



PIC24F16KA102 系列

数据手册

20/28 引脚通用 16 位
超低功耗闪存单片机

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

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

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

商标

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

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

Analog-for-the-Digital Age、Application Maestro、CodeGuard、dsPICDEM、dsPICDEM.net、dsPICworks、dsSPEAK、ECAN、ECONOMONITOR、FanSense、In-Circuit Serial Programming、ICSP、ICEPIC、Mindi、MiWi、MPASM、MPLAB Certified 徽标、MPLIB、MPLINK、mTouch、PICkit、PICDEM、PICDEM.net、PICtail、PIC³² 徽标、PowerCal、PowerInfo、PowerMate、PowerTool、REAL ICE、rfLAB、Select Mode、Total Endurance、WiperLock 和 ZENA 均为 Microchip Technology Inc. 在美国和其他国家或地区的商标。

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

在此提及的所有其他商标均为各持有公司所有。

© 2009, Microchip Technology Inc. 版权所有。

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。

20/28 引脚通用 16 位超低功耗闪存单片机

功耗管理模式:

- 运行——CPU、闪存、SRAM 和外设都工作
- 打盹——相当于 CPU 时钟比外设慢
- 空闲——CPU 不工作，闪存、SRAM 和外设都工作
- 休眠——CPU、闪存和外设都不工作，SRAM 工作
- 深度休眠——CPU、闪存、SRAM 和大多数外设都不工作

高性能 CPU:

- 改进型哈佛架构
- 32 MHz 时最高 16 MIPS 工作速度
- 8 MHz 内部振荡器，带 4x PLL 选项和多个分频选项
- 17 位 x 17 位单周期硬件乘法器
- 32 位 /16 位硬件除法器
- 16 位 x 16 位工作寄存器阵列
- 优化的 C 编译器指令集架构

外设特性:

- 硬件实时时钟和日历 (Real-Time Clock and Calendar, RTCC):
 - 提供时钟、日历和闹钟功能
 - 可在深度休眠模式下运行
- 可编程循环冗余校验 (Cyclic Redundancy Check, CRC)
- 串行通信模块:
 - SPI、I²C™ 和两个 UART 模块
- 3 个带可编程预分频器的 16 位定时器 / 计数器
- 16 位捕捉输入
- 16 位比较 / PWM 输出
- 数字 I/O 引脚上的可配置漏极开路输出
- 最多 3 个外部中断源

模拟特性:

- 最多 9 路通道的 10 位模数转换器:
 - 转换速率为 500 ksp/s
 - 在休眠和空闲期间可以进行转换
- 带可编程输入 / 输出配置的双模拟比较器
- 充电时间测量单元 (Charge Time Measurement Unit, CTMU):
 - 用于电容检测
 - 时间测量，分辨率可达 1 ns
 - 延时 / 脉冲生成，分辨率可达 1 ns

单片机特性:

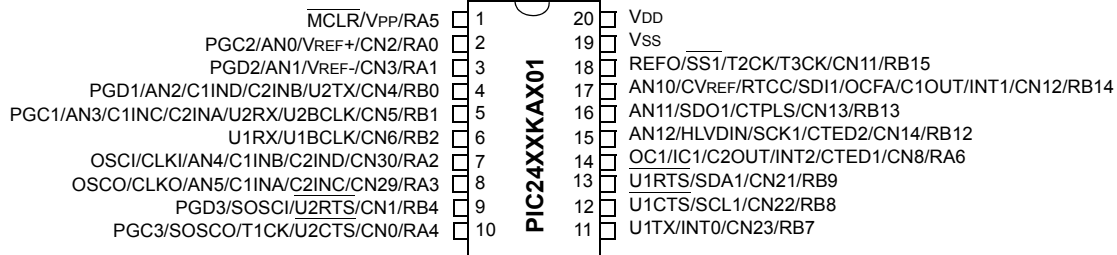
- 工作电压范围为 1.8V 至 3.6V
- 所有 I/O 引脚上的高灌 / 拉电流 (18 mA/18 mA)
- 闪存程序存储器:
 - 擦除 / 写周期: 最小周期为 1,000 次，典型周期为 10,000 次
 - 最少 40 年数据保存时间
- 数据 EEPROM:
 - 擦除 / 写周期: 最小周期为 10,000 次，典型周期为 1,000,000 次
 - 最少 40 年数据保存时间
- 故障保护时钟监视器
- 系统频率范围声明位:
 - 声明频率范围可以帮助优化电流消耗
- 灵活的看门狗定时器 (Watchdog Timer, WDT)，带片上低功耗 RC 振荡器以便可靠地工作
- 通过两个引脚进行在线串行编程 (In-Circuit Serial Programming™, ICSP™) 和在线调试 (In-Circuit Debug, ICD)
- 可编程高 / 低压检测 (High/Low-Voltage Detect, HLVD)
- 欠压复位 (Brown-out Reset, BOR):
 - 带有 3 个可编程跳变点的标准 BOR；在休眠模式下可以禁止
- 用于深度休眠模式的超低功耗 DSBOR，用于所有其他模式的低功耗 BOR

PIC24F 器件	引脚数	程序存储器 (字节)	SRAM (字节)	数据 EEPROM (字节)	16 位定时器	捕捉输入	输出比较 / PWM	UART/ IrDA®	SPI	I ² C™	10 位 A/D (通道)	比较器	CTMU (通道)	RTCC
08KA101	20	8K	1.5K	512	3	1	1	2	1	1	9	2	9	有
16KA101	20	16K	1.5K	512	3	1	1	2	1	1	9	2	9	有
08KA102	28	8K	1.5K	512	3	1	1	2	1	1	9	2	9	有
16KA102	28	16K	1.5K	512	3	1	1	2	1	1	9	2	9	有

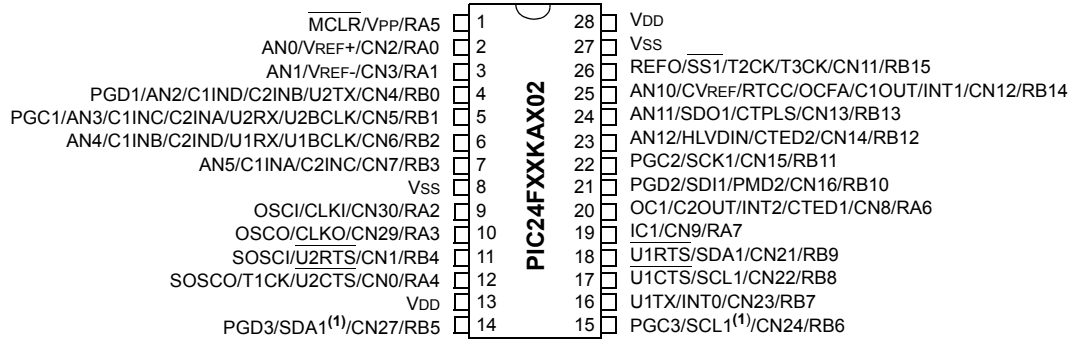
PIC24F16KA102 系列

引脚图

20 引脚 PDIP、SSOP 和 SOIC

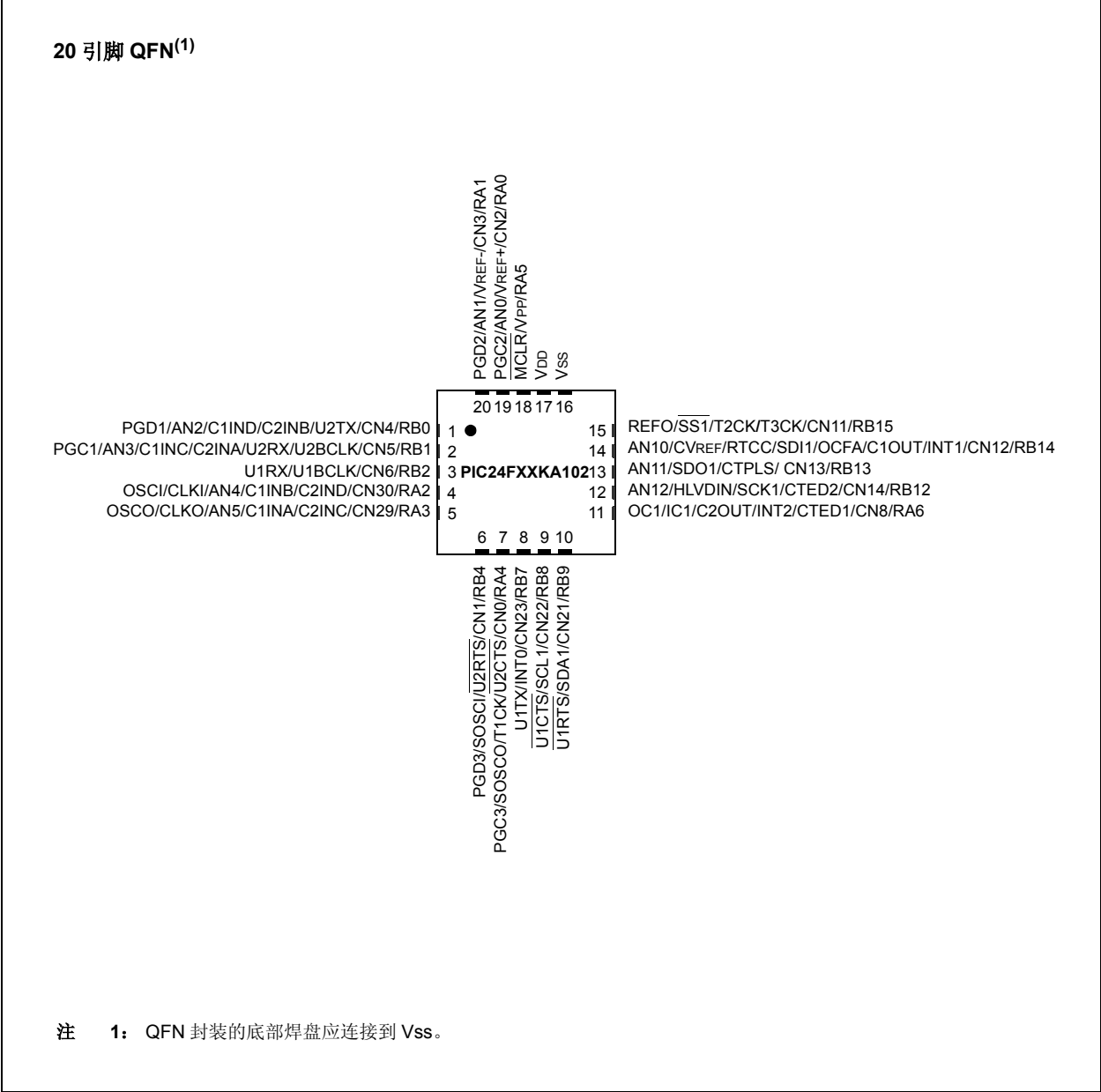


28 引脚 SPDIP、SSOP 和 SOIC



注 1: 当 I2CSEL 配置位置 1 时, 相应的引脚分别用作 SDA1 和 SCL1。

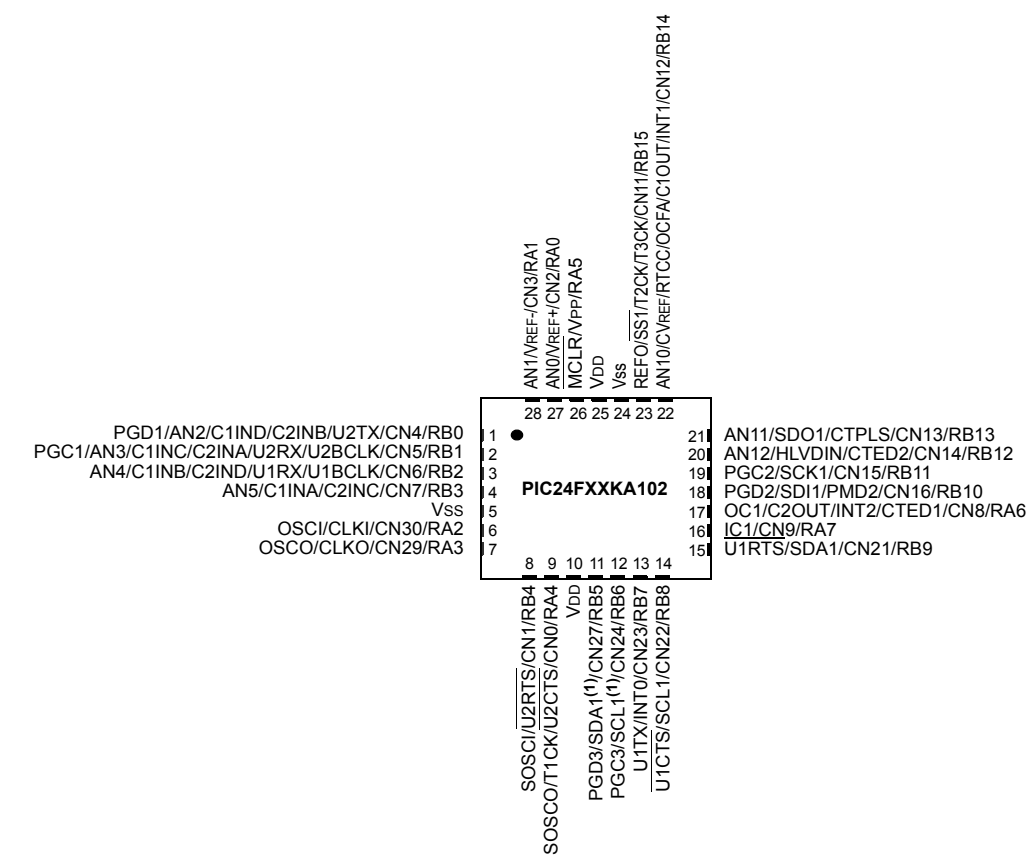
引脚图（续）



PIC24F16KA102 系列

引脚图（续）

28 引脚 QFN⁽²⁾



- 注 1: 当 I2CSEL 配置位置 1 时，相应的引脚分别用作 SDA1 和 SCL1。
2: QFN 封装的底部焊盘应连接到 Vss。

目录

1.0	器件概述	7
2.0	CPU	15
3.0	存储器构成	21
4.0	闪存程序存储器	39
5.0	数据 EEPROM 存储器	47
6.0	复位	53
7.0	中断控制器	57
8.0	振荡器配置	85
9.0	节能特性	95
10.0	I/O 端口	103
11.0	Timer1	105
12.0	Timer2/3	107
13.0	输入捕捉	113
14.0	输出比较	115
15.0	串行外设接口 (SPI)	121
16.0	I ² C™	129
17.0	通用异步收发器 (UART)	137
18.0	实时时钟和日历 (RTCC)	145
19.0	可编程循环冗余校验 (CRC) 发生器	157
20.0	高 / 低压检测 (HLVD)	161
21.0	10 位高速 A/D 转换器	163
22.0	比较器模块	173
23.0	比较器参考电压	177
24.0	充电时间测量单元 (CTMU)	179
25.0	特殊功能	183
26.0	开发支持	193
27.0	指令集汇总	197
28.0	电气特性	205
29.0	封装信息	225
附录 A: 版本历史		237
索引		239
Microchip 网站		243
变更通知客户服务		243
客户支持		243
读者反馈表		244
产品标识体系		245

PIC24F16KA102 系列

致客户

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

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

最新数据手册

欲获得本数据手册的最新版本，请查询我公司的网站：

<http://www.microchip.com>

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

勘误表

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

欲了解某一器件是否存在勘误表，请通过以下方式之一查询：

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

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

客户通知系统

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

1.0 器件概述

本文档涉及以下器件的具体信息：

- PIC24F08KA101
- PIC24F16KA101
- PIC24F08KA102
- PIC24F16KA102

PIC24F16KA102 系列引入了 Microchip 新的超低功耗 16 位单片机系列产品线，该系列器件具有丰富的外设功能部件和增强的计算性能。它还高性能应用提供了新的移植选项，这些应用可能需要超过 8 位的平台，但无需数字信号处理器的数字处理功能。

1.1 内核特性

1.1.1 16 位架构

所有 PIC24F 器件的核心都是 16 位改进型哈佛架构，第一次引入该架构的就是 Microchip 的 dsPIC® 数字信号控制器。PIC24F CPU 内核提供了众多增强功能，例如：

- 16 位数据和 24 位地址路径，能够在数据和存储空间之间传送信息
- 最大 12 MB（程序空间）和 64 KB（数据空间）的线性寻址
- 16 个元素组成的工作寄存器阵列，支持内置软件堆栈
- 17 位 x 17 位的硬件乘法器，支持整数数学运算
- 硬件支持 32 位除以 16 位的除法运算
- 指令集支持多寻址模式，已针对高级语言（如 C 语言）进行了优化
- 工作性能最高可达 16 MIPS

1.1.2 节能技术

PIC24F16KA102 系列的所有器件具有一系列能在工作时显著降低功耗的功能。主要包含以下几项：

- **动态时钟切换：**工作期间，器件时钟可由软件控制更改为 Timer1 时钟源或内部低功耗 RC 振荡器，使用户能在他们的软件设计中融入节能理念。
- **打盹模式工作原理：**当对时序敏感的应用（如串行通信）需要不间断地操作外设时，可有选择地降低 CPU 时钟速度，在不丢失信号的前提下节约功耗。
- **基于指令的节能模式：**有三种基于指令的节能模式：
 - 空闲模式：内核关闭，而外设仍然工作。
 - 休眠模式：内核和需要使用系统时钟的外设关闭，而使用自身时钟或由其他设备提供时钟的外设仍然工作。
 - 深度休眠模式：内核、外设（除 RTCC 和 DSWDT 外）、闪存和 SRAM 都关闭。

1.1.3 振荡器选项和特性

PIC24F16KA102 系列提供了 5 个不同的振荡器选项，使用户在开发应用硬件时有很大的选择范围。这些选项包括：

- 两种晶振模式，使用晶振或陶瓷谐振器。
- 两种外部时钟模式，提供 2 分频时钟输出选项。
- 两个快速内部振荡器（FRC）：一个具有标称 8 MHz 输出，另一个具有标称 500 kHz 输出。这些输出也可以由软件控制分频，提供低至 31 kHz 或 2 kHz 的时钟速度。
- 一个锁相环（Phase Lock Loop，PLL）倍频器，可在外部振荡器模式和 8 MHz FRC 振荡器模式下使用，可使时钟速度最高达到 32 MHz。
- 带有固定 31 kHz 输出的独立内部 RC 振荡器（LPRC），为对时序不敏感的应用提供低功耗选择。

PIC24F16KA102 系列

内部振荡器模块也为故障保护时钟监视器（Fail-Safe Clock Monitor, FSCM）提供稳定的参考源。该部件持续对照内部振荡器提供的参考信号监视主时钟源，使控制器能够切换到内部振荡器，从而允许持续低速工作或安全关闭应用。

1.1.4 移植方便

无论存储器容量如何，所有器件都共享相同的丰富外设，使应用的扩展和升级工作变得轻松而容易。

整个系列的引脚排列设计一致也有助于向下一代更大规模的器件移植。在具有相同引脚数的器件间移植，甚至从 20 引脚向 28 引脚器件的移植都是可以的。

PIC24F 系列的引脚兼容 dsPIC33 系列的器件，并部分兼容 PIC18 和 dsPIC30 系列器件的引脚排列。这就增强了将相对简单的应用扩展为功能强大而复杂的的能力。

1.2 其他特殊功能

- **通信：**PIC24F16KA102 系列包含了一系列串行通信外设，可处理各种应用需求。有一个 I²C™ 模块，支持主 / 从工作模式。它还包含具有内置 IrDA® 编码器 / 解码器的 UART 和一个 SPI 模块。
- **实时时钟 / 日历：**该模块在硬件中实现了带闹钟功能的全功能时钟和日历，释放了定时器资源和程序存储空间供内核应用使用。
- **10 位 A/D 转换器：**该模块具备可编程采集时间，从而不必在选择通道和启动转换之间等待一个采样周期，并且加快了采样速度。16 字深的结果缓冲区可用于在休眠模式下降低功耗，或用于在工作模式下提高吞吐量。
- **充电时间测量单元 (CTMU) 接口：**PIC24F16KA102 系列包含新的 CTMU 接口模块，可用于电容触摸传感、接近传感，还可用于精确时间测量和脉冲生成。

1.3 系列中各产品的详细说明

PIC24F16KA102 系列器件以 20 引脚和 28 引脚封装形式提供。图 1-1 给出了所有器件的一般框图。

这些器件在以下两个方面存在差异：

1. 闪存程序存储器（PIC24F08KA 器件为 8 KB，PIC24F16KA 器件为 16 KB）。
2. 可用的 I/O 引脚数和端口数（20 引脚器件有 2 个端口 18 个引脚，28 引脚器件有 2 个端口 24 个引脚）。
3. 备用 SCL 和 SDA 引脚仅在 28 引脚器件上可用，在 20 引脚器件上不可用。

本系列器件的其他功能都是相同的；表 1-1 汇总了这些功能。

按功能排序的 PIC24F16KA102 系列器件可用引脚功能列表显示在表 1-2 中。

注： 表 1-1 提供的是各外设功能的引脚位置，而非它们如何在同一引脚上复用。这一信息在数据手册第 2、3 和 4 页上的引脚图中提供。复用的功能按功能的优先级排序，优先级最高的外设列在最前面。

PIC24F16KA102 系列

表 1-1: PIC24F16KA102 系列器件的特性

特性	PIC24F08KA101	PIC24F16KA101	PIC24F08KA102	PIC24F16KA102
工作频率	DC – 32 MHz			
程序存储器（字节）	8K	16K	8K	16K
程序存储器（指令）	2816	5632	2816	5632
数据存储器（字节）	1536			
数据 EEPROM 存储器（字节）	512			
中断源（软向量 /NMI 陷阱）	30（26/4）			
I/O 端口	PORTA<6:0> PORTB<15:12, 9:7, 4, 2:0>		PORTA<7:0> PORTB<15:0>	
总 I/O 引脚数	18		24	
定时器：总数（16 位）	3			
32 位（来自成对的 16 位定时器）	1			
输入捕捉通道数	1			
输出比较 /PWM 通道数	1			
输入电平变化通知中断	17		23	
串行通信：UART	2			
SPI（3 线 /4 线）	1			
I ² C™	1			
10 位模数转换模块（输入通道数）	9			
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、REPEAT 指令、硬件陷阱、配置字不匹配（PWRT、OST 和 PLL 锁定）			
指令集	76 条基本指令，多种寻址模式			
封装	20 引脚 PDIP/SSOP/SOIC/QFN		28 引脚 SPDIP/SSOP/SOIC/QFN	

PIC24F16KA102 系列

图 1-1: PIC24F16KA102 系列一般框图

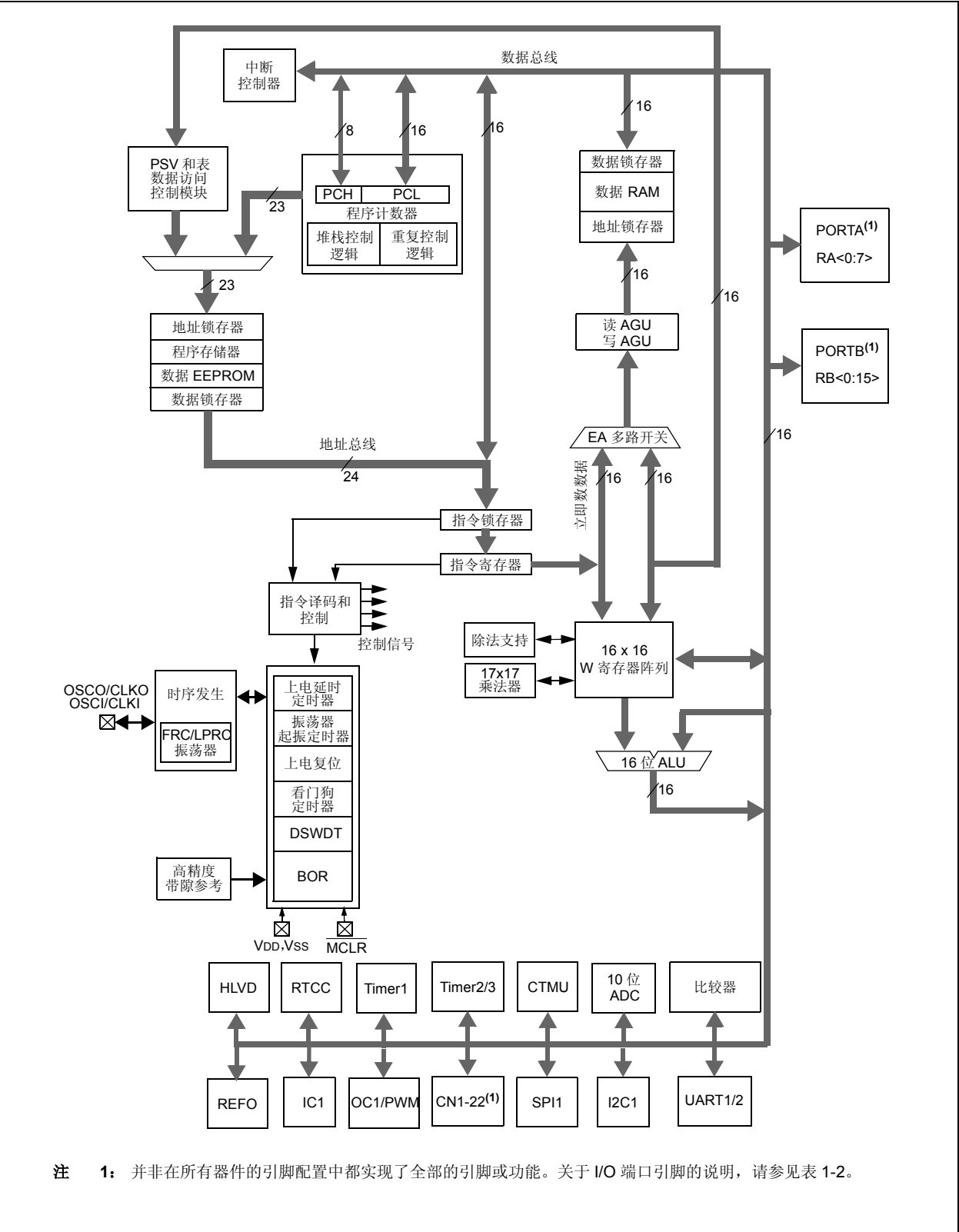


表 1-2: PIC24F16KA102 系列引脚说明

功能	引脚编号				I/O	输入 缓冲器	说明
	20 引脚 PDIP/SSOP/ SOIC	20 引脚 QFN	28 引脚 SPDIP/ SSOP/SOIC	28 引脚 QFN			
AN0	2	19	2	27	I	ANA	A/D 模拟输入
AN1	3	20	3	28	I	ANA	
AN2	4	1	4	1	I	ANA	
AN3	5	2	5	2	I	ANA	
AN4	7	4	6	3	I	ANA	
AN5	8	5	7	4	I	ANA	
AN10	17	14	25	22	I	ANA	
AN11	16	13	24	21	I	ANA	
AN12	15	12	23	20	I	ANA	
U1BCLK	6	3	6	3	O	—	UART1 IrDA® 波特率时钟
U2BCLK	5	2	5	2	O	—	UART2 IrDA 波特率时钟
C1INA	8	5	7	4	I	ANA	比较器 1 的输入 A（正极输入）
C1INB	7	4	6	3	I	ANA	比较器 1 的输入 B（负极输入选项 1）
C1INC	5	2	5	2	I	ANA	比较器 1 的输入 C（负极输入选项 2）
C1IND	4	1	4	1	I	ANA	比较器 1 的输入 D（负极输入选项 3）
C1OUT	17	14	25	22	O	—	比较器 1 的输出
C2INA	5	2	5	2	I	ANA	比较器 2 的输入 A（正极输入）
C2INB	4	1	4	1	I	ANA	比较器 2 的输入 B（负极输入选项 1）
C2INC	8	5	7	4	I	ANA	比较器 2 的输入 C（负极输入选项 2）
C2IND	7	4	6	3	I	ANA	比较器 2 的输入 D（负极输入选项 3）
C2OUT	14	11	20	17	O	—	比较器 2 的输出
CLKI	7	4	9	6	I	ANA	主时钟输入连接
CLKO	8	5	10	7	O	—	系统时钟输出

图注: ST = 施密特触发器输入缓冲器, ANA = 模拟电平输入 / 输出, I²C™ = I²C/SMBus 输入缓冲器

注 1: 当 I2C1SEL 配置位清零时的备用复用引脚。

PIC24F16KA102 系列

表 1-2: PIC24F16KA102 系列引脚说明 (续)

功能	引脚编号				I/O	输入 缓冲器	说明
	20 引脚 PDIP/SSOP/ SOIC	20 引脚 QFN	28 引脚 SPDIP/ SSOP/SOIC	28 引脚 QFN			
CN0	10	7	12	9	I	ST	电平变化中断输入
CN1	9	6	11	8	I	ST	
CN2	2	19	2	27	I	ST	
CN3	3	20	3	28	I	ST	
CN4	4	1	4	1	I	ST	
CN5	5	2	5	2	I	ST	
CN6	6	3	6	3	I	ST	
CN7	—	—	7	4	I	ST	
CN8	14	11	20	17	I	ST	
CN9	—	—	19	16	I	ST	
CN11	18	15	26	23	I	ST	
CN12	17	14	25	22	I	ST	
CN13	16	13	24	21	I	ST	
CN14	15	12	23	20	I	ST	
CN15	—	—	22	19	I	ST	
CN16	—	—	21	18	I	ST	
CN21	13	10	18	15	I	ST	
CN22	12	9	17	14	I	ST	
CN23	11	8	16	13	I	ST	
CN24	—	—	15	12	I	ST	
CN27	—	—	14	11	I	ST	
CN29	8	5	10	7	I	ST	
CN30	7	4	9	6	I	ST	
CVREF	17	14	25	22	O	ANA	比较器参考电压输出
CTED1	14	11	20	17	I	ST	CTMU 触发边沿输入 1
CTED2	15	12	23	20	I	ST	CTMU 触发边沿输入 2
CTPLS	16	13	24	21	O	—	CTMU 脉冲输出
IC1	14	11	19	16	I	ST	输入捕捉 1 输入
INT0	11	8	16	13	I	ST	外部中断输入
INT1	17	14	25	22	I	ST	
INT2	14	11	20	17	I	ST	
HLVDIN	15	12	23	20	I	ANA	HLVD 电压输入
MCLR	1	18	1	26	I	ST	主复位 (器件复位) 输入
OC1	14	11	20	17	O	—	输出比较 /PWM 输出
OCFA	17	14	25	22	I	—	输出比较故障 A
OSCI	7	4	9	6	I	ANA	主振荡器输入连接
OSCO	8	5	10	7	O	ANA	主振荡器输出连接

图注: ST = 施密特触发器输入缓冲器, ANA = 模拟电平输入 / 输出, I²C™ = I²C/SMBus 输入缓冲器

注 1: 当 I2C1SEL 配置位清零时的备用复用引脚。

PIC24F16KA102 系列

表 1-2: PIC24F16KA102 系列引脚说明 (续)

功能	引脚编号				I/O	输入 缓冲器	说明
	20 引脚 PDIP/SSOP/ SOIC	20 引脚 QFN	28 引脚 SPDIP/ SSOP/SOIC	28 引脚 QFN			
PGC1	5	2	5	2	I/O	ST	在线调试器和 ICSP™ 编程时钟
PGD1	4	1	4	1	I/O	ST	在线调试器和 ICSP 编程数据
PGC2	2	19	22	19	I/O	ST	在线调试器和 ICSP 编程时钟
PGD2	3	20	21	18	I/O	ST	在线调试器和 ICSP 编程数据
PGC3	10	7	15	12	I/O	ST	在线调试器和 ICSP 编程时钟
PGD3	9	6	14	11	I/O	ST	在线调试器和 ICSP 编程数据
RA0	2	19	2	27	I/O	ST	PORTA 数字 I/O
RA1	3	20	3	28	I/O	ST	
RA2	7	4	9	6	I/O	ST	
RA3	8	5	10	7	I/O	ST	
RA4	10	7	12	9	I/O	ST	
RA5	1	18	1	26	I/O	ST	
RA6	14	11	20	17	I/O	ST	
RA7	—	—	19	16	I/O	ST	
RB0	4	1	4	1	I/O	ST	PORTB 数字 I/O
RB1	5	2	5	2	I/O	ST	
RB2	6	3	6	3	I/O	ST	
RB3	—	—	7	4	I/O	ST	
RB4	9	6	11	8	I/O	ST	
RB5	—	—	14	11	I/O	ST	
RB6	—	—	15	12	I/O	ST	
RB7	11	8	16	13	I/O	ST	
RB8	12	9	17	14	I/O	ST	
RB9	13	10	18	15	I/O	ST	
RB10	—	—	21	18	I/O	ST	
RB11	—	—	22	19	I/O	ST	
RB12	15	12	23	20	I/O	ST	
RB13	16	13	24	21	I/O	ST	
RB14	17	14	25	22	I/O	ST	
RB15	18	15	26	23	I/O	ST	
REFO	18	15	26	23	O	—	参考时钟输出
RTCC	17	14	25	22	O	—	实时时钟闹钟输出
SCK1	15	12	22	19	I/O	ST	SPI1 串行时钟输入 / 输出
SCL1	12	9	17, 15 ⁽¹⁾	14, 12 ⁽¹⁾	I/O	I ² C	I2C1 同步串行时钟输入 / 输出
SDA1	13	10	18, 14 ⁽¹⁾	15, 11 ⁽¹⁾	I/O	I ² C	I2C1 数据输入 / 输出
SDI1	17	14	21	18	I	ST	SPI1 串行数据输入
SDO1	16	13	24	21	O	—	SPI1 串行数据输出
SOSCI	9	6	11	8	I	ANA	辅助振荡器输入
SOSCO	10	7	12	9	O	ANA	辅助振荡器输出
SS1	18	15	26	23	I/O	ST	从选择输入 / 帧选择输出 (SPI1)

图注: ST = 施密特触发器输入缓冲器, ANA = 模拟电平输入 / 输出, I²C™ = I²C/SMBus 输入缓冲器

注 1: 当 I2C1SEL 配置位清零时的备用复用引脚。

PIC24F16KA102 系列

表 1-2: PIC24F16KA102 系列引脚说明 (续)

功能	引脚编号				I/O	输入 缓冲器	说明
	20 引脚 PDIP/SSOP/ SOIC	20 引脚 QFN	28 引脚 SPDIP/ SSOP/SOIC	28 引脚 QFN			
T1CK	10	7	12	9	I	ST	Timer1 时钟
T2CK	18	15	26	23	I	ST	Timer2 时钟
T3CK	18	15	26	23	I	ST	Timer3 时钟
U1CTS	12	9	17	14	I	ST	UART1 允许发送输入
U1RTS	13	10	18	15	O	—	UART1 请求发送输出
U1RX	6	3	6	3	I	ST	UART1 接收
U1TX	11	8	16	13	O	—	UART1 发送输出
VDD	20	17	13, 28	10, 25	P	—	外设数字逻辑和 I/O 引脚的正电源
VPP	1	18	1	26	P	—	编程模式入口电压
VREF-	3	20	3	28	I	ANA	A/D 和比较器参考电压 (低电压) 输入
VREF+	2	19	2	27	I	ANA	A/D 和比较器参考电压 (高电压) 输入
VSS	19	16	8, 27	5, 24	P	—	逻辑和 I/O 引脚的参考地

图注: ST = 施密特触发器输入缓冲器, ANA = 模拟电平输入 / 输出, I²C™ = I²C/SMBus 输入缓冲器

注 1: 当 I2C1SEL 配置位清零时的备用复用引脚。

2.0 CPU

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于 CPU 的更多信息，请参见《PIC24F 系列参考手册》的“第 2 章 CPU”（DS39703A_CN）。

PIC24F CPU 采用 16 位（数据）的改进型哈佛架构，具有增强指令集以及带有长度可变操作码字段的 24 位指令字。程序计数器（Program Counter，PC）为 23 位宽，可以寻址最大 4M 指令字的用户程序存储空间。单周期指令预取机制可帮助维持吞吐量并使指令的执行具有预测性。除了改变程序流的指令、双字传送（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 指令支持无开销的程序循环结构，该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）作为软件堆栈指针（Software Stack Pointer，SSP），用于中断和调用。

可以选择将数据存储空间的高 32 KB 映射到由 8 位程序空间可视性页地址（Program Space Visibility Page Address，PSVPAG）寄存器定义的程序存储器或数据 EEPROM 存储器的任何 16K 字边界内的程序空间内。程序空间到数据空间的映射功能让任何指令都能像访问数据空间一样访问程序空间。

指令集架构（Instruction Set Architecture，ISA）在 PIC18 的指令集架构基础上显著增强了，但仍保持了可接受程度的后向兼容性。直接支持或通过简单的宏支持所有 PIC18 指令和寻址模式。许多 ISA 增强功能都是为了提高编译器效率。

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寻址和多种寄存器间接寻址模式。每组都提供最多 7 种寻址模式。指令根据其功能要求，与预定义的寻址模式相关联。

对于大多数指令，内核能够在每个指令周期内执行一次数据（或程序数据）存储器读取、一次工作寄存器（数据）读取、一次数据存储器写入以及一次程序（指令）存储器读取操作。因此，支持 3 操作数指令，允许在单个周期内执行三操作数运算（即，A + B = C）。

包含了一个高速 17 位 x 17 位乘法器，显著提高了内核算术能力和吞吐量。乘法器支持有符号、无符号和混合模式的 16 位 x 16 位或 8 位 x 8 位整数乘法。所有乘法指令都在一个周期内执行。

16 位 ALU 用支持迭代非还原除法算法的整数除法支持硬件进行了增强。它和 REPEAT 指令循环机制一起（还可以加上迭代除法指令）使用，支持用 16 位整数除 32 位（或 16 位）的有符号和无符号除法。所有除法运算都需要 19 个周期完成，但可以在任何周期边界上中断。

PIC24F 有一个向量异常机制，具有最多 8 个不可屏蔽陷阱源和最多 118 个中断源。每个中断源都可分配为 7 个优先级之一。

图 2-1 给出了 CPU 的框图。

2.1 编程模型

图 2-2 给出了 PIC24F 的编程模型。编程模型中的所有寄存器都是存储器映射的，可以直接用指令控制。

表 2-1 中提供了每个寄存器的说明。所有与编程模型相关的寄存器都是存储器映射的。

PIC24F16KA102 系列

图 2-1: PIC24F CPU 内核框图

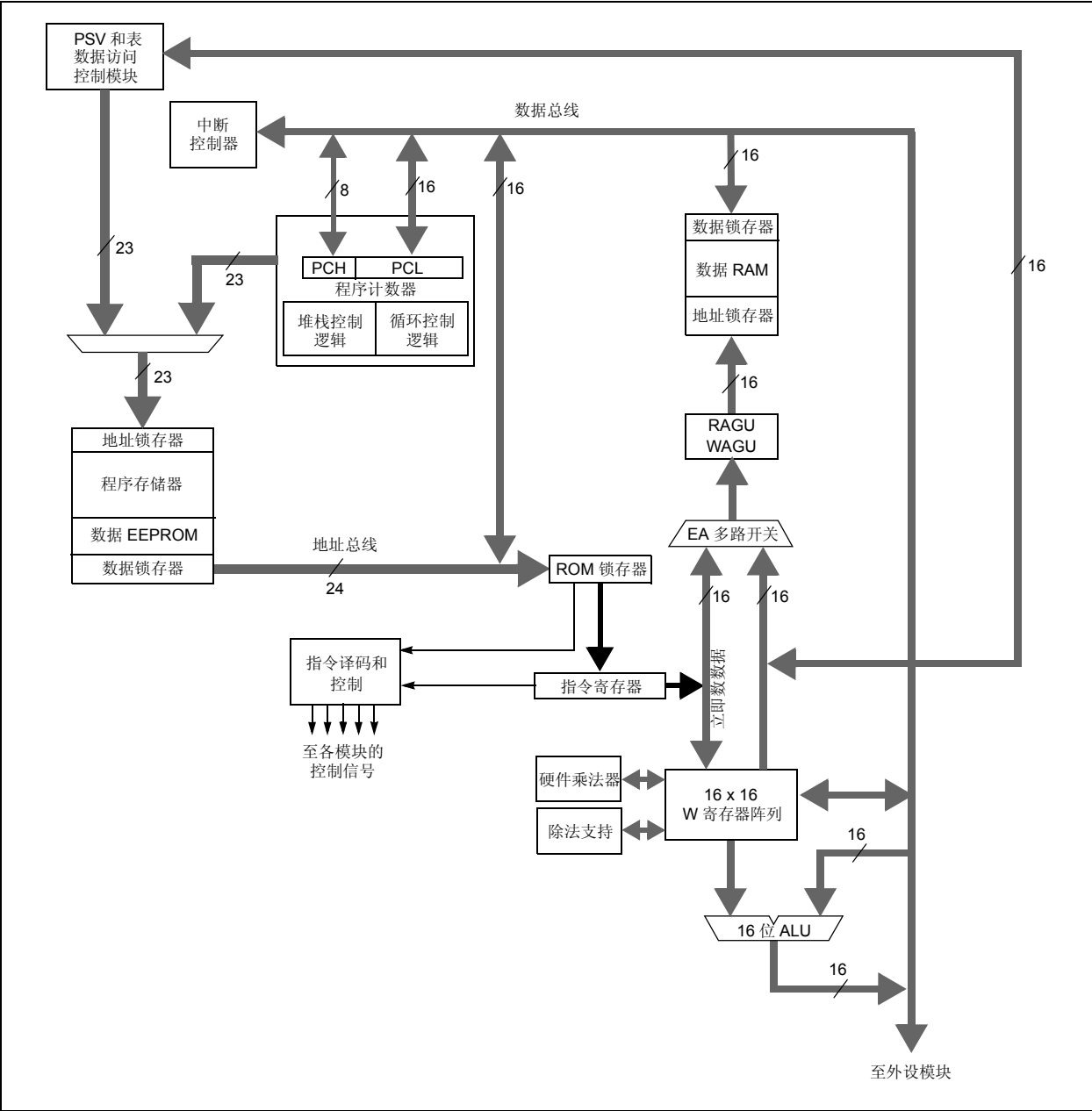
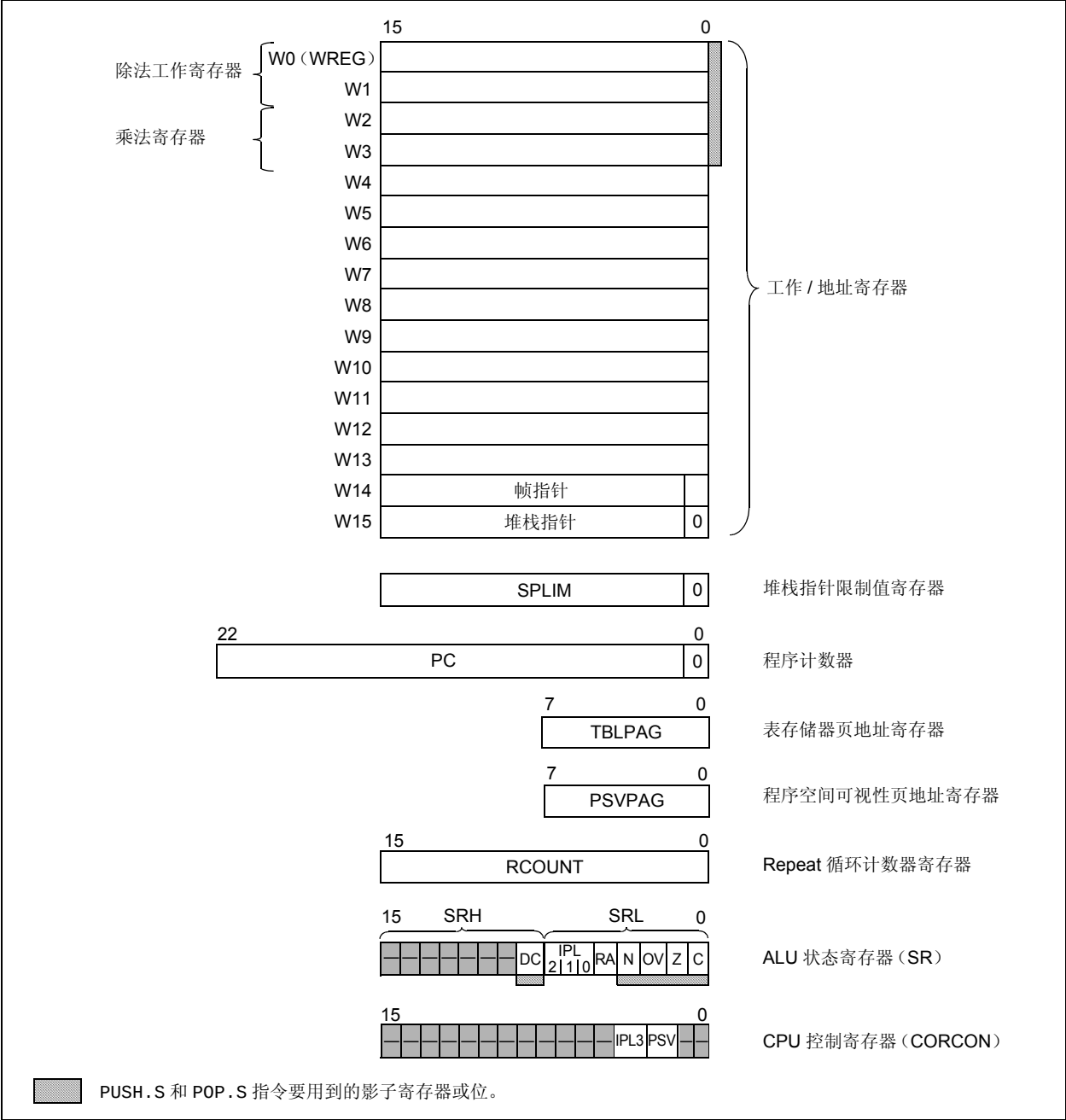


