSPBUF 寄存器写入数据 （必须用软件清零）
0 = 未发生冲突
在从发送模式下：
1 = 正在发送前一个字时，又有数据写入 SSPBUF 寄存器 （必须用软件清零）
0 = 未发生冲突
在接收模式 （主模式或从模式）下：
该位是无关位。

bit 6 **SSPOV**：接收溢出指示位
在接收模式下：
1 = SSPBUF 寄存器仍保存前一字节时，接收到一个新的字节 （必须用软件清零）
0 = 无溢出
在发送模式下：
在发送模式下，该位是无关位。

bit 5 **SSPEN**：主同步串口使能位 (1)
1 = 使能串口并将 SDA 和 SCL 引脚配置为串口引脚
0 = 禁止串口并将上述引脚配置为 I/O 端口引脚

bit 4 **CKP**：SCK 释放控制位
在从模式下：
1 = 释放时钟
0 = 保持时钟低电平 （时钟延长），用来确保数据建立时间
在主模式下：
在此模式下未使用。

bit 3-0 **SSPM<3:0>**：主同步串口模式选择位 (2)
1111 = I²C 从模式， 10 位地址，并允许启动位和停止位中断
1110 = I²C 从模式， 7 位地址，并允许启动位和停止位中断
1011 = I²C 固件控制的主模式 （从器件空闲）
1001 = 装入 SSPADD SFR 地址处的 SSPMSK 寄存器 (3,4)
1000 = I²C 主模式，时钟 = FOSC/(4 * (SSPADD + 1))
0111 = I²C 从模式， 10 位地址
0110 = I²C 从模式， 7 位地址

注 1： 当使能时，必须将 SDA 和 SCL 引脚配置为输入引脚。
2： 此处未列出的位组合被保留或仅在 SPI 模式下实现。
3： 当 SSPM<3:0> = 1001 时，任何对 SSPADD SFR 地址的读或写实际上访问 SSPMSK 寄存器。
4： 该模式仅在选择了 7 位地址掩码模式 （MSSPMSK 配置位为 1）时可用。

寄存器 21-5： SSPCON2：MSSP 控制寄存器 2 （I²C™ 主模式）

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT(1)	ACKEN(2)	RCEN(2)	PEN(2)	RSEN(2)	SEN(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 **GCEN：**广播呼叫使能位
在主模式下未使用。

bit 6 **ACKSTAT：**应答状态位 （仅限主发送模式）
1 = 未收到来自从器件的应答
0 = 收到来自从器件的应答

bit 5 **ACKDT：**应答数据位 （仅限主接收模式） **(1)**
1 = 无应答
0 = 应答

bit 4 **ACKEN：**应答序列使能位 **(2)**
1 = 在 SDA 和 SCL 引脚上发出应答序列，并发送 ACKDT 数据位。由硬件自动清零。
0 = 应答序列空闲

bit 3 **RCEN：**接收使能位 （仅限主接收模式） **(2)**
1 = 使能 I²C™ 接收模式
0 = 接收空闲

bit 2 **PEN：**停止条件使能位 **(2)**
1 = 在 SDA 和 SCL 引脚上发出停止条件。由硬件自动清零。
0 = 停止条件空闲

bit 1 **RSEN：**重复启动条件使能位 **(2)**
1 = 在 SDA 和 SCL 引脚上发出重复启动条件。由硬件自动清零。
0 = 重复启动条件空闲

bit 0 **SEN：**启动条件使能位 **(2)**
1 = 在 SDA 和 SCL 引脚上发出启动条件。由硬件自动清零。
0 = 启动条件空闲

注 1： 当用户在接收结束时发出一个应答序列时，发送该值。
2： 如果 I²C 模块处于工作状态，可能这些位不会被置 1 （无并行工作），并且可能不会写入 SSPBUF （或禁止写 SSPBUF）。

寄存器 21-6： SSPCON2：MSSP 控制寄存器 2 （I²C™ 从模式）

R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT(1)	ACKEN(1)	RCEN(1)	PEN(1)	RSEN(1)	SEN(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 **GCEN：**广播呼叫使能位
1 = 当 SSPSR 中接收到广播呼叫地址 （0000h）时允许中断
0 = 禁止广播呼叫地址

bit 6 **ACKSTAT：**应答状态位
在从模式下未使用。

bit 5 **ACKDT：**应答数据位 （仅限主接收模式）(1)
1 = 无应答
0 = 应答

bit 4 **ACKEN：**应答序列使能位 (1)
1 = 在 SDA 和 SCL 引脚上发出应答序列，并发送 ACKDT 数据位。由硬件自动清零。
0 = 应答序列空闲

bit 3 **RCEN：**接收使能位 （仅限主接收模式）(1)
1 = 使能 I²C™ 接收模式
0 = 接收空闲

bit 2 **PEN：**停止条件使能位 (1)
1 = 在 SDA 和 SCL 引脚上发出停止条件。由硬件自动清零。
0 = 停止条件空闲

bit 1 **RSEN：**重复启动条件使能位 (1)
1 = 在 SDA 和 SCL 引脚上发出重复启动条件。由硬件自动清零。
0 = 重复启动条件空闲

bit 0 **SEN：**时钟延长使能位 (1)
1 = 为从发送和从接收 （已使能时钟延长）使能时钟延长
0 = 禁止时钟延长

注 1： 如果 I²C 模块处于工作状态，可能该位不会被置 1 （无并行工作），并且可能不会写入 SSPBUF （或禁止写 SSPBUF）。

寄存器 21-7： SSPMSK：I²C™ 从地址掩码寄存器 （7 位掩码模式）(1)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0(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 7-0 **MSK<7:0>：**从地址掩码选择位
1 = 使能 SSPADD 对应位的掩码
0 = 禁止 SSPADD 对应位的掩码

注 1： 该寄存器与 SSPADD 共用相同的 SFR 地址，并且仅在某些 MSSP 工作模式下才可寻址该寄存器。更多详细信息，请参见**第 21.4.3.4 节 “7 位地址掩码模式 ”**。
2： 在 7 位寻址时， MSK0 不用作掩码位。

21.4.2 工作原理

MSSP 模块功能通过将 MSSP 使能位 SSPEN（SSPCON1<5>）置 1 使能。

通过 SSPCON1 寄存器可以控制 I^2C 工作模式。可通过设置 4 个模式选择位（SSPCON1<3:0>）选择以下 I^2C 模式之一：

- I^2C 主模式，时钟
- I^2C 从模式（7 位地址）
- I^2C 从模式（10 位地址）
- I^2C 从模式（7 位地址），允许启动位和停止位中断
- I^2C 从模式（10 位地址），允许启动位和停止位中断
- I^2C 固件控制的主模式，从器件空闲

如果通过将相应的 TRISC 位置 1，将 SCL 和 SDA 引脚编程为输入引脚，则在 SSPEN 位置 1 时选择任何 I^2C 模式，将强制上述引脚为漏极开路。要确保此模块正常工作，必须为 SCL 和 SDA 引脚外接上拉电阻。

21.4.3 从模式

在从模式下，SCL 和 SDA 引脚必须被配置为输入（TRISC<4:3> 置 1）。必要时 MSSP 模块将用输出数据改写输入状态（从发送器）。

I^2C 从模式硬件总是在地址匹配时产生中断。地址掩码功能可使硬件在多个地址发生匹配时（7 位寻址模式下最多 31 个，10 位寻址模式下最多 63 个）产生中断。用户也可以通过模式选择位，选择使用启动位或停止位中断。

当地址匹配或在地址匹配后发送的数据被接收时，硬件会自动产生一个应答（$\overline{\text{ACK}}$）脉冲，并将当前 SSPSR 寄存器中接收到的值装入 SSPBUF 寄存器。

只要满足以下条件之一，MSSP 模块就不会产生此 $\overline{\text{ACK}}$ 脉冲：

- 在接收到数据前，缓冲区满位 BF（SSPSTAT<0>）被置 1。
- 在接收到数据前，溢出位 SSPOV（SSPCON1<6>）被置 1。

在上述情况下，SSPSR 寄存器的值不会装入 SSPBUF，但 SSPIF 位会置 1。BF 位是通过读取 SSPBUF 寄存器清零的，而 SSPOV 位是通过软件清零的。

为确保正常工作，SCL 时钟输入必须满足高电平和低电平的最短时间要求。在时序参数 100 和参数 101 中给出了 I^2C 规范的高低电平时间和对 MSSP 模块的具体要求。

21.4.3.1 寻址

一旦 MSSP 模块被使能，它就会等待启动条件出现。启动条件出现后，8 位数据被移入 SSPSR 寄存器。在时钟（SCL）线的上升沿采样所有的输入位。寄存器 SSPSR<7:1> 的值会与 SSPADD 寄存器的值进行比较，该比较是在第 8 个时钟（SCL）脉冲的下降沿进行的。如果地址匹配，并且 BF 位和 SSPOV 位都被清零，会发生以下事件：

1. SSPSR 寄存器的值被装入 SSPBUF 寄存器。
2. 缓冲区满标志位 BF 被置 1。
3. 产生 $\overline{\text{ACK}}$ 脉冲。
4. 在第 9 个 SCL 脉冲的下降沿，MSSP 中断标志位 SSPIF 被置 1（如果允许中断，则产生中断）。

在 10 位寻址模式下，从器件需要接收两个地址字节。第一个地址字节的高 5 位（MSb）将指定这是否是一个 10 位地址。R/$\overline{\text{W}}$ 位（SSPSTAT<2>）必须指定写操作，这样从器件才能接收到第二个地址字节。对于 10 位地址，第一个字节应该是 “11110 A9 A8 0”，其中 “A9” 和 “A8” 是该地址的高 2 位。10 位寻址模式的操作步骤如下，其中 7-9 步是针对从发送器而言的。

1. 接收地址的第一个（高）字节（SSPIF、BF 和 UA 位在地址匹配时被置 1）。
2. 用地址的第二个（低）字节更新 SSPADD 寄存器（清零 UA 位并释放 SCL 线）。
3. 读 SSPBUF 寄存器（清零 BF 位）并清零标志位 SSPIF。
4. 接收地址的第二个（低）字节（SSPIF、BF 和 UA 位置 1）。
5. 用地址的第一个（高）字节更新 SSPADD 寄存器。如果匹配的话就释放SCL线，这将清零UA位。
6. 读 SSPBUF 寄存器（清零 BF 位）并清零标志位 SSPIF。
7. 接收重复启动条件。
8. 接收地址的第一个（高）字节（SSPIF 位和 BF 位置 1）。
9. 读 SSPBUF 寄存器（清零 BF 位）并清零标志位 SSPIF。

21.4.3.2 地址掩码模式

将地址的某一位掩码使该位变为无关位，此时会响应两个地址并产生一个中断。可以一次掩码多个地址位，这可以极大地扩展被应答的地址的数量。

不管是否使用地址掩码，I^2C 从器件的工作方式保持不变。但当使用地址掩码时，I^2C 从器件能够应答多个地址并产生中断，此时需要通过检查 SSPBUF 来判断是哪个地址引起中断。

在 I^2C 从操作中，PIC18F66K80 系列器件能够使用两种不同的地址掩码模式：5 位地址掩码和 7 位地址掩码。掩码模式在器件配置中使用 MSSPMSK 配置位选择。默认的器件配置是 7 位地址掩码。

两种掩码模式都支持 7 位和 10 位地址的地址掩码。掩码模式和地址的每种组合均能提供不同的可应答地址范围。

虽然两种掩码模式的工作方式大致相同，它们使用地址掩码的方式是不同的。

21.4.3.3 5 位地址掩码模式

正如名称所指，5 位地址掩码模式最多可掩码 5 个地址位，用传入地址的 bit 5 至 bit 1 来产生可被应答的地址范围。使用 7 位寻址时，这使模块最多可以应答 31 个地址；使用 10 位寻址时，这使模块最多可以应答 63 个地址（见例 21-2）。当编程 MSSPMSK 配置位（0）时，即选择该掩码模式。

该地址模式下的地址掩码存储在 SSPCON2 寄存器中，在 I^2C 从模式下，该寄存器停止用作控制寄存器（寄存器 21-6）。在 7 位寻址模式下，地址掩码位 ADMSK<5:1>（SSPCON2<5:1>）可用来掩码 SSPADD 寄存器中对应的地址位。如果 ADMSK 的某位被置 1（ADMSK<n> = 1），则对应的地址位被忽略（SSPADD<n> = x）。对于发出地址应答的模块，只要与没被掩码的地址位匹配就可以了。

在 10 位寻址模式下，ADMSK<5:2> 位可用来掩码 SSPADD 寄存器中对应的地址位，而 ADMSK1 可以同时掩码地址的低 2 位（SSPADD<1:0>）。如果 ADMSK 的某位是有效的（ADMSK<n> = 1），则对应的地址位被忽略（SPxADD<n> = x）。另外需要注意的是，尽管在 10 位寻址模式下，地址的高位也要用到 SSPADD 寄存器中的某些位。地址掩码位对这些高位不起作用；地址掩码位只会影响地址的低位。

注	**1：** ADMSK1 掩码地址的低 2 位。
	2： 地址的高 2 位不受地址掩码的影响。

例 21-2： 5 位掩码模式下的地址掩码示例

7 位寻址：

SSPADD<7:1>= A0h （1010000）（SSPADD<0> 假设为 0）

ADMSK<5:1> = 00111

可被应答的地址：A0h, A2h, A4h, A6h, A8h, AAh, ACh, AEh

10 位寻址：

SSPADD<7:0> = A0h （10100000）（本例中地址高 2 位被忽略，因为它们不受掩码影响）

ADMSK<5:1> = 00111

可被应答的地址：A0h, A1h, A2h, A3h, A4h, A5h, A6h, A7h, A8h, A9h, AAh, ABh, ACh, ADh, AEh, AFh

21.4.3.4 7 位地址掩码模式

不同于 5 位地址掩码模式，7 位地址掩码模式最多可掩码 7 个地址位（10 位寻址时），用传入地址的最低位来定义可被应答的地址范围。使用 7 位寻址时，这使模块最多可以应答 127 个不同地址；使用 10 位寻址时，这使模块最多可以应答 255 个地址（见例 21-3）。该模式是模块的默认配置，当 MSSPMSK 未编程时（1），即选择该模式。

7 位地址掩码模式的地址掩码存储在 SSPMSK 寄存器中，而不是 SSPCON2 寄存器中。SSPMSK 是模块中一个独立的硬件寄存器，但它不能被直接访问。它与 SSPADD 寄存器共用 SFR 空间中的一个地址。要访问 SSPMSK 寄存器，必须选择 MSSP 模式 1001（SSPCON1<3:0> = 1001），然后读取或写入 SSPADD 单元。

要使用 7 位地址掩码模式，必须先使用一个值初始化 SSPMSK，然后再选择 I^2C 从寻址模式。所以，必需的事件序列为：

1. 选择 SSPMSK 访问模式（SSPCON2<3:0> = 1001）。
2. 将掩码值写入相应的 SSPADD 寄存器地址（FC8h）。
3. 设置相应的 I^2C 从模式（对于 10 位寻址，SSPCON2<3:0> = 0111；对于 7 位寻址为 0110）。

置 1 或清零 SSPMSK 中掩码位的作用与在 5 位地址掩码模式下置 1 或清零 ADMSK 位的作用相反。即，清零 SSPMSK 中的某位会导致相应的地址位被掩码；置 1 该位则要求相应的位单元匹配。任何复位后，SSPMSK 都会复位到全 1 状态，因此，在写入掩码值之前对标准 MSSP 操作没有影响。

7 位寻址时，SSPMSK<7:1> 位可用来掩码 SSPADD 寄存器中对应的地址位。如果 SSPMSK 的某位是有效的（SSPMSK<n> = 0），则对应的 SSPADD 地址位被忽略（SSPADD<n> = x）。对于发出地址应答的模块，只要与没被掩码的地址位匹配就可以了。

10 位寻址时，SSPMSK<7:0> 位可用来掩码 SSPADD 寄存器中对应的地址位。如果 SSPMSK 的某位是有效的（= 0），则对应的 SSPADD 地址位被忽略（SSPADD<n> = x）。

注： 地址的高 2 位不受地址掩码的影响。

例 21-3： 7 位掩码模式下的地址掩码示例

7 位寻址：

SSPADD<7:1> = 1010 000

SSPMSK<7:1> = 1111 001

可被应答的地址 = ACh, A8h, A4h, A0h

10 位寻址：

SSPADD<7:0> = 1010 0000（本例中地址高 2 位被忽略，因为它们不受影响）

SSPMSK<5:1> = 1111 0011

可被应答的地址 = ACh, A8h, A4h, A0h

21.4.3.5 接收

当地址字节的R/$\overline{W}$位清零并发生地址匹配时，SSPSTAT 寄存器的 R/$\overline{W}$ 位清零。接收的地址被装入 SSPBUF 寄存器，且 SDA 线保持低电平 （$\overline{ACK}$）。

当发生地址字节溢出时，则不会产生应答 （$\overline{ACK}$）脉冲。溢出条件定义为 BF 位 （SSPSTAT<0>）被置 1，或 SSPOV 位 （SSPCON1<6>）被置 1。

每传输一个数据字节都会产生一个 MSSP 中断。中断标志位 SSPIF 必须用软件清零。通过 SSPSTAT 寄存器可以确定该字节的状态。

如果 SEN 被使能 （SSPCON2<0> = 1）， SCL 将在每次数据传输后保持低电平 （时钟延长） 。必须通过将 CKP 位 （SSPCON1<4>）置 1 来释放时钟。更多详细信息，请参见**第 21.4.4 节 “ 时钟延长 ”**。

21.4.3.6 发送

当输入的地址字节的 R/$\overline{W}$ 位置 1 并发生地址匹配时，SSPSTAT 寄存器的 R/$\overline{W}$ 位置 1。接收到的地址被装入 SSPBUF 寄存器。 $\overline{ACK}$ 脉冲在第 9 位发送，同时不管 SEN 的值如何，SCL 引脚保持低电平（更多详细信息，请参见**第 21.4.4 节 “ 时钟延长 ”**）。通过延长时钟，主器件只有在从器件准备好发送数据时，才发出另一个时钟脉冲。发送的数据必须被装入 SSPBUF 寄存器，同时也被装入 SSPSR 寄存器。然后，应通过将 CKP 位（SSPCON1<4>）置 1 来使能 SCL 引脚。 8 个数据位在 SCL 输入的下降沿被移出。这可确保在 SCL 为高电平期间 SDA 信号是有效的 （图 21-10）。

来自主接收器的 $\overline{ACK}$ 脉冲将在第 9 个 SCL 输入脉冲的上升沿锁存。如果 SDA 线为高电平 （无 $\overline{ACK}$ 应答），那么表示数据传输已完成。在这种情况下，如果从器件锁存了 $\overline{ACK}$，将复位从逻辑，同时从器件监视下一个启动位的出现。如果 SDA 线为低电平 （$\overline{ACK}$），则必须将下一个要发送的数据装入 SSPBUF 寄存器。同样，必须通过将 CKP 位置 1 来使能 SCL 引脚。

每传输一个数据字节都会产生一个 MSSP 中断。SSPIF 位必须用软件清零， SSPSTAT 寄存器用于确定字节的状态。 SSPIF 位在第 9 个时钟脉冲的下降沿被置 1。

图 21-8：　　I²C™ 从模式接收时序（SEN = 0，7 位地址）

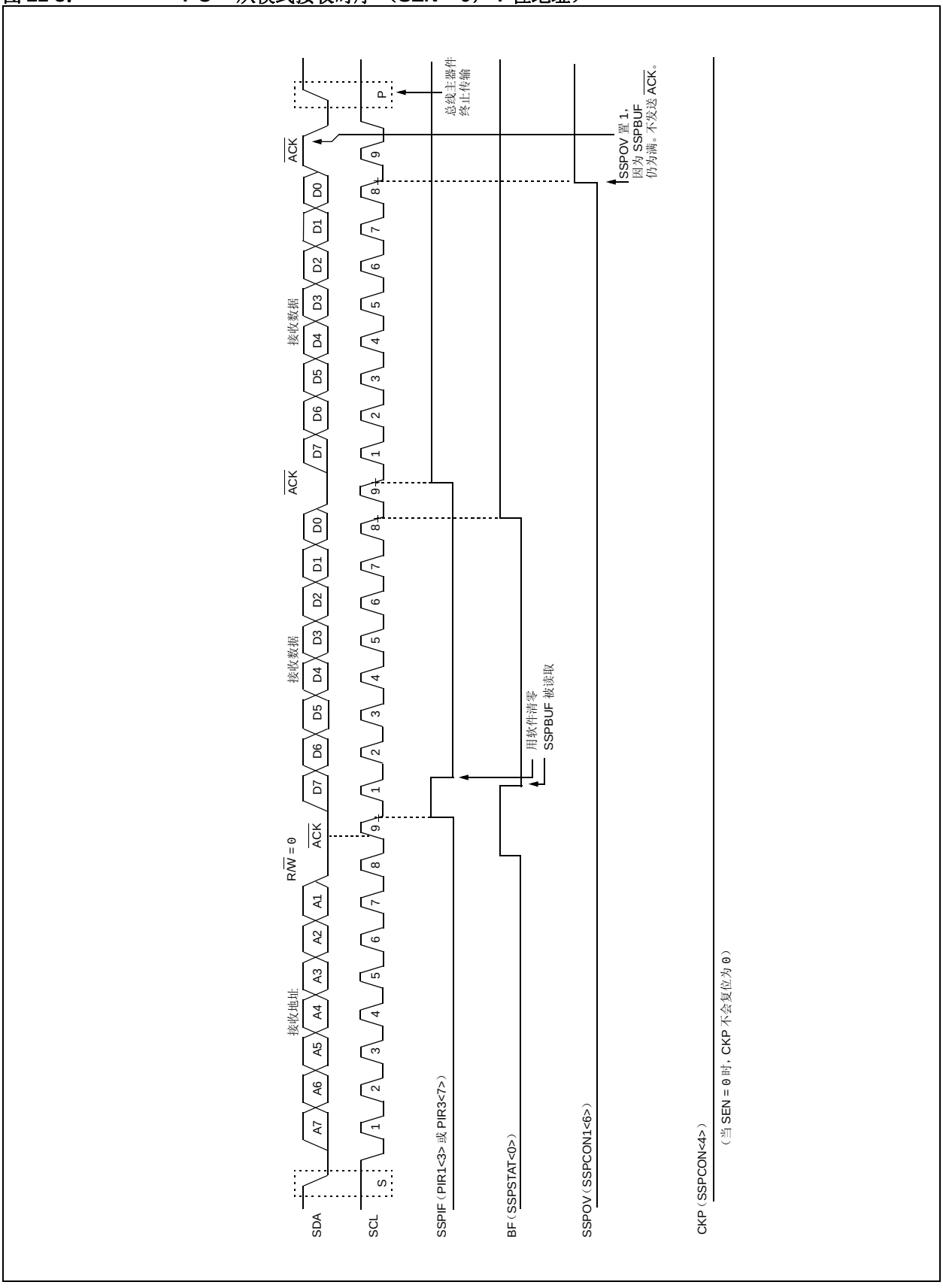
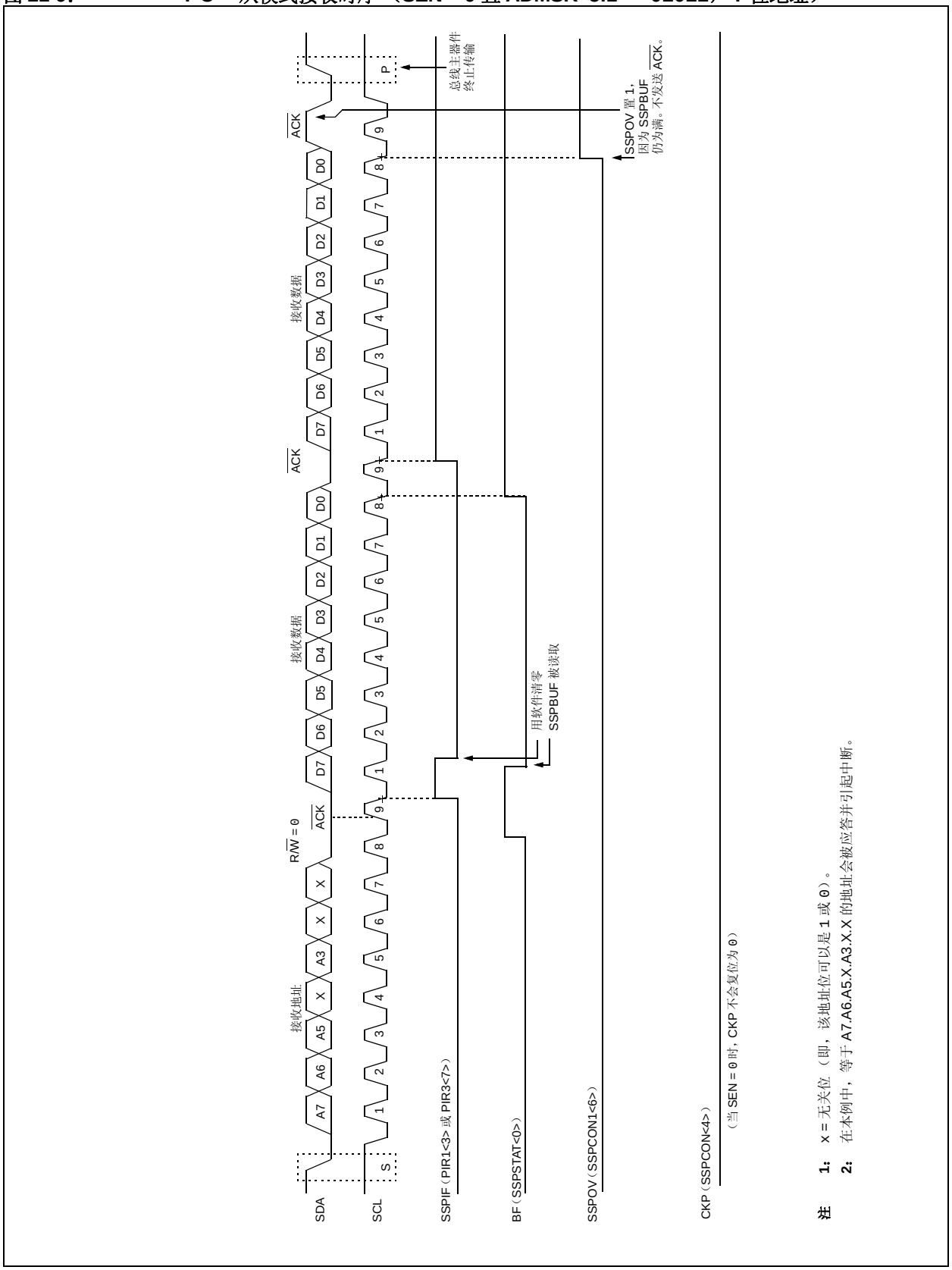


图 21-9：　　I^2C™ 从模式接收时序 （SEN = 0 且 ADMSK<5:1> = 01011， 7 位地址）



注　1： x = 无关位 （即，该地址位可以是 1 或 0）。

2： 在本例中，等于 A7.A6.A5.X.A3.X.X 的地址会被应答并引起中断。

图 21-10：　　I²C™ 从模式发送时序 （7 位地址）

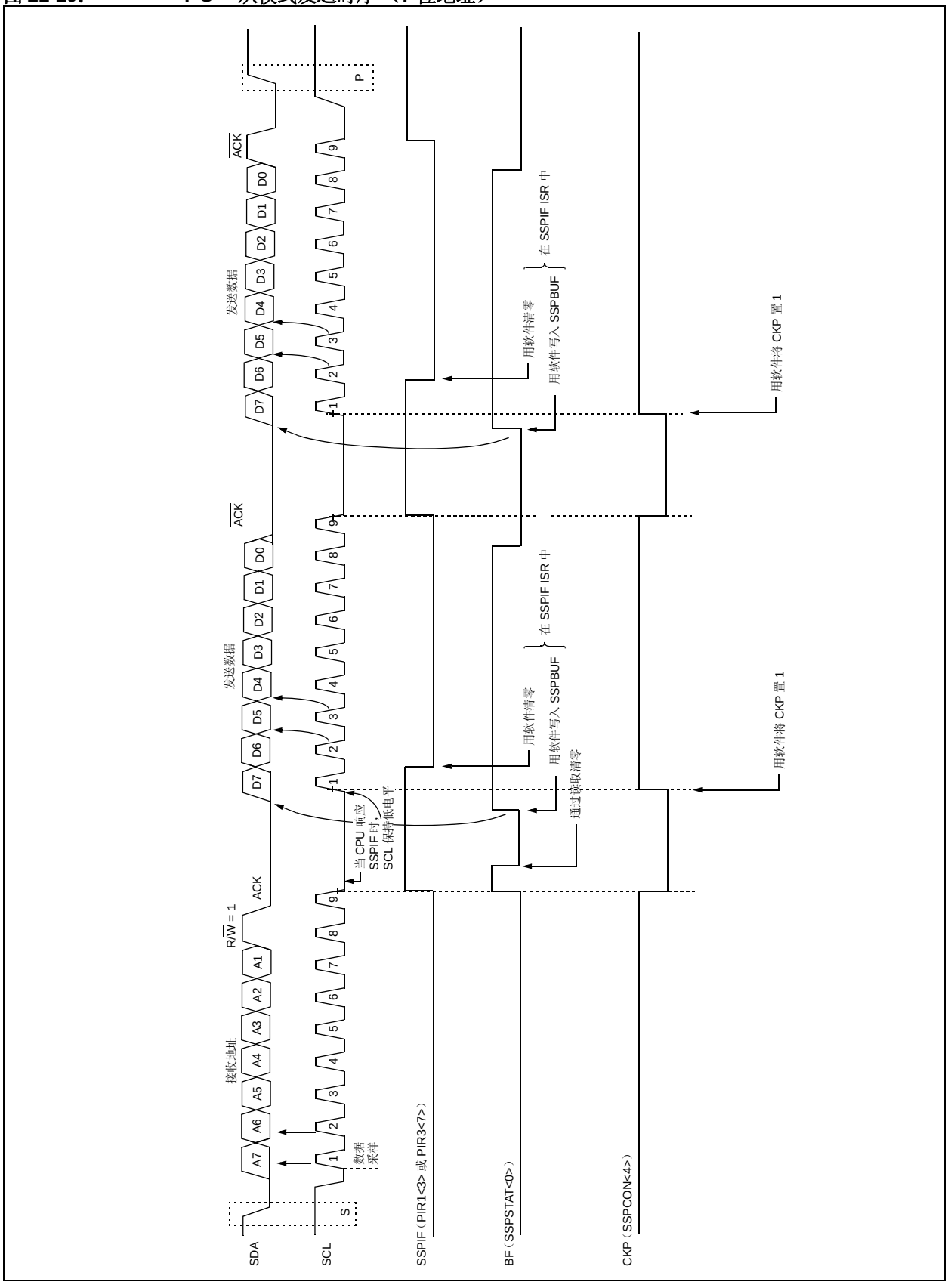
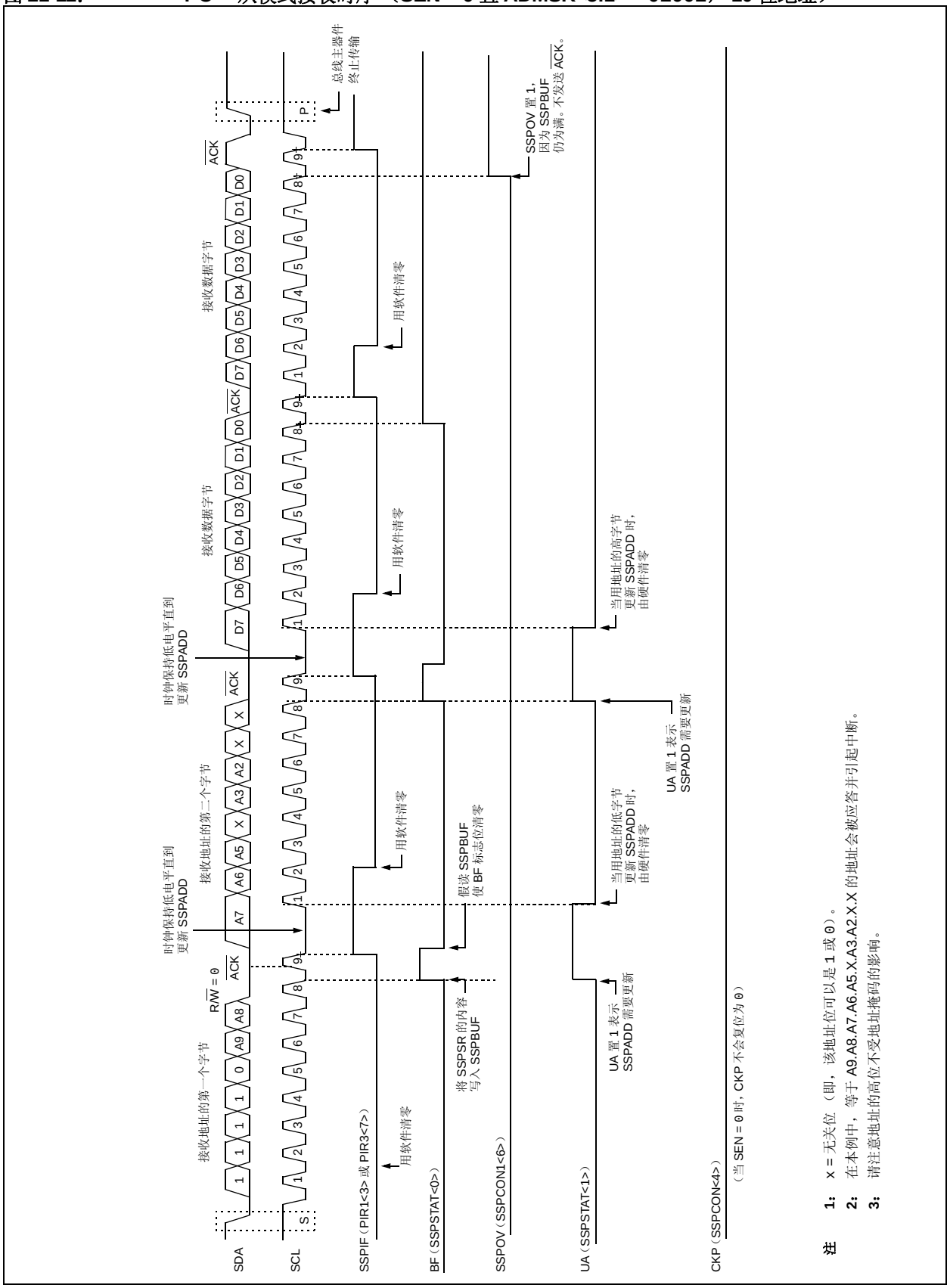


图 21-11：　　I²C™ 从模式接收时序 （SEN = 0 且 ADMSK<5:1> = 01001， 10 位地址）



注　1： x = 无关位（即，该地址位可以是 1 或 0）。

2： 在本例中，等于 A9.A8.A7.A6.A5.X.A3.A2.X.X 的地址会被应答并引起中断。

3： 请注意地址的高位不受地址掩码的影响。

图 21-12：　　I²C™ 从模式接收时序 （SEN = 0， 10 位地址）

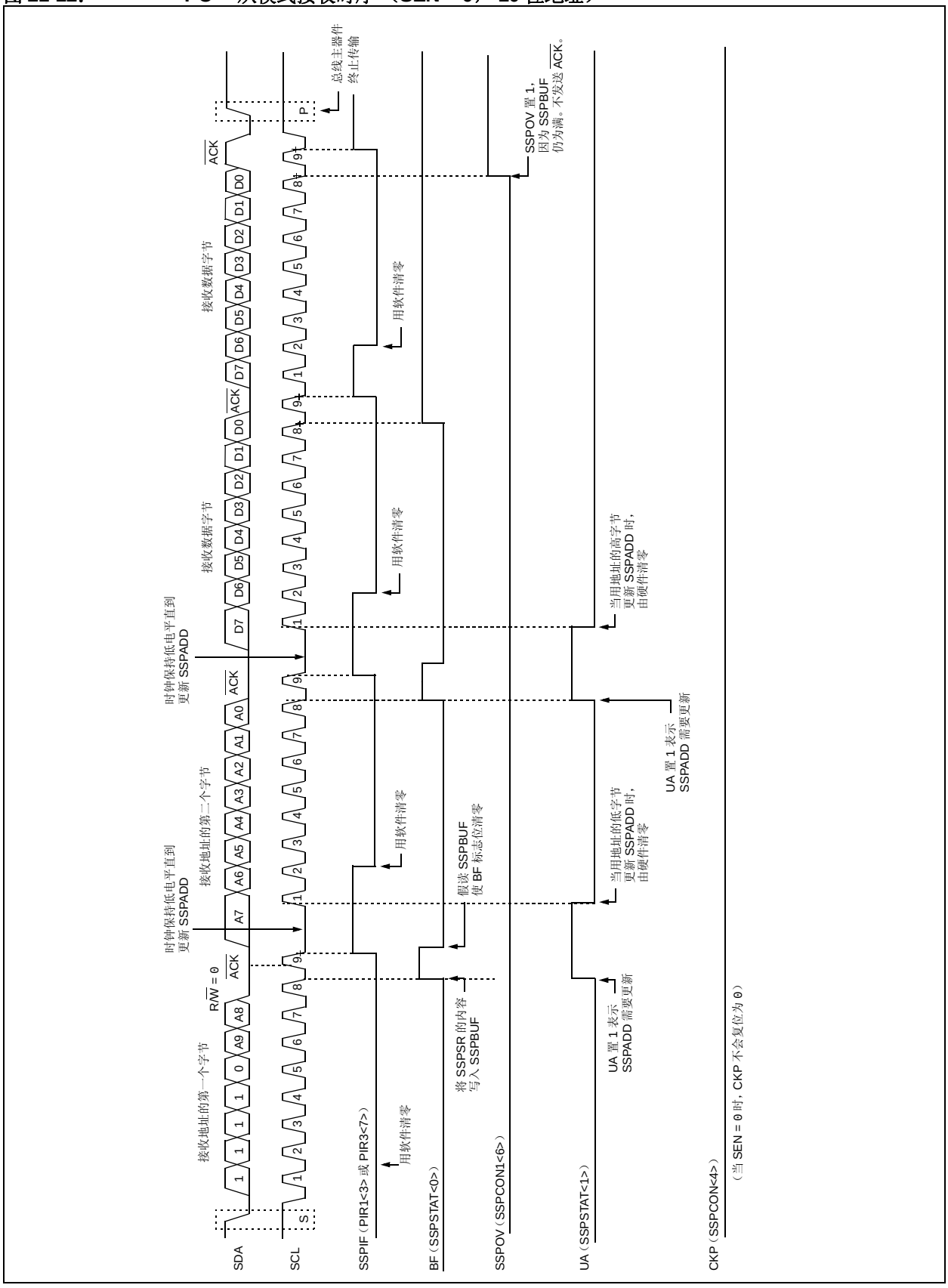
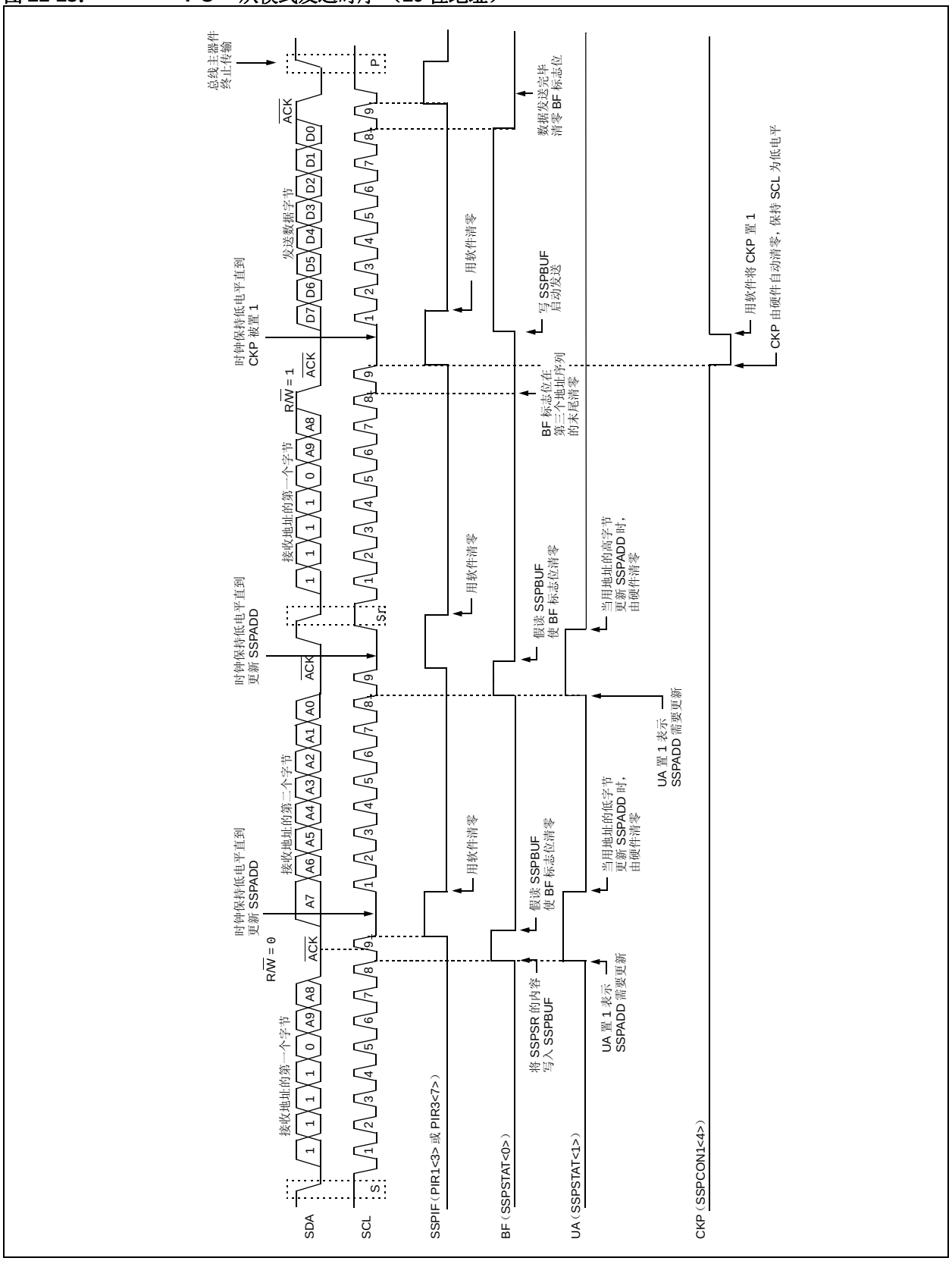


图 21-13：　　I^2C™ 从模式发送时序 （10 位地址）



21.4.4 时钟延长

7 位和 10 位从模式均能在发送序列期间实现自动时钟延长。

SEN 位 （SSPCON2<0>）允许在接收期间使能时钟延长。将 SEN 置 1 将使 SCL 引脚在每个数据接收序列的末尾保持低电平。

21.4.4.1 7 位从接收模式 （SEN = 1）的时钟延长

在 7 位从接收模式下，如果在 ACK 序列末尾的第 9 个时钟的下降沿将 BF 位置 1，则 SSPCON1 寄存器中的 CKP 位就会自动清零，强制 SCL 输出保持在低电平。CKP 位被清零会将 SCL 线拉为低电平。在允许继续接收之前，必须在用户的 ISR 中将 CKP 位置 1。保持 SCL 线为低电平，用户可以在主器件发起另一个接收序列之前，有时间执行 ISR 并读取 SSPBUF 的内容。这将防止发生缓冲区溢出 （见图 21-15）。

注 1： 如果用户在第 9 个时钟的下降沿到来之前读取了 SSPBUF 的内容，使得 BF 位被清零，那么 CKP 位就不会被清零，也不会发生时钟延长。

2： 不管 BF 位的状态如何， CKP 位都可以用软件置 1。为避免溢出，在下一个接收序列开始之前，用户要注意在ISR中清零BF位。

21.4.4.2 10 位从接收模式 （SEN = 1）的时钟延长

在 10 位从接收模式下，在地址序列中会自动发生时钟延长，但是 CKP 位不会被清零。在这期间，如果 UA 位在第 9 个时钟之后置 1，将启动时钟延长。UA 位在接收到 10 位地址的高字节后被置 1，然后接收 10 位地址的第二个字节并清零 R/W 位。在更新 SSPADD 时释放时钟线。如同 7 位模式一样，在每个数据接收序列中均会发生时钟延长。

注： 如果用户在第 9 个时钟的下降沿出现之前查询 UA 位，并通过更新 SSPADD 寄存器清零 UA 位，而且在此之前用户没有读取 SSPBUF 寄存器使 BF 位清零，则 CKP 位的电平仍然不会被拉低。基于 BF 位状态的时钟延长仅在数据序列中出现，不会出现在地址序列中。

21.4.4.3 7 位从发送模式的时钟延长

如果 BF 位被清零， 7 位从发送模式将通过在第 9 个时钟的下降沿出现后清零 CKP 位来实现时钟延长。上述情形与 SEN 位的状态无关。

用户的 ISR 必须先将 CKP 位置 1 才可以继续发送。在保持 SCL 线为低电平期间，用户在主器件发起另一个发送序列之前，将有时间执行 ISR 并装入 SSPBUF 的内容 （见图 21-10）。

注 1： 如果用户在第 9 个时钟的下降沿之前就装入 SSPBUF 的内容，使得 BF 位置 1，那么 CKP 位就不会被清零，也不会发生时钟延长。

2： 不管 BF 位的状态如何， CKP 位都可以用软件置 1。

21.4.4.4 10 位从发送模式的时钟延长

在 10 位从发送模式下，在前两个地址序列中由 UA 位的状态来控制时钟延长，正如同 10 位从接收模式一样。头两个地址后跟着第三个地址序列，该地址序列包含 10 位地址的高位和被置为 1 的 R/W 位。在执行完第三个地址序列后，UA 位不置 1，此时模块配置为发送模式，由 BF 标志位控制时钟延长，正如 7 位从发送模式一样（见图 21-13）。

21.4.4.5 时钟同步和 CKP 位

当 CKP 位被清零时， SCL 输出被强制为 0。然而，将 CKP 位清零不会将 SCL 输出拉为低电平，除非已经采样到 SCL 输出为低电平。因此，CKP 位不会将 SCL 线拉为低电平，除非外部 I^2C 主器件将 SCL 线拉低。SCL 输出将保持低电平，直到 CKP 位置 1 且 I^2C 总线上的所有其他器件将SCL电平拉高为止。这可以确保对CKP位的写操作不会违反 SCL 的最短高电平时间要求 （见图 21-14）。

图 21-14： 时钟同步时序

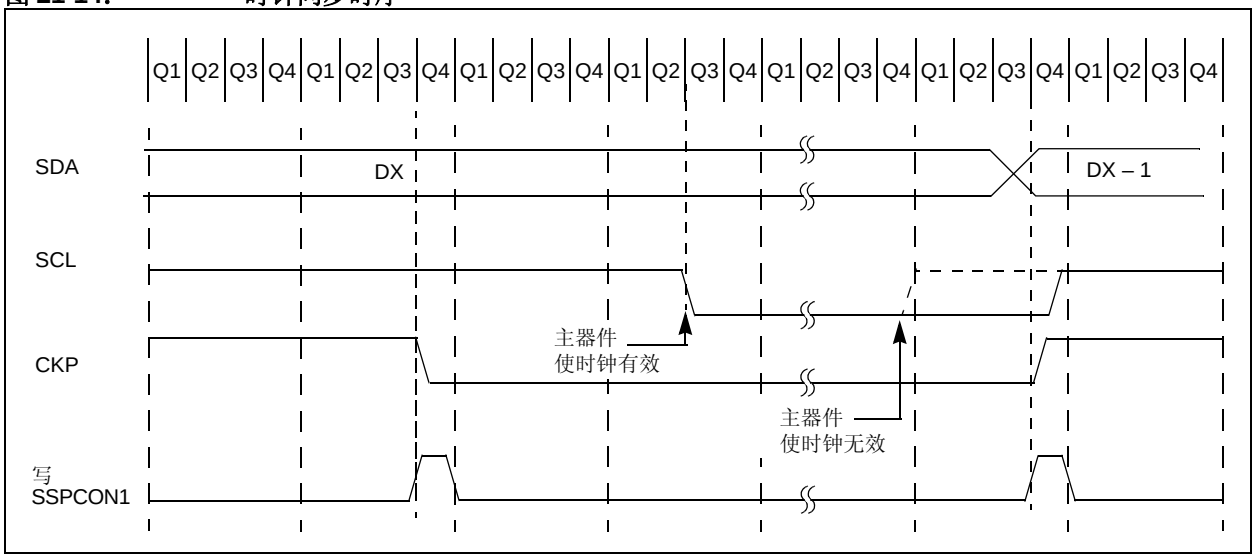


图 21-15：　　I^2C™ 从模式接收时序 （SEN = 1， 7 位地址）

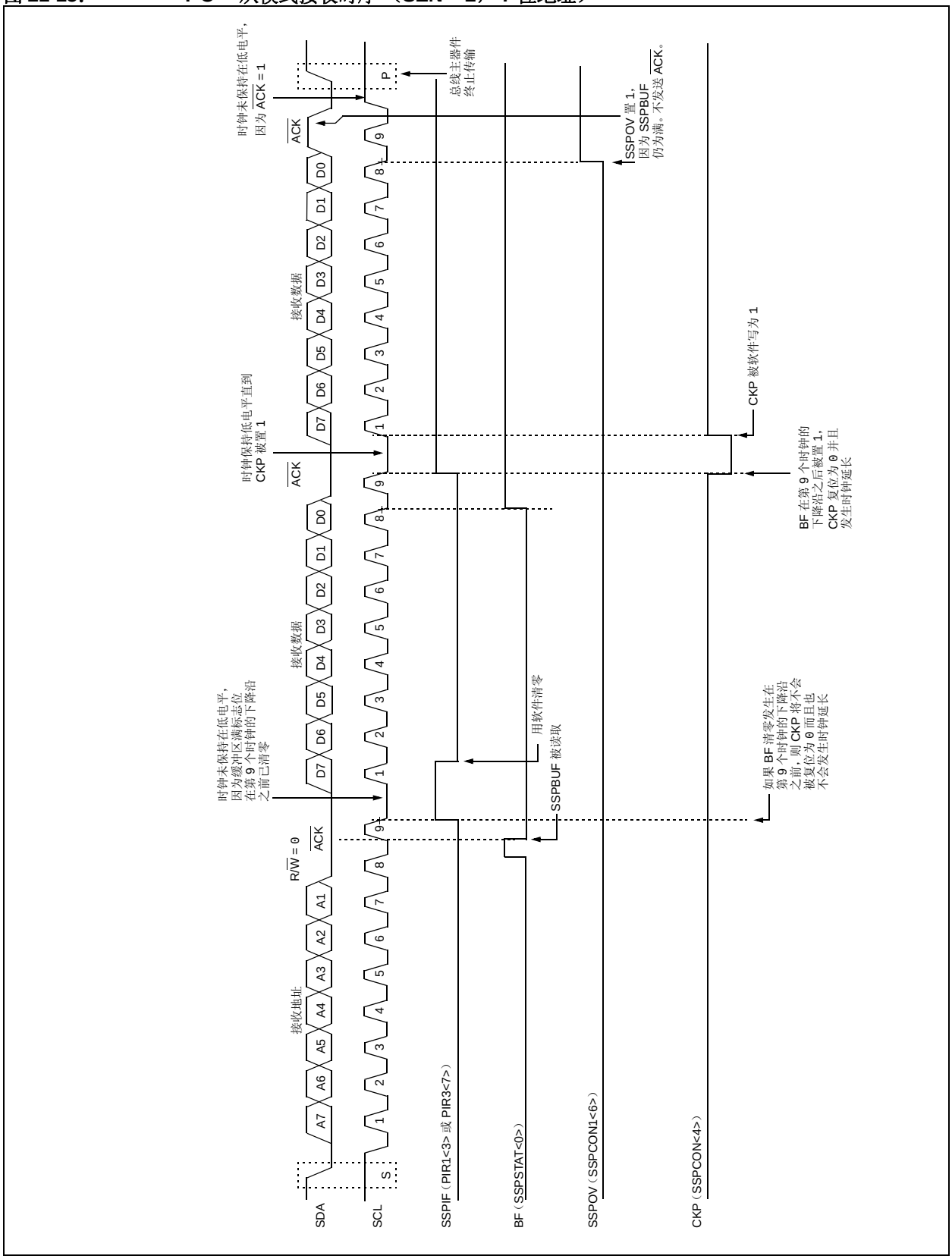
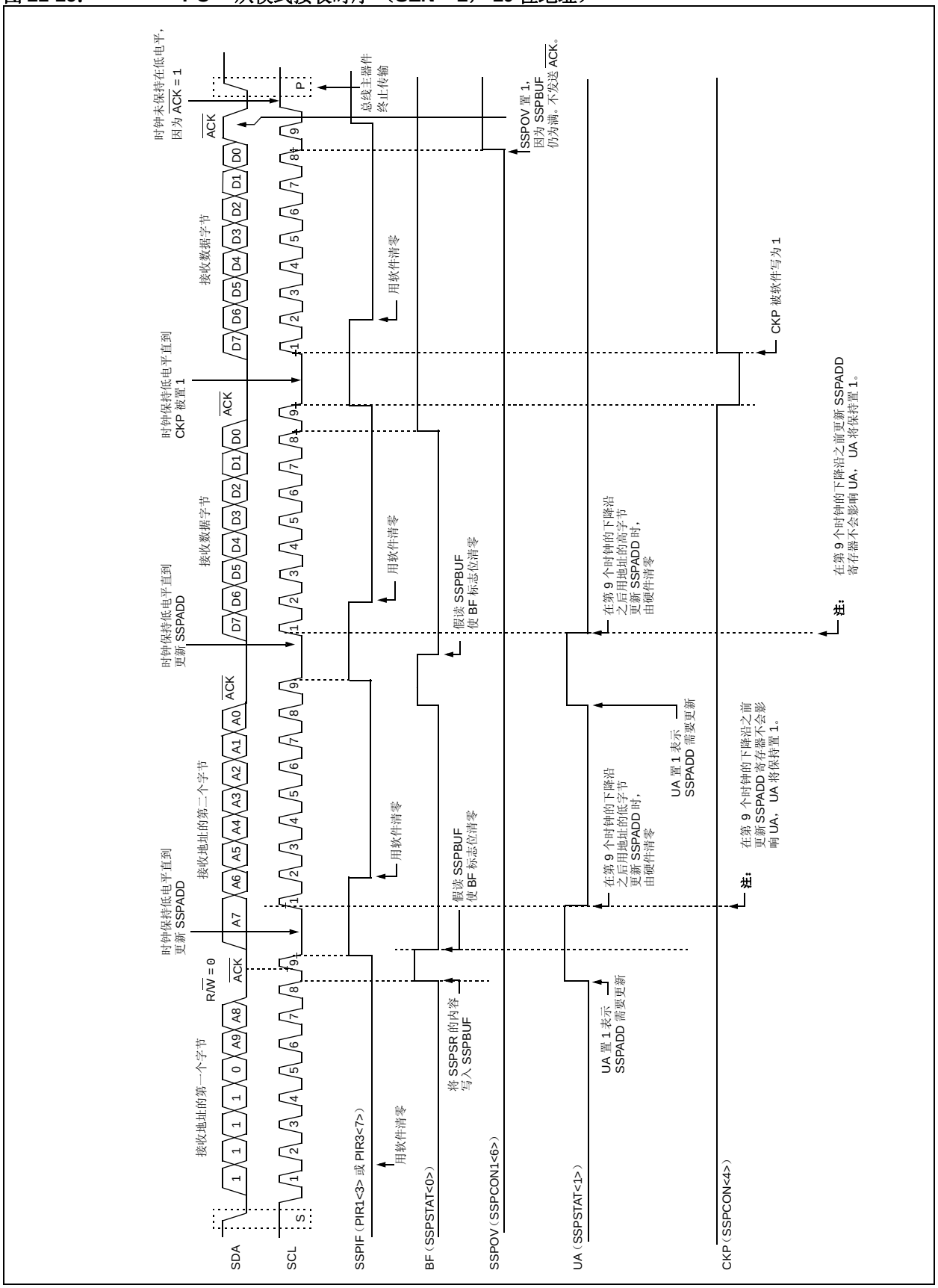


图 21-16：　　I²C™ 从模式接收时序 （SEN = 1， 10 位地址）



21.4.5 广播呼叫地址支持

在 I²C 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，理论上所有的器件都应该发送一个应答信号来响应。

广播呼叫地址是 I²C 协议为特定目的保留的 8 个地址之一。它由全 0 组成，且 R/W = 0。

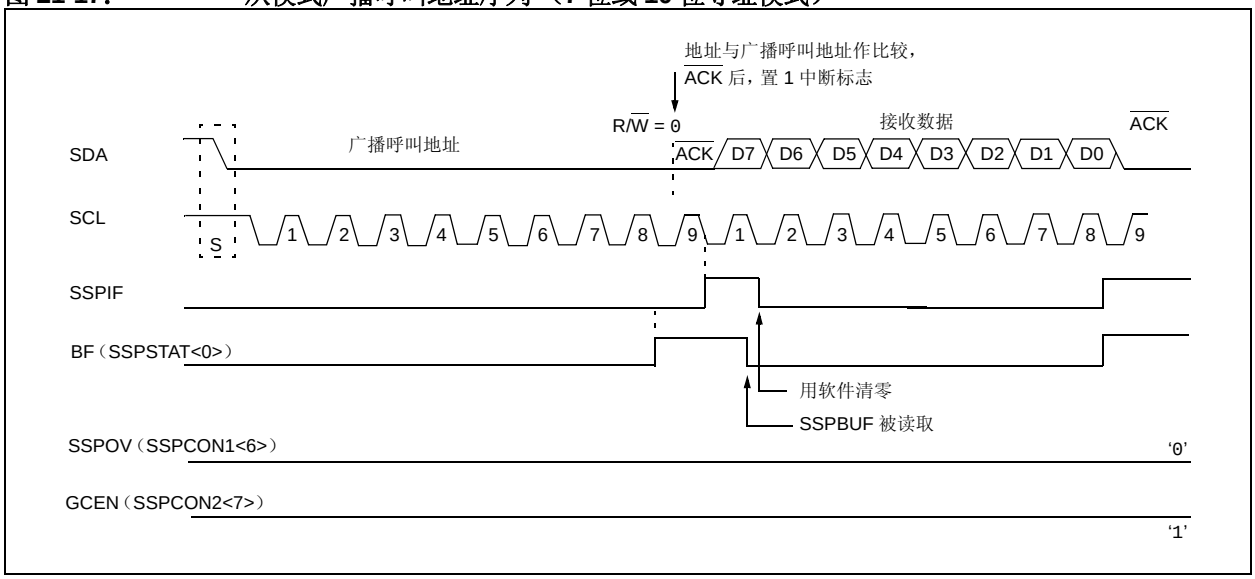
当使能广播呼叫使能位 （GCEN）（SSPCON2<7> 置 1）时，即可识别广播呼叫地址。检测到启动位后，8 位数据会被移入 SSPSR，同时将该地址与 SSPADD 进行比较。它还会与广播呼叫地址进行比较并用硬件设定。

如果与广播呼叫地址匹配， SSPSR 的值将被传输到 SSPBUF， BF 标志位（第 8 位）置 1，并且 SSPIF 中断标志位在第 9 位 （ACK 位）的下降沿置 1。

当中断得到响应时，可以通过读取 SSPBUF 的内容来检查中断源。该值可用于判断是特定器件的地址还是一个广播呼叫地址。

在 10 位寻址模式下，需要更新 SSPADD 来匹配地址的后半部分，同时 UA 位 （SSPSTAT<1>）置 1。如果 GCEN 位置 1 时采样到广播呼叫地址，同时从器件被配置为 10 位寻址模式，则不再需要地址的后半部分，也不会将 UA 位置 1，从器件将在应答后开始接收数据（图 21-17）。

图 21-17： 从模式广播呼叫地址序列 （7 位或 10 位寻址模式）



21.4.6 主模式

通过将 SSPCON1 中的相应 SSPM 位置 1 和清零，同时将 SSPEN 位置 1，可以使能主模式。在主模式下，如果TRIS位被置1，SCL和SDA线由MSSP硬件控制。

主模式通过在检测到启动条件和停止条件时产生中断来工作。停止（P）位和启动（S）位在复位或禁止 MSSP 模块时清零。当 P 位置 1 时，可以取得 I^2C 总线的控制权；或者，总线处于空闲状态， S 位和 P 位都清零。

在固件控制的主模式下，用户代码根据启动位和停止位条件执行所有的 I^2C 总线操作。

一旦使能主模式，用户即可选择以下 6 项操作。

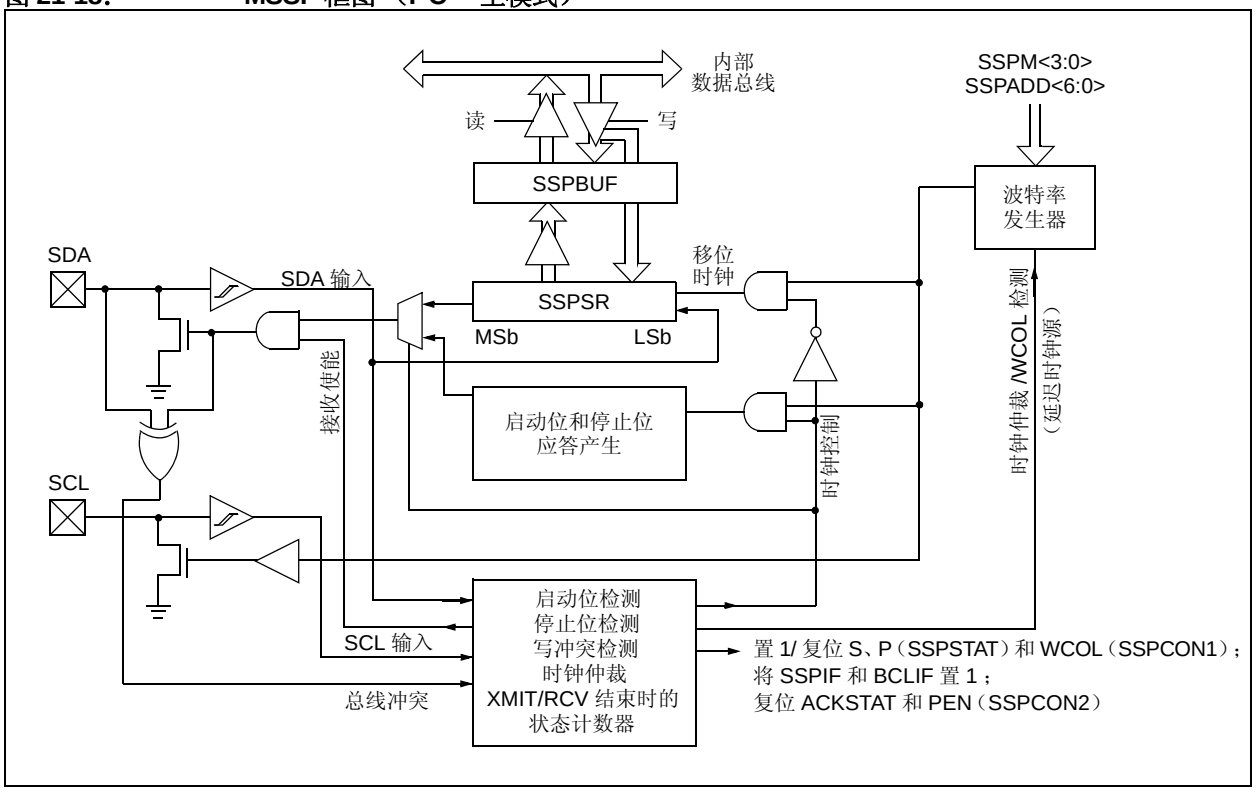
1. 在 SDA 和 SCL 上发出一个启动条件。
2. 在 SDA 和 SCL 上发出一个重复启动条件。
3. 写入 SSPBUF 寄存器，启动数据 / 地址的发送。
4. 配置 I^2C 端口用于接收数据。
5. 在接收数据字节末尾产生应答信号。
6. 在 SDA 和 SCL 上产生一个停止条件。

注：	当配置为 I^2C 主模式时， MSSP 模块不允许事件排队。例如，在启动条件结束前，不允许用户立即写 SSPBUF 寄存器以启动传输。在这种情况下，将不会执行写 SSPBUF， WCOL 位将被置 1，这表明没有发生对 SSPBUF 的写操作。

以下事件会使 MSSP 中断标志位 SSPIF 置 1 （如果允许 MSSP 中断，则产生中断）：

- 启动条件
- 停止条件
- 数据传输字节发送 / 接收
- 应答发送
- 重复启动

图 21-18： MSSP 框图 （I^2C™ 主模式）



21.4.6.1 I²C™ 主模式工作原理

主器件产生所有的串行时钟脉冲、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。

在主发送器模式下，串行数据通过 SDA 输出，而串行时钟由 SCL 输出。发送的第一个字节包括作为接收方的从器件地址（7 位）和读 / 写（R/W）位。在这种情况下，R/W 位将是逻辑 0。一次发送 8 位串行数据。每发送一个字节，会收到一个应答位。输出启动条件和停止条件，表明串行传输的开始和结束。

在主接收模式下，发送的第一个字节包括作为发送方的从器件地址（7 位）和 R/W 位。在这种情况下，R/W 将是逻辑 1。因此，发送的第一个字节是一个 7 位从器件地址，后面跟 1 表示接收。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表明发送的开始和结束。

在 I²C 模式下，将使用 SPI 工作模式中使用的波特率发生器将 SCL 时钟频率设置为 100 kHz、400 kHz 或 1 MHz。更多详细信息，请参见**第 21.4.7 节“波特率”**。

下面是一个典型的发送序列：

1. 用户通过将启动使能位 SEN （SSPCON2<0>）置 1，产生启动条件。
2. SSPIF 置 1。在进行任何其他操作前，MSSP 模块将等待所需的启动时间。
3. 用户将从器件地址装入 SSPBUF 进行发送。
4. 器件地址从 SDA 引脚移出，直到发送完所有 8 位地址数据。
5. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入 SSPCON2 寄存器（SSPCON2<6>）。
6. MSSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
7. 用户将 8 位数据装入 SSPBUF。
8. 数据从SDA引脚移出，直到发送完所有8位数据。
9. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入 SSPCON2 寄存器（SSPCON2<6>）。
10. MSSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
11. 用户通过将停止使能位 PEN （SSPCON2<2>）置 1，产生停止条件。
12. 一旦停止条件完成，将产生一个中断。

21.4.7 波特率

在I²C主模式下，波特率发生器（Baud Rate Generator，BRG）的重载值存放在 SSPADD 寄存器的低 7 位（图 21-19）。当发生对 SSPBUF 的写操作时，波特率发生器将自动开始计数。BRG 会递减计数至 0，然后停止直到再次发生重载。 BRG 计数器会在每个指令周期（TCY）中的 Q2 和 Q4 时钟周期上进行两次递减计数。在 I²C 主模式下，会自动重载 BRG。

如果指定操作完成（即，在传输的最后一个数据位后面跟着 ACK），内部时钟将自动停止计数，SCL 引脚将保持在其最后的状态。

表 21-3 给出了不同的指令周期下的时钟频率以及装入 SSPADD 的 BRG 值。不支持 SSPADD BRG 值为 00h。

图 21-19： 波特率发生器框图

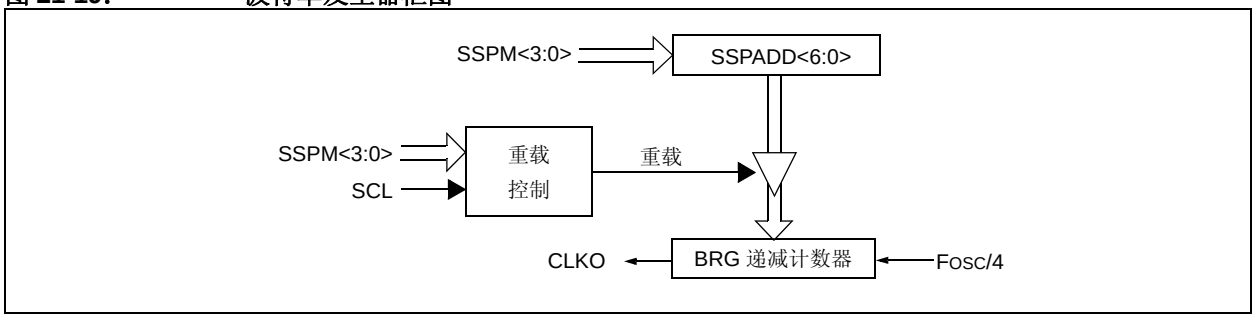


表 21-3： 使用 BRG 的 I²C™ 时钟频率

FOSC	FCY	FCY * 2	BRG 值	FSCL（两次 BRG 计满返回）
40 MHz	10 MHz	20 MHz	18h	400 kHz(1)
40 MHz	10 MHz	20 MHz	1Fh	312.5 kHz
40 MHz	10 MHz	20 MHz	63h	100 kHz
16 MHz	4 MHz	8 MHz	09h	400 kHz(1)
16 MHz	4 MHz	8 MHz	0Ch	308 kHz
16 MHz	4 MHz	8 MHz	27h	100 kHz
4 MHz	1 MHz	2 MHz	02h	333 kHz(1)
4 MHz	1 MHz	2 MHz	09h	100 kHz
16 MHz(2)	4 MHz	8 MHz	03h	1 MHz(1)

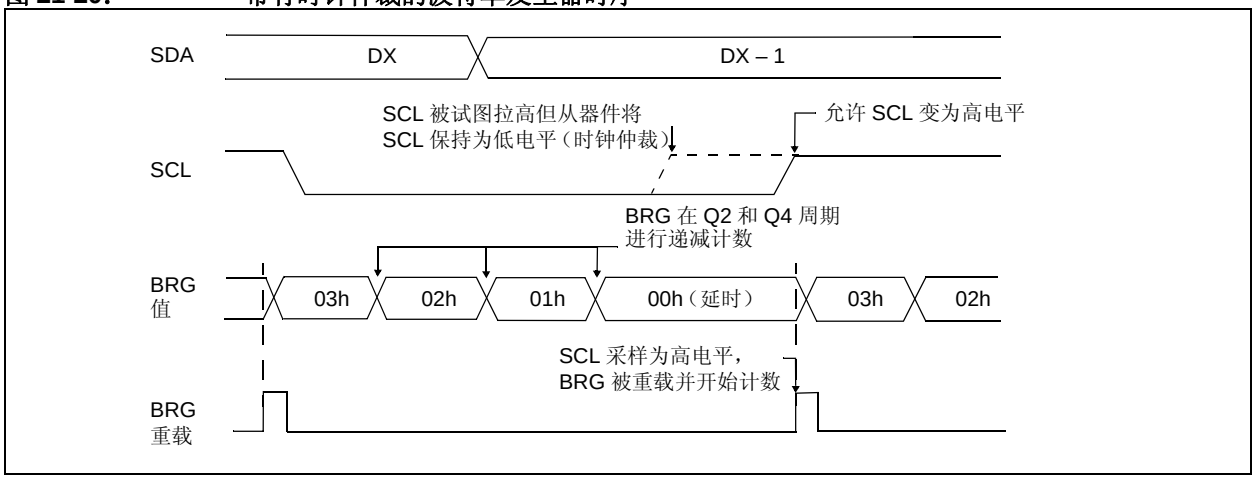
注 1： 虽然 I²C 接口各方面都不符合 400 kHz I²C 规范（该规范适用于大于 100 kHz 的频率），但在需要较高频率的应用场合可以慎重使用。

2： 对于 1 MHz I²C，要求 FOSC 最低频率为 16 MHz。

21.4.7.1 时钟仲裁

如果在任何接收、发送或重复启动 / 停止条件期间，主器件拉高了 SCL 引脚（允许 SCL 引脚悬空为高电平），就会发生时钟仲裁。当允许 SCL 引脚悬空为高电平时，波特率发生器 （BRG）暂停计数，直到 SCL 引脚被实际采样到高电平为止。当 SCL 引脚采样为高电平时，波特率发生器重新装入 SSPADD<6:0> 的内容并开始计数。这可以保证当外部器件将时钟拉低时，SCL 在至少一个BRG计满返回计数周期内保持高电平（图21-20）。

图 21-20： 带有时钟仲裁的波特率发生器时序



21.4.8 I²C™ 主模式启动条件时序

要发出启动条件，用户应将启动条件使能位 SEN（SSPCON2<0>）置 1。当 SDA 和 SCL 引脚采样为高电平时，波特率发生器重新装入 SSPADD<6:0> 的内容并开始计数。如果波特率发生器发生超时（TBRG）时， SCL 和 SDA 都采样为高电平，则 SDA 引脚被驱动为低电平。当 SCL 为高电平时，将 SDA 驱动为低电平将产生启动条件，并使 S 位（SSPSTAT<3>）置 1。随后波特率发生器重新装入 SSPADD<6:0> 的内容并恢复计数。当波特率发生器超时（TBRG）时，SEN 位（SSPCON2<0>）将自动被硬件清零，波特率发生器暂停工作， SDA 线保持低电平，启动条件结束。

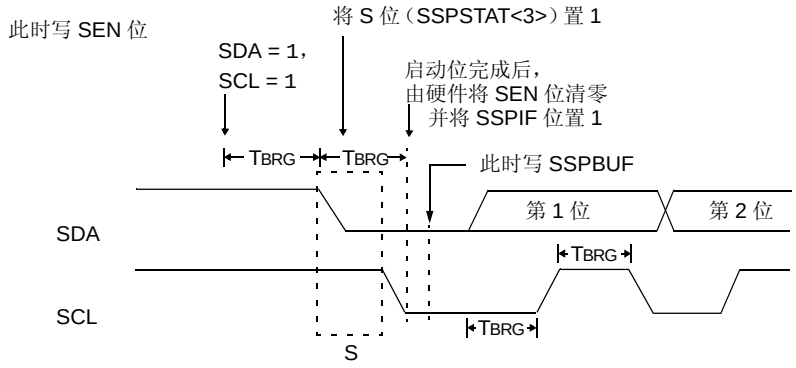
注：	如果在启动条件开始时，SDA 和 SCL 引脚已经采样为低电平，或者在启动条件期间，SCL 在 SDA 线被驱动为低电平之前已经采样为低电平，则会发生总线冲突。总线冲突中断标志位 BCLIF 置 1，启动条件中止， I²C 模块复位到空闲状态。

21.4.8.1 WCOL 状态标志

如果用户在启动序列过程中写 SSPBUF，则 WCOL 位被置 1，同时缓冲区内容不变 （写操作无效）。

注：	由于不允许事件排队，在启动条件结束之前，不能写 SSPCON2 的低 5 位。

图 21-21： 第一个启动位时序



21.4.9 I²C™ 主模式重复启动条件时序

将 RSEN 位（SSPCON2<1>）编程为高电平，并且 I²C 逻辑模块处于空闲状态时，就会产生重复启动条件。当 RSEN 位置 1 时， SCL 引脚被拉为低电平。当 SCL 引脚采样为低电平时，波特率发生器装入 SSPADD<5:0> 的内容并开始计数。在该波特率发生器计数周期（TBRG）内 SDA 引脚被释放 （被拉高）。当波特率发生器超时时，如果 SDA 采样为高电平， SCL 引脚将被拉高。当 SCL 被采样为高电平时，波特率发生器重新装入 SSPADD<6:0> 的内容并开始计数。 SDA 和 SCL 必须在一个计数周期 TBRG 内采样为高电平。接下来，在一个 TBRG 中，将 SDA 引脚驱动为低电平（SDA = 0），同时 SCL 保持高电平。随后 RSEN 位（SSPCON2<1>）将自动清零，这次波特率发生器不会重载，SDA 引脚保持低电平。一旦在 SDA 和 SCL 引脚上检测到启动条件，S 位 （SSPSTAT<3>）将被置 1。直到波特率发生器发生超时后， SSPIF 位才会置 1。

注 1： 有任何其他事件在进行时，编程设置 RSEN 无效。

2： 在重复启动条件期间，以下事件将会导致总线冲突发生：

- 当 SCL 由低电平变为高电平时， SDA 采样为低电平。
- 在 SDA 被置为低电平之前， SCL 变为低电平。这表明另一个主器件正试图发送一个数据 1。

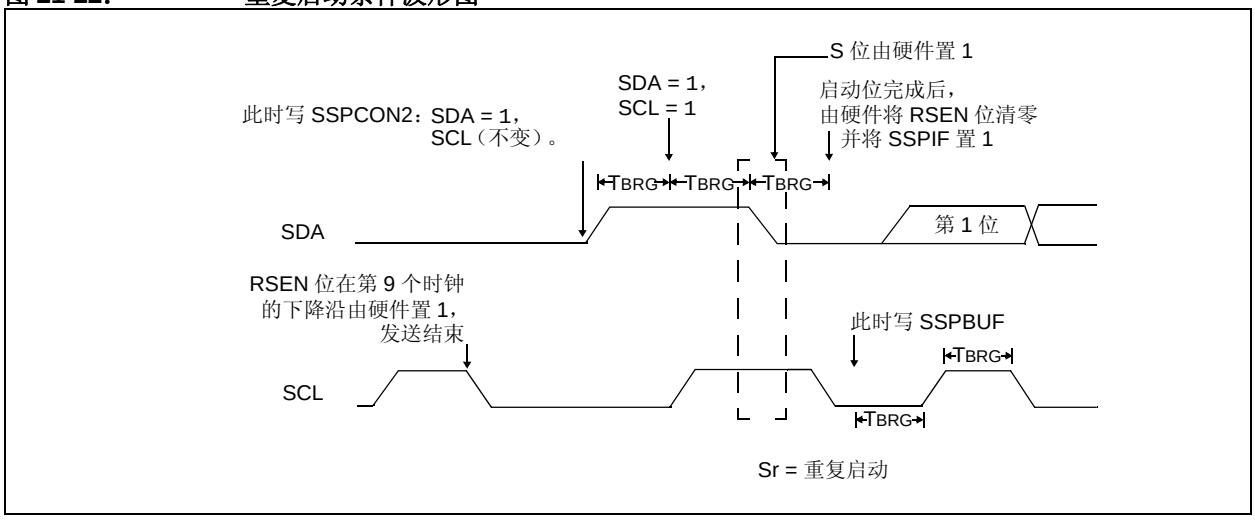
一旦 SSPIF 位被置 1，用户便可以在 7 位地址模式下将 7 位地址，或者在 10 位地址模式下将默认的第一个地址字节写入 SSPBUF。在发送完第一个 8 位数据并接收到一个 ACK 后，用户可以发送另外 8 位地址（10 位模式）或 8 位数据 （7 位模式）。

21.4.9.1 WCOL 状态标志

如果用户在重复启动序列过程中写 SSPBUF，则 WCOL 被置 1，同时缓冲区内容不变 （写操作无效）。

注： 由于不允许事件排队，在重复启动条件结束之前，不能写 SSPCON2 的低 5 位。

图 21-22： 重复启动条件波形图



21.4.10 I²C™ 主模式发送

发送一个数据字节、一个 7 位地址或一个 10 位地址的一半，都是通过写一个值到 SSPBUF 寄存器来实现的。该操作将使缓冲区满标志位 BF 置 1，波特率发生器开始计数，同时开始下一次发送。在 SCL 的下降沿有效后（见数据保持时间规范参数 106），地址 / 数据的每一位被移出至 SDA 引脚。在一个波特率发生器计满返回计数周期（TBRG）内，SCL 保持低电平。数据应该在 SCL 释放为高电平前保持有效（见数据建立时间规范参数 107）。当 SCL 引脚释放为高电平时，它将在一个 TBRG 内保持高电平状态。在此期间以及 SCL 的下一个下降沿之后的一段时间内，SDA引脚上的数据必须保持稳定。在第 8 位数据被移出（第 8 个时钟的下降沿）之后，BF 标志位被清零，同时主器件释放 SDA。此时如果发生地址匹配或是数据被正确接收，被寻址的从器件将在第9个时钟周期发出一个ACK位作为响应。ACK 的状态在第 9 个时钟周期的下降沿写入 ACKDT 位。主器件接收到应答之后，应答状态位 ACKSTAT 会被清零。如果未收到应答，则该位被置 1。第 9 个时钟之后，SSPIF 位会置 1，主时钟（波特率发生器）暂停，直到下一个数据字节装入 SSPBUF，SCL 保持低电平，SDA 保持不变（图 21-23）。

