PIC18F87K22 系列
数据手册

带 12 位 A/D 转换器、
采用 nanoWatt XLP 技术的
64/80 引脚高性能
1Mb 增强型闪存单片机

© 2011 Microchip Technology Inc.

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。

提供本文档的中文版本仅为了便于理解。请勿忽视文档中包含的英文部分，因为其中提供了有关 Microchip 产品性能和使用情况的有用信息。Microchip Technology Inc. 及其分公司和相关公司、各级主管与员工及事务代理机构对译文中可能存在的任何差错不承担任何责任。建议参考 Microchip Technology Inc. 的英文原版文档。

本出版物中所述的器件应用信息及其他类似内容仅为您提供便利，它们可能由更新之信息所替代。确保应用符合技术规范，是您自身应负的责任。**Microchip 对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对其使用情况、质量、性能、适销性或特定用途的适用性的声明或担保。** Microchip 对因这些信息及使用这些信息而引起的后果不承担任何责任。如果将 Microchip 器件用于生命维持和 / 或生命安全应用，一切风险由买方自负。买方同意在由此引发任何一切伤害、索赔、诉讼或费用时，会维护和保障 Microchip 免于承担法律责任，并加以赔偿。在 Microchip 知识产权保护下，不得暗中或以其他方式转让任何许可证。

商标

Microchip 的名称和徽标组合、 Microchip 徽标、 dsPIC、 KEELOQ、 KEELOQ 徽标、 MPLAB、 PIC、 PICmicro、 PICSTART、 PIC32 徽标、 rfPIC 和 UNI/O 均为 Microchip Technology Inc. 在美国和其他国家或地区的注册商标。

FilterLab、 Hampshire、 HI-TECH C、 Linear Active Thermistor、MXDEV、MXLAB、SEEVAL 和 The Embedded Control Solutions Company 均为 Microchip Technology Inc. 在美国的注册商标。

Analog-for-the-Digital Age、 Application Maestro、 chipKIT、 chipKIT 徽标、 CodeGuard、 dsPICDEM、 dsPICDEM.net、 dsPICworks、 dsSPEAK、 ECAN、 ECONOMONITOR、 FanSense、 HI-TIDE、 In-Circuit Serial Programming、 ICSP、 Mindi、 MiWi、 MPASM、 MPLAB Certified 徽标、 MPLIB、MPLINK、mTouch、Omniscient Code Generation、 PICC、 PICC-18、 PICDEM、 PICDEM.net、 PICkit、 PICtail、 REAL ICE、 rfLAB、 Select Mode、 Total Endurance、 TSHARC、 UniWinDriver、 WiperLock 和 ZENA 均为 Microchip Technology Inc. 在美国和其他国家或地区的商标。

SQTP 是 Microchip Technology Inc. 在美国的服务标记。

在此提及的所有其他商标均为各持有公司所有。

© 2011, Microchip Technology Inc. 版权所有。

ISBN：978-1-61341-936-6

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2009

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-16949:2009 认证。 Microchip 的 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严格遵守公司的质量体系流程。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。

© 2011 Microchip Technology Inc.

PIC18F87K22 系列

带 12 位 ADC 模块、采用 nanoWatt XLP 技术的 64/80 引脚高性能 1 Mb 增强型闪存 MCU

低功耗特性：

- 功耗管理模式：
  - 运行模式：CPU 工作，外设工作
  - 空闲模式：CPU 不工作，外设工作
  - 休眠模式：CPU 不工作，外设不工作
- 振荡器双速启动
- 故障保护时钟监视器
- 节能的外设模块禁止 （Peripheral Module Disable， PMD）功能
- 超低功耗唤醒功能
- 快速唤醒，典型值为 1 µs
- 低功耗 WDT，典型值为 300 nA
- 50 nA 的超低输入漏电流
- 运行模式电流可低至 5.5 µA （典型值）
- 空闲模式电流可低至 1.7 µA （典型值）
- 休眠模式电流可低至 20 nA （典型值）
- RTCC 电流低至 700 nA （典型值）

单片机的特殊性能：

- 工作电压范围：1.8V 到 5.5V
- 片上 3.3V 稳压器
- 工作速度最高 64 MHz
- 高达 128 KB 的片上闪存程序存储器
- 1024 字节的数据 EEPROM
- 4K x 8 通用寄存器 （SRAM）
- 可承受 10000 次 （最小值）擦 / 写的闪存程序存储器
- 可承受 1000000 次 （典型值）擦 / 写的数据 EEPROM 存储器
- 闪存的数据保持时间至少为 40 年
- 3 个内部振荡器：LF-INTRC （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
- 扩展的看门狗定时器 （Watchdog Timer，WDT）：
  - 可编程周期从 4 ms 到 4194s （大约 70 分钟）
- 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
  - LVD

器件	程序存储器		数据存储器		I/O	12 位 A/D （通道数）	CCP/ ECCP （PWM）	MSSP			EUSART	比较器	8/16 位定时器	外部总线	CTMU	RTCC
	闪存 （字节）	单字 指令数	SRAM （字节）	EEPROM （字节）					SPI	主 I^2C™						
PIC18F65K22	32K	16383	2K	1K	53	16	5/3	2	有	有	2	3	4/4	无	有	有
PIC18F66K22	64K	32768	4K	1K	53	16	7/3	2	有	有	2	3	6/5	无	有	有
PIC18F67K22	128K	65536	4K	1K	53	16	7/3	2	有	有	2	3	6/5	无	有	有
PIC18F85K22	32K	16383	2K	1K	69	24	5/3	2	有	有	2	3	4/4	有	有	有
PIC18F86K22	64K	32768	4K	1K	69	24	7/3	2	有	有	2	3	6/5	有	有	有
PIC18F87K22	128K	65536	4K	1K	69	24	7/3	2	有	有	2	3	6/5	有	有	有

© 2011 Microchip Technology Inc.

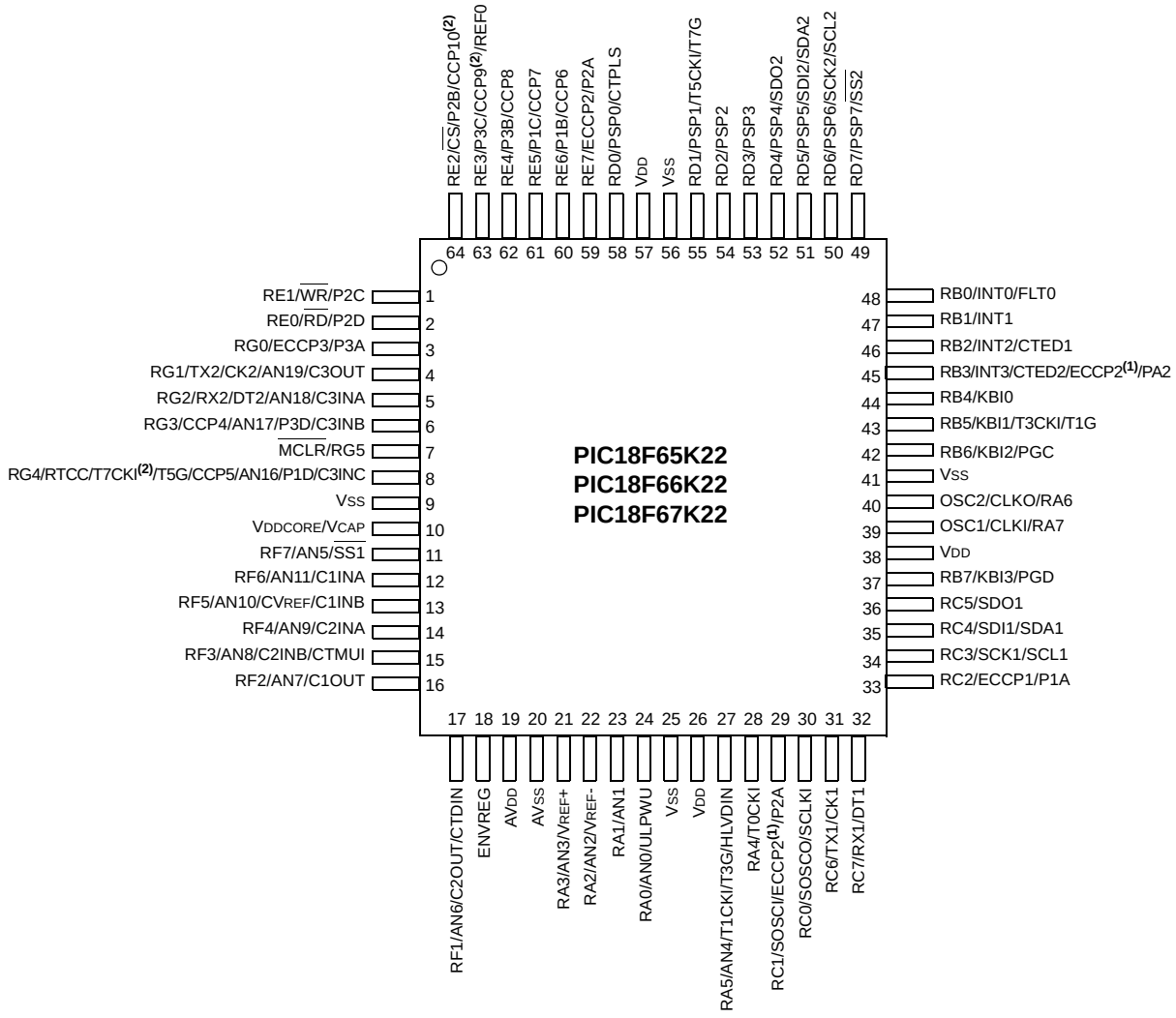
外设特点：

- 多达 10 个 CCP/ECCP 模块：
  - 多达 7 个捕捉 / 比较 /PWM （Capture/Compare/PWM， CCP）模块
  - 3 个增强型捕捉 / 比较 /PWM （Enhanced Capture/Compare/PWM， ECCP）模块
- 多达 11 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 和 3——16 位定时器 / 计数器
  - Timer2、 4、 6 和 8——8 位定时器 / 计数器
  - Timer5 和 7——64 KB 和 128 KB 器件的 16 位定时器 / 计数器
  - Timer10 和 12——64 KB 和 128 KB 器件的 8 位定时器 / 计数器
- 3 个模拟比较器
- 可配置参考时钟输出
- 具有时钟、日历和闹钟功能的硬件实时时钟和日历（Real-Time Clock and Calendar， RTCC）
- 充电时间测量单元 （Charge Time Measurement Unit， CTMU）：
  - 在 mTouch™ 触摸传感解决方案中测量电容
  - 分辨率为 1 ns （典型值）的时间测量
  - 集成温度传感器
- 高灌 / 拉电流 25 mA/25 mA （PORTB 和 PORTC）
- 最多 4 个外部中断
- 2 个主同步串口 （Master Synchronous Serial Port， MSSP）模块：
  - 支持 3/4 线 SPI （支持所有 4 种 SPI 模式）
  - 支持 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）
- 具有最多 24 路通道的 12 位 A/D 转换器：
  - 自动采集和休眠操作
  - 差分输入模式操作
- 集成参考电压

© 2011 Microchip Technology Inc.

引脚图——PIC18F6XK22

64 引脚 TQFP 和 QFN

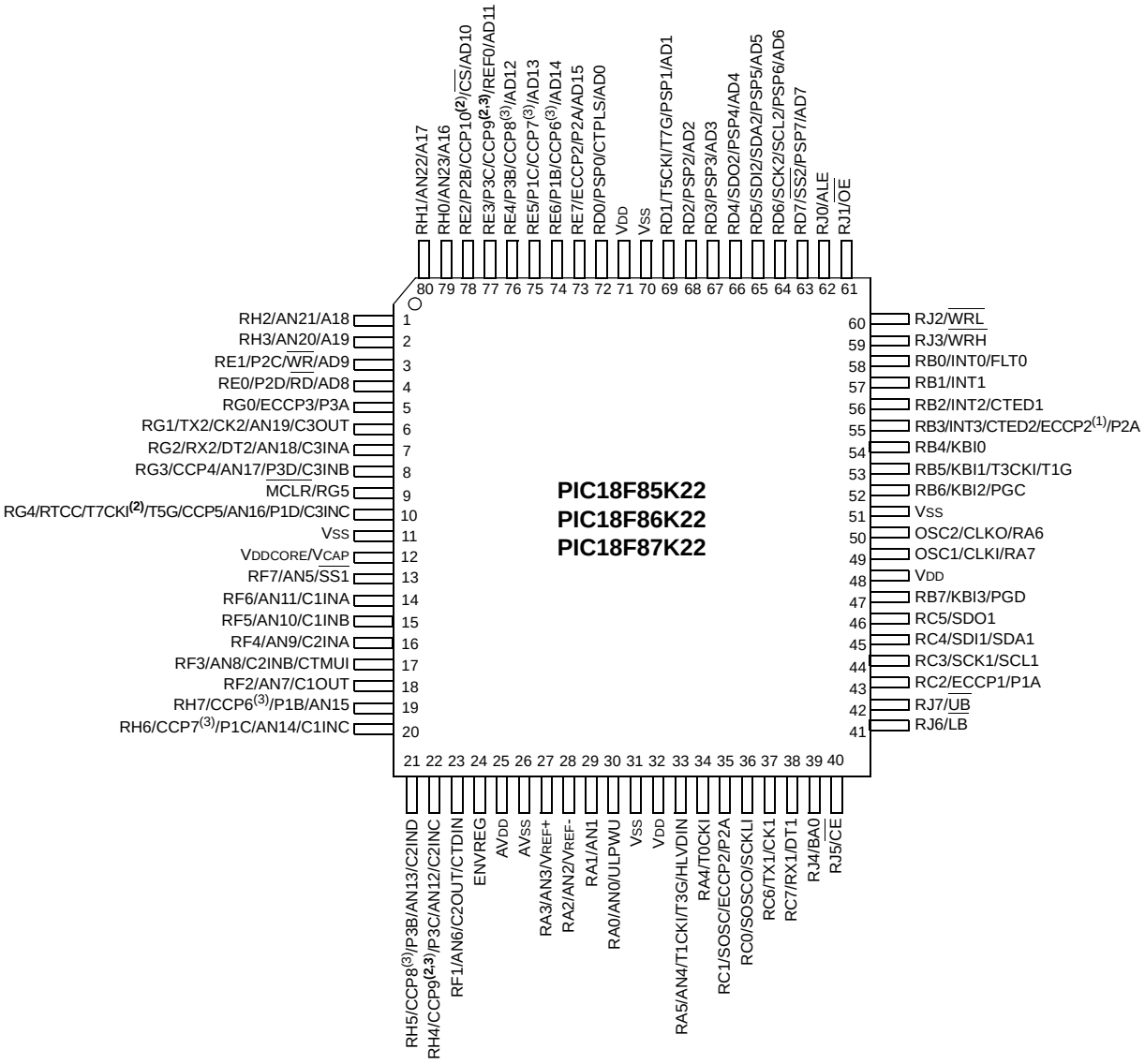


注 **1：** ECCP2 引脚位置取决于 CCP2MX 配置位的设置以及器件是处于单片机模式还是扩展单片机模式。

2： 在 PIC18F65K22 和 PIC18F85K22 器件中不可用。

引脚图——PIC18F8XK22

80 引脚 TQFP



注 **1：** ECCP2 引脚位置取决于 CCP2MX 配置位的设置以及器件是处于单片机模式还是扩展单片机模式。

2： 在 PIC18F65K22 和 PIC18F85K22 器件中不可用。

3： CCP6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

目录

1.0 器件概述 9
2.0 PIC18FXXKXX 单片机入门指南 37
3.0 振荡器配置 43
4.0 功耗管理模式 57
5.0 复位 73
6.0 存储器构成 87
7.0 闪存程序存储器 111
8.0 外部存储器总线 121
9.0 数据 EEPROM 存储器 133
10.0 8 x 8 硬件乘法器 139
11.0 中断 141
12.0 I/O 端口 165
13.0 Timer0 模块 193
14.0 Timer1 模块 197
15.0 Timer2 模块 209
16.0 Timer3/5/7 模块 211
17.0 Timer4/6/8/10/12 模块 223
18.0 实时时钟和日历 （RTCC） 227
19.0 捕捉 / 比较 /PWM （CCP）模块 245
20.0 增强型捕捉 / 比较 /PWM （ECCP）模块 259
21.0 主同步串行端口 （MSSP）模块 281
22.0 增强型通用同步 / 异步收发器 （EUSART） 327
23.0 12 位模数转换器 （A/D）模块 351
24.0 比较器模块 367
25.0 比较器参考电压模块 375
26.0 高 / 低电压检测 （HLVD） 379
27.0 充电时间测量单元 （CTMU） 385
28.0 CPU 的特殊功能 403
29.0 指令集汇总 431
30.0 开发支持 481
31.0 电气特性 485
32.0 封装信息 525
附录 A：版本历史 533
附录 B：从 PIC18F87J11 和 PIC18F8722 到 PIC18F87K22 的移植 534
索引 535
Microchip 网站 547
变更通知客户服务 547
客户支持 547
读者反馈表 548
产品标识体系 549

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。

© 2011 Microchip Technology Inc.

1.0 器件概述

该文档包含针对以下器件的特定信息：

- PIC18F65K22
- PIC18F66K22
- PIC18F67K22
- PIC18F85K22
- PIC18F86K22
- PIC18F87K22

该系列器件在保持极具竞争力的价格的同时，融合了 PIC18 单片机固有的优点（即出色的计算性能和丰富的功能）。这些特性使得 PIC18F87K22 系列器件成为许多将价格作为首选因素的高性能应用的明智选择。

1.1 内核特性

1.1.1 纳瓦技术

PIC18F87K22 系列的所有器件都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 时钟源或内部 RC 振荡器作为单片机时钟源，可使代码执行时的功耗降低。
- **多种空闲模式：**单片机还可在其 CPU 内核禁止而外设仍然工作的情况下运行。在这些状态下，功耗能降得更低。
- **动态模式切换：**在器件工作期间可由用户代码调用该功耗管理模式，允许用户将节能的理念融入到他们的应用软件设计中。
- **nanoWatt XLP：**超低功耗休眠、 BOR、 RTCC 和看门狗定时器

1.1.2 振荡器选项和特性

PIC18F87K22 系列的所有器件都能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，可用于外部振荡器模式，可提供高达 64 MHz 的时钟速度。 PLL 还可与内部振荡器结合使用。
- 一个内部振荡器模块，它提供一个 16 MHz 时钟（精度为 ±2%）和一个 INTRC 时钟源 （频率大约为 31 kHz，在温度和 V_{DD} 变化时保持稳定）
  - 选择的频率为 16 MHz 或 500 kHz 时该模块分别作为 HF-INTOSC 或 MF-INTOSC 工作
  - 空出两个振荡器引脚，以用作额外的通用 I/O

内部振荡器模块提供稳定的参考源，为该系列器件增加了以下额外功能，以使器件更可靠地工作：

- **故障保护时钟监视器：**该选项不停地监视主时钟源，将其与内部振荡器提供的参考信号作比较。如果发生了时钟故障，单片机会切换为内部振荡器，使器件可继续低速工作或安全地关闭。
- **双速启动：**该选项允许器件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，直到主时钟源可正常工作为止。

1.1.3 存储器选项

PIC18F87K22 系列器件为应用代码提供了 32 KB 到 128 KB 的充足代码空间。程序存储器的闪存单元可反复擦写最多 10000 次。如果不刷新，数据保存期保守地估计在 40 年以上。

闪存程序存储器可读写。在正常工作期间，PIC18F87K22 系列器件还为动态应用数据提供了足够的空间和最多 3,862 字节的数据 RAM。

1.1.4 外部存储器总线

对于需要的存储空间高于 128 KB 的应用，PIC18F87K22 系列中的 80 引脚器件还提供一个外部存储器总线 （External Memory Bus， EMB），使得单片机的内部程序计数器可以寻址多达 2 MB 的存储空间。只有少数 8 位器件具有这种数据访问能力：

- 结合使用片上存储器和外部存储器，存储空间可高达 2 MB
- 使用外部闪存存储可重新编程的应用代码或大数据表
- 将外部 RAM 用于存储大量的可变数据

1.1.5 扩展指令集

PIC18F87K22 系列器件在 PIC18 指令集的基础上进行了可选择的扩展，添加了 8 条新指令和变址寻址模式。此扩展指令集可以作为一个器件配置选项使能，它是为优化原先使用高级语言（如 C 语言）开发的重入应用程序代码而特别设计的。

1.1.6 易于移植

所有器件均共享相同的丰富外设集，但具有 32 KB 程序存储器的器件 （PIC18F65K22 和 PIC18F85K22）除外，这些器件少了 2 个 CCP 和 3 个定时器，使得在更改和升级应用时能方便地在器件间移植。

整个系列的引脚排列设计一致也有助于向下一代更大的器件移植。在 64 引脚器件间、 80 引脚器件间移植，甚至是从 64 引脚向 80 引脚器件的移植都是可行的。

该系列中的所有器件均共享相同的丰富外设集，但具有 32 KB 程序存储器的器件 （PIC18F65K22 和 PIC18F85K22）除外，这些器件少了2个CCP和3个定时器。

PIC18F87K22 系列器件的引脚与其他 PIC18 系列器件（例如 PIC18F8720、 PIC18F8722 和 PIC18F85J11）非常兼容。这为应用的更新换代开拓了新的视野，使开发人员能在保留相同功能集的同时在 Microchip PIC18 系列中选择不同价位的器件。

1.2 其他特殊功能

- **通信：**PIC18F87K22 系列器件具有一系列串行通信外设，包括两个支持 LIN/J2602 的增强型 USART （EUSART）和两个具备 SPI 和 I^2C™ （主和从）工作模式的主 SSP 模块。
- **CCP 模块：**PIC18F87K22 系列器件最多包含七个捕捉 / 比较 /PWM （CCP）模块。一次最多可能会采用 6 个不同的时基执行几项不同的操作。
- **ECCP 模块：**PIC18F87K22 系列器件具有 3 个增强型 CCP （ECCP）模块以最大程度地提高控制应用的灵活性：
  - 最多八个不同的时基，用于一次执行几项不同的操作。
  - 每个模块最多 4 个 PWM 输出，一共 12 个 PWM。
  - 其他有用的功能，如极性选择、可编程死区时间、自动关闭和重新启动，以及半桥和全桥输出模式
- **12 位 A/D 转换器：**PIC18F87K22 系列器件具有差分 ADC。它具有可编程采集时间，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 与其他片上模拟模块一起使用可以精确测量时间、测量电容或电容的相对变化，或者生成独立于系统时钟的输出脉冲。
- **LP 看门狗定时器 （WDT）：**该增强型版本增加了一个 22 位预分频器，使得延时范围得到扩展，且此范围在工作电压和温度变化时仍保持不变。延时周期请参见**第 31.0 节 “电气特性”**。
- **实时时钟和日历模块 （RTCC）：**RTCC 模块是专为需要长时间维持精确时间的应用而设计的，无需或很少需要 CPU 干预。
- 该模块是一个能自动检测闰年的百年时钟和日历。时钟范围是从 2000 年 1 月 1 日 00:00:00 （午夜）到 2099 年 12 月 31 日 23:59:59。

© 2011 Microchip Technology Inc.

1.3 系列中各器件的详细信息

PIC18F87K22 系列器件具有 64 引脚和 80 引脚两种封装形式。图 1-1 和图 1-2 分别为这两类器件的框图。

这两类器件在以下方面存在差异：

- 闪存程序存储器：
  - PIC18FX5K22 （PIC18F65K22 和 PIC18F85K22）——32 KB
  - PIC18FX6K22 （PIC18F66K22 和 PIC18F86K22）——64 KB
  - PIC18FX7K22 （PIC18F67K22 和 PIC18F87K22）——128 KB
- 数据 RAM：
  - 除 PIC18FX5K22 之外的所有器件——4 KB
  - PIC18FX5K22——2 KB
- I/O 端口：
  - PIC18F6XK22 （64 引脚器件）——7 个双向端口
  - PIC18F8XK22 （80 引脚器件）——9 个双向端口
- CCP 模块：
  - PIC18FX5K22 （PIC18F65K22 和 PIC18F85K22）——5 个 CCP 模块
  - PIC18FX6K22 和 PIC18FX7K22 （PIC18F66K22、 PIC18F86K22、 PIC18F67K22 和 PIC18F87K22）——7 个 CCP 模块
- 定时器模块：
  - PIC18FX5K22 （PIC18F65K22 和 PIC18F85K22）——4 个 8 位定时器 / 计数器和 4 个 16 位定时器 / 计数器
  - PIC18FX6K22 和 PIC18FX7K22 （PIC18F66K22、 PIC18F86K22、 PIC18F67K22 和 PIC18F87K22）——6 个 8 位定时器 / 计数器和 5 个 16 位定时器 / 计数器
- A/D 通道：
  - PIC18F6XK22 （64 引脚器件）——24 路通道
  - PIC18F6XK22 （80 引脚器件）——16 路通道

该系列器件的所有其他特性都是相同的。表 1-1 和表 1-2 汇总了这些特性。

表 1-3 和表 1-4 列出了所有器件的引脚说明。

表 1-1： PIC18F6XK22 系列器件 （64 引脚器件）的特性

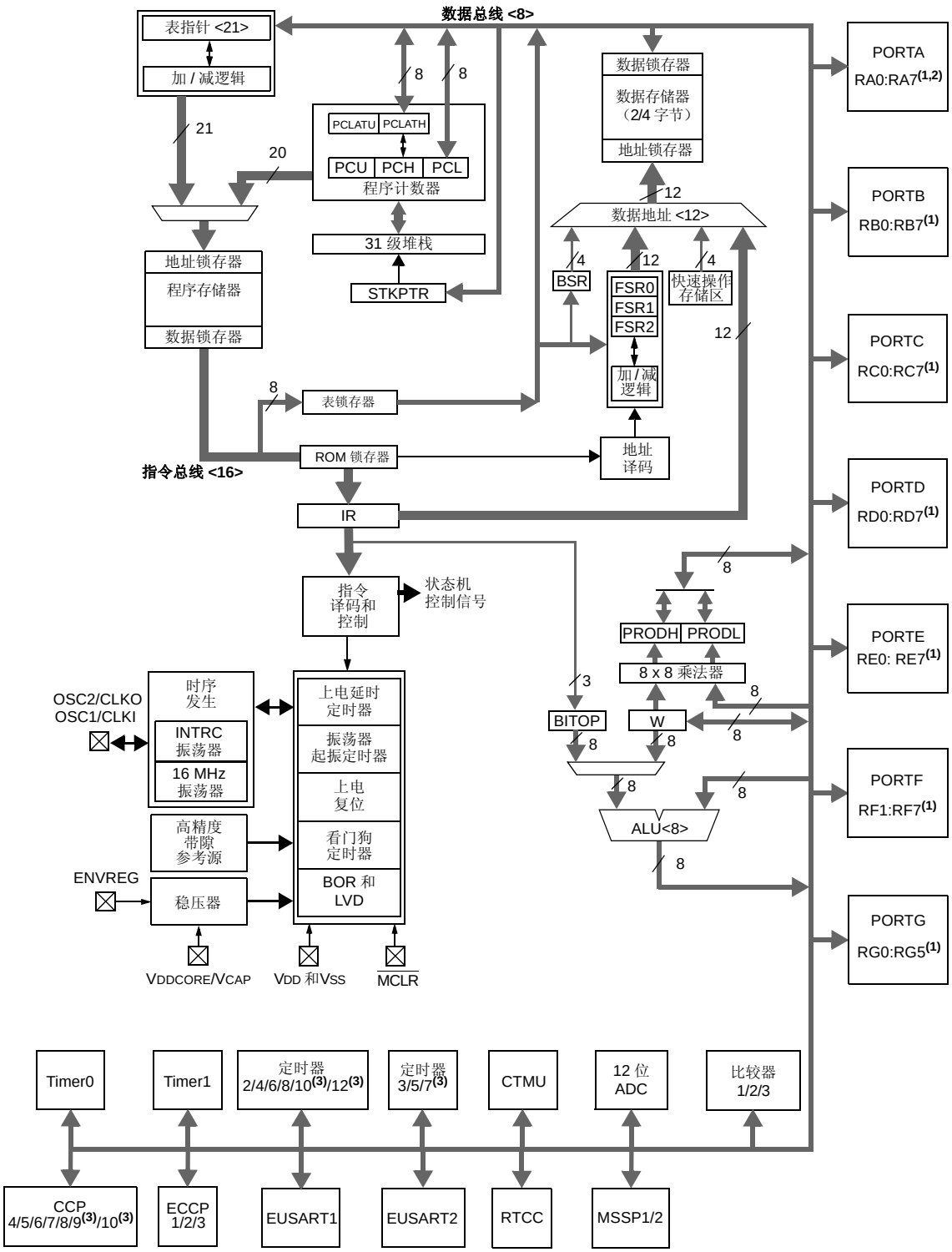
特性	PIC18F65K22	PIC18F66K22	PIC18F67K22
工作频率	DC——64 MHz		
程序存储器 （字节）	32K	64K	128K
程序存储器 （指令）	16,384	32,768	65,536
数据存储器 （字节）	2K	4K	4K
中断源	42	48	
I/O 端口	端口 A、 B、 C、 D、 E、 F 和 G		
并行通信	并行从端口 （PSP）		
定时器	8	11	
比较器	3		
CTMU	有		
RTCC	有		
捕捉 / 比较 /PWM （CCP）模块	5	7	7
增强型 CCP （ECCP）模块	3		
串行通信	两个 MSSP 和两个增强型 USART （EUSART）		
12 位模数转换模块	16 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		
封装	64 引脚 QFN 和 64 引脚 TQFP		

表 1-2： PIC18F8XK22 系列器件 （80 引脚器件）的特性

特性	PIC18F85K22	PIC18F86K22	PIC18F87K22
工作频率	DC——64 MHz		
程序存储器 （字节）	32K	64K	128K
	（使用扩展存储器最多可达 2 MB）		
程序存储器 （指令）	16,384	32,768	65,536
数据存储器 （字节）	2K	4K	4K
中断源	42	48	
I/O 端口	端口 A、 B、 C、 D、 E、 F、 G、 H 和 J		
并行通信	并行从端口 （PSP）		
定时器	8	11	
比较器	3		
CTMU	有		
RTCC	有		
捕捉 / 比较 /PWM （CCP）模块	5	7	7
增强型 CCP （ECCP）模块	3		
串行通信	两个 MSSP 和两个增强型 USART （EUSART）		
12 位模数转换模块	24 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		
封装	80 引脚 TQFP		

© 2011 Microchip Technology Inc.

图 1-1： PIC18F6XK22 （64 引脚）器件框图

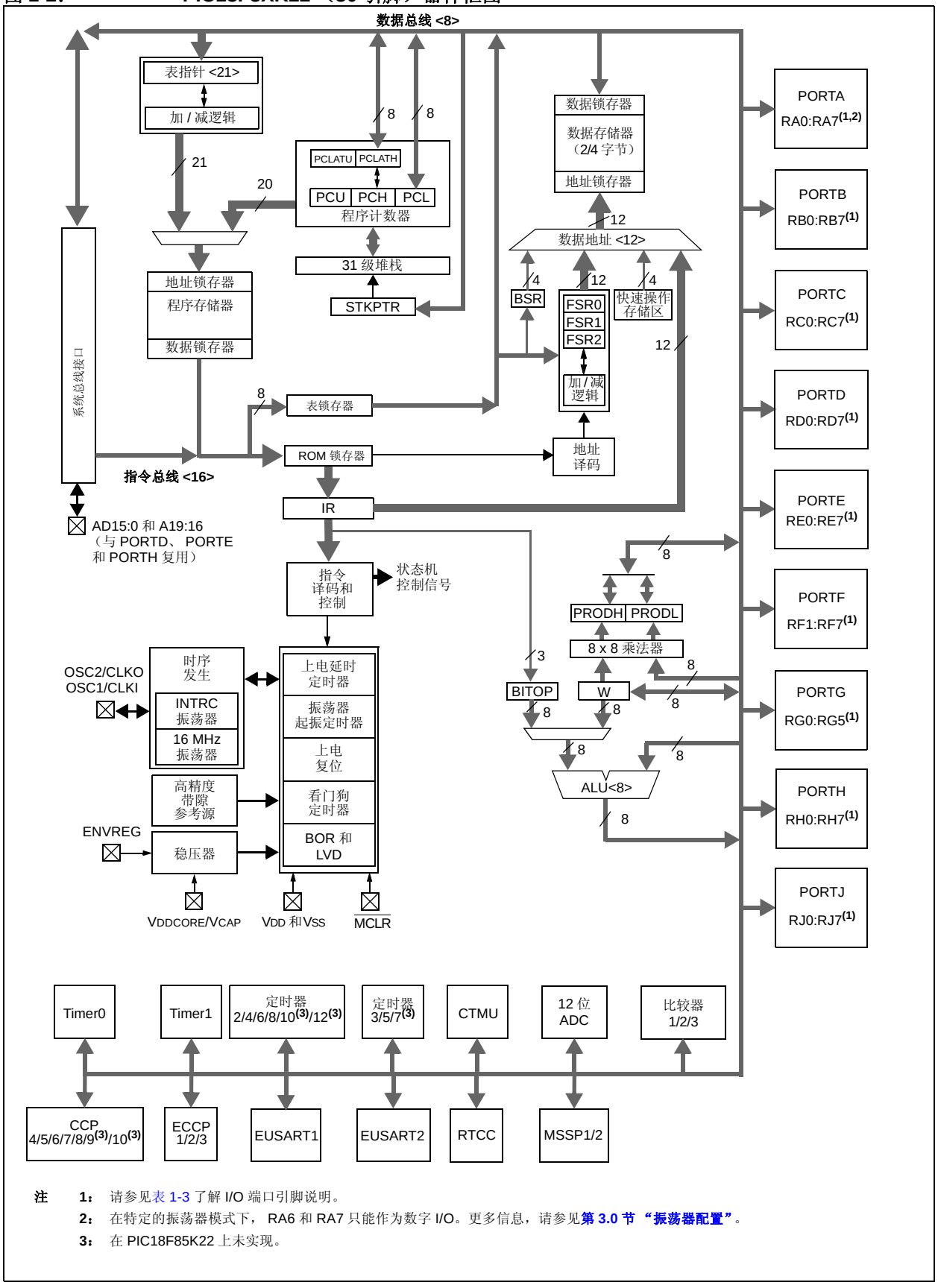


注 1： 请参见表 1-3 了解 I/O 端口引脚说明。

2： 在特定的振荡器模式下， RA6 和 RA7 只能作为数字 I/O。更多信息，请参见第 3.0 节 “振荡器配置”。

3： 在 PIC18F65K22 上未实现。

图 1-2： PIC18F8XK22 （80 引脚）器件框图



注 1： 请参见表 1-3 了解 I/O 端口引脚说明。

2： 在特定的振荡器模式下， RA6 和 RA7 只能作为数字 I/O。更多信息，请参见第 3.0 节 “振荡器配置”。

3： 在 PIC18F85K22 上未实现。

© 2011 Microchip Technology Inc.

表 1-3： PIC18F6XK22 器件 I/O 引脚说明

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
MCLR/RG5	7			主复位 （输入）或编程电压 （输入）。
MCLR		I	ST	此引脚为低电平时器件复位。
RG5		I	ST	通用，仅输入引脚。
OSC1/CLKI/RA7	39			振荡器晶振或外部时钟输入。
OSC1		I	CMOS	振荡器晶振输入。
CLKI		I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能关联。（见相关 OSC1/CLKI 和 OSC2/CLKO 引脚。）
RA7		I/O	TTL	通用 I/O 引脚。
OSC2/CLKO/RA6	40			振荡器晶振或时钟输出。
OSC2		O	—	振荡器晶振输出。在晶振模式下，与晶振或谐振器相连。
CLKO		O	—	在特定振荡器模式下， OSC2 引脚输出 CLKO 信号，该信号是 OSC1 上信号的 4 分频，其频率用于表示指令周期的快慢。
RA6		I/O	TTL	通用 I/O 引脚。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 **1：** 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-3： PIC18F6XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
				PORTA 是双向 I/O 端口。
RA0/AN0/ULPWU	24			
RA0		I/O	TTL	数字 I/O。
AN0		I	Analog	模拟输入 0。
ULPWU		I	Analog	超低功耗唤醒输入。
RA1/AN1	23			
RA1		I/O	TTL	数字 I/O。
AN1		I	Analog	模拟输入 1。
RA2/AN2/VREF-	22			
RA2		I/O	TTL	数字 I/O。
AN2		I	Analog	模拟输入 2。
VREF-		I	Analog	A/D 参考电压 （低电平）输入。
RA3/AN3/VREF+	21			
RA3		I/O	TTL	数字 I/O。
AN3		I	Analog	模拟输入 3。
VREF+		I	Analog	A/D 参考电压 （高电平）输入。
RA4/T0CKI	28			
RA4		I/O	ST	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟输入。
RA5/AN4/T1CKI/T3G/HLVDIN	27			
RA5		I/O	TTL	数字 I/O。
AN4		I	Analog	模拟输入 4。
T1CKI		I	ST	Timer1 时钟输入。
T3G		I	ST	Timer3 外部时钟门控输入。
HLVDIN		I	Analog	高 / 低电压检测输入。
RA6				参见 OSC2/CLKO/RA6 引脚。
RA7				参见 OSC1/CLKI/RA7 引脚。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-3： PIC18F6XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
				PORTB 是双向 I/O 端口。 PORTB 在所有的输入端都可以软件编程为内部弱上拉。
RB0/INT0/FLTO	48			
RB0		I/O	TTL	数字 I/O。
INT0		I	ST	外部中断 0。
FLT0		I	ST	ECCP1/2/3 模块增强型 PWM 故障输入。
RB1/INT1	47			
RB1		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 1。
RB2/INT2/CTED1	46			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
CTED1		I	ST	CTMU 边沿 1 输入。
RB3/INT3/CTED2/ECCP2/P2A	45			
RB3		I/O	TTL	数字 I/O。
INT3		I	ST	外部中断 3。
CTED2		I	ST	CTMU 边沿 2 输入。
ECCP2		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2。
P2A		O	—	增强型 PWM2 输出 A。
RB4/KBI0	44			
RB4		I/O	TTL	数字 I/O。
KBI0		I	TTL	电平变化中断引脚。
RB5/KBI1/T3CKI/T1G	43			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
T3CKI		I	ST	Timer3 时钟输入。
T1G		I	ST	Timer1 外部时钟门控输入。
RB6/KBI2/PGC	42			
RB6		I/O	TTL	数字 I/O。
KBI2		I	TTL	电平变化中断引脚。
PGC		I/O	ST	在线调试器和 ICSP™ 编程时钟引脚。
RB7/KBI3/PGD	37			
RB7		I/O	TTL	数字 I/O。
KBI3		I	TTL	电平变化中断引脚。
PGD		I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-3： PIC18F6XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
				PORTC 是双向 I/O 端口。
RC0/SOSCO/SCLKI	30			
RC0		I/O	ST	数字 I/O。
SOSCO		O	—	SOSC 振荡器输出。
SCLKI		I	ST	数字 SOSC 输入。
RC1/SOSCI/ECCP2/P2A	29			
RC1		I/O	ST	数字 I/O。
SOSCI		I	CMOS	SOSC 振荡器输入。
ECCP2(1)		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
P2A		O	—	增强型 PWM2 输出 A。
RC2/ECCP1/P1A	33			
RC2		I/O	ST	数字 I/O。
ECCP1		I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM1 输出。
P1A		O	—	增强型 PWM1 输出 A。
RC3/SCK1/SCL1	34			
RC3		I/O	ST	数字 I/O。
SCK1		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL1(4)		I/O	I^2C	I^2C™ 模式的同步串行时钟输入 / 输出。
RC4/SDI1/SDA1	35			
RC4		I/O	ST	数字 I/O。
SDI1		I	ST	SPI 数据输入。
SDA1(4)		I/O	I^2C	I^2C 数据 I/O。
RC5/SDO1	36			
RC5		I/O	ST	数字 I/O。
SDO1		O	—	SPI 数据输出。
RC6/TX1/CK1	31			
RC6		I/O	ST	数字 I/O。
TX1		O	—	EUSART 异步发送。
CK1		I/O	ST	EUSART 同步时钟 （见相关的 RX1/DT1 引脚信息）。
RC7/RX1/DT1	32			
RC7		I/O	ST	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 TX1/CK1 引脚信息）。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-3： PIC18F6XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
				PORTD 是双向 I/O 端口。
RD0/PSP0/CTPLS	58			
RD0		I/O	ST	数字 I/O。
PSP0		I/O	TTL	并行从端口数据。
CTPLS		O	—	CTMU 脉冲发生器输出。
RD1/PSP1/T5CKI/T7G	55			
RD1		I/O	ST	数字 I/O。
PSP1		I/O	TTL	并行从端口。
T5CKI		I	ST	Timer5 时钟输入。
T7G		I	ST	Timer7 外部时钟门控输入。
RD2/PSP2	54			
RD2		I/O	ST	数字 I/O。
PSP2		O	TTL	并行从端口。
RD3/PSP3	53			
RD3		I/O	ST	数字 I/O。
PSP3		I/O	TTL	并行从端口。
RD4/PSP4/SDO2	52			
RD4		I/O	ST	数字 I/O。
PSP4		I/O	TTL	并行从端口。
SDO2		O	—	SPI 数据输出。
RD5/PSP5/SDI2/SDA2	51			
RD5		I/O	ST	数字 I/O。
PSP5		I/O	TTL	并行从端口。
SDI2		I	ST	SPI 数据输入。
SDA2		I/O	I^2C	I^2C™ 数据 I/O。
RD6/PSP6/SCK2/SCL2	50			
RD6		I/O	ST	数字 I/O。
PSP6		I/O	TTL	并行从端口。
SCK2		I/O	ST	同步串行时钟。
SCL2(4)		I/O	I^2C	I^2C 模式的同步串行时钟 I/O。
RD7/PSP7/$\overline{SS2}$	49			
RD7		I/O	ST	数字 I/O。
PSP7		I/O	TTL	并行从端口。
$\overline{SS2}$		I	TTL	SPI 从选择输入。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-3： PIC18F6XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
				PORTE 是双向 I/O 端口。
RE0/$\overline{RD}$/P2D	2			
RE0		I/O	ST	数字 I/O。
RD		I	TTL	并行从端口读选通。
P2D		O	—	EECP2 PWM 输出 D。
RE1/$\overline{WR}$/P2C	1			
RE1		I/O	ST	数字 I/O。
WR		I	TTL	并行从端口写选通。
P2C		O	—	ECCP2 PWM 输出 C。
RE2/$\overline{CS}$/P2B/CCP10	64			
RE2		I/O	ST	数字 I/O。
CS		I	TTL	并行从端口片选。
P2B		O	—	ECCP2 PWM 输出 B。
CCP10(3)		I/O	S/T	捕捉 10 输入 / 比较 10 输出 /PWM10 输出。
RE3/P3C/CCP9/REFO	63			
RE3		I/O	ST	数字 I/O。
P3C		O	—	ECCP3 PWM 输出 C。
CCP9(3,4)		I/O	S/T	捕捉 9 输入 / 比较 9 输出 /PWM9 输出。
REFO		O	—	参考时钟输出。
RE4/P3B/CCP8	62			
RE4		I/O	ST	数字 I/O。
P3B		O	—	ECCP3 PWM 输出 B。
CCP8(4)		I/O	S/T	捕捉 8 输入 / 比较 8 输出 /PWM8 输出。
RE5/P1C/CCP7	61			
RE5		I/O	ST	数字 I/O。
P1C		O	—	ECCP1 PWM 输出 C。
CCP7(4)		I/O	S/T	捕捉 7 输入 / 比较 7 输出 /PWM7 输出。
RE6/P1B/CCP6	60			
RE6		I/O	ST	数字 I/O。
P1B		O	—	ECCP1 PWM 输出 B。
CCP6(4)		I/O	S/T	捕捉 6 输入 / 比较 6 输出 /PWM6 输出。
RE7/ECCP2/P2A	59			
RE7		I/O	ST	数字 I/O。
ECCP2(2)		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
P2A		O	—	ECCP2 PWM 输出 A。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-3： PIC18F6XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
				PORTF 是双向 I/O 端口。
RF1/AN6/C2OUT/CTDIN	17			
RF1		I/O	ST	数字 I/O。
AN6		I	Analog	模拟输入 6。
C2OUT		O	—	比较器 2 输出。
CTDIN		I	ST	CTMU 脉冲延时输入。
RF2/AN7/C1OUT	16			
RF2		I/O	ST	数字 I/O。
AN7		I	Analog	模拟输入 7。
C1OUT		O	—	比较器 1 输出。
RF3/AN8/C2INB/CTMUI	15			
RF3		I/O	ST	数字 I/O。
AN8		I	Analog	模拟输入 8。
C2INB		I	Analog	比较器 2 输入 B。
CTMUI		O	—	CTMU 脉冲发生器充电器，用于 C2INB 比较器输入。
RF4/AN9/C2INA	14			
RF4		I/O	ST	数字 I/O。
AN9		I	Analog	模拟输入 9。
C2INA		I	Analog	比较器 2 输入 A。
RF5/AN10/CVREF/C1INB	13			
RF5		I/O	ST	数字 I/O。
AN10		I	Analog	模拟输入 10。
CVREF		O	Analog	比较器参考电压输出。
C1INB		I	Analog	比较器 1 输入 B。
RF6/AN11/C1INA	12			
RF6		I/O	ST	数字 I/O。
AN11		I	Analog	模拟输入 11。
C1INA		I	Analog	比较器 1 输入 A。
RF7/AN5/$\overline{SS1}$	11			
RF7		I/O	ST	数字 I/O。
AN5		O	Analog	模拟输入 5。
$\overline{SS1}$		I	TTL	SPI 从选择输入。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-3： PIC18F6XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
				PORTG 是双向 I/O 端口。
RG0/ECCP3/P3A	3			
RG0		I/O	ST	数字 I/O。
ECCP3		I/O	ST	捕捉 3 输入 / 比较 3 输出 /PWM3 输出。
P3A		O	—	ECCP3 PWM 输出 A。
RG1/TX2/CK2/AN19/C3OUT	4			
RG1		I/O	ST	数字 I/O。
TX2		O	—	EUSART 异步发送。
CK2		I/O	ST	EUSART 同步时钟 （见相关的 RX2/DT2 引脚信息）。
AN19		I	Analog	模拟输入 19。
C3OUT		O	—	比较器 3 输出。
RG2/RX2/DT2/AN18/C3INA	5			
RG2		I/O	ST	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）。
AN18		I	Analog	模拟输入 18。
C3INA		I	Analog	比较器 3 输入 A。
RG3/CCP4/AN17/P3D/C3INB	6			
RG3		I/O	ST	数字 I/O。
CCP4		I/O	S/T	捕捉 4 输入 / 比较 4 输出 /PWM4 输出。
AN17		I	Analog	模拟输入 18。
P3D		O	—	ECCP3 PWM 输出 D。
C3INB		I	Analog	比较器 3 输入 B。
RG4/RTCC/T7CKI/T5G/CCP5/AN16/P1D/C3INC	8			
RG4		I/O	ST	数字 I/O。
RTCC		O	—	RTCC 输出
T7CKI(3)		I	ST	Timer7 时钟输入。
T5G		I	ST	Timer5 外部时钟门输入。
CCP5		I/O	ST	捕捉 5 输入 / 比较 5 输出 /PWM5 输出。
AN16		I	Analog	模拟输入 16。
P1D		O	—	ECCP1 PWM 输出 D。
C3INC		I	Analog	比较器 3 输入 C。
RG5	7			参见 $\overline{\text{MCLR}}$/RG5 引脚。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-3： PIC18F6XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 QFN/TQFP	引脚类型	缓冲器类型	说明
VSS	9, 25, 41, 56	P	—	逻辑电路和 I/O 引脚的参考地。
VDD	26, 38, 57	P	—	逻辑电路和 I/O 引脚的正电源。
AVSS	20	P	—	模拟模块的参考地。
AVDD	19	P	—	模拟模块的正电源。
ENVREG	18	I	ST	使能片上稳压器。
VDDCORE/VCAP VDDCORE VCAP	10	P	—	内核逻辑电源或外部滤波器电容连接端。 外部滤波器电容连接端 （稳压器使能 / 禁止）。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 VDD）

注 **1：** 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-4： PIC18F8XK22 器件 I/O 引脚说明

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
$\overline{\text{MCLR}}$/RG5	9			主复位 （输入）或编程电压 （输入）。
$\overline{\text{RG5}}$		I	ST	此引脚为低电平时器件复位。
MCLR		I	ST	通用，仅输入引脚。
OSC1/CLKI/RA7	49			振荡器晶振或外部时钟输入。
OSC1		I	CMOS	振荡器晶振输入。
CLKI		I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能关联。（见相关 OSC1/CLKI 和 OSC2/CLKO 引脚。）
RA7		I/O	TTL	通用 I/O 引脚。
OSC2/CLKO/RA6	50			振荡器晶振或时钟输出。
OSC2		O	—	振荡器晶振输出。在晶振模式下，与晶振或谐振器相连。
CLKO		O	—	在特定振荡器模式下， OSC2 引脚输出 CLKO 信号，该信号是 OSC1 上信号的 4 分频，其频率用于表示指令周期的快慢。
RA6		I/O	TTL	通用 I/O 引脚。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 **1：** 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTA 是双向 I/O 端口。
RA0/AN0/ULPWU	30			
RA0		I/O	TTL	数字 I/O。
AN0		I	Analog	模拟输入 0。
ULPWU		I	Analog	超低功耗唤醒输入。
RA1/AN1	29			
RA1		I/O	TTL	数字 I/O。
AN1		I	Analog	模拟输入 1。
RA2/AN2/VREF-	28			
RA2		I/O	TTL	数字 I/O。
AN2		I	Analog	模拟输入 2。
VREF-		I	Analog	A/D 参考电压 （低电平）输入。
RA3/AN3/VREF+	27			
RA3		I/O	TTL	数字 I/O。
AN3		I	Analog	模拟输入 3。
VREF+		I	Analog	A/D 参考电压 （高电平）输入。
RA4/T0CKI	34			
RA4		I/O	ST	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟输入。
RA5/AN4/T1CKI/T3G/HLVDIN	33			
RA5		I/O	TTL	数字 I/O。
AN4		I	Analog	模拟输入 4。
T1CKI		I	ST	Timer1 时钟输入。
T3G		I	ST	Timer3 外部时钟门输入。
HLVDIN		I	Analog	高 / 低电压检测输入。
RA6				参见 OSC2/CLKO/RA6 引脚。
RA7				参见 OSC1/CLKI/RA7 引脚。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTB 是双向 I/O 端口。 PORTB 在所有的输入端都可以软件编程为内部弱上拉。
RB0/INT0/FLT0	58			
RB0		I/O	TTL	数字 I/O。
INT0		I	ST	外部中断 0。
FLT0		I	ST	ECCP1/2/3 模块增强型 PWM 故障输入。
RB1/INT1	57			
RB1		I/O	TTL	数字 I/O。
INT1		I	ST	外部中断 1。
RB2/INT2/CTED1	56			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
CTED1		I	ST	CTMU 边沿 1 输入。
RB3/INT3/CTED2/ECCP2/P2A	55			
RB3		I/O	TTL	数字 I/O。
INT3		I	ST	外部中断 3。
CTED2		I	ST	CTMU 边沿 2 输入。
ECCP2		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
P2A		O	ST	增强型 PWM2 输出 A。
RB4/KBI0	54			
RB4		I/O	TTL	数字 I/O。
KBI0		I	TTL	电平变化中断引脚。
RB5/KBI1/T3CKI/T1G	53			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
T3CKI		I	ST	Timer3 时钟输入。
T1G		I	ST	Timer1 外部时钟门控输入。
RB6/KBI2/PGC	52			
RB6		I/O	TTL	数字 I/O。
KBI2		I	TTL	电平变化中断引脚。
PGC		I/O	ST	在线调试器和 ICSP™ 编程时钟引脚。
RB7/KBI3/PGD	47			
RB7		I/O	TTL	数字 I/O。
KBI3		I	TTL	电平变化中断引脚。
PGD		I/O	ST	在线调试器和 ICSP 编程数据引脚。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTC 是双向 I/O 端口。
RC0/SOSCO/SCKLI	36			
RC0		I/O	ST	数字 I/O。
SOSCO		O	—	SOSC 振荡器输出。
SCKLI		I	ST	数字 SOSC 输入。
RC1/SOSCI/ECCP2/P2A	35			
RC1		I/O	ST	数字 I/O。
SOSCI		I	CMOS	SOSC 振荡器输入。
ECCP2(1)		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
P2A		O	—	增强型 PWM2 输出 A。
RC2/ECCP1/P1A	43			
RC2		I/O	ST	数字 I/O。
ECCP1		I/O	ST	捕捉 1 输入 / 比较 1 输出 /PWM1 输出。
P1A		O	—	增强型 PWM1 输出 A。
RC3/SCK1/SCL1	44			
RC3		I/O	ST	数字 I/O。
SCK1		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL1		I/O	I^2C	I^2C™ 模式的同步串行时钟输入 / 输出。
RC4/SDI1/SDA1	45			
RC4		I/O	ST	数字 I/O。
SDI1		I	ST	SPI 数据输入。
SDA1		I/O	I^2C	I^2C 数据 I/O。
RC5/SDO1	46			
RC5		I/O	ST	数字 I/O。
SDO1		O	—	SPI 数据输出。
RC6/TX1/CK1	37			
RC6		I/O	ST	数字 I/O。
TX1		O	—	EUSART 异步发送。
CK1		I/O	ST	EUSART 同步时钟 （见相关的 RX1/DT1 引脚信息）。
RC7/RX1/DT1	38			
RC7		I/O	ST	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 TX1/CK1 引脚信息）。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTD 是双向 I/O 端口。
RD0/PSP0/CTPLS/AD0	72			
RD0		I/O	ST	数字 I/O。
PSP0(4)		I/O	TTL	并行从端口数据。
CTPLS		O	ST	CTMU 脉冲发生器输出。
AD0		I/O	TTL	外部存储器地址 / 数据 0。
RD1/T5CKI/T7G/PSP1/AD1	69			
RD1		I/O	ST	数字 I/O。
T5CKI		I	ST	Timer5 时钟输入。
T7G		I	ST	Timer7 外部时钟门控输入。
PSP1(4)		I/O	TTL	并行从端口数据。
AD1		I/O	TTL	外部存储器地址 / 数据 1。
RD2/PSP2/AD2	68			
RD2		I/O	ST	数字 I/O。
PSP2(4)		I/O	TTL	并行从端口数据。
AD2		I/O	TTL	外部存储器地址 / 数据 2。
RD3/PSP3/AD3	67			
RD3		I/O	ST	数字 I/O。
PSP3(4)		I/O	TTL	并行从端口数据。
AD3		I/O	TTL	外部存储器地址 / 数据 3。
RD4/SDO2/PSP4/AD4	66			
RD4		I/O	ST	数字 I/O。
SDO2		O	—	SPI 数据输出。
PSP4(4)		I/O	TTL	并行从端口数据。
AD4		I/O	TTL	外部存储器地址 / 数据 4。
RD5/SDI2/SDA2/PSP5/AD5	65			
RD5		I/O	ST	数字 I/O。
SDI2		I	ST	SPI 数据输入。
SDA2		I/O	I^2C	I^2C™ 数据 I/O。
PSP5(4)		I/O	TTL	并行从端口数据。
AD5		I/O	TTL	外部存储器地址 / 数据 5。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
RD6/SCK2/SCL2/PSP6/AD6	64			
RD6		I/O	ST	数字 I/O。
SCK2		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL2		I/O	I^2C	I^2C™ 模式的同步串行时钟输入 / 输出。
PSP6(4)		I/O	TTL	并行从端口数据。
AD6		I/O	TTL	外部存储器地址 / 数据 6。
RD7/$\overline{SS2}$/PSP7/AD7	63			
RD7		I/O	ST	数字 I/O。
$\overline{SS2}$		I	TTL	SPI 从选择输入。
PSP7(4)		I/O	TTL	并行从端口数据。
AD7		I/O	TTL	外部存储器地址 / 数据 7。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 VDD）

注 **1**： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTE 是双向 I/O 端口。
RE0/P2D/$\overline{RD}$/AD8	4			
RE0		I/O	ST	数字 I/O。
P2D		O	—	ECCP2 PWM 输出 D。
$\overline{RD}$(4)		I	TTL	并行从端口读选通。
AD8		I/O	TTL	外部存储器地址 / 数据 8。
RE1/P2C/$\overline{WR}$/AD9	3			
RE1		I/O	ST	数字 I/O。
P2C		O	—	ECCP2 PWM 输出 C。
$\overline{WR}$(4)		I	TTL	并行从端口写选通。
AD9		I/O	TTL	外部存储器地址 / 数据 9。
RE2/P2B/CCP10/$\overline{CS}$/AD10	78			
RE2		I/O	ST	数字 I/O。
P2B		O	ST	ECCP2 PWM 输出 B。
CCP10(3)		I/O	ST	捕捉 10 输入 / 比较 10 输出 /PWM10 输出。
$\overline{CS}$(4)		I	TTL	并行从端口片选。
AD10		I/O	TTL	外部存储器地址 / 数据 10。
RE3/P3C/CCP9/REFO AD11	77			
RE3		I/O	ST	数字 I/O。
P3C		O	—	ECCP3 PWM 输出 C。
CCP9(3,5)		I/O	S/T	捕捉 9 输入 / 比较 9 输出 /PWM9 输出。
REFO		O	—	参考时钟输出。
AD11		I/O	TTL	外部存储器地址 / 数据 11。
RE4/P3B/CCP8/AD12	76			
RE4		I/O	ST	数字 I/O。
P3B		O	—	ECCP4 PWM 输出 B。
CCP8(5)		I/O	ST	捕捉 8 输入 / 比较 8 输出 /PWM8 输出。
AD12		I/O	TTL	外部存储器地址 / 数据 12。
RE5/P1C/CCP7/AD13	75			
RE5		I/O	ST	数字 I/O。
P1C		O	—	ECCP1 PWM 输出 C。
CCP7(5)		I/O	ST	捕捉 7 输入 / 比较 7 输出 /PWM7 输出。
AD13		I/O	TTL	外部存储器地址 / 数据 13。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
RE6/P1B/CCP6/AD14	74			
RE6		I/O	ST	数字 I/O。
P1B		O	—	ECCP1 PWM 输出 B。
CCP6(5)		I/O	ST	捕捉 6 输入 / 比较 6 输出 /PWM6 输出。
AD14		I/O	ST	外部存储器地址 / 数据 14。
RE7/ECCP2/P2A/AD15	73			
RE7		I/O	ST	数字 I/O。
ECCP2(2)		I/O	ST	捕捉 2 输入 / 比较 2 输出 /PWM2 输出。
P2A		O	—	ECCP2 PWM 输出 A。
AD15		I/O	ST	外部存储器地址 / 数据 15。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTF 是双向 I/O 端口。
RF1/AN6/C2OUT/CTDIN	23			
RF1		I/O	ST	数字 I/O。
AN6		I	Analog	模拟输入 6。
C2OUT		O	—	比较器 2 输出。
CTDIN		I	ST	CTMU 脉冲延时输入。
RF2/AN7/C1OUT	18			
RF2		I/O	ST	数字 I/O。
AN7		I	Analog	模拟输入 7。
C1OUT		O	—	比较器 1 输出。
RF3/AN8/C2INB/CTMUI	17			
RF3		I/O	ST	数字 I/O。
AN8		I	Analog	模拟输入 8。
C2INB		I	Analog	比较器 2 输入 B。
CTMUI		O	—	CTMU 脉冲发生器充电器，用于 C2INB 比较器输入。
RF4/AN9/C2INA	16			
RF4		I/O	ST	数字 I/O。
AN9		I	Analog	模拟输入 9。
C2INA		I	Analog	比较器 2 输入 A。
RF5/AN10/C1INB	15			
RF5		I/O	ST	数字 I/O。
AN10		I	Analog	模拟输入 10。
C1INB		I	Analog	比较器 1 输入 B。
RF6/AN11/C1INA	14			
RF6		I/O	ST	数字 I/O。
AN11		I	Analog	模拟输入 11。
C1INA		I	Analog	比较器 1 输入 A。
RF7/AN5/$\overline{SS1}$	13			
RF7		I/O	ST	数字 I/O。
AN5		O	Analog	模拟输入 5。
$\overline{SS1}$		I	ST	SPI 从选择输入。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTG 是双向 I/O 端口。
RG0/ECCP3/P3A	5			
RG0		I/O	ST	数字 I/O。
ECCP3		I/O	ST	捕捉 3 输入 / 比较 3 输出 /PWM3 输出。
P3A		O	—	ECCP3 PWM 输出 A。
RG1/TX2/CK2/AN19/C3OUT	6			
RG1		I/O	ST	数字 I/O。
TX2		O	—	EUSART 异步发送。
CK2		I/O	ST	EUSART 同步时钟 （见相关的 RX2/DT2 引脚信息）。
AN19		I	Analog	模拟输入 19。
C3OUT		O	—	比较器 3 输出。
RG2/RX2/DT2/AN18/C3INA	7			
RG2		I/O	ST	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）。
AN18		I	Analog	模拟输入 18。
C3INA		I	Analog	比较器 3 输入 A。
RG3/CCP4/AN17/P3D/C3INB	8			
RG3		I/O	ST	数字 I/O。
CCP4		I/O	ST	捕捉 4 输入 / 比较 4 输出 /PWM4 输出。
AN17		I	Analog	模拟输入 17。
P3D		O	—	ECCP3 PWM 输出 D。
C3INB		I	Analog	比较器 3 输入 B。
RG4/RTCC/T7CKI/T5G/CCP5/AN16/P1D/C3INC	10			
RG4		I/O	ST	数字 I/O。
RTCC		O	—	RTCC 输出。
T7CKI(3)		I	ST	Timer7 时钟输入。
T5G		I	ST	Timer5 外部时钟门控输入。
CCP5		I/O	ST	捕捉 5 输入 / 比较 5 输出 /PWM5 输出。
AN16		I	Analog	模拟输入 16。
P1D		O	—	ECCP1 PWM 输出 D。
C3INC		I	Analog	比较器 3 输入 C。
RG5	9			参见 $\overline{\text{MCLR}}$/RG5 引脚。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTH 是双向 I/O 端口。
RH0/AN23/A16	79			
RH0		I/O	ST	数字 I/O。
AN23		I	Analog	模拟输入 23。
A16		I/O	TTL	外部存储器地址 / 数据 16。
RH1/AN22/A17	80			
RH1		I/O	ST	数字 I/O。
AN22		I	Analog	模拟输入 22。
A17		I/O	TTL	外部地址 / 数据 17。
RH2/AN21/A18	1			
RH2		I/O	ST	数字 I/O。
AN21		I	Analog	模拟输入 21。
A18		I/O	TTL	外部地址 / 数据 18。
RH3/AN20/A19	2			
RH3		I/O	ST	数字 I/O。
AN20		I	Analog	模拟输入 20。
A19		I/O	TTL	外部地址 / 数据 19。
RH4/CCP9/P3C/AN12/C2INC	22			
RH4		I/O	ST	数字 I/O。
CCP9(3,5)		I/O	ST	捕捉 9 输入 / 比较 9 输出 /PWM9 输出。
P3C		O	—	ECCP3 PWM 输出 C。
AN12		I	Analog	模拟输入 12。
C2INC		I	Analog	比较器 2 输入 C。
RH5/CCP8/P3B/AN13/C2IND	21			
RH5		I/O	ST	数字 I/O。
CCP8(5)		I/O	ST	捕捉 8 输入 / 比较 8 输出 /PWM8 输出。
P3B		O	—	ECCP3 PWM 输出 B。
AN13		I	Analog	模拟输入 13。
C2IND		I	Analog	比较器 1 输入 D。
RH6/CCP7/P1C/AN14/C1INC	20			
RH6		I/O	ST	数字 I/O。
CCP7(5)		I/O	ST	捕捉 7 输入 / 比较 7 输出 /PWM7 输出。
P1C		O	—	ECCP1 PWM 输出 C。
AN14		I	Analog	模拟输入 14。
C1INC		I	Analog	比较器 1 输入 C。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
RH7/CCP6/P1B/AN15	19			
RH7		I/O	ST	数字 I/O。
CCP6(5)		I/O	ST	捕捉 6 输入 / 比较 6 输出 /PWM6 输出。
P1B		O	—	ECCP1 PWM 输出 B。
AN15		I	Analog	模拟输入 15。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

表 1-4： PIC18F8XK22 器件 I/O 引脚说明 （续）

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
				PORTJ 是双向 I/O 端口。
RJ0/ALE	62			
RJ0		I/O	ST	数字 I/O。
ALE		O	—	外部存储器地址锁存器使能引脚。
RJ1/$\overline{OE}$	61			
RJ1		I/O	ST	数字 I/O。
$\overline{OE}$		O	—	外部存储器输出使能引脚。
RJ2/$\overline{WRL}$	60			
RJ2		I/O	ST	数字 I/O。
$\overline{WRL}$		O	—	外部存储器写低电平控制引脚。
RJ3/$\overline{WRH}$	59			
RJ3		I/O	ST	数字 I/O。
$\overline{WRH}$		O	—	外部存储器高电平控制引脚。
RJ4/BA0	39			
RJ4		I/O	ST	数字 I/O。
BA0		O	—	外部存储器字节地址 0 控制引脚。
RJ5/$\overline{CE}$	40			
RJ5		I/O	ST	数字 I/O。
$\overline{CE}$		O	—	外部存储器芯片使能控制引脚。
RJ6/$\overline{LB}$	41			
RJ6		I/O	ST	数字 I/O。
$\overline{LB}$		O	—	外部存储器低字节控制引脚。
RJ7/$\overline{UB}$	42			
RJ7		I/O	ST	数字 I/O。
$\overline{UB}$		O	—	外部存储器高字节控制引脚。
VSS	11, 31, 51, 70	P	—	逻辑电路和 I/O 引脚的参考地。
VDD	32, 48, 71	P	—	逻辑电路和 I/O 引脚的正电源。
AVSS	26	P	—	模拟模块的参考地。
AVDD	25	P	—	模拟模块的正电源。
ENVREG	24	I	ST	使能片上稳压器。
VDDCORE/VCAP	12			内核逻辑电源或外部滤波器电容连接端。
VDDCORE				
VCAP		P	—	外部滤波器电容连接端 （稳压器使能 / 禁止）。

图注： TTL = TTL 兼容输入
ST = CMOS 电平的施密特触发器输入
I = 输入
P = 电源
I^2C = I^2C™/SMBus
CMOS = CMOS 兼容输入或输出
Analog = 模拟输入
O = 输出
OD = 漏极开路 （无 P 型二极管连接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。
2： 当 CCP2MX 配置位清零时，对 ECCP2 进行其他分配。
3： 在 PIC18F65K22 和 PIC18F85K22 器件上不可用。
4： PSP 仅可用于单片机模式。
5： CC6、 CCP7、 CCP8 和 CCP9 引脚位置取决于 ECCPMX 配置位 （CONFIG3H<1>）的设置。

© 2011 Microchip Technology Inc.

2.0 PIC18FXXKXX 单片机入门指南

2.1 基本连接要求

在使用 PIC18F87K22 系列 8 位单片机进行开发之前，某些器件引脚的连接必须满足以下最基本的要求。

以下引脚必须始终处于连接状态：

- 所有 VDD 和 VSS 引脚 （见**第 2.2 节 “电源引脚”**）
- 所有 AVDD 和 AVSS 引脚，而与是否使用模拟器件功能无关 （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”**）
- ENVREG （如果存在）和 VCAP/VDDCORE 引脚 （见**第 2.4 节 “稳压器引脚 （ENVREG 和 VCAP/VDDCORE）”**）。

如果在最终应用中使用了以下引脚，那么也必须连接这些引脚：

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

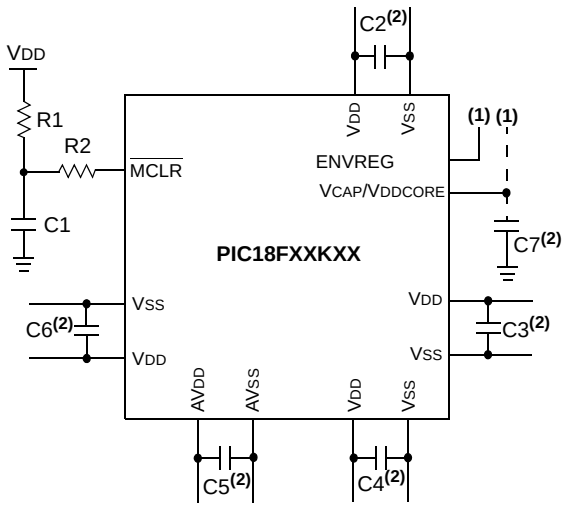
此外，可能需要以下引脚：

- 对模拟模块采用外部参考电压时使用的 VREF+/VREF- 引脚

注：	无论是否使用模拟模块，AVDD 和 AVSS 引脚都必须始终处于已连接状态。

图 2-1 中给出了必需的连接。

图 2-1：建议的最低限度连接



要点 （所有值都是建议值）：

C1 到 C6：0.1 µF、 20V 的陶瓷电容

R1：10 kΩ

R2：100Ω 至 470Ω

注 1： 请参见**第 2.4 节 “稳压器引脚 （ENVREG 和 VCAP/VDDCORE）”**了解 ENVREG 引脚连接说明。

2： 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。其他器件所具有的对数可能有所不同；应相应调整去耦电容数。

2.2 电源引脚

2.2.1 去耦电容

需要在每对电源引脚（如 VDD 与 VSS，以及 AVDD 与 AVSS）上使用去耦电容。

使用去耦电容时，需考虑以下条件：

- **电容的值和类型：**建议使用 0.1 µF（100 nF）、10-20V 的电容。应使用谐振频率在 200 MHz 及更高范围内的低 ESR 电容。建议使用陶瓷电容。
- **印刷电路板上的放置：**去耦电容应尽可能靠近相应的引脚放置。建议将电容放在电路板上与器件相同的一侧。如果空间有限，可使用过孔将电容放到 PCB 的另一层上；但是，需要确保从引脚到电容的走线长度不超过 0.25 英寸（6 mm）。
- **处理高频噪声：**如果电路板存在高达几十兆赫兹的高频噪声，请在上述去耦电容旁并联一个陶瓷类型的辅助电容。辅助电容值的范围为 0.001 µF 至 0.01 µF。请将这个辅助电容靠近每个主去耦电容放置。在高速电路设计中，请考虑在尽可能靠近电源和接地引脚的地方布置一对电容（例如，一个 0.1 µF 的电容与一个 0.001 µF 的电容并联构成一对）。
- **最大程度提高性能：**在从电源电路布置电路板时，请首先连接电源并将走线返回到去耦电容，然后再连接到器件引脚。这可以确保去耦电容在电源链中处于第一位。保持电容和电源引脚之间走线长度尽可能短也同样重要，因为这可以减少 PCB 的走线感抗。

2.2.2 槽路电容

在电源走线长度大于 6 英寸的电路板上，建议对包含单片机在内的集成电路使用槽路电容，以提供本地电源。槽路电容的值应根据以下因素确定：连接电源和器件的走线的电阻值以及应用中器件消耗的最大电流。也就是说，选择槽路电容使之满足器件的可接受的电压骤降要求。典型值范围为 4.7 µF 到 47 µF。

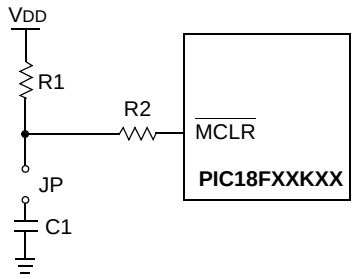
2.3 主复位（$\overline{\text{MCLR}}$）引脚

$\overline{\text{MCLR}}$ 引脚提供两个特定的器件功能：器件复位及器件编程与调试。如果在最终应用中不需要编程和调试，则只需直接将该引脚连接到 VDD。为帮助增加应用的电阻以避免因电压骤降而意外复位，可能需要添加其他元件。图 2-1 给出了典型的配置。可根据应用的要求实现其他电路设计。

在编程和调试期间，必须考虑可能添加到引脚的电阻和电容。器件编程器和调试器驱动 MCLR 引脚。因此，不允许对特殊电压电平（VIH 和 VIL）以及快速信号变化造成不良影响。这就需要根据应用和 PCB 的要求调整 R1 和 C1 的具体值。例如，建议在编程和调试操作期间，通过使用跳线将电容 C1 和 $\overline{\text{MCLR}}$ 引脚隔离（图 2-2）。正常的运行时操作不需要这样的跳线。

与 $\overline{\text{MCLR}}$ 引脚相关的任何元件应放置在距离该引脚 0.25 英寸（6 mm）的范围内。

图 2-2： **$\overline{\text{MCLR}}$ 引脚连接示例**



注 1： 建议 R1 ≤ 10 kΩ。建议的起始值为 10 kΩ。确保满足 $\overline{\text{MCLR}}$ 引脚的 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）或过度电应力（Electrical Overstress，EOS）导致 $\overline{\text{MCLR}}$ 引脚损坏。确保满足 $\overline{\text{MCLR}}$ 引脚的 VIH 和 VIL 规范。

© 2011 Microchip Technology Inc.

2.4 稳压器引脚 （ENVREG 和 VCAP/VDDCORE）

片上稳压器使能引脚 ENVREG 必须始终直接连接到电源电压或接地。将ENVREG连接到VDD会使能稳压器，而将其接地会禁止稳压器。连接和使用片上稳压器的详情，请参见**第 28.3 节 “片上稳压器”**。

使能稳压器后， VCAP/VDDCORE 引脚上需要连接一个低 ESR （< 5Ω）的电容来稳定稳压器输出电压。VCAP/VDDCORE 引脚不能连接到 VDD，并且必须使用一个 10 µF 的电容接地。电容类型可以是陶瓷电容或者钽电容。在表 2-1 中列出了合适的电容示例。可以使用具有同等规格的电容。

设计人员可以使用图 2-3 来计算备选电容的等效 ESR。

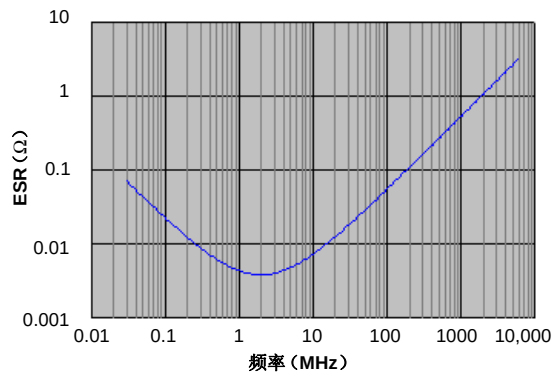
建议走线长度不超过 0.25 英寸 （6 mm）。更多消息，请参见**第 31.0 节 “电气特性”**。

禁止稳压器时， VCAP/VDDCORE 引脚必须连接到 VDDCORE 大小的电源电压。有关 VDD 和 VDDCORE 的更多信息，请参见**第 31.0 节 “电气特性”**。

某些PIC18FXXKXX系列器件或某个系列中的部分器件不提供使能或禁止片上稳压器的选项：

- 一些器件 （名为 PIC18**LF**XXKXX）的稳压器永久禁止。
  这些器件没有 ENVREG 引脚，要求在 VCAP/VDDCORE 引脚上连接一个 0.1 µF 的电容。这些器件的 VDD 电平必须符合**第 31.0 节 “电气特性”**中参数 D001 的 “稳压器禁止”规范。
- 一些器件的稳压器永久使能。
  这些器件同样没有 ENVREG 引脚。需要在 VCAP/VDDCORE 引脚上连接一个 10 µF 的电容。

图 2-3： 建议 VCAP 的频率与 ESR 性能关系



注： 给出的是在 25°C、 0V 直流偏置电压下测量的典型数据。

表 2-1： 合适的等效电容

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

2.4.1 陶瓷电容注意事项

近年来，大容量、低电压的贴片陶瓷电容，在容量达到十几微法时非常具成本效益。低 ESR、体积小巧以及其他特性使得陶瓷电容在许多类型的应用中相当具有吸引力。

陶瓷电容适合用于这类单片机的内部稳压器。然而，在选择电容的时候需要多加注意，以确保它能在应用的期望工作范围内维持足够的电容。

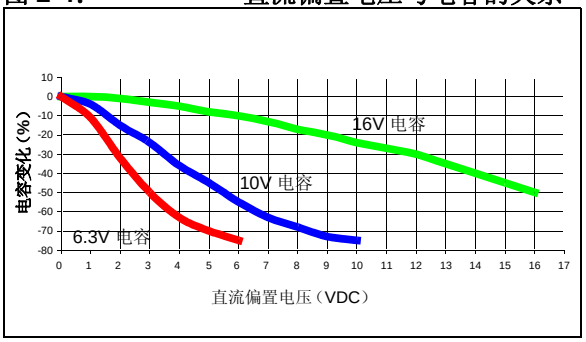
典型的价格低廉的 10 µF 陶瓷电容提供有 X5R，X7R 和 Y5V 电介质等级（也提供有其他类型，但是不太普遍）。这些类型的电容的初始容差规范经常指定为 ±10% 到 ±20% （X5R 和 X7R），或者 -20%/+80% （Y5V）。然而，这些电容在应用电路中提供的有效电容值，还取决于其他因素，例如应用的直流偏置电压和温度。因此，总的在线容差比初始容差规范要宽泛很多。

X5R 和 X7R 电容通常显示出令人满意的温度稳定性（例如：在宽温度范围内为 ±15%，但是关于具体的规范请查询制造商的数据手册）。然而， Y5V 电容通常具有极大的温度容差参数 +22%/-82%。由于此极大的温度容差，标称值为 10 µF 的 Y5V 型电容也许不能提供足够的总电容来满足最低内部稳压器稳定性和瞬态响应需求。因此，如果应用必须在宽温度范围内工作时，不建议将 Y5V 电容用于内部稳压器。

除了温度容差外，根据施加到电容的直流电压量，大容量的陶瓷电容的有效电容会有很大的差异。这个影响非常重要，但是经常被忽视或者没有被记录。

图 2-4 中显示了 X7R 型和 Y5V 型电容的典型直流偏置电压与电容的关系。

图 2-4：直流偏置电压与电容的关系



当选择用于内部稳压器的陶瓷电容时，为了能够让工作电压在最大额定电容电压值中占较小比例，建议选择高压值电容。例如，对于 2.5V 的内核电压应选择额定值为 16V 的陶瓷电容。表 2-1 中给出了建议使用的电容。

2.5 ICSP 引脚

PGC 和 PGD 引脚用于在线串行编程 （ICSP™）和调试。建议保持 ICSP 连接器与器件上 ICSP 引脚之间的走线长度尽可能短。如果预期 ICSP 连接器上会发生 ESD 事件，建议使用一个串联电阻，且电阻值在几十欧姆范围内，不要超过 100Ω。

建议不要在 PGC 和 PGD 引脚上使用上拉电阻、串联二极管和电容，因为它们会干扰编程器 / 调试器与器件之间的通信。如果应用需要这类分立元件，应在编程和调试期间将它们从电路中除去。或者，参阅相关器件闪存编程规范中的交流 / 直流特性和时序要求信息，以了解关于容性负载限制以及引脚高输入电压 （V_{IH}）和低输入电压 （V_{IL}）要求的信息。

对于器件仿真，请确保在程序中设定的 “通信通道选择”（即，PGCx/PGDx 引脚）与 ICSP 到 Microchip 调试器 / 仿真器工具的物理连接一致。

关于可用 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”**。

© 2011 Microchip Technology Inc.

2.6 外部振荡器引脚

许多单片机都至少有两个振荡器选项：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”**）。

振荡器电路应放到电路板上与器件相同的一侧。将振荡器电路靠近相关振荡器引脚放置，且电路元件与引脚之间的距离不要超过 0.5 英寸（12 mm）。负载电容应在电路板的同一侧挨着振荡器本身放置。

在振荡器电路周围设置接地敷铜区，将其与周围电路隔离。接地敷铜区应直接连接到 MCU 地。不要在接地敷铜区内使用任何信号线或电源线。而且，如果使用双面的电路板，请避免在电路板上放置晶振的另一面布线。

图 2-4 给出了建议的电路板布局。可以使用包含完整振荡器引脚的单面布局来处理直插式封装器件。对于引脚排列紧密的器件，单面布局则可能无法始终完全包含所有的引脚和元件。适当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。在所有情况下，保护走线必须返回到地。

在规划应用的走线和 I/O 分配过程中，确保临近的端口引脚和离振荡器非常近的其他信号状态良好（即，没有很高的频率、陡升和陡降以及其他类似的噪声）。

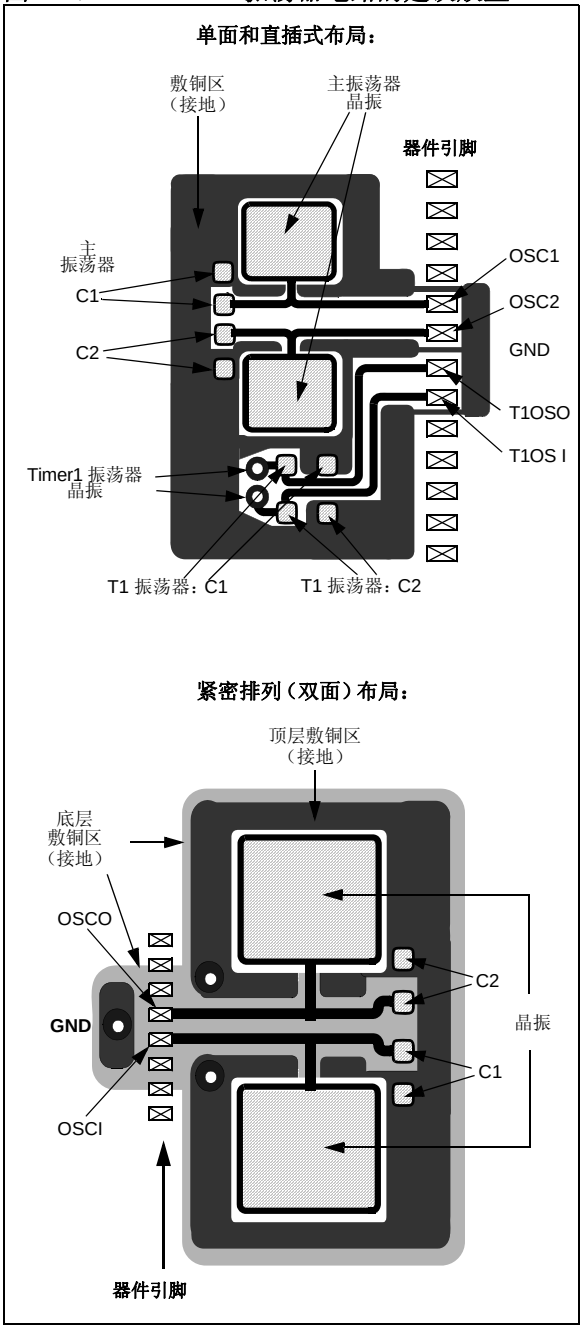
关于振荡器电路的其他信息和设计指导，请参考我公司网站（www.microchip.com）上提供的以下应用笔记：

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

2.7 未使用的 I/O

未使用的 I/O 引脚应配置为输出，并驱动为逻辑低电平状态。或者，使用一个 1 kΩ 至 10 kΩ 电阻将并未使用的引脚连接到 Vss，并将输出驱动为逻辑低电平。

图 2-5： 振荡器电路的建议放置



注：

© 2011 Microchip Technology Inc.

3.0 振荡器配置

3.1 振荡器类型

PIC18F87K22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， RA6 输出时钟信号 （频率为 FOSC/4）
- 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， RA6 输出时钟信号 （频率为 FOSC/4）
- INTIO2 内部振荡器， RA6 和 RA7 为 I/O 引脚
- INTIO1 内部振荡器， RA6 输出频率为 FOSC/4 的时钟信号， RA7 为 I/O 引脚

还可以对输入频率范围为 4 到 16 MHz 的任何时钟源运行 4 x PLL。

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

对于 EC 和 HS 模式， PLLEN （软件）或 PLLCFG （CONFIG）位可用于使能 PLL。

对于 INTIOx 模式 （HF-INTOSC）：

- 只有 PLLEN 可以使能 PLL （忽略 PLLCFG）。
- 如果将振荡器配置为内部振荡器 （FOSC<3:0> = `100x`），则只有当 HF-INTOSC 频率为 8 或 16 MHz 时才可以使能 PLL。

RA6 和 RA7 引脚不用于振荡器功能，也就是 CLKOUT 功能时，将用作通用 I/O。

要在使用 EC/HS/XT/LP/RC 作为主振荡器时优化功耗，可以配置频率输入范围以优化功耗：

- 低功耗配置——外部频率低于 160 kHz
- 中等功耗配置——外部频率介于 160 kHz 到 16 MHz 之间
- 高功耗配置——外部频率高于 16 MHz

用户通过编程 FOSC<3:0> 配置位（CONFIG1H<3:0>）可选择所有这些模式。此外， PIC18F87K22 系列器件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 中显示了 PIC18F87K22 系列器件的时钟源。

对于 HS 和 EC 模式，还有其他功耗工作模式，具体取决于工作频率。

HS1 是中等功耗模式，频率范围为 4 MHz 到 16 MHz。HS2 是高功耗模式，其中振荡器频率范围为 16 MHz 到 25 MHz。通过正确设置 CONFIG1H<3:0> 进入 HS1 和 HS2。（详细信息，请参见第 406 页上的寄存器 28-2。）

有以下三种 EC 工作模式：

- EC1——低功耗，频率范围小于等于 160 kHz
- EC2——中等功耗，频率范围为 160 kHz 到 16 MHz
- EC3——高功耗，频率范围为 16 MHz 到 64 MHz

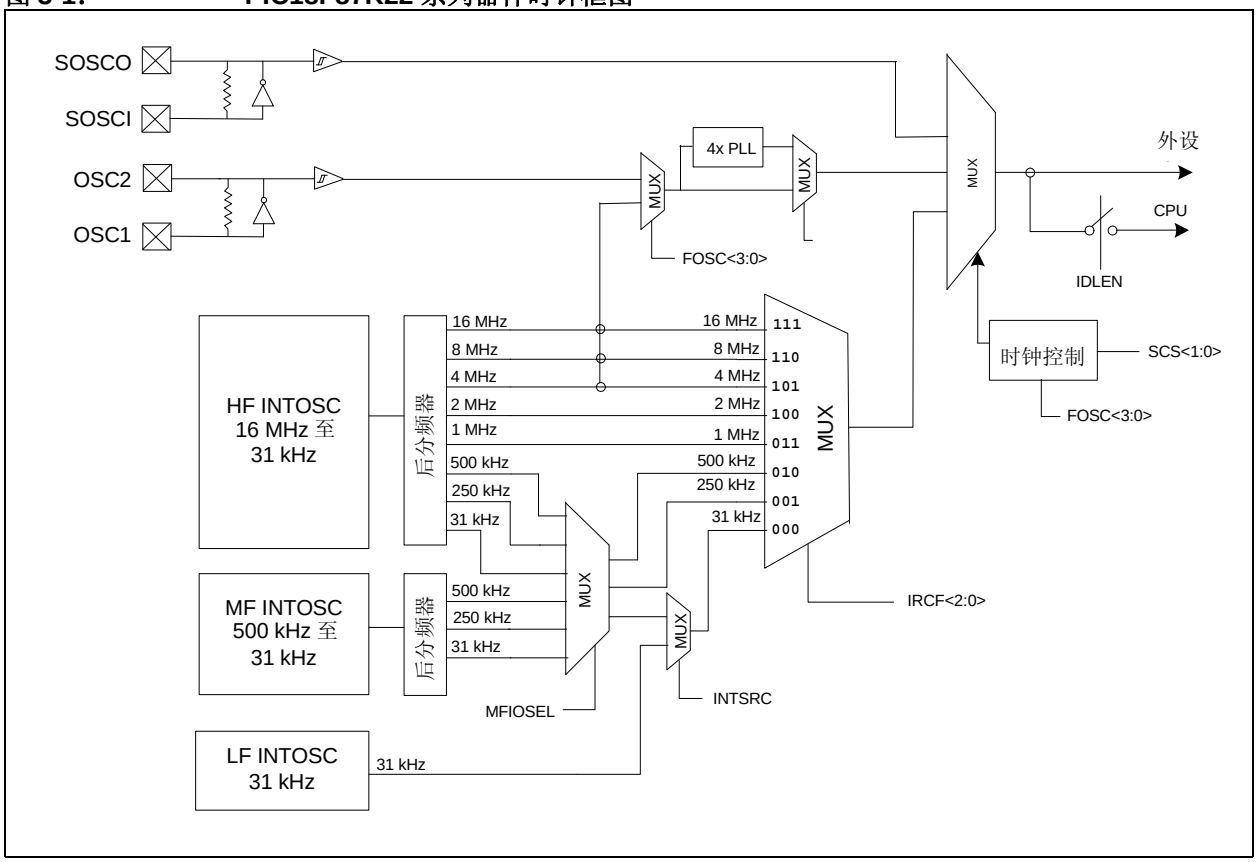
通过正确设置CONFIG1H<3:0>进入EC1、EC2和EC3。（详细信息，请参见第 406 页上的寄存器 28-2。）

表3-1给出了HS和EC模式的频率范围以及FOSC<3:0>设置。

表 3-1： HS、 EC、 XT、 LP 和 RC 模式：频率范围和设置

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： PIC18F87K22 系列器件时钟框图



© 2011 Microchip Technology Inc.

3.2 控制寄存器

OSCCON 寄存器（寄存器 3-1）控制器件时钟运行的主要方面。它选择要使用的振荡器类型、要使用的功耗管理模式以及 INTOSC 源的输出频率。它也提供振荡器的状态。

OSCTUNE 寄存器（寄存器 3-3）控制内部振荡器模块的频率调节和工作。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 (1)

R/W-0	R/W-1	R/W-1	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **IDLEN：**空闲使能位
1 = 在执行 SLEEP 指令后器件进入空闲模式
0 = 在执行 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，默认值）
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）
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；主振荡器正在运行，如 FOSC<3:0> 所定义
0 = 振荡器起振定时器（OST）延时正在运行；主振荡器尚未就绪，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： 如果正由内部振荡器提供器件时钟，更改这些位将导致时钟频率立刻切换。
3： 时钟源由 INTSRC 位（OSCTUNE<7>）选择。
4： 更改这些位将导致时钟源立刻切换。
5： INTSRC = OSCTUNE<7> 且 MFIOSEL = OSCCON2<0>。
6： 内部源的最低功耗选项。

寄存器 3-1： OSCCON：振荡器控制寄存器 (1) （续）

bit 2 **HFIOFS：**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（带/不带PLL的OSC1/OSC2或HF-INTOSC。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： 如果正由内部振荡器提供器件时钟，更改这些位将导致时钟频率立刻切换。
3： 时钟源由 INTSRC 位 （OSCTUNE<7>）选择。
4： 更改这些位将导致时钟源立刻切换。
5： INTSRC = OSCTUNE<7> 且 MFIOSEL = OSCCON2<0>。
6： 内部源的最低功耗选项。

寄存器 3-2： OSCCON2：振荡器控制寄存器 2

U-0	R-0	U-0	U-0	R/W-0	U-0	R-x	R/W-0
—	SOSCRUN	—	—	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 = 上电复位时的值	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 SOSC 以外的振荡器

bit 5-4 **未实现：**读为 0

bit 3 **SOSCGO：**振荡器起振控制位
1 = 振荡器正在运行，即使没有其他源请求也是如此
0 = 如果没有其他源请求，振荡器将关闭（如果 SOSC 选择为使用数字时钟输入而不是外部晶振运行，则此位无效。）

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，而不是频率为 500 kHz、 250 kHz 和 31.25 kHz 的 HF-INTOSC
0 = 不使用 MF-INTOSC

© 2011 Microchip Technology Inc.

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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **INTSRC：**内部振荡器低频源选择位

1 = 来自 16 MHz INTOSC 源的 31.25 kHz 器件时钟 （使能 HF-INTOSC 的 512 分频）
0 = 来自 31 kHz INTRC 振荡器的 31 kHz 器件时钟 （LF-INTOSC）

bit 6 **PLLEN：**PLL 倍频器使能位

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 振荡器在校准后的频率下运行
111111
•
•
100000 = 最低频率

3.3 时钟源和振荡器切换

实质上， PIC18F87K22 系列器件具有以下独立的时钟源：

- 主振荡器
- 辅助振荡器
- 内部振荡器

主振荡器可以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。如果通过 FOSC<3:0> 配置位 （CONFIG1H<3:0>）选择了内部振荡器模块，那么可以认为该模块是主振荡器。内部振荡器模块可以为以下其中之一：

- 31 kHz LF-INTRC 源
- 31 kHz 到 500 kHz MF-INTOSC 源
- 31 kHz 到 16 MHz HF-INTOSC 源

特定的模式由 FOSC 配置位定义。第 3.5 节 “外部振荡器模式” 中包含了这些模式的详情。

辅助振荡器是未连接到 OSC1 或 OSC2 引脚的外部时钟源。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。 PIC18F87K22 系列器件将 SOSC （Timer1/3/5/7）振荡器作为辅助振荡器源。该振荡器在所有功耗管理模式下通常被用作实时时钟 （Real-Time Clock， RTC）等功能的时基。

通过任何请求 SOSC 振荡器的外设可使能该振荡器。可使用以下八种方式使能：如果通过各自的 SOSCEN 位（TxCON<3>）将 SOSC 选择为任何奇数编号的定时器的时钟源；如果通过 RTCOSC 配置位（CONFIG3L<1>）将 SOSC 选择为 RTCC 的时钟源；如果通过设置 SCS 位（OSCCON<1:0>），将 SOSC 选择为 CPU 的时钟源，或将 SOSCGO 位（OSCCON2<3>）置1。SSOSCGO位用于预热SOSC，使之在有任何外设请求前做好准备。

辅助振荡器具有 3 个运行模式。 SOSCEL<1:0> 位（CONFIG1L<4:3>）决定 SOSC 工作模式：

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

如果不需要辅助振荡器，而需要端口引脚 RC0 和 RC1 作为数字 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下的时钟源。 LF-INTOSC 源也可以作为几种特殊功能部件 （例如 WDT 和故障保护时钟监视器）的时钟源。第 3.6 节 “内部振荡器模块” 中对内部振荡器模块进行了更为详细的讨论。

PIC18F87K22 系列器件包含允许器件时钟源从主振荡器（由器件配置选择）切换到其中一个备用时钟源的功能。当使能其他时钟源时，可以使用多种功耗管理工作模式。

3.3.1 OSC1/OSC2 振荡器

OSC1/OSC2 振荡器模块用来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 到 4 MHz
HS	4 MHz 到 25 MHz
EC	0 到 64 MHz （外部时钟）
EXTRC	0 到 4 MHz （外部 RC）

基于晶振的振荡器 （XT、 HS 和 LP）具有内置起振时间。之后将立即执行 EC 和 EXTRC 时钟操作。

3.3.2 时钟源选择

系统时钟选择位 SCS<1:0> （OSCCON2<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，接下来是一段很短的时钟转换延时，然后，时钟源改变。

OSTS （OSCCON<3>）和 SOSCRUN （OSCCON2<6>）位指明当前提供器件时钟的时钟源。OSTS 位置 1 表明振荡器起振定时器 （OST）已超时，且主时钟在主时钟模式下提供器件时钟。SOSCRUN 位置 1 表明 SOSC 振荡器（来自 Timer1/3/5/7）在辅助时钟模式下提供器件时钟。在功耗管理模式下，任何时候这 2 个位只有一个会置 1。如果这 2 位都没有置 1，则表示当前时钟源是 INTRC，或内部振荡器电路刚刚起振且尚未稳定。

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

© 2011 Microchip Technology Inc.

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

> 注 1：必须使能 Timer1/3/5/7 振荡器才能选择辅助时钟源。通过将 Timerx 控制寄存器中的 SOSCEN 位（TxCON<3>）置 1，可以使能 Timerx 振荡器。如果未使能 Timerx 振荡器，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，否则当Timerx振荡器起振时会有很长的延时。

3.3.2.1 系统时钟选择和器件复位

在所有形式的复位中 SCS 位都被清零，这表示 FOSC<3:0> 配置位定义的主振荡器将作为器件复位后的主时钟源。该振荡器可能是内部振荡器模块本身，也可能是其他某个主时钟源（HS、EC、XT、LP、外部 RC 以及使能了 PLL 的模式）。

在不带 PLL 的内部振荡器模块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。它将从 8 MHz 开始起振；这是对应于 IRCF<2:0> 位复位值（110）的后分频比选项。

无论所选的主振荡器为何，器件上电后将总是使能 INTRC。它将作为时钟源直到器件从存储器中装入了它的配置值。此时 FOSC 配置位被读取并选择振荡器的工作模式。

注意，主时钟源或内部振荡器各自均对应有两种可能的 SCS<1:0> 位设置值。

3.3.3 振荡器转换

PIC18F87K22 系列器件包含在时钟源切换时防止时钟产生“毛刺”的电路。在切换时钟时，器件时钟会有短暂停顿。该停顿的长度是旧时钟源的两个周期加上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。

在第 4.1.2 节 “进入功耗管理模式”中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：

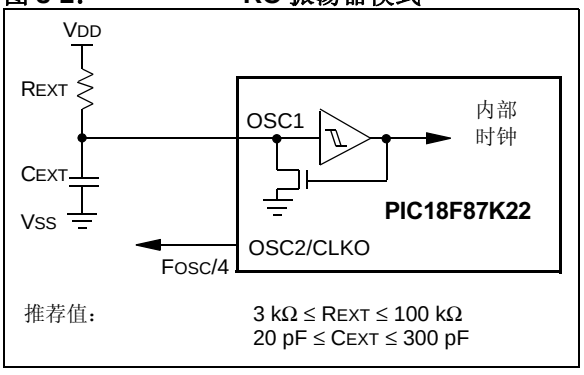
- 电源电压
- 外部电阻（REXT）和电容（CEXT）的值
- 工作温度

在器件型号、工作电压和温度以及元件值都相同的情况下，振荡频率仍然会各不相同。这些频率上的差异是由以下因素引起的：

- 正常制造工艺的差异
- 不同封装类型的引脚间电容的差异（尤其当 CEXT 值较小时）
- REXT 和 CEXT 标准容差的不同

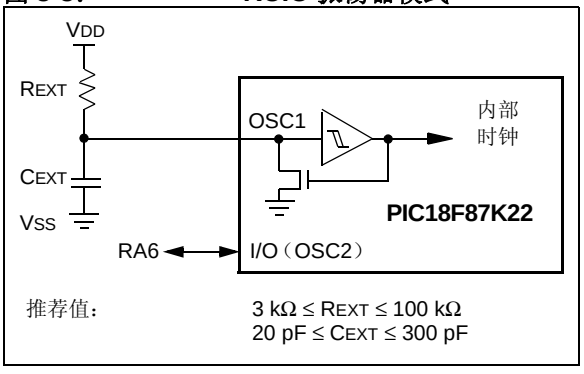
在 RC 振荡器模式下，振荡器频率的 4 分频信号可由 OSC2 引脚输出。此信号可用于测试或同步其他逻辑。图 3-2 显示了外接 R/C 组合电路的连接方式。

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 振荡模式下，晶振或陶瓷谐振器连接到 OSC1 和 OSC2 引脚来产生振荡信号。图 3-4 中显示了引脚连接。

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参数规范。

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。

要得到理想的振荡器工作状况，可能需要不同的电容值。用户应当在期望的 VDD 和温度条件下测试振荡器的性能。请参见以下应用笔记以获取振荡器的具体信息：

- 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”*

欲知更多信息，请参见表 3-3 后面的 “注”。

表 3-3： 晶振的电容选择

振荡类型	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。

要得到理想的振荡器工作状况，可能需要不同的电容值。用户应当在期望的 VDD 和温度条件下测试振荡器的性能。

振荡器的具体信息，请参见表 3-2 列出的 Microchip 应用笔记。欲知更多信息，请参见本表后面的 “注”。

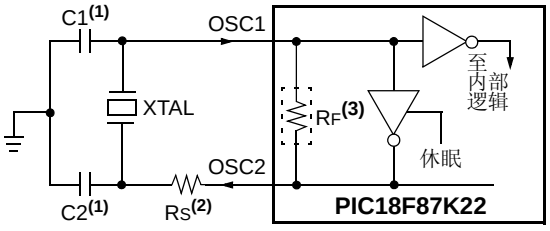
注 1：电容值越大，振荡器的稳定性越高，但同时起振时间也越长。

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 / 晶振制造厂商询问外部元件的适当值。

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：请始终在应用期望的 VDD 和温度条件下验证振荡器性能。

图 3-4： 晶振 / 陶瓷谐振器工作原理 （HS 或 HSPLL 配置）



注 1：请参见表3-2和表3-3以了解C1和C2的初始值。

2：对于AT条形切割的晶体可能会需要一个串联电阻 （RS）。

3：RF 的值取决于选定的振荡器模式。

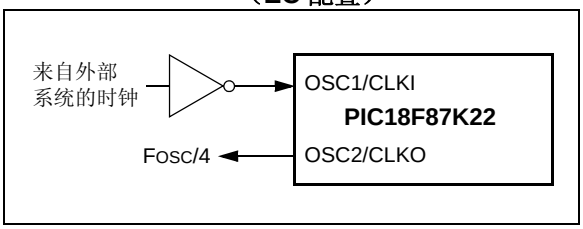
© 2011 Microchip Technology Inc.

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。

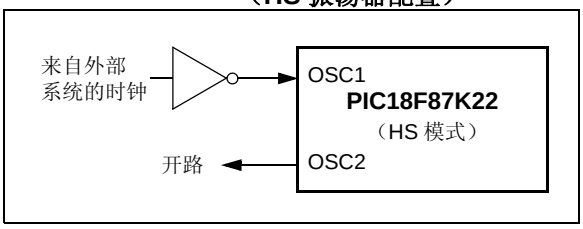
在 EC 振荡器模式下，振荡器频率的 4 分频信号可由 OSC2 引脚输出。此信号可用于测试或同步其他逻辑。图 3-5 显示了 EC 振荡器模式的引脚连接方式。

图 3-5：外部时钟输入工作原理（EC 配置）



在 HS 模式下，OSC1 引脚可能也需要连接一个外部时钟源，如图 3-6 所示。在此配置下，OSC2 的 4 分频输出不可用。此配置下的电流消耗将比 EC 模式稍高，因为将使能内部振荡器的反馈电路（在 EC 模式下，反馈电路是禁止的）。

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

如果用户希望使用低频振荡器电路或通过晶振将器件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，这样做可能会有用。

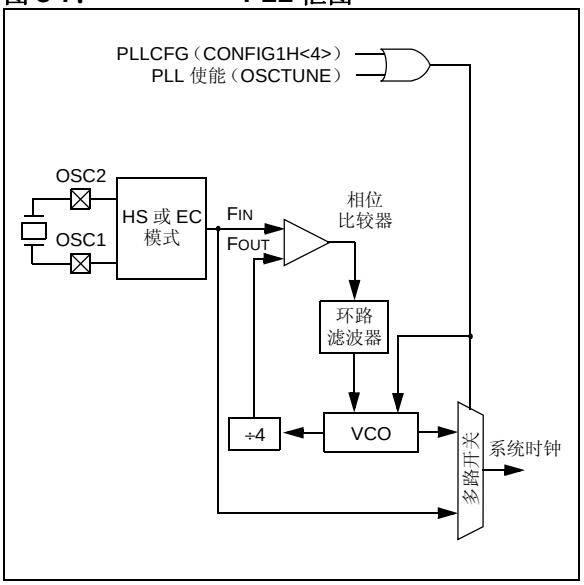
3.5.3.1 HSPLL 和 ECPLL 模式

HSPLL 和 ECPLL 模式可以使器件以外部振荡源频率的 4 倍运行，从而使最高频率可达 64 MHz。

通过将 PLLEN 位（OSCTUNE<6>）或 PLLCFG 位（CONFIG1H<4>）置 1 来使能 PLL。即使 PLLCFG 设置为 0，PLLEN 位也会为 PLL 提供软件控制。仅当 HS 或 EC 振荡器频率处于 4 MHz 到 16 MHz 的输入范围内，才使能 PLL。

这还允许使用软件更灵活地控制应用的时钟速度。只有在输入频率在 4 MHz-16 MHz 范围内时，PLLEN 才应在 HS 或 EC 振荡器模式下使能。

图 3-7：PLL 框图



3.5.3.2 PLL 和 HF-INTOSC

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

第 3.6.2 节 “INTPLL 模式”描述了带 PLL 的 INTOSC 的工作原理。请注意，只有在将 HF-INTOSC 后分频器配置为 8 MHz 或 16 MHz 时才使能 PLL。

3.6 内部振荡器模块

PIC18F87K22 系列器件包含一个内部振荡器模块，该模块可产生两种不同的时钟信号。这两个时钟均可充当单片机的时钟源，从而避免在 OSC1 和 / 或 OSC2 引脚上使用外部振荡器电路。

内部振荡器包括三个模块，具体取决于工作频率。分别是 HF-INTOSC、 MF-INTOSC 和 LF-INTRC。

在 HF-INTOSC 模式下，内部振荡器可以提供从 31 KHz 到 16 MHz 的频率范围，而后分频器决定所选频率（IRCF<2:0>）。

INTSRC 位（OSCTUNE<7>）和 MFIOSEL 位（OSCCON2<0>）还可以决定哪个 INTOSC 提供低频率（500 kHz 到 31 KHz）。要使 HF-INTOSC 提供这些频率，需 INTSRC = 1 且 MFI0SEL = 0。

在 HF-INTOSC 模式下，后分频器（IRCF<2:0>）提供 31 kHz 到 16 MHz 的频率范围。如果 HF-INTOSC 与 PLL 结合使用，那么 PLL 的输入频率应该为 8 MHz 或 16 MHz（IRCF<2:0> = 111 或 110）。

要使 MF-INTOSC 模式提供 31 kHz 到 500 kHz 的频率范围，需 INTSRC = 1 且 MFIOSEL = 1。在此模式下，后分频器（IRCF<2:0>）提供 31 kHz 到 500 kHz 的频率范围。

INTSRC = 0 时， LF-INTRC 只能提供 31 kHz 的频率。

LF-INTRC提供31 kHz的频率，如果选择LF-INTRC作为器件的时钟源，它将被使能。在使能以下功能时将自动使能此模式：

- 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”中详细讨论了这些功能。

通过配置 OSCCON 寄存器的 IRCF 位，以及 INTSRC 和 MFIOSEL 位，可以选择时钟源频率（HF-INTOSC、MF-INTOSC 或 LF-INTRC 直接频率）。器件复位后的默认频率为 8 MHz。

3.6.1 INTIO 模式

使用内部振荡器作为时钟源可以避免使用最多两个外部振荡器引脚，而将它们用作数字 I/O。目前有两种不同的振荡器配置（由 FOSC 配置位决定）：

- 在 INTIO1 模式下， OSC2 引脚（RA6）输出 Fosc/4，而 OSC1 引脚则充当 RA7（见图 3-8），用于数字输入和输出。
- 在 INTIO2 模式下， OSC1 将充当 RA7，而 OSC2 充当 RA6（见图 3-9）。二者都可用作数字输入和输出端口。

图 3-8： INTIO1 振荡器模式

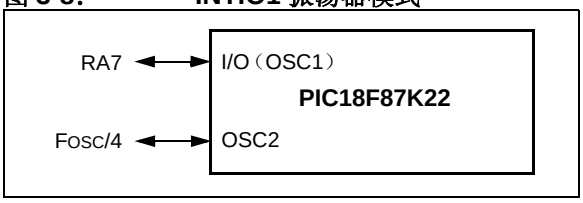
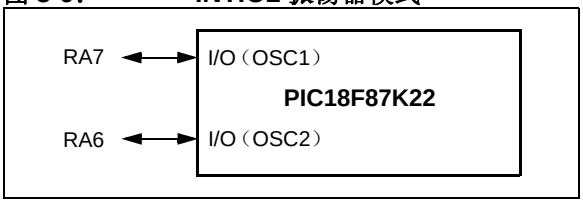


图 3-9： INTIO2 振荡器模式



3.6.2 INTPLL 模式

4 倍锁相环（PLL）可以与 HF-INTOSC 结合使用，以产生比通常与内部振荡器源结合使用时更快的器件时钟速度。使能时， PLL 可产生 32 MHz 或 64 MHz 的时钟速度。

PLL 操作通过软件控制。控制位 PLLEN（OSCTUNE<6>）用来使能或禁止其操作。此外，仅当选定的 HF-INTOSC 频率是 8 MHz 或 16 MHz（OSCCON<6:4> = 111 或 110）时， PLL 才会工作。

与 INTIO 模式类似，也提供两种不同的 INTPLL 模式：

- 在 INTPLL1 模式下， OSC2 引脚输出 Fosc/4，而 OSC1 则充当 RA7，用于数字输入和输出。从外部看来，此模式与 INTIO1（图 3-8）相同。
- 在 INTPLL2 模式下， OSC1 充当 RA7， OSC2 充当 RA6，两者都用于数字输入和输出。从外部看来，此模式与 INTIO2（图 3-9）相同。

© 2011 Microchip Technology Inc.

3.6.3 内部振荡器输出频率与调节

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>）在用户应用程序中对其进行调整。

当修改了 OSCTUNE 寄存器后，INTOSC（HF-INTOSC 和 MF-INTOSC）频率将逐渐变为新的频率。振荡器需要一些时间才能稳定下来。在此变化期间，代码会继续执行，且不会有任何迹象表明发生了变化。

LF-INTOSC 振荡器的工作独立于 HF-INTOSC 或 MF-INTOSC 源。电压和温度导致的 HF-INTOSC 或 MF-INTOSC 源的任何变化并不一定会导致 LF-INTOSC 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 电压或温度的改变而发生漂移，并可能以各种方式影响控制器运行。通过修改 OSCTUNE 寄存器的值可以调节 INTOSC 的频率。根据器件的不同，这可能不会对 LF-INTOSC 时钟源频率造成任何影响。

调节 INTOSC 时钟源需要了解何时调节、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出错时可能需要进行调节。帧错误表示器件时钟的频率太高。要对此进行调节，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；一个由外设时钟作为时钟源，而另一个由一个固定的参考源（如 SOSC 振荡器）作为时钟源。

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得多，则表示内部振荡器模块运行过快。要对此进行调节，需减小 OSCTUNE 寄存器中的值。

3.6.4.3 用捕捉模式下的 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源的独立运行的 Timer1 （或 Timer3）和周期已知的外部事件（即交流电源频率）。CCPRxH:CCPRxL 寄存器捕捉并记录第一个事件的时间，以备后用。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。由于外部事件周期是已知的，因此可以计算事件之间的时间差。

如果测量所得时间远大于计算所得时间，说明内部振荡器模块运行过快。要进行补偿，应减小 OSCTUNE 寄存器的值。如果测量所得时间远小于计算所得时间，说明内部振荡器模块运行过慢。要进行补偿，应增大 OSCTUNE 寄存器的值。

3.7 参考时钟输出

除了特定振荡器模式下的 FOSC/4 时钟输出外，PIC18F87K22 系列器件时钟都可以配置为提供到端口引脚的参考时钟输出信号。此功能在所有振荡器配置下都可用，并允许用户选择更大范围的时钟分频比，以驱动应用中的外部器件。

此参考时钟输出由 REFOCON 寄存器（寄存器 3-4）控制。将 ROON 位 （REFOCON<7>）置 1 将在 REFO（RE3）引脚上提供时钟信号。 RODIV<3:0> 位允许选择 16 种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提供参考时钟输出。 ROSSLP 位决定器件处于休眠模式时 RE3 上是否提供参考源。

要在休眠模式下使用参考时钟输出，必须同时将 ROSSLP 和 ROSEL 位置 1。还必须将器件时钟配置为 EC 或 HS 模式。否则，器件进入休眠模式后，将关闭 OSC1 和 OSC2 上振荡器的电源。清零 ROSEL 位允许参考时钟输出频率在任何时钟切换期间随着系统时钟的变化而变化。

© 2011 Microchip Technology Inc.

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 = 上电复位时的值	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 引脚上提供参考振荡器输出
0 = 禁止参考振荡器输出

bit 6 **未实现**：读为 0

bit 5 **ROSSLP**：休眠模式下参考振荡器输出停止位
1 = 参考振荡器在休眠模式下继续运行
0 = 在休眠模式下禁止参考振荡器

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；基本时钟可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 （REVOCON<4>），主振荡器只有在通过设置 FOSC 位配置为默认振荡器时才可用，而与器件是否处于休眠模式无关。

© 2011 Microchip Technology Inc.

3.8 功耗管理模式对各种时钟源的影响

当选择了 PRI_IDLE 模式时，指定的主振荡器会继续运行而不中断。对于所有其他的功耗管理模式，使用 OSC1 引脚的振荡器会被禁止。OSC1 引脚（以及由振荡器使用的 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。可以直接使用 31 kHz LF-INTOSC 输出来提供时钟，并且可被使能来支持多种特殊功能，而与功耗管理模式无关（欲知更多有关 WDT、故障保护时钟监视器和双速启动的信息，请参见**第 28.2 节 “看门狗定时器（WDT）”**到**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。因为消除了所有晶体管的开关电流，休眠模式能实现最小的器件电流消耗（仅泄漏电流）。

使能任何能在休眠期间工作的片上功能都将会增加休眠时的电流消耗。需要使能 INTOSC 来支持 WDT 工作。

SOSC 振荡器可以用来为实时时钟（RTC）提供时钟源。不需要器件时钟源的其他功能部件（即 MSSP 从器件、INTx 引脚等）也可以工作。**第 31.2 节 “直流特性：掉电电流和供电电流 PIC18F87K22 系列（工业级 / 扩展级）”**列出了会明显增加电流消耗的外设。

3.9 上电延时

有两个定时器控制上电延时，这样大部分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复位状态。欲知有关上电延时的更多信息，请参见**第 5.6 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），在上电时它提供了一段约 64 ms 的固定延时（表 31-13 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），用于在晶振稳定前使芯片保持在复位状态（HS、XT 或 LP 模式）。OST 通过在计数 1024 个振荡周期后允许振荡器为器件提供时钟来实现此功能。

在上电复位之后，会有一段 TCSD（表 31-13 中的参数 38）间隔的延时，此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（时钟的 4 分频输出）
HS 和 HSPLL	反馈反相器被禁止，处于静态电平	反馈反相器被禁止，处于静态电平
INTOSC 和 INTPLL1/2	I/O 引脚 RA6，由 TRISA<6> 控制方向	I/O 引脚 RA6，由 TRISA<7> 控制方向

注： 有关由于休眠和 $\overline{\text{MCLR}}$ 复位引起的延时的信息，请参见**第 5.0 节 “复位”**。

注：

© 2011 Microchip Technology Inc.

4.0 功耗管理模式

为了更有效地管理功耗， PIC18F87K22 系列器件总共提供了七种工作模式。这些模式为在资源受限的应用（如电池供电的器件）中有选择地节省功耗提供了多种选项。

有三种功耗管理模式：

- 运行模式
- 空闲模式
- 休眠模式

还有一个将器件从休眠模式唤醒的超低功耗唤醒（Ultra Low-Power Wake-up， ULPWU）模式。

这些模式定义了器件的哪些部分由时钟源驱动，有时还将定义时钟速度。运行和空闲模式可以使用三种可用时钟源（主时钟源、辅助时钟源或内部振荡器电路）中的任意一种；而休眠模式则不使用时钟源。

ULPWU 模式通过 RA0 引脚上的缓慢下降的电压来产生唤醒信号，将器件从功耗管理模式 （甚至是休眠模式）唤醒，而不需要消耗过多的电流。（见**第 4.7 节 “ 超低功耗唤醒功能 ”**。）

功耗管理模式包括几种在以前的 PIC® 器件上提供的节约功耗的功能。其中之一是其他 PIC18 器件也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。节省功耗的功能还包括所有 PIC 器件都提供的休眠模式，在此模式下器件的所有时钟均停止工作。

4.1 选择功耗管理模式

选择功耗管理模式需要考虑两个因素：

- 是否用时钟源驱动 CPU
- 使用哪个时钟源

IDLEN 位 （OSCCON<7>）控制是否为 CPU 提供时钟，而 SCS<1:0> 位 （OSCCON<1:0>）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> 位可以为功耗管理模式选择三个时钟源中的一个。它们是：

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

4.1.2 进入功耗管理模式

通过装载 OSCCON 寄存器可从一种功耗管理模式切换到另一种功耗管理模式。 SCS<1:0> 位选择时钟源并决定使用哪一种运行模式或空闲模式。更改这些位将导致立即切换到新的时钟源 （假定新的时钟源正在运行）。切换时，可能也会有时钟转换延迟。**第 4.1.3 节 “ 时钟转换和状态指示位 ”** 及其后续章节将会讨论这些问题。

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需要设置上述所有位，这由当前的模式和将要切换到的模式决定。可以通过在发出 SLEEP 之前更改振荡器选择位或更改 IDLEN 位来进行多种模式转换。如果已经正确地配置了IDLEN 位，可能只需通过执行 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		模块时钟控制		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	内部振荡器电路 (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	内部振荡器电路 (2)

注 1： 执行 SLEEP 指令时， IDLEN 的值得到反映。
2： 包括 INTOSC （HF-INTOSC 和 MG-INTOSC）和 INTOSC 预分频器，以及 LF-INTOSC 时钟源。

4.1.3 时钟转换和状态指示位

时钟源转换的时间长度是旧时钟源的2个周期与新时钟源的 3 到 4 个周期的和。此公式假设新时钟源是稳定的。在本章节中，HF-INTOSC 和 MF-INTOSC 统称为 INTOSC。

下面 3 位用于指示当前的时钟源及其状态（如表 4-2 所示）：

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

表 4-2： 系统时钟指示位

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 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 16 MHz 时钟源给分频器，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 LF-INTOSC 时钟源正在为器件提供时钟信号或 INTOSC 时钟源尚未稳定。

如果通过 FOSC<3:0> 配置位 （CONFIG1H<3:0>）将内部振荡器时钟配置为主时钟源，从而在 PRI_RUN 或 PRI_IDLE 模式时，可将 OSTS 和 HFIOFS 或 MFIOFS 位置 1。这表示主时钟 （INTOSC 输出）正在产生稳定的 16 MHz 输出。进入工作频率相同的另一个 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，可能会选择高于 VDD 所能支持的时钟速率。违反 VDD/FOSC 规范会导致器件运行不正常。
>
> **2：** 执行SLEEP指令不一定会使器件进入休眠模式。该指令充当触发条件，根据 IDLEN 位的设置，使控制器进入休眠模式或某种空闲模式。

4.1.4 多条 SLEEP 指令

SLEEP 指令启动的功耗管理模式由这条指令执行时 IDLEN 位的设置决定。如果执行另一条 SLEEP 指令，器件将进入那时由 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。不同运行模式之间的差异在于时钟源的不同。

4.2.1 PRI_RUN 模式

PRI_RUN 模式是单片机正常工作的全功耗执行模式。如果未使能双速启动， PRI_RUN 模式也是器件复位后的默认模式（详情见**第 28.4 节 “双速启动”**）。在此模式下，OSTS 位置 1。如果内部振荡器电路为主时钟源，HFIOFS 或 MFIOFS 位可能被置 1（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。在此模式下， CPU 和外设由 SOSC 振荡器提供时钟。这让用户在仍使用高精度时钟源的情况下适当地降低功耗。

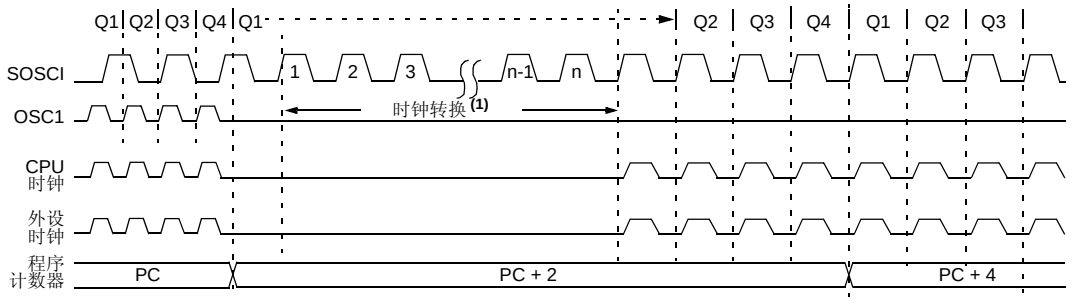
通过将 SCS<1:0> 位设置为 01 进入 SEC_RUN 模式。器件时钟源切换到 SOSC 振荡器 （见图 4-1），关闭主振荡器， SOSCRUN 位 （OSCCON2<6>）置 1 并且 OSTS 位清零。

> **注：** 将 SOSCGO 位（OSCCON2<3>）置 1 可使能 SOSC 振荡器。该位置 1 时，一旦 SCS<1:0> 位设置为 01，时钟将立即切换为 SEC_RUN 模式。

从 SEC_RUN 模式转换到 PRI_RUN 模式时，在主时钟启动期间，外设和 CPU 继续将 SOSC 振荡器作为时钟源。当主时钟准备就绪以后，时钟切换回主时钟（见图 4-2）。当时钟切换完成后， SOSCRUN 位被清零，OSTS 位被置 1，并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。

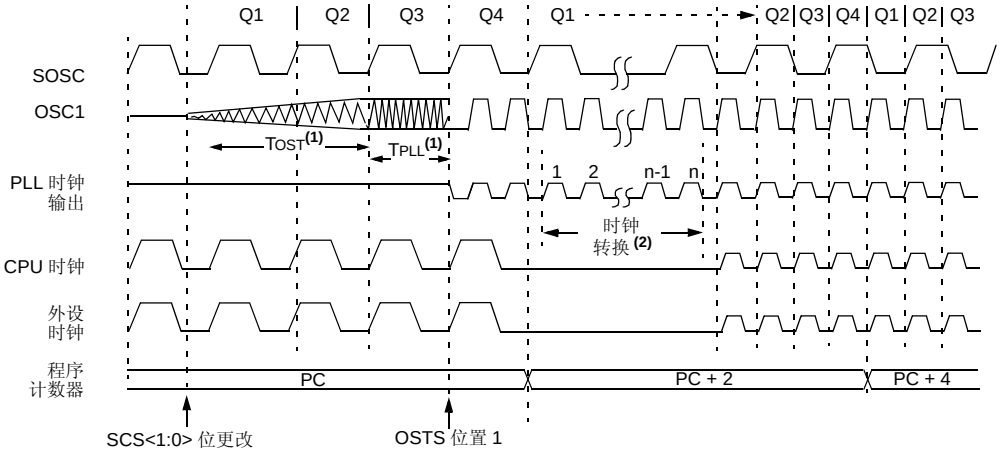
© 2011 Microchip Technology Inc.

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 模式到 PRI_RUN 模式 （HSPLL）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显示。

2：时钟转换通常发生在 2-4 个 TOSC 之内。

4.2.3 RC_RUN 模式

在 RC_RUN 模式下，由使用 INTOSC 复用器的内部振荡器电路作为 CPU 和外设的时钟源；主时钟关闭。在使用 LF-INTOSC 时钟源时，此模式是在代码执行期间所有运行模式中最节省功耗的运行模式。它非常适合于那些对时序要求不高或不是一直需要高速时钟的用户应用。

主时钟源为内部振荡器电路 LF-INTOSC 或 INTOSC（MF-INTOSC 或 HF-INTOSC）时，执行期间，PRI_RUN 和 RC_RUN 模式之间无明显差异。但是，进入或退出 RC_RUN 模式会导致时钟切换延迟。因此，如果主时钟源为内部振荡器电路，建议不要使用 RC_RUN 模式。

通过将 SCS1 位设置为 1 可以进入此模式。虽然 SCS0 的值常常被忽略，但仍建议将其清零，这将保证与未来器件的软件兼容性。当时钟源切换到 INTOSC 复用器（见图 4-3）时，主振荡器关闭，OSTS 位清零。任何时候更改 IRCF 位均可立即更改时钟速度。

> **注：** 在仅修改 IRCF 位时应该特别小心。在 VDD 较低时，可能会选择高于 VDD 所能支持的时钟速率。违反 VDD/FOSC 规范会导致器件运行不正常。

如果 IRCF 位和 INTSRC 位均被清零，不会使能 INTOSC 输出（HF-INTOSC/MF-INTOSC）且 HFIOFS 和 MFIOFS 位将保持清零。不会有当前时钟源的任何指示。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 INTSRC 或 MFIOSEL 被置 1， INTOSC 输出变稳定后 HFIOFS 或 MFIOFS 位将被置 1。详细信息，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 且时钟源是 LF-INTOSC
000	1	0	MFIOFS = 0， HFIOFS = 1 且时钟源是 HF-INTOSC
000	1	1	MFIOFS = 1， HFIOFS = 0 且时钟源是 MF-INTOSC
非零	x	0	MFIOFS = 0， HFIOFS = 1 且时钟源是 HF-INTOSC
非零	x	1	MFIOFS = 1， HFIOFS = 0 且时钟源是 MF-INTOSC

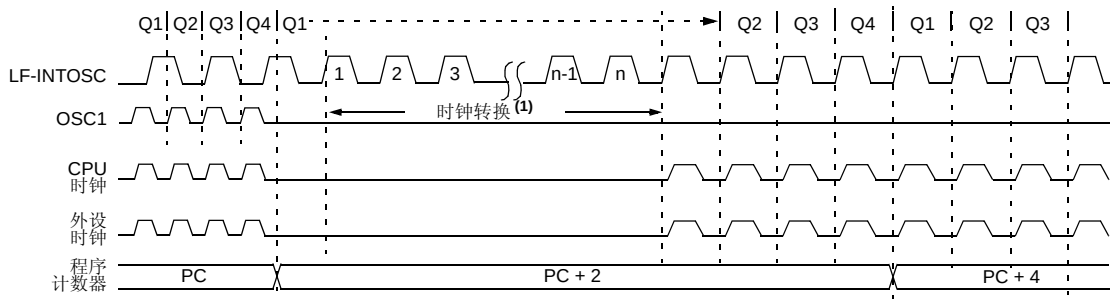
© 2011 Microchip Technology Inc.

INTOSC 时钟源稳定一个 TIOBST 间隔（表 31-13 中的参数 39）之后，将继续为器件提供时钟。

如果 IRCF 位在先前已被设置为一个非零值，或者在设置 SCS1 之前 INTSRC 已经置 1 并且 INTOSC 源已经稳定，那么 HFIOFS 或 MFIOFS 位将保持置 1 状态。

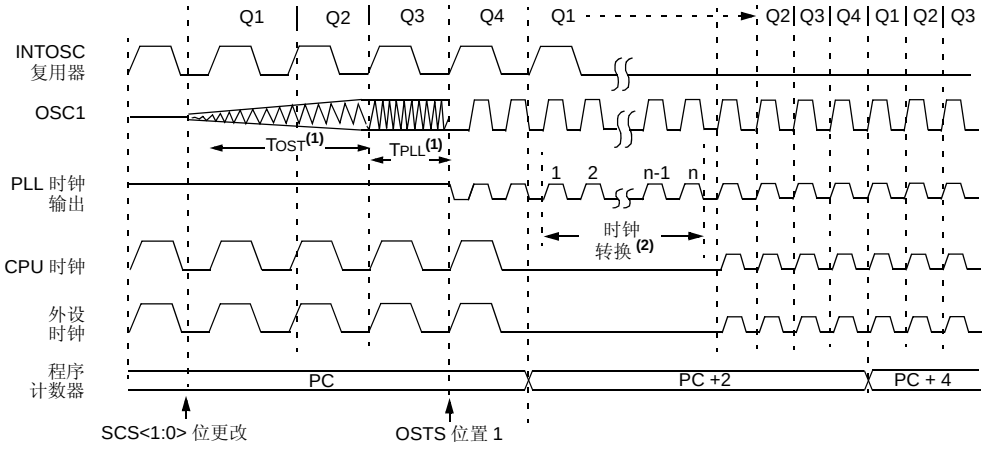
从 RC_RUN 模式转换到 PRI_RUN 模式时，在主时钟启动期间，器件继续使用 INTOSC 复用器作为时钟源。当主时钟准备就绪以后，时钟切换回主时钟（见图 4-4）。当时钟切换完成后， HFIOFS 或 MFIOFS 位被清零，OSTS 位被置 1，主时钟提供器件时钟。切换不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器， LF-INTOSC 源将继续运行。

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显示。
2： 时钟转换通常发生在 2-4 个 TOSC 之内。

4.3 休眠模式

PIC18F87K22 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 指令进入此模式。这将关闭选定的振荡器（图 4-5），并清零所有的时钟源状态位。

从任何其他模式进入休眠模式不需要时钟切换。这是因为一旦控制器进入休眠模式后就不需要时钟了。如果选择了 WDT， LF-INTOSC 源将继续运行。如果使能了 SOSC 振荡器， SOSC 振荡器也将继续运行。

在休眠模式下发生唤醒事件时（由于中断、复位或 WDT 超时），直到由 SCS<1:0> 位选定的时钟源准备就绪后，才为器件提供时钟（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，它会使用内部振荡器电路作为时钟源（见**第 28.0 节 “CPU 的特殊功能”**）。在这两种情况下，当主时钟提供器件时钟时， OSTS 位被置 1。唤醒事件不会影响 IDLEN 和 SCS 位。

4.4 空闲模式

空闲模式允许在外设继续工作时有选择地关闭单片机的 CPU。选择某种特定的空闲模式使用户能进一步管理功耗。

若在 SLEEP 指令执行时，IDLEN 位已置 1，外设将使用由 SCS<1:0> 位选择的时钟作为时钟源，但不会为 CPU 提供时钟。时钟源状态位不受影响。置 1 IDLEN 位并执行 SLEEP 指令将快速从给定运行模式切换到相应的空闲模式。

如果选择了 WDT，LF-INTOSC 源将继续运行。如果使能了 SOSC 振荡器， LF-INTOSC 源也将继续运行。

由于 CPU 没有执行指令，器件只能通过中断、WDT 超时或复位从任一空闲模式退出。当发生唤醒事件时，会有一个 TCSD 间隔（表 31-13 中的参数 38）的延时，之后 CPU 恢复代码执行。当 CPU 开始执行代码时，它将沿用当前空闲模式使用的时钟源。例如，当从 RC_IDLE 模式唤醒时，将使用内部振荡器电路作为 CPU 和外设的时钟源（即 RC_RUN 模式）。唤醒事件不会影响 IDLEN 和 SCS 位。

当处于任何空闲模式或休眠模式时， WDT 超时将导致 WDT 唤醒并进入由 SCS<1:0> 位当前指定的运行模式。

图 4-5： 进入休眠模式的转换时序

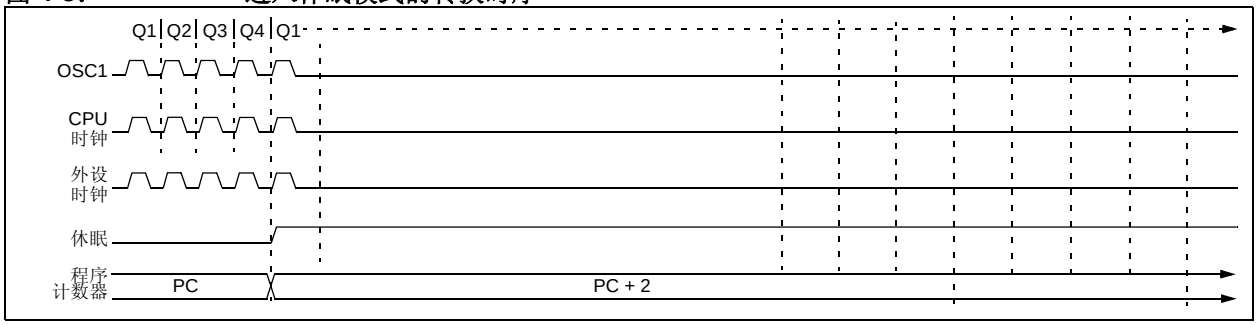
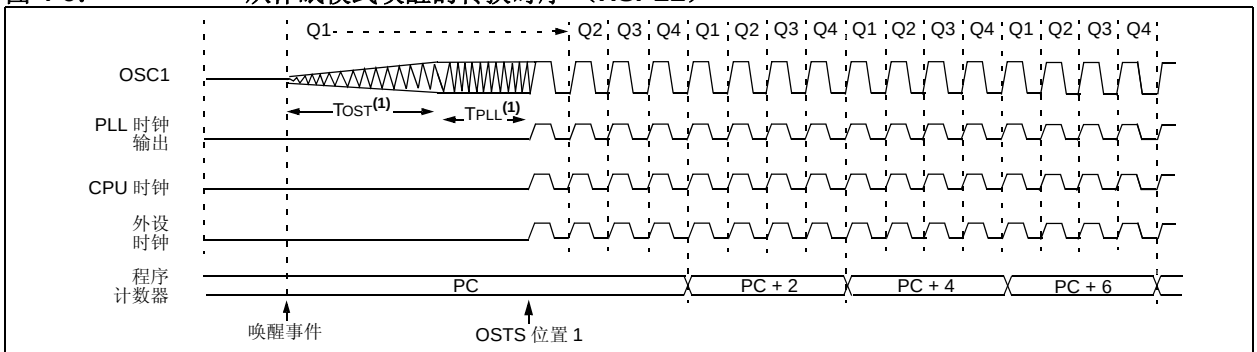


图 4-6： 从休眠模式唤醒的转换时序 （HSPLL）



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

© 2011 Microchip Technology Inc.

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”或从其他振荡器转换过来，选用此模式可以使对时序要求较高的应用以最快的速度恢复器件运行并使用较精确的主时钟源。

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。如果器件处于其他运行模式，请先将 IDLEN 位置 1，然后清零 SCS 位并执行 SLEEP。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。在唤醒事件和代码开始执行之间需要一段间隔为 TCSD（表 31-13 中的参数 39）的延时。这段时间可以使 CPU 做好执行指令的准备。在唤醒之后， OSTS 位保持置 1。唤醒事件不会影响 IDLEN 和 SCS 位 （见图 4-8）。

4.4.2 SEC_IDLE 模式

在 SEC_IDLE 模式中， CPU 被禁止，但外设继续将 SOSC 振荡器作为时钟源。通过将 IDLEN 位置 1 并执行 SLEEP 指令，可以从 SEC_RUN 进入此模式。如果器件处于其他运行模式，请先将 IDLEN 置 1，然后将 SCS<1:0> 位置为 01 并执行 SLEEP。当时钟源切换到 SOSC 振荡器时，主振荡器关闭， OSTS 位清零，SOSCRUN 位置 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 的时间间隔， CPU 开始使用 SOSC 振荡器作为时钟源执行代码。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

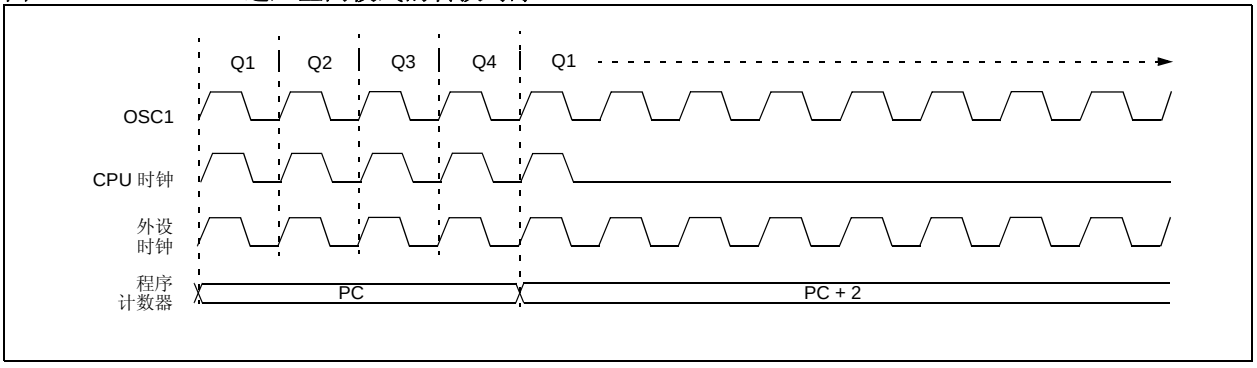
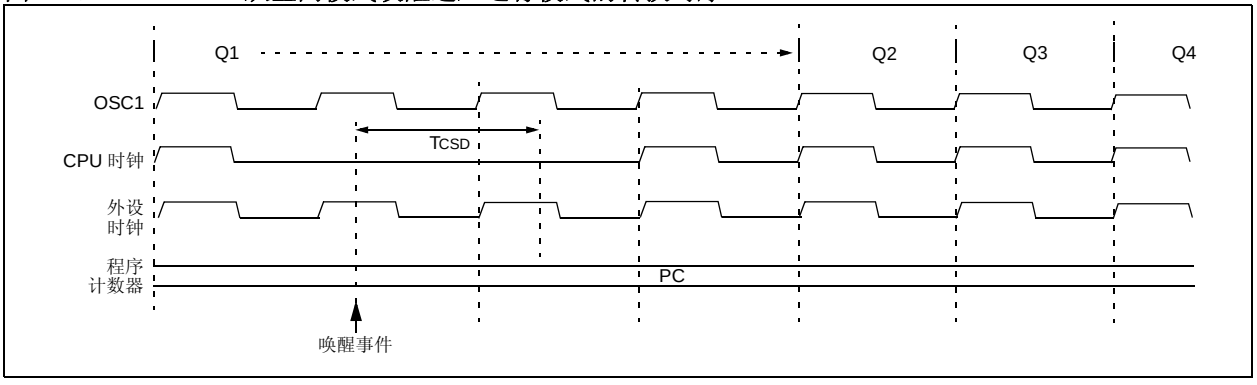


图 4-8： 从空闲模式唤醒进入运行模式的转换时序



4.4.3 RC_IDLE 模式

在 RC_IDLE 模式中，CPU 被禁止，但外设仍继续由使用 INTOSC 复用器的内部振荡器电路作为时钟源。该模式允许在空闲期间对功耗进行控制。

通过将 IDLEN 位置 1 并执行 SLEEP 指令，可以从 RC_RUN 模式进入此模式。如果器件处于其他运行模式，请先将 IDLEN 位置 1，然后将 SCS1 位置 1 并执行 SLEEP。虽然 SCS0 的值常常被忽略，但仍建议将其清零，这将保证与未来器件的软件兼容性。通过在执行 SLEEP指令之前修改IRCF位，可以使用INTOSC复用器来选择更高的时钟频率。当时钟源切换到 INTOSC 复用器时，主振荡器被关闭， OSTS 位被清零。

如果 IRCF 位被设置为任何非零值，或者 INTSRC/MFIOSE 位被置 1，就会使能 INTOSC 输出。在一个 TIOBST 间隔 （表 31-13 中的参数 38）之后， INTOSC 输出趋于稳定，然后 HFIOFS/MFIOFS 位置 1。（关于 HFIOFS/MFIOFS 位的更多信息，请参见表 4-3）。

INTOSC 时钟源稳定后，外设的时钟继续运行。如果 IRCF位先前就为非零值或在执行SLEEP指令和INTOSC 时钟源稳定前 INTSRC 位已置 1，HFIOFS/MFIOFS 将继续保持置 1。如果 IRCF 和 INTSRC 位全部清零，就不会使能 INTOSC 输出，HFIOFS/MFIOFS 位将保持清零状态，此时将不会有当前时钟源的任何指示。

当唤醒事件发生时，外设继续将 INTOSC 复用器作为时钟源。唤醒事件发生后经过一个 TCSD （表 31-13 中的参数 38）的时间间隔， CPU 开始使用 INTOSC 复用器作为其时钟源执行代码。唤醒事件不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器，INTRC 源将继续运行。

4.5 有选择的外设模块控制

空闲模式允许用户通过停止 CPU 时钟来大幅度地减少功耗。即便如此，外设模块时钟仍然存在。有时此模式无法满足应用的需要：比如将绝大部分能源分配给 CPU 处理工作，而只为外设提供最低的功耗。

PIC18F87K22 系列器件通过允许有选择地禁止外设模块以降低或消除功耗来满足这种需求。可通过设置以下两个控制位实现：

- 外设使能位，通常称为 XXXEN——位于各个模块的主控制寄存器中
- 外设模块禁止 （PMD）位，通常称为 XXXMD——位于某个 PMDx 控制寄存器 （PMD0、PMD1、 PMD2 或 PMD3）中

通过清零相应的 XXXEN 位来禁止模块从而禁止其功能，但是仍可对该模块的寄存器进行读写操作。这将降低功耗，但是不如第二种方法降低的多。

大部分外设模块都有一个使能位。

相比之下，模块的 PMD 位置 1 将禁止提供给该模块的所有时钟源，从而将功耗降至绝对最小值。在此状态下，与外设相关的控制寄存器和状态寄存器也被禁止，因此无法写入这些寄存器，且读到的值无效。许多外设模块都有相应的 PMD 位。

PIC18F87K22 系列器件有四个 PMD 寄存器：PMD0、PMD1、PMD2 和 PMD3。这些寄存器具有与每个模块相关的位，用来禁止和使能特定外设。

© 2011 Microchip Technology Inc.

寄存器 4-1： PMD3：外设模块禁止寄存器 3

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CCP10MD(1)	CCP9MD(1)	CCP8MD	CCP7MD	CCP6MD	CCP5MD	CCP4MD	TMR12MD(1)
bit 7							bit 0

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

bit 7 **CCP10MD：**PMD CCP10 使能 / 禁止位 (1)
1 = 使能 CCP10 的外设模块禁止 （PMD），禁止它的所有时钟源
0 = 禁止 CCP10 的 PMD

bit 6 **CCP9MD：**PMD CCP9 使能 / 禁止位 (1)
1 = 使能 CCP9 的外设模块禁止 （PMD），禁止它的所有时钟源
0 = 禁止 CCP9 的 PMD

bit 5 **CCP8MD：**PMD CCP8 使能 / 禁止位
1 = 使能 CCP8 的外设模块禁止 （PMD），禁止它的所有时钟源
0 = 禁止 CCP8 的 PMD

bit 4 **CCP7MD：**PMD CCP7 使能 / 禁止位
1 = 使能 CCP7 的外设模块禁止 （PMD），禁止它的所有时钟源
0 = 禁止 CCP7 的 PMD

bit 3 **CCP6MD：**PMD CCP6 使能 / 禁止位
1 = 使能 CCP6 的外设模块禁止 （PMD），禁止它的所有时钟源
0 = 禁止 CCP6 的 PMD

bit 2 **CCP5MD：**PMD CCP5 使能 / 禁止位
1 = 使能 CCP5 的外设模块禁止 （PMD），禁止它的所有时钟源
0 = 禁止 CCP5 的 PMD

bit 1 **CCP4MD：**PMD CCP4 使能 / 禁止位
1 = 使能 CCP4 的外设模块禁止 （PMD），禁止它的所有时钟源
0 = 禁止 CCP4 的 PMD

bit 0 **TMR12MD：**TMR12MD 禁止位 (1)
1 = 使能 PMD 并禁止所有 TMR12MD 时钟源
0 = 禁止 PMD 并使能 TMR12MD

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

寄存器 4-2： PMD2：外设模块禁止寄存器 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
TMR10MD[(1)]	TMR8MD	TMR7MD[(1)]	TMR6MD	TMR5MD	CMP3MD	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **TMR10MD：**TMR10MD 禁止位 [(1)]
1 = 使能外设模块禁止 （PMD）并禁止所有 TMR10MD 时钟源
0 = 禁止 PMD 并使能 TMR10MD

bit 6 **TMR8MD：**TMR8MD 禁止位
1 = 使能 PMD 并禁止所有 TMR8MD 时钟源
0 = 禁止 PMD 并使能 TMR8MD

bit 5 **TMR7MD：**TMR7MD 禁止位 [(1)]
1 = 使能 PMD 并禁止所有 TMR7MD 时钟源
0 = 禁止 PMD 并使能 TMR7MD

bit 4 **TMR6MD：**TMR6MD 禁止位
1 = 使能 PMD 并禁止所有 TMR6MD 时钟源
0 = 禁止 PMD 并使能 TMR6MD

bit 3 **TMR5MD：**TMR5MD 禁止位
1 = 使能 PMD 并禁止所有 TMR5MD 时钟源
0 = 禁止 PMD 并使能 TMR5MD

bit 2 **CMP3MD：**PMD 比较器 3 使能 / 禁止位
1 = 使能比较器 3 的 PMD，禁止它的所有时钟源
0 = 禁止比较器 3 的 PMD

bit 1 **CMP2MD：**PMD 比较器 2 使能 / 禁止位
1 = 使能比较器 2 的 PMD，禁止它的所有时钟源
0 = 禁止比较器 2 的 PMD

bit 0 **CMP1MD：**PMD 比较器 1 使能 / 禁止位
1 = 使能比较器 1 的 PMD，禁止它的所有时钟源
0 = 禁止比较器 1 的 PMD

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

寄存器 4-3： 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	CTMUMD	RTCCMD(1)	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBMD
bit 7							bit 0

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

bit 7 **PSPMD：**外设模块禁止 （PMD） PSP 使能 / 禁止位
1 = 使能 PSP 的 PMD，禁止它的所有时钟源
0 = 禁止 PSP 的 PMD

bit 6 **CTMUMD：**PMD CTMU 使能 / 禁止位
1 = 使能 CTMU 的 PMD，禁止它的所有时钟源
0 = 禁止 CMTU 的 PMD

bit 5 **RTCCMD：**PMD RTCC 使能 / 禁止位 (1)
1 = 使能 RTCC 的 PMD，禁止它的所有时钟源
0 = 禁止 RTCC 的 PMD

bit 4 **TMR4MD：**TMR4MD 禁止位
1 = 使能 PMD 并禁止所有 TMR4MD 时钟源
0 = 禁止 PMD 并使能 TMR4MD

bit 3 **TMR3MD：**TMR3MD 禁止位
1 = 使能 PMD 并禁止所有 TMR3MD 时钟源
0 = 禁止 PMD 并使能 TMR3MD

bit 2 **TMR2MD：**TMR2MD 禁止位
1 = 使能 PMD 并禁止所有 TMR2MD 时钟源
0 = 禁止 PMD 并使能 TMR2MD

bit 1 **TMR1MD：**TMR1MD 禁止位
1 = 使能 PMD 并禁止所有 TMR1MD 时钟源
0 = 禁止 PMD 并使能 TMR1MD

bit 0 **EMBMD：**PMD EMB 使能 / 禁止位
1 = 使能 EMB 的 PMD，禁止它的所有时钟源
0 = 禁止 EMB 的 PMD

注 1： 只能在 EECON2 解锁序列之后将 RTCCMD 设置为 1。有关解锁序列，请参见**第 18.0 节 “实时时钟和日历 （RTCC）”**（例 18-1）。

寄存器 4-4： 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
CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD
bit 7							bit 0

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

bit 7 **CCP3MD**：PMD ECCP3 使能 / 禁止位
1 = 使能 ECCP3 的外设模块禁止 （PMD），禁止它的所有时钟源
0 = 禁止 ECCP3 的 PMD

bit 6 **CCP2MD**：PMD ECCP2 使能 / 禁止位
1 = 使能 ECCP2 的 PMD，禁止它的所有时钟源
0 = 禁止 ECCP2 的 PMD

bit 5 **CCP1MD**：PMD ECCP1 使能 / 禁止位
1 = 使能 ECCP1 的 PMD，禁止它的所有时钟源
0 = 禁止 ECCP1 的 PMD

bit 4 **UART2MD**：PMD UART2 使能 / 禁止位
1 = 使能 UART2 的 PMD，禁止它的所有时钟源
0 = 禁止 UART2 的 PMD

bit 3 **UART1MD**：PMD UART1 使能 / 禁止位
1 = 使能 UART1 的 PMD，禁止它的所有时钟源
0 = 禁止 UART1 的 PMD

bit 2 **SSP2MD**：PMD MSSP2 使能 / 禁止位
1 = 使能 MSSP2 的 PMD，禁止它的所有时钟源
0 = 禁止 MSSP2 的 PMD

bit 1 **SSP1MD**：PMD MSSP1 使能 / 禁止位
1 = 使能 MSSP1 的 PMD，禁止它的所有时钟源
0 = 禁止 MSSP1 的 PMD

bit 0 **ADCMD**：PMD 模数转换器使能 / 禁止位
1 = 使能模数转换器的 PMD，禁止它的所有时钟源
0 = 禁止模数转换器的 PMD

© 2011 Microchip Technology Inc.