表 2-1: CPU 内核寄存器

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	堆栈指针限制值寄存器
TBLPAG	表存储器页地址寄存器
PSVPAG	程序空间可视性页地址寄存器
RCOUNT	Repeat 循环计数器寄存器
CORCON	CPU 控制寄存器

图 2-2: 编程模型



PIC24F16KA102 系列

2.2 CPU 控制寄存器

寄存器 2-1: SR: ALU 状态寄存器

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

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

图注:	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 = POR 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 个低位 (对于字节大小的数据) 或第 8 个低位 (对于字大小的数据) 发生了进位 0 = 结果的第 4 个或第 8 个低位未发生进位
bit 7-5	IPL<2:0>: CPU 中断优先级状态位 ^(1,2) 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 循环活动位 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 (零或正值)
bit 2	OV: ALU 溢出标志位 1 = 有符号 (二进制补码) 算术运算中发生溢出 (本次运算) 0 = 未发生溢出
bit 1	Z: ALU 全零标志位 1 = 影响 Z 位的任何运算在过去某时已将该位置 1 0 = 影响 Z 位的最近一次运算已将该位清零 (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 (Most Significant bit, MSb) 发生了进位 0 = 结果的最高位 (MSb) 未发生进位

- 注 1: 当 NSTDIS (INTCON1<15>) = 1 时, IPL 状态位是只读的。
- 2: IPL 状态位与 IPL3 位 (CORCON<3>) 组合形成 CPU 中断优先级 (Interrupt Priority Level, IPL)。当 IPL3 = 1 时, 括号中的值表示 IPL。

寄存器 2-2: CORCON: CPU 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, HSC	R/W-0	U-0	U-0
—	—	—	—	IPL3 ⁽¹⁾	PSV	—	—
bit 7						bit 0	

图注:	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 = POR 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:** 数据空间中程序空间可视性使能位
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, 并能进行加法、减法、移位和逻辑运算。除非特别指明, 算术运算一般采用二进制补码方式进行。根据不同的运算, 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。在减法运算中, C 和 DC 状态位分别作为借位位和半借位位。

根据所使用的指令模式, ALU 可执行 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 的输出数据可被写入 W 寄存器阵列或数据存储器单元。

PIC24F CPU 融入了对乘法和除法的硬件支持。它带有专门的硬件乘法器以及支持 16 位除数除法的硬件。

2.3.1 乘法器

- ALU 包含一个高速 17 位 x 17 位乘法器。它支持几种乘法模式下的无符号、有符号或混合符号运算:
- 16 位 x 16 位有符号
 - 16 位 x 16 位无符号
 - 16 位有符号 x 5 位 (立即数) 无符号
 - 16 位无符号 x 16 位无符号
 - 16 位无符号 x 5 位 (立即数) 无符号
 - 16 位无符号 x 16 位有符号
 - 8 位无符号 x 8 位无符号

PIC24F16KA102 系列

2.3.2 除法器

除法模块支持具有以下数据长度的 32 位 /16 位和 16 位 /16 位有符号和无符号整数除法运算：

1. 32 位有符号 /16 位有符号除法
2. 32 位无符号 /16 位无符号除法
3. 16 位有符号 /16 位有符号除法
4. 16 位无符号 /16 位无符号除法

所有除法指令的商都被放在 W0 中，余数放在 W1 中。16 位有符号和无符号 DIV 指令可为 16 位除数指定任一 W 寄存器 (Wn)，为 32 位被除数指定任意两个连续的 W 寄存器 (W(m + 1):Wm)。除法运算中处理除数的每一位需要一个周期，因此 32 位 /16 位和 16 位 /16 位指令的执行周期数相同。

2.3.3 多位移位支持

PIC24F ALU 支持单位和单周期、多位算术和逻辑移位。多位移位使用移位器模块实现，能够在单个周期内执行最多 15 位的算术右移或最多 15 位的左移。所有多位移位指令都只支持操作数源和结果目标的寄存器直接寻址模式。

下面的表 2-2 中提供了使用移位操作的指令的完整汇总。

表 2-2: 使用单位和多位移位操作的指令

指令	说明
ASR	将源寄存器算术右移 1 位或多位。
SL	将源寄存器左移 1 位或多位。
LSR	将源寄存器逻辑右移 1 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
数据存储空间以及总线机制。这一架构同时还允许在代
码执行过程中从数据空间直接访问程序存储器。

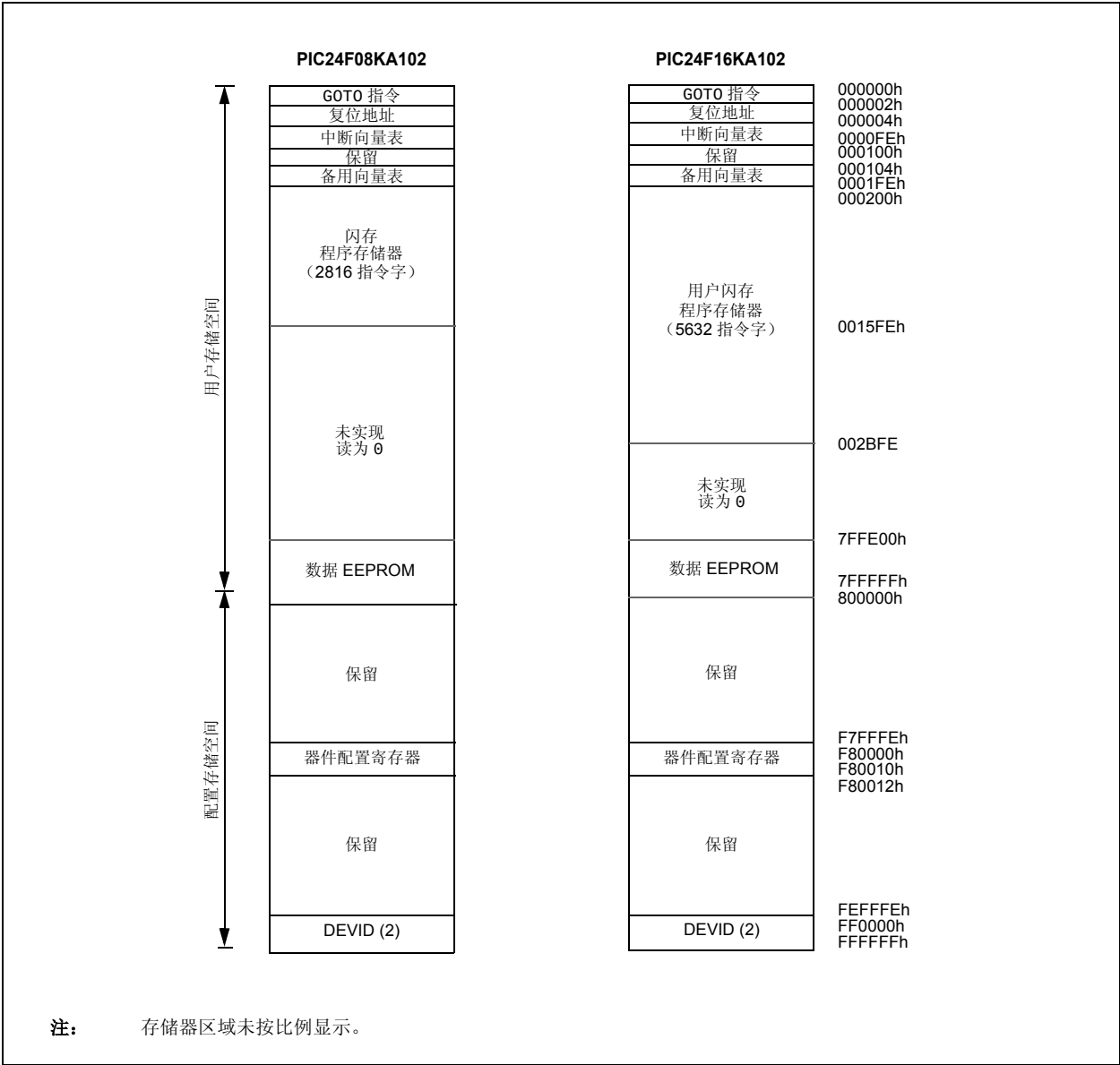
用户只能访问程序存储空间的低半地址部分（地址范围
为 000000h 至 7FFFFFFh）。使用 TBLRD/TBLWT 指令
时，情况有所不同，这两条指令采用 TBLPAG<7> 以允
许访问配置存储空间中的配置位和器件 ID。

图 3-1 给出了 PIC24F16KA102 系列器件的存储器映射。

3.1 程序地址空间

PIC24F 器件的程序地址存储空间可存储 4M 个指令
字。可通过由程序执行过程中 23 位程序计数器（PC）、
表操作或数据空间重映射得到的 24 位值寻址这一空
间，如第 3.3 节“程序存储空间与数据存储空间的内
口”所示。

图 3-1: PIC24F16KA102 系列器件的程序存储空间映射



PIC24F16KA102 系列

3.1.1 程序存储器构成

程序存储空间由可字寻址的块构成。虽然它被视为24位宽，但将程序存储器的每个地址视作一个低位字和一个高位字的组合更加合理，其中高位字的高字节部分没有实现。低位字的地址始终为偶数，而高位字的地址为奇数（图3-2）。

程序存储器地址始终在低位字处按字对齐，并且在代码执行过程中地址将递增或递减2。这种寻址模式与数据存储空间寻址兼容，且为访问程序存储空间中的数据提供了可能。

3.1.2 存储器硬编码向量

所有 PIC24F 器件中从 00000h 到 000200h 之间的地址空间都是保留的，用来存储硬编码的程序执行向量。提供了一个硬件复位向量将代码执行从器件复位时 PC 的默认值重新定位到代码实际起始处。用户可在地址 000000h 处编写一条 GOTO 指令以将代码的实际起始地址设置为 000002h。

PIC24F 器件也具有两个中断向量表，地址分别为从 000004h 到 0000FFh 和 000104h 到 0001FFh。这两个向量表允许使用不同的中断服务程序（Interrupt Service Routine, ISR）处理每个器件中断源。关于中断向量表更详细的讨论，请参见第 7.1 节“中断向量表（IVT）”。

3.1.3 数据 EEPROM

在 PIC24F16KA102 系列中，数据 EEPROM 被映射到用户程序存储空间的顶部，从地址 7FFE00 处开始，最高延伸到地址 7FFFFFFh 处。

数据 EEPROM 存储器的宽度为 16 位，深度为 256 个字。该存储器使用类似于用于用户代码存储区的表读和表写操作进行访问。

3.1.4 器件配置字

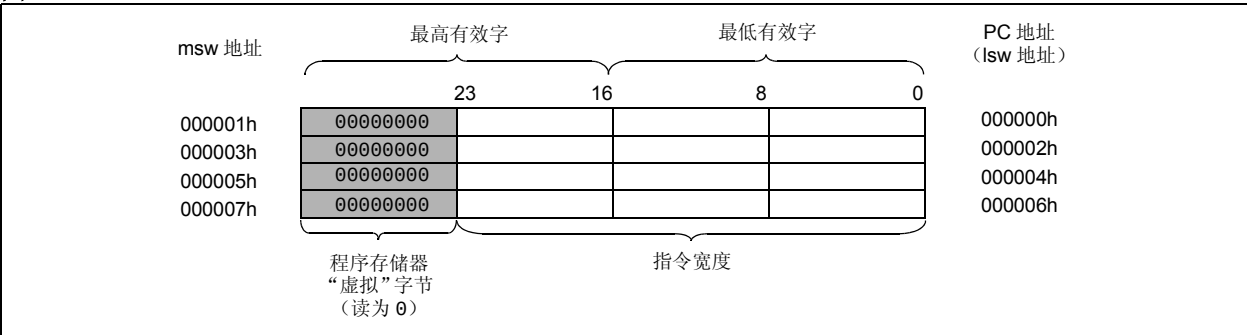
PIC24F16KA102 系列的器件配置字的地址如表 3-1 所示。它们在存储器映射中的位置如图 3-1 所示。

关于器件配置字的更多信息，请参见第 25.1 节“配置位”。

表 3-1: PIC24F16KA102 系列器件的器件配置字

配置字	配置字地址
FBS	F80000
FGS	F80004
FOSCSEL	F80006
FOSC	F80008
FWDT	F8000A
FPOR	F8000C
FICD	F8000E
FDS	F80010

图 3-2: 程序存储器构成



3.2 数据地址空间

PIC24F 内核具有独立的 16 位宽数据存储空间，可将其作为一个线性空间寻址。使用两个地址发生单元（Address Generation Unit，AGU）分别对数据空间执行读写操作。数据存储空间映射如图 3-3 所示。

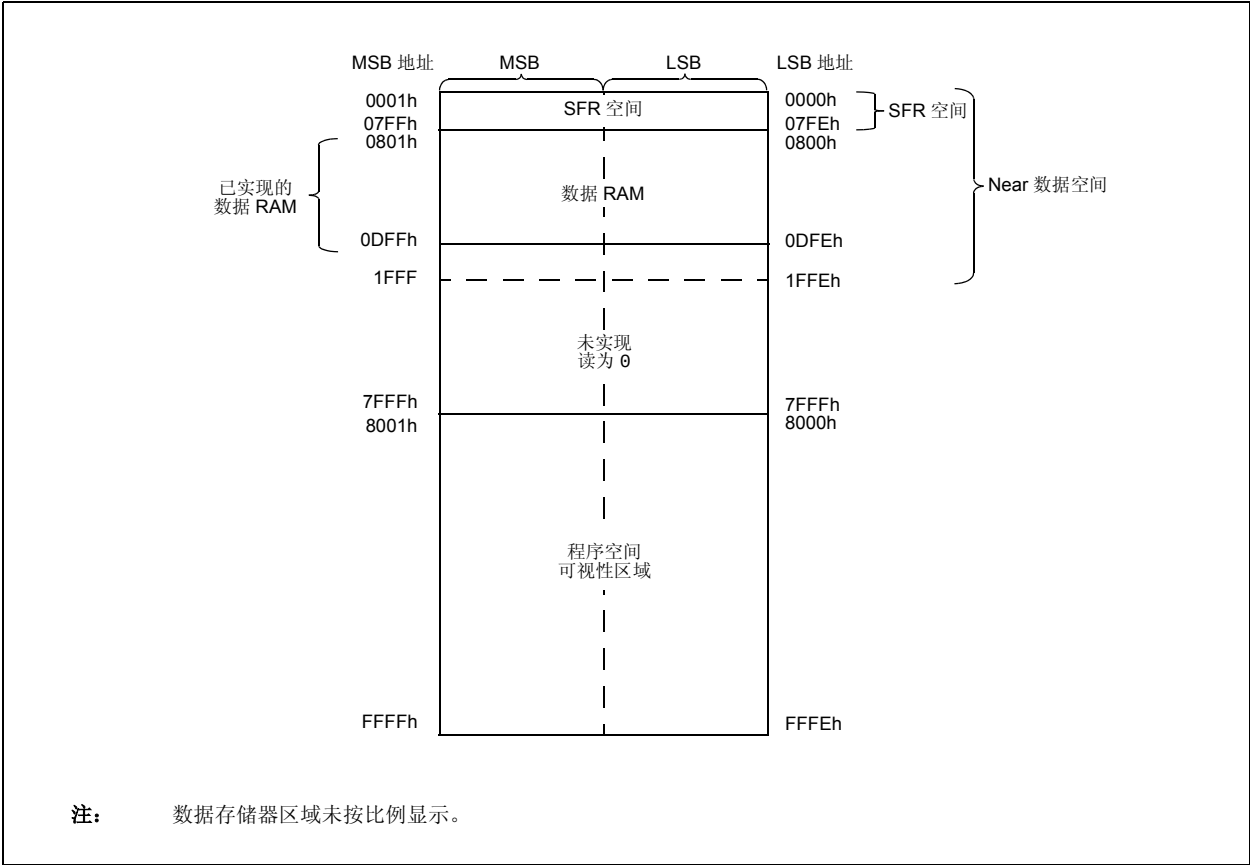
数据存储空间中的所有有效地址（Effective Address，EA）均为 16 位宽，并且指向数据空间内的字节。这种构成方式使得数据空间的地址范围为 64 KB 即 32K 字。数据存储空间的低半地址部分（即当 EA<15> = 0 时）用作实现的存储单元，而高半地址部分（EA<15> = 1）则保留为程序空间可视性（Program Space Visibility，PSV）区域（见第 3.3.3 节“使用程序空间可视性读程序存储器中的数据”）。

PIC24F16KA102 系列器件共实现了 768 字的数据存储器。如果 EA 指向了该区域以外的存储单元，则将返回一个全零的字或字节。

3.2.1 数据空间宽度

数据存储空间由可字节寻址的 16 位宽的块构成。在数据存储器 and 寄存器中的数据是以 16 位字为单位对齐的，但所有数据空间 EA 都将解析为字节。每个字的低字节（Least Significant Byte，LSB）部分具有偶地址，而高字节（Most Significant Byte，MSB）部分则具有奇地址。

图 3-3: PIC24F16KA102 系列器件的数据存储空间映射



PIC24F16KA102 系列

3.2.2 数据存储器构成和对齐方式

为维持与 PIC® 器件的后向兼容性和提高数据存储空间的使用效率，PIC24F 指令集同时支持字和字节操作。字节访问会在内部对按字对齐的存储空间的所有 EA 进行计算调整。例如，对于执行后修改寄存器间接寻址模式 [Ws++] 的结果，字节操作时，内核将其识别为值 Ws + 1；而字操作时，内核将其识别为值 Ws + 2。

使用任何 EA 的 LSB 来确定要选取的字节，数据字节读操作将读取包含字节的整个字。选定的字节被放在数据总线的 LSB 处。这就是说，数据存储器 and 寄存器被组织为两个并行的字节宽的实体，它们共享（字）地址译码，但写入线相互独立。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侧。

所有字访问必须按偶地址对齐。不支持不对齐的字数据读取操作，所以在混合字节和字操作时，或者从 8 位 MCU 代码移植时，必须要小心。如果试图进行不对齐的读或写操作，将产生地址错误陷阱。如果在读操作时产生错误，正在执行的指令将完成；而如果在写操作时产生错误，指令仍将执行，但不会进行写入。无论是哪种情况都将产生陷阱，从而系统和 / 或用户能够检查地址错误发生之前的机器状态。

所有装入 W 寄存器的字节都将装入 W 寄存器的低字节（LSB），W 寄存器的高字节（MSB）不变。

提供了一条符号扩展（SE）指令，允许用户把 8 位有符号数据转换为 16 位有符号值。或者，对于 16 位无符号数据，用户可以通过在适当地址处执行一条零扩展（ZE）指令清零任何 W 寄存器的 MSB。

尽管大多数指令能够对字或字节大小的数据进行操作，但要注意的是，部分指令只对字大小的数据进行操作。

3.2.3 NEAR 数据空间

在 0000h 和 1FFFh 之间的 8 KB 的区域被称为 Near 数据空间。可以使用所有存储器直接寻址指令中的 13 位绝对地址字段直接寻址这一空间中的存储单元。可以间接寻址数据空间的其余部分。此外，还可以使用 MOV 指令寻址整个数据空间，支持使用 16 位地址字段的存储器直接寻址（Memory Direct Addressing, MDA）。对于 PIC24F16KA102 系列器件，整个已实现的数据存储器位于 Near 数据空间（Near Data Space, NDS）中。

3.2.4 SFR 空间

Near 数据空间的前 2 KB 单元（从 0000h 到 07FFh）主要被特殊功能寄存器（Special Function Register, SFR）占用。PIC24F 内核和外设模块使用这些寄存器来控制器件的工作。

SFR 被分配给受其控制的模块，通常一个模块会使用一组 SFR。大部分 SFR 空间包含未用的地址；它们读为 0。显示实际实现的 SFR 位置的 SFR 空间图表如表 3-2 所示。每个已实现区域指示至少有一个地址被实现为 SFR 的 32 字节区域。表 3-3 到表 3-23 给出了已实现 SFR 及其地址的完整列表。

表 3-2: SFR 数据空间的已实现区域

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	I ² C™	UART	SPI		—	—	I/O	
300h	ADC/CMTU		—	—	—	—	—	—
400h	—	—	—	—	—	—	—	—
500h	—	—	—	—	—	—	—	—
600h	—	RTC/ 比较	CRC	—	—			
700h	—	—	系统 /DS/HLVD	NVM/PMD	—	—	—	—

图注： — = 此存储块中未实现的 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	所有复位时的状态
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	堆栈指针限制值寄存器																xxxx
PCL	002E	程序计数器低字节寄存器																0000
PCH	0030	—	—	—	—	—	—	—	—	程序计数器高字节寄存器								0000
TBLPAG	0032	—	—	—	—	—	—	—	—	表存储器页地址寄存器								0000
PSVPAG	0034	—	—	—	—	—	—	—	—	程序空间可视性页地址寄存器								0000
RCOUNT	0036	REPEAT 循环计数器寄存器																xxxx
SR	0042	—	—	—	—	—	—	—	DC	IPL2	IPL1	IPL0	RA	N	OV	Z	C	0000
CORCON	0044	—	—	—	—	—	—	—	—	—	—	—	—	IPL3	PSV	—	—	0000
DISICNT	0052	—	—	禁止中断计数器寄存器														xxxx

图注： — = 未实现，读为 0。复位值以十六进制显示。

表 3-4: 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	所有复位时的状态
CNEN1	0060	CN15IE ⁽¹⁾	CN14IE	CN13IE	CN12IE	CN11IE	—	CN9IE	CN8IE	CN7IE ⁽¹⁾	CN6IE	CN5IE	CN4IE	CN3IE	CN2IE	CN1IE	CN0IE	0000
CNEN2	0062	—	CN30IE	CN29IE	—	CN27IE ⁽¹⁾	—	—	CN24IE ⁽¹⁾	CN23IE	CN22IE	CN21IE	—	—	—	—	CN16IE ⁽¹⁾	0000
CNPU1	0068	CN15PUE ⁽¹⁾	CN14PUE	CN13PUE	CN12PUE	CN11PUE	—	CN9PUE	CN8PUE	CN7PUE ⁽¹⁾	CN6PUE	CN5PUE	CN4PUE	CN3PUE	CN2PUE	CN1PUE	CN0PUE	0000
CNPU2	006A	—	CN30PUE	CN29PUE	—	CN27PUE ⁽¹⁾	—	—	CN24PUE ⁽¹⁾	CN23PUE	CN22PUE	CN21PUE	—	—	—	—	CN16PUE ⁽¹⁾	0000
CNPD1	0070	CN15PDE ⁽¹⁾	CN14PDE	CN13PDE	CN12PDE	CN11PDE	—	CN9PDE	CN8PDE	CN7PDE ⁽¹⁾	CN6PDE	CN5PDE	CN4PDE	CN3PDE	CN2PDE	CN1PDE	CN0PDE	0000
CNPD2	0072	—	CN30PDE	CN29PDE	—	CN27PDE ⁽¹⁾	—	—	CN24PDE ⁽¹⁾	CN23PDE	CN22PDE	CN21PDE	—	—	—	—	CN16PDE ⁽¹⁾	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

注 1: 这些位在 20 引脚器件上未实现。

表 3-5: 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	所有复位时的状态
INTCON1	0080	NSTDIS	—	—	—	—	—	—	—	—	—	—	MATHERR	ADDRERR	STKERR	OSCFail	—	0000
INTCON2	0082	ALTIVT	DISI	—	—	—	—	—	—	—	—	—	—	—	INT2EP	INT1EP	INT0EP	0000
IFS0	0084	NVMIF	—	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPF1IF	T3IF	T2IF	—	—	—	T1IF	OC1IF	IC1IF	INT0IF	0000
IFS1	0086	U2TXIF	U2RXIF	INT2IF	—	—	—	—	—	—	—	—	INT1IF	CNIF	CMIF	MI2C1IF	SI2C1IF	0000
IFS3	008A	—	RTCIF	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
IFS4	008C	—	—	CTMUIF	—	—	—	—	HLVDIF	—	—	—	—	CRCIF	U2ERIF	U1ERIF	—	0000
IEC0	0094	NVMIE	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPF1IE	T3IE	T2IE	—	—	—	T1IE	OC1IE	IC1IE	INT0IE	0000
IEC1	0096	U2TXIE	U2RXIE	INT2IE	—	—	—	—	—	—	—	—	INT1IE	CNIE	CMIE	—	—	0000
IEC3	009A	—	RTCIE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000
IEC4	009C	—	—	CTMUIE	—	—	—	—	HLVDIE	—	—	—	—	CRCIE	U2ERIE	U1ERIE	—	0000
IPC0	00A4	—	T1IP2	T1IP1	T1IP0	—	OC1IP2	OC1IP1	OC1IP0	—	IC1IP2	IC1IP1	IC1IP0	—	INT0IP2	INT0IP1	INT0IP0	4444
IPC1	00A6	—	T2IP2	T2IP1	T2IP0	—	—	—	—	—	—	—	—	—	—	—	—	4444
IPC2	00A8	—	U1RXIP2	U1RXIP1	U1RXIP0	—	SPI1IP2	SPI1IP1	SPI1IP0	—	SPF1IP2	SPF1IP1	SPF1IP0	—	T3IP2	T3IP1	T3IP0	4444
IPC3	00AA	—	NVMIP2	NVMIP1	NVMIP0	—	—	—	—	—	AD1IP2	AD1IP1	AD1IP0	—	U1TXIP2	U1TXIP1	U1TXIP0	4044
IPC4	00AC	—	CNIP2	CNIP1	CNIP0	—	CMIP2	CMIP1	CMIP0	—	MI2C1P2	MI2C1P1	MI2C1P0	—	SI2C1P2	SI2C1P1	SI2C1P0	4444
IPC5	00AE	—	—	—	—	—	—	—	—	—	—	—	—	—	INT1IP2	INT1IP1	INT1IP0	0004
IPC7	00B2	—	U2TXIP2	U2TXIP1	U2TXIP0	—	U2RXIP2	U2RXIP1	U2RXIP0	—	INT2IP2	INT2IP1	INT2IP0	—	—	—	—	4440
IPC15	00C2	—	—	—	—	—	RTCIP2	RTCIP1	RTCIP0	—	—	—	—	—	—	—	—	0400
IPC16	00C4	—	CRCIP2	CRCIP1	CRCIP0	—	U2ERIP2	U2ERIP1	U2ERIP0	—	U1ERIP2	U1ERIP1	U1ERIP0	—	—	—	—	4440
IPC18	00C8	—	—	—	—	—	—	—	—	—	—	—	—	—	HLVDIP2	HLVDIP1	HLVDIP0	0004
IPC19	00CA	—	—	—	—	—	—	—	—	—	CTMUIP2	CTMUIP1	CTMUIP0	—	—	—	—	0040
INTTREG	00E0	CPUIRQ	—	VHOLD	—	ILR3	ILR2	ILR1	ILR0	—	VECNUM6	VECNUM5	VECNUM4	VECNUM3	VECNUM2	VECNUM1	VECNUM0	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

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	所有复位时的状态
TMR1	0100	Timer1 寄存器																0000
PR1	0102	Timer1 周期寄存器																FFFF
T1CON	0104	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	TSYNC	TCS	—	0000
TMR2	0106	Timer2 寄存器																0000
TMR3HLD	0108	Timer3 保持寄存器（仅适用于 32 位定时器操作）																0000
TMR3	010A	Timer3 寄存器																0000
PR2	010C	Timer2 周期寄存器																FFFF
PR3	010E	Timer3 周期寄存器																FFFF
T2CON	0110	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	T32	—	TCS	—	0000
T3CON	0112	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	—	TCS	—	0000

图注： — = 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	所有复位时的状态
IC1BUF	0140	输入捕捉 1 寄存器																FFFF
IC1CON	0142	—	—	ICSIDL	—	—	—	—	—	ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000

图注： — = 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	所有复位时的状态
OC1RS	0180	输出比较 1 辅助寄存器																FFFF
OC1R	0182	输出比较 1 寄存器																FFFF
OC1CON	0184	—	—	OCSIDL	—	—	—	—	—	—	—	—	OCFLT	OCTSEL	OCM2	OCM1	OCM0	0000

图注： — = 未实现，读为 0。复位值以十六进制显示。

表 3-9: I²C™ 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	所有 复位时 的状态
I2C1RCV	0200	—	—	—	—	—	—	—	—	I2C1 接收寄存器								0000
I2C1TRN	0202	—	—	—	—	—	—	—	—	I2C1 发送寄存器								00FF
I2C1BRG	0204	—	—	—	—	—	—	—	I2C1 波特率发生器寄存器									0000
I2C1CON	0206	I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN	GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	1000
I2C1STAT	0208	ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10	IWCOL	I2COV	D $\overline{A}$	P	S	R $\overline{W}$	RBF	TBF	0000
I2C1ADD	020A	—	—	—	—	—	—	I2C1 地址寄存器										0000
I2C1MSK	020C	—	—	—	—	—	—	AMSK9	AMSK8	AMSK7	AMSK6	AMSK5	AMSK4	AMSK3	AMSK2	AMSK1	AMSK0	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

表 3-10: UART 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	所有复位时的状态
U1MODE	0220	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
U1STA	0222	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U1TXREG	0224	—	—	—	—	—	—	—	UART1 发送寄存器									0000
U1RXREG	0226	—	—	—	—	—	—	—	UART1 接收寄存器									0000
U1BRG	0228	波特率发生器预分频器寄存器																0000
U2MODE	0230	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL	0000
U2STA	0232	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U2TXREG	0234	—	—	—	—	—	—	—	UART2 发送寄存器									0000
U2RXREG	0236	—	—	—	—	—	—	—	UART2 接收寄存器									0000
U2BRG	0238	波特率发生器预分频器																0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

表 3-11: SPI 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	所有复位时的状态
SPI1STAT	0240	SPIEN	—	SPIIDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0	SRMPT	SPIROV	SR1MPT	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 / 接收缓冲区																0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

表 3-12: 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 ⁽¹⁾	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	所有复位时的状态
TRISA	02C0	—	—	—	—	—	—	—	—	TRISA7 ⁽⁴⁾	TRISA6	—	TRISA4	TRISA3 ^(5,6)	TRISA2 ⁽⁵⁾	TRISA1	TRISA0	00DF
PORTA	02C2	—	—	—	—	—	—	—	—	RA7 ⁽⁴⁾	RA6	RA5	RA4 ⁽³⁾	RA3 ^(5,6)	RA2 ⁽⁵⁾	RA1 ⁽²⁾	RA0 ⁽²⁾	xxxx
LATA	02C4	—	—	—	—	—	—	—	—	LATA7 ⁽⁴⁾	LATA6	—	LATA4	LATA3 ^(5,6)	LATA2 ⁽⁵⁾	LATA1	LATA0	xxxx
ODCA	02C6	—	—	—	—	—	—	—	—	ODA7 ⁽⁴⁾	ODA6	—	ODA4	ODA3 ^(5,6)	ODA2 ⁽⁵⁾	ODA1	ODA0	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

- 注
- 1: 仅当 MCLRE = 0 时, 位才可用。
 - 2: 当 PGC2/PGD2 引脚上的调试功能有效时, 对 RA1 和 RA0 的读操作将导致读为 0。
 - 3: 当 PGC3/PGD3 引脚上的调试功能有效时, 对 RA4 的读操作将导致读为 0。
 - 4: 这些位在 20 引脚器件上未实现。
 - 5: 仅当禁止主振荡器 (POSCMD1:POSCMD0 = 00) 时, 位才可用; 否则读为 0。
 - 6: 仅当禁止主振荡器或选取 EC 模式 (POSCMD1:POSCMD0 = 00 或 11) 且禁止 CLK0 (OSCIOFNC = 0) 时, 位才可用; 否则读为 0。

表 3-13: 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	所有复位时的状态
TRISB	02C8	TRISB15	TRISB14	TRISB13	TRISB12	TRISB11 ⁽³⁾	TRISB10 ⁽³⁾	TRISB9	TRISB8	TRISB7	TRISB6 ⁽³⁾	TRISB5 ⁽³⁾	TRISB4	TRISB3 ⁽³⁾	TRISB2	TRISB1	TRISB0	FFFF
PORTB	02CA	RB15	RB14	RB13	RB12	RB11 ⁽³⁾	RB10 ⁽³⁾	RB9	RB8	RB7	RB6 ⁽³⁾	RB5 ⁽³⁾	RB4 ⁽²⁾	RB3 ⁽³⁾	RB2	RB1 ⁽¹⁾	RB0 ⁽¹⁾	xxxx
LATB	02CC	LATB15	LATB14	LATB13	LATB12	LATB11 ⁽³⁾	LATB10 ⁽³⁾	LATB9	LATB8	LATB7	LATB6 ⁽³⁾	LATB5 ⁽³⁾	LATB4	LATB3 ⁽³⁾	LATB2	LATB1	LATB0	xxxx
ODCB	02CE	ODB15	ODB14	ODB13	ODB12	ODB11	ODB10	ODB9	ODB8	ODB7	ODB6	ODB5	ODB4	ODB3	ODB2	ODB1	ODB0	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

- 注
- 1: 当 PGEC1/PGED1 引脚上的调试功能有效时, 对 RB1 和 RB0 的读操作将导致读为 0。
 - 2: 当 PGEC3/PGED3 引脚上的调试功能有效时, 对 RB4 的读操作将导致读为 0。
 - 3: PORTB 的 bit 11、bit 10、bit 6、bit 5 和 bit 3 在 20 引脚器件上未实现。

表 3-14: 焊盘配置寄存器映射

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	所有复位时的状态
PADCFG1	02FC	—	—	—	—	—	—	—	—	—	—	—	SMBUSDEL	OC1TRIS	RTSECSSEL1	RTSECSSEL0	—	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	所有复位时的状态
ADC1BUF0	0300	ADC 数据缓冲区 0																xxxx
ADC1BUF1	0302	ADC 数据缓冲区 1																xxxx
ADC1BUF2	0304	ADC 数据缓冲区 2																xxxx
ADC1BUF3	0306	ADC 数据缓冲区 3																xxxx
ADC1BUF4	0308	ADC 数据缓冲区 4																xxxx
ADC1BUF5	030A	ADC 数据缓冲区 5																xxxx
ADC1BUF6	030C	ADC 数据缓冲区 6																xxxx
ADC1BUF7	030E	ADC 数据缓冲区 7																xxxx
ADC1BUF8	0310	ADC 数据缓冲区 8																xxxx
ADC1BUF9	0312	ADC 数据缓冲区 9																xxxx
ADC1BUFA	0314	ADC 数据缓冲区 10																xxxx
ADC1BUFB	0316	ADC 数据缓冲区 11																xxxx
ADC1BUFC	0318	ADC 数据缓冲区 12																xxxx
ADC1BUFD	031A	ADC 数据缓冲区 13																xxxx
ADC1BUFE	031C	ADC 数据缓冲区 14																xxxx
ADC1BUFF	031E	ADC 数据缓冲区 15																xxxx
AD1CON1	0320	ADON	—	ADSIDL	—	—	—	FORM1	FORM0	SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE	0000
AD1CON2	0322	VCFG2	VCFG1	VCFG0	OFFCAL	—	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	—	—	CH0SA4	CH0SA3	CH0SA2	CH0SA1	CH0SA0	0000
AD1PCFG	032C	—	—	—	PCFG12	PCFG11	PCFG10	—	—	—	—	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: — = 未实现, 读为 0。复位值以十六进制显示。

表 3-16: CTMU 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	所有复位时的状态
CTMUCON	033C	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG	EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT	0000
CTMUICON	033E	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	—	—	—	—	—	—	—	—	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

表 3-17: 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	所有复位时的状态
ALRMVAL	0620	基于 ALRMPTR<15:0> 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	基于 RTCPTR<15:0> 的 RTCC 值寄存器窗口																xxxx
RCFGCAL	0626	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

表 3-18: 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	所有复位时的状态
CMSTAT	0630	CMSIDL	—	—	—	—	—	C2EVT	C1EVT	—	—	—	—	—	—	C2OUT	C1OUT	0000
CVRCON	0632	—	—	—	—	—	—	—	—	CVREN	CVROE	CVRR	CVRSS	CVR3	CRV2	CRV1	CRV0	0000
CM1CON	0634	CON	COE	CPOL	CLPWR	—	—	CEVT	COUT	EVPOL1	EVPOL0	—	CREF	—	—	CCH1	CCH0	0000
CM2CON	0636	CON	COE	CPOL	CLPWR	—	—	CEVT	COUT	EVPOL1	EVPOL0	—	CREF	—	—	CCH1	CCH0	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

表 3-19: 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	所有复位时的状态
CRCCON	0640	—	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0	CRCFUL	CRCMPT	—	CRCGO	PLEN3	PLEN2	PLEN1	PLEN0	0040
CRCXOR	0642	X<15:1>																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 3-20: 时钟控制寄存器映射

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	所有复位时的状态
RCON	0740	TRAPR	IOPUWR	SBOREN	—	—	DPSLP	—	PMSLP	EXTR	SWR	SWDTEN	WDTO	SLEEP	IDLE	BOR	POR	(注 1)
OSCCON	0742	—	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0	CLKLOCK	—	LOCK	—	CF	—	SOSCEN	OSWEN	(注 2)
CLKDIV	0744	ROI	DOZE2	DOZE1	DOZE0	DOZEN	RCDIV2	RCDIV1	RCDIV0	—	—	—	—	—	—	—	—	3140
OSCTUN	0748	—	—	—	—	—	—	—	—	—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	0000
REFOCON	074E	ROEN	—	ROSSLP	ROSEL	RODIV3	RODIV2	RODIV1	RODIV0	—	—	—	—	—	—	—	—	0000
HLVDCON	0756	HLVDEN	—	HLSIDL	—	—	—	—	—	VDIR	BGVST	IRVST	—	HLVDL3	HLVDL2	HLVDL1	HLVDL0	0000

图注: — = 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配置熔丝和复位类型。

表 3-21: 深度休眠寄存器映射

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	所有复位时的状态 ⁽¹⁾
DSCON	0758	DSEN	—	—	—	—	—	—	—	—	—	—	—	—	—	DSBOR	RELEASE	0000
DSWSRC	075A	—	—	—	—	—	—	—	DSINT0	DSFLT	—	—	DSWDT	DSRTCC	DSMCLR	—	DSPOR	0000
DSGPR0	075C	深度休眠通用寄存器 0																0000
DSGPR1	075E	深度休眠通用寄存器 1																0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

