PIC18F87J10 系列

数据手册

采用纳瓦技术的 64/80 引脚
高性能 1 Mb 闪存单片机

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下， Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿与那些注重代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是 “牢不可破”的。

代码保护功能处于持续发展中。 Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了《数字器件千年版权法案 （Digital Millennium Copyright Act）》。如果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

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

Microchip 位于美国亚利桑那州 Chandler 和 Tempe 及位于加利福尼亚州 Mountain View 的全球总部、设计中心和晶圆生产厂均于 2003 年 10 月通过了 ISO/TS-16949:2002 质量体系认证。公司在 PICmicro® 8 位单片机、KEELOQ® 跳码器件、串行 EEPROM、单片机外设、非易失性存储器和模拟产品方面的质量体系流程均符合 ISO/TS-16949:2002。此外，Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。

PIC18F87J10 系列

采用纳瓦技术的 64/80 引脚高性能 1 Mb 闪存单片机

单片机的特殊功能：

- 工作电压范围：2.0V 至 3.6V
- 5.5V 容错输入 （仅数字引脚）
- 片上 2.5V 稳压器
- 低功耗、高速 CMOS 闪存技术
- 优化的 C 编译器架构：
  - 可选的扩展指令集，可用于优化重入代码
- 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）：
  - 4 ms 到 131s 的可编程时间
- 通过两个引脚进行单电源供电在线串行编程 （In-Circuit Serial Programming™， ICSP™）
- 通过两个引脚，使用三个断点进行在线调试 （In-Circuit Debug， ICD）
- 功耗管理模式：
  - 运行：打开 CPU 和外设
  - 空闲：关闭 CPU，打开外设
  - 休眠：关闭 CPU 和外设
- 闪存具有自写入能力

灵活的振荡器结构：

- 两种晶振模式，频率最高可达 40 MHz （V_{DD} > 2.15V）
- 4 倍频锁相环 （**Phase Lock Loop， PLL**）
- 两种外部时钟模式，频率最高可达 40MHz
- 31 kHz 内部振荡器
- 辅助振荡器使用 Timer1@32kHz
- 双速振荡器起振
- 故障保护时钟监视器：
  - 在外部时钟停止时允许安全关闭器件

外设特点：

- 灌电流 / 拉电流峰值为 25 mA/25 mA （PORTB 和 PORTC）
- 4 个可编程外部中断
- 4 个输入电平变化中断
- 2 个捕获 / 比较 /PWM （CCP）模块
- 3 个增强型捕获 / 比较 /PWM （ECCP）模块：
  - 1 个、 2 个或 4 个 PWM 输出
  - 可选择的极性
  - 可编程的死区时间
  - 自动关闭和自动重启
- 2 个主控同步串行端口 （Master Synchronous Serial Port， MSSP）模块，支持共 4 种模式下的 3 线 SPI™ 以及 I^2C™ 模块的主控和从动模式
- 两个增强型可寻址 USART 模块：
  - 支持 RS-485、 RS-232 和 LIN 1.2
  - 起始位上自动唤醒
  - 自动波特率检测
- 多达 15 路通道的 10 位模数转换器模块 （A/D）：
  - 自动采集功能
  - 休眠模式下可进行转换
  - 自动校准功能
- 2 个带输入复用的模拟比较器

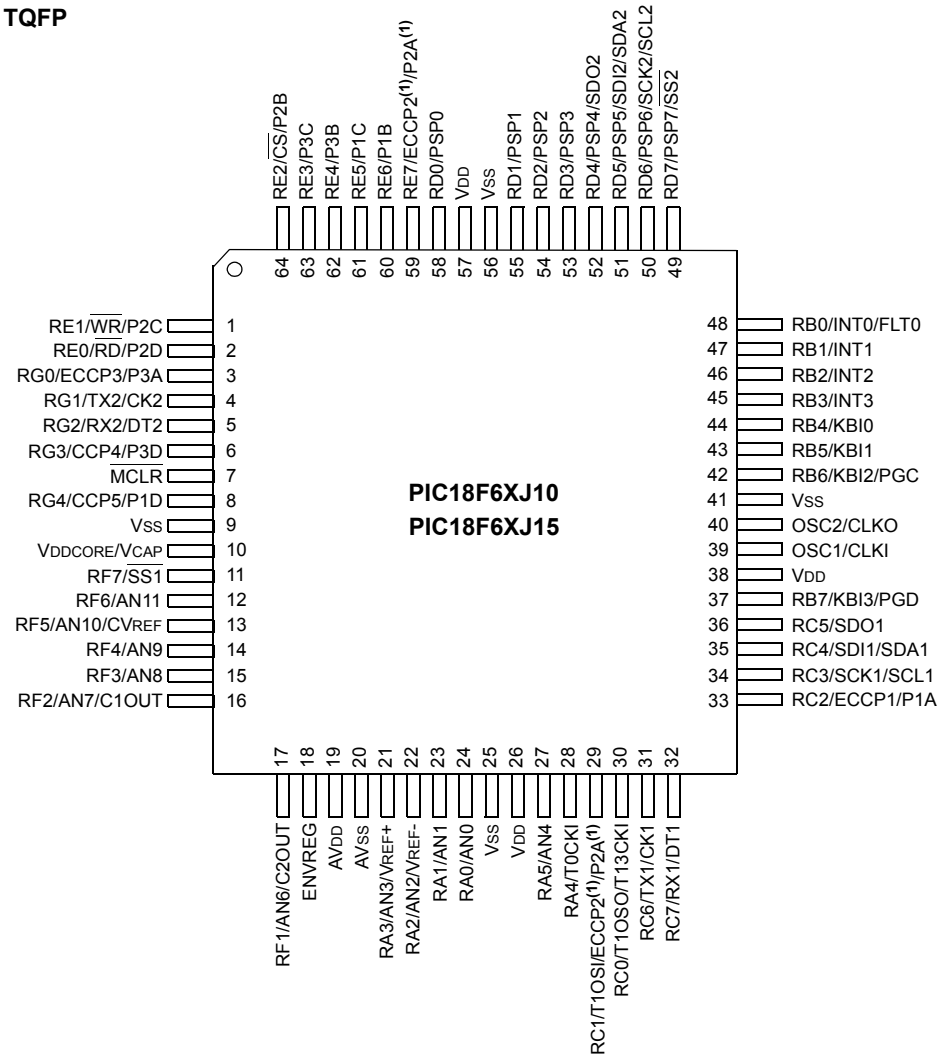
外部存储总线（仅 PIC18F8XJ10/8XJ15 器件）：

- 寻址能力最高可达 2 MB
- 8 位或 16 位接口
- 12 位、 16 位和 20 位寻址模式

器件	程序存储器		SRAM 数据存储器（字节）	I/O	10 位 A/D 通道数	CCP/ ECCP（PWM）	MSSP		EUSART	比较器	定时器 8/16 位	外部总线
	闪存（字节）	# 单字指令						SPI™	主控 I^2C™				
PIC18F65J10	32K	16384	2048	50	11	2/3	2	有	有	2	2	2/3	无
PIC18F65J15	48K	24576	2048	50	11	2/3	2	有	有	2	2	2/3	无
PIC18F66J10	64K	32768	2048	50	11	2/3	2	有	有	2	2	2/3	无
PIC18F66J15	96K	49152	3936	50	11	2/3	2	有	有	2	2	2/3	无
PIC18F67J10	128K	65536	3936	50	11	2/3	2	有	有	2	2	2/3	无
PIC18F85J10	32K	16384	2048	66	15	2/3	2	有	有	2	2	2/3	有
PIC18F85J15	48K	24576	2048	66	15	2/3	2	有	有	2	2	2/3	有
PIC18F86J10	64K	32768	2048	66	15	2/3	2	有	有	2	2	2/3	有
PIC18F86J15	96K	49152	3936	66	15	2/3	2	有	有	2	2	2/3	有
PIC18F87J10	128K	65536	3936	66	15	2/3	2	有	有	2	2	2/3	有

引脚图

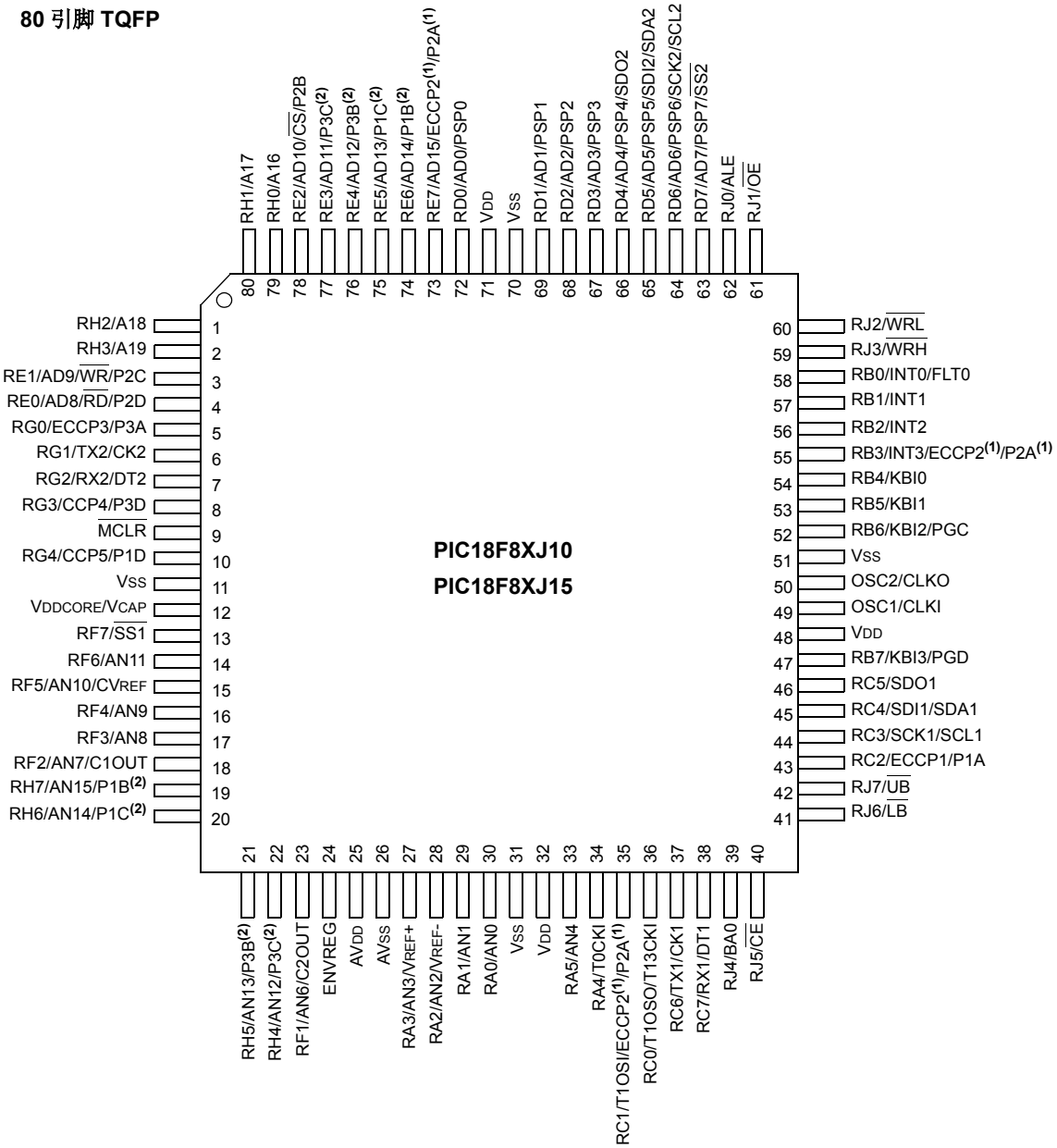
64 引脚 TQFP



注 **1**：ECCP2/P2A 引脚位置取决于 CCP2MX 配置位的设置。

引脚图 （续）

80 引脚 TQFP



注 **1：** CCP2/P2A 引脚位置取决于 CCP2MX 配置位和程序存储器模式的设置。

2： P1B、 P1C、 P3B 和 P3C 引脚位置取决于 ECCPMX 配置位的设置。

目录

1.0 器件概述 5
2.0 振荡器配置 27
3.0 功耗管理模式 35
4.0 复位 43
5.0 存储器构成 55
6.0 闪存程序存储器 81
7.0 外部存储总线 91
8.0 8 x 8 硬件乘法器 103
9.0 中断 105
10.0 I/O 端口 121
11.0 Timer0 模块 147
12.0 Timer1 模块 151
13.0 Timer2 模块 157
14.0 Timer3 模块 159
15.0 Timer4 模块 163
16.0 捕捉 / 比较 /PWM （CCP）模块 165
17.0 增强型捕捉 / 比较 /PWM （ECCP）模块 173
18.0 主控同步串行端口 （MSSP）模块 189
19.0 增强型通用同步 / 异步收发器 235
20.0 10 位模数转换器 （A/D）模块 257
21.0 比较器模块 267
22.0 比较器参考电压模块 273
23.0 CPU 的特殊功能 277
24.0 指令集综述 289
25.0 开发支持 339
26.0 电气规范 343
27.0 封装信息 379
附录 A： 在高端器件系列间移植 383
索引 385
Microchip 网站 397
变更通知客户服务 397
客户支持 397
读者反馈表 398
产品标识体系 399

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。

1.0 器件概述

该文档包含以下器件的特定信息：

- PIC18F65J10
- PIC18F65J15
- PIC18F66J10
- PIC18F66J15
- PIC18F67J10
- PIC18F85J10
- PIC18F85J15
- PIC18F86J10
- PIC18F86J15
- PIC18F87J10

PIC18F87J10 是新面世的低压产品系列，在保留 PIC18 单片机的传统优点（即出色的计算性能以及丰富的功能集）的同时，性价比极高。这些功能使得 PIC18F87J10 系列成为许多高性能、低成本应用的理想选择。

1.1 内核功能

1.1.1 纳瓦技术

PIC18F87J10 系列的所有器件都具有一系列能显著降低工作功耗的功能。主要包括以下几项：

- **备用运行模式：**通过将 Timer1 或内部振荡模块作为单片机时钟源，可使代码执行时的功耗大约降低 90%。
- **多种空闲模式：**单片机还可在其 CPU 内核禁止而外设仍然运行的情况下运行。处于这些状态时，功耗能降得更低，只有正常工作所需的 4%。
- **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，允许用户将节能理念融入到他们应用的软件设计中。

1.1.2 振荡器选项和功能

PIC18F87J10 系列的所有器件提供 5 个不同的振荡器选择，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：

- 2 个晶振模式，使用晶振或陶瓷谐振器。
- 两个外部时钟模式，提供 4 分频时钟输出选项。
- 一个锁相环 （PLL）倍频器，可在外部振荡器模式下使用，允许时钟速度最高达 40 MHz。
- 31kHz 固定频率输出的内部 RC 振荡器，该输出为对时序要求不高的应用提供了极低功耗选项。

内部振荡器电路提供了一个稳定的参考源，给 PIC18F87J10 系列器件增加了额外的功能以使器件高效工作：

- **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，单片机会切换到内部振荡器，使器件可继续低速工作或安全关闭。
- **双速启动：**该功能允许在上电复位或从休眠模式唤醒时将内部振荡器用作时钟源，直到主时钟源可用时止。

1.1.3 扩展的存储器

PIC18F87J10 系列为应用程序代码提供了从 32 KB 到 128 KB 足够的代码空间。程序存储器的闪存单元额定可反复擦写多达 100 次。如果不刷新，保守地估计数据能保存 100 年以上。

PIC18F87J10 系列还为动态应用数据提供了足够的空间和高达 3936 字节的数据 RAM。

1.1.4 外部存储总线

虽然 128 KB 的存储空间对一个应用就已足够，但 PIC18F87J10 系列的 80 引脚器件还是提供了外部存储总线。这样可使单片机的内部程序计数器能寻址高达 2 MB 的存储器空间，允许 8 位器件无法实现的数据访问级别。包括其他存储器选项：

- 使用片内和外部存储器组合，上限为 2MB
- 使用外部闪存存储器存储可再编程应用程序代码或大数据表
- 使用外部 RAM 器件存储大量可变数据

1.1.5 扩展指令集

PIC18F87J10 系列实现了可选的 PIC18 扩展指令集，新增了 8 条指令和变址寻址模式。扩展指令集作为器件配置选项被使能，它是专门为优化原先由高级语言（如：C 语言）开发的重入应用程序代码设计的。

1.1.6 便于移植

无论存储器大小如何，所有的器件均共享相同一组外设，随着应用代码的开发和变化，为各种应用提供便捷的移植路径。

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 引脚器件向 80 引脚的移植都是可以的。

PIC18F87J10 系列的引脚同其他 PIC18 系列器件（如 PIC18F8720 和 PIC18F8722）的引脚兼容。这为不同应用的发展开拓了新的视野，使开发者能在保留相同功能集的同时在MicrochipPIC18 系列中选择价廉的器件。

1.2 其他特殊功能

- **通信：**PIC18F87J10 系列包含了一系列串行通信外设，包括 2 个独立增强型 USART 和 2 个主控 SSP 模块，具备 SPI™ 和 I^2C™（主控和从动）两种工作模式。此外，一个通用 I/O 端口可以重新配置为 8 位并行从动端口，实现处理器到处理器的直接通信。
- **CCP 模块：**该系列中所有的器件均包含 2 个捕捉 / 比较 /PWM（CCP）模块和 3 个增强型 CCP 模块以在控制应用中得到最大的灵活性。可能会采用多达 4 个不同的时基以便能同时执行不同的操作。每个 ECCP（共 3 个）可提供多达 4 路 PWM 输出，共 12 路 PWM 输出。 ECCP 还提供了许多有用的功能，包括极性选择，可编程死区时间，自动关闭和自动重启、半桥和全桥输出模式。
- **10 位 A/D 转换器：**该模块包含了可编程采集时间，允许不必等待一个采样周期，就可选择通道并启动转换，从而减少了代码开销。
- **扩展的看门狗定时器（Watchdog Timer,WDT）：**该扩展的看门狗定时器添加了 16 位预分频器，可扩展的超时范围，在整个工作电压和温度内保持稳定。请参见**第 26.0 节 “电气规范”**了解看门狗的超时时间。

1.3 系列中各产品的具体信息

PIC18F87J10 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 4 个方面存在差异：

1. 闪存程序存储器（6 种规格，范围从 PIC18FX5J10 的 32 KB 到 PIC18FXJ710 的 128 KB）。
2. 数据 RAM（PIC18FX5J10/X5J15/X6J10 器件为 2048 字节，PIC18FX6J15/X7J10 器件为 3936 字节）。
3. A/D 通道（64 引脚器件有 11 个，80 引脚器件有 15 个）。
4. I/O 端口（64 引脚器件上有 7 个双向端口，80 引脚器件上有 9 个双向端口）。

该系列器件的其他功能都是相同的。在表 1-1 和表 1-2 中总结了这些功能。

在表 1-3 和表 1-4 种列举了所有器件的引脚排列。

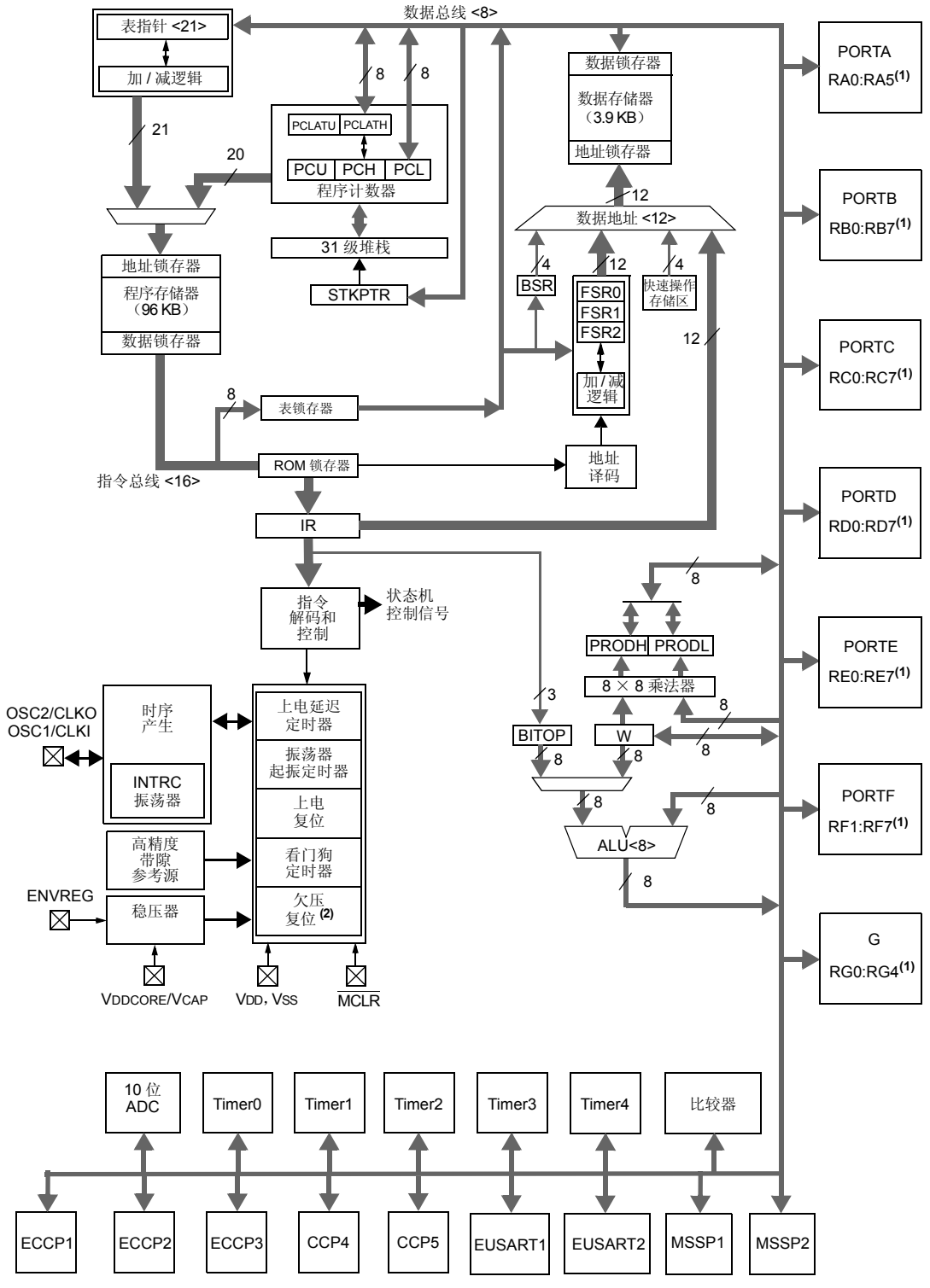
表 1-1： PIC18F87J10 系列器件特性 （64 引脚器件）

特性	PIC18F65J10	PIC18F65J15	PIC18F66J10	PIC18F66J15	PIC18F67J10
工作频率	DC–40 MHz	DC–40 MHz	DC–40 MHz	DC–40 MHz	DC–40 MHz
程序存储器 （字节）	32K	48K	64K	96K	128K
程序存储器 （指令）	16384	24576	32768	49152	65536
数据存储器 （字节）	2048	2048	2048	3936	3936
中断源	27				
I/O 端口	端口 A、 B、 C、 D、 E、 F 和 G				
定时器	5				
捕捉 / 比较 /PWM 模块	2				
增强型捕捉 / 比较 /PWM 模块	3				
串行通信	MSSP （2），增强型 USART （2）				
并行通信 （PSP）	有				
10 位模数转换模块	11 个输入通道				
复位 （和延时）	POR、 BOR、 RESET 指令，堆栈满、堆栈下溢、 MCLR 和 WDT （PWRT 和 OST）				
指令集	75 条指令，启用扩展指令集后总共为 83 条				
封装	64 引脚 TQFP				

表 1-2： PIC18F87J10 系列器件特性 （80 引脚器件）

特性	PIC18F85J10	PIC18F85J15	PIC18F86J10	PIC18F86J15	PIC18F87J10
工作频率	DC–40 MHz	DC–40 MHz	DC–40 MHz	DC–40 MHz	DC–40 MHz
程序存储器 （字节）	32K	48K	64K	96K	128K
程序存储器 （指令）	16384	24576	32768	49152	65536
数据存储器 （字节）	2048	2048	2048	3936	3936
中断源	27				
I/O 端口	端口 A、 B、 C、 D、 E、 F、 G、 H 和 J				
定时器	5				
捕捉 / 比较 /PWM 模块	2				
增强型捕捉 / 比较 /PWM 模块	3				
串行通信	MSSP （2），增强型 USART （2）				
并行通信 （PSP）	有				
10 位模数转换模块	15 个输入通道				
复位 （和延时）	POR、 BOR、 RESET 指令，堆栈满、堆栈下溢、 MCLR 和 WDT （PWRT 和 OST）				
指令集	75 条指令，启用扩展指令集后总共为 83 条				
封装	80 引脚 TQFP				

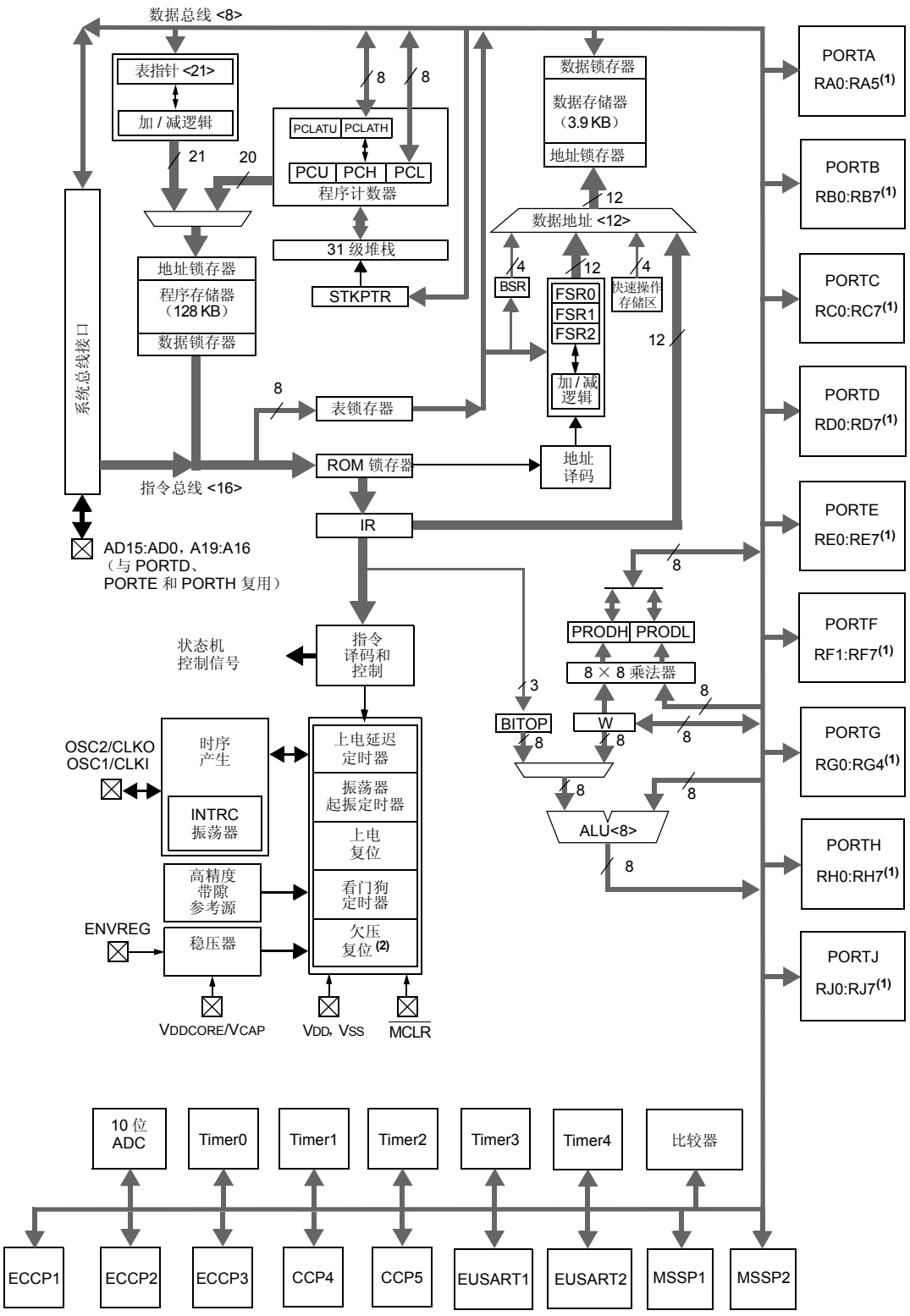
图 1-1：　　PIC18F6XJ10/6XJ15 （64 引脚）框图



注　**1：** 请参见表 1-3 了解 I/O 端口引脚说明。

2： 当主板稳压器使能时提供 BOR 功能。

图 1-2：　　PIC18F8XJ10/8XJ15 （80 引脚）框图



注　**1：** 请参见表 1-4 了解 I/O 端口引脚说明。

2： 当主板稳压器使能时提供 BOR 功能。

表 1-3： PIC18F6XJ10/6XJ15I/O 引脚排列说明

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
$\overline{MCLR}$	7	I	ST	主清零（复位）输入。此引脚为低电平时，器件复位。
OSC1/CLKI	39			振荡器晶振或外部时钟输入。
OSC1		I	ST	振荡器晶振输入或外部时钟源输入。
CLKI		I	CMOS	在 RC 模式下带 ST 缓冲器；否则带 CMOS 缓冲器。 外部时钟源输入。总是与 OSC1 引脚功能复用。（请参见相关的 SC1/CLKI 和 OSC2/CLKO 引脚信息。）
OSC2/CLKO	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，该引脚与晶振或谐振器相连。
CLKO		O	—	在 RC 模式下， OSC2 引脚输出 CLKO 振荡信号，该信号是 OSC1 引脚上振荡信号的 4 分频，该频率等于指令周期的倒数。
				PORTA 是双向 I/O 端口。
RA0/AN0	24			
RA0		I/O	TTL	数字 I/O。
AN0		I	模拟	模拟输入 0。
RA1/AN1	23			
RA1		I/O	TTL	数字 I/O。
AN1		I	模拟	模拟输入 1。
RA2/AN2/VREF-	22			
RA2		I/O	TTL	数字 I/O。
AN2		I	模拟	模拟输入 2。
VREF-		I	模拟	A/D 参考电压（低电平端）输入。
RA3/AN3/VREF+	21			
RA3		I/O	TTL	数字 I/O。
AN3		I	模拟	模拟输入 3。
VREF+		I	模拟	A/D 参考电压（高电平端）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	27			
RA5		I/O	TTL	数字 I/O。
AN4		I	模拟	模拟输入 4。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路（无 P 极二极管接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2/P2A 进行默认设置。
2： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置。

表 1-3： PIC18F6XJ10/6XJ15I/O 引脚排列说明 （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	48			
RB0		I/O	TTL	数字 I/O。
INT0		I	ST	外部中断 0。
FLT0		I	ST	ECCP1/2/3 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	46			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
RB3/INT3	45			
RB3		I/O	TTL	数字 I/O。
INT3		I	ST	外部中断 3。
RB4/KBI0	44			
RB4		I/O	TTL	数字 I/O。
KBI0		I	TTL	电平变化中断引脚。
RB5/KBI1	43			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
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　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2/P2A 进行默认设置。
2： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置。

表 1-3： PIC18F6XJ10/6XJ15I/O 引脚排列说明 （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/T1OSO/T13CKI	30			
RC0		I/O	ST	数字 I/O。
T1OSO		O	—	Timer1 振荡器输出。
T13CKI		I	ST	Timer1/Timer3 外部时钟源输入。
RC1/T1OSI/ECCP2/P2A	29			
RC1		I/O	ST	数字 I/O。
T1OSI		I	CMOS	Timer1 振荡器输入。
ECCP2(1)		I/O	ST	Capture2 输入 /Compare2 输出 /PWM2 输出。
P2A(1)		O	—	ECCP2 PWM 输出 A。
RC2/ECCP1/P1A	33			
RC2		I/O	ST	数字 I/O。
ECCP1		I/O	ST	Capture1 输入 /Compare1 输出 /PWM1 输出。
P1A		O	—	ECCP1 PWM 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		I/O	ST	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		I/O	ST	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	—	EUSART1 异步发送。
CK1		I/O	ST	EUSART1 同步时钟 （请参见相关的 RX1/DT1 引脚信息）。
RC7/RX1/DT1	32			
RC7		I/O	ST	数字 I/O。
RX1		I	ST	EUSART1 异步接收。
DT1		I/O	ST	EUSART1 同步数据 （请参见相关的 TX1/CK1 引脚信息）。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2/P2A 进行默认设置。
2： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置。

表 1-3： PIC18F6XJ10/6XJ15I/O 引脚排列说明 （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	58			
RD0		I/O	ST	数字 I/O。
PSP0		I/O	TTL	并行从动端口数据。
RD1/PSP1	55			
RD1		I/O	ST	数字 I/O。
PSP1		I/O	TTL	并行从动端口数据。
RD2/PSP2	54			
RD2		I/O	ST	数字 I/O。
PSP2		I/O	TTL	并行从动端口数据。
RD3/PSP3	53			
RD3		I/O	ST	数字 I/O。
PSP3		I/O	TTL	并行从动端口数据。
RD4/PSP4/SDO2	52			
RD4		I/O	ST	数字 I/O。
PSP4		I/O	TTL	并行从动端口数据。
SDO2		O	—	SPI™ 数据输出。
RD5/PSP5/SDI2/SDA2	51			
RD5		I/O	ST	数字 I/O。
PSP5		I/O	TTL	并行从动端口数据。
SDI2		I	ST	SPI 数据输入。
SDA2		I/O	ST	I^2C™ 数据 I/O。
RD6/PSP6/SCK2/SCL2	50			
RD6		I/O	ST	数字 I/O。
PSP6		I/O	TTL	并行从动端口数据。
SCK2		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL2		I/O	ST	I^2C 模式的同步串行时钟输入 / 输出。
RD7/PSP7/$\overline{SS2}$	49			
RD7		I/O	ST	数字 I/O。
PSP7		I/O	TTL	并行从动端口数据。
$\overline{SS2}$		I	TTL	SPI 从动选择输入。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2/P2A 进行默认设置。
2： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置。

表 1-3： PIC18F6XJ10/6XJ15I/O 引脚排列说明 （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/$\overline{RD}$/P2D	2			
RE0		I/O	ST	数字 I/O。
$\overline{RD}$		I	TTL	并行从动端口读控制。
P2D		O	—	ECCP2 PWM 输出 D。
RE1/$\overline{WR}$/P2C	1			
RE1		I/O	ST	数字 I/O。
$\overline{WR}$		I	TTL	并行从动端口写控制。
P2C		O	—	ECCP2 PWM 输出 C。
RE2/$\overline{CS}$/P2B	64			
RE2		I/O	ST	数字 I/O。
$\overline{CS}$		I	TTL	并行从动端口片选控制。
P2B		O	—	ECCP2 PWM 输出 B。
RE3/P3C	63			
RE3		I/O	ST	数字 I/O。
P3C		O	—	ECCP3 PWM 输出 C。
RE4/P3B	62			
RE4		I/O	ST	数字 I/O。
P3B		O	—	ECCP3 PWM 输出 B。
RE5/P1C	61			
RE5		I/O	ST	数字 I/O。
P1C		O	—	ECCP1 PWM 输出 C。
RE6/P1B	60			
RE6		I/O	ST	数字 I/O。
P1B		O	—	ECCP1 PWM 输出 B。
RE7/ECCP2/P2A	59			
RE7		I/O	ST	数字 I/O。
ECCP2(2)		I/O	ST	Capture2 输入 /Compare2 输出 /PWM2 输出。
P2A(2)		O	—	ECCP2 PWM 输出 A。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2/P2A 进行默认设置。
2： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置。

表 1-3： PIC18F6XJ10/6XJ15I/O 引脚排列说明 （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	17			
RF1		I/O	ST	数字 I/O。
AN6		I	模拟	模拟输入 6。
C2OUT		O	—	Comparator2 输出。
RF2/AN7/C1OUT	16			
RF2		I/O	ST	数字 I/O。
AN7		I	模拟	模拟输入 7。
C1OUT		O	—	Comparator1 输出。
RF3/AN8	15			
RF3		I/O	ST	数字 I/O。
AN8		I	模拟	模拟输入 8。
RF4/AN9	14			
RF4		I/O	ST	数字 I/O。
AN9		I	模拟	模拟输入 9。
RF5/AN10/CVREF	13			
RF5		I/O	ST	数字 I/O。
AN10		I	模拟	模拟输入 10。
CVREF		O	—	比较器参考电压输出。
RF6/AN11	12			
RF6		I/O	ST	数字 I/O。
AN11		I	模拟	模拟输入 11。
RF7/$\overline{SS1}$	11			
RF7		I/O	ST	数字 I/O。
$\overline{SS1}$		I	TTL	SPI™ 从动选择输入。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2/P2A 进行默认设置。
2： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置。

表 1-3： PIC18F6XJ10/6XJ15I/O 引脚排列说明 （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	3			
RG0		I/O	ST	数字 I/O。
ECCP3		I/O	ST	Capture3 输入 /Compare3 输出 /PWM3 输出。
P3A		O	—	ECCP3 PWM 输出 A。
RG1/TX2/CK2	4			
RG1		I/O	ST	数字 I/O。
TX2		O	—	EUSART2 异步发送。
CK2		I/O	ST	EUSART2 同步时钟 （请参见相关的 RX2/DT2 引脚信息）。
RG2/RX2/DT2	5			
RG2		I/O	ST	数字 I/O。
RX2		I	ST	EUSART2 异步接收。
DT2		I/O	ST	EUSART2 同步数据 （请参见相关的 TX2/CK2 引脚信息）。
RG3/CCP4/P3D	6			
RG3		I/O	ST	数字 I/O。
CCP4		I/O	ST	Capture4 输入 /Compare4 输出 /PWM4 输出。
P3D		O	—	ECCP3 PWM 输出 D。
RG4/CCP5/P1D	8			
RG4		I/O	ST	数字 I/O。
CCP5		I/O	ST	Capture5 输入 /Compare5 输出 /PWM5 输出。
P1D		O	—	ECCP1 PWM 输出 D。
VSS	9, 25, 41, 56	P	—	逻辑电路和 I/O 引脚的参考地。
VDD	26, 38, 57	P	—	外设逻辑电路和 I/O 引脚的正向电源。
AVSS	20	P	—	模拟模块的参考地。
AVDD	19	P	—	模拟模块的正向电源。
ENVREG	18	I	ST	使能片内稳压器。
VDDCORE/VCAP	10			与内核逻辑电路电源或外部过滤电容连接。
VDDCORE		P	—	单片机内核逻辑电路正向电源 （禁止稳压器）。
VCAP		P	—	与外部过滤电容连接 （使能稳压器）。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 VDD）

注 1： 当 CCP2MX 配置位置 1 时，对 ECCP2/P2A 进行默认设置。
2： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明

引脚名称	引脚号 TQFP	引脚类型	缓冲器类型	说明
$\overline{MCLR}$	9	I	ST	主清零（复位）输入。此引脚为低电平时器件复位。
OSC1/CLKI	49			振荡器晶振或外部时钟输入。
OSC1		I	ST	振荡器晶振输入或外部时钟源输入。在 RC 模式配置下带 ST 缓冲器；否则带 CMOS 缓冲器。
CLKI		I	CMOS	外部时钟源输入。总是与 OSC1 引脚功能复用。（请参加相关的 OSC1/CLKI 和 OSC2/CLKO 引脚信息。）
OSC2/CLKO	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，该引脚与晶振或谐振器相连。
CLKO		O	—	在 RC 模式下， OSC2 引脚输出 CLKO 振荡信号，该信号是 OSC1 引脚上振荡信号的 4 分频，该频率等于指令周期的倒数。
				PORTA 是双向 I/O 端口。
RA0/AN0	30			
RA0		I/O	TTL	数字 I/O。
AN0		I	模拟	模拟输入 0。
RA1/AN1	29			
RA1		I/O	TTL	数字 I/O。
AN1		I	模拟	模拟输入 1。
RA2/AN2/V_{REF-}	28			
RA2		I/O	TTL	数字 I/O。
AN2		I	模拟	模拟输入 2。
V_{REF-}		I	模拟	A/D 参考电压（低电平端）输入。
RA3/AN3/V_{REF+}	27			
RA3		I/O	TTL	数字 I/O。
AN3		I	模拟	模拟输入 3。
V_{REF+}		I	模拟	A/D 参考电压（高电平端）输入。
RA4/T0CKI	34			
RA4		I/O	ST	数字 I/O。
T0CKI		I	ST	Timer0 外部时钟源输入。
RA5/AN4	33			
RA5		I/O	TTL	数字 I/O。
AN4		I	模拟	模拟输入 4。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路（无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置（扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置（单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明 （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 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	56			
RB2		I/O	TTL	数字 I/O。
INT2		I	ST	外部中断 2。
RB3/INT3/ECCP2/P2A	55			
RB3		I/O	TTL	数字 I/O。
INT3		I	ST	外部中断 3。
ECCP2(1)		I/O	ST	Capture2 输入 /Compare2 输出 /PWM2 输出。
P2A(1)		O	—	ECCP2 PWM 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	53			
RB5		I/O	TTL	数字 I/O。
KBI1		I	TTL	电平变化中断引脚。
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　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 VDD）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明 （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/T1OSO/T13CKI	36			
RC0		I/O	ST	数字 I/O。
T1OSO		O	—	Timer1 振荡器输出。
T13CKI		I	ST	Timer1/Timer3 外部时钟源输入。
RC1/T1OSI/ECCP2/P2A	35			
RC1		I/O	ST	数字 I/O。
T1OSI		I	CMOS	Timer1 振荡器输入。
ECCP2(2)		I/O	ST	Capture2 输入 /Compare2 输出 /PWM2 输出。
P2A(2)		O	—	ECCP2 PWM 输出 A。
RC2/ECCP1/P1A	43			
RC2		I/O	ST	数字 I/O。
ECCP1		I/O	ST	Capture1 输入 /Compare1 输出 /PWM1 输出。
P1A		O	—	ECCP1 PWM 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	ST	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	ST	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	—	EUSART1 异步发送。
CK1		I/O	ST	EUSART1 同步时钟 （请参见相关的 RX1/DT1 引脚信息）。
RC7/RX1/DT1	38			
RC7		I/O	ST	数字 I/O。
RX1		I	ST	EUSART1 异步接收。
DT1		I/O	ST	EUSART1 同步数据 （请参见相关的 TX1/CK1 引脚信息）。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明 （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/AD0/PSP0	72			
RD0		I/O	ST	数字 I/O。
AD0		I/O	TTL	外部存储器地址 / 数据 0。
PSP0		I/O	TTL	并行从动端口数据。
RD1/AD1/PSP1	69			
RD1		I/O	ST	数字 I/O。
AD1		I/O	TTL	外部存储器地址 / 数据 1。
PSP1		I/O	TTL	并行从动端口数据。
RD2/AD2/PSP2	68			
RD2		I/O	ST	数字 I/O。
AD2		I/O	TTL	外部存储器地址 / 数据 2。
PSP2		I/O	TTL	并行从动端口数据。
RD3/AD3/PSP3	67			
RD3		I/O	ST	数字 I/O。
AD3		I/O	TTL	外部存储器地址 / 数据 3。
PSP3		I/O	TTL	并行从动端口数据。
RD4/AD4/PSP4/SDO2	66			
RD4		I/O	ST	数字 I/O。
AD4		I/O	TTL	外部存储器地址 / 数据 4。
PSP4		I/O	TTL	并行从动端口数据。
SDO2		O	—	SPI™ 数据输出。
RD5/AD5/PSP5/ SDI2/SDA2	65			
RD5		I/O	ST	数字 I/O。
AD5		I/O	TTL	外部存储器地址 / 数据 5。
PSP5		I/O	TTL	并行从动端口数据。
SDI2		I	ST	SPI 数据输入。
SDA2		I/O	ST	I^2C™ 数据 I/O。
RD6/AD6/PSP6/ SCK2/SCL2	64			
RD6		I/O	ST	数字 I/O。
AD6		I/O	TTL	外部存储器地址 / 数据 6。
PSP6		I/O	TTL	并行从动端口数据。
SCK2		I/O	ST	SPI 模式的同步串行时钟输入 / 输出。
SCL2		I/O	ST	I^2C 模式的同步串行时钟输入 / 输出。
RD7/AD7/PSP7/$\overline{SS2}$	63			
RD7		I/O	ST	数字 I/O。
AD7		I/O	TTL	外部存储器地址 / 数据 7。
PSP7		I/O	TTL	并行从动端口数据。
$\overline{SS2}$		I	TTL	SPI 从动选择输入。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明 （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/AD8/$\overline{RD}$/P2D	4			
RE0		I/O	ST	数字 I/O。
AD8		I/O	TTL	外部存储器地址 / 数据 8。
$\overline{RD}$		I	TTL	并行从动端口读控制。
P2D		O	—	ECCP2 PWM 输出 D。
RE1/AD9/$\overline{WR}$/P2C	3			
RE1		I/O	ST	数字 I/O。
AD9		I/O	TTL	外部存储器地址 / 数据 9。
$\overline{WR}$		I	TTL	并行从动端口写控制。
P2C		O	—	ECCP2 PWM 输出 C。
RE2/AD10/$\overline{CS}$/P2B	78			
RE2		I/O	ST	数字 I/O。
AD10		I/O	TTL	外部存储器地址 / 数据 10。
$\overline{CS}$		I	TTL	并行从动端口片选控制。
P2B		O	—	ECCP2 PWM 输出 B。
RE3/AD11/P3C	77			
RE3		I/O	ST	数字 I/O。
AD11		I/O	TTL	外部存储器地址 / 数据 11。
P3C(3)		O	—	ECCP3 PWM 输出 C。
RE4/AD12/P3B	76			
RE4		I/O	ST	数字 I/O。
AD12		I/O	TTL	外部存储器地址 / 数据 12。
P3B(3)		O	—	ECCP3 PWM 输出 B。
RE5/AD13/P1C	75			
RE5		I/O	ST	数字 I/O。
AD13		I/O	TTL	外部存储器地址 / 数据 13。
P1C(3)		O	—	ECCP1 PWM 输出 C。
RE6/AD14/P1B	74			
RE6		I/O	ST	数字 I/O。
AD14		I/O	TTL	外部存储器地址 / 数据 14。
P1B(3)		O	—	ECCP1 PWM 输出 B。
RE7/AD15/ECCP2/P2A	73			
RE7		I/O	ST	数字 I/O。
AD15		I/O	TTL	外部存储器地址 / 数据 15。
ECCP2(4)		I/O	ST	Capture2 输入 /Compare2 输出 /PWM2 输出。
P2A(4)		O	—	ECCP2 PWM 输出 A。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明 （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	23			
RF1		I/O	ST	数字 I/O。
AN6		I	模拟	模拟输入 6。
C2OUT		O	—	Comparator2 输出。
RF2/AN7/C1OUT	18			
RF2		I/O	ST	数字 I/O。
AN7		I	模拟	模拟输入 7。
C1OUT		O	—	Comparator1 输出。
RF3/AN8	17			
RF3		I/O	ST	数字 I/O。
AN8		I	模拟	模拟输入 8。
RF4/AN9	16			
RF4		I/O	ST	数字 I/O。
AN9		I	模拟	模拟输入 9。
RF5/AN10/CVREF	15			
RF5		I/O	ST	数字 I/O。
AN10		I	模拟	模拟输入 10。
CVREF		O	—	比较器参考电压输出。
RF6/AN11	14			
RF6		I/O	ST	数字 I/O。
AN11		I	模拟	模拟输入 11。
RF7/$\overline{SS1}$	13			
RF7		I/O	ST	数字 I/O。
$\overline{SS1}$		I	TTL	SPI 从动选择输入。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 VDD）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明 （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	Capture3 输入 /Compare3 输出 /PWM3 输出。
P3A		O	—	ECCP3 PWM 输出 A。
RG1/TX2/CK2	6			
RG1		I/O	ST	数字 I/O。
TX2		O	—	EUSART2 异步发送。
CK2		I/O	ST	EUSART2 同步时钟 （请参见相关的 RX2/DT2 引脚信息）。
RG2/RX2/DT2	7			
RG2		I/O	ST	数字 I/O。
RX2		I	ST	EUSART2 异步接收。
DT2		I/O	ST	EUSART2 同步数据 （请参见相关的 TX2/CK2 引脚信息）。
RG3/CCP4/P3D	8			
RG3		I/O	ST	数字 I/O。
CCP4		I/O	ST	Capture4 输入 /Compare4 输出 /PWM4 输出。
P3D		O	—	ECCP3 PWM 输出 D。
RG4/CCP5/P1D	10			
RG4		I/O	ST	数字 I/O。
CCP5		I/O	ST	Capture5 输入 /Compare5 输出 /PWM5 输出。
P1D		O	—	ECCP1 PWM 输出 D。

图注： TTL = TTL 兼容输入　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　Analog = 模拟输入
I = 输入　O = 输出
P = 电源　OD = 漏极开路 （无 P 极二极管接到 V_{DD}）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明 （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/A16	79			
RH0		I/O	ST	数字 I/O。
A16		I/O	TTL	外部存储器地址 / 数据 16。
RH1/A17	80			
RH1		I/O	ST	数字 I/O。
A17		I/O	TTL	外部存储器地址 / 数据 17。
RH2/A18	1			
RH2		I/O	ST	数字 I/O。
A18		I/O	TTL	外部存储器地址 / 数据 18。
RH3/A19	2			
RH3		I/O	ST	数字 I/O。
A19		I/O	TTL	外部存储器地址 / 数据 19。
RH4/AN12/P3C	22			
RH4		I/O	ST	数字 I/O。
AN12		I	模拟	模拟输入 12。
P3C(5)		O	—	ECCP3 PWM 输出 C。
RH5/AN13/P3B	21			
RH5		I/O	ST	数字 I/O。
AN13		I	模拟	模拟输入 13。
P3B(5)		O	—	ECCP3 PWM 输出 B。
RH6/AN14/P1C	20			
RH6		I/O	ST	数字 I/O。
AN14		I	模拟	模拟输入 14。
P1C(5)		O	—	ECCP1 PWM 输出 C。
RH7/AN15/P1B	19			
RH7		I/O	ST	数字 I/O。
AN15		I	模拟	模拟输入 15。
P1B(5)		O	—	ECCP1 PWM 输出 B。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 VDD）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

表 1-4： PIC18F8XJ10/8XJ15I/O 引脚排列说明 （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。
RJ1/$\overline{OE}$	61			
RJ1		I/O	ST	数字 I/O。
$\overline{OE}$		O	—	外部存储器输出使能。
RJ2/$\overline{WRL}$	60			
RJ2		I/O	ST	数字 I/O。
$\overline{WRL}$		O	—	外部存储器低电平写控制引脚。
RJ3/$\overline{WRH}$	59			
RJ3		I/O	ST	数字 I/O。
$\overline{WRH}$		O	—	外部存储器高电平写控制引脚。
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	—	外设逻辑电路和 I/O 引脚的正向电源。
AVSS	26	P	—	模拟模块的参考地。
AVDD	25	P	—	模拟模块的正向电源。
ENVREG	24	I	ST	使能片内稳压器。
VDDCORE/VCAP	12			与内核逻辑电源或外部过滤电容连接。
VDDCORE		P	—	单片机内核逻辑电路正向电源 （禁止稳压器）。
VCAP		P	—	与外部过滤电容连接 （使能稳压器）。

图注： TTL = TTL 兼容输入　　CMOS = CMOS 兼容输入或输出
ST = CMOS 电平的施密特触发器输入　　Analog = 模拟输入
I = 输入　　O = 输出
P = 电源　　OD = 漏极开路 （无 P 极二极管接到 VDD）

注 1： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （扩展单片机模式）。
2： 当 CCP2MX 配置位置 1 时，任何器件在任意工作模式下，对 ECCP2/P2A 进行默认设置。
3： 当 ECCPMX 配置位置 1 时，对 P1B/P1C/P3B/P3C 进行默认设置。
4： 当 CCP2MX 配置位清零时，对 ECCP2/P2A 进行其他设置 （单片机模式）。
5： 当 ECCPMX 配置位清零时，对 P1B/P1C/P3B/P3C 进行其他设置。

注：

2.0 振荡器配置

2.1 振荡器类型

PIC18F87J10 系列器件可以在五种不同的振荡器模式下工作：

1. HS 高速晶振 / 谐振器
2. HSPLL 带软件 PLL 控制的高速晶振 / 谐振器
3. EC 带 FOSC/4 输出的外部时钟
4. ECPLL 带软件 PLL 控制的外部时钟
5. INTRC 31 kHz 内部振荡器

用户可以通过编程FOSC2:FOSC0配置位来选择这其中的前四种模式。而第五种模式（INTRC）可在软件控制下调用；也可以将它配置为器件复位时的默认模式。

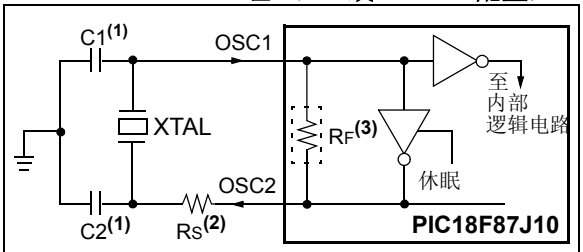
2.2 晶体振荡器 / 陶瓷谐振器 （HS 模式）

在 HS 或 HSPLL 振荡器模式中，晶体振荡器或陶瓷谐振器被连接到 OSC1 和 OSC2 引脚以产生振荡。图 2-1 给出了引脚连接图。

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，会使振荡器产生的频率不在晶体制造厂商所给的参数范围内。

图 2-1： 晶振 / 陶瓷谐振器的工作原理 （HS 或 HSPLL 配置）



注 1： 如需了解 C1 和 C2 的初始值，请参见表 2-1 和表 2-2。

2： AT 条形切割的晶体可能会需要一个串联电阻（RS）。

3： RF 的值根据所选择的振荡器模式而定。

表 2-1： 陶瓷谐振器的电容选择

所使用的典型电容值：			
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。 已测试这些电容搭配下列谐振器时的基本起振和工作情况。**这些值不是最佳值**。 要得到合适的振荡器工作状况，可能需要不同的电容值。用户应在应用的预期 VDD 和温度范围内测试振荡器的性能。 欲知更多信息，请参见表 2-2 后的 “注”。			
所使用的谐振器：			
4.0 MHz			
8.0 MHz			
16.0 MHz			

表 2-2：　　晶体振荡器的电容选择

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。 已测试这些电容搭配下列晶振时的基本起振和工作情况。**这些值不是最佳值。** 要得到合适的振荡器工作状况，可能需要不同的电容值。用户应在应用的预期 VDD 和温度范围内测试振荡器的性能。 欲知更多信息，请参见本表后的“注”。			
所使用的晶振：			
4 MHz			
8 MHz			
20 MHz			

注　1： 较高的电容值可以增加振荡器的稳定性，但同时也会增加起振时间。

2： 因为每种谐振器 / 晶振都有其自身特点，用户应当向谐振器 / 晶振制造厂商询问外部元件的相应值。

3： 为避免对低驱动电平参数的晶振造成过驱，可能会需要使用电阻 Rs。

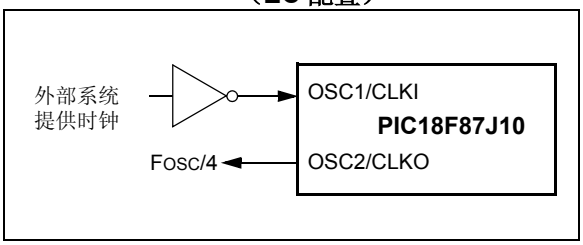
4： 请在应用中的预期 VDD 和温度范围内验证振荡器的性能。

2.3　外部时钟输入（EC 模式）

EC 和 ECPLL 振荡器模式需要在 OSC1 引脚连接一个外部时钟源。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。

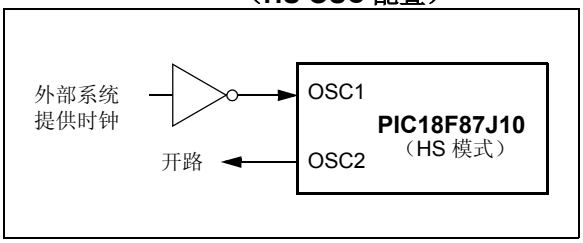
在 EC 振荡器模式下，振荡器频率的 4 分频信号可由 OSC2 引脚输出。此信号可用于测试或同步其他逻辑电路。图 2-2 给出了 EC 振荡器模式的引脚连接图。

图 2-2：　　外部时钟输入工作原理（EC 配置）



如图 2-3 所示，在 HS 模式下，OSC1 引脚也可以连接外部时钟源。在此配置下，不可用 OSC2 引脚上的 4 分频输出。

图 2-3：　　外部时钟输入工作原理（HS OSC 配置）

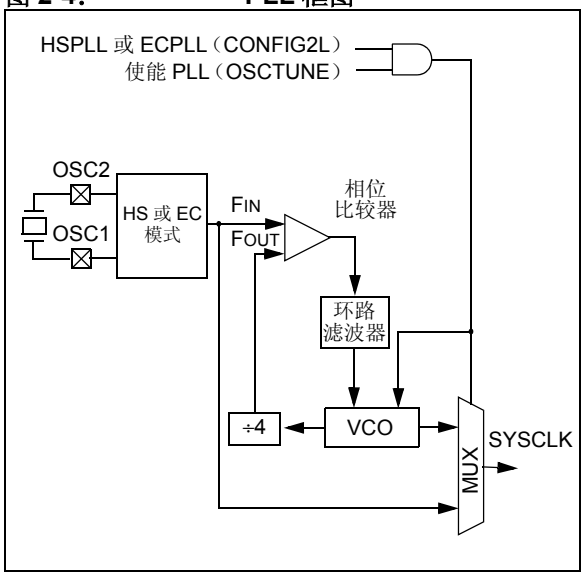


2.4 PLL 倍频器

如果用户希望使用低频晶振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（Phase Locked Loop，PLL）电路。此电路有助于那些担心高频晶振引起 EMI 或需要内部振荡器提供高速时钟的用户。由于这些原因，可使用 HSPLL 和 ECPLL 模式。

HSPLL 和 ECPLL 模式使器件能够以外部振荡源的 4 倍频运行以产生最高为 40 MHz 的频率。 通过在 OSCTUNE 寄存器（寄存器 2-1）中将 PLLEN 位置 1 来使能 PLL。

图 2-4： PLL 框图



寄存器 2-1： OSCTUNE：PLL 控制寄存器

U-0	R/W-0(1)	U-0	U-0	U-0	U-0	U-0	U-0
—	PLLEN(1)	—	—	—	—	—	—
bit 7							bit 0

bit 7 **未用：**读为 0

bit 6 **PLLEN：**倍频器 PLL 使能位 (1)
1 = PLL 已使能
0 = PLL 已禁止

注 1：仅对于 ECPLL 和 HSPLL 振荡器配置可用，否则，此位不可用并读为 0。

bit 5-0 **未用：**读为 0

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

2.5 内部振荡器模块

PIC18F87J10 系列器件包括内部振荡源（Internal Oscillator Source，INTRC），它提供标称的 31 kHz 输出频率。INTRC 在器件上电延迟定时器使能并在器件的配置周期到器件进入工作模式这段时间里作为器件时钟源。如果选择 INTRC 作为器件时钟源或者使能了以下各项之一也会使能 INTRC：

- 故障保护时钟监视器
- 看门狗定时器
- 双速启动

在**第 23.0 节 “CPU 的特殊功能”**中对这些功能进行了更详细的讨论。

也可以通过将 FOSC2 配置位置 1，将 INTRC 配置为器件起振时的默认时钟源。这将在**第 2.6.1 节 “振荡器控制寄存器”**中进行讨论。

2.6 时钟源和振荡器切换

PIC18F87J10 系列提供的功能包括允许将器件时钟源从主振荡器切换到备用时钟源。PIC18F87J10 系列器件提供两种备用时钟源。当备用时钟源使能时，各种功耗管理工作模式都可用。

基本上，这些器件有三种时钟源：

- 主振荡器
- 辅助振荡器
- 内部振荡器

主振荡器包括外部晶振和谐振器模式以及外部时钟模式。特定的模式由 FOSC2:FOSC0 配置位定义。这些模式的具体情况已在本章前面的内容中作过介绍。

辅助振荡器是没有与 OSC1 或 OSC2 引脚连接的外部时钟源。这些时钟源即使在控制器处于功耗管理模式时仍然可以继续工作。

PIC18F87J10 系列器件支持使用 Timer1 振荡器作为辅助振荡器。此振荡器在所有功耗管理模式中通常用作实时时钟等功能的时基。

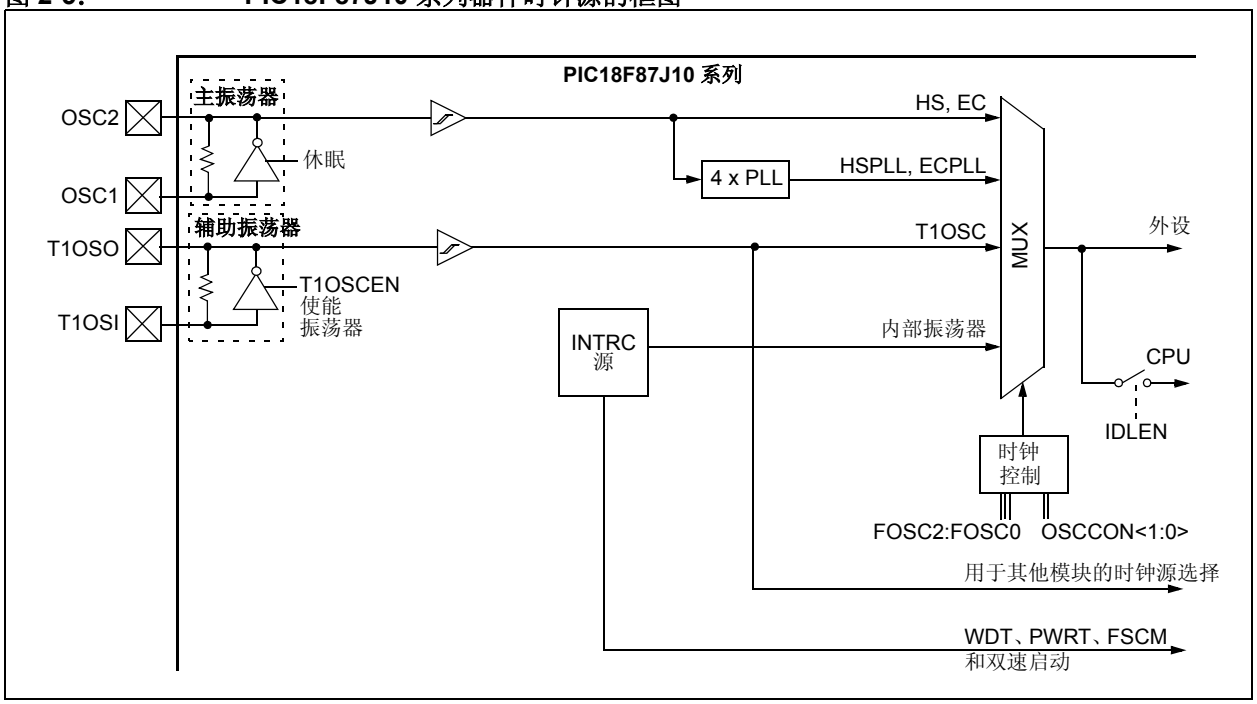
大部分情况下，在 RC0/T1OSO/T13CKI 和 RC1/T1OSI 引脚之间连接一个 32.768 kHz 的时钟晶振。在每个引脚与地之间均接有负载电容。

在**第 12.3 节 “Timer1 振荡器”**中将对 Timer1 振荡器作更详细的讨论。

除了作为主时钟源之外，**内部振荡器**还可以作为功耗管理模式的时钟源。 INTRC 也作为几种特殊功能的时钟源，例如 WDT 和故障保护时钟监视器。

图 2-5 所示为 PIC18F87J10 系列器件的时钟源。欲知配置寄存器的详细信息，请参见**第 23.0 节 “CPU 的特殊功能”**。

图 2-5： PIC18F87J10 系列器件时钟源的框图



2.6.1 振荡器控制寄存器

OSCCON 寄存器（寄存器 2-2）控制全功耗模式和功耗管理模式下系统时钟工作的几个方面。

系统时钟选择位 SCS1:SCS0 选择时钟源。可用的时钟源有主时钟（由 FOSC2:FOSC0 配置位定义）、辅助时钟（Timer1 振荡器）和内部振荡器。在写入一个或多个位后，有一段短的时钟切换间隔，之后，时钟源发生改变。

OSTS（OSCCON<3>）和 T1RUN（T1CON<6>）位表明当前是哪个时钟源提供器件时钟。OSTS 位表明振荡器起振定时器（OST）已超时，主时钟是主时钟模式下的器件时钟源。T1RUN 位表明 Timer1 振荡器是辅助时钟模式下的器件时钟源。在功耗管理模式中，这些位中总是只有一个位被置 1。如果这些位都没有置 1，则当前系统时钟源是 INTRC，或内部振荡器刚刚起振且尚未稳定。

IDLEN 位确定当执行 SLEEP 指令时，器件是进入休眠模式还是某种空闲模式。

第 3.0 节“功耗管理模式”中将更详细的讨论在 OSCCON 寄存器中对于标志位和控制位的使用。

注 1： 要选择辅助时钟源，必须使能 Timer1 振荡器。通过将 Timer1 控制寄存器中的 T1OSCEN 位（T1CON<3>）置 1，可以使能 Timer1 振荡器。如果未使能 Timer1 振荡器，则在执行 SLEEP 指令期间任何选择辅助时钟源的操作都会被忽略。

2： 建议在执行 SLEEP 指令之前 Timer1 振荡器已经在工作并且保持稳定，否则在 Timer1 振荡器起振时可能会有一段较长时间的延迟。

2.6.1.1 系统时钟选择和 FOSC2 配置位

在所有形式的复位中，SCS 位都会被清零。在器件的默认配置中，这意味着由 FOSC1:FOSC0（也就是 HC 或 EC 模式的一种）定义的主振荡器用作器件复位时的主时钟源。

复位时的默认时钟配置可以随着 FOSC2 配置位的改变而改变。此位的作用是设置当 SCS1:SCS0 = 00 时选择的时钟源。当 FOSC2 = 1（默认）时，不管何时 SCS1:SCS0 = 00，都选择由 FOSC1:FOSC0 定义的时钟源。当 FOSC2 = 0 时，不管何时 SCS1:SCS2 = 00，都选择 INTRC 振荡器。因为在复位时 SCS 位被清零，FOSC2 的设置也会更改复位时的默认振荡器模式。

不管 FOSC2 的设置如何，INTRC 总是会在器件上电时被使能。它将作为时钟源直到器件从存储器中载入了它的配置值。此时 FOSC 配置位被读取并选择了振荡器的工作模式。

注意主时钟或内部振荡器在任何给定条件下都会有两种位设置选项，取决于 FOSC2 的设置。

2.6.2 振荡器转换

PIC18F87J10 系列器件包含在时钟源之间切换时防止时钟“毛刺”的电路。在时钟切换时，器件时钟会有短暂的停顿。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。

第 3.1.2 节 “进入功耗管理模式”中将对时钟切换进行更详细的讨论。

寄存器 2-2：　OSCCON：振荡器控制寄存器

R/W-0	U-0	U-0	U-0	R-q(1)	U-0	R/W-0	R/W-0
IDLEN	—	—	—	OSTS	—	SCS1	SCS0
bit 7							bit 0

bit 7　**IDLEN**：空闲使能位
1 = 在执行 SLEEP 指令时，器件进入空闲模式
0 = 在执行 SLEEP 指令时，器件进入休眠模式

bit 6-4　**未用**：读为 0

bit 3　**OSTS**：振荡器起振延时状态位 (1)
1 = 振荡器起振定时器延迟时间已经结束；主振荡器正在运行
0 = 振荡器起振定时器正在运行，主振荡器尚未准备就绪

注　1：当 HS 模式和双速启动模式都使能时，复位值为 0，否则，复位值为 1。

bit 2　**未用**：读为 0

bit 1-0　**SCS1:SCS0**：系统时钟选择位
11 = 内部振荡器
10 = 主振荡器
01 = Timer1 振荡器
当 FOSC2 =1 时：
00 = 主振荡器
当 FOSC2 =0 时：
00 = 内部振荡器

图注：			
U = 未用位，读为 0		q = 由配置确定	
-n = 上电复位时的值	R = 可读位	0 = 清零	W = 可写位

2.7　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 OSC2 的引脚）将会停止振荡。

在辅助时钟模式下（SEC_RUN和SEC_IDLE），Timer1 振荡器作为器件时钟源工作。如果需要，Timer1 振荡器也可以运行在所有功耗管理模式下为 Timer1 或 Timer3 提供时钟。

在 RC_RUN 和 RC_IDLE 模式中，内部振荡器提供器件时钟源。 可以直接使用 31 kHz 的 INTRC 输出提供时钟源并且使能它来支持多种特殊功能，与是否是功耗管理模式无关 （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 23.2 节 “看门狗定时器（WDT）”**至**第 23.5 节 “故障保护时钟监视器”**）。

如果选择了休眠模式，所有的时钟源都会被停止。因为休眠模式消除了所有的晶体管切换电流，休眠模式能实现器件最小的电流消耗 （仅泄漏电流）。

在休眠期间使能任何片上功能将会增加休眠时的电流消耗。需要使能 INTRC 来支持 WDT 工作。 Timer1 振荡器可以用于为实时时钟提供时钟源。不需要系统时钟源的其他功能也可以工作 （即 MSSP 从器件、 PSP 和 INTn 引脚等等）。**第 26.2 节 “直流规范：掉电和供电电流”**列出了可能会显著增加电流消耗的外设。

2.8 上电延迟

由两个定时器控制上电延迟，这样大部分应用都无需外接复位电路。上电延迟可以确保在满足以下条件前器件保持复位状态：工作环境下器件电源稳定，并且主时钟已工作并稳定。欲知有关上电延迟的其他信息，请参见**第 4.5 节 “上电延时定时器 （PWRT）”**。

第一个定时器是上电延时定时器 （PWRT），它在上电时提供一个固定的延迟（表 25-12 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（Oscillator Start-up Timer，OST），用于在晶体振荡器稳定前使芯片保持在复位状态 （HS 模式）。 OST 通过计数 1024 个振荡周期实现此延迟并在延迟后允许振荡器为器件提供时钟。

在上电复位之后，有一个延迟间隔 TCSD （表 25-12 的参数 38），控制器在这段时间中为执行指令做准备。

表 2-3： 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，ECPLL	悬空，由外部时钟拉高	处于逻辑低电平 （时钟 /4 输出）
HS，HSPLL	处于静态电平时，反馈反相器被禁用	处于静态电平时，反馈反相器被禁用

注： 欲知有关由于休眠和 $\overline{\text{MCLR}}$ 复位而产生的延时的信息，请参见**第 4.0 节 “复位”**中的表 4-2。

注：

3.0 功耗管理模式

PIC18F87J10 系列器件提供了只需通过管理 CPU 和外设的时钟源就可以管理功耗的功能。一般而言，较低的时钟频率和由时钟源驱动的电路数目的减少会使功耗降低。为了在应用中管理功耗，提供了三种主要的工作模式：

- 运行模式
- 空闲模式
- 休眠模式

这些模式定义了器件的哪些部分由时钟源驱动，以及以多高的时钟速度驱动。运行和空闲模式可以使用三种可用的时钟源中（主、辅助或内部振荡器电路）的任何一种；而休眠模式不使用时钟源。

功耗管理模式包括几种在以前的 PICmicro® 器件上提供的节省功耗的功能。其中一个功能是在其他 PIC18 器件上提供的时钟切换功能，允许控制器使用 Timer1 振荡器代替主振荡器。还包括的一个功能是所有 PICmicro 器件都提供的休眠模式，在此模式下器件时钟停止。

3.1 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 以及使用哪个时钟源。IDLEN 位（OSCCON<7>）控制 CPU 的时钟驱动，而 SCS1:SCS0 位（OSCCON<1:0>）选择时钟源。表 3-1 总结了各个模式、位设置、时钟源和受影响的模块。

3.1.1 时钟源

SCS1:SCS0 位可以在功耗管理模式下选择三个时钟源中的一个。它们是：

- 主时钟，由 FOSC2:FOSC0 配置位定义。
- 辅助时钟 （Timer1 振荡器）
- 内部振荡器

3.1.2 进入功耗管理模式

从一种功耗管理模式转换到另一种功耗管理模式是通过装载 OSCCON 寄存器开始的。SCS1:SCS0 位选择时钟源并确定使用哪一种运行或空闲模式。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）。切换可能也会遇到时钟转换延迟。在**第 3.1.3 节“时钟切换和状态指示位”**及后续部分中对这些问题有所讨论。

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

更改功耗管理模式并不总是要求设置所有这些位，这由当前工作模式和将要切换到的模式决定。可以通过在发出 SLEEP 指令之前，更改振荡器选择位或更改 IDLEN 位来进行模式转换。如果已经正确地配置了 IDLEN 位，可能只需通过执行 SLEEP 指令就能转换到所需的模式。

表 3-1： 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)	SCS1:SCS0<1:0>	CPU	外设	
休眠	0	N/A	关闭	关闭	无——禁止所有时钟
PRI_RUN	N/A	10	被时钟源驱动	被时钟源驱动	主时钟源——HS、 EC、 HSPLL 和 ECPLL；这是正常的全功耗工作模式。
SEC_RUN	N/A	01	被时钟源驱动	被时钟源驱动	辅助时钟源——Timer1 振荡器
RC_RUN	N/A	11	被时钟源驱动	被时钟源驱动	内部振荡器
PRI_IDLE	1	10	关闭	被时钟源驱动	主时钟源—— HS、 EC、 HSPLL 和 ECPLL
SEC_IDLE	1	01	关闭	被时钟源驱动	辅助时钟源——Timer1 振荡器
RC_IDLE	1	11	关闭	被时钟源驱动	内部振荡器

注 1： 当执行 SLEEP 指令时， IDLEN 反映出它的值。

3.1.3 时钟切换和状态指示位

时钟源之间切换的时间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。

OSTS （OSCCON<3>）和 T1RUN （T1CON<6>）两个位表示当前的时钟源及其状态。通常，在给定功耗管理模式下，这两个位中只有一个将被置 1。当 OSTS 位被置 1 时，主时钟提供器件时钟。当 T1RUN 位被置 1 时，Timer1 振荡器提供器件时钟。如果这两个位都不置 1， INTRC 为器件提供时钟。

注：	执行 SLEEP 指令不一定会使器件进入休眠模式。 它充当触发指令，根据 IDLEN 位的设置，使控制器进入休眠模式和某种空闲模式。

3.1.4 多条 SLEEP 命令

用 SLEEP 指令启动的功耗管理模式由执行这条指令时 IDLEN 位的设置决定。如果执行另一条 SLEEP 指令，器件将进入由那时 IDLEN 位指定的功耗管理模式。如果 IDLEN 位已更改，则器件将进入由新的设置指定的新的功耗管理模式。

3.2 运行模式

在运行模式下，为内核和外设提供时钟的时钟源都在运行。这些模式之间的差异在于时钟源。

3.2.1 PRI_RUN 模式

PRI_RUN 模式是单片机的正常的全功耗工作模式。除非使能双速启动，这也是器件复位后的默认模式（详细信息请参见**第 23.4 节 “双速启动”**）。在此模式下，OSTS 位置 1（见**第 2.6.1 节 “振荡器控制寄存器”**）。

3.2.2 SEC_RUN 模式

SEC_RUN 模式是与其他 PIC18 器件提供的时钟切换功能兼容的。在此模式下， CPU 和外设由 Timer1 振荡器提供时钟。这让用户能在仍使用高精度时钟源的情况下实现较低的功耗。

通过将 SCS1:SCS0 位设置为 “01”，器件进入 SEC_RUN 模式。器件时钟源切换到 Timer1 振荡器（见图 3-1），关闭主振荡器，T1RUN 位（T1CON<6>）置 1 并且 OSTS 位清零。

注： Timer1 振荡器应该在进入 SEC_RUN 模式之前就已经开始运行了。如果当 SCS1:SCS0 位被设置为“01”时，T1OSCEN 位未置 1，将不会进入 SEC_RUN 模式。如果 Timer1 振荡器已经被使能，但仍然没有开始运行，器件时钟将被延迟直到振荡器起振为止。在这种情况下，最初振荡器运行很不稳定并且它的运行结果无法预料。

从 SEC_RUN 模式转换到 PRI_RUN 模式时，在主时钟起振期间外设和 CPU 继续使用 Timer1 振荡器作为时钟源。当主时钟准备就绪以后，时钟开始切换回主时钟（见图 3-2）。当时钟切换完成后， T1RUN 位被清零，OSTS 位被置 1，主时钟提供器件时钟。唤醒不会影响 IDLEN 和 SCS 位； Timer1 振荡器继续运行。

图 3-1： 进入 SEC_RUN 模式的转换时序

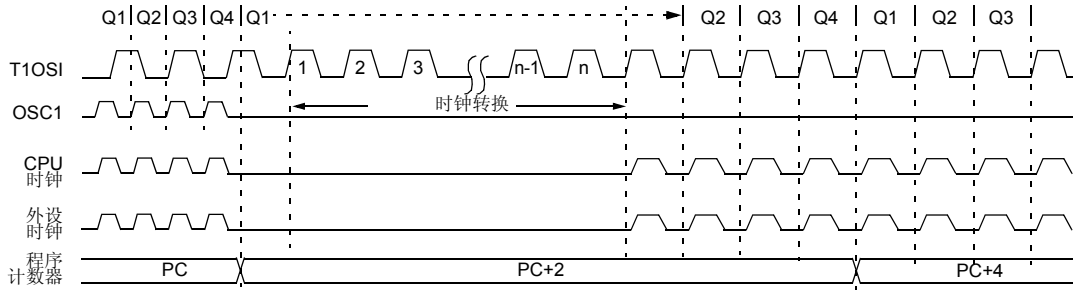
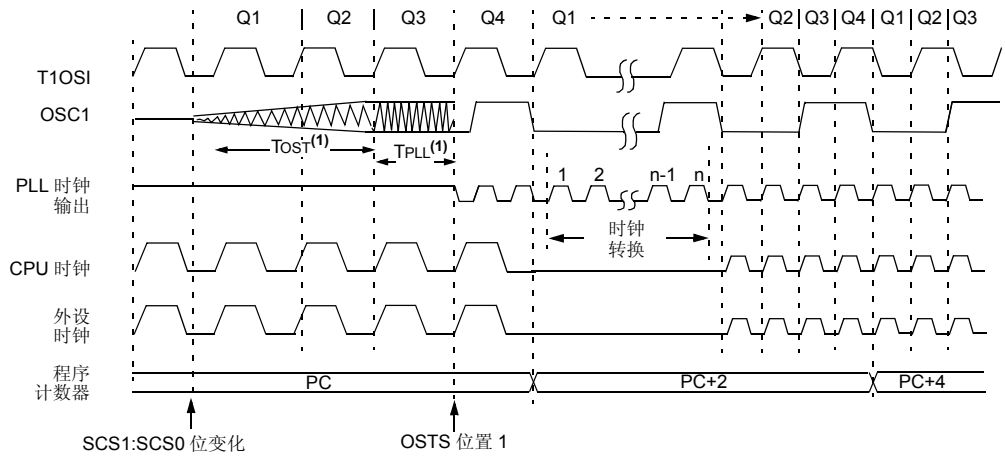


图 3-2： 从 SEC_RUN 模式到 PRI_RUN 模式 （HSPLL）的转换时序



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

3.2.3 RC_RUN 模式

在 RC_RUN 模式下，使用内部振荡器作为 CPU 和外设的时钟源；主时钟关闭。在所有运行模式之中，此模式最节约功耗，且仍然执行代码。在对时间要求不高或不总是需要高速时钟的用户应用中，选用此运行模式非常合适。

通过将 SCS 设置为 “11”可以进入此模式。当时钟源切换到 INTRC（见图 3-3）时，主振荡器关闭，OSTS 位清零。

从 RC_RUN 模式转换到 PRI_RUN 模式时，在主时钟起振期间器件继续使用 INTRC 作为时钟源。当主时钟准备就绪以后，时钟开始切换到主时钟 （见图 3-4）。当时钟切换完成后， 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 WDT 或故障保护时钟监视器， INTRC 时钟源将继续运行。

图 3-3： 到 RC_RUN 模式的转换时序

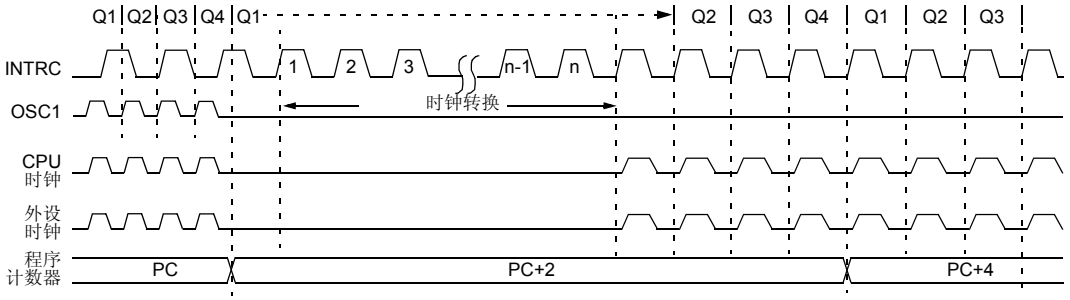
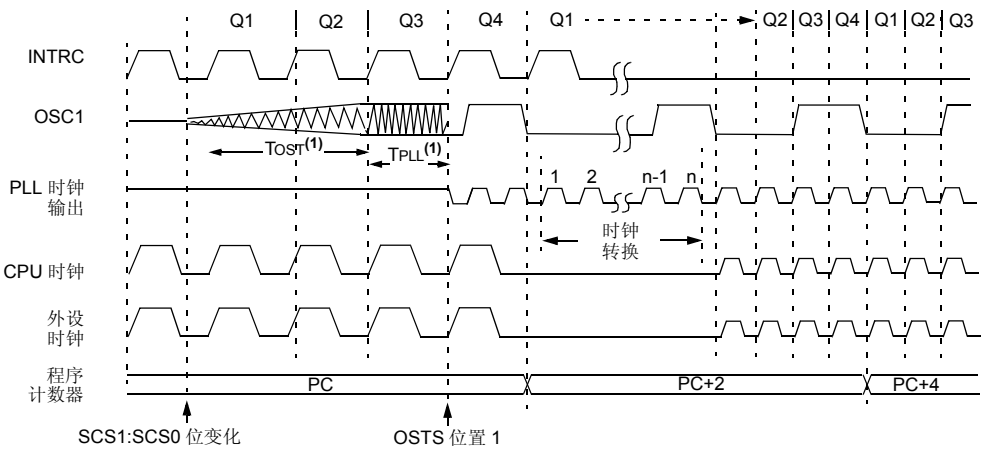


图 3-4： 从 RC_RUN 模式到 PRI_RUN 模式的转换时序



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

3.3 休眠模式

功耗管理休眠模式与所有其他 PICmicro 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 指令进入该模式。这将关闭选定的振荡器（图 3-5）。所有的时钟源状态位被清零。

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任何时钟了。如果选择了 WDT， INTRC 时钟源将继续运行。如果 Timer1 振荡器被使能， INTRC 时钟源也将继续运行。

在休眠模式下发生唤醒事件时（由于中断、复位或 WDT 超时），器件将没有时钟源直到由 SCS1:SCS0 位选定的时钟源准备就绪为止（见图 3-6），或者当双速启动或故障保护监视器被使能时，它会将内部振荡器作为时钟源（见**第 23.0 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。唤醒不会影响 IDLEN 和 SCS 位。

3.4 空闲模式

空闲模式允许在外设继续运行的情况下，有选择地关闭控制器的 CPU。选择某种特定的空闲模式使用户能进一步管理功耗。

执行 SLEEP 指令时，如果 IDLEN 位被置为 1，外设将使用由SCS1:SCS0位选定的时钟源；然而，将不会为CPU提供时钟。时钟源状态位不受影响。将 IDLEN 置 1 并执行 SLEEP 指令是一种从给定运行模式转换到其对应空闲模式的快速方法。

如果选择了 WDT， INTRC 时钟源将继续运行。如果 Timer1 振荡器被使能， INTRC 时钟源也将继续运行。

由于空闲模式CPU没有执行指令，只能通过中断、WDT 超时或复位从任一空闲模式退出。当唤醒事件发生时，CPU 在准备好执行代码前，要延迟一个 TCSD 间隔时间（表 26-12 的参数 38 ）。当 CPU 开始执行代码时，它将使用与当前空闲模式相同的时钟源以恢复工作。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（换句话说就是 RC_RUN 模式）。唤醒不会影响 IDLEN 和 SCS 位。

当处于任何空闲模式或休眠模式时， WDT 超时将导致 WDT 唤醒到由当前 SCS1:SCS0 位指定的运行模式。

图 3-5： 进入休眠模式的转换时序

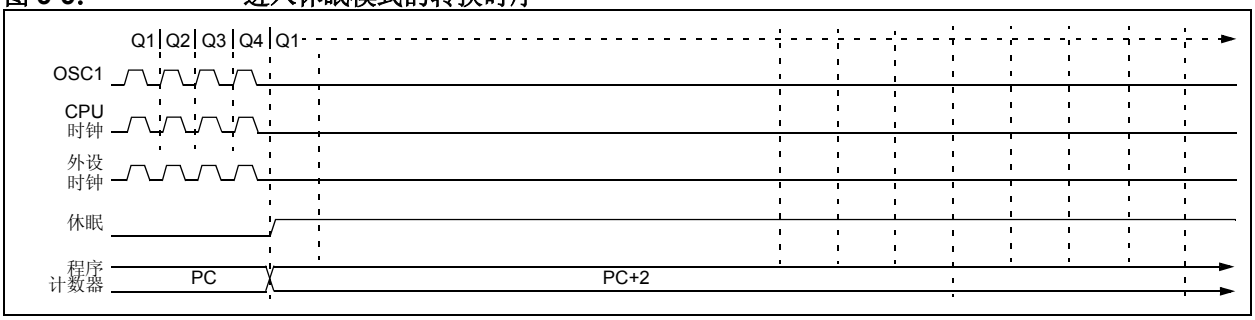
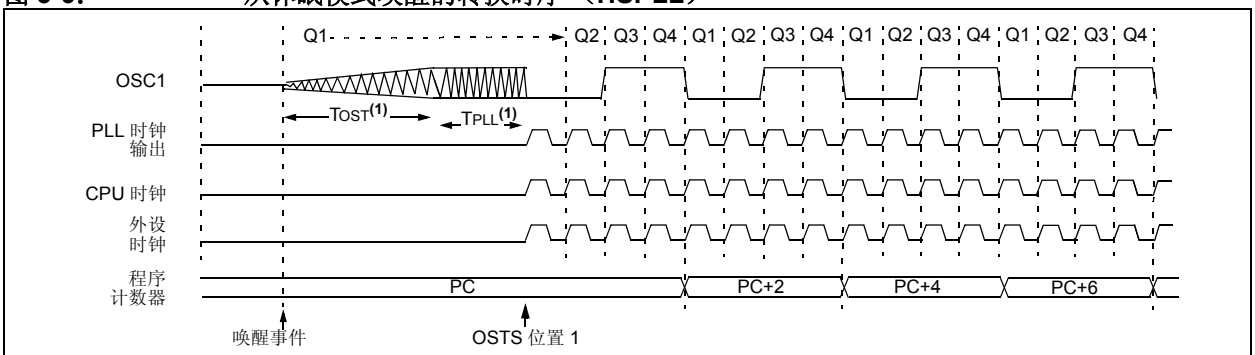


图 3-6： 从休眠模式唤醒的转换时序 （HSPLL）



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

3.4.1 PRI_IDLE 模式

此模式在三种低功耗空闲模式中是唯一不禁止器件主时钟的。对于对时间精度要求很高的应用来说，由于时钟源不需要“热身”或从其他振荡器切换，选用此模式可以使用更加精确的主时钟源，并以最快的速度恢复器件运行。

通过置位 IDLEN 位并执行 SLEEP 指令可以从 PRI_RUN 模式进入 PRI_IDLE 模式。如果器件处于其他运行模式，请先置位 IDLEN，然后将 SCS 位置为“10”并执行 SLEEP 指令。虽然 CPU 被禁止，但外设仍继续使用由 FOSC1:FOSC0 配置位指定的主时钟源作为时钟源。OSTS 位保持置 1（见图 3-7）。

当唤醒事件发生时，CPU 由主时钟源提供时钟。在唤醒事件和代码开始执行之间需要一段 TCSD 间隔的延迟。需要这段时间以使 CPU 为执行指令做好准备。在唤醒之后，OSTS 位保持置 1。唤醒不会影响 IDLEN 和 SCS 位（见图 3-8）。

3.4.2 SEC_IDLE 模式

在 SEC_IDLE 模式中， CPU 被禁止，但外设继续使用 Timer1 振荡器作为时钟源。通过置位 IDLEN 位并执行 SLEEP 指令可以从 SEC_RUN 模式进入此模式。如果器件处于其他运行模式，请先置位 IDLEN，然后将 SCS1:SCS0 置为“01”并执行 SLEEP 指令。当时钟源切换到 Timer1 振荡器时，主振荡器关闭， OSTS 位清零， T1RUN 位被置 1。

当唤醒事件发生时，外设继续将 Timer1 振荡器作为时钟源。在唤醒事件后的 TCSD 间隔之后， CPU 使用 Timer1 振荡器作为时钟源并开始执行代码。唤醒不会影响 IDLEN 和 SCS 位； Timer1 振荡器继续运行 （见图 3-8）。

注：	Timer1 振荡器应该在进入 SEC_IDLE 模式之前已经在运行了。如果执行 SLEEP 指令时 T1OSCEN 位没有被置 1，那么 SLEEP 指令会被忽略并且不会进入 SEC_IDLE 模式。如果 Timer1 振荡器已经被使能，但还没有开始运行，外设时钟将被延迟直到振荡器起振为止。在这种情况下，初始振荡器运行很不稳定并且它的运行结果无法预料。

图 3-7： 进入空闲模式的转换时序

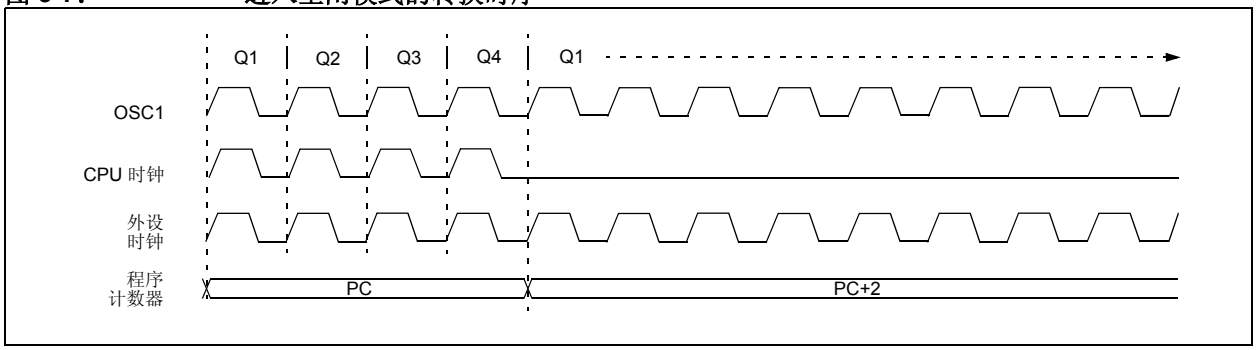
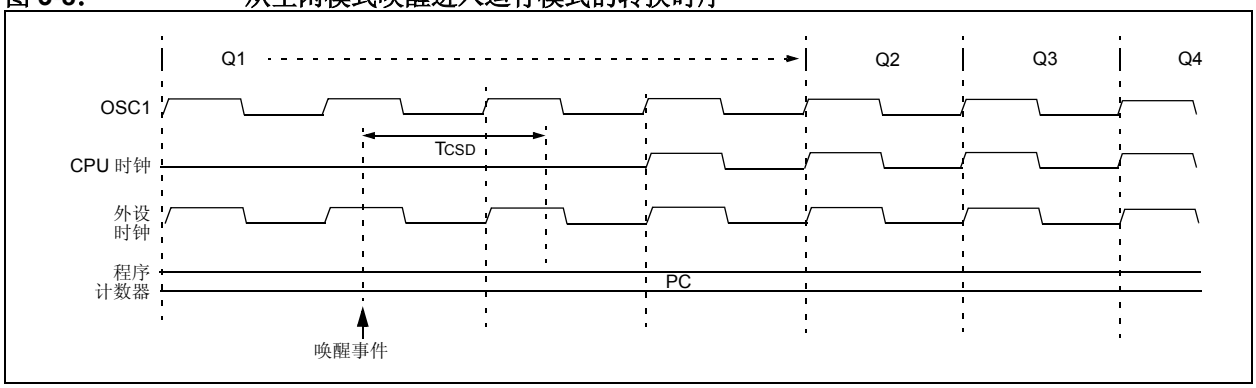


图 3-8： 从空闲模式唤醒进入运行模式的转换时序



3.4.3 RC_IDLE 模式

在 RC_IDLE 模式中，CPU 被禁止，但外设继续使用内部振荡器作为时钟源。此模式允许在器件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，请先置位 IDLEN，然后将 SCS 位清零并执行 SLEEP 指令。当时钟源切换到 INTRC 时，主振荡器关闭， OSTS 位被清零。

当唤醒事件发生时，外设继续将INTRC作为时钟源。在唤醒事件后的 TCSD 延迟后， CPU 使用 INTRC 作为时钟源并开始执行代码。唤醒不会影响IDLEN和SCS位。如果 WDT 或故障保护时钟监视器被使能，INTRC 时钟源将继续运行。

3.5 退出空闲和休眠模式

通过中断、复位或 WDT 超时作为触发事件，从休眠模式或任何空闲模式退出。本节将讨论引起从功耗管理模式退出的触发事件。在每个功耗管理模式小节中，还讨论了时钟控制子系统的操作（见**第 3.2 节“运行模式”**、**第 3.3 节 “休眠模式”**和**第 3.4 节 “空闲模式”**）。

3.5.1 通过中断退出

任何可用的中断源可以引起器件从空闲模式或休眠模式退出，进入运行模式。要使能此功能，必须通过将 INTCON 或 PIE 寄存器中的相应中断允许位置 1 来允许该中断源。当相应的中断标志位被置 1 时，启动退出时序。

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，代码执行就会跳转到中断矢量。否则，代码执行就会继续或恢复，而不发生跳转（见**第 9.0 节 “中断”**）。

从休眠或空闲模式退出时，一个固定的延迟时间间隔 TCSD 是必需的。 CPU 需要这段延迟时间来为执行代码做准备。指令执行在此延迟后的第一个时钟周期恢复。

3.5.2 通过 WDT 超时退出

根据 WDT 超时发生时器件所处的不同功耗管理模式会引起不同的行为。

如果器件没有执行代码 （在所有空闲模式和休眠模式下），超时将导致从功耗管理模式退出 （见**第 3.2 节 “运行模式”**和**第 3.3 节 “休眠模式”**）。如果器件正在执行代码 （在所有运行模式下），超时将导致 WDT 复位 （见**第 23.2 节 “看门狗定时器 （WDT）”**）。

WDT 和后分频器将被以下任一事件清零：

- 执行 SLEEP 或 CLRWDT 指令
- 当前选定的时钟源失效 （故障保护时钟监视器使能时）

3.5.3 通过复位退出

通过复位从空闲或休眠模式退出，自动强制器件使用 INTRC 作为时钟源运行。

3.5.4 在没有振荡器起振延迟的情况下退出

从功耗管理模式的某些退出方式根本不会启动 OST。具体有两种情形：

- 主时钟源处于不会被停止的 PRI_IDLE 模式；
- 主时钟源处于 EC 模式或 ECPLL 模式。

在这两种情况下，主时钟源都不需要振荡器起振延迟，因为 （在 PRI_IDLE 模式下），主时钟源已经运行了，而 （在 EC 或 ECPLL 模式下），一般也不需要振荡器起振延迟。然而，当退出休眠和空闲模式时，在唤醒事件后需要一个固定的延迟时间间隔 TCSD， CPU 利用这段时间为执行代码做准备。指令执行在此延迟后的第一个时钟周期恢复。

注：

4.0 复位

PIC18F87J10 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 （执行程序期间）
e）欠压复位 （Brown-on Reset， BOR）
f） RESET 指令
g）堆栈满复位
h）堆栈下溢复位

本节讨论了由 $\overline{\text{MCLR}}$、POR 和 BOR 产生的复位以及各种起振定时器的操作。**第 5.1.6.4 节 “堆栈满和下溢复位”**将介绍堆栈复位事件。而**第 23.2 节 “看门狗定时器 （WDT）”**将介绍 WDT 复位。

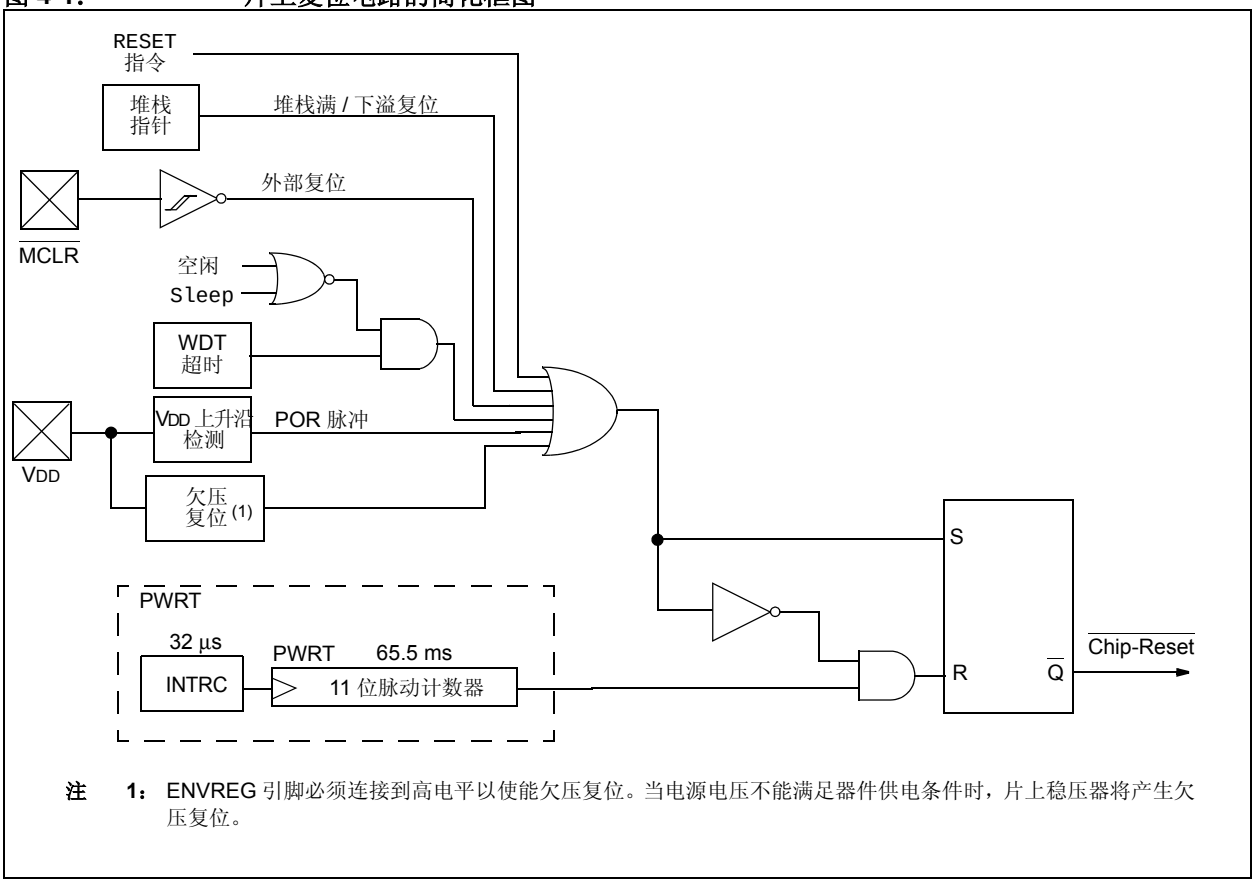
图 4-1 给出了片上复位电路的简化框图。

4.1 RCON 寄存器

可通过 RCON 寄存器 （寄存器 4-1）跟踪器件复位事件。寄存器的低 5 位表示已经发生的特定复位事件。在大部分情况下，只有复位事件可以将这些位置 1 而且它们必须在复位事件之后由应用程序清零。一起读这些标志位的状态可以显示出刚产生的复位的类型。**第 4.6 节 “寄存器的复位状态”**更详细的讨论了这些位。

RCON 寄存器还有一个设置中断优先级的控制位（IPEN）。**第 9.0 节 “中断”**将详细讨论中断优先级。

图 4-1： 片上复位电路的简化框图



注 1： ENVREG 引脚必须连接到高电平以使能欠压复位。当电源电压不能满足器件供电条件时，片上稳压器将产生欠压复位。

寄存器 4-1：　RCON：复位控制寄存器

R/W-0	U-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

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 6-5　**未用**：读为 0

bit 4　**$\overline{RI}$**：RESET 指令标志位
1 = 未执行 RESET 指令 （仅由固件置 1）
0 = 执行 RESET 指令导致器件复位 （必须在欠压复位发生之后用软件置 1）

bit 3　**$\overline{TO}$**：看门狗定时器超时标志位
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 22
1 = 未发生上电复位 （仅由固件置 1）
0 = 发生上电复位 （必须在发生上电复位后用软件置 1）

bit 0　**$\overline{BOR}$**：欠压复位状态位
1 = 未发生欠压复位 （仅由固件置 1）
0 = 发生了欠压复位 （必须在欠压复位发生之后用软件置 1）

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

注　**1**：建议在检测到上电复位后，将 $\overline{POR}$ 位置 1，以便检测后续发生的上电复位。
2：如果禁止了片上稳压器， $\overline{BOR}$ 则总是保持为 0。欲知更多信息，请参见**第 4.4.1 节 “检测 BOR”**。
3：欠压复位是指当 $\overline{BOR}$ 为 0 且 POR 为 1 时发生的复位（假设在上电复位后 $\overline{POR}$ 立即被软件置 1）。

4.2 主复位（MCLR）

MCLR 引脚提供触发硬件外部复位器件的方法。保持引脚为低电平就能触发此复位。PIC18 扩展的单片机器件在 MCLR 复位信号传输路径中有一个噪声滤波器，它可以检测和忽略小脉冲信号的干扰。

任何内部复位包括 WDT 复位都不能将 MCLR 引脚驱动为低电平。

4.3 上电复位（POR）

只要 VDD 上升到一定的门限以上，就会在片上产生上电复位信号。这让器件在 VDD 足以开始工作时，从初始状态开始工作。

要利用 POR 电路，可以将 MCLR 引脚通过一个电阻（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所需的外部 RC 元件。VDD 的最小上升速率已指定（参数 D004）。对于用于延缓 VDD 上升时间的情况，请参见图 4-2。

当器件开始正常工作（即退出复位状态）时，器件的工作参数（电压、频率和温度等）必须满足，以确保其正常工作。如果这些条件不满足，器件必须保持在复位状态，直到工作条件满足为止。

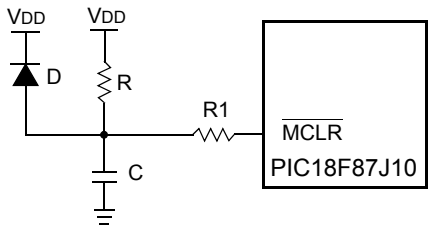
POR 事件由 POR 位（RCON<1>）捕获。只要发生 POR，此位的状态就被置为 0，它不会因任何其他复位事件而发生改变。任何硬件事件都不会将 POR 位复位为 1。为了捕获多个事件，在 POR 后用户用软件手动将该位复位为 1 。

4.4 欠压复位（BOR）

当内部稳压器被使能时（ENVREG 引脚连接到 VDD），PIC18F87J10 系列器件加入了简单的 BOR 功能。只要 VDD 低于 VBOR（参数 D005）的时间大于 TBOR（参数 35）就会复位器件。如果 VDD 降到 VBOR 以下的时间小于 TBOR，器件是否发生复位不确定。芯片将保持欠压复位状态，直至 VDD 电压上升到 VBOR 以上。

一旦发生BOR，上电延时定时器将芯片保持在复位状态的时间就是 TPWRT（参数 33）。如果上电延迟定时器运行时，VDD 电压降到 VBOR 以下，芯片将重新回到欠压复位状态并且初始化上电延时定时器。一旦 VDD 电压上升到VBOR以上，上电延时定时器将再执行一个延时。

图 4-2： 外部上电复位电路（用于延缓 VDD 上电）



注 1： 仅当 VDD 上电很慢时，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 以确保电阻 R 上的压降符合器件的电气规范要求。

3： R1 ≥ 1kΩ 将限制任何电流从外部电容 C 流入 MCLR，以避免由于静电放电（Electrostatic Discharge， ESD）或电超载（Electrical Overstress， EOS）导致 MCLR/VPP 引脚击穿。

4.4.1 检测 BOR

BOR 位在 BOR 或 POR 事件时总是复位为 0。这使得很难仅通过读 BOR 的状态就判断 BOR 事件是否已经发生。更可靠的方法是同时检查 POR 和 BOR 的状态。假设在 POR 事件之后立即用软件将 POR 位复位为 1。如果 BOR 为 0 而 POR 为 1，就可以可靠地判断出已经发生了 BOR 事件。

如果禁止稳压器，也会禁止欠压复位功能。在这种情况下，不能使用 BOR 位来确定 BOR 事件。 POR 事件仍然会将 BOR 位清零。

4.5 上电延时定时器 （PWRT）

PIC18F87J10 系列器件加入了片上上电延时定时器（PWRT）以帮助稳定上电复位过程。PWRT 总是使能的。其主要功能是确保在执行代码之前，器件的电压是稳定的。

PIC18F87J10 系列器件的上电延时定时器（PWRT）是一个 11 位计数器，使用 INTRC 时钟源作为时钟输入。这就产生大约共计 2048 x 32 μs = 65.6 ms 的时间间隔。当 PWRT 计数时，器件保持在复位状态。

上电延迟时间取决于 INTRC 时钟，而且由于温度和制造工艺的变化将导致不同器件的延迟时间各不相同。欲知详细信息，请参见 DC 参数 33。

4.5.1 延时序列

如果 PWRT 被使能的话，在 POR 脉冲被清零后触发 PWRT 延时。总的延时会根据 PWRT 的状态而有所不同。图 4-3、图 4-4、图 4-5和图 4-6都说明了在使能上电延时定时器时的上电延时序列。

由于延时是由 POR 脉冲引起的，因此若 $\overline{\text{MCLR}}$ 引脚保持足够长时间的低电平，延时将结束。将 MCLR 电平拉高后器件将立即进入执行状态 （图 4-5）。这对于测试或同步多个并行的 PIC18FXXXX 器件是非常有用的。

图 4-3： 上电延时序列 （$\overline{\text{MCLR}}$ 连接到 VDD， VDD 上升时间 < TPWRT）

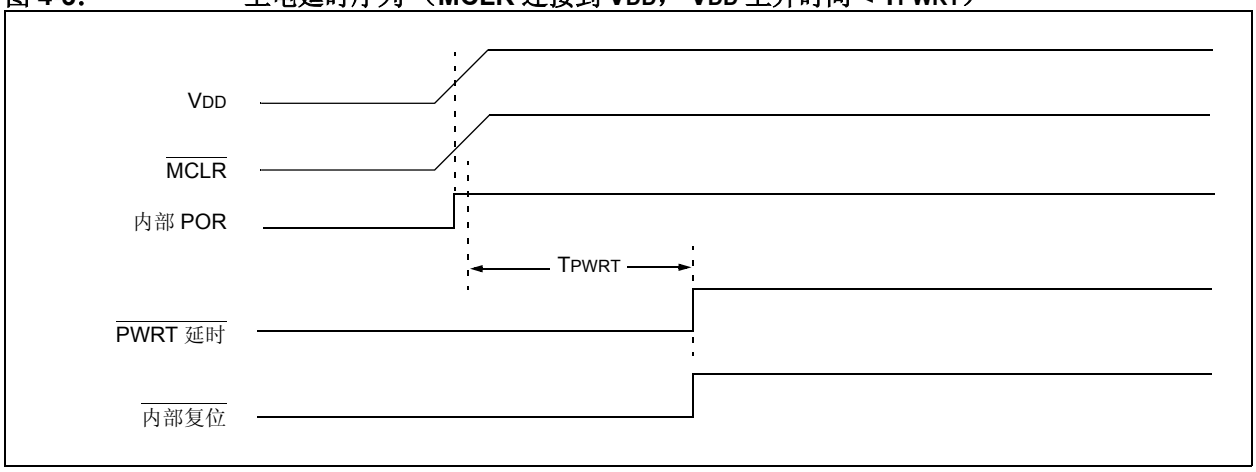


图 4-4： 上电延时序列 （$\overline{\text{MCLR}}$ 不连接到 VDD）：情形 1

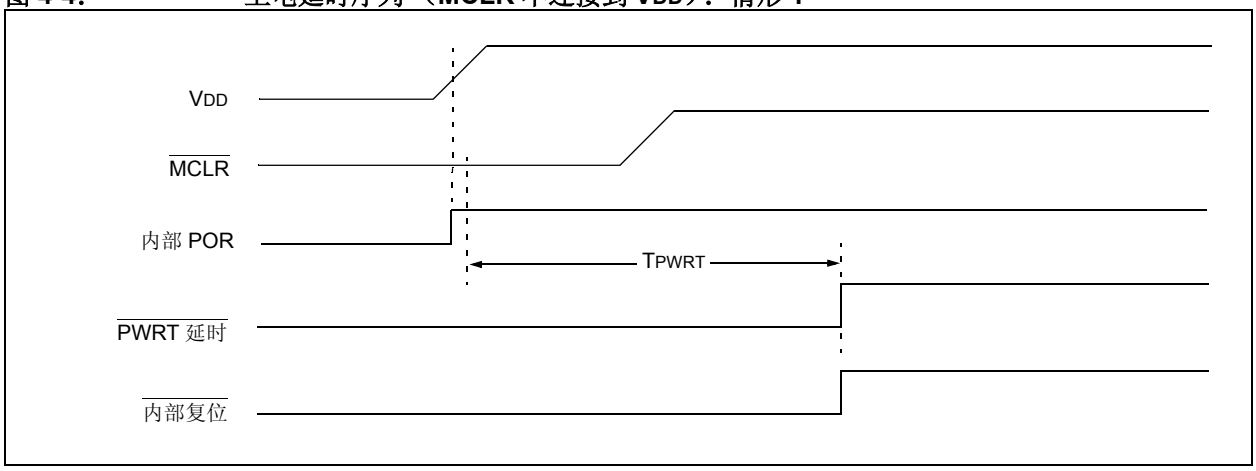


图 4-5：　　上电延时序列（$\overline{\text{MCLR}}$ 未连接到 VDD）：情形 2

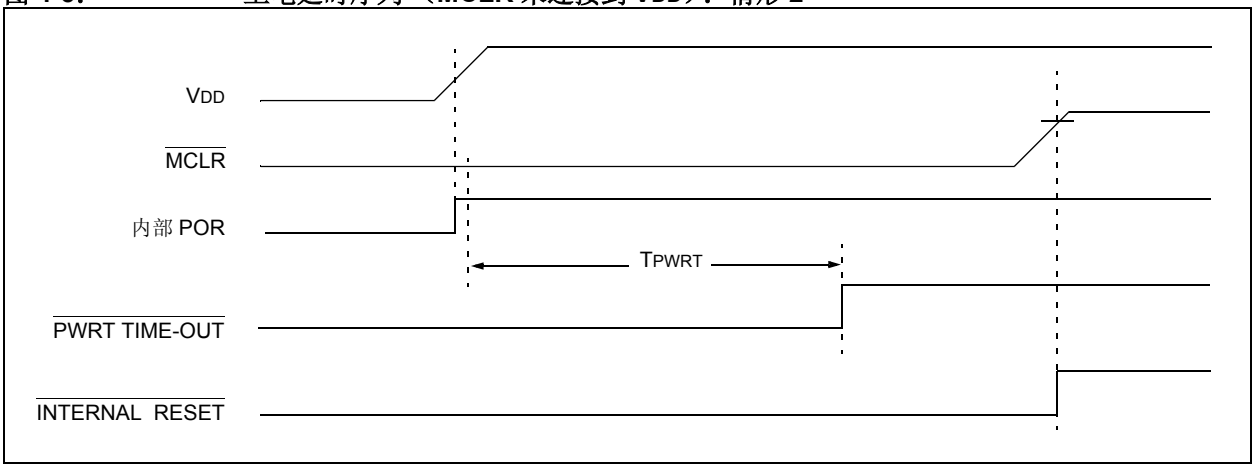
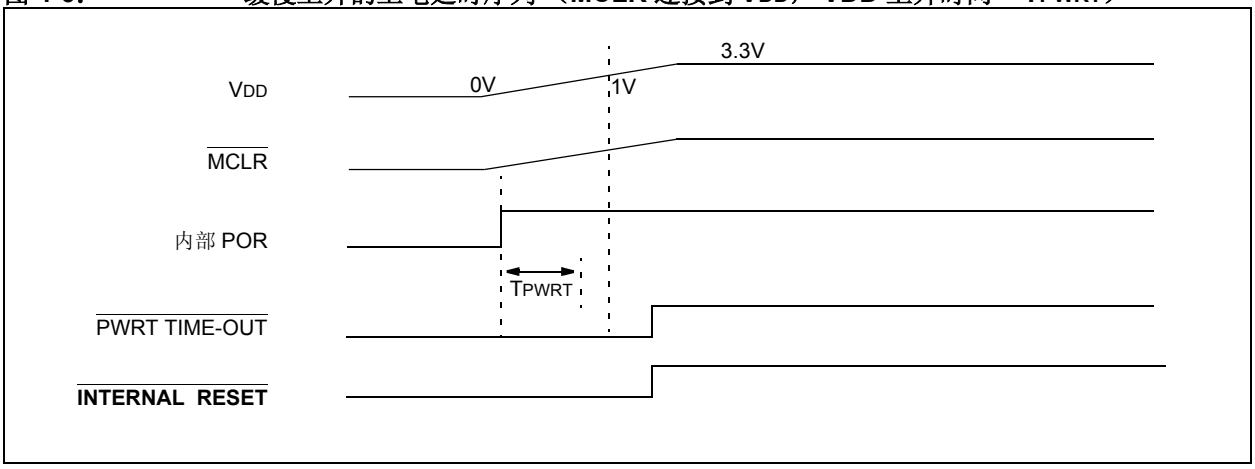


图 4-6：　　缓慢上升的上电延时序列（$\overline{\text{MCLR}}$ 连接到 VDD，VDD 上升时间 > TPWRT）



4.6 寄存器的复位状态

大多数寄存器不受复位的影响。在上电复位时寄存器状态未知，且发生其他复位时寄存器状态不会改变。其他寄存器是否被强行置为“复位状态”，取决于所发生复位的类型。

大多数寄存器不受 WDT 唤醒的影响，这是因为 WDT 唤醒被视为恢复正常工作。如表 4-1 所示，RCON 寄存器的状态位 $\overline{RI}$、$\overline{TO}$、$\overline{PD}$、$\overline{POR}$ 和 $\overline{BOR}$，在不同的复位情况下置 1 和清零的状态也各不相同。这些状态位在软件中用于判断复位的类型。

表 4-2 说明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 唤醒复位几类。

表 4-1： RCON 寄存器的状态位、它们的含义以及初始化条件

条件	程序计数器(1)	RCON 寄存器					STKPTR 寄存器	
		$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	STKFUL	STKUNF
上电复位	0000h	1	1	1	0	0	0	0
RESET 指令	0000h	0	u	u	u	u	u	u
欠压复位	0000h	1	1	1	u	0	u	u
功耗管理运行模式期间的 $\overline{MCLR}$	0000h	u	1	u	u	u	u	u
功耗管理空闲模式和休眠模式期间时的 $\overline{MCLR}$	0000h	u	1	0	u	u	u	u
全功耗或功耗管理运行模式期间的 WDT 超时	0000h	u	0	u	u	u	u	u
全功耗运行期间的 $\overline{MCLR}$	0000h	u	u	u	u	u	u	u
堆栈满复位 （STVREN = 1）	0000h	u	u	u	u	u	1	u
堆栈下溢复位 （STVREN = 1）	0000h	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	1
功耗管理空闲或休眠模式期间的 WDT 超时溢出	PC+2	u	0	0	u	u	u	u
中断使器件从功耗管理模式退出	PC + 2	u	u	0	u	u	u	u

图注： u = 不变

注 1： 当芯片被中断唤醒且 GIEH 或 GIEL 位被置 1 时， PC 装入中断矢量 （0008h 或 0018h）。

表 4-2： 所有寄存器的初始化条件

寄存器	适用器件		上电复位， 欠压复位	$\overline{\text{MCLR}}$ 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
TOSU	PIC18F6XJ1X	PIC18F8XJ1X	---0 0000	---0 0000	---0 uuuu(1)
TOSH	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu(1)
TOSL	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu(1)
STKPTR	PIC18F6XJ1X	PIC18F8XJ1X	00-0 0000	uu-0 0000	uu-u uuuu(1)
PCLATU	PIC18F6XJ1X	PIC18F8XJ1X	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	PIC18F6XJ1X	PIC18F8XJ1X	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F6XJ1X	PIC18F8XJ1X	0000 000x	0000 000u	uuuu uuuu(3)
INTCON2	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu(3)
INTCON3	PIC18F6XJ1X	PIC18F8XJ1X	1100 0000	1100 0000	uuuu uuuu(3)
INDF0	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
POSTINC0	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
POSTDEC0	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
PREINC0	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
PLUSW0	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
FSR0H	PIC18F6XJ1X	PIC18F8XJ1X	---- xxxx	---- uuuu	---- uuuu
FSR0L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
POSTINC1	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
POSTDEC1	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
PREINC1	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
PLUSW1	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
FSR1H	PIC18F6XJ1X	PIC18F8XJ1X	---- xxxx	---- uuuu	---- uuuu
FSR1L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F6XJ1X	PIC18F8XJ1X	---- 0000	---- 0000	---- uuuu