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退出。本节将讨论从功耗管理模式退出的触发条件。在每种功耗管理模式章节中我们已经讨论了时钟源子系统的作用（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

任何可用的中断源可以使器件从空闲模式或休眠模式退出，并进入运行模式。要使能此功能，必须通过将某个 INTCONx 或 PIEx 寄存器中的相应中断允许位置 1 来允许中断源。当相应的中断标志位置1时，触发退出序列。

当通过中断从空闲或休眠模式退出时，如果 GIE/GIEH 位（INTCON<7>）置 1，代码就会跳转到中断向量处执行。否则，代码就会按顺序执行或从头开始执行，而不发生跳转（见**第 11.0 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` 指令将清零 WDT 定时器和后分频器，当前选择的时钟源失效（如果使能了故障保护时钟监视器）并修改 OSCCON 寄存器中的 IRCF 位（如果内部振荡器电路是器件时钟源）。

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。如果以内部振荡器电路作为新的时钟源，则 HFIOFS/MFIOFS 位将置 1。

从复位状态退出到开始执行代码的延迟时间由唤醒前后的时钟源以及振荡器的类型（如果新的时钟源为主时钟）决定。表 4-4 为退出延时汇总。

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马上开始执行代码。由内部振荡器电路驱动的 INTOSC 复用器作为代码执行的时钟源。执行代码时，由内部振荡器电路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

从功耗管理模式退出的某些情况根本不会启动 OST。有两种情形：

- 主时钟源一直工作的 PRI_IDLE 模式；
- 主时钟源不处于 LP、 XT、 HS 或 HSPLL 模式中的任何一个

在这两种情况下，主时钟源不需要振荡器起振延时，因为它已经在运行（PRI_IDLE 模式），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）。然而，当退出休眠和空闲模式时，在唤醒事件后仍需要一个固定的延时 T_{CSD}，以便让 CPU 为执行代码做好准备。在此延时后的第一个时钟周期恢复指令执行。

4.7 超低功耗唤醒功能

RA0 引脚上的超低功耗唤醒 （ULPWU）功能允许缓慢下降的电压来产生中断，而不会消耗很大的电流。

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 引脚上的电容充电。
2. 通过将 RA0 配置为输入停止对电容充电。
3. 将 WDTCON 寄存器中的 ULPEN 和 ULPSINK 位置 1 对电容进行放电。
4. 配置休眠模式。
5. 进入休眠模式。

当 RA0 引脚上的电压下降至低于 VIL 时，器件唤醒并执行下一条指令。

此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 引脚上的 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>）置 1。可由软件在唤醒时检查该位的状态以确定唤醒源。

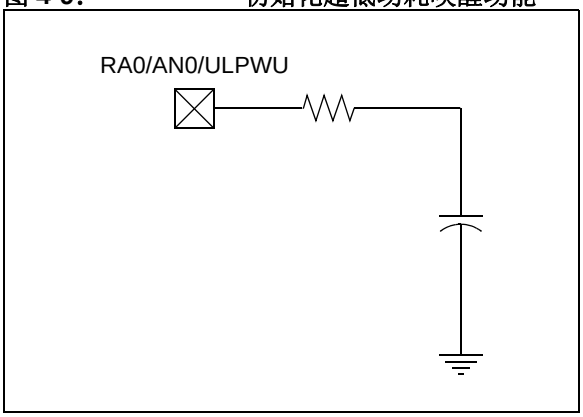
关于初始化 ULPWU 模块的信息，请参见例 4-1。

例 4-1： 初始化超低功耗唤醒功能

```
            //***************************
            //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/AN0/ULPWU 引脚提供了过流保护，同时允许使用软件校准超时时间 （见图 4-9）。

图 4-9： 初始化超低功耗唤醒功能



可使用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。该技术弥补了温度、电压和元件精度的影响。外设也可配置为简单的可编程低电压检测 （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）。

© 2011 Microchip Technology Inc.

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 和 HS	TCSD(1)	OSTS
	HSPLL		
	EC 和 RC		
	HF-INTOSC(2)		HFIOFS
	MF-INTOSC(2)		MFIOFS
	LF-INTOSC		无
SEC_IDLE 模式	SOSC	TCSD(1)	SOSCRUN
RC_IDLE 模式	HF-INTOSC(2)	TCSD(1)	HFIOFS
	MF-INTOSC(2)		MFIOFS
	LF-INTOSC		无
休眠模式	LP、 XT 和 HS	TOST(3)	OSTS
	HSPLL	TOST + t_{rc}(3)	
	EC 和 RC	TCSD(1)	
	HF-INTOSC(2)	TIOBST(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 TCSD （表 31-13 中的参数 38）延时，同时伴有所需的其他延时 （见**第 4.4 节 “空闲模式”**）。

2： 包含后分频器产生的频率。复位时， INTOSC 默认为 8 MHz 的 HF-INTOSC。

3： TOST 为振荡器起振定时器延时 （表 31-13 中的参数 32）。 TRC 为 PLL 锁定定时器延时 （表 31-7 中的参数 F12），也被称为 TPLL。

4： 在 INTOSC 稳定周期 TIOBST （表 31-13 中的参数 39）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。

注：

© 2011 Microchip Technology Inc.

5.0 复位

PIC18F87K22 系列器件有以下几种不同的复位方式：

a) 上电复位 （Power-on Reset， POR）
b) 正常工作状态下的 $\overline{\text{MCLR}}$ 复位
c) 功耗管理模式下的 $\overline{\text{MCLR}}$ 复位
d) 看门狗定时器 （WDT）复位 （代码执行期间）
e) 配置不匹配（Configuration Mismatch，CM）复位
f) 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

本节将讨论由 $\overline{\text{MCLR}}$、POR 和 BOR 产生的各种复位以及各种起振定时器的操作。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。

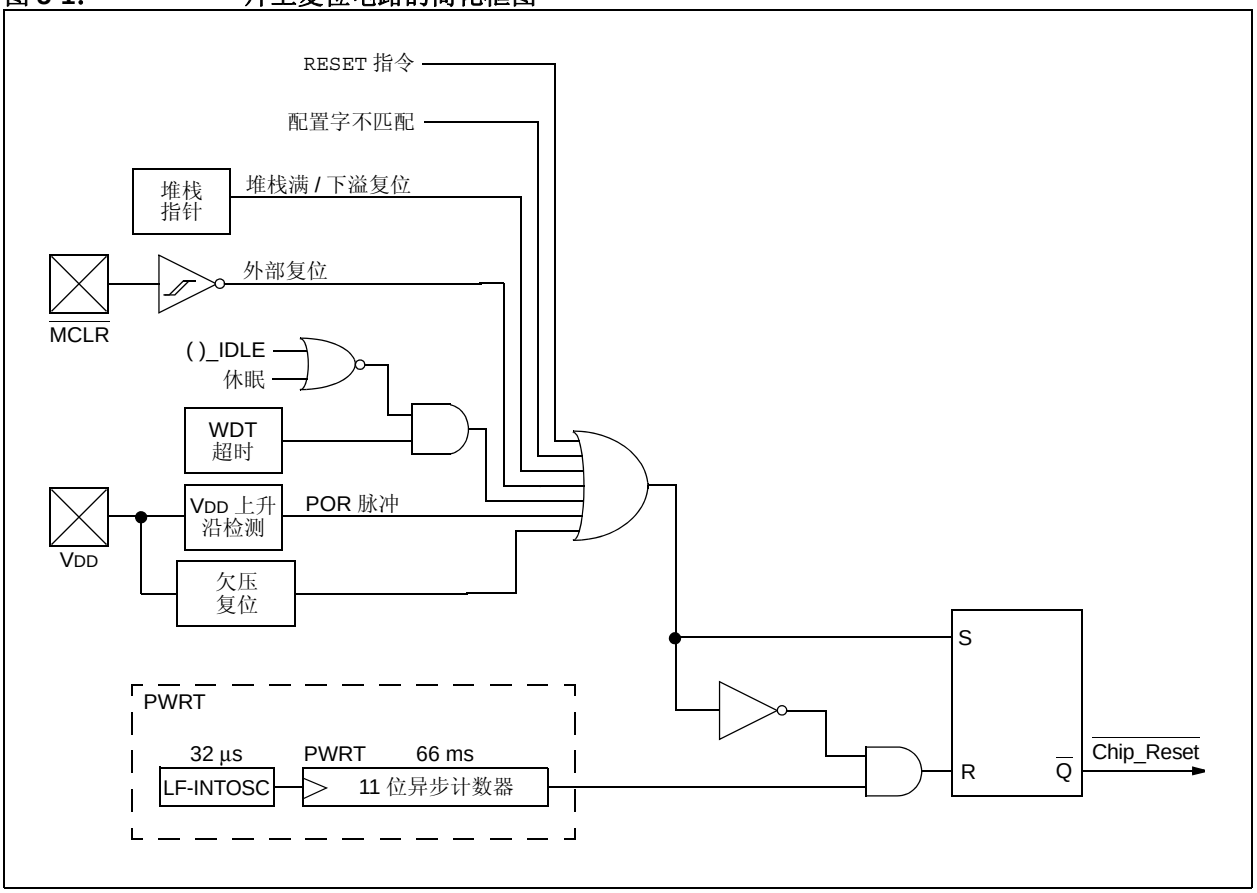
5.1 RCON 寄存器

可通过 RCON 寄存器 （寄存器 5-1）跟踪器件复位事件。该寄存器的低 5 位表示特定的复位事件是否已经发生。在大部分情况下，只能通过复位事件将这些位置 1，而且必须在事件发生后由应用程序将它们清零。

同时读取这 5 个标志位的状态可以获知刚刚发生的复位的类型。**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 寄存器还有一个用于设置中断优先级的控制位（IPEN）。**第 11.0 节 “中断”** 将讨论中断优先级。

图 5-1： 片上复位电路的简化框图



寄存器 5-1： 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 = 上电复位时的值	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
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 （必须在发生配置不匹配复位后由软件置 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 = 已执行 RESET 指令，导致器件复位 （必须在发生欠压复位后由软件置 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
1 = 未发生上电复位 （只能由固件置 1）
0 = 已发生上电复位 （必须在发生上电复位后由软件置 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 = 已发生欠压复位 （必须在发生欠压复位后由软件置 1）

注 1：建议在检测到上电复位后，将 $\overline{POR}$ 位置 1，以便检测后续的上电复位。
2：当 $\overline{BOR}$ 为 0 且 $\overline{POR}$ 为 1 时 （假定在上电复位之后立即由软件将 $\overline{POR}$ 置 1），表示已发生了欠压复位。

© 2011 Microchip Technology Inc.

5.2 主复位（$\overline{\text{MCLR}}$）

$\overline{\text{MCLR}}$ 引脚提供了用外部硬件触发器件复位的方法。将该引脚拉低可以产生复位信号。 PIC18 扩展单片机在 $\overline{\text{MCLR}}$ 复位路径上有一个噪声滤波器，该滤波器检测并滤除小的干扰脉冲。

任何内部复位（包括 WDT 复位）都不能将 $\overline{\text{MCLR}}$ 引脚驱动为低电平。

5.3 上电复位（POR）

只要当 VDD 超过设定的门限值后，就会在片上产生上电复位脉冲。这使得器件会在 VDD 达到满足器件正常工作的数值时以初始化状态启动。

要使用 POR 电路，需要将 $\overline{\text{MCLR}}$ 引脚通过一个电阻（阻值为 1 kΩ 到 10 kΩ）连接到 VDD。这样可以省去产生上电复位延时通常需要的外部 RC 元件。VDD 的最小上升速率在参数 D004 中指定。对于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，必须满足特定的工作参数要求（电压、频率和温度等），才能确保其正常工作。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。

上电复位事件由 $\overline{\text{POR}}$ 位（RCON<1>）捕捉。每当发生上电复位时，该位的状态就会被置为 0，任何其他的复位事件均不能改变它。任何硬件事件都不能将 $\overline{\text{POR}}$ 复位为 1。要捕捉多个事件，用户需要在上电复位后，手动在软件中将该位复位为 1。

5.4 欠压复位（BOR）

PIC18F87K22 系列器件有四种 BOR 功耗模式：

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

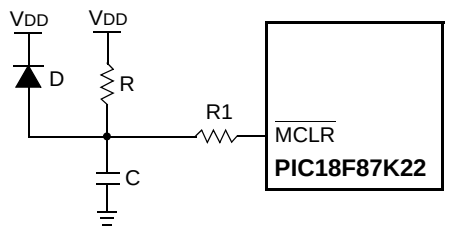
每种功耗模式均通过 BORPWR<1:0> 位（CONFIG2L<6:5>）的设置进行选择。对于低功耗、中等功耗和高功耗 BOR，模块将根据 BORV<1:0>（CONFIG1L<3:2>）的设置监视 VDD。 BOR 事件将重新激活上电复位。还会根据设置的电压跳变点进行复位：1.8V、 2V、 2.7V 或 3V。

BOR 由 BOREN<1:0> 位（CONFIG2L<2:1>）和 SBOREN 位（RCON<6>）使能。典型的能耗如**第 31.0 节 “电气特性”**中的参数 D022A 所示。

在零功耗 BOR（ZPBORMV）模式下，模块监视 VDD 电压，并在电压约为 2V 时重新激活 POR。ZPBORMV 不会导致复位，但是会重新激活 POR。

BOR 精度随着其功耗级别的不同而有所不同。功耗设置越低， BOR 跳变点的精度就越低。因此，高功耗 BOR 的精度最高，低功耗 BOR 的精度最低。跳变点（BVDD，参数 D005）、电流消耗（**第 31.2 节 “直流特性：掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）”**）和低于 BVDD 所需的时间（TBOR，参数 35）都可以在**第 31.0 节 “电气特性”**中找到。

图 5-2： 外部上电复位电路（VDD 慢速上电）



注 1： 仅当 VDD 上电速率过慢时才需要外部上电复位电路。当 VDD 掉电时，二极管 D 帮助电容迅速放电。

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 将限制电流从外部电容 C 流入 $\overline{\text{MCLR}}$，以避免由于静电放电（ESD）或过度电应力（EOS）导致 $\overline{\text{MCLR}}$/VPP 引脚损坏。

5.4.1 检测 BOR

欠压复位或上电复位事件都会使 $\overline{\text{BOR}}$ 位复位为 0。因此仅通过读 $\overline{\text{BOR}}$ 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 $\overline{\text{POR}}$ 和 $\overline{\text{BOR}}$ 的状态。假定在发生任何上电复位事件后，立即用软件将 $\overline{\text{POR}}$ 位复位为 1。如果 $\overline{\text{BOR}}$ 为 0 而 $\overline{\text{POR}}$ 为 1，就可以明确断定已经发生了欠压复位事件。

不能使用 RCON 寄存器中的 $\overline{\text{BOR}}$ 位检测 LP-BOR。LP-BOR 可以重新激活 $\overline{\text{POR}}$，并可能导致发生上电复位。

5.5 配置不匹配 （CM）

配置不匹配（CM）复位用于检测随机存储器损坏事件，并尝试从此事件中恢复。这些事件包括静电放电（ESD）事件，该事件会导致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 PIC18F87K22 系列闪存器件中，在操作期间通过将器件配置寄存器（位于配置存储空间）的值与配对的影子寄存器进行比较来持续监视此配置寄存器。如果在两组寄存器中检测到不匹配现象，则自动发生 CM 复位。这些事件由 $\overline{\text{CM}}$ 位 （RCON<5>）捕捉。每当发生 CM 事件时，该位的状态就会被置为 0，任何其他复位事件均不能改变它。

CM 复位操作与主复位、 RESET 指令、 WDT 超时或堆栈事件复位相似。与所有硬和电源复位事件相同，器件重启时将从程序存储器中的闪存配置字重新加载器件配置字。

5.6 上电延时定时器 （PWRT）

PIC18F87K22 系列器件包含了片上上电延时定时器（PWRT），有助于调节上电复位过程。通过将 PWRTEN 位 （CONFIG2L<0>）置 1 来使能 PWRT。其主要功能是确保在代码执行之前器件电压是稳定的。

PIC18F87K22 系列器件的上电延时定时器 （PWRT）是一个 13 位计数器，使用 LF-INTOSC 时钟源作为时钟输入。产生约 2048 x 32 µs = 66 ms 的时间间隔。当 PWRT 计数时，器件保持在复位状态。

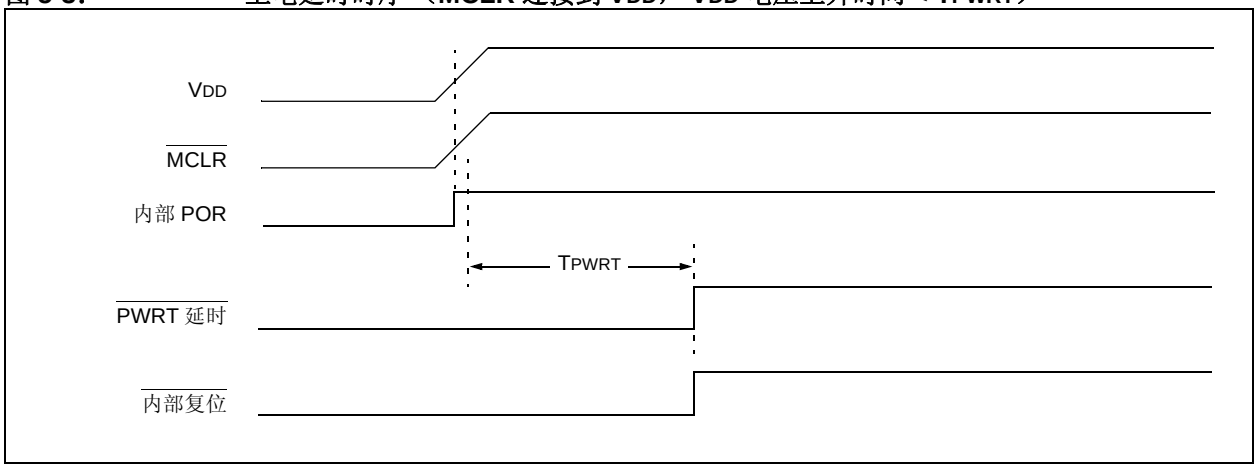
上电延时时间取决于 LF-INTOSC 时钟，并且由于温度和工艺的不同，不同芯片的延时时间也各不相同。详情请参见直流参数 33。

5.6.1 延时时序

在 POR 脉冲被清零后调用 PWRT 延时 （如果使能）。总延时根据 PWRT 的状态而有所不同。图 5-3、图 5-4、图 5-5 和图 5-6 分别描述了上电延时定时器使能时在上电期间的延时顺序。

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， PWRT 将结束。将 MCLR 电平拉高后程序将立即开始执行 （图 5-5）。这对于测试或同步多个并行工作的 PIC18 器件来说非常有用。

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



© 2011 Microchip Technology Inc.

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

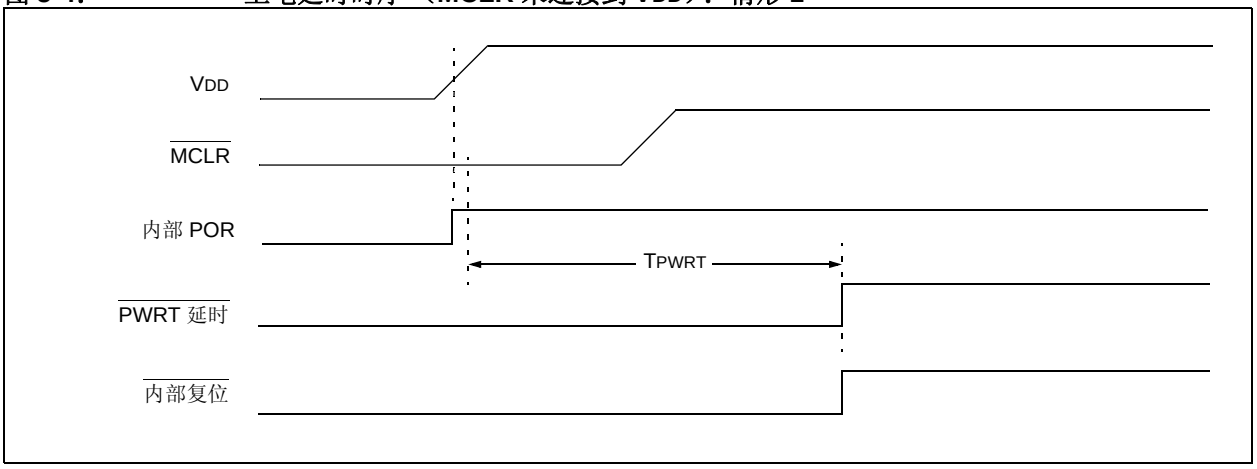


图 5-5： 上电延时时序（$\overline{\text{MCLR}}$ 未连接到 VDD）：情形 2

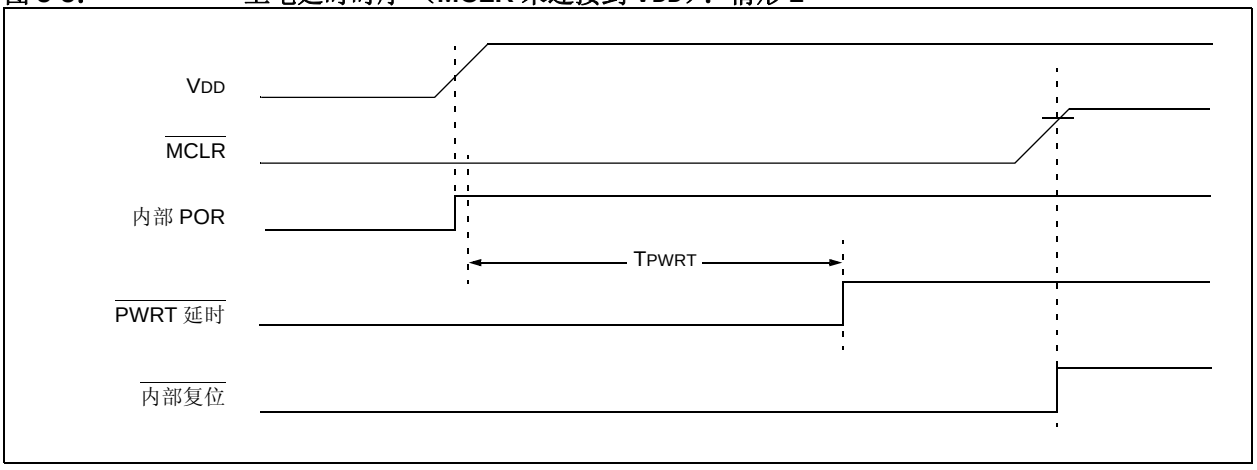
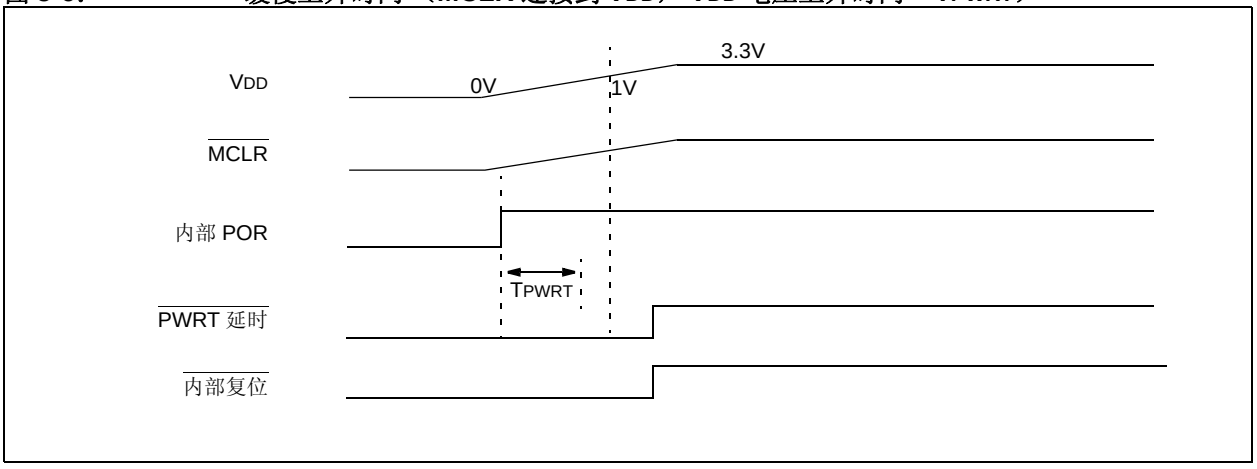


图 5-6： 缓慢上升时间（$\overline{\text{MCLR}}$ 连接到 VDD， VDD 电压上升时间 > TPWRT）



5.7 寄存器的复位状态

大多数寄存器不受复位的影响。在 POR 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发生的不同复位类型被强制为相应的“复位状态”。

因为 WDT 唤醒被视为恢复正常的工作，所以大部分寄存器不受 WDT 唤醒的影响。如表 5-1 所示， RCON 寄存器中的状态位（CM、RI、TO、PD、POR 和 BOR）在不同的复位情形中会分别被置 1 或清零。可在软件中使用这些状态位判断复位的类型。

表 5-2 给出了所有特殊功能寄存器的复位状态。这些复位被分为上电和欠压复位、主复位和 WDT 复位以及 WDT 唤醒复位。

表 5-1： RCON 寄存器的状态位及其含义和初始化状态

条件	程序计数器 (1)	RCON 寄存器						STKPTR 寄存器	
		$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
上电复位	0000h	1	1	1	1	0	0	0	0
RESET 指令	0000h	u	0	u	u	u	u	u	u
欠压复位	0000h	1	1	1	1	u	0	u	u
配置不匹配复位	0000h	0	u	u	u	u	u	u	u
功耗管理运行模式期间的 MCLR 复位	0000h	u	u	1	u	u	u	u	u
功耗管理空闲模式和休眠模式期间的 MCLR 复位	0000h	u	u	1	0	u	u	u	u
全速执行期间的 $\overline{MCLR}$ 复位	0000h	u	u	u	u	u	u	u	u
堆栈满复位 （STVREN= 1）	0000h	u	u	u	u	u	u	1	u
堆栈下溢复位 （STVREN = 1）	0000h	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	u	u	u	u	u	u	1
全速运行或功耗管理运行模式期间的 WDT 超时	0000h	u	u	0	u	u	u	u	u
功耗管理空闲或休眠模式下的 WDT 超时	PC + 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	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 位置 1 时， PC 装入中断向量 （0008h 或 0018h）。

© 2011 Microchip Technology Inc.

表 5-2： 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 CM 复位	通过 WDT 或中断唤醒器件
TOSU	PIC18F6XK22	PIC18F8XK22	---0 0000	---0 0000	---0 uuuu(1)
TOSH	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu(1)
TOSL	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu(1)
STKPTR	PIC18F6XK22	PIC18F8XK22	00-0 0000	uu-0 0000	uu-u uuuu(1)
PCLATU	PIC18F6XK22	PIC18F8XK22	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F6XK22	PIC18F8XK22	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F6XK22	PIC18F8XK22	0000 000x	0000 000u	uuuu uuuu(3)
INTCON2	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu(3)
INTCON3	PIC18F6XK22	PIC18F8XK22	1100 0000	1100 0000	uuuu uuuu(3)
INDF0	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
POSTINC0	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
POSTDEC0	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
PREINC0	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
PLUSW0	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
FSR0H	PIC18F6XK22	PIC18F8XK22	---- 0000	---- 0000	---- uuuu
FSR0L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
POSTINC1	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
POSTDEC1	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
PREINC1	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
PLUSW1	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
FSR1H	PIC18F6XK22	PIC18F8XK22	---- 0000	---- 0000	---- uuuu
FSR1L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F6XK22	PIC18F8XK22	---- 0000	---- 0000	---- uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 取值视情况而定。
阴影单元表示不适用于指定器件。

注 1： 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单元。

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

4： 具体条件下的复位值，请参见表 5-1。

表 5-2： 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 CM 复位	通过 WDT 或中断唤醒器件
INDF2	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
POSTINC2	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
POSTDEC2	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
PREINC2	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
PLUSW2	PIC18F6XK22	PIC18F8XK22	N/A	N/A	N/A
FSR2H	PIC18F6XK22	PIC18F8XK22	---- xxxx	----uuuu	---- uuuu
FSR2L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	PIC18F6XK22	PIC18F8XK22	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F6XK22	PIC18F8XK22	0000 0000	uuuu uuuu	uuuu uuuu
TMR0L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
SPBRGH1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
OSCCON	PIC18F6XK22	PIC18F8XK22	0110 q000	0110 q000	uuuu quuu
IPR5	PIC18F65K22	PIC18F85K22	---1 -111	---1 -111	---u -uuu
IPR5	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	1000 0000	1000 0000	uuuu uuuu
WDTCON	PIC18F6XK22	PIC18F8XK22	0-x0 -000	0-x0 -000	u-uu -uuu
RCON	PIC18F6XK22	PIC18F8XK22	0111 11qq	0uqq qquu	uuuu qquu
TMR1H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F6XK22	PIC18F8XK22	0000 0000	uuuu uuuu	uuuu uuuu
TMR2	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
T2CON	PIC18F6XK22	PIC18F8XK22	-000 0000	-000 0000	-uuu uuuu
SSP1BUF	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP1ADD	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
SSP1STAT	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
SSP1CON1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
SSP1CON2	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
ADRESH	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F6XK22	PIC18F8XK22	-000 0000	-000 0000	-uuu uuuu
ADCON1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
ADCON2	PIC18F6XK22	PIC18F8XK22	0-00 0000	0-00 0000	u-uu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 取值视情况而定。
阴影单元表示不适用于指定器件。

注 1： 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单元。

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

4： 具体条件下的复位值，请参见表 5-1。

© 2011 Microchip Technology Inc.

表 5-2： 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 CM 复位	通过 WDT 或中断唤醒器件
ECCP1AS	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
ECCP1DEL	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
CCPR1H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PIR5	PIC18F65K22	PIC18F85K22	---0 -000	---0 -000	---u -uuu
PIR5	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu
PIE5	PIC18F65K22	PIC18F85K22	---0 0000	---0 0000	---u uuuu(1)
PIE5	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 000	0000 0000	uuuu uuuu(1)
IPR4	PIC18F65K22	PIC18F85K22	--11 1111	--11 1111	--uu uuuu
IPR4	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	1111 1111	1111 1111	uuuu uuuu
PIR4	PIC18F65K22	PIC18F85K22	--00 0000	--00 0000	--uu uuuu(1)
PIR4	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu(1)
PIE4	PIC18F65K22	PIC18F85K22	--00 0000	--00 0000	--uu uuuu
PIE4	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu
CVRCON	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
CMSTAT	PIC18F6XK22	PIC18F8XK22	xxx- ----	xxx- ----	uuu- ----
TMR3H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0x00	uuuu uuuu
T3GCON	PIC18F6XK22	PIC18F8XK22	0000 0x00	0000 0000	uuuu uuuu
SPBRG1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXSTA1	PIC18F6XK22	PIC18F8XK22	0000 0010	0000 0010	uuuu uuuu
RCSTA1	PIC18F6XK22	PIC18F8XK22	0000 000x	0000 000x	uuuu uuuu
T1GCON	PIC18F6XK22	PIC18F8XK22	0000 0x00	0000 0x00	uuuu -uuu
IPR6	PIC18F6XK22	PIC18F8XK22	---1 -111	---1 -111	---u -uuu
HLVDCON	PIC18F6XK22	PIC18F8XK22	0000 0101	0000 0101	uuuu uuuu
PSPCON	PIC18F6XK22	PIC18F8XK22	0000 ----	0000 ----	uuuu ----
PIR6	PIC18F6XK22	PIC18F8XK22	---0 -000	---0 -000	---u -uuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 取值视情况而定。
阴影单元表示不适用于指定器件。

注 1： 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单元。

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

4： 具体条件下的复位值，请参见表 5-1。

表 5-2： 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 CM 复位	通过 WDT 或中断唤醒器件
IPR3	PIC18F6XK22	PIC18F8XK22	1-11 1111	1-11 1111	u-uu uuuu
PIR3	PIC18F6XK22	PIC18F8XK22	0-00 0000	0-00 0000	u-uu uuuu
PIE3	PIC18F6XK22	PIC18F8XK22	0-00 0000	0-00 0000	u-uu uuuu
IPR2	PIC18F6XK22	PIC18F8XK22	1-11 1111	1-11 1111	u-uu uuuu
PIR2	PIC18F6XK22	PIC18F8XK22	0-10 0000	0-10 0000	u-uu uuuu
PIE2	PIC18F6XK22	PIC18F8XK22	0-00 0000	0-00 0000	u-uu uuuu
IPR1	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
PIR1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PIE1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PSTR1CON	PIC18F6XK22	PIC18F8XK22	00-0 0001	00-0 0001	uu-u uuuu
OSCTUNE	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
TRISJ	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
TRISH	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
TRISG	PIC18F6XK22	PIC18F8XK22	---1 1111	---1 1111	---u uuuu
TRISF	PIC18F6XK22	PIC18F8XK22	1111 111-	1111 111-	---u uuuu
TRISE	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
TRISD	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
TRISA	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
LATJ	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG	PIC18F6XK22	PIC18F8XK22	---x xxxx	---u uuuu	---u uuuu
LATF	PIC18F6XK22	PIC18F8XK22	xxxx xxx-	uuuu uuu-	uuuu uuu-
LATE	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTJ	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTH	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTG	PIC18F6XK22	PIC18F8XK22	--xx xxxx	--xx xxxx	--uu uuuu
PORTF	PIC18F6XK22	PIC18F8XK22	xxxx xxx-	xxxx xxx-	uuuu uuu-
PORTE	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTD	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	xxxx xxxx	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 取值视情况而定。
阴影单元表示不适用于指定器件。

注 1： 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单元。

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

4： 具体条件下的复位值，请参见表 5-1。

© 2011 Microchip Technology Inc.

表 5-2： 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 CM 复位	通过 WDT 或中断唤醒器件
PORTC	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	PIC18F6XK22	PIC18F8XK22	xx0x 0000	uu0u 0000	uuuu uuuu
EECON1	PIC18F6XK22	PIC18F8XK22	xx-0 x000	uu-0 u000	uu-u uuuu
EECON2	PIC18F6XK22	PIC18F8XK22	---- ----	---- ----	---- ----
TMR5H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR5L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
T5CON	PIC18F6XK22	PIC18F8XK22	0000 0000	uuuu uuuu	uuuu uuuu
T5GCON	PIC18F6XK22	PIC18F8XK22	0000 0x00	uuuu uuuu	uuuu uuuu
CCPR4H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR4L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP4CON	PIC18F6XK22	PIC18F8XK22	--00 0000	--00 0000	--uu uuuu
CCPR5H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR5L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP5CON	PIC18F6XK22	PIC18F8XK22	--00 0000	--00 0000	--uu uuuu
CCPR6H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR6L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP6CON	PIC18F6XK22	PIC18F8XK22	--00 0000	--00 0000	--uu uuuu
CCPR7H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR7L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP7CON	PIC18F6XK22	PIC18F8XK22	--00 0000	--00 0000	--uu uuuu
TMR4	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	xxxx xxxx	uuuu uuuu
PR4	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	1111 1111
T4CON	PIC18F6XK22	PIC18F8XK22	-111 1111	-111 1111	-uuu uuuu
SSP2BUF	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP2ADD	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
SSP2STAT	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
SSP2CON1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
SSP2CON2	PIC18F6XK22	PIC18F8XK22	0100 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F6XK22	PIC18F8XK22	0100 0-00	0100 0-00	uuuu u-uu
OSCCON2	PIC18F6XK22	PIC18F8XK22	-0-- 0-x0	-0-- 0-u0	-u-- u-uu
EEADRH	PIC18F6XK22	PIC18F8XK22	---- --00	---- --00	---- --uu
EEADR	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
EEDATA	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PIE6	PIC18F6XK22	PIC18F8XK22	---0 -000	---0 -000	---u -uuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 取值视情况而定。
阴影单元表示不适用于指定器件。

注 1： 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单元。

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

4： 具体条件下的复位值，请参见表 5-1。

表 5-2： 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 CM 复位	通过 WDT 或中断唤醒器件
RTCCFG	PIC18F6XK22	PIC18F8XK22	0-00 0000	u-uu uuuu	u-uu uuuu
RTCCAL	PIC18F6XK22	PIC18F8XK22	0000 0000	uuuu uuuu	uuuu uuuu
RTCVALH	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
RTCVALL	PIC18F6XK22	PIC18F8XK22	0000 0000	uuuu uuuu	uuuu uuuu
ALRMCFG	PIC18F6XK22	PIC18F8XK22	0000 0000	uuuu uuuu	uuuu uuuu
ALRMRPT	PIC18F6XK22	PIC18F8XK22	0000 0000	uuuu uuuu	uuuu uuuu
ALRMVALH	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
ALRMVALL	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CTMUCONH	PIC18F6XK22	PIC18F8XK22	0-00 0000	0-00 0000	u-uu uuuu
CTMUCONL	PIC18F6XK22	PIC18F8XK22	0000 00xx	0000 00xx	uuuu uuuu
CTMUICONH	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
CM1CON	PIC18F6XK22	PIC18F8XK22	0001 1111	0001 1111	uuuu uuuu
PADCFG1	PIC18F6XK22	PIC18F8XK22	00-- -00-	uu-- -uu-	uu-- -uu-
PADCFG1	PIC18F6XK22	PIC18F8XK22	000- -00-	uuu- -uu-	uuu- -uu-
ECCP2AS	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
ECCP2DEL	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
CCPR2H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
ECCP3AS	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
ECCP3DEL	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
CCPR3H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR3L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP3CON	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
CCPR8H	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR8L	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP8CON	PIC18F6XK22	PIC18F8XK22	--00 0000	--00 0000	--uu uuuu
CCPR9H	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR9L	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP9CON	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	--00 0000	--00 0000	--uu uuuu
CCPR10H	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR10L	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 取值视情况而定。
阴影单元表示不适用于指定器件。

注 1： 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单元。

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

4： 具体条件下的复位值，请参见表 5-1。

© 2011 Microchip Technology Inc.

表 5-2： 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 CM 复位	通过 WDT 或中断唤醒器件
CCP10CON	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	--00 0000	--00 0000	--uu uuuu
TMR7H	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR7L	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	xxxx xxxx	uuuu uuuu	uuuu uuuu
T7CON	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	uuuu uuuu	uuuu uuuu
T7GCON	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0x00	0000 0x00	uuuu uuuu
TMR6	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PR6	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
T6CON	PIC18F6XK22	PIC18F8XK22	-000 0000	-000 0000	-uuu uuuu
TMR8	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PR8	PIC18F6XK22	PIC18F8XK22	1111 1111	1111 1111	uuuu uuuu
T8CON	PIC18F6XK22	PIC18F8XK22	-000 0000	-000 0000	-uuu uuuu
TMR10	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu
PR10	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	1111 1111	1111 1111	uuuu uuuu
T10CON	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	-000 0000	-000 0000	-uuu uuuu
TMR12	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu
PR12	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	1111 1111	1111 1111	uuuu uuuu
T12CON	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	-000 0000	-000 0000	-uuu uuuu
CM2CON	PIC18F6XK22	PIC18F8XK22	0001 1111	0001 1111	uuuu uuuu
CM3CON	PIC18F6XK22	PIC18F8XK22	0001 1111	0001 1111	uuuu uuuu
CCPTMRS0	PIC18F6XK22	PIC18F8XK22	0000 0000	uuuu uuuu	uuuu uuuu
CCPTMRS1	PIC18F6XK22	PIC18F8XK22	00-0 -000	uu-u -uuu	uu-u -uuu
CCPTMRS2	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	---0 -000	---u -uuu	---u -uuu
CCPTMRS2	PIC18F65K22	PIC18F85K22	---- --00	---- --uu	---- --uu
REFOCON	PIC18F6XK22	PIC18F8XK22	0-00 0000	u-uu uuuu	u-uu uuuu
ODCON1	PIC18F6XK22	PIC18F8XK22	000- ---0	uuu- ---u	uuu- ---u

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 取值视情况而定。
阴影单元表示不适用于指定器件。

注 1： 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单元。

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

4： 具体条件下的复位值，请参见表 5-1。

表 5-2： 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 CM 复位	通过 WDT 或中断唤醒器件
ODCON2	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	uuuu uuuu	uuuu uuuu
ODCON2	PIC18F65K22	PIC18F85K22	--00 0000	--uu uuuu	--uu uuuu
ODCON3	PIC18F6XK22	PIC18F8XK22	00-- ---0	uu-- ---u	uu-- ---u
MEMCON	PIC18F6XK22	PIC18F8XK22	0-00 --00	0-00 --00	u-uu --uu
ANCON0	PIC18F6XK22	PIC18F8XK22	1111 1111	uuuu uuuu	uuuu uuuu
ANCON1	PIC18F6XK22	PIC18F8XK22	1111 1111	uuuu uuuu	uuuu uuuu
ANCON2	PIC18F6XK22	PIC18F8XK22	1111 1111	uuuu uuuu	uuuu uuuu
RCSTA2	PIC18F6XK22	PIC18F8XK22	0000 000x	0000 000x	uuuu uuuu
TXSTA2	PIC18F6XK22	PIC18F8XK22	0000 0010	0000 0010	uuuu uuuu
BAUDCON2	PIC18F6XK22	PIC18F8XK22	0100 0-00	0100 0-00	uuuu u-uu
SPBRGH2	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
SPBRG2	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F6XK22	PIC18F8XK22	xxxx xxxx	xxxx xxxx	uuuu uuuu
PSTR2CON	PIC18F6XK22	PIC18F8XK22	00-0 0001	00-0 0001	uu-u uuuu
PSTR3CON	PIC18F6XK22	PIC18F8XK22	00-0 0001	00-0 0001	uu-u uuuu
PMD0	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PMD1	PIC18F6XK22	PIC18F8XK22	0000 0000	0000 0000	uuuu uuuu
PMD2	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu
PMD2	PIC18F65K22	PIC18F85K22	-0-0 0000	-0-0 0000	-u-u uuuu
PMD3	PIC18F66K22 PIC18F67K22	PIC18F86K22 PIC18F87K22	0000 0000	0000 0000	uuuu uuuu
PMD3	PIC18F65K22	PIC18F85K22	--00 000-	--00 000-	--uu uuu-

图注： u = 不变， x = 未知， - = 未实现位，读为 0， q = 取值视情况而定。
阴影单元表示不适用于指定器件。

注 1： 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单元。

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

4： 具体条件下的复位值，请参见表 5-1。

© 2011 Microchip Technology Inc.

6.0 存储器构成

PIC18F87K22 系列器件有以下类型的存储器：

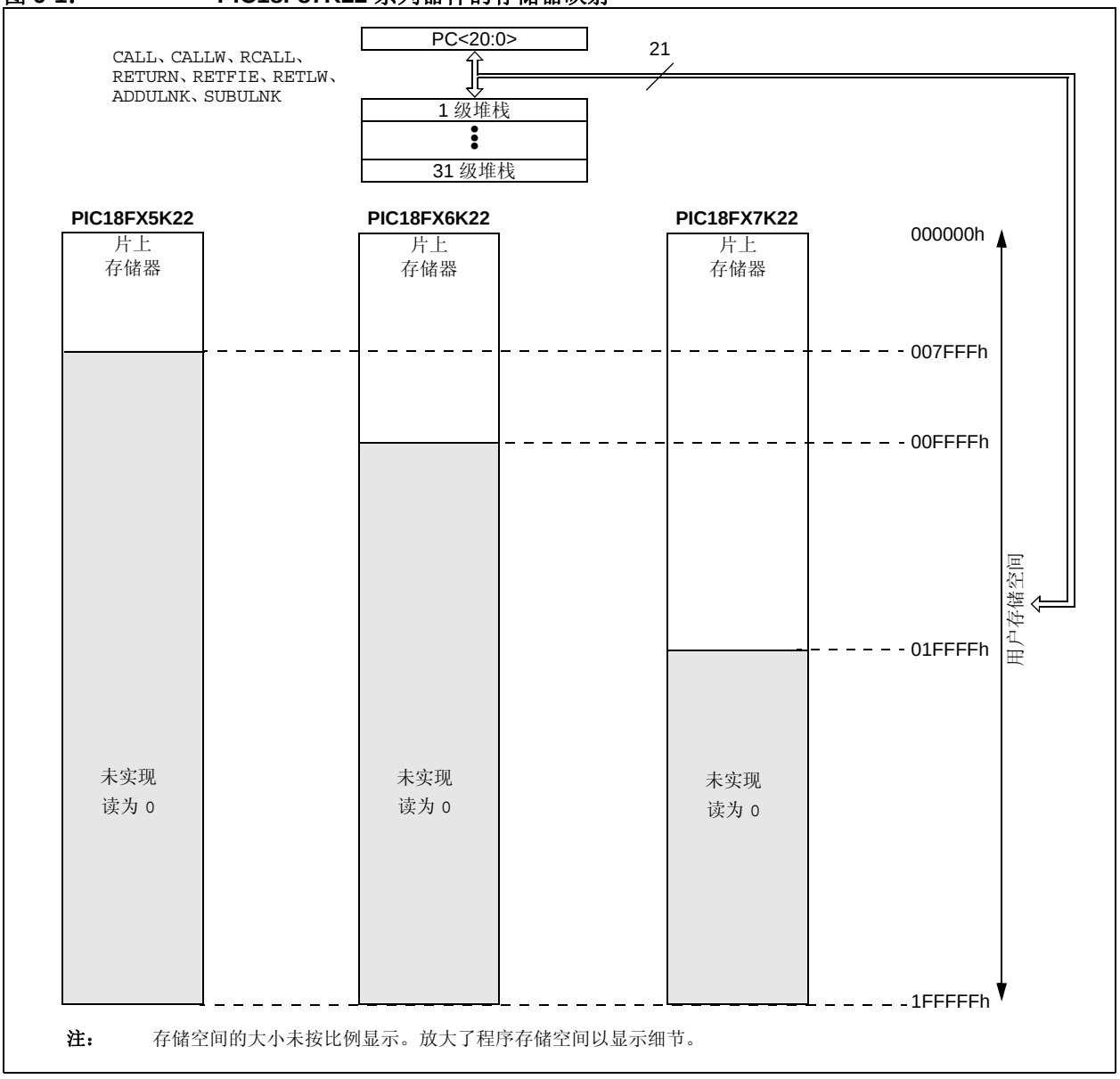
- 程序存储器
- 数据 RAM
- 数据 EEPROM

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

从实际用途而言，可将数据 EEPROM 看作外设，因为它是通过一组控制寄存器来寻址和访问的。

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。第 9.0 节 “数据 EEPROM 存储器” 中将单独讨论数据 EEPROM。

图 6-1： PIC18F87K22 系列器件的存储器映射



注： 存储空间的大小未按比例显示。放大了程序存储空间以显示细节。

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。访问物理实现的存储器的上边界和 2 MB 地址之间的存储单元将会返回全 0 （NOP 指令）。

整个 PIC18F87K22 系列器件提供了一系列容量的片上闪存程序存储器，从 32 KB（至多 16,384 条单字指令）到 128 KB （65,536 条单字指令）。

- PIC18F65K22 和 PIC18F85K22——32 KB 的闪存存储器，至多存储 16,384 条单字指令
- PIC18F66K22 和 PIC18F86K22——64 KB 的闪存存储器，至多存储 32,768 条单字指令
- PIC18F67K22 和 PIC18F87K22——128 KB 的闪存存储器，至多存储 65,536 条单字指令

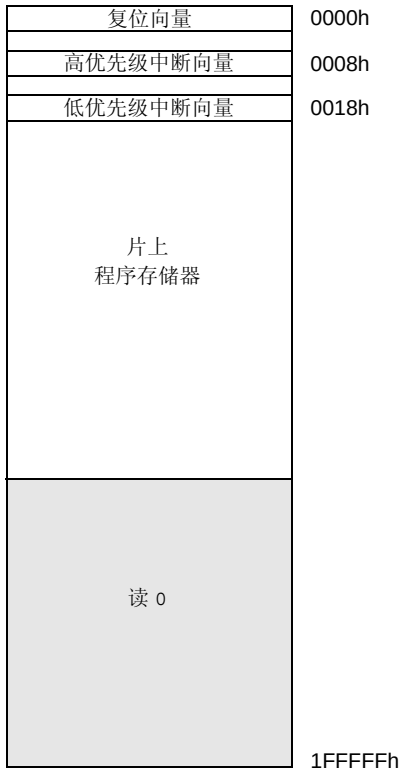
该系列各器件的程序存储器映射如图 6-1 所示。

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。这些向量在程序存储器映射中的相对位置如图 6-2 所示。

图 6-2： PIC18F87K22 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 （有关具体器件的值，请参见图 6-1）。阴影区域代表未实现的存储区。这些区域未按比例显示。

© 2011 Microchip Technology Inc.

6.1.2 程序计数器

程序计数器（Program Counter，PC）指定要取出执行的指令的地址。PC 为 21 位宽，保存在 3 个独立的 8 位寄存器中。

存储低字节的寄存器称为 PCL 寄存器，该寄存器是可读写的。存储高字节的寄存器即 PCH 寄存器，存储 PC<15:8> 位，不可直接读写。可以通过 PCLATH 寄存器更新 PCH 寄存器。存储最高字节的寄存器称为 PCU。该寄存器存储 PC<20:16> 位，它也不可以直接读写。可以通过 PCLATU 寄存器更新 PCU 寄存器。

通过执行写 PCL 的操作，可以将 PCLATH 和 PCLATU 的内容传送到程序计数器。类似的，通过执行读 PCL 的操作，可以将程序计数器的两个高字节传送到 PCLATH 和 PCLATU。这对于计算 PC 的偏移量很有用（见第 6.1.5.1 节 “计算 GOTO”）。

PC 按字节寻址程序存储器。为防止 PC 不能正确获取指令字，需要将 PCL 的最低有效位固定取值为 0。PC 以 2 为增量递增来连续寻址程序存储器中的指令。

CALL、RCALL、GOTO 和程序转移指令直接将地址写入程序计数器。对于这些指令，PCLATH 和 PCLATU 的内容将不会被传送到程序计数器。

6.1.3 返回地址堆栈

返回地址堆栈可保存最多31次程序调用和中断时的PC值。当执行 CALL 或 RCALL 指令或响应中断时，PC 值被压入堆栈。而当执行 RETURN、RETLW 或 RETFIE 指令（若使能了扩展指令集，则还包括 ADDULNK 和 SUBULNK 指令）时，PC 值从堆栈弹出。PCLATU 和 PCLATH 不受 RETURN 或 CALL 指令的影响。

通过 21 位的 RAM 和 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。堆栈指针可以读写，并且通过栈顶的特殊功能寄存器可以读写栈顶地址。也可使用这些寄存器将数据压入堆栈，或将数据从堆栈弹出。

执行 CALL 类型指令引起进栈操作。首先堆栈指针加 1，并且将 PC 的内容写入堆栈指针指向的单元（PC 已指向 CALL 的下一条指令）。执行 RETURN 类型的指令时，引起出栈操作。STKPTR 所指向的单元的内容会被传送给 PC，然后堆栈指针减 1。

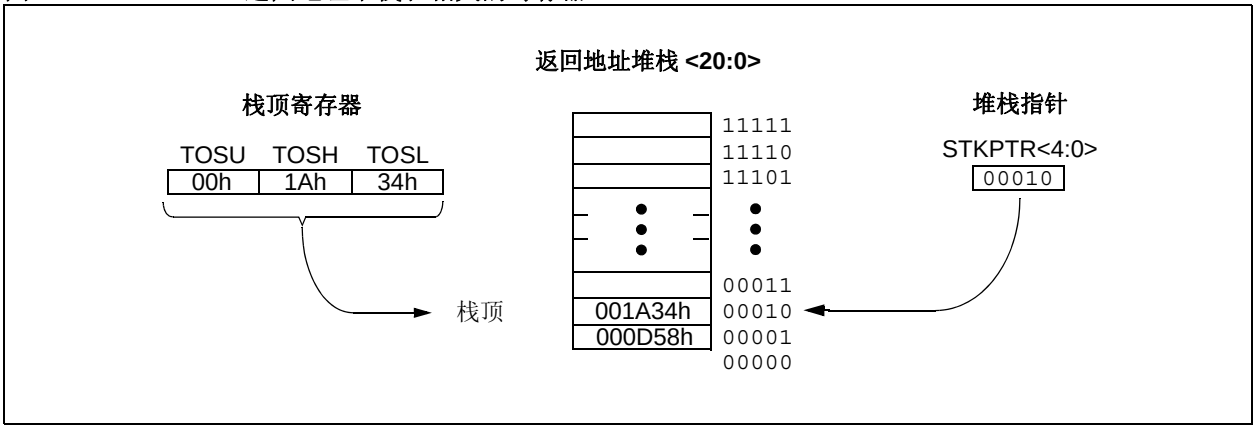
所有复位后，堆栈指针被初始化为 00000。堆栈指针值 00000 不指向任何 RAM 单元，它只是一个复位值。状态位表明堆栈是满、上溢还是下溢。

6.1.3.1 访问栈顶

只有返回地址堆栈的栈顶（Top-of-Stack，TOS）是可读写的。3 个寄存器 TOSU:TOSH:TOSL 用于保存 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 寄存器来读取进栈值。这些值可以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。

为防止对堆栈的意外操作，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 到 31 之间的值。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。

当向堆栈压入 PC 值 31 次（且没有值从堆栈弹出）后，STKFUL位就会置1。通过软件或POR清零STKFUL位。

堆栈满时执行的操作由 STVREN （堆栈上溢复位使能）配置位的状态决定。（有关器件配置位的说明，请参见**第 28.1 节 “配置位”**。）如果 STVREN 位已经置 1 （默认），第 31 次进栈将把 (PC + 2) 处的值压入堆栈，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 位被清零，第 31 次进栈时 STKFUL 位会被置 1，堆栈指针则加 1 变为 31。后续进栈操作不会覆盖第 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 0。STKUNF 位将保持置 1，直到被软件清零或发生 POR。

> **注：** 下溢时，将零值返回给 PC，会使程序执行指向复位向量，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 SFR 的内容不受影响。

6.1.3.3 PUSH 和 POP 指令

因为栈顶（TOS）是可以读写的，因此将值压入堆栈或从堆栈弹出值而不影响程序的正常执行是非常理想的。PIC18 指令集包括两条指令 PUSH 和 POP，使用这 2 条指令可在软件控制下对 TOS 进行操作。可以通过修改 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。这将使堆栈指针加 1，并将当前 PC 值装入堆栈。

POP 指令通过将堆栈指针减 1 来丢弃当前的 TOS 值。然后前一个进栈值成为了 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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **STKFUL：**堆栈满标志位 (1)
1 = 堆栈满或上溢
0 = 堆栈未满或未上溢

bit 6 **STKUNF：**堆栈下溢标志位 (1)
1 = 发生堆栈下溢
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。

© 2011 Microchip Technology Inc.

6.1.3.4 堆栈满和下溢复位

通过将 STVREN 位（CONFIG4L<0>）置 1 可以使得器件在出现堆栈上溢和堆栈下溢条件时复位。当 STVREN 位置 1 时，堆栈满或堆栈下溢条件会将相应的 STKFUL 或 STKUNF 位置 1，然后使器件复位。当 STVREN 位清零时，堆栈满或堆栈下溢条件会将相应的 STKFUL 或 STKUNF 位置 1，但不会使器件复位。通过用户软件或上电复位清零 STKFUL 或 STKUNF 位。

6.1.4 快速寄存器堆栈

为 STATUS、WREG 和 BSR 寄存器提供了快速寄存器堆栈，从而可实现从中断“快速返回”。此堆栈深度仅为 1 级，并且不可读写。当处理器转入中断向量处执行时，此堆栈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，这些寄存器中的值会被重新装回相应的工作寄存器。

如果同时允许了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，用户必须在低优先级中断期间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。要将快速寄存器堆栈用于子程序调用，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

可能在有些编程场合中需要在程序存储器中创建数据结构或查找表。对于 PIC18 器件，有两种方法可以实现查找表：

- 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 给出了一个示例。

使用 ADDWF PCL 指令和一组 RETLW nn 指令可以创建一个查找表。在调用该表前，会先将查找表的偏移量装入 W 寄存器。被调用程序的第一条指令是 ADDWF PCL 指令。接下去执行的是一条 RETLW nn 指令，它将数值 nn 返回给调用函数。

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

在这种方法中，每个指令单元只能存储一个数据字节，并且要求返回地址堆栈还有空闲单元。

例 6-2：使用偏移量的计算 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，该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。

在第 7.1 节“表读和表写”中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信号都会在器件内部被 4 分频用来产生 4 个不重叠的正交时钟信号，即 Q1、Q2、Q3 和 Q4。在内部，程序计数器在每个 Q1 递增，并在 Q4 期间，从程序存储器取指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 到 Q4 周期完成。图 6-4 所示为时钟和指令执行的流程图。

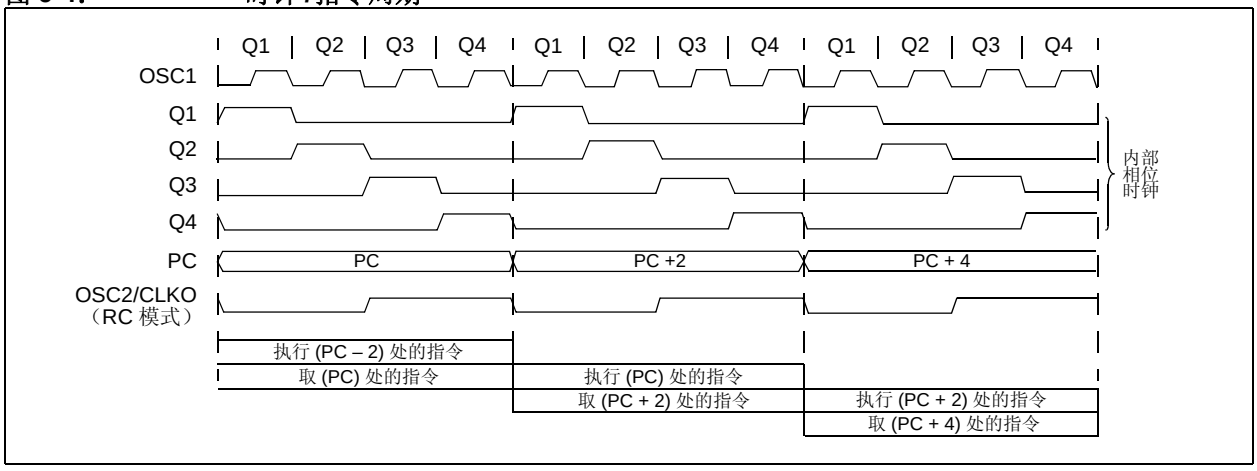
6.2.2 指令流 / 流水线

一个“指令周期”由 4 个 Q 周期组成（即 Q1 到 Q4）。指令的取指和执行是以流水线的形式进行的，在一个指令周期进行取指，而在下一个指令周期译码并执行指令。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 指令）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。）

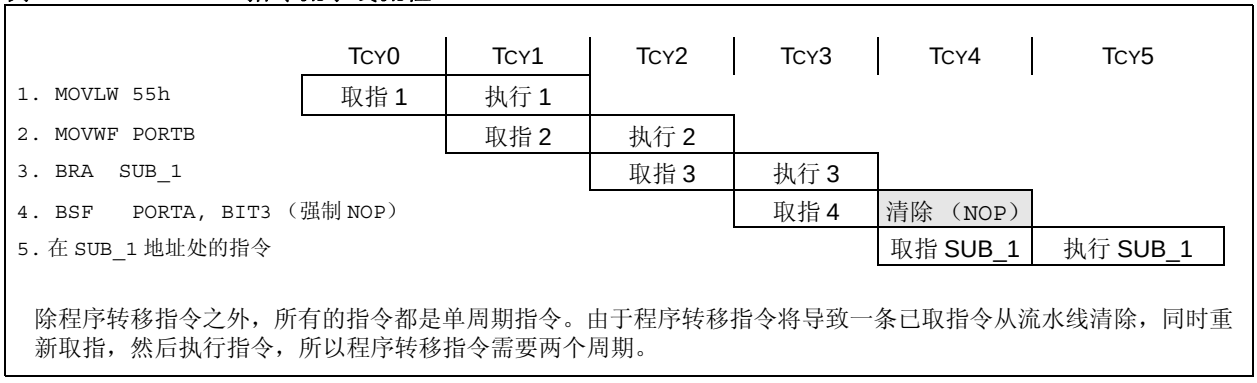
在 Q1 周期，程序计数器 （PC）递增，开始取指。

在指令的执行周期，操作如下：在 Q1 周期，将所取指令锁存到指令寄存器 （IR）。然后在 Q2、Q3 和 Q4 周期中进行指令的译码和执行。其中读数据存储器（读操作数）发生在 Q2 周期，写操作 （写目标单元）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



除程序转移指令之外，所有的指令都是单周期指令。由于程序转移指令将导致一条已取指令从流水线清除，同时重新取指，然后执行指令，所以程序转移指令需要两个周期。

© 2011 Microchip Technology Inc.

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 字节形式存储在程序存储器中。指令字的最低有效字节始终存储在地址为偶数的程序存储单元中 （LSB = 0）。要保证正确指向指令单元， 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 给出了如何将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。指令总是存储为一个字长，因而指令所包含的数据为字地址。字地址会写入 PC<20:1>，由 PC 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。程序转移指令也采取同样的方式对相对地址偏移量进行译码。转移指令中的偏移量值代表单字指令数，PC 将以此作为偏移量跳转到指定的地址单元。欲知指令集的更多详情，请参见**第 29.0 节 “指令集汇总”**。

图 6-5： 程序存储器中的指令

			LSB = 1	LSB = 0	字地址 ↓
程序存储器字节单元 →					000000h
					000002h
					000004h
					000006h
指令 1：	MOVLW	055h	0Fh	55h	000008h
指令 2：	GOTO	0006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
指令 3：	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					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。在所有情况下，这些指令第二个字的高 4 位总是 1111，而其余的 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 1111 代表一条特殊的 NOP 指令。指令的正确执行顺序为：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并自行执行指令的第二个字，那么将转而执行一条 NOP 指令。如果双字指令跟在更改 PC 的条件指令后，就有必要执行此操作。例 6-4 说明了其执行过程。

> **注：** 欲知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 PIC18 扩展指令集时，数据存储器某些方面的操作会有所更改。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以静态 RAM 的方式实现。在数据存储器中，每个寄存器都有一个 12 位的地址，数据存储器容量最高可达 4096 个字节。存储空间被分为 16 个存储区，每个存储区包含 256 个字节。PIC18FX6K22 和 PIC18FX7K22 器件实现了所有 16 个完整的存储区，总共 4 KB。PIC18FX5K22 器件仅实现了前 8 个完整存储区，总共 2 KB。

图 6-6 和图 6-7 显示了器件的数据存储器构成。

数据存储器由特殊功能寄存器（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显示，而 GPR 则用于存储用户应用程序所用数据和高速暂存。任何未实现单元都将读为 0。

此指令集和架构支持跨存储区的操作。可以通过直接、间接或变址寻址模式访问整个数据存储器。本节后面的部分将讨论寻址模式。

为了确保能在一个周期内存取常用寄存器（选定的 SFR 和 GPR），PIC18 器件实现了一个快速操作存储区。这是一个256字节的存储空间，可实现对选定SFR和GPR Bank 0 的低地址部分的快速存取，而无需使用存储区选择寄存器。有关快速操作 RAM 的详情，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

存储空间很大的数据存储器需要有效的寻址机制，以对所有地址进行快速存取。理想状况下，这意味着不必为每次读写操作提供整个地址。 PIC18 器件是使用 RAM 分区机制实现快速存取的。该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、或通过 8 位低字节地址和 4 位存储区指针直接寻址每个地址单元。

PIC18 指令集中的大部分指令都使用存储区指针，即存储区选择寄存器 （Bank Select Register， BSR）。此 SFR 保存存储单元地址的高 4 位。而指令本身则包括低 8 位。仅实现 BSR 的低 4 位 （BSR<3:0>），其高 4 位未用，始终读为 0 且不能被写入。可以通过 MOVLB 指令直接装入 BSR。

BSR 的值指向数据存储器中的存储区。指令中的 8 位地址表示存储区中的存储单元，可以将它看作是以存储区的下边界为起点的偏移量。图 6-7 所示是 BSR 的值与数据存储器中的分区之间的关系。

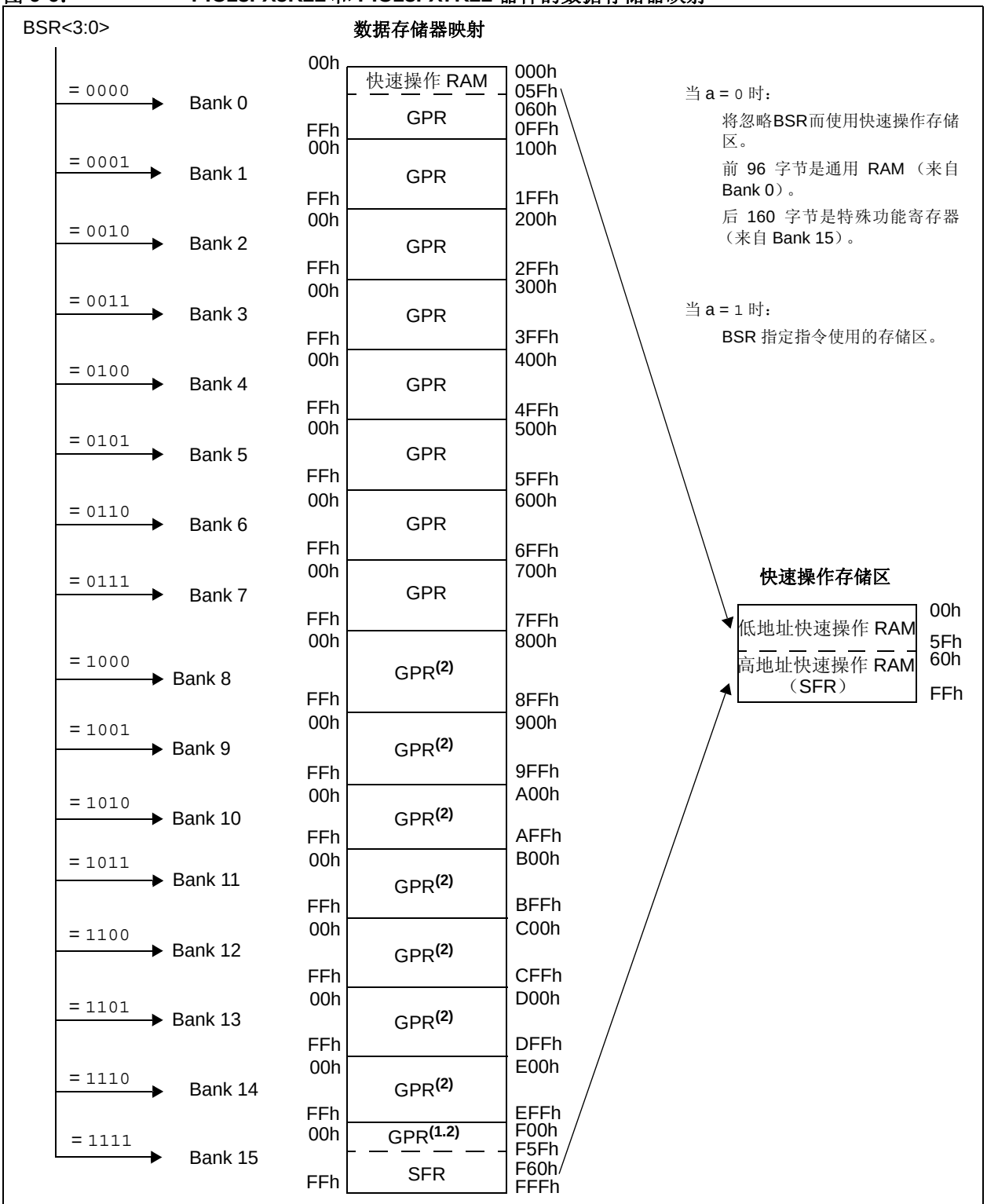
由于最多有 16 个寄存器共享同一个低位地址，用户必须始终非常小心以确保在执行数据读或写之前选择了正确的存储区。例如，当 BSR 为 0Fh 时将程序数据写入地址为 F9h 的 8 位地址单元，将导致程序计数器复位。

虽然可以选择任何存储区，但只有实际实现的存储区才可以被读写。对未实现的存储区的写入操作将被忽略，而读未实现的存储区会返回 0。虽然是这样， STATUS 寄存器仍然会受到影响，如同成功执行了读写操作一样。图6-6中的数据存储器映射指出了已实现的存储区。

在 PIC18 的核心指令集中，只有 MOVFF 指令指定源寄存器和目标寄存器的完整 12 位地址。此指令在执行时完全忽略 BSR 的值。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。

© 2011 Microchip Technology Inc.

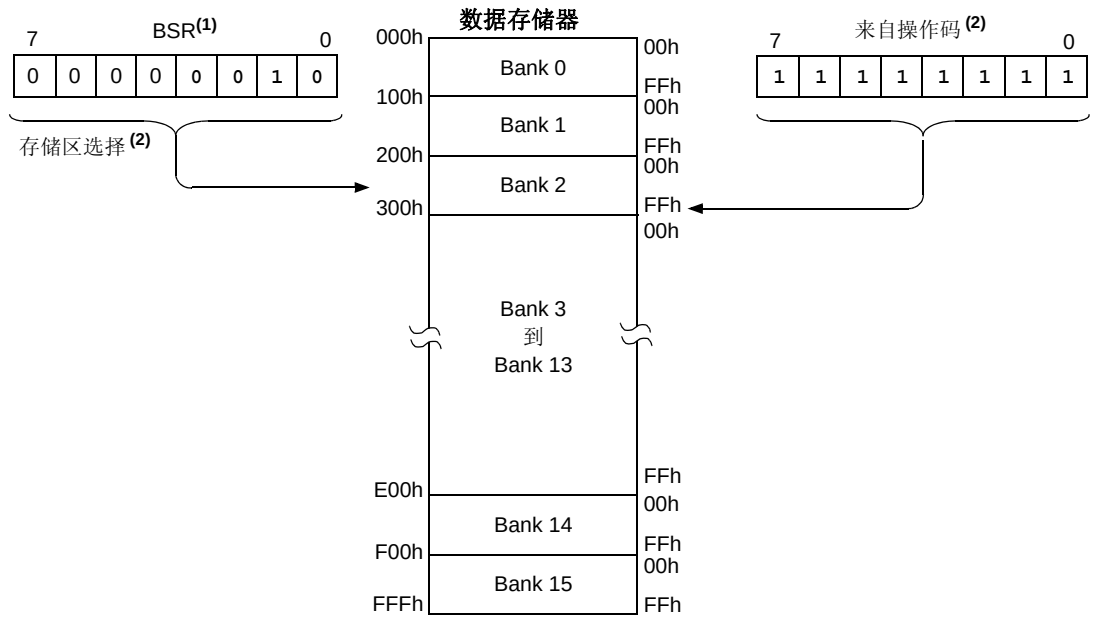
图 6-6： PIC18FX5K22 和 PIC18FX7K22 器件的数据存储器映射



注 1： 地址 F16h 到 F5Fh 也由 SFR 使用，但是不属于快速操作 RAM。用户必须始终使用完整的地址或装入正确的 BSR 值来访问这些寄存器。

2： 具有 32 KB 程序存储器的器件（PIC18FX5K22）未使用这些地址。对于这些器件，在 00h 处读取这些地址。

图 6-7： 使用存储区选择寄存器 （直接寻址）



注 1： 可以使用指令的快速操作 RAM 位强制忽略选定的存储区 （由 BSR<3:0> 选定）直接寻址快速操作存储区的寄存器。

2： MOVFF 指令嵌入了完整的 12 位地址。

6.3.2 快速操作存储区

使用 BSR 和指令内嵌的 8 位地址可以使用户对整个数据存储空间进行寻址，这同时意味着用户必须始终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。如果本来是对 GPR 进行写操作，却将结果写入了 SFR，后果将是非常严重的。但是在每次向数据存储器进行读或写操作时，验证和 / 或更改 BSR 可能会降低代码的执行效率。

为了连续访问大多数常用的数据存储单元，必须为数据存储器配置快速操作存储区，这使得用户无需指定 BSR 即可访问被映射的存储区。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。低半部分就是 “快速操作 RAM”，由 GPR 组成。高半部分被映射为器件的 SFR。这两个区域可以在快速操作存储区中连续映射，并且可以用 8 位地址进行线性寻址 （图 6-6）。

快速操作存储区供包括快速操作 RAM 位 （指令中的 a 参数）的 PIC18 核心指令使用。当 a 等于 1 时，指令使用 BSR 和包含在操作码中的 8 位地址来对数据存储器进行寻址。但是当 a 为 0 时，指令被强制使用快速操作存储区地址映射。在此情况下，BSR 的当前值被完全忽略。

该 “强制” 寻址方式可使指令在一个周期内对数据地址进行操作，而无需首先更新 BSR。这意味着用户可以更有效地对 8 位地址为 60h 及以上的 SFR 进行求值和操作。地址低于 60h 的快速操作 RAM 非常适合于存储那些用户可能需要快速存取的数据值（如直接计算结果或常用程序变量）。

快速操作 RAM 还可以实现更快速、代码效率更高的现场保护和变量切换。

当使能扩展指令集 （XINST 配置位 = 1）时，快速操作存储区的映射略有不同。第 6.6.3 节 “在立即数变址寻址模式下映射快速操作存储区” 中将对此进行更详细的说明。

6.3.3 通用寄存器

PIC18 器件可能在 GRP 区中划分了一部分存储区。这部分存储区为数据 RAM，所有指令都可以访问它。GPR 区从 Bank 0 的底部 （地址 000h）开始向上直到 SFR 区的底部。上电复位不会初始化 GPR，其他复位也不会改变其内容。

© 2011 Microchip Technology Inc.

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器件操作的寄存器。这类寄存器以静态 RAM 的形式实现。 SFR 起始于数据存储器的顶部 （FFFh）并且向下扩展到Bank 15的全部空间（F00h到FFFh）和Bank 14的上半部分 （EF4h 到 EFFh）。

表 6-1 和表 6-2 列出了这些寄存器。

SFR 可分为两类：一类与“核心”器件功能（ALU、复位和中断）有关，另一类与外设功能有关。在相关的章节中将对复位和中断寄存器进行说明，而本章后面的部分将对 ALU 的 STATUS 寄存器进行说明。与外设功能部件的操作相关的寄存器在相应的外设章节中进行说明。

SFR 通常位于受其控制的外设中。未使用的 SFR 单元未实现，且读为 0。

表 6-1： PIC18F87K22 系列器件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 (4)
FFFh	TOSU	FDFh	INDF2(1)	FBFh	ECCP1AS	F9Fh	IPR1	F7Fh	EECON1	F5Fh	RTCCFG
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	ECCP1DEL	F9Eh	PIR1	F7Eh	EECON2	F5Eh	RTCCAL
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCPR1H	F9Dh	PIE1	F7Dh	TMR5H	F5Dh	RTCVALH
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR1L	F9Ch	PSTR1CON	F7Ch	TMR5L	F5Ch	RTCVALL
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCP1CON	F9Bh	OSCTUNE	F7Bh	T5CON	F5Bh	ALRMCFG
FFAh	PCLATH	FDAh	FSR2H	FBAh	PIR5	F9Ah	TRISJ(2)	F7Ah	T5GCON	F5Ah	ALRMRPT
FF9h	PCL	FD9h	FSR2L	FB9h	PIE5	F99h	TRISH(2)	F79h	CCPR4H	F59h	ALRMVALH
FF8h	TBLPTRU	FD8h	STATUS	FB8h	IPR4	F98h	TRISG	F78h	CCPR4L	F58h	ALRMVALL
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	PIR4	F97h	TRISF	F77h	CCP4CON	F57h	CTMUCONH
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	PIE4	F96h	TRISE	F76h	CCPR5H	F56h	CTMUCONL
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD	F75h	CCPR5L	F55h	CTMUICONH
FF4h	PRODH	FD4h	SPBRGH1	FB4h	CMSTAT	F94h	TRISC	F74h	CCP5CON	F54h	CM1CON
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	CCPR6H	F53h	PADCFG1
FF2h	INTCON	FD2h	IPR5	FB2h	TMR3L	F92h	TRISA	F72h	CCPR6L	F52h	ECCP2AS
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ(2)	F71h	CCP6CON	F51h	ECCP2DEL
FF0h	INTCON3	FD0h	RCON	FB0h	T3GCON	F90h	LATH(2)	F70h	CCPR7H	F50h	CCPR2H
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG	F6Fh	CCPR7L	F4Fh	CCPR2L
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF	F6Eh	CCP7CON	F4Eh	CCP2CON
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE	F6Dh	TMR4	F4Dh	ECCP3AS
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD	F6Ch	PR4	F4Ch	ECCP3DEL
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC	F6Bh	T4CON	F4Bh	CCPR3H
FEAh	FSR0H	FCAh	T2CON	FAAh	T1GCON	F8Ah	LATB	F6Ah	SSP2BUF	F4Ah	CCPR3L
FE9h	FSR0L	FC9h	SSP1BUF	FA9h	IPR6	F89h	LATA	F69h	SSP2ADD	F49h	CCP3CON
FE8h	WREG	FC8h	SSP1ADD	FA8h	HLVDCON	F88h	PORTJ(2)	F68h	SSP2STAT	F48h	CCPR8H
FE7h	INDF1(1)	FC7h	SSP1STAT	FA7h	PSPCON	F87h	PORTH(2)	F67h	SSP2CON1	F47h	CCPR8L
FE6h	POSTINC1(1)	FC6h	SSP1CON1	FA6h	PIR6	F86h	PORTG	F66h	SSP2CON2	F46h	CCP8CON
FE5h	POSTDEC1(1)	FC5h	SSP1CON2	FA5h	IPR3	F85h	PORTF	F65h	BAUDCON1	F45h	CCPR9H(3)
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE	F64h	OSCCON2	F44h	CCPR9L(3)
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD	F63h	EEADRH	F43h	CCP9CON(3)
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	EEADR	F42h	CCPR10H(3)
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	EEDATA	F41h	CCPR10L(3)
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	PIE6	F40h	CCP10CON(3)

注 1： 非物理寄存器。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
3： 此寄存器在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上不可用。
4： 地址 F16h 到 F5Fh 也由 SFR 使用，但是不属于快速操作 RAM。用户必须始终装入正确的 BSR 值来访问这些寄存器。

表 6-1： PIC18F87K22 系列器件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
F3Fh	TMR7H(3)	F32h	TMR12(3)	F25h	ANCON0	F18h	PMD1
F3Eh	TMR7L(3)	F31h	PR12(3)	F24h	ANCON1	F17h	PMD2
F3Dh	T7CON(3)	F30h	T12CON(3)	F23h	ANCON2	F16h	PMD3
F3Ch	T7GCON(3)	F2Fh	CM2CON	F22h	RCSTA2		
F3Bh	TMR6	F2Eh	CM3CON	F21h	TXSTA2		
F3Ah	PR6	F2Dh	CCPTMRS0	F20h	BAUDCON2		
F39H	T6CON	F2Ch	CCPTMRS1	F1Fh	SPBRGH2		
F38h	TMR8	F2Bh	CCPTMRS2	F1Eh	SPBRG2		
F37h	PR8	F2Ah	REFOCON	F1Dh	RCREG2		
F36h	T8CON	F29H	ODCON1	F1Ch	TXREG2		
F35h	TMR10(3)	F28h	ODCON2	F1Bh	PSTR2CON		
F34h	PR10(3)	F27h	ODCON3	F1Ah	PSTR3CON		
F33h	T10CON(3)	F26h	MEMCON(3)	F19h	PMD0		

注 1： 非物理寄存器。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
3： 此寄存器在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上不可用。
4： 地址 F16h 到 F5Fh 也由 SFR 使用，但是不属于快速操作 RAM。用户必须始终装入正确的 BSR 值来访问这些寄存器。

表 6-2： PIC18F87K22 系列器件寄存器汇总

地址	寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 与 BOR 时的值
FFFh	TOSU	—	—	—	栈顶最高字节 （TOS<20:16>）					---0 0000
FFEh	TOSH	栈顶高字节 （TOS<15:8>）								0000 0000
FFDh	TOSL	栈顶低字节 （TOS<7:0>）								0000 0000
FFCh	STKPTR	STKFUL	STKUNF	—	返回堆栈指针					uu-0 0000
FFBh	PCLATU	—	—	—	PC<20:16> 的保持寄存器					---0 0000
FFAh	PCLATH	PC<15:8> 的保持寄存器								0000 0000
FF9h	PCL	PC 低字节 （PC<7:0>）								0000 0000
FF8h	TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 （TBLPTR<20:16>）					--00 0000
FF7h	TBLPTRH	程序存储器表指针高字节 （TBLPTR<15:8>）								0000 0000
FF6h	TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								0000 0000
FF5h	TABLAT	程序存储器表锁存器								0000 0000
FF4h	PRODH	乘积寄存器高字节								xxxx xxxx
FF3h	PRODL	乘积寄存器低字节								xxxx xxxx
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x
FF1h	INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111
FF0h	INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000
FEFh	INDF0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值不变 （非物理寄存器）								---- ----
FEEh	POSTINC0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值后增 （非物理寄存器）								---- ----
FEDh	POSTDEC0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值后减 （非物理寄存器）								---- ----
FECh	PREINC0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值预增 （非物理寄存器）								---- ----
FEBh	PLUSW0	使用 FSR0 的内容来寻址数据存储器——FSR0 的值预增 （非物理寄存器）， FSR0 的偏移量由 W 寄存器提供								---- ----
FEAh	FSR0H	—	—	—	—	间接数据存储器地址指针 0 的高字节				---- 0000
FE9h	FSR0L	间接数据存储器地址指针 0 的低字节								xxxx xxxx
FE8h	WREG	工作寄存器								xxxx xxxx
FE7h	INDF1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值不变 （非物理寄存器）								---- ----
FE6h	POSTINC1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值后增 （非物理寄存器）								---- ----

注 1： 当禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 置 1 时，该位未实现。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
3： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

表 6-2： PIC18F87K22 系列器件寄存器汇总 （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 与 BOR 时的值
FE5h	POSTDEC1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值后减 （非物理寄存器）								---- ----
FE4h	PREINC1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值预增 （非物理寄存器）								---- ----
FE3h	PLUSW1	使用 FSR1 的内容来寻址数据存储器——FSR1 的值预增 （非物理寄存器）， FSR1 的偏移量由 W 寄存器提供								---- ----
FE2h	FSR1H	—	—	—	—	间接数据存储器地址指针 1 的高字节				---- xxxx
FE1h	FSR1L	间接数据存储器地址指针 1 的低字节								xxxx xxxx
FE0h	BSR	—	—	—	—	存储区选择寄存器				---- 0000
FDFh	INDF2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值不变 （非物理寄存器）								---- ----
FDEh	POSTINC2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值后增 （非物理寄存器）								---- ----
FDDh	POSTDEC2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值后减 （非物理寄存器）								---- ----
FDCh	PREINC2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值预增 （非物理寄存器）								---- ----
FDBh	PLUSW2	使用 FSR2 的内容来寻址数据存储器——FSR2 的值预增 （非物理寄存器）， FSR2 的偏移量由 W 寄存器提供								---- ----
FDAh	FSR2H	—	—	—	—	间接数据存储器地址指针 2 的高字节				---- xxxx
FD9h	FSR2L	间接数据存储器地址指针 2 的低字节								xxxx xxxx
FD8h	STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx
FD7h	TMR0H	Timer0 寄存器的高字节								0000 0000
FD6h	TMR0L	Timer0 寄存器的低字节								xxxx xxxx
FD5h	T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	TOPS2	TOPS1	TOPS0	1111 1111
FD4h	SPBRGH1	USART1 波特率发生器的高字节								0000 0000
FD3h	OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	HFIOFS	SCS1	SCS0	0110 q000
FD2h	IPR5	TMR7GIP(3)	TMR12IP(3)	TMR10IP(3)	TMR8IP	TMR7IP(3)	TMR6IP	TMR5IP	TMR4IP	1111 1111
FD1h	WDTCON	REGSLP	—	ULPLVL	SRETEN	—	ULPEN	ULPSINK	SWDTEN	0-x0 -000
FD0h	RCON	IPEN	SBOREN	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	0111 11qq
FCFh	TMR1H	Timer1 寄存器的高字节								xxxx xxxx
FCEh	TMR1L	Timer1 寄存器的低字节								xxxx xxxx
FCDh	T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON	0000 0000
FCCh	TMR2	Timer2 寄存器								0000 0000
FCBh	PR2	Timer2 周期寄存器								1111 1111
FCAh	T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000
FC9h	SSP1BUF	MSSP 接收缓冲器 / 发送寄存器								xxxx xxxx
FC8h	SSP1ADD	MSSP 地址寄存器 （I²C™ 从模式）。SSP1 波特率重载寄存器 （I²C 主模式）								0000 0000
FC7h	SSP1STAT	SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF	0000 0000
FC6h	SSP1CON1	WCOL	SSPOV	SSPEN	用软件	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000
FC5h	SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000
FC4h	ADRESH	A/D 结果寄存器的高字节								xxxx xxxx
FC3h	ADRESL	A/D 结果寄存器的低字节								xxxx xxxx
FC2h	ADCON0	—	CHS4	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON	-000 0000
FC1h	ADCON1	TRIGSEL1	TRIGSEL0	VCFG1	VCFG0	VNCFG	CHSN2	CHSN1	CHSN0	0000 0000
FC0h	ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000
FBFh	ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0	0000 0000
FBEh	ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0	0000 0000
FBDh	CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								xxxx xxxx
FBCh	CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								xxxx xxxx
FBBh	CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000
FBAh	PIR5	TMR7GIF(3)	TMR12IF(3)	TMR10IF(3)	TMR8IF	TMR7IF(3)	TMR6IF	TMR5IF	TMR4IF	0000 0000
FB9h	PIE5	TMR7GIE(3)	TMR12IE(3)	TMR10IE(3)	TMR8IE	TMR7IE(3)	TMR6IE	TMR5IE	TMR4IE	0000 0000
FB8h	IPR4	CCP10IP(3)	CCP9IP(3)	CCP8IP	CCP7IP	CCP6IP	CCP5IP	CCP4IP	CCP3IP	1111 1111
FB7h	PIR4	CCP10IF(3)	CCP9IF(3)	CCP8IF	CCP7IF	CCP6IF	CCP5IF	CCP4IF	CCP3IF	0000 0000
FB6h	PIE4	CCP10IE(3)	CCP9IE(3)	CCP8IE	CCP7IE	CCP6IE	CCP5IE	CCP4IE	CCP3IE	0000 0000

注 1： 当禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 置 1 时，该位未实现。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
3： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

表 6-2： PIC18F87K22 系列器件寄存器汇总 （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 与 BOR 时的值
FB5h	CVRCON	CVREN	CVROE	CVRSS	CVR4	CVR3	CVR2	CVR1	CVR0	0000 0000
FB4h	CMSTAT	CMP3OUT	CMP2OUT	CMP1OUT	—	—	—	—	—	xxx- ----
FB3h	TMR3H	Timer3 寄存器的高字节								xxxx xxxx
FB2h	TMR3L	Timer3 寄存器的低字节								xxxx xxxx
FB1h	T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON	0000 0000
FB0h	T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	$\overline{\text{T3GGO}}$/ $\overline{\text{T3DONE}}$	T3GVAL	T3GSS1	T3GSS0	0000 0x00
FAFh	SPBRG1	USART1 波特率发生器								0000 0000
FAEh	RCREG1	USART1 接收寄存器								0000 0000
FADh	TXREG1	USART1 发送寄存器								xxxx xxxx
FACh	TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010
FABh	RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
FAAh	T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	$\overline{\text{T1GGO}}$/ $\overline{\text{T1DONE}}$	T1GVAL	T1GSS1	T1GSS0	0000 0x00
FA9h	IPR6	—	—	—	EEIP	—	CMP3IP	CMP2IP	CMP1IP	---1 -111
FA8h	HLVDCON	VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	0000 0000
FA7h	PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----
FA6h	PIR6	—	—	—	EEIF	—	CMP3IF	CMP2IF	CMP1IF	---0 -000
FA5h	IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP	1-11 1111
FA4h	PIR3	TMR5GiF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF	0-00 0000
FA3h	PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE	0-00 0000
FA2h	IPR2	OSCFIP	—	SSP2IP	BCL2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP	1-11 1111
FA1h	PIR2	OSCFIF	—	SSP2IF	BCL2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF	0-00 0000
FA0h	PIE2	OSCFIE	—	SSP2IE	BCL2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE	0-00 0000
F9Fh	IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP	1111 1111
F9Eh	PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	TMR1GIF	TMR2IF	TMR1IF	0000 0000
F9Dh	PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE	0000 0000
F9Ch	PSTR1CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA	00-0 0001
F9Bh	OSCTUNE	INTSRC	PLLEN	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	0000 0000
F9Ah	TRISJ(2)	TRISJ7	TRISJ6	TRISJ5	TRISJ4	TRISJ3	TRISJ2	TRISJ1	TRISJ0	1111 1111
F99h	TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	1111 1111
F98h	TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	---1 1111
F97h	TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	1111 111-
F96h	TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	—	1111 111-
F95h	TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	1111 1111
F94h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111
F93h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111
F92h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111
F91h	LATJ(2)	LATJ7	LATJ6	LATJ5	LATJ4	LATJ3	LATJ2	LATJ1	LATJ0	xxxx xxxx
F90h	LATH(2)	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0	xxxx xxxx
F8Fh	LATG	—	—	—	LATG4	LATG3	LATG2	LATG1	LATG0	---x xxxx
F8Eh	LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—	xxxx xxx-
F8Dh	LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	xxxx xxxx
F8Ch	LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx xxxx
F8Bh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx xxxx
F8Ah	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx
F89h	LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	xxxx xxxx
F88h	PORTJ(2)	RJ7	RJ6	RJ5	RJ4	RJ3	RJ2	RJ1	RJ0	xxxx xxxx
F87h	PORTH(2)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	xxxx xxxx
F86h	PORTG	—	—	RG5(1)	RG4	RG3	RG2	RG1	RG0	--xx xxxx

注 1： 当禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 置 1 时，该位未实现。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
3： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

表 6-2： PIC18F87K22 系列器件寄存器汇总 （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 与 BOR 时的值
F85h	PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	xxxx xxx-
F84h	PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	xxxx xxxx
F83h	PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx
F82h	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx
F81h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx
F80h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxx xxxx
F7Fh	EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD	xx-0 x000
F7Eh	EECON2	EEPROM 控制寄存器 2 （非物理寄存器）								---- ----
F7Dh	TMR5H	Timer5 寄存器的高字节								xxxx xxxx
F7Ch	TMR5L	Timer5 寄存器的低字节								xxxx xxxx
F7Bh	T5CON	TMR5CS1	TMR5CS0	T5CKPS1	T5CKPS0	SOSCEN	$\overline{\text{T5SYNC}}$	RD16	TMR5ON	0000 0000
F7Ah	T5GCON	TMR5GE	T5GPOL	T5GTM	T5GSPM	T5GGO/ T5DONE	T5GVAL	T5GSS1	T5GSS0	0000 0x00
F79h	CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节								xxxx xxxx
F78h	CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节								xxxx xxxx
F77h	CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	--00 0000
F76h	CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节								xxxx xxxx
F75h	CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节								xxxx xxxx
F74h	CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	--00 0000
F73h	CCPR6H	捕捉 / 比较 /PWM 寄存器 6 的高字节								xxxx xxxx
F72h	CCPR6L	捕捉 / 比较 /PWM 寄存器 6 的低字节								xxxx xxxx
F71h	CCP6CON	—	—	DC6B1	DC6B0	CCP6M3	CCP6M2	CCP6M1	CCP6M0	--00 0000
F70h	CCPR7H	捕捉 / 比较 /PWM 寄存器 7 的高字节								xxxx xxxx
F6Fh	CCPR7L	捕捉 / 比较 /PWM 寄存器 7 的低字节								xxxx xxxx
F6Eh	CCP7CON	—	—	DC7B1	DC7B0	CCP7M3	CCP7M2	CCP7M1	CCP7M0	--00 0000
F6Dh	TMR4	Timer4 寄存器								xxxx xxxx
F6Ch	PR4	Timer4 周期寄存器								1111 1111
F6Bh	T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-111 1111
F6Ah	SSP2BUF	MSSP1 接收缓冲器 / 发送寄存器								xxxx xxxx
F69h	SSP2ADD	MSSP 地址寄存器 （I^2C™ 从模式）。MSSP1 波特率重载寄存器 （I^2C 主模式）								0000 0000
F68h	SSP2STAT	SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF	0000 0000
F67h	SSP2CON1	WCOL	SSPOV	SSPEN	用软件	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000
F66h	SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0100 0000
F65h	BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	—	WUE	ABDEN	0100 0-00
F64h	OSCCON2	—	SOSCRUN	—	—	SOSCGO	—	MFIOFS	MFIOSEL	-0-- 0-x0
F63h	EEADRH	EEPROM 地址寄存器的高字节								0000 0000
F62h	EEADR	EEPROM 地址寄存器的低字节								0000 0000
F61h	EEDATA	EEPROM 数据寄存器								0000 0000
F60h	PIE6	—	—	—	EEIE	—	CMP3IE	CMP2IE	CMP1IE	---0 -000
F5Fh	RTCCFG	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0	0-00 0000
F5Eh	RTCCAL	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	0000 0000
F5Dh	RTCVALH	基于 RTCPTR<1:0> 的 RTCC 值高位寄存器窗口								xxxx xxxx

注 1： 当禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 置 1 时，该位未实现。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
3： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

表 6-2： PIC18F87K22 系列器件寄存器汇总 （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 与 BOR 时的值
F5Ch	RTCVALL	基于 RTCPTR<1:0> 的 RTCC 值低位寄存器窗口								0000 0000
F5Bh	ALRMCFG	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0	0000 0000
F5Ah	ALRMRPT	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0	0000 0000
F59h	ALRMVALH	基于 APTR<1:0> 的闹钟值高位寄存器窗口								xxxx xxxx
F58h	ALRMVALL	基于 APTR<1:0> 的闹钟值低位寄存器窗口								xxxx xxxx
F57h	CTMUCONH	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG	0-00 0000
F56h	CTMUCONL	EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT	0000 0000
F55h	CTMUICONH	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	0000 0000
F54h	CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	0001 1111
F53h	PADCFG1	RDPU	REPU	RJPU(2)	—	—	RTSECSEL1	RTSECSEL0	—	000- -00-
F52h	ECCP2AS	ECCP2ASE	ECCP2AS2	ECCP2AS1	ECCP2AS0	PSS2AC1	PSS2AC0	PSS2BD1	PSS2BD0	0000 0000
F51h	ECCP2DEL	P2RSEN	P2DC6	P2DC5	P2DC4	P2DC3	P2DC2	P2DC1	P2DC0	0000 0000
F50h	CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								xxxx xxxx
F4Fh	CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								xxxx xxxx
F4Eh	CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	0000 0000
F4Dh	ECCP3AS	ECCP3ASE	ECCP3AS2	ECCP3AS1	ECCP3AS0	PSS3AC1	PSS3AC0	PSS3BD1	PSS3BD0	0000 0000
F4Ch	ECCP3DEL	P3RSEN	P3DC6	P3DC5	P3DC4	P3DC3	P3DC2	P3DC1	P3DC0	0000 0000
F4Bh	CCPR3H	捕捉 / 比较 /PWM 寄存器 3 的高字节								xxxx xxxx
F4Ah	CCPR3L	捕捉 / 比较 /PWM 寄存器 3 的低字节								xxxx xxxx
F49h	CCP3CON	P3M1	P3M0	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0	0000 0000
F48h	CCPR8H	捕捉 / 比较 /PWM 寄存器 8 的高字节								xxxx xxxx
F47h	CCPR8L	捕捉 / 比较 /PWM 寄存器 8 的低字节								xxxx xxxx
F46h	CCP8CON	—	—	DC8B1	DC8B0	CCP8M3	CCP8M2	CCP8M1	CCP8M0	--00 0000
F45h	CCPR9H(3)	捕捉 / 比较 /PWM 寄存器 9 的高字节								xxxx xxxx
F44h	CCPR9L(3)	捕捉 / 比较 /PWM 寄存器 9 的低字节								xxxx xxxx
F43h	CCP9CON(3)	—	—	DC9B1	DC9B0	CCP9M3	CCP9M2	CCP9M1	CCP9M0	--00 0000
F42h	CCPR10H(3)	捕捉 / 比较 /PWM 寄存器 10 的高字节								xxxx xxxx
F41h	CCPR10L(3)	捕捉 / 比较 /PWM 寄存器 10 的低字节								xxxx xxxx
F40h	CCP10CON(3)	—	—	DC10B1	DC10B0	CCP10M3	CCP10M2	CCP10M1	CCP10M0	--00 0000
F3Fh	TMR7H(3)	Timer7 寄存器的高字节								xxxx xxxx
F3Eh	TMR7L(3)	Timer7 寄存器的低字节								0000 0000
F3Dh	T7CON(3)	TMR7CS1	TMR7CS0	T7CKPS1	T7CKPS0	SOSCEN	$\overline{\text{T7SYNC}}$	RD16	TMR7ON	0000 0000
F3Ch	T7GCON(3)	TMR7GE	T7GPOL	T7GTM	T7GSPM	T7GGO/ $\overline{\text{T7DONE}}$	T7GVAL	T7GSS1	T7GSS0	0000 0x00
F3Bh	TMR6	Timer6 寄存器								0000 0000
F3Ah	PR6	Timer6 周期寄存器								1111 1111
F39h	T6CON	—	T6OUTPS3	T6OUTPS2	T6OUTPS1	T6OUTPS0	TMR6ON	T6CKPS1	T6CKPS0	-000 0000
F38h	TMR8	Timer8 寄存器								0000 0000
F37h	PR8	Timer8 周期寄存器								1111 1111
F36h	T8CON	—	T8OUTPS3	T8OUTPS2	T8OUTPS1	T8OUTPS0	TMR8ON	T8CKPS1	T8CKPS0	-000 0000
F35h	TMR10(3)	Timer10 寄存器								0000 0000
F34h	PR10(3)	Timer10 周期寄存器								1111 1111
F33h	T10CON(3)	—	T10OUTPS3	T10OUTPS2	T10OUTPS1	T10OUTPS0	TMR10ON	T10CKPS1	T10CKPS0	-000 0000
F32h	TMR12(3)	Timer12 寄存器								0000 0000
F31h	PR12(3)	Timer12 周期寄存器								1111 1111
F30h	T12CON(3)	—	T12OUTPS3	T12OUTPS2	T12OUTPS1	T12OUTPS0	TMR12ON	T12CKPS1	T12CKPS0	-000 0000
F2Fh	CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	0001 1111
F2Eh	CM3CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0	0001 1111
F2Dh	CCPTMRS0	C3TSEL1	C3TSEL0	C2TSEL2	C2TSEL1	C2TSEL0	C1TSEL2	C1TSEL1	C1TSEL0	0000 0000

注 1： 当禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 置 1 时，该位未实现。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
3： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

表 6-2： PIC18F87K22 系列器件寄存器汇总 （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 与 BOR 时的值
F2Ch	CCPTMRS1	C7TSEL1	C7TSEL0	—	C6TSEL0	—	C5TSEL0	C4TSEL1	C4TSEL0	00-0 -000
F2Bh	CCPTMRS2	—	—	—	C10TSEL0(3)	—	C9TSEL0(3)	C8TSEL1	C8TSEL0	---0 -000
F2Ah	REFOCON	ROON	—	ROSSLP	ROSEL	RODIV3	RODIV2	RODIV1	RODIV0	0-00 0000
F29h	ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD	000- ---0
F28h	ODCON2	CCP10OD(3)	CCP9OD(3)	CCP8OD	CCP7OD	CCP6OD	CCP5OD	CCP4OD	CCP3OD	0000 0000
F27h	ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS	00-- ---0
F26h	MEMCON(2)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00
F25h	ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0	1111 1111
F24h	ANCON1	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8	1111 1111
F23h	ANCON2	ANSEL23	ANSEL22	ANSEL21	ANSEL20	ANSEL19	ANSEL18	ANSEL17	ANSEL16	1111 1111
F22h	RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x
F21h	TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010
F20h	BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	-	WUE	ABDEN	0100 0-00
F1Fh	SPBRGH2	USART2 波特率发生器的高字节								0000 0000
F1Eh	SPBRG2	USART2 波特率发生器								0000 0000
F1Dh	RCREG2	接收数据 FIFO								0000 0000
F1Ch	TXREG2	发送数据 FIFO								xxxx xxxx
F1Bh	PSTR2CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA	00-0 0001
F1Ah	PSTR3CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA	00-0 0001
F19h	PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD	0000 0000
F18h	PMD1	PSPMD	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBMD	0000 0000
F17h	PMD2	TMR10MD(3)	TMR8MD	TMR7MD(3)	TMR6MD	TMR5MD	CMP3MD	CMP2MD	CMP1MD	0000 0000
F16h	PMD3	CCP10MD(3)	CCP9MD(3)	CCP8MD	CCP7MD	CCP6MD	CCP5MD	CCP4MD	TMR12MD(3)	0000 0000

注 1： 当禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 置 1 时，该位未实现。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
3： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

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。和其他寄存器一样，STATUS 寄存器可以是任何指令的操作数。如果一条影响 Z、DC、C、OV 或 N 位的指令的目标寄存器是 STATUS 寄存器，则禁止写这 5 位。

根据器件逻辑，这些位会被置 1 或清零。所以当执行一条把 STATUS 寄存器作为目标寄存器的指令后，STATUS 寄存器的结果可能和预想的不一样。例如，`CLRF STATUS`会将Z位置1，而保留其余位不变。此时，读取 STATUS 寄存器的值得到的是 `000u u1uu`。

因此，建议只用 `BCF`、`BSF`、`SWAPF`、`MOVFF` 和 `MOVWF` 指令来改变 STATUS 寄存器的值，因为这些指令不会影响该寄存器中的 Z、 C、 DC、 OV 或 N 位。

关于其他不影响任何状态位的指令，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$\overline{\text{借位}}$和半借位标志位。

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 = 上电复位时的值	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

此位用于有符号的算术运算 （通过 2 进制补码方式进行）。它指示结果是否为负 （ALU MSB = `1`）。

`1` = 结果为负
`0` = 结果为正

bit 3 **OV：**溢出位

此位用于有符号的算术运算 （通过 2 进制补码方式进行）。它指示运算结果超出了 7 位二进制数据的范围，溢出将导致符号位 （bit 7）发生改变。

`1` = 有符号的算术运算中发生溢出 （本次算术运算）
`0` = 没有发生溢出

bit 2 **Z：**全零标志位

`1` = 算术运算或逻辑运算结果为零
`0` = 算术运算或逻辑运算结果不为零

bit 1 **DC：**半借位 / $\overline{\text{借位}}$标志位 (1)

对于 `ADDWF`、 `ADDLW`、 `SUBLW` 和 `SUBWF` 指令：

`1` = 结果的第 4 低位发生了进位
`0` = 结果的第 4 低位未发生进位

bit 0 **C：**进位 / $\overline{\text{借位}}$标志位 (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$\overline{\text{借位}}$，极性是相反的。减法是通过加上第二个操作数的 2 进制补码来实现的。
2： 对于$\overline{\text{借位}}$，极性是相反的。减法是通过加上第二个操作数的 2 进制补码来实现的。

© 2011 Microchip Technology Inc.

6.4 数据寻址模式

> **注：** 当使能 PIC18 扩展指令集时， PIC18 核心指令集中的某些指令的执行会发生改变。更多信息，请参见第 6.6 节 “数据存储器和扩展指令集”。

虽然只能用一种方法（通过程序计数器）对程序存储器进行寻址，但是可以用多种方法来对数据存储空间进行寻址。对于大多数指令，寻址模式是固定的。其余指令最多可以使用三种模式，这取决于使用哪些操作数及是否使能了扩展指令集。

寻址模式有：

- 固有寻址
- 立即数寻址
- 直接寻址
- 间接寻址

当使能扩展指令集（XINST 配置位 = 1）时，可以使用另一种寻址模式，即立即数变址寻址模式。有关此模式操作的详情，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 固有和立即数寻址

很多 PIC18 控制类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，要么仅针对一个寄存器进行操作。该寻址模式称为固有寻址。此模式的示例包括 SLEEP、 RESET 和 DAW。

其他指令的工作方式与此类似，但需要在操作码中明确指定参数。由于指令需要一些立即数作为参数，这种寻址模式被称为立即数寻址模式。此模式的示例包括 ADDLW 和 MOVLW，它们分别将立即数值加到或移入 W 寄存器。其他的立即数寻址指令，例如 CALL 和 GOTO，包含一个 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 PIC18 核心指令集中，位操作类指令和字节操作类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它们的最低有效字节。此地址指定数据 RAM 的某个存储区中的寄存器地址（见第 6.3.3 节 “通用寄存器” ）或快速操作存储区中的地址单元（见第6.3.2节“快速操作存储区”）作为指令的数据源。

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”）的内容和地址一起用于确定寄存器完整的 12 位地址。当 a 为 0 时，此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 寻址有时候也被称为直接强制寻址模式。

有几条指令，比如 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 完全被忽略。

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它原来的内容。当 d 为 0 时，结果被存储到 W 寄存器中。不带 d 参数的指令的目标地址隐含在指令中，它们是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不需要在指令中给出一个固定的地址。这是通过使用文件选择寄存器（File Select Register， FSR）作为指针指向被读取或写入的单元实现的。由于 FSR 本身作为特殊功能寄存器位于 RAM 中，所以也可以在程序控制下直接对它们进行操作。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非常有用。

也可以使用间接指针操作数 （Indirect File Operand，INDF）对寄存器进行间接寻址。此操作允许自动递增、递减或偏移指针，从而自动控制指针的值。它通过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。它也允许用户在数据存储器中进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： 使用间接寻址清零 RAM （BANK 1）的方法

```
        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

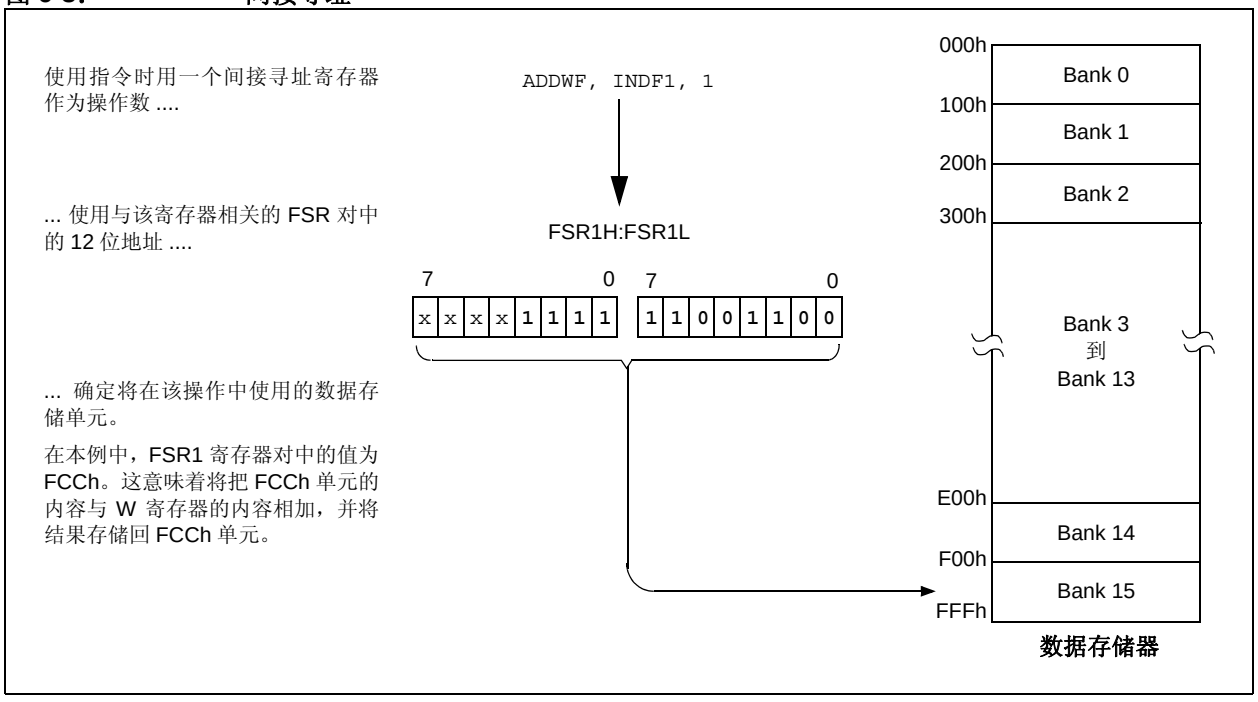
间接寻址的核心是三组寄存器：FSR0、FSR1和FSR2。每组寄存器都代表一对8位寄存器：FSRnH和FSRnL。FSRnH 寄存器的高 4 位未使用，所以每对 FSR 只保存一个 12 位值。这表示一个可以线性寻址整个数据存储空间的值。因此，FSR 寄存器对将作为指向数据存储单元的指针。

间接寻址可用一组间接指针操作数（从 INDF0 到 INDF2）完成。这些操作数可以被看作是“虚拟”寄存器。它们被映射到 SFR 空间中而不是通过物理方式实现。对特定的 INDF 寄存器执行读或写操作实际上访问的是相应的 FSR 寄存器对。例如，读 INDF1 就是读 FSR1H:FSR1L 指向的地址单元中的数据。

使用 INDF 寄存器作为操作数的指令实际上是使用相应 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，因此没有必要对数据 RAM 进行分区。故 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



© 2011 Microchip Technology Inc.

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，这些也是不能直接读写的“虚拟”寄存器。访问这些寄存器其实就是访问相关的 FSR 寄存器对，并对其存储的数据进行特定的操作。

这些操作数为：

- POSTDEC——访问 FSR 的值，然后将它自动减 1
- POSTINC——访问 FSR 的值，然后将它自动加 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 到 128）与 FSR 寄存器中的值相加，并在操作中使用得到的新值。

如前所述，访问 INDF 寄存器使用的是 FSR 寄存器中的值，但不更改此值。同样，访问 PLUSW 寄存器是将 W 寄存器中的值作为FSR值的偏移量，该操作不会改变这两个寄存器中的值。访问其他虚拟寄存器会更改FSR寄存器的值。

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 到 00h 计满返回，会向 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 等）。

PLUSW 寄存器可以用于在数据存储空间实现变址寻址。通过控制 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，该功能可以被用于在数据存储器内部实现某些非常有用的程序控制结构，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或虚拟寄存器作为寻址目标。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 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。

另一方面，使用虚拟寄存器对FSR寄存器对进行写操作可能会产生与预期不同的结果。在这些情形下，值将被写入 FSR 寄存器对，但 FSR 中的值不会有任何增减。因此，写入 INDF2 或 POSTDEC2 时会把相同的值写入 FSR2H:FSR2L。

由于 FSR 是映射到 SFR 空间的物理寄存器，所以可以通过直接寻址对它们进行操作。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

使用扩展指令集不会影响程序存储器的操作。

使能扩展指令集会在现有的PIC18指令集中添加5条额外的双字指令：ADDFSR、CALLW、MOVSF、MOVSS 和 SUBFSR。**第 6.2.4 节“双字指令”**说明了这些指令的执行方式。

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）明显更改了数据存储器及其寻址方式的某些方面。对许多 PIC18 核心指令使用快速操作存储区会引入一种新的数据存储空间寻址模式。此模式还会改变使用 FSR2 及其相关的操作数进行间接寻址的方式。

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

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 寄存器对及其相关指针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（即大多数位和字节操作类指令）可以利用指令中指定的偏移量来执行变址寻址。这种特定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。

在使用扩展指令集时，这种寻址模式有如下要求：

- 使用快速操作存储区 （a = 0）
- 指针地址参数要小于或等于 5Fh

在这些条件下，指令的指针地址不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。相反，该值被解析为由 FSR2 指定的地址指针的偏移量。该偏移量与 FSR2 的内容相加以获得操作的目标地址。

6.6.2 受立即数变址寻址模式影响的指令

任何可以使用直接寻址的PIC18核心指令均会受到立即数变址寻址模式的潜在影响。包括所有字节和位操作类指令，或标准 PIC18 指令集中几乎一半的指令。只有使用固有或立即数寻址模式的指令不受影响。

此外，如果字节和位操作类指令不使用快速操作存储区（快速操作 RAM 位为 1）或包含 60h 或以上的指针地址，它们也不会受影响。符合这些标准的指令会像以前一样继续执行。图 6-9 给出了当使能扩展指令集时，各种可能寻址模式的对比。

那些想要在立即数变址寻址模式中使用字节或位操作类指令的用户，应该注意此模式下汇编语法的改变。**第 29.2.1 节“扩展指令的语法”**中将对此进行更详细的说明。

© 2011 Microchip Technology Inc.

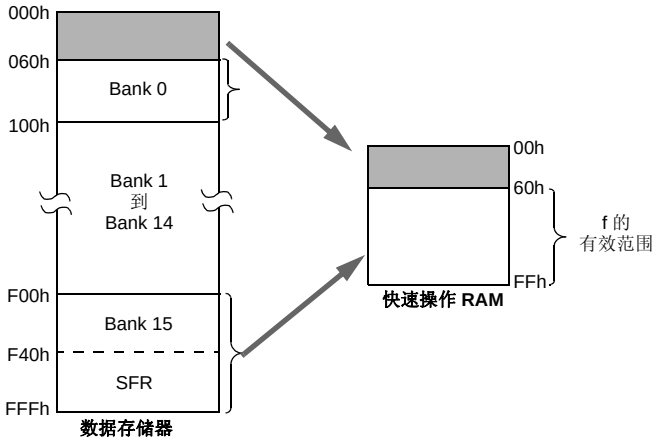
图 6-9： 位和字节操作类指令的寻址方式对比 （使能了扩展指令集）

指令示例：`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 和 FFFh 之间的单元。这实际上是数据存储器中从F60h 到 FFFh（Bank 15）之间的存储单元。

此寻址模式中不可以使用地址低于 60h 的单元。



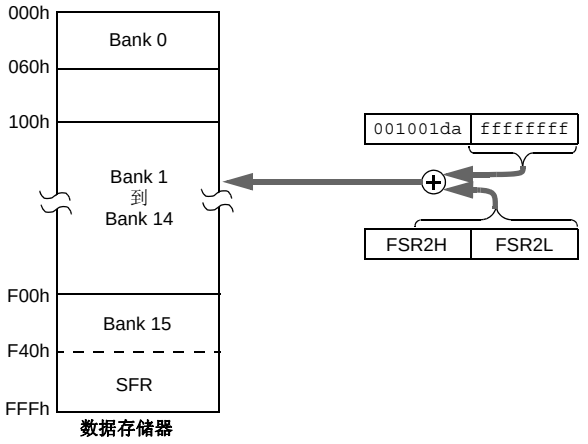
当 a = 0 且 f ≤ 5Fh 时：

该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。这两个值相加以获得指令的目标寄存器的地址。此地址可以位于数据存储空间的任何位置。

注意在此模式中，正确的语法是：

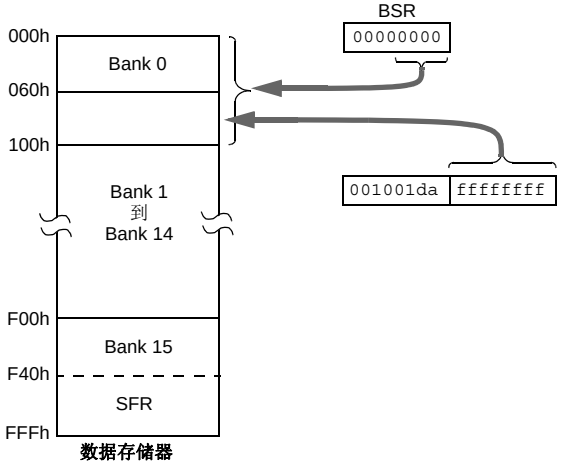
`ADDWF [k], d`

其中 k 就是 f。



当 a = 1（f 为所有值）时：

指令以直接模式 （也被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个存储区中的某个单元的地址。此存储区由存储区选择寄存器（BSR）指定。此地址可以在数据存储空间的任何已实现区域。



6.6.3 在立即数变址寻址模式下映射快速操作存储区

使用立即数变址寻址模式能有效地改变快速操作 RAM 的低地址部分（00h 到 5Fh）的映射方式。除了包含 Bank 0 底部的内容之外，此模式还映射 Bank 0 的内容和由用户定义的可以位于数据存储空间中任何位置的“窗口”内容。

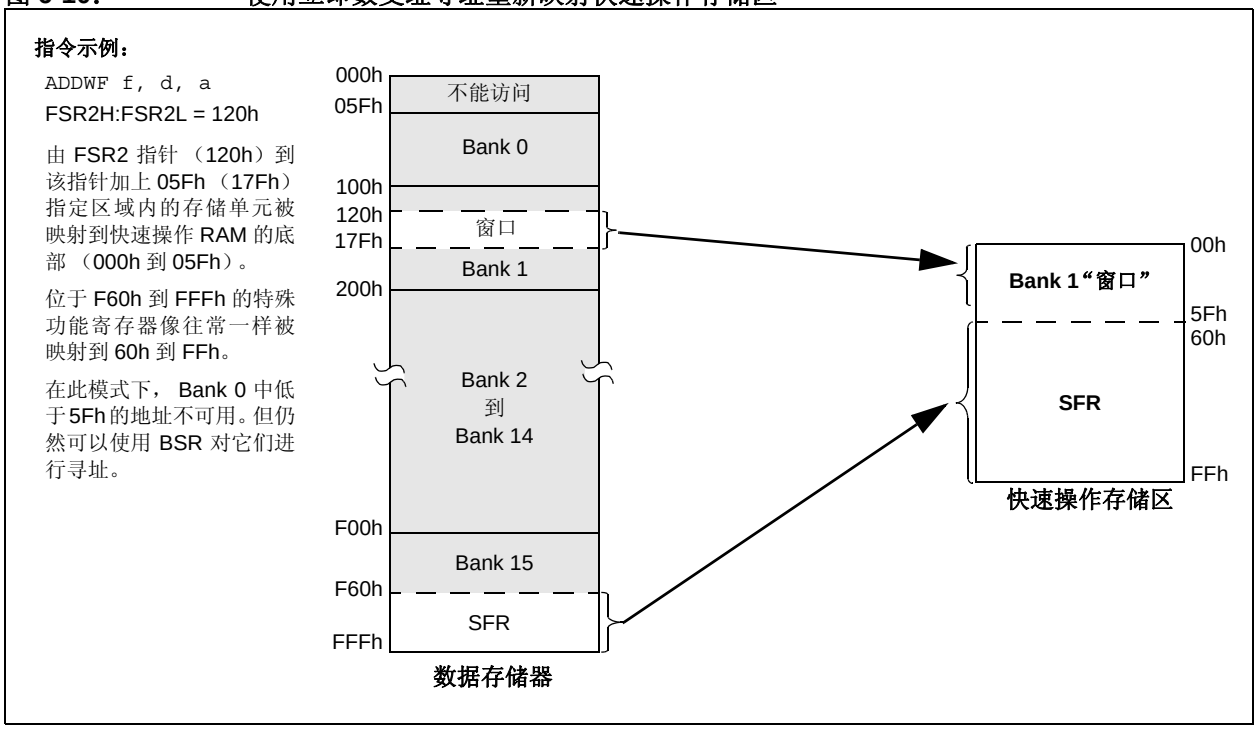
FSR2 的值定义映射到窗口的地址的下边界，而上边界则由 FSR2 加 95（5Fh）确定。地址为 5Fh 以上的快速操作 RAM 的映射方法如前所述。（见**第 6.3.2 节“快速操作存储区”**。）图 6-10 所示为在此寻址模式中重新映射的快速操作存储区示例。

快速操作存储区的重新映射*仅适用于*使用立即数变址寻址模式的操作。使用 BSR（快速操作 RAM 位为 1）的操作和前面一样继续使用直接寻址模式。任何明确使用间接指针操作数（包括 FSR2）进行的间接或变址寻址操作都将继续以标准间接寻址模式进行操作。任何使用快速操作存储区，但是包括大于 05Fh 的寄存器地址的指令都将使用直接寻址和常规快速操作存储区映射。

6.6.4 立即数变址寻址模式中的 BSR

虽然使能了扩展指令集时会重新映射快速操作存储区，但 BSR 的操作将保持不变。使用 BSR 来选择数据存储区的直接寻址模式的操作方式与前面描述的相同。

图 6-10： 使用立即数变址寻址重新映射快速操作存储区



© 2011 Microchip Technology Inc.

7.0 闪存程序存储器

在整个 VDD 范围内，闪存程序存储器在正常工作状态下都是可读写并可擦除的。

对程序存储器的读操作按照每次 1 个字节来执行。对程序存储器执行写或擦除操作：

- 32 KB 和 64 KB 的程序存储器 （PIC18FX5K22 和 PIC18FX6K22 器件）——每次 64 字节块
- 128 KB 的程序存储器 （PIC18FX7K22 器件）——每次 128 字节块

写或擦除程序存储器将中止取指操作，直到写或擦除操作完成为止。在写或擦除期间不能访问程序存储器，因此无法执行代码。内部编程定时器可终止对程序存储器的写入和擦除操作。

写入程序存储器的值不必是有效指令。执行存有无效指令的程序存储单元会导致执行 NOP 指令。

7.1 表读和表写

为了读写程序存储器，有两种操作可以让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）

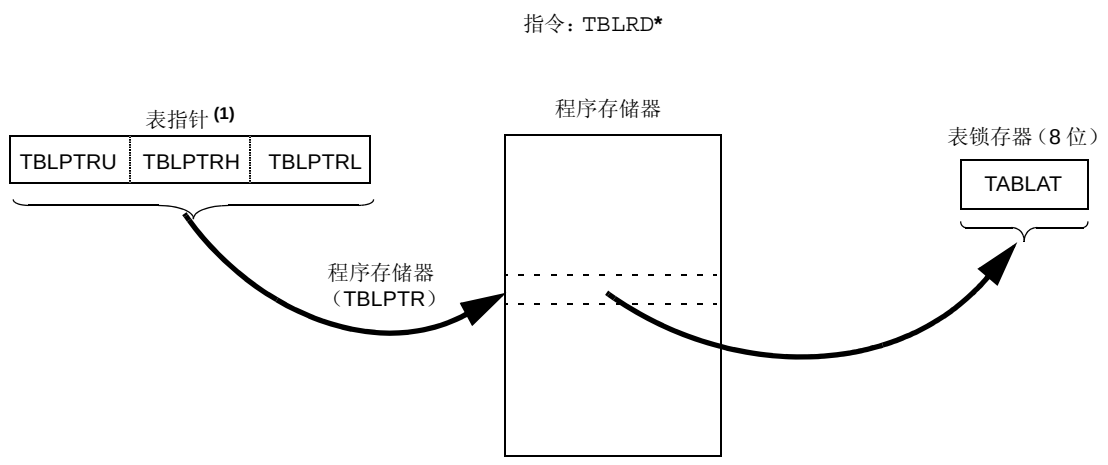
程序存储空间为 16 位宽，而数据 RAM 空间为 8 位宽。表读和表写操作通过一个 8 位寄存器 （TABLAT）在这两个存储空间之间传送数据。

表读操作从程序存储器获取数据并将其存入数据 RAM 空间。图 7-1 显示了程序存储器和数据 RAM 之间的表读操作。

表写操作将数据存储空间中的数据存入程序存储器中的保持寄存器。**第 7.5 节 “写入闪存程序存储器”** 将详细介绍将保持寄存器中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 之间的表写操作。

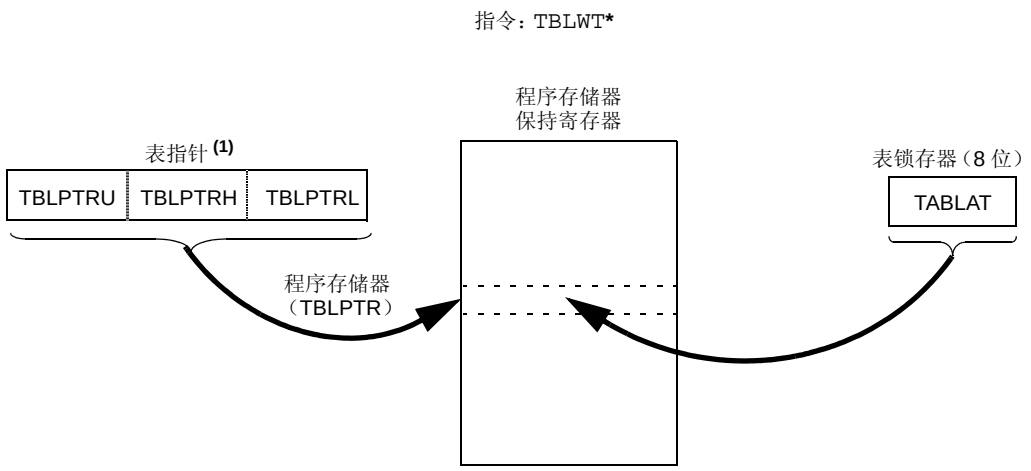
表操作是以字节为单位进行的。一个包含数据而非程序指令的表块不必按字对齐。因此，表块可在任何字节地址处开始和结束。如果使用表写操作将可执行代码写入程序存储器，程序指令就需要按字对齐。

图 7-1： 表读操作



注 1：表指针寄存器指向程序存储器的某一字节。

图 7-2： 表写操作



注 1：表指针实际指向的是 64 个保持寄存器中的某一个，其地址由 TBLPTRL<5:0> 决定。第 7.5 节“写入闪存程序存储器”中将讨论将数据实际写入程序存储器阵列的过程。

7.2 控制寄存器

有几个控制寄存器与 TBLRD 和 TBLWT 指令一起使用。其中包括：

- 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）是访问存储器的控制寄存器。EECON2 寄存器不是物理寄存器，它专用于存储器的擦写操作。读 EECON2 得到的是全 0。

控制位 EEPGD 决定将访问程序存储器还是数据 EEPROM 存储器。若清零该位，任何后续操作均将针对数据 EEPROM 存储器进行。若该位置 1，任何后续操作均将针对程序存储器进行。

控制位 CFGS 决定将访问配置 / 校准寄存器还是程序存储器 / 数据 EEPROM 存储器。置 1 时，无论 EEPGD 状态如何，后续操作均将针对配置寄存器进行（见第 28.0 节“CPU 的特殊功能”）。清零时，存储器访问选择由 EEPGD 决定。

FREE 位置 1 时允许对程序存储器执行擦除操作。FREE 位置 1 时，擦除操作由下一条 WR 命令触发。当 FREE 位清零时，则只使能写操作。

WREN 位置 1 时允许写操作。上电时将清零 WREN 位。当 WR 位置 1 时，WRERR 位将由硬件置 1；当内部编程定时器超时并且写操作完成时，清零 WRERR 位。

注： 如果在正常工作期间，WRERR 读为 1，则表明写操作因复位而提早终止或进行了非法的写操作。

控制位 WR 用于启动写操作。软件只能将此位置 1 而不能清零。该位在写操作完成时由硬件清零。

注： 当写操作完成时 EEIF 中断标志位（PIR6<4>）置 1。此标志位必须由软件清零。

© 2011 Microchip Technology Inc.

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 = 上电复位时的值	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 寻址的程序存储器行（擦除操作完成时清零）
0 = 仅执行写操作

bit 3 **WRERR：**闪存程序存储器 / 数据 EEPROM 错误标志位 [1]
1 = 写操作提早终止 （由正常工作中自定时编程期间的任何复位或不正确写入引起）
0 = 写操作已经完成

bit 2 **WREN：**闪存程序存储器 / 数据 EEPROM 写使能位
1 = 允许闪存程序存储器 / 数据 EEPROM 的写周期
0 = 禁止闪存程序存储器 / 数据 EEPROM 的写周期

bit 1 **WR：**写控制位
1 = 启动数据 EEPROM 擦 / 写周期或程序存储器擦写周期。（该操作是自定时的，一旦写入完成该位即由硬件清零。软件只能将 WR 位置 1 而不能清零。）
0 = 写入 EEPROM 的周期结束

bit 0 **RD：**读控制位
1 =启动EEPROM读操作（读操作需要一个周期。RD位由硬件清零。软件只能将RD位置1而不能清零。当 EEPGD = 1 或 CFGS = 1 时， RD 位不能置 1。）
0 = 未启动 EEPROM 读操作

注 1： 当发生 WRERR 时， EEPGD 和 CFGS 位不会被清零。这允许跟踪错误条件。

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。表锁存寄存器用于保存在程序存储器和数据RAM之间传输的 8 位数据。

7.2.3 TBLPTR——表指针寄存器

表指针（TBLPTR）寄存器指向程序存储器中的一个字节。TBLPTR 由 3 个 SFR 寄存器组成，即：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。器件使用该寄存器的低 21 位寻址最多 2 MB 的程序存储空间。第 22 位可用于访问器件 ID、用户 ID 以及配置位。

TBLRD 和 TBLWT 指令要用到表指针寄存器 TBLPTR。根据不同的表操作，上述指令可使用四种方式更新 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，由 TBLPTR 的全部 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 64 个程序存储器保持寄存器的哪一个。当对程序存储器的定时写操作开始后（通过 WR 位启动），由 TBLPTR 的高 16 位（TBLPTR<21:6>）决定要写入哪一个 64 字节的程序存储块。如需更多详细信息，请参见**第 7.5 节 “写入闪存程序存储器”**。

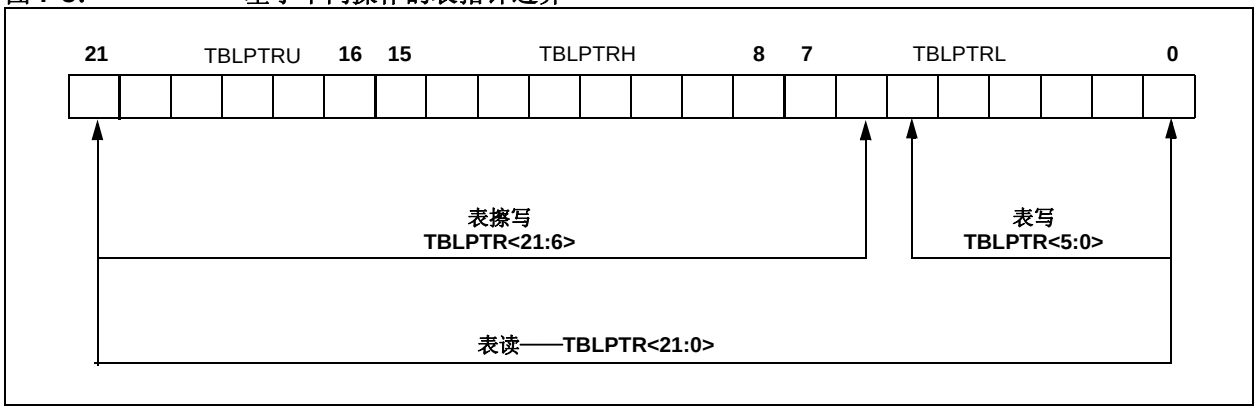
当擦除程序存储器时，表指针寄存器的高 16 位（TBLPTR<21:6>）指向将要擦除的 64 字节块，而忽略低 6 位（TBLPTR<5:0>）。

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：使用 TBLRD 和 TBLWT 指令对表指针进行操作

示例	表指针操作
TBLRD* TBLWT*	不修改 TBLPTR
TBLRD*+ TBLWT*+	读 / 写后， TBLPTR 递增
TBLRD*- TBLWT*-	读 / 写后， TBLPTR 递减
TBLRD+* TBLWT+*	读 / 写前， TBLPTR 递增

图 7-3：基于不同操作的表指针边界



© 2011 Microchip Technology Inc.