- 注 1: 深度休眠寄存器仅在 VDD POR 事件时复位。

表 3-22: 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	所有复位时的状态
NVMCON	0760	WR	WREN	WRERR	PGMONLY	—	—	—	—	—	ERASE	NVMOP5	NVMOP4	NVMOP3	NVMOP2	NVMOP1	NVMOP0	0000 ⁽¹⁾
NVMKEY	0766	—	—	—	—	—	—	—	—	NVMKEY6	NVMKEY6	NVMKEY5	NVMKEY4	NVMKEY3	NVMKEY2	NVMKEY1	NVMKEY0	0000

图注: — = 未实现, 读为 0。复位值以十六进制显示。

- 注 1: 所示复位值仅适用于 POR。其他复位状态下的值取决于复位时存储器写操作或擦除操作的状态。

表 3-23: 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	所有复位时的状态
PMD1	0770	—	—	T3MD	T2MD	T1MD	—	—	—	I2C1MD	U2MD	U1MD	—	SPI1MD	—	—	ADC1MD	0000
PMD2	0772	—	—	—	—	—	—	—	IC1MD	—	—	—	—	—	—	—	OC1MD	0000
PMD3	0774	—	—	—	—	—	CMPMD	RTCCMD	—	CRCMD	—	—	—	—	—	—	—	0000
PMD4	0776	—	—	—	—	—	—	—	—	—	—	—	EEMD	REFOMD	CTMUMD	HLVDM	—	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

除了用作工作寄存器外，PIC24F 器件中的 W15 寄存器也可用作软件堆栈指针。指针总是指向第一个可用的空字，并且从低地址向高地址方向增长。它在弹出堆栈之前递减，而在压入堆栈之后递增，如图 3-4 所示。

对于任何 CALL 指令时的 PC 压栈，在压入堆栈之前，PC 的 MSB 要进行零扩展，从而确保 MSB 始终是清零的。

注： 在异常处理期间，在将 PC 压入堆栈之前，要先将 PC 的 MSB 与 SRL 寄存器组合在一起。

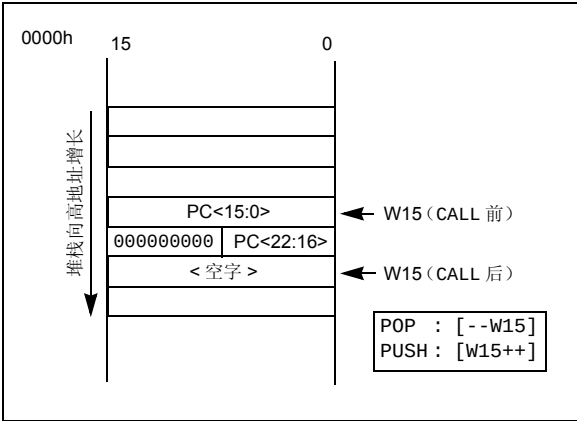
堆栈指针限制值 (SPLIM) 寄存器与堆栈指针相关联，它设置堆栈上边界的值。复位时 SPLIM 不被初始化。与堆栈指针的情况一样，SPLIM<0> 被强制为 0，因为所有的堆栈操作必须是字对齐的。每当使用 W15 作为源指针或目标指针产生 EA 时，有效地址会与 SPLIM 中的值进行比较。如果堆栈指针 (W15) 的内容与 SPLIM 寄存器的内容相等，则会执行压栈操作而不产生堆栈错误陷阱，但在随后的压栈操作时将会产生堆栈错误陷阱。

例如，当堆栈增长超过 RAM 中地址 0DF6 时，如果要想产生堆栈错误陷阱，则需要用值 0DF4 来初始化 SPLIM。

类似地，当堆栈指针地址小于 0800h 时，就会产生堆栈指针下溢 (堆栈错误) 陷阱。这可防止堆栈进入特殊功能寄存器 (SFR) 空间。

注： 在对 SPLIM 寄存器进行写操作之后，不应紧跟着使用 W15 进行间接读操作的指令。

图 3-4: CALL 堆栈帧



3.3 程序存储空间与数据存储空间的接口

PIC24F 架构采用 24 位宽的程序空间和 16 位宽的数据空间。该架构也是一种改进型哈佛结构，这意味着数据也能存放在程序空间内。要成功使用该数据，在访问数据时必须确保这两种存储空间中的信息是对齐的。

除了正常执行外，PIC24F 架构还提供了两种可在操作过程中访问程序空间的方法：

- 使用表指令访问程序空间中任意位置的各个字节或字
- 将程序空间的一部分重映射到数据空间 (PSV)

表指令允许应用程序读写程序存储器的一小块区域。这一功能对于访问需要随时更新的数据表来说非常理想。也可通过表操作访问一个程序字的所有字节。重映射方法允许应用程序访问一大块数据，但只限于读操作，它非常适合于在一个大的静态数据表中进行查找。这一方法只能访问程序字的低位字 (lsb)。

3.3.1 对程序空间进行寻址

由于数据和程序空间的地址范围分别为 16 位和 24 位，因此需要一个从 16 位数据寄存器创建一个 23 位或 24 位程序地址的方法。方法取决于所采用的接口方式。

对于表操作，使用 8 位的表存储器页地址寄存器 (TBLPAG) 定义程序空间内一个 32K 字的区域。这与 16 位 EA 组合形成了一个完整的 24 位程序空间地址。在这种地址形式下，TBLPAG 的最高位 (MSb) 用来决定操作是发生在用户存储区中 (TBLPAG<7> = 0) 还是配置存储区中 (TBLPAG<7> = 1)。

对于重映射操作，使用 8 位的程序空间可视性页地址寄存器 (PSVPAG) 定义程序空间中的 16K 字页。当 EA 的 MSb 为 1 时，PSVPAG 与 EA 的低 15 位组合形成一个 23 位的程序空间地址。与表操作不同，重映射操作被严格限制在用户存储区中。

请参见表 3-24 和图 3-5 了解如何通过表操作和重映射访问来从数据 EA 创建程序 EA。本文中，P<23:0> 指的是程序空间字；而 D<15:0> 指的是数据空间字。

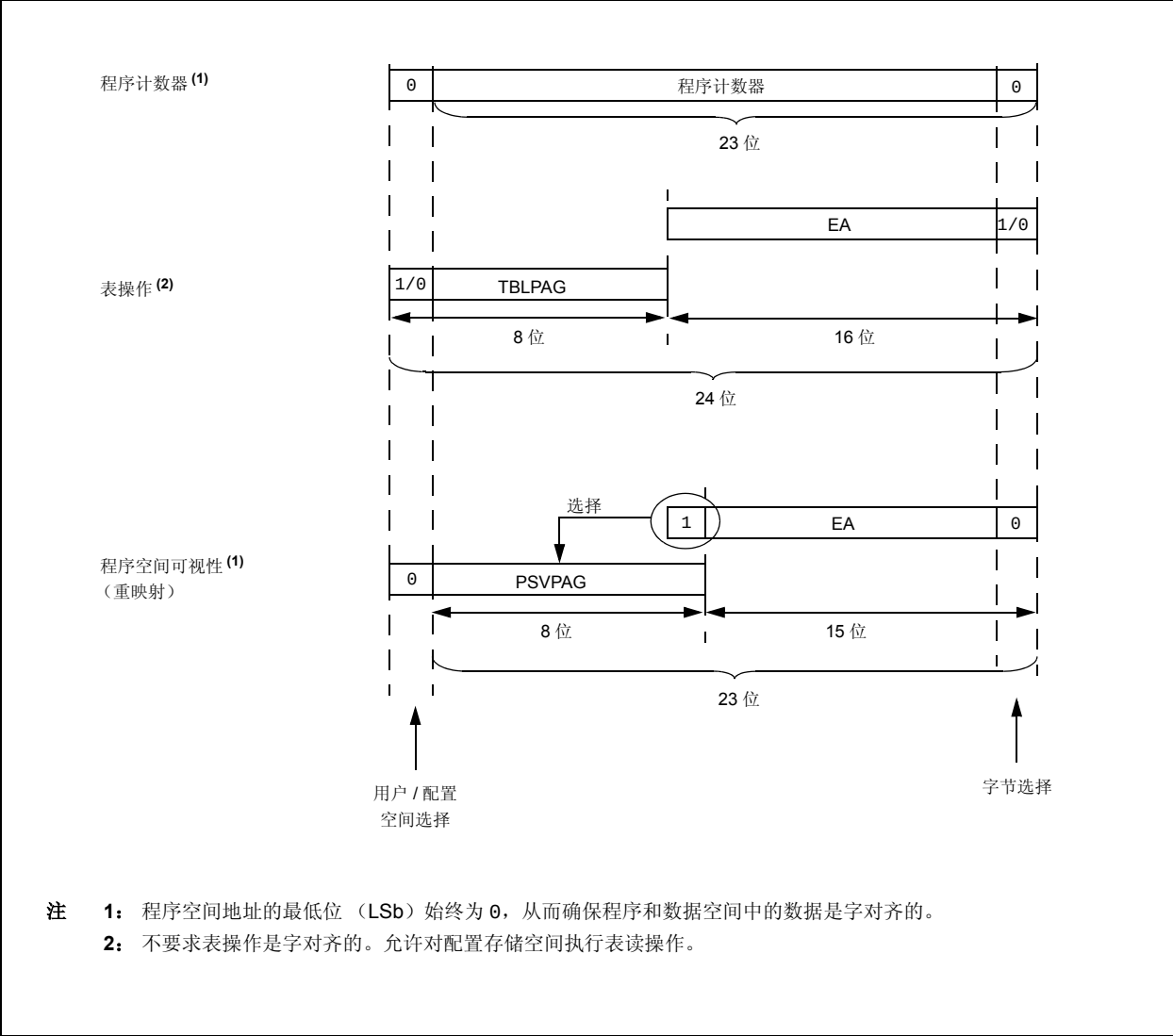
PIC24F16KA102 系列

表 3-24: 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 xxx0				
TBLRD/TBLWT (读 / 写字节或字)	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		
程序空间可视性 (块重映射 / 读)	用户	0	PSVPAG<7:0> ⁽²⁾		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, 但并不用它来计算程序空间地址。地址的 bit 15 为 PSVPAG<0>。
2: 在 PIC24F16KA102 系列中, PSVPAG 只有两个值 (00 用于访问程序存储器, FF 用于访问数据 EEPROM)。

图 3-5: 访问程序空间内数据的地址生成方式



注 1: 程序空间地址的最低位 (LSb) 始终为 0, 从而确保程序和数据空间中的数据是字对齐的。
2: 不要求表操作是字对齐的。允许对配置存储空间执行表读操作。

3.3.2 使用表指令访问程序存储器和数据
EEPROM 存储器中的数据

TBLRDH 和 TBLWTL 指令提供了读或写程序存储器内任何地址的低位字的直接方法，无需通过数据空间。它还提供了读或写数据 EEPROM 存储器内任何地址的字的直接方法。TBLRDH 和 TBLWTH 指令是可以把一个程序空间字的高 8 位作为数据读写的唯一方法。

注： 当访问数据 EEPROM 存储器时，不使用 TBLRDH 和 TBLWTH 指令。

对于每个连续的 24 位程序字，PC 的递增量为 2。这使得程序存储器地址能够直接映射到数据空间地址。于是，程序存储器可以看作是两个 16 位字宽的地址空间，它们并排放置，具有相同的地址范围。TBLRDH 和 TBLWTL 访问存有最低有效数据字的空间，而 TBLRDH 和 TBLWTH 则访问存有高数据字节的空间。

提供了两条表指令来对程序空间执行字节或字（16 位）大小的数据读写。读和写都可以采用字节或字操作的形式。

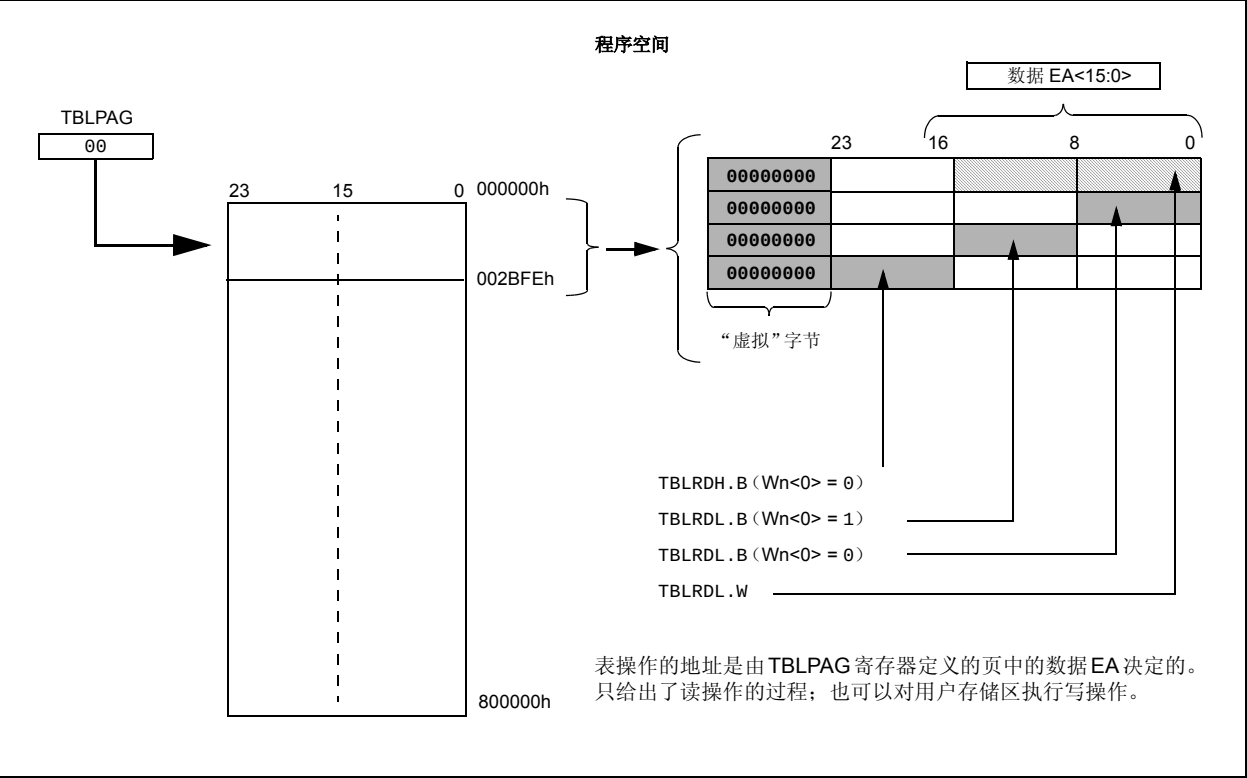
- 1. TBLRDH（表读低位字）：在字模式下，该指令将程序空间地址的低位字（P<15:0>）映射到数据地址（D<15:0>）中。
在字节模式下，低位程序字的高字节或低字节被映射到数据地址的低字节中。当字节选择位为 1 时映射高字节；当字节选择位为 0 时映射低字节。
- 2. TBLRDH（表读高位字）：在字模式下，该指令将程序地址的整个高位字（P<23:16>）映射到数据地址中。注意，D<15:8> 为“虚拟”字节，它始终为 0。
在字节模式下，该指令将程序字的高字节或低字节映射到数据地址的 D<7:0> 中，就如同 TBLRDH 指令。注意，当选择高位“虚拟”字节（字节选择位 = 1）时，数据将始终为 0。

表指令 TBLWTH 和 TBLWTL 以类似的方式向程序空间地址写入各字节或字。第 4.0 节“闪存程序存储器”对这两条指令的详细操作给出了说明。

对于所有的表操作，要访问程序存储空间的哪个区域是由表存储器页地址寄存器（TBLPAG）决定的。TBLPAG 可寻址器件的整个程序存储空间，包括用户和配置空间。当 TBLPAG<7> = 0 时，表页位于用户存储空间中。当 TBLPAG<7> = 1 时，表页位于配置存储空间中。

注： 只有表读操作将在配置存储空间中执行，且只能在已实现区域中执行，例如器件 ID。不允许表写操作。

图 3-6: 使用表指令访问程序存储器



3.3.3 使用程序空间可视性读程序存储器中的数据

可选择将数据空间的高 32 KB 映射到程序空间的 8K 字节中（在 PIC24F08KA1XX 器件中）和 16K 字节中（在 PIC24F16KA1XX 器件中）。这提供了通过数据空间对存储的常量数据的透明访问，而无需使用特殊指令（即 TBLRDL/H）。

如果数据空间 EA 的 MSb 为 1，并且 PSV 使能（方法是将 CPU 控制寄存器中的 PSV 位（CORCON<2>）置 1）时，就能通过数据空间访问程序空间。由程序空间可视性页地址寄存器（PSVPAG）决定要被映射到数据空间中的程序存储空间的位置。这一 8 位的寄存器定义程序空间中 256 个可能的 16K 字节中的一个。事实上，PSVPAG 作为程序存储地址的高 8 位，而 EA 的 15 位则作为地址的低位。

对于每个程序存储字，PC 都将递增 2，数据空间地址的低 15 位将直接映射到相应程序空间地址的低 15 位。

从该区域读取数据的指令，需要增加一个额外的指令周期，因为这类指令需要对程序存储器进行两次数据读取操作。

尽管大于或等于 8000h 的每个数据空间地址直接映射到对应的程序存储器地址（见图 3-7），但只使用 24 位程序字的低 16 位来存放数据。所有用来存放数据的程序存储单元的高 8 位都应被设置为 1111 1111 或 0000 0000，强制为一条 NOP 指令，从而避免了可能出现意外执行这一区域内代码的情况。

注： 在表读 / 写期间，暂时禁止 PSV 访问。

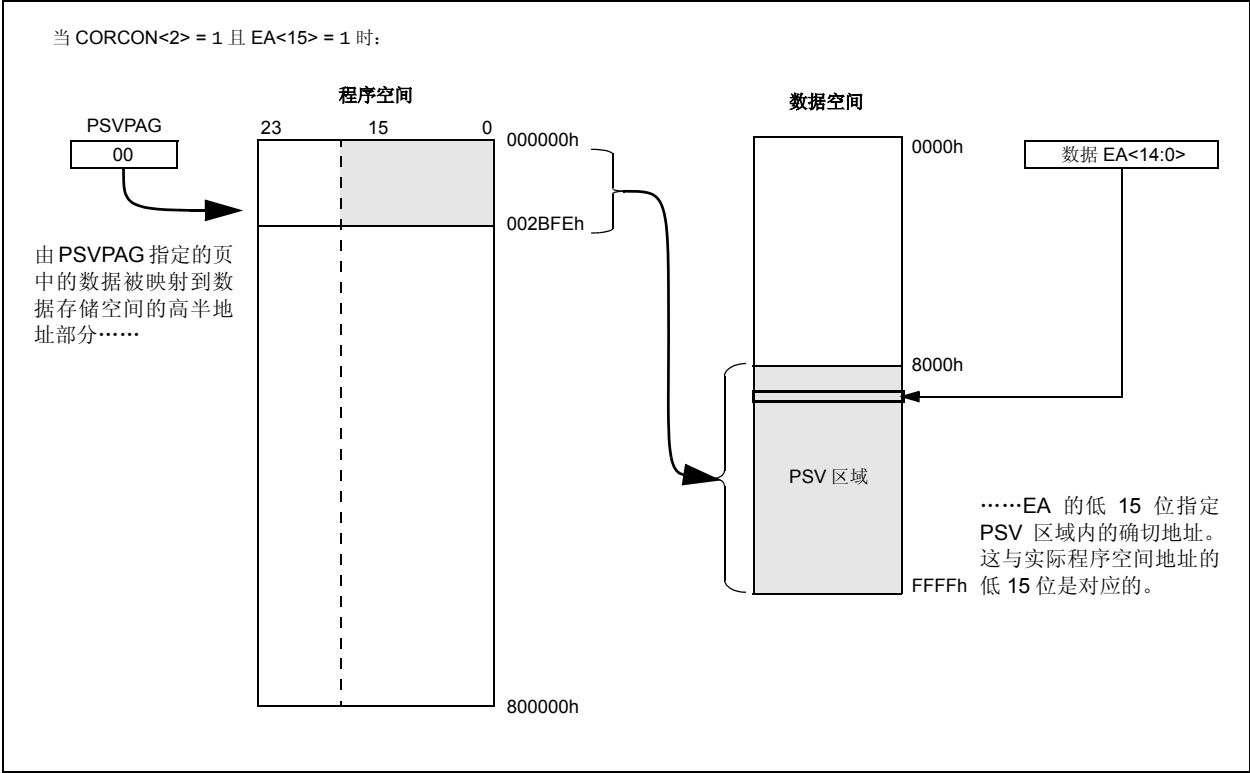
对于使用 PSV 而又在 REPEAT 循环外执行的操作，MOV 和 MOV.D 指令除了规定的执行时间之外，还需要一个额外的指令周期。其他所有指令都需要在规定的执行时间之外额外增加两个指令周期。

对于使用 PSV 而又在 REPEAT 循环内执行的操作，下列情况，除了规定的指令执行时间之外，还需要两个额外的指令周期：

- 在第一次迭代中执行的指令
- 在最后一次迭代中执行的指令
- 由于中断而退出循环之前执行的指令
- 中断得到处理后再次进入循环时执行的指令

REPEAT 循环的所有其他各次迭代，都允许使用 PSV 访问数据的指令在一个周期内执行。

图 3-7：程序空间可视性操作



PIC24F16KA102 系列

注:

4.0 闪存程序存储器

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于闪存编程的更多信息，请参见《PIC24F 系列参考手册》的“第4章 程序存储器”（DS39715A_CN）。

PIC24F16KA102 系列器件包含用于存储和执行应用代码的内部闪存程序存储器。当工作在 VDD 大于 1.8V 时，该存储器都是可读写和可擦除的。

可采用三种方式对闪存进行编程：

- 在线串行编程（ICSP™）
- 运行时自编程（Run-Time Self-Programming, RTSP）
- 增强型在线串行编程（增强型 ICSP）

ICSP 允许在最终的应用电路中对 PIC24F16KA102 器件进行串行编程。只需要使用 5 根线就可以完成编程，它们分别是编程时钟线、编程数据线（分别命名为 PGCx 和 PGDx）、电源线（VDD）、接地线（Vss）和主复位 / 编程模式入口电压线（MCLR/Vpp）。这允许用户在生产电路板时使用未编程器件，而仅在产品交付之前才对单片机进行编程，从而可以使用最新版本的固件或者定制固件进行编程。

使用 TBLRD（表读）和 TBLWT（表写）指令来实现实时流协议（Real-Time Streaming Protocol, RTSP）。使用 RTSP，用户可以一次写入一块 32 条指令（96 字节）的程序存储器数据，一次擦除一块 32、64 和 128 条指令（96、192 和 384 字节）的程序存储区域。

NVMOP<1:0>（NVMCON<1:0>）位用于决定擦除块的大小。

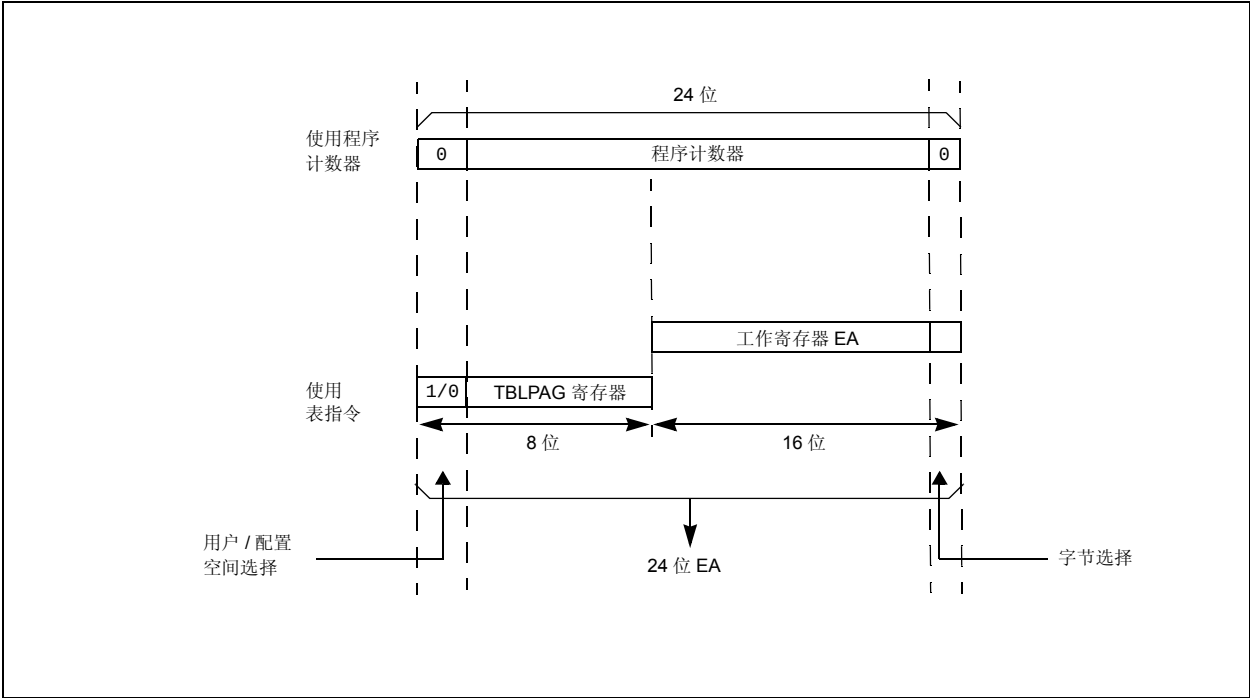
4.1 表指令和闪存编程

闪存的编程都是用表读和表写指令实现的，与使用的编程方法无关。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。程序存储器中 24 位目标地址由 TBLPAG<7:0> 位和表指令中指定 W 寄存器中的有效地址（EA）组成，如图 4-1 所示。

TBLRDL 和 TBLWTL 指令用来读或写程序存储器的 bit<15:0>。TBLRDL 和 TBLWTL 能以字或字节模式访问程序存储器。

TBLRDH 和 TBLWTH 指令用来读或写程序存储器的 bit<23:16>。TBLRDH 和 TBLWTH 同样能以字或字节模式访问程序存储器。

图 4-1： 表寄存器的寻址



4.2 RTSP 工作原理

PIC24F 闪存程序存储器阵列是由 32 条指令或 96 字节的行组成的。RTSP 允许用户一次擦除 1 行、2 行和 4 行（32、64 和 128 条指令）的块，一次编程一行。它也可以编程单个字。

1 行（96 字节）、2 行（192 字节）和 4 行（384 字节）擦除块以及单行写块（96 字节）都是边沿对齐的，从程序存储器起始地址开始。

当使用 TBLWT 指令将数据写入程序存储器时，数据并不直接被写入存储器。相反，使用表写指令写入的数据被存储在保持锁存器中，直到执行编程序列。

可以执行任意数量的 TBLWT 指令，随后将成功执行一次写操作。但是，需要 32 条 TBLWT 指令写入存储器的整行。

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 指令来装载缓冲区。通过设置 NVMCON 寄存器中的控制位来执行编程。

可以任何顺序装入数据，而且在执行写操作前可以多次写入保持寄存器。但是，后续的写操作将擦除任何先前的写操作写入的内容。

注： 不建议对同一个存储单元进行多次写操作而不进行擦除操作。

由于只写缓冲区，所以所有的表写操作都是单字写操作（2 个指令周期）。编程每一行需要一个编程周期。

4.3 增强型在线串行编程

增强型 ICSP 使用板上自举程序（称为编程执行程序）来管理编程过程。使用 SPI 数据帧格式，编程执行程序能够擦除、编程和校验程序存储器。关于增强型 ICSP 的更多信息，请参见器件编程规范。

4.4 控制寄存器

有两个 SFR 用于读写闪存程序存储器：NVMCON 和 NVMKEY。

NVMCON 寄存器（寄存器 4-1）控制要擦除的块、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。要启动编程或擦除序列，用户必须按顺序将 55h 和 AAh 写入 NVMKEY 寄存器。更多详细信息，请参见第 4.5 节“编程操作”。

4.5 编程操作

在 RTSP 模式下，对内部闪存进行编程或擦除需要执行完整的编程序列。编程或擦除操作期间，处理器暂停（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。

寄存器 4-1: NVMCON: 闪存控制寄存器

R/SO-0, HC	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	PGMONLY ⁽⁴⁾	—	—	—	—
bit 15				bit 8			

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ERASE	NVMOP5 ⁽¹⁾	NVMOP4 ⁽¹⁾	NVMOP3 ⁽¹⁾	NVMOP2 ⁽¹⁾	NVMOP1 ⁽¹⁾	NVMOP0 ⁽¹⁾
bit 7				bit 0			

图注:	SO = 只可置 1 位	HC = 硬件清零位	
-n = POR 时的值	1 = 置 1	R = 可读位	W = 可写位
0 = 清零	x = 未知	U = 未实现位, 读为 0	

- bit 15 **WR:** 写控制位
1 = 启动闪存编程或擦除操作。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 = 试图执行不合法的编程或擦除序列, 或者发生终止 (试图将 WR 位置 1 时自动置 1 该位)
0 = 编程或擦除操作正常完成
- bit 12 **PGMONLY:** 仅编程使能位 ⁽⁴⁾
- bit 11-7 未实现: 读为 0
- bit 6 **ERASE:** 擦除 / 编程使能位
1 = 在下一条 WR 命令时执行 NVMOP<5:0> 指定的擦除操作
0 = 在下一条 WR 命令时执行 NVMOP<5:0> 指定的编程操作
- bit 5-0 **NVMOP<5:0>:** 编程操作命令字节位 ⁽¹⁾
擦除操作 (当 ERASE 位为 1 时):
1010xx = 擦除整个引导块 (包括被代码保护的引导块) ⁽²⁾
1001xx = 擦除整个存储器 (包括引导块、配置块和通用块) ⁽²⁾
011010 = 擦除闪存的 4 行 ⁽³⁾
011001 = 擦除闪存的 2 行 ⁽³⁾
011000 = 擦除闪存的 1 行 ⁽³⁾
0101xx = 擦除整个配置块 (代码保护位除外)
0100xx = 擦除整个数据 EEPROM ⁽⁴⁾
0011xx = 擦除整个通用存储器块
编程操作 (当 ERASE 位为 0 时):
0001xx = 写数据 EEPROM 的 1 个字或闪存的 1 行 ⁽³⁾

- 注 1: NVMOP<5:0> 的所有其他组合均无操作。
2: 仅在 ICSP™ 模式下可用。请参见器件编程规范。
3: 表指针中的地址决定将要擦除的行。
4: 该位仅在访问数据 EEPROM 时使用。

4.5.1 闪存程序存储器的编程算法

通过擦除可编程行，用户可以一次对闪存程序存储器的一行进行编程。一般过程如下：

1. 读取程序存储器的一行（32 条指令），并存储在数据 RAM 中。
2. 用所需的新数据更新 RAM 中的程序数据。
3. 擦除一行（见例 4-1）：
 - a) 将 NVMOP 位（NVMCON<5:0>）设置为 011000，配置为行擦除操作。将 ERASE（NVMCON<6>）和 WREN（NVMCON<14>）位置 1。
 - b) 将要被擦除的块的起始地址写入 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。擦除周期开始，在擦除周期中 CPU 会暂停。当擦除完成时，WR 位会被自动清零。

4. 将数据 RAM 中的前 32 条指令写入程序存储器缓冲区（见例 4-1）。
5. 将程序块写入闪存：
 - a) 将 NVMOP 位设置为 011000，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。编程周期开始，在写周期中 CPU 会暂停。当闪存写操作完成时，WR 位会被自动清零。

为防止意外操作，必须向 NVMKEY 写入启动序列，用于允许执行擦除或编程操作。在执行编程命令后，用户必须等待一段编程时间，直至编程完成。紧跟编程启动序列后面的两条指令应为 NOP，如例 4-5 所示。

例 4-1: 擦除程序存储器的一行——汇编语言代码

```
; Set up NVMCON for row erase operation
MOV    #0x4058, 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
                                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: 擦除程序存储器的一行——C 语言代码

```
// C example using MPLAB C30

int __attribute__((space(auto_psv))) progAddr = &progAddr; // Variable located in Pgm Memory
unsigned int offset;

//Set up pointer to the first memory location to be written

TBLPAG = __builtin_tblpage(&progAddr);           // Initialize PM Page Boundary SFR
offset = &progAddr & 0xFFFF;                     // Initialize lower word of address

__builtin_tblwtl(offset, 0x0000);                 // Set base address of erase block
                                                // with dummy latch write

NVMCON = 0x4058;                                  // Initialize NVMCON

asm("DISI #5");                                   // Block all interrupts for next 5 instructions
__builtin_write_NVM();                             // C30 function to perform unlock
                                                // sequence and set WR
```

PIC24F16KA102 系列

例 4-3: 装载写缓冲区——汇编语言代码

```
; Set up NVMCON for row programming operations
MOV    #0x4004, 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
.
.
.
; 32nd_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 4-4: 装载写缓冲区——C 语言代码

```
// C example using MPLAB C30

#define NUM_INSTRUCTION_PER_ROW 64
int __attribute__((space(auto_psv))) progAddr = &progAddr; // Variable located in Pgm Memory
unsigned int offset;
unsigned int i;
unsigned int progData[2*NUM_INSTRUCTION_PER_ROW]; // Buffer of data to write

//Set up NVMCON for row programming
NVMCON = 0x4001; // Initialize NVMCON

//Set up pointer to the first memory location to be written
TBLPAG = __builtin_tblpage(&progAddr); // Initialize PM Page Boundary SFR
offset = &progAddr & 0xFFFF; // Initialize lower word of address

//Perform TBLWT instructions to write necessary number of latches
for(i=0; i < 2*NUM_INSTRUCTION_PER_ROW; i++)
{
    __builtin_tblwtl(offset, progData[i++]); // Write to address low word
    __builtin_tblwth(offset, progData[i]); // Write to upper byte
    offset = offset + 2; // Increment address
}
```

例 4-5: 启动编程序列——汇编语言代码

```

DISI    #5                                ; Block all interrupts
                                           ; 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                                           ; 2 NOPs required after setting WR
NOP                                           ;
BTSC    NVMCON, #15                      ; Wait for the sequence to be completed
BRA     $-2                               ;

```

例 4-6: 启动编程序列——C 语言代码

```

// C example using MPLAB C30

asm("DISI #5");                          // Block all interrupts for next 5 instructions

__builtin_write_NVM();                   // Perform unlock sequence and set WR

```

例 4-7: 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

```

PIC24F16KA102 系列

注:

5.0 数据 EEPROM 存储器

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于数据 EEPROM 的更多信息，请参见《PIC24F 系列参考手册》的“第5章数据EEPROM”（DS39720A_CN）。

数据 EEPROM 存储器是非易失性存储器（Nonvolatile Memory, NVM），不同于程序存储器和易失性数据 RAM。数据 EEPROM 存储器基于与程序存储器相同的闪存技术，并经过优化，可实现长期数据保存和更高的耐擦写次数。

数据 EEPROM 被映射到用户程序存储空间的顶部，最高地址位于程序存储器地址 7FFE00h 到 7FFFFFFh 处。在 PIC24F16KA102 器件中，数据 EEPROM 的大小是 256 字。

数据 EEPROM 由 16 位宽的存储器构成。在整个 VDD 范围内的正常工作期间，每个字都可直接寻址，并且是可读可写的。

不同于闪存程序存储器，在数据 EEPROM 编程或擦除操作期间，不会停止正常的程序执行。

数据 EEPROM 的编程操作使用 3 个 NVM 控制寄存器进行控制：

- NVMCON：非易失性存储器控制寄存器
- NVMKEY：非易失性存储器密钥寄存器
- NVMADR：非易失性存储器地址寄存器

例 5-1： 数据 EEPROM 解锁序列

```
//Disable Interrupts For 5 instructions
asm volatile("disi #5");
//Issue Unlock Sequence
asm volatile("mov #0x55, W0      \n"
             "mov W0, NVMKEY      \n"
             "mov #0xAA, W1      \n"
             "mov W1, NVMKEY      \n");
// Perform Write/Erase operations
asm volatile ("bset NVMCON, #WR   \n"
             "nop                  \n"
             "nop                  \n");
```

5.1 NVMCON 寄存器

NVMCON 寄存器（寄存器 5-1）也是数据 EEPROM 编程 / 擦除操作的主控制寄存器。高字节包含用于启动编程或擦除周期的控制位，以及用于指示操作是否成功执行的标志位。NVMCOM 的低字节用于配置将执行的 NVM 操作的类型。

5.2 NVMKEY 寄存器

NVMKEY 是一个只写寄存器，用于防止数据 EEPROM 存储单元的误写或误擦除。

要启动任何编程或擦除序列，必须严格按顺序先执行以下指令：

1. 将 55h 写入 NVMKEY。
2. 将 AAh 写入 NVMKEY。

在此序列后，就可以在一个指令周期中写入 NVMCON 寄存器。在大多数情况下，用户只需要将 NVMCON 寄存器中的 WR 位置 1 就可以启动编程或擦除周期。在解锁序列中应禁止中断。

MPLAB® C30 C 编译器提供了一个已定义的库过程（builtin_write_nvm）用来执行解锁序列。例 5-1 显示了如何使用行内汇编来执行解锁序列。

PIC24F16KA102 系列

寄存器 5-1: NVMCON: 非易失性存储器控制寄存器

R/S-0, HC	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	PGMONLY	—	—	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ERASE	NVMOP5	NVMOP4	NVMOP3	NVMOP2	NVMOP1	NVMOP0
bit 7							bit 0

图注:	U = 未实现位, 读为 0	HC = 硬件清零位
R = 可读位	W = 可写位	S = 可置 1 位
-n = POR 时的值	1 = 置 1	0 = 清零
		x = 未知

bit 15	WR: 写控制位 (编程或擦除) 1 = 启动数据 EEPROM 的擦除或写周期 (用软件只能将该位置 1, 但不能清零) 0 = 写周期完成 (由硬件自动清零)
bit 14	WREN: 写使能位 (擦除或编程) 1 = 使能擦除或编程操作 0 = 不允许任何操作 (在写 / 擦除操作完成时器件会将该位清零)
bit 13	WRERR: 闪存错误标志位 1 = 写操作过早终止 (编程操作期间的任何 MCLR 复位或 WDT 复位) 0 = 写操作成功完成
bit 12	PGMONLY: 仅编程使能位 1 = 不先擦除目标地址内容即执行写操作 0 = 自动写前擦除: 在执行写操作之前, 自动对目标地址进行擦除操作
bit 11-7	未实现: 读为 0
bit 6	ERASE: 擦除操作选择位 1 = WR 置 1 时执行擦除操作 0 = WR 置 1 时执行写操作
bit 5-0	NVMOP<5:0>: 编程操作命令字节位 <u>擦除操作 (当 ERASE 位为 1 时):</u> 011010 = 擦除 8 个字 011001 = 擦除 4 个字 011000 = 擦除 1 个字 0100xx = 擦除整个数据 EEPROM <u>编程操作 (当 ERASE 位为 0 时):</u> 001xx = 写 1 个字

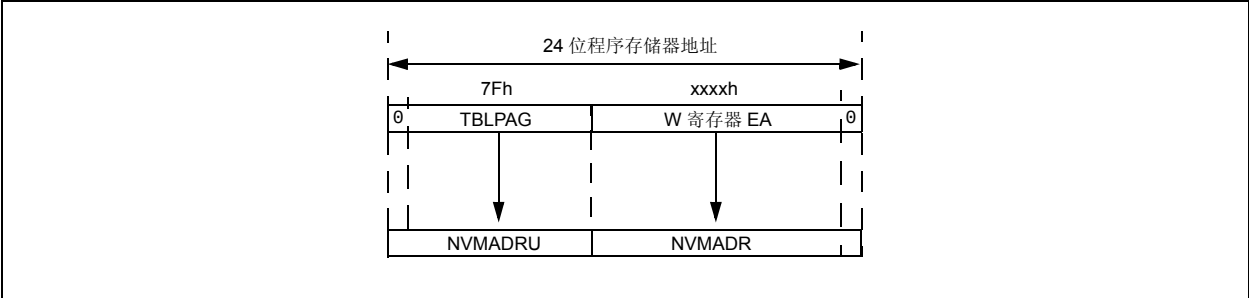
5.3 NVM 地址寄存器

类似于闪存程序存储器，NVM 地址寄存器 NVMADRU 和 NVMADR 构成数据 EEPROM 中要进行操作的选定行或字的 24 位有效地址（EA）。NVMADRU 寄存器用于保存 EA 的高 8 位，而 NVMADR 寄存器用于保存 EA 的低 16 位。这两个寄存器不会被映射到特殊功能寄存器（SFR）空间中；相反，它们直接捕捉已执行的上一条表写指令的 EA<23:0>，并选择要擦除的数据 EEPROM 行。图 5-1 显示了用于编程和擦除操作的程序存储器 EA 是如何构成的。

类似于程序存储器操作，NVMADR 的最低有效位（Least Significant bit, LSb）被限制到偶地址。这是因为数据 EEPROM 空间中的任何给定地址仅包含程序存储器宽度的低位字；高位字（包括最高位的“虚拟字节”）是不可用的。这意味着数据 EEPROM 地址的 LSb 将始终为 0。

类似地，NVMADRU 的最高有效位（MSb）始终为 0，因为所有地址都位于用户程序空间中。

图 5-1: 使用 TBLPAG 和 NVM 地址寄存器对数据 EEPROM 进行寻址



5.4 数据 EEPROM 操作

EEPROM 块使用类似于用于程序存储器的表读和表写操作进行访问。对于 EEPROM 操作，不需要 TBLWTH 和 TBLRDH 指令，因为存储器只有 16 位宽（只有低地址上的数据有效）。对于数据 EEPROM，可以执行以下编程操作：

- 擦除 1、4 或 8 个字
- 批量擦除整个数据 EEPROM
- 写 1 个字
- 读 1 个字

注 1: 如果在编程或擦除操作正在进行时，用户尝试读 EEPROM，则会产生意外的结果。

2: C30 C 编译器包含了一些库过程，用于自动执行表读和表写操作，管理表指针和写缓冲区，以及解锁和启动存储器写序列。这使用户不必用 C 语言为每个应用程序创建汇编器宏或对时间要求严格的程序。

在以下几节详细说明了的代码示例中使用了这些库过程。对于未使用 C30 编译器库的用户，提供了每个过程的大致说明。

5.4.1 擦除数据 EEPROM

可以完全擦除数据 EEPROM，或者部分擦除数据 EEPROM（有三种不同的大小：1 个字、4 个字或 8 个字）。NVMOP<1:0> 位（NVMCON<1:0>）用于决定要擦除的字的数目。若要部分擦除数据 EEPROM，必须遵循以下序列：

1. 配置 NVMCON 以擦除所需的字数：1、4 或 8。
2. 将要擦除的 EEPROM 的地址装入 TBLPAG 和 WREG。
3. 清零 NVMIF 状态位并允许 NVM 中断（可选）。
4. 向 NVMKEY 中写入密钥序列。
5. 将 WR 位置 1 以开始擦除周期。
6. 查询 WR 位，或者等待 NVM 中断（NVMIF 置 1）。

例 5-2 给出了典型的擦除序列。该示例显示了如何执行单字擦除。类似地，也可以执行 4 字擦除和 8 字擦除。该示例使用了一些 C 库过程来管理表指针（`builtin_tblpage` 和 `builtin_tbloffset`）和擦除页指针（`builtin_tblwtl`）。存储器解锁序列（`builtin_write_NVM`）还会将 WR 位置 1 以启动操作，并在完成时返回控制。

例 5-2: 单字擦除