图注： u = 不变， x = 未知， - = 未用 （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： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。 STKPTR 被修改为指向硬件堆栈中的下一个单元。

2： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量地址 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （用以唤醒器件）。

4： 关于特定条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化条件（续）

寄存器	适用器件		上电复位， 欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
INDF2	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
POSTINC2	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
POSTDEC2	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
PREINC2	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
PLUSW2	PIC18F6XJ1X	PIC18F8XJ1X	N/A	N/A	N/A
FSR2H	PIC18F6XJ1X	PIC18F8XJ1X	---- xxxx	---- uuuu	---- uuuu
FSR2L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	PIC18F6XJ1X	PIC18F8XJ1X	---x xxxx	---u uuuu	---u uuuu
TMR0H	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
TMR0L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F6XJ1X	PIC18F8XJ1X	0--- q-00	0--- q-00	u--- q-uu
WDTCON	PIC18F6XJ1X	PIC18F8XJ1X	---- ---0	---- ---0	---- ---u
RCON(4)	PIC18F6XJ1X	PIC18F8XJ1X	0--1 1100	0--q qquu	u--u qquu
TMR1H	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	1111 1111
T2CON	PIC18F6XJ1X	PIC18F8XJ1X	-000 0000	-000 0000	-uuu uuuu
SSP1BUF	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP1ADD	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
SSP1STAT	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
SSP1CON1	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
SSP1CON2	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
ADRESH	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F6XJ1X	PIC18F8XJ1X	0-00 0000	0-00 0000	u-uu uuuu
ADCON1	PIC18F6XJ1X	PIC18F8XJ1X	--00 0000	--00 0000	--uu uuuu
ADCON2	PIC18F6XJ1X	PIC18F8XJ1X	0-00 0000	0-00 0000	u-uu uuuu

图注： u = 不变， x = 未知， - = 未用 （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： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。 STKPTR 被修改为指向硬件堆栈中的下一个单元。

2： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量地址 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （用以唤醒器件）。

4： 关于特定条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化条件（续）

寄存器	适用器件		上电复位， 欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
CCPR1H	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
CCPR2H	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
CCPR3H	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR3L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP3CON	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
ECCP1AS	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
CVRCON	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
CMCON	PIC18F6XJ1X	PIC18F8XJ1X	0000 0111	0000 0111	uuuu uuuu
TMR3H	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	uuuu uuuu	uuuu uuuu
PSPCON	PIC18F6XJ1X	PIC18F8XJ1X	0000 ----	0000 ----	uuuu ----
SPBRG1	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
RCREG1	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
TXREG1	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
TXSTA1	PIC18F6XJ1X	PIC18F8XJ1X	0000 0010	0000 0010	uuuu uuuu
RCSTA1	PIC18F6XJ1X	PIC18F8XJ1X	0000 000x	0000 000x	uuuu uuuu
IPR3	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
PIR3	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu(3)
PIE3	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
IPR2	PIC18F6XJ1X	PIC18F8XJ1X	11-- 1-11	11-- 1-11	uu-- u-uu
PIR2	PIC18F6XJ1X	PIC18F8XJ1X	00-- 0-00	00-- 0-00	uu-- u-uu(3)
PIE2	PIC18F6XJ1X	PIC18F8XJ1X	00-- 0-00	00-- 0-00	uu-- u-uu
IPR1	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
PIR1	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu(3)
PIE1	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
MEMCON	PIC18F6XJ1X	PIC18F8XJ1X	0-00 --00	0-00 --00	u-uu --uu
OSCTUNE	PIC18F6XJ1X	PIC18F8XJ1X	-0-- ----	-0-- ----	-u-- ----

图注： u = 不变， x = 未知， - = 未用 （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： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。 STKPTR 被修改为指向硬件堆栈中的下一个单元。

2： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量地址 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （用以唤醒器件）。

4： 关于特定条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化条件(续)

寄存器	适用器件		上电复位， 欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
TRISJ	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
TRISH	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
TRISG	PIC18F6XJ1X	PIC18F8XJ1X	---1 1111	---1 1111	---u uuuu
TRISF	PIC18F6XJ1X	PIC18F8XJ1X	1111 111-	1111 111-	uuuu uuu-
TRISE	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
TRISD	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	uuuu uuuu
TRISA	PIC18F6XJ1X	PIC18F8XJ1X	--11 1111	--11 1111	--uu uuuu
LATJ	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG	PIC18F6XJ1X	PIC18F8XJ1X	---x xxxx	---u uuuu	---u uuuu
LATF	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxx-	uuuu uuu-	uuuu uuu-
LATE	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA	PIC18F6XJ1X	PIC18F8XJ1X	--xx xxxx	--uu uuuu	--uu uuuu
PORTJ	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTH	PIC18F6XJ1X	PIC18F8XJ1X	0000 xxxx	uuuu uuuu	uuuu uuuu
PORTG	PIC18F6XJ1X	PIC18F8XJ1X	111x xxxx	111u uuuu	uuuu uuuu
PORTF	PIC18F6XJ1X	PIC18F8XJ1X	x000 000-	x000 000-	uuuu uuu-
PORTE	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTA	PIC18F6XJ1X	PIC18F8XJ1X	--0x 0000	--0u 0000	--uu uuuu
SPBRGH1	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
BAUDCON1	PIC18F6XJ1X	PIC18F8XJ1X	01-0 0-00	01-0 0-00	uu-u u-uu
SPBRG2	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
BAUDCON2	PIC18F6XJ1X	PIC18F8XJ1X	01-0 0-00	01-0 0-00	uu-u u-uu

图注： u = 不变， x = 未知， - = 未用 （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： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。 STKPTR 被修改为指向硬件堆栈中的下一个单元。

2： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量地址 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （用以唤醒器件）。

4： 关于特定条件下的复位值，请参见表 4-1。

表 4-2： 所有寄存器的初始化条件(续)

寄存器	适用器件		上电复位， 欠压复位	MCLR 复位 WDT 复位 RESET 指令 堆栈复位	通过 WDT 或中断唤醒器件
ECCP1DEL	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
TMR4	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
PR4	PIC18F6XJ1X	PIC18F8XJ1X	1111 1111	1111 1111	1111 1111
T4CON	PIC18F6XJ1X	PIC18F8XJ1X	-000 0000	-000 0000	-uuu uuuu
CCPR4H	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR4L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP4CON	PIC18F6XJ1X	PIC18F8XJ1X	--00 0000	--00 0000	--uu uuuu
CCPR5H	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR5L	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP5CON	PIC18F6XJ1X	PIC18F8XJ1X	--00 0000	--00 0000	--uu uuuu
SPBRG2	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
RCREG2	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
TXREG2	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
TXSTA2	PIC18F6XJ1X	PIC18F8XJ1X	0000 0010	0000 0010	uuuu uuuu
RCSTA2	PIC18F6XJ1X	PIC18F8XJ1X	0000 000x	0000 000x	uuuu uuuu
ECCP3AS	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
ECCP3DEL	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
ECCP2AS	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
ECCP2DEL	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
SSP2BUF	PIC18F6XJ1X	PIC18F8XJ1X	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP2ADD	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
SSP2STAT	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
SSP2CON1	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu
SSP2CON2	PIC18F6XJ1X	PIC18F8XJ1X	0000 0000	0000 0000	uuuu uuuu

图注： u = 不变， x = 未知， - = 未用 （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： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时，用 PC 的当前值更新 TOSU、 TOSH 和 TOSL。 STKPTR 被修改为指向硬件堆栈中的下一个单元。

2： 当芯片被中断唤醒且 GIEL 或 GIEH 位被置 1 时， PC 装入中断向量地址 （0008h 或 0018h）。

3： INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 （用以唤醒器件）。

4： 关于特定条件下的复位值，请参见表 4-1。

注：

5.0 存储器构成

在 PIC18 闪存单片机器件上有两种类型的存储器：

- 程序存储器
- 数据 RAM

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

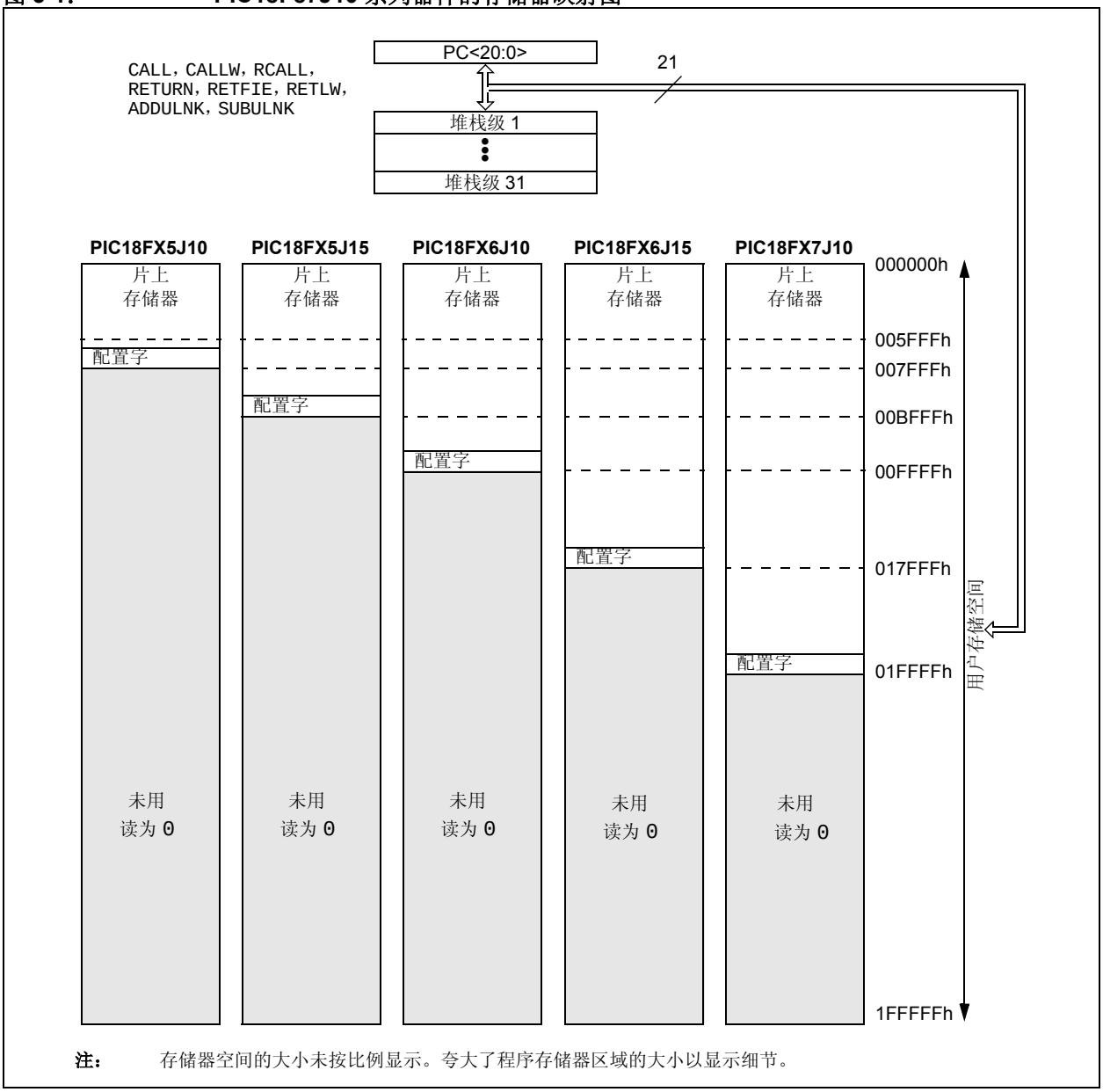
第 6.0 节 “闪存程序存储器”提供了关于闪存程序存储器工作的更多详细信息。

5.1 程序存储器构成

PIC18 单片机具备一个 21 位程序计数器，可以对 2 MB 的程序存储器空间进行寻址。访问物理实现的存储器的上边界和 2MB 地址之间的地址单元将会返回全 “0”（相当于执行 NOP 指令）。

整个 PIC18F87J10 系列提供了一系列片上闪存程序存储空间，从 32 KB（至多 16,384 条单字指令）到 128 KB（65,536 条单字指令）。该系列的各个器件的程序存储器映射图如图 5-3 所示。

图 5-1： PIC18F87J10 系列器件的存储器映射图



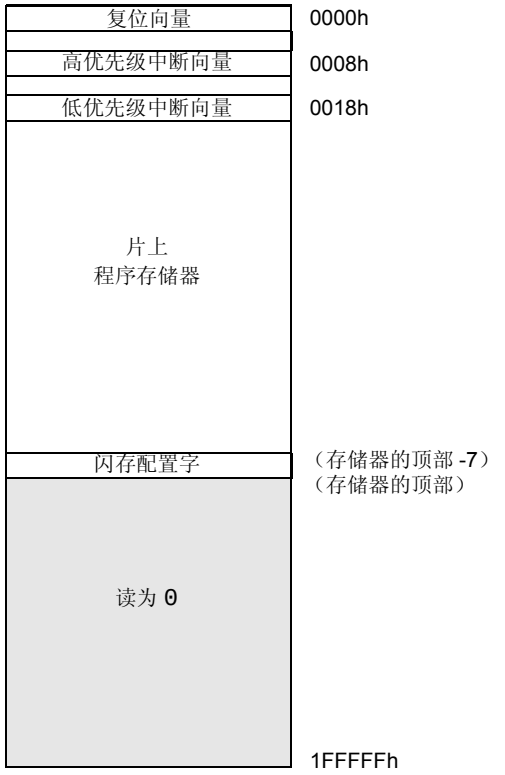
注： 存储器空间的大小未按比例显示。夸大了程序存储器区域的大小以显示细节。

5.1.1 存储器硬件编码向量

所有的 PIC18 器件在它们的程序存储器空间内共有 3 个硬件编码的返回向量。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。它们在程序存储器映射图中的相对位置如图 5-2 所示。

图 5-2： PIC18F87J10 系列器件的硬编码向量和配置字单元



图注： （存储器的顶部）代表片上程序存储器空间的上边界（有关器件特定的参数见图 5-1）。阴影区域表示未用的存储空间。这些区域未按比例显示。

5.1.2 闪存配置字

由于 PIC18F87J10 系列器件没有固定的配置寄存器，所以保留片上程序存储器顶部的 4 个字来保存配置信息。复位时，该配置信息被复制到配置寄存器。

配置字以数字顺序存储在程序存储器单元内，从最低地址开始存放，从 CONFIG1 的低字节开始，到 CONFIG4 的高字节结束。对于这些器件，只使用从 CONFIG1 到 CONFIG3 的配置字，保留 CONFIG4。PIC18F87J10 系列器件的闪存配置字的实际地址如表 5-1 所示。图 5-2中显示了闪存配置字以及其他的存储器向量在存储器映射图中的位置。

第 23.1 节 “配置位”中提供了有关器件配置字的更多详细信息。

表 5-1： PIC18F87J10 系列器件的闪存配置字

器件	程序存储器（KB）	配置字地址
PIC18F65J10	32	7FF8h 到 7FFFh
PIC18F85J10		
PIC18F65J15	48	BFF8h 到 BFFFh
PIC18F85J15		
PIC18F66J10	64	FFF8h 到 FFFFh
PIC18F86J10		
PIC18F66J15	96	17FF8h 到 17FFFh
PIC18F86J15		
PIC18F67J10	128	1FFF8h 到 1FFFFh
PIC18F87J10		

5.1.3 PIC18F8XJ10/8XJ15 程序存储器模式

此系列中的80引脚器件可以对总共2 MB的程序存储器空间进行寻址。这是通过外部存储器总线实现的。控制器有两种不同的工作模式：

- 单片机（MC）
- 扩展单片机（EMC）

通过设置 EMB 配置位（CONFIG3L<5:4>）决定程序存储器的模式，如寄存器 5-1 所示。（欲知有关器件配置位的更多详细信息，请参见**第 23.1 节“配置位”**。）

程序存储器模式的工作方式如下：

- **单片机模式**只访问片上闪存存储器。尝试读片上存储器顶部以上的地址单元会导致全读为 0（相当于执行 NOP 指令）。

  单片机模式也是 64 引脚器件惟一可用的工作模式。

- **扩展单片机模式**允许将内部和外部程序存储器作为一个存储器块进行访问。器件可以访问其整个的片上程序存储器；除此之外，器件可以访问外部程序存储器最大限制为 2MB 的程序存储空间。按照需要自动在两个存储器之间切换执行。

EMB 配置位的设置还能控制外部存储器总线的地址总线宽度。在**第 7.0 节“外部存储总线”**中对此有更详细的论述。

在这两种模式下，单片机都能完全地访问数据 RAM。

图 5-3 比较了不同程序存储器模式的存储器映射图。表 5-2中更充分地说明了片上存储器和外部存储器访问限制之间的差异。

寄存器 5-1：　CONFIG3L：配置寄存器 3 低字节

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0
WAIT	BW	EMB1	EMB0	EASHFT	—	—	—
bit 7							bit 0

bit 7　**WAIT**：外部总线等待使能位
1 = 外部总线上的等待状态被禁止
0 = 通过 MEMCON<5:4> 选择并使能外部总线上的等待状态

bit 6　**BW**：数据总线宽度选择位
1 = 16 位数据宽度模式
0 = 8 位数据宽度模式

bit 5-4　**EMB1:EMB0**：外部存储器总线配置位
11 = 单片机模式，禁止外部总线
10 = 扩展单片机模式，外部总线的地址宽度是 12 位
01 = 扩展单片机模式，外部总线的地址宽度是 16 位
00 = 扩展单片机模式，外部总线的地址宽度是 20 位

bit 3　**EASHFT**：外部地址总线移位使能位
1 = 地址移位使能，外部地址总线被移位以从 000000h 开始
0 = 地址移位禁止，外部地址总线反映 PC 值

bit 2-0　**未用**：读为 0

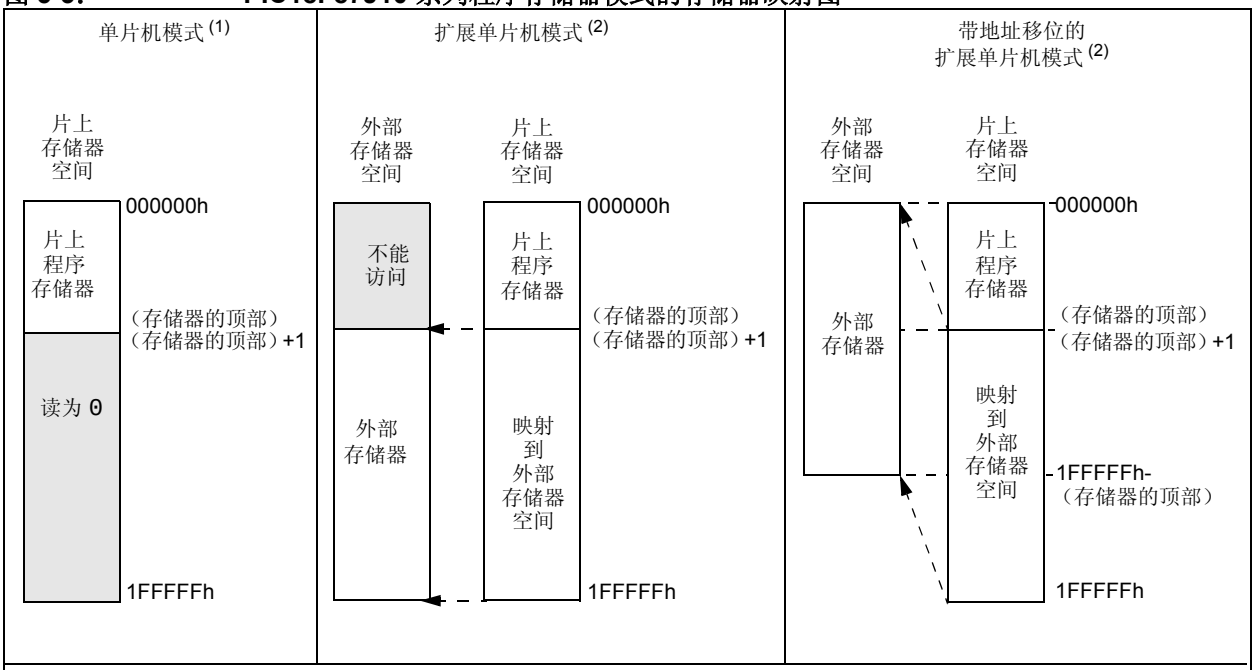
图注：			
R = 可读位	WO = 一次写入位	U = 未用位，读为 0	
-n = 擦除后的值	1 = 置 1	0 = 清零	x = 未知

5.1.4 扩展单片机模式和地址移位

默认情况下，处于扩展单片机模式的器件将指向外部存储器空间范围内的地址的程序计数器值直接放到外部地址总线上。实际上，这意味着外部存储器器件中低于片上存储器顶部的地址都不能被访问。

为了避免这种情况，扩展单片机模式使用了一个地址移位功能以使能自动的地址转换。在此模式下，放在外部总线上的地址值减去片上程序存储器的大小并重新映射到从 0000h 开始的外部存储器地址。这样就实现了对外部存储器器件的存储器空间的完全使用。

图 5-3： PIC18F87J10 系列程序存储器模式的存储器映射图



图注： （存储器的顶部）代表片上程序存储器空间的上边界（有关器件特定参数，见图 5-1）。阴影区域代表未用或不能访问的区域，取决于不同模式。

注 1： 此模式是 64 引脚器件惟一可用的模式，也是 80 引脚器件的默认模式。

2： 这些模式仅在 80 引脚器件上可用。

表 5-2： PIC18F8XJ10/8XJ15 各种程序存储器模式下的存储器访问

工作模式	内部程序存储器			外部程序存储器		
	执行程序	表读	表写	执行程序	表读	表写
单片机	可以	可以	可以	不可以	不可以	不可以
扩展单片机	可以	可以	可以	可以	可以	可以

5.1.5 程序计数器

程序计数器（Program Counter，PC）指定要取出执行的指令地址。PC 为 21 位宽，并且包含在 3 个独立的 8 位寄存器中。其中的低字节称为 PCL 寄存器，该寄存器可读写。高字节，即 PCH 寄存器，包含 PC<15:8> 位，不可直接读写。可以通过 PCLATH 寄存器更新 PCH 寄存器。更高字节称为 PCU。该寄存器包含 PC<20:16> 位，也不可直接读写。可以通过 PCLATU 寄存器更新 PCU 寄存器。

通过任何写 PCL 的操作，PCLATH 和 PCLATU 的内容将被传送到程序计数器。同样，通过读 PCL 的操作，程序计数器的两个高字节将被传送到 PCLATH 和 PCLATU。这有助于计算 PC 的偏移量（见**第 5.1.8.1 节“计算 GOTO”**）。

PC 在程序存储器中按字节寻址。为防止 PC 偏离指令字，PCL 的最低有效位固定为 0 值。PC 在程序存储器中以 2 为增量对连续指令寻址。

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

5.1.6 返回地址堆栈

用于存放返回地址的堆栈允许发生最多 31 次程序调用和中断。当执行 CALL 或 RCALL 指令或响应中断时，PC 值被压入堆栈。当执行 RETURN、RETLW 或 RETFIE 指令（如果使能了扩展指令集，则还包括 ADDULNK 和 SUBULNK 指令）时，PC 值从堆栈弹出。PCLATU 和 PCLATH 不受任何 RETURN 或 CALL 指令的影响 。

31 个字的堆栈操作通过 21 位的 RAM 和一个 5 位的堆栈指针 STKPTP 实现。堆栈既不占用程序存储器空间也不占用数据存储器空间。堆栈指针可以读写，并且通过栈顶特殊功能寄存器可以读写栈顶地址。使用这些寄存器可以将数据压入堆栈，或将数据从堆栈弹出。

执行 CALL 类型的指令会引起进栈操作；堆栈指针首先加 1，并且将 PC 的内容写入堆栈指针指向的单元（PC 已经指向 CALL 后的指令）。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被传送给 PC，然后堆栈指针减 1。

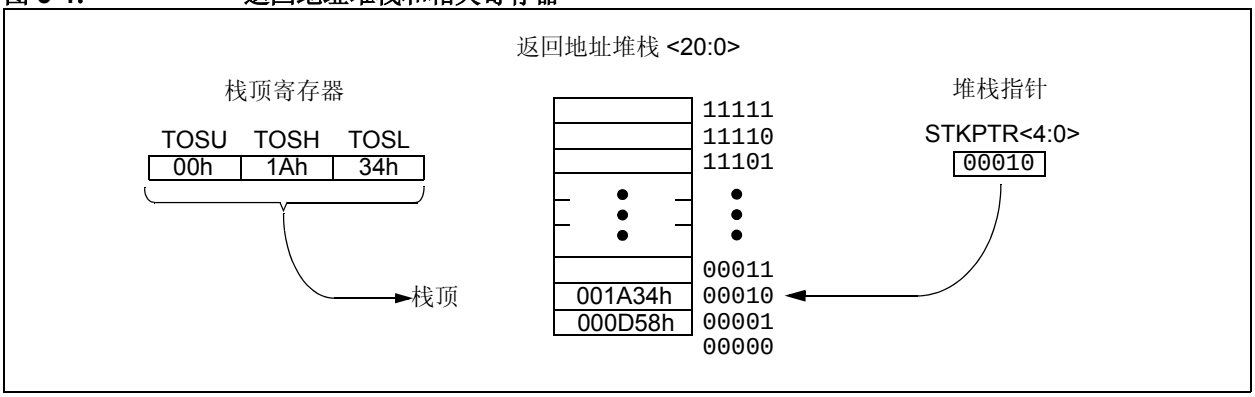
所有复位后，堆栈指针均会初始化为 “00000”。堆栈指针值 “00000” 不与任何 RAM 单元相关连，它仅仅是一个复位值。状态位表明堆栈是满，上溢还是下溢。

5.1.6.1 访问栈顶

只有返回地址堆栈的栈顶（TOS）是可读写的。一组 3 个寄存器 TOSU:TOSH:TOSL 保存 STKPTR 寄存器（图 5-4）所指向的堆栈单元的内容。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。这些值可以被置入用户定义的软件堆栈。返回时，软件可以将这些值返回到 TOSU:TOSH:TOSL 并执行返回。

为防止意外的堆栈破坏，访问堆栈时用户必须禁止全局中断允许位。

图 5-4： 返回地址堆栈和相关寄存器



5.1.6.2 返回堆栈指针 （STKPTR）

STKPTR 寄存器（寄存器 5-2）包含堆栈指针值、STKFUL（堆栈满）状态位和 STKUNF（堆栈下溢）状态位。堆栈指针值是从 0 到 31。堆栈压入前，堆栈指针加 1；堆栈弹出值后，堆栈指针减 1。复位时，堆栈指针值是 0。用户可以读写堆栈指针的值。实时操作系统（Real-Time Operating System，RTOS）可以利用此特性维护返回地址堆栈。

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。只能通过软件或 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。（有关器件配置位的介绍，请参见**第 23.1 节“配置位”**。）如果 STVREN 位已经置 1（默认值），第 31 次进栈将把 （PC+2）值压入堆栈，将 STKFUL 位置 1 并复位器件。 STKFUL 位将保持置 1，而堆栈指针将被置为 0。

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。任何其他进栈都不会覆盖第 31 次的进栈值并且 STKPTR 将保持为 31。

当堆栈弹出次数足够卸载堆栈时，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：	下溢引起的向PC返回零值会将程序指向复位向量，可以用复位向量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。

5.1.6.3 PUSH 和 POP 指令

因为栈顶可以读写，所以能够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，它们允许在软件控制下对 TOS 进行操作。可以修改 TOSU、TOSH 和 TOSL，将数据或返回地址放到堆栈中。

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 指令通过将栈指针减 1 来清除当前的 TOS 值。然后前一个进栈值就成为 TOS 值了。

寄存器 5-2： 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

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 **未用：**读为 0

bit 4-0 **SP4:SP0：**堆栈指针单元位

注 1：通过用户软件或 POR 清零 bit 7 和 bit 6。

图注：			
R = 可读位	W = 可写位	U = 未用位	C = 只可清零位
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

5.1.6.4 堆栈满和下溢复位

通过将配置寄存器 1L 中的 STVREN 位置 1 可以在出现堆栈满和堆栈下溢状态时使能器件复位。当STVREN位置 1 时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，然后使器件复位。当 STVREN 位清零时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，但不会使器件复位。通过用户软件或上电复位清零 STKFUL 或 STKUNF 位。

5.1.7 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，从而为中断提供“快速返回”选项。此堆栈深度仅为 1，并且不可读写。当处理器向量用于中断时，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 和 FAST 指令从中断返回，这些寄存器中的值会被装回工作寄存器。

如果同时使能了低优先级中断和高优先级中断，堆栈寄存器无法可靠地用于返回低优先级中断。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的堆栈寄存器值将被覆盖。在这些情况下，用户必须在低优先级中断期间用软件保存关键寄存器。

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，则快速寄存器堆栈可在子程序调用结束后用于恢复 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 5-1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 5-1： 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
```

5.1.8 程序存储器中的查找表

可能在有些编程情形下需要在程序存储器中创建数据结构和查找表。对于 PIC18 器件而言，有两种方法可以实现查找表：

- 计算 GOTO
- 表读

5.1.8.1 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 5-2 给出了一个示例。

使用 ADDWF PCL 指令和一组 RETLW nn 指令可以组成一个查找表。在调用该表前，会先将查找表中值的偏移量装入W寄存器。被调用子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，它将向调用函数返回值 “nn”。

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，并且需要返回地址堆栈还有空闲空间。

例 5-2： 使用偏移量计算 GOTO

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

5.1.8.2 表读

有一种更好的在程序存储器中存储数据的方法，可以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 6.1 节 “表读与表写”**中进一步讨论表读操作。

5.2 PIC18 指令周期

5.2.1 时钟机制

单片机时钟输入信号，无论来自内部或外部时钟源，都将在单片机内被四分频以产生四个互不重叠的正交时钟，即 Q1、Q2、Q3 和 Q4。单片机工作时，程序计数器在每个 Q1 加 1，并在 Q4 从程序存储器取指令并将指令锁存到指令寄存器中。指令的译码和执行在下一个Q1到Q4中完成。图 5-5所示为时钟和指令执行流程图。

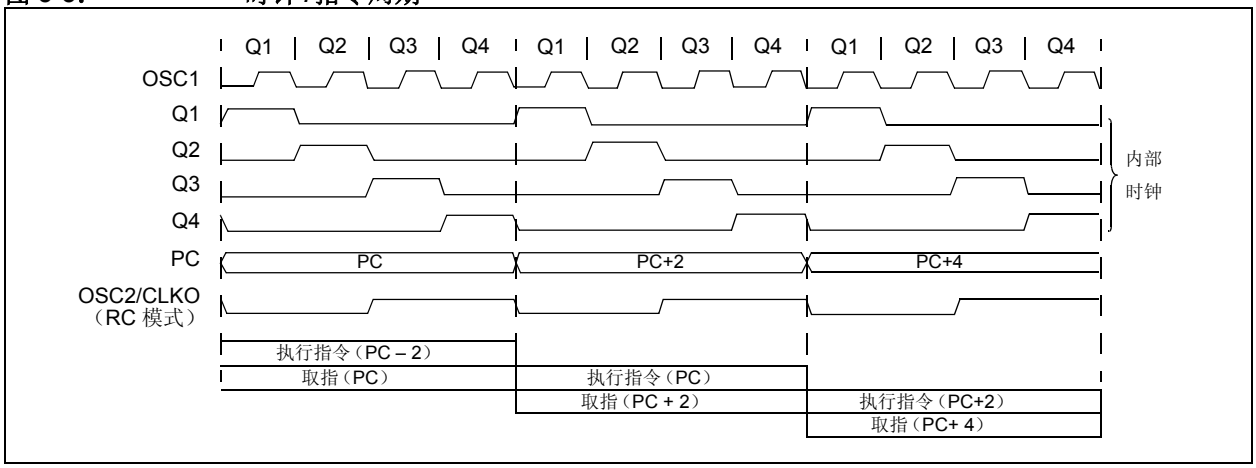
5.2.2 指令流 / 流水线

指令周期由 Q1 到 Q4 四个 Q 周期组成。取指和执行指令是流水执行的，用一个指令周期来取指，而用另一个指令周期译码和执行指令。但由于是流水线操作，所以每条指令的等效执行时间都是一个指令周期。如果某条指令改变了程序计数器（如，GOTO），则需要两个指令周期才能完成这条指令 （见例 5-3）。

在 Q1 周期开始取指操作，程序计数器 （PC）加 1。

指令的执行过程：在 Q1 周期，将所取指令锁存到指令寄存器 （Instruction Register， 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，写操作发生在 Q4 周期 （写目标）。

图 5-5： 时钟 /指令周期



例 5-3： 指令流水线流程图

	TCY0	TCY1	TCY2	TCY3	TCY4	TCY5
1. MOVLW 55h	取指 1	执行 1				
2. MOVWF PORTB		取指 2	执行 2			
3. BRA SUB_1			取指 3	执行 3		
4. BSF PORTA, BIT3 （强制 NOP）				取指 4	清除 （NOP）	
5. 在 SUB_1 地址处的指令					取指 SUB_1	执行 SUB_1

除程序转移指令外，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。

5.2.3 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偶地址的程序存储器单元中 （LSB = 0）。要保持与指令边界对齐，PC 以 2 为单位递增，并且 LSB 总是读为 0 （见**第 5.1.5 节 “程序计数器”**）。

图 5-6 给出了指令字如何存储在程序寄存器中的示例。

CALL 和 GOTO 指令在指令中嵌入了程序存储器的绝对地址。指令总是存储在字范围内，因而指令所包含的地址为字地址。字地址会写入 PC<20:1>，后者则在程序存储器中访问所需的字节地址。图 5-6 中的指令 2 说明了指令“GOTO 0006h”在程序存储器中是如何进行编码的。程序转移指令也采取同样的方式对相对地址偏移量进行编码。在转移指令中的偏移值代表单字指令数，PC 将以此作为偏移量。**第 24.0 节“指令集综述”**提供了指令集的更多详情。

图 5-6： 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

5.2.4 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，这些指令的第二个字总是将“1111”作为 4 个最高有效位；其余 12 位是立即数，通常是数据存储器地址。

使用指令中 4 个最高有效位中的 “1111”可以指定一种特殊形式的 NOP。如果在第一个字之后以正确的顺序立即执行指令，则指令第二个字中的数据将被该指令序列访问并使用。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，则其效果将相当于执行 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 5-4 显示了此操作的工作方式。

注： 有关扩展指令集中双字指令的信息，请参见**第 5.5 节“程序存储器和扩展指令集”**。

例 5-4： 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

5.3 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更改。更多信息，请参见**第 5.6 节“数据存储器和扩展指令集”**。

PIC18 器件中的数据存储器是用静态 RAM 实现的。在数据存储器中，每个寄存器有 12 位地址，数据存储器可达 4096 个字节。存储器空间被分为 16 个存储区，每个存储区包含 256 字节。 PIC18FX5J10/X5J15/X6J10 器件，有 64 KB 的程序存储器，实现了 8 个完整的存储区，总共 2048 字节。 PIC18FX6J15 和 PIC18FX7J10 器件，有 96 或 128 KB 的程序存储器，实现了所有可用的存储区并为用户提供 3936 字节的数据存储器。如图 5-7 和图 5-8 所示为器件的数据存储器构成。

数据存储器由特殊功能寄存器（Special Function Register， SFR）和通用寄存器（General Purpose Register，GPR）组成。SFR 用于单片机和外围功能的控制和状态显示，GPR 则用于在用户应用程序中存储数据和临时存储操作的中间结果。任何未用单元均读为 0。

此指令集和架构支持跨所有存储区的操作。通过直接、间接或索引寻址模式可以访问整个数据存储器。本节稍后将讨论寻址模式。

为了确保能在一个周期内快速访问常用寄存器（所选的 SFR 和 GPR）， PIC18 器件实现了一个快速操作存储区。这是一个 256 字节的存储器空间，能够在不使用 BSR 的情况下提供对 SFR 和 GPR Bank 0 的低地址部分的快速访问。**第 5.3.2 节 “快速操作存储区”**提供了对快速操作 RAM 的详细描述。

5.3.1 存储区选择寄存器

存储容量较大的数据存储器需要有效的寻址机制，以便尽可能对所有地址进行快速访问。理想状况下，这意味着不必为每次读写操作提供完整地址。对于 PIC18 器件，这是由 RAM 存储机制实现的。它将存储器空间分为 16 个相连的存储区，每个存储区包含 256 字节。根据指令，每个单元可通过其完整的 12 位地址或 8 位低位地址和 4 位存储区指针直接被寻址。

PIC18 指令集中的大多数指令使用存储区指针，称为存储区选择寄存器 （Bank Select Register， BSR）。此 SFR 保存着单元地址的 4 个最高有效位；指令本身包含了 8 个最低有效位。只使用 BSR 的低 4 位（BSR3:BSR0）；高 4 位不使用，它们总是读为 0 且不能被写入。通过使用 MOVLB 指令可以直接装载 BSR。

BSR 的值指明数据存储器中的存储区；指令中的 8 位指明在存储区中的位置，并且可被视作从该存储区下边界开始的偏移量。图 5-9 所示为 BSR 的值与数据存储器中存储区划分的关系。

因为至多 16 个寄存器可以共用同一个低位地址，所以用户必须始终小心以确保在执行数据读或写操作之前选择了正确的存储区。例如，在 BSR 为 0Fh 时，将程序数据写入地址为 F9h 的 8 位地址单元，将终止程序计数器的复位。

虽然可以选择任何存储区，但是只有那些实际使用的存储区才能被读取或写入。对未用存储区的写操作将被忽略，而读取未用存储区将返回 0。即使这样， STATUS 寄存器仍将受到影响，就像操作成功时一样。图 5-7 中的数据存储器映射图指出了可使用的存储区。

在 PIC18 核心指令集中，只有 MOVFF 指令完全地指定了源寄存器和目标寄存器的 12 位地址。当执行此指令时，完全忽略 BSR。所有其他指令只包括低位地址作为操作数，并且必须使用 BSR 或快速操作存储区来定位它们的目标寄存器。

图 5-7：　　PIC18FX5J10/X5J15/X6J10 器件的数据存储器映射图

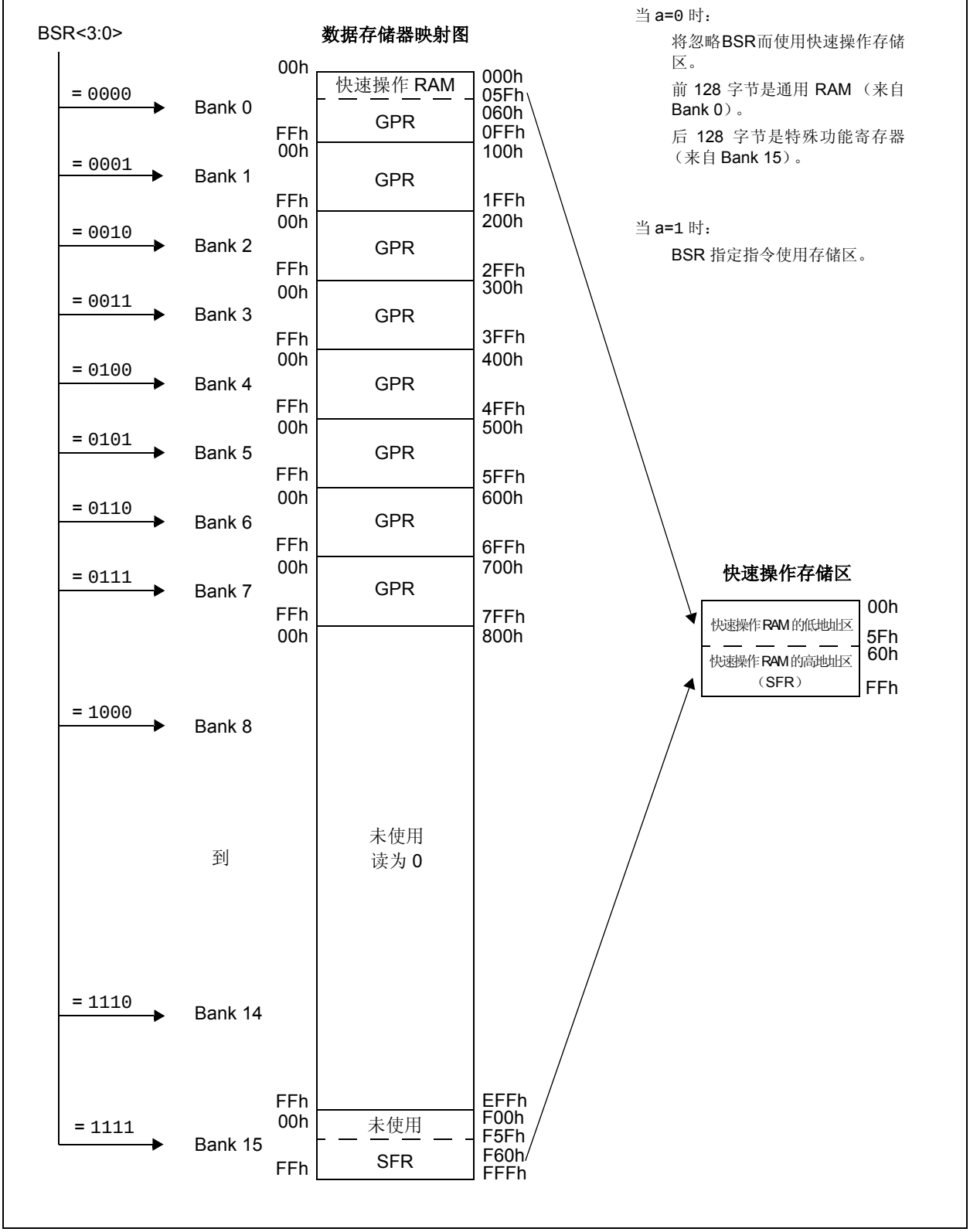


图 5-8： PIC18FX6J15/X7J10 器件的数据存储器映射图

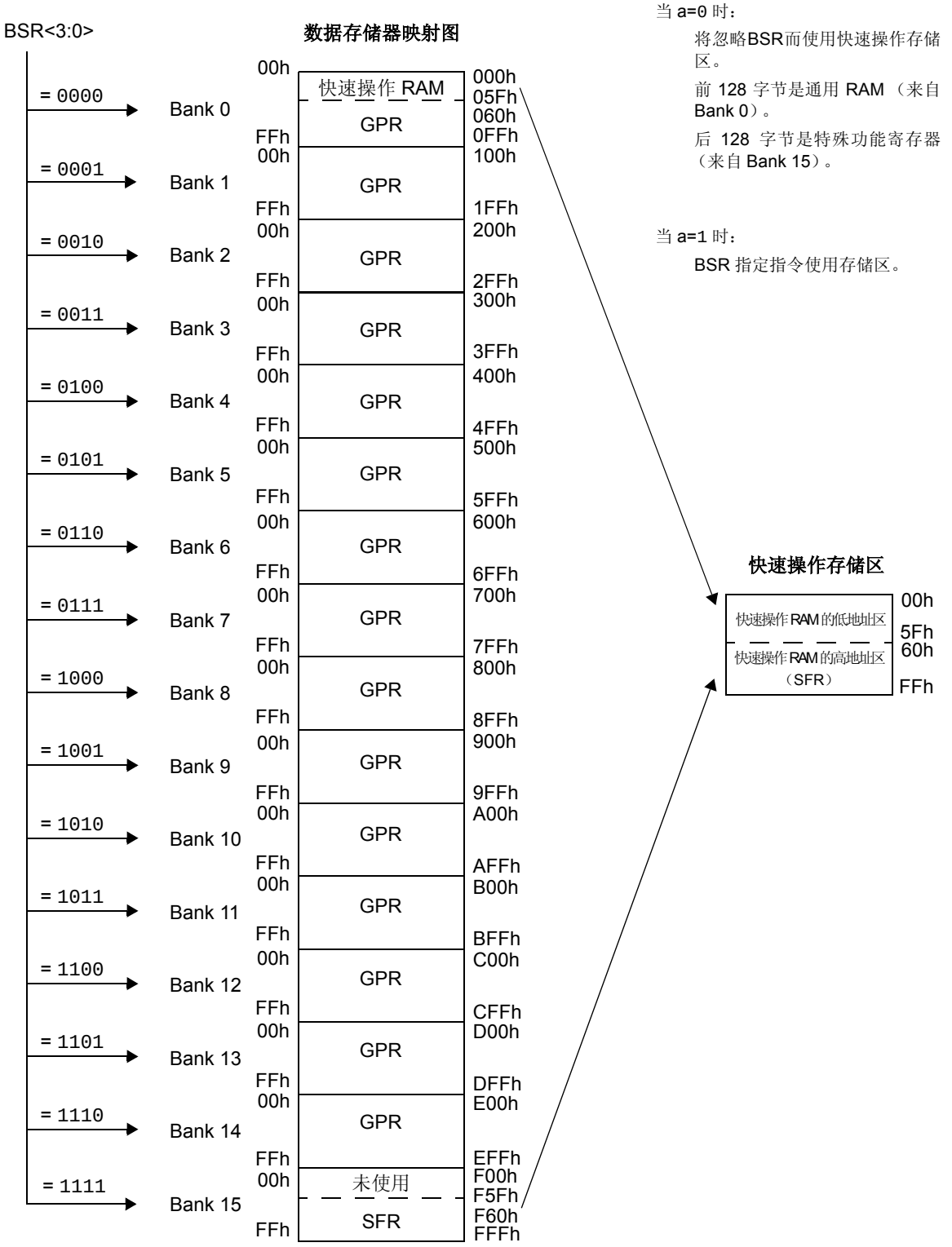
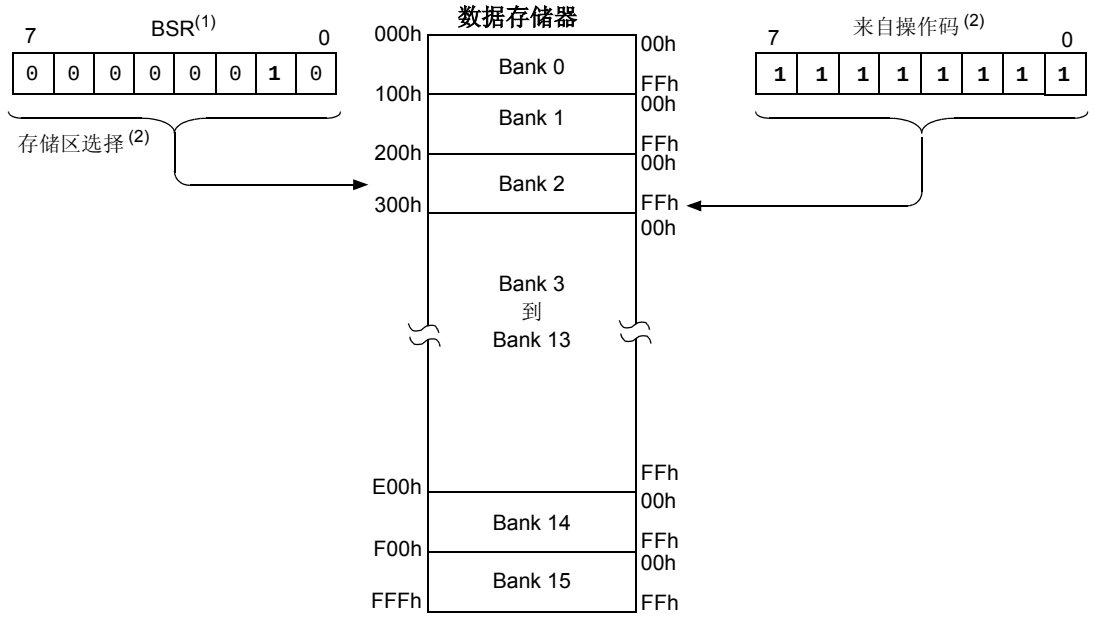


图 5-9：　　使用存储区选择寄存器 （直接寻址）



注　1： 可以用指令的快速操作 RAM 位强制跳过选定的存储区 （由 BSR<3:0> 指定）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。

5.3.2　　快速操作存储区

使用 BSR 和指令内嵌的 8 位地址可以使用户寻址整个数据存储器的范围，这还意味着用户必须总是确保选择了正确的存储区。否则，可能从错误的单元读取数据或将数据写入错误的单元。如果某个操作的预定目标寄存器是 GPR，但是却写入了 SFR，结果可能是灾难性的。但是在每次向数据存储器进行读或写操作时确认和 / 或更改 BSR 会降低代码的执行效率。

为了连续访问最常用的数据存储器单元，必须将数据存储器配置为快速操作存储区，这样可以允许用户访问被映射的存储区而不需要指定 BSR。快速操作存储区由存储器的 Bank 0 中的前 96 个字节（00h-5Fh）和存储器的 Bank 15 中的后 160 个字节（60h-FFh）组成。低地址的那一半称为 “快速操作 RAM”，由 GPR 组成。地址较高的部分被映射为器件的 SFR。这两个区域连续地映射到快速操作存储区，并可以通过 8 位地址以线性方式寻址 （图 5-7）。

通过执行包括快速操作 RAM 位（指令中的参数 “a”）的 PIC18 内核指令使用快速操作存储区。当 “a” 等于 “1” 时，指令使用 BSR 并将包含在操作码中的 8 位地址作为数据存储器的地址。然而当 “a” 等于 “0” 时，强制指令使用快速操作存储区映射；完全忽略 BSR 的当前值。

使用这种 “强制的” 寻址可以在一个周期内让指令对数据地址进行操作，而无需首先更新 BSR。对于 60h 及以上的 8 位地址，这意味着用户可以更高效地对 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访问的数据值（如直接计算结果或常用程序变量）。快速操作 RAM 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有略微不同。**第 5.6.3 节 “在立即数变址寻址模式下映射快速操作存储区”** 中对此有更详细的讨论。

5.3.3　　通用数据寄存器

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，所有指令均可访问它。 GPR 从 Bank0 的底部 （地址 000h）开始向上直到 SFR 区域的底部。上电复位不会将 GPR 初始化，并且所有其他复位也不会改变其内容。

5.3.4 特殊功能寄存器

特殊功能寄存器（Special Function Register，SFR）是CPU和外设模块用来控制所需的器件操作的寄存器。这类寄存器是由静态RAM实现的。SFR起始于数据存储器的顶部（FFFh）并且向下扩展到Bank 15的上半部分（F60h到FFFh）。表5-3和表5-4列出了这些寄存器。

SFR可分为两类：一类与“内核”器件功能（ALU、复位和中断）有关，另一类与外设功能有关。复位和中断寄存器在它们各自的章节中说明，而ALU的STATUS寄存器在本节稍后说明。与外设功能部件的工作相关的寄存器在该外设的章节说明。

SFR常分布在外设中，用来控制外设的功能。未使用的SFR单元是不可使用的并且读为全0。

表5-3： PIC18F87J10系列器件的特殊功能寄存器映射图

地址	名称	地址	名称	地址	名称	地址	名称	地址	名称
FFFh	TOSU	FDFh	INDF2(1)	FBFh	CCPR1H	F9Fh	IPR1	F7Fh	SPBRGH1
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	CCPR1L	F9Eh	PIR1	F7Eh	BAUDCON1
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCP1CON	F9Dh	PIE1	F7Dh	SPBRGH2
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR2H	F9Ch	MEMCON(3)	F7Ch	BAUDCON2
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCPR2L	F9Bh	OSCTUNE	F7Bh	—(2)
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	TRISJ(3)	F7Ah	—(2)
FF9h	PCL	FD9h	FSR2L	FB9h	CCPR3H	F99h	TRISH(3)	F79h	ECCP1DEL
FF8h	TBLPTRU	FD8h	STATUS	FB8h	CCPR3L	F98h	TRISG	F78h	TMR4
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	CCP3CON	F97h	TRISF	F77h	PR4
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS	F96h	TRISE	F76h	T4CON
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD	F75h	CCPR4H
FF4h	PRODH	FD4h	—(2)	FB4h	CMCON	F94h	TRISC	F74h	CCPR4L
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	CCP4CON
FF2h	INTCON	FD2h	—(2)	FB2h	TMR3L	F92h	TRISA	F72h	CCPR5H
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ(3)	F71h	CCPR5L
FF0h	INTCON3	FD0h	RCON	FB0h	PSPCON	F90h	LATH(3)	F70h	CCP5CON
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG	F6Fh	SPBRG2
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF	F6Eh	RCREG2
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE	F6Dh	TXREG2
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD	F6Ch	TXSTA2
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC	F6Bh	RCSTA2
FEAh	FSR0H	FCAh	T2CON	FAAh	—(2)	F8Ah	LATB	F6Ah	ECCP3AS
FE9h	FSR0L	FC9h	SSP1BUF	FA9h	—(2)	F89h	LATA	F69h	ECCP3DEL
FE8h	WREG	FC8h	SSP1ADD	FA8h	—(2)	F88h	PORTJ(3)	F68h	ECCP2AS
FE7h	INDF1(1)	FC7h	SSP1STAT	FA7h	EECON2	F87h	PORTH(3)	F67h	ECCP2DEL
FE6h	POSTINC1(1)	FC6h	SSP1CON1	FA6h	EECON1	F86h	PORTG	F66h	SSP2BUF
FE5h	POSTDEC1(1)	FC5h	SSP1CON2	FA5h	IPR3	F85h	PORTF	F65h	SSP2ADD
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE	F64h	SSP2STAT
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD	F63h	SSP2CON1
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	SSP2CON2
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	—(2)
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	—(2)

注 **1：** 这不是物理寄存器。
2： 不可用的寄存器读为0。
3： 此寄存器在64引脚器件上不存在。

表 5-4： 寄存器汇总 （PIC18F87J10 系列）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 或 BOR 时的值	详情请见：（页）
TOSU	—	—	—	栈顶最高字节 （TOS<20:16>）					---0 0000	49, 59
TOSH	栈顶高字节 （TOS<15:8>）								0000 0000	49, 59
TOSL	栈顶低字节 （TOS<7:0>）								0000 0000	49, 59
STKPTR	STKFUL	STKUNF	—	SP4	SP3	SP2	SP1	SP0	00-0 0000	49, 60
PCLATU	—	—	bit 21(1)	PC<20:16> 的保持寄存器					---0 0000	49, 59
PCLATH	PC<15:8> 的保持寄存器								0000 0000	49, 59
PCL	PC 低字节 （PC<7:0>）								0000 0000	49, 59
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 （TBLPTR<20:16>）					--00 0000	49, 84
TBLPTRH	程序存储器表指针高字节 （TBLPTR<15:8>）								0000 0000	49, 84
TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								0000 0000	49, 84
TABLAT	程序存储器表锁存器								0000 0000	49, 84
PRODH	乘积寄存器高字节								xxxx xxxx	49, 103
PRODL	乘积寄存器低字节								xxxx xxxx	49, 103
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	49, 107
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	1111 1111	49, 108
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	1100 0000	49, 109
INDF0	使用 FSR0 的内容寻址数据存储器， FSR0 的值不改变 （不是物理寄存器）								N/A	49, 75
POSTINC0	使用 FSR0 的内容寻址数据存储器， FSR0 的值后增 （不是物理寄存器）								N/A	49, 76
POSTDEC0	使用 FSR0 的内容寻址数据存储器， FSR0 的值后减 （不是物理寄存器）								N/A	49, 76
PREINC0	使用 FSR0 的内容寻址数据存储器， FSR0 的值预增 （不是物理寄存器）								N/A	49, 76
PLUSW0	使用 FSR0 的内容寻址数据存储器， FSR0 的值预增 （不是物理寄存器）， FSR0 的偏移量由 W 寄存器提供								N/A	49, 76
FSR0H	—	—	—	—	间接数据存储器地址指针 0 的高字节				---- xxxx	49, 75
FSR0L	间接数据存储器地址指针 0 的低字节								xxxx xxxx	49, 75
WREG	工作寄存器								xxxx xxxx	49
INDF1	使用 FSR1 的内容寻址数据存储器， FSR1 的值不改变 （不是物理寄存器）								N/A	49, 75
POSTINC1	使用 FSR1 的内容寻址数据存储器， FSR1 的值后增 （不是物理寄存器）								N/A	49, 76
POSTDEC1	使用 FSR1 的内容寻址数据存储器， FSR1 的值后减 （不是物理寄存器）								N/A	49, 76
PREINC1	使用 FSR1 的内容寻址数据存储器， FSR1 的值预增 （不是物理寄存器）								N/A	49, 76
PLUSW1	使用 FSR1 的内容寻址数据存储器， FSR1 的值预增 （不是物理寄存器）， FSR1 的偏移量由 W 寄存器提供								N/A	49, 76
FSR1H	—	—	—	—	间接数据存储器地址指针 1 的高字节				---- xxxx	49, 75
FSR1L	间接数据存储器地址指针 1 的低字节								xxxx xxxx	49, 75
BSR	—	—	—	—	存储区选择寄存器				---- 0000	49, 64
INDF2	使用 FSR2 的内容寻址数据存储器， FSR2 的值不改变 （不是物理寄存器）								N/A	50, 75
POSTINC2	使用 FSR2 的内容寻址数据存储器， FSR2 的值后增 （不是物理寄存器）								N/A	50, 76
POSTDEC2	使用 FSR2 的内容寻址数据存储器， FSR2 的值后减 （不是物理寄存器）								N/A	50, 76
PREINC2	使用 FSR2 的内容寻址数据存储器， FSR2 的值预增 （不是物理寄存器）								N/A	50, 76
PLUSW2	使用 FSR2 的内容寻址数据存储器， FSR2 的值预增 （不是物理寄存器）， FSR2 的偏移量由 W 寄存器提供								N/A	50, 76
FSR2H	—	—	—	—	间接数据存储器地址指针 2 的高字节				---- xxxx	50, 75
FSR2L	间接数据存储器地址指针 2 的低字节								xxxx xxxx	50, 75
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	50, 73

图注： x = 未知， u = 不变， — = 未用， q = 取值视情况而定

注 1： PC 的 Bit 21 仅在串行编程模式下可用。

2： 这些位和 / 或寄存器仅在 80 引脚器件上可用。在其他器件上，它们未用并读为 0。所示为 80 引脚器件的复位值。

3： 在 64 引脚器件中未用此寄存器及其位。对于处于单片机模式的 80 引脚器件，这些位不可写且读为 0。

4： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

5： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

表 5-4： 寄存器汇总 （PIC18F87J10 系列） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 或 BOR 时的值	详情请见：（页）
TMR0H	Timer0 寄存器的高字节								0000 0000	50, 149
TMR0L	Timer0 寄存器的低字节								xxxx xxxx	50, 149
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	50, 147
OSCCON	IDLEN	—	—	—	OSTS(5)	—	SCS1	SCS0	0--- q-00	29, 50
WDTCON	—	—	—	—	—	—	—	SWDTEN	--- ---0	50, 283
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	0--1 1100	44, 50, 119
TMR1H	Timer1 寄存器的高字节								xxxx xxxx	50, 155
TMR1L	Timer1 寄存器的低字节								xxxx xxxx	50, 155
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	0000 0000	50, 151
TMR2	Timer2 寄存器								0000 0000	50, 158
PR2	Timer2 周期寄存器								1111 1111	50, 158
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	50, 157
SSP1BUF	MSSP1 接收缓冲器 / 发送寄存器								xxxx xxxx	50, 184, 199
SSP1ADD	MSSP1 地址寄存器 （I²C 从动模式）， MSSP1 波特率重载寄存器 （I²C 主控模式）								0000 0000	50, 199
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	50, 190, 200
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	50, 191, 201
SSP1CON2	GCEN	ACKSTAT	ACKDT/ ADMSK5	ACKEN/ ADMSK4	RCEN/ ADMSK3	PEN/ ADMSK2	RSEN/ ADMSK1	SEN	0000 0000	50, 202
ADRESH	A/D 结果寄存器的高字节								xxxx xxxx	50, 266
ADRESL	A/D 结果寄存器的低字节								xxxx xxxx	50, 266
ADCON0	ADCAL	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{DONE}$	ADON	0-00 0000	50, 258
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	--00 0000	50, 259
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	50, 260
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								xxxx xxxx	51, 188
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								xxxx xxxx	51, 188
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	51, 173
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								xxxx xxxx	51, 188
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								xxxx xxxx	51, 188
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	0000 0000	51, 173
CCPR3H	捕捉 / 比较 /PWM 寄存器 3 的高字节								xxxx xxxx	51, 188
CCPR3L	捕捉 / 比较 /PWM 寄存器 3 的低字节								xxxx xxxx	51, 188
CCP3CON	P3M1	P3M0	DC3B1	DC3B0	CCP3M3	CCP3M2	CCP3M1	CCP3M0	0000 0000	51, 173
ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1(2)	PSS1BD0(2)	0000 0000	51, 185
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	51, 273
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0111	51, 268
TMR3H	Timer3 寄存器的高字节								xxxx xxxx	51, 161
TMR3L	Timer3 寄存器的低字节								xxxx xxxx	51, 161
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	0000 0000	51, 159
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	0000 ----	51, 145
SPBRG1	EUSART1 波特率发生器寄存器的低字节								0000 0000	51, 240
RCREG1	EUSART1 接收寄存器								0000 0000	51, 248, 249

图注： x = 未知， u = 不变， — = 未用， q = 取值视情况而定

注 1： PC 的 Bit 21 仅在串行编程模式下可用。

2： 这些位和 / 或寄存器仅在 80 引脚器件上可用。在其他器件上，它们未用并读为 0。所示为 80 引脚器件的复位值。

3： 在 64 引脚器件中未用此寄存器及其位。对于处于单片机模式的 80 引脚器件，这些位不可写且读为 0。

4： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

5： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

表 5-4：　　寄存器汇总 （PIC18F87J10 系列）（续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 或 BOR 时的值	详情请见：（页）
TXREG1	EUSART1 发送寄存器								xxxx xxxx	51, 246, 247
TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	51, 237
RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	51, 238
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	1111 1111	51, 118
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	0000 0000	51, 112
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	0000 0000	51, 115
IPR2	OSCFIP	CMIP	—	—	BCL1IP	—	TMR3IP	CCP2IP	11-- 1-11	51, 117
PIR2	OSCFIF	CMIF	—	—	BCL1IF	—	TMR3IF	CCP2IF	00-- 0-00	51, 111
PIE2	OSCFIE	CMIE	—	—	BCL1IE	—	TMR3IE	CCP2IE	00-- 0-00	51, 114
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	1111 1111	51, 116
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	0000 0000	51, 110
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IF	CCP1IE	TMR2IE	TMR1IE	0000 0000	51, 113
MEMCON(3)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0	0-00 --00	51, 92
OSCTUNE	—	PLLEN(4)	—	—	—	—	—	—	-0-- ----	29, 51
TRISJ(2)	TRISJ7	TRISJ6	TRISJ5	TRISJ4	TRISJ3	TRISJ2	TRISJ1	TRISJ0	1111 1111	52, 143
TRISH(2)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	1111 1111	52, 141
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	---1 1111	52, 139
TRISF	TRISF7	TRISF6	TRISF5	+TRISF4	TRISF3	TRISF2	TRISF1	—	1111 111-	52, 137
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	1111 1111	52, 135
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	1111 1111	52, 132
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	1111 1111	52, 129
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	1111 1111	52, 126
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	--11 1111	52, 123
LATJ(2)	LATJ7	LATJ6	LATJ5	LATJ4	LATJ3	LATJ2	LATJ1	LATJ0	xxxx xxxx	52, 143
LATH(2)	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0	xxxx xxxx	52, 141
LATG	—	—	—	LATG4	LATG3	LATG2	LATG1	LATG0	---x xxxx	52, 139
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—	xxxx xxx-	52, 137
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	xxxx xxxx	52, 135
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx xxxx	52, 132
LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx xxxx	52, 129
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx xxxx	52, 126
LATA	—	—	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	--xx xxxx	52, 123
PORTJ(2)	RJ7	RJ6	RJ5	RJ4	RJ3	RJ2	RJ1	RJ0	xxxx xxxx	52, 143
PORTH(2)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	0000 xxxx	52, 141
PORTG	RDPU	REPU	RJPU(2)	RG4	RG3	RG2	RG1	RG0	111x xxxx	52, 139
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	x000 000-	52, 137
PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	xxxx xxxx	52, 135
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx xxxx	52, 132
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx xxxx	52, 129
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx xxxx	52, 126
PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	--0x 0000	52, 123

图注： x = 未知， u = 不变， — = 未用， q = 取值视情况而定

注 1： PC 的 Bit 21 仅在串行编程模式下可用。

2： 这些位和 / 或寄存器仅在 80 引脚器件上可用。在其他器件上，它们未用并读为 0。所示为 80 引脚器件的复位值。

3： 在 64 引脚器件中未用此寄存器及其位。对于处于单片机模式的 80 引脚器件，这些位不可写且读为 0。

4： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

5： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

表 5-4：　　寄存器汇总 （PIC18F87J10 系列） （续）

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR 或 BOR 时的值	详情请见：（页）
SPBRGH1	EUSART1 波特率发生器寄存器的高字节								0000 0000	52, 240
BAUDCON1	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	52, 239
SPBRGH2	EUSART2 波特率发生器寄存器的高字节								0000 0000	52, 240
BAUDCON2	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	52, 239
ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0	0000 0000	53, 184
TMR4	Timer4 寄存器								0000 0000	53, 164
PR4	Timer4 周期寄存器								1111 1111	53, 164
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	-000 0000	53, 163
CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节								xxxx xxxx	53, 166
CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节								xxxx xxxx	53, 166
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	--00 0000	53, 165
CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节								xxxx xxxx	53, 166
CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节								xxxx xxxx	53, 166
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	--00 0000	53, 165
SPBRG2	EUSART2 波特率发生器寄存器的低字节								0000 0000	53, 240
RCREG2	EUSART2 接收寄存器								0000 0000	53, 248, 249
TXREG2	EUSART2 发送寄存器								0000 0000	53, 246, 247
TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	53, 237
RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	53, 238
ECCP3AS	ECCP3ASE	ECCP3AS2	ECCP3AS1	ECCP3AS0	PSS3AC1	PSS3AC0	PSS3BD1	PSS3BD0	0000 0000	53, 185
ECCP3DEL	P3RSEN	P3DC6	P3DC5	P3DC4	P3DC3	P3DC2	P3DC1	P3DC0	0000 0000	53, 184
ECCP2AS	ECCP2ASE	ECCP2AS2	ECCP2AS1	ECCP2AS0	PSS2AC1	PSS2AC0	PSS2BD1	PSS2BD0	0000 0000	53, 185
ECCP2DEL	P2RSEN	P2DC6	P2DC5	P2DC4	P2DC3	P2DC2	P2DC1	P2DC0	0000 0000	53, 184
SSP2BUF	MSSP2 接收缓冲器 / 发送寄存器								xxxx xxxx	53, 184, 199
SSP2ADD	MSSP2 地址寄存器 （I²C 从动模式）， MSSP2 波特率重载寄存器 （I²C 主控模式）								0000 0000	53, 199
SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	53, 184, 200
SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	53, 191, 201
SSP2CON2	GCEN	ACKSTAT	ACKDT/ ADMSK5	ACKEN/ ADMSK4	RCEN/ ADMSK3	PEN/ ADMSK2	RSEN/ ADMSK1	SEN	0000 0000	53, 202

图注：　x = 未知， u = 不变， — = 未用， q = 取值视情况而定

注　1： PC 的 Bit 21 仅在串行编程模式下可用。

2： 这些位和 / 或寄存器仅在 80 引脚器件上可用。在其他器件上，它们未用并读为 0。所示为 80 引脚器件的复位值。

3： 在 64 引脚器件中未用此寄存器及其位。对于处于单片机模式的 80 引脚器件，这些位不可写且读为 0。

4： 仅当选定了 ECPLL 或 HSPLL 振荡器模式时， PLLEN 位可用；否则，该位读为 0。

5： 使能双速启动时复位值为 0，禁止双速启动时复位值为 1。

5.3.5 STATUS 寄存器

如寄存器 5-3 所示，STATUS 寄存器包含 ALU 的算术运算状态。STATUS 寄存器和任何其他寄存器一样，可以作为任何指令的操作数。如果一条影响 Z、DC、C、OV 或 N 位的指令的目标寄存器是 STATUS 寄存器，则会禁止对这 5 位进行写操作。

根据器件逻辑，这些位会被置位或清零。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。然后 STATUS 寄存器将读出“000u u1uu”。因此，建议仅使用 BCF、BSF、SWAPF、MOVFF 和 MOVWF 指令来改变 STATUS 寄存器，因为这些指令不会影响 STATUS 寄存器中的 Z、C、DC、OV 或 N 位。

关于其他不影响任何状态位的指令，请参见表 24-2 和表 24-3 中的指令集综述。

注：	在减法运算中，C 和 DC 位分别作为 $\overline{\text{borrow}}$ 和 $\overline{\text{digit borrow}}$ 标志位。

寄存器 5-3：　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	C
bit 7							bit 0

bit 7-5　**未用：**读为 0

bit 4　**N：**负标志位
此位用于有符号数的算术运算（二进制补码）。它可以表示结果是否为负（ALU MSB= 1）。
1 = 结果为负
0 = 结果为正

bit 3　**OV：**溢出标志位
此位用于有符号数的算术运算（二进制补码）。表明结果超出了 7 位二进制数的范围，溢出导致了符号位（bit7）改变状态。
1 = 在本次运算中有符号数的算术运算发生溢出
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：**Digit carry/$\overline{\text{borrow}}$ 标志位
用于 ADDWF、ADDLW、SUBLW 和 SUBWF 指令
1 = 结果的第 4 个低位发生了进位
0 = 结果的第 4 个低位未发生进位

注：	对于 $\overline{\text{borrow}}$，极性是相反的。减法指令通过加上第二个操作数的二进制补码来实现。对于移位指令（RRF 和 RLF），此位值来自源寄存器的 bit 4 或 bit 3。

bit 0　**C：**carry/$\overline{\text{borrow}}$ 标志位
用于 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：	对于 $\overline{\text{borrow}}$，极性是相反的。减法指令通过加上第二个操作数的二进制补码来实现。对于移位指令（RRF 和 RLF），此位值来自源寄存器的高位或低位。

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置位	0 = 清零	x = 未知

5.4 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改变。更多信息，请参见**第 5.6 节“数据存储器和扩展指令集”**。

虽然只能用一种方法（即通过程序计数器）对程序存储器进行寻址，但是可以用几种方法来对数据存储器空间进行寻址。对于大多数指令，寻址模式是固定的。其他指令最多可以使用 3 种模式，这取决于使用哪些操作数以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，即使用立即数作为偏移量进行变址寻址的模式。**第 5.6.1 节 “使用立即数作为偏移量进行变址寻址”**中对其操作有更详细的讨论。

5.4.1 固有和立即数寻址

很多 PIC18 控制指令根本不需要任何参数，它们执行对器件有全局影响的操作或者隐式地操作某个寄存器。这种寻址模式称为固有寻址。例如 SLEEP、 RESET 和 DAW。

其他指令的工作方式与此相同但需要操作码中有其他显式的参数。由于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。例如 ADDLW 和 MOVLW，它们分别将立即数值加到或移动到 W 寄存器。其他例子包括 CALL 和 GOTO，这两条指令包括一个 20 位的程序存储器地址。

5.4.2 直接寻址

直接寻址在操作码自身内部指定操作的全部或部分源和 / 或目标地址。这些选项由指令附带的参数指定。

在 PIC18 内核指令集中，针对位的指令和针对字节的指令在默认情况下使用某些类型的直接寻址。所有这些指令都包括某个 8 位立即数地址作为它们的最低有效字节。此地址也指定了数据 RAM 的某个存储区中的一个寄存器地址 （**第 5.3.3 节 “通用数据寄存器”**）或快速操作存储区 （**第 5.3.2 节 “快速操作存储区”**）中的一个单元作为指令的数据源。

快速操作 RAM 位 “a” 决定地址的解析方式。当 “a” 为 “1” 时，BSR（**第 5.3.1 节 “存储区选择寄存器”**）的内容与该地址一起使用来确定完整的 12 位寄存器地址。当 “a” 为 “0” 时，该地址被解析为快速操作存储区中的一个寄存器的地址。使用快速操作 RAM 的寻址有时也称为直接强制寻址模式。

有些指令，比如 MOVFF，在它们的操作码中包括了完整的 12 位地址 （源或目标地址）。在这些情况下，完全忽略 BSR。

该操作的结果的目标寄存器由目标位 “d” 决定。当 “d” 为 “1” 时，结果存回源寄存器，覆盖其原来的内容。当 “d” 为 “0” 时，结果存入 W 寄存器。不带 “d” 参数的指令有一个隐含在指令中的目标寄存器，这些指令的目标寄存器是正在操作的目标寄存器或W寄存器。

5.4.3 间接寻址

间接寻址使用户无需在指令中给出一个固定的地址就能访问数据存储器中的某个单元。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而实现的。因为FSR本身作为特殊功能寄存器位于 RAM 中，所以可以在程序控制下直接对它们进行操作。这就使得FSR在数据存储器中实现诸如表和阵列之类的数据结构时非常有用。

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。INDF 允许使用其他值自动递增、递减或偏移指针，从而自动控制指针的值。这可以实现使用循环的高效代码，例如例 5-5 中清零整个 RAM 存储区的示例。它还允许用户在数据存储器中执行变址寻址和其他针对程序存储器堆栈指针的操作。

例 5-5： 使用间接寻址将 RAM （BANK 1）清零的方法

```
        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
```

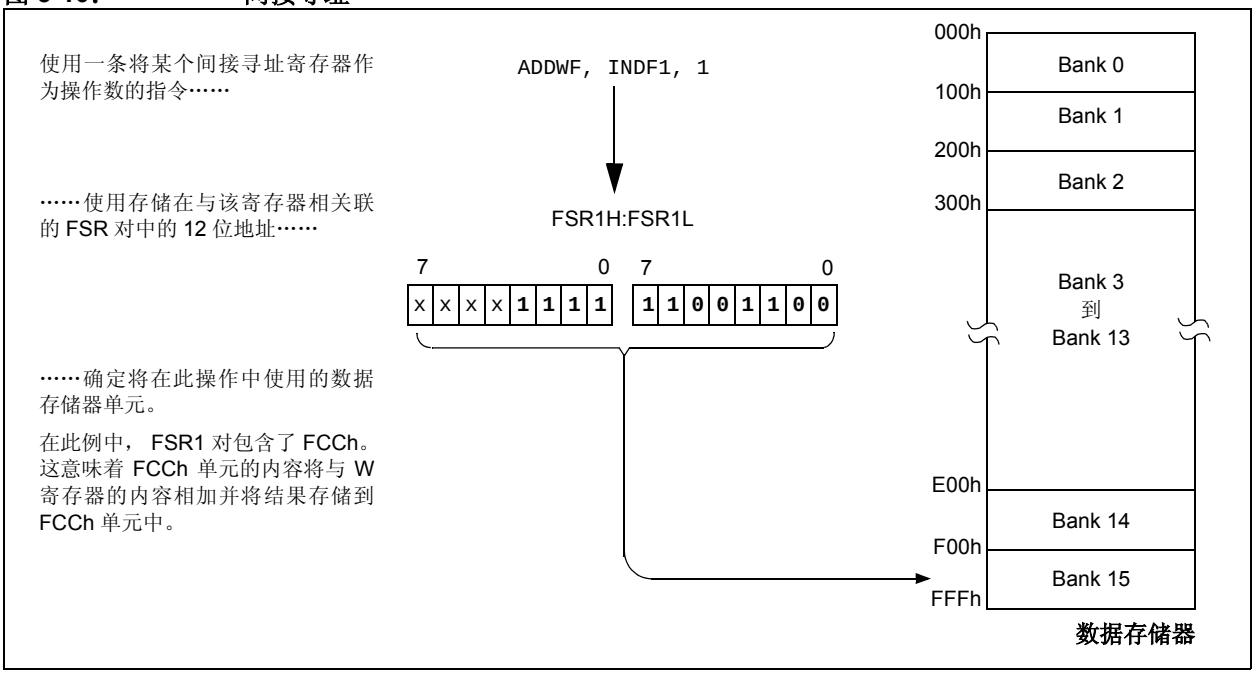
5.4.3.1 FSR 寄存器和 INDF 操作数

间接寻址的核心是三组寄存器：FSR0、FSR1和FSR2。每一组代表一对8位寄存器，FSRnH和FSRnL。FSRnH寄存器的高 4 位未使用，所以每对 FSR 保存一个 12 位值。它代表一个可以以线性方式对整个数据存储区范围进行寻址的值。因而，FSR 寄存器对可以作为指向数据存储器单元的指针。

间接寻址是通过一组间接指针操作数（从 INDF0 到 INDF2）完成的。这些操作数可被视作“虚拟”寄存器：它们被映射到 SFR 空间，但并非物理实现。对某个特定的 INDF 寄存器进行读取或写入实际上是访问其对应的 FSR 寄存器对。例如，从 INDF1 读取数据，读取的是由 FSR1H:FSR1L 指向的地址单元的数据。使用INDF寄存器作为操作数的指令实际上是使用INDF寄存器对应的 FSR 的内容作为指向指令的目标寄存器的指针。 INDF 操作数只是使用该指针的一种简便方法。

因为间接寻址使用完整的 12 位地址，所以没有必要进行数据 RAM 存储体的选择。因此， BSR 和快速操作 RAM 位的当前内容对于确定目标地址没有影响。

图 5-10： 间接寻址



5.4.3.2 FSR 寄存器与 POSTINC、POSTDEC、PREINC 和 PLUSW

除了 INDF 操作数外，每个 FSR 寄存器对还有另外 4 个间接操作数。与 INDF 相同，这些是 “虚拟” 寄存器，不能被直接读取或写入。访问这些寄存器实际上是访问相关的 FSR 寄存器对，但是还会对其存储的值执行特定的操作。它们是：

- POSTDEC：访问 FSR 值，然后在指令操作后自动将其减 “1”
- POSTINC：访问 FSR 值，然后在指令操作后自动将其加 “1”
- PREINC：将 FSR 值加 “1”，然后在指令操作中使用它
- PLUSW：让 W 寄存器的有符号的值 （范围是 -127 到 128）与 FSR 的值相加，然后在指令操作中使用这个新值

在此上下文中，访问 INDF 寄存器将使用 FSR 寄存器中的值而不更改它们。类似地，访问 PLUSW 寄存器会用 W 寄存器中的值作为 FSR 的偏移值；实际上在该操作中 W 寄存器中的值也不作更改。访问其他的虚拟寄存器将更改 FSR 寄存器的值。

用 POSTDEC、POSTINC 和 PREINC 对 FSR 进行操作将影响整个寄存器对，例如：FSRnL 寄存器从 FFh 计满返回到 00h，会向 FSRnH 寄存器进位。另外，这些操作的结果将不会更改 STATUS 寄存器中的任何标志位的值 （例如， Z、 N 和 OV 等）。

可以使用 PLUSW 寄存器在数据存储器空间中实现一种变址寻址形式。通过设置 W 寄存器中的值，用户可以访问指针地址加上固定偏移值后的地址。在有些应用中，这可以用来实现一些强大的程序控制结构，诸如数据存储器内部的软件堆栈。

5.4.3.3 对 FSR 的间接寻址

将其他 FSR 或虚拟寄存器作为目标寄存器的间接寻址操作是一类特殊情况。例如，使用一个 FSR 指向某个虚拟寄存器将会导致操作失败。举一个具体的例子，假定 FSR0H:FSR0L 包含了 FE7h，它是 INDF1 的地址。尝试读取 INDF1 的值时如果使用 INDF0 作为操作数，将返回 00h。尝试写 INDF1 时如果使用 INDF0 作为操作数，将导致 NOP。

另一方面，使用虚拟寄存器写 FSR 对的结果可能与预期的不同。在有些情况下，值将被写入 FSR 对但是不会对该值进行加减。因此，对 INDF2 或 POSTDEC2 进行写操作会将同一个值写入 FSR2H:FSR2L。

因为 FSR 是映射到 SFR 空间的物理寄存器，所以可以通过所有的直接操作对它们进行操作。当使用这些寄存器时，用户应该小心处理，尤其是当它们的代码使用间接寻址时。

类似地，在所有其他 SFR 上通常也允许使用间接寻址的操作。用户在进行此类操作时应该特别小心，以免不小心更改可能影响器件操作的设置。

5.5 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。

使能扩展指令集会向现有的PIC18指令集添加5个新的双字命令：ADDFSR、CALLW、MOVSF、MOVSS 和SUBFSR。**第 5.2.4 节“双字指令”**中说明这些指令的执行方式。

5.6 数据存储器和扩展指令集

使能 PIC18 扩展指令集（XINST 配置位 = 1）将显著地改变数据存储器及其寻址的某些方面。特别是很多 PIC18 内核指令对快速访问存储区的使用是不同的，这是由于引入了针对数据存储器空间的新的寻址模式。此模式还会改变使用 FSR2 及其相关操作数的间接寻址的方式。

了解没有发生改变的内容也同样重要。数据存储器空间的大小及其线性寻址方式都不会改变。SFR 映射保持不变。PIC18 内核指令仍然能在直接和间接寻址模式下操作；固有和立即数指令根本不变化。使用FSR0和FSR1的间接寻址也保持不变。

5.6.1 使用立即数作为偏移量进行变址寻址

使能PIC18扩展指令集将改变使用FSR2寄存器对及其相关文件操作数的间接寻址的行为。在正常条件下，使用快速操作存储区的指令（即大多数针对位的指令和针对字节的指令）可以使用在指令中使用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作为偏移量的变址寻址或立即数变址寻址。

当使用扩展指令集时，这种寻址模式有以下要求：

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

在这些条件下，指令的指针地址不被认为是地址的低字节 （与直接寻址中的 BSR 一起使用）或快速操作存储区中的 8 位地址。相反，该值被看作是由 FSR2 指定的地址指针的偏移量。该偏移量和 FSR2 的内容相加以获得指令操作的目标地址。

5.6.2 受立即数变址寻址模式影响的指令

任何使用直接寻址的PIC18内核指令均会受到立即数变址寻址模式的潜在影响。这包括所有针对字节和针对位的指令，或者几乎一半的标准 PIC18 指令集。只使用固有或立即数寻址模式的指令不受影响。

另外，如果针对字节和针对位的指令使用快速操作存储区 （快速操作 RAM 位为 “0”）或者包括一个大于或等于 60h 的文件地址，它们将不受影响。满足这些标准的指令将继续像以前那样执行。图 5-11中给出了当使能扩展指令集时可用的不同寻址模式的比较。

那些想要在立即数变址寻址模式中使用针对字节或针对位的指令的用户，应该注意此模式下汇编语法的改变。**第 24.0 节 “指令集综述”**中对此有更详细的说明。

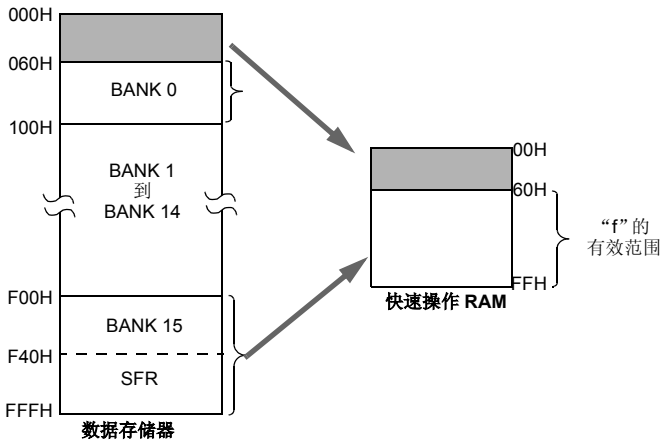
图 5-11：　　针对位和针对字节的指令的寻址方式对比 （使能了扩展指令集）

例如执行指令： ADDWF, f, d, a（操作码：0010 01da ffff ffff）

当 a = 0 且 f ≥ 60h 时：

该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。即数据存储器中的单元 F60h 到 FFFh （Bank 15）。

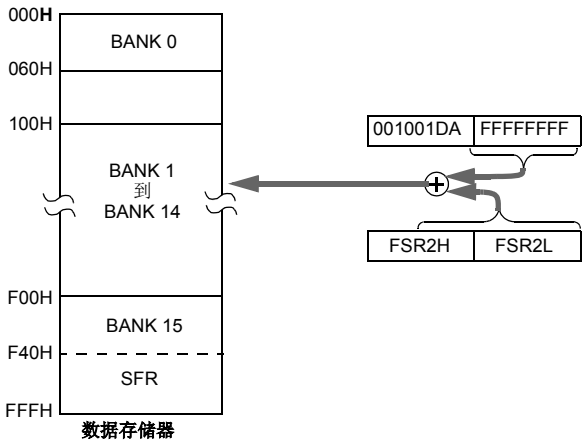
此模式寻址不可以使用低于 060h 的单元。



当 a = 0 且 f ≤ 5Fh 时：

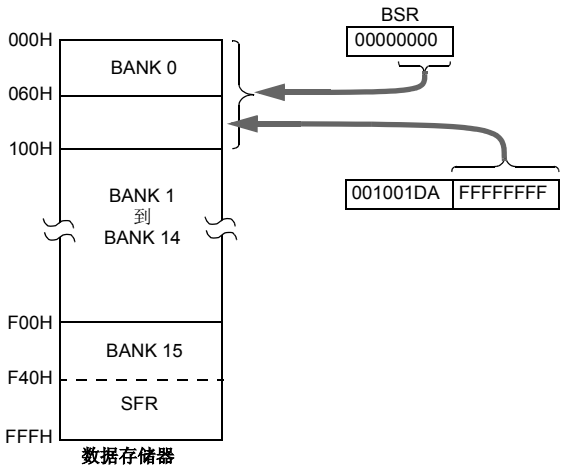
该指令以立即数变址寻址模式执行。“f”被解析为对 FSR2 中地址值的偏移量。两者相加以获得指令的目标寄存器的地址。该地址可以是数据存储器空间中的任何位置。

注意在此模式下，正确的语句应是：
ADDWF [k], d
其中 “k” 与 “f” 相同。



当 a = 1（f 的所有值）：

该指令以直接模式执行（也称为直接长模式）。“f”被解析为数据存储器空间的 16 个存储区中某个存储区中的单元。存储区由存储区选择寄存器（BSR）指定。地址可以位于数据存储器空间中任何未用的存储区。



5.6.3 在立即数变址寻址模式下映射快速操作存储区

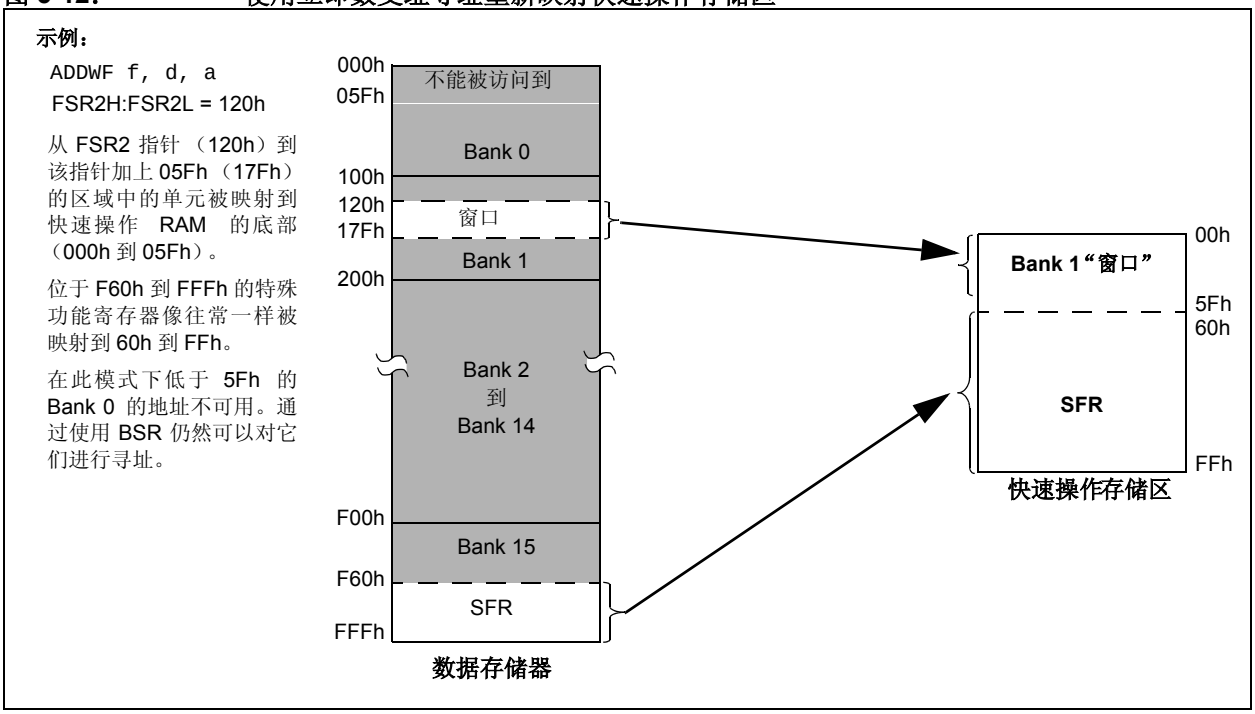
使用立即数变址寻址模式能有效地更改快速操作 RAM 低地址部分（00h 到 5Fh）的映射方式。除了包含 Bank 0 底部的内容之外，此模式还映射来自 Bank 0 和一个可以位于数据存储器空间中任何位置的用户定义的 “窗口”。FSR2 的值确定了映射到该窗口的地址的下边界，而上边界由 FSR2 加 95 （5Fh）定义。快速操作 RAM 中高于 5Fh 的地址的映射方式如前所述（见**第 5.3.2 节 “快速操作存储区”**）。图 5-12 中给出了在这种寻址模式下重新映射快速操作存储区的示例。

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”）的操作将像以前一样继续使用直接寻址。任何明确使用间接指针操作数（包括 FR2）进行的间接或变址操作都将继续以标准的间接寻址模式进行操作。任何使用快速操作存储区，但包括大于 05Fh 的寄存器地址的指令仍将使用直接寻址和常规的快速操作存储区映射。

5.6.4 立即数变址寻址模式中的 BSR

虽然当扩展指令集使能时会重新映射快速操作存储区，但 BSR 的操作将保持不变。直接寻址，使用 BSR 来选择数据存储区，操作方式与以前描述的相同。

图 5-12： 使用立即数变址寻址重新映射快速操作存储区



注：

6.0 闪存程序存储器

在整个 VDD 电压范围内，正常工作期间，闪存程序存储器都是可读、写和擦除的。

读程序存储器每次读取一个字节，写程序存储器每次写入 64 字节，而擦除操作则一次擦除 1024 字节。用户代码不能进行批量擦除的操作。

写或擦除程序存储器时将中止取指，直到写或擦除操作结束。在此期间，不可访问程序存储器，所以代码也不能得到执行。一个内部可编程定时器可终止对程序存储器的写或擦除操作。

写入程序存储空间的值不必是一条有效指令。执行存有无效指令的程序存储单元会导致执行一条 NOP 指令。

6.1 表读与表写

为了能读写程序存储器，有两条指令可使处理器在程序存储空间和数据 RAM 之间传送字节，即：

- 表读 （TBLRD）
- 表写 （TBLWT）

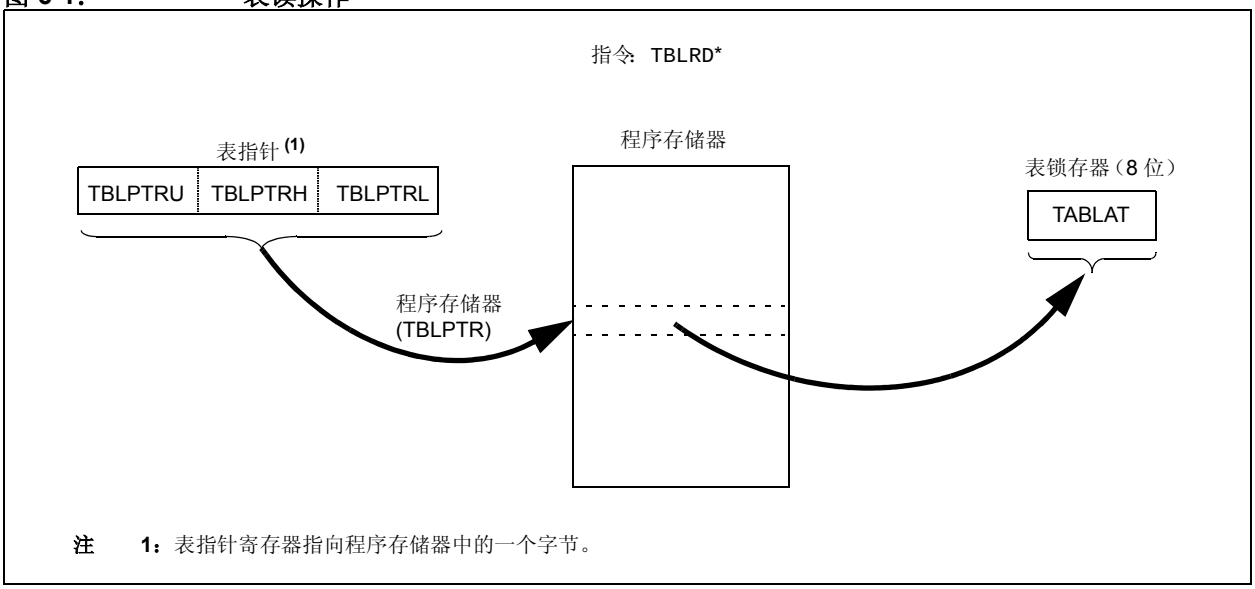
程序存储空间宽度为 16 位，而数据 RAM 空间的宽度为 8 位。表读和表写操作通过一个 8 位寄存器（TABLAT）在这两个存储空间之间传送数据。

表读操作从程序存储器中取出数据然后将其写入数据 RAM。图 6-1 说明了在程序存储器和数据 RAM 之间的表读操作。

表写操作将数据从数据RAM写入程序存储器的保持寄存器中。在**第 6.5 节 “写闪存程序存储器”**中将详细介绍把保持寄存器的内容写入程序存储器的过程。图 6-2 说明了在程序存储器和数据 RAM 之间的表写操作。

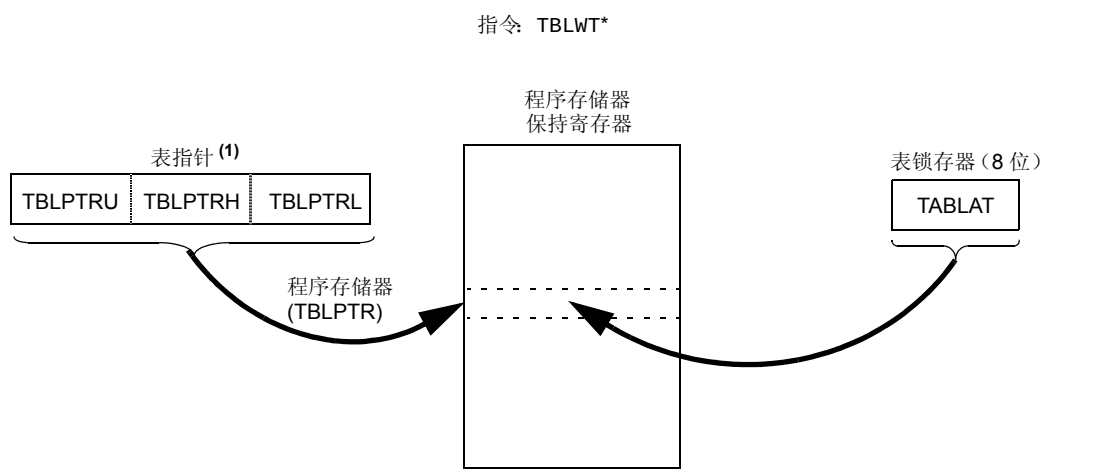
表操作的执行单位为字节。表块包含的是数据而非程序指令，所以不需要按字对齐。因此，表块可在任何字节地址处开始和结束。如果表写用于把可执行代码写入程序存储器，那么程序指令就需要按字对齐了。

图 6-1： 表读操作



注 1：表指针寄存器指向程序存储器中的一个字节。

图 6-2：　　表写操作



注　1：表指针指向 64 个保持寄存器之一，该寄存器的地址是由 TBLPTRL<5:0> 决定的。在**第 6.5 节“写闪存程序存储器”**中将介绍把数据写入程序存储器阵列的物理过程。

6.2　控制寄存器

TBLRD和TBLWT指令需要与几个控制寄存器配合使用，这些控制寄存器是：

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

6.2.1　EECON1 和 EECON2 寄存器

EECON1 寄存器（寄存器 6-1）是访问存储器的控制寄存器。 EECON2 寄存器不是一个物理寄存器，只用于对存储器的写和擦除操作，读 EECON2 将得到全 0。

FREE 位（置 1）将允许对程序存储器进行擦除操作。当 FREE 置 1 时，擦除操作将由下一条 WR 命令启动；当 FREE 位清零时，仅使能写操作。

WREN 位（置 1）将允许对程序存储器进行写操作。上电时 WREN 位被清零。 WRERR 位，在 WR 被置 1 时由硬件置 1，在内部可编程定时器超时、写操作结束时被清零。

注：	在正常工作期间，若 WRERR 的读取值为 1，则表明写操作被复位提早中止，或进行了不合法的写操作。

WR 控制位用于启动写操作。该位不能由软件清零，但可以被其置 1。当写操作结束时，该位由硬件清零。

寄存器 6-1：　EECON1：EEPROM 控制寄存器 1

U-0	U-0	U-0	R/W-0	R/W-x	R/W-0	R/S-0	U-0
—	—	—	FREE	WRERR	WREN	WR	—
bit 7							bit 0

bit 7-5　**未用：**读为 0

bit 4　**FREE：**闪存行擦除使能位
1 = 由下一条 WR 命令擦除 TBLPTR 指向的程序存储器行
（在擦除操作结束时被清零）
0 = 仅进行写操作

bit 3　**WRERR：**闪存程序错误标志位
1 = 写操作被提早终止（由于自定时编程期间的复位，或不合法的写入）
0 = 写操作完成

注：　当 WRERR 被置 1 时， EEPGD 和 CFGS 位并不会被清零。
这可用于追溯错误的原因。

bit 2　**WREN：**闪存程序写入使能位
1 = 允许写入闪存程序存储器
0 = 禁止写入闪存程序存储器

bit 1　**WR：**写控制位
1 = 启动程序存储器的擦除或写周期
（该操作是自定时的，一旦写操作结束，该位将由硬件清零。软件只能使 WR 位置 1 而不能将其清零。）
0 = 写周期完成

bit 0　**未用：**读为 0

图注：			
R = 可读位	W = 可写位		
S = 只能由软件置 1 而不能清零位		U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

6.2.2 表锁存器 （TABLAT）

表锁存器（TABLAT）是映射到 SFR 空间的一个 8 位寄存器。表锁存器用于保存在程序存储空间和数据 RAM 之间传输的 8 位数据。

6.2.3 表指针寄存器 （TBLPTR）

表指针寄存器（TBLPTR）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 个寄存器一起构成了一个 22 位宽的指针。器件使用该指针的低 21 位来寻址多达 2MB 的程序存储器空间，第 22 位可用于访问器件 ID、用户 ID 和配置位。

TBLRD和TBLWT指令要用到表指针寄存器TBLPTR。根据不同的表操作，上述指令可用4种方式更新TBLPTR，如表 6-1 所示。这些针对 TBLPTR 的操作只会影响到指针的低 21 位。

6.2.4 表指针边界

TBLPTR 用于读、写和擦除闪存程序存储器。

当执行 TBLRD 指令时，要用到 TBLPTR 的全部 22 位来决定将程序存储器中的哪一个字节读入 TABLAT。

当执行 TBLWT 指令时，使用表指针寄存器的低 7 位（TBLPTR<6:0>）来指定要写入 64 个程序存储器的哪一个保持寄存器。当对程序存储器的定时写周期开始后（通过 WR 位启动），TBLPTR 的高 12 位（TBLPTR<21:10>）决定要写入哪一个 1024 字节块。欲知更多详情，请参见**第 6.5 节“写闪存程序存储器”**。

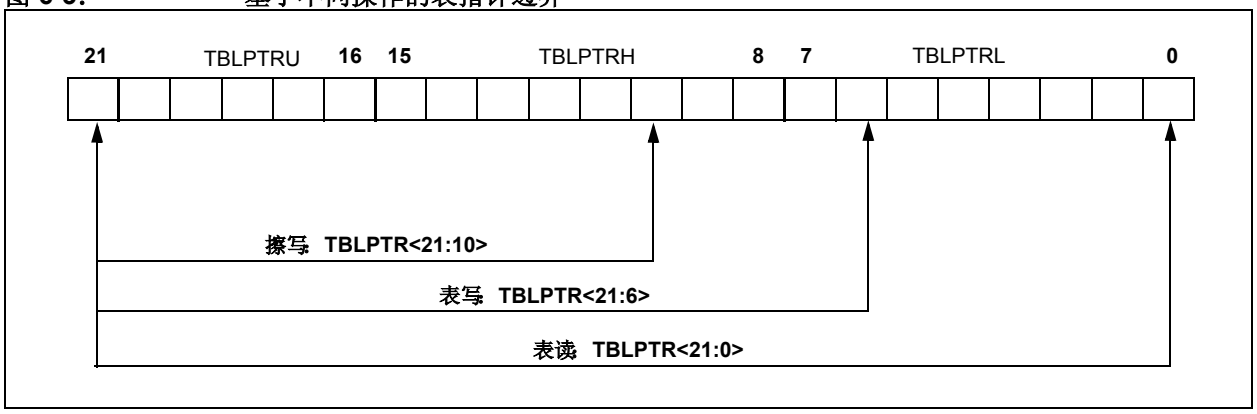
当擦除程序存储器时，表指针寄存器的高 12 位指向要擦除的 1024 字节块，而忽略低位。

图 6-3 给出了基于闪存程序存储器操作的 TBLPTR 边界。

表 6-1： 使用 TBLRD 和 TBLWT 指令对表指针的操作

示例	表指针操作
TBLRD* TBLWT*	TBLPTR 不变
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 6-3： 基于不同操作的表指针边界



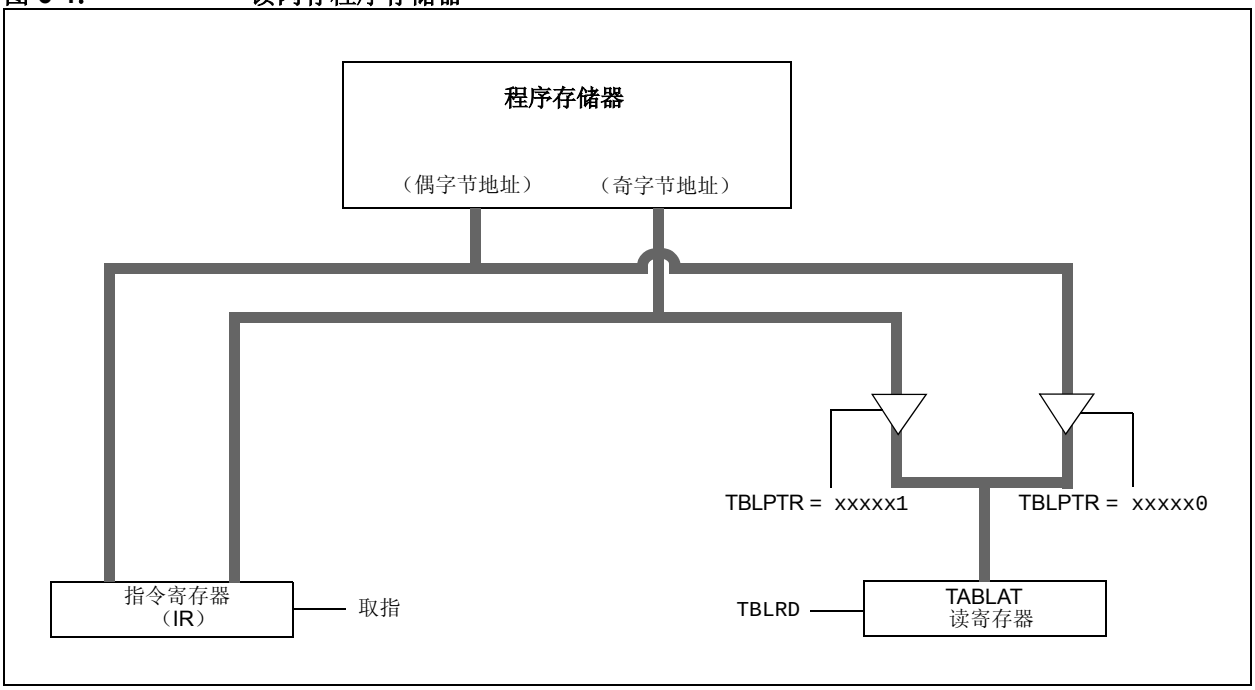
6.3 读闪存程序存储器

TBLRD指令用于从程序存储器获取数据并把数据写入数据 RAM。对程序存储器执行表读操作时，每次读取 1 个字节。

TBLPTR指向程序存储空间内的某个字节。执行TBLRD会将被指向的字节存入 TABLAT，同时自动修改TBLPTR 以进行下一次表读操作。

内部程序存储器通常以字为单位组织。由地址的最低有效位来选择字的高字节和低字节。图 6-4 显示了内部程序存储器和 TABLAT 之间的接口。

图 6-4： 读闪存程序存储器



例 6-1： 读一个闪存程序存储器字

```
            MOVLW    CODE_ADDR_UPPER        ; Load TBLPTR with the base
            MOVWF    TBLPTRU                ; address of the word
            MOVLW    CODE_ADDR_HIGH
            MOVWF    TBLPTRH
            MOVLW    CODE_ADDR_LOW
            MOVWF    TBLPTRL
READ_WORD
            TBLRD*+                         ; read into TABLAT and increment
            MOVF     TABLAT, W              ; get data
            MOVWF    WORD_EVEN
            TBLRD*+                         ; read into TABLAT and increment
            MOVF     TABLAT, W              ; get data
            MOVF     WORD_ODD
```

6.4 擦除闪存程序存储器

最小的擦除块大小为 512 个字或 1024 个字节。只有通过外部编程器或 ICSP 控制，才能批量擦除更大的程序存储块。不支持闪存阵列的单字擦除。

当由单片机本身启动擦除操作时，将擦除程序存储器的一个 1024 字节块。TBLPTR 的高 12 位（TBLPTR<21:10>）指向要擦除的字节块，TBLPTR<9:0> 被忽略。

EECON1 寄存器控制擦除操作。WREN 位必须被置 1 以使能写操作。FREE 位被置 1 选择擦除操作。为了安全起见，必须使用 EECON2 的写启动顺序。

擦除内部闪存需要使用长写操作。在长写周期，指令停止执行。由内部可编程定时器来终止长写周期。

6.4.1 擦除闪存程序存储器的操作顺序

擦除内部程序存储器的步骤如下：

1. 将要擦除的行地址装入表指针寄存器。
2. 将 WREN 和 FREE 位（EECON1<2,4>）置 1，使能擦除操作。
3. 禁止中断。
4. 把 55h 写入 EECON2。
5. 把 0AAh 写入 EECON2。
6. 将 WR 位置 1，启动行擦除周期。
7. 在擦除操作期间，CPU将在Tiw（见参数D133A）时间段内停止工作。
8. 重新允许中断。

例 6-2： 擦除闪存程序存储器中的一行

```
                MOVLW   CODE_ADDR_UPPER     ; load TBLPTR with the base
                MOVWF   TBLPTRU             ; address of the memory block
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
ERASE_ROW
                BSF     EECON1, EEPGD       ; point to Flash program memory
                BCF     EECON1, CFGS        ; access Flash program memory
                BSF     EECON1, WREN        ; enable write to memory
                BSF     EECON1, FREE        ; enable Row Erase operation
                BCF     INTCON, GIE         ; disable interrupts
必要的序列      MOVLW   55h
                MOVWF   EECON2              ; write 55h
                MOVLW   0AAh
                MOVWF   EECON2              ; write 0AAh
                BSF     EECON1, WR          ; start erase (CPU stall)
                BSF     INTCON, GIE         ; re-enable interrupts
```

6.5 写闪存程序存储器

最小的编程字节块大小为 32 个字或 64 个字节。不支持单字或单字节编程。

表写操作在器件内部用来装载保持寄存器，以便对闪存存储器进行编程。编程时的表写操作总共要用到 64 个保持寄存器。

由于表锁存器 （TABLAT）只有一个字节大小，所以每次编程操作需要执行 64 次 TBLWT 指令。因为只需写入保持寄存器，故所有的表写操作都是短写周期。在更新完全部 64 个保持寄存器之后，写入 EECON1 寄存器，以长写方式启动编程操作。

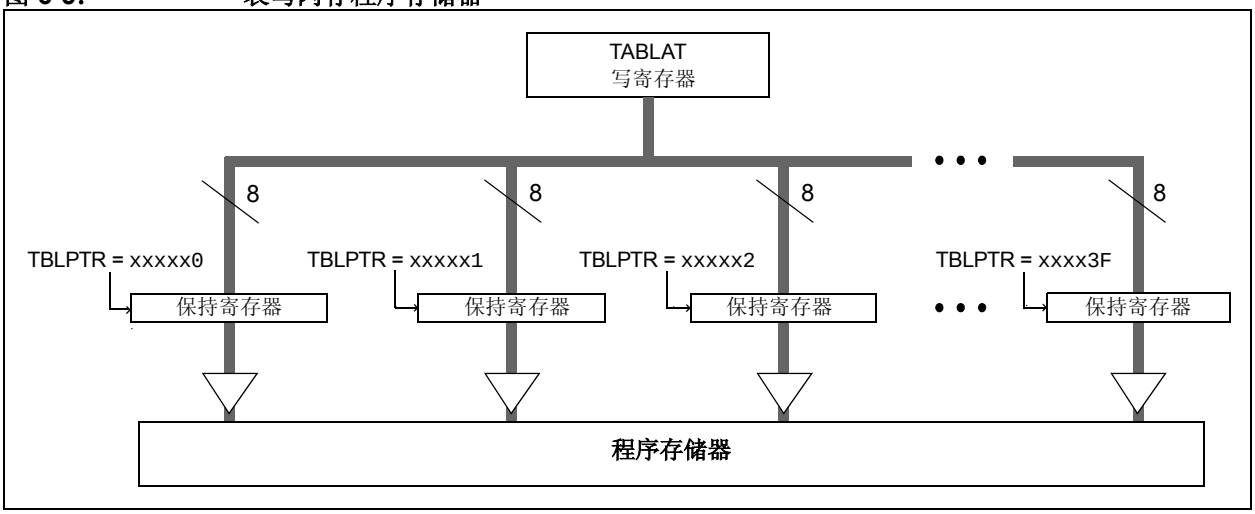
对内部闪存的编程需要使用长写周期，在此期间，指令停止执行。由内部可编程定时器来终止长写周期。

由片上定时器控制写入的时间。写入 / 擦除电压则由片上的电荷泵产生，该电荷泵可以工作在器件的电压范围内。

注 1： 与以前的 PICmicro 器件不同，PIC18F87J10系列器件在写周期开始后并不复位保持寄存器。必须在编程开始之前清零或重写保持寄存器。

2： 为了保证程序存储器的耐擦写次数，在两次擦除操作之间，只允许对闪存字节进行 1 次编程。在对某一字节的内容作第二次修改之前，必须进行行擦除或全部空间的批量擦除。

图 6-5： 表写闪存程序存储器



6.5.1 写闪存程序存储器的操作顺序

对内部程序存储器编程的步骤如下：

1. 读取 1024 个字节存入 RAM。
2. 如需要，更新 RAM 中的数据。
3. 将要擦除的地址装入表指针寄存器。
4. 执行行擦除操作。
5. 将要写入的首字节地址装入表指针寄存器，该值减 1。
6. 采用自动递增的方式把 64 个字节写入保持寄存器。
7. 将 WREN 位（EECON1<2>）置 1，使能字节写。
8. 禁止中断。
9. 把 55h 写入 EECON2。
10. 把 0AAh 写入 EECON2。
11. 将 WR 位置 1，启动写周期。
12. 在写入期间，在 TIW （见参数 D133A）时间段内 CPU 将停止工作。
13. 重新允许中断。
14. 重复步骤6到步骤13，直到全部1024个字节都被写入程序存储器。
15. 对存储器进行校验 （通过表读）。

下一页上的例 6-3 给出了一个代码示例。

注： 在将 WR 位置 1 前，表指针必须指向保持寄存器中的 64 个字节的地址范围内。

例 6-3：　　写闪存程序存储器

```
                    MOVLW   CODE_ADDR_UPPER        ; Load TBLPTR with the base address
                    MOVWF   TBLPTRU                ; of the memory block, minus 1
                    MOVLW   CODE_ADDR_HIGH
                    MOVWF   TBLPTRH
                    MOVLW   CODE_ADDR_LOW
                    MOVWF   TBLPTRL
ERASE_BLOCK
                    BSF     EECON1, WREN           ; enable write to memory
                    BSF     EECON1, FREE           ; enable Row Erase operation
                    BCF     INTCON, GIE            ; disable interrupts
                    MOVLW   55h
                    MOVWF   EECON2                 ; write 55h
                    MOVLW   0AAh
                    MOVWF   EECON2                 ; write 0AAh
                    BSF     EECON1, WR             ; start erase (CPU stall)
                    BSF     INTCON, GIE            ; re-enable interrupts
                    MOVLW   D'16'
                    MOVWF   WRITE_COUNTER          ; Need to write 16 blocks of 64 to write
                                                   ; one erase block of 1024
RESTART_BUFFER
                    MOVLW   D'64'
                    MOVWF   COUNTER
                    MOVLW   BUFFER_ADDR_HIGH       ; point to buffer
                    MOVWF   FSR0H
                    MOVLW   BUFFER_ADDR_LOW
                    MOVWF   FSR0L
FILL_BUFFER
                    ...                            ; read the new data from I2C, SPI,
                                                   ; PSP, USART, etc.
WRITE_BUFFER
                    MOVLW   D’64                   ; 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_WORD_TO_HREGS
PROGRAM_MEMORY
                    BSF     EECON1, WREN           ; enable write to memory
                    BCF     INTCON, GIE            ; disable interrupts
                    MOVLW   55h
    必需的序列      MOVWF   EECON2                 ; write 55h
                    MOVLW   0AAh
                    MOVWF   EECON2                 ; write 0AAh
                    BSF     EECON1, WR             ; start program (CPU stall)
                    BSF     INTCON, GIE            ; re-enable interrupts
                    BCF     EECON1, WREN           ; disable write to memory

