PIC24FJ128GA010 系列数据手册

16 位 60/80/100 引脚通用闪存单片机

请注意以下有关 Microchip 器件代码保护功能的要点：

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

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

ISBN：978-1-60932-842-9

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

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

PIC24FJ128GA010 系列

16 位 60/80/100 引脚通用闪存单片机

高性能 CPU：

- 改进型哈佛架构
- 工作性能高达 16 MIPS @ 32 MHz
- 具有 4 倍频 PLL 选项和多个分频选项的 8 MHz 内部振荡器
- 17 位 x17 位单周期硬件乘法器
- 32 位 /16 位硬件除法器
- 16 位 x16 位工作寄存器阵列
- 优化的 C 编译器指令集架构：
  - 76 条基本指令
  - 灵活的寻址模式
- 两个地址发生单元可分别对数据存储器执行读和写寻址

单片机的特殊性能：

- 工作电压范围为 2.0V 到 3.6V
- 闪存程序存储器：
  - 可耐受 1000 次擦 / 写
  - 数据保存时间至少 20 年
- 软件控制下可自行再编程
- 可选的功耗管理模式：
  - 休眠模式、空闲模式和备用时钟模式
- 故障保护时钟监视器工作：
  - 一旦检测到时钟故障，将时钟源切换到片内低功耗 RC 振荡器
- 片上 2.5V 稳压器
- 支持 JTAG 边界扫描和编程
- 上电复位（Power-on Reset， POR）、上电延时定时器（Power-up Timer， PWRT）和振荡器起振定时器（Oscillator Start-up Timer， OST）
- 灵活的可编程看门狗定时器（Watchdog Timer，WDT）和片上低功耗 RC 振荡器可保证器件可靠工作
- 通过 2 个引脚可实现在线串行编程（In-Circuit Serial Programming™， ICSP™）和在线仿真（In-Circuit Emulation， ICE）

模拟特性：

- 10 位最多 16 路通道的模数转换器
  - 500 ksps 转换速率
  - 可在休眠模式和空闲模式下进行转换
- 2 个具有可编程输入 / 输出配置的模拟比较器

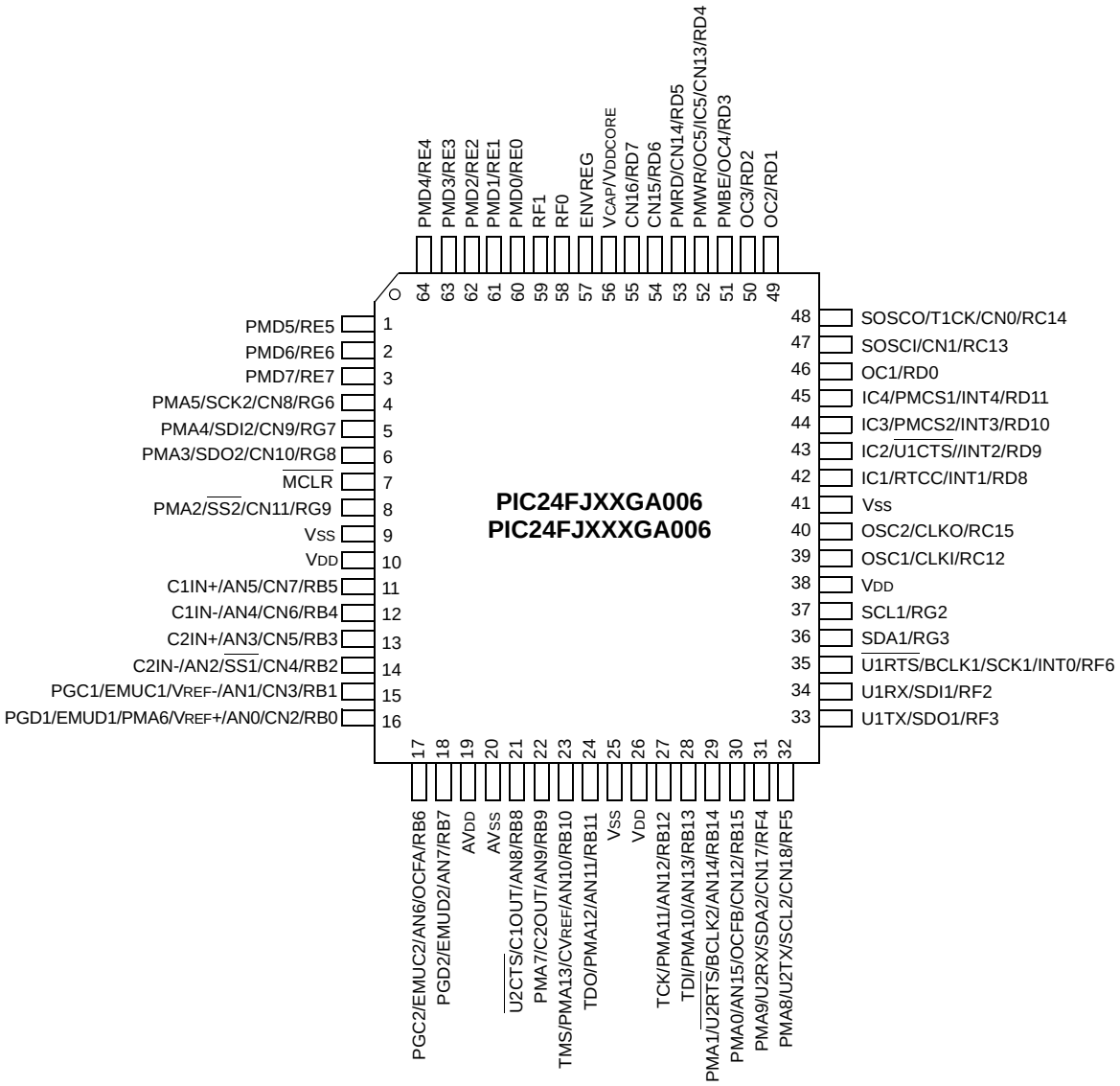
外设特性：

- 两个 3 线 /4 线 SPI 模块，利用 8 级 FIFO 缓冲器支持 4 种帧模式
- 两个支持多主 / 从动模式和 7 位 /10 位寻址的 I^2C™ 模块
- 两个 UART 模块：
  - 支持 RS-232、 RS-485 和 LIN 1.2
  - 支持 IrDA® 功能的片上硬件编码器 / 解码器
  - 起始位自动唤醒
  - 自动波特率检测
  - 4 级 FIFO 缓冲器
- 并行主 / 从端口（PMP/PSP）：
  - 支持 8 位或 16 位数据
  - 支持 16 条地址线
- 硬件实时时钟 / 日历（Real-Time Clock/Calendar，RTCC）：
  - 提供时钟、日历和闹钟功能
- 可编程循环冗余校验（Programmable Cyclic Redundancy Check， CRC）
  - 用户可编程多项式
  - 8/16 级 FIFO 缓冲区
- 5 个带可编程预分频器的 16 位定时器 / 计数器
- 5 个 16 位捕捉输入
- 5 个 16 位比较 /PWM 输出
- 所有 I/O 引脚上的灌 / 拉电流均很高，为 18 mA/18 mA
- 数字 I/O 引脚可配置为漏极开路输出
- 最多 5 个外部中断源
- 可耐受 5.5V 电压的输入（仅数字输入）

器件	引脚	程序存储器（字节）	SRAM（字节）	16 位定时器	捕捉输入	比较 / PWM 输出	UART	SPI	I^2C™	10 位 A/D（通道数）	比较器	PMP/PSP	JTAG
PIC24FJ64GA006	64	64K	8K	5	5	5	2	2	2	16	2	有	有
PIC24FJ96GA006	64	96K	8K	5	5	5	2	2	2	16	2	有	有
PIC24FJ128GA006	64	128K	8K	5	5	5	2	2	2	16	2	有	有
PIC24FJ64GA008	80	64K	8K	5	5	5	2	2	2	16	2	有	有
PIC24FJ96GA008	80	96K	8K	5	5	5	2	2	2	16	2	有	有
PIC24FJ128GA008	80	128K	8K	5	5	5	2	2	2	16	2	有	有
PIC24FJ64GA010	100	64K	8K	5	5	5	2	2	2	16	2	有	有
PIC24FJ96GA010	100	96K	8K	5	5	5	2	2	2	16	2	有	有
PIC24FJ128GA010	100	128K	8K	5	5	5	2	2	2	16	2	有	有

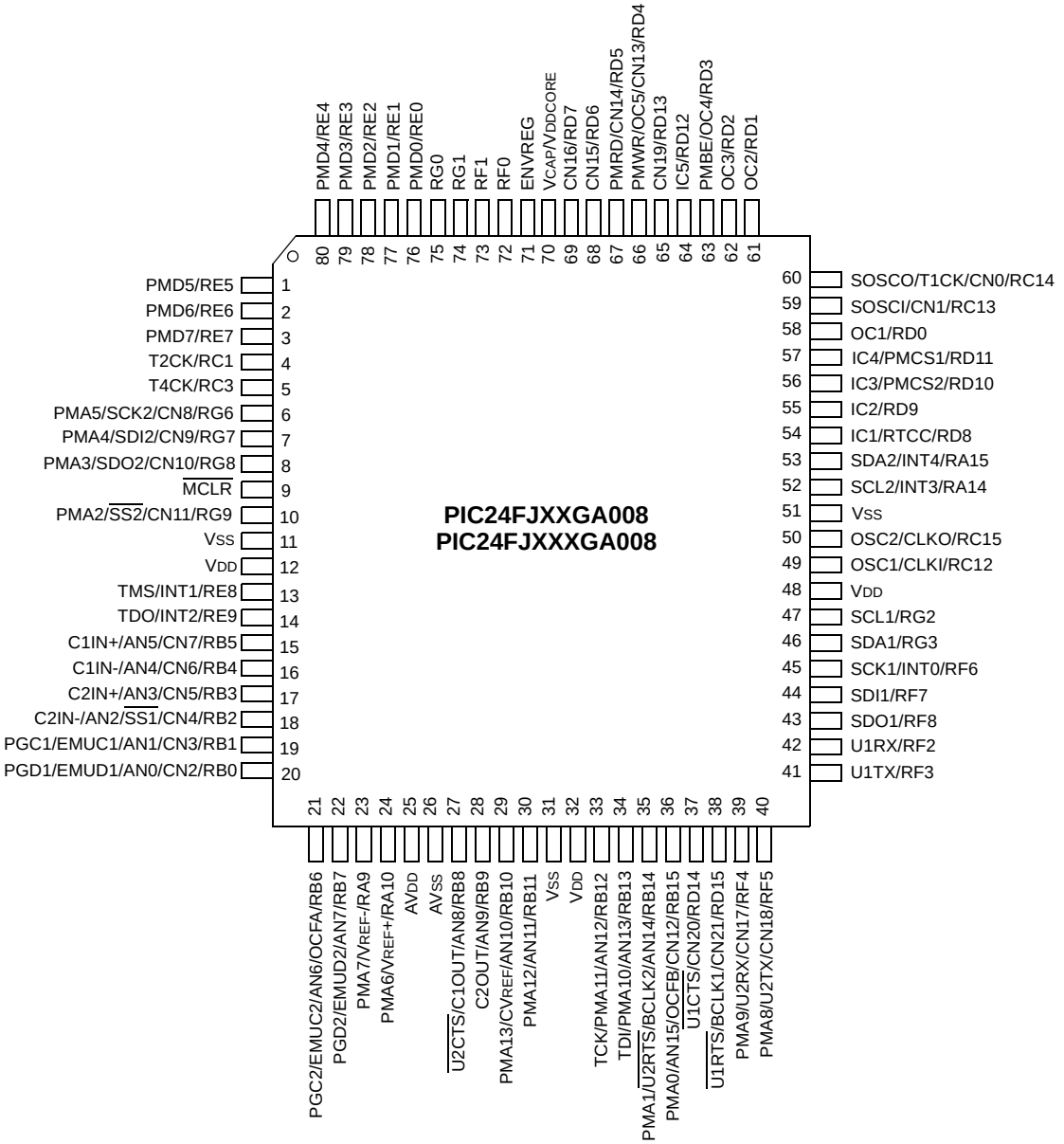
引脚图

64 引脚 TQFP



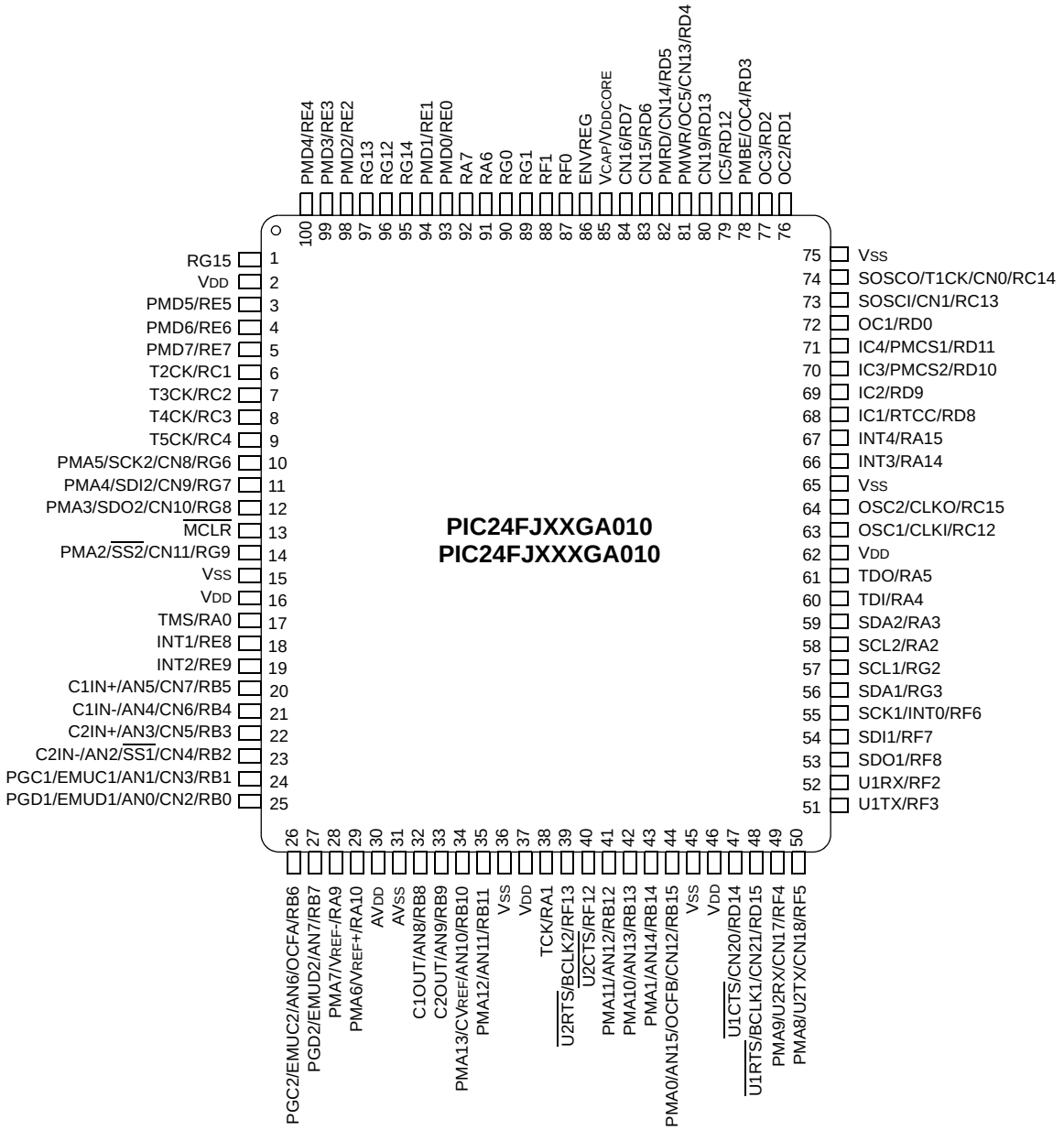
引脚图 （续）

80 引脚 TQFP



引脚图 （续）

100 引脚 TQFP



目录

致客户

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

如果您对本出版物有任何问题和建议，请通过电子邮件联系我公司 TRC 经理，电子邮件地址为 **CTRC@microchip.com**，或将本数据手册后附的 **《读者反馈表》**传真到 86-21-5407 5066。我们期待您的反馈。

最新数据手册

欲获得本数据手册的最新版本，请查询我公司的网站：

http://www.microchip.com

查看数据手册中任意一页下边角处的文献编号即可确定其版本。文献编号中数字串后的字母是版本号，例如：DS30000A是DS30000的 A 版本。

勘误表

现有器件可能带有一份勘误表，描述了实际运行与数据手册中记载内容之间存在的细微差异以及建议的变通方法。一旦我们了解到器件 / 文档存在某些差异时，就会发布勘误表。勘误表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

该文档包含针对以下器件的信息：

- PIC24FJ64GA006
- PIC24FJ64GA008
- PIC24FJ64GA010
- PIC24FJ96GA006
- PIC24FJ96GA008
- PIC24FJ96GA010
- PIC24FJ128GA006
- PIC24FJ128GA008
- PIC24FJ128GA010

此系列是一条新的 Microchip 产品线，即：具有丰富的外设功能集和增强的计算性能的 16 位单片机。PIC24FJ128GA010 系列还提供了一个新的移植通道，主要针对那些适用于高于 8 位的平台而又不需要使用数字信号处理器的应用。

1.1 内核特性

1.1.1 16 位架构

PIC24F 器件的核心是 16 位改进型哈佛架构，这种架构最早是由 Microchip 的 dsPIC® 数字信号控制器采用的。PIC24F CPU 内核提供了大量增强功能，如：

- 16 位数据路径和 24 位地址路径，可在数据空间和存储器空间传递信息
- 线性寻址空间最多可达 8 MB （程序）和 64 KB （数据）
- 利用内建软件堆栈支持 16 x16 工作寄存器阵列
- 支持整数运算的 17 位 x17 位硬件乘法器
- 支持 32 位 /16 位除法运算的硬件
- 支持多种寻址模式并为高级语言 （如 C 语言）而优化的指令集
- 工作性能可达 16 MIPS

1.1.2 节能技术

PIC24FJ128GA010 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：

- **动态时钟切换：**在器件工作过程中，器件时钟可在软件控制下切换为 Timer1 时钟源或内部低功耗 RC
  振荡器，允许用户把节能理念融入到软件设计中去。
- **打盹模式：**当那些对时间要求很高的应用 （如串行通信）要求外设不间断地工作时，该模式可以适当降低 CPU 时钟速度，从而可在不丢失数据的前提下进一步节约功耗。
- **基于指令的节能模式：**通过在软件中使用一条指令，单片机可以暂停所有的操作或仅关闭内核，而让外设处于活动状态。

1.1.3 振荡器选项和性能

PIC24FJ128GA010 系列中的所有器件提供 5 个不同的振荡器选项，使用户在开发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 2 分频时钟输出选项。
- 一个标称输出值为 8 MHz 的快速内部振荡器 （Fast Internal Oscillator， FRC），可以在软件控制下被分频，从而使时钟速度可低至 31 kHz。
- 一个锁相环 （Phase Lock Loop， PLL）倍频器，可在外部振荡器模式和 FRC 振荡器下使用，从而可使时钟速度最高达 32 MHz。
- 具有固定的 31 kHz 输出的独立内部 RC 振荡器 （LPRC），可为对时间要求很高的应用提供低功耗时钟选项。

内部振荡器模块还为故障保护时钟监视器提供了一个稳定的参考源。故障保护时钟监视器不断地监视主时钟源，将之与内部振荡器提供的参考信号作比较。一旦发生时钟故障，允许控制器将时钟源切换到内部振荡器，继续保持低速工作或安全地关闭应用。

1.1.4 简便移植

无论存储器大小如何，所有器件均共享同一组外设，使得应用程序可在升级时很方便地移植。

整个系列使用相同的引脚配置方案也有助于向更大型器件的移植。可以在具有相同引脚数的器件间移植，甚至还可以从 64 引脚器件移植到 80 引脚或再次移植到 100 引脚器件。

PIC24F 系列器件的引脚同 dsPIC33 系列器件的引脚是兼容的，并与 PIC18 和 dsPIC30 的引脚部分兼容。这样全部采用 Microchip 器件，就可将应用从相对简单的功能顺利移植到强大和复杂的功能。

1.2 其他特殊性能

- **通信：**PIC24FJ128GA010 系列器件包含一系列串行通信外设。所有器件均配备两个独立的内置 IrDA 编解码器的 UART，此外还配备了两个独立的 SPI 模块和两个支持主 / 从工作模式的独立的 I^2C 模块。
- **并行主控 / 增强型并行从动端口：**可以将一个通用 I/O 端口配置为用于增强型并行数据通信。在这种模式下，可以将端口配置为工作在主控或从动模式下。在主控模式下支持 8 位或 16 位数据传输并具有最多 16 条外部地址线。
- **实时时钟 / 日历：**此模块通过硬件实现带有闹钟功能的全功能时钟和日历，从而释放了定时器资源和程序存储空间供核心应用使用。
- **10 位 A/D 转换器：**此模块实现了可编程采集时间，允许选择通道并立即开始转换，无需等待采样周期结束。同时提高了采样速度。

1.3 系列中各产品的具体信息

PIC24FJ128GA010 系列器件具有 64 引脚、80 引脚和 100 引脚封装形式。图 1-1 中显示了使用于所有器件的通用框图。

这些器件在以下 2 个方面存在差异：

1. 闪存程序存储器（PIC24FJ64GA 器件为 64 KB，PIC24FJ96GA 器件为 96 KB 而 PIC24FJ128GA 器件为 128 KB）。
2. 可用的 I/O 引脚数和端口数（64 引脚的器件有 6 个端口 56 个引脚，80 引脚的器件有 7 个端口 69 个引脚而 100 引脚的器件有 7 个端口 84 个引脚）。还需注意的是，因为该系列器件的所有 I/O 引脚均具备输入电平变化中断功能，不同封装尺寸的 CN 引脚数也不同。

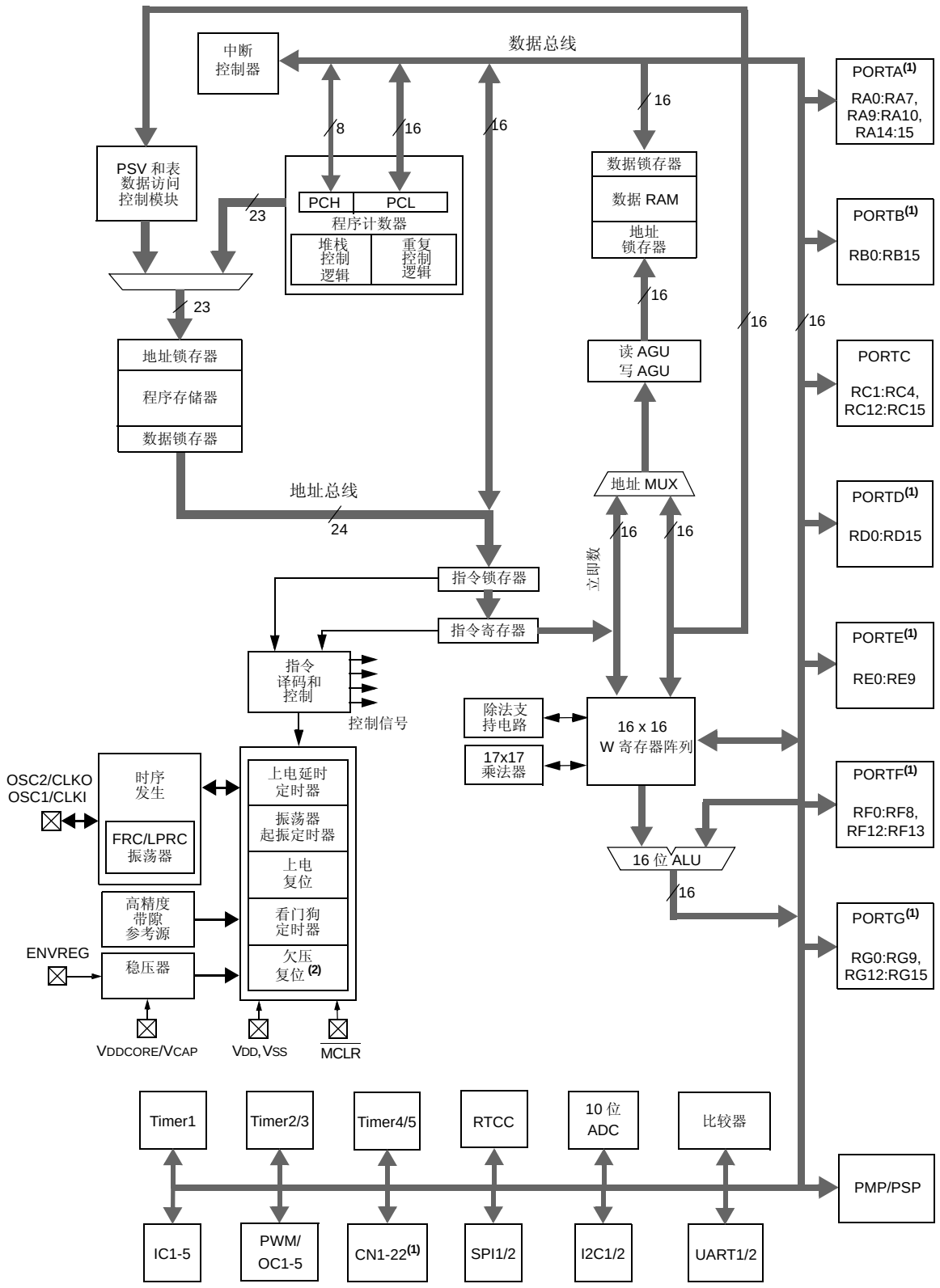
该系列器件的其他指标都是相同的。表 1-1 对此进行了总结。

表 1-2 显示了 PIC24FJ128GA010 系列器件上可用的引脚功能，按功能名称排序。注意此表只显示了各个外设功能所使用的引脚，而没有显示同一引脚上的多种功能的复用方式。在本数据手册开始部分的引脚图中提供了有关引脚复用的信息。复用的功能按功能的优先级排列，最前面的是优先级最高的外设功能。

表 1-1： PIC24FJ128GA010 系列器件的性能指标

性能指标	PIC24FJ64GA006	PIC24FJ96GA006	PIC24FJ128GA006	PIC24FJ64GA008	PIC24FJ96GA008	PIC24FJ128GA008	PIC24FJ64GA010	PIC24FJ96GA010	PIC24FJ128GA010
工作频率	DC – 32 MHz								
程序存储器 （字节）	64K	96K	128K	64K	96K	128K	64K	96K	128K
程序存储器 （指令）	22,016	32,768	44,032	22,016	32,768	44,032	22,016	32,768	44,032
数据存储器 （字节）	8192								
中断源 （软向量 /NMI 陷阱）	43 （39/4）								
I/O 端口	端口 B、 C、 D、 E、 F 和 G			端口 A、B、C、D、E、F 和 G			端口 A、B、C、D、E、F 和 G		
I/O 引脚总数	53			69			84		
定时器：									
总数 （16 位）	5								
32 位（由一对 16 位定时器组成）	2								
输入捕捉通道	5								
输出比较 /PWM 通道	5								
输入变化通知中断	19			22					
串行通信：									
UART	2								
SPI （3 线 /4 线）	2								
I^2C™	2								
并行通信 （PMP/PSP）	有								
JTAG 边界扫描	有								
10 位模数转换模块 （输入通道）	16								
模拟比较器	2								
复位 （和延时）	POR、BOR、RESET 指令、$\overline{\text{MCLR}}$、WDT、非法操作码、配置字不匹配、重复指令和硬件陷阱 （PWRT、 OST 和 PLL 锁定）								
指令集	76 条基本指令和多种寻址模式								
封装	64 引脚 TQFP			80 引脚 TQFP			100 引脚 TQFP		

图 1-1：　　PIC24FJ128GA010 系列的通用框图



注　**1：** 图中引脚或特性并非在所有器件引脚配置中都能实现。请参见表 1-2 了解 I/O 端口引脚说明。

2： 当使能片上稳压器时可提供 BOR 功能。

表 1-2： PIC24FJ128GA010 系列引脚配置说明

功能	引脚数			I/O	输入缓冲器	说明
	64 引脚	80 引脚	100 引脚			
AN0	16	20	25	I	ANA	A/D 模拟输入。
AN1	15	19	24	I	ANA	
AN2	14	18	23	I	ANA	
AN3	13	17	22	I	ANA	
AN4	12	16	21	I	ANA	
AN5	11	15	20	I	ANA	
AN6	17	21	26	I	ANA	
AN7	18	22	27	I	ANA	
AN8	21	27	32	I	ANA	
AN9	22	28	33	I	ANA	
AN10	23	29	34	I	ANA	
AN11	24	30	35	I	ANA	
AN12	27	33	41	I	ANA	
AN13	28	34	42	I	ANA	
AN14	29	35	43	I	ANA	
AN15	30	36	44	I	ANA	
AVDD	19	25	30	P	—	模拟模块的正电源。
AVSS	20	26	31	P	—	模拟模块的参考地。
BCLK1	35	38	48	O	—	UART1 IrDA® 波特率时钟。
BCLK2	29	35	39	O	—	UART2 IrDA® 波特率时钟。
C1IN-	12	16	21	I	ANA	比较器 1 负输入端。
C1IN+	11	15	20	I	ANA	比较器 1 正输入端。
C1OUT	21	27	32	O	—	比较器 1 输出。
C2IN-	14	18	23	I	ANA	比较器 2 负输入端。
C2IN+	13	17	22	I	ANA	比较器 2 正输入端。
C2OUT	22	28	33	O	—	比较器 2 输出。
CLKI	39	49	63	I	ANA	主时钟输入连接。
CLKO	40	50	64	O	—	系统时钟输出。
CN0	48	60	74	I	ST	输入电平变化中断输入。
CN1	47	59	73	I	ST	
CN2	16	20	25	I	ST	
CN3	15	19	24	I	ST	
CN4	14	18	23	I	ST	
CN5	13	17	22	I	ST	
CN6	12	16	21	I	ST	
CN7	11	15	20	I	ST	
CN8	4	6	10	I	ST	
CN9	5	7	11	I	ST	
CN10	6	8	12	I	ST	
CN11	8	10	14	I	ST	
CN12	30	36	44	I	ST	
CN13	52	66	81	I	ST	
CN14	53	67	82	I	ST	
CN15	54	68	83	I	ST	
CN16	55	69	84	I	ST	
CN17	31	39	49	I	ST	

图注： TTL = TTL 输入缓冲器　　ST = 施密特触发器输入缓冲器
ANA = 模拟电平输入 / 输出　　I^2C™ = I^2C/SMBus 输入缓冲器

表 1-2： PIC24FJ128GA010 系列引脚配置说明 （续）

功能	引脚数			I/O	输入缓冲器	说明
	64 引脚	80 引脚	100 引脚			
CN18	32	40	50	I	ST	输入电平变化中断输入。
CN19	—	65	80	I	ST	
CN20	—	37	47	I	ST	
CN21	—	38	48	I	ST	
CVREF	23	29	34	O	ANA	比较器参考电压输出。
EMUC1	15	19	24	I/O	ST	在线仿真器时钟输入 / 输出。
EMUD1	16	20	25	I/O	ST	在线仿真器数据输入 / 输出。
EMUC2	17	21	26	I/O	ST	在线仿真器时钟输入 / 输出。
EMUD2	18	22	27	I/O	ST	在线仿真器数据输入 / 输出。
ENVREG	57	71	86	I	ST	片内稳压器使能端。
IC1	42	54	68	I	ST	输入捕捉输入。
IC2	43	55	69	I	ST	
IC3	44	56	70	I	ST	
IC4	45	57	71	I	ST	
IC5	52	64	79	I	ST	
INT0	35	45	55	I	ST	外部中断输入。
INT1	42	13	18	I	ST	
INT2	43	14	19	I	ST	
INT3	44	52	66	I	ST	
INT4	45	53	67	I	ST	
$\overline{MCLR}$	7	9	13	I	ST	主清零 （器件复位）输入。将此引脚拉为低电平可导致复位。
OC1	46	58	72	O	—	输出比较 /PWM 输出。
OC2	49	61	76	O	—	
OC3	50	62	77	O	—	
OC4	51	63	78	O	—	
OC5	52	66	81	O	—	
OCFA	17	21	26	I	ST	输出比较故障 A 输入。
OCFB	30	36	44	I	ST	输出比较故障 B 输入。
OSC1	39	49	63	I	ANA	主振荡器输入连接。
OSC2	40	50	64	O	ANA	主振荡器输出连接。
PGC1	15	19	24	I/O	ST	在线调试器和 ICSP™ 编程时钟。
PGD1	16	20	25	I/O	ST	在线调试器和 ICSP 编程数据。
PGC2	17	21	26	I/O	ST	在线调试器和 ICSP™ 编程时钟。
PGD2	18	22	27	I/O	ST	在线调试器和 ICSP 编程数据。

图注： TTL = TTL 输入缓冲器　　ST = 施密特触发器输入缓冲器
ANA = 模拟电平输入 / 输出　　I^2C™ = I^2C/SMBus 输入缓冲器

表 1-2： PIC24FJ128GA010 系列引脚配置说明 （续）

功能	引脚数			I/O	输入缓冲器	说明
	64 引脚	80 引脚	100 引脚			
PMA0	30	36	44	I/O	ST/TTL	并行主控端口地址 bit 0 输入 （缓冲从动模式）和输出（主控模式）。
PMA1	29	35	43	I/O	ST/TTL	并行主控端口地址 bit 1 输入 （缓冲从动模式）和输出（主控模式）。
PMA2	8	10	14	O	—	并行主控端口地址 （非复用的主控模式）。
PMA3	6	8	12	O	—	
PMA4	5	7	11	O	—	
PMA5	4	6	10	O	—	
PMA6	16	24	29	O	—	
PMA7	22	23	28	O	—	
PMA8	32	40	50	O	—	
PMA9	31	39	49	O	—	
PMA10	28	34	42	O	—	
PMA11	27	33	41	O	—	
PMA12	24	30	35	O	—	
PMA13	23	29	34	O	—	
PMBE	51	63	78	O	—	并行主控端口字节使能选通。
PMCS1	45	57	71	I/O	ST/TTL	并行主控端口片选 1 选通 / 地址 bit 14。
PMCS2	44	56	70	O	—	并行主控端口片选 2 选通 / 地址 bit 15。
PMD0	60	76	93	I/O	ST/TTL	并行主控端口数据 （非复用的主控模式）或地址 / 数据（复用的主控模式）。
PMD1	61	77	94	I/O	ST/TTL	
PMD2	62	78	98	I/O	ST/TTL	
PMD3	63	79	99	I/O	ST/TTL	
PMD4	64	80	100	I/O	ST/TTL	
PMD5	1	1	3	I/O	ST/TTL	
PMD6	2	2	4	I/O	ST/TTL	
PMD7	3	3	5	I/O	ST/TTL	
PMRD	53	67	82	I/O	ST/TTL	并行主控端口读选通。
PMWR	52	66	81	I/O	ST/TTL	并行主控端口写选通。

图注： TTL = TTL 输入缓冲器　　ST = 施密特触发器输入缓冲器
ANA = 模拟电平输入 / 输出　　I^2C™ = I^2C/SMBus 输入缓冲器

表 1-2： PIC24FJ128GA010 系列引脚配置说明 （续）

功能	引脚数			I/O	输入缓冲器	说明
	64 引脚	80 引脚	100 引脚			
RA0	—	—	17	I/O	ST	PORTA 数字 I/O。
RA1	—	—	38	I/O	ST	
RA2	—	—	58	I/O	ST	
RA3	—	—	59	I/O	ST	
RA4	—	—	60	I/O	ST	
RA5	—	—	61	I/O	ST	
RA6	—	—	91	I/O	ST	
RA7	—	—	92	I/O	ST	
RA9	—	23	28	I/O	ST	
RA10	—	24	29	I/O	ST	
RA14	—	52	66	I/O	ST	
RA15	—	53	67	I/O	ST	
RB0	16	20	25	I/O	ST	PORTB 数字 I/O。
RB1	15	19	24	I/O	ST	
RB2	14	18	23	I/O	ST	
RB3	13	17	22	I/O	ST	
RB4	12	16	21	I/O	ST	
RB5	11	15	20	I/O	ST	
RB6	17	21	26	I/O	ST	
RB7	18	22	27	I/O	ST	
RB8	21	27	32	I/O	ST	
RB9	22	28	33	I/O	ST	
RB10	23	29	34	I/O	ST	
RB11	24	30	35	I/O	ST	
RB12	27	33	41	I/O	ST	
RB13	28	34	42	I/O	ST	
RB14	29	35	43	I/O	ST	
RB15	30	36	44	I/O	ST	
RC1	—	4	6	I/O	ST	PORTC 数字 I/O。
RC2	—	—	7	I/O	ST	
RC3	—	5	8	I/O	ST	
RC4	—	—	9	I/O	ST	
RC12	39	49	63	I/O	ST	
RC13	47	59	73	I/O	ST	
RC14	48	60	74	I/O	ST	
RC15	40	50	64	I/O	ST	

图注： TTL = TTL 输入缓冲器　　ST = 施密特触发器输入缓冲器
ANA = 模拟电平输入 / 输出　　I^2C™ = I^2C/SMBus 输入缓冲器

表 1-2： PIC24FJ128GA010 系列引脚配置说明 （续）

功能	引脚数			I/O	输入缓冲器	说明
	64 引脚	80 引脚	100 引脚			
RD0	46	58	72	I/O	ST	PORTD 数字 I/O。
RD1	49	61	76	I/O	ST	
RD2	50	62	77	I/O	ST	
RD3	51	63	78	I/O	ST	
RD4	52	66	81	I/O	ST	
RD5	53	67	82	I/O	ST	
RD6	54	68	83	I/O	ST	
RD7	55	69	84	I/O	ST	
RD8	42	54	68	I/O	ST	
RD9	43	55	69	I/O	ST	
RD10	44	56	70	I/O	ST	
RD11	45	57	71	I/O	ST	
RD12	—	64	79	I/O	ST	
RD13	—	65	80	I/O	ST	
RD14	—	37	47	I/O	ST	
RD15	—	38	48	I/O	ST	
RE0	60	76	93	I/O	ST	PORTE 数字 I/O。
RE1	61	77	94	I/O	ST	
RE2	62	78	98	I/O	ST	
RE3	63	79	99	I/O	ST	
RE4	64	80	100	I/O	ST	
RE5	1	1	3	I/O	ST	
RE6	2	2	4	I/O	ST	
RE7	3	3	5	I/O	ST	
RE8	—	13	18	I/O	ST	
RE9	—	14	19	I/O	ST	
RF0	58	72	87	I/O	ST	PORTF 数字 I/O。
RF1	59	73	88	I/O	ST	
RF2	34	42	52	I/O	ST	
RF3	33	41	51	I/O	ST	
RF4	31	39	49	I/O	ST	
RF5	32	40	50	I/O	ST	
RF6	35	45	55	I/O	ST	
RF7	—	44	54	I/O	ST	
RF8	—	43	53	I/O	ST	
RF12	—	—	40	I/O	ST	
RF13	—	—	39	I/O	ST	

图注： TTL = TTL 输入缓冲器　　ST = 施密特触发器输入缓冲器
ANA = 模拟电平输入 / 输出　　I^2C™ = I^2C/SMBus 输入缓冲器

表 1-2： PIC24FJ128GA010 系列引脚配置说明 （续）

功能	引脚数			I/O	输入缓冲器	说明
	64 引脚	80 引脚	100 引脚			
RG0	—	75	90	I/O	ST	PORTG 数字 I/O。
RG1	—	74	89	I/O	ST	
RG2	37	47	57	I/O	ST	
RG3	36	46	56	I/O	ST	
RG6	4	6	10	I/O	ST	
RG7	5	7	11	I/O	ST	
RG8	6	8	12	I/O	ST	
RG9	8	10	14	I/O	ST	
RG12	—	—	96	I/O	ST	
RG13	—	—	97	I/O	ST	
RG14	—	—	95	I/O	ST	
RG15	—	—	1	I/O	ST	
RTCC	42	54	68	O	—	实时时钟闹钟输出。
SCK1	35	45	55	O	—	SPI1 串行时钟输出。
SCK2	4	6	10	I/O	ST	SPI2 串行时钟输出。
SCL1	37	47	57	I/O	I^2C	I2C1 同步串行时钟输入 / 输出。
SCL2	32	52	58	I/O	I^2C	I2C2 同步串行时钟输入 / 输出。
SDA1	36	46	56	I/O	I^2C	I2C1 数据输入 / 输出。
SDA2	31	53	59	I/O	I^2C	I2C2 数据输入 / 输出。
SDI1	34	44	54	I	ST	SPI1 串行数据输入。
SDI2	5	7	11	I	ST	SPI2 串行数据输入。
SDO1	33	43	53	O	—	SPI1 串行数据输出。
SDO2	6	8	12	O	—	SPI2 串行数据输出。
SOSCI	47	59	73	I	ANA	辅助振荡器 /Timer1 时钟输入。
SOSCO	48	60	74	O	ANA	辅助振荡器 /Timer1 时钟输出。
$\overline{SS1}$	14	18	23	I/O	ST	从动选择输入 / 帧选择输出 （SPI1）。
$\overline{SS2}$	8	10	14	I/O	ST	从动选择输入 / 帧选择输出 （SPI2）。
T1CK	48	60	74	I	ST	Timer1 时钟。
T2CK	—	4	6	I	ST	Timer2 外部时钟输入。
T3CK	—	—	7	I	ST	Timer3 外部时钟输入。
T4CK	—	5	8	I	ST	Timer4 外部时钟输入。
T5CK	—	—	9	I	ST	Timer5 外部时钟输入。
TCK	27	33	38	I	ST	JTAG 测试时钟 / 编程时钟输入。
TDI	28	34	60	I	ST	JTAG 测试数据 / 编程数据输入。
TDO	24	14	61	O	—	JTAG 测试数据输出。
TMS	23	13	17	I	ST	JTAG 测试模式选择输入。

图注： TTL = TTL 输入缓冲器　　ST = 施密特触发器输入缓冲器
ANA = 模拟电平输入 / 输出　　I^2C™ = I^2C/SMBus 输入缓冲器

表 1-2： PIC24FJ128GA010 系列引脚配置说明 （续）

功能	引脚数			I/O	输入缓冲器	说明
	64 引脚	80 引脚	100 引脚			
$\overline{\text{U1CTS}}$	43	37	47	I	ST	UART1 清零发送输入。
$\overline{\text{U1RTS}}$	35	38	48	O	—	UART1 请求发送输出。
U1RX	34	42	52	I	ST	UART1 接收。
U1TX	33	41	51	O	DIG	UART1 发送输出。
$\overline{\text{U2CTS}}$	21	27	40	I	ST	UART2 清零发送输入。
$\overline{\text{U2RTS}}$	29	35	39	O	—	UART2 请求发送输出。
U2RX	31	39	49	I	ST	UART2 接收输入。
U2TX	32	40	50	O	—	UART2 发送输出。
VDD	10, 26, 38	12, 32, 48	2, 16, 37, 46, 62	P	—	外设数字逻辑和 I/O 引脚的正电源。
VDDCAP	56	70	85	P	—	外部滤波电容连接 （稳压器已使能）。
VDDCORE	56	70	85	P	—	单片机内核逻辑的正电源 （稳压器已禁止）。
VREF-	15	23	28	I	ANA	A/D 和比较器参考电压 （低）输入。
VREF+	16	24	29	I	ANA	A/D 和比较器高参考电压 （高）输入。
VSS	9, 25, 41	11, 31, 51	15, 36, 45, 65, 75	P	—	逻辑电路和 I/O 引脚的参考地。

图注： TTL = TTL 输入缓冲器　　ST = 施密特触发器输入缓冲器
ANA = 模拟电平输入 / 输出　　I^2C™ = I^2C/SMBus 输入缓冲器

注：

2.0 CPU

注：	本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第2章“CPU”**（DS39703A_CN）。

PIC24F CPU 模块采用 16 位（数据）改进型哈佛架构，支持增强的指令集，指令字长 24 位，带有变量长度操作码字段。程序计数器（PC）为 23 位宽，可以寻址最多 4M 指令的用户程序存储空间。单周期指令预取机制用来保证吞吐量和可预测的执行结果。除了改变程序流的指令、双字移动（MOV.D）指令和表指令以外，所有指令都在单个周期内执行。使用 REPEAT 指令支持代码紧凑的程序循环结构，此指令在任何时间都可以被中断。

PIC24F 器件在编程模型中有 16 个 16 位工作寄存器。每个工作寄存器都可以充当数据、地址或地址偏移量寄存器。第 16 个工作寄存器（W15）作为软件堆栈的指针，用于中断和程序调用。

可以选择将数据存储器映射空间的高 32 KB 映射到由 8 位程序空间可视页面（Program Space Visibility Page，PSVPAG）寄存器定义的 16K 字的程序空间内。程序空间到数据空间的映射功能使得任何指令都能象访问数据空间一样访问程序空间。

指令集架构（Instruction Set Architecture，ISA）与 PIC18 的相比有了显著的提升，并保持了一定程度的向下兼容性。该架构直接或通过简单的宏支持所有的 PIC18 指令和寻址模式。编译器的效率也得到了很大的提高。

内核支持固有（无操作数）、相对、立即数、存储器直接寻址和三组寻址模式。所有模式都支持寄存器直接和寄存器间接寻址模式。每组寻址模式都提供了最多 7 种寻址方式。每条指令根据其功能要求，与一组预定义的寻址模式相关。

对于大多数指令，内核可在单个指令周期内执行一次数据（或程序）存储器读操作、一次工作寄存器（数据）读操作、一次数据存储器写操作和一次程序（指令）存储器读操作。因此可以支持 3 个操作数的指令，使 3 个数的运算（A + B = C）能在单周期内执行。

一个高速的 17 位× 17 位乘法器可以极大地加强内核的运算能力和吞吐量。此乘法器支持有符号、无符号和混合模式的 16 位× 16 位或 8 位× 8 位整数乘法。所有的乘法指令都在单周期内执行。

因为引入了支持迭代和不可恢复整数除法的硬件电路，16 位 ALU 的性能也得到了增强。ALU 结合 REPEAT 指令循环机制，使用一组迭代除法指令，支持 32 位（或 16 位）除以 16 位有符号和无符号整数除法。所有除法运算都需要 19 个周期才能完成，但可在任何周期边界被中断。

PIC24F 具有矢量排他（exception）机制，带有多达 8 个不可屏蔽陷阱源和 118 个中断源。可以为每个中断源分配 7 个优先级中的任何一个。

图 2-1 所示为 CPU 的框图。

2.1 编程模型

图 2-2 所示为 PIC24F 的编程模型。编程模型中所有寄存器都是存储器映射的，并且可以由指令直接访问。表 2-1中提供了对每个寄存器的说明。所有与编程模型相关的寄存器都是存储器映射的。

图 2-1： PIC24F CPU 内核框图

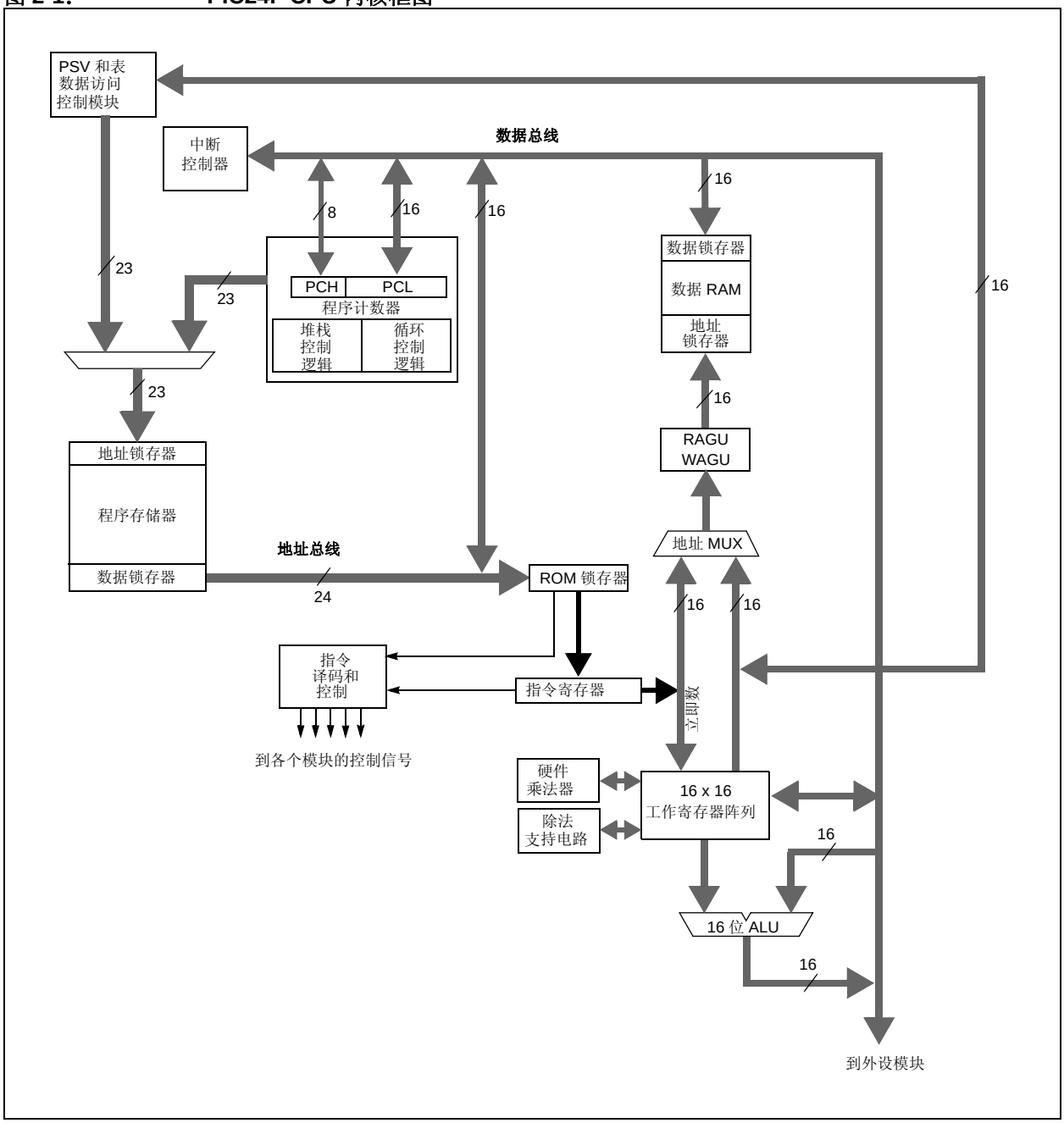
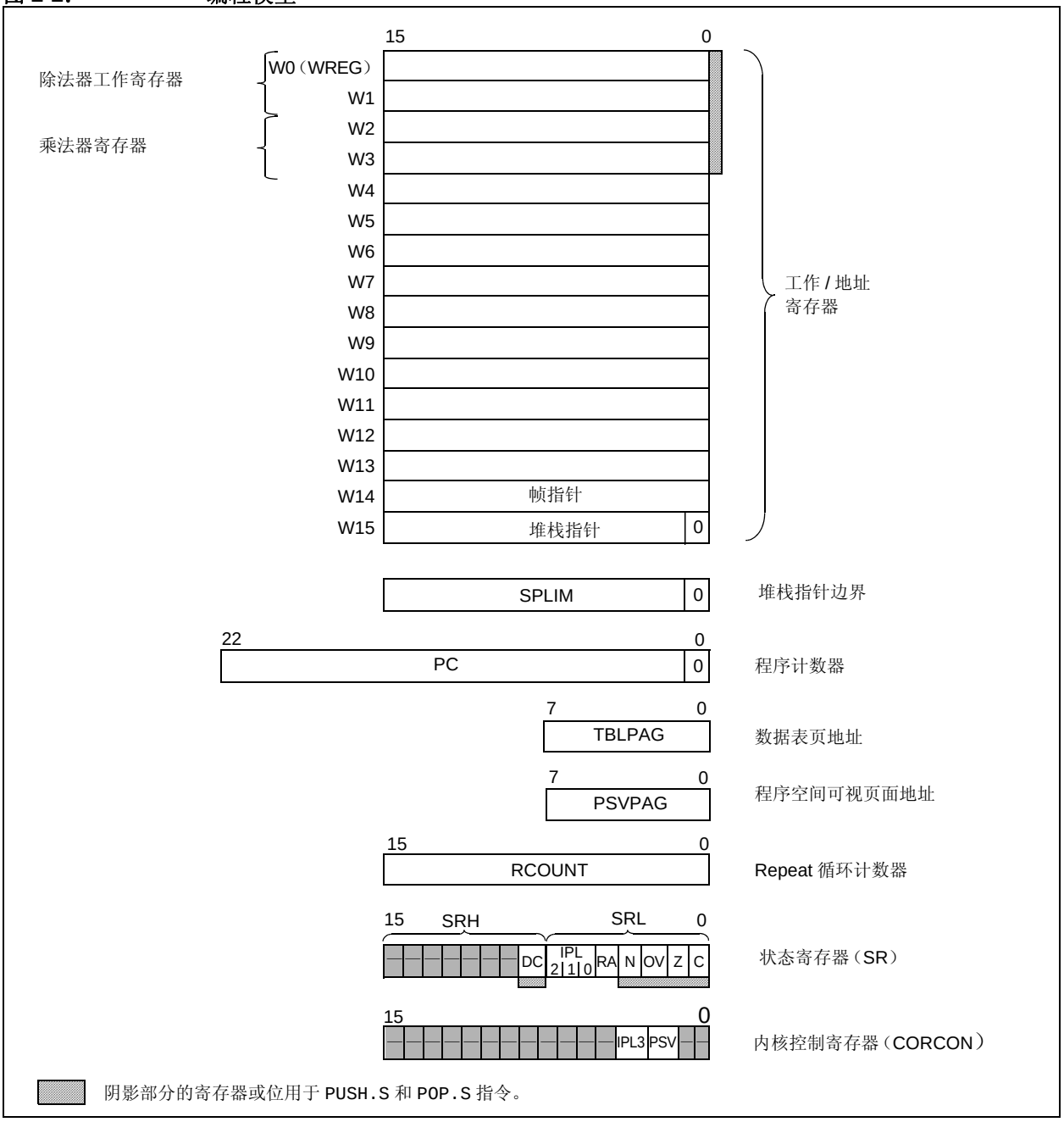


表 2-1： CPU 内核寄存器

寄存器名称	说明
W0 到 W15	工作寄存器阵列
PC	23 位程序计数器
SR	ALU 状态寄存器
SPLIM	堆栈指针边界寄存器
TBLPAG	表存储器页地址寄存器
PSVPAG	程序空间可视页面寄存器
RCOUNT	Repeat 循环计数寄存器
CORCON	CPU 控制寄存器

图 2-2： 编程模型



2.2 CPU 控制寄存器

寄存器 2-1： SR：CPU 状态寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	DC
bit 15							bit 8

R/W-0(1)	R/W-0(1)	R/W-0(1)	R-0	R/W-0	R/W-0	R/W-0	R/W-0
IPL2(2)	IPL1(2)	IPL0(2)	RA	N	OV	Z	C
bit 7							bit 0

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

bit 15-9 **未实现：**读为 0

bit 8 **DC：**ALU 半进位 / 借位标志位
1 = 结果的第 4 位 （数据单位是字节）或第 8 位 （数据单位是字）向高位发生了进位
0 = 结果的第 4 位 （数据单位是字节）或第 8 位 （数据单位是字）没有发生进位

bit 7-5 **IPL2:IPL0：**CPU 中断优先级状态位 (2)
111 = CPU 中断优先级为 7 （15）。禁止用户中断。
110 = CPU 中断优先级为 6 （14）
101 = CPU 中断优先级为 5 （13）
100 = CPU 中断优先级为 4 （12）
011 = CPU 中断优先级为 3 （11）
010 = CPU 中断优先级为 2 （10）
001 = CPU 中断优先级为 1 （9）
000 = CPU 中断优先级为 0 （8）

bit 4 **RA：**REPEAT 循环有效标志位
1 = 正在进行 REPEAT 循环
0 = 不在进行 REPEAT 循环

bit 3 **N：**ALU 负标志位
1 = 结果为负
0 = 结果为非负 （零或正数）

bit 2 **OV：**ALU 溢出标志位
1 = 在本次算术运算中有符号运算 （以 2 的补码方式进行）发生了溢出
0 = 未发生溢出

bit 1 **Z：**ALU 全零标志位
1 = 对 Z 位有影响的操作已在过去的某一时刻将此位置 1
0 = 对 Z 位有影响的最近一次运算已将此位清零 （即，运算结果非零）

bit 0 **C：**ALU 进位 / 借位标志位
1 = 结果的最高位发生了进位
0 = 结果的最高位未发生进位

注 1：当 NSTDIS （INTCON1<15>） = 1 时， IPL 状态位是只读的。

2：IPL 位与 IPL3 位（CORCON<3>）一起构成 CPU 中断优先级。当 IPL3 = 1 时，IPL 等于括号中的数值。

寄存器 2-2：　　CORCON：内核控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	R/C-0	R/W-0	U-0	U-0
—	—	—	—	IPL3	PSV	—	—
bit 7							bit 0

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15-4　**未实现：**读为 0

bit 3　**IPL3：**CPU 中断优先级状态位 **(1)**
1 = CPU 中断优先级大于 7
0 = CPU 中断优先级等于或小于 7

bit 2　**PSV：**数据空间中的程序空间可视化使能位
1 = 程序空间在数据空间中可视
0 = 程序空间在数据空间中不可视

bit 1-0　**未实现：**读为 0

注　1：当 IPL3 = 1 时，禁止用户中断。

2.3 算术逻辑单元 （ALU）

PIC24F ALU 为 16 位宽，能够进行加、减、移位和逻辑运算。除非另外说明，算术运算一般是以 2 的补码方式进行的。根据进行的运算，ALU 可能会影响 SR 寄存器中的进位标志位 （C）、全零标志位 （Z）、负标志位（N）、溢出标志位 （OV）和半进位标志位（DC）的值。在减法操作中， C 和 DC 位分别作为借位和半借位标志位。

根据所使用的指令模式， ALU 可以执行 8 位或 16 位运算。根据指令的寻址模式，ALU 运算的数据可以来自 W 寄存器阵列或数据存储器。同样，ALU 的输出数据可以被写入 W 寄存器阵列或数据存储单元。

PIC24F CPU 具有支持乘法和除法运算的硬件电路，包括专用的硬件乘法器和支持 16 位除法的硬件电路。

2.3.1 乘法器

ALU 包含一个高速的 17 位 x 17 位乘法器。它支持无符号、有符号或混合符号运算等多种乘法模式，其中包括：

1. 16 位 x 16 位有符号乘法
2. 16 位 x 16 位无符号乘法
3. 有符号的 16 位 x 无符号的 5 位 （立即数）
4. 无符号的 16 位 x 无符号的 16 位
5. 无符号的 16 位 x 无符号的 5 位 （立即数）
6. 无符号的 16 位 x 有符号的 16 位
7. 无符号的 8 位 x 无符号的 8 位

2.3.2 除法器

除法运算模块支持有符号和无符号的 32 位 /16 位和 16 位 /16 位整数除法，数据长度为：

1. 有符号的 32 位 / 有符号的 16 位
2. 无符号的 32 位 / 无符号的 16 位
3. 有符号的 16 位 / 有符号的 16 位
4. 无符号的 16 位 / 无符号的 16 位

所有除法指令的商都被放在 W0 中，余数放在 W1 中。有符号和无符号的 16 位 DIV 指令可以为 16 位除数指定任何的 W 寄存器 （Wn），为 32 位被除数指定任何的 W 寄存器对 （对齐的）（W(m+1):Wm）。除法运算每执行 1 位除数需要一个周期，所以 32 位 /16 位和 16 位 /16 位指令执行的周期数相同。

2.3.3 支持多位移位

PIC24F ALU 支持单位移位、单周期移位、多位算术移位和多位逻辑移位。由一个移位器模块执行多位移位，在一个单周期内可执行最多 15 位算术右移或左移。所有的多位移位指令仅支持源操作数和结果寄存器直接寻址模式。

在下面的表 2-2 中总结了所有用于移位运算的指令。

表 2-2： 用于单位和多位移位运算的指令

指令	说明
ASR	源寄存器算术右移一位或多位。
SL	源寄存器左移一位或多位。
LSR	源寄存器逻辑右移一位或多位。

3.0 存储器构成

作为哈佛架构器件， PIC24F 单片机具有独立的程序和数据存储空间和总线。哈佛架构允许在代码执行期间直接通过数据空间访问程序存储器。

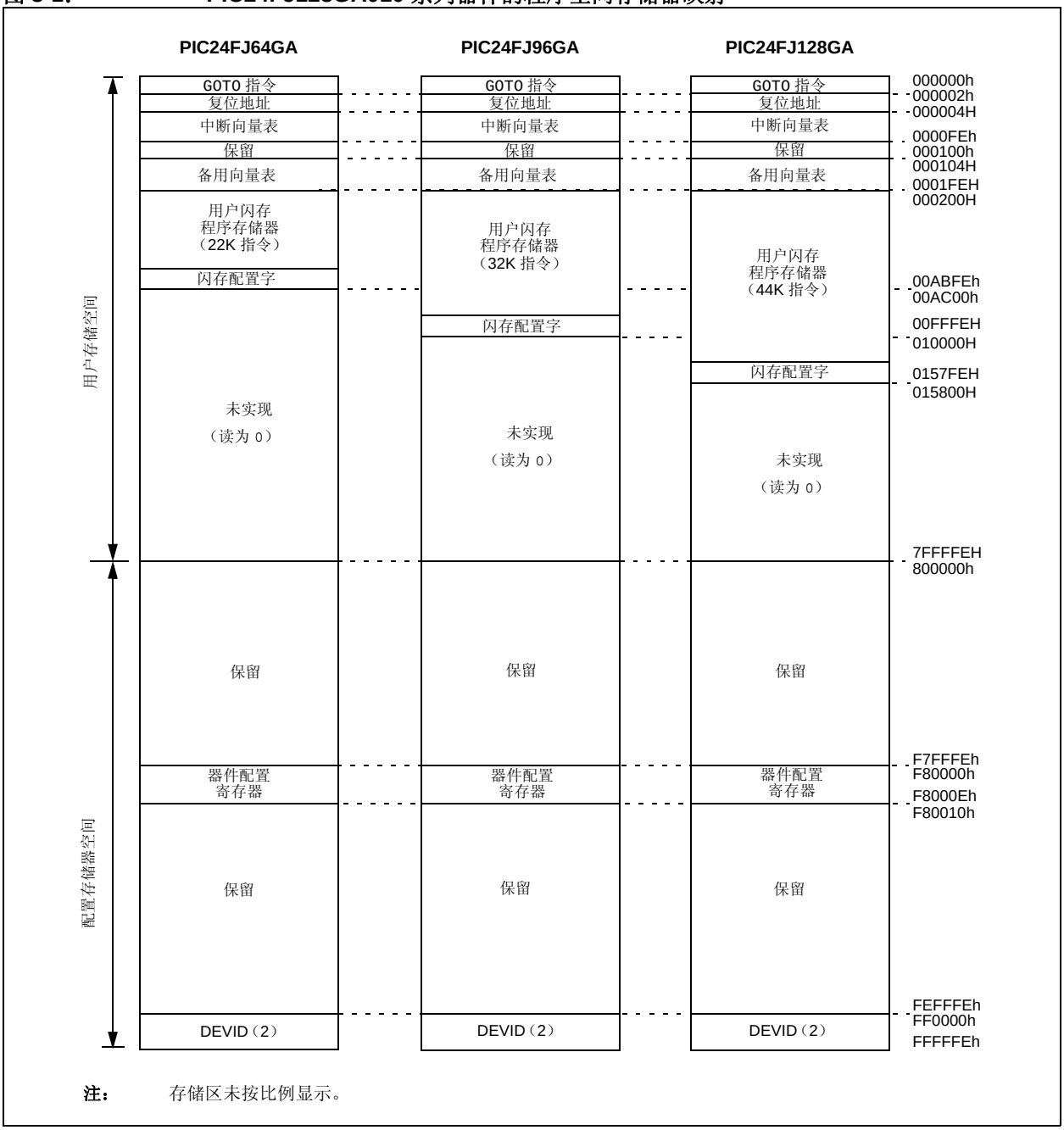
3.1 程序地址空间

PIC24FJ128GA010 系列器件的程序存储器空间可存储 4M 条指令。可通过 24 位地址值寻址程序空间，该地址值可以来自程序执行期间的 23 位程序计数器 （PC），或来自表操作或数据空间重映射，如**第 3.3 节“程序和数据存储空间的接口”**所述。

用户只能访问程序存储空间的低半部分 （地址范围为 000000h 到 7FFFFFh）。TBLRD/TBLWT 指令除外，它们使用 TBLPAG<7> 来访问配置存储空间的配置位和器件 ID 部分。

PIC24FJ128GA010 系列器件的存储器映射如图 3-1 所示。

图 3-1： PIC24FJ128GA010 系列器件的程序空间存储器映射



注： 存储区未按比例显示。

3.1.1 程序存储器构成

程序存储空间以字寻址块为单位构成。虽然存储器被认为是 24 位宽，但把程序存储器的每个地址看作高低字更加合理，其中高字的高字节未使用。低字地址始终为偶地址，而高字地址为奇地址（图 3-2）。

程序存储器地址始终在低字上是字对齐的，在代码执行时地址以 2 为单位递增或递减。这种地址构成方式与数据存储空间寻址方式兼容，从而能在程序存储空间内访问数据。

3.1.2 硬存储器向量

所有 PIC24F 器件都将 000000h 到 000200h 之间的地址单元作为保留空间，用于存放硬编码程序执行向量。硬件复位向量能在器件复位时将代码执行从默认的 PC 值重定位到程序实际开始处。用户在000000h处放置了一条 GOTO 指令使实际代码开始地址在 000002h。

PIC24F 器件具有两个中断向量表，分别位于 000004h 到 0000FFh 和 000100h 到 0001FFh 地址单元。这两个中断向量表保证每个器件中断源都有自己对应的中断服务程序。**第 6.1 节“中断矢量表”**详细讨论了中断向量表。