```
int __attribute__((space(eedata))) eeData = 0x1234; // Variable located in EEPROM
unsigned int offset;

// Set up NVMCON to erase one word of data EEPROM
NVMCON = 0x4058;

// Set up a pointer to the EEPROM location to be erased
TBLPAG = __builtin_tblpage(&eeData); // Initialize EE Data page pointer
offset = __builtin_tbloffset(&eeData); // Initialize lower word of address
__builtin_tblwtl(offset, 0); // Write EEPROM data to write latch

asm volatile ("disi #5"); // Disable Interrupts For 5 Instructions
__builtin_write_NVM(); // Issue Unlock Sequence & Start Write Cycle
```

5.4.1.1 数据 EEPROM 批量擦除

若要擦除整个数据 EEPROM（批量擦除），不需要配置地址寄存器，因为该操作会影响整个数据 EEPROM。以下序列可帮助执行批量擦除：

1. 配置 NVMCON 为批量擦除模式。
2. 清零 NVMIF 状态位并允许 NVM 中断（可选）。
3. 向 NVMKEY 中写入密钥序列。
4. 将 WR 位置 1 以开始擦除周期。
5. 查询 WR 位，或者等待 NVM 中断（NVMIF 置 1）。

例 5-3 给出了典型的批量擦除序列。

5.4.2 单字写

若要写数据 EEPROM 的单个字，必须遵循以下序列：

1. 如果 PGMONLY 位（NVMCON<12>）被设为 1，擦除数据 EEPROM 的一个字（如前面的小节中所述）。
2. 将数据字写入数据 EEPROM 锁存器。
3. 将数据字编程到 EEPROM 中：
 - 配置 NVMCON 寄存器以编程 EEPROM 的一个字（NVMCON<5:0> = 0001xx）。
 - 清零 NVMIF 状态位并允许 NVM 中断（可选）。
 - 向 NVMKEY 中写入密钥序列。
 - 将 WR 位置 1 以开始擦除周期。
 - 查询 WR 位，或者等待 NVM 中断（NVMIF 置 1）。
 - 若要清零，等待直到 NVMIF 置 1。

例 5-4 给出了典型的单字写序列。

例 5-3: 数据 EEPROM 批量擦除

```
// Set up NVMCON to bulk erase the data EEPROM
NVMCON = 0x4050;

// Disable Interrupts For 5 Instructions
asm volatile ("disi #5");

// Issue Unlock Sequence and Start Erase Cycle
__builtin_write_NVM();
```

例 5-4: 将单个字写入数据 EEPROM

```
int __attribute__((space(eedata))) eeData = 0x1234; // Variable located in EEPROM
int newData; // New data to write to EEPROM
unsigned int offset;

// Set up NVMCON to erase one word of data EEPROM
NVMCON = 0x4004;

// Set up a pointer to the EEPROM location to be erased
TBLPAG = __builtin_tblpage(&eeData); // Initialize EE Data page pointer
offset = __builtin_tbloffset(&eeData); // Initilize lower word of address
__builtin_tblwtl(offset, newData); // Write EEPROM data to write latch

asm volatile ("disi #5"); // Disable Interrupts For 5 Instructions
__builtin_write_NVM(); // Issue Unlock Sequence & Start Write Cycle
```

PIC24F16KA102 系列

5.4.3 读数据 EEPROM

若要从数据EEPROM读取一个字，需要使用表读指令。因为EEPROM阵列只有16位宽，所以只需要TBLRD指令。读操作通过将EEPROM单元的地址装入TBLPAG和WREG，后跟一条TBLRD指令执行。

典型的读序列（如例5-5所示）使用了来自C30编译器库的表指针管理（`builtin_tblpage`和`builtin_tbloffset`）和表读（`builtin_tblrdl`）过程。

程序空间可视性（PSV）也可用于读取数据EEPROM中的单元。

例 5-5: 使用 TBLRD 命令读取数据 EEPROM

```
int __attribute__((space(eedata))) eeData = 0x1234;    // Variable located in EEPROM
int data;                                              // Data read from EEPROM
unsigned int offset;

// Set up a pointer to the EEPROM location to be erased
TBLPAG = __builtin_tblpage(&eeData);                // Initialize EE Data page pointer
offset = __builtin_tbloffset(&eeData);                // Initizlize lower word of address
data = __builtin_tblrdl(offset);                      // Write EEPROM data to write latch
```

6.0 复位

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于复位的更多信息，请参见《PIC24F 系列参考手册》的“Section 40. Reset with Programmable Brown-out Reset”（DS39728A）。

复位模块结合了所有复位源并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
- WDTR：看门狗定时器复位
- BOR：欠压复位
- 低功耗 BOR/ 深度休眠 BOR
- TRAPR：陷阱冲突复位
- IOPUWR：非法操作码复位
- UWR：未初始化的 W 寄存器复位

图 6-1 给出了复位模块的简化框图。

任何有效的复位源都将使 **SYSRST** 信号有效。很多与 CPU 和外设相关的寄存器均会被强制为已知的复位状态。大多数寄存器都不受复位影响；它们的状态在上电复位（POR）时未知，而在所有其他复位时不变。

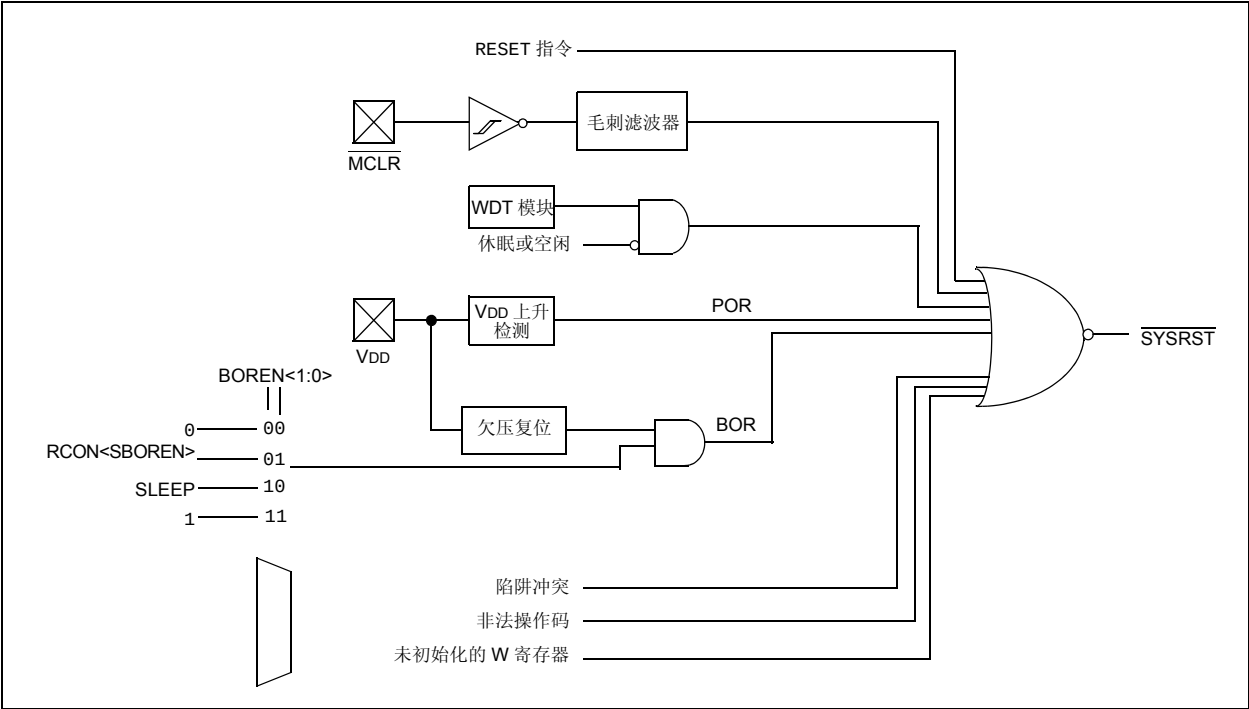
注： 如需了解寄存器复位状态的信息，请参见本手册中特定的外设或 CPU 章节。

任何类型的器件复位都会将 RCON 寄存器中相应的状态位置 1，以表明复位类型（见寄存器 6-1）。POR 将清零 BOR 和 POR 位（RCON<1:0>）之外的所有位，BOR 和 POR 位在 POR 时被置 1。用户可以在代码执行过程中的任何时间置 1 或清零任意位。RCON 寄存器中的位仅用作状态位。用软件将特定的复位状态位置 1 不会导致器件发生复位。

RCON 寄存器还包含与看门狗定时器（WDT）和器件节能状态相关的其他位。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，这样在器件复位后的 RCON 寄存器的下一个值才有意义。

图 6-1： 复位系统框图



PIC24F16KA102 系列

寄存器 6-1: RCON: 复位控制寄存器⁽¹⁾

R/W-0, HS	R/W-0, HS	R/W-0	U-0	U-0	R/C-0, HS	U-0	R/W-0
TRAPR	IOPUWR	SBOREN	—	—	DPSLP	—	PMSLP
bit 15							bit 8

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

图注:	C = 可清零位	HS = 硬件置 1 位
R = 可读位	W = 可写位	U = 未实现位, 读为 0
-n = POR 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的 W 寄存器复位标志位 1 = 检测到非法操作码、非法地址模式或将未初始化的 W 寄存器用作地址指针而导致复位 0 = 未发生非法操作码或未初始化的 W 寄存器复位
bit 13	SBOREN: 软件使能 / 禁止 BOR 位 1 = 用软件开启 BOR 0 = 用软件关闭 BOR
bit 12-11	未实现: 读为 0
bit 10	DPSLP: 深度休眠模式标志位 1 = 发生了深度休眠 0 = 未发生深度休眠
bit 9	未实现: 读为 0
bit 8	PMSLP: 程序存储器休眠期间电源位 1 = 程序存储器偏置电压在休眠期间保持供电 0 = 程序存储器偏置电压在休眠期间掉电
bit 7	EXTR: 外部复位 (MCLR) 引脚位 1 = 发生了主复位 (引脚) 复位 0 = 未发生主复位 (引脚) 复位
bit 6	SWR: 软件复位 (指令) 标志位 1 = 执行了 RESET 指令 0 = 未执行 RESET 指令
bit 5	SWDTEN: 软件使能 / 禁止 WDT 位 ⁽²⁾ 1 = 使能 WDT 0 = 禁止 WDT
bit 4	WDTO: 看门狗定时器超时标志位 1 = 发生了 WDT 超时 0 = 未发生 WDT 超时
bit 3	SLEEP: 从休眠模式唤醒标志位 1 = 器件处于休眠模式 0 = 器件不处于休眠模式
bit 2	IDLE: 从空闲模式唤醒标志位 1 = 器件处于空闲模式 0 = 器件不处于空闲模式
bit 1	BOR: 欠压复位标志位 1 = 发生了欠压复位 (BOR 在 POR 后也将置 1) 0 = 未发生欠压复位

寄存器 6-1: RCON: 复位控制寄存器⁽¹⁾ (续)

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。用软件将这些位中的某一位置 1 不会导致器件复位。
2: 如果 FWDTEN 配置位为 1 (未编程), 则 WDT 始终使能, 而与 SWDTEN 位的设置无关。

6.1 欠压复位 (BOR)

PIC24F16KA102 系列器件具有 BOR 电路, 该电路为用户提供了几种配置和节能选项。BOR 由 <BORV1: BORV0> 和 (BOREN<1:0>) 配置位 (FPOR<6:5, 1:0>) 控制。共有 4 种 BOR 配置, 归纳在表 6.1.1 中。

BOR 门限值由 BORV<1:0> 位设置。如果使能了 BOR (BOREN<1:0> 为除 00 以外的任何值), 只要 VDD 降低到低于所设置的门限值, 就会复位器件。芯片将保持在 BOR 状态, 直到 VDD 升到高于门限值。

如果使能了上电延时定时器, 则在 VDD 升到高于门限值之后, 该定时器就会启动, 因此若 VDD 在上电延时定时器运行的过程中跌落到门限值以下, 那么上电延时定时器就会延长芯片保持复位的时间 (延时长度为 TPWRT)。芯片进入 BOR 状态, 上电延时定时器会被初始化。一旦 VDD 升到高于门限值, 上电延时定时器将执行额外的延时。

BOR 和上电延时定时器是分别配置的。使能 BOR 复位并不会自动使能 PWRT。

6.1.1 用软件使能 BOR

当 BOREN<1:0> = 01 时, 用户可以用软件使能或禁止 BOR。这可通过控制位 SBOREN (RCON<13>) 完成。如前所述, 将 SBOREN 置 1 可使能 BOR。清零 SBOREN 将完全禁止 BOR。SBOREN 位只在该模式下工作; 否则读为 0。

用软件控制 BOR 位可使用户能更灵活地定制应用程序以使其适应环境, 而无需通过对器件再编程来更改 BOR 配置。它还使用户可以定制 BOR 消耗的增量电流。虽然 BOR 的电流通常很小, 但是它可能对低功耗应用有一些影响。

注: 即使当 BOR 受软件控制时, BOR 复位电平仍将由 BORV1: BORV0 配置位设置。该值不能用软件更改。

6.1.2 检测 BOR

当使能 BOR 时, 在发生任何 BOR 或 POR 事件后, BOR 位 (RCON<1>) 总是复位为 1。因此只通过读 BOR 位的状态很难确定是否发生过 BOR 事件。更可靠的方法是同时检查 POR 和 BOR 的状态。假定在发生任何 POR 事件后, POR 和 BOR 位被立即用软件复位为 0。如果 BOR 位为 1 同时 POR 为 0, 那么就可以断定已经发生了 BOR 事件。

注: 即使在器件退出深度休眠模式时, POR 和 BOR 位都会置 1。

6.1.3 在休眠模式下禁止 BOR

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

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

6.1.4 深度休眠 BOR (DSBOR)

深度休眠 BOR 是功耗极低的 BOR 电路。由于电流消耗很低, 所以精确性会因情况而异。DSBOR 在 1.55V 与 1.95V 之间的任意电压下发生。

DSBOR 在配置时通过 BORV<1:0> (FPOR<6:5>) 位 = 00 进行选择。

在电压为 1.55V 与 1.95V 之间的任意时刻, DSBOR 可以重新激活 POR; 但是, 在电压低于 1.55V 时, 一定会发生 POR。

PIC24F16KA102 系列

注:

7.0 中断控制器

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于中断控制器的更多信息，请参见《PIC24F 系列参考手册》的“第 8 章 中断”（DS39707A_CN）。

PIC24F 中断控制器将诸多外设中断请求信号缩减为一个送往 CPU 的中断请求。它具有以下特性：

- 多达 8 个处理器异常和软件陷阱
- 7 个可由用户选择的优先级
- 多达 118 个向量的中断向量表（Interrupt Vector Table, IVT）
- 每个中断或异常源对应一个唯一的向量
- 在指定的用户优先级内具有固定的优先级
- 用于支持调试功能的备用中断向量表（Alternate Interrupt Vector Table, AIVT）
- 固定的中断进入和返回延时

7.1 中断向量表（IVT）

IVT 如图 7-1 所示。IVT 位于程序存储器中，起始存储单元地址是 000004h。IVT 包含 126 个向量，由 8 个不可屏蔽陷阱向量和多达 118 个中断源组成。一般来说，每个中断源都有自己的中断向量。每个中断向量都包含一个 24 位宽的地址。每个中断向量存储单元中设置的值是其相关的中断服务程序（ISR）的起始地址。

中断向量根据它们的自然优先级区分优先次序；也就是说每个中断向量的优先级与其在向量表中的位置有关。如果其他方面都相同，较低地址的中断向量具有较高的自然优先级。例如，与向量 0 相关的中断比任何其他向量地址的中断具有更高的自然优先级。

PIC24F16KA102 系列器件实现了不可屏蔽陷阱和唯一中断；表 7-1 和表 7-2 对此做了总结。

7.1.1 备用中断向量表（AIVT）

备用中断向量表（AIVT）位于 IVT 之后，如图 7-1 所示。由 ALTIVT 控制位（INTCON2<15>）控制对 AIVT 的访问。如果 ALTIVT 位置 1，则所有的中断和异常处理都将使用备用向量，而非默认向量。备用向量与默认向量的组织方式相同。

AIVT 通过提供一种不需要将中断向量再编程就可以在应用程序和支持环境之间切换的方法，来支持仿真和调试功能。此特性也支持运行时在不同应用程序之间切换以便评估各种不同的软件算法。如果不需要 AIVT，则应该用 IVT 中同样的地址编程 AIVT。

7.2 复位过程

器件复位不是真正的异常，因为复位过程中并不涉及到中断控制器。作为对复位的响应，PIC24F 器件清零其寄存器，同时强制程序计数器（PC）为零。然后单片机从地址 000000h 处开始执行程序。用户可以在复位地址中写入 GOTO 指令，将程序执行重定向到相应的启动程序。

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 中所有未实现或未使用的向量存储单元。

PIC24F16KA102 系列

图 7-1: PIC24F 中断向量表

自然优先级降序排列	复位——GOTO 指令	000000h	中断向量表 (IVT) ⁽¹⁾
	复位——GOTO 地址	000002h	
	保留	000004h	
	振荡器故障陷阱向量		
	地址错误陷阱向量		
	堆栈错误陷阱向量		
	数学错误陷阱向量		
	保留		
	保留		
	保留		
	中断向量 0	000014h	
	中断向量 1		
	—		
	—		
	—		
	中断向量 52	00007Ch	
	中断向量 53	00007Eh	
	中断向量 54	000080h	
	—		备用中断向量表 (AIVT) ⁽¹⁾
	—		
	—		
	中断向量 116	0000FCh	
	中断向量 117	0000FEh	
	保留	000100h	
	保留	000102h	
	保留		
	振荡器故障陷阱向量		
	地址错误陷阱向量		
	堆栈错误陷阱向量		
	数学错误陷阱向量		
	保留		
	保留		
	保留		
	中断向量 0	000114h	
	中断向量 1		
	—		
	—		
	—		
	中断向量 52	00017Ch	
	中断向量 53	00017Eh	
	中断向量 54	000180h	
	—		
	—		
	—		
	中断向量 116		
	中断向量 117	0001FEh	
	代码起始单元	000200h	

注 1: 请参见表 7-2 了解中断向量列表。

表 7-1: 陷阱向量详细信息

向量编号	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	数学错误
5	00000Eh	00010Eh	保留
6	000010h	000110h	保留
7	000012h	000112h	保留

表 7-2: 已实现的中断向量

中断源	向量编号	IVT 地址	AIVT 地址	中断位的位置		
				标志	允许	优先级
ADC1 转换完成	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>	IPC1<14:12>
CTMU	77	0000AEh	0001AEh	IFS4<13>	IEC4<13>	IPC19<6:4>
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>
I2C1 主事件	17	000036h	000136h	IFS1<1>	IEC1<1>	IPC4<6:4>
I2C1 从事件	16	000034h	000134h	IFS1<0>	IEC1<0>	IPC4<2:0>
输入捕捉 1	1	000016h	000116h	IFS0<1>	IEC0<1>	IPC0<6:4>
输入电平变化通知	19	00003Ah	00013Ah	IFS1<3>	IEC1<3>	IPC1<14:12>
HLVD 高 / 低压检测	72	0000A4h	0001A4h	IFS4<8>	IEC4<8>	IPC17<2:0>
NVM——NVM 写入完成	15	000032h	000132h	IFS0<15>	IEC0<15>	IPC1<14:12>
输出比较 1	2	000018h	000118h	IFS0<2>	IEC0<2>	IPC0<10:8>
实时时钟 / 日历	62	000090h	000190h	IFS3<14>	IEC3<14>	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>
Timer1	3	00001Ah	00011Ah	IFS0<3>	IEC0<3>	IPC1<14:12>
Timer2	7	000022h	000122h	IFS0<7>	IEC0<7>	IPC1<14:12>
Timer3	8	000024h	000124h	IFS0<8>	IEC0<8>	IPC2<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>	IPC1<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>	IPC1<14:12>

7.3 中断控制和状态寄存器

PIC24F16KA102 系列器件共实现了 22 个用于中断控制器的寄存器：

- INTCON1
- INTCON2
- IFS0、IFS1、IFS3 和 IFS4
- IEC0、IEC1、IEC3 和 IEC4
- IPC0 到 IPC5、IPC7 以及 IPC15 到 IPC19
- INTTREG

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以及 AIV 表的使用。

IFSx 寄存器包含所有中断请求标志。每个中断源都有一个状态位，由各自的外设或外部信号置 1，而由软件清零。

IECx 寄存器包含所有中断允许位。这些控制位用于单独允许外设或外部信号中断。

IPCx 寄存器用于设置每个中断源的中断优先级。可以给每个用户中断源分配为 8 个优先级之一。

INTTREG 寄存器包含相关的中断向量编号和新的 CPU 中断优先级，分别锁存在 INTTREG 寄存器中的向量编号（VECNUM<6:0>）和中断优先级（ILR<3:0>）位域中。新的中断优先级是等待处理中断的优先级。

中断源按表 7-2 中的顺序分配给 IFSx、IECx 和 IPCx 寄存器。例如，INT0（外部中断 0）表示向量编号为 0，自然顺序优先级为 0 的外部中断。所以 INT0IF 状态位在 IFS0<0> 中，INT0IE 允许位在 IEC0<0> 中，INT0IP<2:0> 优先级位在 IPC0 的第一个位域（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。ALU 状态寄存器（SR）包含 IPL<2:0> 位（SR<7:5>）。这些位表示当前 CPU 中断优先级。用户可以通过写 IPL 位来更改当前 CPU 优先级。

CORCON 寄存器包含 IPL3 位，这个位与 IPL<2:0> 位一起表示当前 CPU 优先级。IPL3 是只读位，所以用户软件无法屏蔽陷阱事件。

在下面各小节中的寄存器 7-1 到寄存器 7-21 说明了所有的中断寄存器。

寄存器 7-1: SR: ALU 状态寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R-0, HSC
—	—	—	—	—	—	—	DC ⁽¹⁾
bit 15							bit 8
R/W-0, HSC	R/W-0, HSC	R/W-0, HSC	R-0, HSC	R/W-0, HSC	R/W-0, HSC	R/W-0, HSC	R/W-0, HSC
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:	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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15-9未实现: 读为 0
- bit 7-5IPL<2:0>: 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 了解这些位的说明, 它们并不是专用于中断控制功能。
- 2: IPL 位与 IPL3 位 (CORCON<3>) 组合形成 CPU 中断优先级。如果 IPL3 = 1, 那么括号中的值表示中断优先级。
- 3: 当 NSTDIS (INTCON1<15>) = 1 时, IPL 状态位是只读的。

注: bit 8 以及 bit 4 到 bit 0 如第 2.0 节 “CPU” 中所述。

PIC24F16KA102 系列

寄存器 7-2: CORCON: CPU 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, HSC	R/W-0	U-0	U-0
—	—	—	—	IPL3 ⁽²⁾	PSV ⁽¹⁾	—	—
bit 7				bit 0			

图注:	C = 可清零位	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 = POR 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 ⁽²⁾
1 = CPU 中断优先级大于 7
0 = CPU 中断优先级等于或小于 7

bit 1-0 未实现: 读为 0

注 1: 请参见寄存器 2-2 了解此位的说明, 它并不是专用于中断控制功能。
 2: IPL3 位与 IPL<2:0> 位 (SR<7:5>) 组合形成 CPU 中断优先级。

注: bit 2 如第 2.0 节 “CPU” 中所述。

寄存器 7-3: 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, HS	R/W-0, HS	R/W-0, HS	R/W-0, HS	U-0
—	—	—	MATHERR	ADDRERR	STKERR	OSCFAIL	—
bit 7							bit 0

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

- bit 15

NSTDIS: 中断嵌套禁止位
1 = 禁止中断嵌套
0 = 使能中断嵌套
- bit 14-5

未实现: 读为 0
- bit 4

MATHERR: 算术错误陷阱状态位
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

PIC24F16KA102 系列

寄存器 7-4: INTCON2: 中断控制寄存器 2

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

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

图注:	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 = POR 时的值	1 = 置 1		0 = 清零		x = 未知		

bit 15	ALTIVT: 备用中断向量表使能位 1 = 使用备用中断向量表 0 = 使用标准 (默认) 向量表
bit 14	DISI: DISI 指令状态位 1 = 执行了 DISI 指令 0 = 未执行 DISI 指令
bit 13-3	未实现: 读为 0
bit 2	INT2EP: 外部中断 2 边沿检测极性选择位 1 = 下降沿中断 0 = 上升沿中断
bit 1	INT1EP: 外部中断 1 边沿检测极性选择位 1 = 下降沿中断 0 = 上升沿中断
bit 0	INT0EP: 外部中断 0 边沿检测极性选择位 1 = 下降沿中断 0 = 上升沿中断

寄存器 7-5: IFS0: 中断标志状态寄存器 0

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

R/W-0, HS	U-0	U-0	U-0	R/W-0, HS	R/W-0, HS	R/W-0, HS	R/W-0, HS
T2IF	—	—	—	T1IF	OC1IF	IC1IF	INT0IF
bit 7				bit 0			

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

bit 15	NVMIF: NVM 中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 14	未实现: 读为 0
bit 13	AD1IF: A/D 转换完成中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 12	U1TXIF: UART1 发送器中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 11	U1RXIF: UART1 接收器中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 10	SPI1IF: SPI1 事件中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 9	SPF1IF: SPI1 故障中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 8	T3IF: Timer3 中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 7	T2IF: Timer2 中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 6-4	未实现: 读为 0
bit 3	T1IF: Timer1 中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 2	OC1IF: 输出比较通道 1 中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 1	IC1IF: 输入捕捉通道 1 中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 0	INT0IF: 外部中断 0 标志状态位 1 = 产生了中断请求 0 = 未产生中断请求

PIC24F16KA102 系列

寄存器 7-6: IFS1: 中断标志状态寄存器 1

R/W-0, HS	R/W-0, HS	R/W-0, HS	U-0	U-0	U-0	U-0	U-0
U2TXIF	U2RXIF	INT2IF	—	—	—	—	—
bit 15							bit 8

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

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

bit 15	U2TXIF: UART2 发送器中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 14	U2RXIF: UART2 接收器中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 13	INT2IF: 外部中断 2 标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 12-5	未实现: 读为 0
bit 4	INT1IF: 外部中断 1 标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 3	CNIF: 输入电平变化通知中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 2	CMIF: 比较器中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 1	MI2C1IF: I2C1 主事件中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 0	SI2C1IF: I2C1 从事件中标志状态位 1 = 产生了中断请求 0 = 未产生中断请求

寄存器 7-7: IFS3: 中断标志状态寄存器 3

U-0	R/W-0, HS	U-0	U-0	U-0	U-0	U-0	U-0
—	RTCIF	—	—	—	—	—	—
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:	HS = 硬件置 1 位		
R = 可读位	W = 可写位		U = 未实现位, 读为 0
-n = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15未实现: 读为 0
- bit 14RTCIF: 实时时钟和日历中断标志状态位
1 = 产生了中断请求
0 = 未产生中断请求
- bit 13-0未实现: 读为 0

PIC24F16KA102 系列

寄存器 7-8: IFS4: 中断标志状态寄存器 4

U-0	U-0	R/W-0, HS	U-0	U-0	U-0	U-0	R/W-0, HS
—	—	CTMUIF	—	—	—	—	HLVDIF
bit 15							bit 8

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

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

bit 15-14	未实现: 读为 0
bit 13	CTMUIF: CTMU 中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 12-9	未实现: 读为 0
bit 8	HLVDIF: 高 / 低压检测中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 7-4	未实现: 读为 0
bit 3	CRCIF: CRC 发生器中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 2	U2ERIF: UART2 错误中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 1	U1ERIF: UART1 错误中断标志状态位 1 = 产生了中断请求 0 = 未产生中断请求
bit 0	未实现: 读为 0

寄存器 7-9: IEC0: 中断允许控制寄存器 0

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
NVMIE	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPF1IE	T3IE
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
T2IE	—	—	—	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 = POR 时的值 1 = 置 1 0 = 清零 x = 未知

- bit 15 **NVMIE:** NVM 中断允许位
1 = 允许中断请求
0 = 禁止中断请求
- bit 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 **SPF1IE:** 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 6-4 **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

PIC24F16KA102 系列

寄存器 7-10: IEC1: 中断允许控制寄存器 1

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
U2TXIE	U2RXIE	INT2IE	—	—	—	—	—
bit 15							bit 8

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

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

bit 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 12-5	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: 输入电平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 1-0	未实现: 读为 0

寄存器 7-11: 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	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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15 未实现: 读为 0

bit 14 **RTCIE:** 实时时钟和日历中断允许位

 1 = 允许中断请求

 0 = 禁止中断请求

bit 13-0 未实现: 读为 0

PIC24F16KA102 系列

寄存器 7-12: IEC4: 中断允许控制寄存器 4

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
—	—	CTMUIE	—	—	—	—	HLVDIE
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 = POR 时的值		1 = 置 1		0 = 清零		x = 未知	

bit 15-14	未实现: 读为 0
bit 13	CTMUIE: CTMU 中断允许位 1 = 允许中断请求 0 = 禁止中断请求
bit 12-9	未实现: 读为 0
bit 8	HLVDIE: 高 / 低压检测中断允许位 1 = 允许中断请求 0 = 禁止中断请求
bit 7-4	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 7-13: 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 = POR 时的值	1 = 置 1	0 = 清零 x = 未知

- bit 15 未实现: 读为 0
- bit 14-12 **T1IP<2:0>**: 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 = 禁止中断源
- bit 11 未实现: 读为 0
- bit 10-8 **OC1IP<2:0>**: 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 = 禁止中断源
- bit 7 未实现: 读为 0
- bit 6-4 **IC1IP<2:0>**: 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 = 禁止中断源
- bit 3 未实现: 读为 0
- bit 2-0 **INT0IP<2:0>**: 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 = 禁止中断源

PIC24F16KA102 系列

寄存器 7-14: **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	U-0	U-0	U-0
—	T2IP2	T2IP1	T2IP0	—	—	—	—
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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15	未实现: 读为 0
bit 14-12	T2IP<2:0>: 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 = 禁止中断源
bit 11-0	未实现: 读为 0

寄存器 7-15: 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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15 未实现: 读为 0
- bit 14-12 **U1RXIP<2:0>**: 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 = 禁止中断源
- bit 11 未实现: 读为 0
- bit 10-8 **SPI1IP<2:0>**: 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 = 禁止中断源
- bit 7 未实现: 读为 0
- bit 6-4 **SPF1IP<2:0>**: 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 = 禁止中断源
- bit 3 未实现: 读为 0
- bit 2-0 **T3IP<2:0>**: 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 = 禁止中断源

PIC24F16KA102 系列

寄存器 7-16: **IPC3: 中断优先级控制寄存器 3**

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	NVMIP2	NVMIP1	NVMIP0	—	—	—	—
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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15	未实现: 读为 0
bit 14-12	NVMIP<2:0>: NVM 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 = 禁止中断源
bit 11-7	未实现: 读为 0
bit 6-4	AD1IP<2:0>: 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 = 禁止中断源
bit 3	未实现: 读为 0
bit 2-0	U1TXIP<2:0>: 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 = 禁止中断源

寄存器 7-17: 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
—	MI2C1P2	MI2C1P1	MI2C1P0	—	SI2C1P2	SI2C1P1	SI2C1P0
bit 7				bit 0			

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

- bit 15 未实现: 读为 0
- bit 14-12 **CNIP<2:0>**: 输入电平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 = 禁止中断源
- bit 11 未实现: 读为 0
- bit 10-8 **CMIP<2:0>**: 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 = 禁止中断源
- bit 7 未实现: 读为 0
- bit 6-4 **MI2C1P<2:0>**: I2C1 主事件中断优先级位
111 = 中断优先级为 7 (最高优先级中断)
.
.
.
001 = 中断优先级为 1
000 = 禁止中断源
- bit 3 未实现: 读为 0
- bit 2-0 **SI2C1P<2:0>**: I2C1 从事件中断优先级位
111 = 中断优先级为 7 (最高优先级中断)
.
.
.
001 = 中断优先级为 1
000 = 禁止中断源

PIC24F16KA102 系列

寄存器 7-18: **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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15-3 未实现: 读为 0
- bit 2-0 **INT1IP<2:0>:** 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 = 禁止中断源

寄存器 7-19: 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	U-0	U-0	U-0
—	INT2IP2	INT2IP1	INT2IP0	—	—	—	—
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现: 读为 0

bit 14-12 **U2TXIP<2:0>**: 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 = 禁止中断源

bit 11 未实现: 读为 0

bit 10-8 **U2RXIP<2:0>**: 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 = 禁止中断源

bit 7 未实现: 读为 0

bit 6-4 **INT2IP<2:0>**: 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 = 禁止中断源

bit 3-0 未实现: 读为 0

PIC24F16KA102 系列

寄存器 7-20: **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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15-11	未实现: 读为 0
bit 10-8	RTCIP<2:0>: 实时时钟和日历中断优先级位
	111 = 中断优先级为 7 (最高优先级中断)
	•
	•
	•
	001 = 中断优先级为 1
	000 = 禁止中断源
bit 7-0	未实现: 读为 0

寄存器 7-21: 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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15 未实现: 读为 0

bit 14-12 **CRCIP<2:0>**: 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 = 禁止中断源

bit 11 未实现: 读为 0

bit 10-8 **U2ERIP<2:0>**: 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 = 禁止中断源

bit 7 未实现: 读为 0

bit 6-4 **U1ERIP<2:0>**: 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 = 禁止中断源

bit 3-0 未实现: 读为 0

PIC24F16KA102 系列

寄存器 7-22: IPC18: 中断优先级控制寄存器 18

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
—	—	—	—	—	HLVDIP2	HLVDIP1	HLVDIP0
bit 7							bit 0

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

bit 15-3 未实现: 读为 0
bit 2-0 **HLVDIP<2:0>**: 高 / 低压检测中断优先级位
 111 = 中断优先级为 7 (最高优先级中断)
 .
 .
 .
 001 = 中断优先级为 1
 000 = 禁止中断源

寄存器 7-23: IPC19: 中断优先级控制寄存器 19

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
—	CTMUIP2	CTMUIP1	CTMUIP0	—	—	—	—
bit 7							bit 0

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

bit 15-7 未实现: 读为 0
bit 6-4 **CTMUIP<2:0>**: CTMU 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 = 禁止中断源
bit 3-0 未实现: 读为 0

寄存器 7-24: INTTREG: 中断控制和状态寄存器

R-0	U-0	R/W-0	U-0	R-0	R-0	R-0	R-0
CPUIRQ	—	VHOLD	—	ILR<3:0>			
bit 15				bit 8			
U-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
—	VECNUM<6:0>						
bit 7				bit 0			

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

- bit 15

CPUIRQ: 中断控制器 CPU 中断请求位
1 = 产生了中断请求, 但 CPU 尚未应答 (当 CPU 优先级高于中断优先级时会发生这种情况)
0 = 没有未应答的中断请求
- bit 14

未实现: 读为 0
- bit 13

VHOLD: 允许向量编号捕捉和更改 VECNUM 存储内容的位
1 = VECNUM 将包含优先级最高的待处理中断的向量编号, 而不是当前中断的向量编号
0 = VECNUM 将包含上一次响应的中断 (已产生的优先级高于 CPU 的上一次中断, 即使有其他中断待处理) 的向量编号
- bit 12

未实现: 读为 0
- bit 11-8

ILR<3:0>: 新的 CPU 中断优先级位
1111 = CPU 中断优先级为 15
.
.
.
0001 = CPU 中断优先级为 1
0000 = CPU 中断优先级为 0
- bit 7

未实现: 读为 0
- bit 6-0

VECNUM<6:0>: 待处理中断向量编号位
0111111 = 待处理中断向量的编号为 135
.
.
.
0000001 = 待处理中断向量的编号为 9
0000000 = 待处理中断向量的编号为 8

7.4 中断设置过程

7.4.1 初始化

要配置中断源：

1. 如果不需要嵌套中断，则将 **NSTDIS** 控制位 (**INTCON1<15>**) 置 1。
2. 通过写相应 **IPCx** 寄存器中的控制位为中断源选择由用户分配的优先级。优先级将取决于具体的应用和中断源类型。如果不需要多个优先级，则可以将所有允许中断源的 **IPCx** 寄存器控制位编程为相同的非零值。

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

3. 将相应 **IFSx** 寄存器中与外设相关的中断标志状态位清零。
4. 通过将相应 **IECx** 寄存器中与中断源相关的中断允许控制位置 1 来允许中断源。

7.4.2 中断服务程序

如何声明**ISR**以及怎样使用正确的向量地址初始化**IVT**，取决于编程语言（即，**C** 语言或汇编语言）和用于开发应用程序的语言开发工具包。一般情况下，用户必须将相应 **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，在退出程序后将立即再次进入 **ISR**。如果 **ISR** 用汇编语言编码，则必须使用 **RETFIE** 指令结束 **ISR**，以便将保存的 **PC** 值、**SRL** 值和原先的 **CPU** 优先级弹出堆栈。

7.4.3 陷阱服务程序（TSR）

除了必须清零 **INTCON1** 寄存器中相应的陷阱状态标志来避免重新进入陷阱服务程序（**Trap Service Routine, TSR**）之外，**TSR** 使用与 **ISR** 类似的方式编写。

7.4.4 中断禁止

可以使用以下步骤禁止所有用户中断：

1. 使用 **PUSH** 指令将当前的 **SR** 值压入软件堆栈。
2. 通过将值 **OEH** 与 **SRL** 进行逻辑或运算来强制将 **CPU** 的优先级设置为 7。

要允许用户中断，则可以使用 **POP** 指令恢复先前的 **SR** 值。

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。

8.0 振荡器配置

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于振荡器配置的更多信息，请参见《PIC24F 系列参考手册》的“第 38 章带 500 kHz 低功耗 FRC 的振荡器”（DS39726A_CN）。

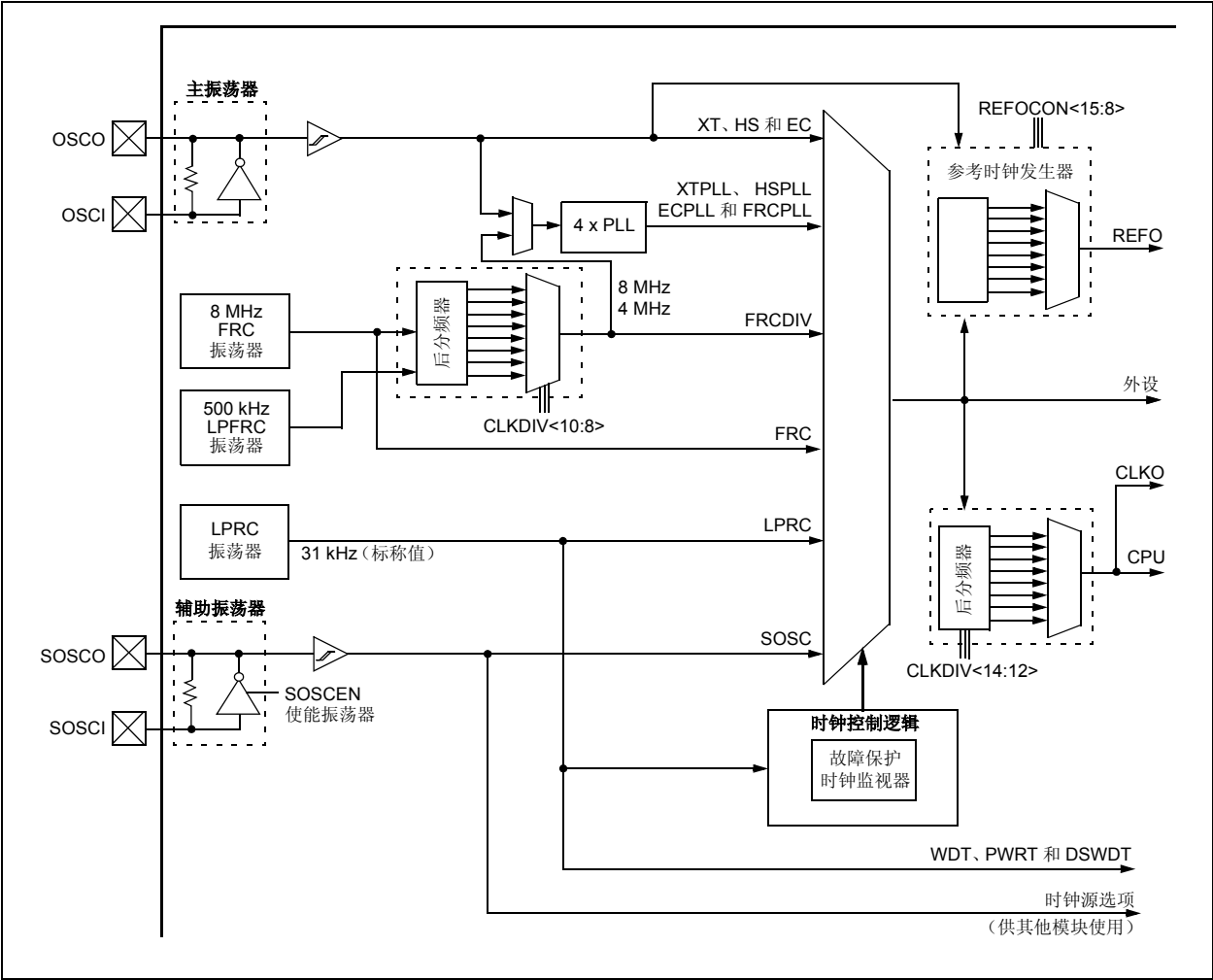
PIC24F16KA102 系列器件的振荡器系统具有以下特性：

- 可选择 5 个外部和内部振荡器作为时钟源，提供 11 种不同的时钟模式。
- 片上 4 倍频锁相环（PLL），可基于选定的内部和外部振荡器源升高内部工作频率。

- 可通过软件控制在各种时钟源之间切换。
- 可通过软件控制预分频器有选择地为 CPU 提供时钟，以节省系统功耗。
- 用于 EC 模式的系统频率范围声明位。使用外部时钟源时，通过将声明位设置为期望的频率范围可以降低电流消耗。
- 具有故障保护时钟监视器（FSCM），可检测时钟故障，以使应用安全地恢复或关闭。

图 8-1 给出了振荡器系统的简化框图。

图 8-1: PIC24F16KA102 系列时钟框图



PIC24F16KA102 系列

8.1 CPU 时钟机制

系统时钟源可以由以下 4 种之一提供：

- OSCI 和 OSCO 引脚上的主振荡器（POSC）
- SOSCI 和 SOSCO 引脚上的辅助振荡器（SOSC）

PIC24F16KA102 系列器件有两种类型的辅助振荡器：

- 高功耗辅助振荡器
- 低功耗辅助振荡器

这些可以通过 SOSCSEL（FOSC<5>）位进行选择。

- 内部快速 RC（FRC）振荡器
 - 8 MHz FRC 振荡器
 - 500 kHz 低功耗 FRC 振荡器
- 内部低功耗 RC（LPRC）振荡器

主振荡器和 8 MHz FRC 源可以选择使用内部 4 倍频 PLL。FRC 时钟源的频率可选择通过可编程时钟分频器降低。选定的时钟源将产生处理器和外设时钟。

处理器时钟源需进行二分频，以产生内部指令周期时钟 Fcy。在本文档中，指令周期时钟也表示为 Fosc/2。内部指令周期时钟 Fosc/2 可以在 OSCO I/O 引脚提供，用于主振荡器的一些工作模式。

8.2 POR 时的初始配置

在发生器件上电复位（POR）事件时使用的振荡器源（以及工作模式）使用配置位设置进行选择。振荡器配置位在程序存储器中的配置寄存器中进行设置（更多详细信息，请参见第 25.1 节“配置位”）。主振荡器配置位 POSCMD<1:0>（FOSC<1:0>）和初始振荡器选择配置位 FNOSC<2:0>（FOSCSEL<2:0>）用于选择在 POR 时使用的振荡器源。默认情况下（未编程时）将选择带后分频器的 FRC 主振荡器（FRCDIV）。通过编程这些位，可以选择辅助振荡器或一个内部振荡器。运行于 EC 模式时，EC 模式频率范围配置位 POSCFREQ<1:0>（FOSC<4:3>）可用于优化功耗。默认配置是“频率范围大于 8 MHz”。

这些配置位使用户可以选择多种时钟模式，如表 8-1 所示。

8.2.1 时钟切换模式配置位

FCKSM 配置位（FOSC<7:6>）用于联合配置器件时钟切换和 FSCM。只有将 FCKSM1 编程（0）时，才会使能时钟切换。只有同时将 FCKSM<1:0> 编程（00）时，才会使能 FSCM。

表 8-1： 时钟选择的配置位值

振荡器模式	振荡器源	POSCMD<1:0>	FNOSC<2:0>	注
带后分频器的 8 MHz FRC 振荡器（FRCDIV）	内部	11	111	1, 2
带后分频器的 500 MHz FRC 振荡器（LPFRCDIV）	内部	11	110	1
低功耗 RC 振荡器（LPRC）	内部	11	101	1
辅助（Timer1）振荡器（SOSC）	辅助	00	100	1
带 PLL 模块的主振荡器（HS）（HSPLL）	主	10	011	
带 PLL 模块的主振荡器（EC）（ECPLL）	主	00	011	
主振荡器（HS）	主	10	010	
主振荡器（XT）	主	01	010	
主振荡器（EC）	主	00	010	
带 PLL 模块的 8 MHz FRC 振荡器（FRCPLL）	内部	11	001	1
8 MHz FRC 振荡器（FRC）	内部	11	000	1

注 1： OSCO 引脚功能由 OSCIOFNC 配置位决定。

2： 这是未编程（已擦除）器件的默认振荡器模式。

8.3 控制寄存器

振荡器的操作由 3 个特殊功能寄存器（SFR）控制：

- OSCCON
- CLKDIV
- OSCTUN

OSCCON 寄存器（寄存器 8-1）是振荡器的主控制寄存器。它监视时钟源并控制时钟源切换。

时钟分频寄存器（寄存器 8-2）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 8-3）使用户可以对 FRC 振荡器在大约 $\pm 12\%$ 的范围内进行微调。每个位的递增或递减都会将 FRC 振荡器的出厂校准频率改变一个固定的量。

寄存器 8-1: OSCCON: 振荡器控制寄存器

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

图注:	CO = 只可清零位	SO = 只可置 1 位
	HS = 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 = POR 时的值	1 = 置 1	0 = 清零
		x = 未知

bit 15 未实现: 读为 0

bit 14-12 **COSC<2:0>**: 当前振荡器选择位

- 111 = 带后分频器的 8 MHz 快速 RC 振荡器 (FRCDIV)
- 110 = 带后分频器的 500 kHz 低功耗快速 RC 振荡器 (FRC) (LPFRCDIV)
- 101 = 低功耗 RC 振荡器 (LPRC)
- 100 = 辅助振荡器 (SOSC)
- 011 = 带 PLL 模块的主振荡器 (XTPLL、HSPLL 和 ECPLL)
- 010 = 主振荡器 (XT、HS 和 EC)
- 001 = 带后分频器和 PLL 模块的 8 MHz FRC 振荡器 (FRCPLL)
- 000 = 8 MHz FRC 振荡器 (FRC)

bit 11 未实现: 读为 0

bit 10-8 **NOSC<2:0>**: 新振荡器选择位 ⁽¹⁾

- 111 = 带后分频器的 8 MHz 快速 RC 振荡器 (FRCDIV)
- 110 = 带后分频器的 500 kHz 低功耗快速 RC 振荡器 (FRC) (LPFRCDIV)
- 101 = 低功耗 RC 振荡器 (LPRC)
- 100 = 辅助振荡器 (SOSC)
- 011 = 带 PLL 模块的主振荡器 (XTPLL、HSPLL 和 ECPLL)
- 010 = 主振荡器 (XT、HS 和 EC)
- 001 = 带后分频器和 PLL 模块的 8 MHz FRC 振荡器 (FRCPLL)
- 000 = 8 MHz FRC 振荡器 (FRC)

注 1: 这些位的复位值由 FNOSC 配置位决定。

注 2: 在任意有效时钟切换期间, 或者每当选择了非 PLL 时钟模式时, 也复位为 0。

PIC24F16KA102 系列

寄存器 8-1: OSCCON: 振荡器控制寄存器 (续)

bit 7	CLKLOCK: 时钟选择锁定使能位 <u>如果使能 FSCM (FCKSM1 = 1):</u> 1 = 时钟和 PLL 选择被锁定 0 = 时钟和 PLL 选择未锁定, 可以通过将 OSWEN 位置 1 进行修改 <u>如果禁止 FSCM (FCKSM1 = 0):</u> 时钟和 PLL 选择始终未锁定, 可以通过将 OSWEN 位置 1 进行修改。
bit 6	未实现: 读为 0
bit 5	LOCK: PLL 锁定状态位 ⁽²⁾ 1 = PLL 模块处于锁定状态, 或 PLL 模块起振定时器延时结束 0 = PLL 模块处于失锁状态, PLL 起振定时器正在运行或 PLL 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 = 未检测到时钟故障
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 = 启动振荡器切换, 切换为由 NOSC<2:0> 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: 在任意有效时钟切换期间, 或者每当选择了非 PLL 时钟模式时, 也复位为 0。

寄存器 8-2: CLKDIV: 时钟分频寄存器

R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	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:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15 **ROI:** 中断恢复位
1 = 发生中断时清零 DOZEN 位, 并将 CPU 和外设时钟比复位为 1:1
0 = 中断对 DOZEN 位没有影响
- bit 14-12 **DOZE<2:0>:** CPU 和外设时钟比选择位
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:** 打盹使能位 ⁽¹⁾
1 = DOZE<2:0> 位指定 CPU 和外设时钟比
0 = CPU 和外设时钟比设置为 1:1
- bit 10-8 **RCDIV<2:0>:** FRC 后分频比选择位
当 OSCCON (COSC<2:0>) = 111 时:
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) (默认)
000 = 8 MHz (1 分频)
当 OSCCON (COSC<2:0>) = 110 时:
111 = 1.95 kHz (256 分频)
110 = 7.81 kHz (64 分频)
101 = 15.62 kHz (32 分频)
100 = 31.25 kHz (16 分频)
011 = 62.5 kHz (8 分频)
010 = 125 kHz (4 分频)
001 = 250 kHz (2 分频) (默认)
000 = 500 kHz (1 分频)
- bit 7-0 未实现: 读为 0

注 1: 该位在 ROI 位置 1 和发生中断时自动清零。

PIC24F16KA102 系列

寄存器 8-3: **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	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	TUN5 ⁽¹⁾	TUN4 ⁽¹⁾	TUN3 ⁽¹⁾	TUN2 ⁽¹⁾	TUN1 ⁽¹⁾	TUN0 ⁽¹⁾
bit 7							bit 0

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

bit 15-6 **未实现: 读为 0**

bit 5-0 **TUN<5:0>: FRC 振荡器调节位 ⁽¹⁾**

 011111 = 最大频率偏移

 011110

 .

 .

 .

 000001

 000000 = 中心频率, 振荡器以出厂校准频率工作

 111111

 .

 .

 .

 100001

 100000 = 最小频率偏移

注 1: TUN<5:0> 的递增或递减不能在 FRC 调节范围内同步更改 FRC 频率, 并且频率的变化也可能不是单调的。

8.4 时钟切换工作原理

在软件控制下，应用可以在任何时候在四个时钟源（POSC、SOSC、FRC 和 LPRC）之间自由切换，几乎没有什么限制。为限制这种灵活性可能产生的负面影响，PIC24F 器件在切换过程中带有安全锁定。

注： 主振荡器模式有三种不同的子模式（XT、HS 和 EC），它们由 POSCMDx 配置位决定。在应用中可以用软件实现从主振荡器模式切换到其他模式，或从其他模式切换到主振荡器模式，但不能在不对器件进行再编程的情况下在主振荡器模式的不同子模式之间进行切换。

8.4.1 使能时钟切换

要使能时钟切换，FOSC 配置寄存器中的 FCKSM1 配置位必须编程为 0。（更多详细信息，请参见第 25.1 节“配置位”。）如果 FCKSM1 配置位未被编程（值为 1），则时钟切换功能和 FSCM 功能被禁止。这是默认设置。

当时钟切换被禁止时，NOSCx 控制位（OSCCON<10:8>）不控制时钟选择。但是，COSCx 位（OSCCON<14:12>）反映由 FNOSCx 配置位选择的时钟源。

在时钟切换被禁止时，OSWEN 控制位（OSCCON<0>）不起作用；它始终保持为 0。

8.4.2 振荡器切换过程

执行时钟切换需要以下基本过程：

1. 如果需要，读 COSCx 位（OSCCON<14:12>）以确定当前的振荡器源。
2. 执行解锁序列以允许写入 OSCCON 寄存器的高字节。
3. 将适当的值写入 NOSCx 位（OSCCON<10:8>）选择新振荡器源。
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 以启动振荡器切换。

一旦基本过程完成，系统时钟硬件将自动进行如下响应：

1. 时钟切换硬件将 NOSCx 位的新值与 COSCx 位进行比较。如果它们相同，则时钟切换是冗余操作。在这种情况下，OSWEN 位自动清零，时钟切换中止。
2. 如果启动了有效的时钟切换，则 LOCK（OSCCON<5>）和 CF（OSCCON<3>）位清零。
3. 如果新振荡器现在不在运行，则硬件会将它开启。如果开启的是晶振，则硬件将等待直到 OST 超时。如果新的振荡器源使用 PLL，则硬件将等待直到检测到 PLL 锁定（LOCK = 1）。
4. 硬件会等待新时钟源的 10 个时钟周期，然后执行时钟切换。
5. 硬件清零 OSWEN 位表示时钟转换成功。此外，NOSCx 位的值被传送到 COSCx 位。
6. 此时旧时钟源被关闭，LPRC（如果以 LPRC 作为时钟源的 WDT、FSCM 或 RTCC 被使能）或 SOSC（如果 SOSCEEN 保持使能）除外。

注 1: 在整个时钟切换过程中，处理器将继续执行代码。对时序敏感的代码不应在此时执行。

2: 不允许直接在使能 PLL 的任何主振荡器模式和 FRCPLL 模式之间进行时钟切换。这适用于任何方向下的时钟切换。在这些情况下，应用必须首先切换到 FRC 模式将其作为两个 PLL 模式之间的过渡时钟源。

PIC24F16KA102 系列

时钟切换的建议代码序列如下：

1. 禁止在OSCCON寄存器解锁和写序列期间的中断。
2. 在两条连续的指令中将 78h 和 9Ah 写入 OSCCON<15:8>，以执行 OSCCON 高字节的解锁序列。
3. 在紧接解锁序列之后的指令中将新的振荡器源写入 NOSC_x 位。
4. 在两条连续的指令中将 46h 和 57h 写入 OSCCON<7:0>，以执行 OSCCON 低字节的解锁序列。
5. 在紧接解锁序列之后的指令中将 OSWEN 位置 1。
6. 继续执行对时钟不敏感的代码（可选）。
7. 调用相应数量的软件延时（周期计数），以使选定的振荡器和 / 或 PLL 启动并稳定。
8. 检查 OSWEN 是否为 0。如果为 0，则说明切换成功。如果 OSWEN 仍然置 1，则检查 LOCK 位以确定故障的原因。

例 8-1 中显示了解锁 OSCCON 寄存器和启动时钟切换的核心序列。

例 8-1： 时钟切换的基本代码序列