                    DECFSZ  WRITE_COUNTER          ; done with one write cycle
                    BRA     RESTART_BUFFER         ; if not done replacing the erase block
```

6.5.2 写校验

根据应用的不同，好的编程习惯一般要求将写入存储器的值与原始值进行比对校验。在连续写入字节数已接近规范极限值的应用中必须使用写校验。

6.5.3 写操作的意外终止

当一次写操作被意外事件，如掉电或意外的复位终止时，必须对刚刚被编程的存储器单元内容进行校验，如需要还要重新对其编程。在正常工作期间，如果写操作被 $\overline{\text{MCLR}}$ 复位或 WDT 超时复位中断，用户可以检查 WRERR 位的状态并重新写入（如需要）。

6.6 代码保护期间的闪存程序存储器操作

请参见**第 23.6 节 “程序校验和代码保护”**了解有关代码保护期间闪存程序存储器操作的详细信息。

表 6-2： 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	复位值所在页
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 （TBLPTR<20:16>）					49
TBPLTRH	程序存储器表指针高字节 （TBLPTR<15:8>）								49
TBLPTRL	程序存储器表指针低字节 （TBLPTR<7:0>）								49
TABLAT	程序存储器表锁存器								49
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
EECON2	程序存储器控制寄存器 2 （非物理寄存器）								51
EECON1	—	—	—	FREE	WRERR	WREN	WR	—	51

图注： — = 未用，读为 0。访问程序存储器操作不使用阴影单元。

注：

7.0 外部存储总线

注： 64 引脚器件内未实现外部存储总线。

外部存储总线使器件能像访问程序存储器或数据存储器一样访问外部存储器件（如闪存、EPROM 和 SRAM 等）。外部存储总线同时支持 8 位和 16 位数据宽度模式和三种高达 20 位的地址宽度模式。

总线包含 28 个引脚，4 个 I/O 端口引脚复用。三个端口（PORTD、PORTE 和 PORTH）同总共 20 根地址 / 数据总线复用， PORTJ 同总线控制信号复用。

表 7-1 罗列了总线引脚并介绍了它们的功能。

表 7-1： PIC18F8XJ10/8XJ15 外部总线——I/O 端口功能

名称	端口	位	外部存储总线功能
RD0/AD0	PORTD	0	地址 bit0 或数据 bit0
RD1/AD1	PORTD	1	地址 bit1 或数据 bit1
RD2/AD2	PORTD	2	地址 bit2 或数据 bit2
RD3/AD3	PORTD	3	地址 bit3 或数据 bit3
RD4/AD4	PORTD	4	地址 bit4 或数据 bit4
RD5/AD5	PORTD	5	地址 bit5 或数据 bit5
RD6/AD6	PORTD	6	地址 bit6 或数据 bit6
RD7/AD7	PORTD	7	地址 bit7 或数据 bit7
RE0/AD8	PORTE	0	地址 bit8 或数据 bit8
RE1/AD9	PORTE	1	地址 bit9 或数据 bit9
RE2/AD10	PORTE	2	地址 bit10 或数据 bit10
RE3/AD11	PORTE	3	地址 bit11 或数据 bit11
RE4/AD12	PORTE	4	地址 bit12 或数据 bit12
RE5/AD13	PORTE	5	地址 bit13 或数据 bit13
RE6/AD14	PORTE	6	地址 bit14 或数据 bit14
RE7/AD15	PORTE	7	地址 bit15 或数据 bit15
RH0/A16	PORTH	0	地址 bit16
RH1/A17	PORTH	1	地址 bit17
RH2/A18	PORTH	2	地址 bit18
RH3/A19	PORTH	3	地址 bit19
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	写低位（$\overline{WRL}$）控制引脚
RJ3/$\overline{WRH}$	PORTJ	3	写高位（$\overline{WRH}$）控制引脚
RJ4/BA0	PORTJ	4	地址字节 bit0（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： 为避免混淆，这里只显示 I/O 端口和外部总线分配。在一些引脚上可能还有 1 个或多个其他的复用功能。

7.1 外部存储总线控制

接口的工作模式由 MEMCON 寄存器控制（寄存器 7-1）。MEMCON寄存器在所有的程序存储器工作模式（除单片机模式）下可用。在单片机模式中，寄存器被禁止且不能写入。

EBDIS 位 （MEMCON<7>）控制总线的工作模式和相关端口的功能。将EBDIS位清零会使能接口和禁止端口的 I/O 功能及其他复用这些引脚的任何功能。将该位置 1 会使能 I/O 端口和其他复用这些引脚的功能，但当需要外部存储器工作时，接口的优先级高于引脚上的任何功能。默认情况下，外部总线总是被使能的并禁止其他所有的 I/O。

对EBDIS位的操作也受正在使用的程序存储器模式的影响。**第 7.5 节 “程序存储模式和外部存储总线”**详细讨论了 EBDIS 位的操作。

WAIT 位给外部存储工作增加等待状态。**第 7.3 节 “等待状态”**讨论了这些位的使用。

WM 位选择总线在 16 位数据宽度模式下使用的特定工作模式。**第 7.6 节 “16 位数据宽度模式”**详细讨论了这些操作。当选择使用 8 位数据宽度的模式时这些位不起作用。

寄存器 7-1： MEMCON：外部存储总线控制寄存器

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
bit7							bit0

bit 7 **EBDIS**：外部总线禁止位
1 = 当单片机访问外部存储器时使能外部总线，否则，所有的外部总线驱动器映射为 I/O 端口
0 = 一直使能外部总线，禁止 I/O 端口

bit 6 **未用**：读为 0

bit 5-4 **WAIT1:WAIT0**：表读和表写总线周期等待计数位
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 **未用**：读为 0

bit 1-0 **WM1:WM0**：16 位数据总线宽度的 TBLWT 操作选择位
1x = 写字模式：TABLAT0 和 TABLAT1 作为字输出，当写 TABLAT1 时 $\overline{WRH}$ 有效
01 = 字节选择模式：TABLAT 数据复制到 MSB 和 LSB， $\overline{WRH}$ 和 （$\overline{UB}$ 或 $\overline{LB}$） 将有效
00 = 写字节模式：TABLAT 数据复制到 MSB 和 LSB， $\overline{WRH}$ 或 $\overline{WRL}$ 将有效

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

7.2 地址和数据宽度

PIC18F87J10 系列器件能在相同的存储总线上为不同的地址和数据宽度独立进行配置。两者均由 CONFIG3L 寄存器中的配置位设置。作为配置位意味着这些选项只能通过对器件编程才能进行配置，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（缺省）会选择 16 位数据宽度。

EMB1:EMB0 位决定程序存储器工作模式和地址总线宽度。可用的总线宽度有：20 位、16 位、12 位和单片机模式（禁止外部总线）。选择 16 位或 12 位宽度会使相应的高位线用于 I/O 功能；这些引脚不再受 EBDIS 位设置的影响。例如，选择 16 位地址模式（EMB1:EMB0 = 01）会禁止 A19:A16 和允许 PORTH<3:0> 工作不受总线中断控制。使用较短地址宽度使用户可为特定的设计定制存储总线以适应外部存储空间的大小，同时留出引脚用于专用I/O 操作。

因为 EMB 位能禁止引脚用于存储总线操作，所以选择一个至少与数据宽度相等的地址宽度尤为重要。如果 12 位地址宽度用于 16 位数据宽度，那么总线上数据的高 4 位将丢失。

所有地址和数据宽度的组合要求在相同的线上复用地址和数据信息。表 7-2 总结了地址和数据复用以及使用短地址宽度时可用的 I/O 端口。

7.2.1 外部总线上的地址移位

默认情况下， PC 值就是外部总线上的地址。实际上对单片机而言，低于片内存储器最高地址的外部存储器件的地址是不可用的。要访问这些物理存储单元，单片机和外部存储器间的连接逻辑必须设法译址。

为简化接口，外部总线提供了一种扩展单片机模式，它能自动进行地址移位。EASHFT 配置位控制该功能。将该位置 1 会使总线上的实际地址发生偏移，偏移量为单片机的片内程序存储器大小，并设置总线的起始地址为 0000h。这样器件就可以访问外部存储器的所有物理地址。

7.2.2 21 位寻址

作为 20 位地址宽度操作的扩展，外部存储总线也能寻址整个 2 MB 的存储空间。将总线地址 Bit 0（BA0）控制线用作地址的最低有效位可以完成上述功能。 UB 和 LB 控制信号还可能同某些存储器件一起使用在 16 位宽的数据字内选择高字节和低字节。

在 8 位和某些 16 位数据宽度模式中可使用该寻址模式。**第 7.6.3 节“16 位字节选择模式”**和**第 7.7 节“8 位模式”**中提供了更多详细信息。

表 7-2： 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（和相应端口）	仅地址线（和相应端口）	可用作 I/O 的端口
8 位	12 位	AD7:AD0（PORTD<7:0>）	AD11:AD8（PORTE<3:0>）	PORTE<7:4>，PORTH 的全部引脚
	16 位		AD15:AD8（PORTE<7:0>）	PORTH 的全部引脚
	20 位		A19:A16, AD15:AD8（PORTH<3:0>，PORTE<7:0>）	—
16 位	16 位	AD15:AD0（PORTD<7:0>，PORTE<7:0>）	—	PORTH 的全部引脚
	20 位		A19:A16（PORTH<3:0>）	—

7.3 等待状态

一般认为外部存储器件将工作在单片机时钟速率下，但经常不是这样。事实上，许多器件读或写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，在使用总线时可以配置外部存储总线给表操作添加一个固定的延时。设置 WAIT 配置位使能等待状态。当使能时，通过 WAIT1:WAIT0 位（MEMCON<5:4>）设置延时长度。延时长度是单片机指令周期时间的倍数，添加在表操作指令周期后。延时范围为 0 至 3 个 TCY（默认值）。

7.4 端口引脚弱上拉

除高位地址线 A19:A16 外，与外部存储总线相关的引脚均配有弱上拉。PORTG 寄存器的高 3 位控制上拉。高三位分别命名为 RDPU、REPU 和 RJPU，分别控制 PORTD、PORTE 和 PORTJ 上的弱上拉。对其中一位进行置位会使能相应端口的上拉。默认情况下，器件复位时禁止所有的上拉。

7.5 程序存储模式和外部存储总线

同时使用片内和外部程序存储器时，PIC18F87J10 系列器件能工作在两种程序存储模式中的任意一种模式下。端口引脚的复用功能取决于选定的程序存储模式和 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， EMB 引脚充当 I/O 端口。

在**扩展单片机模式下**，外部程序存储总线共享引脚的 I/O 端口功能。当器件在外部程序存储空间上进行取指或表读 / 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和访问，EBDIS 控制位将把引脚从外部存储功能改为 I/O 端口功能。当 EBDIS=0 时，引脚作为外部总线使用。当 EBDIS=1 时，引脚作为 I/O 端口使用。

在 EBDIS=1 时，器件对外部存储器进行取指或访问，引脚将切换为外部总线功能。如果在外部存储器执行的程序将 EBDIS 位置 1，那么置 1 动作将被延时，直到程序转到内部存储器为止。那时，引脚将从外部总线功能改为 I/O 端口功能。

在 EBDIS=0 时如果器件在内部存储器外执行，存储总线的地址 / 数据和控制引脚将失效。它们进入下面的状态 有效的地址 / 数据引脚处于三态； CE、OE、WRH、WRL、UB 和 LB 信号为 1， ALE 和 BA0 为 0。注意只有这些与当前地址宽度关联的引脚被强制为三态；其他引脚继续作为 I/O 使用。例如在 16 位地址宽度情况下，只有 AD<15:0>（PORTD 和 PORTE）受到影响；A19:A16 （PORTH<3:0>）继续作为 I/O 使用。

在所有的外部存储模式下，总线的优先级高于其他任一与总线共享引脚的外设。包括并行从动端口和串行通信模块，这两者的优先级高于 I/O 端口。

7.6 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三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

MEMCON 寄存器中的 WM1:WM0 （MEMCON<1:0>）决定了使用的配置。这三种不同的配置使设计人员在使用带16位数据的8位和16位器件时具有最大的灵活性。

对于所有的 16 位模式，地址锁存使能 （ALE）引脚表明在外部存储接口总线上地址位 AD<15:0> 可用。在地址锁存之后，输出使能信号 （OE）将立即使程序存储器的两个字节形成一个 16 位的指令字。在单片机访问外部存储器（不论读或写）的任何时间内芯片使能信号（CE）始终有效。只要器件一进入休眠模式，信号就变为无效 （变为高电平）。

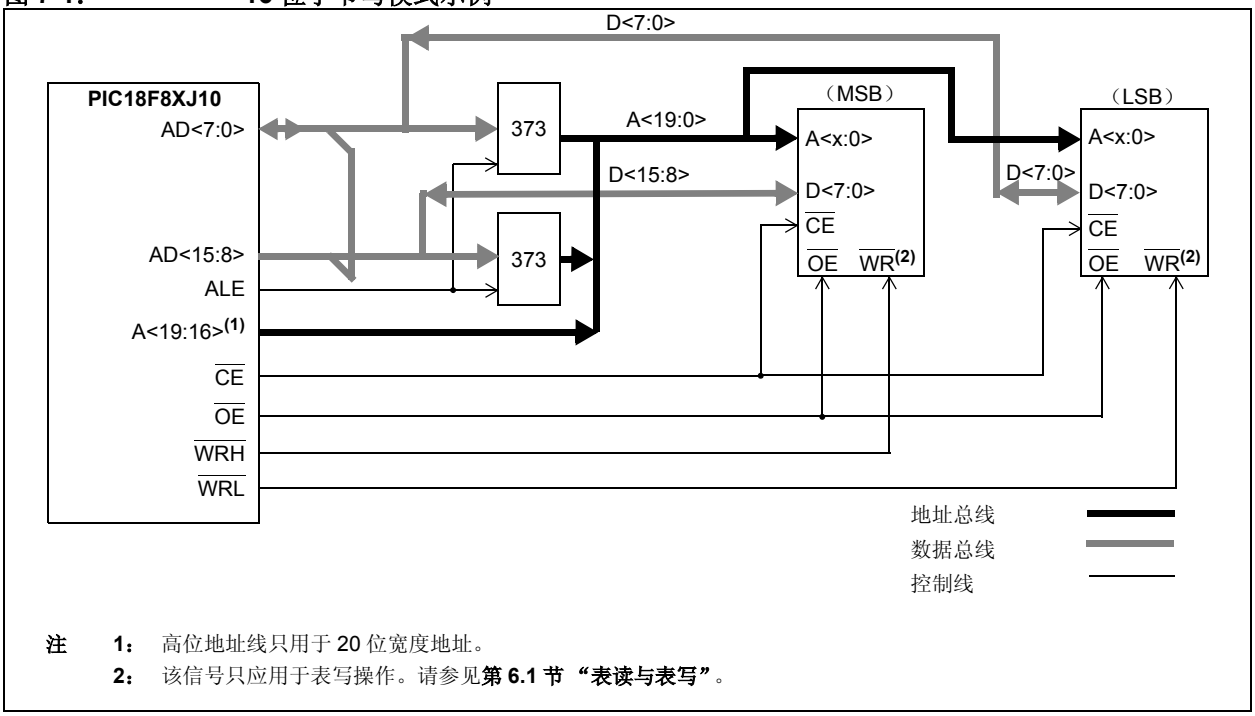
在字节选择模式下，JEDEC标准闪存存储器将需要BA0用于字节地址线和 I/O 线，以在字节和字模式间进行选择。其他 16 位模式不需要 BA0。JEDEC 标准静态 RAM 存储器将使用 UB 或 LB 信号选择字节模式。

7.6.1 16 位字节写模式

图 7-1 举例说明了 PIC18F87J10 系列器件的 16 位字节写模式。该模式用于将 2 个独立的 8 位存储器相连进行 16 位的操作。通常包括基本 EPROM 和闪存器件。允许对字节宽的外部存储器进行表写操作。

在 TBLWT 指令周期内， TABLAT 数据在 AD15:AD0 总线的高低字节中。TBLPTR 的 LSb 选通相应的 WRH 或 WRL 控制线。

图 7-1： 16 位字节写模式示例



注 1： 高位地址线只用于 20 位宽度地址。

2： 该信号只应用于表写操作。请参见**第 6.1 节 “表读与表写”**。

7.6.2 16 位字写模式

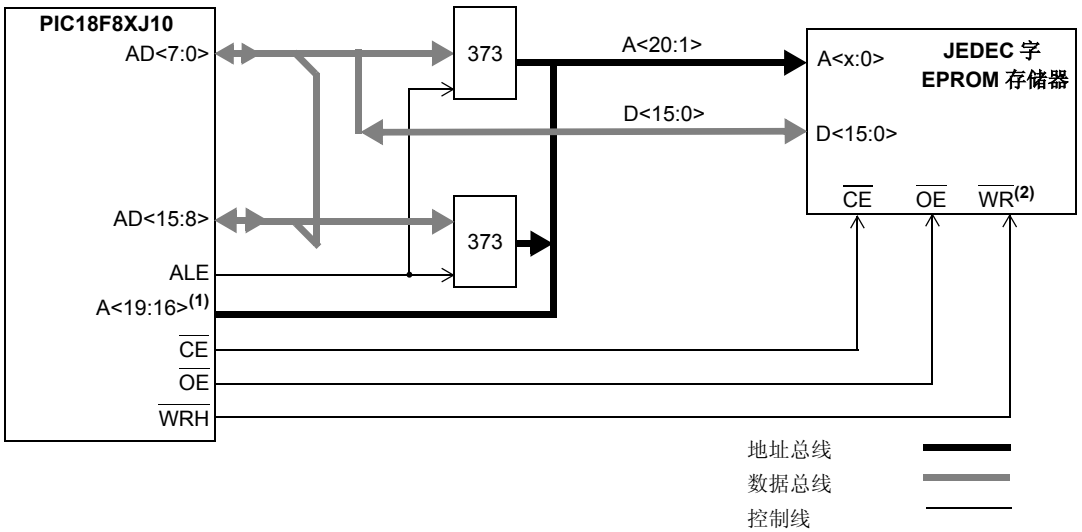
图 7-2 显示了 PIC18F65J10 系列器件的 16 位字写模式。该模式用于字宽存储器，包括一些 EPROM 和闪存类型的存储器。该模式允许从所有的 16 位存储器取操作码和表读以及向所有的字宽外部存储器表写。这种方式把 TBLWT 周期分为写入偶地址或写入奇地址。

在写入偶地址（TBLPTR<0> =0）的 TBLWT 周期中，TABLAT 数据转移到保持锁存器中，外部地址数据总线呈现为三态，用作数据总线。没有有效写信号。

在写入奇地址（TBLPTR<0> = 1）的 TBLWT 周期中，TABLAT 中的数据出现在 AD15:AD0 总线的高字节上。保持锁存器的内容出现在 AD15:AD0 总线的低字节上。

$\overline{WRH}$ 信号选通每个写周期，不使用 $\overline{WRL}$ 引脚。BA0 引脚上的信号表示 TBLPTR 的 LSb，但该引脚是悬空的。相反，$\overline{UB}$ 和 $\overline{LB}$ 信号有效，可用于同时选择奇偶地址的两个字节。这种方法的明显缺陷是表写必须在特定字边界上成对完成，以便正确地写一个字存储单元。

图 7-2： 16 位字写模式示例



注 1： 高位地址线只用于 20 位宽度地址。

2： 该信号只应用于表写操作。请参见**第 6.1 节 “表读与表写”**。

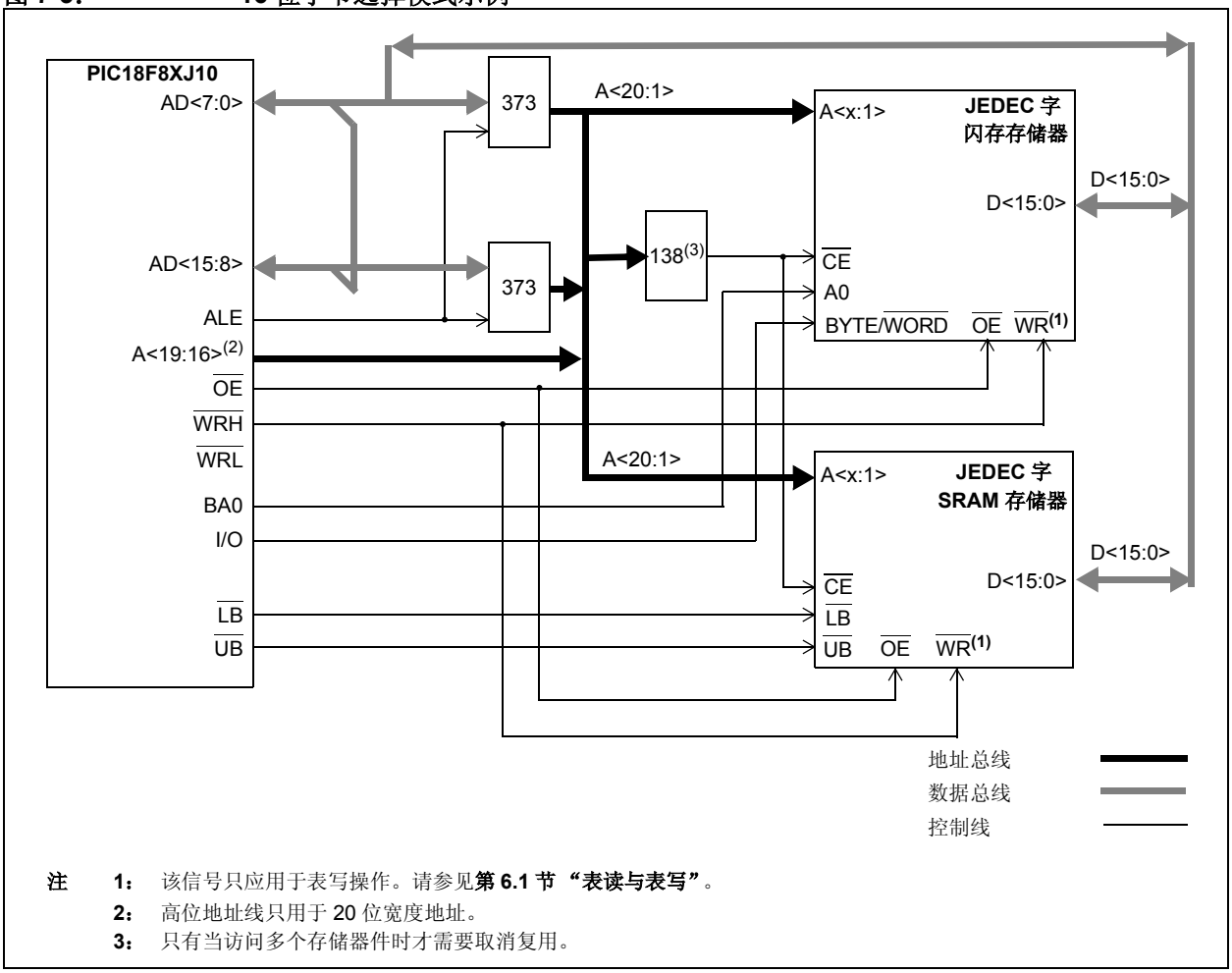
7.6.3 16 位字节选择模式

图 7-3 给出了一个 16 位字节选择模式示例。该模式允许对带字节选择功能的字宽外部存储器进行表写操作。这个通常包括字宽闪存和 SRAM 器件。

在 TBLWT 周期内，TABLAT 中的数据出现在 AD15:AD0 总线的高低字节中。WRH 信号选通每个写周期； WRL 没有使用。 BA0 或 UB/LB 信号用于根据 TBLPTR 寄存器中的最低有效位选择要写的字节。

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控制器 I/O 端口引脚连接到存储器的 BYTE/WORD 引脚，用来提供选择信号。它们还使用控制器的 BA0 信号作为字节地址。在另一方面，JEDEC 标准静态 RAM 存储器使用 UB 和 LB 信号选择字节。

图 7-3： 16 位字节选择模式示例



注 1： 该信号只应用于表写操作。请参见**第 6.1 节 “表读与表写”**。

2： 高位地址线只用于 20 位宽度地址。

3： 只有当访问多个存储器件时才需要取消复用。

7.6.4 16 位模式时序

在外部存储总线上出现的控制信号在不同的操作模式下是不同的。图 7-4 和图 7-5 所示为典型的信号时序图。

图 7-4： 执行 TBLRD 的外部存储总线时序 （扩展单片机模式）

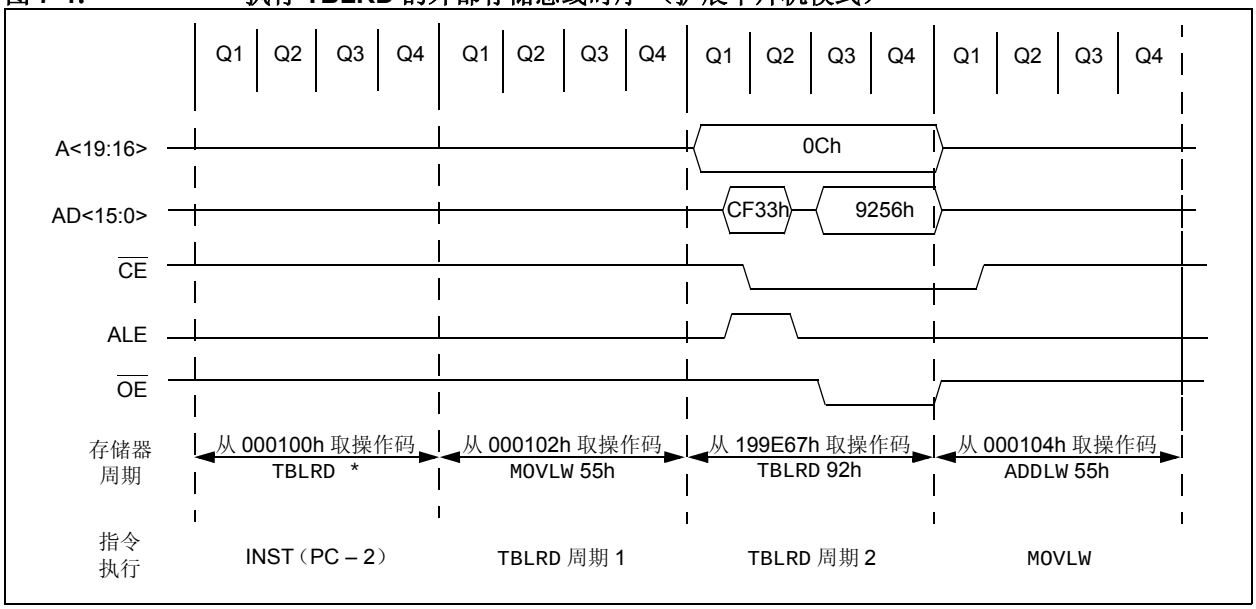
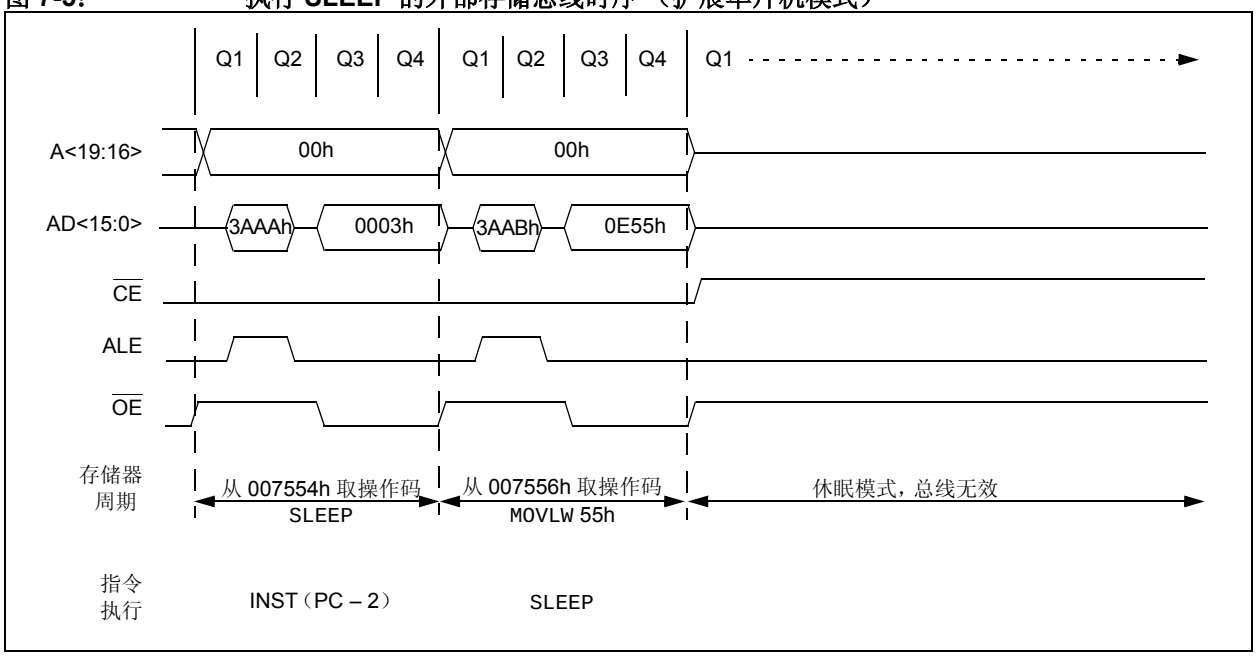


图 7-5： 执行 SLEEP 的外部存储总线时序 （扩展单片机模式）



7.7 8 位模式

在 8 位数据宽度模式下，外部存储总线只工作在复用模式下，即数据总线共享地址总线的低 8 位。

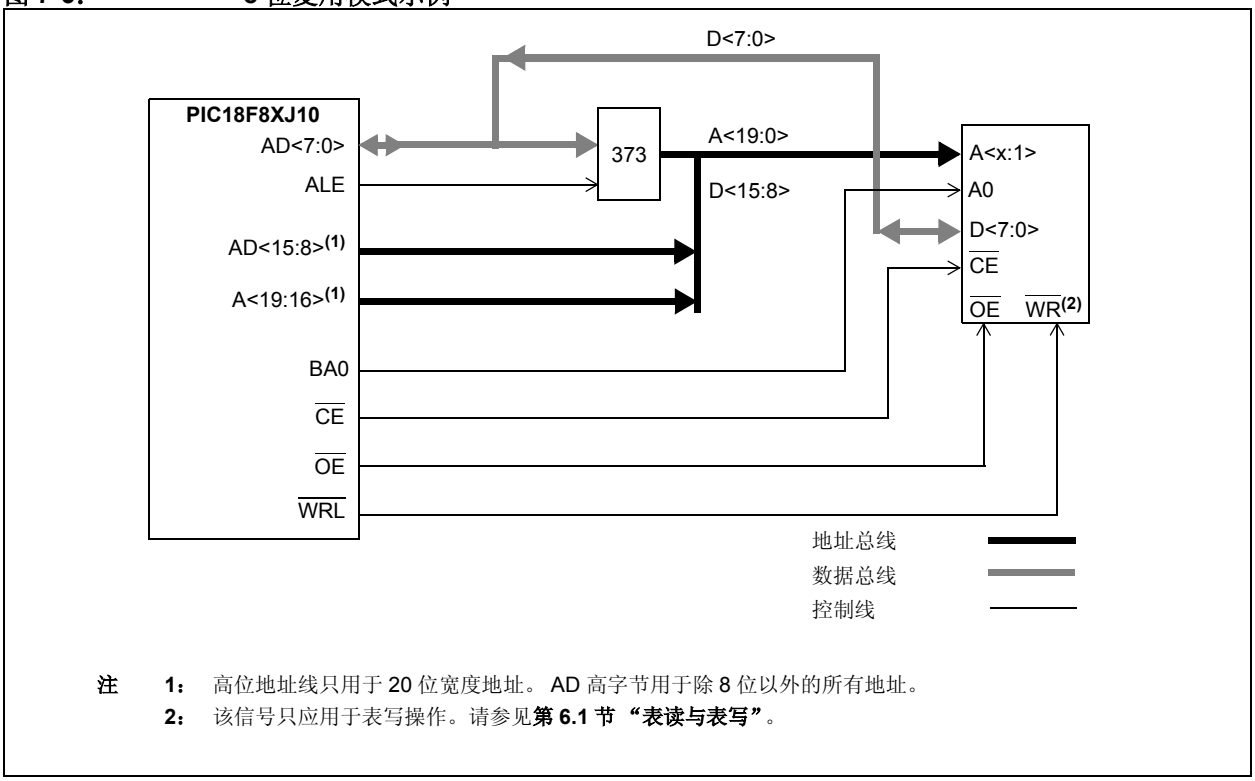
图 7-6 显示了 80 引脚器件的一个 8 位复用模式示例。该模式用于 2 个 8 位存储器连接进行 16 位操作。在共享数据 / 地址总线上取 2 个 8 位字节指令，这两个字节在一个指令周期 （TCY）内顺序取指。所以，设计人员必须选择根据基于 1/2 TCY （2 倍指令速率）的时序计算选择外部存储器件。为了选择合适的存储器速度，必须考虑到连接逻辑传播延迟时间以及建立时间和保持时间。

地址锁存使能 （Address Latch Enable， ALE）引脚表明在外部存储接口总线上地址位 AD<15:0> 可用。输出使能信号 （$\overline{OE}$）将启用程序存储器的一个字节作为指令周期的一部分，然后会更改 BA0 并启用第二个字节形成 16 位的指令字。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（不论读或写）的任何时间内芯片使能信号（$\overline{CE}$）始终有效。只要器件一进入休眠模式，信号就变为无效（变为高电平）。

通常包括基本 EPROM 和闪存器件。允许对字节宽外部存储器进行表写操作。

在一个 TBLWT 指令周期内，TABLAT 数据在 AD15:AD0 总线的高低字节中。TBLPTR 的 LSb 用于选通 BA0 控制线上相应的电平。

图 7-6： 8 位复用模式示例



注 1： 高位地址线只用于 20 位宽度地址。 AD 高字节用于除 8 位以外的所有地址。

2： 该信号只应用于表写操作。请参见**第 6.1 节 “表读与表写”**。

7.7.1 8 位模式时序

在外部存储总线上出现的控制信号在不同的操作模式下是不同的。图 7-7 和图 7-8 所示为典型信号时序图。

图 7-7： 执行 TBLRD 的外部存储总线时序 （扩展单片机模式）

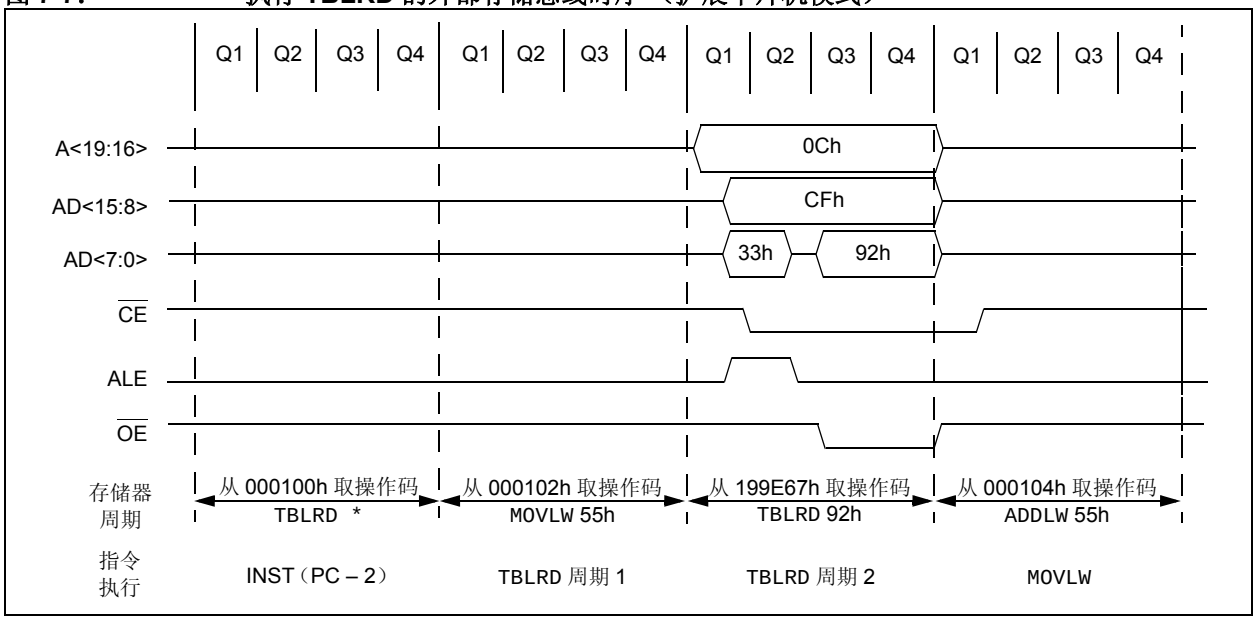
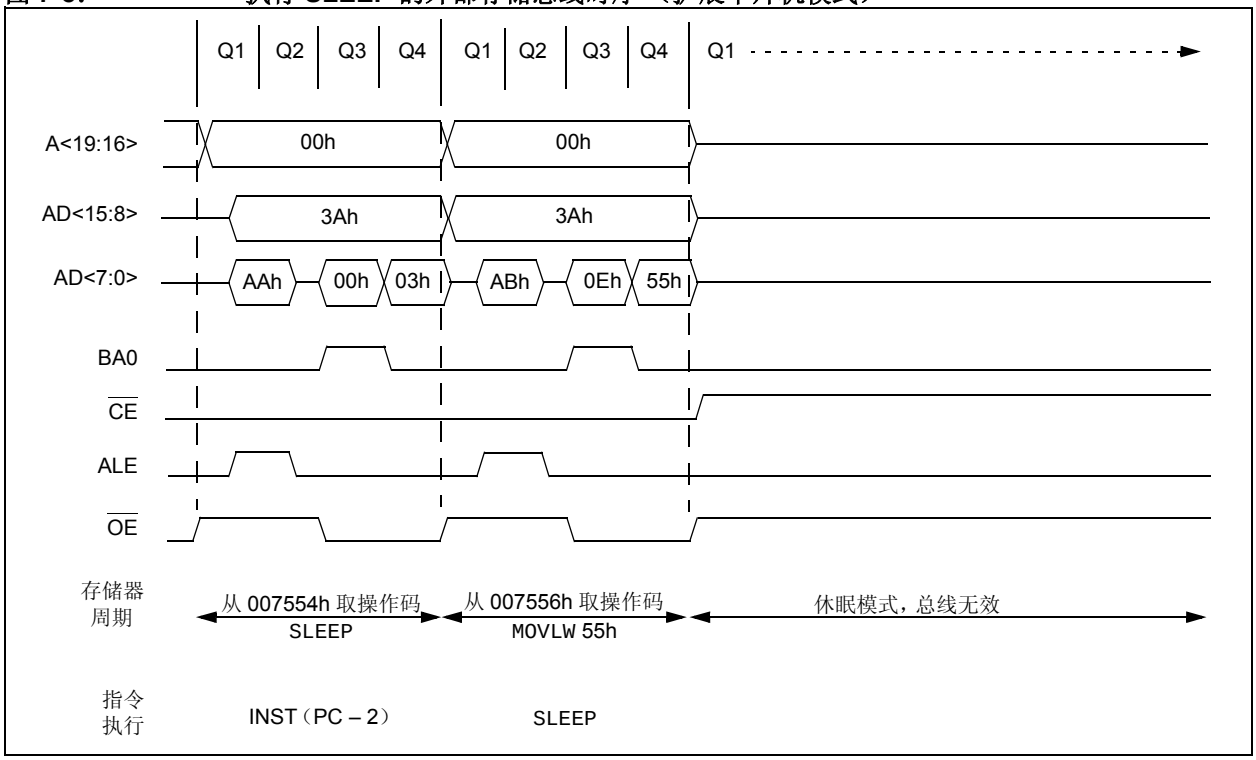


图 7-8： 执行 SLEEP 的外部存储总线时序 （扩展单片机模式）



7.8 工作在功耗管理模式下

在所有功耗管理运行模式下，外部总线都能正常运行。如果选择一个低速的时钟源，总线操作将运行在该速度下。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操作中，可能会导致对外部存储器访问时间过长。如果希望工作在低功耗运行模式下，用户应该在应用中调整在低时钟速度下访问存储器的次数。

在休眠和空闲模式下，单片机内核不需访问数据，总线操作暂停。外部总线的状态被冻结，地址 / 数据引脚和大多数控制引脚保持在进入该模式前的状态。惟一的变化可能是 CE、LB 和 UB 引脚，它们保持为逻辑高电平。

注：

8.0 8 x 8 硬件乘法器

8.1 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位。

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 8-1 给出了各种硬件和软件乘法运算的比较，包括存储器空间和执行时间。

8.2 工作原理

例 8-1 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 8-2 给出了执行 8 x 8 有符号乘法运算的指令序列。要弄清参数的符号位，必须检查每个乘数的最高有效位（MSb），并做相应的减法。

例 8-1： 8 x 8 无符号乘法程序

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

例 8-2： 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 8-1： 各种乘法运算的性能比较

程序	乘法方法	程序存储器空间（字）	周期数（最大）	时间		
				40 MHz 时	10 MHz 时	4 MHz 时
8 x 8 无符号	软件乘法	13	69	6.9 µs	27.6 µs	69 µs
	硬件乘法	1	1	100 ns	400 ns	1 µs
8 x 8 有符号	软件乘法	33	91	9.1 µs	36.4 µs	91 µs
	硬件乘法	6	6	600 ns	2.4 µs	6 µs
16 x 16 无符号	软件乘法	21	242	24.2 µs	96.8 µs	242 µs
	硬件乘法	28	28	2.8 µs	11.2 µs	28 µs
16 x 16 有符号	软件乘法	52	254	25.4 µs	102.6 µs	254 µs
	硬件乘法	35	40	4.0 µs	16.0 µs	40 µs

例 8-3给出了一个16 x 16无符号乘法运算的指令序列。公式 8-1 为所使用的算法。32 位结果存储在4 个寄存器（RES3:RES0）中。

公式 8-1： 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 8-3： 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 c
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
```

例 8-4 给出了 16 x 16 有符号乘法的指令序列。公式 8-2 为所使用的算法。 32 位结果存储在 4 个寄存器（RES3:RES0）中。要弄清乘数的符号位，必须检查每个乘数的最高有效位 （MSb），并做相应的减法。

公式 8-2： 16 x 16 有符号乘法算法

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

例 8-4： 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
   :
```

9.0 中断

PIC18F87J10 系列的器件具有多个中断源及一个中断优先级功能，可以给大多数中断源分配一个高优先级或者低优先级。高优先级中断向量地址为 0008h，低优先级中断向量地址为 0018h。高优先级中断事件将中断所有可能正在处理的低优先级中断。

有 13 个寄存器用于控制中断操作。这些寄存器是：

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

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

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
- **使能位**当标志位置 1 时，使程序执行跳转到中断向量地址
- **优先级位**用于选择是高优先级还是低优先级

通过将 IPEN 位 （RCON<7>）置 1，可使能中断优先级功能。当使能中断优先级时，有2位可允许全局中断。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>）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、允许位以及相应的全局中断允许位均被置 1 时，中断将立即转到地址 0008h 或 0018h，转到哪个地址取决于优先级位的设置。通过对应的允许位可以禁止各个中断。

当 IPEN 位清零 （默认状态）时，便会禁止中断优先级功能，并且中断与 PICmicro® 中档器件相兼容。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，它可使能 / 禁止所有的外设中断源。 INTCON<7> 是 GIE 位，它可允许 / 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。如果使用中断优先级，这个位将是 GIEH 位或者 GIEL 位。高优先级中断源会中断低优先级中断。处理高优先级中断时，低优先级中断将不被处理。

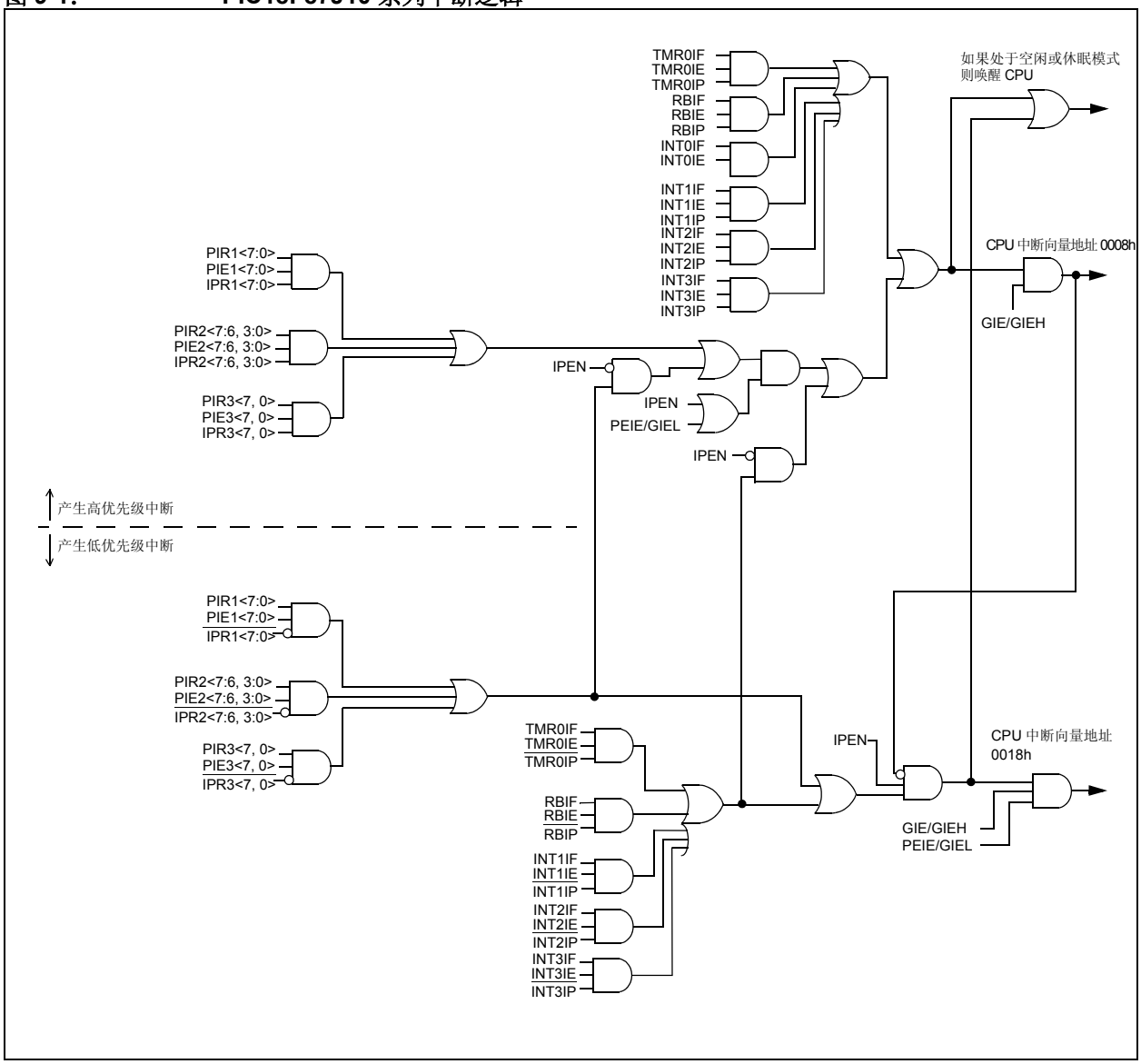
返回地址压入堆栈， PC 中装入中断向量地址 （0008h 或 0018h）。进入中断服务程序之后，就可以通过查询中断标志位来确定中断源。在重新允许中断前，必须用软件将中断标志位清零，以避免重复响应该中断。

执行 “从中断返回” 指令 RETFIE 退出中断程序并将 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 INT 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：	当允许**任何**中断时，不要使用 MOVFF 指令来修改任何中断控制寄存器。否则可能引起单片机执行出错。

图 9-1：　　PIC18F87J10 系列中断逻辑



9.1 INTCON 寄存器

INTCON 寄存器是可读写的寄存器，包含多个使能位、优先级位和标志位。

注： 当一个中断发生时，不管对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。用户软件应在使能一个中断之前，确保先将相应的中断标志位清零。中断标志位可以用软件查询。

寄存器 9-1： 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
bit 7							bit 0

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 = 允许所有低优先级的外设中断
0 = 禁止所有低优先级的外设中断

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

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

bit 3 **RBIE**：RB 端口电平变化中断允许位
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 寄存器已经溢出 （必须用软件清零）
0 = TMR0 寄存器未溢出

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

bit 0 **RBIF**：RB 端口电平变化中断标志位
1 = RB7:RB4 引脚中至少有一个改变了状态 （必须用软件清零）
0 = RB7:RB4 引脚的状态都没有改变

注： 引脚上电平不匹配的情况会一直不断地将此位置 1。而对 PORTB 进行读操作，将结束引脚电平不匹配的情况，并允许该位清零。

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

寄存器 9-2：　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

bit 7　**$\overline{\text{RBPU}}$**：PORTB 上拉使能位
1 = 禁止所有 PORTB 上拉
0 = 按各个端口锁存值使能 PORTB 上拉

bit 6　**INTEDG0**：外部中断 0 边沿选择位
1 = 上升沿时中断
0 = 下降沿时中断

bit 5　**INTEDG1**：外部中断 1 边沿选择位
1 = 上升沿时中断
0 = 下降沿时中断

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

bit 3　**INTEDG3**：外部中断 3 边沿选择位
1 = 上升沿时中断
0 = 下降沿时中断

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

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

注：　当一个中断发生时，不管对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。用户软件应在使能一个中断之前，确保先将相应的中断标志位清零。中断标志位可以用软件查询。

寄存器 9-3：　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

bit 7　**INT2IP**：INT2 外部中断优先级位
1 = 高优先级
0 = 低优先级

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

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

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

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

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

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

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

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

注：　当一个中断发生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置位。用户软件应在使能一个中断之前，确保先将相应的中断标志位清零。中断标志位可以用软件查询。

9.2 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 3 个外设中断请求（标志）寄存器（PIR1、PIR2 和 PIR3）。

注 1： 当有中断条件产生时，不管对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，确保先将相应的中断标志位清零；同时在响应该中断后，也应该将相应的中断标志位清零。

寄存器 9-4： 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	CCP1IF	TMR2IF	TMR1IF
bit 7							bit 0

bit 7 **PSPIF**：并行从动端口读 / 写中断标志位
1 = 发生了读 / 写操作 （必须用软件清零）
0 = 未发生读 / 写操作

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

bit 5 **RC1IF**：EUSART1 接收中断标志位
1 = EUSART1 接收缓冲器 RCREGx 已满 （当读取 RCREGx 时清零）
0 = EUSART1 接收缓冲器为空

bit 4 **TX1IF**：EUSART1 发送中断标志位
1 = EUSART1 发送缓冲器 TXREGx 为空 （当写入 TXREGx 时清零）
0 = EUSART1 发送缓冲器已满

bit 3 **SSP1IF**：主控同步串行端口 1 中断标志位
1 = 完成发送 / 接收 （必须用软件清零）
0 = 等待发送 / 接收

bit 2 **CCP1IF**：ECCP1 中断标志位
捕获模式：
1 = 发生了 TMR1/TMR3 寄存器捕获（必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕获
比较模式：
1 = 发生了与 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生与 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

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

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

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

寄存器 9-5： **PIR2：外设中断请求 （标志）寄存器 2**

R/W-0	R/W-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0
OSCFIF	CMIF	—	—	BCL1IF	—	TMR3IF	CCP2IF
bit 7							bit 0

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 INTRC 提供时钟输入 （必须用软件清零）
0 = 器件时钟正常运行

bit 6 **CMIF**：比较器中断标志位
1 = 比较器输入已改变 （必须用软件清零）
0 = 比较器输入未改变

bit 5-4 **未用**：读为 0

bit 3 **BCL1IF**：总线冲突中断标志位 （MSSP1 模块）
1 = 发生了总线冲突 （必须用软件清零）
0 = 未发生总线冲突

bit 2 **未用**：读为 0

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

bit 0 **CCP2IF**：ECCP2 中断标志位
捕获模式：
1 = 发生了 TMR1/TMR3 寄存器捕获 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕获
比较模式：
1 = 发生了与 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生与 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

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

寄存器 9-6： **PIR3：外设中断请求 （标志）寄存器 3**

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF
bit 7							bit 0

bit 7 **SSP2IF**：主控同步串行端口 2 中断标志位
1 = 完成发送 / 接收 （必须用软件清零）
0 = 等待发送 / 接收

bit 6 **BCL2IF**：总线冲突中断标志位 （MSSP2 模块）
1 = 发生了总线冲突 （必须用软件清零）
0 = 未发生总线冲突

bit 5 **RC2IF**：EUSART2 接收中断标志位
1 = EUSART2 接收缓冲器 RCREGx 已满 （当读取 RCREGx 时清零）
0 = EUSART2 接收缓冲器为空

bit 4 **TX2IF**：EUSART2 发送中断标志位
1 = EUSART2 发送缓冲器 TXREGx 为空 （当写入 TXREGx 时清零）
0 = EUSART2 发送缓冲器已满

bit 3 **TMR4IF**：TMR4 与 PR4 匹配中断标志位
1 = TMR4 与 PR4 匹配（必须用软件清零）
0 = TMR4 与 PR4 不匹配

bit 2 **CCP5IF**：CCP5 中断标志位
捕获模式：
1 = 发生了 TMR1/TMR3 寄存器捕获 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕获
比较模式：
1 = 发生了与 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生与 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

bit 1 **CCP4IF**：CCP4 中断标志位
捕获模式：
1 = 发生了 TMR1/TMR3 寄存器捕获 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕获
比较模式：
1 = 发生了与 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生与 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

bit 0 **CCP3IF**：ECCP3 中断标志位
捕获模式：
1 = 发生了 TMR1/TMR3 寄存器捕获 （必须用软件清零）
0 = 未发生 TMR1/TMR3 寄存器捕获
比较模式：
1 = 发生了与 TMR1/TMR3 寄存器的比较匹配 （必须用软件清零）
0 = 未发生与 TMR1/TMR3 寄存器的比较匹配
PWM 模式：
在此模式下未使用。

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

9.3 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 3 个外设中断允许寄存器（PIE1、PIE2 和 PIE3）。当 IPEN=0 时，要允许任何外设中断就必须将 PEIE 位置 1。

寄存器 9-7：　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	CCP1IE	TMR2IE	TMR1IE
bit 7							bit 0

bit 7　**PSPIE**：并行从动端口读 / 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**：EUSART1 接收中断允许位
1 = 允许 EUSART1 接收中断
0 = 禁止 EUSART1 接收中断

bit 4　**TX1IE**：EUSART1 发送中断允许位
1 = 允许 EUSART1 发送中断
0 = 禁止 EUSART1 发送中断

bit 3　**SSP1IE**：主控同步串行端口 1 中断允许位
1= 允许 MSSP1 中断
0= 禁止 MSSP1 中断

bit 2　**CCP1IE**：ECCP1 中断允许位
1= 允许 ECCP1 中断
0= 禁止 ECCP1 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

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

寄存器 9-8：　PIE2：外设中断允许寄存器 2

R/W-0	R/W-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0
OSCFIE	CMIE	—	—	BCL1IE	—	TMR3IE	CCP2IE
bit 7							bit 0

bit 7　**OSCFIE**：振荡器故障中断允许位
1 = 允许
0 = 禁止

bit 6　**CMIE**：比较器中断允许位
1 = 允许
0 = 禁止

bit 5-4　**未用**：读为 0

bit 3　**BCL1IE**：总线冲突中断允许位 （MSSP1 模块）
1 = 允许
0 = 禁止

bit 2　**未用**：读为 0

bit 1　**TMR3IE**：TMR3 溢出中断允许位
1 = 允许
0 = 禁止

bit 0　**CCP2IE**：ECCP2 中断允许位
1 = 允许
0 = 禁止

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

寄存器 9-9：　PIE3：外设中断允许寄存器 3

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE
bit 7							bit 0

bit 7　**SSP2IE**：主控同步串行端口 2 中断允许位
1 = 允许
0 = 禁止

bit 6　**BCL2IE**：总线冲突中断允许位 （MSSP2 模块）
1 = 允许
0 = 禁止

bit 5　**RC2IE**：EUSART2 接收中断允许位
1 = 允许
0 = 禁止

bit 4　**TX2IE**：EUSART2 发送中断允许位
1 = 允许
0 = 禁止

bit 3　**TMR4IE**：TMR4 与 PR4 匹配中断允许位
1 = 允许
0 = 禁止

bit 2　**CCP5IE**：CCP5 中断允许位
1 = 允许
0 = 禁止

bit 1　**CCP4IE**：CCP4 中断允许位
1 = 允许
0 = 禁止

bit 0　**CCP3IE**：ECCP3 中断允许位
1 = 允许
0 = 禁止

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

9.4 IPR 寄存器

IPR 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 3 个外设中断优先级寄存器（IPR1、IPR2 和 IPR3）。使用优先级位要求将中断优先级使能（IPEN）位置 1。

寄存器 9-10： 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	CCP1IP	TMR2IP	TMR1IP
bit 7							bit 0

bit 7 **PSPIP**：并行从动端口读 / 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**：EUSART1 接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 4 **TX1IP**：EUSART1 发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 3 **SSP1IP**：主控同步串口 1 中断优先级位
1 = 高优先级
0 = 低优先级

bit 2 **CCP1IP**：ECCP1 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

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

寄存器 9-11： **IPR2：外设中断优先级寄存器 2**

R/W-1	R/W1	U-0	U-0	R/W-1	U-0	R/W-1	R/W-1
OSCFIP	CMIP	—	—	BCL1IP	—	TMR3IP	CCP2IP
bit 7							bit 0

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 **CMIP**：比较器中断优先级位
1 = 高优先级
0 = 低优先级

bit 5-4 **未用**：读为 0

bit 3 **BCL1IP**：总线冲突中断优先级位 （MSSP1 模块）
1 = 高优先级
0 = 低优先级

bit 2 **未用**：读为 0

bit 1 **TMR3IP**：TMR3 溢出中断优先级位
1 = 高优先级
0 = 低优先级

bit 0 **CCP2IP**：ECCP2 中断优先级位
1 = 高优先级
0 = 低优先级

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

寄存器 9-12：　IPR3：外设中断优先级寄存器 3

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP
bit 7							bit 0

bit 7　**SSP2IP**：主控同步串行端口 2 中断优先级位
1 = 高优先级
0 = 低优先级

bit 6　**BCL2IP**：总线冲突中断优先级位 （MSSP2 模块）
1 = 高优先级
0 = 低优先级

bit 5　**RC2IP**：EUSART2 接收中断优先级位
1 = 高优先级
0 = 低优先级

bit 4　**TX2IP**：EUSART2 发送中断优先级位
1 = 高优先级
0 = 低优先级

bit 3　**TMR4IE**：TMR4 与 PR4 匹配中断优先级位
1 = 高优先级
0 = 低优先级

bit 2　**CCP5IP**：CCP5 中断优先级位
1 = 高优先级
0 = 低优先级

bit 1　**CCP4IP**：CCP4 中断优先级位
1 = 高优先级
0 = 低优先级

bit 0　**CCP3IP**：ECCP3 中断优先级位
1 = 高优先级
0 = 低优先级

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

9.5 RCON 寄存器

RCON寄存器包含的几个位可以用来确定器件上次复位或从空闲或休眠模式被唤醒的原因。RCON 还包含中断优先级使能位 （IPEN）。

寄存器 9-13： RCON：复位控制寄存器

R/W-0	U-0	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

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 6-5 **未用**：读为 0

bit 4 $\overline{\textbf{RI}}$：RESET 指令标志位
关于位操作的详细信息，请参见寄存器 4-1。

bit 3 $\overline{\textbf{TO}}$：看门狗定时器超时溢出标志位
关于位操作的详细信息，请参见寄存器 4-1。

bit 2 $\overline{\textbf{PD}}$：掉电检测标志位
关于位操作的详细信息，请参见寄存器 4-1。

bit 1 $\overline{\textbf{POR}}$：上电复位状态位
关于位操作的详细信息，请参见寄存器 4-1。

bit 0 $\overline{\textbf{BOR}}$：欠压复位状态位
关于位操作的详细信息，请参见寄存器 4-1。

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

9.6 INTn 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，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 INTxIF 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中断标志位 INTxIF 清零。

如果 INTxIE 位在进入功耗管理模式前被置 1，则所有的外部中断（INT0、INT1、INT2 和 INT3）均能将处理器从功耗管理模式唤醒。如果全局中断允许位 GIE 被置 1，则处理器将在被唤醒之后跳转到中断向量处执行程序。

INT1、INT2 和 INT3 的中断优先级由中断优先级位 INT1IP（INTCON3<6>）、INT2IP（INTCON3<7>）和 INT3IP（INTCON2<1>）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。

9.7 TMR0 中断

在 8 位模式（默认模式）下，TMR0 寄存器的溢出（FFh → 00h）会使 TMR0IF 标志位置 1。在 16 位模式下，TMR0H:TMR0L 寄存器对的溢出（FFFFh → 0000h）会使 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>）决定。欲进一步了解 Timer0 模块的详细内容，请参见**第 11.0 节 “Timer0 模块”**。

9.8 PORTB 电平变化中断

PORTB<7:4> 上的一次输入电平变化，会将标志位 RBIF（INTCON<0>）置 1。通过将使能位 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>）决定。

9.9 中断的现场保护

在中断期间，将返回的 PC 地址压入堆栈。另外，将 WREG、STATUS 以及 BSR 寄存器的值压入快速返回堆栈。如果未使用从中断快速返回（见**第 5.3 节 “数据存储器构成”**），用户可能需要在进入中断服务程序前，保存 WREG、STATUS 以及 BSR 寄存器的值。根据用户的应用，还可能需要保存其他寄存器的值。例 9-1 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 9-1： 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
```

10.0 I/O 端口

根据选定的器件和使能的功能，最多有九个可用端口。I/O 端口的一些引脚与来自器件外设的一些备用功能复用。通常，当外设使能时，其对应的引脚就可能无法作为通用 I/O 引脚使用。

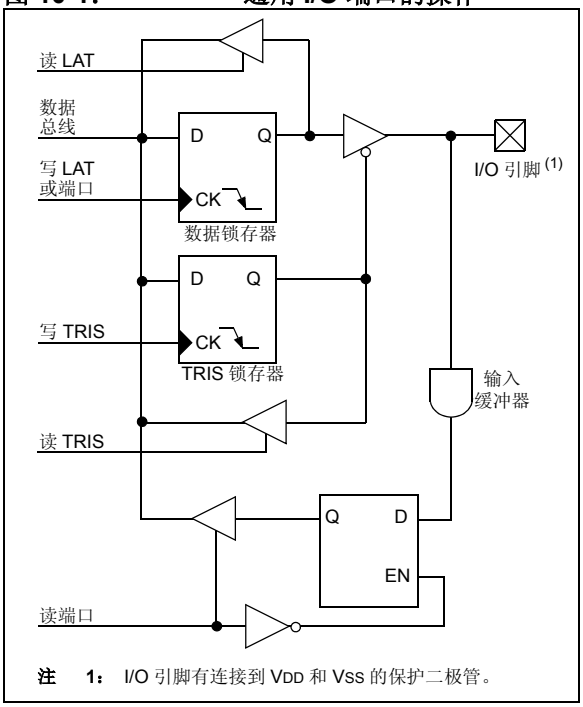
每个端口都有三个寄存器进行操作。这些寄存器是：

- 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）

数据锁存器 （LAT 寄存器）对驱动的 I/O 引脚值进行读 - 修改 - 写操作是非常有用的。

图 10-1 所示为通用 I/O 端口的简化模型，它没有到其他外设的接口。

图 10-1： 通用 I/O 端口的操作



注 1： I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

10.1 I/O 端口引脚功能

在开发应用程序时，必须考虑到端口引脚的功能。某些引脚上的输出驱动电平比其他引脚要高。同样，某些引脚可以接受高于 VDD 的输入电平。

10.1.1 引脚输出驱动电平

满足不同应用要求的引脚组的输出引脚驱动能力是不同的。PORTB 和 PORTC 是为驱动较高的负载而设计的，比如 LED。外部存储器接口端口（PORTD、PORTE 和 PORTJ）是为驱动中等的负载而设计的。除此之外，所有的其他端口都是为驱动较小的负载而设计的，通常只用作指示作用。表 10-1 汇总了输出功能。更多详情请参见**第 26.0 节 “电气规范”**。

表 10-1： 输出驱动电平

端口	驱动	说明
PORTA	低	用作指示作用。
PORTF		
PORTG		
PORTH(1)		
PORTD	中	足够高的驱动电平，用于外部存储器接口以及指示作用。
PORTE		
PORTJ(1)		
PORTB	高	适合直接输出 LED 驱动电平。
PORTC		

注 1： 此端口在 64 引脚器件上不可用。

10.1.2 输入引脚和电压注意事项

用作器件输入的引脚的电压容差取决于该引脚的输入功能。仅用做数字输入的引脚能够接受高达 5.5V 的直流电压，这个电压值是数字逻辑电路的典型电压值。相反，具有模拟输入功能的引脚只可以接受最高为 VDD 的电压值。应该在这些引脚上避免施加超过 VDD 的电压。表 10-2 汇总了这些输入功能。更多详情请参见**第 26.0 节 “电气规范”**。

表 10-2： 输入电平

端口或引脚	可接受的最高输入电平	说明
PORTA<5:0>	VDD	只能接受小于 VDD 输入电压。
PORTC<1:0>		
PORTF<6:1>		
PORTH<7:4>(1)		
PORTB<7:0>	5.5V	可接受的输入电压在 VDD 以上，对于大部分标准逻辑电路很有用处。
PORTC<7:2>		
PORTD<7:0>		
PORTE<7:0>		
PORTF<7>		
PORTG<4:0>		
PORTH<3:0>(1)		
PORTJ<7:0>(1)		

注 1： 此端口在 64 引脚器件上不可用。

10.2 PORTA、TRISA 和 LATA 寄存器

PORTA 是 6 位宽的双向端口。相应的数据方向寄存器为 TRISA。置位 TRISA （= 1）可以让相应 PORTA 引脚作为输入引脚（即将相应的输出驱动器置于高阻态模式）。清零 TRISA 位 （= 0）将使相应的 PORTA 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。

读 PORTA 寄存器就是读引脚状态，而写该寄存器就是写入端口锁存器。

数据锁存器（LATA）也是映射的存储器。LATA 寄存器上的读 - 修改 - 写操作将读写 PORTA 的锁存输出值。

RA4 引脚与 Timer0 模块时钟输入引脚复用为 RA4/T0CKI 引脚。其他 PORTA 引脚与模拟 VREF+ 和 VREF- 输入引脚复用。通过清零或置位 ADCON1 寄存器中的 PCFG3:PCFG0 控制位选择将 RA5:RA0 引脚作为 A/D 转换器输入引脚工作。

> **注：** RA5 和 RA3:RA0 在任何复位时被配置为模拟输入引脚并读为 0。 RA4 被配置为数字输入引脚。

RA4/T0CKI 引脚是施密特触发器输入。而其他的 PORTA 引脚具有 TTL 输入电平和完全的 CMOS 输出驱动器。

TRISA 寄存器控制 PORTA 引脚的方向，甚至是当它们用作模拟输入的时候。当它们用作模拟输入时，用户必须确保 TRISA 寄存器中的位保持为置位状态。

例 10-1： 初始化 PORTA

```
CLRF    PORTA   ; Initialize PORTA by
                ; clearing output
                ; data latches
CLRF    LATA    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   07h     ; Configure A/D
MOVWF   ADCON1  ; for digital inputs
MOVWF   07h     ; Configure comparators
MOVWF   CMCON   ; for digital input
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISA   ; Set RA<3:0> as inputs
                ; RA<5:4> as outputs
```

表 10-3： PORTA 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RA0/AN0	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 时为默认输入配置；不影响数字输出。
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 时为默认输入配置；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。当使能 CVREF 输出时禁止。
		1	I	TTL	PORTA<2> 数据输入。当使能模拟功能时禁止；当使能 CVREF 输出时禁止。
	AN2	1	I	ANA	A/D 输入通道 2 和比较器 C2+ 输入。 POR 时为默认输入配置，不受模拟输出的影响。
	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。在 POR 时为默认输入配置。
	VREF+	1	I	ANA	A/D 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 时为默认配置。
	T0CKI	x	I	ST	Timer0 时钟输入。
RA5/AN4	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。在 POR 时为默认配置。

图注： PWR = 电源， O = 输出， I = 输入， ANA = 模拟信号，DIG = 数字输出， ST = 施密特缓冲输入，
TTL = TTL 缓冲输入， x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

表 10-4： 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	复位值所在页
PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	52
LATA	—	—	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	52
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	52
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	50

图注： — = 未用位，读为 0。 PORTA 不使用阴影单元。

10.3 PORTB、 TRISB 和 LATB 寄存器

PORTB 是 8 位宽的双向端口。它对应的数据方向寄存器是 TRISB。置位 TRISB （= 1）可以让相应 PORTB 引脚作为输入引脚（即将相应的输出驱动器置于高阻态模式）。清零 TRISB 位 （= 0）将使相应的 PORTB 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。PORTB 上的所有引脚都仅仅是数字的，并且可以接受高达 5.5V 的电压。

数据锁存器（LATB）也是映射的存储器。LATB 寄存器上的读 - 修改 - 写操作将读写 PORTB 的锁存输出值。

例 10-2： 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。单个控制位可以开启所有上拉电路。可以通过清零 RBPU 位（INTCON2<7>）来导通上拉电路。当端口引脚配置为输出时，其弱上拉电路会自动切断。此弱上拉功能在上电复位时被禁止。

PORTB 的 4 个引脚（RB7:RB4）都有电平变化中断功能。只有配置为输入的引脚会导致中断发生（即当 RB7:RB4 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比较功能）。当前 RB7:RB4 输入引脚上的电平与上次读 PORTB 时锁存的旧值进行比较。RB7:RB4 输出的 “不匹配” 值一起进行 “或” 运算，产生 RB 端口电平变化中断，使用标志位 RBIF（INTCON<0>）表示。

此中断可以将器件从功耗管理模式唤醒。用户可用以下方式在中断服务程序中清除该中断：

a）读或写 PORTB （使用 MOVFF （ANY）， PORTB 指令时除外）。这将结束不匹配状态。

b）清零标志位 RBIF。

不匹配状态会一直不断地将 RBIF 标志位置 1。而读 PORTB 将结束不匹配状态，并且允许将 RBIF 标志位清零。

对于按键唤醒以及其他仅使用 PORTB 的电平变化触发中断功能的操作，建议使用此电平变化触发中断来实现。在使用电平变化触发中断功能时，建议不要查询 PORTB 的状态。

对于 80 引脚的器件来说，通过清零 CCP2MX 配置位可将 RB3 配置为 ECCP2 模块和增强型 PWM 输出引脚 2A 的备用外设引脚。这种配置仅应用于在扩展单片机模式下工作的 80 引脚器件。如果此器件处于单片机模式，ECCP2 的备用分配为 RE7。根据其他 ECCP2 配置，用户必须确保为要进行的操作将 TRISB<3> 位进行相应的设置。

表 10-5： 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 $\overline{\text{RBPU}}$ 位被清零时为弱上拉。
	INT0	1	I	ST	外部中断 0 输入。
	FLT0	1	I	ST	增强型 PWM 错误输入 （ECCP1 模块）；用软件使能。
RB1/INT1	RB1	0	O	DIG	LATB<1> 数据输出。
		1	I	TTL	PORTB<1> 数据输入；当 $\overline{\text{RBPU}}$ 位被清零时为弱上拉。
	INT1	1	I	ST	外部中断 1 输入。
RB2/INT2	RB2	0	O	DIG	LATB<2> 数据输出。
		1	I	TTL	PORTB<2> 数据输入；当 $\overline{\text{RBPU}}$ 位被清零时为弱上拉。
	INT2	1	I	ST	外部中断 2 输入。
RB3/INT3/ ECCP2/P2A	RB3	0	O	DIG	LATB<3> 数据输出。
		1	I	TTL	PORTB<3> 数据输入；当 $\overline{\text{RBPU}}$ 位被清零时为弱上拉。
	INT3	1	I	ST	外部中断 3 输入。
	ECCP2(1)	0	O	DIG	CCP2 比较输出和 CCP2 PWM 输出；优先级高于端口数据。
		1	I	ST	CCP2 捕捉输入。
	P2A(1)	0	O	DIG	ECCP2 增强型 PWM 输出，通道 A。可能在增强型 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 $\overline{\text{RBPU}}$ 位被清零时为弱上拉。
	KBI0		I	TTL	引脚电平变化时发生中断。
RB5/KBI1	RB5	0	O	DIG	LATB<5> 数据输出。
		1	I	TTL	PORTB<5> 数据输入；当 $\overline{\text{RBPU}}$ 位被清零时为弱上拉。
	KBI1		I	TTL	引脚电平变化时发生中断。
RB6/KBI2/PGC	RB6	0	O	DIG	LATB<6> 数据输出。
		1	I	TTL	PORTB<6> 数据输入；当 $\overline{\text{RBPU}}$ 位被清零时为弱上拉。
	KBI2	1	I	TTL	引脚电平变化时发生中断。
	PGC	x	I	ST	串行执行 ICSP 的 （ICSP™）时钟输入和 ICS 操作。(2)
RB7/KBI3/PGD	RB7	0	O	DIG	LATB<7> 数据输出。
		1	I	TTL	PORTB<7> 数据输入；当 $\overline{\text{RBPU}}$ 位被清零时为弱上拉。
	KBI3	1	I	TTL	引脚电平变化时发生中断。
	PGD	x	O	DIG	执行 ICSP 和 ICD 操作时串行输出数据。(2)
		x	I	ST	执行 ICSP 和 ICD 操作时串行输入数据。(2)

图注： PWR = 电源，O = 输出，I = 输入，ANA = 模拟信号，DIG = 数字输出，ST = 施密特缓冲输入，TTL = TTL 缓冲输入，x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

注 1： 当 CCP2MX 配置位被清零时， ECCP2/P2A 的备用配置 （扩展单片机模式，仅用于 80 引脚器件）。 RC1 为默认配置。

2： 当使能 ICSP 或 ICD 时所有其他引脚功能都被禁止。

表 10-6： 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	复位值所在页
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	52
LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	52
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	52
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	RBIP	49
INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF	49

图注： PORTB 不使用阴影单元。

10.4 PORTC、 TRISC 和 LATC 寄存器

PORTC 是 8 位宽的双向端口。它对应的数据方向寄存器是 TRISIC。置位 TRISC （= 1）可以让相应 PORTC 引脚作为输入引脚（即将相应的输出驱动器置于高阻态模式）。清零 TRISC 位 （= 0）将使相应的 PORTC 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。PORTC 引脚 RC2 到 RC7 只能作为数字引脚并且可以接受高达 5.5V 的输入电压。

数据锁存器（LATC）也是映射的存储器。LATC 寄存器上的读 - 修改 - 写操作将读写 PORTC 的锁存输出值。

PORTC 与几种外设功能复用（表 10-7）。引脚有施密特触发输入缓冲器。通常由配置位 CCP2MX 将 RC1 配置为 ECCP2 模块的默认外设引脚和增强型 PWM 输出 P2A 引脚 （默认状态， CCP2MX = 1）。

当外设功能使能时，应考虑到每个 PORTC 引脚的 TRIS 位方向设置。有些外设使能时，会覆盖相应引脚的 TRIS 位方向设置而将引脚直接定义为输出，而另一些外设使能时，也会覆盖相应引脚的 TRIS 方向设置，但将引脚直接定义为输入。用户应该参考相应的外设章节来正确设置 TRIS 位。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

外设覆盖会影响 TRISC 寄存器的内容。尽管外设器件可能会覆盖一个或多个引脚，读 TRISC 总是会返回当前内容。

例 10-3： 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 10-7： PORTC 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RC0/T1OSO/ T13CKI	RC0	0	O	DIG	LATC<0> 数据输出。
		1	I	ST	PORTC<0> 数据输入。
	T1OSO	x	O	ANA	Timer1 振荡器输出；当 Timer1 振荡器使能时被使能。禁止数字 I/O。
	T13CKI	1	I	ST	Timer1/Timer3 计数器输入。
RC1/T1OSI/ ECCP2/P2A	RC1	0	O	DIG	LATC<1> 数据输出。
		1	I	ST	PORTC<1> 数据输入。
	T1OSI	x	I	ANA	Timer1 振荡器输入；当 Timer1 振荡器使能时被使能。禁止数字 I/O。
	ECCP2(1)	0	O	DIG	CCP2 比较输出和 CCP2 PWM 输出；优先级高于端口数据。
		1	I	ST	CCP2 捕捉输入。
	P2A(1)	0	O	DIG	ECCP2 增强型 PWM 输出，通道 A。可能在增强型 PWM 关闭事件期间被配置为三态。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	CCP1 比较输出和 CCP1 PWM 输出；优先级高于端口数据。
		1	I	ST	CCP1 捕捉输入。
	P1A	0	O	DIG	ECCP1 增强型 PWM 输出，通道 A。可能在增强型 PWM 关闭期间被配置为三态。优先级高于端口数据。
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 （MSSP1 模块）；优先级高于端口数据。
		1	I	ST	SPI 时钟输入 （MSSP1 模块）。
	SCL1	0	O	DIG	I²C™ 时钟输出 （MSSP1 模块）；优先级高于端口数据。
		1	I	ST	I²C 时钟输入 （MSSP1 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	1	I	ST	SPI 数据输入 （MSSP1 模块）。
	SDA1	1	O	DIG	I²C 数据输出 （MSSP1 模块）；优先级高于端口数据。
		1	I	ST	I²C 数据输入 （MSSP1 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 （MSSP1 模块）；优先级高于端口数据。
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 （EUSART1 模块）；优先级高于端口数据。
	CK1	1	O	DIG	同步串行数据输入 （EUSART1 模块）。用户必须将其配置为输入端口。
		1	I	ST	同步串行时钟输入 （EUSART1 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 （EUSART1 模块）。
	DT1	1	O	DIG	同步串行数据输出 （EUSART1 模块）；优先级高于端口数据。
		1	I	ST	同步串行数据输入 （EUSART1 模块）。用户必须将其配置为输入端口。

图注： PWR = 电源，O = 输出，I = 输入，ANA = 模拟信号，DIG = 数字输出，ST = 施密特缓冲输入，TTL = TTL 缓冲输入，x = 任意值 （TRIS 位不影响端口方向或者在此选项中被覆盖）。

注 1： 当 CCP2MX 配置位置 1 时， ECCP2/P2A 为默认配置。

表 10-8： 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	复位值所在页
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	52
LATC	LATC7	LATBC6	LATC5	LATCB4	LATC3	LATC2	LATC1	LATC0	52
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	52

10.5 PORTD、 TRISD 和 LATD 寄存器

PORTD 是 8 位宽的双向端口。它对应的数据方向寄存器是 TRISID。置位 TRISD （= 1）可以让相应 PORTD 引脚作为输入引脚 （即将相应的输出驱动器置于高阻态）。清零 TRISD 位 （= 0）将使相应的 PORTD 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。PORTD 上的所有引脚都仅仅是数字的，并且可以接受高达 5.5V 的电压。

数据锁存器（LATD）也是映射的存储器。LATD 寄存器上的读 - 修改 - 写操作将读写 PORTD 的锁存输出值。

POTD 上的所有引脚使用施密特触发器输入缓冲器实现的。每个引脚都被单独设置为输入或输出。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

在 80 引脚的器件上，PORTD 与系统总线复用作为外部存储器接口的一部分。 I/O 端口和其他功能只有在通过将 EBDIS 位 （MEMCON<7>）位置 1 而禁止接口功能时才可用。当使能接口时， PORTD 是复用的地址 / 数据总线（AD7:AD0）的低位字节。还将覆盖 TRISD 位。

PORTD 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延迟时的输入电平在外部存储器接口的一个已知状态。单个控制位可以关闭所有上拉电路。可以通过清零 RDPU 位 （PORTG<7>）来关闭上拉电路。当端口引脚配置为输出时，其弱上拉电路会自动关闭。在所有器件复位时上拉功能被禁止。

也可以通过将 PSPMODE 控制位 （PSPCON<4>）置 1 来把 PORTD 配置为 8 位宽的并行微处理器端口功能。在此模式中，并行端口数据优先于其他数字 I/O （但外部存储器接口除外）。当激活并行端口时，输入缓冲器为TTL。更多详情，请参见**第10.11节“并行从动端口”**。

例 10-4： 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 10-9： PORTD 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RD0/AD0/PSP0	RD0	0	O	DIG	LATD<0> 数据输出。
		1	I	ST	PORTD<0> 数据输入。
	AD0(2)	x	O	DIG	外部存储器接口，地址 / 数据 bit 0 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 0 输出。(1)
	PSP0		O	DIG	PSP 读输出数据 （LATD<0>）；优先级高于端口数据。
			I	TTL	PSP 写输入数据。
RD1/AD1/PSP1	RD1	0	O	DIG	LATD<1> 数据输出。
		1	I	ST	PORTD<1> 数据输入。
	AD1(2)	x	O	DIG	外部存储器接口，地址 / 数据 bit 1 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 1 输出。(1)
	PSP1	x	O	DIG	PSP 读输出数据 （LATD<1>）；优先级高于端口数据。
		x	I	TTL	PSP 写输入数据。
RD2/AD2/PSP2	RD2	0	O	DIG	LATD<2> 数据输出。
		1	I	ST	PORTD<2> 数据输入。
	AD2(2)	x	O	DIG	外部存储器接口，地址 / 数据 bit 2 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 2 输入。(1)
	PSP2	x	O	DIG	PSP 读输出数据 （LATD<2>）；优先级高于端口数据。
		x	I	TTL	PSP 写输入数据。
RD3/AD3/PSP3	RD3	0	O	DIG	LATD<3> 数据输出。
		1	I	ST	PORTD<3> 数据输入。
	AD3(2)	x	O	DIG	外部存储器接口，地址 / 数据 bit 3 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 3 输入。(1)
	PSP3	x	O	DIG	PSP 读输出数据 （LATD<3>）；优先级高于端口数据。
		x	I	TTL	PSP 写输入数据。
RD4/AD4/ PSP4/SDO2	RD4	0	O	DIG	LATD<4> 数据输出。
		1	I	ST	PORTD<4> 数据输入。
	AD4(2)	x	O	DIG	外部存储器接口，地址 / 数据 bit 4 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 4 输入。(1)
	PSP4	x	O	DIG	PSP 读输出数据 （LATD<4>）；优先级高于端口数据。
		x	I	TTL	PSP 写输入数据。
	SDO2	0	O	DIG	SPI™ 数据输出 （MSSP2 模块）；优先级高于端口数据。
RD5/AD5/ PSP5/SDI2/ SDA2	RD5	0	O	DIG	LATD<5> 数据输出。
		1	I	ST	PORTD<5> 数据输入。
	AD5(2)	x	O	DIG	外部存储器接口，地址 / 数据 bit 5 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 5 输入。(1)
	PSP5	x	O	DIG	PSP 读输出数据 （LATD<5>）；优先级高于端口数据。
		x	I	TTL	PSP 写输入数据。
	SDI2	1	I	ST	SPI 数据输入 （MSSP2 模块）。
	SDA2	1	O	DIG	I^2C™ 数据输出 （MSSP2 模块）；优先级高于端口数据。
		1	I	ST	I^2C 数据输入 （MSSP2 模块）；输入类型取决于模块设置。

图注： PWR = 电源， O = 输出， I = 输入， ANA = 模拟信号，DIG = 数字输出， ST = 施密特缓冲输入，TTL = TTL 缓冲输入，x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

注 1： 外部存储器 I/O 接口优先级高于所有其他数字和 PSP I/O。

2： 仅在 80 引脚的器件上可用。

表 10-9： PORTD 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RD6/AD6/ PSP6/SCK2/ SCL2	RD6	0	O	DIG	LATD<6> 数据输出。
		1	I	ST	PORTD<6> 数据输入。
	AD6(2)	x	O	DIG-3	外部存储器接口，地址 / 数据 bit 6 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 6 输出。(1)
	PSP6	x	O	DIG	PSP 读输出数据 （LATD<6>）；优先级高于端口数据。
		x	I	TTL	PSP 写输入数据。
	SCK1	0	O	DIG	SPI™ 时钟输出 （MSSP2 模块）；优先级高于端口数据。
		1	I	ST	SPI 时钟输入 （MSSP2 模块）。
	SCL1	0	O	DIG	I^2C™ 时钟输出 （MSSP2 模块）；优先级高于端口数据。
		1	I	ST	I^2C 时钟输入 （MSSP2 模块）；输入类型取决于模块设置。
RD7/AD7/ PSP7/$\overline{SS2}$	RD7	0	O	DIG	LATD<7> 数据输出。
		1	I	ST	PORTD<7> 数据输入。
	AD7(2)	x	O	DIG	外部存储器接口，地址 / 数据 bit 7 输出。(1)
		x	I	TTL	外部存储器接口，数据 bit 7 输入。(1)
	PSP7	x	O	DIG	PSP 读输出数据 （LATD<7>）；优先级高于端口数据。
		x	I	TTL	PSP 写输入数据。
	$\overline{SS2}$	x	I	TTL	MSSP 的从动选择输入 （MSSP2 模块）。

图注： PWR = 电源， O = 输出， I = 输入， ANA = 模拟信号，DIG = 数字输出， ST = 施密特缓冲输入，TTL = TTL 缓冲输入， x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

注 1： 外部存储器 I/O 接口优先级高于所有其他数字和 PSP I/O。

2： 仅在 80 引脚的器件上可用。

表 10-10： 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	复位值 所在页
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	52
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	52
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	52
PORTG	RDPU	REPU	RJPU(1)	RG4	RG3	RG2	RG1	RG0	52

图注： PORTD 不使用阴影单元。

注 1： 在 64 引脚器件上未使用，读为 0。

10.6 PORTE， TRISE 和 LATE 寄存器

PORTE 是 7 位宽的双向端口。它对应的数据方向寄存器是 TRISE。置位 TRISE （= 1）可以让相应 PORTE 引脚作为输入引脚（即将相应的输出驱动器置于高阻态模式）。清零 TRISE 位 （= 0）将使相应的 PORTE 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。PORTE 上的所有引脚都仅仅是数字的，并且可以接受高达 5.5V 的电压。

数据锁存器（LATE）也是映射的存储器。LATE 寄存器上的读 - 修改 - 写操作将读写 PORTE 的锁存输出值。

PORTE 上的所有引脚使用施密特触发器输入缓冲器实现的。每个引脚都被单独设置为输入或输出。

注： 这些引脚在任何器件复位时都被配置为数字输入引脚。

在 80 引脚的器件上，PORTE 与系统总线复用作为外部存储器接口的一部分。 I/O 端口和其他功能只有在通过将 EBDIS 位 （MEMCON<7>）位置 1 而禁止接口功能时才可用。当使能接口时，PORTE 是复用的地址 / 数据总线 （AD15:AD8）的高位字节。还将覆盖 TRISE 位。

PORTE 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延迟时的输入电平在外部存储器接口的一个已知状态。单个控制位可以关闭所有上拉电路。可以通过清零 REPU 位 （PORTG<6>）来关闭上拉电路。当端口引脚配置为输出时，其弱上拉电路会自动关闭。发生任何器件复位时，弱上拉功能会被禁止。

PORTE 也可与以下这些功能复用：ECCP1 模块和 ECCP3 的增强型 PWM 输出 B 和 C 以及 ECCP2 模块的输出 B、 C 和 D。对于所有器件来说，它们的默认分配是在PORTE<6:3>上。在80引脚器件上，由ECCPMX 配置位控制 ECCP1 和 ECCP3 的输出复用。清零这些位将把 P1B/P1C 和 P3B/P3C 的输出重新分配到 PORTH。

对于在单片机模式工作的器件，可以将引脚 RE7 配置为 ECCP2 模块和增强型 PWM 输出 2A 的备用外设引脚。可以通过清零 CCP2MX 配置位来完成以上配置。

当 PORTD 上的并行从动端口为激活状态时，三个 PORTE 引脚（RE0、RE1 和 RE2）将被配置为端口的数字控制输入引脚。表 10-11 中汇总了此控制功能。当 PSPMODE 控制位（PSPCON<4>）被置 1 时，会自动重新配置。用户还必须确定将相应的 TRISE 位置 1 以便将这些引脚配置为数字输入。

例 10-5： 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 10-11： PORTE 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE0/AD8/$\overline{RD}$/ P2D	RE0	0	O	DIG	LATE<0> 数据输出。
		1	I	ST	PORTE<0> 数据输入。
	AD8(3)	x	O	DIG	外部存储器接口，地址 / 数据 bit 8 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 8 输入。(2)
	$\overline{RD}$	1	I	TTL	并行从动端口读使能控制输入。
	P2D	0	O	DIG	ECCP2 增强型 PWM 输出，通道 D；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。
RE1/AD9/$\overline{WR}$/ P2C	RE1	0	O	DIG	LATE<1> 数据输出。
		1	I	ST	PORTE<1> 数据输入。
	AD9(3)	x	O	DIG	外部存储器接口，地址 / 数据 bit 9 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 9 输出。(2)
	$\overline{WR}$	1	I	TTL	并行从动端口写使能控制输入。
	P2C	0	O	DIG	ECCP2 增强型 PWM 输出，通道 C；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。
RE2/AD10/$\overline{CS}$/ P2B	RE2	0	O	DIG	LATE<2> 数据输出。
		1	I	ST	PORTE<2> 数据输入。
	AD10(3)	x	O	DIG	外部存储器接口，地址 / 数据 bit 10 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 10 输入。(2)
	$\overline{CS}$	1	I	TTL	并行从动端口片选控制输入。
	P2B	0	O	DIG	ECCP2 增强型 PWM 输出，通道 B；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。
RE3/AD11/ P3C	RE3	0	O	DIG	LATE<3> 数据输出。
		1	I	ST	PORTE<3> 数据输入。
	AD11(3)	x	O	DIG	外部存储器接口，地址 / 数据 bit 11 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 11 输入。(2)
	P3C(1)	0	O	DIG	ECCP3 增强型 PWM 输出，通道 C；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。
RE4/AD12/ P3B	RE4	0	O	DIG	LATE<4> 数据输出。
		1	I	ST	PORTE<4> 数据输入。
	AD12(3)	x	O	DIG	外部存储器接口，地址 / 数据 bit 12 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 12 输入。(2)
	P3B(1)	0	O	DIG	ECCP3 增强型 PWM 输出，通道 B；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。
RE5/AD13/ P1C	RE5	0	O	DIG	LATE<5> 数据输出。
		1	I	ST	PORTE<5> 数据输入。
	AD13(3)	x	O	DIG	外部存储器接口，地址 / 数据 bit 13 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 13 输入。(2)
	P1C(1)	0	O	DIG	ECCP1 增强型 PWM 输出，通道 C；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。

图注： PWR = 电源， O = 输出， I = 输入， ANA = 模拟信号，DIG = 数字输出， ST = 施密特缓冲输入，TTL = TTL 缓冲输入，x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

注 1： 当 ECCPMX 配置位置 1 时， P1B/P1C 和 P3B/P3C 的默认配置 （仅用于 80 引脚器件）。

2： 外部存储器 I/O 接口优先级高于所有其他数字和 PSP I/O。

3： 仅在 80 引脚的器件上可用。

4： 当 CCP2MX 配置位被清零时， ECCP2/P2A 的备用配置 （单片机模式中的所有器件适用）。

表 10-11： PORTE 功能 （续）

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RE6/AD14/ P1B	RE6	0	O	DIG	LATE<6> 数据输出。
		1	I	ST	PORTE<6> 数据输入。
	AD14(3)	x	O	DIG	外部存储器接口，地址 / 数据 bit 14 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 14 输入。(2)
	P1B(1)	0	O	DIG	ECCP1 增强型 PWM 输出，通道 B ；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。
RE7/AD15/ ECCP2/P2A	RE7	0	O	DIG	LATE<7> 数据输出。
		1	I	ST	PORTE<7> 数据输入。
	AD15(3)	x	O	DIG	外部存储器接口，地址 / 数据 bit 15 输出。(2)
		x	I	TTL	外部存储器接口，数据 bit 15 输入。(2)
	ECCP2(4)	0	O	DIG	CCP2 比较输出和 CCP2 PWM 输出；优先级高于端口数据。
		1	I	ST	CCP2 捕捉输入。
	P2A(4)	0	O	DIG	ECCP2 增强型 PWM 输出，通道 A ；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。

图注： PWR = 电源， O = 输出， I = 输入， ANA = 模拟信号，DIG = 数字输出， ST = 施密特缓冲输入，TTL = TTL 缓冲输入，x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

注 1： 当 ECCPMX 配置位置 1 时， P1B/P1C 和 P3B/P3C 的默认配置 （仅用于 80 引脚器件）。

2： 外部存储器 I/O 接口优先级高于所有其他数字和 PSP I/O。

3： 仅在 80 引脚的器件上可用。

4： 当 CCP2MX 配置位被清零时， ECCP2/P2A 的备用配置 （单片机模式中的所有器件适用）。

表 10-12： 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	复位值所在页
PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	52
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	52
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	52
PORTG	RDPU	REPU	RJPU(1)	RG4	RG3	RG2	RG1	RG0	52

图注： PORTE 不使用阴影单元。

注 1： 在 64 引脚器件上未使用，读为 0。

10.7 PORTF、 LATF 和 TRISF 寄存器

PORTF 是 7 位宽的双向端口。它对应的数据方向寄存器是 TRISF。置位 TRISF （= 1）可以让相应 PORTF 引脚作为输入引脚 （即将相应的输出驱动器置于高阻态）。清零 TRISF 位 （= 0）将使相应的 PORTF 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。只有 PORTF 的引脚 7 非模拟输入；它是唯一可以接受高达 5.5V 的电压的引脚。

数据锁存器（LATF）也是映射的存储器。LATF 寄存器上的读 - 修改 - 写操作将读写 PORTF 的锁存输出值。

PORTF 上的所有引脚使用施密特触发器输入缓冲器实现的。每个引脚都被单独设置为输入或输出。

PORTF 与几种模拟外设功能复用，包括 A/D 转换器、比较器输入和比较器输出。可以通过将 CMCON 寄存器中相应的位置 1 将引脚 RF2 到 RF6 用作比较器输入或输出。要使用 RF3:RF6 作为数字输入，还必须关闭比较器。

注 1： 当器件复位时，引脚 RF6:RF1 被配置为模拟输入并读为 0。

2： 要将 PORTF 配置为数字 I/O，可以关闭比较器并设置 ADCON1 的值。

例 10-6： 初始化 PORTF

```
CLRF    PORTF   ; Initialize PORTF by
                ; clearing output
                ; data latches
CLRF    LATF    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   07h     ;
MOVWF   CMCON   ; Turn off comparators
MOVLW   0Fh;
MOVWF   ADCON1  ; Set PORTF as digital I/O
MOVLW   0CEh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISF   ; Set RF3:RF1 as inputs
                ; RF5:RF4 as outputs
                ; RF7:RF6 as inputs
```

表 10-13： PORTF 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RF1/AN6/ C2OUT	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。在 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。
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。在 POR 时为默认配置。
	C1OUT	0	O	TTL	比较器 1 输出，优先级高于端口数据。
RF3/AN8	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 时为默认输入配置，不受模拟输出的影响。
RF4/AN9	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 时为默认输入配置，不受模拟输出的影响。
RF5/AN10/ CVREF	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 输出时禁止。
	AN10	1	I	ANA	A/D 输入通道 10 和比较器 C1+ 输入。 POR 时为默认输入配置。
	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。
RF6/AN11	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 时为默认输入配置，不受模拟输出的影响。
RF7/$\overline{SS1}$	RF7	0	O	DIG	LATF<7> 数据输出。
		1	I	ST	PORTF<7> 数据输入。
	$\overline{SS1}$	1	I	TTL	MSSP 的从动选择输入 （MSSP1 模块）。

图注： PWR = 电源， O = 输出， I = 输入， ANA = 模拟信号，DIG = 数字输出， ST = 施密特缓冲输入，
TTL = TTL 缓冲输入， x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

表 10-14： 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	复位值 所在页
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	52
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—	52
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	52
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	50
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	51
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	51

图注： — = 未用位，读为 0。 PORTF 不使用阴影单元。

10.8 PORTG、 TRISG 和 LATG 寄存器

PORTG 是 5 位宽的双向端口。它对应的数据方向寄存器是 TRISG。置位 TRISG （= 1）可以让相应 PORTG 引脚作为输入引脚 （即将相应的输出驱动器置于高阻态）。清零 TRISG 位 （= 0）将使相应的 PORTG 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。 PORTG 上的所有引脚都仅仅是数字的，并且可以接受高达 5.5V 的电压。

数据锁存器 （LATG）也是映射的存储器。 LATG 寄存器上的读 - 修改 - 写操作将读写 PORTG 的锁存输出值。

PORTG 与 EUSART2 功能复用（表 10-15）。PORTG 的引脚都有施密特触发输入缓冲器。

当外设功能使能时，应考虑到每个 PORTG 引脚的 TRIS 位方向设置。有些外设使能时，会覆盖相应引脚的 TRIS 位方向设置而将引脚直接定义为输出，而另一些外设使能时， 也会覆盖相应引脚的 TRIS 方向设置，但将引脚直接定义为输入。用户应该参考相应的外设章节来正确设置 TRIS 位。引脚覆盖值不载入 TRIS 寄存器。这将允许 TRIS 寄存器的读 - 修改 - 写操作而无需担心外设覆盖。

虽然此端口只有 5 位宽，但是 PORTG<7:5> 位仍将使用。它们将用于控制与外部存储器总线 （PORTD、PORTE 和 PORTJ）相关的 I/O 端口上的弱上拉电路。将这些位置 1 将使能弱上拉电路。由于这些位不是与 I/O 端口相关联的控制位，所以没有使用相应的 TRISG 和 LATG。

例 10-7： 初始化 PORTG

```
CLRF    PORTG   ;  Initialize PORTG by
                ; clearing output
                ; data latches
CLRF    LATG    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   04h     ;  Value used to
                ; initialize data
                ; direction
MOVWF   TRISG   ;  Set RG1:RG0 as outputs
                ; RG2 as input
                ;  RG4:RG3 as inputs
```

表 10-15： 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		O	DIG	CCP3 比较和 PWM 输出，优先级高于端口数据。
			I	ST	CCP3 捕捉输入。
	P3A	0	O	DIG	ECCP3 增强型 PWM 输出，通道 A；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。
RG1/TX2/CK2	R21	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 （EUSART2 模块）；优先级高于端口数据。
	CK2	1	O	DIG	同步串行数据输入 （EUSART2 模块）。用户必须配置为输入。
		1	I	ST	同步串行时钟输入 （EUSART2 模块）。
RG2/RX2/DT2	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 （EUSART2 模块）。
	DT2	1	O	DIG	同步串行数据输出 （EUSART2 模块）；优先级高于端口数据。
		1	I	ST	同步串行数据输入 （EUSART2 模块）。用户必须配置为输入。
RG3/CCP4/ P3D	RG3	0	O	DIG	LATG<3> 数据输出。
		1	I	ST	PORTG<3> 数据输入。
	CCP4	0	O	DIG	CCP4 比较输出和 CCP4 PWM 输出；优先级高于端口数据。
		1	I	ST	CCP4 捕捉输入。
	P3D	0	O	DIG	ECCP3 增强型 PWM 输出，通道 D；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。
RG4/CCP5/ P1D	RG4	0	O	DIG	LATG<4> 数据输出。
		1	I	ST	PORTG<4> 数据输入。
	CCP5	0	O	DIG	CCP5 比较输出和 CCP5 PWM 输出；优先级高于端口数据。
		1	I	ST	CCP5 捕捉输入。
	P1D	0	O	DIG	ECCP1 增强型 PWM 输出，通道 D；优先级高于端口和 PSP 数据。可能在增强型 PWM 关闭期间被配置为三态。

图注： PWR = 电源， O = 输出， I = 输入， ANA = 模拟信号，DIG = 数字输出， ST = 施密特缓冲输入，TTL = TTL 缓冲输入，x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

表 10-16： 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	复位值所在页
PORTG	RDPU	REPU	RJPU(1)	RG4	RG3	RG2	RG1	RG0	52
LATG	—	—	—	LATG4	LATG3	LATG2	LATG1	LATG0	52
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	52

图注： — = 未用位，读为 0。 PORTG 不使用阴影单元。

注 1： 在 64 引脚器件上未使用，读为 0。

10.9 PORTH、 LATH 和 TRISH 寄存器

注： PORTH 只在 80 引脚器件上可用。

PORTH 是 8 位宽的双向 I/O 端口。它对应的数据方向寄存器是 TRISH。置位 TRISH （= 1）可以让相应 PORTH 引脚作为输入引脚 （即将相应的输出驱动器置于高阻态）。清零 TRISH 位（= 0）将使相应的 PORTH 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。 PORTH 引脚 <3:0> 仅仅是数字的，并且可以接受高达 5.5V 的电压。

数据锁存器（LATH）也是映射的存储器。LATH 寄存器上的读 - 修改 - 写操作将读写 PORTH 的锁存输出值。

PORTH 上的所有引脚使用施密特触发器输入缓冲器实现的。每个引脚都被单独设置为输入或输出。

当使能外部存储器接口时， 4 个 PORTH 引脚被用作该接口的高位地址线。来自接口的地址输出优先级高于其他数字 I/O。还将覆盖相应的 TRISH 位。

PORTH 引脚 RH4 到 RH7 与模拟转换器输入复用。可以通过在 ADCON1 寄存器中将 PCFG3:PCFG0 控制位置 1 或清零来选择将这些引脚作为模拟输入来工作。

也可以将 ECCP1 和 ECCP3 模块的 PORTH 配置为备用的增强型 PWM 输出通道 B 和 C。可以通过清零 ECCPMX 配置位来完成以上配置。

例 10-8： 初始化 PORTH

```
CLRF    PORTH       ; Initialize PORTH by
                    ; clearing output
                    ; data latches
CLRF    LATH        ; Alternate method
                    ; to clear output
                    ; data latches
MOVLW   0Fh         ; Configure PORTH as
MOVWF   ADCON1      ; digital I/O
MOVLW   0CFh        ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISH       ; Set RH3:RH0 as inputs
                    ; RH5:RH4 as outputs
                    ; RH7:RH6 as inputs
```

表 10-17： PORTH 功能

引脚名称	功能	TRIS 设置	I/O	I/O 类型	说明
RH0/A16	RH0	0	O	DIG	LATH<0> 数据输出。
		1	I	ST	PORTH<0> 数据输入。
	A16	x	O	DIG	外部存储器接口，地址线 16。优先级高于端口数据。
RH1/A17	RH1	0	O	DIG	LATH<1> 数据输出。
		1	I	ST	PORTH<1> 数据输入。
	A17	x	O	DIG	外部存储器接口，地址线 17。优先级高于端口数据。
RH2/A18	RH2	0	O	DIG	LATH<2> 数据输出。
		1	I	ST	PORTH<2> 数据输入。
	A18	x	O	DIG	外部存储器接口，地址线 18。优先级高于端口数据。
RH3/A19	RH3	0	O	DIG	LATH<3> 数据输出。
		1	I	ST	PORTH<3> 数据输入。
	A19	x	O	DIG	外部存储器接口，地址线 19。优先级高于端口数据。
RH4/AN12/P3C	RH4	0	O	DIG	LATH<4> 数据输出。
		1	I	ST	PORTH<4> 数据输入。
	AN12		I	ANA	A/D 输入通道 12。 POR 时为默认输入配置；不影响数字输出。
	P3C(1)	0	O	DIG	ECCP3 增强型 PWM 输出，通道 C；优先级高于端口数据。可能在增强型 PWM 关闭期间被配置为三态。
RH5/AN13/P3B	RH5	0	O	DIG	LATH<5> 数据输出。
		1	I	ST	PORTH<5> 数据输入。
	AN13		I	ANA	A/D 输入通道 13。 POR 时为默认输入配置；不影响数字输出。
	P3B(1)	0	O	DIG	ECCP3 增强型 PWM 输出，通道 B；优先级高于端口数据。可能在增强型 PWM 关闭期间被配置为三态。
RH6/AN14/P1C	RH6	0	O	DIG	LATH<6> 数据输出。
		1	I	ST	PORTH<6> 数据输入。
	AN14		I	ANA	A/D 输入通道 14。 POR 时为默认输入配置；不影响数字输出。
	P1C(1)	0	O	DIG	ECCP1 增强型 PWM 输出，通道 C；优先级高于端口数据。可能在增强型 PWM 关闭期间被配置为三态。
RH7/AN15/P1B	RH7	0	O	DIG	LATH<7> 数据输出。
		1	I	ST	PORTH<7> 数据输入。
	AN15		I	ANA	A/D 输入通道 15。 POR 时为默认输入配置；不影响数字输出。
	P1B(1)	0	O	DIG	ECCP1 增强型 PWM 输出，通道 B；优先级高于端口数据。可能在增强型 PWM 关闭期间被配置为三态。

图注： PWR = 电源，O = 输出，I = 输入，ANA = 模拟信号，DIG = 数字输出，ST = 施密特缓冲输入，TTL = TTL 缓冲输入，x = 任意值（TRIS 位不影响端口方向或者在此选项中被忽略）。

注 1： 当 ECCPMX 配置位被清零时对 P1B/P1C 和 P3B/P3C 进行其他分配。默认分配是 PORTE<6:3>。

表 10-18： 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	复位值所在页
PORTH	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	52
LATH	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0	52
TRISH	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	52

10.10 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。它对应的数据方向寄存器是 TRISJ。置位 TRISJ（= 1）可以让相应 PORTJ 引脚作为输入引脚（即将相应的输出驱动器置于高阻态）。清零 TRISJ 位（= 0）将使相应的 PORTJ 引脚作为输出引脚（即将输出锁存器的内容置于所选择的引脚上）。PORTJ 上的所有引脚都仅仅是数字的，并且可以接受高达 5.5V 的电压。

数据锁存器（LATJ）也是映射的存储器。LATJ 寄存器上的读 - 修改 - 写操作将读写 PORTJ 的锁存输出值。

PORTJ 上的所有引脚使用施密特触发器输入缓冲器实现的。每个引脚都被单独设置为输入或输出。

注：	这些引脚在任何器件复位时都被配置为数字输入引脚。

当使能外部存储器接口时，所有的 PORTJ 引脚都用于控制接口的输出。当通过清零 EBDIS 控制位（MEMCON<7>）而使能此接口时，这将自动发生。还将覆盖 TRISJ 位。

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延迟时的输入电平在外部存储器接口的一个已知状态。单个控制位可以关闭所有上拉电路。可以通过清零 RJPU 位（PORTG<5>）来关闭上拉电路。当端口引脚配置为输出时，其弱上拉功能会自动关闭。发生任何器件复位时，弱上拉电路会被禁止。