在写 SSPBUF 之后，地址的每一位在 SCL 的下降沿被移出，直到所有 7 个地址位和 R/W 位都被移出。在第 8 个时钟的下降沿，主器件将 SDA 引脚拉为高电平，以允许从器件发出一个应答响应。在第 9 个时钟的下降沿，主器件通过采样 SDA 引脚来判断地址是否被从器件识别。ACK 位的状态被装入 ACKSTAT 状态位（SSPCON2<6>）。在发送地址的第 9 个时钟下降沿之后，SSPIF 标志位置 1，BF 标志位清零，波特率发生器关闭直到下一次写 SSPBUF，且 SCL 保持低电平，允许 SDA 悬空。

21.4.10.1 BF 状态标志

在发送模式下，BF 位（SSPSTAT<0>）在 CPU 写 SSPBUF 时置 1，在所有 8 位数据移出后清零。

21.4.10.2 WCOL 状态标志

如果用户在发送过程中（即，SSPSR 仍在移出数据字节时）写 SSPBUF，则 WCOL 位被置 1，并且在写 SSPBUF 之后的 2 个 TCY 内缓冲区内容不变（写操作无效）。如果在 2 个 TCY 内 SSPBUF 被重新写入，则 WCOL 位被置 1 并且 SSPBUF 被更新。这可能导致传输被破坏。

用户应在每次写 SSPBUF 后检查 WCOL 位是否清零，以确保传输正确。在所有情况下，WCOL 都必须用软件清零。

21.4.10.3 ACKSTAT 状态标志

在发送模式下，当从器件已发送应答响应（$\overline{ACK}$ = 0）时，ACKSTAT 位（SSPCON2<6>）清零；当从器件没有应答（$\overline{ACK}$ = 1）时，该位置 1。从器件在识别出其地址（包括广播呼叫地址）或正确接收数据后，会发送一个应答。

21.4.11 I²C™ 主模式接收

通过编程接收使能位 RCEN（SSPCON2<3>）使能主模式接收。

注：	将 RCEN 位置 1 前，MSSP 必须处于无效状态，否则对 RCEN 位置 1 将无效。

波特率发生器开始计数，每次计满返回时，SCL 引脚的状态发生改变（由高变低或由低变高），数据被移入 SSPSR。第 8 个时钟的下降沿之后，接收使能标志位自动清零，SSPSR 的内容装入 SSPBUF，BF 标志位置 1，SSPIF 标志位置 1，波特率发生器暂停计数，且 SCL 保持为低电平。此时 MSSP 处于空闲状态，等待下一条命令。当 CPU 读缓冲区时，BF 标志位将自动清零。通过将应答序列使能位 ACKEN（SSPCON2<4>）置 1，用户可以在接收结束时发送应答位。

21.4.11.1 BF 状态标志

接收数据过程中，将地址或数据字节从 SSPSR 装入 SSPBUF 时，BF 位置 1。在读 SSPBUF 寄存器时将其清零。

21.4.11.2 SSPOV 状态标志

接收数据过程中，当SSPSR接收到8位数据时，SSPOV 位置 1，BF 标志位已经在上一次接收时置 1。

21.4.11.3 WCOL 状态标志

如果用户在接收过程中（即，SSPSR 仍在移入数据字节时）写 SSPBUF，则 WCOL 位被置 1，同时缓冲区内容不变（写操作无效）。

图 21-23：　　I²C™ 主模式发送波形图 （7 位或 10 位地址）

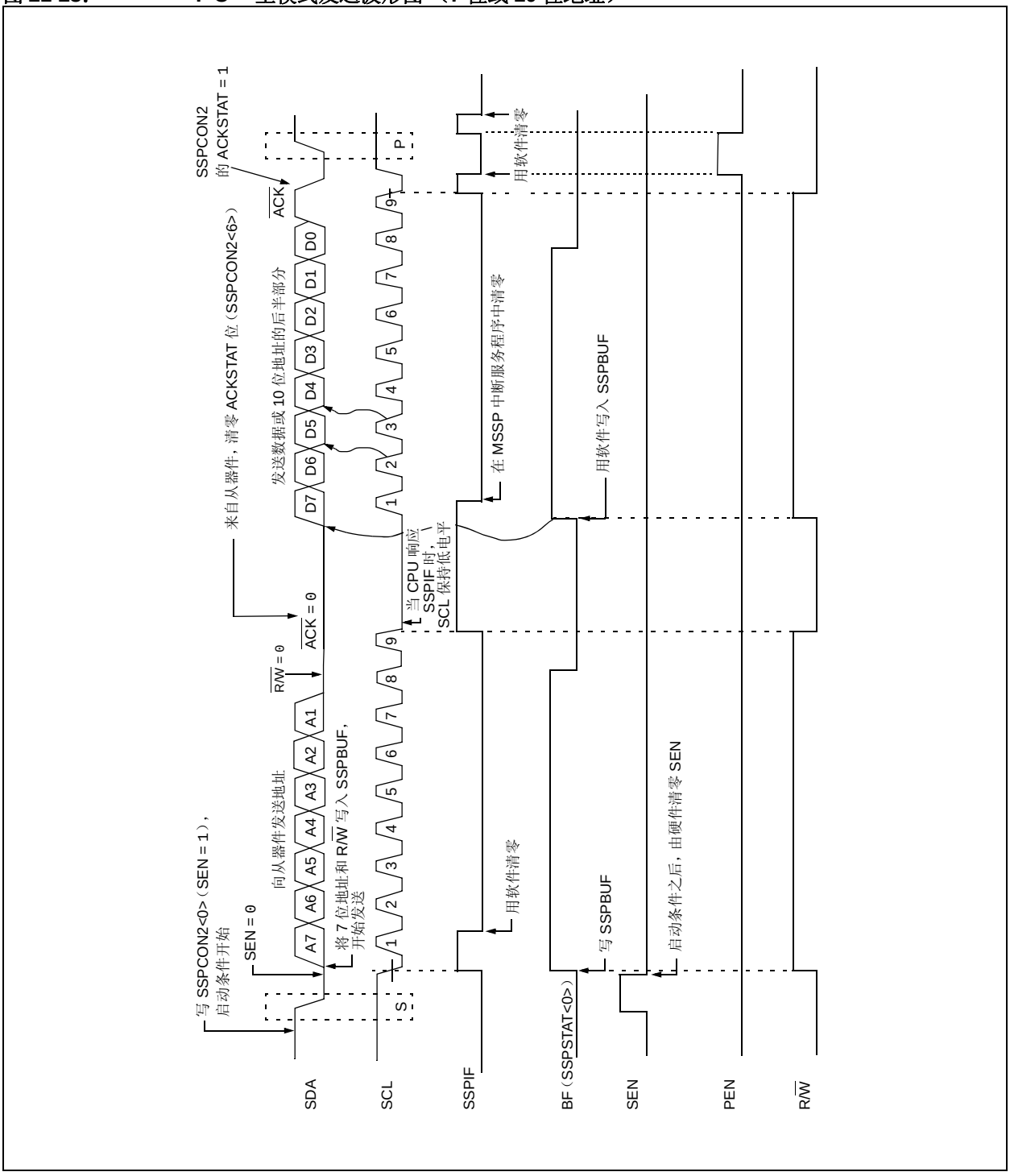
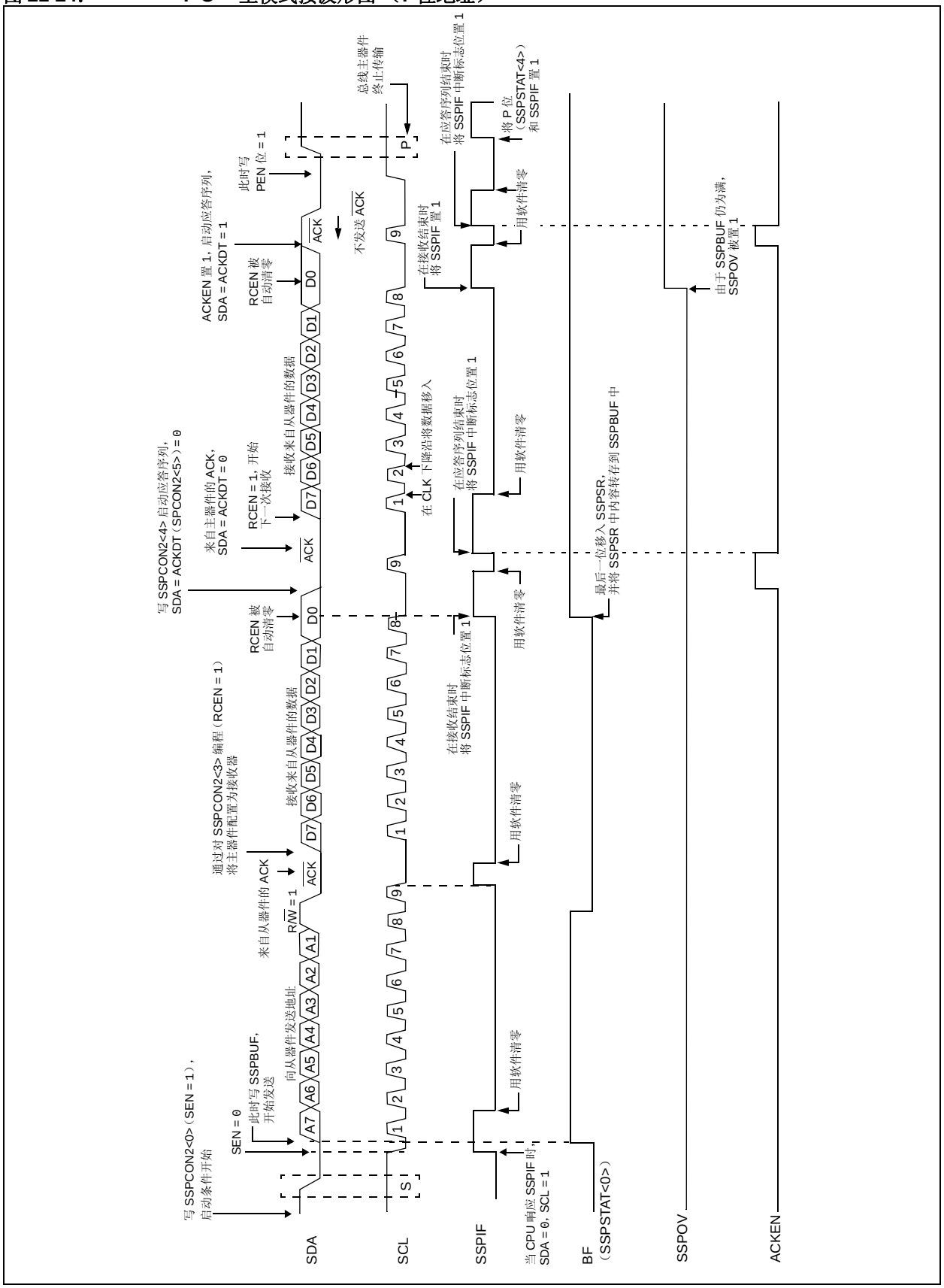


图 21-24：　　I^2C™ 主模式接波形图 （7 位地址）



21.4.12 应答序列时序

将应答序列使能位 ACKEN （SSPCON2<4>）置 1 即可使能应答序列。当该位被置 1 时， SCL 引脚被拉低，应答数据位的内容输出到 SDA 引脚上。如果用户希望产生一个应答，则应该将 ACKDT 位清零。否则，用户要在应答序列开始前将 ACKDT 位置 1。然后波特率发生器进行一个计满返回周期（TBRG）的计数，随后 SCL 引脚电平被拉高。当 SCL 引脚采样为高电平 （时钟仲裁）时，波特率发生器再进行一个 TBRG 周期的计数；然后 SCL 引脚被拉低。随后 ACKEN 位自动清零，波特率发生器关闭， MSSP 模块进入无效状态 （图 21-25）。

21.4.12.1 WCOL 状态标志

如果用户在应答序列进行过程中写SSPBUF，则WCOL被置 1，同时缓冲区内容不变 （写操作无效）。

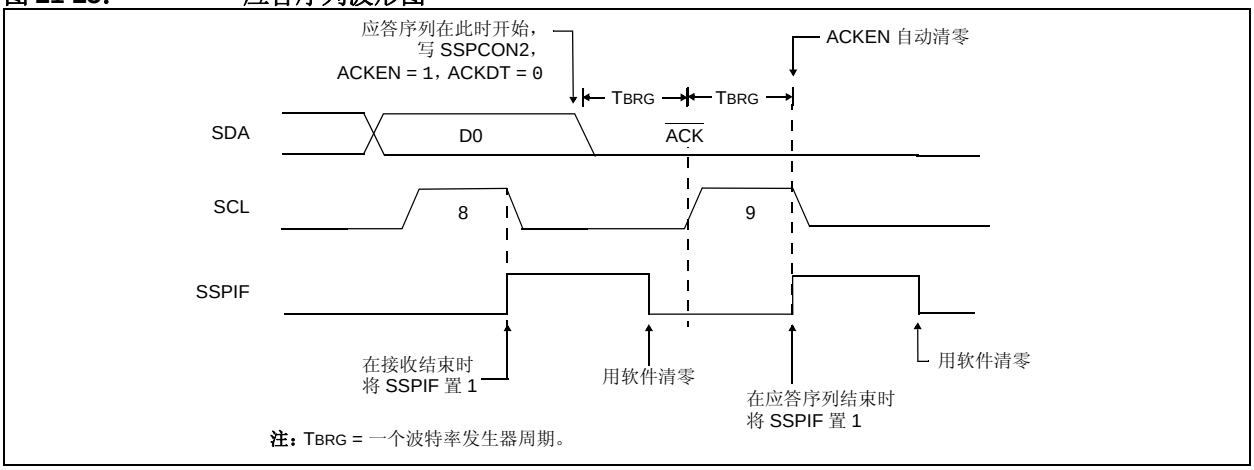
21.4.13 停止条件时序

如果将停止序列使能位 PEN （SSPCON2<2>）置 1，则在接收 / 发送结束时，SDA 引脚上将产生停止位。在接收 / 发送结束时，SCL 线在第 9 个时钟的下降沿后保持低电平。当 PEN 位置 1 时，主器件将 SDA 线置为低电平。当 SDA 线采样为低电平时，波特率发生器被重载并递减计数至 0。当波特率发生器发生超时时， SCL 引脚被拉为高电平，在一个 TBRG （波特率发生器计满返回周期）之后， SDA 引脚将被拉高。当 SDA 引脚采样为高电平且SCL也是高电平时，P位（SSPSTAT<4>）置 1。另一个 TBRG 之后， PEN 位被清零，同时 SSPIF 位被置 1 （图 21-26）。

21.4.13.1 WCOL 状态标志

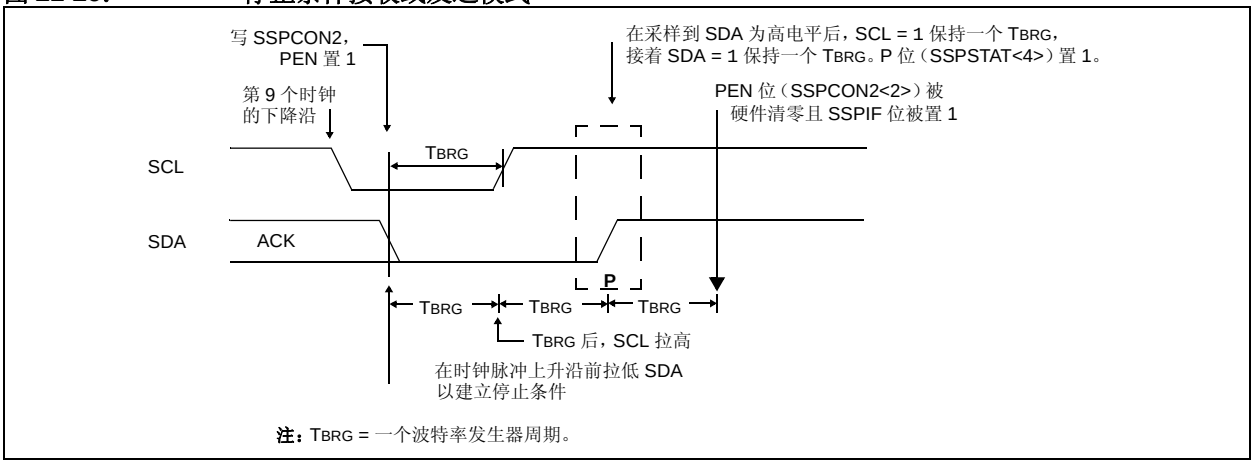
如果用户在停止序列进行过程中写SSPBUF，则WCOL位被置 1，同时缓冲区内容不变 （写操作无效）。

图 21-25： 应答序列波形图



注： TBRG = 一个波特率发生器周期。

图 21-26： 停止条件接收或发送模式



注： TBRG = 一个波特率发生器周期。

21.4.14 休眠模式下的操作

在休眠模式下，I²C 模块能够接收地址或数据，并且当地址匹配或字节传输完成时，如果允许 MSSP 中断，将唤醒处理器。

21.4.15 复位的影响

复位会禁止 MSSP 模块并终止当前的数据传输。

21.4.16 多主器件模式

在多主器件模式下，在检测到启动和停止条件时将产生中断，这可用于判断总线是否空闲。停止（P）位和启动（S）位在复位或禁止 MSSP 模块时清零。当 P 位（SSPSTAT<4>）置 1 时，可以取得 I²C 总线的控制权；或者，总线处于空闲状态， S 位和 P 位都清零。当总线忙时，当发生停止条件时，将产生 MSSP 中断。

在多主器件操作中，必须监视 SDA 线来进行仲裁，以查看信号电平是否为期望的输出电平。此操作由硬件实现，其结果保存在 BCLIF 位中。

可能导致仲裁失败的情况是：

- 地址传输
- 数据传输
- 启动条件
- 重复启动条件
- 应答条件

21.4.17 多主器件通信、总线冲突和总线仲裁

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 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。如果 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 端口复位到空闲状态 （图 21-27）。

如果在发送过程中发生总线冲突，则发送操作停止，BF 标志位被清零， SDA 和 SCL 线被拉高，并且可写入 SSPBUF。当执行总线冲突中断服务程序时，如果 I²C 总线空闲，用户可通过发出启动条件恢复通信。

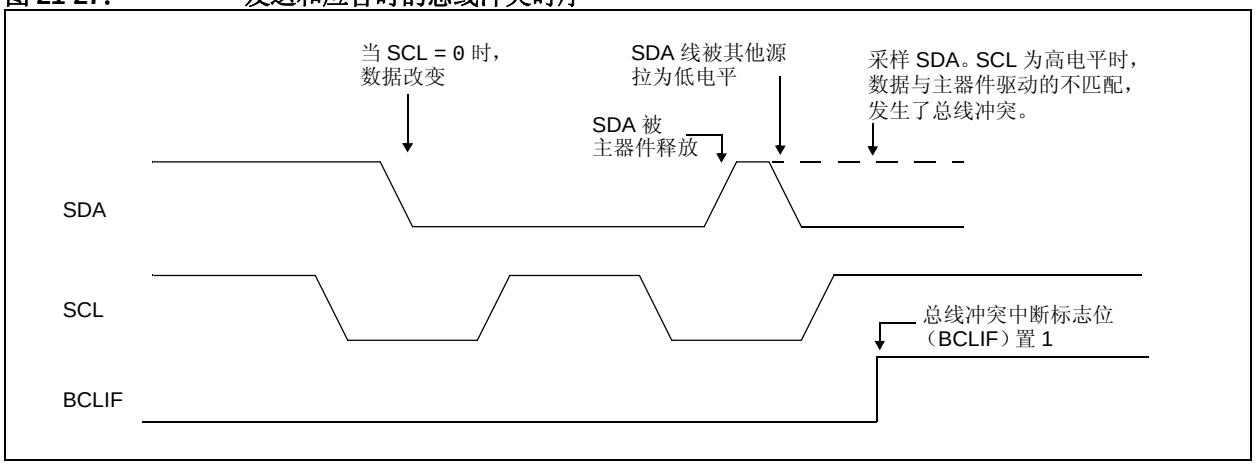
如果在启动、重复启动、停止或应答条件进行过程中发生总线冲突，则条件被中止， SDA 和 SCL 线被拉高，SSPCON2 寄存器中的对应控制位清零。在执行完总线冲突中断服务程序后，如果 I²C 总线空闲，用户可通过发出启动条件恢复通信。

主器件将继续监视 SDA 和 SCL 引脚。一旦出现停止条件， SSPIF 位将被置 1。

发生总线冲突时无论发送的进度如何，写入 SSPBUF 都会从第一个数据位开始发送数据。

在多主器件模式下，通过在检测到启动条件和停止条件时产生中断可以确定总线何时空闲。当 SSPSTAT 寄存器中的 P 位置 1 时，可以取得 I²C 总线的控制权；或者，总线处于空闲状态， S 位和 P 位都清零。

图 21-27： 发送和应答时的总线冲突时序



21.4.17.1 启动条件期间的总线冲突

启动条件期间，以下事件将导致总线冲突：

a) 在启动条件开始时， SDA 或 SCL 被采样为低电平 （图 21-28）。

b) SDA被拉低之前，SCL采样为低电平（图21-29）。

在启动条件期间， SDA 和 SCL 引脚都会被监视。

如果 SDA 引脚已经是低电平，或 SCL 引脚已经是低电平，则：

- 中止启动条件，
- BCLIF 标志位置 1，并且
- MSSP 模块复位为无效状态 （图 21-28）

启动条件从 SDA 和 SCL 引脚被拉高开始。当 SDA 引脚采样为高电平时，波特率发生器装入 SSPADD<6:0> 的内容并递减计数至 0。如果在 SDA 为高电平时，SCL 引脚采样为低电平，则发生总线冲突，因为这表示另一个主器件在启动条件期间试图驱动一个数据 1。

如果 SDA 引脚在该计数周期内采样为低电平，则 BRG 复位，同时 SDA 线保持原值（图 21-30）。但是，如果 SDA 引脚采样为 1，则在 BRG 计数结束时该引脚将被置为低电平。接着，波特率发生器被重载并递减计数至 0。在此期间，如果 SCL 引脚采样到 0，则不会发生总线冲突。在BRG计数结束时，SCL引脚被拉为低电平。

注： 在启动条件期间不太可能发生总线冲突，因为两个总线主器件不可能精确地在同一时刻发出启动条件。因此一个主器件将总是先于另一个主器件将 SDA 拉低。但是上述情况不会引起总线冲突，因为两个主器件一定会对启动条件后的第一个地址进行仲裁。如果地址是相同的，必须继续对数据部分、重复启动条件或停止条件进行仲裁。

图 21-28： 启动条件期间的总线冲突 （仅用于 SDA）

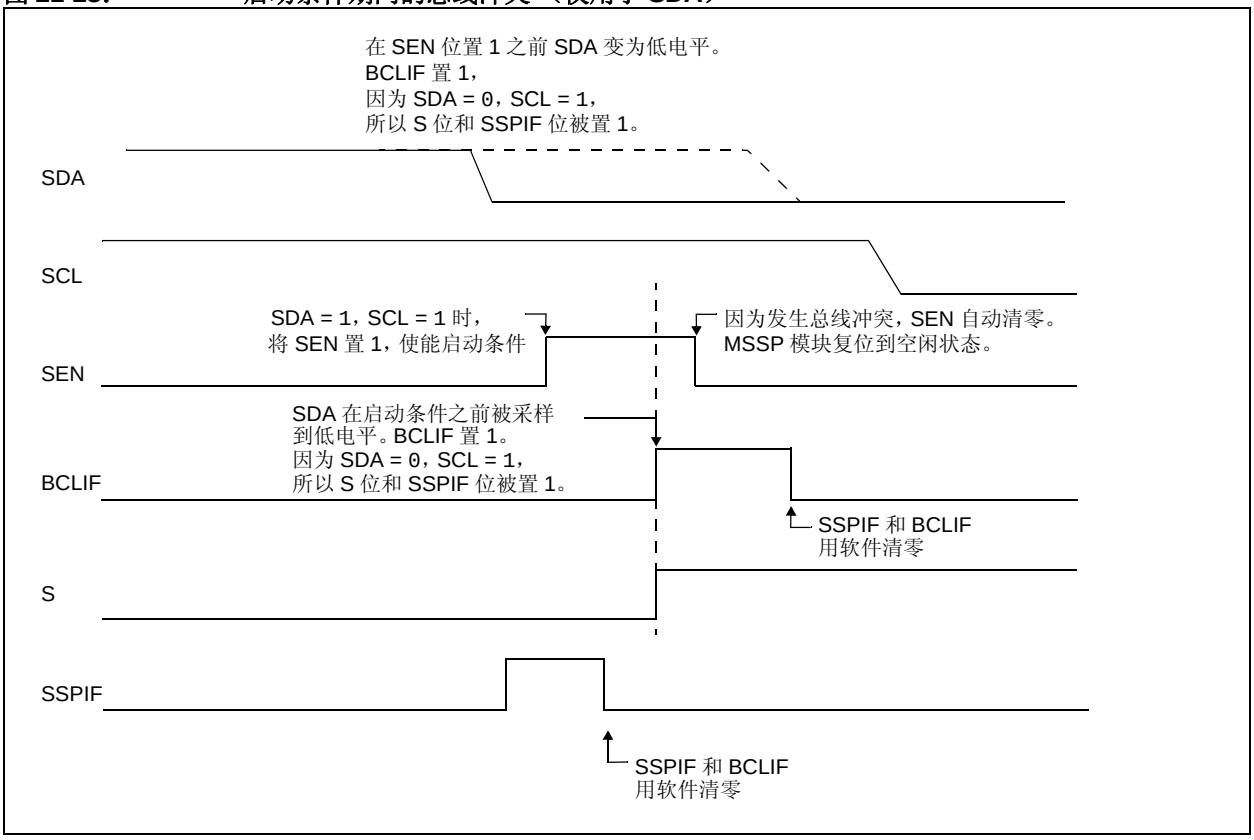


图 21-29：　　启动条件期间的总线冲突 （SCL = 0）

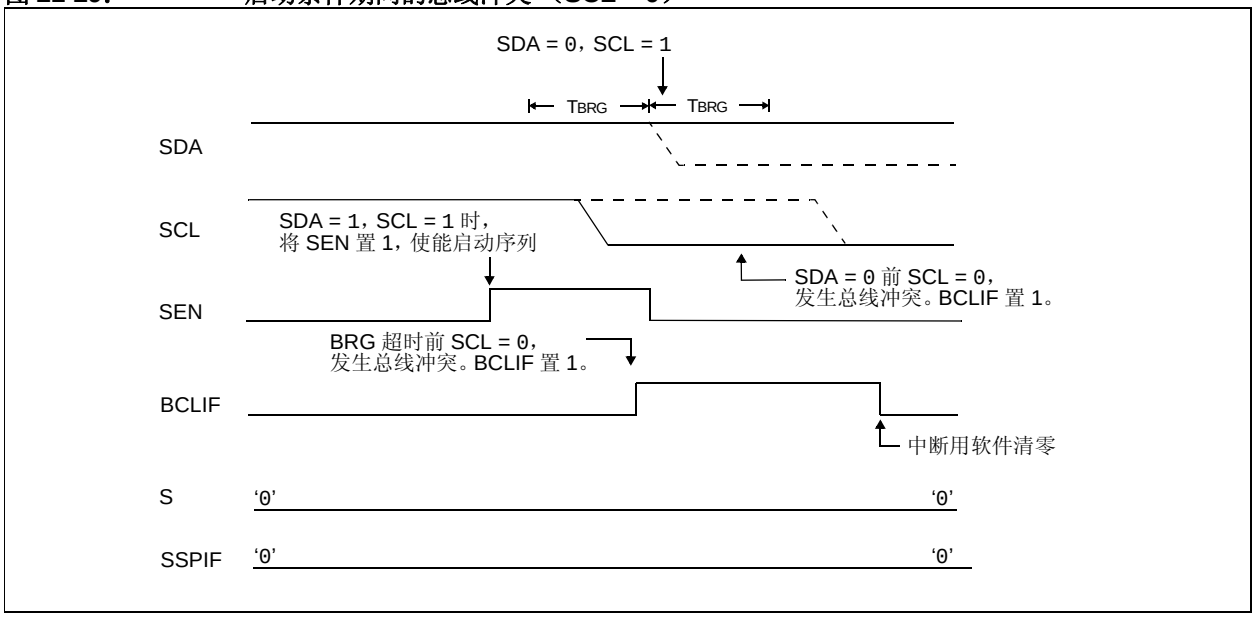
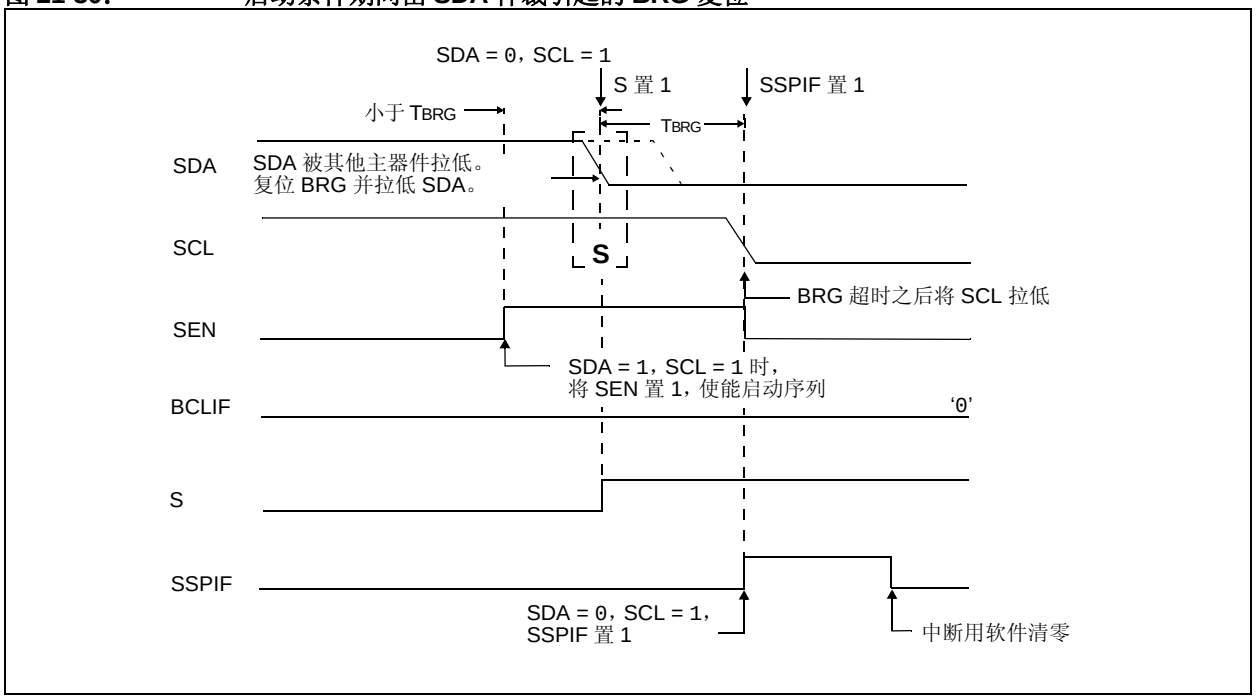


图 21-30：　　启动条件期间由 SDA 仲裁引起的 BRG 复位



21.4.17.2 重复启动条件期间的总线冲突

在以下情况中，重复启动条件期间会发生总线冲突：

a) 在 SCL 由低电平变为高电平期间，在 SDA 上采样到低电平。

b) 在 SDA 被拉为低电平之前， SCL 变为低电平，表示另一个主器件正试图发送一个数据 1。

当用户拉高 SDA 并允许该引脚悬空时， BRG 装入 SSPADD<6:0> 的内容并递减计数至 0。接着 SCL 引脚被拉高，当 SCL 引脚采样到高电平时，对 SDA 引脚进行采样。

如果 SDA 为低电平，则已发生了总线冲突（即，另一个主器件正试图发送一个数据 0，见图 21-31）。如果 SDA 被采样到高电平，则 BRG 被重载并开始计数。如果 SDA 在 BRG 超时之前从高电平变为低电平，则不会发生总线冲突，因为两个主器件不可能精确地在同一时刻将 SDA 拉低。

如果 SCL 在 BRG 超时之前从高电平变为低电平，且 SDA 尚未被拉低，那么将发生总线冲突。在此情况下，另一个主器件在重复启动条件期间正试图发送一个数据 1（见图 21-32）。

如果在 BRG 超时结束时 SCL 和 SDA 都仍然是高电平，则 SDA 引脚被拉低， BRG 被重载并开始计数。在计数结束时，不管 SCL 引脚的状态如何， SCL 引脚都被拉低，重复启动条件结束。

图 21-31： 重复启动条件期间的总线冲突 （情形 1）

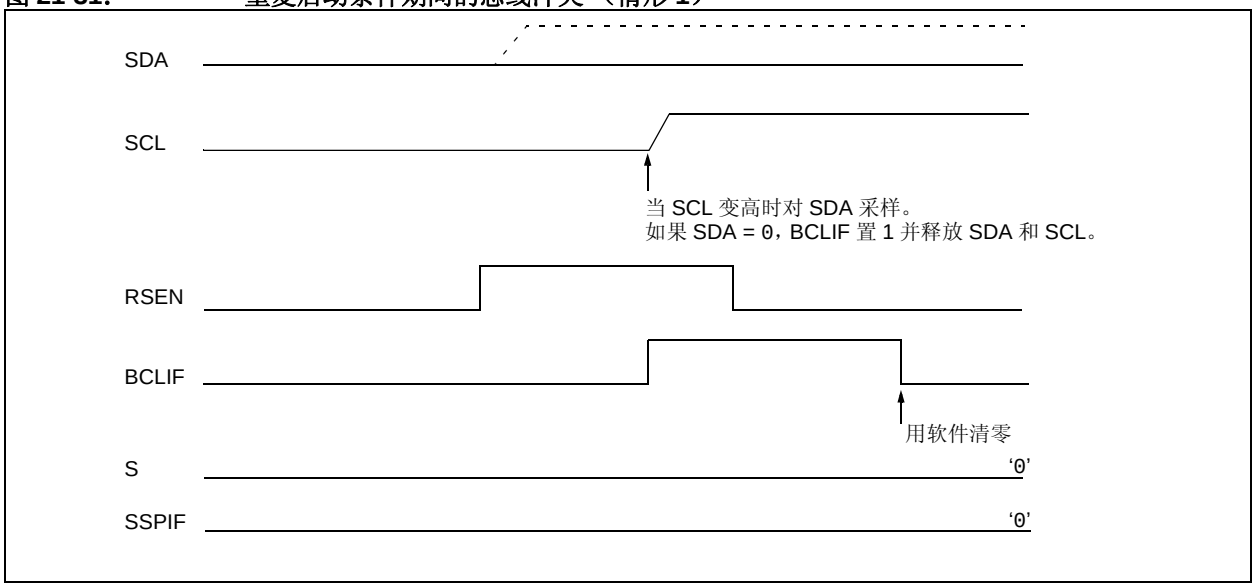
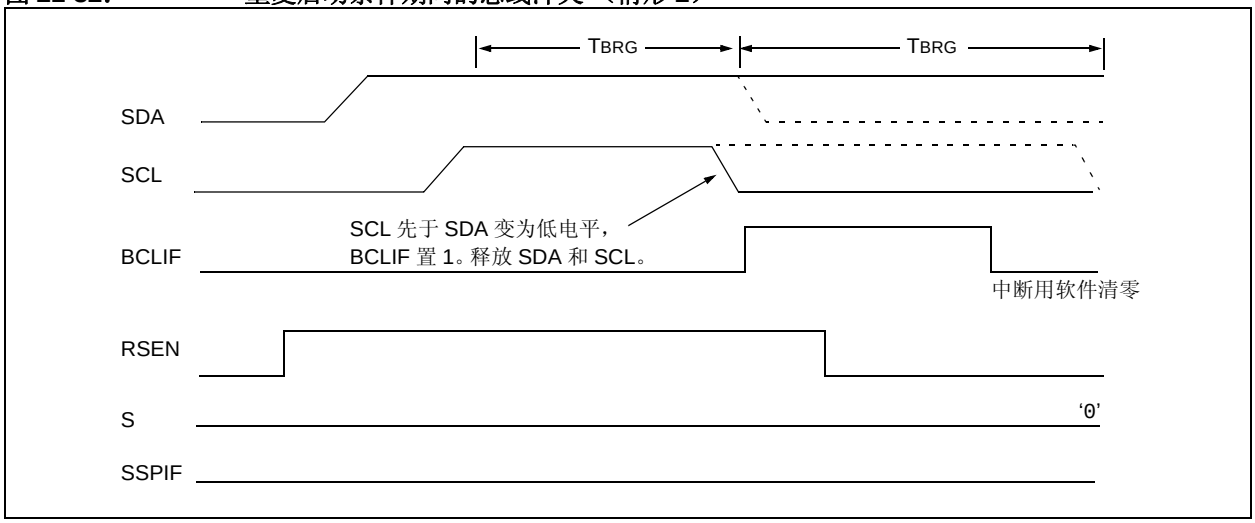


图 21-32： 重复启动条件期间的总线冲突 （情形 2）



21.4.17.3 停止条件期间的总线冲突

以下事件会导致停止条件期间发生总线冲突：

a) SDA已被拉高并允许悬空为高电平之后，SDA在BRG 超时后被采样到低电平。

b) SCL 引脚被拉高之后，SCL 在 SDA 变成高电平之前被采样到低电平。

停止条件从 SDA 被置为低电平开始。当 SDA 采样为低电平时， SCL 引脚被允许悬空。当 SDA 被采样到高电平（时钟仲裁）时，波特率发生器装入 SSPADD<6:0> 的内容并递减计数至 0。BRG 超时后，SDA 被采样。如果 SDA 采样为低电平，则已发生总线冲突。这是因为另一个主器件正试图发送一个数据 0（图 21-33）。如果 SCL 引脚在允许 SDA 悬空为高电平前被采样到低电平，也会发生总线冲突。这是另一个主器件正试图发送一个数据 0 的另外一种情况（图 21-34）。

图 21-33： 停止条件期间的总线冲突（情形 1）

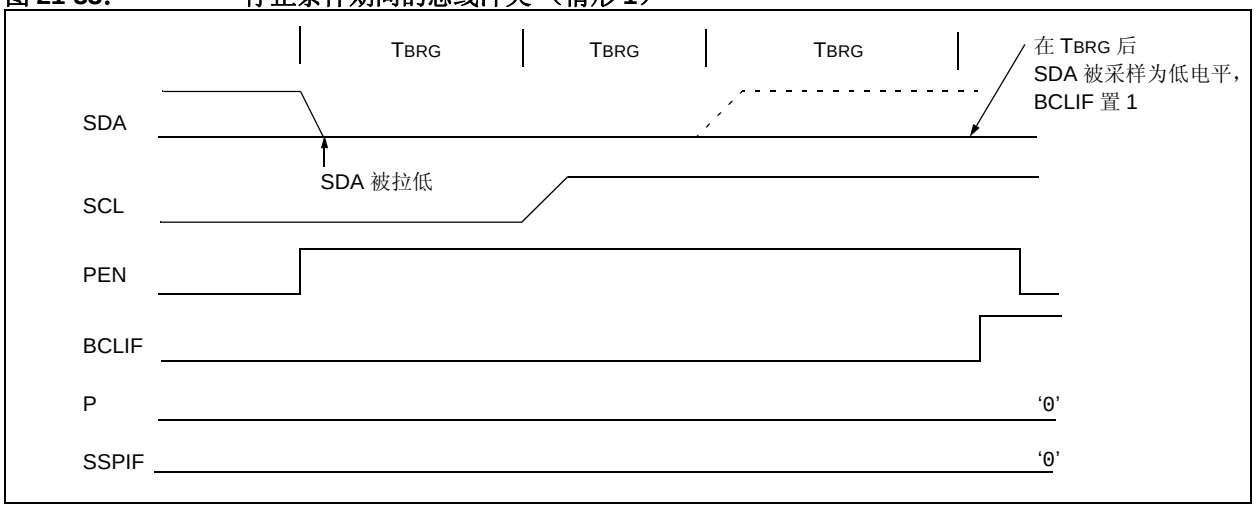


图 21-34： 停止条件期间的总线冲突（情形 2）

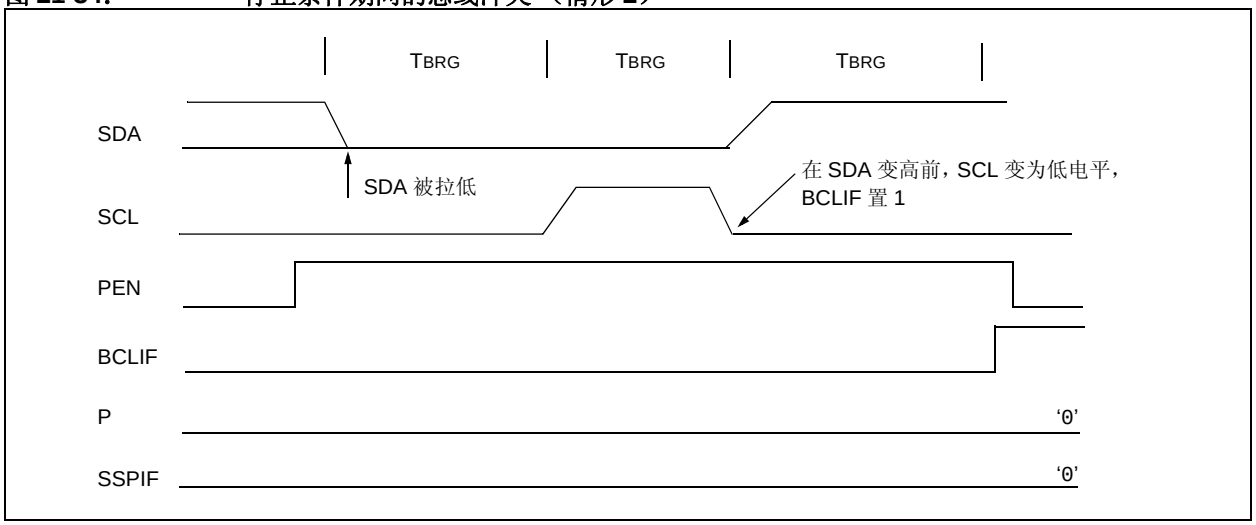


表 21-4：　　与 I²C™ 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR2	OSCFIF	—	—	—	BCLIF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	—	—	BCLIE	HLVDIE	TMR3IE	TMR3GIE
IPR2	OSCFIP	—	—	—	BCLIP	HLVDIP	TMR3IP	TMR3GIP
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
SSPBUF	MSSP 接收缓冲 / 发送寄存器							
SSPADD	MSSP 地址寄存器 （I²C™ 从模式）， MSSP 波特率重载寄存器 （I²C 主模式）							
SSPMSK(1)	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
	GCEN	ACKSTAT	ADMSK5(2)	ADMSK4(2)	ADMSK3(2)	ADMSK2(2)	ADMSK1(2)	SEN
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

图注：　— = 未实现，读为 0。 I²C™ 模式下的 MSSP 模块不使用阴影单元。

注　1： SSPMSK 与 SSPADD 共用 SFR 空间中的相同地址，但是只能在 7 位掩码模式下的某些 I²C™ 从工作模式中访问。更多详细信息，请参见**第 21.4.3.4 节 “7 位地址掩码模式 ”**。

2： 仅在 I²C 从模式操作中使用的备用位定义。

注：

22.0 增强型通用同步 / 异步收发器（EUSART）

增强型通用同步 / 异步收发器（Enhanced Universal Synchronous Asynchronous Receiver Transmitter，EUSART）模块是两个串行 I/O 模块之一。（通常，EUSART 也称为 “ 串行通信接口 ” 或 SCI。）

可以将 EUSART 配置为能与 CRT 终端和个人计算机等外设通信的全双工异步系统。也可以将它配置为能与 A/D 或 D/A 集成电路和串行 EEPROM 等外设通信的半双工同步系统。

增强型 USART 模块还实现了其他特性，包括自动波特率检测和校准、接收到同步间隔字符时自动唤醒和 12 位间隔字符发送。因为具有这些特性，使其成为局域互联网总线 （LIN/J2602 总线）系统的理想选择。

PIC18F66K80 系列的所有产品都配备了两个独立的 EUSART 模块，称为 EUSART1 和 EUSART2。可将这两个模块配置为以下几种工作模式：

- 异步模式 （全双工）：
  - 接收到字符时自动唤醒
  - 自动波特率校准
  - 12 位间隔字符发送
- 同步 —— 主模式 （半双工），时钟极性可选
- 同步 —— 从模式 （半双工），时钟极性可选

EUSART1和EUSART2的引脚与以下端口的功能复用，具体取决于器件的引脚数。请参见表 22-1。

表 22-1： 配置 EUSART 引脚 (1)

引脚数	USART1		USART2	
	端口	引脚	端口	引脚
28 引脚	PORTB	RB6/PGC/TX2/CK2/KBI2 和 RB7/PGD/T3G/RX2/DT2/KBI3	PORTC	RC6/TX1/CK1 和 RC7/RX1/DT1
40/44 引脚	PORTC	RC6/TX1/CK1 和 RC7/RX1/DT1	PORTD	RD6/TX2/CK2/P1C/PSP6 和 RD7/RX2/DT2/P1D/PSP7
64 引脚	PORTE	RE7/TX2/CK2 和 RE6/RX2/DT2	PORTG	RG3/TX1/CK1 和 RG0/RX1/DT1

注 1： EUSART 控制根据需要会自动将引脚从输入重新配置为输出。

要将这些引脚配置为 EUSART：

- 对于 EUSART1：
  - SPEN （RCSTA1<7>）必须置 1 （= 1）
  - TRISx<x> 必须置 1 （= 1）
  - 对于异步和同步主模式， TRISx<x> 必须清零 （= 0）
  - 对于同步从模式， TRISx<x> 必须置 1 （= 1）
- 对于 EUSART2：
  - SPEN （RCSTA2<7>）必须置 1 （= 1）
  - TRISx<x> 必须置 1 （= 1）
  - 对于异步和同步主模式， TRISx<x> 必须清零 （= 0）
  - 对于同步从模式， TRISx<x> 必须置 1 （= 1）

22.1 EUSART 控制寄存器

每个增强型USART模块的操作由以下3个寄存器控制：

- 发送状态和控制 （TXSTAx）
- 接收状态和控制 （RCSTAx）
- 波特率控制 （BAUDCONx）

这些寄存器将在寄存器22-1、寄存器22-2和寄存器22-3中分别详细介绍。

注： 在本章中，与特定 EUSART 模块相关的寄存器名称和位名称的引用一般都采用以 “x” 代替特定模块编号的方式。因此，“RCSTAx” 可能指 EUSART1 或 EUSART2 的接收状态寄存器。

寄存器 22-1： TXSTAx：发送状态和控制寄存器

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R-x	R/W-x
CSRC	TX9	TXEN(1)	SYNC	SENDB	BRGH	TRMT	TX9D
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 **CSRC：**时钟源选择位

异步模式：
无关位。

同步模式：
1 = 主模式 （时钟由内部 BRG 产生）
0 = 从模式 （时钟来自外部时钟源）

bit 6 **TX9：**9 位发送使能位
1 = 选择 9 位发送
0 = 选择 8 位发送

bit 5 **TXEN：**发送使能位 (1)
1 = 使能发送
0 = 禁止发送

bit 4 **SYNC：**EUSART 模式选择位
1 = 同步模式
0 = 异步模式

bit 3 **SENDB：**发送间隔字符位

异步模式：
1 = 在下一次发送时发送同步间隔字符 （完成时由硬件清零）
0 = 同步间隔字符发送完成

同步模式：
无关位。

bit 2 **BRGH：**高波特率选择位

异步模式：
1 = 高速
0 = 低速

同步模式：
在此模式下未使用。

bit 1 **TRMT：**发送移位寄存器状态位
1 = TSR 空
0 = TSR 满

bit 0 **TX9D：**发送数据的第 9 位
可以是地址 / 数据位或奇偶校验位。

注 1： 在同步模式下， SREN/CREN 可改写 TXEN。

寄存器 22-2： RCSTAx：接收状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
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 **SPEN**：串口使能位
1 = 使能串口 （将 RXx/DTx 和 TXx/CKx 引脚配置为串口引脚）
0 = 禁止串口 （保持在复位状态）

bit 6 **RX9**：9 位接收使能位
1 = 选择 9 位接收
0 = 选择 8 位接收

bit 5 **SREN**：单字节接收使能位
异步模式：
无关位。
同步主模式：
1 = 使能单字节接收
0 = 禁止单字节接收
该位在接收完成后清零。
同步从模式：
无关位。

bit 4 **CREN**：连续接收使能位
异步模式：
1 = 使能接收器
0 = 禁止接收器
同步模式：
1 = 使能连续接收，直到使能位 CREN 清零 （CREN 的优先级高于 SREN）
0 = 禁止连续接收

bit 3 **ADDEN**：地址检测使能位
9 位异步模式 （RX9 = 1）：
1 = 当 RSR<8> 置 1 时，使能地址检测，允许中断并装入接收缓冲区
0 = 禁止地址检测、接收所有字节并且第 9 位可作为奇偶校验位
9 位异步模式 （RX9 = 0）：
无关位。

bit 2 **FERR**：帧错误位
1 = 帧错误 （可以通过读 RCREGx 寄存器清零该位并接收下一个有效字节）
0 = 无帧错误

bit 1 **OERR**：溢出错误位
1 = 溢出错误 （可以通过清零 CREN 位来清零该位）
0 = 无溢出错误

bit 0 **RX9D**：接收数据的第 9 位
该位可以是地址 / 数据位或奇偶校验位，并且必须由用户固件计算得到。

寄存器 22-3： BAUDCONx：波特率控制寄存器

R/W-0	R-1	R/W-x	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
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 **ABDOVF**：自动波特率采集计满返回状态位
1 = 在自动波特率检测模式下发生了 BRG 计满返回 （必须用软件清零）
0 = 没有发生 BRG 计满返回

bit 6 **RCIDL**：接收操作空闲状态位
1 = 接收操作处于空闲状态
0 = 接收操作处于活动状态

bit 5 **RXDTP**：接收数据极性选择位 （仅限异步模式）
异步模式：
1 = 接收数据 （RXx）反相
0 = 接收数据 （RXx）未反相

bit 4 **TXCKP**：时钟和数据极性选择位
异步模式：
1 = 发送 （TXx）的空闲状态为低电平
0 = 发送 （TXx）的空闲状态为高电平
同步模式：
1 = 时钟 （CKx）的空闲状态为高电平
0 = 时钟 （CKx）的空闲状态为低电平

bit 3 **BRG16**：16 位波特率寄存器使能位
1 = 16 位波特率发生器 ——SPBRGHx 和 SPBRGx
0 = 8 位波特率发生器 —— 仅限 SPBRGx （兼容模式）， SPBRGHx 值被忽略

bit 2 **未实现**：读为 0

bit 1 **WUE**：唤醒使能位
异步模式：
1 = EUSART 将继续采样 RXx 引脚：在出现下降沿时产生中断；在之后的上升沿由硬件清零该位
0 = 未监视 RXx 引脚或检测到了上升沿
同步模式：
在此模式下未使用。

bit 0 **ABDEN**：自动波特率检测使能位
异步模式：
1 = 使能对下一个字符的波特率测量：需要接收同步字段 （55h）；完成时由硬件清零
0 = 禁止波特率测量或测量已完成
同步模式：
在此模式下未使用。

22.2 波特率发生器 （BRG）

BRG 是一个专用的 8 位或 16 位发生器，支持 EUSART 的异步和同步模式。默认情况下，BRG 工作在 8 位模式下；将 BRG16 位 （BAUDCONx<3>）置 1 可选择 16 位模式。

SPBRGHx:SPBRGx 寄存器对控制自由运行定时器的周期。在异步模式下，BRGH（TXSTAx<2>）和BRG16（BAUDCONx<3>）也用于控制波特率。在同步模式下，BRGH 位被忽略。表 22-2 提供了不同 EUSART 模式下波特率的计算公式，仅适用于主模式（内部生成的时钟）。

给出目标波特率和 Fosc 值，就可以使用表 22-2 中的公式计算 SPBRGHx:SPBRGx 寄存器的最近似整数值。这样就可以确定波特率误差。例 22-1 给出了一个计算示例。表 22-3 给出了不同异步模式下典型的波特率和误差值。使用高波特率（BRGH = 1）或 16 位 BRG 有助于降低波特率误差，或者在快速振荡器频率下取得较缓慢的波特率。

将新值写入 SPBRGHx:SPBRGx 寄存器会导致 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。

22.2.1 在功耗管理模式下的操作

器件时钟用于产生所需的波特率。当进入一种功耗管理模式时，新时钟源可能会工作在一个不同的频率下。这可能需要调整 SPBRGHx:SPBRGx 寄存器对中的值。

22.2.2 采样

择多检测电路对 RXx 引脚（RC7/CANRX/RX1/DT1 或 RB7/PGD/T3G/RX2/DT2/KBI3）上的数据采样三次，以判断 RXx 引脚上出现的是高电平还是低电平。

表 22-2： 波特率公式

配置位			BRG/EUSART 模式	波特率公式
SYNC	BRG16	BRGH		
0	0	0	8 位 / 异步	Fosc/[64 (n + 1)]
0	0	1	8 位 / 异步	Fosc/[16 (n + 1)]
0	1	0	16 位 / 异步	
0	1	1	16 位 / 异步	Fosc/[4 (n + 1)]
1	0	x	8 位 / 同步	
1	1	x	16 位 / 同步	

图注： x = 无关位， n = SPBRGHx:SPBRGx 寄存器对的值

例 22-1：　　计算波特率误差

器件工作在 FOSC = 16 MHz，目标波特率 = 9600 bps，异步模式， 8 位 BRG：

目标波特率 = FOSC/(64 ([SPBRGHx:SPBRGx] + 1))

求解 SPBRGHx:SPBRGx：

X = ((FOSC/ 目标波特率)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25

计算波特率 = 16000000/(64 (25 + 1))
= 9615

误差 = (计算波特率 – 目标波特率)/ 目标波特率
= (9615 – 9600)/9600 = 0.16%

表 22-3：　　与波特率发生器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 波特率发生器寄存器的高字节							
SPBRG1	EUSART1 波特率发生器寄存器的低字节							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 波特率发生器寄存器的高字节							
SPBRG2	EUSART2 波特率发生器寄存器的低字节							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD

图注：　— = 未实现，读为 0。 BRG 不使用阴影单元。

表 22-4： 异步模式下的波特率

波特率（K）	SYNC = 0， BRGH = 0， BRG16 = 0											
	Fosc = 64.000 MHz			Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz		
	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	1.221	1.73	255	1.202	0.16	129
2.4	—	—	—	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64
9.6	9.615	0.16	103	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15
19.2	19.231	0.16	51	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7
57.6	58.824	2.13	16	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2
115.2	111.111	-3.55	8	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1

波特率（K）	SYNC = 0， BRGH = 0， BRG16 = 0											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）
0.3	—	—	—	0.300	0.16	207	0.300	-0.16	103	0.300	-0.16	51
1.2	1.201	-0.16	103	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.403	-0.16	51	2.404	0.16	25	2.403	-0.16	12	—	—	—
9.6	9.615	-0.16	12	8.929	-6.99	6	—	—	—	—	—	—
19.2	—	—	—	20.833	8.51	2	—	—	—	—	—	—
57.6	—	—	—	62.500	8.51	0	—	—	—	—	—	—
115.2	—	—	—	62.500	-45.75	0	—	—	—	—	—	—

波特率（K）	SYNC = 0， BRGH = 1， BRG16 = 0											
	Fosc = 64.000 MHz			Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz		
	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	—	—	—	2.441	1.73	255
9.6	—	—	—	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64
19.2	19.417	1.13	207	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31
57.6	59.701	3.65	68	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10
115.2	121.212	5.22	34	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4

波特率（K）	SYNC = 0， BRGH = 1， BRG16 = 0											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）
0.3	—	—	—	—	—	—	—	—	—	0.300	-0.16	207
1.2	—	—	—	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.403	-0.16	207	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	-0.16	51	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.230	-0.16	25	19.231	0.16	12	—	—	—	—	—	—
57.6	55.555	3.55	8	62.500	8.51	3	—	—	—	—	—	—
115.2	—	—	—	125.000	8.51	1	—	—	—	—	—	—

表 22-4： 异步模式下的波特率 （续）

波特率（K）	SYNC = 0， BRGH = 0， BRG16 = 1											
	Fosc = 64.000 MHz			Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz		
	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）
0.3	0.300	0.00	13332	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082
1.2	1.200	0.00	3332	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520
2.4	2.400	0.00	1666	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259
9.6	9.592	-0.08	416	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64
19.2	19.417	1.13	207	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31
57.6	59.701	3.65	68	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10
115.2	121.212	5.22	34	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4

波特率（K）	SYNC = 0， BRGH = 0， BRG16 = 1											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）
0.3	0.300	-0.04	1665	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.201	-0.16	415	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.403	-0.16	207	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	-0.16	51	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.230	-0.16	25	19.231	0.16	12	—	—	—	—	—	—
57.6	55.555	3.55	8	62.500	8.51	3	—	—	—	—	—	—
115.2	—	—	—	125.000	8.51	1	—	—	—	—	—	—

波特率（K）	SYNC = 0， BRGH = 1， BRG16 = 1 或 SYNC = 1， BRG16 = 1											
	Fosc = 64.000 MHz			Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz		
	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）
0.3	0.300	0.00	53332	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332
1.2	1.200	0.00	13332	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082
2.4	2.400	0.00	6666	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040
9.6	9.598	-0.02	1666	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259
19.2	19.208	0.04	832	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129
57.6	57.348	-0.44	278	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42
115.2	115.108	-0.08	138	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21

波特率（K）	SYNC = 0， BRGH = 1， BRG16 = 1 或 SYNC = 1， BRG16 = 1											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）	实际波特率（K）	% 误差	SPBRG 值（十进制）
0.3	0.300	-0.01	6665	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	-0.04	1665	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.400	-0.04	832	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	-0.16	207	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.230	-0.16	103	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	57.142	0.79	34	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	117.647	-2.12	16	111.111	-3.55	8	—	—	—	—	—	—

22.2.3 自动波特率检测

增强型 USART 模块支持波特率自动检测和校准。该功能仅在异步模式下、 WUE 位清零时有效。

只要接收到启动位并且 ABDEN 位已置 1，就会开始自动波特率测量序列（图 22-1）。波特率计算采用自平均的方式。

在自动波特率检测（ABD）模式下，BRG 的时钟反相。BRG 并不为进入的 RXx 信号提供时钟信号，而是由 RXx 信号为 BRG 定时。在 ABD 模式下，内部波特率发生器用作计数器计算进入的串行字节流的位周期。

一旦 ABDEN 位置 1，状态机就会清零 BRG 并寻找启动位。要计算正确的比特率，自动波特率检测必须接收到值为 55h 的字节（ASCII 码为 “U”，也是 LIN/J2602 总线的同步字符）。为了尽量减少输入信号不对称造成的影响，测量时间段内要包含一个高位和一个低位时间。在启动位后， SPBRGx 使用预先选择的时钟源在 RXx 的第一个上升沿开始递增计数。在 RXx 引脚传输了 8 个位，或在检测到第 5 个上升沿后，会将相应 BRG 周期内的累加值保存在 SPBRGHx:SPBRGx 寄存器对中。当第 5 个时钟边沿出现时（应与停止位对应），ABDEN 位会自动清零。

如果发生了 BRG 计满返回（从 FFFFh 到 0000h 的溢出），会在 ABDOVF 状态位（BAUDCONx<7>）有所反映。该位可在 BRG 计满返回时由硬件置 1，也可以由用户通过软件置 1 或清零。在发生计满返回事件后，ABD 模式继续有效，ABDEN 位保持置 1（图 22-2）。

在校准波特率周期时，BRG 寄存器时钟频率为预配置时钟频率的 1/8。BRG 时钟将由 BRG16 和 BRGH 位配置。必须将 BRG16 位置 1，以将 SPBRG1 和 SPBRGH1 用作 16 位计数器。用户通过检查 SPBRGHx 寄存器中的值是否为 00h，可以验证 8 位模式下是否发生了进位。表 22-5 所示为 BRG 计数器的时钟速率。

当发生 ABD 序列时，EUSART 状态机保持在空闲状态。一旦在 RXx 上检测到第 5 个上升沿，中断标志位 RCxIF 就会置 1。需要读取 RCREGx 中的值以清零中断标志位 RCxIF。应丢弃 RCREGx 的内容。

注 1： 如果 WUE 位与 ABDEN 位同时置 1，自动波特率检测会在间隔字符*之后*的字节开始。

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。由于位错误率的原因，某些振荡器频率和 EUSART 波特率的组合是无法实现的。使用自动波特率检测功能时，必须综合考虑系统总的时序和通信波特率。

3： 要最大限度地提高波特率范围，如果使用了该功能，则建议将 BRG16 位（BAUDCONx<3>）置 1。

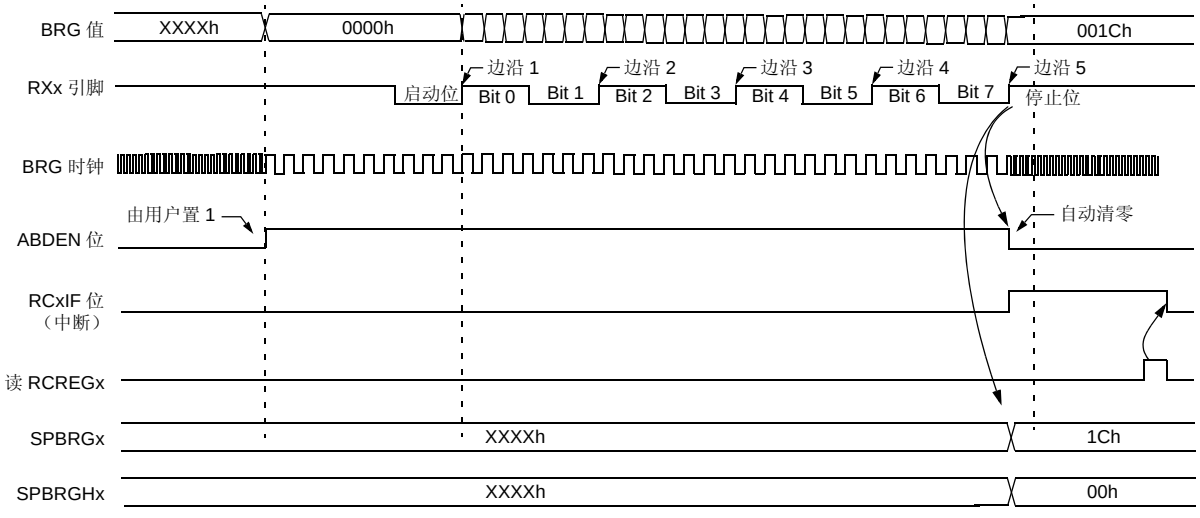
表 22-5： BRG 计数器时钟速率

BRG16	BRGH	BRG 计数器时钟
0	0	Fosc/512
0	1	Fosc/128
1	0	Fosc/128
1	1	Fosc/32

22.2.3.1 ABD 和 EUSART 发送

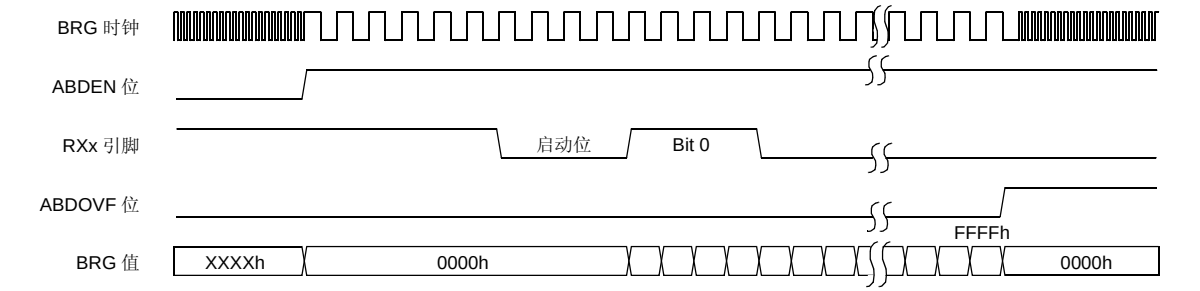
由于 ABD 采集期间 BRG 时钟是反相的，因此在 ABD 期间不能使用 EUSART 发送器。这意味着只要 ABDEN 位置 1，就不能写入 TXREGx。用户还应确保在发送序列期间 ABDEN 不能为置 1 状态，否则可能会导致无法预料的 EUSART 操作。

图 22-1：　　自动波特率计算



注： ABD 序列要求 EUSART 模块配置为异步模式并且 WUE = 0。

图 22-2：　　BRG 溢出序列



22.3 EUSART 异步模式

通过将 SYNC 位（TXSTAx<4>）清零可选择异步工作模式。在该模式下，EUSART 使用标准不归零码（Non-Return-to-Zero，NRZ）格式（1 个启动位、8 个或 9 个数据位和 1 个停止位）。最常见的数据格式为 8 位。片上专用 8 位 /16 位波特率发生器可用于从振荡器产生标准波特率频率。

EUSART 先发送和接收 LSb。EUSART 的发送器和接收器在功能上是相互独立的，但使用相同的数据格式和波特率。根据 BRGH 和 BRG16 位（TXSTAx<2> 和 BAUDCONx<3>）的设置情况，波特率发生器会产生一个 16 倍或 64 倍数据移位速率的时钟信号。硬件不支持奇偶校验，但可通过软件实现并作为第 9 个数据存储。

当工作在异步模式下时，EUSART 模块包括以下重要组成部分：

- 波特率发生器
- 采样电路
- 异步发送器
- 异步接收器
- 接收到同步间隔字符时自动唤醒
- 12 位间隔字符发送
- 自动波特率检测

22.3.1 EUSART 异步发送器

图 22-3 给出了 EUSART 发送器框图。发送器的核心是发送（串行）移位寄存器（Transmit Shift Register，TSR）。移位寄存器从读 / 写发送缓冲寄存器 TXREGx 中获取数据。用软件将数据装入 TXREGx 寄存器。在前一次装入数据的停止位发送前，不会向 TSR 寄存器装入新数据。一旦停止位发送完毕，TXREGx 寄存器中的新数据（如果有）就会被装入 TSR。

一旦 TXREGx 寄存器向 TSR 寄存器传输了数据（在 1 个 TCY 内发生），TXREGx 寄存器就为空，同时 TXxIF 标志位置 1。可以通过将中断允许位 TXxIE 置 1 或清零来允许或禁止该中断。不管 TXxIE 的状态如何，TXxIF 都将置 1；不能用软件清零。TXxIF 也不会在 TXREGx 装入新数据时立即被清零，而是在装入指令后的第二个指令周期被清零。因此在 TXREGx 装入新数据后立即查询 TXxIF，将返回无效结果。

TXxIF 指示 TXREGx 寄存器的状态，而另一个位 TRMT（TXSTAx<1>）则指示 TSR 寄存器的状态。TRMT 是只读位，它在 TSR 寄存器为空时置 1。TRMT 位与任何中断逻辑均无关联，因此要确定 TSR 寄存器是否为空，用户只能对该位进行查询。

注 1： TSR 寄存器不映射到数据存储器中，因此用户无法使用。

2： 当使能位TXEN置1时，标志位TXxIF置1。

设置异步发送操作的步骤如下：

1. 对SPBRGHx:SPBRGx寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，获得所需的波特率。
2. 通过清零 SYNC 位并将 SPEN 位置 1，使能异步串口。
3. 如果需要中断，将中断允许位 TXxIE 置 1。
4. 如果需要 9 位发送，将发送位 TX9 置 1。可以作为地址 / 数据位使用。
5. 通过将TXEN位置1使能发送，这也将导致TXxIF位置 1。
6. 如果选择了 9 位发送，应将第 9 位装入 TX9D 位。
7. 将数据装入 TXREGx 寄存器（启动发送）。
8. 如果使用中断，应确保 GIE 和 PEIE 位（INTCON<7:6>）置 1。

图 22-3： EUSART 发送框图

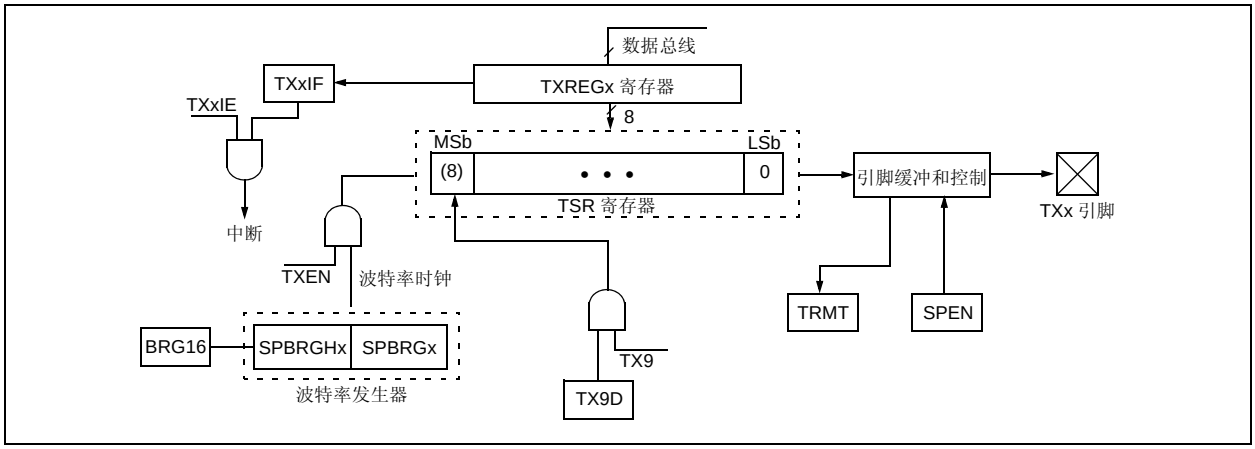


图 22-4：　　异步发送

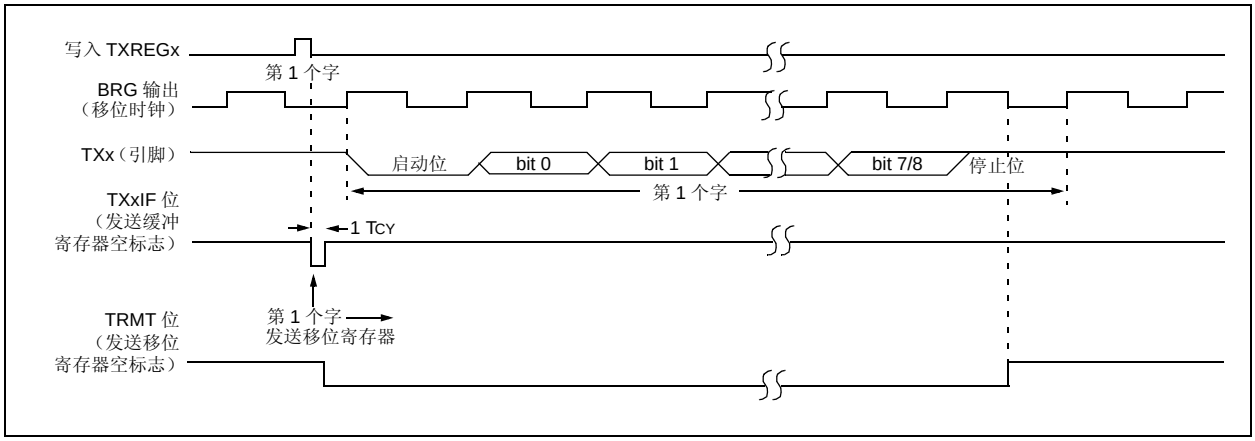
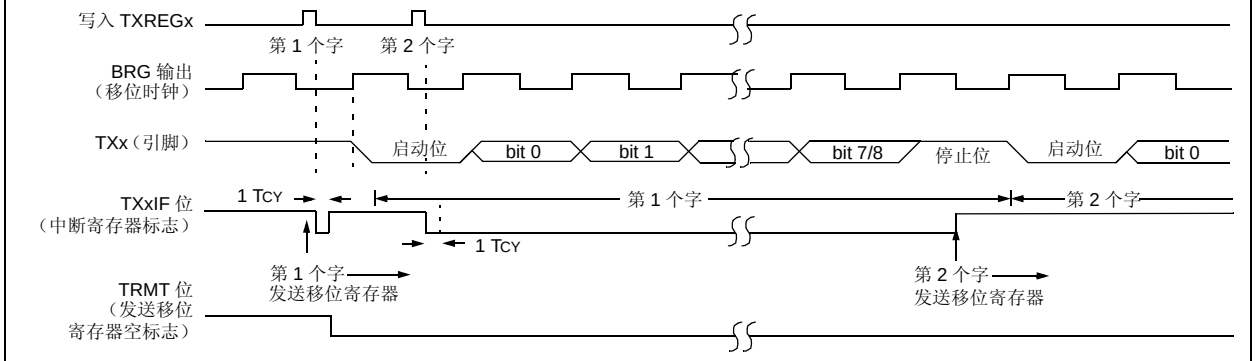


图 22-5：　　异步发送（背对背）



注：　此时序图表示的是两次连续的发送。

表 22-6：　　与异步发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
TXREG1	EUSART1 发送寄存器							
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 波特率发生器寄存器的高字节							
SPBRG1	EUSART1 波特率发生器寄存器的低字节							
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
TXREG2	EUSART2 发送寄存器							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 波特率发生器寄存器的高字节							
SPBRG2	EUSART2 波特率发生器寄存器的低字节							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

图注：　　— = 未实现位，读为 0。异步发送不使用阴影单元。

22.3.2 EUSART 异步接收器

图 22-6 给出了接收器框图。在 RXx 引脚上接收数据，并驱动数据恢复电路。数据恢复模块实际上是一个工作频率为 16 倍波特率的高速移位器，而主接收串行移位器的工作频率等于比特率或 FOSC。该模式通常用于 RS-232 系统中。

设置异步接收操作的步骤如下：

1. 对SPBRGHx:SPBRGx寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，获得所需的波特率。
2. 通过清零 SYNC 位并将 SPEN 位置 1，使能异步串口。
3. 如果需要中断，将中断允许位 RCxIE 置 1。
4. 如果需要接收 9 位数据，将 RX9 位置 1。
5. 通过将 CREN 位置 1 使能接收。
6. 接收完成时标志位 RCxIF 将被置 1，此时如果中断允许位 RCxIE 已置 1，还将产生一个中断。
7. 读取 RCSTAx 寄存器取得第 9 位数据 （如果已使能），并确定接收时是否发生了错误。
8. 读 RCREGx 寄存器来读取接收到的 8 位数据。
9. 如果发生错误，将使能位 CREN 清零以清除错误。
10. 如果使用中断，应确保 GIE 和 PEIE 位（INTCON<7:6>）置 1。

22.3.3 设置带有地址检测功能的 9 位模式

该模式通常用于 RS-485 系统中。设置使能地址检测的异步接收的步骤如下：

1. 对 SPBRGHx:SPBRGx 寄存器进行初始化，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，获得所需的波特率。
2. 通过清零 SYNC 位并将 SPEN 位置 1，使能异步串口。
3. 如果需要中断，将 RCEN 位置 1 并用 RCxIP 位选择所需的优先级。
4. 将 RX9 位置 1 使能 9 位接收。
5. 将 ADDEN 位置 1 使能地址检测。
6. 将 CREN 位置 1 使能接收。
7. 接收完成时 RCxIF 位将被置 1。此时如果 RCxIE 和 GIE 位已置 1，还将响应中断。
8. 读 RCSTAx 寄存器判断在接收时是否发生了错误，同时读取第 9 位数据 （如果适用）。
9. 读 RCREGx 判断是否正在对器件进行寻址。
10. 如果发生错误，将 CREN 位清零。
11. 如果已寻址到器件，将 ADDEN 位清零以允许所有接收到的数据被送入接收缓冲区并发出中断信号给 CPU。

图 22-6： EUSART 接收框图

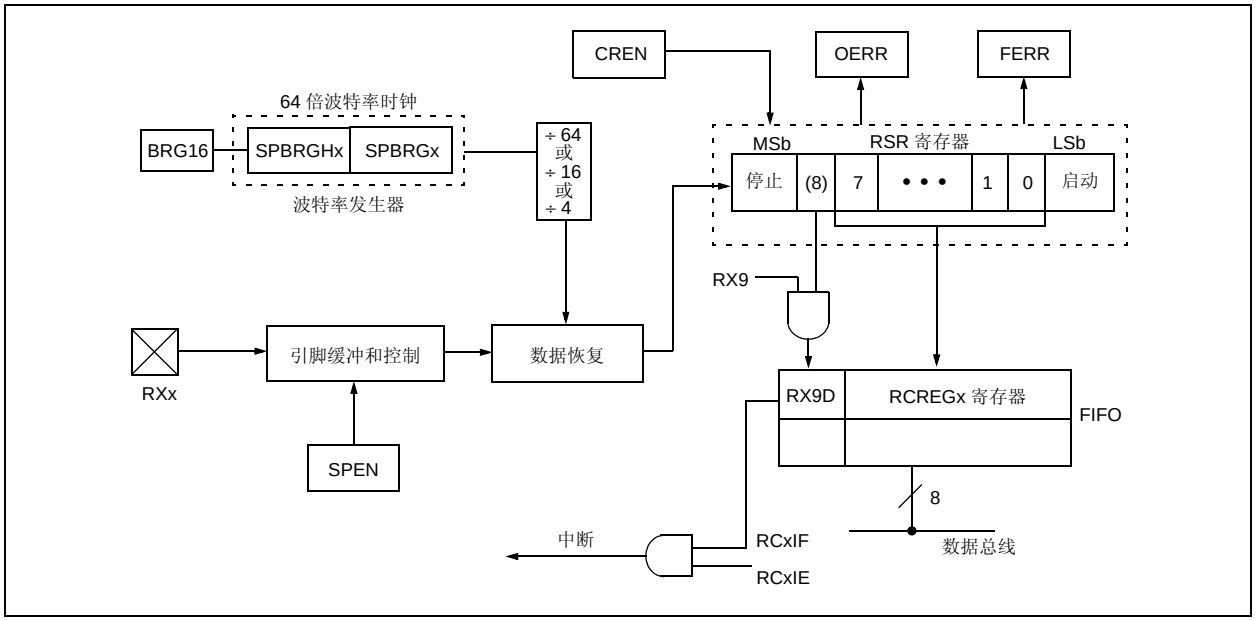
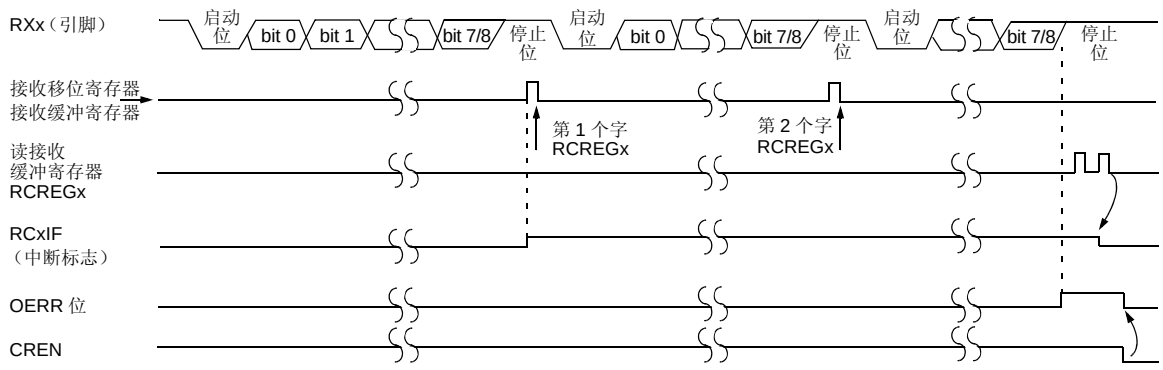