3.1.3 闪存配置字

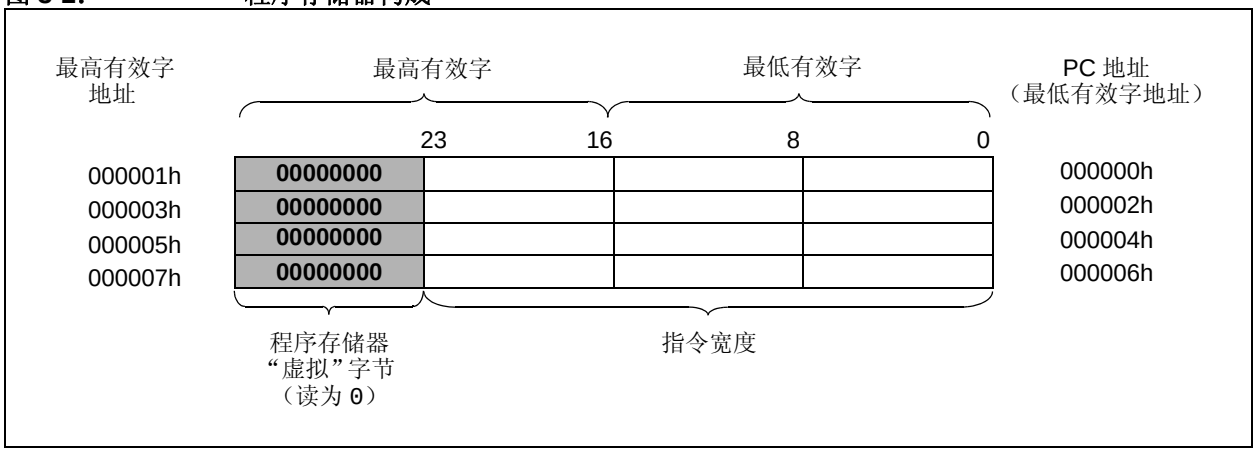
在 PIC24FJ128GA010 系列器件中，片内程序存储器最顶端的两个字被保留用于存放配置信息。当器件复位时，该配置信息被复制到相应的配置寄存器中。PIC24FJ128GA010 系列器件的闪存配置字的地址如表 3-1 所示。图 3-1 中显示了它们以及其他的存储器向量在存储器映射图中的位置。

程序存储器中的配置字是以紧凑格式存储的。实际配置位被映射在配置存储空间的不同寄存器中。它们在闪存配置字中的顺序并不代表它们在配置空间中实际的排列顺序。**第 23.1 节“配置位”**中提供了有关器件配置字的更多信息。

表 3-1： PIC24FJ128GA010 系列器件的闪存配置字

器件	程序存储器（字）	配置字地址
PIC24FJ64GA	22,016	00ABFCh：00ABFEh
PIC24FJ96GA	32,768	00FFFCh：00FFFEh
PIC24FJ128GA	44,032	0157FCh：0157FEh

图 3-2： 程序存储器构成



3.2 数据地址空间

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 3 章“数据存储器”**（DS39717A_CN）。

PIC24F 内核具有一个独立的 16 位宽数据存储空间，可线性寻址。使用两个地址发生单元（Address Generation Unit，AGU）对数据空间分别进行读写操作。数据空间存储器映射如图 3-3 所示。

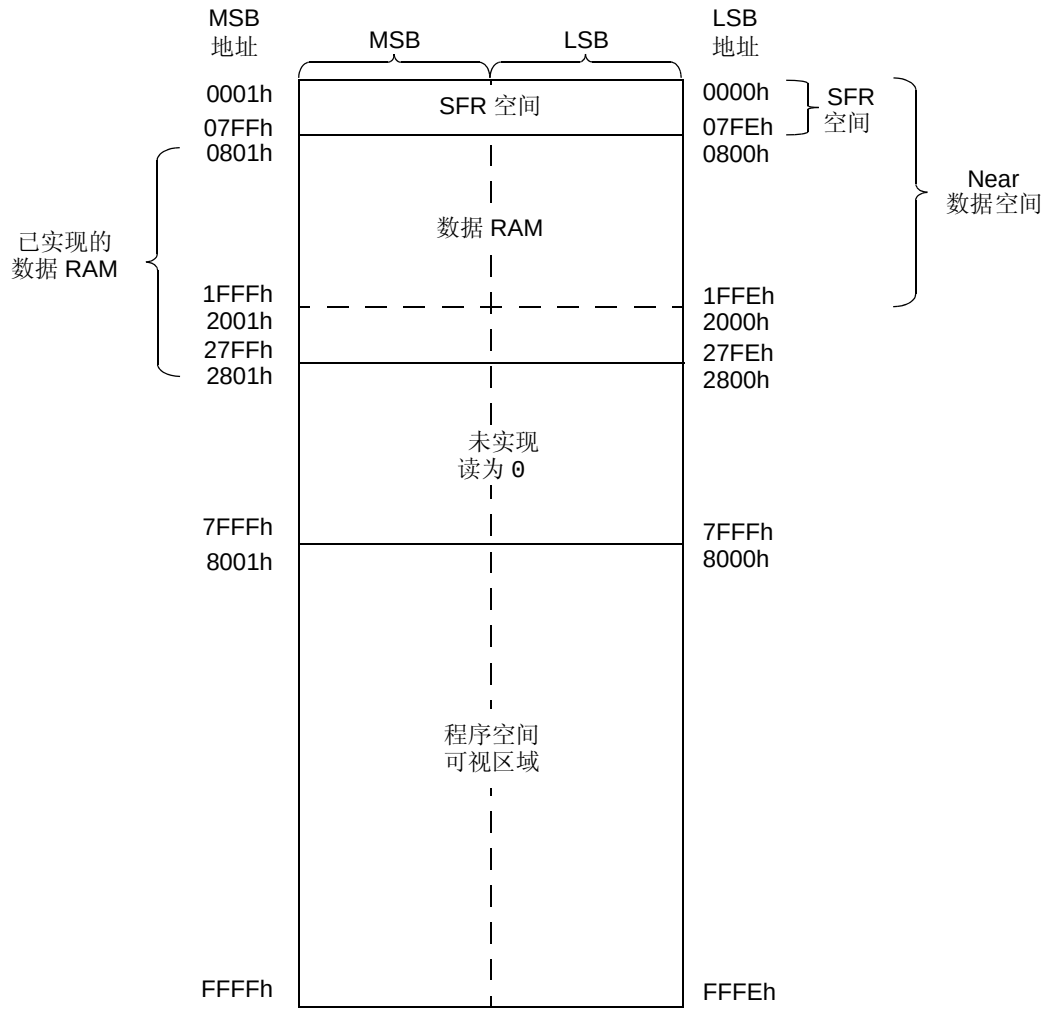
数据存储空间中的所有有效地址（Effective Addresse，EA）都为 16 位宽，指向数据存储器中的字节。这使得数据空间地址范围为 64 KB 或 32 千字。数据存储空间的低半部分（即当 EA<15> = 0）用作存放存储器地址，而高半部分（EA<15> = 1）保留为程序空间可见区域（见**第 3.3.3 节“使用程序空间可视化方法从程序存储器读取数据”**）。

PIC24FJ128GA010 系列器件总共实现了 8 KB 容量的数据存储器。一旦 EA 指向该区域外的地址，则会返回全零字或字节。

3.2.1 数据空间宽度

数据存储器是以 16 位宽数据块为单位构成的，可对字节寻址。数据存储器和寄存器中的数据以16位字对齐，但数据空间所有的 EA 都可被分解成字节。每个字的低字节具有偶地址，但高字节具有奇地址。

图 3-3： PIC24FJ128GA010 系列器件的数据空间存储器映射



注： 数据存储区未按比例显示。

3.2.2 数据存储器构成和对齐方式

为了与 PIC® 器件向下兼容并提高数据存储空间的利用率， PIC24F 指令集同时支持字操作和字节操作。为了得到字节访问操作的正确地址，所有的有效地址计算在内部都进行了调整。例如，对于后更改寄存器间接寻址模式（Post-Modified Indirect Addressing Mode）[Ws++] 来讲，内核认为 Ws+1 是字节操作的结果，而字操作的结果则是 Ws+2。

数据字节读操作将读取包含此字节的整个字，再使用有效地址的最低位（LSb）决定要选择哪个字节。选定字节被放入数据总线的低字节部分。即，数据存储器和寄存器以两个并列的字节实体构成，它们共享（字）地址译码，但写操作线是相互独立的。数据字节写操作只写入阵列或寄存器中与字节地址匹配的那一半。

所有字访问必须与一个偶地址对齐，不支持违背上述原则的存取访问。所以在字节和字的混合操作或转换 8 位 MCU 代码时必须十分小心。若尝试进行违背规定的读或写操作，则会产生地址错误陷阱。如果在读操作时发生该错误，则当前指令马上被停止执行；如果在写操作时发生该错误，指令会继续执行，但不会真正写入。不管哪种情况，都会执行陷阱程序，以允许系统和 / 或用户在执行地址故障恢复程序前检查机器状态。

所有装入W寄存器的字节都被装入其低字节。高字节不变。

符号扩展指令（SE）用于将 8 位有符号数据转换为 16 位有符号数据。而对于 16 位无符号数据，用户通过在相应地址上执行零扩展（ZE）指令可以清零任一 W 寄存器的高字节。

虽然大多数指令都可以对字或字节数据执行操作，但应该注意有些指令却只能针对字进行操作。

3.2.3 NEAR 数据空间

0000h 到 1FFFh 的 8 KB 区域被称为 Near 数据空间。可通过所有存储器直接寻址指令内的 13 位绝对地址字段对该空间内的地址单元直接寻址。数据空间的其余部分是间接寻址的。此外，使用 MOV 指令可寻址整个数据空间，该指令带有 16 位地址字段，支持存储器直接寻址模式。

3.2.4 SFR 空间

Near 数据空间的前 2 KB（从 0000h 到 07FFh）主要用作特殊功能寄存器（Special Function Register，SFR）。PIC24F 内核和外设模块使用这些特殊功能寄存器控制器件的操作。

SFR 分布在受其控制的模块中，通常按受控模块进行分组。许多 SFR 空间包含未实现地址；这些地址读为 0。SFR 空间的分布图显示了 SFR 所在的位置，如表 3-2 所示。每个实现的区域表示一个 32 字节的存储区，其中至少有一个地址单元被用作 SFR。表 3-3 到表 3-30 完整地列出了已实现的 SFR 以及它们的地址。

表 3-2： SFR 数据空间的实现区域

	SFR 空间地址							
	xx00	**xx20**	**xx40**	**xx60**	**xx80**	**xxA0**	**xxC0**	**xxE0**
000h	内核			ICN	中断			—
100h	定时器		捕捉	—	比较	—	—	—
200h	I²C™	UART	SPI		—	—	I/O	
300h	A/D		—	—	—	—	—	—
400h	—	—	—	—	—	—	—	—
500h	—	—	—	—	—	—	—	—
600h	PMP	RTC/ 比较	CRC	—	—	—	I/O	
700h	—	—	系统	NVM/PMD	—	—	—	—

图注： — = 该区域没有 SFR

表 3-3：　　CPU 内核寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
WREG0	0000	工作寄存器 0																0000
WREG1	0002	工作寄存器 1																0000
WREG2	0004	工作寄存器 2																0000
WREG3	0006	工作寄存器 3																0000
WREG4	0008	工作寄存器 4																0000
WREG5	000A	工作寄存器 5																0000
WREG6	000C	工作寄存器 6																0000
WREG7	000E	工作寄存器 7																0000
WREG8	0010	工作寄存器 8																0000
WREG9	0012	工作寄存器 9																0000
WREG10	0014	工作寄存器 10																0000
WREG11	0016	工作寄存器 11																0000
WREG12	0018	工作寄存器 12																0000
WREG13	001A	工作寄存器 13																0000
WREG14	001C	工作寄存器 14																0000
WREG15	001E	工作寄存器 15																0800
SPLIM	0020	堆栈指针边界																xxxx
PCL	002E	程序计数器的低字																0000
PCH	0030	—	—	—	—	—	—	—	—	程序计数器的高字节								0000
TBLPAG	0032	—	—	—	—	—	—	—	—	表页地址指针								0000
PSVPAG	0034	—	—	—	—	—	—	—	—	程序存储器可视性页地址指针								0000
RCOUNT	0036	REPEAT 循环计数器																xxxx
SR	0042	—	—	—	—	—	—	—	DC	IPL2	IPL1	IPL0	RA	N	OV	Z	C	0000
CORCON	0044	—	—	—	—	—	—	—	—	—	—	—	—	IPL3	PSV	—	—	0000
DISICNT	0052	—	—	禁止中断计数器														xxxx

图注：　　x = 复位时未知，— = 未实现 （读为 0）。复位值以十六进制显示。

表 3-4：　中断控制器寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
INTCON1	0080	NSTDIS	—	—	—	—	—	—	—	—	—	—	MATHERR	ADDRERR	STKERR	OSCFAIL	—	0000
INTCON2	0082	ALTIVT	DISI	—	—	—	—	—	—	—	—	—	INT4EP	INT3EP	INT2EP	INT1EP	INT0EP	0000
IFS0	0084	—	—	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPF1IF	T3IF	T2IF	OC2IF	IC2IF	—	T1IF	OC1IF	IC1IF	INT0IF	0000
IFS1	0086	U2TXIF	U2RXIF	INT2IF	T5IF	T4IF	OC4IF	OC3IF	—	—	—	—	INT1IF	CNIF	CMIF	MI2C1IF	SI2C1IF	0000
IFS2	0088	—	—	PMPIF	—	—	—	OC5IF	—	IC5IF	IC4IF	IC3IF	—	—	—	SPI2IF	SPF2IF	0000
IFS3	008A	—	RTCIF	—	—	—	—	—	—	—	INT4IF	INT3IF	—	—	MI2C2IF	SI2C2IF	—	0000
IFS4	008C	—	—	—	—	—	—	—	—	—	—	—	—	CRCIF	U2ERIF	U1ERIF	—	0000
IEC0	0094	—	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPF1IE	T3IE	T2IE	OC2IE	IC2IE	—	T1IE	OC1IE	IC1IE	INT0IE	0000
IEC1	0096	U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	—	—	—	—	INT1IE	CNIE	CMIE	MI2C1IE	SI2C1IE	0000
IEC2	0098	—	—	PMPIE	—	—	—	OC5IE	—	IC5IE	IC4IE	IC3IE	—	—	—	SPI2IE	SPF2IE	0000
IEC3	009A	—	RTCIE	—	—	—	—	—	—	—	INT4IE	INT3IE	—	—	MI2C2IE	SI2C2IE	—	0000
IEC4	009C	—	—	—	—	—	—	—	—	—	—	—	—	CRCIE	U2ERIE	U1ERIE	—	0000
IPC0	00A4	—	T1IP2	T1IP1	T1IP0	—	OC1IP2	OC1IP1	OC1IP0	—	IC1IP2	IC1IP1	IC1IP0	—	INT0IP2	INT0IP1	INT0IP0	4444
IPC1	00A6	—	T2IP2	T2IP1	T2IP0	—	OC2IP2	OC2IP1	OC2IP0	—	IC2IP2	IC2IP1	IC2IP0	—	—	—	—	4440
IPC2	00A8	—	U1RXIP2	U1RXIP1	U1RXIP0	—	SPI1IP2	SPI1IP1	SPI1IP0	—	SPF1IP2	SPF1IP1	SPF1IP0	—	T3IP2	T3IP1	T3IP0	4444
IPC3	00AA	—	—	—	—	—	—	—	—	—	AD1IP2	AD1IP1	AD1IP0	—	U1TXIP2	U1TXIP1	U1TXIP0	0044
IPC4	00AC	—	CNIP2	CNIP1	CNIP0	—	CMIP2	CMIP1	CMIP0	—	MI2C1IP2	MI2C1IP1	MI2C1IP0	—	SI2C1IP2	SI2C1IP1	SI2C1IP0	4444
IPC5	00AE	—	—	—	—	—	—	—	—	—	—	—	—	—	INT1IP2	INT1IP1	INT1IP0	0004
IPC6	00B0	—	T4IP2	T4IP1	T4IP0	—	OC4IP2	OC4IP1	OC4IP0	—	OC3IP2	OC3IP1	OC3IP0	—	—	—	—	4440
IPC7	00B2	—	U2TXIP2	U2TXIP1	U2TXIP0	—	U2RXIP2	U2RXIP1	U2RXIP0	—	INT2IP2	INT2IP1	INT2IP0	—	T5IP2	T5IP1	T5IP0	4444
IPC8	00B4	—	—	—	—	—	—	—	—	—	SPI2IP2	SPI2IP1	SPI2IP0	—	SPF2IP2	SPF2IP1	SPF2IP0	0044
IPC9	00B6	—	IC5IP2	IC5IP1	IC5IP0	—	IC4IP2	IC4IP1	IC4IP0	—	IC3IP2	IC3IP1	IC3IP0	—	—	—	—	4440
IPC10	00B8	—	—	—	—	—	—	—	—	—	OC5IP2	OC5IP1	OC5IP0	—	—	—	—	0040
IPC11	00BA	—	—	—	—	—	—	—	—	—	PMPIP2	PMPIP1	PMPIP0	—	—	—	—	0040
IPC12	00BC	—	—	—	—	—	MI2C2IP2	MI2C2IP1	MI2C2IP0	—	SI2C2IP2	SI2C2IP1	SI2C2IP0	—	—	—	—	0440
IPC13	00BE	—	—	—	—	—	INT4IP2	INT4IP1	INT4IP0	—	INT3IP2	INT3IP1	INT3IP0	—	—	—	—	0440
IPC15	00C2	—	—	—	—	—	RTCIP2	RTCIP1	RTCIP0	—	—	—	—	—	—	—	—	0400
IPC16	00C4	—	CRCIP2	CRCIP1	CRCIP0	—	U2ERIP2	U2ERIP1	U2ERIP0	—	U1ERIP2	U1ERIP1	U1ERIP0	—	—	—	—	4440

图注：　— = 未实现 （读为 0）。复位值以十六进制显示。

表 3-5：　　ICN 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
CNEN1	0060	CN15IE	CN14IE	CN13IE	CN12IE	CN11IE	CN10IE	CN9IE	CN8IE	CN7IE	CN6IE	CN5IE	CN4IE	CN3IE	CN2IE	CN1IE	CN0IE	0000
CNEN2	0062	—	—	—	—	—	—	—	—	—	—	CN21IE(1)	CN20IE(1)	CN19IE(1)	CN21IE(1)	CN17IE	CN16IE	0000
CNPU1	0068	CN15PUE	CN14PUE	CN13PUE	CN12PUE	CN11PUE	CN10PUE	CN9PUE	CN8PUE	CN7PUE	CN6PUE	CN5PUE	CN4PUE	CN3PUE	CN5PUE	CN1PUE	CN0PUE	0000
CNPU2	006A	—	—	—	—	—	—	—	—	—	—	CN21PUE(1)	CN20PUE(1)	CN19PUE(1)	CN21PUE(1)	CN17PUE	CN16PUE	0000

图注：　　— = 未实现 （读为 0）。复位值以十六进制显示。
注　　1：　仅在 80 引脚和 100 引脚器件上实现。

表 3-6：　　定时器寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
TMR1	0100	Timer1 寄存器																xxxx
PR1	0102	周期寄存器 1																FFFF
T1CON	0104	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	TSYNC	TCS	—	0000
TMR2	0106	Timer2 寄存器																xxxx
TMR3HLD	0108	Timer3 保持寄存器 （仅用于 32 位定时器操作）																xxxx
TMR3	010A	Timer3 寄存器																xxxx
PR2	010C	周期寄存器 2																FFFF
PR3	010E	周期寄存器 3																FFFF
T2CON	0110	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	T32	—	TCS	—	0000
T3CON	0112	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	—	TCS	—	0000
TMR4	0114	Timer4 寄存器																xxxx
TMR5HLD	0116	Timer5 保持寄存器 （仅用于 32 位定时器操作）																xxxx
TMR5	0118	Timer5 寄存器																xxxx
PR4	011A	周期寄存器 4																FFFF
PR5	011C	周期寄存器 5																FFFF
T4CON	011E	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	T32	—	TCS	—	0000
T5CON	0120	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	—	TCS	—	0000

图注：　　x = 复位时未知，— = 未实现 （读为 0）。复位值以十六进制显示。

表 3-7：　　输入捕捉寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
IC1BUF	0140	输入 1 捕捉寄存器																xxxx
IC1CON	0142	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC2BUF	0144	输入 2 捕捉寄存器																xxxx
IC2CON	0146	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC3BUF	0148	输入 3 捕捉寄存器																xxxx
IC3CON	014A	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC4BUF	014C	输入 4 捕捉寄存器																xxxx
IC4CON	014E	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC5BUF	0150	输入 5 捕捉寄存器																xxxx
IC5CON	0152	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000

图注：　x = 复位时未知，— = 未实现 （读为 0）。复位值以十六进制显示。

表 3-8：　　输出比较寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
OC1RS	0180	输出比较 1 辅助寄存器																xxxx
OC1R	0182	输出比较 1 寄存器																xxxx
OC1CON	0184	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000
OC2RS	0186	输出比较 2 辅助寄存器																xxxx
OC2R	0188	输出比较 2 寄存器																xxxx
OC2CON	018A	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000
OC3RS	018C	输出比较 3 辅助寄存器																xxxx
OC3R	018E	输出比较 3 寄存器																xxxx
OC3CON	0190	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000
OC4RS	0192	输出比较 4 辅助寄存器																xxxx
OC4R	0194	输出比较 4 寄存器																xxxx
OC4CON	0196	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000
OC5RS	0198	输出比较 5 辅助寄存器																xxxx
OC5R	019A	输出比较 5 寄存器																xxxx
OC5CON	019C	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000

图注：　x = 复位时未知，— = 未实现 （读为 0）。复位值以十六进制显示。

表 3-9： I2C1 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
I2C1RCV	0200	—	—	—	—	—	—	—	—	接收寄存器								0000
I2C1TRN	0202	—	—	—	—	—	—	—	—	发送寄存器								00FF
I2C1BRG	0204	—	—	—	—	—	—	—	波特率发生器									0000
I2C1CON	0206	I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN	GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	1000
I2C1STAT	0208	ACK-STAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10	IWCOL	I2COV	D/A	P	S	R/$\overline{W}$	RBF	TBF	0000
I2C1ADD	020A	—	—	—	—	—	—	地址寄存器										0000
I2C1MSK	020C	—	—	—	—	—	—	地址屏蔽										0000

图注： — = 未实现 （读为 0）。复位值以十六进制显示。

表 3-10： I2C2 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
I2C2RCV	0210	—	—	—	—	—	—	—	—	接收寄存器								0000
I2C2TRN	0212	—	—	—	—	—	—	—	—	发送寄存器								00FF
I2C2BRG	0214	—	—	—	—	—	—	—	波特率发生器									0000
I2C2CON	0216	I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN	GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	1000
I2C2STAT	0218	ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10	IWCOL	I2CPOV	D/A	P	S	R/$\overline{W}$	RBF	TBF	0000
I2C2ADD	021A	—	—	—	—	—	—	地址寄存器										0000
I2C2MSK	021C	—	—	—	—	—	—	地址屏蔽										0000

图注： — = 未实现 （读为 0）。复位值以十六进制显示。

表 3-11： UART1 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
U1MODE	0220	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
U1STA	0222	UTXISEL1	TXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U1TXREG	0224	—	—	—	—	—	—	—	发送寄存器									xxxx
U1RXREG	0226	—	—	—	—	—	—	—	接收寄存器									0000
U1BRG	0228	波特率发生器预分频器																0000

图注： x = 复位时未知，— = 未实现 （读为 0）。复位值以十六进制显示。

表 3-12： UART2 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
U2MODE	0230	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
U2STA	0232	UTXISEL1	TXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U2TXREG	0234	—	—	—	—	—	—	—	发送寄存器									xxxx
U2RXREG	0236	—	—	—	—	—	—	—	接收寄存器									0000
U2BRG	0238	波特率发生器预分频器																0000

图注： x = 复位时未知，— = 未实现 （读为 0）。复位值以十六进制显示。

表 3-13： SPI1 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
SPI1STAT	0240	SPIEN	—	SPISIDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0	SRMPT	SPIROV	SRXMPT	SISEL2	SISEL1	SISEL0	SPITBF	SPIRBF	0000
SPI1CON1	0242	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0	0000
SPI1CON2	0244	FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—	—	—	—	—	—	—	SPIFE	SPIBEN	0000
SPI1BUF	0248	SPI1 发送和接收缓冲器																0000

图注： — = 未实现 （读为 0）。复位值以十六进制显示。

表 3-14： SPI2 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
SPI2STAT	0260	SPIEN	—	SPISIDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0	SRMPT	SPIROV	SRXMPT	SISEL2	SISEL1	SISEL0	SPITBF	SPIRBF	0000
SPI2CON1	0262	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0	0000
SPI2CON2	0264	FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—	—	—	—	—	—	—	SPIFE	SPIBEN	0000
SPI2BUF	0268	SPI2 发送和接收缓冲器																0000

图注： — = 未实现 （读为 0）。复位值以十六进制显示。

表 3-15：　ADC 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
ADC1BUF0	0300	ADC 数据缓冲器 0																xxxx
ADC1BUF1	0302	ADC 数据缓冲器 1																xxxx
ADC1BUF2	0304	ADC 数据缓冲器 2																xxxx
ADC1BUF3	0306	ADC 数据缓冲器 3																xxxx
ADC1BUF4	0308	ADC 数据缓冲器 4																xxxx
ADC1BUF5	030A	ADC 数据缓冲器 5																xxxx
ADC1BUF6	030C	ADC 数据缓冲器 6																xxxx
ADC1BUF7	030E	ADC 数据缓冲器 7																xxxx
ADC1BUF8	0310	ADC 数据缓冲器 8																xxxx
ADC1BUF9	0312	ADC 数据缓冲器 9																xxxx
ADC1BUFA	0314	ADC 数据缓冲器 10																xxxx
ADC1BUFB	0316	ADC 数据缓冲器 11																xxxx
ADC1BUFC	0318	ADC 数据缓冲器 12																xxxx
ADC1BUFD	031A	ADC 数据缓冲器 13																xxxx
ADC1BUFE	031C	ADC 数据缓冲器 14																xxxx
ADC1BUFF	031E	ADC 数据缓冲器 15																xxxx
AD1CON1	0320	ADON	—	ADSIDL	—	—	—	FORM1	FORM0	SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE	0000
AD1CON2	0322	VCFG2	VCFG1	VCFG0	r	—	CSCNA	—	—	BUFS	—	SMPI3	SMPI2	SMPI1	SMPI0	BUFM	ALTS	0000
AD1CON3	0324	ADRC	—	—	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0	ADCS7	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0000
AD1CHS	0328	CH0NB	—	—	—	CH0SB3	CH0SB2	CH0SB1	CH0SB0	CH0NA	—	—	—	CH0SA3	CH0SA2	CH0SA1	CH0SA0	0000
AD1PCFG	032C	PCFG15	PCFG14	PCFG13	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000
AD1CSSL	0330	CSSL15	CSSL14	CSSL13	CSSL12	CSSL11	CSSL10	CSSL9	CSSL8	CSSL7	CSSL6	CSSL5	CSSL4	CSSL3	CSSL2	CSSL1	CSSL0	0000

图注：　x = 复位时未知，— = 未实现（读为 0）。复位值以十六进制显示。r = 保留，保持为 0。复位值以十六进制表示。

表 3-16：　PORTA 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
TRISA	02C0	TRISA15(1)	TRISA14(1)	–	–	–	TRISA10(1)	TRISA9(1)	–	TRISA7	TRISA6	TRISA5(2)	TRISA4(2)	TRISA3(2)	TRISA2(2)	TRISA1(2)	TRISA0(2)	C6FF
PORTA	02C2	RA15(1)	RA14(1)	–	–	–	RA10(1)	RA9(1)	–	RA7	RA6	RA5(2)	RA4(2)	RA3(2)	RA2(2)	RA1(2)	RA0(2)	xxxx
LATA	02C4	LATA15(1)	LATA14(1)	–	–	–	LATA10(1)	LATA9(1)	–	LATA7	LATA6	LATA5(2)	LATA4(2)	LATA3(2)	LATA2(2)	LATA1(2)	LATA0(2)	xxxx
ODCA	06C0	ODA15(1)	ODA14(1)	–	–	–	ODA10(1)	ODA9(1)	–	ODA7)	ODA6	ODA5(2)	ODA4(2)	ODA3(2)	ODA2(2)	ODA1(2)	ODA0(2)	0000

图注：　x = 复位时未知，— = 未实现（读为 0）。 100 引脚器件的复位值以十六进制显示。

注　1：　只在 80 引脚和 100 引脚器件中实现。

2：　只在 100 引脚器件中实现。

表 3-17： PORTB 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
TRISB	02C6	TRISB15	TRISB14	TRISB13(1)	TRISB12(1)	TRISB11(1)	TRISB10(1)	TRISB9	TRISB8	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	FFFF
PORTB	02C8	RB15	RB14	RB13(1)	RB12(1)	RB11(1)	RB10(1)	RB9	RB8	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxx
LATB	02CA	LATB15	LATB14	LATB13(1)	LATB12(1)	LATB11(1)	LATB10(1)	LATB9	LATB8	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx
ODCB	06C6	ODB15	ODB14	ODB13(1)	ODB12(1)	ODB11(1)	ODB10(1)	ODB9	ODB8	ODB7	ODB6	ODB5	ODB4	ODB3	ODB2	ODB1	ODB0	0000

图注： x = 复位时未知，— = 未实现（读为 0）。100 引脚器件的复位值以十六进制显示。

注 1： 当 JTAG 使能时未实现。

表 3-18： PORTC 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
TRISC	02CC	TRISC15	TRISC14	TRISC13	TRISC12	—	—	—	—	—	—	—	TRISC4(2)	TRISC3(1)	TRISC2(2)	TRISC1(1)	—	F01E
PORTC	02CE	RC15	RC14	RC13	RC12	—	—	—	—	—	—	—	RC4(2)	RC3(1)	RC2(2)	RC1(1)	—	xxxx
LATC	02D0	LATC15	LATC14	LATC13	LATC12	—	—	—	—	—	—	—	LATC4(2)	LATC3(1)	LATC2(2)	LATC1(1)	—	xxxx
ODCC	06CC	ODC15	ODC14	ODC13	ODC12	—	—	—	—	—	—	—	ODC4(2)	ODC3(1)	ODC2(2)	ODC1(1)	—	0000

图注： x = 复位时未知，— = 未实现（读为 0）。100 引脚器件的复位值以十六进制显示。

注 1： 只在 80 引脚和 100 引脚器件中实现。

2： 只在 100 引脚器件中实现。

表 3-19： PORTD 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
TRISD	02D2	TRISD15(1)	TRISD14(1)	TRISD13(1)	TRISD12(1)	TRISD11	TRISD10	TRISD9	TRISD8	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0	FFFF
PORTD	02D4	RD15(1)	RD14(1)	RD13(1)	RD12(1)	RD11	RD10	RD9	RD8	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxx
LATD	02D6	LATD15(1)	LATD14(1)	LATD13(1)	LATD12(1)	LATD11	LATD10	LATD9	LATD8	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0	xxxx
ODCD	06D2	ODD15(1)	ODD14(1)	ODD13(1)	ODD12(1)	ODD11	ODD10	ODD9	ODD8	ODD7	ODD6	ODD5	ODD4	ODD3	ODD2	ODD1	ODD0	0000

图注： x = 复位时未知，— = 未实现（读为 0）。100 引脚器件的复位值以十六进制显示。

注 1： 只在 80 引脚和 100 引脚器件中实现。

表 3-20： PORTE 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
TRISE	02D8	—	—	—	—	—	—	TRISE9(1)	TRISE8(1)	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0	03FF
PORTE	02DA	—	—	—	—	—	—	RE9(1)	RE8(1)	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0	xxxx
LATE	02DC	—	—	—	—	—	—	LATE9(1)	LATE8(1)	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0	xxxx
ODCE	06D8	—	—	—	—	—	—	ODE9(1)	ODE8(1)	ODE7	ODE6	ODE5	ODE4	ODE3	ODE2	ODE1	ODE0	0000

图注： x = 复位时未知，— = 未实现（读为 0）。100 引脚器件的复位值以十六进制显示。
注 1： 只在 80 引脚和 100 引脚器件中实现。

表 3-21： PORTF 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
TRISF	02DE	—	—	TRISF13(1)	TRISF12(1)	—	—	—	TRISF8(2)	TRISF7(2)	TRISF6	TRISF5	TRISF4	TRISF3	TRISF2	TRISF1	TRISF0	31FF
PORTF	02E0	—	—	RG13(1)	RG12(1)	—	—	—	RF8(2)	RF7(2)	RF6	RF5	RF4	RF3	RF2	RF1	RF0	xxxx
LATF	02E2	—	—	LATF13(1)	LATF12(1)	—	—	—	LATF8(2)	LATF7(2)	LATF6	LATF5	LATF4	LATF3	LATF2	LATF1	LATF0	xxxx
ODCF	06DE	—	—	ODF13(1)	ODF12(1)	—	—	—	ODF8(2)	ODF7(2)	ODF6	ODF5	ODF4	ODF3	ODF2	ODF1	ODF0	0000

图注： x = 复位时未知，— = 未实现（读为 0）。100 引脚器件的复位值以十六进制显示。
注 1： 只在 100 引脚器件中实现。
2： 只在 80 引脚和 100 引脚器件中实现。

表 3-22： PORTG 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
TRISG	02E4	TRISG15	TRISG14(1)	TRISG13(1)	TRISG12(1	—	—	TRISG9	TRISG8	TRISG7	TRISG6	—	—	TRISG3	TRISG2	TRISG1(2)	TRISG0(2)	F3CF
PORTG	02E6	RG15	RG14(1)	RG13(1)	RG12(1)	—	—	RG9	RG8	RG7	RG6	—	—	RG3	RG2	RG1(2)	RG0(2)	xxxx
LATG	02E8	LATG15	LATG14(1)	LATG13(1)	LATG12(1)	—	—	LATG9	LATG8	LATG7	LATG6	—	—	LATG3	LATG2	LATG1(2)	LATG0(2)	xxxx
ODCG	06E4	ODG15	ODG14(1)	ODG13(1)	ODG12(1)	—	—	ODG9	ODG8	ODG7	ODG6	—	—	ODG3	ODG2	ODG1(2)	ODG0(2)	0000

图注： x = 复位时未知，— = 未实现（读为 0）。100 引脚器件的复位值以十六进制显示。
注 1： 只在 100 引脚器件中实现。
2： 只在 80 引脚和 100 引脚器件中实现。

表 3-23： 填充配置寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
PADCFG1	02FC	—	—	—	—	—	—	—	—	—	—	—	—	—	—	RTSECSEL	PMPTTL	0000

图注： x = 复位时未知，— = 未实现（读为 0）。100 引脚器件的复位值以十六进制显示。

表 3-24：　并行主控 / 从动端口寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
PMCON	0600	PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN	CSF1	CSF0	ALP	CS2P	CS1P	BEP	WRSP	RDSP	0000
PMMODE	0602	BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0	WAITB1	WAITB0	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1	WAITE0	0000
PMADDR(1)	0604	CS2	CS1	并行端口目标地址 <13:0>（主控模式）														0000
PMDOUT1(1)		并行端口数据输出寄存器 1（缓冲器 0 到 1）																0000
PMDOUT2	0606	并行端口数据输出寄存器 2（缓冲器 2 到 3）																0000
PMDIN1	0608	并行端口数据输入寄存器 1（缓冲器 0 到 1）																0000
PMDIN2	060A	并行端口数据输入寄存器 2（缓冲器 2 到 3）																0000
PMAEN	060C	PTEN15	PTEN14	PTEN13	PTEN12	PTEN11	PTEN10	PTEN9	PTEN8	PTEN7	PTEN6	PTEN5	PTEN4	PTEN3	PTEN2	PTEN1	PTEN0	0000
PMSTAT	060E	IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F	OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E	008F

图注：　— = 未实现（读为 0）。复位值以十六进制显示。
注　1：　PMADDR 和 PMDOUT1 共享同一物理寄存器。在从动模式下该寄存器用作 PMDOUT1，在主控模式下该寄存器才可用作 PMADDR。

表 3-25：　实时时钟和日历寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
ALRMVAL	0620	ALRMPTR<1:0> 指定的闹钟值寄存器窗口																xxxx
ALCFGRPT	0622	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0	0000
RTCVAL	0624	RTCPTR<1:0> 指定的 RTCC 值寄存器窗口																xxxx
RCFGCAL(1)	0626	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	0000

图注：　x = 复位时未知，— = 未实现（读为 0）。复位值以十六进制显示。
注　1：　RCFGCAL 寄存器的复位值取决于复位类型。

表 3-26：　双比较器寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
CMCON	0630	CMIDL	—	C2EVT	C1EVT	C2EN	C1EN	C2OUTEN	C1OUTEN	C2OUT	C1OUT	C2INV	C1INV	C2NEG	C2POS	C1NEG	C1POS	0000
CVRCON	0632	—	—	—	—	—	—	—	—	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000

图注：　— = 未实现（读为 0）。复位值以十六进制显示。

表 3-27：　CRC 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
CRCCON	0640	£¦	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0	CRCFUL	CRCMPT	£¦	CRCGO	PLEN3	PLEN2	PLEN1	PLEN0	0000
CRCXOR	0642	CRC 异或多项式寄存器																0000
CRCDAT	0644	CRC 数据输入寄存器																0000
CRCWDAT	0646	CRC 结果寄存器																0000

图注：　— = 未实现 （读为 0）。复位值以十六进制显示。

表 3-28：　系统寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
RCON	0740	TRAPR	IOPUWR	—	—	—	—	CM	VREGS	EXTR	SWR	SWDTEN	WDTO	SLEEP	IDLE	BOR	POR	xxxx(1)
OSCCON	0742	—	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0	CLKLOCK	—	LOCK	—	CF	—	SOSCEN	OSWEN	xxxx(2)
CLKDIV	0744	ROI	DOZE2	DOZE1	DOZE0	DOZEN	RCDIV2	RCDIV1	RCDIV0	—	—	—	—	—	—	—	—	0100
OSCTUN	0748	—	—	—	—	—	—	—	—	—	—	TUN<5:0>						0000

图注：　x = 复位时未知，— = 未实现 （读为 0）。复位值以十六进制显示。

注　1：　RCON 寄存器的复位值取决于复位类型。

2：　OSCCON 寄存器的复位值取决于 FOSC 配置位和复位类型。

表 3-29：　NVM 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
NVMCON	0760	WR	WREN	WRERR	—	—	—	—	—	—	擦除	—	—	NVMOP3	NVMOP2	NVMOP1	NVMOP0	0000(1)
NVMKEY	0766	—	—	—	—	—	—	—	—	NVMKEY<7:0>								0000

图注：　— = 未实现 （读为 0）。复位值以十六进制显示。

注　1：　所示复位值仅为上电复位的值。其他复位值取决于复位时存储器写入或擦除操作的状态。

表 3-30：　PMD 寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值
PMD1	0770	T5MD	T4MD	T3MD	T2MD	T1MD	—	—	—	I2C1MD	U2MD	U1MD	SPI2MD	SPI1MD	—	—	ADC1MD	0000
PMD2	0772	—	—	—	IC5MD	IC4MD	IC3MD	IC2MD	IC1MD	—	—	—	OC5MD	OC4MD	OC3MD	OC2MD	OC1MD	0000
PMD3	0774	—	—	—	—	—	CMPMD	RTCCMD	PMPMD	CRCPMD	—	—	—	—	—	I2C2MD	—	0000

图注：　— = 未实现 （读为 0）。复位值以十六进制显示。

3.2.5 软件堆栈

PIC24F 器件中的 W15 寄存器除用作工作寄存器外，还用作软件堆栈指针。该指针始终指向第一个空字，并从低地址向高地址递增。出栈操作该指针预减 1，压栈操作该指针预加 1，如图 3-4 所示。注意对于任意 CALL 指令期间的 PC 压栈操作， PC 的高字节部分在压栈前要经过零扩展以保证高字节始终清零。

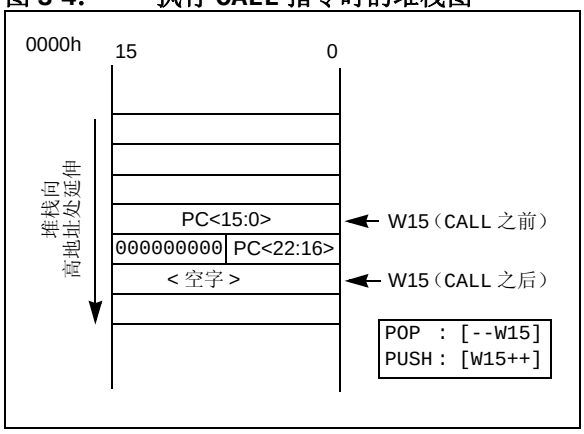
注：	异常事件处理期间的PC压栈操作会在操作前将SRL寄存器与PC的高字节构成一体。

与堆栈指针对应的栈指针边界寄存器（SPLIM）用于设置堆栈的高地址边界。复位时不会初始化SPLIM。由于与栈指针相关联， SPLIM<0> 被强制为 0，因为所有栈操作都是字对齐的。当用 W15 作为源或目标指针生成 EA 时，要将结果地址同 SPLIM 中的值做比较。如果栈指针 （W15）的内容与 SPLIM 寄存器的内容相等并执行了压栈操作，则不会产生堆栈错误陷阱，而是在随后的压栈操作时产生该陷阱。例如，如果希望在栈地址超过 2000h 时产生堆栈错误陷阱，则用 1FFEh 初始化 SPLIM。

同样，在栈指针地址小于 0800h 时会产生栈指针下溢（堆栈错误）陷阱，从而防止堆栈与特殊功能寄存器空间重叠。

对 SPLIM 寄存器执行写操作后，不要立即使用 W15 对该寄存器执行间接读操作。

图 3-4： 执行 CALL 指令时的堆栈图



3.3 程序和数据存储空间的接口

PIC24F架构使用24位宽程序空间和16位宽数据空间。该架构采用了改进的哈佛结构，即可在程序空间中存放数据。要成功使用这些数据，必须保证两个空间内的信息使用相同的对齐方式。

除了正常代码执行， PIC24F 架构提供了两种可以在操作过程中访问程序空间的方法：

- 使用表操作指令访问程序空间中的单独字节或字
- 将程序空间的一部分重新映射到数据空间 （程序空间可视化）

用户可以使用表操作指令来读写程序存储器中的一小块区域，因而该指令非常适合用来访问需要不断更新的数据表，或程序字的所有字节。重新映射的功能允许应用程序只读访问数据空间中的大块数据，这适用于查找大型静态数据表。但只能访问程序字中的较低位。

3.3.1 寻址程序空间

由于数据和程序空间的地址范围分别为 16 位和 24 位，所以需要使用一种方法用 16 位数据寄存器的内容生成 23 位或 24 位程序存储器地址。解决方案取决于使用的接口方式。

对于表操作，使用 8 位表页寄存器 （TBLPAG）定义程序空间中一个 32 千字区域。它与 16 位 EA 一起组成 24 位程序空间地址。在这种格式下，TBLPAG 的最高有效位决定操作是在用户存储器 （TBLPAG<7> = 0）内发生还是在配置存储器 （TBLPAG<7> = 1）内发生。

对于重新映射操作， 8 位程序空间可视页面寄存器（PSVPAG）用于定义程序空间中的 16 千字 （一页）区域。当 EA 的最高有效位为 1 时，PSVPAG 与 EA 的低 15 位一起，形成 23 位程序空间地址。与表操作不同，这种方法将重新映射操作严格地限制在用户存储区中。

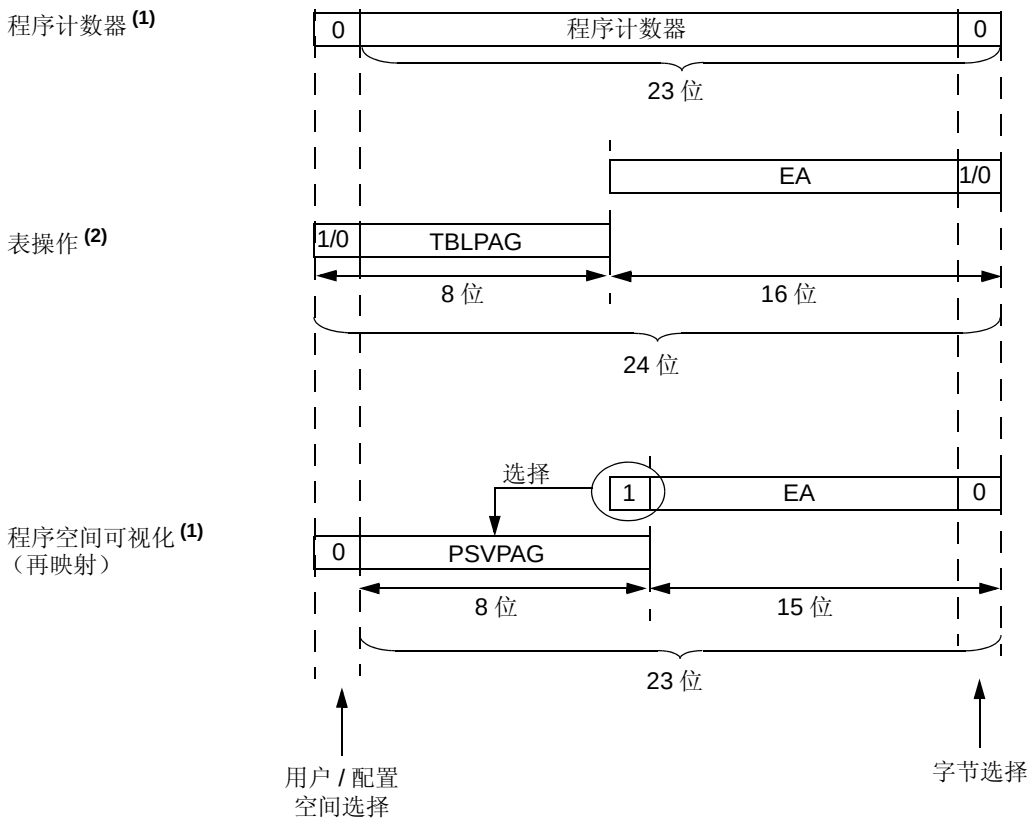
表 3-31和图 3-5说明了如何为表操作和重新映射访问操作从数据 EA 生成程序 EA 的方法。P<23:0> 指程序空间字，而 D<15:0> 指数据空间字。

表 3-31：　　程序空间地址构成

访问类型	访问空间	程序空间地址				
		<23>	<22:16>	<15>	<14:1>	<0>
指令访问（代码执行）	用户	0	PC<22:1>			0
		0xx xxxx xxxx xxxx xxxx xxx0				
TBLRD/TBLWT（字节或字读 / 写）	用户	TBLPAG<7:0>		数据 EA<15:0>		
		0xxx xxxx		xxxx xxxx xxxx xxxx		
	配置	TBLPAG<7:0>		数据 EA<15:0>		
		1xxx xxxx		xxxx xxxx xxxx xxxx		
程序空间可视化（块再映射 / 读）	用户	0	PSVPAG<7:0>		数据 EA<14:0>(1)	
		0	xxxx xxxx		xxx xxxx xxxx xxxx	

注　1： 在这种情况下数据 EA<15> 始终为 1，但不用于计算程序空间地址。地址的 bit 15 是 PSVPAG<0>。

图 3-5：　　用程序空间生成地址访问数据



注　1： 程序空间地址的最低有效位始终为 0，以保持程序空间和数据空间是字对齐的。

2： 表操作不需要字对齐。允许对配置存储空间执行表读操作。

3.3.2 使用表操作指令访问程序存储器

使用 TBLRDL 和 TBLWTL 指令能直接读写程序空间中任何地址的低字，而不必先经过数据空间。TBLRDL 和 TBLWTL指令是像访问数据那样读写程序空间中高8位的惟一方法。

对于连续的 24 位程序字，PC 每次会加 2，从而可将程序存储器地址直接映射到数据空间地址。这样就可以把程序存储器看作是两个 16 位字宽的地址空间，两部分空间边界对齐并且大小相同。TBLRDL 和 TBLWTL 访问包含最低有效数据字的空间，而 TBLRDH 和 TBLWTH 访问包含高数据字节的空间。

两个表操作指令用于将字节或字大小 （16 位）的数据移入 （出）程序空间。两种操作都是字节或字操作。

1. TBLRDL （表读低字）：在字模式下，指令将程序空间的低字单元 （P<15:0>）映射到数据地址（D<15:0>）。

   在字节模式下，程序空间低字中的高字节或低字节被映射到数据地址的低字节。当字节选择位为 1 时选择高字节，为 0 时选择低字节。

2. TBLRDH（表读高字）：在字模式下，指令将程序地址的整个高字（P<23:16> ）映射到数据地址。注意：D<15:8> “虚拟”字节始终为 0。

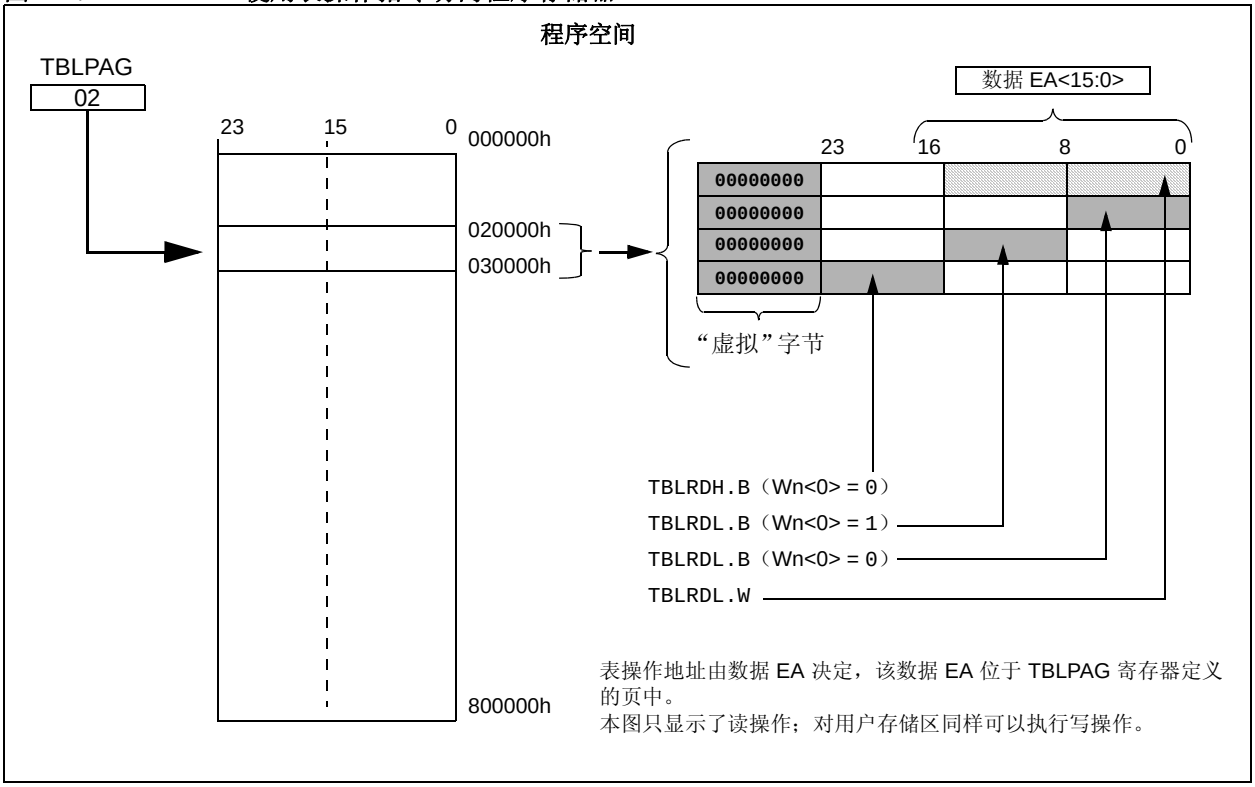
   在字节模式下，指令如前面一样将程序字的高或低字节映射到数据地址的 D<7:0>。注意：当选择高 “虚拟”字节时 （字节选择位 = 1）数据始终为 0。

两条表操作指令 TBLWTH 和 TBLWTL 以同样的方式将单独的字节或字写入程序空间地址。**第 4.0 节“闪存程序存储器”**详细说明了它们的操作原理。

对于所有的表操作，要访问哪部分程序存储空间是由表页寄存器（TBLPAG）决定的。TBLPAG 覆盖器件的整个程序存储空间，包括用户空间和配置空间。当 TBLPAG<7> = 0 时，表页位于用户存储空间中。当 TBLPAG<7> = 1 时，表页位于配置空间中。

注：	只能对配置存储器中已实现的区域 （如器件 ID）执行表读操作。不允许执行表写操作。

图 3-6： 使用表操作指令访问程序存储器



表操作地址由数据 EA 决定，该数据 EA 位于 TBLPAG 寄存器定义的页中。
本图只显示了读操作；对用户存储区同样可以执行写操作。

3.3.3 使用程序空间可视化方法从程序存储器读取数据

可以将数据空间中的高32 KB映射到程序空间的任意16千字页面上。这样无需使用特殊的指令（即 TBLRDL/H），就可以透明地访问储存在数据空间的常数数据。

如果数据空间 EA 的最高有效位为 1，并且通过将内核控制寄存器（CORCON<2>）中的 PSV 位置 1 使能程序空间可视化，则可通过数据空间访问程序空间。具体哪些程序存储单元要映射到数据空间是由程序空间可视页面寄存器（PSVPAG）决定的。该 8 位寄存器指定要映射到程序空间256个16千字页中的哪一个。实际上，PSVPAG 用作程序存储器地址的高 8 位， EA 的 15 位用作低位。注意：每访问一个程序字，PC 就加 2，数据空间的低15位直接映射到相应程序空间地址的低15位。

读该区域内的数据会使指令的执行时间多出一个周期，因为这种操作需要取两次程序字。

虽然每个数据空间中地址大于等于8000h的部分直接映射到相应的程序存储器地址（见图 3-7），但只有 24 位程序字的低 16 位用于存储数据。用作数据存储器的程序存储单元的高 8 位应编程为 1111 1111 或 0000 0000，以强制作为一条NOP指令。这可阻止意外执行该区域内的代码时导致不可预料的后果。

注： 在表读 / 写操作时暂时禁止访问 PSV。

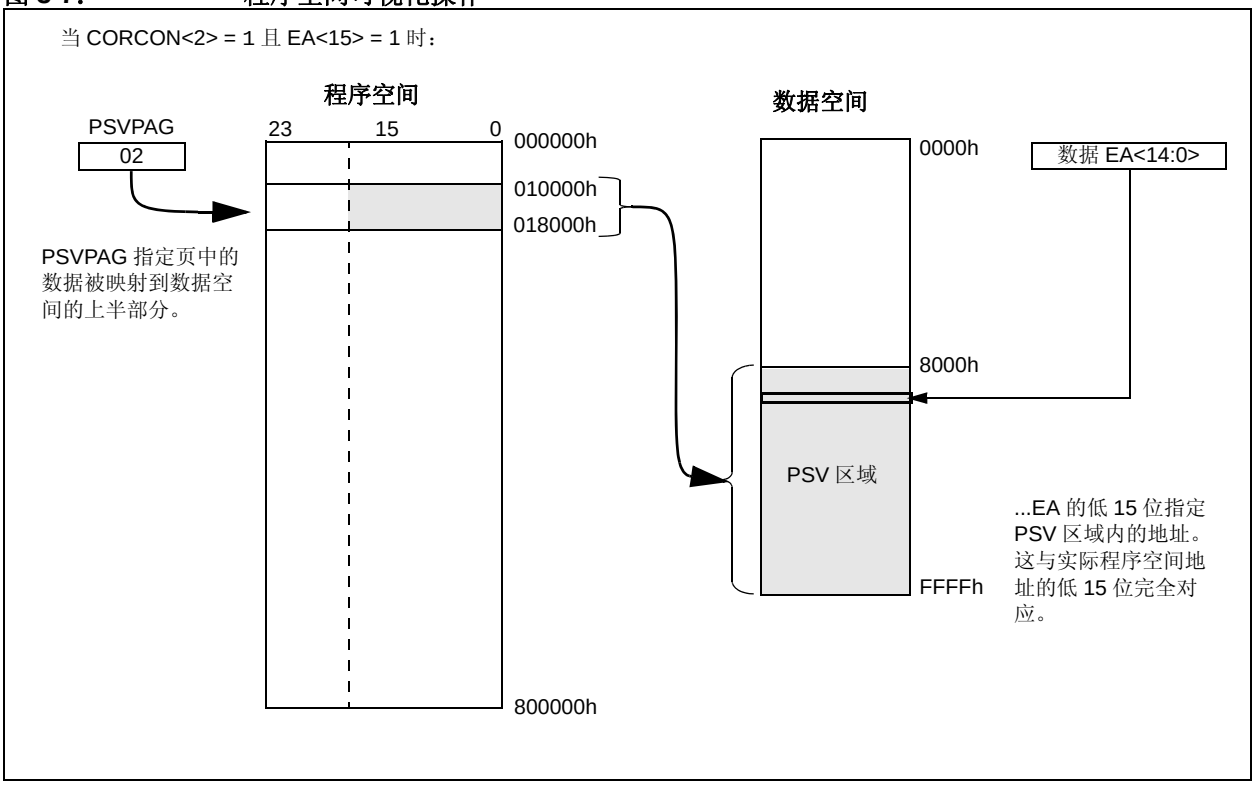
对于那些用到 PSV 并在 REPEAT 循环外执行的操作，使用MOV 和 MOV.D指令会使执行时间额外增加一个指令周期。使用其他指令会额外增加两个指令周期。

对于使用 PSV 并在 REPEAT 循环内执行的操作，在下述情况下会额外增加两个指令周期的执行时间：

- 在第一次迭代时执行指令
- 在最后一次迭代时执行指令
- 由于中断，在退出循环前执行指令
- 中断服务后，在重新进入循环时执行指令

在REPEAT循环的其他迭代中访问数据，执行时间为一个周期。

图 3-7： 程序空间可视化操作



注：

4.0 闪存程序存储器

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 4 章“程序存储器”**（DS39715A_CN）。

PIC24FJ128GA010 系列器件包含内部闪存程序存储器，用于存储和执行应用代码。在整个 VDD 范围内，闪存程序存储器在正常工作状态下都是可读写并可擦除的。

闪存存储器有四种编程方式：

1. 在线串行编程（ICSP™）
2. 运行时自编程（RTSP）
3. JTAG
4. 增强型在线串行编程（In-Circuit Serial Programming， ICSP）

ICSP 允许在最终应用电路中对 PIC24FJ128GA010 系列器件进行串行编程。只需要使用五根线就可以完成编程，它们分别是编程时钟（PGCx）、编程数据（PGDx）、电源（VDD）、地线（VSS）和主复位（$\overline{\text{MCLR}}$）信号线。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。从而可以将最新版本的固件或者定制固件烧写到单片机中。

RTSP 是通过使用 TBLRD（表读）和 TBLWT（表写）指令来实现的。使用 RTSP 用户可以一次将 64 条指令（192 字节）的数据块写入程序存储器，也可以一次擦除 512 条指令（1536 字节）。

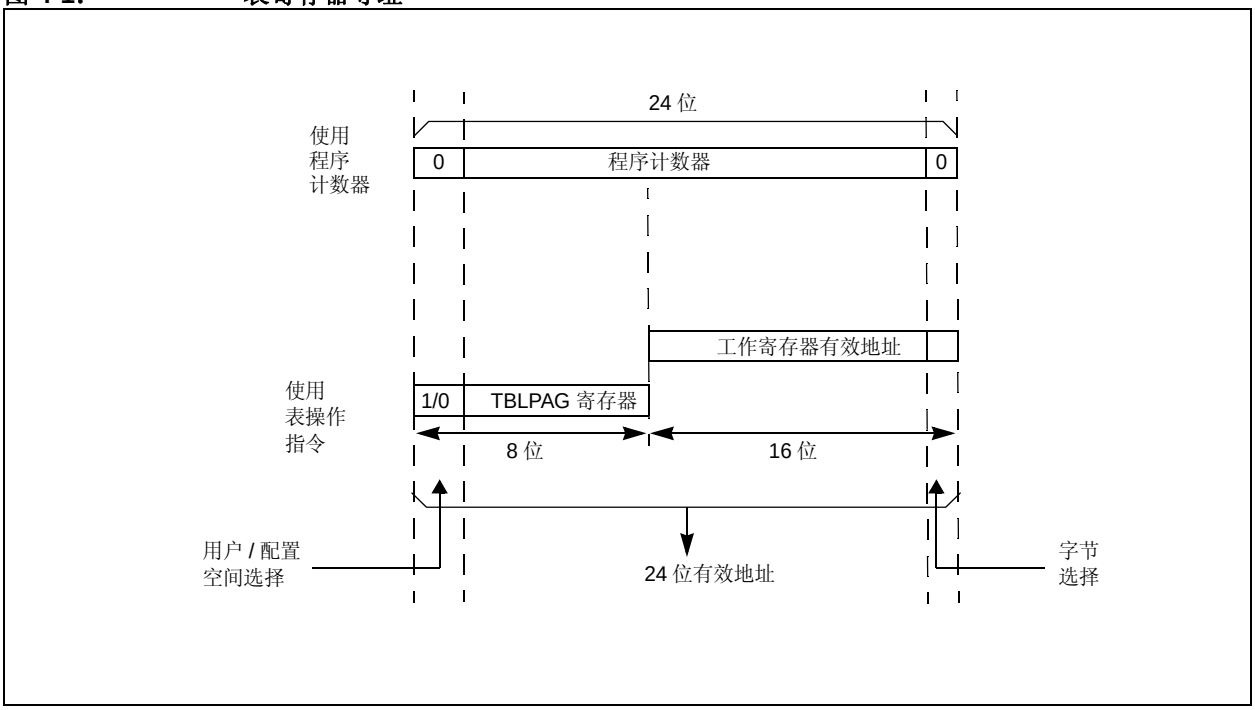
4.1 表操作指令和闪存编程

闪存存储器的编程都是用表读和表写指令实现的，与使用的方法无关。这些指令允许器件在正常工作模式下通过数据存储器直接读写程序存储空间。24 位程序存储器目标地址由 TBLPAG<7:0> 和 W 寄存器中的有效地址（EA）组成，如图 4-1 所示。

TBLRDL 和 TBLWTL 指令用于读写程序存储空间的 bit<15:0>。TBLRDL 和 TBLWTL 能以字模式或字节模式访问程序存储器。

TBLRDH 和 TBLWTH 指令用于读写程序存储空间的 bit<23:16>。TBLRDH 和 TBLWTH 同样能以字模式或字节模式访问程序存储器。

图 4-1： 表寄存器寻址



4.2 RTSP 工作原理

PIC24F 闪存程序存储器阵列以 64 条指令（即 192 字节）为一行。RTSP 允许用户一次擦除 8 行（512 条指令），一次编程一行。它还可以编程单字。

8行擦除块和单行写入块都是边界对齐的，从程序存储器起始地址开始，分别到 1536 字节边界和 192 字节边界。

用 TBLWT 指令将数据写入程序存储器时，数据并未直接写入存储器，而是存储在保持锁存器中，直到执行编程序列。

可以执行任何条数的 TBLWT 指令，且写操作都将成功执行。但是，需要 64 条 TBLWT 指令来写存储器的整行。

要确保在写操作期间没有数据被改动，应将所有未使用的地址编程为 FFFFFFh。这是因为保持锁存器复位为未知状态，因此当地址处于复位状态时，就可能改写未被重写的行上的存储单元。

RTSP 编程的基本步骤是先建立一个表指针，然后执行一系列 TBLWT 指令以将数据装入缓冲器。通过将 NVMCON 寄存器的控制位置 1 来执行编程。

可按任何顺序装入数据，且在执行写操作之前可以多次写入保持寄存器。但后续写操作将覆盖先前的所有写操作。

注：	不推荐多次写入一个存储单元而不擦除其内容。

因为只写缓冲器，因此所有表写操作都是单字写操作（2 个指令周期）。编程每行都需要一个编程周期。

4.3 JTAG 操作

PIC24FJ 系列支持 JTAG 编程和边界扫描。边界扫描可以通过验证引脚到 PCB 的连通性改进制造工艺。编程是通过支持串行向量格式（Serial Vector Format，SVF）的工业标准 JTAG 编程器来执行的。

4.4 增强型在线串行编程

增强型在线串行编程使用片上自举程序（称为编程执行程序）管理编程过程。通过使用 SPI 数据帧格式，编程执行程序可以擦除、编程和校验程序存储器。如需了解更多有关增强型 ICSP 的信息，请参见器件编程规范。

4.5 控制寄存器

读写程序闪存存储器共使用两个 SFR：NVMCON 和 NVMKEY。

NVMCON 寄存器（寄存器 4-1）控制要擦除的块和要编程的存储器类型，以及编程周期的开始。

NVMKEY 是一个只写寄存器，用于写保护。在启动编程或擦除操作前，用户必须连续写 55h 和 AAh 到 NVMKEY 寄存器。更多详细信息请参见**第 4.6 节 “编程操作”**。

4.6 编程操作

在 RTSP 模式下，对内部闪存进行编程或擦除需要完整的编程过程。在编程或擦除操作期间，处理器将停止（等待），直到操作完成。将 WR 位（NVMCON<15>）置 1 启动操作，当操作完成时 WR 位会自动清零。

配置字值储存在程序存储器的最后两个单元中。对程序存储器的最后一页执行页擦出操作，会将配置字值清零并使能代码保护。因此，应避免对程序存储器的最后一页执行页擦除操作。

寄存器 4-1：　　NVMCON：闪存存储器控制寄存器

R/SO-0(1)	R/W-0(1)	R/W-0(1)	U-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0(1)	U-0	U-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)
—	ERASE	—	—	NVMOP3(2)	NVMOP2(2)	NVMOP1(2)	NVMOP0(2)
bit 7							bit 0