```
;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
```

8.5 参考时钟输出

除了某些振荡器模式中可用的 CLKO 输出（Fosc/2）外，PIC24F16KA102 系列器件中的器件时钟也可以配置为向端口引脚提供参考时钟输出信号。该功能在所有振荡器配置中都可用，允许用户选择更大范围的时钟分频因数来驱动应用中的外部器件。

该参考时钟输出由 REFOCON 寄存器（寄存器 8-4）控制。将 ROEN 位（REFOCON<15>）置 1 将使时钟信号在 REFO 引脚上可用。RODIV 位（REFOCON<11:8>）允许选择 16 种不同的时钟分频比。

ROSSLP 和 ROSEL 位（REFOCON<13:12>）控制休眠模式下参考输出的可用性。ROSEL 位决定用 OSC1 和 OSC2 上的振荡器还是当前系统时钟源作为参考时钟输出。ROSSLP 位决定器件处于休眠模式时 REFO 上的参考时钟源是否可用。

要在休眠模式下使用参考时钟输出，ROSSLP 和 ROSEL 位都必须置 1。器件时钟也必须配置为主模式之一（EC、HS 或 XT）；否则，如果 ROSEL 位没有同时置 1，OSC1 和 OSC2 上的振荡器将在器件进入休眠模式后掉电。清零 ROSEL 位允许参考输出频率在任何时钟切换期间随着系统时钟的改变而改变。

寄存器 8-4: REFOCON: 参考振荡器控制寄存器

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ROEN	—	ROSSLP	ROSEL	RODIV3	RODIV2	RODIV1	RODIV0
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 = POR 时的值	1 = 置 1	0 = 清零 x = 未知

- bit 15 **ROEN:** 参考振荡器输出使能位
1 = 在 REFO 引脚上使能参考振荡器
0 = 禁止参考振荡器
- bit 14 **未实现:** 读为 0
- bit 13 **ROSSLP:** 参考振荡器输出在休眠模式下停止位
1 = 参考振荡器在休眠模式下继续运行
0 = 参考振荡器在休眠模式下被禁止
- bit 12 **ROSEL:** 参考振荡器源选择位
1 = 主振荡器用作基本时钟 ⁽¹⁾
0 = 系统时钟用作基本时钟; 基本时钟反映器件的任何时钟切换
- bit 11-8 **RODIV3:RODIV0:** 参考振荡器分频比选择位
1111 = 基本时钟值被 32,768 分频
1110 = 基本时钟值被 16,384 分频
1101 = 基本时钟值被 8,192 分频
1100 = 基本时钟值被 4,096 分频
1011 = 基本时钟值被 2,048 分频
1010 = 基本时钟值被 1,024 分频
1001 = 基本时钟值被 512 分频
1000 = 基本时钟值被 256 分频
0111 = 基本时钟值被 128 分频
0110 = 基本时钟值被 64 分频
0101 = 基本时钟值被 32 分频
0100 = 基本时钟值被 16 分频
0011 = 基本时钟值被 8 分频
0010 = 基本时钟值被 4 分频
0001 = 基本时钟值被 2 分频
0000 = 基本时钟值
- bit 7-0 **未实现:** 读为 0

注 1: 晶振必须使用 FOSC<2:0> 位使能; 在休眠模式下, 晶振会保持工作。

PIC24F16KA102 系列

注:

9.0 节能特性

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于节能特性的更多信息，请参见《PIC24F 系列参考手册》的“第 39 章 具有深度休眠的节能特性”（DS39727A_CN）。

PIC24F16KA102 系列器件提供了管理功耗的功能，该功能是通过有选择地管理 CPU 和外设的时钟源来实现的。一般来说，较低的时钟频率和减少时钟源驱动电路的数目可降低功耗。这些 PIC24F 器件可通过以下四种不同的方式管理功耗：

- 时钟频率
- 基于指令的休眠、空闲和深度休眠模式
- 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关键应用功能（如对于时序敏感的通信）的情况下有选择地调节应用的功耗。

9.1 时钟频率和时钟切换

PIC24F 器件提供的时钟频率范围较宽，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，已在第 8.0 节“振荡器配置”中进行了更详细的讨论。

9.2 基于指令的节能模式

PIC24F16KA102 系列器件有三种特殊的节能模式，通过执行特殊的 PWRSAV 指令可以进入这三种模式。休眠模式下时钟停止工作并暂停所有代码执行；空闲模式下 CPU 暂停工作并暂停代码执行，但是允许外设模块继续工作。深度休眠模式下时钟停止工作、暂停代码执行、外设（除 RTCC 和 DSWDT 外）停止工作、冻结 I/O 状态并且不对 SRAM 和闪存供电。例 9-1 中所示为 PWRSAV 指令的汇编语法。

在被允许的中断产生、WDT 超时或器件复位时，器件会退出休眠和空闲模式。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头文件中定义的常量。

9.2.1 空闲模式

空闲模式具有以下特性：

- CPU 将停止执行指令。
- WDT 被自动清零。
- 系统时钟源保持工作状态。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 9.4 节“选择性外设模块控制”）。
- 如果 WDT 或 FSCM 被使能，则 LPRC 也将保持工作状态。

当发生以下任何事件时，器件将被从空闲模式唤醒：

- 产生任何被单独允许的中断
- 任何器件复位
- WDT 超时

从空闲模式唤醒时，重新为 CPU 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。

9.2.2 休眠模式

休眠模式具有以下特性：

- 系统时钟源关闭。如果使用了片上振荡器，也要关闭它。
- 所有 I/O 引脚均无拉电流，此时器件电流消耗最少。
- 由于系统时钟源被禁止，所以故障保护时钟监视器（FSCM）在休眠模式下不工作。
- 如果 WDT 被使能，则 LPRC 时钟将在休眠模式下继续运行。
- 如果使能了片上稳压器，则其欠压复位（BOR）电路在休眠模式下保持工作。
- 如果 WDT 被使能，则在进入休眠模式之前被自动清零。
- 某些外设可能在休眠模式下继续工作。这些外设包括检测输入信号电平变化的 I/O 引脚，或使用外部时钟输入的外设。

PIC24F16KA102 系列

任何需要使用系统时钟源来工作的外设，在休眠模式下都将被禁止。当发生以下任何事件时，处理器将从休眠模式退出或被唤醒：

- 产生任何被单独允许的中断。
- 任何形式的器件复位。
- WDT 超时。
- 闪存偏置电压由 RCON 中的 PMSLP 位 (RCON<8>) 控制。默认情况下，该位清零；在休眠模式下时，这会关闭闪存的电源。如果 PMSLP 置 1，则在休眠期间，闪存的电源将保持有效，使得可以更快地唤醒。但在这种情况下，休眠模式的电流较高。

当发生以下任何事件时，器件将从休眠模式唤醒：

- 产生任何被单独允许的中断
- 任何形式的器件复位
- WDT 超时

从休眠模式唤醒时，处理器将使用在进入休眠模式时处于工作状态的时钟源重新开始工作，例外情况是从 POR 或 BOR 唤醒或使能了双速启动。

例 9-1: PWRSAV 指令语法

PWRSAV	#SLEEP_MODE	; Put the device into SLEEP mode
PWRSAV	#IDLE_MODE	; Put the device into IDLE mode
BSET	DSCON, DSEN	; Enable Deep Sleep
PWRSAV	#SLEEP_MODE	; Put the device into Deep SLEEP mode

9.2.3 深度休眠模式

深度休眠模式用于实现最低程度的功耗，而无需使用外部开关来完全关闭器件的所有电源。进入深度休眠模式完全由软件控制。从深度休眠模式退出可以通过以下任意事件触发：

- POR 事件
- MCLR 事件
- RTCC 闹钟（如果有 RTCC）
- 外部中断 0
- 深度休眠看门狗定时器超时

在深度休眠模式下，可以让器件实时时钟和日历（RTCC）保持运行，从而不会失去时钟周期。

器件具有专用的深度休眠欠压复位（DSBOR）和深度休眠看门狗定时器（DSWDT）复位电路，用于监视电压和超时事件。DSBOR 和 DSWDT 独立于用于其他功耗管理模式（休眠、空闲和打盹）的标准 BOR 和 WDT。

9.2.3.1 准备深度休眠

因为在深度休眠模式下时，会关闭 SRAM 的电源，所以 SRAM 数据会丢失。退出深度休眠模式也会产生 POR，因此大多数特殊功能寄存器将复位为它们默认的 POR 值。

需要保存关键数据的应用程序可以使用深度休眠通用寄存器 DSGPR0 和 DSGPR1，或者数据 EEPROM。器件处于深度休眠模式时，这些寄存器的内容会被保留。退出深度休眠模式之后，软件可以通过读取寄存器并清零 RELEASE 位（DSCON<0>）来恢复数据。

9.2.3.2 使能 DSWDT

要在深度休眠模式下使能 DSWDT，应将配置位 DSWDTEN（FDS<7>）置 1。要使 DSWDT 正常工作，需要使能器件看门狗定时器（WDT）。进入深度休眠模式时，将会自动复位 DSWDT。

9.2.3.3 配置 DSWDT

DSWDT 的时钟源通过配置位 DSWCKSEL（FDS<4>）进行配置，后分频比可以通过配置位 DSWDTPS（FDS<3:0>）进行配置。可以实现的最小超时周期为 2.1 ms，最大超时周期为 25.7 天（关于 FDS 配置寄存器的更多详细信息，请参见第 25.0 节“特殊功能”）。

9.2.3.4 在深度休眠模式下切换时钟

RTCC 和深度休眠看门狗定时器 (DSWDT) 都可以使用 SOSC 或 LPRC 时钟源。这使 RTCC 和 DSWDT 可以在无需同时使能 LPRC 和 SOSC 的情况下运行,从而降低功耗。

RTCC 使用 LPRC 时钟源运行时, RTCC 的精度会降低大约 5-10%。如果要求 RTCC 保持高精度,则它必须使用 SOSC 时钟源。

在某些情况下,当进入深度休眠模式时,DSWDT 的时钟源可能已关闭。这种情况下,将会自动打开时钟源(如果 DSWDT 已使能),而无需软件干预。但是,这会使 DSWDT 计数器的启动产生延时。在使用 SOSC 作为时钟源时,为了避免这种延时,应使用软件在进入深度休眠模式之前的某一时间先激活 SOSC。

9.2.3.5 进入深度休眠模式

进入深度休眠模式的方式为:将 DSCON 寄存器中的 DSEN 位置 1,然后在一个指令周期内执行 SLEEP 命令(PWRSV #SLEEP_MODE),以最大程度降低意外进入深度休眠的机会。如果未在一个指令周期内执行 SLEEP 命令,硬件会将 DSEN 位清零,在进入深度休眠模式之前,软件必须重新将该位置 1。退出深度休眠模式时,DSEN 位将自动清零。

注: 要在从深度休眠唤醒之后重新进入深度休眠模式,则在清零 RELEASE 位之后,必须经过至少 3 个 Tcy 的延时。

进入深度休眠模式的序列如下:

1. 使能 DSWDT (可选,应通过器件配置位完成)。
 2. 配置 DSWDT 时钟源 (可选,应通过器件配置位完成)。
 3. 使能 DSBOR (可选,应通过器件配置位完成)。
 4. 使能 RTCC (可选)。
 5. 通过写入 DSGPR0 和 DSGPR1 寄存器来保护现场关键参考数据 (可选)。
 6. 配置 RTCC 外设 (可选)。
 7. 通过将 DSEN 位 (DSCON<7>) 置 1 使能深度休眠模式。
 8. 通过发出 PWRSV #0 指令进入深度休眠模式。
- 每次 DSEN 位置 1 时,DSWSRC 寄存器将由硬件复位。

9.2.3.6 退出深度休眠模式

深度休眠模式在发生以下任一事件时退出:

1. VDD 电源发生 POR 事件。如果没有 DSBOR 电路用于重新激活 VDD 电源 POR 电路,则外部 VDD 电源必须降到 POR 电路的自然激活电压。
2. DSWDT 超时。当 DSWDT 定时器发生超时,将会退出深度休眠模式。
3. RTCC 闹钟(如果 RTCEN = 1,并且有 RTCC)。
4. MCLR 引脚置为有效 (0)。
5. INT0 引脚置为有效 (如果在进入深度休眠之前允许了中断)。极性配置用于确定引脚的有效电平 (0 或 1)。进入深度休眠模式时,所有待处理中断都会被清除;处于深度休眠模式时,需要在 INT0 引脚上发生电平变化时才会从深度休眠模式退出。

退出深度休眠模式相当于器件上电复位 (POR),器件通常不会保持原有状态。这种情况的例外情况包括 RTCC (如果有,它在唤醒后将继续保持工作)、DSGPRx 寄存器和 DSWDT。

此外,从退出深度休眠模式到 POR 序列完成之前的所有唤醒事件都会被忽略,不会被捕捉到 DSWSRC 寄存器中。通常,只有多个唤醒事件同时发生时,DSWSRC 寄存器才会有多个位置 1。

9.2.3.7 检查和清除深度休眠状态

在进入深度休眠模式时,状态位 DPSLP (RCON<10>) 会置 1,必须用软件清零。

在引导时,软件应读取该状态位,以确定是否是由于从深度休眠模式退出而发生复位,如果该位置 1,则清零该位。可以考虑 4 种可能的情况:

- 深度休眠位 (DPSLP) 和 POR 位均清零。这种情况下,复位是由于退出深度休眠模式之外的某个其他事件而发生的。
- DPSLP 位清零,但 POR 位置 1;这是正常的 POR。
- DPSLP 位置 1,但 POR 位清零。如果进入深度休眠模式,就会强制发生 POR;所以这种组合是不可能的。
- DPSLP 和 POR 位均置 1。这表示发生了以下情况:进入深度休眠模式、器件发生掉电,然后从深度休眠模式退出。

9.2.3.8 软件恢复

虽然退出深度休眠类似于器件发生POR，但提供了一种机制，用于在退出深度休眠模式时恢复器件状态。进入深度休眠模式时，深度休眠状态保持位 **RELEASE** (**DSCON<0>**) 会置 1。虽然 **TRISx** 和 **LATx** 寄存器会复位，但只要 **RELEASE** 保持置 1，I/O 将维持在进入深度休眠模式之前的状态。通过该功能及 **DSGPRx** 寄存器，软件可以恢复器件的状态。当 **RELEASE** 置 1 时，修改 **TRISx** 和 **LATx** 对 I/O 引脚不起作用。完成器件的重新配置时，可以通过清零 **RELEASE** 来安全地释放 I/O 状态。

注： 当深度休眠唤醒是由于 **MCLR** 或 **POR** 而发生时，则无法进行软件恢复。这种情况下，I/O 将被释放为器件复位状态，**RELEASE** 将被硬件清零。**DSGPRx** 寄存器只能通过器件 **POR** 复位。

9.2.3.9 上电复位 (POR)

器件通过监视 **VDD** 电压来产生 **POR**。因为进入深度休眠在功能上类似于 **POR**，所以应使用第 9.2.3.7 节“**检查和清除深度休眠状态**”中描述的技术来区分深度休眠和真正的 **POR** 事件。

发生 **POR** 时，包括所有深度休眠逻辑（深度休眠寄存器、**RTCC** 和 **DSWDT** 等）在内的整个器件将复位。

9.2.3.10 深度休眠序列汇总

本节提供在器件上使用深度休眠模式的典型事件序列：

1. 器件退出复位。
2. 如果需要 **DSWDT** 功能，则必须将相应的熔丝位置 1。
3. 为 **DSWDT** 和 **RTCC** 选择相应的时钟（可选）。
4. 使能 / 配置 **DSWDT**（可选）。
5. 使能 / 配置 **RTCC**（可选）。
6. 写 **DSGPRx** 寄存器（可选）。
7. 允许 **INT0** 中断（可选）。
8. 将 **DSCON** 寄存器中的 **DSEN** 位置 1。
9. 通过发出 **SLEEP** 命令进入深度休眠模式。
10. 器件由于唤醒事件而退出深度休眠。

11. **DSEN** 位被自动清零。
12. 读取深度休眠状态位并将它清零。
13. 读 **DSGPRx** 寄存器（可选）。
14. 完成所有与状态相关的配置之后，将 **DSCON** 寄存器中的 **RELEASE** 位清零。器件现在会释放所有保持的逻辑和 I/O。在此之前，I/O 的控制和数据位对实际的 I/O 状态不起作用。
15. 软件恢复正常运行。

9.2.3.11 深度休眠期间的 I/O 引脚

在深度休眠期间，通用 I/O 引脚将保持它们原先的状态。在进入深度休眠模式之前配置为输入（**TRIS** 位置 1）的引脚在深度休眠期间将保持高阻态。

在进入深度休眠模式之前配置为输出（**TRIS** 位清零）的引脚在深度休眠期间将保持为输出引脚。处于该模式时，它们驱动的输出电平将由进入深度休眠模式时它们相应的 **LAT** 位决定。

当器件唤醒时，I/O 引脚的行为取决于唤醒源的类型。如果器件是由 **RTCC** 闹钟、**INT0** 中断或 **DSWDT** 事件唤醒，所有 I/O 引脚将继续维持它们原先的状态，即使器件已经完成 **POR** 序列，并再次执行应用程序代码。配置为输入的引脚在深度休眠期间将保持高阻态，配置为输出的引脚将继续驱动输出它们原先的值。唤醒之后，**TRIS** 和 **LAT** 寄存器将复位，但 I/O 引脚将仍然维持它们原先的状态。如果固件修改了 I/O 引脚的 **TRIS** 和 **LAT** 值，它们不会立即切换到新配置的状态。在固件清零 **RELEASE** 位 (**DSCON<0>**) 之后，I/O 引脚将被“释放”。这会导致 I/O 引脚切换为由它们相应的 **TRIS** 和 **LAT** 位值配置的状态。

如果使能了深度休眠 **BOR** (**DSBOR**) 电路，并且 **VDD** 降低到低于 **DSBOR** 和 **VDD** 轨 **POR** 门限值，或 **MCLR** 被置为有效，I/O 引脚将被立即释放，类似于清零 **RELEASE** 位。所有原先的状态信息都将丢失，包括通用寄存器 **DSGPR0** 和 **DSGPR1** 的内容。

寄存器 9-1: DSCON: 深度休眠控制寄存器 (1)

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

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/C-0, HS
—	—	—	—	—	—	DSBOR(2)	RELEASE
bit 7							bit 0

图注:	C = 可清零位	HS = 硬件置 1 位
R = 可读位	W = 可写位	U = 未实现位, 读为 0
-n = POR 时的值	1 = 置 1	0 = 清零 x = 未知

- bit 15 **DSEN:** 深度休眠使能位
1 = 在执行 PWRSAV #0 时进入深度休眠模式
0 = 在执行 PWRSAV #0 时进入正常休眠模式
- bit 14-2 **未实现:** 读为 0
- bit 1 **DSBOR:** 深度休眠 BOR 事件位 (2)
1 = DSBOR 先前在工作, 并且在深度休眠期间检测到 BOR 事件
0 = DSBOR 先前不工作, 或者先前在工作, 但在深度休眠期间未检测到 BOR 事件
- bit 0 **RELEASE:** I/O 引脚状态释放位
1 = 从深度休眠模式唤醒时, I/O 引脚维持在进入深度休眠模式之前的状态
0 = 将 I/O 引脚从进入深度休眠模式之前的状态释放, 允许相应的 TRIS 和 LAT 位控制它们的状态

注 1: 只有在深度休眠模式之外发生 POR 事件时, 所有寄存器位才会复位。
 2: 不同于所有其他事件, 深度休眠 BOR 事件不会导致从深度休眠模式唤醒; 它会重新激活 POR。

PIC24F16KA102 系列

寄存器 9-2: **DSWSRC: 深度休眠唤醒源寄存器** ⁽¹⁾

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0, HS
—	—	—	—	—	—	—	DSINT0
bit 15							bit 8

R/W-0, HS	U-0	U-0	R/W-0, HS	R/W-0, HS	R/W-0, HS	U-0	R/W-0, HS
DSFLT	—	—	DSWDT	DSRTCC	DSMCLR	—	DSPOR ^(2,3)
bit 7							bit 0

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

- bit 15-9 **未实现:** 读为 0
- bit 8 **DSINT0:** 电平变化中断位
1 = 电平变化中断在深度休眠期间发生
0 = 电平变化中断在深度休眠期间未发生
- bit 7 **DSFLT:** 深度休眠故障检测位
1 = 在深度休眠期间发生了故障, 并且某些深度休眠配置设置可能已被破坏
0 = 在深度休眠期间未检测到故障
- bit 6-5 **未实现:** 读为 0
- bit 4 **DSWDT:** 深度休眠看门狗定时器超时位
1 = 深度休眠看门狗定时器在深度休眠期间超时
0 = 深度休眠看门狗定时器在深度休眠期间未超时
- bit 3 **DSRTCC:** 实时时钟和日历闹钟位
1 = 实时时钟和日历在深度休眠期间触发了一次闹钟
0 = 实时时钟和日历在深度休眠期间未触发闹钟
- bit 2 **DSMCLR:** $\overline{\text{MCLR}}$ 事件位
1 = $\overline{\text{MCLR}}$ 引脚先前在工作, 并且在深度休眠期间被置为有效
0 = $\overline{\text{MCLR}}$ 引脚先前不工作, 或者先前在工作, 但在深度休眠期间未被置为有效
- bit 1 **未实现:** 读为 0
- bit 0 **DSPOR:** 上电复位事件位 ^(2,3)
1 = V_{DD} 电源 POR 电路先前在工作, 并且检测到 POR 事件
0 = V_{DD} 电源 POR 电路先前不工作, 或者先前在工作, 但未检测到 POR 事件

- 注 1: 所有寄存器位在 DSCON<DSEN> 位置 1 时清零。
- 2: 只有在深度休眠模式之外发生 POR 事件时, 所有寄存器位才会复位; 但 DSPOR 位除外, 该位在由于退出深度休眠而产生 POR 事件时不会复位。
- 3: 不同于该寄存器中的其他位, 该位可以在不处于深度休眠模式时置 1。

9.2.4 在节能指令执行期间的中断

在执行 **PWRS** 指令时产生的中断都将延迟到进入休眠或空闲模式后才起作用，并导致器件从休眠或空闲模式中唤醒。

9.3 打盹模式

通常，更改时钟速度和使用某种节能模式是降低功耗的首选策略。但在有些情况下可能不可行。例如，某些应用可能必须保持不间断的同步通信，即便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完全停止通信。

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，系统时钟以相同的时钟源和相同的速度继续工作。外设模块时钟速度保持不变，但 CPU 时钟速度降低了。保持这两个时钟域同步，可以保持外设访问 **SFR** 的能力，同时 CPU 以较慢的速度执行代码。

通过将 **DOZEN** 位 (**CLKDIV<11>**) 位置 1 使能打盹模式。外设与内核的时钟速度之比是由 **DOZE<2:0>** 位 (**CLKDIV<14:12>**) 决定的。有八种可能的配置，从 1:1 到 1:128，其中 1:1 是默认设置。

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敏感的功能（如同步通信），而 CPU 保持空闲等待事件调用中断服务程序。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 工作模式。默认情况下，中断事件对打盹模式工作没有影响。

9.4 选择性外设模块控制

空闲和打盹模式使用户可通过降低或停止 CPU 时钟显著降低功耗。即便如此，外设模块仍然保持有时钟提供，因此会有功耗。有些应用可能有这些模式无法提供的需要：即将用电资源分配给 CPU 处理，而使外设功耗最低。

PIC24F 器件通过允许有选择地禁止外设模块以降低或消除其功耗来满足这种需求。这可以通过两个控制位来实现：

- 外设使能位，一般命名为“**XXXEN**”，位于模块的主控制 **SFR** 中。
- 外设模块禁止 (**PMD**) 位，一般命名为“**XXXMD**”，位于相应的 **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。将模块的 **PMD** 位置 1 会禁止该模块的所有时钟源，从而将其功耗降至绝对最低。在该状态下，与外设相关的控制寄存器和状态寄存器也被禁止，因此写入那些寄存器不会有影响，且读取值无效。许多外设模块都有一个对应的 **PMD** 位。

相反，通过清零模块的 **XXXEN** 位以禁止其功能来禁止该模块，会保留其寄存器的读写功能。这也会降低功耗，但不如将 **PMD** 位置 1 降低得多。大多数外设模块都有一个使能位；捕捉、比较和 **RTCC** 则例外。

要实现更多可选择的节能，可在器件进入空闲模式后有选择地禁止外设模块。这可通过通用名称格式为“**XXXIDL**”的控制位实现。默认情况下，所有能在空闲模式下工作的模块都适用。在空闲模式下使用空闲模式禁止功能禁止模块，使功耗在空闲模式下得到进一步降低，从而增强了对功耗要求异常严格的应用的节能功能。

PIC24F16KA102 系列

注:

10.0 I/O 端口

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于 I/O 端口的更多信息，请参见《PIC24F 系列参考手册》的“第 12 章 带外设引脚选择 (PPS) 的 I/O 端口” (DS39711A_CN)。请注意 PIC24F16KA102 系列器件不支持外设引脚选择功能。

所有器件引脚 (VDD 和 VSS 除外) 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器输入，提高了抗噪声能力。

10.1 并行 I/O (PIO) 端口

与某个外设共用一个引脚的并行 I/O 端口通常服从于该外设。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来自外设还是相应的端口。该逻辑电路同时会阻止“环回进入” (loop through)，即一个端口的数字输出可以驱动共用同一个引脚的外设的输入。图 10-1 显示了端口是如何与其他外设复用的以及相关的 I/O 引脚。

如果外设使能，并且外设正在驱动相关引脚时，将禁止该引脚用作通用输出引脚。可以读该 I/O 引脚，但并行端口位的输出驱动器将被禁止。如果使能某外设但该外设没有驱动相应的引脚时，则该引脚可由一个端口驱动。

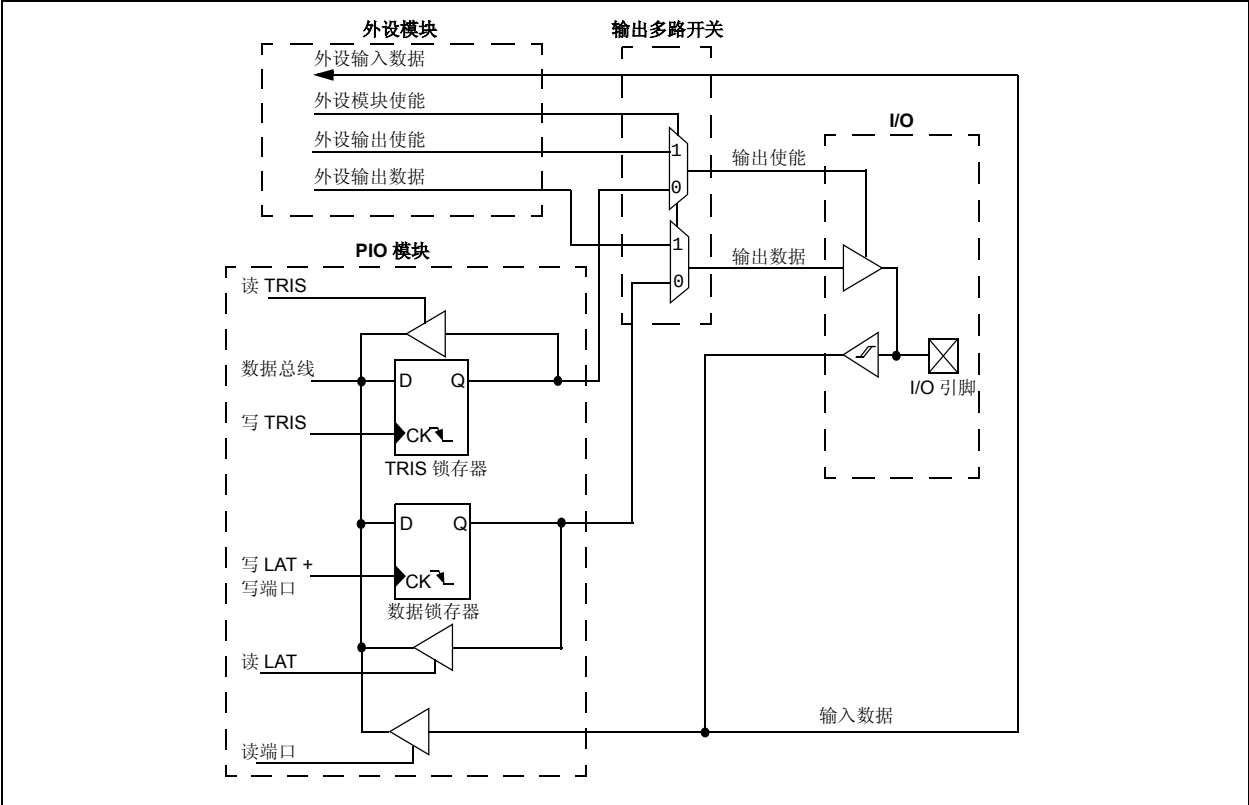
所有端口引脚都有 3 个寄存器与其作为数字 I/O 时的操作直接相关。数据方向寄存器 (TRISx) 决定引脚是输入还是输出。如果数据方向位为 1，则引脚为输入。复位后，所有端口引脚均定义为输入。读数据锁存寄存器 (LATx) 时，读到的是锁存器中的值；写锁存器时，写入的是锁存器。读端口 (PORTx) 时，读到的是端口引脚的值；而写端口引脚时，写入的是锁存器。

对于特定器件无效的任何位及其相关的数据和控制寄存器都将被禁止。这意味着对应的 LATx 和 TRISx 寄存器以及端口引脚都将读为零。

当端口引脚与另一个外设共用或与定义为仅输入的功能共用时，它将被视为专用端口，因为没有任何其他竞争的输出源。

注： 在深度休眠期间，I/O 引脚会保持原有状态。在唤醒时它们将保持该状态，直到软件复位 (RELEASE) 被清零。

图 10-1: 典型共用端口结构框图



PIC24F16KA102 系列

10.1.1 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。这是由与每个端口相关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。

允许的最大漏极开路电压与最大 V_{IH} 规范相同。

10.2 配置模拟端口引脚

AD1PCFG 和 TRIS 寄存器用于控制 A/D 端口引脚的操作。如果要将端口引脚用作模拟输入，则对应的 TRIS 位必须置 1（输入）。如果将 TRIS 位清零（输出），则数字输出电平（ V_{OH} 或 V_{OL} ）将被转换。

当读取端口寄存器时，所有配置为模拟输入通道的引脚均读为零（低电平）。对于任何定义为数字输入的引脚（包括 ANx 引脚），加在引脚上的模拟电压可能导致输入缓冲器消耗的电流超出器件规范。

10.2.1 I/O 端口写 / 读时序

端口方向改变或端口写操作与同一端口的读操作之间需要一个指令周期。这通常通过一条 NOP 指令来实现。

10.3 输入电平变化通知

I/O 端口的输入电平变化通知功能允许 PIC24F16KA102 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。即使在时钟被禁止的休眠模式下，该特性也

可检测输入状态改变。根据器件的引脚数，最多可以选择（允许）23 个外部信号（CN0 到 CN22）在输入状态发生变化时产生中断请求。

有 6 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控制位。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 / 下拉电路。上拉电路为该引脚提供拉电流，下拉电路则提供灌电流，当连接了按钮或键盘设备时，不再需要使用外部电阻。

在任意引脚上，应仅使能上拉电阻或下拉电阻，但不能同时使能它们。如果按钮或键盘连接到 V_{DD} ，则使能下拉电阻；如果连接到 V_{SS} ，则使能上拉电阻。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。使用包含每个 CN 引脚控制位的 CNPD1 和 CNPD2 寄存器可分别使能各个下拉电路。将任一控制位置 1 均可使能相应引脚的弱下拉功能。

当选择内部上拉时，引脚使用 V_{DD} 作为上拉源电压。当选择内部下拉时，引脚电压会被内部电阻下拉为 V_{SS} 。当使能内部上拉 / 下拉时，请确保没有外部上拉拉电流 / 下拉灌电流。

注： 当端口引脚被配置为数字输出时，电平变化通知引脚的上拉和下拉应始终被禁止。

例 10-1: 端口写 / 读示例

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

等价的 C 语言代码
TRISB = 0xFF00;               //Configure PORTB<15:8> as inputs and PORTB<7:0> as outputs
NOP();                         //Delay 1 cycle
if(PORTBbits.RB13 == 1)      // execute following code if PORTB pin 13 is set.
{
}
```