例 10-9： 初始化 PORTJ

```
CLRF    PORTJ   ;  Initialize PORTG by
                ; clearing output
                ; data latches
CLRF    LATJ    ; Alternate method
                ; to clear output
                ; data 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 10-19： 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输出，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输出；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	外部存储器接口高字节使能控制输出；优先级高于数字 I/O。

图注： PWR = 电源，O = 输出，I = 输入，ANA = 模拟信号，DIG = 数字输出，ST = 施密特缓冲输入，TTL = TTL 缓冲输入，x = 任意值 （TRIS 位不影响端口方向或者在此选项中被忽略）。

表 10-20： 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	复位值所在页
PORTJ	RJ7	RJ6	RJ5	RJ4	RJ3	RJ2	RJ1	RJ0	52
LATJ	LATJ7	LATJ6	LATJ5	LATJ4	LATJ3	LATJ2	LATJ1	LATJ0	52
TRISJ	TRISJ7	TRISJ6	TRISJ5	TRISJ4	TRISJ3	TRISJ2	TRISJ1	TRISJ0	52
PORTG	RDPU	REPU	RJPU	RG4	RG3	RG2	RG1	RG0	52

图注： PORTJ 不使用阴影单元。

10.11 并行从动端口

当控制位 PSPMODE（PSPCON<4>）被置 1 时，也可以将 PORTD 用作 8 位宽的并行从动端口。外界可以通过 $\overline{RD}$ 控制输入引脚（RE0/$\overline{RD}$）和 $\overline{WR}$ 控制输入引脚（RE1/$\overline{WR}$）对 PORTD 进行异步读写。

注：	对于 80 引脚器件来说，并行从动端口只在单片机模式可用。

PSP 可以直接连接到 8 位微处理器数据总线。外部微处理器可以将PORTD锁存器作为8位锁存器进行读或写。把 PSPMODE 位置 1 将把端口引脚 RE0/$\overline{RD}$ 使能为 $\overline{RD}$ 输入、把 RE1/$\overline{WR}$ 使能为 $\overline{WR}$ 输入并把 RE2/$\overline{CS}$ 使能为 $\overline{CS}$（片选）输入。用作此功能的时候，必须将 TRISE 寄存器（TRISE<2:0>）相应的数据方向位配置为输入（置 1）。

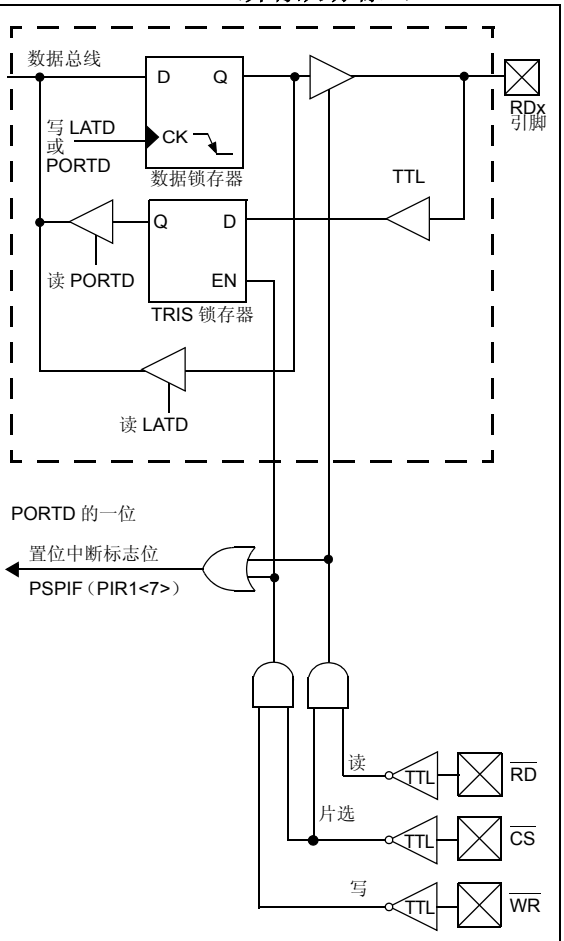
当第一次检测到$\overline{CS}$和$\overline{WR}$线为低电平时，开始写PSP，当检测到它们都为高电平时，写 PSP 结束。当此写操作结束时， PSPIF 和 IBF 标志位都被置 1。

当第一次检测到$\overline{CS}$和$\overline{RD}$线为低电平时，开始读PSP。此时 PORTD 中的数据被读出且 OBF 位被置 1。如果用户将新数据写入 PORTD 并将 OBF 置 1，这些数据将被立即读出，但 OBF 位不置 1。

当检测到 $\overline{CS}$ 或 $\overline{RD}$ 线为高电平时， PORTD 引脚返回到输入状态并且 PSPIF 位被置 1。用户应用程序在为 PSP 提供服务之前应该等待 PSPIF 被置 1；当 PSPIF 被置 1 时，可以查询 IBF 和 OBF 位并进行相应的操作。

图 10-3和图 10-4分别给出了在读和写两种模式下的控制信号的时序。

图 10-2： PORTD 和 PORTE 框图（并行从动端口）



注: I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

寄存器 10-1：　PSPCON：并行从动端口控制寄存器

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

bit7　**IBF**：输入缓冲器满状态位
1 = 接收到第一个数据，等待 CPU 读取
0 = 未接收到任何数据

bit6　**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

bit5　**IBOV**：输入缓冲器溢出检测位
1 = 在尚未读取上一次输入字时发生了一次写入（必须用软件清零）
0 = 没有发生溢出

bit4　**PSPMODE**：并行从动端口模式选择位
1 = 并行从动端口模式
0 = 通用 I/O 模式

bit3-0　**未用**：读为 0

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

图 10-3：　并行从动端口写操作波形

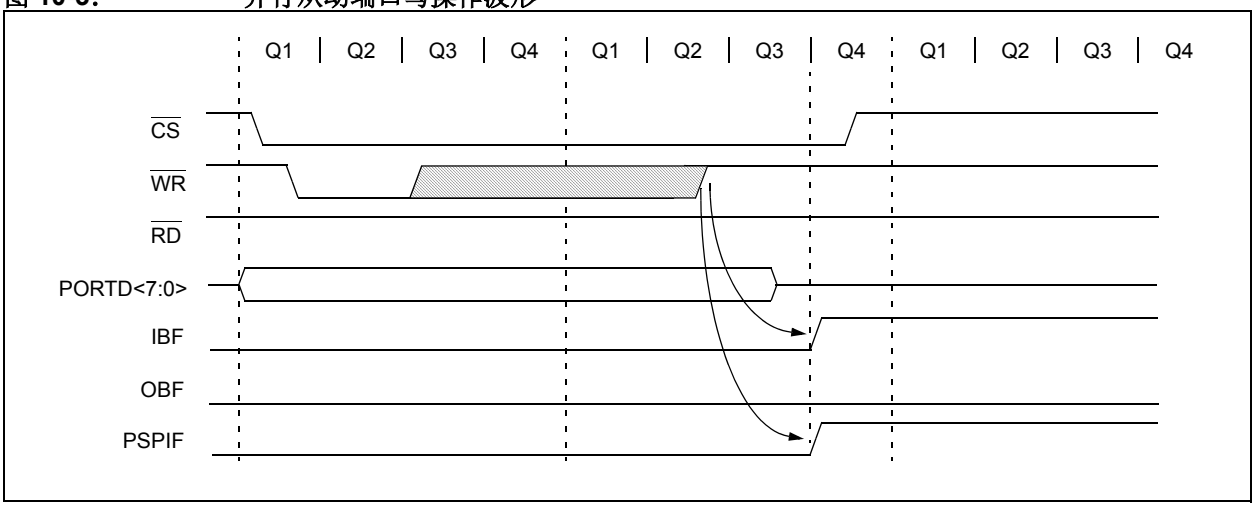


图 10-4：　　并行从动端口读操作波形

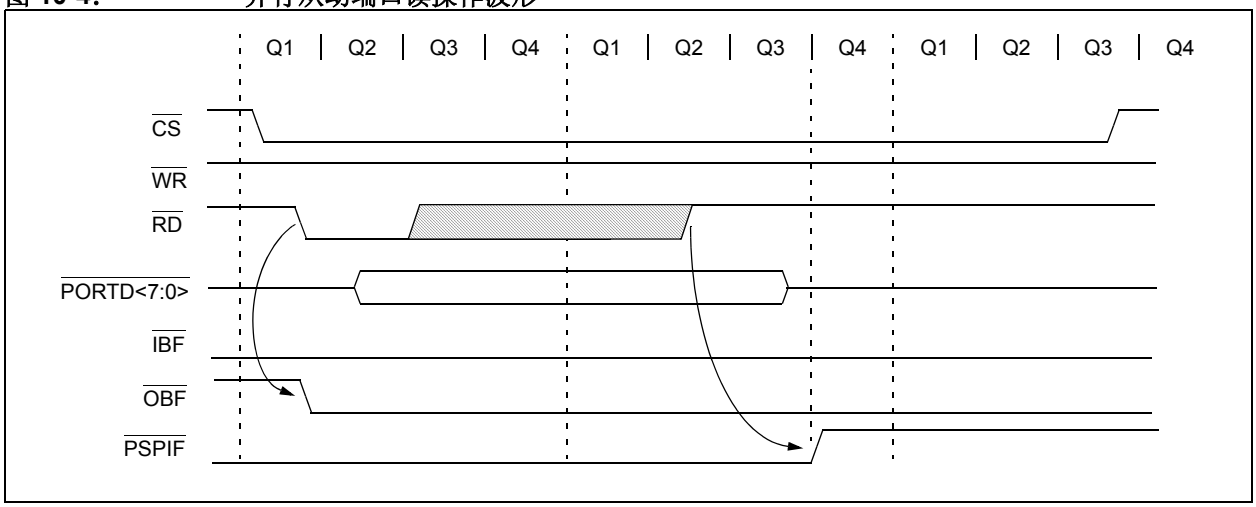


表 10-21：　　与并行从动端口相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	52
LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	52
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	52
PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	52
LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	52
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	52
PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—	51
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51

图注：　　— = 未用位，读为 0。并行从动端口不使用阴影单元。

11.0 TIMER0 模块

Timer0 模块具备以下功能：

- 可通过软件选择，作为 8 位或 16 位定时器 / 计数器工作
- 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）
- 外部时钟边沿选择
- 溢出时中断

T0CON 寄存器（寄存器 11-1）控制模块工作的所有方面，包括预分频选择。该寄存器是可读写的。

图 11-1 显示了 8 位模式下 Timer0 模块的简化框图，

图 11-2 显示了 16 位模式下 Timer0 模块的简化框图。

寄存器 11-1： 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

bit 7 **TMR0ON**：Timer0 开 / 关控制位
1 = 使能 Timer0
0 = 停止 Timer0

bit 6 **T08BIT**：Timer0 8 位 /16 位控制位
1 = Timer0 被配置为 8 位定时器 / 计数器
0 = Timer0 被配置为 16 位定时器 / 计数器

bit 5 T0CS：Timer0 时钟源选择位
1 = T0CKI 引脚上的传输信号作为时钟信号源
0 = 内部指令周期时钟 （CLKO）

bit 4 T0SE：Timer0 时钟源边沿选择位
1 = T0CKI 引脚上信号的下降沿触发递增
0 = T0CKI 引脚上信号的上升沿触发递增

bit 3 **PSA**：Timer0 预分频器分配位
1 = 未分配 Timer0 预分频器。 Timer0 时钟输入不经过预分频器。
0 = 已分配 Timer0 预分频器。 Timer0 时钟输入信号来自预分频器的输出。

bit 2-0 **T0PS2:T0PS0**：Timer0 预分频器选择位
111 = 1:256 预分频比
110 = 1:128 预分频比
101 = 1:64 预分频比
100 = 1:32 预分频比
011 = 1:16 预分频比
010 = 1:8 预分频比
001 = 1:4 预分频比
000 = 1:2 预分频比

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

11.1 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。在定时器模式下（T0CS = 0），除非选择了不同的预分频比，否则，默认情况下在每个时钟周期 Timer0 模块都会增 1（见**第 11.3 节 “预分频器”**）。如果写入 TMR0 寄存器，那么在随后的两个指令周期，它将不再增 1。用户可通过将校正值写入 TMR0 寄存器解决此问题。

通过将 T0CS 位置 1 选择计数器模式。在计数器模式下，Timer0 可在 RA4/T0CKI 引脚信号的每个上升沿或下降沿增加计数。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。下面讨论外部时钟输入的限制条件。

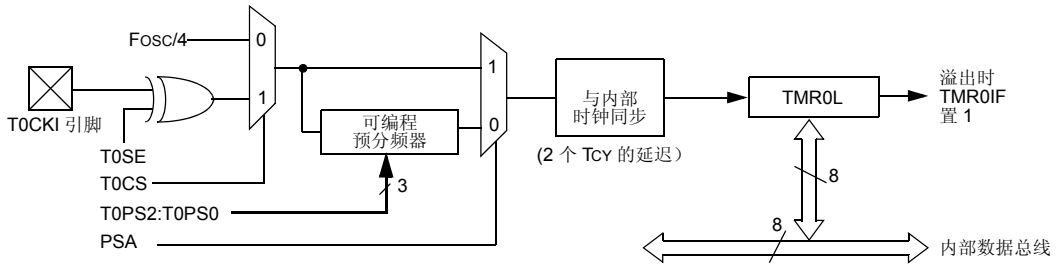
可以使用外部时钟源来驱动 Timer0。但是必须满足一定的要求以确保外部时钟与内部相位时钟（TOSC）同步。在同步之后，定时器 / 计数器仍需要一定的延时才会引发增 1 操作。

11.2 Timer0 的 16 位读写模式

在 16 位模式下，TMR0H 并非 Timer0 的高字节。它实际上是被缓存的 Timer0 的高字节，不可以直接读写（见图 11-2）。在读 TMR0L 时，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，由于可能存在进位，因此需要验证读到字节的有效性）。

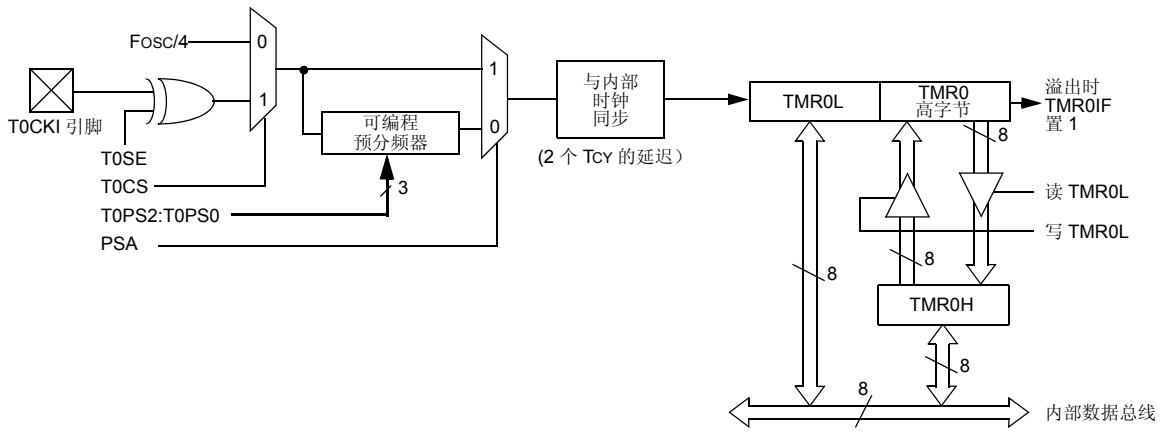
同样，可以使用 TMR0H 缓冲寄存器写入 Timer0 的高字节。在写入 TMR0L 的同时，用 TMR0H 的内容更新 Timer0 的高字节。这样一次就可以完成 Timer0 全部 16 位的更新。

图 11-1： TIMER0 框图 （8 位模式）



注： 复位时，使能 8 位模式下的 Timer0，其时钟输入来自 T0CKI 引脚的最大预分频信号。

图 11-2： TIMER0 框图 （16 位模式）



注： 复位时，使能 8 位模式下的 Timer0，其时钟输入来自 T0CKI 引脚的最大预分频信号。

11.3 预分频器

Timer0 模块的预分频器为一个 8 位计数器。不可直接读写该预分频器。设置 PSA 和 T0PS2:T0PS0 位（T0CON<3:0>）可分别决定预分频器的分配和预分频比。

把 PSA 位清零可将预分频器分配给 Timer0 模块。在分配预分频器时，预分频比可以从 1:2 到 1:256 之间进行选择，分频比以 2 的次幂递增。

当将预分频器分配给 Timer0 模块时，所有写入 TMR0 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：	当将预分频器分配给 Timer0 时写入 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。

11.3.1 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。

11.4 Timer0 中断

8 位模式下，TMR0 寄存器发生从 FFh 到 00h 的溢出，或 16 位模式下，TMR0 寄存器发生从 FFFFh 到 0000h 的溢出，都将产生 TMR0 中断。这种溢出会将标志位 TMR0IF 置 1。清零 TMR0IE 位（INTCON<5>）可屏蔽此中断。在重新允许该中断前，必须在中断服务程序中用软件清零 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 11-1： 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	复位值所在页
TMR0L	Timer0 寄存器的低字节								50
TMR0H	Timer0 寄存器的高字节								50
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	50
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	52

图注： — = 未用位，读为 0。 Timer0 不使用阴影单元。

注：

12.0 TIMER1 模块

Timer1 定时器 / 计数器模块具有以下功能：

- 可通过软件选择，作为 16 位定时器或计数器工作
- 可读写的 8 位寄存器 （TMR1H 和 TMR1L）
- 可选择使用器件时钟作为外部时钟源或选择 Timer1 振荡器作为内部时钟源
- 溢出时中断
- 在触发 CCP 特殊事件时复位
- 器件时钟状态标志位 （T1RUN）

图 12-1 是 Timer1 模块的简化框图。

图 12-2 是此模块在读 / 写模式下的工作原理框图。

模块自身具有低功耗振荡器，可提供额外的时钟选择。Timer1 振荡器也可作为单片机在功耗管理模式下工作时的低功耗时钟源。

Timer1 可以为应用提供实时时钟（RTC），而仅需增加极少的外部元件和代码开销。

Timer1 由 T1CON 控制寄存器 （寄存器 12-1）控制。该寄存器包括 Timer1 振荡器使能位（T1OSCEN）。可以通过将控制位 TMR1ON （T1CON<0>）置 1 或清零来使能或禁止 Timer1。

寄存器 12-1： T1CON：TIMER1 控制寄存器

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7 **RD16**：16 位读 / 写模式使能位
1 = 使能寄存器通过一次 16 位操作对 Timer1 寄存器进行读写
0 = 使能寄存器通过两次 8 位操作对 Timer1 寄存器进行读写

bit 6 **T1RUN**：Timer1 系统时钟状态位
1 = 器件时钟由 Timer1 振荡器产生
0 = 器件时钟由另一个时钟源产生

bit 5-4 **T1CKPS1:T1CKPS0**：Timer1 输入时钟预分频比选择位
11 = 1:8 预分频比
10 = 1:4 预分频比
01 = 1:2 预分频比
00 = 1:1 预分频比

bit 3 **T1OSCEN**：Timer1 振荡器使能位
1 = 使能 Timer1 振荡器
0 = 关闭 Timer1 振荡器
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
当 TMR1CS = 1 时：
1 = 不与外部时钟输入同步
0 = 与外部时钟输入同步
当 TMR1CS = 0 时：
忽略此位。当 TMR1CS=0 时， Timer1 使用内部时钟。

bit 1 **TMR1CS**：Timer1 时钟源选择位
1 = 使用 RC0/T1OSO/T13CKI 引脚上的信号作为外部时钟 （上升沿触发）
0 = 内部时钟 （Fosc/4）

bit 0 **TMR1ON**：Timer1 使能位
1 = 使能 Timer1
0 = 停止 Timer1

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

12.1 Timer1 工作原理

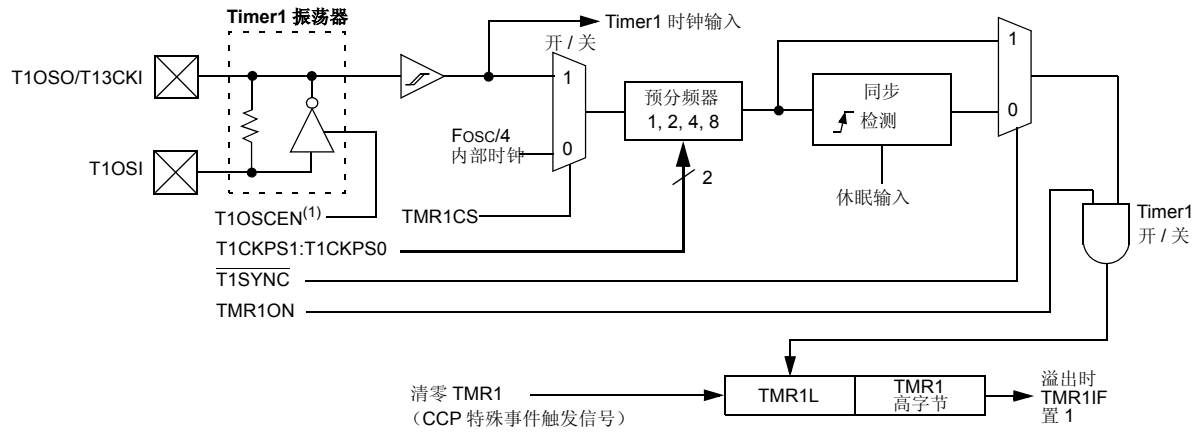
Timer1 可工作在以下模式：

- 定时器
- 同步计数器
- 异步计数器

工作模式由时钟选择位 TMR1CS（T1CON<1>）决定。当 TMR1CS 清零 （= 0）时， Timer1 在每个内部指令周期（FOSC/4）递增。当 TMR1CS 位置 1 时， Timer1 在 Timer1 外部时钟输入信号或 Timer1 振荡器（如果使能）输出信号的每个上升沿递增。

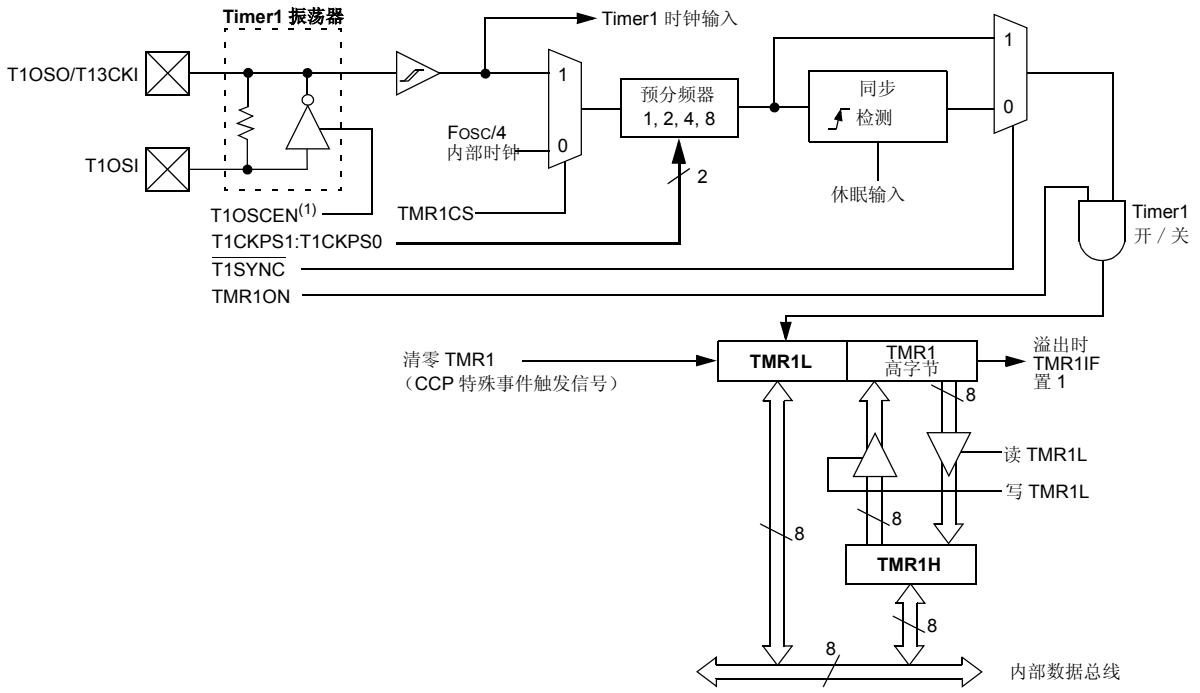
当使能 Timer1 时，RC1/T1OSI 和 RC0/T1OSO/T13CKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

图 12-1： TIMER1 框图



注 1：当使能位 T1OSCEN 清零时，将关闭振荡器的反相器和反馈电阻以降低功耗。

图 12-2： TIMER1 框图 （16 位读 / 写模式）



注 1：当使能位 T1OSCEN 清零时，将关闭振荡器的反相器和反馈电阻以降低功耗。

12.2 Timer1 16 位读 / 写模式

可将 Timer1 配置为 16 位读写模式 （见图 12-2）。当 RD16 控制位 （T1CON<7>）置 1， TMR1H 的地址被映射到 Timer1 的高字节缓冲寄存器。从 TMR1L 的读操作将把 Timer1 的高字节内容装入 Timer1 高字节缓冲寄存器。这种方式使用户可以精确地读取 Time1 的全部 16 位，而不需要像先读高字节再读低字节那样由于两次读取之间可能存在进位，而不得不验证读取的有效性。

对 Timer1 的高字节进行写操作也必须通过 TMR1H 缓冲器进行。在写入 TMR1L 的同时，使用 TMR1H 的内容更新 Timer1 的高字节。这样允许用户将所有的 16 位值一次写入 Timer1 的高字节和低字节。

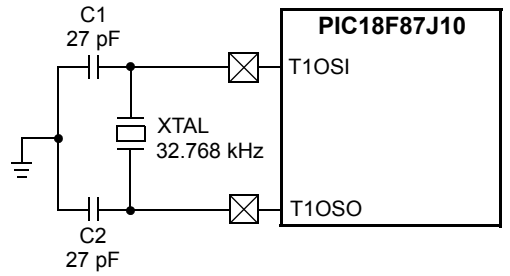
在这一模式下不能直接读写 Timer1 的高字节。所有读写都必须通过 Timer1 高字节缓冲器进行。写入 TMR1H 不会清零 Timer1 预分频器。只有在写 TMR1L 时才会清零预分频器。

12.3 Timer1 振荡器

片内晶体振荡器电路连在 T1OSI（输入）引脚和 T1OSO（放大器输出）引脚之间。通过将 Timer1 振荡器使能位 T1OSCEN （T1CON<3>）置 1 可使能该振荡电路。此振荡电路是一种低功耗电路，它采用了额定振荡频率为 32 kHz 的晶振，在所有的功耗管理模式下都可继续运行。图 12-3 所示为典型的 LP 振荡器电路。表 12-1 显示了 Timer1 振荡器的电容选择。

用户必须提供软件延迟时间来确保 Timer1 振荡器的正常起振。

图 12-3： TIMER1 LP 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 12-1 下的 “注”。

表 12-1： 定时器振荡器 (2,3,4) 的电容选择

振荡器类型	频率	C1	C2
LP	32 kHz	27 pF(1)	27 pF(1)

注 1： Microchip 建议在验证振荡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 / 晶振制造厂商咨询外围元件的相应值。

4： 上述电容值仅供设计参考。

12.3.1 TIMER1 作为时钟源

Timer1 振荡器也可用作功耗管理模式下的时钟源。通过将时钟选择位 SCS1:SCS0 （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 Timer1 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 3.0 节 “功耗管理模式”**。

无论 Timer1 何时提供时钟， Timer1 系统时钟状态标志位 T1RUN （T1CON<6>）均会置 1。这可用于确定控制器的当前时钟模式。该位也可表明故障保护时钟监视器当前正使用的时钟源。如果使能了故障保护时钟监视器并且 Timer1 振荡器在提供时钟信号时发生了故障，查询 T1RUN 位可以确定时钟源是由 Timer1 振荡器还是其他时钟源提供。

12.3.2 低功耗 TIMER1 选项

根据器件配置，Timer1 振荡器可以在两种不同的功耗级别下工作。当 LPT1OSC 配置位置 1 时，Timer1 振荡器在低功耗模式下工作。当 LPT1OSC 清零时，Timer1 在高功耗模式下工作。不管器件工作在什么模式下，特定模式的功耗都是相对固定的。默认 Timer1 配置为工作于功耗较高的模式下。

由于低功耗 Timer1 模式对干扰更加敏感，高噪声环境可能会导致某些振荡器不稳定。因此低功耗选项最适合那些需要重点考虑节省功耗的低噪声应用。

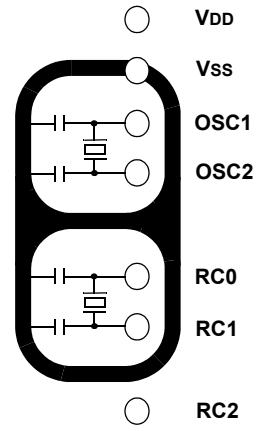
12.3.3 TIMER1 振荡器布线注意事项

Timer1 振荡器在工作期间消耗极少的功率。由于此振荡器的低功耗特性，因此它对在极小范围内快速变化的信号也可能比较敏感。

如图 12-3 所示，振荡电路应该尽可能靠近单片机。除了 VSS 或 VDD 外，在振荡电路附近不应有其他电路。

如果必须要在该振荡器附近布置高速电路（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），当在单面 PCB 板或外加接地层的 PCB 板上使用时，在该振荡电路周围布置接地保护环（如图 12-4 所示）可能会有帮助。

图 12-4： 有接地保护环的振荡电路



注：未按比例绘制。

12.4 Timer1 中断

TMR1 寄存器对（TMR1H:TMR1L）从 0000h 开始递增计数，一直加到 FFFFh，然后从 0000h 重新开始递增计数。如果允许了 Timer1 中断，该中断就会在溢出时产生，由中断标志位 TMR1IF （PIR1<0>）表示。可以通过对 Timer1 中断允许位 TMR1IE （PIE1<0>）置 1 或清零来允许或禁止该中断。

12.5 使用 ECCP 特殊事件触发信号来复位 Timer1

如果 ECCP1 或 ECCP2 被配置为使用 Timer1 并且在比较模式 （CCPxM3:CCPxM0 = 1011）下产生特殊事件触发信号，该信号将复位 Timer3。如果使能了 A/D 模块，来自 ECCP2 的触发信号还将启动 A/D 转换（欲知更多信息，请参见**第 17.2.1 节 “特殊事件触发器”**）。

为了利用这一功能，必须将 Timer1 配置为定时器或同步计数器。在这种情况下，CCPRxH:CCPRxL 这对寄存器实际上变成了 Timer1 的周期寄存器。

如果 Timer1 在异步计数器模式下运行，复位操作可能不起作用。

如果对 Timer1 的写操作和特殊事件触发同时发生，则写操作优先。

注：	来自 ECCPx 模块的特殊事件触发信号不会将中断标志位 TMR1IF （PIR1<0>）置 1。

12.6 使用 Timer1 作为实时时钟

为 Timer1 添加外部 LP 振荡器（如**第 12.3 节 “Timer1 振荡器”**中所述），可以为用户的应用提供包括 RTC 功能的选择。这是通过一个提供精确时基的廉价时钟晶振以及几行计算时间的应用程序代码实现的。当器件工作于休眠模式下并使用电池或超级电容作为电源时，RTC 功能可以完全避免使用独立的 RTC 器件和备用电池。

应用代码程序 RTCisr（如例 12-1 所示），说明了使用中断服务程序以 1/2 秒的间隔递增计数器的简单方法。将 TMR1 寄存器对的值递增直至溢出将触发中断并调用中断服务程序，该程序会使秒计数器增 1。当秒计数器溢出时，分计数器增 1 而小时计数器则会在分计数器溢出时增 1。

由于这对寄存器为 16 位宽，因此直接使用 32.768 kHz 时钟，将其计数到溢出需要 2 秒。 要使溢出按所需的 1 秒间隔进行，必须预先装载这对寄存器。最简单的方法是使用 BSF 指令将 TMR1H 的最高有效位置 1。请注意决不要预先加载或改变 TMR1L 寄存器，因为这样做可能会引起多个周期的累积错误。

要使此方法够精确，Timer1 必须工作在异步模式且必须使能 Timer1 溢出中断（PIE1<0>=1），如程序 RTCinit 所示。同时 Timer1 振荡器也必须使能且始终保持运行。

例 12-1：　使用 TIMER1 中断服务实现实时时钟

```
RTCinit
            MOVLW       80h                 ; Preload TMR1 register pair
            MOVWF       TMR1H               ; for 1 second overflow
            CLRF        TMR1L
            MOVLW       b'00001111'         ; Configure for external clock,
            MOVWF       T1OSC               ; Asynchronous operation, external oscillator
            CLRF        secs                ; Initialize timekeeping registers
            CLRF        mins                ;
            MOVLW       .12
            MOVWF       hours
            BSF         PIE1, TMR1IE        ; Enable Timer1 interrupt
            RETURN
RTCisr
            BSF         TMR1H, 7            ; Preload for 1 sec overflow
            BCF         PIR1, TMR1IF        ; Clear interrupt flag
            INCF        secs, F             ; Increment seconds
            MOVLW       .59                 ; 60 seconds elapsed?
            CPFSGT      secs
            RETURN                          ; No, done
            CLRF        secs                ; Clear seconds
            INCF        mins, F             ; Increment minutes
            MOVLW       .59                 ; 60 minutes elapsed?
            CPFSGT      mins
            RETURN                          ; No, done
            CLRF        mins                ; clear minutes
            INCF        hours, F       ; Increment hours
            MOVLW       .23                 ; 24 hours elapsed?
            CPFSGT      hours
            RETURN                          ; No, done
            CLRF        hours               ; Reset hours
            RETURN                          ; Done
```

表 12-2：　TIMER1 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	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
TMR1L	Timer1 寄存器的低字节								50
TMR1H	Timer1 寄存器的高字节								50
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	50

图注：　Timer1 模块不使用阴影单元。

注：

13.0 TIMER2 模块

Timer2 模块具有以下功能：

- 8 位定时器和周期寄存器 （分别为 TMR2 和 PR2）
- 可读写 （以上两个寄存器）
- 可软件编程的预分频器 （分频比为 1:1、 1:4 和 1:16）
- 可软件编程的后分频器 （分频比为 1:1 至 1:16）
- 当 TMR2 与 PR2 匹配时中断
- 可选作为 MSSP 模块的移位时钟

此模块由 T2CON 寄存器（寄存器 13-1）控制，此寄存器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，以使功耗最低。

图 13-1 所示为该模块的简化框图。

13.1 Timer2 工作原理

在正常工作中， TMR2 在每个时钟周期 （FOSC/4）从 00h 开始加计数。4 位的计数器 / 预分频器提供了对时钟输入不分频、4 分频和 16 分频三种预分频比，可通过预分频控制位 T2CKPS1:T2CKPS0（T2CON<1:0>）进行选择。在每个时钟周期， TMR2 的值都会与其周期寄存器 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 13.2 节 “Timer2 中断”**）。

TMR2 和 PR2 寄存器均可直接读写。在任何器件复位时，TMR2 寄存器都会清零，而 PR2 寄存器则初始化为 FFh。发生以下事件时，预分频和后分频计数器均会清零：

- 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、 $\overline{\text{MCLR}}$ 复位、看门狗定时器复位或者欠压复位）

写 T2CON 时 TMR2 不会清零。

寄存器 13-1： 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

bit 7 **未用：**读为 0

bit 6-3 **T2OUTPS3:T2OUTPS0：**Timer2 输出后分频比选择位
0000 = 1:1 后分频比
0001 = 1:2 后分频比
•
•
•
1111 = 1:16 后分频比

bit 2 **TMR2ON：**Timer2 使能位
1 = Timer2 打开
0 = Timer2 关闭

bit 1-0 **T2CKPS1:T2CKPS0：**Timer2 时钟预分频比选择位
00 = 预分频比为 1
01 = 预分频比为 4
1x = 预分频比为 16

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

13.2 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）为 4 位输出的计数器 / 后分频器提供输入。此计数器产生的 TMR2 匹配中断，由其标志位 TMR2IF（PIR1<1>）锁存。可以通过将 TMR2 匹配中断允许位 TMR2IE（PIE1<1>）置 1 来允许此中断。

可以通过后分频器控制位 T2OUTPS3:T2OUTPS0（T2CON<6:3>）在 16 个后分频比（从 1:1 到 1:16）中进行选择。

13.3 Timer2 输出

TMR2 的不经分频的输出主要用于 CCP 模块，它用作 CCP 模块在 PWM 模式下工作时的时基。

还可选择将 Timer2 用作 MSSP 模块在 SPI 模式下工作时的移位时钟源。**第 18.1 节 “主控 SSP（MSSP）模块概述”**中提供了更多信息。

图 13-1： TIMER2 框图

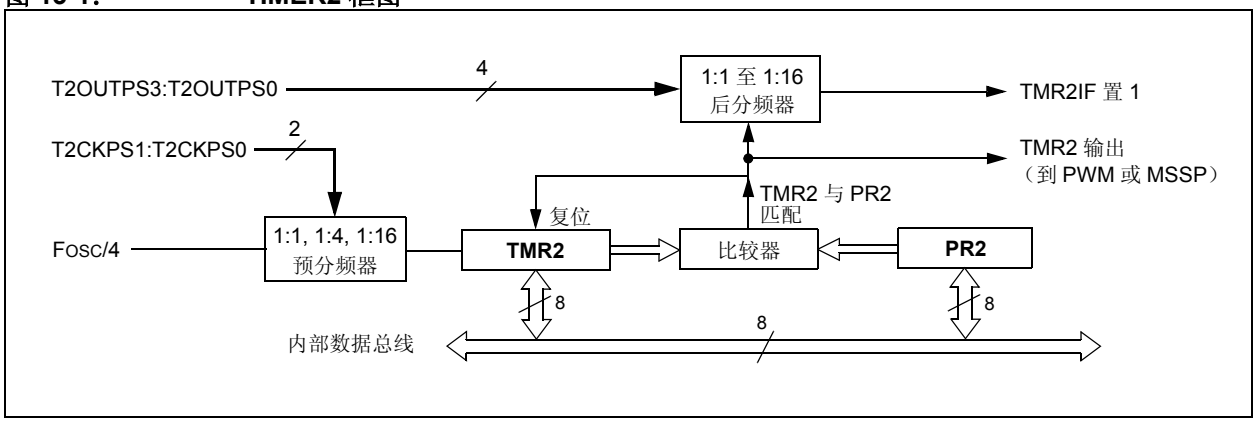


表 13-1： TIMER2 作为定时器 / 计数器时相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
TMR2	Timer2 寄存器								50
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	50
PR2	Timer2 周期寄存器								50

图注： — = 未用位，读为 0。 Timer2 模块不使用阴影单元

14.0 TIMER3 模块

Timer3 定时器 / 计数器模块具有以下功能：

- 可通过软件选择，作为 16 位定时器或计数器运行
- 可读写的 8 位寄存器 （TMR3H 和 TMR3L）
- 可选择使用器件时钟作为外部时钟源或选择 Timer1 振荡器作为内部时钟源
- 溢出时中断
- 模块在触发 CCP 特殊事件时复位

图 14-1 所示为 Timer3 模块的简化框图。

图 14-2 显示了此模块在读 / 写模式下的工作原理框图。

Timer3 模块由 T3CON 寄存器 （寄存器 14-1）控制。此寄存器还可用作 CCP 和 ECCP 模块的可选时钟源。更多信息请参见**第 16.1.1 节 “CCP 模块和定时器资源”**。

寄存器 14-1： T3CON：TIMER3 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON
bit 7							bit 0

bit 7 **RD16**：16 位读 / 写模式使能位
1 = 使能寄存器通过一次 16 位操作对 Timer3 寄存器进行读写
0 = 使能寄存器通过两次 8 位操作对 Timer3 寄存器进行读写

bit 6,3 **T3CCP2:T3CCP1**：CCPx 的时钟源使能位
11 = Timer3 和 Timer4 作为所有 CCP/ECCP 模块的时钟源
10 = Timer3 和 Timer4 作为 ECCP3、 CCP4 和 CCP5 的时钟源；
Timer1 和 Timer2 作为 ECCP1 和 ECCP2 的时钟源
01 = Timer3 和 Timer4 作为 ECCP2、 ECCP3、 CCP4 和 CCP5 的时钟源；
Timer1 和 Timer2 作为 ECCP1 的时钟源
00 = Timer1 和 Timer2 作为所有 CCP/ECCP 模块的时钟源

bit 5-4 **T3CKPS1:T3CKPS0**：Timer3 输入时钟预分频比选择位
11 = 1:8 预分频比
10 = 1:4 预分频比
01 = 1:2 预分频比
00 = 1:1 预分频比

bit 2 **$\overline{\text{T3SYNC}}$**：Timer3 外部时钟输入同步控制位
（不适用于器件时钟来自 Timer1/Timer3 的场合。）
当 TMR3CS=1 时：
1 = 不与外部时钟输入同步
0 = 与外部时钟输入同步
当 TMR3CS=0 时：
忽略此位。当 TMR3CS=0 时， Timer3 使用内部时钟。

bit 1 **TMR3CS**：Timer3 时钟源选择位
1 = 使用 Timer1 振荡器或 T13CKI 引脚信号作为外部时钟输入 （在第一个下降沿之后的上升沿计数）
0 = 内部时钟 （FOSC/4）

bit 0 **TMR3ON**：Timer3 使能位
1 = 使能 Timer3
0 = 停止 Timer3

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

14.1 Timer3 工作原理

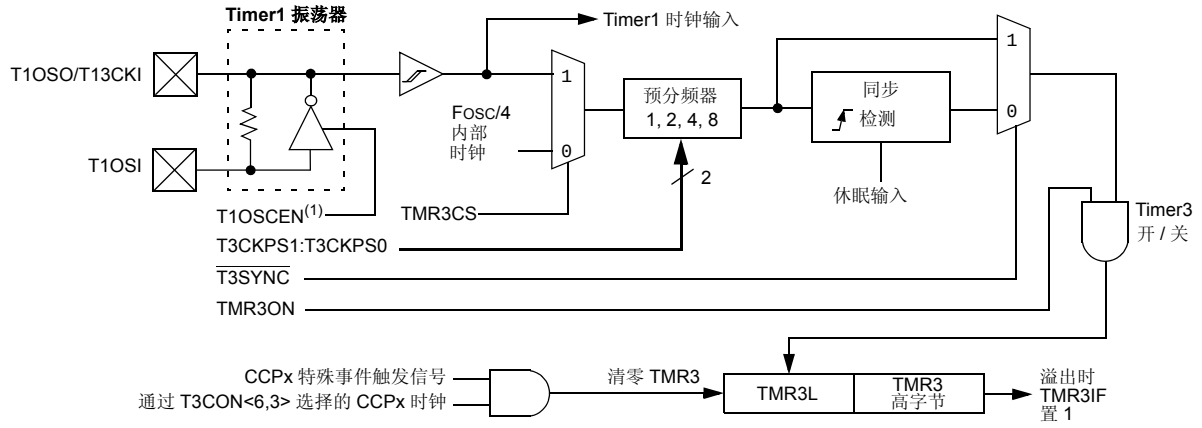
Timer3 有三种工作模式：

- 定时器
- 同步计数器
- 异步计数器

工作模式由时钟选择位 TMR3CS（T3CON<1>）决定。当 TMR3CS 清零（=0）时，Timer3 在每个内部指令周期 （FOSC/4）递增。当 TMR3CS 置 1 时， Timer3 在 Timer1 外部时钟输入信号或 Timer1 振荡器 （如果使能）输出信号的每个上升沿递增。

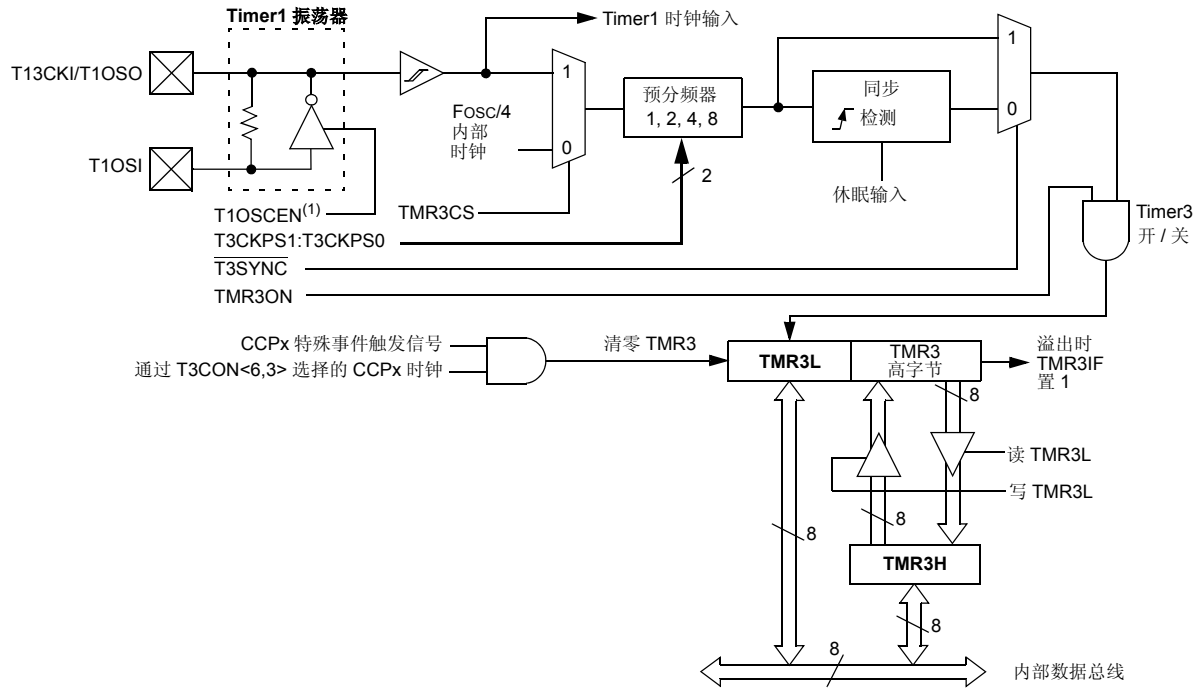
当使能 Timer1 时，RC1/T1OSI 和 RC0/T1OSO/T13CKI 引脚变为输入引脚。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

图 14-1： TIMER3 框图



注 1： 当使能位 T1OSCEN 清零时，将关闭振荡器的反相器和反馈电阻以降低功耗。

图 14-2： TIMER3 框图 （16 位读 / 写模式）



注 1： 当使能位 T1OSCEN 清零时，将关闭振荡器的反相器和反馈电阻以降低功耗。

14.2 Timer3 16 位读 / 写模式

可将 Timer3 配置为 16 位读写模式 （见图 14-2）。当 RD16 控制位 （T3CON<7>）置 1 时， TMR3H 的地址被映射到 Timer3 的高字节缓冲寄存器。从 TMR3L 的读操作将把 Timer3 的高字节内容装入 Timer3 高字节缓冲寄存器。这种方式使用户可以精确地读取 Time3 的全部 16 位，而不需要像先读高字节再读低字节那样由于两次读取之间可能存在进位，而不得不验证读取的有效性。

对 Timer3 的高字节进行写操作也必须通过 TMR3H 缓冲器进行。在写入 TMR3L 的同时，使用 TMR3H 的内容更新 Timer3 的高字节。这样允许用户将所有的 16 位值一次写入 Timer3 的高字节和低字节。

在这一模式下不能直接读写 Timer3 的高字节。所有读写都必须通过 Timer3 高字节缓冲器进行。

写入 TMR3H 不会清零 Timer3 预分频器。只有在写入 TMR3L 时才会清零预分频器。

14.3 使用 Timer1 振荡器作为 Timer3 的时钟源

Timer1 内部振荡器可用作 Timer3 的时钟源。通过将 T1OSCEN （T1CON<3>）置 1，可使能 Timer1 振荡器。要将它用作 Timer3 的时钟源还必须将 TMR3CS 位置 1。如前文所述，这样做也会将 Timer3 配置为在振荡器源的每个上升沿递增。

在**第 12.0 节 “Timer1 模块”**中对 Timer1 振荡器做了说明。

14.4 Timer3 中断

TMR3 寄存器对（TMR3H:TMR3L）从 0000h 开始递增直到 FFFFh 为止，然后溢出重新从 0000h 开始。如果允许了 Timer3 中断，该中断就会在溢出时产生，由中断标志位TMR3IF（PIR2<1>）表示。可以通过对Timer3 中断允许位 TMR3IE （PIE2<1>）置 1 或清零来允许或禁止该中断。

14.5 使用 ECCP 特殊事件触发信号来复位 Timer3

如果 ECCP1 或 ECCP2 被配置为使用 Timer3 并且在比较模式 （CCPxM3:CCPxM0 = 1011）下产生特殊事件触发信号，该信号将复位 Timer3。如果使能了 A/D 模块，来自 ECCP2 的触发信号还将启动 A/D 转换（欲知更多信息，请参见**第 17.2.1 节 “特殊事件触发器”**）。

为了利用这一功能，必须将 Timer3 配置为定时器或同步计数器。在这种情况下，CCPRxH:CCPRxL这对寄存器实际上变成了 Timer3 的周期寄存器。

如果 Timer3 在异步计数器模式下运行，该复位可能不起作用。

如果对 Timer3 的写操作和特殊事件触发同时发生，则写操作优先。

注：	来自ECCPx模块的特殊事件触发信号不会将中断标志位 TMR3IF （PIR2<0>）置 1。

表 14-1： TIMER3 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	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR2	OSCFIF	CMIF	—	—	BCL1IF	—	TMR3IF	CCP2IF	51
PIE2	OSCFIE	CMIE	—	—	BCL1IE	—	TMR3IE	CCP2IE	51
IPR2	OSCFIP	CMIP	—	—	BCL1IP	—	TMR3IP	CCP2IP	51
TMR3L	Timer3 寄存器的低字节								51
TMR3H	Timer3 寄存器的高字节								51
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	50
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{\text{T3SYNC}}$	TMR3CS	TMR3ON	51

图注： — = 未用位，读为 0。 Timer3 模块不使用阴影单元。

注：

15.0 TIMER4 模块

Timer4 定时器模块具有以下功能：

- 8 位定时器寄存器 （TMR4）
- 8 位周期寄存器 （PR4）
- 可读写 （以上两个寄存器）
- 可软件编程的预分频器 （分频比为 1:1、 1:4 和 1:16）
- 可软件编程的后分频器 （分频比为 1:1 至 1:16）
- 当 TMR4 与 PR4 匹配时中断

Timer4 具有如寄存器 15-1 所示的控制寄存器。可以通过清零控制位 TMR4ON （T4CON<2>）关闭 Timer4，以使功耗最低。此寄存器还控制对 Timer4 的预分频比和后分频比的选择。图 15-1 所示为 Timer4 模块的简化框图。

15.1 Timer4 工作原理

Timer4 可以作为 CCP 模块在 PWM 模式下的 PWM 时基。TMR4 寄存器是可读写的，任何方式的器件复位都会使之清零。输入时钟 （Fosc/4）有三种预分频比，分别是 1:1、1:4 或 1:16，可通过控制位 T4CKPS1:T4CKPS0（T4CON<1:0>）选择。TMR4 的匹配输出通过 4 位后分频器 （分频比在 1:1 到 1:16 之间）产生 TMR4 中断，由标志位 TMR4IF （PIR3<3>）表示。

预分频和后分频计数器在发生以下事件时均会清零：

- 对 TMR4 寄存器进行写操作
- 对 T4CON 寄存器进行写操作
- 任何方式的器件复位 （上电复位、 $\overline{\text{MCLR}}$ 复位、看门狗定时器复位或者欠压复位）

写 T4CON 时， TMR4 不会清零。

寄存器 15-1： T4CON：TIMER4 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0
bit 7							bit 0

bit 7 **未用：**读为 0

bit 6-3 **T4OUTPS3:T4OUTPS0：**Timer4 输出后分频比选择位
0000 = 1:1 后分频比
0001 = 1:2 后分频比
•
•
•
1111 = 1:16 后分频比

bit 2 **TMR4ON：**Timer4 使能位
1 = Timer4 打开
0 = Timer4 关闭

bit 1-0 **T4CKPS1:T4CKPS0：**Timer4 时钟预分频比选择位
00 = 预分频比为 1
01 = 预分频比为 4
1x = 预分频比为 16

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

15.2 Timer4 中断

Timer4 模块具有一个 8 位周期寄存器 PR4，该寄存器是可读写的。Timer4 从 00h 开始递增，直到与 PR4 匹配为止，然后在下一个计数周期复位为 00h。在复位时，PR4 寄存器初始化为 FFh。

15.3 TMR4 的输出

TMR4 的输出（在通过后分频器之前）只能用作 CCP 模块的 PWM 时基。它不可以像 Timer2 输出一样用作 MSSP 模块的波特率时钟。

图 15-1： TIMER4 框图

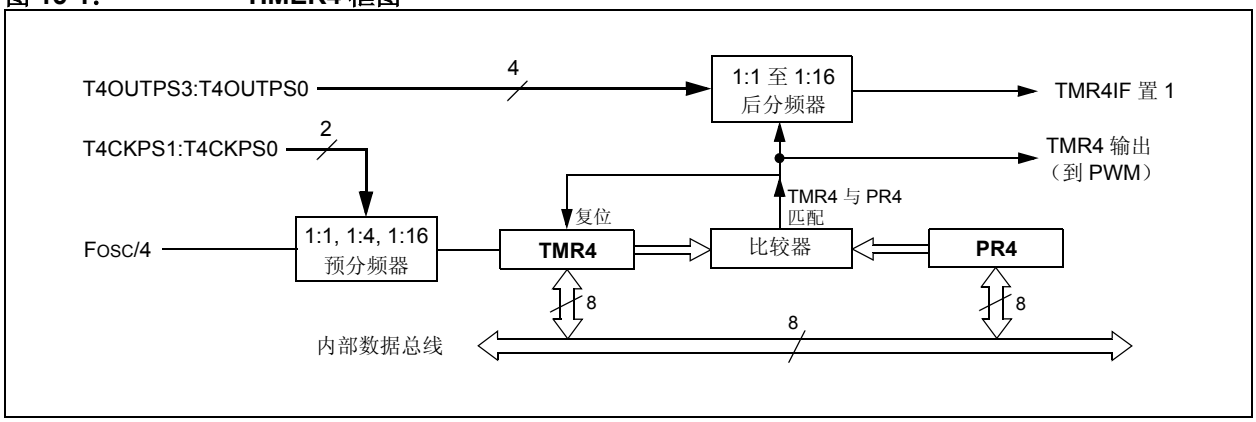


表 15-1： TIMER4 作为定时器 / 计数器时相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
TMR4	Timer4 寄存器								53
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	53
PR4	Timer4 周期寄存器								53

图注： — = 未用位，读为 0。 Timer4 模块不使用阴影单元。

16.0 捕捉 / 比较 /PWM （CCP）模块

所有的 PIC18F87J10 系列器件都有 5 个 CCP （捕捉 / 比较 /PWM）模块，其中两个模块 （CCP4 和 CCP5）实现标准的捕捉、比较和脉宽调制（Pulse-Width Modulation， PWM）模式，在本节将讨论这两个模块。另外三个模块（ECCP1、ECCP2 和 ECCP3）实现标准的捕捉和比较模式，以及增强的 PWM 模式。这些在**第 17.0 节 “增强型捕捉 / 比较 /PWM （ECCP）模块”**中讨论。

每个 CCP/ECCP 模块有一个 16 位寄存器，可用作 16 位捕捉寄存器、16 位比较寄存器或 PWM 主 / 从占空比寄存器。为避免混淆，以下所有的 CCP 模块操作描述均针对 CCP4，但同样适用于 CCP5。

在本章中描述的捕捉和比较操作适用于所有标准和增强的 CCP 模块。**第 16.4 节“PWM 模式”**中描述的 PWM 模式的操作只适用于 CCP4 和 CCP5。

注：	在本节和**第 17.0 节“增强型捕捉 / 比较 /PWM （ECCP）模块”**中，在提到与特定 CCP 模块相关的寄存器和位名称时，一般会使用 “x” 或 “y” 代替特定的模块编号。 因此，“CCPxCON”可能指 ECCP1、ECCP2、ECCP3、 CCP4 或 CCP5 的控制寄存器。

寄存器 16-1： CCPXCON：CCPx 控制寄存器 （CCP4 和 CCP5）

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit 7-6 **未用：**读为 0

bit 5-4 **DCxB1:DCxB0**：CCP 模块 x 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 （bit 1 和 bit 0）。占空比的高 8 位 （DCxB9:DCxB2）在 CCPRxL 中。

bit 3-0 **CCPxM3:CCPxM0**：CCP 模块 x 模式选择位

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，在每个下降沿发生捕捉
0101 = 捕捉模式，在每个上升沿发生捕捉
0110 = 捕捉模式，每 4 个上升沿发生一次捕捉
0111 = 捕捉模式，每 16 个上升沿发生一次捕捉
1000 = 比较模式； CCPx 引脚初始化为低电平，比较匹配时，强制 CCPx 引脚为高电平 （CCPxIF 位置 1）
1001 = 比较模式； CCPx 引脚初始化为高电平，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 = 保留
11xx = PWM 模式

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置位	0 = 清零	x = 未知

16.1 CCP 模块配置

每个捕捉 / 比较 /PWM 模块与一个控制寄存器（通常为 CCPxCON）和一个数据寄存器（CCPRx）相对应。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。

16.1.1 CCP 模块和定时器资源

CCP/ECCP 模块依据所选择的工作模式，使用 Timer1、2、3 或 4。Timer1 和 Timer3 适用于在捕捉或比较模式下的模块，而 Timer2 和 Timer4 适用于在 PWM 模式下的模块。

表 16-1： CCP 模式——定时器资源

CCP 模式	定时器资源
捕捉	Timer1 或 Timer3
比较	Timer1 或 Timer3
PWM	Timer2 或 Timer4

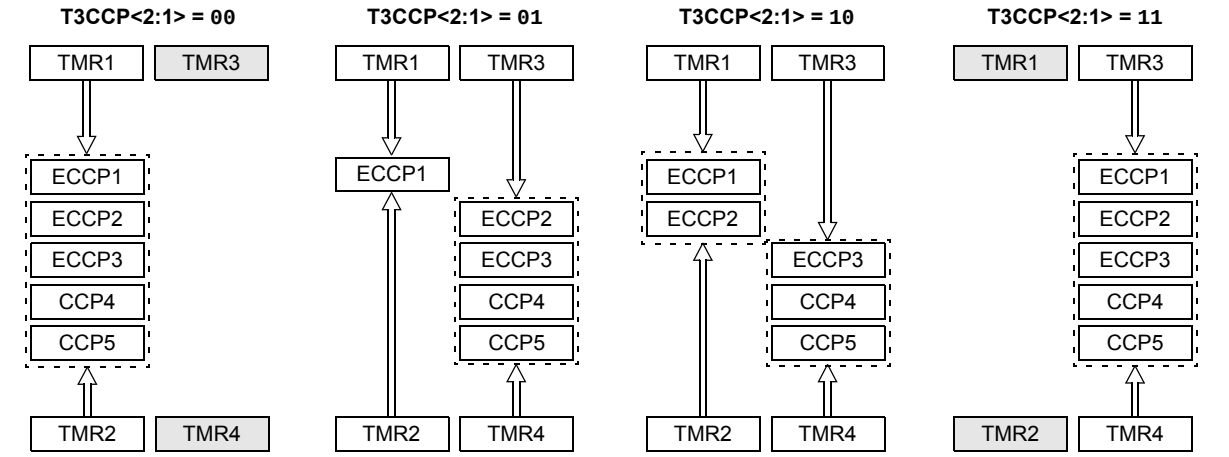
将某个特定的定时器分配给哪个模块是由 T3CON 寄存器 （第 159 页的寄存器 14-1 ）中的 Timer-to-CCP 使能位决定的。根据选定的配置，至多同时可以有 4 个定时器有效，具有相同配置 （捕捉 / 比较或 PWM）的模块共用定时器资源。图 16-1中所示为可能的配置情况。

16.1.2 ECCP2 引脚分配

根据器件配置， ECCP2 的引脚分配 （捕捉输入、比较和 PWM 输出）可以更改。CCP2MX 配置位决定哪个引脚将与 ECCP2 复用。默认情况下，ECCP2 引脚被分配给 RC1 （CCP2MX = 1） 。 如果该配置位被清零，ECCP2 将与 RE7 （64 引脚器件）复用，或与 RB3 或 RE7 （80 引脚器件）复用。

改变 ECCP2 的引脚分配不会自动更改该端口引脚的配置。用户必须始终确认已为 ECCP2 操作正确地配置了相应的 TRIS 寄存器，无论该寄存器的位置在哪里。

图 16-1： CCP/ECCP 和定时器互连配置



Timer1 用于所有 CCP 模块的所有捕捉和比较操作。Timer2 用于所有 CCP 模块的 PWM 操作。模块可以共用这两个定时器资源中的任何一个作为公共时基。

Timer3 和 Timer4 不可用。

Timer1 和 Timer2 仅用于 ECCP1 捕捉和比较或者 PWM 的操作（取决于选定的模式）。

所有其他模块使用 Timer3 或 Timer4。如果模块处于捕捉 / 比较或 PWM 模式，模块可以共用这两个定时器资源中的任何一个作为公共时基。

Timer1 和 Timer2 仅用于 ECCP1 和 ECCP2 捕捉和比较或者 PWM 的操作（取决于为每个模块选定的模式）。如果这两个模块都处于捕捉 / 比较或 PWM 模式，则它们可以使用同一个定时器作为公共时基。

其他模块使用 Timer3 或 Timer4。如果模块处于捕捉 / 比较或 PWM 模式，模块可以共用这两个定时器资源中的任何一个作为公共时基。

Timer3 用于所有 CCP 模块的所有捕捉和比较操作。Timer4 用于所有 CCP 模块的 PWM 操作。模块可以共用这两个定时器资源中的任何一个作为公共时基。

Timer1 和 Timer2 不可用。

16.2 捕捉模式

在捕捉模式下，当对应的 CCPx 引脚发生以下事件时，CCPRxH:CCPRxL 寄存器对捕捉 TMR1 或 TMR3 寄存器的 16 位值。事件定义如下：

- 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 CCPxM3:CCPxM0（CCPxCON<3:0>）选择事件类型。当一个捕捉发生时，中断请求标志位 CCPxIF 置 1，它必须用软件清零。如果在寄存器 CCPRx 中的值被读取之前发生另一个捕捉，那么之前捕捉的值将被新捕捉的值覆盖。

16.2.1 CCP 引脚配置

在捕捉模式下，通过将对应的 TRIS 方向位置 1 将相应的 CCPx 引脚设置为输入。

注：	如果 RG4/CCP5 引脚被配置为输出，对该端口的写操作可能引发一个捕捉事件。

16.2.2 TIMER1/TIMER3 模式选择

用于捕捉功能的定时器（Timer1 和 / 或 Timer3）必须运行在定时器模式或同步计数器模式。在异步计数器模式下，无法进行捕捉操作。用于每个 CCP 模块的定时器在 T3CON 寄存器中进行选择（见**第 16.1.1 节“CCP 模块和定时器资源”**）。

16.2.3 软件中断

当捕捉模式改变时，可能会产生错误的捕捉中断。用户应该保持 CCPxIE 中断允许位清零以避免错误中断。在操作模式发生任何改变之后也应该清零中断标志位 CCPxIF。

16.2.4 CCP 预分频器

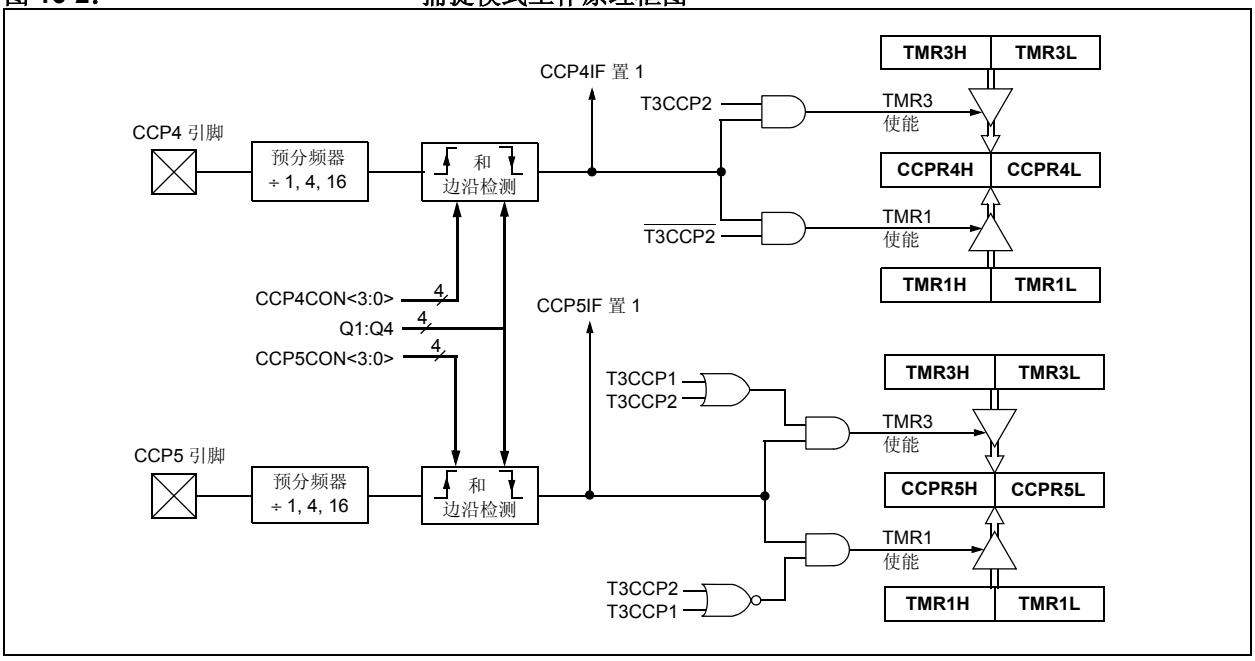
在捕捉模式下有 4 种预分频器设置。它们作为操作模式的一部分由模式选择位（CCPxM3:CCPxM0）指定。每当关闭 CCP 模块或禁止捕捉模式时，就会清零预分频器计数器。这意味着任何复位都将清零预分频器计数器。

从一个捕捉预分频器切换到另一个捕捉预分频器可能会产生中断，但不会将预分频器计数器清零。因此第一个捕捉可能来自一个非零的预分频器。例 16-1 所示为在捕捉预分频器之间进行切换的建议方法。该例也使预分频器计数器清零且不会产生 “错误”中断。

例 16-1： 在捕捉预分频器之间进行切换（以 CCP5 为例）

```
CLRF   CCP5CON     ; Turn CCP module off
MOVLW  NEW_CAPT_PS ; Load WREG with the
                   ; new prescaler mode
                   ; value and CCP ON
MOVWF  CCP5CON     ; Load CCP5CON with
                   ; this value
```

图 16-2： 捕捉模式工作原理框图



16.3 比较模式

在比较模式下，16位CCPRx寄存器的值将不断与TMR1或 TMR3 寄存器对的值相比较。当两者匹配时， CCPx 引脚可能会出现以下几种情况：

- 驱动输出高电平
- 驱动输出低电平
- 输出翻转电平 （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）

引脚的行为取决于模式选择位（CCPxM3:CCPxM0）的值。同时，中断标志位 CCP1xIF 置 1。

16.3.1 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CCP5CON 寄存器会将RG4 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 PORTB 或 PORTC I/O 数据锁存器。

16.3.2 TIMER1/TIMER3 模式选择

如果 CCP 模块使用比较功能，Timer1 和 / 或 Timer3 必须运行在定时器模式或同步计数器模式。在异步计数器模式下，可能无法进行比较操作。

16.3.3 软件中断模式

当选择了产生软件中断的模式时（CCPxM3:CCPxM0 = 1010），对应的 CCPx 引脚不受影响。如果中断被允许并把 CCPxIE 位置 1，只产生一个 CCP 中断。

图 16-3： 比较模式工作原理框图

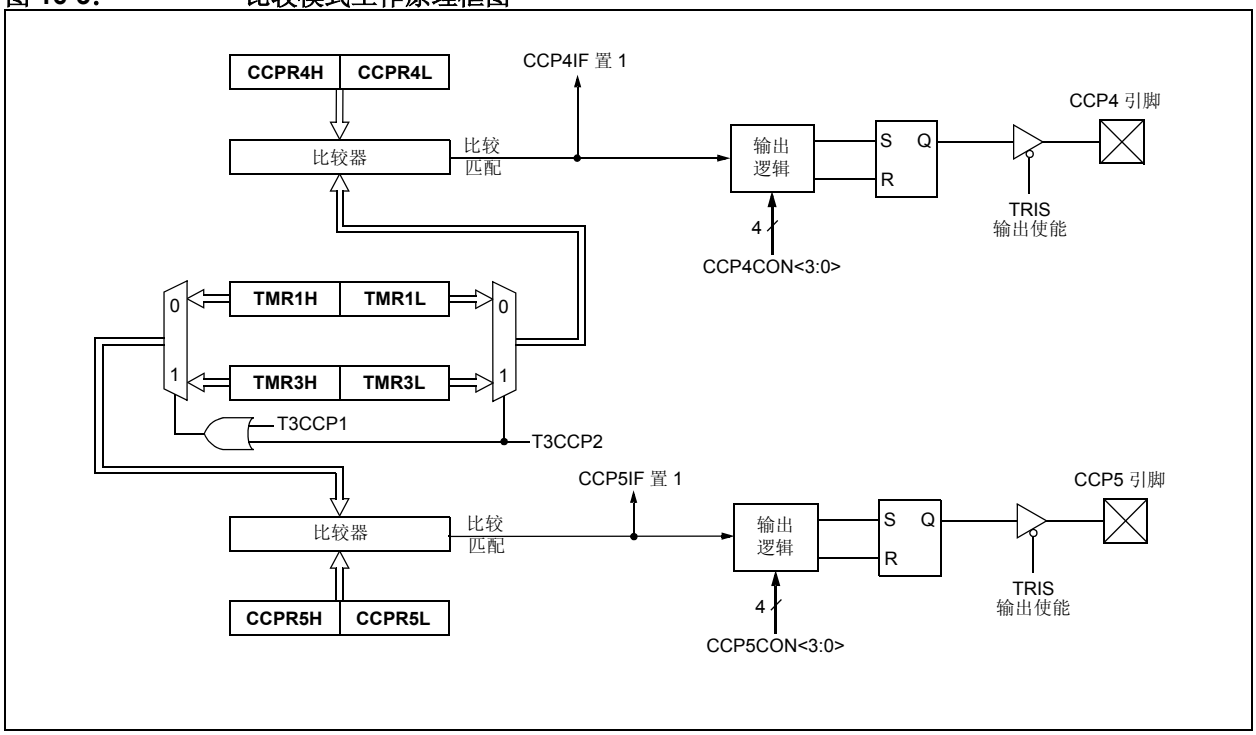


表 16-2： 与捕捉、比较、 TIMER1 和 TIMER3 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	50
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR2	OSCFIF	CMIF	—	—	BCL1IF	—	TMR3IF	CCP2IF	51
PIE2	OSCFIE	CMIE	—	—	BCL1IE	—	TMR3IE	CCP2IE	51
IPR2	OSCFIP	CMIP	—	—	BCL1IP	—	TMR3IP	CCP2IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	52
TMR1L	Timer1 寄存器的低字节								50
TMR1H	Timer1 寄存器的高字节								50
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	50
TMR3H	Timer3 寄存器的高字节								51
TMR3L	Timer3 寄存器的低字节								51
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	51
CCPR4L	捕捉 / 比较 /PWM 寄存器 4 的低字节								53
CCPR4H	捕捉 / 比较 /PWM 寄存器 4 的高字节								53
CCPR5L	捕捉 / 比较 /PWM 寄存器 5 的低字节								53
CCPR5H	捕捉 / 比较 /PWM 寄存器 5 的高字节								53
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	53
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	53

图注： — = 未用位，读为 0。捕捉 / 比较、 Timer1 或 Timer3 不使用阴影单元。

16.4 PWM 模式

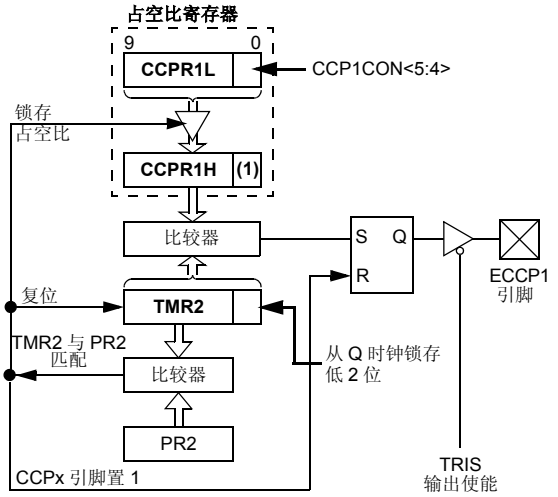
在脉宽调制 （PWM）模式下， CCPx 引脚可输出分辨率高达 10 位的 PWM 信号。由于 CCP4 和 CCP5 引脚与 PORTG 数据锁存器复用，必须清零相应的 TRISG 位才能使 CCP4 或 CCP5 引脚成为输出引脚。

注：	清零 CCP4CON 或 CCP5CON 寄存器会将 RG3 或 RG4 输出锁存器 （取决于器件配置）强制为默认的低电平。这不是 PORTG I/O 数据锁存器。

图 16-4 所示为 PWM 模式下 CCP 模块的简化框图。

如需了解设置 CCP 模块以进行 PWM 操作的详细步骤，请参见**第 16.4.3 节 “进行 PWM 操作所需的设置”**。

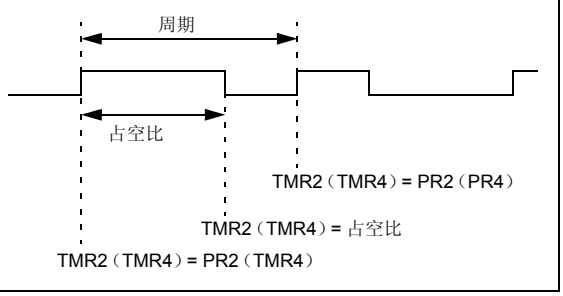
图 16-4： 简化的 PWM 框图



注 1： 占空比寄存器的低 2 位由 1 个 2 位锁存器保存，该锁存器是模块硬件的一部分。它在物理上独立于 CCPR 寄存器。

一个 PWM 输出（图 16-5）包含一个时基（周期）和一段输出高电平的时间（占空比）。PWM 的频率是周期的倒数 （1/ 周期）。

图 16-5： PWM 输出



16.4.1 PWM 周期

PWM 周期是通过写 PR2 （PR4）寄存器来指定的。可以使用公式 16-1 来计算 PWM 周期。

公式 16-1：

$$\text{PWM 周期} = [(\text{PR2}) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

PWM 频率定义为 1 / [PWM 周期]。

当 TMR2 （TMR4）等于 PR2 （PR4）时，在下一个递增计数周期中会发生以下 3 个事件：

- TMR2 （TMR4）清零
- CCPx 引脚被置 1 （例外情况：如果 PWM 占空比 = 0%， CCPx 不被置 1）
- PWM 占空比从 CCPRxL 被锁存到 CCPRxH

注：	Timer2 和 Timer 4 后分频器不被用于决定 PWM 频率（见**第 13.0 节 “Timer2 模块”**和**第 15.0 节 “Timer4 模块”**）。使用后分频器时，其伺服更新速率可与 PWM 输出频率不同。

16.4.2 PWM 占空比

PWM 占空比可通过写入 CCPRxL 寄存器和 CCPxCON<5:4> 位来指定，最高分辨率可达 10 位。CCPRxL 保存高 8 位， CCPxCON<5:4> 保存低 2 位。CCPRxL:CCPxCON<5:4> 代表这 10 位值。公式 16-2 用来计算 PWM 占空比。

公式 16-2：

$$\text{PWM 占空比} = (\text{CCPRxL:CCPxCON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

可以在任何时候写入 CCPRxL 和 CCPxCON<5:4>，但直到 PR2 （PR4）和 TMR2 （TMR4）中的值匹配时（即，当周期结束时），占空比的值才被锁存到 CCPRxH。在 PWM 模式下，CCPRxH 是一个只读寄存器。

CCPRxH 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。这种双重缓冲结构极其重要，可以避免 PWM 切换占空比的时候产生毛刺。

当 CCPRxH 和 2 位锁存器的值与 TMR2 （TMR4）加上内部 2 位 Q 时钟或 TMR2（TMR4）预分频器的 2 位的值匹配时， CCPx 引脚被清零。

对于给定的PWM频率，其最大分辨率（位）由公式 16-3 给出：

公式 16-3：

$$\text{PWM 分辨率（最大）} = \frac{\log\left(\frac{FOSC}{FPWM}\right)}{\log(2)} \text{ 位}$$

注： 如果 PWM 的占空比值大于 PWM 周期，CCPx 引脚将不会被清零。

16.4.3 进行 PWM 操作所需的设置

在为 PWM 操作配置 CCP 模块时应该执行以下步骤：

1. 通过写 PR2 （PR4）寄存器设置 PWM 周期。
2. PWM 占空比可通过写入 CCPRxL 寄存器和 CCPxCON<5:4> 位来设置。
3. 将相应的TRIS位清零以将CCPx引脚设为输出。
4. 通过写 T2CON （T4CON）来设置 TMR2 （TMR4）预分频值，然后使能 Timer2 （Timer4）。
5. 配置 CCPx 模块以进行 PWM 操作。

表 16-3： 40 MHz 下的 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 （1,4,16）	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最大分辨率 （位）	10	10	10	8	7	6.58

表 16-4：　　与 PWM、 TIMER2 和 TIMER4 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	50
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IF	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	52
TMR2	Timer2 寄存器								50
PR2	Timer2 周期寄存器								50
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	50
TMR4	Timer4 寄存器								53
PR4	Timer4 周期寄存器								53
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	53
CCPR4L	捕捉 / 比较 /PWM 寄存器 4 低字节								53
CCPR4H	捕捉 / 比较 /PWM 寄存器 4 高字节								53
CCPR5L	捕捉 / 比较 /PWM 寄存器 5 低字节								53
CCPR5H	捕捉 / 比较 /PWM 寄存器 5 高字节								53
CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0	53
CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0	53

图注：　　— = 未用，读为 0。 PWM、 Timer2 或 Timer4 不使用阴影单元。

17.0 增强型捕捉 / 比较 /PWM （ECCP）模块

在 PIC18F87J10 系列器件中，3 个 CCP 模块被实现为带有增强的 PWM 功能的标准 CCP 模块。这些增强的功能包括提供 2 个或 4 个输出通道、用户可选的极性、死区控制以及自动关闭和重新启动。**第 17.4 节 “增强型 PWM 模式”**中有对增强功能的详细讨论。ECCP 模块的捕捉、比较和单输出 PWM 功能与标准 CCP 模块相同。

增强型 CCP 模块的控制寄存器如寄存器 17-1 所示。它与 CCP4CON/CCP5CON 寄存器的不同之处在于其高 2 位用来控制 PWM 功能。

除了通过增强型的 CCPxCON 寄存器扩展了可用的模式范围之外，每个 ECCP 模块都有 2 个与增强型 PWM 操作以及自动关闭功能相关的寄存器。它们是：

- ECCPxDEL （死区延迟）
- ECCPxAS （自动关闭配置）

寄存器 17-1： CCPxCON：增强型 CCPx 控制寄存器 （ECCP1/ECCP2/ECCP3）

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

bit7-6 **PxM1:PxM0**：增强型 PWM 输出配置位

如果 CCPxM3:CCPxM2 = 00、01 或 10：

xx = PxA 被分配为捕捉 / 比较输入 / 输出； PxB、 PxC 和 PxD 被分配为端口引脚

如果 CCPxM3:CCPxM2 = 11：

00 = 单输出：PxA 调制输出； PxB、 PxC 和 PxD 被分配为端口引脚
01 = 全桥正向输出：P1D 调制输出； P1A 有效； P1B 和 P1C 无效
10 = 半桥输出：P1A 和 P1B 为带死区控制的调制输出； P1C 和 P1D 被分配为端口引脚
11 = 全桥反向输出：P1B 调制输出； P1C 有效； P1A 和 P1D 无效

bit5-4 **DCxB1:DCxB0**：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。占空比的高 8 位在 CCPRxL 中。

bit3-0 **CCPxM3:CCPxM0**：增强型 CCPx 模块模式选择位

0000 = 捕捉 / 比较 /PWM 关闭 （复位 ECCPx 模块）
0001 = 保留
0010 = 比较模式，匹配时输出翻转电平
0011 = 捕捉模式
0100 = 捕捉模式，在每个下降沿发生捕捉
0101 = 捕捉模式，在每个上升沿发生捕捉
0110 = 捕捉模式，每 4 个上升沿发生一次捕捉
0111 = 捕捉模式，每 16 个上升沿发生一次捕捉
1000 = 比较模式， ECCPx 引脚初始化为低电平，比较匹配时置位输出 （置位 CCPxIF）
1001 = 比较模式， ECCPx 引脚初始化为高电平，比较匹配时清零输出 （置位 CCPxIF）
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，将 CCPxIF 位置 1，如果使能了 A/D 模块， ECCP2 触发还会启动 A/D 转换） **(1)**
1100 = PWM 模式； PxA 和 PxC 为高电平有效； PxB 和 PxD 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 为低电平有效

注 1： 仅供 ECCP1 和 ECCP2 使用；就像 “1010”仅供 ECCP3 使用一样。

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

17.1 ECCP 输出和配置

每个增强型 CCP 模块至多有 4 个 PWM 输出，这取决于选定的操作模式。这些输出被指定为从 PxA 到 PxD，与多个 I/O 引脚复用。有些 ECCP 引脚的分配是固定的，而其他引脚的分配根据器件配置而改变。这些会发生分配改变的引脚的控制位是：

- CCP2MX 配置位
- ECCPMX 配置位 （仅 80 引脚器件）
- 程序存储器操作模式，由 EMB 配置位设置 （仅 80 引脚器件）

表 17-1、表 17-2 和表 17-3 中总结了增强型 CCP 模块的引脚分配。要将 I/O 引脚配置为 PWM 输出，必须通过分别将 PxMx 和 CCPxMx 位 （CCPxCON<7:6> 和 <3:0>）置位，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。

17.1.1 ECCP1/ECCP3 输出和程序存储器模式

对于 80 引脚的器件，使用扩展单片机模式对在增强型 PWM 模式下使用 ECCP1 和 ECCP3 有间接影响。默认情况下， PWM 输出 P1B/P1C 和 P3B/P3C 与 PORTE 引脚以及外部存储器总线的高字节复用。当该总线在扩展单片机模式下有效时，它会忽略增强型 CCP 输出并使它们不可用。因此，当器件处于扩展单片机模式并使用默认引脚配置时， ECCP1 和 ECCP3 只能用于兼容（单输出） PWM 模式。

这种配置的例外情况是当为外部总线选定了 12 位地址宽度时 （EMB1:EMB0 配置位 = 01）。在这种情况下，即使外部总线有效， PORTE 的高位引脚仍然继续作为数字 I/O 引脚； P1B/P1C 和 P3B/P3C 仍可被用作增强型 PWM 输出。

如果在增强型单片机运行期间需要使用更多的 PWM 输出，P1B/P1C 和 P3B/P3C 输出可被重新分配给 PORTH 的高位，可以通过清零 ECCPMX 配置位来实现上述配置。

17.1.2 ECCP2 输出和程序存储器模式

对于 80 引脚器件，器件的程序存储器模式（**第 5.1.3 节 “PIC18F8XJ10/8XJ15 程序存储器模式”**）还会影响该模块的引脚复用。

ECCP2 输入 / 输出 （ECCP2/P2A）可以用作 3 个复用功能之一。所有器件的默认分配 （CCP2MX 配置位置 1）是 RC1。清零 CCP2MX 会将 ECCP2/P2A 重新分配为 RE7。

80 引脚器件还有一个选择。当这些器件在单片机模式下运行时，上述复用选项仍然适用。在扩展单片机模式下，清零 CCP2MX 会将 ECCP2/P2A 重新分配给 RB3。

17.1.3 ECCP1 和 ECCP3 对 CCP4 和 CCP5 的使用

只有 ECCP2 模块有 4 个可用的专用输出引脚。假定这些引脚不需要被用作 I/O 端口或其他复用功能，它们随时可以使用，而无需任何其他 CCP 模块的介入。

ECCP1 和 ECCP3 只有 3 个专用的输出引脚：ECCPx/PxA、 PxB 和 PxC。只要这些模块被配置为复双输出 PWM 模式时，通常用作 CCP4 或 CCP5 的引脚分别成为 ECCP3 和 ECCP1 的 PxD 输出引脚。CCP4 和 CCP5 模块保持工作，但是它们的输出被覆盖。

17.1.4 ECCP 模块和定时器资源

与标准 CCP 模块相同，ECCP 模块可以使用定时器 1、2、3 或 4，这取决于选定的模式。Timer1 和 Timer3 在捕捉或比较模式下可用于模块，而 Timer2 和 Timer4 则在 PWM 模式下可用于模块。**第 16.1.1 节 “CCP 模块和定时器资源”** 中提供了更多有关定时器资源的详细信息。

表 17-1： ECCP1 的引脚配置

ECCP 模式	CCP1CON 配置	RC2	RE6	RE5	RG4	RH7	RH6
所有 PIC18F6XJ10/6XJ15 器件：							
兼容 CCP	00xx 11xx	ECCP1	RE6	RE5	RG4/CCP5	N/A	N/A
双输出 PWM	10xx 11xx	P1A	P1B	RE5	RG4/CCP5	N/A	N/A
四输出 PWM	x1xx 11xx	P1A	P1B	P1C	P1D	N/A	N/A
PIC18F8XJ10/8XJ15 器件， ECCPMX = 0，单片机模式：							
兼容 CCP	00xx 11xx	ECCP1	RE6/AD14	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
双输出 PWM	10xx 11xx	P1A	RE6/AD14	RE5/AD13	RG4/CCP5	P1B	RH6/AN14
四输出 PWM	x1xx 11xx	P1A	RE6/AD14	RE5/AD13	P1D	P1B	P1C
PIC18F8XJ10/8XJ15 器件， ECCPMX = 1，**扩展单片机模式， 16 位或 20 位地址宽度：**							
兼容 CCP	00xx 11xx	ECCP1	RE6/AD14	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
PIC18F8XJ10/8XJ15 器件， ECCPMX = 1，单片机模式或扩展单片机模式， 12 位地址宽度：							
兼容 CCP	00xx 11xx	ECCP1	RE6/AD14	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
双输出 PWM	10xx 11xx	P1A	P1B	RE5/AD13	RG4/CCP5	RH7/AN15	RH6/AN14
复双输出 PWM	x1xx 11xx	P1A	P1B	P1C	P1D	RH7/AN15	RH6/AN14

图注： x = 任意值， N/A = 不可用。阴影单元表示在给定模式下 ECCP1 不使用的引脚分配。

注 1： 当 ECCP1 处于四输出 PWM 模式时， CCP5 的输出被 P1D 覆盖；其他情况下， CCP5 完全正常工作。

表 17-2： ECCP2 的引脚配置

ECCP 模式	CCP2CON 配置	RB3	RC1	RE7	RE2	RE1	RE0
所有器件， CCP2MX = 1，任一操作模式：							
兼容 CCP	00xx 11xx	RB3/INT3	ECCP2	RE7	RE2	RE1	RE0
双输出 PWM	10xx 11xx	RB3/INT3	P2A	RE7	P2B	RE1	RE0
复双输出 PWM	x1xx 11xx	RB3/INT3	P2A	RE7	P2B	P2C	P2D
所有器件， CCP2MX = 0，单片机模式：							
兼容 CCP	00xx 11xx	RB3/INT3	RC1/T1OS1	ECCP2	RE2	RE1	RE0
双输出 PWM	10xx 11xx	RB3/INT3	RC1/T1OS1	P2A	P2B	RE1	RE0
复双输出 PWM	x1xx 11xx	RB3/INT3	RC1/T1OS1	P2A	P2B	P2C	P2D
PIC18F8XJ10/8XJ15 器件， CCP2MX = 0，扩展单片机模式：							
兼容 CCP	00xx 11xx	ECCP2	RC1/T1OS1	RE7/AD15	RE2/$\overline{CS}$	RE1/$\overline{WR}$	RE0/$\overline{RD}$
双输出 PWM	10xx 11xx	P2A	RC1/T1OS1	RE7/AD15	P2B	RE1/$\overline{WR}$	RE0/$\overline{RD}$
复双输出 PWM	x1xx 11xx	P2A	RC1/T1OS1	RE7/AD15	P2B	P2C	P2D

图注： x = 任意值。阴影单元表示在给定模式下 ECCP2 不使用的引脚分配。

表 17-3： ECCP3 的引脚配置

ECCP 模式	CCP3CON 配置	RG0	RE4	RE3	RG3	RH5	RH4
所有 PIC18F6XJ10/6XJ15 器件：							
兼容 CCP	00xx 11xx	ECCP3	RE4	RE3	RG3/CCP4	N/A	N/A
双输出 PWM	10xx 11xx	P3A	P3B	RE3	RG3/CCP4	N/A	N/A
复双输出 PWM	x1xx 11xx	P3A	P3B	P3C	P3D	N/A	N/A
PIC18F8XJ10/8XJ15 器件， ECCPMX = 0，单片机模式：							
兼容 CCP	00xx 11xx	ECCP3	RE6/AD14	RE5/AD13	RG3/CCP4	RH7/AN15	RH6/AN14
双输出 PWM	10xx 11xx	P3A	RE6/AD14	RE5/AD13	RG3/CCP4	P3B	RH6/AN14
复双输出 PWM	x1xx 11xx	P3A	RE6/AD14	RE5/AD13	P3D	P3B	P3C
PIC18F8XJ10/8XJ15 器件， ECCPMX = 1，扩展单片机模式， 16 位或 20 位地址宽度：							
兼容 CCP	00xx 11xx	ECCP3	RE6/AD14	RE5/AD13	RG3/CCP4	RH7/AN15	RH6/AN14
PIC18F8XJ10/8XJ15 器件， ECCPMX = 1， 单片机模式或扩展单片机模式， 12 位地址宽度：							
兼容 CCP	00xx 11xx	ECCP3	RE4/AD12	RE3/AD11	RG3/CCP4	RH5/AN13	RH4/AN12
双输出 PWM	10xx 11xx	P3A	P3B	RE3/AD11	RG3/CCP4	RH5/AN13	RH4/AN12
复双输出 PWM	x1xx 11xx	P3A	P3B	P3C	P3D	RH5/AN13	RH4/AN12

图注： x = 任意值， N/A = 不可用。阴影单元表示在给定模式下 ECCP3 不使用的引脚分配。

注 1： 当 ECCP3 处于四输出 PWM 模式时， CCP4 的输出被 P1D 覆盖；其他模式下， CCP4 完全正常工作。

17.2 捕捉和比较模式

除了要在下面讨论的特殊事件触发器的操作，ECCP 模块的捕捉和比较模式在操作上与 CCP4 相同。**第 16.2 节“捕捉模式”**和**第 16.3 节“比较模式”**详细讨论了这些操作。

17.2.1 特殊事件触发器

ECCP1 和 ECCP2 组成了一个内部硬件触发器，在 CCPRx 寄存器对和选定的定时器发生匹配时在比较模式下产生触发信号。触发信号可以用来启用某个操作。通过将 CCPxCON<3:0> 设置为 “1011”选择此模式。

ECCP1 或 ECCP2 的特殊事件触发器输出会复位 TMR1 或 TMR3 寄存器对，这取决于当前选定的定时器资源。因此 CCPRx 寄存器对可被用作 Timer1 或 Timer3 的 16 位可编程周期寄存器。另外，如果 A/D 模块使能，则 ECCP2 特殊事件触发器还将启动 A/D 转换。

没有为 ECCP3，CCP4 或 CCP5 实现特殊事件触发器。为这些模块选择特殊事件触发模式与选择带软件中断的比较模式 （CCPxM3:CCPxM0 = 1010）作用相同。

注：	来自 ECCP2 的特殊事件触发信号不会将 Timer1 或 Timer3 中断标志位置 1。

17.3 标准 PWM 模式

当配置为单输出模式时， ECCP 模块的工作方式与**第 16.4 节 “PWM 模式”**中描述的 PWM 模式下的标准 CCP 模块相同。如同表 17-1 到 17-3 中所述，这种模式有时候也称为 “兼容 CCP”模式。

注：	当设置单输出 PWM 操作时，用户可以自由使用**第 16.4.3 节 “进行 PWM 操作所需的设置”**或**第 17.4.9 节“PWM 操作的设置”**中描述的过程。后者更加常用但是只适用于单输出或多输出 PWM。

17.4 增强型 PWM 模式

增强型 PWM 模式提供了更多的 PWM 输出选项以进行范围更广的控制应用。该模块是标准 CCP 模块的向后兼容版本，提供至多 4 个输出，分别被命名为 PxA 到 PxD。用户还能够选择信号的极性（高电平有效或低电平有效）。通过分别设置 CCPxCON 寄存器的 PxM1:PxM0 和 CCPxM3CCPxM0 位（CCPxCON<7:6> 和 CCPxCON<3:0>）可以配置模块的输出模式和极性。

为了避免混淆，本节以 ECCP1 和 TMR2 模块为例详细介绍了增强型 PWM 模式操作。根据 ECCP1 给出控制寄存器的名称。3 个增强型模块和 2 个定时器资源，可以互换使用并且工作方式相同。通过在 T3CON 中选择适当的位可以为 PWM 操作选择 TMR2 或 TMR4。

图 17-1 所示为 PWM 工作的简化框图。所有的控制寄存器都是双重缓冲的，并且在一个新的 PWM 周期的开头（Timer2 复位时的周期边界）装载以防止在任何输出上出现毛刺。例外的是 PWM 延迟寄存器 ECCP1DEL，该寄存器在占空比边界或周期边界（两者之中先发生的）装载。由于缓冲，模块将不会立即启动，而要等到分配的定时器复位为止。这意味着增强型 PWM 波形并不精确地匹配标准的 PWM 波形，而是偏移一个完整的指令周期（4 TOSC）。

如前所述，用户必须手动地为输出配置相应的 TRIS 位。

17.4.1 PWM 周期

PWM 周期可以通过写 PR2 寄存器来指定。可使用以下公式来计算 PWM 周期。

公式 17-1：

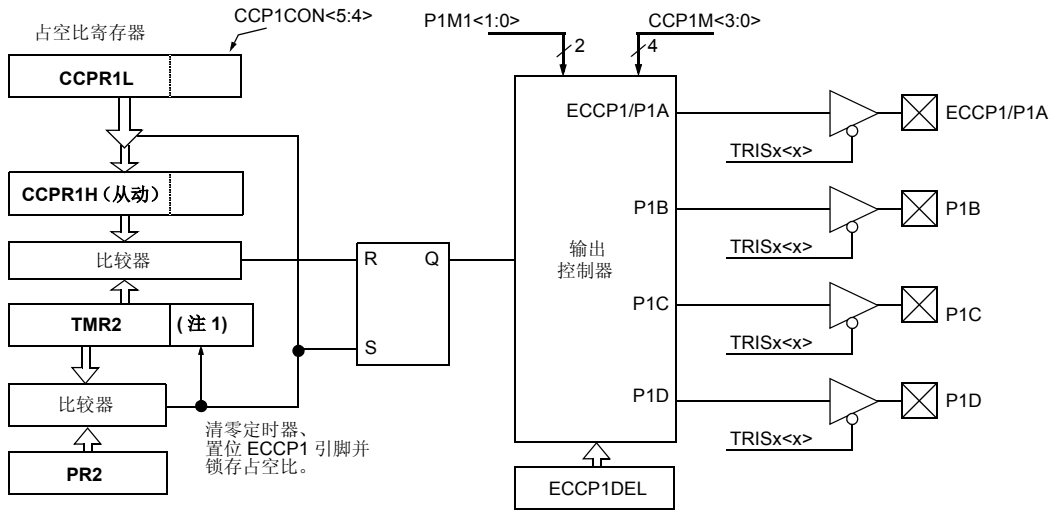
$$\text{PWM 周期} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 预分频值})$$

PWM 频率定义为 1/[PWM 周期]。当 TMR2 等于 PR2 时，在下一个递增计数周期中会发生以下三个事件：

- TMR2 被清零
- ECCP1 引脚被置 1（如果 PWM 占空比 = 0%，ECCP1 引脚将不会被置 1）
- PWM 占空比从 CCPR1L 复制到 CCPR1H

注： 确定 PWM 频率时不使用 Timer2 后分频器（见**第 13.0 节 “Timer2 模块”**）。使用后分频器时，其伺服更新速率可与 PWM 输出频率不同。

图 17-1： 增强型 PWM 模块的简化框图



注： 8 位定时器 TMR2 寄存器结合内部 Q 时钟或预分频器的 2 位产生 10 位时基。

17.4.2 PWM 占空比

PWM 的占空比由写入 CCPR1L 寄存器的值和写入 CCP1CON<5:4> 位的值指定。最高分辨率可达 10 位。CCPR1L 保存高 8 位数值， CCP1CON<5:4> 保存低 2 位数值。由 CCPR1L:CCP1CON<5:4> 表示这个 10 位值。 PWM 占空比由以下公式计算。

公式 17-2：

PWM 占空比 = (CCPR1L:CCP1CON<5:4>) • TOSC • (TMR2 预分频值)

可以在任何时候写入 CCPR1L 和 CCP1CON<5:4>，但直到 PR2 与 TMR2 中的值匹配时（即当周期结束时），占空比的值才被锁存到 CCPR1H。在 PWM 模式中，CCPR1H 是只读寄存器。

CCPR1H 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。这种双重缓冲结构极其重要，可以避免 PWM 切换占空比的时候产生毛刺。当 CCPR1H 和 2 位锁存值与 TMR2 结合内部 Q 时钟或预分频器的 2 位值匹配时，ECCP1 引脚被清零。对于给定的 PWM 频率，其最大 PWM 分辨率 （位）可以由以下公式计算：

公式 17-3：

$$\text{PWM 分辨率（最大）} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ 位}$$

注： 如果 PWM 占空比的值大于 PWM 周期，ECCP1 引脚将不会被清零。

17.4.3 PWM 输出配置

CCP1CON 寄存器中的 P1M1:P1M0 位可以实现以下 4 种配置：

- 单输出
- 半桥输出
- 全桥输出，正向模式
- 全桥输出，反向模式

单输出模式是在**第 17.4 节 “增强型 PWM 模式”**中讨论的标准 PWM 模式。在接下来的各节中将详细介绍半桥和全桥输出模式。

图 17-2 中汇总了各种配置与输出的关系。

表 17-4： 40 MHz 时的 PWM 频率和分辨率示例

PWM 频率	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
定时器预分频器 (1、 4 和 16)	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最大分辨率 (位)	10	10	10	8	7	6.58

图 17-2： PWM 输出关系 （高电平有效状态）

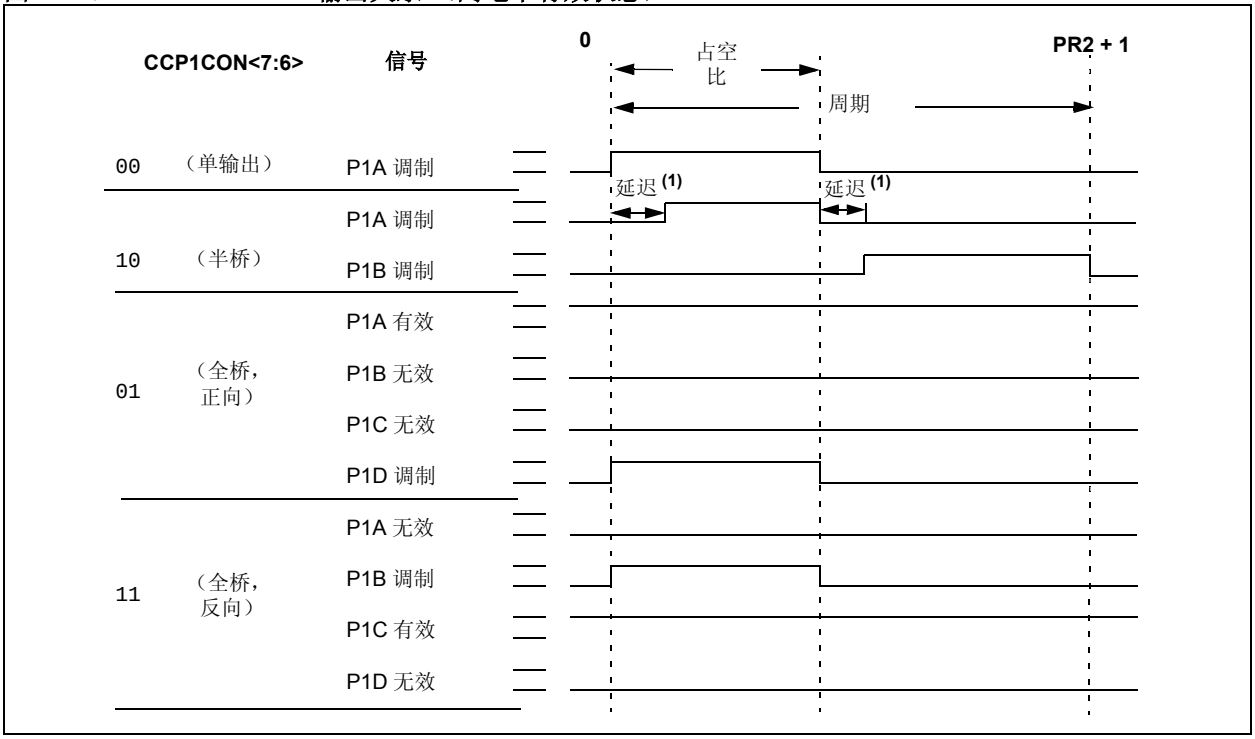
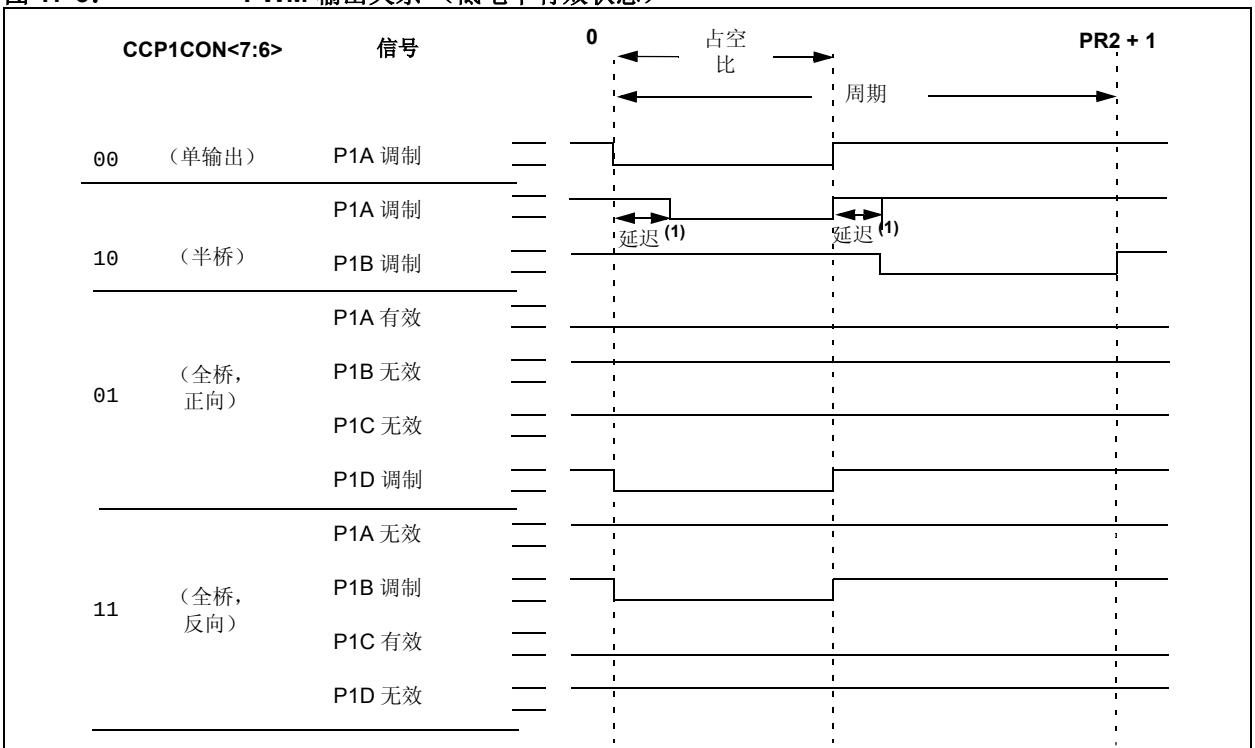


图 17-3： PWM 输出关系 （低电平有效状态）



关系：

- 周期 = 4 * Tosc * (PR2 + 1) * (TMR2 预分频值)
- 占空比 = Tosc * (CCPR1L<7:0>:CCP1CON<5:4>) * (TMR2 预分频值)
- 延迟 = 4 * Tosc * (ECCP1DEL<6:0>)

注 1： 使用 ECCP1DEL 寄存器设置死区延迟 （**第 17.4.6 节 “可编程死区延迟”**）。

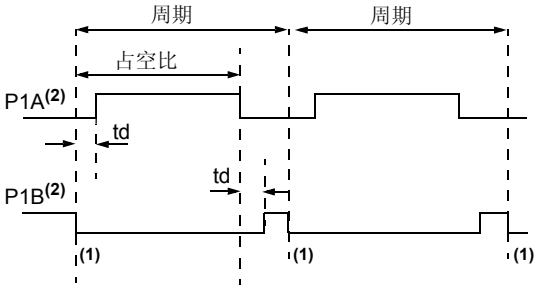
17.4.4 半桥模式

在半桥输出模式下，两个引脚用作输出端来推挽驱动负载。PWM 输出信号在 P1A 引脚上输出，而互补的 PWM 输出信号在 P1B 引脚上输出（图 17-4）。如图 17-5 所示，此模式可用于半桥应用，或那些使用 2 个 PWM 信号来调制 4 个电源开关的全桥应用。

在半桥输出模式下，可编程的死区延迟可用来防止在半桥电源设备中产生直通电流。在输出有效之前，设置 P1DC6:P1DC0 位的值来指定指令周期数。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。有关死区延迟操作的更多详细信息，请参见**第 17.4.6 节“可编程死区延迟”**。

由于 P1A 和 P1B 输出与 PORTC<2> 和 PORTE<6> 数据锁存器复用，必须清零 TRISC<2> 和 TRISE<6> 位以将 P1A 和 P1B 配置为输出。

图 17-4： 半桥 PWM 输出

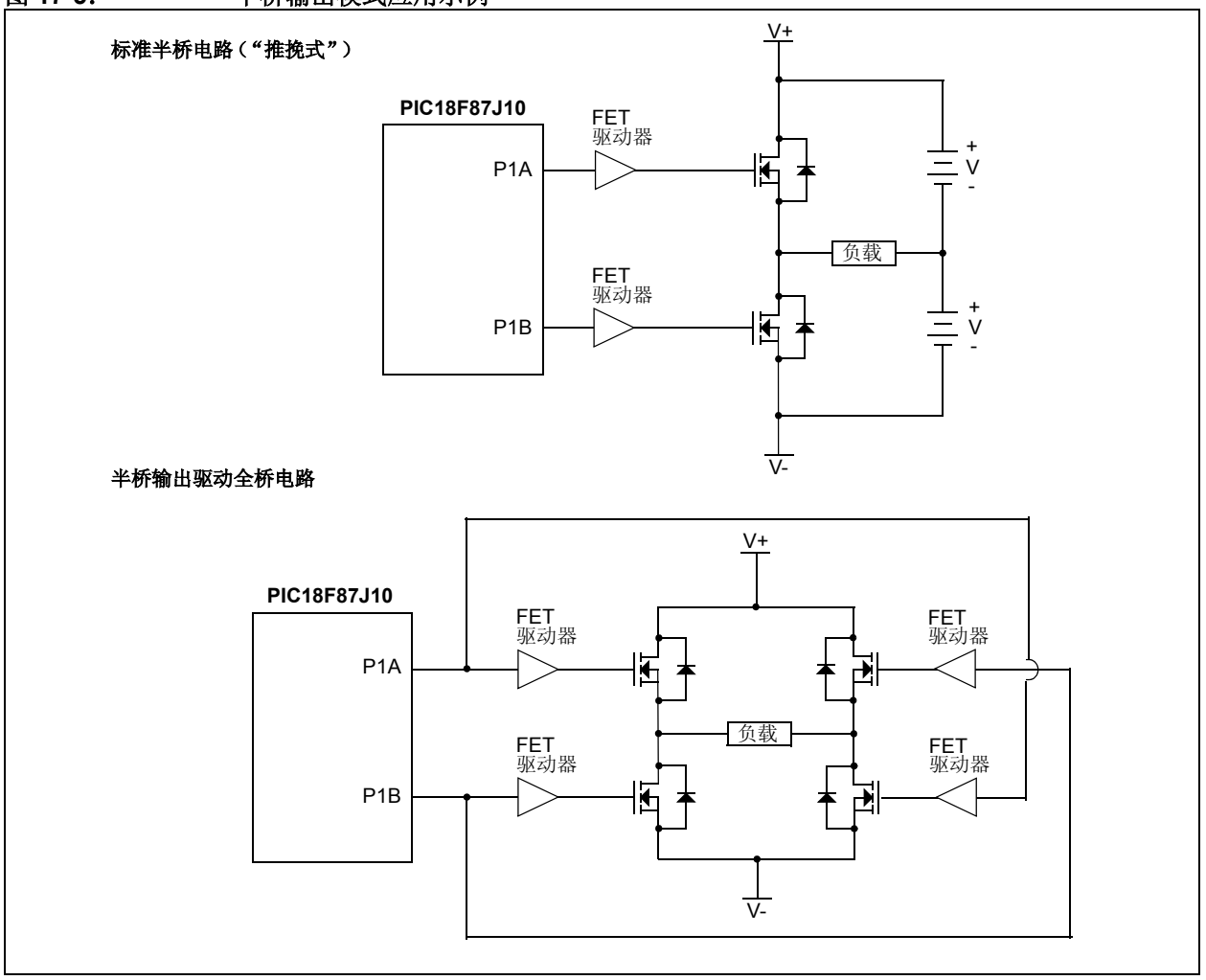


td = 死区延迟

注 1： 此时，TMR2 寄存器的值与 PR2 寄存器的值相等。

2： 输出信号显示为高电平有效。

图 17-5： 半桥输出模式应用示例

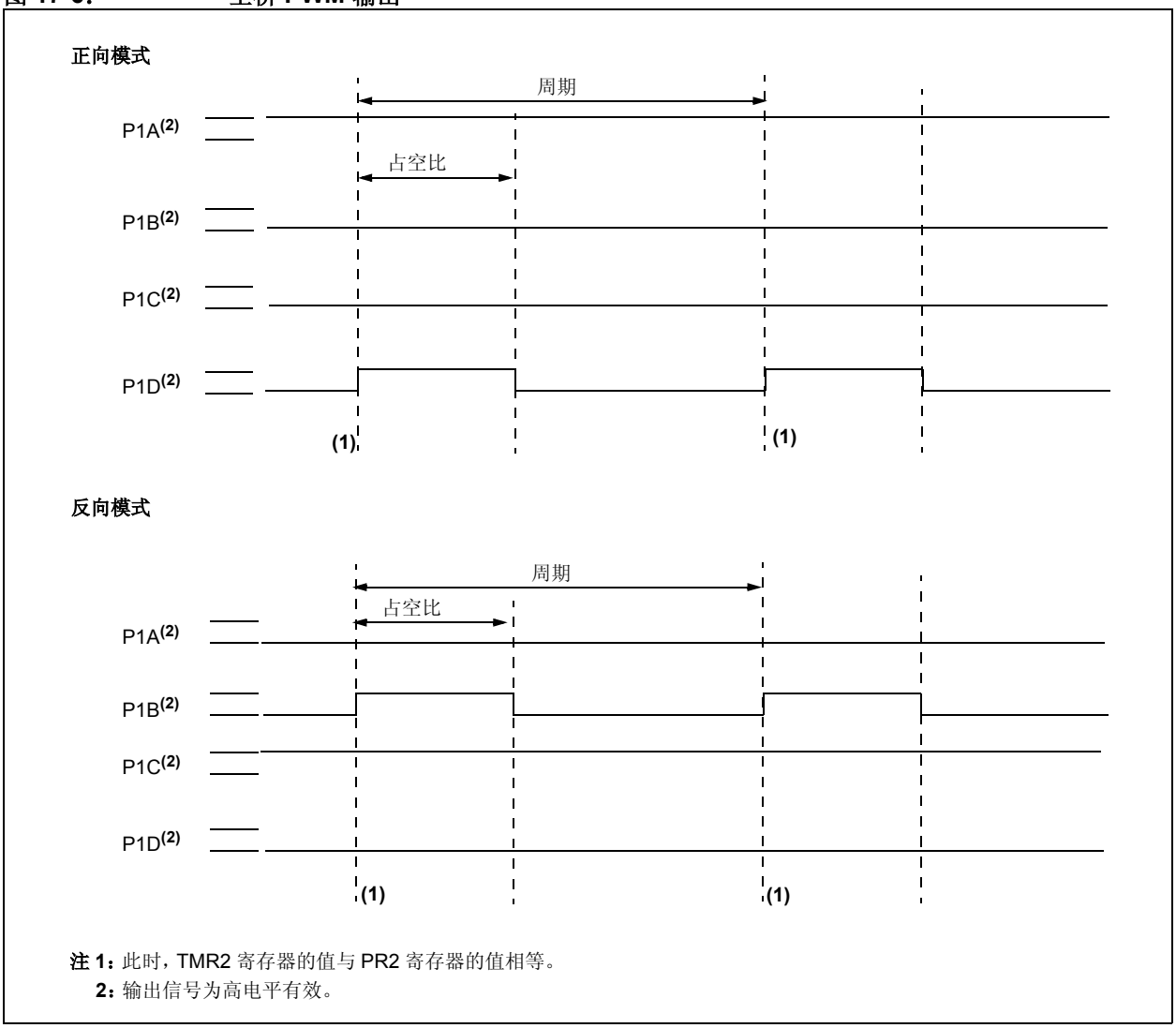


17.4.5 全桥模式

在全桥输出模式下， 4 个引脚被用作输出；但是，一次只能有 2 个输出有效。在正向模式下，引脚 P1A 连续有效而引脚 P1D 为调制输出。在反向模式下，引脚 P1C 连续有效而引脚 P1B 为调制输出。这些在图 17-6 中进行了说明。

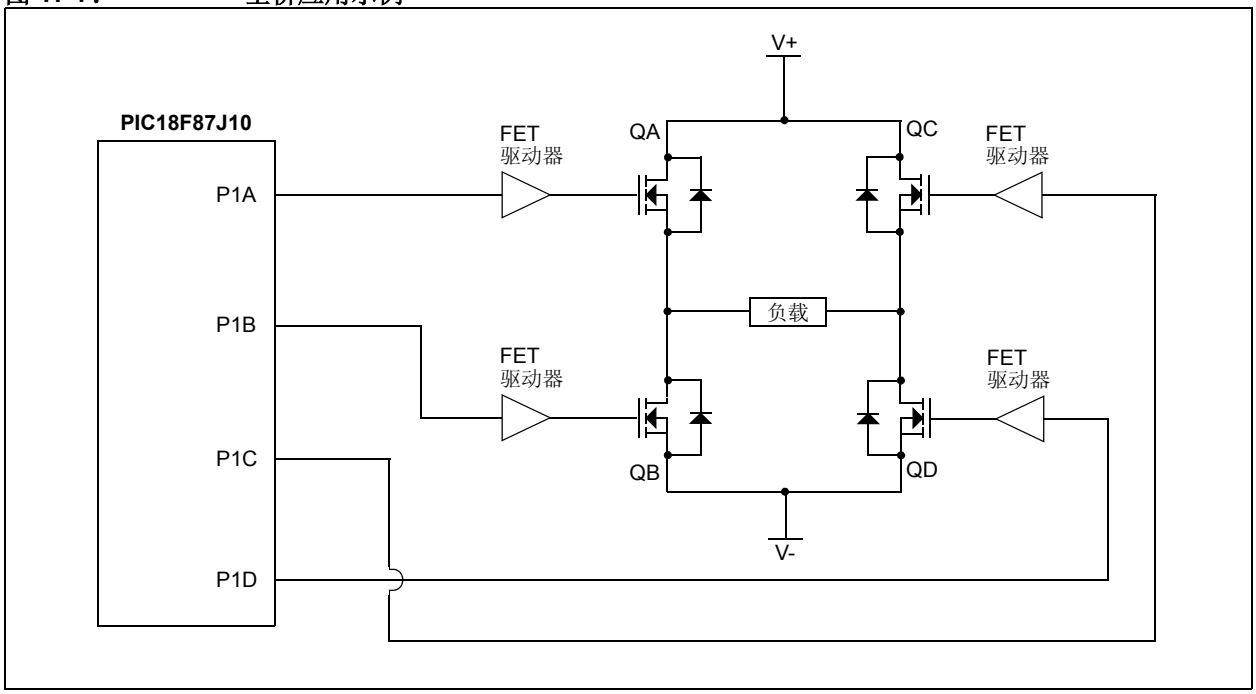
P1A、P1B、P1C 和 P1D 输出与表 17-1、表 17-2 和表 17-3 中的端口引脚复用。相应的 TRIS 位必须清零以使能 P1A、 P1B、 P1C 和 P1D 引脚作为输出。

图 17-6： 全桥 PWM 输出



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 17-7：　　全桥应用示例



17.4.5.1　全桥模式下的方向更改

在全桥输出模式下，CCP1CON 寄存器中的 P1M1 位使用户能控制正向或反向。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。

就在当前PWM周期结束之前，调制输出（P1B和P1D）被置于它们的无效状态，而非调制输出（P1A 和 P1C）被切换到以相反的方向驱动。这发生在下一个 PWM 周期开始前的一段时间间隔内 （4 Tosc * (Timer2 预分频值)）。 Timer2 预分频器的分频比将是 1、 4 或 16，这取决于 T2CKPS 位 （T2CON<1:0>）的值。从非调制输出开始切换输出方向到下一个周期开始之间的这段间隔内，调制输出 （P1B 和 P1D）保持无效。此关系如图 17-8 所示。

注意，在全桥输出模式下， ECCP1 模块不提供任何死区延迟。通常，由于在任何时间只调制一个输出，所以不需要死区延迟。然而，当以下两个条件都为真时，就可能需要死区延迟。

1. 当输出的占空比接近或等于 100% 时， PWM 输出的方向更改。
2. 电源开关（包括电源器件和驱动器电路）的关断时间大于导通时间。

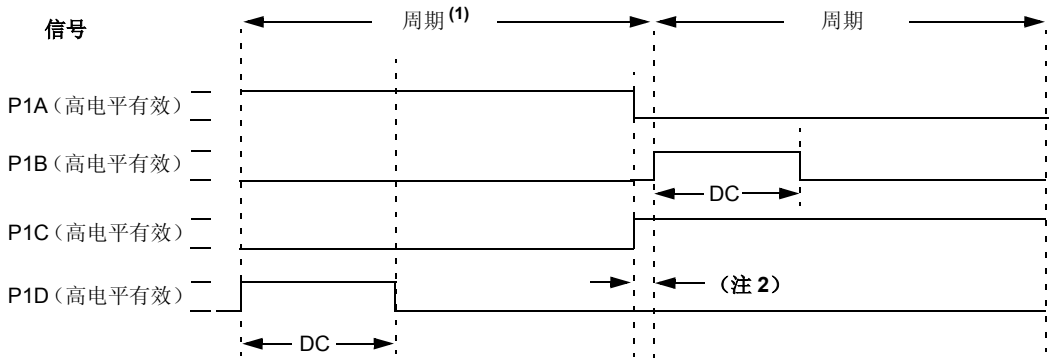
图 17-9 所示为在接近 100% 占空比时，PWM 方向从正向更改为反向的示例。在时间 t1 时，输出 P1A 和 P1D 变为无效，而输出 P1C 变为有效。在此示例中，由于电源器件的关断时间比导通时间长，直通电流可能在时间段 “t” 内流过电源器件 QC 和 QD （见图 17-7）。如果 PWM 方向从反向更改为正向，电源器件 QA 和 QB 将出现相同的现象。

如果需要在高占空比情况下更改 PWM 方向，必须满足以下要求之一：

1. 在更改方向前的那个 PWM 周期减小 PWM 的占空比。
2. 使用可使开关的关断速度比导通速度快的开关驱动器。

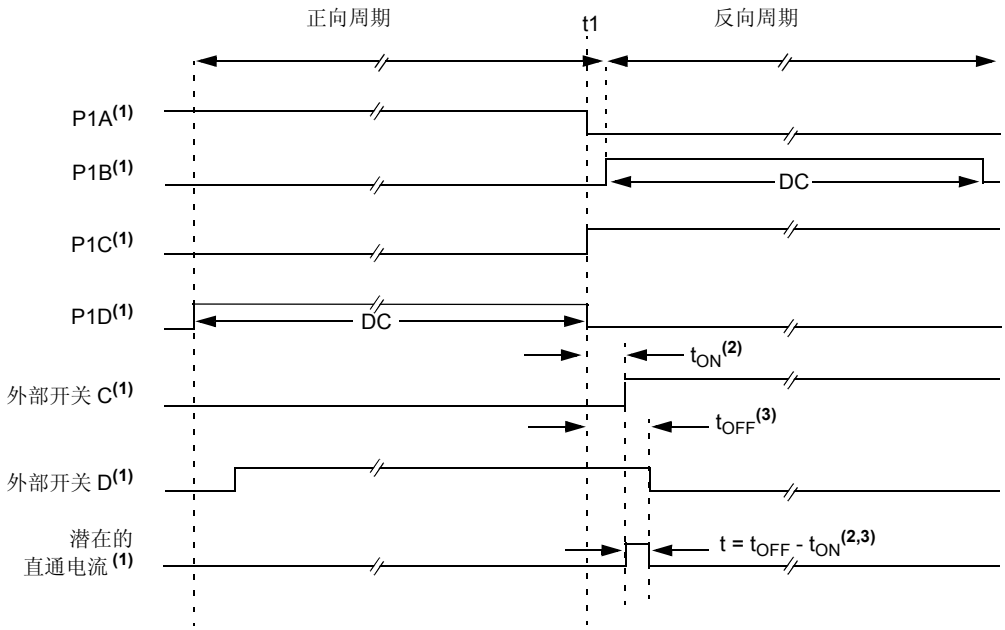
可能还存在防止直通电流的其他方法。

图 17-8：　　PWM 方向更改



注　1： 可在 PWM 周期内的任何时间写 ECCP1 控制寄存器中的方向位 （CCP1CON<7>）。
2： 当更改方向时， P1A 和 P1C 上的信号在当前 PWM 周期结束之前的一段时间间隔内进行切换，间隔可以是 4 TOSC、 16 TOSC 或 64 TOSC，这取决于 Timer2 的预分频器值。此时被调制的 P1B 和 P1D 信号为无效。

图 17-9：　　在接近 100% 占空比时 PWM 更改方向



注　1： 图示的所有信号为高电平有效。
2： t_{ON} 是电源开关 QC 及其驱动器的导通延迟时间。
3： t_{OFF} 是电源开关 QD 及其驱动器的关断延迟时间。

17.4.6 可编程死区延迟

在半桥应用中，模块一直以 PWM 频率的调制信号驱动电源开关，关断开关电源通常比使它导通需要更多的时间。如果上方的电源开关和下方的电源开关同时切换（一个导通，另一个关断），两个开关可能会在一段短时间内都处于导通状态，直到一个开关完全关断为止。在这很短的时间内，很大的电流（*直通电流*）可能流过两个电源开关，导致半桥供电电路短路。为了避免在切换期间流过这种潜在的破坏性直通电流，一般延迟打开其中的一个电源开关以使另一个开关完全关闭。

在半桥输出模式下，可数字编程的死区延迟可用来避免直通电流破坏电源开关。该延迟在进行从非有效状态到有效状态的信号转换时发生（见图 17-4）。ECCP1DEL 寄存器（寄存器 17-2）的低 7 位根据单片机指令周期（TCY 或 4 TOSC）设置延迟时间。

17.4.7 增强型 PWM 自动关闭

当 ECCP1 被编程为任何一种增强型 PWM 模式时，可以将有效输出引脚配置为自动关闭。当发生关闭事件时，自动关闭会立即将增强型 PWM 输出引脚置于定义的关闭状态。

关闭事件可以由两个比较器模块中的任意一个或FLT0引脚（或者这 3 个源的任意组合）引起。比较器可以用来监视电压输入，该电压与桥式电路中监视到的电流成正比。如果电压超过门限值，比较器将切换状态并触发关闭。另外， FLT0 引脚上的数字低电平信号也能触发关闭。通过不选择任何自动关闭源，可以禁止自动关闭功能。通过使用 ECCP1AS2:ECCP1AS0 位（ECCP1AS 寄存器的位 <6:4>）选择将使用的自动关闭源。

当关闭发生时，输出引脚陆续被置于它们的关闭状态，关闭状态由PSS1AC1:PSS1AC0和PSS1BD1:PSS1BD0位（ECCP1AS3:ECCP1AS0）指定。每个引脚对（P1A/P1C 和 P1B/P1D）可被设置为驱动高电平、驱动低电平或三态（不驱动）。还需置位 ECCP1ASE 位（ECCP1AS<7>）以便将增强型 PWM 输出保持在它们的关闭状态。

当关闭事件发生时，ECCP1ASE 被硬件置 1。如果不使能自动重新启动，当关闭源清除后， ECCP1ASE 位将被固件清零。如果使能自动重新启动，当自动关闭源清除后， ECCP1ASE 位将自动被清零。

如果当 PWM 周期开始时 ECCP1ASE 位置 1，PWM 输出将在这整个 PWM 周期内保持关闭状态。当 ECCP1ASE 位清零时，PWM 输出将在下一个 PWM 周期的开头返回到正常运行状态。

注：	当关闭条件有效时，禁止写 ECCP1ASE 位。

寄存器 17-2： ECCPxDEL：PWM 延迟寄存器

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

bit7 **PxRSEN**：PWM 重新启动使能位

1 = 自动关闭时，一旦关闭事件清除， ECCPxASE 位立即自动清零； PWM 自动重新启动

0 = 自动关闭时，必须用软件清零 ECCPxASE 位以重新启动 PWM

bit6-0 **PxDC6:PxDC0**：PWM 延迟计数位

延迟时间，以 FOSC/4 （4 * TOSC）周期为单位，为 PWM 信号变为有效的预定时间和实际时间之差。

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

寄存器 17-3：　ECCPxAS：增强型 CCPx 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

bit7　**ECCPxASE**：ECCP**x** 自动关闭事件状态位
0 = ECCPx 输出正常
1 = 已发生了关闭事件， ECCP**x** 输出处于关闭状态

bit6-4　**ECCPxAS2:ECCPxAS0**：ECCP**x** 自动关闭源选择位
000 = 自动关闭被禁止
001 = 比较器 1 输出
010 = 比较器 2 输出
011 = 比较器 1 或 2
100 = FLT0
101 = FLT0 或比较器 1
110 = FLT0 或比较器 2
111 = FLT0 或比较器 1 或比较器 2

bit3-2　**PSSxAC1:PSSxAC0**：引脚 A 和 C 的关闭状态控制位
00 = 将引脚 A 和 C 驱动为 0
01 = 将引脚 A 和 C 驱动为 1
1x = 引脚 A 和 C 为三态

bit1-0　**PSSxBD1:PSSxBD0**：引脚 B 和 D 的关闭状态控制位
00 = 将引脚 B 和 D 驱动为 0
01 = 将引脚 B 和 D 驱动为 1
1x = 引脚 B 和 D 为三态

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

17.4.7.1　自动关闭和自动重新启动

可以将自动关闭功能配置为在关闭事件后允许模块的自动重新启动。通过置位 ECCP1DEL 寄存器的 P1RSEN 位 （ECCP1DEL<7>）使能此功能。

在 P1RSEN = 1（图 17-10）的关闭模式下，只要关闭源继续存在，ECCP1ASE 位将保持置 1。当关闭条件清除时， ECCP1ASE 位清零。如果 P1RSEN = 0（图 17-11），一旦出现关闭条件，ECCP1ASE 位将保持置 1 直到它被固件清零为止。一旦 ECCP1ASE 被清零，增强型 PWM 将在下一个 PWM 周期的开头恢复运行。

注：	当关闭条件有效时，禁止写 ECCP1ASE 位。

如果自动关闭源是比较器，则关闭条件是电平，与 P1RSEN 位设置无关。只要关闭源继续存在，ECCP1ASE 位就不能被清零。

通过将 “1”写入 ECCP1ASE 位可以强制进入自动关闭模式。

17.4.8　启动注意事项

当在 PWM 模式下使用 ECCP1 模块时，应用硬件必须在 PWM 输出引脚上使用适当的上拉和 / 或下拉电阻。当单片机从复位释放时，所有的I/O引脚都处于高阻态。外部电路必须将电源开关元件保持在关断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。

CCP1M1:CCP1M0 位（CCP1CON<1:0>）允许用户为每对 PWM 输出引脚 （P1A/P1C 和 P1B/P1D）选择是高电平有效还是低电平有效。必须在 PWM 引脚被配置为输出之前选择 PWM 输出的极性。建议不要在 PWM 引脚被配置为输出时更改极性配置，因为这可能造成应用电路的损坏。

当初始化 PWM 模块时，P1A、P1B、P1C 和 P1D 的输出锁存器可能处于不正确的状态。使能 ECCP1 模块同时将 PWM 引脚使能为输出可能导致对应用电路的损害。必须将 ECCP1 模块使能为正确的输出模式并且在配置PWM引脚为输出之前完成一个完整的PWM周期。当第 2 个 PWM 周期开始时 TMR2IF 位会置 1，这可以表明完成了一个完整的 PWM 周期。

图 17-10： PWM 自动关闭 （P1RSEN = 1，使能自动重启）

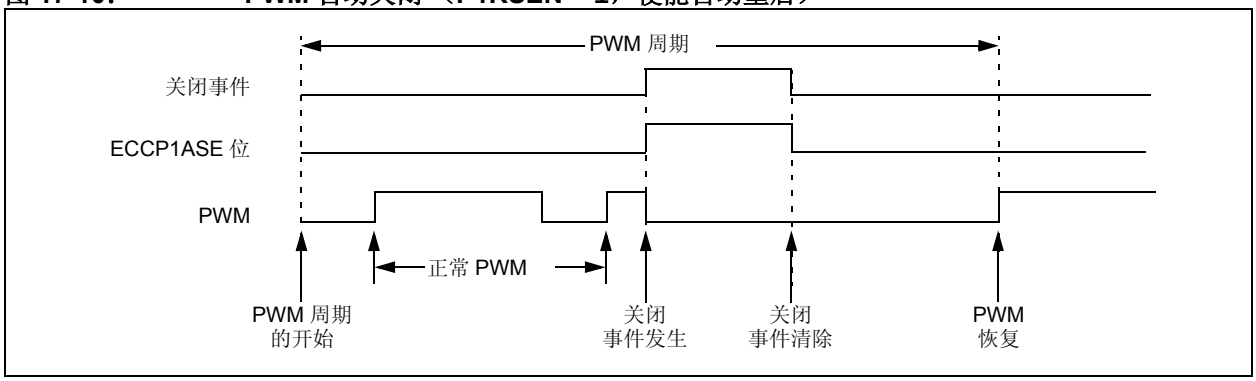
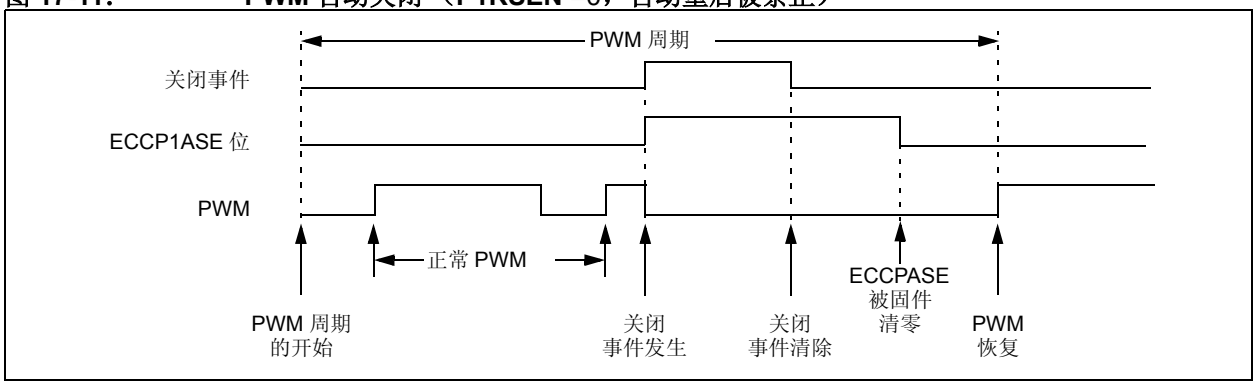


图 17-11： PWM 自动关闭 （P1RSEN =0，自动重启被禁止）



17.4.9 PWM 操作的设置

在为 PWM 操作配置 ECCPx 模块时应该执行以下步骤：

1. 通过将对应的 TRIS 位置 1，将 PWM 引脚 PxA 和 PxB （若使用 PxC 和 PxD，还包括这两个引脚）配置为输入。
2. 通过装载 PR2 （PR4）寄存器设置 PWM 周期。
3. 将 ECCPx 模块配置为所需的 PWM 模式，通过用相应的值装载 CCPxCON 寄存器来完成配置：
   - 用 PxM1:PxM0 位选择一种输出配置和方向。
   - 用 CCPxM3:CCPxM0 位选择 PWM 输出信号的极性。
4. 可通过装载 CCPRxL 寄存器和 CCPxCON<5:4> 位来设置 PWM 占空比。
5. 对于自动关闭：
   - 禁止自动关闭； ECCP1ASE = 0
   - 配置自动关闭源
   - 等待运行条件
6. 对于半桥输出模式，通过用相应的值装载 ECCPxDEL<6:0> 来设置死区延迟。
7. 如果需要自动关闭操作，则装载 ECCPxAS 寄存器：
   - 使用 ECCPxAS2:ECCPxAS0 位选择自动关闭源。
   - 使用 PSSxAC1:PSSxAC0 和 PSSxBD1:PSSxBD0 位选择 PWM 输出引脚的关闭状态。
   - 将 ECCPxASE 位 （ECCPxAS<7>）置 1。
8. 如果需要自动重新启动操作，则将 PxRSEN 位 （ECCPxDEL<7>）置 1。
9. 配置并启动 TMRn （TMR2 或 TMR4）：
   - 通过清零 TMRnIF 位 （Timer2 的 PIR1<1> 或 Timer4 的 PIR3<3>）来清零 TMRn 中断标志位。
   - 通过装载 TnCKPS 位 （TnCON<1:0>）来设置 TMRn 预分频值。
   - 通过将 TMRnON 位 （TnCON<2>）置 1 来使能 Timer2 （或 Timer4）。
10. 在新的 PWM 周期开始后，使能 PWM 输出：
    - 等待直到 TMRn 溢出为止 （TMRnIF 位置 1）。
    - 通过清零对应的 TRIS 位，将 ECCPx/PxA、PxB、 PxC 和 / 或 PxD 引脚使能为输出。
    - 清零 ECCPxASE 位 （ECCPxAS<7>）。

17.4.10 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 寄存器进入它们的复位状态。

这会强制增强型 CCP 模块复位为与标准 CCP 模块兼容的状态。

表 17-5：　　与 ECCP 模块、 TIMER1 到 TIMER4 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在的页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	50
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IF	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR2	OSCFIF	CMIF	—	—	BCL1IF	—	TMR3IF	CCP2IF	51
PIE2	OSCFIE	CMIE	—	—	BCL1IE	—	TMR3IE	CCP2IE	51
IPR2	OSCFIP	CMIP	—	—	BCL1IP	—	TMR3IP	CCP2IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	52
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	52
TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	52
TRISG	—	—	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0	52
TRISH	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	52
TMR1L	Timer1 寄存器的低字节								50
TMR1H	Timer1 寄存器的高字节								50
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	50
TMR2	Timer2 寄存器								50
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	50
PR2	Timer2 周期寄存器								50
TMR3L	Timer3 寄存器的低字节								51
TMR3H	Timer3 寄存器的高字节								51
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	$\overline{T3SYNC}$	TMR3CS	TMR3ON	51
TMR4	Timer4 寄存器								53
T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0	53
PR4	Timer4 周期寄存器								53
CCPRxL(1)	捕捉 / 比较 /PWM 寄存器 x 低字节								51
CCPRxH(1)	捕捉 / 比较 /PWM 寄存器 x 高字节								51
CCPxCON(1)	PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0	51
ECCPxAS(1)	ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0	51，53
ECCPxDEL(1)	PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0	53

图注：　— = 未用，读为 0。在 ECCP 运行期间不使用阴影单元。

注　1：　所有增强型 CCP 模块中具有此名称的相同寄存器的通用称谓，其中 “x”标识了各个模块 （ECCP1、 ECCP2 或 ECCP3）。具有同一个通用名称的所有寄存器的位分配和复位值相同。

18.0 主控同步串行端口 （MSSP）模块

18.1 主控 SSP （MSSP）模块概述

主控同步串行端口 （Master Synchronous Serial Port，MSSP）模块是用于同其他外设或单片机进行通信的串行接口。这些外设器件可能是串行 EEPROM、移位寄存器、显示驱动器或 A/D 转换器等。MSSP 模块有下列两种工作模式：

- 串行外设接口 （SPI™）
- 内部互联接口 （I^2C™）
  - 全主控模式
  - 从动模式 （广播地址呼叫）

I^2C 接口通过硬件支持下列模式：

- 主控模式
- 多主机模式
- 从动模式 （带地址屏蔽的 10 位和 7 位寻址模式）

PIC18F87J10 系列的所有器件都有两个 MSSP 模块，称为 MSSP1 和 MSSP2。每个模块都独立工作。

注：	在本章中，在所有工作模式下，通常指的 MSSP 模块都可以解释为 MSSP1 或 MSSP2。寄存器名称和模块 I/O 信号使用通用标识符 “x”（数字）来区分某个特定模块。控制位名称没有区别。

18.2 控制寄存器

每个 MSSP 模块有三个相关的控制寄存器，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各自的配置位将大相径庭。

下面各章节会提供其他细节。

注：	在具有多个 MSSP 模块的器件中，要特别注意 SSPCON 寄存器名称。 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。

18.3 SPI 模式

SPI 模式允许同时同步发送和接收 8 位数据。该系列器件支持 SPI 的所有四种模式。通常使用以下三个引脚来完成通信：

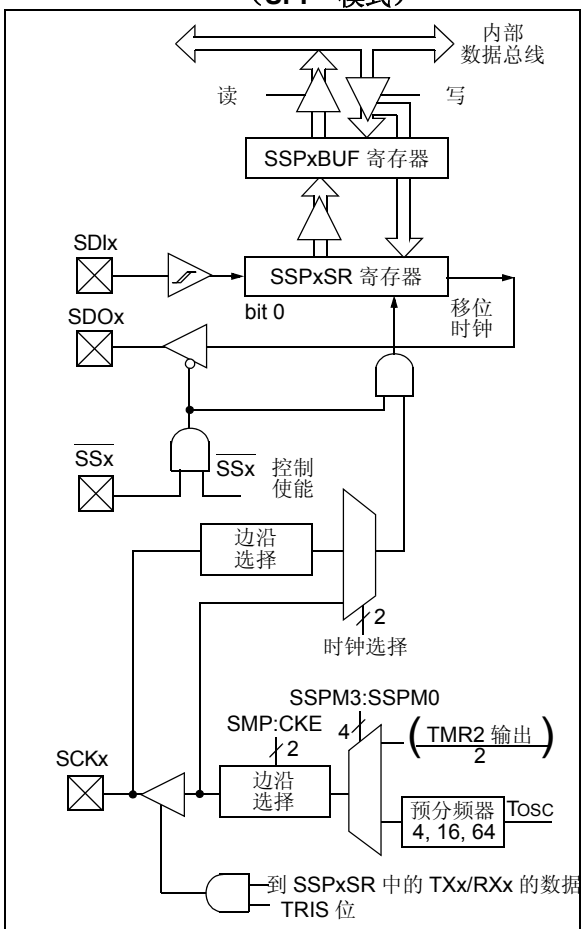
- 串行数据输出 （SDOx）—— RC5/SDO1 或 RD4/SDO2
- 串行数据输入 （SDIx）—— RC4/SDI1/SDA1 或 RD5/SDI2/SDA2
- 串行时钟 （SCKx）—— RC3/SCK1/SCL1 或 RD6/SCK2/SCL2

此外，当处于从动工作模式时可以使用第 4 根引脚：

- 从动选择 （$\overline{SSx}$）—— RF7/$\overline{SS1}$ 或 RD7/$\overline{SS2}$

图 18-1 给出了 MSSP 模块在 SPI 模式下工作时的原理框图。

图 18-1： MSSP 工作原理框图 （SPI™ 模式）



注：	在此框图中，为了简洁起见，仅使用 I/O 端口的名称。复用功能的完整名称请参见正文。

18.3.1 寄存器

MSSP 模块有四个寄存器用于在 SPI 模式下工作。它们是：

- MSSP 控制寄存器 1 （SSPxCON1）
- MSSP 状态寄存器 （SSPxSTAT）
- 串行接收 / 发送缓冲寄存器 （SSPxBUF）
- MSSP 移位寄存器 （SSPxSR）：不能直接访问

SSPxCON1 和 SSPxSTAT 是在 SPI 模式下工作的控制寄存器和状态寄存器。SSPxCON1 寄存器是可读写的。SSPxSTAT 的低六位是只读的，其高两位是可读写的。

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，可用于读写数据字节。

接收时，SSPxSR 和 SSPxBUF 共同构成一个双缓冲接收器。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 18-1： 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	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7 **SMP**：采样位
SPI 主控模式：
1 = 在数据输出时间的末尾采样输入数据
0 = 在数据输出时间的中间采样输入数据
SPI 从动模式：
当 SPI 在从动模式下使用时，必须将 SMP 清零。

bit 6 **CKE**：SPI 时钟选择位
1 = 当时钟从有效状态变为空闲状态时开始发送
0 = 当时钟从空闲状态变为有效状态时开始发送
注： 时钟状态的极性由 CKP 位 （SSPxCON1<4>）设置。

bit 5 **D/$\overline{A}$**：数据 / $\overline{地址}$位
仅在 I^2C 模式中使用。

bit 4 **P**：停止位
仅在 I^2C 模式中使用。当 MSSP 模块被禁止 （SSPEN 清零）时该位被清零。

bit 3 **S**：启动位
仅在 I^2C 模式中使用。

bit 2 **R/$\overline{W}$**：读 / 写信息位
仅在 I^2C 模式中使用。

bit 1 **UA**：更新地址位
仅在 I^2C 模式中使用。

bit 0 **BF**：缓冲器满状态位 （仅接收模式）
1 = 接收完成， SSPxBUF 已满
0 = 接收未完成， SSPxBUF 还为空

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

寄存器 18-2：　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	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位
1 = 当仍然在发送前一个字时发生了对 SSPxBUF 寄存器的写操作（必须用软件清零）
0 = 未发生冲突

bit 6 **SSPOV**：接收溢出指示位
SPI 从动模式：
1 = SSPxBUF 寄存器仍在保存前一数据时，接收到一个新的字节。一旦发生接收溢出，SSPxSR 中的数据会丢失。接收溢出只会在从动模式下发生。即使只是发送数据，用户也必须读 SSPxBUF，以避免置位溢出指示位 （必须由软件清零）。
0 = 无溢出

注： 在主控模式下，溢出位不会被置 1。因为每次接收和发送新数据，开始都要写入 SSPxBUF 寄存器。

bit 5 **SSPEN**：主控同步串行口使能位
1 = 使能串行端口并将 SCKx、 SDOx、 SDIx 和 $\overline{SSx}$ 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

注： 当该位使能时，应将相应的引脚正确配置为输入或输出。

bit 4 **CKP**：时钟极性选择位
1 = 时钟高电平为空闲状态
0 = 时钟低电平为空闲状态

bit 3-0 **SSPM3:SSPM0**：主控同步串行口模式选择位
0101 = SPI 从动模式，时钟 = SCKx 引脚，禁止 $\overline{SSx}$ 引脚控制，可将 $\overline{SSx}$ 用作 I/O 引脚
0100 = SPI 从动模式，时钟 = SCKx 引脚，使能 $\overline{SSx}$ 引脚控制功能
0011 = SPI 主控模式，时钟 = TMR2 输出 /2
0010 = SPI 主控模式，时钟 = Fosc/64
0001 = SPI 主控模式，时钟 = Fosc/16
0000 = SPI 主控模式，时钟 = Fosc/4

注： 这里未列出的位组合为保留的或只在 I^2C 模式中使用。

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

18.3.2 工作方式

当初始化 SPI 时，需要指定几个选项。可以通过编程相应的控制位 （SSPxCON1<5:0> 和 SSPxSTAT<7:6>）来指定。这些控制位用于指定以下选项：

- 主控模式 （SCKx 作为时钟输出）
- 从动模式 （SCKx 作为时钟输入）
- 时钟极性 （SCKx 的空闲状态）
- 输入数据的采样相位 （数据输出时间的中间或末端）
- 时钟边沿 （在 SCKx 上升沿 / 下降沿输出数据）
- 时钟速率 （仅用于主控模式）
- 从动选择模式 （仅用于从动模式）

MSSP 模块由一个发送 / 接收移位寄存器 （SSPxSR）和一个缓冲寄存器 （SSPxBUF）组成。 SSPxSR 将数据移入和移出器件，最高有效位在前。 SSPxBUF 保存上次写入 SSPxSR 的数据直到新接收到的数据就绪为止。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 被置 1。这种双重缓冲数据接收方式 （SSPxBUF）允许在 CPU 读取刚接收的数据之前，就开始接收下一个字节。在数据发送 / 接收期间，任何试图写 SSPxBUF 寄存器的操作都会被忽略，并将写冲突检测位 WCOL （SSPxCON<7>）置 1。此时用户必须用软件将 WCOL 位清零，否则无法判别下一次对 SSPxBUF 的写操作是否成功完成。

应用软件要想接收有效数据，应该在下一个要传送的数据写入 SSPxBUF 之前，将 SSPxBUF 中的前一个数据读出。缓冲器满标志位 BF （SSPxSTAT<0>）用于表示何时 SSPxBUF 已经载入了接收到的数据（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，则不必理会接收的数据。通常可用 MSSP 中断来判断发送或接收何时完成。如果不使用中断来处理数据的收发，用软件查询方法同样可确保不会发生写冲突。例 18-1 显示了装载 SSPxBUF （SSPxSR）进行数据发送的过程。

用户不能直接读写 SSPxSR 寄存器，只能通过访问 SSPxBUF 寄存器访问到它。此外， SSPxSTAT 寄存器表示各种状态信号。

例 18-1： 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
```