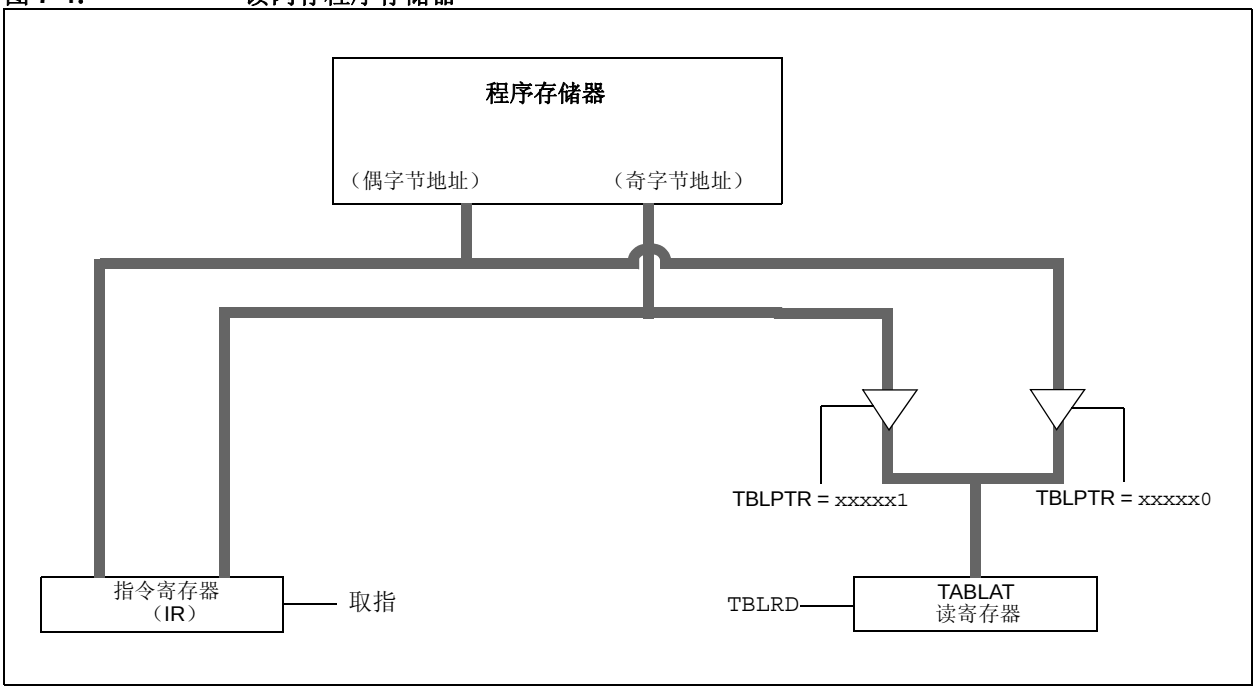
7.3 读闪存程序存储器

TBLRD指令用于从程序存储器获取数据并把数据存入数据 RAM。对程序存储器执行表读操作时，每次读取一个字节。

TBLPTR 指向程序空间内的某个字节地址。执行 TBLRD 会将指向的字节存入 TABLAT。另外，可以自动修改 TBLPTR 以进行下次表读操作。

内部程序存储器通常是以字为单位构成的。由地址的低位选择字的高字节和低字节。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 7-3： 读一个闪存程序存储器字

```
            BCF     EECON1, CFGS            ; point to Flash program memory
            BSF     EECON1, EEPGD           ; access Flash program memory
            MOVLW   CODE_ADDR_UPPER         ; Load TBLPTR with the base
            MOVWF   TBLPTRU                 ; address of the word
            MOVLW   CODE_ADDR_HIGH
            MOVWF   TBLPTRH
            MOVLW   CODE_ADDR_LOW
            MOVWF   TBLPTRL
READ_WORD
            TBLRD*+                         ; read into TABLAT and increment
            MOVF    TABLAT, W               ; get data
            MOVWF   WORD_EVEN
            TBLRD*+                         ; read into TABLAT and increment
            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：

- PIC18FX5K22 和 PIC18FX6K22——32 字或 64 字节
- PIC18FX7K22——64 字或 128 字节

闪存阵列不支持字擦除。

当单片机启动擦除操作时，将擦除程序存储器的一个 64 字节或 128 字节块。TBLPTR<21:6> 的高 16 位指向将要擦除的数据块。 TBLPTR<5:0> 位被忽略。

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。必须将 WREN 位置 1 以使能写操作。必须将 FREE 位置 1 以选择擦除操作。

为了安全起见，必须采用 EECON2 的写操作启动序列。

擦除内部闪存需要使用长写周期。在长写周期中，指令执行暂停。由内部编程定时器终止长周期写操作。

7.4.1 闪存程序存储器擦除序列

擦除内部程序存储器数据块的步骤如下：

1. 将要擦除的行地址装入表指针寄存器。
2. 设置 EECON1 寄存器以执行擦除操作：
   - 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. 向 EECON2 写入 0x55。
5. 向 EECON2 写入 0xAA。
6. 将 WR 位置 1。
   行擦除周期开始。
   在擦除操作期间，CPU将在Tiw（见参数D133A）时间段内停止工作。
7. 重新允许中断。

例 7-3： 擦除闪存程序存储器行

```
                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
必需的          MOVLW   0x55
序列            MOVWF   EECON2                  ; write 55h
                MOVLW   0xAA
                MOVWF   EECON2                  ; write 0AAh
                BSF     EECON1, WR              ; start erase (CPU stall)
                BSF     INTCON, GIE             ; re-enable interrupts
```


© 2011 Microchip Technology Inc.

7.5 写入闪存程序存储器

编程块大小为：

- PIC18FX5K22 和 PIC18FX6K22——32 字或 64 字节
- PIC18FX7K22——64 字或 128 字节

不支持字或字节编程。

表写操作用于在内部将数据装入编程闪存存储器所需的保持寄存器。表写操作用来进行编程的保持寄存器的个数为：

- PIC18FX5K22 和 PIC18FX6K22——64
- PIC18FX7K22——128

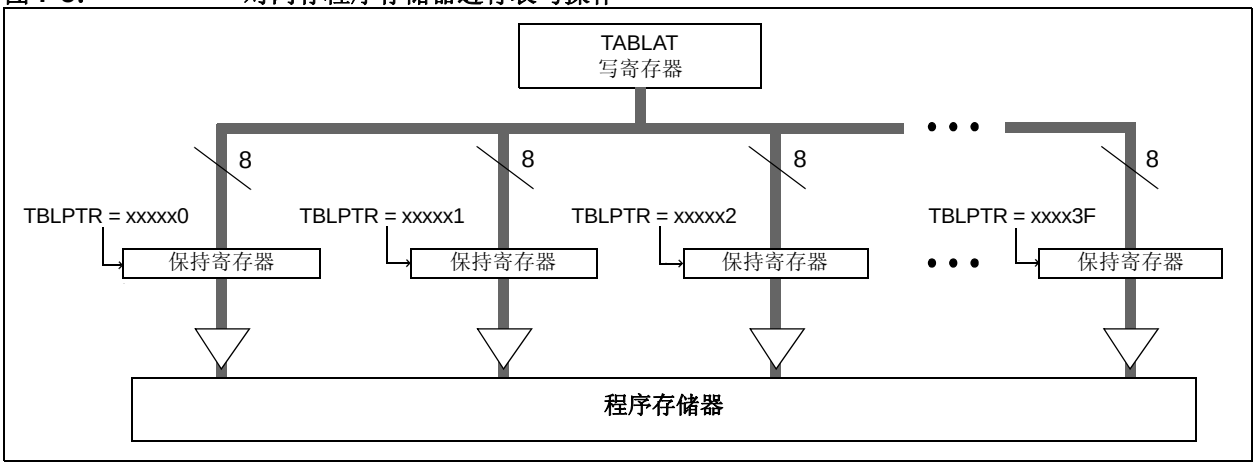
因为表锁存器（TABLAT）只是一个单字节寄存器，所以每次编程操作都必须执行 64 次 `TBLWT` 指令。由于只写入保持寄存器，所有表写操作实际上都是短周期写操作。在更新完 64 或 128 个保持寄存器后，必须写入 EECON1 寄存器，以便启动长写周期编程操作。

擦除内部闪存需要使用长写周期。在长写周期中，指令执行暂停。由内部编程定时器终止长周期写操作。

由 EEPROM 片上定时器控制写操作的时间。写 / 擦除电压由可在器件电压范围内工作的片上电荷泵产生。

> **注：** 器件复位时和写操作后，保持寄存器的默认值为 FFh。向保持寄存器写入 FFh 并不会修改寄存器中存储的字节。这意味着，假如不想将某个位从 `0` 改为 `1`，而仅修改程序存储器中的个别字节也是可以的。当修改个别字节时，在执行写操作前不必装入所有的 64 或 128 个保持寄存器。

图 7-5： 对闪存程序存储器进行表写操作



7.5.1 闪存程序存储器写操作序列

对内部程序存储单元编程的序列应为：

1. 将 64 或 128 个字节读入 RAM。
2. 必要时更新 RAM 中的数据值。
3. 把要擦除的目标地址装入表指针寄存器。
4. 执行行擦除操作。
5. 把要写入的第一个字节的地址装入表指针寄存器。
6. 采用自动递增的方式把 64 或 128 个字节写入保持寄存器。
7. 设置 EECON1 寄存器以执行写操作：
   - 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 位置 1 以使能字节写入
8. 禁止中断。
9. 向 EECON2 写入 0x55。
10. 向 EECON2 写入 0xAA。
11. 将 WR 位置 1。写周期开始。在写操作期间，CPU 将在 Tiw （见参数 D133A）时间段内停止工作。
12. 重新允许中断。
13. 对存储器进行校验 （通过表读）。

在例 7-3 给出了所需代码的示例

注：	在 WR 位置 1 前，表指针必须指向保持寄存器中存储的 64 或 128 字节的地址范围内。

注：	在 LP 振荡器模式 （低功耗模式）下运行时，将无法完成对闪存和 EEPROM 存储器的自写操作。因此，执行自写操作将使器件进入高功耗模式。

© 2011 Microchip Technology Inc.

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   0x55
必需的          MOVWF   EECON2                 ; write 55h
序列            MOVLW   0xAA
                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   0x55
必需的          MOVWF   EECON2              ; write 55h
序列            MOVLW   0xAA
                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

根据具体的应用，好的编程习惯一般要求使用原始值对写入存储器的值进行校验。在连续写入过多的数据已接近规范极限值的应用中，就应使用写校验。

7.5.3　写操作意外终止

如果写操作被意外事件（如掉电或意外复位）终止，则应对刚刚编程的存储单元进行验证，如果需要，还需对其进行重新编程。如果写操作在正常操作过程中被 MCLR 复位或 WDT 超时复位中断，用户就可以检测 WRERR 位的状态并重新写入此存储单元（如果需要）。

7.5.4　防止误写操作

为了防止误写闪存程序存储器，必须遵循写操作启动序列。更多详情请参见第 28.0 节 “CPU 的特殊功能”。

7.6　代码保护期间的闪存程序存储器操作

如需了解有关闪存程序存储器代码保护的详情，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 （非物理寄存器）							
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
IPR6	—	—	—	EEIP	—	CMP3IP	CMP2IP	CMP1IP
PIR6	—	—	—	EEIF	—	CMP3IF	CMP2IF	CMP1IF
PIE6	—	—	—	EEIE	—	CMP3IE	CMP2IE	CMP1IE

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

注　1：　TBLPTRU 的 bit 21 允许访问器件配置位。

© 2011 Microchip Technology Inc.

8.0 外部存储器总线

注： 外部存储器总线在 64 引脚器件上未实现。

外部存储器总线（EMB）使器件能像访问程序或数据存储器一样访问外部存储器件（如闪存、EPROM 或 SRAM）。外部存储器总线支持 8 位和 16 位数据宽度模式，以及 3 种最宽为 20 位的地址宽度模式。

该总线与 4 个 I/O 端口复用，占 28 个引脚。3 个端口（PORTD、PORTE 和 PORTH）与地址 / 数据总线复用，总共可使用 20 根线，而 PORTJ 与总线控制信号复用。

表 8-1 中列出了各个引脚及其功能。

表 8-1： PIC18F87K22 系列外部总线——I/O 端口功能

名称	端口	位	外部存储器总线功能
RD0/AD0	PORTD	0	地址 bit 0 或数据 bit 0
RD1/AD1	PORTD	1	地址 bit 1 或数据 bit 1
RD2/AD2	PORTD	2	地址 bit 2 或数据 bit 2
RD3/AD3	PORTD	3	地址 bit 3 或数据 bit 3
RD4/AD4	PORTD	4	地址 bit 4 或数据 bit 4
RD5/AD5	PORTD	5	地址 bit 5 或数据 bit 5
RD6/AD6	PORTD	6	地址 bit 6 或数据 bit 6
RD7/AD7	PORTD	7	地址 bit 7 或数据 bit 7
RE0/AD8	PORTE	0	地址 bit 8 或数据 bit 8
RE1/AD9	PORTE	1	地址 bit 9 或数据 bit 9
RE2/AD10	PORTE	2	地址 bit 10 或数据 bit 10
RE3/AD11	PORTE	3	地址 bit 11 或数据 bit 11
RE4/AD12	PORTE	4	地址 bit 12 或数据 bit 12
RE5/AD13	PORTE	5	地址 bit 13 或数据 bit 13
RE6/AD14	PORTE	6	地址 bit 14 或数据 bit 14
RE7/AD15	PORTE	7	地址 bit 15 或数据 bit 15
RH0/A16	PORTH	0	地址 bit 16
RH1/A17	PORTH	1	地址 bit 17
RH2/A18	PORTH	2	地址 bit 18
RH3/A19	PORTH	3	地址 bit 19
RJ0/ALE	PORTJ	0	地址锁存使能（ALE）控制引脚
RJ1/$\overline{OE}$	PORTJ	1	输出使能（$\overline{OE}$）控制引脚
RJ2/$\overline{WRL}$	PORTJ	2	低字节写（$\overline{WRL}$）控制引脚
RJ3/$\overline{WRH}$	PORTJ	3	高字节写（$\overline{WRH}$）控制引脚
RJ4/BA0	PORTJ	4	字节地址 bit 0（BA0）
RJ5/$\overline{CE}$	PORTJ	5	芯片使能（$\overline{CE}$）控制引脚
RJ6/$\overline{LB}$	PORTJ	6	低字节使能（$\overline{LB}$）控制引脚
RJ7/$\overline{UB}$	PORTJ	7	高字节使能（$\overline{UB}$）控制引脚

注： 为清楚起见，这里只显示了 I/O 端口和外部总线的分配情况。某些引脚上可能还有一个或多个其他复用功能。

8.1 外部存储器总线控制

接口的操作由 MEMCON 寄存器 （寄存器 8-1）控制。此寄存器在所有程序存储器工作模式 （单片机模式除外）下均可用。在单片机模式下，该寄存器被禁止且不能写入。

EBDIS 位 （MEMCON<7>）控制总线的操作和相关端口的功能。清零 EBDIS 将使能接口并禁止端口的 I/O 功能以及其他复用这些引脚的任何功能。将该位置 1 可使能 I/O 端口和其他功能，但允许接口在要求进行外部存储器操作时改写引脚上的数据。默认情况下，外部总线始终使能，并禁止其他所有 I/O。

对EBDIS位的操作也受到当前所使用的程序存储器模式的影响。**第 8.5 节 “程序存储器模式和外部存储器总线”**中将对此进行详细讨论。

WAIT 位支持延长外部存储器操作的等待状态。**第 8.3 节 “等待状态”**中将讨论这些位的用法。

WM 位用于选择总线工作在 16 位数据宽度模式下所使用的特定工作模式。**第 8.6 节 “16 位数据宽度模式”**中将详细讨论这些位。当选择使用 8 位数据宽度模式时，这些位不起作用。

寄存器 8-1： MEMCON：外部存储器总线控制寄存器 (1)

R/W-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
bit 7							bit 0

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

bit 7 **EBDIS**：外部总线禁止位
1 = 单片机访问外部存储器时使能外部总线，否则，所有外部总线驱动器都将映射为 I/O 端口
0 = 始终使能外部总线，禁止 I/O 端口

bit 6 未实现：读为 0

bit 5-4 **WAIT<1:0>**：表读和表写总线等待周期计数位
11 = 表读和表写将等待 0 个 TCY
10 = 表读和表写将等待 1 个 TCY
01 = 表读和表写将等待 2 个 TCY
00 = 表读和表写将等待 3 个 TCY

bit 3-2 未实现：读为 0

bit 1-0 **WM<1:0>**：16 位数据总线宽度的 TBLWT 操作选择位
1x = 字写模式：TABLAT 字输出；写 TABLAT 时 $\overline{\text{WRH}}$ 有效
01 = 字节选择模式：同时复制 TABLAT 数据的 MSB 和 LSB ；$\overline{\text{WRH}}$ 和 （$\overline{\text{UB}}$ 或 $\overline{\text{LB}}$）将有效
00 = 字节写模式：同时复制 TABLAT 数据的 MSB 和 LSB ； $\overline{\text{WRH}}$ 或 $\overline{\text{WRL}}$ 将有效

注 1： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

© 2011 Microchip Technology Inc.

8.2 地址和数据宽度

PIC18F87K22 系列器件可以将同一存储器总线配置成不同的地址和数据宽度。地址和数据宽度均由 CONFIG3L 寄存器中的配置位设置。作为配置位，这意味着这些选项只能通过编程器件进行配置，而不能由软件控制。

BW 位选择 8 位或 16 位数据总线宽度。该位置 1（默认值）将选择 16 位数据宽度。

ABW<1:0> 位确定程序存储器工作模式和地址总线宽度。可用选项包括 20 位、 16 位和 12 位，以及单片机模式（禁止外部总线）。选择 16 位或 12 位宽度时，多出的高位线用于 I/O 功能。这些引脚不再受 EBDIS 位设置的影响。例如，选择16位寻址模式（ABW<1:0> = 01）会禁止 A<19:16>，并使 PORTH<3:0> 工作不受总线影响。使用较小的地址宽度使用户可根据特定设计的外部存储器空间的大小定制存储器总线，而使空出的引脚专用于 I/O 操作。

由于 ABW 位可禁止引脚用于存储器总线操作，因此，要注意选择的地址宽度要始终大于或等于数据宽度。如果 12 位地址宽度配以 16 位数据宽度，那么总线将无法得到数据的高 4 位。

地址和数据宽度的所有组合都要求在相同线上复用地址和数据信息。表 8-2 中总结了地址和数据复用，以及使用较小的地址宽度时可用的 I/O 端口。

8.2.1 外部总线上的地址平移

默认情况下，外部总线上的地址就是 PC 的值。也就是说对单片机而言，外部存储器中地址低于片内存储器上边界的单元是不可用的。要访问这些物理单元，单片机和外部存储器之间的连接逻辑必须设法对地址进行转换。

为简化接口，外部总线提供了一种扩展的单片机模式，它能自动进行地址平移。此功能由 EASHFT 配置位控制。将该位置 1 会使总线上的地址发生偏移，偏移量为单片机片内程序存储器的大小，且下限地址设置为 0000h。这样器件就可以使用外部存储器的整个物理地址范围。

8.2.2 21 位寻址

作为 20 位地址宽度操作的扩展，外部存储器总线还可以寻址整个 2 MB 存储空间。将总线地址 bit 0 （BA0）控制线用作地址的最低位可实现上述功能。在某些存储器件中，还可以使用 UB 和 LB 控制信号来选择 16 位宽数据字中的高字节和低字节。

此寻址模式可用于 8 位和某些 16 位数据宽度模式。第 8.6.3 节 “16 位字节选择模式”和第 8.7 节 “8 位数据宽度模式”中提供了更多详细信息。

表 8-2： 不同地址和数据宽度的地址和数据线

数据宽度	地址宽度	复用的数据和地址线（及相应的端口）	仅地址线（及相应的端口）	可用作 I/O 的端口
8 位	12 位	AD<7:0>（PORTD<7:0>）	AD<11:8>（PORTE<3:0>）	PORTE<7:4>，PORTH 的所有引脚
	16 位		AD<15:8>（PORTE<7:0>）	PORTH 的所有引脚
	20 位		A<19:16>， AD<15:8>（PORTH<3:0> 和 PORTE<7:0>）	—
16 位	16 位	AD<15:0>（PORTD<7:0> 和 PORTE<7:0>）	—	PORTH 的所有引脚
	20 位		A<19:16>（PORTH<3:0>）	—

8.3 等待状态

人们可能会认为外部存储器件工作在单片机的时钟速率下，但通常不是这样。事实上，许多器件写数据或检索数据所需的时间长于表读或表写操作允许的时间。

为补偿时间差，可以配置外部存储器总线以在每次使用总线进行表操作时添加一个固定延时。通过将 WAIT 配置位置 1 使能等待状态。使能后，可通过 WAIT<1:0> 位（MEMCON<5:4>）设置延迟时间。延时必须是单片机指令周期时间的整数倍，添加在执行表操作的指令周期后。延时范围为 0 到 3 个 TCY （默认值）。

8.4 端口引脚弱上拉

除了高地址线 A<19:16>，与外部存储器总线相关的引脚均配有弱上拉电路。上拉电路由 PADCFG1 寄存器的高 3 位（PADCFG1<7:5>）控制。这三位命名为 RDPU、REPU 和 RJPU，分别用于控制 PORTD、 PORTE 和 PORTJ 上的弱上拉电路。将其中任一位置 1 会使能相应端口的上拉电路。默认情况下，器件复位时禁止所有上拉电路。

在扩展单片机模式下，端口上拉可用于当总线暂时被禁止 （EBDIS = 1）时，在外部总线上保持存储器状态。

8.5 程序存储器模式和外部存储器总线

同时使用片上和外部程序存储器时， PIC18F87K22 系列器件可工作在两种程序存储器模式下。复用端口引脚的功能取决于所选的程序存储器模式以及EBDIS位的设置。

在**单片机模式下**，总线无效，引脚只具有端口功能。不允许写 MEMCOM 寄存器。EBDIS 的复位值（0）被忽略，且 ABW 引脚充当 I/O 端口。

在**扩展单片机模式下**，外部程序存储器总线与引脚的 I/O 端口功能复用。当器件在外部程序存储空间进行取指或表读 / 表写操作时，引脚将具有外部总线功能。

如果器件只是对内部程序存储单元进行取指或访问，EBDIS 控制位将把引脚从外部存储器功能更改为 I/O 端口功能。 EBDIS = 0 时，引脚用作外部总线。EBDIS = 1 时，引脚用作 I/O 端口。

如果器件在 EBDIS = 1 时取指或访问外部存储器，引脚将切换为外部总线功能。如果外部存储器中执行的程序将 EBDIS 位置 1，那么将该位置 1 的操作将延迟到程序跳转到内部存储器才得以执行。此时，引脚的功能将从外部总线更改为 I/O 端口。

如果器件在 EBDIS = 0 时针对内部存储器执行操作，存储器总线的地址 / 数据和控制引脚将无效。它们进入以下状态：有效地址 / 数据引脚为三态， $\overline{CE}$、 $\overline{OE}$、$\overline{WRH}$、$\overline{WRL}$、$\overline{UB}$ 和 $\overline{LB}$ 信号为 1，而 ALE 和 BA0 为 0。请注意，只有那些与当前地址宽度相关的引脚被强制为三态；其他引脚继续充当 I/O。例如，在 16 位地址宽度下，仅 AD<15:0> （PORTD 和 PORTE）受影响； A<19:16> （PORTH<3:0>）继续用作 I/O。

在所有外部存储器模式下，总线的优先级高于任何其他与其共享引脚的外设，包括并行主端口 （Parallel Master Port ， PMP）和串行通信模块，此二者的优先级高于 I/O 端口。

8.6 16 位数据宽度模式

在 16 位数据宽度模式下，外部存储器接口能以 3 种不同的配置连接到外部存储器：

- 16 位字节写
- 16 位字写
- 16 位字节选择

要使用的配置由 MEMCON 寄存器中的 WM<1:0> 位（MEMCON<1:0>）确定。这三种不同的配置使设计人员在使用 8 位和 16 位器件处理 16 位数据时获得最大的灵活性。

对于所有 16 位模式，地址锁存使能 （ALE）引脚表示外部存储器接口总线上的地址位 AD<15:0> 可用。在地址锁存之后，输出使能 （$\overline{OE}$）信号将同时使能程序存储器的两个字节，以形成一个 16 位的指令字。在单片机访问外部存储器（无论读写）的任何时间内芯片使能信号 （$\overline{CE}$）均有效；只要器件处于休眠模式，信号就变为无效 （高电平）。

在字节选择模式下， JEDEC 标准闪存存储器需要 BA0 作为字节地址线，并使用一条 I/O 线在字节模式和字模式之间进行选择。其他 16 位模式不需要 BA0。JEDEC 标准静态 RAM 存储器将使用 $\overline{UB}$ 或 $\overline{LB}$ 信号选择字节。

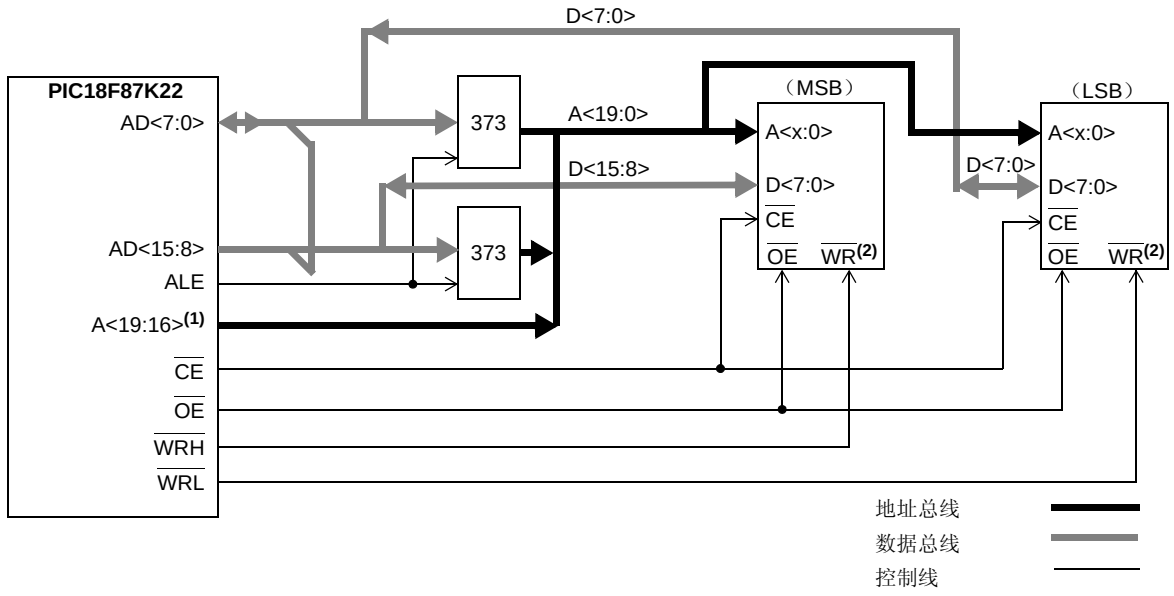
© 2011 Microchip Technology Inc.

8.6.1 16 位字节写模式

图 8-1 给出了 PIC18F87K22 系列器件的 16 位字节写模式的示例。此模式用于将两个单独的 8 位存储器相连以进行 16 位操作。这通常包括基本 EPROM 和闪存器件。它允许对字节宽的外部存储器执行表写操作。

在TBLWT指令周期内，TABLAT中的数据送到AD<15:0>总线的高字节和低字节中。TBLPTR 的 LSb 选通相应的 $\overline{WRH}$ 或 $\overline{WRL}$ 控制线。

图 8-1：16 位字节写模式示例



注 1：高位地址线仅用于 20 位地址宽度。
2：此信号仅适用于表写操作。请参见第 7.1 节 “表读和表写”。

8.6.2 16 位字写模式

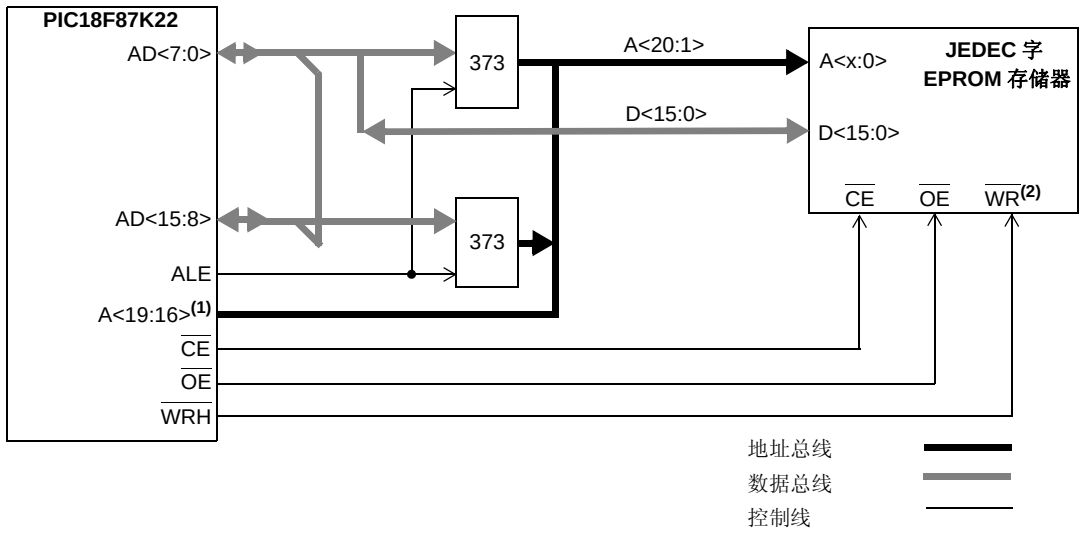
图 8-2 给出了 PIC18F87K22 系列器件的 16 位字写模式的示例。此模式用于字宽度存储器，包括一些 EPROM 和闪存类存储器。此模式允许从所有形式的 16 位存储器取操作码和执行表读操作，以及对任意类型字宽度的外部存储器执行表写操作。此方法将访问偶地址和奇地址的 TBLWT 周期区分开来。

在写入偶地址 （TBLPTR<0> = 0）的 TBLWT 周期中，TABLAT 数据被转送到保持锁存器中，且外部地址数据总线在总线周期的传送数据期间为三态。不会激活任何写信号。

在写入奇地址 （TBLPTR<0> = 1）的 TBLWT 周期中，TABLAT 中的数据传送到 AD<15:0> 总线的高字节。保持锁存器的内容传送到 AD<15:0> 总线的低字节。

每个写周期 $\overline{\text{WRH}}$ 信号均选通；未使用 $\overline{\text{WRL}}$ 引脚。BA0 引脚上的信号表示 TBLPTR 的 LSb，但该引脚是悬空的。而使用 $\overline{\text{UB}}$ 和 $\overline{\text{LB}}$ 信号来选择两个字节。此方法的明显缺陷是表写操作必须在特定边界上成对完成，才能正确地写字单元。

图 8-2： 16 位字写模式示例



注 1： 高位地址线仅用于 20 位地址宽度。
2： 此信号仅适用于表写操作。请参见第 7.1 节 “表读和表写”。

© 2011 Microchip Technology Inc.

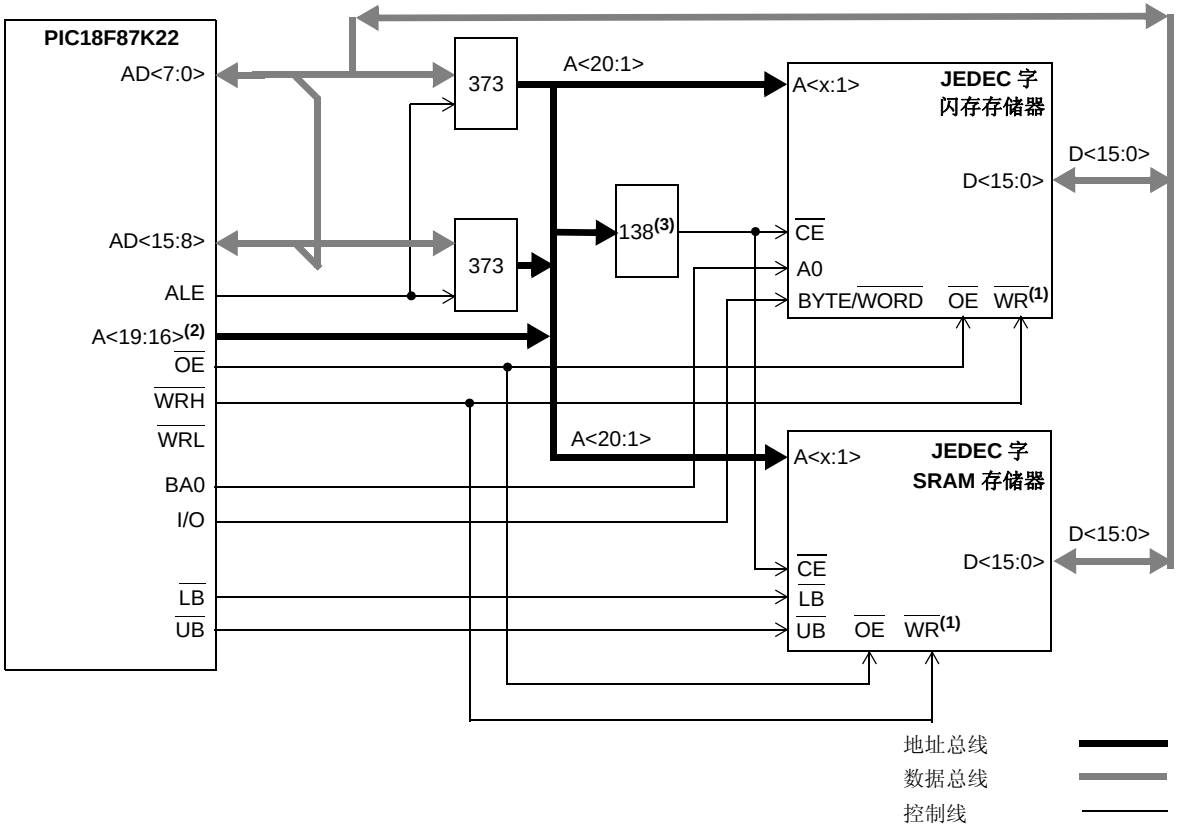
8.6.3 16 位字节选择模式

图 8-3 给出了 16 位字节选择模式的示例。此模式允许对字宽度的外部存储器进行表写操作，同时还可以进行字节选择。这通常包括字宽度的闪存和 SRAM 器件。

在TBLWT周期内，TABLAT中的数据被传送到AD<15:0>总线的高字节和低字节中。在每个写周期都选通 WRH 信号；未使用 WRL 引脚。BA0 或 UB/LB 信号用于根据 TBLPTR 寄存器的最低有效位来选择要写的字节。

闪存和SRAM器件使用不同的控制信号组合来实现字节选择模式。 JEDEC 标准闪存存储器要求将一个控制器 I/O 端口引脚连接到存储器的 BYTE/WORD 引脚，以提供选择信号。它们还将控制器中的 BA0 信号用作字节地址。而 JEDEC 标准静态 RAM 存储器使用 UB 或 LB 信号选择字节。

图 8-3： 16 位字节选择模式示例



注 1： 此信号仅适用于表写操作。请参见第 7.1 节 “表读和表写”。
2： 高位地址线仅用于 20 位地址宽度。
3： 仅当访问多个存储器件时才需要进行信号分离。

8.6.4 16 位模式时序

不同工作模式下外部存储器总线上的控制信号是不同的。图 8-4 和图 8-5 给出了典型的信号时序图。

图 8-4： 执行 TBLRD 指令时的外部存储器总线时序（扩展单片机模式）

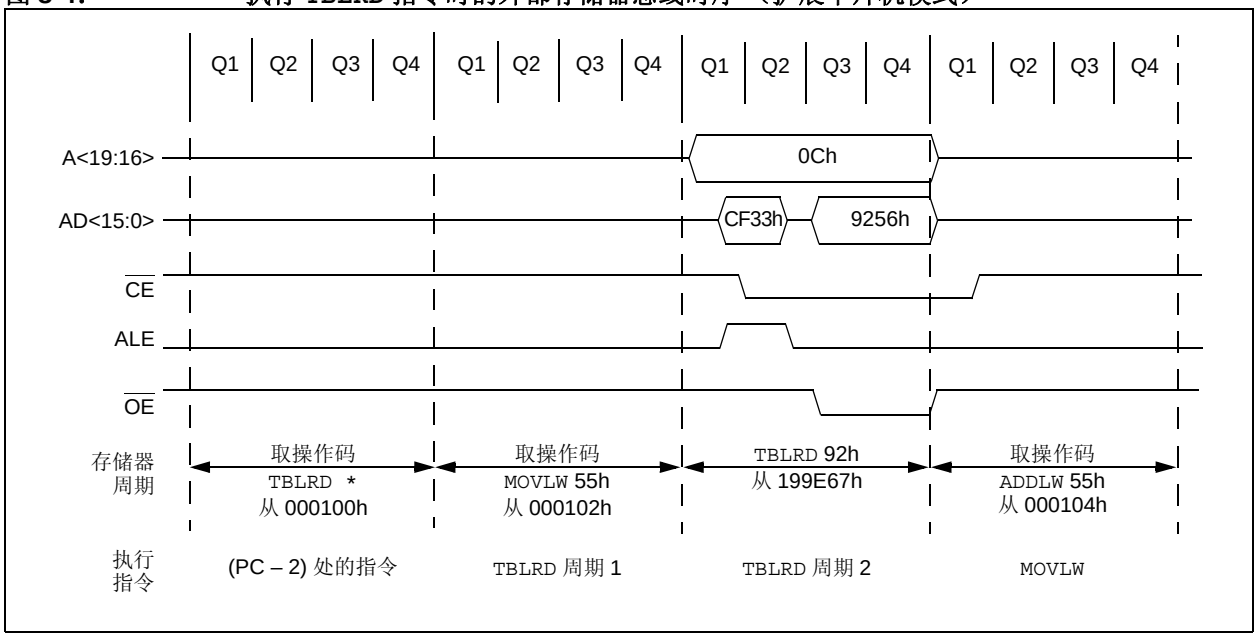
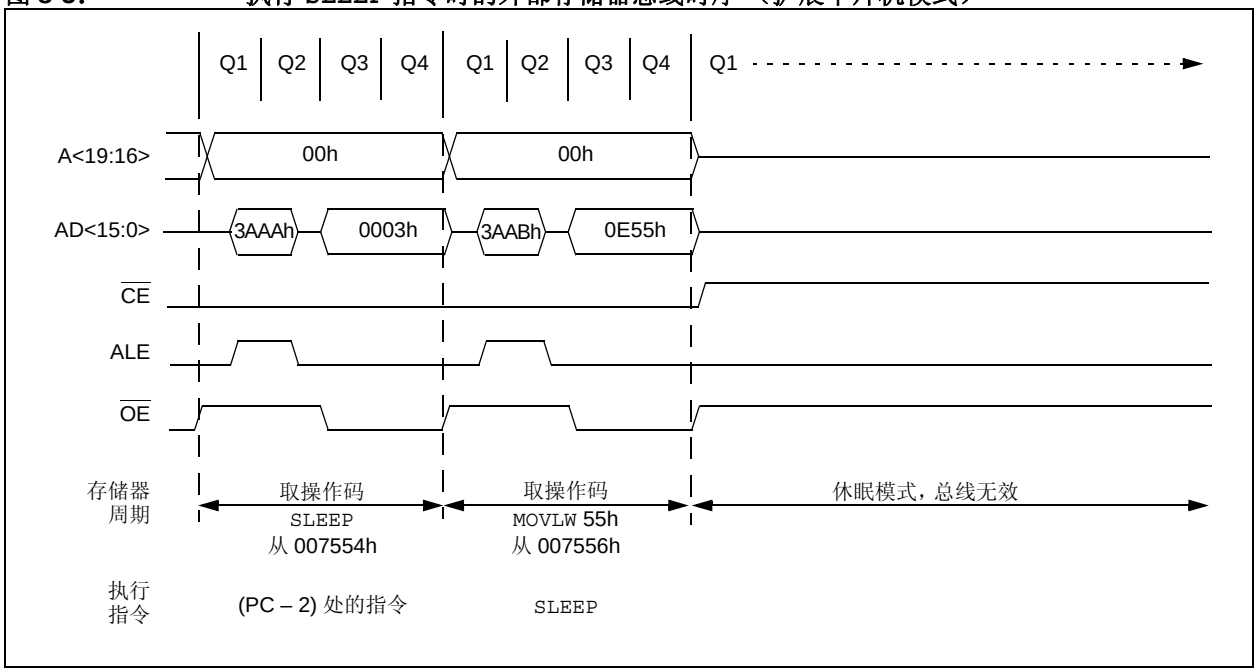


图 8-5： 执行 SLEEP 指令时的外部存储器总线时序（扩展单片机模式）



© 2011 Microchip Technology Inc.

8.7 8 位数据宽度模式

在 8 位数据宽度模式下，外部存储器总线仅工作在复用模式下；即，数据总线与地址总线的低 8 位复用。

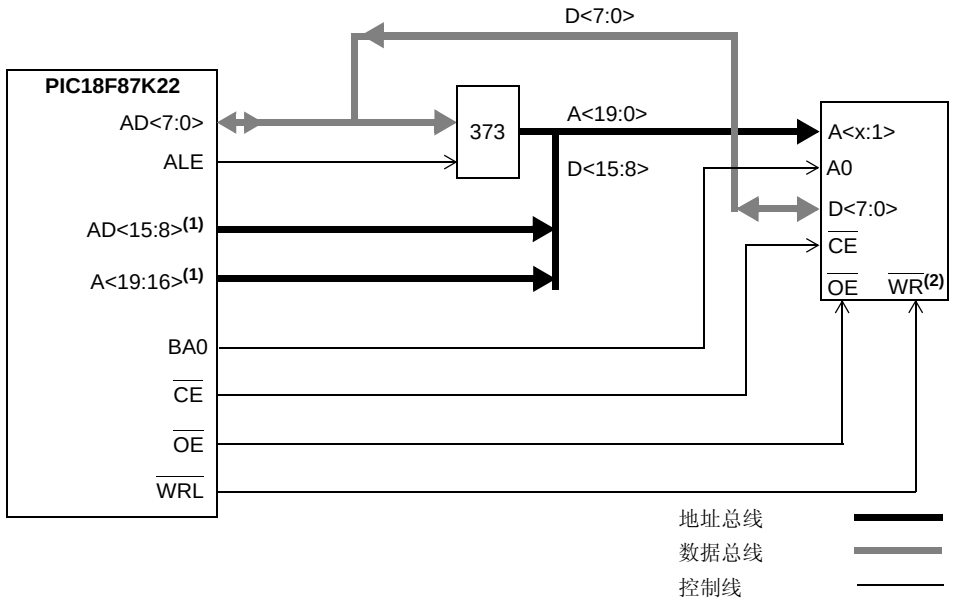
图 8-6 给出了 PIC18F8XK22 器件的 8 位复用模式的示例。此模式用于连接单个 8 位存储器以进行 16 位操作。在数据 / 地址复用的总线上将一条指令作为两个 8 位字节取出，这两个字节在一个指令周期 （TCY）内顺序取指。因此，设计人员必须基于 1/2 TCY（2 倍指令速率）计算时序，并据此选择外部存储器。为了选择合适的存储器速度，必须考虑连接逻辑传输延迟时间，以及建立时间和保持时间。

地址锁存使能 （ALE）引脚指示地址位 AD<15:0> 在外部存储器接口总线上是否可用。输出使能信号 （$\overline{OE}$）将在指令周期的部分时间内使能程序存储器的一个字节，然后 BA0 将发生更改，并将使能第二个字节以构成 16 位指令字。在该模式下，地址的最低有效位 BA0 必须连接到存储器件。在单片机访问外部存储器（无论读写）的任何时间内芯片使能信号 （$\overline{CE}$）均有效；只要器件处于休眠模式，信号就变为无效 （高电平）。

这通常包括基本 EPROM 和闪存器件。它允许对字节宽度的外部存储器执行表写操作。

在 TBLWT 指令周期内， TABLAT 中的数据传送到 AD<15:0> 总线的高字节和低字节。TBLPTR 的 LSb 选通相应级别的 BA0 控制线。

图 8-6： 8 位复用模式示例



注 1： 高位地址位仅用于 20 位地址宽度。地址高字节用于除 8 位以外的所有地址宽度。
2： 此信号仅适用于表写操作。请参见第 7.1 节 “表读和表写”。

8.7.1 8 位模式时序

不同工作模式下外部存储器总线上的控制信号是不同的。图 8-7 和图 8-8 给出了典型的信号时序图。

图 8-7： 执行 TBLRD 指令时的外部存储器总线时序（扩展单片机模式）

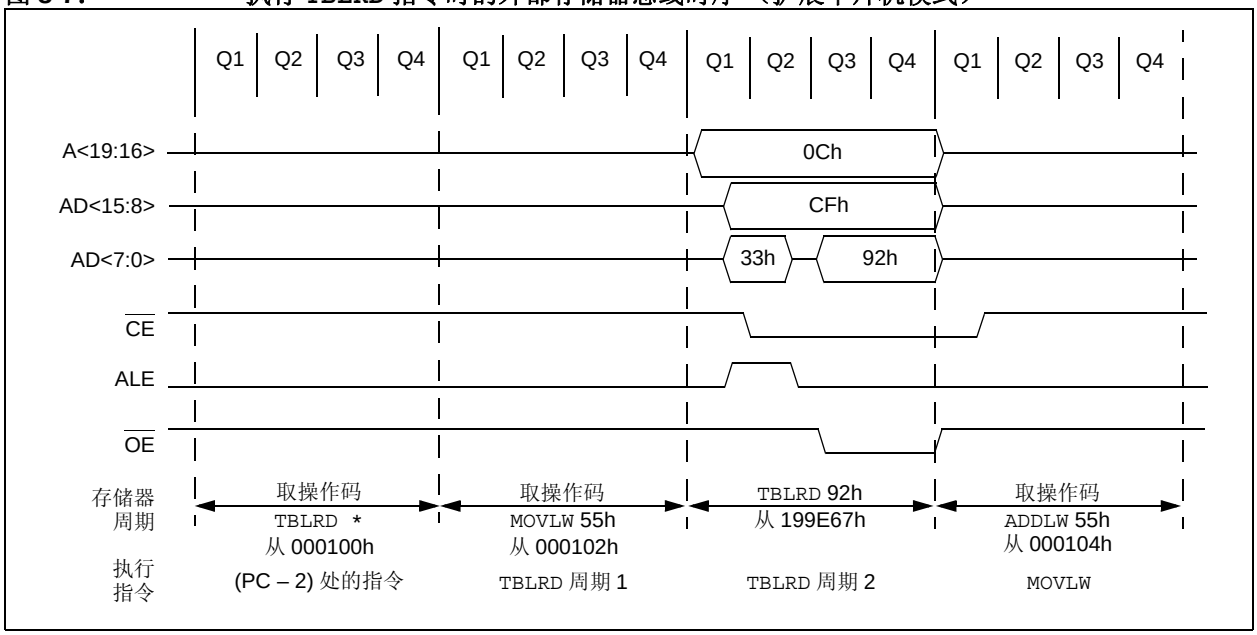
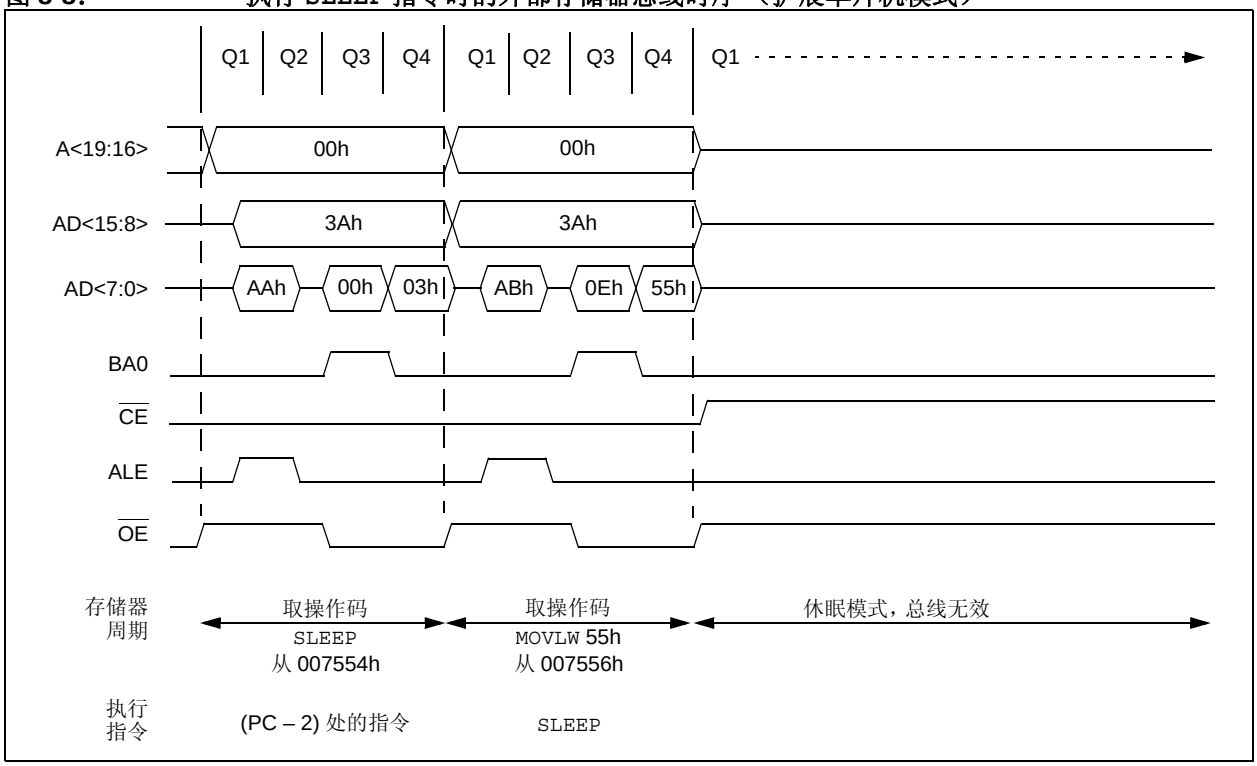


图 8-8： 执行 SLEEP 指令时的外部存储器总线时序（扩展单片机模式）



© 2011 Microchip Technology Inc.

8.8 在功耗管理模式下的操作

在所有功耗管理运行模式下，外部总线都能继续正常工作。如果选择一个低速的时钟源，总线操作将以该速率运行。在这些情况下，如果使能了等待状态并把它添加到外部存储器操作中，可能会导致对外部存储器的访问时间过长。如果希望工作在低功耗运行模式下，用户应该在应用程序中调整在较低的时钟速率下访问存储器的时间。

在休眠和空闲模式下，单片机内核不需要访问数据；总线操作暂停。外部总线的状态被冻结，地址 / 数据引脚和多数控制引脚都保持在进入该模式前的状态。惟一可能变化的是 $\overline{CE}$、$\overline{LB}$ 和 $\overline{UB}$ 引脚，它们保持为逻辑高电平。

表 8-3： 与外部存储器总线相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MEMCON(1)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
PADCFG1	RDPU	REPU	RJPU(1)	—	—	RTSECSEL1	RTSECSEL0	—
PMD1	PSPMD	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBMD

图注： — = 未实现，读为 0。外部存储器总线访问期间不使用阴影单元。

注 1： 在 64 引脚器件（PIC18F6XK22）上未实现，读为 0。

注：

© 2011 Microchip Technology Inc.

9.0 数据 EEPROM 存储器

与数据 RAM 和程序存储器不同，数据 EEPROM 是非易失性存储阵列，用来长期保存程序数据。EEPROM 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。数据 EEPROM 在整个 VDD 范围内正常运行时是可读写的。

有 5 个 SFR 用于读写数据 EEPROM 和程序存储器。它们是：

- EECON1
- EECON2
- EEDATA
- EEADR
- EEADRH

EEPROM 数据存储器允许按字节读写。在与数据存储器模块进行数据交互时，EEDATA 内存放 8 位读 / 写数据，而 EEADRH:EEADR 寄存器对存放被访问的 EEPROM 存储单元地址。

EEPROM 数据存储器具有高耐擦写能力。字节写操作会自动擦除该单元然后写入新的值（即先擦后写）。写入时间由片上定时器控制。根据电压、温度和芯片的不同，写入时间也有差异。具体的限制请参见参数 D122（**第 31.0 节 “电气特性”**中的表 31-1）。

9.1 EEADR 和 EEADRH 寄存器

EEADRH:EEADR 寄存器对用于寻址要进行读写操作的数据 EEPROM。EEADRH 保存该地址的高两位，该寄存器的高 6 位不用，应忽略。 10 位的寄存器对可寻址 1024 字节的存储空间（00h 到 3FFh）。

9.2 EECON1 和 EECON2 寄存器

对数据 EEPROM 的访问由 EECON1 和 EECON2 这两个寄存器控制。EECON1 和 EECON2 这对寄存器也控制对程序存储器的访问，访问方式与访问数据 EEPROM 相同。

EECON1 寄存器（寄存器 9-1）用于控制对数据和程序存储器的访问。控制位 EEPGD 决定访问的是程序存储器还是数据 EEPROM 存储器。清零时，将访问数据 EEPROM 存储器。置 1 时，则访问程序存储器。

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。置 1 时，后续操作会访问配置寄存器。而当 CFGS 清零时，则由 EEPGD 位选择访问闪存程序存储器还是数据 EEPROM 存储器。

若将 WREN 位置 1，则允许写操作。上电时将清零 WREN 位。 WREN 位置 1 时， WRERR 位将由硬件置 1 ；当内部编程定时器超时并且写操作完成时，将清零 WRERR 位。

注：	如果在正常工作期间，WRERR 读为 1，则表明写操作因复位而提早终止或进行了不正确的写操作。

控制位 WR 用于启动写操作。软件只能将此位置 1 而不能清零。该位在写操作完成时由硬件清零。

注：	当写操作完成时 EEIF 中断标志位（PIR6<4>）置 1。此标志位必须由软件清零。

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

在访问程序存储器（EEPGD = 1）时，RD 位无法置 1。程序存储器是通过表读指令读取的。有关表读操作的信息，请参见**第 7.1 节 “表读和表写”**。

EECON2 寄存器不是物理寄存器。它专用于存储器的擦写序列。读 EECON2 得到的是全 0。

寄存器 9-1： 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 = 上电复位时的值	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 = 在下一个写命令中擦除 TBLPTR 寻址到的程序存储器行 （擦除操作完成后将清零）
0 = 仅执行写操作

bit 3 **WRERR**：闪存程序存储器 / 数据 EEPROM 错误标志位 [1]
1 = 写操作提早终止 （由正常工作中自定时编程期间的任何复位或不正确写入引起）
0 = 写操作已经完成

bit 2 **WREN**：闪存程序存储器 / 数据 EEPROM 写使能位
1 = 允许闪存程序存储器 / 数据 EEPROM 的写周期
0 = 禁止闪存程序存储器 / 数据 EEPROM 的写周期

bit 1 **WR**：写控制位
1 = 启动数据 EEPROM 擦 / 写周期或程序存储器擦写周期。（该操作是自定时的，一旦写入完成该位即由硬件清零。软件只能将 WR 位置 1 而不能清零。）
0 = 写入 EEPROM 的周期结束

bit 0 **RD**：读控制位
1 = 启动 EEPROM 读操作
（读操作需要一个周期。RD 位由硬件清零。软件只能将 RD 位置 1 而不能清零。当 EEPGD = 1 或 CFGS = 1 时， RD 位不能置 1。）
0 = 未启动 EEPROM 读操作

注 1： 当发生 WRERR 时， EEPGD 和 CFGS 位不会被清零。这允许跟踪错误条件。

© 2011 Microchip Technology Inc.

9.3 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。一个周期结束后，数据存储在 EEDATA 寄存器中，从而可在一条 NOP 指令之后对其进行读操作。EEDATA 将保持这个值直到另一个读操作开始，或直到它被用户写入 （在写操作过程中）。

读数据的基本过程如例 9-1 所示。

9.4 写入数据 EEPROM 存储器

要写 EEPROM 数据存储单元，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 9-2 中的执行序列来启动写周期。

如果未完全按照以上序列（将 0x55 写入 EECON2、将 0xAA 写入 EECON2，然后将 WR 位置 1）逐字节写入，写操作将不会开始。在执行该代码段时，强烈建议禁止中断。

此外，EECON1 中的 WREN 位也必须被置 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）造成对数据 EEPROM 的误写入。除了在更新 EEPROM 时，WREN 位将一直保持清零状态。WREN 位不能由硬件清零。

写序列启动以后，就不能修改 EECON1、EEADRH:EEADR 和 EEDATA 了。除非将 WREN 位置 1，否则将不允许将 WR 位置 1。 WREN 必须在前一指令中被置 1。 WR 和 WREN 位不能由同一指令置 1。

写周期完成后， WR 位将由硬件清零，同时 EEPROM 中断标志位（EEIF）被置 1。用户可以允许该中断或对 WR 位进行查询。 EEIF 必须由软件清零。

9.5 写校验

根据具体的应用，好的编程习惯一般要求使用原始值对写入存储器的值进行校验。在连续写入过多的数据已接近规范极限值的应用中，就应使用写校验。

注：	在 LP 振荡器模式 （低功耗模式）下运行时，将无法对闪存和 EEPROM 存储器执行自写操作。因此，执行自写操作将使器件进入高功耗模式。

例 9-1：　　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 9-2：　　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   0x55               ;
必需的          MOVWF   EECON2             ; Write 55h
序列            MOVLW   0xAA               ;
                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)
```


© 2011 Microchip Technology Inc.

9.6 代码保护期间的工作原理

数据 EEPROM 存储器在配置字中具有自己的代码保护位。如果使能了代码保护，将禁止对其进行外部读写操作。

无论代码保护配置位的状态如何，单片机本身都可以读写内部数据 EEPROM。更多信息，请参见**第 28.0 节 “CPU 的特殊功能”**。

9.7 防止误写操作

有些情况下，器件可能不希望向数据 EEPROM 写入数据。为了防止误写 EEPROM，实施了多种保护机制。上电时将清零 WREN 位。此外，上电延时期间（表 31-13 中的 TPWRT，参数 33）也禁止写 EEPROM。

在掉电、电源不稳定或软件故障期间，写操作启动序列和 WREN 位可共同防止发生意外写操作。

9.8 使用数据 EEPROM

数据 EEPROM 是高耐擦写、可字节寻址的阵列，已对其进行优化，允许存储频繁更改的信息（例如，程序变量或其他经常更新的数据）。有些值更新较为频繁，可能会超过 D124 规范中的规定。如果情况如此，就必须执行阵列刷新。因此，不常更改的变量（例如常数、ID 和校准值等）应该存储在闪存程序存储器中。

例 9-3 所示为简单的数据 EEPROM 刷新程序。

> **注：** 如果数据EEPROM仅用于存储经常更改的常数和 / 或数据，可能不必进行阵列刷新。请参见 D124 规范。

例 9-3： 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   0x55                ;
        MOVWF   EECON2              ; Write 55h
        MOVLW   0xAA                ;
        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 9-1： 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（非物理寄存器）							
EECON1	EEPGD	CFGS	—	FREE	WRERR	WREN	WR	RD
IPR6	—	—	—	EEIP	—	CMP3IP	CMP2IP	CMP1IP
PIR6	—	—	—	EEIF	—	CMP3IF	CMP2IF	CMP1IF
PIE6	—	—	—	EEIE	—	CMP3IE	CMP2IE	CMP1IE

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

© 2011 Microchip Technology Inc.

10.0 8 x 8 硬件乘法器

10.1 简介

所有的 PIC18 器件均包含一个 8 x 8 硬件乘法器（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 1 个指令周期。硬件乘法器具有更高的计算吞吐量并减少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 10-1 给出了各种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。

10.2 工作原理

例 10-1 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乘数时，实现该运算仅需一条指令。

例 10-2 给出了一个 8 x 8 有符号乘法运算的指令序列。要弄清结果的符号位，必须检查每个乘数的最高有效位（MSb），并做相应的减法。

例 10-1： 8 x 8 无符号乘法程序

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

例 10-2： 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 10-1： 各种乘法运算的性能比较

程序	乘法实现方法	程序存储空间（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

© 2011 Microchip Technology Inc.

例 10-3 给出了 16 x 16 无符号乘法运算的指令序列。公式 10-1 为所使用的算法。32 位结果存储在 4 个寄存器（RES3:RES0）中。

公式 10-1： 16 x 16 无符号乘法算法

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

例 10-3： 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 10-4 给出了一个 16 x 16 有符号乘法运算的指令序列。公式 10-2 为所使用的算法。32 位结果存储在 4 个寄存器（RES3:RES0）中。要弄清结果的符号位，必须检查每个乘数的最高有效位（MSb），并做相应的减法。

公式 10-2： 16 x 16 有符号乘法算法

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \bullet ARG2H:ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H<7> \bullet ARG1H:ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H<7> \bullet ARG2H:ARG2L \bullet 2^{16})
\end{aligned}
$$

例 10-4： 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
    :
```


© 2011 Microchip Technology Inc.

11.0 中断

PIC18F87K22 系列器件的各个成员提供多个中断源及一个中断优先级功能，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。高优先级中断事件将中断任何正在处理的低优先级中断。

用于控制中断操作的寄存器有：

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1、 PIR2 和 PIR3
- PIE1、 PIE2 和 PIE3
- IPR1、 IPR2 和 IPR3

推荐使用由 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内的这些位。

通常，用三个位来控制中断源的操作。它们是：

- **标志位**——表明发生了中断事件
- **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**——用于指定高优先级或低优先级

通过将 IPEN 位 （RCON<7>）置 1，可允许中断优先级功能。当允许了中断优先级时，有 2 个全局中断允许位可用。将 GIEH 位 （INTCON<7>）置 1，可允许所有优先级位已置 1 （高优先级）的中断。将 GIEL 位（INTCON<6>）和 GIEH 位 （INTCON<7>）置 1，可允许所有优先级位已清零（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，程序将立即跳转到地址 0008h 或 0018h，具体地址取决于优先级位的设置。通过设置相应的允许位可以禁止各个中断。

当 IPEN 位被清零 （默认状态）时，便会禁止中断优先级功能，此时中断与 PIC® 中档系列器件相兼容。在兼容模式下，各个中断源的中断优先级位均不起作用。INTCON<6> 是 PEIE 位，用于允许 / 禁止所有的外设中断源。 INTCON<7> 是 GIE 位，用于允许 / 禁止所有的中断源。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其他中断。如果清零 IPEN 位，全局中断允许位就是 GIE 位。如果使用了中断优先级，这个位将是 GIEH 位或者 GIEL 位。高优先级中断源可以中断低优先级中断。处理高优先级中断时，低优先级中断将不被响应。

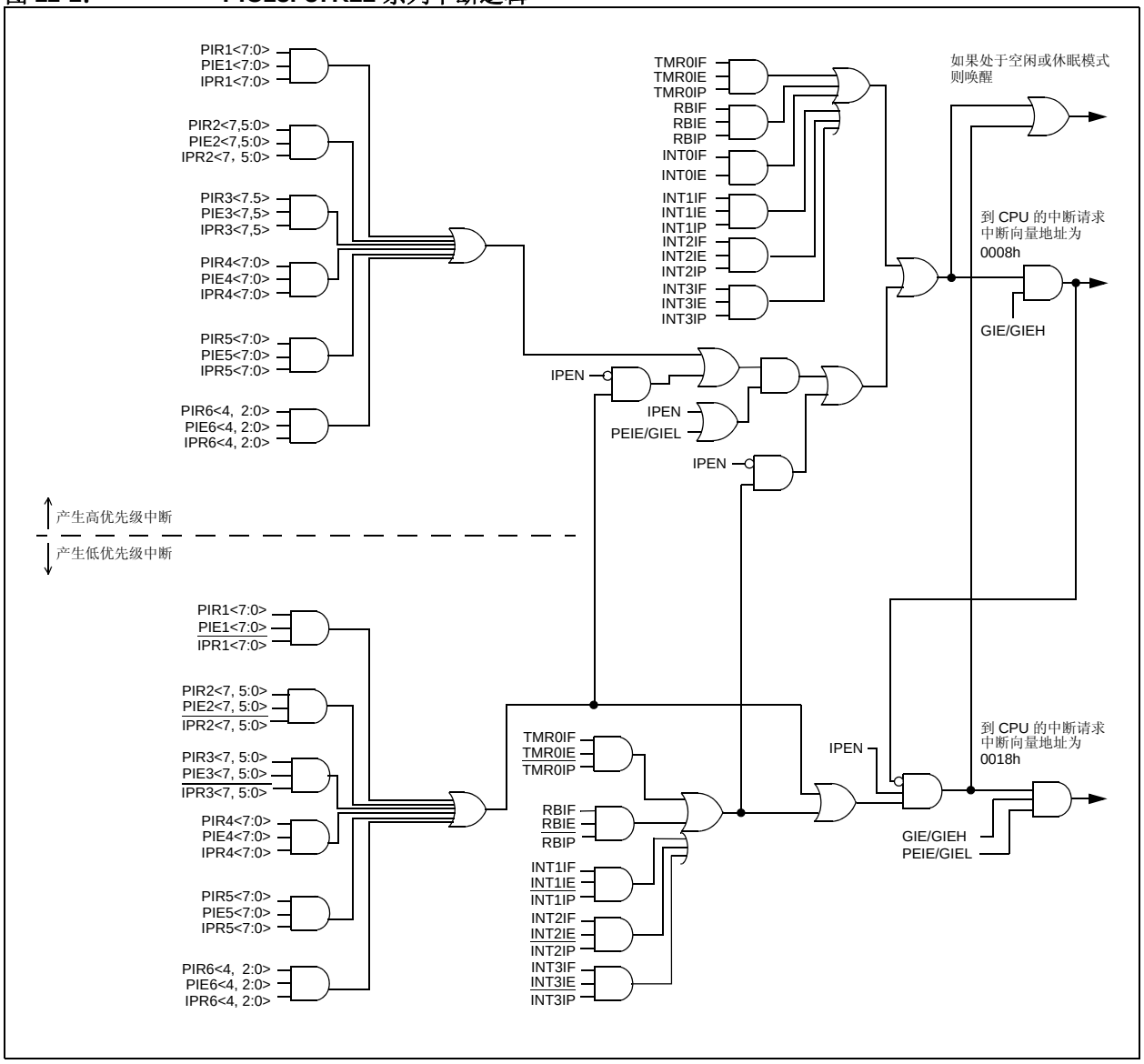
返回地址被压入堆栈，中断向量地址（0008h 或 0018h）被装入 PC。进入中断服务程序 （Interrupt Service Routine， ISR）之后，就可以通过查询中断标志位来确定中断源。在重新允许中断前，必须用软件将中断标志位清零，以避免重复响应该中断。

执行 “中断返回” 指令 `RETFIE` 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，诸如 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 到 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**任何**中断被允许时，不要使用 `MOVFF` 指令来修改中断控制寄存器，否则可能导致单片机操作出错。

图 11-1： PIC18F87K22 系列中断逻辑



© 2011 Microchip Technology Inc.

11.1 INTCON 寄存器

INTCON 寄存器是可读写的寄存器，包含各个允许位、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之前，确保先将相应的中断标志位清零。中断标志位可由软件查询。

寄存器 11-1： 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	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 = 上电复位时的值	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 = 允许所有低优先级外设中断
0 = 禁止所有低优先级外设中断

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
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 寄存器已发生溢出 （必须由软件清零）
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 （必须由软件清零）
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 （必须由软件清零）
0 = RB<7:4> 引脚的电平状态没有改变

注 1： 电平不匹配条件会一直不断地将此位置 1。读取 PORTB，然后等待一个额外的指令周期可以结束这种情况，并将该位清零。

寄存器 11-2： INTCON2：中断控制寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	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 = 上电复位时的值	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 = 按各个 TRIS 寄存器值使能 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之前，确保先将相应的中断标志位清零。中断标志位可由软件查询。

© 2011 Microchip Technology Inc.

寄存器 11-3： 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-0	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 = 上电复位时的值	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 （必须由软件清零）
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 （必须由软件清零）
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 （必须由软件清零）
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之前，确保相应的中断标志位已经清零。中断标志位可由软件查询。

11.2 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，有六个外设中断请求 （标志）寄存器 （PIR1 至 PIR6）。

> 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：用户软件应在允许一个中断之前，确保相应的中断标志位已经清零；同时在处理完中断后，也应该将该中断标志位清零。

寄存器 11-4： 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	SSP1IF	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 = 上电复位时的值	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 = 发生了一次读或写操作 （必须由软件清零）
0 = 未发生读写操作

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

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

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

bit 3 **SSP1IF**：主同步串口中断标志位
1 = 发送 / 接收已完成 （必须由软件清零）
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 （必须由软件清零）
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 = 发生了 TMR2 与 PR2 匹配 （必须由软件清零）
0 = 未发生 TMR2 与 PR2 匹配

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

© 2011 Microchip Technology Inc.

寄存器 11-5： PIR2：外设中断请求 （标志）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
OSCFIF	—	SSP2IF	BCL2IF	BCL1IF	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 = 上电复位时的值	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 （必须由软件清零）
0 = 器件时钟正常运行

bit 6 **未实现：**读为 0

bit 5 **SSP2IF：**主同步串口中断标志位
1 = 发送 / 接收已完成 （必须由软件清零）
0 = 等待发送 / 接收

bit 4 **BCL2IF：**总线冲突中断标志位
1 = 发生了总线冲突 （必须由软件清零）
0 = 未发生总线冲突

bit 3 **BCL1IF：**总线冲突中断标志位
1 = 发生了总线冲突 （必须由软件清零）
0 = 未发生总线冲突

bit 2 **HLVDIF：**高 / 低电压检测中断标志位
1 = 发生高 / 低电压条件 （必须由软件清零）
0 = 器件电压高于稳压器低电压检测跳变点

bit 1 **TMR3IF：**TMR3 溢出中断标志位
1 = TMR3 寄存器已溢出 （必须由软件清零）
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 （必须由软件清零）
0 = 未发生定时器门控中断

寄存器 11-6： PIR3：外设中断请求 （标志）寄存器 3

R/W-0	U-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
bit 7							bit 0

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

bit 7 **TMR5GIF**：Timer5 门控中断标志位
1 = 发生了定时器门控中断 （必须由软件清零）
0 = 未发生定时器门控中断

bit 6 **未实现**：读为 0

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

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

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

bit 2 **CCP2IF**：ECCP2 中断标志位
捕捉模式：
1 = 发生了 TMR 寄存器捕捉 （必须由软件清零）
0 = 未发生 TMR 寄存器捕捉
比较模式：
1 = 发生了 TMR 寄存器比较匹配 （必须由软件清零）
0 = 未发生 TMR 寄存器比较匹配
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 TMR 寄存器捕捉 （必须由软件清零）
0 = 未发生 TMR 寄存器捕捉
比较模式：
1 = 发生了 TMR 寄存器比较匹配 （必须由软件清零）
0 = 未发生 TMR 寄存器比较匹配
PWM 模式：
在此模式下未使用。

bit 0 **RTCCIF**：RTCC 中断标志位
1 = 发生了 RTCC 中断 （必须由软件清零）
0 = 未发生 RTCC 中断

© 2011 Microchip Technology Inc.

寄存器 11-7： 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	R/W-0	R/W-0	R/W-0	R/W-0
CCP10IF(1)	CCP9IF(1)	CCP8IF	CCP7IF	CCP6IF	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-1 **CCP<10:4>IF**：CCP<10:4> 中断标志位 (1)

捕捉模式：
1 = 发生了 TMR 寄存器捕捉 （必须由软件清零）
0 = 未发生 TMR 寄存器捕捉
比较模式：
1 = 发生了 TMR 寄存器比较匹配 （必须由软件清零）
0 = 未发生 TMR 寄存器比较匹配
PWM 模式：
在 PWM 模式下未使用。

bit 0 **CCP3IF**：ECCP3 中断标志位

捕捉模式：
1 = 发生了 TMR 寄存器捕捉 （必须由软件清零）
0 = 未发生 TMR 寄存器捕捉
比较模式：
1 = 发生了 TMR 寄存器比较匹配 （必须由软件清零）
0 = 未发生 TMR 寄存器比较匹配
PWM 模式：
在 PWM 模式下未使用。

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

寄存器 11-8： 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
TMR7GIF(1)	TMR12IF(1)	TMR10IF(1)	TMR8IF	TMR7IF(1)	TMR6IF	TMR5IF	TMR4IF
bit 7							bit 0

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

bit 7 **TMR7GIF：**TMR7 门控中断标志位 (1)
1 = 发生了 TMR 门控中断 （必须由软件清零）
0 = 未发生 TMR 门控中断

bit 6 **TMR12IF：**TMR12 与 PR12 匹配中断标志位 (1)
1 = 发生了 TMR12 与 PR12 匹配 （必须由软件清零）
0 = 未发生 TMR12 与 PR12 匹配

bit 5 **TMR10IF：**TMR10 与 PR10 匹配中断标志位 (1)
1 = 发生了 TMR10 与 PR10 匹配 （必须由软件清零）
0 = 未发生 TMR10 与 PR10 匹配

bit 4 **TMR8IF：**TMR8 与 PR8 匹配中断标志位
1 = 发生了 TMR8 与 PR8 匹配 （必须由软件清零）
0 = 未发生 TMR8 与 PR8 匹配

bit 3 **TMR7IF：**TMR7 溢出中断标志位 (1)
1 = TMR7 寄存器已溢出 （必须由软件清零）
0 = TMR7 寄存器未溢出

bit 2 **TMR6IF：**TMR6 与 PR6 匹配中断标志位
1 = 发生了 TMR6 与 PR6 匹配 （必须由软件清零）
0 = 未发生 TMR6 与 PR6 匹配

bit 1 **TMR5IF：**TMR5 溢出中断标志位
1 = TMR5 寄存器已溢出 （必须由软件清零）
0 = TMR5 寄存器未溢出

bit 0 **TMR4IF：**TMR4 与 PR4 匹配中断标志位
1 = 发生了 TMR4 与 PR4 匹配 （必须由软件清零）
0 = 未发生 TMR4 与 PR4 匹配

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

寄存器 11-9： PIR6：外设中断请求 （标志）寄存器 6

U-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	EEIF	—	CMP3IF	CMP2IF	CMP1IF
bit 7							bit 0

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

bit 7-5 **未实现：**读为 0

bit 4 **EEIF：**数据 EEDATA/ 闪存写操作中断标志位
1 = 写操作已完成 （必须由软件清零）
0 = 写操作未完成或还未开始

bit 3 **未实现：**读为 0

bit 2 **CMP3IF：**CMP3 中断标志位
1 = 发生了 CMP3 中断 （必须由软件清零）
0 = 未发生 CMP3 中断

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

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

11.3 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，有六个外设中断允许寄存器（PIE1 至 PIE6）。当 IPEN（RCON<7>）= 0 时，要允许任何外设中断就必须将 PEIE 位置 1。

寄存器 11-10： 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	SSP1IE	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 = 上电复位时的值	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 **SSP1IE**：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

© 2011 Microchip Technology Inc.

寄存器 11-11： PIE2：外设中断允许寄存器 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	—	SSP2IE	BCL2IE	BCL1IE	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 = 上电复位时的值	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 **未实现：**读为 0

bit 5 **SSP2IE：**主同步串口 2 中断允许位
1 = 允许 MSSP 中断
0 = 禁止 MSSP 中断

bit 4 **BCL2IE：**总线冲突中断允许位
1 = 允许总线冲突中断
0 = 禁止总线冲突中断

bit 3 **BCL1IE：**总线冲突中断允许位
1 = 允许
0 = 禁止

bit 2 **HLVDIE：**高 / 低电压检测中断允许位
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 11-12： PIE3：外设中断允许寄存器 3

R/W-0	U-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
bit 7							bit 0

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

bit 7 **TMR5GIE：**Timer5 门控中断允许位
1 = 允许
0 = 禁止

bit 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：**ECCP2 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 **RTCCIE：**RTCC 中断允许位
1 = 允许
0 = 禁止

寄存器 11-13： PIE4：外设中断允许寄存器 4

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CCP10IE(1)	CCP9IE(1)	CCP8IE	CCP7IE	CCP6IE	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-0 **CCP<10:3>IE：**CCP<10:3> 中断允许位 (1)
1 = 允许
0 = 禁止

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

寄存器 11-14： 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
TMR7GIE(1)	TMR12IE(1)	TMR10IE(1)	TMR8IE	TMR7IE(1)	TMR6IE	TMR5IE	TMR4IE
bit 7							bit 0

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

bit 7 **TMR7GIE**：TMR7 门控中断允许位 (1)
1 = 允许
0 = 禁止

bit 6 **TMR12IE**：TMR12 与 PR12 匹配中断允许位 (1)
1 = 允许 TMR12 与 PR12 匹配中断
0 = 禁止 TMR12 与 PR12 匹配中断

bit 5 **TMR10IE**：TMR10 与 PR10 匹配中断允许位 (1)
1 = 允许 TMR10 与 PR10 匹配中断
0 = 禁止 TMR10 与 PR10 匹配中断

bit 4 **TMR8IE**：TMR8 与 PR8 匹配中断允许位
1 = 允许 TMR8 与 PR8 匹配中断
0 = 禁止 TMR8 与 PR8 匹配中断

bit 3 **TMR7IE**：TMR7 溢出中断允许位 (1)
1 = 允许 TMR7 溢出中断
0 = 禁止 TMR7 溢出中断

bit 2 **TMR6IE**：TMR6 与 PR6 匹配中断允许位
1 = 允许 TMR6 与 PR6 匹配中断
0 = 禁止 TMR6 与 PR6 匹配中断

bit 1 **TMR5IE**：TMR5 溢出中断允许位
1 = 允许 TMR5 溢出中断
0 = 禁止 TMR5 溢出中断

bit 0 **TMR4IE**：TMR4 与 PR4 匹配中断允许位
1 = 允许 TMR4 与 PR4 匹配中断
0 = 禁止 TMR4 与 PR4 匹配中断

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

寄存器 11-15： PIE6：外设中断允许寄存器 6

U-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	EEIE	—	CMP3IE	CMP2IE	CMP1IE
bit 7							bit 0

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

bit 7-5 **未实现：**读为 0

bit 4 **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 3 **未实现：**读为 0

bit 2 **CMP3IE：**CMP3 中断允许位
1 = 允许中断
0 = 禁止中断

bit 1 **CMP2E：**CMP2 中断允许位
1 = 允许中断
0 = 禁止中断

bit 0 **CMP1IE：**CMP1 中断允许位
1 = 允许中断
0 = 禁止中断

© 2011 Microchip Technology Inc.

11.4 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，有六个外设中断优先级寄存器 （IPR1 至 IPR6）。使用优先级位要求将中断优先级使能（IPEN）位 （RCON<7>）置 1。

寄存器 11-16： 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	SSP1IP	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 = 上电复位时的值	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 **SSP1IP**：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 11-17： IPR2：外设中断优先级寄存器 2

R/W-1	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
OSCFIP	—	SSP2IP	BCL2IP	BCL1IP	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 = 上电复位时的值	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 **未实现：**读为 0

bit 5 **SSP2IP：**主同步串口 2 中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **BCL2IP：**总线冲突中断优先级位 （MSSP）
1 = 高优先级
0 = 低优先级

bit 3 **BCL1IP：**总线冲突中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **HLVDIP：**高 / 低电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

© 2011 Microchip Technology Inc.

寄存器 11-18： IPR3：外设中断优先级寄存器 3

R/W-1	U-0	R-1	R-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
bit 7							bit 0

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

bit 7 **TMR5GIP：**Timer5 门控中断优先级位
1 = 高优先级
0 = 低优先级

bit 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：**ECCP2 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 **RTCCIP：**RTCC 中断优先级位
1 = 高优先级
0 = 低优先级

寄存器 11-19： IPR4：外设中断优先级寄存器 4

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
CCP10IP(1)	CCP9IP(1)	CCP8IP	CCP7IP	CCP6IP	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-0 **CCP<10:3>IP：**CCP<10:3> 中断优先级位 (1)
1 = 高优先级
0 = 低优先级

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

寄存器 11-20： 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
TMR7GIP(1)	TMR12IP(1)	TMR10IP(1)	TMR8IP	TMR7IP(1)	TMR6IP	TMR5IP	TMR4IP
bit 7							bit 0

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

bit 7 **TMR7GIP**：TMR7 门中断优先级位 (1)
1 = 高优先级
0 = 低优先级

bit 6 **TMR12IP**：TMR12 与 PR12 匹配中断优先级位 (1)
1 = 高优先级
0 = 低优先级

bit 5 **TMR10IP**：TMR10 与 PR10 匹配中断优先级位 (1)
1 = 高优先级
0 = 低优先级

bit 4 **TMR8IP**：TMR8 与 PR8 匹配中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **TMR7IP**：TMR7 溢出中断优先级位 (1)
1 = 高优先级
0 = 低优先级

bit 2 **TMR6IP**：TMR6 与 PR6 匹配中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **TMR5IP**：TMR5 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **TMR4IP**：TMR4 与 PR4 匹配中断优先级位
1 = 高优先级
0 = 低优先级

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

寄存器 11-21： IPR6：外设中断优先级寄存器 6

U-0	U-0	U-0	R/W-1	U-0	R/W-1	R/W-1	R/W-1
—	—	—	EEIP	—	CMP3IP	CMP2IP	CMP1IP
bit 7							bit 0

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

bit 7-5 **未实现：**读为 0

bit 4 **EEIP：**EE 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 **CMP3IP：**CMP3 中断优先级位
1 = 高优先级
0 = 低优先级

bit 1 **CMP2IP：**CMP2 中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **CMP1IP：**CMP1 中断优先级位
1 = 高优先级
0 = 低优先级

11.5 RCON 寄存器

RCON寄存器中包含的位用来确定器件上次复位或从空闲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位 （IPEN）。

寄存器 11-22： 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	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

图注：			
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	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
欲知位操作的详细信息，请参见寄存器 5-1。

bit 5 **$\overline{\text{CM}}$**：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 （随后必须由软件置 1）

bit 4 **$\overline{\text{RI}}$**：RESET 指令标志位
欲知位操作的详细信息，请参见寄存器 5-1。

bit 3 **$\overline{\text{TO}}$**：看门狗定时器超时标志位
欲知位操作的详细信息，请参见寄存器 5-1。

bit 2 **$\overline{\text{PD}}$**：掉电检测标志位
欲知位操作的详细信息，请参见寄存器 5-1。

bit 1 **$\overline{\text{POR}}$**：上电复位状态位
欲知位操作的详细信息，请参见寄存器 5-1。

bit 0 **$\overline{\text{BOR}}$**：欠压复位状态位
欲知位操作的详细信息，请参见寄存器 5-1。

© 2011 Microchip Technology Inc.

11.6 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），则为上升沿触发；如果该位被清零，则为下降沿触发。

当 RBx/INTx 引脚上出现一个有效边沿时，相应的标志位 INTxF 被置 1。通过清零相应的允许位 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 INTxIF 位清零。

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））均能把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>）中的值决定。

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。

11.7 TMR0 中断

在 8 位模式下 （默认模式下）， TMR0寄存器的溢出（FFh → 00h）将会使标志位 TMR0IF 置 1。在 16 位模式下， TMR0H:TMR0L 寄存器对的溢出（FFFFh → 0000h）将使 TMR0IF 标志位置 1。

通过将允许位 TMR0IE （INTCON<5>）置 1 或清零，可以允许或禁止该中断。 Timer0 的中断优先级由中断优先级位 TMR0IP （INTCON2<2>）中的值决定。欲进一步了解 Timer0模块的详细信息，请参见**第 13.0 节“Timer0 模块”**。

11.8 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。通过将允许位 RBIE（INTCON<3>）置 1 或清零，可以允许或禁止该中断。

PORTB 电平变化中断的优先级由中断优先级位 RBIP （INTCON2<0>）中的值决定。

11.9 中断的现场保护

在中断期间， PC 返回的地址被保存在堆栈中。另外，WREG、 STATUS 以及 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 11-1 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 11-1： 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 11-1： 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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIR2	OSCFIF	—	SSP2IF	BCL2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIR4	CCP10IF(1)	CCP9IF(1)	CCP8IF	CCP7IF	CCP6IF	CCP5IF	CCP4IF	CCP3IF
PIR5	TMR7GIF(1)	TMR12IF(1)	TMR10IF(1)	TMR8IF	TMR7IF(1)	TMR6IF	TMR5IF	TMR4IF
PIR6	—	—	—	EEIF	—	CMP3IF	CMP2IF	CMP1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
PIE2	OSCFIE	—	SSP2IE	BCL2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
PIE4	CCP10IE(1)	CCP9IE(1)	CCP8IE	CCP7IE	CCP6IE	CCP5IE	CCP4IE	CCP3IE
PIE5	TMR7GIE(1)	TMR12IE(1)	TMR10IE(1)	TMR8IE	TMR7IE(1)	TMR6IE	TMR5IE	TMR4IE
PIE6	—	—	—	EEIE	—	CMP3IE	CMP2IE	CMP1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
IPR2	OSCFIP	—	SSP2IP	BCL2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
IPR4	CCP10IP(1)	CCP9IP(1)	CCP8IP	CCP7IP	CCP6IP	CCP5IP	CCP4IP	CCP3IP
IPR5	TMR7GIP(1)	TMR12IP(1)	TMR10IP(1)	TMR8IP	TMR7IP(1)	TMR6IP	TMR5IP	TMR4IP
IPR6	—	—	—	EEIP	—	CMP3IP	CMP2IP	CMP1IP
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 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

12.0 I/O 端口

根据选定的器件和使能的功能不同，最多有 9 个端口可供使用。 I/O 端口的一些引脚与器件上外设备用功能复用。通常，当外设使能时，其对应的引脚就无法作为通用 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 （输出锁存器）

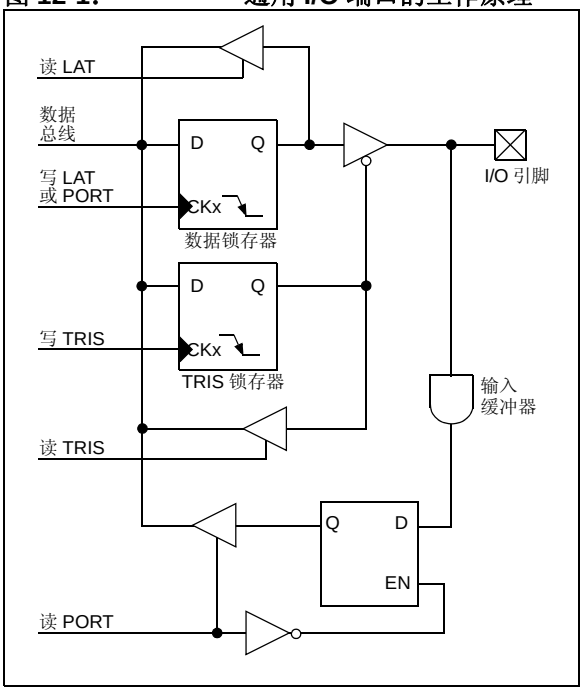
读 PORT 寄存器读入的是引脚的当前状态，而写 PORT 寄存器则是将数据写入输出锁存 （LAT）寄存器。

将 TRIS 位置 1（= 1）可以将对应的端口引脚配置为输入引脚 （即，将对应的输出驱动器置于高阻态） 。将 TRIS 位清零（= 0）可以将对应的端口引脚配置为输出引脚 （即，将对应 LAT 位的内容从选定引脚输出）。

在对 I/O 引脚驱动值进行读 - 修改 - 写操作时会用到输出锁存器 （LAT 寄存器）。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 12-1 给出了通用 I/O 端口的简化模型，图中没有显示到其他外设的接口。

图 12-1： 通用 I/O 端口的工作原理



12.1 I/O 端口引脚的驱动能力

在开发应用时，必须考虑端口引脚的驱动能力。某些引脚上的输出驱动能力要高于其他引脚。同样，某些引脚可以接受高于 VDD 的输入电平。

所有数字端口均可承受 5.5V 的输入电压。模拟端口具有相同的电压承受能力，且已在内部实现了钳位二极管。

12.1.1 引脚输出驱动能力

作为数字 I/O 使用时，各引脚组的输出引脚驱动能力是不同的，这样可以满足不同应用的需求。通常，输出引脚按驱动能力分为两类：

- 输出旨在驱动电流较高的负载，如 LED：
  - PORTA - PORTB
  - PORTC
- 输出驱动电平较低，但可驱动具有高输入阻抗的正常数字电路负载。也可驱动 LED，但只适用于那些电流需求较小的 LED：
  - PORTD - PORTE
  - PORTF - PORTG
  - PORTH(†) - PORTJ(†)

  † 这些端口在 64 引脚器件上不存在。

12.1.2 上拉配置

4 个 I/O 端口（PORTB、PORTD、PORTE 和 PORTJ）在所有引脚上实现了可配置的弱上拉电路。内部上拉电路可使悬空的数字输入信号在不使用外部电阻的情况下被拉至一个稳定电平。

用单个位使能以下各端口的上拉电路 ：对于 PORTB，为 $\overline{\text{RBPU}}$ （INTCON2<7>）位，对于其他端口，为 RDPU、 REPU 和 RJPU （PADCFG1<7:5>）位。

寄存器 12-1： PADCFG1：焊盘配置寄存器

R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	U-0
RDPU	REPU	RJPU(2)	—	—	RTSECSEL1(1)	RTSECSEL0(1)	—
bit 7							bit 0

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

bit 7 **RDPU**：PORTD 上拉使能位
1 = PORTD 上拉电阻通过各端口锁存值使能
0 = 禁止所有 PORTD 上拉电阻

bit 6 **REPU**：PORTE 上拉使能位
1 = PORTE 上拉电阻通过各端口锁存值使能
0 = 禁止所有 PORTE 上拉电阻

bit 5 **RJPU**：PORTJ 上拉使能位 (2)
1 = PORTJ 上拉电阻通过各端口锁存值使能
0 = 禁止所有 PORTJ 上拉电阻

bit 4-3 未实现：读为 0

bit 2-1 **RTSECSEL<1:0>**：RTCC 秒时钟输出选择位 (1)
11 = 保留；不使用
10 = 为 RTCC 引脚选择 RTCC 时钟源 （引脚可以是 LF-INTOSC 或 SOSC，具体取决于 RTCOSC （CONFIG3L<1>）位的设置）
01 = 为 RTCC 引脚选择了 RTCC 秒时钟
00 = 为 RTCC 引脚选择了 RTCC 闹钟脉冲

bit 0 未实现：读为 0

注 **1**： 要使能实际 RTCC 输出，必须将 RTCOE （RTCCFG<2>）位置 1。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

© 2011 Microchip Technology Inc.

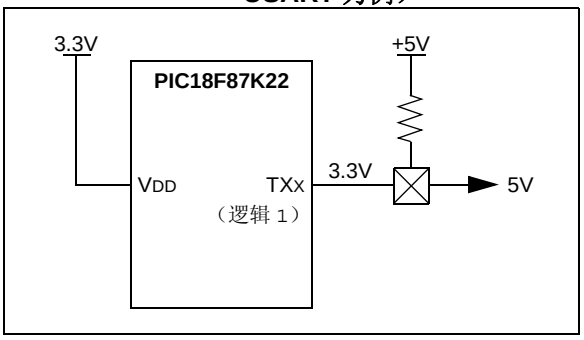
12.1.3 漏极开路输出

某些外设的输出引脚还配备了可配置的漏极开路输出选项。这使得外设可与工作在较高电压下的外部数字逻辑进行通信，而无需电平转换器。

可在某些端口引脚（尤其是与 USART、MSSP 模块（SPI 模式下），以及 CCP 模块的数据和时钟输出相关的端口引脚）上实现漏极开路选项。将寄存器 ODCON1、ODCON2 和 ODCON3 中的漏极开路控制位置 1，可有选择地使能此选项。

需要漏极开路选项时，输出引脚还必须通过由用户提供的外部上拉电阻连接到最高为 5V 的高电平（图 12-2）。当输出数字逻辑高电平信号时，它被拉高至此高电平。

图 12-2： 使用漏极开路输出（以 USART 为例）



寄存器 12-2： ODCON1：外设漏极开路控制寄存器 1

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD
bit 7							bit 0

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

bit 7 **SSP1OD：**MSSP1 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 **CCP2OD：**ECCP2 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 **CCP1OD：**ECCP1 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 4-1 **未实现：**读为 0

bit 0 **SSP2OD：**MSSP2 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

寄存器 12-3： ODCON2：外设漏极开路控制寄存器 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
CCP10OD[1]	CCP9OD[1]	CCP8OD	CCP7OD	CCP6OD	CCP5OD	CCP4OD	CCP3OD
bit 7							bit 0

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

bit 7 **CCP10OD：**CCP10 漏极开路输出使能位 [1]
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 6 **CCP9OD：**CCP9 漏极开路输出使能位 [1]
1 = 使能漏极开路功能
0 = 禁止漏极开路功能

bit 5 **CCP8OD：**CCP8 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 **CCP7OD：**CCP7 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 **CCP6OD：**CCP6 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 **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 1 **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 0 **CCP3OD：**ECCP3 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

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

寄存器 12-4： ODCON3：外设漏极开路控制寄存器 3

R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0	R/W-0
U2OD	U1OD	—	—	—	—	—	CTMUDS
bit 7							bit 0

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

bit 7 **U2OD**：EUSART2 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 **U1OD**：EUSART1 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 5-1 **未实现**：读为 0

bit 0 **CTMUDS**：CTMU 脉冲延时使能位
1 = 已在引脚 RF1 上使能 CTMU 脉冲延时输入
0 = 已在引脚 RF1 上禁止 CTMU 脉冲延时输入

12.1.4 模拟和数字端口

许多端口将模拟与数字功能复用，为硬件设计者提供了更大的灵活性。 PIC18F87K22 系列器件可根据应用的需要将任意模拟引脚配置为模拟或数字引脚。端口的模拟 / 数字功能由以下寄存器控制：ANCON0、ANCON1 和 ANCON2。

这些寄存器置 1 可将相应的引脚配置为模拟引脚，而清零这些寄存器会将端口配置为数字端口。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”**。

12.2 PORTA、 TRISA 和 LATA 寄存器

PORTA 是 8 位宽的双向端口。相应的数据方向寄存器和输出锁存器分别为 TRISA 和 LATA。

RA4/T0CKI 引脚为施密特触发器输入。PORTA 的所有其他引脚都具有 TTL 输入电平和完整的 CMOS 输出驱动器。

RA5 和 RA<3:0> 与 A/D 转换器的模拟输入复用。

通过将ADCON1寄存器中的ANSEL控制位清零或置1，来选择将模拟输入作为 A/D 转换器的输入通道。即使在 PORTA 引脚被用作模拟输入时，相应的 TRISA 位仍控制 PORTA 引脚的方向。在将它们用作模拟输入时，用户必须确保 TRISA 寄存器中相应的位保持为置 1 状态。

> **注：** 在任何复位时，RA5 和 RA<3:0> 均被配置为模拟输入且读为 0。 RA4 被配置为数字输入。

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）的外部电路连接端。在这些情况下，RA6 和 RA7 不可用作数字 I/O 并且它们对应的 TRIS 和 LAT 位读为 0。当器件配置为使用 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，并且振荡器和时钟输入或输出功能被禁止。

RA5 具有可用于 Timer1 和 Timer3 的其他功能。它可配置为 Timer1 时钟输入或 Timer3 外部时钟门控输入。

例 12-1： 初始化 PORTA

```
CLRF     PORTA    ;  Initialize PORTA by
                  ; clearing output latches
CLRF     LATA     ; Alternate method to
                  ; clear output data latches
BANKSEL  ANCON1   ; Select bank with ANCON1 register
MOVLW    00h      ; Configure A/D
MOVWF    ANCON1   ; for digital inputs
BANKSEL  TRISA    ; Select bank with TRISA register
MOVLW    0BFh     ; Value used to initialize
                  ; data direction
MOVWF    TRISA    ; Set RA<7, 5:0> as inputs,
                  ; RA<6> as output
```


© 2011 Microchip Technology Inc.

表 12-1： PORTA 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RA0/AN0/ULPWU	RA0	0	O	DIG	LATA<0> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<0> 数据输入；使能模拟输入时被禁止。
	AN0	1	I	ANA	A/D 输入通道 0。 POR 时默认配置为输入；不影响数字输出。
	ULPWU	1	I	ANA	超低功耗唤醒输入。
RA1/AN1	RA1	0	O	DIG	LATA<1> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<1> 数据输入；使能模拟输入时被禁止。
	AN1	1	I	ANA	A/D 输入通道 1。 POR 时默认配置为输入；不影响数字输出。
RA2/AN2/VREF-	RA2	0	O	DIG	LATA<2> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<2> 数据输入；使能模拟功能时被禁止。
	AN2	1	I	ANA	A/D 输入通道 2； POR 时默认配置为输入。
	VREF-	1	I	ANA	A/D 和比较器低参考电压输入。
RA3/AN3/VREF+	RA3	0	O	DIG	LATA<3> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<3> 数据输入；使能模拟输入时被禁止。
	AN3	1	I	ANA	A/D 输入通道 3； POR 时默认配置为输入。
	VREF+	1	I	ANA	A/D 和比较器高参考电压输入。
RA4/T0CKI	RA4	0	O	DIG	LATA<4> 数据输出。
		1	I	ST	PORTA<4> 数据输入。 POR 时的默认配置。
	T0CKI	x	I	ST	Timer0 时钟输入。
RA5/AN4/T1CKI/ T3G/HLVDIN	RA5	0	O	DIG	LATA<5> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<5> 数据输入；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。
	T1CKI	x	I	ST	Timer1 时钟输入。
	T3G	x	I	ST	Timer3 外部时钟门控输入。
	HLVDIN	1	I	ANA	高 / 低电压检测 （HLVD）外部跳变点输入。
OSC2/CLKO/RA6	OSC2	x	O	ANA	主振荡器反馈输出连接 （XT、 HS 和 LP 模式）。
	CLKO	x	O	DIG	系统周期时钟输出 （FOSC/4、 EC 和 INTOSC 模式）。
	RA6	0	O	DIG	LATA<6> 数据输出； FOSC2 配置位置 1 时被禁止。
		1	I	TTL	PORTA<6> 数据输出； FOSC2 配置位置 1 时被禁止。
OSC1/CLKI/RA7	OSC1	x	I	ANA	主振荡器输入连接 （HS、 XT 和 LP 模式）。
	CLKI	x	I	ANA	主外部时钟源输入 （EC 模式）。
	RA7	0	O	DIG	LATA<7> 数据输出； FOSC2 配置位置 1 时被禁止。
		1	I	TTL	PORTA<7> 数据输出； FOSC2 配置位置 1 时被禁止。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

表 12-2： 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	RA4	RA3	RA2	RA1	RA0
LATA	LATA7(1)	LATA6(1)	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	TRISA4	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： 根据所选择的振荡器模式使能这些位。若未将这些位使能为 PORTA 引脚，则它们将被禁止且读为 x。

12.3 PORTB、TRISB 和 LATB 寄存器

PORTB 是八位宽的双向端口。相应的数据方向寄存器和输出锁存器分别为 TRISB 和 LATB。PORTB 上的所有引脚都只能作为数字引脚。

例 12-2： 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 的每个引脚都有内部弱上拉电路。单个控制位可以接通所有上拉电路。通过将 $\overline{\text{RBPU}}$（INTCON2<7>）位清零可实现此功能。当 PORTB 端口的引脚配置为输出时，其弱上拉电路会自动切断。此弱上拉电路在上电复位时被禁止。

4 个 PORTB 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，才具有此中断功能。当 RB<7:4> 中的任何一个引脚被配置为输出时，该引脚不再具有电平变化中断功能。

当前 RB<7:4> 输入引脚上的电平与 PORTB 上次读入锁存器的旧值进行比较。RB<7:4> 输出的“不匹配”值作逻辑或运算，以将标志位 RBIF （INTCON<0>）置 1，并产生 RB 端口电平变化中断。

该中断可将器件从功耗管理模式中唤醒。在中断服务程序中，用户可通过以下方式清除该中断：

a) 读或写 PORTB （MOVFF （ANY）， PORTB 指令除外）。这将结束不匹配条件。
b) 等待一个指令周期（例如执行一条 NOP 指令）。
c) 清零标志位 RBIF。

电平不匹配条件将会持续地将 RBIF 标志位置 1。而读 PORTB 将结束不匹配条件，并且允许在一个 TCY 延时后将 RBIF 标志位清零。

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> 引脚可复用为 CTMU 边沿输入。 RB5 具有可用于 Timer3 和 Timer1 的其他功能。它可配置为 Timer3 时钟输入或 Timer1 外部时钟门控输入。

表 12-3： PORTB 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RB0/INT0/FLT0	RB0	0	O	DIG	LATB<0> 数据输出。
		1	I	TTL	PORTB<0> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	INT0	1	I	ST	外部中断 0 输入。
	FLT0	x	I	ST	ECCPx 模块的增强型 PWM 故障输入。
RB1/INT1	RB1	0	O	DIG	LATB<1> 数据输出。
		1	I	TTL	PORTB<1> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	INT1	1	I	ST	外部中断 1 输入。
RB2/INT2/CTED1	RB2	0	O	DIG	LATB<2> 数据输出。
		1	I	TTL	PORTB<2> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	INT2	1	I	ST	外部中断 2 输入。
	CTED1	x	I	ST	CTMU 边沿 1 输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

注 1： 当 CCP2MX 配置位清零并处于扩展单片机模式时，对 ECCP2 进行其他分配。

© 2011 Microchip Technology Inc.

表 12-3： PORTB 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RB3/INT3/CTED2/ECCP2/P2A	RB3	0	O	DIG	LATB<3> 数据输出。
		1	I	TTL	PORTB<3> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	INT3	1	I	ST	外部中断 3 输入。
	CTED2	x	I	ST	CTMU 边沿 2 输入。
	ECCP2(1)	0	O	DIG	ECCP2 比较输出和 ECCP2 PWM 输出。 优先级高于端口数据。
		1	I	ST	ECCP2 捕捉输入。
	P2A	0	O	DIG	ECCP2 增强型 PWM 输出，通道 A。 可在增强型 PWM 关闭期间被配置为三态。优先级高于端口数据。
RB4/KBI0	RB4	0	O	DIG	LATB<4> 数据输出。
		1	I	TTL	PORTB<4> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	KBI0	1	I	TTL	引脚电平变化中断。
RB5/KBI1/T3CKI/T1G	RB5	0	O	DIG	LATB<5> 数据输出。
		1	I	TTL	PORTB<5> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	KBI1	1	I	TTL	引脚电平变化中断。
	T3CKI	x	I	ST	Timer3 时钟输入。
	T1G	x	I	ST	Timer1 外部时钟门控输入。
RB6/KBI2/PGC	RB6	0	O	DIG	LATB<6> 数据输出。
		1	I	TTL	PORTB<6> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	KBI2	1	I	TTL	引脚电平变化中断。
	PGC	x	I	ST	供 ICSP 和 ICD 操作使用的串行执行 （ICSP™）时钟输入。
RB7/KBI3/PGD	RB7	0	O	DIG	LATB<7> 数据输出。
		1	I	TTL	PORTB<7> 数据输入；当 $\overline{\text{RBPU}}$ 位清零时启用弱上拉。
	KBI3	1	I	TTL	引脚电平变化中断。
	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。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

注 1： 当 CCP2MX 配置位清零并处于扩展单片机模式时，对 ECCP2 进行其他分配。

表 12-4： 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
ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD

图注： PORTB 不使用阴影单元。

12.4 PORTC、 TRISC 和 LATC 寄存器

PORTC是八位宽的双向端口。相应的数据方向寄存器和输出锁存器分别为 TRISC 和 LATC。 RC2 到 RC7 引脚只能作为数字引脚。

PORTC 与 ECCP、 MSSP 和 EUSART 外设功能复用（表 12-5）。这些引脚配有施密特触发器输入缓冲器。只要 ECCP、SPI 和 EUSART 功能有效，则相关的引脚还可配置为漏极开路输出。通过将 ODCON1 和 ODCON3 寄存器中的 SPIOD、 CCPxOD 和 U1OD 控制位置 1 可选择漏极开路配置。

RC1 通常被配置为 ECCP2 模块的默认外设引脚。配置位 CCP2MX（默认状态， CCP2MX = 1）控制 ECCP2 的分配。

当使能外设功能时，应小心定义每个 PORTC 引脚的 TRIS 位。某些外设会改写 TRIS 位的设置而将引脚定义为输出或输入。用户应参考相应的外设章节来正确设置 TRIS 位。

注：	在任何器件复位时，这些引脚都被配置为数字输入。

外设对引脚的改写会影响TRISC寄存器的内容。尽管某个外设可能会改写一个或多个引脚，但读TRISC总是会返回其当前内容。

例 12-3： 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 12-5： 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	I	ST	SOSC 振荡器输出。
	SCLKI	1	I	ST	数字时钟输入；当 SOSC 振荡器禁止时使能。
RC1/SOSCI/ ECCP2/P2A	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。
	ECCP2(1)	0	O	DIG	ECCP2 比较和 ECCP2 PWM 输出；优先级高于端口数据。
		1	I	ST	ECCP2 捕捉输入。
	P2A	0	O	DIG	ECCP2 增强型 PWM 输出，通道 A。 可在增强型 PWM 关闭期间被配置为三态；优先级高于端口数据。
RC2/ECCP1/ P1A	RC2	0	O	DIG	LATC<2> 数据输出。
		1	I	ST	PORTC<2> 数据输入。
	ECCP1	0	O	DIG	ECCP1 比较和 ECCP1 PWM 输出；优先级高于端口数据。
		1	I	ST	ECCP1 捕捉输入。
	P1A	0	O	DIG	ECCP1 增强型 PWM 输出，通道 A。 可在增强型 PWM 关闭期间被配置为三态；优先级高于端口数据。

图注： O = 输出，= 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入， TTL = TTL 缓冲器输入， I^2C = I^2C™/SMBus 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。

© 2011 Microchip Technology Inc.

表 12-5： PORTC 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RC3/SCK1/SCL1	RC3	0	O	DIG	LATC<3> 数据输出。
		1	I	ST	PORTC<3> 数据输入。
	SCK1	0	O	DIG	SPI 时钟输出 （MSSP 模块）；优先级高于端口数据。
		1	I	ST	SPI 时钟输入 （MSSP 模块）。
	SCL1	0	O	DIG	I^2C 时钟输出 （MSSP 模块）；优先级高于端口数据。
		1	I	I^2C	I^2C 时钟输入 （MSSP 模块）；输入类型取决于模块设置。
RC4/SDI1/SDA1	RC4	0	O	DIG	LATC<4> 数据输出。
		1	I	ST	PORTC<4> 数据输入。
	SDI1		I	ST	SPI 数据输入 （MSSP 模块）。
	SDA1	1	O	DIG	I^2C 数据输出 （MSSP 模块）；优先级高于端口数据。
		1	I	I^2C	I^2C 数据输入 （MSSP 模块）；输入类型取决于模块设置。
RC5/SDO1	RC5	0	O	DIG	LATC<5> 数据输出。
		1	I	ST	PORTC<5> 数据输入。
	SDO1	0	O	DIG	SPI 数据输出 （MSSP 模块）。
RC6/TX1/CK1	RC6	0	O	DIG	LATC<6> 数据输出。
		1	I	ST	PORTC<6> 数据输入。
	TX1	1	O	DIG	同步串行数据输出 （EUSART 模块）；优先级高于端口数据。
	CK1	1	O	DIG	同步串行数据输入 （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）。
RC7/RX1/DT1	RC7	0	O	DIG	LATC<7> 数据输出。
		1	I	ST	PORTC<7> 数据输入。
	RX1	1	I	ST	异步串行接收数据输入 （EUSART 模块）。
	DT1	1	O	DIG	同步串行数据输出 （EUSART 模块）；优先级高于端口数据。
		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。

图注： O = 输出，= 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入， TTL = TTL 缓冲器输入，
I^2C = I^2C™/SMBus 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2 进行默认分配。

表 12-6： 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
ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS

图注： PORTC 不使用阴影单元。

12.5 PORTD、 TRISD 和 LATD 寄存器

PORTD是八位宽的双向端口。相应的数据方向寄存器和输出锁存器分别为 TRISD 和 LATD。

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地配置为输入或输出。

> **注：** 在任何器件复位时，这些引脚都被配置为数字输入。

PORTD 的每个引脚都有内部弱上拉电路。单个控制位可以关闭所有上拉电路。这可通过将 RDPU（PADCFG1<7>）位置 1 实现。当 PORTD 端口的引脚配置为输出时，其弱上拉电路会自动切断。在所有器件复位时上拉功能被禁止。

在 80 引脚器件上，PORTD 与系统总线复用，作为外部存储器接口的一部分。仅当通过将 EBDIS 位（MEMCON<7>）置 1 禁止接口时， I/O 端口和其他功能才可用。使能接口时， PORTD 是复用的地址 / 数据总线的低字节 （AD<7:0>）。 TRISD 位也将被忽略。

将控制位 PSPMODE（TRISE<4>）置 1 也可将 PORTD 配置为 8 位宽的微处理器端口 （并行从端口）。在该模式下，输入缓冲器为 TTL。关于更多信息，请参见第 12.11 节 “并行从端口”。

PORTD 的 RD4、 RD5 和 RD6 引脚还具有 I^2C 和 SPI 功能。用于 SPI 功能的引脚也可配置为漏极开路输出。通过将 SSP2OD （ODCON1<0>）置 1 来选择漏极开路配置。

RD0 具有 CTMU 功能。RD1 具有用于 Timer5 时钟输入和 Timer7 外部时钟门控输入的功能。

例 12-4： 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 12-7： PORTD 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RD0/PSP0/ AD0/CTPLS	RD0	0	O	DIG	LATD<0> 数据输出。
		1	I	ST	PORTD<0> 数据输入。
	PSP0(1)	x	I/O	TTL	并行从端口数据。
	AD0(2)	x	I/O	TTL	外部存储器地址 / 数据 0。
	CTPLS	x	O	DIG	CTMU 脉冲发生器输出。
RD1/PSP1/ AD1/T5CKI/ T7G	RD1	0	O	DIG	LATD<1> 数据输出。
		1	I	ST	PORTD<1> 数据输入。
	PSP1(1)	x	I/O	TTL	并行从端口数据。
	AD1(2)	x	I/O	TTL	外部存储器地址 / 数据 1。
	T5CKI	x	I	ST	Timer5 时钟输入。
	T7G	x	I	ST	Timer7 外部时钟门控输入。
RD2/PSP2/AD2	RD2	0	O	DIG	LATD<2> 数据输出。
		1	I	ST	PORTD<2> 数据输入。
	PSP2(1)	x	I/O	TTL	并行从端口数据。
	AD2(2)	x	I/O	TTL	外部存储器地址 / 数据 2。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
I^2C = I^2C™/SMB 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

注 1： 并行从端口 （Parallel Slave Port， PSP）仅可用于单片机模式。

2： 此功能只在 PIC18F8XK22 器件上可用。

© 2011 Microchip Technology Inc.

表 12-7： PORTD 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RD3/PSP3/AD3	RD3	0	O	DIG	LATD<3> 数据输出。
		1	I	ST	PORTD<3> 数据输入。
	PSP3(1)	x	I/O	TTL	并行从端口数据。
	AD3(2)	x	I/O	TTL	外部存储器地址 / 数据 3。
RD4/PSP4/ AD4/SDO2	RD4	0	O	DIG	LATD<4> 数据输出。
		1	I	ST	PORTD<4> 数据输入。
	PSP4(1)	x	I/O	TTL	并行从端口数据。
	AD4(2)	x	I/O	TTL	外部存储器地址 / 数据 4。
	SDO2	0	P	DOG	SPI 数据输出 （MSSP 模块）。
RD5/PSP5/ AD5/SDI2/ SDA2	RD5	0	O	DIG	LATD<5> 数据输出。
		1	I	ST	PORTD<5> 数据输入。
	PSP5(1)	x	I/O	TTL	并行从端口数据。
	AD5(2)	x	I/O	TTL	外部存储器地址 / 数据 5。
	SDI2	1	I	ST	SPI 数据输入 （MSSP 模块）。
	SDA2	0	O	I^2C	I^2C 数据输入 （MSSP 模块）。输入类型取决于模块设置。
RD6/PSP6/ AD6/SCK2/ SCL2	RD6	0	O	DIG	LATD<6> 数据输出。
		1	I	ST	PORTD<6> 数据输入。
	PSP6(1)	x	I/O	TTL	并行从端口数据。
	AD6(2)	x	I/O	TTL	外部存储器地址 / 数据 6。
	SCK2	0	O	DIG	SPI 时钟输出 （MSSP 模块）；优先级高于端口数据。
		1	I	ST	SPI 时钟输入 （MSSP 模块）。
	SCL2	0	O	DIG	I^2C 时钟输出 （MSSP 模块）；优先级高于端口数据。
		1	I	I^2C	I^2C 时钟输入 （MSSP 模块）。输入类型取决于模块设置。
RD7/PSP7/ AD7/$\overline{SS2}$	RD7	0	O	DIG	LATD<7> 数据输出。
		1	I	ST	PORTD<7> 数据输入。
	PSP7(1)	x	I/O	TTL	并行从端口数据。
	AD7(2)	x	I/O	TTL	外部存储器地址 / 数据 7。
	$\overline{SS2}$	1	I	TTL	MSSP 模块的从选择输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
I^2C = I^2C™/SMB 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

注 1： 并行从端口 （Parallel Slave Port， PSP）仅可用于单片机模式。

2： 此功能只在 PIC18F8XK22 器件上可用。

表 12-8： 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	REPU	RJPU(1)	—	—	RTSECSEL1	RESECSEL0	—
ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD

图注： PORTD 不使用阴影单元。

注 1： 在 PIC18F6XK22 器件上未实现，读为 0。

12.6 PORTE、TRISE 和 LATE 寄存器

PORTE是八位宽的双向端口。相应的数据方向寄存器和输出锁存器分别为 TRISE 和 LATE。

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地设置为输入或输出。当 RE7 引脚上的 ECCP2 有效时，该引脚也可配置为漏极开路输出。通过将 CCP2OD 控制位 （ODCON1<6>）置 1 来选择漏极开路配置。

> **注：** 在任何器件复位时，这些引脚都被配置为数字输入。

PORTE 的每个引脚都有内部弱上拉电路。单个控制位可以关闭所有上拉电路。这可通过将 REPU（PADCFG1<6>）位置 1 实现。当 PORTE 端口的引脚配置为输出时，其弱上拉电路会自动切断。在任何器件复位时上拉功能被禁止。

PORTE 也与增强型 PWM 输出复用，对于 ECCP1 和 ECCP3 为 B 和 C，对于 ECCP2 为 B、C 和 D。对于所有器件，它们默认分配在 PORTE<6:0>。

在 80 引脚器件上， ECCPMX 配置位控制 ECCP1 和 ECCP3 输出复用。清零该位重新将 P1B/P1C 和 P3B/P3C 输出分配给 PORTH。

对于工作在单片机模式下的器件，RE7 引脚可被配置为 ECCP2 模块和增强型 PWM 输出 2A 的备用外设引脚。这可通过清零 CCP2MX 配置位实现。PORTE 还可与并行从端口地址信号线复用。 RE1 和 RE0 可与控制信号（$\overline{WR}$ 和 $\overline{RD}$）复用。

RE3 也可配置为来自系统时钟的参考时钟输出（Reference Clock Output， REFO）。欲知更多详情，请参见第 3.7 节 “参考时钟输出”。

例 12-5： 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 12-9： PORTE 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE0/$\overline{RD}$/P2D AD8	RE0	0	O	DIG	LATE<0> 数据输出。
		1	I	ST	PORTE<0> 数据输入。
	$\overline{RD}$	x	O	DIG	并行从端口读选通引脚。
		x	I	TTL	并行从端口读引脚。
	P2D	0	O	—	ECCP2 PWM 输出 D。 可在增强型 PWM 关闭期间被配置为三态。
	AD8(2)	x	O	DIG	外部存储器接口，数据 Bit 8 输出。
		x	I	TTL	外部存储器接口，数据 Bit 8 输入。
RE1/P2C/$\overline{WR}$/ AD9	RE1	0	O	DIG	LATE<1> 数据输出。
		1	I	ST	PORTE<1> 数据输入。
	P2C	0	O	—	ECCP2 PWM 输出 C。 可在增强型 PWM 关闭期间被配置为三态。
	$\overline{WR}$	x	O	DIG	并行从端口写选通引脚。
		x	I	TTL	并行从端口写引脚。
	AD9(2)	x	O	DIG	外部存储器接口，数据 Bit 9 输出。
		x	I	TTL	外部存储器接口，数据 Bit 9 输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

注 1： 当 CCP2MX 配置位清零并处于单片机模式时，对 ECCP2 进行其他分配。
2： 此功能只在 PIC18F8XKXX 器件上可用。

© 2011 Microchip Technology Inc.

表 12-9： PORTE 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE2/$\overline{CS}$/P2B/CCP10/AD10	RE2	0	O	DIG	LATE<2> 数据输出。
		1	I	ST	PORTE<2> 数据输入。
	$\overline{CS}$	x	I	TTL	并行从端口片选。
	P2B	0	O	—	ECCP2 PWM 输出 B。 可在增强型 PWM 关闭期间被配置为三态。
	CCP10	1	I/O	ST	捕捉 10 输入 / 比较 10 输出 /PWM10 输出。
	AD10(2)	x	O	DIG	外部存储器接口，地址 / 数据 Bit 10 输出。
		x	I	TTL	外部存储器接口，数据 Bit 10 输入。
RE3/P3C/CCP9/REFO/AD11	RE3	0	O	DIG	LATE<3> 数据输出。
		1	I	ST	PORTE<3> 数据输入。
	P3C	0	O	—	ECCP3 PWM 输出 C。 可在增强型 PWM 关闭期间被配置为三态。
	CCP9	0	O	DIG	CCP9 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP9 捕捉输入。
	REFO	x	O	DIG	参考时钟输出。
	AD11(2)	x	O	DIG	外部存储器接口，地址 / 数据 Bit 11 输出。
		x	I	TTL	外部存储器接口，数据 Bit 11 输入。
RE4/P3B/CCP8/AD12	RE4	0	O	DIG	LATE<4> 数据输出。
		1	I	ST	PORTE<4> 数据输入。
	P3B	0	O	—	ECCP3 PWM 输出 B。 可能在增强型 PWM 发生关闭事件期间被配置为三态。
	CCP8	0	O	DIG	CCP8 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP8 捕捉输入。
	AD12(2)	x	O	DIG	外部存储器接口，地址 / 数据 Bit 12 输出。
		x	I	TTL	外部存储器接口，数据 Bit 12 输入。
RE5/P1C/CCP7/AD13	RE5	0	O	DIG	LATE<5> 数据输出。
		1	I	ST	PORTE<5> 数据输入。
	P1C	0	O	—	ECCP1 PWM 输出 C。 可在增强型 PWM 关闭期间被配置为三态。
	CCP7	0	O	DIG	CCP7 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP7 捕捉输入。
	AD13(2)	x	O	DIG	外部存储器接口，地址 / 数据 Bit 13 输出。
		x	I	TTL	外部存储器接口，数据 Bit 13 输入。
RE6/P1B/CCP6/AD14	RE6	0	O	DIG	LATE<6> 数据输出。
		1	I	ST	PORTE<6> 数据输入。
	P1B	0	O	—	ECCP1 PWM 输出 B。 可在增强型 PWM 关闭期间被配置为三态。
	CCP6	0	O	DIG	CCP6 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP6 捕捉输入。
	AD14(2)	x	O	DIG	外部存储器接口，地址 / 数据 Bit 14 输出。
		x	I	TTL	外部存储器接口，数据 Bit 14 输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

注 1： 当 CCP2MX 配置位清零并处于单片机模式时，对 ECCP2 进行其他分配。

2： 此功能只在 PIC18F8XKXX 器件上可用。

表 12-9： PORTE 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE7/ECCP2/ P2A/AD15	RE7	0	O	DIG	LATE<7> 数据输出。
		1	I	ST	PORTE<7> 数据输入。
	ECCP2(1)	0	O	DIG	ECCP2 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	ECCP2 捕捉输入。
	P2A	0	O	—	ECCP2 PWM 输出 A。 可在增强型 PWM 关闭期间被配置为三态。
	AD15(2)	x	O	DIG	外部存储器接口，地址 / 数据位 Bit 15 输出。
		x	I	TTL	外部存储器接口，数据 Bit 15 输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

注 1： 当 CCP2MX 配置位清零并处于单片机模式时，对 ECCP2 进行其他分配。
2： 此功能只在 PIC18F8XKXX 器件上可用。

表 12-10： 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	RE6	RE5	RE4	RE3	RE2	RE1	RE0
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0
PADCFG1	RDPU	REPU	RJPU(2)	—	—	RTSECSEL1	RTSECSEL0	—
ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD
ODCON2	CCP10OD(1)	CCP9OD(1)	CCP8OD	CCP7OD	CCP6OD	CCP5OD	CCP4OD	CCP3OD

图注： PORTE 不使用阴影单元。

注 1： 在 PIC18FX5K22 器件上未实现，读为 0。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

© 2011 Microchip Technology Inc.

12.7 PORTF、 LATF 和 TRISF 寄存器

PORTF 是 7 位宽的双向端口。相应的数据方向寄存器和输出锁存器分别为 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地配置为输入或输出。

通过在 CMCON 寄存器中设置相应的位还可以将 RF1 到 RF6 引脚用作比较器输入或输出。要将 RF<7:1> 用作数字输入，也必须关闭比较器。

> 注 1：当器件复位时，引脚 RF<7:1> 都配置成模拟输入，并且读为 0。
>
> 2：要将 PORTF 配置为数字 I/O，应关闭比较器并清零 ADCON1 和 ANCON2。

例 12-6： 初始化 PORTF

```
CLRF      PORTF     ; Initialize PORTF by
                    ; clearing output
                    ; data latches
CLRF      LATF      ; Alternate method
                    ; to clear output
                    ; data latches
BANKSEL   ANCON1    ; Select bank with ANCON1 register
MOVLW     1Fh       ; Make AN6, AN7 and AN5 digital
MOVWF     ANCON1    ;
MOVLW     0Fh       ; Make AN8, AN9, AN10 and AN11
                      digital
MOVWF     ANCON     ; Set PORTF as digital I/O
BANKSEL   TRISF     ; Select bank with TRISF register
MOVLW     0CEh      ; Value used to
                    ; initialize data
                    ; direction
MOVWF     TRISF     ; Set RF3:RF1 as inputs
                    ; RF5:RF4 as outputs
                    ; RF7:RF6 as inputs
```

表 12-11： PORTF 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RF1/AN6/C2OUT/CTDIN	RF1	0	O	DIG	LATF<1> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<1> 数据输入；使能模拟输入时被禁止。
	AN6	1	I	ANA	A/D 输入通道 6。发生 POR 时为默认配置。
	C2OUT	0	O	DIG	比较器 2 输出；优先级高于端口数据。
	CTDIN	1	I	ST	CTMU 脉冲延时输入。
RF2/AN7/C1OUT	RF2	0	O	DIG	LATF<2> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<2> 数据输入；使能模拟输入时被禁止。
	AN7	1	I	ANA	A/D 输入通道 7。发生 POR 时为默认配置。
	C1OUT	0	O	DIG	比较器 1 输出；优先级高于端口数据。
RF3/AN8/C2INB/CTMUI	RF3	0	O	DIG	LATF<3> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<3> 数据输入；使能模拟输入时被禁止。
	AN8	1	I	ANA	A/D 输入通道 8 和比较器 C2+ 输入。 POR 时默认配置为输入；不受模拟输出的影响。
	C2INB	1	I	ANA	比较器 2 输入 B。
	CTMUI	x	O	—	C2INB 比较器输入的 CTMU 脉冲发生器充电器。
RF4/AN9/C2INA	RF4	0	O	DIG	LATF<4> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<4> 数据输入；使能模拟输入时被禁止。
	AN9	1	I	ANA	A/D 输入通道 9 和比较器 C2- 输入。 POR 时默认配置为输入；不影响数字输出。
	C2INA	1	I	ANA	比较器 2 输入 A。
RF5/AN10/CVREF/C1INB	RF5	0	O	DIG	LATF<5> 数据输出；不受模拟输入影响。当使能 CVREF 输出时被禁止。
		1	I	ST	PORTF<5> 数据输入；使能模拟输入时被禁止。当使能 CVREF 输出时被禁止。
	AN10	1	I	ANA	A/D 输入通道 10 和比较器 C1+ 输入。 POR 时默认配置为输入。
	CVREF	x	O	ANA	比较器参考电压输出。使能此功能将禁止数字 I/O。
	C1INB	1	I	ANA	比较器 1 输入 B。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

表 12-11： PORTF 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RF6/AN11/C1INA	RF6	0	O	DIG	LATF<6> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<6> 数据输入；使能模拟输入时被禁止。
	AN11	1	I	ANA	A/D 输入通道 11 和比较器 C1- 输入。 POR 时默认配置为输入；不影响数字输出。
	C1INA	1	I	ANA	比较器 1 输入 A。
RF7/AN5/$\overline{SS1}$	RF7	0	O	DIG	LATF<7> 数据输出；不受模拟输入影响。
		1	I	ST	PORTF<7> 数据输入；使能模拟输入时被禁止。
	AN5	1	I	ANA	A/D 输入通道 5。 POR 时的默认配置。
	$\overline{SS1}$	1	I	TTL	MSSP 模块的从选择输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
TTL = TTL 缓冲器输入， x = 任意值 （TRIS 位不影响端口方向或在此可被忽略）。

表 12-12： 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	—
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
ANCON1	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8

图注： — = 未实现，读为 0。 PORTF 不使用阴影单元。

© 2011 Microchip Technology Inc.

12.8 PORTG、 TRISG 和 LATG 寄存器

PORTG 是 5 位宽的双向端口。相应的数据方向寄存器和输出锁存器分别为 TRISG 和 LATG。

PORTG 与 AUSART 和 CCP、ECCP、模拟、比较器、RTCC 和定时器输入功能复用（表 12-13）。当作为 I/O 进行操作时，所有 PORTG 引脚都有施密特触发器输入缓冲器。可使用 ODCONx 配置 CCPx 和 UART 的漏极开路功能。

当使能外设功能时，应小心定义每个 PORTG 引脚的 TRIS 位。某些外设会改写 TRIS 位的设置而将引脚定义为输出，而另一些外设也会改写 TRIS 位的设置，但将引脚定义为输入。用户应参考相应的外设章节来正确设置 TRIS 位。改写引脚的值不会被装入 TRIS 寄存器。这将允许对 TRIS 寄存器执行读 - 修改 - 写操作而无需担心外设改写其内容。

例 12-7： 初始化 PORTG

```
CLRF    PORTG       ;  Initialize PORTG by
                    ; clearing output
                    ; data latches
BCF     CM1CON, CON ; disable
                    ; comparator 1
CLRF    LATG        ; Alternate method
                    ; to clear output
                    ; data latches
BANKSEL ANCON2      ; Select bank with ACON2 register
MOVLW   0F0h        ; make AN16 to AN19
                    ; digital
MOVWF   ANCON2
BANKSEL TRISG       ; Select bank with TRISG register
MOVLW   04h         ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISG       ; Set RG1:RG0 as
                    ; outputs
                    ; RG2 as input
                    ; RG4:RG3 as inputs
```

表 12-13： PORTG 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RG0/ECCP3/ P3A	RG0	0	O	DIG	LATG<0> 数据输出。
		1	I	ST	PORTG<0> 数据输入。
	ECCP3	0	O	DIG	ECCP3 比较和 ECCP3 PWM 输出；优先级高于端口数据。
		1	I	ST	ECCP3 捕捉输入。
	P3A	0	O	—	ECCP3 PWM 输出 A。 可在增强型 PWM 关闭期间被配置为三态。
RG1/TX2/CK2/ AN19/C3OUT	RG1	0	O	DIG	LATG<1> 数据输出。
		1	I	ST	PORTG<1> 数据输入。
	TX2	1	O	DIG	同步串行数据输出 （EUSART 模块）；优先级高于端口数据。
	CK2	1	O	DIG	同步串行数据输入 （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）。
	AN19	1	I	ANA	A/D 输入通道 19。 POR 时默认配置为输入。不影响数字输入。
	C3OUT	x	O	DIG	比较器 3 输出。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

表 12-13： PORTG 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RG2/RX2/DT2/AN18/C3INA	RG2	0	O	DIG	LATG<2> 数据输出。
		1	I	ST	PORTG<2> 数据输入。
	RX2	1	I	ST	异步串行接收数据输入 （EUSART 模块）。
	DT2	1	O	DIG	同步串行数据输出 （EUSART 模块）；优先级高于端口数据。
		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。
	AN18	1	I	ANA	A/D 输入通道 18。 POR 时默认配置为输入；不影响数字输出。
	C3INA	x	I	ANA	比较器 3 输入 A。
RG3/CCP4/AN17/P3D/C3INB	RG3	0	O	DIG	LATG<3> 数据输出。
		1	I	ST	PORTG<3> 数据输入。
	CCP4	0	O	DIG	CCP4 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP4 捕捉输入。
	AN17	1	I	ANA	A/D 输入通道 17。 POR 时默认配置为输入；不影响数字输出。
	P3D	0	O	—	ECCP3 PWM 输出 D。 可在增强型 PWM 期间被配置为三态。
	C3INB	x	I	ANA	比较器 3 输入 B。
RG4/RTCC/T7CKI/T5G/CCP5/AN16/P1D/C3INC	RG4	0	O	DIG	LATG<4> 数据输出。
		1	I	ST	PORTG<4> 数据输入。
	RTCC	x	O	DIG	RTCC 输出。
	T7CKI	x	I	ST	Timer7 时钟输入。
	T5G	x	I	ST	Timer5 外部时钟门控输入。
	CCP5	0	O	DIG	CCP5 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP5 捕捉输入。
	AN16	1	I	ANA	A/D 输入通道 17。 POR 时默认配置为输入；不影响数字输出。
	P1D	0	O	—	ECCP1 PWM 输出 D。 可在增强型 PWM 期间被配置为三态。
	C3INC	x	I	ANA	比较器 3 输入 C。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

表 12-14： 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	—	—	RG5(2)	RG4	RG3	RG2	RG1	RG0
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0
ANCON2	ANSEL23	ANSEL22	ANSEL21	ANSEL20	ANSEL19	ANSEL18	ANSEL17	ANSEL16
ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD
ODCON2	CCP10OD(1)	CCP9OD(1)	CCP8OD	CCP7OD	CCP6OD	CCP5OD	CCP4OD	CCP3OD
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS

图注： — = 未实现，读为 0。 PORTG 不使用阴影单元。

注 1： 在 PIC18FX5K22 器件上未实现，读为 0。

2： 当禁止主复位时 （MCLRE = 0），该位可用。当 MCLRE 被置 1 时，该位未实现。

© 2011 Microchip Technology Inc.

12.9 PORTH、 LATH 和 TRISH 寄存器

注： PORTH 只在 80 引脚器件上存在。

PORTH 是 8 位宽的双向 I/O 端口。相应的数据方向寄存器和输出锁存器分别为 TRISH 和 LATH。

PORTH 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地配置为输入或输出。

例 12-8： 初始化 PORTH

```
CLRF     PORTH    ; Initialize PORTH by
                  ; clearing output
                  ; data latches
CLRF     LATH     ; Alternate method
                  ; to clear output
                  ; data latches
BANKSEL  ANCON2   ; Select bank with ANCON2 register
MOVLW    0Fh      ; Configure PORTH as
MOVWF    ANCON2   ; digital I/O
MOVLW    0Fh      ; Configure PORTH as
MOVWF    ANCON1   ; digital I/O
BANKSEL  TRISH    ; Select bank with TRISH register
MOVLW    0CFh     ; Value used to
                  ; initialize data
                  ; direction
MOVWF    TRISH    ; Set RH3:RH0 as inputs
                  ; RH5:RH4 as outputs
                  ; RH7:RH6 as inputs
```

表 12-15： PORTH 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RH0/AN23/A16	RH0	0	O	DIG	LATH<0> 数据输出。
		1	I	ST	PORTH<0> 数据输入。
	AN23	1	I	ANA	A/D 输入通道 23。 POR 时默认配置为输入；不影响数字输入。
	A16	x	O	DIG	外部存储器接口、地址信号线 16；优先级高于端口数据。
RH1/AN22/A17	RH1	0	O	DIG	LATH<1> 数据输出。
		1	I	ST	PORTH<1> 数据输入。
	AN22	1	I	ANA	A/D 输入通道 22。 POR 时默认配置为输入；不影响数字输入。
	A17	x	O	DIG	外部存储器接口、地址信号线 17；优先级高于端口数据。
RH2/AN21/A18	RH2	0	O	DIG	LATH<2> 数据输出。
		1	I	ST	PORTH<2> 数据输入。
	AN21	1	I	ANA	A/D 输入通道 21。 POR 时默认配置为输入；不影响数字输入。
	A18	x	O	DIG	外部存储器接口、地址信号线 18；优先级高于端口数据。
RH3/AN20/A19	RH3	0	O	DIG	LATH<3> 数据输出。
		1	I	ST	PORTH<3> 数据输入。
	AN20	1	I	ANA	A/D 输入通道 20。 POR 时默认配置为输入；不影响数字输入。
	A19	x	O	DIG	外部存储器接口、地址信号线 19；优先级高于端口数据。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

表 12-15： PORTH 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RH4/CCP9/ P3C/AN12/ C2INC	RH4	0	O	DIG	LATH<4> 数据输出。
		1	I	ST	PORTH<4> 数据输入。
	CCP9	0	O	DIG	CCP9 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP9 捕捉输入。
	P3C	0	O	—	ECCP3 PWM 输出 C。 可在增强型 PWM 期间被配置为三态。
	AN12	1	I	ANA	A/D 输入通道 12。 POR 时默认配置为输入；不影响数字输入。
	C2INC	x	I	ANA	比较器 2 输入 C。
RH5/CCP8/ P3B/AN13/ C2IND	RH5	0	O	DIG	LATH<5> 数据输出。
		1	I	ST	PORTH<5> 数据输入。
	CCP8	0	O	DIG	CCP8 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP8 捕捉输入。
	P3B	0	O	—	ECCP3 PWM 输出 B。 可在增强型 PWM 期间被配置为三态。
	AN13	1	I	ANA	A/D 输入通道 13。 POR 时默认配置为输入；不影响数字输入。
	C2IND	x	I	ANA	比较器 2 输入 D。
RH6/CCP7/ P1C/AN14/ C1INC	RH6	0	O	DIG	LATH<6> 数据输出。
		1	I	ST	PORTH<6> 数据输入。
	CCP7	0	O	DIG	CCP7 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP7 捕捉输入。
	P1C	0	O	—	ECCP1 PWM 输出 C。 可在增强型 PWM 期间被配置为三态。
	AN14	1	I	ANA	A/D 输入通道 14。 POR 时默认配置为输入；不影响数字输入。
	C1INC	x	I	ANA	比较器 1 输入 C。
RH7/CCP6/ P1B/AN15	RH7	0	O	DIG	LATH<7> 数据输出。
		1	I	ST	PORTH<7> 数据输入。
	CCP6	0	O	DIG	CCP6 比较 /PWM 输出；优先级高于端口数据。
		1	I	ST	CCP6 捕捉输入。
	P1B	0	O	—	ECCP1 PWM 输出 B。 可在增强型 PWM 期间被配置为三态。
	AN15	1	I	ANA	A/D 输入通道 15。 POR 时默认配置为输入；不影响数字输入。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

© 2011 Microchip Technology Inc.

表 12-16： 与 PORTH 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
PORTH(1)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0
LATH(1)	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0
TRISH(1)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0
ANCON1	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
ANCON2	ANSEL23	ANSEL22	ANSEL21	ANSEL20	ANSEL19	ANSEL18	ANSEL17	ANSEL16
ODCON2	CCP10OD(2)	CCP9OD(2)	CCP8OD	CCP7OD	CCP6OD	CCP5OD	CCP4OD	CCP3OD

注 1： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
2： 在 PIC18FX5K22 器件上未实现，读为 0。

12.10 PORTJ、 TRISJ 和 LATJ 寄存器

注： PORTJ 只在 80 引脚器件上可用。

PORTJ 是 8 位宽的双向端口。相应的数据方向寄存器和输出锁存器分别为 TRISJ 和 LATJ。

PORTJ 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地设置为输入或输出。

注： 在任何器件复位时，这些引脚都被配置为数字输入。

当使能外部存储器接口时，所有的 PORTJ 引脚用作接口的控制输出。当通过清零 EBDIS 控制位（MEMCON<7>）使能接口时，将自动实现上述功能。TRISJ 位也将被忽略。

PORTJ 的每个引脚都有内部弱上拉电路。提供上拉电路是为了保持上电时外部存储器接口上的输入电平处于已知的状态。单个控制位可以关闭所有上拉电路。这可通过清零 RJPU （PADCFG1<5>）位实现。当 PORTJ 端口的引脚配置为输出时，其弱上拉电路会自动切断。在任何器件复位时上拉功能被禁止。

例 12-9： 初始化 PORTJ