图注：	SO = 只可置 1 的位		
R = 可读位	W = 可写位	SO = 只可置 1 的位	U = 未实现位
-n = 复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15　**WR：**写控制位
1 = 启动闪存存储器编程或擦除操作。该操作是自定时的，一旦操作完成该位即由硬件清零。
0 = 编程或擦除操作完成，并处于停止状态

bit 14　**WREN：**写使能位
1 = 使能闪存编程 / 擦除操作
0 = 禁止闪存编程 / 擦除操作

bit 13　**WRERR：**写序列错误标志位
1 = 尝试执行错误的编程或擦除序列或执行终止 （在 WR 位置 1 时该位被自动置 1）
0 = 编程或擦除操作正常完成

bit 12-7　**未实现：**读为 0

bit 6　**ERASE：**擦除 / 编程使能位
1 = 在下一条 WR 命令执行 NVMOP3:NVMOP0 指定的擦除操作
0 = 在下一条 WR 命令执行 NVMOP3:NVMOP0 指定的编程操作

bit 5-4　**未实现：**读为 0

bit 3-0　**NVMOP3:NVMOP0：**NVM 操作选择位 (2)
1111 = 存储块擦除操作 （ERASE = 1）或无操作 （ERASE = 0） (3)
0011 = 存储器字编程操作 （ERASE = 0）或无操作 （ERASE = 1）
0010 = 存储页擦除操作 （ERASE = 1）或无操作 （ERASE = 0）
0001 = 存储行编程操作 （ERASE = 0）或无操作 （ERASE = 1）

注　1：只能在上电复位时复位这些位。
2：NVMOP3:NVMOP0 的其他组合均未实现。
3：仅在 ICSP™ 模式下可用。参见器件编程规范。

4.6.1 闪存程序存储器的编程算法

用户能一次对一行程序闪存存储器进行编程。要实现该操作，首先需要擦除包含该行在内的一个 8 行大小的擦除块。一般操作步骤如下：

1. 读出 8 行程序存储器（512 条指令）的数据，并存储在数据 RAM 中。
2. 使用新数据更新数据 RAM 中对应的程序数据。
3. 擦除程序块 （见例 4-1）：
   a) 将 NVMOP 位 （NVMCON<3:0>）设置为 0010，配置为块擦除操作。将 ERASE 位（NVMCON<6>）和 WREN 位（NVMCON<14>）置 1。
   b) 将要擦除块的起始地址写入 TBLPAG 和 W 寄存器。
   c) 将 55h 写入 NVMKEY。
   d) 将 AAh 写入 NVMKEY。
   e) 将 WR 位 （NVMCON<15>）置 1。启动擦除周期，CPU 停止，等待擦除周期完成。当擦除完成时， WR 位自动清零。
4. 将数据 RAM 中的前 64 条指令写入程序存储器缓冲器 （见例 4-2）。
5. 将程序块写入闪存存储器：
   a) 将 NVMOP 位设置为 0001，配置为行编程操作。将 ERASE 位清零，将 WREN 位置 1。
   b) 将 55h 写入 NVMKEY。
   c) 将 AAh 写入 NVMKEY。
   d) 将 WR 位置 1，启动编程周期，CPU 停止等待写周期完成。当闪存存储器写操作完成时， WR 位自动清零。
6. 将 TBLPAG 值加 1，使用数据 RAM 中下一个 64 条指令块重复步骤 4 和 5，直到所有 512 条指令被写回闪存存储器。

为防止意外操作，必须向 NVMKEY 写入启动序列从而允许执行擦除或编程操作。在执行了编程命令后，用户必须等待编程完成。紧跟编程启动序列后面的两条指令必须为 NOP，如例 4-3 所示。

例 4-1： 擦除程序存储器块

```
; Set up NVMCON for block erase operation
        MOV     #0x4042, W0                 ;
        MOV     W0, NVMCON                  ; Initialize NVMCON
; Init pointer to row to be ERASED
        MOV     #tblpage(PROG_ADDR), W0     ;
        MOV     W0, TBLPAG                  ; Initialize PM Page Boundary SFR
        MOV     #tbloffset(PROG_ADDR), W0   ; Initialize in-page EA[15:0] pointer
        TBLWTL  W0, [W0]                    ; Set base address of erase block
        DISI    #5                          ; Block all interrupts with priority <7
                                            ; for next 5 instructions
        MOV     #0x55, W0
        MOV     W0, NVMKEY                  ; Write the 55 key
        MOV     #0xAA, W1                   ;
        MOV     W1, NVMKEY                  ; Write the AA key
        BSET    NVMCON, #WR                 ; Start the erase sequence
        NOP                                 ; Insert two NOPs after the erase
        NOP                                 ; command is asserted
```

例 4-2：　　装载写缓冲器

```
; Set up NVMCON for row programming operations
       MOV       #0x4001, W0                   ;
       MOV       W0, NVMCON                    ; Initialize NVMCON
; Set up a pointer to the first program memory location to be written
; program memory selected, and writes enabled
       MOV       #0x0000, W0                   ;
       MOV       W0, TBLPAG                    ; Initialize PM Page Boundary SFR
       MOV       #0x6000, W0                   ; An example program memory address
; Perform the TBLWT instructions to write the latches
; 0th_program_word
       MOV       #LOW_WORD_0, W2               ;
       MOV       #HIGH_BYTE_0, W3              ;
       TBLWTL    W2, [W0]                      ; Write PM low word into program latch
       TBLWTH    W3, [W0++]                    ; Write PM high byte into program latch
; 1st_program_word
       MOV       #LOW_WORD_1, W2               ;
       MOV       #HIGH_BYTE_1, W3              ;
       TBLWTL    W2, [W0]                      ; Write PM low word into program latch
       TBLWTH    W3, [W0++]                    ; Write PM high byte into program latch
;  2nd_program_word
       MOV       #LOW_WORD_2, W2               ;
       MOV       #HIGH_BYTE_2, W3              ;
       TBLWTL    W2, [W0]                      ; Write PM low word into program latch
       TBLWTH    W3, [W0++]                    ; Write PM high byte into program latch
       •
       •
       •
; 63rd_program_word
       MOV       #LOW_WORD_31, W2              ;
       MOV       #HIGH_BYTE_31, W3             ;
       TBLWTL    W2, [W0]                      ; Write PM low word into program latch
       TBLWTH    W3, [W0]                      ; Write PM high byte into program latch
```

例 4-3：　　启动编程序列

```
       DISI    #5                      ; Block all interrupts with priority <7
                                       ; for next 5 instructions
       MOV     #0x55, W0
       MOV     W0, NVMKEY              ; Write the 55 key
       MOV     #0xAA, W1               ;
       MOV     W1, NVMKEY              ; Write the AA key
       BSET    NVMCON, #WR             ; Start the program/erase sequence
       BTSC    NVMCON, #15             ; and wait for it to be
       BRA     $-2                     ; completed
```

4.6.2 编程闪存程序存储器的一个字

若已擦除了一个闪存单元，则可用表写指令对此单元进行编程以将一个指令字（24 位）写入写锁存器。将闪存地址的 8 个最高字节装入 TBLPAG 寄存器。TBLWTL 和 TBLWTH 指令将所需数据写入写锁存器，并指定要写入的程序存储器地址的低 16 位。要将 NVMCON 寄存器配置为字写操作，应将NVMOP位（NVMCON<3:0>）设置为 0011。通过执行解锁序列并将 WR 位置 1 来执行此写操作。

例 4-4： 编程闪存程序存储器的一个字

```
; Setup a pointer to data Program Memory
    MOV     #tblpage(PROG_ADDR), W0       ;
    MOV     W0, TBLPAG                    ;Initialize PM Page Boundary SFR
    MOV     #tbloffset(PROG_ADDR), W0     ;Initialize a register with program memory address

    MOV     #LOW_WORD_N, W2               ;
    MOV     #HIGH_BYTE_N, W3              ;
    TBLWTL  W2, [W0]                      ; Write PM low word into program latch
    TBLWTH  W3, [W0++]                    ; Write PM high byte into program latch

; Setup NVMCON for programming one word to data Program Memory
    MOV     #0x4003, W0                   ;
    MOV     W0, NVMCON                    ; Set NVMOP bits to 0011

    DISI    #5                            ; Disable interrupts while the KEY sequence is
                                            written
    MOV     #0x55, W0                     ; Write the key sequence
    MOV     W0, NVMKEY
    MOV     #0xAA, W0
    MOV     W0, NVMKEY
    BSET    NVMCON, #WR                   ; Start the write cycle
```

5.0 复位

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第7章“复位”**（DS39712A_CN）。

复位模块包含了所有复位资源，可控制器件的主复位信号 SYSRST。下面列出了器件的复位源：

- POR：上电复位
- MCLR：引脚复位
- SWR：RESET 指令
- WDT：看门狗定时器复位
- BOR：欠压复位
- CM：配置字不匹配复位
- TRAPR：陷阱冲突复位
- IOPUWR：非法操作码复位
- UWR：未初始化 W 寄存器复位

图 5-1 所示为复位模块的简化框图。

任何激活的复位源都会激活 SYSRST 信号。许多与 CPU 和外设相关的寄存器会被强制为一个已知的复位状态。大多数寄存器不受复位的影响；在 POR 时它们的状态未知，而在其他复位时它们的状态不变。

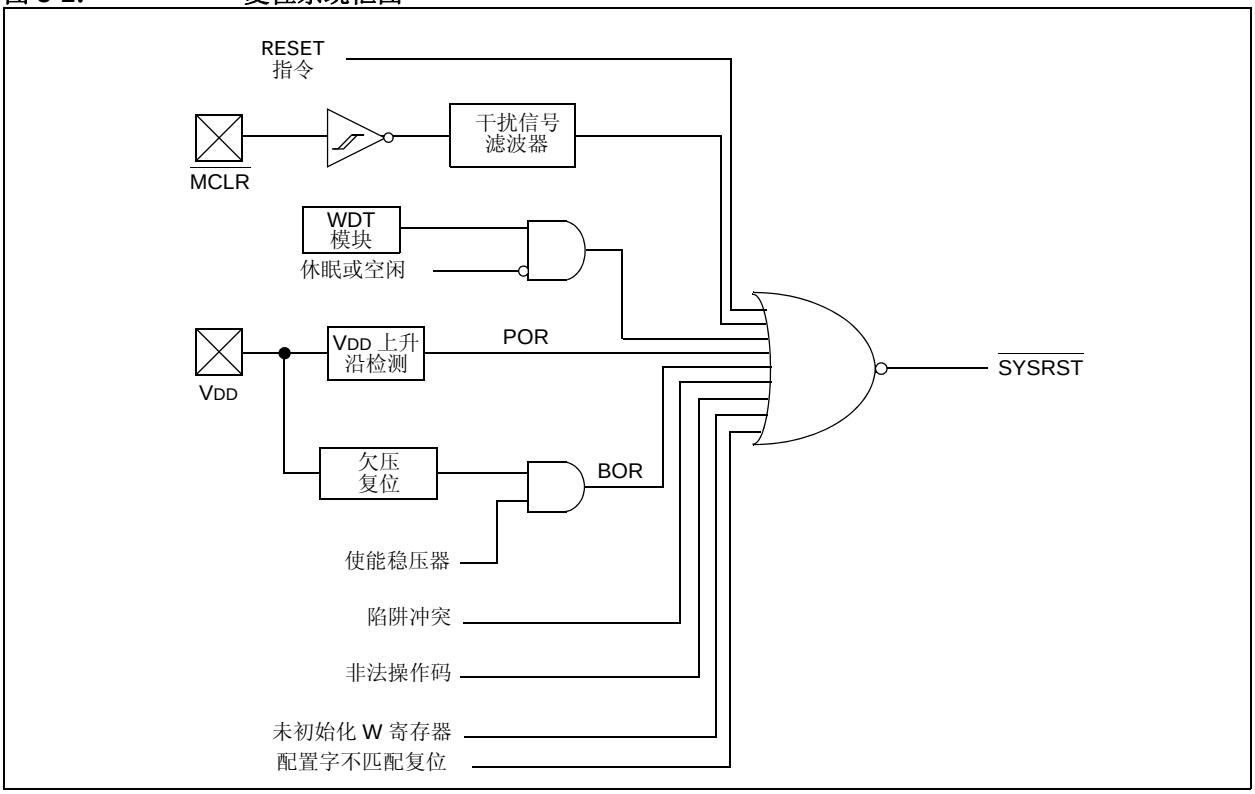
注： 有关寄存器复位状态的信息，请参见本手册中的特定外设或 CPU 章节。

任何的器件复位都会将RCON寄存器中相应的状态位置 1 来指示复位类型（见寄存器 5-1）。上电复位将清零除 POR 和 BOR 位（RCON<1:0>）外的所有位，而 POR 和 BOR 位会被置 1。用户可在代码执行过程中的任何时间置 1 或清零任何位。RCON 位仅作为状态位。用软件将特定的状态位置 1 不会导致器件发生复位。

RCON寄存器也有一些与看门狗定时器和器件节能状态相关的位。本手册的其他章节中将讨论这些位的功能。

注： RCON 寄存器中的状态位应该在被读取后清零，这样下一次器件复位后的 RCON 寄存器的值才有意义。

图 5-1： 复位系统框图



寄存器 5-1：　　RCON：复位控制寄存器 (1)

R/W-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
TRAPR	IOPUWR	—	—	—	—	CM	VREGS
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
EXTR	SWR	SWDTEN(2)	WDTO	SLEEP	IDLE	BOR	POR
bit 7							bit 0

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

bit 15　**TRAPR：**陷阱复位标志位
1 = 发生了陷阱冲突复位
0 = 未发生陷阱冲突复位

bit 14　**IOPUWR：**非法操作码或访问未初始化 W 寄存器复位标志位
1 = 检测到非法操作码、非法寻址模式或未初始化的 W 寄存器 （用作地址指针）而导致的复位
0 = 未发生由非法操作码或未初始化的 W 寄存器而导致的复位

bit 13-10　**未实现：**读为 0

bit 9　**CM：**配置字不匹配复位标志位
1 = 发生了配置字不匹配复位
0 = 未发生配置字不匹配复位

bit 8　**VREGS：**稳压器待机使能位
1 = 稳压器在器件休眠时保持工作
0 = 稳压器在器件休眠时待机

bit 7　**EXTR：**外部复位 （$\overline{\text{MCLR}}$）引脚位
1 = 发生主清零 （引脚）复位
0 = 未发生主清零 （引脚）复位

bit 6　**SWR：**软件复位 （指令）标志位 (2)
1 = 已执行 RESET 指令
0 = 未执行 RESET 指令

bit 5　**SWDTEN：**WDT 软件使能 / 禁止位
1 = 使能 WDT
0 = 禁止 WDT

bit 4　**WDTO：**看门狗定时器超时溢出标志位
1 = WDT 发生超时溢出
0 = WDT 未发生超时溢出

bit 3　**SLEEP：**从休眠模式唤醒标志位
1 = 器件处于休眠模式
0 = 器件未处于休眠模式

bit 2　**IDLE：**从空闲模式唤醒标志位
1 = 器件处于空闲模式
0 = 器件未处于空闲模式

bit 1　**BOR：**欠压复位标志位
1 = 发生了欠压复位。注意 BOR 在上电复位后也被置 1。
0 = 未发生欠压复位

bit 0　**POR：**上电复位标志位
1 = 发生了上电复位
0 = 未发生上电复位

注　1：　所有复位状态位都可以用软件置 1 或清零。用软件将任何一位置 1 不会导致器件复位。
2：　如果 FWDTEN 配置位为 1 （未编程），不管 SWDTEN 位的设置如何， WDT 总是使能的。

表 5-1：　　复位标志位操作

标志位	置 1 事件	清零事件
TRAPR （RCON<15>）	陷阱冲突事件	POR
IOPUWR （RCON<14>）	非法操作码或访问未初始化的 W 寄存器	POR
EXTR （RCON<7>）	$\overline{\text{MCLR}}$ 复位	POR
SWR （RCON<6>）	RESET 指令	POR
WDTO （RCON<4>）	WDT 超时溢出	PWRSAV 指令和 POR
SLEEP （RCON<3>）	PWRSAV #SLEEP 指令	POR
IDLE （RCON<2>）	PWRSAV #IDLE 指令	POR
BOR （RCON<1>）	POR 和 BOR	—
POR （RCON<0>）	POR	—

注：　所有复位标志位可由用户软件置 1 或清零。

5.1　复位时时钟源的选择

使能时钟切换时，器件复位时的系统时钟源如表 5-2 中所示。如果禁止时钟切换，则总是根据振荡器配置位选择系统时钟源。更多详细信息请参见**第 7.0 节 “振荡器配置”**。

表 5-2：　　振荡器选择与复位类型的关系（使能时钟切换）

复位类型	时钟源的确定
POR	振荡器配置位（FNOSC2:FNOSC0）
BOR	
$\overline{\text{MCLR}}$	COSC 控制位（OSCCON<14:12>）
WDTR	
SWR	

5.2　器件复位时间

表 5-3 总结了各种器件复位所需的时间。注意在 POR 延时和 PWRT 延时结束后，系统复位信号 SYSRST 会被释放。

器件实际开始执行代码的时间还取决于系统的振荡器延时，它包括振荡器起振定时器 （OST）和 PLL 锁定时间。OST 和 PLL 锁定时间与相应的 SYSRST 延时同时发生。

FSCM 延时决定在 $\overline{\text{SYSRST}}$ 信号被释放后 FSCM 何时开始监视系统时钟源。

表 5-3：　　各种器件复位的复位延迟时间

复位类型	时钟源	$\overline{\text{SYSRST}}$ 延时	系统时钟延时	FSCM 延时	备注
POR	EC, FRC, FRCDIV, LPRC	Tpor + Tstartup + TRST	—	—	**1, 2, 3**
	ECPLL, FRCPLL	Tpor + Tstartup + TRST	TLOCK	TFSCM	**1, 2, 3, 5, 6**
	XT, HS, SOSC	Tpor + Tstartup + TRST	TOST	TFSCM	**1, 2, 3, 4, 6**
	XTPLL, HSPLL	Tpor + Tstartup + TRST	TOST + TLOCK	TFSCM	**1, 2, 3, 4, 5, 6**
BOR	EC, FRC, FRCDIV, LPRC	Tstartup + TRST	—	—	**2, 3**
	ECPLL, FRCPLL	Tstartup + TRST	TLOCK	TFSCM	**2, 3, 5, 6**
	XT, HS, SOSC	Tstartup + TRST	TOST	TFSCM	**2, 3, 4, 6**
	XTPLL, HSPLL	Tstartup + TRST	TOST + TLOCK	TFSCM	**2, 3, 4, 5, 6**
$\overline{\text{MCLR}}$	任何时钟	TRST	—	—	**3**
WDT	任何时钟	TRST	—	—	**3**
软件	任何时钟	TRST	—	—	**3**
非法操作码	任何时钟	TRST	—	—	**3**
未初始化的 W 寄存器	任何时钟	TRST	—	—	**3**
陷阱冲突	任何时钟	TRST	—	—	**3**

注　1： TPOR = 上电复位延迟 （标称值为 10 µs）。

2： 如果使能片内稳压器，则 TSTARTUP = TVREG （标称值为 10 µs）；如果禁止片内稳压器， TSTARTUP = TPWRT （标称值为 64 ms）。

3： TRST = 内部状态复位时间 （标称值为 20 µs）。

4： TOST = 振荡器起振定时器延时。 10 位计数器计满 1024 个振荡器周期后，才将振荡器时钟释放给系统使用。

5： TLOCK = PLL 锁定时间。

6： TFSCM = 故障保护时钟监视器延时 （标称值为 100 µs）。

5.2.1 POR 和长振荡器起振时间

振荡器起振电路及其相关的延时定时器与上电时发生的器件复位延时没有任何关系。某些晶振电路（尤其是低频晶振）的起振时间会相对较长。因此，在 SYSRST 被释放后，可能会发生下面的一种或多种情况：

- 振荡电路未起振。
- 振荡器起振定时器尚未超时 （如果使用了晶振）。
- PLL 未实现锁定 （如果使用了 PLL）。

在有效时钟源供系统使用前，器件不会开始执行代码。因此，在确定复位延迟时间时，还须考虑振荡器和 PLL 起振延时。

5.2.2 故障保护时钟监视器 （FSCM）和器件复位

如果使能 FSCM，则它将在 $\overline{\text{SYSRST}}$ 被释放后开始监视系统时钟源。如果此时没有可用的有效时钟源，器件将自动切换到 FRC 振荡器，用户可以在陷阱服务程序中将系统时钟源切换到所需的晶体振荡器。

5.2.2.1 晶振和 PLL 时钟源的 FSCM 延时

当系统时钟源由晶体振荡器和 / 或 PLL 提供时，在 POR 和 PWRT 延时后会自动插入一小段延迟（TFSCM）。在此延时结束前， FSCM 不会开始监视系统时钟源。FSCM 延迟时间标称值为 100 µs，为振荡器和 / 或 PLL 达到稳定提供额外的延迟时间。在大多数情况下，当禁止 PWRT 时， FSCM 延时会防止在器件复位时产生振荡器故障陷阱。

5.3 特殊功能寄存器复位状态

大多数与 PIC24F CPU 和外设有关的特殊功能寄存器（SFR）会在器件复位时复位为某个特定值。 SFR 是按外设或 CPU 功能分组的，其复位值在本手册的相关部分中有说明。

除了四个寄存器外，其他所有的 SFR 的复位值都不受复位类型的影响。复位控制寄存器 RCON 的复位值取决于器件复位的类型。振荡器控制寄存器 OSCCON 的复位值取决于复位类型，和在 FOSC 器件配置寄存器中的振荡器配置位的编程值（见表 5-2）。 RCFGCAL 和 NVMCON 寄存器的复位值仅受 POR 影响。

注：

6.0 中断控制器

注：	本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第8章“中断”**（DS39707A_CN）。

PIC24F 中断控制器模块将大量外设中断请求信号减少到一个到 PIC24F CPU 的中断请求信号。该控制器具有以下特点：

- 多达 8 个处理器异常和软件陷阱
- 7 个用户可选择的优先级别
- 具有多达 118 个矢量的中断矢量表 （Interrupt Vector Table， IVT）
- 每个中断或异常源都有唯一的矢量
- 指定的用户优先级中的固定优先级
- 用于支持调试的备用中断矢量表 （Alternate Interrupt Vector Table， AIVT）
- 固定的中断入口和返回延时

6.1 中断矢量表

中断矢量表 （IVT）如图 6-1 所示。IVT 位于程序存储器中，起始单元地址是 000004h。IVT 包含 126 个矢量，这些矢量由 8 个不可屏蔽的陷阱向量和最多 118 个中断源组成。一般来说，每个中断源都有自己的中断矢量。每个中断矢量都包含一个 24 位宽的地址。每个中断矢量单元的存放值是其对应的中断服务程序 （ISR）的起始地址。

中断矢量是按自然优先级来划分的；也就是说每个中断矢量的优先级与其在表中的位置有关。如果其他方面都相同，更低的地址具有更高的优先级。例如，与矢量 0 相关联的中断比其他中断具有更高的优先级。

PIC24FJ128GA010 系列器件实现了不可屏蔽的陷阱和唯一的中断矢量。表 6-1 和表 6-2 对此做了总结。

6.1.1 备用中断矢量表

如图 6-1 所示，备用中断矢量表 （AIVT）位于 IVT 之后。ALTIVT 控制位（INTCON2<15>）可以控制对 AIVT 的访问。如果 ALTIVT 位置 1，所有中断和异常处理将使用备用矢量而不是默认的矢量。备用矢量与默认矢量的结构相同。

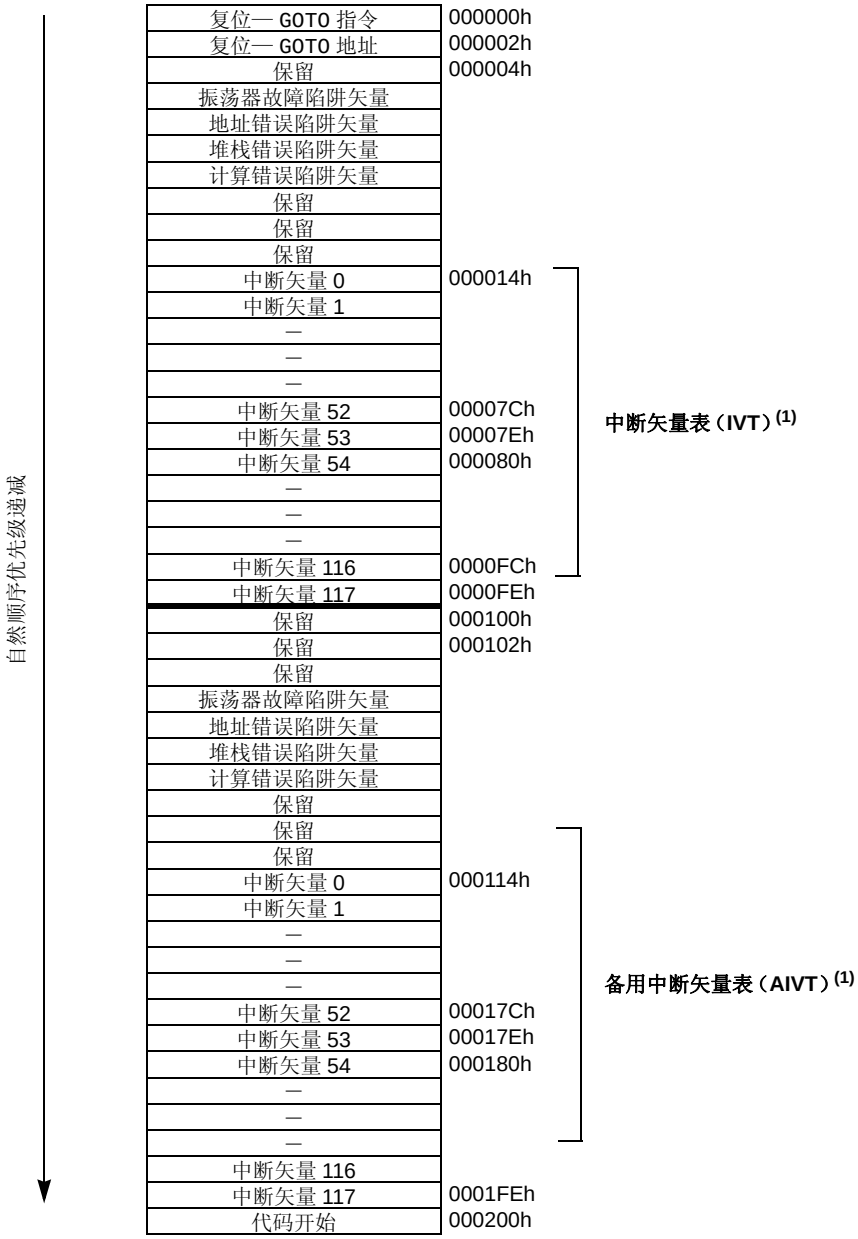
AIVT支持仿真和调试功能，它提供了一种不需要将中断矢量再编程就可以在应用和支持环境之间切换的方法。此特性也支持运行时在不同应用之间切换以便评估各种软件算法。如果不需要 AIVT，应该用 IVT 中使用的地址编程 AIVT。

6.2 复位顺序

由于复位过程中不使用中断控制器，所以器件复位并不是真的异常情况。作为对复位的响应， PIC24F 器件清零它的寄存器，同时 PC 被强制为零。然后处理器开始在地址为000000h的单元处执行程序。用户在复位地址上写入 GOTO 指令会使程序执行重新定位到相应的启动程序。

注：	应该使用包含 RESET 指令的默认中断处理程序的入口地址编程 IVT 和 AIVT 中的所有未执行或未使用的矢量单元。

图 6-1：　　PIC24F 中断矢量表



注　1： 关于中断矢量列表请参见表 6-2。

表 6-1：　　陷阱矢量详细说明

矢量号	IVT 地址	AIVT 地址	陷阱源
0	000004h	000104h	保留
1	000006h	000106h	振荡器故障
2	000008h	000108h	地址错误
3	00000Ah	00010Ah	堆栈错误
4	00000Ch	00010Ch	计算错误
5	00000Eh	00010Eh	保留
6	000010h	000110h	保留
7	000012h	0001172h	保留

表 6-2：　　实现的中断矢量

中断源	矢量号	IVT 地址	AIVT 地址	中断位地址		
				标志	允许	优先级
已完成 ADC1 转换	13	00002Eh	00012Eh	IFS0<13>	IEC0<13>	IPC3<6:4>
比较器事件	18	000038h	000138h	IFS1<2>	IEC1<2>	IPC4<10:8>
CRC 发生器	67	00009Ah	00019Ah	IFS4<3>	IEC4<3>	IPC16<14:12>
外部中断 0	0	000014h	000114h	IFS0<0>	IEC0<0>	IPC0<2:0>
外部中断 1	20	00003Ch	00013Ch	IFS1<4>	IEC1<4>	IPC5<2:0>
外部中断 2	29	00004Eh	00014Eh	IFS1<13>	IEC1<13>	IPC7<6:4>
外部中断 3	53	00007Eh	00017Eh	IFS3<5>	IEC3<5>	IPC13<6:4>
外部中断 4	54	000080h	000180h	IFS3<6>	IEC3<6>	IPC13<10:8>
I2C1 主控事件	17	000036h	000136h	IFS1<1>	IEC1<1>	IPC4<6:4>
I2C1 从动事件	16	000034h	000034h	IFS1<0>	IEC1<0>	IPC4<2:0>
I2C2 主控事件	50	000078h	000178h	IFS3<2>	IEC3<2>	IPC12<10:8>
I2C2 从动事件	49	000076h	000176h	IFS3<1>	IEC3<1>	IPC12<6:4>
输入捕捉 1	1	000016h	000116h	IFS0<1>	IEC0<1>	IPC0<6:4>
输入捕捉 2	5	00001Eh	00011Eh	IFS0<5>	IEC0<5>	IPC1<6:4>
输入捕捉 3	37	00005Eh	00015Eh	IFS2<5>	IEC2<5>	IPC9<6:4>
输入捕捉 4	38	000060h	000160h	IFS2<6>	IEC2<6>	IPC9<10:8>
输入捕捉 5	39	000062h	000162h	IFS2<7>	IEC2<7>	IPC9<14:12>
输入变化通知	19	00003Ah	00013Ah	IFS1<3>	IEC1<3>	IPC4<14:12>
输出比较 1	2	000018h	000118h	IFS0<2>	IEC0<2>	IPC0<10:8>
输出比较 2	6	000020h	000120h	IFS0<6>	IEC0<6>	IPC1<10:8>
输出比较 3	25	000046h	000146h	IFS1<9>	IEC1<9>	IPC6<6:4>
输出比较 4	26	000048h	000148h	IFS1<10>	IEC1<10>	IPC6<10:8>
输出比较 5	41	000066h	000166h	IFS2<9>	IEC2<9>	IPC10<6:4>
并行主控端口	45	00006Eh	00016Eh	IFS2<13>	IEC2<13>	IPC11<6:4>
实时时钟 / 日历	62	000090h	000190h	IFS3<14>	IEC3<13>	IPC15<10:8>
SPI1 错误	9	000026h	000126h	IFS0<9>	IEC0<9>	IPC2<6:4>
SPI1 事件	10	000028h	000128h	IFS0<10>	IEC0<10>	IPC2<10:8>
SPI2 错误	32	000054h	000154h	IFS2<0>	IEC0<0>	IPC8<2:0>
SPI2 事件	33	000056h	000156h	IFS2<1>	IEC2<1>	IPC8<6:4>
Timer1	3	00001Ah	00011Ah	IFS0<3>	IEC0<3>	IPC0<14:12>
Timer2	7	000022h	000122h	IFS0<7>	IEC0<7>	IPC1<14:12>
Timer3	8	000024h	000124h	IFS0<8>	IEC0<8>	IPC2<2:0>
Timer4	27	00004Ah	00014Ah	IFS1<11>	IEC1<11>	IPC6<14:12>
Timer5	28	00004Ch	00014Ch	IFS1<12>	IEC1<12>	IPC7<2:0>
UART1 错误	65	000096h	000196h	IFS4<1>	IEC4<1>	IPC16<6:4>
UART1 接收器	11	00002Ah	00012Ah	IFS0<11>	IEC0<11>	IPC2<14:12>
UART1 发送器	12	00002Ch	00012Ch	IFS0<12>	IEC0<12>	IPC3<2:0>
UART2 错误	66	000098h	000198h	IFS4<2>	IEC4<2>	IPC16<10:8>
UART2 接收器	30	000050h	000150h	IFS1<14>	IEC1<14>	IPC7<10:8>
UART2 发送器	31	000052h	000152h	IFS1<15>	IEC1<15>	IPC7<14:12>

6.3 中断控制和状态寄存器

PIC24FJ128GA010 系列器件有 28 个用于中断控制器的寄存器：

- INTCON1
- INTCON2
- IFS0 至 IFS4
- IEC0 至 IEC4
- IPC0 至 IPC14 和 IPC16

INTCON1 和 INTCON2 控制全局中断。INTCON1 包含中断嵌套禁止（NSTDIS）位，以及处理器陷阱源的控制和状态标志。 INTCON2 寄存器控制外部中断请求信号行为和备用中断矢量表的使用。

IFS 寄存器保存所有中断请求标志。每个中断源都有一个状态位，可由对应的外设或外部信号置 1 并通过软件清零。

IEC 寄存器保存所有中断允许位。这些控制位用于分别允许外设或外部信号的中断。

IPC 寄存器用于为每个中断源设置中断优先级，可为每个用户中断源分配 8 个优先级之一。

中断源按表 6-2 所列的顺序分配给 IFSx、IECx 和 IPCx 寄存器。例如，INT0（外部中断 0）表示矢量号为 0 同时自然的优先级也为0。因此，INT0IF状态位在IFS0<0>中，允许位在 IEC0<0> 中，优先级位在 IPC0（IPC0<2:0>）中。

尽管两个 CPU 控制寄存器不是中断控制硬件的特定组成部分，但它们仍包含控制中断功能的位。CPU 状态寄存器（SR）具有 IPL2:IPL0 位（SR<7:5>）。这些位可以显示当前 CPU 的中断优先级。用户可以通过写 IPL 位改变当前 CPU 的优先级。

CORCON 寄存器包含 IPL3 位，连同 IPL2:IPL0 一起表示当前 CPU 中断的优先级。 IPL3 是只读位，这样陷阱事件就不能被用户软件屏蔽。

在下页的寄存器 6-1到寄存器 6-30中说明了所有的中断寄存器。

寄存器 6-1： SR：CPU 状态寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	DC(1)
bit 15							bit 8

R/W-0(1)	R/W-0(1)	R/W-0(1)	R-0	R/W-0	R/W-0	R/W-0	R/W-0
IPL2(2,3)	IPL1(2,3)	IPL0(2,3)	RA(1)	N(1)	OV(1)	Z(1)	C(1)
bit 7							bit 0

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

bit 7-5 **IPL2:IPL0**：CPU 中断优先级状态位 (2,3)
111 = CPU 中断优先级为 7（15）。禁止用户中断。
110 = CPU 中断优先级为 6（14）
101 = CPU 中断优先级为 5（13）
100 = CPU 中断优先级为 4（12）
011 = CPU 中断优先级为 3（11）
010 = CPU 中断优先级为 2（10）
001 = CPU 中断优先级为 1（9）
000 = CPU 中断优先级为 0（8）

注 **1**：请参见寄存器 2-1 获知与中断控制功能无关的其余位的说明。

2：IPL 位与 IPL3 位（CORCON<3>）一起构成 CPU 的中断优先级。如果 IPL3 = 1，则括号中的值表示中断优先级。

3：当 NSTDIS（INTCON1<15>）= 1 时， IPL 状态位为只读的。

寄存器 6-2： CORCON：内核控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	R/C-0	R/W-0	U-0	U-0
—	—	—	—	IPL3(1)	PSV(1)	—	—
bit 7							bit 0

.

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 3 **IPL3**：CPU 中断优先级状态位 (2)
1 = CPU 中断优先级高于 7；禁止外设中断
0 = CPU 中断优先级等于或低于 7

注 **1**：请参见寄存器 2-2 获知与中断控制功能无关的其余位的说明。

2：IPL3 位与 IPL2:IPL0 位（SR<7:5>）一起构成 CPU 的中断优先级。

寄存器 6-3：　　INTCON1：中断控制寄存器 1

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
NSTDIS	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	MATHERR	ADDRERR	STKERR	OSCFAIL	—
bit 7							bit 0

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

bit 15　**NSTDIS：**中断嵌套禁止位
1 = 禁止中断嵌套
0 = 允许中断嵌套

bit 14-5　**未实现：**读为 0

bit 4　**MATHERR：**计算错误陷阱状态位
1 = 发生了溢出陷阱
0 = 未发生溢出陷阱

bit 3　**ADDRERR：**地址错误陷阱状态位
1 = 发生了地址错误陷阱
0 = 未发生地址错误陷阱

bit 2　**STKERR：**堆栈错误陷阱状态位
1 = 发生了堆栈错误陷阱
0 = 未发生堆栈错误陷阱

bit 1　**OSCFAIL：**振荡器故障陷阱状态位
1 = 发生了振荡器故障陷阱
0 = 未发生振荡器故障陷阱

bit 0　**未实现：**读为 0

.

寄存器 6-4：　　INTCON2：中断控制寄存器 2

R/W-0	R-0	U-0	U-0	U-0	U-0	U-0	U-0
ALTIVT	DISI	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT4EP	INT3EP	INT2EP	INT1EP	INT0EP
bit 7							bit 0

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

bit 15　**ALTIVT**：使能备用中断矢量表位
1 = 使用备用矢量表
0 = 使用标准 （默认）矢量表

bit 14　**DISI**：DISI 指令状态位
1 = DISI 指令有效
0 = DISI 指令无效

bit 13-5　**未实现**：读为 0

bit 4　**INT4EP**：外部中断 4 边沿检测极性选择位
1 = 负边沿产生中断
0 = 正边沿产生中断

bit 3　**INT3EP**：外部中断 3 边沿检测极性选择位
1 = 负边沿产生中断
0 = 正边沿产生中断

bit 2　**INT2EP**：外部中断 2 边沿检测极性选择位
1 = 负边沿产生中断
0 = 正边沿产生中断

bit 1　**INT1EP**：外部中断 1 边沿检测极性选择位
1 = 负边沿产生中断
0 = 正边沿产生中断

bit 0　**INT0EP**：外部中断 0 边沿检测极性选择位
1 = 负边沿产生中断
0 = 正边沿产生中断

寄存器 6-5：　　IFS0：中断标志状态寄存器 0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPF1IF	T3IF
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
T2IF	OC2IF	IC2IF	—	T1IF	OC1IF	IC1IF	INT0IF
bit 7							bit 0

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

bit 15,14 **未实现：**读为 0

bit 13 **AD1IF：**A/D 转换完成中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 12 **U1TXIF：**UART1 发送器中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 11 **U1RXIF：**UART1 接收器中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 10 **SPI1IF：**SPI1 事件中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 9 **SPF1IF：**SPI1 故障中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 8 **T3IF：**Timer3 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 7 **T2IF：**Timer2 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 6 **OC2IF：**输出比较通道 2 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 5 **IC2IF：**输入捕捉通道 2 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 4 **未实现：**读为 0

bit 3 **T1IF：**Timer1 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 2 **OC1IF：**输出比较通道 1 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 1 **IC1IF：**输入捕捉通道 1 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 0 **INT0IF：**外部中断 0 标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

寄存器 6-6：　　IFS1：中断标志状态寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
U2TXIF	U2RXIF	INT2IF	T5IF	T4IF	OC4IF	OC3IF	—
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT1IF	CNIF	CMIF	MI2C1IF	SI2C1IF
bit 7							bit 0

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

bit 15　**U2TXIF**：UART2 发送器中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 14　**U2RXIF**：UART2 接收器中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 13　**INT2IF**：外部中断 2 标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 12　**T5IF**：Timer5 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 11　**T4IF**：Timer4 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 10　**OC4IF**：输出比较通道 4 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 9　**OC3IF**：输出比较通道 3 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 8-5　**未实现**：读为 0

bit 4　**INT1IF**：外部中断 1 标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 3　**CNIF**：输入变化通知中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 2　**CMIF**：比较器中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 1　**MI2C1IF**：主控 I2C1 事件中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 0　**SI2C1IF**：从动 I2C1 事件中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

寄存器 6-7：　　IFS2：中断标志状态寄存器 2

U-0	U-0	R/W-0	U-0	U-0	U-0	R/W-0	U-0
—	—	PMPIF	—	—	—	OC5IF	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
IC5IF	IC4IF	IC3IF	—	—	—	SPI2IF	SPF2IF
bit 7							bit 0

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

bit 15-14 **未实现：**读为 0

bit 13 **PMPIF：**并行主控端口中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 12-10 **未实现：**读为 0

bit 9 **OC5IF：**输出比较通道 5 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 8 **未实现：**读为 0

bit 7 **IC5IF：**输入捕捉通道 5 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 6 **IC4IF：**输入捕捉通道 4 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 5 **IC3IF：**输入捕捉通道 3 中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 4-2 **未实现：**读为 0

bit 1 **SPI2IF：**SPI2 事件中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 0 **SPF2IF：**SPI2 故障中断标志状态位
1 = 已发生故障
0 = 未发生故障

寄存器 6-8：　　IFS3：中断标志状态寄存器 3

U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	RTCIF	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	U-0
—	INT4IF	INT3IF	—	—	MI2C2IF	SI2C2IF	—
bit 7							bit 0

.

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

bit 15　**未实现：**读为 0

bit 14　**RTCIF：**实时时钟 / 日历中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 13-7　**未实现：**读为 0

bit 6　**INT4IF：**外部中断 4 标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 5　**INT3IF：**外部中断 3 标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 4-3　**未实现：**读为 0

bit 2　**MI2C2IF：**主控 I2C2 事件中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 1　**SI2C2IF：**从动 I2C2 事件中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 0　**未实现：**读为 0

寄存器 6-9：　　IFS4：中断标志状态寄存器 4

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	—	CRCIF	U2ERIF	U1ERIF	—
bit 7							bit 0

.

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

bit 15-4　**未实现：**读为 0

bit 3　**CRCIF：**CRC 发生器中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 2　**U2ERIF：**UART2 错误中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 1　**U1ERIF：**UART1 错误中断标志状态位
1 = 已发生中断请求
0 = 未发生中断请求

bit 0　**未实现：**读为 0

寄存器 6-10：　　IEC0：中断允许控制寄存器 0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPF1IE	T3IE
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
T2IE	OC2IE	IC2IE	—	T1IE	OC1IE	IC1IE	INT0IE
bit 7							bit 0

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

bit 15-14 **未实现：**读为 0

bit 13 **AD1IE：**A/D 转换完成中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 12 **U1TXIE：**UART1 发送器中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 11 **U1RXIE：**UART1 接收器中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 10 **SPI1IE：**SPI1 传输完成中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 9 **SPF1IF：**SPI1 故障中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 8 **T3IE：**Timer3 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 7 **T2IE：**Timer2 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 6 **OC2IE：**输出比较通道 2 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 5 **IC2IE：**输入捕捉通道 2 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 4 **未实现：**读为 0

bit 3 **T1IE：**Timer1 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 2 **OC1IE：**输出比较通道 1 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 1 **IC1IE：**输入捕捉通道 1 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 0 **INT0IE：**外部中断 0 允许位
1 = 允许中断请求
0 = 禁止中断请求

寄存器 6-11： IEC1：中断允许控制寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	—
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT1IE	CNIE	CMIE	MI2C1IE	SI2C1IE
bit 7							bit 0

.

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

bit 15 **U2TXIE**：UART2 发送器中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 14 **U2RXIE**：UART2 接收器中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 13 **INT2IE**：外部中断 2 允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 12 **T5IE**：Timer5 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 11 **T4IE**：Timer4 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 10 **OC4IE**：输出比较通道 4 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 9 **OC3IE**：输出比较通道 3 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 8-5 **未实现**：读为 0

bit 4 **INT1IE**：外部中断 1 允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 3 **CNIE**：输入变化通知中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 2 **CMIE**：比较器中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 1 **MI2C1IE**：主控 I2C1 事件中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 0 **SI2C1IE**：从动 I2C1 事件中断允许位
1 = 允许中断请求
0 = 禁止中断请求

寄存器 6-12： IEC2：中断允许控制寄存器 2

U-0	U-0	R/W-0	U-0	U-0	U-0	R/W-0	U-0
—	—	PMPIE	—	—	—	OC5IE	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
IC5IE	IC4IE	IC3IE	—	—	—	SPI2IE	SPF2IE
bit 7							bit 0

.

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

bit 15-14 **未实现：**读为 0

bit 13 **PMPIE：**并行主控端口中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 12-10 **未实现：**读为 0

bit 9 **OC5IE：**输出比较通道 5 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 8 **未实现：**读为 0

bit 7 **IC5IE：**输入捕捉通道 5 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 6 **IC4IE：**输入捕捉通道 4 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 5 **IC3IE：**输入捕捉通道 3 中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 4-2 **未实现：**读为 0

bit 1 **SPI2IE：**SPI2 事件中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 0 **SPF2IE：**SPI2 故障中断允许位
1 = 允许中断请求
0 = 禁止中断请求

寄存器 6-13：　　IEC3：中断允许控制寄存器 3

U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	RTCIE	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	U-0
—	INT4IE	INT3IE	—	—	MI2C2IE	SI2C2IE	—
bit 7							bit 0

.

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

bit 15　**未实现：**读为 0

bit 14　**RTCIE：**实时时钟 / 日历中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 13-7　**未实现：**读为 0

bit 6　**INT4IE：**外部中断 4 允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 5　**INT3IE：**外部中断 3 允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 4-3　**未实现：**读为 0

bit 2　**MI2C2IE：**主控 I2C2 事件中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 1　**SI2C2IE：**从动 I2C2 事件中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 0　**未实现：**读为 0

寄存器 6-14：　　IEC4：中断允许控制寄存器 4

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	—	CRCIE	U2ERIE	U1ERIE	—
bit 7							bit 0

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

bit 15-4　**未实现：**读为 0

bit 3　**CRCIE：**CRC 发生器中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 2　**U2ERIE：**UART2 错误中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 1　**U1ERIE：**UART1 错误中断允许位
1 = 允许中断请求
0 = 禁止中断请求

bit 0　**未实现：**读为 0

寄存器 6-15： IPC0：中断优先级控制寄存器 0

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T1IP2	T1IP1	T1IP0	—	OC1IP2	OC1IP1	OC1IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC1IP2	IC1IP1	IC1IP0	—	INT0IP2	INT0IP1	INT0IP0
bit 7							bit 0

.

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

bit 15 **未实现：**读为 0

bit 14-12 **T1IP2:T1IP0：**Timer1 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 11 **未实现：**读为 0

bit 10-8 **OC1IP2:OC1IP0：**输出比较通道 1 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7 **未实现：**读为 0

bit 6-4 **1C1IP2:1C1IP0：**输入捕捉通道 1 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3 **未实现：**读为 0

bit 2-0 **INT0IP2:INT0IP0：**外部中断 0 优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

寄存器 6-16：　　IPC1：中断优先级控制寄存器 1

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T2IP2	T2IP1	T2IP0	—	OC2IP2	OC2IP1	OC2IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	IC2IP2	IC2IP1	IC2IP0	—	—	—	—
bit 7							bit 0

.

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

bit 15　**未实现：**读为 0

bit 14-12　**T2IP2:T2IP0：**Timer2 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 11　**未实现：**读为 0

bit 10-8　**OC2IP2:OC2IP0：**输出比较通道 2 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7　**未实现：**读为 0

bit 6-4　**IC2IP2:IC2IP0：**输入捕捉通道 2 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3-0　**未实现：**读为 0

寄存器 6-17： IPC2：中断优先级控制寄存器 2

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	U1RXIP2	U1RXIP1	U1RXIP0	—	SPI1IP2	SPI1IP1	SPI1IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	SPF1IP2	SPF1IP1	SPF1IP0	—	T3IP2	T3IP1	T3IP0
bit 7							bit 0

.

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

bit 15 **未实现：**读为 0

bit 14-12 **U1RXIP2:U1RXIP0：**UART1 接收器中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 11 **未实现：**读为 0

bit 10-8 **SPI1IP2:SPI1IP0：**SPI1 事件中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7 **未实现：**读为 0

bit 6-4 **SPF1IP2:SPF1IP0：**SPI1 故障中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3 **未实现：**读为 0

bit 2-0 **T3IP2:T3IP0：**Timer3 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

寄存器 6-18：　　IPC3：中断优先级控制寄存器 3

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	AD1IP2	AD1IP1	AD1IP0	—	U1TXIP2	U1TXIP1	U1TXIP0
bit 7							bit 0

.

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

bit 15-7　**未实现：**读为 0

bit 6-4　**AD1IP2:AD1IP0：**A/D 转换完成中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3　**未实现：**读为 0

bit 2-0　**U1TXIP2:U1TXIP0：**UART1 发送器中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

寄存器 6-19： IPC4：中断优先级控制寄存器 4

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	CNIP2	CNIP1	CNIP0	—	CMIP2	CMIP1	CMIP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	MI2C1IP2	MI2C1IP1	MI2C1IP0	—	SI2C1IP2	SI2C1IP1	SI2C1IP0
bit 7							bit 0

.

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

bit 15 **未实现：**读为 0

bit 14-12 **CNIP2:CNIP0：**输入变化通知中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 11 **未实现：**读为 0

bit 10-8 **CMIP2:CMIP0：**比较器中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7 **未实现：**读为 0

bit 6-4 **MI2C1IP2:MI2C1IP0：**主控 I2C1 事件中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3 **未实现：**读为 0

bit 2-0 **SI2C1IP2:SI2C1IP0：**从动 I2C1 事件中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

寄存器 6-20：　　IPC5：中断优先级控制寄存器 5

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	INT1IP2	INT1IP1	INT1IP0
bit 7							bit 0

.

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

bit 15-3　**未实现：**读为 0

bit 2-0　**INT1IP2:INT1IP0：**外部中断 1 优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

寄存器 6-21： IPC6：中断优先级控制寄存器 6

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	T4IP2	T4IP1	T4IP0	—	OC4IP2	OC4IP1	OC4IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	OC3IP2	OC3IP1	OC3IP0	—	—	—	—
bit 7							bit 0

.

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

bit 15 **未实现：**读为 0

bit 14-12 **T4IP2:T4IP0：**Timer4 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 11 **未实现：**读为 0

bit 10-8 **OC4IP2:OC4IP0：**输出比较通道 4 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7 **未实现：**读为 0

bit 6-4 **OC3IP2:OC3IP0：**输出比较通道 3 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3-0 **未实现：**读为 0

寄存器 6-22：　　IPC7：中断优先级控制寄存器 7

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	U2TXIP2	U2TXIP1	U2TXIP0	—	U2RXIP2	U2RXIP1	U2RXIP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	INT2IP2	INT2IP1	INT2IP0	—	T5IP2	T5IP1	T5IP0
bit 7							bit 0

.

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

bit 15　**未实现：**读为 0

bit 14-12　**U2TXIP2:U2TXIP0：**UART2 发送器中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 11　**未实现：**读为 0

bit 10-8　**U2RXIP2:U2RXIP0：**UART2 接收器中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7　**未实现：**读为 0

bit 6-4　**INT2IP2:INT2IP0：**外部中断 2 优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3　**未实现：**读为 0

bit 2-0　**T5IP2:T5IP0：**Timer5 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

寄存器 6-23：　　IPC8：中断优先级控制寄存器 8

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	SPI2IP2	SPI2IP1	SPI2IP0	—	SPF2IP2	SPF2IP1	SPF2IP0
bit 7							bit 0

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

bit 15-7　**未实现：**读为 0

bit 6-4　**SPI2IP2:SPI2IP0：**SPI2 事件中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3　**未实现：**读为 0

bit 2-0　**SPF2IP2:SPF2IP0：**SPI2 故障中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

寄存器 6-24：　　IPC9：中断优先级控制寄存器 9

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	IC5IP2	IC5IP1	IC5IP0	—	IC4IP2	IC4IP1	IC4IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	IC3IP2	IC3IP1	IC3IP0	—	—	—	—
bit 7							bit 0

.

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

bit 15　**未实现：**读为 0

bit 14-12　**IC5IP2:IC5IP0：**输入捕捉通道 5 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 11　**未实现：**读为 0

bit 10-8　**IC4IP2:IC4IP0：**输入捕捉通道 4 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7　**未实现：**读为 0

bit 6-4　**IC3IP2:IC3IP0：**输入捕捉通道 3 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3-0　**未实现：**读为 0

寄存器 6-25：　　IPC10：中断优先级控制寄存器 10

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	OC5IP2	OC5IP1	OC5IP0	—	—	—	—
bit 7							bit 0

.

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

bit 15-7　**未实现：**读为 0

bit 6-4　**OC5IP2:OC5IP0：**输出比较通道 5 中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3-0　**未实现：**读为 0

寄存器 6-26：　　IPC11：中断优先级控制寄存器 11

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	PMPIP2	PMPIP1	PMPIP0	—	—	—	—
bit 7							bit 0

.

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

bit 15-7　**未实现：**读为 0

bit 6-4　**PMPIP2:PMPIP0：**并行主控端口中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3-0　**未实现：**读为 0

寄存器 6-27：　　IPC12：中断优先级控制寄存器 12

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	MI2C2IP2	MI2C2IP1	MI2C2IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	SI2C2IP2	SI2C2IP1	SI2C2IP0	—	—	—	—
bit 7							bit 0

.

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

bit 15-11 **未实现：**读为 0

bit 10-8 **MI2C2IP2:MI2C2IP0：**主控 I2C2 事件中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7 **未实现：**读为 0

bit 6-4 **SI2C2IP2:SI2C2IP0：**从动 I2C2 事件中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3-0 **未实现：**读为 0

寄存器 6-28：　　IPC13：中断优先级控制寄存器 13

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	INT4IP2	INT4IP1	INT4IP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	INT3IP2	INT3IP1	INT3IP0	—	—	—	—
bit 7							bit 0

.

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

bit 15-11　**未实现：**读为 0

bit 10-8　**INT4IP2:INT4IP0：**外部中断 4 优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7　**未实现：**读为 0

bit 6-4　**INT3IP2:INT3IP0：**外部中断 3 优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3-0　**未实现：**读为 0

寄存器 6-29：　　IPC15：中断优先级控制寄存器 15

U-0	U-0	U-0	U-0	U-0	R/W-1	R/W-0	R/W-0
—	—	—	—	—	RTCIP2	RTCIP1	RTCIP0
bit 15							bit 8

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

.

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

bit 15-11 **未实现：**读为 0

bit 10-8 **RTCIP2:RTCIP0：**实时时钟 / 日历中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7-0 **未实现：**读为 0

寄存器 6-30：　　IPC16：中断优先级控制寄存器 16

U-0	R/W-1	R/W-0	R/W-0	U-0	R/W-1	R/W-0	R/W-0
—	CRCIP2	CRCIP1	CRCIP0	—	U2ERIP2	U2ERIP1	U2ERIP0
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	U1ERIP2	U1ERIP1	U1ERIP0	—	—	—	—
bit 7							bit 0

.

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

bit 15　**未实现：**读为 0

bit 14-12　**CRCIP2:CRCIP0：**CRC 发生器错误中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 11　**未实现：**读为 0

bit 10-8　**U2ERIP2:U2ERIP0：**UART2 错误中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 7　**未实现：**读为 0

bit 6-4　**U1ERIP2:U1ERIP0：**UART1 错误中断优先级位
111 = 中断优先级为 7（最高优先级中断）
•
•
•
001 = 中断优先级为 1
000 = 中断源被禁止

bit 3-0　**未实现：**读为 0

6.4 中断设置过程

6.4.1 初始化

要配置中断源：

1. 如果不需要嵌套中断，将 NSTDIS 控制位（INTCON1<15>）置 1。
2. 通过写相应的 IPCx 控制寄存器中的控制位可以为中断源分配由用户选择的优先级。优先级由特定的应用和中断源类型决定。如果不需要多个优先级，可以将所有被允许中断源的 IPCx 寄存器控制位编程为相同的非零值。

注： 在器件复位时，IPC 寄存器被初始化，且为所有用户中断源分配优先级 4。

3. 将与 IFSx 状态寄存器中外设相关的中断标志状态位清零。
4. 通过将与 IECx 控制寄存器中的中断源相关的中断允许控制位置 1 允许中断源。

6.4.2 中断服务程序

如何声明ISR以及怎样使用正确的矢量地址初始化IVT，将取决于编程语言（即 C 语言或汇编语言）和用于开发此应用程序的语言开发工具包。一般情况下，用户必须将 IFSx 寄存器中与 ISR 处理的中断源相对应的中断标志清零。否则，在退出后会立即再次进入 ISR。如果 ISR 用汇编语言编码，则必须使用 `RETFIE` 指令终止程序，以便使保存的PC值、SRL值和原先的CPU优先级出栈。

6.4.3 陷阱服务程序

陷阱服务程序（TSR）的编码方式类似于 ISR，只是必须将 INTCON1 寄存器中相应的陷阱状态标志清零，以避免重新进入 TSR。

6.4.4 禁止中断

可以使用以下步骤禁止所有用户中断：

1. 使用 `PUSH` 指令将当前 SR 值压入软件堆栈。
2. 通过将值 OEh 与 SRL 进行逻辑或操作来强制把 CPU 的优先级设置为 7。

要允许用户中断，可以使用`POP`指令恢复先前SR的值。

注意只能禁止优先级小于或等于 7 的用户中断。不能禁止陷阱源（优先级为 8-15）。

使用 `DISI` 指令可以方便地将优先级为 1-6 的中断禁止一段固定的时间。`DISI` 指令不能禁止优先级为 7 的中断源。

注：

7.0 振荡器配置

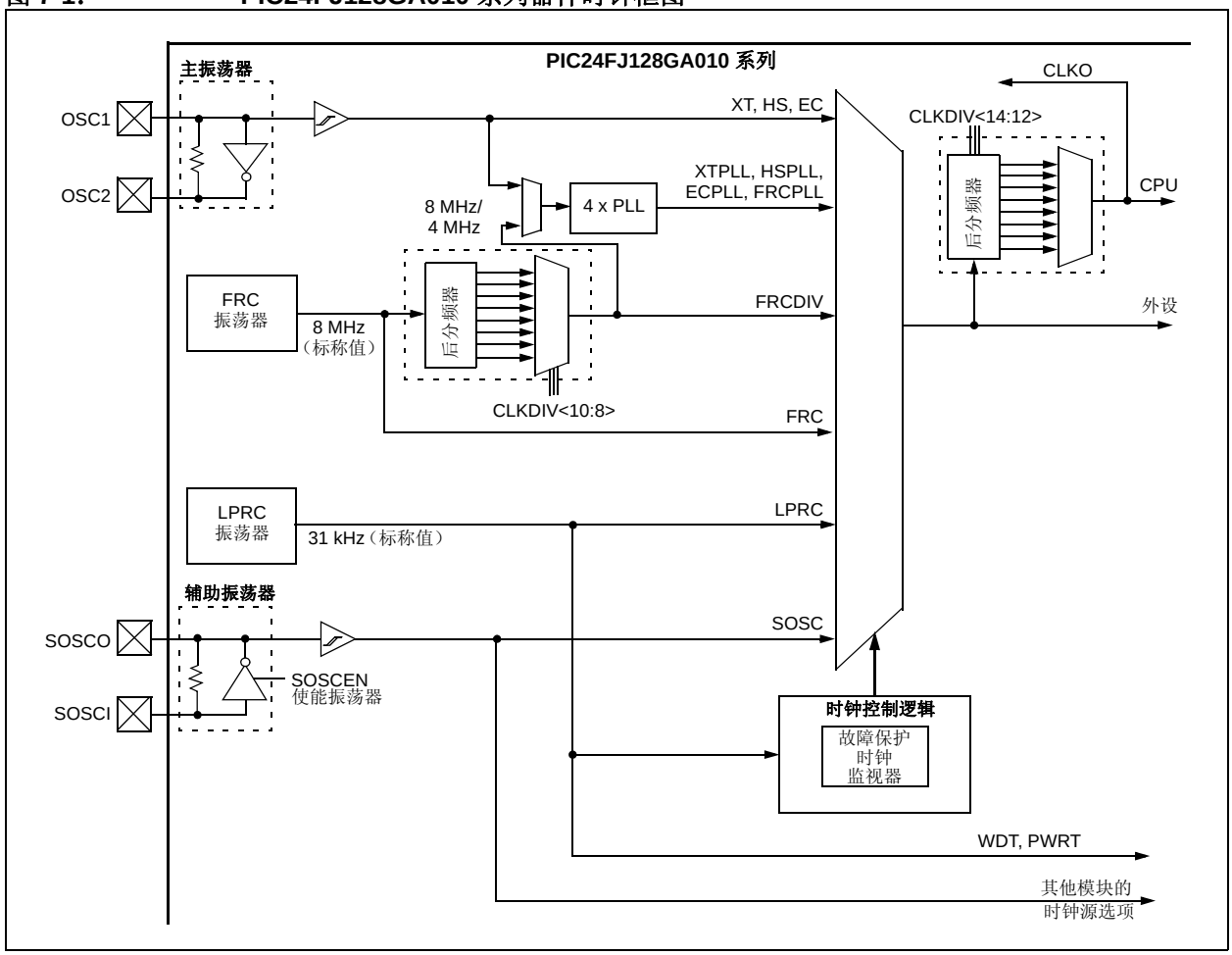
注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 6 章“振荡器”**（DS39700A_CN）。

PIC24FJ128GA010 系列器件的振荡器系统具有以下特征：

- 共有 4 个外部和内部振荡器选项可作为时钟源，提供 11 种不同的时钟模式
- 片内 4x PLL 提高了所选内部或外部振荡源的内部工作频率
- 可由软件控制在各个时钟源间进行切换
- 可由软件控制后分频器来选择 CPU 时钟速率，用以降低系统功耗
- 故障保护时钟监视器（FSCM）可检测到时钟故障，并允许应用安全恢复或关闭

图 7-1 所示为振荡器系统的简化框图。

图 7-1： PIC24FJ128GA010 系列器件时钟框图



7.1 CPU 时钟机制

系统时钟可由四个时钟源之一提供：

- OSC1 和 OSC2 引脚上的主振荡器 （POSC）
- SOSCI 和 SOSCO 引脚上的辅助振荡器 （SOSC）
- 内部快速 RC （FRC）振荡器
- 低功耗内部 RC （LPRC）振荡器

主振荡器和 FRC 源可选择使用内部 4x PLL。通过对时钟分频器进行编程可降低 FRC 时钟源的频率。所选时钟源为处理器和外设提供时钟。

处理器时钟源 2 分频后作为内部指令周期时钟 FCY。在本文档中，指令周期时钟也可由 FOSC/2 表示。在一些主振荡器工作模式下，内部指令周期时钟 FOSC/2 由 OSC2 I/O 引脚提供。

7.2 振荡器配置

使用配置位选择器件上电复位时所使用的振荡源 （和工作模式） 。振荡器配置位位于程序存储器的配置寄存器中（更多细节请参见**第 23.1 节“配置位”**）。主振荡器配置位 POSCMD1:POSCMD0 （配置字 2<1:0>）和初始振荡器选择配置位 FNOSC2:FNOSC0 （配置字 2<10:8>）用于选择上电复位时所使用的振荡器。默认（未编程）选择带后分频器 （FRCDIV）的 FRC 主振荡器。编程这些位可以选择使用辅助振荡器或某个内部振荡器。

用户可通过配置位选择如表 7-1 所示的各种时钟模式。

7.2.1 时钟切换模式配置位

还可以使用 FCKSM 配置位（配置字 2<7:6>）对器件时钟切换和故障保护时钟监视器 （FSCM）进行配置。只有当 FCKSM1 被编程 （为 0）时，才使能时钟切换。只有当 FCKSM1:FCKSM0 被编程 （为 00）时，才使能 FSCM。

表 7-1： 选择不同时钟的配置位值

振荡器模式	振荡源	POSCMD1: POSCMD0	FNOSC2: FNOSC0	注
带后分频器 （FRCDIV）的快速 RC 振荡器	内部	11	111	**1, 2**
（保留）	内部	xx	110	**1**
低功耗 RC 振荡器 （LPRC）	内部	11	101	**1**
辅助 （Timer1）振荡器 （SOSC）	辅助振荡器	11	100	**1**
带有 PLL 模块 （HSPLL）的主振荡器 （HS）	主振荡器	10	011	
带有 PLL 模块 （XTPLL）的主振荡器 （EC）	主振荡器	01	011	
带有 PLL 模块 （ECPLL）的主振荡器 （XT）	主振荡器	00	011	
主振荡器 （HS）	主振荡器	10	010	
主振荡器 （XT）	主振荡器	01	010	
主振荡器 （EC）	主振荡器	00	010	
带有 PLL 模块 （FRCPLL）的快速 RC 振荡器	内部	11	001	**1**
快速 RC 振荡器 （FRC）	内部	11	000	**1**

注 1： OSCIOFNC 配置位决定 OSC2 引脚功能。

2： 对于未编程 （已擦除）器件，这是默认的振荡器模式。

7.3 控制寄存器

由三个特殊功能寄存器控制振荡器的工作方式：

- OSCCON
- CLKDIV
- OSCTUN

OSCCON 寄存器 （寄存器 7-1）是振荡器的主控制寄存器。使用它可控制时钟源切换，并允许监视时钟源。

时钟分频寄存器 （寄存器 7-2）控制与打盹模式相关的功能以及 FRC 振荡器的后分频器。

FRC 振荡器调节寄存器（寄存器 7-3）允许用户在大约 ±12% 的范围内对 FRC 振荡器进行微调。每递增或递减一位会以某固定值改变 FRC 振荡器出厂时的校准频率。

寄存器 7-1： OSCCON：振荡器控制寄存器

U-0	R-0	R-0	R-0	U-0	R/W-x(1)	R/W-x(1)	R/W-x(1)
—	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0
bit 15							bit 8