18.3.3 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，要先将 SSPEN 位清零，重新初始化 SSPxCON 寄存器，然后将 SSPEN 位置 1。这将把 SDIx、SDOx、SCKx 和 $\overline{SSx}$ 引脚配置为串行端口引脚。要将这些引脚用于串行端口功能，还必须将其数据方向位（在 TRIS 寄存器中）正确编程，方法如下：

- SDIx 由 SPI 模块自动控制
- SDOx 必须将 TRISC<5> 或 TRISD<4> 位清零
- SCKx（主控模式）必须将 TRISC<3> 或 TRISD<6> 位清零
- SCKx（从动模式）必须将 TRISC<3> 或 TRISD<6> 位置 1
- $\overline{SSx}$ 必须将 TRISF<7> 或 TRISD<7> 位置 1

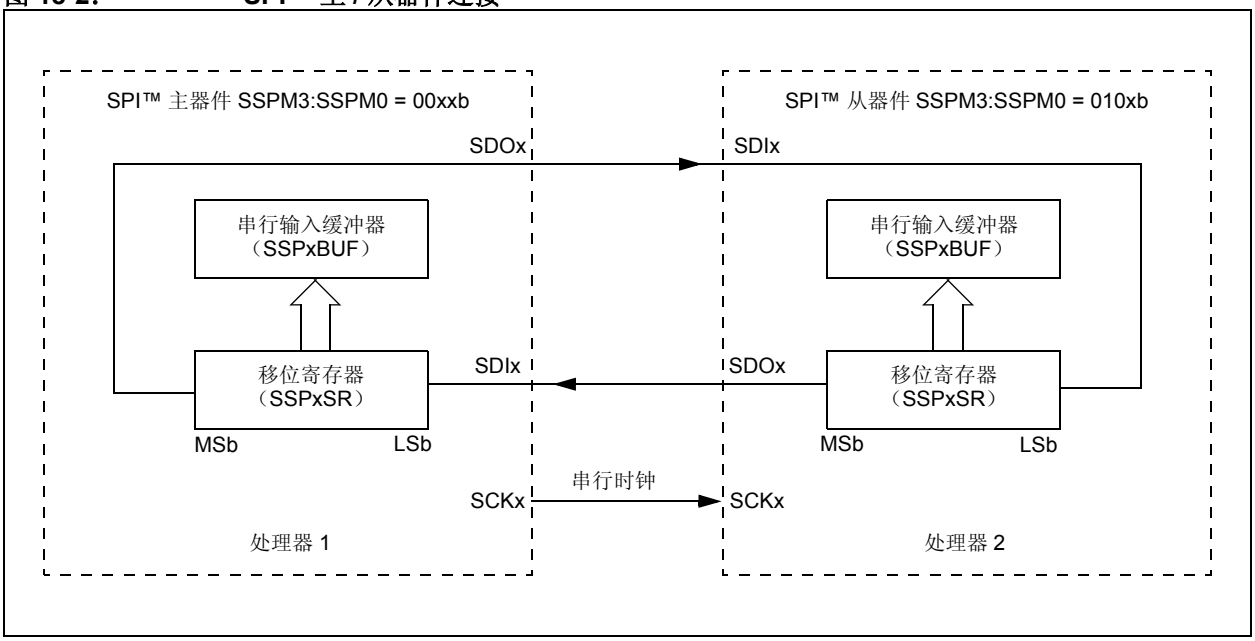
对于不需要的任何串行端口功能，可通过将对应的数据方向（TRIS）寄存器设置为相反值来改写。

18.3.4 典型连接

图 18-2所示为两个单片机之间的典型连接。主器件（处理器 1）通过发送 SCKx 信号来启动数据传输。数据在编程设定的时钟边沿被移出两个处理器的移位寄存器，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）编程设定为相同，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的情况：

- 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 18-2： SPI™ 主 / 从器件连接



18.3.5 主控模式

主器件可以随时启动数据传输，因为它控制 SCKx。主器件根据软件协议确定从器件（处理器 1，图 18-2）应在何时广播数据。

在主控模式下，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（将其编程设定为输入）。SSPxSR 寄存器按设置的时钟速率，对 SDIx 引脚上的信号进行连续移位输入。每个字节接收完后，会被送入 SSPxBUF 寄存器，就像普通的接收字节一样（相应的中断和状态位置位）。这在以“线路操作监控”（Line Activity Monitor）方式工作的接收器应用中很有用。

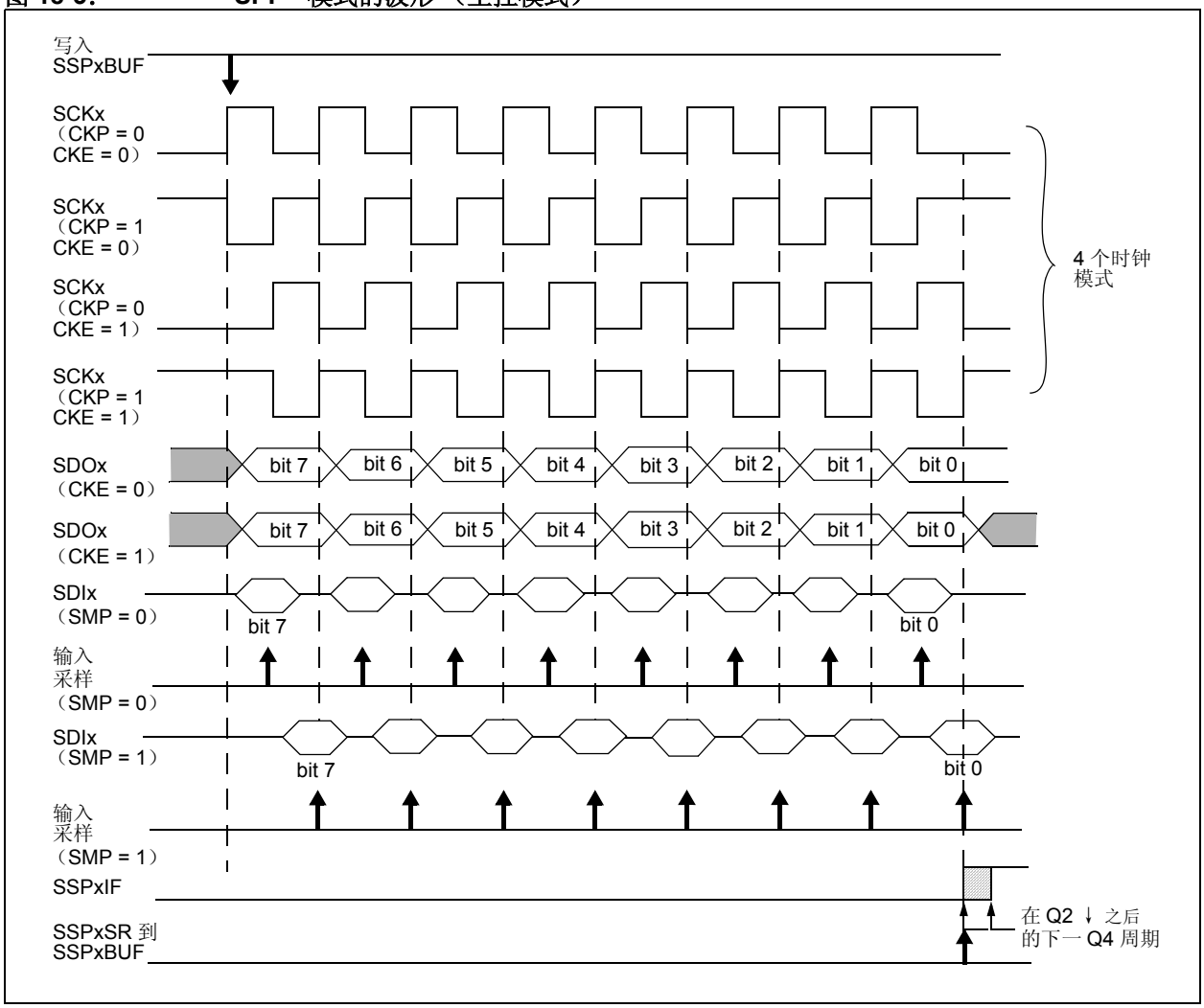
可通过对 CKP 位（SSPxCON1<4>）进行相应的编程来选择时钟极性。图 18-3、图 18-5 和图 18-6 将给出 SPI 通信波形图，其中首先发送的是 MSB。在主控模式下，SPI 时钟速率（位速率）可由用户编程设定为下面几种方式之一：

- FOSC/4（或 TCY）
- FOSC/16（或 4 • TCY）
- FOSC/64（或 16 • TCY）
- Timer2 输出 /2

这里允许的最大数据速率是 10.00 Mbps（当晶振为 40 MHz 时）。

图 18-3 给出了主控模式的波形图。当 CKE 位置 1 时，SDOx 数据在 SCKx 上出现时钟边沿前就有效。图中所示输入采样的变化由 SMP 位的状态决定。图中指出了将接收到的数据装入 SSPxBUF 的时间。

图 18-3： SPI™ 模式的波形（主控模式）



18.3.6 从动模式

在从动模式下，当 SCKx 引脚上出现外部时钟脉冲时，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。

在从动模式下，外部时钟来自于 SCKx 引脚上的外部时钟源。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。

在休眠状态下，从器件仍可发送 / 接收数据。可以将器件配置为当收到一个字节时，从休眠状态中唤醒。

18.3.7 从动选择同步

$\overline{SSx}$ 引脚允许同步从动模式。SPI 必须工作在从动模式下，并使能 $\overline{SSx}$ 引脚控制（SSPxCON1<3:0> = 04h）。当 $\overline{SSx}$ 引脚为低电平时，使能数据的发送和接收，同时 SDO 引脚被驱动。当 $\overline{SSx}$ 引脚为高电平时，即使是在数据的发送过程中，SDOx 引脚也不再被驱动，而是变成悬浮输出。根据应用的需要，可外接上拉/下拉电阻。

注 1： 当 SPI 工作在从动模式下，并且 $\overline{SSx}$ 引脚控制使能（SSPxCON1<3:0> = 0100）时，如果 $\overline{SSx}$ 引脚置为 VDD 电平将使 SPI 模块复位。

2： 如果 SPI 工作在从动模式下并且 CKE 置 1，则必须使能 $\overline{SSx}$ 引脚控制。

当 SPI 模块复位后，位计数器被强制归 0。这可以通过强制将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，可以仿真双线制通信。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（SDI 功能）。

图 18-4： 从动同步波形

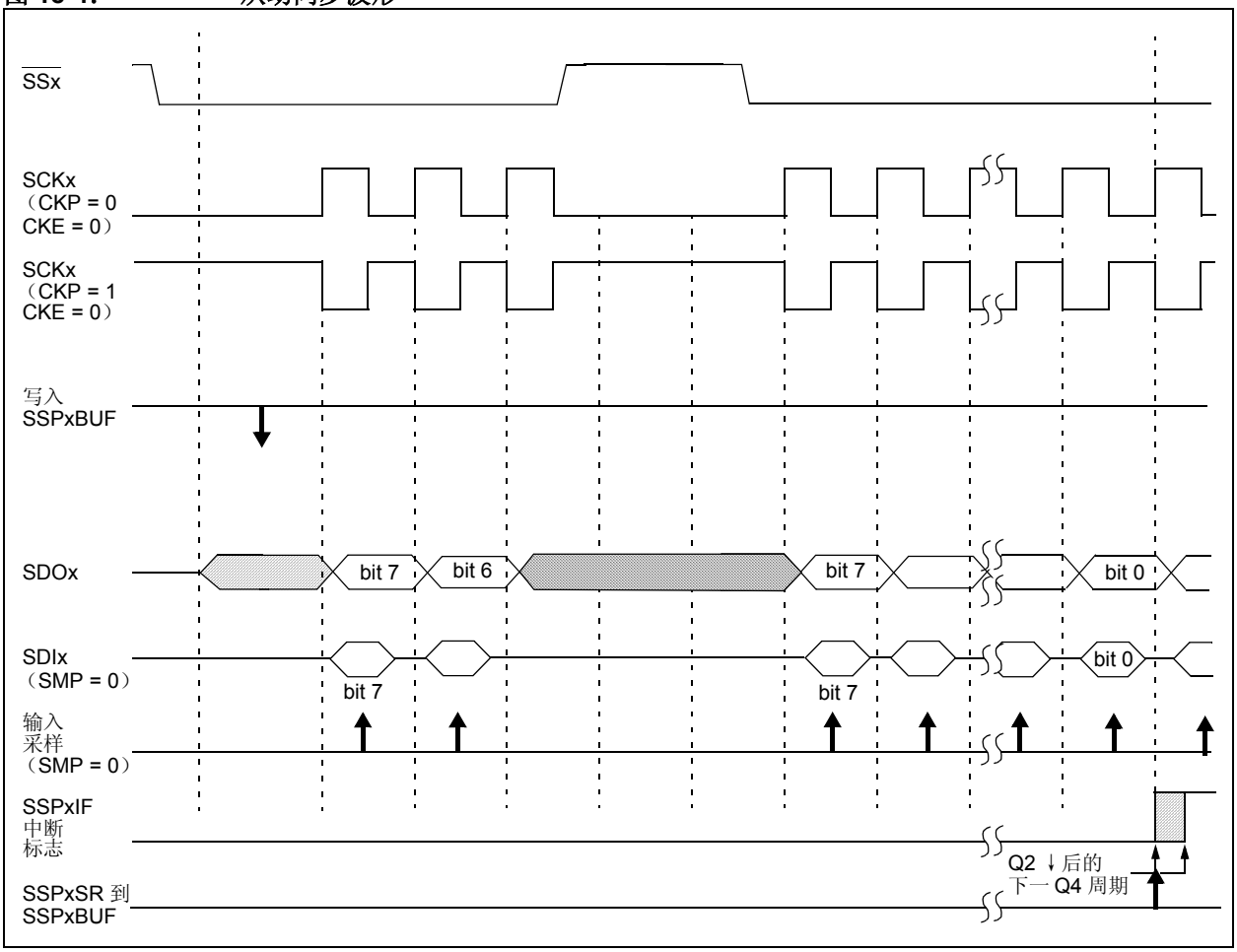


图 18-5：　　SPI™ 模式的波形 （从动模式， CKE = 0）

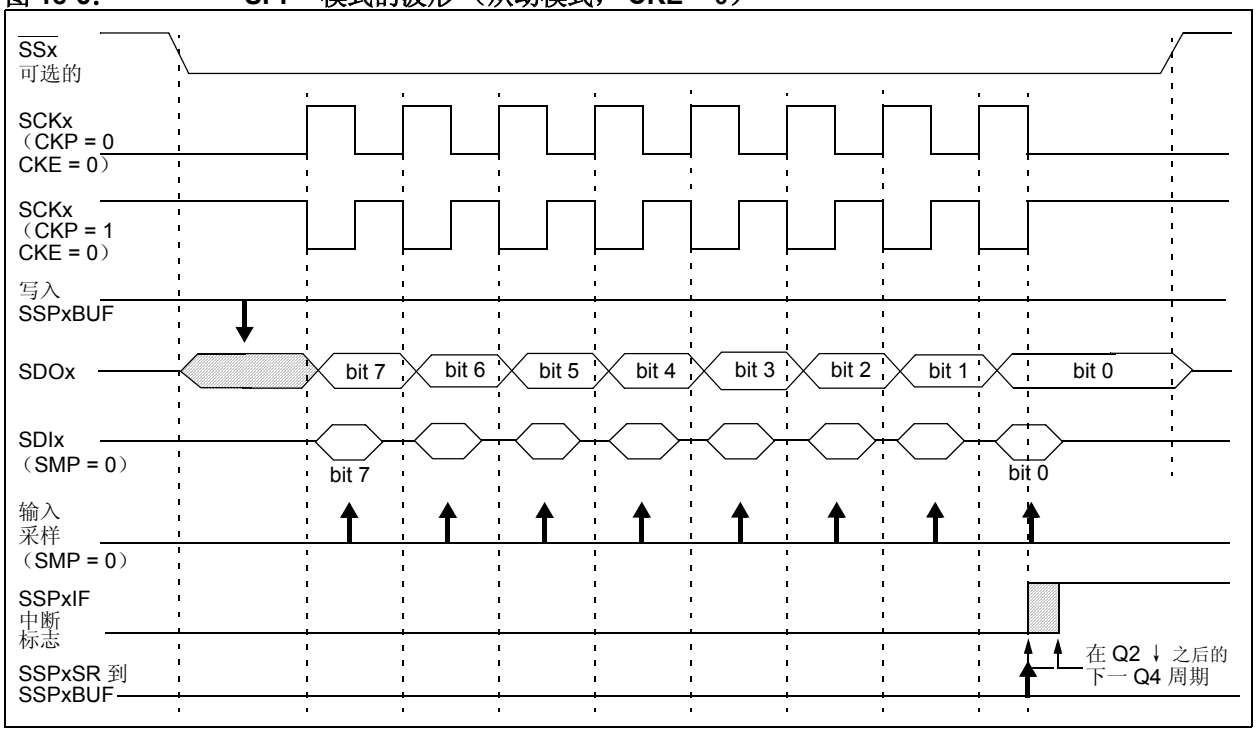
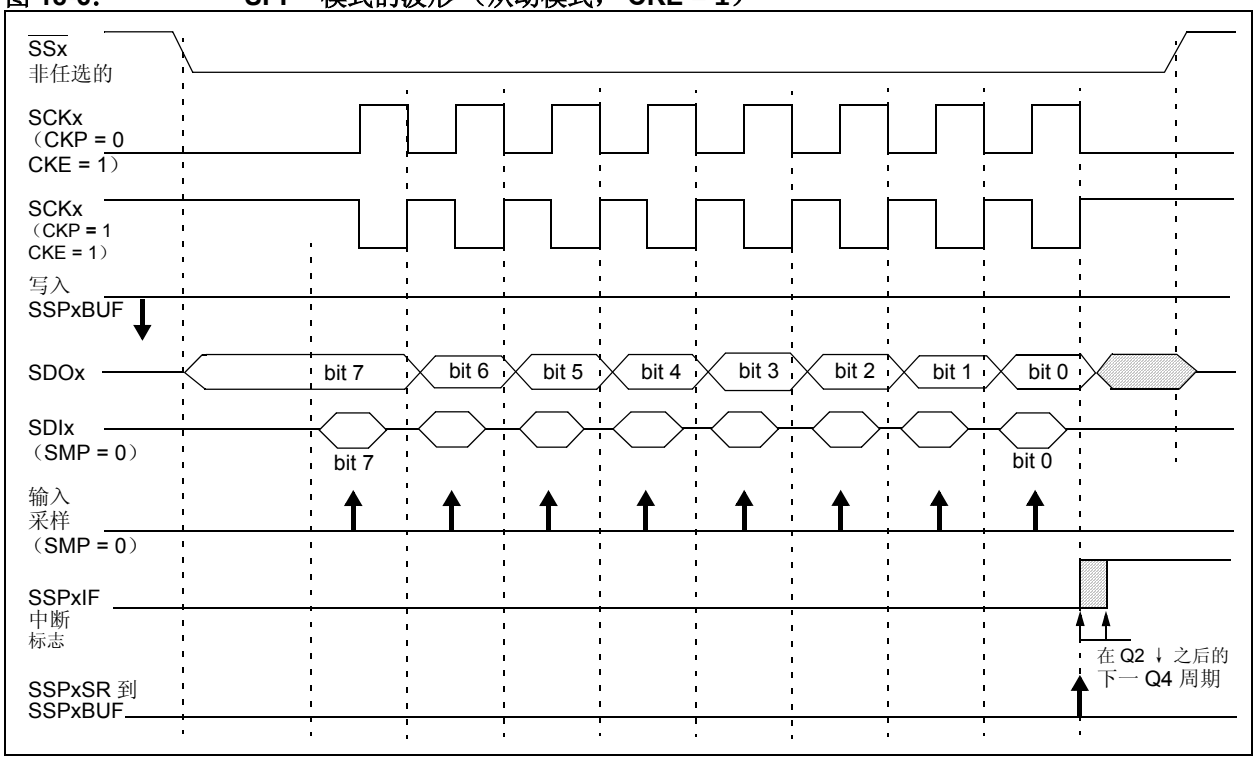


图 18-6：　　SPI™ 模式的波形 （从动模式， CKE = 1）



18.3.8 在功耗管理模式下的工作方式

在 SPI 主控模式下，模块时钟速度可以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。此时钟可以来自于主时钟源、辅助时钟源（Timer1 振荡器）或 INTOSC 时钟源。更多的信息请参见**第 2.6 节“时钟源和振荡器切换”**。

在大多数情况下，主机为 SPI 数据提供的时钟速度并不重要；但是，每个系统都应该评估此因素。

如果允许了 MSSP 中断，那么当主机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 从动模式下， 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，并且如果使能的话，将唤醒器件。

18.3.9 复位的影响

复位会禁止 MSSP 模块并停止当前的数据传输。

18.3.10 总线模式兼容性

表 18-1 中所示是标准 SPI 模式和 CKP 与 CKE 控制位状态的兼容情况。

表 18-1： SPI 总线模式

标准 SPI 模式术语	控制位状态	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

还有一个 SMP 位用来控制数据的采样时间。

18.3.11 SPI 时钟速度和模块相互关系

因为 MSSP1 和 MSSP2 是独立的模块，它们可以以不同的数据速率同时工作。将 SSPxCON1 寄存器的 SSPM3:SSPM0 位置 1 将确定相应模块的速率。

有一种例外情况就是在两个模块都在主控模式下使用 Timer2 作为时基的时候。在这种情况下，任何对 Timer2 工作模式的更改都会对两个 MSSP 模块造成相同的影响。如果每个模块需要不同的位速率，用户应该为一个模块选择其他三种时基中的一种。

表 18-2： 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	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	52
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	52
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	52
SSP1BUF	MSSP1 接收缓冲器 / 发送寄存器								50
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	50，53
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	50，53
SSP2BUF	MSSP2 接收缓冲器 / 发送寄存器								53

图注： 阴影单元在 SPI™ 模式下的 MSSP 模块中未用。

18.4 I²C 模式

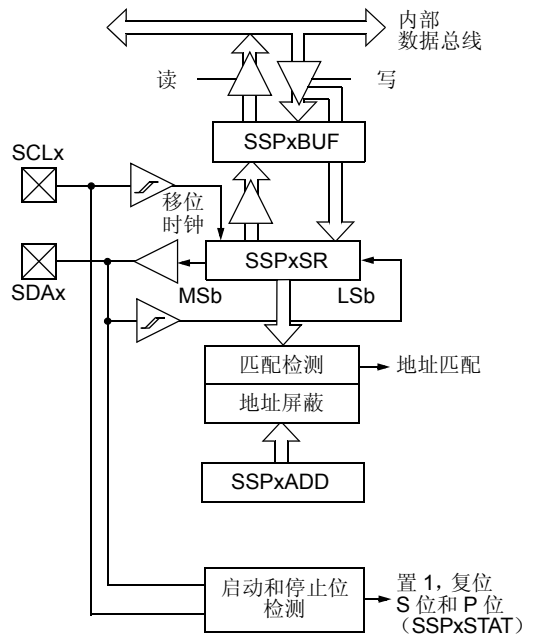
MSSP 模块工作在 I²C 模式时，可以实现所有的主控和从动功能（包括广播呼叫支持），并且用硬件提供启动位和停止位的中断来判断总线何时空闲（多主控功能）。MSSP 模块实现了标准模式规范，以及 7 位寻址和 10 位寻址。

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

用户必须通过将对应的 TRIS 位置 1，将这些引脚配置为输入引脚。

图 18-7：　　MSSP 框图（I²C™ 模式）



注： 在此框图中，为了简洁起见，仅使用 I/O 端口的名称。复用功能的完整名称请参见正文。

18.4.1 寄存器

MSSP 模块有 6 个寄存器用于 I²C 操作。它们是：

- MSSP 控制寄存器 1（SSPxCON1）
- MSSP 控制寄存器 2（SSPxCON2）
- MSSP 状态寄存器（SSPxSTAT）
- 串行接收 / 发送缓冲寄存器（SSPxBUF）
- MSSP 移位寄存器（SSPxSR）：不能直接访问
- MSSP 地址寄存器（SSPxADD）

SSPxCON1、SSPxCON2 和 SSPxSTAT 是在 I²C 模式下工作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 位是只读的，其高 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，可用于读写数据字节。

在 I²C 从动模式下配置 MSSP 时，SSPxADD 寄存器将保存从器件地址。在主控模式下配置 MSSP 时，SSPxADD 的低 7 位用作波特率发生器的重载值。

接收时，SSPxSR 和 SSPxBUF 共同构成一个双缓冲接收器。当 SSPxSR 接收到一个完整的字节后，该字节被送入 SSPxBUF 寄存器，同时置位中断标志位 SSPxIF。

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 18-3：　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	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7　**SMP**：变化率控制位
在主控或从动模式下：
1 = 禁止变化率控制以适应标准速度模式 （100 kHz 和 1 MHz）
0 = 使能变化率控制以适应高速模式 （400KHz）

bit 6　**CKE**：SMBus 选择位
在主控或从动模式下：
1 = 使能 SMBus 特定输入
0 = 禁用 SMBus 特定输入

bit 5　**D/$\overline{A}$**：数据 / $\overline{地址}$位
在主控模式下：
保留。
在从动模式下：
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 = 表示最近检测到了停止位
0 = 表示最近未检测到停止位

注：　当 SSPEN 被清零时该位在复位时清零。

bit 3　**S**：启动位
1= 表示最近检测到了启动位
0= 表示最近未检测到启动位

注：　当 SSPEN 被清零时该位在复位时清零。

bit 2　**R/$\overline{W}$**：读 / $\overline{写}$信息位
在从动模式下：
1 = 读
0 = 写

注：　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 $\overline{ACK}$ 位被检测到的期间有效。

在主控模式下：
1= 正在进行发送
0= 不在进行发送

注：　该位与 SEN、RSEN、PEN、RCEN 或 ACKEN 进行“或”运算的结果表示 MSSP 是否处于活动模式。

bit 1　**UA**：更新地址位 （仅用于 10 位从动模式）
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 已满
0 = SSPxBUF 为空
在接收模式下：
1 = SSPxBUF 已满 （不包括 $\overline{ACK}$ 位和停止位）
0 = SSPxBUF 为空 （不包括 ACK 位和停止位）

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

寄存器 18-4：　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	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位
在主控发送模式下：
1 = 当 I²C 信号无效时，尝试写 SSPxBUF 寄存器 （必须用软件清零）
0 = 未发生冲突
在从动发送模式下：
1 = 当仍然在发送前一个字时发生了对 SSPxBUF 寄存器的写操作 （必须用软件清零）
0 = 未发生冲突
在接收模式 （主控或从动模式）下：
这是 “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，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，此位是 “无关”位。

bit 5 **SSPEN**：主控同步串行口使能位
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

注：　当该位被使能时，必须将 SDAx 和 SCLx 引脚配置为输入引脚。

bit 4 **CKP**：SCKx 释放控制位
在从动模式下：
1= 释放时钟
0= 保持时钟低电平 （延长低电平时间），用来确保数据建立时间
在主控模式下：
在此模式下未使用。

bit 3-0 **SSPM3:SSPM0**：主控同步串行口模式选择位
1111 = I²C 从动模式， 10 位地址，并允许启动和停止位中断
1110 = I²C 从动模式， 7 位地址，并允许启动和停止位中断
1011 = I²C 由固件控制的主控模式 （从动空闲）
1000 = I²C 主控模式，时钟 =FOSC/(4 * (SSPxADD + 1))
0111 = I²C 从动模式， 10 位地址
0110 = I²C 从动模式， 7 位地址
未列出的位组合为保留的或只在 SPI™ 模式下使用。

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

寄存器 18-5：　SSPxCON2：MSSPx 控制寄存器 2 （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
GCEN	ACKSTAT	ACKDT/ ADMSK5	ACKEN(1)/ ADMSK4	RCEN(1)/ ADMSK3	PEN(1)/ ADMSK2	RSEN(1)/ ADMSK1	SEN(1)
bit 7							bit 0

bit 7　**GCEN**：广播呼叫使能位 （仅用于从动模式）
1 = 当在 SSPxSR 中接收到广播呼叫地址 （0000h）时允许中断
0 = 禁止广播呼叫地址

bit 6　**ACKSTAT**：应答状态位 （仅用于主控发送模式）
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/ADMSK5**：应答数据位
在主控接收模式下：
1 = 不应答
0 = 应答

注：　用户在接收结束时启动一个应答序列，同时发送该值。

在从动模式下：
1 = 使能 ADD5 地址屏蔽
0 = 禁止 ADD5 地址屏蔽

bit 4　**ACKEN/ADMSK4**：应答序列使能位
在主控接收模式下：(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
在从动模式下：
1 = 使能 ADD4 地址屏蔽
0 = 禁止 ADD4 地址屏蔽

bit 3　**RCEN/ADMSK3**：接收使能位
在主控接收模式下：(1)
1 = 使能 I²C 接收模式
0 = 接收空闲
在从动模式下：
1 = 使能 ADD3 地址屏蔽
0 = 禁止 ADD3 地址屏蔽

bit 2　**PEN/ADMSK2**：停止条件使能位
在主控模式下：(1)
1 = 在 SDAx 和 SCLx 引脚上启动停止条件。由硬件自动清零。
0 = 停止条件空闲
在从动模式下：
1 = 使能 ADD2 地址屏蔽
0 = 禁止 ADD2 地址屏蔽

bit 1　**RSEN/ADMSK1**：重复启动条件使能位
在主控模式下：(1)
1 = 在 SDAx 和 SCLx 引脚上开始重复启动条件。由硬件自动清零。
0 = 重复启动条件空闲
在从动 （7 位寻址）模式下：
1 = 使能 ADD1 地址屏蔽
0 = 禁止 ADD1 地址屏蔽
在从动 （10 位寻址）模式下：
1 = 使能 ADD0 和 ADD1 地址屏蔽
0 = 禁止 ADD0 和 ADD1 地址屏蔽

bit 0 **SEN**：启动条件使能 / 延长使能位 (1)

在主控模式下：
1 = 在 SDAx 和 SCLx 引脚上开始启动条件。由硬件自动清零。
0 = 启动条件空闲

在从动模式下：
1 = 为从动发送和从动接收使能时钟延长 （延长已使能）
0 = 时钟延长被禁止。

注 1：对于 ACKEN、RCEN、PEN、RSEN 和 SEN 位，如果 I^2C 模块处于激活状态，可能这些位不会被置1（没有缓存）并且也可能不会写入SSPxBUF（或禁止写SSPxBUF）。

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

寄存器 18-6： SSPxADD：MSSP1 和 MSSP2 地址寄存器 (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
ADD7	ADD6	ADD5	ADD4	ADD3	ADD2	ADD1	ADD0
bit 7							bit 0

bit 7 **ADD<7:0>**：MSSP 地址位

注 1： 在 I^2C 从动模式下 MSSP1 和 MSSP2 的地址寄存器。在 I^2C 主控模式下 MSSP1 和 MSSP2 的波特率重载寄存器。

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

18.4.2 工作方式

将 MSSP 使能位 SSPEN（SSPxCON1<5>）置位，可使能 MSSP 模块功能。

SSPxCON1 寄存器可以控制 I^2C 操作。可通过设置（SSPxCON1<3:0>）选择以下几种 I^2C 模式之一：

- I^2C 主控模式，时钟
- I^2C 从动模式 （7 位地址）
- I^2C 从动模式 （10 位地址）
- I^2C 从动模式 （7 位地址），允许启动位和停止位中断
- I^2C 从动模式 （10 位地址），允许启动位和停止位中断
- 由 I^2C 固件控制的主动模式，从器件空闲

通过置位相应的 TRISC 或 TRISD 位将 SCLx 和 SDAx 引脚编程为输入引脚，在 SSPEN 位置位时选择任何 I^2C 模式，这样将强制 SCLx 和 SDAx 引脚为漏极开路。要确保此模块的正常工作，必须为 SCLx 和 SDAx 引脚提供外接上拉电阻。

18.4.3 从动模式

在从动模式下， SCLx 引脚和 SDAx 引脚必须被配置为输入（TRISC<4:3> 置 1）。必要时（从发送器）MSSP 模块将使用输出数据改写输入状态。

在 I^2C 从动模式下，硬件会在一个地址发生匹配时产生一个中断。地址屏蔽功能则可使硬件在多个地址发生匹配时（7 位寻址模式下多达 31 个，10 位寻址模式下多达 63 个）产生一个中断。用户也可以通过模式选择位选择启动位或停止位中断。

当地址匹配时或在地址匹配后传送的数据被接收时，硬件会自动产生一个应答（$\overline{ACK}$）脉冲，并把当时 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：

- 缓冲器满标志位 BF （SSPxSTAT<0>）在接收到发送数据前置位。
- 在接收到发送的数据之前，溢出标志位 SSPOV （SSPxCON1<6>）已被置 1。

在这种情况下， SSPxSR 寄存器的值不会载入 SSPxBUF，但是 SSPxIF 位会置 1。 BF 位是通过读取 SSPxBUF 寄存器清零的，而 SSPOV 位必须通过软件来清零。

为确保正常工作， SCLx 时钟输入必须满足最小高电平时间和最小低电平时间。关于 I^2C 规范所规定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。

18.4.3.1 寻址

一旦 MSSP 模块被使能，它就会等待启动条件产生。在启动条件出现后，8 位数据被移入 SSPxSR 寄存器。在时钟 （SCLx）线的上升沿采样所有的输入位。寄存器 SSPxSR<7:1> 的值会和 SSPxADD 寄存器的值比较，该比较是在第 8 个时钟脉冲 （SCLx）下降沿进行的。如果地址匹配，并且 BF 位和 SSPOV 位被清零，会发生下列事件：

1. SSPxSR 寄存器值装入 SSPxBUF 寄存器。
2. 缓冲器满标志位 BF 置位。
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 置位 （如果允许中断则发生中断）。

在 10 位地址模式下，从器件需要接收两个地址字节。第一个地址字节的高 5 位指定这是否为 10 位地址。R/$\overline{W}$ 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 位地址的工作顺序如下，其中 7-9 步是针对从动发送器而言的。

1. 接收地址的第一个（高）字节 （在地址匹配时，SSPxIF 位、 BF 位和 UA 位置 1）。
2. 用地址的第二个（低）字节刷新 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 位置位）。
5. 使用地址的第一个（高）字节刷新 SSPxADD 寄存器。如果匹配释放 SCLx 线，并将 UA 位清零。
6. 读 SSPxBUF （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 位置位）。
9. 读 SSPxBUF （BF 位清零）并将标志位 SSPxIF 清零。

18.4.3.2 地址屏蔽

将地址的某一位屏蔽意味着该位可以是任意值（0 或 1），此时会对两个地址响应并产生一个中断。由于同一时刻可以有多个地址位被屏蔽，所以在 7 位模式下可响应多达 31 个地址，而在 10 位模式下则可响应多达 63 个地址（见例 18-2）。

不管是否使用地址屏蔽，I^2C 从器件的工作方式保持不变。当使用地址屏蔽时，I^2C 从器件能够响应多个地址并产生中断，此时需要通过查询 SSPxBUF 来判断是哪一个地址引起的中断。

- 7 位地址模式

  地址屏蔽位 ADMSK<5:1> 可用来屏蔽 SSPxADD 寄存器中对应的地址位。如果 ADMSK 的某位是有效的（ADMSK<n> = 1），则对应的地址位可以被忽略（ADD<n> = x）。对于发出地址应答的模块来讲，只要与没被屏蔽的地址位匹配就可以了。

- 10 位地址模式

  地址屏蔽位 ADMSK<5:2> 可用来屏蔽 SSPxADD 寄存器中对应的地址位，而 ADMSK<1> 可以同时屏蔽地址的低 2 位 ADD<1:0>。如果 ADMSK 的某位是有效的（ADMSK<n> = 1），则对应的地址位可以被忽略（ADD<n> = x）。需要注意的是，尽管在 10 位地址模式下，地址的高位也要用到 SSPxADD 寄存器中的某些位，但地址屏蔽位对这些位不起作用，地址屏蔽位只会影响地址低字节中的位。

注 **1**：ADMSK<1> 位屏蔽地址的低 2 位。
2：地址屏蔽不会对地址的高 2 位起作用。

例 18-2： 地址屏蔽

7 位寻址模式：

SSPxADD<7:1> = 1010 0000

ADMSK<5:1> = 00 111

可被应答的地址 = 0xA0, 0xA2, 0xA4, 0xA6
0xA8, 0xAA, 0xAC, 0xAE

10 位寻址模式：

SSPxADD<7:0> = 1010 0000 （此例中地址高 2 位被忽略，因为它们不受影响。）

ADMSK<5:1> = 00 111

可被应答的地址 = 0xA0, 0xA1, 0xA2, 0xA3
0xA4, 0xA5, 0xA6, 0xA7
0xA8, 0xA9, 0xAA 0xAB
0xAC, 0xAD, 0xAE, 0xAF

高 2 位不受地址屏蔽的影响。

18.4.3.3 接收

当地址字节的 R/$\overline{W}$ 位清零并发生地址匹配时，SSPxSTAT 寄存器的 R/W 位清零。接收的地址被装入 SSPxBUF 寄存器，且 SDAx 数据线保持低电平（$\overline{ACK}$）。

当存在地址字节溢出条件时，则不会产生应答脉冲（$\overline{ACK}$）。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）。必须通过将 CKP 位置 1（SSxPCON1<4>）才能释放时钟。更多详情，请参见**第 18.4.4 节 “时钟延长”**。

18.4.3.4 发送

当接收的地址字节的 R/$\overline{W}$ 位置 1 并发生地址匹配时，SSPxSTAT 寄存器的 R/W 位置 1。接收到的地址将装入 SSPxBUF 寄存器。不管 SEN 的值如何， $\overline{ACK}$ 脉冲在第 9 位上发送，同时 SCLx 引脚保持低电平（如需了解更多细节，请参见**第 18.4.4 节 “时钟延长”**）。通过延长时钟，主器件只有在从器件准备发送数据时才能发出另一个时钟脉冲。传输的数据必须装入 SSPxBUF 寄存器，同时也被装入 SSPxSR 寄存器。然后，应该通过置位 CKP（SSPxCON1<4>）使能 SCLx 引脚。8 个数据位在 SCLx 输入的下降沿移出。这可确保在 SCLx 为高电平时 SDAx 信号是有效的 （如图 18-10）。

主接收器的 $\overline{ACK}$ 脉冲将在 SCLx 输入第 9 个脉冲的上升沿锁存。如果 SDAx 线为高电平（无 $\overline{ACK}$），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 $\overline{ACK}$，将复位从动逻辑（复位 SSPxSTAT 寄存器），同时从器件监视下一个启动位的出现。如果 SDAx 线为低电平 （$\overline{ACK}$），则必须将接下去要发送的数据装入 SSPxBUF 寄存器。必须通过置位 CKP 将 SCLx 再次使能。

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 状态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 18-8： I^2C™ 从动模式接收时序 （SEN = 0， 7 位地址）

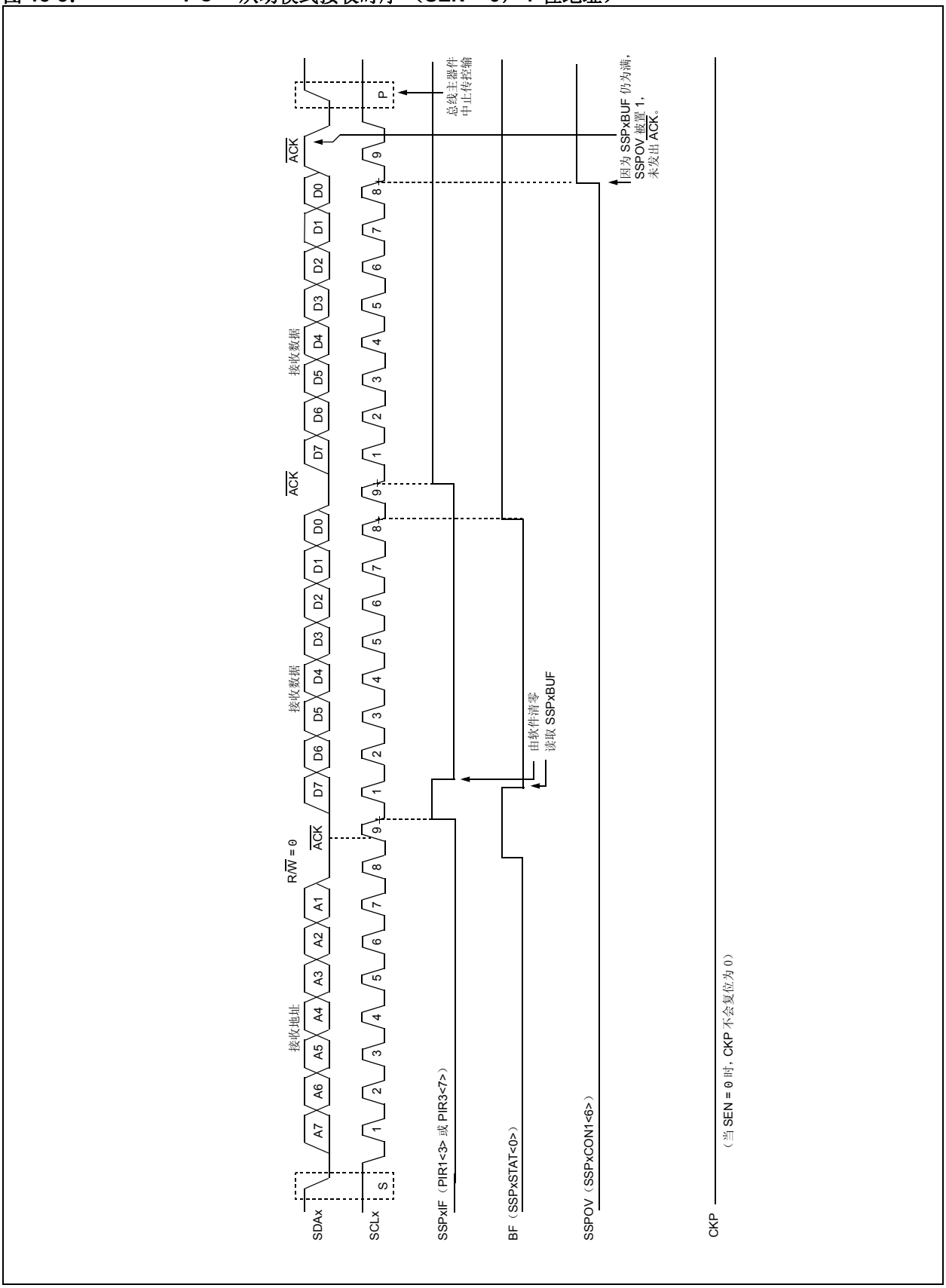
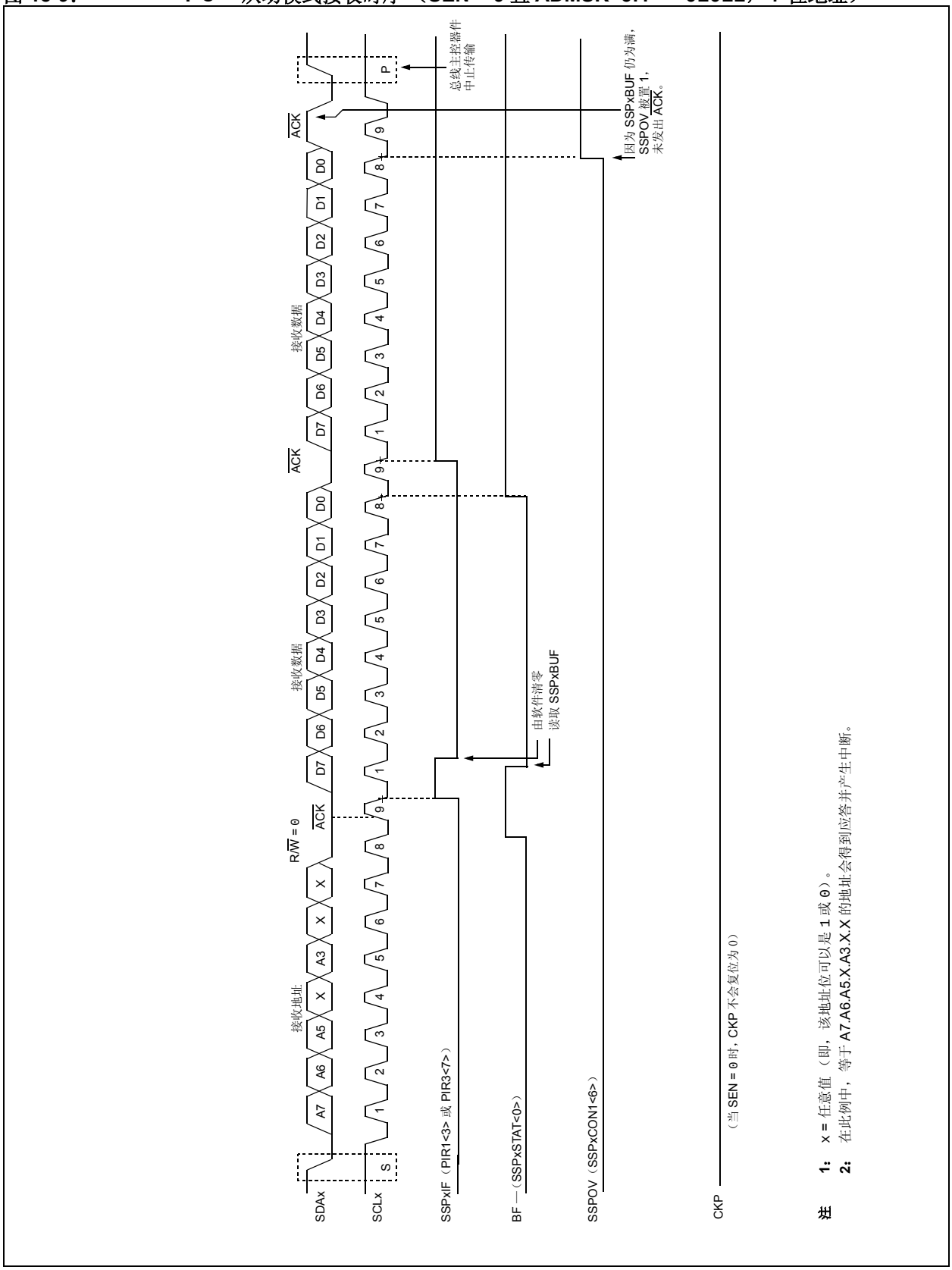


图 18-9： I^2C™ 从动模式接收时序 （SEN = 0 且 ADMSK<5:1> = 01011， 7 位地址）



注 1： x = 任意值 （即，该地址位可以是 1 或 0）。
2： 在此例中，等于 A7.A6.A5.X.A3.X.X 的地址会得到应答并产生中断。

图 18-10：　　I²C™ 从动模式发送时序 （7 位地址）

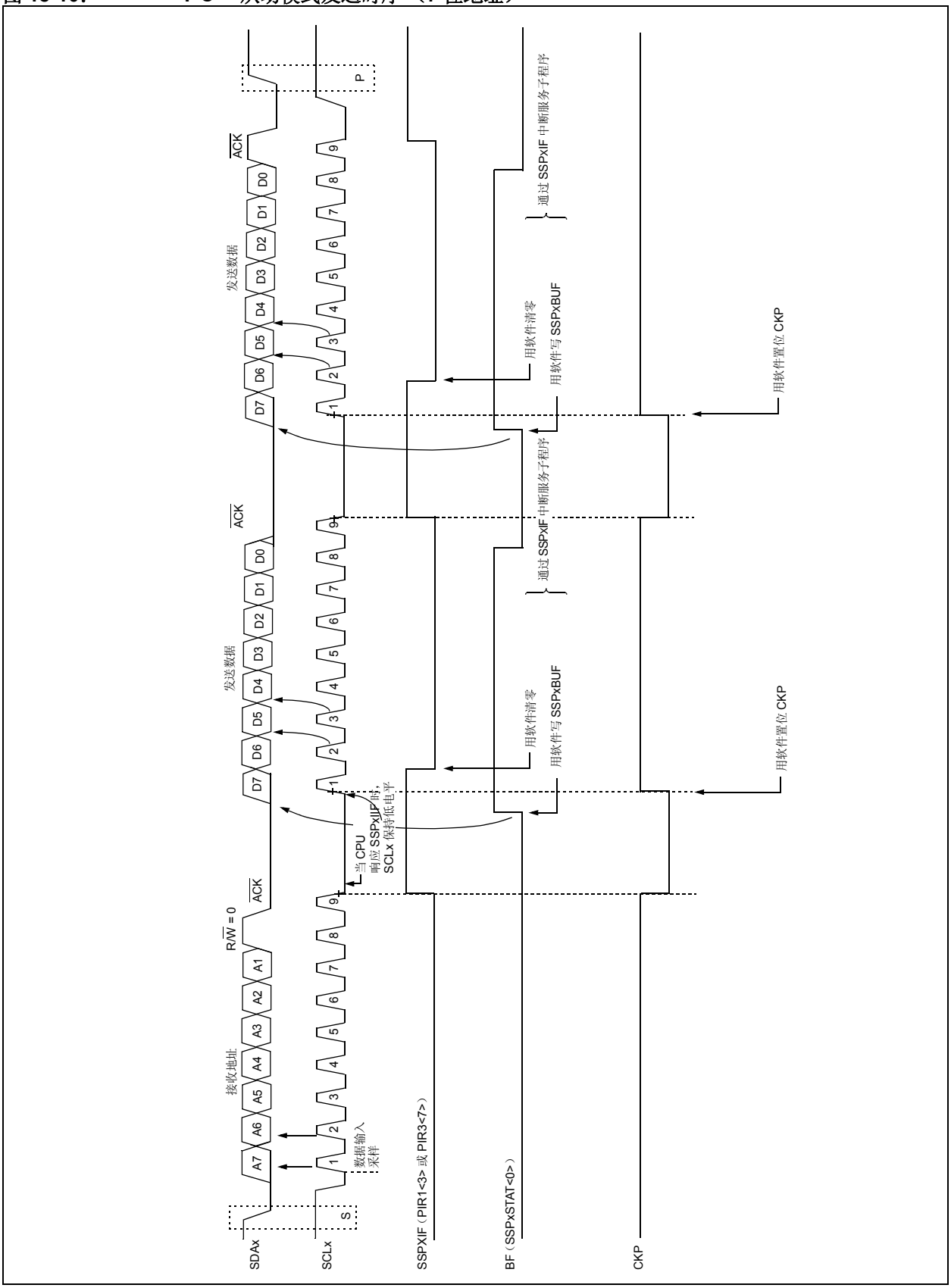


图 18-11： I²C™ 从动模式接收时序 （SEN = 0 且 ADMSK<5:1> = 01001 ， 10 位地址）

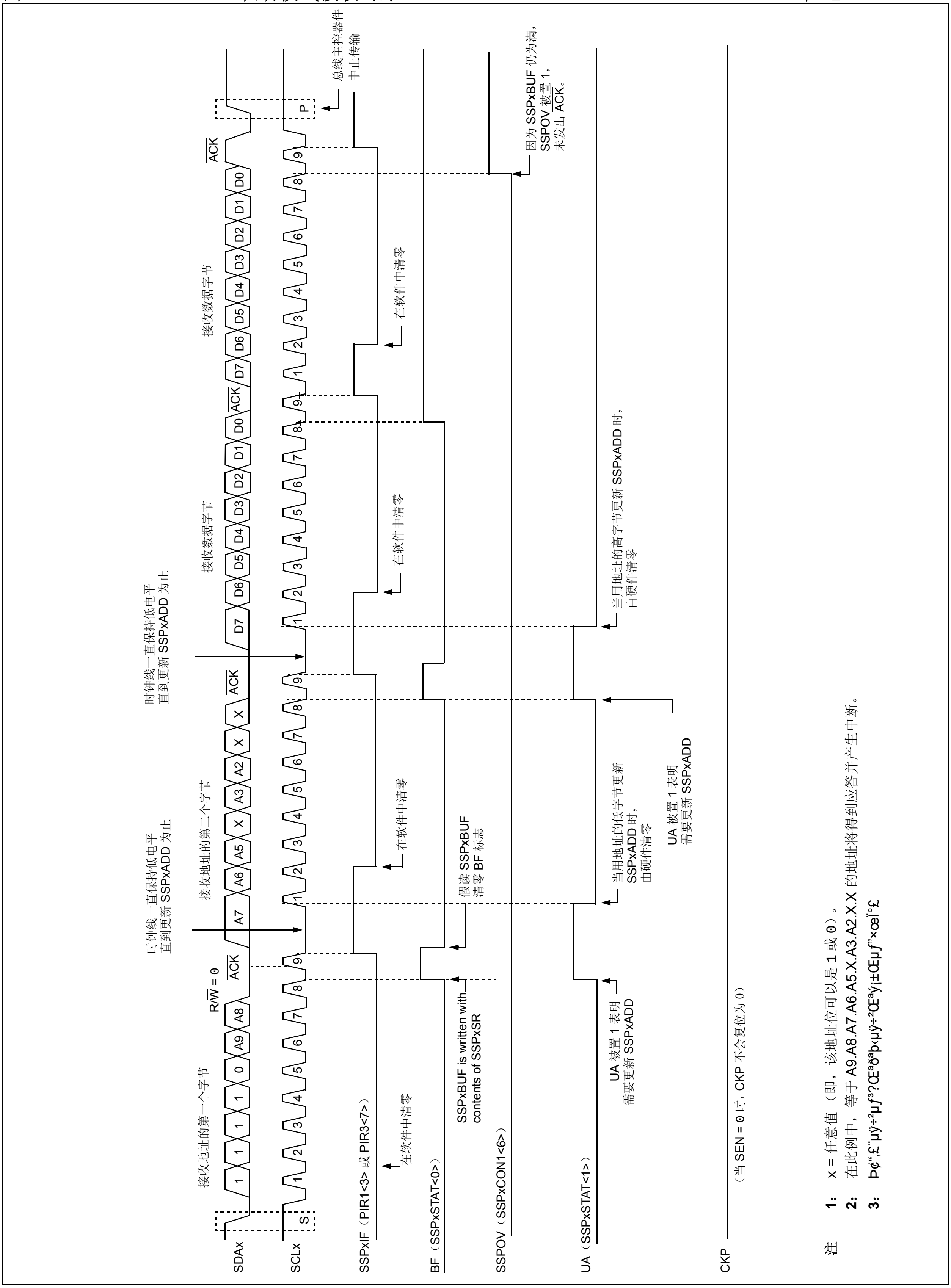


图 18-12：　　I²C™ 从动模式接收时序 （SEN = 0， 10 位地址）

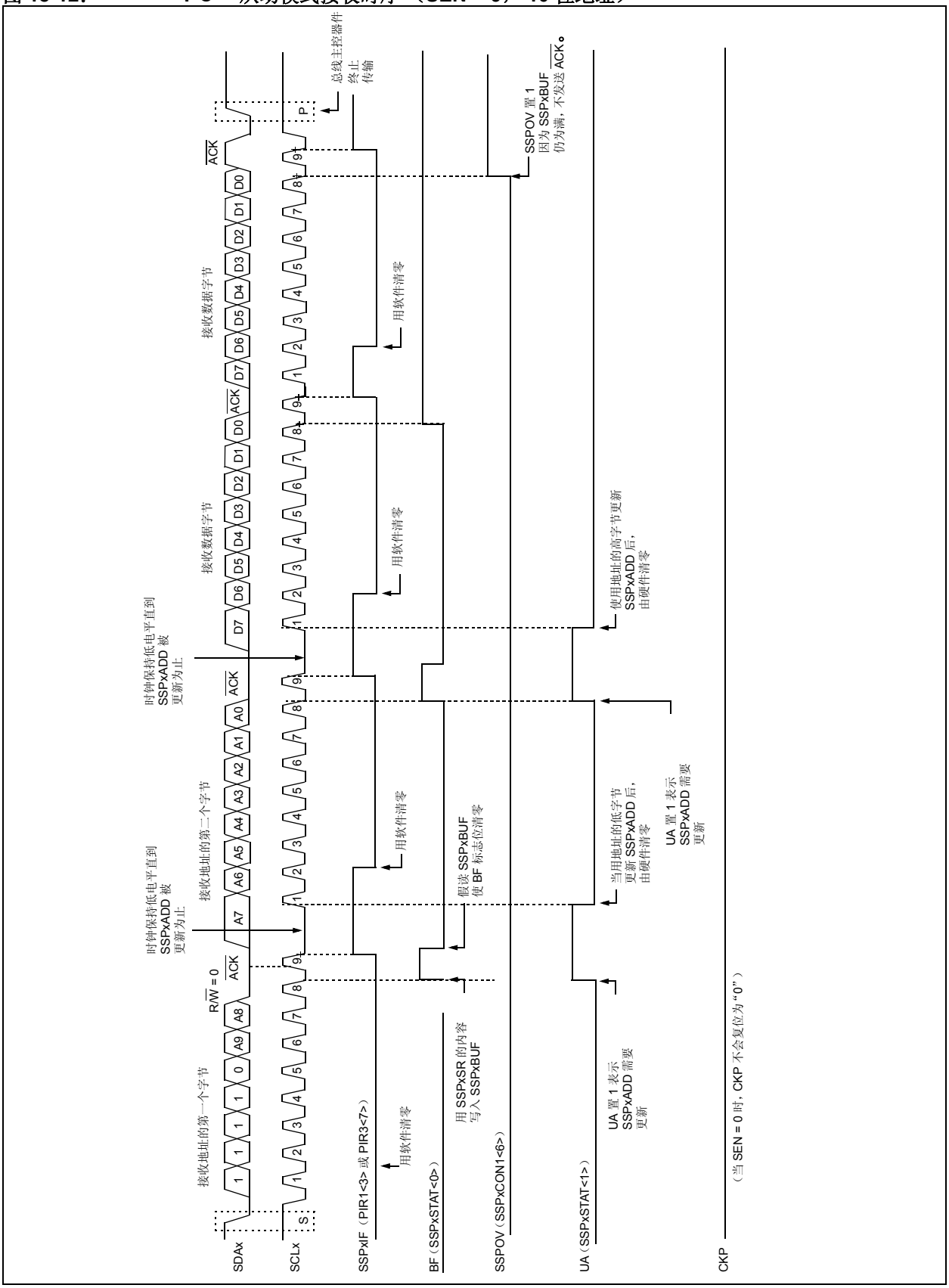
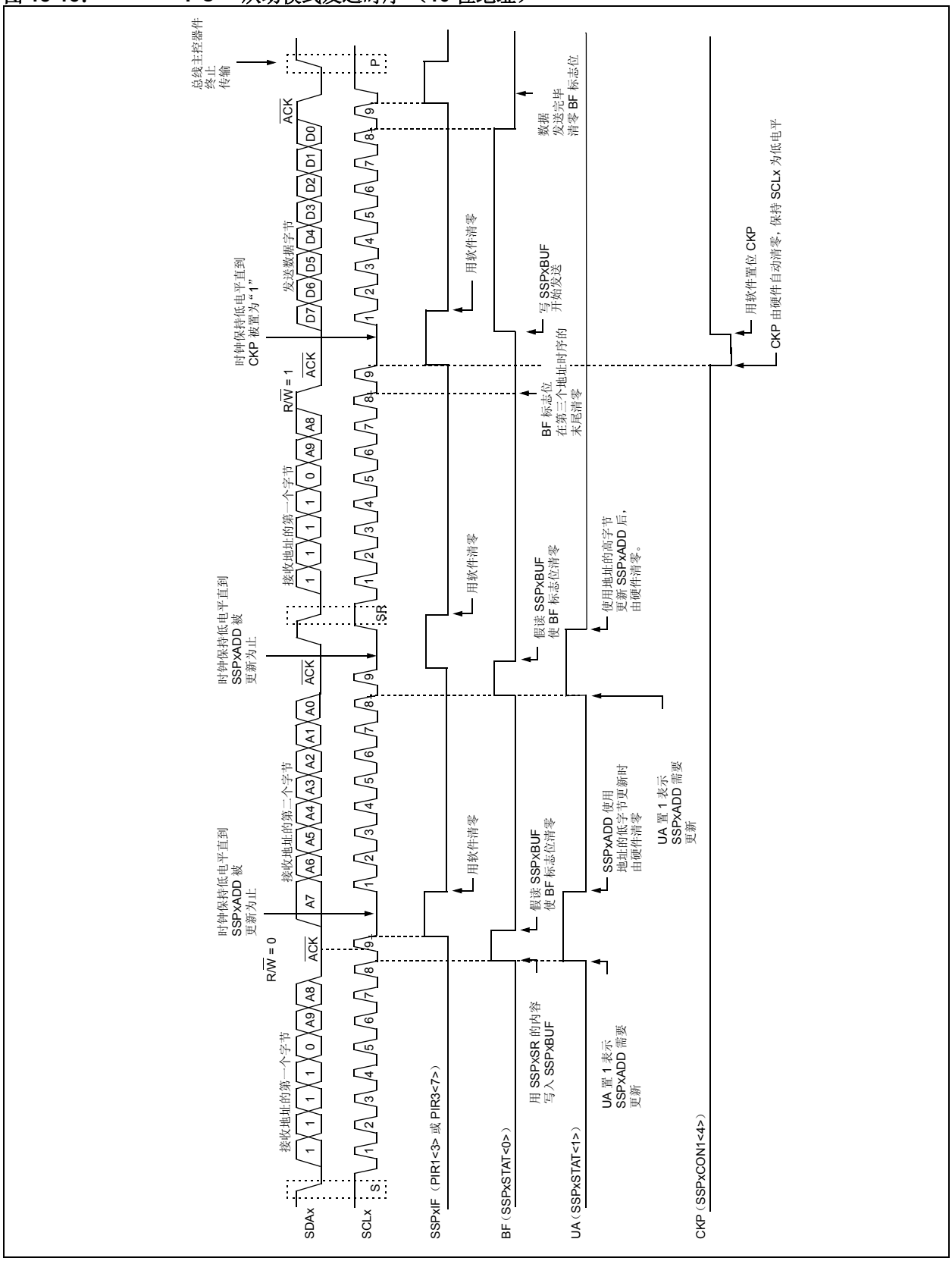


图 18-13：　　I^2C™ 从动模式发送时序 （10 位地址）



18.4.4 时钟延长

7 位和 10 位从动模式都在发送序列上实现了时钟延长。

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末尾将 SCLx 引脚保持在低电平。

18.4.4.1 7 位从动接收模式 （SEN = 1）的时钟延长

在 7 位从动接收模式下，在 $\overline{\text{ACK}}$ 序列末尾的第 9 个时钟的下降沿，如果 BF 置 1，SSPxCON1 寄存器中 CKP 位会自动清零，强制 SCLx 输出保持低电平。清零 CKP 将迫使 SCLx 线保持低电平。在允许继续接收之前，必须在用户的 ISR 中将 CKP 位置 1。保持 SCLx 线为低电平可以让用户在主器件发起另一个接收序列之前，有时间处理 ISR 并读取 SSPxBUF 的内容。这将防止发生缓冲器溢出 （见图 18-15）。

注 1： 如果用户在第 9 个时钟的下降沿前读取了 SSPxBUF 的内容，使得 BF 被清零， CKP 位就不会被清零，也不会产生时钟延长。

2： 不管 BF 位的状态如何， CKP 位都可以用软件置 1。用户在下一个接收序列开始之前，清零 ISR 中的 BF 位时应该小心，以避免溢出。

18.4.4.2 10 位从动接收模式 （SEN = 1）的时钟延长

在 10 位从动接收模式下，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 个时钟之后置位，时钟延长就启动了。 UA 位在接收到 10 位地址的高字节后置 1，然后接收 10 位地址的第二个字节并将 R/W 位清“0”。在更新 SSPxADD 的过程中释放时钟线。如 7 位模式中描述的那样，在每个数据接收序列中会发生时钟延长。

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，而且用户没有提前读取 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数据序列中出现，不会出现在地址序列中。

18.4.4.3 7 位从动发送模式的时钟延长

如果 BF 位被清零， 7 位从动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。不管 SEN 位的状态如何，这种情况都会发生。

用户的 ISR 必须先将 CKP 位置 1 才可以继续发送。通过保持 SCLx 线为低电平，用户在主器件发起另一个发送序列之前，将有时间处理 ISR 并装入 SSPxBUF 的内容 （见图 18-10）。

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，从而使 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。

18.4.4.4 10 位从动发送模式的时钟延长

在 10 位从动发送模式下，在前两个地址序列中由 UA 位的状态来控制时钟延长，正如在 10 位从动接收模式中一样。头两个地址后跟着第三个地址序列，该地址序列包含 10 位地址的高位和被置为 1 的 R/W 位。在执行完第三个地址序列后，UA 位不置 1，此时器件配置为发送模式， BF 标志位控制时钟延长，就像在 7 位从动发送模式中一样 （见图 18-13）。

18.4.4.5 时钟同步和 CKP 位

当 CKP 位清零时，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，实际上，需要外部 I^2C 主器件将 SCLx 线拉低。SCLx 输出将保持低电平，直到 CKP 位置 1，且 I^2C 总线上的其他器件将 SCLx 电平拉高为止。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 18-14）。

图 18-14：　　时钟同步时序

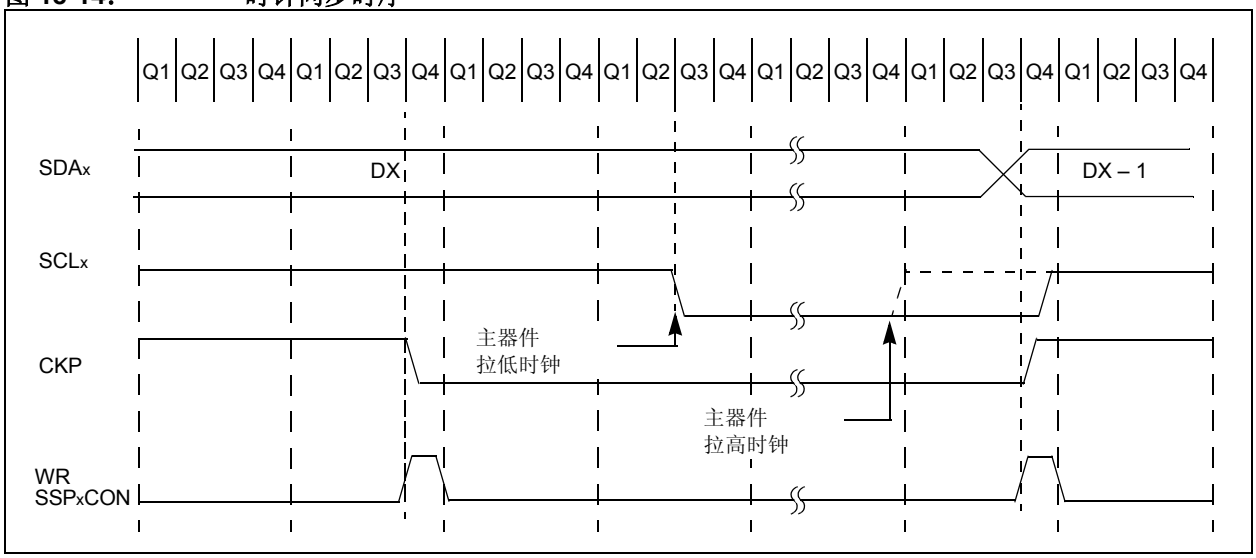


图 18-15：　　I²C™ 从动模式接收时序 （SEN = 1， 7 位地址）

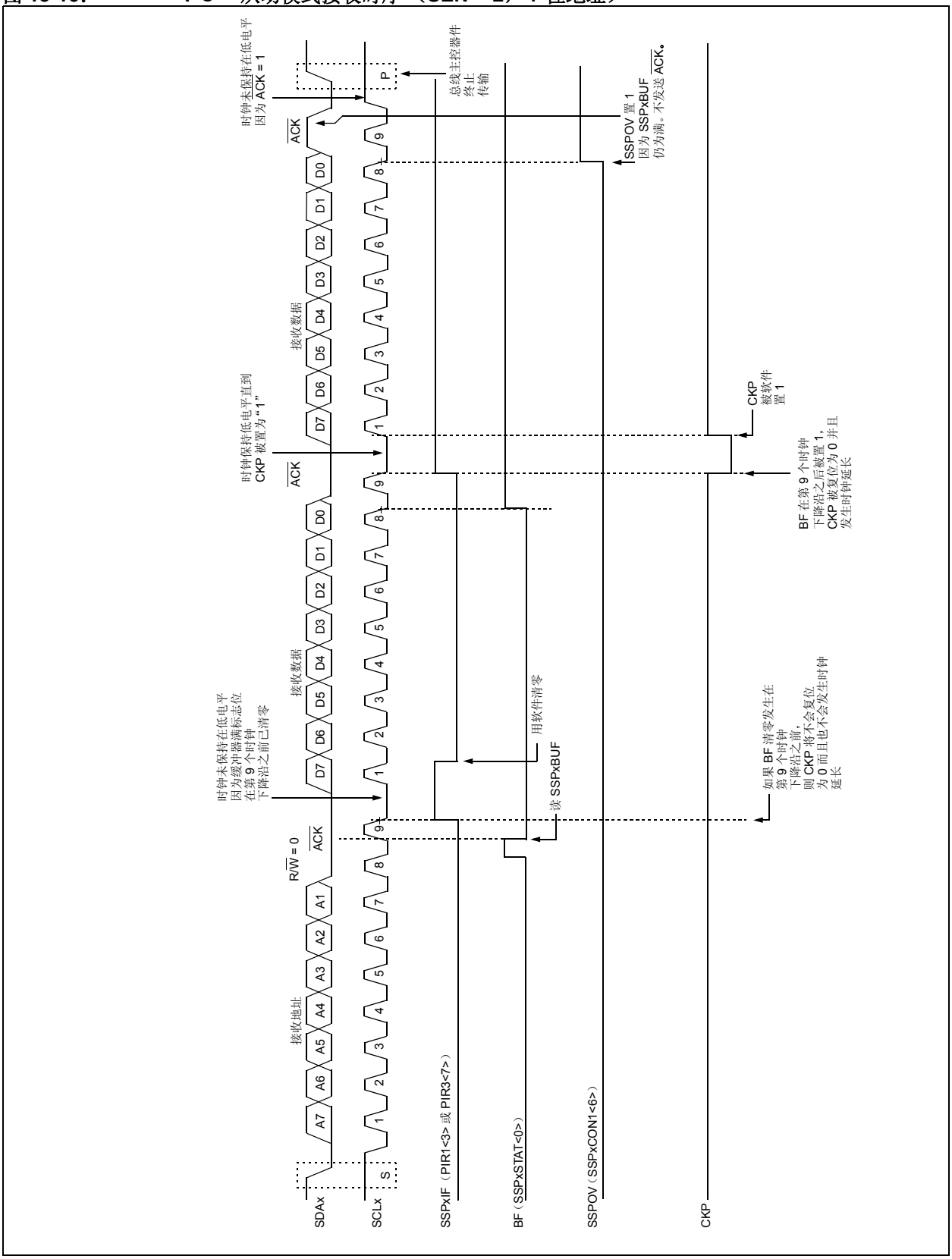
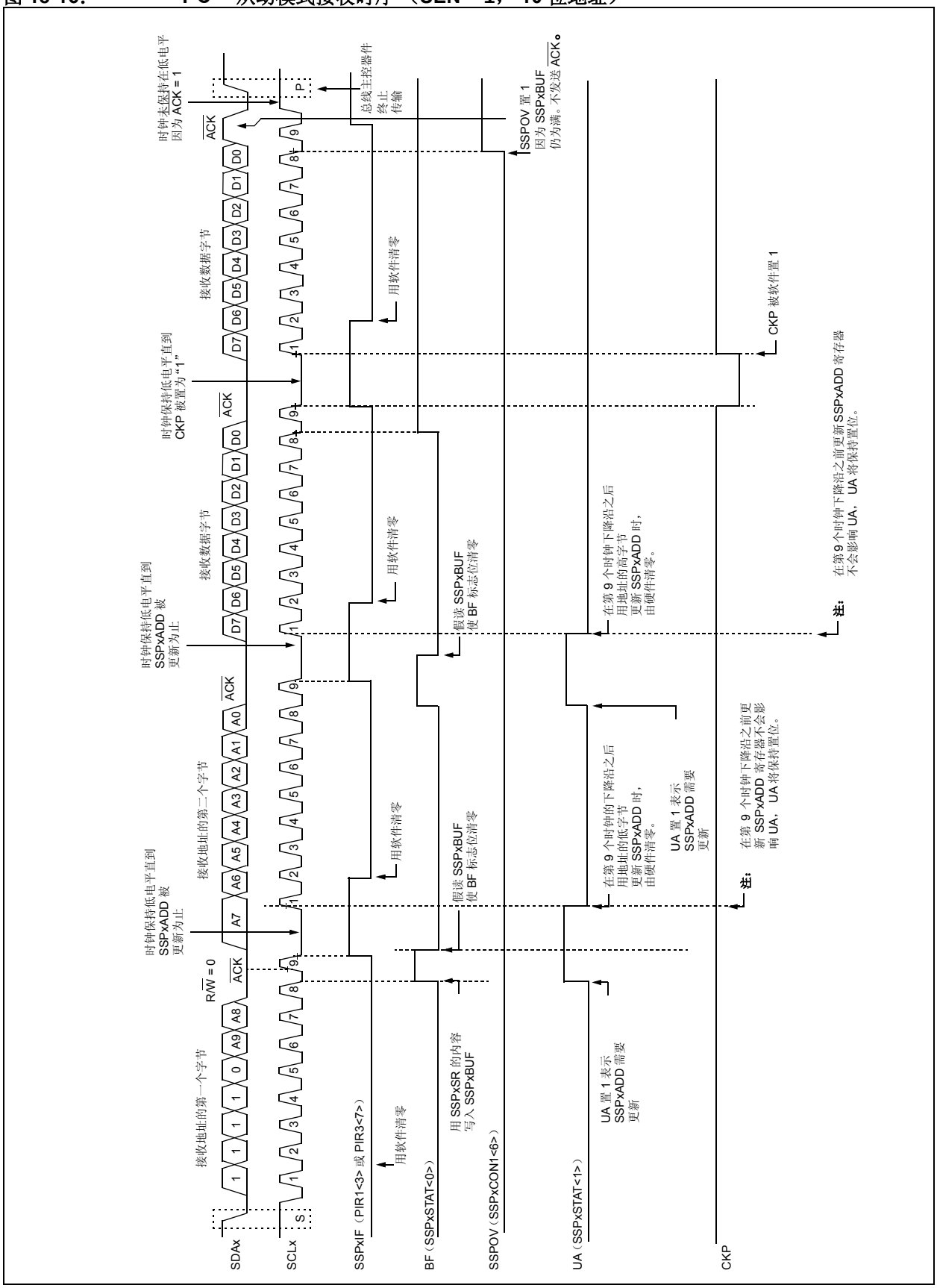


图 18-16：　　I²C™ 从动模式接收时序 （SEN = 1， 10 位地址）



18.4.5 广播呼叫地址支持

在 I²C 总线的寻址过程中，通常由启动条件后的第一个字节决定主器件将寻址哪个从器件。但广播呼叫地址例外，它能寻址所有器件。当使用这个地址时，理论上所有的器件都应该发送一个应答响应。

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。

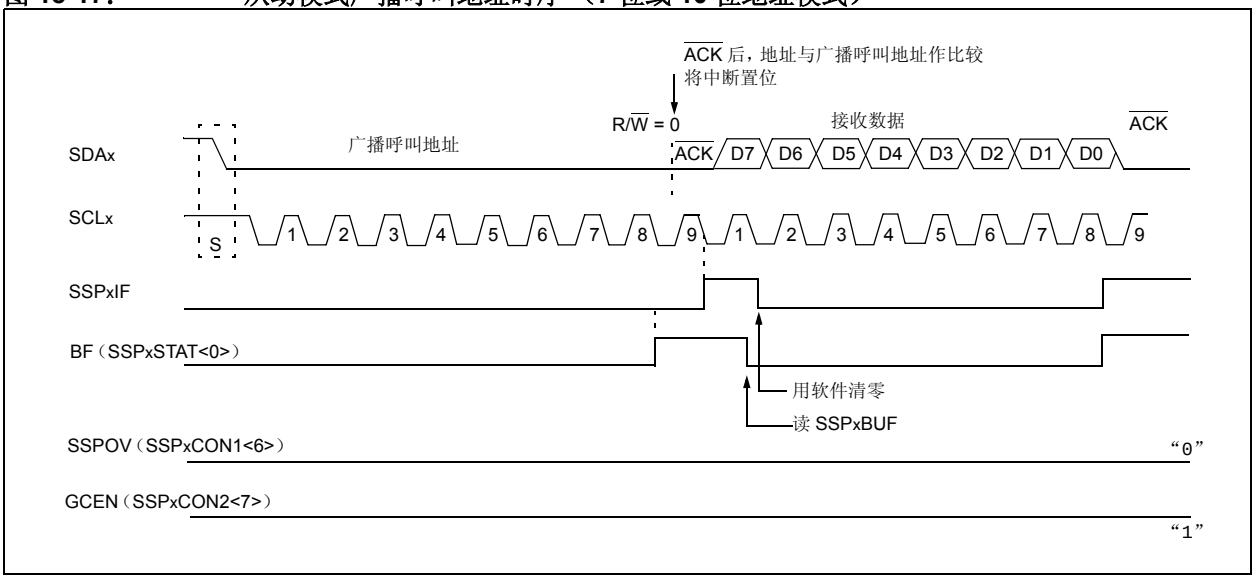
广播呼叫使能位（GCEN）（SSPxCON2<7>）置位时，即可识别广播呼叫地址。检测到启动位后， 8 位数据会移入 SSPxSR，同时将该地址与 SSPxADD 进行比较。它还会与广播呼叫地址进行比较并用硬件设定。

如果与广播呼叫地址匹配， SSPxSR 的值将传输到 SSPxBUF， BF 标志位（第 8 位）置 1，并且 SSPxIF 中断标志位在第 9 位 （$\overline{ACK}$ 位）的下降沿置 1。

当响应中断时，可以通过读取 SSPxBUF 的内容来判断中断源。该值可以用于判断地址是特定于器件的还是一个广播呼叫地址。

在 10 位模式下，需要更新 SSPxADD 来进行地址的后半部分匹配，同时 UA 位置 1（SSPxSTAT<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 位地址模式，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 18-17）。

图 18-17： 从动模式广播呼叫地址时序 （7 位或 10 位地址模式）



18.4.6 主控模式

通过将 SSPxCON1 中的相应 SSPM 位置位和清零，同时将 SSPEN 位置 1，可以使能主控模式。在主控模式下，如果 TRIS 位被置 1，则 SCLx 和 SDAx 线将由 MSSP 硬件控制。

主控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，否则总线处于空闲状态，且 P 位和 S 位都为零。

在固件控制的主控模式下，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。

一旦使能主控模式，用户即可选择以下 6 项操作：

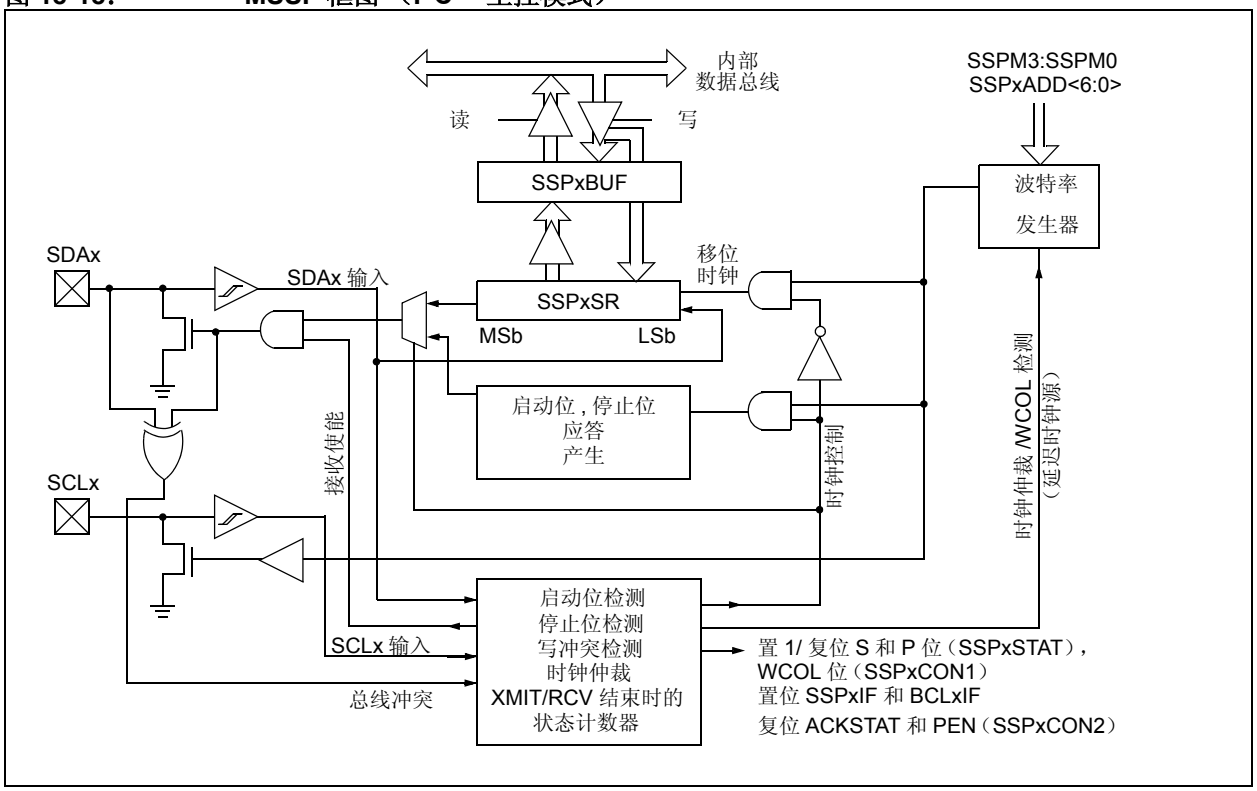
1. 在 SDAx 和 SCLx 上发出一个启动条件。
2. 在 SDAx 和 SCLx 上发出一个重复启动条件。
3. 写入 SSPxBUF 寄存器，开始数据 / 地址的发送。
4. 配置 I^2C 端口接收数据。
5. 在接收的数据字节后产生应答信号。
6. 在 SDAx 和 SCLx 上产生停止条件。

注： 当配置为 I^2C 主控模式时，MSSP 模块不允许事件排队。例如，在启动条件结束前，不允许用户发出另一个启动条件并立即写 SSPxBUF 寄存器以发起传输。这种情况下，将不会写入 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 置位（如果允许 MSSP 中断，则产生中断）：

- 启动条件
- 停止条件
- 数据传输字节已发送 / 已接收
- 应答发送
- 重复启动

图 18-18： MSSP 框图 （I^2C™ 主控模式）



18.4.6.1 I²C 主控模式工作方式

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。

在主控发送器模式下，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接收器件的从器件地址（7 位）和读 / 写（R/W）位。在这种情况下，R/W 位将是逻辑 0。一次发送 8 位串行数据。每发送一个字节，会收到一个应答位。输出启动和停止条件，表明串行传输的开始和结束。

在主控接收模式下，发送的第一个字节包括发送器件的从器件地址（7 位）和 R/W 位。在这种情况下，R/W 位将是逻辑 1。因此，发送的第一个字节是一个 7 位从器件地址，后面跟 1 表示接收。串行数据通过 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表明发送的开始和结束。

在 I²C 模式下，在 SPI 模式中使用的波特率发生器被用于将 SCLx 时钟频率设置为 100 kHz、400 kHz 或 1 MHz。详情，请参见**第 18.4.7 节 “波特率”**。

下面是一个典型的发送序列：

1. 用户通过置 1 启动使能位 SEN（SSPxCON2<0>）产生启动条件。
2. SSPxIF 位置 1。在进行任何其他操作前，MSSP 模块将等待所需的启动时间。
3. 用户将从器件地址装入 SSPxBUF 进行发送。
4. 地址从 SDAx 引脚移出，直到发送完所有 8 位为止。
5. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入SSPxCON2寄存器（SSPxCON2<6>）。
6. MSSP 模块在第 9 个时钟周期的末端将 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 位为止。
9. MSSP 模块移入来自从器件的 ACK 位，并将它的值写入SSPxCON2寄存器（SSPxCON2<6>）。
10. MSSP 模块在第 9 个时钟周期的末端将 SSPxIF 位置 1，产生一个中断。
11. 用户通过置 1 停止使能位 PEN（SSPxCON2<2>）产生停止条件。
12. 一旦停止条件完成，将产生一个中断。

18.4.7 波特率

在 I²C 主控模式下，波特率发生器（BRG）的重载值位于 SSPxADD 寄存器的低 7 位（图 18-19）。当发生对 SSPxBUF的写操作时，波特率发生器将自动开始计数。BRG 会递减计数至 0，然后停止直到下次重载为止。BRG 计数会在每个指令周期 （TCY）中的 Q2 和 Q4 时钟周期上进行两次减计数。在 I²C 主控模式下，会自动重载 BRG。

如果指定操作完成（即，在传输的最后一个数据位后面跟着 ACK），内部时钟将自动停止计数， SCLx 引脚将保持在其最后的状态。

表 18-3 显示了不同的指令周期下的时钟频率以及装入 SSPxADD 的 BRG 值。

18.4.7.1 波特率和模块的相互关系

因为 MSSP1 和 MSSP2 是独立的模块，所以它们可以以不同的波特率同时在 I²C 主控模式下工作。这是通过每个模块使用不同的 BRG 重载值实现的。

由于此模式的基本时钟源来自系统时钟，对系统时钟的任何更改将会同等程度地影响这两个模块。通过更改 BRG 重载值也许可以将一个或两个波特率改回到前一个值。

图 18-19： 波特率发生器框图

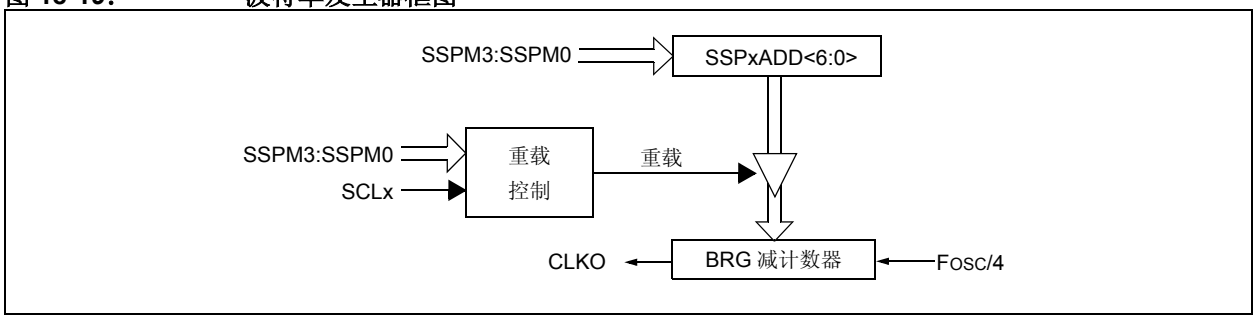


表 18-3： I²C™ 时钟速率和 BRG

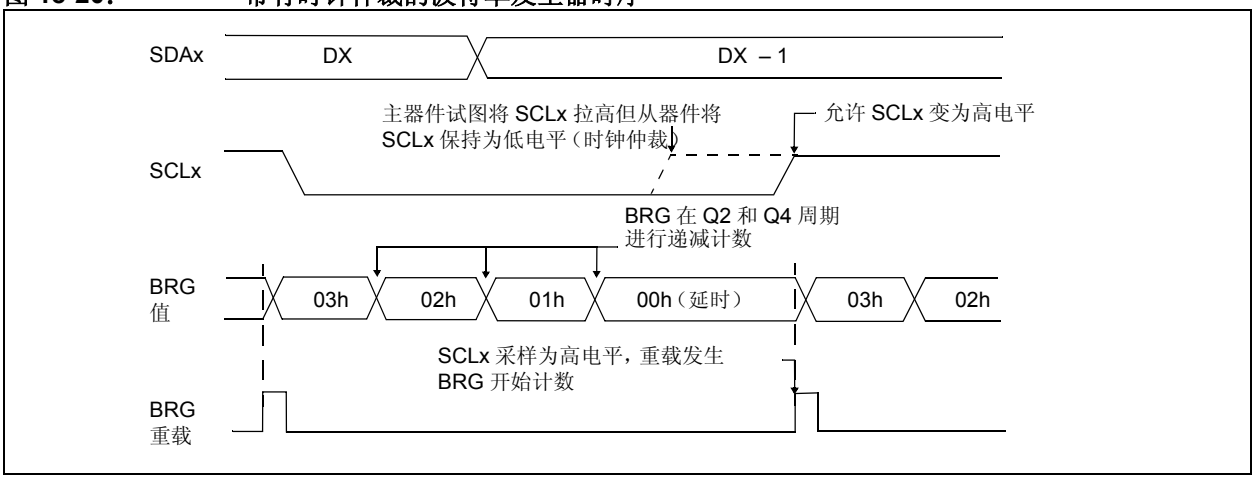
FCY	FCY * 2	BRG 值	FSCL （两次 BRG 计满返回）
10 MHz	20 MHz	18h	400 kHz(1)
10 MHz	20 MHz	1Fh	312.5 kHz
10 MHz	20 MHz	63h	100 kHz
4 MHz	8 MHz	09h	400 kHz(1)
4 MHz	8 MHz	0Ch	308 kHz
4 MHz	8 MHz	27h	100 kHz
1 MHz	2 MHz	02h	333 kHz(1)
1 MHz	2 MHz	09h	100 kHz
1 MHz	2 MHz	00h	1 MHz(1)

注 1： 虽然 I²C 接口各方面都不符合 400 kHz I²C 规范 （该规范适用于大于 100 kHz 的频率），但在需要较高频率的应用场合可以小心使用。

18.4.7.2 时钟仲裁

如果在任何接收、发送或重复启动 / 停止条件期间，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，波特率发生器将被重新装入 SSPxADD<6:0> 的内容并开始计数。这可以保证当外部器件将时钟拉低时，SCLx 始终保持至少一个 BRG 计满返回计数周期的高电平（图 18-20）。

图 18-20： 带有时钟仲裁的波特率发生器时序



18.4.8 I²C 主控模式启动条件序列

要发起启动条件，用户应将启动使能位 SEN（SSPxCON2<0>）置 1。当 SDAx 和 SCLx 引脚都采样为高电平时，波特率发生器重新装入SSPxADD<6:0>的内容并开始计数。当波特率发生器发生超时（TBRG）时，如果 SCLx 和 SDAx 都采样为高电平，则 SDAx 引脚被驱动为低电平。当 SCLx 为高电平时，将 SDAx 驱动为低电平就是启动条件，将使 S 位（SSPxSTAT<3>）置 1。随后波特率发生器重新装入 SSPxADD<6:0> 的内容并恢复计数。当波特率发生器超时（TBRG）时，SEN 位（SSPxCON2<0>）将自动被硬件清零。波特率发生器暂停工作， SDAx 线保持低电平，启动条件结束。

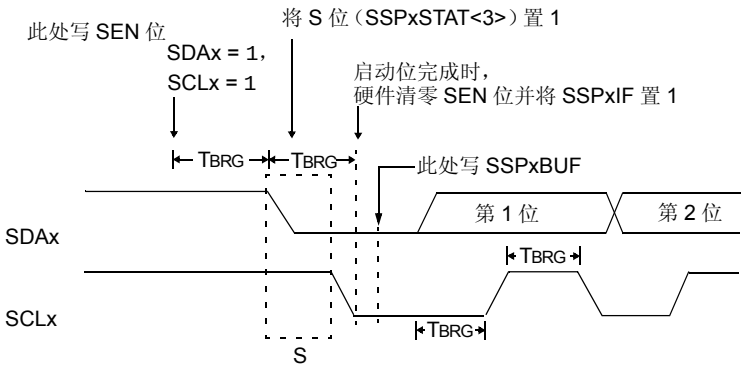
注： 如果在启动条件开始时，SDAx 和 SCLx 引脚已经采样为低电平，或者在启动条件期间，SCLx 在 SDAx 线被驱动为低电平之前已经采样为低电平，则会产生总线冲突。总线冲突中断标志位 BCLxIF 置 1，启动条件中止， I²C 模块复位到空闲状态。

18.4.8.1 WCOL 状态标志位

当启动序列进行时，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，在启动条件结束之前，不能对 SSPxCON2 的低 5 位进行写操作。

图 18-21： 第一个启动位时序



18.4.9 I²C 主控模式重复启动条件序列

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，波特率发生器装入 SSPxADD<5:0> 的内容，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（其引脚电平被拉高）。当波特率发生器超时时，如果 SDAx 采样为高电平， SCLx 引脚将被拉高。当 SCLx 引脚采样为高电平时，波特率发生器将被重新装入 SSPxADD<6:0> 的内容并开始计数。SDAx 和 SCLx 必须在一个计数周期 TBRG 内采样为高电平。在此操作后将 SDAx 引脚拉为低电平（SDAx = 0）并保持一个计数周期 TBRG，同时 SCLx 为高电平。然后 RSEN 位（SSPxCON2<1>）将自动清零，波特率发生器不会重载， SDAx 引脚保持低电平。一旦在 SDAx 和 SCLx 引脚上检测到启动条件， S 位（SSPxSTAT<3>）将被置 1。直到波特率发生器超时后， SSPxIF 位才会置 1。

注 1： 在其他事件进行时，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，SDAx 采样为低电平。
- 在 SDAx 被拉低之前， SCLx 变为低电平。这表示可能有另一个主器件正尝试发送数据 1。

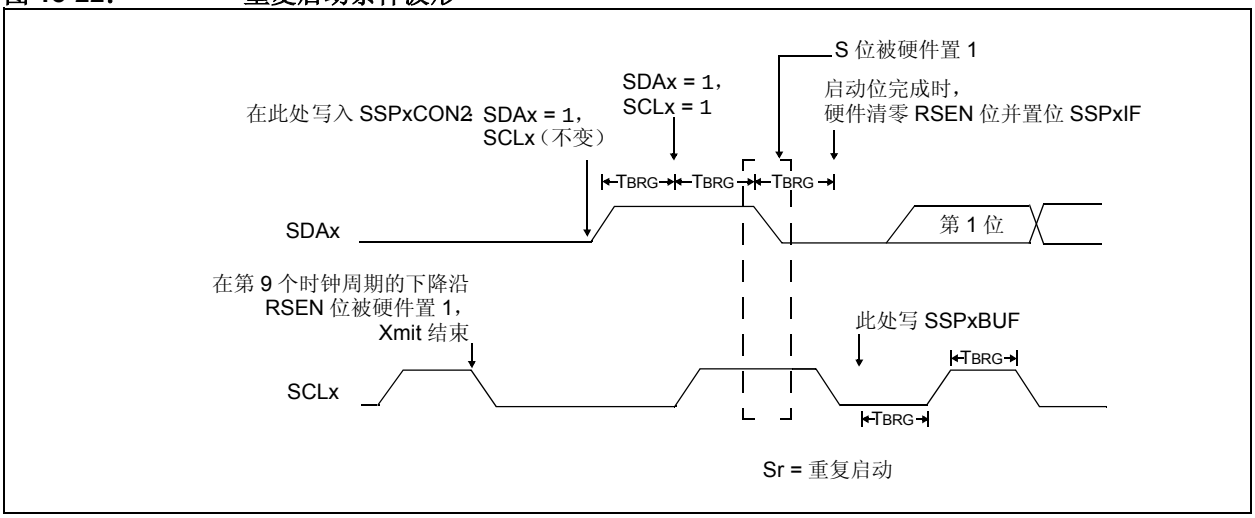
一旦 SSPxIF 位被置 1，用户便可以在 7 位地址模式下将 7 位地址写入 SSPxBUF，或者在 10 位地址模式下写入默认的第一个地址字节。当发送完第一个 8 位并接收到一个 ACK 后，用户可以发送该地址的另外 8 位（10 位地址模式下）或 8 位数据（7 位地址模式下）。

18.4.9.1 WCOL 状态标志位

当重复启动序列进行时，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，在重复启动条件结束之前，不能对 SSPxCON2 的低 5 位进行写操作。

图 18-22： 重复启动条件波形



18.4.10 I²C 主控模式下的发送

发送一个数据字节、一个 7 位地址或一个 10 位地址的另一半，都可以通过写一个值到 SSPxBUF 寄存器来实现。操作将使缓冲器满标志位 BF 置 1，并且波特率发生器开始计数，同时启动下一次发送。在 SCLx 的下降沿有效后 （见数据保持时间规范参数 106），地址 / 数据的每一位都移出至 SDAx 引脚。在一个波特率发生器计满返回计数周期（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 引脚释放为高电平时，它将在 TBRG 中保持高电平状态。在此期间以及下一个 SCLx 下降沿之后的一段时间内，SDAx 引脚上的数据必须保持稳定。在第 8 位被移出（第 8 个时钟周期的下降沿）之后，BF 标志位清零，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 位响应。ACK 的状态在第 9 个时钟周期的下降沿写入 ACKDT 位。主器件接收到应答之后，应答状态位 ACKSTAT 会被清零；如果未收到应答，则该位被置1。第9个时钟周期之后，SSPxIF 位会置 1，主控时钟 （波特率发生器）暂停，直到下一个数据字节装入 SSPxBUF 为止， SCLx 引脚保持低电平， SDAx 保持不变 （图 18-23）。

在写 SSPxBUF 之后，地址的每一位在 SCLx 的下降沿被移出，直至地址的所有 7 位和 R/$\overline{W}$ 位都被移出为止。在第 8 个时钟的下降沿，主器件将 SDAx 引脚拉为高电平，以允许从器件发出一个应答响应。在第 9 个时钟的下降沿，主器件通过采样 SDAx 引脚来判断地址是否被从器件识别。 $\overline{ACK}$ 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 置 1， BF 标志位清零，波特率发生器关闭直到下一次写 SSPxBUF，且 SCLx 引脚保持低电平，允许 SDAx 引脚悬空。

18.4.10.1 BF 状态标志位

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。

18.4.10.2 WCOL 状态标志位

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 置 1，且在写 SSPxBUF 之后的两个 TCY 内缓冲器内容不变（写操作无效）。如果在两个 TCY 内重新写 SSPxBUF，则 WCOL 位置 1，并更新 SSPxBUF。这可能导致传输被破坏。

在每次写 SSPxBUF 之后，用户应该确认 WCOL 位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。

18.4.10.3 ACKSTAT 状态标志位

在发送模式下，当从器件发送应答响应（$\overline{ACK}$ = 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。

18.4.11 I²C 主控模式接收

通过编程接收使能位 RCEN（SSPxCON2<3>）使能主控模式接收。

注：	RCEN 位置 1 前， MSSP 模块必须处于非激活状态，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），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，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。

18.4.11.1 BF 状态标志位

接收时，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。

18.4.11.2 SSPOV 状态标志位

接收时，当 SSPxSR 接收到 8 位数据时，SSPOV 位置 1， BF 标志位已经在上一次接收时置 1。

18.4.11.3 WCOL 状态标志位

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，缓冲器内容不变 （写操作无效）。

图 18-23：　　I^2C 主控模式发送时序 （7 位或 10 位地址）

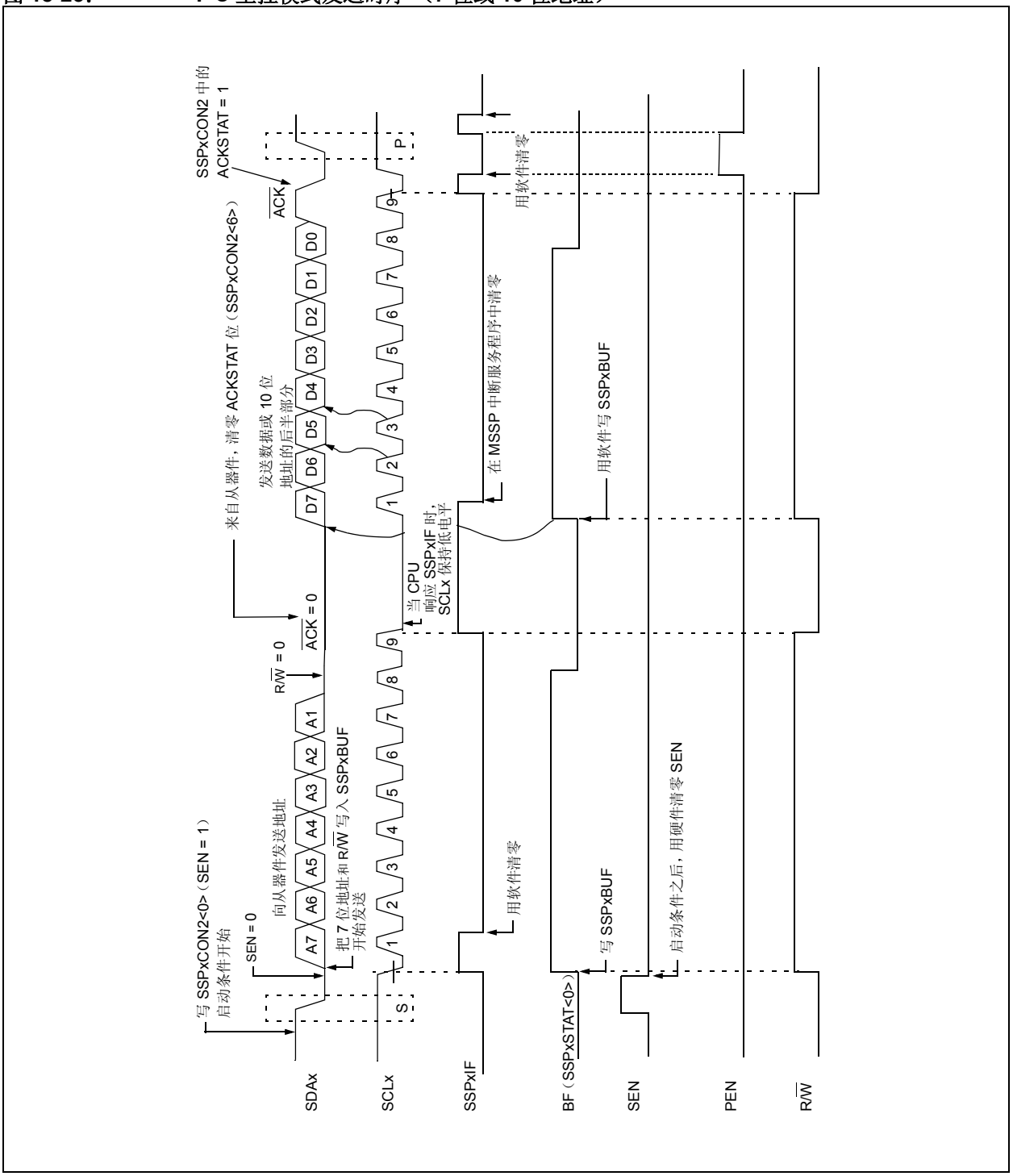
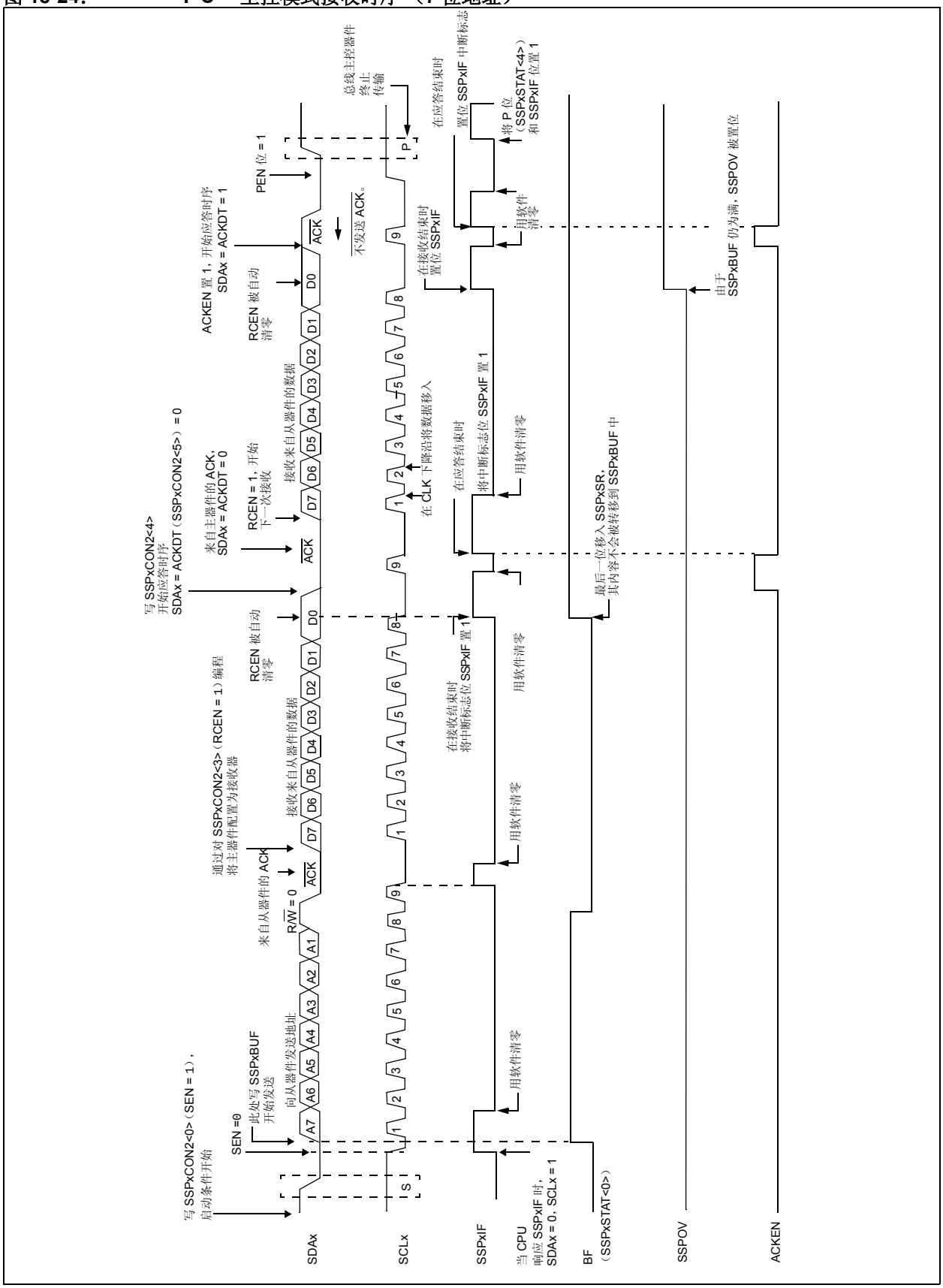