```
CLRF    PORTJ   ; Initialize PORTJ by
                ; clearing output latches
CLRF    LATJ    ; Alternate method
                ; to clear output latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISJ   ; Set RJ3:RJ0 as inputs
                ; RJ5:RJ4 as output
                ; RJ7:RJ6 as inputs
```

表 12-17： PORTJ 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RJ0/ALE	RJ0	0	O	DIG	LATJ<0> 数据输出。
		1	I	ST	PORTJ<0> 数据输入。
	ALE	x	O	DIG	外部存储器接口地址锁存器使能控制输出；优先级高于数字 I/O。
RJ1/$\overline{OE}$	RJ1	0	O	DIG	LATJ<1> 数据输出。
		1	I	ST	PORTJ<1> 数据输入。
	$\overline{OE}$	x	O	DIG	外部存储器接口输出使能控制输出；优先级高于数字 I/O。
RJ2/$\overline{WRL}$	RJ2	0	O	DIG	LATJ<2> 数据输出。
		1	I	ST	PORTJ<2> 数据输入。
	$\overline{WRL}$	x	O	DIG	外部存储器总线写低字节控制；优先级高于数字 I/O。
RJ3/$\overline{WRH}$	RJ3	0	O	DIG	LATJ<3> 数据输出。
		1	I	ST	PORTJ<3> 数据输入。
	$\overline{WRH}$	x	O	DIG	外部存储器接口写高字节控制；优先级高于数字 I/O。
RJ4/BA0	RJ4	0	O	DIG	LATJ<4> 数据输出。
		1	I	ST	PORTJ<4> 数据输入。
	BA0	x	O	DIG	外部存储器接口字节地址 0 控制输出；优先级高于数字 I/O。
RJ5/$\overline{CE}$	RJ5	0	O	DIG	LATJ<5> 数据输出。
		1	I	ST	PORTJ<5> 数据输入。
	$\overline{CE}$	x	O	DIG	外部存储器接口芯片使能控制输出；优先级高于数字 I/O。
RJ6/$\overline{LB}$	RJ6	0	O	DIG	LATJ<6> 数据输出。
		1	I	ST	PORTJ<6> 数据输入。
	$\overline{LB}$	x	O	DIG	外部存储器接口低字节使能控制输出；优先级高于数字 I/O。
RJ7/$\overline{UB}$	RJ7	0	O	DIG	LATJ<7> 数据输出。
		1	I	ST	PORTJ<7> 数据输入。
	$\overline{UB}$	x	O	DIG	外部存储器接口最高字节使能控制输出；优先级高于数字 I/O。

图注： O = 输出， I = 输入， ANA = 模拟信号， DIG = 数字输出， ST = 施密特触发器缓冲器输入，
x = 任意值 （TRIS 位不影响端口方向或在此可忽略）。

表 12-18： 与 PORTJ 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
PORTJ(1)	RJ7	RJ6	RJ5	RJ4	RJ3	RJ2	RJ1	RJ0
LATJ(1)	LATJ7	LATJ6	LATJ5	LATJ4	LATJ3	LATJ2	LATJ1	LATJ0
TRISJ(1)	TRISJ7	TRISJ6	TRISJ5	TRISJ4	TRISJ3	TRISJ2	TRISJ1	TRISJ0
PADCFG1	RDPU	REPU	RJPU(1)	—	—	RTSECSEL1	RTSECSEL0	—

图注： PORTJ 不使用阴影单元。

注 1： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

© 2011 Microchip Technology Inc.

12.11 并行从端口

当控制位 PSPMODE（PSPCON<4>）置 1 时，PORTD 还可用作 8 位宽并行从端口 （PSP）或微处理器端口。通过 RD 控制输入引脚（RE0/P2D/RD/AD8）和 WR 控制输入引脚（RE1/P2C/WR/AD9），该端口可由外部系统进行异步读写。

注：	并行从端口仅可用于单片机模式。

PSP 可直接与 8 位微处理器数据总线接口。外部微处理器可将 PORTD 锁存器作为八位锁存器进行读写。

将 PSPMODE 位置 1 可将端口引脚 RE0/P2D/RD/AD8 使能为 RD 输入，将 RE1/P2C/WR/AD9 使能为 WR 输入，并将 RE2/P2B/CCP10/CS/AD10 使能为 CS （片选）输入。要实现该功能，相应的 TRISE 寄存器的数据方向位 （TRISE<2:0>）必须配置为输入 （= 111）。

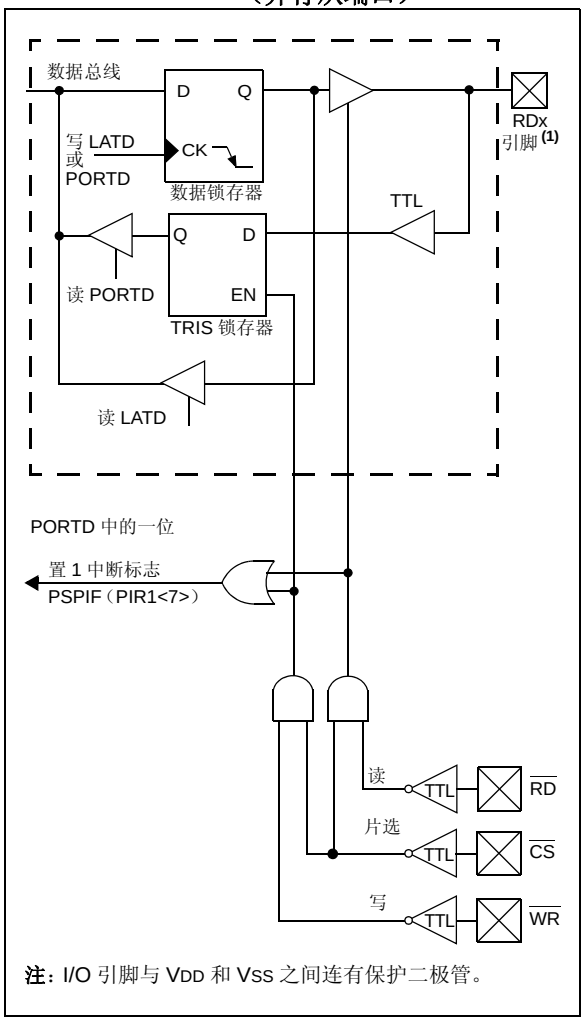
当首次检测到 CS 和 WR 信号线都为低电平时，将对 PSP 进行写操作，并在检测到其中一个为高电平时结束写操作。当写操作结束时，PSPIF 和 IBF 标志位（分别为 PIR1<7> 和 PSPCON<7>）置 1。

当首次检测到 CS 和 RD 信号线都为低电平时，将对 PSP 进行读操作。此时读出 PORTD 中的数据，且 OBF 位（PSPCON<6>）置 1。如果用户将新数据写入 PORTD 来置 1 OBF，数据将被立即读出，但不会置 1 OBF 位。

当检测到 CS 或 RD 信号线为高电平时， PORTD 引脚返回输入状态且 PSPIF 位置 1。用户应用程序应等待 PSPIF 置 1 后再处理 PSP。此时，可查询 IBF 位和 OBF 位并采取相应的操作。

写和读模式下的控制信号时序分别如图 12-4 和图 12-5 所示。

图 12-3： PORTD 和 PORTE 框图（并行从端口）



注：I/O 引脚与 VDD 和 VSS 之间连有保护二极管。

寄存器 12-5： 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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **IBF：**输入缓冲器满状态位
1 = 已接收到一个字，正等待 CPU 读取
0 = 未接收到字

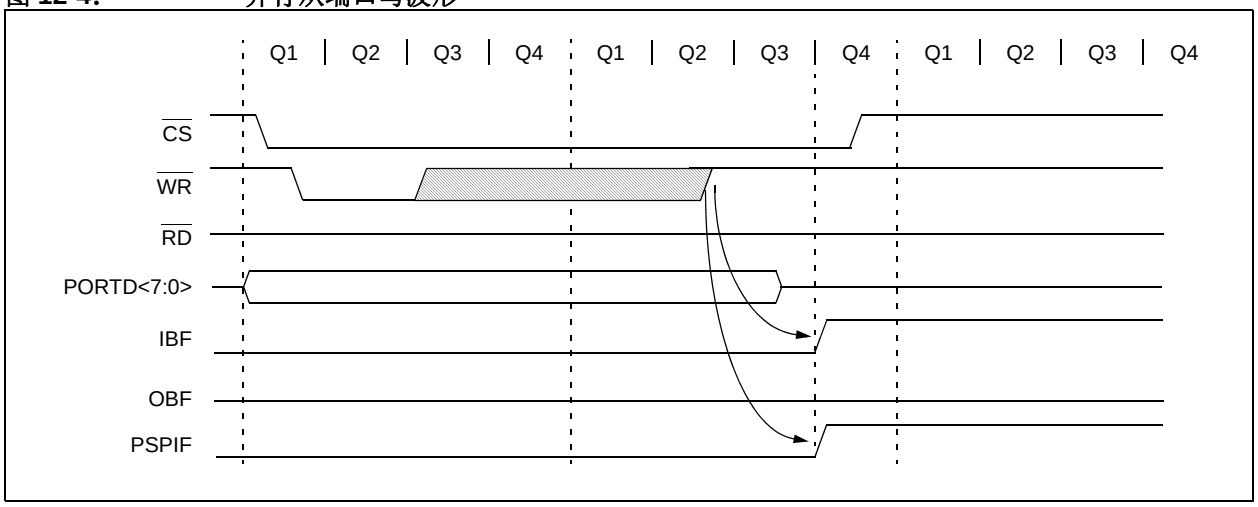
bit 6 **OBF：**输出缓冲器满状态位
1 = 输出缓冲器仍保存着上一次写入的字
0 = 已读取输出缓冲器

bit 5 **IBOV：**输入缓冲器溢出检测位
1 = 在尚未读取上一次输入的字前发生了一次写入（必须由软件清零）
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 12-4： 并行从端口写波形



© 2011 Microchip Technology Inc.

图 12-5： 并行从端口读波形

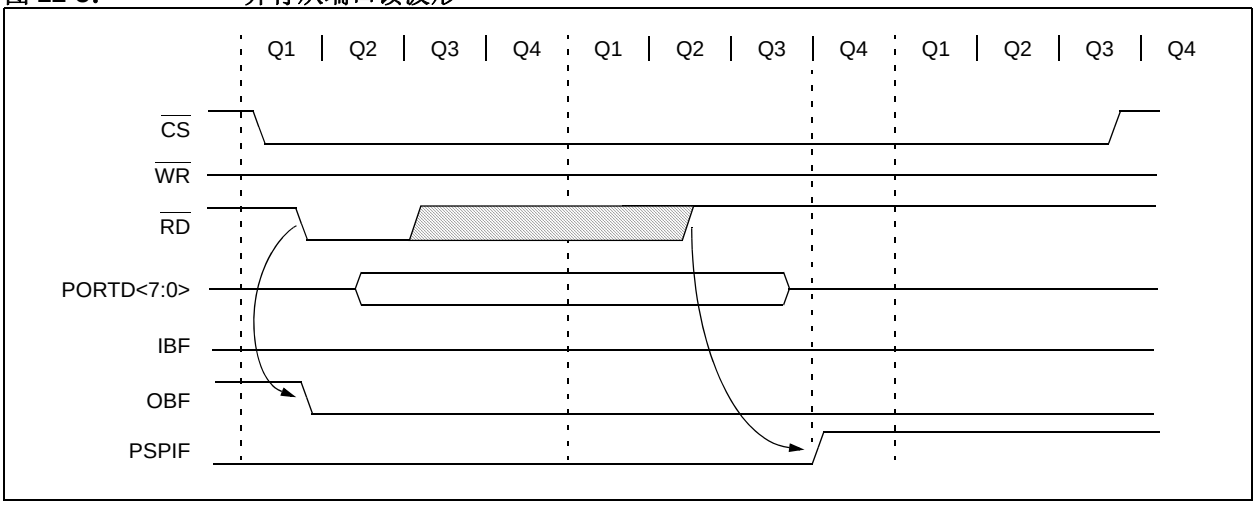


表 12-19： 与并行从端口相关的寄存器

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	LATE3	LATE2	LATE1	LATE0
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PMD1	PSPMD	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBDM

图注： — = 未实现，读为 0。并行从端口不使用阴影单元。

注：

© 2011 Microchip Technology Inc.

13.0 TIMER0 模块

Timer0 模块具有以下特性：

- 可通过软件选择作为 8 位或 16 位定时器 / 计数器
- 可读写的寄存器
- 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
- 溢出中断

T0CON 寄存器（寄存器 13-1）控制模块工作的所有方面，包括预分频比选择。该寄存器是可读写的。

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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **TMR0ON：**Timer0 开关控制位
1 = 使能 Timer0
0 = 禁止 Timer0

bit 6 **T08BIT：**Timer0 8 位 /16 位模式控制位
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 引脚输入电平的跳变沿作为时钟源
0 = 内部时钟 （Fosc/4）

bit 4 **T0SE：**Timer0 时钟源边沿选择位
1 = 在 T0CKI 引脚上电平的下降沿递增
0 = 在 T0CKI 引脚上电平的上升沿递增

bit 3 **PSA：**Timer0 预分频器分配位
1 = 未给 Timer0 分配预分频器； Timer0 时钟输入信号不经过预分频器
0 = 已给 Timer0 分配预分频器； Timer0 时钟输入信号来自预分频器的输出

bit 2-0 **T0PS<2:0>：**Timer0 预分频值选择位
111 = 1:256 预分频值
110 = 1:128 预分频值
101 = 1:64 预分频值
100 = 1:32 预分频值
011 = 1:16 预分频值
010 = 1:8 预分频值
001 = 1:4 预分频值
000 = 1:2 预分频值

13.1 Timer0 工作原理

Timer0 既可用作定时器亦可用作计数器。具体的模式由 T0CS 位 （T0CON<5>）选择。在定时器模式 （T0CS = 0）下，除非选择了不同的预分频值 （见**第 13.3 节 “预分频器”**），否则，默认情况下，在每个时钟周期该模块都会递增。如果写入 TMR0 寄存器，那么在随后的两个指令周期，计数将不再递增。用户可通过将调整值写入 TMR0 寄存器来解决上述问题。

通过将 T0CS 位置 1 （= 1）选择计数器模式。在计数器模式下， Timer0 可在 T0CKI 引脚信号的每个上升沿或下降沿递增。触发递增的边沿由 Timer0 时钟源边沿选择位 T0SE （T0CON<4>）决定。清零此位选择上升沿触发。下面讨论外部时钟输入的限制条件。

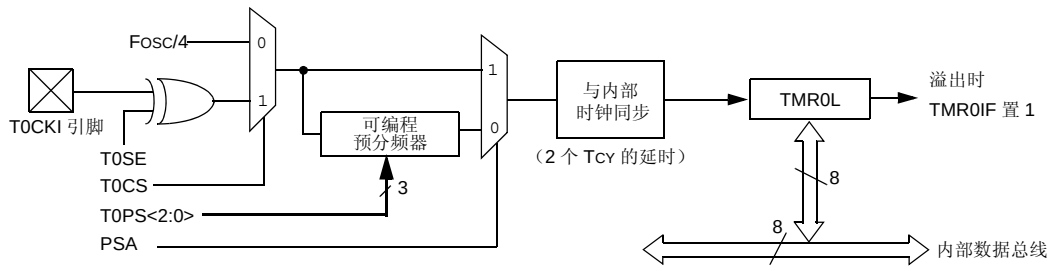
可以使用外部时钟源来驱动 Timer0 ；但是必须满足一定的要求以确保外部时钟与内部相位时钟 （Tosc）同步。在同步之后，仍需要一定的延时才会引发定时器 / 计数器的递增操作。

13.2 Timer0 的 16 位读写模式

TMR0H 不是 16 位模式 Timer0 的高字节。它实际上是 Timer0 高字节的缓冲寄存器，因为 Timer0 的高字节不可以被直接读写 （见图 13-2）。在读 TMR0L 时使用 Timer0 高字节的内容更新 TMR0H。这样可以一次读取 Timer0 的全部 16 位，而无需验证读到的高字节和低字节的有效性。（在高、低字节分两次连续读取的情况下，由于可能存在计满返回，因此需要验证读到字节的有效性）。

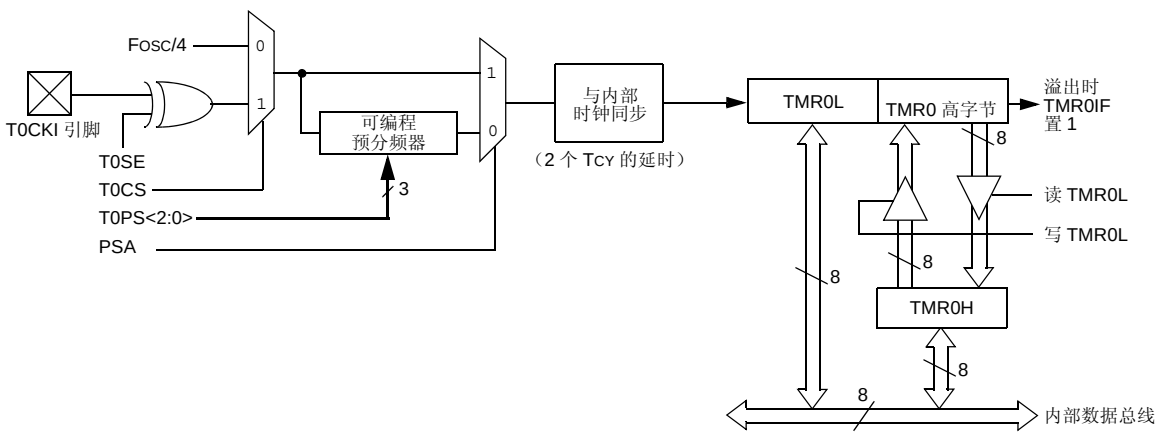
同样，写入 Timer0 的高字节也是通过 TMR0H 缓冲寄存器来操作的。在写入 TMR0L 的同时，使用 TMR0H 的内容更新 Timer0 的高字节。这样一次就可以完成 Timer0 全部 16 位的更新。

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引脚信号的最大预分频信号。

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引脚信号的最大预分频信号。

© 2011 Microchip Technology Inc.

13.3 预分频器

Timer0 模块的预分频器为一个 8 位计数器。此预分频器不可直接读写。其值通过 PSA 和 T0PS<2:0> 位（T0CON<3:0>）设置，这四位决定预分频器的分配和预分频比值。

将 PSA 位清零可将预分频器分配给 Timer0 模块。在分配预分频器时，预分频比值可以在 1:2 到 1:256 之间进行选择，该比值以 2 的整数次幂递增。

若将预分频器分配给 Timer0 模块，所有以 TMR0 寄存器为写入对象的指令（如 `CLRF TMR0`、`MOVWF TMR0` 和 `BSF TMR0` 等）都将使预分频器的计数值清零。

> **注：** 若将预分频器分配给 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

当 TMR0 寄存器发生溢出时（8 位模式下，从 FFh 到 00h；或 16 位模式下，从 FFFFh 到 0000h），将产生 TMR0 中断。这种溢出会将标志位 TMR0IF 置 1。可以通过清零 TMR0IE 位（INTCON<5>）来屏蔽此中断。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

图注： — = 未实现，读为 0。 Timer0 不使用阴影单元。

注：

© 2011 Microchip Technology Inc.

14.0 TIMER1 模块

Timer1 定时器 / 计数器模块具有以下特性：

- 可通过软件选择作为 16 位定时器 / 计数器
- 可读写的 8 位寄存器 （TMR1H 和 TMR1L）
- 可选择使用器件时钟或 SOSC 内部振荡器作为内部或外部时钟源
- 溢出中断
- ECCP 特殊事件触发复位
- 带门控功能的定时器

图 14-1 给出了 Timer1 模块的简化框图。

Timer1 振荡器也可作为单片机处于功耗管理模式下的低功耗时钟源。 Timer1 也可使用 SOSC 振荡器作为时钟源。

Timer1 的工作由 T1CON 控制寄存器 （寄存器 14-1）控制。该寄存器还包含辅助振荡器使能位（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。如果定时器要与捕捉 / 比较功能一起使用，将总是选择其他定时器时钟选项之一。

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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-6 **TMR1CS<1:0>**：Timer1 时钟源选择位
10 = 可以根据 SOSCEN 位选择 Timer1 时钟源是引脚或者振荡器。
SOSCEN = 0：
使用 T1CKI 引脚上的外部时钟 （上升沿）。
SOSCEN = 1：
根据 SOSCSEL 配置位，选择时钟源是 SOSCI/SOSCO 引脚上的晶振还是 SCLKI 引脚的内部时钟。
01 = Timer1 时钟源是系统时钟 （Fosc） (1)
01 = Timer1 时钟源是指令时钟 （Fosc/4）

bit 5-4 **T1CKPS<1:0>**：Timer1 输入时钟预分频比选择位
11 = 1:8 预分频值
10 = 1:4 预分频值
01 = 1:2 预分频值
00 = 1:1 预分频值

bit 3 **SOSCEN**：SOSC 振荡器使能位
1 = 针对 Timer1 使能并使用 SOSC
0 = 针对 Timer1 禁止 SOSC
关闭振荡器的反相器和反馈电阻以降低功耗。

bit 2 **$\overline{\text{T1SYNC}}$**：Timer1 外部时钟输入同步选择位
TMR1CS<1:0> = 10：
1 = 与外部时钟输入不同步
0 = 与外部时钟输入同步
TMR1CS<1:0> = 0x：
忽略此位。当 TMR1CS<1:0>= 1x 时， Timer1 使用内部时钟。

bit 1 **RD16**：16 位读 / 写模式使能位
1 = 使能通过一次 16 位操作对 Timer1 寄存器进行读写
0 = 使能通过两次 8 位操作对 Timer1 寄存器进行读写

bit 0 **TMR1ON**：Timer1 使能位
1 = 使能 Timer1
0 = 停止 Timer1

注 1： 如果定时器将用于 ECCP 捕捉 / 比较功能，将不能选择 Fosc 时钟源。

14.1 Timer1 门控寄存器

Timer1 门控寄存器 （T1GCON）如寄存器 14-2 所示，
用于对 Timer1 进行门控控制。

寄存器 14-2： T1GCON：TIMER1 门控寄存器 (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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **TMR1GE：**Timer1 门控使能位
当 TMR1ON = 0 时：
忽略此位。
当 TMR1ON = 1 时：
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 （当门控信号为高电平时， Timer1 计数）
0 = Timer1 门控低电平有效 （当门控信号为低电平时， 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翻转触发器
Timer1 门控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 门控单脉冲模式并对 Timer1 进行门控控制
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 门控单脉冲采集已就绪，正在等待一个边沿
0 = Timer1 门控单脉冲采集已完成或尚未开始
当 T1GSPM 清零时，该位自动清零。

bit 2 **T1GVAL：**Timer1 门控当前状态位
指示可提供给 TMR1H:TMR1L 的 Timer1 门控的当前状态；不受 Timer1 门控使能位 （TMR1GE）影响。

bit 1-0 **T1GSS<1:0>：**Timer1 门控源选择位
11 = 比较器 2 输出
10 = 比较器 1 输出
01 = TMR2 匹配 PR2 输出
00 = Timer1 门控引脚

注 **1：** 建议在编程 T1CON 之前先编程 T1GCON。

© 2011 Microchip Technology Inc.

14.2 Timer1 工作原理

Timer1 模块是一个 8/16 位递增计数器，可通过 TMR1H:TMR1L 寄存器对访问。

当使用内部时钟源时，Timer1 为定时器，并在每个指令周期递增。当使用外部时钟源时，Timer1 可以是定时器也可以是计数器。它将在外部时钟源的每个所选边沿递增。

通过分别配置T1CON和T1GCON寄存器中的TMR1ON和 TMR1GE 位使能 Timer1。

当 SOSC 选择为晶振模式（通过 SOSCEL）时，RC1/SOSCI/ECCP2/P2A 和 RC0/SOSCO/SCLKI 引脚成为输入引脚。这意味着 TRISC<1:0> 的值被忽略，并且这些引脚的读取值为 0。

14.3 时钟源选择

T1CON 寄存器的 TMR1CS<1:0> 和 SOSCEN 位将用于选择Timer1时钟源。寄存器 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 模块可作为定时器或计数器。

若 Timer1 使能为计数， Timer1 在外部时钟输入 T1CKI 的上升沿递增。这些外部时钟源中的每一个都可以与单片机系统时钟同步，也可以异步运行。

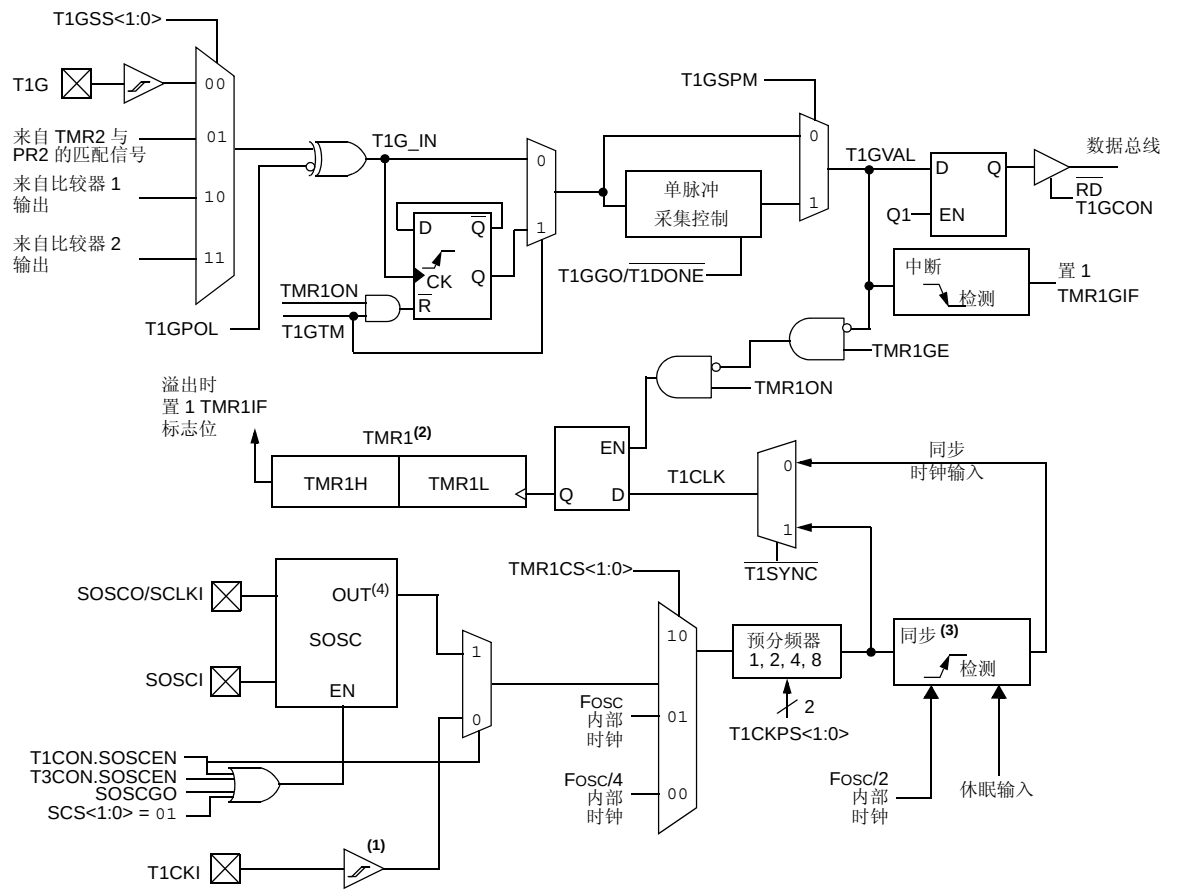
当 Timer1 依靠时钟振荡器作为定时器工作时，可以将外部32.768 kHz 的晶振与专用内部振荡电路一起使用。

> **注：** 计数器模式下，在发生以下任一或几种情况后，计数器必须在第一个递增上升沿之前先记录一个下降沿：
> - 在上电复位后使能 Timer1
> - 写入 TMR1H 或 TMR1L
> - 禁止 Timer1
> - T1CKI 为高电平时禁止 Timer1（TMR1ON= 0）、T1CKI 为低电平时使能 Timer1 （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 寄存器在每个上升沿递增。
3： 处于休眠状态时，无法执行同步。
4： SOSC 的输出由 SOSCSEL<1:0> 配置位决定。

© 2011 Microchip Technology Inc.

14.4 Timer1 16 位读 / 写模式

Timer1 可被配置为 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。对 TMR1L 的读操作将把 Timer1 的高字节内容载入 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计满返回，而不得不验证读取的有效性。

对 Timer1 的高字节进行写操作也必须通过 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 16 位值一次写入 Timer1 的高字节和低字节。

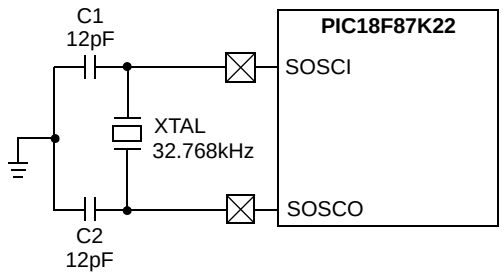
在该模式下不能直接读写 Timer1 的高字节。所有读写都必须通过 Timer1 高字节缓冲寄存器进行。写入 TMR1H 不会清零 Timer1 预分频器。只有在写 TMR1L 时才会清零预分频器。

14.5 SOSC 振荡器

片上晶振电路连接在 SOSCI（输入）和 SOSCO（放大器输出）引脚之间。通过任何请求 SOSC 振荡器的外设可使能该振荡器。可使用以下八种方式使能：如果 SOSC 选择为任何奇数定时器的时钟源，则通过各自的 SOSCEN 位（TxCON<3>）使能；如果 SOSC 选择为 RTCC 时钟源，则通过 RTCOSC 配置位（CONFIG3L<1>）使能；如果 SOSC 选择为 CPU 时钟源，或 SOSCGO 位（OSCCON2<3>）置 1，则通过 SCS 位（OSCCON<1:0>）使能。SOSCGO 位用于预热 SOSC，使之在有任何外设请求前做好准备。该振荡器是采用额定频率为 32 kHz 晶振的低功耗电路。它将可以在所有功耗管理模式下继续运行。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 振荡器正常起振。

图 14-2： SOSC 低功耗振荡器的外部元件



注： 如需了解电容选择的更多信息，请参见表 14-2 下的“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 建议在验证振荡器电路时从这些值开始。

2：选用较大的电容值虽然可以提高振荡器的稳定性，但同时也会延长起振时间。

3：由于谐振器 / 晶振的特性各不相同，因此用户应当向谐振器 / 晶振制造厂商询问外部元件的相应值。

4：上述电容值仅供设计参考。所列的值是当 SOSCSEL= 11 时额定负载电容为 CL = 10 pF 的晶振的典型值。

5：错误的电容值可能导致频率不满足晶振制造厂商的容差规范。

SOSC 晶振驱动电平根据 SOSCSEL<1:0>（CONFIG1L<4:3>）配置位确定。高驱动电平模式（SOSCSEL<1:0> = 11）将驱动多个具有多种负载电容（CL）额定值的 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的。它不适用于驱动所有类型的 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，晶振电路可能无法正常工作。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。该值与图 14-2 中用于 C1 和 C2 的值相关，但不一定相同。

有关为指定晶振选择最佳 C1 和 C2 的详细信息，请参见晶振制造商的应用信息。最佳值部分取决于电路中的寄生电容的数量 （通常未知）。因此，强烈建议在选择电容值后对振荡器进行全面的测试和验证。

14.5.1 使用 SOSC 作为时钟源

功耗管理模式下，SOSC 振荡器也可用作时钟源。通过将系统时钟选择位 SCS<1:0> （OSCCON<1:0>）设置为 01，器件可以切换到 SEC_RUN 模式，此时 CPU 和外设都以 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。欲知更多详情，请参见**第 4.0 节 “功耗管理模式”**。

只要 SOSC 振荡器提供时钟源，SOSC 系统时钟状态标志位 SOSCRUN （OSCCON2<6>）就会置 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 （FSCM）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，轮询SOCSRUN位可以确定时钟源是SOSC 振荡器还是其他时钟源。

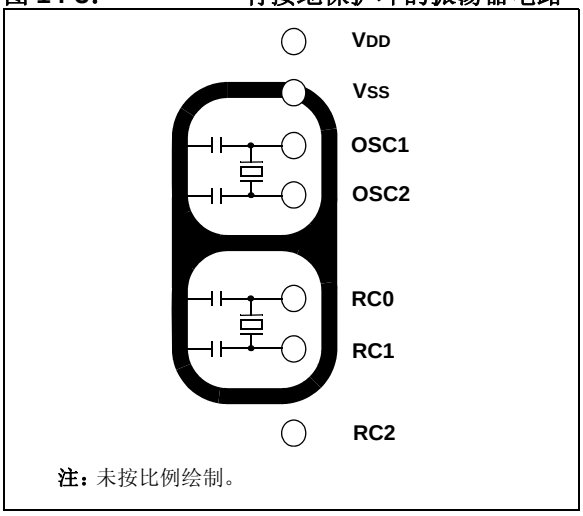
14.5.2 SOSC 振荡器布线注意事项

SOSC 振荡器电路在工作期间的功耗极低。鉴于此振荡器的低能耗特性，它对附近变化较快的信号可能会比较敏感。当振荡器配置为超低功耗模式 SOSCSEL<1:0> （CONFIG1L<4:3>）= 01 时，这一点尤其突出。

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置高速电路，那么在该振荡器电路附近布置接地保护环可能会有所帮助。在单面 PCB 上可使用保护环 （如图 14-3 所示）或外接一个地平面。（如输出比较模式或 PWM 模式下的 ECCP1 引脚或使用 OSC2 引脚的主振荡器等高速电路）

图 14-3： 有接地保护环的振荡器电路



在低驱动电平模式 （SOSCSEL<1:0> = 01）下，有一点十分重要，即将 RC2 I/O 引脚信号与振荡器电路隔离。将RC2配置为数字输出并翻转它的输出信号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。如果可能，可将 RC2 保留为不用，或者将它用作输入引脚，但其信号源的压摆率有一定限制。如果 RC2 必须作为数字输出，则对于许多 PCB 布线设计，可能需要使用高驱动电平振荡器模式（SOSCSEL<1:0> = 11）。

即使在高驱动电平模式下，设计振荡器电路时仍需认真遵循布线流程。

除了要考虑 dV/dt 感应噪声外，确保电路板的清洁也很重要。即使是极少量的导电焊剂残留物也会导致PCB泄漏电流，这可能会对振荡器电路产生极大影响。

14.6 Timer1 中断

TMR1 寄存器对 （TMR1H:TMR1L）从 0000h 开始递增，一直到 FFFFh，然后计满返回到 0000h。如果允许 Timer1 中断，就会在溢出时产生该中断，由中断标志位 TMR1IF（PIR1<0>）表示。可以通过置 1 或清零 Timer1 中断允许位 TMR1IE（PIE1<0>）来允许或禁止该中断。

© 2011 Microchip Technology Inc.

14.7 使用 ECCP 特殊事件触发信号复位 Timer1

如果 ECCP 模块被配置为使用 Timer1，并且在比较模式（CCPxM<3:0> = 1011）下产生特殊事件触发信号，此触发信号将复位 Timer1。如果使能了 A/D 模块，来自 ECCP2的触发信号还将启动A/D转换。（欲知更多信息，请参见**第 20.3.4 节 “特殊事件触发信号”**）。

要使用这一功能，必须将 Timer1 模块配置为定时器或同步计数器。在这种情况下，CCPRxH:CCPRxL 寄存器对实际上变成了 Timer1 的周期寄存器。

如果 Timer1 工作在异步计数器模式下，复位操作将不起作用。

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：** 来自ECCPx模块的特殊事件触发信号将仅清零 TMR1 寄存器的内容，并不会将 TMR1IF 中断标志位（PIR1<0>）置 1。

14.8 Timer1 门控

Timer1 可以被配置为自主计数，或使用 Timer1 门控电路使能或禁止计数。这也称为 Timer1 门控计数使能。

Timer1 门控也可由多个可选时钟源驱动。

14.8.1 TIMER1 门控计数使能

可通过将 T1GCON 寄存器的 TMR1GE 位置 1 使能 Timer1 门控使能模式。使用 T1GPOL 位（T1GCON<6>）配置 Timer1 门控使能模式的极性。

当使能 Timer1 门控使能模式时，Timer1 将在 Timer1 时钟源的上升沿递增。当禁止 Timer1 门控使能模式时，不进行递增计数，且 Timer1 保持当前计数。欲知更多时序信息，请参见图 14-4。

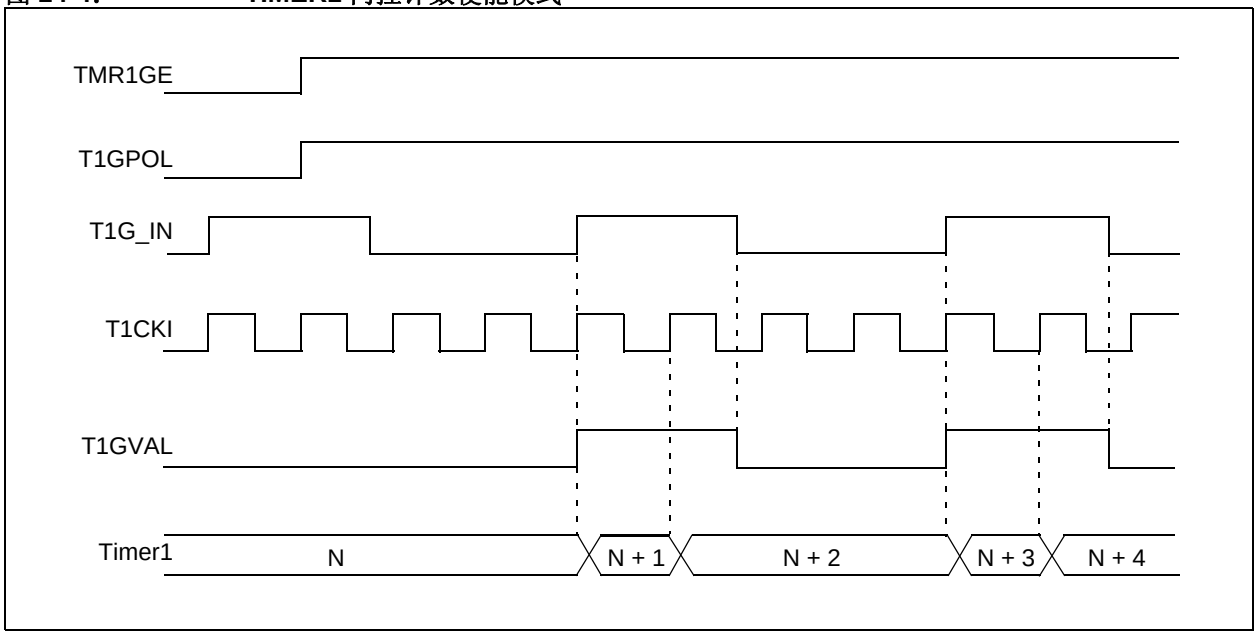
表 14-3： TIMER1 门控使能选择

T1CLK(†)	T1GPOL（T1GCON<6>）	T1G 引脚	Timer1 工作方式
↑	0	0	计数
↑	0	1	保持计数
↑	1	0	保持计数
↑	1	1	计数

† TMR1 的运行时钟。欲知更多信息，请参见图 14-1。

> **注：** 某些模式下，CCP 和 ECCP 模块使用定时器 1 到 8。为 CCP/ECCP 模块分配的特定定时器由 CCPTMRSx 寄存器中的 CCP 定时器使能位决定。欲知更多详细信息，请参见寄存器 19-2、寄存器 19-3 和寄存器 20-2。

图 14-4： TIMER1 门控计数使能模式



14.8.2 TIMER1 门控源选择

Timer1 门控源可从四个时钟源中选择。时钟源选择由 T1GSSx 位 （T1GCON<1:0>）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 输出 （TMR2 递增以匹配 PR2）
10	比较器 1 输出 （比较器逻辑高电平输出）
11	比较器 2 输出 （比较器逻辑高电平输出）

每个可用时钟源的极性也是可选的，由 T1GPOL 位 （T1GCON<6>）控制。

14.8.2.1 T1G 引脚门控操作

T1G 引脚是 Timer1 门控源之一。它可用于向 Timer1 门控电路提供外部时钟源。

14.8.2.2 Timer2 匹配门控操作

TMR2 寄存器将一直递增，直至其与 PR2 寄存器中的值匹配。在紧接着的下一个递增周期，TMR2 将复位为 00h。 TMR2 复位时，将自动产生上升脉冲，并在内部提供给 Timer1 门控电路。此脉冲将保持为高电平一个指令周期，并在下一次匹配时返回低电平状态。

根据 T1GPOL 位的设置，当 TMR2 与 PR2 匹配时，Timer1 计数器递增时间会有所不同。当 T1GPOL = 1 时，Timer1 在 TMR2 与 PR2 匹配后的一个指令周期内递增。当 T1GPOL = 0 时，Timer1 连续递增，但匹配后的第一个指令周期除外，这个周期，门控信号从低变为高。

14.8.2.3 比较器 1 输出门控操作

比较器 1 的输出可以在内部提供给 Timer1 门控电路。在使用 CM1CON 寄存器设置比较器 1 后，Timer1 将根据 CMP1OUT （CMSTAT<5>）位的跳变递增。

14.8.2.4 比较器 2 输出门控操作

比较器 2 的输出可以在内部提供给 Timer1 门控电路。在使用 CM2CON 寄存器设置比较器 2 后，Timer1 将根据 CMP2OUT （CMSTAT<6>）位的跳变递增。

© 2011 Microchip Technology Inc.

14.8.3 TIMER1 门控翻转模式

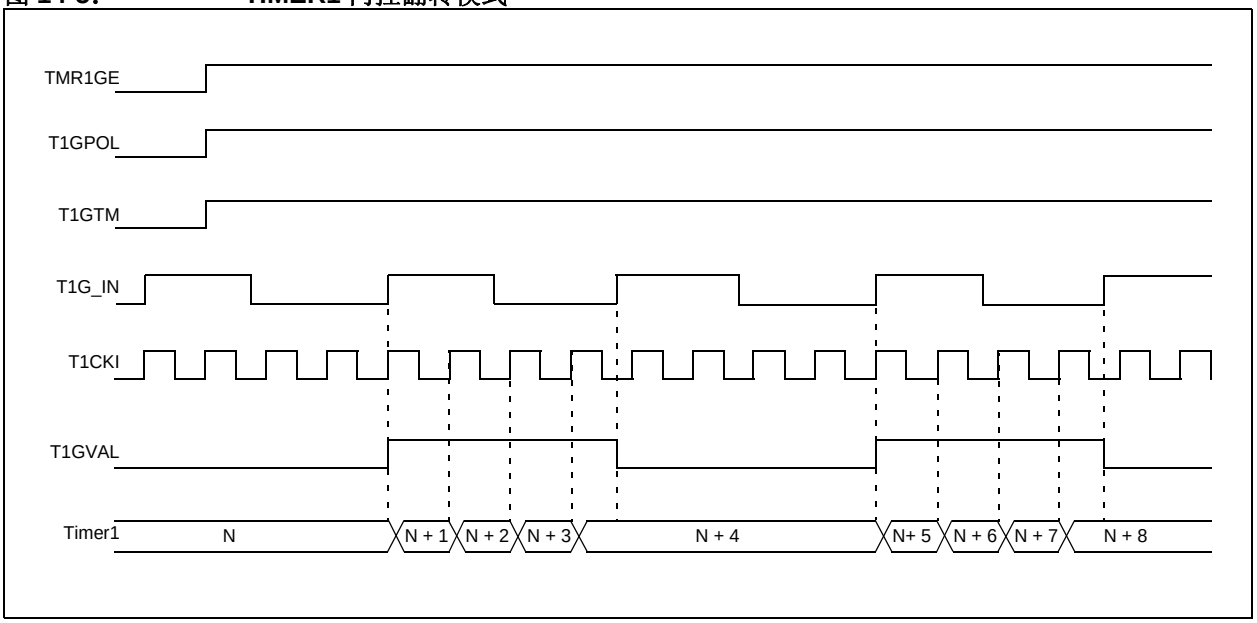
当使能 Timer1 门控翻转模式时，可以测量 Timer1 门控信号的整个周期长度，而不是单个脉冲持续时间。

Timer1 门控源经由在每个信号递增边沿改变状态的触发器传递门控信号。（欲知更多时序信息，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。

可通过将 T1GTM 位（T1GCON<5>）置 1 来使能 Timer1 门控翻转模式。清零 T1GTM 时，触发器清零并保持清零状态。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

当使能 Timer1 门控单脉冲模式时，可以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 将在下一个上升沿完全使能。

在下一个脉冲的后边沿，T1GGO/T1DONE 位将自动清零。将不允许其他门控事件递增 Timer1，直至软件再次将 T1GGO/T1DONE 位置 1。

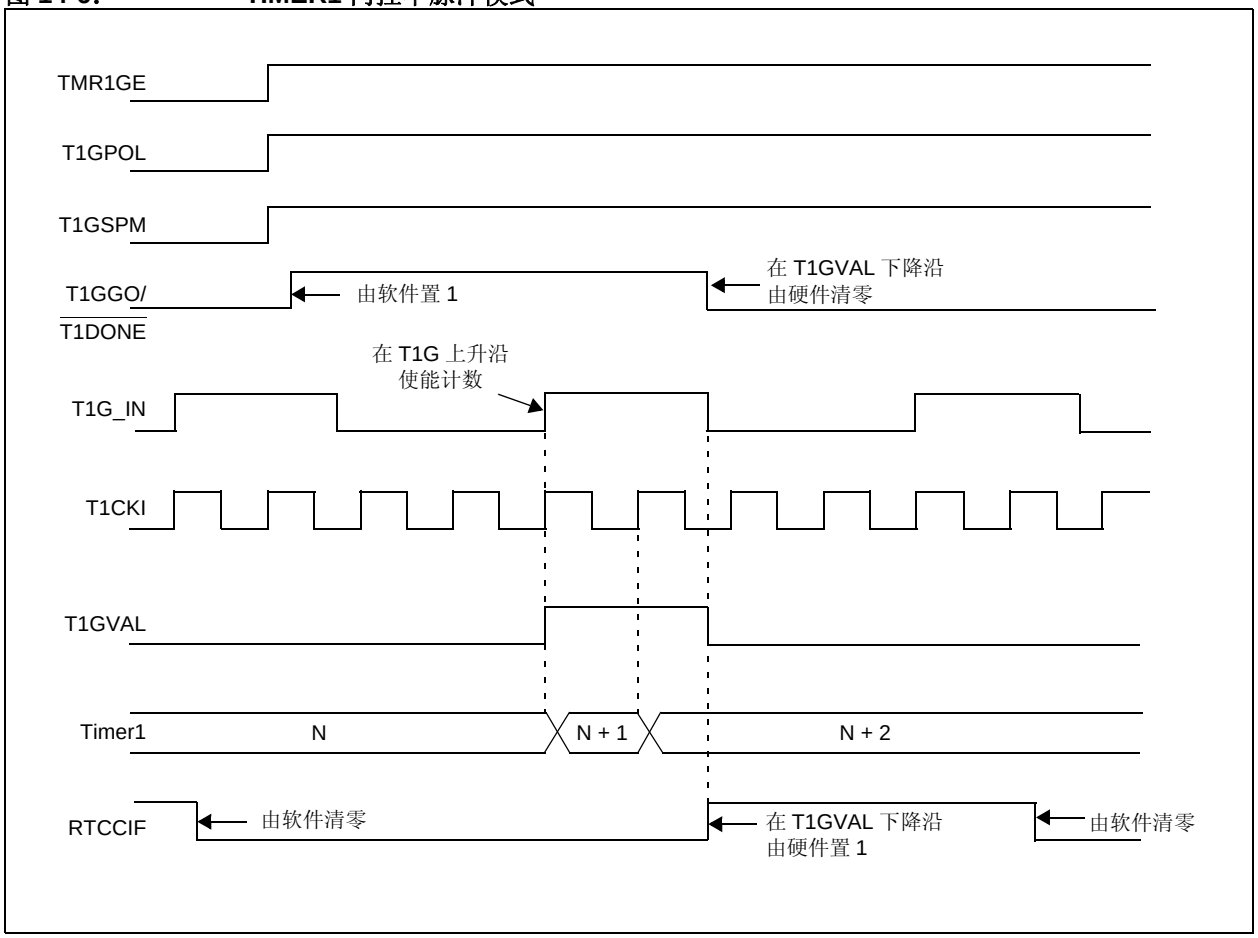
清零T1GCON寄存器的T1GSPM位将同时清零T1GGO/T1DONE 位。（欲知更多时序信息，请参见图 14-6）。

同时使能翻转和单脉冲模式将允许这两部分协同工作。这样可以测量 Timer1 门控源的周期时间。（欲知更多时序信息，请参见图 14-7。）

14.8.5 TIMER1 门控值状态

使用 Timer1 门控值状态时，可以读取门控值的最新电平。该值存储在 T1GVAL 位（T1GCON<2>）中。当禁止 Timer1 门控 （TMR1GE 位清零）时，该位有效。

图 14-6： TIMER1 门控单脉冲模式



© 2011 Microchip Technology Inc.

图 14-7： TIMER1 门控单脉冲和翻转组合模式

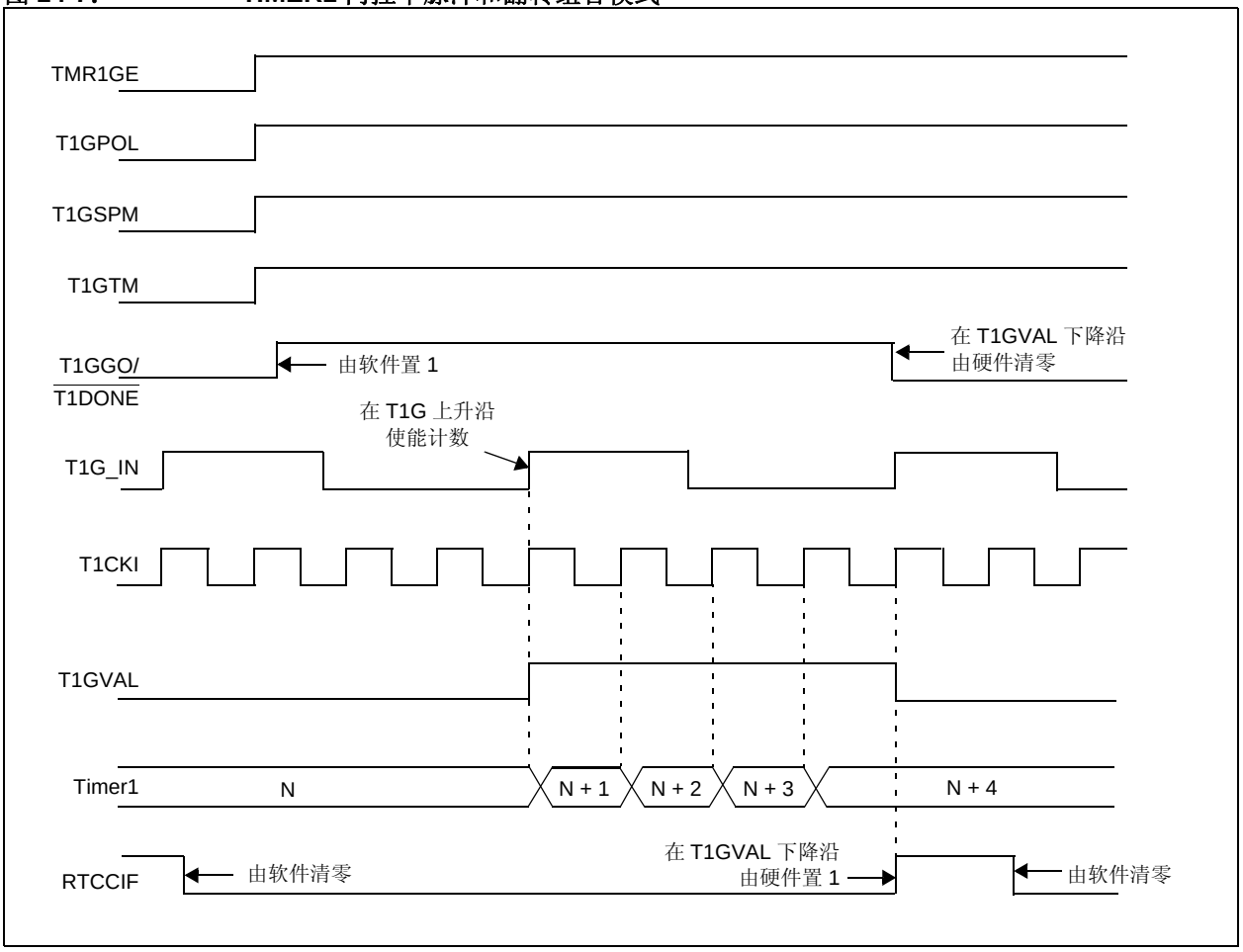


表 14-5： 与 TIMER1 作为定时器 / 计数器相关的寄存器

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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	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	$\overline{\text{T1GGO/T1DONE}}$	T1GVAL	T1GSS1	T1GSS0
OSCCON2	—	SOSCRUN	—	—	SOSCGO	—	MFIOFS	MFIOSEL
PMD1	PSPMD	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBDM

图注： Timer1 模块不使用阴影单元。
注 1： 在 32 KB 器件 （PIC18FX5K22）上未实现。

注：

© 2011 Microchip Technology Inc.

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 （分别是 TMR2 和 PR2）
- 可读写 （以上两个寄存器）
- 软件可编程预分频器 （1:1、 1:4 和 1:16）
- 软件可编程后分频器 （1:1 至 1:16）
- TMR2 与 PR2 匹配时产生中断
- 作为 MSSP 模块的可选移位时钟

此模块的工作由 T2CON 寄存器 （寄存器 15-1）控制，此寄存器用于使能或禁止定时器并配置预分频器和后分频器。可以通过清零控制位 TMR2ON （T2CON<2>）关闭 Timer2，以使功耗最小。

图 15-1 所示为该模块的简化框图。

15.1 Timer2 工作原理

在正常工作中， TMR2 从 00h 开始，在每个时钟周期（Fosc/4）递增 1。4 位计数器 / 预分频器提供了对时钟输入不分频、4 分频和 16 分频三种预分频选项，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。此信号也会使 TMR2 的值在下一个周期复位到 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。在任何方式的器件复位时，TMR2 寄存器都会被清零，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
- 任何方式的器件复位 （上电复位 （POR）、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：** 某些模式下，CCP 和 ECCP 模块使用定时器 1 到 8。为 CCP/ECCP 模块分配的特定定时器由 CCPTMRSx 寄存器中的 CCP 定时器使能位决定。欲知更多详细信息，请参见寄存器 20-2、寄存器 19-2 和寄存器 19-3。

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 = 上电复位时的值	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
•
•
•
0001 = 1:16 后分频比

bit 2 **TMR2ON：**Timer2 使能位
1 = 使能 Timer2
0 = 禁止 Timer2

bit 1-0 **T2CKPS<1:0>：**Timer2 时钟源预分频值选择位
00 = 预分频值为 1
01 = 预分频值为 4
1x = 预分频值为 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 和 PR2 匹配时）可作为 4 位输出计数器 / 后分频器的输入信号。此计数器产生的 TMR2 匹配中断，由其标志位 TMR2IF（PIR1<1>）表示。可以通过将 TMR2 匹配中断允许位 TMR2IE （PIE1<1>）置 1 来允许此中断。

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 到 1:16）中进行选择。

15.3 Timer2 输出

TMR2 不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。

这可将 Timer2 用作 MSSP 模块在 SPI 模式下的移位时钟源。**第 21.0 节 “主同步串行端口 （MSSP）模块”**提供了更多信息。

图 15-1： TIMER2 框图

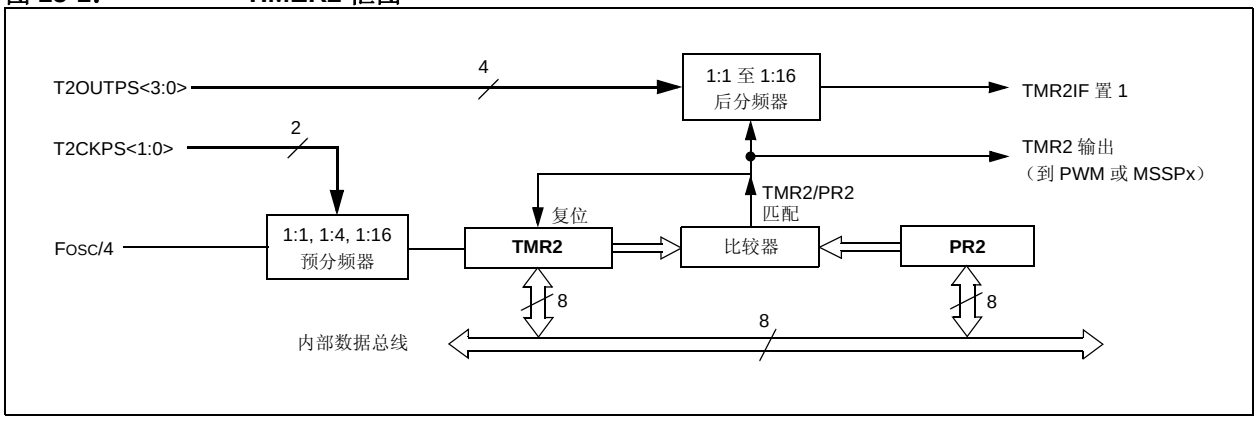


表 15-1： 与 TIMER2 作为定时器 / 计数器时相关的寄存器

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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	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	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBDM

图注： — = 未实现，读为 0。 Timer2 模块不使用阴影单元。

© 2011 Microchip Technology Inc.

16.0 TIMER3/5/7 模块

Timer3/5/7 定时器 / 计数器模块具有以下特性：

- 可通过软件选择作为 16 位定时器 / 计数器
- 可读写的 8 位寄存器 （TMRxH 和 TMRxL）
- 可选择使用器件时钟或 SOSC 内部振荡器作为内部或外部时钟源
- 溢出中断
- ECCP 特殊事件触发模块复位

对于具有 32 KB 程序存储器的器件（PIC18FX5K22），Timer7 未实现。

注：	在本章节中，对于相同的寄存器和位名均采用统称，其中 x 变量表示与 Timer3、 Timer5 或 Timer7 模块相关的项。例如，控制寄存器被命名为 TxCON，其指代 T3CON、 T5CON 和 T7CON。

图 16-1 所示为 Timer3/5/7 模块的简化框图。

Timer3/5/7 模块通过 TxCON 寄存器 （见寄存器 16-1）控制。它还为ECCP模块选择时钟源。（欲知更多信息，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。如果定时器要与捕捉 / 比较功能一起使用，将总是选择其他定时器时钟选项之一。

寄存器 16-1： TxCON：TIMERx 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
TMRxCS1	TMRxCS0	TxCKPS1	TxCKPS0	SOSCEN	$\overline{\text{TxSYNC}}$	RD16	TMRxON
bit 7							bit 0

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

bit 7-6 **TMRxCS<1:0>**：Timerx 时钟源选择位
10 = Timer1 时钟源取决于 SOSCEN 位：
SOSCEN = 0：
使用 T1CKI 引脚上的外部时钟 （上升沿）。
SOSCEN = 1：
根据 SOSCSEL 熔丝位，选择时钟源是 SOSCI/SOSCO 引脚上的晶振还是 SCLKI 引脚的外部时钟。
01 = Timerx 时钟源是系统时钟 （Fosc） (1)
00 = Timerx 时钟源是指令时钟 （Fosc/4）

bit 5-4 **TxCKPS<1:0>**：Timerx 输入时钟预分频比选择位
11 = 1:8 预分频值
10 = 1:4 预分频值
01 = 1:2 预分频值
00 = 1:1 预分频值

bit 3 **SOSCEN**：SOSC 振荡器使能位
1 = 为 Timerx 使能 SOSC/SCLK （基于 SOSCSEL 熔丝位）
0 = 为 Timerx 禁止 SOSC/SCLKI 并使能 TxCKI

bit 2 **$\overline{\text{TxSYNC}}$**：Timerx 外部时钟输入同步控制位
（如果器件时钟来自 Timer1/Timer3，不可用。）
当 TMRxCS<1:0> = 10 时：
1 = 与外部时钟输入不同步
0 = 与外部时钟输入同步
当 TMRxCS<1:0> = 0x 时：
忽略该位； Timer3 使用内部时钟。

bit 1 **RD16**：16 位读 / 写模式使能位
1 = 使能通过一次 16 位操作对 Timerx 寄存器进行读写
0 = 使能通过两次 8 位操作对 Timerx 寄存器进行读写

bit 0 **TMRxON**：Timerx 使能位
1 = 使能 Timerx
0 = 停止 Timerx

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。

© 2011 Microchip Technology Inc.

16.1 Timer3/5/7 门控寄存器

寄存器 14-2 中的 Timer3/5/7 门控寄存器 （TxGCON）
用于对 Timerx 进行门控控制。

寄存器 16-2： TxGCON：TIMERx 门控寄存器 (1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-x	R/W-0	R/W-0
TMRxGE	TxGPOL	TxGTM	TxGSPM	TxGGO/$\overline{\text{TxDONE}}$	TxGVAL	TxGSS1	TxGSS0
bit 7							bit 0

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

bit 7 **TMRxGE：**Timerx 门控使能位
如果 TMRxON = 0：
忽略此位。
如果 TMRxON = 1：
1 = Timerx 计数由 Timerx 门控功能控制
0 = Timerx 计数与 Timerx 门控功能无关

bit 6 **TxGPOL：**Timerx 门控极性位
1 = Timerx 门控高电平有效 （当门控信号为高电平时， Timerx 计数）
0 = Timerx 门控低电平有效 （当门控信号为低电平时， Timerx 计数）

bit 5 **TxGTM：**Timerx 门控翻转模式位
1 = 使能 Timerx 门控翻转模式。
0 = 禁止 Timerx 门控翻转模式并清除翻转触发器
Timerx 门控触发器在每个上升沿触发。

bit 4 **TxGSPM：**Timerx 门控单脉冲模式位
1 = 使能 Timerx 门控单脉冲模式并对 Timerx 进行门控控制
0 = 禁止 Timerx 门控单脉冲模式

bit 3 **TxGGO/$\overline{\text{TxDONE}}$：**Timerx 门控单脉冲采集状态位
1 = Timerx 门控单脉冲采集已就绪，正在等待一个边沿
0 = Timerx 门控单脉冲采集已完成或尚未开始
当 TxGSPM 清零时，该位自动清零。

bit 2 **TxGVAL：**Timerx 门控当前状态位
指示可提供给 TMRxH:TMRxL 的 Timerx 门控的当前状态。不受 Timerx 门控使能位 （TMRxGE）影响。

bit 1-0 **TxGSS<1:0>：**Timerx 门控时钟源选择位
11 = 比较器 2 输出
10 = 比较器 1 输出
01 = TMR(x+1) 匹配 PR(x+1) 输出 (2)
00 = Timer1 门控引脚
当 TMRxGE = 1 时看门狗定时器振荡器关闭，与 TMRxON 状态无关。

注 **1：** 建议在编程 TxCON 前先编程 TxGCON。
2： Timer(x+1) 是 Timer4/6/8，而 Timerx 是 Timer3/5/7。

寄存器 16-3： OSCCON2：振荡器控制寄存器 2

U-0	R-0	U-0	U-0	R/W-0	U-0	R-x	R/W-0
—	SOSCRUN	—	—	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 = 上电复位时的值	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 = 系统时钟来自不同于 SOSC 的时钟

bit 5-4 **未实现：**读为 0

bit 3 **SOSCGO：**振荡器起振控制位
1 = 即使没有其他时钟源请求振荡器，它仍会运行
0 = 当没有其他时钟源请求振荡器时，它将关闭（当 SOSC 选择使用数字时钟输入而不是外部晶振运行时，该位无效）。

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 = MF-INTOSC 用于替代频率为 500 kHz、 250 kHz 和 31.25 kHz 的 HF-INTOSC
0 = 不使用 MF-INTOSC

© 2011 Microchip Technology Inc.

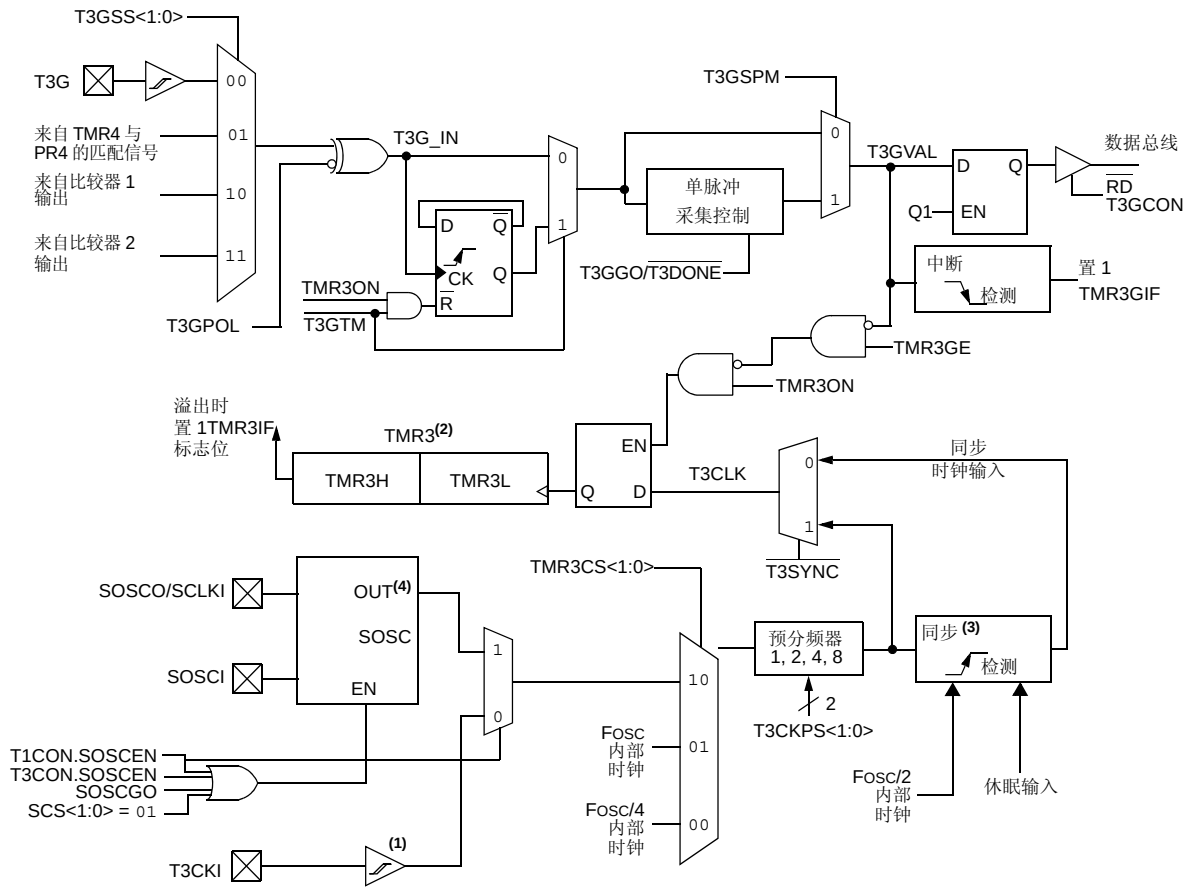
16.2 Timer3/5/7 工作原理

Timer3、 Timer5 和 Timer7 可在以下模式运行：

- 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 TMRxCSx （TxCON<7:6>）决定。当 TMRxCSx 位清零 （= 00）时， Timer3/5/7 在每个内部指令周期 （Fosc/4）递增 1。当 TMRxCSx= 01 时， Timer3/5/7 时钟源为系统时钟 （Fosc）；且当 TMRxCSx = 10 时， Timer3/5/7 为计数器工作，采用来自 TxCKI 引脚（首个下降沿之后的上升沿）或 SOSC 振荡器的外部时钟。

图 16-1： TIMER3/5/7 框图



注 1： 当使用 T3CKI 时， ST 缓冲器为高速型。
2： Timer3 寄存器在每个上升沿递增。
3： 处于休眠状态时，无法执行同步。
4： SOSC 的输出由 SOSCSEL<1:0> 配置位决定。

16.3 Timer3/5/7 16 位读 / 写模式

可将 Timer3/5/7 配置为 16 位读写模式。当 RD16 控制位（TxCON<1>）置 1 时， TMRxH 的地址将映射到 Timer3/5/7 的高字节缓冲寄存器。读 TMRxL 将把 Timer3/5/7 的高字节内容装入 Timerx 高字节缓冲寄存器。这种方式使用户可以精确地读取 Timer3/5/7 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计满返回，而不得不验证读取的有效性。

对 Timer3/5/7 的高字节进行写操作也必须通过 TMRxH 缓冲寄存器进行。在写入 TMRxL 的同时，使用 TMRxH 的内容更新 Timer3/5/7 的高字节。这样允许用户将 16 位值一次写入 Timer3/5/7 的高字节和低字节。

在该模式下不能直接读写 Timer3/5/7 的高字节。所有读写都必须通过 Timerx 高字节缓冲寄存器进行。

写入 TMRxH 不会清零 Timer3/5/7 预分频器。只有在写 TMRxL 时才会清零预分频器。

16.4 使用 SOSC 振荡器作为 Timer3/5/7 时钟源

SOSC 内部振荡器可用作 Timer3/5/7 的时钟源。通过任何请求 SOSC 振荡器的外设可使能该振荡器。可使用以下八种方式使能：如果 SOSC 选择为任何奇数定时器的时钟源，则通过各自的 SOSCEN 位（TxCON<3>）使能；如果 SOSC 选择为 RTCC 时钟源，则通过 RTCOSC 配置位（CONFIG3L<1>）使能；如果 SOSC 选择为 CPU 时钟源，或 SOSCGO 位（OSCCON2<3>）置 1，则通过 SCS 位（OSCCON<1:0>）使能。SOSCGO 位用于预热 SOSC，使之在有任何外设请求前做好准备。要将其作为 Timer3/5/7 时钟源， TMRxCS 位也必须置 1。如前面所述，这还会将 Timer3/5/7 配置为在振荡源的每个上升沿递增 1。

SOSC 振荡器如**第 14.5 节 “SOSC 振荡器”**所述。

© 2011 Microchip Technology Inc.

16.5 Timer3/5/7 门控

Timer3/5/7 可配置为自主计数，或使用 Timer3/5/7 门控电路使能和禁止计数。这也称为 Timer3/5/7 门控计数使能。

Timer3/5/7 门控也可由多个可选时钟源驱动。

16.5.1 TIMER3/5/7 门控计数使能

通过将 TMRxGE 位 （TxGCON<7>）置 1 使能 Timerx 门控使能模式。可使用 TxGPOL 位（TxGCON<6>）配置 Timerx 门控使能模式的极性。

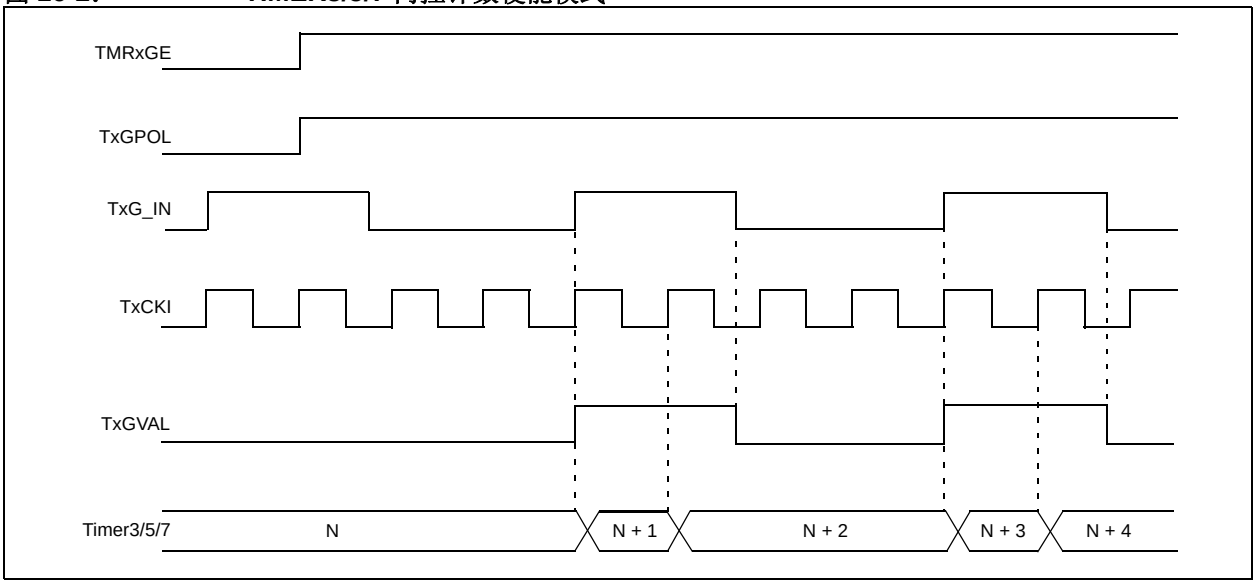
当使能 Timerx 门控使能模式时，Timer3/5/7 将在 Timer3/5/7 时钟源的上升沿递增。当禁止 Timerx 门控使能模式时，不进行计数递增， Timer3/5/7 保持当前计数。有关时序详细信息，请参见图 16-2。

表 16-1： TIMER3/5/7 门控使能选择

TxCLK(†)	TxGPOL （TxGCON<6>）	TxG 引脚	Timerx 工作方式
↑	0	0	计数
↑	0	1	保持计数
↑	1	0	保持计数
↑	1	1	计数

† TTMR3/5/7 的运行时钟。欲知更多信息，请参见图 16-1 中的 TxCLK。

图 16-2： TIMER3/5/7 门控计数使能模式



16.5.2 TIMER3/5/7 门控源选择

Timer3/5/7 门控源可从四个时钟源选择。时钟源的选择由 TxGSS<1:0> 位（TxGCON<1:0>）控制。每个可用时钟源的极性也是可选的，由 TxGPOL 位（TxGCON<6>）控制。

表 16-2： TIMER3/5/7 门控源

TxGSS<1:0>	Timerx 门控源
00	Timerx 门控引脚
01	TMR(x+1) 匹配 PR(x+1) （TMR(x+1) 递增以匹配 PR(x+1)）
10	比较器 1 输出 （比较器逻辑高电平输出）
11	比较器 2 输出 （比较器逻辑高电平输出）

16.5.2.1 TxG 引脚门控操作

TxG 引脚是 Timer3/5/7 门控源之一。它可用于为 Timerx 门控电路提供外部时钟源。

16.5.2.2 Timer4/6/8 匹配门控操作

TMR(x+1) 寄存器将一直递增，直至其与 PR(x+1) 寄存器中的值匹配。在紧接着的下一个递增周期， TMRx 将复位为 00h。复位时，将自动产生上升脉冲，并在内部提供给 Timerx 门控电路。此脉冲将保持为高电平一个指令周期，并在下一次匹配时返回低电平状态。

根据 TxGPOL 位的设置，TMR(x+1) 与 PR(x+1) 匹配时的 Timerx 递增时间会有所不同。当 TxGPOL = 1 时，Timerx 在 TMR(x+1) 与 PR(x+1) 匹配后的一个指令周期内递增。当 TxGPOL = 0 时，Timerx 连续递增，但匹配后的第一个指令周期除外，在这个周期，门控信号由低变高。

16.5.2.3 比较器 1 输出门控操作

比较器 1 的输出可以在内部提供给 Timerx 门控电路。在使用 CM1CON 寄存器设置比较器 1 后，Timerx 将根据 CMP1OUT （CMSTAT<5>）位的跳变递增。

16.5.2.4 比较器 2 输出门控操作

比较器 2 的输出可以在内部提供给 Timerx 门控电路。在使用 CM2CON 寄存器设置比较器 2 后，Timerx 将根据 CMP2OUT （CMSTAT<6>）位的跳变递增。

16.5.3 TIMER3/5/7 门控翻转模式

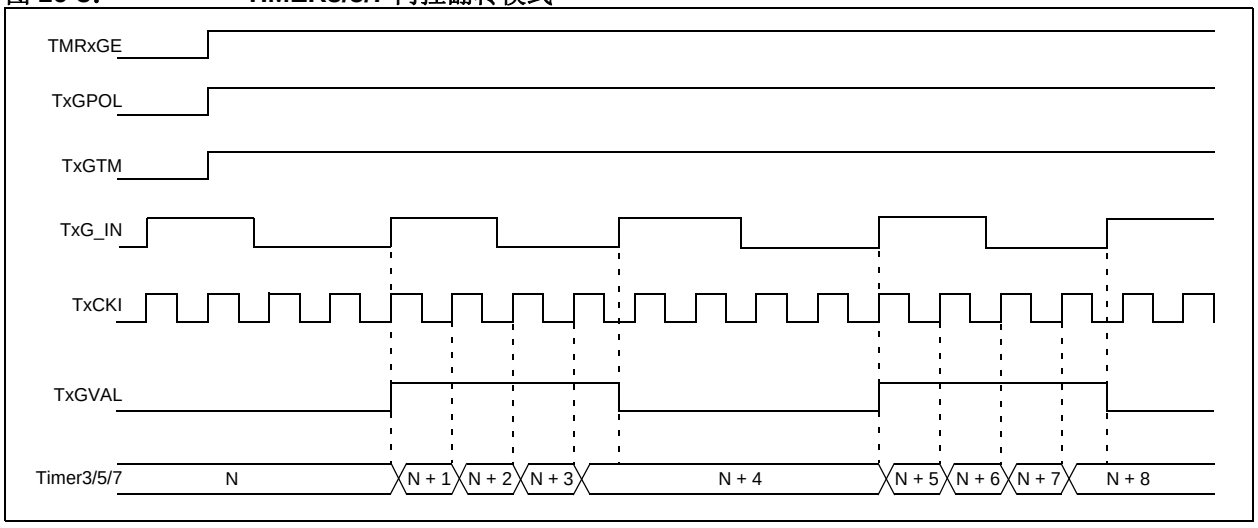
当使能 Timer3/5/7 门控触发模式时，可以测量 Timer3/5/7 门控信号的整个周期长度，而不是单个电平脉冲持续时间。

Timerx 门控源经由在每个信号递增边沿改变状态的触发器传递信号。（欲知更多时序信息，请参见图 16-3）。

TxGVAL 位将指示翻转模式何时工作以及定时器何时计数。

可通过将 TxGTM 位（TxGCON<5>）置 1 来使能 Timer3/5/7 门控翻转模式。清零 TxGTM 位时，触发器清零并保持清零状态。这对于控制测量哪个边沿是必需的。

图 16-3： TIMER3/5/7 门控翻转模式



© 2011 Microchip Technology Inc.

16.5.4 TIMER3/5/7 门控单脉冲模式

当使能 Timer3/5/7 门控单脉冲模式时，可以捕捉到一个单脉冲门控事件。可通过将 TxGSPM 位（TxGCON<4>）置 1 来先使能 Timer3/5/7 门控单脉冲模式。接下来，必须将 TxGGO/TxDONE 位（TxGCON<3>）置 1。

Timer3/5/7 将在下一个上升沿完全使能。在下一个脉冲的后边沿，TxGGO/TxDONE 位将自动清零。不允许其他门控事件递增 Timer3/5/7，直至软件再次将 TxGGO/TxDONE 位置 1。

清零 TxGSPM 位将同时清零 TxGGO/$\overline{\text{TxDONE}}$ 位。（欲知更多时序信息，请参见图 16-4）。

同时使能翻转和单脉冲模式将允许这两部分协同工作。这样就可以测量 Timer3/5/7 门控源的周期时间。（欲知更多时序信息，请参见图 16-5）。

图 16-4： TIMER3/5/7 门控单脉冲模式

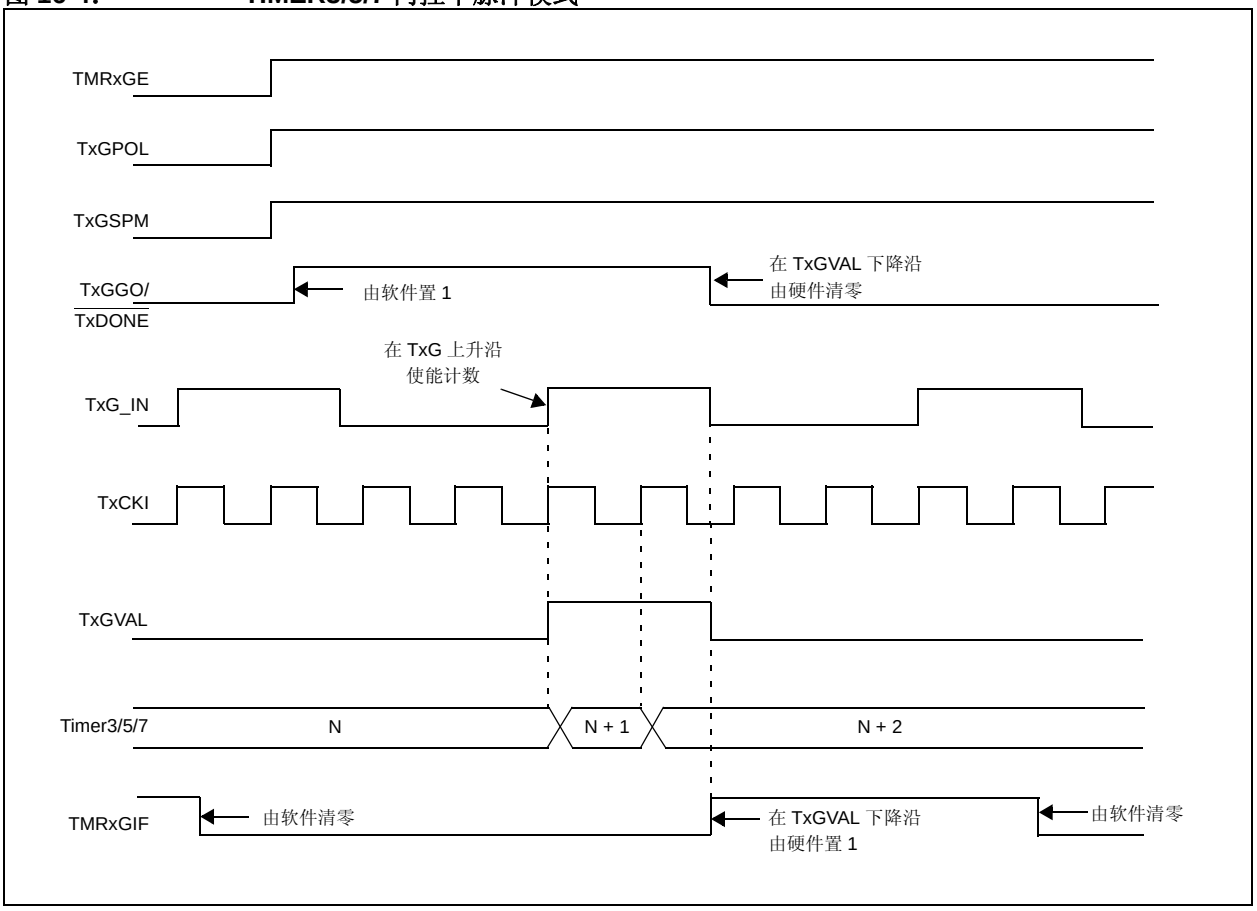
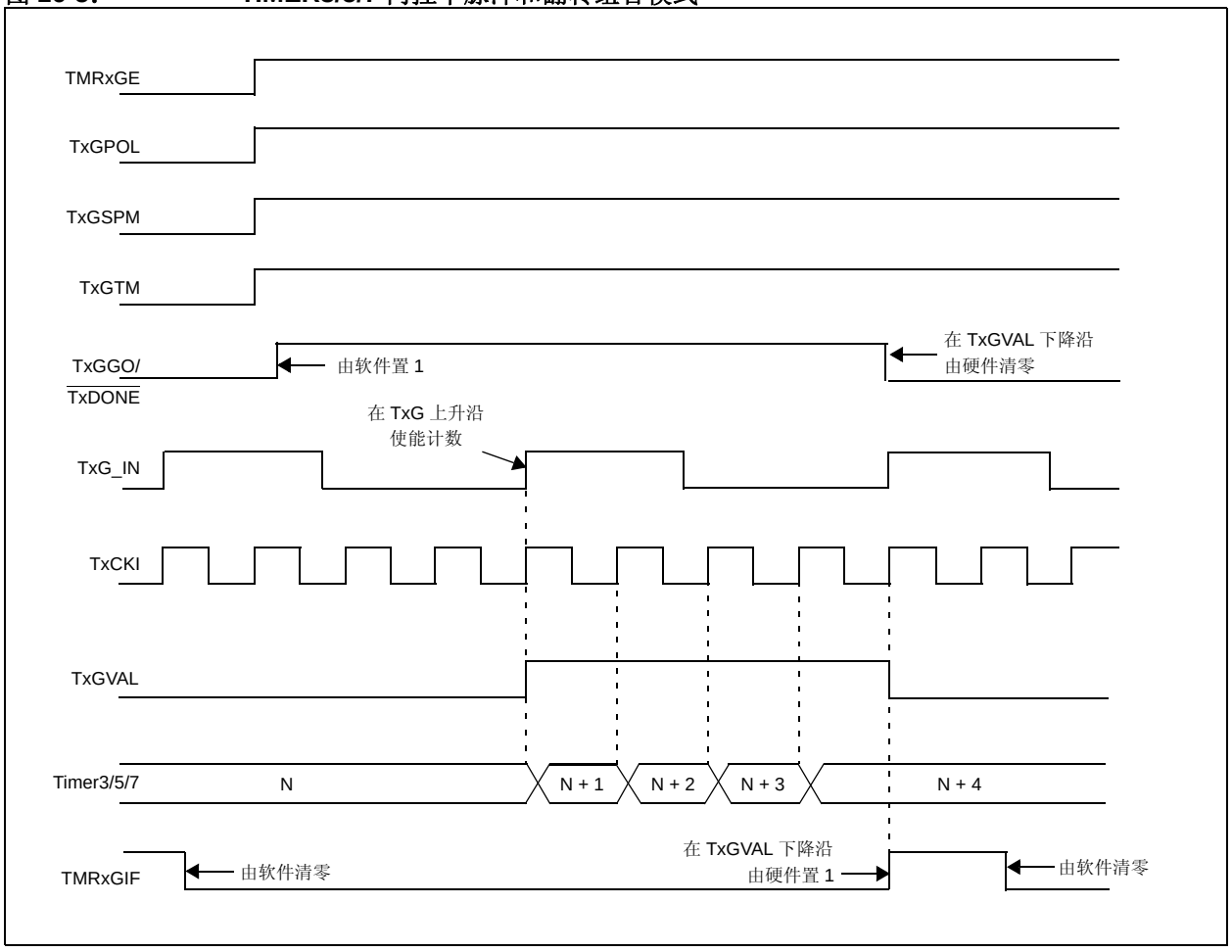


图 16-5： TIMER3/5/7 门控单脉冲和翻转组合模式



16.5.5 TIMER3/5/7 门控值状态

使用 Timer3/5/7 门控值状态时，可以读取门控值的最新电平。该值存储在 TxGVAL 位（TxGCON<2>）中。即使未使能Timer3/5/7门控（TMRxGE位清零），TxGVAL 位仍有效。

16.5.6 TIMER3/5/7 门控事件中断

当允许 Timer3/5/7 门控事件中断时，可以在完成门控事件时产生中断。当出现 TxGVAL 的下降沿时， PIRx 寄存器中的 TMRxGIF 标志位将置 1。如果 PIEx 寄存器中的 TMRxGIE 位置 1，则会识别出这个中断。

即使未允许 Timer3/5/7 门控（TMRxGE 位清零），TMRxGIF 标志位仍有效。

© 2011 Microchip Technology Inc.

16.6 Timer3/5/7 中断

TMRx 寄存器对 （TMRxH:TMRxL）从 0000h 开始递增，一直到 FFFFh，然后计满返回到 0000h。如果允许 Timerx 中断，在溢出时产生此中断，并置 1 中断标志位 TMRxIF。表 16-3 给出了各模块的标志位。

表 16-3： TIMER3/5/7 中断标志位

定时器模块	标志位
3	PIR2<1>
5	PIR5<1>
7	PIR5<3>

将 TMRxIE 位置 1 或清零将分别允许或禁止该中断。表 16-4 给出了各模块的中断允许位。

表 16-4： TIMER3/5/7 中断允许位

定时器模块	允许位
3	PIE2<1>
5	PIE5<1>
7	PIE5<3>

16.7 使用 ECCP 特殊事件触发信号复位 Timer3/5/7

如果 ECCP 模块被配置为使用 Timerx，并且在比较模式（CCPxM<3:0> = 1011）下产生特殊事件触发信号，触发信号将复位 Timerx。如果使能了 A/D 模块，来自 ECCP2 的触发信号还将启动 A/D 转换。（欲知更多信息，请参见**第 20.3.4 节 “特殊事件触发信号”**）。

要使用这一功能，必须将 Timerx 模块配置为定时器或同步计数器。在这种情况下，CCPRxH:CCPRxL 寄存器对实际上变成了 Timerx 的周期寄存器。

如果 Timerx 工作在异步计数器模式下，复位操作将不起作用。

如果对 Timerx 的写操作和特殊事件触发信号同时发生，则写操作优先。

> **注：** 来自ECCPx模块的特殊事件触发信号将仅清零TMR3寄存器的内容，并不将TMR3IF中断标志位 （PIR1<0>）置为 1。

> **注：** 某些模式下，CCP 和 ECCP 模块使用定时器 1 到 8。为 CCP/ECCP 模块分配的特定定时器由 CCPTMRSx 寄存器中的 CCP 定时器使能位决定。欲知更多详细信息，请参见寄存器 19-2、寄存器 19-3 和寄存器 20-2。

表 16-5： 与 TIMER3/5/7 作为定时器 / 计数器时相关的寄存器

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	TMR7GIF(1)	TMR12IF(1)	TMR10IF(1)	TMR8IF	TMR7IF(1)	TMR6IF	TMR5IF	TMR4IF
IPR5	TMR7GIP(1)	TMR12IP(1)	TMR10IP(1)	TMR8IP	TMR7IP(1)	TMR6IP	TMR5IP	TMR4IP
PIE5	TMR7GIE(1)	TMR12IE(1)	TMR10IE(1)	TMR8IE	TMR7IE(1)	TMR6IE	TMR5IE	TMR4IE
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE2	OSCFIE	—	SSP2IE	BCL2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
PIR2	OSCFIF	—	SSP2IF	BCL2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
IPR2	OSCFIP	—	SSP2IP	BCL2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
TMR3H	Timer3 寄存器的高字节							
TMR3L	Timer3 寄存器的低字节							
T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/ T3DONE	T3GVAL	T3GSS1	T3GSS0
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON
TMR5H	Timer5 寄存器的高字节							
TMR5L	Timer5 寄存器的低字节							
T5GCON	TMR5GE	T5GPOL	T5GTM	T5GSPM	T5GGO/ T5DONE	T5GVAL	T5GSS1	T5GSS0
T5CON	TMR5CS1	TMR5CS0	T5CKPS1	T5CKPS0	SOSCEN	$\overline{\text{T5SYNC}}$	RD16	TMR5ON
TMR7H(1)	Timer7 寄存器的高字节							
TMR7L(1)	Timer7 寄存器的低字节							
T7GCON(1)	TMR7GE	T7GPOL	T7GTM	T7GSPM	T7GGO/ T7DONE	T7GVAL	T7GSS1	T7GSS0
T7CON(1)	TMR7CS1	TMR7CS0	T7CKPS1	T7CKPS0	SOSCEN	$\overline{\text{T7SYNC}}$	RD16	TMR7ON
OSCCON2	—	SOSCRUN	—	—	SOSCGO	—	MFIOFS	MFIOSEL
PMD1	PSPMD	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBMD
PMD2	TMR10MD(1)	TMR8MD	TMR7MD(1)	TMR6MD	TMR5MD	CMP3MD	CMP2MD	CMP2MD

图注： — = 未实现，读为 0。 Timer3/5/7 模块不使用阴影单元。

注 1： 在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现。

© 2011 Microchip Technology Inc.

17.0 TIMER4/6/8/10/12 模块

Timer4/6/8/10/12 定时器模块具有以下特性：

- 8 位定时器寄存器 （TMRx）
- 8 位周期寄存器 （PRx）
- 可读写 （所有寄存器）
- 软件可编程预分频器 （1:1、 1:4 和 1:16）
- 软件可编程后分频器 （1:1 至 1:16）
- TMRx 与 PRx 匹配时产生中断

对于具有 32 KB 程序存储器的器件（PIC18FX5K22），Timer10 和 Timer12 未实现。

> **注：** 在本章节中，对于相同的寄存器和位名均采用统称，其中 x 变量表示与 Timer4、Timer6、Timer8、 Timer10 或 Timer12 模块相关的项。例如，控制寄存器被命名为 TxCON，其指代 T4CON、 T6CON、 T8CON、 T10CON 和 T12CON。

如寄存器 17-1 所示， Timer4/6/8/10/12 模块具有一个控制寄存器。可以通过清零控制位 TMRxON（TxCON<2>）关闭 Timer4/6/8/10/12，以使功耗最小。Timer4/6/8/10/12 的预分频器和后分频器选择也由该寄存器控制。图 17-1 为 Timer4/6/8/10/12 模块的简化框图。

17.1 Timer4/6/8/10/12 工作原理

Timer4/6/8/10/12 可用作 ECCP 模块在 PWM 模式下的 PWM 时基。 TMRx 寄存器可读写，并且可以由任意器件复位清零。输入时钟（Fosc/4）具有 1:1、1:4 或 1:16 三种预分频比选项，由控制位 TxCKPS<1:0>（TxCON<1:0>）选择。 TMRx 的匹配输出经过 4 位后分频器（分频范围为 1:1 至 1:16）产生 TMRx 中断并置 1 标志位 TMRxIF。表 17-1 给出了每个模块的标志位。

表 17-1： TIMER4/6/8/10/12 标志位

定时器模块	标志位 PIR5<x>	定时器模块	标志位 PIR5<x>
4	0	10	5
6	2	12	6
8	4		

如表 17-2 所示，可以通过置 1 或清零 Timerx 中断允许位 TMRxIE 来允许或禁止该中断。

表 17-2： TIMER4/6/8/10/12 中断允许位

定时器模块	允许位 PIE5<x>	定时器模块	允许位 PIE5<x>
4	0	10	5
6	2	12	6
8	4		

发生以下任一情况时，预分频器和后分频器计数器清零：

- 对 TMRx 寄存器进行写操作
- 对 TxCON 寄存器进行写操作
- 任何方式的器件复位 （上电复位 （POR）、MCLR 复位、看门狗定时器复位 （WDTR）或欠压复位 （BOR））

写 TxCON 时， TMRx 不清零。

> **注：** 某些模式下，CCP 和 ECCP 模块使用定时器 1 到 8。为 CCP/ECCP 模块分配的特定定时器由 CCPTMRSx 寄存器中的 CCP 定时器使能位决定。欲知更多详细信息，请参见寄存器 19-2、寄存器 19-3 和寄存器 20-2。

寄存器 17-1： TxCON：TIMERx 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
—	TxOUTPS3	TxOUTPS2	TxOUTPS1	TxOUTPS0	TMRxON	TxCKPS1	TxCKPS0
bit 7							bit 0

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

bit 7 **未实现：**读为 0

bit 6-3 **TxOUTPS<3:0>：**Timerx 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 **TMRxON：**Timerx 使能位
1 = 使能 Timerx
0 = 禁止 Timerx

bit 1-0 **TxCKPS<1:0>：**Timerx 时钟预分频值选择位
00 = 预分频值为 1
01 = 预分频值为 4
1x = 预分频值为 16

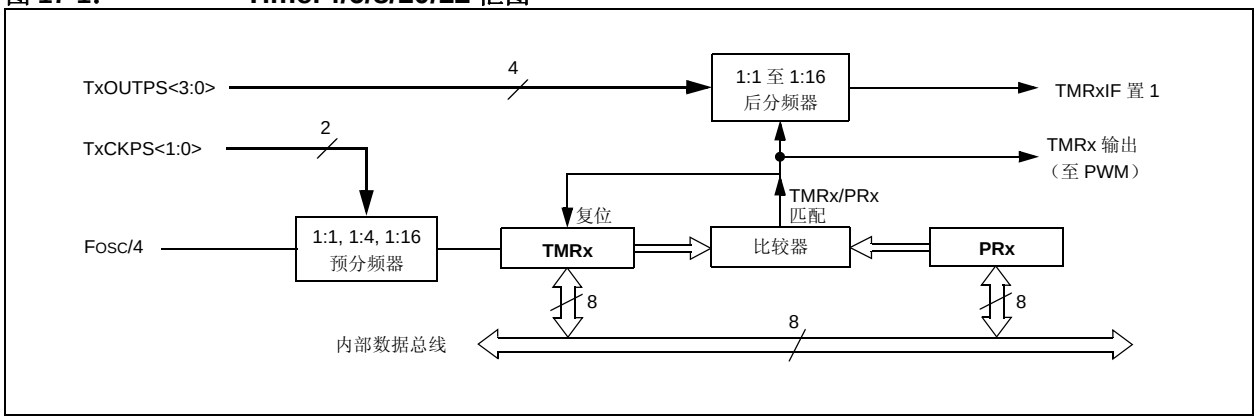
17.2 Timer4/6/8/10/12 中断

Timer4/6/8/10/12 模块具有可读写的 8 位周期寄存器 PRx。Timer4/6/8/10/12 从 00h 开始递增直至它们与 PR4/6/8/10/12 匹配，然后在下一个递增周期复位到 00h。复位时 PRx 寄存器初始化为 FFh。

17.3 TMRx 输出

TMRx 输出（在通过后分频器之前）将仅作为 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： Timer4/6/8/10/12 框图



© 2011 Microchip Technology Inc.

表 17-3： 与 TIMER4/6/8/10/12 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
IPR5	TMR7GIP(1)	TMR12IP(1)	TMR10IP(1)	TMR8IP	TMR7IP(1)	TMR6IP	TMR5IP	TMR4IP
PIR5	TMR7GIF(1)	TMR12IF(1)	TMR10IF(1)	TMR8IF	TMR7IF(1)	TMR6IF	TMR5IF	TMR4IF
PIE5	TMR7GIE(1)	TMR12IE(1)	TMR10IE(1)	TMR8IE	TMR7IE(1)	TMR6IE	TMR5IE	TMR4IE
TMR4	Timer4 寄存器							
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
PR4	Timer4 周期寄存器							
TMR6	Timer6 寄存器							
T6CON	—	T6OUTPS3	T6OUTPS2	T6OUTPS1	T6OUTPS0	TMR6ON	T6CKPS1	T6CKPS0
PR6	Timer6 周期寄存器							
TMR8	Timer8 寄存器							
T8CON	—	T8OUTPS3	T8OUTPS2	T8OUTPS1	T8OUTPS0	TMR8ON	T8CKPS1	T8CKPS0
PR8	Timer8 周期寄存器							
TMR10(1)	Timer10 寄存器							
T10CON(1)	—	T10OUTPS3	T10OUTPS2	T10OUTPS1	T10OUTPS0	TMR10ON	T10CKPS1	T10CKPS0
PR10(1)	Timer10 周期寄存器							
TMR12(1)	Timer12 寄存器							
T12CON(1)	—	T12OUTPS3	T12OUTPS2	T12OUTPS1	T12OUTPS0	TMR12ON	T12CKPS1	T12CKPS0
PR12(1)	Timer12 周期寄存器							
PMD1	PSPMD	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBMD
PMD2	TMR10MD(1)	TMR8MD	TMR7MD(1)	TMR6MD	TMR5MD	CMP3MD	CMP2MD	CMP2MD
PMD3	CCP10MD(1)	CCP9MD(1)	CCP8MD	CCP7MD	CCP6MD	CCP5MD	CCP4MD	TMR12MD(1)

图注： — = 未实现，读为 0。 Timer4/6/8/10/12 模块不使用阴影单元。

注 1： 在具有 32 KB 的程序存储器的器件 （PIC18FX5K22）上未实现。

注：

© 2011 Microchip Technology Inc.

18.0 实时时钟和日历 （RTCC）

实时时钟和日历 （RTCC）模块的主要功能包括：

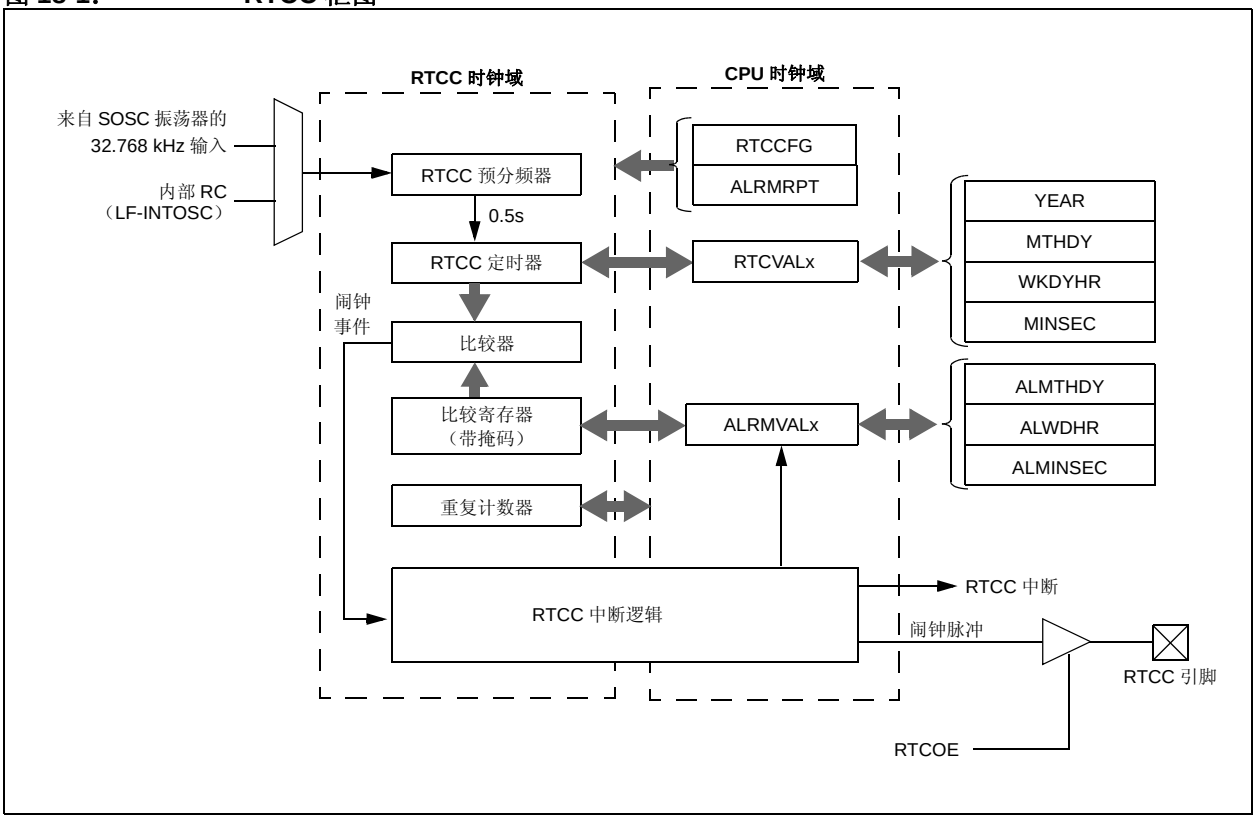
- 时间：小时、分钟和秒
- 24 小时格式 （军用时间）
- 日历：星期、日期、月和年
- 闹钟可配置
- 年份范围：2000 到 2099
- 闰年校正
- 用于获得紧凑固件的 BCD 格式
- 为低功耗工作进行了优化
- 带自动调节的用户校准
- 校准范围：每月 ±2.64 秒误差
- 要求：外部 32.768 kHz 时钟晶振
- RTCC 引脚上的闹钟脉冲或秒时钟输出

RTCC 模块是专为需要长时间维持精确时间的应用设计的，无需或很少需要 CPU 干预。该模块为低功耗使用进行了优化，以便在跟踪时间时延长电池寿命。

该模块是百年时钟和日历，能自动检测闰年。时钟的范围是从 2000 年 1 月 1 日 00:00:00 （午夜）到 2099 年 12 月 31 日 23:59:59。

小时数采用 24 小时 （军用时间）格式计量。该时钟提供一秒的时间粒度，用户可看到半秒的时间间隔。

图 18-1： RTCC 框图



18.1 RTCC 模块寄存器

RTCC 模块寄存器分为以下几类：

RTCC 控制寄存器

- RTCCFG
- RTCCAL
- PADCFG1
- ALRMCFG
- ALRMRPT

RTCC 值寄存器

- RTCVALH
- RTCVALL

  这两种寄存器都可以访问以下寄存器：
  - YEAR
  - MONTH
  - DAY
  - WEEKDAY
  - HOUR
  - MINUTE
  - SECOND

闹钟值寄存器

- ALRMVALH
- ALRMVALL

  这两种寄存器都可以访问以下寄存器：
  - ALRMMNTH
  - ALRMDAY
  - ALRMWD
  - ALRMHR
  - ALRMMIN
  - ALRMSEC

注：	RTCVALH 和 RTCVALL 寄存器可以通过 RTCRPT<1:0>（RTCCFG<1:0>）访问。ALRMVALH 和 ALRMVALL 可以通过 ALRMPTR<1:0>（ALRMCFG<1:0>）访问。

© 2011 Microchip Technology Inc.

18.1.1 RTCC 控制寄存器

寄存器 18-1： RTCCFG：RTCC 配置寄存器 (1)

R/W-0	U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0
RTCEN(2)	—	RTCWREN(4)	RTCSYNC	HALFSEC(3)	RTCOE	RTCPTR1	RTCPTR0
bit 7							bit 0

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

bit 7 **RTCEN：**RTCC 使能位 (2)
1 = 使能 RTCC 模块
0 = 禁止 RTCC 模块

bit 6 **未实现：**读为 0

bit 5 **RTCWREN：**RTCC 值寄存器写使能位 (4)
1 = RTCVALH 和 RTCVALL 寄存器可由用户写入
0 = RTCVALH 和 RTCVALL 寄存器已锁定，用户不能写入

bit 4 **RTCSYNC：**RTCC 值寄存器读同步位
1 = 由于计满返回，RTCVALH、RTCVALL 和 ALRMRPT 寄存器在读操作过程中可能改变，从而导致读取的数据无效。如果两次读取寄存器得到的数据相同，可认为数据是有效的。
0 = 在读取 RTCVALH、 RTCVALL 和 ALCFGRPT 寄存器时无需考虑计满返回

bit 3 **HALFSEC：**半秒状态位 (3)
1 = 一秒的后半秒
0 = 一秒的前半秒

bit 2 **RTCOE：**RTCC 输出使能位
1 = 使能 RTCC 时钟输出
0 = 禁止 RTCC 时钟输出

bit 1-0 **RTCPTR<1:0>：**RTCC 值寄存器窗口指针位
读 RTCVALH 和 RTCVALL 寄存器时，指向相应的 RTCC 值寄存器。RTCPTR<1:0> 值会在每次读 / 写 RTCVALH<15:8> 时递减 1，直到达到 00。

RTCVALH：
00 = 分钟数
01 = 星期
10 = 月
11 = 保留

RTCVALL：
00 = 秒数
01 = 小时数
10 = 日
11 = 年

注 1： RTCCFG 寄存器只受上电复位影响。
2： 只有在 RTCWREN = 1 时，才允许写入 RTCEN 位。
3： 此位为只读位；在写入 MINSEC 寄存器的低半部分时清零。 0
4： 只能使用解锁序列写 RTCWREN 位 （见例 18-1）。

寄存器 18-2： RTCCAL：RTCC 校准寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
bit 7							bit 0

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

bit 7-0 **CAL<7:0>**：RTC 漂移校准位
01111111 = 最大正向调节。每分钟增加 508 个 RTC 时钟脉冲。
.
.
.
00000001 = 最小正向调节。每分钟增加 4 个 RTC 时钟脉冲。
00000000 = 无调节
11111111 = 最小负向调节。每分钟减少 4 个 RTC 时钟脉冲。
.
.
.
10000000 = 最大负向调节。每分钟减少 512 个 RTC 时钟脉冲。

寄存器 18-3： PADCFG1：引脚配置寄存器

R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	U-0
RDPU	REPU	RJPU(2)	—	—	RTSECSEL1(1)	RTSECSEL0(1)	—
bit 7							bit 0

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

bit 7 **RDPU**：PORTD 上拉使能位
1 = PORTD 上拉电阻通过各端口锁存值使能
0 = 禁止所有 PORTD 上拉电阻

bit 6 **REPU**：PORTE 上拉使能位
1 = PORTE 上拉电阻通过各端口锁存值使能
0 = 禁止所有 PORTE 上拉电阻

bit 5 **RJPU**：PORTJ 上拉使能位 (2)
1 = PORTJ 上拉电阻通过各端口锁存值使能
0 = 禁止所有 PORTJ 上拉电阻

bit 2-1 **RTSECSEL<1:0>**：RTCC 秒时钟输出选择位 (1)
11 = 保留；不使用
10 = 选择从 RTCC 引脚输出 RTCC 源时钟 （引脚可以是 LF-INTOSC 或 SOSC，具体取决于 RTCOSC （CONFIG3L<1>）位设置）
01 = 选择从 RTCC 引脚输出 RTCC 秒时钟
00 = 选择从 RTCC 引脚输出 RTCC 闹钟脉冲

bit 0 **未实现**：读为 0

注 **1**： 要使能实际 RTCC 输出，必须将 RTCOE （RTCCFG<2>）位置 1。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

© 2011 Microchip Technology Inc.

寄存器 18-4： ALRMCFG：闹钟配置寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
bit 7							bit 0

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

bit 7 **ALRMEN：**闹钟使能位
1 = 使能闹钟 （每当 ARPT<7:0> = 00 且 CHIME = 0 时，都会在发生闹钟事件后自动清零）
0 = 禁止闹钟

bit 6 **CHIME：**响铃使能位
1 = 使能响铃；允许 ALRMPTR<1:0> 位从 00h 计满返回到 FFh
0 = 禁止响铃； ALRMPTR<1:0> 达到 00h 后即停止

bit 5-2 **AMASK<3:0>：**闹钟掩码配置位
0000 = 每半秒
0001 = 每秒
0010 = 每 10 秒
0011 = 每分
0100 = 每 10 分
0101 = 每小时
0110 = 每天一次
0111 = 每周一次
1000 = 每月一次
1001 = 每年一次 （配置为 2 月 29 日时除外，该闹钟每 4 年一次）
101x = 保留——不使用
11xx = 保留——不使用

bit 1-0 **ALRMPTR<1:0>：**闹钟值寄存器窗口指针位
读 ALRMVALH 和 ALRMVALL 寄存器时，指向相应的闹钟值寄存器。ALRMPTR<1:0> 值会在每次读 / 写 ALRMVALH 时递减 1，直到达到 00。

ALRMVALH：
00 = ALRMMIN
01 = ALRMWD
10 = ALRMMNTH
11 = 未实现

ALRMVALL：
00 = ALRMSEC
01 = ALRMHR
10 = ALRMDAY
11 = 未实现

寄存器 18-5： ALRMRPT：闹钟重复寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
bit 7							bit 0

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

bit 7-0 **ARPT<7:0>：**闹钟重复计数器值位
11111111 = 闹钟将再重复 255 次
.
.
.
00000000 = 闹钟将不重复
每当发生闹钟事件时计数器就递减 1。除非 CHIME = 1，否则计数器不能从 00h 计满返回到 FFh。

18.1.2 RTCVALH 和 RTCVALL 寄存器映射

寄存器 18-6： 保留寄存器

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

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

bit 7-0 **未实现：**读为 0

寄存器 18-7： YEAR：年份值寄存器 (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
YRTEN3	YRTEN2	YRTEN1	YRTEN0	YRONE3	YRONE2	YRONE1	YRONE0
bit 7							bit 0

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

bit 7-4 **YRTEN<3:0>：**年份的十位数的二十进制码值位
包含从 0 到 9 的值。

bit 3-0 **YRONE<3:0>：**年份的个位数的二十进制码值位
包含从 0 到 9 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入 YEAR 寄存器。

© 2011 Microchip Technology Inc.

寄存器 18-8： MONTH：月份值寄存器 (1)

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 7							bit 0

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

bit 7-5 **未实现：**读为 0

bit 4 **MTHTEN0：**月份的十位数的二十进制码值位
包含值 0 或 1。

bit 3-0 **MTHONE<3:0>：**月份的个位数的二十进制码值位
包含从 0 到 9 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入此寄存器。

寄存器 18-9： DAY：日期值寄存器 (1)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

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

bit 7-6 **未实现：**读为 0

bit 5-4 **DAYTEN<1:0>：**日期的十位数的二十进制码值位
包含从 0 到 3 的值。

bit 3-0 **DAYONE<3:0>：**日期的个位数的二十进制码值位
包含从 0 到 9 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入此寄存器。

寄存器 18-10： WEEKDAY：星期值寄存器 (1)

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 7							bit 0

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

bit 7-3 **未实现：**读为 0

bit 2-0 **WDAY<2:0>：**星期的二十进制码值位
包含从 0 到 6 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入此寄存器。

寄存器 18-11： HOUR：小时值寄存器 (1)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

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

bit 7-6 **未实现：**读为 0

bit 5-4 **HRTEN<1:0>：**小时的十位数的二十进制码值位
包含从 0 到 2 的值。

bit 3-0 **HRONE<3:0>：**小时的个位数的二十进制码值位
包含从 0 到 9 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入此寄存器。

寄存器 18-12： MINUTE：分钟值寄存器

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 7							bit 0

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

bit 7 **未实现：**读为 0

bit 6-4 **MINTEN<2:0>：**分钟的十位数的二十进制码值位
包含从 0 到 5 的值。

bit 3-0 **MINONE<3:0>：**分钟的个位数的二十进制码值位
包含从 0 到 9 的值。

寄存器 18-13： SECOND：秒值寄存器

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

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

bit 7 **未实现：**读为 0

bit 6-4 **SECTEN<2:0>：**秒的十位数的二十进制码值位
包含从 0 到 5 的值。

bit 3-0 **SECONE<3:0>：**秒的个位数的二十进制码值位
包含从 0 到 9 的值。

© 2011 Microchip Technology Inc.

18.1.3 ALRMVALH 和 ALRMVALL 寄存器映射

寄存器 18-14： ALRMMNTH：闹钟月份值寄存器 (1)

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 7							bit 0

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

bit 7-5 **未实现：**读为 0

bit 4 **MTHTEN0：**月份的十位数的二十进制码值位
包含值 0 或 1。

bit 3-0 **MTHONE<3:0>：**月份的个位数的二十进制码值位
包含从 0 到 9 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入此寄存器。

寄存器 18-15： ALRMDAY：闹钟日期值寄存器 (1)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

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

bit 7-6 **未实现：**读为 0

bit 5-4 **DAYTEN<1:0>：**日期的十位数的二十进制码值位
包含从 0 到 3 的值。

bit 3-0 **DAYONE<3:0>：**日期的个位数的二十进制码值位
包含从 0 到 9 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入此寄存器。

寄存器 18-16： ALRMWD：闹钟星期值寄存器 (1)

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 7							bit 0

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

bit 7-3 **未实现：**读为 0

bit 2-0 **WDAY<2:0>：**星期的二十进制码值位
包含从 0 到 6 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入此寄存器。

寄存器 18-17： ALRMHR：闹钟小时值寄存器 (1)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

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

bit 7-6 **未实现：**读为 0

bit 5-4 **HRTEN<1:0>：**小时的十位数的二十进制码值位
包含从 0 到 2 的值。

bit 3-0 **HRONE<3:0>：**小时的个位数的二十进制码值位
包含从 0 到 9 的值。

注 1： 只有在 RTCWREN = 1 时，才允许写入此寄存器。

寄存器 18-18： ALRMMIN：闹钟分钟值寄存器

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 7							bit 0

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

bit 7 **未实现：**读为 0

bit 6-4 **MINTEN<2:0>：**分钟的十位数的二十进制码值位
包含从 0 到 5 的值。

bit 3-0 **MINONE<3:0>：**分钟的个位数的二十进制码值位
包含从 0 到 9 的值。

寄存器 18-19： ALRMSEC：闹钟秒值寄存器

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

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

bit 7 **未实现：**读为 0

bit 6-4 **SECTEN<2:0>：**秒的十位数的二十进制码值位
包含从 0 到 5 的值。

bit 3-0 **SECONE<3:0>：**秒的个位数的二十进制码值位
包含从 0 到 9 的值。

© 2011 Microchip Technology Inc.

18.1.4 RTCEN 位写入

写入 RTCEN 前，必须将 RTCWREN（RTCCFG<5>）置 1。 RTCWREN = 0 时，将忽略对 RTCEN 位的任何写入操作。

与 RTCEN 位相同，只有在 RTCWREN = 1 时才能写 RTCVALH 和 RTCVALL 寄存器。 RTCWREN = 0 时，将忽略对这些寄存器的写入操作。

18.2 工作原理

18.2.1 寄存器接口

RTCC 和闹钟值的寄存器接口是采用二十进制码（BCD）格式实现的。这就在使用该模块时简化了固件，因为每个数字都包含在它自己的 4 位值中了（见图 18-2 和图 18-3）。

图 18-2： 定时器位格式

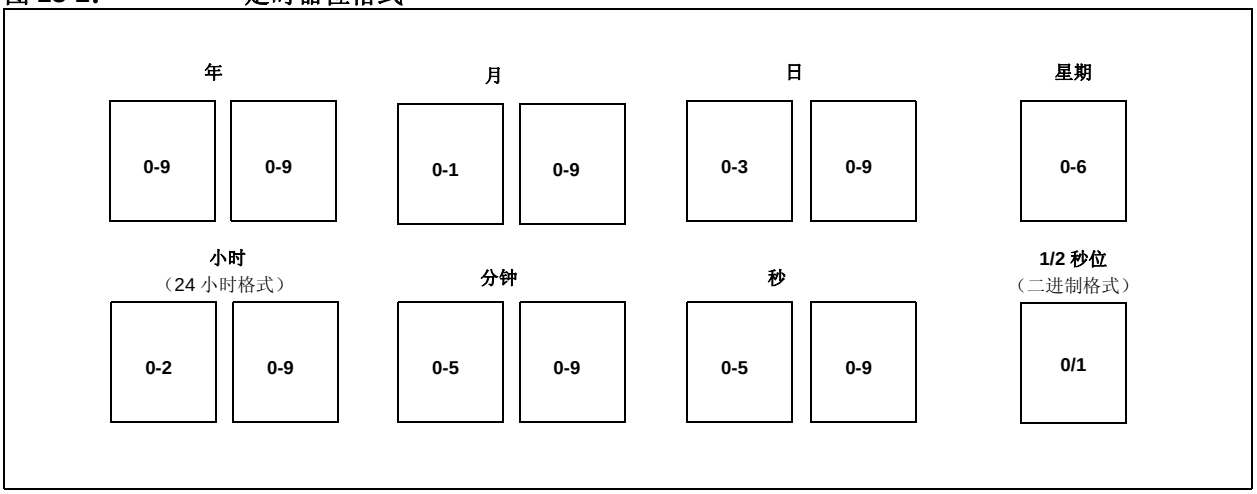
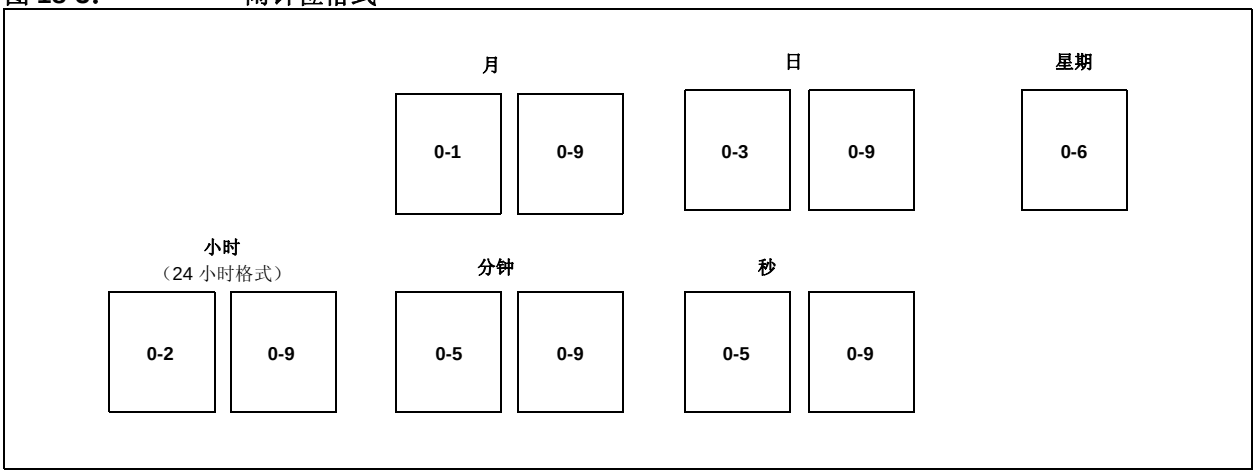


图 18-3： 闹钟位格式

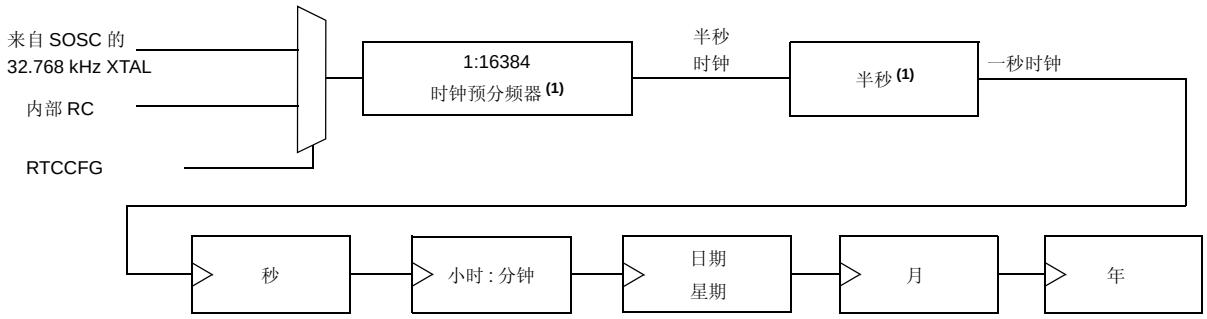


18.2.2 时钟源

如上文所述，RTCC 模块由外部实时时钟（RTC）晶振（振荡频率为 32.768 kHz）提供时钟，但也可以使用内部振荡器。RTCC 时钟选择由 RTCOSC（CONFIG3L<0>）位决定。

晶振校准可以通过此模块完成，产生的误差为每月 3 秒或更小。（欲知更多详情，请参见第 18.2.9 节“校准”。）

图 18-4： 时钟源复用



注 1： 写入 MINSEC 寄存器的低半部分会复位所有计数器，从而允许在几分之一秒内同步； RTCEN = 0 时，时钟预分频器保持在复位状态。

18.2.2.1 实时时钟使能

RTCC 模块可以由外部 32.768 kHz 晶振（SOSC 振荡器）或 LF-INTOSC 振荡器提供时钟，振荡器可以用 CONFIG3L<0> 选择。

如果使用外部时钟，应该使能 SOSC 振荡器。如果正由 LF-INTOSC 提供时钟，可以通过 RTSECSEL<1:0> 位（PADCFG<2:1>）将 INTOSC 时钟提供给 RTCC 引脚。

18.2.3 进位规则

本节介绍计满返回时受影响的定时器值：

- 日时间：从 23:59:59 到 00:00:00，向日字段进位
- 月：从 12/31 到 01/01，向年字段进位
- 星期：从 6 到 0，无进位（见表 18-1）
- 年进位：从 99 到 00；这也标志着 RTCC 使用的终结

有关日到月的进位设定，请参见表 18-2。

考虑到以下值是 BCD 格式，进位到 BCD 的高位将在计数为 10 时发生，而不是在计数为 16 时发生（SECONDS、MINUTES、HOURS、WEEKDAY、DAYS 和 MONTHS）。

表 18-1： 星期设定

星期	
星期日	0
星期一	1
星期二	2
星期三	3
星期四	4
星期五	5
星期六	6

表 18-2： 日到月计满返回设定

月	最大日字段
01（1 月）	31
02（2 月）	28 或 29(1)
03（3 月）	31
04（4 月）	30
05（5 月）	31
06（6 月）	30
07（7 月）	31
08（8 月）	31
09（9 月）	30
10（10 月）	31
11（11 月）	30
12（12 月）	31

注 1： 请参见第 18.2.4 节“闰年”。

© 2011 Microchip Technology Inc.

18.2.4 闰年

由于 RTCC 模块的年份范围是 2000 到 2099，闰年是通过以上范围内的年份能否被 4 整除来确定的。闰年中惟一受影响的月份是二月。

在闰年中，二月将有 29 天，而在其他年份中，为 28 天。

18.2.5 常规功能

所有包含秒时间值或更大的时间值的定时器寄存器都是可写的。用户通过寄存器指针向定时器寄存器写所需年份、月份、日、小时、分钟和秒来配置时间。（见**第 18.2.8 节 “寄存器映射”**。）

定时器使用新写入的值从所需起始点继续计数。

RTCC 通过将 RTCEN 位 （RTCCFG<7>）位置 1 来使能。如果定时器在调节这些寄存器时使能，则定时器值仍将继续递增。但是，只要写入 MINSEC 寄存器，两个定时器预分频器都会复位为 0。这就允许在几分之一秒内同步。

在 CPU 执行写指令的同时更新定时器寄存器。用户必须确保，在 RTCEN = 1 时，不会同时递增更新的寄存器。这可以通过多种方式实现：

- 通过检查 RTCSYNC 位 （RTCCFG<4>）
- 通过检查前面可能发生进位的位
- 通过在秒脉冲 （或闹钟中断）之后立即更新寄存器

用户能够看到计数器的半秒字段。此值为只读，只能通过写入 SECONDS 寄存器的后半部分复位。

18.2.6 寄存器读写的安全窗口

RTCSYNC 位指示在不考虑进位的情况下可以安全读写 RTCC 时钟域寄存器的时间窗口。 RTCSYNC = 0 时，CPU 可以安全访问这些寄存器。

无论 RTCSYNC = 1 还是 0，用户都应采用固件方法来确保不在进位的边界进行数据读取，这样会产生无效读取或部分读取。该固件方法为读取每个寄存器两遍，然后比较两个值。如果两个值匹配，则表示没有发生进位。

18.2.7 写锁定

为了对任意 RTCC 定时器寄存器进行写入，必须将 RTCWREN 位 （RTCCFG<5>）置 1。

为了避免意外写入 RTCC 定时器寄存器，建议在不写入寄存器时，保持 RTCWREN 位（RTCCFG<5>）清零。要将 RTCWREN 位置 1，在 55h/AA 序列和 RTCWREN 置 1 之间，只允许一个指令周期的时间窗口。为此，建议用户遵循例 18-1 中的代码示例。

例 18-1： 将 RTCWREN 位置 1

```
movlw   0x55
movwf   EECON2
movlw   0xAA
movwf   EECON2
bsf     RTCCFG,RTCWREN
```

18.2.8 寄存器映射

为了限制寄存器接口，通过相应的寄存器指针访问 RTCC 定时器和闹钟定时器寄存器。RTCC 值寄存器窗口 （RTCVALH 和 RTCVALL）使用 RTCPTRx 位（RTCCFG<1:0>）来选择所需定时器寄存器对。

通过读 / 写 RTCVALH 寄存器，RTCC 指针值（RTCPTR<1:0>）递减 1，直到其达到 00。达到 00 时，MINUTES 和 SECONDS 值可通过 RTCVALH 和 RTCVALL 进行访问，直到手动更改指针值为止。

表 18-3： RTCVALH 和 RTCVALL 寄存器映射

RTCPTR<1:0>	RTCC 值寄存器窗口	
	RTCVALH	RTCVALL
00	MINUTES	SECONDS
01	WEEKDAY	HOURS
10	MONTH	DAY
11	—	YEAR

闹钟值寄存器窗口 （ALRMVALH 和 ALRMVALL）使用 ALRMPTR 位（ALRMCFG<1:0>）选择所需的闹钟寄存器对。

通过读 / 写 ALRMVALH 寄存器，闹钟指针值（ALRMPTR<1:0>）递减 1，直到其达到 00。达到 00 时， ALRMMIN 和 ALRMSEC 值可通过 ALRMVALH 和 ALRMVALL 进行访问，直到手动更改指针值为止。

© 2011 Microchip Technology Inc.

表 18-4： ALRMVAL 寄存器映射

ALRMPTR<1:0>	闹钟值寄存器窗口	
	ALRMVALH	ALRMVALL
00	ALRMMIN	ALRMSEC
01	ALRMWD	ALRMHR
10	ALRMMNTH	ALRMDAY
11	—	—

18.2.9 校准

实时晶振输入可以使用周期性自动调节功能进行校准。正确校准后， RTCC 可提供每月小于 3 秒的误差。

要执行此校准，需要找到误差时钟脉冲数，并将该值存储到 RTCCAL 寄存器的低半部分。装入 RTCCAL 的 8 位有符号值乘以 4，每分钟一次将其加到 RTCC 定时器中或从 RTCC 定时器中减去。

要校准 RTCC 模块：

1. 使用器件上另一个定时器资源来查找32.768 kHz 晶振的误差。
2. 转换每分钟误差时钟脉冲数 （见公式 18-1）。

公式 18-1： 转换误差时钟脉冲

(理想频率 (32,768) – 测得频率)* 60 =
每分钟误差时钟

- 如果振荡器的速度*快于*理想速度（步骤 2 中得到负值结果），那么 RCFGCALL 寄存器值必需是负值。这将导致每分钟从定时器计数器中减去指定的时钟脉冲数。
- 如果振荡器的速度*慢于*理想速度（步骤 2 中得到正值结果），那么 RCFGCALL 寄存器值必需是正值。这将导致每分钟在定时器计数器中加上指定的时钟脉冲数。

3. 使用正确的值装入 RTCCAL 寄存器。

只有在关闭定时器或紧接着秒脉冲的上升沿时，才能对 RTCCAL 寄存器进行写入。

> **注：** 在确定晶振误差值时，是否包含由温度或晶振老化而产生的漂移引起的晶振初始误差，由用户自行决定。

18.3 闹钟

闹钟功能和特性如下：

- 可配置为半秒到一年
- 使用 ALRMEN 位 （ALRMCFG<7>，寄存器 18-4）使能
- 提供一次性和重复闹钟选项

18.3.1 配置闹钟

闹钟功能使用 ALRMEN 位使能。发出闹钟时，此位清零。如果 CHIME 位 = 1 或 ALRMRPT ≠ 0，将不清零该位。

闹钟的时间间隔选择通过 ALRMCFG 位（AMASK<3:0>）配置 （见图 18-5）。这些位决定了要触发闹钟，闹钟的哪些位以及多少位必须和时钟值匹配。

可以配置闹钟使之根据预先配置的间隔时间重复。使能闹钟后，闹钟发生的次数存储在 ALRMRPT 寄存器中。

> **注：** 使能闹钟（ALRMEN = 1）时，对除 RTCCAL、ALRMCFG 和 ALRMRPT 寄存器之外的任何寄存器和CHIME位进行更改会导致误闹钟事件，进而导致错误的闹钟中断。为避免发生这种情况，在禁止闹钟（ALRMEN = 0）时只能更改定时器和闹钟值。建议在 RTCSYNC = 0 时更改 ALRMCFG 和ALRMRPT 寄存器和 CHIME 位。

© 2011 Microchip Technology Inc.

图 18-5： 闹钟掩码设置

闹钟掩码设置 AMASK<3:0>	星期	月	日	小时	分钟	秒
0000 ——每半秒 0001 ——每秒						
0010 ——每 10 秒						s
0011 ——每分钟						s s
0100 ——每 10 分钟					m	s s
0101 ——每小时					m m	s s
0110 ——每天				h h	m m	s s
0111 ——每周	d			h h	m m	s s
1000 ——每月			d d	h h	m m	s s
1001 ——每年 (1)		m m	d d	h h	m m	s s

注 1： 每年，配置为 2 月 29 日除外。

ALRMCFG = 00 且 CHIME 位 （ALRMCFG<6>） = 0 时，禁止重复功能，仅发出单次闹钟。通过将 FFh 装入 ALRMRPT 寄存器，闹钟最多可以重复 255 次。

每次发出闹钟后， ALRMRPT 寄存器都将减 1。寄存器达到 00 后，将最后一次发出闹钟。

最后一次发出闹钟后，ALRMEN 位将自动清零，闹钟关闭。只有 CHIME 位 = 1 时才能无限次重复发出闹钟。

CHIME = 1 且 ALRMRPT 寄存器达到 00 不会禁止闹钟，而是计满返回到 FF，并继续无限期进行计数。

18.3.2 闹钟中断

每次发生闹钟事件时，都将产生中断。此外，将提供闹钟脉冲输出，其频率是闹钟频率的一半。

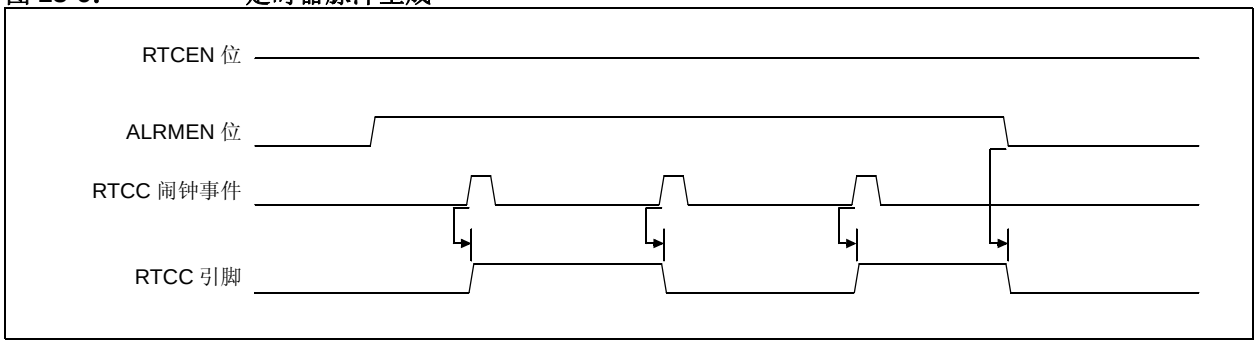
闹钟脉冲输出与 RTCC 时钟完全同步，可用作其他外设的触发时钟。此输出在 RTCC 引脚上可用。输出脉冲是占空比为 50% 且频率为闹钟事件一半的时钟（见图 18-6）。

RTCC 引脚还可以输出秒时钟。用户可以在 RTCC 模块生成的闹钟脉冲和秒时钟输出之间选择。

RTSECSEL<1:0> 位 （PADCFG1<2:1>）用于在以下两个输出之间选择：

- 闹钟脉冲——RTSECSEL<1:0> = 00
- 秒时钟——RTSECSEL<1:0> = 01

图 18-6： 定时器脉冲生成



18.4 休眠模式

定时器和闹钟可以在休眠模式下继续工作。闹钟的运行不受休眠模式的影响，因为闹钟事件总是能唤醒 CPU。

空闲模式不影响定时器或闹钟的工作。

18.5 复位

18.5.1 器件复位

发生器件复位时，将强制 ALRMRPT 寄存器进入其复位状态，从而导致禁止闹钟 （如果在复位前使能）。如果使能了 RTCC，它将在发生基本器件复位时继续工作。

18.5.2 上电复位 （POR）

RTCCFG 和 ALRMRPT 寄存器仅在上电复位时复位。器件退出 POR 状态后，时钟寄存器应该重新装入所需值。

定时器预分频器值只能通过写入 SECONDS 寄存器复位。没有器件复位能够影响预分频器。

© 2011 Microchip Technology Inc.

18.6 寄存器映射

表 18-5、表 18-6 和表 18-7 汇总了与 RTCC 模块关联的寄存器。

表 18-5： RTCC 控制寄存器

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RTCCFG	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0
RTCCAL	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
PADCFG1	RDPU	REPU	RJPU(1)	—	—	RTSECSEL1	RTSECSEL0	—
ALRMCFG	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
ALRMRPT	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
PMD1	PSPMD	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBDM

图注： — = 未实现，读为 0。 80 引脚器件的复位值以十六进制形式显示。

注 1： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

表 18-6： RTCC 值寄存器

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RTCVALH	基于 RTCPTR<1:0> 的 RTCC 值高位寄存器窗口							
RTCVALL	基于 RTCPTR<1:0> 的 RTCC 值低位寄存器窗口							

表 18-7： 闹钟值寄存器

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ALRMVALH	基于 ALRMPTR<1:0> 的闹钟值高位寄存器窗口							
ALRMVALL	基于 ALRMPTR<1:0> 的闹钟值低位寄存器窗口							

© 2011 Microchip Technology Inc.

注：

© 2011 Microchip Technology Inc.

19.0 捕捉 / 比较 /PWM （CCP）模块

PIC18F87K22系列器件具有七个CCP（捕捉/比较/PWM）模块，命名为 CCP4 到 CCP10。所有模块均实现标准的捕捉、比较和脉宽调制（Pulse-Width Modulation，PWM）模式。

> **注：** 在本章节中，对于相同的寄存器和位名均采用统称，其中 x 变量表示与特定 CCP 模块相关的项。例如，控制寄存器命名为 CCPxCON，其指代 CCP4CON 至 CCP10CON。

每个 CCP 模块都有一个 16 位寄存器，它可以用作 16 位捕捉寄存器、16 位比较寄存器或 PWM 主 / 从占空比寄存器。为了清楚起见，以下章节中描述的所有 CCP 模块操作均是针对 CCP4 的，但同样也适用于 CCP5 至 CCP10。

> **注：** CCP9 和 CCP10 模块在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上被禁止。

寄存器 19-1： CCPxCON：CCPx 控制寄存器 （CCP4-CCP10 模块）(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
PxM1	PxM0	DCxB1	DCxB0	CCPxM3(2)	CCPxM2(2)	CCPxM1(2)	CCPxM0(2)
bit 7							bit 0

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

bit 7-6 **PxM<1:0>：**PWM 输出配置位

如果 CCPxM<3:2> = 00、 01 或 10：

xx = PxA 被指定为捕捉 / 比较输入 / 输出； PxB、 PxC 和 PxD 被指定为端口引脚

如果 CCPxM<3:2> = 11：

00 = 单输出：PxA、 PxB、 PxC 和 PxD 通过转向控制

01 = 全桥正向输出：PxD 被调制； PxA 有效； PxB 和 PxC 无效

10 = 半桥输出：PxA 和 PxB 为带死区控制的调制输出； PxC 和 PxD 被指定为端口引脚

11 = 全桥反向输出：PxB 被调制； PxC 有效； PxA 和 PxA 无效

bit 5-4 **DCxB<1:0>：**CCPx 模块的 PWM 占空比位 bit 1 和 bit 0

捕捉模式：

未用。

比较模式：

未用。

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。

注 **1：** CCP9 和 CCP10 模块在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上不可用。

2： 如果 CCPxM<3:0> = 1011，在发生 CCPx 匹配时将只复位定时器而不启动 A/D 转换。

寄存器 19-1： CCPxCON：CCPx 控制寄存器 （CCP4-CCP10 模块）[1] （续）

bit 3-0 **CCPxM<3:0>**：CCPx 模块模式选择位 [2]

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：每个下降沿捕捉
0101 = 捕捉模式：每个上升沿捕捉
0110 = 捕捉模式：每 4 个上升沿捕捉
0111 = 捕捉模式：每 16 个上升沿捕捉
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：发生 CCPx 匹配时，触发特殊事件并复位定时器 （CCPxIF 位置 1）
11xx = PWM 模式

注 1： CCP9 和 CCP10 模块在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上不可用。
2： 如果 CCPxM<3:0> = 1011，在发生 CCPx 匹配时将只复位定时器而不启动 A/D 转换。

寄存器 19-2： CCPTMRS1：CCP 定时器选择寄存器 1

R/W-0	R/W-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
C7TSEL1	C7TSEL0	—	C6TSEL0	—	C5TSEL0	C4TSEL1	C4TSEL0
bit 7							bit 0

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

bit 7-6 **C7TSEL<1:0>**：CCP7 定时器选择位
00 = CCP7 基于 TMR1/TMR2 工作
01 = CCP7 基于 TMR5/TMR4 工作
10 = CCP7 基于 TMR5/TMR6 工作
11 = CCP7 基于 TMR5/TMR8 工作

bit 5 **未实现**：读为 0

bit 4 **C6TSEL0**：CCP6 定时器选择位
0 = CCP6 基于 TMR1/TMR2 工作
1 = CCP6 基于 TMR5/TMR2 工作

bit 3 **未实现**：读为 0

bit 2 **C5TSEL0**：CCP5 定时器选择位
0 = CCP5 基于 TMR1/TMR2 工作
1 = CCP5 基于 TMR5/TMR4 工作

bit 1-0 **C4TSEL<1:0>**：CCP4 定时器选择位
00 = CCP4 基于 TMR1/TMR2 工作
01 = CCP4 基于 TMR3/TMR4 工作
10 = CCP4 基于 TMR3/TMR6 工作
11 = 保留；未用

© 2011 Microchip Technology Inc.

寄存器 19-3： CCPTMRS2：CCP 定时器选择寄存器 2

U-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	C10TSEL0[1]	—	C9TSEL0[1]	C8TSEL1	C8TSEL0
bit 7							bit 0

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

bit 7-5 **未实现：**读为 0

bit 4 **C10TSEL0：**CCP10 定时器选择位 [1]
0 = CCP10 基于 TMR1/TMR2 工作
1 = CCP10 基于 TMR7/TMR2 工作

bit 3 **未实现：**读为 0

bit 2 **C9TSEL0：**CCP9 定时器选择位 [1]
0 = CCP9 基于 TMR1/TMR2 工作
1 = CCP9 基于 TMR7/TMR4 工作

bit 1-0 **C8TSEL<1:0>：**CCP8 定时器选择位
在非 32 字节器件中：
00 = CCP8 基于 TMR1/TMR2 工作
01 = CCP8 基于 TMR7/TMR4 工作
10 = CCP8 基于 TMR7/TMR6 工作
11 = 保留；未用
在 32 字节器件中 （PIC18F65K22 和 PIC18F85K22）：
00 = CCP8 基于 TMR1/TMR2 工作
01 = CCP8 基于 TMR1/TMR4 工作
10 = CCP8 基于 TMR1/TMR6 工作
11 = 保留；未用

注 1： 该位在具有 32 KB 程序存储器的器件 （PIC18FX5K22）上未实现并读为 0。

寄存器 19-4： CCPRxL：CCPx 周期寄存器低字节

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 = 上电复位时的值	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：捕捉寄存器低字节
比较模式：比较寄存器低字节
PWM 模式：占空比寄存器

寄存器 19-5： CCPRxH：CCPx 周期寄存器高字节

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 = 上电复位时的值	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：捕捉寄存器高字节
比较模式：比较寄存器高字节
PWM 模式：占空比缓冲寄存器

© 2011 Microchip Technology Inc.

19.1 CCP 模块配置

每个捕捉 / 比较 /PWM 模块均与一个控制寄存器（通常为 CCPxCON）和一个数据寄存器（CCPRx）相关联。而数据寄存器由 2 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 模块根据选定的工作模式使用定时器 1 到 8。CCP 模块在捕捉、比较或 PWM 模式下可用的各种定时器，如表 19-1 所示。

表 19-1： CCP 模式与定时器资源

CCP 模式	定时器资源
捕捉	Timer1、 Timer3、 Timer5 或 Timer7
比较	
PWM	Timer2、 Timer4、 Timer6 或 Timer8

要将哪个特定的定时器分配给 CCP 模块是由 CCPTMRSx 寄存器中的 CCP 定时器使能位决定的（见寄存器 19-2 和寄存器 19-3）。如果将所有模块配置为在相同的模式（捕捉 / 比较或 PWM）下工作，那么这些模块可同时处于工作状态且可共用相同的定时器资源。

CCPTMRS1 寄存器为 CCP 模块 4、 5、 6 和 7 选择定时器，而 CCPTMRS2 寄存器为 CCP 模块 8、 9 和 10 选择定时器。表 19-2 和表 19-3 列出了可能的配置。

表 19-2： CCP 模块 4、 5、 6 和 7 的定时器分配

CCPTMRS1 寄存器											
CCP4			CCP5			CCP6			CCP7		
C4TSEL <1:0>	捕捉 / 比较模式	PWM 模式	C5TSEL0	捕捉 / 比较模式	PWM 模式	C6TSEL0	捕捉 / 比较模式	PWM 模式	C7TSEL <1:0>	捕捉 / 比较模式	PWM 模式
0 0	TMR1	TMR2	0	TMR1	TMR2	0	TMR1	TMR2	0 0	TMR1	TMR2
0 1	TMR3	TMR4	1	TMR5	TMR4	1	TMR5	TMR2	0 1	TMR5	TMR4
1 0	TMR3	TMR6							1 0	TMR5	TMR6
1 1	保留 (1)								1 1	TMR5	TMR8

注 1： 保留位未使用。

表 19-3： CCP 模块 8、 9 和 10 的定时器分配

CCPTMRS2 定时器											
CCP8			CCP8 具有 32 KB 程序存储器的器件			CCP9(1)			CCP10(1)		
C8TSEL <1:0>	捕捉 / 比较模式	PWM 模式	C8TSEL <1:0>	捕捉 / 比较模式	PWM 模式	C9TSEL0	捕捉 / 比较模式	PWM 模式	C10TSEL0	捕捉 / 比较模式	PWM 模式
0 0	TMR1	TMR2	0 0	TMR1	TMR2	0	TMR1	TMR2	0	TMR1	TMR2
0 1	TMR7	TMR4	0 1	TMR1	TMR4	1	TMR7	TMR4	1	TMR7	TMR2
1 0	TMR7	TMR6	1 0	TMR1	TMR6						
1 1	保留 (2)		1 1	保留 (2)							

注 1： 此模块在具有 32 KB 程序存储器的器件（PIC18F65K22 和 PIC18F85K22）上不可用。
2： 不要使用保留设置。

19.1.2 漏极开路输出选项

在输出模式（即比较或 PWM 模式）下工作时，可将 CCPx 引脚的驱动器配置为漏极开路输出。该功能允许通过外部上拉电阻将 CCPx 引脚上的电平拉为高电平，并允许漏极开路输出与外部电路通信，而无需额外的电平转换器。

漏极开路输出选项由 CCPxOD 位（ODCON2<7:0>）控制。通过置 1 适当的位来配置相应模块的引脚以执行漏极开路操作。

19.1.3 CCP6、CCP7、CCP8 和 CCP9 引脚分配

可根据器件配置改变 CCP6/7/8/9 的引脚分配（捕捉输入、比较和 PWM 输出）。

ECCPMX 配置位（CONFIG3H<1>）决定哪个引脚与 CCP6/7/8/9 复用。表 19-4 给出了这些 CCP 模块的引脚分配。

表 19-4： CCP 引脚分配

ECCPMX 值	引脚映射到			
	CCP6	CCP7	CCP8	CC9
1（默认值）	RE6	RE5	RE4	RE3
0	RH7	RH6	RH5	RH4

19.2 捕捉模式

在捕捉模式下，当 CCP4 引脚上有事件发生时，CCPR4H:CCPR4L 寄存器对将捕捉 CCPTMRS1 中选择的定时器寄存器的 16 位值。事件定义为下列情况之一：

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

通过模式选择位 CCP4M<3:0>（CCP4CON<3:0>）选择事件。当一次捕捉完成时，中断请求标志位 CCP4IF（PIR4<1>）置 1；它必须由软件清零。如果在读取 CCPR4 的值之前发生了另一次捕捉，那么之前捕捉的值将被新捕捉的值覆盖。

图 19-1 给出了捕捉模式框图。

19.2.1 CCP 引脚配置

在捕捉模式下，应该通过将对应的 TRIS 方向位置 1 来将 CCPx 引脚配置为输入。

> **注：** 如果 RC1 或 RE7 被配置为 CCP4 输出，对该端口的写操作会产生捕捉条件。

19.2.2 TIMER1/3/5/7 模式选择

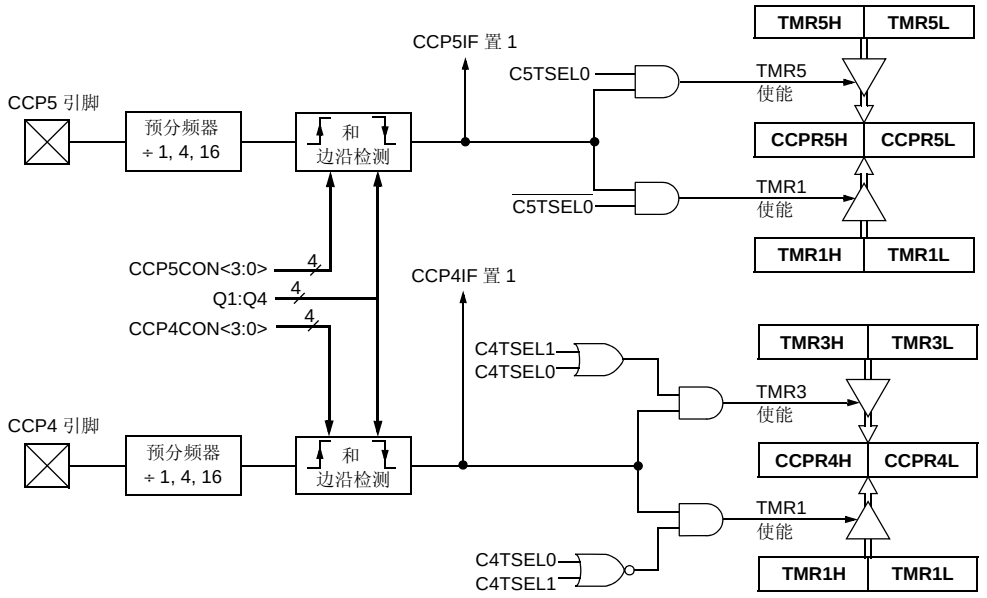
用于捕捉功能的定时器（1/3/5/7）必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，无法进行捕捉操作。

用于每个 CCP 模块的定时器由 CCPTMRSx 寄存器选择（见**第 19.1.1 节 “CCP 模块和定时器资源”**）。

表 19-2 和表 19-3 给出了 CCP 模块的定时器分配的详细信息。

© 2011 Microchip Technology Inc.

图 19-1： 捕捉模式工作原理框图



注： 本框图以 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 和表 19-3。

19.2.3 软件中断

当捕捉模式改变时，可能会产生错误捕捉中断。用户应该保持 CCP4IE （PIE4<1>）中断允许位清零，以避免错误中断，还应该在工作模式发生任何改变之后清零中断标志位 CCP4IF。

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（CCP4M<3:0>）选择。当关闭 CCP 模块或 CCP 模块不处于捕捉模式下时，预分频计数器就会被清零。这意味着任何复位都将清零预分频计数器。

从一种捕捉预分频比切换到另一种捕捉预分频比可能会产生中断，且不会清零预分频计数器；因此切换后的第一次捕捉可能来自于一个非零的预分频器。

例 19-1 是切换捕捉预分频比时建议采用的方法。该示例使预分频计数器清零且不会产生 “误” 中断。

例 19-1： 改变捕捉预分频比