R/SO-0	U-0	R-0(2)	U-0	R/CO-0	U-0	R/W-0	R/W-0
CLKLOCK	—	LOCK	—	CF	—	SOSCEN	OSWEN
bit 7							bit 0

图注：	CO = 只清零位	SO = 只置 1 位	
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15 **未实现：** 读为 0

bit 14-12 **COSC2:COSC0：** 当前振荡器选择位
111 = 带有后分频器的快速 RC 振荡器 （FRCDIV）
110 = 保留
101 = 低功耗 RC 振荡器 （LPRC）
100 = 辅助振荡器 （SOSC）
011 = 带有 PLL 模块的主振荡器 （XTPLL、 HSPLL 和 ECPLL）
010 = 主振荡器 （XT、 HS 和 EC）
001 = 带有后分频器和 PLL 模块的快速 RC 振荡器 （FRCPLL）
000 = 快速 RC 振荡器 （FRC）

bit 11 **未实现：** 读为 0

bit 10-8 **NOSC2:NOSC0：** 新振荡器选择位
111 = 带有后分频器的快速 RC 振荡器 （FRCDIV）
110 = 保留
101 = 低功耗 RC 振荡器 （LPRC）
100 = 辅助振荡器 （SOSC）
011 = 带有 PLL 模块的主振荡器 （XTPLL、 HSPLL 和 ECPLL）
010 = 主振荡器 （XT、 HS 和 EC）
001 = 带有后分频器和 PLL 模块的快速 RC 振荡器 （FRCPLL）
000 = 快速 RC 振荡器 （FRC）

bit 7 **CLKLOCK：** 时钟选择锁定使能位
如果使能 FSCM （FCKSM1 = 1）：
1 = 锁定时钟和 PLL 的选项
0 = 没有锁定时钟和 PLL 的选项，可通过设置 OSWEN 来更改时钟和 PLL
如果禁止 FSCM （FCKSM1 = 0）：
时钟和 PLL 选项始终不锁定，可通过设置 OSWEN 对其进行更改。

bit 6 **未实现：** 读为 0

注 1： 这些位的复位值由 FNOSC 配置位决定。
2： 在有效时钟切换期间或选择了非 PLL 时钟模式时复位到 0。

寄存器 7-1：　　OSCCON：振荡器控制寄存器 （续）

bit 5　**LOCK**：PLL 锁定状态位
1 = PLL 模块处于锁定态，或者 PLL 模块启动定时器已结束定时
0 = PLL 模块未锁定， PLL 启动定时器正在运行或 PLL 已被禁止

bit 4　**未实现**：读为 0

bit 3　**CF**：时钟故障检测位
1 = FSCM 检测到时钟故障
0 = 没有检测到时钟故障

bit 2　**未实现**：读为 0

bit 1　**SOSCEN**：32 kHz 辅助振荡器 （SOSC）使能位
1 = 使能辅助振荡器
0 = 禁止辅助振荡器

bit 0　**OSWEN**：振荡器切换使能位
1 = 将振荡器切换到由 NOSC2:NOSC0 位指定的时钟源
0 = 振荡器切换完成

注　1：这些位的复位值由 FNOSC 配置位决定。
2：在有效时钟切换期间或选择了非 PLL 时钟模式时复位到 0。

寄存器 7-2：　　CLKDIV：时钟分频器寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
ROI	DOZE2	DOZE1	DOZE0	DOZEN(1)	RCDIV2	RCDIV1	RCDIV0
bit 15							bit 8

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

图注：	CO = 只可清零位	SO = 只可置 1 位	
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15　**ROI：**中断恢复位
1 = 中断将 DOZEN 位清零， CPU 外设时钟分频比复位为 1:1
0 = 中断对 DOZEN 位没有影响

bit 14-12　**DOZE2:DOZE0：**CPU 外设时钟分频比选择位
111 = 1:128
110 = 1:64
101 = 1:32
100 = 1:16
011 = 1:8
010 = 1:4
001 = 1:2
000 = 1:1

bit 11　**DOZEN：**DOZE 使能位 (1)
1 = 由 DOZE2:DOZE0 位指定 CPU 外设时钟分频比
0 = CPU 外设时钟分频比设为 1:1

bit 10-8　**RCDIV2:RCDIV0：**FRC 后分频比选择位
111 = 31.25 kHz （256 分频）
110 = 125 kHz （64 分频）
101 = 250 kHz （32 分频）
100 = 500 kHz （16 分频）
011 = 1 MHz （8 分频）
010 = 2 MHz （4 分频）
001 = 4 MHz （2 分频）
000 = 8 MHz （1 分频）

bit 7-0　**未实现：**读为 0

注　1：　当 ROI 置 1 时会发生中断，该位自动清零。

寄存器 7-3：　　OSCTUN：FRC 振荡器调节寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
bit 7							bit 0

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

bit 15-6　**未实现：**读为 0

bit 5-0　**TUN5:TUN0：**FRC 振荡器调节位

011111 = 最大频率偏差
011110 =
•
•
•
000001 =
000000 = 中心频率，振荡器在出厂校准过的频率下运行
111111 =
•
•
•
100001 =
100000 = 最大频率偏差

7.4 时钟切换工作原理

在软件控制下，应用可以随时在四种时钟源 （POSC、SOSC、FRC 和 LPRC）之间切换，几乎不受任何限制。为了限制这种灵活切换可能带来的副作用， PIC24F 器件在切换过程中加入了保护锁定功能。

注：	主振荡器模式有 3 种不同子模式（XT、HS 和 EC），模式选择由 POSCMD 配置位决定。应用可在软件控制下在主振荡器和其他振荡器之间切换，但如果不重新编程器件，就无法在不同的主振荡器子模式间切换。

7.4.1 使能时钟切换

要使能时钟切换，闪存配置字 2 寄存器中的 FCKSM1 配置位必须被编程为 0。（详细信息请参见**第 23.1 节“配置位”**。）如果 FCKSM1 配置位未编程 （为 1），则时钟切换功能和故障保护时钟监视器功能将被禁止。这是默认设置。

当禁止时钟切换时，NOSC 控制位（OSCCON<10:8>）将不能控制时钟的选择。但COSC位（OSCCON<14:12>）会反映 FNOSC 配置位所选的是哪一个时钟源。

当禁止时钟切换时， OSWEN 控制位（OSCCON<0>）将不起作用。该位始终为 0。

7.4.2 振荡器切换步骤

时钟切换至少需要以下基本步骤：

1. 如需要，读 COSC 位（OSCCON<14:12>），判断当前的振荡源。
2. 执行解锁序列以允许写入 OSCCON 寄存器的高字节。
3. 将适当值写入NOSC控制位（OSCCON<10:8>）来选择新的振荡源。
4. 执行解锁序列以允许写入 OSCCON 寄存器的低字节。
5. 将 OSWEN 位置 1，启动振荡器切换。

完成以上基本步骤后，系统时钟硬件将自动作出以下响应：

1. 时钟切换硬件将 COSC 状态位和 NOSC 控制位的新值作比较。如果相同，则不需要切换时钟。在这种情况下， OSWEN 位自动清零，中止时钟切换。
2. 如果启动了有效时钟切换，LOCK （OSCCON<5>）和 CF （OSCCON<3>）状态位被清零。
3. 如果新振荡器处于非运行状态，硬件会将它启动。如果启动的是晶体振荡器，启动之前硬件将等待 OST 超时；如果新的振荡源使用 PLL，硬件将等待直到检测到 PLL 已锁定（LOCK = 1）。
4. 在新时钟源工作 10 个时钟周期之后，硬件开始执行时钟切换。
5. 硬件清零 OSWEN 位表示时钟转换成功。此外，NOSC 位的值被传送给 COSC 状态位。
6. 此时关闭旧时钟源，但 LPRC （如果 WDT 或 FSCM 被使能）或 SOSC （如果 SOSCEN 保持置 1）时钟源除外。

注	**1：** 在整个时钟切换过程中，处理器将继续执行代码。对时序要求高的代码不应在此期间执行。
	2： 不允许在带 PLL 的主振荡器模式和 FRCPLL 模式之间直接切换。在上述情况下，应用必须先切换到作为过渡的 FRC 模式。

推荐的时钟切换代码序列包括：

1. 在 OSSCCON 寄存器解锁或写序列过程中要禁止中断。
2. 通过使用两个背靠背指令将 78h 和 9Ah 写入 OSCCON<15:8>，解除 OSCCON 高字节锁定。
3. 在解锁序列后立即用指令将新振荡源的代码写入 NOSC 控制位。
4. 通过使用两个背靠背指令将 46h 和 57h 写入OSCCON<7:0>，解除 OSCCON 低字节锁定。
5. 在解锁操作后立即用指令将 OSWEN 位置 1。
6. 继续执行对时钟要求不高的代码（可选）。
7. 执行一段适当的软件延时（周期数），允许选定的振荡器和 / 或 PLL 启动并稳定下来。
8. 检测 OSWEN 是否为 0。如果是，则表示切换成功。如果 OSWEN 位仍为置 1 状态，检查 LOCK 位以确定故障原因。

例 7-1 给出了 OSCCON 寄存器解锁和时钟切换启动操作的核心代码。

例 7-1：　　时钟切换基本代码序列

```
;Place the new oscillator selection in W0
;OSCCONH (high byte) Unlock Sequence
MOV         #OSCCONH, w1
MOV         #0x78, w2
MOV         #0x9A, w3
MOV.b       w2, [w1]
MOV.b       w3, [w1]
;Set new oscillator selection
MOV.b       WREG, OSCCONH
;OSCCONL (low byte) unlock sequence
MOV         #OSCCONL, w1
MOV         #0x46, w2
MOV         #0x57, w3
MOV.b       w2, [w1]
MOV.b       w3, [w1]
;Start oscillator switch operation
BSET OSCCON, #0
```

8.0 节能特性

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 10 章“节能特性”**（DS39698A_CN）。

PIC24FJ128GA010 系列提供了管理功耗的功能，该功能是通过有效地管理 CPU 和外设的时钟源来实现的。一般而言，较低的时钟频率和减少时钟源驱动电路的数目会使功耗降低。所有 PIC24F 器件用以下四种方法管理功耗：

- 时钟频率
- 基于指令的休眠和空闲模式
- 软件控制的打盹模式
- 用软件有选择地进行外设控制

可以组合使用这些方法从而在保证应用关键指标（如对于时序要求高的通信）的情况下有选择地调节应用的功耗。

8.1 时钟频率和时钟切换

PIC24F 器件提供的时钟频率范围较大，用户可根据应用需要进行选择。如果未锁定系统时钟配置，用户只需更改 NOSC 位即可选择低功耗或高精度振荡器。在工作期间更改系统时钟的过程以及相应的限制，将在**第 7.0 节“振荡器配置”**中进行更详细地讨论。

8.2 基于指令的节能模式

PIC24F 器件有两种特殊的节能模式，通过执行特殊的 PWRSAV 指令可以进入这两种模式。休眠模式下时钟停止运行并停止所有代码执行；空闲模式下 CPU 停止工作，代码停止执行，但是允许外设模块继续工作。例 8-1 中所示为 PWRSAV 指令的汇编语法。

在中断产生（已停止）、WDT 超时或器件复位时，器件会退出休眠模式和空闲模式。器件退出这两种模式的过程称为“唤醒”。

注： SLEEP_MODE 和 IDLE_MODE 是在所选器件的汇编器包含文件中定义的常数。

8.2.1 休眠模式

休眠模式具有下列特征：

- 系统时钟源关闭。如果使用了片上振荡器，也要关闭它。
- 如果没有 I/O 引脚拉电流，则器件电流消耗将降至最低。
- 由于系统时钟源被禁止，所以故障保护时钟监视器在休眠模式下不工作。
- 如果 WDT 被使能， LPRC 时钟将继续在休眠模式下运行。
- WDT 如果被使能，则在进入休眠模式之前自动清零。
- 有些器件功能或外设可能在休眠模式下继续工作，包括 I/O 端口上的输入变化通知功能或使用外部时钟输入的外设等。任何需要使用系统时钟源来工作的外设在休眠模式下将被禁止。

当发生以下任何事件时，器件将从休眠模式唤醒：

- 产生被单独允许的中断
- 任何形式的器件复位
- WDT 超时

唤醒时，处理器将使用在进入休眠模式时的有效时钟源重新启动。

例 8-1： PWRSAV 指令语法

```
PWRSAV    #SLEEP_MODE       ; Put the device into SLEEP mode
PWRSAV    #IDLE_MODE        ; Put the device into IDLE mode
```

8.2.2 空闲模式

空闲模式具有下列特征：

- CPU 将停止执行指令。
- WDT 自动清零。
- 系统时钟源保持活动状态。默认情况下，所有外设模块将继续使用系统时钟源正常工作，也可以有选择地禁止它们（见**第 8.4 节 “有选择的外设模块控制”**）。
- 如果 WDT 或 FSCM 被使能， LPRC 也将保持活动状态。

当发生以下任何事件时，器件将从空闲模式唤醒：

- 产生被单独允许的中断。
- 任何器件复位。
- WDT 超时。

从空闲模式唤醒时，为 CPU 重新提供时钟，且立即从 PWRSAV 指令之后的下一条指令或 ISR 中的第一条指令开始执行。

8.2.3 在节能指令执行期间的中断

在 PWRSAV 指令执行期间发生的中断都将延迟到进入休眠或空闲模式后才产生，并导致器件从休眠或空闲模式中唤醒。

8.3 打盹模式

通常，更改时钟速度和进入某种节能模式是降低功耗的首选策略。然而，有些情况下不可行。例如，某些应用可能必须保持不间断的同步通信，即使在它不执行任何其他操作时也不例外。降低系统时钟速度可能会带来通信错误，而使用节能模式可能会终止通信。

打盹模式是另一种简单有效的节能方法，它可以在器件仍然执行代码的情况下降低功耗。在此模式下，继续以相同的时钟源和相同的速度驱动系统时钟。外设模块时钟速度保持不变，但 CPU 时钟的速度降低了。保持这两个时域同步，可以保持外设存取 SFR 的能力，同时 CPU 以较慢的速率执行代码。

通过将 DOZEN 位（CLKDIV<11>）置 1 使能打盹模式。外设与内核的时钟速度之比是由 DOZE2:DOZE0 位（CLKDIV<14:12>）决定的。有八种可能的配置，从 1:1 到 1:256，其中 1:1 是默认值。

在事件驱动的应用中，使用打盹模式有选择地降低功耗是可行的。这样就可以实现不间断地运行对时序要求高的功能 （如同步通信），而 CPU 保持空闲，等待事件调用中断程序。通过将 ROI 位 （CLKDIV<15>）置 1，可以使器件在产生中断时自动返回到全速 CPU 操作模式。默认情况下，中断事件对打盹模式操作没有影响。

8.4 有选择的外设模块控制

空闲模式和打盹模式使用户可以通过减慢或停止 CPU 时钟大幅度地降低功耗。即使如此，外设模块仍然使用时钟源并因此耗能。可能在有些情况下应用需要这些模式无法提供的功能，比如将绝大部分能源分配给CPU处理工作，而只为外设提供最低的功耗。

PIC24F 器件允许有选择地禁止外设模块，从而降低或消除它们的功耗，以此满足上述需求。可以用两个控制位执行此操作：

- 外设使能位 （通常称为 “XXXEN”），位于模块的主控 SFR 中。
- 外设模块禁止 （PMD）位 （通常称为 “XXXMD”），位于某个 PMD 控制寄存器中。

这两个位在使能或禁止其相关模块方面具有相似的功能。将某个模块的 PMD 位置 1 会禁止该模块的所有时钟源，从而将其功耗降至绝对最小值。在此状态下，与此外设相关的控制和状态寄存器也会被禁止，所以无法写这些寄存器且读取值无效。很多外设模块都有对应的 PMD 位。

通过清零某个模块的 XXXEN 位将会禁止其功能，但是仍然允许对其寄存器进行读写操作。这样做同样会降低功耗，但是没有将 PMD 位置 1 所降低的程度高。大多数外设模块都有一个使能位，但捕捉、比较和 RTCC 模块除外。

要节省更多的能耗，也可在器件进入空闲模式时有选择地禁止外设模块。使用通用名格式为 “XXXIDL”的控制位可以执行此操作。默认情况下，可以在空闲模式下工作的所有模块都可以执行此操作。禁止处于空闲状态下的功能模块可以进一步降低空闲模式下的功耗，从而可将节省下来的能源用于关键的应用部分。

9.0 I/O 端口

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 12 章“带外设引脚（PPS）的 I/O 端口”**（DS39711A_CN）。

所有器件引脚（除 VDD、VSS、MCLR 和 OSC1/CLKI 以外）均由外设和并行 I/O 端口共用。所有 I/O 输入端口都为施密特触发输入，提高了抗噪声能力。

9.1 并行 I/O（PIO）端口

与某个外设共用一个引脚的并行 I/O 端口总是服务于该外设。外设的输出缓冲器数据和控制信号提供给一对多路开关。这对多路开关用于选择 I/O 引脚的输出数据和控制信号是用于外设还是相应的端口。该逻辑电路同时会阻止“环回进入（loopthrough）”，即一个端口的数字输出会驱动共用相同引脚的外设输入。图 9-1 中显示出端口与外设是如何复用的，以及对应的 I/O 引脚。

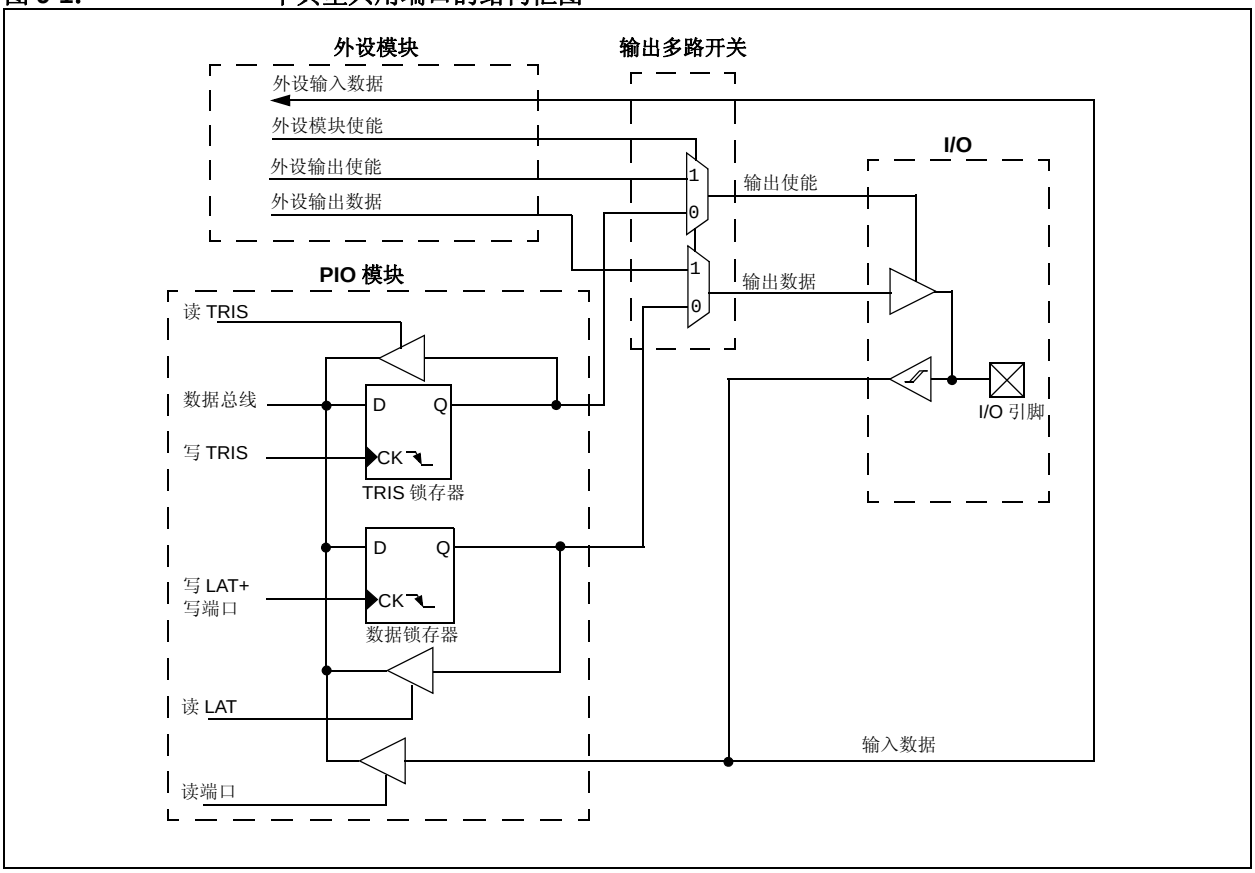
当使能某外设并驱动与其相对应的引脚时，将禁止此引脚的通用输出功能。可以读该 I/O 引脚，但并行端口引脚的输出驱动器将被禁止。若使能某外设但没有驱动引脚时，则该引脚可由一个端口驱动。

所有端口引脚都有三个寄存器，这些寄存器与端口引脚作为数字 I/O 时的工作直接相关。数据方向寄存器（TRISx）决定引脚是输入引脚还是输出引脚。如果数据方向位为 1，则为输入引脚。复位以后，所有端口引脚被定义为输入引脚。可以直接读写锁存器（LATx）。但读取端口（PORTx）时，读的是端口引脚的值；而写入端口引脚时，写入的是相应的锁存器。

对某个外设器件而言，无效的位及其相关的数据和控制寄存器都将被禁止。这意味着相应的 LATx 和 TRISx 寄存器，以及该端口引脚将读为 0。

当一个只用作输入的引脚与另一个外设或功能复用时，由于没有其他相互竞争的输出源，它将不再是一个专用端口。INT4 引脚就是这样一个例子。

图 9-1：　　一个典型共用端口的结构框图



9.1.1 漏极开路配置

除 PORT、LAT 和 TRIS 寄存器用于数据控制外，每个端口引脚也可被单独地配置为数字输出或漏极开路输出端口。这是由与每个端口相对应的漏极开路控制寄存器（ODCx）控制的，将其中的任何位置 1 即可将相应的引脚配置为漏极开路输出端口。

这种开漏特性允许通过使用外部上拉电阻在数字引脚上产生高于 VDD（如 5V）的输出电平。允许的最大开漏电压与最大 VIH 规范相同。

9.2 配置模拟端口引脚

AD1PCFG 和 TRIS 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引脚，则必须将相应的 TRIS 位置 1（输入）。如果将 TRIS 位清零（输出），则该引脚的数字输出电平（VOH 或 VOL）将被转换。

读取 PORT 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不对模拟输入信号进行转换。对任何定义为数字输入的引脚（包括 ANx 引脚）施加模拟电平可能导致输入缓冲器的电流消耗超出规范值。

9.2.1 I/O 端口写 / 读时序

改变端口方向或对端口执行写操作，与对同一端口执行读操作之间需要间隔一个指令周期。通常在两者之间插入一条 NOP 指令。

9.3 输入状态变化通知

I/O 端口的输入状态变化通知功能允许 PIC24FJ128GA010 系列器件在选定输入引脚的状态变化时，向处理器发出中断请求。当禁止时钟时，该特性可在休眠模式下检测到输入状态改变事件。取决于器件的引脚数，最多可以允许（使能）22 个外部信号（CN0 到 CN21）产生该类型中断请求。

有四个与 CN 模块相关的控制寄存器。CNEN1 和 CNEN2 寄存器包含每个 CN 输入引脚的中断允许位。将其中任一位置 1 将允许相应引脚的 CN 中断。

每个 CN 引脚都有一个与之相连的弱上拉电路。弱上拉电路充当连接到该引脚的电流源，当连接了按钮或键盘设备时，不再需要使用外部电阻。使用包含每个 CN 引脚控制位的 CNPU1 和 CNPU2 寄存器可分别使能各个上拉电路。将任一控制位置 1 均可使能相应引脚的弱上拉功能。

当选择内部上拉电路时，引脚使用 VDDCORE 作为上拉源电压。当使能内部上拉电路时，确保没有外部上拉源，因为电压差会形成电流通路。

注：	只要该引脚被配置为数字输出引脚，状态变化通知引脚上的弱上拉电路将始终被禁止。

例 9-1： 端口写 / 读示例

```
MOV   0xFF00, W0          ; Configure PORTB<15:8> as inputs
MOV   W0, TRISBB          ; and PORTB<7:0> as outputs
NOP                       ; Delay 1 cycle
btss  PORTB, #13          ; Next Instruction
```

10.0 TIMER1

注：	本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 14 章“定时器”**（DS39704A_CN）。

Timer1 模块是一个 16 位定时器，可以用作实时时钟的时间计数器，或用作独立运行的定时 / 计数器。

Timer1 能工作在以下 3 种模式下：

- 16 位定时器
- 16 位同步计数器
- 16 位异步计数器

Timer1 还支持以下功能：

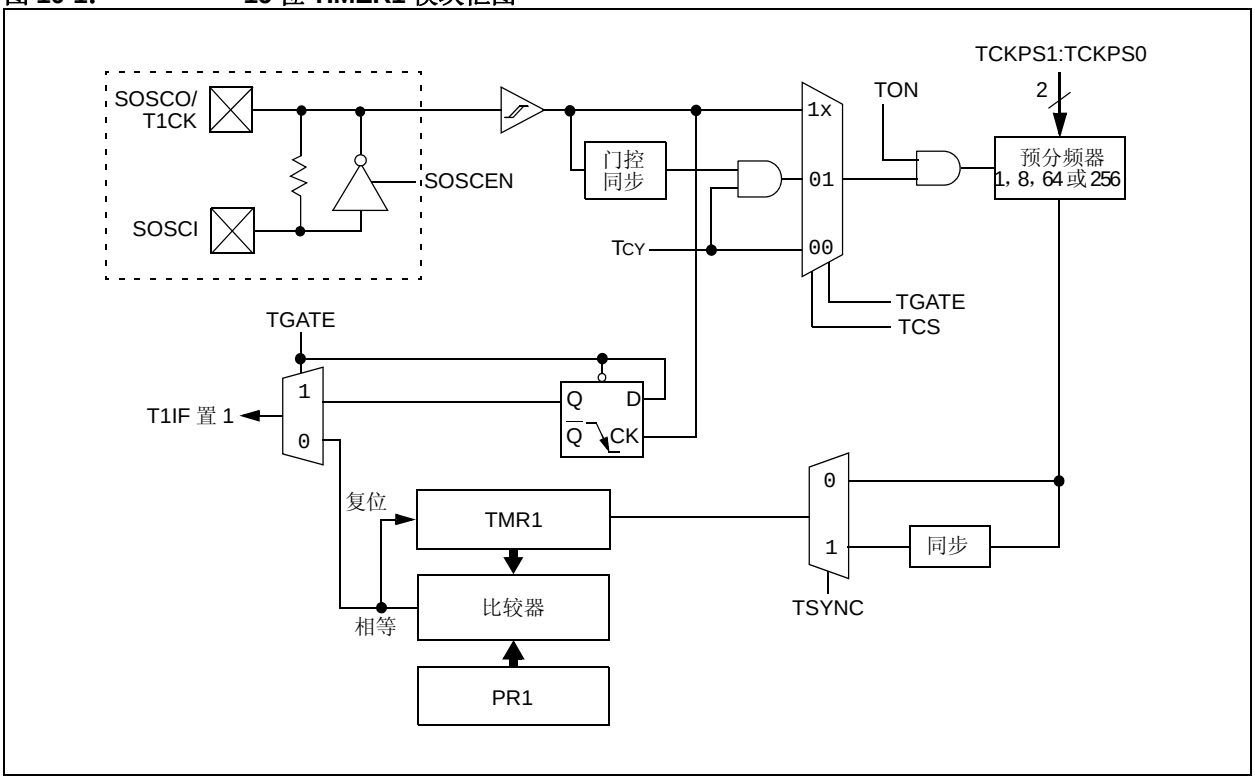
- 作为定时器门控工作
- 进行预分频比设置
- CPU 空闲和休眠模式下作为定时器工作
- 当 16 位周期寄存器匹配或出现外部门控信号下降沿时产生中断

图 10-1 给出了 16 位定时器模块的框图。

要配置 Timer1：

1. 将 TON 位置 1 （= 1）。
2. 使用 TCKPS1:TCKPS0 位选择定时器的预分频比。
3. 使用 TCS 和 TGATE 位选择时钟和门控模式。
4. 将 TSYNC 位置 1 或清零分别配置为同步或异步操作。
5. 将定时器的周期值装载到 PR1 寄存器。
6. 如果需要中断，将中断允许位 T1IE 置 1。使用优先级 T1IP2:T1IP0 位设置中断优先级。

图 10-1： 16 位 TIMER1 模块框图



寄存器 10-1：　　T1CON：TIMER1 控制寄存器

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	U-0
—	TGATE	TCKPS1	TCKPS0	—	TSYNC	TCS	—
bit 7							bit 0

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

bit 15　**TON：**Timer1 启动位
1 = 启动 16 位 Timer1
0 = 停止 16 位 Timer1

bit 14　**未实现：**读为 0

bit 13　**TSIDL：**空闲模式停止位
1 = 当器件进入空闲模式时，模块停止工作
0 = 模块在空闲模式继续工作

bit 12-7　**未实现：**读为 0

bit 6　**TGATE：**Timer1 门控时间累加使能位
当 TCS = 1 时：
此位与模块操作无关。
当 TCS = 0 时：
1 = 使能门控时间累加
0 = 禁止门控时间累加

bit 5-4　**TCKPS1:TCKPS0：**Timer1 输入时钟预分频比选择位
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

bit 3　**未实现：**读为 0

bit 2　**TSYNC：**Timer1 外部时钟输入同步选择位
当 TCS = 1 时：
1 = 同步外部时钟输入
0 = 不同步外部时钟输入
当 TCS = 0 时：
此位与模块操作无关。

bit 1　**TCS：**Timer1 时钟源选择位
1 = 来自 T1CK 引脚的外部时钟 （上升沿触发定时器递增 / 递减）
0 = 内部时钟 （FOSC/2）

bit 0　**未实现：**读为 0

11.0 TIMER2/3 和 TIMER4/5

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 14 章“定时器”**（DS39704A_CN）。

Timer2/3 和 Timer4/5 模块各为一个 32 位定时器，它们也可以被配置为 4 个独立的、工作模式可选的 16 位定时器。

作为 32 位定时器，Timer2/3 和 Timer4/5 可以工作在以下三种模式：

- 两个独立的 16 位定时器 （Timer2 和 Timer3），可实现所有 16 位工作模式
- 一个 32 位定时器
- 一个 32 位同步计数器

它们还支持以下功能：

- 定时器门控操作
- 可选的预分频比设置
- 空闲和休眠模式下作为定时器工作
- 在 32 位周期寄存器匹配时产生中断
- ADC 事件触发器 （仅 Timer4/5）

在作为 16 位定时器时，所有 4 个定时器都可以单独用作同步定时器或计数器。它们也可以提供上面列出的功能，但 ADC 事件触发器除外，它只能在 Timer5 上实现。 工作模式和是否使能由在 T2CON、T3CON、T4CON 和 T5CON 寄存器中的位的设置决定。寄存器 11-1 给出了 T2CON 和 T4CON 的一般形式；寄存器 11-2 给出了 T3CON 和 T5CON 的一般形式。

当工作在32位定时器/计数器模式时，Timer2和Timer4作为两个低字，而 Timer3 和 Timer5 则作为两个高字。

注： 当定时器工作在 32 位模式时，将不使用T3CON和T5CON的控制位。只有T2CON和 T4CON 的控制位用于设置和控制。Timer2 和 Timer4 提供时钟和门控输入，但由 Timer3 或 Timer5 的中断触发产生模块中断。

要将 Timer2/3 或 Timer4/5 配置为工作在 32 位模式下：

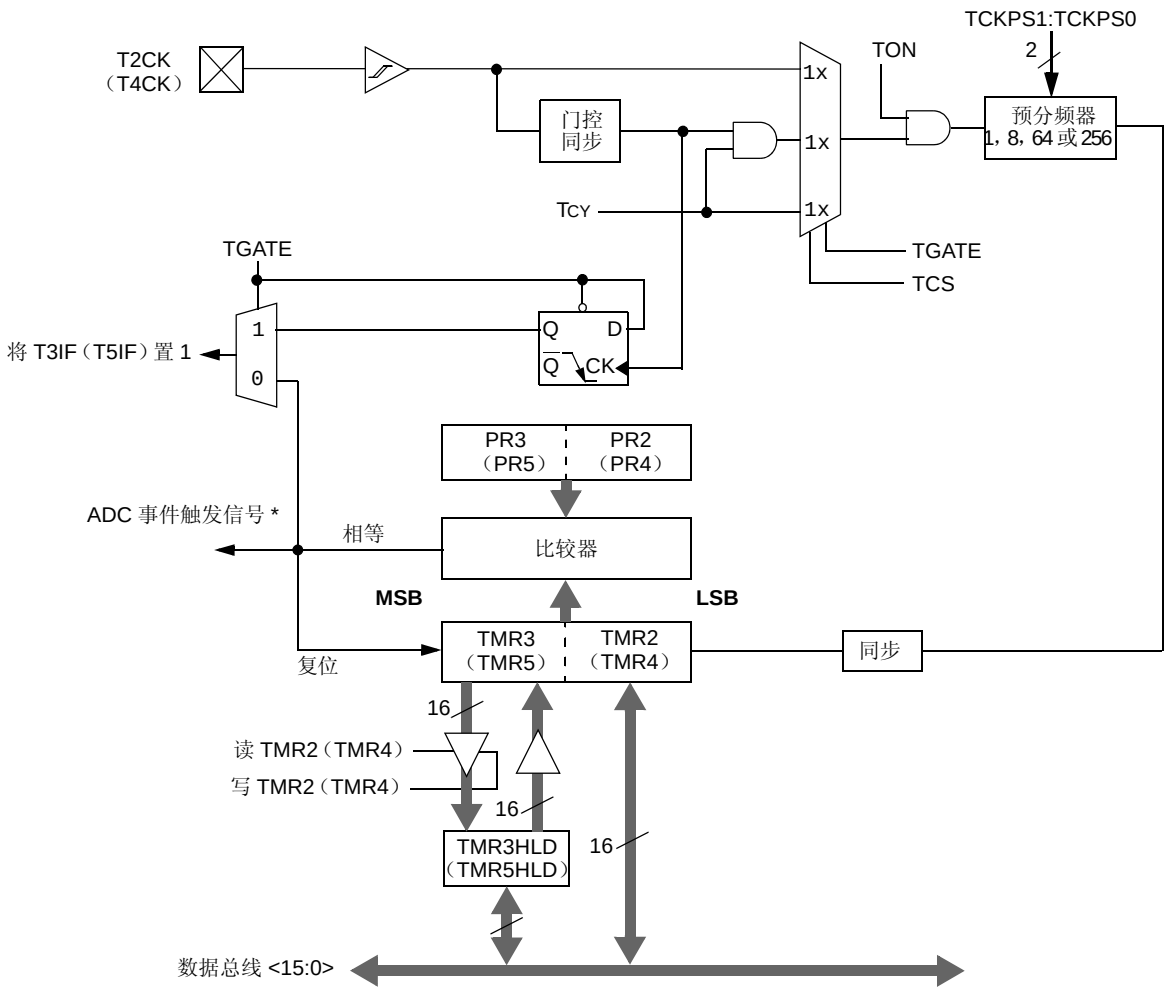
1. 将 T32 位置 1（T2CON<3> 或 T4CON<3> = 1）。
2. 使用 TCKPS1:TCKPS0 位为 Timer2 或 Timer4 选择预分频比。
3. 使用 TCS 和 TGATE 位设置时钟和门控模式。
4. 装载定时器周期值。 高字被装载到 PR3 （或 PR5），而低字则被装载到 PR2 （或 PR4）。
5. 如果需要中断，将中断允许位 T3IE 或 T5IE 置 1，并使用优先级位 T3IP2:T3IP0 或 T5IP2:T5IP0 设置中断优先级。请注意虽然模块由 Timer2 或 Timer4控制，但其中断却表现为Timer3或Timer5中断。
6. 将 TON 位置 1 （= 1）。

定时器的值可以被随时存储到 TMR3:TMR2 （或 TMR5:TMR4）寄存器对中。TMR3（TMR5）始终存放高字，而 TMR2 （TMR4）始终存放低字。

要将定时器配置为工作在独立的 16 位模式下：

1. 将与该定时器对应的模式选择位 T32 清零（对于 Timer2 和 Timer3 来说是 T2CON<3> ；对于 Timer4 和 Timer5 来说是 T4CON<3>）。
2. 使用 TCKPS1:TCKPS0 位选择定时器的预分频比。
3. 使用 TCS 和 TGATE 位设置时钟和门控模式。
4. 将定时器的周期值装载到 PRx 寄存器。
5. 如果需要中断，将中断允许位 TxIE 置 1，并使用优先级位 TxIP2:TxIP0 设置中断优先级。
6. 将 TON 位置 1 （TxCON<15> = 1）。

图 11-1：　　TIMER2/3 和 TIMER4/5 （32 位）框图



注： 必须将 32 位定时器配置位 T32 置 1 才能使定时器工作在 32 位定时器 / 计数器模式下。所有控制位都在 T2CON 寄存器和 T4CON 寄存器中。

* 只有 Timer4/5 可以产生 ADC 事件触发信号。

图 11-2：　　TIMER2 和 TIMER4（16 位同步）框图

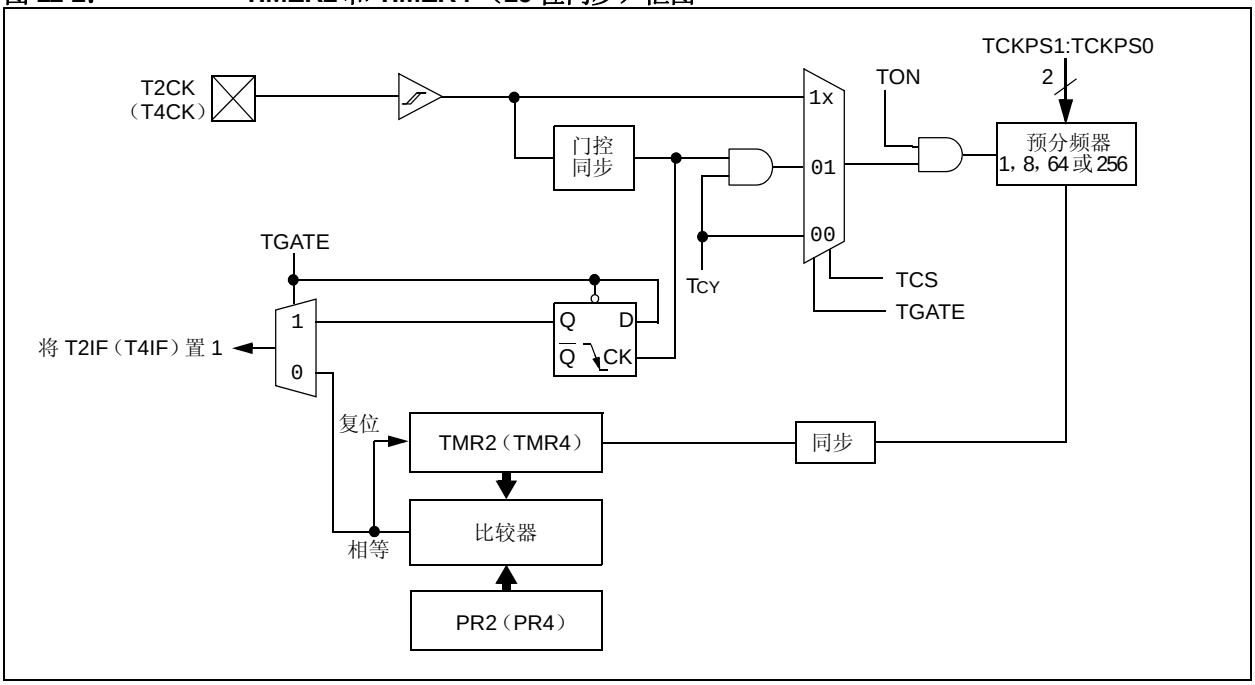
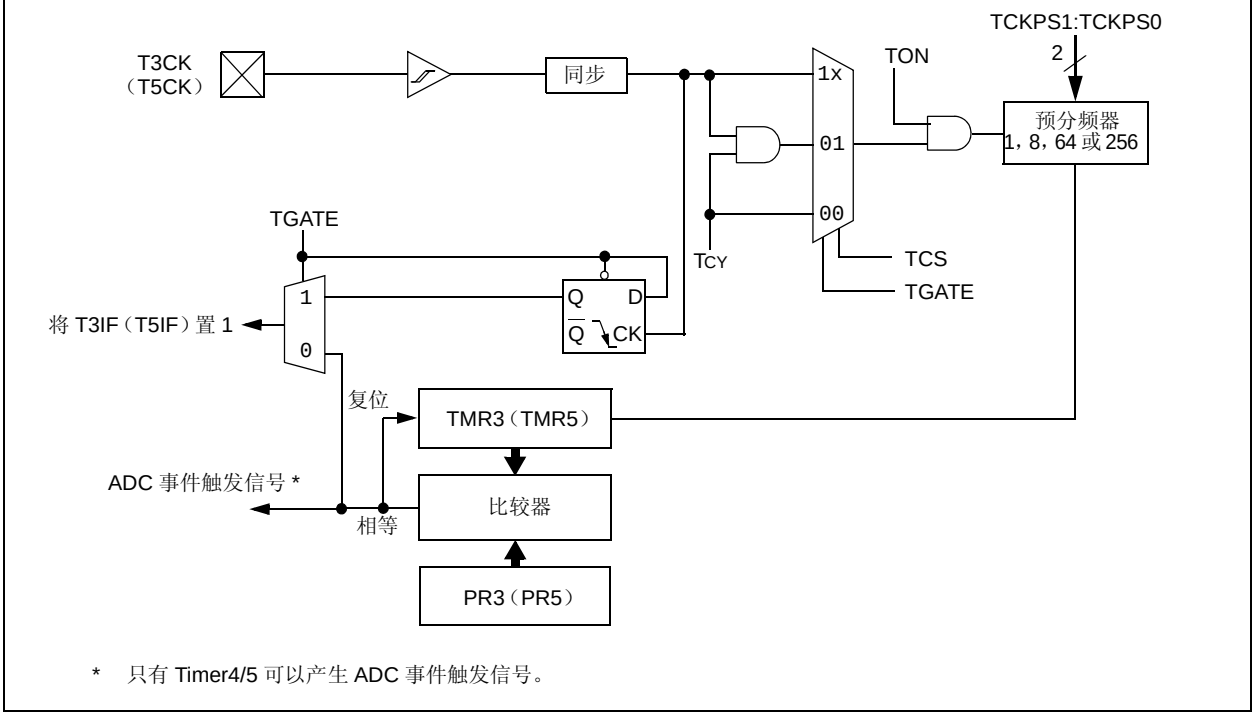


图 11-3：　　TIMER3 和 TIMER5（16 位同步）框图



*　只有 Timer4/5 可以产生 ADC 事件触发信号。

寄存器 11-1：　　TxCON：TIMER2 和 TIMER4 控制寄存器

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON	—	TSIDL	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	U-0
—	TGATE	TCKPS1	TCKPS0	T32 (1)	—	TCS	—
bit 7							bit 0

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

bit 15　**TON**：Timerx 使能位
当 TxCON<3> = 1 时：
1 = 启动 32 位 Timerx/y
0 = 停止 32 位 Timerx/y
当 TxCON<3> = 0 时：
1 = 启动 16 位 Timerx
0 = 停止 16 位 Timerx

bit 14　**未实现**：读为 0

bit 13　**TSIDL**：空闲模式停止位
1 = 当器件进入空闲模式后，模块停止工作
0 = 在空闲模式下模块继续工作

bit 12-7　**未实现**：读为 0

bit 6　**TGATE**：Timerx 门控时间累加使能位
当 TCS = 1 时：
此位可被忽略。
当 TCS = 0 时：
1 = 使能门控时间累加
0 = 禁止门控时间累加

bit 5-4　**TCKPS1:TCKPS0**：Timer2 输入时钟预分频比选择位
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

bit 3　**T32**：32 位定时器模式选择位 (1)
1 = Timerx 和 Timery 构成一个 32 位定时器
0 = Timerx 和 Timery 作为两个 16 位定时器

bit 2　**未实现**：读为 0

bit 1　**TCS**：Timerx 时钟源选择位
1 = 来自 TxCK 引脚的外部时钟 （上升沿触发定时器递增 / 递减）
0 = 内部时钟 （FOSC/2）

bit 0　**未实现**：读为 0

注　1： 在 32 位模式下， T3CON 或 T5CON 控制位不影响 32 位定时器的工作。

寄存器 11-2： TyCON：TIMER3 和 TIMER5 控制寄存器

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON(1)	—	TSIDL(1)	—	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	U-0
—	TGATE(1)	TCKPS1(1)	TCKPS0(1)	—	—	TCS(1)	—
bit 7							bit 0

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

bit 15 **TON**：Timery 使能位 (1)
1 = 启动 16 位 Timery
0 = 停止 16 位 Timery

bit 14 **未实现**：读为 0

bit 13 **TSIDL**：在空闲模式停止位 (1)
1 = 当器件进入空闲模式后，模块停止工作
0 = 在空闲模式下模块继续工作

bit 12-7 **未实现**：读为 0

bit 6 **TGATE**：Timery 门控时间累加使能位 (1)
当 TCS = 1 时：
此位可被忽略。
当 TCS = 0 时：
1 = 使能门控时间累加
0 = 禁止门控时间累加

bit 5-4 **TCKPS1:TCKPS0**：Timery 输入时钟预分频比选择位 (1)
11 = 1:256
10 = 1:64
01 = 1:8
00 = 1:1

bit 3-2 **未实现**：读为 0

bit 1 **TCS**：Timery 时钟源选择位 (1)
1 = 来自 TyCK 引脚的外部时钟 （上升沿触发定时器递增 / 递减）
0 = 内部时钟 （Fosc/2）

bit 0 **未实现**：读为 0

注 1： 当使能 32 位工作模式时 （T2CON<3> = 1），这些位不会对 Timery 的工作产生影响。定时器的所有功能都通过 T2CON 寄存器设置。

注：

12.0 输入捕捉

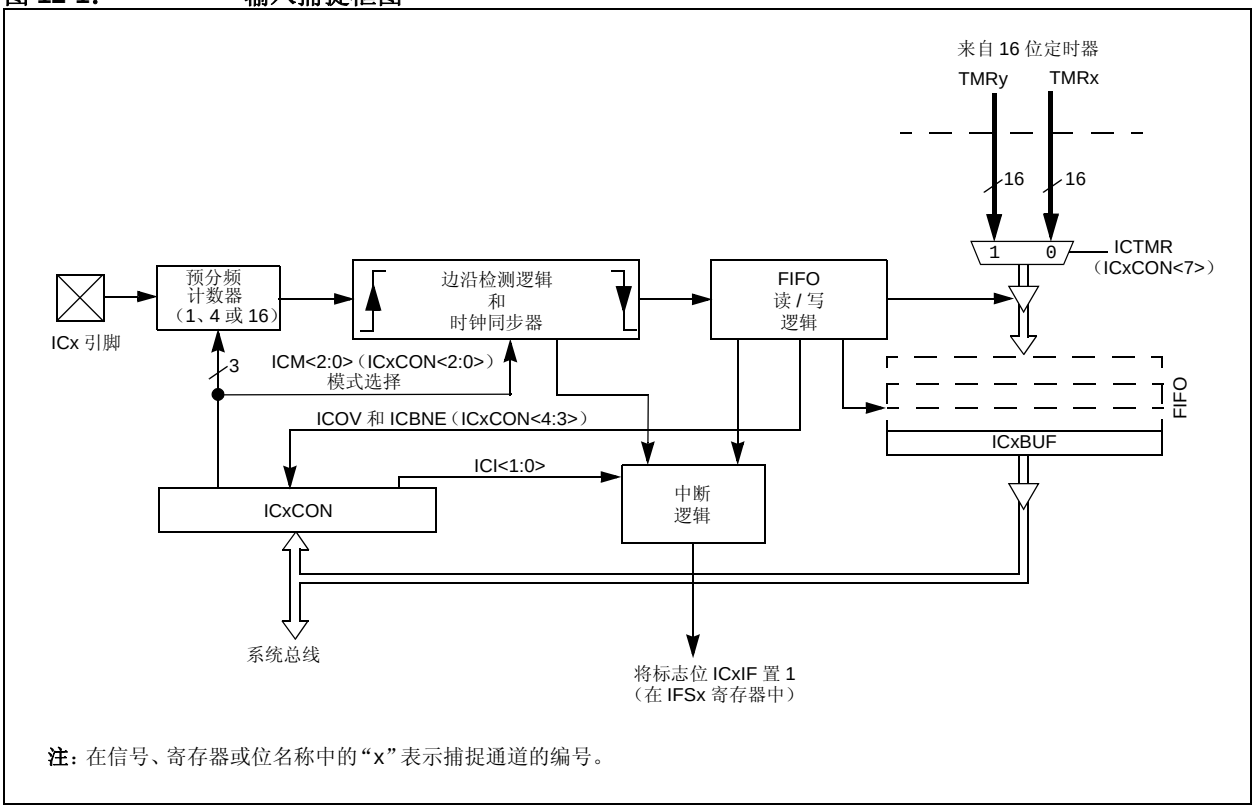
注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 15 章“输入捕捉”**（DS39701A_CN）。

输入捕捉模块有多种工作模式，可通过 ICxCON 寄存器进行选择。工作模式包括：

- 在施加给 ICx 引脚的输入信号的每个下降沿捕捉定时器值
- 在施加给 ICx 引脚的输入信号的每个上升沿捕捉定时器值
- 在施加给 ICx 引脚的输入信号的每四个上升沿捕捉定时器值
- 在施加给ICx引脚的输入信号的每16个上升沿捕捉定时器值
- 在施加给 ICx 引脚的输入信号的每个上升沿和每个下降沿均捕捉定时器值
- 捕捉引脚上的信号将器件从 CPU 休眠和空闲模式下唤醒

输入捕捉模块有一个四级深的 FIFO 缓冲区。产生 CPU 中断所需要的捕捉事件数由用户选择。

图 12-1： 输入捕捉框图



注： 在信号、寄存器或位名称中的“x”表示捕捉通道的编号。

12.1 输入捕捉寄存器

寄存器 12-1： ICxCON：输入捕捉通道 x 控制寄存器

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	ICSIDL	—	—	—	—	—
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-0, HC	R/W-0, HC	R/W-0	R/W-0	R/W-0
ICTMR(1)	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0
bit 7							bit 0

图注：	HC = 由硬件清零位		
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15-14 **未实现：**读为 0

bit 13 **ICSIDL：**输入捕捉模块空闲停止控制位
1 = 输入捕捉模块在 CPU 空闲模式下将停止工作
0 = 输入捕捉模块在 CPU 空闲模式下将继续工作

bit 12-8 **未实现：**读为 0

bit 7 **ICTMR：**输入捕捉通道 x 定时器选择位 (1)
1 = 发生捕捉事件时捕捉 TMR2 的内容
0 = 发生捕捉事件时捕捉 TMR3 的内容

bit 6-5 **ICI1:ICI0：**每次中断捕捉次数选择位
11 = 每 4 次捕捉事件中断一次
10 = 每 3 次捕捉事件中断一次
01 = 每 2 次捕捉事件中断一次
00 = 每 1 次捕捉事件中断一次

bit 4 **ICOV：**输入捕捉通道 x 溢出状态标志位 （只读）
1 = 发生了输入捕捉溢出
0 = 未发生输入捕捉溢出

bit 3 **ICBNE：**输入捕捉通道 x 缓冲器空状态位 （只读）
1 = 输入捕捉缓冲器非空，但至少可以再读一次捕捉值
0 = 输入捕捉缓冲器为空

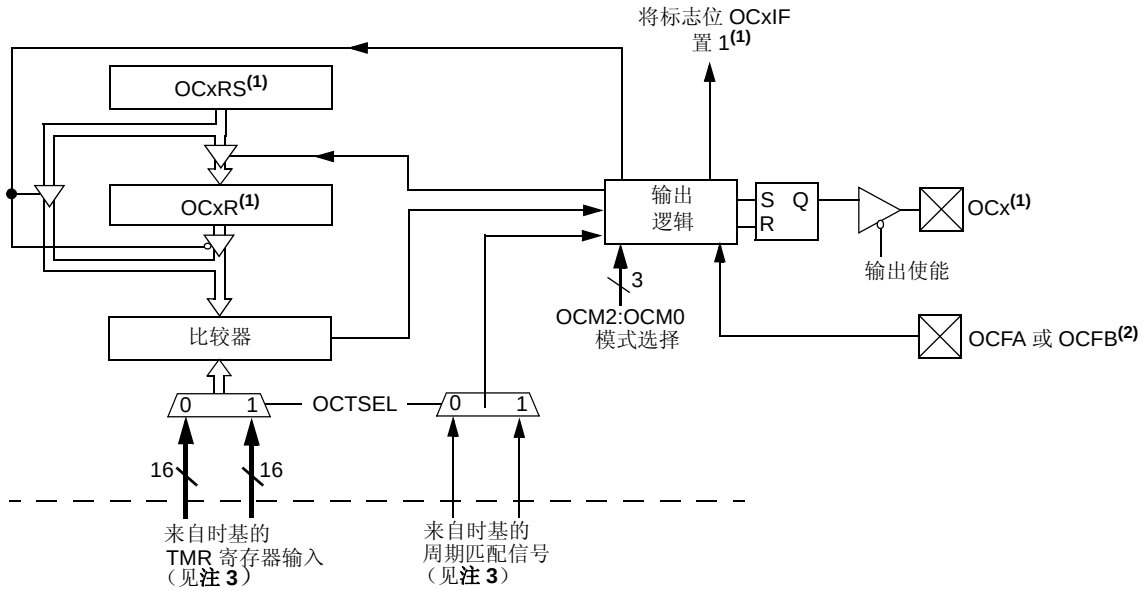
bit 2-0 **ICM2:ICM0：**输入捕捉通道 x 模式选择位
111 = 器件处于休眠或空闲模式时，输入捕捉通道只用作中断引脚
（只检测上升沿，所有其他控制位均不适用）
110 = 未使用 （模块禁止）
101 = 捕捉模式，每 16 个上升沿捕捉一次
100 = 捕捉模式，每 4 个上升沿捕捉一次
011 = 捕捉模式，每 1 个上升沿捕捉一次
010 = 捕捉模式，每 1 个下降沿捕捉一次
001 = 捕捉模式，每个边沿 （上升沿或下降沿）——该模式下 ICI<1:0> 不控制
中断产生
000 = 输入捕捉模块关闭

注 1：选择的定时器可能会和上述不同。更多详细信息请参见器件数据手册。

13.0 输出比较

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 16 章“输出比较”**（DS39706A_CN）。

图 13-1： 输出比较模块框图



注 1： 名称中带 “x”的寄存器指的是与输出比较通道 1 到 5 相对应的寄存器。
2： OCFA 引脚控制 OC1 到 OC4 通道。 OCFB 引脚控制 OC5 通道。
3： 每个输出比较通道可以使用 Timer2 或 Timer3 中的一个。

13.1 工作模式

每个输出比较模块具有以下工作模式：

- 单比较匹配模式
- 双比较匹配模式产生：
  - 单输出脉冲模式
  - 连续输出脉冲模式
- 简单脉宽调制模式：
  - 带故障保护输入
  - 不带故障保护输入

13.2 设置产生单输出脉冲

当 OCM 控制位 （OCxCON<2:0>）被设置为 100 时，选定的输出比较通道将OCx引脚初始化为低电平状态并输出一个单脉冲。

要产生单输出脉冲，需要遵循以下步骤（这些步骤假设定时器源在开始时是关闭的，但这并不是模块工作的必要条件）：

1. 确定指令时钟周期。要考虑提供给定时器的外部时钟（如果使用了）的频率和定时器预分频值的设置。
2. 计算从 TMRy 起始值（0000h）开始到输出脉冲上升沿之间的时间。
3. 根据所需的脉冲宽度和脉冲上升沿时间计算出现脉冲下降沿的时间。
4. 将在步骤 2 和步骤 3 中计算出的值分别写入比较寄存器 OCxR 和辅助比较寄存器 OCxRS。
5. 将定时器周期寄存器 PRy 的值设置为大于或等于辅助比较寄存器 OCxRS 中的值。
6. 将 OCM 位设置为 100，并将 OCTSEL（OCxCON<3>）位设置为所需的定时器源。现在 OCx 引脚被驱动为低电平。
7. 将 TON（TyCON<15>）位置为 1，使能计数的比较时基。
8. 当 TMRy 和 OCxR 第一次匹配时，OCx 引脚将被驱动为高电平。
9. 当递增计数器 TMRy 和辅助比较寄存器 OCxRS 发生匹配时，脉冲的第二个边沿（即后沿，下降沿）在 OCx 引脚上输出。此时 OCx 引脚不再驱动输出额外的脉冲，将一直保持低电平。第二次比较匹配事件会导致 OCxIF 中断标志位置 1，如果已通过将 OCxIE 位置 1 允许了该中断，那么将产生中断。有关外设中断的更多信息请参见**第 6.0 节 “中断控制器”**。
10. 要输出另一个单脉冲，首先改写定时器和比较寄存器的设置（如果需要的话），并随后通过执行写操作将 OCM 位设置为 100。并不一定需要禁止或重新使能定时器并清零 TMRy 寄存器，但这样做有利于确定脉冲的时间边界。

在出现输出脉冲的下降沿后不必禁止输出比较模块。重写 OCxCON 寄存器的值可以触发产生另一个脉冲。

13.3 设置产生连续输出脉冲

当控制位 OCM（OCxCON<2:0>）被设置为 101 时，选定的输出比较通道将 OCx 引脚初始化为低电平，并在每次比较匹配时产生输出脉冲。

用户要配置模块产生连续输出脉冲，需要遵循以下步骤（这些步骤假设定时器源在开始时是关闭的，但是这并不是模块工作的必要条件）：

1. 确定指令时钟周期。要考虑提供给定时器的外部时钟（如果使用了）的频率和定时器预分频值的设置。
2. 计算从 TMRy 起始值（0000h）开始到出现输出脉冲上升沿之间的时间。
3. 根据所需的脉冲宽度和脉冲上升沿时间计算出现脉冲下降沿的时间。
4. 将在步骤 2 和步骤 3 中计算出的值分别写入比较寄存器 OCxR 和辅助比较寄存器 OCxRS。
5. 将定时器周期寄存器 PRy 的值设置为大于或等于辅助比较寄存器 OCxRS 中的值。
6. 将 OCM 位设置为 101，并将 OCTSEL 位设置为所需的定时器源。现在 OCx 引脚被驱动为低电平。
7. 将 TON（TyCON<15>）位置 1，使能比较时基。
8. 在 TMRy 和 OCxR 第一次匹配时，OCx 引脚将被驱动为高电平。
9. 当比较时基 TMRy 和辅助比较寄存器 OCxRS 发生匹配时，脉冲的第二个边沿（即后沿，下降沿）在 OCx 引脚上输出。
10. 第二次比较匹配事件会导致 OCxIF 中断标志位置 1。
11. 当比较时基和相应的周期寄存器中的值匹配时，TMRy 寄存器复位为 0x0000 并重新开始计数。
12. 重复步骤 8 到步骤 11，可产生无限连续脉冲。OCxIF 标志位在每次 OCxRS-TMRy 比较事件发生时置 1。

13.4 脉宽调制模式

当将输出比较模块配置为 PWM 操作模式时，需遵循以下步骤：

1. 通过写选定的定时器周期寄存器 （PRy）设置 PWM 周期。
2. 通过写 OCxRS 寄存器设置 PWM 占空比。
3. 将初始占空比写入 OxCR 寄存器。
4. 如果需要的话，允许定时器和输出比较模块的中断。如需要使用 PWM 故障引脚，则会用到输出比较中断。
5. 通过写输出比较模式位 OCM<2:0>（OCxCON<2:0>）将输出比较模块配置为两种 PWM 工作模式中的一种。
6. 设置 TMRy 预分频值并通过设置 TON （TxCON<15>） = 1 使能时基。

注： OCxR 寄存器应该在输出比较模块第一次使能之前被初始化。当模块工作在 PWM 模式时， OCxR 寄存器成为只读占空比寄存器。 OCxR 寄存器中保存的值成为第一个 PWM 周期的占空比。直到发生时基周期匹配，才将占空比缓冲寄存器 OCxRS 的内容传送到 OCxR。

13.4.1 PWM 周期

PWM 周期可通过写入 PRy （定时器周期寄存器）来设定。可使用公式 13-1 计算 PWM 周期。

公式 13-1： 计算 PWM 周期 (1)

PWM 周期 = [(PRy) + 1] • TCY • (定时器预分频值)
其中：
PWM 频率 = 1/[PWM 周期]

注 1： 上述公式基于 TCY = TOSC * 2，打盹模式和 PLL 是被禁止的。

注： 若 PRy 值为 N，则 PWM 周期将等于 （N + 1）个时基计数周期。例如：将 7 写入 PRy 寄存器，将产生 8 个时基周期的 PWM 周期。

13.4.2 PWM 占空比

PWM 占空比是通过写 OCxRS 寄存器设定的。在任何时间都可以写入 OCxRS 寄存器，但是在 PRy 和 TMRy 发生匹配 （即周期完成）前占空比值不会被锁存到 OCxR。这一机制为 PWM 占空比提供了双重缓冲，对于消除 PWM 操作中产生的毛刺至关重要。在 PWM 模式中， OCxR 是只读寄存器。

以下是 PWM 占空比的部分重要边界参数：

- 如果占空比寄存器 OCxR 中的值为 0000h，则 OCx 引脚将保持低电平 （0% 占空比）。
- 如果 OCxR 中的值大于 PRy （定时器周期寄存器）中的值，则引脚将保持高电平 （占空比为 100%）。
- 如果 OCxR 中的值等于 PRy 中的值， OCx 引脚在一个时基计数周期内为低电平，而在其余计数周期内均为高电平。

欲知 PWM 模式时序的详细信息，请参见例 13-1。表 13-1 给出了当器件工作速度为 10 个 MIPS 时 PWM 频率和分辨率的示例。

公式 13-2： 计算最大 PWM 分辨率 (1)

$$\text{最大 PWM 分辨率（位）} = \frac{\log_{10}\left(\frac{F_{CY}}{F_{PWM} \bullet (\text{定时器预分频值})}\right)}{\log_{10}(2)} \text{ 位}$$

注 1： 上述公式基于 FCY = FOSC/2，打盹模式和 PLL 是被禁止的。

例 13-1：　　计算 PWM 周期和占空比 (1)

1. 给定 PWM 频率为 52.08 kHz，计算周期寄存器的值。其中 FOSC = 8 MHz 且带有 PLL （32 MHz 器件时钟速率），Timer2 预分频比被设置为 1:1。

   TCY = 2/FOSC = 62.5 ns

   PWM 周期　= 1/PWM 频率 = 1/52.08 kHz = 19.2 µs

   PWM 周期　= (PR2 + 1) • TCY • (Timer2 预分频值)

   19.2 µs　= (PR2 + 1) • 62.5 ns • 1

   PR2　= 306

2. PWM 频率为 52.08 kHz，器件时钟速率为 32MHz 时，计算占空比的最大分辨率：

   PWM 分辨率　= $\log_{10}(F_{CY}/F_{PWM})/\log_{10}2)$ 位

   = $(\log_{10}(16\ MHz/52.08\ kHz)/\log_{10}2)$ 位

   = 8.3 位

注　1： 上述公式基于 TCY = TOSC * 2，打盹模式和 PLL 是被禁止的。

表 13-1：　　器件工作速度为 4 MIPS 时的 PWM 频率和分辨率示例 （FCY = 4 MHz） (1)

PWM 频率	7.6 Hz	61 Hz	122 Hz	977 Hz	3.9 kHz	31.3 kHz	125 kHz
定时器预分频比	8	1	1	1	1	1	1
周期寄存器值	FFFFh	FFFFh	7FFFh	0FFFh	03FFh	007Fh	001Fh
分辨率 （位）	16	16	15	12	10	7	5

注　1： 表中内容基于 TCY = TOSC * 2，打盹模式和 PLL 是被禁止的。

表 13-2：　　器件工作速度为 16 MIPS 时的 PWM 频率和分辨率示例 （FCY = 16 MHz） (1)

PWM 频率	30.5 Hz	244 Hz	488 Hz	3.9 kHz	15.6 kHz	125 kHz	500 kHz
定时器预分频比	8	1	1	1	1	1	1
周期寄存器值	FFFFh	FFFFh	7FFFh	0FFFh	03FFh	007Fh	001Fh
分辨率 （位）	16	16	15	12	10	7	5

注　1： 表中内容基于 TCY = TOSC * 2，打盹模式和 PLL 是被禁止的。

寄存器 13-1：　　OCxCON：输出比较通道 x 控制寄存器

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
—	—	OCSIDL	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	R-0,HC	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	OCFLT	OCTSEL(1)	OCM2	OCM1	OCM0
bit 7							bit 0

图注：	HC = 由硬件清零位		
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15-14 **未实现：**读为 0

bit 13 **OCSIDL：**输出比较通道 x 空闲模式停止控制位
1 = 输出比较通道 x 将在 CPU 空闲模式下停止工作
0 = 输出比较通道 x 将在 CPU 空闲模式下继续工作

bit 12-5 **未实现：**读为 0

bit 4 **OCFLT：**PWM 故障条件状态位 (1)
1 = 已经发生了 PWM 故障条件 （只可由硬件清零）
0 = 没有发生 PWM 故障条件 （只有在 OCM<2:0> = 111 时才使用该位）

bit 3 **OCTSEL：**输出比较通道 x 定时器选择位 (1)
1 = Timer3 是输出比较通道 x 的时钟源
0 = Timer2 是输出比较通道 x 的时钟源

bit 2-0 **OCM2:OCM0：**输出比较通道 x 模式选择位
111 = OCx 处于 PWM 模式，使能故障引脚 (2)
110 = OCx 处于 PWM 模式，禁止故障引脚 (2)
101 = 初始化 OCx 引脚为低电平，在 OCx 引脚上产生连续的输出脉冲
100 = 初始化 OCx 引脚为低电平，在 OCx 引脚上产生单个输出脉冲
011 = 比较匹配事件使 OCx 引脚的电平翻转
010 = 初始化 OCx 引脚为高电平，比较匹配事件强制 OCx 引脚为低电平
001 = 初始化 OCx 引脚为低电平，比较匹配事件强制 OCx 引脚为高电平
000 = 禁止输出比较通道