图 18-24：　　I²C™ 主控模式接收时序 （7 位地址）



18.4.12 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，应答数据位的内容出现在 SDAx 引脚上。如果用户希望产生一个应答，则应该将 ACKDT 位清零；否则，用户应该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，SCLx 引脚电平被拉高。当 SCLx 引脚采样为高电平时（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，MSSP 模块进入非工作模式（图 18-25）。

18.4.12.1 WCOL 状态标志位

如果用户在应答序列进行过程中试图写 SSPxBUF，则 WCOL 将置 1，缓冲器的内容不会改变（写操作无效）。

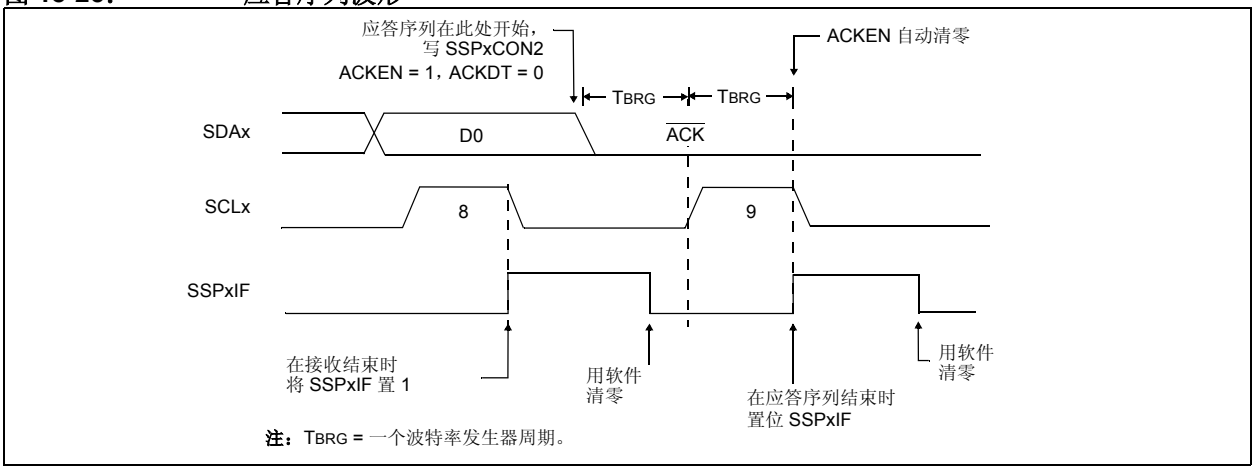
18.4.13 停止条件序列

通过将停止序列使能位 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 / 发送结束后， SCLx 线在第 9 个时钟的下降沿之后保持低电平。当 PEN 位置 1 时，主器件将 SDAx 线置为低电平。当 SDAx 线采样为低电平时，波特率发生器被重载并递减计数至 0。当波特率发生器超时时，SCLx 引脚被拉为高电平，在一个 TBRG （波特率发生器计满返回计数周期）之后， SDAx 引脚将被拉高。当 SDAx 引脚采样为高电平且 SCLx 也是高电平时， P 位（SSPxSTAT<4>）置 1。一个 TBRG 之后， PEN 位清零，同时 SSPxIF 位置 1 （图 18-26）。

18.4.13.1 WCOL 状态标志位

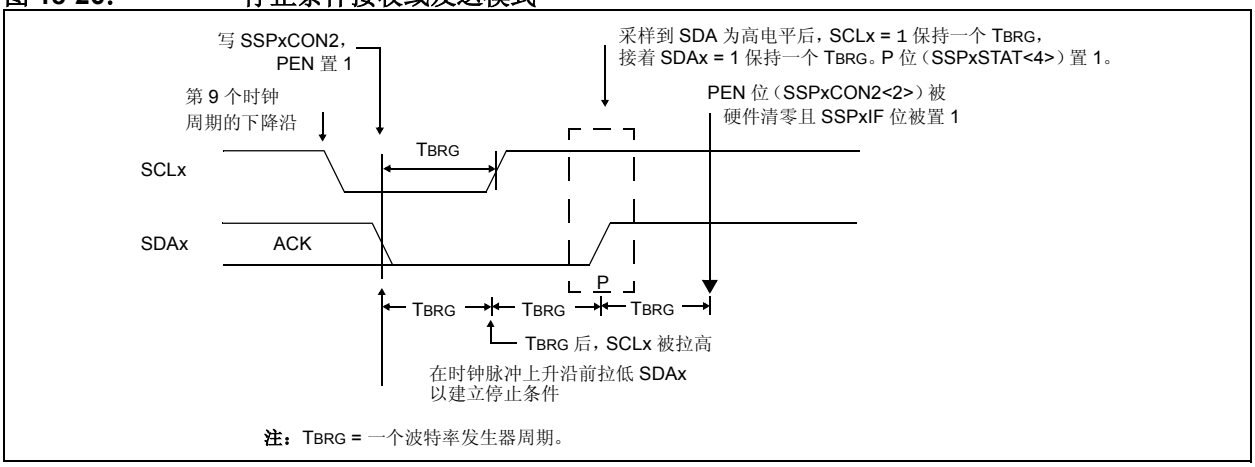
如果用户在停止序列过程中试图写 SSPxBUF，则 WCOL 位将置 1，缓冲器的内容不会改变 （写操作无效）。

图 18-25： 应答序列波形



注： TBRG = 一个波特率发生器周期。

图 18-26： 停止条件接收或发送模式



注： TBRG = 一个波特率发生器周期。

18.4.14 休眠工作方式

在休眠模式下，I^2C 模块能够接收地址或数据。并且地址匹配或字节传输完成后，如果允许 MSSP 中断，将唤醒处理器。

18.4.15 复位的影响

复位会禁止 MSSP 模块并中止当前的数据发送。

18.4.16 多主机模式

在多主机模式下，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。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>）置位时，可以取得 I^2C 总线的控制权；否则总线处于空闲状态，且 P 位和 S 位清零。当总线忙时，如果出现停止条件，则允许 MSSP 中断将产生中断。

在多主机模式工作中，必须监视 SDAx 线来进行仲裁，查看信号电平是否为期望的输出电平。此检查由硬件完成，其结果放在 BCLxIF 位。

仲裁可能失败的状态是：

- 地址传输
- 数据传输
- 启动条件
- 重复启动条件
- 应答信号

18.4.17 多主控通信、总线冲突与总线仲裁

多主机模式是通过总线仲裁来支持的。当主器件将地址 / 数据位输出到 SDAx 引脚时，如果一个主器件通过将 SDAx 引脚悬空为高电平以在 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，而实际在 SDAx 引脚上采样到的数据是 0，则发生了总线冲突。主器件将把总线冲突中断标志位 BCLxIF 置 1，并将 I^2C 端口复位到空闲状态（图 18-27）。

如果在发送过程中发生总线冲突，则发送停止，BF 标志位清零，SDAx 和 SCLx 线被拉高，并且可对 SSPxBUF 进行写操作。当执行完总线冲突中断服务程序后，如果 I^2C 总线空闲，用户可通过发出启动条件恢复通信。

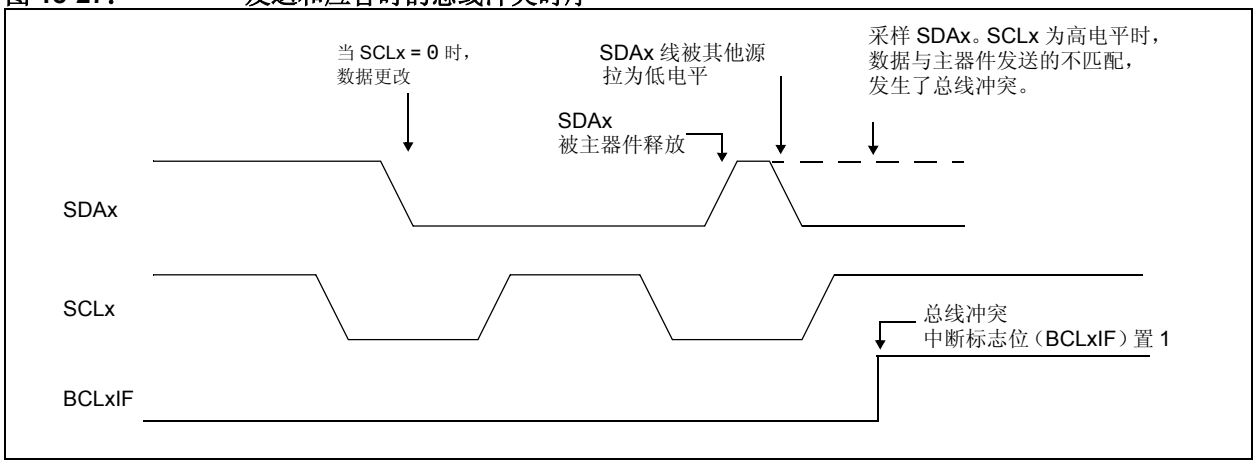
如果在启动、重复启动、停止或应答信号的执行过程中发生总线冲突，则这种状态被中止，SDAx 和 SCLx 线被拉高，SSPxCON2 寄存器中的对应控制位清零。当执行完总线冲突中断服务程序后，如果 I^2C 总线空闲，用户可通过发出启动条件恢复通信。

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 SSPxBUF 都会从第一个数据位开始发送数据。

在多主机模式下，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，可以获取 I^2C 总线的控制权；否则总线处于空闲状态且 S 和 P 位清零。

图 18-27： 发送和应答时的总线冲突时序



18.4.17.1 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 18-28）。

b）SDAx 被拉低之前，SCLx 采样为低电平（图 18-29）。

在启动条件期间，SDAx 和 SCLx 引脚都会被监视。

如果 SDAx 引脚已经是低电平，或 SCLx 引脚已经是低电平，则：

- 中止启动条件，
- BCLxIF 标志位置 1，
- MSSP 模块复位为非工作状态（图 18-28）。

启动条件从 SDAx 和 SCLx 引脚被拉高开始。当 SDAx 引脚采样为高电平时，波特率发生器装入 SSPxADD<6:0> 的值并递减计数到 0。如果在 SDAx 为高电平时，SCLx 引脚采样为低电平，则发生总线冲突，因为这表示另一台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 18-30）。但是，如果 SDAx 引脚采样为 1，SDAx 引脚将在 BRG 计数结束时被置为低电平。接着，波特率发生器被重新装入值并递减计数至 0。在此期间，如果 SCLx 引脚采样到 0，则没有发生总线冲突。在 BRG 计数结束时，SCLx 引脚被拉为低电平。

注： 在启动条件期间不会发生总线冲突是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。因此总是有一个主器件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，因为允许两个主器件对启动条件后的第一个地址进行仲裁。如果地址是相同的，将继续对数据部分、重复启动条件或停止条件进行仲裁。

图 18-28： 启动条件期间的总线冲突（仅 SDAx）

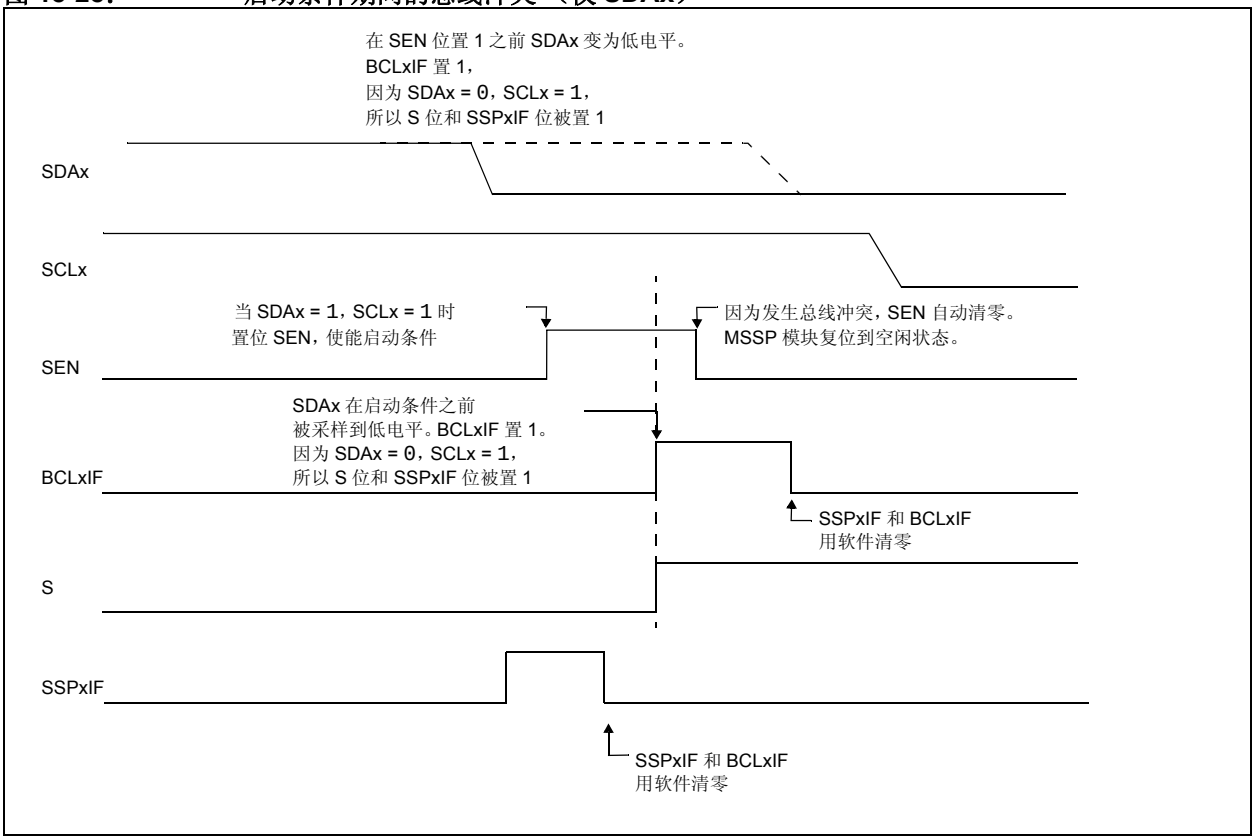


图 18-29：　　启动条件期间的总线冲突 （SCLx = 0）

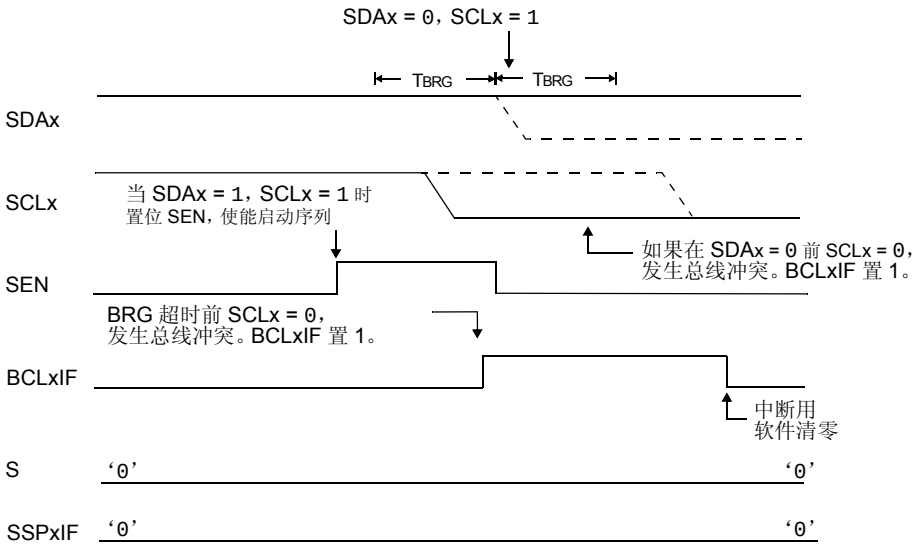
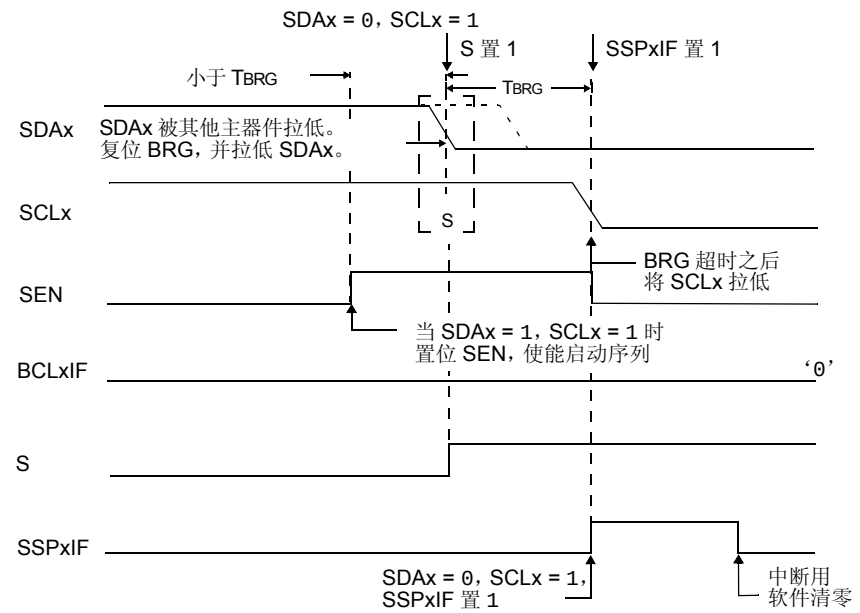


图 18-30：　　启动条件期间由 SDAx 仲裁引起的 BRG 复位



18.4.17.2 重复启动条件期间的总线冲突

在下列情况中，重复启动条件期间会发生总线冲突：

a）在 SCLx 由低电平变为高电平的过程中，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一个数据 “1”。

当用户拉高 SDAx 并允许该引脚悬空时，BRG 装入 SSPxADD<6:0> 中的值并递减计数至 0。接着 SCLx 引脚被置为高电平，当 SCLx 采样到高电平时，对 SDAx 引脚进行采样。

如果 SDAx 为低电平，则已发生了总线冲突（即，另一个主器件正试图发送一个数据 0，见图 18-31）。如果 SDAx 采样为高电平，则 BRG 被重新装入值并开始计数。如果 SDAx 在 BRG 超时之前从高电平变为低电平，则没有发生总线冲突，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 尚未变为低电平，表示发生了总线冲突。在此情况下，在重复启动条件期间另一个主器件正试图发送一个数据 1（见图 18-32）。

如果在 BRG 超时结束时 SCLx 和 SDAx 都仍然是高电平，则 SDAx 引脚被拉低，BRG 重新装入值并开始计数。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 18-31： 重复启动条件期间的总线冲突 （情形 1）

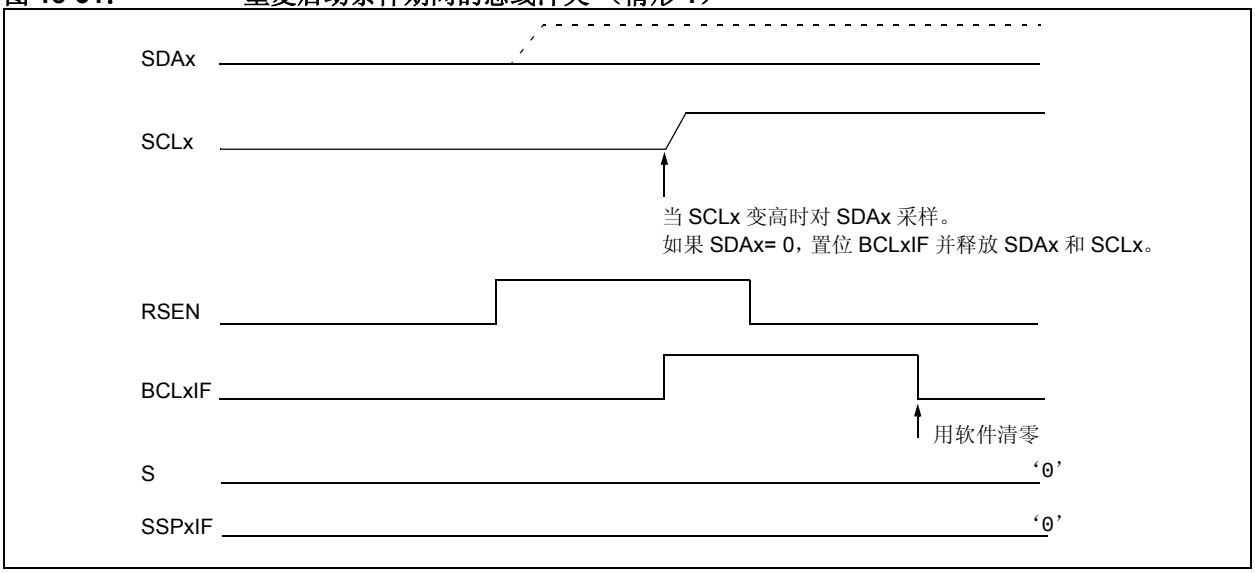
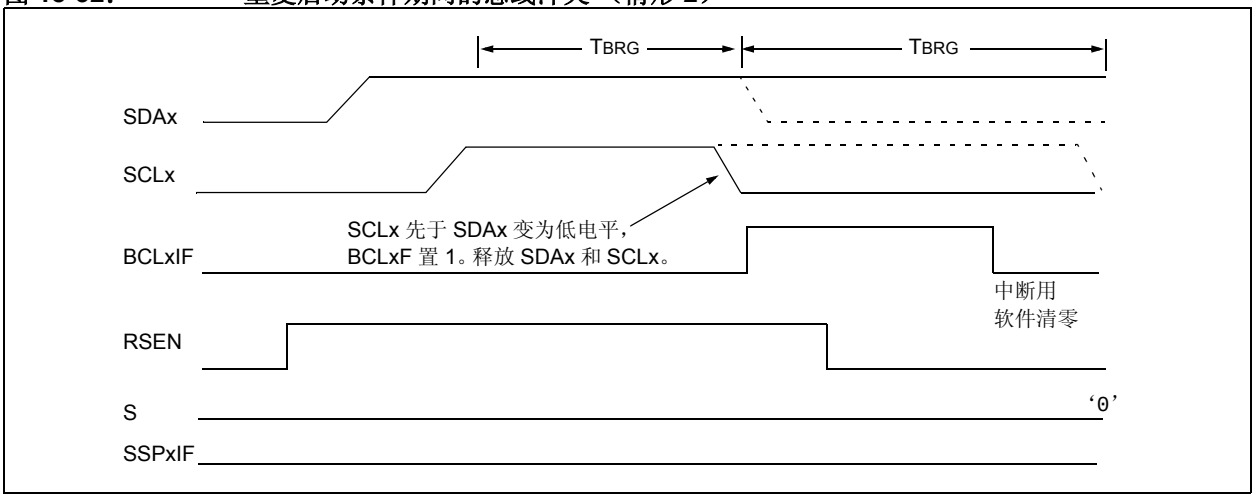


图 18-32： 重复启动条件期间的总线冲突 （情形 2）



18.4.17.3 停止条件期间的总线冲突

以下事件会导致停止条件期间的总线冲突：

a）SDAx 已被拉高并允许悬空为高电平之后，SDAx 在 BRG 超时后被采样到低电平。

b）SCLx 引脚被拉高之后，SCLx 在 SDAx 变成高电平之前被采样到低电平。

停止条件从 SDAx 被拉低开始。当 SDAx 采样为低电平时，SCLx 引脚就被允许为高阻悬空。当引脚被采样到高电平时（时钟仲裁），波特率发生器装入 SSPxADD<6:0> 的值并递减计数到 0。BRG 超时后，SDAx 被采样。如果 SDAx 采样到低电平，则已发生总线冲突。这是因为另一个主器件正试图发送一个数据 0（图 18-33）。如果 SCLx 引脚在允许 SDAx 悬空为高电平前被采样到低电平，也会发生总线冲突。这是另一个主器件正试图发送一个数据 0 的另一种情况（图 18-34）。

图 18-33： 停止条件期间的总线冲突（情形 1）

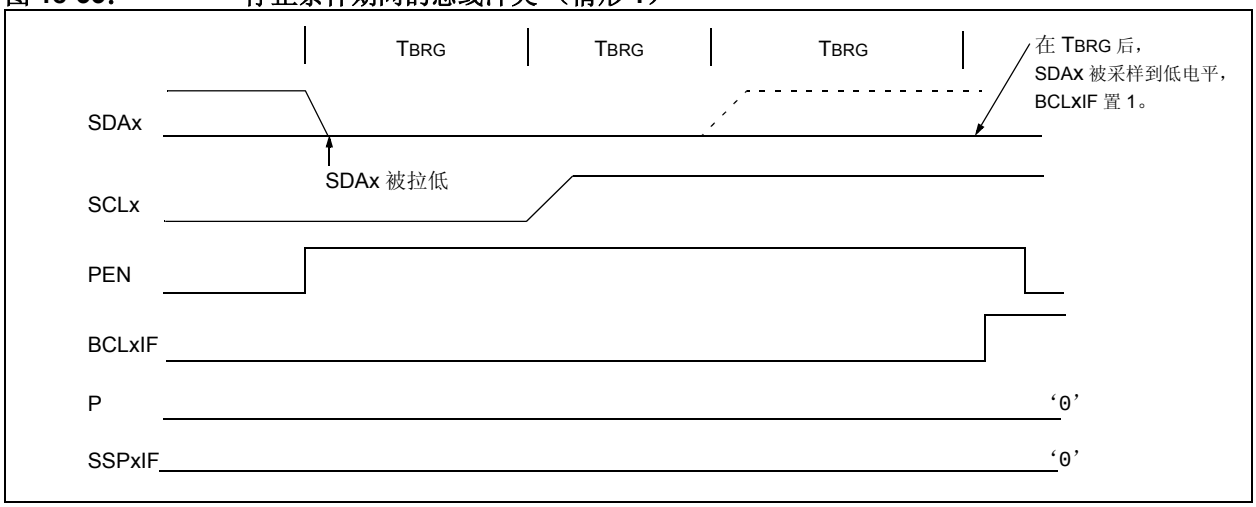


图 18-34： 停止条件期间的总线冲突（情形 2）

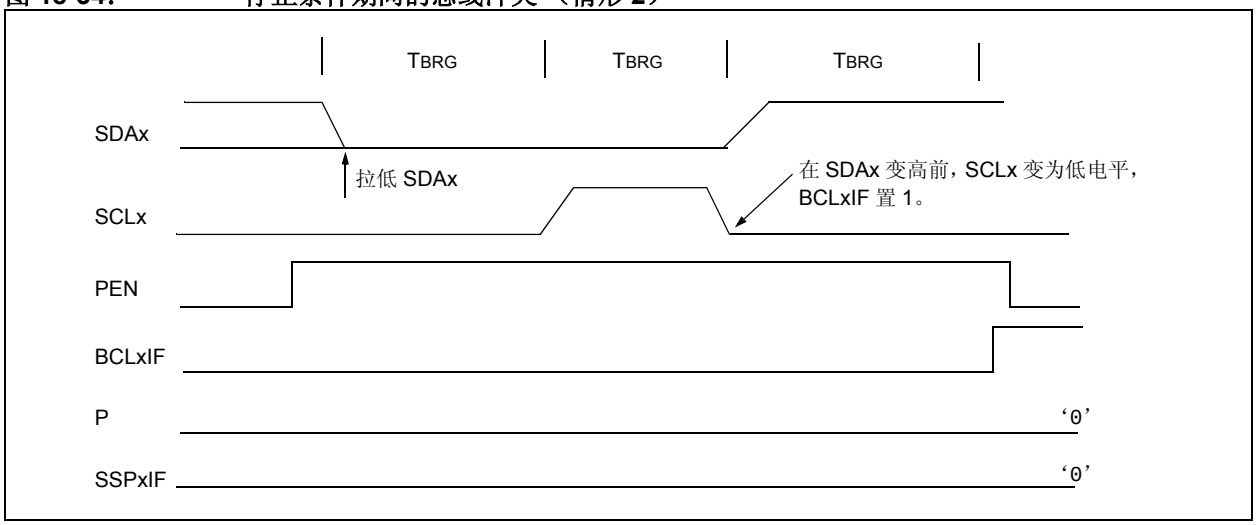


表 18-4： 与 I²C™ 工作模式相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR2	OSCFIF	CMIF	—	—	BCL1IF	—	TMR3IF	CCP2IF	51
PIE2	OSCFIE	CMIE	—	—	BCL1IE	—	TMR3IE	CCP2IE	51
IPR2	OSCFIP	CMIP	—	—	BCL1IP	—	TMR3IP	CCP2IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	52
TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	52
SSP1BUF	MSSP1 接收缓冲器 / 发送寄存器								50
SSP1ADD	MSSP1 地址寄存器 （I²C™ 从动模式）， MSSP1 波特率重载寄存器 （I²C 主控模式）								53
SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	50， 53
SSP1CON2	GCEN	ACKSTAT	ACKDT/ ADMSK5	ACKEN/ ADMSK4	RCEN/ ADMSK3	PEN/ ADMSK2	RSEN/ ADMSK1	SEN	50， 53
SSP1STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	50， 53
SSP2BUF	MSSP2 接收缓冲器 / 发送寄存器								50
SSP2ADD	MSSP2 地址寄存器 （I²C 从动模式）， MSSP2 波特率重载寄存器 （I²C 主控模式）								53

图注： — = 未用，读为 0。在 I²C 模式下， MSSP 模块未使用阴影单元。

注：

19.0 增强型通用同步 / 异步收发器

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（一般也将 EUSART 称为串行通信接口或 SCI）。可以将 EUSART 配置为能与 CRT 终端和个人计算机等外设通信的全双工异步系统，也可以将它配置成能够与 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 EUSART 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“同步中断”符号和发送 12 位间隔字符时自动唤醒。这些功能使 EUSART 模块成为局域互连网络（Local Interconnect Network，LIN）总线系统非常理想的选择。

PIC18F87J10 系列所有的器件都配备有两个独立的 EUSART 模块，分别称为 EUSART1 和 EUSART2。可将这两个模块配置为以下几种工作模式：

- 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控模式
- 时钟极性可选的半双工同步从动模式

EUSART1 和 EUSART2 的引脚分别与 PORTC（RC6/TX1/CK1 和 RC7/RX1/DT1）和 PORTG（RG1/TX2/CK2 和 RG2/RX2/DT2）的功能复用。要把这些引脚配置为 EUSART 需进行如下设置：

- 对于 EUSART1：
  - SPEN （RCSTA1<7>）位必须置 1
  - TRISC<7> 位必须置 1
  - TRISC<6> 位必须清 0 以使该模块工作于异步和同步主控模式
  - TRISC<6> 位必须置 1 以使该模块工作于同步从动模式
- 对于 EUSART2：
  - SPEN （RCSTA2<7>）位必须置 1
  - TRISG<2> 位必须置 1
  - TRISG<1> 位必须清 0 以使该模块工作于异步和同步主控模式
  - TRISC<6> 位必须置 1 以使该模块工作于同步从动模式

注：	USART控制在需要时会自动将引脚从输入重新配置为输出。

增强型 USART 模块的操作是通过 3 个寄存器控制的：

- 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）

在下页的寄存器 19-1、寄存器 19-2和寄存器 19-3中分别对这些寄存器进行了详细说明。

注：	在本章中，凡是涉及与特定 EUSART 模块相关的寄存器和位的名称一般都采用以“x”代替特定模块编号的方式。因此，“RCSTAx”可能指 EUSART1 的接收状态寄存器，也可能指 EUASRT2 的接收状态寄存器。

寄存器 19-1：　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	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7　**CSRC**：时钟源选择位

异步模式：
忽略。

同步模式：
1 = 主控模式 （时钟来自内部 BRG）
0 = 从动模式 （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 = 使能发送
0 = 禁止发送

注：　同步模式下 SREN/CREN 的优先级高于 TXEN。

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 “同步中断”字符 （在完成时用硬件清零）
0 = “同步中断”字符发送完成

同步模式：
忽略。

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。

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

寄存器 19-2：　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

bit 7　**SPEN**：串口使能位
1 = 使能串口 （配置 RXx/DTx 和 TXx/CKx 引脚作为串口引脚）
0 = 禁止串口 （保持在复位状态）

bit 6　**RX9**：9 位接收使能位
1 = 选择 9 位接收
0 = 选择 8 位接收

bit 5　**SREN**：单字节接收使能位
异步模式：
忽略。
同步主控模式：
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。
同步从动模式：
忽略。

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 比 SREN 优先级高）
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、接收所有字节并且第 9 位可用作奇偶校验位
9 位异步模式 （RX9 = 0）：
忽略。

bit 2　**FERR**：帧错误位
1 = 帧错误 （可以通过读 RCREGx 寄存器刷新并接收下一个有效字节）
0 = 没有帧错误

bit 1　**OERR**：溢出错误位
1 = 溢出错误 （可以通过清除 CREN 位清零）
0 = 没有溢出错误

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。

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

寄存器 19-3：　BAUDCONx：波特率控制寄存器

R/W-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

bit 7　**ABDOVF**：自动波特率采样进位状态位
1 = 在自动波特率检测模式下发生了 BRG 进位（必须用软件清零）
0 = 没有发生 BRG 进位

bit 6　**RCIDL**：接收操作空闲状态位
1 = 接收操作空闲
0 = 接收操作有效

bit 5　**未用**：读为 0

bit 4　**SCKP**：同步时钟极性选择位
异步模式：
在此模式下未使用。
同步模式：
1 = 空闲状态时钟 （CKx）为高电平
0 = 空闲状态时钟 （CKx）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　**未用**：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 = 不监视 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。

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

19.1 波特率发生器 （BRG）

BRG 是一个专用的 8 位或 16 位发生器，支持 EUSART 的异步和同步模式。默认情况下，BRG 工作在 8 位模式下，将 BRG16 位 （BAUDCONx<3>）置 1 可选择 16 位模式。

SPBRGHx:SPBRGx 这对寄存器控制独立运行定时器的周期。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 19-1 所示为不同 EUSART 模式的波特率计算公式，但仅适用于主控模式 （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 19-1 中的公式计算 SPBRGHx:SPBRGx 寄存器的近似整数值，从而确定波特率误差。例 19-1给出了计算示例。表 19-2 中显示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，或者有助于在高振荡频率条件下实现低波特率。

向 SPBRGHx:SPBRGx 寄存器写入新值会使 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。

19.1.1 在功耗管理模式下的操作

器件时钟用于产生所需的波特率。当进入一种功耗管理模式时，新时钟源可能会工作在与先前不同的频率下。这可能需要调整 SPBRGx 寄存器对中的值。

19.1.2 采样

择多检测电路对 RXx 引脚 （可以是 RC7/RX1/DT1 引脚或 RG2/RX2/DT2 引脚）采样三次，以判定 RXx 引脚上出现的是高电平还是低电平。

表 19-1： 波特率计算公式

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 19-1：　　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 9600bps 的器件：

目标波特率　　=　FOSC/(64 ([SPBRGHx:SPBRGx] + 1))

求解 SPBRGHx:SPBRGx 中的值：

X　=　((FOSC/ 目标波特率)/64) – 1

=　((16000000/9600)/64) – 1

=　[25.042] = 25

计算得到的波特率　=　16000000/(64 (25 + 1))

=　9615

误差　=　(计算得到的波特率 – 目标波特率)/ 目标波特率

=　(9615 – 9600)/9600 = 0.16%

表 19-2：　　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	复位值 所在页
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
BAUDCONx	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	52
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								52
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								52

图注：　— = 未用位，读为 0。 BRG 未使用阴影单元。

表 19-3： 异步模式下的波特率

目标波特率（kbps）	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（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 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	-32.1	1	—	—	—

目标波特率（kbps）	SYNC = 0, BRGH = 0, BRG16 = 0								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 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	-45.7	0	—	—	—	—	—	—

目标波特率（kbps）	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（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 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（kbps）	SYNC = 0, BRGH = 1, BRG16 = 0								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 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 19-3： 异步模式下的波特率 （续）

目标波特率（kbps）	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（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 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（kbps）	SYNC = 0, BRGH = 0, BRG16 = 1								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 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（kbps）	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（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 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（kbps）	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（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 进制）	实际波特率（kbps）	% 误差	SPBRG 值（10 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	—	—	—	—	—	—

19.1.3 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 AEDEN 位已置 1，就会开始自动波特率检测（图 19-1）。波特率是采用自平均的方式计算的。

在自动波特率检测（Auto-Baud Rate Detect，ABD）模式下，BRG 的时钟是反向的。不是由 BRG 为进入的 RXx 信号提供时钟，而是由 RXx 信号为 BRG 提供时钟。在 ABD 模式下，内部波特率发生器被用作计数器来计算进入的串行字节流的位周期。

一旦 ABDEN 位置 1，状态机就会将 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 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，在接收低位和高位的时间内都要进行测量。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递增计数。在 RXx 引脚传输了 8 个位，或在检测到第 5 个上升沿后，会将在相应 BRG 周期内累计的值保存在 SPBRGHx:PBRGx 寄存器对中。当第 5 个时钟周期出现时（应与停止位对应），ABDEN 位会自动清零。

如果发生了 BRG 进位（从 FFFFh 到 0000h 的溢出），那么该事件将会在ABDOVF状态位（BAUDCONx<7>）上反映出来。当 BRG 溢出时，该位由硬件置 1，它可以由用户在软件中清零。在发生溢出事件后，ABD 模式继续有效并且 ABDEN 位保持置 1（图 19-2）。

当校准波特率周期时，以 1/8 预配置时钟速率为 BRG 寄存器提供时钟。请注意 BRG 时钟将由 BRG16 和 BRGH 位配置。无论 BRG16 位的设置如何，SPBRGx 和 SPBRGHx 都将被用作 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。参见表 19-4 可获取 BRG 计数器的时钟速率。

当发生 ABD 事件时，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。要清除中断标志位RCxIF，需要读取RCREGx 中的值。应丢弃 RCREGx 的内容。

注 1： 如果 WUE 位与 ABDEN 位一起置 1，自动波特率检测会在间隔字符*之后*的字节开始。

2： 由用户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，必须考虑系统总体时序和通信波特率。

表 19-4： BRG 计数器时钟速率

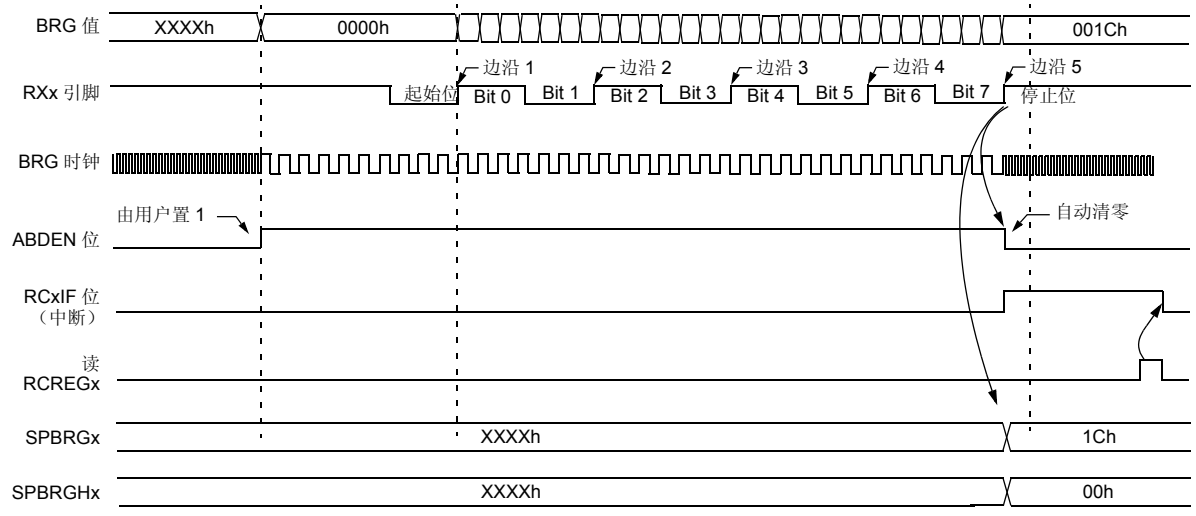
BRG16	BRGH	BRG 计数器时钟
0	0	FOSC/512
0	1	FOSC/128
1	0	FOSC/128
1	1	FOSC/32

注： 在 ABD 过程中，不管 BRG16 设置如何，SPBRGx 和 SPBRGHx 都用作 16 位计数器。

19.1.3.1 ABD 和 EUSART 发送

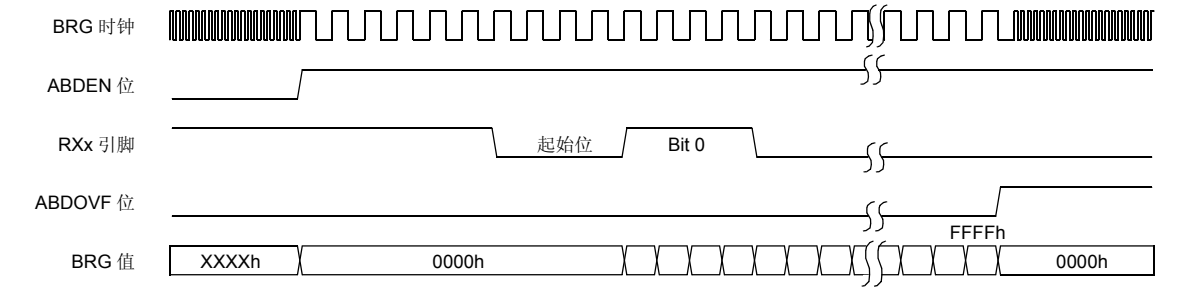
由于在 ABD 采集期间 BRG 时钟是反向的，因此在 ABD 期间不能使用 EUSART 发送器。这意味着只要 ABDEN 位置 1，就不能写入 TXREGx。用户还应确保在发送期间 ABDEN 不会被置 1。若不能满足这一条件，则可能会导致无法预期的 EUSART 操作。

图 19-1：　　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 才能进行 ABD 过程。

图 19-2：　　BRG 溢出时序



19.2 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 使用标准的非归零码（non-return-to-zero，NRZ）格式（1 个起始位、8 个或 9 个数据位和 1 个停止位）。最常用的数据格式为 8 位。片上专用 8 位 /16 位波特率发生器可用于通过振荡器产生标准波特率频率。

EUSART 首先发送和接收的是 LSb。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>）的值产生一个为移位速率 16 倍或 64 倍的波特率时钟。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。

当工作在异步模式下时，EUSART 模块由以下重要的部件组成：

- 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

19.2.1 EUSART 异步发送器

图 19-3 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。TXREGx 寄存器中的数据由软件写入。在前一次装入的停止位发送完成前，不会向 TSR 寄存器装入数据。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。可以通过将中断使能位 TXxIE （置 1 或清零来允许 / 禁止该中断。只要中断发生，不管 TXxIE 的状态如何，TXxIF 都置 1。TXxIF 不能用软件清零，也不会在装入 TXREGx 时立即清零，而是在装入指令后的第二个指令周期清零。因此在装入 TXREGx 后立即查询 TXxIF 会得到无效结果。

TXxIF 表示的是 TXREGx 寄存器的状态，而另一个位 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查询。

注 1： TSR 寄存器并未映射到数据存储器中，因此用户不能访问它。

2： 当使能位 TXEN 置 1 时，标志位 TX1IF 也会置 1。

要设置异步发送：

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，以获得所需的波特率。
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，请将中断允许位 TXxIE 置 1。
4. 若需要发送 9 位数据，请将发送位 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，应该将第 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 19-3： EUSART 发送框图

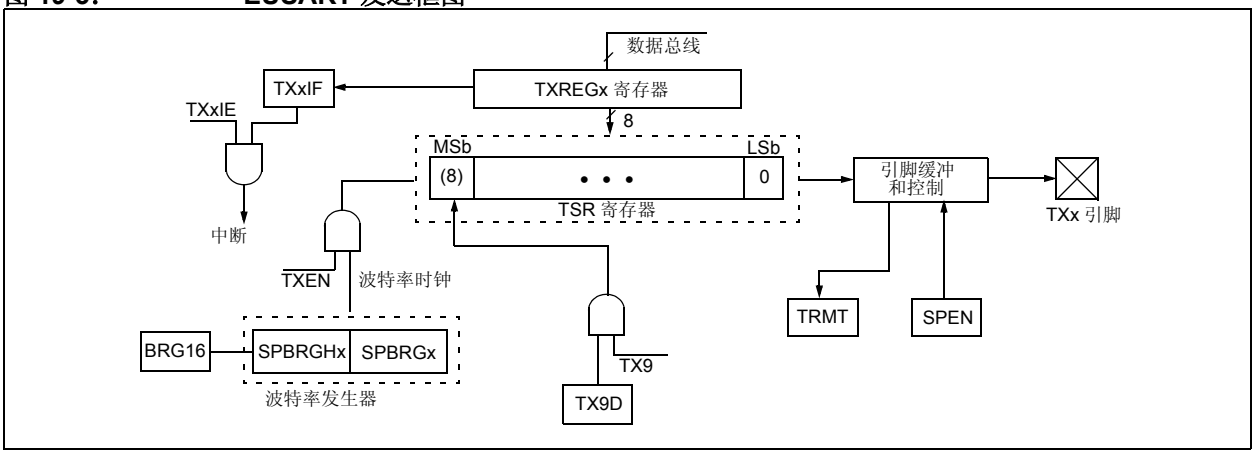


图 19-4：　　异步发送时序

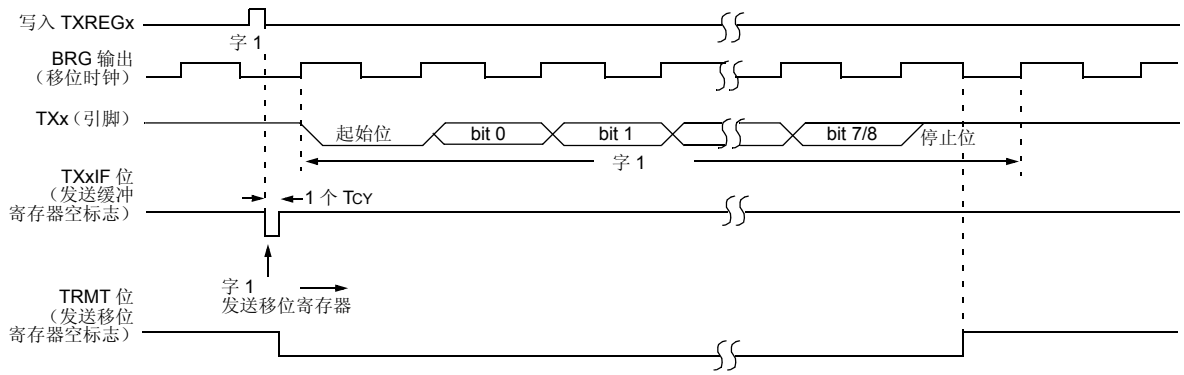
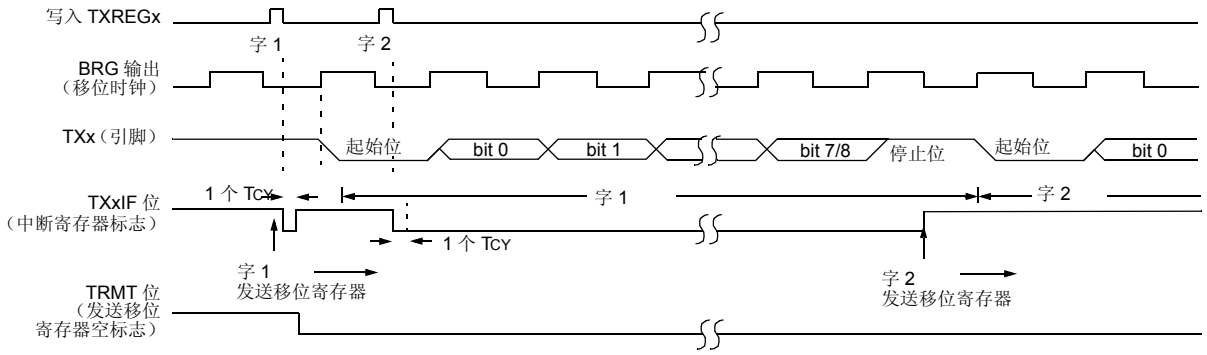


图 19-5：　　异步发送时序（背对背）



注：　此时序图表示的是两次连续的发送。

表 19-5：　　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	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
TXREGx	EUSARTx 发送寄存器								51
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
BAUDCONx	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	52
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								52
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								52

图注：　— = 未用单元，读为 0。异步发送不使用阴影单元。

19.2.2 EUSART 异步接收器

图 19-6 显示了接收器的框图。在 RXxT 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，而主接收串行移位寄存器则以比特率或 FOSC 工作。此模式通常用于 RS-232 系统。

要设置异步接收：

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，以获得所需的波特率。
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，请将中断允许位 RCxIE 置 1。
4. 若需要接收 9 位数据，请将 RX9 位置 1。
5. 将 CREN 位置 1 使能接收。
6. 当接收完成时标志位 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。
7. 读 RCSTAx 寄存器获取第 9 位数据 （如果已使能），并判断在接收过程中是否发生了错误。
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 清零以清除错误。
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。

19.2.3 设置带有地址检测功能的 9 位模式

此模式通常用在 RS-485 系统中。要设置使能地址检测的异步接收：

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，以获得所需的波特率。
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，请将 RCEN 位置 1 并使用 RCxIP 位选择所需的优先级。
4. 将 RX9 位置 1 使能 9 位接收。
5. 将 ADDEN 位置 1 使能地址检测。
6. 将 CREN 位置 1 使能接收。
7. 当接收完成时 RC1IF 位将置 1。此时如果 RC1IE 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 以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. 如果已经找到器件，将 ADDEN 位清零以允许所有接收到的数据进入接收缓冲器，并中断 CPU。

图 19-6： EUSART 接收框图

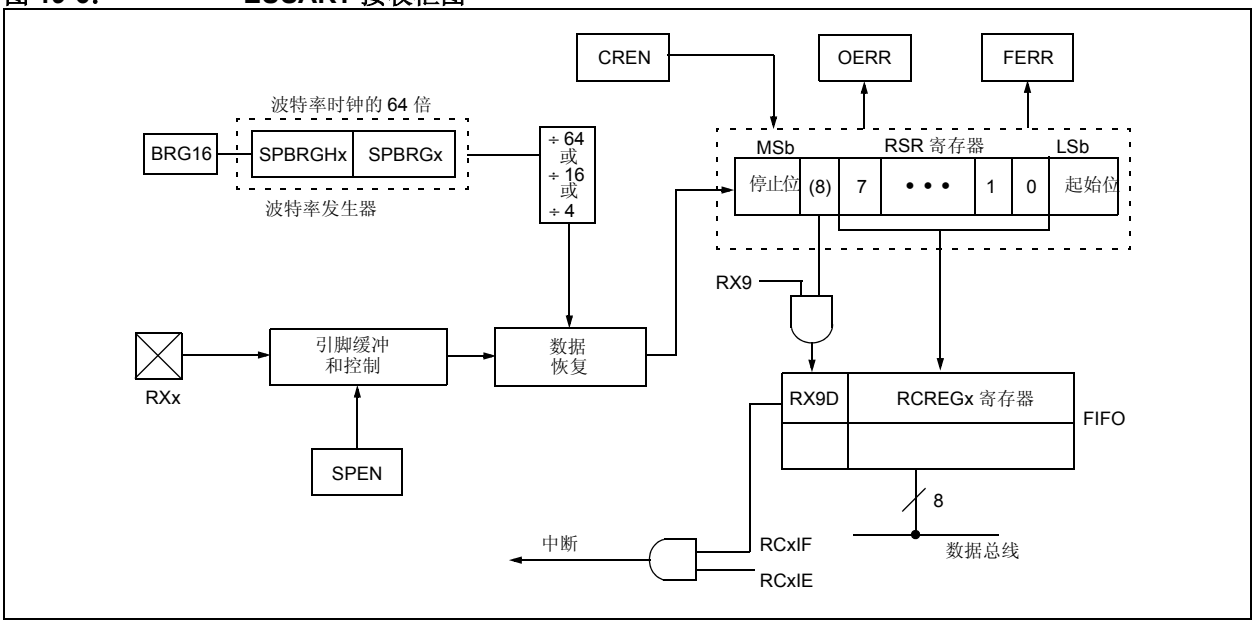
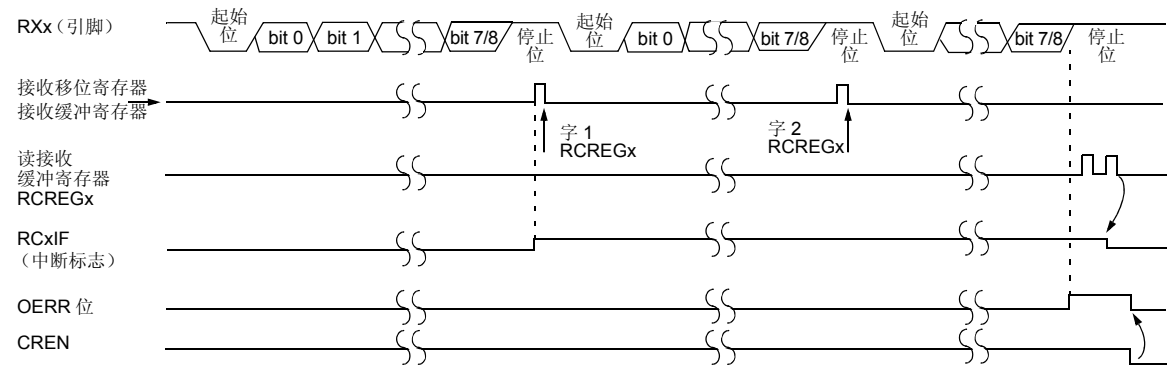


图 19-7：　　异步接收



注： 此时序图显示了在 RXx 输入引脚上顺序接收 3 个字的情况。在第 3 个字之后读 RCREGx（接收缓冲寄存器）会使 OERR（溢出）位置 1。

表 19-6：　　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	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
RCREGx	EUSARTx 接收寄存器								51
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
BAUDCONx	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	52
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								52
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								52

图注：　— = 未用，读为 0。异步接收不使用阴影单元。

19.2.4　同步间隔字符自动唤醒

在休眠模式下，所有的 EUSART 时钟都会暂停。因此，波特率发生器处于非活动状态，并且无法进行正确的字节接收。自动唤醒功能允许在 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 可以使能自动唤醒功能。置 1 后，将禁止 RXx/DTx 上的典型接收，并且 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 线上从高电平到低电平的转换（这与遇到“同步间隔”字符或 LIN 协议唤醒信号字符的起始位相同）。

发生唤醒事件后，模块会产生一个 RCxIF 中断。在正常工作模式下，中断会与 Q 时钟同步产生（图 19-8）；如果器件处于休眠模式，则两者异步产生（图 19-9）。通过读 RCREGx 寄存器可清除中断条件。

唤醒事件后，当 RXx 线上出现低电平向高电平的转换时，WUE 位自动清零。此时，EUSART 模块将从空闲状态返回正常工作模式。这将通知用户“同步中断”事件已经结束。

19.2.4.1 使用自动唤醒时需特别注意的事项

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，所以在停止位前的任何状态改变都可能会产生错误的字符结束信号并导致数据或帧错误。因此，为了正确工作，必须首先发送全 0 字符。对于标准的 RS-232 器件，该字符是 00h （8 位），而对于 LIN 总线则是 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迟较长的振荡器 （即 XT 或 HS 模式）的应用中更要注意这一点。“同步中断”（或唤醒信号）字符必须足够长，并且跟有足够长的时间间隔，以便使选定的振荡器有足够的时间起振并使 EUSART 正确初始化。

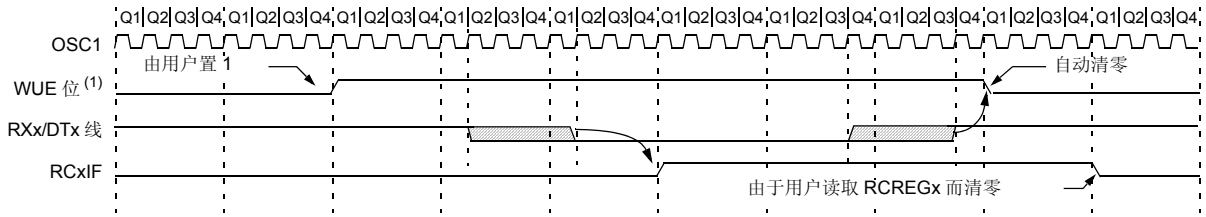
19.2.4.2 使用 WUE 位时需特别注意的事项

当用其来判断接收数据的有效性时，弄清WUE和RCxIF事件的时序可能会比较困难。如前所述，将 WUE 位置 1 会使 EUSART 进入空闲模式。通过将 RCxIF 位置 1，唤醒事件会产生一个接收中断。此后当 RXx/DTx 上出现上升沿时 WUE 位清零。然后可以通过读 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）和 RCxIF 标志位置 1 不应作为 RCREGx 中数据接收完整的标志。用户还应该考虑使用固件验证是否完整地接收了数据。

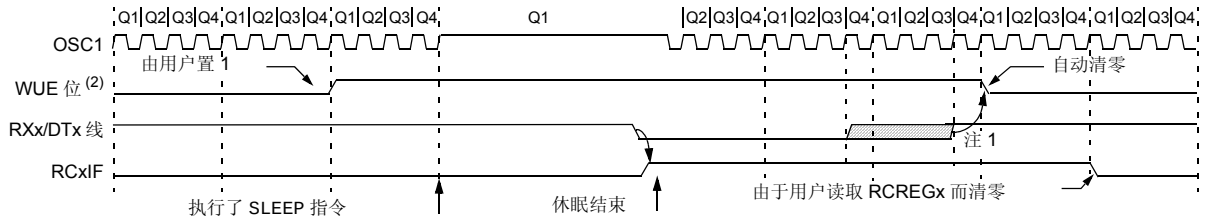
要确保没有丢失有效数据，应检查RCIDL 位以验证是否还在接收数据。如果接收已完成，则可将 WUE 位置 1，使器件立即进入休眠模式。

图 19-8： 正常工作状态下的自动唤醒位 （WUE）时序



注 1： WUE 位置 1 时 EUSART 保持在空闲状态。

图 19-9： 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 位可以自动清零。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。

19.2.5 间隔字符序列

EUSART 模块能够发送符合 LIN 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 数据。

在发送了相应的停止位后，硬件会自动将 SENDB 位清零。这允许用户在发送完间隔字符（在 LIN 规范中通常是同步字符）后预先将下一个要发送的字节装入发送 FIFO。

请注意写入 TXREGx 中作为间隔字符的数据值会被忽略。写入仅仅是为了启动正确的发送序列。

TRMT 位表明发送处于活动状态还是空闲状态，这与在正常发送过程中是相同的。关于间隔字符序列的时序，请参见图 19-10。

19.2.5.1 中断和同步发送序列

接下来要发送的序列是一个报文帧头，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 总线主控制器。

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. 将无效数据装入 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 TXREGx 为空时 （如 TXxIF 所示），下一个数据字节会写入 TXREGx。

19.2.6 接收间隔字符

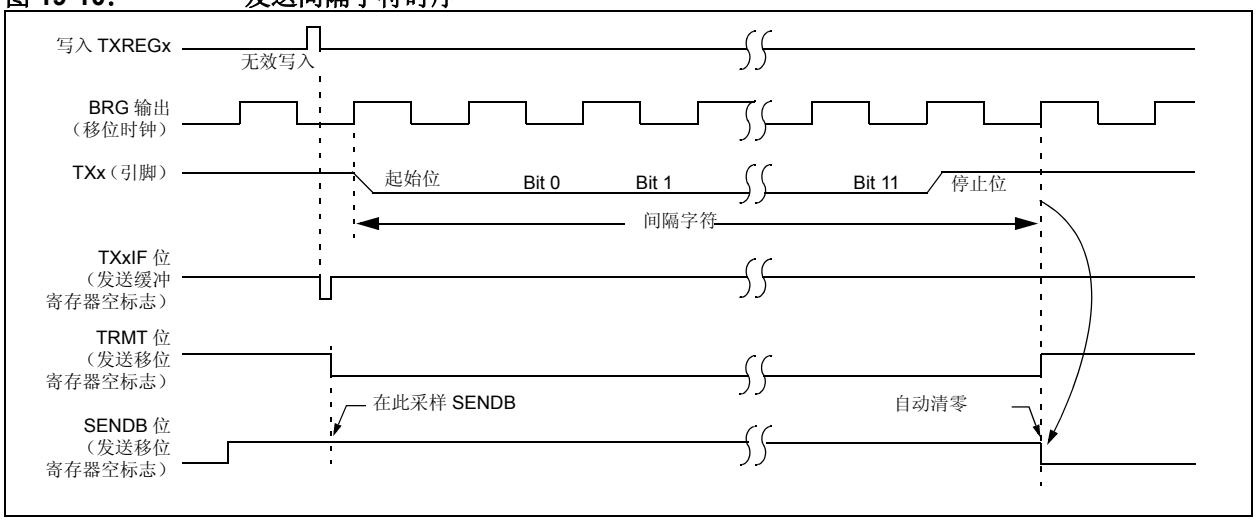
增强型 USART 模块接收间隔字符有两种方法。

第一种方法是强制将波特率配置为典型速度的 9/13。这可以使停止位在正确的采样点（对于间隔字符为起始位之后的 13 位，对于典型数据则是 8 个数据位）被接收。

第二种方法是使用**第 19.2.4 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 上电平的下两次跳变，产生一个 RCxIF 中断并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 19-10： 发送间隔字符时序



19.3 EUSART 同步主控模式

将 CSRC 位 （TXSTAx<7>）置 1 可以进入同步主控模式。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 以把 TXx 和 RXx 引脚分别配置为 CKx（时钟）和 DTx（数据）线。

主控模式意味着处理器在CKx时钟线上发送主控时钟信号。时钟极性是通过 SCKP 位 （BAUDCONx<4>）选择的。将 SCKP 置 1 将空闲状态时的 CKx 设为高电平，将该位清零则将空闲状态时的 CKx 设为低电平。此选项同时支持配有本模块的 Microwire 器件。

19.3.1 EUSART 同步主控发送

图 19-3 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。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前，不会向 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。

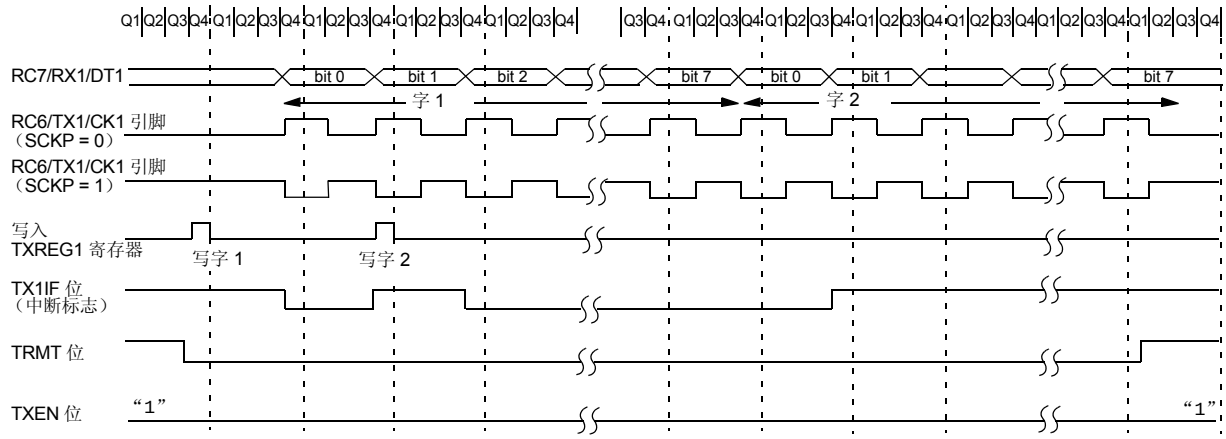
一旦 TXREGx 寄存器向 TSR 寄存器传输了数据 （在 1 个 TCY 内发生），TXREGx 寄存器就为空，同时标志位 TXxIF 置 1。可以通过将中断允许位 TXxIE 置 1 或清零来允许 / 禁止该中断。只要中断发生，不管 TXxIE 的状态如何， TXxIF 都会置 1。 TXxIF 不能用软件清零，只有把新数据写入 TXREGx 寄存器后， TXxIF 位才会复位。

TXxIF 表示的是 TXREGx 寄存器的状态，而另一个位 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查询。TSR 寄存器并未映射到数据存储器中，因此用户不能访问它。

要设置同步主控发送：

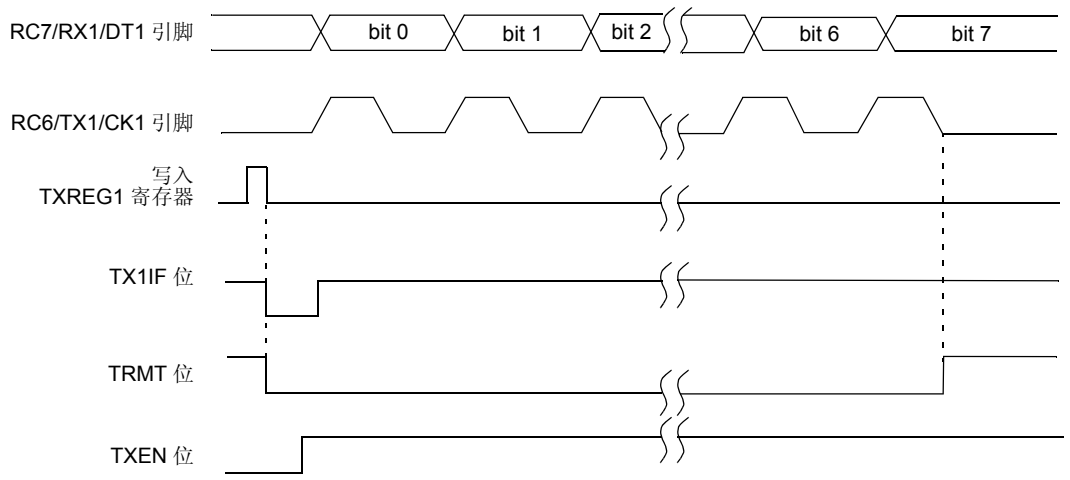
1. 初始化 SPBRGHx:SPBRGx 寄存器，选择合适的波特率。按需要将 BRG16 位置 1 或清零，以获得所需的波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1 使能同步主控串口。
3. 若需要中断，请将中断允许位 TX1IE 置 1
4. 若需要发送 9 位数据，请将发送位 TX9 置 1。
5. 将 TXEN 位置 1 使能发送。
6. 如果选择发送 9 位数据，应该将第 9 位数据装入 TX9D 位。
7. 将数据装入 TXREGx 寄存器开始发送。
8. 若想使用中断，请确保将 INTCON 寄存器中的 GIE 和 PEIE 位 （INTCON<7:6>）置 1。

图 19-11： 同步发送时序



注： 同步主控模式， SPBRGx=0，两个 8 位字连续发送。本例同样适用于 EUSART2 （RG1/TX2/CK2 和 RG2/RX2/DT2）

图 19-12：　　同步发送 （由 TXEN 位控制）时序



注： 本例同样适用于 EUSART2 （RG1/TX2/CK2 和 RG2/RX2/DT2）

表 19-7：　　与同步主控发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
TXREGx	EUSARTx 发送寄存器								51
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
BAUDCONx	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	52
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								52
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								52

图注：　　— = 未用位，读为 0。同步主控发送不使用阴影单元。

19.3.2 EUSART 同步主控接收

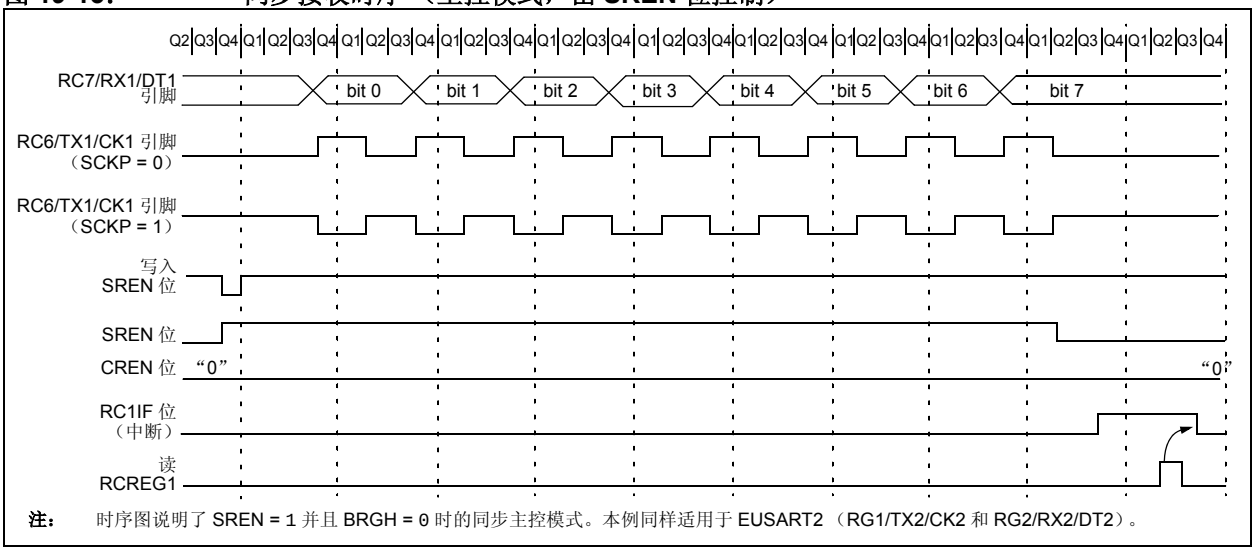
一旦选择了同步模式，只要将单字接收使能位 SREN（RCSTAx<5>）或连续接收使能位 CREN（RCSTAx<4>）置 1 即可使能接收。在时钟的下降沿采样 RXx 引脚上的数据。

如果将 SREN 置 1，则只接收单个字。如果将 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。

要设置同步主控接收：

1. 选择合适的波特率，初始化 SPBRGHx:SPBRGx 寄存器。按需要将 BRG16 位置 1 或清零，以获得所需的波特率。
2. 通过将 SYNC、SPEN 和 CSRC 位置 1 使能同步主控串口。
3. 确保将 CREN 和 SREN 位清零。
4. 若需要中断，请将中断允许位 RCxIE 置 1。
5. 若需要接收 9 位数据，请将 RX9 位置 1。
6. 若需要单字节接收，请将 SREN 位置 1；若需要连续接收，请将 CREN 位置 1。
7. 当接收完成时标志位 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 寄存器获取第 9 位数据（如果已使能），并判断在接收过程中是否发生了错误。
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，通过将使能位 CREN 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 19-13：　同步接收时序（主控模式，由 SREN 位控制）



注： 时序图说明了 SREN = 1 并且 BRGH = 0 时的同步主控模式。本例同样适用于 EUSART2（RG1/TX2/CK2 和 RG2/RX2/DT2）。

表 19-8：　与同步主控接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
RCREGx	EUSARTx 接收寄存器								51
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
BAUDCONx	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	52
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								52
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								52

图注： — = 未用位，读为 0。同步主控接收不使用阴影单元。

19.4 EUSART 同步从动模式

将 CSRC（TXSTAx<7>）清零可进入同步从动模式。此模式与同步主控模式的区别在于移位时钟由 CKx 引脚上的外部时钟提供（不同于主控模式中由内部时钟源提供）。这可以使器件能在任一低功耗模式下发送或接收数据。

19.4.1 EUSART 同步从动发送

除了休眠模式以外，同步主控模式和从动模式的工作方式是完全相同的。

如果向缓冲器 TXREGx 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 寄存器并发送。

b）第二个字仍保存在 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，中断将把芯片从休眠状态唤醒。如果允许了全局中断，程序则会跳转到中断向量处执行。

要设置同步从动发送：

1. 通过将 SYNC 和 SPEN 位置 1 并将 CSRC 位清零使能同步从动串口。
2. 将 CREN 和 SREN 位清零。
3. 若需要中断，请将中断允许位 TX1IE 置 1
4. 若需要发送 9 位数据，请将发送位 TX9 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，应该将第 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。

表 19-9： 与同步从动发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
TXREGx	EUSARTx 发送寄存器								51
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
BAUDCONx	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	52
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								52
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								52

图注： — = 未用位，读为 0。同步从动发送不使用阴影单元。

19.4.2 EUSART 同步从动接收

除了休眠模式、空闲模式以及在从动模式下忽略 SREN 位以外，同步主控和从动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 以使能接收，那么在该低功耗模式下仍可以接收一个字。接收到该字后，RSR 寄存器将把数据发送到 RCREGx 寄存器，如果中断允许位 RCxIE 已置 1，产生的中断将把芯片从低功耗模式唤醒。如果允许了全局中断，程序则会跳转到中断向量处执行。

要设置同步从动接收：

1. 通过将 SYNC 和 SPEN 位置 1 并将 CSRC 位清零使能同步从动串口。
2. 若需要中断，请将中断允许位 RCxIE 置 1
3. 若需要接收 9 位数据，请将 RX9 位置 1。
4. 将使能位 CREN 置 1 使能接收。
5. 当接收完成时， RCxIF 位将置 1。如果使能位 RCxIE 置 1，还将产生一个中断。
6. 读 RCSTAx 寄存器获取第 9 位数据 （如果已使能），并判断在接收过程中是否发生了错误。
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 清零以清除错误。
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 19-10： 与同步从动接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR3	SSP2IF	BCL2IF	RC2IF	TX2IF	TMR4IF	CCP5IF	CCP4IF	CCP3IF	51
PIE3	SSP2IE	BCL2IE	RC2IE	TX2IE	TMR4IE	CCP5IE	CCP4IE	CCP3IE	51
IPR3	SSP2IP	BCL2IP	RC2IP	TX2IP	TMR4IP	CCP5IP	CCP4IP	CCP3IP	51
RCSTAx	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	51
RCREGx	EUSARTx 接收寄存器								51
TXSTAx	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	51
BAUDCONx	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	52
SPBRGHx	EUSARTx 波特率发生器寄存器的高字节								52
SPBRGx	EUSARTx 波特率发生器寄存器的低字节								52

图注： — = 未用位，读为 0。同步从动接收不使用阴影单元。

注：

20.0 10 位模数转换器 （A/D）模块

64 引脚器件的模数（A/D）转换器模块具有 11 路输入，而 80 引脚器件的则具有 15 路输入。A/D 模块能将一个模拟输入信号转换成相应的 10 位数字信号。

此模块有 5 个寄存器：

- A/D 结果高位寄存器 （ADRESH）
- A/D 结果低位寄存器 （ADRESL）
- 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）

如寄存器 20-1 所示，A/D 模块的工作由 ADCON0 寄存器控制。如寄存器 20-2所示，端口引脚功能由ADCON1 寄存器配置。如寄存器 20-3 所示，由 ADCON2 寄存器配置 A/D 时钟源、可编程的采集时间和输出结果的对齐方式。

寄存器 20-1： ADCON0：A/D 控制寄存器 0

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCAL	—	CHS3	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

bit 7-6 **ADCAL**：A/D 校准位
1 = 在下一次 A/D 转换时进行校准
0 = 正常的 A/D 转换器操作 （没有进行校准）

bit 6 **未用**：读为 0

bit 5-2 **CHS3:CHS0**：模拟通道选择位
0000 = 通道 00 （AN0）
0001 = 通道 01 （AN1）
0010 = 通道 02 （AN2）
0011 = 通道 03 （AN3）
0100 = 通道 04 （AN4）
0101 = 未使用
0110 = 通道 06 （AN6）
0111 = 通道 07 （AN7）
1000 = 通道 08 （AN8）
1001 = 通道 09 （AN9）
1010 = 通道 10 （AN10）
1011 = 通道 11 （AN11）
1100 = 通道 12 （AN12）(1, 2)
1101 = 通道 13 （AN13）(1, 2)
1110 = 通道 14 （AN14）(1, 2)
1111 = 通道 15 （AN15）(1, 2)

bit 1 **GO/$\overline{\text{DONE}}$**：A/D 转换状态位
当 ADON = 1 时：
1 = A/D 转换正在进行
0 = A/D 空闲

bit 0 **ADON**：A/D 模块使能位
1 = 使能 A/D 转换器模块
0 = 禁止 A/D 转换器模块

注 1：在 64 引脚器件上不存在这些通道。
2：在不存在的通道上执行转换会返回随机值。

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

寄存器 20-2：　ADCON1：A/D 控制寄存器 1

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7-6　**未用：**读为 0

bit 5　**VCFG1：**参考电压配置位 （VREF- 电压源）
1 = VREF- （AN2）
0 = AVSS

bit 4　**VCFG0：**参考电压配置位 （VREF+ 电压源）
1 = VREF+ （AN3）
0 = AVDD

bit 3-0　**PCFG3:PCFG0：**A/D 端口配置控制位：

PCFG3: PCFG0	AN15	AN14	AN13	AN12	AN11	AN10	AN9	AN8	AN7	AN6	AN4	AN3	AN2	AN1	AN0
0000	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A
0111	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

A = 模拟输入　　D = 数字 I/O

注　1： AN12 至 AN15 仅在 80 引脚器件上可用。

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

寄存器 20-3：　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

bit 7　**ADFM**：A/D 结果格式选择位
1 = 右对齐
0 = 左对齐

bit 6　**未用**：读为 0

bit 5-3　**ACQT2:ACQT0**：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　**ADCS2:ADCS0**：A/D 转换时钟选择位
111 = FRC（由 A/D RC 振荡器产生时钟信号）(1)
110 = FOSC/64
101 = FOSC/16
100 = FOSC/4
011 = FRC（由 A/D RC 振荡器产生时钟信号）(1)
010 = FOSC/32
001 = FOSC/8
000 = FOSC/2

注　1：如果选择了 A/D FRC 时钟源，在 A/D 时钟启动之前会添加一个 TCY（指令周期）的延迟。这允许在开始转换之前执行 SLEEP 指令。

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

模拟参考电压可通过软件来选择，该参考电压可以是器件的正电源电压和负电源电压（AVDD 和 AVSS）或 RA3/AN3/VREF+/SEG17 和 RA2/AN2/VREF-/SEG16 引脚上的电压。

A/D 转换器具备一个独特的特性，它可在器件处于休眠模式时正常工作。要使 A/D 转换器在休眠模式下运行，A/D 转换时钟必须来自于 A/D 模块内部的 RC 振荡器。

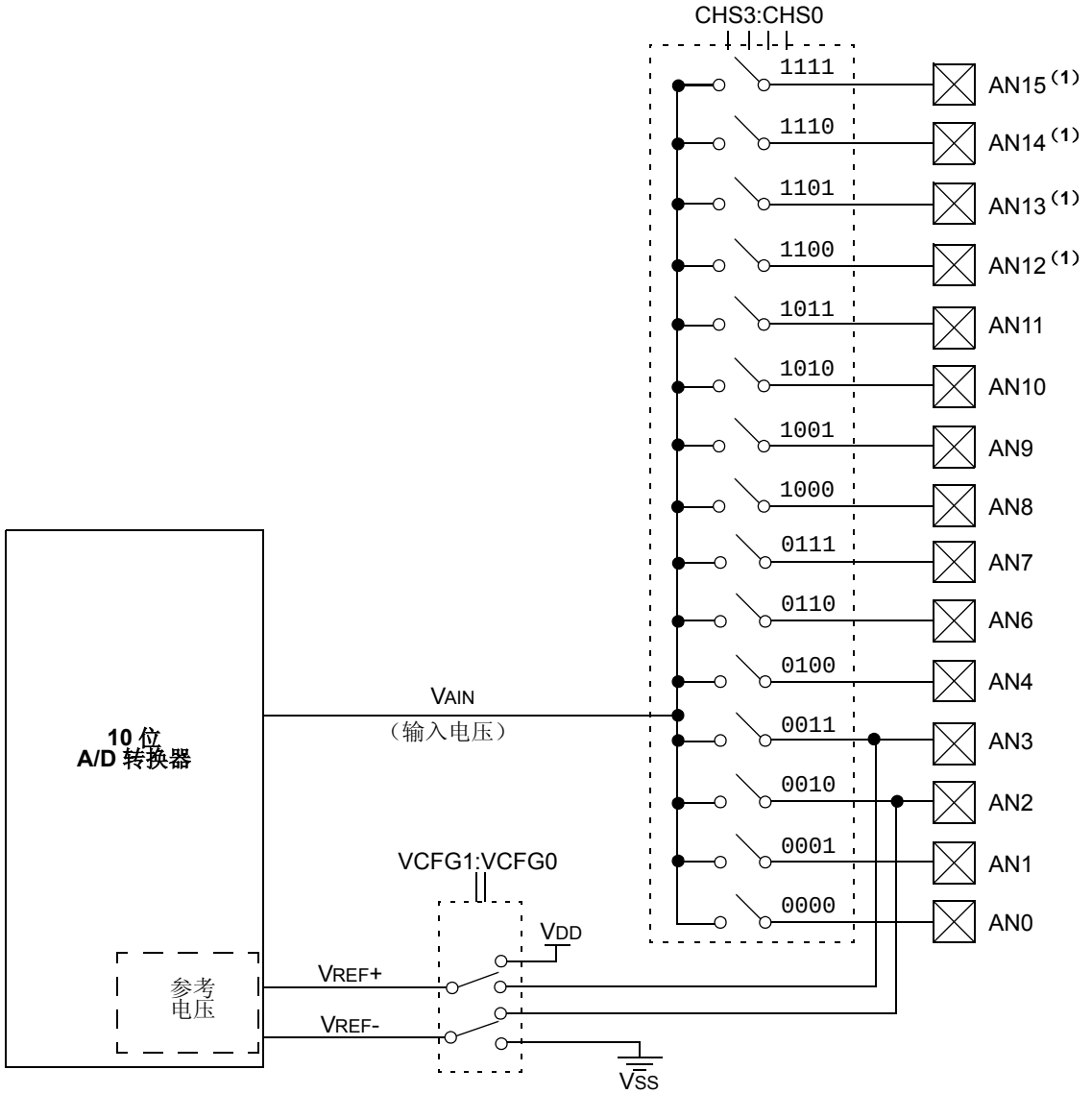
采样保持电路的输出是转换器的输入，A/D 转换器采用逐次逼近法产生转换结果。

可以将每个与 A/D 转换器相关的端口引脚配置为模拟输入或数字 I/O。ADRESH 和 ADRESL 寄存器保存 A/D 转换的结果。当 A/D 转换完成时，转换结果被载入 ADRESH:ADRESL 寄存器对，GO/DONE 位（ADCON0<1>）被清零且 A/D 中断标志位 ADIF 置 1。

器件复位强制所有寄存器进入复位状态，同时迫使 A/D 模块关闭并中止任何正在进行的转换。上电复位时，ADRESH:ADRESL 寄存器对中的值保持不变。上电复位后，这两个寄存器中的值不确定。

A/D 模块的结构框图见图 20-1。

图 20-1：　　A/D 的框图



注　1：在 64 引脚器件上不存在通道 AN12 到 AN15。
2：I/O 引脚上有连到 VDD 和 VSS 的保护二极管。

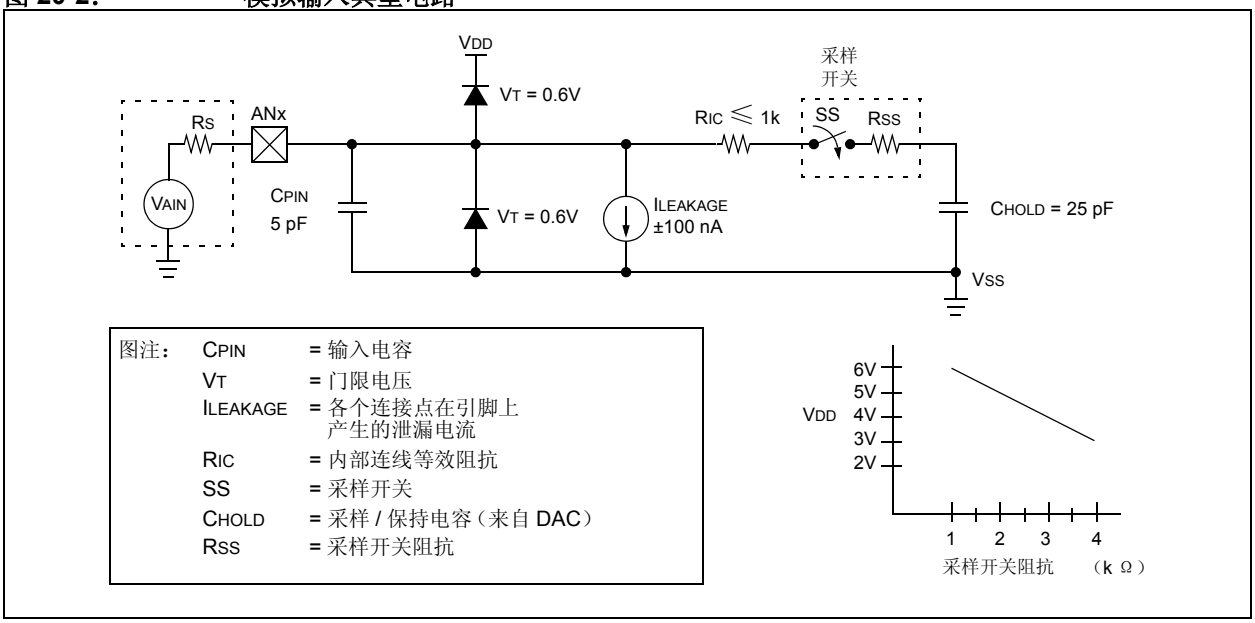
在根据需要配置好 A/D 模块之后，必须在转换开始之前对所选择的通道进行采集。必须将模拟输入通道相应的 TRIS 位选择为输入。采集时间的确定请参见**第 20.1 节“A/D 采集要求”**。在采集完成之后， A/D 转换即可开始。可以将采集时间编程设定为在 GO/DONE 位置 1 和实际转换启动之间。

在执行 A/D 转换时应该遵循以下步骤：

1. 配置 A/D 模块：
   - 配置模拟引脚、参考电压和数字 I/O （通过 ADCON1 寄存器）
   - 选择 A/D 输入通道 （通过 ADCON0 寄存器）
   - 选择 A/D 采集时间 （通过 ADCON2 寄存器）
   - 选择 A/D 转换时钟 （通过 ADCON2 寄存器）
   - 打开 A/D 模块 （通过 ADCON0 寄存器）
2. 需要时，配置 A/D 中断：
   - ADIF 位清零
   - 将 ADIE 位置 1
   - 将 GIE 位置 1
3. 如果需要的话，等待所要求的采集时间。
4. 启动转换：
   - 将 GO/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可判断转换是否完成：
   - 查询 GO/DONE 位是否被清零

   或
   - 等待 A/D 转换中断
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，请根据要求返回步骤 1 或步骤 2。将每位的 A/D 转换时间定义为 TAD。在下一次采集开始前至少需要等待 2 个 TAD。

图 20-2： 模拟输入典型电路



20.1 A/D 采集要求

为了使 A/D 转换器达到规定精度，必须让充电保持电容（CHOLD）充满至输入通道的电平。图 20-2 显示了模拟输入的典型电路。源阻抗（RS）和内部采样开关（RSS）阻抗直接影响给电容 CHOLD 充电所需要的时间。采样开关阻抗（RSS）随器件电压（VDD）变化而改变。源阻抗影响模拟输入的偏移电压（由于引脚泄漏电流）。**建议模拟信号源的最大阻抗为 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开始转换，采样时间必须大于最小采集时间。

注：	当开始转换时，应把保持电容从输入引脚断开。

可以使用公式 20-1 来计算最小采集时间。该公式假设使用的量化误差为 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 20-3 显示了所需的最小采集时间 TACQ 的计算过程。计算结果是基于以下对应用系统的假设得出的：

CHOLD	=	25 pF
Rs	=	2.5 kΩ
转换误差	≤	1/2 LSb
VDD	=	3V → Rss = 2 kΩ
温度	=	85°C （系统最大值）

公式 20-1： 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 20-2： 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 20-3： 计算所需要的最小采集时间

$$T_{ACQ} = T_{AMP} + T_C + T_{COFF}$$
$$T_{AMP} = 0.2\ \mu s$$
$$T_{COFF} = (\text{Temp} - 25°C)(0.02\ \mu s/°C)$$
$$= (85°C - 25°C)(0.02\ \mu s/°C)$$
$$= 1.2\ \mu s$$

只有在温度 > 25°C 时才需要温度系数。当温度低于 25°C 时， $T_{COFF} = 0$ ms。

$$T_C = -(C_{HOLD})(R_{IC} + R_{SS} + R_S) \ln(1/2047)\ \mu s$$
$$= -(25\ pF)(1\ k\Omega + 2\ k\Omega + 2.5k\Omega) \ln(0.0004883)\ \mu s$$
$$= 1.05\ \mu s$$
$$T_{ACQ} = 0.2\ \mu s + 1\ \mu s + 1.2\ \mu s$$
$$= 2.4\ \mu s$$

20.2 选择和配置自动采集时间

ADCON2 寄存器允许用户选择当 GO/DONE 位置 1 后的采集时间。

当 GO/DONE 位置 1，停止采样，启动转换。用户需确保在选择所需要的输入通道和 GO/DONE 置 1 之间留有必需的采集时间。当 ACQT2:ACQT0 位（ADCON2<5:3>）保持在复位状态（“000”）时，就需要由用户确保必需的采集时间。这点与不提供可编程采集时间的器件兼容。

可通过设置 ACQT 位来 为 A/D 模块选择可编程采集时间。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，然后自动开始转换。由于采集时间已被编程，因此 GO/DONE 位会立即置 1 而不需要在选择通道以后等待一个采集时间。

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 并且 A/D 再次开始对当前选择的通道进行采样。如果编程了采集时间，那么将不会有标志显示采集时间是否结束和转换是否开始。

20.3 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 10 位 A/D 转换需要 11 个 TAD 。可用软件选择 A/D 转换的时钟源。

TAD 有以下 7 种可能的选择：

- 2 TOSC
- 4 TOSC
- 8 TOSC
- 16 TOSC
- 32 TOSC
- 64 TOSC
- 内部 RC 振荡器

要进行正确的 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 （欲知更多信息请见表 26-27 参数 130）。

表 20-1 显示了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 20-1： 不同器件工作频率下的 TAD

AD 时钟源 （TAD）		最高器件频率
工作模式	ADCS2:ADCS0	
2 TOSC	000	2.86 MHz
4 TOSC	100	5.71 MHz
8 TOSC	001	11.43 MHz
16 TOSC	101	22.86 MHz
32 TOSC	010	40.0 MHz
64 TOSC	110	40.0 MHz
RC(2)	x11	1.00 MHz(1)

注 1： RC 时钟源的典型 TAD 时间为 4 µs。
2： 当器件工作频率高于 1 MHz 时，整个转换过程必须在休眠模式下进行，否则 A/D 转换精度可能超出规范所允许的范围。

20.4 配置模拟端口引脚

ADCON1、TRISA、TRISE 和 TRISH 寄存器控制 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 TRIS 位置 1 （输入）。如果将 TRIS 位清零（输出），则将该引脚的输出转换为数字电平 （VOH 或 VOL）。

A/D转换操作与CHS3:CHS0位及TRIS位的状态无关。

注 1： 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对模拟输入信号进行转换，引脚上的模拟电平将被正确转换为数字电平。
2： 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。

20.5 A/D 转换

图 20-3 显示了在 GO/DONE 位置 1 且 ACQT2:ACQT0 位被清零后 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 20-4 显示了在 GO/DONE 位置 1 且 ACQT2:CQT0 位被设置为“010”（即在转换开始之前选择了 4 TAD 的采集时间）后 A/D 转换器的工作状态。

在转换期间将 GO/DONE 位清零将中止当前的 A/D 转换。不会用部分完成的 A/D 转换结果更新 A/D 结果寄存器对。这意味着 ADRESH:ADRESL 寄存器对仍将保持上一次转换完成后的值（即上一次写入 ADRESH:ADRESL 寄存器的值）。

在 A/D 转换完成或停止以后，需要等待 2 个 TAD 才能开始下一次采集。等待结束了，将自动开始对所选通道进行采集。

注：	不应在打开 A/D 模块的同一指令中将 GO/DONE 位置 1。

20.6 使用 ECCP2 触发信号

可以通过 ECCP2 模块的“特殊事件触发信号”启动 A/D 转换。这要求将 CCP2M3:CCP2M0 位（CCP2CON<3:0>）编程为 1011，且使能 A/D 模块（ADON 位置 1）。发生触发事件时，GO/DONE 位被置 1，启动 A/D 采集和转换并将 Timer1（或 Timer3）计数器复位为 0。复位 Timer1（或 Timer3）可自动重复 A/D 采集周期，最大限度地降低了软件开销（将 ADRESH/ADRESL 移到目标单元）。在“特殊事件触发信号”将 GO/DONE 位置 1（启动转换）之前，用户必须选择正确的模拟输入通道并设定最小采集时间或者选择合适的 TACQ 时间。

如果未使能 A/D 模块（ADON 清零），则“特殊事件触发信号”将被 A/D 模块忽略，但它仍会将 Timer1（或 Timer3）计数器复位。

图 20-3： A/D 转换 TAD 周期（ACQT2:ACQT0 = 000，TACQ = 0）

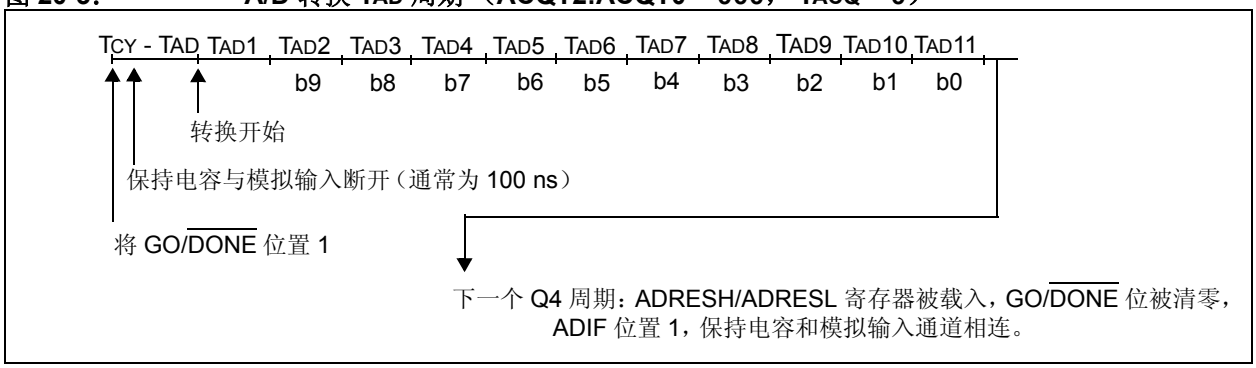
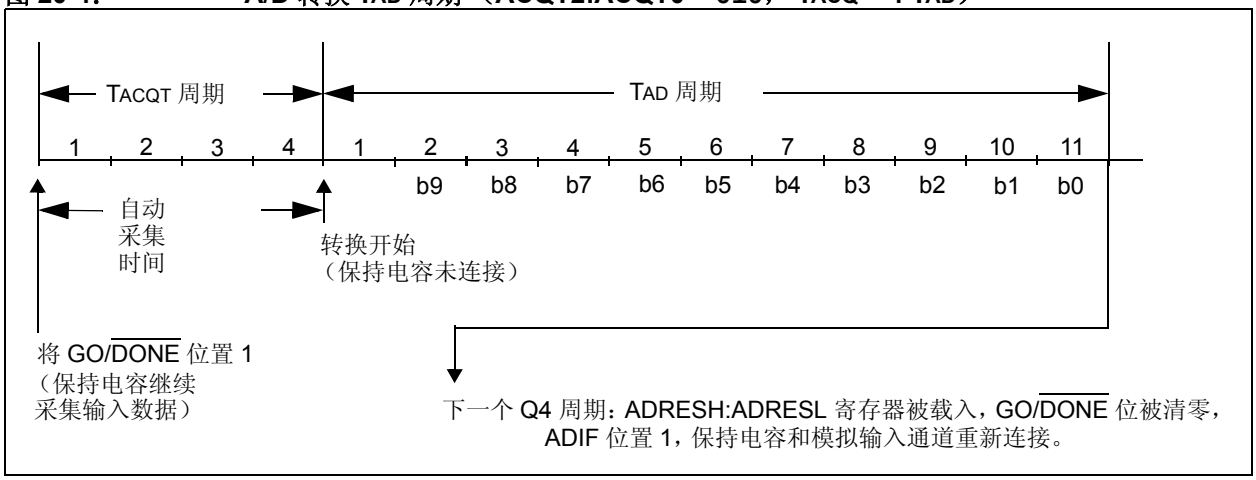


图 20-4： A/D 转换 TAD 周期（ACQT2:ACQT0 = 010，TACQ = 4 TAD）



20.7 A/D 转换器校准

PIC18F87J10 系列器件中的 A/D 转换器包含自校准特性，它可以补偿模块中所产生的偏移。可通过将 ADCAL 位（ADCON0<7>）置 1 来启动自动校准。下一次 GO/$\overline{\text{DONE}}$ 位被置 1 时，此模块将执行“虚拟”转换（即不读取任何输入通道）并将转换结果存储在内部以补偿偏移。因此，后续的偏移都将得到补偿。

在校准过程中假设器件处于相对稳定的工作状态。如果使用 A/D 校准，它应该在每次器件复位之后或在工作条件发生重要变化时执行。

20.8 在功耗管理模式下的操作

在功耗管理模式中，自动采集时间和 A/D 转换时钟的选择在一定程度上可由时钟源和频率决定。

如果要在器件处于功耗管理模式时进行 A/D 转换，ADCON2 中的 ACQT2:ACQT0 和 ADCS2:ADCS0 位就应该根据将使用的功耗管理模式时钟进行更新。在进入功耗管理模式之后（两种功耗管理运行模式中的一种），就可以开始 A/D 采集或转换。采集或转换开始以后，器件仍应继续使用与功耗管理模式相同的时钟源直到转换完成。如果需要的话，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 RC 时钟源。

在休眠模式下工作需要选择 A/D RC 时钟源。如果将 ACQT2:ACQT0 位设置为“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 位必须在转换开始之前被清零。

表 20-2： A/D 寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	51
PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IF	CCP1IE	TMR2IE	TMR1IE	51
IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	CCP1IP	TMR2IP	TMR1IP	51
PIR2	OSCFIF	CMIF	—	—	BCL1IF	—	TMR3IF	CCP2IF	51
PIE2	OSCFIE	CMIE	—	—	BCL1IE	—	TMR3IE	CCP2IE	51
IPR2	OSCFIP	CMIP	—	—	BCL1IP	—	TMR3IP	CCP2IP	51
ADRESH	A/D 结果寄存器高字节								50
ADRESL	A/D 结果寄存器低字节								50
ADCON0	ADCAL	—	CHS3	CHS3	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADON	50
ADCON1	—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0	50
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	50
CCP2CON	P2M1	P2M0	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	51
PORTA	—	—	RA5	RA4	RA3	RA2	RA1	RA0	52
TRISA	—	—	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	52
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	52
TRISF	TRISF5	TRISF4	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	52
PORTH(1)	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0	52
TRISH(1)	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0	52

图注： — = 未用位，读为 0。 A/D 转换未使用阴影单元。

注 1： 该寄存器在 64 引脚器件上不存在。

注：

21.0 比较器模块

模拟比较器模块包含两个比较器，可以用多种方式对它们进行配置。可以选用与 RF1 到 RF6 引脚复用的模拟输入及片上参考电压（见**第 22.0 节 “比较器参考电压模块”**）作为输入。数字输出（正常或翻转）可从引脚电平获取也可通过控制寄存器读取。

CMCON 寄存器（寄存器 21-1）选择比较器的输入和输出配置。图 21-1 给出了各种比较器配置。

寄存器 21-1：　CMCON：比较器模块控制寄存器

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

bit 7　**C2OUT**：比较器 2 输出位
当 C2INV=0 时：
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-
当 C2INV=1 时：
1 = C2 VIN+ < C2 VIN-
0 = C2 VIN+ > C2 VIN-

bit 6　**C1OUT**：比较器 1 输出位
当 C1INV=0 时：
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-
当 C1INV=1 时：
1 = C1 VIN+ < C1 VIN-
0 = C1 VIN+ > C1 VIN-

bit 5　**C2INV**：比较器 2 输出翻转位
1=C2 输出翻转
0=C2 输出不翻转

bit 4　**C1INV**：比较器 1 输出翻转位
1=C1 输出翻转
0=C1 输出不翻转

bit 3　**CIS**：比较器输入选择开关位
当 CM2:CM0 = 110 时：
1 = C1 VIN- 连接到 RF5/AN10/CVREF
C2 VIN- 连接到 RF3/AN8
0 = C1 VIN- 连接到 RF6/AN11
C2 VIN- 连接到 RF4/AN9

bit 2-0　**CM2:CM0**：比较器模式位
图 21-1 给出了比较器的几种模式以及相应 CM2:CM0 位的设置。

图注：			
R = 可读位	W = 可写位	U= 未用位，读为 0	
-n = 上电复位时的值	1 = 置位	0 = 清零	x = 未知

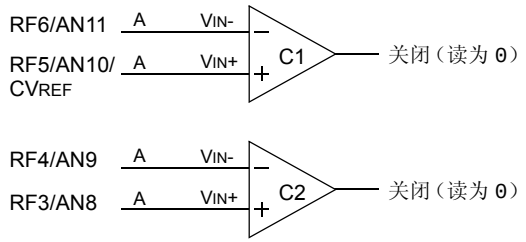
21.1 比较器配置

比较器有 8 种工作模式，如图 21-1 所示。 CMCON 寄存器的 CM2:CM0 位用于选择这些模式。TRISF 寄存器控制每种模式下比较器引脚的数据方向。如果改变比较器模式，由于存在特定的模式改变延迟 （如**第 26.0 节 “电气规范”**所示），比较器的输出电平可能会在此延迟期间无效。

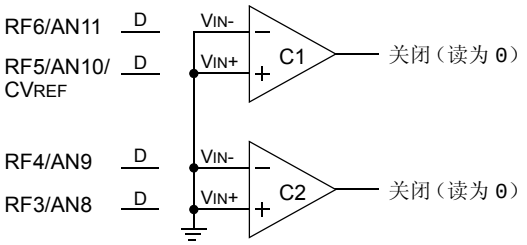
注：	改变比较器工作模式的过程中，应禁止比较器的中断，否则会产生错误中断。

图 21-1： 比较器 I/O 工作模式

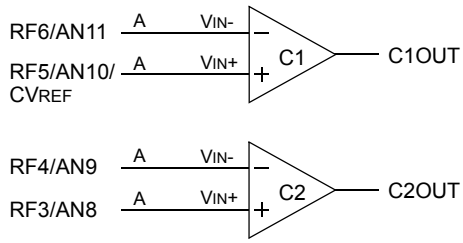
比较器输出禁止
CM2:CM0 = 000



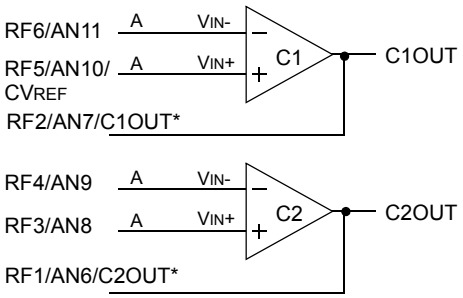
比较器关闭（POR 时的默认值）
CM2:CM0 = 111



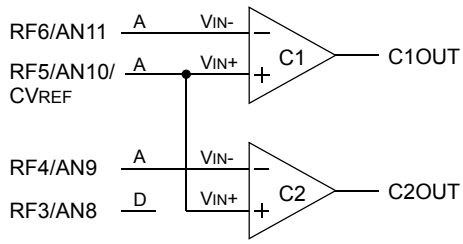
两个独立的比较器
CM2:CM0 = 010



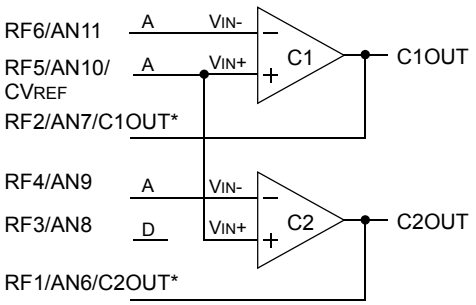
两个独立的带输出的比较器
CM2:CM0 = 011



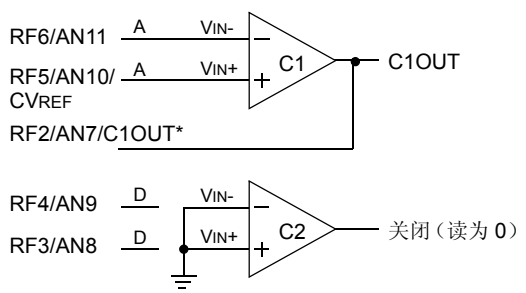
两个具有公共参考端的比较器
CM2:CM0 = 100



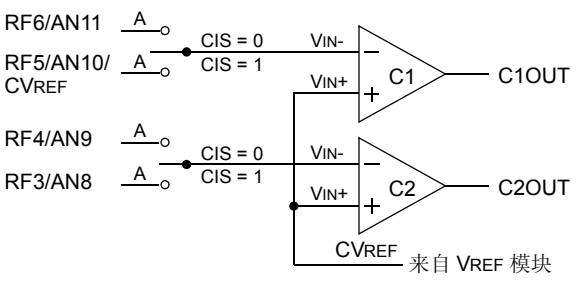
两个具有公共参考端且带输出的比较器
CM2:CM0 = 101



一个带输出的独立比较器
CM2:CM0 = 001



四路输入复用给两个比较器
CM2:CM0 = 110



A = 模拟输入，端口始终读为 0； D = 数字输入； CIS（CMCON<3>）是比较器输入开关

* 将 TRISF<2:1> 位置 1 会通过把引脚配置为输入来禁止比较器输出。

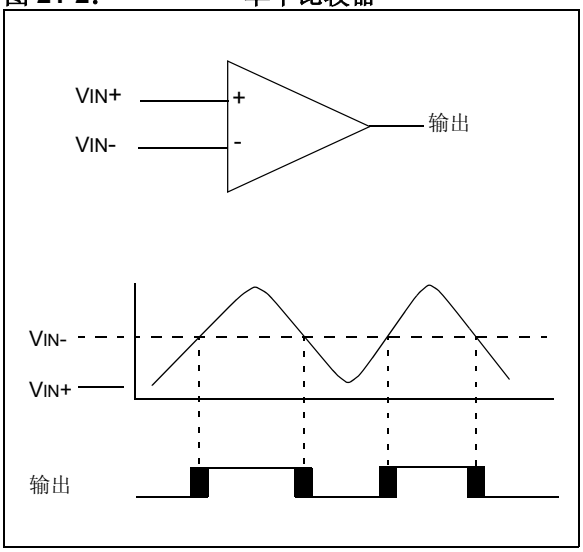
21.2 比较器的工作原理

图 21-2 所示为单个比较器以及模拟输入电平和数字输出之间的关系。如果 VIN+ 上的模拟输入电平小于 VIN- 上的模拟输入电平，那么比较器将输出数字低电平。当 VIN+ 上的模拟输入电平高于 VIN- 上的模拟输入电平时，比较器输出数字高电平。图 21-2 中比较器输出的阴影部分表示由于输入偏移和响应时间所造成的不确定区。

21.3 比较器参考电压

根据不同的工作模式，比较器可使用外部或内部参考电压。将加在 VIN- 上的模拟信号和加在 VIN+ 上的信号相比较，并相应地调整比较器的数字输出 （图 21-2）。

图 21-2： 单个比较器



21.3.1 外部参考信号

当使用外部参考电压时，可将比较器模块中的两个比较器配置为使用同一个参考源或使用不同的参考源。然而，门限检测器应用可能要求使用同一个参考源。参考信号必须在 VSS 和 VDD 之间，并且可被施加到比较器的任一引脚上。

21.3.2 内部参考信号

比较器模块也可以选择使用比较器参考电压模块内部产生的参考电压。**第 22.0 节 “比较器参考电压模块”**详细介绍了该模块。

只有在两个比较器复用四路输入的模式（CM2:CM0 = 110）中才可使用内部参考电压。在该模式下，内部参考电压被施加到两个比较器的 VIN+ 引脚上。

21.4 比较器的响应时间

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。如果内部参考电压发生了改变，在使用比较器的输出时必须考虑到内部参考电压的最大延时。否则，应该使用比较器的最大延时 （见**第 26.0 节 “电气规范”**）。

21.5 比较器输出

通过读 CMCON 寄存器中的相应位可读取比较器的输出。这些位是只读的。比较器的输出也可以直接输出到 I/O 引脚 RF1 和 RF2。当使能时， RF1 和 RF2 引脚输出路径上的多路开关将发生切换，并且各个引脚的输出将与比较器的输出异步。每个比较器的不确定区的大小与规范里给出的输入偏移电压和响应时间有关。图 21-3 为比较器输出的框图。

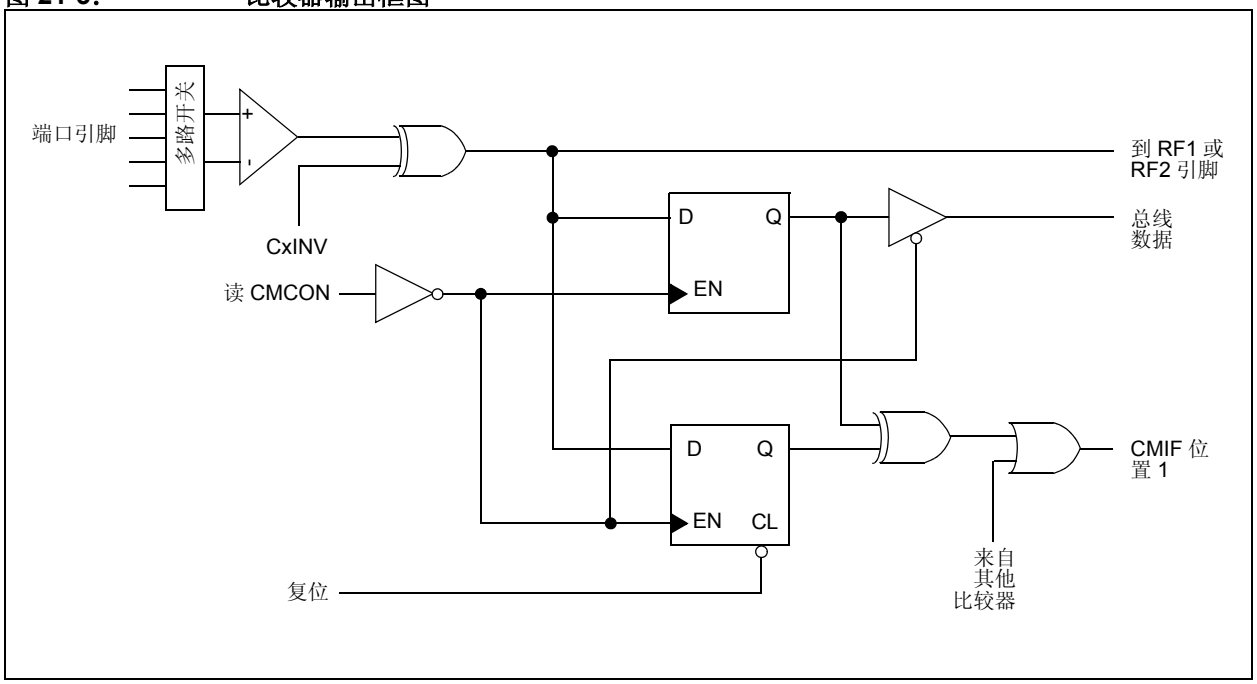
在该模式下， TRISF 仍作为 RF2 和 RF1 引脚的输出使能 / 禁止位。

使用 C2INV 和 C1INV 位（CMCON<5:4>）可以改变比较器输出的极性。

注 1： 当读取端口寄存器时，所有配置为模拟输入的引脚都读为 0。配置为数字输入的引脚将根据施密特触发器输入规范转换模拟输入信号。

2： 对被定义为数字输入的任何引脚施加模拟电平均可能会使输入缓冲器的电流消耗超过规定值。

图 21-3：　　比较器输出框图



21.6　比较器中断

一旦两个比较器中的任意一个的输出值发生了变化，就会将相应比较器的中断标志位置 1。需要用软件来保持输出位的状态信息，即读取 CMCON<7:6>，来判断实际发生的变化。CMIF 位（PIR2<6>）是比较器中断标志位。CMIF 位必须通过清零复位。因为也可以向 CMCON 寄存器写入“1”，所以可以模拟中断的发生。

必须将 CMIE 位（PIE2<6>）和 PEIE 位（INTCON<6>）置位以允许中断。此外，也必须将 GIE（INTCON<7>）位置 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 CMIF 位仍会置 1，但却不允许中断。

注：	当执行读操作时（Q2 周期开始时），如果 CMCON 寄存器（C1OUT 或 C2OUT）发生变化，那么 CMIF（PIR2 寄存器）中断标志位可能不会被置 1。

用户可用以下方式在中断服务程序中清除该中断：

a）对 CMCON 的任何读或写均将中止电平不匹配状态。

b）清零中断标志位 CMIF。

引脚上电平不匹配的情况会不断地将CMIF标志位置1。读 CMCON 寄存器将结束引脚上电平不匹配的情况，并允许将 CMIF 标志位清零。

21.7　比较器在休眠模式下的工作方式

当比较器处于活动状态而器件处于休眠模式时，比较器仍可正常工作并可使用比较器中断（如果允许的话）。在允许中断时，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，如比较器规范中所示。若要把休眠状态下的功耗减少到最小，可在进入休眠模式前关闭比较器模块（CM2:CM0 = 111）。器件从休眠模式被唤醒时，CMCON 寄存器的内容不受影响。

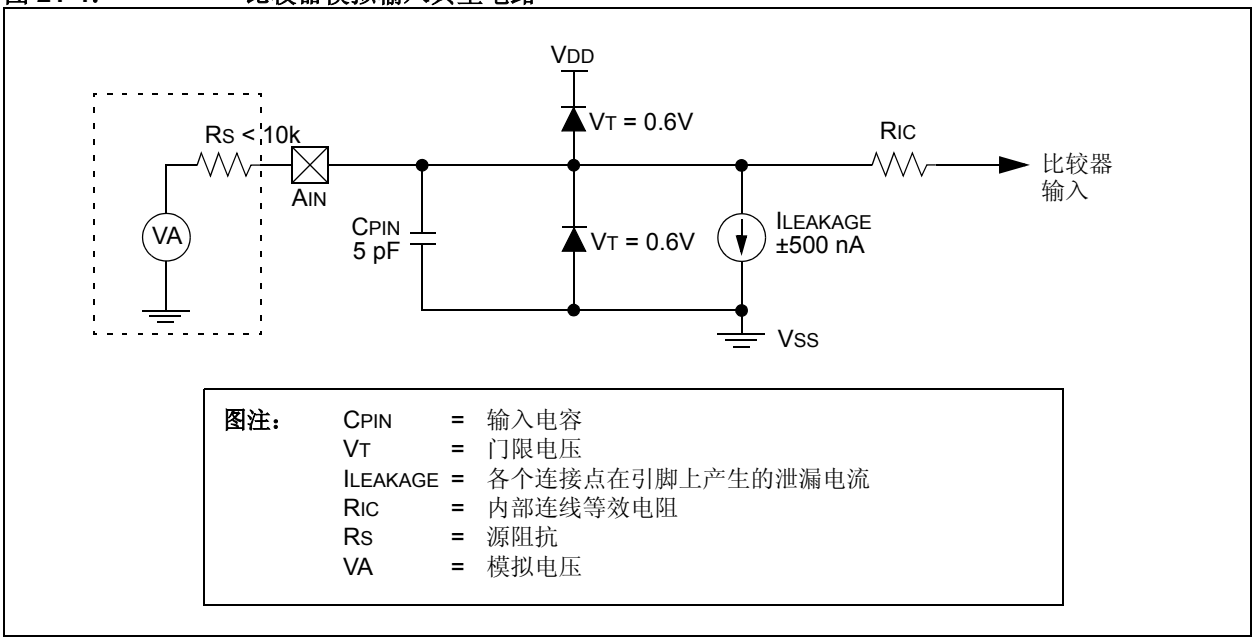
21.8　复位的影响

器件复位强制 CMCON 寄存器进入复位状态，导致比较器模块关闭（CM2:CM0 = 111）。但是，在器件复位时输入引脚（RF3 到 RF6）被默认配置为模拟输入。PCFG3:PCFG0 位（ADCON1<3:0>）决定这些引脚的 I/O 配置。因此，当复位时引脚呈现模拟输入状态，此时器件电流达到最小。