```
CLRF   CCP4CON      ; Turn CCP module off
MOVLW  NEW_CAPT_PS  ; Load WREG with the
                    ; new prescaler mode
                    ; value and CCP ON
MOVWF  CCP4CON      ; Load CCP4CON with
                    ; this value
```

19.3 比较模式

在比较模式下， 16 位 CCPR4 寄存器的值将不断与 CCPTMR1 寄存器中选择的定时器寄存器对的值相比较。当两者匹配时， CCP4 引脚将被：

- 驱动为高电平
- 驱动为低电平
- 翻转电平 （高电平变为低电平或低电平变为高电平）
- 保持不变 （即反映 I/O 锁存器的状态）

引脚动作取决于模式选择位（CCP4M<3:0>）的值。同时，中断标志位 CCP4IF 置 1。

图 19-2 给出了比较模式框图。

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 CCP4CON 寄存器会将 RC1 或 RE7 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 PORTC 或 PORTE I/O 数据锁存器。

19.3.2 TIMER1/3/5/7 模式选择

CCP 模块使用比较功能并采用 Timer1/3/5/7 定时器时，这些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，无法进行比较操作。

注：	表 19-2 和表 19-3 给出了 CCP 模块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时（CCP4M<3:0> = `1010`），对应的 CCP4 引脚不受影响。如果中断被允许（CCP4IE 位置 1），将只产生一个 CCP 中断。

19.3.4 特殊事件触发信号

两个 CCP 模块都配备一个特殊事件触发器。产生的触发信号为内部硬件信号，是在比较模式下产生以触发其他模块的动作。通过选择比较特殊事件触发模式（CCP4M<3:0> = `1011`）使能特殊事件触发器。

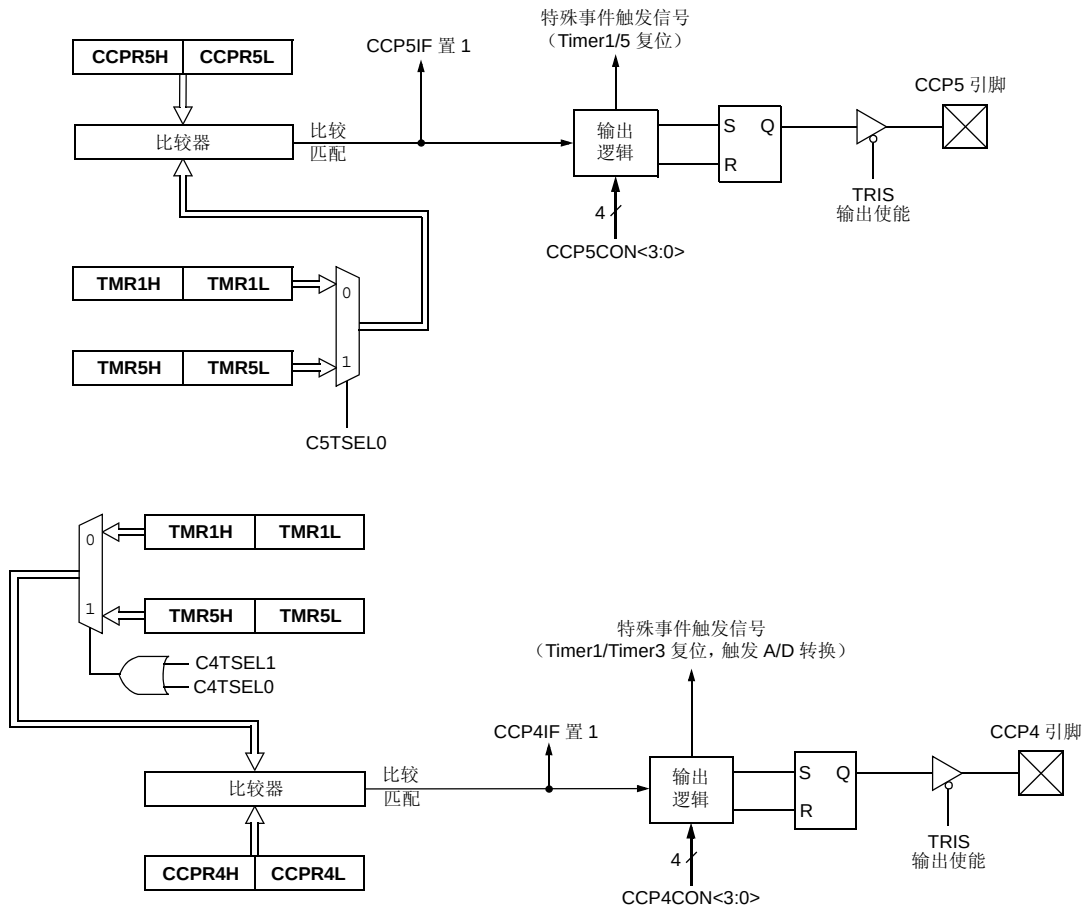
对于任何一个 CCP 模块，无论当前使用哪个定时器资源作为模块的时基，特殊事件触发信号都将把对应的定时器寄存器对复位。这样 CCPRx 寄存器就可用作任一个定时器的可编程周期寄存器。

CCP4 的特殊事件触发信号不能启动 A/D 转换。

注：	ECCP2 的特殊事件触发信号会启动 A/D 转换，但需要使能 A/D转换器。关于更多信息，请参见**第 19.0 节 “捕捉 / 比较 /PWM（CCP）模块”**。

© 2011 Microchip Technology Inc.

图 19-2： 比较模式工作原理框图



注： 本框图以 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 和表 19-3。

表 19-5： 与捕捉、比较和 TIMER1/3/5/7 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	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
PIR4	CCP10IF(1)	CCP9IF(1)	CCP8IF	CCP7IF	CCP6IF	CCP5IF	CCP4IF	CCP3IF
PIE4	CCP10IE(1)	CCP9IE(1)	CCP8IE	CCP7IE	CCP6IE	CCP5IE	CCP4IE	CCP3IE
IPR4	CCP10IP(1)	CCP9IP(1)	CCP8IP	CCP7IP	CCP6IP	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	TRISE3	TRISE2	TRISE1	TRISE0
TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0
TMR1L	Timer1 寄存器的低字节							

图注： — = 未实现，读为 0。捕捉 / 比较或 Timer1/3/5/7 不使用阴影单元。

注 1： 在具有 32 KB 程序存储器的器件 （PIC18F65K22 和 PIC18F85K22）上未实现。

2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

表 19-5： 与捕捉、比较和 TIMER1/3/5/7 相关的寄存器 （续）

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
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							
TMR5L	Timer5 寄存器的低字节							
TMR5H	Timer5 寄存器的高字节							
TMR7L(1)	Timer7 寄存器的低字节							
TMR7H(1)	Timer7 寄存器的高字节							
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON
T5CON	TMR5CS1	TMR5CS0	T5CKPS1	T5CKPS0	SOSCEN	$\overline{\text{T5SYNC}}$	RD16	TMR5ON
T7CON(1)	TMR7CS1	TMR7CS0	T7CKPS1	T7CKPS0	SOSCEN	$\overline{\text{T7SYNC}}$	RD16	TMR7ON
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							
CCPR6L	捕捉 / 比较 /PWM 寄存器 6 的低字节							
CCPR6H	捕捉 / 比较 /PWM 寄存器 6 的高字节							
CCPR7L	捕捉 / 比较 /PWM 寄存器 7 的低字节							
CCPR7H	捕捉 / 比较 /PWM 寄存器 7 的高字节							
CCPR8L	捕捉 / 比较 /PWM 寄存器 8 的低字节							
CCPR8H	捕捉 / 比较 /PWM 寄存器 8 的高字节							
CCPR9L(1)	捕捉 / 比较 /PWM 寄存器 9 的低字节							
CCPR9H(1)	捕捉 / 比较 /PWM 寄存器 9 的高字节							
CCPR10L(1)	捕捉 / 比较 /PWM 寄存器 10 的低字节							
CCPR10H(1)	捕捉 / 比较 /PWM 寄存器 10 的高字节							
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0
CCP6CON	—	—	DC6B1	DC6B0	CCP6M3	CCP6M2	CCP6M1	CCP6M0
CCP7CON	—	—	DC7B1	DC7B0	CCP7M3	CCP7M2	CCP7M1	CCP7M0
CCP8CON	—	—	DC8B1	DC8B0	CCP8M3	CCP8M2	CCP8M1	CCP8M0
CCP9CON(1)	—	—	DC9B1	DC9B0	CCP9M3	CCP9M2	CCP9M1	CCP9M0
CCP10CON(1)	—	—	DC10B1	DC10B0	CCP10M3	CCP10M2	CCP10M1	CCP10M0
CCPTMRS1	C7TSEL1	C7TSEL0	—	C6TSEL0	—	C5TSEL0	C4TSEL1	C4TSEL0
CCPTMRS2	—	—	—	C10TSEL0(1)	—	C9TSEL0(1)	C8TSEL1	C8TSEL0
PMD3	CCP10MD(1)	CCP9MD(1)	CCP8MD	CCP7MD	CCP6MD	CCP5MD	CCP4MD	TMR12MD(1)

图注： — = 未实现，读为 0。捕捉 / 比较或 Timer1/3/5/7 不使用阴影单元。

注 1： 在具有 32 KB 程序存储器的器件 （PIC18F65K22 和 PIC18F85K22）上未实现。

2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

© 2011 Microchip Technology Inc.

19.4 PWM 模式

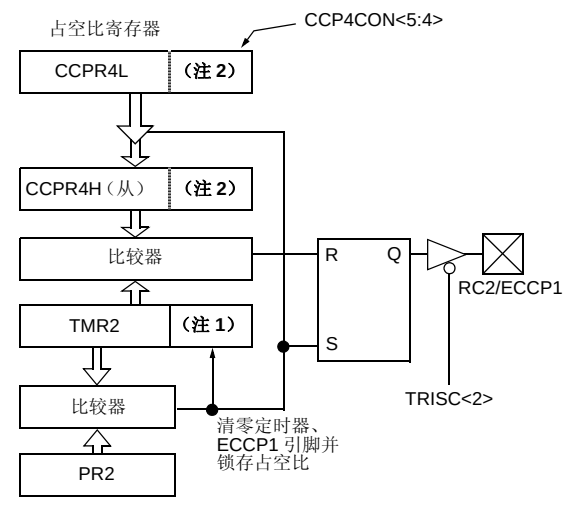
在脉宽调制（PWM）模式下，CCP4 引脚会产生最高 10 位分辨率的 PWM 输出信号。由于 CCP4 引脚与 PORTC 或 PORTE 数据锁存器复用，因此必须清零相应的 TRIS 位才能使 CCP4 引脚用作输出引脚。

> **注：** 清零 CCP4CON 寄存器会将 RC1 或 RE7 输出锁存器（取决于器件配置）强制为默认的低电平。这不是 PORTC 或 PORTE I/O 数据锁存器。

图 19-3 所示为在 PWM 模式下的 ECCP1 模块的简化框图。

如需了解设置 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： PWM 简化框图

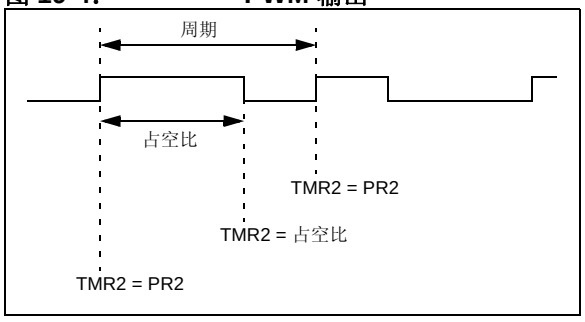


注 1： 8 位 TMR2 值与 2 位内部 Q 时钟或预分频器的 2 位一起构成 10 位时基。

2： 此处以 CCP4 及其相应定时器为例。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 和表 19-3。

PWM 输出信号（图 19-4）包含一个时基（周期）和一段输出保持为高电平的时间（占空比）。PWM 频率为周期的倒数（1/ 周期）。

图 19-4： PWM 输出



19.4.1 PWM 周期

通过写 PR2 寄存器来指定 PWM 周期。可以使用以下公式来计算 PWM 周期：

公式 19-1：

$$\text{PWM 周期} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

PWM 频率定义为 1/[PWM 周期]。

当 TMR2 等于 PR2 时，在下一递增周期中发生以下三个事件：

- TMR2 被清零
- CCP4 引脚被置 1
  （例外情况：如果 PWM 占空比 = 0%，则 CCP4 引脚不被置 1）
- PWM 占空比从 CCPR4L 锁存到 CCPR4H

> **注：** 确定 PWM 频率时不会用到 Timer2 后分频器（见**第 15.0 节 “Timer2 模块”**）。使用后分频器时，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

PWM 占空比可通过写 CCPR4L 寄存器和 CCP4CON<5:4> 位来指定（以 CCP4 为例）。最高分辨率可达 10 位。CCPR4L 包含占空比的高 8 位，而 CCP4CON<5:4> 包含低 2 位。CCPR4L:CCP4CON<5:4> 表示这个 10 位值。下面的公式用于计算 PWM 占空比（以时间为单位）。

公式 19-2：

$$\text{PWM 占空比} = (\text{CCPR4L:CCP4CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

可在任何时候写入 CCPR4L 和 CCP4CON<5:4>，但直到 PR2 与 TMR2 中的值匹配（即周期结束）时，占空比的值才被锁存到 CCPR4H。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为 PWM 占空比提供双重缓冲。这种双重缓冲结构非常重要，可以避免在 PWM 操作中产生毛刺。

当 CCPR4H 和 2 位锁存器的值与 TMR2（加上内部 2 位 Q 时钟或 TMR2 预分频器的 2 位）匹配时，CCP4 引脚被清零。

在给定 PWM 频率的情况下，其最大 PWM 分辨率（位）由公式 19-3 计算：

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 占空比值大于 PWM 周期，将不会清零 CCP4 引脚。

表 19-6： 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 或 16）	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最高分辨率（位）	10	10	10	8	7	6.58

19.4.3 设置 PWM 操作

在将 CCP 模块配置为 PWM 操作时应该遵循以下步骤（以 CCP4 为例）：

1. 通过写 PR2 寄存器设置 PWM 周期。
2. 通过写入 CCPR4L 寄存器和 CCP4CON<5:4> 位来设置 PWM 占空比。
3. 通过清零相应的 TRIS 位将 CCP4 引脚配置为输出。
4. 通过写 T2CON 来设置 TMR2 预分频值并使能 Timer2。
5. 配置 CCP4 模块使之工作于 PWM 模式。

表 19-7： 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	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
PIR4	CCP10IF(1)	CCP9IF(1)	CCP8IF	CCP7IF	CCP6IF	CCP5IF	CCP4IF	CCP3IF
PIE4	CCP10IE(1)	CCP9IE(1)	CCP8IE	CCP7IE	CCP6IE	CCP5IE	CCP4IE	CCP3IE
IPR4	CCP10IP(1)	CCP9IP(1)	CCP8IP	CCP7IP	CCP6IP	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	TRISE3	TRISE2	TRISE1	TRISE0

图注： — = 未实现，读为 0。 PWM 或 Timer2/4/6/8 不使用阴影单元。

注 1： 在具有 32 KB 程序存储器的器件（PIC18F65K22 和 PIC18F85K22）上未实现。

2： 在 64 引脚器件（PIC18F6XK22）上未实现，读为 0。

© 2011 Microchip Technology Inc.

表 19-7： 与 PWM 和定时器相关的寄存器 （续）

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0
TMR2	Timer2 寄存器							
TMR4	Timer4 寄存器							
TMR6	Timer6 寄存器							
TMR8	Timer8 寄存器							
PR2	Timer2 周期寄存器							
PR4	Timer4 周期寄存器							
PR6	Timer6 周期寄存器							
PR8	Timer8 周期寄存器							
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
T6CON	—	T6OUTPS3	T6OUTPS2	T6OUTPS1	T6OUTPS0	TMR6ON	T6CKPS1	T6CKPS0
T8CON	—	T8OUTPS3	T8OUTPS2	T8OUTPS1	T8OUTPS0	TMR8ON	T8CKPS1	T8CKPS0
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							
CCPR6L	捕捉 / 比较 /PWM 寄存器 6 的低字节							
CCPR6H	捕捉 / 比较 /PWM 寄存器 6 的高字节							
CCPR7L	捕捉 / 比较 /PWM 寄存器 7 的低字节							
CCPR7H	捕捉 / 比较 /PWM 寄存器 7 的高字节							
CCPR8L	捕捉 / 比较 /PWM 寄存器 8 的低字节							
CCPR8H	捕捉 / 比较 /PWM 寄存器 8 的高字节							
CCPR9L(1)	捕捉 / 比较 /PWM 寄存器 9 的低字节							
CCPR9H(1)	捕捉 / 比较 /PWM 寄存器 9 的高字节							
CCPR10L(1)	捕捉 / 比较 /PWM 寄存器 10 的低字节							
CCPR10H(1)	捕捉 / 比较 /PWM 寄存器 10 的高字节							
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0
CCP6CON	—	—	DC6B1	DC6B0	CCP6M3	CCP6M2	CCP6M1	CCP6M0
CCP7CON	—	—	DC7B1	DC7B0	CCP7M3	CCP7M2	CCP7M1	CCP7M0
CCP8CON	—	—	DC8B1	DC8B0	CCP8M3	CCP8M2	CCP8M1	CCP8M0
CCP9CON(1)	—	—	DC9B1	DC9B0	CCP9M3	CCP9M2	CCP9M1	CCP9M0
CCP10CON(1)	—	—	DC10B1	DC10B0	CCP10M3	CCP10M2	CCP10M1	CCP10M0
CCPTMRS1	C7TSEL1	C7TSEL0	—	C6TSEL0	—	C5TSEL0	C4TSEL1	C4TSEL0
CCPTMRS2	—	—	—	C10TSEL0(1)	—	C9TSEL0(1)	C8TSEL1	C8TSEL0
PMD3	CCP10MD(1)	CCP9MD(1)	CCP8MD	CCP7MD	CCP6MD	CCP5MD	CCP4MD	TMR12MD(1)

图注： — = 未实现，读为 0。 PWM 或 Timer2/4/6/8 不使用阴影单元。

注 1： 在具有 32 KB 程序存储器的器件 （PIC18F65K22 和 PIC18F85K22）上未实现。

2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

注：

© 2011 Microchip Technology Inc.

20.0 增强型捕捉 / 比较 /PWM（ECCP）模块

PIC18F87K22 系列器件有三个增强型捕捉 / 比较 /PWM（ECCP）模块：ECCP1、 ECCP2 和 ECCP3。这些 ECCP 模块都有一个 16 位寄存器，它可以用作 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 兼容。

注：	在本章节中 , 对于相同的寄存器和位名均采用统称，其中x变量表示与ECCP1、ECCP2 或 ECCP3 模块相关的项。例如，控制寄存器命名为 CCPxCON，其指代 CCP1CON、CCP2CON 和 CCP3CON。

ECCP1、ECCP2 和 ECCP3 被实现为带有增强型 PWM 功能的标准 CCP 模块。这些增强的功能包括：

- 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 （增强模式）”将详细讨论这些增强功能。

ECCP1、 ECCP2 和 ECCP3 模块使用控制寄存器：CCP1CON、CCP2CON 和 CCP3CON。CCP4CON 至 CCP10CON 为 CCP4 至 CCP10 模块的控制寄存器。

寄存器 20-1： CCPxCON：增强型捕捉 / 比较 /PWMx 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
PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

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

bit 7-6 **PxM<1:0>**：增强型 PWM 输出配置位

如果 CCPxM<3:2> = 00、 01 或 10：

xx = PxA 被指定为捕捉 / 比较输入 / 输出； PxB、 PxC 和 PxD 被指定为端口引脚

如果 CCPxM<3:2> = 11：

00 = 单输出：PxA、 PxB、 PxC 和 PxD 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：PxD 被调制； PxA 有效； PxB 和 PxC 无效
10 = 半桥输出：PxA 和 PxB 被调制，带死区控制； PxC 和 PxD 被指定为端口引脚
11 = 全桥反向输出：PxB 被调制； PxC 有效； PxA 和 PxD 无效

bit 5-4 **DCxB<1:0>**：PWM 占空比 Bit 1 和 Bit 0

捕捉模式：
未用。

比较模式：
未用。

PWM 模式：
这两位是 10 位 PWM 占空比的低 2 位 （LSb）。占空比的高 8 位 （MSb）在 CCPRxL 中。

bit 3-0 **CCPxM<3:0>**：ECCPx 模式选择位

0000 = 捕捉 / 比较 /PWM 关闭 （复位 ECCPx 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：每个下降沿捕捉
0101 = 捕捉模式：每个上升沿捕捉
0110 = 捕捉模式：每 4 个上升沿捕捉
0111 = 捕捉模式：每 16 个上升沿捕捉
1000 = 比较模式：初始化 ECCPx 引脚为低电平；比较匹配时输出高电平 （CCPxIF 置 1）
1001 = 比较模式：初始化 ECCPx 引脚为高电平；比较匹配时输出低电平 （CCPxIF 置 1）
1010 = 比较模式：仅产生软件中断； ECCPx 引脚恢复到 I/O 状态
1011 = 比较模式：触发特殊事件 （ECCPx 复位 TMR1 或 TMR3，启动 A/D 转换器，将 CCxIF 位置 1）
1100 = PWM 模式：PxA 和 PxC 为高电平有效； PxB 和 PxD 也为高电平有效
1101 = PWM 模式：PxA 和 PxC 为高电平有效； PxB 和 PxD 为低电平有效
1110 = PWM 模式：PxA 和 PxC 为低电平有效； PxB 和 PxD 为高电平有效
1111 = PWM 模式：PxA 和 PxC 为低电平有效； PxB 和 PxD 也为低电平有效

© 2009-2011 Microchip Technology Inc.

寄存器 20-2： CCPTMRS0：CCP 定时器选择寄存器 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
C3TSEL1	C3TSEL0	C2TSEL2	C2TSEL1	C2TSEL0	C1TSEL2	C1TSEL1	C1TSEL0
bit 7							bit 0

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

bit 7-6 **C3TSEL<1:0>：**ECCP3 定时器选择位
00 = ECCP3 基于 TMR1/TMR2 工作
01 = ECCP3 基于 TMR3/TMR4 工作
10 = ECCP3 基于 TMR3/TMR6 工作
11 = ECCP3 基于 TMR3/TMR8 工作

bit 5-3 **C2TSEL<2:0>：**ECCP2 定时器选择位
000 = ECCP2 基于 TMR1/TMR2 工作
001 = ECCP2 基于 TMR3/TMR4 工作
010 = ECCP2 基于 TMR3/TMR6 工作
011 = ECCP2 基于 TMR3/TMR8 工作
100 = ECCP2 基于 TMR3/TMR10 工作：该选项在 32 KB 器件上保留；不用
101 = 保留；不用
110 = 保留；不用
111 = 保留；不用

bit 2-0 **C1TSEL<2:0>：**ECCP1 定时器选择位
000 = ECCP1 基于 TMR1/TMR2 工作
001 = ECCP1 基于 TMR3/TMR4 工作
010 = ECCP1 基于 TMR3/TMR6 工作
011 = ECCP1 基于 TMR3/TMR8 工作
100 = ECCP1 基于 TMR3/TMR10 工作：该选项在 32 KB 器件上保留；不用
101 = ECCP1 基于 TMR3/TMR12 工作：该选项在 32 KB 器件上保留；不用
110 = 保留；不用
111 = 保留；不用

© 2009-2011 Microchip Technology Inc.

除了通过 CCPxCON 寄存器和 ECCPxAS寄存器可获得扩展的模式范围以外， 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：

- ECCPxDEL——增强型 PWM 控制
- PSTRxCON——脉冲转向控制

20.1 ECCP 输出和配置

根据选定的工作模式，每个增强型 CCP 模块最多有 4 路 PWM 输出。 CCPxCON 寄存器改为允许控制四个 PWM 输出：ECCPx/PxA、PxB、PxC 和 PxD。应用可使用这些输出中的一个、两个或四个。

输出是否有效取决于选定的 ECCP 工作模式。表 20-3 中总结了引脚的分配。

要将 I/O 引脚配置为 PWM 输出，必须通过设置 PxM<1:0> 和 CCPxM<3:0> 位来选择适当的 PWM 模式。还必须将对应端口引脚的相应 TRIS 方向位设置为输出。

20.1.1 ECCP 模块和定时器资源

根据选定的模式，ECCP 模块使用定时器 1、2、3、4、6、8、10 或 12。这些定时器可用在捕捉、比较或 PWM 模式下的 CCP 模块，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、 Timer4、 Timer6、 Timer8、 Timer10 或 Timer12

要将哪个特定的定时器分配给 ECCP 模块是由 CCPTMRSx 寄存器 （寄存器 20-2）中的 ECCP 定时器使能位决定的。图 20-1 描述了两个模块之间的相互关系。捕捉操作被设计为在定时器配置为同步计数器模式时使用。如果相关的定时器配置为异步计数器模式，捕捉操作可能无法按预期工作。

20.1.2 ECCP 引脚分配

可根据器件配置改变 ECCPx 的引脚分配 （捕捉输入、比较和 PWM 输出）。ECCPMX（CONFIG3H<1>）配置位决定 ECCP1 和 ECCP3 与哪个引脚复用。

- 默认值 /ECCPMX = 1：
  - ECCP1 （P1B/P1C）与 RE6 和 RE5 引脚复用
  - ECCP3 （P3B/P3C）与 RE4 和 RE3 引脚复用
- ECCPMX = 0：
  - ECCP1 （P1B/P1C）与 RH7 和 RH6 引脚复用
  - ECCP3 （P3B/P3C）与 RH5 和 RH4 引脚复用

可根据器件配置改变 ECCP2 的引脚分配 （捕捉输入、比较和 PWM 输出）。

CCP2MX 配置位（CONFIG3H<0>）决定 ECCP2 与哪个引脚复用。

- 如果 CCP2MX = 1 （默认值）——ECCP2 与 RC1 复用
- 如果 CCP2MX = 0——ECCP2 与 RE7 复用

© 2009-2011 Microchip Technology Inc.

20.2 捕捉模式

在捕捉模式下，当对应的 ECCPx 引脚上发生以下事件时，CCPRxH:CCPRxL 寄存器对将捕捉 TMR1 或 TMR3 寄存器的 16 位值。事件定义为下列情况之一：

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

通过模式选择位 CCPxM<3:0>（CCPxCON<3:0>）选择事件。当一次捕捉完成时，中断请求标志位 CCPxIF（见表 20-2）置 1；该标志位必须用软件清零。如果在读取寄存器 CCPRxH/L 之前发生了另一次捕捉，那么之前捕捉的值将被新捕捉的值覆盖。

表 20-2： ECCP1/2/3 中断标志位

ECCP 模式	标志位
1	PIR3<1>
2	PIR3<2>
3	PIR4<0>

20.2.1 ECCP 引脚配置

在捕捉模式下，应该通过将对应的 TRIS 方向位置 1 来将相应的 ECCPx 引脚配置为输入。

> **注：** 如果 ECCPx 引脚被配置成输出，对该端口的写操作会产生捕捉条件。

20.2.2 TIMER1/2/3/4/6/8/10/12 模式选择

用于捕捉功能的定时器（Timer1、2、3、4、6、8、10 或 12）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，无法进行捕捉操作。用于每个 ECCP 模块的定时器由 CCPTMRS0 寄存器（寄存器 20-2）进行选择。

20.2.3 软件中断

当捕捉模式改变时，可能会产生错误捕捉中断。用户应该保持 CCPxIE 中断允许位清零，以避免错误中断。还应该在工作模式发生任何改变之后清零中断标志位 CCPxIF。

20.2.4 ECCP 预分频器

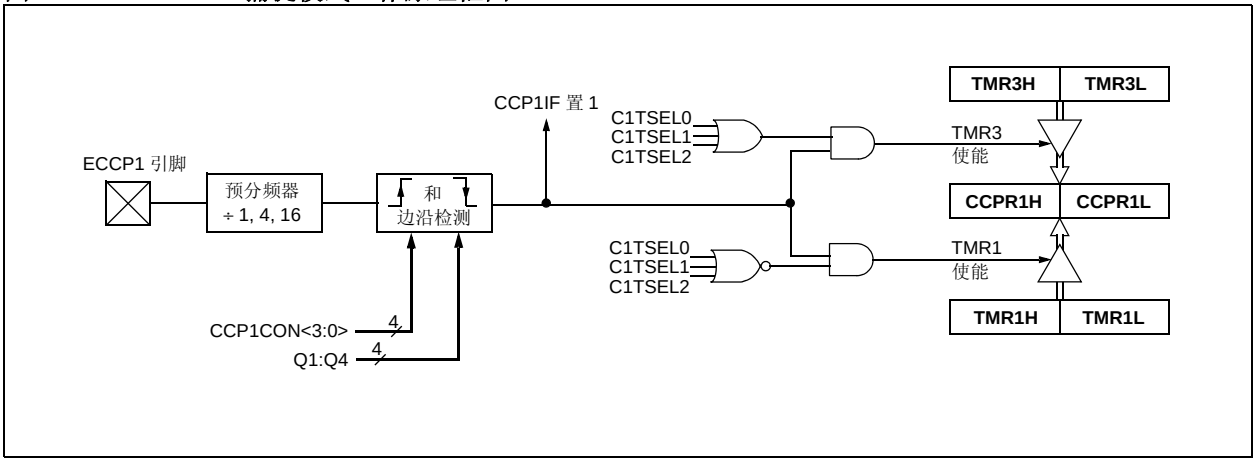
在捕捉模式下有 4 种预分频比设置。它们可作为工作模式的一部分由模式选择位（CCPxM<3:0>）选择。当关闭 ECCP 模块或禁止捕捉模式时，预分频计数器就会被清零。这意味着任何复位都将清零预分频计数器。

从一种捕捉预分频比切换到另一种捕捉预分频比可能会产生中断。而且，不会清零预分频计数器，因此切换后的第一次捕捉可能来自于一个非零的预分频器。例 20-1 是切换捕捉预分频比时建议采用的方法。该示例清零预分频计数器且不会产生“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 CCP1CON 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 CCPRx 寄存器的值将不断与 CCPTMR1 寄存器中选择的定时器寄存器对的值相比较。当两者匹配时， ECCPx 引脚将被：

- 驱动为高电平
- 驱动为低电平
- 翻转电平 （高电平变为低电平或低电平变为高电平）
- 保持不变 （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。

20.3.1 ECCP 引脚配置

用户必须通过将相应的 TRIS 位清零来将 ECCPx 引脚配置为输出。

注：	清零 CCPxCON 寄存器会将 ECCPx 比较输出锁存器 （取决于器件配置）强制为默认低电平。这不是PORTx I/O数据锁存器。

20.3.2 TIMER1/2/3/4/6/8/10/12 模式选择

如果 ECCP 模块使用比较功能，Timer1、2、3、4、6、8、 10 或 12 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，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（CCPxM<3:0> = `1010`）时，ECCPx 引脚不受影响，只有 CCPxIF 中断标志位受影响。

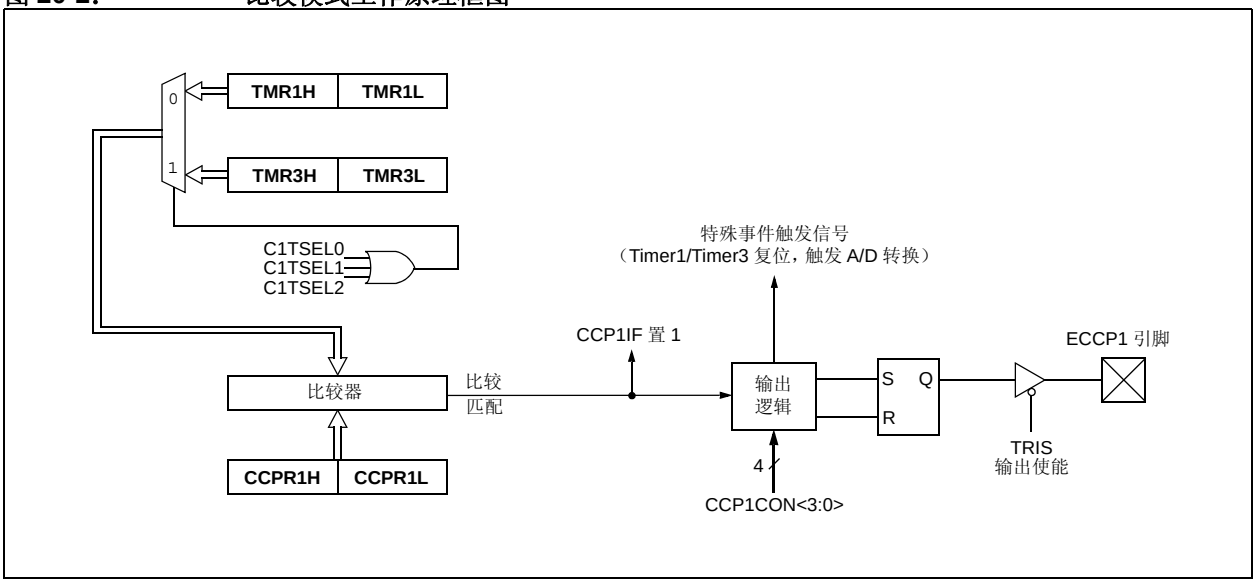
20.3.4 特殊事件触发信号

ECCP 模块配备一个特殊事件触发器。该触发信号为内部硬件信号，在比较模式下产生以触发其他模块的动作。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。

无论当前使用哪个定时器资源作为模块的时基，特殊事件触发信号都将把对应的定时器寄存器对复位。这样 CCPRx 寄存器就可用作任一个定时器的可编程周期寄存器。

ECCP 的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



© 2009-2011 Microchip Technology Inc.

20.4 PWM （增强模式）

增强型 PWM 模式可在最多四个不同输出引脚上产生最高10位分辨率的PWM输出信号。可通过四个不同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，CCPxCON 寄存器的 PxM 位必须相应地置 1。

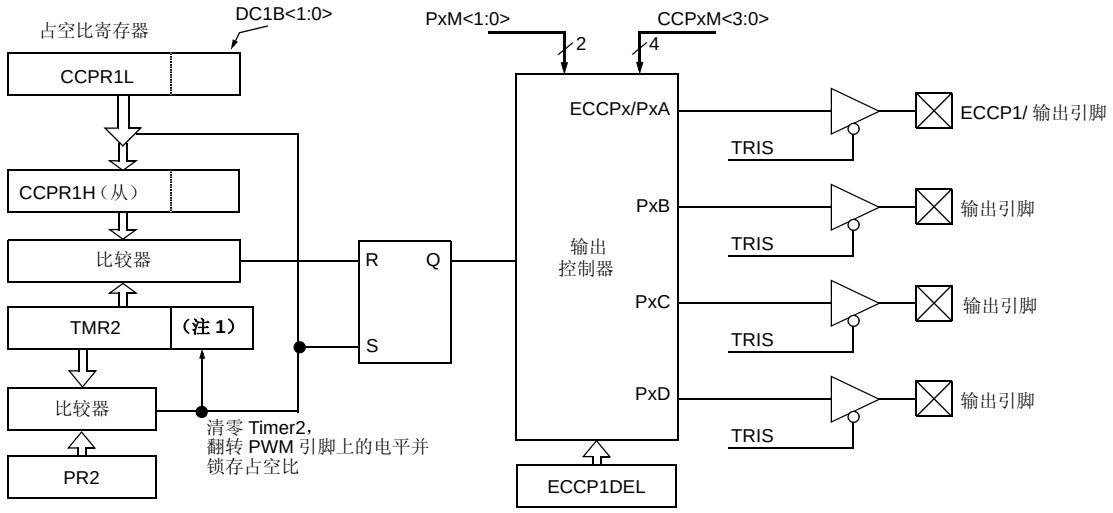
PWM 输出与 I/O 引脚复用且命名为：PxA、PxB、PxC 和 PxD。 PWM 引脚的极性是可配置的，并通过适当地将 CCPxCON 寄存器的 CCPxM 位置 1 来选择。

表 20-1 为每种增强型 PWM 模式的引脚分配。

图 20-3 为增强型 PWM 模块的简化框图示例。

> **注：** 为了防止在首次使能 PWM 时产生不完整波形， ECCP 模块在产生 PWM 信号之前会等待直到开始新的 PWM 周期。

图 20-3： 增强型 PWM 模式的简化框图示例



注 1： 8 位 TMR2 寄存器与 2 位内部 Q 时钟或预分频器的 2 位一起构成 10 位时基。

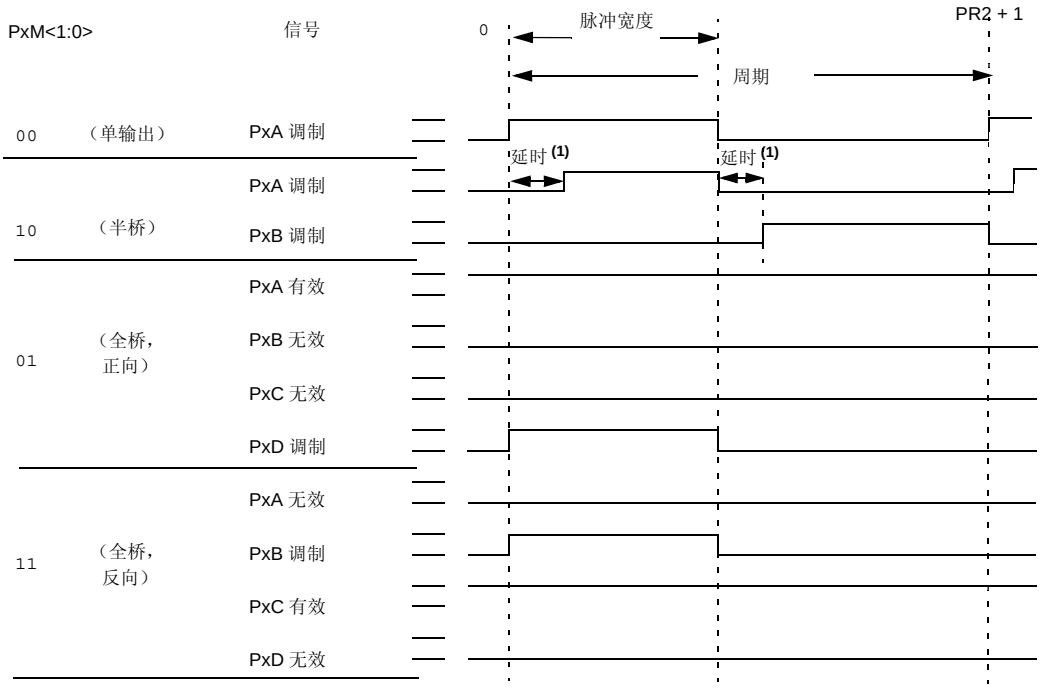
> **注 1：** 每路 PWM 输出对应的 TRIS 寄存器值必须进行适当配置。
> **2：** 未被增强型 PWM 模式使用的任何引脚都可用作备用引脚。

表 20-3： 各种 PWM 增强模式的引脚分配示例

ECCP 模式	PxM<1:0>	PxA	PxB	PxC	PxD
单	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，输出由脉冲转向使能 （见寄存器 20-5）。

图 20-4： PWM （增强模式）输出关系示例 （高电平有效状态）



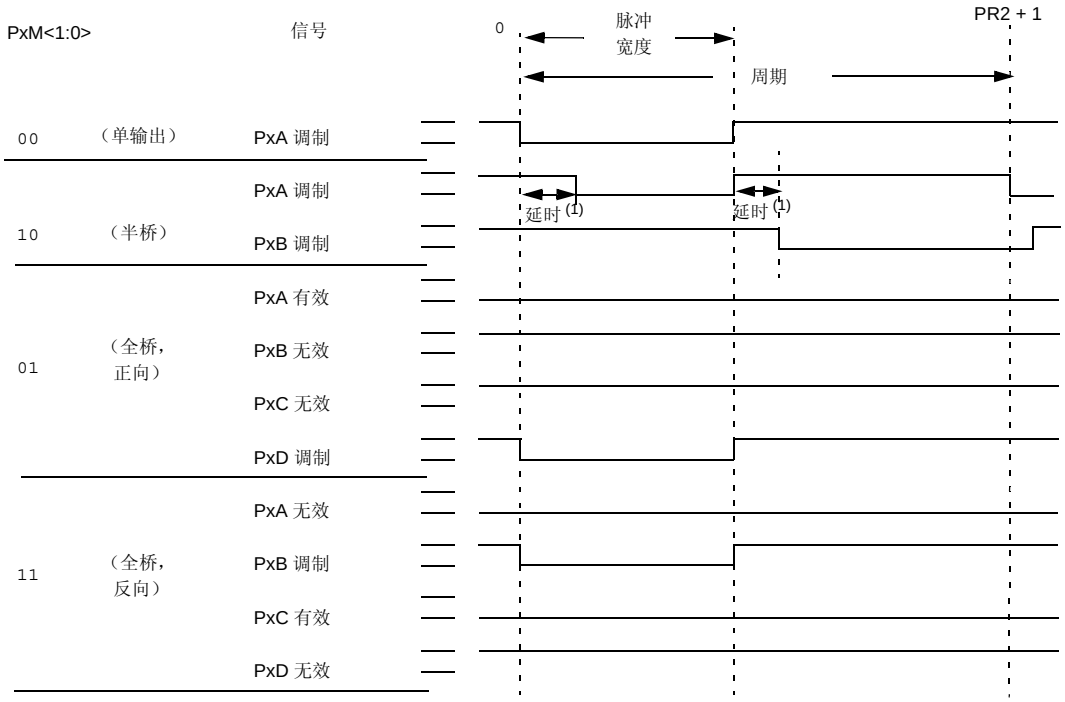
关系：

- 周期 = 4 * Tosc * (PR2 + 1) * (TMR2 预分频值)
- 脉冲宽度 = Tosc * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMR2 预分频值)
- 延时 = 4 * Tosc * (ECCPxDEL<6:0>)

注 1： 使用 ECCPxDEL 寄存器对死区延时进行编程设置 （见第 20.4.6 节 “可编程死区延时模式”）。

© 2009-2011 Microchip Technology Inc.

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 * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMR2 预分频值)
- 延时 = 4 * Tosc * (ECCPxDEL<6:0>)

注 1： 使用 ECCP1DEL 寄存器对死区延时进行编程设置 （见**第 20.4.6 节 “可编程死区延时模式”**）。

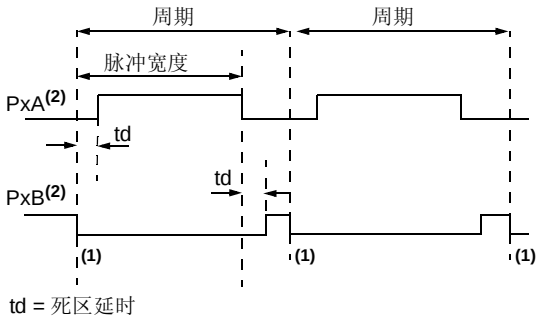
20.4.1 半桥模式

在半桥模式下，两个引脚用作输出端来驱动推挽式负载。PWM 输出信号在 PxA 引脚上输出，而互补的 PWM 输出信号在 PxB 引脚上输出（见图 20-6）。此模式可用于半桥应用（如图 20-7 所示）或那些使用 2 路 PWM 信号来调制 4 个功率开关的全桥应用。

在半桥模式下，可编程的死区延时用来防止直通（shoot-through）电流从半桥功率器件中流过。ECCPxDEL 寄存器的 PxDC<6:0> 位的值可用来设置输出被驱动为有效之前的指令周期数。如果该值大于占空比，在整个周期内对应的输出将保持无效。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 PxA 和 PxB 输出与端口数据锁存器复用，必须清零相应的 TRIS 位以将 PxA 和 PxB 配置为输出。

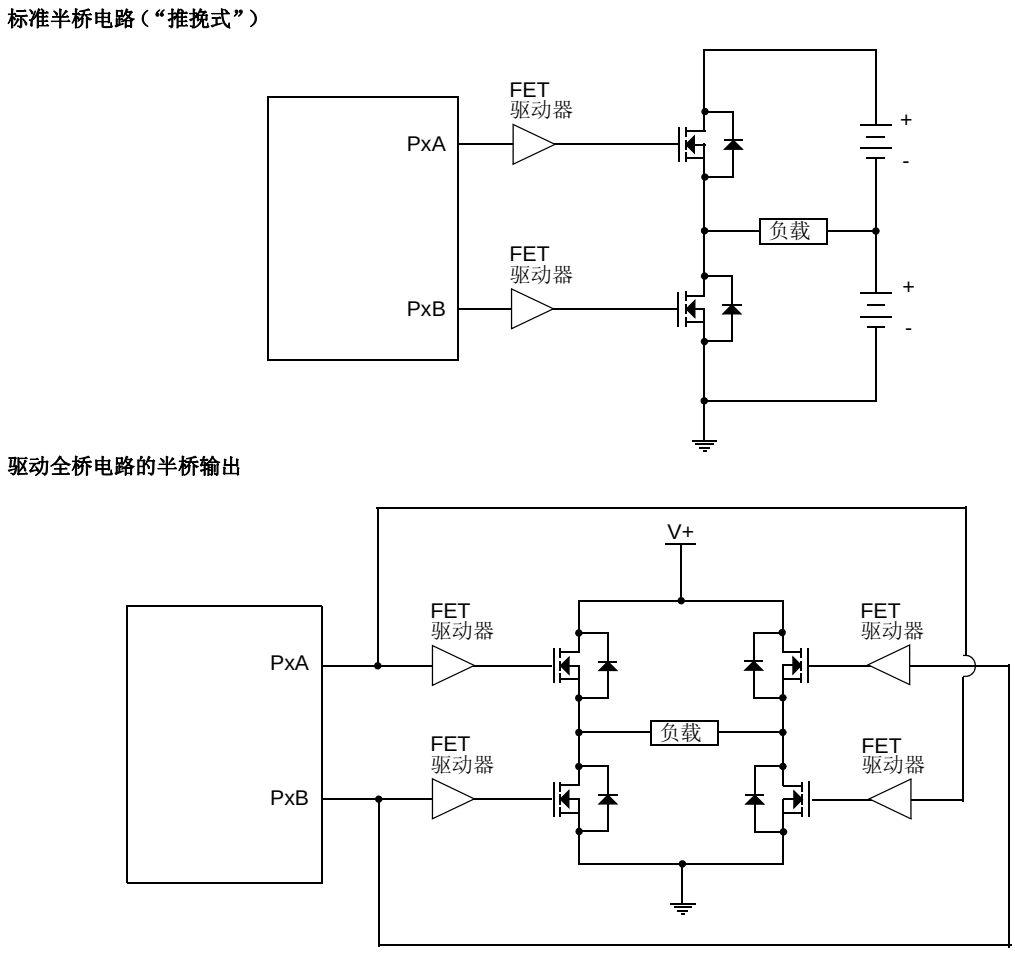
图 20-6： 半桥 PWM 输出示例



注 1： 此时，TMR2 寄存器的值与 PR2 寄存器的值相等。

2： 输出信号为高电平有效。

图 20-7： 半桥应用示例



© 2009-2011 Microchip Technology Inc.

20.4.2 全桥模式

在全桥模式下，所有 4 个引脚都被用作输出。图 20-8 为全桥输出应用的一个示例。

在正向模式下，引脚 PxA 被驱动为有效状态，引脚 PxD 被调制。而 PxB 和 PxC 被驱动为无效状态，如图 20-9 所示。

在反向模式下，引脚 PxC 被驱动为有效状态，引脚 PxB 被调制，而 PxA 和 PxD 被驱动为无效状态，如图 20-9 所示。

PxA、PxB、PxC 和 PxD 输出与端口数据锁存器复用。必须清零相应的 TRIS 位以将 PxA、PxB、PxC 和 PxD 引脚配置为输出引脚。

图 20-8： 全桥应用示例

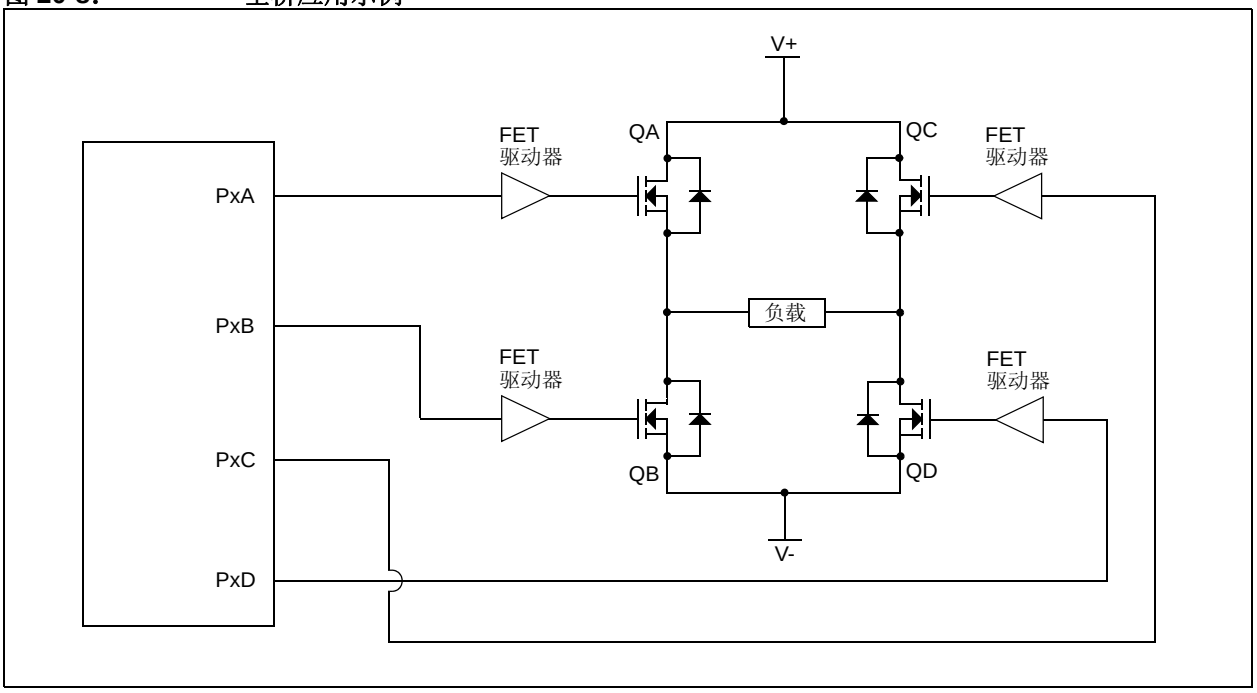
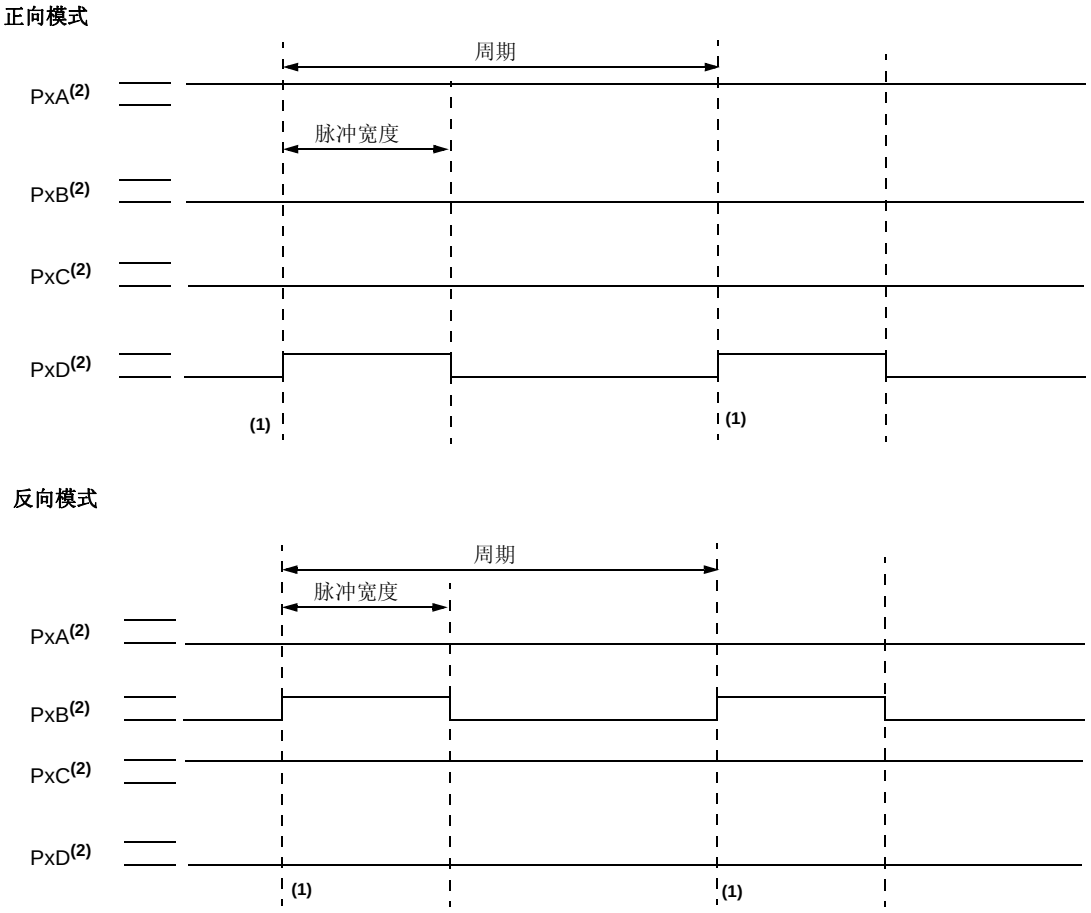


图 20-9： 全桥 PWM 输出示例



注 1： 此时， TMR2 寄存器的值与 PR2 寄存器的值相等。
2： 输出信号为高电平有效。

© 2009-2011 Microchip Technology Inc.

20.4.2.1 全桥模式下的方向更改

在全桥模式下，CCPxCON 寄存器中的 PxM1 位使用户能控制负载中电流为正向还是反向。当应用固件更改此方向控制位时，模块将在下一个 PWM 周期采用新的方向。

用软件通过更改 CCPxCON 寄存器的 PxM1 位的值来启动方向更改。在当前 PWM 周期结束之前发生以下序列：

- 调制输出 （PxB 和 PxD）被置于无效状态。
- 相关的未调制输出 （PxA 和 PxC）被切换到以相反的方向驱动负载。
- PWM 调制在下一个周期开始时继续。

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。通常，由于在任何时间只调制一个输出，所以不需要死区延时。然而，有一种情况可能需要死区延时。即，当以下两个条件同时满足时：

- 当输出的占空比达到或接近 100% 时， PWM 输出的方向发生改变。
- 功率开关 （包括功率器件和驱动电路）的关断时间大于导通时间。

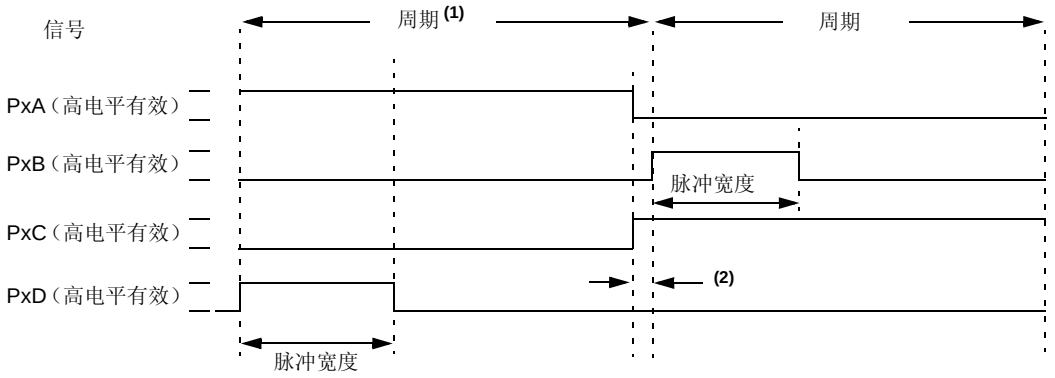
图 20-11 所示为占空比接近 100% 时， PWM 方向从正向变为反向的示例。在本例中，在时间 t1，输出 PxA 和 PxD 变为无效，而输出 PxC 变为有效。由于功率器件的关断时间比导通时间长，直通电流可能在时间段“t”内流过功率器件 QC 和 QD （见图 20-8）。如果 PWM 方向从反向变为正向，功率器件 QA 和 QB 将出现相同的现象。

如果应用需要在高占空比情况下更改 PWM 方向，有两种方法可消除直通电流：

- 在更改方向前的那个 PWM 周期缩小 PWM 的占空比。
- 使用可使功率器件的关断速度比导通速度快的开关驱动器。

可能还存在防止直通电流的其他方法。

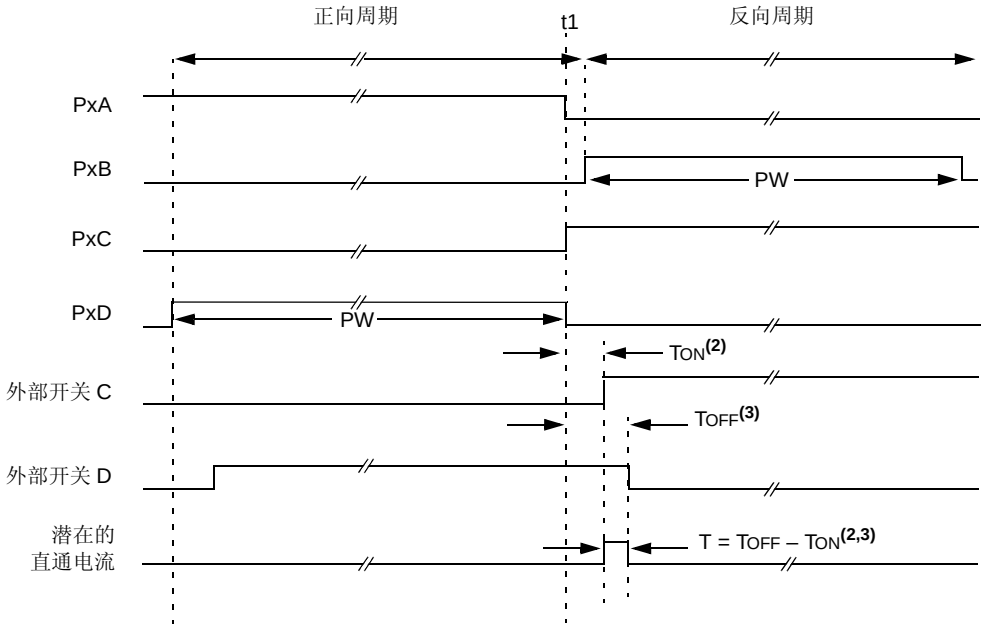
图 20-10： PWM 方向更改示例



注 1： 可在 PWM 周期内的任何时间写 CCPxCON 寄存器中的方向位 PxM1。

2： 当更改方向时， PxA 和 PxC 上的信号在当前 PWM 周期结束之前进行切换。在此期间被调制的 PxB 和 PxD 信号无效。时间长度是：(1/FOSC) • TMR2 预分频值。

图 20-11： 在占空比接近 100% 时更改 PWM 方向示例 (1)



注 1： 图中所有信号均为高电平有效。
2： TON 为功率开关 QC 及其驱动器的导通延时。
3： TOFF 为功率开关 QD 及其驱动器的关断延时。

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 输出引脚上使用适当的外部上拉和 / 或下拉电阻。

> **注：** 当单片机从复位状态恢复时，所有的 I/O 引脚都处于高阻态。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用适当的信号电平驱动 I/O 引脚，或激活 PWM 输出为止。

CCPxCON 寄存器的 CCPxM<1:0> 位允许用户为每对 PWM 输出引脚（PxA/PxC 和 PxB/PxD）选择 PWM 输出信号是高电平有效还是低电平有效。必须在使能 PWM 引脚输出驱动器之前选择 PWM 输出的极性。建议不要在使能 PWM 引脚输出驱动器时更改极性配置，因为这可能造成应用电路损坏。

当初始化 PWM 模块时，PxA、PxB、PxC 和 PxD 的输出锁存器可能处于不正确的状态。使能增强型 PWM 模式同时使能 PWM 引脚输出驱动器可能导致应用电路损坏。必须将增强型 PWM 模式使能为正确的输出模式并且必须经过一个完整的 PWM 周期，才能使能 PWM 引脚输出驱动器。当第 2 个 PWM 周期开始时，PIR1 寄存器或 PIR5 寄存器的 TMR2IF 位或 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支持自动关闭模式，即当发生外部关闭事件时将禁止 PWM 输出。自动关闭模式会将 PWM 输出引脚置于预先确定的状态。该模式有助于防止 PWM 破坏应用。

使用 ECCPxAS<2:0> 位 （ECCPxAS<6:4>）来选择自动关闭源。关闭事件可由以下情况产生：

- 充当 FLT0 输入功能的引脚的状态为逻辑 0
- 比较器 C1
- 比较器 C2
- 由固件将 ECCPxASE 位置 1

关闭状态通过 ECCPxASE （自动关闭事件状态）位（ECCPxAS<7>）指示。如果该位为 0，则 PWM 引脚正常工作。如果该位为 1，则 PWM 输出处于关闭状态。

© 2009-2011 Microchip Technology Inc.

当关闭事件发生时，发生以下两个事件：

- ECCPxASE 位被设置为 1。ECCPxASE 位将保持置 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 引脚被陆续地置于关闭状态。 PWM 输出引脚分为 PxA/PxC 和 PxB/PxD 引脚对。每对引脚的状态分别由 PSSxAC 和 PSSxBD 位（ECCPxAS<3:2> 和 <1:0>）指定。

每对引脚状态可被设置为以下三种状态之一：

- 驱动逻辑 1
- 驱动逻辑 0
- 三态 （高阻态）

寄存器 20-3： ECCPxAS：ECCPx 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
ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0
bit 7							bit 0

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

bit 7 **ECCPxASE：**ECCP 自动关闭事件状态位
1 = 已发生了关闭事件； ECCP 输出处于关闭状态
0 = ECCP 输出正常工作

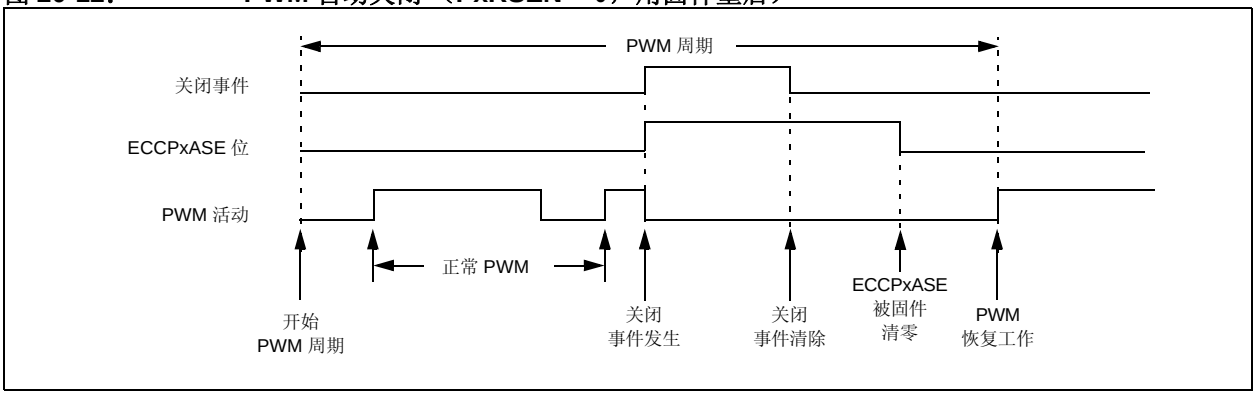
bit 6-4 **ECCPxAS<2:0>：**ECCP 自动关闭源选择位
000 = 禁止自动关闭
001 = 比较器 C1OUT 输出为高电平
010 = 比较器 C2OUT 输出为高电平
011 = 比较器 C1OUT 或 C2OUT 为高电平
100 = FLT0 引脚电压为 VIL
101 = FLT0 引脚电压为 VIL 或比较器 C1OUT 输出为高电平
110 = FLT0 引脚电压为 VIL 或比较器 C2OUT 输出为高电平
111 = FLT0 引脚电压为 VIL、比较器 C1OUT 或比较器 C2OUT 输出为高电平

bit 3-2 **PSSxAC<1:0>：**引脚 PxA 和 PxC 关闭状态控制位
00 = 驱动 PxA 和 PxC 引脚为 0
01 = 驱动 PxA 和 PxC 引脚为 1
1x = PxA 和 PxC 引脚为三态

bit 1-0 **PSSxBD<1:0>：**引脚 PxB 和 PxD 关闭状态控制位
00 = 驱动 PxB 和 PxD 引脚为 0
01 = 驱动 PxB 和 PxD 引脚为 1
1x = PxB 和 PxD 引脚为三态

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，自动关闭条件就一直保持。
2：自动关闭条件存在时，禁止写 ECCPxASE 位。
3：一旦清除自动关闭条件并且 PWM 自动重启（由固件或者自动重启），PWM 信号将始终在下一个 PWM 周期的开始重启。

图 20-12： PWM 自动关闭 （PxRSEN = 0，用固件重启）



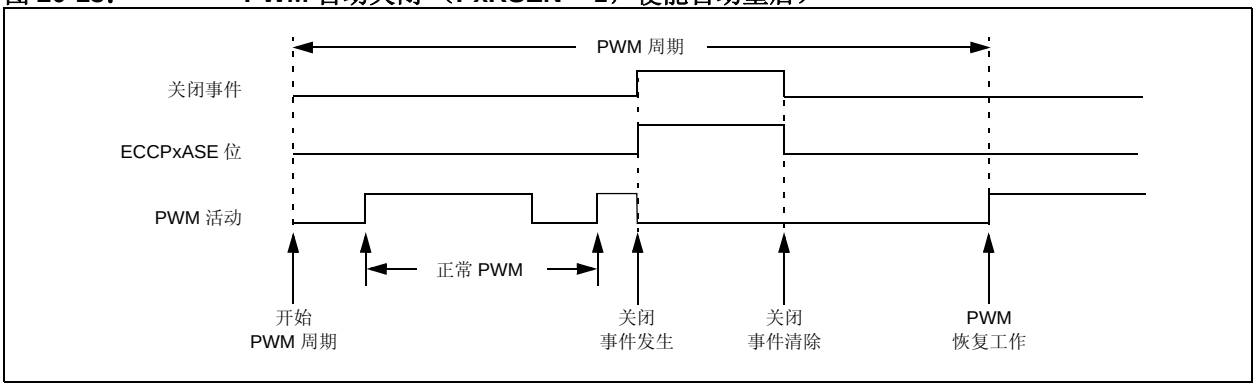
20.4.5 自动重启模式

增强型 PWM 可配置在清除自动关闭条件时自动重启 PWM 信号。通过将 PxRSEN 位 （ECCPxDEL<7>）置 1 使能自动重启。

如果使能自动重启，只要自动关闭条件有效，ECCPxASE 位就将保持置 1。当自动关闭条件被清除后，将由硬件清零 ECCPxASE 位且增强型 PWM 恢复正常操作。

但是，该模块在重新使能输出引脚之前将等待直到下一个PWM周期开始。这种行为使得可以在基于当前PWM控制模式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 （PxRSEN = 1，使能自动重启）



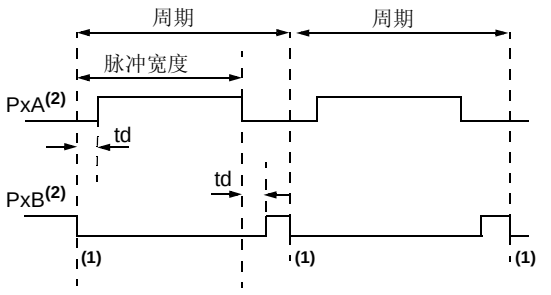
© 2009-2011 Microchip Technology Inc.

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切换（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内，会有一股很大的电流（直通电流）流过这两个功率开关，从而导致桥式供电电路短路。为了避免在切换期间流过这种潜在的破坏性直通电流，一般延迟导通其中一个功率开关以使另一个开关完全关断。

在半桥模式下，可数字编程的死区延时可用来防止直通电流损坏桥式电路中的功率开关。该延迟在信号从无效状态到有效状态转换时发生，见图 20-14 中的说明。相关 ECCPxDEL 寄存器（寄存器 20-4）的低 7 位以单片机指令周期（TCY 或 4 TOSC）为单位来设置延时。

图 20-14： 半桥 PWM 输出示例

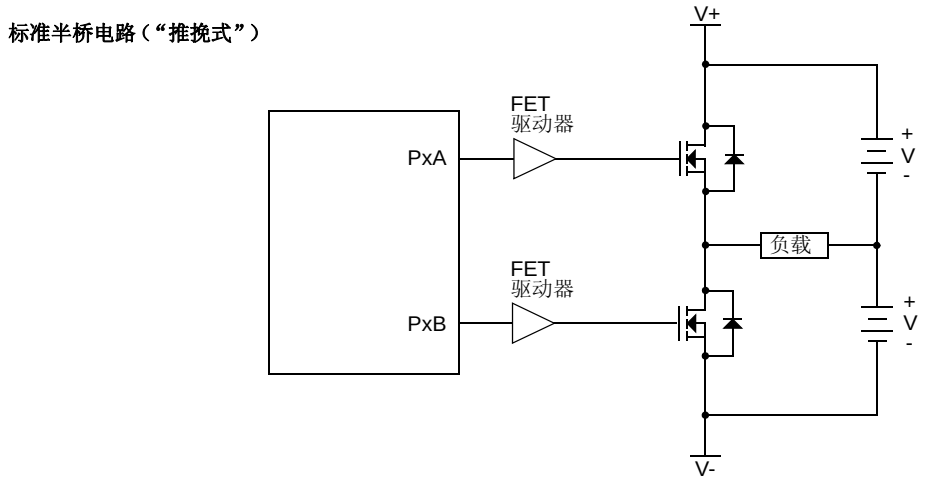


td = 死区延时

注 1：此时，TMR2 寄存器的值与 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： ECCPxDEL：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
PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0
bit 7							bit 0

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

bit 7 **PxRSEN：**PWM 重启使能位

1 = 自动关闭时，一旦关闭事件被清除， ECCPxASE 位立即自动清零； PWM 自动重启

0 = 自动关闭时，必须用软件清零 ECCPxASE 位以重启 PWM

bit 6-0 **PxDC<6:0>：**PWM 延时计数位

PxDCn = 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，脉冲转向允许任一 PWM 引脚输出调制信号。此外，相同的 PWM 信号可同时提供给多个引脚。

一旦选择单输出模式（CCPxCON 寄存器的 CCPxM<3:2> = 11 和 PxM<1:0> = 00），用户固件便可通过设置相应的 STR<D:A> 位（PSTRxCON<3:0>）将相同的 PWM 信号输出到一个、两个、三个或四个输出引脚，如表 20-3 所示。

> **注：** 为了在引脚上输出PWM信号，相关的TRIS位必须设置为输出（0）以使能引脚输出驱动器。

当 PWM 转向模式有效时，通过 CCPxM<1:0> 位（CCPxCON<1:0>）选择 Px<D:A> 引脚的 PWM 输出极性。

PWM 自动关闭工作原理也适用于 PWM 转向模式，如**第 20.4.4 节“增强型 PWM 自动关闭模式”**所述。自动关闭将只影响已使能为 PWM 输出的引脚。

© 2009-2011 Microchip Technology Inc.

寄存器 20-5： PSTRxCON：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 = 上电复位时的值	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 = 参见 STR<D:A>
01 = PA 和 PB 被选择为互补输出对
10 = PA 和 PC 被选择为互补输出对
11 = PA 和 PD 被选择为互补输出对

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 = 在指令周期边界开始时发生输出转向更新

bit 3 **STRD：**转向使能位 D
1 = PxD 引脚的 PWM 波形极性由 CCPxM<1:0> 控制
0 = PxD 引脚被指定为端口引脚

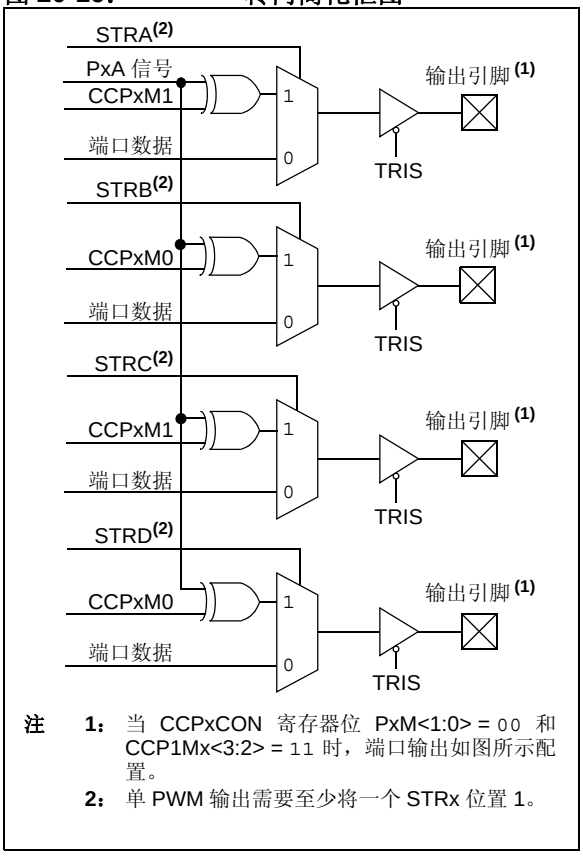
bit 2 **STRC：**转向使能位 C
1 = PxC 引脚的 PWM 波形极性由 CCPxM<1:0> 控制
0 = PxC 引脚被指定为端口引脚

bit 1 **STRB：**转向使能位 B
1 = PxB 引脚的 PWM 波形极性由 CCPxM<1:0> 控制
0 = PxB 引脚被指定为端口引脚

bit 0 **STRA：**转向使能位 A
1 = PxA 引脚的 PWM 波形极性由 CCPxM<1:0> 控制
0 = PxA 引脚被指定为端口引脚

注 1： PWM 转向模式仅在 CCPxCON 寄存器位 CCPxM<3:2> = 11 和 PxM<1:0> = 00 时可用。

图 20-16： 转向简化框图



注 1： 当 CCPxCON 寄存器位 PxM<1:0> = 00 和 CCP1Mx<3:2> = 11 时，端口输出如图所示配置。

2： 单 PWM 输出需要至少将一个 STRx 位置 1。

20.4.7.1 转向同步

PSTRxCON 寄存器的 STRSYNC 位向用户提供了何时发生转向事件的两种选择。当 STRSYNC 位为 0 时，转向事件将在写入 PSTRxCON 寄存器的指令结束时发生。在这种情况下，Px<D:A> 引脚上的输出信号可能为不完整 PWM 波形。当用户固件需要立即清除该引脚的 PWM 信号时，该操作非常有用。

当 STRSYNC 位为 1 时，有效的转向更新将在下一个 PWM 周期开始时发生。在这种情况下，转向开/关 PWM 输出将始终产生完整的 PWM 波形。

图 20-17 和 20-18 提供了基于 STRSYNC 设置的 PWM 转向时序图。

图 20-17： 指令周期结束时的转向事件示例 （STRSYNC = 0）

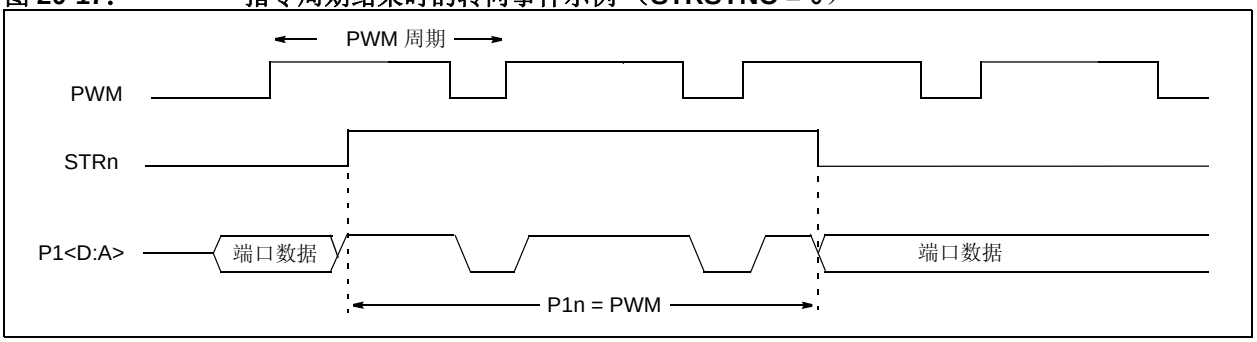
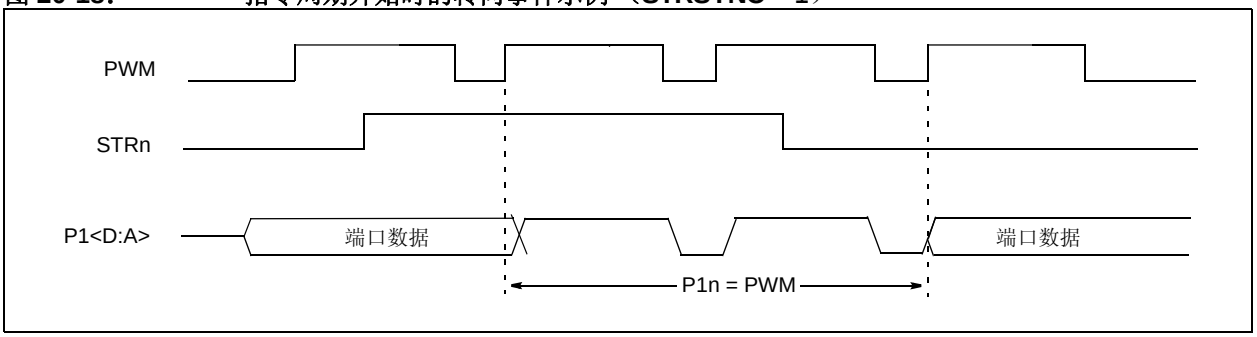


图 20-18： 指令周期开始时的转向事件示例 （STRSYNC = 1）



© 2009-2011 Microchip Technology Inc.

20.4.8 在功耗管理模式下工作

在休眠模式下，禁止所有时钟源。Timer2/4/6/8 将不会递增并且模块的状态将不发生变化。如果 ECCPx 引脚正在驱动一个值，则将继续驱动该值。当器件被唤醒时，它将从该状态继续。如果使能了双速启动，HF-INTOSC 和后分频器的初始起振频率可能不会立即稳定。

在 PRI_IDLE 模式下，主时钟将继续为 ECCPx 模块提供时钟，而不发生任何变化。

20.4.8.1 故障保护时钟监视器（FSCM）的相关操作

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功耗管理 RC_RUN 模式并且 PIR2 寄存器的 OSCFIF 位将置 1。然后由内部振荡器时钟源为 ECCPx 提供时钟，该时钟的频率与主时钟不同。

20.4.9 复位的影响

上电复位和后续的复位会将所有端口强制为输入模式，并强制 ECCP 寄存器进入复位状态。这会强制 ECCP 模块复位为与在其他 PIC18 和 PIC16 器件中使用的早期版本非增强型 CCP 模块兼容的状态。

表 20-4： 与 ECCP1/2/3 模块和 TIMER1/2/3/4/6/8/10/12 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	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIR4	CCP10IF(1)	CCP9IF(1)	CCP8IF	CCP7IF	CCP6IF	CCP5IF	CCP4IF	CCP3IF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
PIE4	CCP10IE(1)	CCP9IE(1)	CCP8IE	CCP7IE	CCP6IE	CCP5IE	CCP4IE	CCP3IE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
IPR4	CCP10IP(1)	CCP9IP(1)	CCP8IP	CCP7IP	CCP6IP	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	TRISE3	TRISE2	TRISE1	TRISE0
TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0
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							
TMR6	Timer6 寄存器							
TMR8	Timer8 寄存器							
TMR10(1)	Timer10 寄存器							
TMR12(1)	Timer12 寄存器							
PR2	Timer2 周期寄存器							
PR4	Timer4 周期寄存器							
PR6	Timer6 周期寄存器							
PR8	Timer8 周期寄存器							
PR10(1)	Timer10 周期寄存器							
PR12(1)	Timer12 周期寄存器							

注 1： 在具有 32 KB 程序存储器的器件（PIC18F65K22 和 PIC18F85K22）上未实现。
2： 在 64 引脚器件（PIC18F6XK22）上未实现，读为 0。

表 20-4： 与 ECCP1/2/3 模块和 TIMER1/2/3/4/6/8/10/12 相关的寄存器 （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
T6CON	—	T6OUTPS3	T6OUTPS2	T6OUTPS1	T6OUTPS0	TMR6ON	T6CKPS1	T6CKPS0
T8CON	—	T8OUTPS3	T8OUTPS2	T8OUTPS1	T8OUTPS0	TMR8ON	T8CKPS1	T8CKPS0
T10CON(1)	—	T10OUTPS3	T10OUTPS2	T10OUTPS1	T10OUTPS0	TMR10ON	T10CKPS1	T10CKPS0
T12CON(1)	—	T12OUTPS3	T12OUTPS2	T12OUTPS1	T12OUTPS0	TMR12ON	T12CKPS1	T12CKPS0
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	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0
CCP3CON	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

注 1： 在具有 32 KB 程序存储器的器件 （PIC18F65K22 和 PIC18F85K22）上未实现。
2： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

© 2009-2011 Microchip Technology Inc.

21.0 主同步串行端口 （MSSP）模块

21.1 主 SSP （MSSP）模块概述

主同步串行端口 （MSSP）模块是用于同其他外设或单片机进行通信的串行接口。这些外设包括串行 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 （SPI）
- I^2C™
  - 全主模式
  - 从模式 （广播地址呼叫）

I^2C 接口硬件上支持下列模式：

- 主模式
- 多主器件模式
- 带有 5 位和 7 位地址掩码的从模式 （具有用于 10 位或 7 位寻址的地址掩码）

PIC18F87K22 系列的所有器件都有两个 MSSP 模块，分别指定为 MSSP1 和 MSSP2。这两个模块独立工作。

> **注：** 在本章节中，在所有工作模式下，通常指的 MSSP 模块都可以解释为 MSSP1 或 MSSP2。在需要时，寄存器名称和模块 I/O 信号使用通用标识符 “x”（数字）来识别某个特定模块。控制位名称没有区别。

21.2 控制寄存器

每个 MSSP 模块都有 3 个相关的控制寄存器。其中包括一个状态寄存器 （SSPxSTAT）和两个控制寄存器 （SSPxCON1 和 SSPxCON2）。根据 MSSP 模块是在 SPI 模式还是 I^2C 模式下工作，这些寄存器的用途及其各自的配置位将完全不同。

下面各节提供了更多详细信息。

> **注：** 在具有多个 MSSP 模块的器件中，要特别注意 SSPxCON 寄存器的名称。SSP1CON1 和 SSP1CON2 控制同一模块工作的不同方面，而 SSP1CON1 和 SSP2CON1 控制两个不同模块的相同功能。

> **注：** 不能对 SSPxBUF 寄存器使用读 - 修改 - 写指令，如 BCF 和 COMF 等。
> 为了避免在主模式下丢失数据，在两次传输之间必须读 SSPxBUF，以将缓冲器满 （BF）检测位 （SSPxSTAT<0>）清零。

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。通常使用以下三个引脚来实现通信：

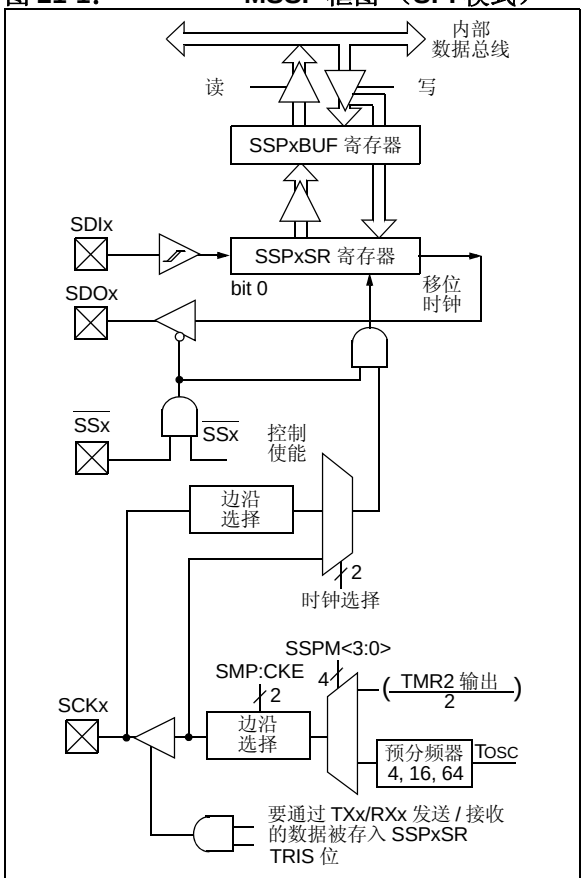
- 串行数据输出 （SDOx）——RC5/SDO1 或 RD4/PSP4/SDO2
- 串行数据输入 （SDIx）——RC4/SDI1/SDA1 或 RD5/PSP5/SDI2/SDA2
- 串行时钟 （SCKx）——RC3/SCK1/SCL1 或 RD6/PSP6/SCK2/SCL2

此外，当工作在从模式下时要使用第 4 根引脚：

- 从选择 （$\overline{SSx}$）——RF7/AN5/$\overline{SS1}$ 或 RD7/$\overline{SS2}$

图 21-1 给出了 MSSP 模块在 SPI 模式下工作时的原理图。

图 21-1： MSSP 框图 （SPI 模式）



注： 为了简便起见，本图中仅使用端口 I/O 名称。复用功能的完整列表请参见正文。

21.3.1 寄存器

每个 MSSP 模块都有四个寄存器用于 SPI 模式。这些寄存器包括：

- MSSPx 控制寄存器 1 （SSPxCON1）
- MSSPx 状态寄存器 （SSPxSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPxBUF）
- MSSPx 移位寄存器 （SSPxSR）——不可直接访问

SSPxCON1 和 SSPxSTAT 是 SPI 工作模式下的控制寄存器和状态寄存器。 SSPxCON1 寄存器是可读写的。SSPxSTAT 的低 6 位是只读位，而高 2 位是可读写的。

SSPxSR 是用来将数据移入或移出的移位寄存器。SSPxBUF 是缓冲寄存器，用于写入和读出数据字节。

接收时，SSPxSR 和 SSPxBUF 共同构成一个双重缓冲接收器。当 SSPxSR 接收到一个完整的字节后，该字节被送入 SSPxBUF 寄存器，同时将中断标志位 SSPxIF 置 1。

在发送过程中， SSPxBUF 并不是双重缓冲的。对 SSPxBUF 的写操作将同时写入 SSPxBUF 和 SSPxSR。

寄存器 21-1： SSPxSTAT：MSSPx 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 = 上电复位时的值	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端采样输入数据
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 = 当时钟状态从有效变为空闲时开始发送
0 = 当时钟状态从空闲变为有效时开始发送

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 MSSPx 模块时，该位被清零； SSPEN 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**：缓冲器满状态位 （仅用于接收模式）
1 = 接收完成， SSPxBUF 满
0 = 接收未完成， SSPxBUF 空

注 1： 时钟极性由 CKP 位 （SSPxCON1<4>）设置。

© 2011 Microchip Technology Inc.

寄存器 21-2： SSPxCON1：MSSPx 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 = 上电复位时的值	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 SSPxBUF 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 = 当 SSPxBUF 寄存器中仍然保留前一数据时，又接收到一个新字节。如果溢出，SSPxSR 中的数据会丢失。溢出只发生在从模式下。即使只是发送数据，用户也必须读 SSPxBUF，以避免将溢出标志位置 1 （必须用软件清零）。
0 = 无溢出

bit 5 **SSPEN：**主同步串行端口使能位 (2)
1 = 使能串行端口并将 SCKx、 SDOx、 SDIx 和 $\overline{\text{SSx}}$ 配置为串行端口引脚
0 = 禁止串行端口并将这些引脚配置为 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>：**主同步串行端口模式选择位 (3)
1010 = SPI 主模式：时钟 = Fosc/8
0101 = SPI 从模式：时钟 = SCKx 引脚；禁止 $\overline{\text{SSx}}$ 引脚控制；可将 $\overline{\text{SSx}}$ 用作 I/O 引脚
0100 = SPI 从模式：时钟 = SCKx 引脚；使能 $\overline{\text{SSx}}$ 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 SSPxBUF 寄存器启动的。
2： 当该位使能时，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为保留或只用于 I^2C™ 模式。

21.3.2 工作原理

当初始化 SPI 时，需要指定几个选项。可通过编程相应的控制位 （SSPxCON1<5:0> 和 SSPxSTAT<7:6>）来指定。这些控制位用于指定以下选项：

- 主模式 （SCKx 作为时钟输出）
- 从模式 （SCKx 作为时钟输入）
- 时钟极性 （SCKx 处于空闲状态）
- 输入数据的采样阶段 （数据输出时间的中间或末端）
- 时钟边沿 （在 SCKx 的上升沿 / 下降沿输出数据）
- 时钟速率 （仅用于主模式）
- 从选择模式 （仅用于从模式）

每个 MSSP 模块都由一个发送 / 接收移位寄存器（SSPxSR）和一个串行输入缓冲寄存器 （SSPxBUF）组成。SSPxSR 将数据移入 / 移出器件，最高有效位在前。在新数据接收完毕前， SSPxBUF 保存上次写入 SSPxSR 的数据。一旦 8 位数据接收完毕，该字节就被移入 SSPxBUF 寄存器。然后，缓冲器满检测位 BF（SSPxSTAT<0>）和中断标志位 SSPxIF 均被置 1。这种双重缓冲数据接收方式 （使用 SSPxBUF）允许在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，任何写 SSPxBUF 寄存器的操作均无效，并且将写冲突检测位 WCOL（SSPxCON1<7>）置 1。用户必须用软件将 WCOL 位清零才能判断以后对 SSPxBUF 寄存器的写入是否成功。

为确保应用软件能有效地接收数据，应该在下一个要发送的数据字节写入 SSPxBUF 之前，读取 SSPxBUF 中现有的数据。缓冲器满标志位 BF （SSPxSTAT<0>）用于表示何时 SSPxBUF 载入了已接收的数据 （发送完成）。当 SSPxBUF 中的数据被读取后，BF 位即被清零。如果 SPI 仅仅作为一个发送器，则不必理会该数据。通常可用 MSSP 中断来判断发送 / 接收何时完成。如果不使用中断来处理数据的收发，用软件查询的方法同样可确保不会发生写冲突。例 21-1 说明了装载 SSPxBUF（SSPxSR）进行数据发送的过程。

用户不能直接读写 SSPxSR 寄存器，只能通过寻址 SSPxBUF 寄存器来访问。此外， SSPxSTAT 寄存器用于指示各种状态。

21.3.3 漏极开路输出选项

可选择将 SDOx 输出和 SCKx 时钟引脚的驱动器配置为漏极开路输出。此功能允许通过外部上拉电阻将引脚上的电平拉至更高的电平，以允许输出与外部电路通信，而无需额外的电平转换器。更多详细信息，请参见**第 12.1.3 节 “漏极开路输出”**。

漏极开路输出选项由 SSP2OD 位和 SSP1OD 位（ODCON3<1:0>）控制。置 1 SSPxOD 位可将对应模块的 SDOx 和 SCKx 引脚配置为漏极开路操作。

> **注：** 为了避免在主模式下丢失数据，在两次传输之间必须读 SSPxBUF，以将缓冲器满（BF）检测位 （SSPxSTAT<0>）清零。

例 21-1： 装载 SSP1BUF （SSP1SR）寄存器