注 1：想要了解输出比较模块可用的特定时基，请参见器件数据手册。
2： OCFA 引脚控制 OC1-OC4 通道。 OCFB 引脚控制 OC5 通道。

注：

14.0 串行外设接口 （SPI）

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 23 章 “串行外设接口 （SPI）”**（DS39699A_CN）。

串行外设接口（Serial Peripheral Interface，SPI）模块是一个同步串行接口，可用于与其他外设或者单片机进行通信。这些外设可以是串行 EEPROM、移位寄存器、显示驱动器和 A/D 转换器等。 SPI 模块与 Motorola 的 SPI 和 SIOP 接口兼容。

该模块可在两种缓冲器模式下工作。在标准模式下，通过一个串行缓冲器移动数据。在增强型缓冲器模式下，通过一个 8 级深的 FIFO 缓冲器移动数据。

注： 无论是在标准还是增强型缓冲器模式下，都不要对 SPIxBUF 寄存器执行读—修改—写操作 （如位操作指令）。

不论工作在主控模式或从动模式下，该模块都支持一个基本帧 SPI 协议。支持全部四种帧 SPI 配置。

SPI 串行接口由以下四个引脚组成：

- SDIx：串行数据输入
- SDOx：串行数据输出
- SCKx：移位时钟输入或输出
- SSx：低电平有效从动选择模式或帧同步 I/O 脉冲

SPI 模块可以被配置为使用 2 个、3 个或 4 个引脚工作。在 3 引脚模式下，不使用 SSx。在 2 引脚模式下，不使用 SDOx 和 SSx。

图 14-1 和图 14-2 为该模块的框图。

注： 在本章中， SPI 模块统称为 SPIx，或分别称为 SPI1 和 SPI2。特殊功能寄存器也使用类似的符号表示方法。例如， SPIxCON 指 SPI1 或 SPI2 模块的控制寄存器。

要将 SPI 模块设置为工作在标准主控模式下，请遵循以下步骤：

1. 如果使用中断：
   a) 将相应的 IFSx 寄存器中的 SPIxIF 位清零。
   b) 将相应的 IECx 寄存器中的 SPIxIE 位置 1。
   c) 写相应的 IPCx 寄存器中的 SPIxIP 位以设置中断优先级。
2. 将所需设置写入 SPIxCON 寄存器，同时 MSTEN （SPIxCON1<5>） = 1。
3. 将 SPIROV 位 （SPIxSTAT<6>）清零。
4. 通过将 SPIEN 位 （SPIxSTAT<15>）置 1 使能 SPI 工作。
5. 将待发送数据写入 SPIxBUF 寄存器。数据一旦写入，发送 （或接收）就会立即开始。

要将 SPI 模块设置为工作在标准从动模式下，请遵循以下步骤：

1. 将 SPIxBUF 寄存器清零。
2. 如果使用中断：
   a) 将相应的 IFSx 寄存器中的 SPIxIF 位清零。
   b) 将相应的 IECx 寄存器中的 SPIxIE 位置 1。
   c) 写相应的 IPCx 寄存器中的 SPIxIP 位以设置中断优先级。
3. 将所需设置写入 SPIxCON1 和 SPIxCON2 寄存器，同时 MSTEN （SPIxCON1<5>） = 0。
4. 将 SMP 位清零。
5. 如果 CKE 位置 1，则 SSEN 位（SPIxCON1<7>）也必须置 1 以使能 SSx 引脚。
6. 将 SPIROV 位 （SPIxSTAT<6>）清零。
7. 通过将 SPIEN 位 （SPIxSTAT<15>）置 1 使能 SPI 工作。

要将 SPI 模块设置为工作在增强型缓冲器主控模式下，请遵循以下步骤：

1. 如果使用中断：
   a) 将相应的 IFSx 寄存器中的 SPIxIF 位清零。
   b) 将相应的 IECx 寄存器中的 SPIxIE 位置 1。
   c) 写入相应的 IPCx 寄存器中的 SPIxIP 位。
2. 将所需设置写入 SPIxCON1 和 SPIxCON2 寄存器，同时 MSTEN （SPIxCON1<5>） = 1。
3. 将 SPIROV 位 （SPIxSTAT<6>）清零。
4. 通过将 SPIBEN 位 （SPIxCON2<0>）置 1 选择增强型缓冲器模式。
5. 通过将 SPIEN 位 （SPIxSTAT<15>）置 1 使能 SPI 工作。
6. 将待发送数据写入 SPIxBUF 寄存器。数据一旦写入发送 （或接收）就会立即开始。

要将 SPI 模块设置为工作在增强型缓冲器从动模式下，请遵循以下步骤：

1. 将 SPIxBUF 寄存器清零。
2. 如果使用中断：
   - 将相应的 IFSx 寄存器中的 SPIxIF 位清零。
   - 将相应的 IECx 寄存器中的 SPIxIE 位置 1。
   - 写相应 IPCx 寄存器中的 SPIxIP 位以设置中断优先级。
3. 将所需设置写入 SPIxCON1 和 SPIxCON2 寄存器，同时 MSTEN （SPIxCON1<5>） = 0。
4. 将 SMP 位清零。
5. 如果 CKE 位置 1，则 SSEN 位也必须置 1 以使能 SSx 引脚。
6. 将 SPIROV 位 （SPIxSTAT<6>）清零。
7. 通过将 SPIBEN 位 （SPIxCON2<0>）置 1 选择增强型缓冲器模式。
8. 通过将 SPIEN 位 （SPIxSTAT<15>）置 1 使能 SPI 工作。

图 14-1： SPIx 模块框图 （标准模式）

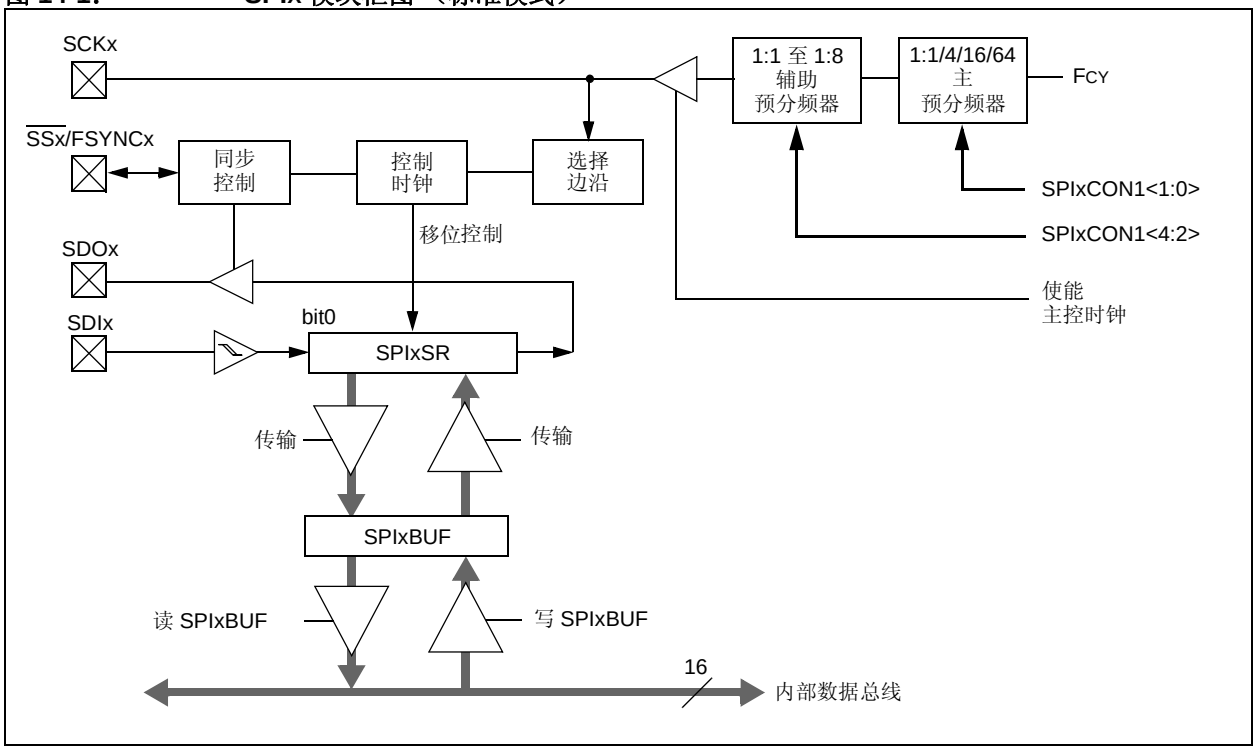
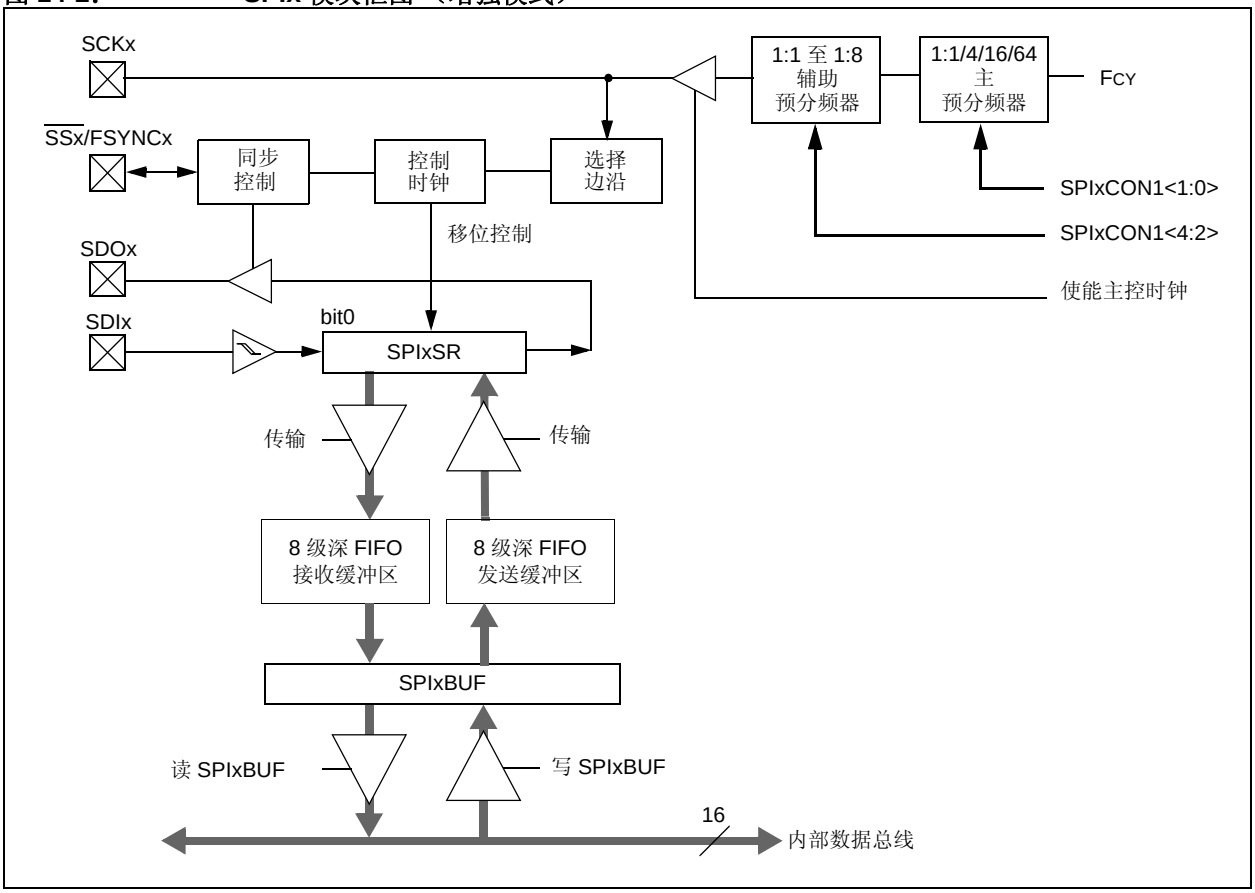


图 14-2：　　SPIx 模块框图 （增强模式）



寄存器 14-1： SPIxSTAT：SPIx 状态和控制寄存器

R/W-0	U-0	R/W-0	U-0	U-0	R-0	R-0	R-0
SPIEN	—	SPISIDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0
bit 15							bit 8

R/W-0	R/C-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0
SRMPT	SPIROV	SRXMPT	SISEL2	SISEL1	SISEL0	SPITBF	SPIREF
bit 7							bit 0

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15 **SPIEN**：SPIx 使能位
1 = 使能模块并将 SCKx、 SDOx、 SDIx 和 SSx 配置为串口引脚
0 = 禁止模块

bit 14 **未实现**：读为 0

bit 13 **SPISIDL**：空闲模式停止位
1 = 当器件进入空闲模式后，模块停止工作。
0 = 在空闲模式下模块继续工作

bit 12-11 **未实现**：读为 0

bit 10-8 **SPIBEC2:SPIBEC0**：SPIx 缓冲器数据个数位
主控模式：
等待传输的 SPI 数据个数
从动模式：
未读取的 SPI 传输数据个数

bit 7 **SRMPT**：移位寄存器 （SPIxSR）空位 （在增强型缓冲模式下有效）
1 = SPIx 移位寄存器为空，准备发送或接收
0 = SPIx 移位寄存器非空，读为 0

bit 6 **SPIROV**：接收溢出标志位
1 = 一个新字节 / 字已接收并丢弃。由于用户软件尚未读取 SPIxBUF 寄存器中已接收的数据。
0 = 没有发生溢出

bit 5 **SRXMPT**：接收 FIFO 空位 （在增强型缓冲模式下有效）
1 = 接收 FIFO 为空
0 = 接收 FIFO 非空

bit 4-2 **SISEL2:SISEL0**：SPIx 缓冲区中断模式位 （在增强型缓冲模式下有效）
111 = 当 SPIx 发送缓冲器满时产生中断 （SPITBF 位置 1）
110 = 最后一位移入 SPIxSR 导致发送 FIFO 为空时产生中断
101 = 最后一位被移出 SPIxSR 时产生中断，此时发送完成
100 = 有数据移入 SPIxSR 导致发送 FIFO 有待发送数据时产生中断
011 = 当 SPIx 接收缓冲器满时产生中断 （SPIRBF 位置 1）
010 = 当 SPIx 接收缓冲器被填满 3/4 或更多时产生中断
001 = 接收缓冲器中有数据时产生中断 （SRMPT 位置 1）
000 = 读取接收缓冲器中的最后一个数据，导致缓冲器为空时产生中断 （SRXMPT 位置 1）

寄存器 14-1：SPIxSTAT：SPIx 状态和控制寄存器 （续）

bit 1 **SPITBF：**SPIx 发送缓冲器满状态位
1 = 未开始发送， SPIxTXB 满
0 = 发送开始， SPIxTXB 空
在标准缓冲器模式下：
当 CPU 通过写 SPIxBUF 地址单元装载 SPIxTXB 时，该位由硬件自动置 1。
当 SPIx 模块将数据从 SPIxTXB 传输到 SPIxSR 时，该位由硬件自动清零。
在增强型缓冲器模式下：
当 CPU 通过写 SPIxBUF 地址单元装载最后一个可用缓冲单元时，该位由硬件自动置 1。
当有空的缓冲器单元可由 CPU 写入时由硬件自动清零。

bit 0 **SPIRBF：**SPIx 接收缓冲器满状态位
1 = 接收完成， SPIxRXB 满
0 = 接收未完成， SPIxRXB 空
在标准缓冲器模式下：
当 SPIx 将数据从 SPIxSR 传输到 SPIxRXB 时，该位由硬件自动置 1。
当内核通过读 SPIxBUF 地址单元读 SPIxRXB 时，该位由硬件自动清零。
在增强型缓冲器模式下：
当 SPIx 将数据从 SPIxSR 传输到缓冲器填充了最后一个未读的缓冲单元时，该位由硬件自动置 1。
当有空的缓冲器单元可接收来自 SPIxSR 的传输数据时，该位由硬件自动清零。

寄存器 14-2： SPIxCON1：SPIx 控制寄存器 1

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE(1)
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0
bit 7							bit 0

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

bit 15-13 **未实现：**读为 0

bit 12 **DISSCK：**SCKx 引脚禁止位 （仅用于 SPI 主控模式）
1 = 禁止内部 SPI 时钟，引脚用作 I/O
0 = 使能内部 SPI 时钟

bit 11 **DISSDO：**SDOx 引脚禁止位
1 = 模块不使用 SDOx 引脚，引脚用作 I/O
0 = SDOx 引脚受模块控制

bit 10 **MODE16：**字 / 字节通信选择位
1 = 每次通信数据为字宽 （16 位）
0 = 每次通信数据为字节宽 （8 位）

bit 9 **SMP：**SPIx 数据输入采样点选择位
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
从动模式：
当在从动模式下使用 SPIx 时，必须将该位清零。

bit 8 **CKE：**SPIx 时钟边沿选择位 (1)
1 = 串行输出数据在时钟由有效状态变为空闲状态时改变 （见 bit 6）
0 = 串行输出数据在时钟由空闲状态变为有效状态时改变 （见 bit 6）

bit 7 **SSEN：**从动选择使能 （从动模式）位
1 = $\overline{SS}$x 引脚用于从动模式
0 = 模块不使用 $\overline{SS}$x 引脚，引脚用作端口。

bit 6 **CKP：**时钟极性选择位
1 = 时钟信号空闲状态为高电平；有效状态为低电平
0 = 时钟信号空闲状态为低电平；有效状态为高电平

bit 5 **MSTEN：**主控模式使能位
1 = 主控模式
0 = 从动模式

bit 4-2 **SPRE2:SPRE0：**辅助预分频比 （主控模式）位
111 = 辅助预分频比为 1:1
110 = 辅助预分频比为 2:1
...
000 = 辅助预分频比为 8:1

bit 1-0 **PPRE1:PPRE0：**主预分频比 （主控模式）位
11 = 主预分频比为 1:1
10 = 主预分频比为 4:1
01 = 主预分频比为 16:1
00 = 主预分频比为 64:1

注 1：在帧 SPI 模式下不使用 CKE 位。在该模式 （FRMEN = 1）下，用户应将该位编程为 0。

寄存器 14-3： SPIxCON2：SPIx 控制寄存器 2

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	SPIFE	SPIBEN
bit 7							bit 0

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

bit 15 **FRMEN**：帧 SPIx 支持位
1 = 使能帧 SPIx 支持
0 = 禁止帧 SPIx 支持

bit 14 **SPIFSD**：$\overline{SSx}$ 引脚上的帧同步脉冲方向控制位
1 = 帧同步脉冲输入 （从动模式）
0 = 帧同步脉冲输出 （主控模式）

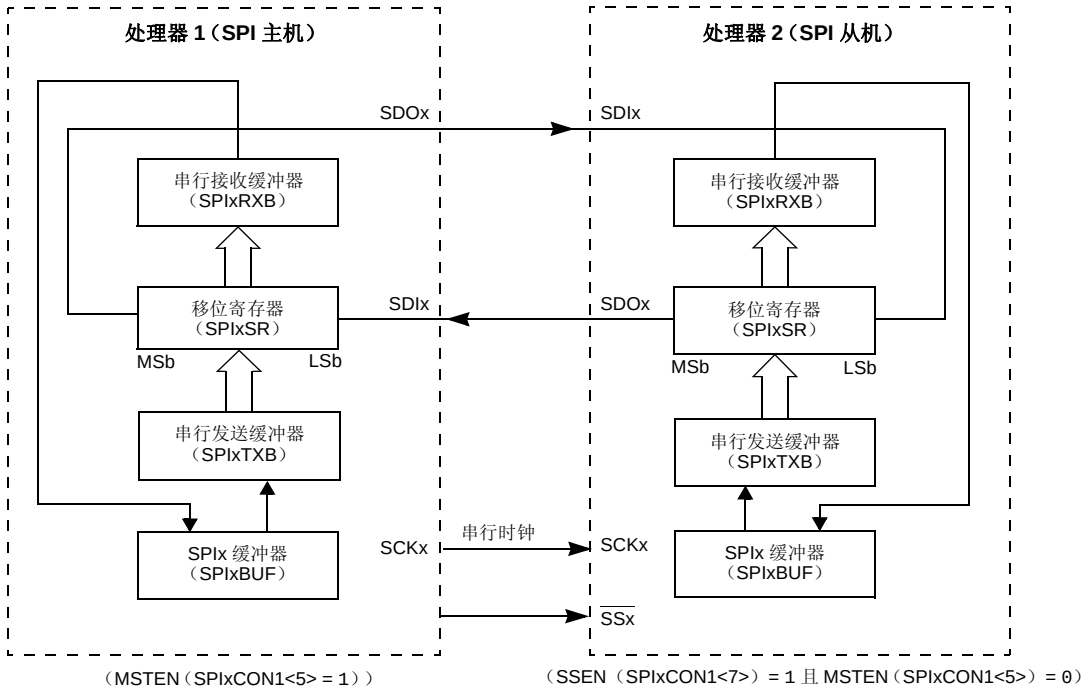
bit 13 **SPIFPOL**：帧同步脉冲极性位 （仅用于帧模式）
1 = 帧同步脉冲高电平有效
0 = 帧同步脉冲低电平有效

bit 12-2 **未实现**：读为 0

bit 1 **SPIFE**：帧同步脉冲边沿选择位
1 = 帧同步脉冲与第一位时钟同步
0 = 帧同步脉冲比第一位时钟超前

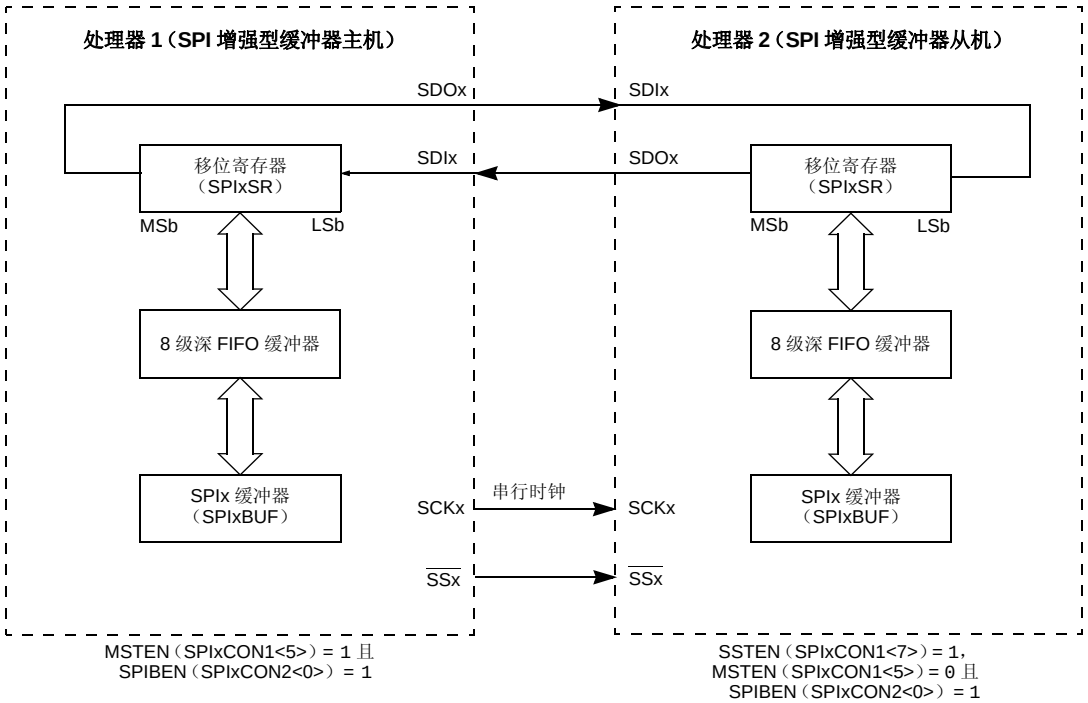
bit 0 **SPIBEN**：增强型缓冲器使能位
1 = 使能增强型缓冲器
0 = 禁止增强型缓冲器 （传统模式）

图 14-3：　　SPI 主 / 从连接 （标准模式）



注　1： 在从动模式下 $\overline{SSx}$ 引脚是可选的。
2： 用户必须向 SPIxBUF 写入发送数据，从 SPIxBUF 读取接收数据。SPIxTXB 和 SPIxRXB 寄存器是通过存储器映射到 SPIxBUF 的。

图 14-4：　　SPI 主 / 从连接 （增强型缓冲器模式）



注　1： 在从动模式下 $\overline{SSx}$ 引脚是可选的。
2： 用户必须向 SPIxBUF 写入发送数据，从 SPIxBUF 读取接收数据。SPIxTXB 和 SPIxRXB 寄存器是通过存储器映射到 SPIxBUF 的。

图 14-5：　　SPI 主机、帧主机连接图

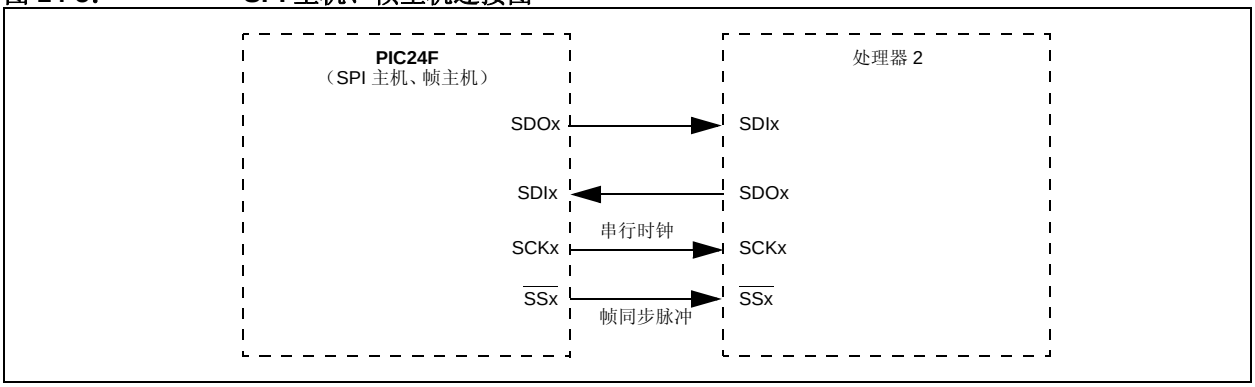


图 14-6：　　SPI 主机、帧从机连接图

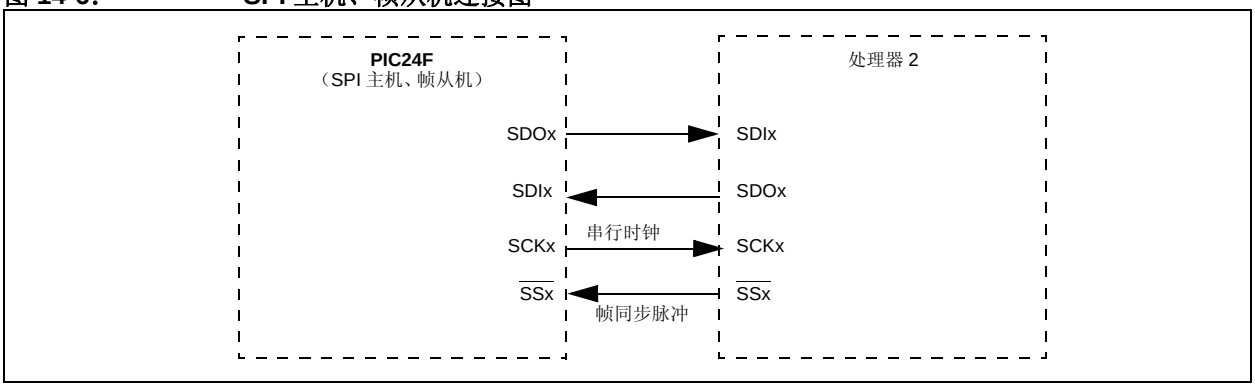


图 14-7：　　SPI 从机、帧主机连接图

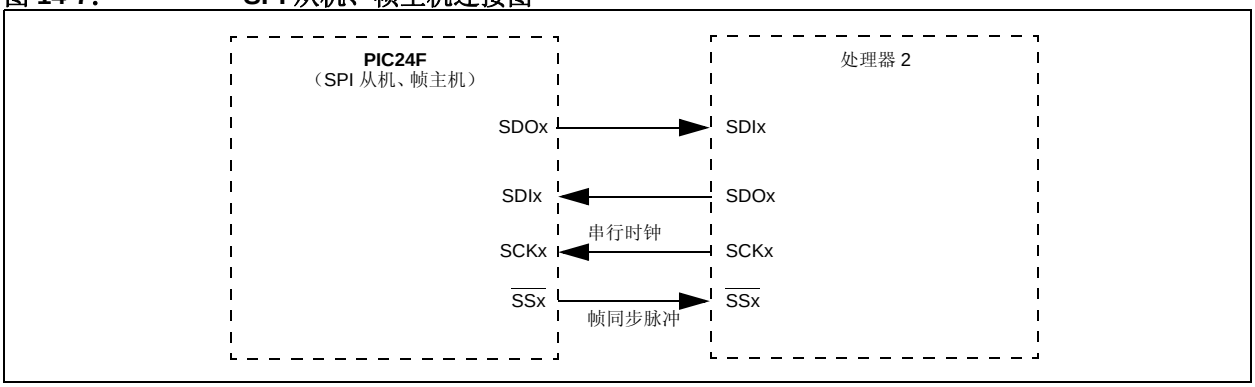
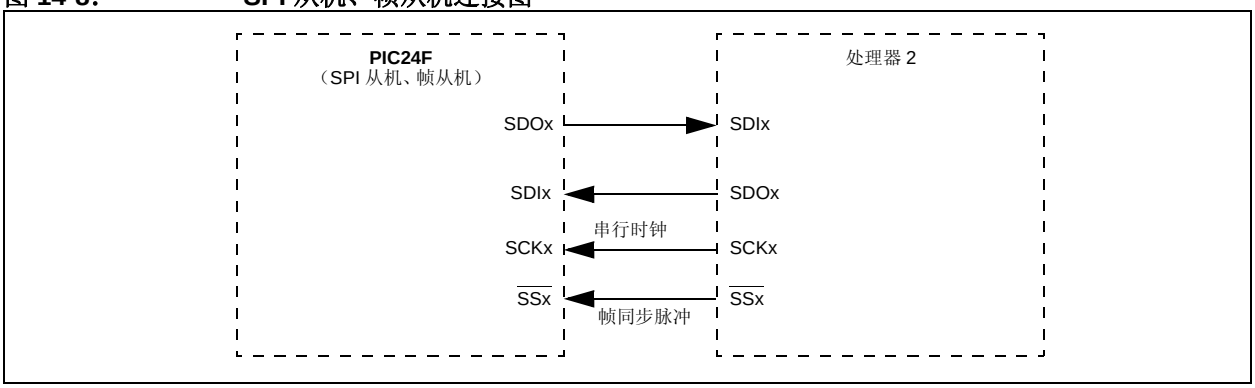


图 14-8：　　SPI 从机、帧从机连接图



公式 14-1：　　器件和 SPI 时钟速度之间的关系 (1)

$$FSCK = \frac{FCY}{\text{主预分频值} * \text{辅助预分频值}}$$

注　1： 上述公式基于 FCY = FOSC/2，且打盹模式和 PLL 是被禁止的。

表 14-1：　　采样 SCK 频率 (1,2)

FCY = 16 MHz		**辅助预分频比设置**				
		1:1	**2:1**	**4:1**	**6:1**	**8:1**
主预分频比设置	1:1	无效	8000	4000	2667	2000
	4:1	4000	2000	1000	667	500
	16:1	1000	500	250	167	125
	64:1	250	125	63	42	31
FCY = 5 MHz						
主预分频比设置	1:1	5000	2500	1250	833	625
	4:1	1250	625	313	208	156
	16:1	313	156	78	52	39
	64:1	78	39	20	13	10

注　1： 上述内容基于 FCY = FOSC/2，且打盹模式和 PLL 是被禁止的。

2： 表中 SCKx 频率的单位为 kHz。

15.0 I²C™

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 24 章 “I²C™”**（DS39702A_CN）。

I²C（Inter-Integrated Circuit）模块是一个串行接口，用来与其他外设或单片机通信。这些外设可以是串行 EEPROM、显示驱动器和 A/D 转换器等。

I²C 模块支持以下特征：

- 独立的主控和从动逻辑
- 7 位和 10 位器件地址
- 由 I²C 协议定义的广播呼叫地址
- 时钟延长，为处理器响应从动数据请求提供延时
- 100 kHz 和 400 kHz 总线规范
- 可配置的地址屏蔽
- 多主机模式可防止在仲裁时丢失报文
- 总线重复器模式，作为从机接收所有的报文
- 自动 SCL

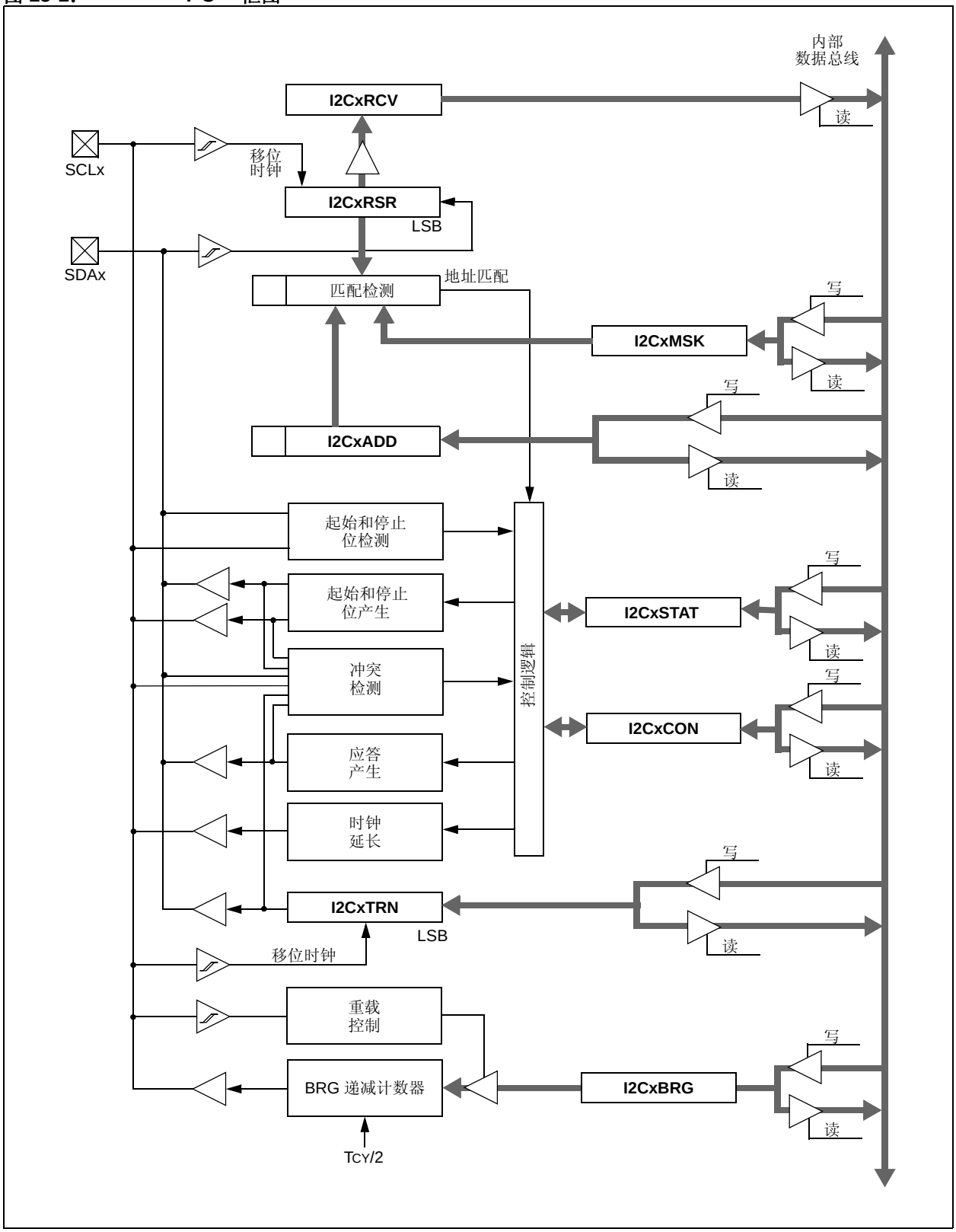
图 15-1 为该模块的框图。

15.1 作为主机在单主机环境中通信

在主控模式下发送报文的具体过程由与之通信的器件所使用的通信协议决定。通常来讲，过程如下：

1. 在 SDAx 和 SCLx 上发起一个启动条件。
2. 发送一个 I²C 器件地址字节到从器件，表明将要进行写操作。
3. 等待并验证来自从器件的应答。
4. 将第一个数据字节（有时是命令）发送到从器件。
5. 等待并验证来自从器件的应答。
6. 将串行存储器地址的低字节发送到从器件。
7. 重复步骤 4 和步骤 5 直到数据字节发送完毕。
8. 在 SDAx 和 SCLx 上发起一个重复启动条件。
9. 将器件地址字节发送给从器件，表明将要进行读操作。
10. 等待并验证来自从器件的应答。
11. 使能主控接收模式来接收串行存储器数据。
12. 在数据字节接收完毕时产生 ACK 或 NACK 条件。
13. 在 SDAx 和 SCLx 上产生停止条件。

图 15-1：　　I²C™ 框图



15.2 作为总线主控器件工作时设置波特率

可使用以下公式计算波特率发生器重载值：

公式 15-1：(1)

$$I2CxBRG = (F_{CY}/F_{SCL} - F_{CY}/10{,}000{,}000) - 1$$

注 1： 上述公式基于 FCY = FOSC/2，且打盹模式和 PLL 是被禁止的。

15.3 从动地址屏蔽

I2CxMSK 寄存器（寄存器 15-3）可以指定 7 位地址或 10 位地址中的某些位可以是任意值。将 I2CxMSK 寄存器中的某位置 1（= 1），将使从器件在相应地址位为 0 或 1 时都作出响应。例如，当 I2CxMSK 被设置为 00100000 时，从动模块能够识别地址 0000000 和 00100000。

要使能地址屏蔽，必须通过清零 IPMIEN 位（I2CxCON<11>）以禁止 IPMI（Intelligent Peripheral Management Interface）。

表 15-1： I²C™ 时钟速率 (1,3,4)

所需的系统 FSCL	FCY	I2CxBRG 值		实际 FSCL
		（10 进制）	（16 进制）	
100 kHz	16 MHz	157	9D	100 kHz
100 kHz	8 MHz	78	4E	100 kHz
100 kHz	4 MHz	39	27	99 kHz
400 kHz	16 MHz	37	25	404 kHz
400 kHz	8 MHz	18	12	404 kHz
400 kHz	4 MHz	9	9	385 kHz(2)
400 kHz	2 MHz	4	4	385 kHz(2)
1 MHz	16 MHz	13	D	1,026 KHz
1 MHz	8 MHz	6	6	1,026 KHz
1 MHz	4 MHz	3	3	909 KHz

图注： 阴影行表示在给定 FSCL 和 FCY 下无效的重载值。

注 1： 上述数据基于 TCY = TOSC * 2(2)，且打盹模式和 PLL 是被禁止的。

2： 对于这个 FCY 值，这是最接近 400 kHz 的值。

3： 如需得到 FSCL = 1 MHz， FCY = 2 MHz 是所需的最小输入时钟频率。

4： I2CxBRG 的值不能小于 2。

对 I²C 协议进行了修改，保留了一些 I²C 地址，在从模式下，不会对这些地址进行应答。地址掩码不会对操作产生影响。表 15-2 汇总了这些保留的地址。

表 15-2： I²C™ 保留地址 (1)

从地址	R/$\overline{W}$ 位	说明
0000 000	0	广播呼叫地址 (2)
0000 000	1	启动字节
0000 001	x	CBus 地址
0000 010	x	保留
0000 011	x	保留
0000 1xx	x	HS 模式主代码
1111 1xx	x	保留
1111 0xx	x	10 位从地址高字节 (3)

注 1： 上述地址位永远不会导致地址匹配，无论地址掩码的设置如何。

2： 仅当 GCEN = 1 时，才会应答地址。

3： 仅在 10 位寻址模式下才会与地址的高字节匹配。

寄存器 15-1： I2CxCON：I2Cx 控制寄存器

R/W-0	U-0	R/W-0	R/W-1, HC	R/W-0	R/W-0	R/W-0	R/W-0
I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0, HC	R/W-0, HC	R/W-0, HC	R/W-0, HC	R/W-0, HC
GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
bit 7							bit 0

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

bit 15 **I2CEN：**I2Cx 使能位
1 = 使能 I2Cx 模块并将 SDAx 和 SCLx 引脚配置为串行端口引脚
0 = 禁止 I2Cx 模块。所有 I^2C 引脚都由端口功能控制。

bit 14 **未实现：**读为 0

bit 13 **I2CSIDL：**在空闲模式停止位
1 = 当器件进入空闲模式后，模块停止工作
1 = 在空闲模式下模块继续工作

bit 12 **SCLREL：**SCLx 释放控制位 （作为 I^2C 从器件工作时）
1 = 释放 SCLx 时钟
0 = 保持 SCLx 时钟为低电平 （时钟延长）
如果 STREN = 1：
该位为 R/W （即可由软件写入 0 启动时钟延长，写入 1 释放时钟）。
在从器件开始发送时由硬件清零。
在从器件接收结束时由硬件清零。
如果 STREN = 0：
该位为 R/S （即只能由软件写入 1 释放时钟）。
在从器件开始发送时由硬件清零。

bit 11 **IPMIEN：**IPMI 使能位
1 = 使能 IPMI 支持模式；对所有地址做出应答
0 = 禁止 IPMI 模式

bit 10 **A10M：**10 位从器件地址位
1 = I2CxADD 是一个 10 位从器件地址
1 = I2CxADD 是一个 7 位从器件地址

bit 9 **DISSLW：**变化率控制禁止位
1 = 禁止变化率控制
0 = 使能变化率控制

bit 8 **SMEN：**SMBus 输入电平位
1 = 使能符合 SMBus 规范的 I/O 引脚门限值
0 = 禁止 SMBus 输入门限

bit 7 **GCEN：**广播呼叫使能位 （当作为 I^2C 从器件工作时）
1 = 允许在 I2CxRSR 接收到广播呼叫地址时产生中断
（已使能模块接收模式）
0 = 禁止广播呼叫地址

bit 6 **STREN：**SCLx 时钟延长使能位 （当作为 I^2C 从器件工作时）
与 SCLREL 位一起使用。
1 = 使能软件或接收时钟延长
1 = 禁止软件或接收时钟延长

寄存器 15-1：I2CxCON：I2Cx 控制寄存器（续）

bit 5 **ACKDT**：应答数据位 （当作为 I^2C 主器件工作时。适用于主器件接收操作。）
当软件启动应答序列时将发送的值。
1 = 在应答时发送 NACK
0 = 在应答时发送 ACK

bit 4 **ACKEN**：应答序列使能位
（当作为 I^2C 主器件工作时。适用于主器件接收操作。）
1 = 在 SDAx 和 SCLx 引脚启动应答序列，并发送 ACKDT 数据位。
在主器件应答序列结束时由硬件清零。
0 = 没有启动应答序列

bit 3 **RCEN**：接收使能位 （当作为 I^2C 主器件工作时）
1 = 使能 I^2C 接收模式
在主器件接收到数据字节的 8 位后由硬件清零。
0 = 没有启动接收序列

bit 2 **PEN**：停止条件使能位 （当作为 I^2C 主器件工作时）
1 = 在 SDAx 和 SCLx 引脚上产生停止条件
在主器件停止序列结束时由硬件清零。
0 = 没有启动停止条件

bit 1 **RSEN**：重复启动条件使能位 （当作为 I^2C 主器件工作时）
1 = 在 SDAx 和 SCLx 引脚产生重复启动条件
在主器件重复启动序列结束时由硬件清零。
0 = 没有发起重复启动条件

bit 0 **SEN**：启动条件使能位 （当作为 I^2C 主器件工作时）
1 = 在 SDAx 和 SCLx 引脚上产生启动条件
在主器件启动序列结束时由硬件清零。
0 = 没有发起启动条件

寄存器 15-2： I2CXSTAT：I2CX 状态寄存器

R-0,HSC	R-0,HSC	U-0	U-0	U-0	R/C-0,HS	R-0,HSC	R-0,HSC
ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10
bit 15							bit 8

R/C-0,HS	R/C-0,HS	R-0,HSC	R/C-0,HSC	R/C-0,HSC	R-0,HSC	R-0,HSC	R-0,HSC
IWCOL	I2COV	D/$\overline{A}$	P	S	R/$\overline{W}$	RBF	TBF
bit 7							bit 0

图注：	HS = 由硬件置 1 位	HSC = 硬件置 1/ 清零位	
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15 **ACKSTAT：**应答状态位
1 = 接收到来自从器件的 NACK
0 = 接收到来自从器件的 ACK
在从器件应答结束时由硬件置 1 或清零。

bit 14 **TRSTAT：**发送状态位
（当作为 I^2C 主器件工作时。适用于主器件发送操作。）
1 = 主器件正在进行发送 （8 位 + ACK）
0 = 主器件未进行发送
在主器件发送开始时由硬件置 1。
在从器件应答结束时由硬件清零。

bit 13-11 **未实现：**读为 0

bit 10 **BCL：**主器件总线冲突检测位
1 = 在主器件工作期间检测到了总线冲突
0 = 未发生冲突
在检测到总线冲突时由硬件置 1。

bit 9 **GCSTAT：**广播呼叫状态位
1 = 接收到了广播呼叫地址
0 = 未接收到广播呼叫地址
当地址与广播呼叫地址匹配时由硬件置 1。
在检测到停止条件时由硬件清零。

bit 8 **ADD10：**10 位地址状态位
1 = 10 位地址匹配
0 = 10 位地址不匹配
当与 10 位地址的第二个字节匹配时由硬件置 1。
在检测到停止条件时由硬件清零。

bit 7 **IWCOL：**写冲突检测位
1 = 由于 I^2C 总线忙，尝试写 I2CxTRN 寄存器的操作失败
0 = 未发生冲突
当在总线忙的情况下写 I2CxTRN 时由硬件置 1 （由软件清零）。

bit 6 **I2COV：**接收溢出标志位
1 = 当 I2CxRCV 寄存器仍然保存原先的字节时又接收了新字节
0 = 未发生溢出
尝试将数据从 I2CxRSR 传输到 I2CxRCV 时由硬件置 1 （由软件清零）。

bit 5 **D/$\overline{A}$：**数据 / $\overline{地址}$位 （当作为 I^2C 从器件工作时）
1 = 表示上次接收的字节是数据
0 = 表示上次接收的字节是器件地址
器件地址匹配时由硬件清零。
通过写 I2CxTRN 或接收从器件字节由硬件置 1。

寄存器 15-2： I2CXSTAT：I2CX 状态寄存器 （续）

bit 4 **P：**停止位

1 = 上一次检测到了停止位
0 = 上一次没有检测到停止位
当检测到启动、重复启动或停止条件时由硬件置 1 或清零。

bit 3 **S：**启动位

1 = 上一次检测到了启动 （或重复启动）位
0 = 上一次未检测到启动位
当检测到启动、重复启动或停止条件时由硬件置 1 或清零。

bit 2 **R/$\overline{W}$：**读 / $\overline{写}$信息位 （当作为 I^2C 从器件工作时）

1 = 读——表示从器件输出数据
0 = 写——表示从器件接收数据
当接收到 I^2C 器件地址字节后由硬件置 1 或清零。

bit 1 **RBF：**接收缓冲器满状态位

1 = 接收完成， I2CxRCV 满
0 = 接收没有完成， I2CxRCV 空
当接收到的字节写入 I2CxRCV 后由硬件置 1。
当软件读 I2CxRCV 时由硬件清零。

bit 0 **TBF：**发送缓冲器满状态位

1 = 正在发送， I2CxTRN 满
0 = 发送完成， I2CxTRN 空
当软件写 I2CxTRN 时由硬件置 1。
当数据发送完成时由硬件清零。

寄存器 15-3：　　I2CxMSK：I2Cx 从动模式地址屏蔽寄存器

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	AMSK9	AMSK8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
AMSK7	AMSK6	AMSK5	AMSK4	AMSK3	AMSK2	AMSK1	AMSK0
bit 7							bit 0

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

bit 15-10 **未实现：**读为 0

bit 9-0 **AMSK9:AMSK0：**地址位 x 屏蔽选择位
1 = 使能屏蔽输入报文地址的 bit x；不要求该位匹配
0 = 禁止屏蔽 bit x；要求该位匹配

16.0 通用异步收发器 （UART）

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 21 章“UART”**（DS39708A_CN）。

通用异步收发器（Universal Asynchronous Receiver Transmitter， UART）模块是 PIC24F 系列器件配备的串行 I/O 模块之一。UART 是可以和外设（如个人电脑、LIN、RS-232 和 RS-485 接口）通信的全双工异步系统。该模块通过使用 UxCTS 和 UxRTS 引脚支持硬件流量控制，它还包含一个 IrDA 编解码器。

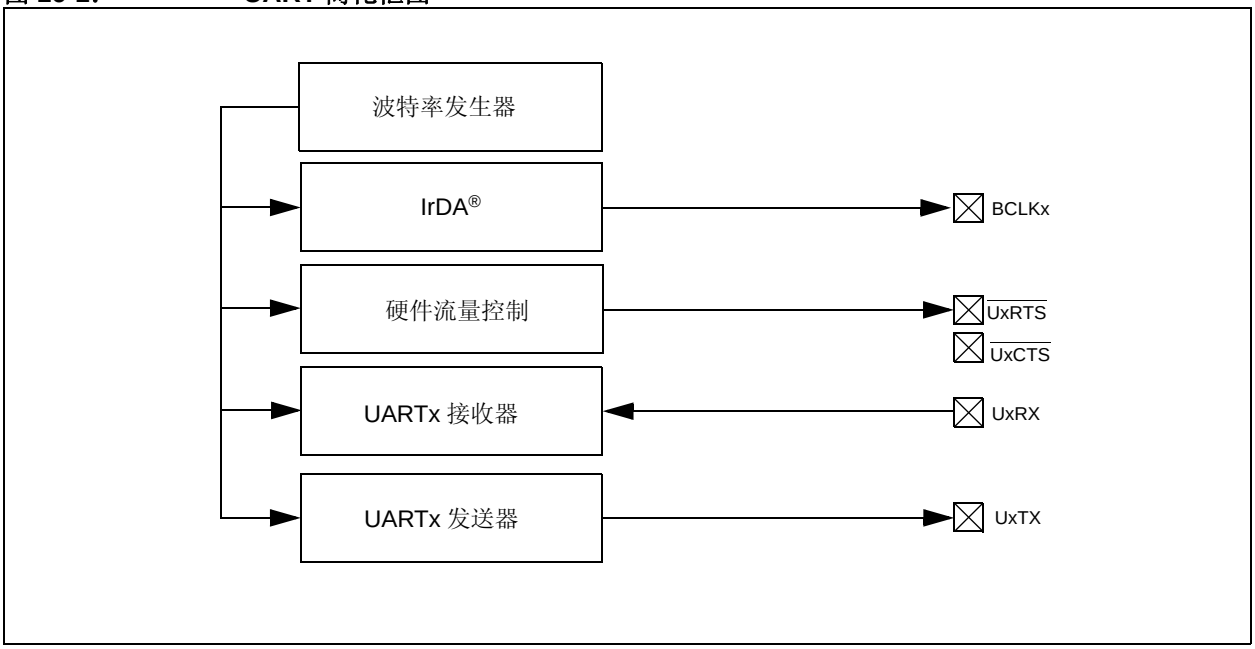
UART 模块的主要特性有：

- 通过 UxTX 和 UxRX 引脚可进行全双工 8 位或 9 位数据传输
- 偶、奇或无奇偶校验选项 （对于 8 位数据）
- 一个或两个停止位
- 使用 UxCTS 和 UxRTS 引脚可实现硬件流量控制
- 全集成的、具有 16 位预分频器的波特率发生器
- 在 16 MIPS 下波特率范围为 1 Mbps 到 15 bps
- 4 级 FIFO 发送数据缓冲器
- 4 级 FIFO 接收数据缓冲器
- 奇偶，帧和缓冲器溢出错误检测
- 支持带地址检测的 9 位模式（第 9 位 = 1）
- 发送和接收中断
- 支持诊断的环回模式
- 支持同步和间隔字符
- 支持波特率自动检测
- IrDA 编解码器逻辑
- 支持 IrDA 的 16x 波特率时钟输出

图 16-1 显示了 UART 的简化框图。UART 模块由下列主要硬件组成：

- 波特率发生器
- 异步发送器
- 异步接收器

图 16-1： UART 简化框图



16.1 UART 波特率发生器 （BRG）

UART 模块包含一个专用 16 位波特率发生器。BRGx 寄存器控制独立运行的 16 位定时器的周期。公式 16-1 是在 BRGH = 0 时计算波特率的公式。

公式 16-1： BRGH = 0 时的 UART 波特率 (1)

$$\text{波特率} = \frac{FCY}{16 \cdot (BRGx + 1)}$$

$$BRGx = \frac{FCY}{16 \cdot \text{波特率}} - 1$$

注 1： 上述公式基于 FCY = FOSC/2，且打盹模式和 PLL 是被禁止的。

例 16-1 显示了下列条件下波特率误差的计算方法：

- FCY = 4 MHz
- 目标波特率 = 9600

波特率（BRGH = 0）的可能值最大是 FCY/16（BRGx = 0），最小是 FCY/(16 * 65536)。

公式 16-2 给出了 BRGH = 1 时计算波特率的公式。

公式 16-2： BRGH = 1 时的 UART 波特率 (1)

$$\text{波特率} = \frac{FCY}{4 \cdot (BRGx + 1)}$$

$$BRGx = \frac{Fcy}{4 \cdot \text{波特率}} - 1$$

注 1： 上述公式基于 FCY = FOSC/2，且打盹模式和 PLL 是被禁止的。

波特率（BRGH = 1）的可能值最大是 FCY/4（BRGx = 0），最小是 FCY/(4 * 65536)。

向 BRGx 寄存器中写新值会导致 BRG 定时器复位（清零）。这保证了在 BRG 产生新的波特率之前不需要等待定时器溢出。

例 16-1： 波特率误差计算公式 （BRGH = 0） (1)

目标波特率 = FCY/(16 (BRGx + 1))

计算 BRGx 值：

BRGx = ((FCY/ 目标波特率)/16) – 1
BRGx = ((4000000/9600)/16)–1
BRGx = 25

计算波特率 = 4000000/(16 (25 + 1))
= 9615

误差 = $\frac{(\text{计算波特率} - \text{目标波特率})}{\text{目标波特率}}$
= (9615 – 9600)/9600
= 0.16%

注 1： 上述公式基于 FCY = FOSC/2 ； 且打盹模式和 PLL 是被禁止的。

16.2 按 8 位数据模式发送

1. 设置 UART：
   a) 将相应值写入数据、奇偶校验和停止位。
   b) 将相应的波特率值写入 BRGx 寄存器。
   c) 设置发送和接收中断允许位和优先级位。
2. 使能 UART。
3. 将 UTXEN 位置 1（产生发送中断）。
4. 将数据字节写入TXxREG字的低字节。该值会被立即送入发送移位寄存器（TSR），在波特率时钟的下一个上升沿移出串行比特流。
5. 另外，数据字节也可以在 UTXEN = 0 时发送，然后用户将 UTXEN 位置 1。由于波特率时钟是从清零状态开始的，所以该行为能立即启动发送串行比特流。
6. 根据中断控制位 UTXISELx 的设置产生发送中断。

16.3 按 9 位数据模式发送

1. 设置 UART（见**第 16.2 节“按 8 位数据模式发送”**）。
2. 使能 UART。
3. 将 UTXEN 位置 1（产生发送中断）。
4. 只将 16 位值写入 UxTXREG。
5. 将一个字写入 UxTXREG 触发将 9 位数据送入 TSR的操作。串行比特流在波特率时钟的第一个上升沿开始移出。
6. 根据控制位 UTXISELx 的设置产生发送中断。

16.4 间隔和同步发送序列

以下序列会发送一个报文帧头，包括一个间隔字符和其后的自动波特率同步字节。

1. 将 UART 配置为所需的模式。
2. 将UTXEN和UTXBRK位置1——设置间隔字符。
3. 将虚拟字符装入 UxTXREG，启动发送（该值会被忽略）。
4. 将 55h 写入 UxTXREG——同步字符被装入发送 FIFO。
5. 间隔字符发送完毕后，硬件会将 UTXBRK 位复位。开始发送同步字符。

16.5 按 8 位或 9 位数据模式接收

1. 设置 UART（见**第 16.2 节“按 8 位数据模式发送”**）。
2. 使能 UART。
3. 根据每个中断控制位 URXISELx 的设置，当接收一个或多个数据字符时会产生一个接收中断。
4. 读 OERR 位判断是否发生溢出错误。OERR 位必须用软件复位。
5. 读 UxRXREG。

读 UxRXREG 字符的同时会将下一字符移动到接收 FIFO 的顶部，包括一组新的 PERR 和 FERR 值。

16.6 $\overline{\text{UxCTS}}$ 和 $\overline{\text{UxRTS}}$ 控制引脚操作

$\overline{\text{UxCTS}}$（UARTx Clear to Send）和 $\overline{\text{UxRTS}}$（UARTx Request to Send）是与 UART 模块相关的两个硬件控制引脚。这两个引脚允许 UART 工作在单工和流量控制模式下，用于控制数据终端设备（Data Terminal Equipment，DTE）间的数据发送和接收。UxMODE 寄存器内的 UEN<1:0> 位用于配置这两个引脚。

16.7 红外支持

UART 模块支持两种红外 UART 模式：一种可输出 IrDA 时钟，它支持外部 IrDA 编码和解码器件（支持传统模块），另一种完全集成了 IrDA 编解码器。

16.8 外部 IrDA 支持——IrDA 时钟输出

要支持外部编码和解码器件，可以将 BCLKx 引脚（同 UxRTS 引脚）配置产生 16x 波特率时钟。在 UEN<1:0> = 11 且使能了 UART 模块时，BCLKx 引脚输出 16x 波特率时钟。可用于支持 IrDA 编解码芯片。

16.9 内置 IrDA 编解码器

UART 完全实现了 IrDA 编解码器，作为 UART 模块的组件。使用 IREN 位（UxMODE<12>）可以使能内置 IrDA 编解码器。当使能时（IREN = 1），接收引脚（UxRX）作为红外接收器的输入端。发送引脚（UxTX）作为红外发送器的输出端。

寄存器 16-1： UxMODE：UARTx 模式寄存器

R/W-0	U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0
bit 15							bit 8

R/W-0,HC	R/W-0	R/W-0,HC	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL
bit 7							bit 0

图注：	HC = 硬件清零位		
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15 **UARTEN**：UARTx 使能位
1 = 使能 UARTx； UARTx 根据 UEN<1:0> 的定义控制所有 UARTx 引脚
0 = 禁止 UARTx； PORT 锁存器控制所有 UARTx 引脚； UARTx 功耗最小

bit 14 **未实现**：读为 0

bit 13 **USIDL**：空闲模式停止位
1 = 当器件进入空闲模式时，模块停止工作
0 = 在空闲模式下模块继续工作

bit 12 **IREN**：IrDA 编解码器使能位 **(1)**
1 = 使能 IrDA 编解码器
0 = 禁止 IrDA 编解码器

bit 11 **RTSMD**：$\overline{\text{UxRTS}}$ 引脚的模式选择位
1 = $\overline{\text{UxRTS}}$ 引脚工作在单工模式下
0 = UxRTS 引脚工作在流量控制模式下

bit 10 **未实现**：读为 0

bit 9-8 **UEN1:UEN0**：UARTx 使能位
11 = 使能并使用 UxTX、 UxRX 和 BCLKx 引脚； PORT 锁存器控制 $\overline{\text{UxCTS}}$ 引脚
10 = 使能并使用 UxTX、 UxRX、 $\overline{\text{UxCTS}}$ 和 UxRTS 引脚
01 = 使能并使用 UxTX、 UxRX 和 UxRTS 引脚； PORT 锁存器控制 $\overline{\text{UxCTS}}$ 引脚
00 = 使能并使用 UxTX 和 UxRX 引脚； PORT 锁存器控制 UxCTS 和 UxRTS/BCLKx 引脚

bit 7 **WAKE**：在休眠模式下检测到启动位时唤醒使能位
1 = UARTx 会继续采样 UxRX 引脚；在下降沿产生中断，在下一个上升沿该位由硬件清零
0 = 不使能唤醒

bit 6 **LPBACK**：UARTx 环回模式选择位
1 = 使能环回模式
0 = 禁止环回模式

bit 5 **ABAUD**：自动波特率使能位
1 = 在下一个字符上使能波特率测量——要求接收同步字段 （55h）； 完成后由硬件清零
0 = 禁止波特率测量或已完成

bit 4 **RXINV**：接收极性翻转位
1 = UxRX 空闲状态为 0
0 = UxRX 空闲状态为 1

bit 3 **BRGH**：高波特率使能位
1 = 每个位周期 BRG 产生 4 个时钟脉冲 （4x 波特率时钟，高速模式）
0 = 每个位周期 BRG 产生 16 个时钟脉冲 （16x 波特率时钟，标准模式）

注 1： 该功能只用于 16x BRG 模式 （BRGH = 0）。

寄存器 16-1： UxMODE：UARTx 模式寄存器 （续）

bit 2-1 **PDSEL1:PDSEL0**：奇偶校验和数据选择位
11 = 9 位数据，无奇偶校验
10 = 8 位数据，奇校验
01 = 8 位数据，偶校验
00 = 8 位数据，无奇偶校验

bit 0 **STSEL**：停止位选择位
1 = 两个停止位
0 = 一个停止位

注 1： 该功能只用于 16x BRG 模式 （BRGH = 0）。

寄存器 16-2： UxSTA：UARTx 状态和控制寄存器

R/W-0	R/W-0	R/W-0	U-0	R/W-0 HC	R/W-0	R-0	R-1
UTXISEL1	TXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/C-0	R-0
URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7							bit 0

图注：	C = 可清零位	HC = 硬件清零位	
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15,13 **UTXISEL1:UTXISEL0：**发送中断模式选择位
11 = 保留；不使用
10 = 当一个字符被送入发送移位寄存器导致发送缓冲器为空时，产生中断
01 = 当最后一个字符移出发送移位寄存器时产生中断；发送操作全部完成
00 = 当一个字符被送入发送移位寄存器时产生中断 （表示在发送缓冲器中至少有一个字符的剩余空间）

bit 14 **TXINV：**发送极性翻转位
IREN = 0：
1 = TX 空闲状态为 0
0 = TX 空闲状态为 1
IREN = 1：
1 = IrDA 编码的 TX 空闲状态为 1
0 = IrDA 编码的 TX 空闲状态为 0

bit 12 **未实现：**读为 0

bit 11 **UTXBRK：**发送间隔位
1 = 在下一次发送时发送同步间隔字符——起始位，后面紧跟 12 个 0，最后是停止位；发送完成后由硬件清零
0 = 禁止同步间隔字符发送或发送已完成

bit 10 **UTXEN：**发送使能位
1 = 使能发送，由 UARTx 控制 UxTX 引脚
0 = 禁止发送，中止所有后续发送，缓冲器复位。由 PORT 控制 UxTX 引脚。

bit 9 **UTXBF：**发送缓冲器满状态位 （只读）
1 = 发送缓冲器满
0 = 发送缓冲器未满，至少还可以写入一个字符

bit 8 **TRMT：**发送移位寄存器空位 （只读）
1 = 发送移位寄存器为空，同时发送缓冲器为空 （上一个发送已经完成）
0 = 发送移位寄存器非空，发送正在进行或等待发送

bit 7-6 **URXISEL1:URXISEL0：**接收中断模式选择位
11 = 在 RSR 传送使接收缓冲器满时产生中断 （即接收了 4 个数据字符）
10 = 在 RSR 传送使接收缓冲器 3/4 满时产生中断 （即接收了 3 个数据字符）
0x = 当接收缓冲器接收到从 RSR 传来的任何字符时产生中断。接收缓冲器有一个或多个字符。

bit 5 **ADDEN：**地址字符检测位 （接收数据的 bit 8 = 1）
1 = 使能地址检测模式。如果没有选择 9 位模式，该设置不起作用。
0 = 禁止地址检测模式