11.0 TIMER1

注： 本数据手册总结了PIC24F系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于定时器的更多信息，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，可作为实时时钟（RTC）的时间计数器，或作为自由运行的间隔定时器 / 计数器。**Timer1** 可在以下三种模式下工作：

- 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:

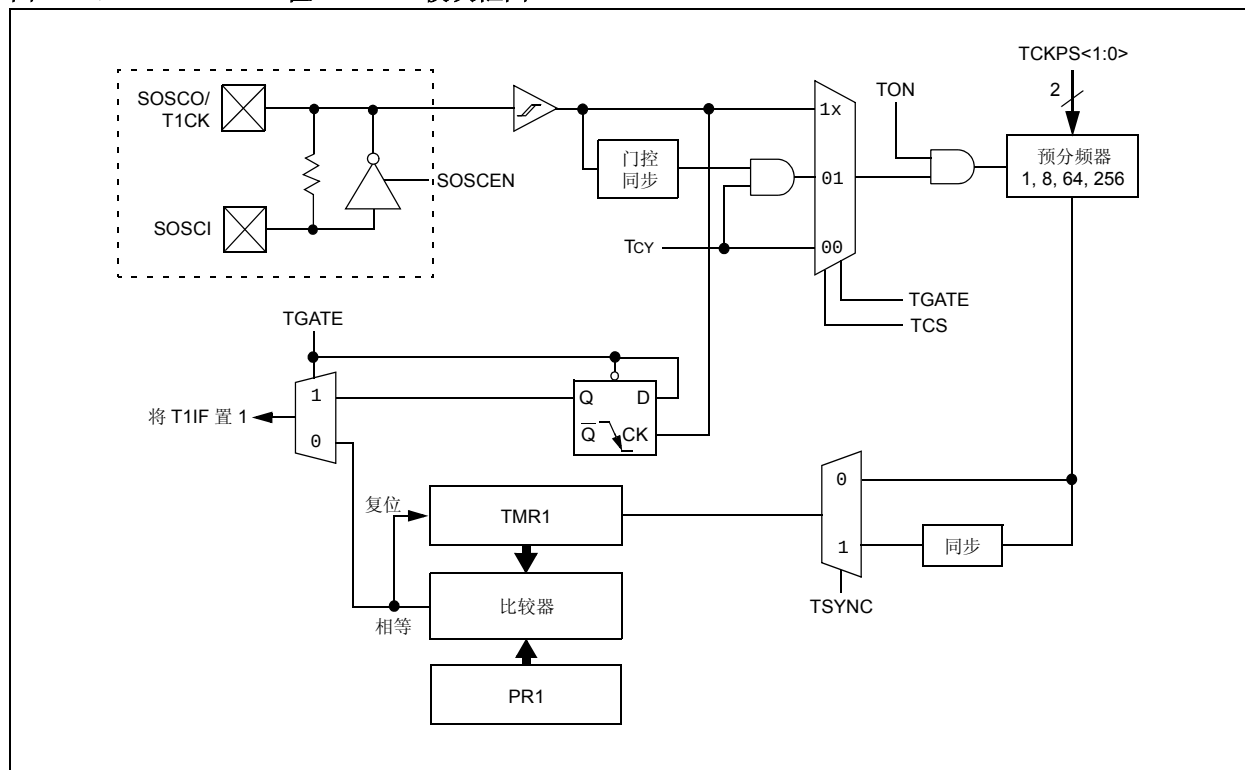
- 定时器门控操作
- 可选的预分频比设置
- 在 CPU 空闲和休眠模式期间的定时器工作
- 在 16 位周期寄存器匹配时或外部门控信号的下降沿产生中断

图 11-1 给出了 16 位 Timer1 模块的框图。

配置 Timer1 的操作:

1. 将 TON 位置 1 (= 1)。
2. 使用 TCKPS<1:0> 位选择定时器预分频比。
3. 使用 TCS 和 TGATE 位设置时钟和门控模式。
4. 将 TSYNC 位置 1 或清零来配置同步或异步操作。
5. 将定时器的周期值装入 PR1 寄存器。
6. 如果需要中断，将中断允许位 T1IE 置 1。使用优先级位 T1IP<2:0> 来设置中断优先级。

图 11-1: 16 位 TIMER1 模块框图



PIC24F16KA102 系列

寄存器 11-1: 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 = POR 时的值		1 = 置 1		0 = 清零		x = 未知	

bit 15	TON: Timer1 使能位 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 = 在空闲模式下模块继续工作
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: 该位为无关位。 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	TCKPS<1:0>: 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: 该位为无关位。
bit 1	TCS: Timer1 时钟源选择位 1 = 来自 T1CK 引脚的外部时钟 (上升沿触发计数) 0 = 内部时钟 (Fosc/2)
bit 0	未实现: 读为 0

12.0 TIMER2/3

注： 本数据手册总结了PIC24F系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于定时器的更多信息，请参见《PIC24F系列参考手册》的“第14章 定时器”（DS39704A_CN）。

Timer2/3 模块是一个 32 位定时器，也可被配置为两个具有可选工作模式的独立 16 位定时器。

作为 32 位定时器，Timer2/3 具有三种工作模式：

- 具有所有 16 位工作模式（异步计数器模式除外）的两个独立 16 位定时器（Timer2 和 Timer3）
- 单个 32 位定时器
- 单个 32 位同步计数器

Timer2/3 还支持以下功能：

- 定时器门控操作
- 可选的预分频比设置
- 空闲和休眠模式期间的定时器工作
- 在 32 位周期寄存器匹配时产生中断
- 触发 ADC 事件

两个 16 位定时器都能单独用作同步定时器或计数器。它们也提供上面所列的功能，但 ADC 事件触发功能除外（这仅由 Timer3 实现）。通过设置 T2CON 和 T3CON 寄存器中的相应位来确定工作模式和要被使能的功能。T2CON 和 T3CON 分别在寄存器 12-1 和寄存器 12-2 中作了一般介绍。

作为 32 位定时器 / 计数器工作时，Timer2 是 32 位定时器的低位字（lsw），而 Timer3 是高位字（msw）。

注： T3CON 控制位对处于 32 位工作模式的定时器无效。设置和控制只使用 T2CON 控制位。32 位定时器模块采用 Timer2 时钟和门控输入，但产生中断时会将 Timer3 中断标志位置 1。

要将 Timer2/3 配置为 32 位定时器：

1. 将 T32 位置 1（T2CON<3> = 1）。
2. 使用 TCKPS<1:0> 位为 Timer2 选择预分频比。
3. 使用 TCS 和 TGATE 位设置时钟和门控模式。
4. 装入定时器的周期值。PR3 包含值的 msw，而 PR2 包含 lsw。
5. 如果需要中断，将中断允许位 T3IE 置 1；使用中断优先级位 T3IP<2:0> 来设置中断优先级。

Timer2 控制定时器，而中断则表现为 Timer3 中断。

6. 将 TON 位置 1（= 1）。

任意时刻定时器的值被存储在寄存器对 TMR<3:2> 中。TMR3 总是包含计数值的 msw，而 TMR2 包含 lsw。

要将任一定时器配置为独立的 16 位定时器：

1. 清零 T2CON<3> 中的 T32 位。
2. 使用 TCKPS<1:0> 位选择定时器预分频比。
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入 PRx 寄存器。
5. 如果需要中断，将中断允许位 TxIE 置 1；使用中断优先级位 TxIP<2:0> 来设置中断优先级。
6. 将 TON 位置 1（TxCON<15> = 1）。

PIC24F16KA102 系列

图 12-1: TIMER2/3 (32 位) 框图

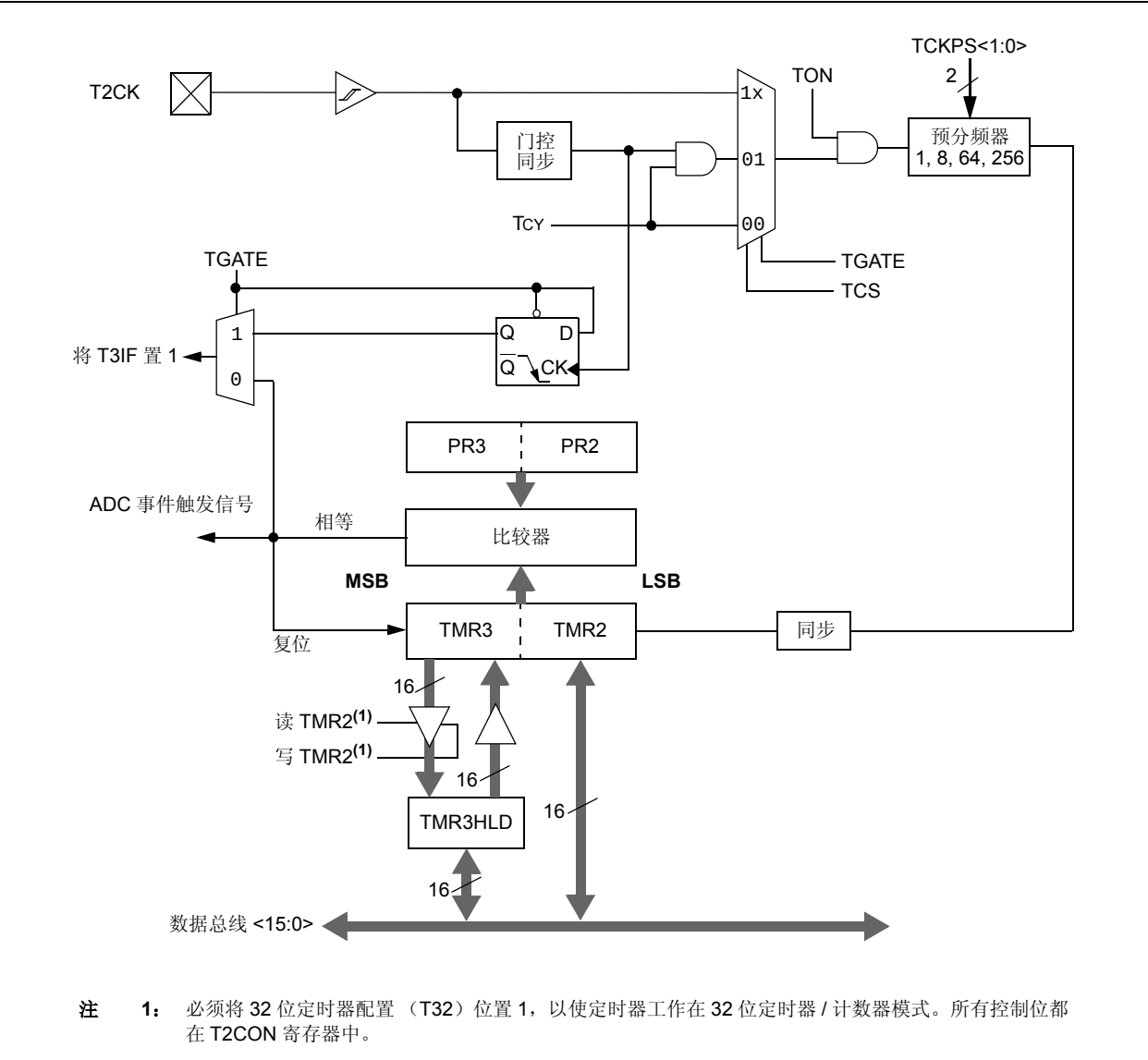


图 12-2: TIMER2（16 位同步）框图

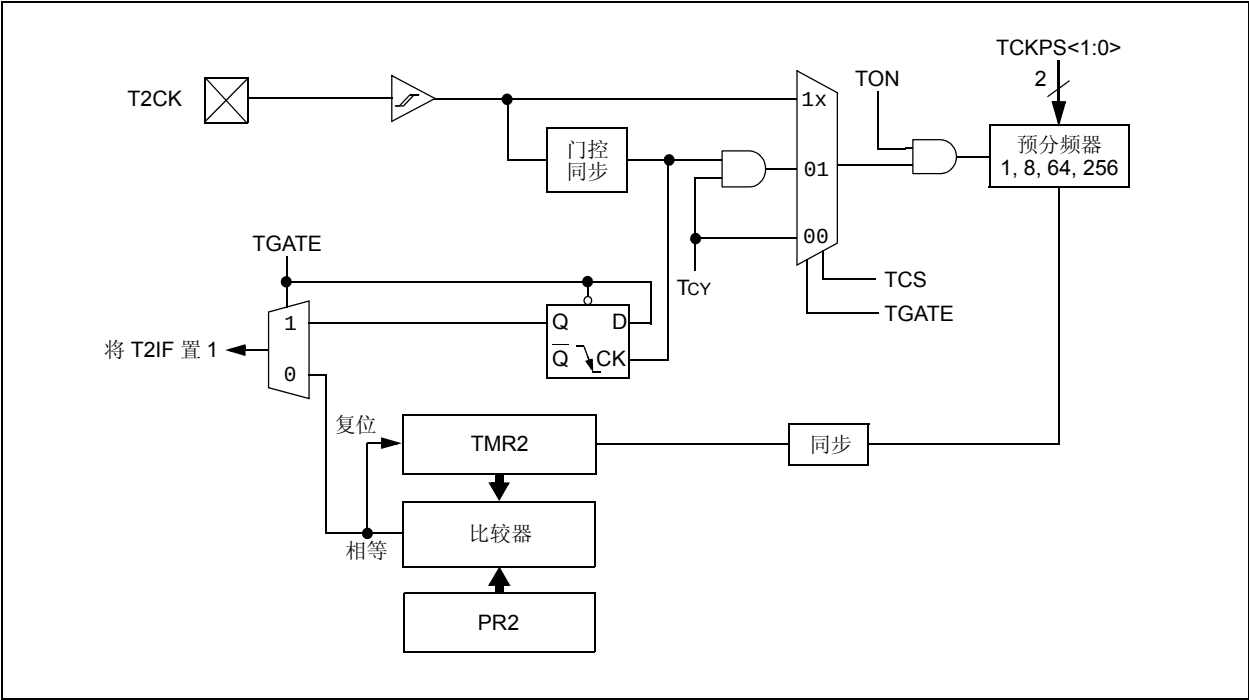
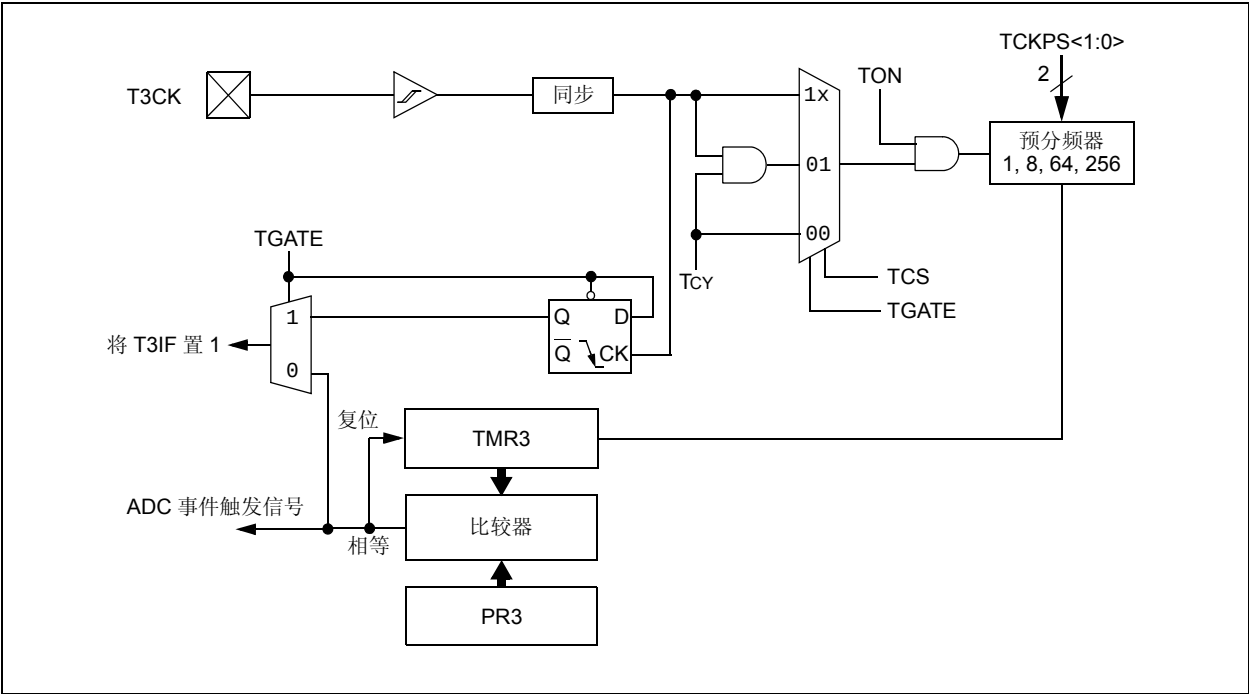


图 12-3: TIMER3（16 位同步）框图



PIC24F16KA102 系列

寄存器 12-1: T2CON: TIMER2 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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15	TON: Timer2 使能位 当 T2CON<3> = 1 时: 1 = 启动 32 位 Timer2/3 0 = 停止 32 位 Timer2/3 当 T2CON<3> = 0 时: 1 = 启动 16 位 Timer2 0 = 停止 16 位 Timer2
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 = 在空闲模式下模块继续工作
bit 12-7	未实现: 读为 0
bit 6	TGATE: Timer2 门控时间累加使能位 当 TCS = 1 时: 该位为无关位。 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	TCKPS<1:0>: 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 = Timer2 和 Timer3 形成一个 32 位定时器 0 = Timer2 和 Timer3 作为两个 16 位定时器
bit 2	未实现: 读为 0
bit 1	TCS: Timer2 时钟源选择位 1 = 来自 T2CK 引脚的外部时钟 (上升沿触发计数) 0 = 内部时钟 (Fosc/2)
bit 0	未实现: 读为 0

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

寄存器 12-2: T3CON: TIMER3 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 = POR 时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15 **TON:** Timer3 使能位 ⁽¹⁾
1 = 启动 16 位 Timer3
0 = 停止 16 位 Timer3
- bit 14 **未实现:** 读为 0
- bit 13 **TSIDL:** 空闲模式停止位 ⁽¹⁾
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 12-7 **未实现:** 读为 0
- bit 6 **TGATE:** Timer3 门控时间累加使能位 ⁽¹⁾
当 TCS = 1 时:
该位为无关位。
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 **TCKPS<1:0>:** Timer3 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:** Timer3 时钟源选择位 ⁽¹⁾
1 = 来自 T3CK 引脚的外部时钟 (上升沿计数)
0 = 内部时钟 (Fosc/2)
- bit 0 **未实现:** 读为 0

注 1: 当使能 32 位工作 (T2CON<3> = 1) 时, 这些位对 Timer3 的工作没有影响; 所有定时器功能都通过 T2CON 进行设置。

PIC24F16KA102 系列

注:

13.0 输入捕捉

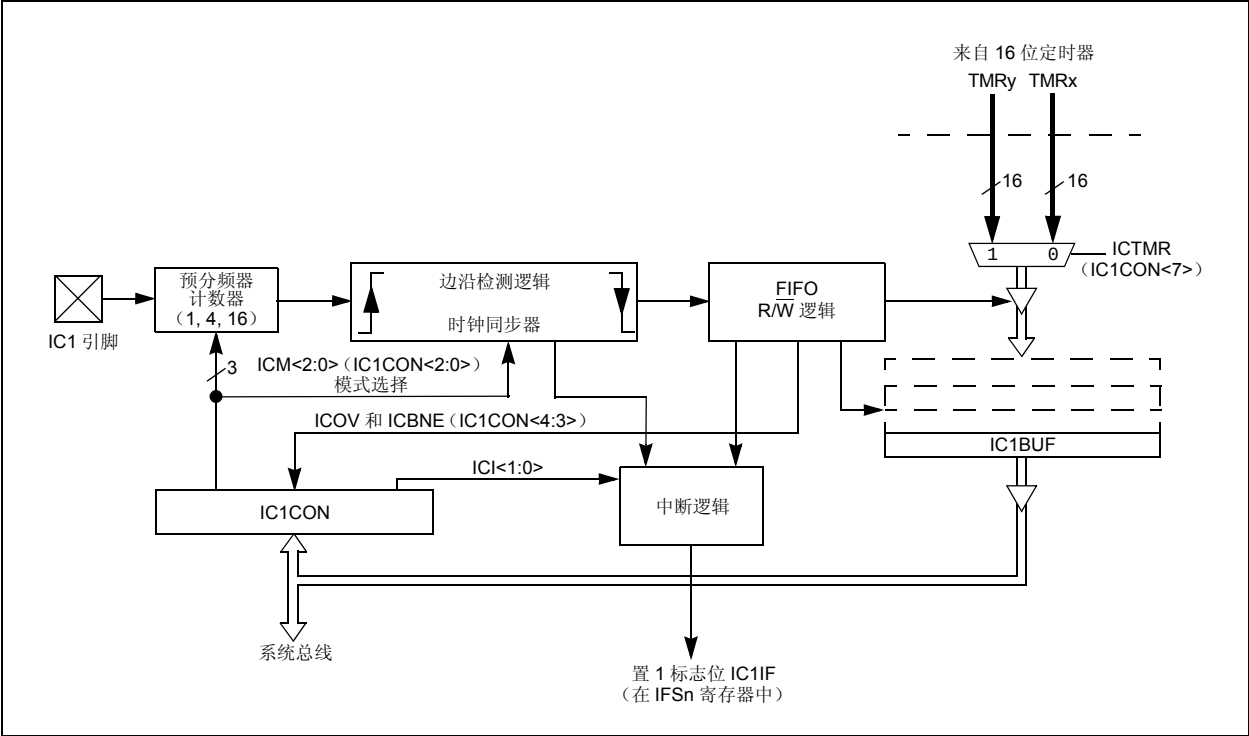
注： 本数据手册总结了PIC24F系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于输入捕捉的更多信息，请参见《PIC24F系列参考手册》的“第15章 输入捕捉”（DS39701A_CN）。

输入捕捉模块用于在输入引脚上有事件发生时，捕捉来自两个可选时基之一的定时器值。

输入捕捉功能在需要进行频率（时间周期）和脉冲测量的应用中很有用。图 13-1 给出了输入捕捉模块的简化框图。

- PIC24F16KA102 系列器件具有一路输入捕捉通道。输入捕捉模块有多种工作模式，可通过 IC1CON 寄存器选择。这些工作模式包括：
- 每当 IC1 引脚上的输入信号出现下降沿时捕捉定时器值
 - 每当 IC1 引脚上的输入信号出现上升沿时捕捉定时器值
 - IC1 引脚上的输入信号每出现 4 个上升沿捕捉一次定时器值
 - IC1 引脚上的输入信号每出现 16 个上升沿捕捉一次定时器值
 - 每当 IC1 引脚上的输入信号出现上升沿和下降沿时都捕捉定时器值
 - 当 CPU 在休眠和空闲模式时通过捕捉引脚上的信号将器件唤醒
- 输入捕捉模块具有 4 级深 FIFO 缓冲区。用户可以选择产生 CPU 中断所需的捕捉事件数。

图 13-1: 输入捕捉框图



PIC24F16KA102 系列

13.1 输入捕捉寄存器

寄存器 13-1: IC1CON: 输入捕捉 1 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-0, HC	R/W-0	R/W-0	R/W-0
ICTMR	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0
bit 7							bit 0

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

- bit 15-14 **未实现:** 读为 0
- bit 13 **ICSIDL:** 输入捕捉 1 模块在空闲模式下停止的控制位
1 = 在 CPU 空闲模式下输入捕捉模块将停止工作
0 = 在 CPU 空闲模式下输入捕捉模块将继续工作
- bit 12-8 **未实现:** 读为 0
- bit 7 **ICTMR:** 输入捕捉 1 定时器选择位
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 **ICI<1:0>:** 选择每次发生中断时捕捉的次数的位
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 = 每次捕捉事件中断一次
- bit 4 **ICOV:** 输入捕捉 1 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 1 缓冲区空状态位 (只读)
1 = 输入捕捉缓冲区非空, 至少可以再读一个捕捉值
0 = 输入捕捉缓冲区为空
- bit 2-0 **ICM<2:0>:** 输入捕捉 1 模式选择位
111 = 当器件处于休眠或空闲模式时, 输入捕捉仅用作中断引脚 (只检测上升沿, 所有其他控制位都不适用)
110 = 未使用 (模块被禁止)
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, 每个上升沿捕捉一次
010 = 捕捉模式, 每个下降沿捕捉一次
001 = 捕捉模式, 每个边沿 (上升沿和下降沿) 捕捉一次——ICI<1:0> 位不控制该模式下的中断产生
000 = 输入捕捉模块关闭

14.0 输出比较

注： 本数据手册总结了PIC24F系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于输出比较的更多信息，请参见《PIC24F系列参考手册》的“第16章 输出比较”（DS39706A_CN）。

14.1 设置产生单个输出脉冲

当 OCM 控制位（OC1CON<2:0>）被设置为 100 时，所选的输出比较通道将 OC1 引脚初始化为低电平状态并产生单输出脉冲。

若要产生单输出脉冲，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，但这并不是对模块工作的要求）：

1. 确定指令时钟周期时间。考虑定时器源的外部时钟频率（如果使用）和定时器预分频比的设置。
2. 计算从 TMRy 起始值（0000h）到输出脉冲的上升沿所需的时间。
3. 根据所需的脉冲宽度和到脉冲上升沿的时间计算出出现脉冲下降沿的时间。
4. 将以上步骤 2 和步骤 3 中计算出的值分别写入输出比较 1 寄存器 OC1R 和输出比较 1 辅助寄存器 OC1RS。
5. 将定时器周期寄存器 PRy 的值设置为等于或大于输出比较 1 辅助寄存器 OC1RS 中的值。
6. 将 OCM 位设置为 100，并将 OCTSEL（OC1CON<3>）位设置为要求的定时器源。此时 OC1 引脚状态被驱动为低电平。
7. 将 TON（TyCON<15>）位设为 1 以使比较时基计数。
8. 在 TMRy 和 OC1R 第一次匹配时，OC1 引脚将被驱动为高电平。
9. 当递增定时器 TMRy 和输出比较 1 辅助寄存器 OC1RS 发生匹配时，在 OC1 引脚上驱动脉冲的第二个边沿（即下降沿）。OC1 引脚上不会驱动输出额外的脉冲，OC1 引脚将保持为低电平。第二次比较匹配事件会导致 OC1IF 中断标志位置 1。如果已通过将 OC1IE 位置 1 允许中断，将产生中断。关于外设中断的更多信息，请参见第 7.0 节“中断控制器”。

10. 要发出另一个单脉冲输出，在需要的情况下要更改定时器和比较寄存器的设置，然后执行写操作，将 OCM 位设置为 100。不需要禁止和重新使能定时器以及将 TMRy 寄存器清零，但这样做可能对确定一个已知事件时间边界发出的脉冲有好处。

在输出脉冲下降沿后不一定要禁止输出比较模块。重写 OC1CON 寄存器的值可以发出另一个脉冲。

14.2 设置产生连续输出脉冲

当 OCM 控制位（OC1CON<2:0>）被设置为 101 时，所选的输出比较通道将 OC1 引脚初始化为低电平状态，并在每次比较匹配事件发生时在该引脚输出脉冲。

用户若要将模块配置为产生连续的输出脉冲流，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，但这并不是对模块工作的要求）：

1. 确定指令时钟周期时间。考虑定时器源的外部时钟频率（如果使用）和定时器预分频比的设置。
2. 计算从 TMRy 起始值（0000h）到输出脉冲的上升沿所需的时间。
3. 根据所需的脉冲宽度和到脉冲上升沿的时间计算出出现脉冲下降沿的时间。
4. 将以上步骤 2 和步骤 3 中计算出的值分别写入输出比较 1 寄存器 OC1R 和输出比较 1 辅助寄存器 OC1RS。
5. 将定时器周期寄存器 PRy 的值设置为等于或大于 OC1RS 中的值。
6. 将 OCM 位设置为 101，并将 OCTSEL 位设置为要求的定时器源。此时 OC1 引脚状态被驱动为低电平。
7. 通过将 TON（TyCON<15>）位设为 1 使能比较时基。
8. 在 TMRy 和 OC1R 第一次匹配时，OC1 引脚将被驱动为高电平。
9. 当比较时基 TMRy 和 OC1RS 发生匹配时，在 OC1 引脚上驱动脉冲的第二个边沿（即下降沿）。
10. 第二次比较匹配事件会导致 OC1IF 中断标志位置 1。
11. 当比较时基和相应的定时器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，可无限地产生连续脉冲流。OC1IF 标志位在每次 OC1RS 与 TMRy 的比较匹配事件发生时置 1。

PIC24F16KA102 系列

14.3 脉宽调制 (PWM) 模式

当将输出比较模块配置为 PWM 操作时，需要遵循以下步骤：

1. 通过写所选的定时器周期寄存器 (PRy) 设置 PWM 周期。
2. 通过写 OC1RS 寄存器设置 PWM 占空比。
3. 向 OC1R 寄存器中写入初始占空比。
4. 如果需要的话，允许定时器和输出比较模块的中断。如果要使用 PWM 故障引脚，则必须设置输出比较中断。
5. 通过写输出比较模式位 OCM<2:0> (OC1CON<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： 在第一次使能输出比较模块之前，必须先初始化 OC1R 寄存器。当模块工作于 PWM 模式时，OC1R 寄存器变为只读的占空比寄存器。OC1R 中保存的值成为第一个 PWM 周期的 PWM 占空比。直到时基周期匹配发生，输出比较 1 辅助寄存器 OC1RS 的内容才会被传输到 OC1R。

14.3.1 PWM 周期

PWM 周期可通过写入 PRy (定时器周期寄存器) 来指定。可使用公式 14-1 计算 PWM 周期。

公式 14-1: 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: 基于 Tcy = 2 • TOSC，打盹模式和 PLL 被禁止。

注： 如果 PRy 的值为 N，则会使 PWM 周期为 N + 1 个时基计数周期。例如，如果写入 PRy 寄存器的值为 7，则将产生由 8 个时基周期组成的 PWM 周期。

14.3.2 PWM 占空比

PWM 占空比是通过写 OC1RS 寄存器指定的。可以在任何时候写 OC1RS 寄存器，但是在 PRy 和 TMRy 发生匹配 (即周期结束) 前占空比值不会被锁存到 OC1R 中。这可以为 PWM 占空比提供双重缓冲，对于 PWM 的无毛刺操作是极其重要的。在 PWM 模式下，OC1R 是只读寄存器。

PWM 占空比有一些重要的边界参数，包括：

- 如果输出比较 1 寄存器 OC1R 中装入 0000h，则 OC1 引脚将保持低电平 (占空比为 0%)。
- 如果 OC1R 大于 PRy (定时器周期寄存器)，则引脚将保持高电平 (占空比为 100%)。
- 如果 OC1R 等于 PRy，则 OC1 引脚在一个时基计数周期内为低电平，而在其余所有的计数周期内均为高电平。

请参见例 14-1 了解 PWM 模式时序的详细信息。表 14-1 给出了器件工作速度为 10 MIPS 时的 PWM 频率和分辨率示例。

公式 14-2: 计算最大 PWM 分辨率⁽¹⁾

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

注 1: 基于 Fcy = Fosc/2，打盹模式和 PLL 被禁止。

例 14-1: PWM 周期和占空比计算⁽¹⁾

1. 确定对应于所需的 PWM 频率 52.08 kHz 的定时器周期寄存器值，其中 $F_{osc} = 8 \text{ MHz}$ 、PLL（32 MHz 器件时钟速率）和 Timer2 预分频比设置为 1:1。

$$T_{cy} = 2 * T_{osc} = 62.5 \text{ ns}$$

$$\text{PWM 周期} = 1/\text{PWM 频率} = 1/52.08 \text{ kHz} = 19.2 \mu\text{s}$$

$$\text{PWM 周期} = (\text{PR2} + 1) * T_{cy} * (\text{Timer 2 预分频值})$$

$$19.2 \mu\text{s} = (\text{PR2} + 1) * 62.5 \text{ ns} * 1$$

$$\text{PR2} = 306$$

2. 在 PWM 频率为 52.08 kHz 且器件的时钟速率为 32 MHz 时，计算占空比的最大分辨率。

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

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

$$= 8.3 \text{ 位}$$

注 1: 基于 $T_{cy} = 2 * T_{osc}$ ，打盹模式和 PLL 被禁止。

表 14-1: 4 MIPS ($F_{cy} = 4 \text{ MHz}$) 时的 PWM 频率和分辨率示例⁽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
周期寄存器的值	FFFFh	FFFFh	7FFFh	0FFFh	03FFh	007Fh	001Fh
分辨率 (位)	16	16	15	12	10	7	5

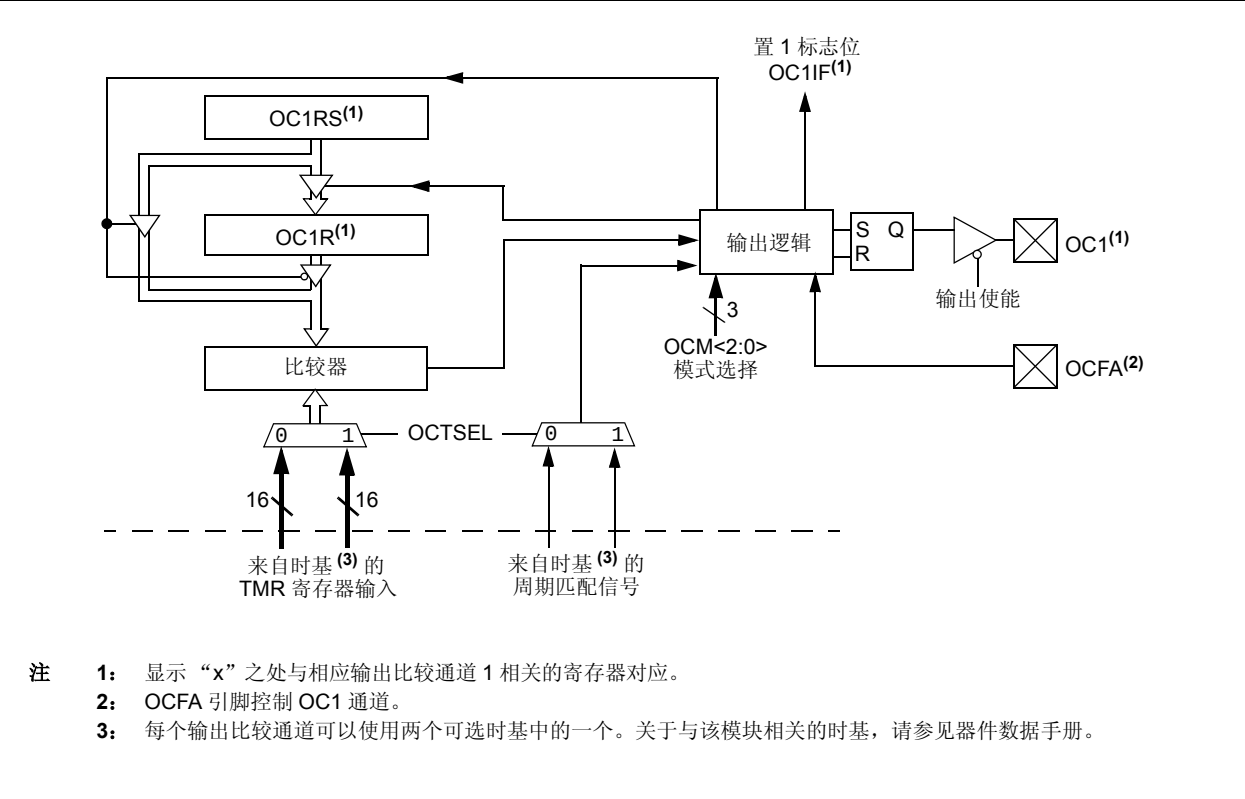
注 1: 基于 $F_{cy} = F_{osc}/2$ ，打盹模式和 PLL 被禁止。

表 14-2: 16 MIPS ($F_{cy} = 16 \text{ MHz}$) 时的 PWM 频率和分辨率示例⁽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
周期寄存器的值	FFFFh	FFFFh	7FFFh	0FFFh	03FFh	007Fh	001Fh
分辨率 (位)	16	16	15	12	10	7	5

注 1: 基于 $F_{cy} = F_{osc}/2$ ，打盹模式和 PLL 被禁止。

图 14-1: 输出比较模块框图



14.4 输出比较寄存器

寄存器 14-1: OC1CON: 输出比较 1 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	OCM2	OCM1	OCM0
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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15-14 未实现: 读为 0
- bit 13 **OCSIDL**: 在空闲模式下停止输出比较 1 控制位
 1 = 输出比较 1 将在 CPU 空闲模式下停止工作
 0 = 输出比较 1 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 = 发生了 PWM 故障条件 (仅可用硬件清零)
 0 = 未发生 PWM 故障条件 (仅当 OCM<2:0> = 111 时, 才使用该位)
- bit 3 **OCTSEL**: 输出比较 1 定时器选择位
 1 = Timer3 是输出比较 1 的时钟源
 0 = Timer2 是输出比较 1 的时钟源
 关于输出比较模块可用的特定时基, 请参见器件数据手册。
- bit 2-0 **OCM<2:0>**: 输出比较 1 模式选择位
 111 = OC1 处于 PWM 模式, 故障引脚: 使能 OCF1⁽¹⁾
 110 = OC1 处于 PWM 模式, 故障引脚: 禁止 OCF1⁽¹⁾
 101 = 初始化 OC1 引脚为低电平, 在 OC1 引脚上产生连续输出脉冲
 100 = 初始化 OC1 引脚为低电平, 在 OC1 引脚上产生单个输出脉冲
 011 = 比较事件使 OC1 引脚的电平翻转
 010 = 初始化 OC1 引脚为高电平, 比较事件强制 OC1 引脚为低电平
 001 = 初始化 OC1 引脚为低电平, 比较事件强制 OC1 引脚为高电平
 000 = 禁止输出比较通道