```
LOOP    BTFSS   SSP1STAT, BF    ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSP1BUF, W      ;WREG reg = contents of SSP1BUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSP1BUF         ;New data to xmit
```


© 2011 Microchip Technology Inc.

21.3.4 使能 SPI I/O

要使能串行端口，MSSP 使能位 SSPEN（SSPxCON1<5>）必须置 1。要复位或重新配置 SPI 模式，应先将 SSPEN 位清零，重新初始化 SSPxCON 寄存器，然后将 SSPEN 位置 1。这样就将把 SDIx、SDOx、SCKx 和 SSx 引脚配置为串行端口引脚。要让上述引脚充当串行端口功能，必须按如下说明正确设置其中一些引脚的数据方向位（在 TRIS 寄存器中）：

- 必须将 SDIx 对应的 TRISC<4> 或 TRISD<5> 位置 1
- 必须将 SDOx 对应的 TRISC<5> 或 TRISD<4> 位清零
- 必须将 SCKx（主模式）对应的 TRISC<3> 或 TRISD<6> 位清零
- 必须将 SCKx（从模式）对应的 TRISC<3> 或 TRISD<6> 位置 1
- 必须将 SSx 对应的 TRISF<7> 或 TRISD<7> 位置 1

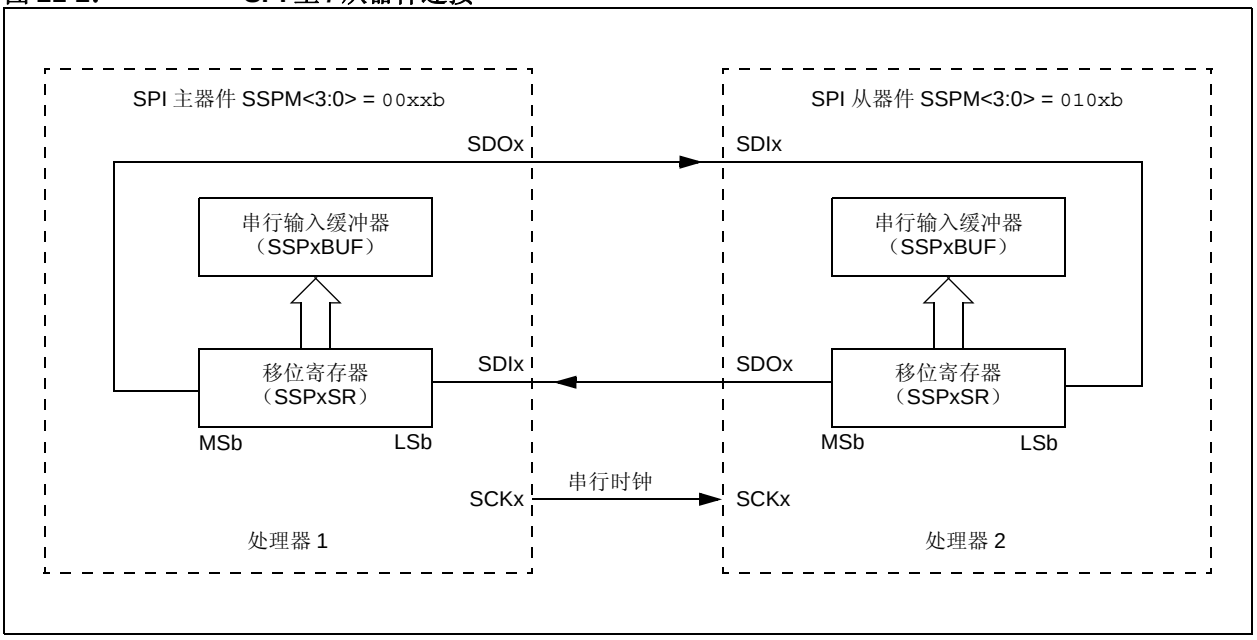
对于任何不需要的串行端口功能，可通过将对应的数据方向（TRIS）寄存器设置为相反值来屏蔽。

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 SCKx 信号来启动数据传输。在两个处理器的移位寄存器之间，数据在编程设定的时钟边沿被传送，并在相反的时钟边沿被锁存。必须将两个处理器的时钟极性（CKP）设定为相同，这样两个控制器就可以同时收发数据。数据是否有效取决于应用软件。这就导致以下三种数据传输场景：

- 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（Dummy）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 SCKx 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）应在何时广播数据。

在主模式下，数据一旦写入 SSPxBUF 寄存器就开始发送或接收。如果 SPI 仅作为接收器，则可以禁止 SDOx 输出（通过编程将其设置为输入）。SSPxSR 寄存器按设定的时钟速率连续移入 SDIx 引脚上的信号。每收到一个字节，就将其装入 SSPxBUF 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“线路活动监视器”（Line Activity Monitor）方式工作的接收器应用中很有用。

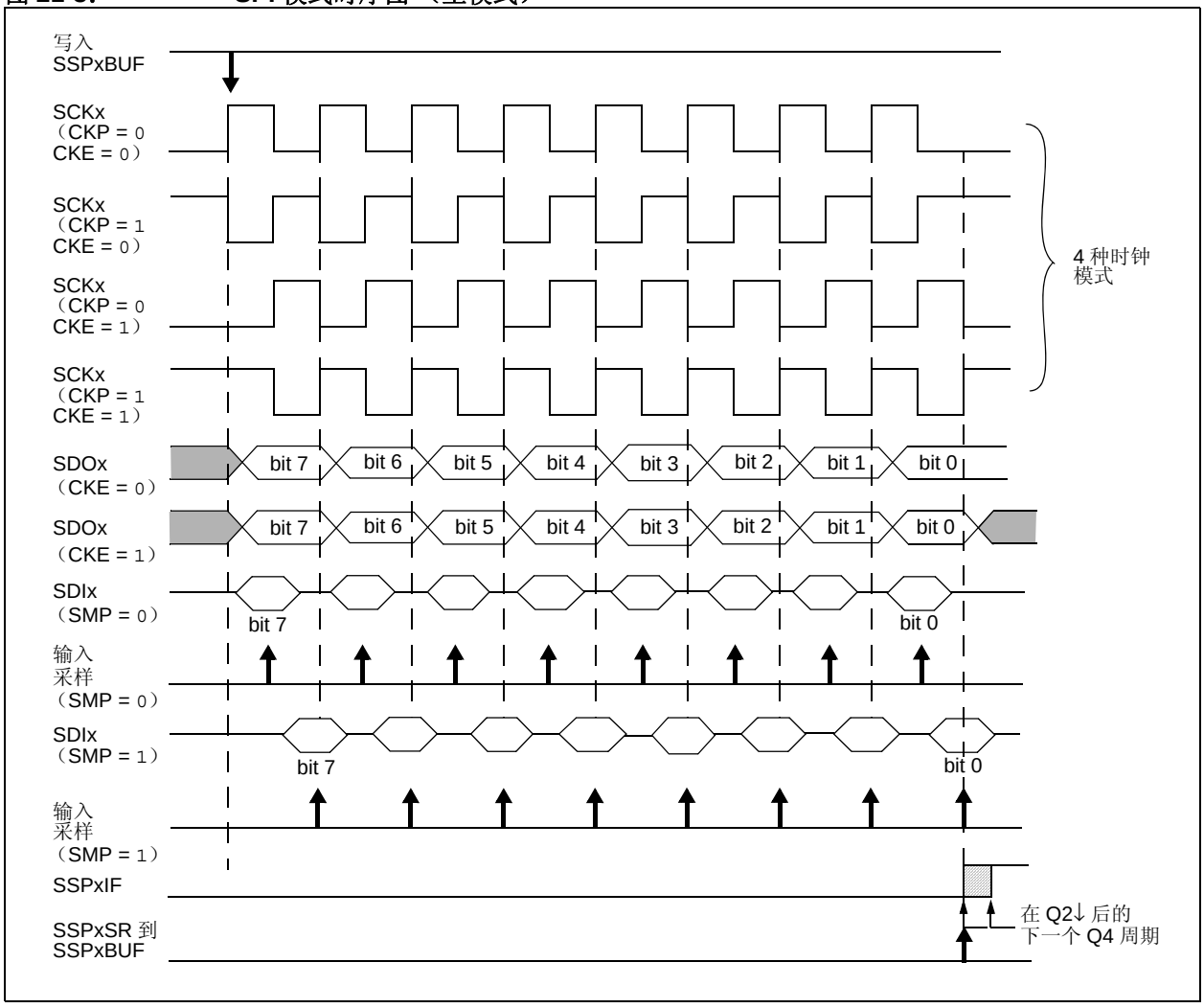
可通过对 CKP 位（SSPxCON1<4>）进行适当的编程来选择时钟极性。图 21-3、图 21-5 和图 21-6 将给出 SPI 通信的时序图，其中最先发送最高有效位。在主模式下，SPI 时钟速率（比特率）可由用户编程设定为下面某种速率：

- 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为 10.00 Mbps（时钟速率为 40 MHz）。

图 21-3 给出了主模式的时序图。当 CKE 位置 1 时，SDOx 数据在 SCKx 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 位的状态决定。图中给出了接收到的数据装入 SSPxBUF 的时间。

图 21-3： SPI 模式时序图（主模式）



© 2011 Microchip Technology Inc.

21.3.7 从模式

在从模式下，当 SCKx 引脚上有外部时钟脉冲出现时发送和接收数据。当最后一位数据被锁存后，中断标志位 SSPxIF 置 1。

在从模式下，外部时钟由 SCKx 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。当收到一个字节时，可将器件配置为从休眠模式下唤醒。

21.3.8 从选择同步

$\overline{\text{SSx}}$ 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 $\overline{\text{SSx}}$ 引脚控制功能（SSPxCON1<3:0> = 04h）。当 $\overline{\text{SSx}}$ 引脚为低电平时，使能数据的发送和接收，同时驱动 SDOx 引脚。当 $\overline{\text{SSx}}$ 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 SDOx 引脚，而是将其变成悬空输出状态。根据应用需要，可在 SDOx 引脚上外接上拉 / 下拉电阻。

> **注：** 当 SPI 处于从模式，并且 $\overline{\text{SSx}}$ 引脚控制功能使能（SSPxCON1<3:0> = 0100）时，如果 $\overline{\text{SSx}}$ 引脚设置为 V_{DD} 电平，将使 SPI 模块复位。
>
> 如果 SPI 工作在从模式下并且 CKE 置 1，则必须使能 $\overline{\text{SSx}}$ 引脚控制功能。

当 SPI 模块复位后，位计数器被强制为 0。这可以通过强制将 $\overline{\text{SSx}}$ 引脚拉为高电平或将 SSPEN 位清零来实现。

将 SDOx 引脚和 SDIx 引脚相连，可以仿真二线制通信。当 SPI 需要作为接收器工作时，SDOx 引脚可以被配置为输入。这样就禁止了从 SDOx 发送数据。因为 SDIx 不会引起总线冲突，因而总是可以将其保留为输入（SDIx 功能）。

图 21-4： 从同步模式时序图

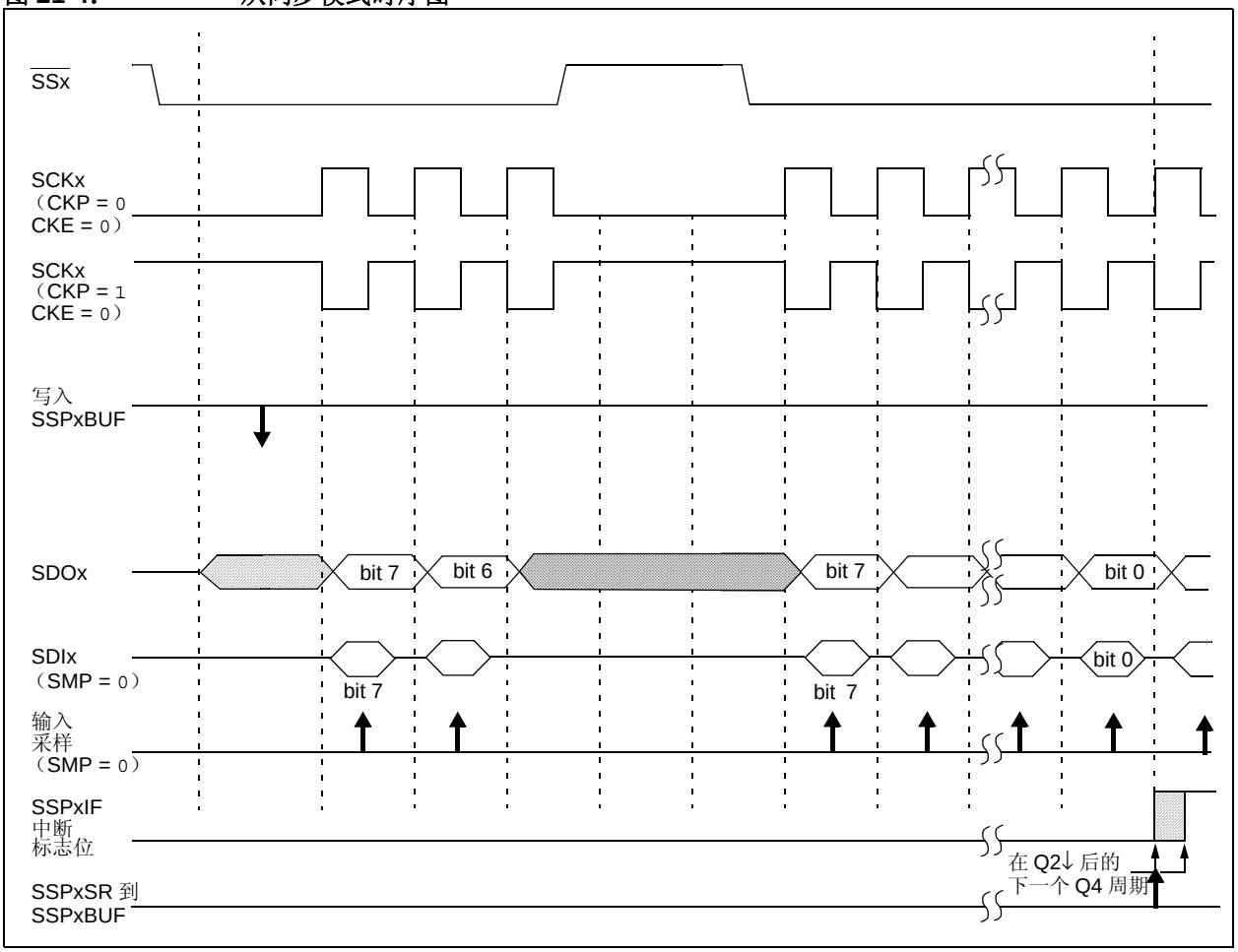


图 21-5： SPI 模式时序图 （从模式， CKE = 0）

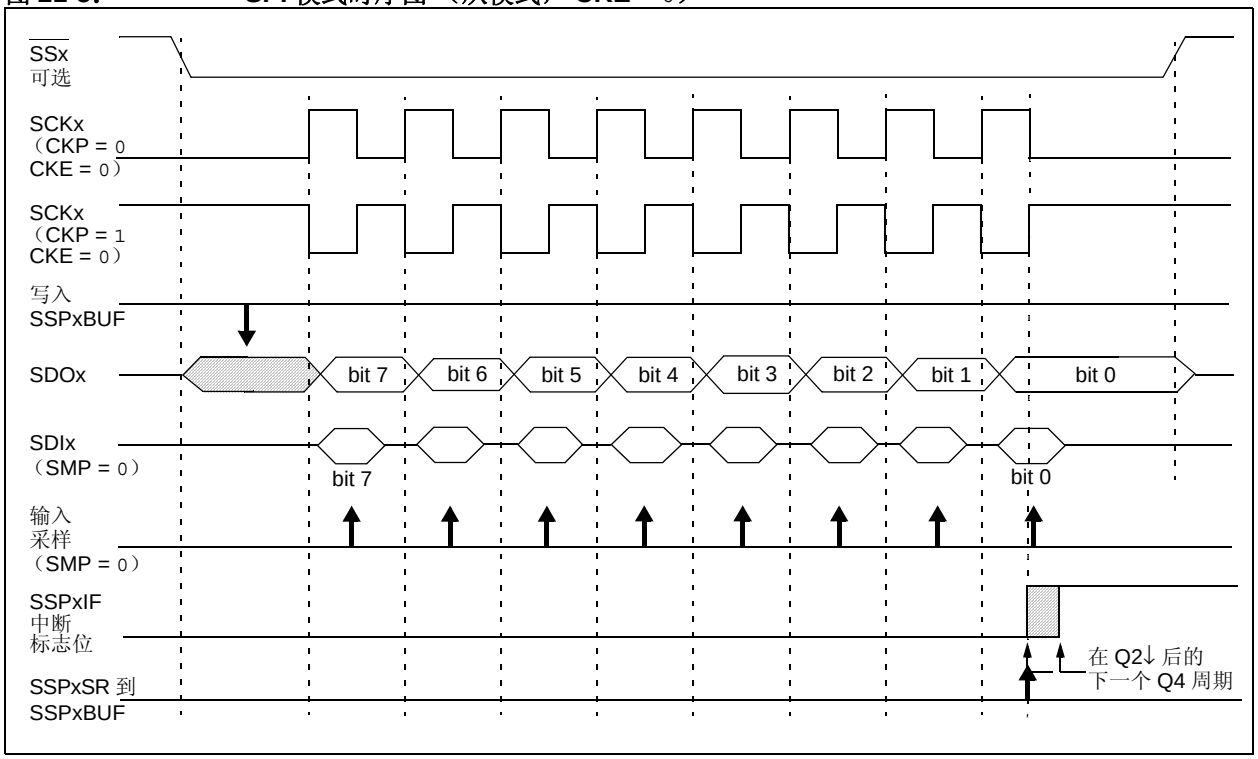
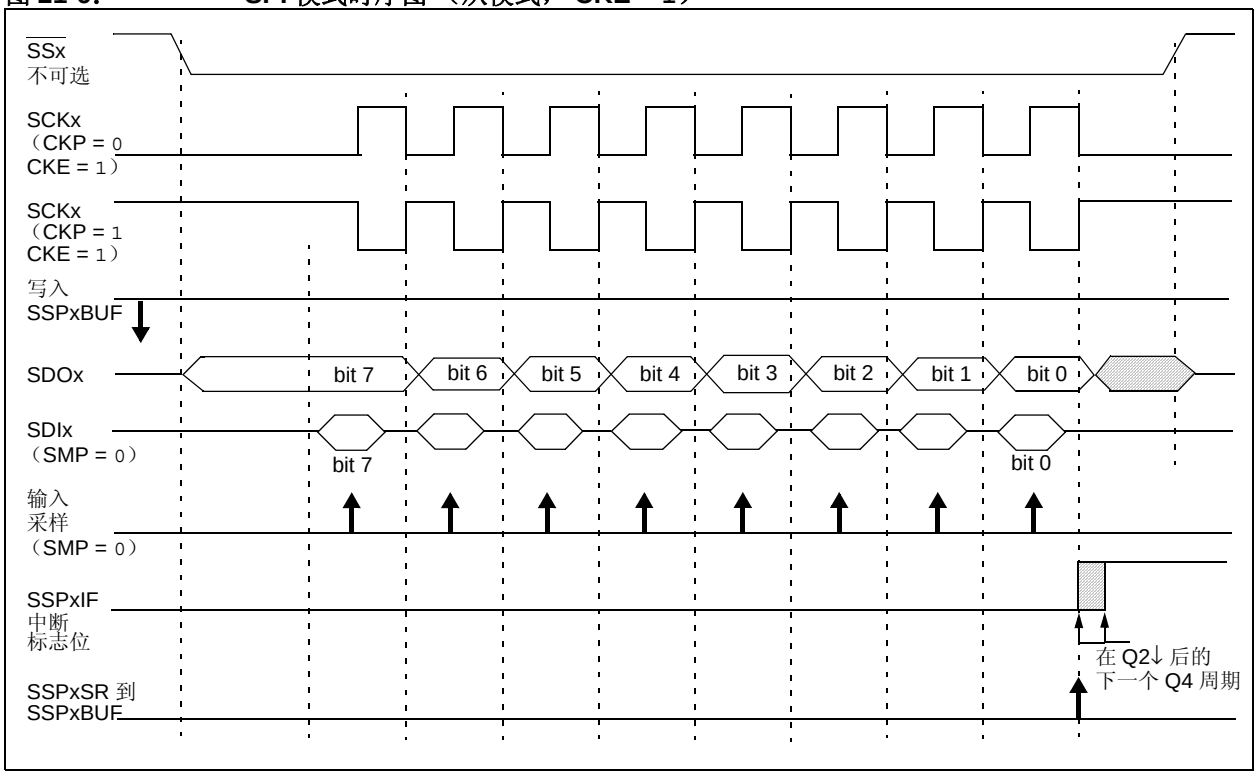


图 21-6： SPI 模式时序图 （从模式， CKE = 1）



© 2011 Microchip Technology Inc.

21.3.9 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。该时钟可以来自于主时钟源、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 “时钟源和振荡器切换”**。

在大多数情况下，主器件为 SPI 数据提供的时钟速率并不重要；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 / 接收将保持此停滞状态。当器件返回到运行模式后，该模块将恢复发送和接收数据。

在 SPI 从模式下， SPI 发送 / 接收移位寄存器与器件异步工作。这可以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。当 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，还将唤醒器件。

21.3.10 复位的影响

复位会禁止 MSSP 模块并终止当前的传输。

21.3.11 总线模式兼容性

表 21-1 中所示是标准 SPI 模式与 CKP 和 CKE 控制位状态的对应关系。

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 位用来控制何时采样数据。

21.3.12 SPI 时钟速率和模块的相互关系

由于 MSSP1 和 MSSP2 是独立的模块，因此可以不同的数据速率同时运行。设置 SSPxCON1 寄存器的 SSPM<3:0> 位可确定对应模块的速率。

有一种例外情况就是在主模式下两个模块都使用 Timer2 作为时基时。在这种情况下，任何对 Timer2 模块工作的更改都会对两个 MSSP 模块造成相同的影响。如果每个模块需要不同的比特率，用户应该为一个模块选择其他三种时基选项之一。

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
PIR2	OSCFIF	—	SSP2IF	BCL2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	SSP2IE	BCL2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
IPR2	OSCFIP	—	SSP2IP	BCL2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—
SSP1BUF	MSSP1 接收缓冲器 / 发送寄存器							
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
SSP2BUF	MSSP2 接收缓冲器 / 发送寄存器							
ODCON1	SSP1OD	CCP2OD	CCP1OD	—	—	—	—	SSP2OD
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： SPI 模式下的 MSSP 模块不使用阴影单元。

© 2011 Microchip Technology Inc.

21.4 I²C™ 模式

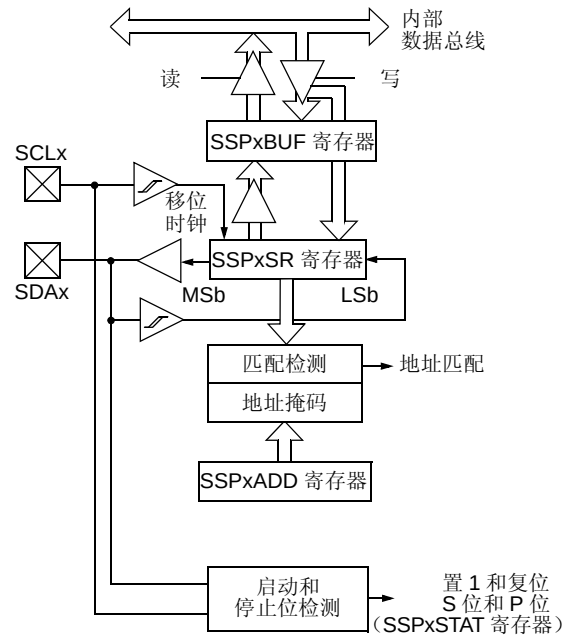
MSSP 模块工作在 I²C 模式时，可以实现所有的主和从功能（包括广播呼叫支持），并且用硬件提供启动位和停止位的中断来判断总线何时空闲（多主器件功能）。MSSP 模块实现了标准模式规范以及 7 位和 10 位寻址。

有两个引脚用于数据传输：

- 串行时钟（SCLx）—RC3/SCK1/SCL1 或 RD6/SCK2/SCL2
- 串行数据（SDAx）—RC4/SDI1/SDA1 或 RD5/SDI2/SDA2

用户必须通过将相关 TRIS 位置 1 将上述引脚配置为输入。

图 21-7： MSSP 框图（I²C™ 模式）



注： 为了简便起见，本图中仅使用端口 I/O 名称。复用功能的完整列表请参见文本。

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：

- MSSPx 控制寄存器 1（SSPxCON1）
- MSSPx 控制寄存器 2（SSPxCON2）
- MSSPx 状态寄存器（SSPxSTAT）
- 串行接收 / 发送缓冲寄存器（SSPxBUF）
- MSSPx 移位寄存器（SSPxSR）—不可直接访问
- MSSPx 地址寄存器（SSPxADD）
- I²C 从地址掩码寄存器（SSPxMSK）

SSPxCON1、SSPxCON2 和 SSPxSTAT 是I²C 模式下的控制寄存器和状态寄存器。 SSPxCON1 和 SSPxCON2 寄存器是可读写的。 SSPxSTAT 的低 6 位是只读位，而高 2 位是可读写的。

SSPxSR 是用来将数据移入或移出的移位寄存器。SSPxBUF 是缓冲寄存器，用于写入和读出数据字节。

SSPxADD 包含 MSSP 被配置为 I²C 从模式时的从器件地址。MSSP 被配置为主模式时，SSPxADD 的低 7 位用作波特率发生器的重载值。

模块被配置为 7 位地址掩码模式时，SSPxMSK 保存从地址掩码值。该寄存器作为独立寄存器时，将与 SSPxADD 共享同一 SFR 地址；只有对 SSPM<3:0> 位进行特定的设置以允许访问时，才可以访问它。欲知更多详细信息，请参见第 21.4.3.4 节 “7 位地址掩码模式”。

接收时，SSPxSR 和 SSPxBUF 共同构成一个双重缓冲接收器。当 SSPxSR 接收到一个完整的字节后，该字节被送入 SSPxBUF 寄存器，同时将中断标志位 SSPxIF 置 1。

在发送过程中， SSPxBUF 并不是双重缓冲的。对 SSPxBUF的写操作将同时写入SSPxBUF和SSPxSR。

寄存器 21-3： SSPxSTAT：MSSPx 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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **SMP：**压摆率控制位
在主或从模式下：
1 = 标准速率模式 （100 kHz 和 1 MHz）下禁止压摆率控制
0 = 高速模式 （400 kHz）下使能压摆率控制

bit 6 **CKE：**SMBus 选择位
在主或从模式下：
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 = 表示上一个接收或发送的字节是数据
0 = 表示上一个接收或发送的字节是地址

bit 4 **P：**停止位 (1)
1 = 表示上一次检测到了停止位
0 = 表示上一次未检测到停止位

bit 3 **S：**启动位 (1)
1 = 表示上一次检测到了启动位
0 = 表示上一次未检测到启动位

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 （仅用于 10 位从模式）
1 = 表示用户需要更新 SSPxADD 寄存器中的地址
0 = 不需要更新地址

bit 0 **BF：**缓冲器满状态位
在发送模式下：
1 = SSPxBUF 满
0 = SSPxBUF 空
在接收模式下：
1 = SSPxBUF 满 （不包括 $\overline{ACK}$ 位和停止位）
0 = SSPxBUF 空 （不包括 ACK 位和停止位）

注 1： SSPEN 被清零时该位在复位时清零。
2： 该位用来保存在最近一次地址匹配后的 R/$\overline{W}$ 位信息。该位仅在从地址匹配开始到下一个启动位、停止位或非 $\overline{ACK}$ 位期间有效。
3： 该位与 SEN、RSEN、PEN、RCEN 或 ACKEN 进行逻辑“或”运算的结果表示 MSSPx 是否处于工作模式。

© 2011 Microchip Technology Inc.

寄存器 21-4： SSPxCON1：MSSPx 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 = 上电复位时的值	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 SSPxBUF 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 SSPxBUF 寄存器 （必须用软件清零）

0 = 未发生冲突

在接收模式下 （主或从模式）：

此位为无关位。

bit 6 **SSPOV**：接收溢出指示位

在接收模式下：

1 = SSPxBUF 寄存器仍在保存前一字节时，又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，此位为无关位。

bit 5 **SSPEN**：主同步串行端口使能位 (1)

1 = 使能串行端口并将 SDAx 和 SCLx 引脚配置为串行端口引脚

0 = 禁止串行端口并将这些引脚配置为 I/O 端口引脚

bit 4 **CKP**：SCKx 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>**：主同步串行端口模式选择位 (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 由固件控制的主模式 （从器件空闲）

1001 = 将 SSPxADD SFR 地址处的内容装入 SSPMSK 寄存器 (3,4)

1000 = I²C 主模式：时钟 = Fosc/(4 * (SSPxADD + 1))

0111 = I²C 从模式：10 位地址

0110 = I²C 从模式：7 位地址

注 **1**： 当该位被使能时，必须将 SDAx 和 SCLx 引脚配置为输入引脚。

2： 此处未列出的位组合为保留或只用于 SPI 模式。

3： 当 SSPM<3:0> = 1001 时，对 SSPxADD SFR 地址的任何读或写实际上是访问 SSPxMSK 寄存器。

4： 只有在选择了 7 位地址掩码模式时 （MSSPMSK 配置位为 1），此模式才可用。

寄存器 21-5： SSPxCON2：MSSPx 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 = 上电复位时的值	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 （仅用于主发送模式）
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 （仅用于主接收模式） (1)
1 = 无应答
0 = 应答

bit 4 **ACKEN：**应答序列使能位 (2)
1 = 在 SDAx 和 SCLx 引脚上发起应答序列，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 （仅用于主接收模式） (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 SDAx 和 SCLx 引脚上发起停止条件。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 SDAx 和 SCLx 引脚上发起重复启动条件。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 SDAx 和 SCLx 引脚上发起启动条件。由硬件自动清零。
0 = 启动条件空闲

注 1： 用户在接收结束时发起一个应答序列，同时发送该值。
2： 如果 I²C 模块处于工作状态，这些位可能不被置 1 （没有缓存），并且可能不会写入 SSPxBUF （或者禁止写 SSPxBUF）。

© 2011 Microchip Technology Inc.

寄存器 21-6： SSPxCON2：MSSPx 控制寄存器 2 （I²C™ 从模式）

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(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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **GCEN：**广播呼叫使能位
1 = 允许在 SSPxSR 接收到广播呼叫地址 （0000h）时产生中断
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 （仅用于主接收模式）(1)
1 = 无应答
0 = 应答

bit 4 **ACKEN：**应答序列使能位 (1)
1 = 在 SDAx 和 SCLx 引脚上发起应答序列，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 （仅用于主接收模式）(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 SDAx 和 SCLx 引脚上发起停止条件。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 SDAx 和 SCLx 引脚上发起重复启动条件。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 = 为从发送和从接收使能时钟延长 （已使能时钟延长）
0 = 时钟延长被禁止

注 1： 如果 I²C 模块处于工作状态，此位可能不被置 1 （没有缓存），并且也可能不会写入 SSPxBUF （或者禁止写 SSPxBUF）。

寄存器 21-7： SSPxMSK：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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-0 **MSK<7:0>：**从地址掩码选择位
1 = 使能掩码 SSPxADD 的相应位
0 = 禁止掩码 SSPxADD 的相应位

注 1： 此寄存器与 SSPxADD 共享同一 SFR 地址，并且仅在选定的 MSSPx 工作模式下才可寻址。更多详细信息，请参见第 21.4.3.4 节 “7 位地址掩码模式”。
2： MSK0 不用作 7 位寻址模式下的掩码位。

21.4.2 工作原理

通过将 MSSP 使能位 SSPEN（SSPxCON1<5>）置 1，可使能 MSSP 模块。

SSPxCON1 寄存器用于控制 I^2C的工作模式。可通过设置 4 个模式选择位 （SSPxCON1<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 由固件控制的主模式，从器件空闲

通过将相应的 TRISC 或 TRISD 位置 1，将 SCLx 和 SDAx 引脚编程为输入引脚；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 SCLx 和 SDAx 引脚提供外接上拉电阻。

21.4.3 从模式

在从模式下， SCLx 引脚和 SDAx 引脚必须被配置为输入 （TRISC<4:3> 置 1）。必要时， MSSP 模块将使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。地址掩码还允许硬件在多个地址发生匹配时 （7 位寻址模式下最多 31 个地址， 10 位寻址模式下最多 63 个地址）产生中断。用户也可以通过模式选择位，选择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（ACK）脉冲，并把当前 SSPxSR 寄存器中接收到的数据装入 SSPxBUF 寄存器。

只要满足下列条件之一，MSSP 模块就不会产生此 $\overline{ACK}$ 脉冲：

- 在接收到传输的数据前，缓冲器满标志位 BF （SSPxSTAT<0>）置 1。
- 在接收到传输的数据前，溢出位 SSPOV （SSPxCON<6>）置 1。

在这种情况下， SSPxSR 寄存器的值不会被装入 SSPxBUF，但是 SSPxIF 位会置 1。 BF 位是通过读取 SSPxBUF 寄存器清零的，而 SSPOV 位通过软件清零。

为确保正常工作， SCLx 时钟输入必须满足最小高电平和最小低电平时间要求。关于 I^2C 规范的高电平和低电平时间以及对 MSSP 模块的具体要求，请参见时序参数 100 和 101。

21.4.3.1 寻址

一旦使能了 MSSP 模块，它就会等待启动条件出现。在启动条件出现后，8 位数据被移入 SSPxSR 寄存器。在时钟线 （SCLx）的上升沿采样所有的输入位。寄存器 SSPxSR<7:1> 的值会和 SSPxADD 寄存器的值进行比较。地址会在第 8 个时钟 （SCLx）脉冲的下降沿进行比较。如果地址匹配，并且 BF 位和 SSPOV 位都被清零，则会发生下列事件：

1. SSPxSR 寄存器的值被装入 SSPxBUF 寄存器。
2. 缓冲器满标志位 BF 置 1。
3. 产生 $\overline{ACK}$ 脉冲。
4. 在第 9 个 SCLx 脉冲的下降沿，MSSP 中断标志位 SSPxIF 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五位 （MSb）指定这是否为 10 位地址。 R/W 位 （SSPxSTAT<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` 是该地址的两个最高有效位。10 位寻址的操作步骤如下，其中第 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 （地址匹配时 SSPxIF 位、 BF 位和 UA 位置 1）。
2. 用地址的第二个（低）字节更新 SSPxADD 寄存器 （UA 位清零并释放 SCLx 线）。
3. 读 SSPxBUF 寄存器 （BF 位清零）并将标志位 SSPxIF 清零。
4. 接收地址的第二个（低）字节（SSPxIF 位、BF 位和 UA 位置 1）。
5. 使用地址的第一个（高）字节更新 SSPxADD 寄存器。如果匹配的话就释放 SCLx 线，这将清零 UA 位。
6. 读 SSPxBUF 寄存器 （BF 位清零）并将标志位 SSPxIF 清零。
7. 接收重复启动条件。
8. 接收地址的第一个（高）字节（SSPxIF 位和 BF 位置 1）。
9. 读 SSPxBUF 寄存器 （BF 位清零）并将标志位 SSPxIF 清零。

© 2011 Microchip Technology Inc.

21.4.3.2 地址掩码模式

将地址的某一位掩码意味着该位可以为任意值。若将某个地址位掩码，则将应答两个地址并产生中断。由于可以同时掩码多个地址位，这样便大大扩展了被应答地址的数量。

无论是否使用地址掩码，I^2C 从器件的操作方式都不会改变。但是，当使用了地址掩码时，I^2C 从器件可应答多个地址并产生中断。当出现这样的情况时，需要通过检查 SSPxBUF 来确定是哪个地址引起的中断。

在 I^2C 从工作模式下，PIC18F87K22 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。在器件配置时使用 MSSPMSK 配置位来选择掩码模式。默认的器件配置是 7 位地址掩码。

这两种掩码模式支持 7 位和 10 位地址的掩码。将为每种掩码模式和地址的组合提供不同的可应答地址范围。

两种掩码模式的工作方式大致相同，但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

正如名称所示，5 位地址掩码模式使用最多 5 位的地址掩码，用传入地址的 bit 5 至 bit 1 来产生可被应答的地址范围。这允许该模块在使用 7 位寻址模式时最多应答 31 个地址，在使用 10 位寻址模式时最多应答 63 个地址（见例 21-2）。编程 MSSPMSK 配置位（0）时，将选择此掩码模式。

此模式下的地址掩码存储在 SSPxCON2 寄存器中，该寄存器在 I^2C 从模式下不作为控制寄存器（寄存器21-6）工作。在 7 位地址掩码模式下，地址掩码位 ADMSK<5:1>（SSPxCON2<5:1>）掩码 SSPxADD 寄存器中的相应地址位。对于任何置 1 的 ADMSK 位（ADMSK<n> = 1），将忽略相应的地址位（SSPxADD<n> = x）。要使模块发出地址应答，只要与没被掩码的地址位匹配就可以了。

在 10 位地址掩码模式下，ADMSK<5:2> 位掩码 SSPxADD 寄存器中相应的地址位。此外，ADMSK1 同时掩码该地址的低 2 位（SSPxADD<1:0>）。对于任何置 1 的 ADMSK 位（ADMSK<n> = 1），将忽略相应的地址位（SSPxADD<n> = x）。另请注意，虽然在 10 位地址掩码模式下，高地址位会复用 SSPxADD 寄存器位的某几位，但是地址掩码位不会影响这些位；而只影响低地址位。

> **注 1：** ADMSK1 掩码该地址的低 2 位。
>
> **2：** 地址掩码不会对该地址的高 2 位产生影响。

例 21-2：地址掩码示例（5 位掩码模式）

7 位寻址：

SSPxADD<7:1>= A0h（1010000）（SSPxADD<0> 假定为 0）

ADMSK<5:1> = 00111

应答的地址：A0h、A2h、A4h、A6h、A8h、AAh、ACh 和 AEh

10 位寻址：

SSPxADD<7:0> = A0h（10100000）（由于掩码不会对地址的高 2 位产生影响，所以在本示例中将其忽略）

ADMSK<5:1> = 00111

应答的地址：A0h、A1h、A2h、A3h、A4h、A5h、A6h、A7h、A8h、A9h、AAh、ABh、ACh、ADh、AEh 和 AFh

21.4.3.4 7 位地址掩码模式

与 5 位掩码不同，7 位地址掩码模式使用最多 8 位的掩码（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。这允许该模块在使用 7 位寻址模式时最多应答 127 个不同地址，在使用 10 位寻址模式时最多应答 255 个地址（见例 21-3）。此模式是该模块的默认配置，在未编程 MSSPMSK （1）时选择此模式。

7 位地址掩码模式的地址掩码存储在 SSPxMSK 中，而不是 SSPxCON2 寄存器中。 SSPxMSK 是模块中的独立硬件寄存器，但是不可直接寻址。而是与 SSPxADD 寄存器共用 SFR 空间中的一个地址。要访问 SSPxMSK 寄存器，需要选择 MSSP 模式 1001（SSPxCON1<3:0> = 1001），然后读 / 写 SSPxADD 的单元。

要使用 7 位地址掩码模式，需要在选择 I^2C 从寻址模式前使用一个值来初始化 SSPxMSK。因此，所需操作步骤如下：

1. 选择 SSPxMSK 访问模式 （SSPxCON2<3:0> = 1001）。
2. 将掩码值写入正确的 SSPADD 寄存器地址（MSSP1 为 FC8h， MSSP2 为 F6Eh）。
3. 设置适当的 I^2C 从模式 （10 位寻址模式下为 SSPxCON2<3:0> = 0111， 7 位寻址模式下位 0110）。

置 1 或清零 SSPxMSK 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 SSPxMSK 中的某个位会导致相应的地址位被掩码；置 1 该位则要求相应的位匹配。任何复位后 SSPxMSK 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。

使用 7 位寻址模式时， SSPxMSK<7:1> 位将掩码 SSPxADD 寄存器中相应的地址位。对于任何有效的 SSPxMSK 位 （SSPxMSK<n> = 0），将忽略相应的SSPxADD地址位（SSPxADD<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。

使用 10 位寻址模式时， SSPxMSK<7:0> 位将掩码 SSPxADD 寄存器中相应的地址位。对于任何有效的 SSPxMSK 位（= 0），将忽略相应的 SSPxADD 地址位（SSPxADD<n> = x）。

注： 地址掩码不会对该地址的高2位产生影响。

例 21-3： 地址掩码示例 （7 位掩码模式）

7 位寻址：

SSPxADD<7:1> = 1010 000

SSPxMSK<7:1> = 1111 001

应答的地址 = ACh、 A8h、 A4h 和 A0h

10 位寻址：

SSPxADD<7:0> = 1010 0000 （由于掩码不会对地址的高 2 位产生影响，因此在本示例中将其忽略）

SSPxMSK<5:1> = 1111 0011

应答的地址 = ACh、 A8h、 A4h 和 A0h

© 2011 Microchip Technology Inc.

21.4.3.5 接收

当地址字节的 R/$\overline{W}$ 位清零且地址匹配时，SSPxSTAT 寄存器的 R/W 位清零。接收的地址被装入 SSPxBUF 寄存器，且 SDAx 信号线保持低电平 （$\overline{ACK}$）。

当存在地址字节溢出时，会产生无应答脉冲 （$\overline{ACK}$）。溢出条件是指 BF 位 （SSPxSTAT<0>）置 1，或者 SSPOV 位 （SSPxCON1<6>）置 1。

每个数据传输字节都会产生一个 MSSP 中断。必须用软件将中断标志位 SSPxIF 清零。使用 SSPxSTAT 寄存器可以确定该字节的状态。

如果 SEN 被使能 （SSPxCON2<0> = 1）， SCLx 将在每个数据传输之后保持为低电平 （时钟延长）。必须将 CKP 位（SSPxCON1<4>）置 1 才能释放时钟。更多详细信息，请参见**第 21.4.4 节 “时钟延长”**。

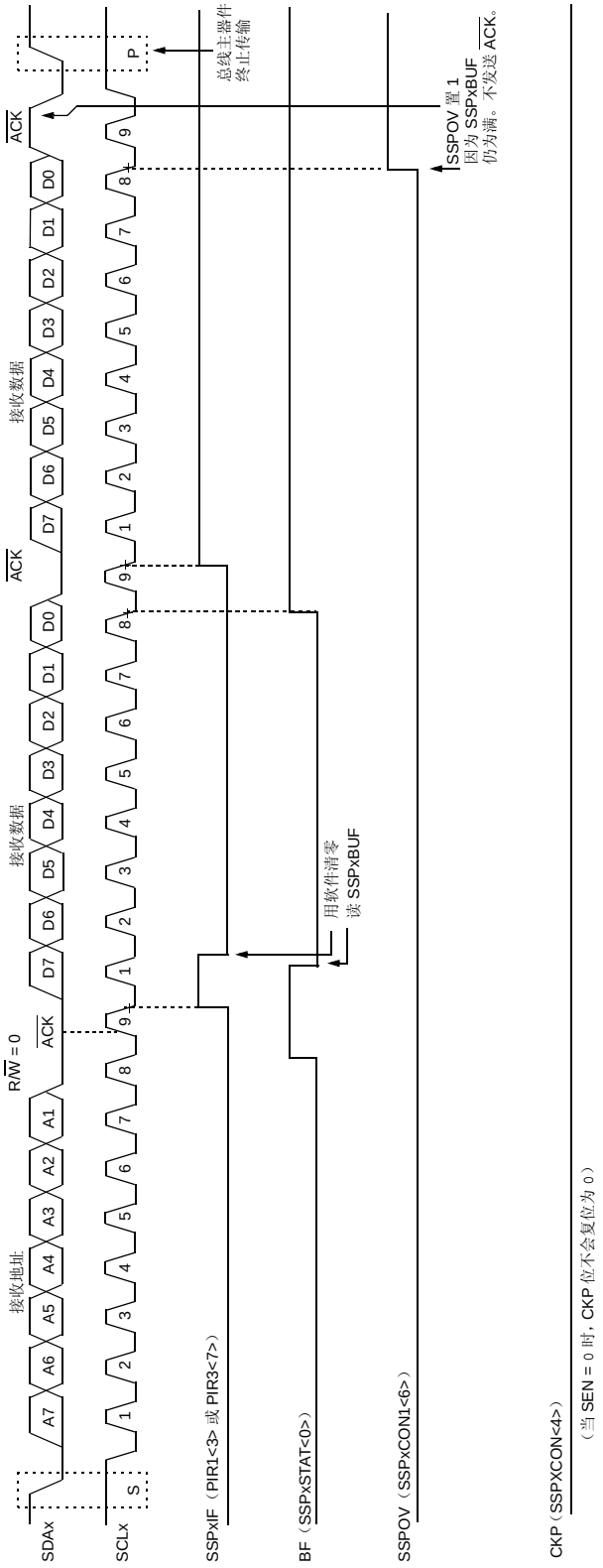
21.4.3.6 发送

当输入的地址字节的 R/$\overline{W}$ 位置 1 并发生地址匹配时，SSPxSTAT 寄存器的 R/W 位置 1。接收到的地址被装入 SSPxBUF 寄存器。$\overline{ACK}$ 脉冲在第 9 位上发送，同时不管 SEN 的值如何，SCLx 都将保持低电平（如需了解更多详细信息，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能发送另一个时钟脉冲。发送的数据必须装入 SSPxBUF 寄存器，同时也被装入 SSPxSR 寄存器。然后，通过将 CKP 位（SSPxCON1<4>）置 1 来使能 SCLx 引脚。8 个数据位在 SCLx 输入的下降沿被移出。这可确保在 SCLx 为高电平期间 SDAx 信号是有效的（图 21-10）。

主接收器发出的 $\overline{ACK}$ 脉冲将在第 9 个 SCLx 输入脉冲的上升沿被锁存。如果 SDAx 线为高电平 （无 ACK 应答信号），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 SDAx 线为低电平（$\overline{ACK}$），则必须将下一个要发送的数据装入 SSPxBUF 寄存器。同时，必须通过将 CKP 位置 1 来使能 SCLx。

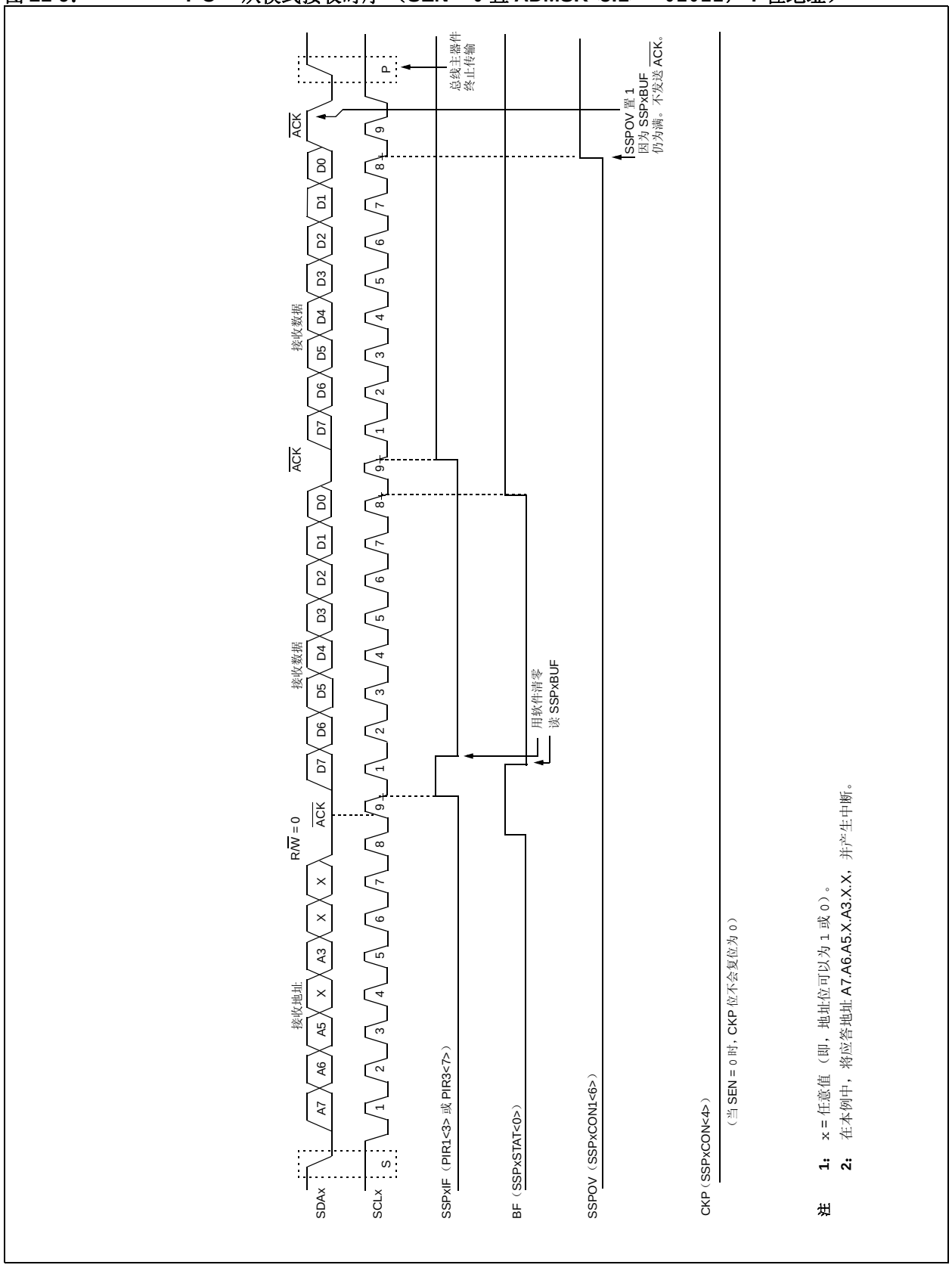
每个数据传输字节都会产生一个 MSSP 中断。 SSPxIF 位必须用软件清零，SSPxSTAT 寄存器用于确定字节的状态。 SSPxIF 位在第 9 个时钟脉冲的下降沿被置 1。

图 21-8： I²C™ 从模式接收时序 （SEN = 0， 7 位地址）



© 2011 Microchip Technology Inc.

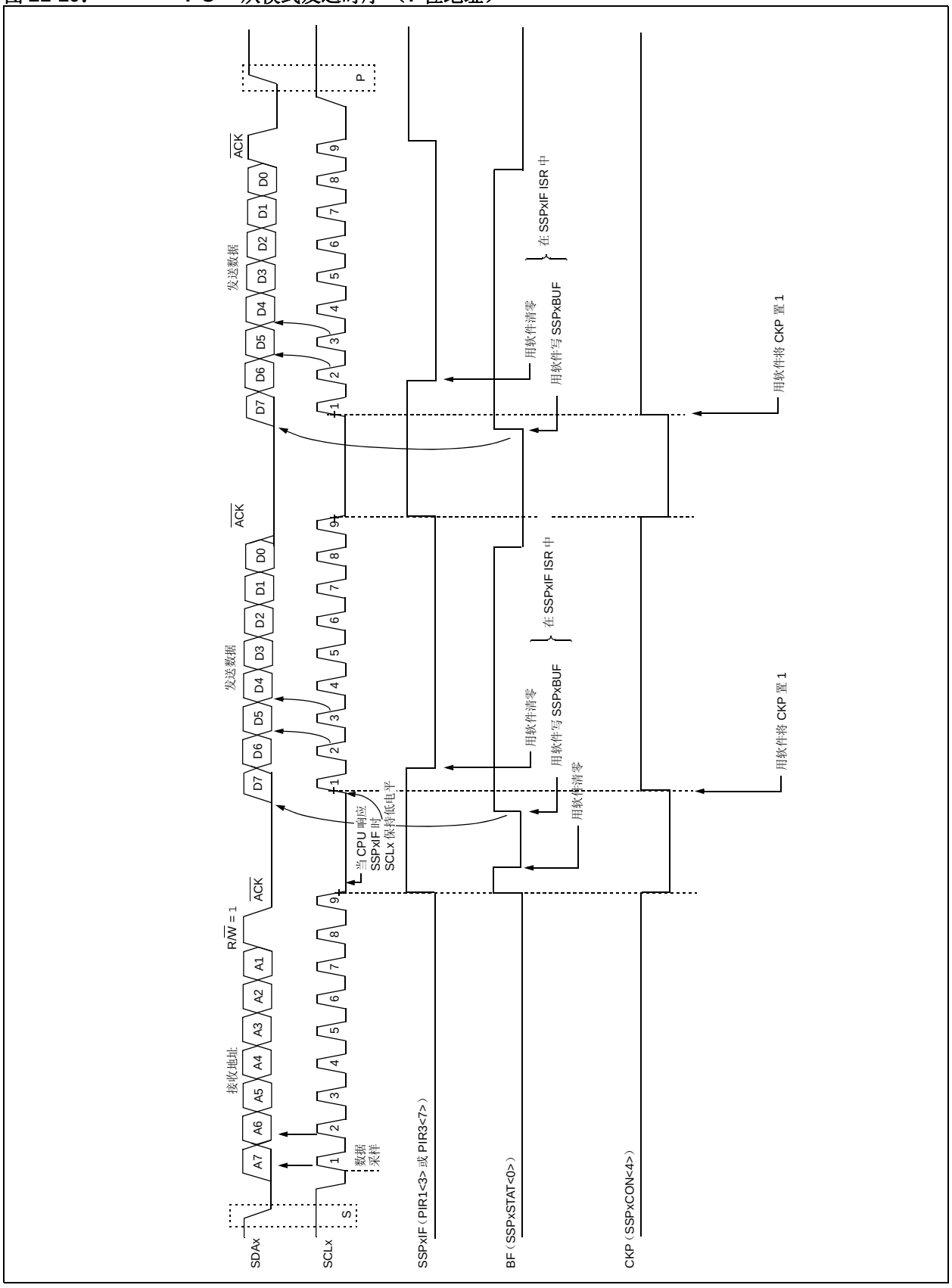
图 21-9： I²C™ 从模式接收时序 （SEN = 0 且 ADMSK<5:1> = 01011， 7 位地址）



注 1： x = 任意值 （即，地址位可以为 1 或 0）。

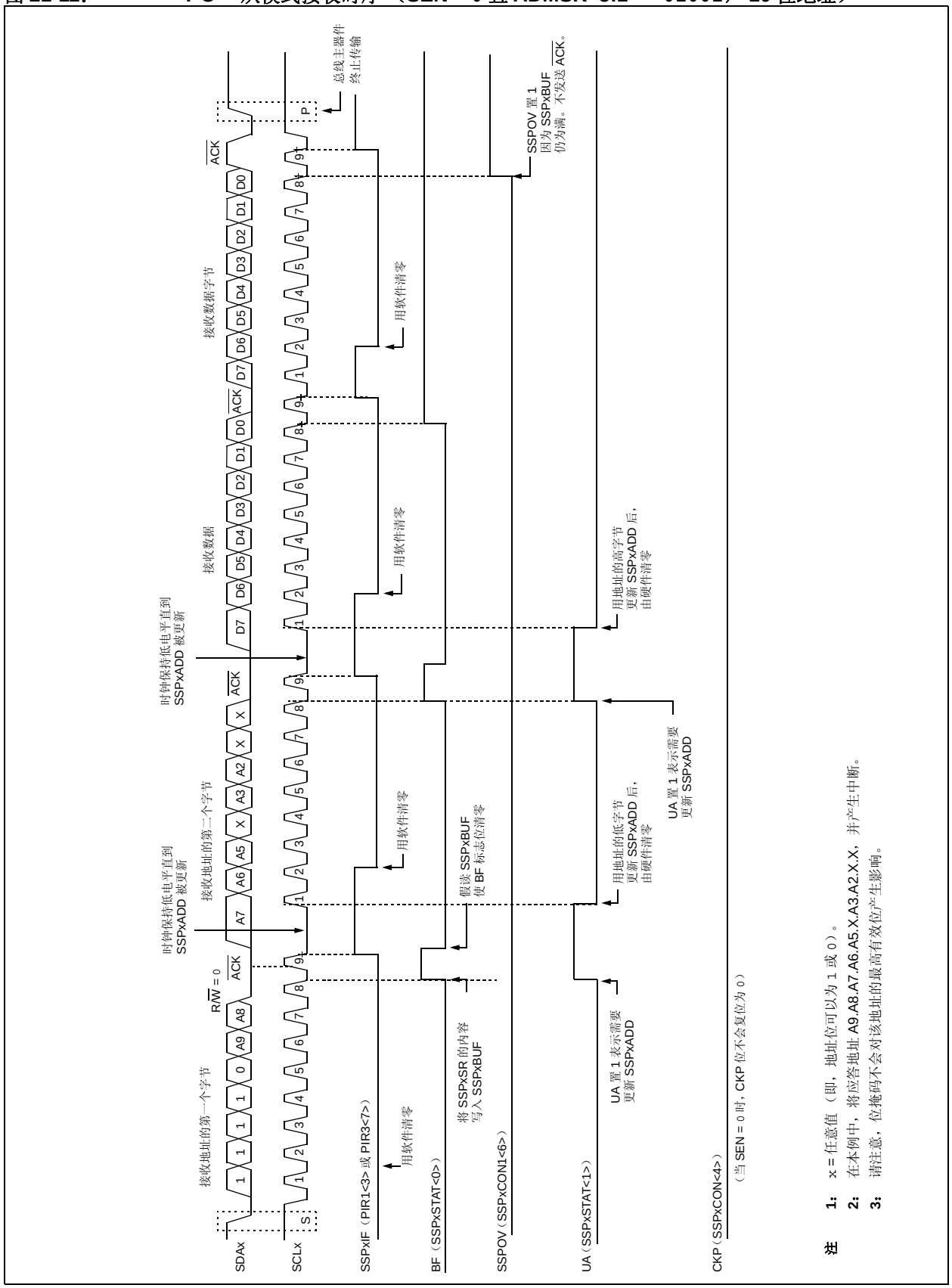
2： 在本例中，将应答地址 A7.A6.A5.X.A3.X.X，并产生中断。

图 21-10： I²C™ 从模式发送时序 （7 位地址）



© 2011 Microchip Technology Inc.

图 21-11： I²C™ 从模式接收时序 （SEN = 0 且 ADMSK<5:1> = 01001， 10 位地址）

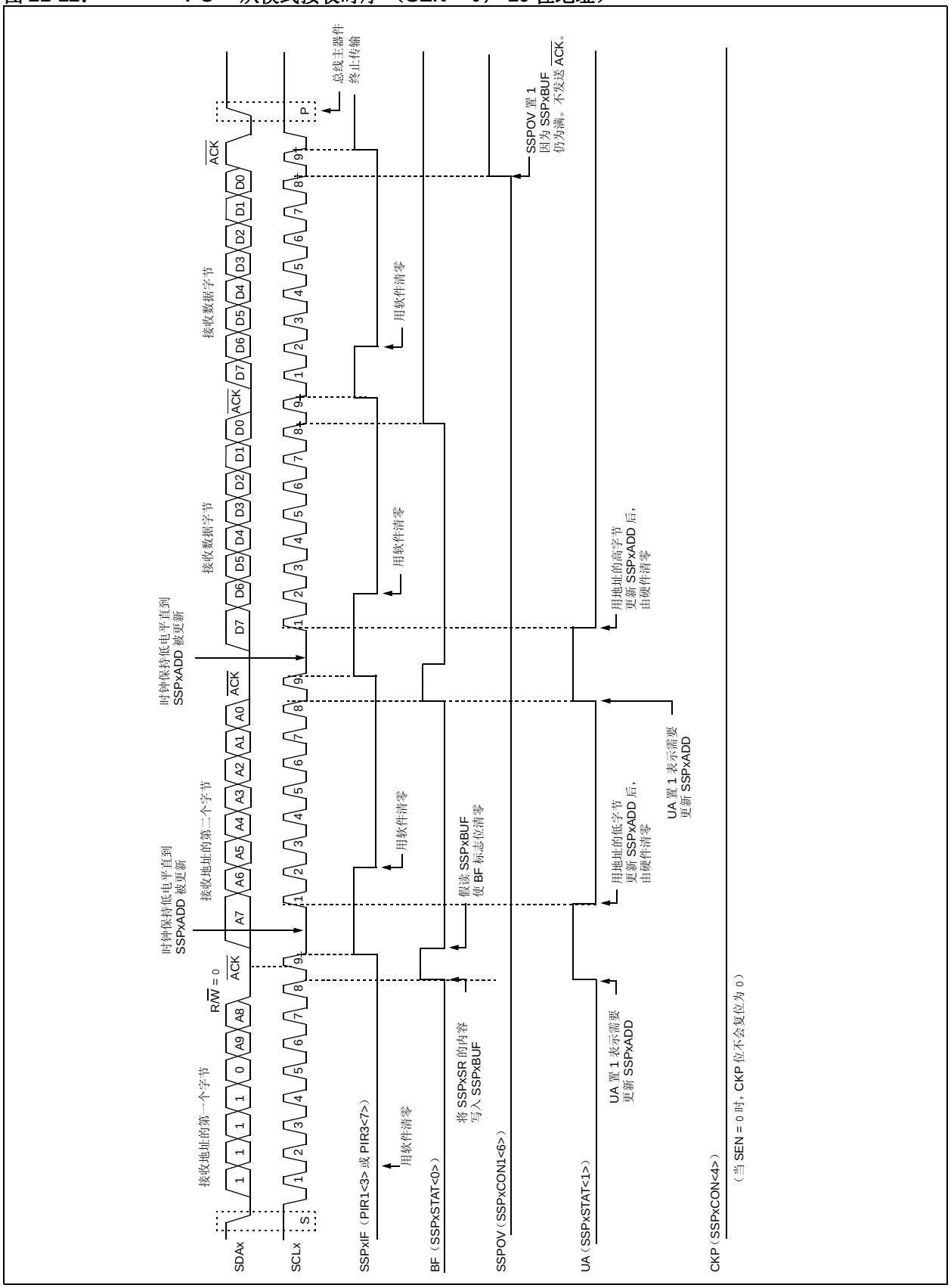


注 1： x = 任意值 （即，地址位可以为 1 或 0）。

2： 在本例中，将应答地址 A9.A8.A7.A6.A5.X.A3.A2.X.X，并产生中断。

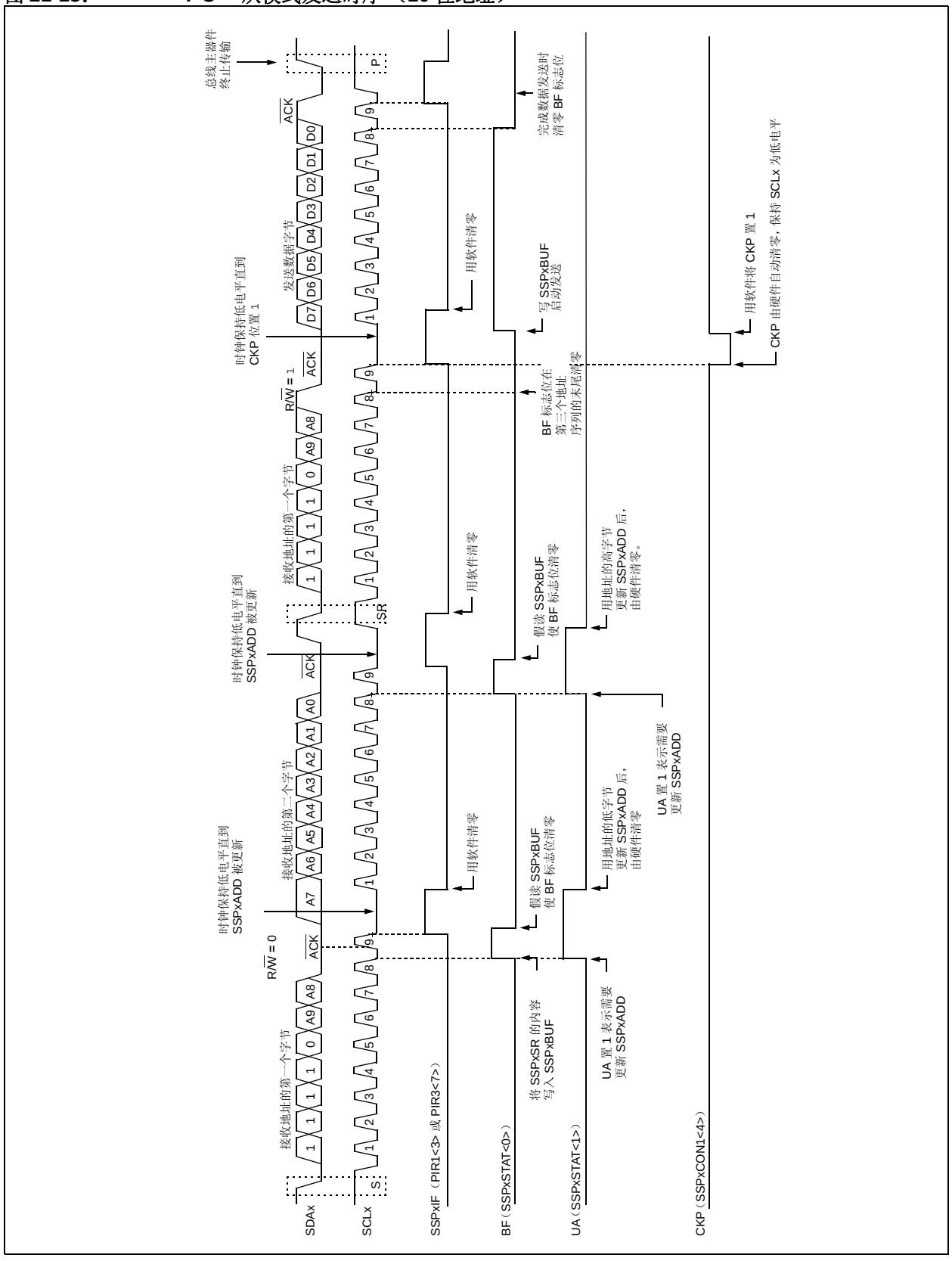
3： 请注意，位掩码不会对该地址的最高有效位产生影响。

图 21-12： I²C™ 从模式接收时序 （SEN = 0， 10 位地址）



© 2011 Microchip Technology Inc.

图 21-13： I²C™ 从模式发送时序 （10 位地址）



21.4.4 时钟延长

7 位和 10 位从模式都在发送序列中实现了自动时钟延长。

SEN 位 （SSPxCON2<0>）允许在接收过程中使能时钟延长。将 SEN 置 1 会导致在每个数据接收序列的末尾将 SCLx 引脚保持在低电平。

21.4.4.1 7 位从接收模式的时钟延长（SEN = 1）

在 7 位从接收模式下，如果在 $\overline{\text{ACK}}$ 序列末尾的第 9 个时钟的下降沿 BF 位置 1，则 SSPxCON1 寄存器中的 CKP 位就会自动清零，强制 SCLx 输出保持低电平。 CKP 被清为 0 会将 SCLx 线拉为低电平。在允许继续接收之前，必须在用户的中断服务程序中将 CKP 位置 1。保持 SCLx 线为低电平，让用户在主器件可以发起另一个接收序列之前，有时间处理中断服务程序并读取 SSPxBUF 的内容。这将防止发生缓冲器溢出 （见图 21-15）。

> **注 1：** 如果用户在第 9 个时钟的下降沿前读取了 SSPxBUF 的内容而导致 BF 位被清零，则 CKP 位将不会被清零，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。在下一个接收序列开始之前，用户要注意在中断服务程序中清零BF位，以避免溢出。

21.4.4.2 10 位从接收模式的时钟延长（SEN = 1）

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，就会启动时钟延长。在接收到 10 位地址的高字节后，UA 位被置 1，然后接收 10 位地址的第二个字节且 R/$\overline{\text{W}}$ 位清为 0。在更新 SSPxADD 的时候释放时钟线。如同 7 位模式那样，在每个数据接收序列中都会发生时钟延长。

> **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 SSPxADD 寄存器清零 UA 位，而且在此之前用户没有读取 SSPxBUF 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传输的数据序列中出现，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 位被请零， 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。

用户的中断服务程序必须先将 CKP 位置 1 才可以继续发送。保持 SCLx 线为低电平，让用户在主器件发起另一个发送序列之前，有时间处理中断服务程序并装入 SSPxBUF 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SSPxBUF 的内容，使 BF 位置 1，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。前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 1 的 R/$\overline{\text{W}}$ 位。在执行完第三个地址序列后， UA 位不置 1，此时模块被配置为发送模式，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）。

© 2011 Microchip Technology Inc.

21.4.4.5 时钟同步和 CKP 位

当 CKP 位被清零时，SCLx 输出被强制为 0。然而，清零 CKP 位不会将 SCLx 输出拉为低电平，除非已经采样到 SCLx 输出为低电平。因此， CKP 位将不会把 SCLx 线拉为低电平，除非外部 I^2C 主器件已将 SCLx 线拉低。 SCLx 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 SCLx 电平拉高。这可以确保对 CKP 位的写操作不会违反 SCLx 的最小高电平时间要求 （见图 21-14）。

图 21-14： 时钟同步时序

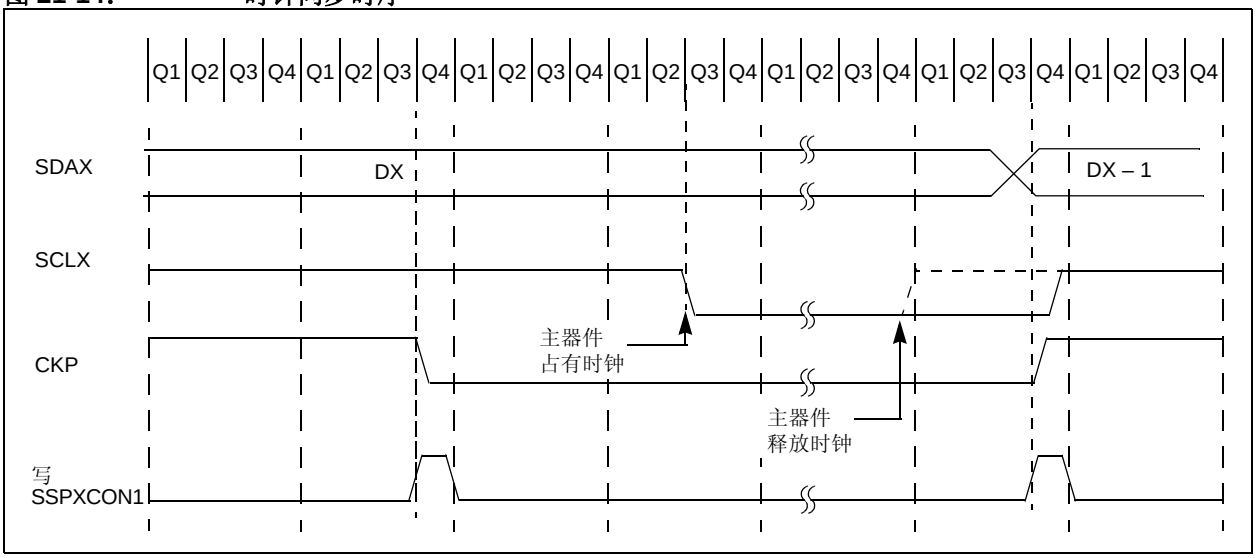
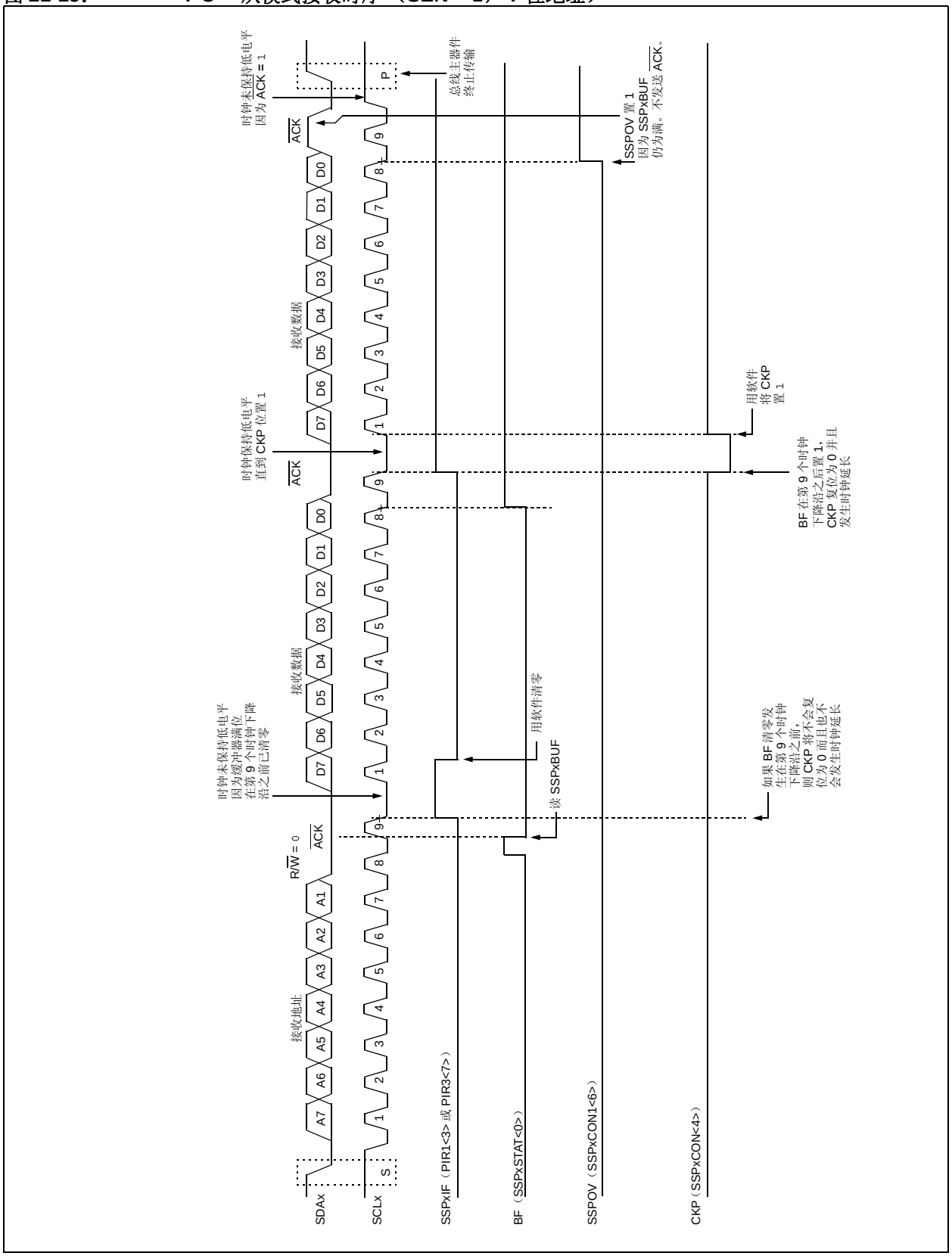
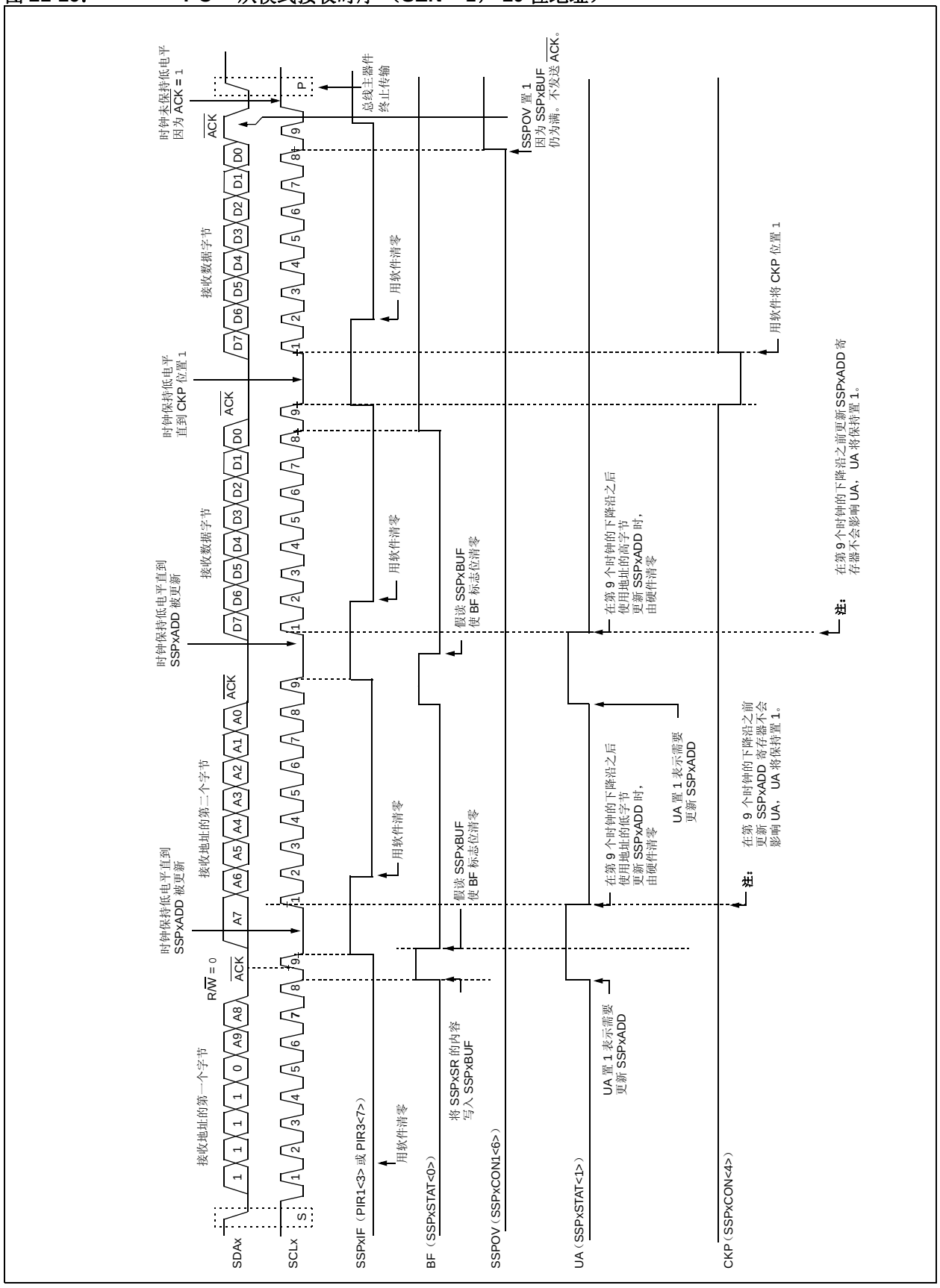


图 21-15： I²C™ 从模式接收时序 （SEN = 1， 7 位地址）



© 2011 Microchip Technology Inc.

图 21-16： I²C™ 从模式接收时序 （SEN = 1， 10 位地址）



21.4.5 支持广播呼叫地址

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。

广播呼叫地址是由 I²C 协议为特定目的保留的八个地址之一。它由全 0 组成，且 R/$\overline{W}$ = 0。

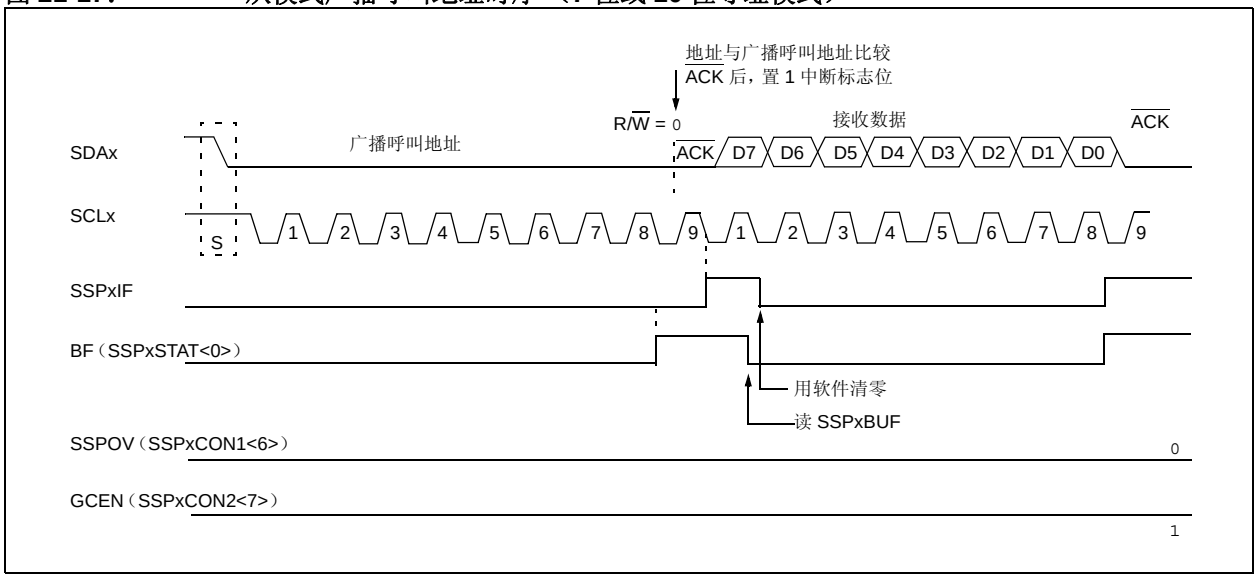
当使能广播呼叫使能位 GCEN（SSPxCON2<7> 置 1）时，即可识别广播呼叫地址。检测到启动位后，8 位数据会被移入 SSPxSR，同时将该地址与 SSPxADD 进行比较。它还会与广播呼叫地址进行比较并用硬件设定。

如果与广播呼叫地址匹配，SSPxSR 的值将被传输到 SSPxBUF，BF 标志位（第 8 位）置 1，并且在第 9 位（$\overline{ACK}$ 位）的下降沿 SSPxIF 中断标志位置 1。

当响应中断时，可以通过读取 SSPxBUF 的内容来检查中断源。该值可以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 SSPxADD 以匹配地址的后半部分，同时 UA 位（SSPxSTAT<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时序（7 位或 10 位寻址模式）



© 2011 Microchip Technology Inc.

21.4.6 主模式

通过将 SSPxCON1 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 位置 1，则 SCLx 和 SDAx 线由 MSSP 硬件控制。

主模式通过在检测到启动和停止条件时产生中断来工作。停止（P）位和启动（S）位在复位时或禁止 MSSP 模块时清零。当 P 位置 1 时，可以取得 I^2C 总线的控制权；否则 P 位和 S 位都清零，总线处于空闲状态。

在固件控制的主模式下，用户代码根据启动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：

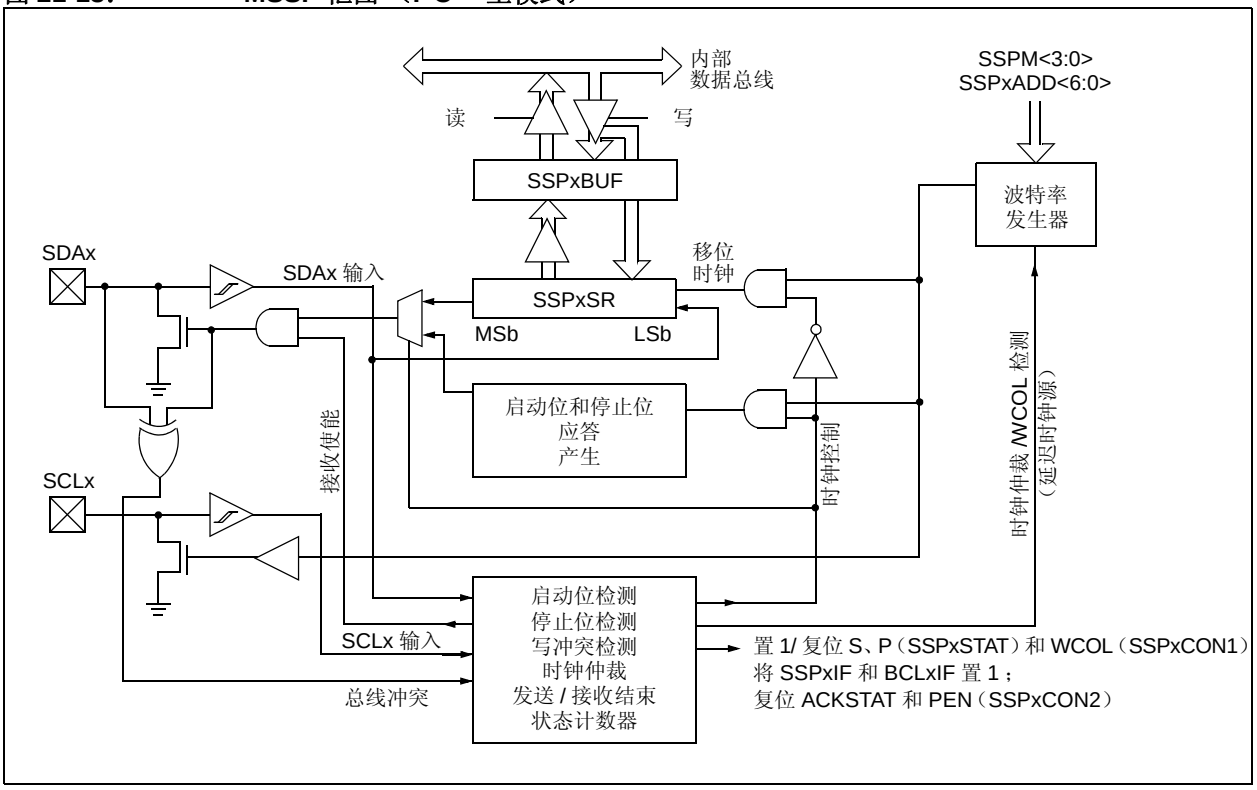
1. 在 SDAx 和 SCLx 上发出启动条件。
2. 在 SDAx 和 SCLx 上发出重复启动条件。
3. 写入 SSPxBUF 寄存器，启动数据 / 地址的发送。
4. 配置 I^2C 端口用于接收数据。
5. 在接收到数据字节后产生应答条件。
6. 在 SDAx 和 SCLx 上产生停止条件。

> **注：** 当被配置为 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 SSPxBUF 寄存器以启动传输。在这种情况下，将不会写 SSPxBUF，WCOL 位将被置 1，这表明没有发生对 SSPxBUF 的写操作。

下列事件会使 MSSP 中断标志位 SSPxIF 置 1（如果允许 MSSP 中断，则产生该中断）：

- 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

主器件产生所有串行时钟脉冲和启动 / 停止条件。以停止条件或重复启动条件结束传输。因为重复启动条件也是下一次串行传输的开始，因此不会释放 I²C 总线。

在主发送器模式下，串行数据通过 SDAx 输出，而串行时钟由 SCLx 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 位将为逻辑 0。一次发送 8 位串行数据。每发送一个字节，都会收到一个应答位。输出启动和停止条件来指示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 位将为逻辑 1。因此，发送的第一个字节就是一个 7 位从器件地址，后面跟 1 表示接收位。串行数据通过 SDAx 接收，而串行时钟由 SCLx 输出。每次接收 8 位串行数据。每接收到一个字节，都会发送一个应答位。启动和停止条件分别代表发送的开始和结束。

在 I²C 模式下，在 SPI 模式中使用的波特率发生器用于将 SCLx 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（SSPxCON2<0>）置 1，产生启动条件。
2. SSPxIF 置 1。在进行下一步操作前， MSSP 模块将等待所需的启动时间。
3. 用户将从器件地址装入 SSPxBUF 进行发送。
4. 地址从 SDAx 引脚移出，直到发送完所有 8 位地址数据。
5. MSSP 模块移入从器件的 ACK 位，并将它的值写入 SSPxCON2 寄存器 （SSPxCON2<6>）。
6. MSSP 模块在第 9 个时钟周期的末尾通过将 SSPxIF 位置 1，产生一个中断。
7. 用户将 8 位数据装入 SSPxBUF。
8. 数据从 SDAx 引脚移出，直到发送完所有 8 位数据。
9. MSSP 模块移入从器件的 ACK 位，并将它的值写入 SSPxCON2 寄存器 （SSPxCON2<6>）。
10. MSSP 模块在第 9 个时钟周期的末尾通过将 SSPxIF 位置 1，产生一个中断。
11. 用户通过将停止使能位 PEN（SSPxCON2<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。

© 2011 Microchip Technology Inc.

21.4.7 波特率

在 I²C 主模式下，波特率发生器（BRG）的重载值位于 SSPxADD 寄存器的低 7 位（图 21-19）。当发生对 SSPxBUF的写操作时，波特率发生器将自动开始计数。BRG 会递减计数到 0，然后停止，直到再次发生重载。BRG 的值会在每个指令周期（TCY）中的 Q2 和 Q4 时钟周期上进行两次减计数。在 I²C 主模式下，会自动重载 BRG。

如果指定操作完成（即，在传输最后一个数据位后传输 ACK），内部时钟将自动停止计数，SCLx 引脚将保持在其最后的状态。

表 21-3 显示了不同的指令周期下的时钟频率以及装入 SSPxADD 的 BRG 值。不支持 SSPxADD BRG 值为 0x00。

21.4.7.1 波特率和模块的相互关系

由于MSSP1和MSSP2是相互独立的，因此它们可以不同的波特率同时在 I²C 主模式下工作。这可以通过对每个模块使用不同的 BRG 重载值来实现。

由于此模式的基本时钟源是从系统时钟获取的，所以对时钟进行的任何更改都会对这两个模块产生相同程度上的影响。通过更改BRG重载值可以将其中一个或两个波特率都改回先前的值。

图 21-19： 波特率发生器框图

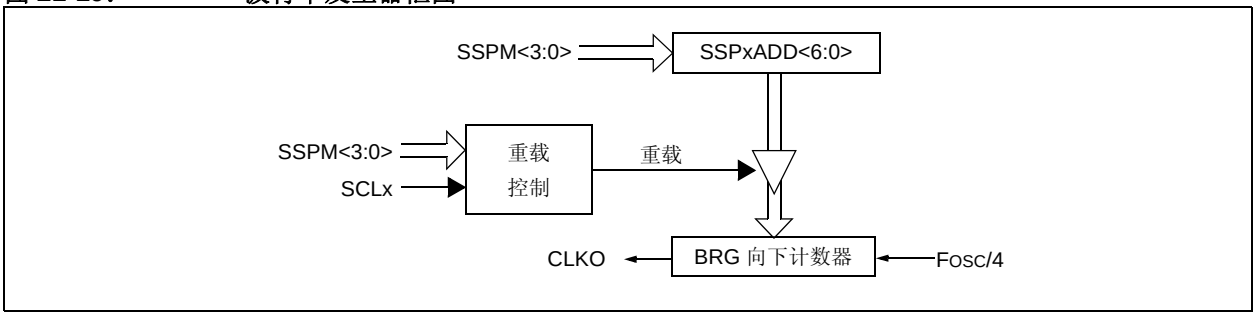


表 21-3： I²C™ 时钟频率与 BRG 的关系

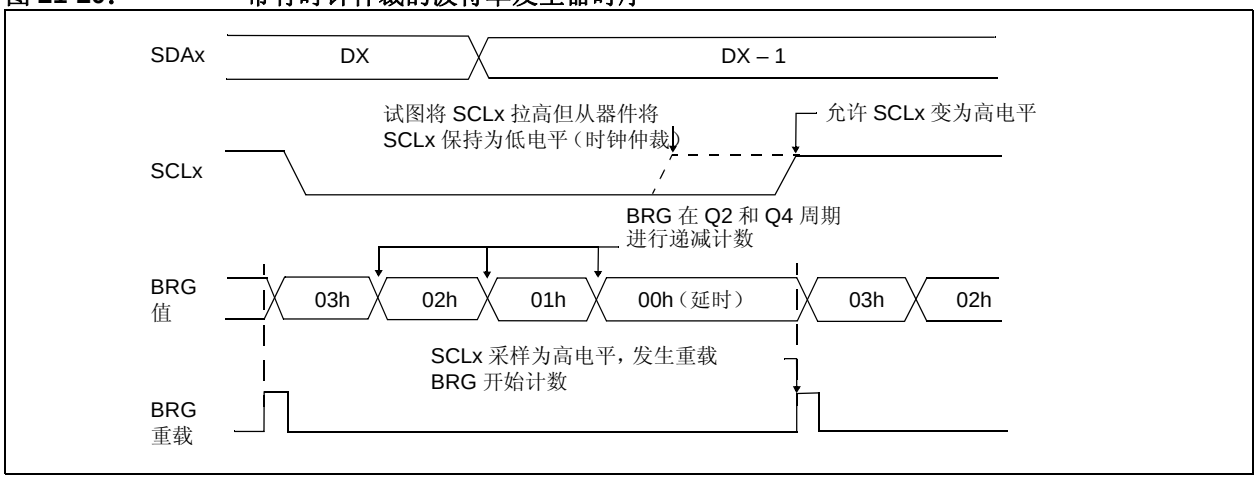
FOSC	FCY	FCY * 2	BRG 值	FSCL（两次 BRG 计满返回）
40 MHz	10 MHz	20 MHz	18h	400 kHz
40 MHz	10 MHz	20 MHz	1Fh	312.5 kHz
40 MHz	10 MHz	20 MHz	63h	100 kHz
16 MHz	4 MHz	8 MHz	09h	400 kHz
16 MHz	4 MHz	8 MHz	0Ch	308 kHz
16 MHz	4 MHz	8 MHz	27h	100 kHz
4 MHz	1 MHz	2 MHz	02h	333 kHz
4 MHz	1 MHz	2 MHz	09h	100 kHz
16 MHz	4 MHz	8 MHz	03h	1 MHz(1)

注 1： I²C 模式下，最少需要 16 MHz 的 FOSC 才能获得 1 MHz 的时钟频率。

21.4.7.2 时钟仲裁

如果在任何接收、发送或重复启动 / 停止条件过程中，主器件拉高了 SCLx 引脚（允许 SCLx 引脚悬空为高电平），就会发生时钟仲裁。如果允许 SCLx 引脚悬空为高电平，波特率发生器（BRG）将暂停计数直到实际采样到 SCLx 引脚为高电平为止。当 SCLx 引脚采样为高电平时，波特率发生器将重新装入 SSPxADD<6:0> 的值并开始计数。这可以保证当外部器件将时钟拉低时，SCLx 始终在至少一个 BRG 计满返回周期内保持高电平（图 21-20）。

图 21-20：带有时钟仲裁的波特率发生器时序



© 2011 Microchip Technology Inc.

21.4.8 I²C™ 主模式启动条件时序

要产生启动条件，用户应将启动使能位 SEN（SSPxCON2<0>）置 1。当 SDAx 和 SCLx 引脚采样为高电平时，波特率发生器重新装入 SSPxADD<6:0> 的值并开始计数。当波特率发生器发生超时（TBRG）时，如果 SCLx 和 SDAx 都被采样为高电平，则 SDAx 引脚被驱动为低电平。当 SCLx 为高电平时将 SDAx 驱动为低电平将产生启动条件，并使 S 位（SSPxSTAT<3>）置 1。随后波特率发生器重新装入SSPxADD<6:0>的值并恢复计数。当波特率发生器发生超时（TBRG）时，SEN 位（SSPxCON2<0>）将由硬件自动清零。波特率发生器将暂停工作， SDAx 线保持低电平，启动条件结束。

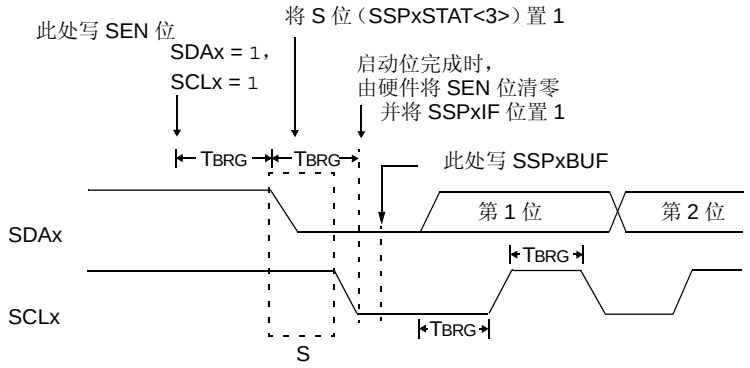
> **注：** 如果在启动条件开始时，SDAx 和 SCLx 引脚已采样为低电平，或者在启动条件期间， SCLx 线在 SDA 线被驱动为低电平之前已经采样为低电平，则会产生总线冲突，总线冲突中断标志位 BCLxIF 置 1，启动条件中止， I²C 模块复位为空闲状态。

21.4.8.1 WCOL 状态标志

在启动时序进行当中，如果用户写SSPxBUF，则WCOL 位被置 1，同时缓冲器内容不变 （写操作无效）。

> **注：** 由于不允许事件排队，在启动条件结束之前，不能对 SSPxCON2 的低 5 位进行写操作。

图 21-21： 第一个启动位的时序



21.4.9 I²C™ 主模式重复启动条件时序

将 RSEN 位 （SSPxCON2<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，SCLx 引脚被拉为低电平。当 SCLx 引脚采样为低电平时，波特率发生器将重新装入 SSPxADD<5:0> 的值，并开始计数。在一个波特率发生器计数周期 （TBRG）内 SDAx 引脚被释放 （引脚电平被拉高）。当波特率发生器超时时，如果 SDAx 被采样为高电平， SCLx 引脚将被释放 （引脚电平被拉高）。当 SCLx 被采样为高电平时，波特率发生器重新装入 SSPxADD<6:0> 的值并开始计数。SDAx 和 SCLx 必须在一个 TBRG 内被采样为高电平。接下来，在一个 TBRG 中将 SDAx 引脚拉为低电平 （SDAx = 0），同时 SCLx 保持高电平。然后 RSEN 位 （SSPxCON2<1>）将自动清零，波特率发生器不会被重载并且 SDAx 引脚保持低电平。一旦在 SDAx 和 SCLx 引脚上检测到启动条件，就会置 1 S 位 （SSPxSTAT<3>）。直到波特率发生器发生超时后， SSPxIF 位才会置 1。

> **注 1：** 有任何其他事件在进行时，编程设置对 RSEN 无效。
>
> **2：** 在重复启动条件期间，下列事件将会导致总线冲突：
>
> - 当 SCLx 由低电平变为高电平时，采样到 SDAx 为低电平。
> - 在SDAx被拉低之前，SCLx变为低电平。这表示另一个主器件正尝试发送数据 1。

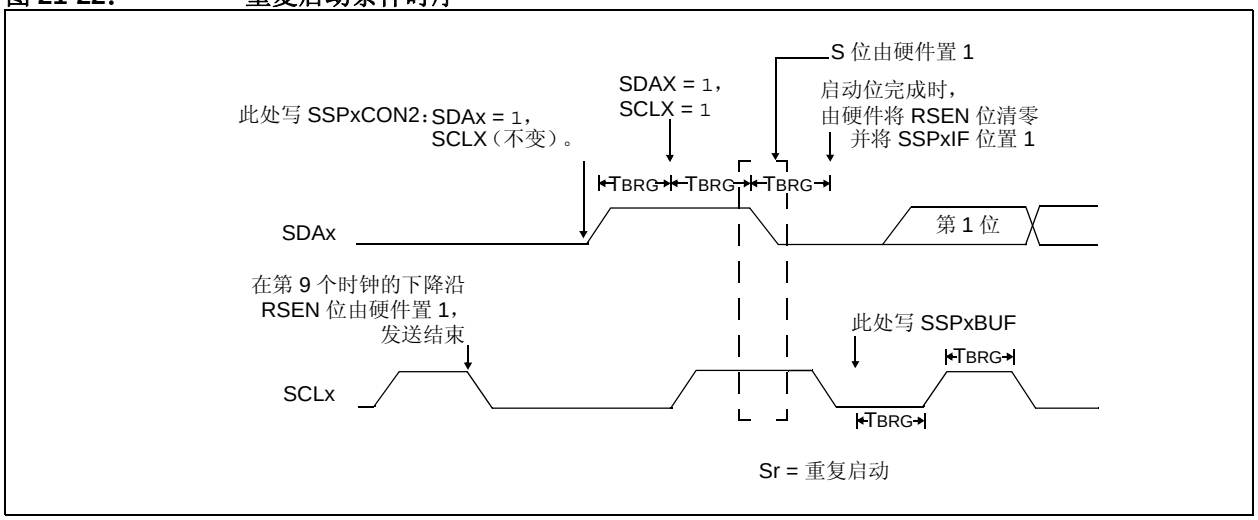
一旦 SSPxIF 位被置 1，用户便可以在 7 位寻址模式下将 7 位地址写入 SSPxBUF，或者在 10 位寻址模式下将默认的第一个地址字节写入 SSPxBUF。当发送完第一个 8 位数据并接收到一个 ACK 后，用户可以发送另外 8 位地址 （10 位寻址模式）或 8 位数据 （7 位寻址模式）。

21.4.9.1 WCOL 状态标志

在重复启动时序进行当中，如果用户写 SSPxBUF，则 WCOL 被置 1，同时缓冲器内容不变 （写操作无效）。

> **注：** 由于不允许事件排队，在重复启动条件完成之前，不能对 SSPxCON2 的低 5 位进行写操作。

图 21-22： 重复启动条件时序



© 2011 Microchip Technology Inc.

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，都可以通过写一个值到 SSPxBUF 寄存器来实现。该操作将使缓冲器满标志位 BF 置 1，波特率发生器开始计数，同时启动下一次发送。在出现 SCLx 的下降沿后（见数据保持时间规范参数 106），地址 / 数据的每一位被移出至 SDAx 引脚。在一个波特率发生器计满返回周期（TBRG）内，SCLx 保持低电平。数据应该在 SCLx 释放为高电平前保持有效（见数据建立时间规范参数107）。当SCLx引脚被释放为高电平时，它将在一个 TBRG 内保持为高电平。在此期间以及 SCLx 的下一个下降沿之后的一段保持时间内，SDAx 引脚上的数据必须保持稳定。在第 8 位数据被移出（第 8 个时钟周期的下降沿）之后，BF 标志位被清零，同时主器件释放 SDAx。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位时以一个ACK 位做出响应。ACK 的状态在第 9 个时钟的下降沿写入 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 个时钟周期之后，SSPxIF 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 SSPxBUF，同时 SCLx 引脚保持低电平，并且 SDAx 保持不变（图 21-23）。

在写 SSPxBUF 之后，地址的每一位在 SCLx 的下降沿被移出，直至所有 7 位地址和 R/W 位都被移出。在第 8 个时钟的下降沿，主器件将释放 SDAx 引脚，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 SDAx 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（SSPxCON2<6>）。在发送地址的第 9 个时钟下降沿之后，SSPxIF 位置 1，BF 标志位清零，波特率发生器关闭直到下一次写SSPxBUF，同时SCLx引脚保持低电平并且允许 SDAx 引脚悬空。

21.4.10.1 BF 状态标志

在发送模式下，BF 位（SSPxSTAT<0>）在 CPU 写 SSPxBUF 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，SSPxSR 仍在移出数据字节时）写 SSPxBUF，则 WCOL 位置 1 并且在写入 SSPxBUF 之后的 2 个 TCY 内缓冲器内容不变（写操作无效）。如果在 2 个 TCY 内重写 SSPxBUF，则 WCOL 位置 1 并且更新 SSPxBUF。这可能导致传输被破坏。

在每次写 SSPxBUF 之后，用户应该验证 WCOL 位是否清零以确保传输是正确的。在所有情况下，WCOL 必须用软件清零。

21.4.10.3 ACKSTAT 状态标志

在发送模式下，当从器件发送应答（$\overline{ACK}$ = 0）时，ACKSTAT 位（SSPxCON2<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（SSPxCON2<3>）使能主模式接收。

> **注：** RCEN 位置 1 前，MSSP 模块必须处于空闲状态，否则对 RCEN 位的置 1 操作将无效。

波特率发生器开始计数，每次计满返回时，SCLx 引脚的状态发生改变（由高变低或由低变高），同时数据被移入 SSPxSR。第 8 个时钟的下降沿之后，接收使能标志位自动清零，SSPxSR 的内容装入 SSPxBUF，BF 标志位置 1，SSPxIF 标志位置 1 并且波特率发生器暂停计数，SCLx 保持为低电平。此时 MSSP 处于空闲状态，等待下一条命令。当 CPU 读缓冲器时，BF 标志位将自动清零。通过将应答序列使能位 ACKEN（SSPxCON2<4>）置 1，用户可以在接收结束后发送应答位。

21.4.11.1 BF 状态标志

接收过程中，当将地址或数据字节从 SSPxSR 装入 SSPxBUF 时，BF 位置 1。在读 SSPxBUF 寄存器时清零。

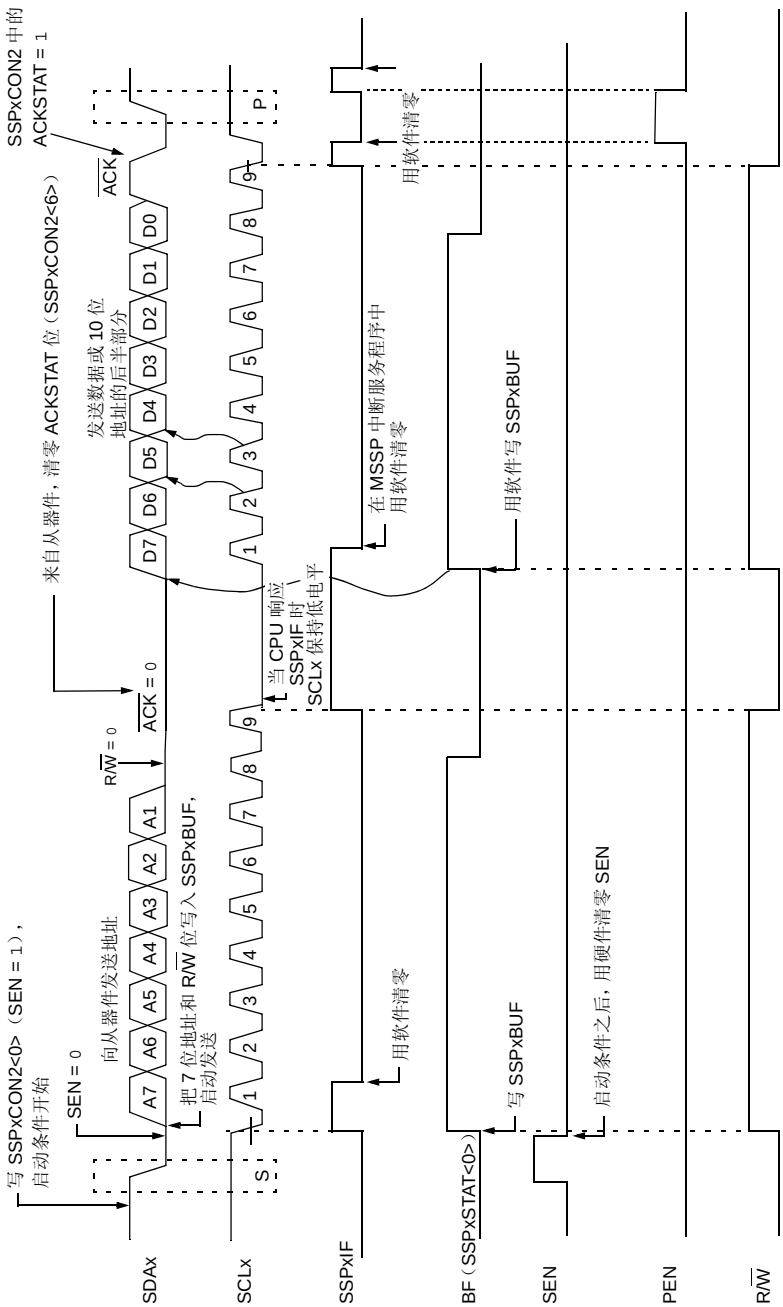
21.4.11.2 SSPOV 状态标志

接收过程中，当 SSPxSR 接收到 8 位数据且 BF 标志位已经在上一次接收时置 1 时，SSPOV 位置 1。

21.4.11.3 WCOL 状态标志

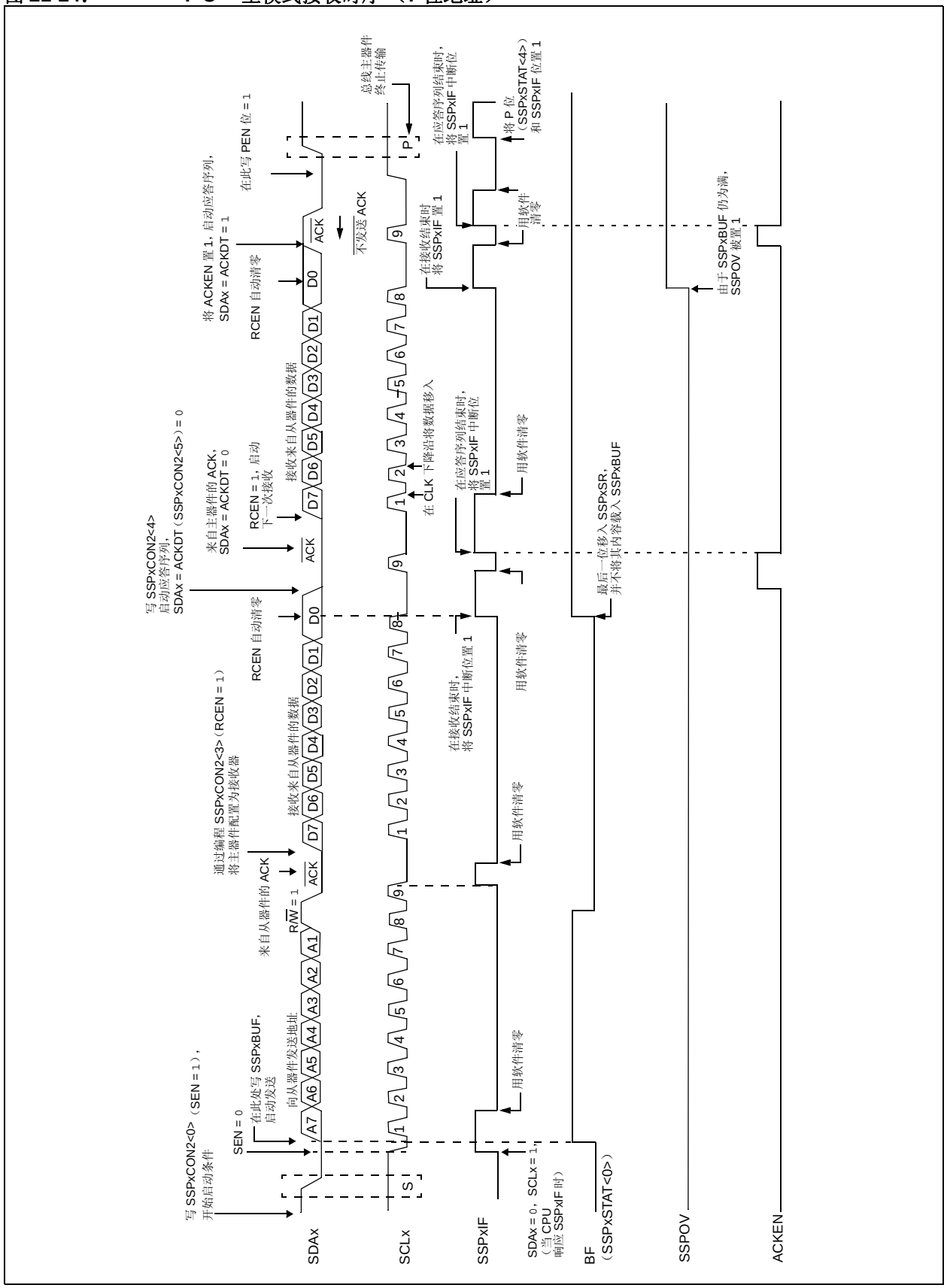
如果用户在接收过程中（即，SSPxSR 仍在移入数据字节时）写 SSPxBUF，则 WCOL 位置 1 并且缓冲器内容不变（写操作无效）。

图 21-23： I²C™ 主模式发送时序 （7 位或 10 位地址）



© 2011 Microchip Technology Inc.

图 21-24： I²C™ 主模式接收时序 （7 位地址）



21.4.12 应答序列时序

将应答序列使能位 ACKEN（SSPxCON2<4>）置 1 即可使能应答序列。当该位被置 1 时，SCLx 引脚被拉低，应答数据位的内容输出到 SDAx 引脚。如果用户希望产生一个应答，则应该将 ACKDT 位清零。否则，用户应在应答序列开始前将 ACKDT 位置 1。然后波特率发生器进行一个计满返回周期（TBRG）的计数，随后 SCLx 引脚电平被释放（拉为高电平）。当 SCLx 引脚被采样为高电平时（时钟仲裁），波特率发生器进行一个 TBRG 周期的计数，然后 SCLx 引脚被拉低。在这之后，ACKEN 位自动清零，波特率发生器关闭并且 MSSP 模块进入非工作模式（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 SSPxBUF，则 WCOL 将置 1 并且缓冲器的内容不会改变（写操作无效）。

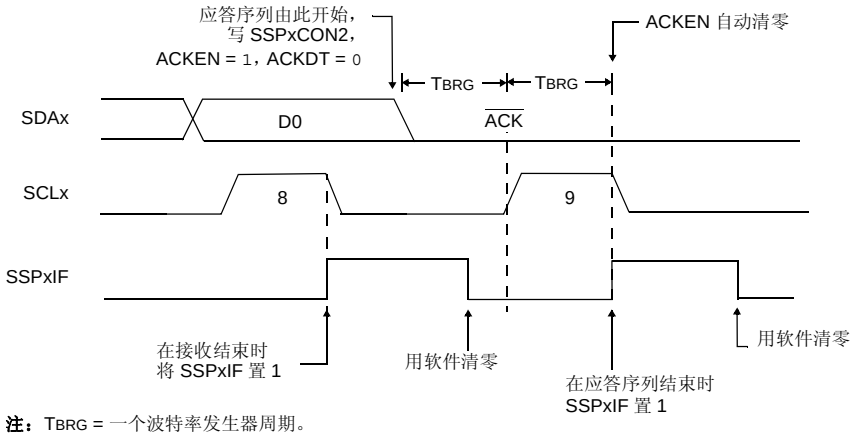
21.4.13 停止条件时序

将停止序列使能位 PEN（SSPxCON2<2>）置 1，在接收 / 发送结束后，SDAx 引脚上将产生停止位。在接收 / 发送结束时，SCLx 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 SDAx 线拉为低电平。当 SDAx 线被采样为低电平时，波特率发生器发生重载并递减计数到 0。当波特率发生器发生超时，SCLx 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，SDAx 引脚将被拉高。当 SDAx 引脚被采样为高电平且 SCLx 也是高电平时，P 位（SSPxSTAT<4>）置 1。另一个 TBRG 之后，PEN 位被清零，同时 SSPxIF 位被置 1（见图 21-26）。

21.4.13.1 WCOL 状态标志

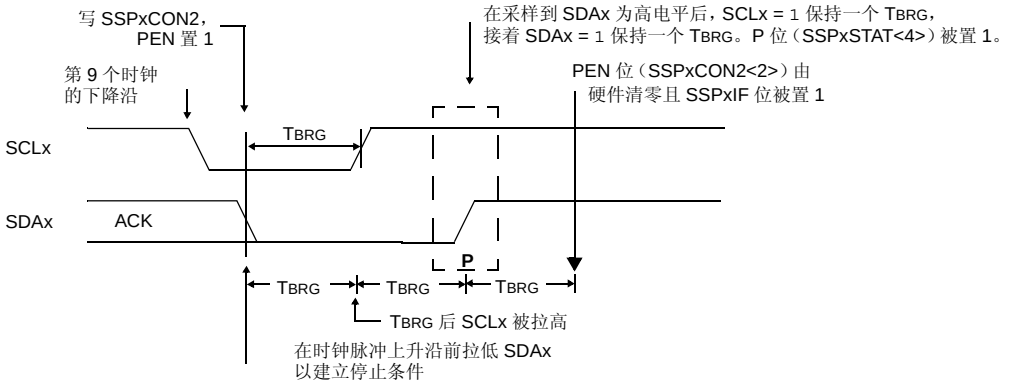
如果用户在停止序列进行过程中写 SSPxBUF，则 WCOL 位将被置 1，缓冲器内容不会改变（写操作无效）。

图 21-25： 应答序列时序



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。

© 2011 Microchip Technology Inc.

21.4.14 休眠状态下的操作

在休眠模式下，I²C 模块能够接收地址或数据，并且在地址匹配时或字节传输完成后，如果允许 MSSP 中断，会将处理器从休眠模式唤醒。

21.4.15 复位的影响

复位会禁止 MSSP 模块并终止当前的传输。

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，这可用于判断总线何时空闲。停止（P）位和启动（S）位在复位时或禁止 MSSP 模块时清零。当 P 位（SSPxSTAT<4>）置 1 时，可以取得 I²C 总线的控制权；否则 P 位和 S 位清零且总线空闲。当总线忙时，一旦出现停止条件，允许 MSSP 中断的情况下将产生中断。

在多主器件模式下，必须一直监视 SDAx 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 BCLxIF 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、总线冲突与总线仲裁

通过总线仲裁来支持多主器件模式。当主器件将地址 / 数据位输出到 SDAx 引脚时，如果主器件通过将 SDAx 悬空为高电平以在 SDAx 上输出 1，而另一个主器件输出 0，就会发生总线仲裁。如果 SDAx 引脚上期望的数据是 1，而实际采样到的数据是 0，则表示发生了总线冲突。主器件将把总线冲突中断标志位 BCLxIF 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，则会暂停发送，并且 BF 标志位被清零，SDAx 和 SCLx 线被拉高同时将 SSPxBUF置于可写入状态。当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。

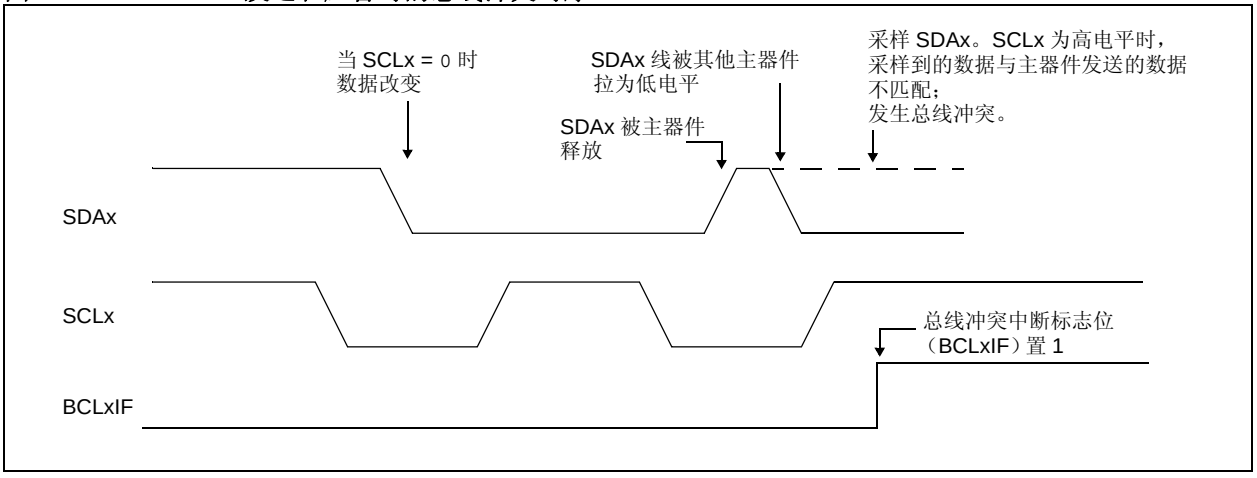
如果在启动、重复启动、停止或应答条件进行过程中发生总线冲突，那么相应的条件将被中止，SDAx 和 SCLx 线被拉高，SSPxCON2 寄存器中的对应控制位清零。当执行完总线冲突中断服务程序（ISR）后，如果 I²C 总线空闲，用户可通过发出启动条件恢复通信。

主器件将继续监视 SDAx 和 SCLx 引脚。如果出现停止条件，SSPxIF 位将被置 1。

无论发生总线冲突时发送的进度如何，写入 SSPxBUF 都会重新从第一个数据位开始发送数据。

在多主器件模式下，通过在检测到启动和停止条件时产生中断可以确定总线何时空闲。SSPxSTAT寄存器中的P 位置 1 时，可以取得 I²C 总线的控制权；否则 P 位和 S 位清零，总线空闲。

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， SDAx 或 SCLx 被采样为低电平 （图 21-28）。

b) 将 SDAx 拉低之前， SCLx 被采样为低电平（图 21-29）。

在启动条件期间，将监视 SDAx 和 SCLx 引脚。

如果 SDAx 引脚或 SCLx 引脚已经是低电平，则会发生下面的情况：

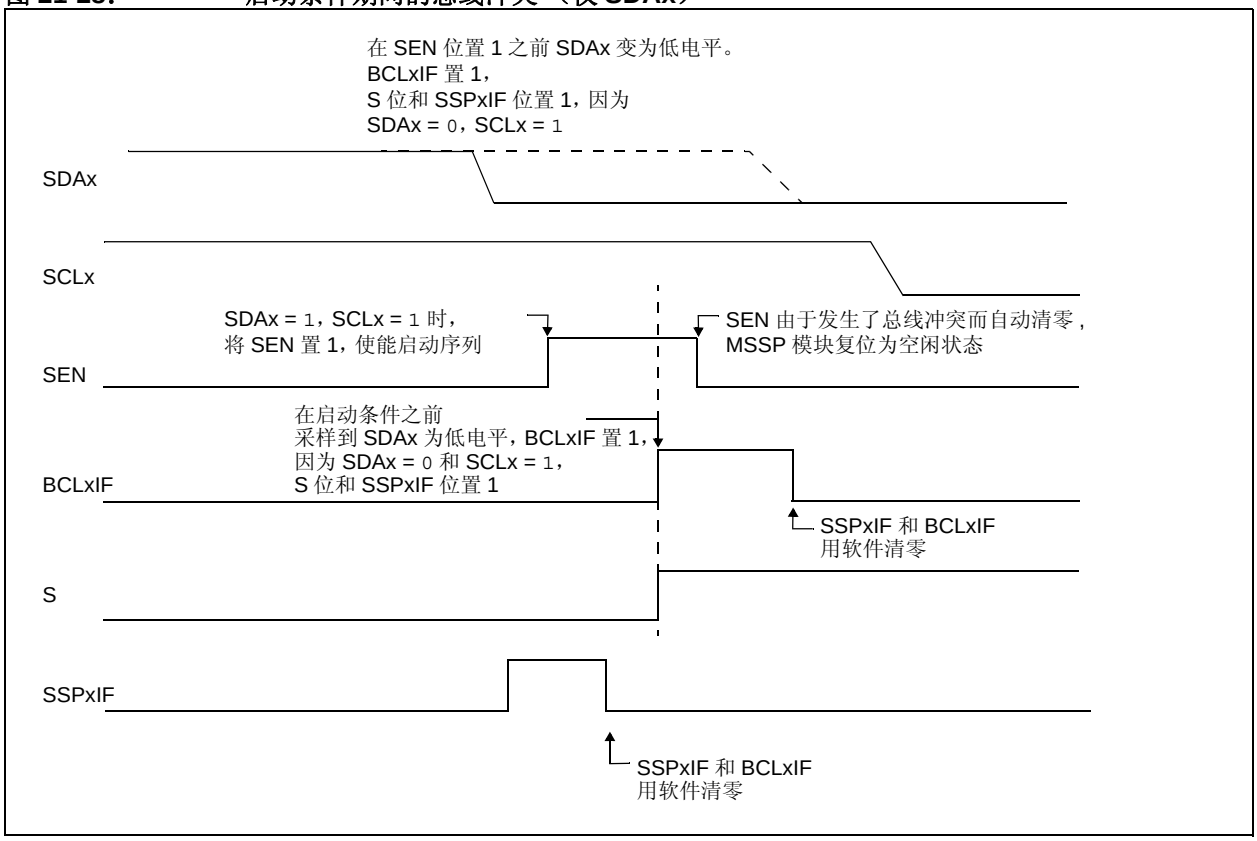
- 中止启动条件
- BCLxIF 标志位置 1
- MSSP 模块复位为空闲状态 （图 21-28）

拉高 SDAx 和 SCLx 引脚将建立启动条件。当 SDAx 引脚被采样为高电平时，波特率发生器装入 SSPxADD<6:0> 的值并递减计数到 0。如果在 SDAx 引脚为高电平时， SCLx 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发送一个数据 1。

如果 SDAx 引脚在该计数周期内采样为低电平，则 BRG 复位，同时 SDAx 线保持原值 （图 21-30）。但是，如果 SDAx 引脚被采样为 1，SDAx 引脚将在 BRG 计数结束时被拉为低电平。接着，波特率发生器被重载并递减计数到 0。在此期间，如果 SCLx 引脚采样到 0，则不会发生总线冲突。在 BRG 计数结束时，SCLx 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SDAx拉低。但是这一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、重复启动或停止条件进行仲裁。

图 21-28： 启动条件期间的总线冲突 （仅 SDAx）



© 2011 Microchip Technology Inc.

图 21-29： 启动条件期间的总线冲突 （SCLx = 0）

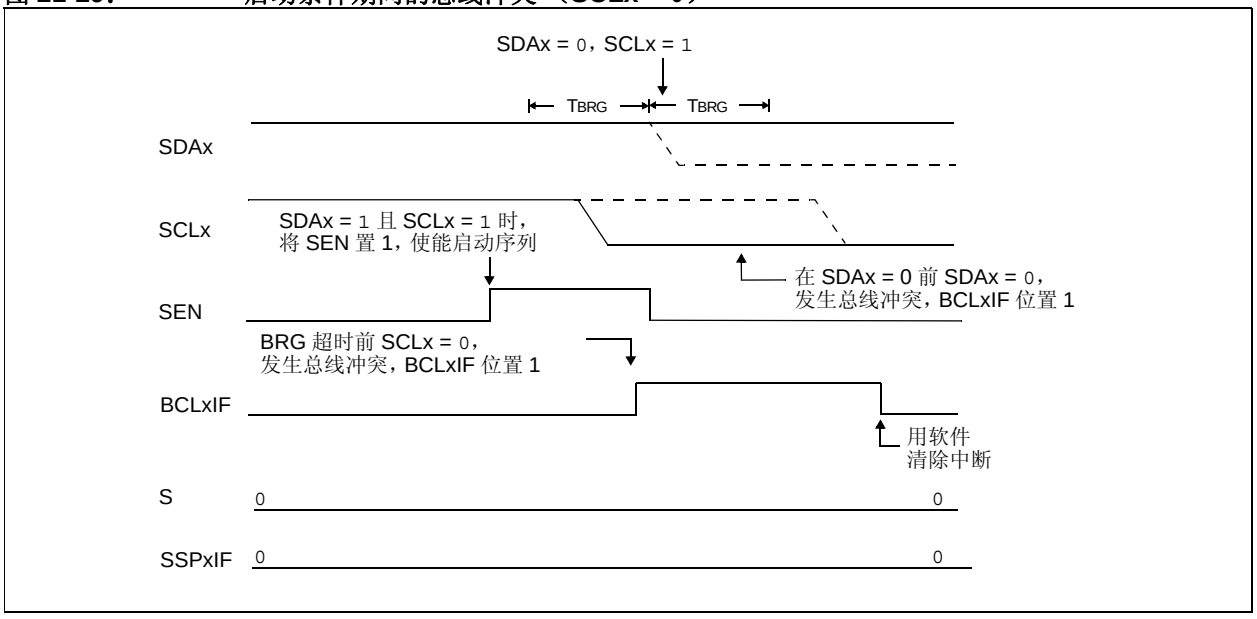
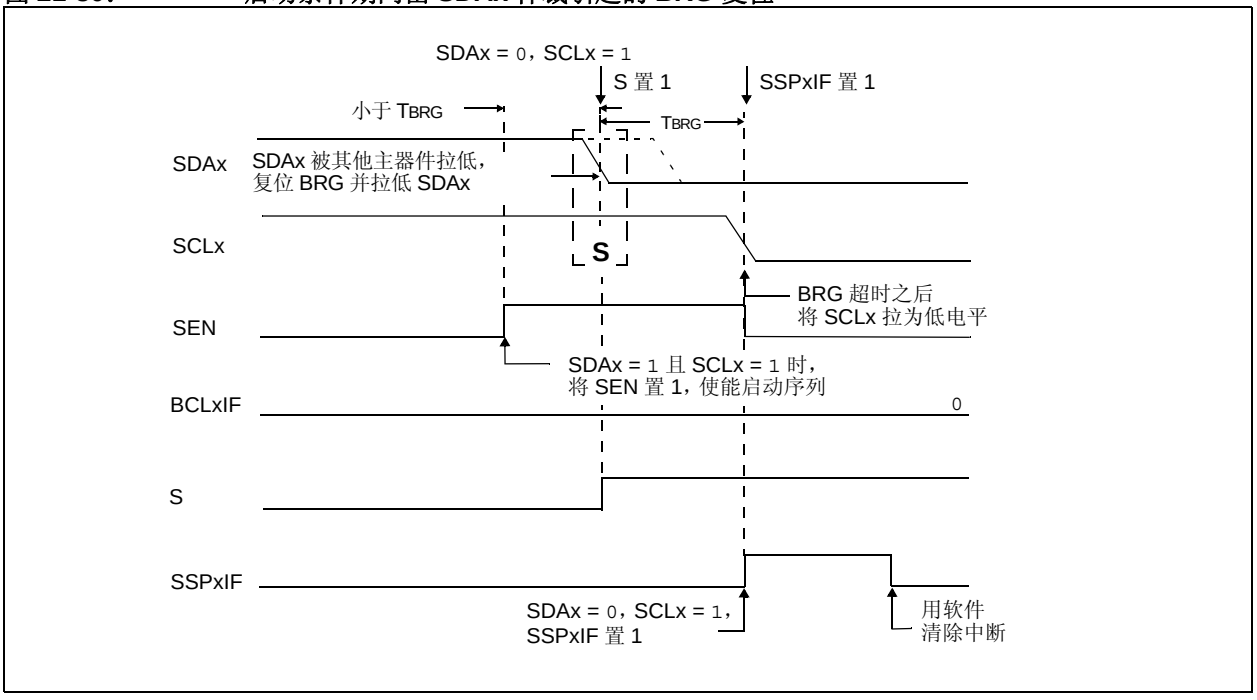


图 21-30： 启动条件期间由 SDAx 仲裁引起的 BRG 复位



21.4.17.2 重复启动条件期间的总线冲突

在重复启动条件期间，以下事件会导致发生总线冲突：

a) 在 SCLx 由低电平变为高电平期间，在 SDAx 上采样到低电平。

b) 在 SDAx 被拉为低电平之前， SCLx 变为低电平，表示另一个主器件正试图发送数据 1。

当用户释放 SDAx 且允许该引脚悬空为高电平时，BRG 装入SSPxADD<6:0>中的值并递减计数到0。接着SCLx 引脚被拉为高电平，当采样到 SCLx 引脚为高电平时，对 SDAx 引脚进行采样。

如果 SDAx 为低电平，则表示已发生了总线冲突（即，另一个主器件正试图发送数据 0，图 21-31）。如果SDAx 被采样为高电平，则BRG被重载并开始计数。如果SDAx 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 SDAx 拉低。

如果 SCLx 在 BRG 超时之前从高电平变为低电平，且 SDAx 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数据 1（见图 21-32）。

如果在 BRG 计时结束时，SCLx 和 SDAx 都仍然是高电平，则 SDAx 引脚被拉低， BRG 被重载并开始计数。在计数结束时，不管 SCLx 引脚的状态如何，SCLx 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）

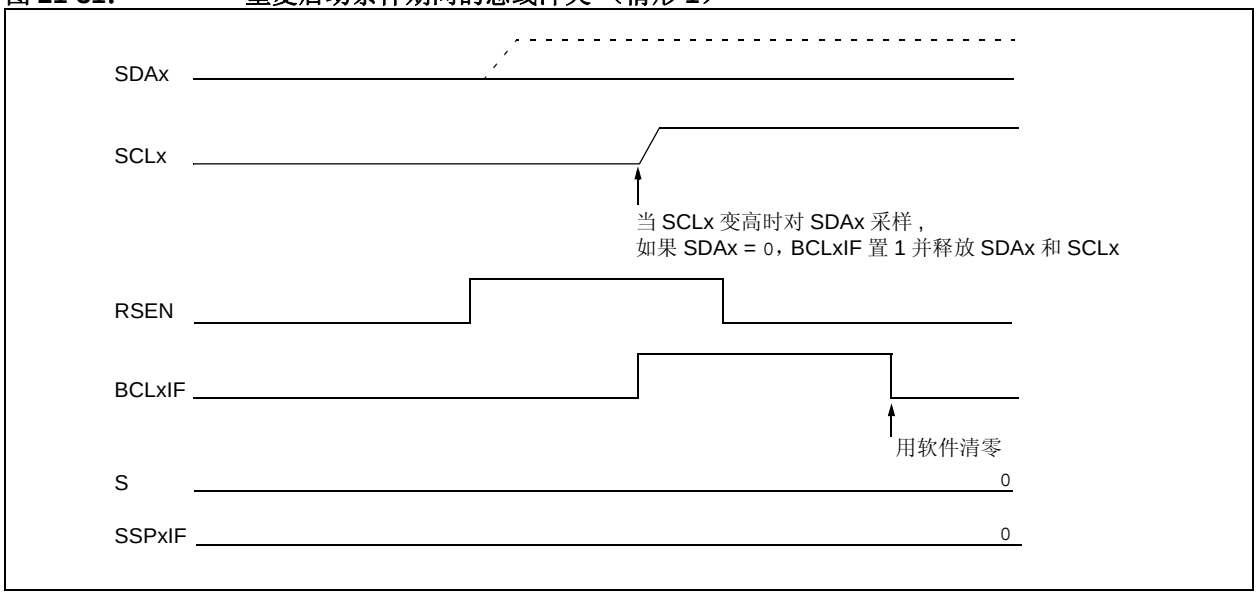
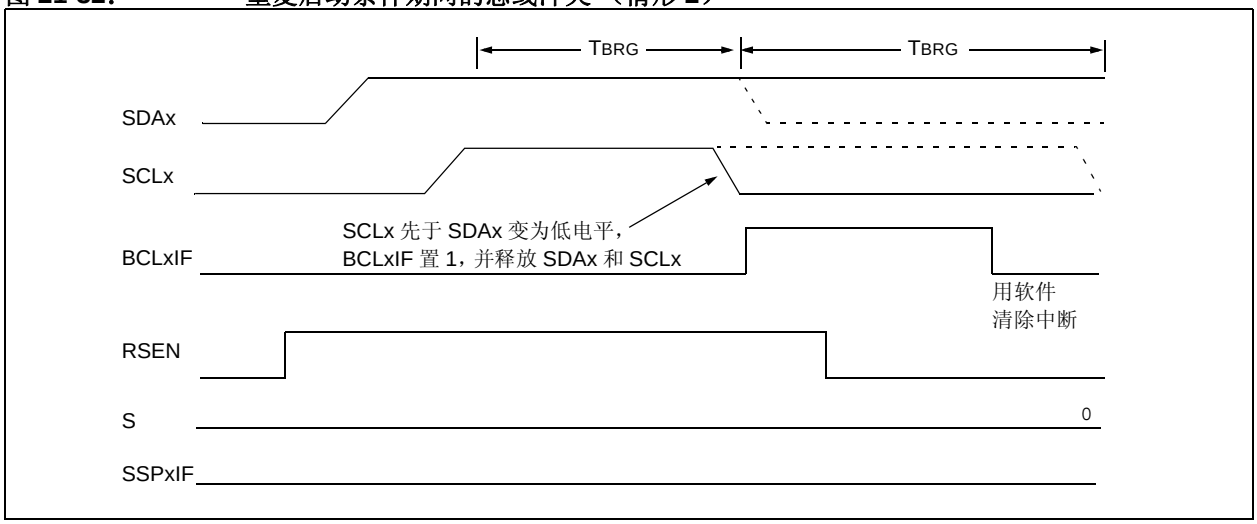


图 21-32： 重复启动条件期间的总线冲突 （情形 2）



© 2011 Microchip Technology Inc.

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) SDAx 引脚被释放并允许悬空为高电平之后，SDAx 在 BRG 超时后采样为低电平。

b) SCLx 引脚被拉高之后，SCLx 在 SDAx 变成高电平之前采样为低电平。

停止条件从 SDAx 被拉低开始。当 SDAx 被采样为低电平时， SCLx 引脚就可以悬空。当 SDAx 引脚被采样为高电平（时钟仲裁）时，波特率发生器装入 SSPxADD<6:0> 的值并递减计数到 0。 BRG 超时后，采样 SDAx。如果 SDAx 被采样为低电平，则表示已发生总线冲突。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 SCLx 引脚在允许 SDAx 悬空为高电平前被采样为低电平，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 的另一个情形（图 21-34）。

图 21-33： 停止条件期间的总线冲突 （情形 1）

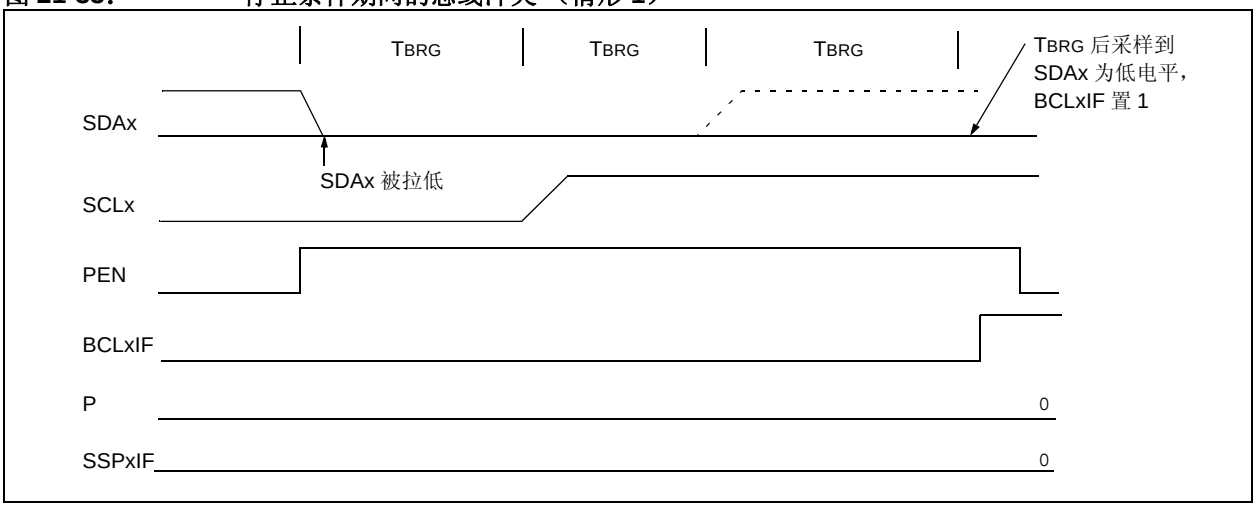


图 21-34： 停止条件期间的总线冲突 （情形 2）

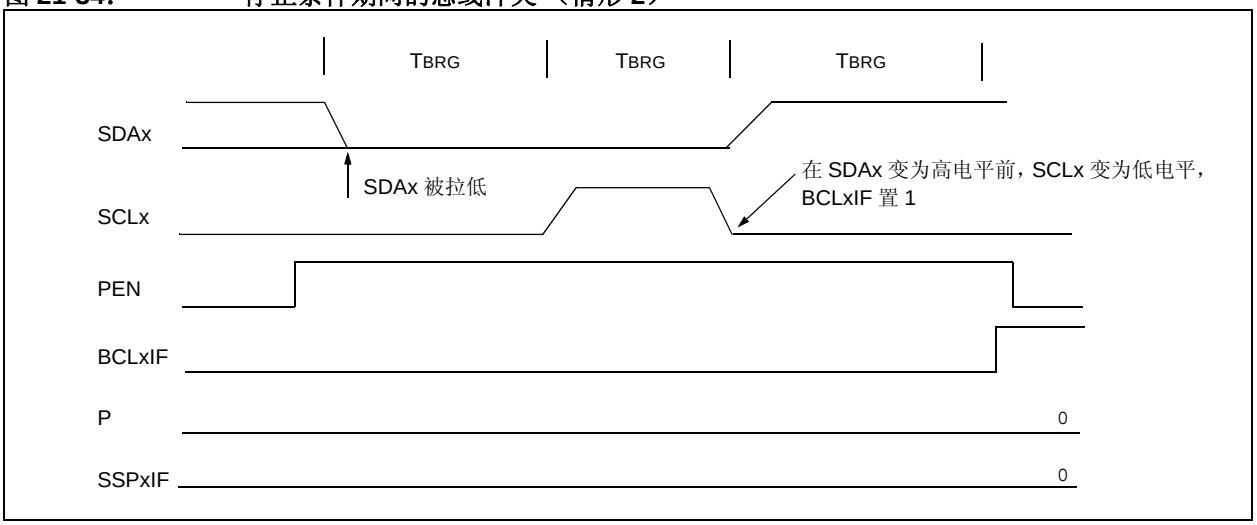


表 21-4： 与 I²C™ 操作相关的寄存器

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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PIR2	OSCFIF	—	SSP2IF	BLC2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	SSP2IE	BLC2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
IPR2	OSCFIP	—	SSP2IP	BLC2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0
SSP1BUF	MSSP1 接收缓冲器 / 发送寄存器							
SSP1ADD	MSSP1 地址寄存器 （I²C 从模式）， MSSP1 波特率重载寄存器 （I²C 主模式）							
SSP1MSK(1)	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
	GCEN	ACKSTAT	ADMSK5(2)	ADMSK4(2)	ADMSK3(2)	ADMSK2(2)	ADMSK1(2)	SEN
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
SSP2BUF	MSSP2 接收缓冲器 / 发送寄存器							
SSP2ADD	MSSP2 地址寄存器 （I²C 从模式）， MSSP2 波特率重载寄存器 （I²C 主模式）							
SSP2MSK(1)	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
	GCEN	ACKSTAT	ADMSK5(2)	ADMSK4(2)	ADMSK3(2)	ADMSK2(2)	ADMSK1(2)	SEN
SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

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： SSPxMSK 与 SSPxADD 共用 SFR 空间中的同一地址，但只有在 7 位掩码模式的特定 I²C 从工作模式中才能访问。更多详细信息，请参见第 21.4.3.4 节 “7 位地址掩码模式”。

2： 只有在 I²C 从模式下工作时才可使用备用位定义。

© 2011 Microchip Technology Inc.

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 也被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。也可以将其配置成能够与 A/D 或 D/A 集成电路、串行 EEPROM 等外设通信的半双工同步系统。

增强型 USART 模块实现了更多的功能，包括自动波特率检测和校准，以及在接收到“同步间隔”字符时自动唤醒和 12 位间隔字符发送。这些功能使 EUSART 模块成为局域互连网（LIN/J2602 总线）系统的理想选择。

PIC18F87K22 系列的所有器件都配备有两个独立的 EUSART 模块，分别是 EUSART1 和 EUSART2。可将这两个模块配置为以下几种模式：

- 带有以下功能的全双工异步模式：
  - 字符接收自动唤醒
  - 自动波特率校准
  - 12 位间隔字符发送
- 时钟极性可选的半双工同步主模式
- 时钟极性可选的半双工同步从模式

EUSART1 和 EUSART2 的引脚分别与 PORTC（RC6/TX1/CK1 和 RC7/RX1/DT1）和 PORTG（RG1/TX2/CK2/AN19/C3OUT 和 RG2/RX2/DT2/AN18/C3INA）功能复用。为了将这些引脚配置为 EUSART：

- 对于 EUSART1：
  - SPEN（RCSTA1<7>）位必须置 1（= 1）
  - TRISC<7> 位必须置 1（= 1）
  - 对于异步和同步主模式，TRISC<6> 位必须清零（= 0）
  - 对于同步从模式，TRISC<6> 位必须置 1（= 1）
- 对于 EUSART2：
  - SPEN（RCSTA2<7>）位必须置 1（= 1）
  - TRISG<2> 位必须置 1（= 1）
  - 对于异步和同步主模式，TRISG<1> 位必须清零（= 0）
  - 对于同步从模式，TRISC<6> 位必须置 1（= 1）

注：	EUSART 控制逻辑在需要时会自动将引脚从输入重新配置为输出。

增强型 USART 模块的操作由以下 3 个寄存器控制：

- 发送状态和控制寄存器（TXSTAx）
- 接收状态和控制寄存器（RCSTAx）
- 波特率控制寄存器（BAUDCONx）

这些寄存器分别如后面几页中的寄存器22-1、寄存器22-2 和寄存器 22-3 所述。

注：	在本章节中，通常在引用可能与特定 EUSART 模块相关的寄存器和位名称时使用 x 来代替特定模块号。因此，RCSTAx 可能是指 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-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
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 = 上电复位时的值	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 （时钟来自内部 BRG）
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 （在完成时由硬件清零）
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
该位可以是地址 / 数据位或奇偶校验位。

注 1： 同步模式下 SREN/CREN 的优先级高于 TXEN。

© 2011 Microchip Technology Inc.

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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **SPEN：**串口使能位
1 = 使能串口
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
此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、允许中断并装载接收缓冲器
0 = 禁止地址检测，接收到所有字节并且第 9 位可用作奇偶校验位
8 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 寄存器清除，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 位清除）
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，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-0	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 = 上电复位时的值	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 （必须由软件清零）
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：**数据 / 接收极性选择位
异步模式：
1 = 接收的数据 （RXx）反相 （低电平有效）
0 = 接收的数据 （RXx）未反相 （高电平有效）
同步模式：
1 = 数据 （DTx）反相 （低电平有效）
0 = 数据 （DTx）未反相 （高电平有效）

bit 4 **TXCKP：**同步时钟极性选择位
异步模式：
1 = 发送 （TXx）空闲状态为低电平
0 = 发送 （TXx）空闲状态为高电平
同步模式：
1 = 时钟 （CKx）空闲状态为高电平
0 = 时钟 （CKx）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 SPBRGx （兼容模式），忽略 SPBRGHx 的值

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——中断在下降沿产生，在下一个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 = 在下一个字符使能波特率检测。需要收到 “同步”字段 （55h）；完成时由硬件清零。
0 = 禁止波特率检测或检测已完成
同步模式：
在此模式下未使用。

© 2011 Microchip Technology Inc.

22.1 波特率发生器 （BRG）

BRG 是一个专用的 8 位或 16 位波特率发生器，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>）位和 BRG16（BAUDCONx<3>）位也控制波特率。在同步模式下，会忽略 BRGH 位。表 22-1 所示为不同 EUSART 模式下的波特率计算公式，但仅适用于主模式 （由内部产生时钟信号）。

给定目标波特率和 FOSC 的情况下，可以使用表 22-1 中的公式来计算 SPBRGHx:SPBRGx 寄存器的最近似整数值，从而确定波特率误差。例 22-1 给出了一个计算示例。表 22-2 中给出了各种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 有利于减少波特率误差，或者在快速振荡频率条件下实现低波特率。

向 SPBRGHx:SPBRGx 寄存器写入新值会引起 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。当工作在同步模式下时，不支持 0000h 和 0001h 的 SPBRGH:SPBRG 值。在异步模式下，可使用所有的 BRG 值。

22.1.1 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与先前不同的频率下。这可能需要调整 SPBRGx 寄存器对中的值。

22.1.2 采样

择多检测电路对 RXx 引脚数据 （RC7/RX1/DT1 或 RG2/RX2/DT2/AN18/C3INA）采样三次，以判定 RXx 引脚上出现的是高电平还是低电平。

表 22-1： 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 = 任意值， n = SPBRGHx:SPBRGx 寄存器对的值

例 22-1：　　计算波特率误差

针对工作在异步模式下、工作频率 FOSC 为 16 MHz、采用 8 位 BRG、目标波特率为 9600 bps 的器件：

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

波特率计算结果 = 16000000/(64 (25 + 1))

= 9615

误差 = (波特率计算结果 – 目标波特率)/ 目标波特率

= (9615 – 9600)/9600 = 0.16%

表 22-2：　　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							
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							
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注：　— = 未实现，读为 0。 BRG 不使用阴影单元。

© 2011 Microchip Technology Inc.

表 22-3： 异步模式下的波特率

目标波特率（K）	SYNC = 0， BRGH = 0， BRG16 = 0											
	FOSC = 40. 000 MHz			FOSC = 20. 000 MHz			FOSC = 10. 000 MHz			FOSC = 8. 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	1.201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2.403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9.615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

目标波特率（K）	SYNC = 0， BRGH = 0， BRG16 = 0								
	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）
0.3	0.300	0.16	207	0.300	-0.16	103	0.300	-0.16	51
1.2	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.404	0.16	25	2.403	-0.16	12	—	—	—
9.6	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	—	—	—	—	—	—

目标波特率（K）	SYNC = 0， BRGH = 1， BRG16 = 0											
	FOSC = 40. 000 MHz			FOSC = 20. 000 MHz			FOSC = 10. 000 MHz			FOSC = 8. 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	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

目标波特率（K）	SYNC = 0， BRGH = 1， BRG16 = 0								
	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）
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	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

表 22-3： 异步模式下的波特率 （续）

目标波特率（K）	SYNC = 0， BRGH = 0， BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 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	8332	0.300	0.02	4165	0.300	0.02	2082	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

目标波特率（K）	SYNC = 0， BRGH = 0， BRG16 = 1								
	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）
0.3	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

目标波特率（K）	SYNC = 0， BRGH = 1， BRG16 = 1 或 SYNC = 1， BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 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	33332	0.300	0.00	16665	0.300	0.00	8332	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-2.12	16

目标波特率（K）	SYNC = 0， BRGH = 1， BRG16 = 1 或 SYNC = 1， BRG16 = 1								
	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）
0.3	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

© 2011 Microchip Technology Inc.

22.1.3 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。此功能仅在异步模式下当 WUE 位清零时有效。

只要接收到起始位并且 ABDEN 位已置 1，就会开始自动波特率检测（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 的时钟是反向的。不是由 BRG 为 RXx 输入信号提供时钟源，而是由 RXx 信号为 BRG 定时。在 ABD 模式下，内部波特率发生器被用作计数器来计算输入的串行字节流的位间隔时间。

一旦 ABDEN 位置 1，状态机就会将 BRG 清零并寻找起始位。为了正确计算比特率，自动波特率检测必须接收到一个值为 55h （ASCII 字符 U，也是 LIN/J2602 总线的同步字符）的字节。为了尽量减少输入信号不对称所造成的影响，测量时段内要包含一个高位和一个低位时间。在起始位后， SPBRGx 使用预先选择的时钟源在 RXx 引脚上的第一个上升沿开始计数。在 RXx 引脚传输了 8 个位，或在检测到第 5 个上升沿后，会将在相应 BRG 周期内累加的值保存在 SPBRGHx:SPBRGx 寄存器对中。当第 5 个边沿出现时 （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。当 BRG 计满返回时，该位由硬件置 1，用户也可用软件将其置 1 或清零。在发生计满返回事件后，继续保持 ABD 模式， 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。请注意 BRG 时钟将由 BRG16 和 BRGH 位配置。必须将 BRG16 位置 1 以使 SPBRG1 和 SPBRGH1 都被用作一个 16 位计数器。通过检查 SPBRGHx 寄存器中的值是否为 00h，用户可以验证在 8 位模式下是否发生了进位。表 22-4 所示为 BRG 计数器的时钟速率。

当产生 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，来清零中断标志位 RCxIF。应丢弃 RCREGx 的值。

> **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：** 需要用户来判断进入的字符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，某些振荡频率和 EUSART 波特率的组合是无法实现的。在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：** 要最大程度扩大波特率范围，建议将 BRG16（BAUDCONx<3>）位置 1（如果已使用自动波特率功能）。

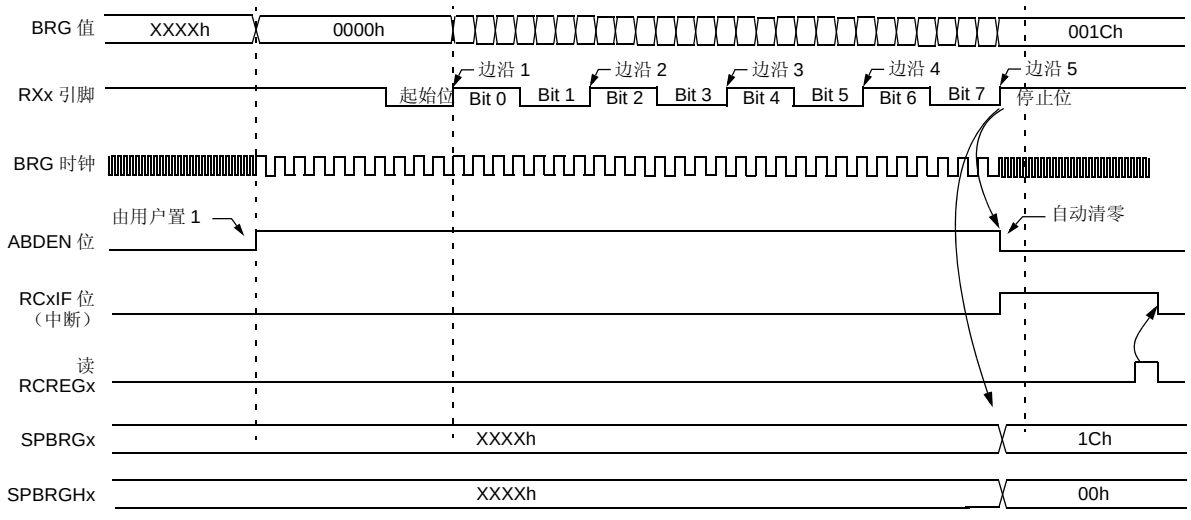
表 22-4： BRG 计数器时钟速率

BRG16	BRGH	BRG 计数器时钟
0	0	Fosc/512
0	1	Fosc/128
1	0	Fosc/128
1	1	Fosc/32

22.1.3.1 ABD 和 EUSART 发送

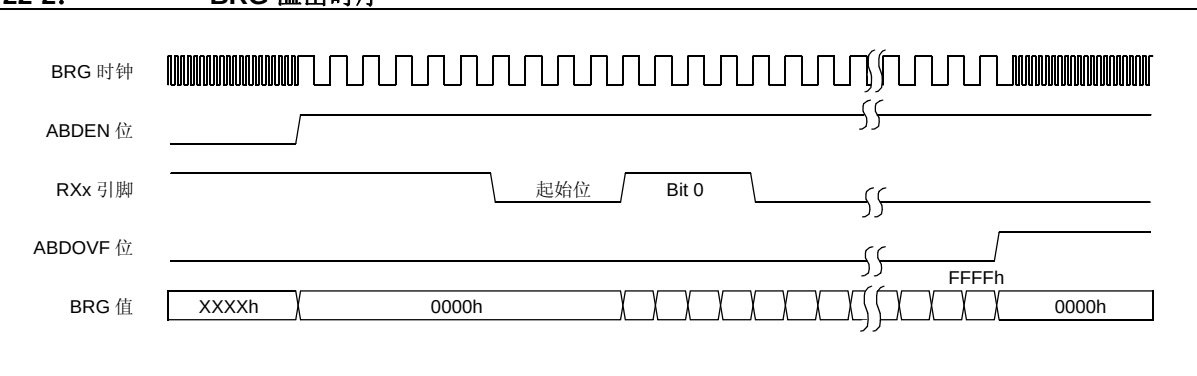
由于在 ABD 采集期间 BRG 时钟是反向的，因此在 ABD 期间不能使用 EUSART 发送器。这意味着只要 ABDEN 位置 1，就不能写入 TXREGx。用户还应确保在发送期间 ABDEN 不能为置 1 状态，否则可能会导致无法预料的 EUSART 操作。

图 22-1： 自动波特率计算



注： 需要将 EUSART 模块配置为异步模式，并且 WUE = 0 才能产生正确的 ABD 序列。

图 22-2： BRG 溢出时序



© 2011 Microchip Technology Inc.

22.2 EUSART 异步模式

通过将 SYNC 位 （TXSTAx<4>）清零可选择异步工作模式。在此模式下， EUSART 使用标准的不归零（Non-Return-to-Zero， NRZ）格式（1 个起始位， 8 个或 9 个数据位， 1 个停止位）。最常用的数据格式为 8 位。片上专用 8 位 /16 位波特率发生器可借助于振荡器产生标准波特率频率。

EUSART 首先发送和接收的是最低有效位。EUSART 的发送器和接收器在功能上是独立的，但采用相同的数据格式和波特率。波特率发生器可以根据 BRGH 和 BRG16 位 （TXSTAx<2> 和 BAUDCONx<3>）的设置产生两种不同的波特率时钟，频率分别为移位速率的 16 倍和 64 倍。硬件不支持奇偶校验，但可以用软件实现，校验位保存在第 9 个数据位中。

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
- 同步间隔字符自动唤醒
- 12 位间隔字符发送
- 自动波特率检测

22.2.1 EUSART 异步发送器

图 22-3 显示了 EUSART 发送器的框图。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。而 TXREGx 寄存器中的数据由软件装入。直到前一次装入的停止位已被发送，才会向 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 （如果有的话）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，同时标志位 TXxIF 置 1。可以通过将中断允许位 TXxIE 置 1 或清零来允许 / 禁止该中断。不管 TXxIE 的状态如何，只要中断发生， TXxIF 就会置 1 并且不能用软件清零。 TXxIF 也不能在 TXREGx 装入新数据后立即被清零，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。

标志位 TXxIF 指示的是 TXREGx 寄存器的状态，而另一位 TRMT （TXSTAx<1>）则指示 TSR 寄存器的状态。TRMT 是只读位，它在 TSR 寄存器为空时被置 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，用户只能对此位进行轮询。

> **注 1：** TSR 寄存器并未映射到数据存储器中，因此用户不能直接访问它。
>
> **2：** 当使能位 TXxEN 置 1 时，标志位 TXxIF 也置 1。

设置异步发送操作的步骤如下：

1. 初始化 SPBRGHx:SPBRGx 寄存器，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC 位清零并将 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. 若需要发送 9 位数据，将发送位 TX9 置 1 ；发送的第 9 位可以是地址位也可以是数据位。
5. 通过将 TXEN 位置 1 使能发送，此操作同时也会将 TXxIF 位置 1。
6. 如果选择发送 9 位数据，将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREGx 寄存器 （开始发送）。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

图 22-3： EUSART 发送框图

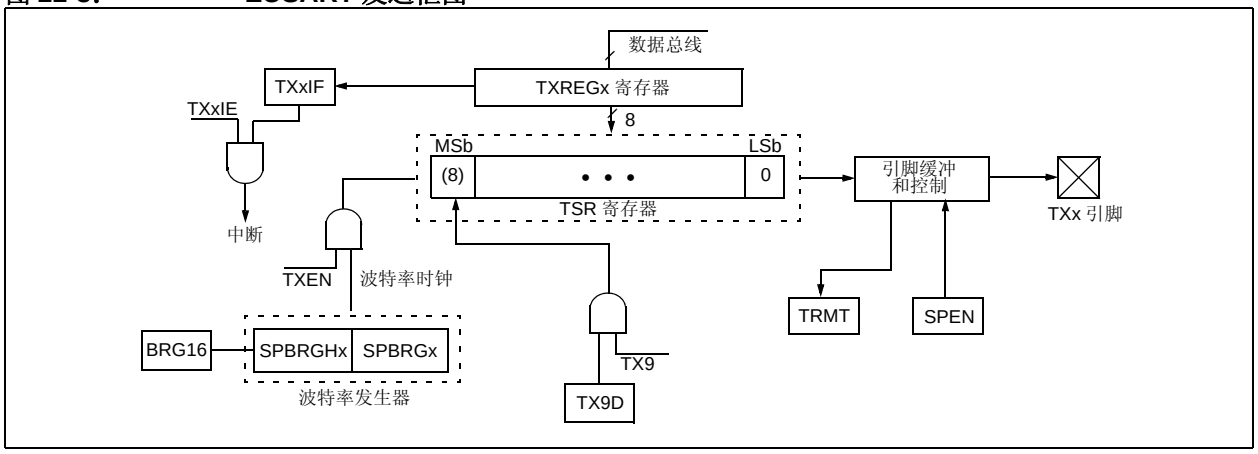


图 22-4： 异步发送

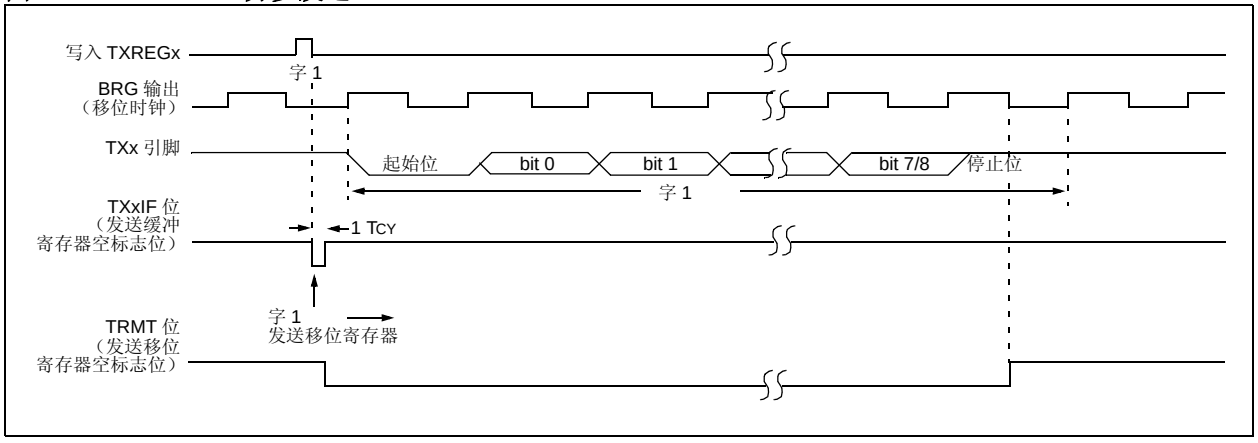
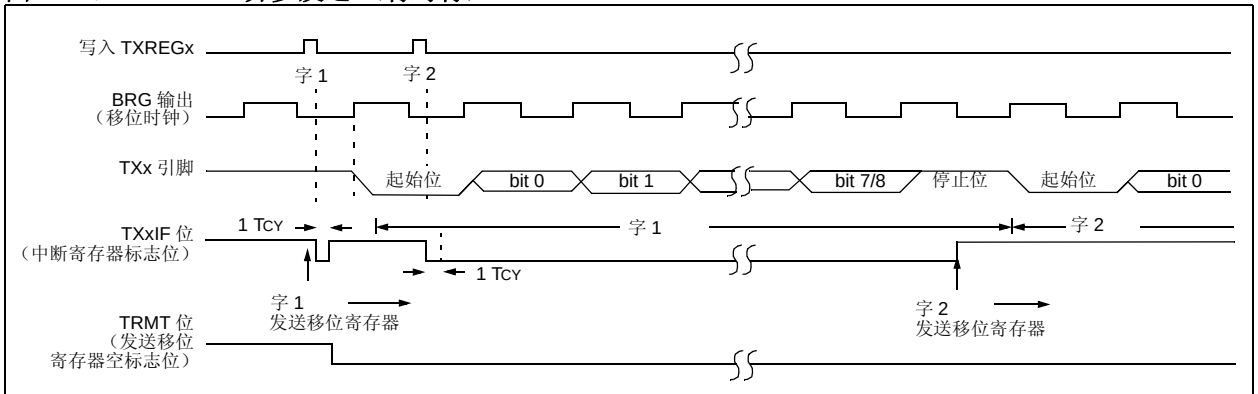


图 22-5： 异步发送 （背对背）



注： 此时序图表示的是两次连续的发送。

© 2011 Microchip Technology Inc.

表 22-5： 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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
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							
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							
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： — = 未实现，读为 0。异步发送不使用阴影单元。

22.2.2 EUSART 异步接收器

图 22-6 显示了接收器的框图。在 RXx 引脚上接收数据，并驱动数据恢复电路。数据恢复电路实际上是一个以 16 倍波特率为工作频率的高速移位器，而主接收串行移位器的工作频率等于比特率或 Fosc。此模式通常用于 RS-232 系统。

设置异步接收操作的步骤如下：

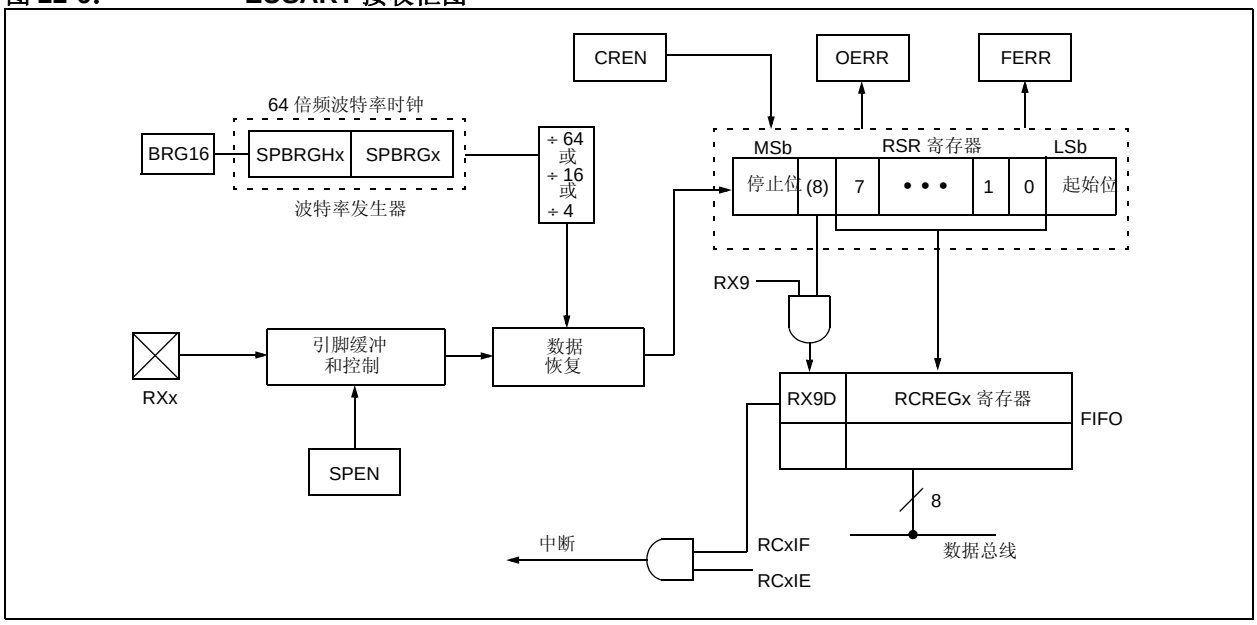
1. 初始化 SPBRGHx:SPBRGx 寄存器，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC 位清零并将 SPEN 位置 1 使能异步串口。
3. 如果需要中断，将允许位 RCxIE 置 1。
4. 若需要接收 9 位数据，将 RX9 位置 1。
5. 通过将 CREN 位置 1，使能接收。
6. 当接收完成时标志位 RCxIF 将被置 1，此时如果允许位 RCxIE 已置 1，还会产生一个中断。
7. 读 RCSTAx 寄存器以获取第 9 位数据（如果已使能），并判断是否在接收过程中发生了错误。
8. 通过读 RCREGx 寄存器来读取接收到的 8 位数据。
9. 如果发生错误，通过将使能位 CREN 清零来清除错误。
10. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位（INTCON<7:6>）置 1。

22.2.3 设置带有地址检测功能的 9 位模式

此模式通常用于 RS-485 系统。按如下步骤设置带有地址检测功能的异步接收操作：

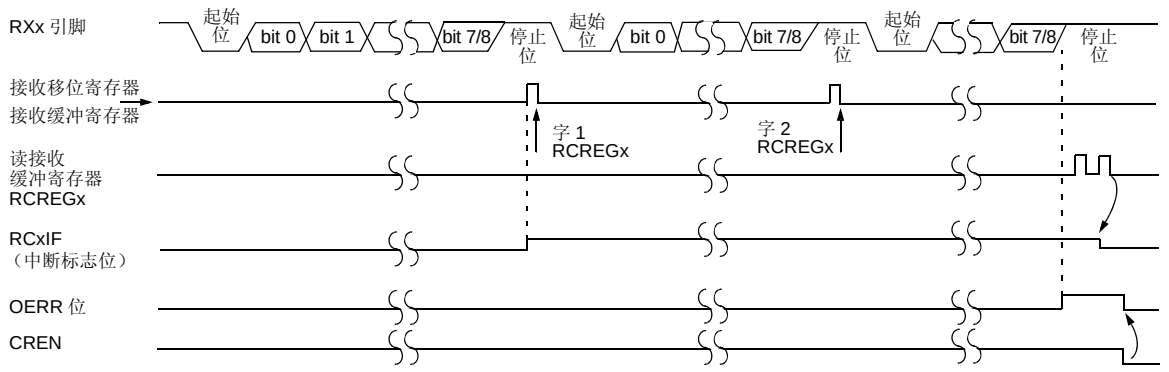
1. 初始化 SPBRGHx:SPBRGx 寄存器，设置合适的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC 位清零并将 SPEN 位置 1 使能异步串口。
3. 若需要中断，将 RCEN 位置 1 并使用 RCxIP 位设置优先级。
4. 将 RX9 位置 1，使能 9 位接收。
5. 将 ADDEN 位置 1，使能地址检测。
6. 将 CREN 位置 1，使能接收。
7. 当接收完成时 RCxIF 位将被置 1。此时如果 RCxIE 和 GIE 位已置 1，还将响应中断。
8. 读 RCSTAx 寄存器判断在接收时是否发生了错误，同时读取第 9 位数据（如果适用）。
9. 读 RCREGx 来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，允许接收到的所有数据进入接收缓冲器，并中断 CPU。

图 22-6： EUSART 接收框图



© 2011 Microchip Technology Inc.

图 22-7： 异步接收



注： 此时序图显示了在 RXx 输入引脚上的 3 个字。在第 3 个字后读取 RCREGx （接收缓冲器）会使 OERR （溢出错误）位置 1。

表 22-6： 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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
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							
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							
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： — = 未实现，读为 0。异步接收不使用阴影单元。

22.2.4 同步间隔字符自动唤醒

在休眠模式下，EUSART 的所有时钟都会暂停。因此，波特率发生器处于非激活状态，并且无法进行正确的字节接收。自动唤醒功能允许当RXx/DTx线上有事件发生时唤醒控制器，该功能需要 EUSART 工作在异步模式下。

通过将 WUE 位 （BAUDCONx<1>）置 1，使能自动唤醒功能。该功能启用后，将禁止 RXx/DTx 上的典型接收操作，且 EUSART 保持在空闲状态并监视唤醒事件（与 CPU 运行模式无关）。唤醒事件是指 RXx/DTx 线上发生由高到低的电平跳变。（这与 “同步间隔” 字符或 LIN/J2602 协议唤醒信号字符的启动条件一致。）

唤醒事件后，模块产生一个 RCxIF 中断。在正常工作模式下，中断会与 Q 时钟同步产生 （图 22-8）；如果器件处于休眠模式，则两者不同步 （图 22-9） 。通过读 RCREGx 寄存器可清除中断条件。

唤醒事件后，当 RXx 线上出现由低到高的电平跳变时，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。由此用户可知 “同步间隔” 事件已经结束。

22.2.4.1 使用自动唤醒功能的特别注意事项

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产生错误的字符结束（End-of-Character， EOC）信号并导致数据或帧错误。因此，为了确保正确的传输，必须首先发送全 0 字符。对于标准的 RS-232 器件，这可以是 00h （8 位），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，尤其是在采用起振延时较长的振荡器 （即， HS 或 HSPLL 模式）的应用中更要注意这一点。“同步间隔”（或唤醒信号）字符必须足够长，并且跟有足够长的时间间隔，以便使所选振荡器有充足的时间起振并保证 EUSART 正确初始化。

© 2011 Microchip Technology Inc.

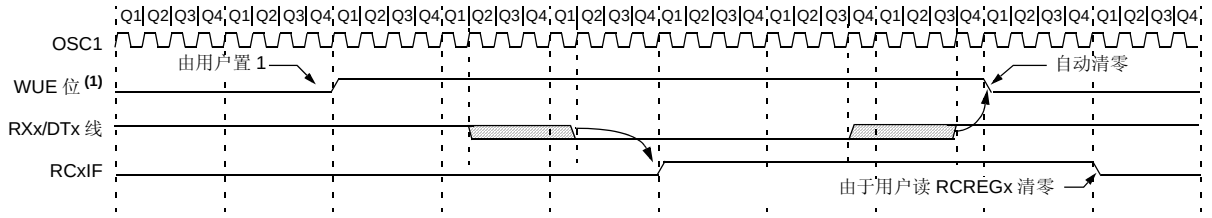
22.2.4.2 使用 WUE 位时的特殊注意事项

用 WUE 和 RCxIF 事件的时序来判断接收数据的有效性可能会引起混淆。如前所述，将 WUE 位置 1 会使 EUSART 进入空闲模式。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 出现上升沿时 WUE 位清零。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）且 RCxIF 标志位置 1 并不能表明 RCREGx 中的数据是完整的。用户还应该考虑使用固件验证是否完整地接收了数据。

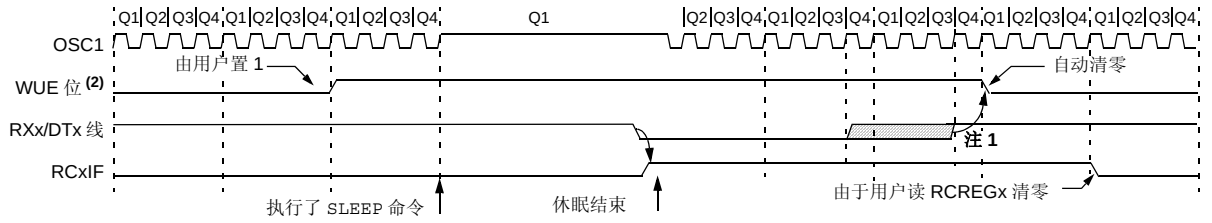
要确保没有丢失有效数据，应检查 RCIDL 位来验证是否还在接收数据。如果不在接收数据，则可将 WUE 位置 1，使器件立即进入休眠模式。

图 22-8： 正常工作模式下的自动唤醒位（WUE）时序



注 1：WUE 位置 1 时， EUSART 保持在空闲状态。

图 22-9： 休眠模式下的自动唤醒位（WUE）时序



注 1： 如果唤醒事件需要较长的振荡器预热时间，在振荡器准备就绪之前 WUE 位会自动清零。此过程与 Q 时钟出现与否无关。
2： WUE 位置 1 时， EUSART 保持在空闲状态。

22.2.5 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。发送的间隔字符包括 1 个起始位，后面跟有 12 个 0 位和一个停止位。当发送移位寄存器装有数据时，只要 SENDB 和 TXEN 位（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，并会发送全 0。

在发送了相应的停止位后，硬件会自动将 SENDB 位清零。这样用户可以在发送间隔字符（在 LIN/J2602 规范中通常是同步字符）后将下一个要发送的字节预先装入发送 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。

如其在正常发送操作中一样，TRMT 位用于指示发送正在进行还是处于空闲状态。关于间隔字符序列，请参见图 22-10。

22.2.5.1 间隔和同步发送序列

下列序列会发送一个报文帧头，包括一个间隔字符和其后的自动波特率同步字节。此序列适用于典型的 LIN/J2602 总线主器件。

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，以便把同步字符装入发送 FIFO 缓冲器。
5. 间隔字符发送后，硬件会将 SENDB 位复位。此时，同步字符会以预先配置的模式发送。

当 TXxIF 指出 TXREGx 为空以后，下一个数据字节会被写入 TXREGx。

22.2.6 接收间隔字符

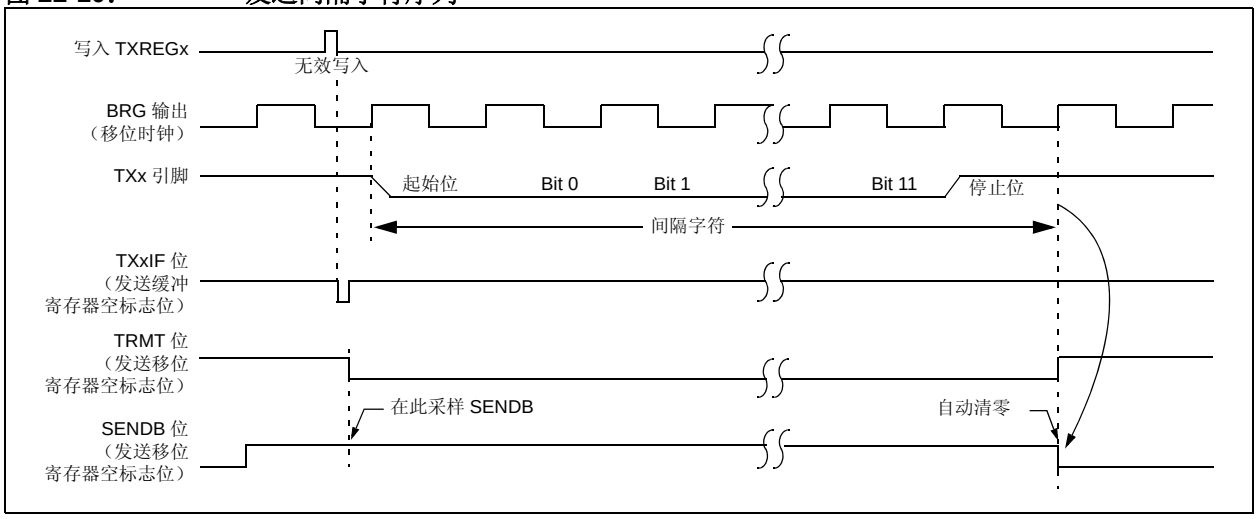
增强型 USART 模块接收间隔字符有两种方法。

第一种方法是强制将波特率配置为典型速率的 9/13。这可以使停止位在正确的采样点（对于间隔字符为起始位之后的 13 位，对于典型数据则是 8 个数据位）产生。

第二种方法使用**第 22.2.4 节“同步间隔字符自动唤醒”**中描述的自动唤醒功能。通过使能此功能，EUSART 将采样 RXx/DTx 引脚上电平的下两次跳变，产生一个 RCxIF 中断，接收下一个数据字节，并在随后产生另一个中断。

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。无论使用哪种方法，用户都可以在检测到 TXxIF 中断时马上将 ABDEN 位置 1。

图 22-10：发送间隔字符序列



© 2011 Microchip Technology Inc.

22.3 EUSART 同步主模式

将 CSRC 位（TXSTAx<7>）置 1 可以进入同步主模式。在此模式中，数据以半双工方式（即发送和接收不同时进行）发送。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，分别把 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 时钟线上发送主时钟信号。时钟极性是通过 TXCKP 位 （BAUDCONx<4>）选择的。将 TXCKP 置 1 可将 CKx 的空闲状态设置为高电平，将该位清零则可将空闲状态设置为低电平。此选项支持将本模块与 Microwire 器件配合使用。

22.3.1 EUSART 同步主发送

图 22-3 显示了 EUSART 发送器的框图。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。而TXREGx寄存器中的数据由软件装入。在前一次装入数据的最后一位发送完成后，才会向 TSR 寄存器装入新数据。一旦最后一位发送完成，就会将TXREGx寄存器中的新数据（如果有的话）装入 TSR。

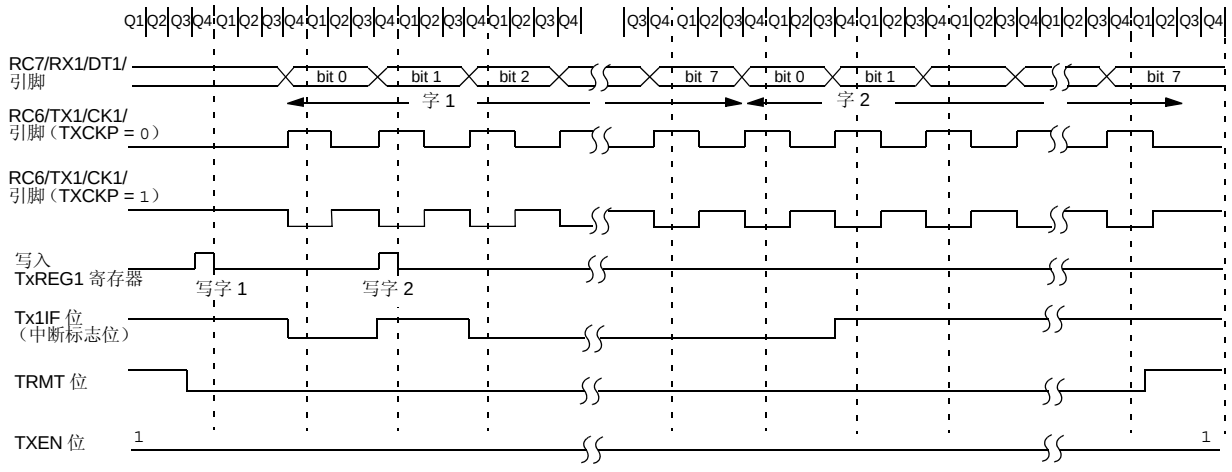
一旦 TXREGx 寄存器向 TSR 寄存器传输了数据 （在 1 个 TCY 内发生），TXREGx 寄存器就为空，同时标志位 TXxIF 置 1。可以通过将中断允许位 TXxIE 置 1 或清零来允许 / 禁止该中断。不管中断允许位 TXxIE 的状态如何，在发生中断时 TXxIF 都会置 1 ；且不能用软件清零。只有在新数据装入 TXREGx 寄存器时，TXxIF 才会复位。

TXxIF 标志位指示 TXREGx 寄存器的状态，而另一位 TRMT（TXSTAx<1>）则表示TSR寄存器的状态。TRMT 是只读位，它在 TSR 寄存器为空时被置 1。TRMT 位与任何中断逻辑均无关联，因此要判断 TSR 寄存器是否为空，用户只能对此位进行轮询。TSR 寄存器并未映射到数据存储器中，因此用户不能直接访问它。

设置同步主发送操作的步骤如下：

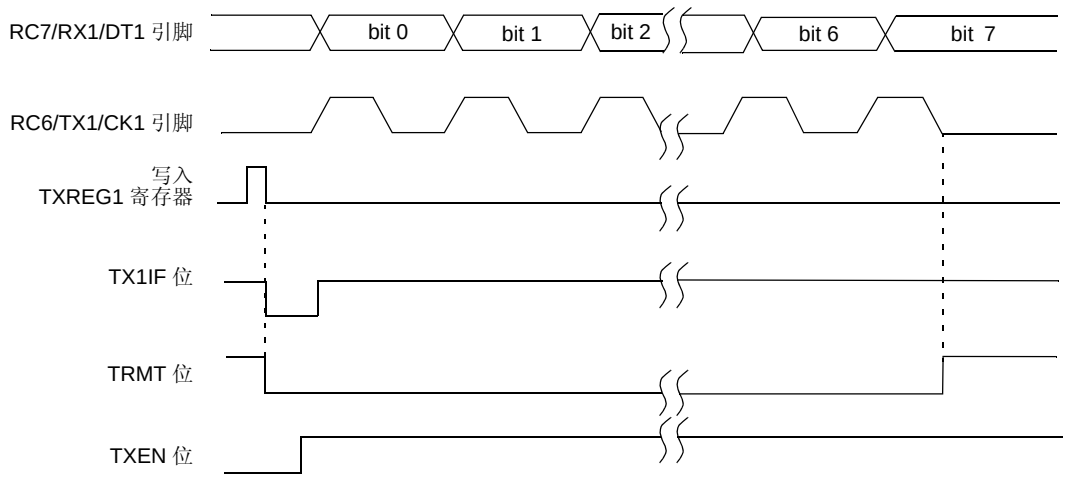
1. 初始化 SPBRGHx:SPBRGx 寄存器，设置合适的波特率。按需要将 BRG16 位置 1 或清零，以获得目标波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1，使能同步主串口。
3. 如果需要中断，将中断允许位 TXxIE 置 1。
4. 若需要发送 9 位数据，将 TX9 位置 1。
5. 将 TXEN 位置 1，使能发送。
6. 如果选择发送9位数据，将第9位数据装入TX9D位。
7. 将数据装入 TXREGx 寄存器，启动发送。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

图 22-11： 同步发送



注： 此为SPBRGx = 0且连续发送两个8位字的同步主模式。本例同样适用于EUSART2（RG1/TX2/CK2/AN19/C3OUT和RG2/RX2/DT2/AN18/C3INA）。

图 22-12： 同步发送 （由 TXEN 位控制）



注： 本例同样适用于 EUSART2 （RG1/TX2/CK2/AN19/C3OUT 和 RG2/RX2/DT2/AN18/C3INA）。

表 22-7： 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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
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							
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							
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： — = 未实现，读为 0。同步主发送不使用阴影单元。

© 2011 Microchip Technology Inc.

22.3.2 EUSART 同步主接收

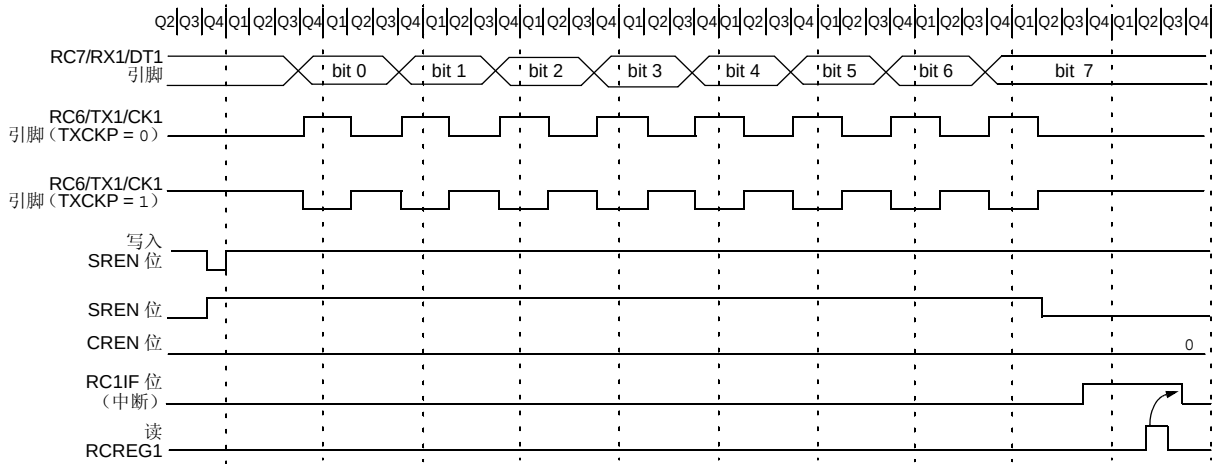
一旦选择了同步模式，只要将单字节接收使能位 SREN（RCSTAx<5>）或连续接收使能位CREN（RCSTAx<4>）置 1，即可使能接收。在时钟的下降沿采样 RXx 引脚上的数据。

如果将使能位 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，直到将 CREN 位清零。如果两个位均被置 1，则 CREN 具有优先权。

设置同步主接收操作的步骤如下：

1. 初始化 SPBRGHx:SPBRGx 寄存器，设置合适的波特率。按需要将 BRG16 位置 1 或清零，以获得目标波特率。
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，将允许位 RCxIE 置 1。
5. 若需要接收 9 位数据，将 RX9 位置 1。
6. 若需单字节接收，将 SREN 位置 1。若需连续接收，将 CREN 位置 1。
7. 当接收完成时中断标志位 RCxIF 将置 1，此时如果中断允许位RCxIE已置1，还将产生一个中断。
8. 读 RCSTAx 寄存器以获取第 9 位数据 （如果已使能），并判断是否在接收过程中发生了错误。
9. 通过读 RCREGx 寄存器来读取接收到的 8 位数据。
10. 如果发生错误，通过将 CREN 位清零以清除错误。
11. 若想使用中断，请确保将INTCON寄存器中的GIE和 PEIE 位 （INTCON<7:6>）置 1。

图 22-13： 同步主接收模式 （由 **SREN** 位控制）



注： 此时序图为SREN = 1且BRGH = 0时的同步主模式。本例同样适用于EUSART2（RG1/TX2/CK2/AN19/C3OUT和RG2/RX2/DT2/AN18/C3INA）。

表 22-8： 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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
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							
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							
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： — = 未实现，读为 0。同步主接收不使用阴影单元。

© 2011 Microchip Technology Inc.

22.4 EUSART 同步从模式

将 CSRC（TXSTAx<7>）位清零可进入同步从模式。此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（主模式中由内部时钟提供）。这使得器件能在任何低功耗模式下发送或接收数据。

22.4.1 EUSART 同步从发送

除了休眠模式以外，同步主模式和同步从模式的工作方式完全相同。

如果向 TXREGx 寄存器写入两个字，然后执行 SLEEP 指令，则将发生以下事件：

a) 第一个字立即传送到 TSR 寄存器进行发送。
b) 第二个字仍保留在 TXREGx 寄存器中。
c) 不会将标志位 TXxIF 置 1。
d) 当第一个字移出 TSR 后，TXREGx 寄存器将把第二个字送入 TSR，同时将标志位 TXxIF 置 1。
e) 如果中断允许位 TXxIE 已置 1，中断将把器件从休眠状态唤醒。如果允许了全局中断，程序则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. 将 CREN 和 SREN 位清零。
3. 如果需要中断，将中断允许位 TXxIE 置 1。
4. 若需要发送 9 位数据，将 TX9 位置 1。
5. 将使能位 TXEN 置 1，使能发送。
6. 如果选择发送 9 位数据，将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREGx 寄存器，启动发送。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位（INTCON<7:6>）置 1。

表 22-9： 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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
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							
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							
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： — = 未实现，读为 0。同步从发送不使用阴影单元。

22.4.2 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为无关位的 SREN 位以外，同步主模式和同步从模式的工作方式完全相同。

如果在进入休眠或空闲模式前将 CREN 位置 1 使能接收，那么在低功耗模式下可以接收到一个数据字。接收到该字后，RSR 寄存器将把数据传输到 RCREGx 寄存器。如果中断允许位 RCxIE 已置 1，则产生的中断将把器件从低功耗模式唤醒。如果允许了全局中断，程序则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 位清零使能同步主串口。
2. 如果需要中断，将中断允许位 RCxIE 置 1。
3. 若需要接收 9 位数据，将 RX9 位置 1。
4. 将使能位 CREN 置 1，使能接收。
5. 当接收完成时， RCxIF 标志位将被置 1。如果中断允许位 RCxIE 已置 1，还将产生一个中断。
6. 读 RCSTAx 寄存器以获取第 9 位数据 （如果已使能），并判断是否在接收过程中发生了错误。
7. 通过读 RCREGx 寄存器来读取接收到的 8 位数据。
8. 如果发生错误，通过将 CREN 位清零来清除错误。
9. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

表 22-10： 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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
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							
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							
ODCON3	U2OD	U1OD	—	—	—	—	—	CTMUDS
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： — = 未实现，读为 0。同步从接收不使用阴影单元。

© 2011 Microchip Technology Inc.

23.0 12 位模数转换器 （A/D）模块

PIC18F87K22 系列器件中的 64 引脚器件的模数（Analog-to-Digital，A/D）转换器模块有 16 路输入，而 80 引脚器件有 24 路输入。此模块能将一个模拟输入信号转换成相应的 12 位有符号数字信号。

此模块有 5 个寄存器：

- 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）
- A/D 端口配置寄存器 2 （ANCON2）
- ADRESH （A/D 结果寄存器高字节）
- ADRESL （A/D 结果寄存器低字节）

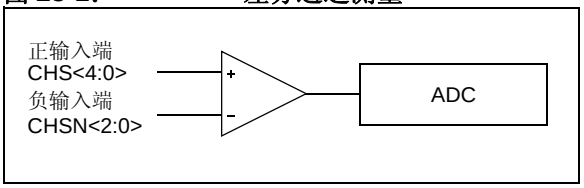
ADCON0 寄存器（如寄存器 23-1 所示）控制 A/D 模块的工作。ADCON1 寄存器（如寄存器 23-2 所示）配置参考电压和特殊事件触发信号选择。 ADCON2 寄存器（如寄存器 23-3 所示）配置 A/D 时钟源、可编程采集时间和输出结果的对齐方式。

23.1 差分 A/D 转换器

PIC18F87K22 系列器件的转换器实现为差分 A/D，用于测量出两个通道之间的差分电压并转换为数字信号（见图 23-1）。

也可通过清零 CHSN 位（ADCON1<2:0>）将转换器配置为测量单个输入的电压。在此配置下，负通道输入从内部连接到 AVss （见图 23-2）。

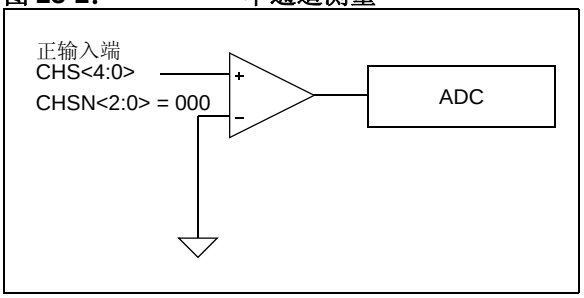
图 23-1： 差分通道测量



差分转换为单位增益差分放大器提供了两个输入通道。使用 CHS 位 （ADCON0<6:2>）选择正输入通道，而使用 CHSN 位 （ADCON1<2:0>）选择负输入通道。

放大器的输出提供给 A/D 转换器，如图 23-1 所示。 12 位结果保存在 ADRESH 和 ADRESL 寄存器中。除此之外，还有一个符号位，用于表征 12 位结果是正值还是负值。

图 23-2： 单通道测量



在单通道测量模式中，通过清零CHSN位（ADCON1<2:0>）将负输入通道连接到 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 = 上电复位时的值	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）
00110 = 通道 06 （AN6）
00111 = 通道 07 （AN7）
01000 = 通道 08 （AN8）
01001 = 通道 09 （AN9）
01010 = 通道 10 （AN10）
01011 = 通道 11 （AN11）
01100 = 通道 12 （AN12）(1,2)
01101 = 通道 13 （AN13）(1,2)
01110 = 通道 14 （AN14）(1,2)
01111 = 通道 15 （AN15）(1,2)
10000 =通道 16 （AN16）
10001 =通道 17 （AN17）
10010 =通道 18 （AN18）
10011 =通道 19 （AN19）
10100 =通道 20 （AN20）(1,2)
10101 =通道 21 （AN21）(1,2)
10110 =通道 22 （AN22）(1,2)
10111 =通道 23 （AN23）(1,2)
11000 =（保留）(2)
11001 =（保留）(2)
11010 =（保留）(2)
11011 =（保留）(2)
11100 = 通道 28 （保留，用于 CTMU）
11101 =通道 29 （内部温度二极管）
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（或校准）周期正在进行。将该位置 1 时启动 A/D 转换周期。A/D 转换完成后，通过硬件自动将该位清零。
0 = A/D 转换已完成或未进行

bit 0 **ADON：**A/D 使能位

1 = A/D 转换器正在工作
0 = A/D 转换器模块已关闭且不消耗任何工作电流

注 1： 这些通道在 64 引脚器件上未实现。
2： 在未实现通道上执行转换操作将返回随机值。

© 2011 Microchip Technology Inc.

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-0	R/W-0	R/W-0	R/W-0
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 = 上电复位时的值	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 RTCC 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 ECCP2 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>**：模拟负通道选择位
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 = 上电复位时的值	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 个 TAD
110 = 16 个 TAD
101 = 12 个 TAD
100 = 8 个 TAD
011 = 6 个 TAD
010 = 4 个 TAD
001 = 2 个 TAD
000 = 0 个 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 模块的 RC 振荡器）(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC （时钟来自 A/D 模块的 RC 振荡器）(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

注 1： 如果选择了 A/D 模块的 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。这可以确保在开始转换之前执行 SLEEP 指令。

© 2011 Microchip Technology Inc.

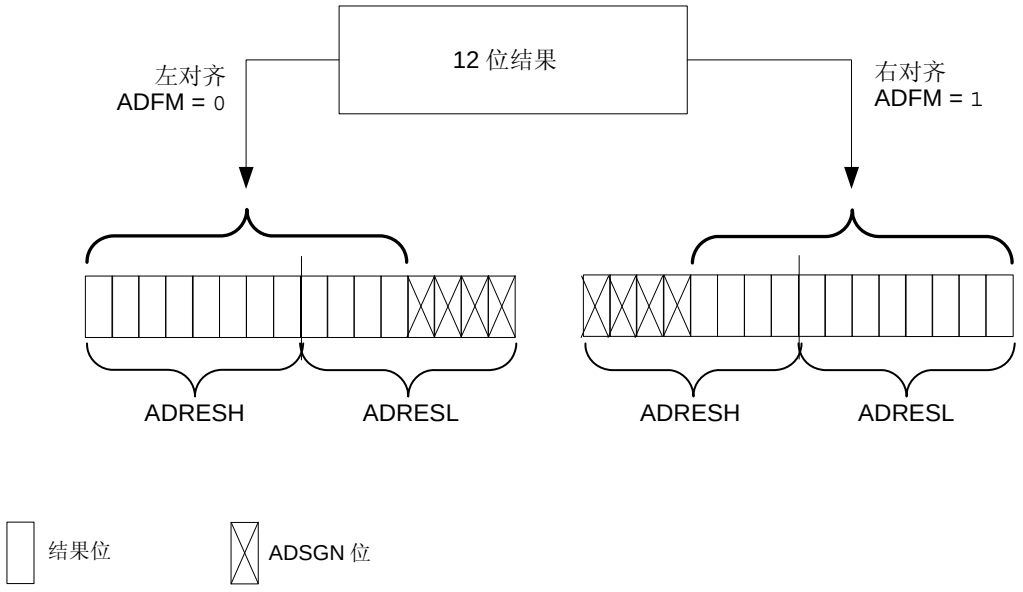
23.2.2 A/D 结果寄存器

ADRESH:ADRESL 寄存器对用于装入转换完成后的 12 位 A/D 结果和扩展符号位 （ADSGN）。此寄存器对为 16 位宽。A/D 模块使 16 位结果寄存器中的 12 结果可灵活地左对齐或者右对齐。A/D 结果格式选择位（ADFM）控制对齐方式。

图 23-3 给出了 A/D 结果对齐方式和扩展符号位（ADSGN）的位置。通过扩展符号位能够更加轻松地对结果执行 16 位运算。结果表示为一个二进制补码值。这意味着当符号位和数值位被视为右对齐时，ADRESH 和 ADRESL 寄存器可以读为单个有符号整数值。

当 A/D 转换器被禁止时，可将这两个 8 位寄存器用作两个通用寄存器。

图 23-3： A/D 结果对齐方式



二进制补码示例结果数字行

左对齐		右对齐	
十六进制	十进制	十六进制	十进制
0xFFF0	4095	0x0FFF	4095
0xFFE0	4094	0x0FFE	4094
...	...	...	...
0x0020	2	0x0002	2
0x0010	1	0x0001	1
0x0000	0	0x0000	0
0xFFFF	-1	0xFFFF	-1
0xFFEF	-2	0xFFFE	-2
...	...	...	...
0x001F	-4095	0xF001	-4095
0x000F	-4096	0xF000	-4096

寄存器 23-4： ADRESH：A/D 结果寄存器高字节 （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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-0 **ADRES<11:4>**：A/D 结果寄存器高字节位

寄存器 23-5： ADRESL：A/D 结果寄存器低字节 （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
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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-4 **ADRES<3:0>**：A/D 结果寄存器低字节位

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

© 2011 Microchip Technology Inc.

寄存器 23-6： ADRESH：A/D 结果寄存器高字节 （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
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 = 上电复位时的值	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 结果寄存器高字节位

寄存器 23-7： ADRESL：A/D 结果寄存器低字节 （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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-0 **ADRES<7:0>：**A/D 结果寄存器低字节位

ANCONx 寄存器用于配置与每个模拟通道相关联的 I/O 引脚的操作。清零 ANSELx 位将相应的引脚（ANx）配置为仅作为数字 I/O 进行操作。置 1 ANSELx 将 ANx 引脚配置为作为 A/D 转换器或比较器模块的模拟输入进行操作，并且禁止所有数字外设，数字输入均读为 0。

通常，发生任何器件复位时，与模拟输入复用的 I/O 引脚默认为模拟操作。

寄存器 23-8： ANCON0：A/D 端口配置寄存器 0

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANSEL7	ANSEL6	ANSEL5	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-0 **ANSEL<7:0>：**模拟端口配置位 （AN7 和 AN0）
1 = 引脚配置为模拟通道——禁止数字输入且任何输入都读为 0
0 = 引脚配置为数字端口

寄存器 23-9： ANCON1：A/D 端口配置寄存器 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
ANSEL15(1)	ANSEL14(1)	ANSEL13(1)	ANSEL12(1)	ANSEL11	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-0 **ANSEL<15:8>：**模拟端口配置位 （AN8 至 AN15）(1)
1 = 引脚配置为模拟通道——禁止数字输入且任何输入都读为 0
0 = 引脚配置为数字端口

注 1： AN12 至 AN15 以及 AN20 至 AN23 仅在 80 引脚器件上实现。对于 64 引脚器件，也实现了这些通道的相应 ANSELx 位，但是不起作用。

© 2011 Microchip Technology Inc.

寄存器 23-10： ANCON2：A/D 端口配置寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANSEL23[(1)]	ANSEL22[(1)]	ANSEL21[(1)]	ANSEL20[(1)]	ANSEL19	ANSEL18	ANSEL17	ANSEL16
bit 7							bit 0

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

bit 7-0 **ANSEL<23:16>：**模拟端口配置位 （AN16 至 AN23）[(1)]
1 = 引脚配置为模拟通道——禁止数字输入且任何输入都读为 0
0 = 引脚配置为数字端口

注 1： AN12 至 AN15 以及 AN20 至 AN23 仅在 80 引脚器件上实现。对于 64 引脚器件，也实现了这些通道的相应 ANSELx 位，但是不起任何作用。

可通过软件选择器件的正电源电压和负电源电压（AVDD和AVSS）或RA3/AN3/VREF+和RA2/AN2/VREF-引脚上的电压作为模拟参考电压。VREF+ 还有两个内部参考电压供选择：2.048V 和 4.096V。

A/D 转换器具有可在器件休眠模式下工作的独特特性。要使 A/D 转换器在休眠模式下工作，A/D 转换时钟必须来自 A/D 转换器内部的 RC 振荡器。

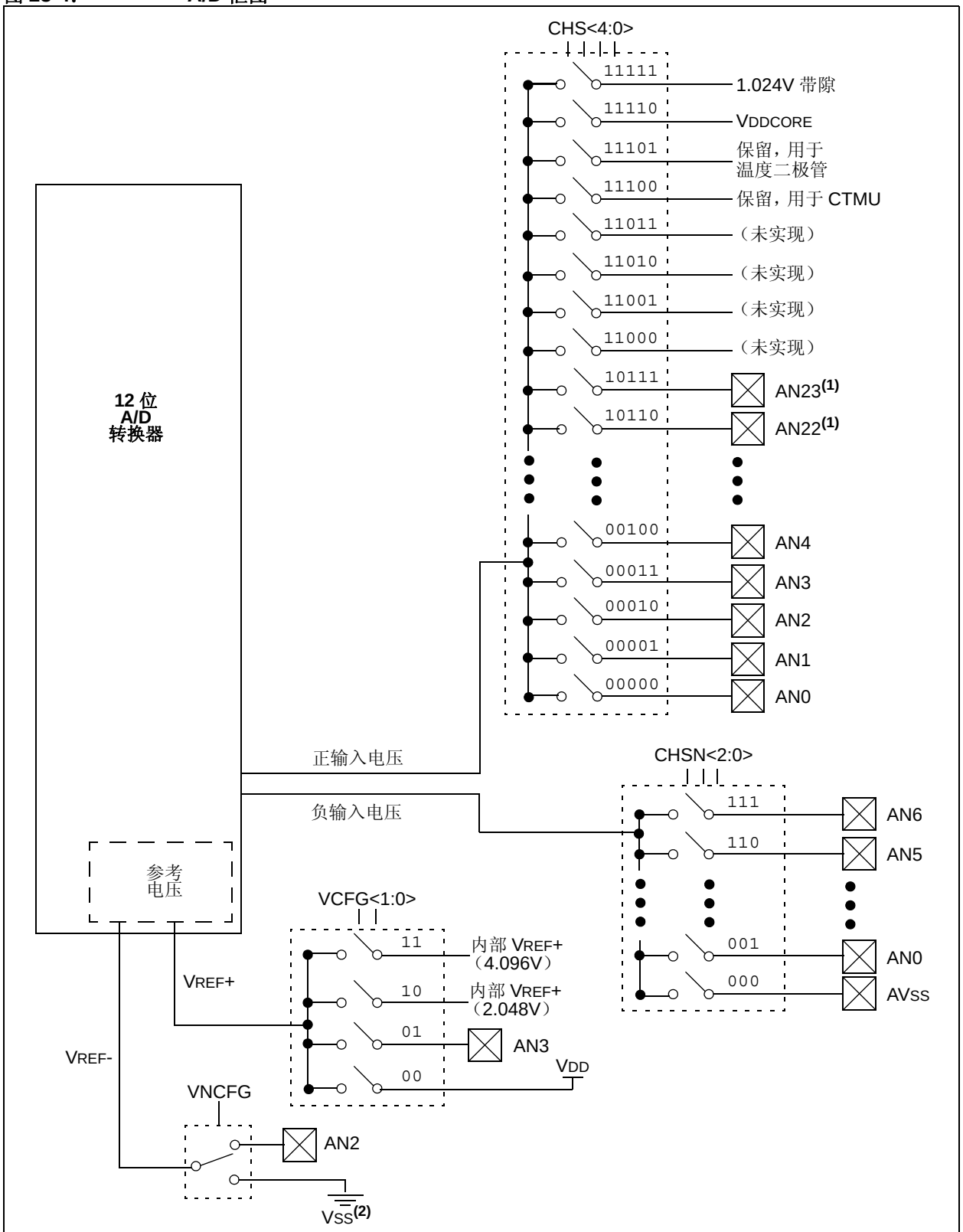
采样保持 （Sample-and-Hold， S/H）电路的输出是转换器的输入，转换器采用逐次逼近法得到转换结果。

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 转换完成之后，转换结果被装入 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>）置 1。

器件复位操作强制所有寄存器进入复位状态。这将迫使 A/D 模块关闭，并中止所有正在进行的转换。上电复位时， ADRESH:ADRESL 寄存器对中的值保持不变。上电复位后，这两个寄存器中的数据不确定。

A/D 模块的框图如图 23-4 所示。

图 23-4： A/D 框图



注 1：通道 AN12 至 AN15 以及 AN20 至 AN23 在 64 引脚器件上不可用。
2：在 I/O 引脚与 VDD 和 VSS 之间连有保护二极管。

© 2011 Microchip Technology Inc.

在根据需要配置好 A/D 模块后，在转换开始之前必须先对所选择的通道进行采集。必须将模拟输入通道相应的 TRIS 位设置为输入。采集时间的确定请参见**第 23.3 节 “A/D 采集要求”**。在采集完成后，即可启动 A/D 转换。采集时间可以被编程为位于 GO/DONE 位置 1 和实际启动转换之间。

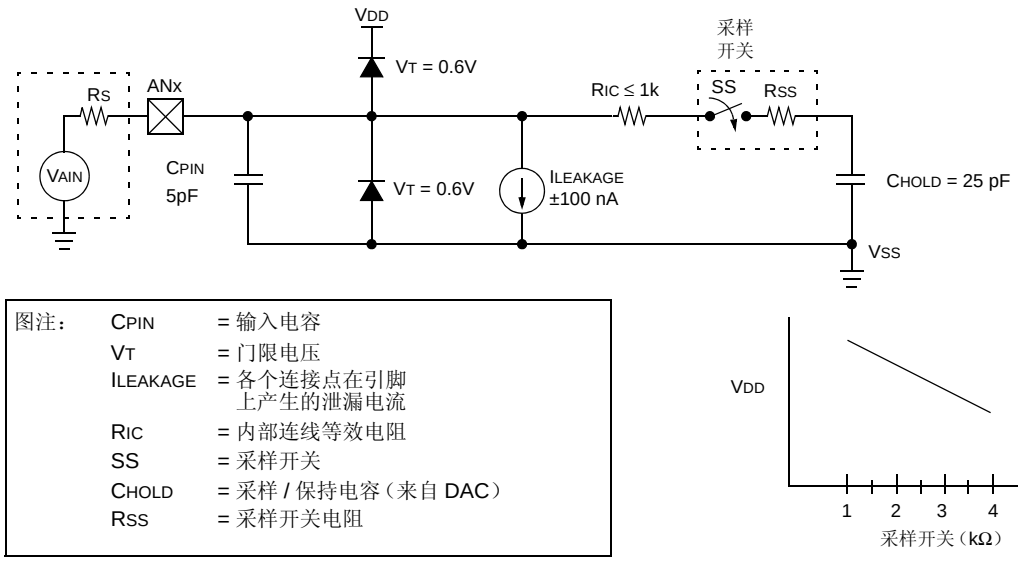
执行 A/D 转换时应该遵循以下步骤：

1. 配置 A/D 模块：
   - 将所需的 ADC 引脚配置为模拟引脚 （通过 ANCON0、 ANCON1 和 ANCON2）
   - 设置参考电压 （通过 ADCON1）
   - 选择 A/D 正负输入通道 （通过 ADCON0 和 ADCON1）
   - 选择 A/D 采集时间 （通过 ADCON2）
   - 选择 A/D 转换时钟 （通过 ADCON2）
   - 使能 A/D 模块 （通过 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/$\overline{\text{DONE}}$ 位 （ADCON0<1>）置 1
5. 等待 A/D 转换完成，通过以下两种方法之一判断转换是否完成：
   - 查询 GO/$\overline{\text{DONE}}$ 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），需要时将 ADIF 位清零。
7. 如需再次进行 A/D 转换，请根据需要从步骤 1 或步骤 2 重新开始执行。

   每位的 A/D 转换时间被定义为 TAD。在下一次采集开始之前至少需要等待 2 个 TAD。

图 23-5： 模拟输入电路模型



图注：

CPIN	=	输入电容
VT	=	门限电压
ILEAKAGE	=	各个连接点在引脚上产生的泄漏电流
RIC	=	内部连线等效电阻
SS	=	采样开关
CHOLD	=	采样 / 保持电容（来自 DAC）
RSS	=	采样开关电阻

23.3 A/D 采集要求

为了使 A/D 转换器达到规定精度，必须使充电保持电容（CHOLD）充满至输入通道的电平。模拟输入的电路模型如图 23-5 所示。模拟信号源的阻抗（RS）和内部采样开关阻抗（RSS）直接影响给电容 CHOLD 充电所需的时间。采样开关阻抗（RSS）的值随器件电压（VDD）不同而变化。

信号源阻抗影响模拟输入的失调电压（由于引脚泄漏电流引起）。**模拟信号源的最大阻抗推荐值为 2.5 kΩ**。选择（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：	当开始转换时，要将保持电容与输入引脚断开。

可使用公式 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 1024 步）。1/2 LSb 的误差是 A/D 模块达到规定分辨率所能允许的最大误差。

公式 23-3 给出了计算所需的最小采集时间 TACQ 的过程。计算结果基于以下应用系统假设：

- CHOLD = 25 pF
- RS = 2.5 kΩ
- 转换误差 ≤ 1/2 LSb
- VDD = 3V → RSS = 2 kΩ
- 温度 = 85°C

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， TCOFF = 0 ms。

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S)\ \ln(1/2048)\ \mu s$$
$$-(25\ pF)\ (1\ k\Omega + 2\ k\Omega + 2.5\ k\Omega)\ \ln(0.0004883)\ \mu s$$
$$1.05\ \mu s$$

$$T_{ACQ} = 0.2\ \mu s + 1.05\ \mu s + 1.2\ \mu s$$
$$2.45\ \mu s$$

© 2011 Microchip Technology Inc.

23.4 选择和配置自动采集时间

用户可以利用 ADCON2 寄存器选择采集时间，该采集时间发生在每次 GO/DONE 位置 1 之后。

当 GO/DONE 位置 1 时，采样停止并启动转换。用户应确保在选定所需输入通道之后到 GO/DONE 位置 1 之前经过了所需的采集时间。

数据采集在 ACQT<2:0> 位 （ADCON2<5:3>）保持在复位状态 （000）时发生，这一点与不提供可编程采集时间的器件兼容。

如果需要，可设置 ACQTx 位来选择 A/D 模块的可编程采集时间。当 GO/DONE 位置 1 时，A/D 模块继续对输入进行采样，采样时间为所选择的采集时间，然后自动启动转换。由于采集时间已被编程，因此在选择通道和 GO/DONE 位置 1 之间无需另外等待一个采集时间。

在这两种情况下，当转换完成时，GO/DONE位被清零、ADIF 标志位被置 1 且 A/D 再次开始对当前选择的通道进行采样。如果采集时间被编程，那么将不会有任何标志显示采集时间是否结束或转换是否开始。

23.5 选择 A/D 转换时钟

每位的 A/D 转换时间被定义为 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转换时钟源。

TAD 有以下 7 种可能的选择：

- 2 个 TOSC
- 4 个 TOSC
- 8 个 TOSC
- 16 个 TOSC
- 32 个 TOSC
- 64 个 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）必须尽可能小，但它必须大于最小 TAD（欲知更多信息，请参见表 31-28 中的参数 130）。

表 23-1 所示为器件在不同的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

AD 时钟源 （TAD）		最高器件频率
工作状态	ADCS<2:0>	
2 个 TOSC	000	2.50 MHz
4 个 TOSC	100	5.00 MHz
8 个 TOSC	001	10.00 MHz
16 个 TOSC	101	20.00 MHz
32 个 TOSC	010	40.00 MHz
64 个 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： 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 精度可能超出规范允许范围。

23.6 配置模拟端口引脚

ANCON0、 ANCON1、 ANCON2、 TRISA、 TRISF、 TRISG 和 TRISH 寄存器均可控制 A/D 端口引脚的操作。若希望端口引脚为模拟输入，则必须将相应的 TRISx 位置 1 （输入）。如果将 TRISx 位清零 （输出），则转换数字输出电平 （VOH 或 VOL）。

A/D转换操作与CHS<3:0>位以及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 0 （低电平）。配置为数字输入的引脚将转换模拟输入信号。配置为数字输入的引脚上的模拟电平会被精确转换。
2： 任何定义为数字输入的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的限制。

23.7 A/D 转换

图 23-6 所示为 GO/$\overline{\text{DONE}}$ 位置 1 且 ACQT<2:0> 位清零后 A/D 转换器的工作状态。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 所示为 GO/$\overline{\text{DONE}}$ 位置 1、ACQT<2:0> 位被设置为 010、且选择了 4 TAD 的采集时间之后 A/D 转换器的工作状态。

在转换期间将 GO/$\overline{\text{DONE}}$ 位清零将中止当前转换。不会用尚未完成的 A/D 转换结果来更新 A/D 结果寄存器对。这意味着 ADRESH:ADRESL 寄存器将仍保持上一次的转换结果（即上一次写入 ADRESH:ADRESL 寄存器的值）。

在 A/D 转换完成或中止以后，需要等待 2 个 TAD 才能开始下一次采集。等待时间一到，将自动开始对所选通道进行采集。

注：	不应使用启动 A/D 模块的指令将 GO/$\overline{\text{DONE}}$ 位置 1。

图 23-6： A/D 转换 TAD 周期 （ACQT<2:0> = 000， TACQ = 0）

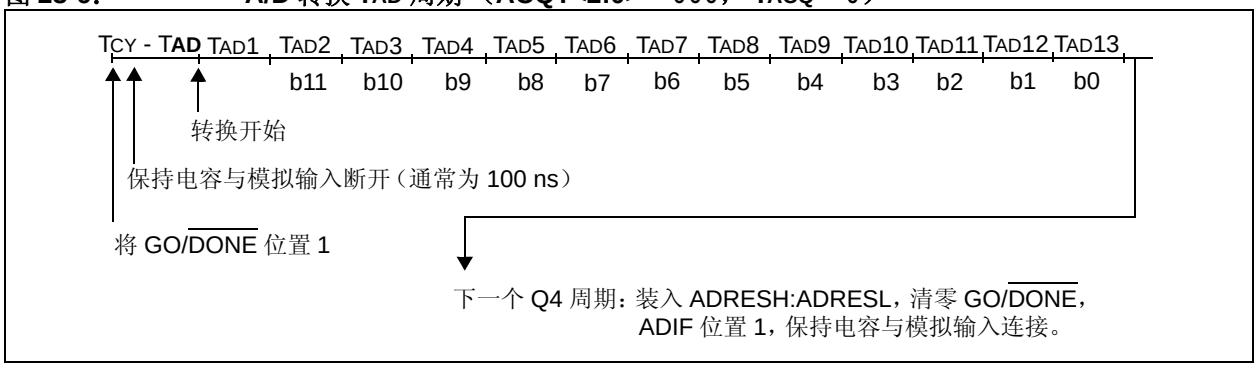
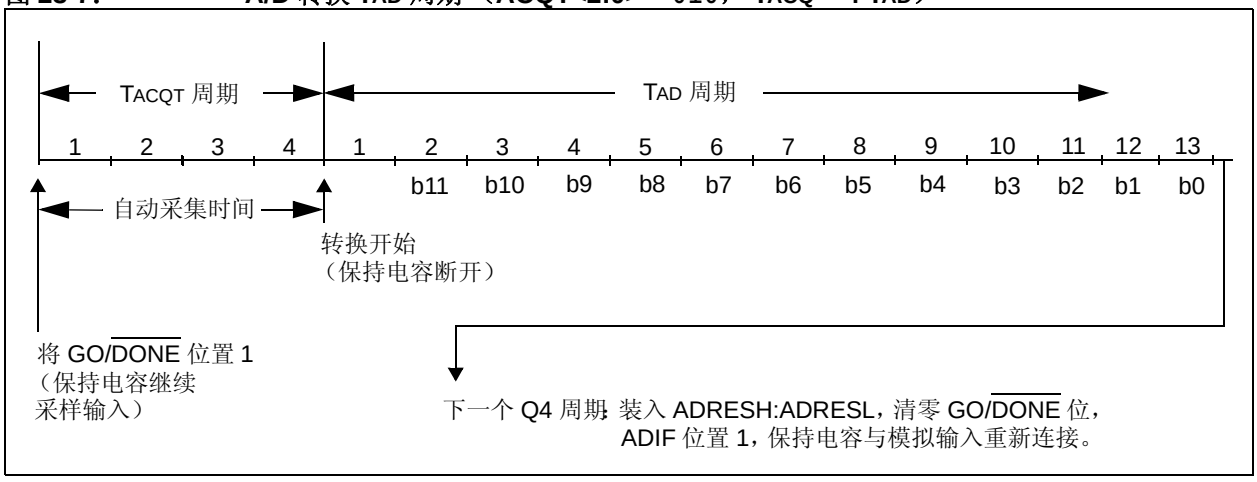


图 23-7： A/D 转换 TAD 周期 （ACQT<2:0> = 010， TACQ = 4 TAD）



© 2011 Microchip Technology Inc.

23.8 特殊事件触发信号的使用

以下模块的特殊事件触发信号可以启动 A/D 转换：

- ECCP2——需要将 CCP2M<3:0> 位（CCP2CON<3:0>）设置为 1011
- CTMU——需要将 CTTRIG 位（CTMUCONH<0>）置 1
- Timer1
- RTCC

要启动 A/D 转换：

- 必须使能 A/D 模块（ADON = 1）
- 选择合适的模拟输入通道
- 可通过以下两种方法之一设定最小采集周期：
  - 用户手动计时
  - 选择适当的 TACQ 时间

当这些条件均满足时，触发信号将 GO/$\overline{\text{DONE}}$ 位置 1，启动 A/D 采集。

如果 A/D 模块未使能（ADON = 0），则特殊事件触发信号对 A/D 模块不产生任何影响。

> **注：** 对于 ECCP2 触发信号，会清零 Timer1 或 Timer3。定时器复位以自动重复 A/D 采集周期，最大限度地降低了软件开销（将 ADRESH:ADRESL 内容移到目标单元）。如果 A/D 模块未使能，则特殊事件触发信号对 A/D 模块不产生任何影响，但会复位定时器的计数值。

23.9 在功耗管理模式下工作

处于功耗管理模式时，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。

如果希望在器件处于功耗管理模式时进行 A/D 转换，ADCON2 中的 ACQT<2:0> 和 ADCS<2:0> 位就应该根据该模式下使用的时钟进行更新。

在进入功耗管理模式之后（某种功耗管理运行模式），就可以启动 A/D 采集或转换。采集或转换启动以后，器件应继续使用功耗管理模式中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。

如果功耗管理模式时钟频率小于 1 MHz，就应该选择 A/D 模块的 RC 时钟源。

在休眠模式下工作需要选择 A/D 模块的 RC 时钟。如果 ACQT<2:0> 位被置为 000 并启动转换，转换将延迟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	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
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	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
ANCON2	ANSEL23	ANSEL22	ANSEL21	ANSEL20	ANSEL19	ANSEL18	ANSEL17	ANSEL16
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0
PORTA	RA7(2)	RA6(2)	RA5	RA4	RA3	RA2	RA1	RA0
TRISA	TRISA7(2)	TRISA6(2)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—
PORTG	—	—	RG5(3)	RG4	RG3	RG2	RG1	RG0
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0
PORTH(1)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0
TRISH(1)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： — = 未实现，读为 0。 A/D 转换不使用阴影单元。

注 1： 此寄存器在 64 引脚器件上未实现。

2： 当 FOSC2 配置位 = 0 时，这些位仅在特定的振荡器模式下可用。若此配置位被清零，则相应的引脚信号将无效。

3： 当禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 被置 1 时，该位未实现。

© 2011 Microchip Technology Inc.

24.0 比较器模块

模拟比较器模块包含三个比较器，可以用多种方式对它们进行单独配置。比较器的输入可以是模拟输入，也可以是两个内部参考电压。数字输出可从引脚电平获取，也可通过控制寄存器读取。此外，还提供了多种输出和产生中断事件的方式。该模块的通用单比较器如图 24-1 所示。

该模块的主要特性包括：

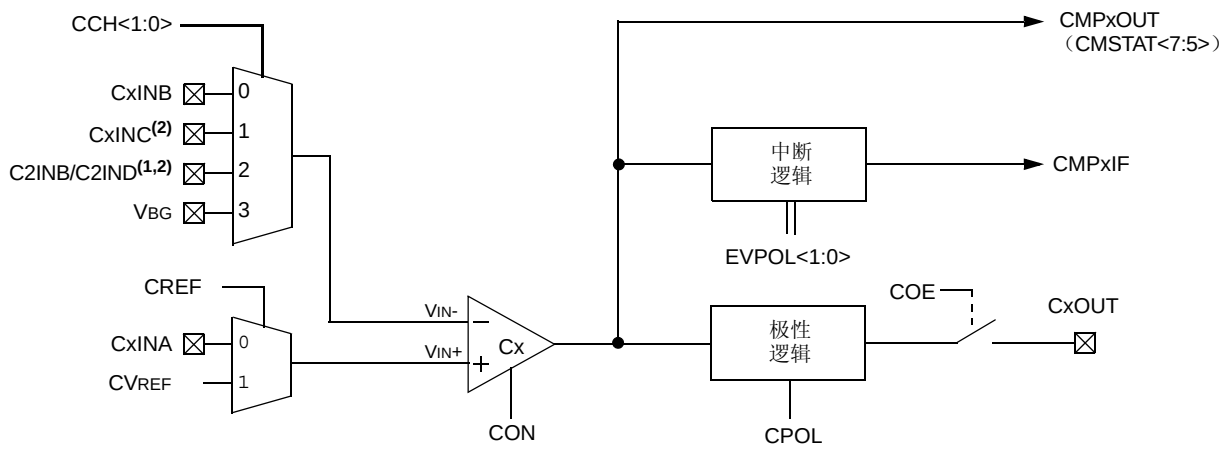
- 独立的比较器控制
- 可编程输入配置
- 输出到引脚和寄存器的电平
- 可编程输出极性
- 采用可配置的电平变化中断，使每个比较器单独产生中断

24.1 寄存器

CMxCON 寄存器（CM1CON、CM2CON 和 CM3CON）选择每个比较器的输入和输出配置以及产生中断的设置（见寄存器 24-1）。

CMSTAT 寄存器（寄存器 24-2）存储比较器的输出结果。该寄存器中的位是只读的。

图 24-1： 比较器简化框图



注 1： 比较器 1 和 3 使用 C2INB 作为反相输入端。比较器 2 使用 C2IND 作为反相输入端。
2： C1INC、 C2INC 和 C2IND 在 64 引脚器件 （PIC18F6XK22）上均不可用。

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-0	R/W-0	R/W-0	R/W-0	R/W-0
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 = 上电复位时的值	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 = 比较器输出到 CxOUT 引脚
0 = 比较器只输出到内部

bit 5 **CPOL：**比较器输出极性选择位
1 = 比较器输出反相
0 = 比较器输出同相

bit 4-3 **EVPOL<1:0>：**中断极性选择位
11 = 输出发生任何变化时产生中断 (1)
10 = 仅在输出由高电平跳变为低电平时产生中断
01 = 仅在输出由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 = 同相输入连接到内部 CVREF 电压
0 = 同相输入连接到 CxINA 引脚

bit 1-0 **CCH<1:0>：**比较器通道选择位
11 = 比较器反相输入连接到 VBG
10 = 比较器反相输入连接到 C2INB 或 C2IND 引脚 (2,3)
01 = 比较器反相输入连接到 CxINC 引脚
00 = 比较器反相输入连接到 CxINB 引脚

注 **1：** 任何时候选择了该模式时， CMPxIF 位都将自动置 1，且必须在初始配置之后由应用程序清零。
2： 比较器 1 和 3 使用 C2INB 作为反相输入端。比较器 2 使用 C2IND 作为反相输入端。
3： C1INC、 C2INC 和 C2IND 在 64 引脚器件 （PIC18F6XK22）上均不可用。

© 2011 Microchip Technology Inc.

寄存器 24-2： CMSTAT：比较器状态寄存器

R-x	R-x	R-x	U-0	U-0	U-0	U-0	U-0
CMP3OUT	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-5 **CMP<3:1>OUT：**比较器 x 状态位

如果 CPOL （CMxCON<5>） = 0 （极性同相）：

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 $V_{IN+} < V_{IN-}$

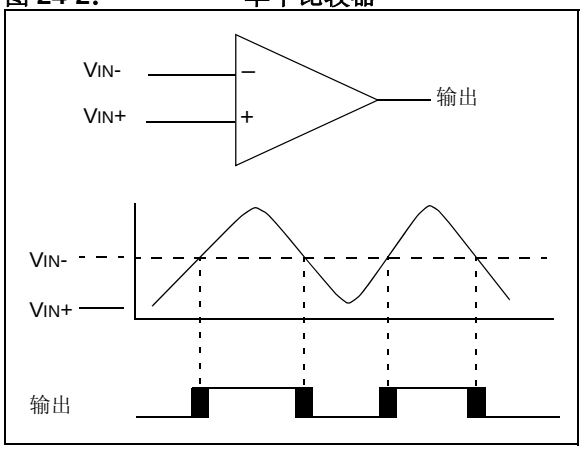
0 = 比较器 x 的 $V_{IN+} > V_{IN-}$

bit 4-0 **未实现：**读为 0

24.2 比较器工作原理

图 24-2 显示了单个比较器，以及其模拟输入电平和数字输出之间的关系。如果 VIN+ 上的模拟输入电平值小于 VIN- 上的模拟输入值，那么比较器将输出数字低电平。如果 VIN+ 上的模拟输入电平值大于 VIN- 上的模拟输入值，那么比较器将输出数字高电平。图 24-2 中比较器输出的阴影部分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： 单个比较器



24.3 比较器响应时间

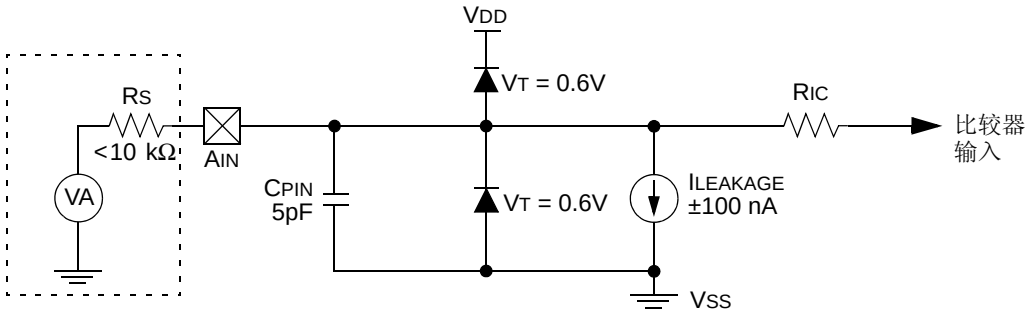
响应时间是指从选定一个新的参考电压或输入源到比较器输出达到一个有效电平的最短时间。比较器的响应时间不同于参考电压的稳定时间。因此，在确定改变比较器输入的总响应时间时，必须同时考虑这两个时间；否则，应使用比较器的最大延时（见**第 31.0 节 “电气特性”**）。

24.4 模拟输入连接注意事项

图 24-3 是一个简化的模拟输入电路。由于模拟引脚和数字输出端相连，因此在 VDD 和 VSS 之间连有反向偏置二极管。这样，即将模拟输入电压限制在 VSS 和 VDD 之间。一旦输入电压超出该范围 0.6V 以上，其中一个二极管就会发生正向偏置从而使输入电压被钳位。

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上的外部元件（如电容或齐纳二极管等），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	=	各个连接点在引脚上产生的泄漏电流
	RIC	=	内部连线等效电阻
	RS	=	信号源阻抗
	VA	=	模拟电压

© 2011 Microchip Technology Inc.

24.5 比较器控制和配置

每个比较器最多有八种可能的输入组合：最多四个外部模拟输入和两个内部参考电压之一。

所有比较器均可选择来自同相通道上的 CxINA 引脚信号或比较器参考电压 （CVREF）。将该信号与反相通道上的 CxINB、CxINC 和 C2INB/C2IND 或单片机的内部固定参考电压（VBG，标称值为 1.024V）作比较。比较器输入和输出与固定 I/O 引脚相连，如表 24-1 所定义。图 24-4 给出了可用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+）	RF6
	C1INB （VIN-）	RF5
	C1INC(1) （VIN-）	RH6
	C2INB （VIN-）	RF3
	C1OUT	RF2
2	C2INA （VIN+）	RF4
	C2INB （VIN-）	RF3
	C2INC(1) （VIN-）	RH4
	C2IND(1) （VIN-）	RH5
	C2OUT	RF1
3	C3INA （VIN+）	RG2
	C3INB （VIN-）	RG3
	C3INC （VIN-）	RG4
	C2INB （VIN-）	RF3
	C3OUT	RG1

注 1： C1INC、 C2INC 和 C2IND 在 64 引脚器件 （PIC18F6XK22）上均不可用。

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 使能比较器操作。清零 CON 位将禁止比较器，以使电流消耗最低。

CMxCON 寄存器的 CCH<1:0> 位 （CMxCON<1:0>）选择将三个模拟输入引脚之一或内部参考电压 （VBG）连接到比较器的 VIN-。根据不同的比较器工作模式，可使用外部或内部参考电压。对于在 64 引脚器件中不可用的外部模拟引脚（C1INC、C2INC 和 C2IND），将其用作输入的相应配置也不可用。

比较 VIN- 上的模拟信号与 VIN+ 上的信号，对比较器的数字输出作相应的调整。

当 CREF （CMxCON<2>） = 0 且 VIN+ 与 CxINA 引脚相连时，使用外部参考电压。使用外部参考电压时，比较器模块可配置为使用外部参考电压源。参考信号必须介于 VSS 和 VDD 之间且可应用于比较器的任一引脚。

比较器模块也可以选择使用内部比较器参考电压（CVREF）模块产生的参考电压。**第 25.0 节 “比较器参考电压模块”** 中将对此模块进行更详细的说明。来自比较器参考电压模块的参考电压仅在 CREF = 1 时可用。在此模式下，内部参考电压施加于比较器的 VIN+ 引脚。

> **注：** 通过 CCH<1:0> 选择的比较器输入引脚，必须通过将 TRISF、TRISG 或 TRISH 位和 ANCONx 寄存器中相应的 ANSELx 位置 1 将其配置为输入引脚。

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<5> 位读取比较器 1 输出，CMSTAT<6> 读取比较器 2 输出，而 CMSTAT<7> 读取比较器 3 输出。这些位是只读位。

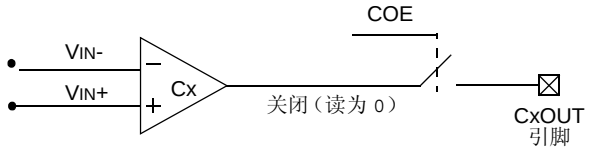
比较器输出也可通过将 COE 位 （CMxCON<6>）置 1 直接输出到 RF2、RF1 和 RG1 I/O 引脚。使能时，引脚输出路径中的多路开关将切换为比较器输出。在此模式下，TRISF<2:1> 和 TRISG<1> 位还可用作 RF2、RF1 和 RG1 引脚的数字输出使能位。

默认情况下，只要 VIN+ 上的电压大于 VIN- 上的电压，比较器的输出即为逻辑高电平。使用 CPOL 位（CMxCON<5>）可翻转比较器的输出极性。

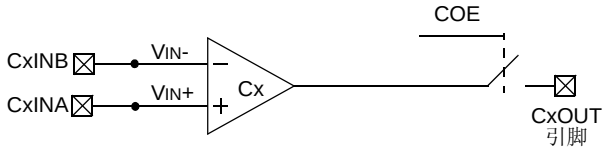
每个比较器的不确定区域大小与规范中给出的输入失调电压和响应时间相关，如**第 24.2 节 “比较器工作原理”** 中所述。

图 24-4： 比较器配置

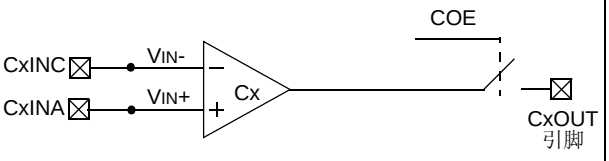
比较器关闭
CON = 0，CREF = x，CCH<1:0> = xx



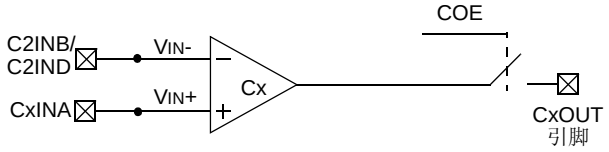
比较器 **CxINB > CxINA**
CON = 1，CREF = 0，CCH<1:0> = 00



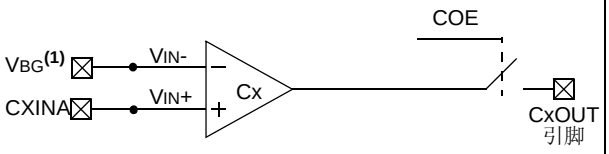
比较器 **CxINC > CxINA**(2,3)
CON = 1，CREF = 0，CCH<1:0> = 01



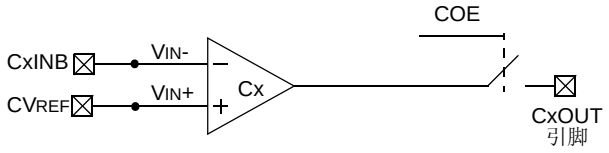
比较器 **CxIND > CxINA**(3)
CON = 1，CREF = 0，CCH<1:0> = 10



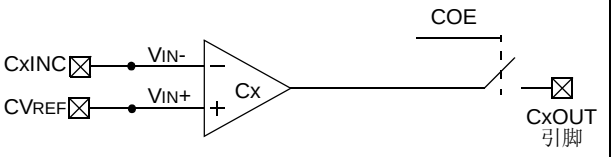
比较器 **VIRV > CxINA**
CON = 1，CREF = 0，CCH<1:0> = 11



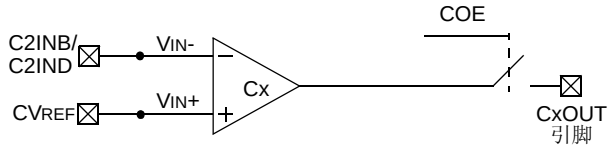
比较器 **CxINB > CVREF**
CON = 1，CREF = 1，CCH<1:0> = 00



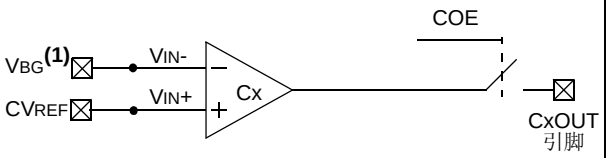
比较器 **CxINC > CVREF**(2,3)
CON = 1，CREF = 1，CCH<1:0> = 01



比较器 **CxIND > CVREF**(3)
CON = 1，CREF = 1，CCH<1:0> = 10



比较器 **VIRV > CVREF**
CON = 1，CREF = 1，CCH<1:0> = 11



注 1： VBG 为内部参考电压 （标称值为 1.024V）。
2： 此配置对于 64 引脚器件 （PIC18F6XK22）上的 CM1CON 不可用。
3： 此配置对于 64 引脚器件 （PIC18F6XK22）上的 CM2CON 不可用。

© 2011 Microchip Technology Inc.

24.6 比较器中断

任何时候发生以下任一情况时，就会将比较器的中断标志位置 1：

- 比较器输出由低电平跳变为高电平
- 比较器输出由高电平跳变为低电平
- 比较器输出发生任何变化

通过设置 CMxCON 寄存器中的 EVPOL<1:0> 位（CMxCON<4:3>）完成比较器中断选择。

为了提供最大灵活性，可使用 CMxCON 寄存器中的 CPOL 位（CMxCON<5>）翻转比较器的输出。这在功能上等效于对于特定模式交换比较器的同相输入和反相输入。

当比较器的输出由低电平跳变为高电平或高电平跳变为低电平时产生中断。这种中断产生的模式依赖于 CMxCON 寄存器中的 EVPOL<1:0>。当 EVPOL<1:0> = 01 或 10 时，在比较器的输出由低电平跳变为高电平或高电平跳变为低电平时产生中断。一旦产生中断，需要由软件清除中断标志位。

当 EVPOL<1:0> = 11 时，任何一个比较器的输出电平一旦发生了变化，就会将比较器的中断标志位置 1。为确定实际发生的变化，需要用软件来保存输出位的状态信息（从 CMSTAT<7:5> 中读取数据）。

CMPxIF 位（PIR6<2:0>）是比较器中断标志位。CMPxIF 位必须通过清零来复位。因为可以向该寄存器写入 1，所以也可以模拟中断产生。表 24-2 为对应于不同比较器输入电压和 EVPOL 位设置时的中断产生情况。

必须将 CMPxIE 位（PIE6<2:0>）和 PEIE 位（INTCON<6>）置 1 以允许中断。此外，还必须将 GIE 位（INTCON<7>）置 1。只要这些位中的任何一位被清零，虽然当有中断条件产生时 CMPxIF 位仍会置 1，但却不允许产生中断。

图 24-3 为中断部分的简化框图。

> **注：** 当 EVPOL<1:0> = 00 时，CMPxIF 将不会被置 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-	低电平跳变为高电平	无
		VIN+ < VIN-	高电平跳变为低电平	无
	01	VIN+ > VIN-	低电平跳变为高电平	有
		VIN+ < VIN-	高电平跳变为低电平	无
	10	VIN+ > VIN-	低电平跳变为高电平	无
		VIN+ < VIN-	高电平跳变为低电平	有
	11	VIN+ > VIN-	低电平跳变为高电平	有
		VIN+ < VIN-	高电平跳变为低电平	有
1	00	VIN+ > VIN-	高电平跳变为低电平	无
		VIN+ < VIN-	低电平跳变为高电平	无
	01	VIN+ > VIN-	高电平跳变为低电平	无
		VIN+ < VIN-	低电平跳变为高电平	有
	10	VIN+ > VIN-	高电平跳变为低电平	有
		VIN+ < VIN-	低电平跳变为高电平	无
	11	VIN+ > VIN-	高电平跳变为低电平	有
		VIN+ < VIN-	低电平跳变为高电平	有

© 2011 Microchip Technology Inc.

24.7 休眠模式下的比较器工作原理

当比较器处于运行模式而器件处于休眠模式时，比较器仍保持工作并可产生中断 （如果允许中断）。当允许中断时，中断会把器件从休眠模式唤醒。每个比较器工作时都会消耗额外的电流。

若要把休眠模式下的功耗减少到最小，可在进入休眠模式前关闭比较器 （CON = 0）。当器件从休眠模式唤醒时， 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 寄存器进入复位状态，导致比较器模块和参考电压处于关闭状态。

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
PIR6	—	—	—	EEIF	—	CMP3IF	CMP2IF	CMP1IF
PIE6	—	—	—	EEIE	—	CMP3IE	CMP2IE	CMP1IE
IPR6	—	—	—	EEIP	—	CMP3IP	CMP2IP	CMP1IP
CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CM3CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
CVRCON	CVREN	CVROE	CVRSS	CVR4	CVR3	CVR2	CVR1	CVR0
CMSTAT	CMP3OUT	CMP2OUT	CMP1OUT	—	—	—	—	—
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—
PORTG	—	—	RG5(1)	RG4	RG3	RG2	RG1	RG0
LATG	—	—	—	LATG4	LATG3	LATG2	LATG1	LATG0
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0
PORTH	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0
LATH	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0
TRISH	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
ANCON1	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
ANCON2	ANSEL23	ANSEL22	ANSEL21	ANSEL20	ANSEL19	ANSEL18	ANSEL17	ANSEL16
PMD0	CCP3MD	CCP2MD	CCP1MD	UART2MD	UART1MD	SSP2MD	SSP1MD	ADCMD

图注： — = 未实现位，读为 0。

注 1： 禁止主复位 （MCLRE = 0）时，该位可用。当 MCLRE 被置 1 时，该位未实现。

© 2011 Microchip Technology Inc.

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 显示了此模块的框图。梯形电阻经过分段可提供多个 CVREF 值，并具有断电功能，可以在不使用参考电压时节省功耗。此模块的参考电源可由器件 VDD/VSS 或外部参考电压提供。

25.1 配置比较器参考电压

比较器参考电压源模块由 CVRCON 寄存器（寄存器 25-1）控制。比较器参考电压模块提供 32 种输出电压。

CVR<4:0> 选择位 （CVRCON<4:0>）用于选择输出电压范围。公式 25-1 显示了比较器参考电压的计算方法。

公式 25-1：

当 CVRSS = 1 时：

$$CVREF = (VREF-) + (CVR<4:0>/32) \cdot (VREF+ - VREF-)$$

当 CVRSS = 0 时：

$$CVREF = (AVSS) + (CVR<4:0>/32) \cdot (AVDD - AVSS)$$

比较器参考电压模块的电源可来自 VDD 和 VSS，也可以来自与 RA3 和 RA2 复用的外部 VREF+ 和 VREF-。电压源由 CVRSS 位 （CVRCON<5>）选择。

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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

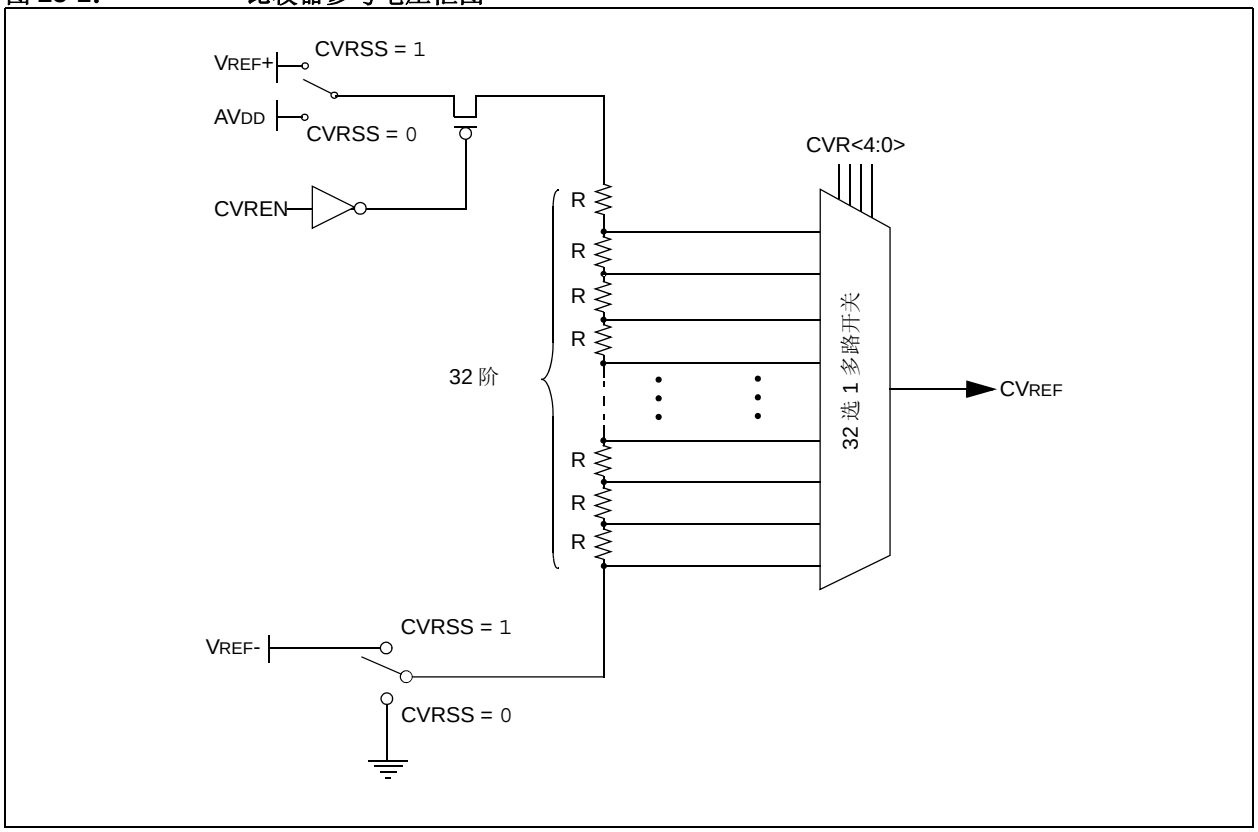
bit 7 **CVREN：**比较器参考电压使能位
1 = CVREF 电路上电
0 = CVREF 电路断电

bit 6 **CVROE：**比较器 VREF 输出使能位
1 = CVREF 电压是 CVREF 引脚上的输出
0 = CVREF 电压与 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： 比较器参考电压框图



25.2 比较器参考电压精度 / 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 无法达到参考电压源的上下限。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。测试到的参考电压的绝对精度请参见**第 31.0 节 “电气特性”**。

25.3 在休眠模式下工作

如果器件因中断或看门狗定时器超时从休眠模式唤醒，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

器件复位通过将 CVREN （CVRCON<7>）位清零从而禁止比较器参考电压模块。此复位功能还通过将 CVROE （CVRCON<6>）位清零，使参考电压与 RF5 引脚断开。

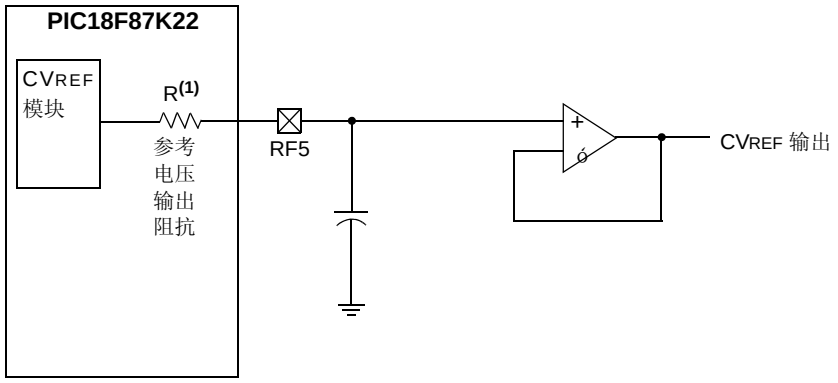
25.5 连接注意事项

比较器参考电压模块的工作独立于比较器模块。如果 CVROE 位置 1，那么参考电压发生器的输出可能与 RF5 引脚相连。当 RF5 配置为数字输入时，将参考电压输出连接到 RF5 引脚上将增加电流消耗。使能 CVRSS 时，将 RF5 用作数字输出引脚也将增加电流消耗。

RF5 引脚可被直接用作 D/A 输出，但是其驱动能力有限。由于电流驱动能力有限，参考电压输出端必须外接一个缓冲器到 VREF。图 25-2 举例说明了这一缓冲技术。

© 2011 Microchip Technology Inc.

图 25-2： 比较器参考电压输出缓冲器示例



注 1： R 的值取决于比较器参考电压配置位 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
CM3CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
ANCON1	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8

图注： — = 未实现，读为 0。比较器参考电压不使用阴影单元。

注：

© 2011 Microchip Technology Inc.

26.0 高 / 低电压检测 （HLVD）

PIC18F87K22 系列器件具有一个高 / 低电压检测模块（HLVD）。该模块是一个可编程电路，用于设置器件电压跳变点以及电压在该点处发生改变的方向。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，中断标志位就会置 1。如果允许中断，程序将跳转到中断向量地址处执行，并由软件响应该中断。

高 / 低电压检测控制寄存器 （寄存器 26-1）完全控制 HLVD 模块的运行。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 所示为模块的框图。

寄存器 26-1： HLVDCON：高 / 低电压检测控制寄存器

R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-1	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **VDIRMAG：**电压方向大小选择位
1 = 当电压等于或高于跳变点 （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 = 表示电压检测逻辑将在指定的电压范围内产生中断标志。
0 = 表示电压检测逻辑在指定的电压范围内不产生中断标志且不允许 HLVD 中断

bit 4 **HLVDEN：**高 / 低电压检测电源使能位
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 （输入信号来自 HLVDIN 引脚）
1110 = 最大设置
.
.
.
0000 = 最小设置

注 1： 有关电气规范，请参阅参数 D420。

通过将 HLVDEN 位 （HLVDCON<4>）置 1 使能该模块。每次使能 HLVD 模块时，电路都需要一些时间才能稳定下来。 IRVST 位 （HLVDCON<5>）是用于指示电路何时稳定的只读位。只有在电路稳定且 IRVST 置 1 的情况下，该模块才能产生中断。

VDIRMAG 位 （HLVDCON<7>）决定模块的整体工作状态。VDIRMAG 清零时，模块监视 VDD 看它是否降到预先确定的设定点以下。当该位置 1 时，模块监视 VDD 看它是否上升到设定点以上。

26.1 工作原理

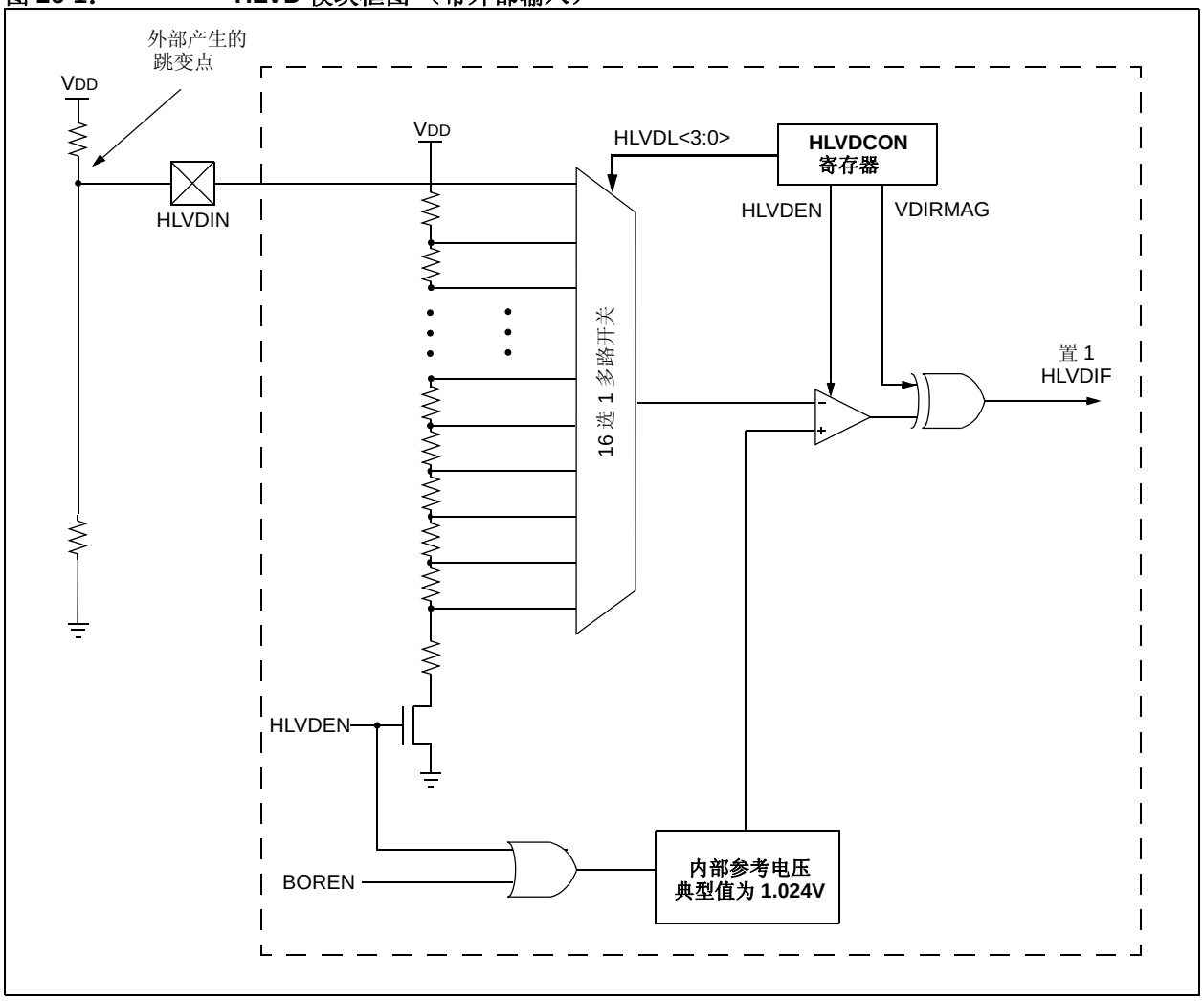
使能 HLVD 模块时，比较器使用内部生成的参考电压作为设定点。该设定点与跳变点相比较，其中电阻分压器中的每个节点均代表一个电压跳变点。“跳变点”电压是器件检测到的高压或低压事件的电压值，它取决于该模块的配置。

当电源电压与跳变点电压相等时，电阻阵列的节点电压输出值等于由参考电压模块产生的内部参考电压。然后比较器通过将 HLVDIF 位置 1，产生一个中断信号。

跳变点电压是软件可编程的任意 16 位值。通过编程 HLVDL<3:0> 位 （HLVDCON<3:0>）选择跳变点。

HLVD 模块还允许用户通过外部电源向模块提供跳变电压。当 HLVDL<3:0> 位设置为 1111 时，使能该模块。在这种情况下，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 / 低电压检测中断，使之可以在有效的工作电压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）



© 2011 Microchip Technology Inc.

26.2 HLVD 设置

要设置 HLVD 模块：

1. 通过将值写入 HLVDL<3:0> 位选择期望的 HLVD 跳变点。
2. 将 VDIRMAG 位置 1 以检测高电压（VDIRMAG= 1）或低电压（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>），该位可能已由先前的中断置为 1。
5. 如果希望产生中断，可通过将 HLVDIE 和 GIE 位（分别为 PIE2<2> 和 INTCON<7>）置 1 允许 HLVD 中断。

   在 IRVST 位置 1 前，不会产生中断。

注：	在改变模块设置（VDIRMAG、HLVDL<3:0>）之前，首先禁止该模块（HLVDEN = 0），然后再更改设置并重新使能模块。这将阻止错误的 HLVD 事件发生。

26.3 电流消耗

使能该模块便同时使能了 HLVD 比较器和分压器，并消耗静态电流。在电气规范参数 D022B（表 31-13）中指定了使能该模块时的总电流消耗。

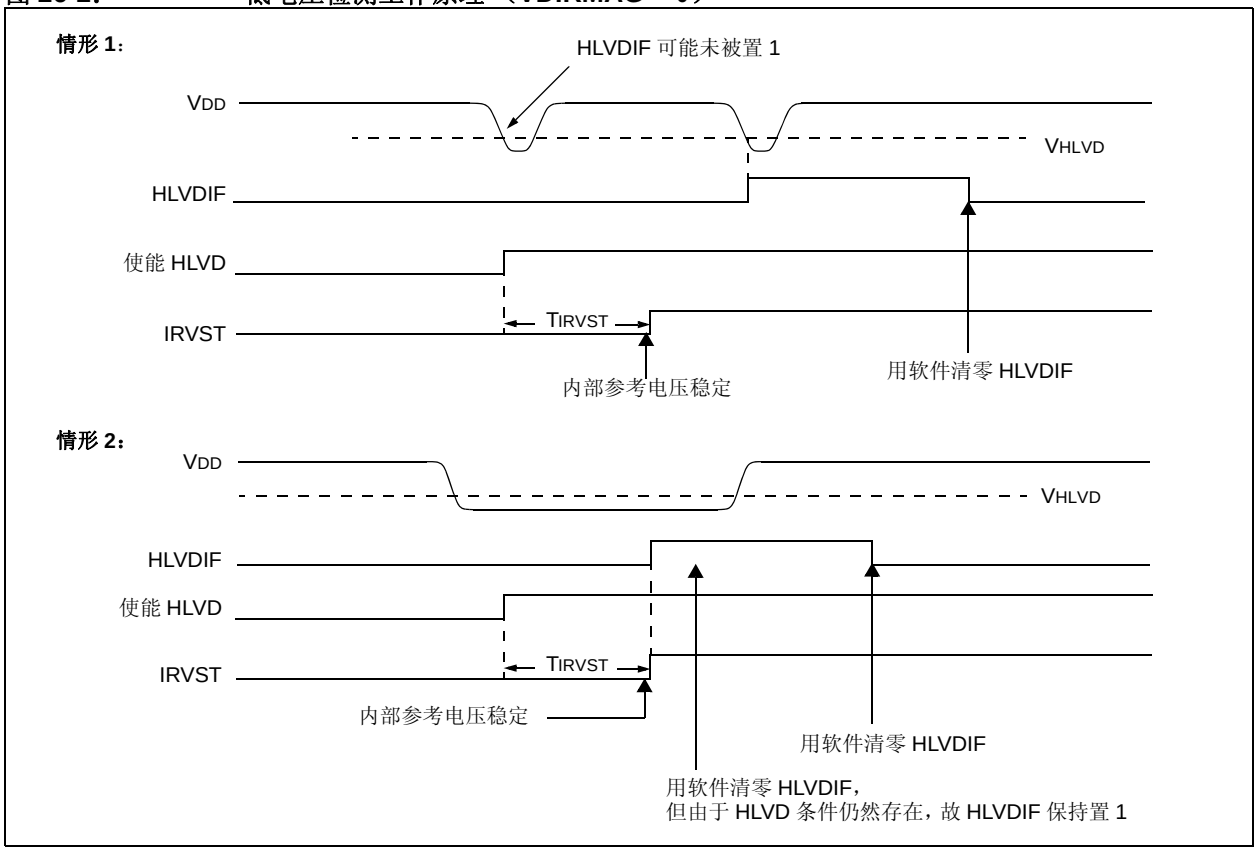
HLVD 模块无需一直工作，工作与否取决于具体的应用。要降低电流消耗，只需要在检测电压时，短时间地使能HLVD电路，在检测完成之后马上禁止HLVD模块。

26.4 HLVD 启动时间

在电气规范参数 37 （**第 31.0 节 “电气特性”**）中指定了 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。如果使用参考电压的HLVD或其他电路被禁止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电压或高电压条件。启动时间 TIRVST 是与器件时钟速度无关的一段时间间隔。它在电气规范参数 37 （表 31-13）中指定。

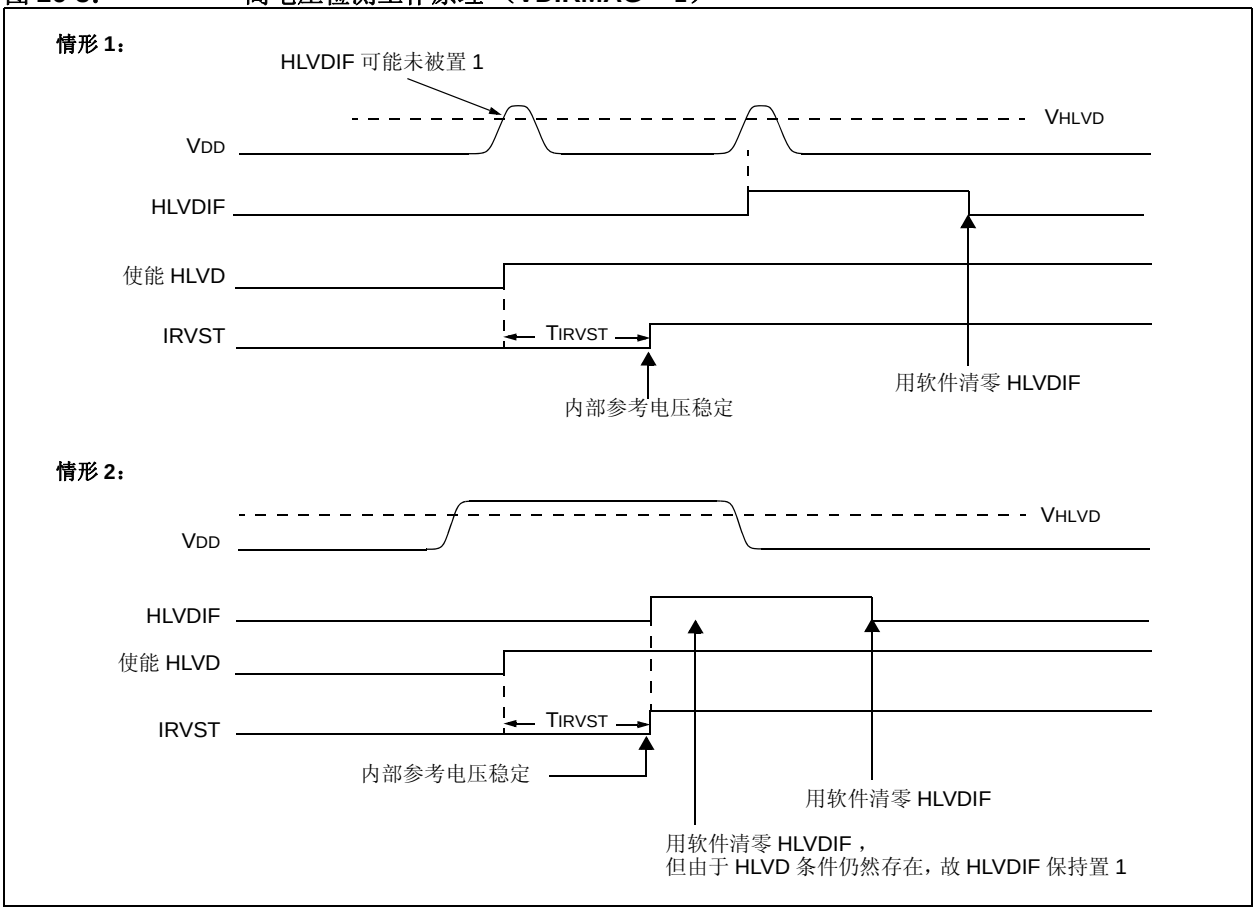
直到TIRVST结束并且参考电压达到稳定后才会将HLVD中断标志置 1。基于此原因，在此时间间隔期间，超出设定点的短暂偏离可能不会被检测到 （见图 26-2 或图 26-3）。

图 26-2： 低电压检测工作原理 （VDIRMAG = 0）



© 2011 Microchip Technology Inc.

图 26-3： 高电压检测工作原理 （VDIRMAG = 1）

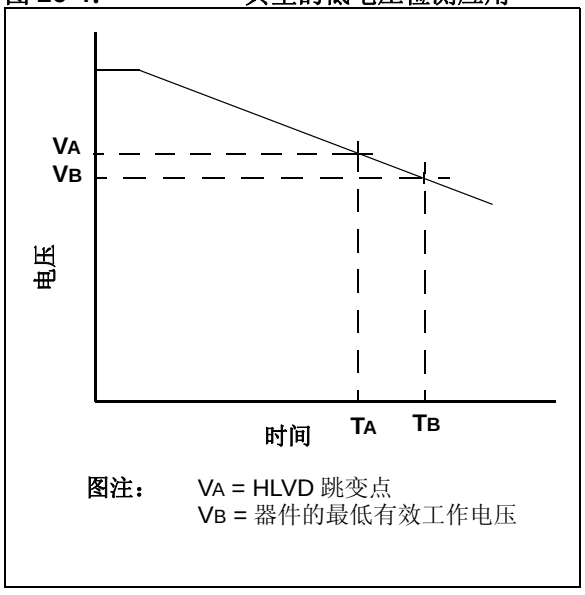


26.5 应用

在很多应用中，系统希望可以检测到电压低于或高于某个阈值的事件。例如，可以定期使能 HLVD 模块来检测是否连接了通用串行总线 （Universal Serial Bus，USB）。假定在不连接 USB 的情况下器件的供电电压低于 USB 电压。若连接了 USB 将检测到 3.3V 到 5V 的高压 （USB 上的电压），如果断开连接，情况正好相反。该特性可以在设计中省去一些额外的元件和连接信号（输入引脚）。

对于一般的电池应用，图 26-4 显示了可能的电压曲线。器件电压会随时间逐渐下降。当器件电压达到电压 VA 时，HLVD 逻辑会在 TA 时产生中断。该中断将导致执行中断服务程序 （ISR），从而使应用程序能在器件电压退出有效工作范围 （对应的时间为 TB）之前执行 “日常任务”，随后安全关闭。因此， HLVD 将会提供一个时间窗（表示为 TA 和 TB 的时间差）使应用程序能安全地退出。

图 26-4： 典型的低电压检测应用



26.6 休眠期间的工作原理

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进入复位状态。这将强制 HLVD 模块关闭。

表 26-1： 与高 / 低电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	—	SSP2IF	BLC2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	—	SSP2IE	BLC2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
IPR2	OSCFIP	—	SSP2IP	BLC2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
TRISA	TRISA7(1)	TRISA6(1)	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
ANCON0	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0

图注： — = 未实现，读为 0。 HLVD 模块不使用阴影单元。

注 1： 可以根据不同的主振荡器模式将 PORTA<7:6> 配置为端口引脚并设置其方向位。禁止时，这些位读为 0。

© 2011 Microchip Technology Inc.

27.0 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。CTMU 可与其他片上模拟模块一起，用于精确地测量时间、电容以及电容的相对变化或生成具有特定延时的输出脉冲。CTMU 模块是与电容传感器接口的理想选择。

该模块的主要功能包括：

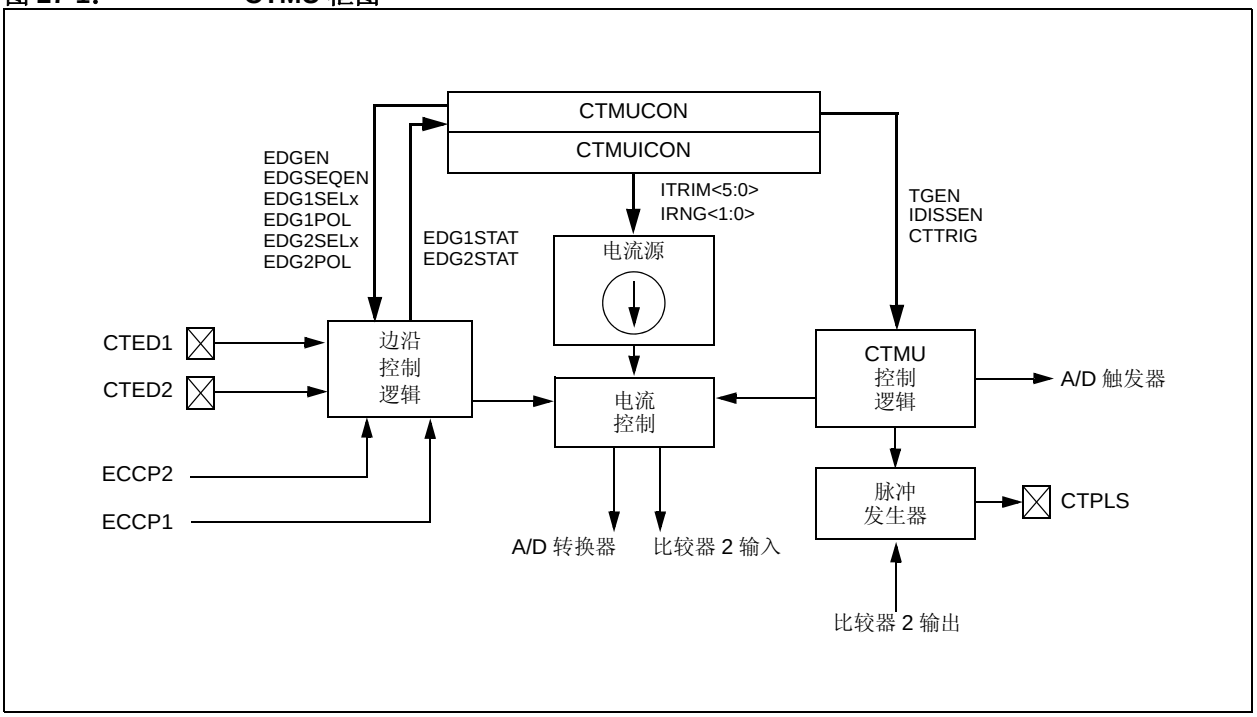
- 最多 24 路通道，可用于电容或时间测量输入
- 片上精确电流源
- 4 个边沿输入触发源
- 每个边沿源的极性控制
- 边沿序列控制
- 控制对边沿的响应
- 1 ns 的时间测量分辨率
- 高精度时间测量
- 与系统时钟异步的外部或内部信号的延时
- 适合测量电容的精确电流源

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 24 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可从 4 个源中选择：两个外部输入或 ECCP1/ECCP2 特殊事件触发信号。

CTMU 特殊事件可触发模数转换器模块。

图 27-1 为 CTMU 的框图。

图 27-1： CTMU 框图



27.1 CTMU 寄存器

CTMU 的控制寄存器为：

- CTMUCONH
- CTMUCONL
- CTMUICON

CTMUCONH 和 CTMUCONL 寄存器（寄存器 27-1 和寄存器 27-2）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、边沿序列、A/D 触发器以及模拟电路电容放电和使能。CTMUICON 寄存器（寄存器 27-3）包含用于选择电流源范围和电流源微调的位。

寄存器 27-1： CTMUCONH：CTMU 控制寄存器高字节

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 = 上电复位时的值	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 = 模块在空闲模式下继续工作

bit 4 **TGEN：**时间生成使能位
1 = 使能边沿延时生成
0 = 禁止边沿延时生成

bit 3 **EDGEN：**边沿使能位
1 = 不阻止边沿
0 = 阻止边沿

bit 2 **EDGSEQEN：**边沿序列使能位
1 = 边沿 1 事件必须在边沿 2 事件之前发生
0 = 无需边沿序列

bit 1 **IDISSEN：**模拟电流源控制位
1 = 模拟电流源输出接地
0 = 模拟电流源输出不接地

bit 0 **CTTRIG：**触发器控制位
1 = 使能触发器输出
0 = 禁止触发器输出

© 2011 Microchip Technology Inc.

寄存器 27-2： CTMUCONL：CTMU 控制寄存器低字节

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **EDG2POL：**边沿 2 极性选择位
1 = 边沿 2 编程为正边沿响应
0 = 边沿 2 编程为负边沿响应

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 = ECCP2 特殊事件触发信号

bit 4 **EDG1POL：**边沿 1 极性选择位
1 = 边沿 1 编程为正边沿响应
0 = 边沿 1 编程为负边沿响应

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 = ECCP2 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 27-3： 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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-2 **ITRIM<5:0>：**电流源微调位

`011111` = 标称电流的最大正向调整
`011110`
.
.
.
`000001` = 标称电流的最小正向调整
`000000` = IRNG<1:0> 指定的标称电流输出
`111111` = 标称电流的最小负向调整
.
.
.
`100010`
`100001` = 标称电流的最大负向调整

bit 1-0 **IRNG<1:0>：**电流源范围选择位

`11` = 100 倍基本电流
`10` = 10 倍基本电流
`01` = 基本电流 （标称值为 0.55 µA）
`00` = 禁止电流源

© 2011 Microchip Technology Inc.

27.2 CTMU 工作原理

CTMU 的工作方式是使用固定电流源对电路进行充电。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。这样，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。

27.2.1 工作原理

CTMU 的工作原理基于以下电荷公式：

$$C = I \cdot \frac{dV}{dT}$$

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）。电荷也可定义为：以法拉为单位的电容（C）乘以电路的电压（V）。可得：

$$I \cdot t = C \cdot V$$

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，当电路中的电容固定且已知时，利用以下关系求得时间：

$$t = (C \cdot V)/I$$

当电流源施加于电路的时间固定时，利用以下关系求得电容：

$$C = (I \cdot t)/V$$

27.2.2 电流源

CTMU 的核心是高精度电流源，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平中选择电流级别，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 00 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。请注意，其中一半的范围用于增大电流，另一半用于降低电流。值 000000 是中间位置（无变化）。值 100000 代表最大负调整（约 -62%），011111 代表最大正调整（约 +62%）。

27.2.3 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 特殊事件触发信号的输入脉冲。输入通道是电平敏感通道，响应通道中的瞬时电平，而不是电平跳变。输入使用 EDG1SEL 和 EDG2SEL 位对（CTMUCONL<3:2, 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序列（边沿1在边沿2之前发生），方法是将EDGSEQEN 位（CTMUCONH<2>）置 1。

27.2.4 边沿状态

CTMUCON 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，如果通道的配置及其电流状态发生改变，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。也就是说可以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。

27.2.5 中断

每当电流源先使能，然后禁止时，CTMU 就会将其中断标志（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。如果未使能边沿序列（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。

27.3 CTMU 模块初始化

以下序列是用于初始化 CTMU 模块的通用方法：

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. 通过分别设置 EDG1SEL 和 EDG2SEL 位（CTMUCONL<3:2> 和 <6:5>）配置边沿 1 和边沿 2 的边沿输入源。
4. 使用 EDG2POL 和 EDG1POL 位（CTMUCONL<7,4>）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>）使能边沿序列。

   默认情况下，将禁止边沿序列。
6. 使用 TGEN 位选择工作模式（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，会禁止转换触发器。
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。
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>）。
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，以与 CTMU 模块配合使用：

- 边沿源生成：除了外部边沿输入引脚之外， CCPx 特殊事件触发信号也可以用作 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。

27.4 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。精度较低的应用示例为电容触摸开关，在该应用中，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。

27.4.1 电流源校准

CTMU 模块随附的电流源具有三种电平范围，其中每种范围都可以在其标称值 ±60% 的范围内进行调节。因此，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 27-2 给出了示例电路。

电流源测量使用以下步骤执行：

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. 产生稳定延时。
5. 执行 A/D 转换。
6. 使用 $I = V/R_{CAL}$ 计算电流源电流，其中， R_{CAL} 是高精度电阻， V 通过执行 A/D 转换来测量。

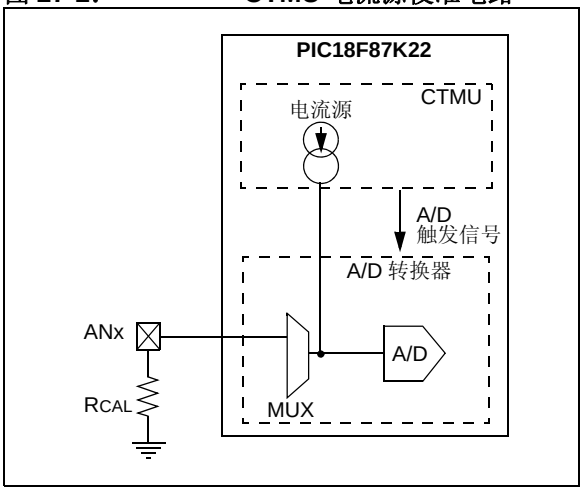
© 2011 Microchip Technology Inc.

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 的值，必须选择标称电流，然后就可以计算电阻。

例如，如果 A/D 转换器参考电压为 3.3V，使用满量程的 70%（或 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，所需的电阻值使用 RCAL = 2.31V/0.55 µA 计算，得到值为 4.2 MΩ。类似地，如果电流源选择为 5.5 µA，RCAL 将为 420,000Ω；如果电流源设置为 55 µA，则为 42,000Ω。

图 27-2： 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。请记住，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 27-1 显示如何初始化 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 27-2 给出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，这么做是为了逐步演示整个过程。也可以通过将 CTMU 的 CTTRIG 位（CTMUCONH<0>）置 1 来自动触发转换。

例 27-1： CTMU 校准程序