图 22-7：　　异步接收



注：　此时序图显示了在 RXx 输入引脚上接收的 3 个字。在第 3 个字后读 RCREGx （接收缓冲寄存器）会使 OERR （溢出）位置 1。

表 22-7：　　与异步接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
RCREG1	EUSART1 接收寄存器							
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 波特率发生器寄存器的高字节							
SPBRG1	EUSART1 波特率发生器寄存器的低字节							
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
RCREG2	EUSART2 接收寄存器							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 波特率发生器寄存器的高字节							
SPBRG2	EUSART2 波特率发生器寄存器的低字节							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

图注：　— = 未实现位，读为 0。异步接收不使用阴影单元。

22.3.4 接收到同步间隔字符时自动唤醒

在休眠模式下，EUSART 的所有时钟都会暂停。因此，波特率发生器处于不工作状态，不能正常进行字节接收。当 EUSART 工作在异步模式下时，自动唤醒功能使控制器可被 RXx/DTx 线上的活动唤醒。

通过将 WUE 位（BAUDCONx<1>）置 1 使能自动唤醒功能。一旦置1，RXx/DTx上的典型接收序列就被禁止，EUSART 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 RXx/DTx 线上由高至低的跳变。（这与同步间隔字符的开始或 LIN/J2602 协议的唤醒信号字符一致。）

唤醒事件后，模块产生一个 RCxIF 中断。在正常工作模式下，中断产生与 Q 时钟同步（图 22-8），而器件处于休眠模式时则异步发生（图 22-9）。通过读 RCREGx 寄存器可清除中断条件。

唤醒事件后，一旦 RXx 线上出现由低至高的跳变，WUE 位自动清零。此时，EUSART 模块将从空闲模式返回正常工作模式。这向用户表明同步间隔事件结束。

22.3.4.1 使用自动唤醒功能的特殊注意事项

因为自动唤醒功能是通过检测 RXx/DTx 上的上升沿跳变实现的，所以在停止位前该引脚上任何的状态改变都可能会产生错误的结束字符（End-Of-Character，EOC）并导致数据或帧错误。因此，为了确保正常的传输，必须首先发送全 0 字符。对于标准的 RS-232 器件，该字符是 00h（8 字节）；而对于 LIN/J2602 总线器件则是 000h （12 位）。

另外还必须考虑振荡器起振时间，尤其在使用具有较长起振时间的振荡器（即，HS 或 HSPLL 模式）的应用中。同步间隔（或唤醒信号）字符必须足够长，并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正确初始化 EUSART。

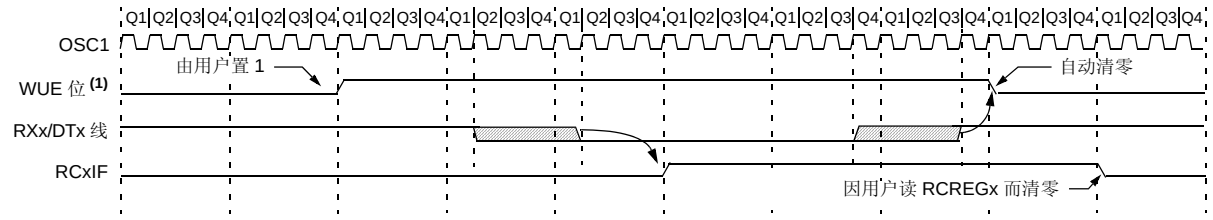
22.3.4.2 使用 WUE 位的特殊注意事项

使用 WUE 和 RCxIF 事件的时序来判断接收数据的有效性时，有可能会引起一些混淆。如前所述，将 WUE 位置 1 会使 EUSART 进入空闲状态。唤醒事件会通过将 RCxIF 位置 1 产生一个接收中断。此后当 RXx/DTx 上出现上升沿时， WUE 位被清零。然后通过读 RCREGx 寄存器清除中断条件。一般情况下，RCREGx 中的数据是无效数据，应该丢弃。

WUE 位清零 （或仍然置 1）同时 RCxIF 标志位置 1 并不能表明 RCREGx 中数据接收是完整的。用户应考虑在固件中同时验证是否完整地接收了数据。

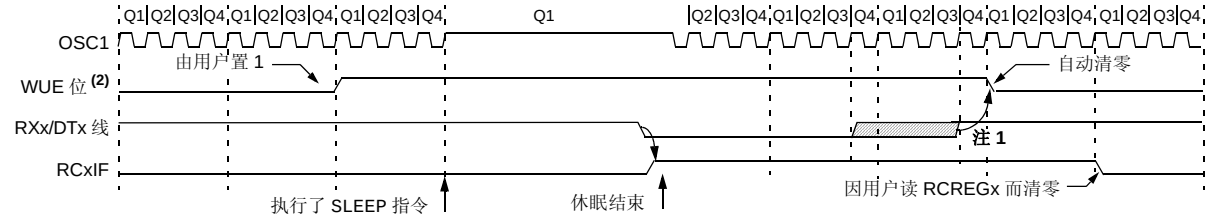
要确保不丢失实际数据，应检查 RCIDL 位来验证没有接收操作在进行。如果未发生接收操作，可在进入休眠模式前将 WUE 位置 1。

图 22-8： 正常操作时的自动唤醒位 （WUE）时序



注 1： WUE 位置 1 时， EUSART 保持在空闲状态。

图 22-9： 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，在振荡器就绪前可能发生 WUE 位自动清零。该时序与 Q 时钟出现与否无关。

2： WUE 位置 1 时， EUSART 保持在空闲状态。

22.3.5 间隔字符序列

EUSART 模块能够发送符合 LIN/J2602 总线标准的特殊间隔字符序列。发送的间隔字符包含 1 个启动位，随后的 12 个 0 位和 1 个停止位。当发送移位寄存器装入数据时，只要 SENDB 和 TXEN 位（分别为 TXSTAx<3> 和 TXSTAx<5>）置 1，就会发送帧间隔字符。请注意写入 TXREGx 的数据值会被忽略并发送全 0。

在发送了相应的停止位后，硬件会自动将 SENDB 位复位。这样用户可以在间隔字符（在 LIN/J2602 规范中通常是同步字符）后预先将下一个要发送字节装入发送 FIFO。

请注意在发送间隔字符时写入TXREGx的数据值会被忽略。写入仅仅是为了启动正确的序列。

正如在正常发送操作中一样，TRMT 位表明发送正在进行还是处于空闲状态。关于发送间隔字符的时序，请参见图 22-10。

22.3.5.1 间隔和同步发送序列

以下序列将发送一个报文帧头，它由一个间隔字符和其后的一个自动波特率同步字节组成。这是 LIN/J2602 总线主器件的典型序列。

1. 将 EUSART 配置为所需的模式。
2. 将 TXEN 和 SENDB 位置 1 以设置间隔字符。
3. 将无效字符装入 TXREGx，启动发送（该值会被忽略）。
4. 将 “55h” 写入 TXREGx，以便将同步字符装入 FIFO 缓冲区。
5. 发送间隔字符后，SENDB 位被硬件复位。此时，同步字符会以预先配置的模式发送。

当 TXREGx 为空时（由 TXxIF 指示），下一个数据字节会写入 TXREGx。

22.3.6 接收间隔字符

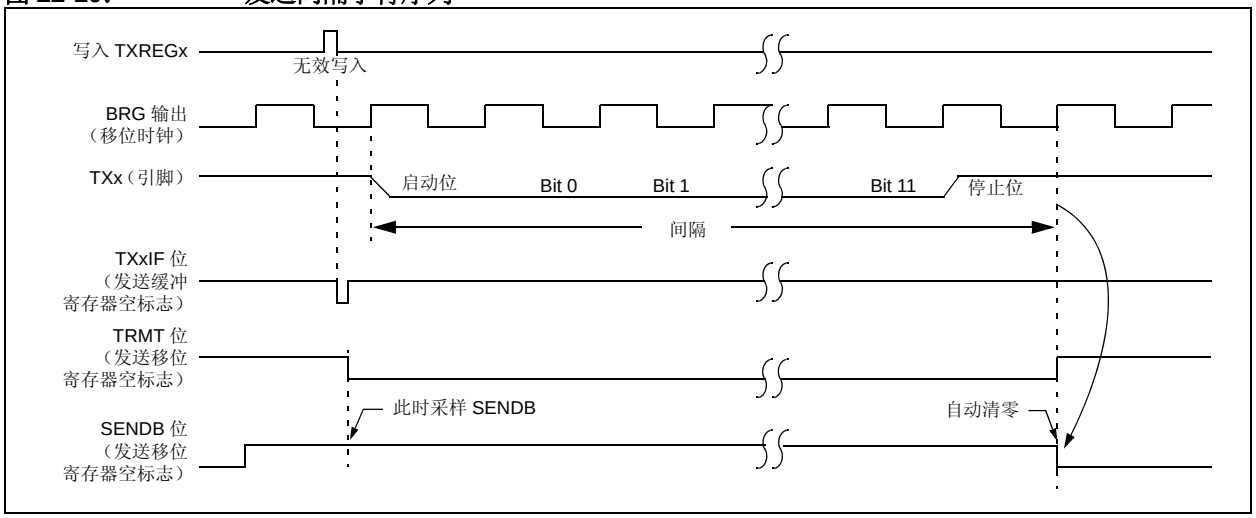
增强型 USART 模块接收间隔字符有两种方法。

第一种方法是强制将波特率配置为典型速度的 9/13。这可以使停止位在正确的采样点（对于间隔字符为启动位之后的 13 位，对于典型数据则是 8 个数据位）出现。

第二种方法采用**第 22.3.4 节 “ 接收到同步间隔字符时自动唤醒 ”** 中所述的自动唤醒功能。通过使能该功能，EUSART 将采样 RXx/DTx 上的下两次跳变，产生一个 RCxIF 中断，并接收下一个数据字节并再产生一次中断。

请注意在间隔字符后，用户通常希望使能自动波特率检测功能。采用这两种方法时，用户均可在检测到 TXxIF 中断时立即将 ABDEN 位置 1。

图 22-10： 发送间隔字符序列



22.4 EUSART 同步主模式

将 CSRC 位（TXSTAx<7>）置 1 可进入同步主模式。在该模式下，数据以半双工方式发送（即发送和接收不能同时进行）。发送数据时禁止接收，反之亦然。将 SYNC 位（TXSTAx<4>）置 1 可进入同步模式。此外，应将使能位 SPEN（RCSTAx<7>）置 1，分别将 TXx 和 RXx 引脚配置为 CKx（时钟）和 DTx（数据）线。

主模式意味着处理器在 CKx 线上发送主时钟信号。时钟极性通过TXCKP位（BAUDCONx<4>）选择。将TXCKP置 1 将 CKx 的空闲状态设为高电平，将该位清零则将 CKx 的空闲状态设为低电平。该选项支持将本模块与 Microwire 器件配合使用。

22.4.1 EUSART 同步主发送

图 22-3 给出了 EUSART 发送器框图。发送器的核心是发送（串行）移位寄存器（TSR）。移位寄存器从读 / 写发送缓冲寄存器 TXREGx 中获取数据。用软件将数据装入 TXREGx 寄存器。在前一次装入数据的最后一位发送完成后，才向 TSR 寄存器装入新数据。一旦最后一位发送完成，就会将 TXREGx 寄存器的新数据（如果有）装入 TSR。

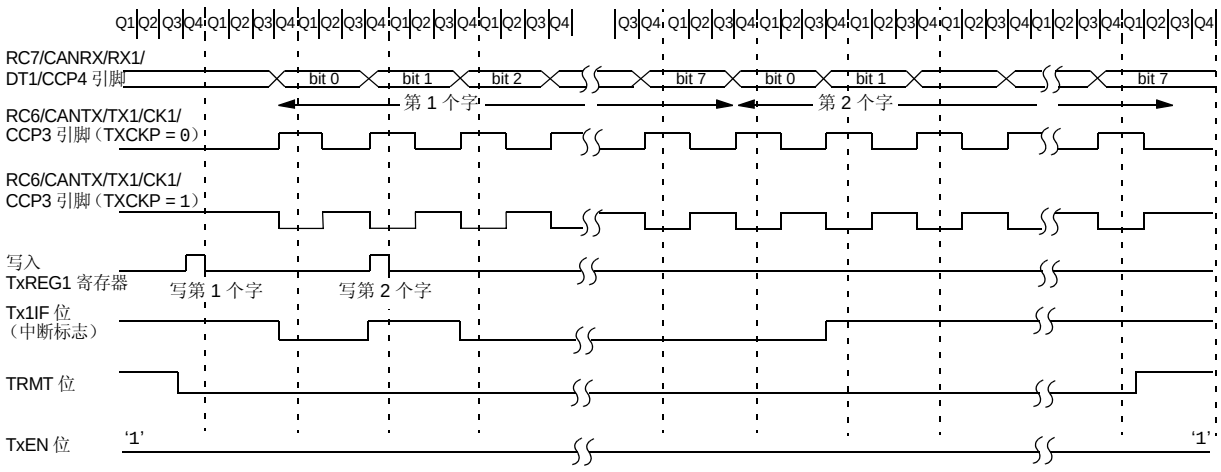
一旦 TXREGx 寄存器向 TSR 寄存器传输了数据（在 1 个 TCY 内发生），TXREGx 寄存器就为空，同时 TXxIF 标志位被置 1。可以通过将中断允许位 TXxIE 置 1 或清零来允许或禁止该中断。TXxIF 的设置不受使能位 TXxIE 状态的影响，且不能用软件清零。只有在新数据装入 TXREGx 寄存器时， TXxIF 才会复位。

标志位 TXxIF 指示 TXREGx 寄存器的状态，而另一个标志位 TRMT （TXSTAx<1>）则指示 TSR 寄存器的状态。TRMT 位是一个只读位，当 TSR 为空时，TRMT 被置1。TRMT位与任何中断均无关联，因此要确定TSR 寄存器是否为空，用户只能对该位进行查询。TSR 不映射到数据存储器中，因此用户无法使用。

设置同步主发送操作的步骤如下：

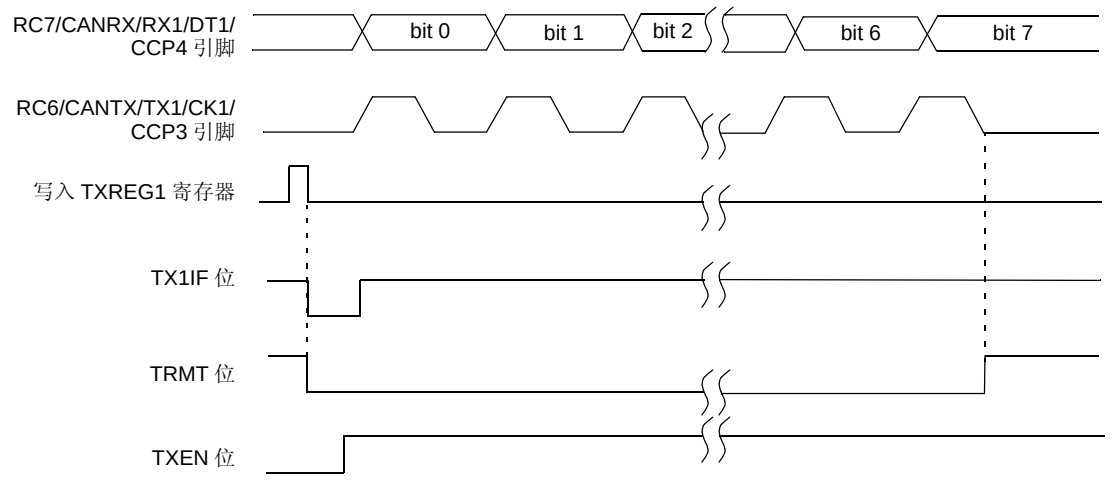
1. 对 SPBRGHx:SPBRGx 寄存器进行初始化，设置合适的波特率。按需要将 BRG16 位置 1 或清零，获得所需的波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1，使能同步主串口。
3. 如果需要中断，将中断允许位 TXxIE 置 1。
4. 如果需要 9 位发送，将 TX9 位置 1。
5. 通过将 TXEN 位置 1 使能发送。
6. 如果选择了 9 位发送，应将第 9 位装入 TX9D 位。
7. 将数据装入 TXREGx 寄存器，启动发送。
8. 如果使用中断，应确保 GIE 和 PEIE 位（INTCON<7:6>）置 1。

图 22-11： 同步发送



注： 同步主模式，SPBRGx = 0；连续发送两个 8 位字。本例同样适用于 EUSART2（RB6/PGC/TX2/CK2/KBI2 和 RB7/PGD/T3G/RX2/DT2/KBI3）。

图 22-12：　　同步发送 （由 TXEN 位控制）



注： 本例同样适用于 EUSART2 （RB6/PGC/TX2/CK2/KBI2 和 RB7/PGD/T3G/RX2/DT2/KBI3）。

表 22-8：　　与同步主发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
TXREG1	EUSART1 发送寄存器							
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 波特率发生器寄存器的高字节							
SPBRG1	EUSART1 波特率发生器寄存器的低字节							
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
TXREG2	EUSART2 发送寄存器							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 波特率发生器寄存器的高字节							
SPBRG2	EUSART2 波特率发生器寄存器的低字节							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

图注： — = 未实现，读为 0。同步主发送不使用阴影单元。

22.4.2 EAUSART 同步主接收

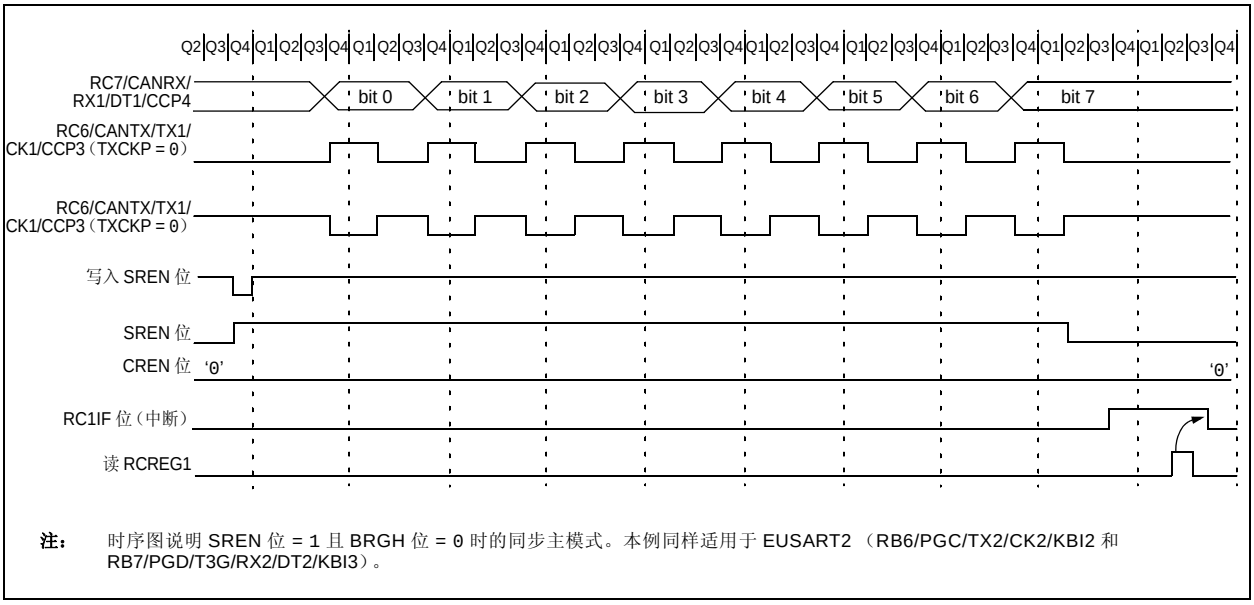
一旦选择了同步模式，可通过将单字节接收使能位 SREN（RCSTAx<5>）或连续接收使能位CREN（RCSTAx<4>）置1使能接收。在时钟的下降沿采样RXx引脚上的数据。

如果使能位 SREN 置 1，则只接收单个字。如果将使能位 CREN 置 1，则会连续接收数据，直到 CREN 位清零。如果两个位均被置 1，则 CREN 优先。

设置同步主接收操作的步骤如下：

1. 对 SPBRGHx:SPBRGx 寄存器进行初始化，设置合适的波特率。按需要将 BRG16 位置 1 或清零，获得所需的波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1，使能同步主串口。
3. 确保将 CREN 和 SREN 位清零。
4. 如果需要中断，将中断允许位 RCxIE 置 1。
5. 如果需要接收 9 位数据，将 RX9 位置 1。
6. 如果需要单字接收，将 SREN 位置 1。如果需要连续接收，将 CREN 位置 1。
7. 接收完成时中断标志位 RCxIF 将被置 1，此时如果中断允许位RCxIE已置1，还将产生一个中断。
8. 读取 RCSTAx 寄存器取得第 9 位数据 （如果已使能），并确定接收时是否发生了错误。
9. 读 RCREGx 寄存器来读取接收到的 8 位数据。
10. 如果发生错误，将 CREN 位清零以清除错误。
11. 如果使用中断，应确保 GIE 和 PEIE 位（INTCON<7:6>）置 1。

图 22-13： 同步接收 （主模式， SREN）



注： 时序图说明 SREN 位 = 1 且 BRGH 位 = 0 时的同步主模式。本例同样适用于 EUSART2 （RB6/PGC/TX2/CK2/KBI2 和 RB7/PGD/T3G/RX2/DT2/KBI3）。

表 22-9：　　与同步主接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
RCREG1	EUSART1 接收寄存器							
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 波特率发生器寄存器的高字节							
SPBRG1	EUSART1 波特率发生器寄存器的低字节							
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
RCREG2	EUSART2 接收寄存器							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 波特率发生器寄存器的高字节							
SPBRG2	EUSART2 波特率发生器寄存器的低字节							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

图注：　　— = 未实现，读为 0。同步主接收不使用阴影单元。

22.5 EUSART 同步从模式

将 CSRC（TXSTAx<7>）清零可进入同步从模式。该模式与同步主模式的区别在于移位时钟由CKx引脚上的外部时钟提供（主模式下由内部时钟提供）。这使器件能在任何低功耗模式下发送或接收数据。

22.5.1 EUSART 同步从发送

除了休眠模式以外，同步主模式和从模式的工作原理是相同的。

如果向 TXREGx 写两个字，然后执行 SLEEP 指令，则会发生以下事件：

a) 第一个字将立即传送到 TSR 寄存器并发送。
b) 第二个字将保留在 TXREGx 寄存器中。
c) 标志位 TXxIF 不会被置 1。
d) 第一个字移出 TSR 后，TXREGx 寄存器会将第二个字传送到 TSR，此时标志位 TXxIF 将置 1。
e) 如果中断允许位 TXxIE 置 1，中断会将芯片从休眠状态唤醒。如果允许了全局中断，程序将会跳转到中断向量处执行。

设置同步从发送操作的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1 并将 CSRC 位清零，使能同步从串口。
2. 清零 CREN 和 SREN 位。
3. 如果需要中断，将中断允许位 TXxIE 置 1。
4. 如果需要 9 位发送，将 TX9 位置 1。
5. 通过将使能位 TXEN 置 1 使能发送。
6. 如果选择了 9 位发送，应将第 9 位装入 TX9D 位。
7. 将数据装入 TXREGx 寄存器，启动发送。
8. 如果使用中断，应确保 GIE 和 PEIE 位（INTCON<7:6>）置 1。

表 22-10： 与同步从发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
TXREG1	EUSART1 发送寄存器							
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 波特率发生器寄存器的高字节							
SPBRG1	EUSART1 波特率发生器寄存器的低字节							
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
TXREG2	EUSART2 发送寄存器							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 波特率发生器寄存器的高字节							
SPBRG2	EUSART2 波特率发生器寄存器的低字节							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

图注： — = 未实现，读为 0。同步从发送不使用阴影单元。

22.5.2 EUSART 同步从接收

除了休眠模式、任何空闲模式以及在从模式下忽略 SREN 位的情况以外，同步主模式和从模式的工作原理是相同的。

如果在进入休眠或任何空闲模式前将 CREN 位置 1 使能接收，那么在低功耗模式下可以接收一个数据字。接收到数据字后，RSR 寄存器会将数据传送到 RCREGx 寄存器。如果中断允许位 RCxIE 已置 1，产生的中断会将芯片从低功耗模式唤醒。如果允许了全局中断，程序将会跳转到中断向量处执行。

设置同步从接收操作的步骤如下：

1. 通过将 SYNC 和 SPEN 位置 1 并将 CSRC 位清零，使能同步从串口。
2. 如果需要中断，将中断允许位 RCxIE 置 1。
3. 如果需要接收 9 位数据，将 RX9 位置 1。
4. 将使能位 CREN 置 1 以使能接收。
5. 接收完成时 RCxIF 位将被置 1。如果中断允许位 RCxIE 置 1，还将产生一个中断。
6. 读取 RCSTAx 寄存器取得第 9 位数据 （如果已使能），并确定接收时是否发生了错误。
7. 读 RCREGx 寄存器来读取接收到的 8 位数据。
8. 如果发生错误，将 CREN 位清零以清除错误。
9. 如果使用中断，应确保 GIE 和 PEIE 位（INTCON<7:6>）置 1。

表 22-11： 与同步从接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
PIR3	—	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	—
PIE3	—	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	—
IPR3	—	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	—
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
RCREG1	EUSART1 接收寄存器							
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH1	EUSART1 波特率发生器寄存器的高字节							
SPBRG1	EUSART1 波特率发生器寄存器的低字节							
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
RCREG2	EUSART2 接收寄存器							
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN
SPBRGH2	EUSART2 波特率发生器寄存器的高字节							
SPBRG2	EUSART2 波特率发生器寄存器的低字节							
PMD0	CCP5MD	CCP4MD	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSPMD
ODCON	SSPOD	CCP5OD	CCP4OD	CCP3OD	CCP2OD	CCP1OD	U2OD	U1OD

图注： — = 未实现，读为 0。同步从接收不使用阴影单元。

23.0 12 位模数转换器 （A/D）模块

PIC18F66K80 系列器件中的模数（Analog-to-Digital，A/D）转换器模块有 8 路输入 （对于 28 引脚器件）和 11 路输入 （对于 40/44 引脚和 64 引脚器件）。该模块能将一个模拟输入信号转换成相应的 12 位数字。

该模块具有以下寄存器：

- A/D 控制寄存器 0 （ADCON0）
- A/D 控制寄存器 1 （ADCON1）
- A/D 控制寄存器 2 （ADCON2）
- A/D 端口配置寄存器 0 （ANCON0）
- A/D 端口配置寄存器 1 （ANCON1）
- ADRESH （A/D 结果寄存器的高字节）
- ADRESL （A/D 结果寄存器的低字节）

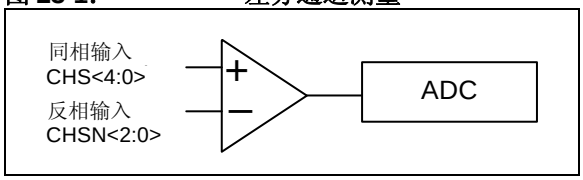
ADCON0 寄存器 （如寄存器 23-1 所示）用于控制 A/D 模块的工作。 ADCON1 寄存器 （如寄存器 23-2 所示）用于配置参考电压和特殊触发信号选择。ADCON2 寄存器 （如寄存器 23-3 所示）用于配置 A/D 时钟源、可编程采集时间和对齐方式。

23.1 差分 A/D 转换器

PIC18F66K80 系列器件中的转换器以差分 A/D 的形式实现，它可以测量两个通道之间的差分电压并转换为数字值 （见图 23-1）。

通过清零 CHSN 位（ADCON1<2:0>），还可以将转换器配置为测量单个输入的电压。使用该配置时，反相通道输入会在内部连接到 AVSS （见图 23-2）。

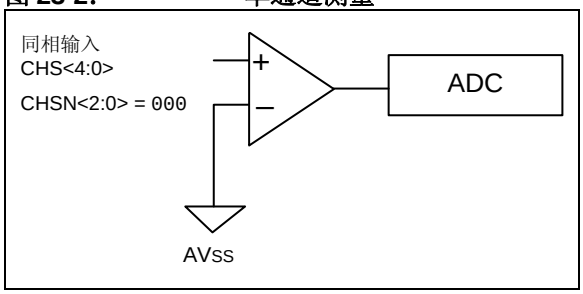
图 23-1： 差分通道测量



差分转换会将两个输入通道送至单位增益差分放大器。同相通道输入使用 CHS 位（ADCON0<6:2>）来选择，反相通道输入使用 CHSN 位（ADCON1<2:0>）来选择。

放大器的输出会被送至 A/D 转换器，如图 23-1 所示。12 位的结果将存储在 ADRESH 和 ADRESL 寄存器中。另外还有一个指示 12 位结果是正值还是负值的位。

图 23-2： 单通道测量



在单通道测量模式下，反相输入通过清零 CHSN 位（ADCON1<2:0>）连接到 AVSS。

23.2 A/D 寄存器

23.2.1 A/D 控制寄存器

寄存器 23-1： ADCON0：A/D 控制寄存器 0

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CHS4	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
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 6-2 **CHS<4:0>：**模拟通道选择位

00000 = 通道 00（AN0）
00001 = 通道 01（AN1）
00010 = 通道 02（AN2）
00011 = 通道 03（AN3）
00100 = 通道 04（AN4）
00101 = 通道 05（AN5）(1,2)
00110 = 通道 06（AN6）(1,2)
00111 = 通道 07（AN7）(1,2)
01000 = 通道 08（AN8）
01001 = 通道 09（AN9）
01010 = 通道 10（AN10）
01011 =（保留）(2)
01100 =（保留）(2)
01101 =（保留）(2)
01110 =（保留）(2)
01111 =（保留）(2)
10000 =（保留）(2)
10001 =（保留）(2)
10010 =（保留）(2)
10011 =（保留）(2)
10100 =（保留）(2)
10101 =（保留）(2)
10110 =（保留）(2)
10111 =（保留）(2)
11000 =（保留）(2)
11001 =（保留）(2)
11010 =（保留）(2)
11011 =（保留）(2)
11100 =（多路开关断开）(3)
11101 = 通道 29（温度检测二极管）
11110 = 通道 30（V_{DDCORE}）
11111 = 通道 31（1.024V 带隙）

bit 1 **GO/$\overline{\text{DONE}}$：**A/D 转换状态位
1 = A/D 周期正在进行。将该位置 1 可启动 A/D 转换周期。当 A/D 转换完成时，该位由硬件自动清零。
0 = A/D 转换已完成或未进行

bit 0 **ADON：**A/D 使能位
1 = A/D 转换器工作
0 = A/D 转换模块关闭且不消耗工作电流

注 1： 这些通道在 28 引脚器件上未实现。
2： 对未实现通道执行转换将返回随机值。
3： 通道 28 会关闭模拟多路开关，以使 ADC 输入的容性负载达到最低值，从而实现更高分辨率的 CTMU 时间测量。

寄存器 23-2： ADCON1：A/D 控制寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-x	R/W-x	R/W-x	R/W-x
TRIGSEL1	TRIGSEL0	VCFG1	VCFG0	VNCFG	CHSN2	CHSN1	CHSN0
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 **TRIGSEL<1:0>：**特殊事件触发信号选择位
11 = 选择来自 CCP2 的特殊事件触发信号
10 = 选择来自 Timer1 的特殊事件触发信号
01 = 选择来自 CTMU 的特殊事件触发信号
00 = 选择来自 ECCP1 的特殊事件触发信号

bit 5-4 **VCFG<1:0>：**A/D V_{REF+} 配置位
11 = 内部 V_{REF+} （4.096V）
10 = 内部 V_{REF+} （2.048V）
01 = 外部 V_{REF+}
00 = AV_{DD}

bit 3 **VNCFG：**A/D V_{REF-} 配置位
1 = 外部 V_{REF}
0 = AV_{SS}

bit 2-0 **CHSN<2:0>：**模拟反相通道选择位
111 = 通道 07 （AN6）
110 = 通道 06 （AN5）
101 = 通道 05 （AN4）
100 = 通道 04 （AN3）
011 = 通道 03 （AN2）
010 = 通道 02 （AN1）
001 = 通道 01 （AN0）
000 = 通道 00 （AV_{SS}）

寄存器 23-3：　　ADCON2：A/D 控制寄存器 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
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　**ADFM：**A/D 结果格式选择位
1 = 右对齐
0 = 左对齐

bit 6　**未实现：**读为 0

bit 5-3　**ACQT<2:0>：**A/D 采集时间选择位
111 = 20 TAD
110 = 16 TAD
101 = 12 TAD
100 = 8 TAD
011 = 6 TAD
010 = 4 TAD
001 = 2 TAD
000 = 0 TAD(1)

bit 2-0　**ADCS<2:0>：**A/D 转换时钟选择位
111 = FRC（时钟来自 A/D RC 振荡器）(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC（时钟来自 A/D RC 振荡器）(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

注　1：　如果选择了A/D FRC时钟源，在A/D时钟启动之前会加上一个TCY（指令周期）的延时。这允许在启动转换之前执行 SLEEP 指令。

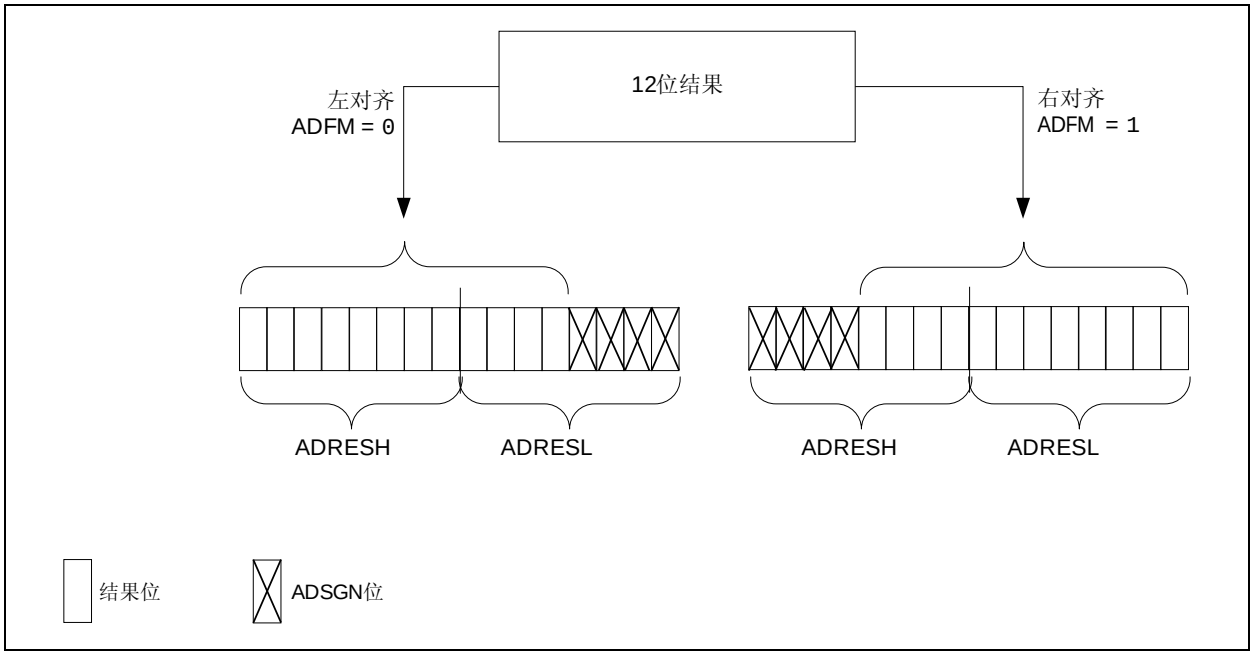
23.2.2 A/D 结果寄存器

在转换完成时，12位A/D结果和符号扩展位（ADSGN）将装入ADRESH:ADRESL 寄存器对。该寄存器对为16位宽。A/D 模块提供了对 16 位结果寄存器中的 12 位结果进行左对齐或右对齐的灵活功能。 A/D 格式选择位（ADFM）用于控制该对齐方式。

图 23-3 显示了 A/D 结果的对齐操作和符号扩展位（ADSGN）的位置。符号扩展位使得可以更方便地对结果执行 16 位数学运算。

当禁止 A/D 转换器时，这两个 8 位寄存器可以用作两个通用寄存器。

图 23-3： A/D 结果对齐方式



寄存器 23-4： ADRESH：A/D 结果高字节寄存器，左对齐（ADFM = 0）

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
ADRES11	ADRES10	ADRES9	ADRES8	ADRES7	ADRES6	ADRES5	ADRES4
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 **ADRES<11:4>：**A/D 结果高字节位

寄存器 23-5： ADRESL：A/D 结果低字节寄存器，左对齐（ADFM = 0）

R/W-x	R/W-x	U-x	U-x	U-x	U-x	U-x	R/W-x
ADRES3	ADRES2	ADRES1	ADRES0	ADSGN	ADSGN	ADSGN	ADSGN
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 7-4 **ADRES<3:0>**：A/D 结果低字节位

bit 3-0 **ADSGN**：A/D 结果符号位
1 = A/D 结果为负
0 = A/D 结果为正

寄存器 23-6： ADRESH：A/D 结果高字节寄存器，右对齐（ADFM = 1）

U-x	U-x	U-x	U-x	R/W-x	R/W-x	R/W-x	R/W-x
ADSGN	ADSGN	ADSGN	ADSGN	ADRES11	ADRES10	ADRES9	ADRES8
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 7-4 **ADSGN**：A/D 结果符号位
1 = A/D 结果为负
0 = A/D 结果为正

bit 3-0 **ADRES<11:8>**：A/D 结果高字节位

寄存器 23-7： ADRESL：A/D 结果低字节寄存器，右对齐（ADFM = 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
ADRES7	ADRES6	ADRES5	ADRES4	ADRES3	ADRES2	ADRES1	ADRES0
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 **ADRES<7:0>**：A/D 结果低字节位

ANCONx 寄存器用于配置与每个模拟通道关联的 I/O 引脚的操作。清零 ANSELx 位时，相应引脚（ANx）将配置为数字 I/O。将该位置 1 时，引脚将作为模拟输入（用于 A/D 转换器或比较器模块），并且所有数字外设都会被禁止，数字输入将读为 0。

通常，在发生任意器件复位时，与模拟输入复用的 I/O 引脚在默认情况下将用作模拟功能。

寄存器 23-8：　　ANCON0：A/D 端口配置寄存器 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
ANSEL7(1)	ANSEL6(1)	ANSEL5(1)	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
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　　**ANSEL<7:0>：**模拟端口配置位 （AN7 至 AN0）(1)
1 = 引脚配置为模拟通道：禁止数字输入，所有输入都读为 0
0 = 引脚配置为数字端口

注　1： AN14 至 AN11 以及 AN7 至 AN5 仅在 40/44 引脚和 64 引脚器件上实现。对于 28 引脚器件，仍然为这些通道实现了相应的 ANSELx 位，但它们不起作用。

寄存器 23-9：　　ANCON1：A/D 端口配置寄存器 1

U-1	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	ANSEL14(1)	ANSEL13(1)	ANSEL12(1)	ANSEL11(1)	ANSEL10	ANSEL9	ANSEL8
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 6-0　　**ANSEL<14:8>：**模拟端口配置位 （AN14 至 AN8）(1)
1 = 引脚配置为模拟通道：禁止数字输入，所有输入都读为 0
0 = 引脚配置为数字端口

注　1： AN14 至 AN11 以及 AN7 至 AN5 仅在 40/44 引脚和 64 引脚器件上实现。对于 28 引脚器件，仍然为这些通道实现了相应的 ANSELx 位，但它们不起作用。

可通过软件选择将器件的正负电源电压（AVDD 和 AVSS）或 RA3/VREF+/AN3 和 RA2/VREF-/AN2 引脚上的电压作为模拟参考电压。VREF+ 具有两个额外的内部参考电压选项：2.048V 和 4.096V。

A/D 转换器可以在器件处于休眠模式时单独工作。要在**休眠**模式下工作， A/D 转换时钟必须来自 A/D 的内部 RC 振荡器。

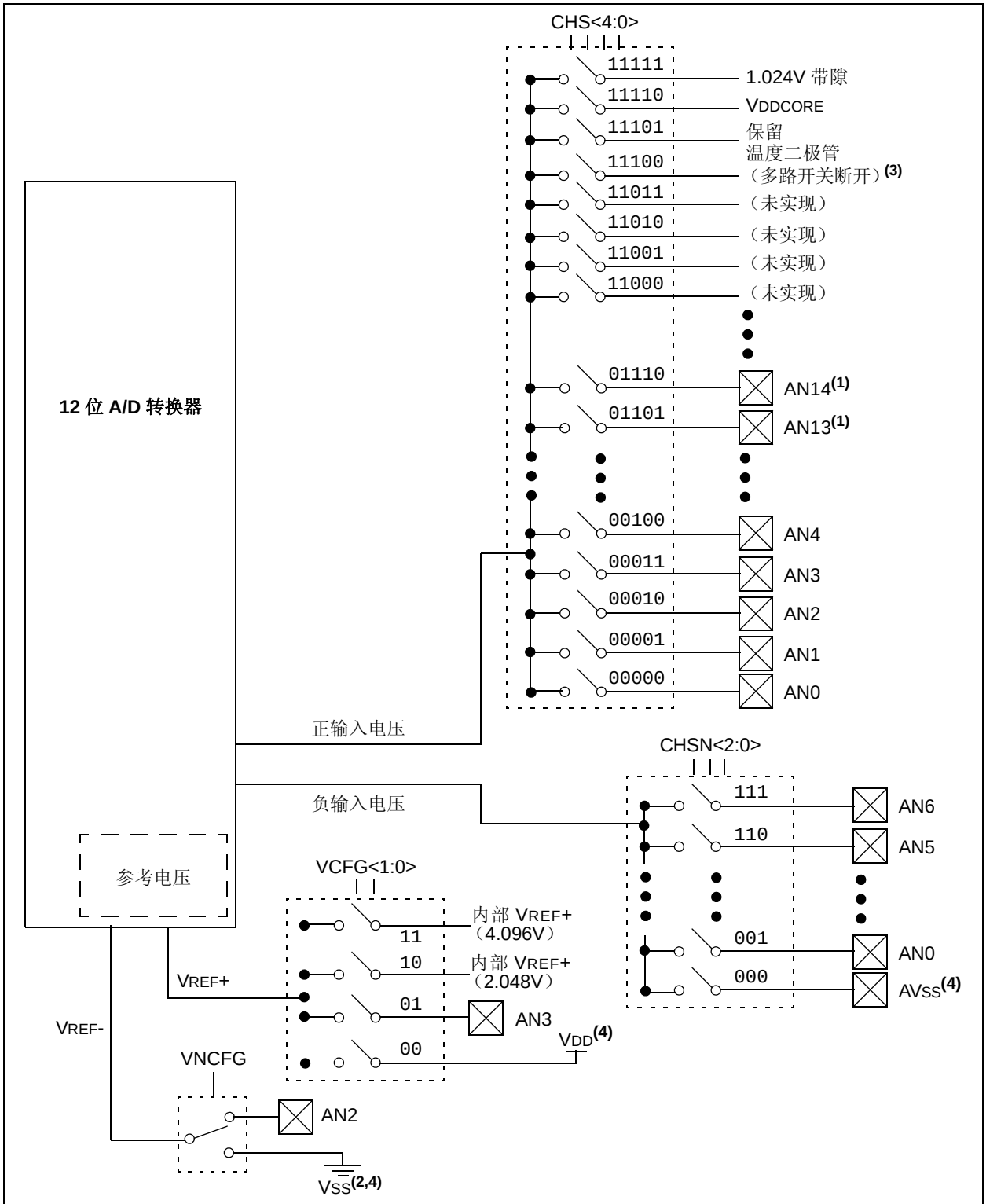
采样 / 保持电路的输出是转换器的输入，A/D 转换器采用逐次逼近法得到转换结果。

与 A/D 转换器相关的每个端口引脚都可以被配置为模拟输入或数字 I/O。ADRESH 和 ADRESL 寄存器保存 A/D 转换的结果。当 A/D 转换完成时，结果被装入 ADRESH:ADRESL 寄存器对， GO/DONE 位（ADCON0<1>）被清零且 A/D 中断标志位 ADIF（PIR1<6>）被置 1。

器件复位将强制所有寄存器为复位状态。这将强制关闭 A/D 模块并中止任何正在进行的转换。上电复位时，ADRESH:ADRESL 寄存器对的值保持不变。上电复位后，这些寄存器的值不确定。

图 23-4 给出了 A/D 模块的框图。

图 23-4：　　A/D 框图



注　**1：** 通道 AN14 至 AN11 以及 AN7 至 AN5 仅在 40/44 引脚和 64 引脚器件上实现。对于 28 引脚器件，仍然为这些通道实现了相应的 ANSELx 位，但它们不起作用。

2： 在 I/O 引脚与 VDD 和 VSS 之间接有保护二极管。

3： 通道 28 会关闭模拟多路开关，以使 ADC 输入的容性负载达到最低值，从而实现更高分辨率的 CTMU 时间测量。

4： 在 I/O 引脚与 VDD 和 VSS 之间接有保护二极管。

在根据需要配置好 A/D 模块之后，必须在转换开始之前对选定的通道进行采集。必须将模拟输入通道的相应 TRIS 位选择为输入。采集时间的确定，请参见**第 23.3 节 “A/D 采集要求 ”**。采集时间一结束，即可启动 A/D 转换。可将采集时间编程设定在 GO/DONE 位置 1 和实际转换启动之间。

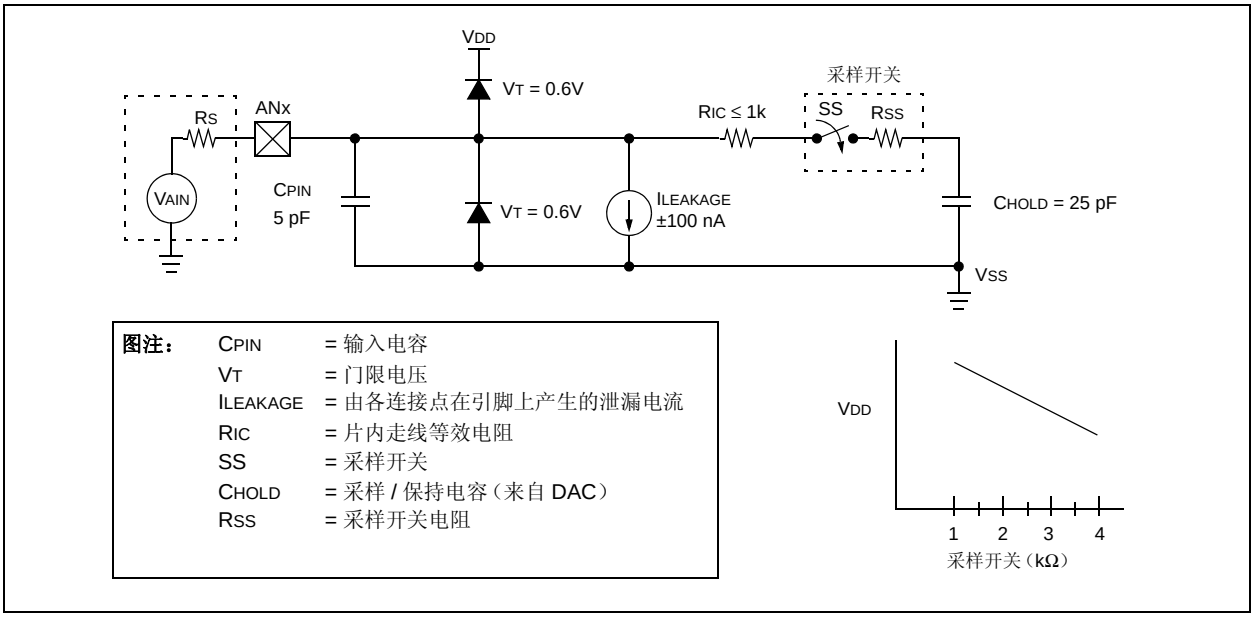
要执行 A/D 转换，请遵循以下步骤：

1. 配置 A/D 模块：
   - 将所需的 ADC 引脚配置为模拟引脚（ANCON0 和 ANCON1）
   - 设置参考电压 （ADCON1）
   - 选择 A/D 同相和反相输入通道 （ADCON0 和 ADCON1）
   - 选择 A/D 采集时间 （ADCON2）
   - 选择 A/D 转换时钟 （ADCON2）
   - 开启 A/D 模块 （ADCON0）
2. 配置 A/D 中断 （如果需要）：
   - 清零 ADIF 位 （PIR1<6>）
   - 将 ADIE 位 （PIE1<6>）置 1
   - 将 GIE 位 （INTCON<7>）置 1
3. 等待所需的采集时间 （如果需要）。
4. 启动转换：
   - 将 GO/DONE 位 （ADCON0<1>）置 1
5. 等待 A/D 转换完成，可通过以下两种方法之一判断转换是否完成：
   - 查询 GO/DONE 位是否被清零

   或
   - 等待 A/D 中断
6. 读取 A/D 结果寄存器（ADRESH:ADRESL），需要时清零 ADIF 位。
7. 如需进行下一次转换，请从步骤 1 或 2 开始。

   每位的 A/D 转换时间定义为 TAD。在下一次采集开始前需要等待至少 2 个 TAD 的时间。

图 23-5： 模拟输入模型



23.3 A/D 采集要求

为了使 A/D 转换器达到规定的精度，必须使充电保持电容（CHOLD）充满至输入通道的电压值。模拟输入模型见图 23-5。信号源阻抗（RS）和内部采样开关阻抗（RSS）直接影响为电容 CHOLD 充电所需的时间。采样开关阻抗值（RSS）随器件电压（VDD）不同而改变。

信号源阻抗会影响模拟输入的失调电压（由于引脚泄漏电流的原因）。**模拟信号源的最大阻抗推荐值为 2.5 kΩ**。选择或改变模拟输入通道后，必须对通道进行采样才能启动转换，采样时间必须大于最小采集时间。

注：	当启动转换时，要将保持电容与输入引脚断开。

可以使用公式 23-1 来计算最小采集时间。该公式假设误差为 1/2 LSb（A/D 转换需要 1,024 步）。1/2 LSb 的误差是 A/D 达到规定分辨率所能允许的最大误差。

公式 23-3 给出了所需的最小采集时间 TACQ 的计算过程。计算结果基于以下对应用系统的假设：

CHOLD	=	25 pF
RS	=	2.5 kΩ
转换误差	≤	1/2 LSb
VDD	=	3V → RSS = 2 kΩ
温度	=	85°C （系统最大值）

公式 23-1： 采集时间

$$T_{ACQ} = \text{放大器稳定时间} + \text{保持电容充电时间} + \text{温度系数}$$
$$= T_{AMP} + T_C + T_{COFF}$$

公式 23-2： A/D 最小充电时间

$$V_{HOLD} = (V_{REF} - (V_{REF}/2048)) \cdot (1 - e^{(-T_C/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

或

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2048)$$

公式 23-3： 计算所需要的最小采集时间

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$

$$T_{AMP} = 0.2\ \mu s$$

$$T_{COFF} = (\text{温度} - 25°C)(0.02\ \mu s/°C)$$
$$= (85°C - 25°C)(0.02\ \mu s/°C)$$
$$= 1.2\ \mu s$$

只有在温度 > 25°C 时才需要温度系数。当温度低于 25°C 时， $T_{COFF} = 0$ ms。

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2048)\ \mu s$$
$$= -(25\ pF)(1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega) \ln(0.0004883)\ \mu s$$
$$= 1.05\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 1.05\ \mu s + 1.2\ \mu s$$
$$= 2.45\ \mu s$$

23.4 选择和配置自动采集时间

ADCON2 寄存器允许用户选择采集时间，该时间在每当 GO/DONE 位置 1 时发生。

当 GO/DONE 位被置 1 时，采样停止并启动转换。用户应确保在选择所需的输入通道和将 GO/DONE 位置 1 之间经过了所需的采集时间。

这发生在 ACQT<2:0> 位（ADCON2<5:3>）保持在其复位状态（000）的情况下，与不提供可编程采集时间的器件相兼容。

如果需要，可设置 ACQTx 位以便为 A/D 模块选择可编程采集时间。当 GO/DONE 位被置 1 时，A/D 模块会继续在选定采集时间内采样输入通道，然后自动启动一次转换。由于采集时间已被编程，因此没有必要在选择通道和将 GO/DONE 位置 1 之间等待一个采集时间。

在这两种情况下，当转换完成时，GO/DONE 位均被清零，ADIF 标志位均被置 1 并且 A/D 开始再次对当前选定的通道进行采样。如果采集时间已被编程，那么将不会有任何采集时间结束、转换开始的指示。

23.5 选择 A/D 转换时钟

每位的 A/D 转换时间定义为 TAD。每完成一次 12 位 A/D 转换需要 14 个 TAD。可用软件选择 A/D 转换的时钟源。

TAD 有以下可能的选择：

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- 使用内部 RC 振荡器

为了实现正确的 A/D 转换，A/D 转换时钟（TAD）必须尽可能得小，但它必须大于最小 TAD。（更多信息，请参见表 31-26 中的参数 130。）

表 23-1 给出了器件在不同工作频率下和选择不同的 A/D 时钟源时得到的 TAD。

表 23-1： 不同器件工作频率下的 TAD

A/D 时钟源 （TAD）		最高器件频率
工作时钟源	ADCS<2:0>	
2 TOSC	000	2.50 MHz
4 TOSC	100	5.00 MHz
8 TOSC	001	10.00 MHz
16 TOSC	101	20.00 MHz
32 TOSC	010	40.00 MHz
64 TOSC	110	64.00 MHz
RC(2)	x11	1.00 MHz(1)

注 1： RC 时钟源的典型 TAD 时间为 4 µs。

2： 当器件工作频率高于 1 MHz 时，整个转换过程必须在休眠模式下进行，否则 A/D 转换精度可能超出规范。

23.6 配置模拟端口引脚

ANCON0、ANCON1、TRISA、TRISB 和 TRISC 寄存器用于控制 A/D 端口引脚的工作。必须将相应的 TRISx 位置 1（输入）才能将引脚配置为模拟输入引脚。如果将 TRISx 位清零（输出），则数字输出电平（VOH 或 VOL）将被转换。

A/D 转换操作与 CHS<3:0> 位和 TRISx 位的状态无关。

注 1： 当读取端口寄存器时，所有配置为模拟输入通道的引脚均读为零（低电平）。配置为数字输入的引脚将按模拟输入进行转换。配置为数字输入的引脚上的模拟电平将被精确转换。

2： 定义为数字输入引脚上的模拟电平可能会导致数字输入缓冲器消耗的电流超出器件规范。

23.7 A/D 转换

图 23-6 显示了在 GO/$\overline{\text{DONE}}$ 位置 1 且 ACQT<2:0> 位被清零后 A/D 转换器的工作。转换在下一条指令执行之后开始，以允许器件在转换开始之前进入休眠模式。

图 23-7 显示了在 GO/$\overline{\text{DONE}}$ 位置 1，ACQT<2:0> 位被设置为 010 且选择 4 TAD 采集时间后 A/D 转换器的工作。

在转换期间清零 GO/$\overline{\text{DONE}}$ 位将中止当前的 A/D 转换。不会用部分完成的 A/D 转换结果更新 A/D 结果寄存器对。这意味着 ADRESH:ADRESL 寄存器将仍然保存上一次的转换结果（或上一次写入 ADRESH:ADRESL 寄存器的值）。

在 A/D 转换完成或中止后，需要等待 2 个 TAD 才能开始下一次采集。等待时间一到，将自动开始对选定通道进行采集。

注：	**不应**在开启 A/D 模块的指令中将 GO/$\overline{\text{DONE}}$ 位置 1。

图 23-6： A/D 转换 TAD 周期 （ACQT<2:0> = 000， TACQ = 0）

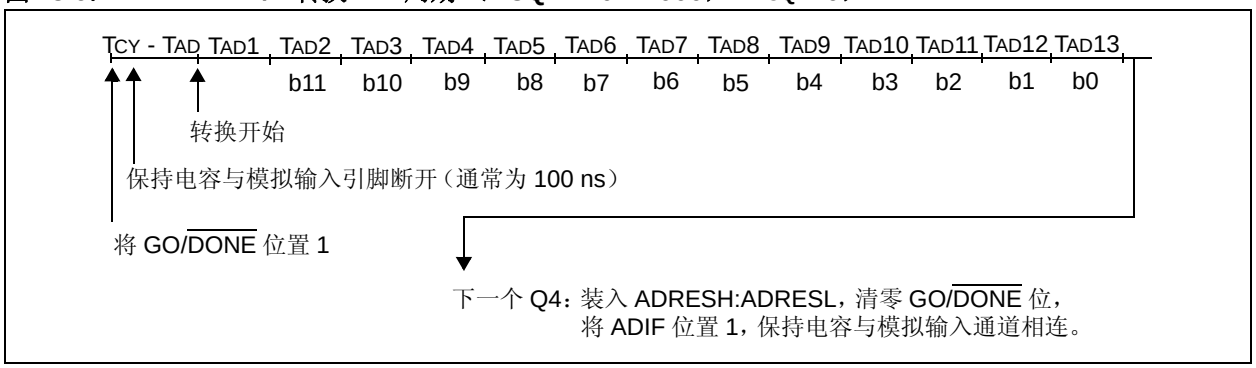
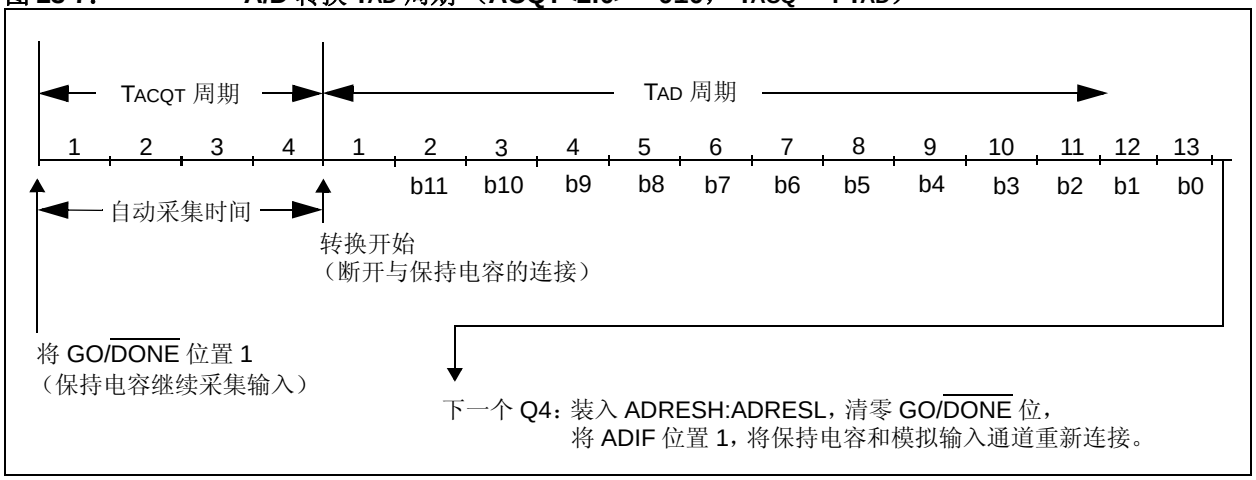


图 23-7： A/D 转换 TAD 周期 （ACQT<2:0> = 010， TACQ = 4 TAD）



23.8 特殊事件触发信号的使用

以下任一模块的特殊事件触发信号可以启动A/D转换：

- CCP2—— 需要将 CCP2M<3:0> 位（CCP2CON<3:0>）设置为 1011(†)
- ECCP1
- CTMU——需要将CTTRIG位（CTMUCONH<0>）置 1
- Timer1

要启动 A/D 转换：

- 必须使能 A/D 模块 （ADON = 1）
- 选择相应的模拟输入通道
- 通过以下方式之一设置最小采集周期：
  - 由用户提供的时序
  - 由相应 TACQ 时间所作的选择

满足以上条件时，触发信号将 GO/$\overline{\text{DONE}}$ 位置 1 并且开始 A/D 采集。

如果未使能 A/D 模块 （ADON = 0），则模块会忽略特殊事件触发信号。

注：	使用ECCP1或CCP2触发信号时，Timer1或 Timer3 被清零。定时器复位可自动重复 A/D 采集周期，最大限度地降低了软件开销 （将 ADRESH:ADRESL 内容传送到所需存储单元）。如果未使能 A/D 模块，则特殊事件触发信号将被模块忽略，但它仍会将定时器计数器复位。

23.9 在功耗管理模式下的操作

在功耗管理模式下，自动采集时间和 A/D 转换时钟的选择一定程度上可由时钟源和频率决定。

如果希望器件处于功耗管理模式时 A/D 继续工作，就应该根据要在功耗管理模式下使用的时钟对 ADCON2 中的 ACQT<2:0> 和 ADCS<2:0> 位进行更新。

在进入功耗管理模式后 （两种功耗管理运行模式之一），就可以开始 A/D 采集或转换。采集或转换开始以后，器件应继续使用与功耗管理模式相同的时钟源直到转换完成。如果需要，在转换期间也可以将器件置于相应的功耗管理空闲模式。

如果功耗管理模式的时钟频率小于 1 MHz，就应该选择 A/D RC 时钟源。

在休眠模式下工作需要选择 A/D RC 时钟。如果将 ACQT<2:0> 位设置为 000 并启动转换，转换将延时一个指令周期以允许执行 SLEEP 指令并进入休眠模式。OSCCON 寄存器中的 IDLEN 和 SCS<1:0> 位必须在启动转换之前被清零。

表 23-2： 与 A/D 模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSPIF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSPIE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSPIP	TMR1GIP	TMR2IP	TMR1IP
ADRESH	A/D 结果寄存器的高字节							
ADRESL	A/D 结果寄存器的低字节							
ADCON0	—	CHS4	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
ADCON1	TRIGSEL1	TRIGSEL0	VCFG1	VCFG0	VNCFG	CHSN2	CHSN1	CHSN0
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
ANCON1	—	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
PORTA	RA7(1)	RA6(1)	RA5	—	RA3	RA2	RA1	RA0
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
PORTE	RE7	RE6	RE5	RE4	RE3(2)	—	RE1	RE0
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	—	TRISE2	TRISE1	TRISE0
PMD1	PSPMD	CTMUMD	ADCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD

图注： — = 未实现，读为 0。 A/D 转换不使用阴影单元。

注 1： 这些位仅在某些振荡器模式下，在 FOSC2 配置位 = 0 时可用。如果该配置位被清零，则不实现该信号。

2： 当禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 置 1 时，该位未实现。

24.0 比较器模块

模拟比较器模块包含两个比较器，可以用多种方式对它们进行独立配置。输入可以从模拟输入和两个内部参考电压中选择。数字输出可从引脚电平读取，也可通过控制寄存器读取。此外，还提供多种输出和中断事件产生方式。图 24-1 给出了模块的通用单比较器的图示。

该模块的主要特性包括：

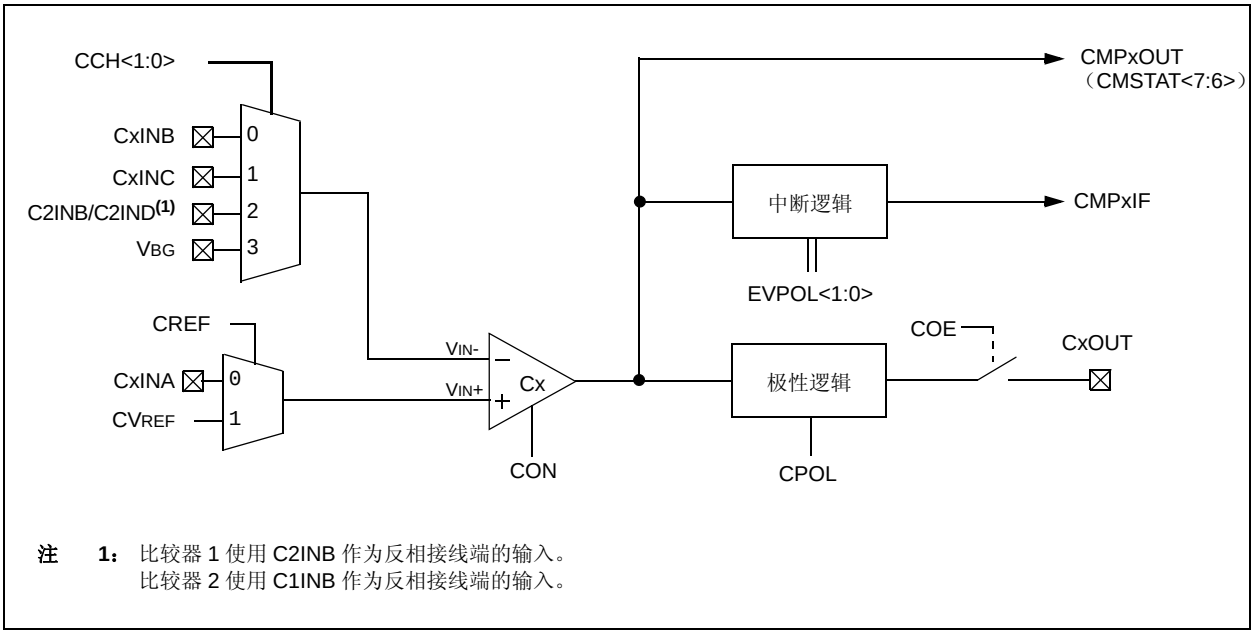
- 独立的比较器控制
- 可编程输入配置
- 输出到引脚和寄存器
- 可编程输出极性
- 每个比较器可独立产生中断，并可配置电平变化中断

24.1 寄存器

CMxCON 寄存器（CM1CON 和 CM2CON）用于选择每个比较器的输入和输出配置，以及中断产生的设置（见寄存器 24-1）。

CMSTAT 寄存器（寄存器 24-2）用于提供比较器的输出结果。该寄存器中的位是只读位。

图 24-1： 比较器的简化框图



注 1： 比较器 1 使用 C2INB 作为反相接线端的输入。
比较器 2 使用 C1INB 作为反相接线端的输入。

寄存器 24-1：　　CMxCON：比较器控制 x 寄存器

R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
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　**CON：**比较器使能位
1 = 使能比较器
0 = 禁止比较器

bit 6　**COE：**比较器输出使能位
1 = 比较器输出出现在 CxOUT 引脚上
0 = 比较器输出仅在内部有效

bit 5　**CPOL：**比较器输出极性选择位
1 = 比较器输出反相
0 = 比较器输出不反相

bit 4-3　**EVPOL<1:0>：**中断极性选择位
11 = 在输出发生任何变化时产生中断 **(1)**
10 = 仅在输出从高电平跳变为低电平时产生中断
01 = 仅在输出从低电平跳变为高电平时产生中断
00 = 禁止产生中断

bit 2　**CREF：**比较器参考电压选择位（同相输入）
1 = 同相输入连接至内部 CV_{REF} 电压
0 = 同相输入连接至 CxINA 引脚

bit 1-0　**CCH<1:0>：**比较器通道选择位
11 = 比较器的反相输入连接至 V_{BG}
10 = 比较器的反相输入连接至 C2INB 引脚 **(2)**
01 = 比较器的反相输入连接至 CxINC 引脚
00 = 比较器的反相输入连接至 C1INB 引脚 **(2)**

注　1： 每次选择该模式时， CMPxIF 会自动置 1，在进行初始配置之后，该位必须由应用程序清零。
2： 比较器 1 使用 C2INB 作为反相接线端的输入。比较器 2 使用 C1INB 作为反相接线端的输入。

寄存器 24-2：　　CMSTAT：比较器状态寄存器

R-x	R-x	U-0	U-0	U-0	U-0	U-0	U-0
CMP2OUT	CMP1OUT	—	—	—	—	—	—
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　**CMP2OUT:CMP1OUT：**比较器 x 状态位

如果 CPOL （CMxCON<5>） = 0 （极性不反相）：
1 = 比较器 x 的 $V_{IN+} > V_{IN-}$
0 = 比较器 x 的 $V_{IN+} < V_{IN-}$

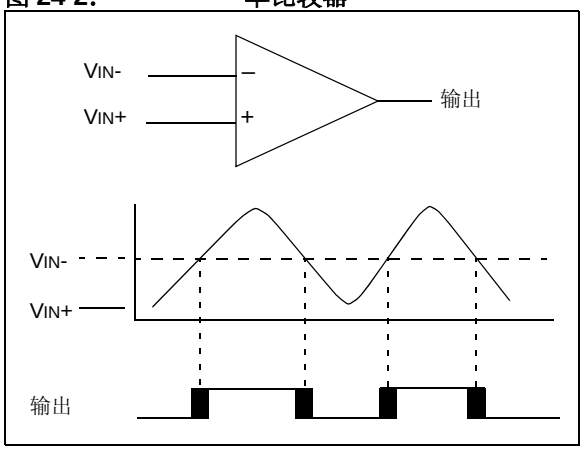
如果 CPOL = 1 （极性反相）：
1 = 比较器 x 的 $V_{IN+} < V_{IN-}$
0 = 比较器 x 的 $V_{IN+} > V_{IN-}$

bit 5-0　**未实现：**读为 0

24.2 比较器工作原理

图24-2所示为单比较器以及模拟输入电平与数字输出之间的关系。当 VIN+ 上的模拟输入电压小于 VIN- 上的模拟输入电压时，比较器输出为数字低电平。当 VIN+ 上的模拟输入电压大于 VIN- 上的模拟输入电压时，比较器输出为数字高电平。图 24-2 中比较器输出的黑色区域表示因输入失调和响应时间所造成的输出不确定区域。

图 24-2：　　单比较器



24.3 比较器响应时间

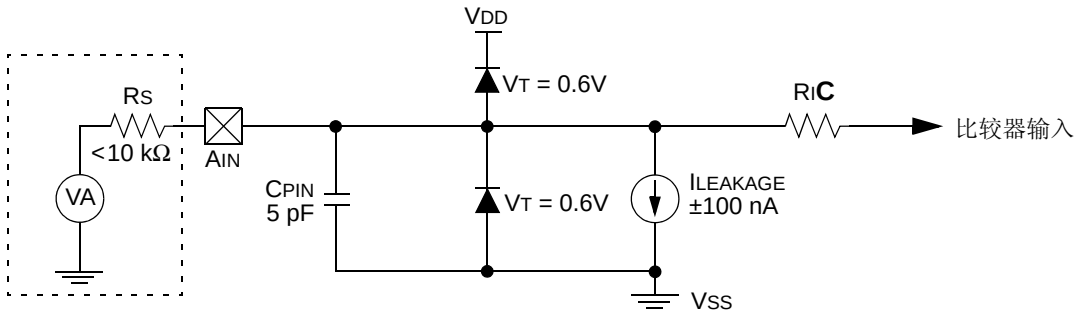
响应时间是指在选择新的参考电压或输入源后，比较器输出达到有效电平的最短时间。比较器的响应时间不同于参考电压的稳定时间。因此，在确定比较器输入改变的总响应时间时，必须同时考虑这两个时间。否则，应使用比较器的最大延时（见**第 31.0 节 “ 电气特性 ”**）。

24.4 模拟输入连接注意事项

模拟输入的简化电路如图 24-3 所示。由于模拟引脚被连接到数字输出端，它们在 VDD 和 VSS 之间连有反向偏置的二极管。因此，模拟输入必须在 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导通，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），要保证其泄漏电流极小。

图 24-3：　　比较器模拟输入模型



图注：

CPIN	=	输入电容
VT	=	门限电压
ILEAKAGE	=	由各连接点在引脚上产生的泄漏电流
RIC	=	片内走线等效电阻
RS	=	信号源阻抗
VA	=	模拟电压

24.5 比较器控制和配置

每个比较器具有最多 8 种可能的输入组合：最多 4 种外部模拟输入，以及两个内部参考电压之一。

所有比较器都允许选择来自引脚 CxINA 的信号，或者来自同相通道上的比较器参考电压（CVREF）。该输入与 C1INB、CxINC、C2INB 或反相通道上的单片机固定内部参考电压（VBG，标称值为 1.024V）进行比较。表 24-1 列出了与固定 I/O 引脚关联的比较器输入和输出。图 24-4 显示了可用的比较器配置及其相应的位设置。

表 24-1： 比较器输入和输出

比较器	输入或输出	I/O 引脚 (†)
1	C1INA （VIN+）	RB0/RD0
	C1INB （VIN-）	RB1/RD1
	C1INC （VIN-）	RA1
	C2INB （VIN-）	RA5/RD3
	C1OUT	RB2/RE1
2	C2INA （VIN+）	RB4/RD2
	C2INB （VIN-）	RA5/RD3
	C2INC （VIN-）	RA2
	C2OUT	RB3/RE2

† I/O 引脚取决于封装类型。

24.5.1 比较器使能和输入选择

将 CMxCON 寄存器的 CON 位 （CMxCON<7>）置 1 可以使能比较器操作。清零 CON 位可以禁止比较器，以使电流消耗降至最低。

CMxCON寄存器中的CCH<1:0>位（CMxCON<1:0>）指示 3 个模拟输入引脚之一或内部参考电压（VBG）连接到比较器 VIN-。根据不同的比较器工作模式，可选择使用外部或内部参考电压。

将 VIN- 上的模拟信号与 VIN+ 上的信号作比较，并相应地调整比较器的数字输出。

当 CREF（CMxCON<2>）= 0，且 VIN+ 连接到 CxINA 引脚时，将使用外部参考电压。使用外部参考电压时，比较器模块可以配置为使用外部参考电压源。参考电压信号必须在 VSS 和 VDD 之间，并且可被施加到比较器的任一引脚上。

比较器模块也可以选择使用内部比较器参考电压模块产生的参考电压 （CVREF）。在**第 25.0 节 “ 比较器参考电压模块 ”** 中详细介绍了该模块。仅当 CREF = 1 时，来自比较器参考电压模块的参考电压才可用。在该模式下，内部参考电压被施加到比较器的 VIN+ 引脚上。

注：	通过 CCH<1:0> 选择的比较器输入引脚必须配置为输入，即将 ANCONx 寄存器中的相应 TRIS 位和 ANSELx 位置 1。

24.5.2 比较器使能和输出选择

通过 CMSTAT 寄存器可读取比较器输出。CMSTAT<6> 位读取比较器 1 的输出，CMSTAT<7> 读取比较器 2 的输出。这两位是只读位。

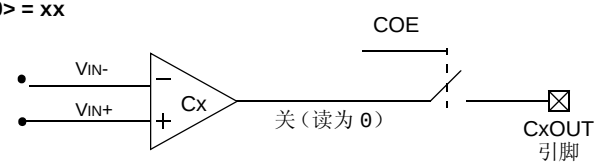
通过将 COE 位（CMxCON<6>）置 1，比较器输出也可以直接输出到 RE2 和 RE1 引脚。在使能时，引脚输出路径中的多路开关将切换到比较器输出。处于该模式时， TRISE<2:1> 位将仍然用作 RE2 和 RE1 引脚的数字输出使能位。

默认情况下，每当 VIN+ 上的电压高于 VIN- 上的电压时，比较器的输出为逻辑高电平。比较器输出的极性可以使用 CPOL 位 （CMxCON<5>）进行反相。

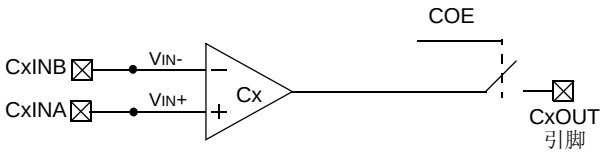
每个比较器输出的不确定区域的大小与规范中给出的输入失调电压和响应时间有关，如**第 24.2 节 “ 比较器工作原理 ”** 中所讨论。

图 24-4：　　比较器配置

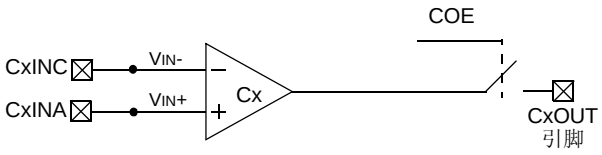
比较器关闭
CON = 0，CREF = x，CCH<1:0> = xx



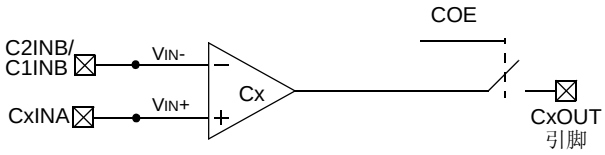
比较器 CxINB > CxINA 比较
CON = 1，CREF = 0，CCH<1:0> = 00



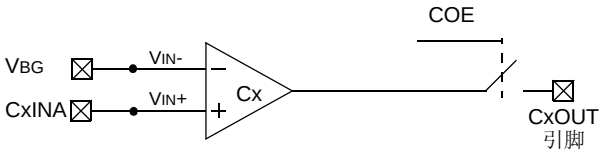
比较器 CxINC > CxINA 比较
CON = 1，CREF = 0，CCH<1:0> = 01



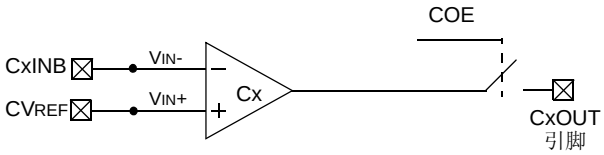
比较器 C2INB/C1INB > CxINA 比较
CON = 1，CREF = 0，CCH<1:0> = 10



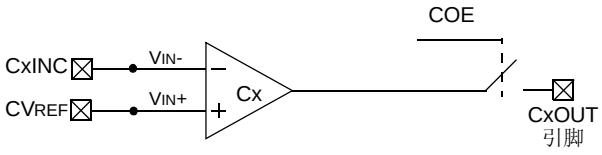
比较器 VBG > CxINA 比较
CON = 1，CREF = 0，CCH<1:0> = 11



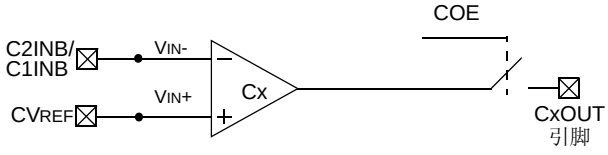
比较器 CxINB > CVREF 比较
CON = 1，CREF = 1，CCH<1:0> = 00



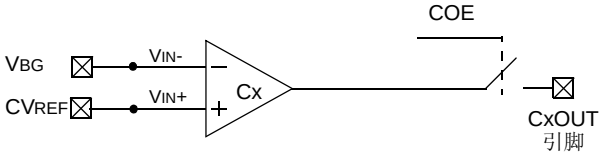
比较器 CxINC > CVREF 比较
CON = 1，CREF = 1，CCH<1:0> = 01



比较器 C2INB/C1INB > CVREF 比较
CON = 1，CREF = 1，CCH<1:0> = 10



比较器 VBG > CVREF 比较
CON = 1，CREF = 1，CCH<1:0> = 11



注　1： VBG 是内部参考电压 （见表 31-2）。

24.6 比较器中断

每当发生以下任意事件时，比较器中断标志都将被置 1：

- 比较器输出从低电平跳变为高电平
- 比较器输出从高电平跳变为低电平
- 比较器输出发生任何变化

比较器中断的产生方式通过 CMxCON 寄存器中的 EVPOL<1:0> 位 （CMxCON<4:3>）选择。

为了提供最大的灵活性，比较器的输出可以使用 CMxCON 寄存器中的 CPOL 位（CMxCON<5>）进行反相。这在功能上等效于对于特定模式颠倒比较器的反相和同相输入。