寄存器 16-2： UxSTA：UARTx 状态和控制寄存器 （续）

bit 4 **RIDLE：**接收器空闲位 （只读）
1 = 接收器空闲
0 = 接收器处于工作状态

bit 3 **PERR：**奇偶校验错误状态位 （只读）
1 = 当前字符 （接收 FIFO 顶端字符）中发现奇偶校验错误
0 = 没有检测到奇偶校验错误

bit 2 **FERR：**帧错误状态位 （只读）
1 = 当前字符 （接收 FIFO 顶端字符）中发现帧错误
0 = 没有检测到帧错误

bit 1 **OERR：**接收缓冲器溢出错误状态位 （清零 / 只读）
1 = 接收缓冲器溢出
0 = 接收缓冲器未溢出 （将先前置 1 的 OERR 位清零 （1 → 0）会将接收缓冲器和 RSR 复位到空状态）

bit 0 **URXDA：**接收缓冲器中可用数据位 （只读）
1 = 接收缓冲器中有数据，至少还有一个字符可读
0 = 接收缓冲器为空

注：

17.0 并行主控端口（PMP）

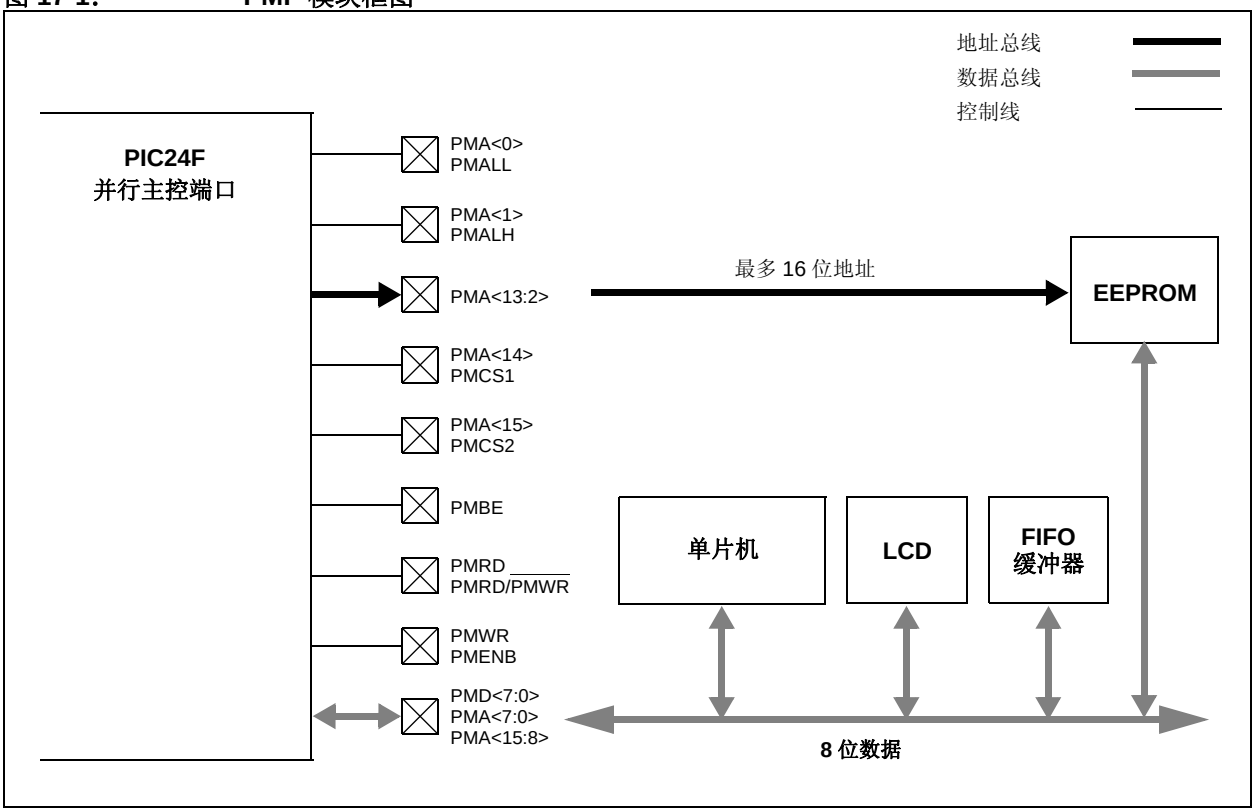
注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 13 章“并行主控端口（PMP）”**（DS39713A_CN）。

并行主控端口（Parallel Master Port，PMP）模块是专为与多种并行器件，如通信外设、LCD、外部存储器和单片机通信而设计的。由于并行外设的接口差异很大，因此 PMP 具有很强的可配置能力。

PMP 模块的主要特性包括：

- 最多 16 条可编程地址线
- 最多两条片选信号线
- 可编程选通选项
  - 单独的读和写选通
  - 带有使能端的读 / $\overline{\text{写}}$选通
- 地址自动递增 / 自动递减
- 可编程地址 / 数据复用
- 可编程的控制信号的极性
- 支持传统的并行从动端口
- 支持增强型并行从动端口
  - 地址支持
  - 4 字节深自动递增缓冲器
- 可编程等待状态
- 可选择的输入电压电平

图 17-1： PMP 模块框图



寄存器 17-1：　　PMCON：并行端口控制寄存器

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PMPEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN
bit 15							bit 8

R/W-0	R/W-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0	R/W-0	R/W-0
CSF1	CSF0	ALP	CS2P	CS1P	BEP	WRSP	RDSP
bit 7							bit 0

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

bit 15　**PMPEN：**并行主控端口使能位
1 = 使能 PMP
0 = 禁止 PMP，不执行任何片外访问

bit 14　**未实现：**读为 0

bit 13　**PSIDL：**空闲模式停止位
1 = 当器件进入空闲模式后，模块停止工作
0 = 在空闲模式下模块继续工作

bit 12-11　**ADRMUX1:ADRMUX0：**地址 / 数据复用选择位
11 = 保留
10 = 所有 16 位地址与 PMD<7:0> 引脚复用
01 = 地址的低 8 位与 PMD<7:0> 引脚复用，高 8 位与 PMA<15:8> 引脚复用
00 = 地址和数据使用独立的引脚

bit 10　**PTBEEN：**字节使能端口使能位 （16 位主控模式）
1 = 使能 PMBE 端口
0 = 禁止 PMBE 端口

bit 9　**PTWREN：**写使能选通端口使能位
1 = 使能 PMWR/PMENB 端口
0 = 禁止 PMWR/PMENB 端口

bit 8　**PTRDEN：**读 / 写选通端口使能位
1 = 使能 PMRD/PMWR 端口
0 = 禁止 PMRD/PMWR 端口

bit 7-6　**CSF1:CSF0：**片选功能位
11 = 保留
10 = PMCS1 和 PMCS2 用作片选功能
01 = PMCS2 用作片选功能， PMCS1 用作地址的 bit 14
00 = PMCS1 和 PMCS2 分别用作地址的 bit 15 和 bit 14

bit 5　**ALP：**地址锁存极性位 (1)
1 = 高电平有效 （PMALL 和 PMALH）
0 = 低电平有效 （PMALL 和 PMALH）

bit 4　**CS2P：**片选 2 极性位 (1)
1 = 高电平有效 （PMCS2）
0 = 低电平有效 （PMCS2）

bit 3　**CS1P：**片选 1 极性位 (1)
1 = 高电平有效 （PMCS1/PMCS）
0 = 低电平有效 （PMCS1/PMCS）

注 1：　当对应的引脚用作地址线时，这些位无效。

寄存器 17-1：　　PMCON：并行端口控制寄存器 （续）

bit 2　**BEP：**字节使能极性位
1 = 字节使能高电平有效 （PMBE）
0 = 字节使能低电平有效 （PMBE）

bit 1　**WRSP：**写选通极性位
对于从动模式或主控模式 2 （PMMODE<9:8> = 00,01,10）：
1 = 写选通高电平有效 （PMWR）
0 = 写选通低电平有效 （PMWR）
对于主控模式 1 （PMMODE<9:8> = 11）：
1 = 使能选通高电平有效 （PMENB）
0 = 使能选通低电平有效 （PMENB）

bit 0　**RDSP：**读选通极性位
对于从动模式或主控模式 2 （PMMODE<9:8> = 00,01,10）：
1 = 读选通高电平有效 （PMRD）
0 = 读选通低电平有效 （PMRD）
对于主控模式 1 （PMMODE<9:8> = 11）：
1 = 读 / 写选通高电平有效 （PMRD/PMWR）
0 = 读 / 写选通低电平有效 （PMRD/PMWR）

注　1：　当对应的引脚用作地址线时，这些位无效。

寄存器 17-2： PMMODE：并行端口模式寄存器

R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAITB1(1)	WAITB0(1)	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1(1)	WAITE0(1)
bit 7							bit 0

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

bit 15 **BUSY**：忙位 （仅用于主控模式）
1 = 端口忙 （当处理器暂停时无用）
0 = 端口不忙

bit 14-13 **IRQM1:IRQM0**：中断请求模式位
11 = 当读取读缓冲器 3 或写入写缓冲器 3 时产生中断 （缓冲的 PSP 模式）
或当 PMA<1:0> = 11 时执行读或写操作时产生中断 （仅可寻址 PSP 模式）
10 = 不产生中断，处理器暂停
01 = 当读 / 写周期结束时产生中断
00 = 不产生中断

bit 12-11 **INCM1:INCM0**：递增模式位
11 = PSP 读和写缓冲器自动递增 （传统 PSP 模式）
10 = 每个读 / 写周期都将 ADDR<15,13:0> 减 1
01 = 每个读 / 写周期都将 ADDR<15,13:0> 加 1
00 = 地址不发生自动增减

bit 10 **MODE16**：8/16 位模式位
1 = 16 位模式：数据寄存器为 16 位，读或写数据寄存器需要两次 8 位数据的传输
0 = 8 位模式：数据寄存器为 8 位，读或写数据寄存器需要一次 8 位数据的传输

bit 9-8 **MODE1:MODE0**：并行端口模式选择位
11 = 主控模式 1 （PMCSx、 PMRD/PMWR、 PMENB、 PMBE、 PMA<x:0> 和 PMD<7:0>）
10 = 主控模式 2 （PMCSx、 PMRD、 PMWR、 PMBE、 PMA<x:0> 和 PMD<7:0>）
01 = 增强型 PSP，控制信号 （PMRD、 PMWR、 PMCS、 PMD<7:0> 和 PMA<1:0>）
00 = 传统并行从动端口，控制信号 （PMRD、 PMWR、 PMCS 和 PMD<7:0>）

bit 7-6 **WAITB1:WAITB0**：从数据建立到执行读 / 写的等待状态配置位 (1)
11 = 数据等待时间为 4 个 TCY；地址复用时间为 4 个 TCY
10 = 数据等待时间为 3 个 TCY；地址复用时间为 3 个 TCY
01 = 数据等待时间为 2 个 TCY；地址复用时间为 2 个 TCY
00 = 数据等待时间为 1 个 TCY；地址复用时间为 1 个 TCY

bit 5-2 **WAITM3:WAITM0**：从执行读操作到字节使能选通的等待状态配置位
1111 = 额外等待 15 个 TCY
...
0001 = 额外等待 1 个 TCY
0000 = 无额外等待周期 （操作被强制在 1 个 TCY 内完成）

bit 1-0 **WAITE1:WAITE0**：选通后数据保持的等待状态配置位 (1)
11 = 等待 4 个 TCY
10 = 等待 3 个 TCY
01 = 等待 2 个 TCY
00 = 等待 1 个 TCY

注 1： 只要 WAITM3:WAITM0 = 0000， WAITB 和 WAITE 位就与端口操作无关。

寄存器 17-3： PMADDR：并行端口地址寄存器 (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
CS2	CS1	ADDR<13:8>					
bit 15							bit 8

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

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

bit 15 **CS2**：片选 2 位
1 = 片选 2 有效
0 = 片选 2 无效（引脚用作 PMA<15>）

bit 14 **CS1**：片选 1 位
1 = 片选 1 有效
0 = 片选 1 无效（引脚用作 PMA<14>）

bit 13-0 **ADDR13:ADDR0**：并行端口目标地址位

注 1： PMADDR 和 PMDOUT1 寄存器共享同一物理地址。 PMDOUT1 功能仅在从动模式下可用，而 PMADDR 仅用于主控模式。

寄存器 17-4： PMAEN：并行端口使能寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTEN15	PTEN14	PTEN13	PTEN12	PTEN11	PTEN10	PTEN9	PTEN8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTEN7	PTEN6	PTEN5	PTEN4	PTEN3	PTEN2	PTEN1	PTEN0
bit 7							bit 0

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

bit 15-14 **PTEN15:PTEN14**：PMCSx 选通使能位
1 = PMA15 和 PMA14 用作 PMA<15:14> 或分别用作 PMCS2 和 PMCS1
0 = PMA15 和 PMA14 用作端口 I/O

bit 13-2 **PTEN13:PTEN2**：PMP 地址端口使能位
1 = PMA<13:2> 用作 PMP 地址线
0 = PMA<13:2> 用作端口 I/O

bit 1-0 **PTEN1:PTEN0**：PMALH/PMALL 选通使能位
1 = PMA1 和 PMA0 用作 PMA<1:0> 或分别用作 PMALH 和 PMALL
0 = PMA1 和 PMA0 用作端口 I/O

寄存器 17-5：　PMSTAT：并行端口状态寄存器

R-0	R/W-0 HS	U-0	U-0	R-0	R-0	R-0	R-0
IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F
bit 15							bit 8

R-1	R/W-0 HS	U-0	U-0	R-1	R-1	R-1	R-1
OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E
bit 7							bit 0

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

bit 15 **IBF：**输入缓冲器满状态位
1 = 所有可写的输入缓冲寄存器为满
0 = 某些或全部可写的输入缓冲寄存器为空

bit 14 **IBOV：**输入缓冲器溢出状态位
1 = 企图写入一个已满的输入字节寄存器 （必须由软件清零）
0 = 没有发生溢出

bit 13-12 **未实现：**读为 0

bit 11-8 **IB3F:IB0F：**输入缓冲器 n 满状态位
1 = 输入缓冲器中包含未被读取的数据 （读缓冲器将清零该位）
0 = 输入缓冲器中没有任何未读取的数据

bit 7 **OBE：**输出缓冲器空状态位
1 = 所有可读的输出缓冲寄存器为空
0 = 某些或全部可读的输出缓冲寄存器为满

bit 6 **OBUF：**输出缓冲器下溢状态位
1 = 读取了一个为空的输出字节寄存器 （必须由软件清零）
0 = 没有发生下溢

bit 5-4 **未实现：**读为 0

bit 3-0 **OB3E:OB0E：**输出缓冲器 n 空状态位
1 = 输出缓冲器为空 （将数据写入该缓冲器会将该位清零）
0 = 输出缓冲器中包含未被发送的数据

寄存器 17-6：　PADCFG1：填充配置控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	RTSECSEL(1)	PMPTTL(2)
bit 7							bit 0

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

bit 15-2　**未实现：**读为 0

bit 1　**RTSECSEL：**RTCC 秒时钟输出选择位 (1)
1 = 选择从 RTCC 引脚输出 RTCC 秒时钟
0 = 选择从 RTCC 引脚输出 RTCC 闹钟脉冲

bit 0　**PMPTTL：**PMP 模块 TTL 输入缓冲器选择位 (2)
1 = PMP 模块使用 TTL 输入缓冲器
0 = PMP 模块使用施密特输入缓冲器

注　1：　要使能实际 RTCC 输出，需要将 RTCCFG （RTCOE）位置 1。
2：　有关受影响的 PMP 输入，请参见表 1-2。

图 17-2：　　传统并行从动端口示例

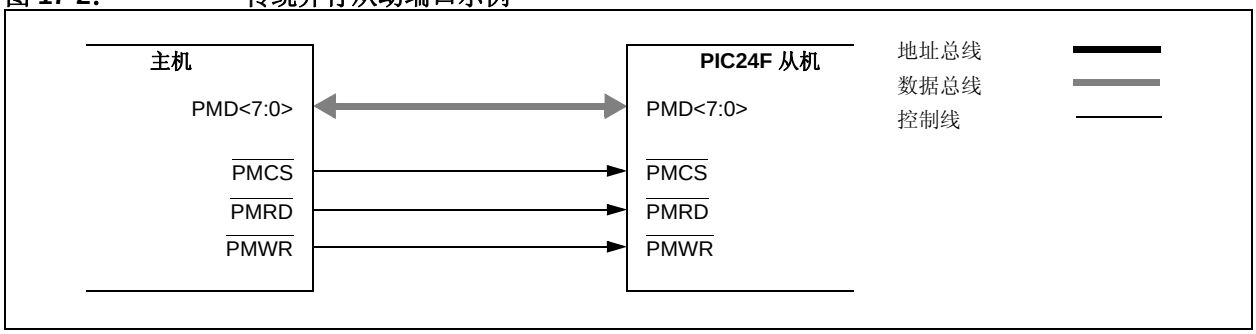


图 17-3：　　可寻址并行从动端口示例

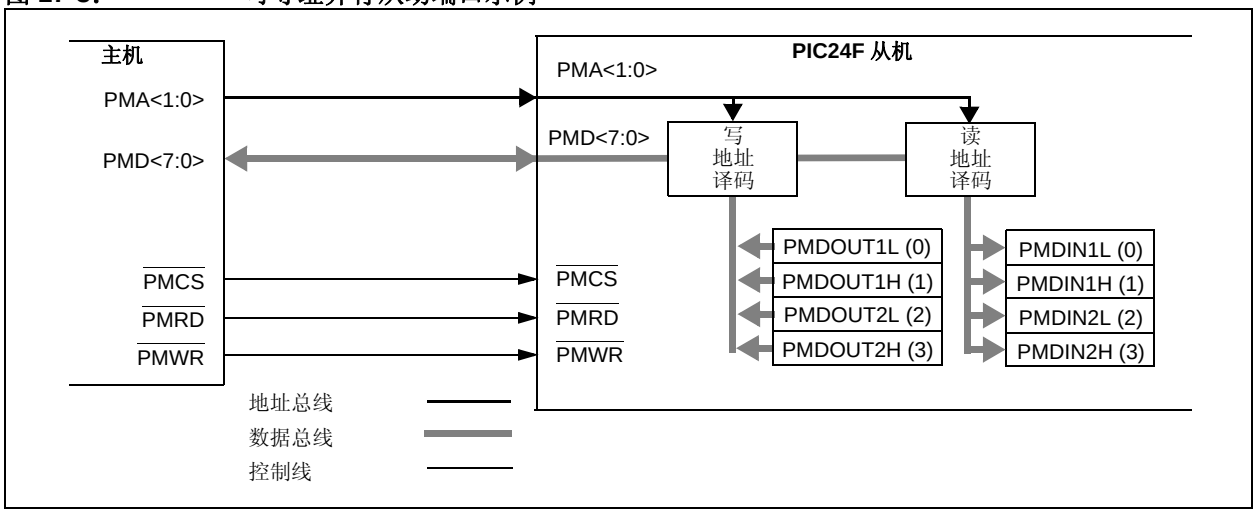


表 17-1：　　从动模式地址解析方案

PMA<1:0>	输出寄存器 （缓冲器）	输入寄存器 （缓冲器）
00	PMDOUT1<7:0> （0）	PMDIN1<7:0> （0）
01	PMDOUT1<15:8> （1）	PMDIN1<15:8> （1）
10	PMDOUT2<7:0> （2）	PMDIN2<7:0> （2）
11	PMDOUT2<15:8> （3）	PMDIN2<15:8> （3）

图 17-4：　　主控模式下的解复用寻址 （单独的读和写选通、两个片选信号）

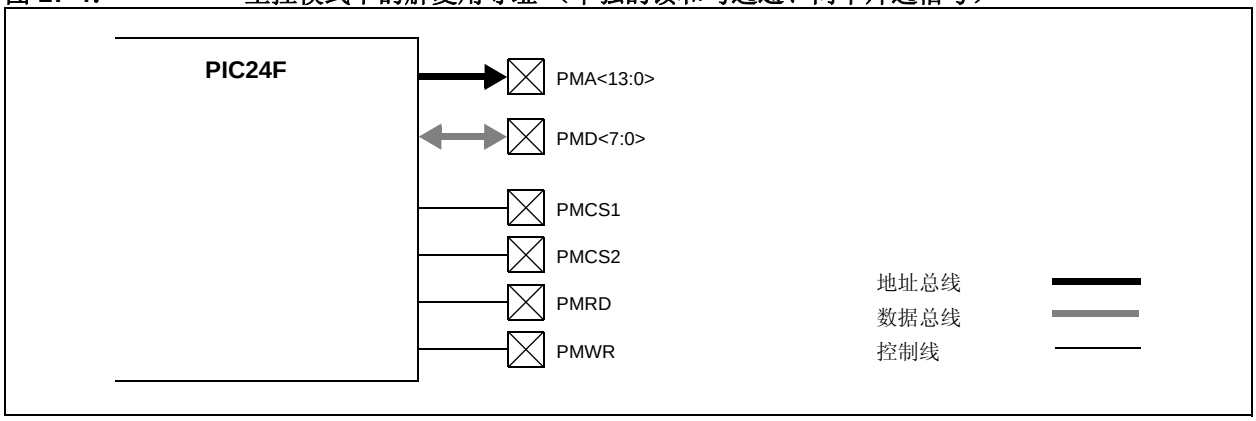


图 17-5：　　主控模式下的部分复用寻址 （单独的读和写选通、两个片选信号）

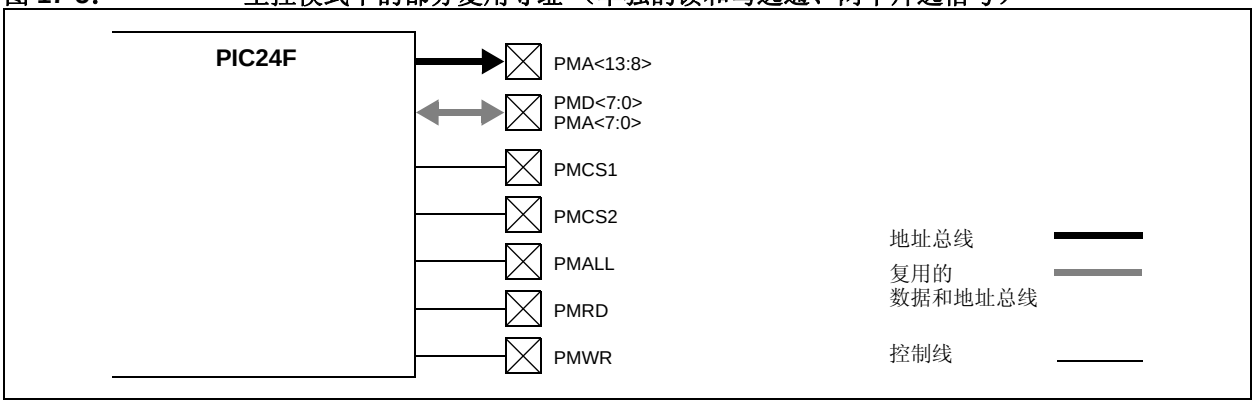


图 17-6：　　主控模式下的完全复用寻址 （单独的读和写选通、两个片选信号）

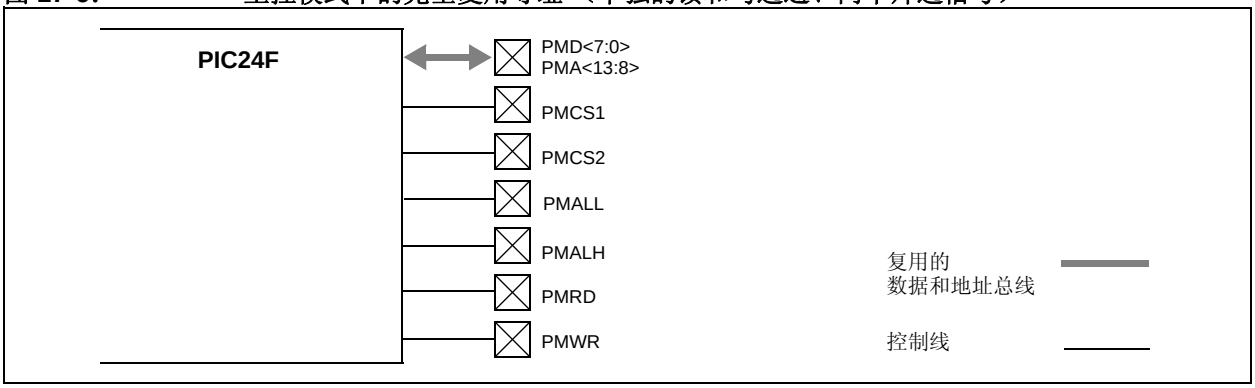


图 17-7：　　复用寻址应用示例

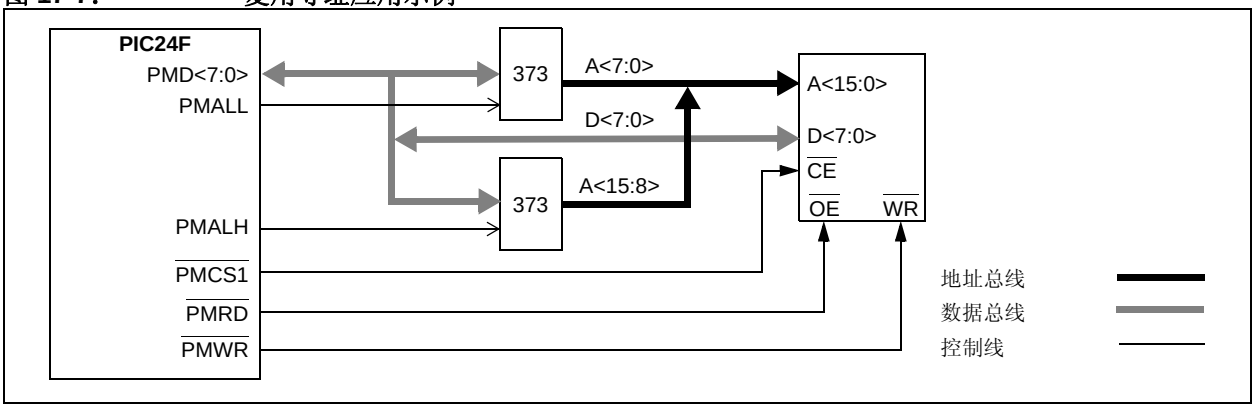


图 17-8：　　部分复用寻址应用示例

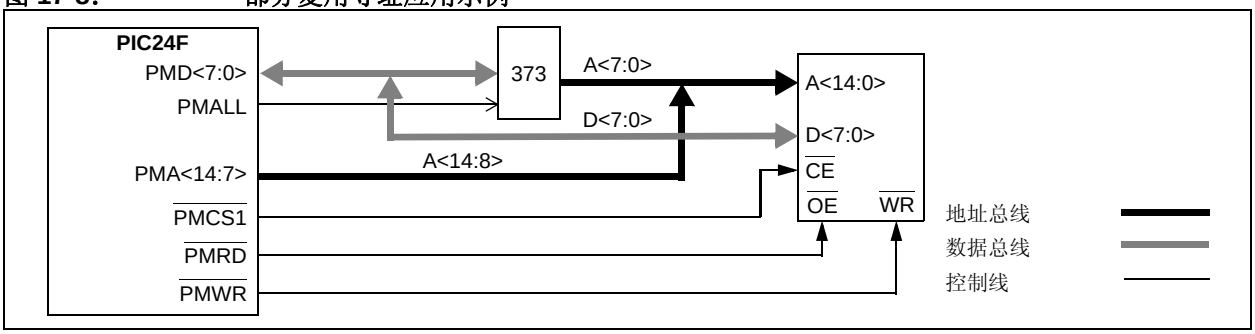


图 17-9：　　8 位地址和数据复用的应用示例

图 17-10：　　并行 EEPROM 示例 （最多 15 位地址 8 位数据）

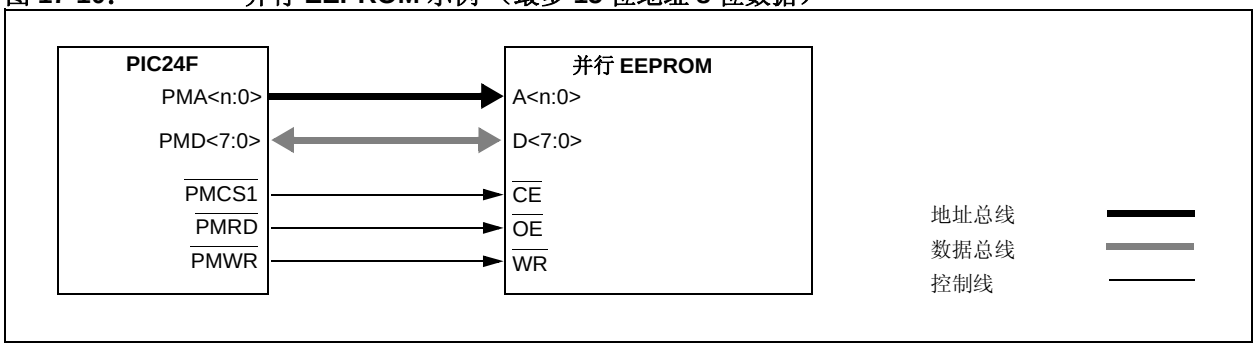


图 17-11：　　并行 EEPROM 示例 （最多 15 位地址 16 位数据）

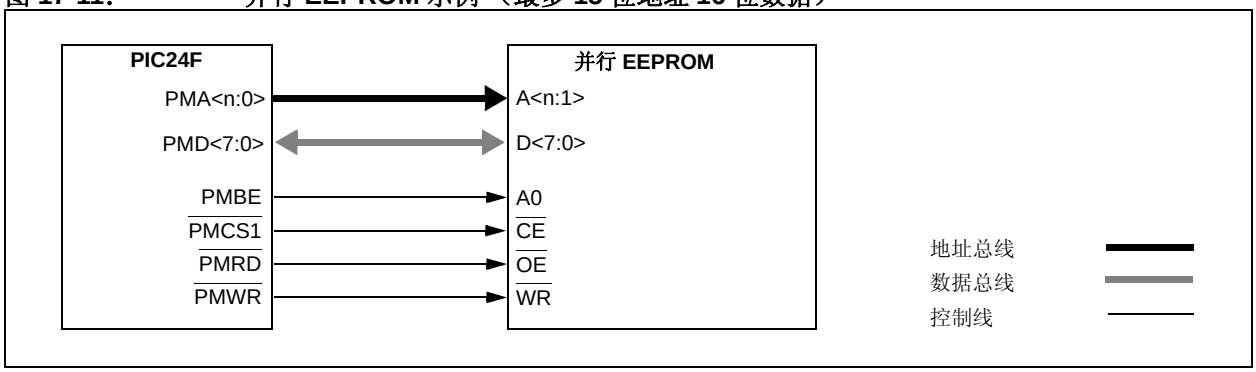
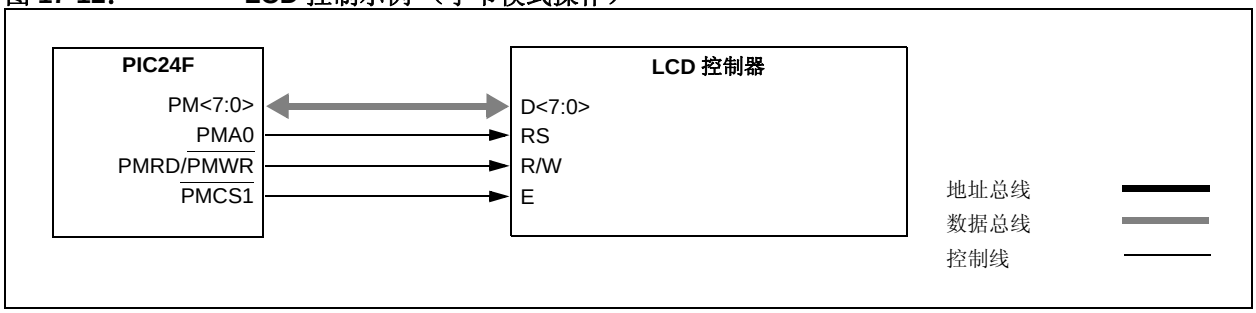


图 17-12：　　LCD 控制示例 （字节模式操作）



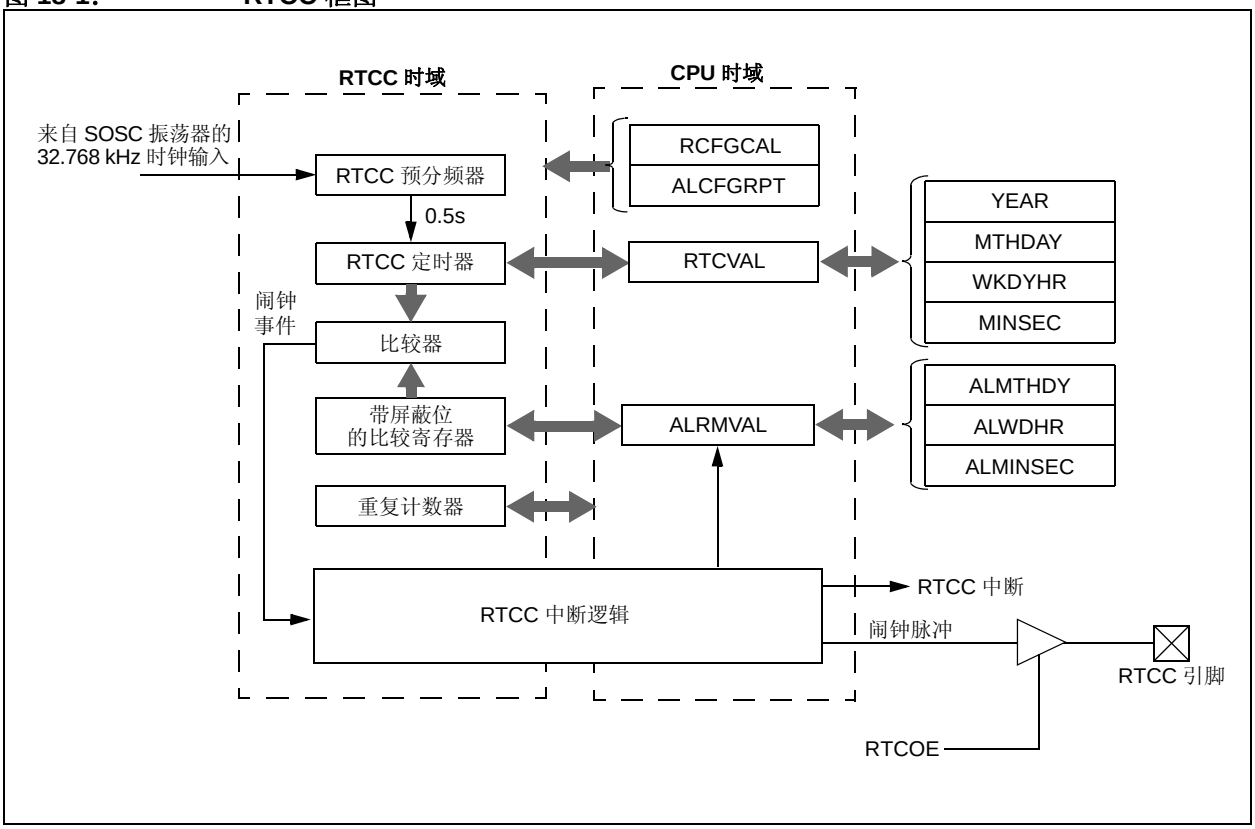
18.0 实时时钟和日历（RTCC）

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 29 章“实时时钟和日历（RTCC）”**（DS39696A_CN）。

实时时钟和日历（Real-Time Clock and Calendar，RTCC）硬件模块具有以下功能：

- 时间：时、分、秒
- 24 小时格式（军用时间）
- 日历：星期、日期、月和年
- 可配置闹钟
- 年份范围：2000 至 2099
- 闰年修正
- 采用 BCD 格式使固件更紧凑
- 针对低功耗工作进行了优化
- 带自动调节的用户校正
- 校正范围：每月误差数为 ±2.64 秒
- 要求：外部 32.768 kHz 时钟晶振
- 闹钟脉冲或秒时钟至 RTCC 引脚输出

图 18-1： RTCC 框图



18.1 RTCC 模块寄存器

RTCC 模块寄存器可分为以下 3 类：

- RTCC 控制寄存器
- RTCC 值寄存器
- 闹钟值寄存器

18.1.1 寄存器映射

为了限制寄存器接口，RTCC 定时器和闹钟时间寄存器需通过相应的寄存器指针来访问。RTCC 值寄存器窗口（RTCVALH 和 RTCVALL）使用 RTCPTR 位（RCFGCAL<9:8>）选择所需的定时器寄存器对（见表 18-1）。通过写 RTCVALH 字节使 RTCC 指针值 RTCPTR<1:0> 每次减 1，直至 00 为止。一旦递减至 00，在手动更改指针之前，可以通过 RTCVALH 和 RTCVALL 访问 MINUTES 和 SECONDS 的值。

表 18-1： RTCVAL 寄存器映射

RTCPTR <1:0>	RTCC 值寄存器窗口	
	RTCVAL<15:8>	RTCVAL<7:0>
00	MINUTES	SECONDS
01	WEEKDAY	HOURS
10	MONTH	DAY
11	—	YEAR

闹钟值寄存器窗口（ALRMVALH 和 ALRMVALL）使用 ALRMPTR 位（ALCFGRPT<9:8>）选择所需的闹钟寄存器对（见表 18-2）。

通过写 ALRMVALH 字节使闹钟指针值 ALRMPTR<1:0> 每次减 1，直至 00 为止。一旦递减至 00，在手动更改指针值之前，可以通过 ALRMVALH 和 ALRMVALL 访问 ALRMMIN 和 ALRMSEC 的值。

表 18-2： ALRMVAL 寄存器映射

ALRMPTR <1:0>	闹钟值寄存器窗口	
	ALRMVAL<15:8>	ALRMVAL<7:0>
00	ALRMMIN	ALRMSEC
01	ALRMWD	ALRMHR
10	ALRMMNTH	ALRMDAY
11	—	—

考虑到 16 位内核不能区别 8 位和 16 位读操作，用户必须注意当读 ALRMVALH 或 ALRMVALL 字节时会使 ALRMPTR<1:0> 值减 1。同样读 RTCVALH 或 RTCVALL 字节会使 RTCPTR<1:0> 值减 1。

注： 仅适用于读操作而非写操作。

18.1.2 写锁定

为了对任何 RTCC 定时器寄存器执行写操作，必须将 RTCWREN 位（RCFGCAL<13>）置 1（见例 18-1）。

例 18-1： 将 RTCWREN 位置 1

```
asm volatile("disi #5");
asm volatile("mov #0x55, w7");
asm volatile("mov w7, _NVMKEY");
asm volatile("mov #0xAA, w8");
asm volatile("mov w8, _NVMKEY");
asm volatile("bset _RCFGCAL, #13");     //set the RTCWREN bit
```

注： 为避免意外写入定时器，建议在除要特意执行写操作之外的任何时候都保持 RTCWREN 位（RCFGCAL<13>）清零。对于要使 RTCWREN 位置 1 的操作，由于在 55h/AAh 序列和 RTCWREN 置 1 之间仅允许有一个指令周期的时间；因此，建议按照例 18-1 中的过程执行代码。

18.1.3 RTCC 控制寄存器

寄存器 18-1： RCFGCAL：RTCC 校准和配置寄存器 (1)

R/W-0	U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0
RTCEN(2)	—	RTCWREN	RTCSYNC	HALFSEC(3)	RTCOE	RTCPTR1	RTCPTR0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
bit 7							bit 0

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

bit 15 **RTCEN：**RTCC 使能位 (2)
1 = 使能 RTCC 模块
0 = 禁止 RTCC 模块

bit 14 **未实现：**读为 0

bit 13 **RTCWREN：**RTCC 值寄存器写使能位
1 = 允许用户写 RTCVALH 和 RTCVALL 寄存器
0 = RTCVALH 和 RTCVALL 寄存器已被锁定不能由用户写入

bit 12 **RTCSYNC：**RTCC 值寄存器读同步位
1 = 由于下溢返回，RTCVALH、RTCVALL 和 ALCFGRPT 寄存器会更改，导致读取的数据无效。如果读取寄存器两次得到的结果相同，就可认为数据有效。
0 = 可以读 RTCVALH、 RTCVALL 或 ALCFGRPT 寄存器，不用担心下溢返回

bit 11 **HALFSEC：**半秒状态位 (3)
1 = 一秒的后半秒
0 = 一秒的前半秒

bit 10 **RTCOE：**RTCC 输出使能位
1 = 使能 RTCC 输出
0 = 禁止 RTCC 输出

bit 9-8 **RTCPTR1:RTCPTR0：**RTCC 值寄存器窗口指针位
当读 RTCVALH 和 RTCVALL 寄存器时，指针指向相应的 RTCC 值寄存器。每读写 RTCVALH 时，RTCPTR<1:0> 值减 1 直至 00。
RTCVAL<15:8>：
00 = MINUTES
01 = WEEKDAY
10 = MONTH
11 = 保留
RTCVAL<7:0>：
00 = SECONDS
01 = HOURS
10 = DAY
11 = YEAR

注 1： RCFGCAL 寄存器的复位值取决于复位的类型。
2： 只有当 RTCWREN = 1 时才允许写 RTCEN 位。
3： 该位是只读的。当写 MINSEC 寄存器的低半字节时该位清零。

寄存器 18-1： RCFGCAL：RTCC 校准和配置寄存器 (1) （续）

bit 7-0 **CAL7:CAL0：**RTC 漂移校准位

01111111 = 最大正调整：每分钟增加 508 次 RTC 时钟脉冲
...
01111111 = 最小正调整：每分钟增加 4 次 RTC 时钟脉冲
00000000 = 不调整
11111111 = 最小负调整：每分钟减少 4 次 RTC 时钟脉冲
...
10000000 = 最大负调整：每分钟减少 512 次 RTC 时钟脉冲

注 1： RCFGCAL 寄存器的复位值取决于复位的类型。

2： 只有当 RTCWREN = 1 时才允许写 RTCEN 位。

3： 该位是只读的。当写 MINSEC 寄存器的低半字节时该位清零。

寄存器 18-2： PADCFG1：填充配置控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	RTSECSEL(1)	PMPTTL(2)
bit 7							bit 0

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

bit 15-2 **未实现：**读为 0

bit 1 **RTSECSEL：**RTCC 秒时钟输出选择位 (1)
1 = RTCC 引脚输出 RTCC 秒时钟
0 = RTCC 引脚输出 RTCC 闹钟脉冲

bit 0 **PMPTTL：**PMP 模块 TTL 输入缓冲器选择位 (2)
1 = PMP 模块使用 TTL 输入缓冲器
0 = PMP 模块使用施密特输入缓冲器

注 1：要使能实际的 RTCC 输出，需要将 RTCCFG （RTCOE）位置 1。

2：有关受影响的 PMP 输入，请参见表 1-2。

寄存器 18-3： ALCFGRPT：闹钟配置寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
bit 7							bit 0

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

bit 15 **ALRMEN：**闹钟使能位

1 = 使能闹钟 （只要 ARPT<7:0> = 00 且 CHIME = 0，该位就会在闹钟事件后自动清零）
0 = 禁止闹钟

bit 14 **CHIME：**持续鸣叫使能位

1 = 使能持续鸣叫，允许 ARPT<7:0> 位从 00h 到 FFh 计满返回
0 = 禁止持续鸣叫， ARPT<7:0> 位到达 00h 时停止

bit 13-10 **AMASK3:AMASK0：**闹钟屏蔽配置位

0000 = 每半秒一次
0001 = 每秒一次
0010 = 每 10 秒一次
0011 = 每分钟一次
0100 = 每 10 分钟一次
0101 = 每小时一次
0110 = 每天一次
0111 = 每星期一次
1000 = 每月一次
1001 = 每年一次 （配置为 2 月 29 日则为每 4 年）
101x = 保留－不用
11xx = 保留－不用

bit 9-8 **ALRMPTR1:ALRMPTR0：**闹钟值寄存器窗口指针位

当读 ALRMVALH 和 ALRMVALL 寄存器时，指针指向相应的闹钟值寄存器。每读写 ALRMVALH 时，ALRMPTR<1:0> 值减 1 直至 00。

ALRMVAL<15:8>：
00 = ALRMMIN
01 = ALRMWD
10 = ALRMMNTH
11 = 未实现

ALRMVAL<7:0>：
00 = ALRMSEC
01 = ALRMHR
10 = ALRMDAY
11 = 未实现

bit 7-0 **ARPT7:ARPT0：**闹钟重复计数器值位

11111111 = 闹钟再重复 255 次
...
00000000 = 闹钟不重复
每发生一次闹钟事件，计数器就减 1。计数器不会在递减到 00h 后返回到 FFh，除非 CHIME = 1。

18.1.4 RTCVAL 寄存器映射

寄存器 18-4： YEAR：年值寄存器 (1)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
YRTEN3	YRTEN2	YRTEN1	YRTEN0	YRONE3	YRONE2	YRONE1	YRONE0
bit 7							bit 0

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

bit 15-8 **未实现：**读为 0
bit 7-4 **YRTEN3:YRTEN0：**年的十位数字的 BCD 码；值为 0 到 9
bit 3-0 **YRONE3:YRONE0：**年的个位数字的 BCD 码；值为 0 到 9

注 1： 只有在 RTCWREN = 1 时才允许写 YEAR 寄存器。

寄存器 18-5： MTHDY：月和天值寄存器 (1)

U-0	U-0	U-0	R-x	R-x	R-x	R-x	R-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 15							bit 8

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

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

bit 15-13 **未实现：**读为 0
bit 12 **MTHTEN0：**月的十位数字的 BCD 码；值为 0 或 1
bit 11-8 **MTHONE3:MTHONE0：**月的个位数字的 BCD 码；值为 0 到 9
bit 7-6 **未实现：**读为 0
bit 5-4 **DAYTEN1:DAYTEN0：**天的十位数字的 BCD 码；值为 0 到 3
bit 3-0 **DAYONE3:DAYONE0：**天的个位数字的 BCD 码；值为 0 到 9

注 1： 只有当 RTCWREN = 1 时才允许写该寄存器。

寄存器 18-6： WKDYHR：星期和小时值寄存器 (1)

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 15							bit 8

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

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

bit 15-11 **未实现：**读为 0
bit 10-8 **WDAY2:WDAY0：**星期数字的 BCD 码；值为 0 到 6
bit 7-6 **未实现：**读为 0
bit 5-4 **HRTEN1:HRTEN0：**小时的十位数字的 BCD 码；值为 0 到 2
bit 3-0 **HRONE3:HRONE0：**小时的个位数字的 BCD 码；值为 0 到 9

注 1： 只有当 RTCWREN = 1 时才允许写该寄存器。

寄存器 18-7： MINSEC：分钟和秒值寄存器

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 15							bit 8

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

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

bit 15 **未实现：**读为 0
bit 14-12 **MINTEN2:MINTEN0：**分钟的十位数字的 BCD 码；值为 0 到 5
bit 11-8 **MINONE3:MINONE0：**分钟的个位数字的 BCD 码；值为 0 到 9
bit 7 **未实现：**读为 0
bit 6-4 **SECTEN2:SECTEN0：**秒的十位数字的 BCD 码；值为 0 到 5
bit 3-0 **SECONE3:SECONE0：**秒的个位数字的 BCD 码；值为 0 到 9

18.1.5 ALRMVAL 寄存器映射

寄存器 18-8： ALMTHDY：闹钟月和天值寄存器 (1)

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 15							bit 8

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

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

bit 15-13 **未实现：**读为 0
bit 12 **MTHTEN0：**月的十位数字的 BCD 码；值为 0 或 1
bit 11-8 **MTHONE3:MTHONE0：**月的个位数字的 BCD 码；值为 0 到 9
bit 7-6 **未实现：**读为 0
bit 5-4 **DAYTEN1:DAYTEN0：**天的十位数字的 BCD 码；值为 0 到 3
bit 3-0 **DAYONE3:DAYONE0：**天的个位数字的 BCD 码；值为 0 到 9

注 1：只有当 RTCWREN = 1 时才允许写该寄存器。

寄存器 18-9： ALWDHR：闹钟星期和小时值寄存器 (1)

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 15							bit 8

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

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

bit 15-11 **未实现：**读为 0
bit 10-8 **WDAY2:WDAY0：**星期数字的 BCD 码；值为 0 到 6
bit 7-6 **未实现：**读为 0
bit 5-4 **HRTEN1:HRTEN0：**小时的十位数字的 BCD 码；值为 0 到 2
bit 3-0 **HRONE3:HRONE0：**小时的个位数字的 BCD 码；值为 0 到 9

注 1：只有当 RTCWREN = 1 时才允许写该寄存器。

寄存器 18-10：　　ALMINSEC：闹钟分钟和秒值寄存器

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 15							bit 8

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

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

bit 15　**未实现：**读为 0

bit 14-12　**MINTEN2:MINTEN0：**分钟的十位数字的 BCD 码；值为 0 到 5

bit 11-8　**MINONE3:MINONE0：**分钟的个位数字的 BCD 码；值为 0 到 9

bit 7　**未实现：**读为 0

bit 6-4　**SECTEN2:SECTEN0：**秒的十位数字的 BCD 码；值为 0 到 5

bit 3-0　**SECONE3:SECONE0：**秒的个位数字的 BCD 码；值为 0 到 9

18.2 校准

可使用周期自动调整功能校准实时晶振输入。当正确校准后，RTCC 产生的误差每月少于 3 秒。标准是通过确定时钟误差脉冲数并将其存储到RCFGCAL 寄存器的低半字节来实现的。装入 RCFGCAL 低半字节的 8 位有符号数乘以 4，在每分钟均会与 RTCC 定时器的值相加或从中减去。请参见以下步骤执行 RTCC 校准：

1. 用户必须使用器件上的另一个定时器来确定 32.768 kHz 晶振的误差。
2. 一旦确定了误差，就必须将其转换为每分钟的时钟误差脉冲数。

公式 18-1：

(理想频率 † – 实测频率) * 60 = 每分钟时钟数

† 理想频率 = 32,768 Hz

3. a) 如果振荡器频率高于理想频率（步骤 2 中计算结果为负）， RCFGCAL 寄存器值就需要为负。这样每分钟会从定时器计数值中减去指定的时钟脉冲数。

   b) 如果振荡器频率低于理想频率（步骤 2 中计算结果为正）， RCFGCAL 寄存器值就需要为正。这样每分钟会将定时器计数器的值加上指定的时钟脉冲数。
4. 将每分钟的误差时钟数除以 4 得到正确的校准值并将该值装入 RCFGCAL 寄存器。（校准值的每 1 位递增 1 都会加上或减去 4 个脉冲）。用正确的值装载 RCFGCAL 寄存器。

   只能当定时器关闭时或在秒脉冲的上升沿后立即写 RCFGCAL 寄存器的低半字节。

注： 由用户决定误差值中是否包含因温度和晶振老化造成的晶振初始误差。

18.3 闹钟

- 闹钟时间间隔可配置为半秒到一年
- 使用 ALRMEN 位（ALCFGRPT<15>，寄存器 18-3）使能闹钟功能
- 具有单次闹钟和重复闹钟选项

18.3.1 配置闹钟

使用 ALRMEN 位使能闹钟功能。当闹钟事件发生后该位清零。只有当 ALRMEN = 0 时才可写 ALRMVALH:ALRMVALL。

如图 18-2 所示，闹钟的间隔时间通过 AMASK 位（ALCFGRPT<13:10>）配置。这些位决定哪些闹钟以及闹钟设置时间的哪几位必须与时钟值匹配才能导致闹钟事件。也可配置闹钟以预先设定的时间间隔重复。一旦使能了闹钟，闹钟次数就被存储在 ALCFGRPT 寄存器的低半字节。

当 ALCFGRPT = 00 且 CHIME 位（ALCFGRPT<14>）= 0 时，重复功能被禁止，只可发生单次闹钟。如果将 FFh 装载到 ALCFGRPT 寄存器的低半字节，可实现 255 次重复闹钟。

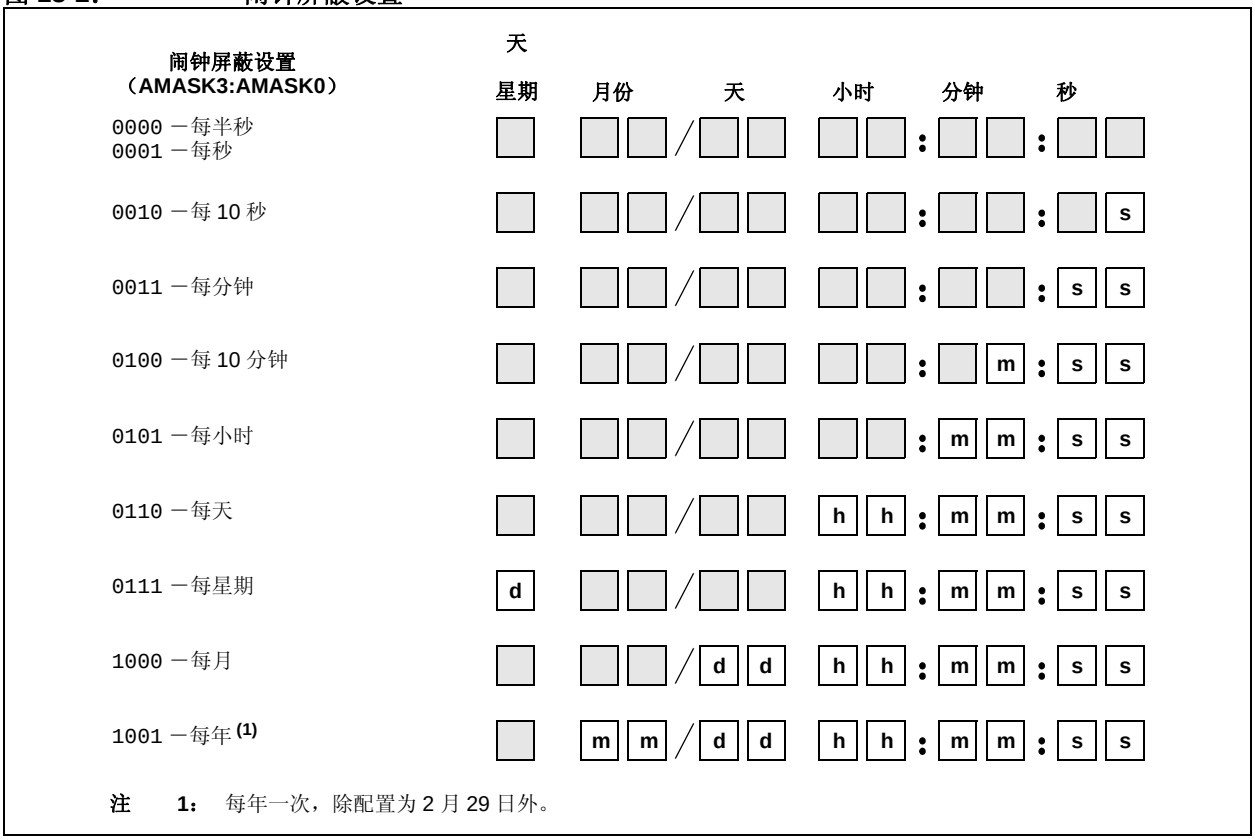
闹钟每发出一次报警，ALCFGRPT 寄存器就会减 1。寄存器一旦到达 00，就会发出最后一次报警，之后 ALRMEN 位会自动清零，然后关闭闹钟功能。如果 CHIME 位 = 1，就会无限制地重复鸣叫。在这种情况下，当 ALCFGRPT 到达 00 时并不会禁止闹钟，而是寄存器会下溢返回到 FFh，然后继续不停计数。

18.3.2 闹钟中断

伴随每次闹钟事件都会产生一次中断。此外，还将输出一个频率为闹钟时钟频率一半的闹钟脉冲序列。闹钟脉冲输出与 RTCC 时钟完全同步，可以作为其他外设的触发时钟。

注： 当使能闹钟（ALRMEN = 1）时，更改除 RCFGCAL 和 ALCFGRPT 以外的任何寄存器或寄存器 CHIME 位，都会导致产生错误闹钟事件，从而产生错误闹钟中断。要避免错误闹钟事件，只能在禁止闹钟（ALRMEN = 0）时更改定时器和闹钟设置值。建议在 RTCSYNC = 0 时更改 ALCFGRPT 寄存器和 CHIME 位。

图 18-2：　　闹钟屏蔽设置



注　　1：　每年一次，除配置为 2 月 29 日外。

注：

19.0 可编程循环冗余校验（CRC）发生器

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 30 章“可编程循环冗余校验（CRC）”**（DS39714A_CN）。

可编程循环冗余校验（Cyclic Redundancy Check，CRC）发生器具有以下特性：

- 用户可编程的 CRC 多项式方程
- 中断输出
- 数据 FIFO

19.1 寄存器

有四个寄存器用于控制可编程 CRC 的操作：

- CRCCON
- CRCXOR
- CRCDAT
- CRCWDAT

19.2 概述

模块实现了一个可由软件配置的 CRC 发生器。可分别使用 CRCXOR（X<15:1>）位和 CRCCON（PLEN3:PLEN0）位来设置多项式的项和长度。

考虑以下公式：

公式 19-1： CRC 多项式

$$x^{16} + x^{12} + x^5 + 1$$

在 CRC 发生器中编程该多项式，应按照表 19-1 所示来设置 CRC 寄存器的位。

表 19-1： CRC 设置示例

位名	位值
PLEN3:PLEN0	1111
X<15:1>	000100000010000

请注意根据该方程的要求，应将 X<15:1> 的第 12 位和第 5 位置 1。方程的第 0 位要求是其余位（第 16 位除外）的异或值。对于 16 次多项式，第 16 位也被认为是其余位（bit 0 除外）的异或值，因此 X<15:1> 位不包含 bit 0 或第 16 位。

图 19-2 给出了标准 CRC 发生器的结构。

图 19-1： CRC 移位器详细结构

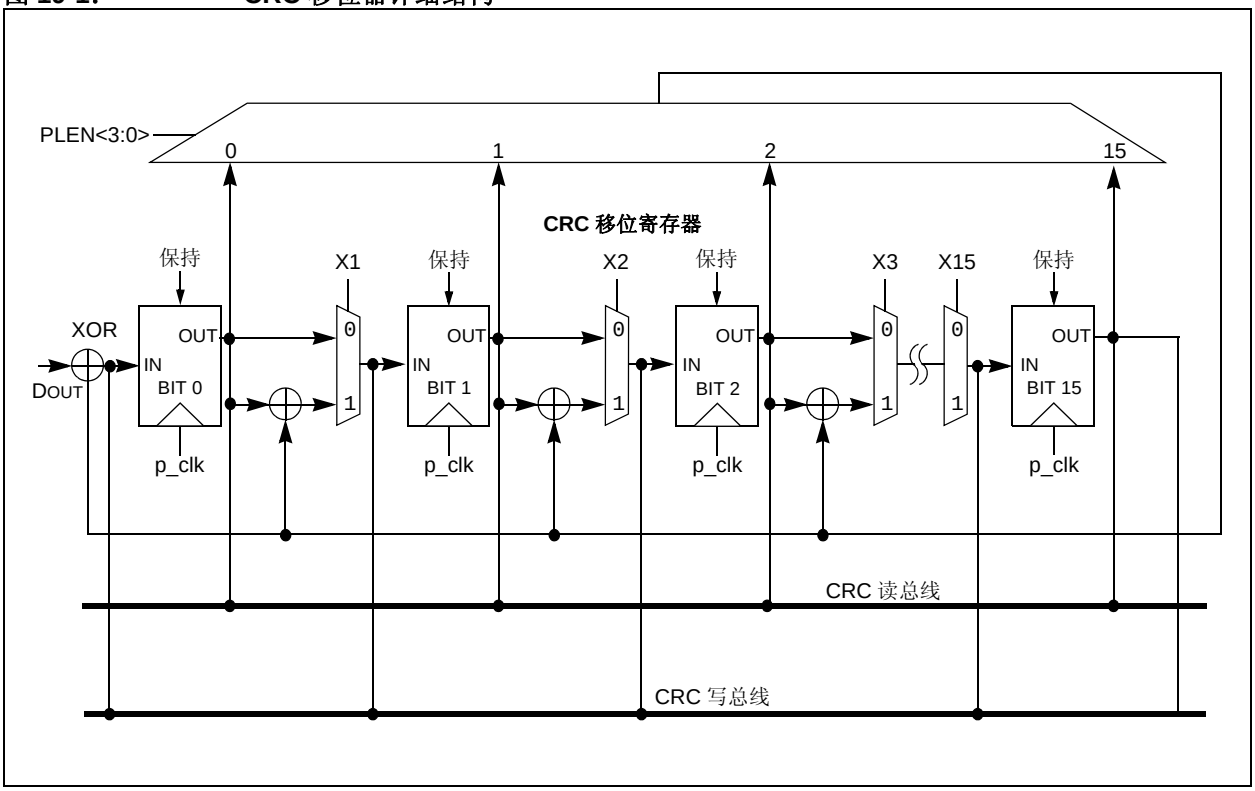
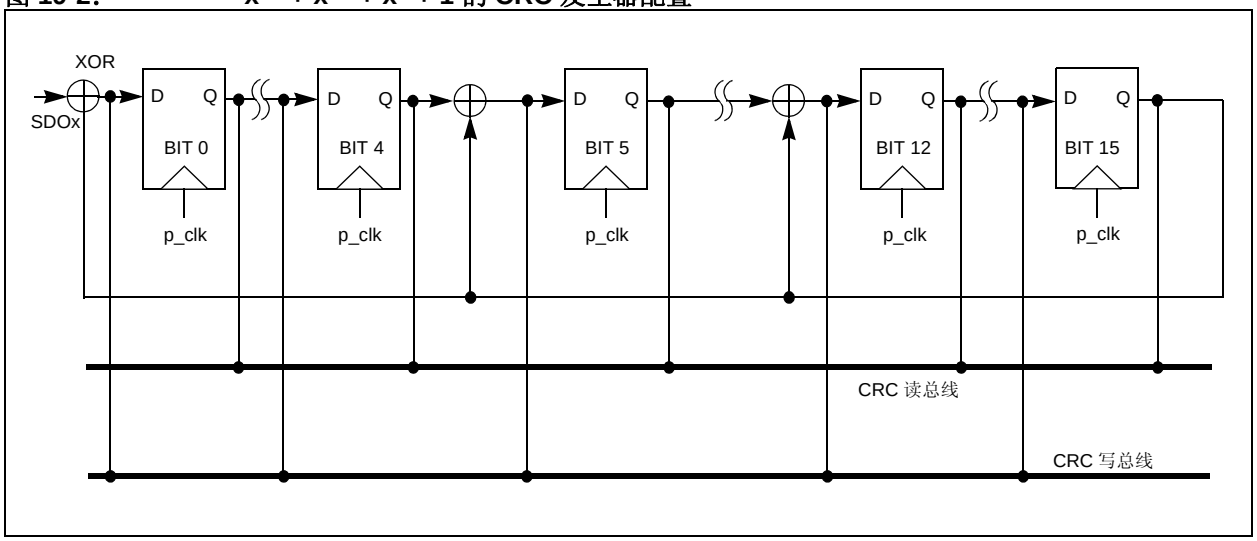


图 19-2：　　$x^{16} + x^{12} + x^5 + 1$ 的 CRC 发生器配置



19.3　用户接口

19.3.1　数据接口

要启动串行移位，必须将 CRCGO 位置 1。

该模块配备有一个 FIFO 缓冲器，当 PLEN（PLEN<3:0>）> 7 时，FIFO 为 8 级深，其他情况下为 16 级深。首先将要计算 CRC 的数据写入 FIFO。可被写入 FIFO 的最小数据长度为 1 个字节。例如，如果 PLEN = 5，那么数据的长度为 PLEN + 1 = 6。该数据必须以如下方式写入：

data[5:0] = crc_input[5:0]

data[7:6] = 'bxx

数据一旦被写入 CRCWDAT 的最高位（由 PLEN 定义），VWORD（VWORD<4:0>）的值就会加 1。当 CRCGO = 1 且 VWORD > 0 时，串行移位器开始将数据移入 CRC 引擎。当最高位移出后，VWORD 减 1。串行移位器会继续移出数据直到 VWORD 达到 0。因此对于给定的 PLEN 值，需要 (PLEN + 1) * VWORD 个时钟周期才能完成 CRC 计算。

当 VWORD 达到 8（或 16）时，CRCFUL 位将置 1。当 VWORD 达到 0 时，CRCMPT 位将置 1。

要将数据持续地送入 CRC 引擎，建议以足够数量的数据字预先将 FIFO 填满，这样在写下一个字之前将不会产生中断。完成后，通过将 CRCGO 位置 1 开始 CRC 计算。同时应开始查询 VWORD 位。如果读取的字节数小于 8 或 16，则可向 FIFO 写入另一个字。

要将写入 FIFO 的字清空，必须将 CRCGO 位置 1，并且允许 CRC 移位器持续工作直到 CRCMPT 置 1。

同样为了获取正确的 CRC 值，在读取 CRCWDAT 寄存器前必须等待 CRCMPT 位置 1。

如果当 CRCFUL 位置 1 时写入了一个字，则 VWORD 指针将返回至 0。随后的硬件操作假设 FIFO 为空。由于并未满足产生中断的条件，因而不会产生中断（见 **第 19.3.2 节 “中断操作”**）。

在写入 CRCWDAT 之后，必须至少经过一个指令周期才可以读 VWORD 位。

19.3.2　中断操作

当 VWORD4:VWORD0 的值从 1 变为 0 时，产生中断。

寄存器 19-1：　　CRCCON：CRC 控制寄存器

U-0	U-0	R/W-0	R-0	R-0	R-0	R-0	R-0
—	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0
bit 15							bit 8

R-0	R-1	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CRCFUL	CRCMPT	—	CRCGO	PLEN3	PLEN2	PLEN1	PLEN0
bit 7							bit 0

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

bit 15-14 **未实现：**读为 0

bit 13 **CSIDL：**空闲模式 CRC 停止位
1 = 当器件进入空闲模式后，模块停止工作
0 = 在空闲模式下模块继续工作

bit 12-8 **VWORD4:VWORD0：**指针值位
指出 FIFO 中有效字的个数。当 PLEN3:PLEN0 > 7 时，最大值为 8；
当 PLEN3:PLEN0 ≤ 7 时，最大值为 16。

bit 7 **CRCFUL：**FIFO 满位
1 = FIFO 满
0 = FIFO 未满

bit 6 **CRCMPT：**FIFO 空位
1 = FIFO 空
0 = FIFO 未空

bit 5 **未实现：**读为 0

bit 4 **CRCGO：**启动 CRC 位
1 = 启动 CRC 串行移位器
0 = 关闭 CRC 串行移位器

bit 3-0 **PLEN3:PLEN0：**多项式长度位
表示将要生成的多项式的长度减 1。

19.4　在节电模式下的操作

19.4.1　休眠模式

如果器件在模块运行过程中进入休眠模式，模块将暂停工作，保持当前状态直到时钟恢复工作。

19.4.2　空闲模式

要在空闲模式下继续工作，必须在进入空闲模式前将 CSIDL 位清零。

如果 CSIDL = 1，模块的行为和在休眠模式下一样；但是即使模块时钟不可用，待处理的中断事件也会被传递给 CPU。

注：

20.0 10 位高速 A/D 转换器

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 17 章“10 位 A/D 转换器”**（DS39705A_CN）。