注 1: OCFA 引脚控制 OC1 通道。

PIC24F16KA102 系列

寄存器 14-2: **PADCFG1: 焊盘配置控制寄存器**

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	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	SMBUSDEL ⁽³⁾	OC1TRIS	RTSECSEL1 ^(1,4)	RTSECSEL0 ^(1,4)	—
bit 7							bit 0

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

- bit 15-5 **未实现:** 读为 0
- bit 3 **OC1TRIS:** OC1 输出三态选择位
1 = 引脚上的 OC1 输出将无效; OCPWM1 仍然用于内部触发
0 = 引脚上的 OC1 输出将有效, 基于 OCPWM1 模块的设置
- bit 0 **未实现:** 读为 0

- 注** **1:** 要使能实际 RTCC 输出, RTCOE (RCFGCAL) 位必须置 1。
2: 要使能实际 OC1 输出, 必须使能 OCPWM1 模块。
3: bit 4 如第 16.0 节 “I2C ±” 中所述。
4: bit 2 和 bit 1 如第 18.0 节 “实时时钟和日历 (RTCC)” 中所述。

15.0 串行外设接口 (SPI)

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于串行外设接口的更多信息，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) 模块是用于同其他外设或单片机进行通信的同步串行接口。这些外设器件可以是串行数据 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。

该模块支持在两种缓冲模式下工作。在标准模式下，数据通过单个串行缓冲区移动。在增强型缓冲模式下，数据通过一个 8 级深 FIFO 缓冲区移动。

注： 在标准或增强型缓冲模式下，都不要在 SPI1BUF 寄存器上执行读 - 修改 - 写操作 (如针对位的指令)。

工作于主或从模式时，模块还支持基本的帧 SPI 协议。共支持 4 种帧 SPI 配置。

SPI 串行接口由 4 个引脚组成：

- SDI1: 串行数据输入
- SDO1: 串行数据输出
- SCK1: 移位时钟输入或输出
- SS1: 低电平有效从选择或帧同步 I/O 脉冲

SPI 模块可以配置为使用 2、3 或 4 个引脚工作。在 3 引脚模式下，SS1 不使用。在 2 引脚模式下，SDO1 和 SS1 均不使用。

标准和增强型缓冲模式下的模块框图如图15-1和图15-2所示。

PIC24F16KA102 系列器件为每个器件提供了一个 SPI 模块。

注： 在本章中，SPI 模块被称为 SPI1。特殊功能寄存器 (SFR) 也使用类似的符号表示。例如，SPI1CON1 或 SPI1CON2 指 SPI1 模块的控制寄存器。

要将 SPI 模块设置为工作于标准主模式：

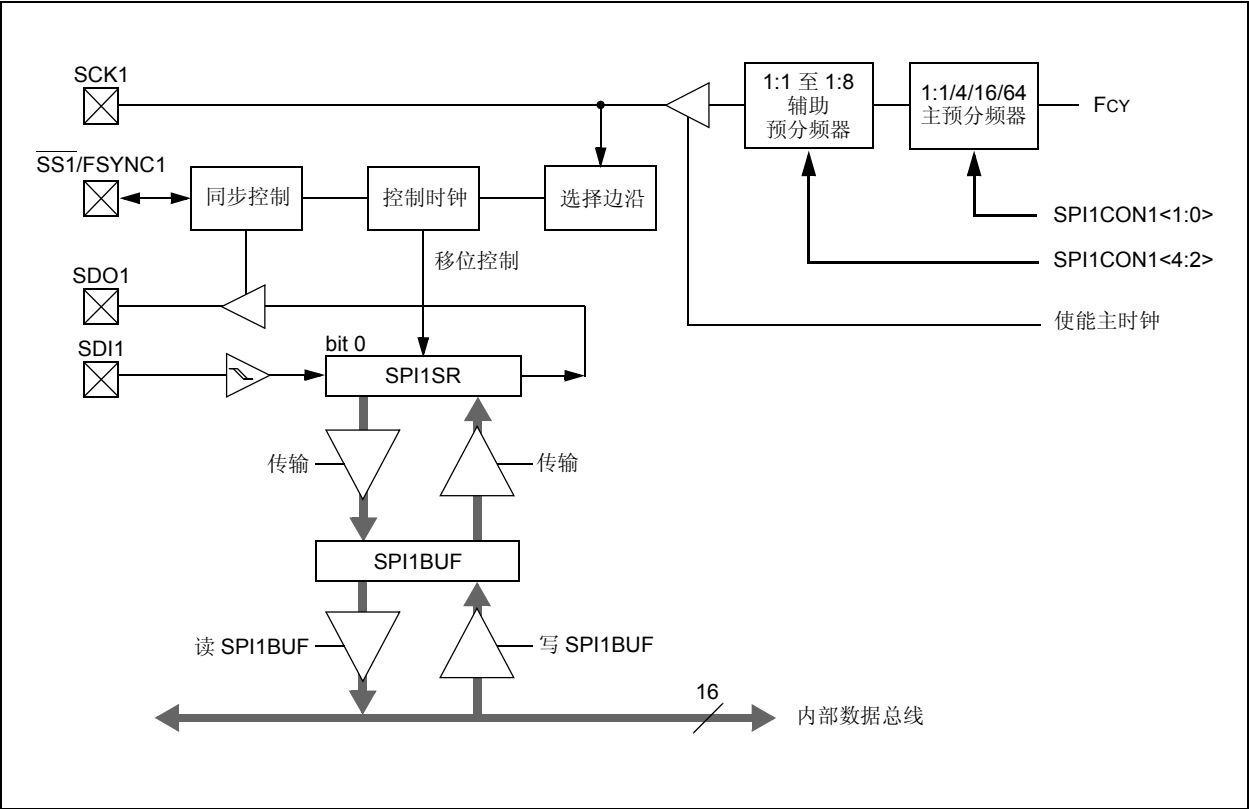
1. 如果使用中断：
 - a) 将 IFS0 寄存器中的相应 SPI1IF 位清零。
 - b) 将 IEC0 寄存器中的相应 SPI1IE 位置 1。
 - c) 通过写 IPC2 寄存器中的相应 SPI1IP 位来设置中断优先级。
2. 将所需的设置写入 SPI1CON1 和 SPI1CON2 寄存器，且 MSTEN 位 (SPI1CON1<5>) = 1。
3. 将 SPIROV 位 (SPI1STAT<6>) 清零。
4. 通过将 SPIEN 位 (SPI1STAT<15>) 置 1 使能 SPI 工作。
5. 将待发送数据写入 SPI1BUF 寄存器。发送 (和接收) 在数据写入 SPI1BUF 寄存器时立即开始。

要将 SPI 模块设置为工作于标准从模式：

1. 将 SPI1BUF 寄存器清零。
2. 如果使用中断：
 - a) 将 IFS0 寄存器中的相应 SPI1IF 位清零。
 - b) 将 IEC0 寄存器中的相应 SPI1IE 位置 1。
 - c) 通过写 IPC2 寄存器中的相应 SPI1IP 位来设置中断优先级。
3. 将所需的设置写入 SPI1CON1 和 SPI1CON2 寄存器，且 MSTEN 位 (SPI1CON1<5>) = 0。
4. 将 SMP 位清零。
5. 如果 CKE 位置 1，则必须将 SSEN 位 (SPI1CON1<7>) 置 1 来使能 SS1 引脚。
6. 将 SPIROV 位 (SPI1STAT<6>) 清零。
7. 通过将 SPIEN 位 (SPI1STAT<15>) 置 1 使能 SPI 工作。

PIC24F16KA102 系列

图 15-1: SPI1 模块框图 (标准缓冲模式)



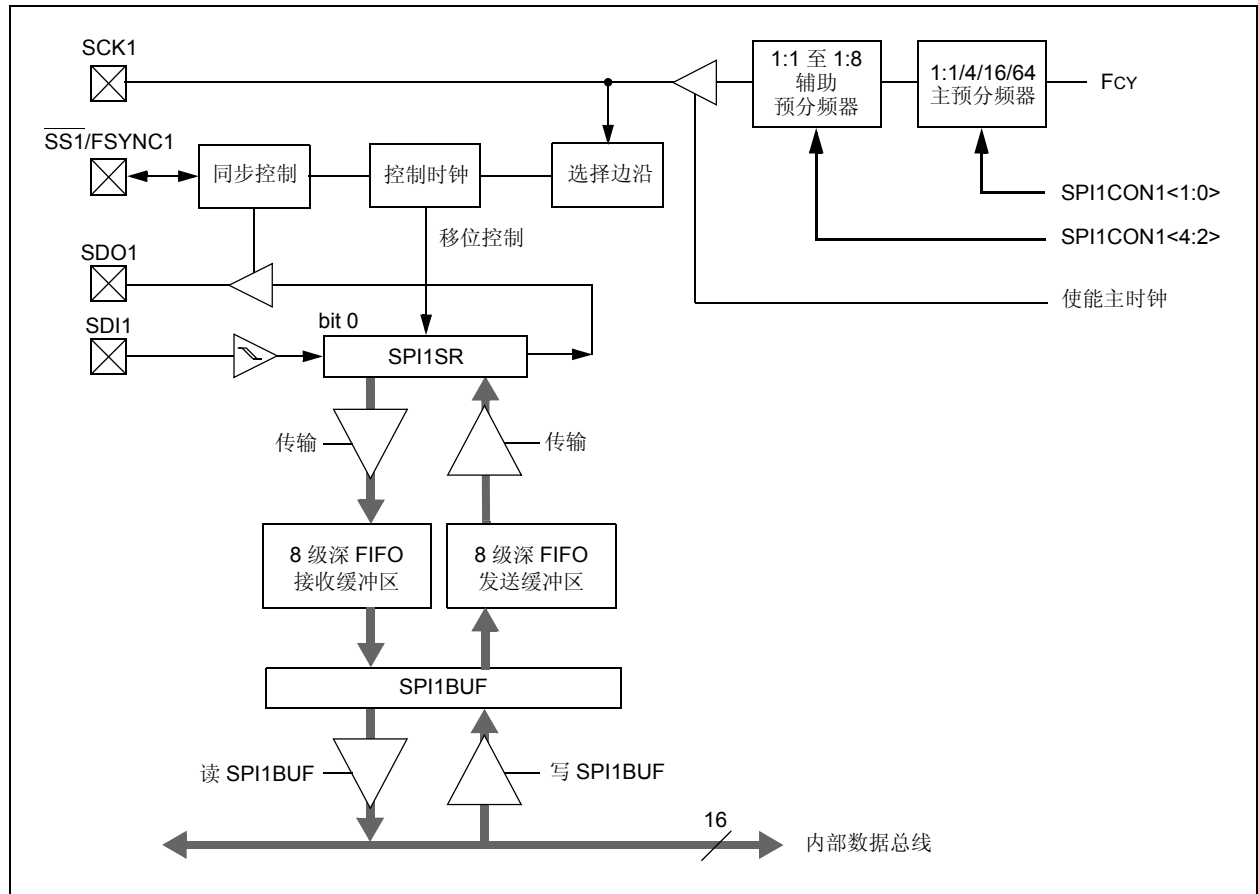
要将 SPI 模块设置为工作于增强型缓冲主（Enhanced Buffer Master，EBM）模式：

1. 如果使用中断：
 - a) 将 IFS0 寄存器中的相应 SPI1IF 位清零。
 - b) 将 IEC0 寄存器中的相应 SPI1IE 位置 1。
 - c) 写入 IPC 寄存器中的相应 SPI1IP 位。
2. 将所需的设置写入 SPI1CON1 和 SPI1CON2 寄存器，且 MSTEN 位（SPI1CON1<5>）= 1。
3. 将 SPIROV 位（SPI1STAT<6>）清零。
4. 通过将 SPIBEN 位（SPI1CON2<0>）置 1 选择增强型缓冲模式。
5. 通过将 SPIEN 位（SPI1STAT<15>）置 1 使能 SPI 工作。
6. 将待发送数据写入 SPI1BUF 寄存器。发送（和接收）在数据写入 SPI1BUF 寄存器时立即开始。

要将 SPI 模块设置为工作于增强型缓冲从模式：

1. 将 SPI1BUF 寄存器清零。
2. 如果使用中断：
 - a) 将 IFS0 寄存器中的相应 SPI1IF 位清零。
 - b) 将 IEC0 寄存器中的相应 SPI1IE 位置 1。
 - c) 通过写 IPC2 寄存器中的相应 SPI1IP 位来设置中断优先级。
3. 将所需的设置写入 SPI1CON1 和 SPI1CON2 寄存器，且 MSTEN 位（SPI1CON1<5>）= 0。
4. 将 SMP 位清零。
5. 如果 CKE 位置 1，则必须将 SS1EN 位置 1 来使能 SS1 引脚。
6. 将 SPIROV 位（SPI1STAT<6>）清零。
7. 通过将 SPIBEN 位（SPI1CON2<0>）置 1 选择增强型缓冲模式。
8. 通过将 SPIEN 位（SPI1STAT<15>）置 1 使能 SPI 工作。

图 15-2: SPI1 模块框图（增强型缓冲模式）



PIC24F16KA102 系列

寄存器 15-1: SPI1STAT: SPI1 状态和控制寄存器

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

R-0,HSC	R/C-0, HS	R/W-0, HSC	R/W-0	R/W-0	R/W-0	R-0, HSC	R-0, HSC
SRMPT	SPIROV	SR1MPT	SISEL2	SISEL1	SISEL0	SPITBF	SPIRBF
bit 7					bit 0		

图注:	HS = 硬件置 1 位	HSC = 硬件置 1/ 清零位	C = 可清零位
R = 可读位	W = 可写位	U = 未实现位, 读为 0	
-n = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15 **SPIEN:** SPI1 使能位
1 = 使能模块并将 SCK1、SDO1、SDI1 和 $\overline{SS1}$ 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时, 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 **SPIBEC<2:0>:** SPI1 缓冲区计数位 (在增强型缓冲模式下有效)
主模式:
SPI 传输等待的数目。
从模式:
SPI 传输未读的数目。
- bit 7 **SRMPT:** 移位寄存器 (SPI1SR) 空位 (在增强型缓冲模式下有效)
1 = SPI1 移位寄存器为空, 等待发送或接收
0 = SPI1 移位寄存器非空
- bit 6 **SPIROV:** 接收溢出标志位
1 = 一个新字节 / 字已完全接收并丢弃。用户软件没有读先前保存在 SPI1BUF 寄存器中的数据。
0 = 未发生溢出
- bit 5 **SR1MPT:** 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 **SISEL<2:0>:** SPI1 缓冲区中断模式位 (在增强型缓冲模式下有效)
111 = 当 SPI1 发送缓冲区已满时产生中断 (SPITBF 位置 1)
110 = 当最后一位移入 SPI1SR 时产生中断; 此时发送 FIFO 为空
101 = 当最后一位移出 SPI1SR 时产生中断; 此时发送完成
100 = 当一个数据字节移入 SPI1SR 时产生中断; 此时发送 FIFO 至少有一个空位
011 = 当 SPI1 接收缓冲区已满时产生中断 (SPIRBF 位置 1)
010 = 当 SPI1 接收缓冲区为 3/4 满或更满时产生中断
001 = 当接收缓冲区中有数据时产生中断 (SRMPT 位置 1)
000 = 当接收缓冲区中的最后数据被读取时产生中断; 此时缓冲区为空 (SR1MPT 位置 1)

寄存器 15-1: SPI1STAT: SPI1 状态和控制寄存器 (续)

bit 1 **SPITBF:** SPI1 发送缓冲区满状态位

1 = 发送尚未开始, SPI1TXB 为满

0 = 发送已开始, SPI1TXB 为空

在标准缓冲模式下:

当 CPU 通过写 SPI1BUF 存储单元装入 SPI1TXB 时, 该位由硬件自动置 1。

当 SPI1 模块将数据从 SPI1TXB 传输到 SPI1SR 时, 该位由硬件自动清零。

在增强型缓冲模式下:

当 CPU 通过写 SPI1BUF 存储单元装入最后的可用缓冲单元时, 该位由硬件自动置 1。

当有缓冲单元可用于 CPU 写操作时, 该位由硬件自动清零。

bit 0 **SPIRBF:** SPI1 接收缓冲区满状态位

1 = 接收完成, SPI1RXB 为满

0 = 接收未完成, SPI1RXB 为空

在标准缓冲模式下:

当 SPI1 将数据从 SPI1SR 传输到 SPI1RXB 时, 该位由硬件自动置 1。

当内核通过读 SPI1BUF 存储单元读 SPI1RXB 时, 该位由硬件自动清零。

在增强型缓冲模式下:

当 SPI1 通过将数据从 SPI1SR 传输到缓冲区填充最后未读的缓冲单元时, 该位由硬件自动置 1。

当有缓冲单元可用于从 SPI1SR 进行传输时, 该位由硬件自动清零。

PIC24F16KA102 系列

寄存器 15-2: SPI1CON1: SPI1 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 = POR 时的值		1 = 置 1		0 = 清零		x = 未知	

bit 15-13	未实现: 读为 0
bit 12	DISSCK: 禁止 SCK1 引脚位 (仅限 SPI 主模式) 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: 禁止 SDO1 引脚位 1 = 模块不使用 SDO1 引脚; 引脚用作 I/O 0 = SDO1 引脚由模块控制
bit 10	MODE16: 字 / 字节通信选择位 1 = 采用字宽 (16 位) 通信 0 = 采用字节宽 (8 位) 通信
bit 9	SMP: SPI1 数据输入采样阶段位 <u>主模式:</u> 1 = 在数据输出时间的末端采样输入数据 0 = 在数据输出时间的中间采样输入数据 <u>从模式:</u> 当在从模式下使用 SPI1 时, 必须将 SMP 清零。
bit 8	CKE: SPI1 时钟边沿选择位 ⁽¹⁾ 1 = 串行输出数据在时钟从工作状态转变为空闲状态时变化 (见 bit 6) 0 = 串行输出数据在时钟从空闲状态转变为工作状态时变化 (见 bit 6)
bit 7	SSEN: 从选择使能位 (从模式) 1 = $\overline{SS1}$ 引脚用于从模式 0 = 模块不使用 $\overline{SS1}$ 引脚; 引脚由端口功能控制
bit 6	CKP: 时钟极性选择位 1 = 空闲状态时时钟信号为高电平; 工作状态时为低电平 0 = 空闲状态时时钟信号为低电平; 工作状态时为高电平
bit 5	MSTEN: 主模式使能位 1 = 主模式 0 = 从模式
bit 4-2	SPRE<2:0>: 辅助预分频比位 (主模式) 111 = 辅助预分频比 1:1 110 = 辅助预分频比 2:1 . . . 000 = 辅助预分频比 8:1

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

寄存器 15-2: SPI1CON1: SPI1 控制寄存器 1 (续)

bit 1-0 **PPRE<1:0>**: 主预分频比位 (主模式)
 11 = 主预分频比 1:1
 10 = 主预分频比 4:1
 01 = 主预分频比 16:1
 00 = 主预分频比 64:1

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

寄存器 15-3: SPI1CON2: SPI1 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 = POR 时的值 1 = 置 1 0 = 清零 x = 未知

bit 15 **FRMEN**: 帧 SPI1 支持位
 1 = 使能帧 SPI1 支持
 0 = 禁止帧 SPI1 支持

bit 14 **SPIFSD**: $\overline{SS1}$ 引脚帧同步脉冲方向控制位
 1 = 帧同步脉冲输入 (从器件)
 0 = 帧同步脉冲输出 (主器件)

bit 13 **SPIFPOL**: 帧同步脉冲极性位 (仅限帧模式)
 1 = 帧同步脉冲为高电平有效
 0 = 帧同步脉冲为低电平有效

bit 12-2 **未实现**: 读为 0

bit 1 **SPIFE**: 帧同步脉冲边沿选择位
 1 = 帧同步脉冲与第一个位时钟一致
 0 = 帧同步脉冲比第一个位时钟提前

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

PIC24F16KA102 系列

公式 15-1: 器件工作频率和 SPI 时钟速度之间的关系⁽¹⁾

$F_{SCK} = \frac{F_{CY}}{\text{主预分频比} * \text{辅助预分频比}}$	
注 1: 基于 $F_{CY} = F_{OSC}/2$; 打盹模式和 PLL 被禁止。	

表 15-1: 示例 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: 基于 $F_{CY} = F_{OSC}/2$; 打盹模式和 PLL 被禁止。
2: 表中 SCK1 频率的单位是 kHz。

16.0 I²C™

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于 I²C 的更多信息，请参见《PIC24F 系列参考手册》的“第 24 章 I²C™”（DS39702A_CN）。

I²C™ 模块是用于同其他外设或单片机器件进行通信的串行接口。这些外设器件可以是串行数据 EEPROM、显示驱动器、A/D 转换器等。

I²C 模块支持以下特性：

- 独立的主 / 从逻辑
- 7 位和 10 位器件地址
- I²C 协议中所定义的广播呼叫地址
- 时钟延长功能，为处理器响应从器件数据请求提供延时
- 100 kHz 和 400 kHz 总线规范
- 可配置的地址掩码
- 多主机模式以防仲裁时报文丢失
- 总线转发器模式，允许作为从器件接收所有消息，与地址无关
- 自动 SCL

图 16-1 给出了该模块的框图。

16.1 引脚重映射选项

I²C 模块与固定引脚连接。为了灵活地复用外设，在器件配置期间，可以将 28 引脚器件中的 I²C1 模块重新分配到备用引脚（称为 SCL1 和 SDA1）。

引脚分配由 I²C1SEL 配置位控制。编程该位（= 0）时，模块被分配给 SCL1 和 SDA1 引脚。

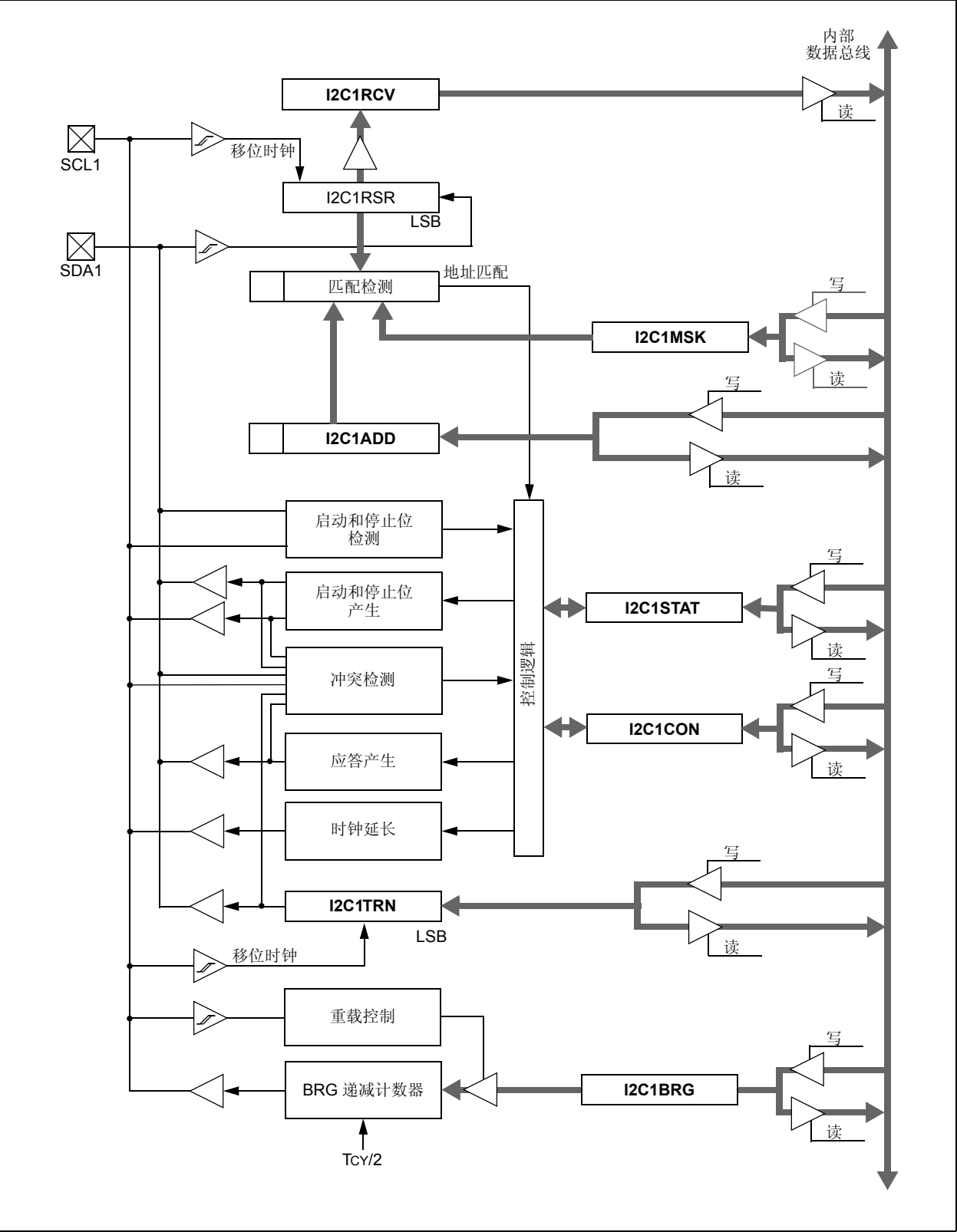
16.2 在单主机环境下作为主器件进行通信

在主模式下发送报文的详细信息取决于用于与器件通信的通信协议。通常，该事件的序列如下所示：

1. 在 SDA1 和 SCL1 上发出一个启动条件。
2. 向从器件发送 I²C 器件地址字节和写操作指示。
3. 等待并验证来自从器件的应答。
4. 向从器件发送第一个数据字节（有时称为命令）。
5. 等待并验证来自从器件的应答。
6. 向从器件发送串行存储器地址低字节。
7. 重复步骤 4 和 5，直到发送完所有数据字节。
8. 在 SDA1 和 SCL1 上发出一个重复启动条件。
9. 向从器件发送器件地址字节和读操作指示。
10. 等待并验证来自从器件的应答。
11. 使能主器件接收，以接收串行存储器数据。
12. 在数据字节接收结束时产生 ACK 或 NACK 条件。
13. 在 SDA1 和 SCL1 上产生一个停止条件。

PIC24F16KA102 系列

图 16-1: I²C™ 框图



16.3 设置作为总线主器件工作时的波特率

要计算波特率发生器（Baud Rate Generator，BRG）的重载值，可使用公式 16-1。

公式 16-1: 计算波特率重载值⁽¹⁾

$$F_{SCL} = \frac{F_{CY}}{I2C1BRG + 1 + \frac{F_{CY}}{10,000,000}}$$

或

$$I2C1BRG = \left(\frac{F_{CY}}{F_{SCL}} - \frac{F_{CY}}{10,000,000} \right) - 1$$

注 1: 基于 $F_{CY} = F_{OSC}/2$ ，打盹模式和 PLL 被禁止。

16.4 从地址掩码

I2C1MSK 寄存器（寄存器 16-3）将 7 位和 10 位寻址模式下地址中的某些位指定为“无关位”。将 I2C1MSK 寄存器中某个特定位置 1（= 1），不论相应的地址位的值是 0 还是 1，工作在从模式下的模块都会作出响应。例如，当将 I2C1MSK 设置为 00100000 时，工作在从模式下的模块将检测两个地址 00000000 和 00100000。

要使用地址掩码，必须通过将 IPMIEN 位（I2C1CON<11>）清零来禁止智能外设管理接口（Intelligent Peripheral Management Interface，IPMI）。

注：新修改的 I²C 协议使得表 16-2 中的地址保留，而且在从模式下不会应答。这包括包含任何这些地址的任何地址掩码设置。

表 16-1: I²C™ 时钟速率⁽¹⁾

必需的系统 F _{SCL}	F _{CY}	I2C1BRG 值		实际 F _{SCL}
		(十进制)	(十六进制)	
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
400 kHz	2 MHz	4	4	385 kHz
1 MHz	16 MHz	13	D	1.026 MHz
1 MHz	8 MHz	6	6	1.026 MHz
1 MHz	4 MHz	3	3	0.909 MHz

注 1: 基于 $F_{CY} = F_{OSC}/2$ ，打盹模式和 PLL 被禁止。

表 16-2: I²C™ 保留的地址⁽¹⁾

从器件地址	R/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 模式主机码
1111 1xx	x	保留
1111 0xx	x	10 位从地址高字节 ⁽³⁾

注 1: 这里所列的地址位将永远不会导致地址匹配，与地址掩码设置无关。

2: 仅当 GCEN = 1 时才会应答地址。

3: 只有 10 位寻址模式下的高字节才会与该地址发生匹配。

PIC24F16KA102 系列

寄存器 16-1: I2C1CON: I2C1 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:	HC = 硬件清零位		
R = 可读位	W = 可写位		U = 未实现位, 读为 0
-n = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15	I2CEN: I2C1 使能位 1 = 使能 I2C1 模块, 并将 SDA1 和 SCL1 引脚配置为串口引脚 0 = 禁止 I2C1 模块; 所有 I ² C™ 引脚由端口功能控制
bit 14	未实现: 读为 0
bit 13	I2CSIDL: 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	SCLREL: SCL1 释放控制位 (作为 I ² C 从器件工作时) 1 = 释放 SCL1 时钟 0 = 保持 SCL1 时钟为低电平 (时钟延长) <u>如果 STREN = 1:</u> 该位可读可写 (即软件可以写入 0 来启动时钟延长或写入 1 来释放时钟)。在从器件发送开始时由硬件清零。在从器件接收结束时由硬件清零。 <u>如果 STREN = 0:</u> 该位可读且可被置 1 (即软件只能写入 1 来释放时钟)。在从器件发送开始时由硬件清零。
bit 11	IPMIEN: 智能外设管理接口 (IPMI) 使能位 1 = 使能 IPMI 支持模式; 应答所有地址 0 = 禁止 IPMI 支持模式
bit 10	A10M: 10 位从器件寻址位 1 = I2C1ADD 为 10 位从器件地址 0 = I2C1ADD 为 7 位从器件地址
bit 9	DISSLW: 禁止斜率控制位 1 = 禁止斜率控制 0 = 使能斜率控制
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 输入门限值
bit 7	GCEN: 广播呼叫使能位 (作为 I ² C 从器件工作时) 1 = 允许在 I2C1RSR 接收到广播呼叫地址时产生中断 (已使能模块接收) 0 = 禁止广播呼叫地址
bit 6	STREN: SCL1 时钟延长使能位 (作为 I ² C 从器件工作时) 与 SCLREL 位配合使用。 1 = 使能软件或接收时钟延长 0 = 禁止软件或接收时钟延长

寄存器 16-1: I2C1CON: I2C1 控制寄存器 (续)

- bit 5 **ACKDT:** 应答数据位 (作为 I²C 主器件工作时; 适用于主器件接收过程)
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 (作为 I²C 主器件工作时; 适用于主器件接收过程)
1 = 在 SDA1 和 SCL1 引脚上发出应答序列, 并发送 ACKDT 数据位; 在主器件应答序列结束时由硬件清零
0 = 应答序列不在进行中
- bit 3 **RCEN:** 接收使能位 (作为 I²C 主器件工作时)
1 = 使能 I²C 接收模式; 在主器件接收完数据字节的第 8 位时由硬件清零
0 = 接收序列不在进行中
- bit 2 **PEN:** 停止条件使能位 (作为 I²C 主器件工作时)
1 = 在 SDA1 和 SCL1 引脚上发出停止条件; 在主器件停止序列结束时由硬件清零
0 = 停止条件不在进行中
- bit 1 **RSEN:** 重复启动条件使能位 (作为 I²C 主器件工作时)
1 = 在 SDA1 和 SCL1 引脚上发出重复启动条件; 在主器件重复启动序列结束时由硬件清零
0 = 重复启动条件不在进行中
- bit 0 **SEN:** 启动条件使能位 (作为 I²C 主器件工作时)
1 = 在 SDA1 和 SCL1 引脚上发出启动条件; 在主器件启动序列结束时由硬件清零
0 = 启动条件不在进行中

PIC24F16KA102 系列

寄存器 16-2: I2C1STAT: I2C1 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/A	P	S	R/W	RBF	TBF
bit 7							bit 0

图注:	C = 可清零位	HS = 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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15 **ACKSTAT:** 应答状态位
1 = 最后检测到 NACK
0 = 最后检测到 ACK
在应答结束时由硬件置 1 或清零。
- bit 14 **TRSTAT:** 发送状态位 (作为 I²C™ 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 = 主器件不在进行发送
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 = 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
在与匹配的 10 位地址的第 2 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 = 因为 I²C 模块忙, 尝试写 I2C1TRN 寄存器失败
0 = 未发生冲突
当总线忙时写 I2C1TRN 会使硬件将该位置 1 (由软件清零)。
- bit 6 **I2COV:** 接收溢出标志位
1 = 当 I2C1RCV 寄存器仍然保存原先的字节时接收到了新字节
0 = 未溢出
尝试将数据从 I2C1RSR 传输到 I2C1RCV 时由硬件置 1 (由软件清零)。
- bit 5 **D/A:** 数据 / 地址位 (作为 I²C 从器件工作时)
1 = 表示上次接收的字节为数据
0 = 表示上次接收的字节为器件地址
在器件地址匹配时由硬件清零; 写 I2C1TRN 或接收到从器件字节时由硬件置 1。
- bit 4 **P:** 停止位
1 = 表示上次检测到停止位
0 = 上次未检测到停止位
当检测到启动、重复启动或停止条件时由硬件置 1 或清零。

寄存器 16-2: I2C1STAT: I2C1 状态寄存器 (续)

- bit 3 **S:** 启动位
1 = 表示上次检测到启动 (或重复启动) 位
0 = 上次未检测到启动位
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/W:** 读 / 写信息位 (作为 I²C 从器件工作时)
1 = 读——表示数据自从器件输出
0 = 写——表示数据输入到从器件
接收到 I²C 器件地址字节后由硬件置 1 或清零。
- bit 1 **RBF:** 接收缓冲区满状态位
1 = 接收完成, I2C1RCV 为满
0 = 接收未完成, I2C1RCV 为空
用接收到的字节写 I2C1RCV 时由硬件置 1; 用软件读 I2C1RCV 时由硬件清零。
- bit 0 **TBF:** 发送缓冲区满状态位
1 = 发送正在进行中, I2C1TRN 为满
0 = 发送完成, I2C1TRN 为空
用软件写 I2C1TRN 时由硬件置 1; 数据发送完成时由硬件清零。

PIC24F16KA102 系列

寄存器 16-3: I2C1MSK: I2C1 从模式地址掩码寄存器

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 = POR 时的值 1 = 置 1 0 = 清零 x = 未知

bit 15-10 **未实现:** 读为 0
 bit 9-0 **AMSK<9:0>:** 地址中 bit x 的掩码选择位
 1 = 使能输入报文的地址中 bit x 的掩码; 在此位置上不需要位匹配
 0 = 禁止 bit x 的掩码; 在此位置上需要位匹配

寄存器 16-4: PADCFG1: 焊盘配置控制寄存器

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	R/W-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	SMBUSDEL	OC1TRIS ⁽³⁾	RTSECSEL1 ^(1,3)	RTSECSEL0 ^(1,3)	—
bit 7						bit 0	

图注:

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

bit 15-5 **未实现:** 读为 0
 bit 4 **SMBUSDEL:** SMBus SDA 输入延时选择位
 1 = I²C 模块配置为较长的 SMBus 输入延时 (标称延时为 300 ns)
 0 = I²C 模块配置为传统的输入延时 (标称延时为 150 ns)
 bit 0 **未实现:** 读为 0

- 注** 1: 要使能实际 RTCC 输出, RTCOE (RCFGCAL<10>) 位必须置 1。
 2: 要使能实际 OC1 输出, 必须使能 OCPWM1 模块。
 3: bit 3、bit 2 和 bit 1 如相关章节中所述。

17.0 通用异步收发器（UART）

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于通用异步收发器的更多信息，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 器件系列提供的串行 I/O 模块之一。UART 是可以与外设（例如个人计算机、LIN、RS-232 和 RS-485 接口）通信的全双工异步系统。模块还通过 UxCTS 和 UxRTS 引脚支持硬件流控制选项，其中还包括 IrDA® 编码器和解码器。

UART 模块的主要特性有：

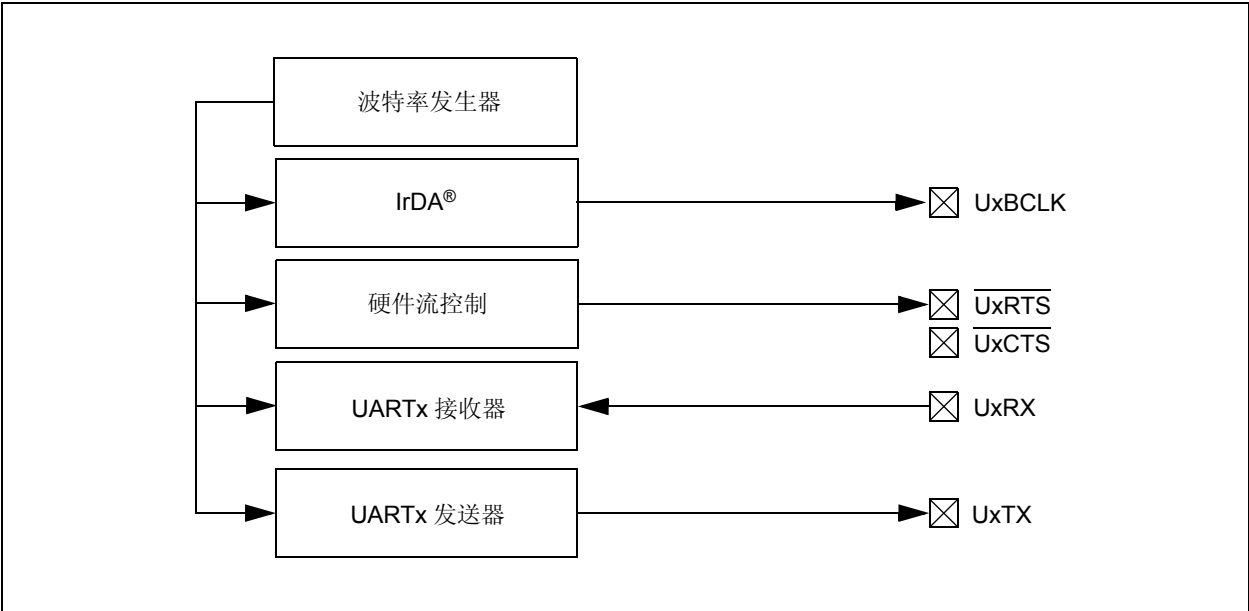
- 通过 UxTX 和 UxRX 引脚进行全双工 8 位或 9 位数据传输
- 偶校验、奇校验或无奇偶校验选项（对于 8 位数据）
- 一个或两个停止位
- 通过 UxCTS 和 UxRTS 引脚支持硬件流控制选项

- 完全集成的波特率发生器（Integrated Baud Rate Generator, IBRG），具有 16 位预分频器
- 当器件工作在 16 MIPS 时，波特率范围从 1 Mbps 到 15 Mbps
- 4 级深度先进先出（First-In-First-Out, FIFO）发送数据缓冲区
- 4 级深度 FIFO 接收数据缓冲区
- 奇偶校验、帧和缓冲区溢出错误检测
- 支持带地址检测的 9 位模式（第 9 位 = 1）
- 发送和接收中断
- 用于诊断支持的环回模式
- 支持同步和间隔字符
- 支持自动波特率检测
- IrDA 编码器和解码器逻辑
- 用于 IrDA 支持的 16 倍波特率时钟输出

图 17-1 给出了 UART 的简化框图。UART 模块由以下至关重要的硬件组件组成：

- 波特率发生器
- 异步发送器
- 异步接收器

图 17-1: UART 简化框图



PIC24F16KA102 系列

17.1 UART 波特率发生器 (BRG)

UART 模块包含一个专用的 16 位波特率发生器 (BRG)。UxBRG 寄存器控制一个自由运行的 16 位定时器的周期。公式 17-1 给出了 BRGH = 0 时计算波特率的公式。

公式 17-1: UART 波特率 (BRGH = 0) ⁽¹⁾

$$\text{波特率} = \frac{F_{CY}}{16 \cdot (UxBRG + 1)}$$

$$UxBRG = \frac{F_{CY}}{16 \cdot \text{波特率}} - 1$$

注 1: 基于 $F_{CY} = F_{OSC}/2$ ，打盹模式和 PLL 被禁止。

例 17-1 给出了如下条件下的波特率误差计算：

- $F_{CY} = 4 \text{ MHz}$
- 目标波特率 = 9600

例 17-1: 波特率误差计算 (BRGH = 0) ⁽¹⁾

$$\text{目标波特率} = F_{CY}/(16 (UxBRG + 1))$$

求解 UxBRG 值：

$$UxBRG = ((F_{CY}/\text{目标波特率})/16) - 1$$

$$UxBRG = ((4000000/9600)/16) - 1$$

$$UxBRG = 25$$

$$\begin{aligned}\text{计算波特率} &= 4000000/(16 (25 + 1)) \\ &= 9615\end{aligned}$$

$$\begin{aligned}\text{误差} &= \frac{(\text{计算波特率} - \text{目标波特率})}{\text{目标波特率}} \\ &= (9615 - 9600)/9600 \\ &= 0.16\%\end{aligned}$$

注 1: 基于 $F_{CY} = F_{OSC}/2$ ，打盹模式和 PLL 被禁止。

最大可能波特率 (BRGH = 0) 是 $F_{CY}/16$ (当 $UxBRG = 0$ 时)，最小可能波特率是 $F_{CY}/(16 * 65536)$ 。