在比较器输出从低电平跳变为高电平或从高电平跳变为低电平时产生中断。这种中断产生模式依赖于 CMxCON 寄存器中的 EVPOL<1:0> 位。当 EVPOL<1:0> = 01 或 10 时，在比较器输出从低电平跳变为高电平或从高电平跳变为低电平时产生中断。在产生中断后，需要用软件清零中断标志。

当 EVPOL<1:0> = 11 时，任一比较器的输出值发生变化，都会将该比较器的中断标志位置 1。需要用软件保存输出位的状态信息（从 CMSTAT<7:6> 读取），以确定实际发生的变化。

CMPxIF<2:0> （PIR4<5:4）位是比较器中断标志。CMPxIF 位必须通过清零复位。由于可以向该寄存器写入 1，因此可以产生模拟中断。表 24-2 列出了对应于比较器输入电压和 EVPOL 位设置的中断产生情况。

必须将 CMPxIE 位（PIE4<5:4>）和 PEIE 位（INTCON<6>）置 1 以允许中断。此外，还必须将 GIE 位 （INTCON<7>）置 1。如果这些位中的任何一个被清零，将无法允许中断，尽管中断条件发生时仍会将 CMPxIF 位置 1。

图 24-3 给出了中断部分的简化框图。

注： 当 EVPOL<1:0> = 00 时，CMPxIF 不会置 1。

表 24-2： 比较器中断产生

CPOL	EVPOL<1:0>	比较器输入变化	CxOUT 跳变	中断产生
0	00	VIN+ > VIN-	低电平到高电平	否
		VIN+ < VIN-	高电平到低电平	否
	01	VIN+ > VIN-	低电平到高电平	是
		VIN+ < VIN-	高电平到低电平	否
	10	VIN+ > VIN-	低电平到高电平	否
		VIN+ < VIN-	高电平到低电平	是
	11	VIN+ > VIN-	低电平到高电平	是
		VIN+ < VIN-	高电平到低电平	是
1	00	VIN+ > VIN-	高电平到低电平	否
		VIN+ < VIN-	低电平到高电平	否
	01	VIN+ > VIN-	高电平到低电平	否
		VIN+ < VIN-	低电平到高电平	是
	10	VIN+ > VIN-	高电平到低电平	是
		VIN+ < VIN-	低电平到高电平	否
	11	VIN+ > VIN-	高电平到低电平	是
		VIN+ < VIN-	低电平到高电平	是

24.7 休眠期间的比较器操作

当比较器处于工作状态而器件处于休眠模式时，比较器仍保持工作状态并可产生中断（如果中断被允许）。中断会将器件从休眠模式唤醒。每个处于工作状态的比较器都将额外消耗电流。

若要将休眠模式下的功耗降至最低，可在进入休眠模式前关闭比较器（CON = 0）。如果器件从休眠模式唤醒，CMxCON 寄存器的内容不受影响。

24.8 复位的影响

器件复位强制 CMxCON 寄存器为复位状态。这使比较器和参考电压被强制为 “ 关闭 ” 状态。

表 24-3： 与比较器模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CVRCON	CVREN	CVROE	CVRSS	CVR4	CVR3	CVR2	CVR1	CVR0
CMSTAT	CMP2OUT	CMP1OUT	—	—	—	—	—	—
PIR4	TMR4IF	EEIF	CMP2IF	CMP1IF	—	CCP5IF	CCP4IF	CCP3IF
PIE4	TMR4IE	EEIE	CMP2IE	CMP1IE	—	CCP5IE	CCP4IE	CCP3IE
IPR4	TMR4IP	EEIP	CMP2IP	CMP1IP	—	CCP5IP	CCP4IP	CCP3IP
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
ANCON1	—	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
PMD2	—	—	—	—	MODMD	ECANMD	CMP2MD	CMP1MD

图注： — = 未实现，读为 0。

25.0 比较器参考电压模块

比较器参考电压模块是一个 32 阶的梯形电阻网络，可提供多个参考电压供选择。虽然它的主要目的是为模拟比较器提供参考电压，但也可将它用于其他场合。

图 25-1 给出了该模块的框图。梯形电阻网络经过分段可提供一系列 CVREF 值，并且该网络还具有断电功能，可以在不使用参考电压的情况下节省功耗。器件的 VDD/VSS 或外部参考电压都可以作为该模块的参考电压源。

25.1 配置比较器参考电压

比较器参考电压模块是通过 CVRCON 寄存器（寄存器 25-1）来控制的。比较器参考电压模块提供一系列输出电压，具有 32 个不同的电压。

CVR<4:0> 选择位（CVRCON<4:0>）用于提供一系列输出电压。公式 25-1 显示了如何计算比较器参考电压。

公式 25-1：

如果 CVRSS = 1：

$$CV_{REF} = \left(V_{REF-} + \frac{CVR<4:0>}{32}\right) \bullet (V_{REF+} - V_{REF-})$$

如果 CVRSS = 0：

$$CV_{REF} = \left(AV_{SS} + \frac{CVR<4:0>}{32}\right) \bullet (AV_{DD} - AV_{SS})$$

比较器参考电压模块的电压源可以来自 VDD 和 VSS，也可以来自与 RA3 和 RA2 复用的外部 VREF+ 和 VREF-。电压源通过 CVRSS 位（CVRCON<5>）选择。

在更改 CVREF 输出值时，必须考虑比较器参考电压的稳定时间（见**第 31.0 节“电气特性”**中的表 31-2）。

寄存器 25-1： CVRCON：比较器参考电压控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE	CVRSS	CVR4	CVR3	CVR2	CVR1	CVR0
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

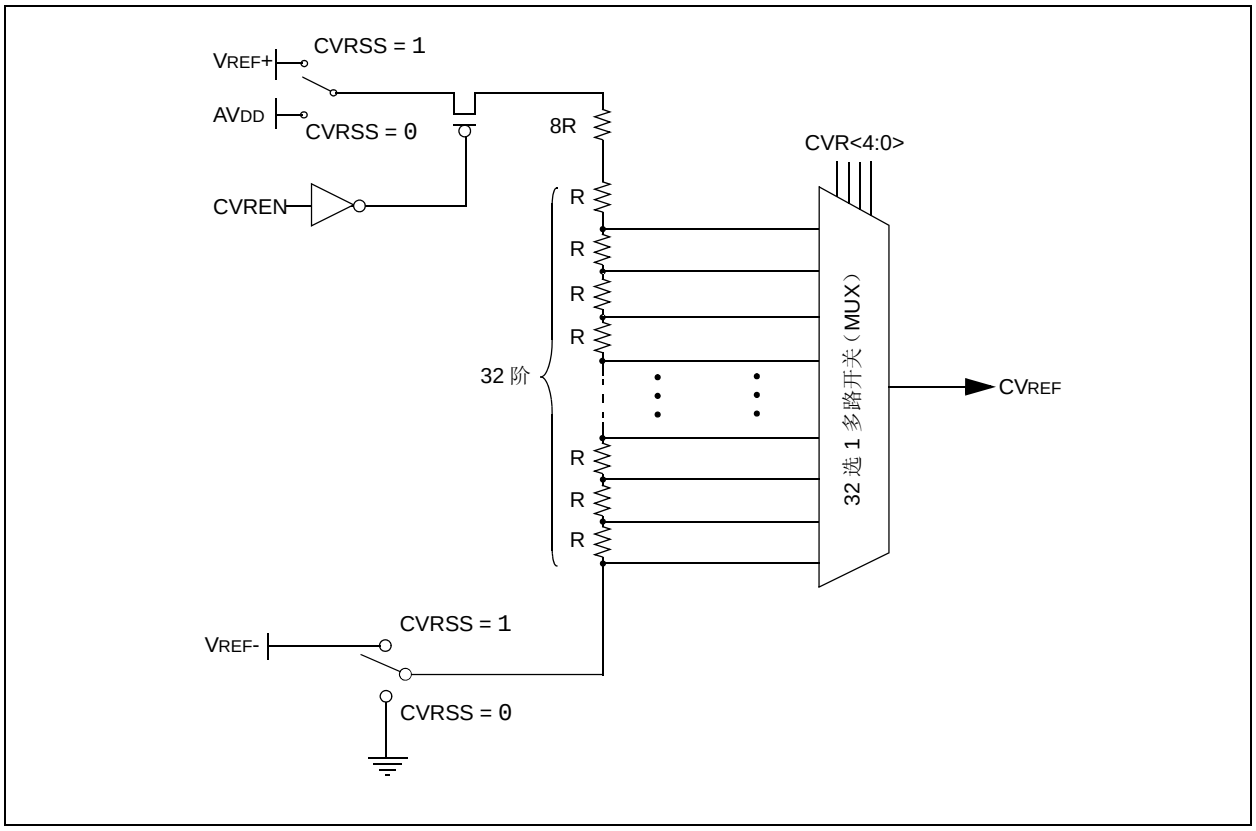
bit 7 **CVREN：**比较器参考电压使能位
1 = CVREF 电路上电
0 = CVREF 电路断电

bit 6 **CVROE：**比较器 VREF 输出使能位
1 = CVREF 电压从 CVREF 引脚输出
0 = CVREF 电压从 CVREF 引脚断开

bit 5 **CVRSS：**比较器 VREF 源选择位
1 = 比较器参考电压源， CVRSRC = VREF+ – VREF-
0 = 比较器参考电压源， CVRSRC = AVDD – AVSS

bit 4-0 **CVR<4:0>：**比较器 VREF 值选择位（0 ≤ CVR<4:0> ≤ 31）
当 CVRSS = 1 时：
CVREF = (VREF-) + (CVR<4:0>/32) • (VREF+ – VREF-)
当 CVRSS = 0 时：
CVREF = (AVSS) + (CVR<4:0>/32) • (AVDD – AVSS)

图 25-1：　　比较器参考电压模块框图



25.2　参考电压精度 / 误差

由于模块结构的限制，并不能实现整个参考电压范围的满量程输出。梯形电阻网络顶部和底部的晶体管（图 25-1）使 CVREF 值不能达到参考电压源的满幅值。参考电压是由参考电压源分压而来的，因此 CVREF 输出随参考电压源的波动而变化。经过测试的参考电压的绝对精度，请参见**第 31.0 节“电气特性”**。

25.3　休眠期间的操作

如果因中断或看门狗定时器超时将器件从休眠模式唤醒，CVRCON 寄存器的内容将不受影响。为了降低休眠模式下的电流消耗，应禁止参考电压模块。

25.4　复位的影响

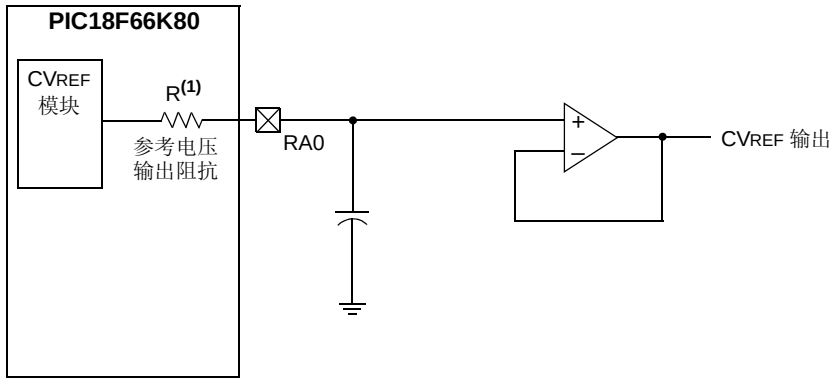
器件复位时，CVREN 位（CVRCON<7>）将被清零从而禁止参考电压模块。复位还将 CVROE 位（CVRCON<6>）清零，使参考电压从 RF5 引脚断开。

25.5　连接注意事项

参考电压模块的工作独立于比较器模块。如果 CVROE 位被置 1，那么参考电压发生器的输出可与 RA0 引脚相连。当 RA0 被配置为数字输入引脚时，将参考电压输出连接到 RA0 引脚，将会增加电流消耗。CVRSS 使能时，RA0 被配置为数字输出引脚也将增加电流消耗。

RA0 引脚可被用作简单的 D/A 输出，但是其驱动能力有限。要提高电流驱动能力，VREF 参考电压输出端必须外接缓冲器。图 25-2 举例说明了这一缓冲技术。

图 25-2：　　参考电压输出缓冲示例



注　　1： R 的值取决于参考电压配置位 CVRCON<3:0> 和 CVRCON<5>。

表 25-1：　　与比较器参考电压相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CVRCON	CVREN	CVROE	CVRSS	CVR4	CVR3	CVR2	CVR1	CVR0
CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
TRISA	TRISA7	TRISA6	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0

图注：　　— = 未实现，读为 0。比较器参考电压不使用阴影单元。

注：

26.0 高 / 低压检测 （HLVD）

PIC18F66K80 系列器件具有一个高 / 低压检测 （High/Low-Voltage Detect， HLVD）模块。该模块是一个可编程的电路，可以设置器件的电压跳变点和变化方向。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，就会将中断标志位置 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。

高 / 低压检测控制寄存器（寄存器 26-1）完全控制 HLVD 模块的工作。用户可通过软件控制该寄存器将电路 “关闭 ”，从而使器件的电流消耗降至最低。

图 26-1 给出了模块的框图。

寄存器 26-1： HLVDCON：高 / 低压检测控制寄存器

R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3(1)	HLVDL2(1)	HLVDL1(1)	HLVDL0(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 **VDIRMAG：**电压方向大小选择位
1 = 当电压等于或超过跳变点 （HLVDL<3:0>）时，事件发生
0 = 当电压等于或低于跳变点 （HLVDL<3:0>）时，事件发生

bit 6 **BGVST：**带隙参考电压稳定状态标志位
1 = 内部带隙参考电压稳定
0 = 内部带隙参考电压不稳定

bit 5 **IRVST：**内部参考电压稳定标志位
1 = 指示电压检测逻辑在检测到指定的电压范围时，产生中断标志
0 = 指示电压检测逻辑在检测到指定的电压范围时，不会产生中断标志，并且 HLVD 中断不被允许

bit 4 **HLVDEN：**高 / 低压检测模块使能位
1 = 使能 HLVD
0 = 禁止 HLVD

bit 3-0 **HLVDL<3:0>：**电压检测限制位 (1)
1111 = 使用外部模拟输入 （输入来自于 HLVDIN 引脚）
1110 = 最大设置
.
.
.
0000 = 最小设置

注 1： 关于电气规范，请参见**第 31.0 节 “ 电气特性 ”** 中的参数 D042。

通过将 HLVDEN 位 （HLVDCON<4>）置 1 使能该模块。每次使能 HLVD 模块时，电路需要一定时间才能稳定下来。 IRVST 位 （HLVDCON<5>）是一个只读位，用于指示电路何时稳定。仅当电路稳定且 IRVST 位置 1 后，该模块才能产生中断。

VDIRMAG 位 （HLVDCON<7>）决定该模块的整体工作状态。当 VDIRMAG 清零时，模块监视 VDD 看它是否降到预先确定的设置点以下。当该位置 1 时，模块监视 VDD 看它是否上升到设置点以上。

26.1 工作原理

当使能了 HLVD 模块时，比较器使用内部产生的参考电压作为设置点。将设置点的电压与跳变点电压作比较，其中电阻分压器中的每个节点均代表一个电压跳变点。

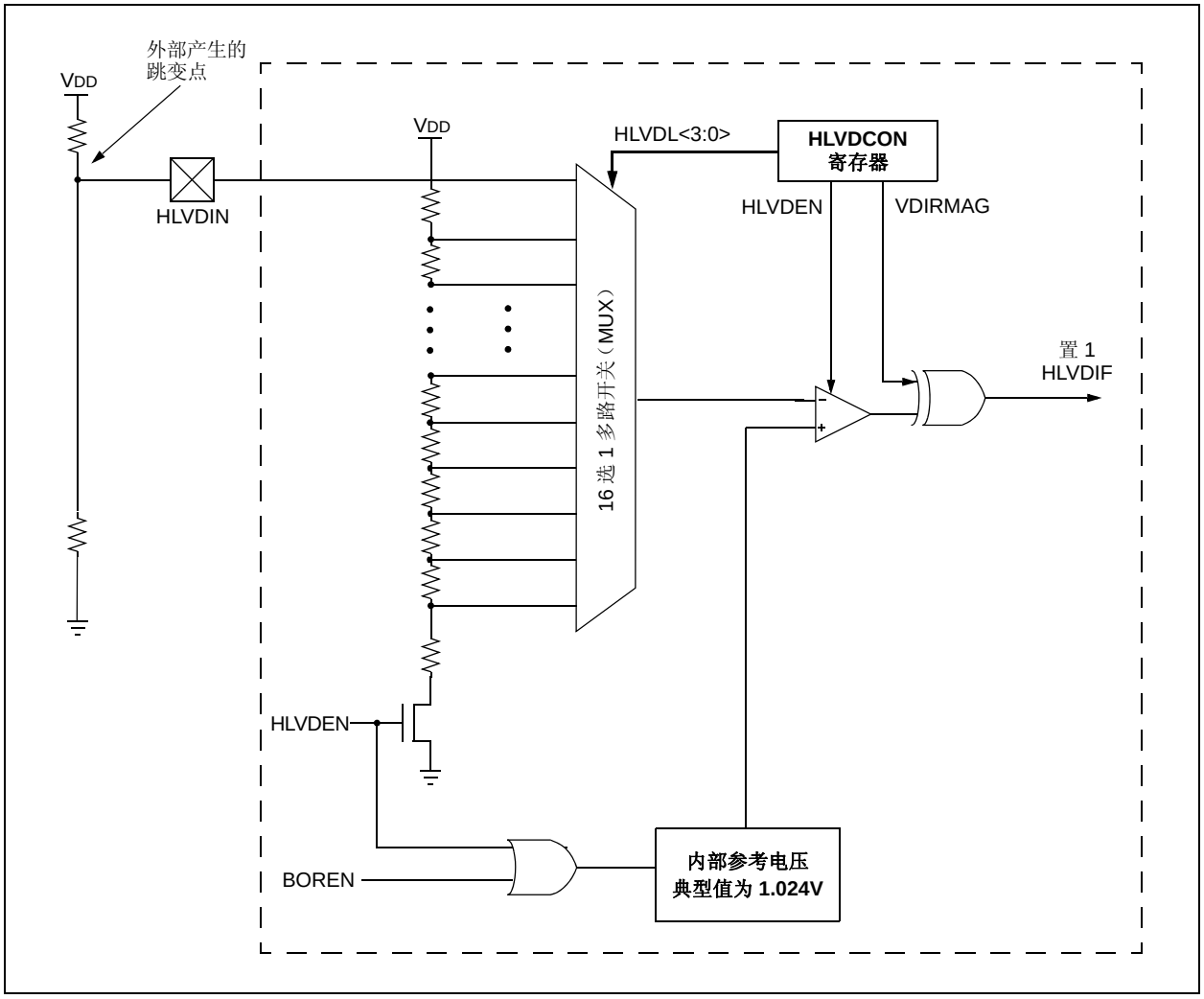
“跳变点”电压是被检测到的高压或低压事件的电平，它取决于该模块的配置。

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。

可用软件设定跳变点电压为 16 个值中的任何一个。通过对 HLVDL<3:0> 位（HLVDCON<3:0>）进行编程可以选择跳变点。

HLVD 模块还有一个额外的功能，允许用户通过外部电源向模块提供跳变电压。当 HLVDL<3:0> 位被设置为 1111 时，使能该模式。在此状态下，比较器输入与外部输入引脚 HLVDIN 复用。因此用户可以灵活地配置高 / 低压检测中断，使之可以在有效工作范围内的任何电压点上产生。

图 26-1： HLVD 模块框图 （带外部输入）



26.2 HLVD 设置

设置 HLVD 模块：

1. 通过将值写入HLVDL<3:0>位，选择所需的HLVD跳变点。
2. 将 VDIRMAG 位设置为检测高压（VDIRMAG = 1）或低压 （VDIRMAG = 0）。
3. 通过将 HLVDEN 位置 1，使能 HLVD 模块。
4. 清零 HLVD 中断标志位 （PIR2<2>），该位可能被上次中断置 1。
5. 如果需要中断，通过将 HLVDIE 和 GIE 位 （分别为 PIE2<2> 和 INTCON<7>）置 1，允许 HLVD 中断。

   直到 IRVST 位也置 1 时才会发生中断。

注： 在更改任何模块设置（VDIRMAG 和 HLVDL <3:0>）之前，请先禁止模块（HLVDEN = 0），然后再进行更改并重新使能模块。这可以防止产生虚假 HLVD 事件。

26.3 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。电气规范中的参数 D022B（ΔIhlvd）（表 31-11）规定了使能该模块时的电流总消耗。

HLVD 模块无需一直工作，工作与否取决于具体的应用。要降低电流消耗，只需要在检测电压时，短时间地使能 HLVD 电路。在检测完成之后可以禁止该模块。

26.4 HLVD 启动时间

电气规范中的参数 37 （**第 31.0 节 “ 电气特性 ”**）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以后才能可靠地检测低压或高压条件。HLVD 启动时间 TIRVST 与器件时钟速度无关。它由电气规范中的参数 37（表 31-11）规定。

直到 TIRVST 结束并且参考电压达到稳定后才会允许 HLVD 中断标志。基于此原因，在此时间间隔期间，超出设置点的短暂偏离可能不会被检测到 （见图 26-2 或图 26-3）。

图 26-2：　　低压检测工作原理 （VDIRMAG = 0）

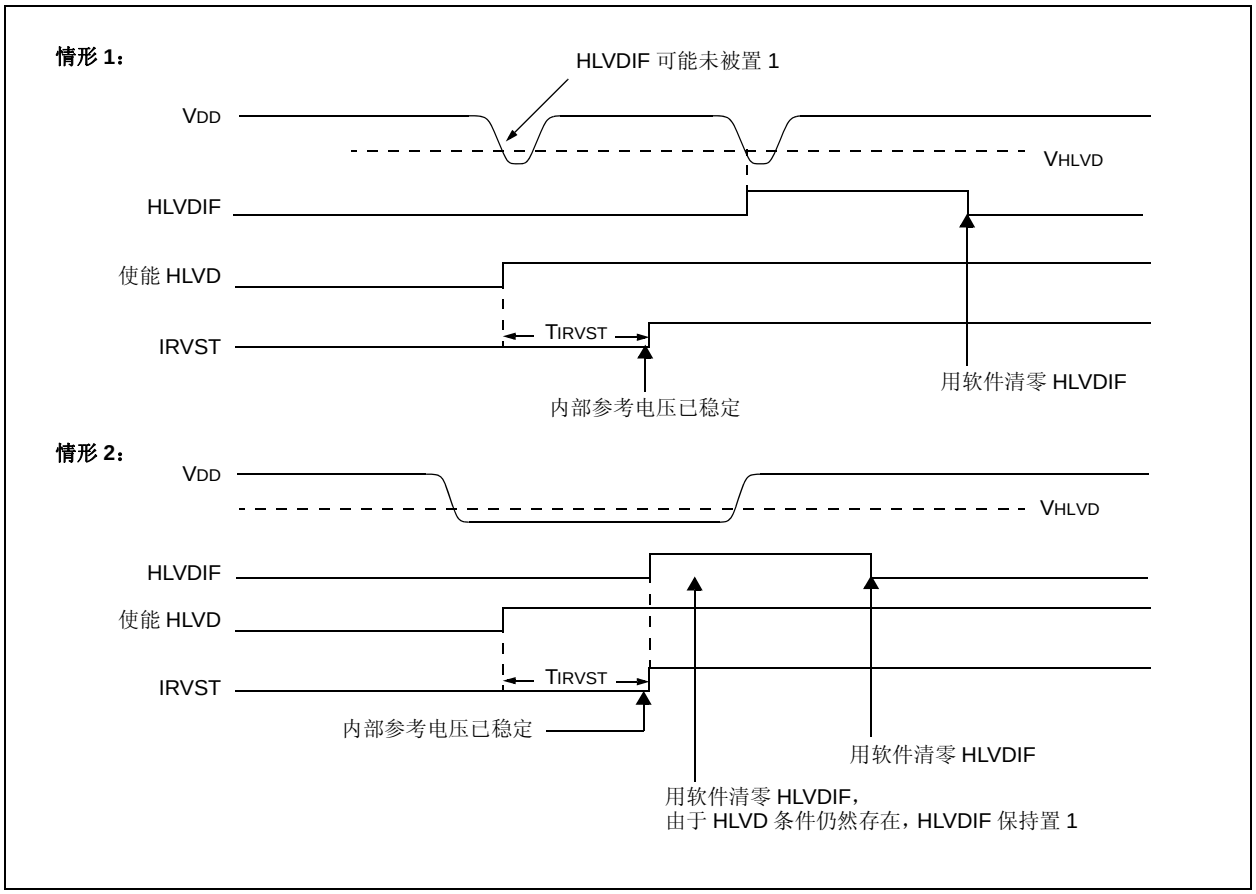
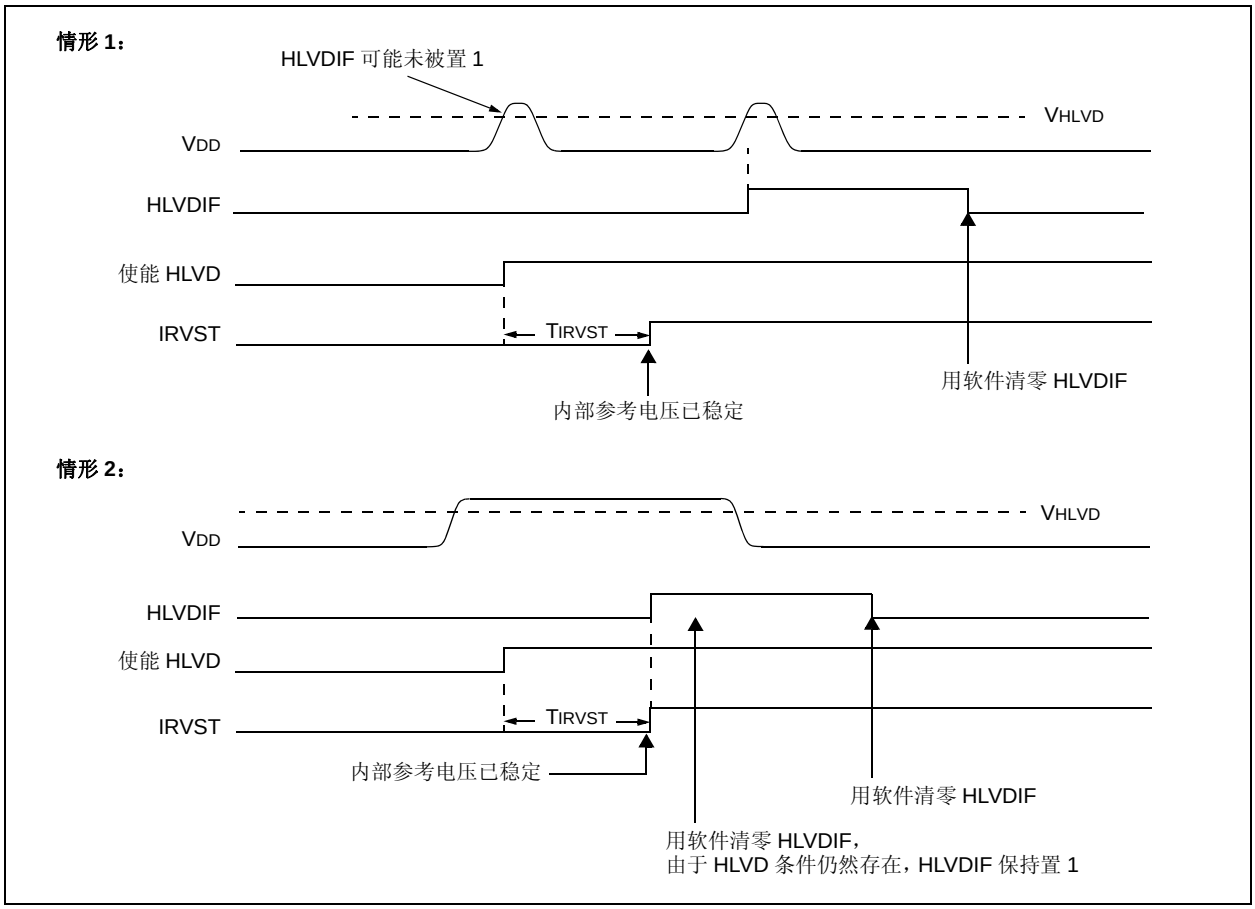


图 26-3：　　高压检测工作原理 （VDIRMAG = 1）

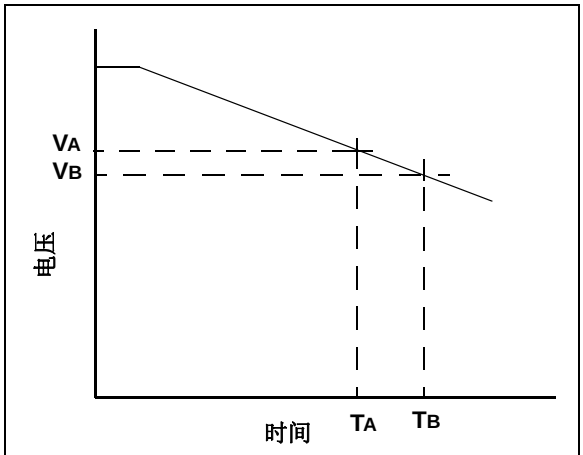


26.5　应用

在许多应用中，当电压低于或高于某个门限值时，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）。这里假设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 至 5V 的高压 （USB 上的电压）；如果断开连接，情况正好相反。此功能可以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 26-4 给出了一个近似的电压曲线。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 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 26-4：　　典型低压检测应用



图注： VA = HLVD 跳变点
VB = 器件的最低有效工作电压

26.6 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。

26.7 复位的影响

器件复位将强制所有寄存器为复位状态。这会强制关闭 HLVD 模块。

表 26-1： 与高 / 低压检测模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
HLVDCON	VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR2	OSCFIF	—	—	—	BCLIF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	—	—	BCLIE	HLVDIE	TMR3IE	TMR3GIE
IPR2	OSCFIP	—	—	—	BCLIP	HLVDIP	TMR3IP	TMR3GIP
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	—	TRISA3	TRISA2	TRISA1	TRISA0

图注： — = 未实现，读为 0。 HLVD 模块不使用阴影单元。

注 1： 根据不同的主振荡器模式设置相应的方向位将 PORTA<7:6> 单独配置为端口引脚。当被禁止时，这些位读为 0。

27.0 ECAN 模块

PIC18F66K80 系列器件包含一个增强型控制器局域网（Enhanced Controller Area Network，ECAN）模块。ECAN 模块完全向后兼容 PIC18CXX8 和 PIC18FXX8 器件中提供的CAN模块和PIC18FXX80器件中的ECAN模块。

控制器局域网（CAN）模块是一个串行接口，用于同其他外设或单片机器件进行通信。此接口或协议是针对允许在噪声环境下通信而设计的。

ECAN 模块是一个通信控制器，实现了 BOSCH 规范中定义的 CAN 2.0A 或 B 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。该模块实现了一种完整的 CAN 系统；但是本数据手册不讨论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：

- 实现了 CAN 协议 CAN 1.2、CAN 2.0A 和 CAN 2.0B
- 支持 DeviceNet™ 数据字节过滤器
- 支持标准数据帧和扩展数据帧
- 0-8 字节数据长度
- 最高 1 Mb/s 的可编程比特率
- 完全向后兼容 PIC18XXX8 CAN 模块
- 三种工作模式：
  - 模式 0—— 传统模式
  - 模式 1—— 带有 DeviceNet 支持的增强型传统模式
  - 模式 2—— 带有 DeviceNet 支持的 FIFO 模式
- 支持远程帧，具有自动处理功能
- 双重缓冲接收器，具有两个带优先级的接收报文存储缓冲区
- 6 个缓冲区，可设定为 RX 和 TX 报文缓冲区
- 16 个完全（标准 / 扩展标识符）接收过滤器，可与 4 个屏蔽器中的任意一个配合使用
- 2 个可分配给任意过滤器的完全接收过滤器屏蔽器
- 1 个可用作接收过滤器或接收过滤器屏蔽器的完全接收过滤器
- 3 个专用发送缓冲区，具有应用指定的优先级和中止功能
- 集成了低通滤波器的可编程唤醒功能
- 支持自检操作的可编程环回模式
- 通过中断功能在出现任何 CAN 接收器和发送器错误状态时发出中断信号
- 可编程时钟源
- 可编程为使用定时器模块，以进行时间标记和网络同步
- 低功耗休眠模式

27.1 模块概述

CAN 总线模块由协议引擎与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测状态和错误。将对在 CAN 总线上检测到的任何报文进行错误检测，并随后将其与过滤器进行比较以判断是否要将其接收并存储到两个接收寄存器之一中。

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

CAN 模块使用 RB2/CANTX 和 RB3/CANRX 引脚来与 CAN 总线接口。通过将 CANMX（CONFIG3H<0>）配置位置 1，可以将 CANTX 和 CANRX 引脚置于备用 I/O 引脚上。

对于 PIC18F2XK80 和 PIC18F4XK80，备用引脚位置为 RC6/CANTX 和 RC7/CANRX。对于 PIC18F6XK80，备用引脚位置为 RE4/CANRX 和 RE5/CANTX。

在正常模式下， CAN 模块会自动改写 CANTX 的相应 TRIS 位。用户必须确保 CANRX 的相应 TRIS 位置 1。

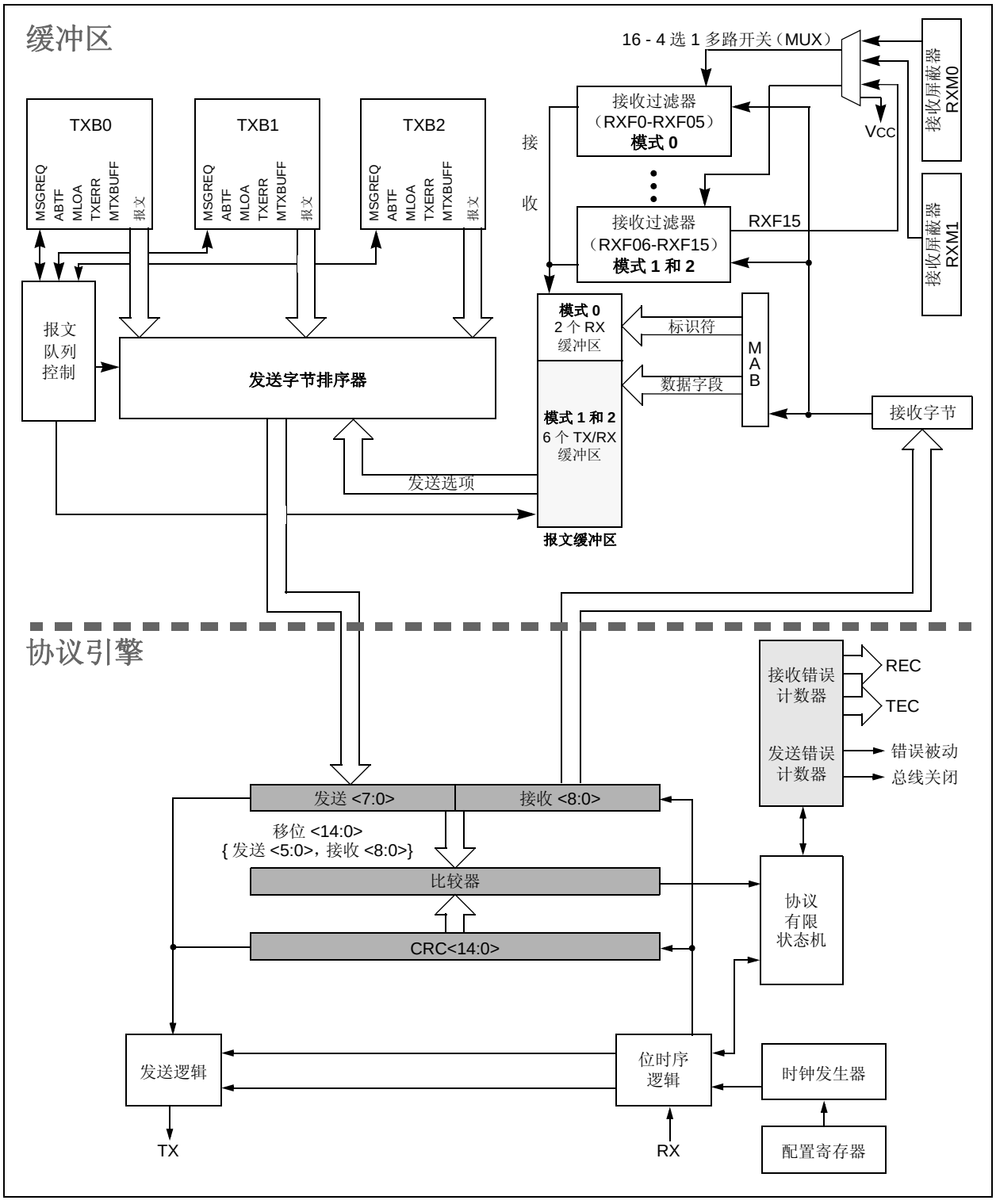
27.1.1 模块功能

CAN 总线模块由协议引擎与报文缓冲和控制模块组成（见图 27-1）。通过定义要由模块发送和接收的数据帧类型，可以最好地理解协议引擎。

以下序列说明了在使用ECAN模块来发送或接收报文之前，必须执行的初始化步骤。根据应用的需求，可以增加或删除一些步骤。

1. 初始化 RX 和 TX CAN 的 LAT 和 TRIS 位。
2. 确保 ECAN 模块处于配置模式。
3. 选择 ECAN 工作模式。
4. 设置波特率寄存器。
5. 设置过滤器和屏蔽寄存器。
6. 将ECAN模块设置为正常模式或应用逻辑所需的任何其他模式。

图 27-1：　　CAN 缓冲区和协议引擎框图



27.2 CAN 模块寄存器

注：	并非所有CAN寄存器都可在快速操作存储区中访问。

有许多与 CAN 模块相关的控制和数据寄存器。为方便起见，本文将它们划分到几节中进行介绍：

- 控制和状态寄存器
- 专用发送缓冲寄存器
- 专用接收缓冲寄存器
- 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

以下几节将详细介绍每个寄存器及其用法。

27.2.1 CAN 控制和状态寄存器

本节介绍的寄存器用于控制 CAN 模块的整体操作，以及显示其工作状态。

寄存器 27-1：　　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：			
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：**中止所有等待发送的位

1 = 中止所有等待发送的数据 （在所有发送缓冲区中）(1)
0 = 继续正常发送

bit 3-1　模式 0：

WIN<2:0>：窗口地址位

这些位用于选择要将哪些 CAN 缓冲区切换到快速操作存储区中。这样将可以从任意数据存储区中访问缓冲寄存器。在某个帧产生中断之后，可以通过将 ICODE<3:0> 位复制到 WIN<2:0> 位来选择正确的缓冲区。代码示例请参见例 27-2。

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：　该位将在所有数据发送都中止之后清零。

寄存器 27-2：　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 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 = 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 3-1,4-0　模式 0：
ICODE<2:0>：中断编码位
在发生中断时，会有一个经过优先级编码的中断值送入这些位中。该编码可以指示中断源。通过将 ICODE<3:1> 复制到 WIN<3:0>（模式 0）或将 EICODE<4:0> 复制到 EWIN<4:0>（模式 1 和 2）中，可以选择要映射到快速操作存储区中的正确缓冲区。代码示例请参见例 27-2。为了简化说明，下表列出了全部 5 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	00010	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 2：
EICODE<4:0>：中断编码位
请参见上面的 ICODE<3:1>。

注　1：为了最大程度地节能和 / 或能够在发生 CAN 总线活动时唤醒，在将器件置为休眠模式之前，需要先将 CAN 模块切换为禁止 / 休眠模式。
2：如果缓冲区被配置为接收器，则在发生中断时 EICODE 位将包含 10000。

例 27-1：　　更改为配置模式

```
    ; 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 27-2：　　在中断服务程序中通过使用 WIN 和 ICODE 位来访问 TX/RX 缓冲区

```
    ; 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 27-2：　在中断服务程序中通过使用 WIN 和 ICODE 位来访问 TX/RX 缓冲区（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 27-3： 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 （模式 0，默认）
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 高水位线位 (2)
1 = 将在剩余 1 个接收缓冲区时产生 FIFO 中断
0 = 将在剩余 4 个接收缓冲区时产生 FIFO 中断 (3)

bit 4-0 **EWIN<4:0>：**增强型窗口地址位
这些位用于将由 16 个分区的 CAN SFR 组成的寄存器组映射到快速操作存储区地址 0F60-0F6Dh。要映射的确切的寄存器组由这些位的二进制值决定。

模式 0：
未实现：读为 0

模式 1 和 2：
00000 = 接收过滤器 0、 1 和 2 以及 BRGCON2 和 BRGCON3
00001 = 接收过滤器 3、 4 和 5 以及 BRGCON1 和 CIOCON
00010 = 接收过滤器屏蔽器、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 = 接收过滤器 6、 7 和 8
00111 = 接收过滤器 9、 10 和 11
01000 = 接收过滤器 12、 13 和 14
01001 = 接收过滤器 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： 这些位只能在配置模式下进行更改。要切换到配置模式，请参见寄存器 27-1。
2： 该位仅在模式 2 下使用。
3： 如果 FIFO 配置为包含 4 个或更少的缓冲区，则会触发 FIFO 中断。

寄存器 27-4：　　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：			
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 已溢出
0 = 接收缓冲区 0 未溢出
模式 1：
未实现：读为 0
模式 2：
$\overline{\text{FIFOEMPTY}}$：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 已溢出
0 = 接收缓冲区 1 未溢出
模式 1 和 2：
RXBnOVFL：接收缓冲区 n 溢出位
1 = 接收缓冲区 n 已溢出
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 位的标志位。
1 = RXWARN 或 TXWARN 位置 1
0 = RXWARN 和 TXWARN 位均未置 1

27.2.2 专用 CAN 发送缓冲寄存器

本节介绍专用 CAN 发送缓冲寄存器及其相关的控制寄存器。

寄存器 27-5： TXBnCON：发送缓冲区 n 控制寄存器 [0 ≤ n ≤ 2]

模式 0	U-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)

模式 1 和 2	R/C-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发送，可以向其中重新装入数据
0 = 发送缓冲区尚未完成报文发送

bit 6 **TXABT**：发送中止状态位 (1)
1 = 中止报文
0 = 未中止报文

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 = 报文发送时发生总线错误
0 = 报文发送时未发生总线错误

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 = 在报文成功发送后自动清零

bit 2 **未实现**：读为 0

bit 1-0 **TXPRI<1:0>**：发送优先级位 (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，发送缓冲寄存器将保持只读。在该位置 1 时用软件清零该位会产生中止报文的请求。
3： 这些位定义发送缓冲区的传输顺序。它们不会改变 CAN 报文标识符。

寄存器 27-6： 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 （如果 EXIDE （TXBnSIDL<3>） = 0）
扩展标识符位， EID<28:21> （如果 EXIDE = 1）。

寄存器 27-7： 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 （如果 EXIDE （TXBnSIDL<3>） = 0）
扩展标识符位， EID<20:18> （如果 EXIDE = 1）。

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> 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 27-8： 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 （在发送标准标识符报文时不使用）

寄存器 27-9： 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 （在发送标准标识符报文时不使用）

寄存器 27-10： 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）
每个发送缓冲区都具有一个寄存器阵列。例如，发送缓冲区 0 具有 8 个寄存器：TXB0D0 至 TXB0D7。

寄存器 27-11： TXBnDLC：发送缓冲区 n 数据长度编码寄存器 [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 = 发送报文的 TXRTR 位将置 1
0 = 发送报文的 TXRTR 位将清零

bit 5-4 **未实现：**读为 0

bit 3-0 **DLC<3:0>：**数据长度编码位
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 字节
0111 = 数据长度 = 7 字节
0110 = 数据长度 = 6 字节
0101 = 数据长度 = 5 字节
0100 = 数据长度 = 4 字节
0011 = 数据长度 = 3 字节
0010 = 数据长度 = 2 字节
0001 = 数据长度 = 1 字节
0000 = 数据长度 = 0 字节

寄存器 27-12： 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
该寄存器中包含一个基于错误发生率而得到的值。当错误计数器发生溢出时，将会出现总线关闭状态。当总线上出现 128 次 11 个连续隐性位时，计数器的值会被清零。

例 27-3：　　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 褻ANKED” 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 27-4：　　通过使用 WIN 位来发送 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.
```

27.2.3 专用 CAN 接收缓冲寄存器

本节介绍专用 CAN 接收缓冲寄存器及其相关的控制寄存器。

寄存器 27-13： 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 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 = 接收缓冲区中包含一个接收到的报文
0 = 接收缓冲区可用于接收新报文

bit 6,6-5 模式 0：
RXM<1:0>：接收缓冲区模式位 1（与 RXM0 组合构成 RXM<1:0> 位，请参见 bit 5）
11 = 接收所有报文（包括带有错误的报文）；忽略过滤条件
10 = 仅接收带有扩展标识符的有效报文； RXFnSIDL 中的 EXIDEN 必须为 1
01 = 仅接收带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 2：
RXM1：接收缓冲区模式位 1
1 = 接收所有报文（包括带有错误的报文）；忽略接收过滤器
0 = 根据接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 组合构成 RXM<1:0> 位，请参见 bit 6）
模式 1 和 2：
RTRRO：接收报文远程发送请求位（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 2：
FILHIT<4:0>：过滤器命中位 4
该位与其他一些位组合构成过滤器接收位 <4:0>。

bit 3 模式 0：
RXRTRRO：接收报文远程发送请求位（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 2：
FILHIT<4:0>：过滤器命中位 3
该位与其他一些位组合构成过滤器接收位 <4:0>。

注 1： 该位在接收到报文时由 CAN 模块置 1，必须在读取缓冲区之后用软件清零。只要 RXFUL 置 1，就不会装入新的报文，并将缓冲区视为已满。清零 RXFUL 标志之后，可以清零 PIR5 位 RXB0IF。如果 RXB0IF 清零，但 RXFUL 未清零，则 RXB0IF 会再次置 1。

2： 该位允许对于 RXB0CON 和 RXB1CON 使用同一过滤器跳转表。

寄存器 27-13： 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 双重缓冲使能位
1 = 接收缓冲区 0 溢出内容将写入接收缓冲区 1
0 = 接收缓冲区 0 溢出内容不写入接收缓冲区 1
模式 1 和 2：
FILHIT<4:0>：过滤器命中位 2
该位与其他一些位组合构成过滤器接收位 <4:0>。

bit 1 模式 0：
JTOFF：跳转表偏移位 （RXB0DBEN 的只读副本） **(2)**
1 = 允许 6 和 7 之间的跳转表偏移
0 = 允许 1 和 0 之间的跳转表偏移
模式 1 和 2：
FILHIT<4:0>：过滤器命中位 1
该位与其他一些位组合构成过滤器接收位 <4:0>。

bit 0 模式 0：
FILHIT0：过滤器命中位 0
该位指示哪个接收过滤器允许接收的报文进入接收缓冲区 0。
1 = 接收过滤器 1 （RXF1）
0 = 接收过滤器 0 （RXF0）
模式 1 和 2：
FILHIT<4:0>：过滤器命中位 0
该位与 FILHIT<4:1> 组合使用，指示哪个接收过滤器允许接收的报文进入该接收缓冲区。
01111 = 接收过滤器 15 （RXF15）
01110 = 接收过滤器 14 （RXF14）
...
00000 = 接收过滤器 0 （RXF0）

注 1： 该位在接收到报文时由 CAN 模块置 1，必须在读取缓冲区之后用软件清零。只要 RXFUL 置 1，就不会装入新的报文，并将缓冲区视为已满。清零 RXFUL 标志之后，可以清零 PIR5 位 RXB0IF。如果 RXB0IF 清零，但 RXFUL 未清零，则 RXB0IF 会再次置 1。

2： 该位允许对于 RXB0CON 和 RXB1CON 使用同一过滤器跳转表。

寄存器 27-14： 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 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 = 接收缓冲区中包含一个接收到的报文
0 = 接收缓冲区可用于接收新报文

bit 6-5, 6 模式 0：
RXM<1:0>：接收缓冲区模式位 1（与 RXM0 组合构成 RXM<1:0> 位，请参见 bit 5）
11 = 接收所有报文（包括带有错误的报文）；忽略过滤条件
10 = 仅接收带有扩展标识符的有效报文； RXFnSIDL 中的 EXIDEN 必须为 1
01 = 仅接收带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 2：
RXM1：接收缓冲区模式位
1 = 接收所有报文（包括带有错误的报文）；忽略接收过滤器
0 = 根据接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 组合构成 RXM<1:0> 位，请参见 bit 6）

模式 1 和 2：
RTRRO：接收报文远程发送请求位（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：
FILHIT24：过滤器命中位 4

模式 1 和 2：
FILHIT<4:0>：过滤器命中位 4
该位与其他一些位组合构成过滤器接收位 <4:0>。

bit 3 模式 0：
RXRTRRO：接收报文远程发送请求位（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 2：
FILHIT<4:0>：过滤器命中位 3
该位与其他一些位组合构成过滤器接收位 <4:0>。

注 1： 该位在接收到报文时由 CAN 模块置 1，必须在读取缓冲区之后用软件清零。只要 RXFUL 置 1，就不会装入新的报文，并将缓冲区视为已满。

寄存器 27-14： 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

这些位指示哪个接收过滤器允许上次接收的报文进入接收缓冲区 1。

111 = 保留
110 = 保留
101 = 接收过滤器 5 （RXF5）
100 = 接收过滤器 4 （RXF4）
011 = 接收过滤器 3 （RXF3）
010 = 接收过滤器 2 （RXF2）
001 = 接收过滤器 1 （RXF1），仅当 RXB0DBEN 位置 1 时才可能
000 = 接收过滤器 0 （RXF0），仅当 RXB0DBEN 位置 1 时才可能

模式 1 和 2：

FILHIT<4:0>：过滤器命中位 <2:0>

这些位与 FILHIT<4:3> 组合使用，指示哪个接收过滤器允许接收的报文进入该接收缓冲区。

01111 = 接收过滤器 15 （RXF15）
01110 = 接收过滤器 14 （RXF14）
...
00000 = 接收过滤器 0 （RXF0）

注 1： 该位在接收到报文时由 CAN 模块置 1，必须在读取缓冲区之后用软件清零。只要 RXFUL 置 1，就不会装入新的报文，并将缓冲区视为已满。

寄存器 27-15： 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 （如果 EXID （RXBnSIDL<3>） = 0）

扩展标识符位， EID<28:21>（如果 EXID = 1）。

寄存器 27-16： 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 （如果 EXID = 0）
扩展标识符位， EID<20:18> （如果 EXID = 1）。

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> 为 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 27-17： 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 27-18： 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 27-19： RXBnDLC：接收缓冲区 n 数据长度编码寄存器 [0 ≤ n ≤ 1]

U-0	R-x	R-x	R-x	R-x	R-x	R-x	R-x
—	RXRTR	RB1	R0	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 = 远程传输请求
0 = 无远程传输请求

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>：**数据长度编码位
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 字节
0111 = 数据长度 = 7 字节
0110 = 数据长度 = 6 字节
0101 = 数据长度 = 5 字节
0100 = 数据长度 = 4 字节
0011 = 数据长度 = 3 字节
0010 = 数据长度 = 2 字节
0001 = 数据长度 = 1 字节
0000 = 数据长度 = 0 字节

寄存器 27-20： 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）
每个接收缓冲区都具有一个寄存器阵列。例如，接收缓冲区 0 具有 8 个寄存器：RXB0D0 至 RXB0D7。

寄存器 27-21： 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

该寄存器包含根据 CAN 规范定义的接收错误值。当 RXERRCNT > 127 时，模块会进入错误被动状态。 RXERRCNT 没有将模块置为 “ 总线关闭 ” 状态的功能。

例 27-5： 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.
    ...
```

27.2.3.1 可编程 TX/RX 和自动 RTR 缓冲区

ECAN 模块包含 6 个报文缓冲区，这些缓冲区可设定为发送或接收缓冲区。此外，所有这些缓冲区还可以设定为自动处理 RTR 报文。

注： 模式 0 下不使用这些寄存器。

寄存器 27-22： BnCON：TX/RX 缓冲区 n 控制寄存器 （接收模式）
[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 = 接收缓冲区中包含一个接收到的报文
0 = 接收缓冲区可用于接收新报文

bit 6 **RXM1：**接收缓冲区模式位
1 = 接收包括不完整报文和无效报文在内的所有报文（忽略接收过滤器）
0 = 根据接收过滤器接收所有有效报文

bit 5 **RXRTRRO：**接收报文远程发送请求位 （只读）
1 = 接收到的报文是远程发送请求
0 = 接收到的报文不是远程发送请求

bit 4-0 **FILHIT<4:0>：**过滤器命中位
这些位指示哪个接收过滤器允许上次接收的报文进入该缓冲区。
01111 = 接收过滤器 15 （RXF15）
01110 = 接收过滤器 14 （RXF14）
...
00001 = 接收过滤器 1 （RXF1）
00000 = 接收过滤器 0 （RXF0）

注 1： 这些寄存器仅在模式 1 和 2 下可用。
2： 该位在接收到报文时由 CAN 模块置 1，必须在读取缓冲区之后用软件清零。只要 RXFUL 置 1，就不会装入新的报文，并将缓冲区视为已满。

寄存器 27-23： BnCON：TX/RX 缓冲区 n 控制寄存器 （发送模式）
[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 = 未发送任何报文

bit 6 **TXABT：**发送中止状态位 (3)
1 = 中止报文
0 = 未中止报文

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 = 报文发送时发生总线错误
0 = 报文发送时未发生总线错误

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 = 在报文成功发送后自动清零

bit 2 **RTREN：**自动远程发送请求使能位
1 = 当接收到远程发送请求时，自动将 TXREQ 置 1
0 = 当接收到远程发送请求时， TXREQ 不受影响

bit 1-0 **TXPRI<1:0>：**发送优先级位 (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： 这些寄存器仅在模式 1 和 2 下可用。
2： 在该位置 1 时用软件清零该位会产生中止报文的请求。
3： 当 TXREQ 置 1 时自动清零该位。
4： 在 TXREQ 置 1 或正在进行发送时，发送缓冲寄存器将保持只读。
5： 这些位用于设置发送缓冲寄存器的传输顺序。它们不会改变 CAN 报文标识符。

寄存器 27-24： BnSIDH：TX/RX 缓冲区 n 标准标识符寄存器，高字节 （接收模式）
[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 （如果 EXIDE （BnSIDL<3>） = 0）
扩展标识符位， EID<28:21> （如果 EXIDE = 1）。

注 1： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-25： BnSIDH：TX/RX 缓冲区 n 标准标识符寄存器，高字节 （发送模式）
[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 （如果 EXIDE （BnSIDL<3>） = 0）
扩展标识符位， EID<28:21> （如果 EXIDE = 1）。

注 1： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-26： BnSIDL：TX/RX 缓冲区 n 标准标识符寄存器，低字节 （接收模式）
[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 （如果 EXID = 0）
扩展标识符位， EID<20:18> （如果 EXID = 1）。

bit 4 **SRR：**替代远程发送请求位
当 EXID = 1 时，该位总是为 1；或者当 EXID = 0 时，它等于 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> 为 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： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-27： BnSIDL：TX/RX 缓冲区 n 标准标识符寄存器，低字节 （发送模式）
[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（如果 EXIDE = 0）
扩展标识符位， EID<20:18>（如果 EXIDE = 1）。

bit 4 **未实现：**读为 0

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> 为 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： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-28： BnEIDH：TX/RX 缓冲区 n 扩展标识符寄存器，高字节 （接收模式）
[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： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-29： BnEIDH：TX/RX 缓冲区 n 扩展标识符寄存器，高字节 （发送模式）
[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： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-30： BnEIDL：TX/RX 缓冲区 n 扩展标识符寄存器，低字节 （接收模式）
[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： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-31： BnEIDL：TX/RX 缓冲区 n 扩展标识符寄存器，低字节 （发送模式）
[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	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： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-32： BnDm：TX/RX 缓冲区 n 数据字段字节 m 寄存器 （接收模式）
[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）
每个接收缓冲区都具有一个寄存器阵列。例如，接收缓冲区 0 具有 8 个寄存器 ： B0D0 至 B0D7。

注 1： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-33： BnDm：TX/RX 缓冲区 n 数据字段字节 m 寄存器 （发送模式）
[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）
每个发送缓冲区都具有一个寄存器阵列。例如，发送缓冲区 0 具有 8 个寄存器：TXB0D0 至 TXB0D7。

注 1： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-34： BnDLC：TX/RX 缓冲区 n 数据长度编码寄存器 （接收模式）
[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>：**数据长度编码位
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 字节
0111 = 数据长度 = 7 字节
0110 = 数据长度 = 6 字节
0101 = 数据长度 = 5 字节
0100 = 数据长度 = 4 字节
0011 = 数据长度 = 3 字节
0010 = 数据长度 = 2 字节
0001 = 数据长度 = 1 字节
0000 = 数据长度 = 0 字节

注 1： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-35： BnDLC：TX/RX 缓冲区 n 数据长度编码寄存器 （发送模式） [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 = 发送报文的 RTR 位将置 1
0 = 发送报文的 RTR 位将清零

bit 5-4 **未实现：**读为 0

bit 3-0 **DLC<3:0>：**数据长度编码位
1111-1001 = 保留
1000 = 数据长度 = 8 字节
0111 = 数据长度 = 7 字节
0110 = 数据长度 = 6 字节
0101 = 数据长度 = 5 字节
0100 = 数据长度 = 4 字节
0011 = 数据长度 = 3 字节
0010 = 数据长度 = 2 字节
0001 = 数据长度 = 1 字节
0000 = 数据长度 = 0 字节

注 1： 这些寄存器仅在模式 1 和 2 下可用。

寄存器 27-36： 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 = 缓冲区被配置为发送模式
0 = 缓冲区被配置为接收模式

bit 1-0 **未实现：**读为 0

注 1： 这些寄存器仅在模式 1 和 2 下可用。

27.2.3.2 报文接收过滤器和屏蔽器

本节介绍CAN接收缓冲区的报文接收过滤器和屏蔽器。

寄存器 27-37： RXFnSIDH：接收过滤器 n 标准标识符过滤寄存器，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>：**标准标识符过滤器位（如果 EXIDEN = 0）
扩展标识符过滤器位， EID<28:21>（如果 EXIDEN = 1）。

注 1： 寄存器 RXF6SIDH:RXF15SIDH 仅在模式 1 和 2 下可用。

寄存器 27-38： RXFnSIDL：接收过滤器 n 标准标识符过滤寄存器，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>：**标准标识符过滤器位（如果 EXIDEN = 0）
扩展标识符过滤器位， EID<20:18>（如果 EXIDEN = 1）。

bit 4 **未实现：**读为 0

bit 3 **EXIDEN：**扩展标识符过滤器使能位 (2)
1 = 过滤器将只接收扩展 ID 报文
0 = 过滤器将只接收标准 ID 报文

bit 2 **未实现：**读为 0

bit 1-0 **EID<17:16>：**扩展标识符过滤器位

注 1： 寄存器 RXF6SIDL:RXF15SIDL 仅在模式 1 和 2 下可用。
2： 在模式 0 下，不论相应的屏蔽寄存器的值如何，都必须根据需要置 1/ 清零该位。

寄存器 27-39： RXFnEIDH：接收过滤器 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>：**扩展标识符过滤器位

注 1： 寄存器 RXF6EIDH:RXF15EIDH 仅在模式 1 和 2 下可用。

寄存器 27-40： RXFnEIDL：接收过滤器 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>：**扩展标识符过滤器位

注 1： 寄存器 RXF6EIDL:RXF15EIDL 仅在模式 1 和 2 下可用。

寄存器 27-41： RXMnSIDH：接收屏蔽器 n 标准标识符屏蔽寄存器，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 27-42： RXMnSIDL：接收屏蔽器 n 标准标识符屏蔽寄存器，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 2：
EXIDEN：扩展标识符过滤器使能屏蔽位 (1)
1 = 接收由 RXFnSIDL 中的 EXIDEN 位选择的报文
0 = 同时接收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： 该位仅在模式 1 和 2 下可用。

寄存器 27-43： RXMnEIDH：接收屏蔽器 n 扩展标识符屏蔽寄存器，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 27-44： RXMnEIDL：接收屏蔽器 n 扩展标识符屏蔽寄存器，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 27-45： 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： 该寄存器仅在模式 1 和 2 下可用。

注： 寄存器 27-46 至寄存器 27-51 仅在配置模式下可写。

寄存器 27-46： 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：
不使用：强制为 00000。

00000-10010 = 0	有 18 个位可用于标准数据字节过滤器。实际使用位数取决于所接收报文的 DLC<3:0> 位（RXBnDLC<3:0> 或 BnDLC<3:0>，如果配置为 RX 缓冲区）。
如果 DLC<3:0>= 0000	不将任何位与传入数据位进行比较。
如果 DLC<3:0>= 0001	最多将 RXFnEID<7:0> 的 8 个数据位（由 FLC<2:0> 决定）与传入报文中相应数量的数据位进行比较。
如果 DLC<3:0>= 0010	最多将 RXFnEID<15:0> 的 16 个数据位（由 FLC<3:0> 决定）与传入报文中相应数量的数据位进行比较。
如果 DLC<3:0>= 0011	最多将 RXFnEID<17:0> 的 18 个数据位（由 FLC<4:0> 决定）与传入报文中相应数量的数据位进行比较。

注 1： 该寄存器仅在模式 1 和 2 下可用。

寄存器 27-47： 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 关联
0001 = 过滤器 n 与 RXB1 关联
0010 = 过滤器 n 与 B0 关联
0011 = 过滤器 n 与 B1 关联
...
0111 = 过滤器 n 与 B5 关联
1111-1000 = 保留

注 1： 该寄存器仅在模式 1 和 2 下可用。

寄存器 27-48： 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
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 5-4 **FIL2_<1:0>：**过滤器 2 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 3-2 **FIL1_<1:0>：**过滤器 1 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 1-0 **FIL0_<1:0>：**过滤器 0 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

注 1： 该寄存器仅在模式 1 和 2 下可用。

寄存器 27-49： 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
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 5-4 **FIL6_<1:0>：**过滤器 6 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 3-2 **FIL5_<1:0>：**过滤器 5 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 1-0 **FIL4_<1:0>：**过滤器 4 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

注 1： 该寄存器仅在模式 1 和 2 下可用。

寄存器 27-50： 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
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 5-4 **FIL10_<1:0>：**过滤器 10 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 3-2 **FIL9_<1:0>：**过滤器 9 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 1-0 **FIL8_<1:0>：**过滤器 8 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

注 1： 该寄存器仅在模式 1 和 2 下可用。

寄存器 27-51： 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
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 5-4 **FIL14_<1:0>：**过滤器 14 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 3-2 **FIL13_<1:0>：**过滤器 13 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

bit 1-0 **FIL12_<1:0>：**过滤器 12 选择位 1 和 0
11 = 无屏蔽
10 = 过滤器 15
01 = 接收屏蔽器 1
00 = 接收屏蔽器 0

注 1： 该寄存器仅在模式 1 和 2 下可用。

27.2.4 CAN 波特率寄存器

本节介绍 CAN 波特率寄存器。

注：	这些寄存器仅在配置模式下可写。

寄存器 27-52： 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 27-53： 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（Information Processing Time， 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 27-54：　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：**选择是否使用 CAN 总线线路滤波器唤醒的位
1 = 使用 CAN 总线线路滤波器来唤醒
0 = 不使用 CAN 总线线路滤波器来唤醒

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： 如果 SEG2PHTS 位（BRGCON2<7>）为 0 则忽略。

27.2.5 CAN 模块 I/O 控制寄存器

该寄存器控制 CAN 模块 I/O 引脚涉及到单片机其他部分的操作。

寄存器 27-55： CIOCON：CAN I/O 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0
TX2SRC	TX2EN	ENDRHI(1)	CANCAP	—	—	—	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 **TX2SRC**：CANTX2 引脚数据源位
1 = CANTX2 引脚将输出 CAN 时钟
0 = CANTX2 引脚将输出 CANTX

bit 6 **TX2EN**：CANTX 引脚使能位
1 = CANTX2 引脚将输出 CANTX 或 CAN 时钟，具体由 TX2SRC 位选择
0 = CANTX2 引脚将具有数字 I/O 功能

bit 5 **ENDRHI**：使能驱动高电平位 (1)
1 = 隐性时 CANTX 引脚将驱动 VDD
0 = 隐性时 CANTX 引脚将为三态

bit 4 **CANCAP**：CAN 报文接收捕捉使能位
1 = 使能 CAN 捕捉； CAN 报文接收信号取代 RC2/CCP1 上的输入
0 = 禁止 CAN 捕捉； RC2/CCP1 输入送至 CCP1 模块

bit 3-1 **未实现**：读为 0

bit 0 **CLKSEL**：CAN 时钟源选择位
1 = 使用振荡器作为 CAN 系统时钟源
0 = 使用 PLL 作为 CAN 系统时钟源

注 1： 使用差分总线时请始终将该位置 1，以避免其他邻近引脚在 CANTX 上产生信号串扰。

27.2.6 CAN 中断寄存器

本节介绍的寄存器与**第 10.0 节 “ 中断 ”** 中介绍的相同。
此处重复介绍是为了方便起见。

寄存器 27-56： PIR5：外设中断请求 （标志）寄存器 5

模式 0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	IRXIF	WAKIF	ERRIF	TXB2IF	TXB1IF(1)	TXB0IF(1)	RXB1IF	RXB0IF
模式 1 和 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
	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**：CAN 总线错误报文接收中断标志位
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 总线上未发生任何活动

bit 5 **ERRIF**：CAN 模块错误中断标志位
1 = CAN 模块中发生了错误 （多个来源；请参见**第 27.15.6 节 “ 错误中断 ”**）
0 = 未发生 CAN 模块错误

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 已完成报文发送，可以向其中重新装入数据
0 = 发送缓冲区 2 尚未完成报文发送
当 CAN 处于模式 1 或模式 2 时：
TXBnIF：任意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发送，可以向其中重新装入数据
0 = 没有任何发送缓冲区已准备好重新装入数据

bit 3 **TXB1IF**：CAN 发送缓冲区 1 中断标志位 (1)
1 = 发送缓冲区 1 已完成报文发送，可以向其中重新装入数据
0 = 发送缓冲区 1 尚未完成报文发送

bit 2 **TXB0IF**：CAN 发送缓冲区 0 中断标志位 (1)
1 = 发送缓冲区 0 已完成报文发送，可以向其中重新装入数据
0 = 发送缓冲区 0 尚未完成报文发送

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 已接收到新报文
0 = 接收缓冲区 1 未接收到新报文
当 CAN 处于模式 1 或模式 2 时：
RXBnIF：任意接收缓冲区中断标志位
1 = 一个或多个接收缓冲区已接收到新报文
0 = 没有任何接收缓冲区已接收到新报文

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 已接收到新报文
0 = 接收缓冲区 0 未接收到新报文
当 CAN 处于模式 1 时：
未实现：读为 0
当 CAN 处于模式 2 时：
FIFOWMIF：FIFO 水位线中断标志位
1 = 已达到 FIFO 高水位线
0 = 未达到 FIFO 高水位线

注 1： 在 CAN 模式 1 和 2 下，这些位被强制为 0。

寄存器 27-57： PIE5：外设中断允许寄存器 5

模式 0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	IRXIE	WAKIE	ERRIE	TXB2IE	TXB1IE(1)	TXB0IE(1)	RXB1IE	RXB0IE

模式 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
	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**：CAN 总线错误报文接收中断允许位
1 = 允许无效报文接收中断
0 = 禁止无效报文接收中断

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；通过设置 TXBIE 和 BIE0 来允许单独的中断
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；通过设置 BIE0 来允许单独的中断
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 水位线中断允许位
1 = 允许 FIFO 水位线中断
0 = 禁止 FIFO 水位线中断

注 1： 在 CAN 模式 1 和 2 下，这些位被强制为 0。

寄存器 27-58：　　IPR5：外设中断优先级寄存器 5

模式 0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	IRXIP	WAKIP	ERRIP	TXB2IP	TXB1IP(1)	TXB0IP(1)	RXB1IP	RXB0IP

模式 1 和 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
	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**：CAN 总线错误报文接收中断优先级位
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

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 水位线中断优先级位
1 = 高优先级
0 = 低优先级

注　1： 在 CAN 模式 1 和 2 下，这些位被强制为 0。

寄存器 27-59： 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： 该寄存器仅在模式 1 和 2 下可用。
2： 要产生中断， PIE5 寄存器中的 TXBnIE 必须置 1。

寄存器 27-60： 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： 该寄存器仅在模式 1 和 2 下可用。
2： 要产生中断， PIE5 寄存器中的 TXBnIE 或 RXBnIE 必须置 1。

27.3 CAN 工作模式

PIC18F66K80 系列具有 6 种主要工作模式：

- 配置模式
- 禁止 / 休眠模式
- 正常工作模式
- 监听模式
- 环回模式
- 错误识别模式

所有模式（错误识别除外）都通过设置 REQOP 位（CANCON<7:5>）来进行请求。错误识别模式需要通过接收缓冲寄存器的 RXM 位来进行请求。是否进入某种模式可通过监视 OPMODE 位来确认。

更改模式时，只有所有待发送报文均完成之后，才会实际更改模式。因此，用户必须在执行进一步的操作之前，先确认器件已实际切换为所请求的模式。

27.3.1 配置模式

在激活之前，必须先初始化 CAN 模块。只有在模块处于配置模式时才能执行该操作。配置模式通过将 REQOP2 位置 1 来请求。只有状态位 OPMODE2 为高电平时，才能执行初始化。之后，可以写入配置寄存器、接收屏蔽寄存器和接收过滤寄存器。模块通过将 REQOP 控制位设置为 0 来激活。

模块会防止用户因为编程错误而意外违反CAN协议。当模块在线时，所有控制模块配置的寄存器都不能被修改。在进行发送或接收时，不允许 CAN 模块进入配置模式。配置模式会作为锁来保护以下寄存器：

- 配置寄存器
- 功能模式选择寄存器
- 位时序寄存器
- 标识符接收过滤寄存器
- 标识符接收屏蔽寄存器
- 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。错误计数器被清零且中断标志位保持不变。编程人员可以访问在其他模式下不可访问的配置寄存器。 I/O 引脚将恢复为普通 I/O 功能。

27.3.2 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。由于总线活动，模块能够将 WAKIF 位置 1；但是，等待处理的中断将继续等待，且错误计数器的值也将保持不变。

如果 REQOP<2:0> 位设置为 001，模块将进入模块禁止 / 休眠模式。该模式类似于通过关闭模块使能来禁止其他外设模块。除非模块处于工作状态（例如，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，它将等待检测到 CAN 总线上出现 11 个隐性位，表明总线空闲，然后才能执行模块禁止 / 休眠命令。OPMODE<2:0> = 001 指示模块是否成功进入模块禁止 / 休眠模式。

WAKIF 中断是处于禁止 / 休眠模式时惟一仍然有效的模块中断。如果 WAKDIS 清零且 WAKIE 置 1，则每次模块检测到从隐性到显性的电平变换时，处理器将会接收到一个中断。在唤醒时，模块会被自动设置为先前的工作模式。例如，如果模块之前是从正常模式切换为禁止 / 休眠模式，则在发生总线活动唤醒时，模块会自动进入正常模式，并且导致模块唤醒的第一个报文会丢失。模块将不会产生任何错误帧。固件逻辑必须检测到这一情况，并确保请求重新发送。如果在处理器正在休眠时接收到唤醒中断，则可能会丢失多个报文。丢失的实际报文数量取决于处理器振荡器起振时间和传入报文的比特率。

在模块处于禁止 / 休眠模式时，TXCAN 引脚将保持在隐性状态。

27.3.3 正常模式

这是 PIC18F66K80 系列器件的标准工作模式。在该模式下，器件会主动监视所有总线报文并产生应答位和错误帧等。这也是 PIC18F66K80 系列器件会通过 CAN 总线发送报文的惟一模式。

27.3.4 监听模式

监听模式为 PIC18F66K80 系列器件提供了一种接收所有报文（包括带有错误的报文）的方法。该模式可用于总线监视应用或用于检测“ 热插拔 ”情形中的波特率。对于自动波特率检测，必须至少要有两个相互通信的其他节点。波特率可以使用经验方式通过测试不同的值来检测，直到接收到有效报文为止。监听模式是一种静默模式，即处于该状态时模块不会发送任何报文，包括错误标志或应答信号。通过使用过滤器和屏蔽器，可以仅允许特定报文装入接收寄存器；或者，也可以将过滤器屏蔽器设置为全零，以允许带有任意标识符的报文通过。在该状态下，错误计数器会发生复位且不再工作。监听模式通过设置 CANCON 寄存器中的模式请求位来激活。

27.3.5 环回模式

该模式允许在内部从发送缓冲区向接收缓冲区发送报文，而无需在 CAN 总线上实际发送报文。该模式可用于系统开发和测试。在该模式下，ACK 位会被忽略，器件将允许来自自身的传入报文，就如同它们来自另一个节点。环回模式是一种静默模式，即处于该状态时模块不会发送任何报文，包括错误标志或应答信号。在器件处于该模式时，TXCAN 引脚将恢复为端口 I/O。通过使用过滤器和屏蔽器，可以仅允许特定报文装入接收寄存器。屏蔽器可以设置为全零，提供一种接收所有报文的模式。环回模式通过设置 CANCON 寄存器中的模式请求位来激活。

27.3.6 错误识别模式

该模块能够设置为忽略所有错误并接收所有报文。在功能模式 0 下，错误识别模式通过将 RXBnCON 寄存器中的 RXM<1:0> 位设置为 11 来激活。处于该模式时，在发生错误之前进入报文组合缓冲区的数据会被复制到接收缓冲区中，并且可以通过 CPU 接口读取。

27.4 CAN 模块的功能模式

除了 CAN 工作模式之外，ECAN 模块还提供了总共 3 种功能模式。这些模式使用模式 0、模式 1 和模式 2 表示。

27.4.1 模式 0—— 传统模式

模式 0 设计为完全兼容 PIC18CXX8 和 PIC18FXX8 器件中使用的 CAN 模块。它是所有复位条件下的默认工作模式。因此，针对 PIC18XX8 CAN 模块编写的模块代码可以在无需任何代码更改的情况下用于 ECAN 模块。

以下列出了模式 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 个接收屏蔽器，每个接收缓冲区各一个：RXM0 和 RXM1
- 6 个接收过滤器，2 个用于 RXB0，4 个用于 RXB1 它们分别是 RXF0、RXF1、RXF2、RXF3、RXF4 和 RXF5

27.4.2 模式 1—— 增强型传统模式

模式 1 类似于模式 0，只是模式 1 下可用的资源更多。它有 16 个接收过滤器和 2 个接收屏蔽寄存器。接收过滤器 15 可以用作接收过滤器，也可以用作接收屏蔽寄存器。除了 3 个发送缓冲区和 2 个接收缓冲区之外，另外还有 6 个报文缓冲区。用户可以将这些缓冲区中的一个或多个设定为发送或接收缓冲区。此外，这些缓冲区还可以设定为自动处理 RTR 报文。

在 16 个接收过滤寄存器中，有 14 个可以与任意接收缓冲区和接收屏蔽寄存器动态关联。用户可以利用该功能将多个过滤器与任意一个缓冲区关联。

当接收缓冲区设定为使用标准标识符报文时，完全接收过滤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设定为 0 至 18 位。该功能可以简化高级协议的实现，例如 DeviceNet™ 协议。

以下列出了模式 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 个可设定为 TX 或 RX 的缓冲区：B0-B5
- B0-B5 上的自动 RTR 处理
- 16 个动态分配的接收过滤器：RXF0-RXF15
- 2 个专用接收屏蔽寄存器； RXF15 可设定为第 3 个屏蔽器：RXM0-RXM1 和 RXF15
- 用于标准标识符报文的可编程数据过滤器：SDFLC

27.4.3 模式 2—— 增强型 FIFO 模式

在模式 2 下，模块使用两个或更多的接收缓冲区来构成接收 FIFO（先进先出）缓冲区。接收缓冲区和接收过滤寄存器之间不存在一一对应关系。任何已使能并链接到任意 FIFO 接收缓冲区的过滤器都可以产生报文接收并导致 FIFO 发生更新。

FIFO 长度可由用户设定为 2-8 个缓冲区。 FIFO 长度由配置为发送缓冲区的第一个可编程缓冲区决定。例如，如果将缓冲区 2（B2）设定为发送缓冲区，则 FIFO 将由 RXB0、RXB1、B0 和 B1 组成，构成一个长度为 4 的 FIFO。如果所有可编程缓冲区都配置为接收缓冲区，则 FIFO 长度将为最大长度 8。

以下列出了模式 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 上的自动 RTR 处理
- 16 个接收过滤器：RXF0-RXF15
- 2 个专用接收屏蔽寄存器；RXF15 可设定为第 3 个屏蔽器：RXM0-RXM1 和 RXF15
- 用于标准标识符报文的可编程数据过滤器：SDFLC，对于 DeviceNet 协议很有用

27.5 CAN 报文缓冲区

27.5.1 专用发送缓冲区

PIC18F66K80 系列器件实现了 3 个专用发送缓冲区 ——TXB0、 TXB1 和 TXB2。其中每个缓冲区都占用 14 字节的 SRAM，并映射到 SFR 存储器映射中。在模式 0 下，只有这些发送缓冲区可供使用。模式 1 和 2 可以访问它们以及另外一些缓冲区。

每个发送缓冲区都包含 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）。

27.5.2 专用接收缓冲区

PIC18F66K80 系列器件实现了 2 个专用接收缓冲区：RXB0 和 RXB1。其中每个缓冲区都占用 14 字节的 SRAM，并映射到 SFR 存储器映射中。在模式 0 下，只有这些接收缓冲区可供使用。模式 1 和 2 可以访问它们以及另外一些缓冲区。

每个接收缓冲区都包含 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）。

此外还有一个独立的报文组合缓冲区（ Message Assembly Buffer，MAB），它用作一个额外的接收缓冲区。MAB 总是用于从总线上接收下一个报文，用户固件不能直接访问它。MAB 会逐个组合所有传入报文。只有在满足相应的接收过滤器条件时，报文才会被传输到相应的接收缓冲区中。

27.5.3 可编程发送 / 接收缓冲区

ECAN 模块实现了 6 个新的缓冲区：B0-B5。这些缓冲区可单独设定为发送或接收缓冲区。这些缓冲区仅在模式 1 和 2 下可用。与专用发送和接收缓冲区一样，这些可编程缓冲区各占用14字节的SRAM，并且映射到SFR存储器映射中。

每个缓冲区都包含 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）。这些寄存器中的每一个都包含两组控制位。根据缓冲区是配置为发送还是接收缓冲区，模块会使用相应的一组控制位。默认情况下，所有缓冲区都配置为接收缓冲区。通过设置 BSEL0 寄存器中的相应 TXENn位，每个缓冲区都可以单独配置为发送或接收缓冲区。

当配置为发送缓冲区时，用户固件可以按任意顺序访问发送缓冲区，类似于访问专用发送缓冲区。在使能模式 1 的接收配置下，用户固件还可以按所需的任意顺序访问接收缓冲区。但在模式 2 下，所有接收缓冲区将组合构成单个 FIFO。实际 FIFO 长度由用户固件设定。对 FIFO 的访问必须通过 CANCON 寄存器中的 FIFO 指针位（FP<4:0>）来完成。必须注意，不存在任何用于防止 FIFO 读操作顺序错误的硬件保护。