```
#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
    // ANCON0
    ANCON0 = 0X04;
    // ANCON1
    ANCON1 = 0XE0;

    // ADCON2
    ADCON2bits.ADFM=1;            // Resulst format 1= Right justified
    ADCON2bits.ACQT=1;            // Acquition time 7 = 20TAD 2 = 4TAD 1=2TAD
    ADCON2bits.ADCS=2;            // Clock conversion bits 6= FOSC/64 2=FOSC/32

    // ADCON0
    ADCON1bits.VCFG0 =0;          // Vref+ = AVdd
    ADCON0bits.VCFG1 =0;          // Vref+ = AVdd
    ADCON0bits.VCFG = 0;          // Vref- = AVss
    ADCON0bits.CHS=2;             // Select ADC channel

    ADCON0bits.ADON=1;            // Turn on ADC

}
```


© 2011 Microchip Technology Inc.

例 27-2： 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

}
```

27.4.2 电容校准

内部 A/D 转换器采样电容和电路板走线与焊垫的杂散电容虽然容值较小但仍会影响电容测量的精度。在确保先取下期望测量的电容的情况下，可以对杂散电容进行测量。

取下要测量的电容后，执行以下步骤：

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：

$$C_{OFFSET} = C_{STRAY} + C_{AD} = (I \cdot t)/V$$

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 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% （2.31V），那么 t 为：

$$(4\ pF + 11\ pF) \cdot 2.31V/0.55\ \mu A$$

或 63 µs。

关于 CTMU 电容校准的典型程序，请参见例 27-3。

© 2011 Microchip Technology Inc.

例 27-3： 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;
}
```

27.5 使用 CTMU 测量电容

使用 CTMU 测量电容有两种方法。绝对方法测量实际电容值。相对方法只需要测量电容的变化量。

27.5.1 绝对电容测量

对于绝对电容测量，应遵循**第 27.4 节 “校准 CTMU 模块”**中的电流和电容校准步骤。

要执行电容测量，使用以下步骤：

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 $C_{TOTAL} = (I * T)/V$，其中：
   - I 从电流源测量步骤获知 （**第 27.4.1 节 “电流源校准”**）
   - T 是固定延时
   - V 通过执行 A/D 转换来测量
8. 从 C_{TOTAL} 中减去杂散电容和 A/D 采样电容（C_{OFFSET} 来自**第 27.4.2 节 “电容校准”**）确定被测电容的值。

27.5.2 相对电荷测量

有些应用可能并不需要精确的电容测量。例如，在检测基于电容的开关的有效按压时，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 转换器以及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。

使用 CTMU 检测电容变化可以使用以下步骤轻松实现：

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电容的变化。请注意，在这种情况下，不需要对电流源或电路电容测量进行校准。（例 27-4 给出了电容触摸开关的软件程序示例。）

© 2011 Microchip Technology Inc.

例 27-4： 用于电容触摸开关的程序

```
#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;
    }
}
```

27.6 使用 CTMU 模块测量时间

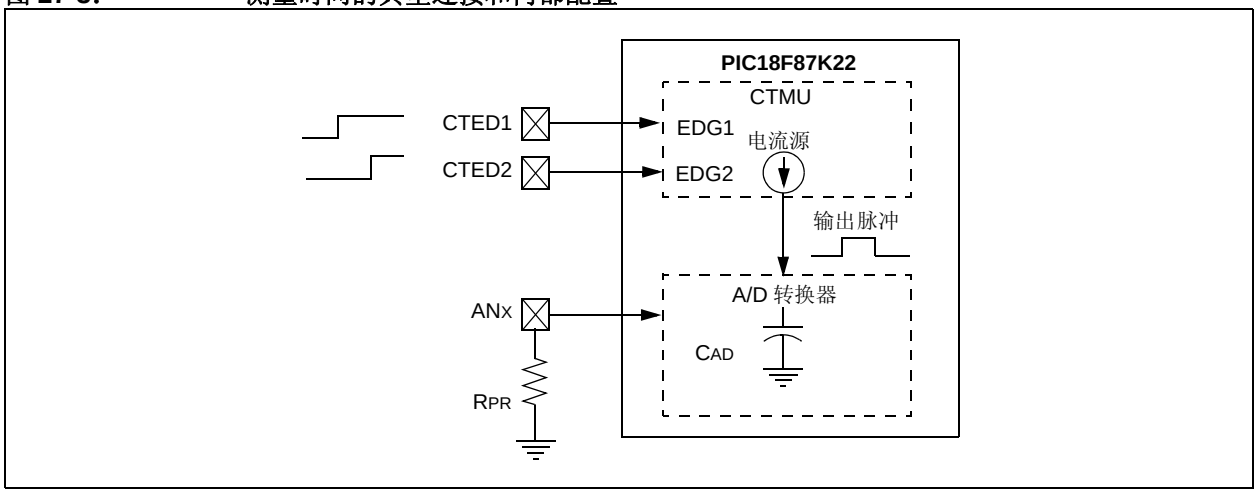
通过电流和电容校准步骤测量比率 （C/I）之后，可以精确测量时间。采用以下步骤：

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 27.4.1 节 “电流源校准”）中计算
   - C 在电容校准步骤 （第 27.4.2 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 COFFSET 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 通道选择寄存器 （AD1CHS）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。这可以最大程度降低所增加的杂散电容，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 27-3： 测量时间的典型连接和内部配置



© 2011 Microchip Technology Inc.

27.7 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。当使用外部电压时，这通过使用 CTDIN 输入引脚作为脉冲延时的触发器来实现。当使用外部电容时，这通过使用内部比较器参考电压模块和比较器 2 输入引脚实现。脉冲输出到CTPLS引脚。要使能该模式，需将TGEN位置1。

图 27-4 给出了示例电路。当 CTMUDS（ODCON3<0>）清零时，脉冲延时由比较器 2 的输出决定，而当该位置 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 27.4.1 节 “电流源校准”**）获得，V 是内部参考电压 （CVREF）。

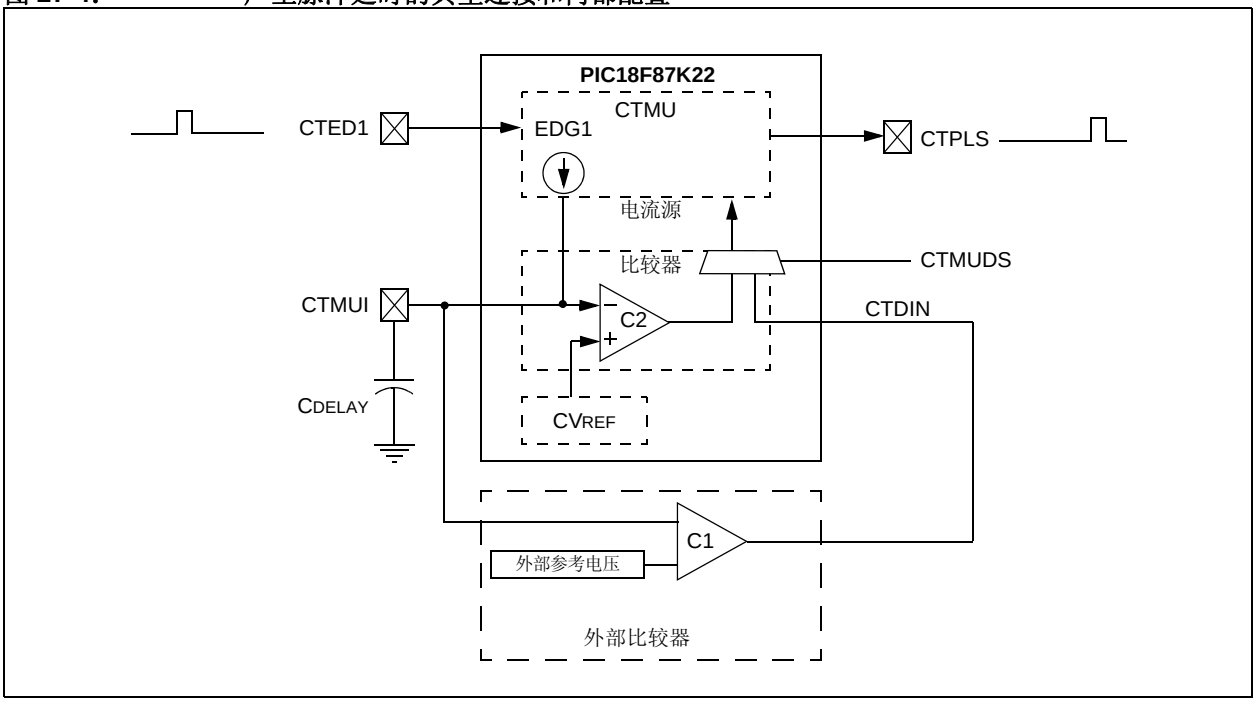
连接电容可变的传感器，（例如湿度传感器）是使用外部电容的一个示例。当湿度发生变化时，CTPLS 上的输出脉宽也会变化。与数字传感器连接是使用 CTDIN 引脚的一个示例。CTPLS输出引脚可以连接到输入捕捉引脚，通过测量变化的脉冲宽度来确定应用的传感器输出。

执行以下步骤来使用该功能：

1. 如果 CTMUDS 清零，初始化比较器 2。
2. 如果 CTMUDS 置 1，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。
4. 将 EDG1STAT 置 1。

当 CTMUDS 清零时，一旦 CDELAY 充电到参考电压跳变点的值时，在 CTPLS 上就会产生输出脉冲。当 CTMUDS 置 1 时，一旦 CTDIN 置 1，在 CTPLS 上就会产生输出脉冲。

图 27-4： 产生脉冲延时的典型连接和内部配置



27.8 使用 CTMU 模块测量温度

CTMU 和内部二极管一起可用于测量温度。ADC 可与内部二极管连接，且 CTMU 模块可作为二极管的电流源。

ADC 读数将反映温度。随着温度的升高， ADC 读数将变小。这可用于低成本的温度测量应用。

例 27-5： 使用内部二极管测量温度的程序

```
// 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： 温度二极管未校准或标准化；用户必须在应用中校准二极管。

© 2011 Microchip Technology Inc.

27.9 休眠 / 空闲模式期间的操作

27.9.1 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。

27.9.2 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。如果调用空闲模式时，模块正在执行操作，这种情况下，结果将类似于休眠模式的结果。

27.10 复位对 CTMU 的影响

在复位时， CTMU 的所有寄存器都会被清零。这使 CTMU 模块处于禁止状态，它的电流源被关闭，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。电路放电的方法是在 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 27-1： 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	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP

图注： — = 未实现，读为 0。 ECCP 操作期间不使用阴影单元。

注：

© 2011 Microchip Technology Inc.

28.0 CPU 的特殊功能

PIC18F87K22 系列器件具有几项特殊的功能，旨在最大限度地提高系统可靠性，并通过减少外部元件把成本降到最低。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 单元
- 在线串行编程 （ICSP™）

根据具体应用对频率、功耗、精度和成本的要求来配置振荡器。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经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， PIC18F87K22 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由软件控制（如果配置为禁止的话）。

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。双速启动使得代码几乎可在起振发生时立即执行，与此同时，主时钟源正在完成起振延时。

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从 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的范围。事实上，它属于配置存储空间（300000h-3FFFFFh），该空间仅能通过表读和表写操作进行访问。

使用软件对配置寄存器编程类似于对闪存程序存储器编程。EECON1 寄存器中的 WR 位可启动对配置寄存器的自定时写操作。在正常工作模式下，执行 TBLWT 指令时，如果 TBLPTR 指向配置寄存器地址，将按照此指令指定的地址和数据写配置寄存器。将 WR 位置 1 可启动对配置寄存器的长写操作。每次向配置寄存器写入 1 个字节。要写入或擦除一个配置单元，可用 TBLWT 指令向该单元写入 1 或 0。欲知更多详情，请参见**第 7.5 节 “写入闪存程序存储器”**。

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 1--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
300004h	CONFIG3L	WAIT(2)	BW(2)	ABW1(2)	ABW0(2)	EASHFT(2)	—	—	RTCOSC	---- ---1
300005h	CONFIG3H	MCLRE	—	—	—	MSSPMSK	—	ECCPMX(2)	CCP2MX	1--- 1-11
300006h	CONFIG4L	$\overline{\text{DEBUG}}$	—	—	BBSIZ0	—	—	—	STVREN	1--1 ---1
300008h	CONFIG5L	CP7(1)	CP6(1)	CP5(1)	CP4(1)	CP3	CP2	CP1	CP0	1111 1111
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah	CONFIG6L	WRT7(1)	WRT6(1)	WRT5(1)	WRT4(1)	WRT3	WRT2	WRT1	WRT0	1111 1111
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch	CONFIG7L	EBTR7(1)	EBTR6(1)	EBTR5(1)	EBTR4(1)	EBTR3	EBTR2	EBTR1	EBTR0	1111 1111
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh	DEVID1(3)	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx
3FFFFFh	DEVID2(3)	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	xxxx xxxx

图注： x = 未知， u = 不变， - = 未实现， q = 取值视情况而定。阴影单元未实现，读为 0。

注 1： 仅在 PIC18F67K22 和 PIC18F87K22 器件中实现。

2： 仅在 80 引脚器件（PIC18F8XK22）中实现。

3： 关于 DEVID1 的值，请参见寄存器 28-14。 DEVID 寄存器为只读寄存器，用户不能对其进行编程。

© 2011 Microchip Technology Inc.

寄存器 28-1： CONFIG1L：配置寄存器 1 的低字节 （字节地址 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 = 上电复位时的值	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 电源选择和模式配置位
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 = LF-INTOSC 在休眠期间处于高功耗模式
0 = LF-INTOSC 在休眠期间处于低功耗模式

bit 1 **未实现：**读为 0

bit 0 **$\overline{\text{RETEN}}$：**VREG 休眠使能位
1 = 稳压器电源处于休眠模式时，由 VREGSLP （WDTCON<7>）控制
0 = 稳压器电源处于休眠模式时，由 SRETEN （WDTCON<4>）控制。将使能超低功耗稳压器。

寄存器 28-2： CONFIG1H：配置寄存器 1 的高字节 （字节地址 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 = 上电复位时的值	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 = 振荡器输出 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， RA6 用作 CLKOUT 功能的 EC 振荡器 （**低功耗， DC-160 kHz**）
1011 = EC2， EC 振荡器 （**中等功耗， 160 kHz-16 MHz**）
1010 = EC2IO， RA6 用作 CLKOUT 功能的 EC 振荡器 （**中等功耗， DC-160 kHz-16 MHz**）
1001 = INTIO1， RA6 用作 CLKOUT 功能的内部 RC 振荡器
1000 = INTIO2，内部 RC 振荡器
0111 = RC，外部 RC 振荡器
0110 = RCIO， RA6 用作 CLKOUT 功能的外部 RC 振荡器
0101 = EC3， EC 振荡器 （**高功耗， 4 MHz-64 MHz**）
0100 = EC3IO， RA6 用作 CLKOUT 功能的 EC 振荡器 （**高功耗， 4 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

注 **1：** 对于 INTIOx PLL 模式无效。
2： INTIO+PLL 只能由 PLLEN 位 （OSCTUNE<6>）使能。其他 PLL 模式由 PLLEN 位或 PLLCFG （CONFIG1H<4>）位使能。

© 2011 Microchip Technology Inc.

寄存器 28-3： CONFIG2L：配置寄存器 2 的低字节 （字节地址 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 = 上电复位时的值	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 = V_{BORMV} 设置为 1.8V
10 = V_{BORMV} 设置为 2.0V
01 = V_{BORMV} 设置为 2.7V
00 = V_{BORMV} 设置为 3.0V

bit 2-1 **BOREN<1:0>：**欠压复位使能位 (2)
11 = 欠压复位只能由硬件使能 （禁止 SBOREN）
10 = 欠压复位只能由硬件使能，且在休眠模式下禁止 （禁止 SBOREN）
01 = 欠压复位由软件使能和控制 （使能 SBOREN）
00 = 欠压复位由硬件和软件禁止

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：** 有关规范，请参考**第 31.1 节 “直流特性：供电电压 PIC18F87K22 系列 （工业级 / 扩展级）”**。
2： 上电延时定时器与欠压复位是相互独立的，这样可以分别控制两者的功能。

寄存器 28-4： CONFIG2H：配置寄存器 2 的高字节 （字节地址 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 = 上电复位时的值	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

10101-11111 = 保留
10100 = 1:1,048,576 （4,194.304 秒）
10011 = 1:524,288 （2,097.152 秒）
10010 = 1:262,144 （1,048.576 秒）
10001 = 1:131,072 （524.288 秒）
10000 = 1:65,536 （262.144 秒）
01111 = 1:32,768 （131.072 秒）
01110 = 1:16,384 （65.536 秒）
01101 = 1:8,192 （32.768 秒）
01100 = 1:4,096 （16.384 秒）
01011 = 1:2,048 （8.192 秒）
01010 = 1:1,024 （4.096 秒）
01001 = 1:512 （2.048 秒）
01000 = 1:256 （1.024 秒）
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 = WDT 由硬件使能；禁止 SWDTEN 位
10 = WDT 由 SWDTEN 位的设置控制
01 = WDT 仅在器件处于工作状态时使能，在休眠模式下禁止；禁止 SWDTEN 位
00 = WDT 由硬件禁止；禁止 SWDTEN 位

© 2011 Microchip Technology Inc.

寄存器 28-5： CONFIG3L：配置寄存器 3 的低字节 （字节地址 300004h）

U-1	U-1	U-1	U-1	U-1	U-0	U-0	R/P-1
WAIT(1)	BW(1)	ABW1(1)	ABW0(1)	EASHFT(1)	—	—	RTCOSC
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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **WAIT**：外部总线等待使能位 (1)
1 = 禁止外部总线上的等待状态
0 = 通过 MEMCON<5:4> 使能和选择外部总线上的等待状态

bit 6 **BW**：数据总线宽度选择位 (1)
1 = 16 位数据宽度模式
0 = 8 位数据宽度模式

bit 5-4 **ABW<1:0>**：外部存储器总线配置位 (1)
11 = 8 位地址模式 （单片机模式）
10 = 12 位地址模式
01 = 16 位地址模式
00 = 20 位地址模式

bit 3 **EASHFT**：外部地址总线平移使能位 (1)
1 = 使能地址平移；外部地址平移至以 000000h 为起始地址的位置
0 = 禁止地址平移；外部地址总线反映 PC 值

bit 2-1 **未实现**：读为 0

bit 0 **RTCOSC**：RTCC 参考时钟选择位
1 = RTCC 使用 SOSC 作为参考时钟
0 = RTCC 使用 LF-INTOSC 作为参考时钟

注 1： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。

寄存器 28-6： CONFIG3H：配置寄存器 3 的高字节 （字节地址 300005h）

R/P-1	U-0	U-0	U-0	R/P-1	U-0	R/P-1	R/P-1
MCLRE	—	—	—	MSSPMSK	—	ECCPMX(1)	CCP2MX
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 = 上电复位时的值	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 RG5 输入引脚
0 = 使能 RG5 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 **未实现：**读为 0

bit 1 **ECCPMX：**ECCP 多路复用位 (1)
1 =
- ECCP1 （P1B/P1C）与 RE6 和 RE5 复用， CCP6 与 RE6 复用，而 CCP7 与 RE5 复用
- ECCP3 （P3B/P3C）与 RE4 和 RE3 复用， CCP8 与 RE4 复用，而 CCP9 与 RE3 复用

0 =
- ECCP1 （P1B/P1C）与 RH7 和 RH6 复用， CCP6 与 RH7 复用，而 CCP7(2) 与 RH6 复用
- ECCP3 （P3B/P3C）与 RH5 和 RH4 复用， CCP8 与 RH5 复用，而 CCP9(2) 与 RH4 复用

bit 0 **CCP2MX：**ECCP2 多路复用位
1 = ECCP2 与 RC1 复用
0 = 在扩展单片机模式下， ECCP2 与 RB3 复用；在单片机模式下， ECCP2 与 RE7 复用

注 1： 在 64 引脚器件 （PIC18F6XK22）上未实现，读为 0。
2： 在 32K 器件 （PIC18F65K22 和 PIC18F85K22）上未实现。

© 2011 Microchip Technology Inc.

寄存器 28-7： CONFIG4L：配置寄存器 4 的低字节 （字节地址 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 = 上电复位时的值	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 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 千字引导区大小
0 = 1 千字引导区大小

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 / 下溢将导致复位
0 = 堆栈满 / 下溢不会导致复位

寄存器 28-8： CONFIG5L：配置寄存器 5 的低字节 （字节地址 300008h）

R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1
CP7(1)	CP6(1)	CP5(1)	CP4(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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **CP7：**代码保护位 (1)
1 = Block 7 无代码保护 (2)
0 = Block 7 有代码保护 (2)

bit 6 **CP6：**代码保护位 (1)
1 = Block 6 无代码保护 (2)
0 = Block 6 有代码保护 (2)

bit 5 **CP5：**代码保护位 (1)
1 = Block 5 无代码保护 (2)
0 = Block 5 有代码保护 (2)

bit 4 **CP4：**代码保护位 (1)
1 = Block 4 无代码保护 (2)
0 = Block 4 有代码保护 (2)

bit 3 **CP3：**代码保护位
1 = Block 3 无代码保护 (2)
0 = Block 3 有代码保护 (2)

bit 2 **CP2：**代码保护位
1 = Block 2 无代码保护 (2)
0 = Block 2 有代码保护 (2)

bit 1 **CP1：**代码保护位
1 = Block 1 无代码保护 (2)
0 = Block 1 有代码保护 (2)

bit 0 **CP0：**代码保护位
1 = Block 0 无代码保护 (2)
0 = Block 0 有代码保护 (2)

注 **1：** 此位只在 PIC18F67K22 和 PIC18F87K22 器件上可用。
2： 有关存储区大小的信息，请参见图 28-6。

© 2011 Microchip Technology Inc.

寄存器 28-9： CONFIG5H：配置寄存器 5 的高字节 （字节地址 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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **CPD：**数据 EEPROM 代码保护位
1 = 数据 EEPROM 无代码保护
0 = 数据 EEPROM 有代码保护

bit 6 **CPB：**引导区代码保护位
1 = 引导区无代码保护 (1)
0 = 引导区有代码保护 (1)

bit 5-0 **未实现：**读为 0

注 1： 有关存储区大小的信息，请参见图 28-6。引导区大小根据 BBSIZ0 的不同而变化。

寄存器 28-10： CONFIG6L：配置寄存器 6 的低字节 （字节地址 30000Ah）

R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1
WRT7[1]	WRT6[1]	WRT5[1]	WRT4[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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **WRT7：**写保护位 [1]
1 = Block 7 无代码保护 [2]
0 = Block 7 有代码保护 [2]

bit 6 **WRT6：**写保护位 [1]
1 = Block 6 无代码保护 [2]
0 = Block 6 有代码保护 [2]

bit 5 **WRT5：**写保护位 [1]
1 = Block 5 无代码保护 [2]
0 = Block 5 有代码保护 [2]

bit 4 **WRT4：**写保护位 [1]
1 = Block 4 无代码保护 [2]
0 = Block 4 有代码保护 [2]

bit 3 **WRT3：**写保护位
1 = Block 3 无代码保护 [2]
0 = Block 3 有代码保护 [2]

bit 2 **WRT2：**写保护位
1 = Block 2 无代码保护 [2]
0 = Block 2 有代码保护 [2]

bit 1 **WRT1：**写保护位
1 = Block 1 无代码保护 [2]
0 = Block 1 有代码保护 [2]

bit 0 **WRT0：**写保护位
1 = Block 0 无代码保护 [2]
0 = Block 0 有代码保护 [2]

注 1： 此位只在 PIC18F67K22 和 PIC18F87K22 器件上可用。
2： 有关存储区大小的信息，请参见图 28-6。

© 2011 Microchip Technology Inc.

寄存器 28-11： CONFIG6H：配置寄存器 6 的高字节 （字节地址 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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **WRTD：**数据 EEPROM 写保护位
1 = 数据 EEPROM 无写保护
0 = 数据 EEPROM 有写保护

bit 6 **WRTB：**引导区写保护位
1 = 引导区无代码保护 (2)
0 = 引导区有代码保护 (2)

bit 5 **WRTC：**配置寄存器写保护位 (1)
1 = 配置寄存器无写保护 (2)
0 = 配置寄存器有写保护 (2)

bit 4-0 **未实现：**读为 0

注 **1：** 在正常执行模式下，该位是只读的；该位仅在编程模式下才可写。
2： 有关存储区大小的信息，请参见图 28-6。

寄存器 28-12： CONFIG7L：配置寄存器 7 的低字节 （字节地址 30000Ch）

R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1
EBTR7[1,3]	EBTR6[1,3]	EBTR5[1,3]	EBTR4[1,3]	EBTR3[3]	EBTR2[3]	EBTR1[3]	EBTR0[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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **EBTR7**：表读保护位 [1,3]
1 = Block 7 无表读保护，可从其他存储区对其执行表读操作 [2]
0 = Block 7 有表读保护，不能从其他存储区对其执行表读操作 [2]

bit 6 **EBTR6**：表读保护位 [1,3]
1 = Block 6 无表读保护，可从其他存储区对其执行表读操作 [2]
0 = Block 6 有表读保护，不能从其他存储区对其执行表读操作 [2]

bit 5 **EBTR5**：表读保护位 [1,3]
1 = Block 5 无表读保护，可从其他存储区对其执行表读操作 [2]
0 = Block 5 有表读保护，不能从其他存储区对其执行表读操作 [2]

bit 4 **EBTR4**：表读保护位 [1,3]
1 = Block 4 无表读保护，可从其他存储区对其执行表读操作 [2]
0 = Block 4 有表读保护，不能从其他存储区对其执行表读操作 [2]

bit 3 **EBTR3**：表读保护位 [3]
1 = Block 3 无表读保护，可从其他存储区对其执行表读操作 [2]
0 = Block 3 有表读保护，不能从其他存储区对其执行表读操作 [2]

bit 2 **EBTR2**：表读保护位 [3]
1 = Block 2 无表读保护，可从其他存储区对其执行表读操作 [2]
0 = Block 2 有表读保护，不能从其他存储区对其执行表读操作 [2]

bit 1 **EBTR1**：表读保护位 [3]
1 = Block 1 无表读保护，可从其他存储区对其执行表读操作 [2]
0 = Block 1 有表读保护，不能从其他存储区对其执行表读操作 [2]

bit 0 **EBTR0**：表读保护位 [3]
1 = Block 0 无表读保护，可从其他存储区对其执行表读操作 [2]
0 = Block 0 有表读保护，不能从其他存储区对其执行表读操作 [2]

注 **1**： 此位只在 PIC18F67K22 和 PIC18F87K22 器件上可用。
2： 有关存储区大小的信息，请参见图 28-6。
3： 使能相应的 CPx 位以防止对存储区进行外部读操作。

© 2011 Microchip Technology Inc.

寄存器 28-13： CONFIG7H：配置寄存器 7 的高字节 （字节地址 30000Dh）

U-0	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB[2]	—	—	—	—	—	—
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 = 上电复位时的值	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 [2]

1 = 引导区无表读保护，可从其他存储区对其执行表读操作 [1]

0 = 引导区有表读保护，不能从其他存储区对其执行表读操作 [1]

bit 5-0 **未实现：**读为 0

注 **1：** 有关存储区大小的信息，请参见图 28-6。

2： 使能相应的 CPx 位以防止对存储区进行外部读操作。

寄存器 28-14： DEVID1：PIC18F87K22 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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-5 **DEV<2:0>：**器件 ID 位

DEV<10:3> 为 0101 0010 的器件（见 DEVID2）：
010 = PIC18F65K22
000 = PIC18F66K22
101 = PIC18F85K22
011 = PIC18F86K22

DEV<10:3> 为 0101 0001 的器件：
000 = PIC18F67K22
010 = PIC18F87K22

bit 4-0 **REV<4:0>：**版本 ID 位
这些位用于表明器件的版本。

寄存器 28-15： DEVID2：PIC18F87K22 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 = 上电复位时的值	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> 位配合使用，来确定器件编号。
0101 0010 = PIC18F65K22、 PIC18F66K22、 PIC18F85K22 和 PIC18F86K22
0101 0001 = PIC18F67K22 和 PIC18F87K22

注 1： DEV<10:3> 的值可能会用于其他器件。特定器件始终通过使用整个 DEV<10:0> 位序列来标识。

© 2011 Microchip Technology Inc.

28.2 看门狗定时器 （WDT）

对于 PIC18F87K22 系列器件，WDT 由 LF-INTOSC 源驱动。当使能 WDT 时，也将同时使能时钟源。 WDT 超时溢出周期的标称值为 4 ms，其稳定性与 LF-INTOSC振荡器相同。

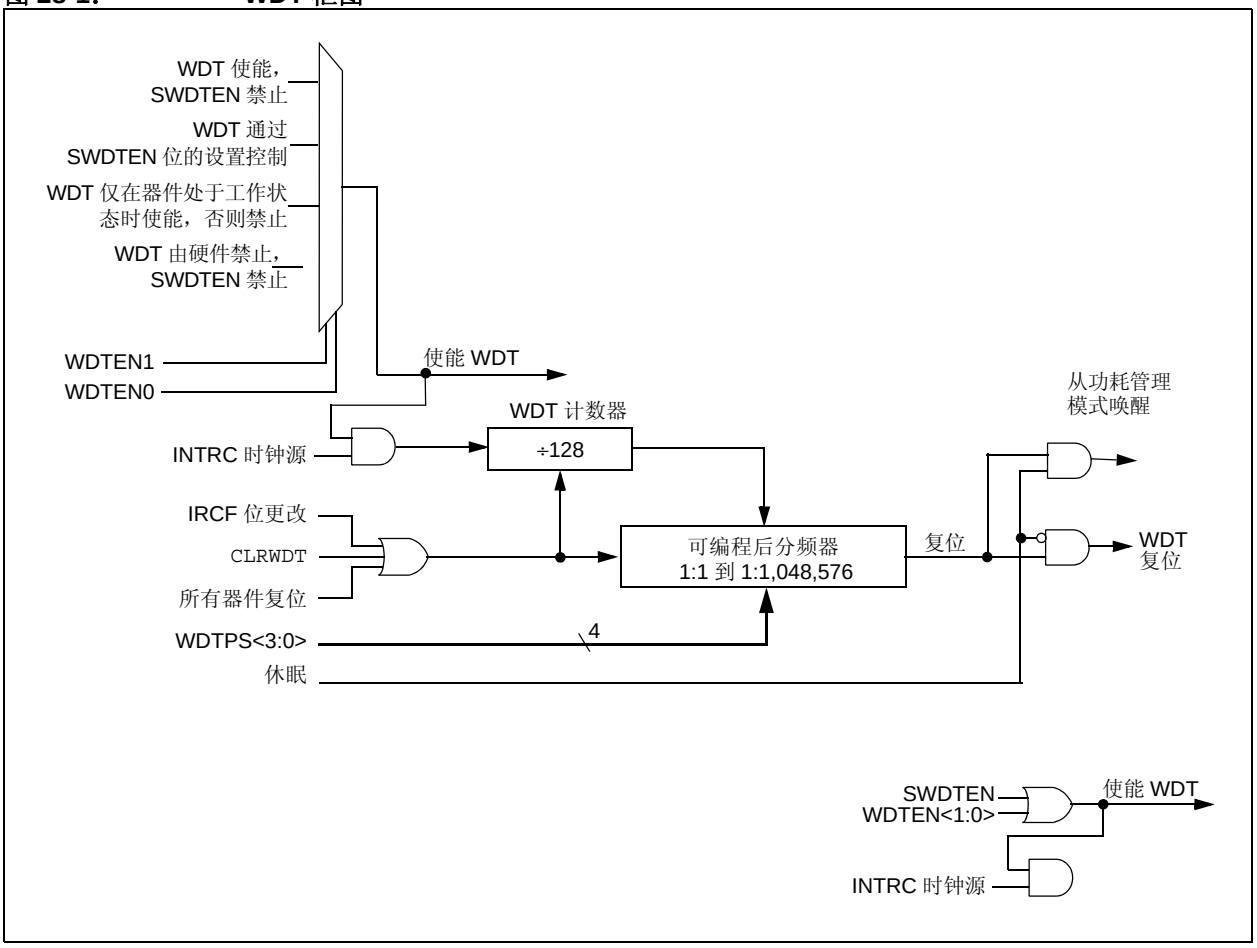
4 ms 的 WDT 超时溢出周期与 16 位后分频器的值相乘。通过配置寄存器 2H 中的位控制一个多路开关以对 WDT 后分频器的输出进行选择。可用的超时溢出周期范围是从 4 ms 到 4194 秒（大约 1 小时）。发生以下任一事件时， WDT 和后分频器将清零：执行 SLEEP 或 CLRWDT 指令，更改 IRCF 位 （OSCCON<6:4>）或发生时钟故障。

WDT 可以工作在以下 4 种模式，具体由 WDTEN<1:0>（CONFIG2H<1:0>）位确定。这 4 种模式包括：

- WDT 使能
- WDT 禁止
- WDT 由软件控制 （SWDTEN 位 WDTCON<0>）
- WDT
  - 在正常工作期间使能
  - 在休眠期间禁止

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 28-16 给出了 WDTCON 寄存器。这是一个可读写的寄存器，它包含一个控制位，仅当通过配置位禁止 WDT 时，该控制位才允许使用软件来改写 WDT 使能配置位。

寄存器 28-16： 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-0	R/W-0	R/W-0
REGSLP	—	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 = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 7 **REGSLP：**稳压器休眠使能位
1 = 使能器件的休眠模式时，稳压器进入低功耗模式
0 = 激活器件的休眠模式时，稳压器保持在正常工作模式下

bit 6 **未实现：**读为 0

bit 5 **ULPLVL：**超低功耗唤醒输出位
无效，除非 ULPEN = 1。
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 且使能稳压器，那么在休眠期间器件将进入超低功耗模式
0 = 使能器件的休眠模式，且低功耗模式由 REGSLP 控制时，稳压器处于工作状态

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
无效，除非 ULPEN = 1。
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：**由软件控制的看门狗定时器使能位 (1)
1 = 使能看门狗定时器
0 = 禁止看门狗定时器

注 1： 如果使能配置位 WDTEN<1:0>，此位不起作用。
2： 此位仅在 ENVREG = 1 且 $\overline{\text{RETEN}}$ = 0 时可用。

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。

© 2011 Microchip Technology Inc.

28.3 片上稳压器

所有 PIC18F87K22 系列器件使用标称值为 3.3V 的电压为其内核数字逻辑供电。对于需要更高典型工作电压（如 5V）的设计，由于系列的所有器件均包含两个片上稳压器，因此可使器件内核逻辑运行在 VDD 下。这些稳压器包括：

- 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相同，并且在**第 28.3.1 节 “稳压器由硬件使能 / 禁止”**中进行说明。唯一不同之处在于器件进入休眠状态的时间，如**第 28.3.2 节 “稳压器在休眠模式下的操作”**中所述。

28.3.1 稳压器由硬件使能 / 禁止

稳压器只能由硬件使能或禁止。稳压器由 ENVREG 引脚和 VDDCORE/VCAP 引脚控制。

28.3.1.1 稳压器使能模式

将 VDD 与引脚相连将使能稳压器，而稳压器通过其他 VDD 引脚为内核提供电源。

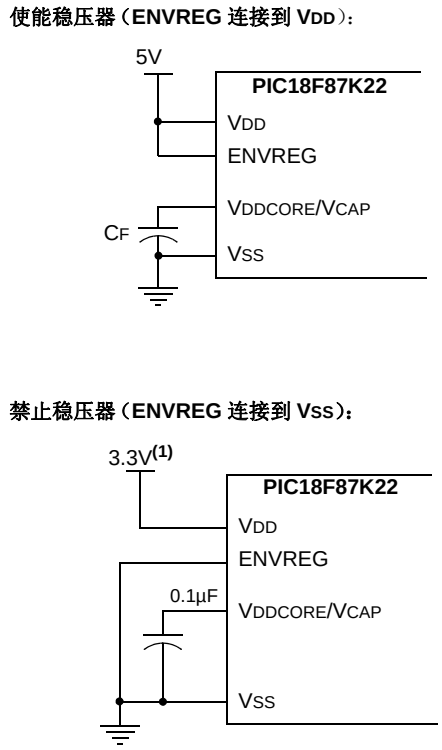
使能稳压器时，必须将低 ESR 滤波电容连接到 VDDCORE/VCAP 引脚（见图 28-2）。这有助于维持稳压器的稳定性。滤波电容的推荐值在**第 31.2 节 “直流特性：掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）”**中给出。

28.3.1.2 稳压器禁止模式

如果稳压器通过将 VSS 连接到 ENVREG 引脚禁止，将通过 VDD 直接为内核提供电源。VDD 的电压不得超出指定的 VDDCORE 电压。在稳压器禁止模式下，应该在 VDDCORE/VCAP 引脚连接一个 0.1 µF 的电容（见图 28-2）。

使用稳压器时，整体电压预算非常紧张。稳压器应能使器件在低至 1.8V 的电压下工作。当 VDD 下降到 3.3V 以下时，稳压器将不再稳定，但是在 VDD 达到 1.8V 前，输出电压跟随输入电压。低于 1.8V 后，稳压器的输出电压可能降至 0V。

图 28-2： 片上稳压器的连接



注 1： 这些为典型工作电压。有关 VDD 和 VDDCORE 的全部工作电压范围，请参见**第 31.2 节 “直流特性：掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）”**。

28.3.2 稳压器在休眠模式下的操作

休眠模式下，两个稳压器之间的操作会有所不同。在稳压器使能模式下，超低功耗稳压器将为器件提供最低电流。

器件进入休眠状态时，片上稳压器可以通过将 REGSLP 位（WDTCON<7>）置 1 来进入较低功耗模式。这会使稳压器进入待机模式，以使器件消耗较少的电流。

片上稳压器还可以进入超低功耗模式，此时消耗的电流是在使能稳压器的情况下消耗的最少电流。此模式可以通过 RETEN 位（CONFIG1L<0>）和 SRETEN 位（WDTCON<4>）控制。

稳压器的各种工作模式如表 28-3 所示。

超低功耗稳压器处于休眠模式时，芯片中的内部参考电压将关闭，且与内部参考电压相关的任何中断都不会唤醒器件。如果使能 BOR 或 LVD，稳压器将保持内部参考电压处于开启状态，但不会达到最低可能电流。

在休眠模式下使用超低功耗稳压器时，器件在唤醒后通常需要 250 µs 才能开始执行代码。

表 28-3： 休眠模式稳压器设置 (1)

稳压器	功耗模式	VREGSLP WDTCON<7>	SRETEN WDTCON<4>	$\overline{\text{RETEN}}$ CONFIG1L<0>
使能	正常工作 （休眠）	0	x	1
使能	低功耗模式 （休眠）	1	x	1
使能	正常工作 （休眠）	0	0	x
使能	低功耗模式 （休眠）	1	0	x
使能	超低功耗模式 （休眠）	x	1	0

注 1： x = 表示 VIT 状态无效。

© 2011 Microchip Technology Inc.

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 INTRC（LF-INTOSC、MF-INTOSC 或 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；可通过置1 IESO配置位使能双速启动。

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。

当使能双速启动时，在使能上电复位且上电延时定时器发生超时后，器件复位或从休眠模式唤醒都会使器件将自身配置成使用内部振荡器模块作为时钟源来运行。这样几乎可使在主振荡器起振、OST 运行的同时立即执行代码。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更快的时钟速率，可以选择 INTOSC 或后分频器时钟源以提供更快的时钟速率，这可以通过在复位发生后立即设置 IRCF<2:0> 位实现。对于从休眠模式唤醒的情况，可以通过在进入休眠模式之前设置 IRCF<2:0> 位来选择 INTOSC 或后分频器时钟源。

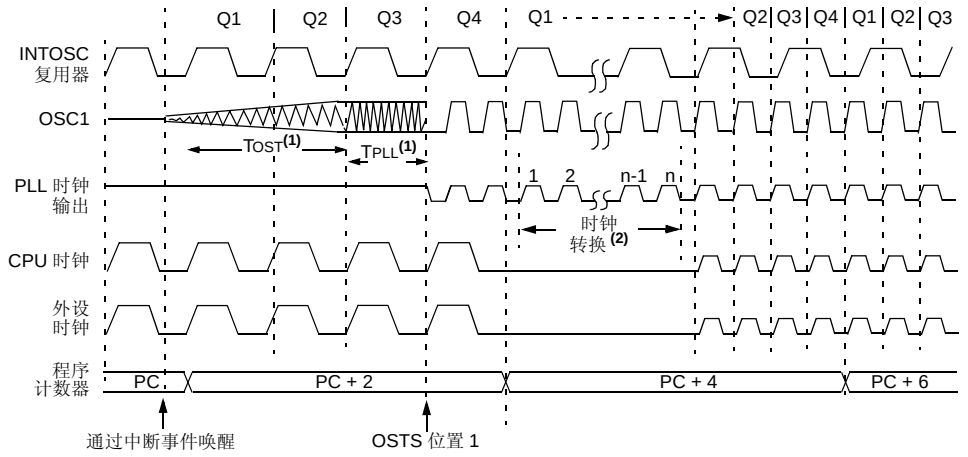
在所有其他功耗管理模式下，不使用双速启动。器件将使用当前选定的时钟源提供时钟，直到主时钟源可用为止。 IESO 位的设置被忽略。

28.4.1 使用双速启动时的特别注意事项

当在双速启动中使用 INTOSC 振荡器时，器件仍将遵守进入功耗管理模式的正常命令序列，包括执行多条 SLEEP 指令（见**第 4.1.4 节“多条 Sleep 指令”**）。实际上，这意味着在 OST 超时前用户代码可以改变 SCS<1:0> 位的设置或执行 SLEEP 指令。这就使应用程序能短暂地唤醒器件，执行“日常事务”并在器件开始使用主振荡器前返回休眠状态。

用户代码还能通过检查 OSTS 位（OSCCON<3>）的状态来确定当前主时钟源是否正在为器件提供时钟。若该位置 1，则表示主振荡器正在提供时钟。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电路提供时钟。

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显示。

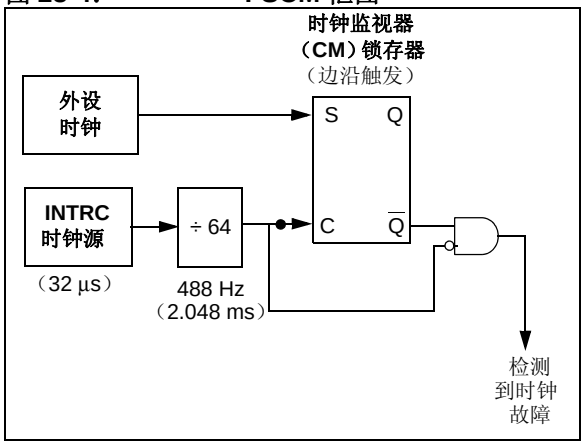
2：时钟转换通常发生在 2-4 个 TOSC 内。

28.5 故障保护时钟监视器

故障保护时钟监视器（FSCM）可使单片机在发生外部振荡器故障时，自动将器件时钟切换到内部振荡器模块以保持器件继续运行。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提供备用时钟。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 输出的 64 分频。这样就使得 FSCM 采样时钟脉冲之间有充足的时间间隔，从而保证在此时间段内至少有一个外设时钟边沿出现。外设时钟和采样时钟作为时钟监视器（Clock Monitor，CM）锁存器的输入。CM 在器件时钟源的下降沿置 1，在采样时钟的上升沿清零。

图 28-4： FSCM 框图



在采样时钟的下降沿检测时钟故障。如果在出现采样时钟的下降沿时，CM 仍置 1，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 （PIR2<7>）置 1，由 FSCM 产生振荡器故障中断
- 器件时钟源切换为内部振荡器电路（OSCCON 不会被更新，因此无法显示当前时钟源，这就是故障保护状态）
- WDT 复位

切换过程中，对于时序要求较高的应用，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安全关闭。请参见**第 4.1.4 节 “多条 Sleep 指令”**和**第 28.4.1 节“使用双速启动时的特别注意事项”**了解更多详情。

为了在唤醒器件时使用更快的时钟速率，可以选择 INTOSC 或后分频器时钟源以提供更快的时钟速率，这可以通过在复位发生后立即设置 IRCF<2:0> 位实现。对于从休眠模式唤醒的情况，可以通过在进入休眠模式之前设置 IRCF<2:0> 位来选择 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，将无法检测到故障，当然也就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工作，当使能 FSCM 时，禁止 WDT 不影响 INTOSC 振荡器的运行。

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变化。如果用小预分频值使能 WDT，时钟速率的下降将使 WDT 发生超时并在随后使器件复位。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，从而使 WDT 从运行速度发生变化的那一刻起开始计时，因而减少了发生错误超时的可能性。

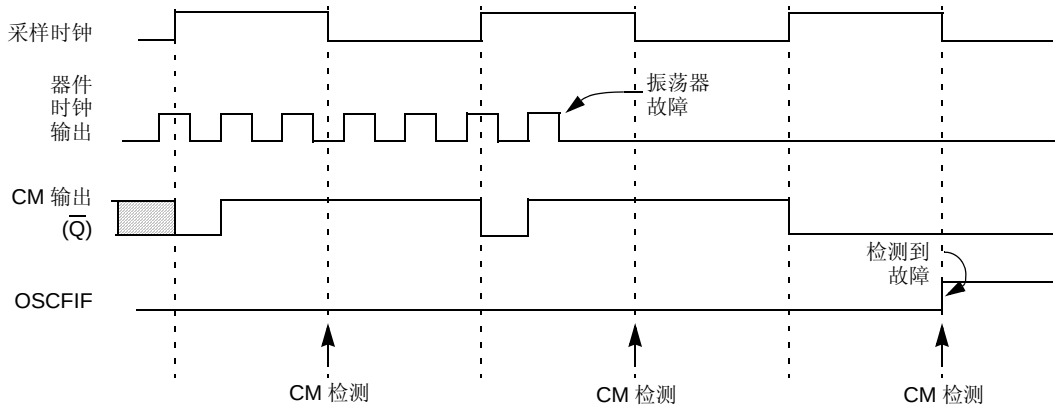
28.5.2 退出故障保护运行模式

器件复位或进入功耗管理模式均可终止故障保护状态。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（伴有如 OST 或 PLL 定时器等振荡器模式所需的起振延时）。INTOSC 复用器将在主时钟源就绪前提供器件时钟（类似于双速启动）。当主时钟源可用时，时钟源将切换回主时钟（OSCCON 寄存器中的 OSTS 位置 1 以表明当前使用的是主时钟源）。然后，故障保护时钟监视器恢复对外设时钟的监视。

在启动期间，主时钟源可能永远不能就绪。在这种情况下，器件运行将以 INTOSC 复用器作为时钟源。OSCCON寄存器将保持复位状态直到进入功耗管理模式为止。

© 2011 Microchip Technology Inc.

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。为便于图示，本例中两者的相对值与实际不相符。

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。在功耗管理模式下将恢复对功耗管理时钟源的故障保护时钟监视。

如果在功耗管理运行期间发生振荡器故障，随后发生的事件取决于是否允许了振荡器故障中断标志位（OSCFIF = 1）。如果允许中断，代码执行将以INTOSC 复用器作为时钟源。并不会自动转回到发生故障的时钟源。

如果禁止了该中断，空闲模式下随后发生的中断将使CPU 开始执行指令，此时由 INTOSC 时钟源为 CPU 提供时钟。

28.5.4 POR 或从休眠中唤醒

FSCM 在器件退出上电复位（POR）或低功耗休眠模式后开始检测振荡器故障。当系统主时钟为 EC、 RC 或 INTRC 模式时，监视会在这些事件发生后立即开始。

当振荡器模式使用了晶振或谐振器 （HS、 HSPLL、 LP 或 XT）时，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 采样时钟的时间长很多，因此可能会检测到假时钟故障。为了避免这一情况，内部振荡器模块会被自动配置为器件时钟并一直工作直到主时钟稳定下来为止 （OST 和 PLL 定时器已超时）。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：	防止在发生POR或从休眠状态唤醒时发生假振荡器故障中断的逻辑同样也将阻止在发生这些事件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标记任何振荡器故障中断。

正如第 28.4.1 节“使用双速启动时的特别注意事项”中所述，在等待系统主时钟稳定的过程中，可以选择另一种时钟配置并进入其他功耗管理模式。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

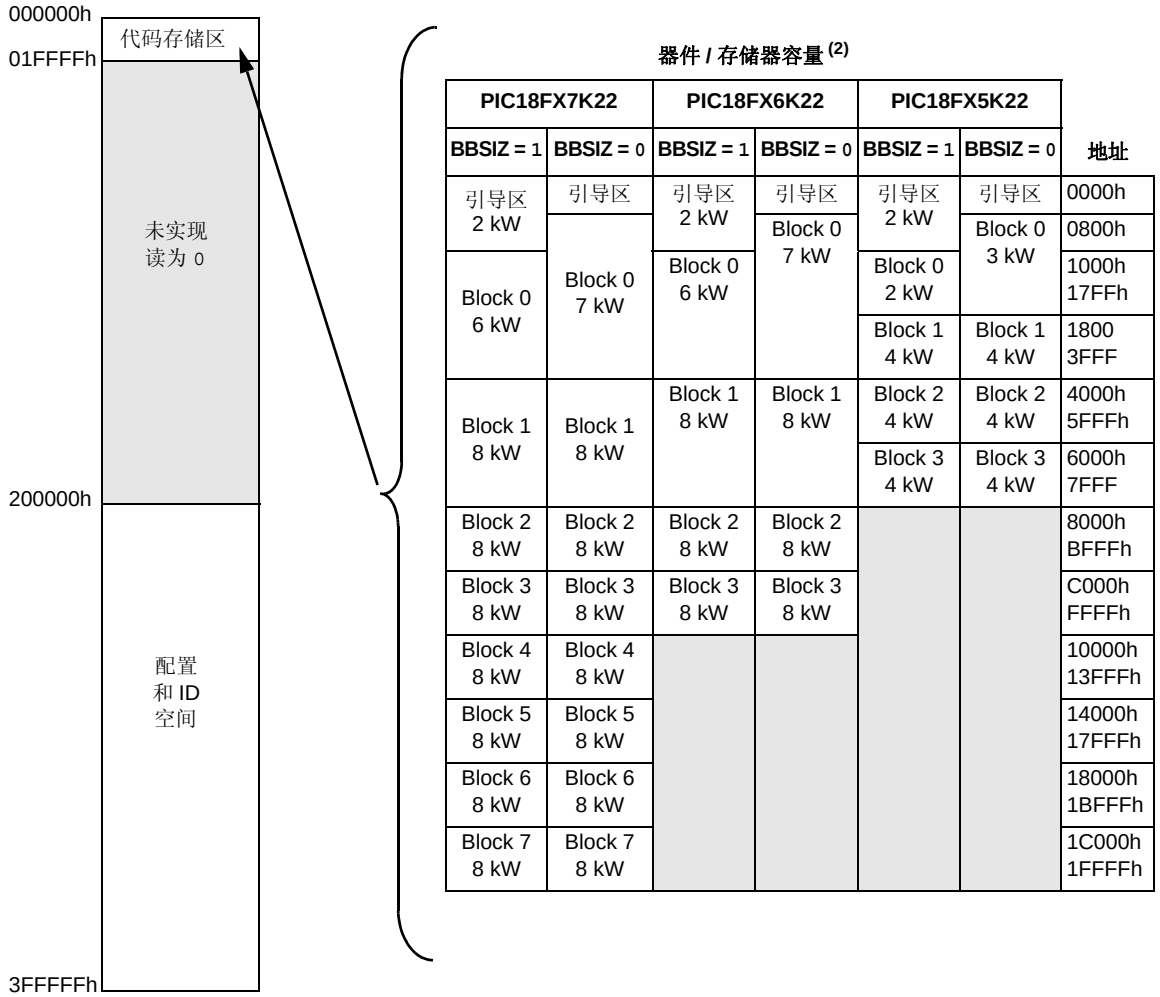
对于 PIC18FX5K22 器件和 PIC18FX6K22 器件，用户程序存储器分为 4 个存储区，而对于 PIC18FX7K22 器件，分为 8 个存储区。其中一个存储区是 1 KB 或 2 KB 的引导区。存储器的剩余空间根据二进制边界被分为若干存储区。

每个存储区都有三个代码保护位与之相关联。它们是：

- 代码保护位 （CPx）
- 写保护位 （WRTx）
- 外部存储区表读位 （EBTRx）

图 28-6 显示了 48 KB、 64 KB、 96 KB 和 128 KB 器件的程序存储器结构，以及与每个存储区相关的特定代码保护位。表 28-4 总结了这些位的实际位置。

图 28-6： PIC18F87K22 系列器件的受代码保护的程序存储器 (1)



注 1： 存储区大小未按比例显示。

2： 引导区大小由 BBSIZ0 位 （CONFIG4L<4>）确定。

© 2011 Microchip Technology Inc.

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	CP7(1)	CP6(1)	CP5(1)	CP4(1)	CP3	CP2	CP1	CP0
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—
30000Ah	CONFIG6L	WRT7(1)	WRT6(1)	WRT5(1)	WRT4(1)	WRT3	WRT2	WRT1	WRT0
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—
30000Ch	CONFIG7L	EBTR7(1)	EBTR6(1)	EBTR5(1)	EBTR4(1)	EBTR3	EBTR2	EBTR1	EBTR0
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—

图注： 阴影单元未实现。
注 1： 此位只在 PIC18F67K22 和 PIC18F87K22 器件上可用。

28.6.1 程序存储器代码保护

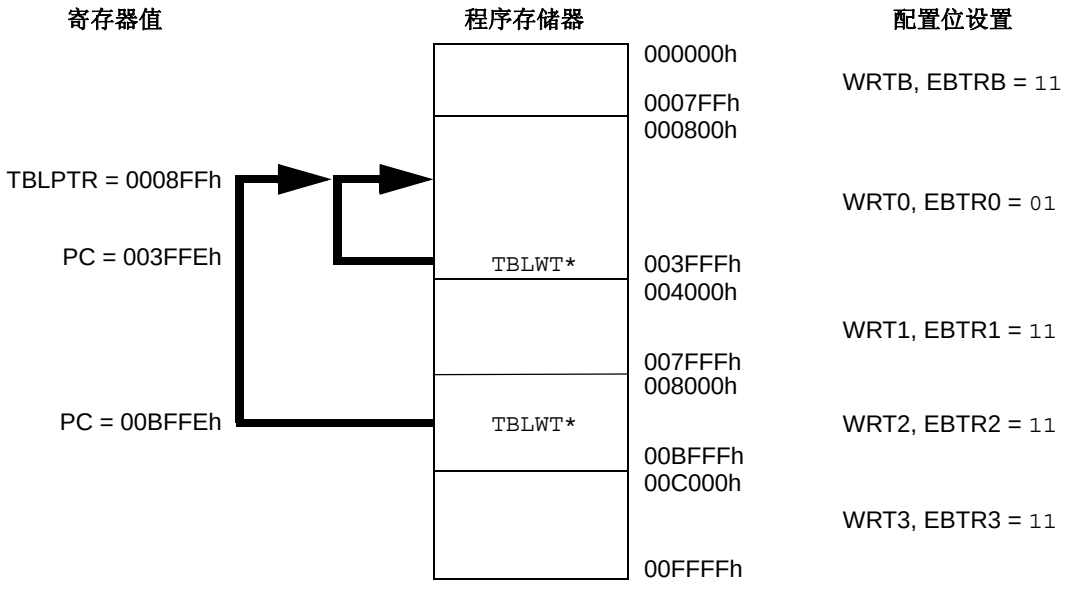
使用表读和表写指令可以对程序存储器的任何单元进行读写操作。器件 ID 可由表读指令读取。使用表读和表写指令可以对配置寄存器进行读写操作。

在正常执行模式下，CPx 位没有直接的影响。CPx 位用于禁止外部读写。如果 WRTx 配置位为 0，那么可以保护相应的用户存储区不受表写操作的影响。

EBTRx 位控制表读操作。对于 EBTRx 位置为 0 的用户存储区，允许在该存储区内执行表读指令。不允许从该存储区以外的存储单元执行表读指令，读到的结果将为 0。图 28-7 到 28-9 举例说明了表读和表写保护。

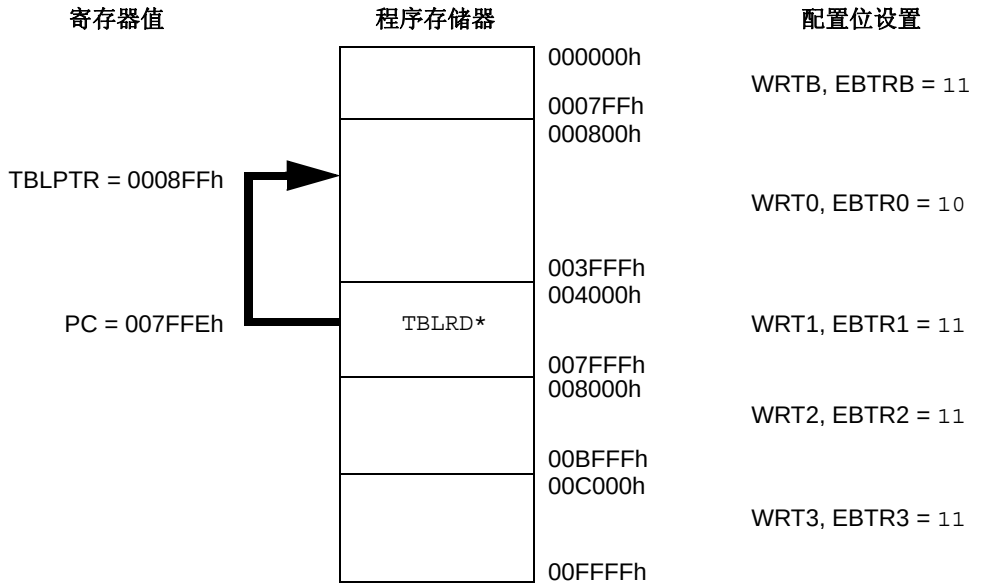
> **注：** 代码保护位只能从 1 状态写为 0。不能将 1 写入状态为 0 的位。只有通过整片擦除或块擦除功能才能将代码保护位设置为1。而这些功能只能通过 ICSP 或外部编程器启动。欲知更多信息，请参见器件编程规范。

图 28-7： 禁止表写操作 （WRTx）



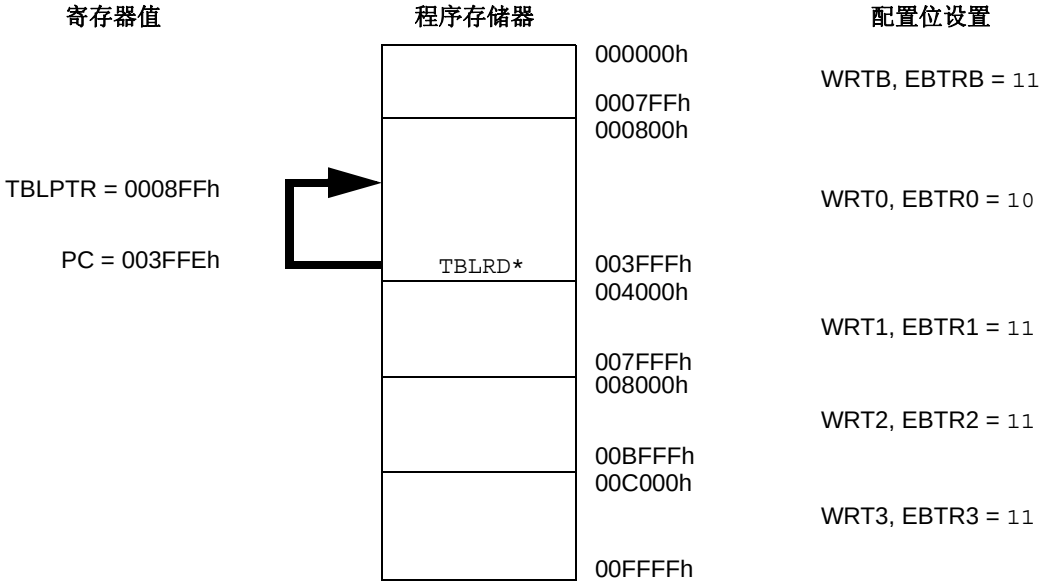
结果：只要 WRTx = 0，所有对 Blockn 的表写操作都将被禁止。

图 28-8： 禁止外部存储区表读操作 （EBTRx）



结果： 只要 EBTRx = 0，所有从外部存储区对 Blockn 的表读操作都将被禁止。
TABLAT 寄存器返回值 0。

图 28-9： 允许外部存储区表读操作 （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 指向的存储单元中的数据值。

© 2011 Microchip Technology Inc.

28.6.2 数据 EEPROM 代码保护

整个数据 EEPROM 受外部读写保护，这通过 CPD 和 WRTD 两个位来实现。CPD 禁止对数据 EEPROM 的外部读写操作。WRTD 禁止从内部和外部写数据 EEPROM。在正常操作时，不论保护位设置如何，CPU 都始终能读数据 EEPROM。

28.6.3 配置寄存器保护

配置寄存器可以被写保护。WRTC 位控制对配置寄存器的保护。在正常执行模式下，WRTC 位是只读的。WRTC 仅能通过 ICSP 或外部编程器写入。

28.7 ID 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号。在正常执行期间，可通过 TBLRD 和 TBLWT 指令读写这些单元；也可在编程 / 校验期间读写这些单元。当器件有代码保护时，仍可读取 ID 单元。

28.8 在线串行编程

PIC18F87K22 系列器件可以在最终应用电路中进行串行编程。只需要 5 根线就可以完成这一操作，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。这允许用户使用未编程器件制造电路板，仅在产品交付前才对单片机进行编程，从而可使用最新固件或定制固件进行编程。

关于各种编程模式，请参见相应的器件编程规范。

28.9 在线调试器

当 $\overline{\text{DEBUG}}$ 配置位被编程为 0 时，可使能在线调试器（ICD）功能。这一功能允许使用 MPLAB® IDE 进行一些简单的调试。当使能了单片机的这项功能时，某些资源就不再是通用的了。表 28-5 显示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器功能，在设计时就必须实现与 MCLR/RG5/VPP、VDD、VSS、RB7 和 RB6 的在线串行编程连接，从而与 Microchip 或某个第三方开发工具公司提供的在线调试器模块接口。

注：

© 2011 Microchip Technology Inc.

29.0 指令集汇总

PIC18F87K22 系列器件具有一个含有 75 条 PIC18 内核指令的标准指令集和一个含有 8 条新指令（优化递归或软件堆栈代码）的扩展指令集。本章后面的部分将讨论该扩展指令集。

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 PIC MCU 指令集移植的特点。大部分指令只占用一个程序存储字（16 位），只有 4 条指令需要两个程序存储单元。

每条单字指令都是一个 16 位字，由操作码（指定指令类型）和一个或多个操作数（进一步指定指令操作）组成。

整个指令集具有高度的正交性，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 中的 PIC18 指令集汇总列出了**字节**操作类指令、**位**操作类指令以及**立即数**和**控制**操作类指令。表 29-1 给出了操作码字段的说明。

大部分**字节**操作类指令含有三种操作数：

1. 文件选择寄存器（由 f 指定）
2. 保存结果的目标寄存器（由 d 指定）
3. 被访问的存储区（由 a 指定）

文件选择寄存器标识符 f 指定了指令将会使用哪一个文件选择寄存器。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。如果 d 为 1，操作结果将存入指令指定的文件选择寄存器中。

所有**位**操作类指令都含有三种操作数：

1. 文件选择寄存器（由 f 指定）
2. 文件选择寄存器中的位（由 b 指定）
3. 被访问的存储区（由 a 指定）

位域标识符 b 选择操作所影响的位的编号，而文件选择寄存器标识符 f 则代表该位所在寄存器的编号。

立即数操作类指令可以使用以下操作数：

- 要装入文件选择寄存器中的立即数（由 k 指定）
- 要装入立即数的目标 FSR 寄存器（由 f 指定）
- 不需要操作数（由 — 指定）

控制操作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，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。如果第二个字作为一条指令执行，它会执行 NOP 指令。

除非条件测试结果为 true 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，执行指令需要两个指令周期，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，对于频率为 4 MHz 的振荡器，其正常的指令执行时间为 1 µs。如果条件测试为 true 或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 true）的执行需要 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” 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 单元 （BSR 寄存器被忽略） a = 1：由 BSR 寄存器指定的 RAM 存储区
bbb	8 位文件选择寄存器的位地址 （0 到 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, N	ALU 状态位：进位标志位、半进位标志位、全零标志位、溢出标志位和负标志位。
d	目标寄存器选择位： d = 0：结果保存至 WREG 寄存器 d = 1：结果保存至文件选择寄存器 f
dest	目标寄存器：可以是 WREG 寄存器或指定的寄存器单元。
f	8 位寄存器地址 （00h 到 FFh）或 2 位 FSR 标识符 （0h 到 3h）。
f_s	12 位寄存器地址 （000h 到 FFFh）。这是源地址。
f_d	12 位寄存器地址 （000h 到 FFFh）。这是目标地址。
GIE	全局中断允许位。
k	立即数字段、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。 只与表读与表写指令一起使用： 不改变寄存器 （如用于表读和表写的 TBLPTR） 后增寄存器 （如用于表读和表写的 TBLPTR） 后减寄存器 （如用于表读和表写的 TBLPTR） 预增寄存器 （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）或调用 / 跳转和返回指令的直接地址。
PC	程序计数器。
PCL	程序计数器的低字节。
PCH	程序计数器的高字节。
PCLATH	程序计数器的高字节锁存器。
PCLATU	程序计数器的最高字节锁存器。
$\overline{PD}$	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对应的主寄存器 s = 1：将寄存器的值存入影子寄存器或把影子寄存器的值装入其对应的寄存器 （快速模式）
TBLPTR	21 位表指针 （指向程序存储单元）。
TABLAT	8 位表锁存器。
$\overline{TO}$	超时溢出位。
TOS	栈顶。
u	未用或不变。
WDT	看门狗定时器。
WREG	工作寄存器 （累加器）。
x	任意值 （0 或 1）。汇编器将产生 x = 0 的代码。为了与所有 Microchip 软件工具兼容，建议使用这种格式。
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 指定的寄存器中的位 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）。

© 2011 Microchip Technology Inc.

图 29-1：　　指令的一般格式

面向字节的文件寄存器操作　　**指令示例**

15 … 10	9	8	7 … 0
操作码	d	a	f（寄存器地址）

`ADDWF MYREG, W, B`

d = 0 表示结果存入 WREG 寄存器
d = 1 表示结果存入文件寄存器（f）
a = 0 强制使用快速操作存储区
a = 1 根据 BSR 选择存储区
f = 8 位文件寄存器地址

字节到字节的传送操作（双字）

15 … 12	11 … 0
操作码	f（源寄存器地址）
1111	f（目标寄存器地址）

`MOVFF MYREG1, MYREG2`

f = 12 位文件寄存器地址

面向位的文件寄存器操作

15 … 12	11 … 9	8	7 … 0
操作码	b（位编号）	a	f（寄存器地址）

`BSF MYREG, bit, B`

b = 占 3 位，表示文件选择寄存器（f）中位的位置
a = 0 强制使用快速操作存储区
a = 1 根据 BSR 选择存储区
f = 8 位文件寄存器地址

立即数操作

15 … 8	7 … 0
操作码	k（立即数）

`MOVLW 7Fh`

k = 8 位立即数值

控制操作

CALL、**GOTO** 和**跳转类**操作

15 … 8	7 … 0
操作码	n<7:0>（立即数）
1111	n<19:8>（立即数）

`GOTO Label`

n = 20 位立即数值

15 … 8	7	7 … 0
操作码	S	n<7:0>（立即数）
1111	n<19:8>（立即数）	

`CALL MYFUNC`

S = 快速位

15 … 11	10 … 0
操作码	n<10:0>（立即数）

`BRA MYFUNC`

15 … 8	7 … 0
操作码	n<7:0>（立即数）

`BC MYFUNC`

表 29-2： PIC18F87K22 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
字节操作类指令						
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 与 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	将 f 取反	1	0001 11da	ffff ffff	Z, N	1, 2
CPFSEQ f, a	f 与 WREG 做比较，相等则跳过	1 （2 或 3）	0110 001a	ffff ffff	无	4
CPFSGT f, a	f 与 WREG 做比较，大于则跳过	1 （2 或 3）	0110 010a	ffff ffff	无	4
CPFSLT f, a	f 与 WREG 做比较，小于则跳过	1 （2 或 3）	0110 000a	ffff ffff	无	1, 2
DECF f, d, a	f 减 1	1	0000 01da	ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ f, d, a	f 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 减 1，非 0 则跳过	1 （2 或 3）	0100 11da	ffff ffff	无	1, 2
INCF f, d, a	f 加 1	1	0010 10da	ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ f, d, a	f 加 1，为 0 则跳过	1 （2 或 3）	0011 11da	ffff ffff	无	4
INFSNZ f, d, a	f 加 1，非 0 则跳过	1 （2 或 3）	0100 10da	ffff ffff	无	1, 2
IORWF f, d, a	WREG 与 f 作逻辑或运算	1	0001 00da	ffff ffff	Z, N	1, 2
MOVF f, d, a	移动 f	1	0101 00da	ffff ffff	Z, N	1
MOVFF f_s, f_d	f_s （源）地址移入第一个字 f_d （目标）地址移入第二个字	2	1100 ffff 1111 ffff	ffff ffff ffff ffff	无	
MOVWF f, a	将 WREG 移入 f	1	0110 111a	ffff ffff	无	
MULWF f, a	WREG 乘以 f	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	对 f 执行带进位的循环左移	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	对 f 执行带进位的循环右移	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 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 与 f 作逻辑异或运算	1	0001 10da	ffff ffff	Z, N	

注 1： PORT 寄存器的值随端口功能的变化而不断修改 （例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一个引脚配置为输入，虽然其数据锁存器中的值为 1，但如果此时外部器件将该引脚驱动为低电平，则被写回数据总线的数据值将是 0。

2： 当对 TMR0 寄存器 （并且 d = 1）执行该指令时，如果已将预分频器分配给了 TMR0，则将其清零。

3： 如果程序计数器 （PC）被修改或者条件检测为 true，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存储的都是合法的指令。

© 2011 Microchip Technology Inc.

表 29-2： PIC18F87K22 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
位操作类指令							
BCF	f, d, a	将 f 中的某位清零	1	1001 bbba	ffff ffff	无	1, 2
BSF	f, d, a	将 f 中的某位置 1	1	1000 bbba	ffff ffff	无	1, 2
BTFSC	f, d, a	检测 f 中的某位，为 0 则跳过	1 （2 或 3）	1011 bbba	ffff ffff	无	3, 4
BTFSS	f, d, a	检测 f 中的某位，为 1 则跳过	1 （2 或 3）	1010 bbba	ffff ffff	无	3, 4
BTG	f, d, a	将 f 中的某位取反	1	0111 bbba	ffff ffff	无	1, 2
控制操作类指令							
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	$\overline{TO}$ 和 $\overline{PD}$	
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）	1	0000 0000	0000 0110	无	
PUSH	—	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		用软件使器件复位	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	$\overline{TO}$ 和 $\overline{PD}$	

注 1： PORT 寄存器的值随端口功能的变化而不断修改 （例如， `MOVF PORTB, 1, 0`），修改时使用的值是引脚上的当前值。例如，如果将一个引脚配置为输入，虽然其数据锁存器中的值为 1，但如果此时外部器件将该引脚驱动为低电平，则被写回数据总线的数据值将是 0。

2： 当对 TMR0 寄存器 （并且 d = 1）执行该指令时，如果已将预分频器分配给了 TMR0，则将其清零。

3： 如果程序计数器 （PC）被修改或者条件检测为 true，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存储的都是合法的指令。

表 29-2： PIC18F87K22 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	立即数与 WREG 作逻辑与运算	1	0000 1011 kkkk kkkk	Z, N	
IORLW k	立即数与 WREG 作逻辑或运算	1	0000 1001 kkkk kkkk	Z, N	
LFSR f, k	将立即数 （12 位）第二个字	2	1110 1110 00ff kkkk	无	
	移入 FSR(f) 第一个字		1111 0000 kkkk kkkk		
MOVLB k	将立即数移入 BSR<3:0>	1	0000 0001 0000 kkkk	无	
MOVLW k	将立即数移入 WREG	1	0000 1110 kkkk kkkk	无	
MULLW k	立即数与 WREG 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	立即数与 WREG 作逻辑异或运算	1	0000 1010 kkkk kkkk	Z, N	
数据存储器 ↔ 程序存储器操作类指令					
TBLRD*	表读	2	0000 0000 0000 1000	无	
TBLRD*+	后增表读		0000 0000 0000 1001	无	
TBLRD*-	后减表读		0000 0000 0000 1010	无	
TBLRD+*	预增表读		0000 0000 0000 1011	无	
TBLWT*	表写	2	0000 0000 0000 1100	无	
TBLWT*+	后增表写		0000 0000 0000 1101	无	
TBLWT*-	后减表写		0000 0000 0000 1110	无	
TBLWT+*	预增表写		0000 0000 0000 1111	无	

注 1： PORT 寄存器的值随端口功能的变化而不断修改 （例如， MOVF PORTB, 1, 0），修改时使用的值是引脚上的当前值。例如，如果将一个引脚配置为输入，虽然其数据锁存器中的值为 1，但如果此时外部器件将该引脚驱动为低电平，则被写回数据总线的数据值将是 0。

2： 当对 TMR0 寄存器 （并且 d = 1）执行该指令时，如果已将预分频器分配给了 TMR0，则将其清零。

3： 如果程序计数器 （PC）被修改或者条件检测为 true，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存储的都是合法的指令。

© 2011 Microchip Technology Inc.

29.1.1 标准指令集

ADDLW **W 与立即数相加**

语法： ADDLW k

操作数： $0 \le k \le 255$

操作： (W) + k → W

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0000	1111	kkkk	kkkk

说明： W 寄存器中的内容与 8 位立即数 k 相加，结果保存在 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 寄存器相加**

语法： ADDWF f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

操作： (W) + (f) → 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 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 $f \le 95$ （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字节和位操作类指令”**。

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，那么指令语法将变为：{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、进位标志位和数据存储单元 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字节和位操作类指令”**。

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 = 0C2h
W = 4Dh

执行指令后
进位标志位 = 0
REG = 0C2h
W = 50h

ANDLW **立即数与 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： W 的内容与 8 位立即数 k 作逻辑与运算。结果保存在 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

© 2011 Microchip Technology Inc.

ANDWF 将 W 与 f 作逻辑与运算

语法： ANDWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) .AND. (f) → 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 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字节和位操作类指令”**。

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，那么程序将跳转。

二进制补码 2n 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址都将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

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字节和位操作类指令”**。

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 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，那么程序将跳转。

二进制补码 2n 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址都将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

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）

© 2011 Microchip Technology Inc.

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，那么程序将跳转。 二进制补码 2n 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址都将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。
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，那么程序将跳转。 二进制补码 2n 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址都将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。
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，那么程序将跳转。 二进制补码 2n 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址都将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。
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，那么程序将跳转。 二进制补码 2n 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址都将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。
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）

© 2011 Microchip Technology Inc.

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： 将二进制补码 2n 与 PC 相加。因为 PC 要先递增才能取下一条指令，所以新地址都将为 PC + 2 + 2n。该指令是一条双周期指令。

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 中的位 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字节和位操作类指令”**。

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 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 检测寄存器中的位，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 则跳转

受影响的状态位： 无

机器码：

1011	bbba	ffff	ffff

说明： 如果寄存器 f 中的位 b 为 0，则跳过下一条指令。即在位 b 为 0 时，丢弃下一条指令（执行当前指令期间取的指令）而执行一条 NOP 指令，使该指令变成双周期指令。

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字节和位操作类指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    BTFSC    FLAG, 1, 0
FALSE   :
TRUE    :
```

执行指令前
PC = 地址（HERE）
执行指令后
如果 FLAG<1> = 0；
PC = 地址（TRUE）
如果 FLAG<1> = 1；
PC = 地址（FALSE）

BTFSS 检测寄存器中的位，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 则跳转

受影响的状态位： 无

机器码：

1010	bbba	ffff	ffff

说明： 如果寄存器 f 中的位 b 为 1，则跳过下一条指令。即在位 b 为 1 时，丢弃下一条指令（执行当前指令期间取的指令）而执行一条 NOP 指令，使该指令变成双周期指令。

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字节和位操作类指令”**。

指令字数： 1

指令周期数： 1（2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    BTFSS    FLAG, 1, 0
FALSE   :
TRUE    :
```

执行指令前
PC = 地址（HERE）
执行指令后
如果 FLAG<1> = 0；
PC = 地址（FALSE）
如果 FLAG<1> = 1；
PC = 地址（TRUE）

© 2011 Microchip Technology Inc.

BTG 将 f 中的某位取反

语法： BTG f, b {,a}

操作数： $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

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字节和位操作类指令”**。

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 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： $-128 \le n \le 127$

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，那么程序将跳转。

二进制补码 2n 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址都将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

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，那么程序将跳转。

二进制补码 2n 与 PC 相加。因为 PC 要先递增以便取出下一条指令，所以新地址都将为 PC + 2 + 2n。这种情况下，该指令是一条双周期指令。

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。然后，将 20 位的值 k 装入 PC<20:1>。CALL 指令是一条双周期指令。

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

© 2011 Microchip Technology Inc.

CLRF **将 f 清零**

语法： CLRF f {,a}

操作数： $0 \le f \le 255$
$a \in [0,1]$

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 $f \le 95$ （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字节和位操作类指令”**。

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 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 → $\overline{PD}$

受影响的状态位： $\overline{TO}$ 和 $\overline{PD}$

机器码：

0000	0000	0000	0100

说明： CLRWDT 指令复位看门狗定时器。而且还会复位 WDT 的后分频器。状态位 $\overline{TO}$ 和 $\overline{PD}$ 被置 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
$\overline{PD}$ = 1

COMF — 将 f 取反

语法： COMF f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

操作： $\overline{f} \rightarrow$ 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 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 $f \le 95$ （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字节和位操作类指令”。

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： $0 \le f \le 255$
$a \in [0,1]$

操作： (f) – (W)，
如果 (f) = (W)，则跳过 （无符号比较）

受影响的状态位： 无

机器码：

0110	001a	ffff	ffff

说明： 通过执行无符号减法，将数据存储单元 f 的内容与 W 的内容做比较。

如果 f = W，则所取的指令被丢弃，转而执行一条 NOP 指令，从而使该指令变成双周期指令。

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 $f \le 95$ （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字节和位操作类指令”。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE      CPFSEQ REG, 0
NEQUAL    :
EQUAL     :
```

执行指令前
PC 地址 = HERE
W = ?
REG = ?
执行指令后
如果 REG = W；
PC = 地址 （EQUAL）
如果 REG ≠ W；
PC = 地址 （NEQUAL）

© 2011 Microchip Technology Inc.

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： 通过执行无符号减法，将数据存储单元 f 的内容与 W 的内容做比较。

如果 f 的内容大于 WREG 的内容，则所取的指令会被丢弃，转而执行一条 NOP 指令，从而使该指令变成双周期指令。

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字节和位操作类指令”**。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE       CPFSGT REG, 0
NGREATER   :
GREATER    :
```

执行指令前
PC = 地址 （HERE）
W = ?
执行指令后
如果 REG > W；
PC = 地址 （GREATER）
如果 REG ≤ W；
PC = 地址 （NGREATER）

CPFSLT 比较 f 和 W，如果 f < W 则跳过

语法： CPFSLT f {,a}

操作数： 0 ≤ 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： 通过执行无符号减法，将数据存储单元 f 的内容与 W 的内容做比较。

如果 f 的内容小于 W 的内容，则所取指令会被丢弃，转而执行一条 NOP，使该指令变成双周期指令。

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： 如果跳过的指令后面跟有双字指令，则执行该指令需要 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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     CPFSLT REG, 1
NLESS    :
LESS     :
```

执行指令前
PC = 地址 （HERE）
W = ?
执行指令后
如果 REG < W；
PC = 地址 （LESS）
如果 REG ≥ W；
PC = 地址 （NLESS）

DAW — 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]，那么
(W<3:0>) + 6 → W<3:0>；
否则
(W<3:0>) → W<3:0>；

如果 [W<7:4> > 9] 或 [C = 1]，那么
(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 内的 8 位数值，即前两个压缩 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	写入 W

示例 1： DAW

执行指令前
W = A5h
C = 0
DC = 0
执行指令后
W = 05h
C = 1
DC = 0

示例 2：

执行指令前
W = CEh
C = 0
DC = 0
执行指令后
W = 34h
C = 1
DC = 0

DECF — f 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 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字节和位操作类指令”**。

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

© 2011 Microchip Technology Inc.

DECFSZ — f 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 中。 如果结果为 0，则丢弃已取的下一条指令，而执行一条 NOP 指令，使该指令变成双周期指令。 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字节和位操作类指令”。
指令字数：	1
指令周期数：	1 （2） 注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE      DECFSZ   CNT, 1, 1
          GOTO     LOOP
CONTINUE
```

执行指令前
PC = 地址 （HERE）
执行指令后
CNT = CNT – 1
如果 CNT = 0；
PC = 地址 （CONTINUE）
如果 CNT ≠ 0；
PC = 地址 （HERE + 2）

DCFSNZ — f 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 中。 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，使该指令变成双周期指令。 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字节和位操作类指令”。
指令字数：	1
指令周期数：	1 （2） 注： 如果跳过的指令后面跟有双字指令，则执行该指令需要 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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE     DCFSNZ   TEMP, 1, 0
ZERO     :
NZERO    :
```

执行指令前
TEMP = ?
执行指令后
TEMP = TEMP – 1,
如果 TEMP = 0；
PC = 地址 （ZERO）
如果 TEMP ≠ 0；
PC = 地址 （NZERO）

GOTO 无条件跳转

语法： GOTO k

操作数： $0 \le k \le 1048575$

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 位的 k 值装入 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 加 1

语法： INCF f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

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 的内容加 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 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 $f \le 95$ （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字节和位操作类指令”**。

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

© 2011 Microchip Technology Inc.

INCFSZ **f 加 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 则跳转

受影响的状态位： 无

机器码：

0011	11da	ffff	ffff

说明： 将寄存器 f 的内容加 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 中。

如果结果为 0，则丢弃已取的下一条指令，而执行一条 NOP，使该指令变成双周期指令。

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字节和位操作类指令”**。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    INCFSZ   CNT, 1, 0
NZERO   :
ZERO    :
```

执行指令前
PC = 地址 （HERE）
执行指令后
CNT = CNT + 1
如果 CNT = 0；
PC = 地址 （ZERO）
如果 CNT ≠ 0；
PC = 地址 （NZERO）

INFSNZ **f 加 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 的内容加 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 中。

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，使该指令变成双周期指令。

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字节和位操作类指令”**。

指令字数： 1

指令周期数： 1 （2）
注： 如果跳过的指令后面跟有双字指令，则执行该指令需要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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    INFSNZ   REG, 1, 0
ZERO
NZERO
```

执行指令前
PC = 地址 （HERE）
执行指令后
REG = REG + 1
如果 REG ≠ 0；
PC = 地址 （NZERO）
如果 REG = 0；
PC = 地址 （ZERO）

IORLW 立即数与 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：	W 的内容与 8 位立即数 k 作逻辑或运算。结果保存在 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 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 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字节和位操作类指令”**。
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

© 2011 Microchip Technology Inc.

LFSR **装入 FSR**

语法： LFSR f, k

操作数： $0 \le f \le 2$
$0 \le k \le 4095$

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 位的立即数 k 装入由 f 指向的文件选择寄存器。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 k 的高字节	处理数据	将立即数 k 的高字节写入 FSRfH
译码	读立即数 k 的低字节	处理数据	将立即数 k 的低字节写入 FSRfL

示例： `LFSR 2, 3ABh`

执行指令后
FSR2H = 03h
FSR2L = ABh

MOVF **移动 f**

语法： MOVF f {,d {,a}}

操作数： $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

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 的内容移入目标单元。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 中。f 可以为 256 字节存储区中的任何单元。

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 $f \le 95$ （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字节和位操作类指令”**。

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 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 **将源寄存器的内容移入目标寄存器**

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 的内容移入目标寄存器 f_d。源寄存器 f_s 可以是 4096 字节数据空间（000h 到 FFFh）中的任何单元，目标寄存器 f_d 也可以是从 000h 到 FFFh 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移入外设寄存器（如发送缓冲器或 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 非无效读取	空操作	写入 寄存器 f（目标寄存器）

示例： `MOVFF REG1, REG2`

执行指令前
REG1 = 33h
REG2 = 11h
执行指令后
REG1 = 33h
REG2 = 33h

MOVLB **将立即数移入 BSR 的低半字节**

语法： MOVLB 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）。BSR<7:4> 的值始终保持为 0，不管 k_7:k_4 的值如何。

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

© 2011 Microchip Technology Inc.

MOVLW **将立即数移入 W**

语法： MOVLW k

操作数： $0 \le k \le 255$

操作： $k \rightarrow W$

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 的内容移入 f**

语法： MOVWF f {,a}

操作数： $0 \le f \le 255$
$a \in [0,1]$

操作： $(W) \rightarrow f$

受影响的状态位： 无

机器码：

0110	111a	ffff	ffff

说明： 将 W 寄存器的数据移入寄存器 f。f 可以为 256 字节存储区中的任何单元。

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 $f \le 95$ （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字节和位操作类指令”**。

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 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 的内容相乘**

语法： MULLW k

操作数： $0 \le k \le 255$

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 的内容和 8 位立即数 k 进行无符号的乘法运算。 16 位结果存储在 PRODH:PRODL 寄存器对中。 PRODH 存储高字节。

W 的内容不变。

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： $0 \le f \le 255$
$a \in [0,1]$

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 的内容和寄存器单元 f 的内容执行无符号的乘法运算。 16 位结果存储在 PRODH:PRODL 寄存器对中。 PRODH 存储高字节。 W 和 f 的内容不变。

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字节和位操作类指令”**。

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

© 2011 Microchip Technology Inc.

NEGF 对 f 取补

语法： NEGF f {,a}

操作数： $0 \le f \le 255$
$a \in [0,1]$

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 $f \le 95$ （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字节和位操作类指令”**。

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： 空操作

受影响的状态位： 无

机器码：

0000 1111	0000 xxxx	0000 xxxx	0000 xxxx

说明： 空操作

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) → 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 = 新值

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

© 2011 Microchip Technology Inc.

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： 从当前地址单元（最多 1 KB）来调用子程序。首先，将返回地址（PC + 2）压入返回堆栈。将二进制补码 2n 与 PC 相加。因为 PC 要先递增才能取下一条指令，因此新地址将为 PC + 2 + 2n。该指令是一条双周期指令。

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 $\overline{\text{MCLR}}$ 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	启动复位	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。执行出栈操作时，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。
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 和 PEIE/GIEL =  1
```

RETLW 返回时将立即数送入 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。高位地址锁存器（PCLATH）的内容保持不变。
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
```


© 2011 Microchip Technology Inc.

RETURN 从子程序返回

语法： RETFIE {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，将影子寄存器 WS、 STAUTSS 和 BSRS 的内容装入相应的寄存器 W、STATUS 和 BSR。如果 s = 0，则不更新这些寄存器。

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

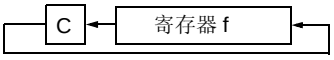
机器码：

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 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字节和位操作类指令”**。



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

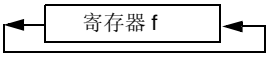
机器码：

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 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字节和位操作类指令”。



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

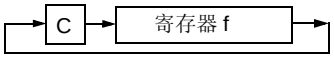
机器码：

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 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字节和位操作类指令”。



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
```


© 2011 Microchip Technology Inc.

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

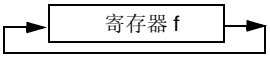
机器码：

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 中。

如果 a 为 0，选择快速操作存储区，覆盖 BSR 的值。如果 a 为 1，使用 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字节和位操作类指令”**。



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 1： `RRNCF REG, 1, 0`

执行指令前
REG = 1101 0111
执行指令后
REG = 1110 1011

示例 2： `RRNCF REG, 0, 0`

执行指令前
W = ?
REG = 1101 0111
执行指令后
W = 1110 1011
REG = 1101 0111

SETF **将 f 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字节和位操作类指令”**。

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 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 → $\overline{PD}$

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，则此位将被清零。

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 的内容减去寄存器 f 和进位（借位）（通过二进制补码方法进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存储在寄存器 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字节和位操作类指令”**。

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 1： SUBFWB REG, 1, 0

执行指令前
REG = 3
W = 2
C = 1

执行指令后
REG = FF
W = 2
C = 0
Z = 0
N = 1 ；结果为负

示例 2： SUBFWB REG, 0, 0

执行指令前
REG = 2
W = 5
C = 1

执行指令后
REG = 2
W = 3
C = 1
Z = 0
N = 0 ；结果为正

示例 3： SUBFWB REG, 1, 0

执行指令前
REG = 1
W = 2
C = 0

执行指令后
REG = 0
W = 2
C = 1
Z = 1 ；结果为零
N = 0

© 2011 Microchip Technology Inc.