10 位 A/D 转换器具有以下主要特性：

- 逐次逼近 （SAR）转换
- 转换速度最大达到 500 ksps
- 最多 16 个模拟输入引脚
- 外部参考电压输入引脚
- 自动通道扫描模式
- 可选转换触发源
- 16 字转换结果缓冲器
- 可选缓冲器填充模式
- 四种结果对齐方式
- 可在 CPU 休眠和空闲模式下工作

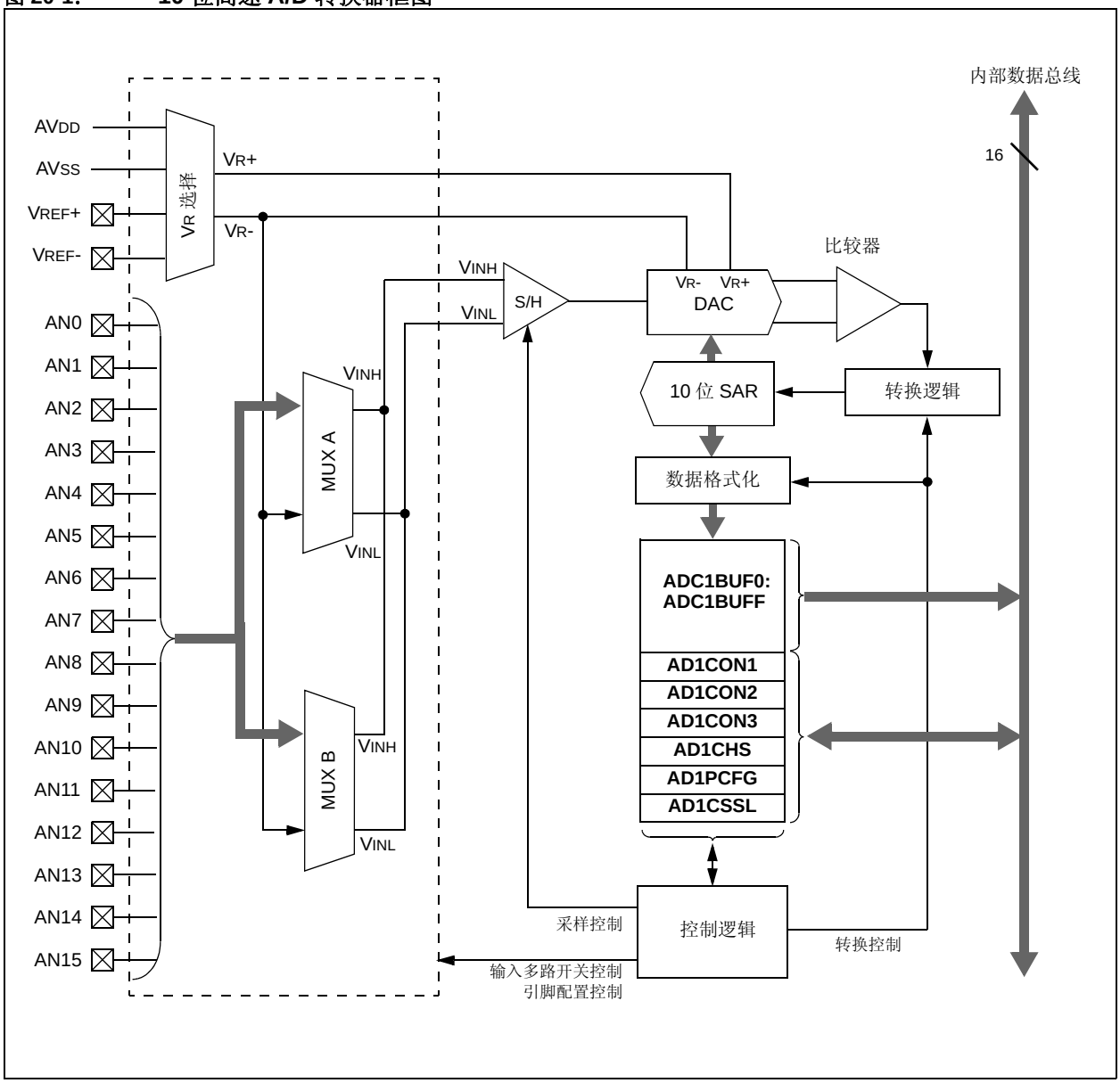
根据特定器件的引脚配置， 10 位 A/D 转换器最多可以有 16 个模拟输入引脚，即 AN0 到 AN15。此外，还有两个可用于连接外部参考电压的模拟输入引脚。这两个参考电压输入可以和其他模拟输入引脚复用。实际的模拟输入引脚数和外部参考电压输入配置取决于具体的器件。更多详细信息请参见器件数据手册。

A/D 转换器的框图请参见图 20-1。

要执行 A/D 转换：

1. 配置 A/D 模块：
   a) 选择用作模拟输入的端口引脚（AD1PCFG<15:0>）。
   b) 选择符合模拟输入预期范围的参考电压源（AD1CON2<15:13>）。
   c) 选择模拟转换时钟从而使预期的数据速率与处理器时钟匹配 （AD1CON3<7:0>）。
   d) 选择适合的采样/转换序列（AD1CON1<7:0>和AD1CON3<12:8>）。
   e) 选择转换结果出现在缓冲器中的方式（AD1CON1<9:8>）。
   f) 选择中断率 （AD1CON2<5:2>）。
   g) 启动 A/D 模块 （AD1CON1<15>）。
2. 配置 A/D 中断 （如果需要）：
   a) 将 AD1IF 位清零。
   b) 选择 A/D 中断优先级。

图 20-1： 10 位高速 A/D 转换器框图



寄存器 20-1：　AD1CON1：A/D 控制寄存器 1

R/W-0	U-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
ADON	—	ADSIDL	—	—	—	FORM1	FORM0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0,HCS	R/C-0,HCS
SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE
bit 7							bit 0

图注：	C = 可清零位	HSC = 硬件清零 / 置 1 位	
R = 可读位	W = 可写位	0 = 只清零位	U = 未实现位，读为 0
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15 **ADON**：A/D 工作模式位
1 = A/D 转换器模块正在工作
0 = A/D 转换器已关闭

bit 14 **未实现**：读为 0

bit 13 **ADSIDL**：空闲模式停止位
1 = 当器件进入空闲模式时，模块停止工作
0 = 在空闲模式下模块继续工作

bit 12-10 **未实现**：读为 0

bit 9-8 **FORM1:FORM0**：数据输出格式位
11 = 有符号小数 （sddd dddd dd00 0000）
10 = 小数 （sddd dddd dd00 0000）
01 = 有符号整数 （ssss sssd dddd dddd）
00 = 整数 （0000 00dd dddd dddd）

bit 7-5 **SSRC2:SSRC0**：转换触发源选择位
111 = 内部计数器结束采样并启动转换 （自动转换）
110 = 保留
10x = 保留
011 = 保留
010 = Timer3 比较器结束采样并启动转换
001 = INT0 引脚的有效电平转换边沿结束采样并启动转换
000 = SAMP 位清零结束采样并启动转换

bit 4-3 **未实现**：读为 0

bit 2 **ASAM**：A/D 采样自动启动位
1 = 采样在上一次转换结束后立即开始。 SAMP 位自动置 1。
0 = 采样在 SAMP 位置 1 后开始

bit 1 **SAMP**：A/D 采样使能位
1 = A/D 采样 / 保持放大器正在采样输入
0 = A/D 采样 / 保持放大器正在保持

bit 0 **DONE**：A/D 转换状态位
1 = A/D 转换完成
0 = A/D 转换未完成

寄存器 20-2： AD1CON2：A/D 控制寄存器 2

R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	U-0	U-0
VCFG2	VCFG1	VCFG0	r	—	CSCNA	—	—
bit 15							bit 8

R-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUFS	—	SMPI3	SMPI2	SMPI1	SMPI0	BUFM	ALTS
bit 7							bit 0

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

bit15-13 **VCFG2:VCFG0：**参考电压配置位：

VCFG2:VCFG0	VR+	VR-
000	AVDD	AVSS
001	外部 VREF+ 引脚	AVSS
010	AVDD	外部 VREF- 引脚
011	外部 VREF+ 引脚	外部 VREF- 引脚
1xx	AVDD	AVSS

bit 12 **保留**

bit 11 **未实现：**读为 0

bit 10 **CSCNA：**由 MUX A 输入多路开关控制的 CH0+ S/H 输入的扫描输入选择位
1 = 扫描输入
0 = 不扫描输入

bit 9-8 **未实现：**读为 0

bit 7 **BUFS：**缓冲器填充状态位 （仅当 BUFM = 1 时有效）
1 = A/D 当前正在填充缓冲器 08-0F，用户应该访问 00-07 中的数据
0 = A/D 当前正在填充缓冲器 00-07，用户应该访问 08-0F 中的数据

bit 6 **未实现：**读为 0

bit 5-2 **SMPI3:SMPI0：**每次中断的采样 / 转换过程数选择位
1111 = 每完成 16 个采样 / 转换过程后产生中断
1110 = 每完成 15 个采样 / 转换过程后产生中断
.....
0001 = 每完成两个采样 / 转换过程后产生中断
0000 = 在完成每个采样 / 转换过程后产生中断

bit 1 **BUFM：**缓冲器模式选择位
1 = 缓冲器被配置为两个 8 字缓冲器 （ADC1BUFx<15:8> 和 ADC1BUFx<7:0>）
0 = 缓冲器被配置为一个 16 字缓冲器 （ADC1BUFx<15:0>）

bit 0 **ALTS：**交替输入采样模式选择位
1 = 使用 MUX A 输入多路开关设置进行第一次采样，然后交替使用 MUX B 和 MUX A 输入多路开关设置进行后续采样
0 = 总是使用 MUX A 输入多路开关设置

寄存器 20-3： AD1CON3：A/D 控制寄存器 3

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADRC	—	—	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS7	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0
bit 7							bit 0

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

bit 15 **ADRC**：A/D 转换时钟源位
1 = A/D 内部 RC 时钟
0 = 时钟由系统时钟产生

bit 14-13 **未实现**：读为 0

bit 12-8 **SAMC4:SAMC0**：自动采样时间位
11111 = 31 TAD
• • • • • •
00001 = 1 TAD
00000 = 0 TAD （不建议）

bit 7-0 **ADCS7:ADCS0**：A/D 转换时钟选择位
11111111 = 256 * TCY
• • • • • •
00000001 = 2 * TCY
00000000 = TCY

寄存器 20-4：　　AD1CHS：A/D 输入选择寄存器

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NB1	CH0NB0	—	—	CH0SB3	CH0SB2	CH0SB1	CH0SB0
bit 15							bit 8

R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NA	—	—	—	CH0SA3	CH0SA2	CH0SA1	CH0SA0
bit 7							bit 0

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

bit 15　**CH0NB：**MUX B 多路开关的通道 0 负输入选择位
1 = 通道 0 负输入为 AN1
0 = 通道 0 负输入为 VR-

bit 14-12　**未实现：**读为 0

bit 11-8　**CH0SB3:CH0SB0：**MUX B 多路开关的通道 0 正输入选择位
1111 = 通道 0 正输入为 AN15
1110 = 通道 0 正输入为 AN14
• • • • • •
0001 = 通道 0 正输入为 AN1
0000 = 通道 0 正输入为 AN0

bit 7　**CH0NA：**MUX A 多路开关的通道 0 负输入选择位
1 = 通道 0 负输入为 AN1
0 = 通道 0 负输入为 VR-

bit 6-4　**未实现：**读为 0

bit 3-0　**CH0SA3:CH0SA0：**MUX A 多路开关的通道 0 正输入选择位
1111 = 通道 0 正输入为 AN15
1110 = 通道 0 正输入为 AN14
• • • • •
0001 = 通道 0 正输入为 AN1
0000 = 通道 0 正输入是 AN0

寄存器 20-5： AD1PCFG：A/D 端口配置寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG15	PCFG14	PCFG13	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

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

bit 15-0 **PCFG15:PCFG0：**模拟输入引脚配置控制位

1 = 对应模拟通道的引脚被配置为数字模式；使能 I/O 端口读取

0 = 引脚被配置为模拟模式；禁止 I/O 端口读取， A/D 对引脚电压进行采样

寄存器 20-6： AD1CSSL：A/D 输入扫描选择寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSSL15	CSSL14	CSSL13	CSSL12	CSS1L11	CSSL10	CSSL9	CSSL8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSSL7	CSSL6	CSSL5	CSSL4	CSSL3	CSSL2	CSSL1	CSSL0
bit 7							bit 0

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

bit 15-0 **CSSL15:CSSL0：**A/D 输入引脚扫描选择位

1 = 选择对应的模拟通道进行输入扫描

0 = 不对模拟通道进行输入扫描

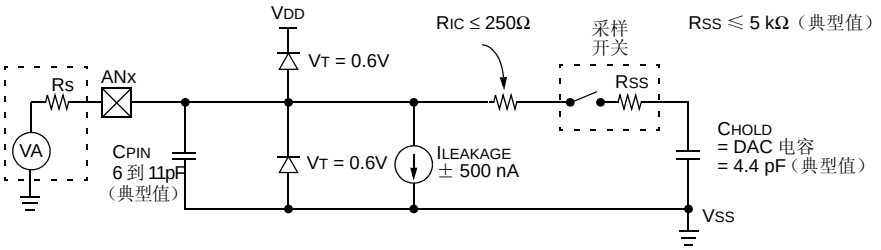
公式 20-1：　　A/D 转换时钟周期 (1)

$$T_{AD} = T_{CY}(ADCS + 1)$$

$$ADCS = \frac{T_{AD}}{T_{CY}} - 1$$

注　1： 上述公式基于 TCY = TOSC * 2，且打盹模式和 PLL 是被禁止的。

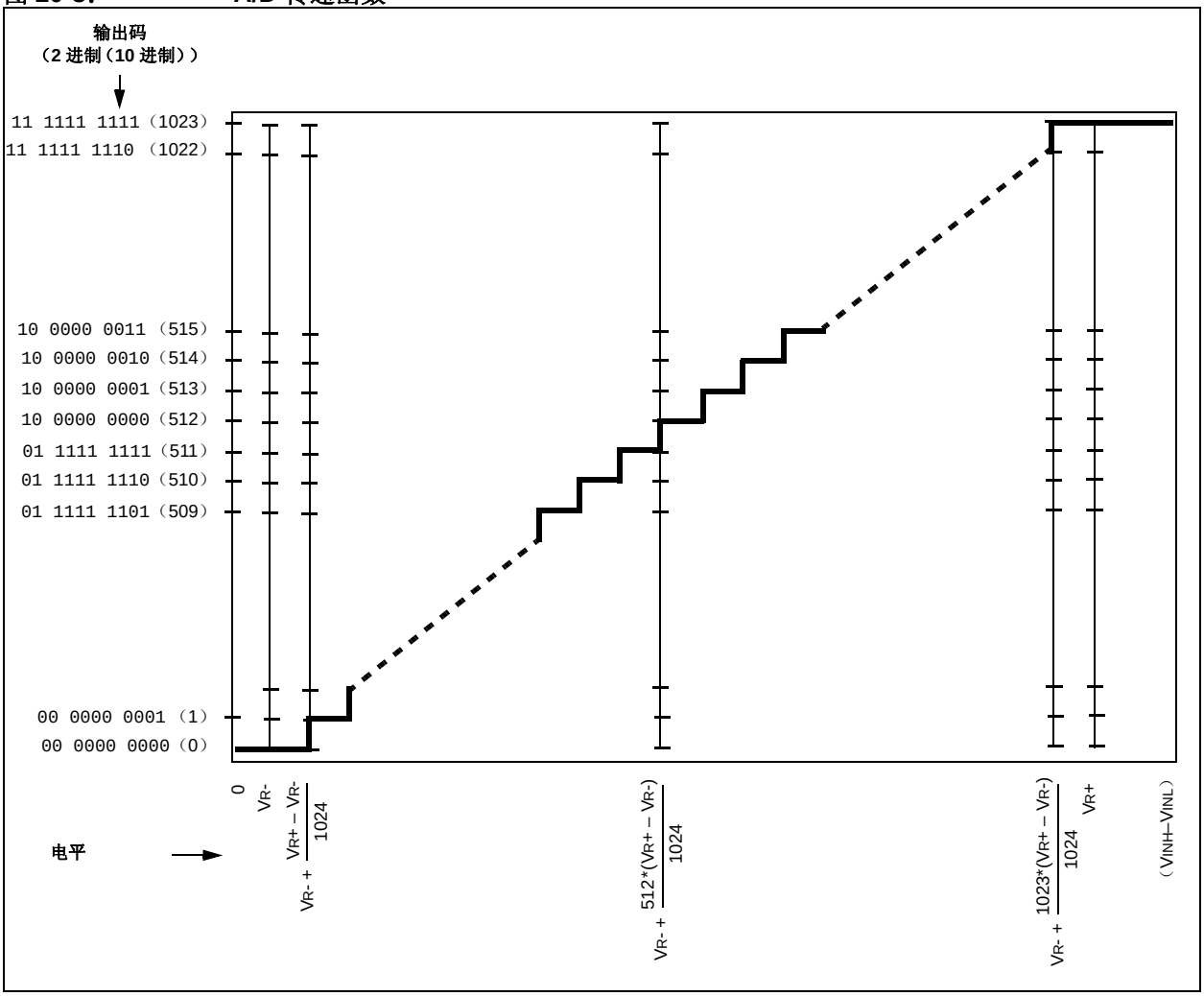
图 20-2：　　10 位 A/D 转换器模拟输入模型



图注：		
	CPIN	= 输入电容
	VT	= 门限电压
	ILEAKAGE	= 各结点在引脚上引起的泄漏电流
	RIC	= 片内走线等效电阻
	RSS	= 采样开关电阻
	CHOLD	= 采样 / 保持电容（来自 DAC）

注： CPIN 值取决于器件的封装并且未经测试。如果 RS ≤ 5 kΩ， CPIN 的影响可忽略。

图 20-3：　　A/D 传递函数


输出码
（2 进制（10 进制））
11 1111 1111（1023）
11 1111 1110（1022）
10 0000 0011（515）
10 0000 0010（514）
10 0000 0001（513）
10 0000 0000（512）
01 1111 1111（511）
01 1111 1110（510）
01 1111 1101（509）
00 0000 0001（1）
00 0000 0000（0）
电平
0
VR-
VR- + (VR+ – VR-)/1024
VR- + 512*(VR+ – VR-)/1024
VR- + 1023*(VR+ – VR-)/1024
VR+
（VINH–VINL）

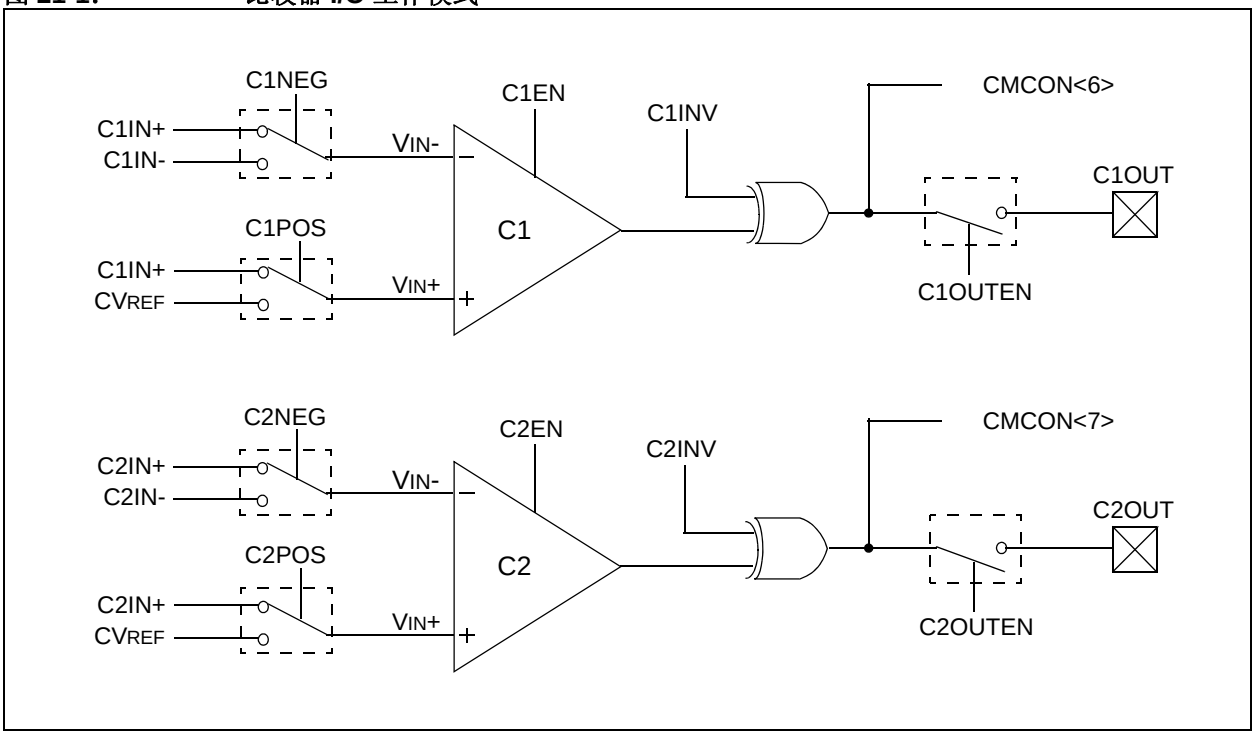
注：

21.0 比较器模块

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 2 章“比较器模块”**（DS39710A_CN）。

比较器模块包含两个比较器，有多种配置方法。可通过与 I/O 引脚复用的模拟输入选择比较器的输入和片上参考电压。图 21-1 示出了比较器各种配置的框图。

图 21-1： 比较器 I/O 工作模式



寄存器 21-1：　　CMCON：比较器控制寄存器

R/W-0	U-0	R/C-0	R/C-0	R/W-0	R/W-0	R/W-0	R/W-0
CMIDL	—	C2EVT	C1EVT	C2EN	C1EN	C2OUTEN	C1OUTEN
bit 15							bit 8

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	C2INV	C1INV	C2NEG	C2POS	C1NEG	C1POS
bit 7							bit 0

图注：	C = 可清零位		
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15　**CMIDL：**空闲模式停止位
1 = 当器件进入空闲模式时，模块不产生中断。模块继续使能。
0 = 在空闲模式下模块继续正常工作

bit 14　**未实现：**读为 0

bit 13　**C2EVT：**比较器 2 事件位
1 = 比较器输出更改了状态
0 = 比较器输出未更改状态

bit 12　**C1EVT：**比较器 1 事件位
1 = 比较器输出更改了状态
0 = 比较器输出未更改状态

bit 11　**C2EN：**比较器 2 使能位
1 = 比较器使能
0 = 比较器禁止

bit 10　**C1EN：**比较器 1 使能位
1 = 比较器使能
0 = 比较器禁止

bit 9　**C2OUTEN：**比较器 2 输出使能位
1 = 使用比较器输出驱动输出引脚
0 = 不使用比较器输出驱动输出引脚

bit 8　**C1OUTEN：**比较器 1 输出使能位
1 = 使用比较器输出驱动输出引脚
0 = 不使用比较器输出驱动输出引脚

bit 7　**C2OUTEN：**比较器 2 输出位
当 C2INV = 0 时：
1 = C2 V_{IN+} > C2 V_{IN-}
0 = C2 V_{IN+} < C2 V_{IN-}
当 C2INV = 1 时：
0 = C2 V_{IN+} > C2 V_{IN-}
1 = C2 V_{IN+} < C2 V_{IN-}

bit 6　**C1OUT：**比较器 1 输出位
当 C1INV = 0 时：
1 = C1 V_{IN+} > C1 V_{IN-}
0 = C1 V_{IN+} < C1 V_{IN-}
当 C1INV = 1 时：
0 = C1 V_{IN+} > C1 V_{IN-}
1 = C1 V_{IN+} < C1 V_{IN-}

寄存器 21-1： CMCON：比较器控制寄存器 （续）

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 **C2NEG**：比较器 2 反相输入配置位
1 = 输入连接到 VIN+
0 = 输入连接到 VIN-
请参见图 21-1，了解比较器模式。

bit 2 **C2POS**：比较器 2 同相输入配置位
1 = 输入连接到 VIN+
0 = 输入连接到 CVREF
请参见图 21-1，了解比较器模式。

bit 1 **C1NEG**：比较器 1 反相输入配置位
1 = 输入连接到 VIN+
0 = 输入连接到 VIN-
请参见图 21-1，了解比较器模式。

bit 0 **C1POS**：比较器 1 同相输入配置位
1 = 输入连接到 VIN+
0 = 输入连接到 CVREF
请参见图 21-1，了解比较器模式。

注：

22.0 比较器参考电压

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 20 章“比较器参考电压模块”**（DS39709A_CN）。

22.1 配置比较器参考电压

参考电压模块由 CVRCON 寄存器（寄存器 22-1）控制。比较器参考电压提供两种范围的输出电压，每种范围都具有 16 种不同的电平。CVRR 位（CVRCON<5>）用于选择电压的范围。这两种电压范围的主要区别在于 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。

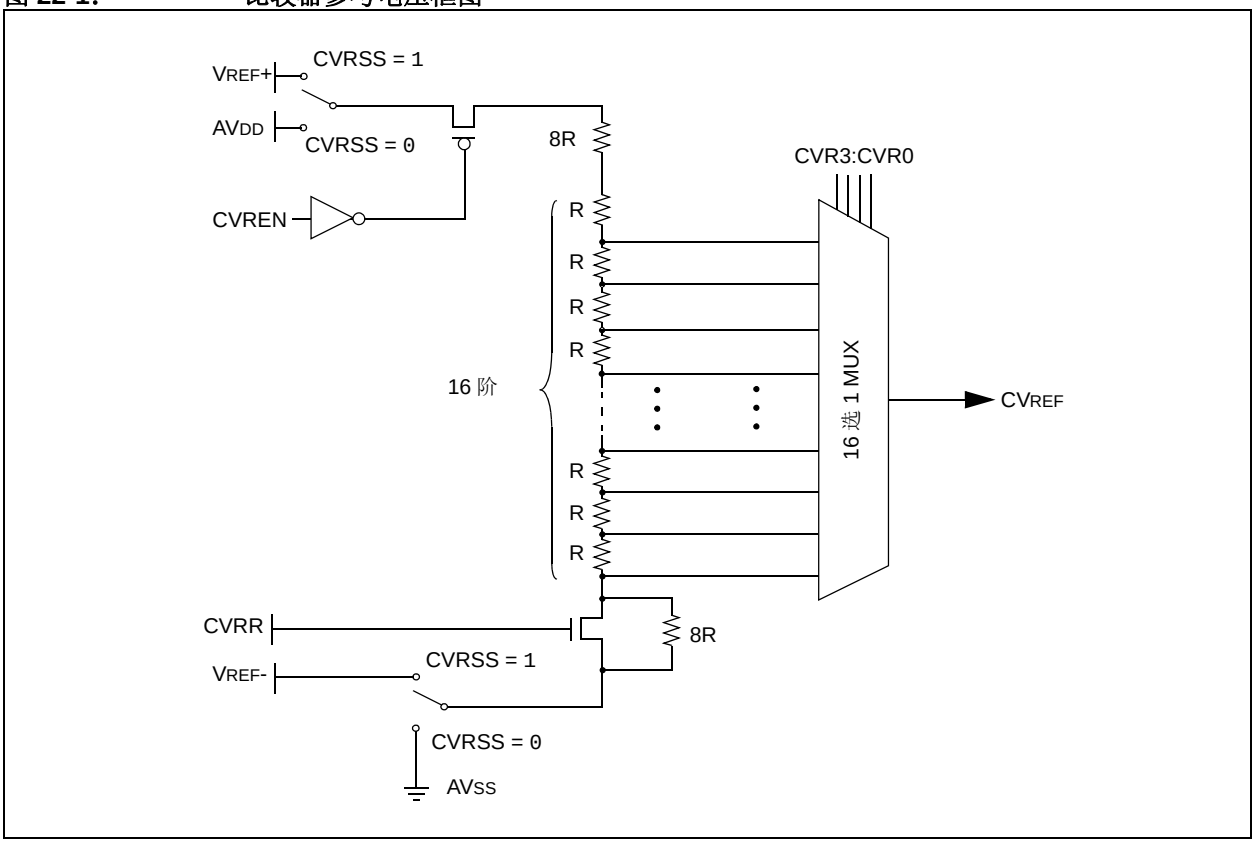
比较器参考电压模块的供电电压来自于 VDD 和 VSS，也可以来自于外部的 VREF+ 和 VREF-。使用 CVRSS 位（CVRCON<4>）选择电压源。

当改变 CVREF 输出时，要考虑比较器参考电压的稳定时间。

CVRR（比较器 VREF 范围选择位）：1 = 0 至 0.625 CVRSRC，相邻电压之间相差 CVRSRC/24

0 = 0.25 CVRSRC 至 0.72 CVRSRC，相邻电压之间相差 CVRSRC/32

图 22-1： 比较器参考电压框图



寄存器 22-1：　　CVRCON：比较器参考电压控制寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 15							bit 8

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	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

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

bit 15-8　**未实现：**读为 0

bit 7　**CVREN：**比较器参考电压使能位
1 = 启动 CVREF 电路
0 = 关闭 CVREF 电路

bit 6　**CVROE：**比较器 VREF 输出使能位
1 = CVREF 电压输出到 CVREF 引脚
0 = CVREF 电压没有连接到 CVREF 引脚

bit 5　**CVRR：**比较器 VREF 量程选择位
1 = 0 至 0.625 CVRSRC，步长为 CVRSRC/24
0 = 0.25 CVRSRC 至 0.72 CVRSRC，步长为 CVRSRC/32

bit 4　**CVRSS：**比较器 VREF 电压源选择位
1 = 比较器参考电压源 CVRSRC = VREF+ - VREF-
0 = 比较器参考电压源 CVRSRC = AVDD - AVSS

bit 3-0　**CVR3:CVR0：**比较器 VREF 值选择位 （0 ≤ CVR3:CVR0 ≤ 15）
当 CVRR = 1 时：
CVREF = (CVR<3:0>/ 24) • (CVRSRC)
当 CVRR = 0 时：
CVREF = 1/4 • (CVRSRC) + (CVR<3:0>/32) • (CVRSRC)

23.0 特殊性能

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 32 章“高级器件集成”**（DS39719A_CN）。

PIC24FJ128GA010 系列器件包含的功能旨在最大限度地提高应用的灵活性和可靠性，并通过减去外部元件把成本降到最低。这些功能包括：

- 灵活的配置
- 看门狗定时器 （WDT）
- 代码保护
- JTAG 边界扫描接口
- 在线串行编程
- 在线仿真

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 F80000h 开始的单元中。表 23-1 给出了详细的地址。从寄存器 23-1 到寄存器 23-4 详细说明了各配置位的不同功能。

注意地址 F80000h 超出了用户程序存储空间的范围。事实上，它属于配置存储空间 （800000h-FFFFFFh），这一空间仅能通过表读和表写进行访问。

23.1.1 配置 PIC24FJ128GA010 系列器件的注意事项

在 PIC24FJ128GA010 系列器件中，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的数据进行编程。配置数据存储在片上程序存储空间顶部的 2 个字中，这些字被称为闪存配置字。在表 23-1 中显示了它们的位置。这些字是实际器件配置位的紧凑表现形式，这些配置位实际上散布在配置空间的五个单元中。器件复位时，配置数据会被自动从闪存配置字装入相应的配置寄存器中。

注： 所有类型的器件复位均会重载配置数据。

表 23-1： PIC24FJ128GA010 系列器件的闪存配置字地址

器件	配置字地址	
	1	2
PIC24FJ64GA	00ABFEh	00ABFCh
PIC24FJ96GA	00FFFEh	00FFFCh
PIC24FJ128GA	0157FEh	0157FCh

当为这些器件创建应用程序时，用户就该为配置数据分配特定的闪存配置字单元。这是为了确保在编译代码时不会把程序代码存储在该地址上。

任何器件复位均会导致从闪存配置字重新装载配置位。

程序存储器中的两个闪存配置字的高字节应始终为 1111 1111。这样当这些配置字被远程事件意外执行时，被当作一条 NOP 指令。由于配置位并未真正保存在对应的单元内，因此向这些单元写 1 不会影响器件工作。

寄存器 23-1：　　闪存配置字 1

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

r-x	R/PO-1	R/PO-1	R/PO-1	R/PO-1	r-1	U-1	R/PO-1
r	JTAGEN(1)	GCP	GWRP	DEBUG	r	—	ICS
bit 15							bit 8

R/PO-1	R/PO-1	U-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1
FWDTEN	WINDIS	—	FWPSA	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

图注：			
R = 可读位	PO = 一次编程位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 23-16 **未实现：**读为 1

bit 15 **保留：**编程为 0。复位值未知。

bit 14 **JTAGEN：**JTAG 端口使能位 (1)
1 = 使能 JTAG 端口
0 = 禁止 JTAG 端口

bit 13 **GCP：**通用段程序存储器代码保护位
1 = 禁止代码保护
0 = 对整个程序存储器空间使能代码保护

bit 12 **GWRP：**通用段代码闪存写保护位
1 = 允许写入程序存储器
0 = 禁止写入程序存储器

bit 11 **$\overline{\text{DEBUG}}$：**后台调试器使能位
1 = 器件复位到工作模式
0 = 器件复位到调试模式

bit 10 **保留：**编程为 1

bit 9 **未实现：**读为 1

bit 8 **ICS：**仿真器引脚位置选择位
1 = 仿真器 / 调试器使用 EMUC2/EMUD2
0 = 仿真器 / 调试器使用 EMUC1/EMUD1

bit 7 **FWDTEN：**看门狗定时器使能位
1 = 使能看门狗定时器
0 = 禁止看门狗定时器

bit 6 **WINDIS：**带窗口的看门狗定时器禁止位
1 = 使能标准看门狗定时器
0 = 使能带窗口的看门狗定时器； FWDTEN 必须置 1

bit 5 **未实现：**读为 1

bit 4 **FWPSA：**WDT 预分频比选择位
1 = 预分频比为 1:128
0 = 预分频比为 1:32

注　1： 使用 JTAG 编程无法修改 JTAGEN 位，只能通过在线串行编程 （ICSP™）对它进行更改。

寄存器 23-1：　　闪存配置字 1 （续）

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

注　1： 使用 JTAG 编程无法修改 JTAGEN 位，只能通过在线串行编程 （ICSP™）对它进行更改。

寄存器 23-2：　　闪存配置字 2

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

R/PO-1	U-1	U-1	U-1	U-1	R/PO-1	R/PO-1	R/PO-1
IESO	—	—	—	—	FNOSC2	FNOSC1	FNOSC0
bit 15							bit 8

R/PO-1	R/PO-1	R/PO-1	U-1	U-1	U-1	R/PO-1	R/PO-1
FCKSM1	FCKSM0	OSCIOFCN	—	—	—	POSCMD1	POSCMD0
bit 7							bit 0

图注：			
R = 可读位	PO = 一次编程位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 23-16 **未实现：**读为 1

bit 15 **IESO：**内部外部切换位
1 = 使能 IESO 模式 （双速启动）
0 = 禁止 IESO 模式 （双速启动）

bit 14-11 **未实现：**读为 1

bit 10-8 **FNOSC2:FNOSC0：**初始振荡器选择位
111 = 带有后分频器的快速 RC 振荡器 （FRCDIV）
110 = 保留
101 = 低功耗 RC 振荡器 （LPRC）
100 = 辅助振荡器 （SOSC）
011 = 带有 PLL 模块的主振荡器 （XTPLL、 HSPLL 和 ECPLL）
010 = 主振荡器 （XT、 HS 和 EC）
001 = 带有后分频器和 PLL 模块的快速 RC 振荡器 （FRCPLL）
000 = 快速 RC 振荡器 （FRC）

bit 7-6 **FCKSM1:FCKSM0：**时钟切换和故障保护时钟监视器配置位
1x = 禁止时钟切换和故障保护时钟监视器
01 = 使能时钟切换，禁止故障保护时钟监视器
00 = 使能时钟切换，使能故障保护时钟监视器

bit 5 **OSCIOFCN：**OSC2 引脚配置位
如果 POSCMD1:POSCMD0 = 11 或 00：
1 = OSC2/CLKO/RC15 用作 CLKO （FOSC/2）
0 = OSC2/CLKO/RC15 用作端口 I/O （RC15）
如果 POSCMD1:POSCMD0 = 10 或 01：
OSCIOFCN 对 OSC2/CLKO/RC15 没有影响。

bit 4-2 **未实现：**读为 1

bit 1-0 **POSCMD1:POSCMD0：**主振荡器配置位
11 = 禁止主振荡器
10 = 选择 HS 振荡器模式
01 = 选择 XT 振荡器模式
00 = 选择 EC 振荡器模式

寄存器 23-3：　　DEVID：器件 ID 寄存器

U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—
bit 23							bit 16

U	U	R	R	R	R	R	R
—	—	FAMID7	FAMID6	FAMID5	FAMID4	FAMID3	FAMID2
bit 15							bit 8

R	R	R	R	R	R	R	R
FAMID1	FAMID0	DEV5	DEV4	DEV3	DEV2	DEV1	DEV0
bit 7							bit 0

图注：			
R = 可读位	PO = 一次编程位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 23-14 **未实现：**读为 0

bit 13-6 **FAMID7:FAMID0：**器件系列标识符位
00010000 = PIC24FJ128GA010 系列

bit 5-0 **DEV5:DEV0：**单个器件标识符位
000101 = PIC24FJ64GA006
000110 = PIC24FJ96GA006
000111 = PIC24FJ128GA006
001000 = PIC24FJ64GA008
001001 = PIC24FJ96GA008
001010 = PIC24FJ128GA008
001011 = PIC24FJ64GA010
001100 = PIC24FJ96GA010
001101 = PIC24FJ128GA010

寄存器 23-4：　　DEVREV：器件版本寄存器

U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—
bit 23							bit 16

R	R	R	R	U	U	U	R
r	r	r	r	—	—	—	MAJRV2
bit 15							bit 8

R	R	U	U	U	R	R	R
MAJRV1	MAJRV0	—	—	—	DOT2	DOT1	DOT0
bit 7							bit 0

图注：			
R = 可读位	PO = 一次编程位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 23-16 **未实现：**读为 0

bit 15-12 **保留：**仅供工厂使用

bit 11-9 **未实现：**读为 0

bit 8-6 **MAJRV2:MAJRV0：**主要版本标识符位

bit 5-3 **未实现：**读为 0

bit 2-0 **DOT2:DOT0：**次要版本标识符位

23.2 片内稳压器

所有的 PIC24FJ128GA010 系列 器件使用 2.5V（标称值）电压为其内核数字逻辑供电。对于需要工作在一个更高的典型电压值（如 3.3V）的设计来讲，这可能会带来问题。为简化系统设计，PIC24FJ128GA010 系列中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 引脚控制稳压器。把 VDD 连到该引脚将使能稳压器，然后稳压器通过其他 VDD 引脚向内核供电。当使能稳压器时，必须在 VDDCORE/VCAP 引脚连接低 ESR 电容（如钽电容）（见图 23-1）。这有利于保持稳压器的稳定性。**第 26.1 节“直流特性”**中提供了该滤波电容 CEFC 的推荐值。

如果 ENVREG 与 VSS 相连，则禁止稳压器。在这种情况下，必须通过 VDDCORE/VCAP 引脚对器件内核逻辑单独供标称值为 2.5V 的电压，从而使 I/O 引脚可以具有较高的电压（通常为 3.3V）。或者，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 23-1 了解可能的配置。

23.2.1 片内稳压器和 POR

使能稳压器后，需等待大约 20 µs 才能产生输出。在这段称为 TSTARTUP 的时间内，禁止代码执行。每次上电器件恢复工作时都需要 TSTARTUP，从休眠模式恢复时也是如此。

如果禁止稳压器，将自动使能上电延时定时器（PWRT）。在器件启动时，PWRT 会产生固定的 64ms（标称值）的延时。

23.2.2 片内稳压器和 BOR

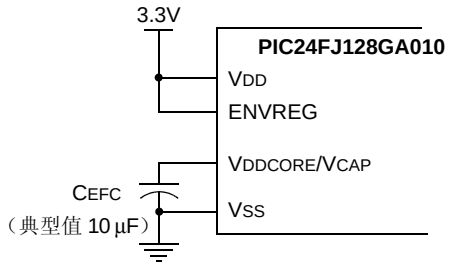
当使能片内稳压器时，PIC24FJ128GA010 系列器件也具有简单的欠压功能。如果向稳压器提供的电压不足以维持一个稳定的电平，那么稳压器复位电路将产生欠压复位。BOR 标志位（RCON<0>）可捕捉该事件。《PIC24F 系列参考手册》的“复位”章节（DS39712A_CN）规定了欠压电压值。

23.2.3 上电要求

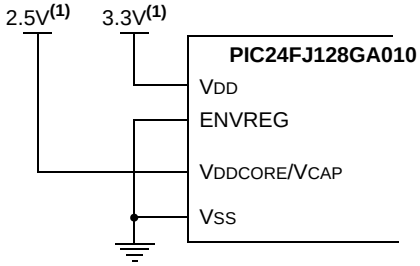
片内稳压器是为了满足器件的上电要求而设计的。如果不使用稳压器，那就必须严格遵守上电条件。在上电时，VDDCORE 决不能超出 VDD 电平 0.3V 以上。

图 23-1： 片内稳压器连接

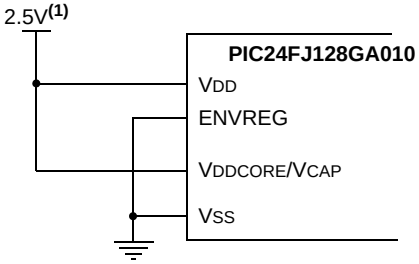
使能稳压器（ENVREG 连到 VDD）：



禁止稳压器（ENVREG 接地）：



禁止稳压器（VDD 连到 VDDCORE）：



注 1： 这些为典型的工作电压值。请参见**第 26.1 节“直流特性”**了解 VDD 和 VDDCORE 的全部工作范围。

23.3 看门狗定时器 （WDT）

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 9 章 “看门狗定时器 （WDT）”**（DS39697A_CN）。

PIC24FJ128GA010 系列器件的 WDT 是由 LPRC 振荡器驱动的。当使能 WDT 时，也将同时使能该时钟源。

WDT 的时钟源 LPRC 产生 32 kHz（标称值）的时钟信号。 将此时钟源输入到可配置为 5 位（32 分频）或 7 位（128 分频）工作模式的预分频器，具体工作模式由 FWPSA 配置位设置。 32 kHz 的输入使预分频器生成 WDT 超时周期（TWDT）—— 5 位模式为 1 ms，7 位模式为 4 ms。

可变后分频器将 WDT 预分频器的输出信号再次进行分频，从而获得更大范围的超时周期。后分频器由 WDTPS3:WDTPS0 配置位（闪存配置字 1<3:0>）控制，这样总共有 16 种设置可供选择（从 1:1 到 1:32,768）。使用预分频器和后分频器后，可获得 1 ms 到 131 秒的超时周期。

WDT、预分频器和后分频器在以下情形下会被复位：

- 任何类型的器件复位
- 在时钟切换完成时，由软件 （即改变 NOSC 位后将 OSWEN 位置 1）或硬件引起 （即故障保护时钟监视器）
- 执行 PWRSAV 指令时 （即进入休眠或空闲模式）
- 器件退出休眠或空闲模式恢复正常工作时
- 在正常执行过程中，执行 CLRWDT 指令

如果WDT被使能，它将在休眠或空闲模式下继续运行。当发生 WDT 超时时，器件将被唤醒且代码将从执行 PWRSAV 指令处继续执行。器件被唤醒后，需要使用软件将相应的 SLEEP 或 IDLE 位（RCON<3:2>）清零。

WDT 标志位 WDTO（RCON<4>）不会在 WDT 超时后自动清零。要检测后续的 WDT 事件，必须用软件将该标志清零。

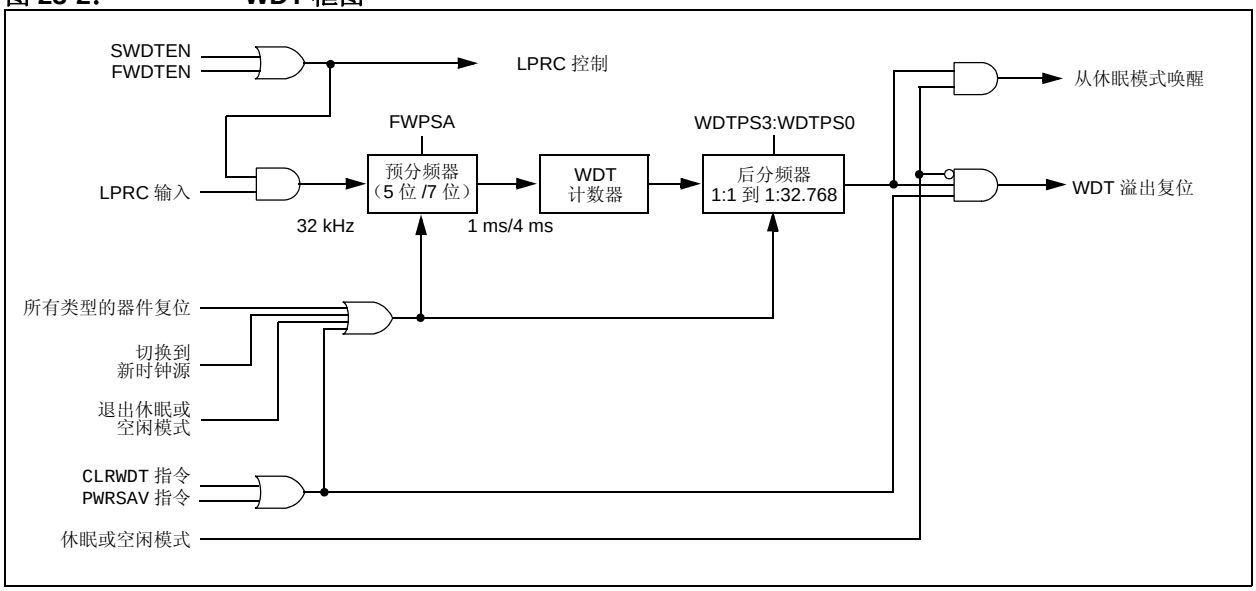
注： 当执行 CLRWDT 和 PWRSAV 指令时，预分频器和后分频器的计数值将被清零。

23.3.1 控制寄存器

使用 FWDTEN 器件配置位使能或禁止 WDT。当 FWDTEN 配置位置 1 时，使能 WDT。

FWDTEN 配置位被编程为 0 后，也可以使用软件来控制 WDT。通过在软件中将 SWDTEN 控制位（RCON<5>）置 1 使能 WDT。任何类型的器件复位都会使 SWDTEN 控制位清零。软件控制 WDT 选项允许用户在关键代码段使能 WDT 并在非关键代码段禁止 WDT，以最大限度地降低功耗。

图 23-2： WDT 框图



23.4 JTAG 接口

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 33 章“编程和诊断”**（DS39716A_CN）。

PIC24FJ128GA010 系列器件实现了 JTAG 接口，可以支持边界扫描器件测试和在线编程。

请访问 Microchip 网站（www.microchip.com）获取 JTAG 支持文件和更多信息。

23.5 程序校验和代码保护

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 33 章“编程和诊断”**（DS39716A_CN）。

对于 PIC24FJ128GA010 系列中的所有器件，片内程序存储空间被视为一个独立的存储区。配置位 GCP 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操作。但对正常的执行模式没有直接影响。

写保护受配置字中的 GWRP 位控制。将 GWRP 设置为 0，阻止对程序存储器的内部写和擦除操作。

23.5.1 保护配置寄存器

有两种方法可以保护配置寄存器使其免遭破坏性的改写或读取。主要的保护方式与影子寄存器相同，寄存器中包含预存的值，该值将不断与实际值进行比较。出于从防范不可预见事件方面的考虑，由于电池故障（如 ESD 事件）引起的配置位更改将导致奇偶校验错误并触发器件的配置字不匹配复位。

配置寄存器的数据来自于程序存储器中的闪存配置字。当 GCP 位置 1 时，也将保护器件配置的源数据。

23.6 在线串行编程

注： 本数据手册总结了该系列 PIC24F 器件的功能。但它并不是一份无所不包的参考资料。欲知更多信息，请参见《PIC24F 系列参考手册》的**第 33 章“编程和诊断”**（DS39716A_CN）。

PIC24FJ128GA010 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，包括时钟（PGCx）线和数据（PGDx）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。

23.7 在线调试器

如果选择 MPLAB® ICD 2 作为调试器，将使能在线调试功能。这一功能允许结合 MPLAB IDE 进行一些简单的调试。通过 EMUCx（仿真 / 调试时钟）和 EMUDx（仿真 / 调试数据）引脚控制调试功能。

要使用器件的在线调试功能，在设计中必须实现 MCLR、VDD、VSS、PGCx、PGDx 和 EMUDx/EMUCx 引脚对的正确连接。 此外，如果使能 ICSP 功能，某些资源将无法通用。这些资源包括数据 RAM 的前 80 个字节和两个 I/O 引脚。

注：

24.0 指令集综述

标准的 PIC24F 指令集与以前的 PIC® 指令集相比，添加了很多增强功能，并保持了易于从其他 PIC MCU 指令集移植的特点。大部分指令为单字指令，只有 3 条指令需要占用两个程序存储单元。

每条单字指令都是一个 24 位字，由一个 8 位的操作码（指明指令类型）和一个或多个操作数（指定指令具体操作）组成。整个指令集具有高度的正交性，分为以下 4 种基本类型：

- 面向字或字节的操作类指令
- 面向位的操作类指令
- 立即数操作类指令
- 控制操作类指令

表 24-1给出了说明指令时所要用到的通用符号。表 24-2 中的 PIC24F 指令集汇总列出了所有指令及每条指令影响的状态位。

大部分面向字或字节的 W 寄存器指令（包含桶形移位寄存器指令）含有三个操作数：

- 第一个源操作数通常是不带地址修改符的 “Wb” 寄存器
- 第二个源操作数通常是带或不带地址修改符的 “Ws” 寄存器。
- 结果的目标地址通常是带或不带地址修改符的 “Wd” 寄存器。

而面向字或字节的文件寄存器指令具有两个操作数：

- 文件寄存器（由 “f” 指定）
- 目标寄存器（可以是文件寄存器 “f” 也可以是记作 “WREG” 的 W0 寄存器）

大部分面向位的操作类指令（包括简单翻转/移位指令）含有两个操作数：

- W 寄存器（带或不带地址修改符）或文件寄存器（由 “Ws” 或 “f” 的值指定）
- W 寄存器或文件寄存器中的位（由立即数直接指定或由 “Wb” 寄存器中的内容间接指定）

涉及数据传送的立即数指令可能使用以下操作数：

- 将被装载到 W 寄存器或文件寄存器的立即数值（由 “k” 指定）
- 将要装载立即数的 W 寄存器或文件寄存器（由 “Wb” 或 “f” 指定）

而涉及算术或逻辑运算的立即数指令使用以下操作数：

- 第一个源操作数是不带地址修改符的 “Wb” 寄存器
- 第二个源操作数是立即数值
- 结果的目标地址（只有在与第一个源操作数不同的情况下）通常为带或不带地址修改符的 “Wd” 寄存器

控制操作类指令使用以下操作数：

- 程序存储器地址
- 表读和表写指令的模式

除某些双字指令外所有指令都是单字指令。双字指令中所有必需的信息都在低 48 位中，第二个字的高 8 位都是 0。如果第二个字作为一条指令执行，它将会被当作 NOP 指令执行。

除非条件测试结果为真或者执行后改变了程序计数器的值，否则执行所有的单字指令都只需要一个指令周期。对于上述两种特殊情况，执行指令需要两个指令周期，第二个指令周期中执行一条 NOP 指令。值得注意的特殊指令有：BRA（无条件 / 相对转移指令）、间接 CALL/GOTO指令、所有表读和表写指令以及RETURN/RETFIE 指令，这些指令都是单字指令，但执行起来需要 2 或 3 个指令周期。

某些涉及跳过下一条指令的指令，在执行跳过时需要两或三个指令周期，具体周期数取决于被跳过的指令是单字指令还是双字指令。此外需要传送两个字的指令需要两个周期。执行双字指令需要两个指令周期。

表 24-1： 说明操作码时使用的符号

字段	说明
#text	表示由 “text” 指定的立即数
(text)	表示 “text 的内容”
[text]	表示 “由 text 指定的地址单元”
{ }	可选字段或操作
<n:m>	寄存器位域
.b	字节模式选择
.d	双字模式选择
.S	影子寄存器选择
.w	字模式选择 （默认）
bit4	4 位位选择字段 （用于字寻址指令） ∈ {0...15}
C、 DC、 N、 OV 和 Z	MCU 状态位：进位、半进位、负标志、溢出标志和全零标志
Expr	绝对地址、标号或表达式 （由链接器解析）
f	文件寄存器地址 ∈ {0000h...1FFFh}
lit1	1 位无符号立即数 ∈ {0,1}
lit4	4 位无符号立即数 ∈ {0...15}
lit5	5 位无符号立即数 ∈ {0...31}
lit8	8 位无符号立即数 ∈ {0...255}
lit10	10 位无符号立即数，字节模式时 ∈ {0...255}，字模式下 ∈ {0:1023}
lit14	14 位无符号立即数 ∈ {0...16384}
lit16	16 位无符号立即数 ∈ {0...65535}
lit23	23 位无符号立即数 ∈ {0...8388608}； LSB 必须为 0
无	该字段不必有输入项，可以为空白
PC	程序计数器
Slit10	10 位有符号立即数 ∈ {-512...511}
Slit16	16 位有符号立即数 ∈ {-32768...32767}
Slit6	6 位有符号立即数 ∈ {-16...16}
Wb	基本 W 寄存器 ∈ {W0..W15}
Wd	目标 W 寄存器 ∈ { Wd, [Wd], [Wd++], [Wd--], [++Wd], [--Wd] }
Wdo	目标 W 寄存器 ∈{ Wnd, [Wnd], [Wnd++], [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] }
Wm,Wn	被除数和除数工作寄存器对 （直接寻址）
Wn	16 个工作寄存器之一 ∈ {W0..W15}
Wnd	16 个目标工作寄存器之一 ∈ {W0..W15}
Wns	16 个源工作寄存器之一 ∈ {W0..W15}
WREG	W0 （文件寄存器指令中的工作寄存器）
Ws	源 W 寄存器 ∈ { Ws, [Ws], [Ws++], [Ws--], [++Ws], [--Ws] }
Wso	源 W 寄存器 ∈ { Wns, [Wns], [Wns++], [Wns--], [++Wns], [--Wns], [Wns+Wb] }

表 24-2：　　指令集汇总

汇编指令助记符	汇编语法		说明	字数	周期数	受影响的标志位
ADD	ADD	f	f = f + WREG	1	1	C、DC、N、OV 和 Z
	ADD	f,WREG	WREG = f + WREG	1	1	C、DC、N、OV 和 Z
	ADD	#lit10,Wn	Wd = lit10 + Wd	1	1	C、DC、N、OV 和 Z
	ADD	Wb,Ws,Wd	Wd = Wb + Ws	1	1	C、DC、N、OV 和 Z
	ADD	Wb,#lit5,Wd	Wd = Wb + lit5	1	1	C、DC、N、OV 和 Z
ADDC	ADDC	f	f = f + WREG + (C)	1	1	C、DC、N、OV 和 Z
	ADDC	f,WREG	WREG = f + WREG + (C)	1	1	C、DC、N、OV 和 Z
	ADDC	#lit10,Wn	Wd = lit10 + Wd + (C)	1	1	C、DC、N、OV 和 Z
	ADDC	Wb,Ws,Wd	Wd = Wb + Ws + (C)	1	1	C、DC、N、OV 和 Z
	ADDC	Wb,#lit5,Wd	Wd = Wb + lit5 + (C)	1	1	C、DC、N、OV 和 Z
AND	AND	f	f = f .AND.WREG	1	1	N 和 Z
	AND	f,WREG	WREG = f .AND.WREG	1	1	N 和 Z
	AND	#lit10,Wn	Wd = lit10 .AND. Wd	1	1	N 和 Z
	AND	Wb,Ws,Wd	Wd = Wb .AND. Ws	1	1	N 和 Z
	AND	Wb,#lit5,Wd	Wd = Wb .AND. lit5	1	1	N 和 Z
ASR	ASR	f	f = 算术右移 f	1	1	C、 N、 OV 和 Z
	ASR	f,WREG	WREG = 算术右移 f	1	1	C、 N、 OV 和 Z
	ASR	Ws,Wd	Wd = 算术右移 Ws	1	1	C、 N、 OV 和 Z
	ASR	Wb,Wns,Wnd	Wnd = 将 Wb 算术右移 Wns 位	1	1	N 和 Z
	ASR	Wb,#lit5,Wnd	Wnd = 将 Wb 算术右移 lit5 位	1	1	N 和 Z
BCLR	BCLR	f,#bit4	将 f 寄存器中的某位清零	1	1	无
	BCLR	Ws,#bit4	将 Ws 中的某位清零	1	1	无
BRA	BRA	C,Expr	进位则跳转	1	1 (2)	无
	BRA	GE,Expr	如果大于等于则跳转	1	1 (2)	无
	BRA	GEU,Expr	如果无符号大于等于则跳转	1	1 (2)	无
	BRA	GT,Expr	如果大于则跳转	1	1 (2)	无
	BRA	GTU,Expr	如果无符号大于则跳转	1	1 (2)	无
	BRA	LE,Expr	如果小于等于则跳转	1	1 (2)	无
	BRA	LEU,Expr	如果无符号小于等于则跳转	1	1 (2)	无
	BRA	LT,Expr	如果小于则跳转	1	1 (2)	无
	BRA	LTU,Expr	如果无符号小于则跳转	1	1 (2)	无
	BRA	N,Expr	为负则跳转	1	1 (2)	无
	BRA	NC,Expr	无进位则跳转	1	1 (2)	无
	BRA	NN,Expr	不为负则跳转	1	1 (2)	无
	BRA	NOV,Expr	不溢出则跳转	1	1 (2)	无
	BRA	NZ,Expr	不为零则跳转	1	1 (2)	无
	BRA	OV,Expr	溢出则跳转	1	1 (2)	无
	BRA	Expr	无条件跳转	1	2	无
	BRA	Z,Expr	为零则跳转	1	1 (2)	无
	BRA	Wn	相对跳转	1	2	无
BSET	BSET	f,#bit4	将 f 寄存器中的某位置 1	1	1	无
	BSET	Ws,#bit4	将 Ws 中的某位置 1	1	1	无
BSW	BSW.C	Ws,Wb	将 C 位写入 Ws<Wb>	1	1	无
	BSW.Z	Ws,Wb	将 Z 位写入 Ws<Wb>	1	1	无
BTG	BTG	f,#bit4	将 f 中的某位取反	1	1	无
	BTG	Ws,#bit4	将 Ws 中的某位取反	1	1	无
BTSC	BTSC	f,#bit4	检测 f 中的某位，为 0 则跳过	1	1 (2 或 3)	无
	BTSC	Ws,#bit4	测试 Ws 中的某位，为 0 则跳过	1	1 (2 或 3)	无

表 24-2： 指令集汇总 （续）

汇编指令助记符	汇编语法		说明	字数	周期数	受影响的标志位
BTSS	BTSS	f,#bit4	检测 f 中的某位，为 1 则跳过	1	1 (2 或 3)	无
	BTSS	Ws,#bit4	检测 Ws 中的某位，为 1 则跳过	1	1 (2 或 3)	无
BTST	BTST	f,#bit4	检测 f 中的某位	1	1	Z
	BTST.C	Ws,#bit4	检测 Ws 中的位并将结果存储到 C	1	1	C
	BTST.Z	Ws,#bit4	检测 Ws 中的位并将结果存储到 Z	1	1	Z
	BTST.C	Ws,Wb	检测 Ws<Wb> 并将结果存储到 C	1	1	C
	BTST.Z	Ws,Wb	检测 Ws<Wb> 并将结果存储到 Z	1	1	Z
BTSTS	BTSTS	f,#bit4	检测 f 寄存器中的位并将该位置 1	1	1	Z
	BTSTS.C	Ws,#bit4	检测 Ws 中的位并将结果存储到 C，然后将被检测位置 1	1	1	C
	BTSTS.Z	Ws,#bit4	检测 Ws 中的位并将结果存储到 Z，然后将被检测位置 1	1	1	Z
CALL	CALL	lit23	调用子程序	2	2	无
	CALL	Wn	间接调用子程序	1	2	无
CLR	CLR	f	f = 0x0000	1	1	无
	CLR	WREG	WREG = 0x0000	1	1	无
	CLR	Ws	Ws = 0x0000	1	1	无
CLRWDT	CLRWDT		将看门狗定时器清零	1	1	WDTO，休眠
COM	COM	f	$f = \overline{f}$	1	1	N 和 Z
	COM	f,WREG	$WREG = \overline{f}$	1	1	N 和 Z
	COM	Ws,Wd	$Wd = \overline{Ws}$	1	1	N 和 Z
CP	CP	f	将 f 寄存器与 WREG 作比较	1	1	C、DC、N、OV 和 Z
	CP	Wb,#lit5	将 Wb 与 lit5 作比较	1	1	C、DC、N、OV 和 Z
	CP	Wb,Ws	将 Wb 与 Ws 作比较	1	1	C、DC、N、OV 和 Z
CP0	CP0	f	将 f 寄存器与 0x0000 作比较	1	1	C、DC、N、OV 和 Z
	CP0	Ws	将 Ws 寄存器与 0x0000 作比较	1	1	C、DC、N、OV 和 Z
CPB	CPB	f	将 f 寄存器与 WREG 作比较 （通过带借位减法实现）	1	1	C、DC、N、OV 和 Z
	CPB	Wb,#lit5	将 Wb 与 lit5 作比较 （通过带借位减法实现）	1	1	C、DC、N、OV 和 Z
	CPB	Wb,Ws	将 Wb 与 Ws 作比较 （通过带借位减法实现） （Wb - Ws - $\overline{C}$）	1	1	C、DC、N、OV 和 Z
CPSEQ	CPSEQ	Wb,Wn	将 Wb 与 Wn 作比较，如果相等则跳过	1	1 (2 或 3)	无
CPSGT	CPSGT	Wb,Wn	将 Wb 与 Wn 作比较，如果大于则跳过	1	1 (2 或 3)	无
CPSLT	CPSLT	Wb,Wn	将 Wb 与 Wn 作比较，如果小于则跳过	1	1 (2 或 3)	无
CPSNE	CPSNE	Wb,Wn	将 Wb 与 Wn 作比较，如果不等则跳过	1	1 (2 或 3)	无
DAW	DAW	Wn	Wn = 对 Wn 进行十进制调整	1	1	C
DEC	DEC	f	f = f - 1	1	1	C、DC、N、OV 和 Z
	DEC	f,WREG	WREG = f - 1	1	1	C、DC、N、OV 和 Z
	DEC	Ws,Wd	Wd = Ws - 1	1	1	C、DC、N、OV 和 Z
DEC2	DEC2	f	f = f - 2	1	1	C、DC、N、OV 和 Z
	DEC2	f,WREG	WREG = f - 2	1	1	C、DC、N、OV 和 Z
	DEC2	Ws,Wd	Wd = Ws - 2	1	1	C、DC、N、OV 和 Z
DISI	DISI	#lit14	在 k 个指令周期内禁止中断	1	1	无
DIV	DIV.SW	Wm,Wn	有符号 16/16 位整数除法	1	18	N、 Z、 C 和 OV
	DIV.SD	Wm,Wn	有符号 32/16 位整数除法	1	18	N、 Z、 C 和 OV
	DIV.UW	Wm,Wn	无符号 16/16 位整数除法	1	18	N、 Z、 C 和 OV
	DIV.UD	Wm,Wn	无符号 32/16 位整数除法	1	18	N、 Z、 C 和 OV
EXCH	EXCH	Wns,Wnd	将 Wns 和 Wnd 交换	1	1	无
FF1L	FF1L	Ws,Wnd	从左边第一位 （MSb）开始查找	1	1	C
FF1R	FF1R	Ws,Wnd	从右边第一位 （LSb）开始查找	1	1	C