21.9 模拟输入连接注意事项

图 21-4 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，因此它们与 VDD 和 VSS 之间加有反向偏置的二极管，从而将模拟输入电压限制在 VSS 和 VDD 之间。一旦输入电压大于或小于该范围 0.6V，就会有一个二极管正偏从而发生钳位。模拟信号源的最大阻抗值推荐为 10 k Ω。任何连接到模拟输入引脚上的外部元件（如电容和齐纳二极管等）的泄漏电流应该极小。

图 21-4：　　比较器模拟输入典型电路



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 = 源阻抗
VA = 模拟电压

表 21-1：　　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	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
PIR2	OSCFIF	CMIF	—	—	BCL1IF	—	TMR3IF	CCP2IF	51
PIE2	OSCFIE	CMIE	—	—	BCL1IE	—	TMR3IE	CCP2IE	51
IPR2	OSCFIP	CMIP	—	—	BCL1IP	—	TMR3IP	CCP2IP	51
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	51
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	51
PORTF	RF7	RF6	RF5	RF4	RF3	RF2	RF1	—	52
LATF	LATF7	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	—	52
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	52

图注：　— = 未用位，读为 0。比较器模块未使用阴影单元。

注：

22.0 比较器参考电压模块

比较器参考电压模块是一个 16 阶的梯形电阻网络，提供多个参考电压以供选择。虽然它的主要作用是给模拟比较器提供参考电压，但也可以独立于模拟比较器使用。

图 22-1 所示为模块的框图。分割后的梯形电阻可提供两种量程范围的 CVREF 值，并且还具有断电功能以在不使用参考电压时降低功耗。模块的供电参考电压由器件 VDD/VSS 或外部参考电压提供。

22.1 配置比较器参考电压

比较器参考电压模块由 CVRCON 寄存器（寄存器 22-1）控制，可提供两种范围的输出电压，每种范围都具有 16 个电压等级。CVRR 位（CVRCON<5>）选择要用的电压范围。这两种范围的主要区别在于由 CVREF 选择位（CVR3:CVR0）选定的步长不同，其中一个范围具有更高的分辨率。下面是计算比较器参考电压输出值的公式：

如果 CVRR = 1：
CVREF = ((CVR3:CVR0)/24) x (CVRSRC)

如果 CVRR = 0：
CVREF = (CVRSRC/4) + ((CVR3:CVR0)/32) x (CVRSRC)

比较器参考供电电压可以来自 VDD 和 VSS，或者与 RA2 和 RA3 复用的外部 VREF+ 和 VREF-。CVRSS 位（CVRCON<4>）用于选择电压源。

在改变 CVREF 输出值时，必须考虑到比较器参考电压的稳定时间（见**第 26.0 节 “电气规范”**中的表 25-3）。

寄存器 22-1： 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(1)	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

bit 7 **CVREN**：比较器参考电压使能位
1= 开 CVREF 电路
0= 关 CVREF 电路

bit 6 **CVROE**：比较器 VREF 输出使能位 (1)
1= 在 RF5/AN10/CVREF 引脚上输出 CVREF 电压电平
0= CVREF 电压与 RF5/AN10/CVREF 引脚断开

注 1：CVROE 优先级高于 TRISF<5> 位。

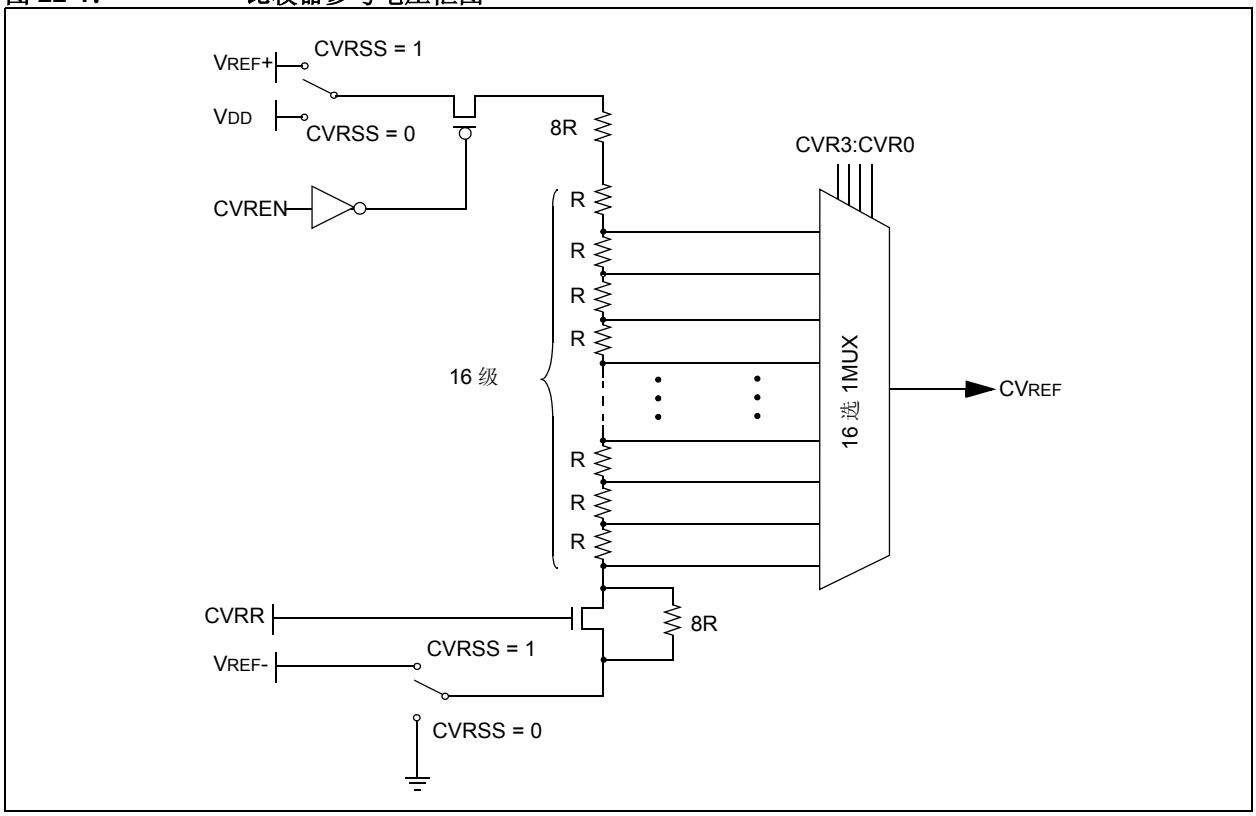
bit 5 **CVRR**：比较器 VREF 范围选择位
1= 0 到 0.667CVRSRC，步长为 CVRSRC/24（低电压范围）
0= 0.25CVRSRC 到 0.75CVRSRC，步长为 CVRSRC/32（高电压范围）

bit 4 **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 = VDD – VSS

bit 3-0 **CVR3:CVR0**：比较器 VREF 值选择位（0 ≤（CVR3:CVR0）≤ 15）
当 CVRR = 1：
CVREF=((CVR3:CVR0)/24) • (CVRSRC)
当 CVRR = 0：
CVREF=(CVRSRC/4)+((CVR3:CVR0)/32)•(CVRSRC)

图注：			
R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置位	0 = 清零	x = 未知

图 22-1：　　比较器参考电压框图



22.2　参考电压精度 / 误差

由于模块结构的原因，模块无法实现满量程参考电压。梯形电阻网络中顶端和底部的晶体管（图 22-1）使 CVREF 无法达到参考电压源的满幅值。参考电压来自于参考电压源；因此，CVREF 输出电平会随参考电压源一起波动。**第 26.0 节 “电气规范”**中可找到经测试得到的参考电压绝对精度值。

22.3　在休眠模式下工作

当中断或看门狗定时器超时而唤醒器件时， CVRCON 寄存器内容不受影响。为了最大限度降低休眠模式下的电流消耗，应关闭参考电压模块。

22.4　复位的影响

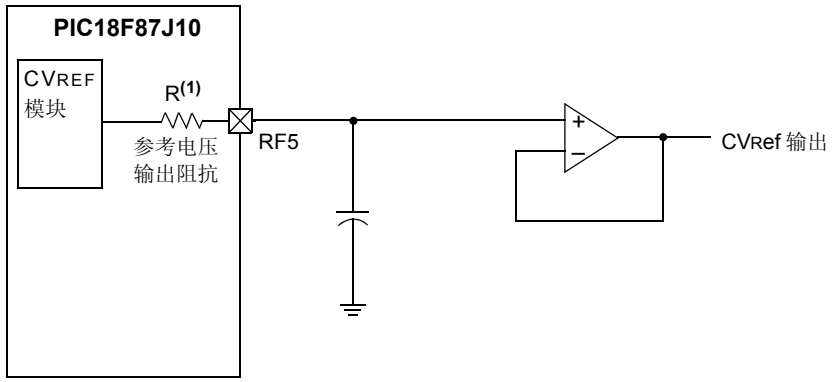
器件复位通过清零 CVREN （CVRCON<7>）从而禁止参考电压模块；通过清零 CVROE 位（CVRCON<6>），复位还可将参考电压与 RA2 引脚断开，并且通过清零 CVRR（CVRCON<5>），可选择高电压范围。同时 CVR 值选择位也被清零。

22.5　连接注意事项

参考电压模块独立于比较器模块工作。如果 CVROE 位置 1，参考电压发生器的输出引脚可能会连到 RF5 引脚。使能参考电压输出到 RA2 引脚上 （如果引脚被配置为数字输入）将增加电流消耗。在使能 CVRSS 时，将 RF5 作为数字输出连接也将增加电流消耗。

RF5 引脚可以用作简单 D/A 输出，但其驱动能力有限。由于驱动能力有限，因此当VREF有外部连接时必须在参考电压输出端上使用缓冲器。图 22-2 举例说明了这一缓冲技术。

图 22-2：　　比较器参考电压输出缓冲示例



注　1： R 取决于参考电压配置位 CVRCON<3:0> 和 CVRCON<5> 的设置。

表 22-1：　　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	复位值所在页
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	51
CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	51
TRISF	TRISF7	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	—	52

图注：　— = 未用，读为 0。比较器参考电压模块不使用阴影单元。

注：

23.0 CPU 的特殊功能

PIC18F87J10 系列器件包含几项特殊功能旨在最大限度地提高系统的可靠性，并通过减少外部元件将成本降至最低。这些功能包括：

- 振荡器选择
- 复位：
  - 上电复位 （POR）
  - 上电延迟定时器 （PWRT）
  - 振荡器起振定时器 （OST）
  - 欠压复位 （BOR）
- 中断
- 看门狗定时器 （WDT）
- 故障保护时钟监视器
- 双速启动
- 代码保护
- 在线串行编程

根据具体应用对频率、功耗、精度和成本的要求配置振荡器。在**第 2.0 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。

PIC18F87J10 系列器件除了为复位提供上电延时定时器和振荡器起振定时器之外，还具有一个可配置的看门狗定时器，该定时器由软件控制。

器件自带的 RC 振荡器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有益功能。 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，此时主时钟源正在进行自身的起振延时。

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。

23.1 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中从 300000h 开始的单元中。表 23-2 列出了所有的配置位。从寄存器 23-1 到寄存器 23-6 详细解释了各配置位的不同功能。

23.1.1 配置 PIC18F87J10 系列器件的注意事项

与先前的 PIC18 单片机不同， PIC18F87J10 系列器件不再使用耐久性存储寄存器存储配置信息。配置字节以易失性存储方式实现，这就意味着在器件每次上电时都必须对配置数据进行编程。

配置数据存储在片内程序存储空间顶部的 4 个字中，这些字被称为闪存配置字。配置位数据按表 23-2 中相同的次序存储在程序存储器中，CONFIG1L 位于地址最低的单元，CONFIG3H 位于地址最高的单元。在器件上电时这些数据被自动装入正确的配置寄存器。

当为这些器件创建应用程序时，用户应该为配置数据特别分配闪存配置字单元，以确保当编译代码时程序代码不会存储在该地址上。

在上电复位时用于配置位的易失性存储单元始终复位为 1。对于其他类型的复位事件，将保留和使用先前已编程的值，而无需从程序存储器重新装入数据。

程序存储器中 CONFIG1H、CONFIG2H 和 CONFIG3H 的高 4 位也应为 1111。这样当这些配置字被意外执行到时，被当作一条 NOP 指令。由于配置位在对应的单元中未实现的，因此向这些单元写 1 不会影响器件工作。

为了避免在代码执行期间配置被意外更改，可编程配置位只可被写入一次。在上电周期内对位进行初始化之后就不能再次写入该位了。改变器件的配置需要对器件重新上电。

表 23-1： 配置寄存器与闪存配置字的映射

配置字节	代码地址	配置寄存器地址
CONFIG1L	XXXF8h	300000h
CONFIG1H	XXXF9h	300001h
CONFIG2L	XXXFAh	300002h
CONFIG2H	XXXFBh	300003h
CONFIG3L	XXXFCh	300004h
CONFIG3H	XXXFDh	300005h
CONFIG4L(1)	XXXFEh	300006h
CONFIG4H(1)	XXXFFh	300007h

注 1： PIC18F87J10 器件未用。

表 23-2： 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	缺省 / 未编程的值 (1)
300000h	CONFIG1L	$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN	111- ---1
300001h	CONFIG1H	—(2)	—(2)	—(2)	—(2)	—(3)	CP0	—	—	---- 01--
300002h	CONFIG2L	IESO	FCMEN	—	—	—	FOSC2	FOSC1	FOSC0	11-- -111
300003h	CONFIG2H	—(2)	—(2)	—(2)	—(2)	WDTPS3	WDTPS2	WDTPS1	WDTPS0	---- 1111
300004h	CONFIG3L	WAIT(4)	BW(4)	EMB1(4)	EMB0(4)	EASHFT(4)	—	—	—	1111 1---
300005h	CONFIG3H	—(2)	—(2)	—(2)	—(2)	—	—	ECCPMX(4)	CCP2MX	---- --11
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx(5)
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0000 10x1(5)

图注： x = 未知， u = 不变，— = 未用 （读为 0）。未使用阴影单元。

注 1： 这些值反映出厂时和上电复位后的未编程状态。在所有其他复位状态中，配置字节保持原先的编程状态。

2： 程序存储器中这些位的值应始终为 1。这样可确保如果意外地执行了这些单元，将会执行 NOP 指令。

3： 该位应始终保持为 0。

4： 只在 80 引脚器件中实现。

5： 参见寄存器 23-7 和寄存器 23-8 查询 DEVID 的值。这些寄存器为只读寄存器，用户不能对其进行编程。

寄存器 23-1：　CONFIG1L：配置寄存器 1 的低字节 （字节地址为 300000h）

R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0	U-0	R/WO-1
$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	WDTEN
bit 7							bit 0

bit 7　**$\overline{\text{DEBUG}}$**：后台调试器使能位
1= 禁止后台调试器， RB6 和 RB7 被配置为通用 I/O 引脚
0= 使能后台调试器， RB6 和 RB7 专用于在线调试

bit 6　**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　**STVREN**：堆栈上溢 / 下溢复位使能位
1= 使能堆栈上溢 / 下溢复位
0= 禁止堆栈上溢 / 下溢复位

bit 4-1　**未用**：读为 0

bit 0　**WDTEN**：看门狗定时器使能位
1= 使能 WDT
0= 禁止 WDT （控制位为 SWDTEN 位）

图注：		
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0
-n = 未对器件编程时的值	1 = 置位	0 = 清零

寄存器 23-2：　CONFIG1H：配置寄存器 1 的高字节 （字节地址为 300001h）

U-0	U-0	U-0	U-0	U-0	R/WO-1	U-0	U-0
—	—	—	—	—(1)	CP0	—	—
bit 7							bit 0

bit 7-3　**未用**：读为 0

bit 2　**CP0**：代码保护位
1= 程序存储器未受代码保护
0= 程序存储器受代码保护

bit 1-0　**未用**：读为 0

注　1：该位应始终保持为 0。

图注：		
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0
-n = 未对器件编程时的值	1 = 置位	0 = 清零

寄存器 23-3：　CONFIG2L：配置寄存器 2 的低字节 （字节地址为 300002h）

R/WO-1	R/WO-1	U-0	U-0	U-0	R/WO-1	R/WO-1	R/WO-1
IESO	FCMEN	—	—	—	FOSC2	FOSC1	FOSC0
bit 7							bit 0

bit 7　**IESO**：双速启动 （内部 / 外部振荡器切换）控制位
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 5-3　**未用**：读为 0

bit 2　**FOSC2**：默认 / 复位系统时钟选择位
1= 当 OSCCON<1:0>=00 时，使用由 FOSC1:FOSC0 选择的时钟作为系统时钟
0= 当 OSCCON<1:0>=00 时，使用 INTRC 作为系统时钟

bit 1-0　**FOSC1:FOSC0**：振荡器选择位
11= EC 振荡器， PLL 使能并由软件控制， OSC2 用作 CLKO 功能
10= EC 振荡器， OSC2 用作 CLKO 功能
01= HS 振荡器， PLL 使能并由软件控制
00= HS 振荡器

图注：		
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0
-n = 未对器件编程时的值	1 = 置位	0 = 清零

寄存器 23-4：　CONFIG2H：配置寄存器 2 的高字节 （字节地址为 300003h）

U-0	U-0	U-0	U-0	R/WO-1	R/WO-1	R/WO-1	R/WO-1
—	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

bit 7-4　**未用**：读为 0

bit 3-0　**WDTPS3:WDTPS0**：看门狗定时器后分频比选择位
1111 = 1:32,768
1110 = 1:16,384
1101 = 1:8,192
1100 = 1:4,096
1011 = 1:2,048
1010 = 1:1,024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

图注：		
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0
-n = 未对器件编程时的值	1 = 置位	0 = 清零

寄存器 23-5：　CONFIG3L：配置寄存器 3 的低字节 （字节地址为 300004h）

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-0	U-0	U-0
WAIT(1)	BW(1)	EMB1(1)	EMB0(1)	EASHFT(1)	—	—	—
bit 7							bit 0

bit 7　**WAIT**：外部总线等待使能位 (1)
1= 禁止外部存储器总线上的操作的等待状态
0= 使能外部存储器总线上的操作的等待状态

bit 6　**BW**：数据总线宽度选择位 (1)
1= 16 位外部总线模式
0= 8 位外部总线模式

bit 5-4　**EMB1:EMB0**：外部存储器总线配置位 (1)
11= 单片机模式，禁止外部总线
10= 扩展单片机模式， 12 位地址模式
01= 扩展单片机模式， 16 位地址模式
00= 扩展单片机模式， 20 位地址模式

bit 3　**EASHFT**：外部地址总线移位使能位 (1)
1= 使能地址移位；外部总线上的地址是以 000000h 为起始地址的偏移值
0= 禁止地址移位；外部总线上的地址反映 PC 值

bit 2-0　**未用**：读为 0

注　1：仅在 80 引脚的器件上可用。

图注：		
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0
-n = 未对器件编程时的值	1 = 置位	0 = 清零

寄存器 23-6：　CONFIG3H：配置寄存器 3 的高字节 （字节地址为 300005h）

U-0	U-0	U-0	U-0	U-0	U-0	R/WO-1	R/WO-1
—	—	—	—	—	—	ECCPMX(1)	CCP2MX
bit 7							bit 0

bit 7-2　**未用**：读为 0

bit 1　**ECCPMX**：ECCPx 多路复用位 (1)
1 = ECCP1 输出 （P1B/P1C）与 RE6 和 RE5 复用；
　　ECCP3 输出 （P3B/P3C）与 RE4 和 RE3 复用
0 = ECCP1 输出 （P1B/P1C）与 RH7 和 RH6 复用；
　　ECCP3 输出 （P3B/P3C）与 RH5 和 RH4 复用

bit 0　**CCP2MX**：ECCP2 多路复用位
1 = ECCP2/P2A 与 RC1 复用
0 = 在单片机模式下 ECCP2/P2A 与 RE7 复用 （所有器件）
　　或在扩展单片机模式下与 RB3 复用 （仅 80 引脚器件）

注　1：仅在 80 引脚的器件上可用。

图注：		
R = 可读位	WO = 一次性写入位	U = 未用位，读为 0
-n = 未对器件编程时的值	1 = 置位	0 = 清零

寄存器 23-7：　PIC18F87J10 系列器件的器件 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

bit 7-5 **DEV2:DEV0**：器件 ID 位
111 = PIC18F85J10
101 = PIC18F67J10
100 = PIC18F66J15
011 = PIC18F66J10 或 PIC18F87J10
010 = PIC18F65J15 或 PIC18F86J15
001 = PIC18F65J10 或 PIC18F86J10
000 = PIC18F85J15

注： 其中DEV2:DEV0的值被多个器件号共享使用，某个特定器件需使用整个DEV10:DEV0位序列标识。

bit 4-0 **REV4:REV0**：版本 ID 位
这些位用于表明器件版本。

图注：	
R = 只读位	U = 未用位，读为 0
-n = 未对器件编程时的值	u = 编程后状态不变

寄存器 23-8：　PIC18F87J10 系列器件的器件 ID 寄存器 2

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

bit 7-0 **DEV10:DEV3**：器件 ID 位
这些位与器件 ID 寄存器 1 中的 DEV2:DEV0 结合使用以标识器件号。
0001 0101 = PIC18F65J10/65J15/66J10/66J15/67J10/85J10 器件
0001 0111 = PIC18F85J15/86J10/86J15/87J10 器件

注： DEV10:DEV3的值可能会与其他器件系列共享使用。通过使用整个DEV10:DEV0位序列标识某一特定器件。

图注：	
R = 只读位	U = 未用位，读为 0
-n = 未对器件编程时的值	u = 编程后状态不变

23.2 看门狗定时器 （WDT）

对于 PIC18F87J10 系列器件， WDT 由 INTRC 振荡器驱动。当使能 WDT 时，也将使能时钟源。WDT 超时溢出周期的标称值为 4 ms，其稳定性与 INTRC 振荡器相同。

4 ms的WDT超时溢出周期与16位的后分频比值相乘。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，因此可获得的超时溢出周期范围为 4 ms 至 131.072 秒 （2.18 分钟）。当发生以下任一事件时， WDT 和后分频器将被清零，这些事件包括：执行 SLEEP 或 CLRWDT 指令和发生时钟故障 （主时钟或 Timer1 振荡器）。

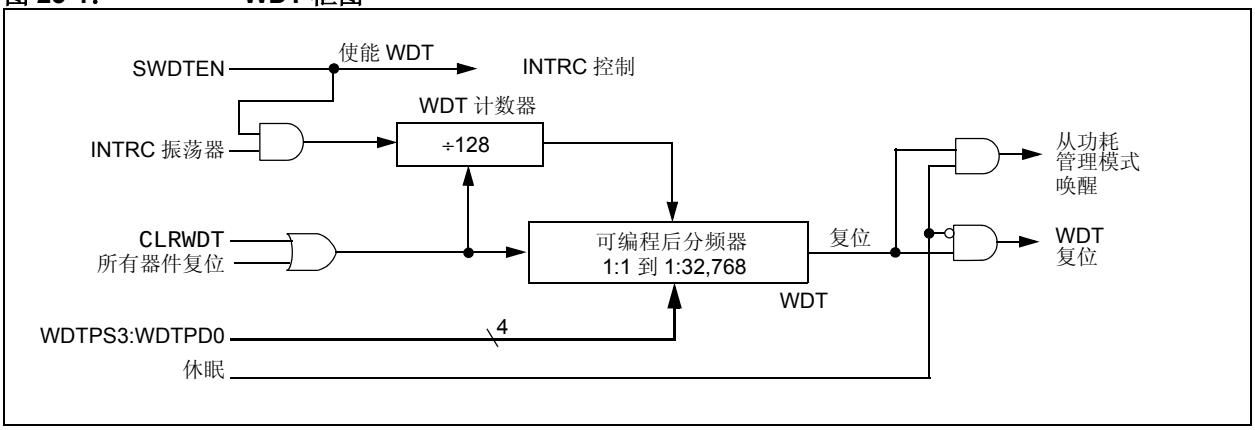
注 1： 当执行 CLRWDT 或 SLEEP 指令时， WDT 和后分频器的计数值将被清零。

2： 当执行 CLRWDT 指令时，后分频器的计数值将被清零。

23.2.1 控制寄存器

WDTCON 寄存器 （寄存器 23-9）为可读写寄存器。SWDTEN 位使能或禁止 WDT 的操作。仅当 WDT 配置位禁止 WDT 时，允许使用软件改写该配置位来使能 WDT。

图 23-1： WDT 框图



寄存器 23-9： WDTCON：看门狗定时器控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	SWDTEN(1)
bit 7							bit 0

bit 7-1 **未用：** 读为 0

bit 0 **SWDTEN：** 由软件控制的看门狗定时器使能位 (1)

1= 打开看门狗定时器
0= 关闭看门狗定时器

注 1： 当 WDTEN 配置位使能时该位不起作用。

图注：

R = 可读位	W = 可写位	U = 未用位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

表 23-3： 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	复位值所在页
RCON	IPEN	—	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	50
WDTCON	—	—	—	—	—	—	—	SWDTEN	50

图注： — = 未用，读为 0。看门狗定时器不使用阴影单元。

23.3 片内稳压器

所有的 PIC18F87J10 系列器件使用标称值为 2.5V 的电压为其内核数字逻辑供电。对于需要工作在更高电压（如典型电压值为 3.3V）下的应用，PIC18F87J10 系列的所有器件均包含一个片内稳压器，可使器件内核逻辑运行在 VDD 下。

ENVREG 引脚控制该稳压器。把 VDD 连到该引脚将使能稳压器，然后稳压后的电压通过其他 VDD 引脚向内核供电。当使能稳压器时，低 ESR 滤波器电容必须连接到 VDDCORE/VCAP 引脚（图 23-2），这有利于保持稳压器的稳定性。**第 26.3 节“直流规范：PIC18F87J10 系列（工业级）”**中提供了该滤波电容的推荐值。

如果 ENVREG 与 VSS 相连，则禁止稳压器。在这种情况下，独立的 2.5V 标称值的内核逻辑电压必须通过 VDDCORE/VCAP 引脚向器件供电，从而将 I/O 引脚驱动为一个较高的电平，通常为 3.3V。另外，VDDCORE/VCAP 和 VDD 引脚可以连在一起，使器件工作在较低的标称电压下。请参见图 23-2了解可能的配置。

23.3.1 片内稳压器和 BOR

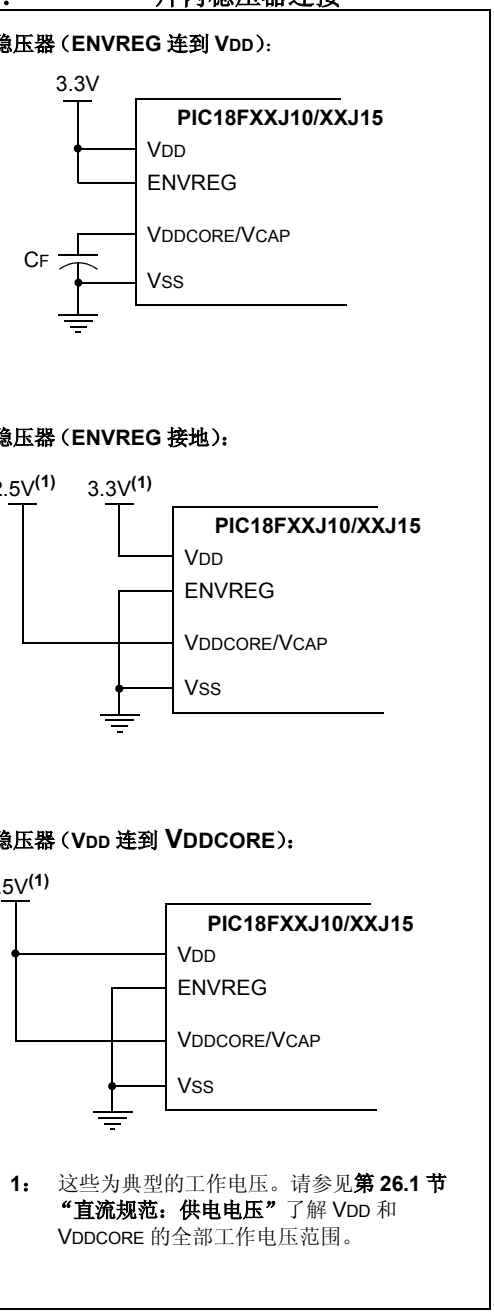
当使能片内稳压器时，PIC18F87J10 系列器件也会有一个简单的欠压保护功能。如果向稳压器提供的电压不足以维持一个稳定的电平，那么稳压器复位电路将产生 BOR 复位。 BOR 标志位 （RCON<0>）捕捉该事件。

第 4.4 节“欠压复位（BOR）”和**第 4.4.1 节“检测 BOR”**详细描述了 BOR 的原理。**第 26.1 节“直流规范：供电电压，PIC18F87J10 系列（工业级）”**中指定了欠压电压值。

23.3.2 上电要求

片内稳压器是为了满足器件的上电要求而设计的。如果应用不使用稳压器，那就必须严格遵守上电条件。在上电时， VDDCORE 决不能比 VDD 高出 0.3V 以上。

图 23-2： 片内稳压器连接



注 1： 这些为典型的工作电压。请参见**第 26.1 节“直流规范：供电电压”**了解 VDD 和 VDDCORE 的全部工作电压范围。

23.4 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振荡器作为时钟源，从而帮助器件最大限度地缩短从振荡器起振到代码执行之间的延时。通过将IESO配置位置1可使能该功能。

仅当主振荡器模式为HS或HSPLL（基于晶振的模式）时才可使用双速启动。由于EC和ECPLL模式不需要OST起振延时，因此应禁止双速启动。

当使能双速启动时，器件复位（在发生上电复位并且上电延时定时器发生超时后）和从休眠模式唤醒都会使器件将自身配置为使用内部振荡器电路作为时钟源。这样几乎可使代码在主振荡器起振、OST运行的同时立即执行。一旦OST超时，器件就自动切换到PRI_RUN模式。

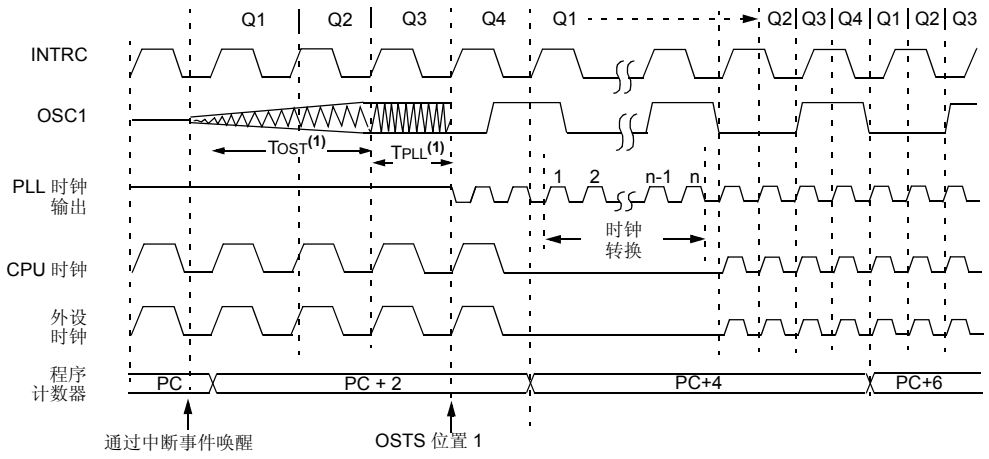
在其他功耗管理模式下不使用双速启动。器件将使用当前选定的时钟源直到主时钟源可用为止。该操作与IESO位的设置无关。

23.4.1 使用双速启动时需特别注意的事项

当在双速启动中使用INTRC振荡器时，器件仍将遵守进入功耗管理模式（包括执行SLEEP指令）的正常命令时序（见**第 3.1.4 节“多条 Sleep 命令”**）。实际上，这意味着在OST超时前用户代码可以改变SCS1:SCS0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时钟源前返回休眠状态。

用户代码还能通过检查OSTS位（OSCCON<3>）的状态来确定当前主时钟源是否正在为系统提供时钟。若该位置1，则表示主振荡器正在为系统提供时钟。否则，表示当器件从复位或休眠模式唤醒期间由内部振荡器电路为系统提供时钟。

图 23-3： 双速启动时钟转换的时序图（从 INTRC 切换到 HSPLL）



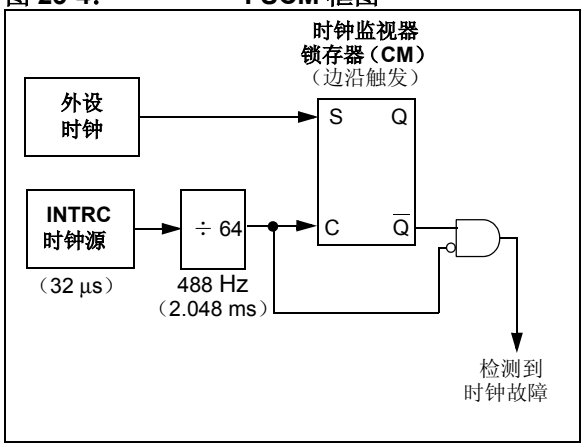
注 1： TOST = 1024 TOSC； TPLL = 2 ms（大约）。这些时间间隔未按比例显示。

23.5 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时钟故障时，自动将系统时钟切换到内部振荡器电路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， INTRC 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立即提供备用时钟。时钟监视（如图 23-4 所示）通过创建一个采样时钟信号实现，该信号为 INTRC 输出的 64 分频。这样就使得 FSCM 采样时钟沿之间有充足的时间间隔，从而保证在此时间段内必然会有外设时钟沿出现。外设器件时钟和采样时钟作为时钟监视器锁存器（CM）的输入。CM在器件时钟的下降沿被置1，在采样时钟的上升沿被清零。

图 23-4： FSCM 框图



在采样时钟的下降沿检测外部时钟故障。如果在出现采样时钟的下降沿时， CM 仍置 1，就表示检测到外部时钟故障 （图 23-5）。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，内部振荡器电路的后分频频率可能不够稳定。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的关闭。请参见**第 3.1.4 节 “多条 Sleep 命令”**和**第 23.4.1 节 “使用双速启动时需特别注意的事项”**了解更多详细信息。

FSCM 只能检测出主时钟源或辅助时钟源的故障。如果内部振荡器电路发生故障，将无法被检测到，当然也就不可能采取任何措施。

23.5.1 FSCM 和看门狗定时器

FSCM 和 WDT 均以 INTRC 振荡器作为时钟源。由于 WDT 使用独立的分频器和计数器，当使能 FSCM 时，禁止 WDT 不影响 INTRC 振荡器的运行。

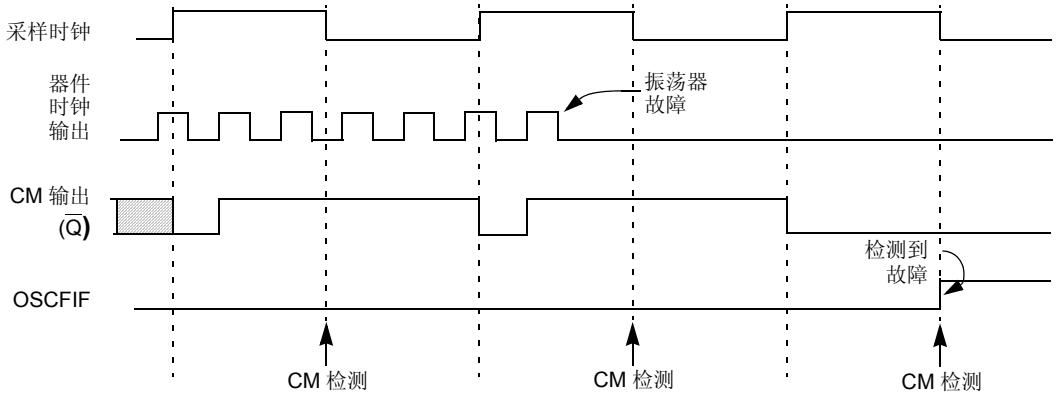
如前所述，当检测到时钟故障时，时钟源将切换到 INTRC 时钟源；这可能意味着代码执行速度会发生很大变化。如果用小分频值使能 WDT，时钟速率的下降可能导致 WDT 发生超时并在随后使器件复位。由于这个原因，故障保护事件也会使 WDT 和后分频器复位，从而使 WDT 从执行速度发生变化的时刻开始重新计数，因而减少了发生错误超时的可能性。

23.5.2 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 2L 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）。INTRC 振荡器将在主时钟源就绪前提供系统时钟 （类似于双速启动）。当主时钟源可用时，系统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。

在起振期间，主时钟源可能永远不能就绪。在这种情况下，器件运行将以INTRC振荡器作为时钟源。OSCCON寄存器将保持复位状态直到进入功耗模式为止。

图 23-5：　　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并不相符。

23.5.3　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监视。

如果在功耗管理运行期间发生了振荡器故障，随后将会发生的事件取决于是否允许了振荡器故障中断。如果允许中断（OSCFIF = 1），代码执行将以 INTRC 复用器作为时钟源，并且不会自动转回到发生故障的时钟源。

如果禁止该中断，休闲模式下故障所导致的中断将使 CPU 开始执行指令，此时由 INTRC 时钟源提供时钟。

23.5.4　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的任一时刻都可以检测到振荡器故障。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。

对于 HS 或 HSPLL 模式，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的时钟故障。为了避免这一情况，内部振荡器电路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 定时器已完成延时）。这与双速启动模式相同。一旦主时钟稳定下来， INTRC 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中断的电路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在振荡器故障中断标志位上有所反映。

正如**第 23.4.1 节 “使用双速启动时需特别注意的事项”**中所述，在等待主时钟稳定的过程中，可以选择另一种时钟配置并进入某一功耗管理模式。当选择新的功耗管理模式时，主时钟将被禁止。

23.6 程序校验和代码保护

对于 PIC18F87J10 系列中的所有器件，片内程序存储器被视为一个存储区。配置位 CP0 控制该存储区的代码保护。该位阻止外部对程序存储空间的读写。但对正常的代码执行没有直接影响。

23.6.1 配置寄存器保护

有两种方法保护配置寄存器使其免遭破坏性的改写或读取。主要的保护方式是配置位的一次写入功能，该功能阻止对在上电周期内完成编程的位再次进行配置。要阻止不可预见的事件，由于电池故障（如 ESD 事件）产生的配置位更改将导致奇偶校验错误并触发器件复位。

配置寄存器的数据来自于程序存储器中的闪存配置字。当 CP0 位置 1 时，也将保护器件配置的源数据。

23.7 在线串行编程

PIC18F87J10 系列单片机可以在最终应用电路中进行串行编程。只需要 5 根线即可完成这一操作，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。从而可使固件版本保持最新或定制固件。

23.8 在线调试器

将 DEBUG 配置位清 0 可使能在线调试功能。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 23-4 显示了后台调试器所需的资源。

表 23-4： 调试器资源

I/O 引脚：	RB6， RB7
堆栈：	2 级
程序存储器：	512 字节
数据存储器：	10 字节

24.0 指令集综述

PIC18F87J10 系列器件具有一个含有 75 个 PIC18 内核指令的标准指令集和一个含有优化递归代码或利用软件堆栈的 8 个新指令的扩展指令集。本章后面部分将讨论这一扩展的指令集。

24.1 标准指令集

标准的 PIC18C 指令集与以前的 PICmicro® 指令集相比，添加了很多增强功能，并保持了易于从其他 PICmicro 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），但有 4 个指令需要两个程序存储字。

每个单字指令都是一个 16 位字，包括指定指令类型的操作码和指定指令具体操作的一个或多个操作数。

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 24-2中的PIC18指令集汇总列出了**字节**操作类指令、**位**操作类指令、**立即数**操作类指令和**控制**操作类指令。表 24-1 给出了对操作码字段的说明。

大部分**字节**操作类的指令都含有三种操作数：

1. 数据寄存器 （由 “f”指定）
2. 保存结果的目标寄存器 （由 “d”指定）
3. 被访问的存储器 （由 “a”指定）

数据寄存器指示符 “f”指定指令将会使用哪一个数据寄存器。目标符 “d”指定操作结果的存放位置。如果 “d”为 0，操作结果存入 WREG 寄存器中。如果 “d”为 1，操作结果存入指令指定的数据寄存器中。

所有**位**操作类指令都含有三种操作数：

1. 数据寄存器 （由 “f”指定）
2. 数据寄存器中的位 （由 “b”指定）
3. 被访问的存储器 （由 “a”指定）

位域指示符 “b”指定操作所影响的位的编号，而数据寄存器指示符 “f”则代表这些位所在的寄存器编号。

立即数操作指令可以使用以下某些操作数：

- 要装入数据寄存器中的立即数值 （由 “k”指定）
- 希望装入立即数值的 FSR 寄存器 （由 “f”指定）
- 不要求操作数 （由 “－”指定）

控制操作指令可以使用以下某些操作数：

- 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）
- 不要求操作数 （由 “－”指定）

除了 4 个双字指令外，其他所有的指令都是单字指令。双字指令将所需的信息保存在 32 个位中。在第二个字中，高 4 位都是 1。如果第二个字本身作为一条指令执行，它就会作为 NOP 指令执行。

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周期。对于前面的两个特殊指令，执行指令需要两个指令周期，第二个周期中执行的是一条 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 个振荡器周期组成。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。如果条件测试结果为 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）的执行则需要 3 µs。

图 24-1 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 24-2）列出了可被 Microchip MPASM™ 汇编器识别的标准指令。

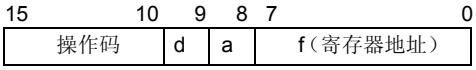
第 24.1.1 节 “标准指令集”中对每个指令进行了介绍。

表 24-1： 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：RAM 存储区由 BSR 寄存器指定
bbb	某 8 位数据寄存器内的位地址 （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 和 N	ALU 状态位：**C** 进位标志位、 **DC** 辅助进位标志位、 **Z** 全零标志位、 **Ov** 溢出标志位和 **N** 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：结果保存至数据寄存器 f。
dest	目标单元：可以是 WREG 寄存器或指定的数据寄存器单元。
f	8 位寄存器地址 （00h 到 FFh）或 2 位 FSR 指示符 （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、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。 只和表读和表写指令一起使用： 不改变寄存器 （如用于表读和表写的 TBLPTR）。 后增寄存器 （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	相关跳转指令的相对地址 （二进制补码）或 Call/Branch 和 Return 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其他寄存器 s = 1：将寄存器的值存入影子寄存器或把影子寄存器的值载入寄存器 （快速模式）
TBLPTR	21 位表指针 （指向程序存储器单元）。
TABLAT	8 位表锁存器。
$\overline{TO}$	超时溢出位。
TOS	栈顶。
u	未使用或未改变。
WDT	看门狗定时器。
WREG	工作寄存器 （累加器）。
x	忽略 （0 或 1）。汇编器将产生 x = 0 的代码。为了与所有的 Microchip 软件工具兼容，建议使用这种格式。
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>	指定由指针 expr 指定的寄存器中的位 n。
→	赋值给。
< >	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）。

图 24-1：　　指令的通用格式

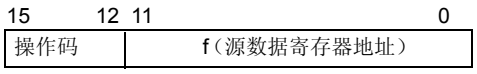
针对字节的数据寄存器操作　　**指令示例**



ADDWF MYREG, W, B

d = 0 表示结果存入 WREG 寄存器
d = 1 表示结果存入数据寄存器（f）
a = 0 强制使用快速操作存储区
a = 1 使用 BSR 选择存储区
f = 8 位数据寄存器地址

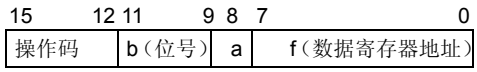
从字节到字节的移动操作（双字）



MOVFF MYREG1, MYREG2

f = 12 位数据寄存器地址

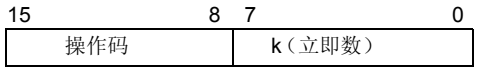
针对位的数据寄存器操作



BSF MYREG, bit, B

b = 占 3 位，表示数据寄存器（f）中位的位置
a = 0 强制使用快速操作存储区
a = 1 使用 BSR 选择存储区
f = 8 位数据寄存器地址

立即数操作

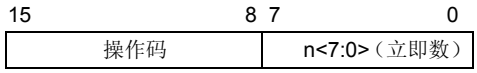


MOVLW 7Fh

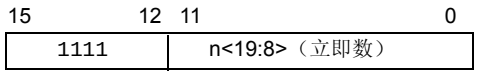
k = 8 位立即数的值

控制操作

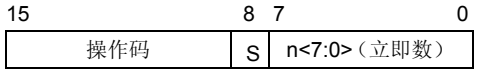
CALL、GOTO 和跳转类操作



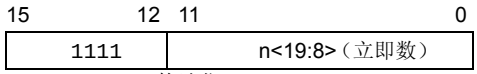
GOTO Label



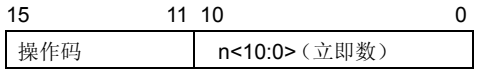
n = 20 位立即数的值



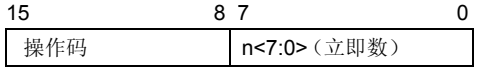
CALL MYFUNC



S = 快速位



BRA MYFUNC



BC MYFUNC

表 24-2： PIC18F87J10 系列指令集

助记符，操作数		说明	周期数	16 位宽指令字 MSb LSb	受影响的状态位	注
针对字节的操作类指令						
ADDWF	f, d, a	WREG 与 f 相加	1	0010 01da ffff ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	WREG 与 f 带进位相加	1	0010 00da ffff ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	WREG 与 f 作逻辑与运算	1	0001 01da ffff ffff	Z, N	1,2
CLRF	f, a	f 清零	1	0110 101a ffff ffff	Z	2
COMF	f, d, a	f 取反	1	0001 11da ffff ffff	Z, N	1, 2
CPFSEQ	f, a	将 f 与 WREG 做比较，相等则跳过	1（2 或 3）	0110 001a ffff ffff	无	4
CPFSGT	f, a	将 f 与 WREG 做比较，大于则跳过	1（2 或 3）	0110 010a ffff ffff	无	4
CPFSLT	f, a	将 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 和 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	将 f_s（源）地址放入第 1 个字	2	1100 ffff ffff ffff	无	
	f, a	将 f_d（目标）地址放入第 2 个字		1111 ffff ffff ffff		
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, d, a	将 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, a	f 循环右移（不带进位）	1	0100 00da ffff ffff	Z, N	
SETF	f, d, 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	从 f 减去 WREG	1	0101 11da ffff ffff	C, DC, Z, OV, N	1, 2
SUBWFB	f, d, a	从 f 减去 WREG（带借位）	1	0101 10da ffff ffff	C, DC, Z, OV, N	
SWAPF	f, a	将 f 中的两个半字节进行交换	1	0011 10da ffff ffff	无	4
TSTFSZ	f, d, a	测试 f，为 0 则跳过	1（2 或 3）	0110 011a ffff ffff	无	1, 2
XORWF		WREG 和 f 作逻辑异或运算	1	0001 10da ffff ffff	Z, N	

注 1： 端口寄存器的值随端口状态的变化而不断修改（例如，`MOVF PORTB, 1, 0`），它使用引脚上的当前值更新自身内容。例如，如果将一引脚配置为输入，其对应数据锁存器中的值也将为 1，但此时若有外部器件将该引脚拉为低电平，则被写回数据锁存器的数据值将是 0。

2： 当该指令针对 TMR0 寄存器执行时（指令中的 d = 1），如果已对预分频器进行了赋值，则将其清零。

3： 如果执行一条程序计数器（PC）被修改或者条件检测为 true 的指令，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。如果指令的第一个字无法取得后面 16 位中嵌入的信息，则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

表 24-2： PIC18F87J10 系列指令集 （续）

助记符，操作数		说明	周期数	16 位宽指令字 MSb LSb	受影响的状态位	注
针对位的操作类指令						
BCF	f, b, a	将 f 中的某位清零	1	1001 bbba ffff ffff	无	1, 2
BSF	f, b, a	将 f 中的某位置 1	1	1000 bbba ffff ffff	无	1, 2
BTFSC	f, b, a	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, b, 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, b, 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	非零则跳转	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	—	看门狗定时器清零	1	0000 0000 0000 0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	对 WREG 进行十进制调整	1	0000 0000 0000 0111	C	
GOTO	n	跳转到地址 （第一个字）	2	1110 1111 kkkk kkkk	无	
		（第二个字）		1111 kkkk kkkk kkkk		
NOP	—	空操作	1	0000 0000 0000 0000	无	
NOP	—	空操作	1	1111 xxxx xxxx xxxx	无	4
POP	—	从返回堆栈栈顶 （TOS）出栈	1	0000 0000 0000 0110	无	
PUSH	—	从返回堆栈栈顶 （TOS）进栈	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	中断返回使能	2	0000 0000 0001 000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	返回时将立即数送入 WREG	2	0000 1100 kkkk kkkk	无	
RETURN	s	从子程序返回	2	0000 0000 0001 001s	无	
SLEEP	—	进入待机模式	1	0000 0000 0000 0011	$\overline{TO}$, $\overline{PD}$	

注 1： 端口寄存器的值随端口状态的变化而不断修改 （例如， MOVF PORTB, 1, 0），它使用引脚上的当前值更新自身内容。例如，如果将一引脚配置为输入，其对应数据锁存器中的值也将为 1，但此时若有外部器件将该引脚拉为低电平，则被写回数据锁存器的数据值将是 0。

2： 当该指令针对 TMR0 寄存器执行时 （指令中的 d = 1），如果已对预分频器进行了赋值，则将其清零。

3： 如果执行一条程序计数器 （PC）被修改或者条件检测为 true 的指令，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。如果指令的第一个字无法取得后面 16 位中嵌入的信息，则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

表 24-2： PIC18F87J10 系列指令集 （续）

助记符，操作数		说明	周期数	16 位宽指令字 MSb LSb	受影响的状态位	注
立即数操作类指令						
ADDLW	k	WREG 与立即数相加	1	0000 1111 kkkk kkkk	C, DC, Z, OV, N	
ANDLW	k	WREG 和立即数作逻辑与运算	1	0000 1011 kkkk kkkk	Z, N	
IORLW	k	WREG 和立即数作逻辑或运算	1	0000 1001 kkkk kkkk	Z, N	
LFSR	f, k	将立即数（12 位）第二个字移入 FSR （f）的第一个字	2	1110 1110 00ff kkkk 1111 0000 kkkk kkkk	无	
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	WREG 和立即数相乘	1	0000 1101 kkkk kkkk	无	
RETLW	k	返回时将立即数送入 WREG	2	0000 1100 kkkk kkkk	无	
SUBLW	k	立即数减去 WREG	1	0000 1000 kkkk kkkk	C, DC, Z, OV, N	
XORLW	k	WREG 和立即数作逻辑异或运算	1	0000 1010 kkkk kkkk	Z, N	
数据存储器 ↔ 程序存储器操作						
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： 端口寄存器的值随端口状态的变化而不断修改 （例如， MOVF PORTB, 1, 0），它使用引脚上的当前值更新自身内容。例如，如果将一引脚配置为输入，其对应数据锁存器中的值也将为 1，但此时若有外部器件将该引脚拉为低电平，则被写回数据锁存器的数据值将是 0。

2： 当该指令针对 TMR0 寄存器执行时 （指令中的 d = 1），如果已对预分频器进行了赋值，则将其清零。

3： 如果执行一条程序计数器 （PC）被修改或者条件检测为 true 的指令，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。如果指令的第一个字无法取得后面 16 位中嵌入的信息，则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

24.1.1 标准指令集

ADDLW **将 W 的内容与立即数相加**

语法： ADDLW k

操作数： 0 ≤ k ≤ 255

操作： (W) + k → W

受影响的状态位： N、 OV、 C、 DC 和 Z

机器码：

0000	1111	kkkk	kkkk

说明： 将 W 寄存器的内容与 8 位立即数 “k” 相加，结果存入 W 寄存器。

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

指令执行前
W = 10h
指令执行后
W = 25h

ADDWF **将 W 与 f 相加**

语法： ADDWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) + (f) → dest

受影响的状态位： N、OV、C、DC 和 Z

机器码：

0010	01da	ffff	ffff

说明： 将 W 寄存器的内容与 “f” 寄存器的内容相加。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： ADDWF REG, 0, 0

指令执行前
W = 17h
REG = 0C2h
指令执行后
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，以在符号寻址中使用。如果使用了标号，那么指令语法将变为：{label} 指令参数。

ADDWFC **将 W 与 f 带进位相加**

语法： ADDWFC f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) + (f) + (C) → dest

受影响的状态位： N、 OV、C、DC 和 Z

机器码：

0010	00da	ffff	ffff

说明： 将 W 的内容、进位标志位与数据存储单元 “f” 的内容相加。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存储在数据存储单元 “f” 中。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： ADDWFC REG, 0, 1

指令执行前
进位标志位 = 1
REG = 02h
W = 4Dh

指令执行后
进位标志位 = 0
REG = 02h
W = 50h

ANDLW **将立即数和 W 寄存器作逻辑与运算**

语法： ANDLW k

操作数： 0 ≤ k ≤ 255

操作： (W) .AND. k → W

受影响的状态位： N 和 Z

机器码：

0000	1011	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 “k” 进行逻辑与运算。结果保存在 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

指令执行前
W = A3h

指令执行后
W = 03h

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” 的内容进行逻辑与运算。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写入目标寄存器

示例： ANDWF REG, 0, 0

指令执行前
W = 17h
REG = C2h
指令执行后
W = 02h
REG = C2h

BC **进位则跳转**

语法： BC n

操作数： -128 ≤ n ≤ 127

操作： 如果进位标志位为 1
(PC)+2+2n → PC

受影响的状态位： 无

机器码：

1110	0010	nnnn	nnnn

说明： 如果进位标志位为 1，程序将跳转。

二进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。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

指令执行前
PC = 地址 （HERE）
指令执行后
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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写寄存器 “f”

示例： BCF FLAG_REG, 7, 0

指令执行前
FLAG_REG = C7h
指令执行后
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。

2 进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。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

指令执行前
PC = 地址 （HERE）
指令执行后
如果负标志位 = 1；
PC = 地址 （Jump）
如果负标志位 = 0；
PC = 地址 （HERE + 2）

BNC **无进位则跳转**

语法： BNC n

操作数： -128 ≤ n ≤ 127

操作： 如果进位标志位为 0
(PC)+2+2n → PC

受影响的状态位： 无

机器码：

1110	0011	nnnn	nnnn

说明： 如果进位标志位为 0，那么程序将跳转。

2 进制补码 “2n”与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。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

指令执行前
PC = 地址 （HERE）
指令执行后
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。

2 进制补码 “2n”与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。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

指令执行前
PC = 地址 （HERE）
指令执行后
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。

2 进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。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

指令执行前
PC = 地址 （HERE）
指令执行后
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 **非零则跳转**

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。

2 进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。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

指令执行前
PC = 地址 （HERE）
指令执行后
如果全零标志位 = 0；
PC = 地址 （Jump）
如果全零标志位 = 1；
PC = 地址 （HERE + 2）

BRA **无条件跳转**

语法： BRA n

操作数： -1024 ≤ n ≤ 1023

操作： (PC)+2+2n → PC

受影响的状态位： 无

机器码：

1101	0nnn	nnnn	nnnn

说明： 2 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

指令执行前
PC = 地址 （HERE）
指令执行后
PC = 地址 （Jump）

BSF **将 f 中的某位置 1**

语法： BSF f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

操作： 1 → f<b>

受影响的状态位： 无

机器码：

1000	bbba	ffff	ffff

说明： 将寄存器 “f” 的位 “b” 置 1。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写寄存器 “f”

示例： BSF FLAG_REG, 7, 1

指令执行前
FLAG_REG = 0Ah
指令执行后
FLAG_REG = 8Ah

BTFSC **测试寄存器中的位，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 则跳过

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，丢弃在当前指令执行期间所取的下一条指令，而是执行一条 NOP 指令取而代之，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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

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

如果被跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

指令执行前
PC = 地址 （HERE）
指令执行后
如果 FLAG<1> = 0；
PC = 地址 （TRUE）
如果 FLAG<1> = 1；
PC = 地址 （FALSE）

BTFSS **测试寄存器中的位，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 则跳过

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，丢弃在当前指令执行期间所取的下一条指令，而是执行一条 NOP 指令取而代之，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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

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

如果被跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：
```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

指令执行前
PC = 地址 （HERE）
指令执行后
如果 FLAG<1> = 0；
PC = 地址 （FALSE）
如果 FLAG<1> = 1；
PC = 地址 （TRUE）

BTG **将 f 中的某位取反**

语法： BTG f, b {,a}

操作数： 0 ≤ f ≤ 255
0 ≤ b<7
a ∈ [0,1]

操作： $(\overline{f<b>}) \rightarrow f<b>$

受影响的状态位： 无

机器码：

0111	bbba	ffff	ffff

说明： 将数据存储单元 “f” 中的位 b 取反。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写寄存器 “f”

示例： BTG PORTC, 4, 0

指令执行前
PORTC = 0111 0101 [75h]
指令执行后
PORTC = 0110 0101[65h]

BOV **溢出则跳转**

语法： BOV n

操作数： -128 ≤ n ≤ 127

操作： 如果溢出标志位为 1
(PC)+2+2n → PC

受影响的状态位： 无

机器码：

1110	0100	nnnn	nnnn

说明： 如果溢出标志位为 1，那么程序将跳转。

2 进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。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

指令执行前
PC = 地址 （HERE）
指令执行后
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。

2 进制补码 “2n” 与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。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

指令执行前
PC = 地址 （HERE）
指令执行后
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 寄存器) → STATUSS，
(BSR) → BSRS

受影响的状态位： 无

机器码：
第 1 个字 （k<7:0>）
第 2 个字 （k<19:8>）

1110 1111	110s k_{19}kkk	$k_7$kkk kkkk	kkkk_0 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 寄存器也压入它们各自的影子寄存器 WS、 STATUSS 和 BSRS。如果 s = 0，将不会进行任何更新 （默认）。然后，将 20 位的值 “k” 装入 PC<20:1>。 CALL 是一条双周期指令。

指令字数： 2

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 “k” <7:0>，	将 PC 压入堆栈	读立即数 “k” <19:8>，写入 PC
空操作	空操作	空操作	空操作

示例： HERE CALL THERE,1

指令执行前
PC = 地址 （HERE）
指令执行后
PC = 地址 （THERE）
TOS = 地址 （HERE + 4）
WS = W
BSRS = BSR
STATUSS = STATUS

CLRF **将 f 清零**

语法： CLRF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： 000h → f
1 → Z

受影响的状态位： Z

机器码：

0110	101a	ffff	ffff

说明： 清零指定寄存器的内容。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写寄存器 “f”

示例： CLRF FLAG_REG,1

指令执行前
FLAG_REG = 5Ah
指令执行后
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 复位看门狗定时器及其后分频器。状态位 $\overline{TO}$ 和 $\overline{PD}$ 置 1。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	空操作

示例： CLRWDT

指令执行前
WDT 计数器 = ?
指令执行后
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 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： $(\bar{f}) \rightarrow$ dest

受影响的状态位： N 和 Z

机器码：

0001	11da	ffff	ffff

说明： 寄存器 “f” 的内容取反。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： COMF REG, 0, 0

指令执行前
REG = 13h
指令执行后
REG = 13h
W = ECh

CPFSEQ **比较 f 和 W，如果 f = W 则跳过**

语法： CPFSEQ f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (f) – (W)，
如果 (f) = (W) 则跳过
（无符号比较）

受影响的状态位： 无

机器码：

0110	001a	ffff	ffff

说明： 通过执行无符号减法，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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

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

如果被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   :
```

指令执行前
PC 地址 = HERE
W = ?
REG = ?
指令执行后
如果 REG = W；
PC = 地址 （EQUAL）
如果 REG ≠ W；
PC = 地址 （NEQUAL）

CPFSGT **比较 f 和 W，如果 f > W 则跳过**

语法： CPFSGT f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (f) – (W)，
如果 (f) > (W) 则跳过
（无符号比较）

受影响的状态位： 无

机器码：

0110	010a	ffff	ffff

说明： 通过执行无符号减法，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（默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节“立即数变址寻址模式中针对字节和位的指令”**。

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

如果被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   :
```

指令执行前
PC = 地址 （HERE）
W = ?
指令执行后
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，则所取的指令被丢弃，转而执行一条 NOP 指令，从而使该指令成为一条双周期指令。

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（默认）。

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

如果被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    :
```

指令执行前
PC = 地址 （HERE）
W = ?
指令执行后
如果 REG < W；
PC = 地址 （LESS）
如果 REG ≥ W；
PC = 地址 （NLESS）

DAW **对 W 寄存器进行十进制调整**

语法： DAW

操作数： 无

操作： 如果 [W<3:0> > 9] 或 [DC = 1]，则
(W<3:0>) + 6 → W<3:0>;
否则
(W<3:0>) → W<3:0>

如果 [W<7:4> > 9] 或 [C = 1]，则
(W<7:4>) + 6 → W<7:4>;
C = 1；
否则
(W<7:4>) → W<7:4>

受影响的状态位： C

机器码：

0000	0000	0000	0111

说明： DAW 调整 W 内的 8 位值，即前一条加法指令中两个变量（均为压缩的 BCD 格式）的和，并产生一个正确的压缩 BCD 格式的结果。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 W	处理数据	写 W

示例 1： DAW

指令执行前
W = A5h
C = 0
DC = 0
指令执行后
W = 05h
C = 1
DC = 0

示例 2：

指令执行前
W = CEh
C = 0
DC = 0
指令执行后
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、 Z、 OV 和 N

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”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： DECF CNT, 1, 0

指令执行前
CNT = 01h
Z = 0
指令执行后
CNT = 00h
Z = 1

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
结果为 0 时跳过

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”（默认）。

如果结果为 0，则丢弃已取的指令，而是执行一条 NOP 指令取而代之，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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

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

如果被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
```

指令执行前
PC = 地址 （HERE）
指令执行后
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，
结果≠ 0 时跳过

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”（默认）。

如果结果不为 0，则丢弃已取的指令，转而执行一条 NOP 指令，从而使该指令成为双周期指令。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

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

如果被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   :
```

指令执行前
TEMP = ?
指令执行后
TEMP = TEMP – 1，
如果 TEMP = 0；
PC = 地址 （ZERO）
如果 TEMP ≠ 0；
PC = 地址 （NZERO）

GOTO **无条件跳转**

语法： GOTO k

操作数： 0 ≤ k ≤ 1048575

操作： k → PC<20:1>

受影响的状态位： 无

机器码：
第 1 个字 （k<7:0>）
第 2 个字 （k<19:8>）

1110 1111	1111 k_{19}kkk	$k_7$kkk kkkk	kkkk$_0$ kkkk$_8$

说明： GOTO 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 位的值 “k” 装入 PC<20:1>。 GOTO 始终为一条双周期指令。

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>	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

指令执行后
PC = 地址 （THERE）

INCF **f 加 1**

语法： INCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f) + 1 → dest

受影响的状态位： C、 DC、 N、 OV 和 Z

机器码：

0010	10da	ffff	ffff

说明： 将寄存器 “f” 的内容加 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”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： INCF CNT, 1, 0

指令执行前
CNT = FFh
Z = 0
C = ?
DC = ?
指令执行后
CNT = 00h
Z = 1
C = 1
DC = 1

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 则跳过

受影响的状态位： 无

机器码：

0011	11da	ffff	ffff

说明： 将寄存器 “f” 的内容加 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”（默认）。

如果结果为 0，则丢弃已取的指令，而是执行一条 NOP 指令取而代之，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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

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

如果被跳过的指令后面跟有 2 字指令：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例：

```
HERE    INCFSZ   CNT, 1, 0
NZERO   :
ZERO    :
```

指令执行前
PC = 地址 （HERE）
指令执行后
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，
结果≠ 0 时跳过

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”（默认）。

如果结果不为 0，则丢弃已取的指令，转而执行一条 NOP 指令，从而使该指令成为双周期指令。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

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

如果被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
```

指令执行前
PC = 地址 （HERE）
指令执行后
REG = REG + 1
如果 REG ≠ 0；
PC = 地址 （NZERO）
如果 REG = 0；
PC = 地址 （ZERO）

IORLW **将立即数与 W 作逻辑或运算**

语法： IORLW k

操作数： 0 ≤ k ≤ 255

操作： (W) .OR. k → W

受影响的状态位： N 和 Z

机器码：

0000	1001	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 “k” 进行逻辑或运算。结果保存在 W 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

指令执行前
W = 9Ah
指令执行后
W = BFh

IORWF **将 W 与 f 作逻辑或运算**

语法： IORWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) .OR.(f) → dest

受影响的状态位： N 和 Z

机器码：

0001	00da	ffff	ffff

说明： 将 W 与寄存器 “f” 进行逻辑或运算。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区（默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95（5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： IORWF RESULT, 0, 1

指令执行前
RESULT = 13h
W = 91h
指令执行后
RESULT = 13h
W = 93h

LFSR **载入 FSR**

语法： LFSR f, k

操作数： 0 ≤ f ≤ 2
0 ≤ k ≤ 4095

操作： k → FSRf

受影响的状态位： 无

机器码：

1110 1111	1110 0000	00ff $k_7$kkk	k_{11}kkk kkkk

说明： 将 12 位立即数 “k” 装入由 “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 MSB	处理数据	写立即数 “k” 的 MSB 到 FSRfH
译码	读立即数 “k” 的 LSB	处理数据	将立即数 “k” 写入 FSRfL

示例： LFSR 2, 3ABh

指令执行后
FSR2H = 03h
FSR2L = ABh

MOVF **移动 f**

语法： MOVF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： f → dest

受影响的状态位： N 和 Z

机器码：

0101	00da	ffff	ffff

说明： 根据 “d” 的状态，将寄存器 “f” 的内容移入目标单元。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”（默认）。“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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写 W

示例： MOVF REG, 0, 0

指令执行前
REG = 22h
W = FFh
指令执行后
REG = 22h
W = 22h

MOVFF **将源单元的内容移动至目标单元**

语法： MOVFF f_s,f_d

操作数： $0 \leqslant f_s \leqslant 4095$
$0 \leqslant f_d \leqslant 4095$

操作： $(f_s) \rightarrow f_d$

受影响的状态位： 无

机器码：

第 1 个字 （源）	1100	ffff	ffff	$ffff_s$
第 2 个字 （目标）	1111	ffff	ffff	$ffff_d$

说明： 将源寄存器 “f_s” 的内容移入目标寄存器 “f_d”。
源存储单元 “f_s” 可以是 4096 字节数据空间 （000h 到 FFFh）中的任何存储单元，目标存储单元 “f_d” 也可以是 000h 到 FFFh 中的任何存储单元。

源或目标都可以是 W （这是个有用的特例）。

MOVFF 尤其适合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）。

MOVFF 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）	处理数据	空 操作
译码	空 操作 非虚拟读取	空 操作	写 寄存器 “f” （目标）

示例：

```
MOVFF REG1, REG2
```

指令执行前
REG1 = 33h
REG2 = 11h
指令执行后
REG1 = 33h
REG2 = 33h

MOVLB **将立即数移入 BSR 的低半字节**

语法： MOVLW k

操作数： $0 \leqslant k \leqslant 255$

操作： k → BSR

受影响的状态位： 无

机器码：

0000	0001	kkkk	kkkk

说明： 将 8 位立即数 “k” 装入存储区选择寄存器 （BSR）。不管 k_7:k_4 的值如何，BSR<7:4> 的值将始终保持为 0。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 “k”	处理数据	将立即数 “k” 写入 BSR

示例：

```
MOVLB 5
```

指令执行前
BSR 寄存器 = 02h
指令执行后
BSR 寄存器 = 05h

MOVLW **将立即数移入 W**

语法： MOVLW k

操作数： 0 ≤ k ≤ 255

操作： k → W

受影响的状态位： 无

机器码：

0000	1110	kkkk	kkkk

描述： 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

指令执行后
W = 5Ah

MOVWF **将 W 的内容移入 f**

语法： MOVWF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (W) → f

受影响的状态位： 无

机器码：

0110	111a	ffff	ffff

说明： 将 W 寄存器中的数据移入寄存器 “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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写 寄存器 “f”

示例： MOVWF REG, 0

指令执行前
W = 4Fh
REG = FFh
指令执行后
W = 4Fh
REG = 4Fh

MULLW **将立即数与 W 中的内容相乘**

语法： MULLW k

操作数： 0 ≤ k ≤ 255

操作： (W) x k → PRODH:PRODL

受影响的状态位： 无

机器码：

0000	1101	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 “k” 执行无符号的乘法运算。 16 位的结果存储在 PRODH:PRODL 寄存器对中。 PRODH 存放高字节。

W 不改变。

所有状态标志位都不受影响。

请注意此操作不可能发生溢出或进位。结果有可能为零，但不会在标志位上反映出来。

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

指令执行前
W = E2h
PRODH = ?
PRODL = ?

指令执行后
W = E2h
PRODH = ADh
PRODL = 08h

MULWF **将 W 与 f 相乘**

语法： MULWF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： (W) x (f) → PRODH:PRODL

受影响的状态位： 无

机器码：

0000	001a	ffff	ffff

说明： 将 W 的内容与寄存器单元 “f” 的内容执行无符号的乘法运算。 16 位的结果存储在 PRODH:PRODL 寄存器对中。 PRODH 存放高字节。 W 和 “f” 都不改变。

所有状态标志位都不受影响。

请注意此操作不可能发生溢出或进位。结果有可能为零，但不会在标志位上反映出来。

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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写 寄存器 PRODH: PRODL

示例： MULWF REG, 1

指令执行前
W = C4h
REG = B5h
PRODH = ?
PRODL = ?

指令执行后
W = C4h
REG = B5h
PRODH = 8Ah
PRODL = 94h

NEGF **将 f 取补**

语法： NEGF f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： ($\bar{f}$) + 1 → f

受影响的状态位： N、OV、C、DC 和 Z

机器码：

0110	110a	ffff	ffff

说明： 对地址单元 “f” 的内容取 2 进制补码。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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写寄存器 “f”

示例：

```
NEGF    REG, 1
```

指令执行前
REG = 0011 1010 [3Ah]
指令执行后
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： 不进行任何操作。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作

示例：

无

POP **弹出返回堆栈栈顶的内容**

语法： POP

操作数： 无

操作： (TOS) → 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以实现软件堆栈。

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
```

指令执行前
TOS = 0031A2h
堆栈 （向下 1 级） = 014332h

指令执行后
TOS = 014332h
PC = NEW

PUSH **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
```

指令执行前
TOS = 345Ah
PC = 0124h

指令执行后
PC = 0126h
TOS = 0126h
堆栈 （向下 1 级） = 345Ah

RCALL **相对调用**

语法： RCALL n

操作数： -1024 ≤ n ≤ 1023

操作： (PC) + 2 → TOS，
(PC) + 2 + 2n → PC

受影响的状态位： 无

机器码：

1101	1nnn	nnnn	nnnn

说明： 从当前地址跳转（最多 1K）来执行子程序调用。首先，将返回地址（PC+2）压入返回堆栈。然后，将 2 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

指令执行前
PC = 地址 （HERE）
指令执行后
PC = 地址 （Jump）
TOS = 地址 （HERE + 2）

RESET **复位**

语法： RESET

操作数： 无

操作： 将所有受 $\overline{\text{MCLR}}$ 复位影响的寄存器和标志位复位。

受影响的状态位： 全部

机器码：

0000	0000	1111	1111

说明： 该指令可用于执行软件复位。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	开始复位	空操作	空操作

示例： RESET

指令执行后
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 寄存器，
(BSRS) → BSR，
PCLATU 和 PCLATH 不变

受影响的状态位： GIE/GIEH 和 PEIE/GIEL。

机器码：

0000	0000	0001	000s

说明： 从中断返回。执行出栈操作，将栈顶（TOS）的内容装入 PC。通过将高或低优先级全局中断允许位置 1 来允许中断。如果 “s” = 1，则影子寄存器 WS、STATUSS 和 BSRS 的内容将被装入对应的寄存器 W、 STATUS 和 BSR。如果 “s” = 0，则不更新这些寄存器（默认）。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空 操作	空 操作	从堆栈弹出 PC 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 寄存器 = STATUSS
GIE/GIEH、PEIE/GIEL = 1

RETLW **返回并把立即数装入 W**

语法： RETLW k

操作数： 0 ≤ k ≤ 255

操作： k → W，
(TOS) → PC，
PCLATU 和 PCLATH 不变

受影响的状态位： 无

机器码：

0000	1100	kkkk	kkkk

说明： 将 8 位立即数 “k” 装入 W。将栈顶内容（返回地址）装入程序计数器。高字节地址锁存器（PCLATH）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，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

    指令执行前
        W      =   07h
    指令执行后
        W      =   kn 的值
```

RETURN **从子程序返回**

语法： RETURN {s}

操作数： s ∈ [0,1]

操作： (TOS) → PC，
如果 s = 1
(WS) → W，
(STATUSS) → STATUS 寄存器，
(BSRS) → BSR，
PCLATU 和 PCLATH 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）内容装入程序计数器。如果“s” = 1，将影子寄存器 WS、STATUSS 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 （默认）。

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	从堆栈弹出 PC
空操作	空操作	空操作	空操作

示例： RETURN

指令执行后
PC = TOS

RLCF **对 f 执行带进位的循环左移**

语法： RLCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f<n>) → dest<n+1>，
(f<7>) → C，
(C) → dest<0>

受影响的状态位： C、 N 和 Z

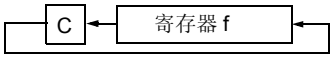
机器码：

0011	01da	ffff	ffff

说明： 将寄存器 “f” 的内容连同进位标志位一起循环左移 1 位。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。



指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写入目标寄存器

示例：
```
RLCF    REG, 0, 0
```
指令执行前
```
REG  =  1110 0110
C    =  0
```
指令执行后
```
REG  =  1110 0110
W    =  1100 1100
C    =  1
```

RLNCF **将 f 循环左移 （不带进位）**

语法： RLNCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f<n>) → dest<n+1>,
(f<7>) → dest<0>

受影响的状态位： N 和 Z

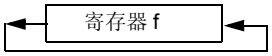
机器码：

0100	01da	ffff	ffff

说明： 将寄存器 “f” 的内容循环左移 1 位。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。



指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： RLNCF REG, 1, 0

指令执行前
REG = 1010 1011
指令执行后
REG = 0101 0111

RRCF **对 f 执行带进位的循环右移**

语法： RRCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f<n>) → dest<n–1>,
(f<0>) → C,
(C) → dest<7>

受影响的状态位： C、 N 和 Z

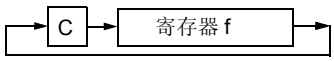
机器码：

0011	00da	ffff	ffff

说明： 将寄存器 “f” 的内容连同进位标志位一起循环右移 1 位。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。



指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： RRCF REG, 0, 0

指令执行前
REG = 1110 0110
C = 0
指令执行后
REG = 1110 0110
W = 0111 0011
C = 0

RRNCF **将 f 循环右移 （不带进位）**

语法： RRNCF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (f<n>) → dest<n–1>,
(f<0>) → dest<7>

受影响的状态位： N 和 Z

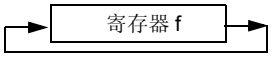
机器码：

0100	00da	ffff	ffff

说明： 将寄存器 “f” 的内容循环右移 1 位。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区，忽略 BSR 的值。如果 “a” 为 1，则会根据 BSR 的值选择存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。



指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例 1：
```
RRNCF   REG, 1, 0
```
指令执行前
REG = 1101 0111
指令执行后
REG = 1110 1011

示例 2：
```
RRNCF   REG, 0, 0
```
指令执行前
W = ?
REG = 1101 0111
指令执行后
W = 1110 1011
REG = 1101 0111

SETF **将 f 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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写 寄存器 “f”

示例：
```
SETF    REG,1
```
指令执行前
REG = 5Ah
指令执行后
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

指令执行前
$\overline{TO}$ = ?
$\overline{PD}$ = ?

指令执行后
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† 如果由 WDT 引起唤醒，则此位将被清零。

SUBFWB **W 减 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： 用 W 的内容减去寄存器 “f” 的内容和进位标志位（借位）（通过 2 进制补码方式实现）。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写入目标寄存器

示例 1：
```
SUBFWB   REG, 1, 0
```
指令执行前
REG = 3
W = 2
C = 1

指令执行后
REG = FF
W = 2
C = 0
Z = 0
N = 1 ；结果为负

示例 2：
```
SUBFWB   REG, 0, 0
```
指令执行前
REG = 2
W = 5
C = 1

指令执行后
REG = 2
W = 3
C = 1
Z = 0
N = 0 ；结果为正

示例 3：
```
SUBFWB   REG, 1, 0
```
指令执行前
REG = 1
W = 2
C = 0

指令执行后
REG = 0
W = 2
C = 1
Z = 1 ；结果为 0
N = 0

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 的内容，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

指令执行前
W = 01h
C = ?
指令执行后
W = 01h
C = 1 ; 结果为正
Z = 0
N = 0

示例 2： SUBLW 02h

指令执行前
W = 02h
C = ?
指令执行后
W = 00h
C = 1 ; 结果为 0
Z = 1
N = 0

示例 3： SUBLW 02h

指令执行前
W = 03h
C = ?
指令执行后
W = FFh ; （2 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 2 进制补码方式实现）。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例 1： SUBWF REG, 1, 0

指令执行前
REG = 3
W = 2
C = ?
指令执行后
REG = 1
W = 2
C = 1 ; 结果为正
Z = 0
N = 0

示例 2： SUBWF REG, 0, 0

指令执行前
REG = 2
W = 2
C = ?
指令执行后
REG = 2
W = 0
C = 1 ; 结果为 0
Z = 1
N = 0

示例 3： SUBWF REG, 1, 0

指令执行前
REG = 1
W = 2
C = ?
指令执行后
REG = FFh ; （2 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： 用寄存器 “f” 的内容减去 W 的内容和进位标志位 （借位）（通过 2 进制补码方式实现）。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写入目标寄存器

示例 1：
```
SUBWFB  REG, 1, 0
```
指令执行前
```
REG  =  19h   (0001 1001)
W    =  0Dh   (0000 1101)
C    =  1
```
指令执行后
```
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
```
指令执行前
```
REG  =  1Bh   (0001 1011)
W    =  1Ah   (0001 1010)
C    =  0
```
指令执行后
```
REG  =  1Bh   (0001 1011)
W    =  00h
C    =  1
Z    =  1     ; 结果为 0
N    =  0
```
示例 3：
```
SUBWFB  REG, 1, 0
```
指令执行前
```
REG  =  03h   (0000 0011)
W    =  0Eh   (0000 1101)
C    =  1
```
指令执行后
```
REG  =  F5h   (1111 0100)
                ; [2 进制补码 ]
W    =  0Eh   (0000 1101)
C    =  0
Z    =  0
N    =  1     ; 结果为负
```

SWAPF **交换 f 的高、低半字节**

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”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写入目标寄存器

示例：
```
SWAPF   REG, 1, 0
```
指令执行前
```
REG  =  53h
```
指令执行后
```
REG  =  35h
```

TBLRD **表读**

语法： TBLRD (*; *+; *-; +*)

操作数： 无

操作： 如果执行 TBLRD *，
(程序存储器 (TBLPTR)) → TABLAT ；
TBLPTR 不改变。
如果执行 TBLRD *+，
(程序存储器 (TBLPTR)) → TABLAT ；
(BLPTR) + 1 → TBLPTR。
如果执行 TBLRD *-，
(程序存储器 (TBLPTR)) → TABLAT ；
(TBLPTR)–1 → TBLPTR。
如果执行 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： 该指令用于读取程序存储器的内容。使用称为表指针 （TBLPTR）的指针对程序存储器进行寻址。

TBLPTR （一个 21 位指针）指向程序存储器的每个字节。 TBLPTR 的寻址范围为 2 MB。

TBLPTR[0] = 0：程序存储器字的最低有效字节

TBLPTR[0] = 1：程序存储器字的最高有效字节

TBLRD 指令可用如下方法修改 TBLPTR 的值：

- 不变
- 后加
- 后减
- 预加

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 *+ ；

指令执行前
TABLAT = 55h
TBLPTR = 00A356h
存储单元 （00A356h） = 34h
指令执行后
TABLAT = 34h
TBLPTR = 00A357h

示例 2： TBLRD +* ；

指令执行前
TABLAT = AAh
TBLPTR = 01A357h
存储单元 （01A357h） = 12h
存储单元 （01A358h） = 34h
指令执行后
TABLAT = 34h
TBLPTR = 01A358h

TBLWT **表写**

语法： TBLWT (*; *+; *-; +*)

操作数： 无

操作： 如果执行 TBLWT*，
(TABLAT) → 保持寄存器；
TBLPTR 不改变。
如果执行 TBLWT*+，
(TABLAT) → 保持寄存器；
(TBLPTR) + 1 → TBLPTR。
如果执行 TBLWT*-，
(TABLAT) → 保持寄存器；
(TBLPTR)–1 → TBLPTR。
如果执行 TBLWT+*，
(TBLPTR) + 1 → TBLPTR；
(TABLAT) → 保持寄存器。

受影响的状态位： 无

机器码：

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

说明： 此指令使用 TBLPTR 的低 3 位来确定要将 TABLAT 中的内容写入 8 个保持寄存器中的哪一个。该保持寄存器用于对程序存储器的内容编程 （关于对闪存存储器编程的更多详情，请参见**第 5.0 节 “存储器构成”**）。

TBLPTR （一个 21 位指针）指向程序存储器的每个字节。 TBLPTR 的寻址范围为 2 MB。 TBLPTR 的 LSb 选择要访问程序存储单元的哪个字节。

TBLPTR[0] = 0：程序存储器字的最低有效字节

TBLPTR[0] = 1：程序存储器字的最高有效字节

TBLWT 指令可用如下方法修改 TBLPTR 的值：

- 不变
- 后加
- 后减
- 预加

指令字数： 1

指令周期数： 2

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作
空操作	空操作（读 TABLAT）	空操作	空操作（写保持寄存器）

TBLWT **表写 （续）**

示例 1： TBLWT *+;

指令执行前
TABLAT = 55h
TBLPTR = 00A356h
保持寄存器（00A356h） = FFh
指令执行后 （表写操作完成）
TABLAT = 55h
TBLPTR = 00A357h
保持寄存器（00A356h） = 55h

示例 2： TBLWT +*;

指令执行前
TABLAT = 34h
TBLPTR = 01389Ah
保持寄存器（01389Ah） = FFh
保持寄存器（01389Bh） = FFh
指令执行后 （表写操作完成）
TABLAT = 34h
TBLPTR = 01389Bh
保持寄存器（01389Ah） = FFh
保持寄存器（01389Bh） = 34h

TSTFSZ **测试 f，为 0 则跳过**

语法： TSTFSZ f {,a}

操作数： 0 ≤ f ≤ 255
a ∈ [0,1]

操作： 如果 f = 0 则跳过

受影响的状态位： 无

机器码：

0110	011a	ffff	ffff

说明： 如果 “f” = 0 时，则会丢弃已取的指令，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 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

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

如果被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    :
```

指令执行前
PC = 地址 （HERE）
指令执行后
如果 CNT = 00h，
PC = 地址 （ZERO）
如果 CNT ≠ 00h，
PC = 地址 （NZERO）

XORLW **将立即数与 W 作逻辑异或运算**

语法： XORLW k

操作数： 0 ≤ k ≤ 255

操作： (W) .XOR. k → W

受影响的状态位： N 和 Z

机器码：

0000	1010	kkkk	kkkk

说明： 将 W 的内容与 8 位立即数 “k” 进行逻辑异或运算。结果保存在 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
```

指令执行前
W = B5h
指令执行后
W = 1Ah

XORWF **将 W 与 f 作逻辑异或运算**

语法： XORWF f {,d {,a}}

操作数： 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

操作： (W) .XOR. (f) → dest

受影响的状态位： N 和 Z

机器码：

0001	10da	ffff	ffff

说明： 将 W 的内容与寄存器 “f” 的内容作逻辑异或运算。如果 “d” 为 0，结果存储在 W 中。如果 “d” 为 1，结果存回寄存器 “f”（默认）。

如果 “a” 为 0，选择快速操作存储区。如果 “a” 为 1，使用 BSR 选择 GPR 存储区 （默认）。

如果 a 为 0 且使能了扩展的指令集，只要 f ≤ 95 （5Fh），指令就将以立即数变址寻址模式进行操作。详情请见**第 24.2.3 节 “立即数变址寻址模式中针对字节和位的指令”**。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 “f”	处理数据	写入 目标寄存器

示例： XORWF REG, 1, 0

指令执行前
REG = AFh
W = B5h
指令执行后
REG = 1Ah
W = B5h

24.2 扩展的指令集

除了PIC18标准指令集的75个指令之外，PIC18F87J10系列器件还提供针对内核CPU功能的可选扩展指令。这些添加的功能包括 8 个额外的指令，它们增加了间接和变址寻址操作以及对许多标准PIC18指令执行变址立即数偏移寻址模式的功能。

在未编程器件上，扩展指令集的这些额外功能在默认情况下是被使能的。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类为立即数操作类指令，它们既可以对文件选择寄存器进行操作，也可利用其进行变址寻址。其中的两个指令ADDFSR和SUBFSR，可直接对 FSR2 进行操作。这两个特殊指令（ADDULNK 和 SUBULNK）允许在执行后自动返回。

这些扩展的指令是专门针对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）而实现的。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
- 功能指针调用
- 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 24-3 提供了扩展指令集中的指令汇总。**第 24.2.2 节“扩展指令集”**提供了对这些指令的详细说明。表 24-1（第 290 页）提供了应用于标准和扩展的 PIC18 指令集的操作码字段的说明。

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。这些命令的语法作为参考提供给可能会查看编译器生成代码的用户。

24.2.1 扩展指令的语法

绝大多数的扩展指令使用变址参数，利用某个文件选择寄存器和某一偏移量来指定源和目标寄存器。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。这样做是为了表示此参数用作变址地址或偏移量。如果 MPASM™ 汇编器判断出一个变址地址或偏移量没有被括起来，它就会给出错误报告。

当使能扩展的指令集时，方括号也用于表示针对字节和位的指令中的变址参数。这是对标准指令语法的另一处更改。如需更多信息，请参见**第24.2.3.1节“标准PIC18命令的扩展指令语法”**。

注： 以前，在 PIC18 指令集和更早的指令集中使用方括号来表示可选参数。在本文和以后的文档中，可选参数将用大括号 （“{}”）表示。

表 24-3： PIC18 指令集的扩展

助记符，操作数		说明	周期数	16 位宽指令字 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）地址装入第一个字 将 f_d （目标）地址装入第二个字	2	1110 1111	1011 ffff	0zzz ffff	zzzz ffff	无
MOVSS	z_s, z_d	将 z_s （源）地址装入第一个字 将 z_d （目标）地址装入第二个字	2	1110 1111	1011 xxxx	1zzz xzzz	zzzz zzzz	无
PUSHL	k	将立即数保存在 FSR2，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

24.2.2 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

指令执行前
FSR2 = 03FFh

指令执行后
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 位的立即数 “k”。然后通过将 TOS 的值装入 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（二进制数 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

指令执行前
FSR2 = 03FFh
PC = 0100h

指令执行后
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，以在符号寻址中使用。如果使用了标号，那么指令语法将变为：{label} 指令参数。

CALLW **使用 WREG 调用子程序**

语法： CALLW

操作数： 无

操作： (PC+2) → TOS，
(W) → PCL，
(PCLATH) → PCH，
(PCLATU) → PCU

受影响的状态位： 无

机器码：

0000	0000	0001	0100

说明： 首先，返回地址 （PC+2）被压入返回堆栈。接下来，将 W 寄存器的内容写入 PCL， PCL 现有的值被丢弃。然后，PCLATH 和 PCLATU 的内容被分别锁存到 PCH 和 PCU。第二个周期数执行一条 NOP 指令，并同时读取下一条新的指令。

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

指令执行前
PC = 地址 （HERE）
PCLATH = 10h
PCLATU = 00h
W = 06h

指令执行后
PC = 001006h
TOS = 地址 （HERE + 2）
PCLATH = 10h
PCLATU = 00h
W = 06h

MOVSF **将变址寻址单元的内容移入 f**

语法： MOVSF $[z_s]$, f_d

操作数： $0 \leq z_s \leq 127$
$0 \leq f_d \leq 4095$

操作： ((FSR2) + z_s) → f_d

受影响的状态位： 无

机器码：

第 1 个字 （源） 1110	1011	0zzz	$zzzz_s$
第 2 个字 （目标） 1111	ffff	ffff	$ffff_d$

说明： 将源寄存器的内容移入目标寄存器 “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”指定目标寄存器的地址。两个地址均可以处于 4096 字节的数据空间 （000h 到 FFFh）中的任何位置。

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 非虚拟读取	空操作	写目标寄存器 “f”

示例： MOVSF [05h], REG2

指令执行前
FSR2 = 80h
85h 单元的内容 = 33h
REG2 = 11h

指令执行后
FSR2 = 80h
85h 单元的内容 = 33h
REG2 = 33h

MOVSS **源和目标都采用变址寻址的数据移动**

语法： MOVSS [z_s], [z_d]

操作数： $0 \leqslant z_s \leqslant 127$
$0 \leqslant z_d \leqslant 127$

操作： ((FSR2) + z_s) → ((FSR2) + z_d)