SUBLW 立即数减去 W

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。结果保存在 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 1： SUBLW 02h

```
执行指令前
    W    =  01h
    C    =  ?
执行指令后
    W    =  01h
    C    =  1     ；结果为正
    Z    =  0
    N    =  0
```

示例 2： SUBLW 02h

```
执行指令前
    W    =  02h
    C    =  ?
执行指令后
    W    =  00h
    C    =  1     ；结果为零
    Z    =  1
    N    =  0
```

示例 3： SUBLW 02h

```
执行指令前
    W    =  03h
    C    =  ?
执行指令后
    W    =  FFh   ；（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 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字节和位操作类指令”**。

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 1： SUBWF REG, 1, 0

```
执行指令前
    REG  =  3
    W    =  2
    C    =  ?
执行指令后
    REG  =  1
    W    =  2
    C    =  1     ；结果为正
    Z    =  0
    N    =  0
```

示例 2： SUBWF REG, 0, 0

```
执行指令前
    REG  =  2
    W    =  2
    C    =  ?
执行指令后
    REG  =  2
    W    =  0
    C    =  1     ；结果为零
    Z    =  1
    N    =  0
```

示例 3： SUBWF REG, 1, 0

```
执行指令前
    REG  =  1
    W    =  2
    C    =  ?
执行指令后
    REG  =  FFh   ；（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) – ($\overline{C}$) → 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（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 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字节和位操作类指令”**。

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 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
```

示例 2： `SUBWFB REG, 0, 0`

```
执行指令前
    REG  =  1Bh    (0001 1011)
    W    =  1Ah    (0001 1010)
    C    =  0
执行指令后
    REG  =  1Bh    (0001 1011)
    W    =  00h
    C    =  1
    Z    =  1      ; 结果为零
    N    =  0
```

示例 3： `SUBWFB REG, 1, 0`

```
执行指令前
    REG  =  03h    (0000 0011)
    W    =  0Eh    (0000 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相交换**

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： 将寄存器 f 的高半字节和低半字节互相交换。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。

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字节和位操作类指令”**。

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
```


© 2011 Microchip Technology Inc.

TBLRD 表读

语法： TBLRD (*; *+; *-; +*)

操作数： 无

操作： 如果 TBLRD *，
(程序存储器 (TBLPTR)) → TABLAT，
TBLPTR 不改变；
如果 TBLRD *+，
(程序存储器 (TBLPTR)) → TABLAT，
(TBLPTR) + 1 → TBLPTR；
如果 TBLRD *-，
(程序存储器 (TBLPTR)) → TABLAT，
(TBLPTR) – 1 → TBLPTR；
如果 TBLRD +*，
(TBLPTR) + 1 → TBLPTR，
(程序存储器 (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器 （P.M.）的内容。要对程序存储器进行寻址，需要使用表指针 （TBLPTR）。

TBLPTR （一个 21 位指针）可以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：程序存储器字的低有效字节

TBLPTR<0> = 1：程序存储器字的高有效字节

TBLRD 指令可以按如下方法来修改 TBLPTR 的值：

- 不变
- 后增
- 后减
- 预增

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）

示例 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

示例 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 TBLWT*，
(TABLAT) → 保持寄存器，
TBLPTR——不改变；
如果 TBLWT*+，
(TABLAT) → 保持寄存器，
(TBLPTR) + 1 → TBLPTR；
如果 TBLWT*-，
(TABLAT) → 保持寄存器，
(TBLPTR) – 1 → TBLPTR；
如果 TBLWT+*，
(TBLPTR) + 1 → TBLPTR，
(TABLAT) → 保持寄存器

受影响的状态位： 无

机器码：

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

说明： 此指令使用 TBLPTR 的低 3 个字节来确定要将 TABLAT 写入 8 个保持寄存器中的哪一个。保持寄存器用于对程序存储器（P.M.）的内容编程。（关于对闪存程序存储器编程的更多详情，请参见**第 6.0 节 “存储器构成”**。）

TBLPTR （一个 21 位指针）可以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 选择要访问的程序存储单元字节。

TBLPTR[0] = 0：程序存储器字的低有效字节

TBLPTR[0] = 1：程序存储器字的高有效字节

TBLWT指令可以按如下方法来修改TBLPTR的值：

- 不变
- 后增
- 后减
- 预增

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（写入保持寄存器）

TBLWT 表写 （续）

示例 1： TBLWT *+;

执行指令前
TABLAT = 55h
TBLPTR = 00A356h
保持寄存器 (00A356h) = FFh
执行指令后 （表写操作完成）
TABLAT = 55h
TBLPTR = 00A357h
保持寄存器 (00A356h) = 55h

示例 2： TBLWT +*;

执行指令前
TABLAT = 34h
TBLPTR = 01389Ah
保持寄存器 (01389Ah) = FFh
保持寄存器 (01389Bh) = FFh
执行指令后 （表写操作完成）
TABLAT = 34h
TBLPTR = 01389Bh
保持寄存器 (01389Ah) = FFh
保持寄存器 (01389Bh) = 34h

© 2011 Microchip Technology Inc.

TSTFSZ 测试 f，为 0 则跳过

语法： TSTFSZ f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： 如果 f = 0 则跳转

受影响的状态位： 无

机器码：

0110	011a	ffff	ffff

说明： 如果 f = 0，丢弃下一条指令（执行当前指令期间取的指令）而执行一条 NOP 指令，使该指令变成双周期指令。

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字节和位操作类指令”。

指令字数： 1

指令周期数： 1（2）

注： 如果跳过的指令后面跟有双字指令，则执行该指令需要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

如果跳过的指令后面跟有双字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```

执行指令前
PC = 地址（HERE）
执行指令后
如果 CNT = 00h，
PC = 地址（ZERO）
如果 CNT ≠ 00h，
PC = 地址（NZERO）

XORLW 立即数与 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： W 的内容与 8 位立即数 k 作逻辑异或运算。结果保存在 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	**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：	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。 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字节和位操作类指令”**。
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

© 2011 Microchip Technology Inc.

29.2 扩展指令集

除了PIC18指令集的75条标准指令之外，PIC18F87K22系列器件还提供针对CPU内核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，它们可以实现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。

默认情况下，这些扩展指令集的额外功能在未编程器件上是使能的。用户必须在编程过程中将 XINST 配置位置 1 或清零来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，它们既可以控制文件选择寄存器，也可以使用这些寄存器进行变址寻址。其中两条指令ADDFSR和SUBFSR的一种特殊用法是直接对 FSR2 进行操作，而 ADDULNK 和 SUBULNK 指令允许在执行后自动返回。

这些扩展指令专门用于优化用高级语言特别是C语言编写的重入程序代码（也就是递归调用或使用软件堆栈的代码）。此外，它们使用户能更有效地用高级语言对数据结构执行特定的操作。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控制
- 对软件堆栈中的变量进行控制

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 432 页）提供了标准和扩展的 PIC18 指令集的操作码字段说明。

> **注：** 扩展的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器中直接使用这些指令。对于那些查看编译器生成的代码的用户，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，同时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（“[ ]”）把它括起来。这时表示此参数用作变址地址或偏移量。如果 MPASM™ 汇编器发现一个变址地址或偏移量没有被括起来，它就会给出错误信息。

当使能扩展指令集时，括号也用于表示字节和位操作类的指令中的变址参数。这是对指令语法的额外更改。欲知更多详情，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。在本文和以后的文章中，可选参数将用大括号（“{ }”）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 MSb			LSb	受影响的状态位
ADDFSR	f, k	将立即数与 FSR 相加	1	1110	1000	ffkk	kkkk	无
ADDULNK	k	将立即数与 FSR2 相加并返回	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 f_d（目标）移入第二个字	2	1110 1111	1011 ffff	0zzz ffff	zzzz ffff	无
MOVSS	z_s, z_d	将 z_s（源）移入第一个字 z_d（目标）移入第二个字	2	1110 1111	1011 xxxx	1zzz xzzz	zzzz zzzz	无
PUSHL	k	将立即数保存在 FSR2 后， FSR2 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 **将立即数与 FSR 相加**

语法：	ADDFSR f, k
操作数：	0 ≤ k ≤ 63 f ∈ [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：	将由 f 指定的 FSR 的内容加上一个 6 位的立即数 k。
指令字数：	1
指令周期数：	1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 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 **将立即数与 FSR2 相加并返回**

语法：	ADDULNK 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 1000 11kk kkkk
说明：	将 FSR2 的内容加上一个 6 位立即数 k。然后通过将 TOS 装入 PC，执行一条 RETURN 指令。 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，那么指令语法将变为：{label} 指令参数。

© 2011 Microchip Technology Inc.

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，并同时取下一条指令。

和 CALL 指令不一样，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移入 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： 源寄存器的内容移入目标寄存器 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 字节数据空间 （000h 到 FFFh）中的任何单元。

MOVSF 指令中的目标寄存器不能是 PCL、 TOSU、 TOSH 或 TOSL。

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 非无效读取	空操作	写入寄存器 f （目标寄存器）

示例： MOVSF [05h], REG2

执行指令前
FSR2 = 80h
85h 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 **变址寻址传送数据**

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 或 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 到 FFFh）中的任意单元。

MOVSS 指令中的目标寄存器不能是 PCL、 TOSU、 TOSH 或 TOSL。

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，指令将作为一条 NOP 指令执行。

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	确定 目标地址	确定 目标地址	写入 目标寄存器

示例： MOVSS [05h], [06h]

执行指令前
FSR2 = 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 **将立即数保存在 FSR2 后， FSR2 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：

1110	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。操作完成后 FSR2 减 1。
该指令允许用户将值压入软件堆栈。

指令字数： 1

指令周期数： 1

Q 周期操作：

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

© 2011 Microchip Technology Inc.

SUBFSR **FSR 减去立即数**

语法： SUBFSR f, k

操作数： $0 \le k \le 63$

$f \in [0, 1, 2]$

操作： FSRf – k → FSRf

受影响的状态位： 无

机器码：

1110	1001	ffkk	kkkk

说明： 用寄存器 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： $0 \le k \le 63$

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 FSR2 的内容减去 6 位立即数 k。然后通过将 TOS 装入 PC，执行一条 RETURN 指令。

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字节和位操作类指令

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 8 条新指令之外，还可以使用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”**）。这将导致很多标准 PIC18 指令的地址解析方法有很大变化。

当禁用扩展指令集时，被嵌入在操作码中的地址被视作立即数存储单元：可以是快速操作存储区中的单元（a = 0），或由 BSR 指定的 GPR 存储区中的单元（a = 1）。当使能扩展指令集且 a = 0 时，地址为 5Fh 或以下的文件选择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（即所有字节或位操作类指令或几乎半数的 PIC18 内核指令）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，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 程序中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控制很有用处，但是如果不小心在简单的算术运算中误用了寄存器，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。

下面是在立即数变址寻址模式中，一些典型的字节和位操作类指令示例，通过示例可以看出指令如何受到影响。示例中的操作数条件适用于所有这一类的指令。

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 被用来替换标准的字节和位操作类命令中的文件选择寄存器参数 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，该偏移量必须用方括号（“[ ]”）标出。因为在扩展指令集中，方括号指示编译器将括号中的数值解析为变址地址或偏移量。省略括号，或在括号内使用大于 5Fh 的值会在 MPASM™ 汇编器中产生错误。

如果变址寻址模式下的变址参数已被正确的加上括号，那么就不再需要指定快速操作 RAM 参数；自动将此参数假定为 0。在 a 根据目标地址置 1 时，这与标准操作（禁止扩展指令集）正好相反。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，必须明确调用对扩展的指令集的语言支持。可以通过命令行选项 /y 或在源代码中加入 PE 伪指令进行调用。

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编写代码的用户，使用扩展指令集并不会获得益处。

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。这是因为常规代码中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，这些地址会被解析为相对于 FSR2 的立即数偏移量，所以应用程序会读或写错误的地址。

将应用程序移植到 PIC18F87K22 系列器件时，代码的类型是非常重要的。在使用扩展指令集时，代码较长的重入应用程序（用 C 语言编写且可高效编译）会运行的很好，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。

© 2011 Microchip Technology Inc.

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： $0 \le k \le 95$
$d \in [0,1]$

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。

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： $0 \le f \le 95$
$0 \le b \le 7$

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： $0 \le k \le 95$

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	写入寄存器

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 MICROCHIP MPLAB® IDE 工具的特别注意事项

最新版本的 Microchip 软件工具完全支持 PIC18F87K22 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（IDE）。

在选择了要针对之开发软件的目标器件后， MPLAB IDE 将把该器件的默认配置位自动置 1。XINST 配置位的默认设置是 0，即禁止扩展指令集和立即数变址寻址。在编程过程中必须将 XINST 位置 1 才能正确地利用扩展指令集开发应用程序。

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实现：

- 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信息。

© 2011 Microchip Technology Inc.

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

© 2011 Microchip Technology Inc.

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 V_{DDMIN} 和 V_{DDMAX} 点对其可编程电压进行校验以确保可靠性最高。它有一个用来显示菜单和错误消息的大 LCD 显示器（128 x 64），以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 ICSP™ 电缆。在单机模式下，MPLAB PM3 器件编程器不必与 PC 相连即可对 PIC 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）。

© 2011 Microchip Technology Inc.

31.0 电气特性

绝对最大值 (†)

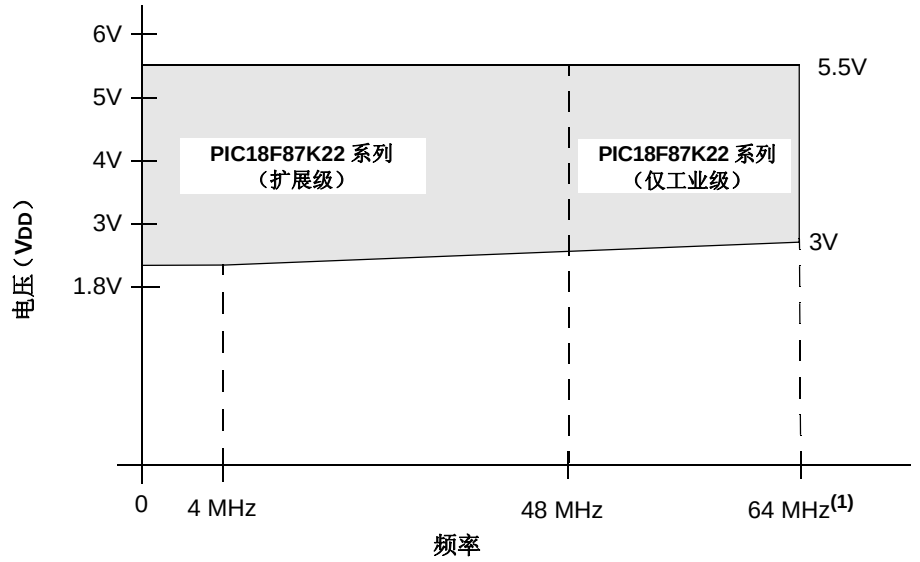
偏置电压下的环境温度 ..... -40°C 至 +125°C
储存温度 ..... -65°C 至 +150°C
任何仅数字 I/O 引脚 （V_{DD} 除外）相对于 V_{SS} 的电压 ..... -0.3V 至 7.5V
$\overline{MCLR}$ 相对于 V_{SS} 的电压 ..... 0.3V 至 9.0V
任何模数组合引脚 （V_{DD} 和 $\overline{MCLR}$ 除外）相对于 V_{SS} 的电压 ..... -0.3V 至 （V_{DD} + 0.3V）
V_{DD} 相对于 V_{SS} 的电压 （使能稳压器） ..... -0.3V 至 5.5V
V_{DD} 相对于 V_{SS} 的电压 （禁止稳压器） ..... -0.3V 至 3.6V
总功耗 （注 1） 1W
V_{SS} 引脚最大输出电流 ..... 300 mA
V_{DD} 引脚最大输入电流 ..... 250 mA
输入钳位电流 I_{IK} （V_I < 0 或 V_I > V_{DD}） ..... ±20 mA
输出钳位电流 I_{OK} （V_O < 0 或 V_O > V_{DD}） ..... ±20 mA
PORTA<7:6> 以及任何 PORTB 和 PORTC I/O 引脚的最大输出灌电流 ..... 25 mA
任何 PORTD、 PORTE 和 PORTJ I/O 引脚的最大输出灌电流 ..... 8 mA
PORTA<5:0> 以及任何 PORTF、 PORTG 和 PORTH I/O 引脚的最大输出灌电流 ..... 2 mA
PORTA<7:6> 以及任何 PORTB 和 PORTC I/O 引脚的最大输出拉电流 ..... 25 mA
任何 PORTD、 PORTE 和 PORTJ I/O 引脚的最大输出拉电流 ..... 8 mA
PORTA<5:0> 以及任何 PORTF、 PORTG 和 PORTH I/O 引脚的最大输出拉电流 ..... 2 mA
所有组合端口的最大灌电流 ..... 200 mA

注 1： 功耗按如下公式计算：

$$Pdis = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum(V_{OL} \times I_{OL})$$

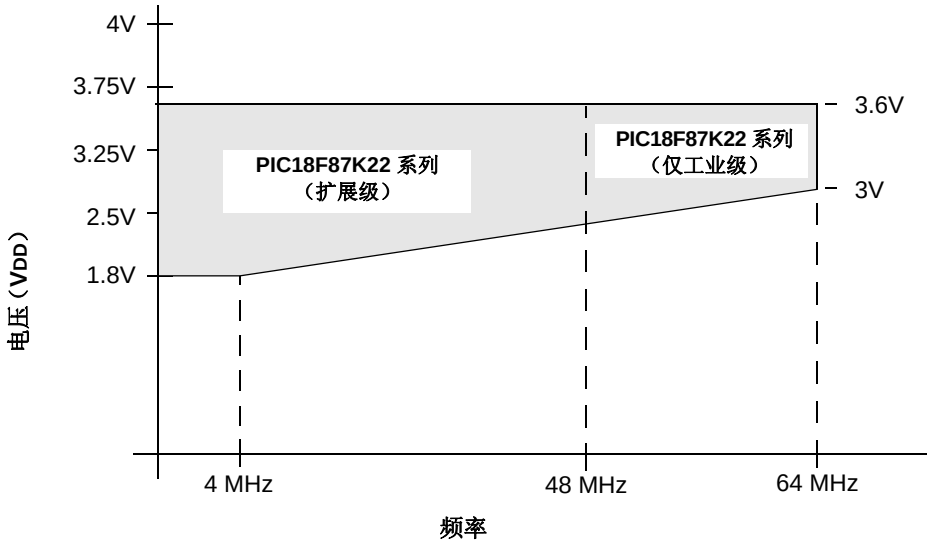
† 注：如果器件工作条件超过上述 “绝对最大值”，可能会对器件造成永久性损坏。上述值仅为运行条件极大值，我们不建议器件在该规范规定的范围以外运行。器件长时间工作在最大值条件下，其稳定性会受到影响。

图 31-1： 电压——频率关系图 （使能稳压器，工业级 / 扩展级）(1)



注 1： 在 8 位外部存储器模式下， FMAX = 25 MHz。如果 VDD 值为 1.8V 至 3V， FMAX = (VDD – 1.72)/0.02 MHz。
2： 在所有其他模式下，FMAX = 64 MHz。如果 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 必须保持原来的状态，从而保证 VDD ≤ 3.6V。
2： 如果 VDD 值为 1.8V 至 3V， FMAX = (VDD – 1.72)/0.02 MHz。

© 2011 Microchip Technology Inc.

31.1 直流特性： 供电电压 PIC18F87K22 系列 （工业级 / 扩展级）

PIC18F87K22 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	ENVREG 连接到 VSS ENVREG 连接到 VDD
D001C	AVDD	模拟电源	VDD – 0.3	—	VDD + 0.3	V	
D001D	AVSS	模拟地的电势	VSS – 0.3	—	VSS + 0.3	V	
D002	VDR	**RAM 数据保存电压** (1)	1.5	—	—	V	
D003	VPOR	**VDD 启动电压** 确保产生内部上电复位信号	—	—	0.7	V	详情请参见第 **5.3** 节 “上电复位 （**POR**）”
D004	SVDD	**VDD 上升率** 确保产生内部上电复位信号	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 / 中等 / 低功耗模式） (2) BORV<1:0> = 11 (3) 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		

注 1： 该电压是休眠模式或器件复位状态下，在不丢失 RAM 数据的前提下的最小 VDD。

2： 下面的值是在 HP-BOR 模式下获得的。

3： 尽管 VDD 可能低于 VDDMIN，但是在发生欠压复位前器件仍将正常工作。

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）

PIC18F87K22 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)					
	所有器件	10	500	nA	-40°C	VDD = 1.8V(4) （休眠模式） 禁止稳压器
		20	500	nA	+25°C	
		120	600	nA	+60°C	
		630	1800	nA	+85°C	
		4	9	µA	+125ºC	
	所有器件	50	700	nA	-40°C	VDD = 3.3V(4) （休眠模式） 禁止稳压器
		60	700	nA	+25°C	
		170	800	nA	+60°C	
		700	2700	nA	+85°C	
		5	11	µA	+125ºC	
	所有器件	350	1300	nA	-40°C	VDD = 5V(5) （休眠模式） 使能稳压器
		400	1400	nA	+25°C	
		550	1500	nA	+60°C	
		1350	4000	nA	+85°C	
		6	12	µA	+125ºC	

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD ；
MCLR = VDD ； 根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

© 2011 Microchip Technology Inc.

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）（续）

PIC18F87K22 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)						
	所有器件	5.3	10	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 31 kHz （RC_RUN 模式、LF-INTOSC）
		5.5	10	μA	+25°C		
		5.5	10	μA	+85°C		
		12	24	μA	+125ºC		
	所有器件	10	15	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		10	16	μA	+25°C		
		11	17	μA	+85°C		
		15	35	μA	+125ºC		
	所有器件	70	180	μA	-40°C	VDD = 5V(5) 使能稳压器	
		80	185	μA	+25°C		
		90	190	μA	+85°C		
		200	500	μA	+125ºC		
	所有器件	410	850	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 1 MHz （RC_RUN 模式、HF-INTOSC)
		410	800	μA	+25°C		
		410	830	μA	+85°C		
		700	1500	μA	+125ºC		
	所有器件	680	990	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		680	960	μA	+25°C		
		670	950	μA	+85°C		
		800	1700	μA	+125ºC		
	所有器件	760	1400	μA	-40°C	VDD = 5V(5) 使能稳压器	
		780	1400	μA	+25°C		
		800	1500	μA	+85°C		
		1200	2400	μA	+125ºC		
	所有器件	760	1300	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 4 MHz （RC_RUN 模式、HF-INTOSC)
		760	1400	μA	+25°C		
		770	1500	μA	+85°C		
		800	1700	μA	+125ºC		
	所有器件	1.4	2.5	mA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		1.4	2.5	mA	+25°C		
		1.4	2.5	mA	+85°C		
		1.5	3.0	mA	+125ºC		
	所有器件	1.5	2.7	mA	-40°C	VDD = 5V(5) 使能稳压器	
		1.5	2.7	mA	+25°C		
		1.5	2.7	mA	+85°C		
		1.6	3.3	mA	+125ºC		

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD ；
MCLR = VDD ；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）（续）

PIC18F87K22 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)						
	所有器件	2.1	5.5	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 31 kHz （**RC_IDLE** 模式、 LF-INTOSC）
		2.1	5.7	μA	+25°C		
		2.2	6.0	μA	+85°C		
		10	20	μA	+125ºC		
	所有器件	3.7	7.5	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		3.9	7.8	μA	+25°C		
		3.9	8.5	μA	+85°C		
		12	24	μA	+125ºC		
	所有器件	70	180	μA	-40°C	VDD = 5V(5) 使能稳压器	
		80	190	μA	+25°C		
		80	200	μA	+85°C		
		200	420	μA	+125ºC		
	所有器件	330	650	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 1 MHz （**RC_IDLE** 模式、 HF-INTOSC)
		330	640	μA	+25°C		
		330	630	μA	+85°C		
		500	850	μA	+125ºC		
	所有器件	520	850	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		520	900	μA	+25°C		
		520	850	μA	+85°C		
		800	1200	μA	+125ºC		
	所有器件	590	940	μA	-40°C	VDD = 5V(5) 使能稳压器	
		600	960	μA	+25°C		
		620	990	μA	+85°C		
		1000	1400	μA	+125ºC		
	所有器件	470	770	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 4 MHz （**RC_IDLE** 模式、 内部 HF-INTOSC)
		470	770	μA	+25°C		
		460	760	μA	+85°C		
		700	1000	μA	+125ºC		
	所有器件	800	1400	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		800	1350	μA	+25°C		
		790	1300	μA	+85°C		
		1100	1400	μA	+125ºC		
	所有器件	880	1600	μA	-40°C	VDD = 5V(5) 使能稳压器	
		890	1700	μA	+25°C		
		910	1800	μA	+85°C		
		1200	2200	μA	+125ºC		

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD；
MCLR = VDD；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

© 2011 Microchip Technology Inc.

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）（续）

PIC18F87K22 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)						
	所有器件	130	390	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 1 MHz （**PRI_RUN** 模式、 EC 振荡器）
		130	390	μA	+25°C		
		130	390	μA	+85°C		
		250	500	μA	+125ºC		
	所有器件	270	790	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		270	790	μA	+25°C		
		270	790	μA	+85°C		
		400	900	μA	+125ºC		
	所有器件	430	990	μA	-40°C	VDD = 5V(5) 使能稳压器	
		450	980	μA	+25°C		
		460	980	μA	+85°C		
		600	1300	μA	+125ºC		
	所有器件	430	860	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 4 MHz （**PRI_RUN** 模式、 EC 振荡器）
		530	900	μA	+25°C		
		490	880	μA	+85°C		
		750	1600	μA	+125ºC		
	所有器件	850	1750	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		850	1700	μA	+25°C		
		850	1800	μA	+85°C		
		1150	2400	μA	+125ºC		
	所有器件	1.1	2.7	mA	-40°C	VDD = 5V(5) 使能稳压器	
		1.1	2.6	mA	+25°C		
		1.1	2.6	mA	+85°C		
		2.0	4.0	mA	+125ºC		
	所有器件	12	19	mA	-40°C	VDD = 3.3V(4) 禁止稳压器	FOSC = 64 MHz （**PRI_RUN** 模式、 EC 振荡器）
		12	19	mA	+25°C		
		12	19	mA	+85°C		
		13	22	mA	+125ºC(6)		
	所有器件	13	20	mA	-40°C	VDD = 5V(4) 使能稳压器	
		13	20	mA	+25°C		
		13	20	mA	+85°C		
		14	23	mA	+125ºC(6)		

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD；
MCLR = VDD；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级） （续）

PIC18F87K22 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)						
	所有器件	3.3	5.6	mA	-40°C	VDD = 3.3V(4) 禁止稳压器	FOSC = 16 MHz （**PRI_RUN** 模式，带有 PLL 的 4 MHz EC 振荡器）
		3.3	5.5	mA	+25°C		
		3.3	5.5	mA	+85°C		
		3.6	6.0	mA	+125ºC		
	所有器件	3.5	5.9	mA	-40°C	VDD = 5V(5) 使能稳压器	
		3.5	5.8	mA	+25°C		
		3.5	5.8	mA	+85°C		
		3.8	7.0	mA	+125ºC		
	所有器件	12	18	mA	-40°C	VDD = 3.3V(4) 禁止稳压器	FOSC = 64 MHz （**PRI_RUN** 模式，带有 PLL 的 16 MHz EC 振荡器）
		12	18	mA	+25°C		
		12	18	mA	+85°C		
		13	22	mA	+125ºC(6)		
	所有器件	13	20	mA	-40°C	VDD = 5V(5) 使能稳压器	
		13	20	mA	+25°C		
		13	20	mA	+85°C		
		14	24	mA	+125ºC(6)		

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD ；
MCLR = VDD ； 根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

© 2011 Microchip Technology Inc.

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级） （续）

PIC18F87K22 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)						
	所有器件	42	73	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 1 MHz （**PRI_IDLE** 模式、EC 振荡器）
		42	73	μA	+25°C		
		43	74	μA	+85°C		
		53	100	μA	+125ºC		
	所有器件	110	190	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		110	195	μA	+25°C		
		110	195	μA	+85°C		
		130	250	μA	+125ºC		
	所有器件	280	450	μA	-40°C	VDD = 5V(5) 使能稳压器	
		290	440	μA	+25°C		
		300	460	μA	+85°C		
		330	500	μA	+125ºC		
	所有器件	160	360	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 4 MHz （**PRI_IDLE** 模式、EC 振荡器）
		160	360	μA	+25°C		
		170	370	μA	+85°C		
		200	400	μA	+125ºC		
	所有器件	330	650	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		340	660	μA	+25°C		
		340	660	μA	+85°C		
		370	700	μA	+125ºC		
	所有器件	510	900	μA	-40°C	VDD = 5V(4) 使能稳压器	
		520	950	μA	+25°C		
		540	990	μA	+85°C		
		600	1200	μA	+125ºC		
	所有器件	4.7	9	mA	-40°C	VDD = 3.3V(4) 禁止稳压器	FOSC = 64 MHz （**PRI_IDLE** 模式、EC 振荡器）
		4.8	9	mA	+25°C		
		4.8	10	mA	+85°C		
		5.2	12	mA	+125ºC(6)		
	所有器件	5.1	11	mA	-40°C	VDD = 5V(4) 使能稳压器	
		5.1	11	mA	+25°C		
		5.2	12	mA	+85°C		
		5.7	14	mA	+125ºC(6)		

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD；
MCLR = VDD；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级） （续）

PIC18F87K22 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)						
	所有器件	3.7	8.5	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 32 kHz(3) （**SEC_RUN** 模式， SOSCSEL = 01）
		5.4	10	μA	+25°C		
		6.6	13	μA	+85°C		
		13	30	μA	+125ºC		
	所有器件	8.7	18	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		10	20	μA	+25°C		
		12	23	μA	+85°C		
		25	60	μA	+125ºC		
	所有器件	60	160	μA	-40°C	VDD = 5V(4) 使能稳压器	
		90	190	μA	+25°C		
		100	240	μA	+85°C		
		200	450	μA	+125ºC		
	所有器件	1.2	4	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	FOSC = 32 kHz(3) （**SEC_IDLE** 模式， SOSCSEL = 01）
		1.7	5	μA	+25°C		
		2.6	6	μA	+85°C		
		9	20	μA	+125ºC		
	所有器件	1.6	7	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		2.8	9	μA	+25°C		
		4.1	10	μA	+85°C		
		17	40	μA	+125ºC		
	所有器件	60	150	μA	-40°C	VDD = 5V(5) 使能稳压器	
		80	180	μA	+25°C		
		100	240	μA	+85°C		
		180	440	μA	+125ºC		

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD；
MCLR = VDD； 根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

© 2011 Microchip Technology Inc.

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级） （续）

PIC18F87K22 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		
	模块差分电流 （ΔIWDT、 ΔIBOR、 ΔIHLVD、 ΔIOSCB 和 ΔIAD）						
	看门狗定时器						
D022 （ΔIWDT） D022 （ΔIWDT）	所有器件	0.3	1	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	
		0.3	1	μA	+25°C		
		0.3	1	μA	+85°C		
		0.5	2	μA	+125ºC		
	所有器件	0.6	2	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		0.6	2	μA	+25°C		
		0.7	2	μA	+85°C		
		1	3	μA	+125ºC		
	所有器件	0.6	2	μA	-40°C	VDD = 5V(5) 使能稳压器	
		0.6	2	μA	+25°C		
		0.7	2	μA	+85°C		
		1.5	4	μA	+125ºC		
D022A （ΔIBOR） （ΔIBOR）	欠压复位						
	所有器件	4.6	19	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	高功耗 BOR
		4.5	20	μA	+25°C		
		4.7	20	μA	+85°C		
		18	40	μA	+125ºC		
	所有器件	4.2	20	μA	-40°C	VDD = 5V(5) 使能稳压器	高功耗 BOR
		4.3	20	μA	+25°C		
		4.4	20	μA	+85°C		
		20	40	μA	+125ºC		
D022B （ΔIHLVD）	高 / 低电压检测						
	所有器件	3.8	9	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	
		4.2	9	μA	+25°C		
		4.3	10	μA	+85°C		
		4.5	12	μA	+125ºC		
	所有器件	4.5	11	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		4.8	12	μA	+25°C		
		4.8	12	μA	+85°C		
		5.0	14	μA	+125ºC		
	所有器件	4.9	13	μA	-40°C	VDD = 5V(5) 使能稳压器	
		4.9	13	μA	+25°C		
		4.9	13	μA	+85°C		
		5.3	15	μA	+125ºC		

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD；
MCLR = VDD；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

31.2 直流特性： 掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）（续）

PIC18F87K22 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		
D025 （ΔIRTCC）	带有 SOSC 振荡器的实时时钟 / 日历						
	所有器件	0.7	2.7	μA	-40°C	VDD = 1.8V(4) 禁止稳压器	32.768 kHz， SOSCRUN = 1
		0.7	2.8	μA	+25°C		
		1.1	2.8	μA	+60°C		
		1.1	2.9	μA	+85°C		
		2.2	4.4	μA	+125ºC		
	所有器件	1.2	2.9	μA	-40°C	VDD = 3.3V(4) 禁止稳压器	
		1.1	2.8	μA	+25°C		
		2	4.6	μA	+60°C		
		2	4.8	μA	+85°C		
		4	6.5	μA	+125ºC		
	所有器件	1.5	4.4	μA	-40°C	VDD = 5V(5) 使能稳压器	
		1.5	4.4	μA	+25°C		
		1.7	4.7	μA	+60°C		
		1.7	4.7	μA	+85°C		
		3.5	6.9	μA	+125ºC		

注 1： 休眠模式下的掉电电流不是由振荡器类型决定的。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并且连接到 VDD 或者 VSS，以及禁止所有会带来新增电流的功能部件 （比如 WDT、 SOSC 振荡器或 BOR 等）时测得的。

2： 供电电流主要是由工作电压、频率和模式一起决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型和电路、内部代码执行模式和温度也会影响电流消耗。
在正常工作模式下，所有 IDD 测量的测试条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD；
MCLR = VDD；根据具体应用使能或禁止 WDT。

3： 标准低成本 32 kHz 晶振的工作温度范围为 -10°C 到 +70°C。扩展级温度的晶振成本要高很多。

4： 禁止稳压器 （ENVREG = 0、连接到 VSS 且 RETEN （CONFIG1L<0>） = 1）。

5： 使能稳压器 （ENVREG = 1、连接到 VDD、 SRETEN （WDTCON<4>） = 1 且 RETEN （CONFIG1L<0>） = 0）。

6： +125ºC 时的最大频率为 48 MHz。

© 2011 Microchip Technology Inc.

31.3 直流特性： PIC18F87K22 系列 （工业级 / 扩展级）

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：				
D030		带 TTL 缓冲器	VSS	0.15 VDD	V	VDD < 4.5V
D031		带施密特触发缓冲器	—	0.8	V	4.5V ≤ VDD ≤ 5.5V
		RC3 和 RC4	VSS	0.2 VDD	V	VDD < 4.5
		RD5 和 RD6	VSS	1.5	V	4.5V ≤ VDD ≤ 5.5V
D031A		RC3 和 RC4	VSS	0.3 VDD	V	使能 I²C™
D031B		RD5 和 RD6	VSS	0.8	V	使能 SMBus
D032		$\overline{\text{MCLR}}$	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				
		带模拟功能的 I/O 端口：				
D040		带 TTL 缓冲器	0.25 VDD	VDD	V	VDD < 4.5V
			2.0	VDD	V	4.5V ≤ VDD ≤ 5.5V
D041		带施密特触发缓冲器	0.8	VDD	V	VDD < 4.5V
		RC3 和 RC4	0.7 VDD	VDD	V	VDD < 4.5V
		RD5 和 RD6	3V	5.5	V	4.5V ≤ VDD ≤ 5.5V
D041A		RC3 和 RC4	0.7 VDD	VDD	V	使能 I²C
D041B		RD5 和 RD6	2.1	VDD	V	使能 SMBus
D042		$\overline{\text{MCLR}}$	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	±200	nA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态
D061		$\overline{\text{MCLR}}$	—	±5	μA	VSS ≤ VPIN ≤ VDD， +85°C
D063		OSC1	—	±5	μA	VSS ≤ VPIN ≤ VDD
	IPU	弱上拉电流				
D070	IPURB	PORTB 弱上拉电流	50	400	μA	VDD = 3.3V， VPIN = VSS

注 1： 负电流定义为引脚的拉电流。

31.3 直流特性： PIC18F87K22 系列 （工业级 / 扩展级）（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 = 4.5V，-40°C 至 +125°C
		PORTD、 PORTE、PORTF、PORTG、 PORTH 和 PORTJ	—	0.6	V	IOL = 3.5 mA， VDD = 4.5V, -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 = 4.5V，-40°C 至 +125°C
		PORTD、 PORTE、 PORTF、PORTG、PORTH 和 PORTJ	VDD – 0.7	—	V	IOH = -2 mA， VDD = 4.5V，-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	COSC2	OSC2 引脚	—	20	pF	在外部时钟用于驱动 OSC1 时处于 HS 模式
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 （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 (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， 10 倍基本电流范围	—	5.5	—	µA	CTMUICON<1:0> = 10
	IOUT3	CTMU 电流源， 100 倍基本电流范围	—	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）。

© 2011 Microchip Technology Inc.

表 31-1：　存储器编程要求

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	典型值 †	最大值	单位	条件
		内部程序存储器编程规范 (1)					
D110	VPP	MCLR/VPP/RE5 引脚上的电压	VDD + 1.5	—	10	V	（注 3 和注 4）
D113	IDDP	编程期间的供电电流	—	—	10	mA	
		数据 EEPROM 存储器					（注 2）
D120	ED	字节耐擦写能力	100K	—	—	E/W	-40°C 至 +125°C
D121	VDRW	用于读 / 写的 VDD	1.8	—	5.5	V	使用 EECON 读 / 写，ENVREG 连接至 VDD
			1.8	—	3.6	V	使用 EECON 读 / 写，ENVREG 连接至 VSS
D122	TDEW	擦 / 写周期	—	4	—	ms	
D123	TRETD	保存时间	40	—	—	年	假设没有违反其他规范
D124	TREF	刷新之前的总擦除 / 写周期数 (2)	1M	10M	—	E/W	-40°C 至 +125°C
		闪存程序存储器					
D130	EP	芯片耐擦写能力	10K	—	—	E/W	-40°C 至 +125°C
D131	VPR	用于读操作的 VDD	1.8	—	5.5	V	ENVREG 连接到 VDD
			1.8	—	3.6	V	ENVREG 连接到 VSS
D132B	VPEW	用于自定时擦除或写操作的电压 VDD	1.8	—	5.5	V	ENVREG 连接到 VDD
D133A	TIW	自定时写周期	—	2	—	ms	
D134	TRETD	保存时间	40	—	—	年	假设没有违反其他规范
D135	IDDP	编程期间的供电电流	—	—	10	mA	
D140	TWE	每个擦除周期中的写时间	—	—	1		每一个物理地址

† 除非另外声明，否则 “典型值” 栏中的数据均为 3.3V， 25°C 下的值。这些参数仅供设计参考，未经测试。

注 1： 这些规范适用于通过表写指令对片上程序存储器进行编程的情况。
2： 有关数据 EEPROM 耐擦写能力的更多详细信息，请参见第 9.8 节 “使用数据 EEPROM”。
3： 仅当禁止单电源编程时需要。
4： MPLAB® ICD 2 不支持可变 VPP 输出。当使用 MPLAB ICD 2 进行编程或调试时，用于限制 MPLAB ICD 2 VPP 电压的电路必须置于 MPLAB ICD 2 和目标系统之间。

表 31-2：　　比较器规范

工作条件：1.8V ≤ VDD ≤ 5V， -40°C ≤ TA ≤ +125°C （除非另外声明）							
参数编号	符号	特性	最小值	典型值	最大值	单位	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施加电压 (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 ≤ 5V，-40°C ≤ TA ≤ +125°C （除非另外声明）							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
D310	VRES	分辨率	—	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 （除非另外声明）							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
	VRGOUT	稳压器输出电压	—	3.3	—	V	
	CEFC	外部滤波器电容值	4.7	10	—	µF	电容的等效串联阻抗 ESR 必须很低 （< 5Ω）

© 2011 Microchip Technology Inc.

31.5 交流（时序）特性

31.5.1 时序参数符号

可根据以下任一格式来创建时序参数符号：

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（高阻态）	V	有效
L	低	Z	高阻态
仅用于 I²C 模式			
AA	输出通道	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		

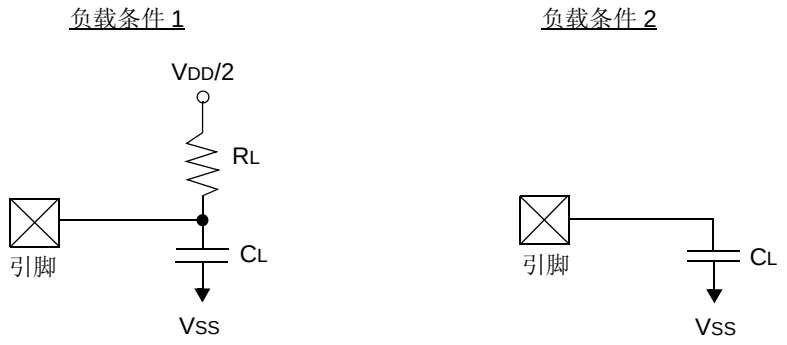
31.5.2 时序条件

除非另外声明，否则表 31-5 中指定的温度和电压适用于所有时序规范。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**中所示。

图 31-3：器件时序规范的负载条件



RL = 464Ω

CL = 50 pF 适用于除 OSC2/CLKO/RA6 之外的所有引脚，包括作为输出的 D 和 E 端口

CL = 15 pF 适用于 OSC2/CLKO/RA6 引脚

© 2011 Microchip Technology Inc.

31.5.3 时序图和规范

图 31-4： 外部时钟时序

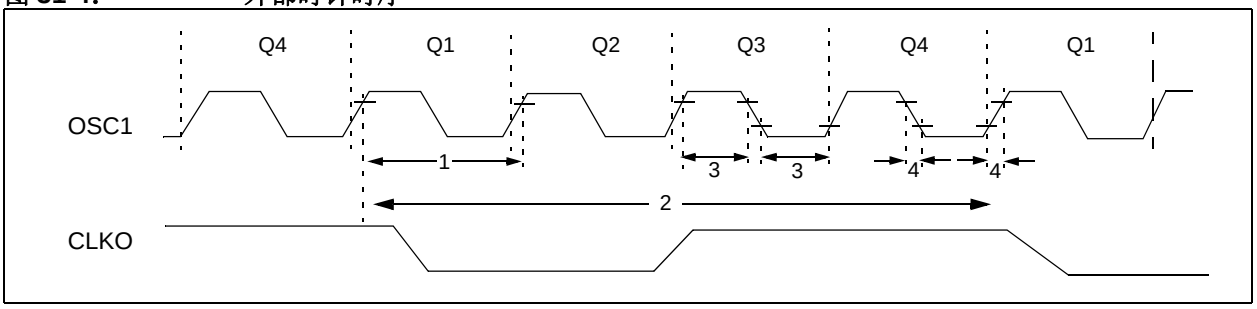


表 31-6： 外部时钟时序要求

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 -40°C ≤ TA ≤ +85°C
			DC	48	MHz	-40°C ≤ TA ≤ +125°C
		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
		振荡器周期 (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 的所有配置来说，指令周期（TCY）等于输入振荡器时基周期的 4 倍。所有指定值均为在特定振荡器模式下，器件在标准工作条件下执行代码时获得的特征数据。超过规定值可能导致振荡器运行不稳定和 / 或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”（没有时钟）。

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 +85°C
			4	—	12	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 +85°C
			16	—	48	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 （抗抖动性能）	-2	—	+2	%	

† 除非另外声明，否则 “典型值”栏中的数据均为 3V， 25°C 下测得的值。这些参数仅供设计参考，未经测试。

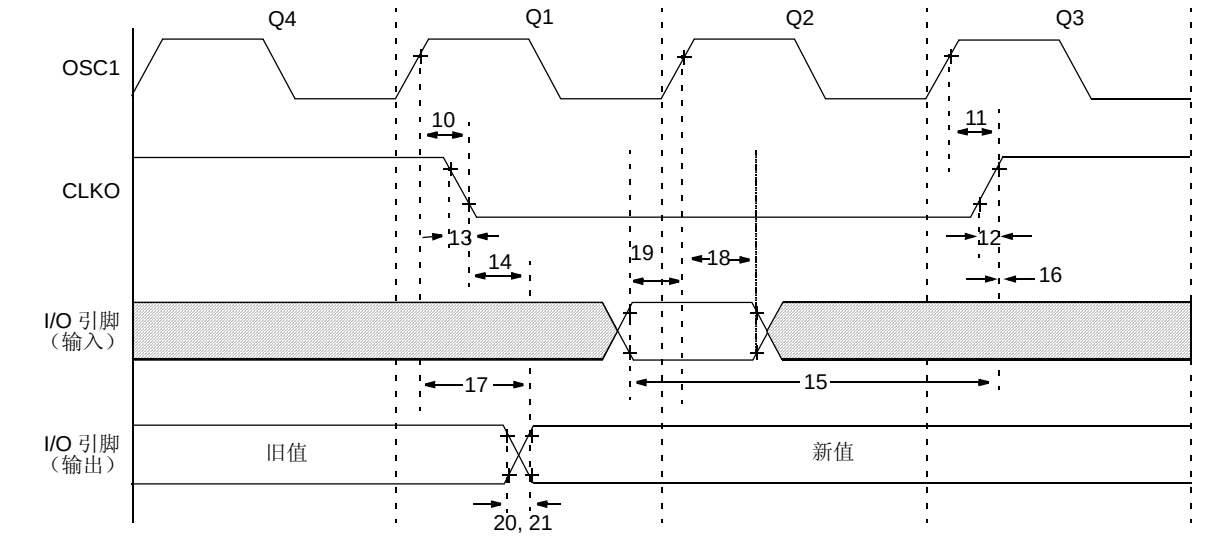
表 31-8： 内部 RC 精度 （INTOSC）

PIC18F87K22 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	**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 = 3.0-5.5V
		-5	—	5	%		
		-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	**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。

© 2011 Microchip Technology Inc.

图 31-5： CLKO 和 I/O 时序



注： 请参见图 31-3 以了解负载条件。

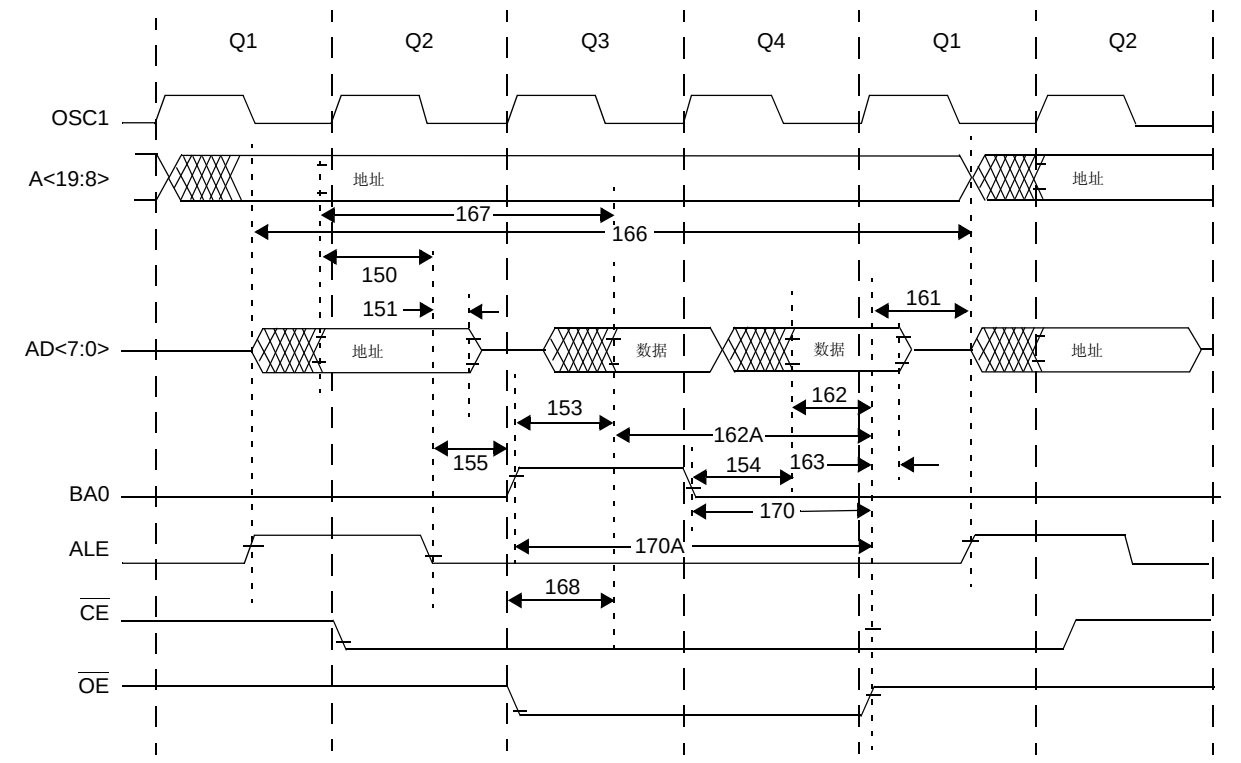
表 31-9： CLKO 和 I/O 时序要求

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	在 CLKO↑之前端口输入有效的时间	0.25 TCY + 25	—	—	ns	
16	TCKH2IOI	在 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。

图 31-6： 程序存储器取指时序图 （8 位）



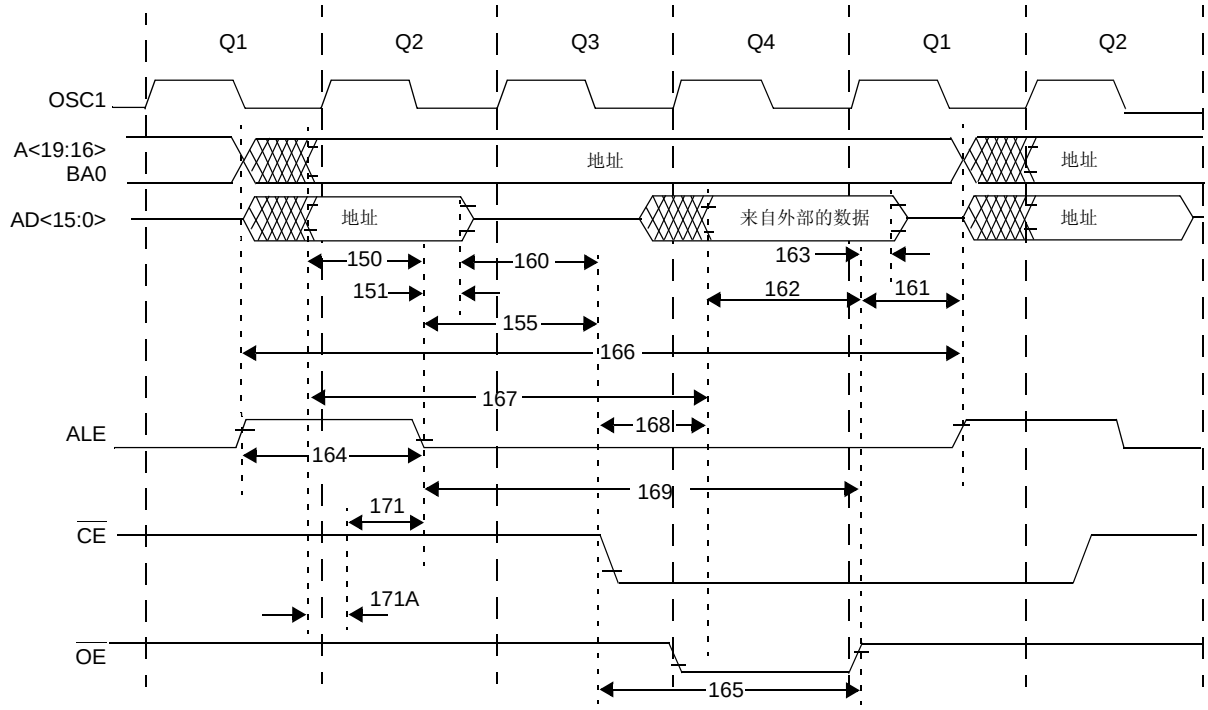
注：在 8 位外部存储器模式下，Fmax = 25 MHz。

表 31-10： 程序存储器取指时序要求 （8 位）

参数编号	符号	特性	最小值	典型值	最大值	单位
150	TadV2aIL	地址输出有效到 ALE ↓ 的时间 （地址建立时间）	0.25 TCY – 10	—	—	ns
151	TaIL2adl	ALE ↓ 到地址输出无效的时间 （地址保持时间）	5	—	—	ns
153	BA01	BA0 ↑ 到数据最高位有效的时间	0.125 TCY	—	—	ns
154	BA02	BA0 ↓ 到数据最低位有效的时间	0.125 TCY	—	—	ns
155	TaIL2oeL	ALE ↓ 到 $\overline{OE}$ ↓ 的时间	0.125 TCY	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ 到 A/D 驱动的时间	0.125 TCY – 5	—	—	ns
162	TadV2oeH	在 $\overline{OE}$ ↑ 之前最低数据有效的时间（数据建立时间）	20	—	—	ns
162A	TadV2oeH	在 $\overline{OE}$ ↑ 之前最高数据有效的时间（数据建立时间）	0.25 TCY + 20	—	—	ns
163	ToeH2adI	$\overline{OE}$ ↑ 到数据输入无效的时间 （数据保持时间）	0	—	—	ns
166	TaIH2aIH	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	0.5 TCY – 10	—	—	ns
168	Toe	$\overline{OE}$ ↓ 到数据有效的时间	—	—	0.125 TCY + 5	ns
170	TubH2oeH	在 $\overline{OE}$ ↑ 之前 BA0 = 0 有效的时间	0.25 TCY	—	—	ns
170A	TubL2oeH	在 $\overline{OE}$ ↑ 之前 BA0 = 1 有效的时间	0.5 TCY	—	—	ns

© 2011 Microchip Technology Inc.

图 31-7：　　程序存储器读时序图

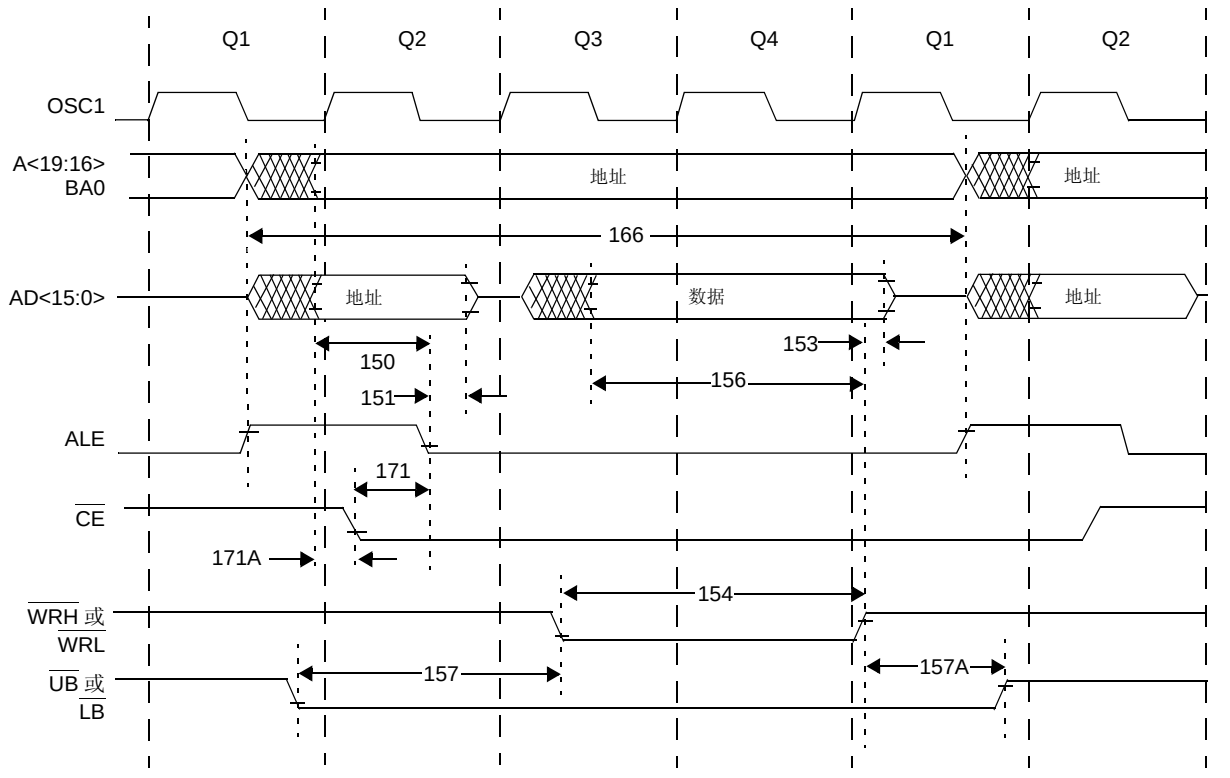


工作条件：2.0V < VCC < 3.6V，-40°C < TA < +125°C（除非另外声明）。

表 31-11：　　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 （总线释放到 $\overline{OE}$）	0	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ 到 AD 驱动的时间	0.125 TCY – 5	—	—	ns
162	TadV2oeH	在 $\overline{OE}$ ↑ 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 31-8： 程序存储器写时序图



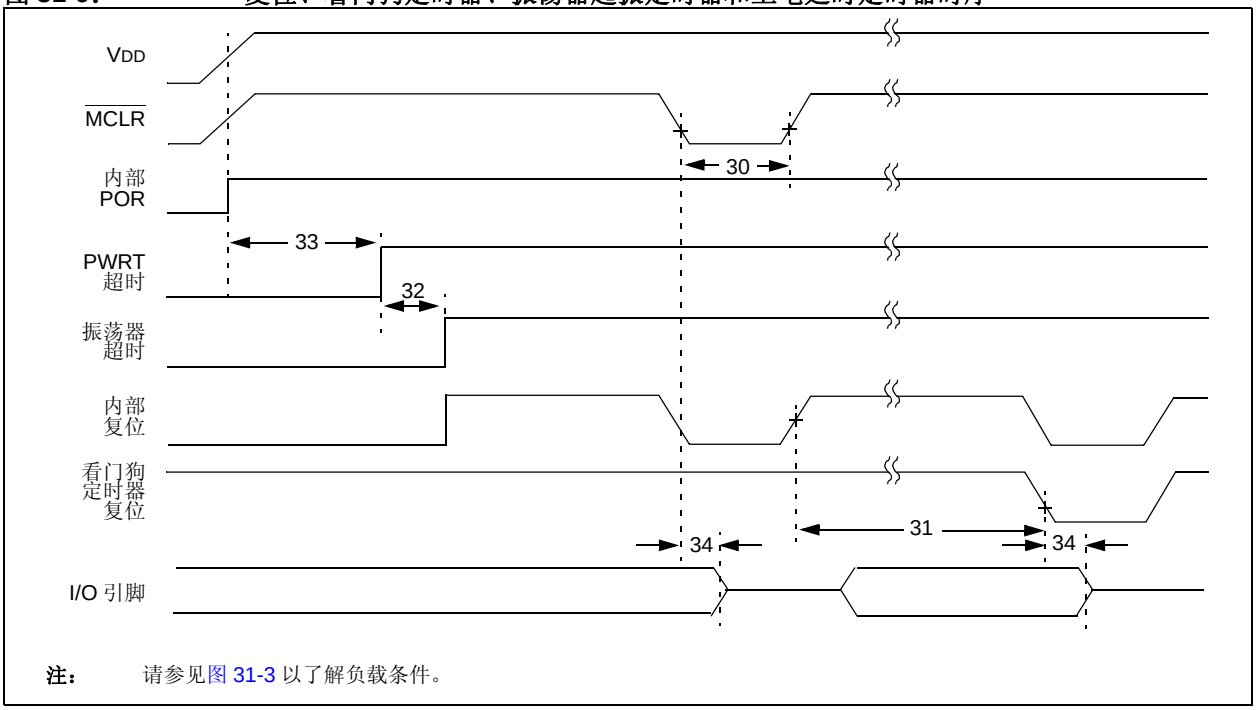
工作条件：2.0V < VCC < 3.6V，-40°C < TA < +125°C（除非另外声明）。

表 31-12： 程序存储器写时序要求

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
153	TwrH2adl	$\overline{WRn}$ ↑ 到数据输出无效的时间 （数据保持时间）	5	—	—	ns
154	TwrL	$\overline{WRn}$ 脉冲宽度	0.5 TCY – 5	0.5 TCY	—	ns
156	TadV2wrH	在 $\overline{WRn}$ ↑ 之前数据有效的时间 （数据建立时间）	0.5 TCY – 10	—	—	ns
157	TbsV2wrL	在 $\overline{WRn}$ ↓ 之前字节选择有效的时间 （字节选择建立时间）	0.25 TCY	—	—	ns
157A	TwrH2bsI	$\overline{WRn}$ ↑ 到字节选择无效的时间 （字节选择保持时间）	0.125 TCY – 5	—	—	ns
166	TalH2alH	ALE ↑ 到 ALE ↑ 的时间 （周期时间）	—	TCY	—	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

© 2011 Microchip Technology Inc.

图 31-9： 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： 请参见图 31-3 以了解负载条件。

图 31-10： 欠压复位时序

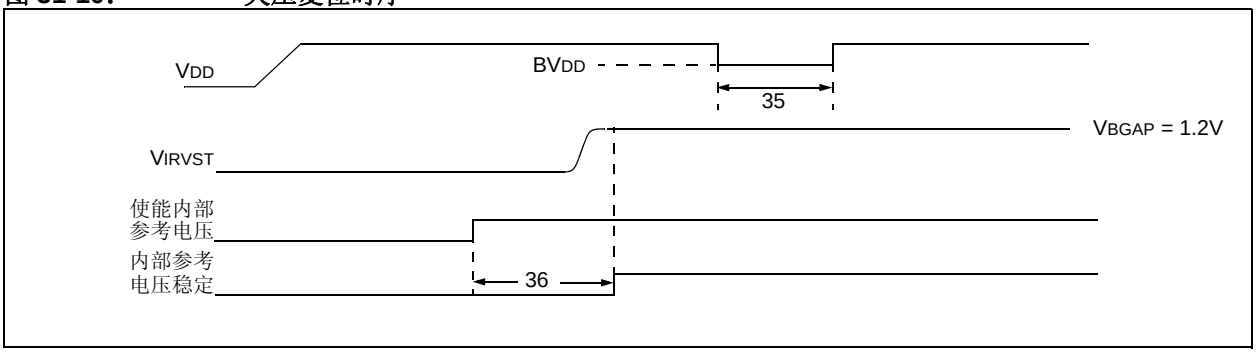


表 31-13： 复位、看门狗定时器、振荡器起振定时器、上电延时定时器和欠压复位要求

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	140	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	TIRVST	内部参考电压稳定的时间	—	25	—	µs	
37	THLVD	高 / 低电压检测脉冲宽度	200	—	—	µs	VDD ≤ VHLVD
38	TCSD	CPU 的起振时间	5	—	10	µs	
39	TIOBST	INTOSC 电路稳定时间	—	1	—	µs	

© 2011 Microchip Technology Inc.

图 31-11： 高 / 低电压检测特性

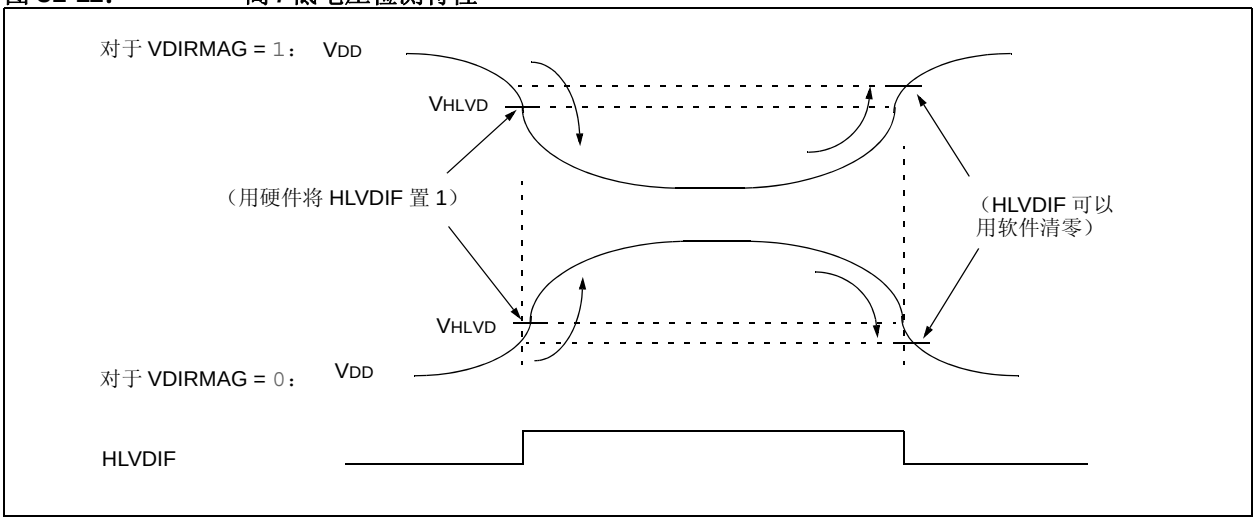
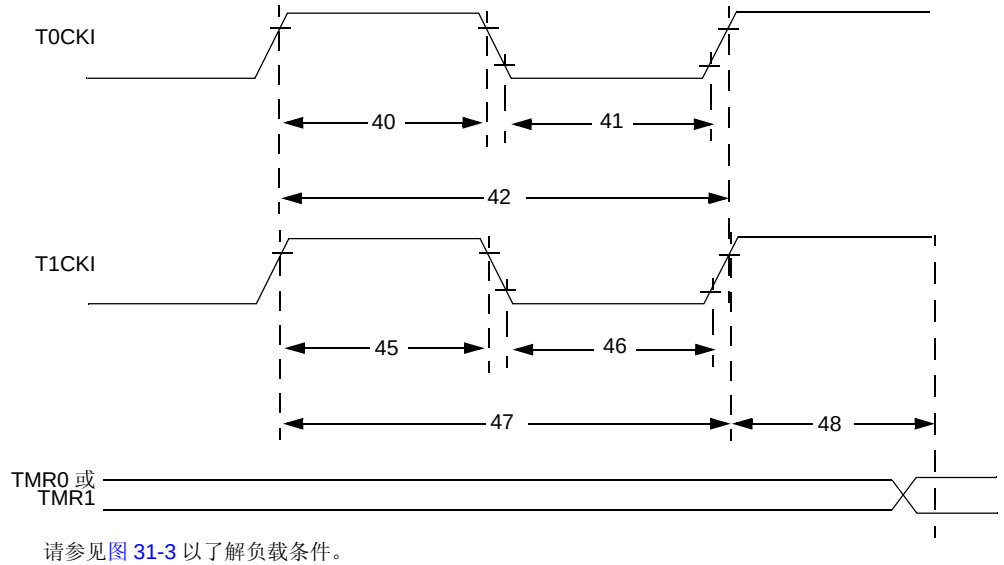


表 31-14： 高 / 低电压检测特性

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 由高电平跳变到低电平时的 HLVD 电压	HLVDL<3:0> = 0000	1.80	1.86	1.90	V	
			HLVDL<3:0> = 0001	2.03	2.12	2.13	V	
			HLVDL<3:0> = 0010	2.24	2.33	2.35	V	
			HLVDL<3:0> = 0011	2.40	2.49	2.53	V	
			HLVDL<3:0> = 0100	2.50	2.59	2.62	V	
			HLVDL<3:0> = 0101	2.70	2.75	2.84	V	
			HLVDL<3:0> = 0110	2.82	2.93	2.97	V	
			HLVDL<3:0> = 0111	2.95	3.07	3.10	V	
			HLVDL<3:0> = 1000	3.24	3.30	3.41	V	
			HLVDL<3:0> = 1001	3.42	3.48	3.59	V	
			HLVDL<3:0> = 1010	3.61	3.67	3.79	V	
			HLVDL<3:0> = 1011	3.82	3.87	4.01	V	
			HLVDL<3:0> = 1100	4.06	4.21	4.26	V	
			HLVDL<3:0> = 1101	4.33	4.42	4.55	V	
			HLVDL<3:0> = 1110	4.64	4.77	4.87	V	

图 31-12： TIMER0 和 TIMER1 外部时钟时序



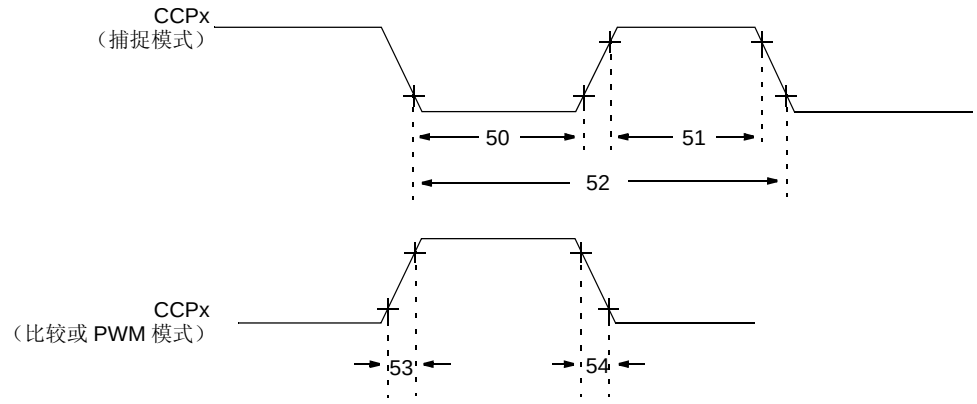
注： 请参见图 31-3 以了解负载条件。

表 31-15： 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	
			有预分频器	10	—	ns	
41	TT0L	T0CKI 低电平脉冲宽度	无预分频器	0.5 TCY + 20	—	ns	
			有预分频器	10	—	ns	
42	TT0P	T0CKI 周期	无预分频器	TCY +10	—	ns	
			有预分频器	取较大值：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，有预分频器	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，有预分频器	10	—	ns	
			异步	30	—	ns	
47	TT1P	T1CKI 输入周期	同步	取较大值：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	—	

© 2011 Microchip Technology Inc.

图 31-13： 捕捉 / 比较 /PWM 时序 （ECCP1 和 ECCP2 模块）

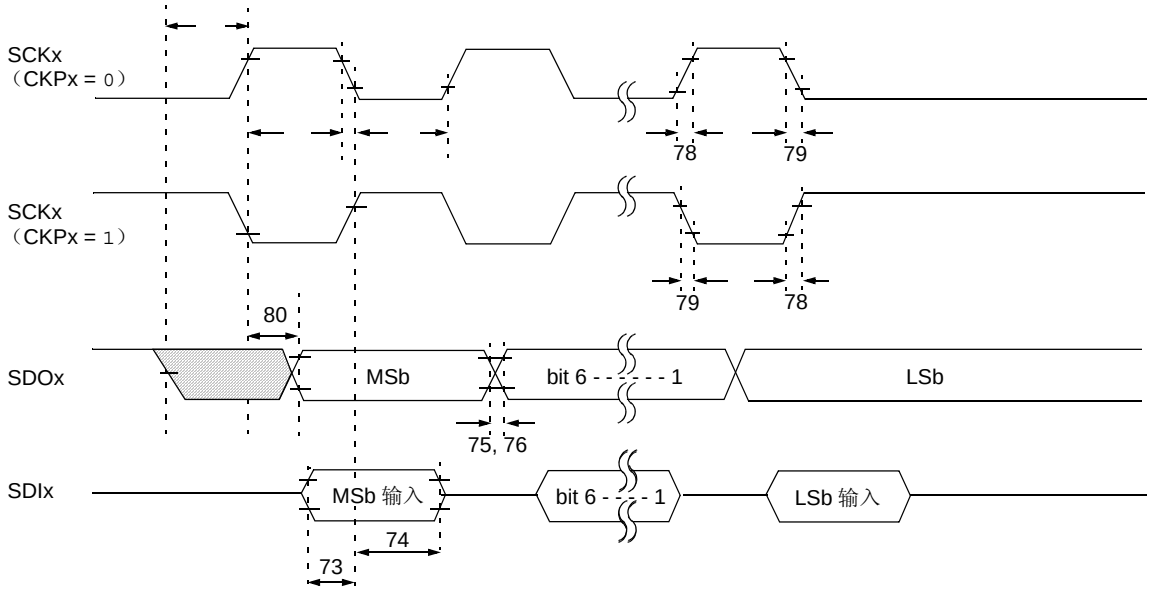


注： 请参见图 31-3 以了解负载条件。

表 31-16： 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	
			有预分频器	10	—	ns	
51	TccH	CCPx 输入高电平时间	无预分频器	0.5 TCY + 20	—	ns	
			有预分频器	10	—	ns	
52	TccP	CCPx 输入周期		3 TCY + 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 31-14： SPI 主模式时序示例 （CKE = 0）



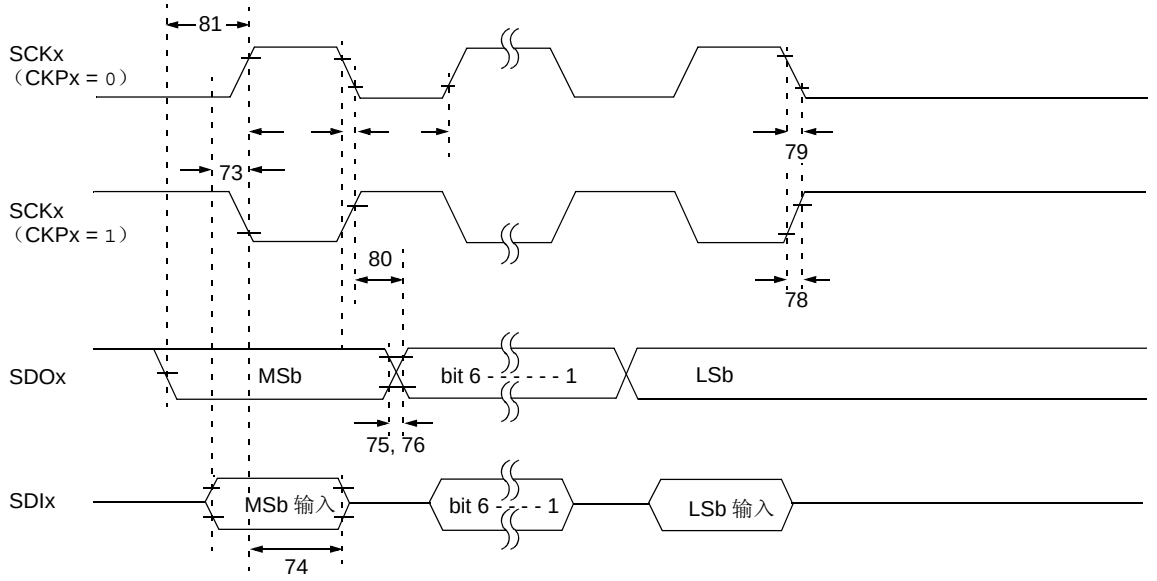
注： 请参见图 31-3 以了解负载条件。

表 31-17： 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 边沿后 SDOx 数据输出有效的时间	—	50	ns	

© 2011 Microchip Technology Inc.

图 31-15： SPI 主模式时序示例 （CKE = 1）

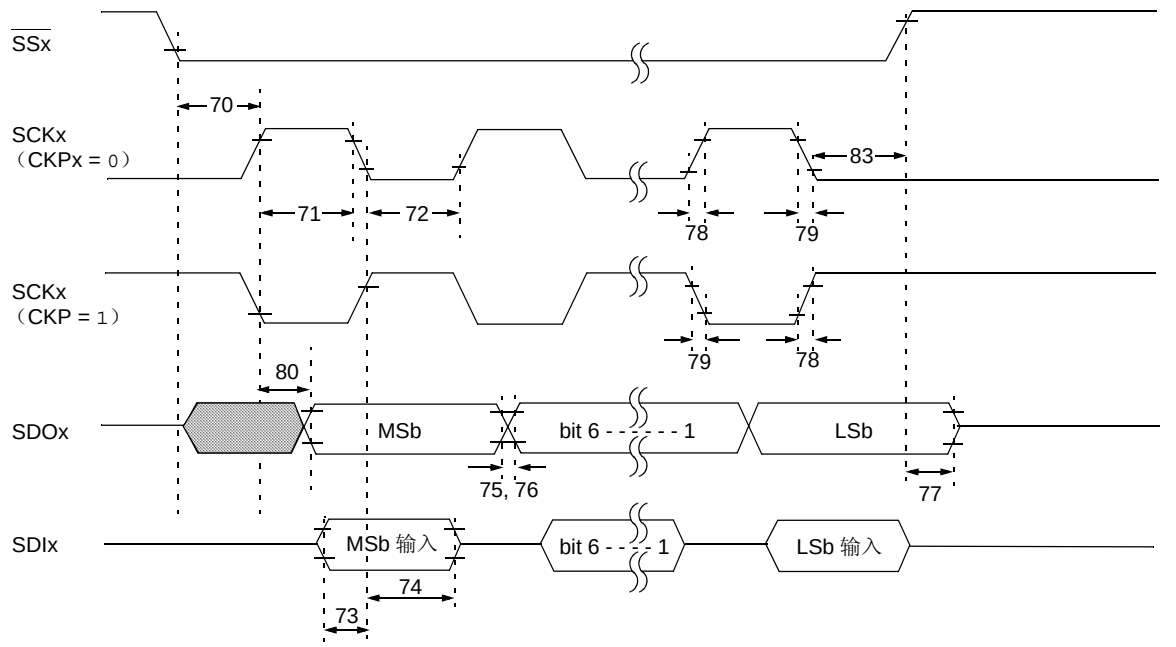


注： 请参见图 31-3 以了解负载条件。

表 31-18： 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 边沿后 SDOx 数据输出有效的时间	—	50	ns	
81	TDOV2SCH，TDOV2SCL	SDOx 数据输出到 SCKx 边沿的建立时间	TCY	—	ns	

图 31-16： SPI 从模式时序示例 （CKE = 0）



注： 请参见图 31-3 以了解负载条件。

表 31-19： SPI 模式要求示例 （从模式时序， CKE = 0）

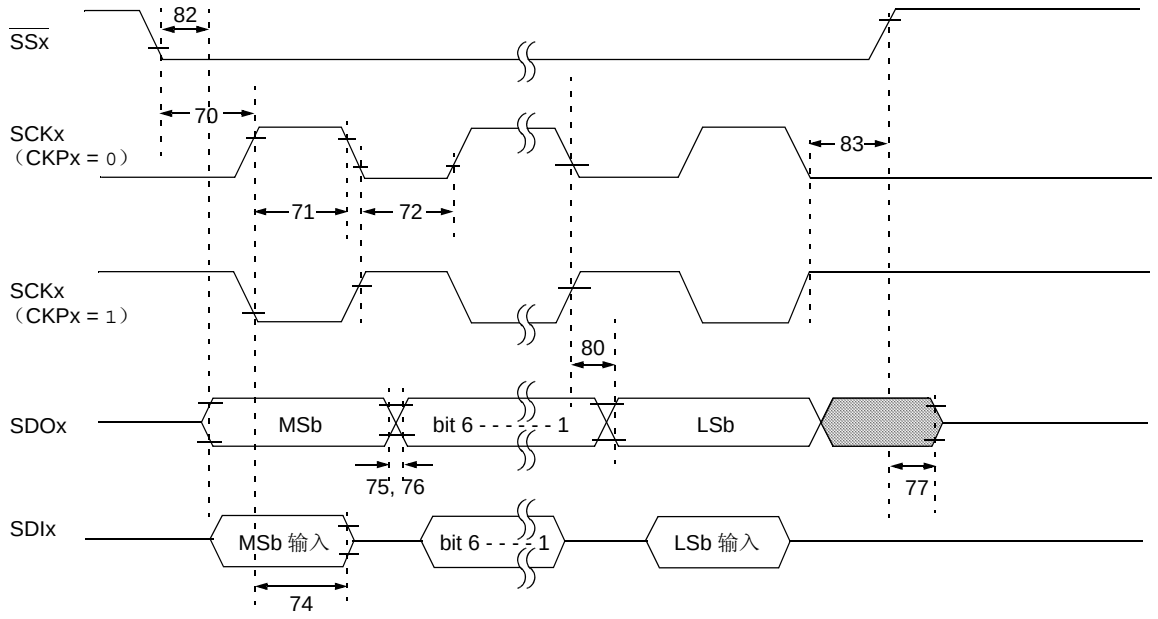
参数编号	符号	特性		最小值	最大值	单位	条件
70	TSSL2SCH，TSSL2SCL	$\overline{SSx}$ ↓ 到 SCKx ↓ 或 SCKx ↑ 输入的时间		3 TCY	—	ns	
70A	TSSL2WB	$\overline{SSx}$ 到写 SSPBUF 的时间		3 TCY	—	ns	
71	TSCH	SCKx 输入高电平时间（从模式）	连续	1.25 TCY + 30	—	ns	
71A			单字节	40	—	ns	（注 1）
72	TSCL	SCKx 输入低电平时间（从模式）	连续	1.25 TCY + 30	—	ns	
72A			单字节	40	—	ns	（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 边沿后 SDOx 数据输出有效的时间		—	50	ns	
83	TSCH2SSH，TSCL2SSH	SCKx 边沿后 $\overline{SSx}$ ↑ 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。

© 2011 Microchip Technology Inc.

图 31-17： SPI 从模式时序示例 （CKE = 1）



注： 请参见图 31-3 以了解负载条件。

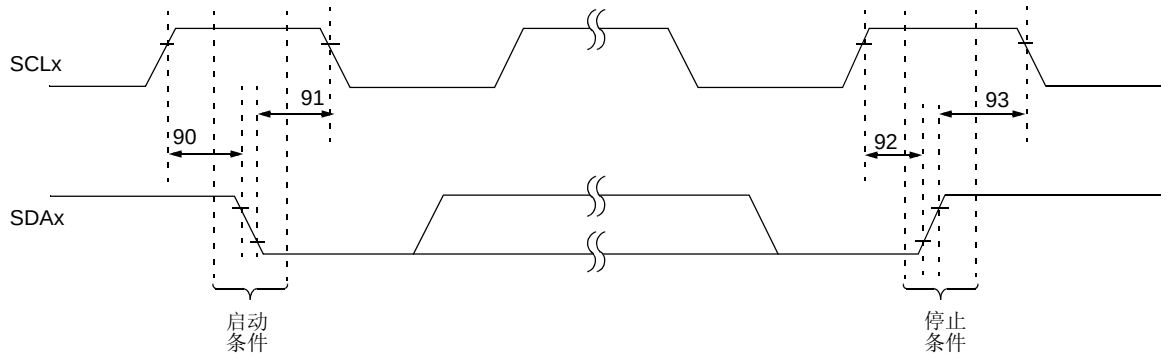
表 31-20： SPI 从模式要求示例 （CKE = 1）

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 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 边沿后 SDOx 数据输出有效的时间		—	50	ns	
82	TSSL2DOV	$\overline{SSx}$ ↓ 边沿后 SDOx 数据输出有效的时间		—	50	ns	
83	TSCH2SSH，TSCL2SSH	SCKx 边沿后 $\overline{SSx}$ ↑ 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 31-18： I²C™ 总线启动 / 停止位时序



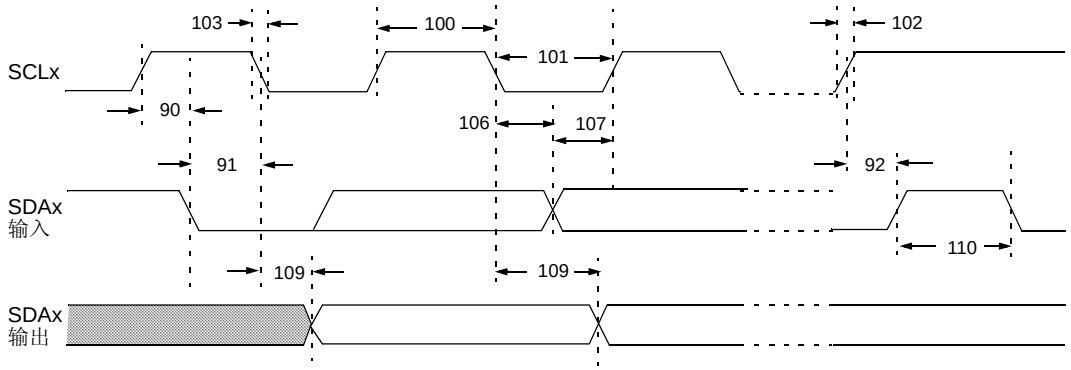
注： 请参见图 31-3 以了解负载条件。

表 31-21： I²C™ 总线启动 / 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	—		

© 2011 Microchip Technology Inc.

图 31-19： I²C™ 总线数据时序



注： 请参见图 31-3 以了解负载条件。

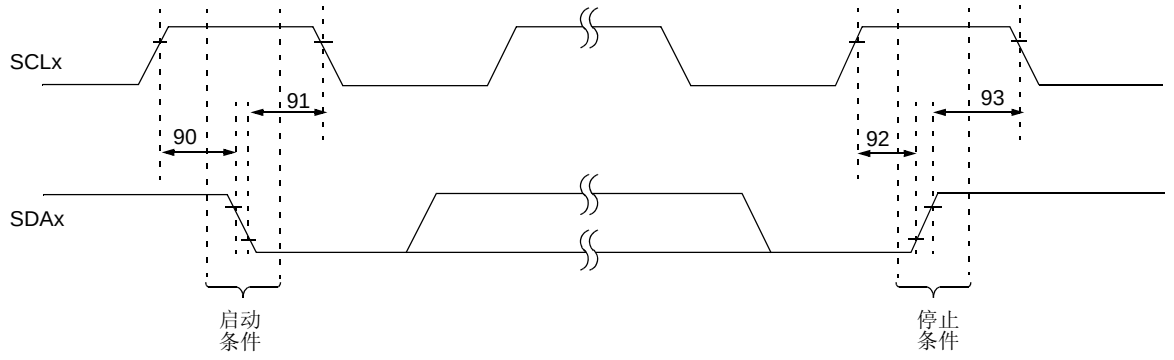
表 31-22： I²C™ 总线数据要求 （从模式）

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 值在 10 到 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 值在 10 到 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	时钟输出有效时间	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： 为避免意外产生启动或停止条件，作为发送器的器件必须提供此内部最小延迟时间，以覆盖 SCLx 下降沿的未定义区域（最小值 300 ns）。

2： 在标准模式的 I²C 总线系统中可以使用快速模式的 I²C™ 总线器件，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 信号的低电平时间，则必然满足此条件。如果该器件延长了 SCLx 信号的低电平时间，必须将下一个数据位输出到 SDAx 线， SCLx 线被释放前，根据标准模式 I²C 总线规范， TR max. + TSU:DAT = 1000 + 250 = 1250 ns。

图 31-20： MSSP I²C™ 总线启动 / 停止位时序波形



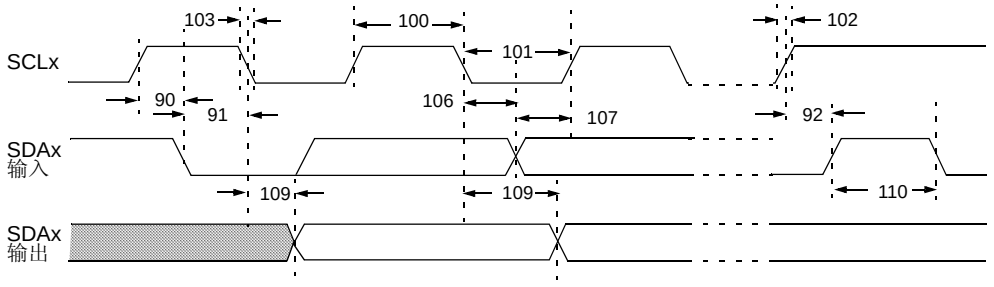
注： 请参见图 31-3 以了解负载条件。

表 31-23： MSSP I²C™ 总线启动 / 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，最大引脚电容为 10 pF。

图 31-21： MSSP I²C™ 总线数据时序



注： 请参见图 31-3 以了解负载条件。

© 2011 Microchip Technology Inc.

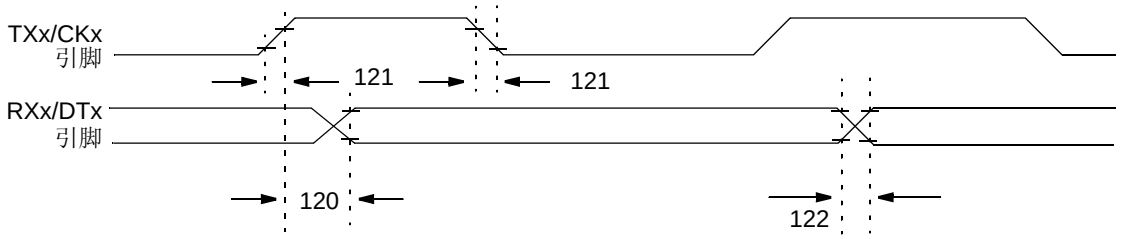
表 31-24： MSSP I²C™ 总线数据要求

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 值在 10 到 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 值在 10 到 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	—	ns	
			400 kHz 模式	0	0.9	µs	
			1 MHz 模式 (1)	—	—	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	时钟输出有效时间	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，最大引脚电容为 10 pF。

2： 在标准模式的 I²C 总线系统中，可以使用快速模式的 I²C 总线器件，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 信号的低电平时间，则必然满足此条件。如果该器件延长了 SCLx 信号的低电平时间，它必须将下一个数据位输出到 SDAx 线， SCLx 线被释放前，参数 102 + 参数 107 = 1000 + 250 = 1250 ns（100 kHz 模式下）。

图 31-22： EUSART 同步发送 （主 / 从）时序

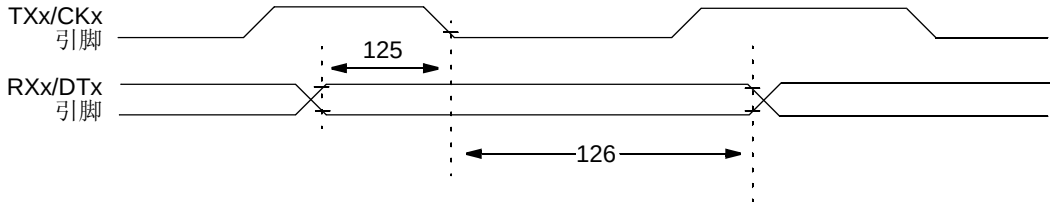


注： 请参见图 31-3 以了解负载条件。

表 31-25： 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 （主和从） 从时钟高电平到数据输出有效的时间	—	40	ns	
121	TCKRF	时钟输出的上升时间和下降时间 （主模式）	—	20	ns	
122	TDTRF	数据输出的上升时间和下降时间	—	20	ns	

图 31-23： EUSART/AUSART 同步接收 （主 / 从）时序



注： 请参见图 31-3 以了解负载条件。

表 31-26： 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 （主和从） CKx ↓ 前数据保持时间 （DTx 保持时间）	10	—	ns	
126	TCKL2DTL	CKx ↓ 后数据保持时间 （DTx 保持时间）	15	—	ns	

© 2011 Microchip Technology Inc.

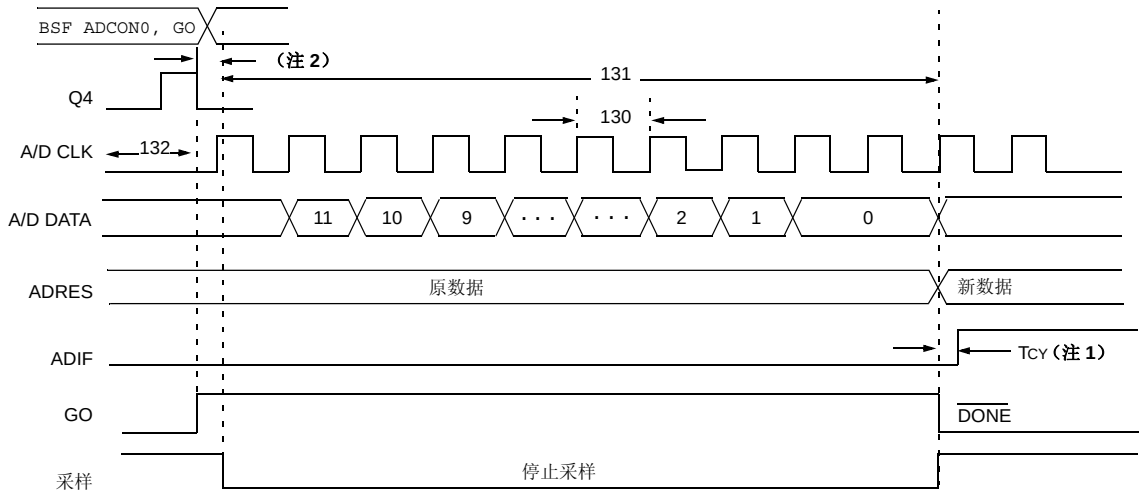
表 31-27： A/D 转换器特性：PIC18F87K22 系列 （工业级）

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 ≥ 5.0V
A03	EIL	积分线性误差	—	±1	±6.0	LSB	ΔVREF = 5.0V
A04	EDL	微分线性误差	—	±1	+3.0/-1.0	LSB	ΔVREF = 5.0V
A06	EOFF	失调误差	—	±1	±9.0	LSB	ΔVREF = 5.0V
A07	EGN	增益误差	—	±1	±8.0	LSB	ΔVREF = 5.0V
A10	—	单调性 (1)	—	—	—	—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	参考电压范围 （VREFH – VREFL）	3	—	VDD – VSS	V	
A21	VREFH	参考电压高电平	VSS + 3.0V	—	VDD + 0.3V	V	
A22	VREFL	参考电压低电平	VSS – 0.3V	—	VDD – 3.0V	V	
A25	VAIN	模拟输入电压	VREFL	—	VREFH	V	
A30	ZAIN	模拟电源阻抗推荐值	—	—	2.5	kΩ	
A50	IREF	VREF 输入电流 (2)	— —	— —	5 150	μA μA	在 VAIN 采集期间 在 A/D 转换期间

注 1： A/D 转换结果不会因输入电压增加而减小。

2： VREFH 电流来自作为 VREFH 源的 RA3/AN3/VREF+ 引脚或 VDD， VREFL 电流来自作为 VREFL 源的 RA2/AN2/VREF- 引脚或 VSS。

图 31-24： A/D 转换时序



注 1： 如果选择 RC 作为 A/D 时钟源，在 A/D 时钟启动前要加上一个 TCY 的时间，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要与模拟输入端断开。

表 31-28： 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，VREF 满量程范围
			—	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）		
137	TDIS	放电时间	0.2	—	μs	-40°C 至 +125°C

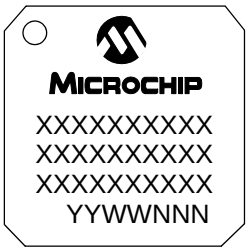
注 1： A/D 时钟周期取决于器件频率和 TAD 时钟分频比。
2： 可在后续 TCY 周期内读取 ADRES 寄存器。
3： 转换完成后当电压满量程变化时 （VDD 到 VSS 或 VSS 到 VDD），保持电容采集一个 “新” 输入电压所需的时间。输入通道上的信号源阻抗 （*R*S）为 50Ω。
4： 在器件时钟的下一个周期上。

© 2011 Microchip Technology Inc.

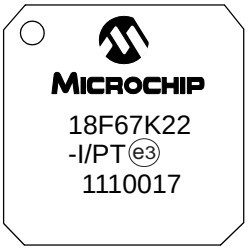
32.0 封装信息

32.1 封装标识信息

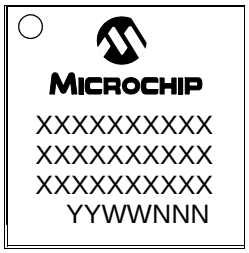
64 引脚 TQFP



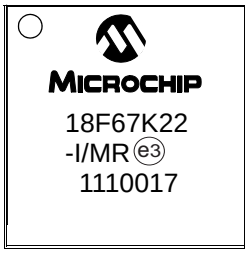
示例



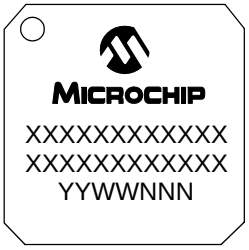
64 引脚 QFN



示例



80 引脚 TQFP



示例

图注：	XX...X	客户指定信息
	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
	*	表示无铅封装。 JEDEC 无铅标志 (e3) 标示于此种封装的外包装上。

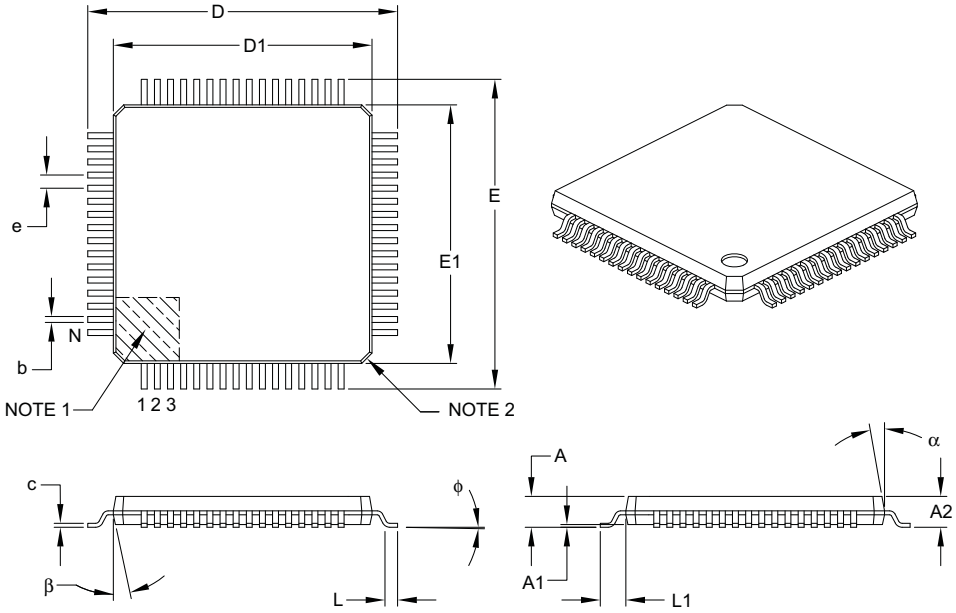
注： Microchip 元器件编号如果无法在同一行内完整标注，将换行标出，因此会限制表示客户指定信息的字符数。

32.2 封装详细信息

以下部分将介绍各种封装的技术细节。

64 引脚塑封薄型四方扁平封装 （PT）——主体 10x10x1 mm，底脚占位 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.

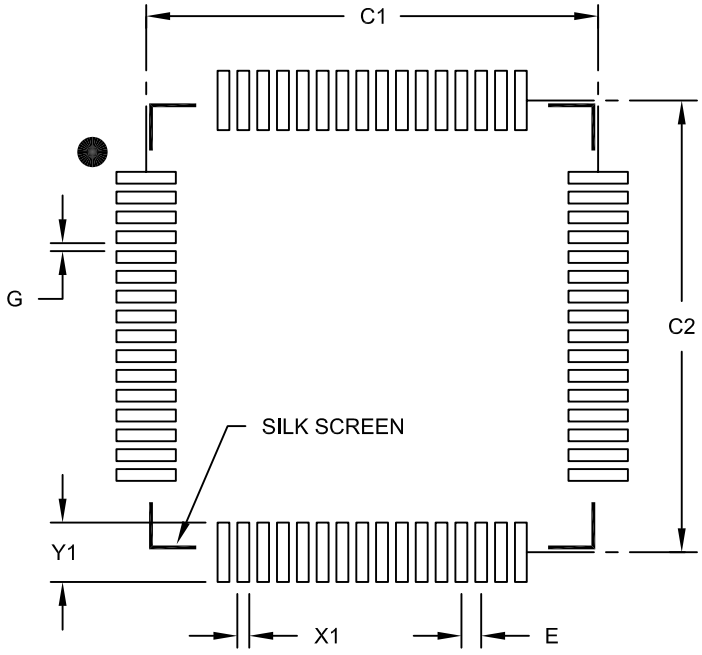
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-085B

© 2011 Microchip Technology Inc.

64 引脚塑封薄型四方扁平封装 （PT）——主体 10x10x1 mm，底脚占位 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:

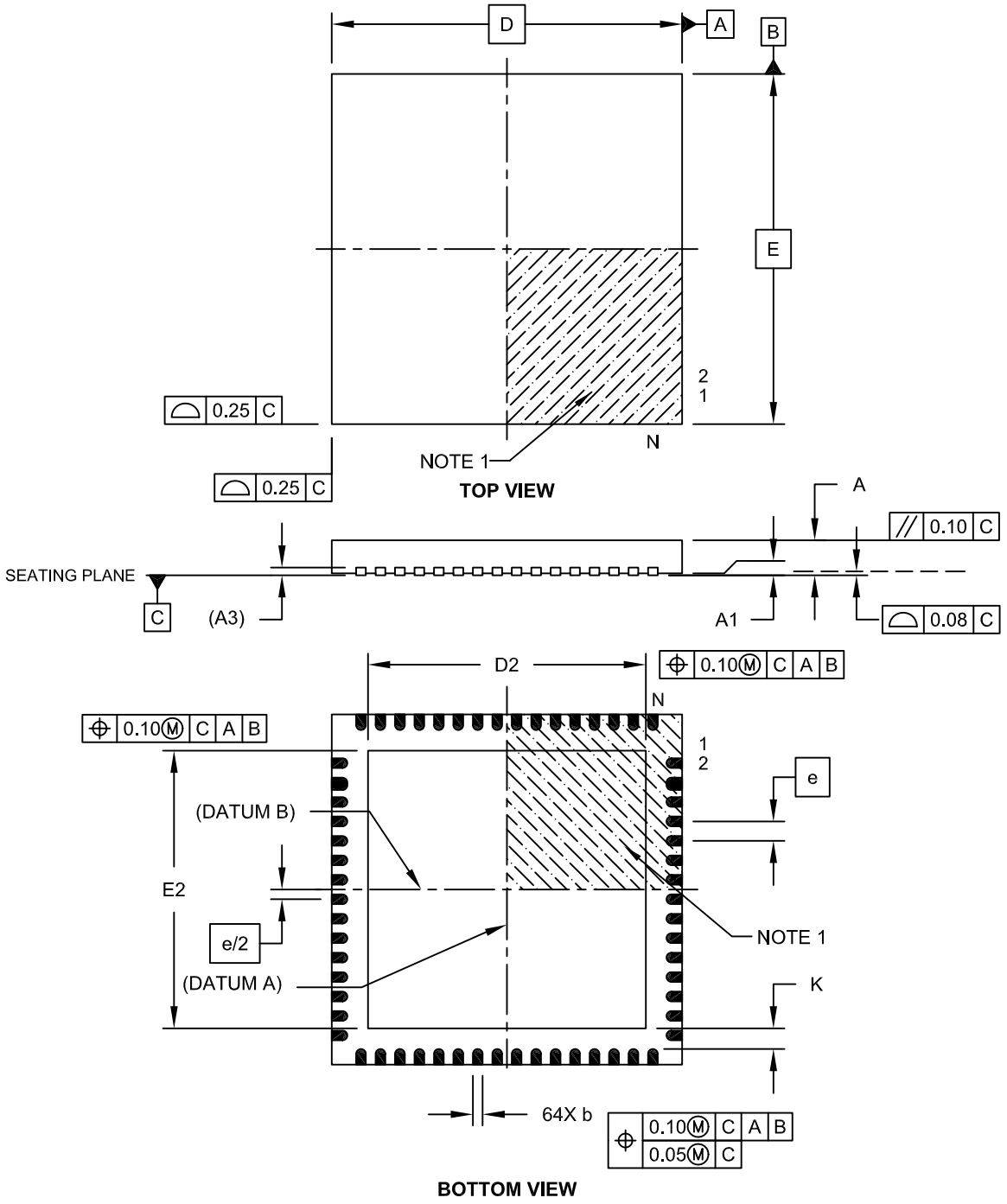
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2085B

64 引脚塑封四方扁平无脚封装 （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。

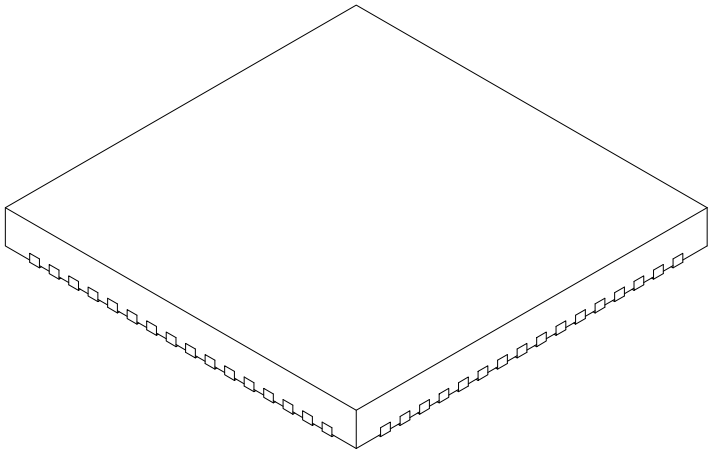


Microchip Technology Drawing C04-149B Sheet 1 of

© 2011 Microchip Technology Inc.

64 引脚塑封四方扁平无脚封装 （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	-	-

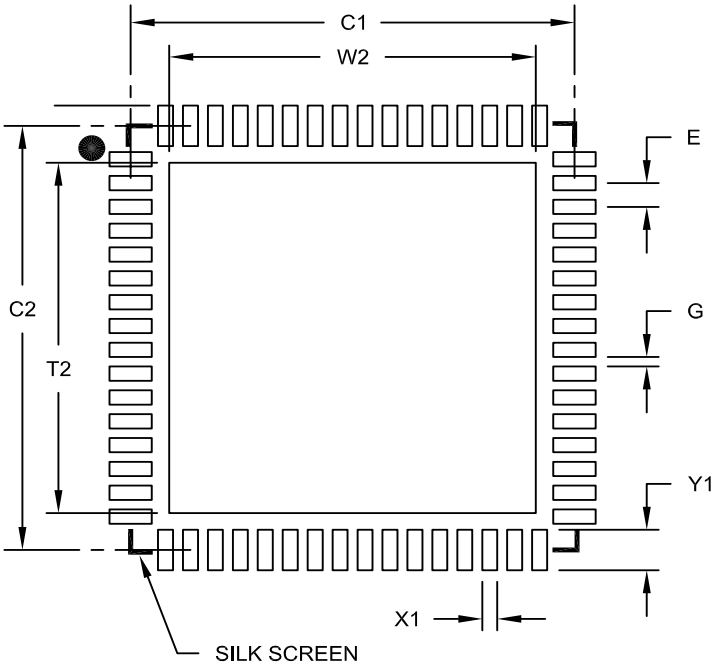
Notes:

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 引脚塑封四方扁平无脚封装 （MR）——主体 9x9x0.9 mm，触点长度 0.40 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		0.50 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

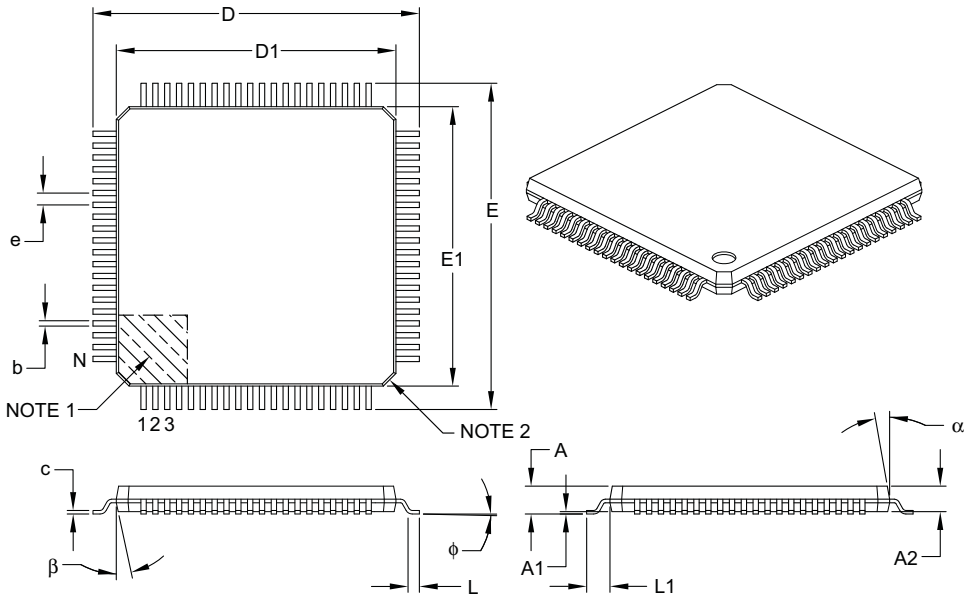
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2149A

© 2011 Microchip Technology Inc.

80 引脚塑封薄型四方扁平封装 （PT）——主体 12x12x1 mm，底脚占位 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	80		
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	14.00 BSC		
Overall Length	D	14.00 BSC		
Molded Package Width	E1	12.00 BSC		
Molded Package Length	D1	12.00 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.

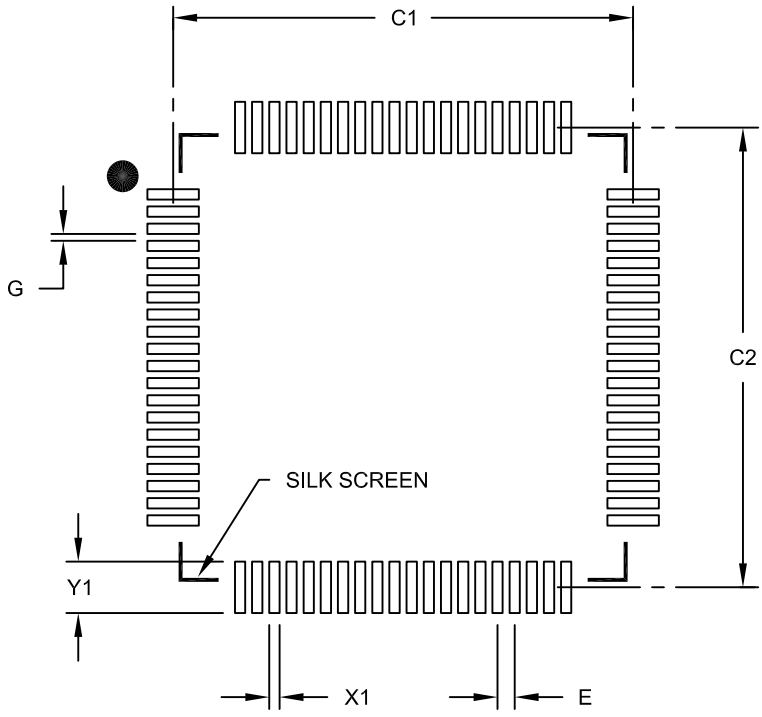
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-092B

80 引脚塑封薄型四方扁平封装 （PT）——主体 12x12x1 mm，底脚占位 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		13.40	
Contact Pad Spacing	C2		13.40	
Contact Pad Width (X80)	X1			0.30
Contact Pad Length (X80)	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. C04-2092B

© 2011 Microchip Technology Inc.

附录 A： 版本历史

版本 A （2009 年 11 月）

PIC18F87K22 系列器件数据手册的最初版本。

版本 B （2010 年 5 月）

对整个文档进行了较小的文本编辑。将所有TBD替换为正确的值。

版本 C （2011 年 3 月）

第 2.4 节“稳压器引脚（ENVREG 和 VCAP/VDDCORE）”更换为新的更详细的说明，且更新了 80 引脚 TQFP 图。对整个文档进行了较小的文本编辑。

版本 D （2011 年 6 月）

使用新的电气规范信息更新了**第 31.2 节 “直流特性：掉电电流和供电电流 PIC18F87K22 系列 （工业级 / 扩展级）”**和表 31-8。此版本中包含了扩展温度信息。

附录 B： 从 PIC18F87J11 和 PIC18F8722 到 PIC18F87K22 的移植

PIC18F87K22、 PIC18F87J11 和 PIC18F8722 系列器件具有非常相似的功能和特性。代码可从其他两个系列器件移植到 PIC18F87K22 系列器件，而不需要过多修改。表 B-1 列出了这三个器件系列之间的差异。

表 B-1： PIC18F87K22、 PIC18F87J11 和 PIC18F8722 系列器件之间的显著差异

特性	PIC18F87K22 系列	PIC18F87J11 系列	PIC18F8722 系列
最大工作频率	64 MHz	48 MHz	40 MHz
最大程序存储器	128 KB	128 KB	128 KB
数据存储器	3862 字节	3930 字节	3930 字节
程序存储器耐擦写性	10000 个写 / 擦除周期（最小值）	10000 个写 / 擦除周期（最小值）	10000 个写 / 擦除周期（最小值）
针对闪存的单字写操作	不支持	支持	不支持
振荡器选项	PLL 可与 INTOSC 一起使用	PLL 可与 INTOSC 一起使用	PLL 可与 INTOSC 一起使用
CTMU	支持	不支持	不支持
RTCC	支持	不支持	不支持
SOSC 振荡器选项 TICKI 时钟	SOSC 的低功耗振荡器选项 T1CKI 可用作时钟，而无需使能 SOSC 振荡器	不支持	不支持
INTOSC	高达 16 MHz	高达 8 MHz	高达 8 MHz
SPI/I^{2}C™	2	2	2
定时器	11	5	5
ECCP	3	3	
CCP	7	2	2
数据 EEPROM	支持	不支持	支持
可编程 BOR	多级 BOR	1 级 BOR	多级 BOR
WDT 预分频选项	22	16	16
5V 工作电压	支持	不支持 （3.3V）	支持
nanoWatt XLP	支持	不支持	不支持
稳压器	支持	支持	不支持
低功耗 BOR	支持	不支持	不支持
A/D 转换器	24 通道差分输入（12 位分辨率）	15 通道非差分输入（10 位分辨率）	16 通道非差分输入（10 位分辨率）
内部温度传感器	支持	不支持	不支持
可编程 HLVD	支持	不支持	支持
EUSART	2 个 EUSART	2 个 EUSART	2 个 EUSART
比较器	3	2	2
振荡器选项	14 个选项（通过 Fosc<3:0> 选择）	8 个选项（通过 Fosc<3:0> 选择）	12 个选项（通过 Fosc<3:0> 选择）
超低功耗唤醒功能 （ULPW）	支持	不支持	不支持
上电延时定时器	支持	支持	支持
用作输入端口的 $\overline{\text{MCLR}}$ 引脚	支持	不支持	支持

© 2011 Microchip Technology Inc.

索引

A

A/D 351
采集要求 362
差分转换器 351
控制寄存器 352
模拟端口引脚，配置 363
配置模块 361
特殊事件触发信号的使用 365
相关的寄存器 366
在功耗管理模式下的操作 365
转换 364
转换器特性 523
转换器中断 , 配置 361
转换时钟 （TAD） 363
转换顺序 361
转换要求 524
转换状态 （GO/DONE 位） 359
自动采集时间 , 选择和配置 363
ACKSTAT 317
ACKSTAT 状态标志 317
ADCON0 寄存器
GO/DONE 位 359
ADDFSR 474
ADDLW 437
ADDWF 437
ADDWFC 438
ADDULNK 474
ANDLW 438
ANDWF 439

B

版本历史 533
BC 439
BCF 440
BF 317
BF 状态标志 317
比较 （CCP 模块） 252
CCPR4 寄存器 252
CCP 引脚配置 252
软件中断 252
Timer1/3/5/7 模式选择 252
特殊事件触发器 252
比较 （ECCP 模块） 264
CCPRx 寄存器 264
软件中断 264
Timer1/2/3/4/6/8/10/12 模式选择 264
特殊事件触发器 221, 264
引脚配置 264
比较器 367
复位的影响 374
工作原理 370
模拟输入连接注意事项 370
配置和控制 371
使能和输出选择 371
使能和输入选择 371
相关的寄存器 374
响应时间 370
在休眠模式下工作 374
中断 373
比较器参考电压 375
复位的影响 376
精度和误差 376
连接注意事项 376
配置 375
相关的寄存器 377
在休眠模式下工作 376
比较器规范 500
编程，器件指令 431
变更通知客户服务 547
表读 / 表写 91
表指针操作 （表） 114
并行从端口 （PSP） 189
PORTD 189
相关的寄存器 191
BN 440
BNC 441
BNN 441
BNOV 442
BNZ 442
波特率发生器 313
BOR。*请参见*欠压复位。
BOV 445
BRA 443
BRG。*参见*波特率发生器。
捕捉 （CCP 模块） 250
CCPR4H:CCPR4L 寄存器 250
软件中断 251
Timer1/3/5/7 模式选择 250
引脚配置 250
预分频器 251
捕捉 （ECCP 模块） 263
CCPRxH:CCPRxL 寄存器 263
软件中断 263
Timer1/2/3/4/6/8/10/12 模式选择 263
引脚配置 263
预分频器 263
捕捉 / 比较 /PWM （CCP） 245
比较模式。*参见*比较。
捕捉模式。*参见*捕捉。
CCP6/7/8/9 引脚分配 250
CCPRxH 寄存器 249
CCPRxL 寄存器 249
漏极开路输出选项 250
配置 249
BSF 443
BTFSC 444
BTFSS 444
BTG 445
BZ 446

C

C 编译器
MPLAB C18 482
CALL 446
CALLW 475
参考电压规范 500
参考时钟输出 53
超低功耗唤醒功能
概述 70
退出延时 71

程序存储器
查找表 91
存储器硬编码向量 88
存储器映射 87
代码保护 427
复位向量 88
扩展指令集 107
指令 93
双字 93
中断向量 88
程序计数器 89
PCL、 PCH 和 PCU 寄存器 89
PCLATH 和 PCLATU 寄存器 89
程序校验和代码保护 426
相关的寄存器 427
充电时间测量单元 （CTMU） 385
测量时间 398
测量温度 400
产生延时 399
复位的影响 401
工作原理 389
休眠 / 空闲模式期间 401
模块初始化 390
使用 CTMU 测量电容 396
相关的寄存器 401
校准模块 390
串行时钟， SCKx 281
串行数据输出 （SDOx） 281
串行数据输入 （SDIx） 281
串行外设接口。*参见* SPI 模式。
CLRF 447
CLRWDT 447
COMF 448
从 PIC18F87J11 和 PIC18F8722 到 PIC18F87K22 的移植 534
从选择 （SSx） 281
CPFSEQ 448
CPFSGT 449
CPFSLT 449
存储器编程要求 499
存储器构成 87
程序存储器映射 87
数据存储器 94

D

DAW 450
代码保护 403
代码示例
16 x 16 无符号乘法程序 140
16 x 16 有符号乘法程序 140
8 x 8 无符号乘法程序 139
8 x 8 有符号乘法程序 139
擦除闪存程序存储器块 116
CTMU 校准设置程序 392
超低功耗唤醒初始化 70
初始化 PORTA 170
初始化 PORTB 172
初始化 PORTC 174
初始化 PORTD 176
初始化 PORTE 178
初始化 PORTF 181
初始化 PORTG 183
初始化 PORTH 185
初始化 PORTJ 187
电流校准程序 393
电容校准程序 395
读数据 EEPROM 136
读一个闪存程序存储器字 115
改变捕捉预分频比 251, 263
将 RTCWREN 位置 1 239
将 Status、 WREG 和 BSR 寄存器的值保存在 RAM 中 163
快速寄存器堆栈 91
使用间接寻址清零 RAM （Bank 1）的方法 105
使用内部二极管测量温度的程序 400
使用偏移量计算 GOTO 91
数据 EEPROM 刷新程序 137
写数据 EEPROM 136
写闪存程序存储器 ..... 119 – 120
用于容性触摸开关的程序 397
装载 SSPBUF （SSP1SR）寄存器 284
DCFSNZ 451
DECF 450
DECFSZ 451
电气特性 485
读者反馈表 548
对标准 PIC18 指令的影响 478
堆栈满 / 下溢复位 91

E

EUSART
波特率发生器
在功耗管理模式下工作 331
波特率发生器 （BRG） 331
波特率，异步模式 333
波特率误差，计算 332
采样 331
高波特率选择位 （BRGH 位） 331
相关的寄存器 332
自动波特率检测 335
同步从模式 349
发送 349
发送序列 349
接收 350
接收序列 350
相关寄存器，发送 349
相关寄存器，接收 350
同步主模式 345
发送 345
发送序列 345
接收 347
接收序列 347
相关寄存器，发送 346
相关寄存器，接收 348
异步模式 337
12 位间隔发送和接收 344
发送器 337
发送序列 337
接收器 340
接收序列 340
设置带有地址检测功能的 9 位模式 340
使能地址检测功能的接收序列 340
同步间隔字符自动唤醒 342
相关寄存器，发送 339
相关寄存器，接收 341

F

返回地址堆栈 89

© 2011 Microchip Technology Inc.

返回堆栈指针 （STKPTR） 90
访问栈顶 89
封装 525
    标识 525
    详细信息 526
复位 73, 403
    堆栈满 73
    堆栈下溢 73
    功耗管理模式下的 MCLR 复位 73
    看门狗定时器 （WDT） 73
    配置不匹配 （CM） 73
    欠压复位 （BOR） 73, 403
    RESET 指令 73
    上电复位 （POR） 73, 403
    上电延时定时器 （PWRT） 403
    振荡器起振定时器 （OST） 403
    正常工作状态下的 MCLR 复位 73
FSCM。*参见*故障保护时钟监视器。

G

高 / 低电压检测
    电流消耗 381
    典型应用 383
    复位的影响 384
    工作原理 380
        休眠期 384
    启动时间 381
    设置 381
    相关的寄存器 384
    应用 383
GOTO 452
功耗管理模式对各种时钟源的影响 55
功耗管理模式 57
    多条 Sleep 指令 58
    和 EUSART 操作 331
    和 PWM 工作原理 279
    和 SPI 模式的工作原理 289
    汇总 （表） 57
    进入 57
    空闲模式 62
        PRI_IDLE 63
        RC_IDLE 64
        SEC_IDLE 63
    时钟转换和状态指示位 58
    退出空闲和休眠模式 69
        通过复位 69
        通过 WDT 超时 69
        通过中断 69
        在没有振荡器起振延时的情况下退出 69
    休眠模式 62
        OSC1 和 OSC2 引脚状态 55
    选择 57
    运行模式 58
        PRI_RUN 58
        RC_RUN 60
        SEC_RUN 58
公式
    16 x 16 无符号乘法算法 140
    16 x 16 有符号乘法算法 140
    A/D 采集时间 362
    A/D 最小充电时间 362
    计算所需的最小采集时间 362
固件指令 431
故障保护时钟监视器 403, 424
    功耗管理模式下的中断 425
    POR 或从休眠中唤醒 425
    退出运行模式 424
    振荡器故障期间的 WDT 424

H

汇编器
    MPASM 汇编器 482
HLVD。*见*高 / 低电压检测。

I

I/O 端口 165
    漏极开路输出 167
    上拉配置 165
    引脚的驱动能力 165
    引脚输出驱动能力 165
I/O 引脚说明
    PIC18F6XK22 15
    PIC18F8XK22 14
I^2C 模式 （MSSP）
    波特率发生器 313
    串行时钟 （RC3/SCKx/SCLx） 299
    从模式 296
        地址掩码模式
            5 位 297
            7 位 298
        发送 299
        接收 299
        寻址 296
    带有 BRG 的 I^2C 时钟速率 313
    读 / 写位信息 （R/W 位） 296, 299
    多主器件模式 321
    多主器件通信、总线冲突与总线仲裁 321
    复位的影响 321
    工作原理 296
    寄存器 291
    时钟同步和 CKP 位 307
    时钟延长 306
        10 位从发送模式 306
        10 位从接收模式 （SEN = 1） 306
        7 位从发送模式 306
        7 位从接收模式 （SEN = 1） 306
    时钟仲裁 314
    停止条件时序 320
    相关的寄存器 326
    休眠状态下的操作 321
    应答序列时序 320
    支持广播呼叫地址 310
    主模式 311
        发送 317
        发送序列 312
        接收 317
        工作原理 312
        启动条件时序 315
        重复启动条件时序 316
    总线冲突
        停止条件期间 325
        在重复启动条件期间 324
ID 单元 403, 429
INCF 452
INCFSZ 453
INFSNZ 453

INTCON 寄存器
RBIF 位 172
INTOSC， INTRC。*请参考*内部振荡器模块。
IORLW 454
IORWF 454

J

集成电路。*参见* I²C。
寄存器
ADCON0 （A/D 控制寄存器 0） 352
ADCON1 （A/D 控制寄存器 1） 353
ADCON2 （A/D 控制寄存器 2） 354
ADRESH （A/D 结果寄存器高字节，右对齐，ADFM = 1） 357
ADRESH （A/D 结果寄存器高字节，左对齐，ADFM = 0） 356
ADRESL （A/D 结果寄存器低字节，右对齐，ADFM = 1） 357
ADRESL （A/D 结果寄存器高字节，左对齐，ADFM = 0） 356
ALRMCFG （闹钟配置） 231
ALRMDAY （闹钟日期值） 235
ALRMHR （闹钟小时值） 236
ALRMMIN （闹钟分钟值） 236
ALRMMNTH （闹钟月份值） 235
ALRMRPT （闹钟重复） 232
ALRMSEC （闹钟秒值） 236
ALRMWD （闹钟星期值） 235
ANCON0 （A/D 端口配置寄存器 0） 358
ANCON1 （A/D 端口配置寄存器 1） 358
ANCON2 （A/D 端口配置寄存器 2） 359
BAUDCONx （波特率控制寄存器） 330
保留 232
CCPRxH （CCPx 周期高字节） 248
CCPRxL （CCPx 周期低字节） 248
CCPTMRS0 （CCP 定时器选择 0） 261
CCPTMRS1 （CCP 定时器选择 1） 246
CCPTMRS2 （CCP 定时器选择 2） 247
CCPxCON （CCP4-CCP10 控制） 245
CCPxCON （增强型捕捉 / 比较 /PWMx 控制寄存器） 260
CMSTAT （比较器状态） 369
CMxCON （比较器控制 x） 368
CONFIG1H （配置 1 的高字节） 406
CONFIG1L （配置 1 的低字节） 405
CONFIG2H （配置 2 的高字节） 408
CONFIG2L （配置 2 的低字节） 407
CONFIG3H （配置 3 的高字节） 410
CONFIG3L （配置 3 的低字节） 409
CONFIG4L （配置 4 的低字节） 411
CONFIG5H （配置 5 的高字节） 413
CONFIG5L （配置 5 的低字节） 412
CONFIG6H （配置 6 的高字节） 415
CONFIG6L （配置 6 的低字节） 414
CONFIG7H （配置 7 的高字节） 417
CONFIG7L （配置 7 的低字节） 416
CTMUCONH （CTMU 控制寄存器高字节） 386
CTMUCONL （CTMU 控制寄存器低字节） 387
CTMUICON （CTMU 电流控制寄存器） 388
CVRCON （比较器参考电压控制） 375
DAY （日期值） 233
DEVID1 （器件 ID 寄存器 1） 418
DEVID2 （器件 ID 寄存器 2） 418
ECCPxAS （ECCPx 自动关闭控制） 273
ECCPxDEL （增强型 PWM 控制） 276
EECON1 （数据 EEPROM 控制 1) 113, 134
HLVDCON （高 / 低电压检测控制） 379
HOUR （小时值） 234
INTCON （中断控制） 143
INTCON2 （中断控制 2） 144
INTCON3 （中断控制 3） 145
IPR1 （外设中断优先级 1） 157
IPR2 （外设中断优先级 2） 158
IPR3 （外设中断优先级 3） 159
IPR4 （外设中断优先级 4） 159
IPR5 （外设中断优先级 5） 160
IPR6 （外设中断优先级 6） 161
MEMCON （外部存储器总线控制） 122
MINUTE （分钟值） 234
MONTH （月份值） 233
ODCON1 （外设漏极开路控制 1） 167
ODCON2 （外设漏极开路控制 2） 168
ODCON3 （外设漏极开路控制 3） 169
OSCCON （振荡器控制） 45
OSCCON2 （振荡器控制 2） 214
OSCTUNE （振荡器调节） 47
PADCFG1 （填充配置） 166
PADCFG1 （引脚配置） 230
PIE1 （外设中断允许 1） 152
PIE2 （外设中断允许 2） 153
PIE3 （外设中断允许 3） 154
PIE4 （外设中断允许 4） 154
PIE5 （外设中断允许 5） 155
PIE6 （外设中断允许 6） 156
PIR1 （外设中断请求 （标志） 1） 146
PIR2 （外设中断请求 （标志） 2） 147
PIR3 （外设中断请求 （标志） 3） 148
PIR4 （外设中断请求 （标志） 4） 149
PIR5 （外设中断请求 （标志） 5） 150
PIR6 （外设中断请求 （标志） 6） 150
PMD0 （外设模块禁止 0） 68
PMD1 （外设模块禁止 1） 67
PMD2 （外设模块禁止 2） 66
PMD3 （外设模块禁止 3） 65
PSPCON （并行从端口控制） 190
PSTRxCON （脉冲转向控制） 277
RCON （复位控制） 74, 162
RCSTAx （接收状态和控制寄存器） 329
REFOCON （参考振荡器控制） 54
RTCCAL （RTCC 校准） 230
RTCCFG （RTCC 配置） 229
SECOND （秒值） 234
SSPxCON1 （MSSPx 控制 1， I²C 模式） 293
SSPxCON1 （MSSPx 控制 1， SPI 模式） 283
SSPxCON2 （MSSPx 控制 2， I²C 从模式） 295
SSPxCON2 （MSSPx 控制 2， I²C 主模式） 294
SSPxMSK （I²C 从地址掩码， 7 位掩码模式） 295
SSPxSTAT （MSSPx 状态， I²C 模式） 292
SSPxSTAT （MSSPx 状态寄存器， SPI 模式） 282
STATUS 104
STKPTR （堆栈指针） 90
T0CON （Timer0 控制） 193
T1CON （Timer1 控制） 197
T1GCON （Timer1 门控 ） 198
T2CON （Timer2 控制） 209

© 2011 Microchip Technology Inc.

TxCON （Timerx 控制， Timer3/5/7） 212
TxCON （Timerx 控制， Timer4/6/8/10/12） 224
TxGCON （Timerx 门控） 213
TXSTAx （发送状态和控制寄存器） 328
WDTCON （看门狗定时器控制） 420
WEEKDAY （星期值） 233
YEAR （年份值） 232
寄存器汇总 ..... 98 – 103
计算 GOTO 91
间隔字符 （12 位）发送和接收 344
间接寻址 106
交流 （时序）特性 501
参数符号 501
器件时序规范的负载条件 502
时序条件 502
温度和电压规范 502
晶振 / 陶瓷谐振器 50
绝对最大值 485

K

开发支持 481
看门狗定时器 （WDT） 403, 419
编程注意事项 419
控制寄存器 420
相关的寄存器 420
振荡器故障期间 424
勘误表 8
客户通知服务 547
客户支持 547
快速寄存器堆栈 91
框图
16 位模式的 Timer0 194
8 位模式下的 Timer0 194
16 位字节写模式 125
16 位字节选择模式 127
16 位字写模式 126
8 位复用模式示例 129
A/D 360
A/D 结果对齐方式 355
半桥输出应用 275
比较模式工作原理 253, 264
比较器参考电压 376
比较器参考电压输出缓冲器示例 377
比较器简化 367
比较器模拟输入模型 370
比较器配置 372
表读操作 111
表写操作 112
波特率发生器 313
捕捉模式工作原理 251, 263
CTMU 385
CTMU 电流源校准电路 391
测量时间的 CTMU 典型连接和内部配置 398
差分通道测量 351
超低功耗唤醒初始化 70
单个比较器 370
单通道测量 351
读闪存程序存储器 115
对闪存程序存储器进行表写操作 117
EUSART 发送 338
EUSART 接收 340
故障安全时钟监视器 （FSCM） 424
INTIO1 振荡器模式 52
INTIO2 振荡器模式 52
简化转向 278
晶振 / 陶瓷谐振器工作原理 （HS 或 HSPLL 配置） .. 50
看门狗定时器 419
MSSP （SPI 模式） 281
MSSPx （I^2C 模式） 291
MSSPx （I^2C 主模式） 311
模拟输入模型 361
PIC18F6XK22 13
PIC18F8XK22 14
PLL 51
PORTD 和 PORTE （并行从端口） 189
PWM 工作原理 （简化） 255
片上复位电路 73
片上稳压器的连接 421
器件时钟 44
全桥应用示例 269
RCIO 振荡器模式 49
RC 振荡器模式 49
RTCC 227
RTCC 时钟源复用 238
SOSC 低功耗振荡器的外部组件 201
SPI 主 / 从器件连接 285
生成脉冲延时的 CTMU 典型连接和内部配置 399
使用漏极开路输出 （USART） 167
使用外部输入的高 / 低电压检测 380
Timer2 210
Timer4 224
Timer3/5/7 215
Timer1 200
通用 I/O 端口的工作原理 165
外部上电复位电路 （V_{DD} 慢速上电 ） 75
外部时钟输入工作原理 （EC） 51
外部时钟输入工作原理 （HS） 51
增强型 PWM 模式 265
中断逻辑 142
扩展指令集
ADDFSR 474
ADDULNK 474
CALLW 475
MOVSF 475
MOVSS 476
PUSHL 476
SUBFSR 477
SUBULNK 477

L

LFSR 455
立即数变址寻址
和标准 PIC18 指令 478
立即数变址寻址模式 478

M

脉冲转向 276
脉宽调制。*参见* PWM （CCP 模块）。
默认系统时钟 49
模数转换器。*参见* A/D。
MOVF 455
MOVFF 456
MOVLB 456
MOVLW 457
MOVSF 475
MOVSS 476

MOVWF 457
MPLAB ASM30 汇编器、链接器和库管理器 482
MPLAB PM3 器件编程器 484
MPLAB REAL ICE 在线仿真器系统 483
MPLAB 集成开发环境软件 481
MPLINK 目标链接器 /MPLIB 目标库管理器 482
MSSP
    ACK 脉冲 296, 299
    I²C 模式。*参见* I²C 模式。
    模块概述 281
    SPI 主 / 从器件连接 285
    时钟移位的 TMRx 输出 224
MULLW 458
MULWF 458

N

NEGF 459
内部 RC 振荡器
    与 WDT 一起使用 419
内部稳压器规范 500
内部振荡器模块 52
    调节 53
    INTIO 模式 52
    INTOSC
        频率漂移 53
        输出频率 53
    INTPLL 模式 52
内核特性
    扩展指令集 9
    纳瓦技术 9
    内存选项 9
    外部存储器总线 （EMB） 9
    易于移植 10
    振荡器选项和特性 9
NOP 459

P

P1A/P1B/P1C/P1D。*参见*增强型捕捉 / 比较 /PWM （ECCP） 265
配置不匹配 （CM）复位 76
配置寄存器保护 429
配置位 403
PIC18F87K22、 PIC18F87J11 和 PIC18F8722 系列之间的显著差异 534
片上
    稳压器 421
        禁止 421
        使能 421
        休眠 422
PLL 51
    HSPLL 和 ECPLL 振荡器模式 51
    与 HF-INTOSC 一起使用 51
POP 460
POR。*请参见*上电复位。
PORTA
    LATA 寄存器 170
    PORTA 寄存器 170
    TRISA 寄存器 170
    相关的寄存器 171
PORTB
    LATB 寄存器 172
    PORTB 寄存器 172
    RB7:RB4 电平变化中断标志 （RBIF 位） 172
    TRISB 寄存器 172
    相关的寄存器 173
PORTC
    LATC 寄存器 174
    PORTC 寄存器 174
    RC3/SCKx/SCLx 引脚 299
    TRISC 寄存器 174
    相关的寄存器 175
PORTD
    LATD 寄存器 176
    PORTD 寄存器 176
    TRISD 寄存器 176
    相关的寄存器 177
PORTE
    LATE 寄存器 178
    PORTE 寄存器 178
    RE0/P2D/RD/AD8 引脚 189
    RE1/P2C/WR/AD9 引脚 189
    RE2/P2B/CCP10/CS/AD10 引脚 189
    TRISE 寄存器 178
    相关的寄存器 180
PORTF
    LATF 寄存器 181
    PORTF 寄存器 181
    TRISF 寄存器 181
    相关的寄存器 182
PORTG
    LATG 寄存器 183
    PORTG 寄存器 183
    TRISG 寄存器 183
    相关的寄存器 184
PORTH
    LATH 寄存器 185
    PORTH 寄存器 185
    TRISH 寄存器 185
    相关的寄存器 187
PORTJ
    LATJ 寄存器 187
    PORTJ 寄存器 187
    TRISJ 寄存器 187
    相关的寄存器 188
PRI_IDLE 模式 63
PRI_RUN 模式 58
PSP。*请参见*并行从端口。
PWM （CCP 模块）
    频率 / 分辨率示例 256
    设置 PWM 操作 256
    TMR2 与 PR2 匹配 255
    相关的寄存器 256
    占空比 256
    周期 255
PWM （ECCP 模块）
    复位的影响 279
    故障保护时钟监视器的相关操作 279
    脉冲转向模式 276
    在功耗管理模式下工作 279
    转向同步 278
PWM 模式。*参见*增强型捕捉 / 比较 /PWM 265
PUSH 460
PUSH 和 POP 指令 90
PUSHL 476

© 2011 Microchip Technology Inc.

Q

Q 时钟 256
器件概述 9
特性 （64 引脚器件） 12
特性 （80 引脚器件） 12
欠压复位 （BOR） 75
检测 75

R

RAM。*参见*数据存储器。
RC_IDLE 模式 64
RC_RUN 模式 60
RCALL 461
RCON 寄存器
初始化期间的位状态 78
RESET 461
RETFIE 462
RETLW 462
RETURN 463
RLCF 463
RLNCF 464
RRCF 464
RRNCF 465
RTCC
复位 242
器件 242
上电复位 （POR） 242
工作原理 237
常规功能 239
寄存器读写的安全窗口 239
寄存器映射 239
ALRMVAL 240
RTCVAL 239
进位规则 238
闰年 239
时钟源 238
校准 240
写锁定 239
寄存器接口 237
寄存器映射 243
控制寄存器 229
闹钟 240
配置 240
掩码设置 241
中断 241
闹钟值寄存器 （ALRMVALL 和 ALRMVALH)） 235
相关控制寄存器 243
相关闹钟值寄存器 243
相关值寄存器 243
休眠模式 242
值寄存器 （RTCVAL） 232
RTCEN 位写入 237
软件模拟器 （MPLAB SIM） 483

S

SCKx 281
SDIx 281
SDOx 281
SEC_IDLE 模式 63
SEC_RUN 模式 58
SETF 465
所有寄存器的初始化状态 ..... 79 – 86
闪存程序存储器 111
表读和表写 111
表指针
基于不同操作的边界 114
擦除 116
擦除序列 116
代码保护期间的操作 120
读 115
控制寄存器 112
EECON1 和 EECON2 112
TABLAT （表锁存）寄存器 114
TBLPTR （表指针）寄存器 114
TBLPTR （表指针）寄存器，边界 114
相关的寄存器 120
写 117
防止误写操作 120
写校验 120
意外终止 120
写序列 118
上电复位 （POR） 75
上电延时 55
上电延时定时器 （PWRT） 55, 76
延时时序 76
上电延时定时器 （PWRT） 55
实时时钟和日历 （RTCC） 227
寄存器 228
时序图
A/D 转换 524
半桥 PWM 输出 268, 275
并行从端口 （PSP）读 191
并行从端口 （PSP）写 190
BRG 溢出时序 336
捕捉 / 比较 /PWM 513
CLKO 和 I/O 505
重复启动条件 316
重复启动条件期间的总线冲突 （情形 1） 324
重复启动条件期间的总线冲突 （情形 2） 324
从空闲模式唤醒进入运行模式的转换 63
从 RC_RUN 模式到 PRI_RUN 模式的转换 61
从 SEC_RUN 模式到 PRI_RUN 模式 （HSPLL）
的转换 59
从同步 287
从休眠模式唤醒的转换 （HSPLL） 62
程序存储器读 507
程序存储器取指 （8 位） 506
程序存储器写 508
带有时钟仲裁的波特率发生器 314
到 RC_RUN 模式的转换 61
低电压检测工作原理 （VDIRMAG=0） 382
第一个启动位的时序 315
定时器脉冲生成 242
EUSART/AUSART 同步接收 （主 / 从） 522
EUSART 同步发送 （主 / 从） 522
发送和应答时的总线冲突 321
发送间隔字符序列 344
复位、看门狗定时器 （WDT）、振荡器起振定时器
（OST）和上电延时定时器 （PWRT） 509
高电压检测工作原理 （VDIRMAG=1） 383
故障安全时钟监视器 （FSCM） 425
HLVD 特性 511
缓慢上升时间 （$\overline{\text{MCLR}}$ 连接到 VDD， VDD 上升时间 >
TPWRT） 77

I²C 从模式 （7 位或 10 位发送） 318
I²C 从模式 （7 位接收， SEN = 0） 300
I²C 从模式 （7 位接收， SEN = 0 且
ADMSK = 01011） 301
I²C 从模式广播呼叫地址时序 （7 或 10 位
寻址模式） 310
I²C 停止条件接收或发送模式 320
I²C 应答序列 320
I²C 主模式 （7 位接收） 319
I²C 总线启动 / 停止位 518
I²C 总线数据 519
进入空闲模式的转换 63
进入 SEC_RUN 模式的转换 59
进入休眠模式的转换 62
MSSP I²C 总线启动 / 停止位 520
MSSP I²C 总线数据 520
PWM 方向更改 271
PWM 输出 255
PWM 输出 （低电平有效） 267
PWM 输出 （高电平有效） 266
PWM 自动关闭 （PxRSEN = 0，用固件重启） 274
PWM 自动关闭 （PxRSEN = 1，使能自动重启） 274
启动条件期间的总线冲突 （仅 SDA） 322
启动条件期间的总线冲突 （SCLx=0） 323
启动条件期间由 SDAx 仲裁引起的 BRG 复位 323
上电延时时序 （MCLR 连接到 VDD， VDD 电压上升时间
Tpwrt） 76
上电延时时序 （MCLR 未连接到 VDD），情形 1..... 77
上电延时时序 （MCLR 未连接到 VDD），情形 2..... 77
时钟 / 指令周期 92
时钟同步 307
双速起动时钟转换 （从 INTOSC 切换到 HSPLL） ... 423
全桥 PWM 输出 270
欠压复位 （BOR）..... 509
SLEEP 的外部存储器总线
（扩展单片机模式）..... 128, 130
SPI 模式 （从模式， CKE=0） 288
SPI 模式 （从模式， CKE=1） 288
SPI 模式 （主模式） 286
SPI 从模式示例 （CKE=0） 516
SPI 从模式示例 （CKE=1） 517
SPI 主模式示例 （CKE=0） 514
SPI 主模式示例 （CKE=1） 515
TBLRD 的外部存储器总线
（扩展单片机模式）..... 128, 130
Timer0 和 Timer1 外部时钟 512
TIMER3/5/7 门控计数使能模式 217
Timer3/5/7 门控单脉冲 / 翻转组合模式 220
Timer3/5/7 门控单脉冲模式 219
Timer3/5/7 门控翻转模式 218
Timer1 门控单脉冲 / 翻转组合模式 207
Timer1 门控单脉冲模式..... 206
Timer1 门控翻转模式 205
Timer1 门控计数使能模式 204
停止条件期间的总线冲突 （情形 1） 325
停止条件期间的总线冲突 （情形 2） 325
同步发送..... 345
同步发送 （由 TXEN 位控制） 346
同步主接收模式 （由 SREN 位控制） 347
外部时钟..... 503
休眠模式下的自动唤醒位 （WUE） 343
异步发送..... 338
异步发送 （背对背）..... 338
异步接收 341
正常工作状态下的自动唤醒位 （WUE） 343
指令周期结束时的转向事件示例 （STRSYNC = 0） 278
指令周期开始时的转向事件 （STRSYNC = 1） 278
自动波特率计算 336
在占空比接近 100% 时更改 PWM 方向 272
时序图和规范
捕捉 / 比较 /PWM 要求 513
CLKO 和 I/O 要求 505, 507
程序存储器取指要求 （8 位）..... 506
程序存储器写要求 508
EUSART/AUSART 同步发送要求 522
EUSART/AUSART 同步接收要求 522
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 510
HLVD 特性 511
I²C 总线启动 / 停止位要求 （从模式） 518
I²C 总线数据要求 （从模式）..... 519
MSSP I²C 总线启动 / 停止位要求 520
MSSP I²C 总线数据要求 521
内部 RC 精度 （INTOSC）..... 504
PLL 时钟 504
SPI 从模式要求示例 （CKE=1） 517
SPI 模式要求示例 （从模式， CKE= 0） 516
SPI 模式要求示例 （主模式， CKE=0） 514
SPI 模式要求示例 （主模式， CKE=1） 515
Timer0 和 Timer1 外部时钟要求 512
外部时钟要求 503
时钟源..... 48
复位后的默认系统时钟 49
选择 48
数据存储器..... 94
存储器映射
PIC18FX5K22/X7KJ22 器件 95
特殊功能寄存器 97
存储区选择寄存器 （BSR） 94
快速操作存储区 96
扩展指令集 108
特殊功能寄存器 （SFR） 97
通用寄存器 96
数据 EEPROM 代码保护 429
数据 EEPROM 存储器
代码保护期间的操作 137
读 135
EEADR 和 EEADRH 寄存器 133
EECON1 和 EECON2 寄存器 133
防止误写入操作 137
概述 133
使用 137
相关的寄存器 138
写 135
写校验 135
数据寻址模式 105
比较使能了扩展的指令集的寻址模式 109
变址立即数偏移 108
BSR 110
映射快速操作存储区 110
影响的指令 108
固有和立即数 105
间接寻址 105
直接寻址 105

© 2011 Microchip Technology Inc.

双速启动 403, 423
IESO （CONFIG1H），内部 / 外部振荡器切换位 406
双字指令
示例情形 93
SLEEP 466
SPI 模式 （MSSP） 281
串行时钟 281
串行数据输出 281
串行数据输入 281
从模式 287
从选择 281
从选择同步 287
典型连接 285
复位的影响 289
工作原理 284
SPI 时钟 286
SSPxBUF 寄存器 286
SSPxSR 寄存器 286
使能 SPI I/O 285
时钟速率，互连 289
相关的寄存器 290
在功耗管理模式下的操作 289
主 / 从器件连接 285
主模式 286
总线模式兼容性 289
SSPOV 317
SSPOV 状态标志 317
SSPxSTAT 寄存器
____ R/W 位 296, 299
SSx 281
SWAPF 468
SUBFSR 477
SUBFWB 466
SUBLW 467
SUBWF 467
SUBWFB 468
SUBULNK 477

T

TBLRD 469
TBLWT 470
特殊事件触发器。*参见*比较 （ECCP 模式）
特殊事件触发器。*参见*比较 （CCP 模块）。
Timer0 193
16 位读写模式 194
工作原理 194
时钟源边沿选择位 （T0CS 位） 194
时钟源选择位 （T0CS 位） 194
相关的寄存器 195
溢出中断 195
预分频器 195
切换分配 195
预分频器分配 （PSA 位） 195
预分频器选择 （T0PS2:T0PS0 位） 195
Timer2 209
工作原理 209
PR2 寄存器 255
输出 210
TMR2 与 PR2 匹配中断 255
相关的寄存器 210
中断 210
Timer4
MSSP 时钟移位 224
Timer3/5/7 211
16 位读 / 写模式 216
工作原理 215
门控 217
使用 SOSCO 振荡器作为时钟源 216
TMRxL 寄存器 211
特殊事件触发器 （ECCP） 221
相关的寄存器 222
溢出中断 211, 221
振荡器 211
Timer4/6/8/10/12 223
工作原理 223
后分频器。*请参见*后分频器， Timer4/6/8/10/12。
PRx 寄存器 223
输出 224
TMRx 寄存器 223
相关的寄存器 225
预分频器。*请参见*预分频器， Timer4/6/8/10/12。
中断 224
Timer1 197
16 位读 / 写模式 201
复位，使用 ECCP 特殊事件触发器 203
工作原理 199
门控 203
SOSC 振荡器 201
使用 SOSC 作为时钟源 202
时钟源选择 199
TMR1L 寄存器 197
TMR1H 寄存器 197
相关的寄存器 207
振荡器 197
SOSC 布线注意事项 202
振荡器 , 作为辅助时钟 48
中断 202
同步间隔字符自动唤醒 342
TSTFSZ 471
TXSTAx 寄存器
BRGH 位 331

W

外部存储器总线 （EMB） 121
16 位模式时序 128
16 位数据宽度模式 124
16 位字节写模式 125
16 位字节选择模式 127
16 位字写模式 126
8 位模式时序 130
8 位数据宽度模式 129
不同地址和数据宽度的地址和数据线 （表） 123
程序存储器模式 124
单片机 124
扩展单片机 124
等待状态 124
地址和数据宽度 123
地址移位 123
端口引脚上的弱上拉电路 124
I/O 端口功能 121
控制 122
相关的寄存器 131
在功耗管理模式下工作 131
外部振荡器模式
HS 50
时钟输入 （EC 模式） 51

WCOL 315, 316, 317, 320
WCOL 状态标志 315, 316, 317, 320
WWW 在线技术支持 8

X

系列中各器件的详细信息 11
XORLW 471
XORWF 472
选择外设模块控制 64

Y

引脚功能
AVDD 23, 36
AVSS 23, 36
ENVREG 23, 36
MCLR/RG5 ..... 15, 24
OSC1/CLKI/RA7 ..... 15, 24
OSC2/CLKO/RA6 ..... 15, 24
RA0/AN0/ULPWU 16, 25
RA1/AN1 ..... 16, 25
RA2/AN2/VREF- 16, 25
RA3/AN3/VREF+ 16, 25
RA4/T0CKI 16, 25
RA5/AN4/T1CKI/T3G/HLVDIN 16, 25
RB0/INT0/FLT0 26
RB0/INT0/FLTO 17
RB1/INT1 ..... 17, 26
RB2/INT2/CTED1 ..... 17, 26
RB3/INT3/CTED2/ECCP2/P2A 17, 26
RB4/KBI0 ..... 17, 26
RB5/KBI1/T3CKI/T1G 17, 26
RB6/KBI2/PGC 17, 26
RB7/KBI3/PGD 17, 26
RC0/SOSCO/SCKLI 27
RC0/SOSCO/SCLKI 18
RC1/SOSCI/ECCP2/P2A 18, 27
RC2/ECCP1/P1A 18, 27
RC3/SCK1/SCL1 ..... 18, 27
RC4/SDI1/SDA1 ..... 18, 27
RC5/SDO1 ..... 18, 27
RC6/TX1/CK1 ..... 18, 27
RC7/RX1/DT1 ..... 18, 27
RD0/PSP0/CTPLS 19
RD0/PSP0/CTPLS/AD0 28
RD1/PSP1/T5CKI/T7G 19
RD1/T5CKI/T7G/PSP1/AD1 28
RD2/PSP2 19
RD2/PSP2/AD2 28
RD3/PSP3 19
RD3/PSP3/AD3 28
RD4/PSP4/SDO2 19
RD4/SDO2/PSP4/AD4 28
RD5/PSP5/SDI2/SDA2 19
RD5/SDI2/SDA2/PSP5/AD5 28
RD6/PSP6/SCK2/SCL2 19
RD6/SCK2/SCL2/PSP6/AD6 29
RD7/PSP7/SS2 19
RD7/SS2/PSP7/AD7 29
RE0/P2D/RD/AD8 30
RE0/RD/P2D 20
RE1/P2C/WR/AD9 30
RE1/WR/P2C 20
RE2/CS/P2B/CCP10 20
RE2/P2B/CCP10/CS/AD10 30
RE3/P3C/CCP9/REFO 20
RE3/P3C/CCP9/REFO/AD11 30
RE4/P3B/CCP8 20
RE4/P3B/CCP8/AD12 30
RE5/P1C/CCP7 20
RE5/P1C/CCP7/AD13 30
RE6/P1B/CCP6 20
RE6/P1B/CCP6/AD14 31
RE7/ECCP2/P2A 20
RE7/ECCP2/P2A/AD15 31
RF1/AN6/C2OUT/CTDIN 21, 32
RF2/AN7/C1OUT 21, 32
RF3/AN8/C2INB/CTMUI 21, 32
RF4/AN9/C2INA 21, 32
RF5/AN10/C1INB 32
RF5/AN10/CVREF/C1INB 21
RF6/AN11/C1INA 21, 32
RF7/AN5/SS1 ..... 21, 32
RG0/ECCP3/P3A 22, 33
RG1/TX2/CK2/AN19/C3OUT 22, 33
RG2/RX2/DT2/AN18/C3INA 22, 33
RG3/CCP4/AN17/P3D/C3INB 22, 33
RG4/RTCC/T7CKI/T5G/CCP5/AN16/P1D/C3INC 22, 33
RH0/AN23/A16 34
RH1/AN22/A17 34
RH2/AN21/A18 34
RH3/AN20/A19 34
RH4/CCP9/P3C/AN12/C2INC 34
RH5/CCP8/P3B/AN13/C2IND 34
RH6/CCP7/P1C/AN14/C1INC 34
RH7/CCP6/P1B/AN15 35
RJ0/ALE 36
RJ1/OE 36
RJ2/WRL 36
RJ3/WRH 36
RJ4/BA0 36
RJ5/CE 36
RJ6/LB 36
RJ7/UB 36
VDD 23, 36
VDDCORE/VCAP 23, 36
VSS 23, 36
硬件乘法器 139
8 x 8 乘法算法 139
工作原理 139
性能比较（表） 139
与捕捉、比较和 Timer1/3/5/7
相关的寄存器 253
预分频器， Timer0 195
预分频器， Timer2 256

Z

在线串行编程（ICSP） 403, 429
在线调试器 429
增强型捕获 / 比较 /PWM（ECCP）和 Timer1/2/3/4/6/8/10/12
相关的寄存器 279
增强型捕捉 / 比较 /PWM（ECCP） 259
比较模式。*参见*比较。
捕捉模式。*参见*捕捉。
定时器资源 262
输出和配置 262

© 2011 Microchip Technology Inc.

增强型 PWM 模式 265
半桥模式 268
半桥输出应用 268
半桥输出应用示例 275
可编程死区延时 275
启动注意事项 272
全桥模式 269
全桥输出模式下的方向更改 271
全桥输出应用 269
输出关系（高电平有效和低电平有效）..... 266
输出关系图 267
直通电流 275
自动关闭 272
自动重启 274
增强型通用同步 / 异步收发器（EUSART）。*参见* EUSART。
振荡器， Timer3/5/7 211
振荡器， Timer1 197
振荡器配置 43
EC 43
ECIO 43
HS 43
INTIO1 43
INTIO2 43
LP 43
内部振荡器模块 52
RC 43
RCIO 43
XT 43
振荡器起振定时器（OST）..... 55
振荡器切换 48
振荡器选择 403
振荡器转换 49
直接寻址 106
指令集 431
ADDLW 437
ADDWF 437
ADDWF（立即数变址寻址模式）..... 479
ADDWFC 438
ANDLW 438
ANDWF 439
BC 439
BCF 440
BN 440
BNC 441
BNN 441
BNOV 442
BNZ 442
BOV 445
BRA 443
BSF 443
BSF（立即数变址寻址模式）..... 479
BTFSC 444
BTFSS 444
BTG 445
BZ 446
标准指令 431
CALL 446
CLRF 447
CLRWDT 447
COMF 448
CPFSEQ 448
CPFSGT 449
CPFSLT 449
操作码字段说明 432
DAW 450
DCFSNZ 451
DECF 450
DECFSZ 451
GOTO 452
INCF 452
INCFSZ 453
INFSNZ 453
IORLW 454
IORWF 454
扩展指令 473
使能时注意事项 478
使用 MPLAB IDE 工具 480
语法 473
LFSR 455
MOVF 455
MOVFF 456
MOVLB 456
MOVLW 457
MOVWF 457
MULLW 458
MULWF 458
NEGF 459
NOP 459
POP 460
PUSH 460
RCALL 461
RESET 461
RETFIE 462
RETLW 462
RETURN 463
RLCF 463
RLNCF 464
RRCF 464
RRNCF 465
SETF 465
SETF（立即数变址寻址模式）..... 479
SLEEP 466
SWAPF 468
SUBFWB 466
SUBLW 467
SUBWF 467
SUBWFB 468
TBLRD 469
TBLWT 470
TSTFSZ 471
XORLW 471
XORWF 472
一般格式 433
指令周期 92
时钟机制 92
指令流 / 流水线 92
直流特性
CTMU 电流源 498
掉电电流和供电电流 488
供电电压 487
PIC18F87K22 系列（工业级）..... 497
直通电流 275
中断 141
INTx 引脚 163

PORTB，电平变化中断 163
期间，现场保护 163
TMR0 163
相关的寄存器 164
中断，标志位
电平变化中断 （RB7:RB4）标志 （RBIF 位） 172
中断源 403
A/D 转换完成..... 361
比较完成 （CCP）..... 252
比较完成 （ECCP）..... 264
捕捉完成 （CCP）..... 251
捕捉完成 （ECCP）..... 263
电平变化中断 （RB7:RB4）..... 172
TMR0 溢出 195
TMR1 溢出 202
TMR2 与 PR2 匹配 （PWM）..... 255
TMRx 溢出 211, 221
主复位 （MCLR） 75
主同步串行端口 （MSSP）。*参见* MSSP。

© 2011 Microchip Technology Inc.

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、应用笔记和样本程序、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: PIC18F87K22 系列　　文献编号: DS39960D_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？

© 2011 Microchip Technology Inc.

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销售办事处联系。

部件编号	X	/XX	XXX
器件	温度范围	封装	定制编号

器件 (1,2)	PIC18F65K22 和 PIC18F65K22T PIC18F66K22 和 PIC18F66K22T PIC18F67K22 和 PIC18F67K22T PIC18F85K22 和 PIC18F85K22T PIC18F86K22 和 PIC18F86K22T PIC18F87K22 和 PIC18F87K22T
温度范围	I = -40°C 到 +85°C （工业级） E = -40°C 到 +125°C （扩展级）
封装	PT = TQFP （塑封薄型四方扁平封装） MR = QFN （塑封四方扁平封装）
定制编号	QTP、 SQTP、编码或特殊要求 （其他情况均为空白）

示例：

a) PIC18F87K22-I/PT 301 = 工业级温度范围，TQFP 封装， QTP 的定制编号为 #301。

b) PIC18F87K22T-I/PT = 卷带式，工业级温度范围， TQFP 封装。

c) PIC18F87K22T-E/PT = 卷带式，扩展级温度范围， TQFP 封装

注 1：F = 标准电压范围
2：T = 仅为卷带式 PLCC 和 TQFP 封装
3：RSL = 硅片版本 A3

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: 86-571-2819-3187
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: 886-7-536-4818
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: 886-3-5778-366
Fax: 886-3-5770-955

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 - Osaka
Tel: 81-66-152-7160
Fax: 81-66-152-9310

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

11/29/11

© 2009-2011 Microchip Technology Inc.