表 24-2：　　指令集汇总 （续）

汇编指令助记符	汇编语法		说明	字数	周期数	受影响的标志位
GOTO	GOTO	Expr	跳转到指定地址	2	2	无
	GOTO	Wn	跳转到直接地址处	1	2	无
INC	INC	f	f = f + 1	1	1	C、DC、N、OV 和 Z
	INC	f,WREG	WREG = f + 1	1	1	C、DC、N、OV 和 Z
	INC	Ws,Wd	Wd = Ws + 1	1	1	C、DC、N、OV 和 Z
INC2	INC2	f	f = f + 2	1	1	C、DC、N、OV 和 Z
	INC2	f,WREG	WREG = f + 2	1	1	C、DC、N、OV 和 Z
	INC2	Ws,Wd	Wd = Ws + 2	1	1	C、DC、N、OV 和 Z
IOR	IOR	f	f = f .IOR.WREG	1	1	N 和 Z
	IOR	f,WREG	WREG = f .IOR.WREG	1	1	N 和 Z
	IOR	#lit10,Wn	Wd = lit10 .IOR. Wd	1	1	N 和 Z
	IOR	Wb,Ws,Wd	Wd = Wb .IOR. Ws	1	1	N 和 Z
	IOR	Wb,#lit5,Wd	Wd = Wb .IOR. lit5	1	1	N 和 Z
LNK	LNK	#lit14	链接帧指针	1	1	无
LSR	LSR	f	f = 逻辑右移 f	1	1	C、 N、 OV 和 Z
	LSR	f,WREG	WREG = 逻辑右移 f	1	1	C、 N、 OV 和 Z
	LSR	Ws,Wd	Wd = 逻辑右移 Ws	1	1	C、 N、 OV 和 Z
	LSR	Wb,Wns,Wnd	Wnd = 将 Wb 逻辑右移 Wns 位	1	1	N 和 Z
	LSR	Wb,#lit5,Wnd	Wnd = 将 Wb 逻辑右移 lit5 位	1	1	N 和 Z
MOV	MOV	f,Wn	将 f 寄存器的内容传送给 Wn	1	1	无
	MOV	[Wns+Slit10],Wnd	将 [Wns+Slit10] 地址单元的内容传送给 Wnd	1	1	无
	MOV	f	将源寄存器的内容送入目标寄存器	1	1	N 和 Z
	MOV	f,WREG	将 f 寄存器的内容传送给 WREG	1	1	N 和 Z
	MOV	#lit16,Wn	将 16 位立即数传送给 Wn	1	1	无
	MOV.b	#lit8,Wn	将 8 位立即数传送给 Wn	1	1	无
	MOV	Wn,f	将 Wn 的内容传送给 f 寄存器	1	1	无
	MOV	Wns,[Wns+Slit10]	将 Wns 传送给 [Wns+Slit10] 地址单元	1	1	
	MOV	Wso,Wdo	将 Ws 的内容传送给 Wd	1	1	无
	MOV	WREG,f	将 WREG 传送给 f	1	1	N 和 Z
	MOV.D	Wns,Wd	从 W(ns):W(ns+1) 传送双字给 Wd	1	2	无
	MOV.D	Ws,Wnd	从 Ws 传送双字给 W(nd+1):W(nd)	1	2	无
MUL	MUL.SS	Wb,Ws,Wnd	{Wnd+1, Wnd} = Signed(Wb) * Signed(Ws)	1	1	无
	MUL.SU	Wb,Ws,Wnd	{Wnd+1, Wnd} = Signed(Wb) * Unsigned(Ws)	1	1	无
	MUL.US	Wb,Ws,Wnd	{Wnd+1, Wnd} = Unsigned(Wb) * Signed(Ws)	1	1	无
	MUL.UU	Wb,Ws,Wnd	{Wnd+1, Wnd} = Unsigned(Wb) * Unsigned(Ws)	1	1	无
	MUL.SU	Wb,#lit5,Wnd	{Wnd+1, Wnd} = Signed(Wb) * Unsigned(lit5)	1	1	无
	MUL.UU	Wb,#lit5,Wnd	{Wnd+1, Wnd} = Unsigned(Wb) * Unsigned(lit5)	1	1	无
	MUL	f	W3:W2 = f * WREG	1	1	无
NEG	NEG	f	f = $\bar{f}$ + 1	1	1	C、DC、N、OV 和 Z
	NEG	f,WREG	WREG = $\bar{f}$ + 1	1	1	C、DC、N、OV 和 Z
	NEG	Ws,Wd	Wd = $\overline{Ws}$ + 1	1	1	C、DC、N、OV 和 Z
NOP	NOP		空操作	1	1	无
	NOPR		空操作	1	1	无
POP	POP	f	将 f 寄存器的内容从栈顶 （TOS）弹出	1	1	无
	POP	Wdo	将栈顶 （TOS）的内容弹出到 Wdo 中	1	1	无
	POP.D	Wnd	将栈顶 （TOS）的内容弹出到 W(nd):W(nd+1) 中	1	2	无
	POP.S		将影子寄存器中的内容弹出	1	1	全部
PUSH	PUSH	f	将 f 寄存器的内容压入栈顶 （TOS）	1	1	无
	PUSH	Wso	将 Wso 的内容压入栈顶 （TOS）	1	1	无
	PUSH.D	Wns	将 W(ns):W(ns+1) 压入栈顶 （TOS）	1	2	无
	PUSH.S		压入影子寄存器	1	1	无

表 24-2： 指令集汇总 （续）

汇编指令助记符	汇编语法		说明	字数	周期数	受影响的标志位
PWRSAV	PWRSAV	#lit1	进入休眠或空闲模式	1	1	WDTO，休眠
RCALL	RCALL	Expr	相对调用	1	2	无
	RCALL	Wn	计算调用	1	2	无
REPEAT	REPEAT	#lit14	将下一条指令重复 lit14 + 1 次	1	1	无
	REPEAT	Wn	将下一条指令重复 (Wn) + 1 次	1	1	无
RESET	RESET		用软件使器件复位	1	1	无
RETFIE	RETFIE		从中断返回	1	3 (2)	无
RETLW	RETLW	#lit10,Wn	返回并将立即数存入 Wn	1	3 (2)	无
RETURN	RETURN		从子程序返回	1	3 (2)	无
RLC	RLC	f	f = 对 f 执行带进位循环左移	1	1	C、 N 和 Z
	RLC	f,WREG	WREG = 对 f 执行带进位循环左移	1	1	C、 N 和 Z
	RLC	Ws,Wd	Wd = 对 Ws 执行带进位循环左移	1	1	C、 N 和 Z
RLNC	RLNC	f	f = 将 f 循环左移 （不带进位）	1	1	N 和 Z
	RLNC	f,WREG	WREG = 将 f 循环左移 （不带进位）	1	1	N 和 Z
	RLNC	Ws,Wd	Wd = 将 Ws 循环左移 （不带进位）	1	1	N 和 Z
RRC	RRC	f	f = 对 f 执行带进位循环右移	1	1	C、 N 和 Z
	RRC	f,WREG	WREG = 对 f 执行带进位循环右移	1	1	C、 N 和 Z
	RRC	Ws,Wd	Wd = 对 Ws 执行带进位循环右移	1	1	C、 N 和 Z
RRNC	RRNC	f	f = 将 f 循环右移 （不带进位）	1	1	N 和 Z
	RRNC	f,WREG	WREG = 将 f 循环右移 （不带进位）	1	1	N 和 Z
	RRNC	Ws,Wd	Wd = 将 Ws 循环右移 （不带进位）	1	1	N 和 Z
SE	SE	Ws,Wnd	Wnd = 对 Ws 进行符号扩展	1	1	C、 N 和 Z
SETM	SETM	f	f = FFFFh	1	1	无
	SETM	WREG	WREG = FFFFh	1	1	无
	SETM	Ws	Ws = FFFFh	1	1	无
SL	SL	f	f = 左移 f	1	1	C、 N、 OV 和 Z
	SL	f,WREG	WREG = 左移 f	1	1	C、 N、 OV 和 Z
	SL	Ws,Wd	Wd = 左移 Ws	1	1	C、 N、 OV 和 Z
	SL	Wb,Wns,Wnd	Wnd = 将 Wb 左移 Wns 位	1	1	N 和 Z
	SL	Wb,#lit5,Wnd	Wnd = 将 Wb 左移 lit5 位	1	1	N 和 Z
SUB	SUB	f	f = f - WREG	1	1	C、DC、N、OV 和 Z
	SUB	f,WREG	WREG = f - WREG	1	1	C、DC、N、OV 和 Z
	SUB	#lit10,Wn	Wn = Wn - lit10	1	1	C、DC、N、OV 和 Z
	SUB	Wb,Ws,Wd	Wd = Wb - Ws	1	1	C、DC、N、OV 和 Z
	SUB	Wb,#lit5,Wd	Wd = Wb - lit5	1	1	C、DC、N、OV 和 Z
SUBB	SUBB	f	f = f - WREG - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
	SUBB	f,WREG	WREG = f - WREG - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
	SUBB	#lit10,Wn	Wn = Wn - lit10 - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
	SUBB	Wb,Ws,Wd	Wd = Wb - Ws - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
	SUBB	Wb,#lit5,Wd	Wd = Wb - lit5 - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
SUBR	SUBR	f	f = WREG - f	1	1	C、DC、N、OV 和 Z
	SUBR	f,WREG	WREG = WREG - f	1	1	C、DC、N、OV 和 Z
	SUBR	Wb,Ws,Wd	Wd = Ws - Wb	1	1	C、DC、N、OV 和 Z
	SUBR	Wb,#lit5,Wd	Wd = lit5 - Wb	1	1	C、DC、N、OV 和 Z
SUBBR	SUBBR	f	f = WREG - f - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
	SUBBR	f,WREG	WREG = WREG - f - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
	SUBBR	Wb,Ws,Wd	Wd = Ws - Wb - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
	SUBBR	Wb,#lit5,Wd	Wd = lit5 - Wb - ($\overline{C}$)	1	1	C、DC、N、OV 和 Z
SWAP	SWAP.b	Wn	Wn = 将 Wn 的两个半字节相交换	1	1	无
	SWAP	Wn	Wn = 将 Wn 的两个字节相交换	1	1	无
TBLRDH	TBLRDH	Ws,Wd	将程序计数器 PC<23:16> 读出到 Wd<7:0> 中	1	2	无

表 24-2：　　指令集汇总 （续）

汇编指令助记符	汇编语法		说明	字数	周期数	受影响的标志位
TBLRDL	TBLRDL	Ws,Wd	将程序计数器 PC<15:0> 读出到 Wd 中	1	2	无
TBLWTH	TBLWTH	Ws,Wd	将 Ws<7:0> 写入程序计数器 PC<23:16>	1	2	无
TBLWTL	TBLWTL	Ws,Wd	将 Ws 写入程序计数器 PC<15:0>	1	2	无
ULNK	ULNK		断开与帧指针的链接	1	1	无
XOR	XOR	f	f = f .XOR.WREG	1	1	N 和 Z
	XOR	f,WREG	WREG = f .XOR.WREG	1	1	N 和 Z
	XOR	#lit10,Wn	Wd = lit10 .XOR. Wd	1	1	N 和 Z
	XOR	Wb,Ws,Wd	Wd = Wb .XOR. Ws	1	1	N 和 Z
	XOR	Wb,#lit5,Wd	Wd = Wb .XOR. lit5	1	1	N 和 Z
ZE	ZE	Ws,Wnd	Wnd = 对 Ws 进行零扩展	1	1	C、 N 和 Z

注：

25.0 开发支持

一系列软件及硬件开发工具对 PIC® 单片机和 dsPIC® 数字信号控制器提供支持：

- 集成开发环境
  - MPLAB® IDE 软件
- 编译器 / 汇编器 / 链接器
  - 适用于各种器件系列的 MPLAB C 编译器
  - 适用于各种器件系列的 HI-TECH C 编译器
  - MPASM™ 汇编器
  - MPLINK™ 目标链接器 / MPLIB™ 目标库管理器
  - 适用于各种器件系列的 MPLAB 汇编器 / 链接器 / 库管理器
- 模拟器
  - MPLAB SIM 软件模拟器
- 仿真器
  - MPLAB REAL ICE™ 在线仿真器
- 在线调试器
  - MPLAB ICD 3
  - PICkit™ 3 Debug Express
- 器件编程器
  - PICkit™ 2 编程器
  - MPLAB PM3 器件编程器
- 低成本演示 / 开发板、评估工具包及入门工具包

25.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16/32 位单片机市场提供了前所未有的易于使用的软件开发平台。 MPLAB IDE 是基于 Windows® 操作系统的应用软件，包括：

- 一个包含所有调试工具的图形界面
  - 模拟器
  - 编程器 （单独销售）
  - 在线仿真器 （单独销售）
  - 在线调试器 （单独销售）
- 具有彩色上下文代码显示的全功能编辑器
- 多项目管理器
- 内容可直接编辑的可定制式数据窗口
- 高级源代码调试
- 鼠标停留在变量上进行查看的功能
- 将变量从源代码窗口拖放到 Watch （观察）窗口
- 丰富的在线帮助
- 集成了可选的第三方工具，如 IAR C 编译器

MPLAB IDE 可以让您：

- 编辑源文件 （C 语言或汇编语言）
- 点击一次即可完成编译或汇编，并将代码下载到仿真器和模拟器工具中 （自动更新所有项目信息）
- 可使用如下各项进行调试：
  - 源文件 （C 语言或汇编语言）
  - 混合 C 语言和汇编语言
  - 机器码

MPLAB IDE 在单个开发范例中支持使用多种调试工具，包括从成本效益高的模拟器到低成本的在线调试器，再到全功能的仿真器。这样缩短了用户升级到更加灵活而功能强大的工具时的学习时间。

25.2 适用于各种器件系列的 MPLAB C 编译器

MPLAB C 编译器代码开发系统是完全的 ANSI C 编译器，适用于 Microchip 的 PIC18、PIC24 和 PIC32 系列单片机及 dsPIC30 和 dsPIC33 系列数字信号控制器。这些编译器提供强大的集成功能和出众的代码优化能力，且使用方便。

为便于源代码调试，编译器提供针对 MPLAB IDE 调试器优化的符号信息。

25.3 适用于各种器件系列的 HI-TECH C 编译器

HI-TECH C 编译器代码开发系统是完全的 ANSI C 编译器，适用于 Microchip 的 PIC 系列单片机及 dsPIC 系列数字信号控制器。这些编译器提供强大的集成功能和全知代码生成能力，且使用方便。

为便于源代码调试，编译器提供针对 MPLAB IDE 调试器优化的符号信息。

编译器包括一个宏汇编器、链接器、预处理程序和单步驱动程序，可以在多种平台上运行。

25.4 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于 PIC10/12/16/18 MCU。

MPASM 汇编器可生成用于 MPLINK 目标链接器的可重定位目标文件、Intel® 标准 HEX 文件、详细描述存储器使用状况和符号参考的 MAP 文件、包含源代码行及生成机器码的绝对 LST 文件以及用于调试的 COFF 文件。

MPASM 汇编器具有如下特性：

- 集成在 MPLAB IDE 项目中
- 用户定义的宏可简化汇编代码
- 对多用途源文件进行条件汇编
- 允许完全控制汇编过程的指令

25.5 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器包含了由 MPASM 汇编器、MPLAB C18 C 编译器产生的可重定位目标。通过使用链接器脚本中的指令，它还可链接预编译库中的可重定位目标。

MPLIB目标库管理器管理预编译代码库文件的创建和修改。当从源文件调用库中的一段子程序时，只有包含此子程序的模块被链接到应用程序。这样可使大型库在许多不同应用中被高效地利用。

目标链接器 / 库管理器具有如下特性：

- 高效地连接单个的库而不是许多小文件
- 通过将相关的模块组合在一起来增强代码的可维护性
- 只要列出、替换、删除和抽取模块，便可灵活地创建库

25.6 适用于各种器件系列的 MPLAB 汇编器、链接器和库管理器

MPLAB 汇编器为 PIC24、PIC32 和 dsPIC 器件从符号汇编语言生成可重定位机器码。 MPLAB C 编译器使用该汇编器生成目标文件。汇编器产生可重定位目标文件之后，可将这些目标文件存档，或与其他可重定位目标文件和存档链接以生成可执行文件。该汇编器有如下显著特性：

- 支持整个器件指令集
- 支持定点数据和浮点数据
- 命令行界面
- 丰富的指令集
- 灵活的宏语言
- MPLAB IDE 兼容性

25.7 MPLAB SIM 软件模拟器

MPLAB SIM 软件模拟器通过在指令级对 PIC MCU 和 dsPIC® DSC 进行模拟，可在 PC 主机环境下进行代码开发。对于任何给定的指令，都可以对数据区进行检查或修改，并通过一个全面的激励控制器来施加激励。可以将各寄存器记录在文件中，以便进行进一步的运行时分析。跟踪缓冲区和逻辑分析器的显示使软件模拟器还能记录和跟踪程序的执行、 I/O 的动作、大部分的外设及内部寄存器。

MPLAB SIM 软件模拟器完全支持使用 MPLAB C 编译器以及 MPASM 和 MPLAB 汇编器的符号调试。该软件模拟器可用于在硬件实验室环境外灵活地开发和调试代码，是一款完美且经济的软件开发工具。

25.8 MPLAB REAL ICE 在线仿真器系统

MPLAB REAL ICE 在线仿真器系统是 Microchip 针对其闪存 DSC 和 MCU 器件而推出的新一代高速仿真器。结合 MPLAB 集成开发环境 （IDE）所具有的易于使用且功能强大的图形用户界面，该仿真器可对 PIC® 闪存 MCU 和 dsPIC® 闪存 DSC 进行调试和编程。IDE 是随每个工具包一起提供的。

该仿真器通过高速 USB 2.0 接口与设计工程师的 PC 相连，并利用与在线调试器系统兼容的连接器（RJ11）或新型抗噪声、高速低压差分信号 （LVDS）互连电缆（CAT5）与目标板相连。

可通过 MPLAB IDE 下载将来版本的固件，对该仿真器进行现场升级。在即将推出的 MPLAB IDE 版本中，会支持许多新器件，还将增加一些新特性。在同类仿真器中， MPLAB REAL ICE 的优势十分明显：低成本、全速仿真、运行时变量查看、跟踪分析、复杂断点、耐用的探针接口及较长 （长达 3 米）的互连电缆。

25.9 MPLAB ICD 3 在线调试器系统

MPLAB ICD 3 在线调试器系统是 Microchip 成本效益最高的高速硬件调试器 / 编程器，适用于 Microchip 闪存数字信号控制器 （DSC）和单片机 （MCU）器件。结合 MPLAB 集成开发环境 （IDE）所具有的功能强大但易于使用的图形用户界面，该调试器可对 PIC® 闪存单片机和 dsPIC® DSC 进行调试和编程。

MPLAB ICD 3 在线调试器通过高速 USB 2.0 接口与设计工程师的PC相连，并利用与MPLAB ICD 2或MPLAB REAL ICE 系统兼容的连接器（RJ-11）与目标板相连。MPLAB ICD 3 支持所有 MPLAB ICD 2 转接器。

25.10 PICkit 3 在线调试器 / 编程器及 PICkit 3 Debug Express

结合 MPLAB 集成开发环境 （IDE）所具有的功能强大的图形用户界面，MPLAB PICkit 3 可对 PIC® 闪存单片机和 dsPIC® 数字信号控制器进行调试和编程，且价位较低。MPLAB PICkit 3 通过全速 USB 接口与设计工程师的 PC 相连，并利用 Microchip 调试（RJ-11）连接器（与 MPLAB ICD 3 和 MPLAB REAL ICE 兼容）与目标板相连。连接器使用两个器件 I/O 引脚和复位线来实现在线调试和在线串行编程。

PICkit 3 Debug Express 包括 PICkit 3、演示板和单片机、连接电缆和光盘（内含用户指南、课程、教程、编译器和 MPLAB IDE 软件）。

25.11 PICkit 2 开发编程器 / 调试器及 PICkit 2 Debug Express

PICkit™ 2 开发编程器 / 调试器是一款低成本开发工具，具有易于使用的界面，适用于对 Microchip 的闪存系列单片机进行编程和调试。这一全功能的 Windows® 编程界面支持低档（PIC10F、PIC12F5xx 和 PIC16F5xx）、中档 （PIC12F6xx 和 PIC16F）、 PIC18F、 PIC24F、dsPIC30、dsPIC33 和 PIC32 系列的 8 位、16 位及 32 位单片机，以及许多 Microchip 串行 EEPROM 产品。结合 Microchip 功能强大的 MPLAB 集成开发环境（IDE），PICkit 2 可对大多数 PIC® 单片机进行在线调试。即使 PIC 单片机已嵌入应用，在线调试功能仍可以运行、暂停和单步执行程序。在断点处暂停时，可以检查和修改文件寄存器。

PICkit 2 Debug Express 包括 PICkit 2、演示板和单片机、连接电缆和光盘（内含用户指南、课程、教程、编译器和 MPLAB IDE 软件）。

25.12 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款符合 CE 规范的通用器件编程器，在 VDDMIN 和 VDDMAX 点对其可编程电压进行校验以确保可靠性最高。它有一个用来显示菜单和错误消息的大 LCD 显示器（128 x 64），以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 ICSP™ 电缆。在单机模式下，MPLAB PM3 器件编程器不必与 PC 相连即可对 PIC 器件进行读取、校验和编程。在该模式下它还可设置代码保护。MPLAB PM3 通过 RS-232 或 USB 电缆连接到 PC 主机上。MPLAB PM3 具备高速通信能力以及优化算法，可对具有大存储器的器件进行快速编程。它还包含了 MMC 卡，用于文件存储及数据应用。

25.13 演示 / 开发板、评估工具包及入门工具包

有许多演示、开发和评估板可用于各种 PIC MCU 和 dsPIC DSC，实现对全功能系统的快速应用开发。大多数的演示、开发和评估板都有实验布线区，供用户添加定制电路；还有应用固件和源代码，用于检查和修改。

这些板支持多种功能部件，包括 LED、温度传感器、开关、扬声器、 RS-232 接口、 LCD 显示器、电位计和附加 EEPROM 存储器。

演示和开发板可用于教学环境，在实验布线区设计定制电路，从而掌握各种单片机应用。

除了 PICDEM™ 和 dsPICDEM™ 演示 / 开发板系列电路外，Microchip 还有一系列评估工具包和演示软件，适用于模拟滤波器设计、KEELOQ® 数据安全产品 IC、CAN、IrDA®、 PowerSmart 电池管理、 SEEVAL® 评估系统、Σ–Δ ADC、流速传感器，等等。

同时还提供入门工具包，其中包含体验指定器件功能所需的所有软硬件。通常提供单个应用以及调试功能，都包含在一块电路板上。

有关演示、开发和评估工具包的完整列表，请访问 Microchip 网站 （www.microchip.com）。

26.0 电气特性

本节概述了 PIC24FJ128GA010 系列的电气特性。在文档后续版本中会添加其他信息。

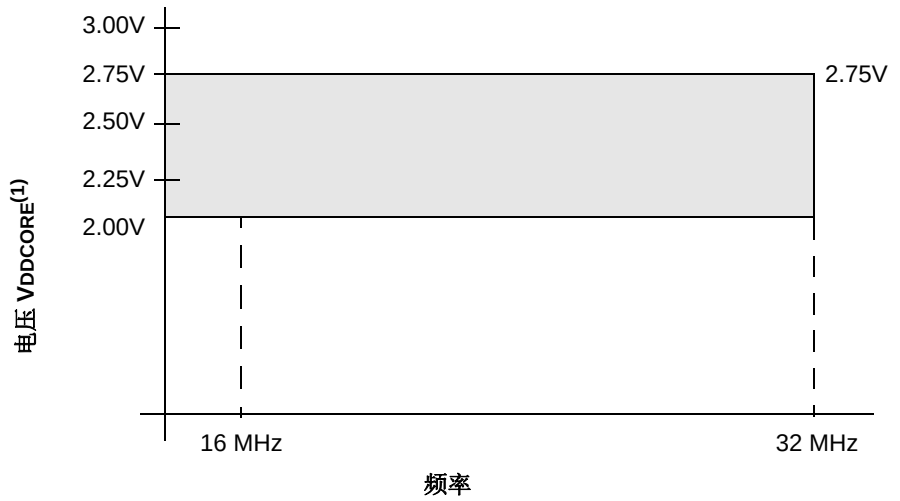
PIC24FJ128GA010 系列的绝对最大值如下所列。器件长时间工作在最大值条件下，其稳定性会受到影响。我们建议不要使器件在该规范规定的参数范围外工作。

绝对最大值 (†)

环境温度..... -40°C 至 +85°C
储存温度..... -65°C 至 +150°C
VDD 相对于 VSS 的电压.....-0.3V 至 +4.0V
模拟数字组合引脚和 $\overline{\text{MCLR}}$ 相对于 VSS 的电压 .....-0.3V 到 （VDD + 0.3V）
仅数字引脚相对于 VSS 的电压 ..... -0.3V 到 +6.0V
VDDCORE 相对于 VSS 的电压.....-0.3V 至 +2.8V
VSS 引脚最大输出电流 .....300 mA
VDD 引脚最大输入电流 **（注 1）**.....250 mA
任一 I/O 引脚的最大灌电流 .....25 mA
任一 I/O 引脚的最大拉电流 .....25 mA
所有端口的最大灌电流 .....200 mA
所有端口的最大拉电流 **（注 1）**.....200 mA

注 1： 最大允许电流随器件最大功耗变化 （见表 26-2）。

图 26-1： 频率 / 电压关系图



注 1： 当禁止稳压器时，VDD 和 VDDCORE 必须保持如下关系，使得 VDDCORE ≤ VDD ≤ 3.6V。

† 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。

26.1 直流特性

表 26-1： 工作速度 （单位 MIPS）与电压的关系

VDD 范围 （伏）	温度范围 （°C）	最大 MIPS
		PIC24FJ128GA010 系列
2.0-3.6V	-40°C 至 +85°C	16

表 26-2： 温度工作条件

额定指标	符号	最小值	典型值	最大值	单位
PIC24FJ128GA010 系列：					
工作结点温度范围	TJ	-40	—	+125	°C
工作环境温度范围	TA	-40	—	+85	°C
功耗： 内部芯片功耗： PINT = VDD x (IDD – Σ IOH) I/O 引脚功耗： PI/O = Σ ({VDD – VOH} x IOH) + Σ (VOL x IOL)	PD	PINT + PI/O			W
最大允许功耗	PDMAX	(TJ – TA)/θJA			W

表 26-3： 封装热阻特性

特性	符号	典型值	最大值	单位	注
封装热阻， 14x14x1 mm TQFP	θJA	50	—	°C/W	（注 1）
封装热阻， 12x12x1 mm TQFP	θJA	69.4	—	°C/W	（注 1）
封装热阻， 10x10x1 mm TQFP	θJA	76.6	—	°C/W	（注 1）

注 1： 结点到环境的热阻，通过封装模拟得到的 Theta-JA （θJA）数。

表 26-4： 直流温度和电压规范

直流特性			标准工作条件：2.0V 到 3.6V （除非另外说明） 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 (1)	最大值	单位	条件
工作电压							
DC10	供电电压						
	VDD		2.7	—	3.6	V	使能稳压器
	VDD		VDDCORE	—	3.6	V	禁止稳压器
	VDDCORE		2.0	—	2.75	V	禁止稳压器
DC12	VDR	RAM 数据保持电压 (2)	1.5	—	—	V	
DC16	VPOR	VDD 启动电压 确保能够产生内部上电复位信号	—	VSS	—	V	
DC17	SVDD	VDD 上升速率 确保能够产生内部上电复位信号	0.05	—	—	V/ms	0-3.3V/0.1s 0-2.5V/60 ms

注 1： 除非另外说明，“典型值”栏中的数据均为 3.3V、 25°C 下的值。这些参数仅供设计参考，未经测试。

2： 这是在不丢失 RAM 数据的前提下， VDD 所能降到的最小电压值。

表 26-5： 直流特性：工作电流 （IDD）

直流特性			标准工作条件：2.0V 到 3.6V （除非另行说明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）			
参数编号	**典型值 (1)**	**最大值**	**单位**	**条件**		
工作电流 （IDD）(2)						
DC20	1.6	4.0	mA	-40°C	2.5V(3)	1 MIPS
DC20a	1.6	4.0	mA	+25°C		
DC20b	1.6	4.0	mA	+85°C		
DC20d	1.6	4.0	mA	-40°C	3.6V(4)	
DC20e	1.6	4.0	mA	+25°C		
DC20f	1.6	4.0	mA	+85°C		
DC23	6.0	12	mA	-40°C	2.5V(3)	4 MIPS
DC23a	6.0	12	mA	+25°C		
DC23b	6.0	12	mA	+85°C		
DC23d	6.0	12	mA	-40°C	3.6V(4)	
DC23e	6.0	12	mA	+25°C		
DC23f	6.0	12	mA	+85°C		
DC24	20	32	mA	-40°C	2.5V(3)	16 MIPS
DC24a	20	32	mA	+25°C		
DC24b	20	32	mA	+85°C		
DC24d	20	32	mA	-40°C	3.6V(4)	
DC24e	20	32	mA	+25°C		
DC24f	20	32	mA	+85°C		
DC31	70	150	µA	-40°C	2.5V(3)	LPRC （31 kHz）
DC31a	100	200	µA	+25°C		
DC31b	200	400	µA	+85°C		
DC31d	70	150	µA	-40°C	3.6V(4)	
DC31e	100	200	µA	+25°C		
DC31f	200	400	µA	+85°C		

注 1： 除非另外说明，“典型值”栏中的数据均为 3.3V、 25°C 下的值。这些参数仅供设计参考，未经测试。

2： 供电电流主要是由工作电压和频率决定的。其他因素，如 I/O 引脚负载和开关频率、振荡器类型、内部代码执行模式和温度也会影响电流消耗。所有 IDD 测量的测试条件如下：OSC1 由满幅外部方波驱动；所有 I/O 引脚配置为输入，并拉至 VDD； MCLR = VDD；禁止 WDT 和 FSCM； CPU、SRAM、程序存储器和数据存储器正常工作；外设模块不工作且 PMD 位置 1。

3： 禁止片内稳压器 （ENVREG 连接到 VSS）。

4： 使能片内稳压器 （ENVREG 连接到 VDD）。

表 26-6： 直流特性：空闲电流 （IIDLE）

直流特性			标准工作条件：2.0V 到 3.6V （除非另外说明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）			
参数编号	**典型值 (1)**	**最大值**	**单位**	**条件**		
空闲电流 （IIDLE）：内核关闭、时钟启用时的基本电流 (2)						
DC40	0.7	2	mA	-40°C	2.5V(3)	1 MIPS
DC40a	0.7	2	mA	+25°C		
DC40b	0.7	2	mA	+85°C		
DC40d	0.7	2	mA	-40°C	3.6V(4)	
DC40e	0.7	2	mA	+25°C		
DC40f	0.7	2	mA	+85°C		
DC43	2.1	4	mA	-40°C	2.5V(3)	4 MIPS
DC43a	2.1	4	mA	+25°C		
DC43b	2.1	4	mA	+85°C		
DC43d	2.1	4	mA	-40°C	3.6V(4)	
DC43e	2.1	4	mA	+25°C		
DC43f	2.1	4	mA	+85°C		
DC47	6.8	8	mA	-40°C	2.5V(3)	16 MIPS
DC47a	6.8	8	mA	+25°C		
DC47b	6.8	8	mA	+85°C		
DC47c	6.8	8	mA	-40°C	3.6V(4)	
DC47d	6.8	8	mA	+25°C		
DC47e	6.8	8	mA	+85°C		
DC51	150	500	µA	-40°C	2.5V(3)	LPRC （31 kHz）
DC51a	150	500	µA	+25°C		
DC51b	150	500	µA	+85°C		
DC51d	150	500	µA	-40°C	3.6V(4)	
DC51e	150	500	µA	+25°C		
DC51f	150	500	µA	+85°C		

注 1： 除非另外说明，“典型值”栏中的数据均为 3.3V、 25°C 下的值。这些参数仅供设计参考，未经测试。

2： 基本 IIDLE 电流是在 PMD 位置 1，关闭内核和所有模块，但保持时钟工作的条件下测得的。

3： 禁止片内稳压器 （ENVREG 连接到 VSS）。

4： 使能片内稳压器 （ENVREG 连接到 VDD）。

表 26-7： 直流特性：掉电电流 （IPD）

直流特性			标准工作条件：2.0V 到 3.6V （除非另外说明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）			
参数编号	**典型值 (1)**	**最大值**	**单位**	**条件**		
掉电电流 （IPD）(2)						
DC60	3	25	µA	-40°C	2.0V(3)	基本掉电电流 (5)
DC60a	3	45	µA	+25°C		
DC60b	100	600	µA	+85°C		
DC60f	20	40	µA	-40°C	3.6V(4)	
DC60g	27	60	µA	+25°C		
DC60h	120	600	µA	+85°C		
模块差分电流						
DC61	10	25	µA	-40°C	2.0V(3)	看门狗定时器电流：ΔIWDT(5)
DC61a	10	25	µA	+25°C		
DC61b	10	25	µA	+85°C		
DC61f	10	25	µA	-40°C	3.6V(4)	
DC61g	10	25	µA	+25°C		
DC61h	10	25	µA	+85°C		
DC62	8	15	µA	-40°C	2.0V(3)	RTCC + Timer1 （带有 32 kHz 晶振）：ΔIRTCC(5)
DC62a	8	15	µA	+25°C		
DC62b	8	15	µA	+85°C		
DC62f	8	15	µA	-40°C	3.6V(4)	
DC62g	8	15	µA	+25°C		
DC62h	8	15	µA	+85°C		

注 1： 除非另外说明，“典型值”栏中的数据均为 3.3V、 25°C 下的值。这些参数仅供设计参考，未经测试。

2： 基本 IPD 是在关闭所有外设和时钟的条件下测得的。所有 I/O 配置为输入，并拉为高电平。关闭 WDT 等所有模块。未使用的 PMD 位置 1。 VREGS 位清零。

3： 禁止片内稳压器 （ENVREG 连接到 VSS）。

4： 使能片内稳压器 （ENVREG 连接到 VDD）。

5： 当使能模块时， Δ 电流是额外消耗电流。该电流应与基本 IPD 电流相加。

表 26-8： 直流特性：I/O 引脚输入规范

直流特性			标准工作条件：2.0V 到 3.6V （除非另外说明） 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 (1)	最大值	单位	条件
	VIL	**输入低电压 (4)**					
DI10		带 ST 缓冲器的 I/O 引脚	VSS	—	0.2 VDD	V	
DI11		带 TTL 缓冲器的 I/O 引脚	VSS	—	0.15 VDD	V	
DI15		$\overline{MCLR}$	VSS	—	0.2 VDD	V	
DI16		OSC1 （XT 模式）	VSS	—	0.2 VDD	V	
DI17		OSC1 （HS 模式）	VSS	—	0.2 VDD	V	
DI18		带 I2C™ 缓冲器的 I/O 引脚：	VSS	—	0.3 VDD	V	
DI19		带 SMBus 缓冲器的 I/O 引脚：	VSS	—	0.8	V	使能 SMBus
	VIH	**输入高电压 (4)**					
DI20		带 ST 缓冲器的 I/O 引脚：					
		具有模拟功能，	0.8 VDD	—	VDD	V	
		仅有数字功能	0.8 VDD	—	5.5	V	
DI21		带 TTL 缓冲器的 I/O 引脚：					
		具有模拟功能，	0.25 VDD + 0.8	—	VDD	V	
		仅有数字功能	0.25 VDD + 0.8	—	5.5	V	
DI25		$\overline{MCLR}$	0.8 VDD	—	VDD	V	
DI26		OSC1 （XT 模式）	0.7 VDD	—	VDD	V	
DI27		OSC1 （HS 模式）	0.7 VDD	—	VDD	V	
DI28		带 I2C 缓冲器的 I/O 引脚：					
		具有模拟功能，	0.7 VDD	—	VDD	V	
		仅有数字功能	0.7 VDD	—	5.5	V	
DI29		带 SMBus 缓冲器的 I/O 引脚：	2.1		VDD	V	
		具有模拟功能，	2.1		5.5	V	
		仅有数字功能					2.5V ≤ VPIN ≤ VDD
DI30	ICNPU	**CNx 上拉电流**	50	250	400	µA	VDD = 3.3V， VPIN = VSS
	IIL	**输入泄漏电流 (2,3)**					
DI50		I/O 端口	—	—	±1	µA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态
DI51		模拟输入引脚	—	—	±1	µA	VSS ≤ VPIN ≤ VDD，引脚处于高阻态
DI55		$\overline{MCLR}$	—	—	±1	µA	VSS ≤ VPIN ≤ VDD
DI56		OSC1	—	—	±1	µA	VSS ≤ VPIN ≤ VDD, XT 和 HS 模式

注 1： 除非另外说明，“典型值”栏中的数据均为 3.3V、 25°C 下的值。这些参数仅供设计参考，未经测试。

2： $\overline{MCLR}$ 引脚上的泄漏电流主要由施加在该引脚上的电平决定。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能会测得更高的泄漏电流。

3： 负电流定义为自引脚流出的电流。

4： I/O 引脚缓冲器类型请参见表 1-2。

表 26-9： 直流特性：I/O 引脚输出规范

直流特性			标准工作条件：2.0V 到 3.6V （除非另外说明） 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 (1)	最大值	单位	条件
	VOL	**输出低电压**					
DO10		I/O 端口	—	—	0.4	V	IOL = 8.5 mA， VDD = 3.6V
			—	—	0.4	V	IOL = 6.0 mA， VDD = 2.0V
DO16		OSC2/CLKO	—	—	0.4	V	IOL = 8.5 mA， VDD = 3.6V
			—	—	0.4	V	IOL = 6.0 mA， VDD = 2.0V
	VOH	**输出高电压**					
DO20		I/O 端口	3.0	—	—	V	IOH = -3.0 mA， VDD = 3.6V
			2.4	—	—	V	IOH = -6.0 mA， VDD = 3.6V
			1.65	—	—	V	IOH = -1.0 mA， VDD = 2.0V
			1.4	—	—	V	IOH = -3.0 mA， VDD = 2.0V
DO26		OSC2/CLKO	2.4	—	—	V	IOH = -6.0 mA， VDD = 3.6V
			1.4	—	—	V	IOH = -3.0 mA， VDD = 2.0V

注 1： 除非另外说明，“典型值”栏中的数据均为 3.3V、 25°C 下的值。这些参数仅供设计参考，未经测试。

表 26-10： 直流特性：程序存储器

直流特性			标准工作条件：2.0V 到 3.6V （除非另外说明） 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 (1)	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	读取使用的 VDD	VMIN	—	3.6	V	VMIN = 最小工作电压
D132B	VPEW	自定时写 / 擦除使用的 VDD	2.25	—	3.6	V	VMIN = 最小工作电压
D133A	TIW	自定时写周期时间	—	3	—	ms	
D134	TRETD	保存时间	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	

注 1： 除非另外说明，“典型值”栏中的数据均为 3.3V、 25°C 下的值。

表 26-11： 内部稳压器规范

工作条件：-40°C < TA < +85°C （除非另外说明）							
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	外部滤波电容值	4.7	10	—	µF	推荐等效串联阻抗 < 3Ω 的电容；至少要 < 5Ω
	TVREG		—	10	—	µs	ENVREG = VDD
	TPWRT		—	64	—	ms	ENVREG = VSS

26.2 交流特性和时序参数

本小节包含的信息定义 PIC24FJ128GA010 系列交流特性和时序参数。

表 26-12： 温度和电压规范——交流

交流特性	**标准工作条件：2.0V 到 3.6V （除非另外说明）** 工作温度 -40°C ≤ TA ≤ +85°C （工业级） 工作电压 VDD 范围如**第 26.1 节 “直流特性”**所示。

图 26-2： 器件时序规范的负载条件

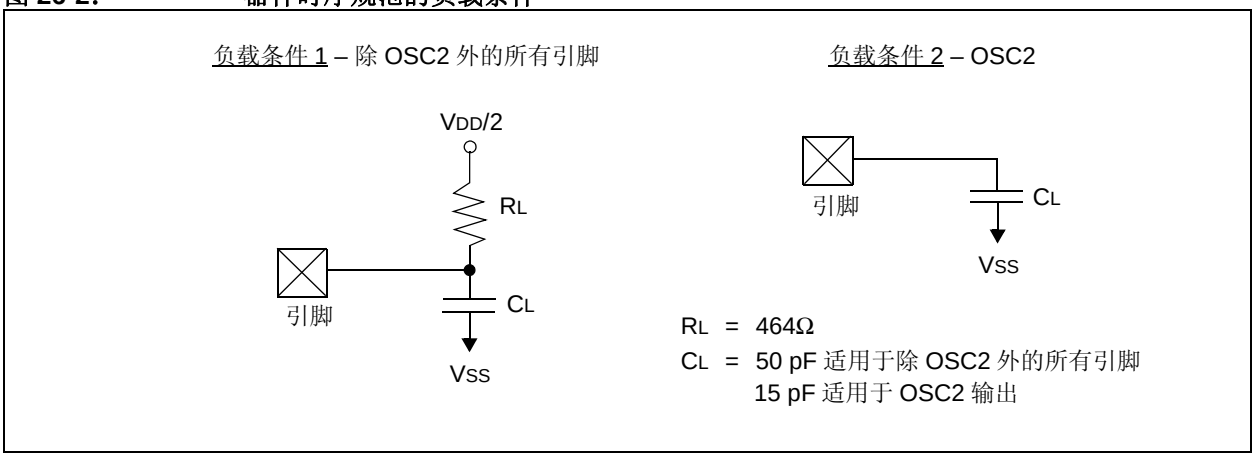


表 26-13： 输出引脚的容性负载要求

参数编号	符号	特性	最小值	典型值 (1)	最大值	单位	条件
DO50	COSC2	OSC2/CLKO 引脚	—	—	15	pF	当外部时钟用于驱动 OSC1 时，处于 XT 和 HS 模式。
DO56	CIO	所有 I/O 引脚和 OSC2	—	—	50	pF	EC 模式
DO58	CB	SCLx 和 SDAx	—	—	400	pF	处于 I^2C™ 模式

注 1： 除非另外说明，“典型值”栏中的数据均为 3.3V、 25°C 下的值。这些参数仅供设计参考，未经测试。

图 26-3：　　外部时钟时序

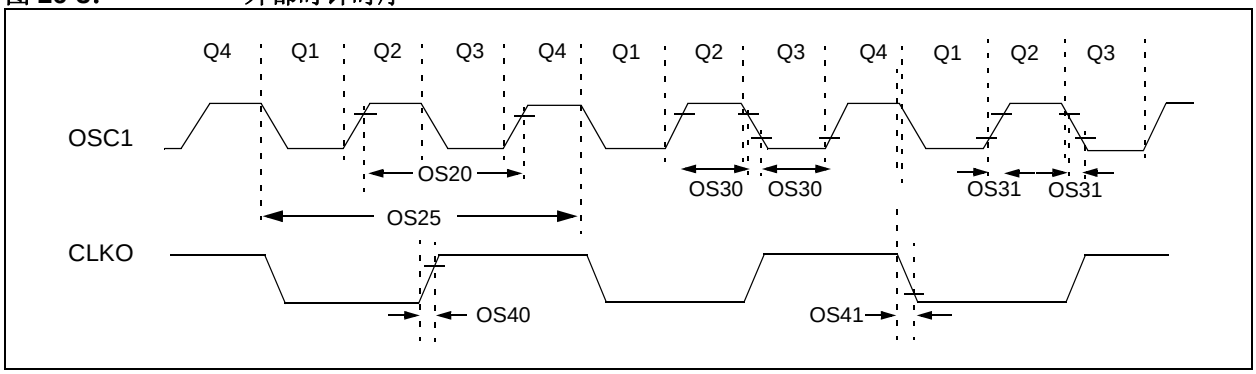


表 26-14：　　外部时钟时序要求

交流特性			**标准工作条件：2.0V 到 3.6V （除非另外说明）** 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 (1)**	**最大值**	**单位**	**条件**
OS10	FOSC	外部 CLKI 频率 （仅在 EC 模式下允许使用外部时钟）	DC 3	— —	32 8	MHz MHz	EC ECPLL
		振荡器频率	3.5 3.5 10 31	— — — —	10 8 32 33	MHz MHz MHz kHz	XT XTPLL HS SOSC
OS20	TOSC	TOSC = 1/FOSC	—	—	—	—	见参数 OS10 获取 FOSC 值
OS25	TCY	指令周期 (2)	62.5	—	DC	ns	
OS30	TosL, TosH	外部时钟输入 （OSC1） 高电平或低电平时间	0.45 x TOSC	—	—	ns	EC
OS31	TosR, TosF	外部时钟输入 （OSC1） 上升或下降时间	—	—	20	ns	EC
OS40	TckR	CLKO 上升时间 (3)	—	6	10	ns	
OS41	TckF	CLKO 下降时间 (3)	—	6	10	ns	

注　1： 除非另外说明，否则 “典型值”栏中的数据均为 3.3V、 25°C 条件下的值。这些参数仅供设计参考，未经测试。

2： 指令周期 （TCY）等于输入振荡器时基周期的 2 倍。所有规范值均为器件在标准工作条件下执行代码时对应特定振荡器类型的特性数据。超过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值。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”周期时间限制为 “DC”（没有时钟）。

3： 数据是在 EC 模式下测得的。CLKO 信号是在 OSC2 引脚上测得的。CLKO 在 Q1-Q2 周期为低电平 （1/2 TCY）， Q3-Q4 周期为高电平 （1/2 TCY）。

表 26-15： PLL 时钟时序规范 （VDD = 2.0V 至 3.6V）

交流特性			标准工作条件：2.0V 到 3.6V （除非另外说明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	符号	特性 (1)	最小值	典型值 (2)	最大值	单位	条件
OS50	FPLLI	PLL 输入频率范围 (2)	3	—	8	MHz	ECPLL、 HSPLL 和 XTPLL 模式
OS51	FSYS	片内 VCO 系统频率	8	—	32	MHz	
OS52	TLOCK	PLL 启动时间（锁定时间）	—	—	2	ms	
OS53	DCLK	CLKO 稳定性 （抗抖动性能）	-2	1	+2	%	在大于 100 ms 时间内测得。

注 1： 这些参数仅为特征值，未经生产测试。

2： 除非另外说明，否则 “典型值”栏中的数据均为 3.3V、 25°C 下的值。这些参数仅供设计参考，未经测试。

表 26-16： 交流特性：内部 RC 精度

交流特性		标准工作条件：2.0V 到 3.6V （除非另外说明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	特性	最小值	典型值	最大值	单位	条件	
F20	**8 MHz 下的内部 FRC 精度 (1)**						
	FRC	-2	—	+2	%	+25°C	V_{dd} = 3.0 - 3.6V
		-5	—	+5	%	-40°C ≤ TA ≤ +85°C	V_{dd} = 3.0 - 3.6V

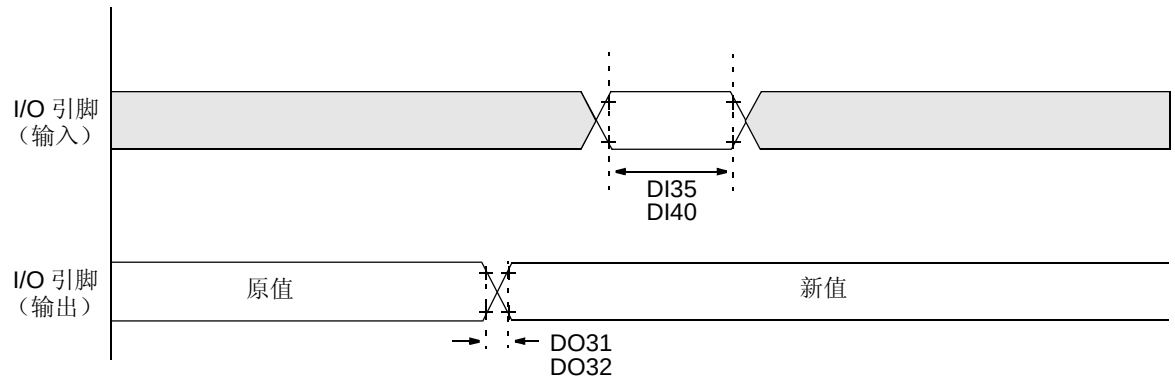
注 1： 频率在 25°C 和 3.3V 条件下校准。OSCTUN 位可用于补偿温度漂移。

表 26-17： 内部 RC 精度

交流特性		标准工作条件：2.0V 到 3.6V （除非另外说明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）					
参数编号	特性	最小值	典型值	最大值	单位	条件	
	31 kHz 下的 LPRC(1)						
F21		-15	—	+15	%	-40°C ≤ TA ≤ +85°C	VDD = 3.0 - 3.6V

注 1： LPRC 频率随 VDD 改变而变化。

图 26-4：　　CLKO 和 I/O 时序特性



注：请参见图 26-2 了解负载条件。

表 26-18：　　CLKO 和 I/O 时序要求

交流特性			**标准工作条件：2.0V 到 3.6V（除非另外说明）** 工作温度　　-40°C ≤ TA ≤ +85°C （工业级）				
参数编号	**符号**	**规范**	**最小值**	**典型值 (1)**	**最大值**	**单位**	**条件**
DO31	TIOR	端口输出上升时间	—	10	25	ns	
DO32	TIOF	端口输出下降时间	—	10	25	ns	
DI35	TINP	INTx 引脚高电平或低电平时间（输出）	20	—	—	ns	
DI40	TRBP	CNx 高电平或低电平时间（输入）	2	—	—	TCY	

注　1：除非另外说明，否则“典型值”栏中的数据均为 3.3V、25°C 下的值。

表 26-19： ADC 模块规范

交流特性			标准工作条件：2.0V 至 3.6V （除非另外说明） 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
器件电源							
AD01	AVDD	模块电源 VDD	取 VDD - 0.3 或 2.0 中的较大值	—	取 VDD + 0.3 或 3.6 中的较小值	V	
AD02	AVSS	模块电源 VSS	VSS - 0.3	—	VSS + 0.3	V	
参考输入							
AD05	VREFH	参考电压高电平	AVSS + 1.7	—	AVDD	V	
AD06	VREFL	参考电压低电平	AVSS	—	AVDD - 1.7	V	
AD07	VREF	绝对参考电压	AVSS - 0.3	—	AVDD + 0.3	V	
模拟输入							
AD10	VINH-VINL	满量程输入范围	VREFL	—	VREFH	V	**（注 2）**
AD11	VIN	绝对输入电压	AVSS – 0.3	—	AVDD + 0.3	V	
AD12	—	泄漏电流	—	±0.001	±0.610	µA	VINL = AVSS = VREFL = 0V， AVDD = VREFH = 5V， 源阻抗 = 2.5 kΩ
AD13	—	泄漏电流	—	±0.001	±0.610	µA	VINL = AVSS = VREFL = 0V， AVDD = VREFH = 3V， 源阻抗 = 2.5 kΩ
AD14	VINL	绝对 VINL 输入电压	AVSS – 0.3	—	AVDD/2	V	
AD17	RIN	模拟信号源的推荐阻抗	—	—	2.5K	Ω	
ADC 精度							
AD20a	NR	分辨率	—	10	—	位	
AD21a	INL	积分非线性误差 **(2)**	—	±1	<±2	LSb	VINL = AVSS = VREFL = 0V， AVDD = VREFH = 3V
AD22a	DNL	微分非线性误差 **(2)**	—	±0.5	<±1	LSb	VINL = AVSS = VREFL = 0V， AVDD = VREFH = 3V
AD23a	GERR	增益误差 **(2)**	—	±1	±3	LSb	VINL = AVSS = VREFL = 0V， AVDD = VREFH = 3V
AD24a	EOFF	失调误差 **(2)**	—	±1	±2	LSb	VINL = AVSS = VREFL = 0V， AVDD = VREFH = 3V
AD25a	—	单调性 **(1)**	—	—	—	—	保证

注 1： A/D 转换结果将不会随着输入电压的增加而减小，而且不会丢失编码。

2： 测量是在使用外部 VREF+ 和 VREF- 作为 ADC 参考电压的情况下进行的。

表 26-20： ADC 转换时序要求 (1)

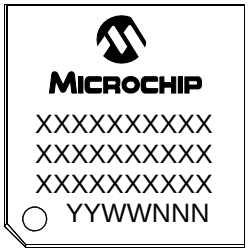
交流特性			标准工作条件：2.0V 至 3.6V（除非另外说明） 工作温度 -40°C ≤ TA ≤ +85°C （工业级）				
参数编号	符号	特性	最小值	典型值 (1)	最大值	单位	条件
AD50	TAD	ADC 时钟周期	75	—	—	ns	TCY = 75 ns， ADxCON3 处于默认状态
AD51	tRC	ADC 内部 RC 振荡器周期	—	250	—	ns	
转换速率							
AD55	tCONV	转换时间	—	12	—	TAD	
AD56	FCNV	吞吐率	—	—	500	ksps	
AD57	tSAMP	采样时间	—	1	—	TAD	
时钟参数							
AD61	tPSS	从采样位 （SAMP）置 1 到采样启动的延时	2	—	3	TAD	

注 1： 因为采样电容最终将释放电荷，因此低于 10 kHz 的时钟频率可能会影响线性性能，尤其在温度较高时。

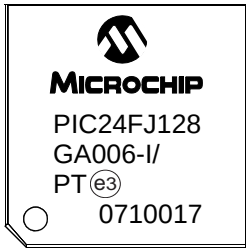
27.0 封装信息

27.1 封装标识信息

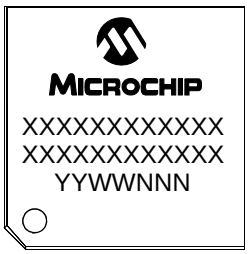
64 引脚 TQFP（10x10x1 mm）



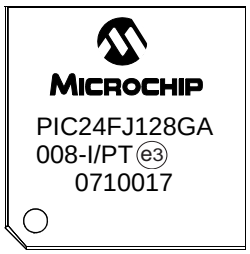
示例



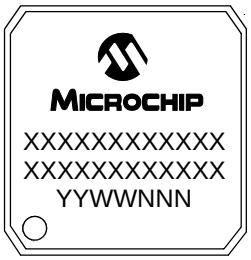
80 引脚 TQFP（12x12x1 mm）



示例

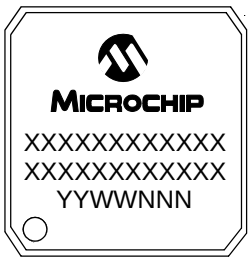


100 引脚 TQFP（12x12x1 mm）



示例

100 引脚 TQFP（14x14x1 mm）



示例

图注：	XX...X	客户指定信息
	Y	年份代码 （日历年的最后一位数字）
	YY	年份代码 （日历年的最后两位数字）
	WW	星期代码 （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)	雾锡 （Matte Tin， Sn）的 JEDEC 无铅标志
	*	表示无铅封装。 JEDEC 无铅标志 （(e3)） 标示于此种封装的外包装上。

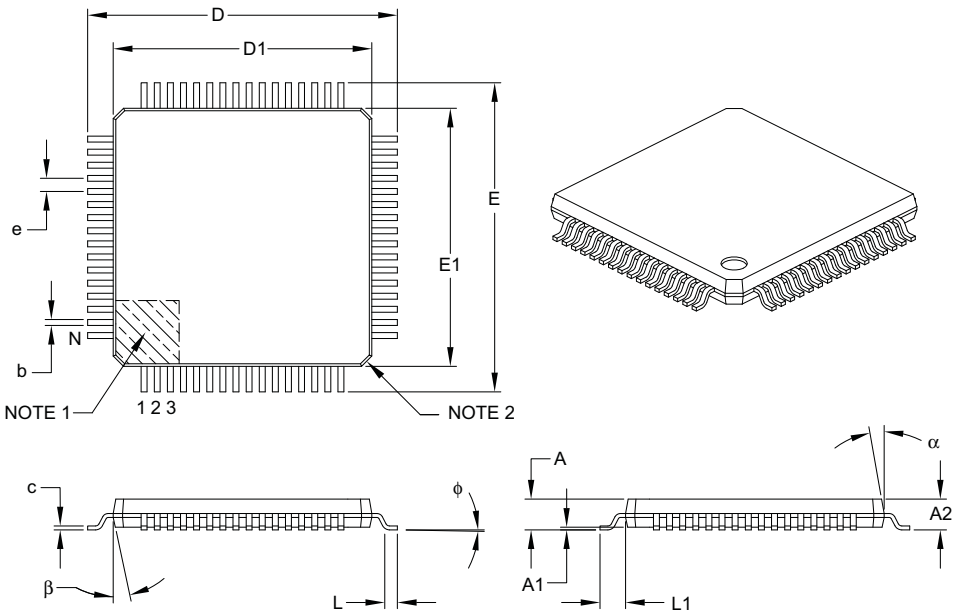
注： Microchip 元器件编号如果无法在同一行内完整标注，将换行标出，因此会限制表示客户信息的字符数。

27.2 封装详细信息

以下部分将介绍各种封装技术的细节。

64 引脚塑封薄型四方扁平封装 （PT）——主体 10x10x1 mm，引脚占位长度 2.00 mm [TQFP]

注： 最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		64	
Lead Pitch	e		0.50 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E		12.00 BSC	
Overall Length	D		12.00 BSC	
Molded Package Width	E1		10.00 BSC	
Molded Package Length	D1		10.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

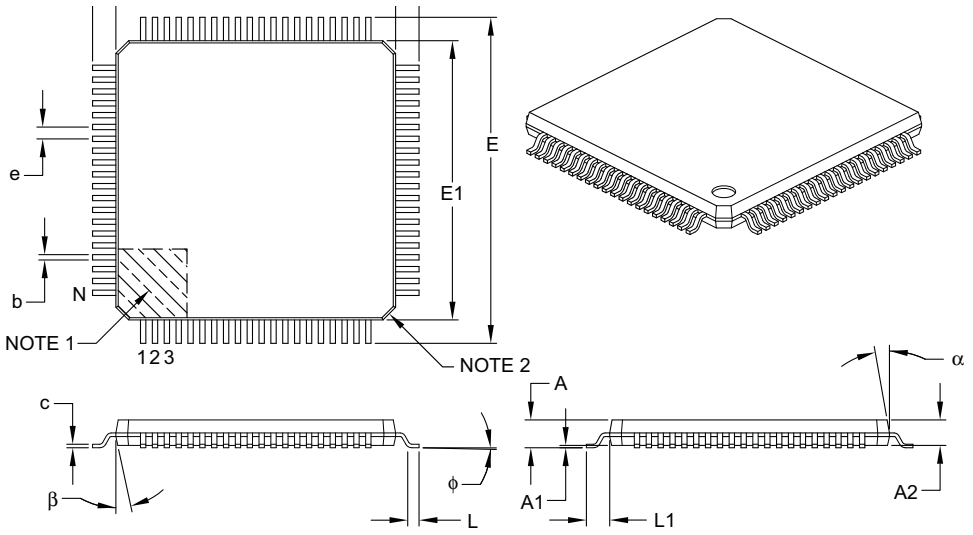
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-085B

80 引脚塑封薄型四方扁平封装 （PT）——主体 12x12x1 mm，引脚占位长度 2.00 mm [TQFP]

注： 最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		80	
Lead Pitch	e		0.50 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E		14.00 BSC	
Overall Length	D		14.00 BSC	
Molded Package Width	E1		12.00 BSC	
Molded Package Length	D1		12.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

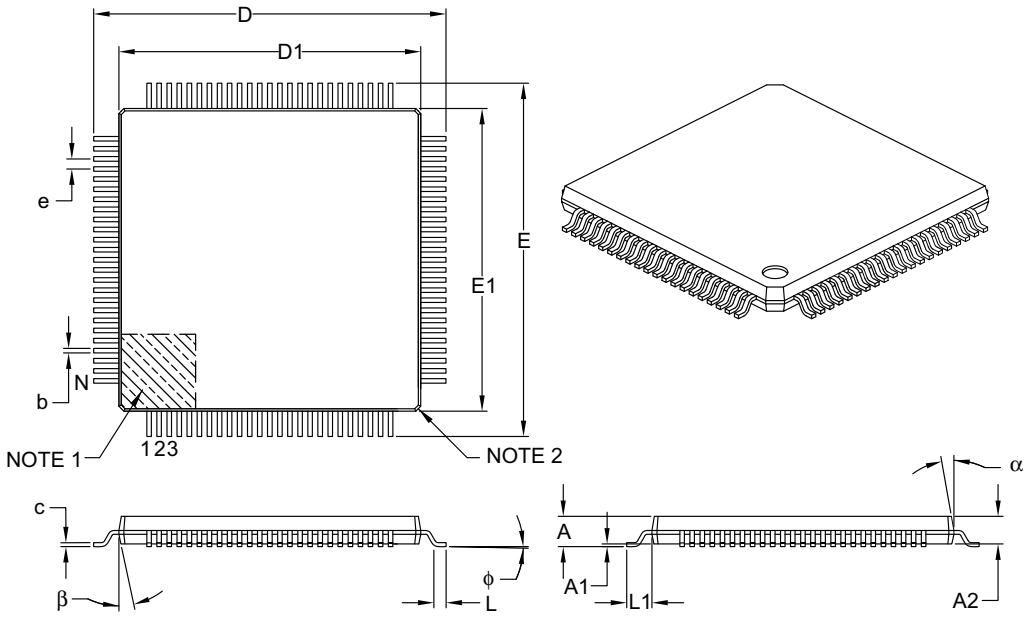
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-092B

100 引脚塑封薄型四方扁平封装 （PT）——主体 12x12x1 mm，引脚占位长度 2.00 mm [TQFP]

注： 最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		100	
Lead Pitch	e		0.40 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E		14.00 BSC	
Overall Length	D		14.00 BSC	
Molded Package Width	E1		12.00 BSC	
Molded Package Length	D1		12.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.13	0.18	0.23
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

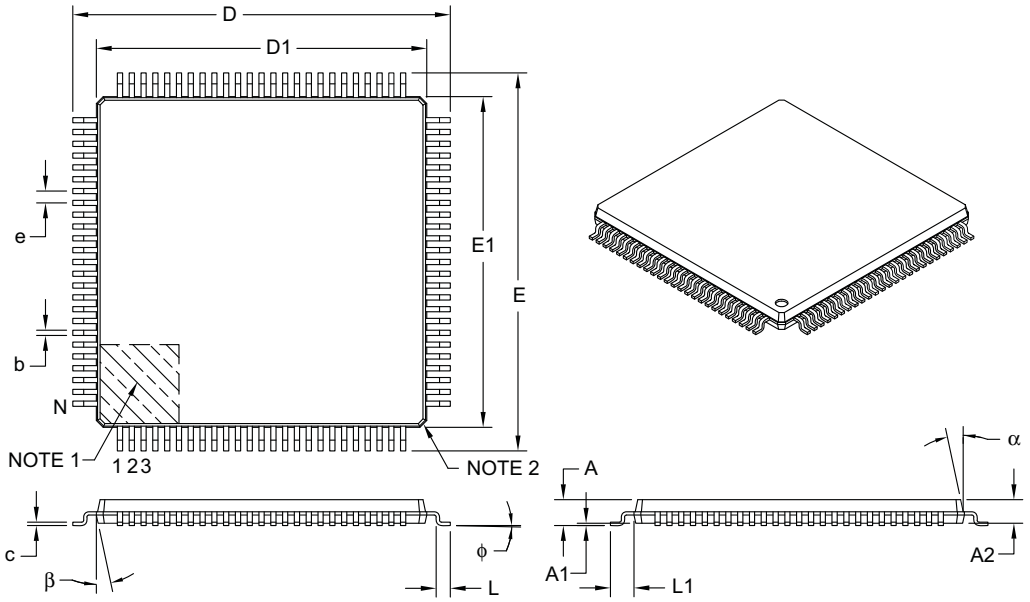
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-100B

100 引脚塑封薄型四方扁平封装 （PF）——主体 14x14x1 mm，引脚占位长度 2.00 mm [TQFP]

注: 最新封装图请至 http://www.microchip.com/packaging 查看 Microchip 封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		100	
Lead Pitch	e		0.50 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E		16.00 BSC	
Overall Length	D		16.00 BSC	
Molded Package Width	E1		14.00 BSC	
Molded Package Length	D1		14.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-110B

注：

附录 A： 版本历史

版本 A （2005 年 9 月）

PIC24FJ128GA010 系列器件的初始数据手册。

版本 B （2006 年 3 月）

更新了电气特性。

版本 C （2006 年 6 月）

更新了电气特性。

版本 D （2007 年 9 月）

对整个数据手册做了少量修改。

版本 E （2009 年 10 月）

做了更新，去掉了 “初稿” 状态。

注：

索引

K

Y

Z

MICROCHIP 网站

Microchip 网站（www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的因特网浏览器即可访问。网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及存档软件
- **一般技术支持**——常见问题（FAQ）、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

变更通知客户服务

Microchip 的变更通知客户服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 www.microchip.com。在“支持”（Support）下，点击“变更通知客户”（Customer Change Notification）服务后按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师（FAE）
- 技术支持

客户应联系其代理商、代表或应用工程师（FAE）寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过http://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: PIC24FJ128GA010 系列　　文献编号: DS39747E_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。

PIC 24 FJ 128 GA0 10 T - I / PT - XXX

- **Microchip 商标**
- **架构**
- **闪存存储器系列**
- **程序存储器容量（KB）**
- **产品类别**
- **引脚数**
- **卷带式标志（如果适用）**
- **温度范围**
- **封装**
- **模式**

架构	24	= 不带 DSP 功能的 16 位改进型哈佛架构
闪存存储器系列	FJ	= 闪存程序存储器
产品类别	GA0	= 通用单片机
引脚数	06 08 10	= 64 引脚 = 80 引脚 = 100 引脚
温度范围	I	= -40°C 至 +85°C （工业级）
封装	PT PF	= 64 引脚、80 引脚和 100 引脚 （12x12x1 mm） TQFP （薄型正方扁平封装） = 100 引脚（14x14x1 mm）TQFP（薄型正方扁平封装）
模式	3 位数字表示 QTP、 SQTP、编码或特殊要求（空白为其他情况） ES	= 工程样片

示例：

a) PIC24FJ128GA008-I/PT 301：
通用 PIC24F、 96 KB 程序存储器、80 引脚、工业级温度范围、 TQFP 封装和 QTP 模式 301。

b) PIC24FJ128GA010-I/PT：
通用 PIC24F、 128 KB 程序存储器、 100 引脚、工业级温度范围和 TQFP 封装。

全球销售及服务网点

美洲

公司总部 Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://support.microchip.com
网址：www.microchip.com

亚特兰大 Atlanta
Duluth, GA
Tel: 1-678-957-9614
Fax:1-678-957-1455

波士顿 Boston
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

克里夫兰 Cleveland
Independence, OH
Tel: 1-216-447-0464
Fax: 1-216-447-0643

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Farmington Hills, MI
Tel: 1-248-538-2250
Fax: 1-248-538-2260

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

圣克拉拉 Santa Clara
Santa Clara, CA
Tel: 1-408-961-6444
Fax: 1-408-961-6445

加拿大多伦多 Toronto
Mississauga, Ontario,
Canada
Tel: 1-905-673-0699
Fax: 1-905-673-6509

亚太地区

亚太总部 Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 北京
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

中国 – 成都
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

中国 – 重庆
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

中国 – 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 南京
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

中国 – 青岛
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 – 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 – 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 – 深圳
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

中国 – 武汉
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 – 西安
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

中国 – 厦门
Tel: 86-592-238-8138
Fax: 86-592-238-8130

中国 – 珠海
Tel: 86-756-321-0040
Fax: 86-756-321-0049

台湾地区 – 高雄
Tel: 886-7-213-7830
Fax: 886-7-330-9305

台湾地区 – 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

亚太地区

台湾地区 – 新竹
Tel: 886-3-6578-300
Fax: 886-3-6578-370

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

印度 India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

印度 India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

日本 Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

韩国 Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

韩国 Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

马来西亚 Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

马来西亚 Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

菲律宾 Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

新加坡 Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

泰国 Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

欧洲

奥地利 Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

丹麦 Denmark-Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

法国 France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

意大利 Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

荷兰 Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

西班牙 Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

英国 UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

07/15/10