受影响的状态位： 无

机器码：

第 1 个字 （源）	1110	1011	1zzz	$zzzz_s$
第 2 个字 （目标）	1111	xxxx	xzzz	$zzzz_d$

说明： 将源寄存器的内容移到目标寄存器。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 字节数据存储器空间 （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，其值将返回 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]
```
指令执行前
FSR2 = 80h
85h 单元的内容 = 33h
86h 单元的的内容 = 11h
指令执行后
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 所指的单元，FSR2 减 1**

语法： PUSHL k

操作数： $0 \leqslant k \leqslant 255$

操作： k → (FSR2),
FSR2 –1 → FSR2

受影响的状态位： 无

机器码：

1111	1010	kkkk	kkkk

说明： 8 位立即数 “k” 被写入由 FSR2 指定的数据存储单元。随后 FSR2 减 1。

此指令允许用户将值压入软件堆栈。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取 “k”	处理数据	写入目标寄存器

示例：
```
PUSHL  08h
```
指令执行前
FSR2H:FSR2L = 01ECh
存储单元 （01ECh） = 00h
指令执行后
FSR2H:FSR2L = 01EBh
存储单元 （01ECh） = 08h

SUBFSR **FSR 减去立即数**

语法： SUBFSR f, k

操作数： 0 ≤ k ≤ 63

f ∈ [0,1, 2]

操作： FSR(f) – k → FSR(f)

受影响的状态位： 无

机器码：

1110	1001	ffkk	kkkk

说明： 用 “f” 指定的 FSR 的内容减去 6 位立即数 “k”。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 “f”	处理数据	写入目标寄存器

示例： SUBFSR 2, 23h

指令执行前
FSR2 = 03FFh

指令执行后
FSR2 = 03DCh

SUBULNK **FSR2 减去立即数并返回**

语法： SUBULNK k

操作数： 0 ≤ k ≤ 63

操作： FSR2 – k → FSR2

(TOS) → PC

受影响的状态位： 无

机器码：

1110	1001	11kk	kkkk

说明： 用 FSR2 的内容减去 6 位立即数 “k” 然后通过将 TOS 的值装入 PC，执行一条 RETURN 指令。

执行该指令需要两个周期，在第二个周期中执行一条 NOP 指令。

该指令可以被认为是 SUBFSR 指令的特例，其中 f = 3 （二进制数 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

指令执行前
FSR2 = 03FFh
PC = 0100h

指令执行后
FSR2 = 03DCh
PC = （TOS）

24.2.3 立即数变址寻址模式中针对字节和位的指令

注：	使能 PIC18 扩展指令集可能导致常规应用程序运行不正常或完全失败。

使能扩展的指令集除了使能了其中的 8 条新命令之外，还使能了立即数变址寻址模式（**第 5.6.1 节 “使用立即数作为偏移量进行变址寻址”**）。这对于标准 PIC18 指令集的很多命令的地址解析方法有很大的影响。

当禁用扩展的指令集时，将嵌入在操作码中的地址作为立即数地址处理：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）。当使能扩展的指令集且a = 0时，地址为5Fh或以下的数据寄存器参数被解析为相对于 FSR2 中的指针值的偏移量，而不是一个立即数地址。对于实际应用来说，这意味着所有使用快速操作 RAM 位作为参数的指令，即所有针对字节或位的指令，或者几乎半数的 PIC18 内核指令，在使能了扩展的指令集时操作都会有所不同。

当 FSR2 的内容为 00h 时，快速操作 RAM 的边界会被重新映射到它们的原始值。这对于创建向后兼容的代码会很有用处。如果使用此技术，可能有必要在 C 程序调用汇编子程序时保存 FSR2 的值，并在返回时将它恢复，这样做的目的是保护堆栈指针。用户还必须牢记扩展指令集的语法要求（见**第 24.2.3.1 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，在使能了扩展的指令集后，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，典型的针对字节和位的指令的示例。通过示例可以看出执行将如何受到影响。示例中的操作数条件适用于所有指令。

24.2.3.1 标准 PIC18 命令的扩展指令语法

当使能了扩展的指令集时，将用立即数偏移量 “k” 来替换标准的针对字节和位的命令中的数据寄存器参数 “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。因为在扩展的指令集中，编译器将括号中的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。

如果变址参数已被正确加上了括号来进行立即数偏移变址寻址，那么就不再需要指定快速操作 RAM 参数；此参数被自动假设为 0。这与标准操作（禁止扩展的指令集时）刚好相反。在变址寻址模式中，定义快速操作 RAM 位也将在 MPASM 汇编器中产生错误。

目标参数 “d” 的功能和以前一样。

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 或在源代码清单中加入 PE 伪指令进行调用。

24.2.4 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的指令集，尤其是那些不使用软件堆栈的用户。

此外，立即数变址寻址模式可能会给写入 PIC18 汇编器的常规应用程序带来问题。这是因为常规的指令会尝试寻址快速操作存储区中地址低于 5Fh 的寄存器。当使能了扩展的指令集时，这些地址被解析为相对于 FSR2 的立即数偏移量，所以应用程序会读或写错误的地址。

将应用程序移植到PIC18F87J10系列器件时，考虑代码的类型是非常重要的。在使用扩展的指令集时，用 C 语言编写的代码较长的重入应用程序会运行的很好，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。

ADDWF **将 W 与变址寻址单元的内容相加（立即数变址寻址模式）**

语法： ADDWF [k] {,d}

操作数： 0 ≤ k ≤ 95
d ∈ [0,1]

操作： (W) + ((FSR2) + k) → dest

受影响的状态位： N、OV、C、DC 和 Z

机器码：

0010	01d0	kkkk	kkkk

说明： 将 W 的内容与由 FSR2 加上偏移量“k”指定的寄存器的内容相加。

如果“d”为 0，结果存储在 W 中。如果“d”为 1，结果存回寄存器“f”（默认）。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取“k”	处理数据	写入目标寄存器

示例： ADDWF [OFST],0

指令执行前
W = 17h
OFST = 2Ch
FSR2 = 0A00h
0A2Ch 单元的内容 = 20h
指令执行后
W = 37h
0A2Ch 单元的内容 = 20h

BSF **将变址寻址单元的相应位置 1（立即数变址寻址模式）**

语法： BSF [k], b

操作数： 0 ≤ f ≤ 95
0 ≤ b ≤ 7

操作： 1 → ((FSR2) + k)<b>

受影响的状态位： 无

机器码：

1000	bbb0	kkkk	kkkk

说明： 将由 FSR2 加上偏移量“k”指定的寄存器中的位“b”置 1。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器“f”	处理数据	写入目标寄存器

示例： BSF [FLAG_OFST], 7

指令执行前
FLAG_OFST = 0Ah
FSR2 = 0A00h
0A0Ah 单元的内容 = 55h
指令执行后
0A0Ah 单元的内容 = D5h

SETF **将变址寻址单元置为全 1（立即数变址寻址模式）**

语法： SETF [k]

操作数： 0 ≤ k ≤ 95

操作： FFh ((FSR2) + k)

受影响的状态位： 无

机器码：

0110	1000	kkkk	kkkk

说明： 将由 FSR2 加上偏移量“k”指定的寄存器的内容置为 FFh。

指令字数： 1

指令周期数： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读取“k”	处理数据	写寄存器

示例： SETF [OFST]

指令执行前
OFST = 2Ch
FSR2 = 0A00h
0A2Ch 单元的内容 = 00h
指令执行后
0A2Ch 单元的内容 = FFh

24.2.5 使用 MICROCHIP MPLAB® IDE 工具的注意事项

最新版本的 Microchip 软件工具，可以完全支持 PIC18F87J10 系列器件的扩展指令集。包括 MPLAB C18 C 语言编译器、MPASM 汇编语言和 MPLAB 集成开发环境（IDE）。

在选择用于软件开发的目标器件时，MPLAB IDE 将把该器件的默认配置位自动置 1。在禁用扩展指令集和立即数变址寻址模式时，XINST 配置位的默认设置是 0。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执行应用程序。

要使用扩展的指令集开发软件，用户必须设置他们的语言工具以实现对扩展指令和变址寻址模式的支持。根据所使用的环境，可以通过以下几种方法：

- 开发环境中的菜单选项或对话框，允许用户配置项目的语言工具及其设置
- 命令行选项
- 源代码中的伪指令

这些选项在不同的编译器、汇编器和开发环境中将有所不同。建议用户在其开发系统所附带的文档中查询相应的信息。

25.0 开发支持

一系列硬件及软件开发工具对 PICmicro® 单片机提供支持：

- 集成开发环境
  - MPLAB® IDE 软件
- 汇编器 / 编译器 / 链接器
  - MPASM™ 汇编器
  - MPLAB C18 和 MPLAB C30 C 编译器
  - MPLINK™ 目标链接器 / MPLIB™ 目标库管理器
  - MPLAB ASM30 汇编器 / 链接器 / 库
- 模拟器
  - MPLAB SIM 软件模拟器
- 仿真器
  - MPLAB ICE 2000 在线仿真器
  - MPLAB ICE 4000 在线仿真器
- 在线调试器
  - MPLAB ICD 2
- 器件编程器
  - PICSTART® Plus 开发编程器
  - MPLAB PM3 器件编程器
- 低成本演示和开发板及评估工具包

25.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16 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）
  - 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
- 可视化器件初始化程序，便于进行寄存器的初始化
- 鼠标停留在变量上进行查看的功能
- 通过拖放把变量从源代码窗口拉到观察窗口
- 丰富的在线帮助
- 集成了可选的第三方工具，如 HI-TECH 软件 C 编译器和 IAR C 编译器

MPLAB IDE 可以让您：

- 编辑源文件 （汇编语言或 C 语言）
- 点击一次即可完成汇编 （或编译）并将代码下载到 PICmicro MCU 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 （汇编语言或 C 语言）
  - 混合汇编语言和 C 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更强大的工具时的学习时间。

25.2 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于所有的 PICmicro 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 汇编器具有如下特征：

- 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

25.3 MPLAB C18 和 MPLAB C30 C 编译器

MPLAB C18 和 MPLAB C30 代码开发系统是完全的 ANSI C 编译器，分别适用于 Microchip 的 PIC18 系列单片机和 dsPIC30F 系列数据信号控制器。这些编译器可提供其他编译器并不具备的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，编译器提供了针对 MPLAB IDE 调试器的优化符号信息。

25.4 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK目标链接器包含了由MPASM汇编器和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中。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征：

- 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

25.5 MPLAB ASM30 汇编器、链接器和库管理器

MPLAB ASM30 汇编器为 dsPIC30F 器件提供转换自符号汇编语言的可重定位机器码。 MPLAB C30 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征：

- 支持整个 dsPIC30F 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

25.6 MPLAB SIM 软件模拟器

MPLAB SIM 软件模拟器在指令级对 PICmicro MCU 和 dsPIC® DSC 进行模拟，使得用户可以在 PC 主机的环境下进行代码开发。对于任何给定的指令，用户均可对数据区进行检查或修改，并通过各种触发机制来产生激励。可以将各寄存器的情况记录在文件中，以便进行进一步地运行时分析。跟踪缓冲器和逻辑分析器的显示使模拟器还能记录和跟踪程序的执行、 I/O 的动作以及内部寄存器的状况。

MPLAB SIM 软件模拟器完全支持使用 MPLAB C18 和 MPLAB C30 C 编译器以及MPASM和MAPLAB ASM30 汇编器的符号调试。该软件模拟器可用于在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。

25.7 MPLAB ICE 2000 高性能在线仿真器

MPLAB ICE 2000 在线仿真器旨在为产品开发工程师提供一整套用于 PICmicro 单片机的设计工具。 MPLAB ICE 2000 在线仿真器的软件控制由 MPLAB 集成开发环境平台提供，它允许在单一环境下进行编辑、编译、下载以及源代码调试。

MPLAB ICE 2000 是全功能仿真器系统，它具有增强的跟踪、触发和数据监控功能。处理器模块可插拔，使系统可轻松进行重新配置以适应各种不同处理器的仿真需要。MPLAB ICE 2000 在线仿真器的架构允许对其进行扩展以支持新的 PICmicro 单片机。

MPLAB ICE 2000 在线仿真器系统设计为一款实时仿真系统，该仿真系统具备通常只有昂贵的开发工具中才有的高级功能。选择 PC 平台和 Microsoft® Windows® 32 位操作系统可使这些功能在一个简单而统一的应用中得到很好的利用。

25.8 MPLAB ICE 4000 高性能在线仿真器

MPLAB ICE 4000 在线仿真器旨在为产品开发工程师提供一整套用于高端 PICmicro MCU 和 dsPIC DSC 的设计工具。 MPLAB ICE 4000 在线仿真器的软件控制由 MPLAB 集成开发环境平台提供，它允许在单一环境下进行编辑、编译、下载以及源代码调试。

MPLAB ICE 4000 是高级的仿真系统，除具备 MPLAB ICE 2000 的所有功能外，它还增加了适用于 dsPIC30F 和 PIC18XXXX 器件的仿真存储容量以及高速性能。该仿真器的先进特性包括复杂触发和定时功能及高达 2 Mb 的仿真存储容量。

MPLAB ICE 4000 在线仿真系统设计为一款实时仿真系统，该仿真系统具备通常只有在更加昂贵的开发工具中才有的高级功能。选择 PC 平台和 Microsoft Windows 32 位操作系统可使这些功能在一个简单而统一的应用程序中得以很好的利用。

25.9 MPLAB ICD 2 在线调试器

Microchip 的在线调试器 MPLAB ICD 2 是一款功能强大而成本低廉的运行时开发工具，通过 RS-232 或高速 USB 接口与 PC 主机相连。该工具基于闪存 PICmicro MCU，可用于开发本系列及其他 PICmicro MCU 和 dsPIC DSC。 MPLAB ICD 2 使用了闪存器件中内建的在线调试功能。该功能结合 Microchip 的在线串行编程（In-Circuit Serial Programming™，ICSP™）协议，可在 MPLAB 集成开发环境的图形用户界面上提供成本效益很高的在线闪存调试。这使设计人员可通过设置断点、单步运行以及对变量、CPU 状态以及外设寄存器进行监视的方法实现源代码的开发和调试。其全速运行特性可对硬件和应用进行实时测试。MPLAB ICD 2 还可用作某些 PICmicro 器件的开发编程器。

25.10 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款通用的、符合 CE 规范的器件编程器，其可编程电压设置在 V_{DDMIN} 和 V_{DDMAX} 之间时可靠性最高。它有一个用来显示菜单和错误信息的大 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 PICmicro 器件进行读取、验证和编程。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，可对存储器很大的器件进行快速编程，它还采用 SD/MMC 卡用作文件存储及数据安全应用。

25.11 PICSTART Plus 开发编程器

PICSTART Plus开发编程器是一款易于使用而成本低廉的原型编程器。它通过 COM（RS-232）端口与 PC 相连。 MPLAB 集成开发环境软件使得该编程器的使用简便、高效。PICSTART Plus 开发编程器支持采用 DIP 封装的大部分 PICmicro 器件，其引脚数最多可达 40 个。引脚数更多的器件，如 PIC16C92X 和 PIC17C76X，可通过连接一个转接插槽来获得支持。PICSTART Plus 开发编程器符合 CE 规范。

25.12 演示、开发和评估板

有许多演示、开发和评估板可用于各种 PICmicro 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，用于测试和修改。

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。

有关演示、开发和评估工具包的完整列表，请查阅 Microchip 公司网页（www.microchip.com）以及最新的 *“Product Selector Guide（产品选型指南）”*（DS00148）。

26.0 电气规范

绝对最大值（†）

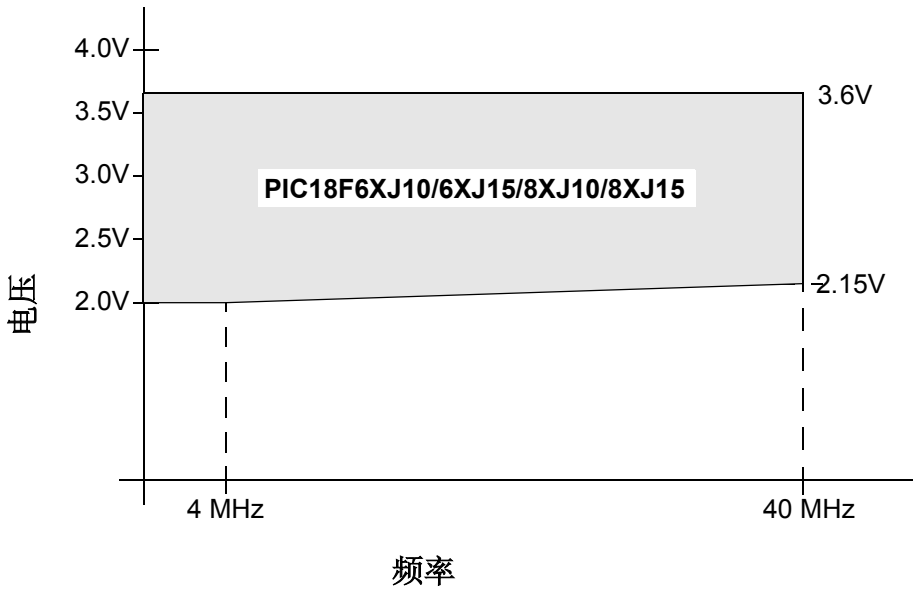
偏置电压下的环境温度 ..... -40°C 至 +125°C
储存温度 ..... -65°C 至 +150°C
数字 I/O 引脚 （除了 VDD 和 $\overline{\text{MCLR}}$ 之外）相对于 VSS 的电压 ..... -0.3V 至 6.0V
数模组合引脚 （除了 VDD 和 $\overline{\text{MCLR}}$ 之外）相对于 VSS 的电压 ..... -0.3V 至 （VDD + 0.3V）
VDDCORE 相对于 VSS 的电压 ..... -0.3V 至 2.75V
VDD 相对于 VSS 的电压 ..... -0.3V 至 3.6V
$\overline{\text{MCLR}}$ 相对于 VSS 的电压 **（注 2）** ..... 0V 至 4.0V
总功耗 **（注 1）** 1.0W
VSS 引脚的最大输出电流 ..... 300 mA
VDD 引脚的最大输入电流 ..... 250 mA
PORTB 和 PORTC I/O 引脚的最大灌电流 ..... 25 mA
PORTD、 PORTE 和 PORTJ I/O 引脚的最大灌电流 ..... 8 mA
PORTA、 PORTF、 PORTG 和 PORTH I/O 引脚的最大灌电流 ..... 2 mA
PORTB 和 PORTC I/O 引脚的最大拉电流 ..... 25 mA
PORTD、 PORTE 和 PORTJ I/O 引脚的最大拉电流 ..... 8 mA
PORTA、 PORTF、 PORTG 和 PORTH I/O 引脚的最大拉电流 ..... 2 mA
所有端口的最大灌电流 ..... 200 mA
所有端口的最大拉电流 ..... 200 mA

注 1： 功耗按如下公式计算：
$Pdis = V_{DD} \times \{I_{DD} - \Sigma\ I_{OH}\} + \Sigma\ \{(V_{DD} - V_{OH}) \times I_{OH}\} + \Sigma\ (V_{OL} \times I_{OL})$

2： 如果 $\overline{\text{MCLR}}$ 引脚上的尖峰电压低于 VSS，感应电流大于 80 mA，可能会引起器件锁死。因此当 $\overline{\text{MCLR}}$ 引脚被驱动为低电平时，应该在该引脚上串连一个 50-100Ω 的电阻，而不是直接将该引脚连到 VSS。

† 注意： 如果器件的工作条件超过 “绝对最大值”列出的范围，就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，我们建议不要使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额定值条件下，其稳定性会受到影响。

图 **26-1：** **PIC18F87J10 系列电压－频率关系图 （工业级）**



26.1 直流规范： 供电电压， PIC18F87J10 系列 （工业级）

PIC18F87J10 系列 （工业级）			标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
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	VDDCORE 2.5	— —	3.6 3.6	V V	ENVREG = 0 ENVREG = 1
D001B	VDDCORE	单片机内核的外部供电电源	2.0	—	2.75	V	ENVREG = 0
D002	VDR	RAM 数据保持电压 (1)	1.5	—	—	V	
D003	VPOR	VDD 启动电压 确保能够产生内部上电复位信号	—	—	0.7	V	如需详细信息，请参见第 4.3 节 “上电复位 （POR）”。
D004	SVDD	VDD 上升率 确保能够产生内部上电复位信号	0.05	—	—	V/ms	如需详细信息，请参见第 4.3 节 “上电复位 （POR）”。

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。

26.2 直流规范： 掉电和供电电流 PIC18F87J10 系列 （工业级）

PIC18F87J10 系列 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
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	27	69	µA	-40°C	VDD = 2.0V (5), （**休眠**模式）
		43	69	µA	+25°C	
		121	149	µA	+85°C	
	所有器件	49	104	µA	-40°C	VDD = 2.5 V (5), （**休眠**模式）
		69	104	µA	+25°C	
		166	184	µA	+85°C	
	所有器件	85	203	µA	-40°C	VDD = 3.3V (6), （**休眠**模式）
		100	203	µA	+25°C	
		220	289	µA	+85°C	

图注： TBD = 待定

注 1： 休眠模式下的掉电电流不取决于振荡器类型。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，禁止所有会增大新增电流的功能部件 （比如 WDT， Timer1 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。
在正常的工作模式下，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： 当器件振荡器配置为 RC 模式时，该电流不包括流经 REXT 的电流。流经该电阻的电流可以由公式 Ir=VDD/2REXT （mA）来估算，其中 REXT 的单位是 kΩ。

4： 标准低成本 32 kHz 晶振的工作温度范围为从 -10°C 到 +70°C。扩展级温度晶振的成本要高很多。

5： ENVREG = 0，禁止稳压器。

6： ENVREG = 1，使能稳压器。

26.2 直流规范： 掉电和供电电流
PIC18F87J10 系列（工业级）（续）

PIC18F87J10 系列 （工业级）		标准工作条件（除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
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	7.0	10.0	μA	-40°C	VDD = 2.0V	FOSC = 31 kHz（**RC_RUN** 模式，内部振荡器作为时钟源）
		6.4	10.0	μA	+25°C		
		6.5	10.0	μA	+85°C		
	所有器件	13.6	14.7	μA	-40°C	VDD = 2.5V	
		12.2	14.7	μA	+25°C		
		11.5	14.7	μA	+85°C		
	所有器件	21.0	29.7	μA	-40°C	VDD = 3.3V	
		19.1	29.7	μA	+25°C		
		17.6	29.7	μA	+85°C		
	所有器件	7.0	10.0	μA	-40°C	VDD = 2.0V	FOSC = 31 kHz（**RC_IDLE** 模式，内部振荡器作为时钟源）
		6.4	10.0	μA	+25°C		
		6.5	10.0	μA	+85°C		
	所有器件	13.6	14.7	μA	-40°C	VDD = 2.5V	
		12.2	14.7	μA	+25°C		
		11.5	14.7	μA	+85°C		
	所有器件	21.0	29.7	μA	-40°C	VDD = 3.3V	
		19.1	29.7	μA	+25°C		
		17.6	29.7	μA	+85°C		

图注： TBD = 待定

注 1： 休眠模式下的掉电电流不取决于振荡器类型。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，禁止所有会增大新增电流的功能部件（比如 WDT， Timer1 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。
在正常的工作模式下，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： 当器件振荡器配置为 RC 模式时，该电流不包括流经 REXT 的电流。流经该电阻的电流可以由公式 Ir=VDD/2REXT（mA）来估算，其中 REXT 的单位是 kΩ。

4： 标准低成本 32 kHz 晶振的工作温度范围为从 -10°C 到 +70°C。扩展级温度晶振的成本要高很多。

5： ENVREG = 0，禁止稳压器。

6： ENVREG = 1，使能稳压器。

26.2 直流规范： 掉电和供电电流
PIC18F87J10 系列 （工业级）（续）

PIC18F87J10 系列 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
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	7.2	14.7	mA	-40°C	VDD = 2.0V(5)	FOSC = 1 MHz （PRI_RUN 模式，EC 振荡器）
		6.6	14.7	mA	+25°C		
		6.4	14.7	mA	+85°C		
	所有器件	13.6	19.7	mA	-40°C	VDD = 2.5V(5)	
		12.1	19.7	mA	+25°C		
		11.4	19.7	mA	+85°C		
	所有器件	21.0	34.7	mA	-40°C	VDD = 3.3V(6)	
		19.1	34.7	mA	+25°C		
		17.8	34.7	mA	+85°C		
	所有器件	7.5	16.7	mA	-40°C	VDD = 2.0V(5)	FOSC = 4 MHz （PRI_RUN 模式，EC 振荡器）
		6.8	16.7	mA	+25°C		
		6.6	16.7	mA	+85°C		
	所有器件	14.1	21.7	mA	-40°C	VDD = 2.5V(5)	
		12.3	21.7	mA	+25°C		
		11.7	21.7	mA	+85°C		
	所有器件	21.6	36.7	mA	-40°C	VDD = 3.3V(6)	
		19.8	36.7	mA	+25°C		
		18.4	36.7	mA	+85°C		
	所有器件	20.1	39.3	mA	-40°C	VDD = 2.5V(5)	FOSC = 40 MHz （PRI_RUN 模式，EC 振荡器）
		18.7	39.3	mA	+25°C		
		17.8	39.3	mA	+85°C		
	所有器件	27.5	47.7	mA	-40°C	VDD = 3.3V(6)	
		25.8	47.7	mA	+25°C		
		24.4	47.7	mA	+85°C		

图注： TBD = 待定

注 1： 休眠模式下的掉电电流不取决于振荡器类型。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，禁止所有会增大新增电流的功能部件 （比如 WDT， Timer1 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。
在正常的工作模式下，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： 当器件振荡器配置为 RC 模式时，该电流不包括流经 REXT 的电流。流经该电阻的电流可以由公式 Ir=VDD/2REXT （mA）来估算，其中 REXT 的单位是 kΩ。

4： 标准低成本 32 kHz 晶振的工作温度范围为从 -10°C 到 +70°C。扩展级温度晶振的成本要高很多。

5： ENVREG = 0，禁止稳压器。

6： ENVREG = 1，使能稳压器。

26.2 直流规范： 掉电和供电电流 PIC18F87J10 系列（工业级）（续）

PIC18F87J10 系列（工业级）		标准工作条件（除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	供电电流（IDD）(2)						
	所有器件	16.1	35.3	mA	-40°C	VDD = 2.5V(5)	FOSC = 4 MHz，内部 16 MHz 时钟源（**PRI_RUN HSPLL** 模式）
		14.4	35.3	mA	+25°C		
		13.7	35.3	mA	+85°C		
	所有器件	23.6	37.1	mA	-40°C	VDD = 3.3V(6)	FOSC = 4 MHz，内部 16 MHz 时钟源（**PRI_RUN HSPLL** 模式）
		21.8	37.1	mA	+25°C		
		20.4	37.1	mA	+85°C		
	所有器件	20.1	39.3	mA	-40°C	VDD = 2.5V(5)	FOSC = 10 MHz，内部 40 MHz 时钟源（**PRI_RUN HSPLL** 模式）
		18.7	39.3	mA	+25°C		
		17.8	39.3	mA	+85°C		
	所有器件	27.5	47.7	mA	-40°C	VDD = 3.3V(6)	FOSC = 10 MHz，内部 40 MHz 时钟源（**PRI_RUN HSPLL** 模式）
		25.8	47.7	mA	+25°C		
		24.4	47.7	mA	+85°C		

图注： TBD = 待定

注 1： 休眠模式下的掉电电流不取决于振荡器类型。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，禁止所有会增大新增电流的功能部件（比如 WDT， Timer1 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。
在正常的工作模式下，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： 当器件振荡器配置为 RC 模式时，该电流不包括流经 REXT 的电流。流经该电阻的电流可以由公式 Ir=VDD/2REXT（mA）来估算，其中 REXT 的单位是 kΩ。

4： 标准低成本 32 kHz 晶振的工作温度范围为从 -10°C 到 +70°C。扩展级温度晶振的成本要高很多。

5： ENVREG = 0，禁止稳压器。

6： ENVREG = 1，使能稳压器。

26.2 直流规范： 掉电和供电电流 PIC18F87J10 系列 （工业级）（续）

PIC18F87J10 系列 （工业级）		标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
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	7.2	20	mA	-40°C	VDD = 2.0V (5)	FOSC = 1 MHz （PRI_IDLE 模式，EC 振荡器）
		6.6	20	mA	+25°C		
		6.4	20	mA	+85°C		
	所有器件	13.6	29.3	mA	-40°C	VDD = 2.5V (5)	
		12.1	29.3	mA	+25°C		
		11.4	29.3	mA	+85°C		
	所有器件	21.0	29.7	mA	-40°C	VDD = 3.3V (6)	
		19.1	29.7	mA	+25°C		
		17.8	29.7	mA	+85°C		
	所有器件	7.5	23.3	mA	-40°C	VDD = 2.0V (5)	FOSC = 4 MHz （PRI_IDLE 模式，EC 振荡器）
		6.8	23.3	mA	+25°C		
		6.6	23.3	mA	+85°C		
	所有器件	14.1	33.3	mA	-40°C	VDD = 2.5V (5)	
		12.3	33.3	mA	+25°C		
		11.7	33.3	mA	+85°C		
	所有器件	21.6	31.7	mA	-40°C	VDD = 3.3V (6)	
		19.8	31.7	mA	+25°C		
		18.4	31.7	mA	+85°C		
	所有器件	20.1	39.3	mA	-40°C	VDD = 2.5V (5)	FOSC = 40 MHz （PRI_IDLE 模式，EC 振荡器）
		18.7	39.3	mA	+25°C		
		17.8	39.3	mA	+85°C		
	所有器件	27.5	47.7	mA	-40°C	VDD = 3.3V (6)	
		25.8	47.7	mA	+25°C		
		24.4	47.7	mA	+85°C		

图注： TBD = 待定

注 1： 休眠模式下的掉电电流不取决于振荡器类型。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，禁止所有会增大新增电流的功能部件 （比如 WDT， Timer1 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。
在正常的工作模式下，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： 当器件振荡器配置为 RC 模式时，该电流不包括流经 REXT 的电流。流经该电阻的电流可以由公式 Ir=VDD/2REXT （mA）来估算，其中 REXT 的单位是 kΩ。

4： 标准低成本 32 kHz 晶振的工作温度范围为从 -10°C 到 +70°C。扩展级温度晶振的成本要高很多。

5： ENVREG = 0，禁止稳压器。

6： ENVREG = 1，使能稳压器。

26.2 直流规范： 掉电和供电电流
PIC18F87J10 系列（工业级）（续）

PIC18F87J10 系列（工业级）		标准工作条件（除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
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	7.0	10.0	mA	-10°C	VDD = 2.0V(5)	FOSC = 32 kHz(4)（**SEC_RUN** 模式，Timer1 作为时钟源）
		6.4	10.0	mA	+25°C		
		6.5	10.0	mA	+70°C		
	所有器件	13.6	14.7	mA	-10°C	VDD = 2.5V(5)	
		12.2	14.7	mA	+25°C		
		11.5	14.7	mA	+70°C		
	所有器件	21.0	29.7	mA	-10°C	VDD = 3.3V(6)	
		19.1	29.7	mA	+25°C		
		17.6	29.7	mA	+70°C		
	所有器件	7.0	10.0	mA	-10°C	VDD = 2.0V(5)	FOSC = 32 kHz(4)（**SEC_IDLE** 模式，Timer1 作为时钟源）
		6.4	10.0	mA	+25°C		
		6.5	10.0	mA	+70°C		
	所有器件	13.6	14.7	mA	-10°C	VDD = 2.5V(5)	
		12.2	14.7	mA	+25°C		
		11.5	14.7	mA	+70°C		
	所有器件	21.0	29.7	mA	-10°C	VDD = 3.3V(6)	
		19.1	29.7	mA	+25°C		
		17.6	29.7	mA	+70°C		

图注： TBD = 待定

注 1： 休眠模式下的掉电电流不取决于振荡器类型。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，禁止所有会增大新增电流的功能部件（比如 WDT， Timer1 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。
在正常的工作模式下，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： 当器件振荡器配置为 RC 模式时，该电流不包括流经 REXT 的电流。流经该电阻的电流可以由公式 Ir=VDD/2REXT（mA）来估算，其中 REXT 的单位是 kΩ。

4： 标准低成本 32 kHz 晶振的工作温度范围为从 -10°C 到 +70°C。扩展级温度晶振的成本要高很多。

5： ENVREG = 0，禁止稳压器。

6： ENVREG = 1，使能稳压器。

26.2 直流规范： 掉电和供电电流
PIC18F87J10 系列（工业级）（续）

PIC18F87J10 系列 （工业级）		标准工作条件（除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	器件	典型值	最大值	单位	条件		
	模块差分电流（ΔIWDT、ΔIOSCB 和 ΔIAD）						
D022 （ΔIWDT）	看门狗定时器	TBD	7.3	μA	-40°C	VDD = 2.0V	
		TBD	7.3	μA	+25°C		
		TBD	5.1	μA	+85°C		
		TBD	6.0	μA	-40°C	VDD = 2.5V	
		TBD	6.0	μA	+25°C		
		TBD	6.0	μA	+85°C		
		TBD	10.5	μA	-40°C	VDD = 3.3V	
		TBD	10.5	μA	+25°C		
		TBD	10.5	μA	+85°C		
D025 （ΔIOSCB）	Timer1 振荡器	TBD	18.5	μA	-40°C	VDD = 2.0V	Timer1(3) 的频率为 32 kHz
		TBD	19.1	μA	+25°C		
		TBD	19.1	μA	+85°C		
		TBD	18.6	μA	-40°C	VDD = 2.5V	Timer1(3) 的频率为 32 kHz
		TBD	19.2	μA	+25°C		
		TBD	19.1	μA	+85°C		
		TBD	19.1	μA	-40°C	VDD = 3.3V	Timer1(3) 的频率为 32 kHz
		TBD	19.1	μA	+25°C		
		TBD	19.1	μA	+85°C		
D026 （ΔIAD）	A/D 转换器	TBD	10.9	μA	-40°C 至	VDD = 2.0V	A/D 启动，但不转换
		TBD	11.4	μA	-40°C 至	VDD = 2.5V	
		TBD	11.9	μA	-40°C 至	VDD = 3.3V	

图注： TBD = 待定

注 1： 休眠模式下的掉电电流不取决于振荡器类型。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，禁止所有会增大新增电流的功能部件（比如 WDT， Timer1 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。
在正常的工作模式下，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： 当器件振荡器配置为 RC 模式时，该电流不包括流经 REXT 的电流。流经该电阻的电流可以由公式 Ir=VDD/2REXT（mA）来估算，其中 REXT 的单位是 kΩ。

4： 标准低成本 32 kHz 晶振的工作温度范围为从 -10°C 到 +70°C。扩展级温度晶振的成本要高很多。

5： ENVREG = 0，禁止稳压器。

6： ENVREG = 1，使能稳压器。

26.3 直流规范： PIC18F87J10 系列 （工业级）

直流规范			标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）			
参数编号	符号	特性	最小值	最大值	单位	条件
	VIL	**输入低电压**				
		I/O 端口：				
D030		带 TTL 缓冲器	VSS	0.15 VDD	V	VDD < 3.3V
D030A			—	0.8	V	3.3V ≤ Vdd ≤ 3.6V
D031		带施密特触发缓冲器	VSS	0.2 VDD	V	
D032		$\overline{MCLR}$	VSS	0.2 VDD	V	
D033		OSC1	VSS	0.3 VDD	V	HS 和 HSPLL 模式
D033A		OSC1	VSS	0.2 VDD	V	EC 模式 (1)
D034		T13CKI	VSS	0.3 VDD	V	
	VIH	**输入高电压**				
		I/O 端口：				
D040		带 TTL 缓冲器	0.25 VDD + 0.8V	VDD	V	VDD < 3.3V
D040A			2.0	VDD	V	3.3V ≤ VDD ≤ 3.6V
D041		带施密特触发缓冲器	0.8 VDD	VDD	V	
D042		$\overline{MCLR}$	0.8 VDD	VDD	V	
D043		OSC1	0.7 VDD	VDD	V	HS 和 HSPLL 模式
D043A		OSC1	0.8 VDD	VDD	V	EC 模式
D044		T13CKI	1.6	VDD	V	
	IIL	**输入泄漏电流** (2,3)				
D060		I/O 端口	—	±1	μA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态
D061		$\overline{MCLR}$	—	±5	μA	Vss ≤ VPIN ≤ VDD
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： 在 RC 振荡器配置中， OSC1/CLKI 引脚上的信号为施密特触发器的输入信号。在 RC 模式中建议不要使用外部时钟驱动 PICmicro® 器件。

2： $\overline{MCLR}$ 引脚上的泄漏电流主要取决于施加的电平。规定电平为正常工作条件下的电平。在不同的输入电压下可测得更高的泄漏电流。

3： 负电流定义为引脚输出的电流。

26.3 直流规范： PIC18F87J10 系列 （工业级）

直流规范			标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）			
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	—	0.6	V	IOL = 2.4 mA， VOL = 0.4V， VDD = 3.3V，-40°C 到 +85°C
D083		OSC2/CLKO （EC 和 ECIO 模式）	—	0.6	V	IOL = 2.4 mA， VOL = 0.4V， VDD = 3.3V，-40°C 到 +85°C
	VOH	**输出高电压 (3)**				
D090		I/O 端口	VDD -0.7	—	V	IOH = -3.0 mA， VDD = 3.3V， -40°C 到 +85°C
D092		OSC2/CLKO （RC、 RCIO、 EC 和 ECIO 模式）	VDD -0.7	—	V	IOH = -1.3 mA， VDD = 3.3V， -40°C 到 +85°C
		输出引脚上的容性负载规范				
D100(4)	COSC2	OSC2 引脚	—	15	pF	适用于使用外部时钟驱动 OSC1 的 HS 模式
D101	CIO	所有的 I/O 引脚和 OSC2 引脚 （RC 模式）	—	50	pF	满足交流时序规范
D102	CB	SCLx 和 SDAx	—	400	pF	I^2C™ 规范

注 1： 在 RC 振荡器配置中， OSC1/CLKI 引脚上的信号为施密特触发器的输入信号。在 RC 模式中建议不要使用外部时钟驱动 PICmicro® 器件。

2： $\overline{MCLR}$ 引脚上的泄漏电流主要取决于施加的电平。规定电平为正常工作条件下的电平。在不同的输入电压下可测得更高的泄漏电流。

3： 负电流定义为引脚输出的电流。

表 26-1： 存储器编程要求

直流规范			标准工作条件 （除非另行声明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
		闪存程序存储器					
D130	EP	耐擦写能力	100	1K	—	E/W	-40°C 至 +85°C
D131	VPR	用于读入的 VDD	VMIN	—	3.6	V	VMIN= 最小工作电压
D132B	VPEW	用于自定时写的 VDD	VMIN	—	3.6	V	VMIN= 最小工作电压
D133A	TIW	自定时写周期时间	—	2.8	—	ms	
D134	TRETD	保存时间	10	20	—	年	假如没有违反其他规范
D135	IDDP	编程期间的供电电流	—	10	—	mA	
D1xxx	TWE	每个擦写周期的写入次数	—	—	1		

† 除非另行声明，否则 “典型值”栏中的数据均为 5.0V、 25 ℃条件下的值。这些参数仅作为设计参考，未经测试。

表 26-2： 比较器规范

工作条件：3.0V < VDD < 3.6V，-40°C < TA < +85°C （除非另行声明）							
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**备注**
D300	VIOFF	输入偏移电压	—	± 5.0	± 10	mV	
D301	VICM	输入共模电压 *	0	—	VDD – 1.5	V	
D302	CMRR	共模抑制比 *	55	—	—	dB	
300	TRESP	响应时间 (1) *	—	150	400	ns	
301	TMC2OV	比较器模式改变到输出有效之间的时间 *	—	—	10	µs	

* 这些参数仅为特征值，未经测试。

注 1： 响应时间是在比较器一个输入端的电压为 （VDD – 1.5） /2，而另一个输入端的电压从 VSS 变化到 VDD 的过程中测得的。

表 26-3： 参考电压规范

工作条件：3.0V < VDD < 3.6V，-40°C < TA < +85°C （除非另行声明）							
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/24	—	VDD/32	LSb	
D311	VRAA	绝对精度	—	—	1/2	LSb	
D312	VRUR	单位电阻值 （R）	—	2k	—	Ω	
310	TSET	建立时间 (1)	—	—	10	µs	

注 1： 建立时间是在 CVRR = 1 且 CVR3:CVR0 从 “0000”跳变到 “1111”时测得的。

表 26-4： 内部稳压器规范

工作条件：-40°C < TA < +85°C （除非另行声明）							
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	—	2.5	—	V	
	CEFC	外部滤波器电容值	1	10	—	µF	该电容必须是低 ESR 的。

* 这些参数仅为特征值，未经测试。尚未给这些规范参数分配编号。

26.4 交流 （时序）规范

26.4.1 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）及其含意：

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	T13CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

大写字母及其含意：

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	输出访问	High	高
BUF	总线空闲	Low	低

TCC:ST （仅用于 I²C 规范）

CC			
HD	保持	SU	建立
ST			
DAT	数据输入保持	STO	停止条件
STA	启动条件		

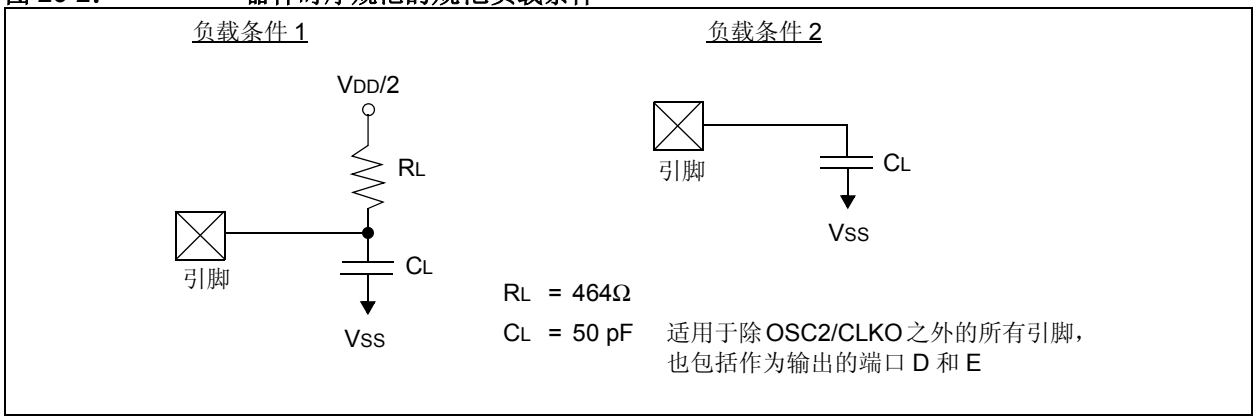
26.4.2 时序条件

表 26-5 中指定的温度和电压适用于所有的时序规范（除非另外指明）。图 26-2 规定了时序规范的负载条件。

表 26-5： 温度和电压规范——交流

交流规范	**标准工作条件（除非另行声明）** 工作温度 -40°C ≤ TA ≤ +85°C （工业级） 直流规范**第 26.1 节**和**第 26.3 节**说明了工作电压 VDD 的范围。

图 26-2： 器件时序规范的规范负载条件



26.4.3 时序图和规范

图 26-3： 外部时钟时序 （除 PLL 之外的所有模式）

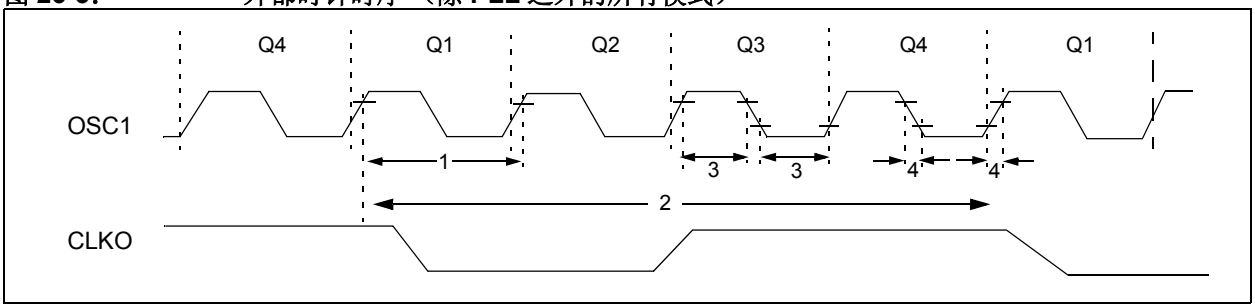


表 26-6： 外部时钟时序要求

参数编号	符号	特性	最小值	最大值	单位	条件
1A	FOSC	外部 CLKI 频率 (1) 振荡器频率 (1)	DC DC	40 40	MHz MHz	HS 振荡器模式 HS 振荡器模式
1	TOSC	外部 CLKI 周期 (1) 振荡器周期 (1)	25 25	— 250	ns ns	HS 振荡器模式 HS 振荡器模式
2	TCY	指令周期时间 (1)	100	—	ns	TCY = 4/FOSC，工业级
3	TOSL，TOSH	外部时钟输入 （OSC1）高电平或低电平时间	10	—	ns	HS 振荡器模式
4	TOSR，TOSF	外部时钟输入 （OSC1）上升或下降时间	—	7.5	ns	HS 振荡器模式

注 1： 对于除 PLL 外的所有配置，指令周期时间 （TCY）等于输入振荡器时基周期的 4 倍。所有规范值均基于器件在标准工作条件下执行代码所对应的特定振荡器类型的特征数据。超过这些规范值可能导致振荡器运行不稳定和 / 或电流消耗超出预期。所有器件在测试 “最小值”时，都在 OSC1/CLKI 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 “最大值”周期时限为 “DC”（没有时钟）。

表 26-7： PLL 时钟时序规范 （VDD = 2.15V 至 3.6V）

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	—	10	MHz	仅 HS 模式
F11	FSYS	片上 VCO 系统频率	16	—	40	MHz	仅 HS 模式
F12	t_{rc}	PLL 起振时间 （锁定时间）	—	—	2	ms	
F13	ΔCLK	CLKO 稳定性 （抗抖动）	-2	—	+2	%	

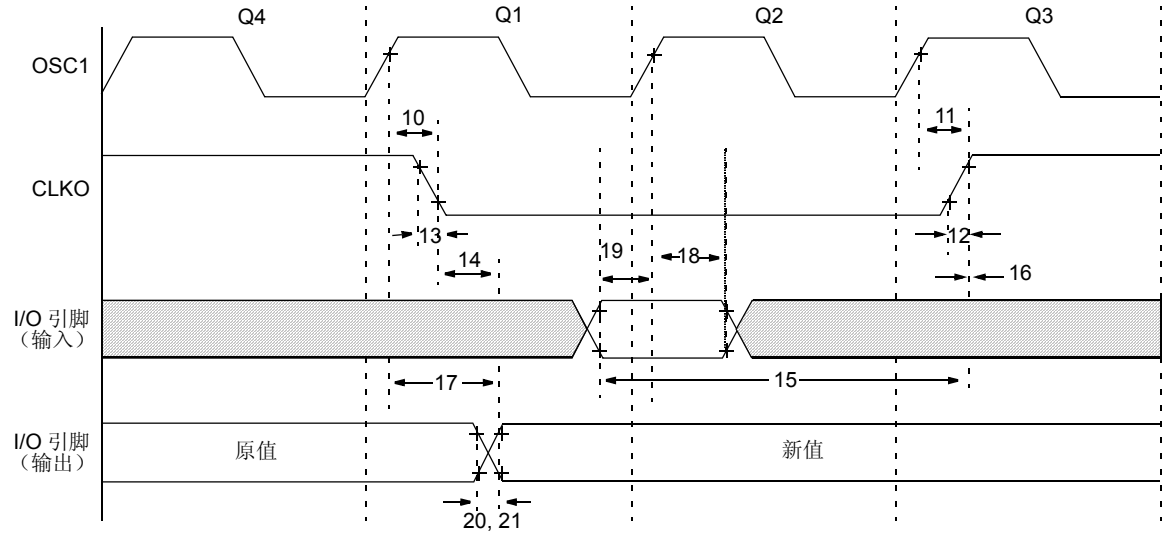
† 除非另行声明，否则 “典型值”栏中的数据均为 5.0V， 25°C 条件下的值。这些参数仅供设计参考，未经测试。

表 26-8： 交流规范： 内部 RC 精度
PIC18F87J10 系列 （工业级）

参数编号	特性	最小值	典型值	最大值	单位	条件
	频率为 31 kHz 时的 INTRC 精度 (1)	26.35	—	35.65	kHz	-40°C 到 +85°C， VDD = 2.0-3.3V

注 1： 校准后的 INTRC 频率。 INTRC 频率随 VDD 的改变而改变。

图 26-4：　　CLKO 和 I/O 时序



注：　负载条件请参考图 26-2。

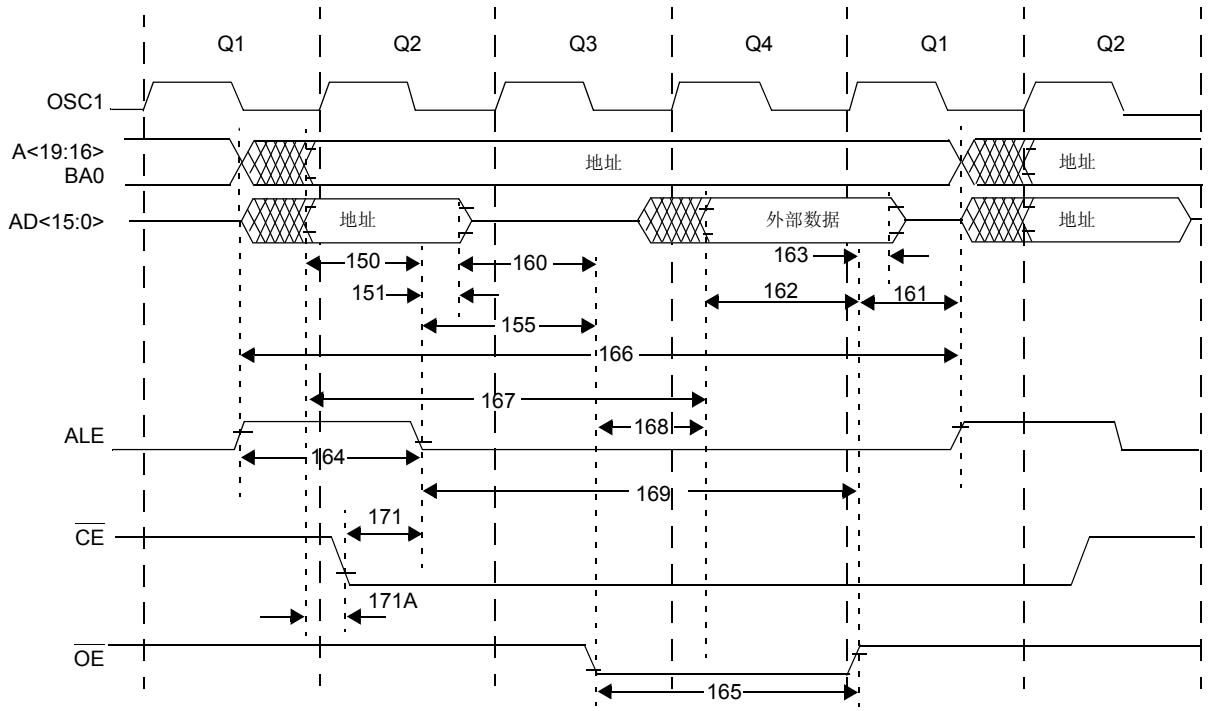
表 26-9：　　CLKO 和 I/O 时序要求

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	—	35	100	ns	（注 1）
13	TCKF	CLKO 下降时间	—	35	100	ns	（注 1）
14	TCKL2IOV	CLKO ↓ 到端口输出有效的时间	—	—	0.5 TCY + 20	ns	
15	TIOV2CKH	在出现 CLKO ↑ 前端口输入有效的时间	0.25 TCY +25	—	—	ns	
16	TCKH2IOI	在 CLKO ↑ 后保持端口输入的时间	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	
18A			200	—	—	ns	VDD = 2.0V
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	
20A			—	—	60	ns	VDD = 2.0V
21	TIOF	端口输出下降时间	—	10	25	ns	
21A			—	—	60	ns	VDD = 2.0V
22†	TINP	INT 引脚高电平或低电平时间	TCY	—	—	ns	
23†	TRBP	RB7:RB4 端口引脚电平变化中断的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： 在 RC 模式下测量的，其中 CLKO 输出为 4xTosc。

图 26-5：　　读程序存储器时序图

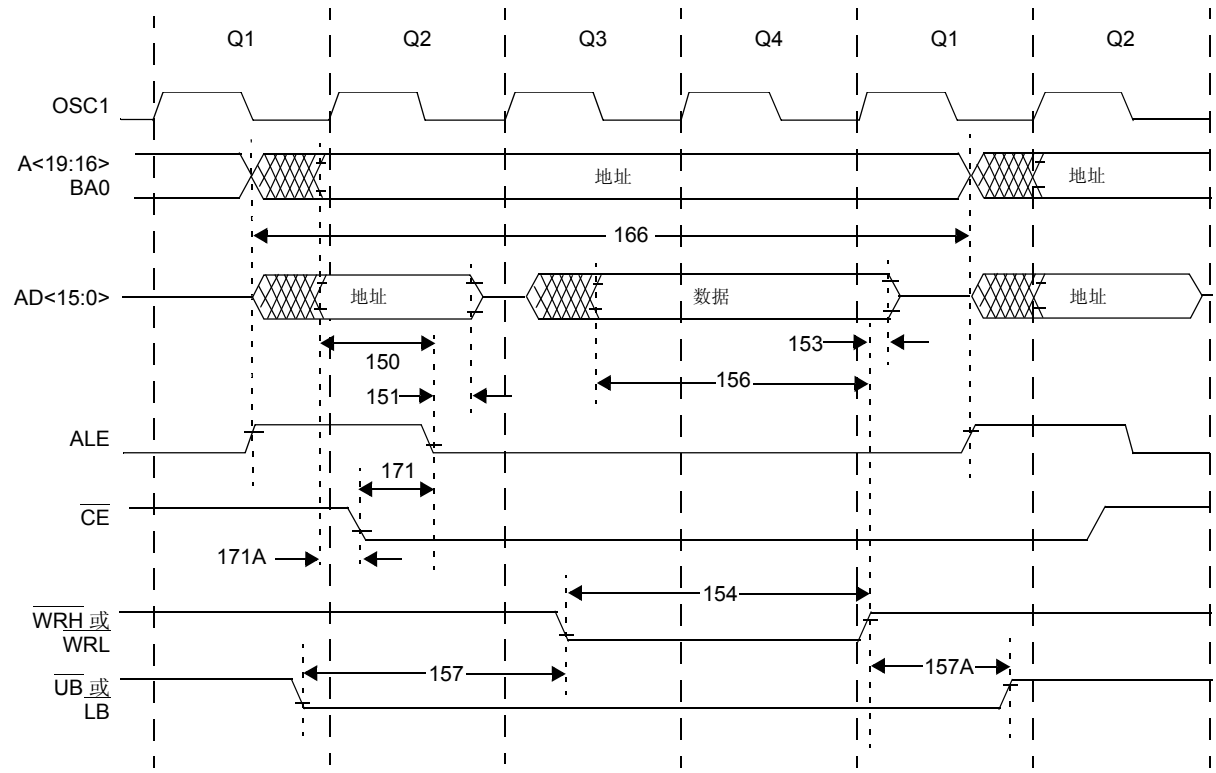


工作条件：2.0V < VCC < 3.6V，-40°C < TA < +125°C（除非另行声明）。

表 26-10：　　CLKO 和 I/O 时序要求

参数编号	符号	特性	最小值	典型值	最大值	单位
150	TadV2alL	地址输出有效到 ALE ↓ 的时间（地址建立时间）	.25 TCY – 10	—	—	ns
151	TalL2adI	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	.125 TCY	—	ns
160	TadZ2oeL	AD 高阻态到 $\overline{OE}$ ↓ 的时间（总线释放 $\overline{OE}$）	0	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑到驱动 AD 的时间	0.125 TCY – 5	—	—	ns
162	TadV2oeH	在 $\overline{OE}$ ↑ 前LS数据有效的时间（数据建立时间）	20	—	—	ns
163	ToeH2adI	$\overline{OE}$ ↑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	—	0.25TCY	—	ns
165	ToeL2oeH	$\overline{OE}$ 脉冲宽度	0.5 TCY – 5	.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 26-6：　　程序存储器写时序图

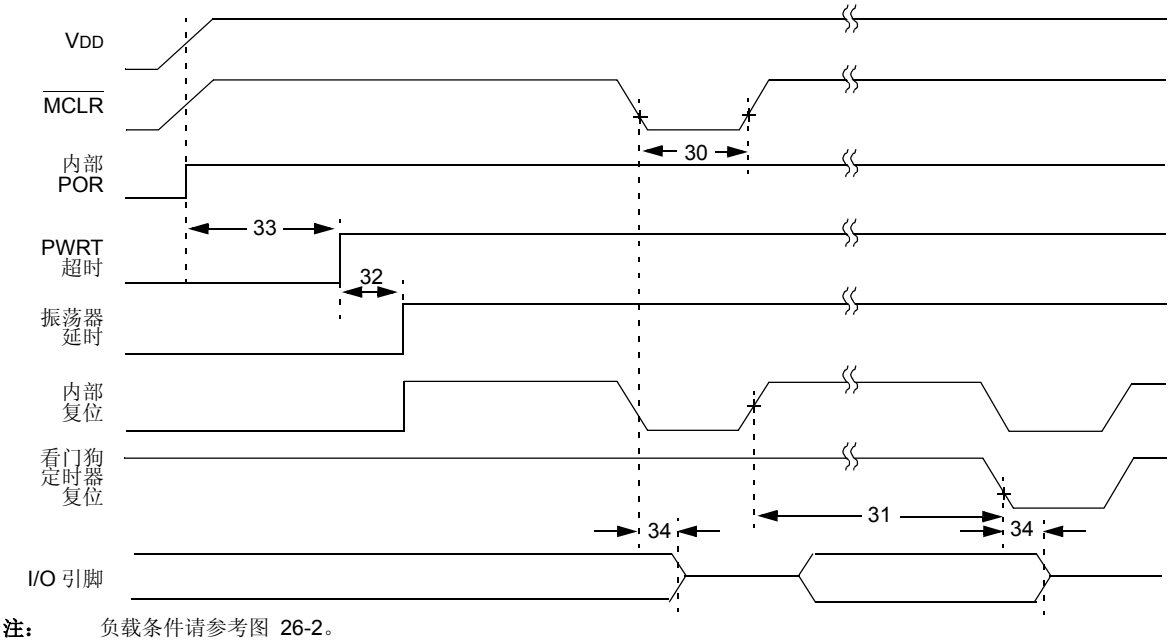


工作条件：2.0V < VCC < 3.6V，-40°C < TA < +125°C（除非另行声明）。

表 26-11：　　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 $\overline{WRn}$ ↑ 之前的数据有效时间 （数据建立时间）	0.5 TCY – 10	—	—	ns
157	TbsV2wrL	在 $\overline{WRn}$ ↓ 之前的字节选择有效时间（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

图 26-7：　　复位、看门狗定时器、振荡器起振定时器和上电延时定时器时序

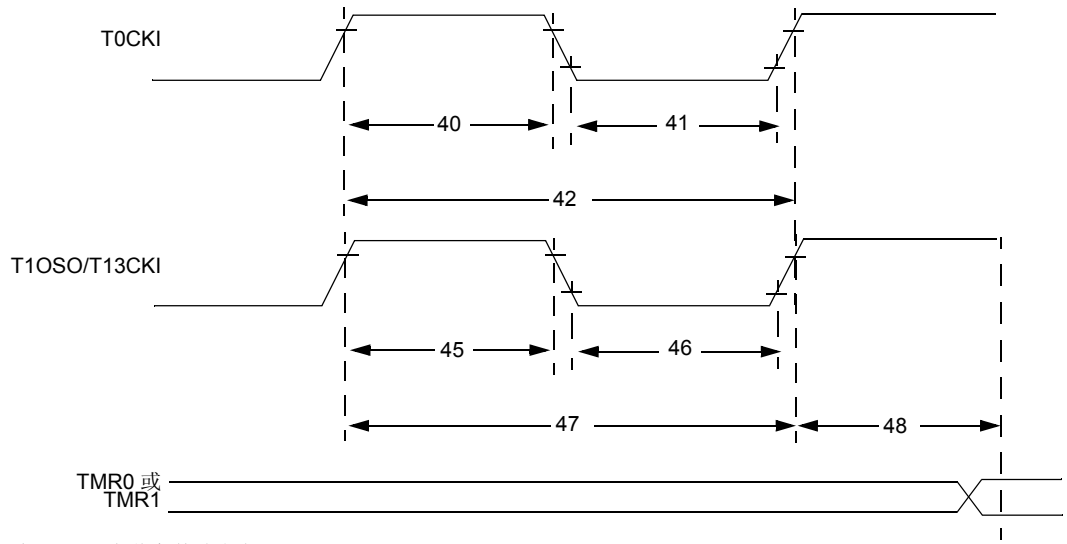


注：　负载条件请参考图 26-2。

表 26-12：　　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）	3.4	4.0	4.6	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	55.6	65.5	75	ms	
34	TIOZ	$\overline{\text{MCLR}}$ 低电平或看门狗定时器复位引起的 I/O 高阻态时间	—	2	—	µs	
38	TCSD	CPU 的启动时间	—	200	—	µs	
39	TIOBST	INTOSC 稳定时间	—	1	—	µs	

图 26-8：　　TIMER0 和 TIMER1 外部时钟时序

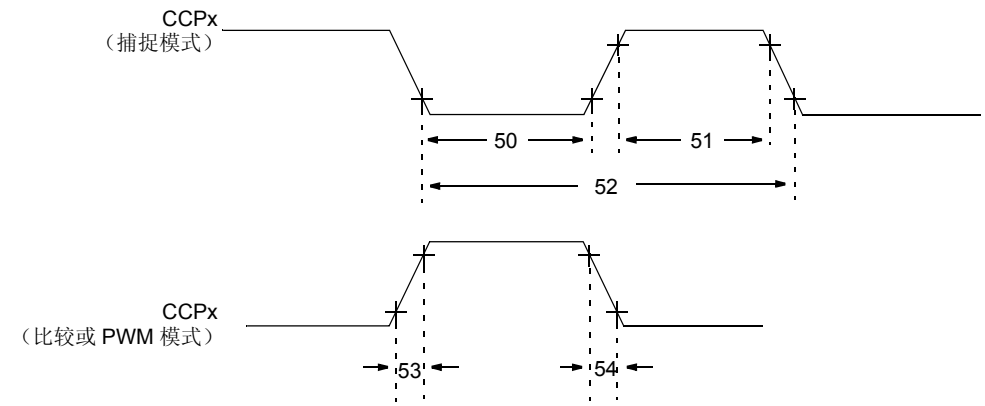


注：　负载条件请参考图 26-2。

表 26-13：　　TIMER0 和 TIMER1 外部时钟要求

参数编号	符号	特性			最小值	最大值	单位	条件
40	TT0H	T0CKI 高电平脉冲宽度		无预分频器	0.5 TCY + 20	—	ns	
				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	取以下两者中的较大值：20 ns 或（TCY + 40）/N	—	ns	N = 预分频值（1，2，4，...，256）
45	TT1H	T13CKI 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	T13CKI 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	T13CKI 输入周期	同步		取以下两者中的较大值：20 ns 或（TCY + 40）/N	—	ns	N = 预分频值（1，2，4，8）
			异步		60	—	ns	
	FT1	T13CKI 振荡器输入频率范围			DC	50	kHz	
48	TCKE2TMRI	从出现外部 T13CKI 时钟沿到定时器递增的延时			2 TOSC	7 TOSC	—	

图 26-9： 捕捉 / 比较 /PWM 时序 （包括 ECCP 模块）



注： 负载条件请参考图 26-2。

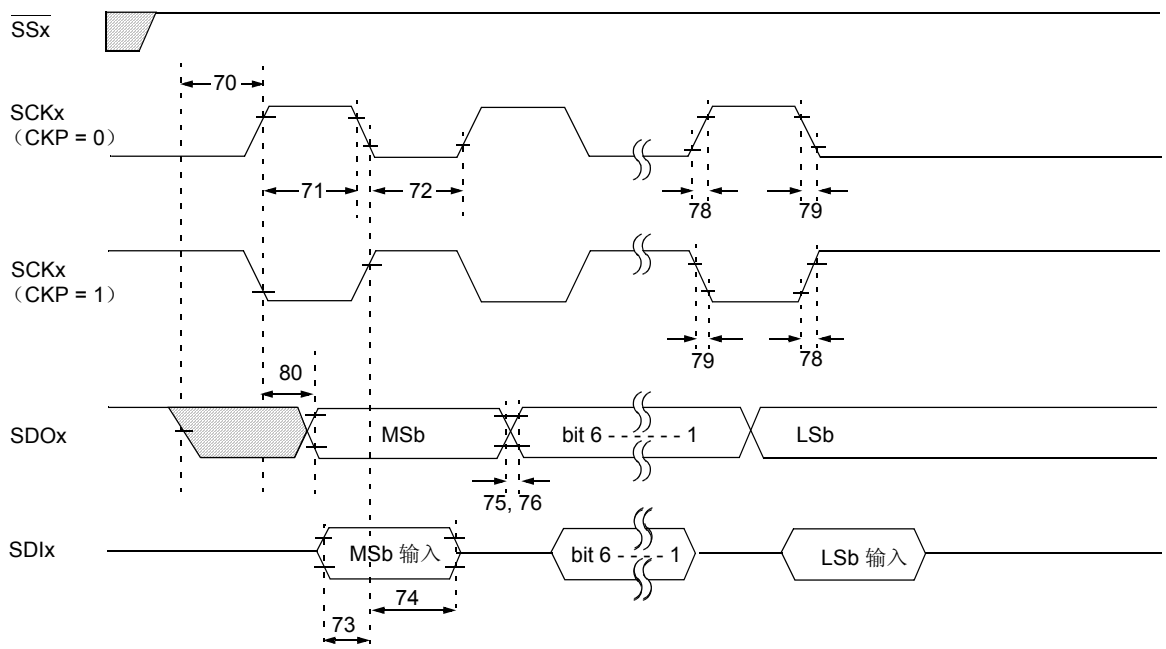
表 26-14： 捕捉 / 比较 /PWM 要求 （包括 ECCP 模块）

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 输出下降时间		—	25	ns	
54	TccF	CCPx 输出下降时间		—	25	ns	

表 26-15： 并行从动端口要求

参数编号	符号	特性	最小值	最大值	单位	条件
62	TdtV2wrH	在 $\overline{WR}$ ↑ 或 $\overline{CS}$ ↑ 之前的数据输入有效时间（建立时间）	20	—	ns	
63	TwrH2dtI	$\overline{WR}$ ↑ 或 $\overline{CS}$ ↑ 到数据输入无效的时间 （保持时间）	20	—	ns	
64	TrdL2dtV	$\overline{RD}$ ↓ 和 $\overline{CS}$ ↓ 到数据输出有效的时间	—	80	ns	
65	TrdH2dtI	$\overline{RD}$ ↑ 或 $\overline{CS}$ ↓ 到数据输出无效的时间	10	30	ns	
66	TibfINH	禁止 IBF 标志位被 $\overline{WR}$ ↑ 或 $\overline{CS}$ ↑ 清零	—	3 TCY		

图 26-10： SPI™ 主控模式时序示例 （CKE = 0）



注： 负载条件请参考图 26-2。

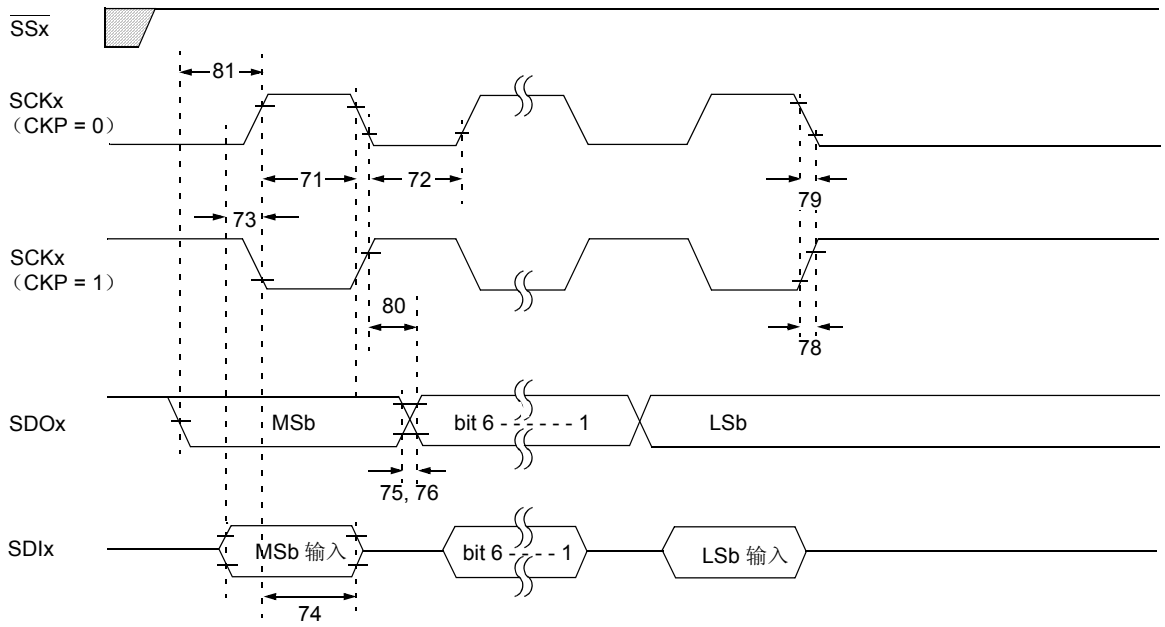
表 26-16： SPI™ 模式要求示例 （主控模式， CKE = 0）

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		TCY	—	ns	
71	TscH	SCKx 输入高电平时间（从动模式）	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（从动模式）	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		100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		100	—	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 （主控模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	在 SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	

注 1： 要求使用参数 #73A。

2： 仅当使用参数 #71A 和 #72A 时。

图 26-11： SPI™ 主控模式时序示例 （CKE = 1）



注： 负载条件请参考图 26-2。

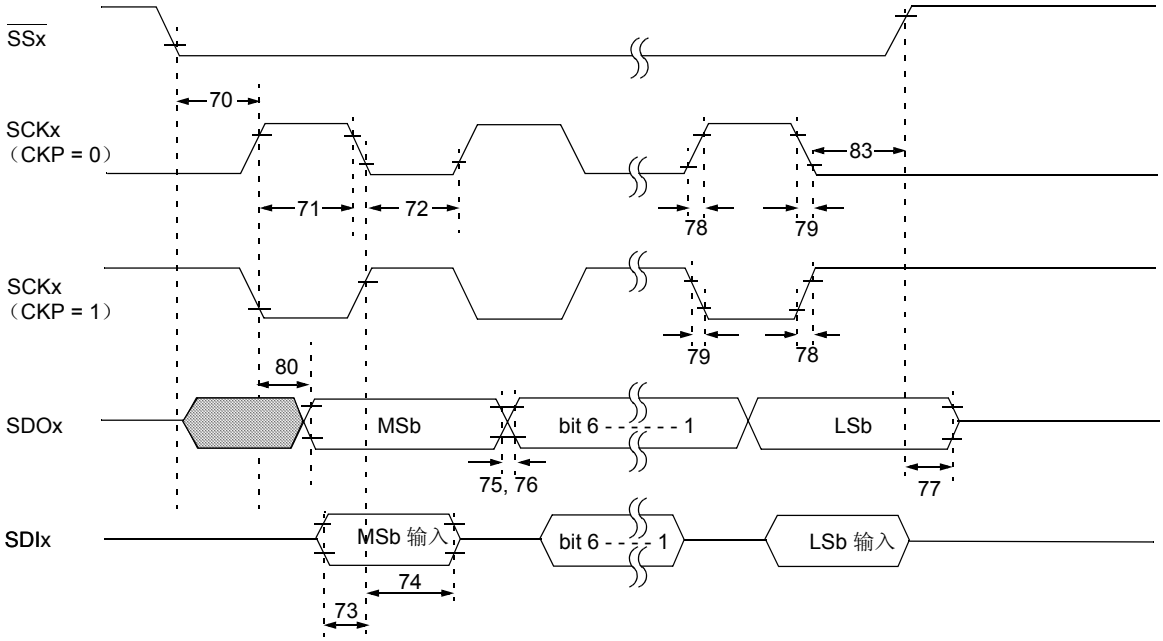
表 26-17： SPI™ 模式要求示例 （主控模式， CKE = 1）

参数编号	符号	特性		最小值	最大值	单位	条件
71	TscH	SCKx 输入高电平时间（从动模式）	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（从动模式）	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		100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 Tcy +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		100	—	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 （主控模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	在 SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
81	TdoV2scH，TdoV2scL	SDOx 数据输出建立到出现 SCKx 边沿的时间		Tcy	—	ns	

注 1： 要求使用参数 #73A。

2： 仅当使用参数 #71A 和 #72A 时。

图 26-12： SPI™ 从动模式时序示例 （CKE = 0）



注： 负载条件请参考图 26-2。

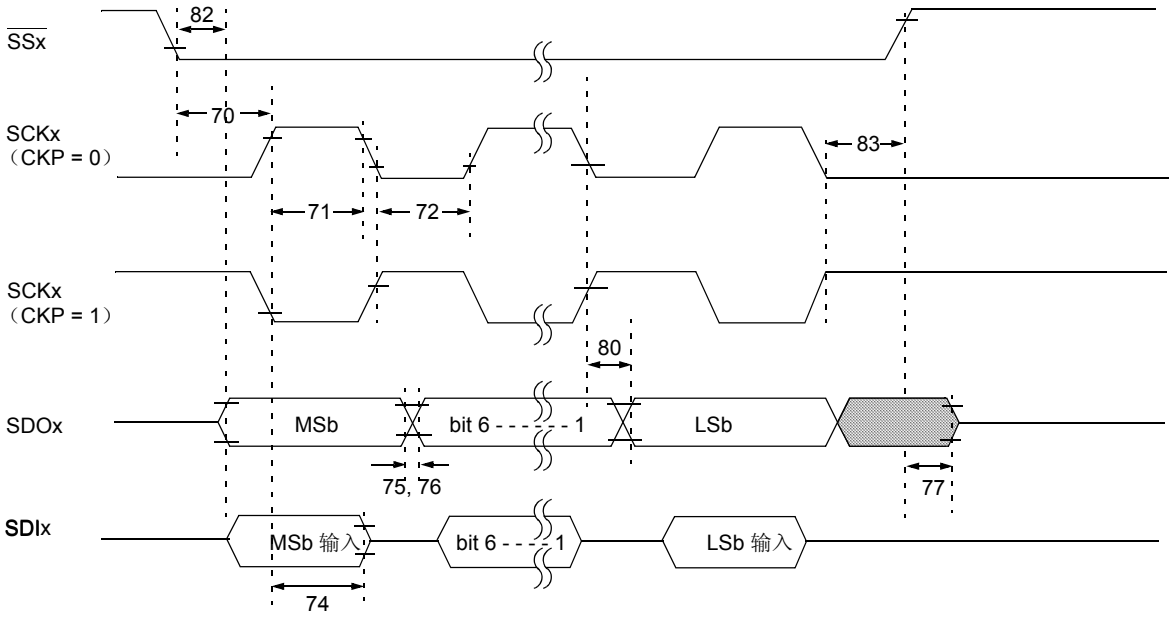
表 26-18： SPI™ 模式要求示例 （从动模式时序， CKE = 0）

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH，TssL2scL	SSx ↓ 到 SCKx ↓ 或 SCKx ↑ 输入的时间		TCY	—	ns	
71	TscH	SCKx 输入高电平时间	连续	1.25 TCY + 30	—	ns	
71A		（从动模式）	单字节	40	—	ns	（注 1）
72	TscL	SCKx 输入低电平时间	连续	1.25 TCY + 30	—	ns	
72A		（从动模式）	单字节	40	—	ns	（注 1）
73	TdiV2scH，TdiV2scL	SDIx 数据输入到 SCKx 边沿的建立时间		100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		100	—	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	SSx ↑ 到 SDOx 输出高阻态的时间		10	50	ns	
78	TscR	SCKx 输出上升时间 （主控模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	在 SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
83	TscH2ssH，TscL2ssH	在 SCKx 边沿后出现 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 26-13： SPI™ 从动模式时序示例 （CKE = 1）



注： 负载条件请参考图 26-2。

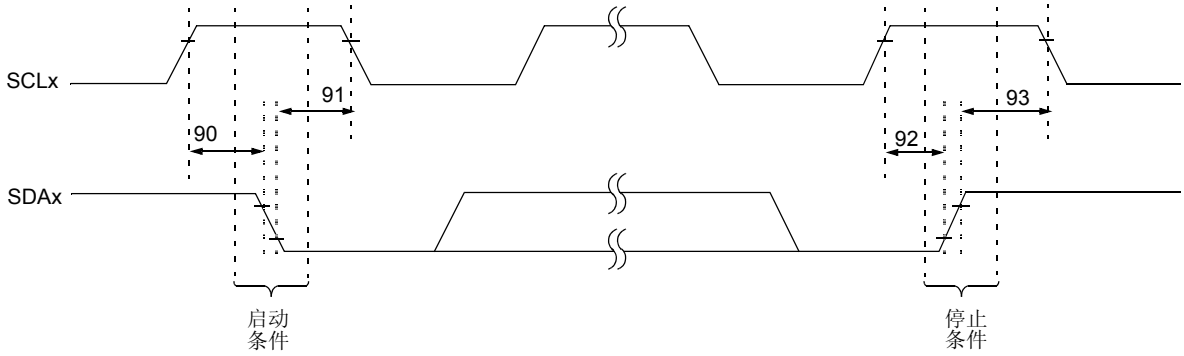
表 26-19： SPI™ 从动模式要求示例 （CKE = 1）

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		TCY	—	ns	
71	TscH	SCKx 输入高电平时间	连续	1.25 TCY + 30	—	ns	
71A		（从动模式）	单字节	40	—	ns	（注 1）
72	TscL	SCKx 输入低电平时间	连续	1.25 TCY + 30	—	ns	
72A		（从动模式）	单字节	40	—	ns	（注 1）
73A	TB2B	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY +40	—	ns	（注 2）
74	TscH2diL，TscL2diL	SDIx 数据输入到 SCKx 边沿的保持时间		100	—	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 （主控模式）		—	25	ns	
79	TscF	SCKx 输出下降时间 （主控模式）		—	25	ns	
80	TscH2doV，TscL2doV	在 SCKx 边沿之后 SDOx 数据输出有效的时间		—	50	ns	
82	TssL2doV	在 $\overline{SSx}$ ↓ 之后 SDOx 数据输出有效的时间		—	50	ns	
83	TscH2ssH，TscL2ssH	在 SCKx 边沿后出现 $\overline{SSx}$ ↑ 的时间		1.5 TCY +40	—	ns	

注 1： 要求使用参数 #73A。

2： 仅当使用参数 #71A 和 #72A 时。

图 26-14：　　I²C™ 总线启动 / 停止位时序

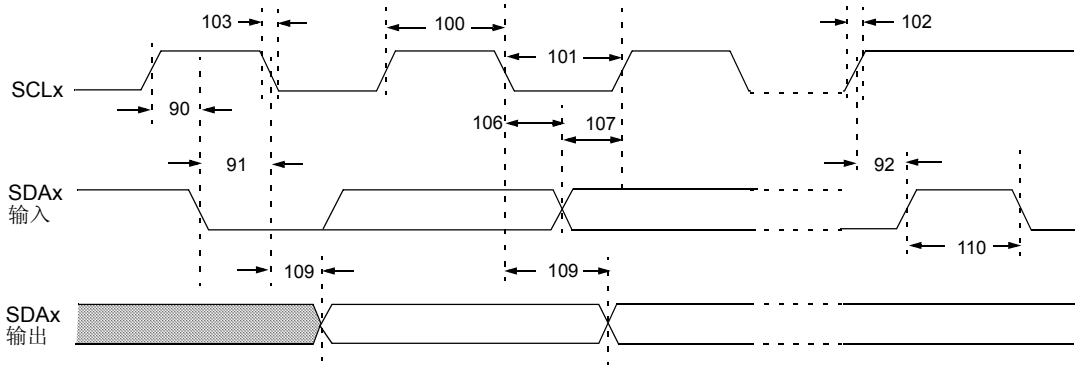


注：　负载条件请参考图 26-2。

表 26-20：　　I²C™ 总线启动 / 停止位要求 （从动模式）

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	100 kHz 模式	4700	—	ns	仅与重复启动条件相关
		建立时间	400 kHz 模式	600	—		
91	THD:STA	启动条件	100 kHz 模式	4000	—	ns	这个周期后产生第一个时钟脉冲
		保持时间	400 kHz 模式	600	—		
92	TSU:STO	停止条件	100 kHz 模式	4700	—	ns	
		建立时间	400 kHz 模式	600	—		
93	THD:STO	停止条件	100 kHz 模式	4000	—	ns	
		保持时间	400 kHz 模式	600	—		

图 26-15：　　I²C™ 总线数据时序



注：　负载条件请参考图 26-2。

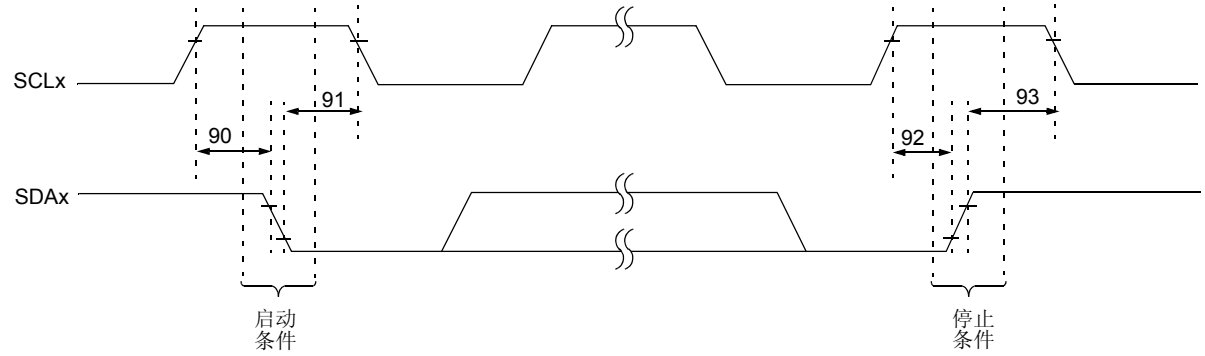
表 26-21： I²C™ 总线数据要求 （从动模式）

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	PIC18F87J10 的工作频率不得低于 10 MHz
			MSSP 模块	1.5 TCY	—		
102	TR	SDAx 和 SCLx 的上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值被规定在 10 到 400 pF 之间
103	TF	SDAx 和 SCLx 的下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值被规定在 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	总线的容性负载		—	400	pF	

注 1： 为避免产生意外的启动或停止条件，作为发送器的器件必须提供这个内部最小延时以覆盖 SCL 下降沿的未定义区域（最小值 300ns）。

2： 快速模式的 I²C 总线器件也可在标准模式的 I²C™ 总线系统中使用，但必须满足 TSU:DAT ≥ 250 ns 的要求。如果快速模式器件没有延长 SCLx 信号的低电平周期，则必然满足此条件。如果该器件延长了 SCLx 信号的低电平周期，其下一个数据位必须输出到 SDAx 线。 SCL 线被释放前，根据标准模式 I²C 总线规范， TRmax.+TSU:DAT = 1000 + 250 = 1250 ns。

图 26-16：　　主控 SSP I²C™ 总线启动 / 停止位时序波形



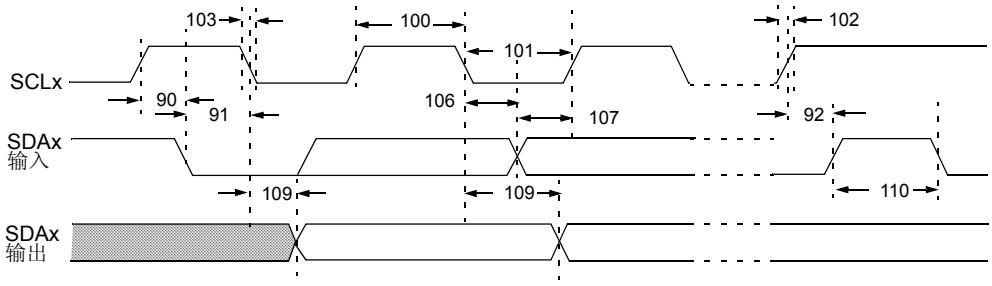
注： 负载条件请参考图 26-2。

表 26-22：　　主控 SSP 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	仅与重复启动条件有关
			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，最小引脚电容均为 10 pF。

图 26-17：　　MSSP I²C™ 总线数据时序



注： 负载条件请参考图 26-2。

表 26-23： MSSP I²C™ 总线数据要求

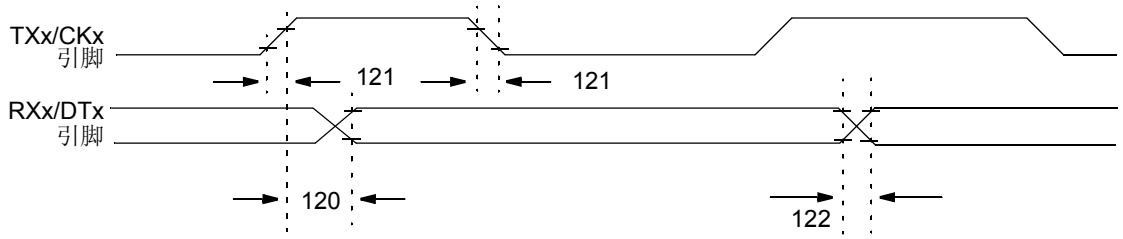
参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
101	TLOW	时钟低电平时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
102	TR	SDAx 和 SCLx 上升时间	100 kHz 模式	—	1000	ns	CB 值被规定在 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 值被规定在 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)	—	ms	仅与重复启动条件有关
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
91	THD:STA	启动条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	ms	
			1 MHz 模式 (1)	TBD	—	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)	TBD	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1)	—	ms	
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	—	ms	在启动新的传输前总线必须保持空闲的时间
			400 kHz 模式	1.3	—	ms	
			1 MHz 模式 (1)	TBD	—	ms	
D102	CB	总线的容性负载		—	400	pF	

图注： TBD = 待定

注 1： 对于所有 I²C™ 引脚，最小引脚电容均为 10 pF。

2： 快速模式的 I²C 总线器件可用于标准模式的 I²C 总线系统中，但必须满足参数 #107 ≥ 250 ns 的要求。如果快速模式器件没有延长 SCLx 信号的低电平周期，则必然满足此条件。如果该器件延长 SCLx 信号的低电平周期，它必须将下一个数据位输出到 SDAx 线。 SCLx 线被释放前，在 100 kHz 模式下，参数 #102 + 参数 #107 = 1000 + 250 = 1250 ns。

图 26-18：　　EUSART 同步发送 （主控 / 从动）时序

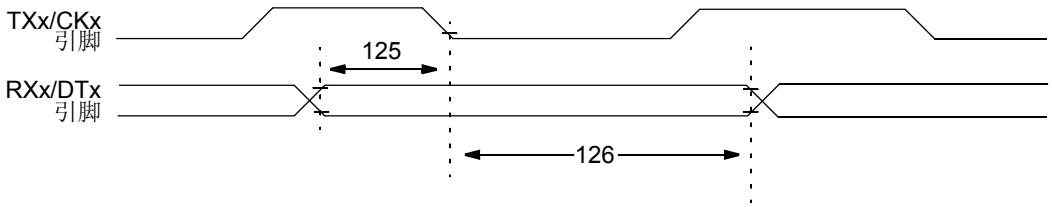


注：　负载条件请参考图 26-2。

表 26-24：　　EUSART 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 XMIT （主控和从动） 时钟高电平到数据输出有效的时间	—	40	ns	
121	TCKRF	时钟输出上升时间和下降时间 （主控模式）	—	20	ns	
122	TDTRF	数据输出上升时间和下降时间	—	20	ns	

图 26-19：　　EUSART 同步接收 （主控 / 从动）时序



注：　负载条件请参考图 26-2。

表 26-25：　　EUSART 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 （主控和从动）在 CKx ↓之前 数据的保持时间 （DTx 保持时间）	10	—	ns	
126	TCKL2DTL	在 CKx ↓之后数据的保持时间 （DTx 保持时间）	15	—	ns	

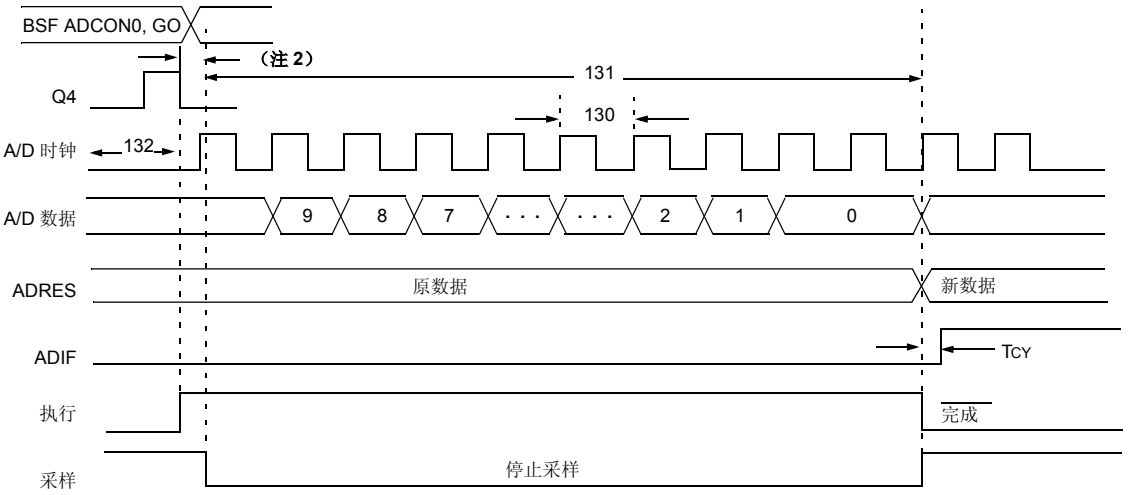
表 26-26： A/D 转换器规范：PIC18F87J10 系列 （工业级）

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	—	—	10	位	DVREF ≥ 3.0V
A03	EIL	积分线性误差	—	—	<±1	LSb	DVREF ≥ 3.0V
A04	EDL	微分线性误差	—	—	<±1	LSb	DVREF ≥ 3.0V
A06	EOFF	偏移误差	—	—	<±3	LSb	DVREF ≥ 3.0V
A07	EGN	增益误差	—	—	<±3	LSb	DVREF ≥ 3.0V
A10	—	单调性	保证 (1)			—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	参考电压范围 (VREFH - VREFL)	2.0 3	— —	— —	V V	VDD < 3.0V VDD ≥ 3.0V
A21	VREFH	参考电压高电平	VSS	—	VREFH	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	建议的模拟电压源阻抗	—	—	2.5	kΩ	
A50	IREF	VREF 输入电流 (2)	— —	— —	5 150	μA μA	在采集 VAIN 期间。 在 A/D 转换期间。

注 1： A/D 转换结果不会因输入电压的增加而减少，并且不会丢失代码。

2： VREFH 电流来自 RA3/AN3/VREF+ 引脚或 VDD，可以选择两者之一作为 VREFH 源。
VREFL 电流来自 RA2/AN2/VREF- 引脚或 VSS，可以选择两者之一作为 VREFL 源。

图 26-20： A/D 转换时序



注 1： 如果选择 A/D 时钟源作为 RC，在 A/D 时钟开始前要加上一个 TCY 时间。用于执行 SLEEP 指令。

2： 这是最小的 RC 延时 （典型值为 100 nS），在这段时间保持电容与模拟输入端断开。

表 26-27： 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	0.7	25.0(1)	μA	基于 TOSC， VREF ≥ 3.0V
			TBD	1	μA	A/D RC 模式
131	TCNV	转换时间 （不包括采集时间）（注 2）	11	12	TAD	
132	TACQ	采集时间 （注 3）	1.4 TBD	— —	μA μA	-40°C 到 +85°C 0°C 到 +85°C
135	TSWC	从转换→采样的切换时间	—	（注 4）		
TBD	TDIS	放电时间	0.2	—	μA	

图注： TBD = 待定

注 1： A/D 时钟周期的时间取决于器件频率和 TAD 时钟分频器。

2： ADRES 寄存器中的内容可在下一个 TCY 周期读出。

3： 转换完成后当电压满量程变化时 （VDD 至 VSS，或 VSS 至 VDD），保持电容获取一个 “新的”输入电压所需的时间。在输入通道上的源阻抗 （*R*S）为 50Ω。

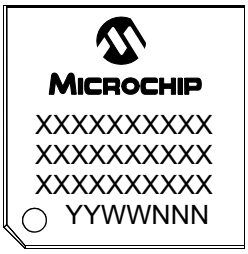
4： 在器件时钟的下一个周期。

注：

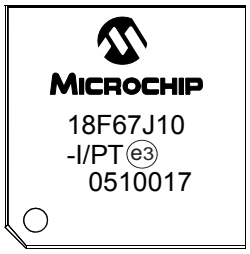
27.0 封装信息

27.1 封装标识信息

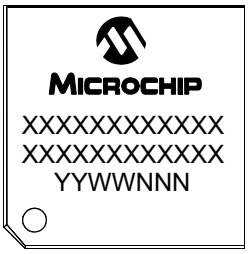
64 引脚 TQFP



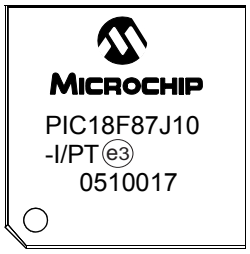
示例



80 引脚 TQFP



示例



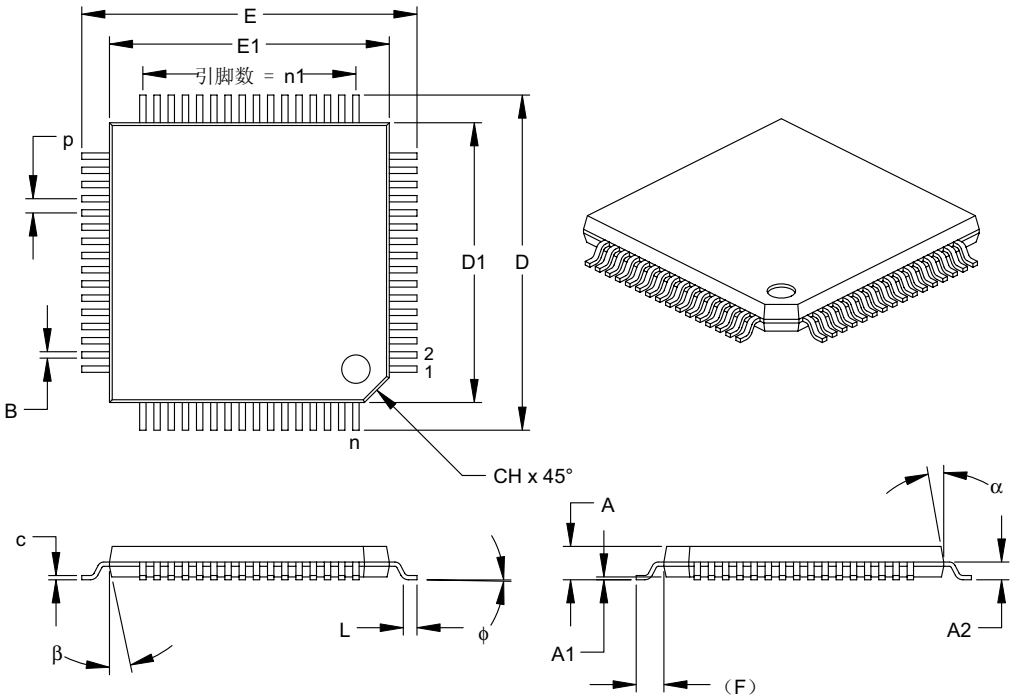
图注：		XX...X客户信息 Y年份代码 （公历年的最后一位数字） YY年份代码 （公历年的最后两位数字） WW星期代码 （1 月 1 日的星期代码为 “01”） NNN以字母数字排序的追踪代码
	e3	无铅 JEDEC 镀锡 （Sn）标识符 *该封装不含铅。无铅 JEDEC 标识符（e3）能在此类封装的外表面上找到。

注： 若所有的 Microchip 器件编号在一行中不能完全标出，它将换行继续，因此会限制客户信息的可用字符数。

27.2 封装详细信息

以下部分将介绍各种封装的技术细节。

64 引脚塑封薄型正方扁平封装 （PT）主体 10x10x1 mm， 1.0/0.10 mm 引脚形式 （TQFP）



	单位	英寸			毫米*		
	尺寸范围	最小值	正常值	最大值	最小值	正常值	最大值
引脚数	n		64			64	
引脚间距	p		.020			0.50	
每侧引脚数	n1		16			16	
总高度	A	.039	.043	.047	1.00	1.10	1.20
塑模封装厚度	A2	.037	.039	.041	0.95	1.00	1.05
悬空间隙	A1	.002	.006	.010	0.05	0.15	0.25
底脚长度	L	.018	.024	.030	0.45	0.60	0.75
引脚投影长度（参考）	(F)		.039			1.00	
底脚倾角	φ	0	3.5	7	0	3.5	7
总宽度	E	.463	.472	.482	11.75	12.00	12.25
总长度	D	.463	.472	.482	11.75	12.00	12.25
塑模封装宽度	E1	.390	.394	.398	9.90	10.00	10.10
塑模封装长度	D1	.390	.394	.398	9.90	10.00	10.10
引脚厚度	c	.005	.007	.009	0.13	0.18	0.23
引脚宽度	B	.007	.009	.011	0.17	0.22	0.27
引脚1切角斜面	CH	.025	.035	.045	0.64	0.89	1.14
塑模顶部锥度	α	5	10	15	5	10	15
塑模底部锥度	β	5	10	15	5	10	15

*控制参数

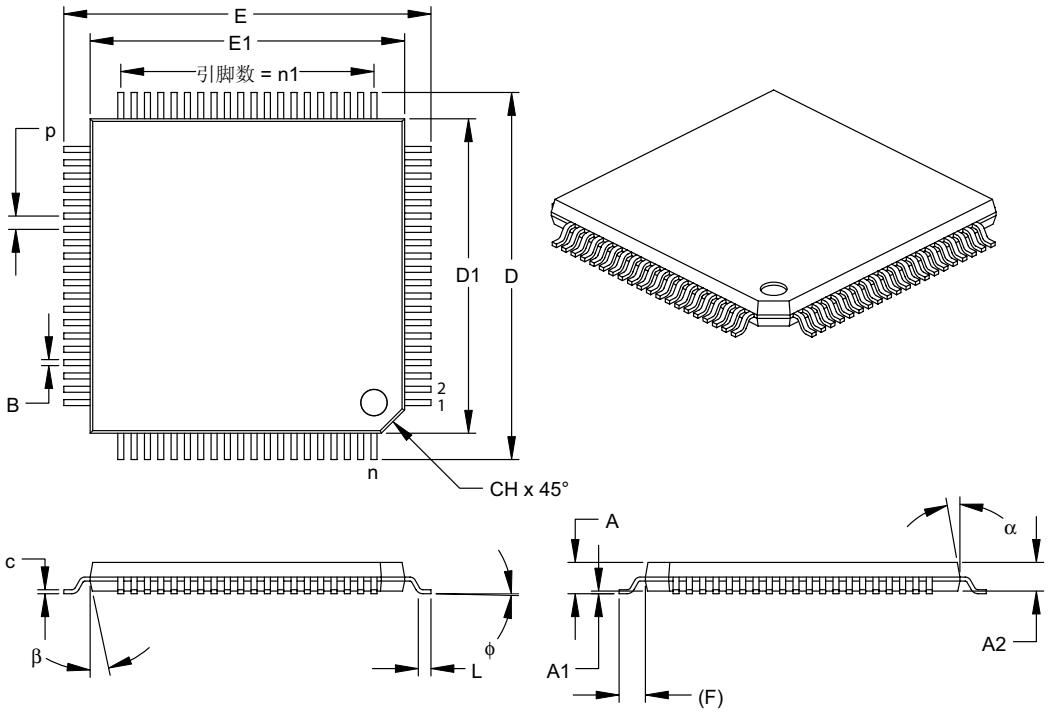
注：

尺寸D1和E1不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过.010英寸（0.254mm）。

等同于JEDEC号：MS-026

图号：C04-085

80 引脚塑封薄型正方扁平封装 （PT）主体 12x12x1 mm， 1.0/0.10 mm 引脚形式 （TQFP）



单位		英寸			毫米*		
尺寸范围		最小值	正常值	最大值	最小值	正常值	最大值
引脚数	n		80			80	
引脚间距	p		.020			0.50	
每侧引脚数	n1		20			20	
总高度	A	.039	.043	.047	1.00	1.10	1.20
塑模封装厚度	A2	.037	.039	.041	0.95	1.00	1.05
悬空间隙	A1	.002	.004	.006	0.05	0.10	0.15
底角长度	L	.018	.024	.030	0.45	0.60	0.75
引脚投影长度	(F)		.039			1.00	
底角倾角	ϕ	0	3.5	7	0	3.5	7
总宽度	E	.541	.551	.561	13.75	14.00	14.25
总长度	D	.541	.551	.561	13.75	14.00	14.25
塑模封装宽度	E1	.463	.472	.482	11.75	12.00	12.25
塑模封装长度	D1	.463	.472	.482	11.75	12.00	12.25
引脚厚度	c	.004	.006	.008	0.09	0.15	0.20
引脚宽度	B	.007	.009	.011	0.17	0.22	0.27
引脚1切角斜面	CH	.025	.035	.045	0.64	0.89	1.14
塑模顶部锥度	α	5	10	15	5	10	15
塑模底部锥度	β	5	10	15	5	10	15

*控制参数

注：

尺寸D1和E1不包括塑模毛边或突起。塑模每侧的毛边或突起不得超过 .010英寸（0.254mm）。

等同于JEDEC号：MS-026

图号：C04-092

注：

附录 A： 在高端器件系列间移植

PIC18F87J10和PIC18F8722系列器件的功能和指标非常相似。然而，当在不同器件系列间移植应用程序时，应注意两者之间一些重要的差异。在表 A-1 中总结了这些差异。本章随后将详细讨论对移植有重大影响的不同之处。

表 A-1： PIC18F8722 和 PIC18F87J10 系列间明显的差异

特性	PIC18F87J10 系列	PIC18F8722 系列
工作频率	40 MHz @ 2.15V	40 MHz @ 4.2V
供电电压	2.0V-3.6V，要求双电源电压	2.0V-5.5V
工作电流	低	更低
程序存储器耐擦写能力	1,000 次擦写周期 （典型值）	100,000 次擦写周期 （典型值）
25 mA 的 I/O 灌 / 拉电流	仅 PORTB 和 PORTC	所有端口
I/O 引脚上的输入电压允差	仅数字引脚上为 5.5V	所有 I/O 引脚上均为 VDD
I/O 端口数	66 （RA7、 RA6、 RE3 和 RF0 不可用）	70
上拉端口	PORTB、 PORTD、 PORTE 和 PORTJ	PORTB
振荡器选项	有限选项 （EC， HS， PLL 和固定为 32 kHz 的 INTRC）	更多选项 （EC， HS， XT， LP， RC， PLL 和频率可变的 INTRC）
程序存储器数据保存期	10 年 （最少）	40 年 （最少）
自写程序存储器	不可用	可用
编程时间 （标称值）	43.8 µs/B （2.8 ms/64 字节块）	15.6 µs/B （1 ms/64 字节块）
编程开始条件	低电压，密钥序列	VPP 和 LVP
代码保护	单个存储区、全部或没有	多个代码保护存储区
配置字	存储在程序存储空间地址最高的 4 个字中	存储在 300000h 开始的配置空间中
从休眠状态唤醒的起振时间	200 µs （典型值）	10 µs （典型值）
上电延时定时器	总是启用	可配置
数据 EEPROM	不可用	可用
BOR	使用稳压器的简单 BOR	可编程 BOR
LVD	不可用	可用
A/D 通道	15	16
A/D 校准	必需	无需
微处理器模式 （EMB）	不可用	可用
外部存储器寻址	可用地址移位	不可用地址移位
在线仿真器	不可用	可用

A.1 电源要求差异

PIC18F87J10和PIC18F8722系列器件间的最大差异是对电源的要求不同。PIC18F87J10 器件设计为用于规模较小的应用；因此它的最大电压较低，泄漏电流较大。

PIC18F87J10 器件的工作电压范围为 2.0V 到 3.6V。此外，这些器件需要两路电源供电：一路用于内核逻辑而另一路用于 I/O。一个 VDD 引脚单独用作内核逻辑供电引脚（VDDCORE）。如**第 26.0 节“电气规范”**所述，该引脚具有特定的电压和电容要求。

A.2 引脚差异

PIC18F87J10和PIC18F8722系列器件在引脚上存在以下一些差异：

- 输入电压允差
- 输出电流的能力
- 可用 I/O 端口

PIC18F87J10 上仅具有数字输入功能的引脚的输入电压最高可达 5.5V，因而可以允许输入电压高于 VDD。**第 10.0 节“I/O 端口”**中表 10-1 列出了完整的 I/O 引脚。

除了输入差异外，还存在着输出差异。PIC18F87J10 器件具有 3 类引脚输出电流能力：高、中和低。并非所有的 I/O 引脚都具有相同的拉 / 灌电流能力。只有 PORTB 和 PORTC 支持 25 mA 的拉 / 灌电流，而 PIC18F8722 的所有输出引脚都可达到该指标。**第 10.0 节“I/O 端口”**中的表 10-2 给出了完整的输出能力列表。

PIC18F87J10器件中引脚功能的使用方式也有所不同。首先，OSC1/OSC2 振荡器引脚严格专用于外部振荡器功能，不可以像 PIC18F8722 器件那样将这些引脚重新分配给 I/O（RA6 或 RA7）。其次，MCLR 引脚只专用于 MCLR，不能配置为输入引脚（RG5）。最后，PIC18F87J10 器件中不存在 RF0 引脚。

在 PIC18F8722 和 PIC18F87J10 器件间转换时应注意所有的这些引脚差异（包括电源引脚的差异）。

A.3 振荡器差异

PIC18F8722器件比PIC18F87J10器件具有更广泛的振荡器选择范围。后者主要支持 HS 和 EC 振荡器的工作模式。

此外，PIC18F87J10 具有一个仅可输出固定的 32 kHz 频率信号的 RC 振荡器。不支持 PIC18F8722 系列中的较高频率的 RC 模式。

两种器件系列都具有内部 PLL。但 PIC18F87J10 系列必须用软件使能 PLL。

当在 PIC18F8722 和 PIC18F87J10 系列器件间转换时应考虑时钟差异。

A.4 外设

当在 PIC18F8722 和 PIC18F87J10 系列器件间转换时还必须考虑外设差异。

- **外部存储总线：**PIC18F87J10 上的外部存储总线不支持单片机模式，但支持外部地址偏移模式。
- **A/D 转换器：**PIC18F87J10 器件上只有 15 路通道。在正常工作前还需要对转换器进行校准。
- **数据 EEPROM：**PIC18F87J10 器件没有这一模块。
- **BOR：**PIC18F87J10 器件不支持可编程 BOR，而是通过使用内部稳压器提供简单的欠压复位功能。
- **LVD：**PIC18F87J10 器件没有这一模块。

索引

A

A/D 257
A/D 转换器中断，配置 261
ADCAL 位 265
ADCON0 寄存器 257
ADCON1 寄存器 257
ADCON2 寄存器 257
ADRESH 寄存器 ..... 257，260
ADRESL 寄存器 257
采集要求 262
计算所需要的最小采集时间 262
模拟端口引脚 144
配置模块 261
配置模拟端口引脚 263
使用 ECCP2 触发信号 264
特殊事件触发器 （ECCP） 176
特殊事件触发信号 （ECCP） 264
相关的寄存器 265
校准 265
在功耗管理模式下的操作 265
转换 264
转换器特性 376
转换时钟 （TAD） 263
转换要求 377
转换状态 （GO/DONE 位） 260
自动采集时间 263
ACKSTAT 224
ACKSTAT 状态标志位 224
ADCAL 位 265
ADCON0 寄存器 257
GO/DONE 位 260
ADCON1 寄存器 257
ADCON2 寄存器 257
ADDFSR 332
ADDLW 295
ADDWF 295
ADDWFC 296
ADDULNK 332
ADRESH 寄存器 257
ADRESL 寄存器 ..... 257，260
ANDLW 296
ANDWF 297

B

BC 297
BCF 298
BF 224
BF 状态标志位 224
BN 298
BNC 299
BNN 299
BNOV 300
BNZ 300
BOR。*参见*欠压复位。
BOV 303
BRA 301
BRG。*参见*波特率发生器。
BSF 301
BTFSC 302
BTFSS 302
BTG 303
BZ 304
比较器 267
参考
内部信号 269
外部信号 269
参考电压 269
复位的影响 270
工作原理 269
模拟输入连接注意事项 271
配置 268
输出 269
相关的寄存器 271
响应时间 269
在休眠模式下的工作原理 270
中断 270
比较器参考电压
复位影响 274
精度和误差 274
连接注意事项 274
配置 273
在休眠模式下的工作方式 274
比较器参考电压模块 273
相关寄存器 275
比较器规范 356
比较 （CCP 模块） 168
CCPRx 寄存器 168
定时器 1/ 定时器 3 模式选择 168
软件中断 168
相关的寄存器 169
引脚配置 168
比较 （ECCP 模块） 176
特殊事件触发器 176
特殊事件触发信号 ..... 161，264
编程，器件指令 289
变更通知客户服务 395
表指针操作 （表） 84
并行从动端口 （PSP） 144
RE0/RD 引脚 144
RE1/WR 引脚 144
RE2/CS 引脚 144
选择 （PSPMODE 位） 144
并行从动端口 （PSP） 144
PORTD 144
相关的寄存器 146
波特率发生器 220
捕捉 / 比较 /PWM （CCP） 165
比较模式。*参见*比较。
捕捉模式。*参见*捕捉。
CCP 模式与定时器资源 166
CCPRxH 寄存器 166
CCPRxL 寄存器 166
定时器互连配置 166
模块配置 166
捕捉 （CCP 模块） 167
CCPRxH:CCPRxL 寄存器 167
CCP 引脚配置 167
定时器 1/ 定时器 3 模式选择 167
软件中断 167
相关的寄存器 169
预分频器 167
捕捉 （ECCP 模块） 176

C

CALL 304
CALLW 333
C 编译器
MPLAB C18 340
MPLAB C30 340

CLRF 305
CLRWDT 305
COMF 306
CPFSEQ 306
CPFSGT 307
CPFSLT 307
CPU 的特殊功能 277
参考电压规范 356
程序存储器
  查找表 61
  存储器映射图 55
    模式 58
    硬编码向量和配置字 56
  复位向量 56
  扩展的指令集 77
  模式 57
    存储器访问 （表） 58
    单片机 57
    扩展单片机 57
    扩展单片机 （地址移位） 58
  闪存配置字 56
  硬件编码向量 56
  指令 63
    双字 63
  中断向量 56
程序计数器 59
  PCLATH 和 PCLATU 寄存器 59
  PCL、 PCH 和 PCU 寄存器 59
程序校验和代码保护 288
串行时钟， SCKx 189
串行数据输出 （SDOx） 189
串行数据输入 （SDIx） 189
串行外设接口。*参见* SPI 模式。
从动选择 （SSx） 189
存储器编程要求 355
存储器构成 55
  程序存储器 55
  数据存储器 64

D

DAW 308
DCFSNZ 309
DECF 308
DECFSZ 309
代码保护 277
代码示例
  16 × 16 无符号乘法程序 104
  16 × 16 有符号乘法程序 104
  8 × 8 无符号乘法程序 103
  8 × 8 有符号乘法程序 103
  擦除闪存程序存储器一行 86
  初始化 PORTA 122
  初始化 PORTB 124
  初始化 PORTC 127
  初始化 PORTD 130
  初始化 PORTE 133
  初始化 PORTF 136
  初始化 PORTG 138
  初始化 PORTH 140
  初始化 PORTJ 142
  读一个闪存程序存储器字 85
  快速寄存器堆栈 61
  闪存程序存储器 88
  使用间接寻址将 RAM （Bank 1）清零的方法 74
  使用偏移量计算 GOTO 61
  使用 Timer1 中断服务实现实时时钟 155
  在捕捉预分频器之间进行切换 167
  在 RAM 内保存 STATUS、 WREG 和 BSR 寄存器 . 120
  装载 SSP1BUF （SSP1SR）寄存器 192
电气规范 343
定时器 1 151
对标准 PIC 指令的影响 336
堆栈满 / 下溢复位 61

E

ECCP
  标准 PWM 模式 176
  捕捉和比较模式 176
  相关寄存器 188
  增强型 PWM 模式 177
ECCP1 和 ECCP3 对 CCP4 和 CCP5 的使用 174
ENVREG 引脚 284
EUSART
  波特率发生器
    在功耗管理模式下的操作 239
  波特率发生器 （BRG） 239
    波特率误差计算 240
    采样 239
    高波特率选择位 （BRGH 位） 239
    相关的寄存器 240
    异步模式下的波特率 241
    自动波特率检测 243
  同步从动模式 254
    发送 254
    接收 255
    相关寄存器 （发送） 254
    相关寄存器 （接收） 255
  同步主控模式 251
    发送 251
    接收 253
    相关寄存器 （发送） 252
    相关寄存器 （接收） 253
  异步模式 245
    12 位间隔字符的发送和接收 250
    发送器 245
    接收器 247
    设置带有地址检测功能的 9 位模式 247
    同步间隔字符自动唤醒 248
    相关寄存器 （发送） 246
    相关寄存器 （接收） 248

F

FSCM。*参见*故障保护时钟监视器。
返回地址堆栈 59
返回堆栈指针 （STKPTR） 60
访问栈顶 59
封装 379
  标识 379
  详细信息 380
复位 ..... 43， 277
  堆栈满复位 43
  堆栈下溢复位 43
  功耗管理模式下的 MCLR 复位 43
  看门狗定时器 （WDT）复位 43
  MCLR 复位，正常工作 43
  欠压复位 （BOR） ..... 43， 277
  RESET 指令 43
  上电复位 （POR） ..... 43， 277
  上电延迟定时器 （PWRT） 277
  振荡器起振定时器 （OST） 277

G

GOTO 310
功耗管理模式 35
  多条 Sleep 命令 36
  和 EUSART 的工作原理 239
  和 SPI 模式的工作原理 197
  进入 35
  空闲模式 39
    PRI_IDLE 40
    RC_IDLE 41
    SEC_IDLE 40
  时钟转换和状态指示位 36
  退出空闲和休眠模式 41
    通过复位 41
    通过 WDT 超时 41
    通过中断 41
    在没有振荡器起振延迟的情况下 41
  休眠模式 39
  选择 35
  运行模式 36
    PRI_RUN 36
    RC_RUN 38
    SEC_RUN 36
功耗管理模式对各种时钟源的影响 32
公式
  A/D 采集时间 262
  A/D 最小充电时间 262
固件指令 289
故障保护时钟监视器 ..... 277，286
  功耗管理模式下的中断 287
  POR 或从休眠中唤醒 287
  振荡器故障期间的 WDT 286

H

汇编器
  MPASM 汇编器 340

J

I/O 端口 121
  引脚功能 121
I/O 引脚排列说明
  PIC18F6XJ10/6XJ15 10
  PIC18F8XJ10/8XJ15 17
I²C 模式 （MSSP）
  波特率发生器 220
  串行时钟 （RC3/SCKx/SCLx） 206
  从动模式 204
    发送 206
    接收 206
    寻址 204
  读 / 写位信息 （R/W 位） ..... 204，206
  多主机模式 228
  多主机通信、总线冲突与总线仲裁 228
  复位的影响 228
  工作方式 204
  广播呼叫地址支持 217
  I²C 时钟速率和 BRG 220
  寄存器 199
  时钟同步与 CKP 位 214
  时钟延长 213
    10 位从动发送模式 213
    10 位从动接收模式 （SEN = 1） 213
    7 位从动发送模式 213
    7 位从动接收模式 （SEN = 1） 213
  时钟仲裁 221
  停止条件期间的
    总线冲突 ..... 231， 232
  停止条件时序 227
  相关的寄存器 233
  休眠工作方式 228
  主控模式 218
    发送 224
    工作方式 219
    接收 224
    启动条件时序 222
    重复启动条件时序 223
寄存器
  ADCON0 （A/D 控制寄存器 0） 257
  ADCON1 （A/D 控制寄存器 1） 258
  ADCON2 （A/D 控制寄存器 2） 259
  BAUDCONx （波特率控制） 238
  CCPxCON （CCPx 控制， CCP4 和 CCP5） 165
  CCPxCON （ECCPx 控制） 173
  CMCON （比较器控制） 267
  CONFIG1H （配置寄存器 1 的高字节） 279
  CONFIG1L （配置寄存器 1 的低字节） 279
  CONFIG2H （配置寄存器 2 的高字节） 280
  CONFIG2L （配置寄存器 2 的低字节） 280
  CONFIG3H （配置寄存器 3 的高字节） 281
  CONFIG3L （配置寄存器 3 的低字节） 281
  CONFIG3L （配置寄存器 3 低字节） 57
  CVRCON （比较器参考电压控制） 273
  ECCPxAS （ECCPx 自动关闭控制） 185
  ECCPxDEL （PWM 延迟） 184
  EECON1 （EEPROM 控制寄存器 1） 83
  INTCON2 （中断控制寄存器 2） 108
  INTCON3 （中断控制寄存器 3） 109
  INTCON （中断控制寄存器） 107
  IPR1 （外设中断优先级 1） 116
  IPR2 （外设中断优先级 2） 117
  IPR3 （外设中断优先级 3） 118
  MEMCON （外部存储总线控制） 92
  OSCCON （振荡器控制） 32
  OSCTUNE （PLL 控制） 29
  PIE1 （外设中断允许寄存器 1） 113
  PIE2 （外设中断允许寄存器 2） 114
  PIE3 （外设中断允许寄存器 3） 115
  PIR1 （外设中断请求 （标志） 1） 110
  PIR2 （外设中断请求 （标志） 2） 111
  PIR3 （外设中断请求 （标志） 3） 112
  PSPCON （并行从动端口控制） 145
  器件 ID 寄存器 1 282
  器件 ID 寄存器 2 282
  RCON （复位控制） ..... 44， 119
  RCSTAx （接收状态和控制） 237
  SSPxADD （MSSPx 地址） 203
  SSPxCON1 （MSSPx 控制 1， I²C 模式） 201
  SSPxCON1 （MSSPx 控制 1， SPI 模式） 191
  SSPxCON2 （MSSPx 控制 2， I²C 模式） 202
  SSPxSTAT （MSSPx 状态， I²C 模式） 200
  SSPxSTAT （MSSPx 状态， SPI 模式） 190
  STATUS 73
  STKPTR （堆栈指针） 60
  T0CON （Timer0 控制寄存器） 147
  T1CON （Timer1 控制） 151
  T2CON （Timer2 控制） 157
  T3CON （Timer3 控制） 159
  T4CON （Timer4 控制） 163
  TXSTAx （发送状态和控制） 236
  WDTCON （看门狗定时器控制） 283

寄存器文件 67
INCF 310
INCFSZ 311
INFSNZ 311
INTCON 寄存器 107
RBIF 位 124
INTOSC， INTRC。*参见*内部振荡器模块。
IORLW 312
IORWF 312
IPR 寄存器 116
计算 GOTO 61
间隔字符 （12 位）发送和接收 250
间接寻址 75
交流 （时序）规范 357
参数符号 357
时序条件 358
温度和电压规范 358
晶体振荡器 / 陶瓷谐振器 27
绝对最大值 343

K

开发支持 339
看门狗定时器 （WDT） ..... 277， 283
编程注意事项 283
控制寄存器 283
相关的寄存器 283
振荡器故障期间 286
勘误表 4
客户通知服务 395
客户支持 395
快速寄存器堆栈 61
框图
16 位模式 Timer0 148
16 位字节写模式 95
16 位字节选择模式 97
16 位字写模式 96
8 位复用模式 99
8 位模式 Timer0 148
A/D 260
比较模式操作 168
比较器参考电压 274
比较器参考电压输出缓冲示例 275
比较器 I/O 工作模式 268
比较器模拟输入典型电路 271
比较器输出 270
表读操作 81
表写操作 82
表写闪存程序存储器 87
波特率发生器 220
捕捉模式操作 167
单个比较器 269
读闪存程序存储器 85
EUSART 发送 245
EUSART 接收 247
故障保护时钟监视器 286
看门狗定时器 283
MSSP （I²C 模式） 199
MSSP （I²C 主控模式） 218
MSSP （SPI 模式） 189
模拟输入典型电路 261
PIC18F6XJ10/6XJ15 8
PIC18F8XJ10/8XJ15 9
PLL 29
PORTD 和 PORTE （并行从动端口） 144
PWM 操作 （简化） 170
片内稳压器连接 284
片上复位电路 43
器件时钟 30
Timer2 158
Timer3 160
Timer4 164
Timer3 （16 位读 / 写模式） 160
Timer1 152
Timer1 （16 位读 / 写模式） 152
通用 I/O 端口操作 121
外部上电复位电路 （缓慢的 VDD 上电） 45
中断逻辑 106
扩展指令集
ADDFSR 332
ADDULNK 332
CALLW 333
MOVSF 333
MOVSS 334
PUSHL 334
SUBFSR 335
SUBULNK 335

L

LFSR 313
立即数偏移变址寻址
和标准的 PIC18 指令 336
立即数偏移变址寻址模式 336

M

Microchip 因特网网站 395
MOVF 313
MOVFF 314
MOVLB 314
MOVLW 315
MOVSF 333
MOVSS 334
MOVWF 315
MPLAB ASM30 汇编器、链接器和库管理器 340
MPLAB ICD 2 在线调试器 341
MPLAB ICE 2000 高性能通用在线仿真器 341
MPLAB ICE 4000 高性能通用在线仿真器 341
MPLAB PM3 器件编程器 341
MPLAB 集成开发环境软件 339
MPLINK 目标链接器 /MPLIB 目标库管理器 340
MSSP
ACK 脉冲 ..... 204， 206
I²C 模式。*参见* I²C 模式。
控制寄存器 （通用） 189
模块概述 189
SPI 主 / 从器件连接 193
MULLW 316
MULWF 316
脉宽调制。*参见* PWM（CCP 模块）和 PWM（ECCP 模块）。
默认系统时钟 31
模数转换器。*参见* A/D。

N

NEGF 317
NOP 317
内部互联。*参见* I²C。
内部 RC 振荡器
与 WDT 一起使用 283
内部稳压器规范 356
内部振荡器模块 30
内核功能
快速移植 6
扩展的存储器 5

扩展指令集 5
纳瓦技术 5
外部存储总线 5
振荡器选项和功能 5

P

PIC18F8722 和 PIC18F87J10 系列间明显的差异 383
电源要求 384
外设 384
振荡器选项 384
PICSTART Plus 开发编程器 342
PIE 寄存器 113
PIR 寄存器 110
PLL 29
ECPLL 振荡模式 29
HSPLL 振荡模式 29
POP 318
PORTA
LATA 寄存器 122
PORTA 寄存器 122
TRISA 寄存器 122
相关的寄存器 123
PORTB
LATB 寄存器 124
PORTB 寄存器 124
RB7:RB4 电平变化中断标志 （RBIF 位） 124
TRISB 寄存器 124
相关的寄存器 126
PORTC
LATC 寄存器 127
PORTC 寄存器 127
RC3/SCKx/SCLx 引脚 206
TRISC 寄存器 127
相关的寄存器 129
PORTD
LATD 寄存器 130
PORTD 寄存器 130
TRISD 寄存器 130
相关的寄存器 132
PORTE
LATE 寄存器 133
模拟端口引脚 144
PORTE 寄存器 133
PSP 模式选择 （PSPMODE 位） 144
RE0/RD 引脚 144
RE1/WR 引脚 144
RE2/CS 引脚 144
TRISE 寄存器 133
相关的寄存器 135
PORTF
LATF 寄存器 136
PORTF 寄存器 136
TRISF 寄存器 136
相关的寄存器 137
PORTG
LATG 寄存器 138
PORTG 寄存器 138
TRISG 寄存器 138
相关的寄存器 139
PORTH
LATH 寄存器 140
PORTH 寄存器 140
TRISH 寄存器 140
相关的寄存器 141
PORTJ
LATJ 寄存器 142
PORTJ 寄存器 142
TRISJ 寄存器 142
相关的寄存器 143
POR。*参见*上电复位。
PRI_IDLE 模式 40
PRI_RUN 模式 36
PSP。*参见*并行从动端口。
PWM （CCP 模块）
操作设置 171
PR2/PR4 寄存器 170
TMR2 与 PR2 匹配 177
TMR2 （TMR4）与 PR2 （PR4）匹配 170
TMR4 与 PR4 匹配 163
相关的寄存器 172
占空比 170
周期 170
PWM （ECCP 模块） 177
半桥模式 180
半桥输出模式应用示例 180
CCPR1H:CCPR1L 寄存器 177
复位的影响 187
可编程死区延迟 184
PWM 操作的设置 187
启动注意事项 185
全桥模式 181
全桥输出模式下的方向更改 182
全桥应用示例 182
示例频率 / 分辨率 178
输出关系 （低电平有效） 179
输出关系 （高电平有效） 179
输出配置 178
增强型 PWM 自动关闭 184
占空比 178
周期 177
PUSH 318
PUSH 和 POP 指令 60
PUSHL 334
配置寄存器保护 288
配置位 277

Q

器件概述 5
特性 （64 引脚器件） 7
特性 （80 引脚器件） 7
系列中各产品的具体信息 6
Q 时钟 171
欠压复位 （BOR） 45
和片内稳压器 284
检测 45

R

RAM。*参见*数据存储器。
RC_IDLE 模式 41
RC_RUN 模式 38
RCALL 319
RCON 寄存器
初始化期间的位状态 48
RESET 319
RETFIE 320
RETLW 320
RETURN 321
RLCF 321
RLNCF 322
RRCF 322
RRNCF 323
软件模拟器 （MPLAB SIM） 340

S

SCKx 189
SDIx 189
SDOx 189
SEC_IDLE 模式 40
SEC_RUN 模式 36
SETF 323
SLEEP 324
SPI 模式 （MSSP） 189
  串行时钟 189
  串行数据输出 189
  串行数据输入 189
  从动模式 195
  从动选择 189
  从动选择同步 195
  典型连接 193
  复位的影响 197
  工作方式 192
  SPI 时钟 194
  SSPxBUF 寄存器 194
  SSPxSR 寄存器 194
  使能 SPI I/O 193
  时钟速度，模块相互关系 197
  相关的寄存器 198
  在功耗管理模式下的工作方式 197
  主 / 从器件连接 193
  主控模式 194
  总线模式兼容性 197
SSP
  用于时钟移位的 TMR4 输出 164
SSPOV 224
SSPOV 状态标志位 224
SSPxSTAT 寄存器
  R/W 位 ..... 204，206
SSx 189
SWAPF 326
SUBFSR 335
SUBFWB 324
SUBLW 325
SUBWF 325
SUBWFB 326
SUBULNK 335
闪存程序存储器 81
  表读与表写 81
  擦除 86
  擦除操作顺序 86
  代码保护期间的操作 89
  读 85
  控制寄存器 82
    TABLAT 寄存器 （表锁存器） 84
  相关的寄存器 89
  写 87
    写校验 89
    意外终止 89
  写操作顺序 87
闪存程序寄存器
  表指针边界 84
  基于不同操作的表指针边界 84
  控制寄存器
    EECON1 和 EECON2 82
    TBLPTR （表指针）寄存器 84
闪存配置字 277
上电复位 （POR） 45
上电延迟 33
上电延时定时器 （PWRT） ..... 33，46
  延时序列 46
时序图
  A/D 转换 376
  BRG 溢出时序 244
  半桥 PWM 输出 180
  并行从动端口 （PSP） 146
  并行从动端口 （PSP）写 145
  捕捉 / 比较 /PWM （包括 ECCP 模块） 366
  CLKO 和 I/O 361
  程序存储器写 363
  从动同步 195
  从空闲模式唤醒进入运行模式的转换 40
  从 RC_RUN 模式到 PRI_RUN 模式的转换 38
  从 SEC_RUN 模式到 PRI_RUN 模式
    （HSPLL）的转换 37
  从休眠模式唤醒的转换 （HSPLL） 39
  带有时钟仲裁的波特率发生器 221
  到 RC_RUN 模式的转换 38
  第一个启动位时序 222
  读程序存储器 362
  EUSART 同步发送 （主控 / 从动） 375
  EUSART 同步接收 （主控 / 从动） 375
  发送和应答时的总线冲突 228
  发送间隔字符序列 250
  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） 364
  故障保护时钟监视器 287
  缓慢上升时间 （MCLR 连接到 VDD，
    VDD 电压上升时间 > TPWRT） 47
  I2C 从动模式 （10 接收， SEN = 0，
    ADMSK = 01001） 210
  I2C 从动模式 （7 位接收， SEN = 0） 207
  I2C 从动模式 （7 位接收， SEN = 0，
    ADMSK = 01011） 208
  I2C 从动模式广播呼叫地址时序
    （7 位或 10 位地址模式） 217
  I2C 从动模式 （10 位发送） 212
  I2C 从动模式 （10 位接收， SEN = 0） 211
  I2C 从动模式 （10 位接收， SEN = 1） 216
  I2C 从动模式 （7 位发送） 209
  I2C 从动模式 （7 位接收， SEN = 1） 215
  I2C 停止条件接收或发送模式 227
  I2C 应答时序 227
  I2C 主控模式 （7 位或 10 位地址发送） 225
  I2C 主控模式 （7 位接收） 226
  I2C 总线启动 / 停止位 371
  I2C 总线数据 371
  进入空闲模式的转换 40
  进入 SEC_RUN 模式的转换 37
  进入休眠模式的转换 39
  PWM 方向更改 183
  PWM 输出 170
  PWM 自动关闭 （P1RSEN = 0，
    自动重新启动被禁止） 186
  PWM 自动关闭 （P1RSEN = 1，
    使能自动重新启动） 186
  启动条件期间的总线冲突 （仅 SDAx） 229
  启动条件期间的总线冲突 （SCLx = 0） 230
  启动条件期间由 SDAx 仲裁引起的 BRG 复位 230
  全桥 PWM 输出 181
  SPI 从动模式示例 （CKE = 0） 369
  SPI 从动模式示例 （CKE=1） 370
  SPI 模式 （从动模式， CKE = 0） 196
  SPI 模式 （从动模式， CKE = 1） 196
  SPI 模式 （主控模式） 194

SPI 主控模式时序示例 （CKE = 0） 367
SPI 主控模式时序示例 （CKE = 1） 368
上电时的延时序列 （MCLR 连接到 VDD，
VDD 电压上升时间 < TPWRT） 46
上电延时序列 （MCLR 未连接到 VDD）情形 1 46
上电延时序列 （MCLR 未连接到 VDD）情形 2 47
时钟 / 指令周期 62
时钟同步 214
双速启动时钟转换 （从 INTRC 切换到 HSPLL） 285
Timer0 和 Timer1 外部时钟 365
停止条件期间的总线冲突 （情形 1） 232
停止条件期间的总线冲突 （情形 2） 232
同步发送 251
同步发送 （由 TXEN 位控制） 252
同步接收 （主控模式，由 SREN 位控制） 253
外部时钟 （除 PLL 之外的所有模式） 359
休眠模式下的外部存储总线
（扩展单片机模式） ..... 98， 100
休眠模式下的自动唤醒位 （WUE） 249
异步发送 246
异步发送 （背对背） 246
异步接收 248
在接近 100% 占空比时 PWM 更改方向 183
正常工作状态下的自动唤醒位 （WUE） 249
执行 TBLRD 的外部存储总线 （扩展单片机模式） 98
重复启动条件 223
重复启动条件期间的总线冲突 （情形 1） 231
重复启动条件期间的总线冲突 （情形 2） 231
主控 SSP I2C 总线启动 / 停止位 373
主控 SSP I2C 总线数据 373
自动波特率计算 244
时序图和规范
并行从动端口要求 366
捕捉 / 比较 /PWM 要求 （包括 ECCP 模块） 366
CLKO 和 I/O 要求 ..... 361， 362
程序存储器写要求 363
EUSART 同步发送要求 375
EUSART 同步接收要求 375
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 364
I2C 总线启动 / 停止位要求 （从动模式） 371
I2C 总线数据要求 （从动模式） 372
交流规范
内部 RC 精度 360
PLL 时钟 360
SPI 从动模式要求示例 （CKE = 1） 370
SPI 模式要求示例 （从动模式， CKE = 0） 369
SPI 模式要求示例 （主控模式， CKE = 0） 367
SPI 模式要求示例 （主控模式， CKE = 1） 368
Timer0 和 Timer1 外部时钟要求 365
外部时钟要求 359
主控 SSP I2C 总线启动 / 停止位要求 373
主控 SSP I2C 总线数据要求 374
时钟源 30
复位时的默认系统时钟 31
使用 OSCCON 寄存器进行选择 31
数据存储器 64
存储器映射图
PIC18FX5J10/X5J15/X6J10 器件 65
PIC18FX6J15/X7J10 器件 66
特殊功能寄存器 68
存储区选择寄存器 （BSR） 64
快速操作存储区 67
扩展指令集 77
特殊功能寄存器 68
通用寄存器 67
数据寻址模式 74
固有和立即数 74
间接寻址 74
立即数偏移变址寻址 77
BSR 79
快速操作存储区映射 79
受影响的指令 77
使能扩展指令集时的寻址模式对比 78
直接寻址 74
双速启动 277
双字指令
示例情形 63
所有寄存器的初始化条件 49–53

T

TBLRD 327
TBLWT 328
Timer0 147
16 位读写模式 148
分频比选择 （T0PS2:T0PS0 位） 149
工作原理 148
时钟源边沿选择 （T0SE 位） 148
时钟源选择 （T0CS 位） 148
相关的寄存器 149
溢出中断 149
预分频器 149
切换分配 149
预分频器分配 （PSA 位） 149
预分频器。*参见*预分频器， Timer0。
Timer2 157
工作原理 157
PR2 寄存器 177
输出 158
TMR2 与 PR2 匹配中断 177
相关的寄存器 158
中断 158
Timer3 159
16 位读 / 写模式 161
工作原理 160
TMR3H 寄存器 159
TMR3L 寄存器 159
特殊事件触发信号 （ECCP） 161
相关的寄存器 161
溢出中断 ..... 159， 161
振荡器 ..... 159， 161
Timer4 163
工作原理 163
后分频器。*参见*后分频器， Timer4。
MSSP 时钟移位 164
PR4 寄存器 163
TMR4 寄存器 163
TMR4 与 PR4 匹配中断 ..... 163， 164
相关的寄存器 164
预分频器。*参见*预分频器， Timer4。
Timer1
16 位读 / 写模式 153
低功耗选项 153
复位，使用 ECCP 特殊事件触发信号 154
工作原理 152
TMR1H 寄存器 151
TMR1L 寄存器 151
特殊事件触发器 （ECCP） 176
相关的寄存器 155
溢出中断 151
用作时钟源 153

振荡器 ..... 151， 153
布线注意事项 154
振荡器，作为辅助时钟 30
中断 154
作为实时时钟 154
TRISE 寄存器
PSPMODE 位 144
TSTFSZ 329
TXSTAx 寄存器
BRGH 位 239
特殊事件触发器。*参见*比较 （ECCP 模块）。
同步间隔字符自动唤醒 248

W

WCOL ..... 222， 223， 224， 227
WCOL 状态标志位 ..... 222， 223， 224， 227
VDDCORE/VCAP 引脚 284
WWW 地址 395
WWW 在线技术支持 4
外部存储总线 91
16 位模式时序 98
16 位数据宽度模式 94
16 位字节写模式 95
16 位字节选择模式 97
16 位字写模式 96
8 位模式 99
8 位模式时序 100
程序存储模式 94
单片机 94
扩展单片机 94
等待状态 94
地址和数据宽度 93
地址和数据线用法 （表格） 93
地址移位 93
端口引脚弱上拉 94
工作在功耗管理模式下 101
I/O 端口功能 91
控制 92
外部时钟输入 （EC 模式） 28
稳压器 （片内） 284

X

XORLW 329
XORWF 330
校准 （A/D 转换器） 265
休眠
OSC1 和 OSC2 引脚状态 33

Y

引脚功能
AVDD 25
AVDD 16
AVSS 25
AVSS 16
ENVREG ..... 16， 25
MCLR ..... 10， 17
OSC1/CLKI ..... 10， 17
OSC2/CLKO ..... 10， 17
RA0/AN0 ..... 10， 17
RA1/AN1 ..... 10， 17
RA2/AN2/VREF- ..... 10， 17
RA3/AN3/VREF+ ..... 10， 17
RA4/T0CKI ..... 10， 17
RA5/AN4 ..... 10， 17
RB0/INT0/FLT0 ..... 11， 18
RB1/INT1 ..... 11， 18
RB2/INT2 ..... 11， 18
RB3/INT3 11
RB3/INT3/ECCP2/P2A 18
RB4/KBI0 ..... 11， 18
RB5/KBI1 ..... 11， 18
RB6/KBI2/PGC ..... 11， 18
RB7/KBI3/PGD ..... 11， 18
RC0/T1OSO/T13CKI ..... 12， 19
RC1/T1OSI/ECCP2/P2A ..... 12， 19
RC2/ECCP1/P1A ..... 12， 19
RC3/SCK1/SCL1 ..... 12， 19
RC4/SDI1/SDA1 ..... 12， 19
RC5/SDO1 ..... 12， 19
RC6/TX1/CK1 ..... 12， 19
RC7/RX1/DT1 ..... 12， 19
RD0/AD0/PSP0 20
RD0/PSP0 13
RD1/AD1/PSP1 20
RD1/PSP1 13
RD2/AD2/PSP2 20
RD2/PSP2 13
RD3/AD3/PSP3 20
RD3/PSP3 13
RD4/AD4/PSP4/SDO2 20
RD4/PSP4/SDO2 13
RD5/AD5/PSP5/SDI2/SDA2 20
RD5/PSP5/SDI2/SDA2 13
RD6/AD6/PSP6/SCK2/SCL2 20
RD6/PSP6/SCK2/SCL2 13
RD7/AD7/PSP7/SS2 20
RD7/PSP7/SS2 13
RE0/AD8/RD/P2D 21
RE0/RD/P2D 14
RE1/AD9/WR/P2C 21
RE1/WR/P2C 14
RE2/AD10/CS/P2B 21
RE2/CS/P2D 14
RE3/AD11/P3C 21
RE3/P3C 14
RE4/AD12/P3B 21
RE4/P3B 14
RE5/AD13/P1C 21
RE5/P1C 14
RE6/AD14/P1B 21
RE6/P1B 14
RE7/AD15/ECCP2/P2A 21
RE7/ECCP2/P2A 14
RF1/AN6/C2OUT ..... 15， 22
RF2/AN7/C1OUT ..... 15， 22
RF3/AN8 ..... 15， 22
RF4/AN9 ..... 15， 22
RF5/AN10/CVREF ..... 15， 22
RF6/AN11 ..... 15， 22
RF7/SS1 ..... 15， 22
RG0/ECCP3/P3A ..... 16， 23
RG1/TX2/CK2 ..... 16， 23
RG2/RX2/DT2 ..... 16， 23
RG3/CCP4/P3D ..... 16， 23
RG4/CCP5/P1D ..... 16， 23
RH0/A16 24
RH1/A17 24
RH2/A18 24
RH3/A19 24
RH4/AN12/P3C 24
RH5/AN13/P3B 24

RH6/AN14/P1C 24
RH7/AN15/P1B 24
RJ0/ALE 25
RJ1/OE 25
RJ2/WRL 25
RJ3/WRH 25
RJ4/BA0 25
RJ5/CE 25
RJ6/LB 25
RJ7/UB 25
VDD 25
VDD 16
VDDCORE/VCAP ..... 16， 25
VSS 25
VSS 16
因特网地址 395
硬件乘法器 103
工作原理 103
简介 103
性能比较 103
预分频器， Timer0 149
预分频器， Timer2 （Timer4） 171

Z

在线串行编程 （ICSP） ..... 277， 288
在线调试器 288
增强型
PWM 框图 177
增强型捕捉 / 比较 /PWM （ECCP） ..... 173， 174
捕捉模式。*参见*捕捉 （ECCP 模块）。
定时器资源 174
ECCP1/ECCP3 输出和程序存储器模式 174
ECCP1 的引脚配置 175
ECCP1 和 ECCP3 对 CCP4 和 CCP5 的使用 174
ECCP2 的引脚配置 175
ECCP2 输出和程序存储器模式 174
ECCP3 的引脚配置 176
PWM 模式。*参见* PWM （ECCP 模块）。
输出和配置 174
增强型通用同步 / 异步收发器 （EUSART）。*参见* EUSART。
振荡器配置 27
EC 27
ECPLL 27
HS 27
HS 模式 27
HSPLL 27
INTRC 27
内部振荡器模块 30
振荡器起振定时器 （OST） 33
振荡器切换 30
振荡器选择 277
振荡器转换 31
振荡器， Timer3 159
振荡器， Timer1 ..... 151， 161
直接寻址 75
指令集 289
ADDLW 295
ADDWF 295
ADDWF （立即数变址寻址模式） 337
ADDWFC 296
ANDLW 296
ANDWF 297
BC 297
BCF 298
BN 298
BNC 299
BNN 299
BNOV 300
BNZ 300
BOV 303
BRA 301
BSF 301
BSF （立即数变址寻址模式） 337
BTFSC 302
BTFSS 302
BTG 303
BZ 304
标准指令集 289
CALL 304
CLRF 305
CLRWDT 305
COMF 306
CPFSEQ 306
CPFSGT 307
CPFSLT 307
操作码字段 290
DAW 308
DCFSNZ 309
DECF 308
DECFSZ 309
GOTO 310
INCF 310
INCFSZ 311
INFSNZ 311
IORLW 312
IORWF 312
扩展的指令 331
使能时的注意事项 336
使用 MPLAB IDE 工具 338
语法 331
LFSR 313
MOVF 313
MOVFF 314
MOVLB 314
MOVLW 315
MOVWF 315
MULLW 316
MULWF 316
NEGF 317
NOP 317
POP 318
PUSH 318
RCALL 319
RESET 319
RETFIE 320
RETLW 320
RETURN 321
RLCF 321
RLNCF 322
RRCF 322
RRNCF 323
SETF 323
SETF （立即数变址寻址模式） 337
SLEEP 324
SWAPF 326
SUBFWB 324
SUBLW 325
SUBWF 325
SUBWFB 326
TBLRD 327
TBLWT 328

TSTFSZ 329
通用格式 291
XORLW 329
XORWF 330
指令周期 62
时钟机制 62
指令流 / 流水线 62
直流规范
掉电和供电电流 346
供电电压 345
中断 105
电平变化中断 （RB7:RB4） 124
INTn 引脚 120
PORTB 电平变化中断 120
TMR0 120
现场保护 120
中断源 277
A/D 转换完成 261
比较完成 （CCP） 168
捕捉完成 （CCP） 167
TMR0 溢出 149
TMR1 溢出 151
TMR2 与 PR2 匹配 （PWM） 177
TMR3 溢出 ..... 159， 161
TMR4 与 PR4 匹配 164
TMR4 与 PR4 匹配 （PWM） 163
中断，标志位
电平变化中断 （RB7:RB4）标志 （RBIF 位） 124
主复位 （MCLR） 45
主控同步串行端口 （Master Synchronous Serial Port，MSSP）。*参见* MSSP。

注：

注：

MICROCHIP 网站

Microchip 网站（www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的因特网浏览器即可访问。网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和样本程序、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，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
- 开发系统信息热线

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://support.microchip.com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: PIC18F87J10　　文献编号: DS39663B_CN

问题:

1. 本文档中哪些部分最有特色？

2. 本文档是否满足了您的软硬件开发要求？如何满足的？

3. 您认为本文档的组织结构便于理解吗？如果不便于理解，那么问题何在？

4. 您认为本文档应该添加哪些内容以改善其结构和主题？

5. 您认为本文档中可以删减哪些内容，而又不会影响整体使用效果？

6. 本文档中是否存在错误或误导信息？如果存在，请指出是什么信息及其具体页数。

7. 您认为本文档还有哪些方面有待改进？

产品标识体系

欲订货或获取价格、交货等信息，请与我公司生产厂或销售办事处联系。

器件编号	X	/XX	XXX
器件	温度范围	封装	编程模式

器件	PIC18F65J10/65J15/66J10/66J15/67J10(1), PIC18F85J10/85J15/86J10/86J15/87J10(1), PIC18F65J10/65J15/66J10/66J15/67J10(2), PIC18F85J10/85J15/86J10/86J15/87J10T(2) ;	
温度范围	I	表示-40°C 至 +85°C （工业级）
封装	PT	表示TQFP （薄型正方扁平封装）
编程模式	QTP、 SQTP、代码或特殊要求 （空白为其他情况）	

示例：

a） PIC18F86J10-I/PT 301 表示工业级温度， TQFP 封装， QTP 模式 #301。

b） PIC18F65J15T-I/PT 表示卷带式，工业级温度， TQFP 封装。

注 1： F 表示标准电压范围
2： T 表示卷带式封装

全球销售及服务网点

美洲

公司总部 Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://support.microchip.com
网址：www.microchip.com

亚特兰大 Atlanta
Alpharetta, GA
Tel: 1-770-640-0034
Fax: 1-770-640-0307

波士顿 Boston
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Farmington Hills, MI
Tel: 1-248-538-2250
Fax: 1-248-538-2260

科科莫 Kokomo
Kokomo, IN
Tel: 1-765-864-8360
Fax: 1-765-864-8387

洛杉矶 Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608

圣何塞 San Jose
Mountain View, CA
Tel: 1-650-215-1444
Fax: 1-650-961-0286

加拿大多伦多 Toronto
Mississauga, Ontario,
Canada
Tel: 1-905-673-0699
Fax: 1-905-673-6509

亚太地区

中国 – 北京
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

中国 – 成都
Tel: 86-28-8676-6200
Fax: 86-28-8676-6599

中国 – 福州
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

中国 – 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

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 – 顺德
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

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

台湾地区 – 高雄
Tel: 886-7-536-4818
Fax: 886-7-536-4803

台湾地区 – 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

台湾地区 – 新竹
Tel: 886-3-572-9526
Fax: 886-3-572-6459

亚太地区

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-2229-0061
Fax: 91-80-2229-0062

印度 India - New Delhi
Tel: 91-11-5160-8631
Fax: 91-11-5160-8632

印度 India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

日本 Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

韩国 Korea - Gumi
Tel: 82-54-473-4301
Fax: 82-54-473-4302

韩国 Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

马来西亚 Malaysia - Penang
Tel: 60-4-646-8870
Fax: 60-4-646-5086

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: 43-7242-2244-399
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

10/31/05