27.5.4 可编程自动 RTR 缓冲区

在模式 1 和 2 下， 6 个可编程发送 / 接收缓冲区中的任意一个都可以设定为自动响应预定义的 RTR 报文，而无需用户固件干预。自动 RTR 处理通过将 BSEL0 寄存器中的 TX2EN 位和 BnCON 寄存器中的 RTREN 位置 1 来使能。进行此项设置之后，当接收到 RTR 请求时，TXREQ 位会自动置 1，并且当前缓冲区内容会被自动排入发送队列，作为 RTR 响应。与所有发送缓冲区一样，在 TXREQ 位置 1 之后，缓冲寄存器会变为只读状态，对于它们的所有写操作都会被忽略。

以下简要介绍了自动处理 RTR 报文需要执行的步骤：

1. 通过将 BSEL0 寄存器中的 TXnEN 位设置为 1，将缓冲区设置为发送模式。
2. 必须至少将一个接收过滤器与该缓冲区关联，并预先装入所期望的 RTR 标识符。
3. BnCON 寄存器中的 RTREN 位必须设置为 1。
4. 缓冲区中必须预先装入要作为 RTR 响应发送的数据。

通常情况下，用户固件会将缓冲数据寄存器保持为最新内容。如果固件尝试在发送自动 RTR 响应的过程中更新缓冲区，则对于缓冲区的所有写操作都会被忽略。

27.6 CAN 报文发送

27.6.1 启动发送

为了让 MCU 可以对报文缓冲区进行写访问，TXREQ 位必须清零，表明报文缓冲区中没有任何待发送的报文。至少，必须向 SIDH、SIDL 和 DLC 寄存器中装入数据。如果报文中存在数据字节，则还必须向数据寄存器中装入数据。如果报文要使用扩展标识符，则还必须向 EIDH:EIDL 寄存器中装入数据，并且将 EXIDE 位置 1。

要启动报文发送，必须将每个待发送缓冲区的 TXREQ 位置 1。当 TXREQ 置 1 时，TXABT、TXLARB 和 TXERR 位会被清零。要成功完成发送，网络上必须至少存在一个波特率匹配的节点。

将 TXREQ 位置 1 并不会启动报文发送；它只是将报文缓冲区标记为已发送就绪。发送将在器件检测到总线可用时开始。然后，器件会开始发送优先级最高的已就绪报文。

当发送成功完成时，TXREQ 位会被清零，TXBnIF 位会被置 1，并且如果 TXBnIE 位置 1，则还会产生中断。

如果报文发送失败，TXREQ 位将保持置 1，指示该报文仍然等待发送，并且以下条件标志位之一将置 1。如果报文开始发送但遇到错误条件，则 TXERR 和 IRXIF 位将被置 1，并会产生中断。如果报文仲裁失败，TXLARB 位将被置 1。

27.6.2 中止发送

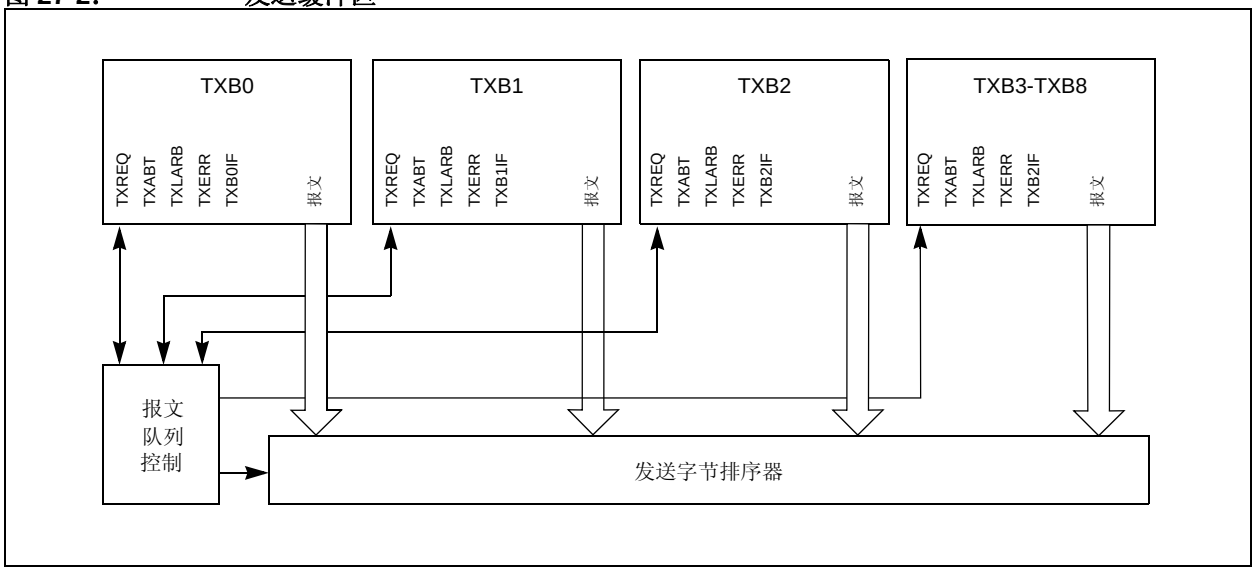
MCU 可以通过清零与相应报文缓冲区关联的 TXREQ 位（TXBnCON<3> 或 BnCON<3>）来请求中止报文。将 ABAT 位 （CANCON<4>）置 1 将请求中止所有等待发送的报文。如果报文还未开始发送或者报文已开始发送但由于仲裁失败或错误而被中断，那么将会执行中止。当模块将相应缓冲区的 TXABT 位 （TXBnCON<6> 或 BnCON<6>）置 1 时，指示报文已被中止。如果报文已经开始发送，则它会尝试完全发送当前报文。如果当前报文已完全发送，并且未发生仲裁失败或错误，则 TXABT 位将不会置 1，因为报文已成功发送。类似地，如果在中止请求期间发送某个报文，但报文发生仲裁失败或错误，则不会重新发送报文， TXABT 位将被置 1，指示报文已被成功中止。

通过将 ABAT 或 TXABT 位置 1 而请求中止之后，将无法通过清零它来取消中止请求。只有 CAN 模块硬件或 POR 条件可以清除它。

27.6.3 发送优先级

发送优先级指在 PIC18F66K80 系列器件内待发送报文的优先级。它独立于 CAN 协议内置的报文仲裁方案中暗含的任何优先级，与之无关。在发送帧起始（Start-of-Frame，SOF）之前，模块会对发送队列中所有缓冲区的优先级进行比较。优先级最高的发送缓冲区最先发送。如果两个缓冲区的优先级设置相同，则缓冲区编号最高的缓冲区最先发送。发送优先级有 4 级。如果特定报文缓冲区的 TXP 位设置为 11，则该缓冲区的优先级最高。如果特定报文缓冲区的 TXP 位设置为 00，则该缓冲区的优先级最低。

图 27-2：　　发送缓冲区



27.7 报文接收

27.7.1 接收报文

在所有接收缓冲区中，MAB总是用于从总线上接收下一个报文。 MCU 可以访问一个缓冲区，同时另一个缓冲区用于报文接收或保存先前接收到的报文。

注：	接收到报文之后，MAB的全部内容会被移入接收缓冲区中。这意味着，无论标识符类型（标准或扩展）和所接收的数据字节数如何， MAB 内容会覆盖整个接收缓冲区。因此，在接收到任何报文时，都必须假定缓冲区中所有寄存器的内容都已被修改。

当报文移入任意一个接收缓冲区时，关联的RXFUL位会置 1。该位必须由 MCU 在完成缓冲区中报文的处理时清零，以允许在缓冲区中接收新的报文。该位用于提供明确锁定功能，以确保在模块尝试向接收缓冲区中装入新报文之前，固件可以完成报文处理。如果允许接收中断，则模块会产生一个中断，指示已接收到有效报文。

将报文装入任意匹配的缓冲区中之后，用户固件可以通过检查 RXBnCON 或 BnCON 寄存器中的过滤器命中位来准确地确定导致此次接收的过滤器。在模式 0 下， RXBnCON 的 FILHIT<2:0> 用作过滤器命中位。在模式 1 和 2 下，BnCON 的 FILHIT<4:0> 位用作过滤器命中位。相同的寄存器还可以指示当前报文是否是 RTR 帧。如果 RXBnSIDL 或 BnSIDL 寄存器中的 EXID/EXIDE 位清零，则模块会认为接收到的报文是标准标识符报文。反之， EXID 位置 1 则指示接收到的报文是扩展标识符报文。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。对于扩展标识符报文，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，则用户固件应还需要通过访问 RXBnDm 或 BnDm 寄存器来读取相应的数据字节数。如果接收到的报文是 RTR，并且如果当前缓冲区未配置为进行自动 RTR 处理，则用户固件必须执行相应的操作并手动进行响应。

每个接收缓冲区都包含用于设置特殊接收模式的 RXM 位。在模式 0 下，RXBnCON 中的 RXM<1:0> 位总共可定义 4 种接收模式。在模式 1 和 2 下， RXM1 位结合 EXID 屏蔽器和过滤器位，可定义相同的 4 种接收模式。通常，这些位设置为 00 来允许接收由相应的接收过滤器确定为有效的所有报文。在这种情况下，是否接收标准或扩展报文由接收过滤寄存器中的EXIDE位决定。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有标准或扩展标识符的报文。如果接收过滤器的 EXIDE 位置 1，从而它不对应于 RXM 模式，则接收过滤器将不再起效。在模式 1 和 2 下，将 SIDL 屏蔽寄存器中的 EXID 位置 1 可以确保仅接收标准或扩展标识符。RXM位的这两种模式可以在已知总线上只有标准或扩展报文的系统中使用。如果 RXM 位设置为 11（在模式 1 和 2 下，RXM1 = 1），则无论接收过滤器的值如何，缓冲区都将接收所有报文。此外，如果报文在帧结束之前出错，则在错误帧之前在 MAB 中组合的报文部分将会被装入缓冲区。该模式可以用作给定CAN网络一种宝贵的调试工具。在实际系统环境中不应使用它，因为实际系统总是会发生一些总线错误，并且希望总线上的所有节点都忽略它们。

在模式 1 和 2 下，当一个可编程缓冲区配置为发送缓冲区，并且有一个或多个接收过滤器与之相关联时，所有与该接收过滤器条件匹配的传入报文都会被丢弃。为了避免这种情况，用户固件必须确保没有任何接收过滤器与配置为发送缓冲区的缓冲区相关联。

27.7.2 接收优先级

处于模式 0 时， RXB0 是优先级较高的缓冲区，并且有 2 个报文接收过滤器与之关联。RXB1 是优先级较低的缓冲区，并且有 4 个接收过滤器与之关联。由于接收过滤器的数量较少，所以 RXB0 的匹配条件更严格，且该缓冲区的优先级较高。此外， RXB0CON 寄存器还可以如此配置：如果 RXB0 包含一个有效报文，同时又接收到另一个有效报文，模块将不会发生溢出错误，无论 RXB1 的接收条件如何，新报文总是移入 RXB1。此外还有两个可编程的接收过滤器屏蔽器可供使用，每个接收缓冲区对应一个 （见**第 27.5 节 “CAN 报文缓冲区 ”**）。

在模式 1 和 2 下，共有 16 个接收过滤器可用，并且每个过滤器都可以动态分配给任意接收缓冲区。编号较低的缓冲区将具有较高的优先级。因此，如果传入报文与两个或更多接收缓冲区的接收条件匹配，报文将装入编号较低的缓冲区。

27.7.3 增强型 FIFO 模式

当配置为模式 2 时，两个专用接收缓冲区将与一个或多个可编程发送 / 接收缓冲区进行组合，用于构造一个最大深度为 8 个缓冲区的 FIFO 缓冲区。在这种模式下，过滤器和接收缓冲寄存器之间没有直接关系。任何已使能的过滤器都可以产生报文接收。当某个报文被接收时，它会被存储到下一个可用的接收缓冲寄存器中，并且内部写指针会递增。 FIFO 的最大深度可以为 8 个缓冲区。整个 FIFO 必须由连续的接收缓冲区构成。FIFO 头部从RXB0缓冲区开始，尾部可一直延伸到B5。FIFO 的最大长度受从 B0 开始所存在的第一个发送缓冲区限制。如果某个缓冲区配置为发送缓冲区，则 FIFO 长度会相应减小。例如，如果 B3 配置为发送缓冲区，则实际 FIFO 将包含 RXB0、RXB1、B0、B1 和 B2，共 5 个缓冲区。如果 B0 配置为发送缓冲区，则 FIFO 长度将为 2。如果没有任何一个可编程缓冲区配置为发送缓冲区，则 FIFO 的深度将为 8 个缓冲区。需要较多发送缓冲区的系统应尽量将发送缓冲区放置在B0-B5缓冲区的最末端，以最大程度提高可用 FIFO 长度。

在 FIFO 模式下接收到报文时， CANSTAT 寄存器中的中断标志编码位 （EICODE<4:0>）的值将为 10000，指示 FIFO 已接收到报文。 CANCON 寄存器中的 FIFO 指针位 FP<3:0> 将指向包含尚未读取数据的缓冲区。这种情况下，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，致使模块更新 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。如果它置 1，则 FIFO 可能包含一个或多个报文。在模式 2 下，模块还在 ECANCON 寄存器中提供一个称为 FIFO 高水位线 （FIFOWM）的位。该位可用于在每次 FIFO 仅包含 1 或 4 个空缓冲区时产生中断。 FIFO 高水位线中断可以用作 FIFO 满条件的提前警告。

27.7.4 时间标记

CAN 模块可以设定为对于接收到的每个报文产生一个时间标记。在使能时，模块为 CCP1 产生一个捕捉信号，使后者捕捉 Timer1 或 Timer3 的值。该值可以用作报文的时间标记。

要使用时间标记功能，CANCAP 位（CIOCON<4>）必须置 1。这将会使用 CAN 模块产生的信号取代 CCP1 的捕捉输入。此外，CCP1CON<3:0> 必须设置为 0011，以使能 CCP 特殊事件触发来产生 CAN 事件。

27.8 报文接收过滤器和屏蔽器

报文接收过滤器和屏蔽器用于决定报文组合缓冲区中的报文是否应被装入某个接收缓冲区中。在 MAB 中接收有效报文之后，模块会将报文的标识符字段与过滤值进行比较。如果匹配的话，该报文就会被装入相应的接收缓冲区。过滤器屏蔽器用于确定要使用过滤器检查标识符中的哪些位。表 27-1 给出了一个真值表，说明模块如何将标识符中的每个位与屏蔽器和过滤器进行比较，以确定是否将报文装入接收缓冲区。屏蔽器主要决定将对哪些位应用接收过滤器。如果任何屏蔽位被设置为零，无论过滤器位为何值该位都会被自动接收。

表 27-1： 过滤器 / 屏蔽器真值表

屏蔽器位 n	过滤器位 n	报文标识符位 n001	接收或拒绝位 n
0	x	x	接收
1	0	0	接收
1	0	1	拒绝
1	1	0	拒绝
1	1	1	接收

图注： x = 无关位

在模式 0 下，接收过滤器 RXF0 和 RXF1 以及过滤器屏蔽器 RXM0，与 RXB0 相关联。过滤器 RXF2、RXF3、RXF4 和 RXF5 以及屏蔽器 RXM1，与 RXB1 相关联。

在模式 1 和 2 下，另外还有 10 个接收过滤器 RXF6-RXF15，共有 16 个过滤器可供使用。RXF15 可以用作接收过滤器，也可以用作接收屏蔽寄存器。通过置 1 或清零 RXFCONn 寄存器中的 RXFENn 位，可以单独使能或禁止其中的每个接收过滤器。16 个接收过滤器中的任意一个都可以与任意接收缓冲区动态关联。实际关联通过设置 RXFBCONn 寄存器中的相应位来完成。每个 RXFBCONn 寄存器中都包含对应于每个过滤器的半字节。该半字节用于将一个特定过滤器与任意可用接收缓冲区相关联。用户固件可以将多个过滤器与任意一个特定接收缓冲区相关联。

除了动态关联过滤器与缓冲区之外，在模式 1 和 2 下，每个过滤器还可以与可用的接收屏蔽寄存器动态关联。MSELn 寄存器中的 FILn_m 位可用于将一个特定接收过滤器与接收屏蔽寄存器相链接。与过滤器 / 缓冲区关联一样，用户也可以将多个屏蔽器与一个特定接收过滤器相关联。

当某个过滤器发生匹配、一个报文被装入接收缓冲区中时，使能报文接收的过滤器编号会被装入 FILHIT 位。在模式 0 下，对于 RXB1，RXB1CON 寄存器将包含 FILHIT<2:0> 位。它们的编码如下：

- 101 = 接收过滤器 5 （RXF5）
- 100 = 接收过滤器 4 （RXF4）
- 011 = 接收过滤器 3 （RXF3）
- 010 = 接收过滤器 2 （RXF2）
- 001 = 接收过滤器 1 （RXF1）
- 000 = 接收过滤器 0 （RXF0）

注：	只有 RXB0CON 寄存器中的 RXB0DBEN 位置 1，允许 RXB0 报文续转到 RXB1 中时，才能出现 000 和 001。

通过 RXB0DBEN 位的编码，可以用类似于 FILHIT 位的方式使用这 3 位，并用以区分命中的是过滤器 RXF0 还是 RXF1，以及是在 RXB0 中还是在续转到 RXB1 中之后。

- 111 = 接收过滤器 1 （RXF1）
- 110 = 接收过滤器 0 （RXF0）
- 001 = 接收过滤器 1 （RXF1）
- 000 = 接收过滤器 0 （RXF0）

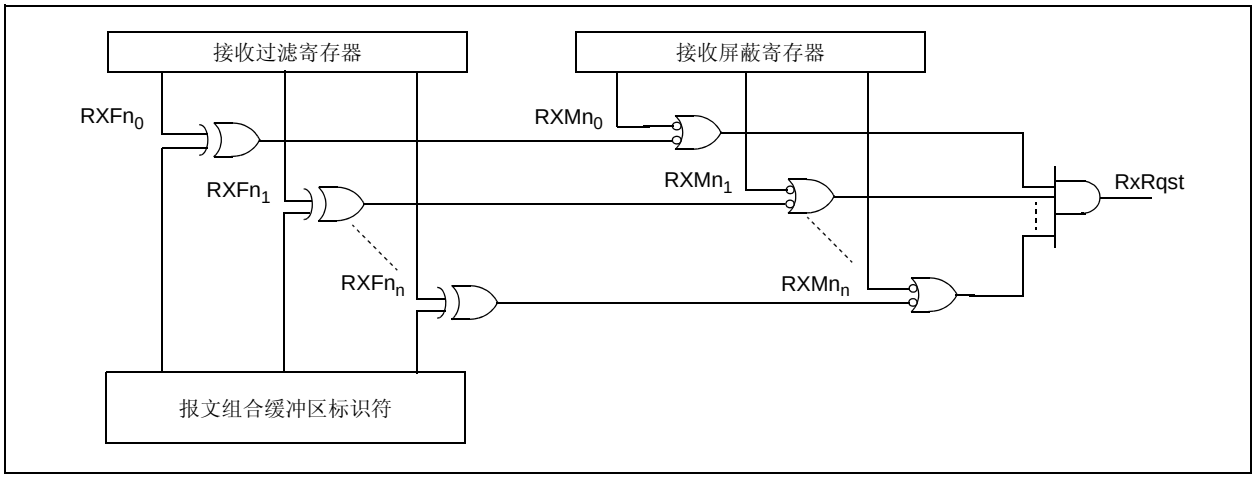
如果 RXB0DBEN 位清零，则有 6 个编码对应于 6 个过滤器。如果 RXB0DBEN 位置 1，则有 6 个编码对应于 6 个过滤器，外加两个额外的编码对应于报文续转到 RXB1 中的 RXF0 和 RXF1 过滤器。

在模式 1 和 2 下，每个缓冲区控制寄存器均包含 5 个过滤器命中位（FILHIT<4:0>）。二进制值 0 指示命中产生自 RXF0， 15 指示产生自 RXF15。

如果有多个接收过滤器发生匹配，FILHIT 位的编码将为发生匹配的编号最低的过滤器的二进制值。即，如果过滤器 RXF2 和过滤器 RXF4 发生匹配，FILHIT 中将装入 RXF2 的值。这实际上是按这种方式划分接收过滤器的优先级：过滤器的编号越低，优先级越高。报文将根据过滤器编号按升序与过滤器进行比较。

屏蔽器和过滤寄存器只能在 PIC18F66K80 系列器件处于配置模式时修改。

图 27-3： 报文接收屏蔽器和过滤器操作



27.9 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 协议采用不归零 （NRZ）编码，它不会对数据流的时钟进行编码。因此，接收节点必须恢复接收时钟，并与发送器的时钟进行同步。

由于振荡器和发送时间可能会因节点而不同，所以接收器必须具有与数据发送边沿同步的锁相环 （PLL），用以同步并维持接收器时钟。由于数据采用 NRZ 编码，所以需要进行位填充，以确保每隔 6 个位时间至少出现一个脉冲边沿，从而维持数字锁相环 （DPLL）同步。

PIC18F66K80 系列的位时序使用一个 DPLL 实现，它被配置为与传入数据进行同步，并为发送的数据提供标称时序。 DPLL 会将每个位时间拆分为多个时间段，时间段由称为*时间份额*（TQ）的最小时间周期组成。

在位时间帧内执行的总线时序功能（例如，与本地振荡器同步、网络传输延时补偿和采样点定位）都是通过 DPLL 的可编程位时序逻辑定义的。

CAN 总线上的所有器件必须使用相同的比特率。然而，并不要求所有的器件具有相同的主振荡器时钟频率。对于各个器件不同的时钟频率，必须通过适当地设置波特率预分频比和每个时间段中的时间份额数来调节比特率。

“ 标称比特率 ” 是假设使用理想发送器和理想振荡器，且无需重新同步的情况下，每秒发送的位数。标称比特率的最大值定义为 1 Mbps。

“ 标称位时间 ” 定义如下：

公式 27-1：

$$T_{BIT} = 1/\text{标称比特率}$$

可以认为，标称位时间可划分成几个互不重叠的时间段。这些时间段 （图 27-4）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）

因而，时间段（以及标称位时间）是由整数个称为时间份额或 TQ 的时间单位组成的（见图 27-4）。根据定义，标称位时间可设定为最少由 8 个 TQ 组成，最多由 25 个 TQ 组成。同样根据定义，最小标称位时间是 1 µs，对应于 1 Mbps 的最大比特率。实际持续时间由以下关系决定。

公式 27-2：

$$\text{标称位时间} = T_Q * (\text{Sync\_Seg} + \text{Prop\_Seg} + \text{Phase\_Seg1} + \text{Phase\_Seg2})$$

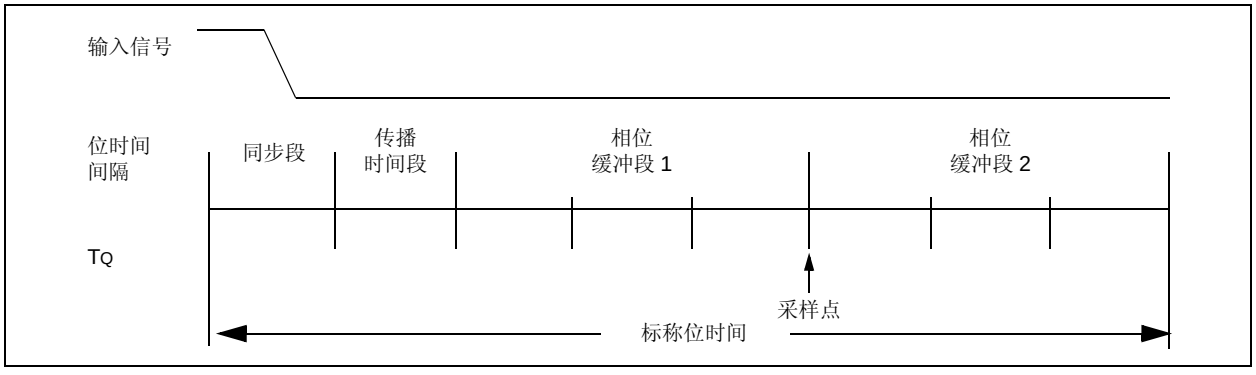
时间份额是基于振荡器周期而得到的固定时间单位。它还取决于可编程的波特率预分频比（整数值1至64），以及生成时钟时采用的固定 2 分频设置。数学表示形式为：

公式 27-3：

$$T_Q(\mu s) = (2 * (BRP + 1))/F_{OSC}(MHz)$$
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 BRGCON1<5:0> 二进制值表示的一个整数（0 至 63）。以上公式中的时钟频率指的是单片机使用的实际时钟频率。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 模式下使用同一 10 MHz 晶振，则实际频率为 FOSC = 40 MHz， TOSC = 25 ns。

图 27-4： 位时间划分



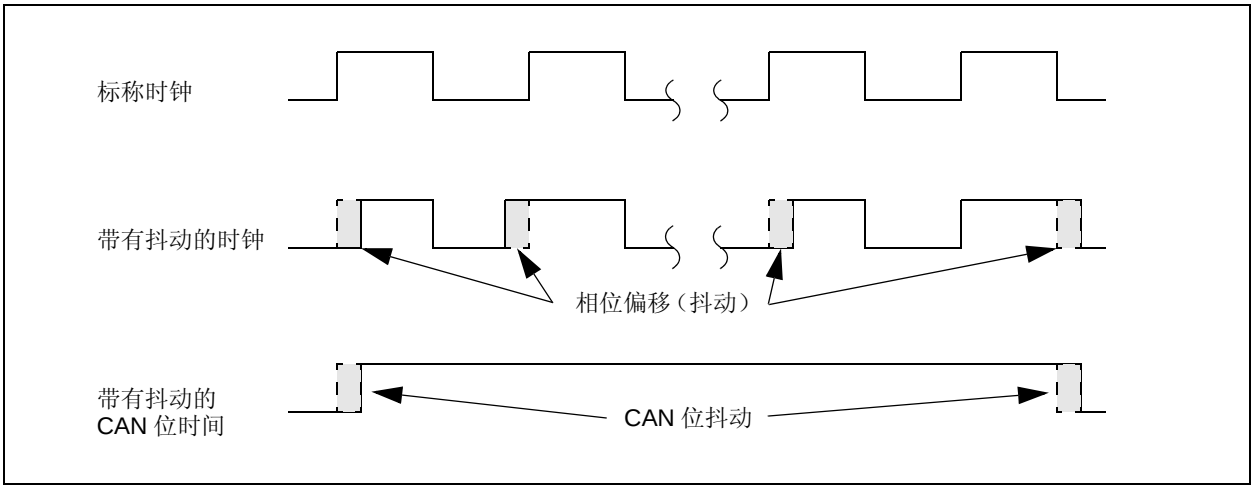
27.9.1 基于 HS-PLL 的振荡器中的外部时钟、内部时钟和可测量抖动

通过 PLL 电路产生单片机时钟频率会受抖动影响，也定义为相位抖动或相位偏移。对于 PIC18 增强型单片机，Microchip 将相位抖动（P_{jitter}）规定为 2%（高斯分布，3 个标准差以内，请参见表 31-7 中的参数 F13），将总抖动（T_{jitter}）规定为 2 * P_{jitter}。

CAN 协议采用位填充技术，会在 5 个相同极性位之后插入一个极性相反的位。这样，模块可以在无需重新同步（补偿抖动或相位误差）的情况下总共发送 10 位。

由于抖动误差的产生具有随机性，可以证明由抖动造成的总误差会随时间的推移而自行抵消。对于 10 位的周期，只需要增加两个抖动间隔来修正抖动引起的误差：其中一个间隔位于 10 位周期开始处，另一个位于结束处。图 27-5 显示了总体的效果。

图 27-5： 相位抖动对单片机时钟和 CAN 位时间的影响



将这些因素都考虑在内，可以证明，抖动和总频率误差之间的关系可以定义为：

$$\Delta f = \frac{T_{jitter}}{10 \times NBT} = \frac{2 \times P_{jitter}}{10 \times NBT}$$

其中，抖动使用时间来表示， NBT 是标称位时间。

举例来说，假设 CAN 比特率为 125 Kbps，可以得到 NBT 为 8 µs。对于通过 4x PLL 产生的 16 MHz 时钟，该时钟频率下的抖动为：

$$2\% \times \frac{1}{16\ MHz} = \frac{0.02}{16 \times 10^{6}} = 1.25ns$$

由此产生的频率误差为：

$$\frac{2 \times (1.25 \times 10^{-9})}{10 \times (8 \times 10^{-6})} = 3.125 \times 10^{-5} = 0.0031\%$$

表 27-2 显示了通过 PLL 产生的时钟和由于抖动产生的频率误差之间的关系（抖动引起的误差为 2%，高斯分布，3 个标准差以内），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 或 0.01%）。如果将抖动与振荡器漂移相加，可以得到总频率漂移为 0.0132%。表 27-3 显示了常用时钟频率和比特率的振荡器频率总误差（包括漂移和抖动）。

表 27-2：　　各种 PLL 产生的时钟速度下，由于抖动而产生的频率误差

PLL 输出	P_{jitter}	T_{jitter}	各种标称位时间（比特率）时的频率误差			
			8 µs（125 Kb/s）	4 µs（250 Kb/s）	2 µs（500 Kb/s）	1 µs（1 Mb/s）
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 27-3：　　各种 PLL 产生的时钟速度下的总频率误差（100 PPM 的振荡器漂移，包括抖动产生的误差）

标称 PLL 输出	各种标称位时间（比特率）时的频率误差			
	8 µs（125 Kb/s）	4 µs（250 Kb/s）	2 µs（500 Kb/s）	1 µs（1 Mb/s）
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%

27.9.2 时间份额

前面已经提到，时间份额是基于振荡器周期和波特率预分频比得到的固定时间单位。例 27-6 中给出了它与 TBIT 和标称比特率的关系。

例 27-6： 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 数量

标称比特率 （位 /s） = 1/TBIT

该频率 （FOSC）指的是所用的实际频率。例如，如果将 10 MHz 的外部信号与 PLL 配合使用，则实际频率将为 4 x 10 MHz，即等于 40 MHz。

情形 1：

对于 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} 位 /s （1 Mb/s）

情形 2：

对于 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 位 /s （625 Kb/s）

情形 3：

对于 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 位 /s （7.8 Kb/s）

不同节点中的振荡器频率必须进行协调，以提供系统范围规定的标称位时间。这意味着，所有振荡器的 TOSC 都必须为 TQ 的整因子。此外还需要注意，虽然 TQ 的数量可设定为 4 至 25，但可用的最小值为 8 个 TQ。位时间的长度小于 8 个 TQ 时，将无法保证正常工作。

27.9.3 同步段

位时间的这一部分用于同步总线上的各个 CAN 节点。输入信号的边沿需要出现在同步段中。持续时间为 1 个 TQ。

27.9.4 传播时间段

这部分位时间用来补偿网络内的物理延时。这些延时包括总线线路上的信号传播时间以及节点的内部延时。通过设置 PRSEG<2:0> 位，传播时间段的长度可设定为从 1 个 TQ 到 8 个 TQ。

27.9.5 相位缓冲段

相位缓冲段用于将接收位的采样点放置在标称位时间内的最佳位置。采样点出现在相位缓冲段 1 和相位缓冲段 2 之间。这些时间段可以通过重新同步过程延长或缩短。相位缓冲段 1 的结束决定位时间内的采样点。相位缓冲段 1 的持续时间可设定为 1 个 TQ 到 8 个 TQ。相位缓冲段 2 用于在下一次发送数据电平变换之前提供一个延时，它的持续时间也可以设定为从 1 个 TQ 到 8 个 TQ。但是，由于 IPT 的要求，相位缓冲段 2 的实际最小长度为 2 个 TQ，也可以定义为大于等于相位缓冲段 1 或信息处理时间（IPT）。采样点应尽可能晚，或者约位于位时间的 80% 处。

27.9.6 采样点

采样点是读总线电平并确定接收位的值的一个时间点。采样点出现在相位缓冲段 1 结束时。如果位时序很缓慢，并且包含许多 TQ，则可以指定在采样点处对总线线路进行多次采样。接收位的值将由 3 个值中的多数决定。三次采样中，一次在采样点处执行，两次在采样点之前执行，每次采样之间间隔的时间为 TQ/2。

27.9.7 信息处理时间

信息处理时间 （IPT）是从采样点开始的时间段，它保留用于计算随后的位电平。 CAN 规范将该时间定义为小于等于 2 个 TQ。PIC18F66K80 系列器件将该时间定义为 2 个 TQ。因此，相位缓冲段 2 的长度必须至少为 2 个 TQ。

27.10 同步

为了补偿总线上每个节点间振荡器频率的相移，每个 CAN 控制器必须能够与输入信号的相关信号沿同步。当检测到发送数据中的一个边沿时，逻辑电路会将该边沿的位置与预期时间（Sync_Seg）比较。然后，电路会根据需要调节相位缓冲段 1 和相位缓冲段 2 的值。有两种同步机制。

27.10.1 硬同步

仅当在总线空闲期间有一个从隐性转变到显性的边沿时（它指示报文传输的开始），才会执行硬同步。硬同步后，位时间计数器从 Sync_Seg 重新开始计数。硬同步会强制已发生的边沿处于重新开始的位时间的同步段之内。由于同步规则的原因，如果发生硬同步，则该位时间内将不会进行重新同步。

27.10.2 重新同步

由于重新同步的原因，相位缓冲段 1 可能会被延长，或者相位缓冲段 2 可能会被缩短。相位缓冲段的延长或缩短量存在一个由同步跳转宽度（Synchronization Jump Width，SJW）规定的上限。SJW 的值将与相位缓冲段 1（见图 27-6）相加，或与相位缓冲段 2（见图 27-7）相减。 SJW 可设定为介于 1 个 TQ 和 4 个 TQ 之间。

时钟信息将仅基于从隐性到显性的电平变换而得到。值相同的位连续出现的次数存在一个固定的最大数量限制，这一特性确保可以在帧期间与位流进行重新同步。

边沿的相位误差根据边沿相对于 Sync_Seg 的位置而得到，以 TQ 为单位测量。使用 TQ 定义的相位误差如下：

- e = 0，如果边沿位于 Sync_Seg 内。
- e > 0，如果边沿位于采样点之前。
- e < 0，如果边沿位于前一个位的采样点之后。

如果相位误差小于等于同步跳转宽度的设定值，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，并且如果相位误差为正，则相位缓冲段 1 会被延长，延长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，并且如果相位误差为负，则相位缓冲段 2 会被缩短，缩短量等于同步跳转宽度。

27.10.3 同步规则

- 一个位时间内仅允许一次同步。
- 只有在先前采样点检测到的值（先前读取的总线值）不同于紧跟在边沿之后的总线值时，才会使用一个边沿进行同步。
- 正在发送显性位的节点（它不会执行重新同步，因为从隐性变为显性的边沿带有正相位误差）除外，所有遵循规则 1 和 2 的其他隐性到显性的边沿都将用于重新同步。

图 27-6： 延长位周期 （将相位缓冲段 1 加上 SJW）

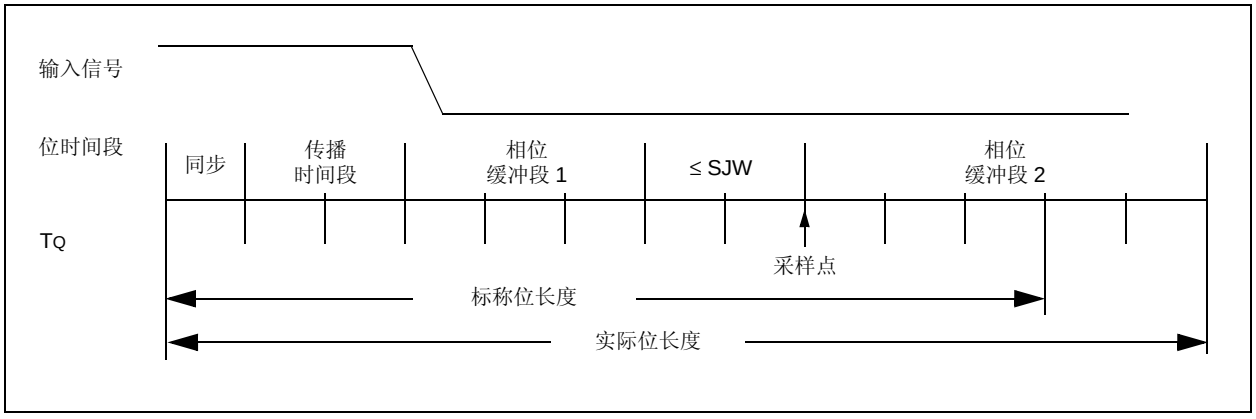
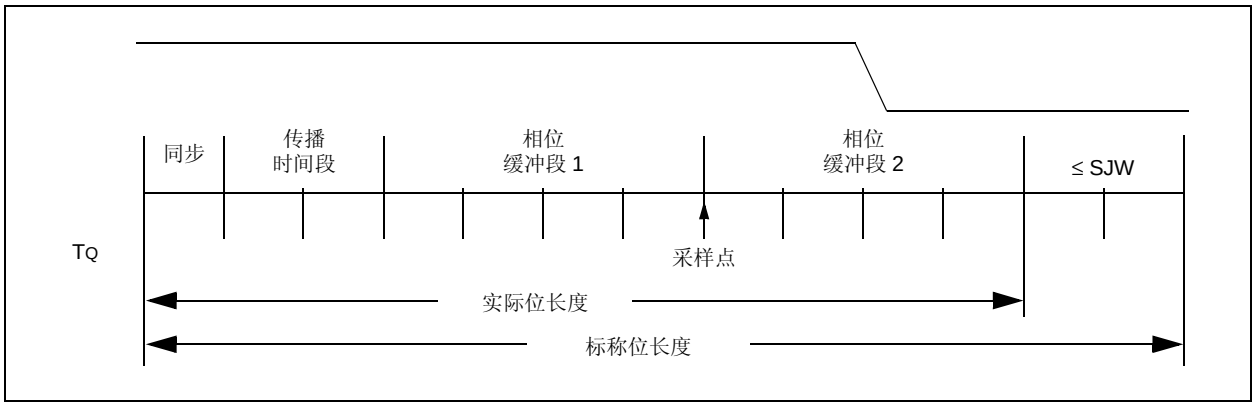


图 27-7：　　缩短位周期 （从相位缓冲段 2 中减去 SJW）



27.11　时间段编程

时间段编程存在一些要求：

- 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 的 CAN 波特率，使用 20 MHz 作为 FOSC。 TOSC 为 50 ns，波特率预分频值为 04h，得到的 TQ 为 500 ns。为了得到 125 kHz 的标称比特率，标称位时间必须为 8 µs 或 16 个 TQ。

对于 Sync_Seg 使用 1 个 TQ，对于 Prop_Seg 使用 2 个 TQ，对于相位缓冲段 1 使用 7 个 TQ 时，采样点将位于电平变换之后的 10 个 TQ 处。这为相位缓冲段 2 留下 6 个 TQ。

根据上面的规则，同步跳转宽度的最大值为 4 个 TQ。不过，通常只有在不同节点生成的时钟不精确或不稳定时（例如采用陶瓷谐振器），才需要使用很大的 SJW。通常， SJW 值为 1 就足够了。

27.12　振荡器容差

作为一个经验法则，在发送速率最高为 125 Kbps 的应用中，位时序要求允许应用使用陶瓷谐振器。要达到 CAN 协议的完全总线速度范围，则需要使用石英振荡器。关于振荡器的容差要求，请参见 ISO11898-1。

27.13 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。这些寄存器只能在 PIC18F66K80 系列器件处于配置模式时修改。

27.13.1 BRGCON1

BRP 位用于控制波特率预分频比。SJW<1:0> 位用于选择以 TQ 倍数表示的同步跳转宽度。

27.13.2 BRGCON2

PRSEG 位用于设置以 TQ 为单位表示的传播时间段长度。SEG1PH 位用于设置以 TQ 为单位表示的相位缓冲段 1 的长度。SAM 位用于控制对 RXCAN 引脚进行多少次采样。将该位设置为 1 会导致对总线执行三次采样：两次在采样点之前的 TQ/2 处，一次在正常采样点处（它处于相位缓冲段 1 结束处）。总线的值将由至少两次采样的读取值决定。如果 SAM 位设置为 0，则在采样点仅对 RXCAN 引脚采样一次。 SEG2PHTS 位用于控制如何确定相位缓冲段 2 的长度。如果该位设置为 1，则相位缓冲段 2 的长度由 BRGCON3 的 SEG2PH 位决定。如果 SEG2PHTS 位设置为 0，则相位缓冲段 2 的长度大于相位缓冲段 1 和信息处理时间（对于 PIC18F66K80 系列，它固定为 2 个 TQ）。

27.13.3 BRGCON3

如果 SEG2PHTS 位设置为 1， PHSEG2<2:0> 位将设置相位缓冲段 2 的长度（以 TQ 表示）。如果 SEG2PHTS 位设置为 0，则 PHSEG2<2:0> 位没有任何作用。

27.14 错误检测

CAN 协议提供了先进的错误检测机制。可以检测以下错误。

27.14.1 CRC 错误

利用循环冗余校验（Cyclic Redundancy Check，CRC），发送器会计算位序列（从帧起始开始直到数据字段结束）的一些特殊校验位。该 CRC 序列会被送入 CRC 字段。接收节点也会使用相同公式计算 CRC 序列，并与接收到的序列进行比较。如果检测到不匹配，则说明发生 CRC 错误，并会产生错误帧。报文将会重新发送。

27.14.2 应答错误

在报文的应答字段中，发送器会检查应答时隙（发送器将它以隐性位发送）是否包含显性位。如果不包含，则表示其他节点没有正确接收到该帧。说明发生了应答错误，模块会产生一个错误帧，报文必须重新发送。

27.14.3 格式错误

如果节点在 4 个时间段（包括帧结束（End-of-Frame，EOF）、帧间间隔、应答定界符或 CRC 定界符）的某个段中检测到显性位，则说明发生了格式错误，并会产生一个错误帧。报文将会重新发送。

27.14.4 位错误

在监视实际总线电平并将它与刚刚发送的位进行比较时，如果发送器发送显性位并检测到隐性位，或者如果它发送隐性位并检测到显性位，则表示发生了位错误。如果发送器在仲裁字段和应答时隙期间发送隐性位并检测到显性位，由于已经发生正常仲裁，所以不会产生位错误。

27.14.5 填充位错误

如果在帧起始（SOF）和 CRC 定界符之间，检测到 6 个连续的同极性位，则说明已经违反位填充规则。此时会出现填充位错误，并产生一个错误帧。报文将会重新发送。

27.14.6 错误状态

检测到的错误会通过错误帧告知所有其他节点。错误报文的发送会被中止，并尽快重新发送帧。此外，根据内部错误计数器的值，每个 CAN 节点将处于三种错误状态之一：“错误主动”、“错误被动”或“总线关闭”。错误主动状态是正常状态，这种状态下总线节点可以发送报文并激活错误帧（由显性位组成），不存在任何限制。在错误被动状态下，可以发送报文和被动错误帧（由隐性位组成）。总线关闭状态会让节点暂时无法参与总线通信。在这种状态下，既不能接收报文，也不能发送报文。

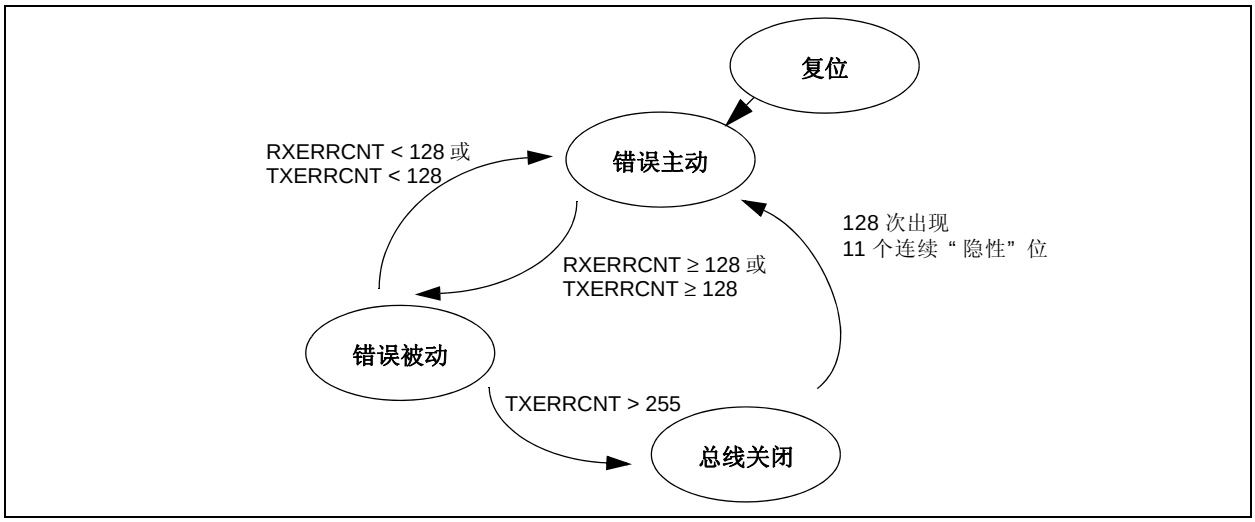
27.14.7 错误模式和错误计数器

PIC18F66K80 系列器件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）。两个计数器的值都可以由MCU读取。这两个计数器根据 CAN 总线规范进行递增或递减。

如果两个错误计数器均小于错误被动限制值128，PIC18F66K80 系列器件将处于错误主动状态。如果至少有一个错误计数器大于等于 128，则器件处于错误被动状态。如果发送错误计数器大于等于总线关闭限制值256，器件将进入总线关闭状态。器件将一直保持该状态，直到总线关闭恢复序列完成为止。总线关闭恢复序列由 128 组 11 个连续隐性位组成（见图 27-8）。请注意，在进入总线关闭状态之后，如果总线保持空闲的时间达到 128 x 11 个位时间，则 CAN 模块会恢复为错误主动状态，无需 MCU 进行任何干预。如果不希望出现这一情况，应通过错误中断服务程序来解决这一问题。MCU 可以通过 COMSTAT 寄存器读取 CAN 模块的当前错误模式。

此外，还有一个错误状态警告标志位 EWARN，如果至少有一个错误计数器大于等于错误警告限制值96，则该位会置 1。如果两个错误计数器都小于错误警告限制值，则 EWARN 会复位。

图 27-8：　　错误模式状态图



27.15　CAN 中断

模块具有几个中断源。其中每个中断都可以单独允许或禁止。PIR5 寄存器包含中断标志位。PIE5 寄存器包含 8 个主要中断允许位。 CANSTAT 寄存器中存在一组特殊的只读位（ICODE 位），它们可以与跳转表组合用于高效地处理中断。

在模式 1 和 2 下，错误中断和缓冲区除外，所有中断都具有一个中断源。任意错误中断源都可以将错误中断标志置 1。错误中断的中断源可以通过读取通信状态寄存器 COMSTAT 确定。在模式 1 和 2 下，存在两个中断允许 / 禁止和标志位 —— 一个对应于所有发送缓冲区，另一个对应于所有接收缓冲区。

中断可以分为两类：接收和发送中断。

与接收相关的中断有：

- 接收中断
- 唤醒中断
- 接收器溢出中断
- 接收器警告中断
- 接收器错误被动中断

与发送相关的中断有：

- 发送中断
- 发送器警告中断
- 发送器错误被动中断
- 总线关闭中断

27.15.1 中断编码位

为了简化用户固件中的中断处理过程，ECAN 模块提供了一组特殊的位。在模式 0 下，这些位是 CANSTAT 寄存器中的 ICODE<3:1>。在模式 1 和 2 下，这些位是 CANSTAT 寄存器中的 EICODE<4:0>。中断会在内部按这种方式设置优先级：优先级较高的中断分配较小的值。清除最高优先级的中断条件之后，ICODE 位将反映下一个待处理的最高优先级中断（如果有）的编码（见表 27-4）。请注意，ICODE 位中只会反映相关的中断允许位置 1 的那些中断源。

在模式 2 下，当发生接收报文中断时，EICODE 位将总是包含 10000。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。

27.15.2 发送中断

如果允许发送中断，则在相关的发送缓冲区为空，并准备好装入新报文时，将会产生中断。在模式 0 下，3 个专用发送缓冲区中的每一个都具有独立的中断允许 / 禁止和标志位。TXBnIF 位将被置 1，用以指示中断源。中断由 MCU 通过将 TXBnIF 位复位为 0 来清除。在模式 1 和 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 0 下， PIE5 中的 TXBnIE 和 PIR5 中的 TXBnIF 用于指示发送缓冲区是否已经完成报文发送。在模式 1 和 2 下，将不使用 PIR5、PIE5 和 IPR5 中相应的 TXBnIF、TXBnIE 和 TXBnIP。通过置 1 或清零 TXBnIE 和 B0IE 寄存器位，可以单独允许或禁止各个发送缓冲区中断。当发生共用中断时，用户固件必须通过查询所有发送缓冲区的 TXREQ 位来检测中断源。

27.15.3 接收中断

如果允许接收中断，则当成功接收到报文并装入相关的接收缓冲区中时，将会产生中断。接收到帧结束（EOF）字段后立即激活中断。

在模式 0 下，RXBnIF 位会被置 1，用以指示中断源。中断由 MCU 通过将 RXBnIF 位复位为 0 来清除。

在模式 1 和 2 下，所有接收缓冲区共用 PIE5、PIR5 和 IPR5 中相应的 RXBnIE、RXBnIF 和 RXBnIP。RXBnIE、RXBnIF 和 RXBnIP 位将不使用。各个接收缓冲区中断可以通过 TXBnIE 和 BIE0 寄存器单独控制。在模式 1 下，当发生共用接收中断时，用户固件必须通过查询每个接收缓冲区的 RXFUL 位来检测中断源。在模式 2 下，接收中断指示有新报文装入 FIFO 中。FIFO 可以通过使用 FIFO 指针位 FP 来读取。

表 27-4： ICODE<2:0> 的值

ICODE <2:0>	中断	布尔表达式
000	无	$\overline{ERR}\bullet\overline{WAK}\bullet\overline{TX0}\bullet\overline{TX1}\bullet\overline{TX2}\bullet\overline{RX0}\bullet\overline{RX1}$
001	错误	ERR
010	TXB2	$\overline{ERR}\bullet\overline{TX0}\bullet\overline{TX1}\bullet TX2$
011	TXB1	$\overline{ERR}\bullet\overline{TX0}\bullet TX1$
100	TXB0	$\overline{ERR}\bullet TX0$
101	RXB1	$\overline{ERR}\bullet\overline{TX0}\bullet\overline{TX1}\bullet\overline{TX2}\bullet\overline{RX0}\bullet RX1$
110	RXB0	$\overline{ERR}\bullet\overline{TX0}\bullet\overline{TX1}\bullet\overline{TX2}\bullet RX0$
111	唤醒中断	$\overline{ERR}\bullet\overline{TX0}\bullet\overline{TX1}\bullet\overline{TX2}\bullet\overline{RX0}\bullet\overline{RX1}\bullet WAK$

图注：

ERR = ERRIF * ERRIE　　RX0 = RXB0IF * RXB0IE
TX0 = TXB0IF * TXB0IE　　RX1 = RXB1IF * RXB1IE
TX1 = TXB1IF * TXB1IE　　WAK = WAKIF * WAKIE
TX2 = TXB2IF * TXB2IE

27.15.4 报文错误中断

在发送或接收报文期间发生错误时，报文错误标志 IRXIF 会被置 1，如果 IRXIE 位也置 1，则会产生中断。该功能是为了在与监听模式配合使用时，方便确定波特率。

27.15.5 总线活动唤醒中断

当 PIC18F66K80 系列器件处于休眠模式且允许总线活动唤醒中断时，如果在 CAN 总线上检测到活动，则会产生中断，并且 WAKIF 位会被置 1。该中断会导致 PIC18F66K80 系列器件退出休眠模式。中断由 MCU 通过清零 WAKIF 位来复位。

27.15.6 错误中断

当允许 CAN 模块错误中断（PIE5 中的 ERRIE）时，如果出现溢出条件，或者如果发送器或接收器的错误状态发生改变，则会产生中断。COMSTAT 中的错误标志会指示以下条件之一。

27.15.6.1 接收器溢出

当 MAB 已经组合了一个有效的接收报文（报文满足接收过滤器的条件），而与过滤器关联的接收缓冲区无法用于装入新报文时，将会发生溢出条件。COMSTAT 寄存器中的相关 RXBnOVFL 位将会置 1，指示存在溢出条件。该位必须由 MCU 清零。

27.15.6.2 接收器警告

接收错误计数器已达到 MCU 警告限制值 96。

27.15.6.3 发送器警告

发送错误计数器已达到 MCU 警告限制值 96。

27.15.6.4 接收器总线被动

当器件已经由于接收错误计数器大于等于 128 而进入错误被动状态时，将会出现该情况。

27.15.6.5 发送器总线被动

当器件已经由于发送错误计数器大于等于 128 而进入错误被动状态时，将会出现该情况。

27.15.6.6 总线关闭

发送错误计数器已超出 255，器件已进入总线关闭状态。

注：

28.0 CPU 的特殊功能

PIC18F66K80 系列器件包含的功能旨在最大限度地提高系统可靠性，并通过减少外部元件将系统成本降到最低。这些功能包括：

- 振荡器选择
- 复位：
  - 上电复位 （POR）
  - 上电延时定时器 （PWRT）
  - 振荡器起振定时器 （OST）
  - 欠压复位 （BOR）
- 中断
- 看门狗定时器 （WDT）和片上稳压器
- 故障保护时钟监视器
- 双速启动
- 代码保护
- ID 存储单元
- 在线串行编程

要根据具体应用对频率、功耗、精度和成本的要求来选择振荡器。在**第 3.0 节 “ 振荡器配置 ”** 中详细讨论了所有的选项。

在本数据手册的前面几章中已完整地讨论了器件的复位和中断。

除了为复位提供了上电延时定时器和振荡器起振定时器之外，PIC18F66K80 系列器件还提供了一个看门狗定时器，该定时器可通过配置位永久使能或用软件控制（如果配置为禁止）。

器件自带的内部 RC 振荡器（LF-INTOSC）还提供了故障保护时钟监视器 （FSCM）和双速启动这两个额外的功能。FSCM 对外设时钟进行后台监视，并在外设时钟发生故障时自动切换时钟源。双速启动使得几乎可在启动发生那一刻立即执行代码，同时主时钟源进行其起振延时。

通过设置相应的配置寄存器位可以使能和配置所有这些功能。

28.1 配置位

可以通过对配置位编程（读为 0）或不编程（读为 1）来选择不同的器件配置。这些配置位被映射到程序存储器以 300000h 开始的单元中。

用户会注意到地址 300000h 超出了用户程序存储空间范围。事实上，它属于配置存储空间（300000h-3FFFFFh），这一空间仅能通过表读和表写进行访问。

通过软件对配置寄存器进行编程的方式与对闪存进行编程的方式类似。 EECON1 寄存器中的 WR 位启动自定时写入配置寄存器。在正常操作模式下，TBLPTR 指向配置寄存器的 TBLWT 指令会设置写配置寄存器要用到的地址和数据。将 WR 位置 1 会启动对配置寄存器的长写操作。配置寄存器一次被写入一个字节。TBLWT 指令可以将 1 或 0 写入单元来改写配置单元的内容。关于闪存编程的更多详细信息，请参见**第 7.5 节 “ 写闪存程序存储器 ”**。

表 28-1：　　配置位和器件 ID

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	默认 / 未编程值
300000h	CONFIG1L	—	XINST	—	SOSCSEL1	SOSCSEL0	INTOSCSEL	—	$\overline{\text{RETEN}}$	-1-1 11-1
300001h	CONFIG1H	IESO	FCMEN	—	PLLCFG	FOSC3	FOSC2	FOSC1	FOSC0	00-0 1000
300002h	CONFIG2L	—	BORPWR1	BORWPR0	BORV1	BORV0	BOREN1	BOREN0	$\overline{\text{PWRTEN}}$	-111 1111
300003h	CONFIG2H	—	WDTPS4	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN1	WDTEN0	-111 1111
300005h	CONFIG3H	MCLRE	—	—	—	MSSPMSK	T3CKMX(1,3)	T0CKMX(1)	CANMX	1--- 1qq1
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	—	—	BBSIZ0	—	—	—	STVREN	1--1 ---1
300008h	CONFIG5L	—	—	—	—	CP3	CP2	CP1	CP0	---- 1111
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah	CONFIG6L	—	—	—	—	WRT3	WRT2	WRT1	WRT0	---- 1111
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch	CONFIG7L	—	—	—	—	EBTR3	EBTR2	EBTR1	EBTR0	---- 1111
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh	DEVID1(2)	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx
3FFFFFh	DEVID2(2)	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	xxxx xxxx

图注：　x = 未知， u = 不变， - = 未实现， q = 值取决于具体条件。阴影单元未实现，读为 0。

注　1：　仅在 64 引脚器件 （PIC18F6XK80）上实现。

2：　DEVID1 值请参见寄存器 28-13。 DEVID 寄存器为只读寄存器，用户不能对其进行编程。

3：　在 28 引脚、 40 引脚和 44 引脚器件上保持为 0。

寄存器 28-1： CONFIG1L：配置寄存器 1 的低字节 （字节地址为 300000h）

U-0	R/P-1	U-0	R/P-1	R/P-1	R/P-1	U-0	R/P-1
—	XINST	—	SOSCSEL1	SOSCSEL0	INTOSCSEL	—	$\overline{\text{RETEN}}$
bit 7							bit 0

图注：	P = 可编程位		
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 **未实现：**读为 0

bit 6 **XINST：**扩展指令集使能位
1 = 使能指令集扩展和变址寻址模式
0 = 禁止指令集扩展和变址寻址模式 （传统模式）

bit 5 **未实现：**读为 0

bit 4-3 **SOSCSEL<1:0>：**SOSC 功耗选择和模式配置位
11 = 选择高功耗 SOSC 电路
10 = 数字 （SCLKI）模式；使能 RC0 和 RC1 的 I/O 端口功能
01 = 选择低功耗 SOSC 电路
00 = 保留

bit 2 **INTOSCSEL：**LF-INTOSC 低功耗使能位
1 = 休眠期间 LF-INTOSC 处于高功耗模式
0 = 休眠期间 LF-INTOSC 处于低功耗模式

bit 1 **未实现：**读为 0

bit 0 **$\overline{\text{RETEN}}$：**V_{REG} 休眠使能位
1 = 禁止超低功耗稳压器。休眠模式下的稳压器功耗由 REGSLP （WDTCON<7>）控制。
0 = 使能超低功耗稳压器。休眠模式下的稳压器功耗由 SRETEN （WDTCON<4>）控制。

寄存器 28-2： CONFIG1H：配置寄存器 1 的高字节 （字节地址为 300001h）

R/P-0	R/P-0	U-0	U-0	R/P-1	R/P-0	R/P-0	R/P-0
IESO	FCMEN	—	PLLCFG(1)	FOSC3(2)	FOSC2(2)	FOSC1(2)	FOSC0(2)
bit 7							bit 0

图注：	P = 可编程位		
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 **IESO**：内部 / 外部振荡器切换位
1 = 使能双速启动
0 = 禁止双速启动

bit 6 **FCMEN**：故障保护时钟监视器使能位
1 = 使能故障保护时钟监视器
0 = 禁止故障保护时钟监视器

bit 5 **未实现**：读为 0

bit 4 **PLLCFG**：4X PLL 使能位 (1)
1 = 振荡器进行 4 倍频
0 = 直接使用振荡器

bit 3-0 **FOSC<3:0>**：振荡器选择位 (2)
1101 = EC1， EC 振荡器 （**低功耗， DC-160 kHz**）
1100 = EC1IO， EC 振荡器， RA6 用作 CLKOUT 引脚 （**低功耗， DC-160 kHz**）
1011 = EC2， EC 振荡器 （**中等功耗， 160 kHz-16 MHz**）
1010 = EC2IO， EC 振荡器， RA6 用作 CLKOUT 引脚 （**中等功耗， 160 kHz-16 MHz**）
0101 = EC3， EC 振荡器 （**高功耗， 16 MHz-64 MHz**）
0100 = EC3IO， EC 振荡器， RA6 用作 CLKOUT 引脚 （**高功耗， 16 MHz-64 MHz**）
0011 = HS1， HS 振荡器 （**中等功耗， 4 MHz-16 MHz**）
0010 = HS2， HS 振荡器 （**高功耗， 16 MHz-25 MHz**）
0001 = XT 振荡器
0000 = LP 振荡器
0111 = RC，外部 RC 振荡器
0110 = RCIO，外部 RC 振荡器， RA6 用作 CKLOUT 引脚
1000 = INTIO2，内部 RC 振荡器
1001 = INTIO1，内部 RC 振荡器， RA6 用作 CKLOUT 引脚

注 1： 对于 INTIOx PLL 模式无效。
2： INTIO + PLL 只能通过 PLLEN 位（OSCTUNE<6>）使能。其他 PLL 模式可以通过 PLLEN 位或 PLLCFG（CONFIG1H<4>）位使能。

寄存器 28-3： CONFIG2L：配置寄存器 2 的低字节 （字节地址为 300002h）

U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	BORPWR1(1)	BORPWR0(1)	BORV1(1)	BORV0(1)	BOREN1(2)	BOREN0(2)	$\overline{\text{PWRTEN}}$(2)
bit 7							bit 0

图注：	P = 可编程位		
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 **未实现：**读为 0

bit 6-5 **BORPWR<1:0>：**BORMV 功耗级别位 (1)
11 = 选择 ZPBORVMV 而不是 BORMV
10 = BORMV 设置为高功耗级别
01 = BORMV 设置为中等功耗级别
00 = BORMV 设置为低功耗级别

bit 4-3 **BORV<1:0>：**欠压复位电压位 (1)
11 = BVDD 设置为 1.8V
10 = BVDD 设置为 2.0V
01 = BVDD 设置为 2.7V
00 = BVDD 设置为 3.0V

bit 2-1 **BOREN<1:0>：**欠压复位使能位 (2)
11 = 只能由硬件使能欠压复位（禁止 SBOREN）
10 = 由硬件使能欠压复位，休眠模式下被禁止（禁止 SBOREN）
01 = 用软件使能和控制欠压复位（使能 SBOREN）
00 = 用硬件和软件禁止欠压复位

bit 0 **$\overline{\text{PWRTEN}}$：**上电延时定时器使能位 (2)
1 = 禁止 PWRT
0 = 使能 PWRT

注 1： 请参见**第 31.1 节 “ 直流特性：供电电压 PIC18F66K80 系列 （工业级 / 扩展级） ”**获取规范信息。
2： 上电延时定时器与欠压复位是相互独立的，可以分别控制两者的操作。

寄存器 28-4： CONFIG2H：配置寄存器 2 的高字节 （字节地址为 300003h）

U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	WDTPS4	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN1	WDTEN0
bit 7							bit 0

图注：	P = 可编程位		
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 **未实现：**读为 0

bit 6-2 **WDTPS<4:0>：**看门狗定时器后分频比选择位

11111 = 保留
10100 = 1:1,048,576 （4,194.304s）
10011 = 1:524,288 （2,097.152s）
10010 = 1:262,144 （1,048.576s）
10001 = 1:131,072 （524.288s）
10000 = 1:65,536 （262.144s）
01111 = 1:32,768 （131.072s）
01110 = 1:16,384 （65.536s）
01101 = 1:8,192 （32.768s）
01100 = 1:4,096 （16.384s）
01011 = 1:2,048 （8.192s）
01010 = 1:1,024 （4.096s）
01001 = 1:512 （2.048s）
01000 = 1:256 （1.024s）
00111 = 1:128 （512 ms）
00110 = 1:64 （256 ms）
00101 = 1:32 （128 ms）
00100 = 1:16 （64 ms）
00011 = 1:8 （32 ms）
00010 = 1:4 （16 ms）
00001 = 1:2 （8 ms）
00000 = 1:1 （4 ms）

bit 1-0 **WDTEN<1:0>：**看门狗定时器使能位

11 = 由硬件使能 WDT ； 禁止 SWDTEN 位
10 = WDT 由 SWDTEN 位设置控制
01 = 仅在器件处于活动状态时使能 WDT，在器件处于休眠模式时禁止；禁止 SWDTEN 位
00 = 由硬件禁止 WDT ； 禁止 SWDTEN 位

寄存器 28-5： CONFIG3H：配置寄存器 3 的高字节 （字节地址为 300005h）

R/P-1	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1
MCLRE	—	—	—	MSSPMSK	T3CKMX(1)	T0CKMX(1)	CANMX
bit 7							bit 0

图注：	P = 可编程位		
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 **MCLRE：** $\overline{\text{MCLR}}$ 引脚使能位
1 = 使能 $\overline{\text{MCLR}}$ 引脚；禁止 RE3 输入引脚
0 = 使能 RE3 输入引脚；禁止 $\overline{\text{MCLR}}$

bit 6-4 **未实现：** 读为 0

bit 3 **MSSPMSK：** MSSP V3 7 位地址掩码模式使能位
1 = 使能 7 位地址掩码模式
0 = 使能 5 位地址掩码模式

bit 2 **T3CKMX：** Timer3 时钟输入复用位 (1)
1 = Timer3 从 64 引脚封装上的 RG2/T3CKI 引脚获取它的时钟输入
0 = Timer3 从 64 引脚封装上的 RB5/T3CKI 引脚获取它的时钟输入

bit 1 **T0CKMX：** Timer0 时钟输入复用位 (1)
1 = Timer0 从 64 引脚封装上的 RB5/T0CKI 引脚获取它的时钟输入
0 = Timer0 从 64 引脚封装上的 RG4/T0CKI 引脚获取它的时钟输入

bit 0 **CANMX：** ECAN 复用位
1 = CANTX 和 CANRX 引脚分别位于 RB2 和 RB3 上
0 = CANTX 和 CANRX 引脚分别位于 RC6 和 RC7 上（28 引脚和 40/44 引脚封装）或分别位于 RE4 和 RE5 上（64 引脚封装）

注 1： 仅在 64 引脚器件 （PIC18F6XK80）上实现。在 28 引脚、 40 引脚和 44 引脚器件上保持为 0。

寄存器 28-6： CONFIG4L：配置寄存器 4 的低字节 （字节地址为 300006h）

R/P-1	U-0	U-0	R/P-0	U-0	U-0	U-0	R/P-1
$\overline{\text{DEBUG}}$	—	—	BBSIZ0	—	—	—	STVREN
bit 7							bit 0

图注：	P = 可编程位		
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 **$\overline{\text{DEBUG}}$：**后台调试器使能位
1 = 禁止后台调试器， RB6 和 RB7 被配置为通用 I/O 引脚
0 = 使能后台调试器， RB6 和 RB7 专用于在线调试

bit 6-5 **未实现：**读为 0

bit 4 **BBSIZ0：**引导区大小选择位
1 = 2 kW 引导区大小
0 = 1 kW 引导区大小

bit 3-1 **未实现：**读为 0

bit 0 **STVREN：**堆栈满 / 下溢复位使能位
1 = 堆栈满 / 下溢导致复位
0 = 堆栈满 / 下溢不会导致复位

寄存器 28-7：　　CONFIG5L：配置寄存器 5 的低字节 （字节地址为 300008h）

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	CP3	CP2	CP1	CP0
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 7-4　**未实现：**读为 0

bit 3　**CP3：**代码保护位
1 = Block 3 不受代码保护 (1)
0 = Block 3 受代码保护 (1)

bit 2　**CP2：**代码保护位
1 = Block 2 不受代码保护 (1)
0 = Block 2 受代码保护 (1)

bit 1　**CP1：**代码保护位
1 = Block 1 不受代码保护 (1)
0 = Block 1 受代码保护 (1)

bit 0　**CP0：**代码保护位
1 = Block 0 不受代码保护 (1)
0 = Block 0 受代码保护 (1)

注　1： 关于存储块大小的信息，请参见图 28-6。

寄存器 28-8：　　CONFIG5H：配置寄存器 5 的高字节 （字节地址为 300009h）

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD	CPB	—	—	—	—	—	—
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　　**CPD**：数据 EEPROM 代码保护位
1 = 数据 EEPROM 不受代码保护
0 = 数据 EEPROM 受代码保护

bit 6　　**CPB**：引导区代码保护位
1 = 引导区不受代码保护 **(1)**
0 = 引导区受代码保护 **(1)**

bit 5-0　　**未实现**：读为 0

注　1： 关于存储块大小的信息，请参见图 28-6。引导区大小随 BBSIZ0 而变化。

寄存器 28-9：　　CONFIG6L：配置寄存器 6 的低字节 （字节地址为 30000Ah）

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	WRT3	WRT2	WRT1	WRT0
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 7-4　**未实现：**读为 0

bit 3　**WRT3：**写保护位
1 = Block 3 不受写保护 **(1)**
0 = Block 3 受写保护 **(1)**

bit 2　**WRT2：**写保护位
1 = Block 2 不受写保护 **(1)**
0 = Block 2 受写保护 **(1)**

bit 1　**WRT1：**写保护位
1 = Block 1 不受写保护 **(1)**
0 = Block 1 受写保护 **(1)**

bit 0　**WRT0：**写保护位
1 = Block 0 不受写保护 **(1)**
0 = Block 0 受写保护 **(1)**

注　1： 关于存储块大小的信息，请参见图 28-6。

寄存器 28-10：　CONFIG6H：配置寄存器 6 的高字节 （字节地址为 30000Bh）

R/C-1	R/C-1	R-1	U-0	U-0	U-0	U-0	U-0
WRTD	WRTB	WRTC(1)	—	—	—	—	—
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　**WRTD：**数据 EEPROM 写保护位
1 = 数据 EEPROM 不受写保护
0 = 数据 EEPROM 受写保护

bit 6　**WRTB：**引导区写保护位
1 = 引导区不受写保护 (2)
0 = 引导区受写保护 (2)

bit 5　**WRTC：**配置寄存器写保护位 (1)
1 = 配置寄存器不受写保护 (2)
0 = 配置寄存器受写保护 (2)

bit 4-0　**未实现：**读为 0

注　1： 在正常执行模式下，该位是只读位；该位仅在编程模式下可写入。
2： 关于存储块大小的信息，请参见图 28-6。

寄存器 28-11： CONFIG7L：配置寄存器 7 的低字节 （字节地址为 30000Ch）

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	EBTR3	EBTR2	EBTR1	EBTR0
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 7-4 **未实现：**读为 0

bit 3 **EBTR3：**表读保护位
1 = 其他块可对 Block 3 执行表读操作 **(1)**
0 = 禁止其他块对 Block 3 执行表读操作 **(1)**

bit 2 **EBTR2：**表读保护位
1 = 其他块可对 Block 2 执行表读操作 **(1)**
0 = 禁止其他块对 Block 2 执行表读操作 **(1)**

bit 1 **EBTR1：**表读保护位
1 = 其他块可对 Block 1 执行表读操作 **(1)**
0 = 禁止其他块对 Block 1 执行表读操作 **(1)**

bit 0 **EBTR0：**表读保护位
1 = 其他块可对 Block 0 执行表读操作 **(1)**
0 = 禁止其他块对 Block 0 执行表读操作 **(1)**

注 1： 关于存储块大小的信息，请参见图 28-6。

寄存器 28-12：　　CONFIG7H：配置寄存器 7 的高字节 （字节地址为 30000Dh）

U-0	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB	—	—	—	—	—	—
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　　**未实现：**读为 0

bit 6　　**EBTRB：**引导区表读保护位
1 = 其他块可对引导区执行表读操作 (1)
0 = 禁止其他块对引导区执行表读操作 (1)

bit 5-0　　**未实现：**读为 0

注　1： 关于存储块大小的信息，请参见图 28-6。

寄存器 28-13： DEVID1：PIC18F66K80 系列的器件 ID 寄存器 1

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
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 **DEV<2:0>**：器件 ID 位
这些位与器件 ID 寄存器 2 中的 DEV<10:3> 位一起用于标识器件编号。
000 = PIC18F46K80 和 PIC18LF26K80
001 = PIC18F26K80 和 PIC18LF65K80
010 = PIC18F65K80 和 PIC18LF45K80
011 = PIC18F45K80 和 PIC18LF25K80
100 = PIC18F25K80
110 = PIC18LF66K80
111 = PIC18F66K80 和 PIC18LF46K80

bit 4-0 **REV<4:0>**：版本 ID 位
这些位用于指示器件版本。

寄存器 28-14： DEVID2：PIC18F66K80 系列的器件 ID 寄存器 2

R	R	R	R	R	R	R	R
DEV10(1)	DEV9(1)	DEV8(1)	DEV7(1)	DEV6(1)	DEV5(1)	DEV4(1)	DEV3(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 7-0 **DEV<10:3>**：器件 ID 位 (1)
这些位与器件 ID 寄存器 1 中的 DEV<2:0> 位一起用于标识器件编号。

注 1： DEV<10:3> 的值可能会用于其他器件。特定器件总是通过使用整个 DEV<10:0> 位序列来标识的。

28.2 看门狗定时器 （WDT）

对于 PIC18F66K80 系列器件，WDT 由 LF-INTOSC 时钟源驱动。当使能WDT时，时钟源也将同时使能。WDT 周期的标称值为 4 ms，其稳定性与 LF-INTOSC 振荡器相同。

4 ms 的 WDT 周期将与 16 位后分频器的值相乘来得到更长的时间周期。通过配置寄存器 2H 中的位来控制一个多路开关以对 WDT 后分频器的输出进行选择。因此可获得的周期范围为 4 ms 至 4,194 秒（大约 1 小时）。当发生以下任一事件时，WDT 和后分频器将被清零，这些事件包括：执行了 SLEEP 或 CLRWDT 指令、改变了 IRCF 位 （OSCCON<6:4>）或发生了时钟故障。

WDT 可工作于 4 种模式之一，模式由 WDTEN<1:0>（CONFIG2H<1:0>）决定。这 4 种模式为：

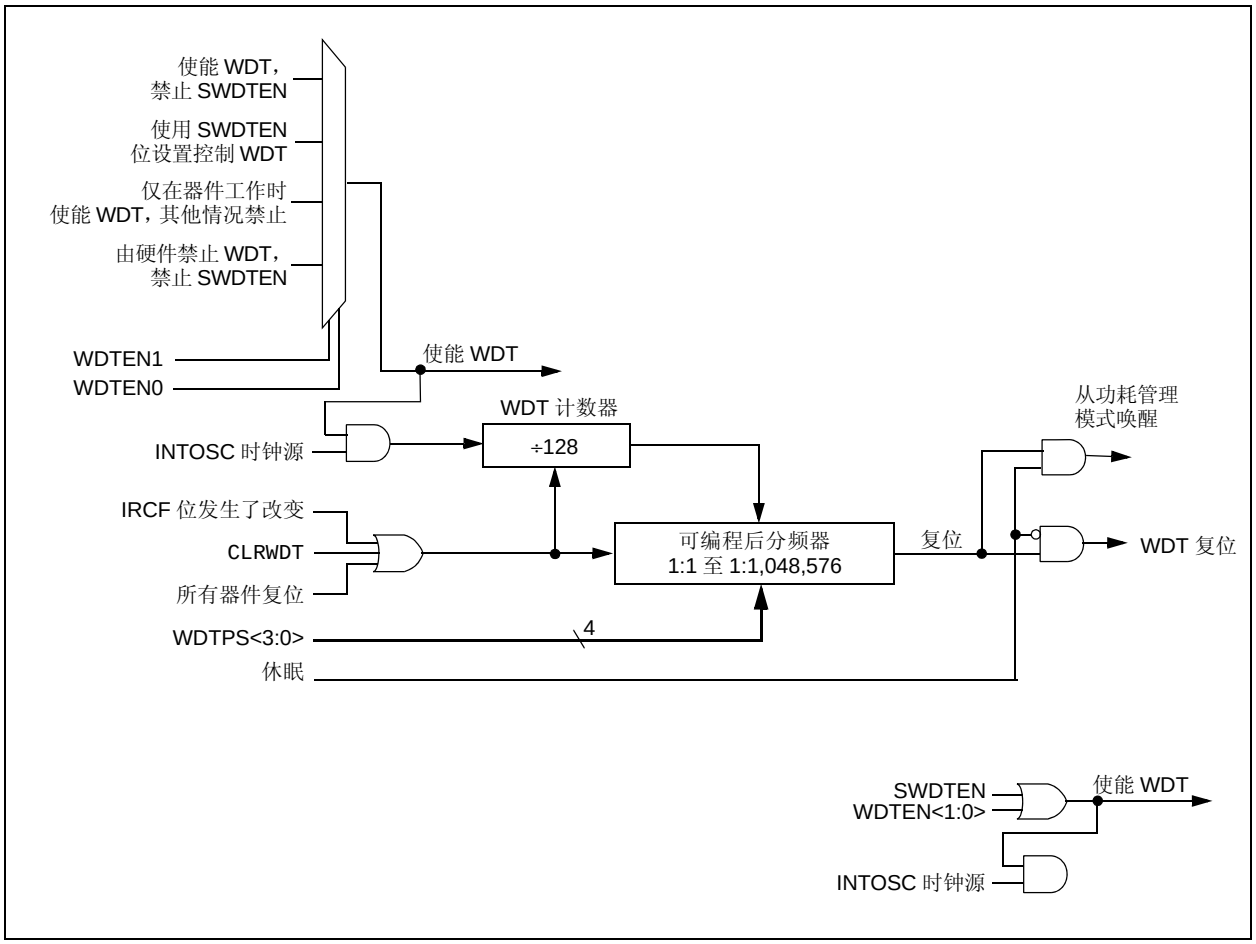
- 使能 WDT
- 禁止 WDT
- WDT 由软件控制， SWDTEN （WDTCON<0>）
- WDT
  - 正常工作期间使能
  - 休眠期间禁止

注 1： 当执行 CLRWDT 和 SLEEP 指令时，WDT 和后分频器的计数值将被清零。

2： 更改 IRCF 位（OSCCON<6:4>）的设置会清零 WDT 和后分频器的计数值。

3： 当执行 CLRWDT 指令时，后分频器的计数值将被清零。

图 28-1： WDT 框图



28.2.1 控制寄存器

寄存器 28-15 所示为 WDTCON 寄存器。它是可读写寄存器并包含一个控制位，仅当用配置位禁止 WDT 时，该控制位才允许使用软件改写 WDT 使能配置位。

寄存器 28-15： WDTCON：看门狗定时器控制寄存器

R/W-0	U-0	R-x	R/W-0	U-0	R/W-x	R/W-x	R/W-0
REGSLP(3)	—	ULPLVL	SRETEN(2)	—	ULPEN	ULPSINK	SWDTEN(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 **REGSLP：**稳压器休眠使能位 (3)
1 = 当使能器件的休眠模式时，稳压器进入低功耗模式
0 = 当激活器件的休眠模式时，稳压器保持在正常模式

bit 6 **未实现：**读为 0

bit 5 **ULPLVL：**超低功耗唤醒输出位
只有 ULPEN = 1 时才有效。
1 = RA0 引脚上的电压 > ~ 0.5V
0 = RA0 引脚上的电压 < ~ 0.5V

bit 4 **SRETEN：**稳压器休眠禁止位 (2)
1 = 如果 $\overline{\text{RETEN}}$ （CONFIG1L<0>） = 0，并且稳压器已使能，则器件在休眠时进入超低功耗模式
0 = 稳压器在使能器件休眠模式时开启，并且低功耗模式由 REGSLP 控制

bit 3 **未实现：**读为 0

bit 2 **ULPEN：**超低功耗唤醒模块使能位
1 = 使能超低功耗唤醒模块； ULPLVL 位指示比较器输出
0 = 禁止超低功耗唤醒模块

bit 1 **ULPSINK：**超低功耗唤醒灌电流使能位
只有 ULPEN = 1 时才有效。
1 = 使能超低功耗唤醒灌电流
0 = 禁止超低功耗唤醒灌电流

bit 0 **SWDTEN：**软件控制的看门狗定时器使能位 (1)
1 = 看门狗定时器开启
0 = 看门狗定时器关闭

注 1： 如果使能了配置位 WDTEN<1:0>，则该位不起作用。
2： 该位仅在 $\overline{\text{RETEN}}$ = 0 时可用。
3： 在 PIC18LF 器件上，该位会被禁止。

表 28-2： 看门狗定时器寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RCON	IPEN	SBOREN	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
WDTCON	REGSLP	—	ULPLVL	SRETEN	—	ULPEN	ULPSINK	SWDTEN

图注： — = 未实现，读为 0。看门狗定时器不使用阴影单元。

28.3 片上稳压器

PIC18F66K80 系列中的所有器件都使用标称值为 3.3V 的电压为其内核数字逻辑供电。对于需要工作在一个更高的典型电压值如 5V 的设计中，系列中的所有器件都包含两个片上稳压器，可使器件内核逻辑在 VDD 下工作。这两个稳压器是：

- 常规片上稳压器
- 超低功耗片上稳压器

这些稳压器的硬件配置是相同的，将在**第 28.3.1 节**中进行介绍。稳压器之间惟一的区别与器件进入休眠时有关，将在**第 28.3.3 节**中进行介绍。

28.3.1 稳压器使能模式（PIC18FXXKXX 器件）

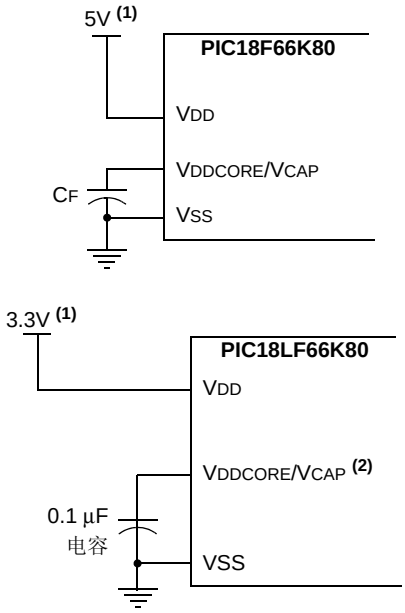
在 PIC18FXXKXX 器件上，使能稳压器时，必须将一个低 ESR 滤波电容连接到 VDDCORE/VCAP 引脚（见图 28-2）。这有助于维持稳压器的稳定性。**第 31.1 节“直流特性”**给出了滤波电容的推荐值。

28.3.2 稳压器禁止模式（PIC18LFXXKXX 器件）

在 PIC18**LF**XXKXX 器件上，稳压器会被禁止，内核通过 VDD 直接供电。VDD 的电压不得超出所规定的 VDDCORE 电压。VDDCORE/VCAP 引脚上应连接一个 0.1 µF 的电容。

在 PIC18FXXKXX 器件上，总电压预算很紧。稳压器应可在最低 1.8V 的电压下运行器件。当 VDD 降至低于 3.3V 时，稳压器将不再进行稳压，但输出电压会追随输入，直到 VDD 达到 1.8V 为止。低于该电压时，稳压器的输出可能会降至 0V。

图 28-2： F 和 LF 器件的连接



注 1： 这些为典型的工作电压值。关于VDD和VDDCORE的全部工作电压范围，请参见**第 31.1 节“直流特性”**。

2： 当禁止稳压器时，VDDCORE/VCAP 必须连接一个 0.1 µF 的电容。

28.3.3 休眠模式下的稳压器操作

两种稳压器的操作之间的区别在于休眠模式。在使能稳压器的模式下，超低功耗稳压器会使器件电流达到最低。

通过将 REGSLP 位（WDTCON<7>）置 1，在器件进入休眠模式时，片上稳压器可以进入较低功耗的模式。这会将稳压器置为待机模式，从而极大地降低器件消耗的电流。

片上稳压器也可以进入超低功耗模式，这会使器件在使能稳压器情况下消耗的电流达到尽可能最低。该模式通过 $\overline{\text{RETEN}}$ 位（CONFIG1L<0>）和 SRETEN 位（WDTCON<4>）来控制。

表 28-3 列出了稳压器的各种工作模式。

当超低功耗稳压器处于休眠模式时，芯片中的内部参考电压会被关闭，任何使用内部参考电压的中断都不会唤醒器件。如果使能 BOR 或 LVD，稳压器会将内部参考电压保持开启，此时将无法达到尽可能最低的电流。

在休眠模式下使用超低功耗稳压器时，器件大约需要 250 µs 的时间才会在唤醒之后执行代码。

表 28-3： 休眠模式稳压器设置 (1)

器件	功耗模式	REGSLP WDTCON<7>	SRETEN WDTCON<4>	$\overline{\text{RETEN}}$ CONFIG1L<0>
PIC18**F**XXK80	正常工作 （休眠）	0	x	1
PIC18**F**XXK80	低功耗模式 （休眠）	1	x	1
PIC18**F**XXK80	正常工作 （休眠）	0	0	0
PIC18**F**XXK80	低功耗模式 （休眠）	1	0	0
PIC18**F**XXK80	超低功耗模式 （休眠）	x	1	0
PIC18**LF**XXK80	保留 (2)	x	无关位	0
PIC18**LF**XXK80	稳压器旁路模式 （休眠） (2)	x	x	1

注 1： x—— 指示该 VIT 状态无效。

2： 要达到尽可能最低的休眠电流，需要在 PIC18**LF**XXK80 器件上禁止超低功耗稳压器 （$\overline{\text{RETEN}}$ = 1，禁止 ULP）。

28.4 双速启动

双速启动功能允许单片机在主时钟源可用之前使用 INTOSC（LF-INTOSC、MF-INTOSC 和 HF-INTOSC）振荡器作为时钟源，从而帮助器件最大限度地缩短从振荡器起振到代码执行之间的延时。通过将 IESO 配置位置 1 可使能该功能。

仅当主振荡器模式为 LP、XT 或 HS（基于晶振的模式）时才可使能双速启动。其他时钟源不需要 OST 起振延时；对于这些时钟源，应禁止双速启动。

使能双速启动时，当器件复位或从休眠模式唤醒时，器件将被配置为：在使能上电复位后、接着上电延时定时器发生超时后，使用内部振荡器模块作为时钟源。这使得在主振荡器起振、OST 运行的同时，代码几乎立即开始执行。一旦 OST 超时，器件就自动切换到 PRI_RUN 模式。

为了在唤醒器件时使用速度更快的时钟，通过在复位发生后立即设置 IRCF<2:0>，可以选择 INTOSC 或后分频器时钟源以提供更快的时钟速度。对于从休眠模式唤醒的情况，可以在进入休眠模式前设置 IRCF<2:0> 来选择 INTOSC 或后分频器时钟源。

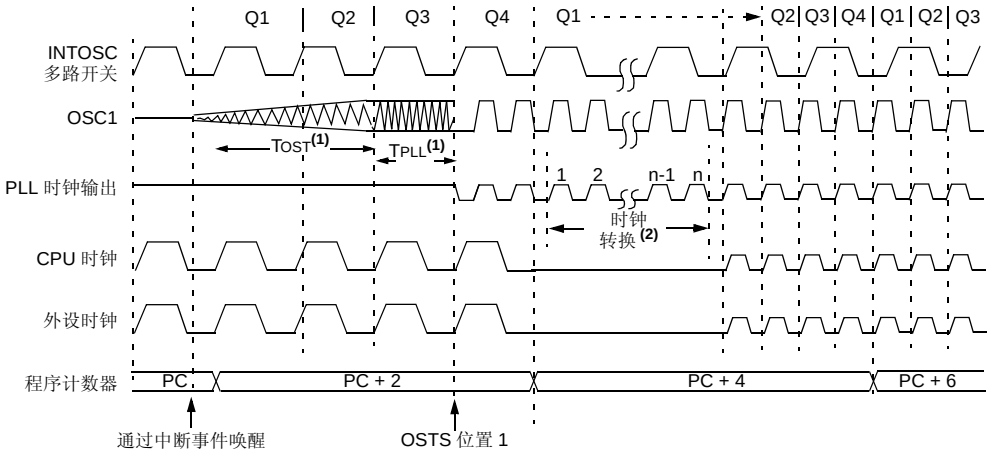
在所有其他功耗管理模式下，不使用双速启动。器件将使用当前选定的时钟源直到主时钟源可用为止。 IESO 位的设置被忽略。

28.4.1 使用双速启动时的特殊注意事项

当在双速启动模式下使用 INTOSC 振荡器时，器件仍将遵守进入功耗管理模式的正常指令顺序，包括执行多条 SLEEP 指令（见**第 4.1.4 节 “ 多条 SLEEP 命令 ”**）。实际上，这意味着在 OST 超时前用户代码可以改变 SCS<1:0> 位的设置或发出 SLEEP 指令。这就使应用程序能短暂地唤醒器件，执行 “ 日常任务 ”，并在器件开始使用主时钟源前返回休眠状态。

用户代码还能通过检查 OSTS 位（OSCCON<3>）的状态来确定主时钟源是否正在为器件提供时钟。如果该位置 1，则表示主振荡器正在为器件提供时钟。否则，表示当器件复位或从休眠模式唤醒期间由内部振荡器模块为器件提供时钟。

图 28-3： 双速启动时钟转换的时序图 （从 INTOSC 切换到 HSPLL）



注 1： TOST = 1024 TOSC； TPLL = 2 ms （近似值）。这些时间间隔未按比例绘制。

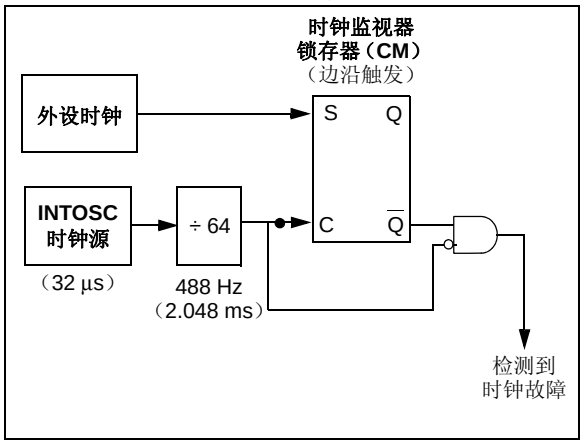
2： 时钟转换通常发生在 2-4 个 TOSC 内。

28.5 故障保护时钟监视器

故障保护时钟监视器（FSCM）可使单片机在外部振荡器发生故障时，自动将器件时钟切换到内部振荡器模块以保证器件能继续工作。将 FCMEN 配置位置 1 可使能 FSCM 功能。

当使能 FSCM 时，LF-INTOSC 振荡器将一直保持运行以监视外设时钟，并且在外设时钟发生故障时作为备用时钟。时钟监视（如图 28-4 所示）通过创建一个采样时钟信号实现，该信号为 LF-INTOSC 的 64 分频输出。这样就使得 FSCM 采样时钟脉冲之间有充足的时间间隔，从而保证在此期间至少有一个外设时钟沿出现。外设时钟和采样时钟作为时钟监视（CM）锁存器的输入。CM 在器件时钟源的下降沿被置 1，在采样时钟的上升沿被清零。

图 28-4： FSCM 框图



在采样时钟的下降沿检测时钟故障。如果在 CM 仍置 1 时出现采样时钟的下降沿，就表示检测到时钟故障（图 28-5）。这将引发以下事件：

- 通过将 OSCFIF 位（PIR2<7>）置 1，由 FSCM 产生振荡器故障中断
- 器件时钟源切换到内部振荡器模块（OSCCON 不会被更新，因此无法显示当前时钟源——这就是故障保护条件）
- WDT 复位

切换过程中，对于时序敏感的应用，内部振荡器模块的后分频器频率可能不够稳定。在这些情况下，最好选择另一种时钟配置并进入其他功耗管理模式。可以尝试部分恢复或执行受控关闭。更多详细信息，请参见**第 4.1.4 节“多条 SLEEP 命令”**和**第 28.4.1 节“使用双速启动时的特殊注意事项”**。

为了在唤醒器件时使用速度更快的时钟，通过在复位发生后立即设置 IRCF<2:0>，可以选择 INTOSC 或后分频器时钟源以提供更快的时钟速度。对于从休眠模式唤醒的情况，可以在进入休眠模式前设置 IRCF<2:0> 来选择 INTOSC 或后分频器时钟源。

FSCM 只能检测出主时钟源或辅助时钟源的故障。如果内部振荡器模块发生故障，将不会检测到故障，当然也不可能采取任何措施。

28.5.1 FSCM 和看门狗定时器

FSCM 和 WDT 均以 INTOSC 振荡器作为时钟源。由于 WDT 使用独立的分频器和计数器，使能 FSCM 时，禁止 WDT 对 INTOSC 振荡器的操作没有任何影响。

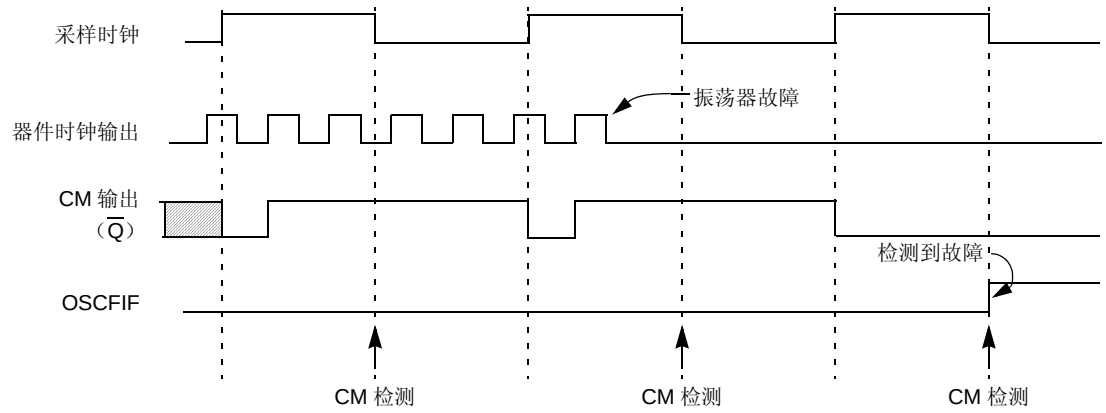
如前所述，当检测到时钟故障时，时钟源将切换到 INTOSC 时钟。根据由 IRCF<2:0> 位选择的频率的不同，这可能意味着代码执行速度会发生很大的变化。如果使能 WDT 时使用的是小预分频值，时钟速度的下降将引起 WDT 超时，随后使器件复位。由于这个原因，故障保护时钟事件也会使 WDT 和后分频器复位，使 WDT 从执行速度发生变化那一刻起开始重新计数，从而降低发生错误超时的可能。

28.5.2 退出故障保护工作模式

器件复位或进入功耗管理模式均可终止故障保护条件。发生复位时，控制器启动在配置寄存器 1H 中指定的主时钟源（具有如 OST 或 PLL 定时器等振荡器模式所需的起振延时）。INTOSC 多路开关将在主时钟源就绪之前提供系统时钟（类似于双速启动）。随后时钟源切换为主时钟（由 OSCCON 寄存器中的 OSTS 位置 1 指示）。然后，故障保护时钟监视器恢复监视外设时钟。

在起振期间，主时钟源可能永远不能就绪。在这种情况下，器件工作将以 INTOSC 多路开关作为时钟源。OSCCON 寄存器将保持复位状态直到进入功耗管理模式为止。

图 28-5：　　FSCM 时序图



注：　通常，器件时钟的频率比采样时钟频率高很多。本例选用的相对频率仅是为了便于说明。

28.5.3　功耗管理模式下的 FSCM 中断

进入功耗管理模式时，时钟多路开关选择由OSCCON寄存器选定的时钟源。在功耗管理模式下会恢复对功耗管理时钟源的故障保护时钟的监视。

如果在功耗管理工作模式下发生了振荡器故障，后续事件取决于是否允许了振荡器故障中断。如果允许了该中断（OSCFIF = 1），代码执行将以 INTOSC 多路开关作为时钟源，不会自动转换回发生故障的时钟源。

如果禁止了该中断，空闲模式下产生的后续中断将使 CPU 以 INTOSC 时钟源作为时钟源开始执行指令。

28.5.4　POR 或从休眠中唤醒

FSCM 设计为在器件退出上电复位（POR）或低功耗休眠模式后检测振荡器故障。当器件主时钟为 EC、RC 或 INTOSC 模式时，会在这些事件后立即开始监视。

当振荡器模式使用了晶振或谐振器时（HS、HSPLL、LP 或 XT），情况会有所不同。由于这类振荡器需要的起振时间可能比 FCSM 采样时钟时间长很多，因此可能会检测到假的时钟故障。为避免这种情况，内部振荡器模块会被自动配置为器件时钟并一直工作到主时钟稳定为止（OST 和 PLL 定时器发生超时时）。

这与双速启动模式相同。一旦主时钟稳定下来，INTOSC 就将重新作为 FSCM 时钟源。

注：　用于防止在POR或从休眠状态唤醒时发生错误中断的逻辑，同样也将阻止随后对振荡器故障的检测。通过监视 OSTS 位，并使用定时程序来确定振荡器起振时间是否过长，可避免这个问题。即使如此，在检测到振荡器故障时也不会将振荡器故障中断标志位置 1。

正如**第 28.4.1 节 “ 使用双速启动时的特殊注意事项 ”** 中所述，在等待主时钟稳定的过程中，可以选择另一种时钟配置和进入另一种功耗管理模式。当选择了新的功耗管理模式时，主时钟将被禁止。

28.6 程序校验和代码保护

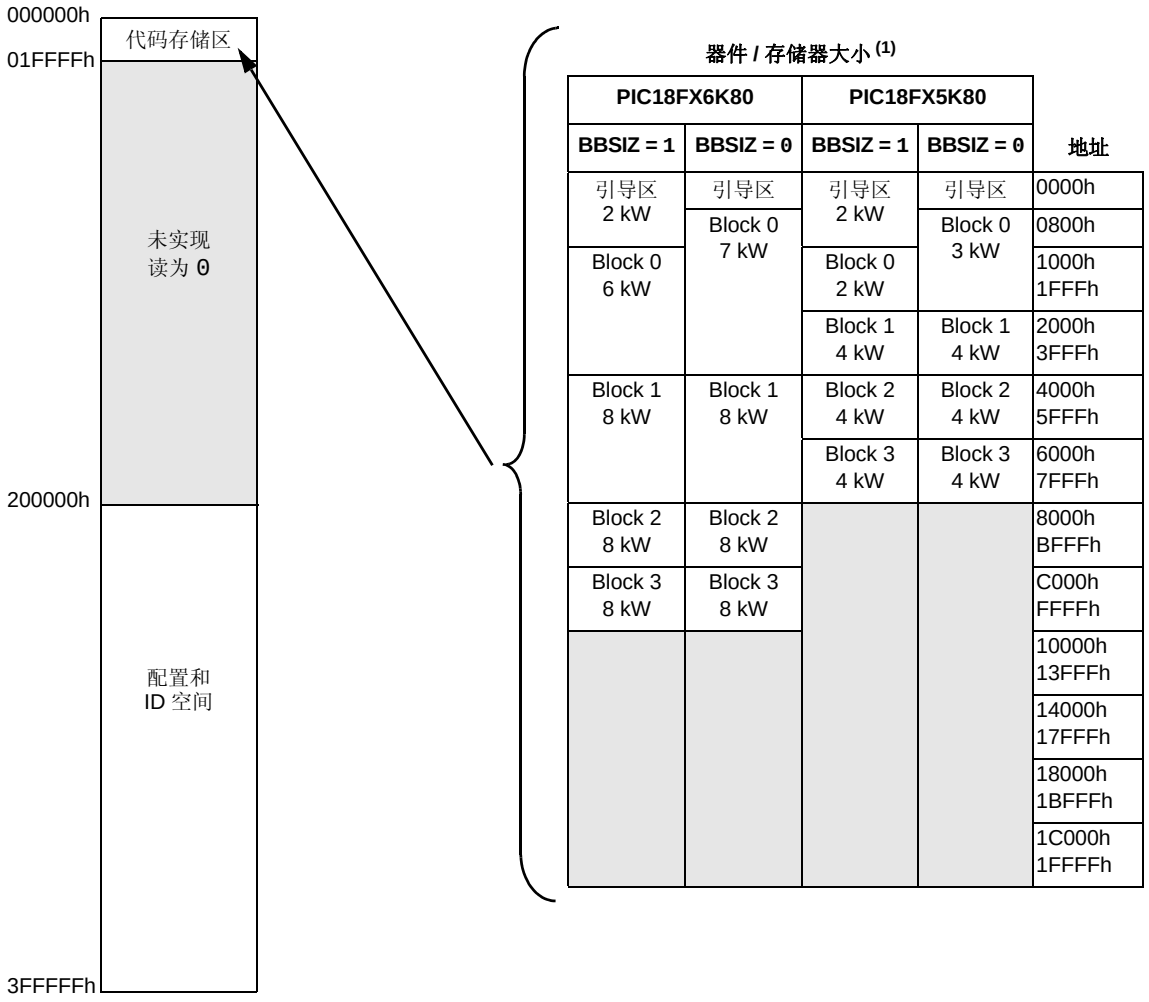
用户程序存储器被划分为 4 个存储块。其中一个是 1 KB 或 2 KB 的引导区。存储器的剩余部分按二进制边界划分为存储块。

每个存储块都有与其相关的 3 个代码保护位。它们是：

- 代码保护位 （CPx）
- 写保护位 （WRTx）
- 外部存储块表读位 （EBTRx）

图 28-6 给出了 48 KB、64 KB、96 KB 和 128 KB 器件的程序存储器构成以及与每个存储块相关的特定代码保护位。表 28-4 中总结了这些位的实际地址。

图 28-6： PIC18F66K80 系列的受代码保护的程序存储器



器件 / 存储器大小 (1)

PIC18FX6K80		PIC18FX5K80		
BBSIZ = 1	BBSIZ = 0	BBSIZ = 1	BBSIZ = 0	地址
引导区 2 kW	引导区	引导区 2 kW	引导区	0000h
	Block 0 7 kW		Block 0 3 kW	0800h
Block 0 6 kW		Block 0 2 kW		1000h 1FFFh
		Block 1 4 kW	Block 1 4 kW	2000h 3FFFh
Block 1 8 kW	Block 1 8 kW	Block 2 4 kW	Block 2 4 kW	4000h 5FFFh
		Block 3 4 kW	Block 3 4 kW	6000h 7FFFh
Block 2 8 kW	Block 2 8 kW			8000h BFFFh
Block 3 8 kW	Block 3 8 kW			C000h FFFFh
				10000h 13FFFh
				14000h 17FFFh
				18000h 1BFFFh
				1C000h 1FFFFh

注 **1：** 存储区的大小未按比例显示。

2： 引导区大小由 BBSIZ0 位 （CONFIG4L<4>）决定。

表 28-4：　　代码保护寄存器汇总

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h	CONFIG5L	—	—	—	—	CP3	CP2	CP1	CP0
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—
30000Ah	CONFIG6L	—	—	—	—	WRT3	WRT2	WRT1	WRT0
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—
30000Ch	CONFIG7L	—	—	—	—	EBTR3	EBTR2	EBTR1	EBTR0
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—

图注：　阴影单元未实现。

28.6.1　程序存储器代码保护

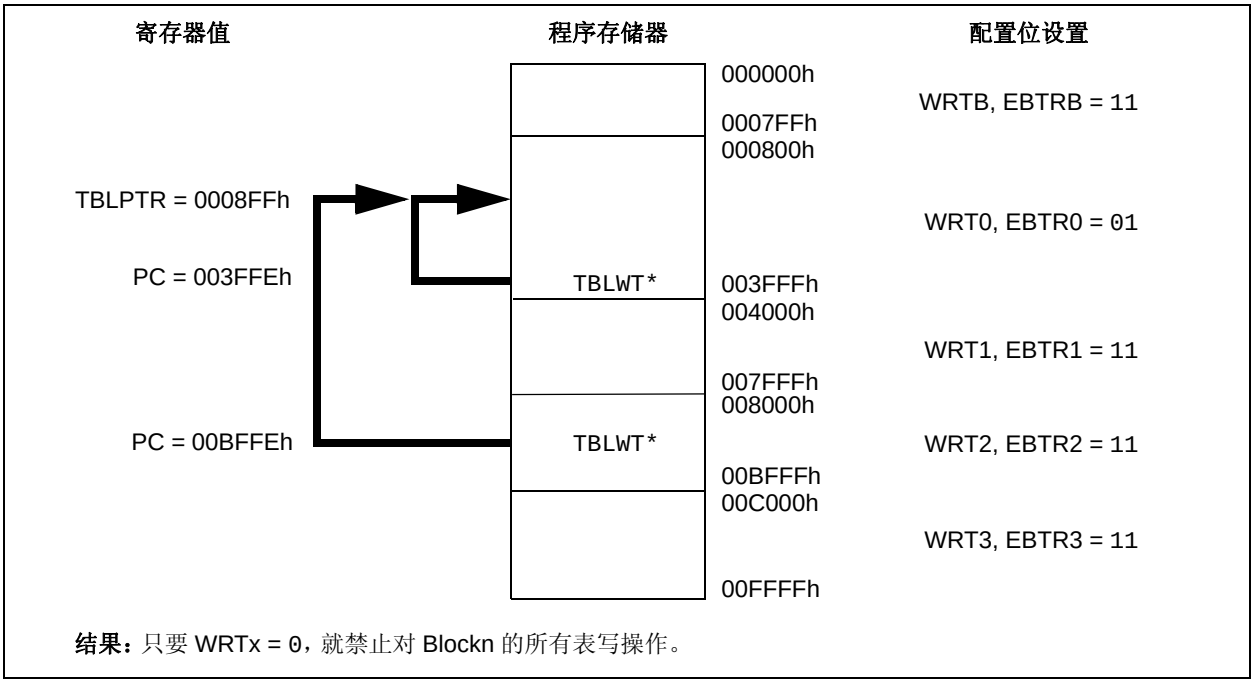
可使用表读和表写指令从任何存储单元读写程序存储器。器件 ID 可以通过表读指令进行读取。配置寄存器可以通过表读和表写指令进行读写操作。

在正常执行模式下，CPx 位不起直接作用。CPx 位禁止外部读写。如果 WRTx 配置位为 0，则用户存储区的存储块可被保护不受表写指令的影响。

EBTRx 位控制表读操作。对于 EBTRx 位设置为 0 的用户存储区中的存储块，允许在该存储块内执行表读指令。该存储块以外的存储单元执行的表读指令则不被允许，将导致读为 0。图 28-7 至图 28-9 给出了表写和表读保护的图示。

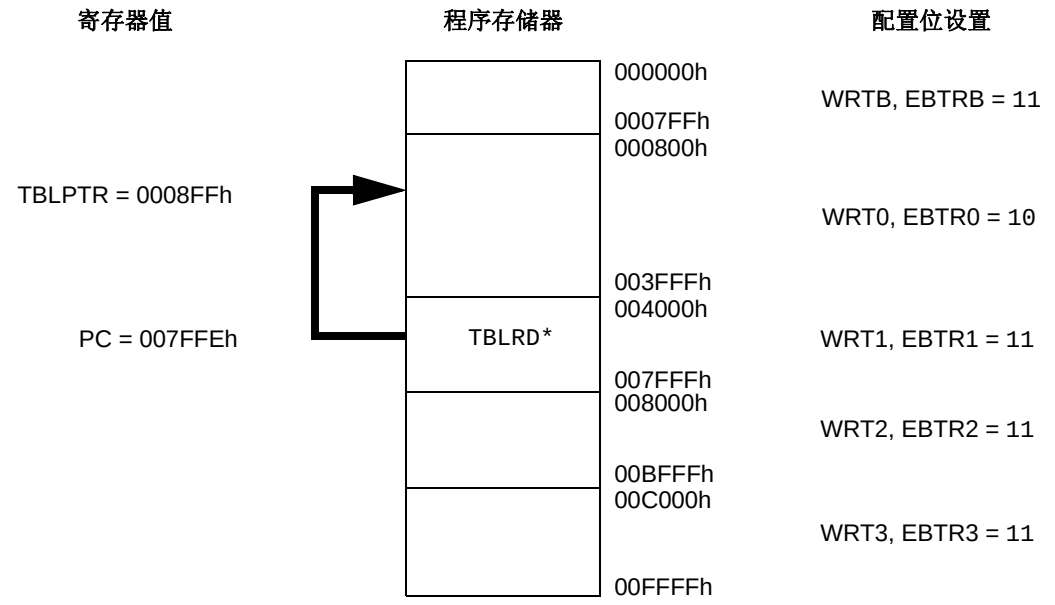
> **注：**　代码保护位只能从 1 改写到 0 状态。不可能将处于 0 状态的位改写为 1。只有通过整片擦除或块擦除功能才能将代码保护位设置为 1。整片擦除和块擦除功能只能通过 ICSP 或外部编程器启动。更多信息，请参见器件编程规范。

图 28-7：　　不允许表写 （WRTx）



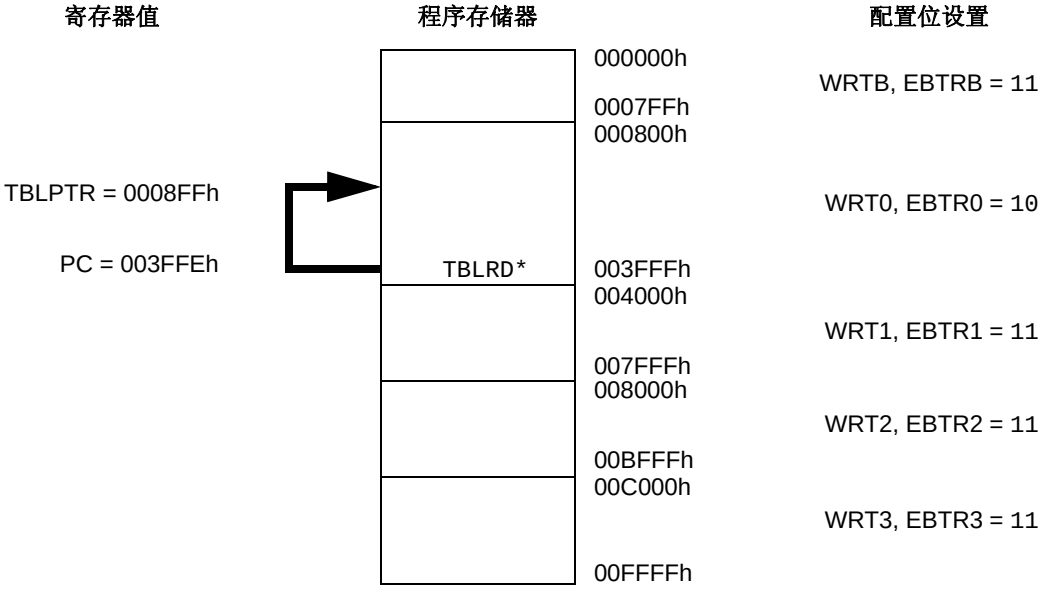
结果：只要 WRTx = 0，就禁止对 Blockn 的所有表写操作。

图 28-8：　　不允许外部存储块的表读 （EBTRx）



结果：只要 EBTRx = 0，就禁止来自外部存储块对 Blockn 的所有表读操作。
TABLAT 寄存器返回 0。

图 28-9：　　允许外部存储块的表读 （EBTRx）



结果：即使当 EBTRBx = 0 时，也允许在 Blockn 内执行表读操作。
TABLAT 寄存器返回 TBLPTR 指向的存储单元的值。

28.6.2 数据 EEPROM 代码保护

CPD 和 WRTD 两个位可保护整个数据 EEPROM 不被从外部读写。CPD 禁止数据 EEPROM 的外部读写。WRTD 禁止从内部和外部写数据 EEPROM。在正常操作下，CPU 可以始终读数据 EEPROM，与保护位的设置无关。

28.6.3 配置寄存器保护

配置寄存器可以是写保护的。WRTC 位控制配置寄存器的保护。在正常执行模式下，WRTC 位是只读位。WRTC 只能通过 ICSP 或外部编程器写入。

28.7 ID 存储单元

有 8 个存储单元（200000h-200007h）被指定为 ID 单元，供用户存储校验和或其他代码标识编号。在执行程序时可通过 TBLRD 和 TBLWT 指令读写这些单元；在编程 / 校验时，也可读写这些单元。当器件有代码保护时，也可读取 ID 单元。

28.8 在线串行编程

可以在最终的应用电路中对 PIC18F66K80 系列器件进行串行编程。只需要 5 根线即可实现这一操作，其中时钟线、数据线各一根，其余 3 根分别是电源线、接地线和编程电压线。这允许用户在生产电路板时使用未编程器件，仅在产品交付之前才对单片机进行编程，从而可以使用最新版本的固件或者定制固件进行编程。

关于各种编程模式的信息，请参见编程规范。

28.9 在线调试器

将 $\overline{\text{DEBUG}}$ 配置位编程为 0，可使能在线调试功能。该功能允许与 MPLAB® IDE 配合使用进行简单的调试。当使能了单片机的这项功能时，有些资源就不再是通用的了。表 28-5 给出了后台调试器所需的资源。

表 28-5： 调试器资源

I/O 引脚：	RB6 和 RB7
堆栈：	2 级
程序存储器：	512 字节
数据存储器：	10 字节

要使用单片机的在线调试功能，设计必须实现 $\overline{\text{MCLR}}$/RE3、VDD、VSS、RB7 和 RB6 的在线串行编程连接，从而为 Microchip 或第三方开发工具公司提供在线调试器模块接口。

29.0 指令集汇总

PIC18F66K80 系列器件具有一个包含 75 条 PIC18 核心指令的标准指令集，和一个包含 8 条新指令的扩展指令集，扩展指令集用于优化递归代码或使用软件堆栈的代码。本章后面的部分将讨论扩展指令集。

29.1 标准指令集

标准的 PIC18 MCU 指令集与以前的 PIC® MCU 指令集相比，添加了很多增强功能，并保持了易于从这些 PIC MCU 指令集移植的特点。大部分指令为单程序存储字指令（16 位），只有 4 条指令需要两个程序存储单元。

每个单字指令都是一个 16 位字，由操作码 （指定指令类型）和一个或多个操作数 （指定指令操作）组成。

整个指令集具有高度的正交性，可以分为以下 4 种基本类型：

- **字节**操作类指令
- **位**操作类指令
- **立即数**操作类指令
- **控制**操作类指令

表 29-2 为 PIC18 指令集汇总，列出了上述四类指令。表 29-1 给出了操作码字段的说明。

大部分**字节**操作类的指令都含有三种操作数：

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. 被访问存储区 （由 “a” 指定）

文件寄存器标识符“f”指定了指令将会使用哪一个文件寄存器。目标寄存器标识符 “d” 指定了操作结果的存放位置。如果 “d” 为 0，操作结果存入 WREG 寄存器中。如果 “d” 为 1，操作结果存入指令指定的文件寄存器中。

所有**位**操作类指令都含有三种操作数：

1. 文件寄存器 （由 “f” 指定）
2. 文件寄存器中的位 （由 “b” 指定）
3. 被访问存储区 （由 “a” 指定）

位域标识符 “b” 选择操作所影响的位的编号，而文件寄存器标识符 “f” 则代表这些位所在的文件寄存器地址。

立即数操作类指令使用以下操作数：

- 要装入到文件寄存器中的立即数 （由 “k” 指定）
- 要装入立即数的 FSR 寄存器 （由 “f” 指定）
- 不需要操作数 （由 “—” 指定）

控制类指令可以使用以下操作数：

- 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 4 条双字指令外，其他所有的指令都是单字指令。双字指令将所需的信息保存在 32 位中。第二个字的高 4 位都是 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。

除非条件测试结果为真或者指令执行改变了程序计数器的值，否则执行所有的单字指令都只需要一个指令周期。对于上述两种特殊情况，指令执行需要两个指令周期，在第二个指令周期中执行一条 NOP 指令。

执行双字指令需要两个指令周期。

每个指令周期由 4 个振荡周期组成。因此，如果振荡器频率为 4 MHz，正常的指令执行时间为 1 µs。如果条件测试结果为真或者指令执行改变了程序计数器的值，则该指令的执行时间为 2 µs。双字跳转指令 （如果为真）的执行则需要 3 µs。

图 29-1 给出了指令的几种通用格式。所有示例均使用 “nnh” 来表示十六进制数。

指令集汇总（见表29-2）列出了可被Microchip MPASM™ 汇编器识别的标准指令。

第 29.1.1 节 “ 标准指令集 ” 中对每条指令进行了介绍。

表 29-1： 操作码字段说明

字段	说明
a	快速操作 RAM 位： a = 0：快速操作 RAM 内的 RAM 存储单元 （BSR 寄存器被忽略） a = 1：由 BSR 寄存器指定 RAM 存储区
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。用于选择当前的 RAM 存储区。
C、DC、Z、OV 和 N	ALU 状态位：进位、半进位、全零、溢出和负标志位。
d	目标寄存器选择位： d = 0：结果保存至 WREG 寄存器 d = 1：结果保存至文件寄存器 f
dest	目标寄存器：可以是 WREG 寄存器或指定的文件寄存器地址。
f	8 位文件寄存器地址 （00h 至 FFh），或 2 位 FSR 标识符 （0h 至 3h）。
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。
GIE	全局中断允许位。
k	立即数、常数或者标号 （可能是 8 位、 12 位或 20 位的值）。
label	标号名称。
mm * *+ *- +*	表读和表写指令的 TBLPTR 寄存器模式。 只与表读和表写指令一起使用： 不改变寄存器 （如用于表读和表写的 TBLPTR） 后递增寄存器 （如用于表读和表写的 TBLPTR） 后递减寄存器 （如用于表读和表写的 TBLPTR） 预递增寄存器 （如用于表读和表写的 TBLPTR）
n	相对跳转指令的相对地址 （二进制补码形式），或 Call/ 跳转和 Return 指令的直接地址。
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器。
PCLATU	程序计数器最高字节锁存器。
$\overline{PD}$	掉电位。
PRODH	乘积的高字节。
PRODL	乘积的低字节。
s	快速调用 / 返回模式选择位： s = 0：不对影子寄存器进行更新，也不用影子寄存器的内容更新其他寄存器 s = 1：将寄存器的值装入影子寄存器或将影子寄存器中的值装入寄存器 （快速模式）
TBLPTR	21 位表指针 （指向程序存储器地址）。
TABLAT	8 位表锁存器。
$\overline{TO}$	超时位。
TOS	栈顶。
u	未使用或未改变。
WDT	看门狗定时器。
WREG	工作寄存器 （累加器）。
x	无关位 （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，建议使用这种形式。
z_s	对寄存器 （源）进行间接寻址的 7 位偏移量。
z_d	对寄存器 （目标）进行间接寻址的 7 位偏移量。
{ }	可选参数。
[text]	表示变址地址。
(text)	text 的内容。
[expr]<n>	表示由指针 expr 指向的寄存器中的 bit n。
→	赋值。
< >	寄存器位域。
∈	表示属于某个集合。
斜体文字	用户定义项 （字体为 Courier New）。

图 29-1：　　指令的通用格式

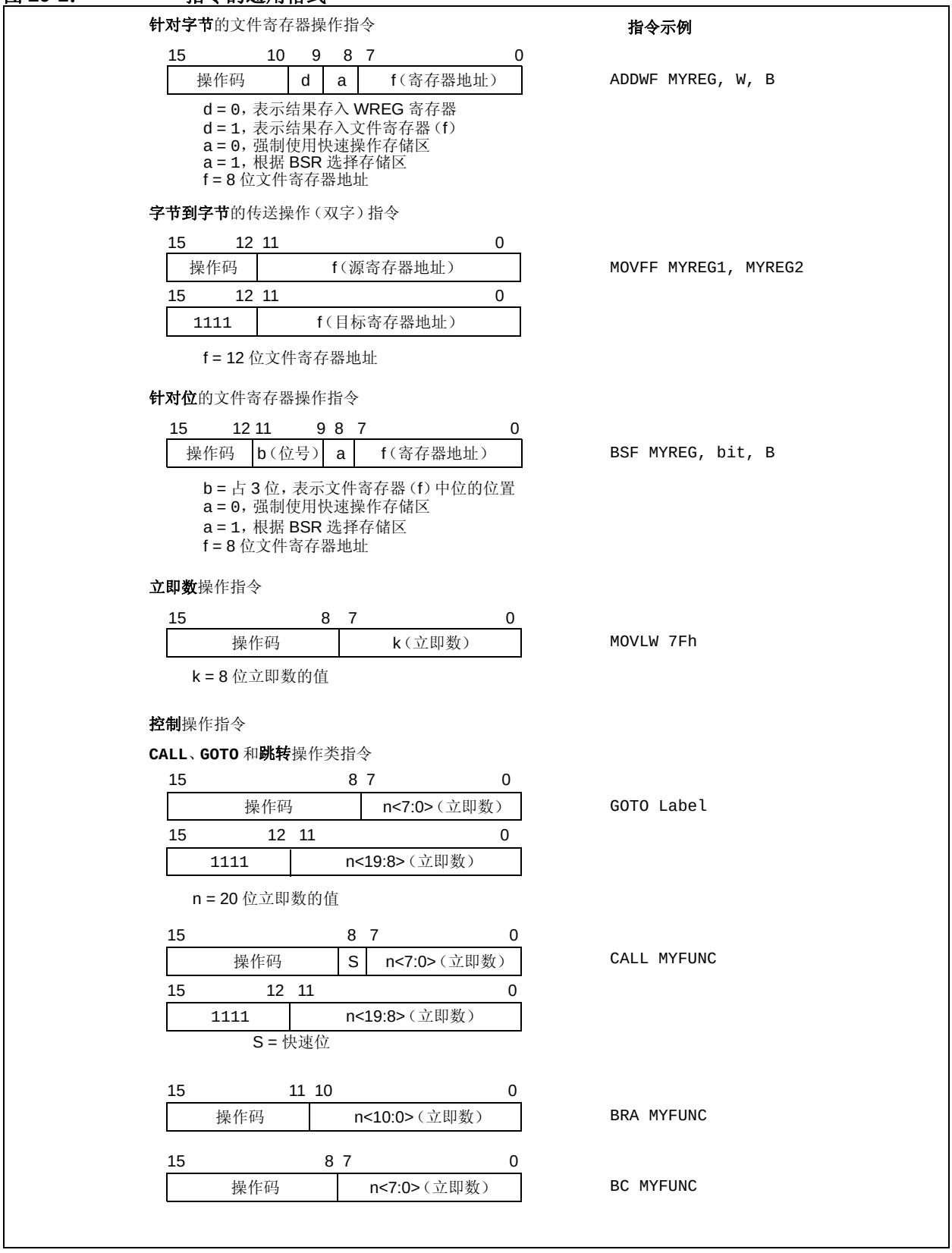


表 29-2： PIC18F66K80 系列指令集

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
针对字节的操作类指令							
ADDWF	f, d, a	WREG 与 f 相加	1	0010 01da	ffff ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	WREG 与 f 带进位相加	1	0010 00da	ffff ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	WREG 和 f 作逻辑与运算	1	0001 01da	ffff ffff	Z, N	1,2
CLRF	f, a	将 f 清零	1	0110 101a	ffff ffff	Z	2
COMF	f, d, a	对 f 取反	1	0001 11da	ffff ffff	Z, N	1, 2
CPFSEQ	f, a	将f与WREG作比较，相等则跳过	1（2或3）	0110 001a	ffff ffff	无	4
CPFSGT	f, a	将f与WREG作比较，大于则跳过	1（2或3）	0110 010a	ffff ffff	无	4
CPFSLT	f, a	将f与WREG作比较，小于则跳过	1（2或3）	0110 000a	ffff ffff	无	1, 2
DECF	f, d, a	f 递减 1	1	0000 01da	ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	f 递减 1，为 0 则跳过	1（2或3）	0010 11da	ffff ffff	无	1, 2, 3, 4
DCFSNZ	f, d, a	f 递减 1，非 0 则跳过	1（2或3）	0100 11da	ffff ffff	无	1, 2
INCF	f, d, a	f 递增 1	1	0010 10da	ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	f 递增 1，为 0 则跳过	1（2或3）	0011 11da	ffff ffff	无	4
INFSNZ	f, d, a	f 递增 1，非 0 则跳过	1（2或3）	0100 10da	ffff ffff	无	1, 2
IORWF	f, d, a	WREG 和 f 作逻辑或运算	1	0001 00da	ffff ffff	Z, N	1, 2
MOVF	f, d, a	传送 f	1	0101 00da	ffff ffff	Z, N	1
MOVFF	f_s, f_d	从 f_s（源）送到 f_d（目标）	2	1100 ffff 1111 ffff	ffff ffff ffff ffff	无 第一个字 第二个字	
MOVWF	f, a	将 WREG 内容传送到 f	1	0110 111a	ffff ffff	无	
MULWF	f, a	WREG 与 f 相乘	1	0000 001a	ffff ffff	无	1, 2
NEGF	f, a	对 f 取补	1	0110 110a	ffff ffff	C, DC, Z, OV, N	
RLCF	f, d, a	f 带进位循环左移	1	0011 01da	ffff ffff	C, Z, N	1, 2
RLNCF	f, d, a	f 循环左移 （不带进位）	1	0100 01da	ffff ffff	Z, N	
RRCF	f, d, a	f 带进位循环右移	1	0011 00da	ffff ffff	C, Z, N	
RRNCF	f, d, a	f 循环右移 （不带进位）	1	0100 00da	ffff ffff	Z, N	
SETF	f, a	将 f 的内容置为全 1	1	0110 100a	ffff ffff	无	1, 2
SUBFWB	f, d, a	WREG 减去 f （带借位）	1	0101 01da	ffff ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	f 减去 WREG	1	0101 11da	ffff ffff	C, DC, Z, OV, N	1, 2
SUBWFB	f, d, a	f 减去 WREG （带借位）	1	0101 10da	ffff ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	将 f 中的两个半字节进行交换	1	0011 10da	ffff ffff	无	4
TSTFSZ	f, a	测试 f，为 0 则跳过	1（2或3）	0110 011a	ffff ffff	无	1, 2
XORWF	f, d, a	WREG 和 f 作逻辑异或运算	1	0001 10da	ffff ffff	Z, N	

注 1： 当端口寄存器修改自身时 （例如， MOVF PORTB, 1, 0），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 1，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 0。

2： 当对 TMR0 寄存器执行该指令（并且 d = 1）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器（PC）被修改或者条件测试为真，则该指令需要两个周期。第二个周期执行一条 NOP 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

表 29-2： PIC18F66K80 系列指令集 （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
针对位的操作类指令						
BCF	f, b, a	将 f 中的某位清零	1	1001 bbba ffff ffff	无	1, 2
BSF	f, b, a	将 f 中的某位置 1	1	1000 bbba ffff ffff	无	1, 2
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	3, 4
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	3, 4
BTG	f, b, a	将 f 中的某位取反	1	0111 bbba ffff ffff	无	1, 2
控制类指令						
BC	n	进位则跳转	1 （2）	1110 0010 nnnn nnnn	无	
BN	n	为负则跳转	1 （2）	1110 0110 nnnn nnnn	无	
BNC	n	无进位则跳转	1 （2）	1110 0011 nnnn nnnn	无	
BNN	n	不为负则跳转	1 （2）	1110 0111 nnnn nnnn	无	
BNOV	n	不溢出则跳转	1 （2）	1110 0101 nnnn nnnn	无	
BNZ	n	不为零则跳转	1 （2）	1110 0001 nnnn nnnn	无	
BOV	n	溢出则跳转	1 （2）	1110 0100 nnnn nnnn	无	
BRA	n	无条件跳转	2	1101 0nnn nnnn nnnn	无	
BZ	n	为零则跳转	1 （2）	1110 0000 nnnn nnnn	无	
CALL	n, s	调用子程序 第一个字	2	1110 110s kkkk kkkk	无	
		第二个字		1111 kkkk kkkk kkkk		
CLRWDT	—	将看门狗定时器清零	1	0000 0000 0000 0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	对 WREG 进行十进制调整	1	0000 0000 0000 0111	C	
GOTO	n	跳转到地址 第一个字	2	1110 1111 kkkk kkkk	无	
		第二个字		1111 kkkk kkkk kkkk		
NOP	—	空操作	1	0000 0000 0000 0000	无	
NOP	—	空操作	1	1111 xxxx xxxx xxxx	无	4
POP	—	弹出返回堆栈栈顶（TOS）的内容	1	0000 0000 0000 0110	无	
PUSH	—	将数据压入返回堆栈栈顶（TOS）	1	0000 0000 0000 0101	无	
RCALL	n	相对调用	2	1101 1nnn nnnn nnnn	无	
RESET		软件器件复位	1	0000 0000 1111 1111	全部	
RETFIE	s	中断返回允许	2	0000 0000 0001 000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	返回时将立即数送入 WREG	2	0000 1100 kkkk kkkk	无	
RETURN	s	从子程序返回	2	0000 0000 0001 001s	无	
SLEEP	—	进入待机模式	1	0000 0000 0000 0011	$\overline{TO}$, $\overline{PD}$	

注 1： 当端口寄存器修改自身时 （例如， MOVF PORTB, 1, 0），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 1，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 0。

2： 当对 TMR0 寄存器执行该指令（并且 d = 1）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器（PC）被修改或者条件测试为真，则该指令需要两个周期。第二个周期执行一条 NOP 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

表 29-2： PIC18F66K80 系列指令集 （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
立即数操作类指令					
ADDLW k	WREG 与立即数相加	1	0000 1111 kkkk kkkk	C, DC, Z, OV, N	
ANDLW k	WREG 和立即数进行逻辑与运算	1	0000 1011 kkkk kkkk	Z, N	
IORLW k	WREG 和立即数进行逻辑或运算	1	0000 1001 kkkk kkkk	Z, N	
LFSR f, k	传送立即数 （12 位）第二个字	2	1110 1110 00ff kkkk	无	
	到 FSR （f） 第一个字		1111 0000 kkkk kkkk		
MOVLB k	将立即数送入 BSR<3:0>	1	0000 0001 0000 kkkk	无	
MOVLW k	将立即数送入 WREG	1	0000 1110 kkkk kkkk	无	
MULLW k	WREG 和立即数相乘	1	0000 1101 kkkk kkkk	无	
RETLW k	返回时将立即数送入 WREG	2	0000 1100 kkkk kkkk	无	
SUBLW k	立即数减去 WREG	1	0000 1000 kkkk kkkk	C, DC, Z, OV, N	
XORLW k	WREG 和立即数进行逻辑异或运算	1	0000 1010 kkkk kkkk	Z, N	
数据存储器 ↔ 程序存储器操作					
TBLRD*	表读	2	0000 0000 0000 1000	无	
TBLRD*+	后递增表读		0000 0000 0000 1001	无	
TBLRD*-	后递减表读		0000 0000 0000 1010	无	
TBLRD+*	预递增表读		0000 0000 0000 1011	无	
TBLWT*	表写	2	0000 0000 0000 1100	无	
TBLWT*+	后递增表写		0000 0000 0000 1101	无	
TBLWT*-	后递减表写		0000 0000 0000 1110	无	
TBLWT+*	预递增表写		0000 0000 0000 1111	无	

注 1： 当端口寄存器修改自身时 （例如， MOVF PORTB, 1, 0），修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 1，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 0。

2： 当对 TMR0 寄存器执行该指令（并且 d = 1）时，如果已为其分配了预分频器，则将该预分频器清零。

3： 如果程序计数器（PC）被修改或者条件测试为真，则该指令需要两个周期。第二个周期执行一条 NOP 指令。

4： 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

29.1.1 标准指令集

ADDLW **W 与立即数相加**

语法： ADDLW k

操作数： 0 ≤ k ≤ 255

操作： (W) + k → W

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0000	1111	kkkk	kkkk

说明： 将 W 寄存器的内容与 8 位立即数 k 相加，结果存储在 W 寄存器中。

指令字数： 1

指令周期数： 1

Q 周期操作：

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 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) + (f) → dest

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0010	01da	ffff	ffff

说明： 将 W 的内容与 f 寄存器的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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： 所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数，用于符号寻址。如果使用了标号，那么指令格式将变为：{label} 指令参数。

ADDWFC **W 与 f 带进位相加**

语法： ADDWFC f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) + (f) + (C) → dest

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

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，结果存储在 W 中。如果 d 为 1，结果存储在数据存储单元 f 中。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： ADDWFC REG, 0, 1

执行指令前
进位标志位 = 1
REG = 02h
W = 4Dh

执行指令后
进位标志位 = 0
REG = 02h
W = 50h

ANDLW **立即数和 W 寄存器作逻辑与运算**

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

机器码：

0000	1011	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 k 进行逻辑与运算。结果存储在 W 寄存器中。

指令字数： 1

指令周期数： 1

Q 周期操作：

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 **将 W 和 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) → dest

受影响的状态位： N 和 Z

机器码：

0001	01da	ffff	ffff

说明： 将 W 的内容与寄存器 f 的内容进行逻辑与运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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 **进位则跳转**

语法： BC n

操作数： -128 ≤ n ≤ 127

操作： 如果进位标志位为 1，
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0010	nnnn	nnnn

说明： 如果进位标志位为 1，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。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： 1（2）

Q 周期操作：
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
如果进位标志位 = 1；
PC = 地址（HERE + 12）
如果进位标志位 = 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

机器码：

1001	bbba	ffff	ffff

说明： 将寄存器 f 中的位 b 清零。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 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 **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

机器码：

1110	0110	nnnn	nnnn

说明： 如果负标志位为 1，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。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： 1（2）

Q 周期操作：
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 **无进位则跳转**

语法： BNC n

操作数： -128 ≤ n ≤ 127

操作： 如果进位标志位为 0，
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0011	nnnn	nnnn

说明： 如果进位标志位为 0，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。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： 1（2）

Q 周期操作：
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
如果进位标志位 = 0；
PC = 地址（Jump）
如果进位标志位 = 1；
PC = 地址（HERE + 2）

BNN **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

机器码：

1110	0111	nnnn	nnnn

说明： 如果负标志位为 0，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。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： 1（2）

Q 周期操作：
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 **不溢出则跳转**

语法： BNOV n

操作数： -128 ≤ n ≤ 127

操作： 如果溢出标志位为 0，
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0101	nnnn	nnnn

说明： 如果溢出标志位为 0，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。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： 1（2）

Q 周期操作：
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
如果溢出标志位 = 0；
PC = 地址（Jump）
如果溢出标志位 = 1；
PC = 地址（HERE + 2）

BNZ **不为零则跳转**

语法： BNZ n

操作数： -128 ≤ n ≤ 127

操作： 如果全零标志位为 0，
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0001	nnnn	nnnn

说明： 如果全零标志位为 0，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。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： 1（2）

Q 周期操作：
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
如果全零标志位 = 0；
PC = 地址（Jump）
如果全零标志位 = 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

机器码：

1101	0nnn	nnnn	nnnn

说明： “2n”（以二进制补码表示）与 PC 相加。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 周期操作：

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

机器码：

1000	bbba	ffff	ffff

说明： 将寄存器 f 的位 b 置 1。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 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 **测试文件寄存器中的某位，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

机器码：

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。即在 b 位为 0 时，丢弃执行当前指令过程中已取的下一条指令，转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 BTFSC 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 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，且后面跟有 2 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 **测试文件寄存器中的某位，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

机器码：

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。即在 b 位为 1 时，丢弃执行当前指令过程中已取的下一条指令，转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 BTFSS 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 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，且后面跟有 2 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 中的某位取反**

语法： BTG f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b < 7
a ∈ [0,1]

操作： $(\overline{f<b>}) \rightarrow f<b>$

受影响的状态位： 无

机器码：

0111	bbba	ffff	ffff

说明： 将数据存储单元 f 中的位 b 取反。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 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 **溢出则跳转**

语法： BOV n

操作数： -128 ≤ n ≤ 127

操作： 如果溢出标志位为 1，
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0100	nnnn	nnnn

说明： 如果溢出标志位为 1，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。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： 1（2）

Q 周期操作：
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
如果溢出标志位 = 1；
PC = 地址（Jump）
如果溢出标志位 = 0；
PC = 地址（HERE + 2）

BZ **为零则跳转**

语法： BZ n

操作数： -128 ≤ n ≤ 127

操作： 如果全零标志位为 1，
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1110	0000	nnnn	nnnn

说明： 如果全零标志位为 1，程序将跳转。

“2n”（以二进制补码表示）与 PC 相加。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： 1（2）

Q 周期操作：
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
如果全零标志位 = 1；
PC = 地址（Jump）
如果全零标志位 = 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

机器码：

第一个字（k<7:0>）	1110	110s	$k_7$kkk	kkkk$_0$
第二个字（k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明： 可在整个 2 MB 的存储器范围内进行子程序调用。首先，将返回地址（PC + 4）压入返回堆栈。如果 s = 1，还会将 W、STATUS 和 BSR 寄存器的内容存入它们各自的影子寄存器 WS、STATUSS 和 BSRS。如果 s = 0，将不会进行任何更新（默认）。然后，将 20 位的值 k 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 周期操作：

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

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

机器码：

0110	101a	ffff	ffff

说明： 清零指定寄存器的内容。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 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 PD

机器码：

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 PD 置 1。

指令字数： 1

指令周期数： 1

Q 周期操作：

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 **对 f 取反**

语法： COMF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： $\bar{f} \rightarrow$ dest

受影响的状态位： N 和 Z

机器码：

0001	11da	ffff	ffff

说明： 将寄存器 f 的内容取反。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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

机器码：

0110	001a	ffff	ffff

说明： 通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容作比较。

如果 f = W，则丢弃已取的指令转而执行一条执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 CPFSEQ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 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，且后面跟有 2 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

机器码：

0110	010a	ffff	ffff

说明： 通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容作比较。

如果 f > W，则丢弃已取的指令转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 CPFSLT 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 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，且后面跟有 2 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 ≤ 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

机器码：

0110	000a	ffff	ffff

说明： 通过执行无符号的减法，将数据存储单元 f 的内容与 W 的内容作比较。

如果 f < W，则丢弃已取的指令转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过，且后面跟有 2 字指令，则执行 CPFSLT 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 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，且后面跟有 2 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 **对 W 寄存器进行十进制调整**

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> > 9] 或 [C = 1]，则
(W<7:4>) + 6 → W<7:4> ；
C = 1
否则
(W<7:4>) → W<7:4>

受影响的状态位： C

机器码：

0000	0000	0000	0111

说明： DAW 指令调整 W 寄存器内的 8 位值，即之前两个压缩 BCD 格式的变量之和，并产生一个正确的压缩 BCD 格式结果。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 W	处理数据	写 W

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
```

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 → dest

受影响的状态位： C、 DC、 N、 OV 和 Z

机器码：

0000	01da	ffff	ffff

说明： 将寄存器 f 的内容递减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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 → dest，
如果结果 = 0 则跳过

受影响的状态位： 无

机器码：

0010	11da	ffff	ffff

说明： 将寄存器 f 的内容递减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果结果为 0，则丢弃已取的下一条指令转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 DECFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过，且后面跟有 2 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，非 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 → dest，
如果结果 ≠ 0 则跳过

受影响的状态位： 无

机器码：

0100	11da	ffff	ffff

说明： 将寄存器 f 的内容递减 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果结果不为 0，则丢弃已取的下一条指令转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 DCFSNZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过，且后面跟有 2 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

机器码：

第一个字 （k<7:0>）	1110	1111	$k_7$kkk	kkkk$_0$
第二个字 （k<19:8>）	1111	k_{19}kkk	kkkk	kkkk$_8$

说明： GOTO 指令允许无条件跳转到整个 2 MB 存储器范围中的任何位置。将 20 位值 k 装入 PC<20:1>。 GOTO 始终为双周期指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

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 → dest

受影响的状态位： C、 DC、 N、 OV 和 Z

机器码：

0010	10da	ffff	ffff

说明： 将寄存器 f 的内容递增 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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 → dest，
如果结果 = 0 则跳过

受影响的状态位： 无

机器码：

0011	11da	ffff	ffff

说明： 将寄存器 f 的内容递增 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果结果为 0，则丢弃已取的下一条指令转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 INCFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过，且后面跟有 2 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，非 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 → dest，
如果结果 ≠ 0 则跳过

受影响的状态位： 无

机器码：

0100	10da	ffff	ffff

说明： 将寄存器 f 的内容递增 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果结果不为 0，则丢弃已取的下一条指令转而执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 INFSNZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

如果跳过：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过，且后面跟有 2 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 **将立即数与 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

机器码：

0000	1001	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 k 进行逻辑或运算。结果存储在 W 寄存器中。

指令字数： 1

指令周期数： 1

Q 周期操作：

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 **将 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) → dest

受影响的状态位： N 和 Z

机器码：

0001	00da	ffff	ffff

说明： 将 W 的内容与寄存器 f 的内容进行逻辑或运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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 **装入 FSR**

语法： LFSR f, k

操作数： 0 ≤ f ≤ 2
0 ≤ k ≤ 4095

操作： k → FSRf

受影响的状态位： 无

机器码：

1110 1111	1110 0000	00ff $k_7$kkk	k_{11}kkk kkkk

说明： 将 12 位立即数 k 装入 f 所指向的文件选择寄存器。

指令字数： 2

指令周期数： 2

Q 周期操作：

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 的 LSB 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 → dest

受影响的状态位： N 和 Z

机器码：

0101	00da	ffff	ffff

说明： 根据 d 的状态，将寄存器 f 的内容送入目标寄存器。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。f 可以为 256 字节存储区中的任何地址单元。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 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 **将源寄存器的内容送入目标寄存器**

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

机器码：

第一个字（源） 1100	ffff	ffff	$ffff_s$
第二个字（目标） 1111	ffff	ffff	$ffff_d$

说明： 将源寄存器 f_s 的内容送入目标寄存器 f_d。源寄存器 f_s 可以是 4096 字节数据空间（000h 至 FFFh）中的任何单元，目标寄存器 f_d 也可以是 000h 至 FFFh 中的任何单元。

源或目标寄存器都可以是 W（这是个有用的特例）。

MOVFF 指令对于将数据存储单元中的内容送入外设寄存器（如发送缓冲区或 I/O 端口）的场合非常有用。

MOVFF 指令不能使用 PCL、TOSU、TOSH 或 TOSL 作为目标寄存器。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f（源寄存器）	处理数据	空操作
译码	空操作 无假读	空操作	写寄存器 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

MOVLB **将立即数送入 BSR 的低半字节**

语法： MOVLW k

操作数： $0 \le k \le 255$

操作： $k \rightarrow BSR$

受影响的状态位： 无

机器码：

0000	0001	kkkk	kkkk

说明： 将 8 位立即数 k 装入存储区选择寄存器（BSR）。不管 k_7:k_4 的值如何，BSR<7:4> 的值将始终保持为 0。

指令字数： 1

指令周期数： 1

Q 周期操作：

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 **将立即数送入 W**

语法： MOVLW k

操作数： 0 ≤ k ≤ 255

操作： k → W

受影响的状态位： 无

机器码：

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 周期操作：

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

机器码：

0110	111a	ffff	ffff

说明： 将 W 寄存器中的数据送入寄存器 f。f 可以是256字节存储区中的任何地址单元。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 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 **将立即数与 W 中的内容相乘**

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

机器码：

0000	1101	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 k 进行无符号的乘法运算。16 位的结果存储在 PRODH:PRODL 寄存器对中，其中 PRODH 用于存储高字节。

W 的内容不改变。

所有状态标志位都不受影响。

请注意此操作不可能发生溢出或进位。结果有可能为全零，但不会被检测到。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写寄存器 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 **将 W 与 f 的内容相乘**

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

机器码：

0000	001a	ffff	ffff

说明： 将 W 的内容与寄存器单元 f 的内容执行无符号的乘法运算。运算的 16 位结果保存在 PRODH:PRODL 寄存器对中，其中 PRODH 用于存储高字节。W 和 f 的内容都不改变。

所有状态标志位都不受影响。

请注意此操作不可能发生溢出或进位。结果有可能为全零，但不会被检测到。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 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 **对 f 取补**

语法： NEGF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： $(\bar{f}) + 1 \rightarrow f$

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0110	110a	ffff	ffff

说明： 用二进制补码对存储单元 f 取补，结果存储在数据存储单元 f 中。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 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： 无操作

受影响的状态位： 无

机器码：

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 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作

示例：

无。

POP **弹出返回堆栈栈顶的内容**

语法： POP

操作数： 无

操作： (TOS) → 位桶 （即丢弃）

受影响的状态位： 无

机器码：

0000	0000	0000	0110

说明： 从返回堆栈弹出 TOS 值并丢弃。然后，前一个压入返回堆栈的值成为 TOS 值。此指令可以让用户正确管理返回堆栈，从而实现软件堆栈。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	弹出 TOS 值	空操作

示例：

```
POP
GOTO    NEW
```

执行指令前
TOS = 0031A2h
堆栈 （下一级） = 014332h

执行指令后
TOS = 014332h
PC = NEW

PUSH **将数据压入返回堆栈栈顶**

语法： PUSH

操作数： 无

操作： (PC + 2) → TOS

受影响的状态位： 无

机器码：

0000	0000	0000	0101

说明： PC + 2 的值被压入返回堆栈的栈顶。原先的 TOS 值被压入堆栈的下一级。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 周期操作：

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 （下一级） = 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

机器码：

1101	1nnn	nnnn	nnnn

说明： 从当前地址跳转（最多 1K）来调用子程序。首先，将返回地址（PC + 2）压入堆栈。然后，将 “2n”（以二进制补码表示）与 PC 相加。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 周期操作：

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： 将所有受 $\overline{\text{MCLR}}$ 复位影响的寄存器和标志位复位。

受影响的状态位： 全部

机器码：

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 周期操作：

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
标志位 * = 复位值

RETFIE **从中断返回**

语法： RETFIE {s}

操作数： s ∈ [0,1]

操作： (TOS) → PC，
1 → GIE/GIEH 或 PEIE/GIEL；
如果 s = 1，
(WS) → W，
(STATUSS) → STATUS，
(BSRS) → BSR，
PCLATU 和 PCLATH 保持不变

受影响的状态位： GIE/GIEH 和 PEIE/GIEL

机器码：

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。通过将高或低优先级全局中断允许位置 1，来允许中断。如果 s = 1，则影子寄存器 WS、STATUSS 和 BSRS 的内容将被装入对应的寄存器 W、STATUS 和 BSR。如果 s = 0，则不更新这些寄存器（默认）。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	从堆栈弹出 PC 值 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
    W                   =   WS
    BSR                 =   BSRS
    STATUS              =   STATUSS
    GIE/GIEH，PEIE/GIEL =   1
```

RETLW **将立即数返回到 W**

语法： RETLW k

操作数： 0 ≤ k ≤ 255

操作： k → W，
(TOS) → PC，
PCLATU 和 PCLATH 保持不变

受影响的状态位： 无

机器码：

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。高字节地址锁存器（PCLATH）内容保持不变。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	从堆栈弹出 PC 值，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 的值
```

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 保持不变

受影响的状态位： 无

机器码：

0000	0000	0001	001s

说明： 从子程序返回。执行出栈操作，将栈顶（TOS）的内容装入程序计数器。如果 s = 1，则影子寄存器 WS、STATUSS 和 BSRS的内容将被装入对应的寄存器W、STATUS 和 BSR。如果 s = 0，则不更新这些寄存器（默认）。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	从堆栈弹出 PC 值
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

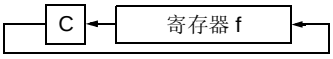
机器码：

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，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。



指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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

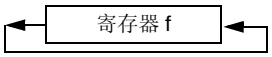
机器码：

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，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。



指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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

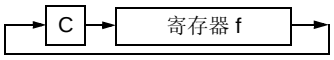
机器码：

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，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。



指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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

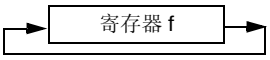
机器码：

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，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果 a 为 0，选择快速操作存储区，忽略 BSR 的值。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，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。



指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1：

```
RRNCF   REG, 1, 0
```

执行指令前
REG = 1101 0111
执行指令后
REG = 1110 1011

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 的内容置为全 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

机器码：

0110	100a	ffff	ffff

说明： 将指定寄存器的内容置为 FFh。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 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 $\overline{PD}$

机器码：

0000	0000	0000	0011

说明： 掉电状态位（$\overline{PD}$）清零。超时状态位（$\overline{TO}$）置 1。看门狗定时器及其后分频器清零。

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 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	进入休眠模式

示例： SLEEP

执行指令前
$\overline{TO}$ = ?
$\overline{PD}$ = ?

执行指令后
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† 如果由 WDT 引起唤醒，则该位将被清零。

SUBFWB **W 减去 f （带借位）**

语法： SUBFWB f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) – (f) – ($\overline{C}$) → dest

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0101	01da	ffff	ffff

说明： 将 W 的内容减去 f 寄存器的内容和进位标志位（借位）（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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

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

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 ；结果为全零
N = 0

SUBLW **立即数减去 W 的内容**

语法： SUBLW k

操作数： 0 ≤ k ≤ 255

操作： k – (W) → W

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0000	1000	kkkk	kkkk

说明： 用 8 位立即数 k 减去 W。结果存储在 W 寄存器中。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

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

例 2： SUBLW 02h

执行指令前
W = 02h
C = ?
执行指令后
W = 00h
C = 1 ; 结果为全零
Z = 1
N = 0

例 3： SUBLW 02h

执行指令前
W = 03h
C = ?
执行指令后
W = FFh ; （二进制补码）
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) → dest

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0101	11da	ffff	ffff

说明： 用寄存器 f 中的内容减去 W 寄存器的内容（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f （默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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

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 ; 结果为全零
Z = 1
N = 0

例 3： SUBWF REG, 1, 0

执行指令前
REG = 1
W = 2
C = ?
执行指令后
REG = FFh ; （二进制补码）
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) – (C̄) → dest

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0101	10da	ffff	ffff

说明： 用 f 寄存器的内容减去 W 的内容和进位标志位（借位）（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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 1011）
    W    =  0Dh     （0000 1101）
    C    =  1
    Z    =  0
    N    =  0       ；结果为正
```

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       ；结果为全零
    N    =  0
```

例 3：

```
SUBWFB  REG, 1, 0
执行指令前
    REG  =  03h     （0000 0011）
    W    =  0Eh     （0000 1101）
    C    =  1
执行指令后
    REG  =  F5h     （1111 0100）
                    ；[二进制补码]
    W    =  0Eh     （0000 1101）
    C    =  0
    Z    =  0
    N    =  1       ；结果为负
```

SWAPF **将 f 的高半字节和低半字节交换**

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

机器码：

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，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节 “ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

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： 如果执行 TBLRD *,
(程序存储单元 (TBLPTR)) → TABLAT ;
TBLPTR 不改变
如果执行 TBLRD *+,
(程序存储单元 (TBLPTR)) → TABLAT ;
(TBLPTR) + 1 → TBLPTR
如果执行 TBLRD *-,
(程序存储单元 (TBLPTR)) → TABLAT ;
(TBLPTR) – 1 → TBLPTR
如果执行 TBLRD +*,
(TBLPTR) + 1 → TBLPTR ;
(程序存储单元 (TBLPTR)) → TABLAT

受影响的状态位： 无

机器码：

0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*

说明： 此指令用于读取程序存储单元（P.M.）的内容。使用表指针（TBLPTR） 对程序存储单元进行寻址。

TBLPTR（一个 21 位指针）指向程序存储器中的每个字节。TBLPTR 的寻址范围为 2 MB。

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 指令可用如下方法修改 TBLPTR 的值 ：

- 不变
- 后递增
- 后递减
- 预递增

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作 （读程序存储器）	空操作	空操作 （写 TABLAT）

TBLRD **表读 （续）**

例 1： `TBLRD *+ ;`

执行指令前
TABLAT = 55h
TBLPTR = 00A356h
存储单元 （00A356h） = 34h
执行指令后
TABLAT = 34h
TBLPTR = 00A357h

例 2： `TBLRD +* ;`

执行指令前
TABLAT = AAh
TBLPTR = 01A357h
存储单元 （01A357h） = 12h
存储单元 （01A358h） = 34h
执行指令后
TABLAT = 34h
TBLPTR = 01A358h

TBLWT **表写**

语法： TBLWT (*; *+; *-; +*)

操作数： 无

操作： 如果执行 TBLWT*，
(TABLAT) → 保持寄存器；
TBLPTR 不改变
如果执行 TBLWT*+，
(TABLAT) → 保持寄存器；
(TBLPTR) + 1 → TBLPTR
如果执行 TBLWT*-，
(TABLAT) → 保持寄存器；
(TBLPTR) – 1 → TBLPTR
如果执行 TBLWT+*，
(TBLPTR) + 1 → TBLPTR；
(TABLAT) → 保持寄存器

受影响的状态位： 无

机器码：

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

说明： 此指令使用 TBLPTR 的低 3 位来确定要将 TABLAT 中的内容写入 8 个保持寄存器中的哪一个。该保持寄存器用于对程序存储单元（P.M.）的内容编程。（关于对闪存程序存储器编程的更多详细信息，请参见**第 6.0 节“存储器构成”**。）

TBLPTR（一个 21 位指针）指向程序存储器中的每个字节。TBLPTR 的寻址范围为 2 MB。TBLPTR 的 LSb 选择要访问程序存储单元的哪个字节。

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 指令可用如下方法修改 TBLPTR 的值：

- 不变
- 后递增
- 后递减
- 预递增

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作（读 TABLAT）	空操作	空操作（写保持寄存器）

TBLWT **表写（续）**

例 1： TBLWT *+;

执行指令前
TABLAT = 55h
TBLPTR = 00A356h
保持寄存器（00A356h）= FFh
执行指令后（表写操作完成）
TABLAT = 55h
TBLPTR = 00A357h
保持寄存器（00A356h）= 55h

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（表写操作完成）
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： f = 0 则跳过

受影响的状态位： 无

机器码：

0110	011a	ffff	ffff

说明： 如果 f = 0，丢弃执行当前指令过程中已取的下一条指令并执行一条 NOP 指令，使该指令成为双周期指令。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中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，且后面跟有 2 字指令，则执行 TSTFSZ 需要 3 个周期。

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 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，且后面跟有 2 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 **将立即数与 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

机器码：

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。结果存储在 W 寄存器中。

指令字数： 1

指令周期数： 1

Q 周期操作：

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 **将 W 与 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) → dest

受影响的状态位： N 和 Z

机器码：

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，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 f≤95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请参见**第 29.2.3 节“ 立即数变址寻址模式中针对字节和针对位的指令 ”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例：

```
XORWF   REG, 1, 0
```

执行指令前
REG = AFh
W = B5h
执行指令后
REG = 1Ah
W = B5h

29.2 扩展指令集

除了PIC18指令集的75条标准指令之外，PIC18F66K80系列器件还提供了针对核心CPU功能的可选扩展指令。这些新增的功能包括 8 条额外的指令，增加了间接和变址寻址操作，并使得许多标准 PIC18 指令可以实现立即数变址寻址。

扩展指令集的额外功能在默认情况下对未编程器件是使能的。用户必须在编程期间将 XINST 配置位正确置 1 或清零，从而使能或禁止这些功能。

扩展指令集中的指令可以全部被归为立即数操作类指令，它们既可以对文件选择寄存器进行操作，也可以使用这些寄存器进行变址寻址。还为其中两条指令 ADDFSR 和 SUBFSR 提供了使用 FSR2 的特例形式，即 ADDULNK 和 SUBULNK，这两条指令允许在执行后自动返回。

这些扩展的指令专门用于优化用高级语言特别是C语言编写的可重入程序代码（也就是递归或使用软件堆栈的代码）。此外，它们使用户能更高效地用高级语言对数据结构执行某些操作。这些操作包括：

- 在进入和退出子程序时对软件堆栈空间进行动态分配和释放
- 函数指针调用
- 对软件堆栈指针进行操作
- 对软件堆栈中的变量进行操作

表 29-3 提供了扩展指令集中的指令汇总。**第 29.2.2 节 “ 扩展指令集 ”** 对这些指令进行了详细说明。表 29-1（第 488 页）提供了标准和扩展的 PIC18 指令集的操作码字段说明。

> **注：** 扩展指令集和立即数变址寻址模式是专为优化用 C 语言编写的应用程序而设计的，用户可能不会在汇编程序中直接使用这些指令。对于那些需要查看编译器生成代码的用户，这些命令的语法可作为参考。

29.2.1 扩展指令的语法

大部分扩展指令都使用变址参数，使用一个文件选择寄存器和某一偏移量来指定源寄存器或目标寄存器。当指令的参数作为变址寻址的一部分时，会用方括号 （“[ ]”）将它括起来。这用于表示此参数用作变址或偏移量。如果 MPASM™ 汇编器发现一个变址或偏移量没有被括起来，它就会给出出错信息。

当使能扩展指令集时，方括号也用于表示针对字节和针对位的指令中的变址参数。这是对指令语法的额外更改。更多详细信息，请参见**第 29.2.3.1 节 “ 标准 PIC18 命令的扩展指令语法 ”**。

> **注：** 以前，在 PIC18 和早期的指令集中使用方括号来表示可选参数。在此文本和以后的文本中，可选参数将用大扩号 （“{ }”）表示。

表 29-3： 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	
ADDFSR	f, k	将立即数加到 FSR	1	1110	1000	ffkk	kkkk	无
ADDULNK	k	将立即数加到 FSR2 并返回	2	1110	1000	11kk	kkkk	无
CALLW		使用 WREG 调用子程序	2	0000	0000	0001	0100	无
MOVSF	z_s, f_d	将 z_s （源）移入 第一个字	2	1110	1011	0zzz	zzzz	无
		f_d （目标） 第二个字		1111	ffff	ffff	ffff	
MOVSS	z_s, z_d	将 z_s （源）移入 第一个字	2	1110	1011	1zzz	zzzz	无
		z_d （目标） 第二个字		1111	xxxx	xzzz	zzzz	
PUSHL	k	将立即数保存到 FSR2， FSR2 递减 1	1	1110	1010	kkkk	kkkk	无
SUBFSR	f, k	FSR 减去立即数	1	1110	1001	ffkk	kkkk	无
SUBULNK	k	FSR2 减去立即数并返回	2	1110	1001	11kk	kkkk	无

29.2.2 扩展指令集

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

机器码：

1110	1000	ffkk	kkkk

说明： 将 6 位立即数 k 加到由 f 指定的 FSR 的内容。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
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

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

机器码：

1110	1000	11kk	kkkk

说明： 将 6 位立即数 k 加到 FSR2 的内容。然后通过将 TOS 的值装入 PC，执行 RETURN。

执行该指令需要两个周期；在第二个周期执行一条 NOP 指令。

该指令可以被认为是 ADDFSR 指令的特例，其中 f = 3（二进制 “11”），它仅针对 FSR2 进行操作。

指令字数： 1

指令周期数： 2

Q 周期操作：

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： 所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数，用于符号寻址。如果使用了标号，那么指令格式将变为：{label} 指令参数。

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

机器码：

0000	0000	0001	0100

说明 首先，返回地址（PC + 2）被压入返回堆栈。接下来，将 W 寄存器的内容写入 PCL，PCL 现有的值被丢弃。然后，PCLATH 和 PCLATU 的内容被分别锁存到 PCH 和 PCU。第二个周期执行一条 NOP 指令，并同时取下一条新指令。

和 CALL 不一样，该指令没有更新 W、STATUS 或 BSR 寄存器的选项。

指令字数： 1

指令周期数： 2

Q 周期操作：

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

MOVSF **将变址寻址单元内容送入 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

机器码：

第一个字（源）	1110	1011	0zzz	$zzzz_s$
第二个字（目标）	1111	ffff	ffff	$ffff_d$

说明： 将源寄存器的内容送入目标寄存器 f_d。通过将第一个字中的 7 位立即数偏移量 z_s 与FSR2的值相加来确定源寄存器的实际地址。第二个字中的 12 位立即数 f_d 指向目标寄存器的地址。两个地址均可以是 4096 字节的数据空间（000h 至 FFFh）中的任何存储单元。

MOVSF 指令不能使用 PCL、TOSU、TOSH 或 TOSL 作为目标寄存器。

如果计算得到的源地址指向间接寻址寄存器，将返回 00h。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	读源寄存器
译码	空操作 无假读	空操作	写寄存器 f（目标寄存器）

示例： MOVSF [05h], REG2

执行指令前
FSR2 = 80h
85h 单元的内容 = 33h
REG2 = 11h

执行指令后
FSR2 = 80h
85h 单元的内容 = 33h
REG2 = 33h

MOVSS **在变址寻址单元之间传送数据**

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

机器码：

第一个字（源） 1110	1011	1zzz	$zzzz_s$
第二个字（目标） 1111	xxxx	xzzz	$zzzz_d$

说明： 将源寄存器的内容送入目标寄存器。通过将 FSR2 中的值分别加上 7 位立即数偏移量 z_s 和 z_d 来确定源寄存器和目标寄存器的地址。两个寄存器都可以是 4096 字节数据存储空间（000h 至 FFFh）中的任意存储单元。

MOVSS 指令不能使用 PCL、TOSU、TOSH 或 TOSL 作为目标寄存器。

如果计算得到的源地址指向间接寻址寄存器，将返回 00h。如果计算得到的目标地址指向间接寻址寄存器，将执行一条 NOP 指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	确定源地址	确定源地址	确定源地址
译码	确定目标地址	确定目标地址	写目标寄存器

示例： MOVSS [05h], [06h]

执行指令前
FSR2 = 80h
85h 单元的内容 = 33h
86h 单元的内容 = 11h

执行指令后
FSR2 = 80h
85h 单元的内容 = 33h
86h 单元的内容 = 33h

PUSHL **将立即数保存到 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

机器码：

1111	1010	kkkk	kkkk

说明： 8 位立即数 k 被写入由 FSR2 指定的数据存储单元。操作完后 FSR2 递减 1。

此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 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写入目标寄存器

示例： PUSHL 08h

执行指令前
FSR2H:FSR2L = 01ECh
存储单元（01ECh） = 00h

执行指令后
FSR2H:FSR2L = 01EBh
存储单元（01ECh） = 08h

SUBFSR **FSR 减去立即数**

语法： SUBFSR f, k

操作数： 0 ≤ k ≤ 63

f ∈ [0, 1, 2]

操作： FSRf – k → FSRf

受影响的状态位： 无

机器码：

1110	1001	ffkk	kkkk

说明： 用 f 指定的 FSR 的内容减去 6 位立即数 k。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： SUBFSR 2, 23h

执行指令前
FSR2 = 03FFh

执行指令后
FSR2 = 03DCh

SUBULNK **FSR2 减去立即数并返回**

语法： SUBULNK k

操作数： 0 ≤ k ≤ 63

操作： FSR2 – k → FSR2，
(TOS) → PC

受影响的状态位： 无

机器码：

1110	1001	11kk	kkkk

说明： 用 FSR 的内容减去 6 位立即数 k，然后通过将 TOS 的值装入 PC，执行 RETURN。

执行该指令需要两个指令周期，第二个指令周期执行一条 NOP 指令。

该指令可以被认为是 SUBFSR 指令的特例，其中 f = 3（二进制 “11”）；它仅针对 FSR2 进行操作。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器
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)

29.2.3 立即数变址寻址模式中针对字节和针对位的指令

注：	使能 PIC18 扩展指令集可能导致常规应用程序运行不正常或完全失败。

一旦使能扩展指令集，除了可以使用扩展指令集中的8条新命令之外，还将使能立即数变址寻址模式（**第 6.6.1 节 “ 使用立即数偏移量进行变址寻址 ”**）。这将导致标准 PIC18 指令集中大部分指令的地址解析方法有很大变化。

当禁止扩展指令集时，嵌入在操作码中的地址被视为立即数存储单元：可以是快速操作存储区中的存储单元（a = 0），或由 BSR 指定的 GPR 存储区中的存储单元（a = 1）。当使能扩展指令集且 a = 0 时，地址小于或等于 5Fh 的文件寄存器参数被解析为 FSR2 中的指针值的偏移量，而不是一个立即数地址。对于实际应用来说，这意味着所有使用快速操作 RAM 位作为参数的指令，即所有针对字节或针对位的指令，或者几乎半数的核心 PIC18 指令，在使能了扩展指令集时操作都会有所不同。

当 FSR2 的内容为 00h 时，快速操作 RAM 的边界会被重新映射到它们的原始值。这对于编写向下兼容的代码很有用处。如果使用此技术，有必要在 C 程序调用汇编子程序时保存 FSR2 的值并在返回时将它恢复，这样做的目的是保护堆栈指针。用户还必须记住扩展指令集的语法要求（见**第 29.2.3.1 节 “ 标准 PIC18 命令的扩展指令语法 ”**）。

虽然立即数变址寻址模式对于动态堆栈和指针操作很有用处，但是如果不小心对错误的寄存器进行了简单的算术运算也会非常麻烦。已经习惯使用 PIC18 编程的用户必须记住，在使能了扩展指令集后，地址小于或等于 5Fh 的寄存器用于立即数变址寻址。

下页提供了在立即数变址寻址模式中，一些针对字节和位的指令的代表示例，通过这些示例可以看出指令执行如何受到影响。示例中的操作数条件适用于所有这些类型的指令。

29.2.3.1 标准 PIC18 命令的扩展指令语法

当使能了扩展指令集时，立即数偏移量 “k” 被用来替换标准的针对字节和位的命令中的文件寄存器参数 “f”。如前所述，只有在 “f” 小于或等于 5Fh 时才会发生这种情况。当使用偏移量时，偏移量必须用方括号 “[ ]” 标出。因为在扩展指令集中，编译器将方括号中的值解析为变址地址或偏移量。省略方括号，或在方括号内使用大于 5Fh 的值会在 MPASM™ 汇编器中产生错误。

如果变址参数已被正确加上了方括号，那么就不再需要指定快速操作 RAM 参数；此参数被自动假定为 0。这与标准操作（禁止扩展指令集时）刚好相反。在变址寻址模式中，声明快速操作 RAM 位也将在 MPASM 汇编器中产生错误。

目标参数 “d” 的操作和以前一样。

在 MPASM 汇编器的最新版本中，必须明确启用对扩展指令集的语言支持。可以通过命令行选项 /y 或在源代码中加入 PE 伪指令启用。

29.2.4 使能扩展指令集时的注意事项

需要注意的是，并非所有用户都有必要使用扩展指令集，尤其是那些不使用软件堆栈的用户。

此外，立即数变址寻址模式可能会给写入 PIC18 汇编器的常规应用程序带来问题。这是因为常规的指令会尝试寻址快速操作存储区中地址低于 5Fh 的寄存器。当使能了扩展指令集时，这些地址被解析为相对于 FSR2 的立即数偏移量，所以应用程序可能会读或写错误的地址。

将应用程序移植到 PIC18F66K80 系列器件时，考虑代码的类型是非常重要的。在使用扩展指令集时，用 C 语言编写的代码较长的可重入应用程序会运行得很好，而大量使用快速操作存储区的常规应用程序不会获得任何益处。

ADDWF **将 W 与变址寻址单元的内容相加（立即数变址寻址模式）**

语法： ADDWF [k] {,d}

操作数： 0 ≤ k ≤ 95
d ∈ [0,1]

操作： (W) + ((FSR2) + k) → dest

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0010	01d0	kkkk	kkkk

说明： 将 W 的内容与由 FSR2 加上偏移量 k 指定的寄存器的内容相加。

如果 d 为 0，结果存储在 W 中。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 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写入目标寄存器

示例： ADDWF [OFST],0

执行指令前
W = 17h
OFST = 2Ch
FSR2 = 0A00h
0A2Ch 单元的内容 = 20h
执行指令后
W = 37h
0A2Ch 单元的内容 = 20h

BSF **将变址寻址单元相应位置 1（立即数变址寻址模式）**

语法： BSF [k], b

操作数： 0 ≤ f ≤ 95
0 ≤ b ≤ 7

操作： 1 → ((FSR2) + k)<b>

受影响的状态位： 无

机器码：

1000	bbb0	kkkk	kkkk

说明： 将由 FSR2 加上偏移量 k 指定的寄存器中的位 b 置 1。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例： BSF [FLAG_OFST], 7

执行指令前
FLAG_OFST = 0Ah
FSR2 = 0A00h
0A0Ah 单元的内容 = 55h
执行指令后
0A0Ah 单元的内容 = D5h

SETF **将变址寻址单元置全 1（立即数变址寻址模式）**

语法： SETF [k]

操作数： 0 ≤ k ≤ 95

操作： FFh → ((FSR2) + k)

受影响的状态位： 无

机器码：

0110	1000	kkkk	kkkk

说明： 将由 FSR2 加上偏移量 k 指定的寄存器的内容置为 FFh。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 k	处理数据	写寄存器

示例： SETF [OFST]

执行指令前
OFST = 2Ch
FSR2 = 0A00h
0A2Ch 单元的内容 = 00h
执行指令后
0A2Ch 单元的内容 = FFh

29.2.5 使用 MPLAB® IDE 工具的注意事项

最新版本的 Microchip 软件工具完全支持 PIC18F66K80 系列器件的扩展指令集。软件工具包括 MPLAB C18 C 编译器、MPASM 汇编器和 MPLAB 集成开发环境（Integrated Development Environment，IDE）。

在选择了目标器件进行软件开发后，MPLAB IDE 将自动按默认模式设置该器件的配置位。XINST 配置位的默认设置是 0，禁止扩展指令集和立即数变址寻址模式。在编程时必须将 XINST 位置 1 才能确保使用扩展指令集开发的应用程序能够正确执行。

要使用扩展指令集开发软件，用户必须设置他们的语言工具以实现对扩展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法：

- 开发环境中的菜单选项或对话框，允许用户配置项目的语言工具及其设置
- 命令行选项
- 源代码中的伪指令

这些选项在不同的编译器、汇编器和开发环境中将有所不同。建议用户在其开发系统所附带的文档中查询相应的文档。

30.0 开发支持

一系列软件及硬件开发工具对PIC® 单片机和dsPIC® 数字信号控制器提供支持：

- 集成开发环境
  - MPLAB® IDE 软件
- 编译器 / 汇编器 / 链接器
  - 适用于各种器件系列的 MPLAB C 编译器
  - 适用于各种器件系列的 HI-TECH C 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 SIM 软件模拟器
- 仿真器
  - MPLAB REAL ICE™ 在线仿真器
- 在线调试器
  - MPLAB ICD 3
  - PICkit™ 3 Debug Express
- 器件编程器
  - PICkit™ 2 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

30.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16/32 位单片机市场提供了前所未有的易于使用的软件开发平台。 MPLAB IDE 是基于 Windows® 操作系统的应用软件，包括：

- 一个包含所有调试工具的图形界面
  - 模拟器
  - 编程器 （单独销售）
  - 在线仿真器 （单独销售）
  - 在线调试器 （单独销售）
- 具有彩色上下文代码显示的全功能编辑器
- 多项目管理器
- 内容可直接编辑的可定制式数据窗口
- 高级源代码调试
- 鼠标停留在变量上进行查看的功能
- 将变量从源代码窗口拖放到 Watch （观察）窗口
- 丰富的在线帮助
- 集成了可选的第三方工具，如 IAR C 编译器

MPLAB IDE 可以让您：

- 编辑源文件 （C 语言或汇编语言）
- 点击一次即可完成编译或汇编，并将代码下载到仿真器和模拟器工具中 （自动更新所有项目信息）
- 可使用如下各项进行调试：
  - 源文件 （C 语言或汇编语言）
  - 混合 C 语言和汇编语言
  - 机器码

MPLAB IDE 在单个开发范例中支持使用多种调试工具，包括从成本效益高的模拟器到低成本的在线调试器，再到全功能的仿真器。这样缩短了用户升级到更加灵活而功能强大的工具时的学习时间。

30.2 适用于各种器件系列的 MPLAB C 编译器

MPLAB C 编译器代码开发系统是完全的 ANSI C 编译器，适用于 Microchip 的 PIC18、PIC24 和 PIC32 系列单片机及 dsPIC30 和 dsPIC33 系列数字信号控制器。这些编译器提供强大的集成功能和出众的代码优化能力，且使用方便。

为便于源代码调试，编译器提供针对 MPLAB IDE 调试器优化的符号信息。

30.3 适用于各种器件系列的 HI-TECH C 编译器

HI-TECH C 编译器代码开发系统是完全的 ANSI C 编译器，适用于 Microchip 的 PIC 系列单片机及 dsPIC 系列数字信号控制器。这些编译器提供强大的集成功能和全知代码生成能力，且使用方便。

为便于源代码调试，编译器提供针对 MPLAB IDE 调试器优化的符号信息。

编译器包括一个宏汇编器、链接器、预处理程序和单步驱动程序，可以在多种平台上运行。

30.4 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 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

30.5 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器包含了由 MPASM 汇编器、MPLAB C18 C 编译器产生的可重定位目标。通过使用链接器脚本中的指令，它还可链接预编译库中的可重定位目标。

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

30.6 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、PIC32 和 dsPIC 器件从符号汇编语言生成可重定位机器码。 MPLAB C 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
- 支持定点数据和浮点数据
- 命令行界面
- 丰富的指令集
- 灵活的宏语言
- MPLAB IDE 兼容性

30.7 MPLAB SIM 软件模拟器

MPLAB 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 SIM 软件模拟器完全支持使用 MPLAB C 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。

30.8 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 器件而推出的新一代高速仿真器。结合 MPLAB 集成开发环境 （IDE）所具有的易于使用且功能强大的图形用户界面，该仿真器可对PIC®闪存MCU和 dsPIC® 闪存 DSC 进行调试和编程。IDE 是随每个工具包一起提供的。

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（RJ11）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 IDE 下载将来版本的固件，对该仿真器进行现场升级。在即将推出的 MPLAB IDE 版本中，会支持许多新器件，还将增加一些新特性。在同类仿真器中，MPLAB REAL ICE 的优势十分明显：低成本、全速仿真、运行时变量查看、跟踪分析、复杂断点、耐用的探针接口及较长 （长达 3 米）的互连电缆。

30.9 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 闪存数字信号控制器 （DSC）和单片机 （MCU）器件。结合 MPLAB 集成开发环境 （IDE）所具有的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 转接器。

30.10 PICkit 3 在线调试器 / 编程器及 PICkit 3 Debug Express

结合 MPLAB 集成开发环境 （IDE）所具有的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（RJ-11）连接器（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。

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）。

30.11 PICkit 2 开发编程器 / 调试器及 PICkit 2 Debug Express

PICkit™ 2 开发编程器 / 调试器是一款低成本开发工具，具有易于使用的界面，适用于对 Microchip 的闪存系列单片机进行编程和调试。这一全功能的 Windows® 编程界面支持低档（PIC10F、PIC12F5xx 和 PIC16F5xx）、中档（PIC12F6xx 和 PIC16F）、PIC18F、PIC24、dsPIC30、dsPIC33 和 PIC32 系列的 8 位、16 位及 32 位单片机，以及许多 Microchip 串行 EEPROM 产品。结合 Microchip 功能强大的 MPLAB 集成开发环境（IDE），PICkit 2 可对大多数 PIC® 单片机进行在线调试。即使 PIC 单片机已嵌入应用，在线调试功能仍可以运行、暂停和单步执行程序。在断点处暂停时，可以检查和修改文件寄存器。

PICkit 2 Debug Express 包括 PICkit 2、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）。

30.12 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 VDDMIN 和 VDDMAX 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 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。

30.13 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）。

31.0 电气特性

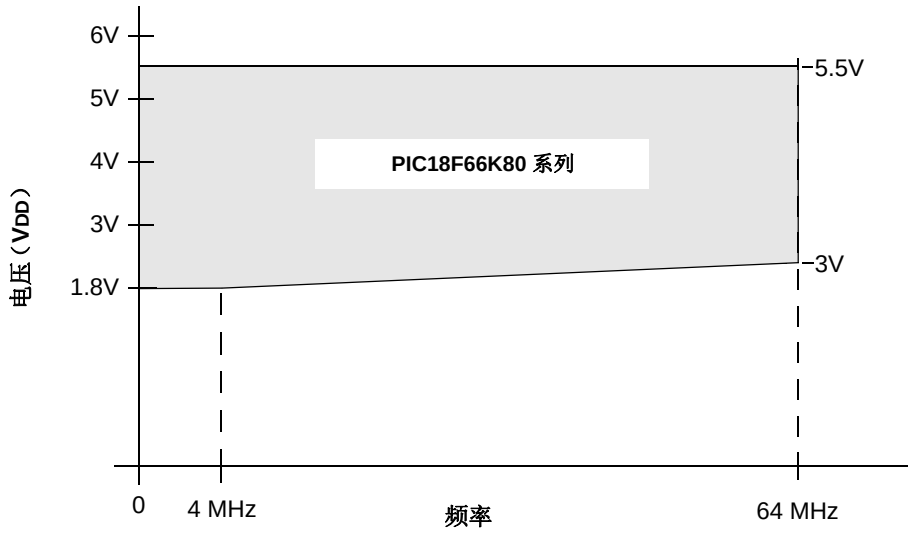
绝对最大值 (†)

环境温度..... -40°C 至 +125°C
储存温度..... -65°C 至 +150°C
$\overline{MCLR}$ 引脚相对于 VSS 的电压 .....-0.3V 至 9.0V
任一仅用作数字功能的 I/O 引脚相对于 VSS 的电压 （VDD 除外）.....-0.3V 至 7.5V
任一数模组合引脚相对于 VSS 的电压 （VDD 和 $\overline{MCLR}$ 除外）.....-0.3V 至 （VDD + 0.3V）
VDD 引脚相对于 VSS 的电压 （PIC18F66K80）.....-0.3V 至 7.5V
VDD 引脚相对于 VSS 的电压 （PIC18LF66K80）.....-0.3V 至 3.66V
总功耗 **（注 1）**.....1W
流出 VSS 引脚的最大电流.....300 mA
流入 VDD 引脚的最大电流 .....250 mA
输入钳位电流 IIK （VI < 0 或 VI > VDD）.....±20 mA
输出钳位电流 IOK （VO < 0 或 VO > VDD）.....±20 mA
PORTA<7:6>、任一 PORTB 和 PORTC I/O 引脚的最大输出灌电流 .....25 mA
任一 PORTD 和 PORTE I/O 引脚的最大输出灌电流.....8 mA
PORTA<5:0>、任一 PORTF 和 PORTG I/O 引脚的最大输出灌电流 .....2 mA
PORTA<7:6>、任一 PORTB 和 PORTC I/O 引脚的最大输出拉电流 .....25 mA
任一 PORTD、 PORTE 和 PORTJ I/O 引脚的最大输出拉电流.....8 mA
PORTA<5:0>、任一 PORTF、 PORTG 和 PORTH I/O 引脚的最大输出拉电流 .....2 mA
所有端口的最大总灌电流 .....200 mA

注 1： 功耗按如下公式计算：
$Pdis = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$

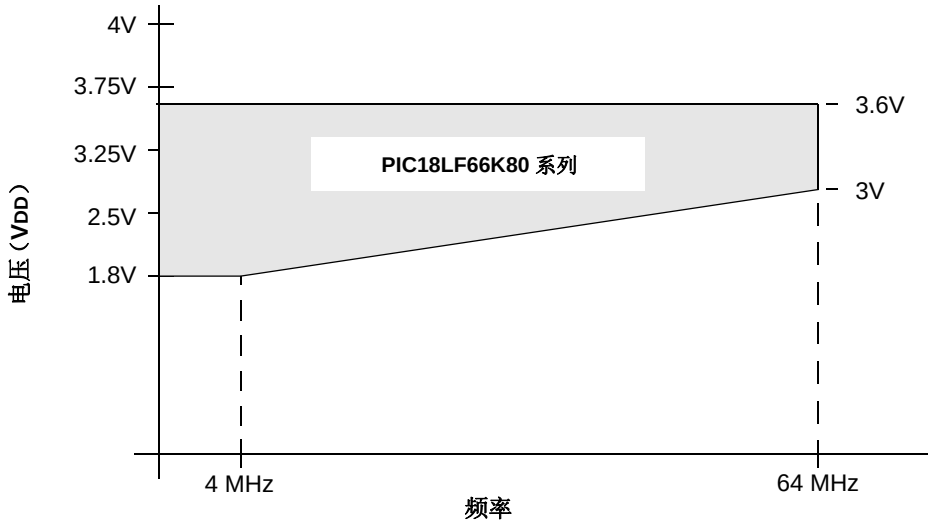
† **注：** 如果运行条件超过了上述 “ 绝对最大值 ”，即可能对器件造成永久性损坏。这仅是极限参数，我们不建议器件工作在极限值甚至超过上述极限值。器件长时间工作在极限条件下可能会影响其可靠性。

图 31-1：　　电压 — 频率关系图，使能稳压器 （工业级 / 扩展级）(1)



注　1： 对于 VDD 值为 1.8V 至 3V， FMAX = (VDD – 1.72)/0.02 MHz。

图 31-2：　　电压 — 频率关系图，禁止稳压器 （工业级 / 扩展级）(1,2)



注　1： 当禁止片上稳压器时， VDD 必须保持在 VDD ≤ 3.6V。

2： 对于 VDD 值为 1.8V 至 3V， FMAX = (VDD – 1.72)/0.02 MHz。

31.1 直流特性： 供电电压 PIC18F66K80 系列 （工业级 / 扩展级）

PIC18F66K80 系列 （工业级 / 扩展级）			标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）				
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
D001	VDD	供电电压	1.8 1.8	— —	3.6 5.5	V V	对于 LF 器件 对于 F 器件
D001C	AVDD	模拟供电电压	VDD – 0.3	—	VDD + 0.3	V	
D001D	AVSS	模拟地电位	VSS – 0.3	—	VSS + 0.3	V	
D002	VDR	RAM 数据保持电压 (1)	1.5	—	—	V	
D003	VPOR	确保内部上电复位信号的 VDD 启动电压	—	—	0.7	V	详情请参见第 5.3 节 “ 上电复位 （POR） ”
D004	SVDD	确保内部上电复位信号的 VDD 上升速率	0.05	—	—	V/ms	详情请参见第 5.3 节 “ 上电复位 （POR） ”
D005	BVDD	欠压复位电压 （高、中和低功耗模式） BORV<1:0> = 11 (2) BORV<1:0> = 10 BORV<1:0> = 01 BORV<1:0> = 00	 1.69 1.88 2.53 2.82	 1.8 2.0 2.7 3.0	 1.91 2.12 2.86 3.18	 V V V V	

注 1： 这是在不丢失 RAM 数据的前提下，休眠模式或器件复位期间 VDD 所能降到的最小电压值。

2： 在发生欠压复位之前，器件将正常工作，即使 VDD 可能低于 VDDMIN。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）				
参数编号	器件	典型值	最大值	单位	条件	
	掉电电流 （IPD）(1)					
	PIC18**LF**XXK80	8	400	nA	-40°C	VDD = 1.8V （**休眠**模式） 禁止稳压器
		13	500	nA	+25°C	
		35	750	nA	+60°C	
		218	980	nA	+85°C	
		3	6	µA	+125°C	
	PIC18**LF**XXK80	14	500	nA	-40°C	VDD = 3.3V （**休眠**模式） 禁止稳压器
		34	600	nA	+25°C	
		92	850	nA	+60°C	
		312	1250	nA	+85°C	
		4	8	µA	+125°C	
	PIC18**F**XXK80	200	700	nA	-40°C	VDD = 3.3V （**休眠**模式） 使能稳压器
		230	800	nA	+25°C	
		320	1050	nA	+60°C	
		510	1500	nA	+85°C	
		5	9	µA	+125°C	
	PIC18**F**XXK80	220	1000	nA	-40°C	VDD = 5V （**休眠**模式） 使能稳压器
		240	1000	nA	+25°C	
		340	1100	nA	+60°C	
		540	1580	nA	+85°C	
		5	10	µA	+125°C	

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD） (2,3)						
	PIC18**LF**XXK80	4	8	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 31 kHz （**RC_RUN** 模式， LF-INTOSC）
		4	8	µA	+25°C		
		4	8	µA	+60°C		
		5	9	µA	+85°C		
		9	12	µA	+125°C		
	PIC18**LF**XXK80	7	11	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		7	11	µA	+25°C		
		7	11	µA	+60°C		
		8	12	µA	+85°C		
		13	15	µA	+125°C		
	PIC18**F**XXK80	51	150	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		70	150	µA	+25°C		
		75	150	µA	+60°C		
		80	170	µA	+85°C		
		88	190	µA	+125°C		
	PIC18**F**XXK80	75	180	µA	-40°C	VDD = 5V(5) 使能稳压器	
		75	180	µA	+25°C		
		75	180	µA	+60°C		
		80	190	µA	+85°C		
		95	200	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	274	600	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 1 MHz （**RC_RUN** 模式， HF-INTOSC）
		274	600	µA	+25°C		
		274	600	µA	+60°C		
		280	650	µA	+85°C		
		290	700	µA	+125°C		
	PIC18**LF**XXK80	410	820	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		410	820	µA	+25°C		
		410	820	µA	+60°C		
		420	840	µA	+85°C		
		430	990	µA	+125°C		
	PIC18**F**XXK80	490	860	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		490	860	µA	+25°C		
		490	860	µA	+60°C		
		500	890	µA	+85°C		
		510	1060	µA	+125°C		
	PIC18**F**XXK80	490	910	µA	-40°C	VDD = 5V(5) 使能稳压器	
		490	910	µA	+25°C		
		490	910	µA	+60°C		
		500	970	µA	+85°C		
		510	1125	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	520	820	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 4 MHz （**RC_RUN** 模式， HF-INTOSC）
		520	820	µA	+25°C		
		520	820	µA	+60°C		
		530	880	µA	+85°C		
		540	1000	µA	+125°C		
	PIC18**LF**XXK80	941	1600	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		941	1600	µA	+25°C		
		941	1600	µA	+60°C		
		950	1610	µA	+85°C		
		960	1800	µA	+125°C		
	PIC18**F**XXK80	981	1640	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		981	1640	µA	+25°C		
		981	1640	µA	+60°C		
		990	1650	µA	+85°C		
		1000	1900	µA	+125°C		
	PIC18**F**XXK80	1	2.2	mA	-40°C	VDD = 5V(5) 使能稳压器	
		1	2.2	mA	+25°C		
		1	2.2	mA	+60°C		
		1	2.2	mA	+85°C		
		1	2.2	mA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	880	1600	nA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 31 kHz （**RC_IDLE** 模式， LF-INTOSC）
		880	1600	nA	+25°C		
		880	1600	nA	+60°C		
		1	2	µA	+85°C		
		5	10	µA	+125°C		
	PIC18**LF**XXK80	1.6	5	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		1.6	5	µA	+25°C		
		1.6	5	µA	+60°C		
		2	6	µA	+85°C		
		7	12	µA	+125°C		
	PIC18**F**XXK80	41	130	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		59	130	µA	+25°C		
		64	130	µA	+60°C		
		70	150	µA	+85°C		
		80	175	µA	+125°C		
	PIC18**F**XXK80	53	160	µA	-40°C	VDD = 5V(5) 使能稳压器	
		62	160	µA	+25°C		
		70	160	µA	+60°C		
		85	170	µA	+85°C		
		100	180	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	260	380	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 1 MHz （**RC_IDLE** 模式， HF-INTOSC）
		260	380	µA	+25°C		
		260	380	µA	+60°C		
		270	390	µA	+85°C		
		280	420	µA	+125°C		
	PIC18**LF**XXK80	400	500	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		400	500	µA	+25°C		
		400	500	µA	+60°C		
		410	520	µA	+85°C		
		420	580	µA	+125°C		
	PIC18**F**XXK80	430	560	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		430	560	µA	+25°C		
		430	560	µA	+60°C		
		450	580	µA	+85°C		
		480	620	µA	+125°C		
	PIC18**F**XXK80	450	620	µA	-40°C	VDD = 5V(5) 使能稳压器	
		450	620	µA	+25°C		
		450	620	µA	+60°C		
		470	640	µA	+85°C		
		500	680	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	330	480	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 4 MHz （**RC_IDLE** 模式， 内部 HF-INTOSC）
		330	480	µA	+25°C		
		330	480	µA	+60°C		
		340	500	µA	+85°C		
		350	540	µA	+125°C		
	PIC18**LF**XXK80	522	720	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		522	720	µA	+25°C		
		522	720	µA	+60°C		
		540	740	µA	+85°C		
		550	780	µA	+125°C		
	PIC18**F**XXK80	540	760	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		540	760	µA	+25°C		
		540	760	µA	+60°C		
		560	780	µA	+85°C		
		580	810	µA	+125°C		
	PIC18**F**XXK80	600	1250	µA	-40°C	VDD = 5V(5) 使能稳压器	
		600	1250	µA	+25°C		
		600	1250	µA	+60°C		
		610	1300	µA	+85°C		
		620	1340	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	90	260	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 1 MHz （**PRI_RUN** 模式， EC 振荡器）
		90	260	µA	+25°C		
		90	260	µA	+60°C		
		100	270	µA	+85°C		
		110	300	µA	+125°C		
	PIC18**LF**XXK80	163	540	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		163	540	µA	+25°C		
		163	540	µA	+60°C		
		170	560	µA	+85°C		
		180	600	µA	+125°C		
	PIC18**F**XXK80	201	560	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		217	560	µA	+25°C		
		224	560	µA	+60°C		
		228	580	µA	+85°C		
		236	620	µA	+125°C		
	PIC18**F**XXK80	240	740	µA	-40°C	VDD = 5V(5) 使能稳压器	
		240	740	µA	+25°C		
		240	740	µA	+60°C		
		250	840	µA	+85°C		
		260	940	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	270	600	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 4 MHz （**PRI_RUN** 模式， EC 振荡器）
		270	600	µA	+25°C		
		270	600	µA	+60°C		
		300	700	µA	+85°C		
		320	850	µA	+125°C		
	PIC18**LF**XXK80	540	1000	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		540	1000	µA	+25°C		
		540	1000	µA	+60°C		
		550	1100	µA	+85°C		
		560	1200	µA	+125°C		
	PIC18**F**XXK80	566	1020	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		585	1020	µA	+25°C		
		590	1020	µA	+60°C		
		595	1120	µA	+85°C		
		600	1220	µA	+125°C		
	PIC18**F**XXK80	630	2000	µA	-40°C	VDD = 5V(5) 使能稳压器	
		630	2000	µA	+25°C		
		630	2000	µA	+60°C		
		640	2000	µA	+85°C		
		650	2000	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	7	11	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	FOSC = 64 MHz （**PRI_RUN** 模式， EC 振荡器）
		7	11	µA	+25°C		
		7	11	µA	+60°C		
		7	11	µA	+85°C		
		7	11	µA	+125°C		
	PIC18**F**XXK80	7	11	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		7	11	µA	+60°C		
		7	11	µA	+25°C		
		7	11	µA	+85°C		
		7	11	µA	+125°C		
	PIC18**F**XXK80	8	12	µA	-40°C	VDD = 5V(5) 使能稳压器	
		8	12	µA	+60°C		
		8	12	µA	+25°C		
		8	12	µA	+85°C		
		8	12	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	2	5	µA	-40°C	VDD = 3.3V(4)	FOSC = 16 MHz （**PRI_RUN** 模式，带 PLL 的 4 MHz EC 振荡器）
		2	5	µA	+25°C		
		2	5	µA	+60°C		
		2	5	µA	+85°C		
		2	5	µA	+125°C		
	PIC18**F**XXK80	2	5	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		2	5	µA	+25°C		
		2	5	µA	+60°C		
		2	5	µA	+85°C		
		2	5	µA	+125°C		
	PIC18**F**XXK80	2.2	6	µA	-40°C	VDD = 5V(5) 使能稳压器	
		2.2	6	µA	+25°C		
		2.2	6	µA	+60°C		
		2.2	6	µA	+85°C		
		2.2	6	µA	+125°C		
	PIC18**LF**XXK80	7	11	µA	-40°C	VDD = 3.3V(4)	FOSC = 64 MHz （**PRI_RUN** 模式，带 PLL 的 16 MHz EC 振荡器）
		7	11	µA	+25°C		
		7	11	µA	+60°C		
		7	11	µA	+85°C		
		7	11	µA	+125°C		
	PIC18**F**XXK80	7	11	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		7	11	µA	+25°C		
		7	11	µA	+60°C		
		7	11	µA	+85°C		
		7	11	µA	+125°C		
	PIC18**F**XXK80	8	12	µA	-40°C	VDD = 5V(5) 使能稳压器	
		8	12	µA	+25°C		
		8	12	µA	+60°C		
		8	12	µA	+85°C		
		8	12	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	20	70	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 1 MHz （**PRI_IDLE** 模式， EC 振荡器）
		20	70	µA	+25°C		
		20	70	µA	+60°C		
		25	80	µA	+85°C		
		30	100	µA	+125°C		
	PIC18**LF**XXK80	37	120	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		37	120	µA	+25°C		
		37	120	µA	+60°C		
		40	130	µA	+85°C		
		45	150	µA	+125°C		
	PIC18**F**XXK80	85	140	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		100	140	µA	+25°C		
		105	140	µA	+60°C		
		110	150	µA	+85°C		
		120	170	µA	+125°C		
	PIC18**F**XXK80	110	225	µA	-40°C	VDD = 5V(5) 使能稳压器	
		110	225	µA	+25°C		
		110	225	µA	+60°C		
		120	230	µA	+85°C		
		130	250	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	75	160	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 4 MHz （**PRI_IDLE** 模式， EC 振荡器）
		75	160	µA	+25°C		
		75	160	µA	+60°C		
		76	170	µA	+85°C		
		82	180	µA	+125°C		
	PIC18**LF**XXK80	148	300	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		148	300	µA	+25°C		
		148	300	µA	+60°C		
		150	400	µA	+85°C		
		157	460	µA	+125°C		
	PIC18**F**XXK80	187	320	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		204	320	µA	+25°C		
		212	320	µA	+60°C		
		218	420	µA	+85°C		
		230	480	µA	+125°C		
	PIC18**F**XXK80	230	500	µA	-40°C	VDD = 5V(5) 使能稳压器	
		230	500	µA	+25°C		
		230	500	µA	+60°C		
		240	600	µA	+85°C		
		250	700	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	2.3	4	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	FOSC = 64 MHz （**PRI_IDLE** 模式， EC 振荡器）
		2.3	4	µA	+25°C		
		2.3	4	µA	+60°C		
		2.3	5	µA	+85°C		
		2.3	5	µA	+125°C		
	PIC18**F**XXK80	2.3	4	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		2.3	4	µA	+25°C		
		2.3	4	µA	+60°C		
		2.3	5	µA	+85°C		
		2.3	5	µA	+125°C		
	PIC18**F**XXK80	2.5	5	µA	-40°C	VDD = 5V(5) 使能稳压器	
		2.5	5	µA	+25°C		
		2.5	5	µA	+60°C		
		2.5	6	µA	+85°C		
		2.5	6	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	2	8	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 32 kHz(3) （**SEC_RUN** 模式， **SOSCSEL** = 01）
		5	10	µA	+25°C		
		6	15	µA	+60°C		
		8	20	µA	+85°C		
		13	30	µA	+125°C		
	PIC18**LF**XXK80	3	15	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		16	22	µA	+25°C		
		17	28	µA	+60°C		
		19	39	µA	+85°C		
		25	60	µA	+125°C		
	PIC18**F**XXK80	70	170	µA	-40°C	VDD = 3.3V(5) 使能稳压器	
		70	170	µA	+25°C		
		70	170	µA	+60°C		
		75	180	µA	+85°C		
		90	190	µA	+125°C		
	PIC18**F**XXK80	75	180	µA	-40°C	VDD = 5V(5) 使能稳压器	
		75	180	µA	+25°C		
		75	180	µA	+60°C		
		80	190	µA	+85°C		
		95	200	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流 （IDD）续 (2,3)						
	PIC18**LF**XXK80	1.4	4	µA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 32 kHz(3) （**SEC_IDLE** 模式， **SOSCSEL** = 01）
		2.4	6	µA	+25°C		
		3.6	10	µA	+60°C		
		4.6	12	µA	+85°C		
		9.0	20	µA	+125°C		
	PIC18**LF**XXK80	2	5	µA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		10	18	µA	+25°C		
		11	22	µA	+60°C		
		13	30	µA	+85°C		
		17	40	µA	+125°C		
	PIC18**F**XXK80	55	150	µA	+25°C	VDD = 3.3V(5) 使能稳压器	
		55	150	µA	+60°C		
		55	150	µA	+85°C		
		60	160	µA	+125°C		
		75	170	µA	+25°C		
	PIC18**F**XXK80	53	160	µA	-40°C	VDD = 5V(5) 使能稳压器	
		62	160	µA	+25°C		
		70	160	µA	+60°C		
		85	170	µA	+85°C		
		100	180	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
D022 （ΔIWDT）	**模块增加电流 （ΔIWDT、 ΔIOSCB 和 ΔIAD）**						
	看门狗定时器						
	PIC18**LF**XXK80	0.4	1	µA	-40°C	VDD = 1.8V 禁止稳压器	
		0.4	1	µA	+25°C		
		0.5	1	µA	+60°C		
		0.5	1	µA	+85°C		
		0.5	2	µA	+125°C		
	PIC18**LF**XXK80	0.6	2	µA	-40°C	VDD = 3.3V 禁止稳压器	
		0.6	2	µA	+25°C		
		0.7	2	µA	+60°C		
		0.7	2	µA	+85°C		
		1	3	µA	+125°C		
	PIC18**F**XXK80	0.6	2	µA	-40°C	VDD = 3.3V 使能稳压器	
		0.6	2	µA	+25°C		
		0.7	2	µA	+60°C		
		0.7	2	µA	+85°C		
		1	3	µA	+125°C		
	PIC18**F**XXK80	0.8	2	µA	-40°C	VDD = 5V 使能稳压器	
		0.8	2	µA	+25°C		
		0.9	2	µA	+60°C		
		0.9	2	µA	+85°C		
		1.5	4	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
D022A （ΔIBOR）	欠压复位						
	PIC18**LF**XXK80	4.6	19	µA	-40°C	VDD = 3.3V 禁止稳压器	高功耗 BOR
		4.5	20	µA	+25°C		
		4.7	20	µA	+60°C		
		4.7	20	µA	+85°C		
		4.7	20	µA	+125°C		
	PIC18**F**XXK80	4.6	19	µA	-40°C	VDD = 3.3V 使能稳压器	高功耗 BOR
		4.6	20	µA	+25°C		
		4.7	20	µA	+60°C		
		4.7	20	µA	+85°C		
		4.7	20	µA	+125°C		
	PIC18**F**XXK80	4.2	20	µA	-40°C	VDD = 5V 使能稳压器	高功耗 BOR
		4.3	20	µA	+25°C		
		4.4	20	µA	+60°C		
		4.4	20	µA	+85°C		
		4.4	20	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流
PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	**器件**	**典型值**	**最大值**	**单位**	**条件**		
D022B	**高 / 低压检测**						
（ΔIHLVD）	PIC18**LF**XXK80	3.8	9	µA	-40°C	VDD = 1.8V 禁止稳压器	
		4.2	9	µA	+25°C		
		4.3	10	µA	+60°C		
		4.3	10	µA	+85°C		
		4.3	10	µA	+125°C		
	PIC18**LF**XXK80	4.5	11	µA	-40°C	VDD = 3.3V 禁止稳压器	
		4.8	12	µA	+25°C		
		4.8	12	µA	+60°C		
		4.8	12	µA	+85°C		
		4.8	12	µA	+125°C		
	PIC18**F**XXK80	3.8	11	µA	-40°C	VDD = 3.3V 使能稳压器	
		4.2	12	µA	+25°C		
		4.3	12	µA	+60°C		
		4.3	12	µA	+85°C		
		4.3	12	µA	+125°C		
	PIC18**F**XXK80	4.9	13	µA	-40°C	VDD = 5V 使能稳压器	
		4.9	13	µA	+25°C		
		4.9	13	µA	+60°C		
		4.9	13	µA	+85°C		
		4.9	13	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
MCLR = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， RETEN （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0。

31.2 直流特性： 掉电电流和供电电流 PIC18F66K80 系列 （工业级 / 扩展级）（续）

PIC18F66K80 系列 （工业级 / 扩展级）		标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）					
参数编号	器件	典型值	最大值	单位	条件		
D026 （ΔIAD）	**A/D 转换器**						
	PIC18**LF**XXK80	0.4	1	µA	-40°C	VDD = 1.8V 禁止稳压器	A/D 开启，但不进行转换
		0.4	1	µA	+25°C		
		0.4	1	µA	+60°C		
		0.4	1	µA	+85°C		
		0.6	1.5	µA	+125°C		
	PIC18**LF**XXK80	0.5	1	µA	-40°C	VDD = 3.3V 禁止稳压器	A/D 开启，但不进行转换
		0.5	1	µA	+25°C		
		0.5	1	µA	+60°C		
		0.5	1	µA	+85°C		
		0.8	2	µA	+125°C		
	PIC18**F**XXK80	0.5	1	µA	-40°C	VDD = 3.3V 使能稳压器	A/D 开启，但不进行转换
		0.5	1	µA	+25°C		
		0.5	1	µA	+60°C		
		0.5	1	µA	+85°C		
		0.8	2	µA	+125°C		
	PIC18**F**XXK80	1	2	µA	-40°C	VDD = 5V 使能稳压器	A/D 开启，但不进行转换
		1	2	µA	+25°C		
		1	2	µA	+60°C		
		1	2	µA	+85°C		
		1	3	µA	+125°C		

图注： 阴影行是为了增强表的可读性。

注 1： 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 VDD 或 VSS，禁止所有会带来新增电流的功能部件 （如 WDT、 SOSC 振荡器和 BOR 等）时测得的。

2： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。
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 引脚均为三态，上拉至 VDD ；
$\overline{MCLR}$ = VDD ； 根据具体应用使能 / 禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 至 +70°C。扩展级温度晶振的成本则高很多。

4： 对于 LF 器件， $\overline{RETEN}$ （CONFIG1L<0>） = 1。

5： 对于 F 器件， SRETEN （WDTCON<4>） = 1 且 $\overline{RETEN}$ （CONFIG1L<0>） = 0。

31.3 直流特性： PIC18F66K80 系列 （工业级）

直流特性			标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）			
参数编号	**符号**	**特性**	**最小值**	**最大值**	**单位**	**条件**
	VIL	**输入低电压**				
		所有 I/O 端口：				
D031		施密特触发器缓冲器	VSS	0.2 VDD	V	1.8V ≤ Vdd ≤ 5.5V
D031A		RC3 和 RC4	VSS	0.3 VDD	V	使能 I2C™
D031B			VSS	0.8	V	使能 SMBus
D032		$\overline{MCLR}$	VSS	0.2 VDD	V	
D033		OSC1	VSS	0.2 VDD	V	LP、 XT 和 HS 模式
D033A		OSC1	VSS	0.2 VDD	V	EC 模式
D034		SOSCI	VSS	0.3 VDD	V	
	VIH	**输入高电压**				
		所有 I/O 端口：				
D041		施密特触发器缓冲器	0.8	VDD	V	1.8V ≤ Vdd ≤ 5.5V
D041A		RC3 和 RC4	0.7 VDD	VDD	V	使能 I2C
D041B			2.1	VDD	V	使能 SMBus
D042		$\overline{MCLR}$	0.8 VDD	VDD	V	
D043		OSC1	0.9 VDD	VDD	V	RC 模式
D043A		OSC1	0.7 VDD	VDD	V	HS 模式
D044		SOSCI	0.7 VDD	VDD	V	
	IIL	**输入泄漏电流 (1)**				
D060		I/O 端口	±50	±500	nA	VSS ≤ VPIN ≤ VDD， 引脚处于高阻态
D061		$\overline{MCLR}$	—	±500	nA	Vss ≤ VPIN ≤ VDD
D063		OSC1	—	1	µA	Vss ≤ VPIN ≤ VDD
	IPU	**弱上拉电流**				
D070		弱上拉电流	50	400	µA	VDD = 5.5V， VPIN = VSS

注 1： 负电流定义为引脚的拉电流。

31.3 直流特性： PIC18F66K80 系列 （工业级）（续）

直流特性			标准工作条件 （除非另外声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）			
参数编号	符号	特性	最小值	最大值	单位	条件
	VOL	**输出低电压**				
D080		I/O 端口：				
		PORTA、PORTB 和 PORTC	—	0.6	V	IOL = 8.5 mA， VDD = 5.5V，-40°C 至 +125°C
		PORTD、PORTE、PORTF 和 PORTG	—	0.6	V	IOL = 3.5 mA， VDD = 5.5V，-40°C 至 +125°C
D083		OSC2/CLKO （EC 模式）	—	0.6	V	IOL = 1.6 mA， VDD = 5.5V，-40°C 至 +125°C
	VOH	**输出高电压 (1)**				
D090		I/O 端口：			V	
		PORTA、PORTB 和 PORTC	VDD – 0.7	—	V	IOH = -3 mA， VDD = 5.5V，-40°C 至 +125°C
		PORTD、PORTE、PORTF 和 PORTG	VDD – 0.7	—	V	IOH = -2 mA， VDD = 5.5V，-40°C 至 +125°C
D092		OSC2/CLKO （INTOSC 和 EC 模式）	VDD – 0.7	—	V	IOH = -1 mA， VDD = 5.5V，-40°C 至 +125°C
		输出引脚上的容性负载规范				
D100(4)	COSC2	OSC2 引脚	—	20	pF	当外部时钟用于驱动 OSC1 时处于 HS 模式下
D101	CIO	所有 I/O 引脚和 OSC2	—	50	pF	满足交流时序规范
D102	CB	SCLx 和 SDAx	—	400	pF	I2C™ 规范

注 1： 负电流定义为引脚的拉电流。

31.4 直流特性：CTMU 电流源规范

直流特性			标准工作条件：1.8V 至 5.5V 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）				
参数编号	符号	特性	最小值	典型值 (1)	最大值	单位	条件
	IOUT1	CTMU 电流源，基本范围	—	550	—	nA	CTMUICON<1:0> = 01
	IOUT2	CTMU 电流源，10x 范围	—	5.5	—	µA	CTMUICON<1:0> = 10
	IOUT3	CTMU 电流源，100x 范围	—	55	—	µA	CTMUICON<1:0> = 11

注 1： 电流微调范围的中点为标称值 （CTMUICON<7:2> = 000000）。

表 31-1： 存储器编程要求

直流特性			标准工作条件 工作温度 -40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）				
参数编号	**符号**	**特性**	**最小值**	**典型值†**	**最大值**	**单位**	**条件**
		内部程序存储器编程规范 (1)					
D110	VPP	$\overline{MCLR}$/VPP/RE5 引脚上的电压	VDD + 1.5	—	10	V	**（注 3，注 4）**
D113	IDDP	编程时的供电电流	—	—	10	mA	
		数据 EEPROM 存储器					**（注 2）**
D120	ED	字节耐擦写能力	100K	1000K	—	E/W	-40°C 至 +125°C
D121	VDRW	读 / 写操作时的 VDD	1.8	—	5.5	V	使用 EECON 读 / 写 PIC18FXXKXX 器件
			1.8	—	3.6	V	使用 EECON 读 / 写 PIC18LFXXKXX 器件
D122	TDEW	擦除 / 写周期时间	—	4	—	ms	
D123	TRETD	特性保持时间	20	—	—	年	假设没有违反其他规范
D124	TREF	刷新前的总擦除 / 写次数 (2)	1M	10M	—	E/W	-40°C 至 +125°C
		闪存程序存储器					
D130	EP	单元耐擦写能力	1K	10K	—	E/W	-40°C 至 +125°C
D131	VPR	读操作时的 VDD	1.8	—	5.5	V	PIC18FXXKXX 器件
			1.8	—	3.6	V	PIC18LFXXKXX 器件
D132B	VPEW	自定时擦除或写操作的电压 VDD	1.8	—	5.5	V	PIC18FXXKXX 器件
D133A	TIW	自定时写周期时间	—	2	—	ms	
D134	TRETD	特性保持时间	20	—	—	年	假设没有违反其他规范
D135	IDDP	编程时的供电电流	—	—	10	mA	
D140	TWE	每个擦除周期的写入次数	—	—	1		对于每个物理地址

† 除非另外声明，否则“典型值”栏中的数据均为3.3V和25°C条件下的值。这些参数仅供设计参考，未经测试。

注 1： 这些规范用于通过使用表写指令对片上程序存储器进行编程。

2： 关于数据 EEPROM 耐擦写能力的详细讨论，请参见**第 8.8 节 “ 使用数据 EEPROM”**。

3： 仅当禁止单电源编程时才需要。

4： MPLAB ICD2 不支持可变 VPP 输出。当使用 ICD2 进行编程或调试时，用于限制 ICD2 VPP 电压的电路必须置于 ICD2 和目标系统之间。

表 31-2： 比较器规范

工作条件：1.8V ≤ VDD ≤ 5.5V， -40°C ≤ TA ≤ +125°C							
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**备注**
D300	VIOFF	输入失调电压	—	±5.0	40	mV	
D301	VICM	输入共模电压	—	—	AVDD – 1.5	V	
D302	CMRR	共模抑制比	55	—	—	dB	
D303	TRESP	响应时间 (1)	—	150	400	ns	
D304	TMC2OV	比较器模式改变到输出有效的时间 *	—	—	10	µs	

注 1： 响应时间是在比较器的一个输入端电压为 (AVDD – 1.5)/2，而另一个输入端从 VSS 跳变到 VDD 时测得的。

表 31-3： 参考电压规范

工作条件：1.8V ≤ VDD ≤ 5.5V， -40°C ≤ TA ≤ +125°C							
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**备注**
D310	VRES	分辨率	VDD/24	—	VDD/32	LSb	
D311	VRAA	绝对精度	—	—	1/2	LSb	
D312	VRUR	单位电阻值 （R）	—	2k	—	Ω	
D313	TSET	稳定时间 (1)	—	—	10	µs	

注 1： 稳定时间是在 CVRR = 1 并且 CVR<3:0> 从 0000 跳变到 1111 时测得的。

表 31-4： 内部稳压器规范

工作条件：-40°C ≤ TA ≤ +125°C							
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**备注**
	VRGOUT	稳压器输出电压	—	3.3	—	V	
	CEFC	外部滤波电容值	4.7	10	—	µF	电容必须具有较低的等效串联电阻 （< 5Ω）

31.5 交流 （时序）特性

31.5.1 时序参数符号体系

时序参数符号采用以下格式之一进行创建：

1. TppS2ppS
2. TppS
3. TCC:ST （仅用于 I²C 规范）
4. Ts （仅用于 I²C 规范）

T			
	F	频率	T 时间

小写字母 （pp）及其含义：

pp			
	cc	CCP1	osc OSC1
	ck	CLKO	rd $\overline{RD}$
	cs	$\overline{CS}$	rw $\overline{RD}$ 或 $\overline{WR}$
	di	SDI	sc SCK
	do	SDO	ss $\overline{SS}$
	dt	数据输入	t0 T0CKI
	io	I/O 端口	t1 T1CKI
	mc	$\overline{MCLR}$	wr $\overline{WR}$

大写字母及其含义：

S			
	F	下降	P 周期
	H	高	R 上升
	I	无效 （高阻）	V 有效
	L	低	Z 高阻
仅用于 I²C			
	AA	输出访问	High 高
	BUF	总线空闲	Low 低

TCC:ST （仅用于 I²C 规范）

CC			
	HD	保持	SU 建立
ST			
	DAT	数据输入保持	STO 停止条件
	STA	启动条件	

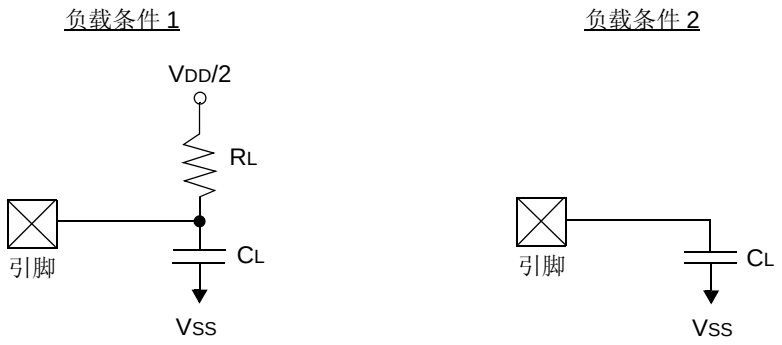
31.5.2 时序条件

表 31-5 中指定的温度和电压适用于所有的时序规范（除非另外声明）。图 31-3 规定了时序规范的负载条件。

表 31-5： 温度和电压规范 —— 交流

交流特性	**标准工作条件 （除非另外声明）**	
	工作温度	-40°C ≤ TA ≤ +85°C （工业级） -40°C ≤ TA ≤ +125°C （扩展级）
	工作电压 VDD 范围如**第 31.1 节**和**第 31.3 节**中所述。	

图 31-3： 器件时序规范的负载条件



RL = 464Ω

CL = 50 pF 对于除 OSC2/CLKO/RA6 外的所有引脚包括作为端口的 D 和 E 输出

CL = 15 pF 对于 OSC2/CLKO/RA6

31.5.3 时序图和规范

图 31-4： 外部时钟时序

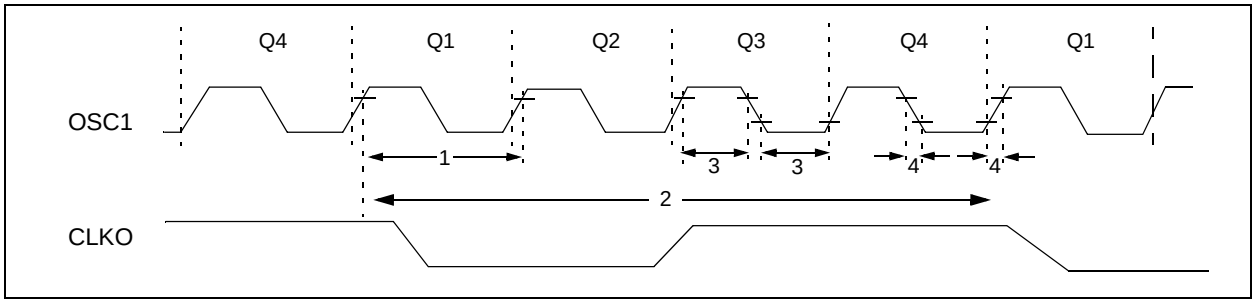


表 31-6： 外部时钟时序要求

参数编号	符号	特性	最小值	最大值	单位	条件
1A	FOSC	外部 CLKIN 频率 (1)	DC	64	MHz	EC 和 ECIO 振荡器模式
		振荡器频率 (1)	DC	4	MHz	RC 振荡器模式
			0.1	4	MHz	XT 振荡器模式
			4	16	MHz	HS 振荡器模式
			4	16	MHz	HS + PLL 振荡器模式
			5	33	kHz	LP 振荡器模式
1	TOSC	外部 CLKIN 周期 (1)	15.6	—	ns	EC 和 ECIO 振荡器模式
		振荡周期 (1)	250	—	ns	RC 振荡器模式
			250	10,000	ns	XT 振荡器模式
			40	250	ns	HS 振荡器模式
			62.5	250	ns	HS + PLL 振荡器模式
			5	200	µs	LP 振荡器模式
2	TCY	指令周期 (1)	62.5	—	ns	TCY = 4/FOSC
3	TOSL，TOSH	外部时钟输入（OSC1）的高电平或低电平时间	30	—	ns	XT 振荡器模式
			2.5	—	µs	LP 振荡器模式
			10	—	ns	HS 振荡器模式
4	TOSR，TOSF	外部时钟输入（OSC1）的上升或下降时间	—	20	ns	XT 振荡器模式
			—	50	ns	LP 振荡器模式
			—	7.5	ns	HS 振荡器模式

注 1： 对于除 PLL 外的所有配置，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/CLKIN 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）。

表 31-7： PLL 时钟时序规范 （VDD = 1.8V 至 5.5V）

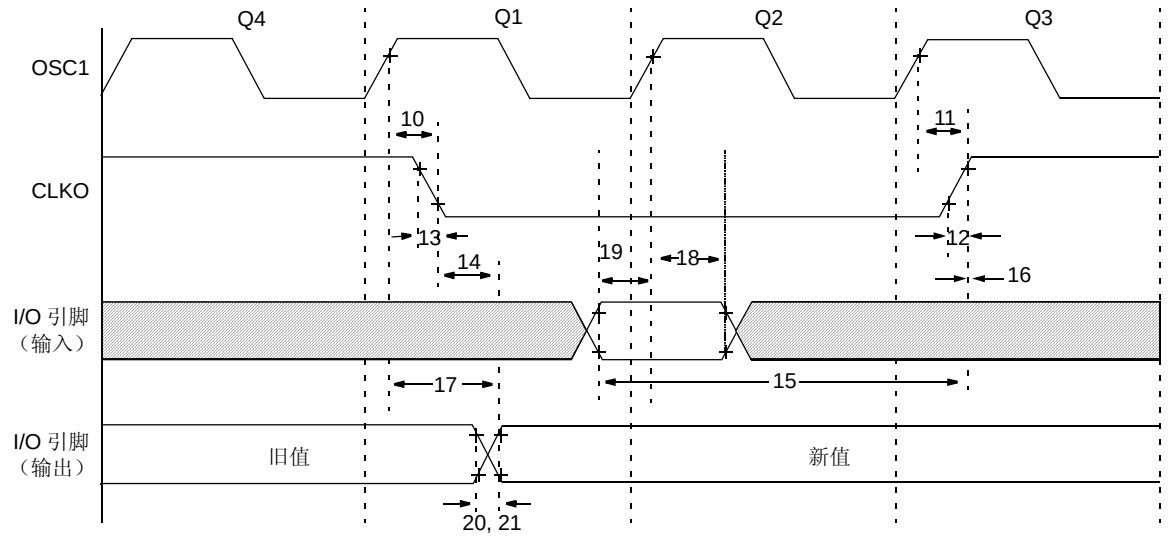
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
F10	FOSC	振荡器频率范围	4	—	5	MHz	VDD = 1.8-5.5V
			4	—	16	MHz	VDD = 3.0-5.5V，-40°C 至 +125°C
F11	FSYS	片上 VCO 系统频率	16	—	20	MHz	VDD = 1.8-5.5V
			16	—	64	MHz	VDD = 3.0-5.5V，-40°C 至 +125°C
F12	t_{rc}	PLL 起振时间 （锁定时间）	—	—	2	ms	
F13	ΔCLK	CLKOUT 稳定性 （抗抖动性）	-2	—	+2	%	

表 31-8： 内部 RC 精度 （INTOSC）

PIC18F66K80 系列		**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		**最小值**	**典型值**	**最大值**	**单位**	**条件**	
OA1	**在频率为 16 MHz、8 MHz、4 MHz、2 MHz、1 MHz、500 kHz 和 250 kHz 时的 HFINTOSC/MFINTOSC 精度 (1)**						
		-2	—	+2	%	+25°C	VDD = 3-5.5V
		-5	—	+5	%	-40°C至+85°C	VDD = 1.8-5.5V
		-10	—	+10	%	-40°C至+125°C	VDD = 1.8-5.5V
OA2	**在频率为 31 kHz 时的 LFINTOSC 精度**						
		-15	—	15	%	-40°C 至 +125°C	VDD = 1.8-5.5V

注 1： 频率校准温度为 25°C。OSCTUNE 寄存器可用于补偿温度漂移。

图 31-5：　　CLKO 和 I/O 时序



注：　负载条件请参见图 31-3。

表 31-9：　　CLKO 和 I/O 时序要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
10	TosH2ckL	OSC1 ↑ 到 CLKO ↓ 的时间	—	75	200	ns	（注 1）
11	TosH2ckH	OSC1 ↑ 到 CLKO ↑ 的时间	—	75	200	ns	（注 1）
12	TckR	CLKO 上升时间	—	15	30	ns	（注 1）
13	TckF	CLKO 下降时间	—	15	30	ns	（注 1）
14	TckL2ioV	CLKO ↓ 到端口输出有效的时间	—	—	0.5 TCY + 20	ns	
15	TioV2ckH	CLKO ↑ 之前端口输入有效的时间	0.25 TCY + 25	—	—	ns	
16	TckH2ioI	CLKO ↑ 之后端口输入保持的时间	0	—	—	ns	
17	TosH2ioV	OSC1 ↑ （Q1 周期）到端口输出有效的时间	—	50	150	ns	
18	TosH2ioI	OSC1 ↑ （Q2 周期）到端口输入无效的时间 （I/O 输入保持时间）	100	—	—	ns	
19	TioV2osH	端口输入有效到 OSC1 ↑ 的时间 （I/O 输入建立时间）	0	—	—	ns	
20	TioR	端口输出上升时间	—	10	25	ns	
21	TioF	端口输出下降时间	—	10	25	ns	
22†	TINP	INTx 引脚高电平或低电平时间	20	—	—	ns	
23†	TRBP	RB<7:4> 电平变化 INTx 高电平或低电平时间	TCY	—	—	ns	

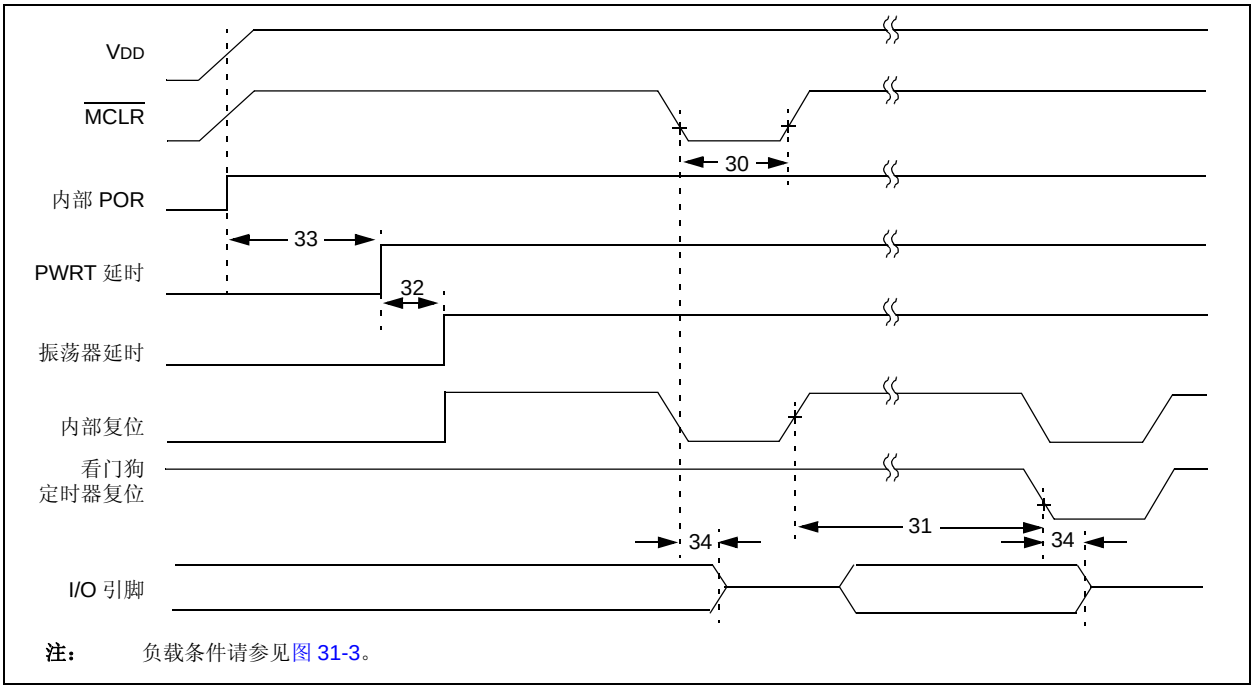
†　这些参数是与任何内部时钟边沿无关的异步事件。

注　1：　测量是在 EC 模式下进行的，其中 CLKO 输出为 4 x TOSC。

表 31-10： CLKO 和 I/O 时序要求

参数编号	符号	特性	最小值	典型值	最大值	单位
150	TadV2alL	地址输出有效到 ALE ↓ 的时间（地址建立时间）	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ 到地址输出无效的时间（地址保持时间）	5	—	—	ns
155	TalL2oeL	ALE ↓ 到 $\overline{OE}$ ↓ 的时间	10	0.125 TCY	—	ns
160	TadZ2oeL	AD 高阻态到 $\overline{OE}$ ↓ 的时间 （总线释放 $\overline{OE}$）	0	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ 到驱动 AD 的时间	0.125 TCY – 5	—	—	ns
162	TadV2oeH	$\overline{OE}$ ↑之前LS数据有效的时间（数据建立时间）	20	—	—	ns
163	ToeH2adl	$\overline{OE}$ ↑ 到数据输入无效的时间（数据保持时间）	0	—	—	ns
164	TalH2alL	ALE 脉冲宽度	—	0.25 TCY	—	ns
165	ToeL2oeH	$\overline{OE}$ 脉冲宽度	0.5 TCY – 5	0.5 TCY	—	ns
166	TalH2alH	ALE ↑ 到 ALE ↑ 的时间 （周期时间）	—	TCY	—	ns
167	Tacc	地址有效到数据有效的时间	0.75 TCY – 25	—	—	ns
168	Toe	$\overline{OE}$ ↓ 到数据有效的时间		—	0.5 TCY – 25	ns
169	TalL2oeH	ALE ↓ 到 $\overline{OE}$ ↑ 的时间	0.625 TCY – 10	—	0.625 TCY + 10	ns
171	TalH2csL	芯片使能有效到 ALE ↓ 的时间	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	AD 有效到芯片使能有效的时间	—	—	10	ns

图 31-6： 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： 负载条件请参见图 31-3。

图 31-7：　　欠压复位时序

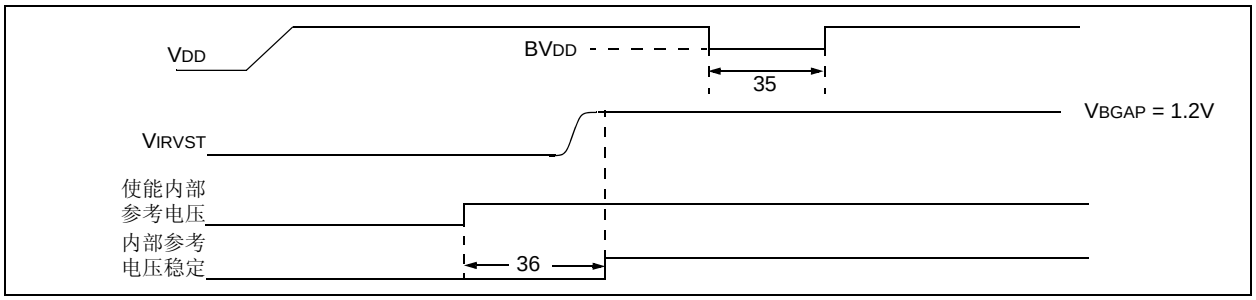


表 31-11：　　复位、看门狗定时器、振荡器起振定时器、上电延时定时器和欠压复位要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
30	TmcL	$\overline{\text{MCLR}}$ 脉冲宽度（低电平）	2	—	—	µs	
31	TWDT	看门狗定时器超时周期（无后分频器）	—	4.00	—	ms	
32	TOST	振荡器起振定时器周期	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 周期
33	TPWRT	上电延时定时器周期	—	65.5	—	ms	
34	TIOZ	自 $\overline{\text{MCLR}}$ 低电平或看门狗定时器复位起 I/O 处于高阻态的时间	—	2	—	µs	
35	TBOR	欠压复位脉冲宽度	200	—	—	µs	VDD ≤ BVDD（见 D005）
36	TIVRST	内部参考电压稳定时间	—	25	—	µs	
37	THLVD	高 / 低压检测脉冲宽度	200	—	—	µs	VDD ≤ VHLVD
38	TCSD	CPU 启动时间	5	—	10	µs	
39	TIOBST	INTOSC 稳定时间	—	1	—	µs	

图 31-8：　　高 / 低压检测特性

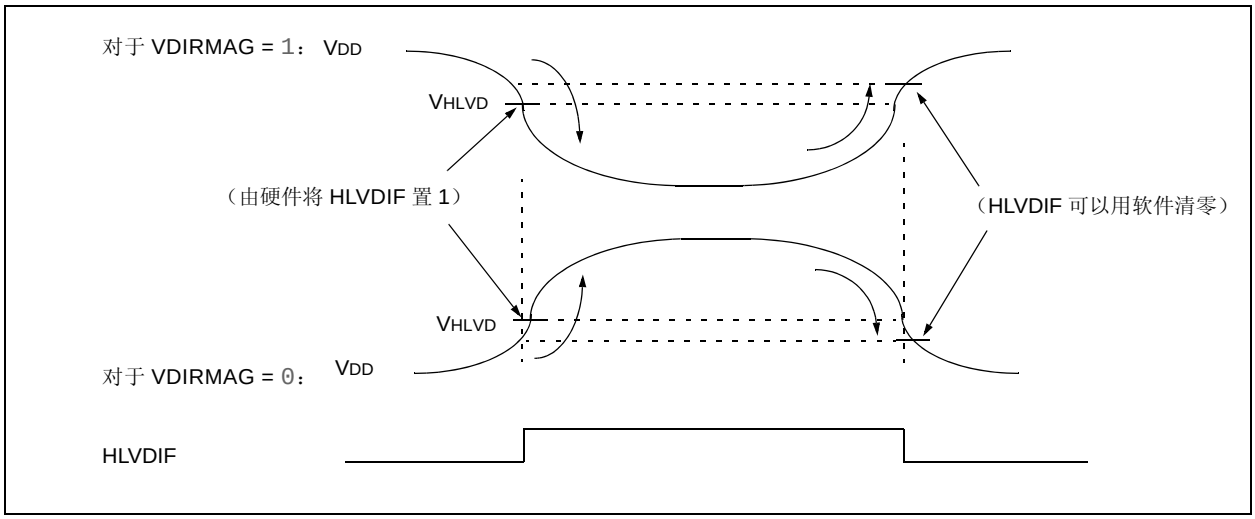
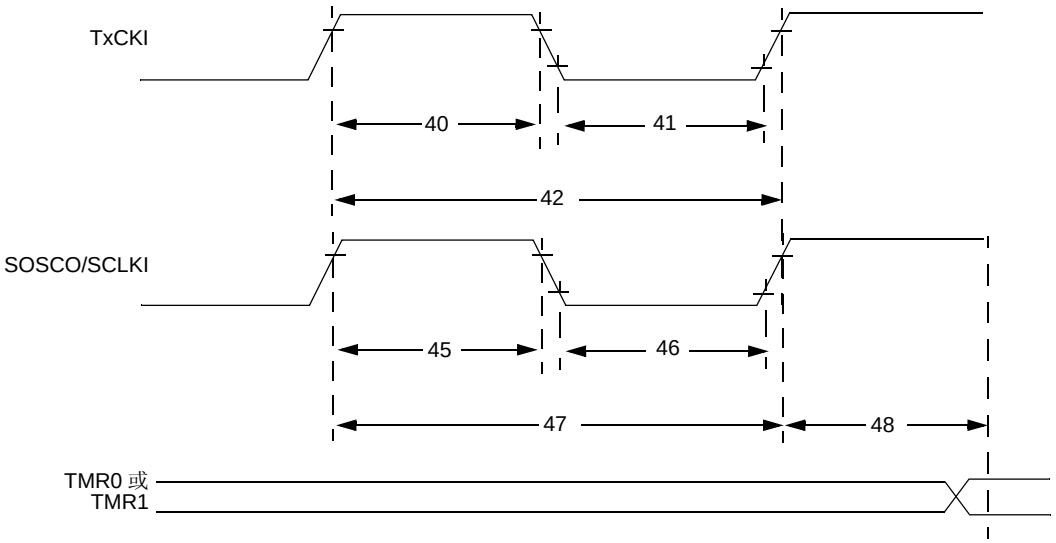


表 31-12：　　高 / 低压检测特性

标准工作条件（除非另外声明） 工作温度　　-40°C ≤ TA ≤ +85°C（工业级） 　　-40°C ≤ TA ≤ +125°C（扩展级）								
参数编号	**符号**	**特性**		**最小值**	**典型值**	**最大值**	**单位**	**条件**
D420		VDD由高转变为低时的 HLVD 电压	HLVDL<3:0> = 0000	1.80	1.85	1.90	V	
			HLVDL<3:0> = 0001	2.03	2.08	2.13	V	
			HLVDL<3:0> = 0010	2.24	2.29	2.35	V	
			HLVDL<3:0> = 0011	2.40	2.46	2.53	V	
			HLVDL<3:0> = 0100	2.50	2.56	2.62	V	
			HLVDL<3:0> = 0101	2.70	2.77	2.84	V	
			HLVDL<3:0> = 0110	2.82	2.89	2.97	V	
			HLVDL<3:0> = 0111	2.95	3.02	3.10	V	
			HLVDL<3:0> = 1000	3.24	3.32	3.41	V	
			HLVDL<3:0> = 1001	3.42	3.50	3.59	V	
			HLVDL<3:0> = 1010	3.61	3.70	3.79	V	
			HLVDL<3:0> = 1011	3.82	3.91	4.10	V	
			HLVDL<3:0> = 1100	4.06	4.16	4.26	V	
			HLVDL<3:0> = 1101	4.33	4.44	4.55	V	
			HLVDL<3:0> = 1110	4.64	4.75	4.87	V	

图 31-9：　　TIMER0 和 TIMER1 外部时钟时序

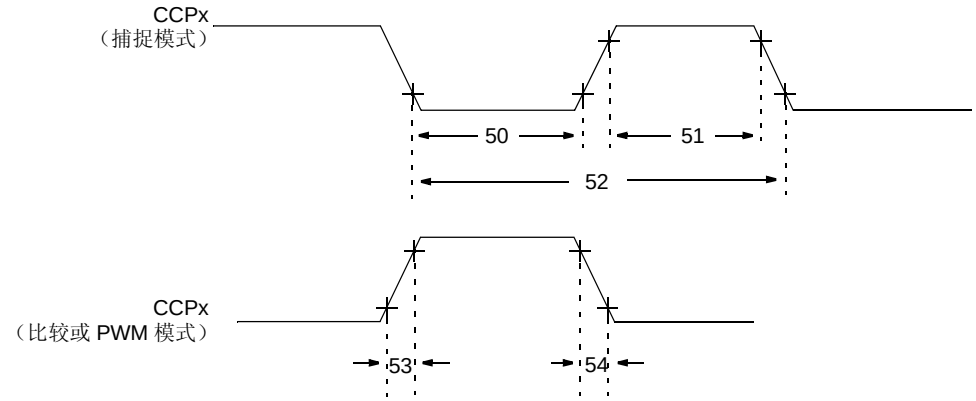


注：　负载条件请参见图 31-3。

表 31-13：　　TIMER0 和 TIMER1 外部时钟要求

参数编号	符号	特性		最小值	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	无预分频器	TCY + 10	—	ns	
			带预分频器	取如下二者中较大值： 20 ns 或 (TCY + 40)/N	—	ns	N = 预分频值（1, 2, 4,..., 256）
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	10	—	ns	
			异步	30	—	ns	
46	TT1L	T1CKI 低电平时间	同步，无预分频器	0.5 TCY + 5	—	ns	
			同步，带预分频器	10	—	ns	
			异步	30	—	ns	
47	TT1P	T1CKI 输入周期	同步	取如下二者中较大值： 20 ns 或 (TCY + 40)/N	—	ns	N = 预分频值（1、2、4 和 8）
			异步	60	—	ns	
	FT1	T1CKI 振荡器输入频率范围		DC	50	kHz	
48	TCKE2TMRI	从外部 T1CKI 时钟边沿到定时器递增的延时		2 TOSC	7 TOSC	—	

图 31-10：　　捕捉 / 比较 /PWM 时序 （ECCP1 和 ECCP2 模块）

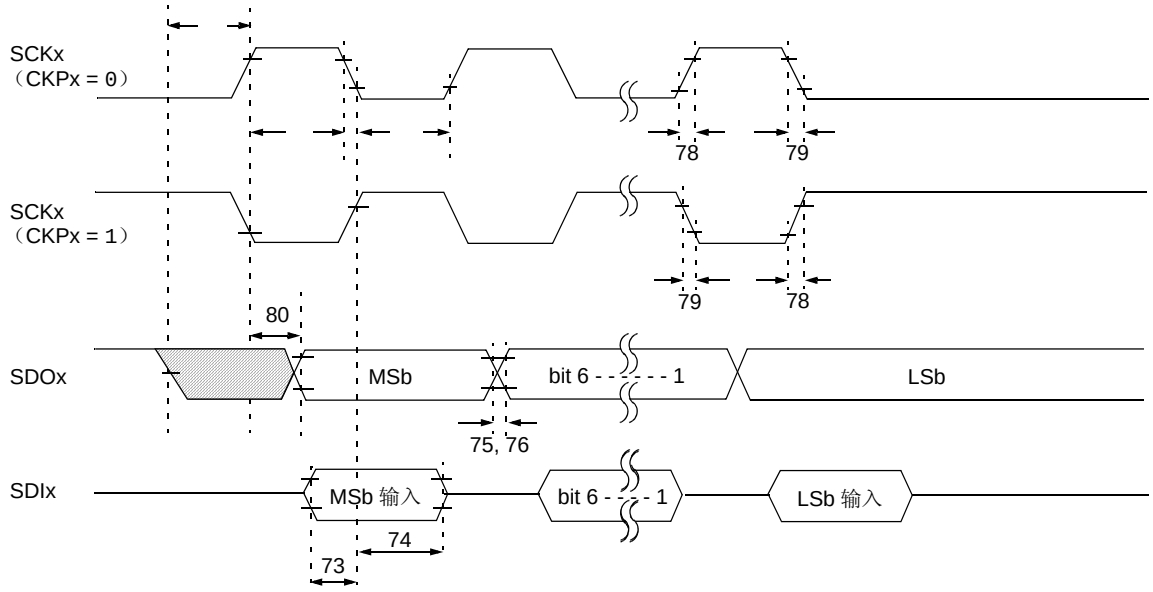


注：　　负载条件请参见图 31-3。

表 31-14：　　捕捉 / 比较 /PWM 要求 （ECCP1 和 ECCP2 模块）

参数编号	符号	特性		最小值	最大值	单位	条件
50	TccL	CCPx输入低电平时间	无预分频器	0.5 TCY + 20	—	ns	
			带预分频器	10	—	ns	
51	TccH	CCPx输入高电平时间	无预分频器	0.5 TCY + 20	—	ns	
			带预分频器	10	—	ns	
52	TccP	CCPx 输入周期		$\frac{3\ T_{CY} + 40}{N}$	—	ns	N = 预分频值（1、 4 或 16）
53	TccR	CCPx 输出上升时间		—	25	ns	
54	TccF	CCPx 输出下降时间		—	25	ns	

图 31-11： SPI 主模式时序示例 （CKE = 0）

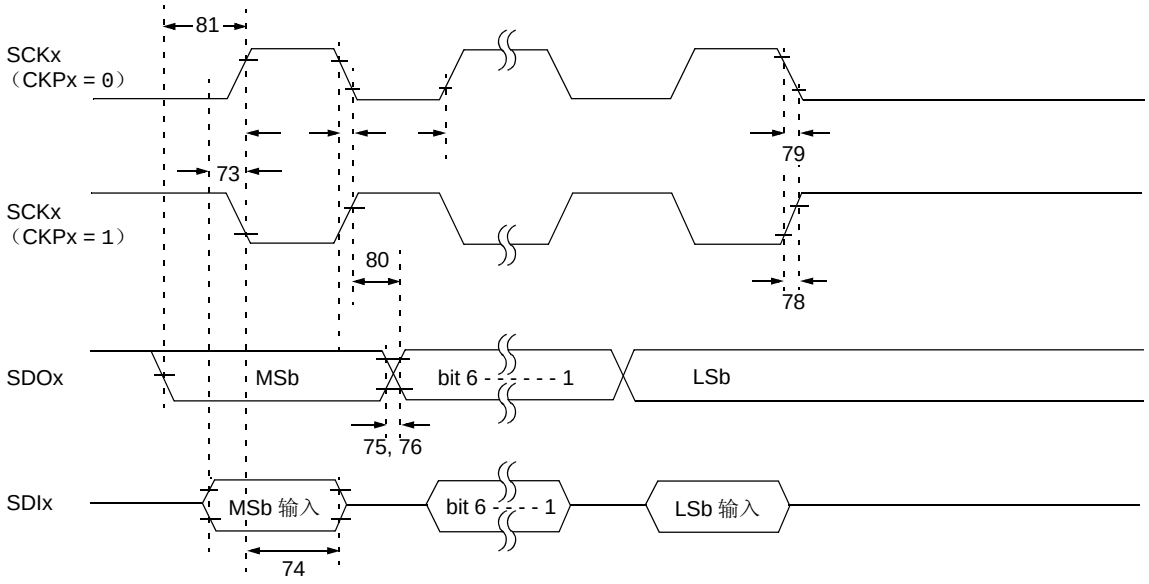


注： 负载条件请参见图 31-3。

表 31-15： SPI 模式要求示例 （主模式， CKE = 0）

参数编号	符号	特性	最小值	最大值	单位	条件
73	TDIV2SCH，TDIV2SCL	SDIx 数据输入到 SCKx 边沿的建立时间	20	—	ns	
73A	TB2B	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间	1.5 TCY + 40	—	ns	
74	TSCH2DIL，TSCL2DIL	SDIx 数据输入到 SCKx 边沿的保持时间	40	—	ns	
75	TDOR	SDOx 数据输出上升时间	—	25	ns	
76	TDOF	SDOx 数据输出下降时间	—	25	ns	
78	TSCR	SCKx 输出上升时间 （主模式）	—	25	ns	
79	TSCF	SCKx 输出下降时间 （主模式）	—	25	ns	
80	TSCH2DOV，TSCL2DOV	SCKx 边沿之后 SDOx 数据输出有效的时间	—	50	ns	

图 31-12： SPI 主模式时序示例 （CKE = 1）

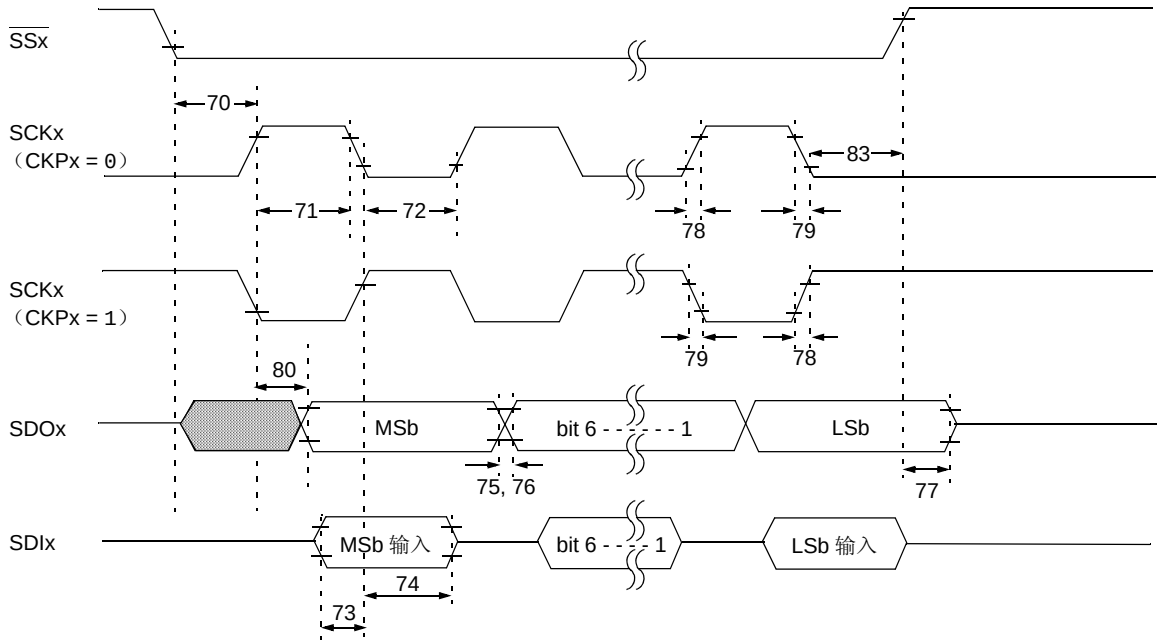


注： 负载条件请参见图 31-3。

表 31-16： SPI 模式要求示例 （主模式， CKE = 1）

参数编号	符号	特性	最小值	最大值	单位	条件
73	TDIV2SCH，TDIV2SCL	SDIx 数据输入到 SCKx 边沿的建立时间	20	—	ns	
73A	TB2B	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间	1.5 TCY + 40	—	ns	
74	TSCH2DIL，TSCL2DIL	SDIx 数据输入到 SCKx 边沿的保持时间	40	—	ns	
75	TDOR	SDOx 数据输出上升时间	—	25	ns	
76	TDOF	SDOx 数据输出下降时间	—	25	ns	
78	TSCR	SCKx 输出上升时间（主模式）	—	25	ns	
79	TSCF	SCKx 输出下降时间（主模式）	—	25	ns	
80	TSCH2DOV，TSCL2DOV	SCKx 边沿之后 SDOx 数据输出有效的时间	—	50	ns	
81	TDOV2SCH，TDOV2SCL	SDOx 数据输出建立到出现 SCKx 边沿的时间	TCY	—	ns	

图 31-13：　　SPI 从模式时序示例 （CKE = 0）



注：　负载条件请参见图 31-3。

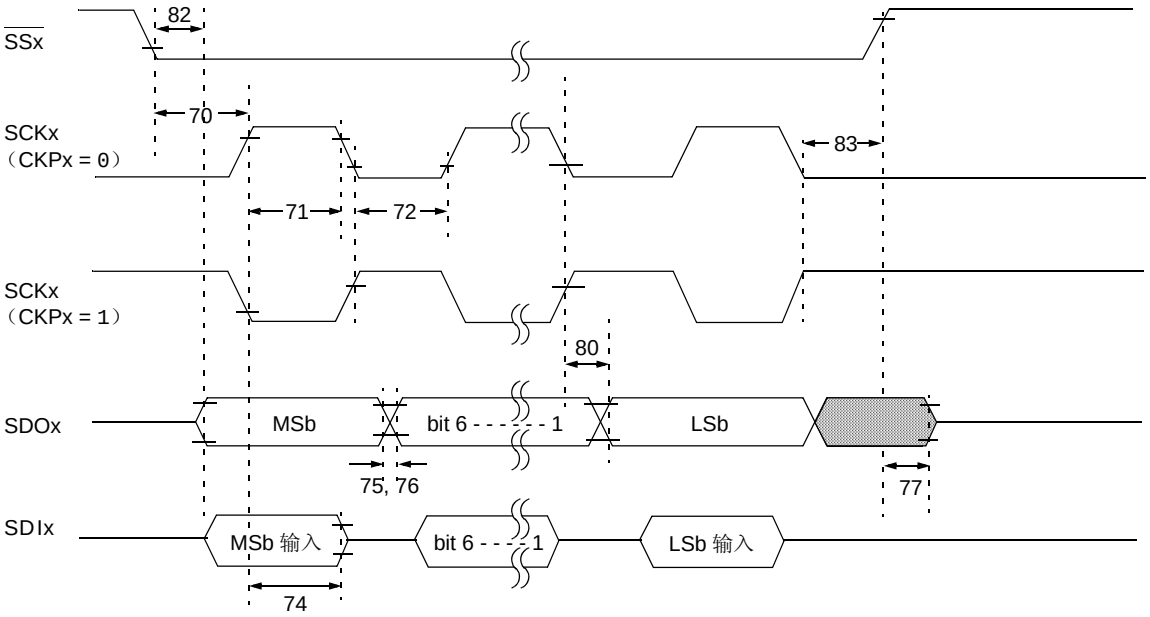
表 31-17：　　SPI 模式要求示例 （从模式时序， CKE = 0）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH, TssL2scL	$\overline{SSx}$ ↓到 SCKx ↓或 SCKx ↑输入的时间		3 TCY	—	ns	
70A	TssL2WB	$\overline{SSx}$ 到写入 SSPBUF 的时间		3 TCY	—	ns	
71	TscH	SCKx 输入高电平时间（从模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCKx 输入低电平时间（从模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73	TdiV2scH, TdiV2scL	SDIx 数据输入到 SCKx 边沿的建立时间		20	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY + 40	—	ns	（注 2）
74	TscH2diL, TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		40	—	ns	
75	TdoR	SDOx 数据输出上升时间		—	25	ns	
76	TdoF	SDOx 数据输出下降时间		—	25	ns	
77	TssH2doZ	$\overline{SSx}$ ↑到 SDOx 输出高阻态的时间		10	50	ns	
78	TscR	SCKx 输出上升时间 （主模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主模式）		—	25	ns	
80	TscH2doV, TscL2doV	SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
83	TscH2ssH, TscL2ssH	SCKx 边沿之后出现 $\overline{SSx}$ ↑的时间		1.5 TCY + 40	—	ns	

注　1：　要求使用参数 73A。

2：　仅当使用参数 71A 和 72A 时。

图 31-14：　　SPI 从模式时序示例 （CKE = 1）



注：　负载条件请参见图 31-3。

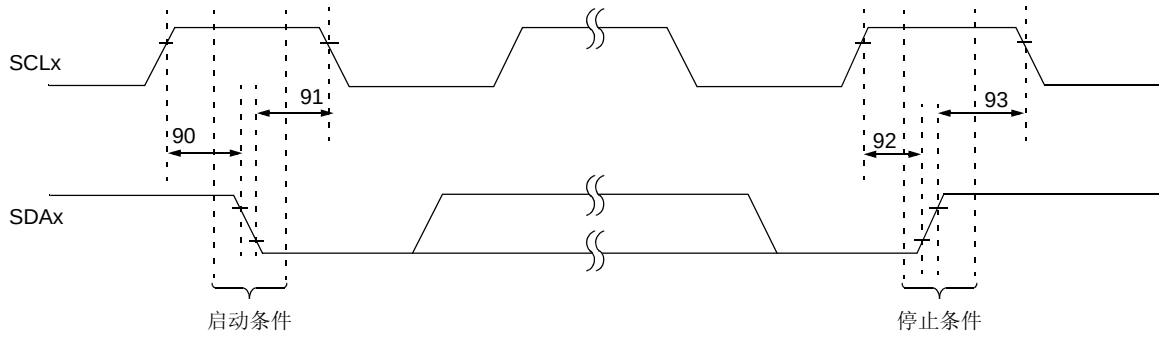
表 31-18：　　SPI 从模式要求示例 （CKE = 1）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	$\overline{SSx}$ ↓ 到 SCKx ↓ 或 SCKx ↑ 输入的时间		3 Tcy	—	ns	
70A	TssL2WB	$\overline{SSx}$ 到写入 SSPBUF 的时间		3 Tcy	—	ns	
71	TscH	SCKx 输入高电平时间（从模式）	连续	1.25 Tcy + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TscL	SCKx 输入低电平时间（从模式）	连续	1.25 Tcy + 30	—	ns	
72A			单字节	40	—	ns	（注 1）
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 Tcy + 40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		40	—	ns	
75	TdoR	SDOx 数据输出上升时间		—	25	ns	
76	TdoF	SDOx 数据输出下降时间		—	25	ns	
77	TssH2doZ	$\overline{SSx}$↑ 到 SDOx 输出高阻态的时间		10	50	ns	
78	TscR	SCKx 输出上升时间 （主模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主模式）		—	25	ns	
80	TscH2doV，TscL2doV	SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
82	TssL2doV	$\overline{SSx}$ ↓ 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
83	TscH2ssH，TscL2ssH	SCKx 边沿之后出现 $\overline{SSx}$ ↑ 的时间		1.5 Tcy + 40	—	ns	

注　1： 要求使用参数 73A。

2： 仅当使用参数 71A 和 72A 时。

图 31-15：　　I²C™ 总线启动位 / 停止位时序

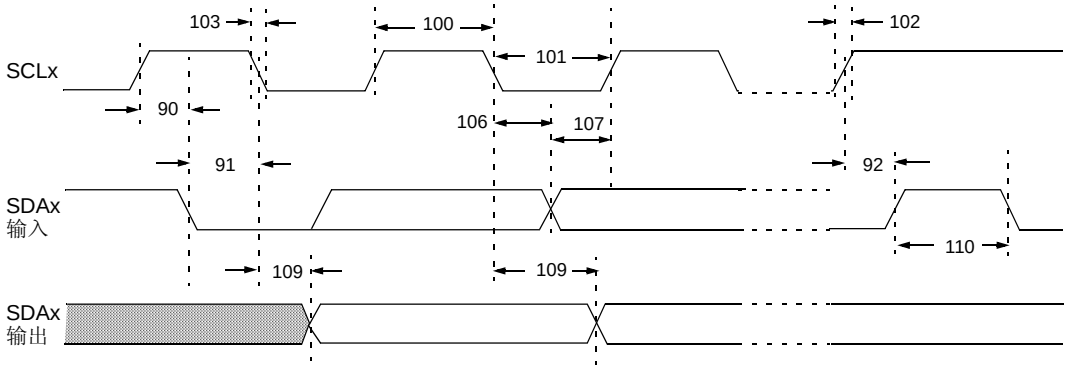


注：　负载条件请参见图 31-3。

表 31-19：　　I²C™ 总线启动位 / 停止位要求 （从模式）

参数编号	符号	特性		最小值	最大值	单位	条件
90	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	—		
91	THD:STA	启动条件保持时间	100 kHz 模式	4000	—	ns	这个周期后产生第一个时钟脉冲
			400 kHz 模式	600	—		
92	TSU:STO	停止条件建立时间	100 kHz 模式	4700	—	ns	
			400 kHz 模式	600	—		
93	THD:STO	停止条件保持时间	100 kHz 模式	4000	—	ns	
			400 kHz 模式	600	—		

图 31-16：　　I²C™ 总线数据时序



注：　负载条件请参见图 31-3。

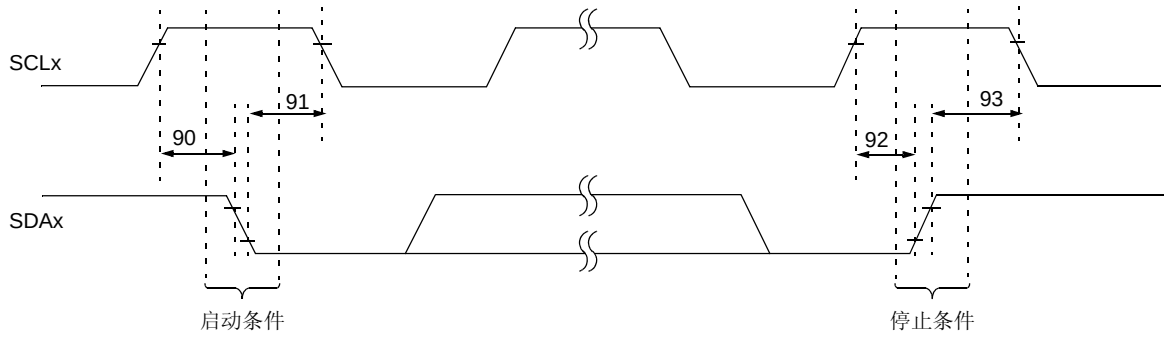
表 31-20：　　I²C™ 总线数据要求 （从模式）

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	4.0	—	µs	
			400 kHz 模式	0.6	—	µs	
			MSSP 模块	1.5 TCY	—		
101	TLOW	时钟低电平时间	100 kHz 模式	4.7	—	µs	
			400 kHz 模式	1.3	—	µs	
			MSSP 模块	1.5 TCY	—		
102	TR	SDAx 和 SCLx 上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值规定在 10 至 400 pF 之间
103	TF	SDAx 和 SCLx 下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值规定在 10 至 400 pF 之间
90	TSU:STA	启动条件建立时间	100 kHz 模式	4.7	—	µs	仅与重复启动条件相关
			400 kHz 模式	0.6	—	µs	
91	THD:STA	启动条件保持时间	100 kHz 模式	4.0	—	µs	这个周期后产生第一个时钟脉冲
			400 kHz 模式	0.6	—	µs	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	µs	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	**（注 2）**
			400 kHz 模式	100	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	4.7	—	µs	
			400 kHz 模式	0.6	—	µs	
109	TAA	从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	
110	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	
D102	CB	总线容性负载		—	400	pF	

注　1：　为避免产生意外的启动或停止条件，作为发送器的器件必须提供这个内部最小延时以补偿 SCLx 下降沿的未定义区域（最小值 300 ns）。

2：　快速模式的 I²C™ 总线器件也可在标准模式的 I²C 总线系统中使用，但必须满足 TSU:DAT ≥ 250 ns 的要求。如果快速模式器件没有延长 SCLx 信号的低电平周期，则必然满足此条件。如果该器件延长了 SCLx 信号的低电平周期，则在 SCLx 线被释放前，它必须将下一个数据位输出到 SDAx 线。根据标准模式 I²C 总线规范，TR max. + TSU:DAT = 1000 + 250 = 1250 ns。

图 31-17：　　MSSP I²C™ 总线启动位 / 停止位时序波形



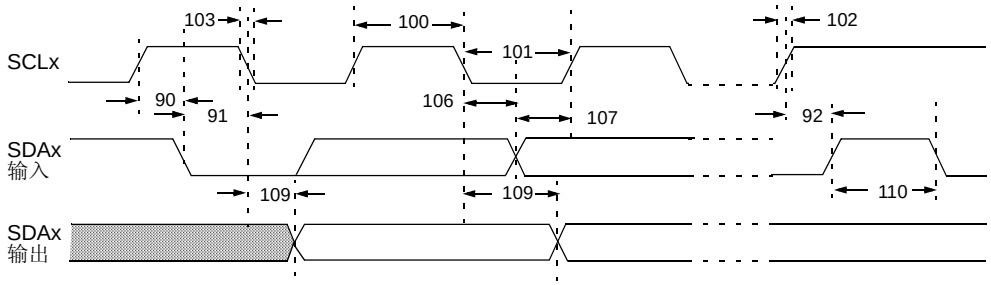
注：　负载条件请参见图 31-3。

表 31-21：　　MSSP I²C™ 总线启动位 / 停止位要求

参数编号	符号	特性		最小值	最大值	单位	条件
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ns	仅与重复启动条件相关
			400 kHz 模式	2(Tosc)(BRG + 1)	—		
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—		
91	THD:STA	启动条件保持时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ns	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(Tosc)(BRG + 1)	—		
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—		
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ns	
			400 kHz 模式	2(Tosc)(BRG + 1)	—		
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—		
93	THD:STO	停止条件保持时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ns	
			400 kHz 模式	2(Tosc)(BRG + 1)	—		
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—		

注　1：　对于所有 I²C™ 引脚，最大引脚电容均为 10 pF。

图 31-18：　　MSSP I²C™ 总线数据时序



注：　负载条件请参见图 31-3。

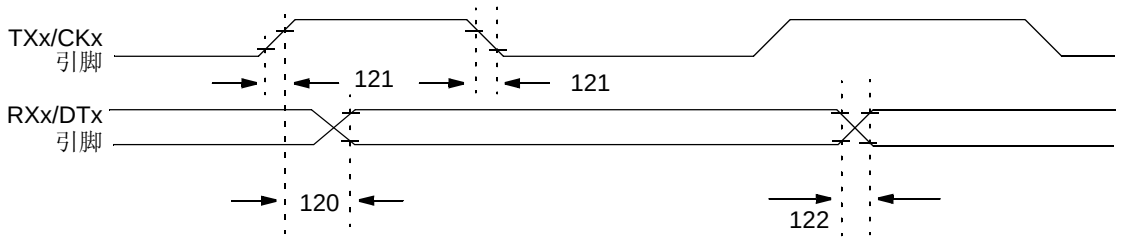
表 31-22： MSSP I²C™ 总线数据要求

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	—	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	—	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	—	
101	TLOW	时钟低电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	—	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	—	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	—	
102	TR	SDAx 和 SCLx 上升时间	100 kHz 模式	—	1000	ns	CB 值规定在 10 至 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	300	ns	
103	TF	SDAx 和 SCLx 下降时间	100 kHz 模式	—	300	ns	CB 值规定在 10 至 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	100	ns	
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	—	仅与重复启动条件相关
			400 kHz 模式	2(TOSC)(BRG + 1)	—	—	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	—	
91	THD:STA	启动条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	—	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1)	—	—	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	—	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	—	
			400 kHz 模式	0	0.9	µs	
			1 MHz 模式 (1)	—	µs	ns	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	**（注 2）**
			400 kHz 模式	100	—	ns	
			1 MHz 模式 (1)	—	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	—	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	—	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	—	
109	TAA	从时钟有效到输出有效的时间	100 kHz 模式	—	3500	ns	
			400 kHz 模式	—	1000	ns	
			1 MHz 模式 (1)	—	—	ns	
110	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	
			1 MHz 模式 (1)	—	—	µs	
D102	CB	总线容性负载		—	400	pF	

注 1： 对于所有 I²C™ 引脚，最大引脚电容均为 10 pF。

2： 快速模式的 I²C 总线器件也可在标准模式的 I²C 总线系统中使用，但必须满足参数 107 ≥ 250 ns 的要求。如果快速模式器件没有延长 SCLx 信号的低电平周期，则必然满足此条件。如果该器件延长了 SCLx 信号的低电平周期，则在 SCLx 线被释放前，它必须将下一个数据位输出到 SDAx 线。在 100 kHz 模式下，参数 102 + 参数 107 = 1000 + 250 = 1250 ns。

图 31-19：　　EUSART 同步发送 （主 / 从）时序

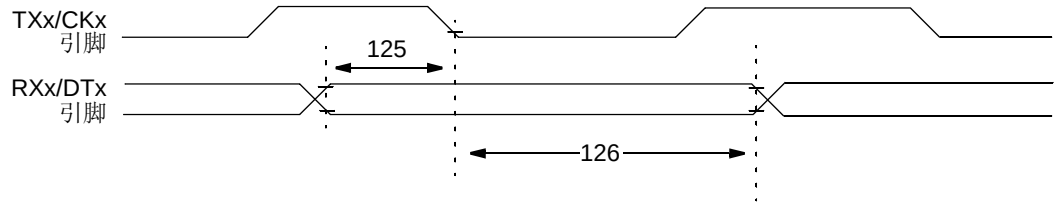


注：　负载条件请参见图 31-3。

表 31-23：　　EUSART/AUSART 同步发送要求

参数编号	符号	特性	最小值	最大值	单位	条件
120	TCKH2DTV	同步发送 （主 / 从） 时钟高电平到数据输出有效的时间	—	40	ns	
121	TCKRF	时钟输出上升时间和下降时间 （主模式）	—	20	ns	
122	TDTRF	数据输出上升时间和下降时间	—	20	ns	

图 31-20：　　EUSART/AUSART 同步接收 （主 / 从）时序



注：　负载条件请参见图 31-3。

表 31-24：　　EUSART/AUSART 同步接收要求

参数编号	符号	特性	最小值	最大值	单位	条件
125	TDTV2CKL	同步接收 （主 / 从） CKx↓之前数据的保持时间（DTx保持时间）	10	—	ns	
126	TCKL2DTL	CKx↓之后数据的保持时间（DTx保持时间）	15	—	ns	

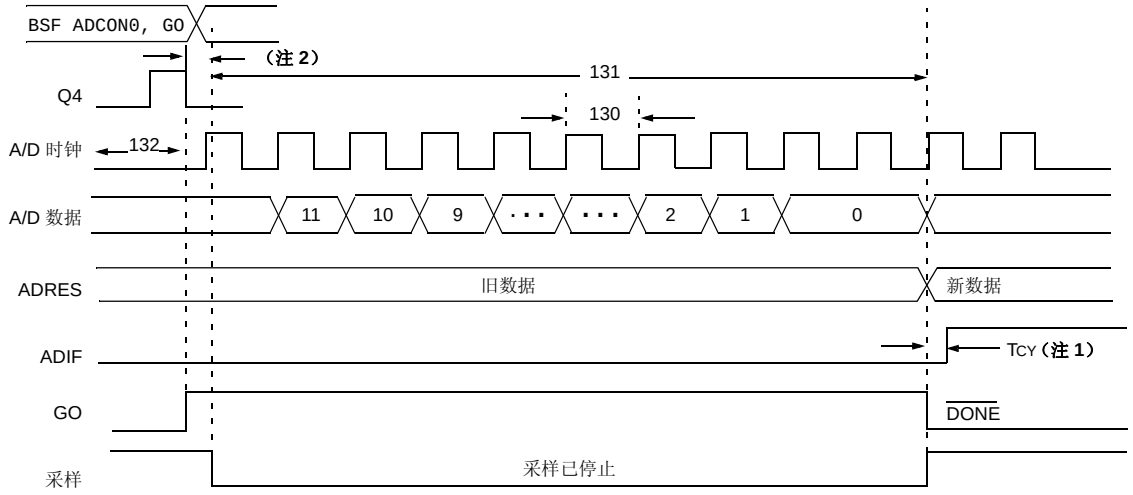
表 31-25： A/D 转换器特性：PIC18F66K80 系列（工业级 / 扩展级）

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
A01	NR	分辨率	—	—	12	位	ΔVREF ≥ 3.0V
A03	EIL	积分线性误差	—	<±1	±2.0	LSb	VDD = 3.0V （ΔVREF ≥ 3.0V）
			—	—	±2.0	LSb	VDD = 5.0V
A04	EDL	微分线性误差	—	<±1	+1.5/-1.0	LSb	VDD = 3.0V （ΔVREF ≥ 3.0V）
			—	—	+1.5/-1.0	LSb	VDD = 5.0V
A06	EOFF	失调误差	—	<±1	±5	LSb	VDD = 3.0V （ΔVREF ≥ 3.0V）
			—	—	±3	LSb	VDD = 5.0V
A07	EGN	增益误差	—	<±1	±1.25	LSb	VDD = 3.0V （ΔVREF ≥ 3.0V）
			—	—	±2.00	LSb	VDD = 5.0V
A10	—	单调性	保证 (1)			—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	参考电压范围（VREFH – VREFL）	3	—	VDD – VSS	V	适用于 12 位分辨率
A21	VREFH	参考电压高电压	AVSS + 3.0V	—	AVDD + 0.3V	V	适用于 12 位分辨率
A22	VREFL	参考电压低电压	AVSS – 0.3V	—	AVDD – 3.0V	V	适用于 12 位分辨率
A25	VAIN	模拟输入电压	VREFL	—	VREFH	V	
A28	AVDD	模拟供电电压	VDD – 0.3	—	VDD + 0.3	V	
A29	AVSS	模拟供电电压	VSS – 0.3	—	VSS + 0.3	V	
A30	ZAIN	模拟信号源的推荐阻抗	—	—	2.5	kΩ	
A50	IREF	VREF 输入电流 (2)	— —	— —	5 150	μA μA	在采集 VAIN 期间。 在 A/D 转换期间。

注 1： A/D 转换结果不会因输入电压的增加而减小，并且不会丢失编码。

2： VREFH 电流来自选择作为 VREFH 源的 RA3/AN3/VREF+ 引脚或 VDD。VREFL 电流来自选择作为 VREFL 源的 RA2/AN2/VREF-/CVREF 引脚或 VSS。

图 31-21： A/D 转换时序



注 1： 如果选择 RC 作为 A/D 转换的时钟源，在 A/D 时钟启动前要加上一个 TCY 时间，用以执行 SLEEP 指令。

2： 这是最小的 RC 延时 （典型值为 100 ns），在这段时间保持电容与模拟输入端断开。

表 31-26： A/D 转换要求

参数编号	符号	特性	最小值	最大值	单位	条件
130	TAD	A/D 时钟周期	0.8	12.5(1)	µs	基于 TOSC， VREF ≥ 3.0V
			1.4	25(1)	µs	VDD = 3.0V ； 基于 TOSC，整个 VREF 范围
			—	**1**	µs	A/D RC 模式
			—	3	µs	VDD = 3.0V ； A/D RC 模式
131	TCNV	转换时间 （不包括采集时间）(2)	14	15	TAD	
132	TACQ	采集时间 (3)	1.4	—	µs	-40°C 至 +125°C
135	TSWC	转换 → 采样的切换时间	—	（注 4）		
TBD	TDIS	电容放电时间	0.2	—	µs	-40°C 至 +125°C

注 1： A/D 时钟周期取决于器件频率和 TAD 时钟分频比。

2： ADRES 寄存器可在下一个 TCY 周期被读取。

3： 转换完成后当电压满量程变化时 （VDD 至 VSS 或 VSS 至 VDD），保持电容采集一个 “ 新 ” 输入电压所需的时间。在输入通道上的信号源阻抗（*R*S）为 50Ω。

4： 在器件时钟的下一个周期。

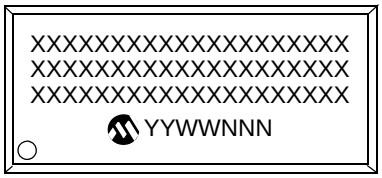
32.0 封装信息

32.1 封装标识信息

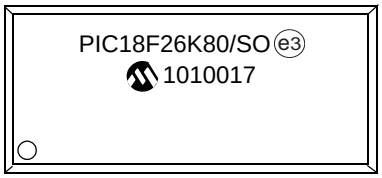
28 引脚 QFN

示例

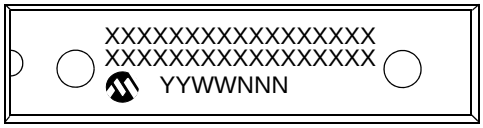
28 引脚 SOIC



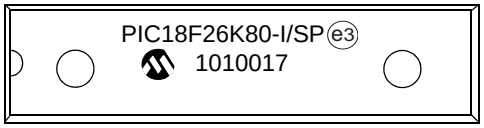
示例



28 引脚 SPDIP



示例



28 引脚 SSOP

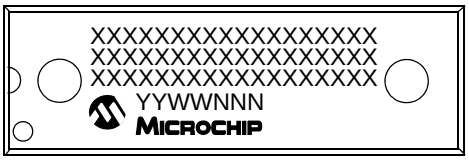
示例

图注：	XX...X	客户指定的信息
	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	以字母数字排序的追踪代码
	e3	雾锡 （Matte Tin， Sn）的 JEDEC 无铅标志
	*	本封装为无铅封装。 JEDEC 无铅标志 （e3） 标示于此种封装的外包装上。

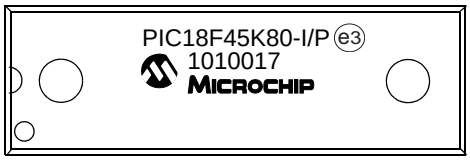
注： Microchip 元器件编号如果无法在同一行内完整标注，将换行标出，因此会限制表示客户信息的字符数。

32.1 封装标识信息 （续）

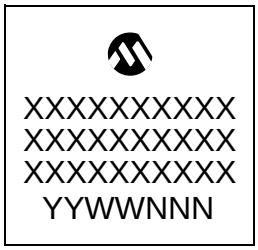
40 引脚 PDIP



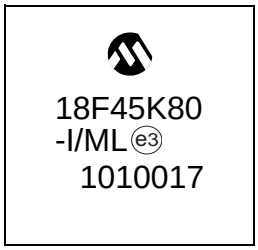
示例



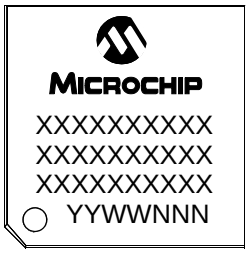
44 引脚 QFN



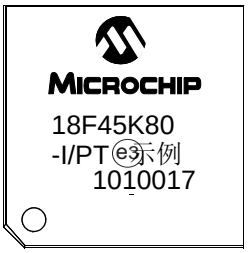
示例



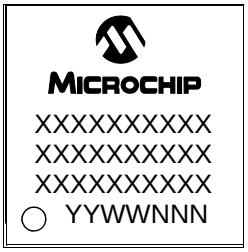
44 引脚 TQFP



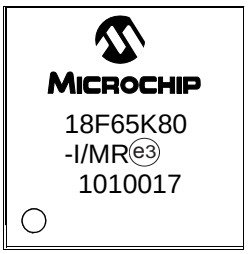
示例



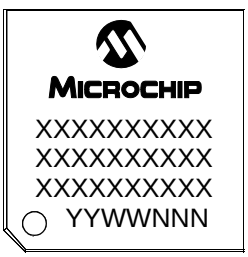
64 引脚 QFN



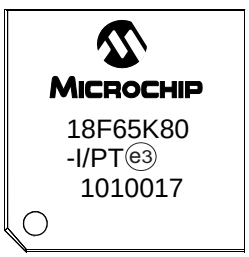
示例



64 引脚 TQFP



示例

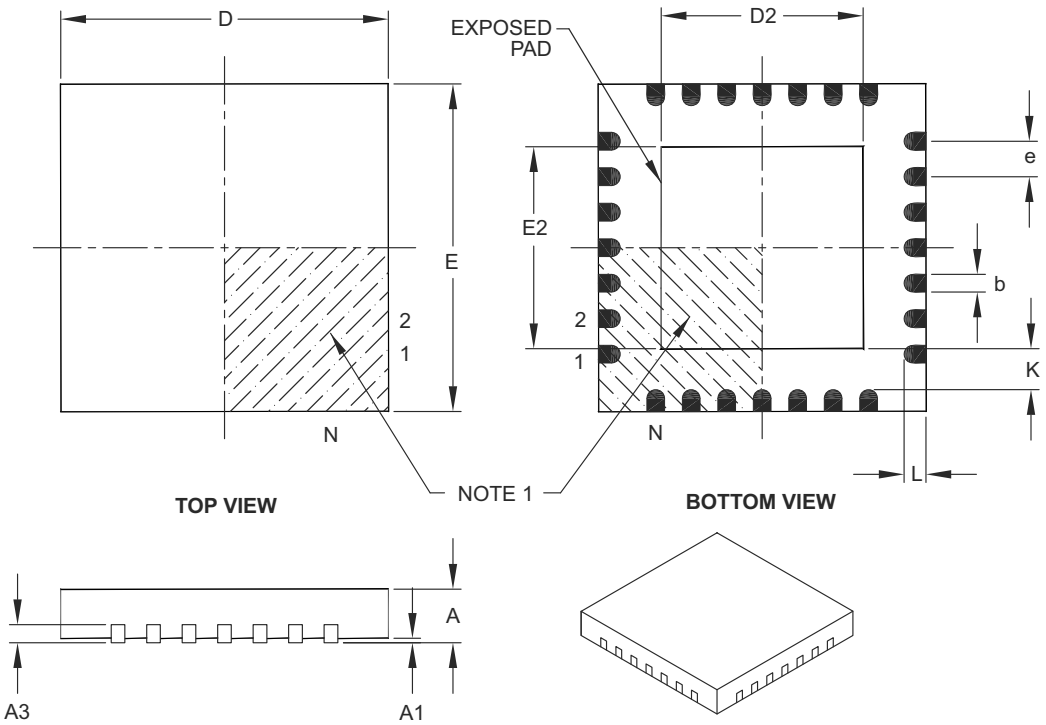


32.2 封装详细信息

以下部分将介绍各种封装的技术细节。

28 引脚塑封正方扁平无脚封装 （MM） —— 主体 6x6x0.9 mm [QFN-S]，触点长度为 0.40 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
Contact Thickness	A3		0.20 REF	
Overall Width	E		6.00 BSC	
Exposed Pad Width	E2	3.65	3.70	4.70
Overall Length	D		6.00 BSC	
Exposed Pad Length	D2	3.65	3.70	4.70
Contact Width	b	0.23	0.38	0.43
Contact Length	L	0.30	0.40	0.50
Contact-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.

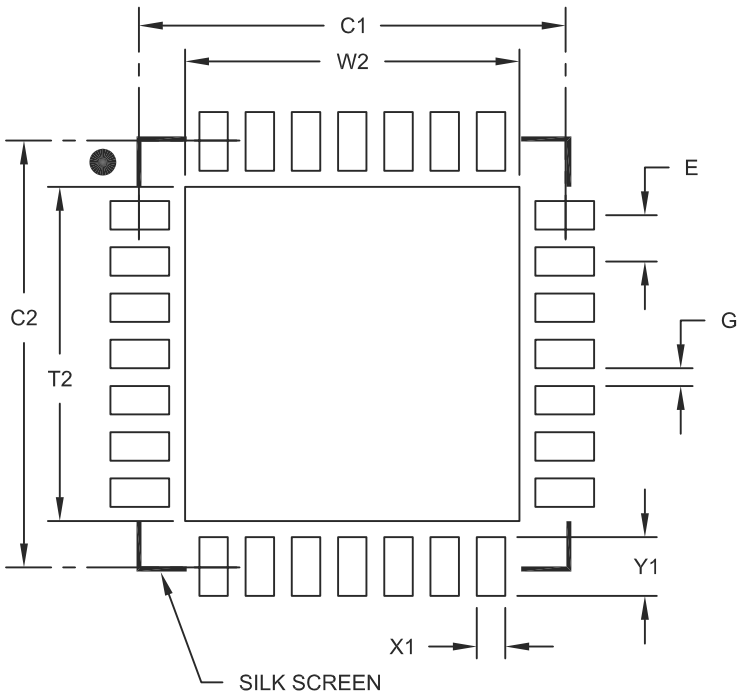
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-124B

28 引脚塑封正方扁平无脚封装 （MM） —— 主体 6x6x0.9 mm [QFN-S]，触点长度为 0.40 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			4.70
Optional Center Pad Length	T2			4.70
Contact Pad Spacing	C1		6.00	
Contact Pad Spacing	C2		6.00	
Contact Pad Width (X28)	X1			0.40
Contact Pad Length (X28)	Y1			0.85
Distance Between Pads	G	0.25		

Notes:

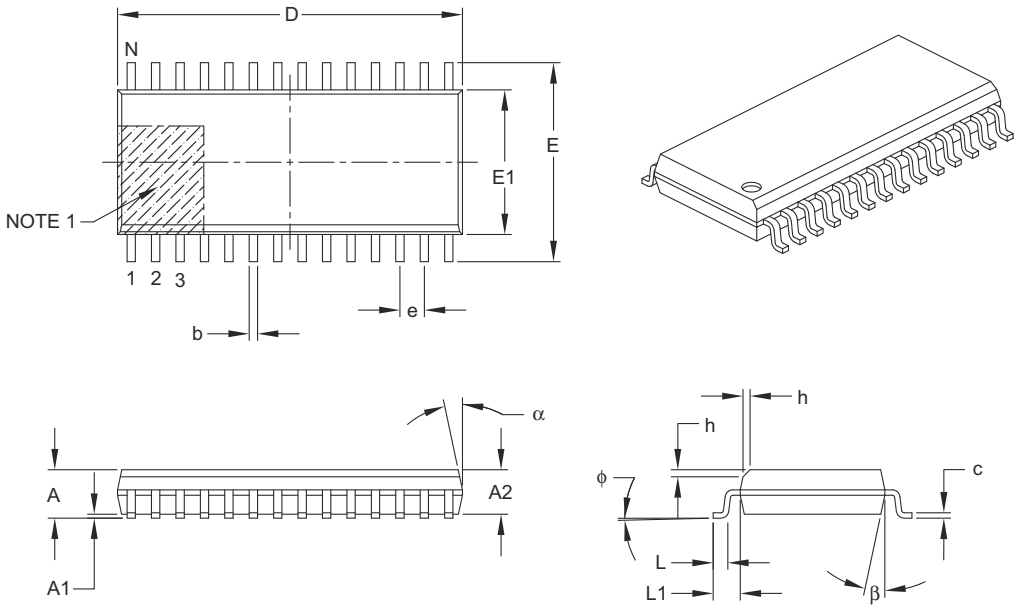
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2124A

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。



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	
Foot Angle Top	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. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

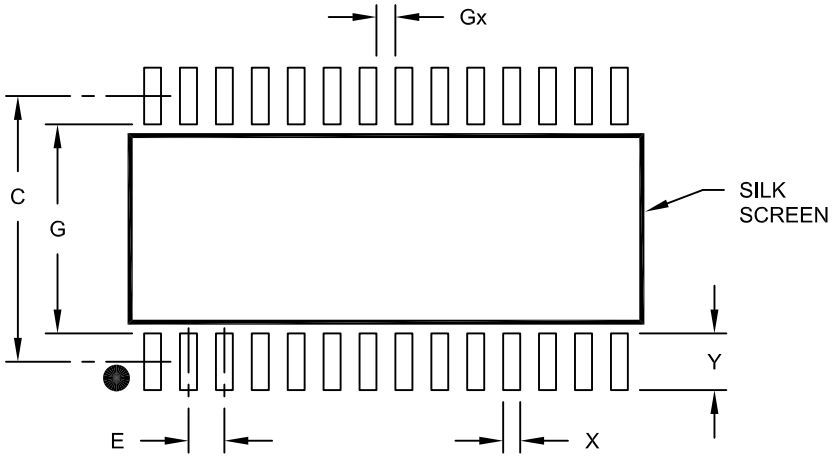
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-052B

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:

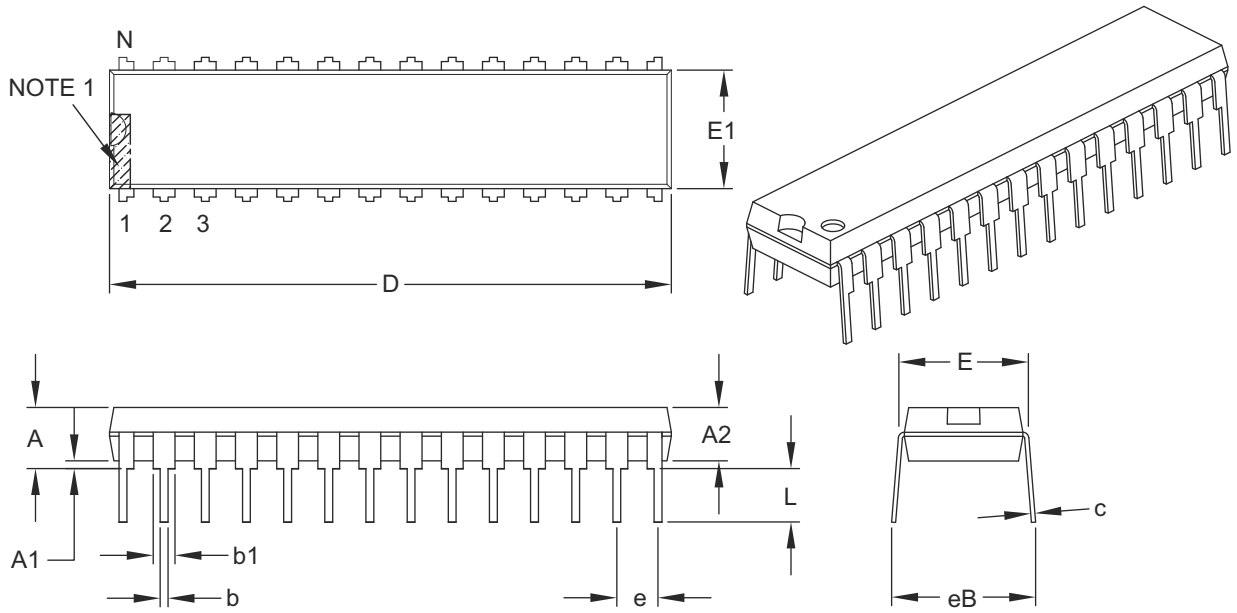
1. Dimensioning and tolerancing per ASME Y14.5M

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:

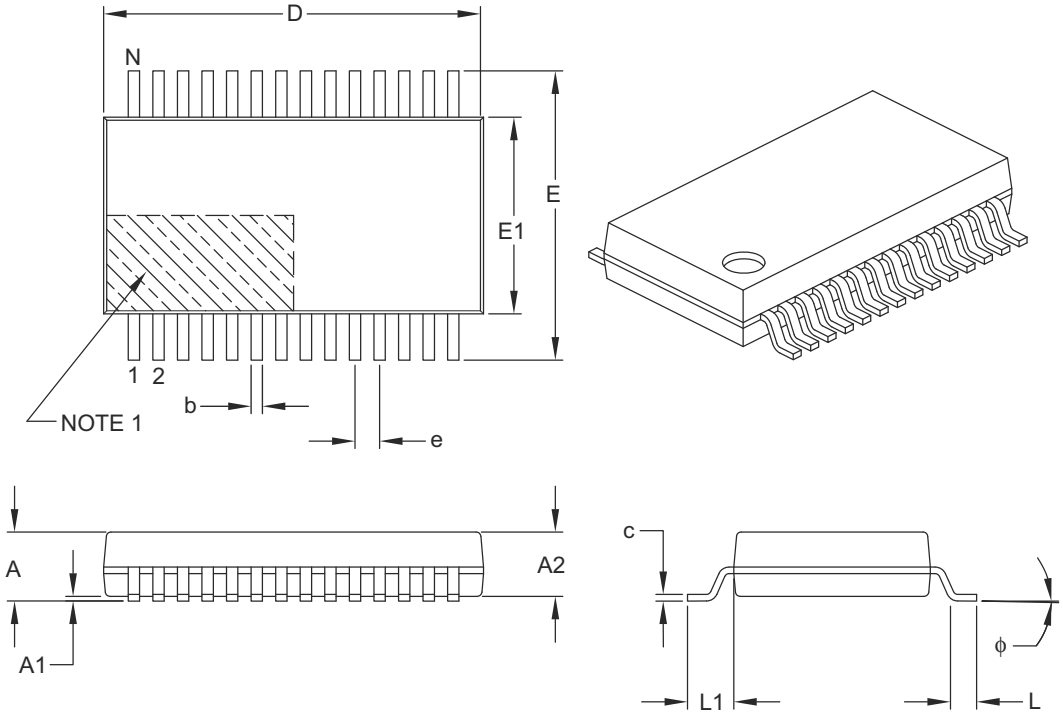
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

28 引脚塑封缩小型小外形封装 （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:

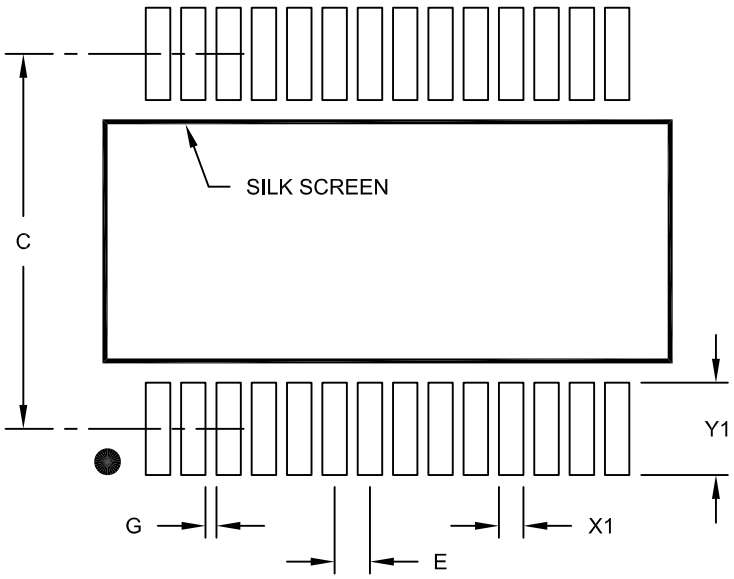
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

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 引脚塑封缩小型小外形封装 （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:

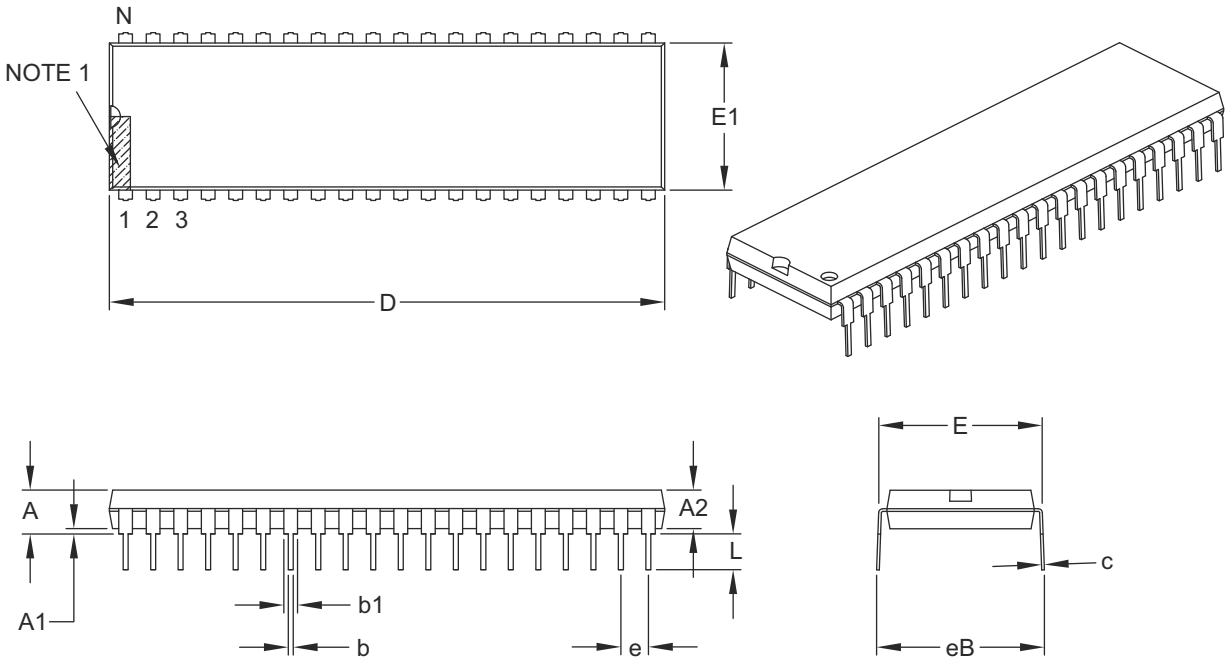
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2073A

40 引脚塑封双列直插式封装 （P） —— 主体 600 mil [PDIP]

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		40	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.250
Molded Package Thickness	A2	.125	–	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.590	–	.625
Molded Package Width	E1	.485	–	.580
Overall Length	D	1.980	–	2.095
Tip to Seating Plane	L	.115	–	.200
Lead Thickness	c	.008	–	.015
Upper Lead Width	b1	.030	–	.070
Lower Lead Width	b	.014	–	.023
Overall Row Spacing §	eB	–	–	.700

Notes:

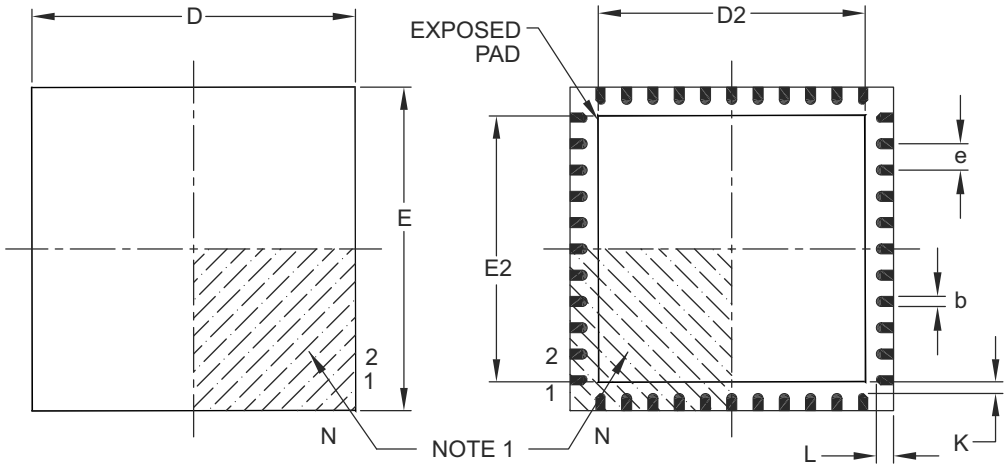
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

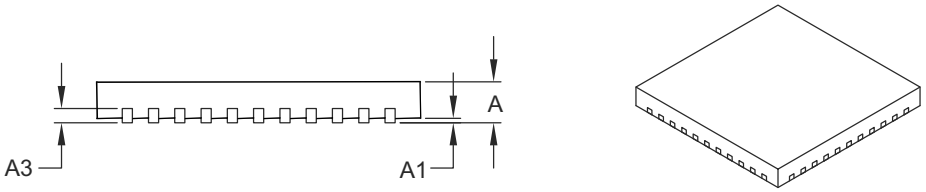
Microchip Technology Drawing C04-016B

44 引脚塑封正方扁平无脚封装 （ML） —— 主体 8x8 mm [QFN]

注： 最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



TOP VIEW **BOTTOM VIEW**



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		44	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.20 REF	
Overall Width	E		8.00 BSC	
Exposed Pad Width	E2	6.30	6.45	6.80
Overall Length	D		8.00 BSC	
Exposed Pad Length	D2	6.30	6.45	6.80
Contact Width	b	0.25	0.30	0.38
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	–	–

Notes:

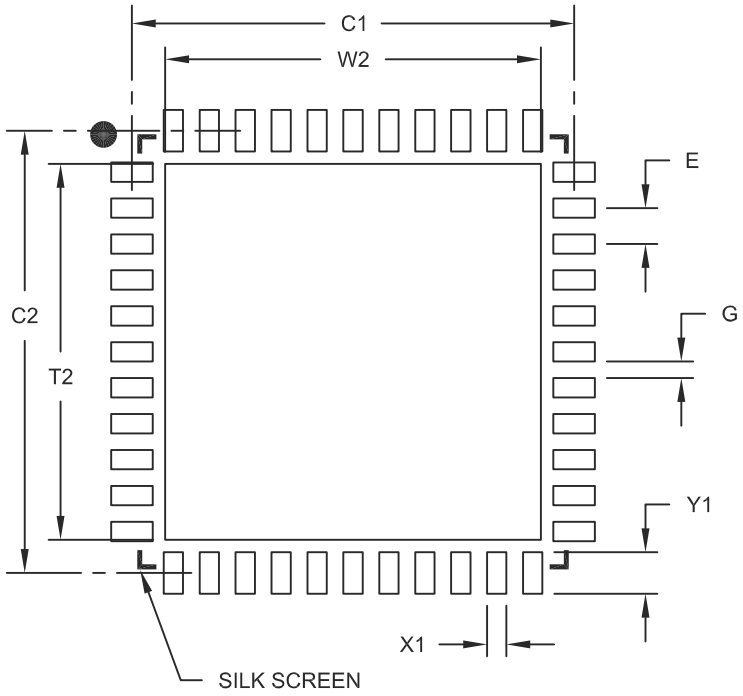
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-103B

44 引脚塑封正方扁平无脚封装 （ML） —— 主体 8x8 mm [QFN]

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			6.80
Optional Center Pad Length	T2			6.80
Contact Pad Spacing	C1		8.00	
Contact Pad Spacing	C2		8.00	
Contact Pad Width (X44)	X1			0.35
Contact Pad Length (X44)	Y1			0.80
Distance Between Pads	G	0.25		

Notes:

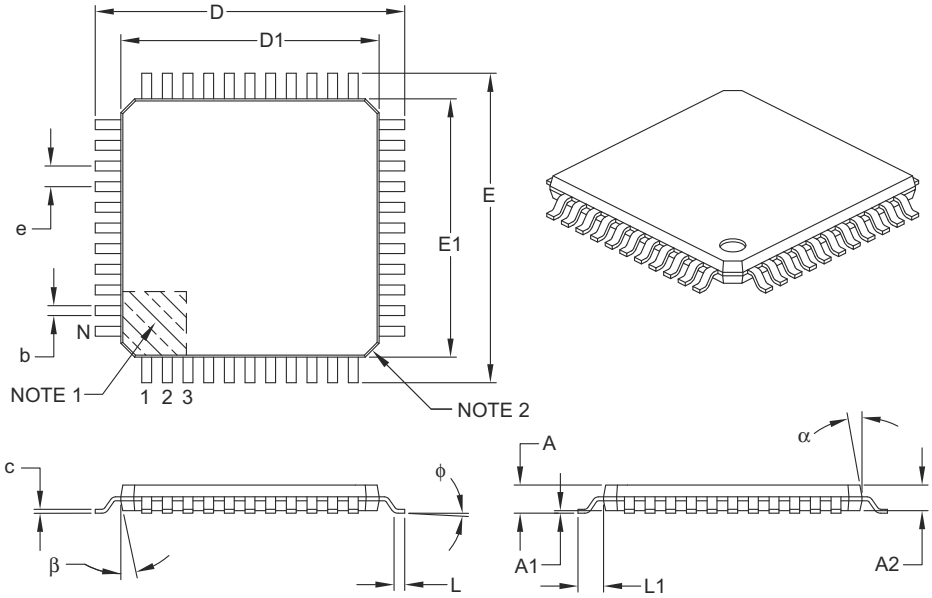
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2103A

44 引脚塑封薄型正方扁平封装 （PT） —— 主体 10x10x1 mm， 2.00 mm [TQFP]

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 Leads	N		44	
Lead Pitch	e		0.80 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	φ	0°	3.5°	7°
Overall Width	E		12.00 BSC	
Overall Length	D		12.00 BSC	
Molded Package Width	E1		10.00 BSC	
Molded Package Length	D1		10.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.30	0.37	0.45
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

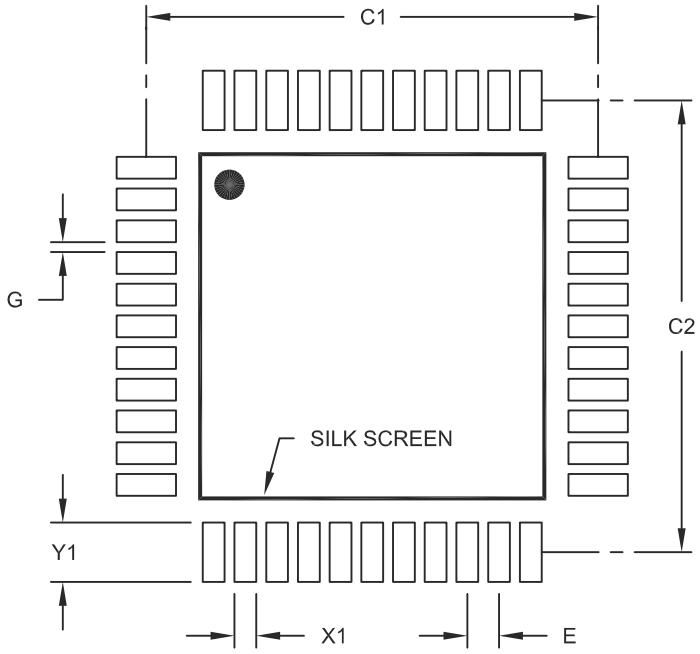
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-076B

44 引脚塑封薄型正方扁平封装 （PT） —— 主体 10x10x1 mm， 2.00 mm [TQFP]

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	0.80 BSC		
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width (X44)	X1			0.55
Contact Pad Length (X44)	Y1			1.50
Distance Between Pads	G	0.25		

Notes:

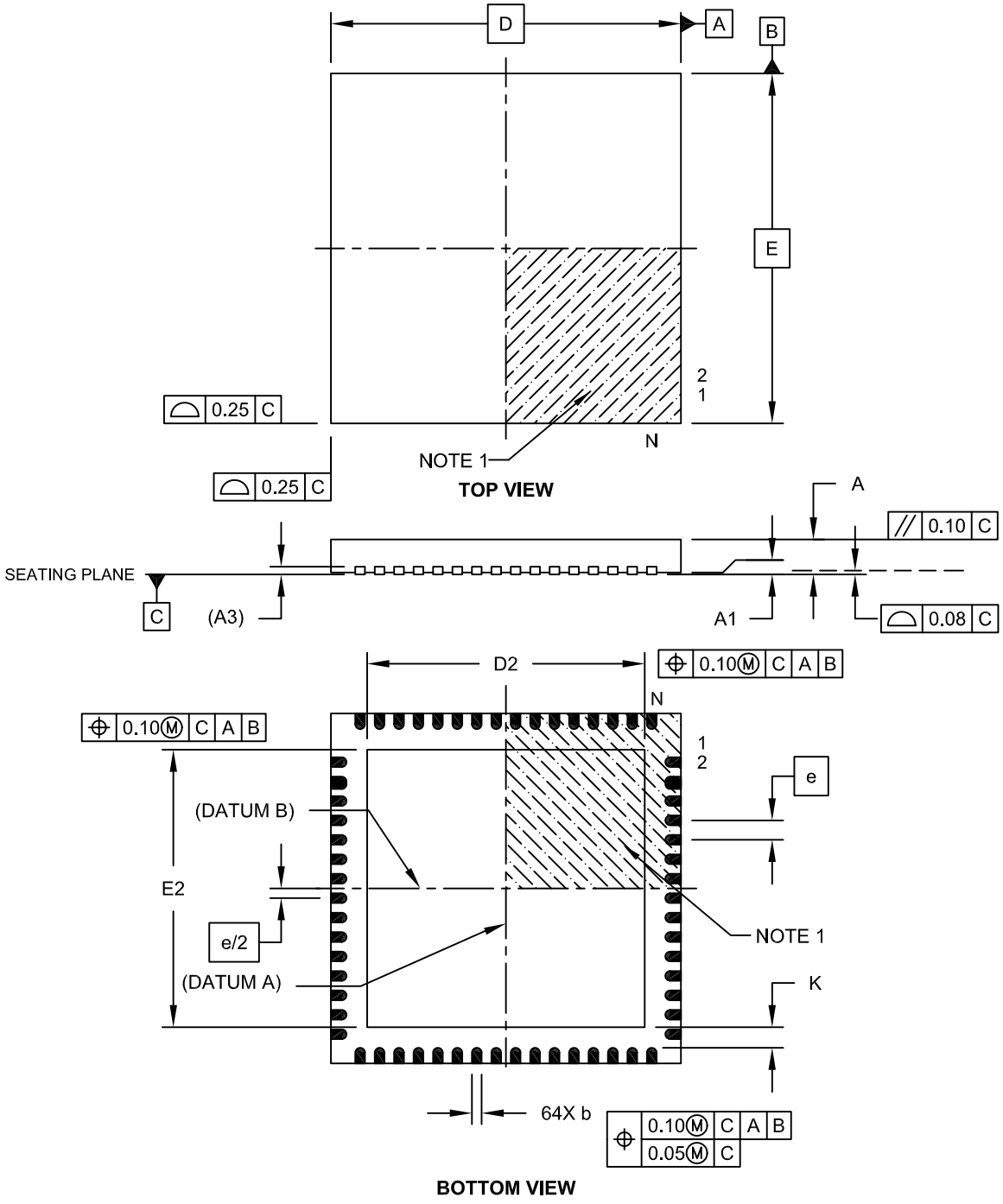
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2076A

64 引脚塑封正方扁平无脚封装 （MR） —— 主体 9x9x0.9 mm [QFN]

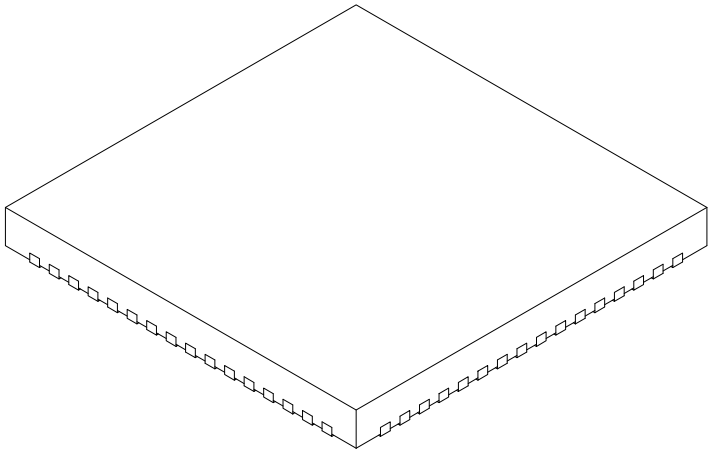
注：	最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



Microchip Technology Drawing C04-149B Sheet 1 of 2

64 引脚塑封正方扁平无脚封装 （MR） —— 主体 9x9x0.9 mm [QFN]

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		64	
Pitch	e		0.50 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.20 REF	
Overall Width	E		9.00 BSC	
Exposed Pad Width	E2	7.05	7.15	7.50
Overall Length	D		9.00 BSC	
Exposed Pad Length	D2	7.05	7.15	7.50
Contact Width	b	0.18	0.25	0.30
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	-	-

Notes:

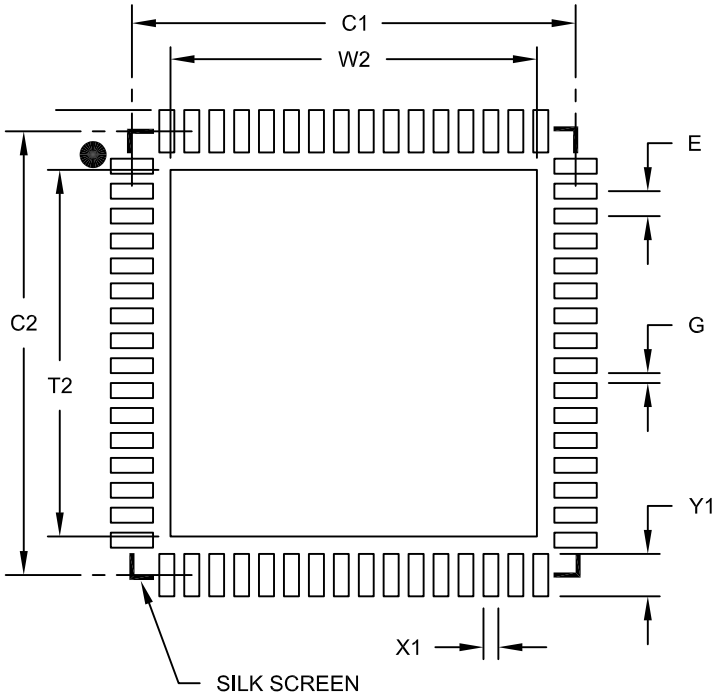
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-149B Sheet 2 of 2

64 引脚塑封正方扁平无脚封装 （MR） —— 主体 9x9x0.9 mm [QFN]，触点长度为 0.40 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		0.50 BSC	
Optional Center Pad Width	W2			7.35
Optional Center Pad Length	T2			7.35
Contact Pad Spacing	C1		8.90	
Contact Pad Spacing	C2		8.90	
Contact Pad Width (X64)	X1			0.30
Contact Pad Length (X64)	Y1			0.85
Distance Between Pads	G	0.20		

Notes:

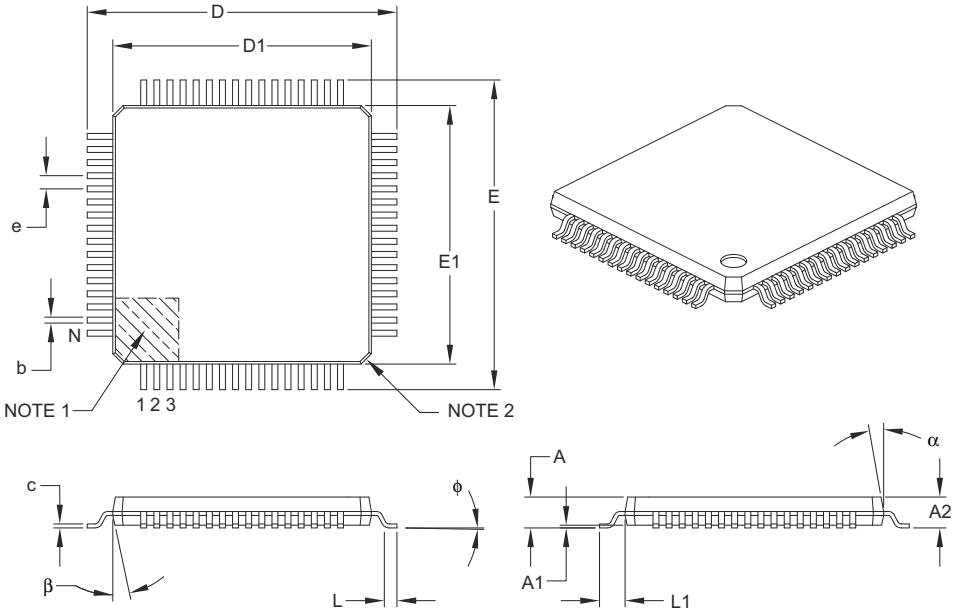
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2149A

64 引脚塑封薄型正方扁平封装 （PT） —— 主体 10x10x1 mm， 2.00 mm [TQFP]

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 Leads	N		64	
Lead Pitch	e		0.50 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	φ	0°	3.5°	7°
Overall Width	E		12.00 BSC	
Overall Length	D		12.00 BSC	
Molded Package Width	E1		10.00 BSC	
Molded Package Length	D1		10.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

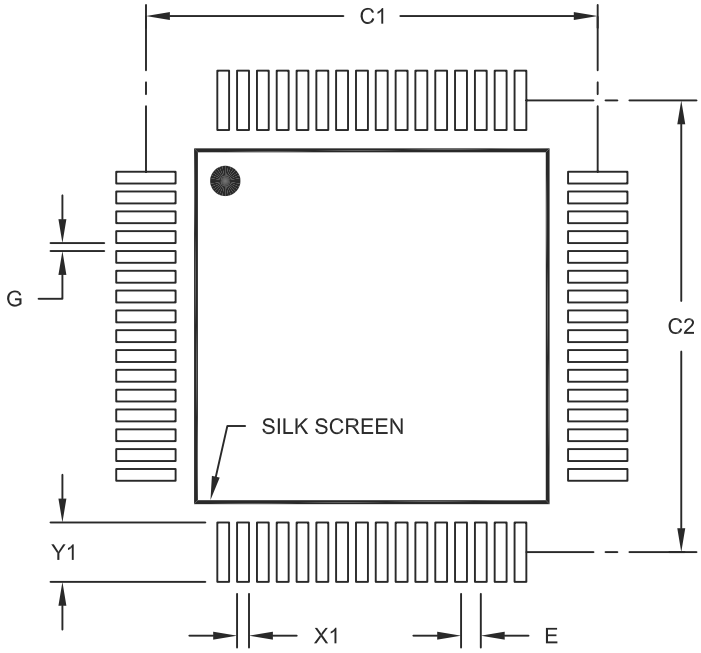
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-085B

64 引脚塑封薄型正方扁平封装 （PT） —— 主体 10x10x1 mm， 2.00 mm [TQFP]

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		0.50 BSC	
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width (X64)	X1			0.30
Contact Pad Length (X64)	Y1			1.50
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. C04-2085A

注：

附录 A： 版本历史

版本 A （2010 年 8 月）

PIC18F66K80 系列器件的原始数据手册。

版本 B （2010 年 12 月）

更改了**第 31.0 节 “ 电气特性 ”**，并对整篇文档的文字进行了少量编辑。

版本 C （2011 年 1 月）

向数据手册中增加了**第 2.0 节 “PIC18FXXKXX 单片机入门指南 ”**。在**第 31.0 节 “ 电气特性 ”**中更改了 PIC18F66K80 系列器件的相关内容。对整篇文档的文字进行了少量编辑。

附录 B： 移植到 PIC18F66K80 系列

PIC18F66K80、PIC18F4580、PIC18F4680和PIC18F8680系列器件的功能和特性类似。可以将代码从其他系列移植到PIC18F66K80而无需进行许多更改。表B-1和表B-2列出了这些器件系列之间的差异。关于移植到PIC18F66K80的更多详细信息，请参见《PIC18FXX80到 PIC18FXXK80 移植指南》（DS39982A_CN）。

表 B-1： 28、40 和 44 引脚器件 ——PIC18F66K80、PIC18F4580 和 PIC18F4680 系列之间的显著差异

特性	PIC18F66K80 系列	PIC18F4680 系列	PIC18F4580 系列
最高工作频率	64 MHz	40 MHz	40 MHz
最大程序存储器	64 KB	64 KB	32 KB
数据存储器 （字节）	3,648	3,328	1,536
CTMU	有	无	无
SOSC 振荡器选项	SOSC 的低功耗振荡器选项	无选项	无选项
T1CKI 时钟	T1CKI 可以在不使能 SOSC 振荡器的情况下用作时钟	无	无
INTOSC	最高 16 MHz	最高 8 MHz	最高 8 MHz
定时器	2 个 8 位， 3 个 16 位	1 个 8 位， 3 个 16 位	1 个 8 位， 3 个 16 位
ECCP	所有器件均有 1 个	40 和 44 引脚器件 ——1 个 28 引脚器件 —— 无	40 和 44 引脚器件 ——1 个 28 引脚器件 —— 无
CCP	4 个	1 个	1 个
数据 EEPROM （字节）	1,024	1,024	256
WDT 预分频比选项	22	16	16
5V 工作电压	18FXXK80 器件 ——5V 工作电压 18LFXXK80 器件 ——3.3V 工作电压	有	有
nanoWatt XLP	有	无	无
稳压器	18FXXK80 器件 —— 有 18LFXXK80 器件 —— 无	无	无
低功耗 BOR	有	无	无
A/D 转换器	12 位有符号差分	10 位	10 位
A/D 通道	28 引脚器件 ——8 路通道 40 和 44 引脚器件 ——15 路通道	28 引脚器件有 8 路通道 / 40 和 44 引脚器件有 11 路通道	28 引脚器件有 8 路通道 / 40 和 44 引脚器件有 11 路通道
内部温度传感器	有	无	无
EUSART	2 个	1 个	1 个
比较器	2 个	28 引脚器件 —— 无 40 和 44 引脚器件 ——2 个	28 引脚器件 —— 无 40 和 44 引脚器件 ——2 个
振荡器选项	14 个	9 个	9 个
超低功耗唤醒 （ULPW）	有	无	无
针对 I/O 的可调节压摆率	有	无	无
PLL	可用于所有振荡器选项	仅用于高速晶振和内部振荡器	仅用于高速晶振和内部振荡器
TXM 调制器	无	无	无

表 B-2： 64 引脚器件 ——PIC18F66K80 和 PIC18F8680 系列之间的显著差异

特性	PIC18F66K80 系列	PIC18F8680 系列
最高工作频率	64 MHz	40 MHz
最大程序存储器	64K	64K
数据存储器 （字节）	3,648	3,328
CTMU	有	无
SOSC 振荡器选项	SOSC 的低功耗振荡器选项	无选项
T1CKI 时钟	T1CKI 可以在不使能 SOSC 振荡器的情况下用作时钟	无
INTOSC	最高 16 MHz	无内部振荡器
SPI/I^2C™	1 个模块	1 个模块
定时器	2 个 8 位， 3 个 16 位	2 个 8 位， 3 个 16 位
ECCP	1	1
CCP	4	1
数据 EEPROM （字节）	1,024	1,024
WDT 预分频比选项	22	16
5V 工作电压	18FXXK80 器件 ——5V 工作电压 18LFXXK80 器件 ——3.3V 工作电压	有
nanoWatt XLP	有	无
片上 3.3V 稳压器	18FXXK80 器件 —— 有 18LFXXK80 器件 —— 无	无
低功耗 BOR	有	无
A/D 转换器	12 位有符号差分	10 位
A/D 通道	15 路通道	12 路通道
内部温度传感器	有	无
EUSART	2 个	1 个
比较器	2 个	2 个
振荡器选项	14 个	7 个
超低功耗唤醒 （ULPW）	有	无
针对 I/O 的可调节压摆率	有	无
PLL	可用于所有振荡器选项	仅用于高速晶振和外部振荡器
数据信号调制器	有	无

索引

A

B

C

D

E

F

G

H

I

J

K

L

M

N

P

Q

R

S

T

W

X

Y

Z

注：

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。只要使用常用的因特网浏览器即可访问。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存档软件
- **一般技术支持** —— 常见问题 （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。

读者反馈表

我们努力为您提供最佳文档，以确保您能够成功使用 Microchip 产品。如果您对文档的组织、条理性、主题及其他有助于提高文档质量的方面有任何意见或建议，请填写本反馈表并传真给我公司 TRC 经理，传真号码为 86-21-5407-5066。

请填写以下信息，并从下面各方面提出您对本文档的意见。

致： TRC 经理　　总页数 _____

关于： 读者反馈

发自： 姓名 _____

公司 _____

地址 _____

国家 / 省份 / 城市 / 邮编 _____

电话：(_____) _____ - _____　　传真：(_____) _____ - _____

应用（选填）：

您希望收到回复吗？是 ____ 否____

器件：PIC18F66K80 系列　　文献编号：DS39977C_CN

问题：

1. 本文档中哪些部分最有特色？

2. 本文档是否满足了您的软硬件开发要求？如何满足的？

3. 您认为本文档的组织结构便于理解吗？如果不便于理解，那么问题何在？

4. 您认为本文档应该添加哪些内容以改善其结构和主题？

5. 您认为本文档中可以删减哪些内容，而又不会影响整体使用效果？

6. 本文档中是否存在错误或误导信息？如果存在，请指出是什么信息及其具体页数。

7. 您认为本文档还有哪些方面有待改进？

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。

器件编号	X	/XX	XXX
器件	温度范围	封装	模式

器件 (1,2)	PIC18F25K80、 PIC18F26K80、 PIC18F45K80、 PIC18F46K80、 PIC18F65K80 和 PIC18F66K80 VDD 范围为 1.8V 至 5V PIC18LF25K80、 PIC18LF26K80、 PIC18LF45K80、 PIC18LF46K80、 PIC18LF65K80 和 PIC18LF66K80 VDD 范围为 1.8V 至 3.6V
温度范围	I = -40°C 至 +85°C （工业级） E = -40°C 至 +125°C （扩展级）
封装	PDIP = 塑封双列直插式封装 QFN = 塑封正方扁平无脚封装 SOIC = 塑封小外形封装 SPDIP = 窄型塑封双列直插式封装 SSOP = 塑封缩小型小外形封装 TQFP = 塑封薄型正方扁平封装
模式	a) QTP、 SQTP、代码或特殊要求 （其他情况空白）

示例：

a) PIC18F66K80-I/MR 301 = 工业级温度，QFN 封装，扩展 VDD 范围， QTP 模式 #301。

b) PIC18F66K80-I/PT = 工业级温度， TQFP 封装，扩展 VDD 范围。

注 1： F = 标准电压范围
LF = 宽电压范围
2： T = 卷带式，仅限 TQFP 封装。

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

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

克里夫兰 Cleveland
Independence, OH
Tel: 1-216-447-0464
Fax: 1-216-447-0643

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Farmington Hills, MI
Tel: 1-248-538-2250
Fax: 1-248-538-2260

印第安纳波利斯 Indianapolis
Noblesville, IN
Tel: 1-317-773-8323
Fax: 1-317-773-5453

洛杉矶 Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608

圣克拉拉 Santa Clara
Santa Clara, CA
Tel: 1-408-961-6444
Fax: 1-408-961-6445

加拿大多伦多 Toronto
Mississauga, Ontario, Canada
Tel: 1-905-673-0699
Fax: 1-905-673-6509

亚太地区

亚太总部 Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 北京
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

中国 – 成都
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

中国 – 重庆
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

中国 – 杭州
Tel: 86-571-2819-3180
Fax: 86-571-2819-3189

中国 – 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 南京
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

中国 – 青岛
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 – 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 – 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 – 深圳
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

中国 – 武汉
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 – 西安
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

中国 – 厦门
Tel: 86-592-238-8138
Fax: 86-592-238-8130

中国 – 珠海
Tel: 86-756-321-0040
Fax: 86-756-321-0049

亚太地区

台湾地区 – 高雄
Tel: 886-7-213-7830
Fax: 886-7-330-9305

台湾地区 – 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

台湾地区 – 新竹
Tel: 886-3-6578-300
Fax: 886-3-6578-370

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

印度 India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

印度 India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

日本 Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

韩国 Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

韩国 Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

马来西亚 Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

马来西亚 Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

菲律宾 Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

新加坡 Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

泰国 Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

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

法国 France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

意大利 Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

荷兰 Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

西班牙 Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

英国 UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

05/02/11