公式 17-2 给出了 BRGH = 1 时计算波特率的公式。

公式 17-2: UART 波特率 (BRGH = 1) ⁽¹⁾

$$\text{波特率} = \frac{F_{CY}}{4 \cdot (UxBRG + 1)}$$

$$UxBRG = \frac{F_{CY}}{4 \cdot \text{波特率}} - 1$$

注 1: 基于 $F_{CY} = F_{OSC}/2$ ，打盹模式和 PLL 被禁止。

最大可能波特率 (BRGH = 1) 是 $F_{CY}/4$ (当 $UxBRG = 0$ 时)，最小可能波特率是 $F_{CY}/(4 * 65536)$ 。

向 UxBRG 寄存器写入新值会使 BRG 定时器复位 (清零)。这可以确保 BRG 无需等待定时器溢出就可以产生新的波特率。

17.2 8 位数据模式下的发送

1. 设置 UART：
 - a) 将适当的值写入数据位、奇偶校验位和停止位。
 - b) 将适当的波特率值写入 UxBRG 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（置 1 后两个周期产生发送中断）。
4. 将数据字节写入 UxTXREG 字的低字节。该值将被立即传输到发送移位寄存器（TSR），且在波特率时钟的下一个上升沿开始移出串行比特流。
5. 或者，当 UTXEN = 0 时，数据字节也可被发送，且随后用户可将 UTXEN 置 1。由于波特率时钟将从清零状态启动，这将立即开始发送串行比特流。
6. 中断控制位 UTXISELx 的设置决定何时产生发送中断。

17.3 9 位数据模式下的发送

1. 设置 UART（如第 17.2 节“8 位数据模式下的发送”中所述）。
2. 使能 UART。
3. 将 UTXEN 位置 1（置 1 后两个周期产生发送中断）。
4. 仅向 UxTXREG 写入一个 16 位的值。
5. 向 UxTXREG 写入一个字可触发 9 位数据向 TSR 的传输。串行比特流将会在波特率时钟的第一个上升沿开始移出。
6. 中断控制位 UTXISELx 的设置决定何时产生发送中断。

17.4 间隔和同步发送序列

下述序列会发送一个报文帧头，包括一个间隔字符和其后的一个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 置 1——设置间隔字符。
3. 将一个无效字符装入 UxTXREG 以启动发送（值被忽略）。
4. 向 UxTXREG 写入“55h”——将同步字符装入发送 FIFO 中。
5. 间隔字符发送后，硬件会将 UTXBRK 位复位。然后开始发送同步字符。

17.5 8 位或 9 位数据模式下的接收

1. 设置 UART（如第 17.2 节“8 位数据模式下的发送”中所述）。
2. 使能 UART。
3. 当接收到一个或多个数据字符时，将会根据中断控制位 URXISELx 的设置产生接收中断。
4. 读 OERR 位以确定是否发生了溢出错误。OERR 位必须用软件复位。
5. 读 UxRXREG。

读取 UxRXREG 字符的行为会将下一个字符传送到接收 FIFO 的顶部，其中包含一组新的 PERR 和 FERR 值。

17.6 UxCTS 和 UxRTS 控制引脚的操作

UARTx 允许发送（UxCTS）和请求发送（UxRTS）是两个与 UART 模块相关、由硬件控制的引脚。这两个引脚允许 UART 运行在单工模式和流控制模式下。它们用于控制 UART 与数据终端设备（Data Terminal Equipment, DTE）之间的发送和接收。UxMODE 寄存器中的 UEN<1:0> 位用来配置这两个引脚。

17.7 红外支持

UART 模块提供两种类型的红外 UART 支持：一种是 IrDA 时钟输出，用于支持外部 IrDA 编码器和解码器（传统模块支持）；另一种是完全实现的 IrDA 编码器和解码器。

由于 IrDA 模式需要 16 倍波特率时钟，它们仅在 BRGH 位（UxMODE<3>）为 0 时才能工作。

17.7.1 外部 IrDA 支持——IrDA 时钟输出

为了支持外部 IrDA 编码器和解码器，可将 UxBCLK 引脚（和 UxRTS 引脚相同）配置为产生 16 倍波特率时钟。当使能了 UART 模块且 UEN<1:0> = 11 时，UxBCLK 引脚将输出 16 倍波特率时钟；它可用于支持 IrDA 编解码器芯片。

17.7.2 内置 IrDA 编码器和解码器

UART 模块在其内部完全实现了 IrDA 编码器和解码器。内置 IrDA 编码器和解码器的功能可通过 IREN 位（UxMODE<12>）来使能。当使能（IREN = 1）时，接收引脚（UxRX）可作为红外接收器的输入引脚。发送引脚（UxTX）可作为红外发送器的输出引脚。

PIC24F16KA102 系列

寄存器 17-1: 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 ⁽²⁾	R/W-0 ⁽²⁾
UARTEN	—	USIDL	IREN ⁽¹⁾	RTSMD	—	UEN1	UEN0
bit 15							bit 8

R/C-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:	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 = POR 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; 由端口锁存器控制所有 UARTx 引脚; 此时 UARTx 的功耗最小
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 [®] 编码器和解码器使能位 ⁽¹⁾ 1 = 使能 IrDA 编码器和解码器 0 = 禁止 IrDA 编码器和解码器
bit 11	RTSMD: <u>UxRTS</u> 引脚模式选择位 1 = <u>UxRTS</u> 引脚处于单工模式 0 = <u>UxRTS</u> 引脚处于流控制模式
bit 10	未实现: 读为 0
bit 9-8	UEN<1:0>: UARTx 使能位 ⁽²⁾ 11 = 使能并使用 UxTX、UxRX 和 <u>UxBCLK</u> 引脚; <u>UxCTS</u> 引脚由端口锁存器控制 10 = 使能并使用 UxTX、UxRX、 <u>UxCTS</u> 和 <u>UxRTS</u> 引脚 01 = 使能并使用 UxTX、UxRX 和 <u>UxRTS</u> 引脚; <u>UxCTS</u> 引脚由端口锁存器控制 00 = 使能并使用 UxTX 和 UxRX 引脚; <u>UxCTS</u> 和 <u>UxRTS/UxBCLK</u> 引脚由端口锁存器控制
bit 7	WAKE: 在休眠模式下检测到启动位唤醒使能位 1 = UARTx 将继续采样 UxRX 引脚; 在出现下降沿时产生中断, 在出现下一个上升沿时由硬件清零该位 0 = 禁止唤醒
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 = 使能对下一个字符的波特率测量——需要在接收其他数据前接收同步字段 (55h); 完成时由硬件清零 0 = 禁止波特率测量或测量已完成
bit 4	RXINV: 接收极性翻转位 1 = UxRX 的空闲状态为 0 0 = UxRX 的空闲状态为 1

注 1: 此功能只能在 16 倍 BRG 模式 (BRGH = 0) 下使用。

2: 位的可用性取决于引脚的可用性。

寄存器 17-1: UxMODE: UARTx 模式寄存器 (续)

bit 3	BRGH: 高波特率使能位 1 = BRG 在每个位周期内产生 4 个时钟信号 (4 倍波特率时钟, 高速模式) 0 = BRG 在每个位周期内产生 16 个时钟信号 (16 倍波特率时钟, 标准模式)
bit 2-1	PDSEL<1:0>: 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 = 2 个停止位 0 = 1 个停止位

- 注 1: 此功能只能在 16 倍 BRG 模式 (BRGH = 0) 下使用。
2: 位的可用性取决于引脚的可用性。

PIC24F16KA102 系列

寄存器 17-2: 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, HSC	R-1, HSC
UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R-1, HSC	R-0, HSC	R-0, HSC	R/C-0, HS	R-0, HSC
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	
	HS = 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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15,13 **UTXISEL<1:0>**: 发送中断模式选择位
11 = 保留; 不要使用
10 = 当一个字符被传输到发送移位寄存器 (TSR) 导致发送缓冲区为空时, 产生中断
01 = 当最后一个字符被移出发送移位寄存器; 所有发送操作执行完毕时产生中断
00 = 当一个字符被传输到发送移位寄存器 (这意味着发送缓冲区中至少还有一个字符) 时产生中断
- bit 14 **UTXINV**: IrDA® 编码器发送极性翻转位
如果 IREN = 0:
1 = UxTX 的空闲状态为 0
0 = UxTX 的空闲状态为 1
如果 IREN = 1:
1 = UxTX 的空闲状态为 1
0 = UxTX 的空闲状态为 0
- bit 12 **未实现**: 读为 0
- bit 11 **UTXBRK**: 发送间隔位
1 = 在下次发送时发出同步间隔字符——启动位, 后跟 12 个 0 位, 然后是停止位; 完成时由硬件清零
0 = 禁止或已完成同步间隔字符的发送
- bit 10 **UTXEN**: 发送使能位
1 = 使能发送, UARTx 控制 UxTX 引脚
0 = 禁止发送, 中止所有等待的发送, 缓冲区被复位。由 PORT 寄存器控制 UxTX 引脚。
- bit 9 **UTXBF**: 发送缓冲区满状态位 (只读)
1 = 发送缓冲区满
0 = 发送缓冲区未满, 至少还可写入一个或多个字符
- bit 8 **TRMT**: 发送移位寄存器空位 (只读)
1 = 发送移位寄存器为空, 同时发送缓冲区为空 (上一次发送已完成)
0 = 发送移位寄存器非空, 发送在进行中或在发送缓冲区中排队
- bit 7-6 **URXISEL<1:0>**: 接收中断模式选择位
11 = 当 RSR 传输使接收缓冲区为满时 (即有 4 个数据字符), 中断标志位置 1
10 = 当 RSR 传输使接收缓冲区 3/4 满时 (即有 3 个数据字符), 中断标志位置 1
0x = 当接收到任何一个字符且将字符从 RSR 传输到接收缓冲区时, 中断标志位置 1。接收缓冲区有一个或多个字符。

寄存器 17-2: UxSTA: UARTx 状态和控制寄存器 (续)

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, 这个控制位将无效。 0 = 禁止地址检测模式
bit 4	RIDLE: 接收器空闲位 (只读) 1 = 接收器空闲 0 = 接收器工作
bit 3	PERR: 奇偶校验错误状态位 (只读) 1 = 检测到当前字符 (接收 FIFO 顶部的字符) 的奇偶校验错误 0 = 未检测到奇偶校验错误
bit 2	FERR: 帧错误状态位 (只读) 1 = 检测到当前字符 (接收 FIFO 顶部的字符) 的帧错误 0 = 未检测到帧错误
bit 1	OERR: 接收缓冲区溢出错误状态位 (清零 / 只读) 1 = 接收缓冲区已溢出 0 = 接收缓冲区未溢出 (清零原来置 1 的 OERR 位 (1 → 0 的跳变) 将使接收缓冲区和 RSR 复位为空状态)
bit 0	URXDA: 接收缓冲区中是否有数据标志位 (只读) 1 = 接收缓冲区中有数据; 有至少一个或多个字符可被读取 0 = 接收缓冲区为空

PIC24F16KA102 系列

寄存器 17-3: UxTXREG: UARTx 发送寄存器

U-x	U-x	U-x	U-x	U-x	U-x	U-x	W-x
—	—	—	—	—	—	—	UTX8
bit 15							bit 8
W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
UTX7	UTX6	UTX5	UTX4	UTX3	UTX2	UTX1	UTX0
bit 7							bit 0

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

bit 15-9 未实现: 读为 0
bit 8 **UTX8**: 已发送字符的数据位 (在 9 位模式下)
bit 7-0 **UTX<7:0>**: 已发送字符的数据位

寄存器 17-4: UxRXREG: UARTx 接收寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R-0, HSC
—	—	—	—	—	—	—	URX8
bit 15							bit 8
R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC
URX7	URX6	URX5	URX4	URX3	URX2	URX1	URX0
bit 7							bit 0

图注:			
R = 可读位	HSC = 硬件置 1/ 清零位	W = 可写位	U = 未实现位, 读为 0
-n = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15-9 未实现: 读为 0
bit 8 **URX8**: 已接收字符的数据位 (在 9 位模式下)
bit 7-0 **URX<7:0>**: 已接收字符的数据位

18.0 实时时钟和日历（RTCC）

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于实时时钟和日历的更多信息，请参见《PIC24F 系列参考手册》的“第 29 章 实时时钟和日历（RTCC）”（DS39696A_CN）。

RTCC 向用户提供了可进行校准的实时时钟和日历（RTCC）功能。

RTCC 模块的主要特性包括：

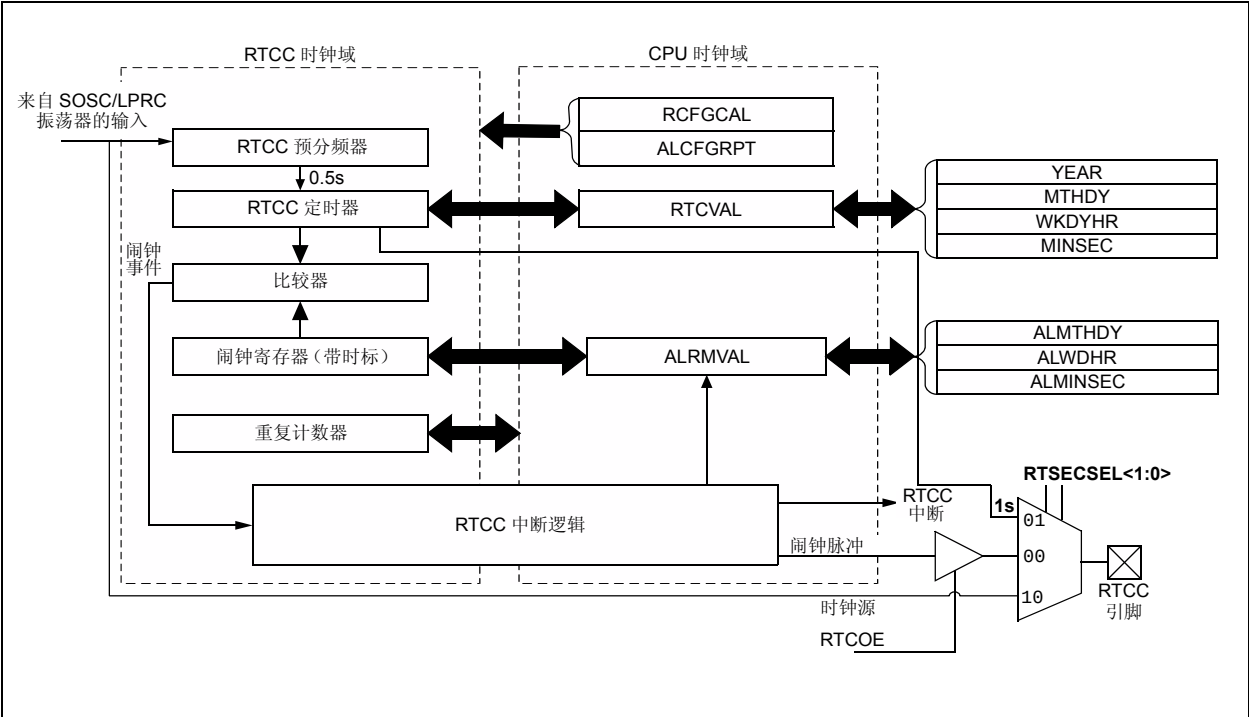
- 可在深度休眠模式下工作
- 可选择的时钟源
- 使用 24 小时格式提供小时、分钟和秒钟
- 可分辨半秒的时长
- 提供日历——星期、日期、月和年

- 闹钟可配置为半秒、1 秒、10 秒、1 分钟、10 分钟、1 小时、1 天、1 周、1 个月或 1 年
- 闹钟使用递减计数器进行重复
- 闹钟具有无限重复响铃
- 年份为 2000 至 2099，带闰年修正
- BCD 格式以减少软件开销
- 为长期电池工作进行了优化
- 32.768 kHz 时钟晶振的用户校准 / 带周期性自动调整的 32K INTRC 频率

18.1 RTCC 源时钟

用户可以选择SOSC晶振或LPRC内部振荡器作为RTCC模块的参考时钟。可通过 RTCCSEL（FDS<5>）配置位进行配置。这为用户提供了一个基于系统总需求来权衡系统成本、精度和功耗的选择。

图 18-1: RTCC 框图



PIC24F16KA102 系列

18.2 RTCC 模块寄存器

RTCC 模块寄存器可分为以下三类：

- RTCC 控制寄存器
- RTCC 值寄存器
- 闹钟值寄存器

18.2.1 寄存器映射

为限制寄存器接口，RTCC 定时器和闹钟定时器寄存器通过相应的寄存器指针访问。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，RTCC 指针值 RTCPTR<1:0> 位递减 1，直到达到 00。一旦达到 00，分钟和秒值可通过 RTCVALH 和 RTCVALL 访问，直到手动更改指针值。

表 18-1: RTCC 值寄存器映射

RTCPTR<1:0>	RTCC 值寄存器窗口	
	RTCVAL<15:8>	RTCVAL<7:0>
00	分钟	秒
01	星期	小时
10	月	日
11	—	年

闹钟值寄存器窗口（ALRMVALH 和 ALRMVALL）使用 ALRMPTR 位（ALCFGRT<9:8>）选择所需的闹钟寄存器对（见表 18-2）。

通过写 ALRMVALH 字节，闹钟指针值 ALRMPTR<1:0> 位递减 1，直到达到 00。一旦达到 00，ALRMMIN 和 ALRMSEC 值可通过 ALRMVALH 和 ALRMVALL 访问，直到手动更改指针值。

例 18-1: 将 RTCWREN 位置 1

```
asm volatile("push w7");
asm volatile("push w8");
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
asm volatile("pop w8");
asm volatile("pop w7");
```

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 位内核并不区分 8 位和 16 位读操作，用户要注意，读 ALRMVALH 或 ALRMVALL 字节时都会使 ALRMPTR<1:0> 值递减 1。同样的规律也适用于 RTCVALH 或 RTCVALL 字节，读取它们时会使 RTCPTR<1:0> 递减 1。

注： 这只适用于读操作，不适用于写操作。

18.2.2 写锁定

为执行对任何 RTCC 定时器寄存器的写入，RTCWREN 位（RCFGCAL<13>）都必须置 1（见例 18-1）。

注： 为避免意外写入定时器，建议其他任何时候 RTCWREN 位（RCFGCAL<13>）都保持清零。要将 RTCWREN 位置 1，在 55h/AA 序列和 RTCWREN 置 1 之间只允许一个指令周期的时间窗；因此，建议遵循例 18-1 中的代码示例。

18.2.3 选择 RTCC 时钟源

RTCC 模块的时钟源可通过 RTCKSEL（FDS<5>）位进行选择。当将该位设为 1 时，辅助振荡器（SOSC）用作参考时钟；当将该位设为 0 时，LPRC 用作参考时钟。

18.2.4 RTCC 控制寄存器

寄存器 18-1: RCFGAL: RTCC 校准和配置寄存器⁽¹⁾

R/W-0	U-0	R/W-0	R-0, HSC	R-0, HSC	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:	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 = POR 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 = RTCVALH 和 RTCVALL 寄存器可由用户写入
0 = RTCVALH 和 RTCVALL 寄存器已锁定, 不可由用户写入

bit 12 **RTCSYNC:** RTCC 值寄存器读同步位

1 = 由于计满返回, RTCVALH、RTCVALL 和 ALCFGRPT 寄存器在读操作过程中可能改变, 从而导致读取的数据无效。如果两次读取寄存器得到的数据相同, 可认为数据是有效的。
0 = RTCVALH、RTCVALL 或 ALCFGRPT 寄存器在读取时无需考虑计满返回

bit 11 **HALFSEC:** 半秒状态位⁽³⁾

1 = 一秒的后一半
0 = 一秒的前一半

bit 10 **RTCOE:** RTCC 输出使能位

1 = 使能 RTCC 输出
0 = 禁止 RTCC 输出

bit 9-8 **RTCPTR<1:0>:** RTCC 值寄存器窗口指针位

读取 RTCVALH 和 RTCVALL 寄存器时, 指向相应的 RTCC 值寄存器。每当读或写 RTCVALH 时 RTCPTR<1:0> 的值就递减 1, 直到达到 00。

RTCVAL<15:8>:

00 = 分钟数
01 = 星期
10 = 月
11 = 保留

RTCVAL<7:0>:

00 = 秒数
01 = 小时数
10 = 日
11 = 年

注 1: RCFGAL 寄存器只受 POR 的影响。

2: 仅当 RTCWREN = 1 时允许写入 RTCEN 位。

3: 该位是只读位; 写入 MINSEC 寄存器的低半部分时, 它被清零。

PIC24F16KA102 系列

寄存器 18-1: RCFGAL: RTCC 校准和配置寄存器⁽¹⁾ (续)

bit 7-0 **CAL<7:0>**: RTC 漂移校准位
01111111 = 最大正向调整; 每分钟增加 508 个 RTC 时钟脉冲
.
.
.
01111111 = 最小正向调整; 每分钟增加 4 个 RTC 时钟脉冲
00000000 = 无调整
11111111 = 最小负向调整; 每分钟减少 4 个 RTC 时钟脉冲
.
.
.
10000000 = 最大负向调整; 每分钟减少 512 个 RTC 时钟脉冲

- 注 1: RCFGAL 寄存器只受 POR 的影响。
 2: 仅当 RTCWREN = 1 时允许写入 RTCEN 位。
 3: 该位是只读位; 写入 MINSEC 寄存器的低半部分时, 它被清零。

寄存器 18-2: PADCFG1: 焊盘配置控制寄存器

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	R/W-0	R/W-0	R/W-0	R/W-0	U-0
—	—	—	SMBUSDEL	OC1TRIS	RTSECSEL1 ⁽¹⁾	RTSECSEL0 ⁽¹⁾	—
bit 7							bit 0

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

bit 15-5 **未实现**: 读为 0
bit 4-3 如第 14.0 节 “输出比较” 和第 16.0 节 “I2C ±” 中所述。
bit 2-1 **RTSECSEL<1:0>**: RTCC 秒时钟输出选择位⁽¹⁾
 11 = 保留; 不要使用
 10 = 为 RTCC 引脚选择了 RTCC 源时钟 (可以是 LPRC 或 SOSC, 取决于 RTCKSEL (FDS<5>) 位的设置)
 01 = 为 RTCC 引脚选择了 RTCC 秒时钟
 00 = 为 RTCC 引脚选择了 RTCC 闹钟脉冲
bit 0 **未实现**: 读为 0

- 注 1: 要启用实际 RTCC 输出, RTCOE (RCFGAL<10>) 位必须置 1。

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 = POR 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> = 00h 且 CHIME = 0, 发生闹钟事件后都会自动清零)

0 = 禁止闹钟

bit 14

CHIME: 响铃 (Chime) 使能位

1 = 使能响铃; ARPT<7:0> 位允许从 00h 返回到 FFh

0 = 禁止响铃; ARPT<7:0> 到达 00h 就停止

bit 13-10

AMASK<3:0>: 闹钟时标配置位

0000 = 每半秒

0001 = 每秒

0010 = 每 10 秒

0011 = 每分钟

0100 = 每 10 分钟

0101 = 每小时

0110 = 一天一次

0111 = 一周一次

1000 = 一月一次

1001 = 一年一次 (除非配置在 2 月 29 日, 亦即每 4 年一次的情况外)

101x = 保留——不要使用

11xx = 保留——不要使用

bit 9-8

ALRMPTR<1:0>: 闹钟值寄存器窗口指针位

读取 ALRMVALH 和 ALRMVALL 寄存器时, 指向相应的闹钟值寄存器。每当读或写 ALRMVALH 时 ALRMPTR<1:0> 的值就递减 1, 直到达到 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

ARPT<7:0>: 闹钟重复计数器值位

11111111 = 闹钟将再重复 255 次

.

.

.

00000000 = 闹钟将不再重复

每当发生闹钟事件时计数器就递减 1; 除非 CHIME = 1, 否则计数器不能从 00h 返回到 FFh。

PIC24F16KA102 系列

18.2.5 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	YRTEN2	YRTEN1	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 = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-8 未实现: 读为 0

bit 7-4 **YRTEN<3:0>**: 年份的十位数的二 - 十进制码 (Binary Coded Decimal, BCD) 值为 0 到 9。

bit 3-0 **YRONE<3:0>**: 年份的个位数的 BCD 值为 0 到 9。

注 1: 仅当 RTCWREN = 1 时允许写入 YEAR 寄存器。

寄存器 18-5: MTHDY: 月和日值寄存器⁽¹⁾

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MHTTEN0	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 = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现: 读为 0

bit 12 **MHTTEN0**: 月份的十位数的 BCD 值为 0 或 1。

bit 11-8 **MTHONE<3:0>**: 月份的个位数的 BCD 值为 0 到 9。

bit 7-6 未实现: 读为 0

bit 5-4 **DAYTEN<1:0>**: 日的十位数的 BCD 值为 0 到 3。

bit 3-0 **DAYONE<3:0>**: 日的个位数的 BCD 值为 0 到 9。

注 1: 仅当 RTCWREN = 1 时允许写入该寄存器。

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 = POR 时的值 1 = 置 1 0 = 清零 x = 未知

- bit 15-11 **未实现:** 读为 0
bit 10-8 **WDAY<2:0>:** 星期的 BCD 值
 值为 0 到 6。
bit 7-6 **未实现:** 读为 0
bit 5-4 **HRTEN<1:0>:** 小时的十位数的 BCD 值
 值为 0 到 2。
bit 3-0 **HRONE<3:0>:** 小时的个位数的 BCD 值
 值为 0 到 9。

注 1: 仅当 RTCWREN = 1 时允许写入该寄存器。

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 = POR 时的值 1 = 置 1 0 = 清零 x = 未知

- bit 15 **未实现:** 读为 0
bit 14-12 **MINTEN<2:0>:** 分钟的十位数的 BCD 值
 值为 0 到 5。
bit 11-8 **MINONE<3:0>:** 分钟的个位数的 BCD 值
 值为 0 到 9。
bit 7 **未实现:** 读为 0
bit 6-4 **SECTEN<2:0>:** 秒的十位数的 BCD 值
 值为 0 到 5。
bit 3-0 **SECONE<3:0>:** 秒的个位数的 BCD 值
 值为 0 到 9。

PIC24F16KA102 系列

18.2.6 ALRMVAL 寄存器映射

寄存器 18-8: ALMTHDY: 闹钟月和日值寄存器⁽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 = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现: 读为 0

bit 12 **MTHTEN0**: 月份的十位数的 BCD 值
值为 0 或 1。

bit 11-8 **MTHONE<3:0>**: 月份的个位数的 BCD 值
值为 0 到 9。

bit 7-6 未实现: 读为 0

bit 5-4 **DAYTEN<1:0>**: 日的十位数的 BCD 值
值为 0 到 3。

bit 3-0 **DAYONE<3:0>**: 日的个位数的 BCD 值
值为 0 到 9。

注 1: 仅当 RTCWREN = 1 时允许写入该寄存器。

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 = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-11 未实现: 读为 0

bit 10-8 **WDAY<2:0>**: 星期的 BCD 值
值为 0 到 6。

bit 7-6 未实现: 读为 0

bit 5-4 **HRTEN<1:0>**: 小时的十位数的 BCD 值
值为 0 到 2。

bit 3-0 **HRONE<3:0>**: 小时的个位数的 BCD 值
值为 0 到 9。

注 1: 仅当 RTCWREN = 1 时允许写入该寄存器。

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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15 未实现：读为 0
- bit 14-12 **MINTEN<2:0>**: 分钟的十位数的 BCD 值
 值为 0 到 5。
- bit 11-8 **MINONE<3:0>**: 分钟的个位数的 BCD 值
 值为 0 到 9。
- bit 7 未实现：读为 0
- bit 6-4 **SECTEN<2:0>**: 秒的十位数的 BCD 值
 值为 0 到 5。
- bit 3-0 **SECONE<3:0>**: 秒的个位数的 BCD 值
 值为 0 到 9。

18.3 校准

实时晶振输入可用周期性自动调节功能校准。正确校准后，RTCC可提供小于每月3秒的误差。为实现该校准，需要找到误差时钟脉冲数并将该值存储到RCFGCAL寄存器的低半部分。装入RCFGCAL低半部分的8位有符号值乘以4，每分钟从RTCC定时器中加上或减去一次。关于RTCC校准，请参见以下步骤：

1. 使用器件上的其他定时器资源，用户必须找出32.768 kHz晶振的误差。
2. 知道误差后，必须将它转换为每分钟误差时钟脉冲数。
3. a) 如果振荡器快于理想频率（从步骤2得出负的结果），RCFGCAL寄存器值必须为负。这会导致每分钟从定时器计数器中减去指定的时钟脉冲数。
b) 如果振荡器慢于理想频率（从步骤2得出正的结果），RCFGCAL寄存器值必须为正。这会导致每分钟在定时器计数器中加上指定的时钟脉冲数。

将每分钟误差时钟数除以4得到正确的校准值，并将正确的值装入RCFGCAL寄存器。（校准值中每1位递增会加上或减去4个脉冲）。

公式 18-1:

$$\begin{aligned} &(\text{理想频率} \uparrow - \text{测得频率}) * 60 = \text{每分钟误差时钟数} \\ &\uparrow \text{理想频率} = 32,768 \text{ Hz} \end{aligned}$$

只有当定时器关闭或紧接秒脉冲的上升沿时，才会发生对RCFGCAL寄存器低半部分的写操作。

注： 是否在误差值中包含晶振初始误差、温度造成的漂移和晶振老化造成的漂移，由用户自行决定。

18.4 闹钟

- 可在半秒到一年的范围内配置
- 使用ALRMEN位（ALCFGRPT<15>）使能
- 有一次性闹钟和重复闹钟选项可用

18.4.1 配置闹钟

闹钟功能用ALRMEN位使能。发出闹钟后该位清零。只能在ALRMEN = 0时对ALRMVAL执行写操作。

闹钟的间隔时间通过AMASK位（ALCFGRPT<13:10>）选择，如图18-2所示。这些位决定了要触发闹钟，闹钟的哪些位、多少位必须和时钟值匹配。

也可以配置闹钟使之根据预先配置的间隔时间重复。闹钟使能后发生的总次数存储在ARPT<7:0>位（ALCFGRPT<7:0>）中。当ARPT位的值等于00h且CHIME位（ALCFGRPT<14>）清零时，重复功能被禁止，只发生单次闹钟。通过将FFh装入ARPT<7:0>，闹钟可重复最多255次。

每个闹钟发出后，ARPT位的值都递减1。值达到00h后，将最后一次发出闹钟，此后ALRMEN位将自动清零，闹钟将关闭。

如果CHIME位 = 1，闹钟可能不断重复。当CHIME置1时，ARPT位的值达到00h时不会禁止闹钟，而是返回到FFh，继续无限计数。

18.4.2 闹钟中断

每个闹钟事件发生时，都会产生中断。此外会提供闹钟脉冲输出，其频率是闹钟频率的一半。该输出完全和RTCC时钟同步，可用作其他外设的触发时钟。

注： 使能闹钟（ALRMEN = 1）时，更改除RCFGCAL和ALCFGRPT寄存器以外的任何寄存器以及CHIME位，都会导致误闹钟事件，进而导致错误的闹钟中断。为避免误闹钟事件，只应在禁止闹钟（ALRMEN = 0）时更改定时器和闹钟值。建议在RTCSYNC = 0时更改ALCFGRPT寄存器和CHIME位。

图 18-2: 闹钟时标设置

闹钟时标设置 (AMASK<3:0>)	星期	月	日	小时	分钟	秒
0000 ——每半秒 0001 ——每秒						
0010 ——每 10 秒						s
0011 ——每分钟						s s
0100 ——每 10 分钟					m	s s
0101 ——每小时					m m	s s
0110 ——每天				h h	m m	s s
0111 ——每周	d			h h	m m	s s
1000 ——每月			d d	h h	m m	s s
1001 ——每年 ⁽¹⁾		m m	d d	h h	m m	s s

注 1: 每年，除非配置为 2 月 29 日。

PIC24F16KA102 系列

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 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于可编程循环冗余校验的更多信息，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）。

PIC24F 器件中的可编程循环冗余校验（CRC）模块是可用软件配置的 CRC 校验和发生器。CRC 算法将报文视为二进制比特流，并将它除以固定的二进制数值。这种除法运算产生的余数将视为校验和。与除法运算一样，CRC 计算也是一个迭代过程。唯一的区别在于这些运算是通过基于模 2 的模运算完成。例如，除法运算被替换为异或运算（即，不带借位的减法）。CRC 算法使用多项式来执行所有计算。使用数字表示的除数、被除数和余数构造为带有二进制系数的多项式。

可编程 CRC 发生器提供以下特性：

- 用户可编程多项式 CRC 公式
- 中断输出
- 数据 FIFO

该模块实现了可用软件配置的 CRC 发生器。多项式的项及其长度可分别用 CRCXOR（X<15:1>）位和 CRCCON（PLEN<3:0>）位设定。

考虑以下 CRC 公式：

$$x^{16} + x^{12} + x^5 + 1$$

要将该多项式编程到 CRC 发生器中，CRC 寄存器中的位应按表 19-1 所示进行设置。

表 19-1: CRC 设置示例

位名称	位值
PLEN<3:0>	1111
X<15:1>	000100000010000

对于 X<15:1> 的值，第 12 位和第 5 位按照公式要求设为 1。公式的第 0 位总是其他位逻辑异或运算得到的结果。对于 16 位多项式，第 16 位也总是逻辑异或运算的结果；因此 X<15:1> 位没有第 0 位，也没有第 16 位。

标准 CRC 发生器的拓扑结构如图 19-2 所示。

图 19-1: CRC 移位寄存器详细信息

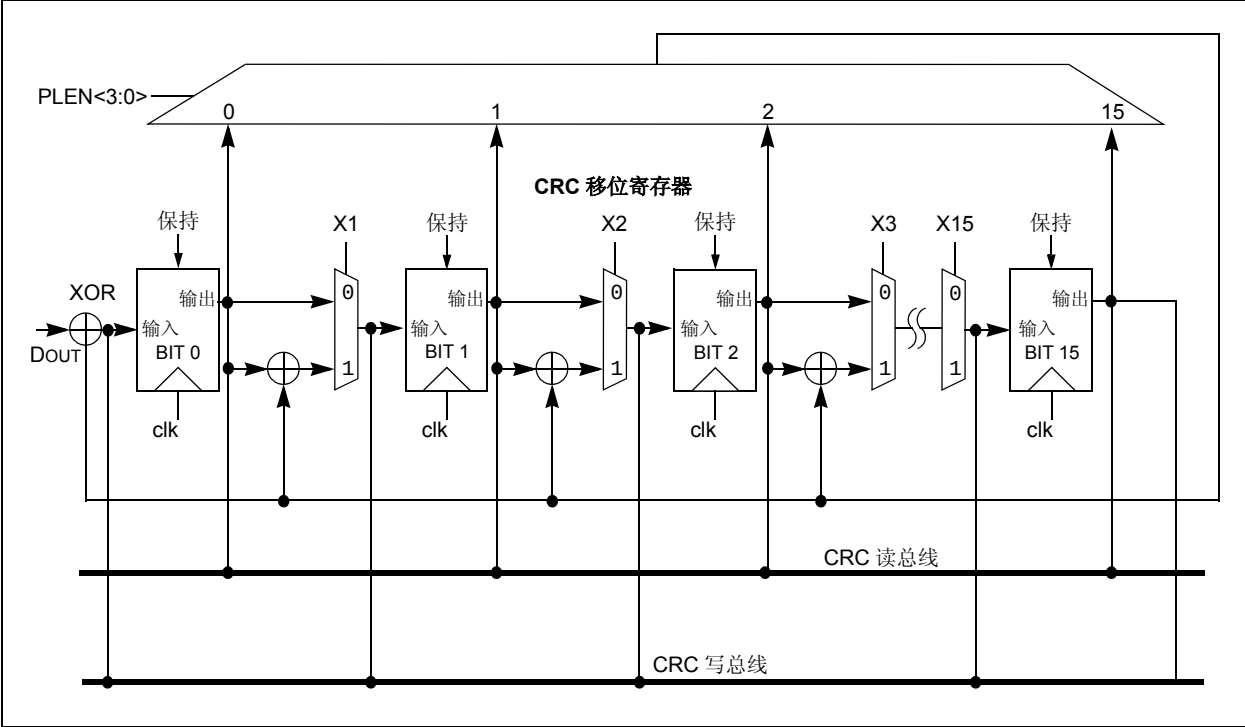
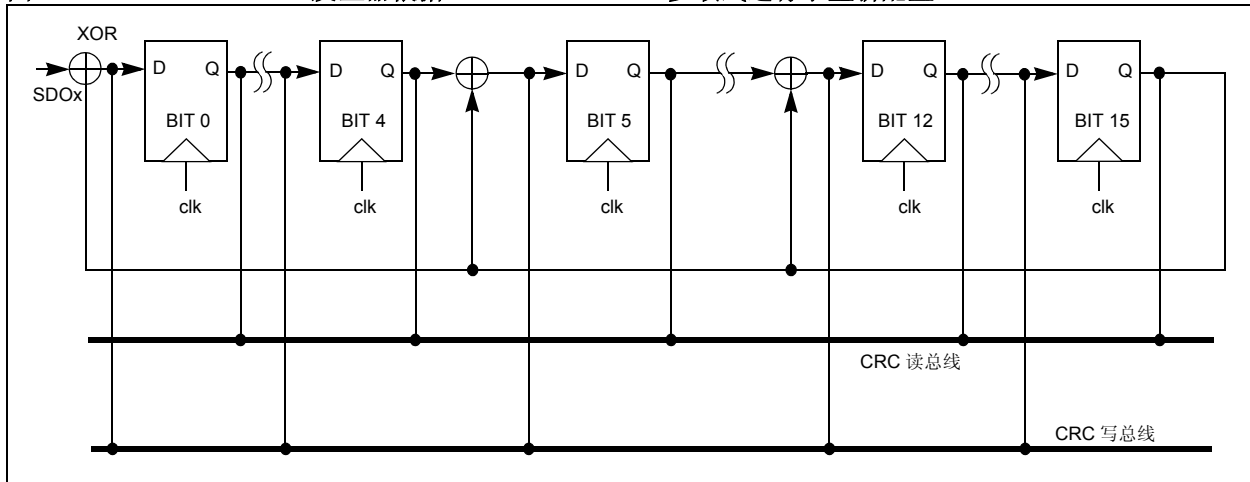


图 19-2: CRC 发生器根据 $x^{16} + x^{12} + x^5 + 1$ 多项式进行了重新配置



19.1 用户接口

19.1.1 数据接口

要开始串行移位，必须将 1 写入 CRCGO 位。

该模块包含的 FIFO 在 $PLEN < 3:0 > > 7$ 时为 8 级深，其他情况下为 16 级深。CRC 计算所需数据必须首先写入 FIFO。可写入 FIFO 的最小数据元素是 1 个字节。

例如，如果 $PLEN = 5$ ，则数据大小是 $PLEN + 1 = 6$ 。数据必须按如下方式写入：

```
data<5:0> = crc_input<5:0>
data<7:6> = bxx
```

数据写入 CRCWDAT MSb（由 PLEN 定义）后，VWORD 位（ $CRCCON < 12:8 >$ ）的值递增 1。当 $CRCGO = 1$ 且 $VWORD < 4:0 > > 0$ 时，串行移位器开始将数据移入 CRC 引擎。最高有效位（MSb）移出后，VWORD 位递减 1。串行移位器继续移位，直到 VWORD 位达到 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。

VWORD 位达到 8（或 16）时，CRCFUL 位将置 1。VWORD 位达到 0 时，CRCMPT 位将置 1。

要连续将数据传送到 CRC 引擎中，建议的工作模式是在 FIFO 中预留足够的存字空间，这样在写入下个字之前不会产生中断。完成后，通过将 CRCGO 位设为 1 来启动 CRC。在真正开始 CRC 之前将查询 VWORD 位。如果它们读到的数小于 8 或 16，将把另一个字写入 FIFO。

要清空已写入 FIFO 的字，CRCGO 位必须设为 1 并允许 CRC 移位器运行，直到 CRCMPT 位置 1。

同时，为了正确地读 CRC，必须等待 CRCMPT 位转为高电平才能读 CRCWDAT 寄存器。

如果 CRCFUL 位置 1 时写入一个字，VWORD 指针将计满返回到 0。硬件的行为就像 FIFO 为空时一样。但是，产生中断的条件并不满足；因此，不会产生中断（见第 19.1.2 节“中断操作”）。

写入 CRCWDAT 后必须经过至少一个指令周期才能读 VWORD 位。

19.1.2 中断操作

当 $VWORD < 4:0 >$ 位的值从 1 跳变到 0 时，将产生中断。

19.2 节能模式下的操作

19.2.1 休眠模式

如果模块工作时进入休眠模式，模块将暂停于当前状态，直到时钟恢复运行。

19.2.2 空闲模式

要在空闲模式下继续完整的模块工作，CSIDL 位必须在进入该模式前清零。

如果 $CSIDL = 1$ ，该模块的行为与在休眠模式下相同；暂停的中断事件将传递下去，即使模块时钟不可用。

19.3 寄存器

有 4 个寄存器用于控制可编程 CRC 的工作：

- CRCCON
- CRCXOR
- CRCDAT
- CRCWDAT

寄存器 19-1: CRCCON: CRC 控制寄存器

U-0	U-0	R/W-0	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC
—	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0
bit 15							bit 8
R-0, HSC	R-1, HSC	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:	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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15-14	未实现: 读为 0
bit 13	CSIDL: CRC 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 12-8	VWORD<4:0>: 指针值位 表示 FIFO 中有效字的个数。PLEN<3:0> > 7 时最大值是 8, PLEN<3:0> ≤ 7 时最大值是 16。
bit 7	CRCFUL: FIFO 满位 1 = FIFO 已满 0 = FIFO 未满
bit 6	CRCMPT: FIFO 空位 1 = FIFO 为空 0 = FIFO 非空
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 = CRC 串行移位器关闭
bit 3-0	PLEN<3:0>: 多项式长度位 表示生成多项式的长度减去 1。

PIC24F16KA102 系列

寄存器 19-2: CRCXOR: CRC XOR 多项式寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
X15	X14	X13	X12	X11	X10	X9	X8
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	U-0
X7	X6	X5	X4	X3	X2	X1	—
bit 7							bit 0

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

bit 15-1 **X<15:1>**: 多项式的项 X^n 的异或使能位
bit 0 未实现: 读为 0

20.0 高 / 低压检测（HLVD）

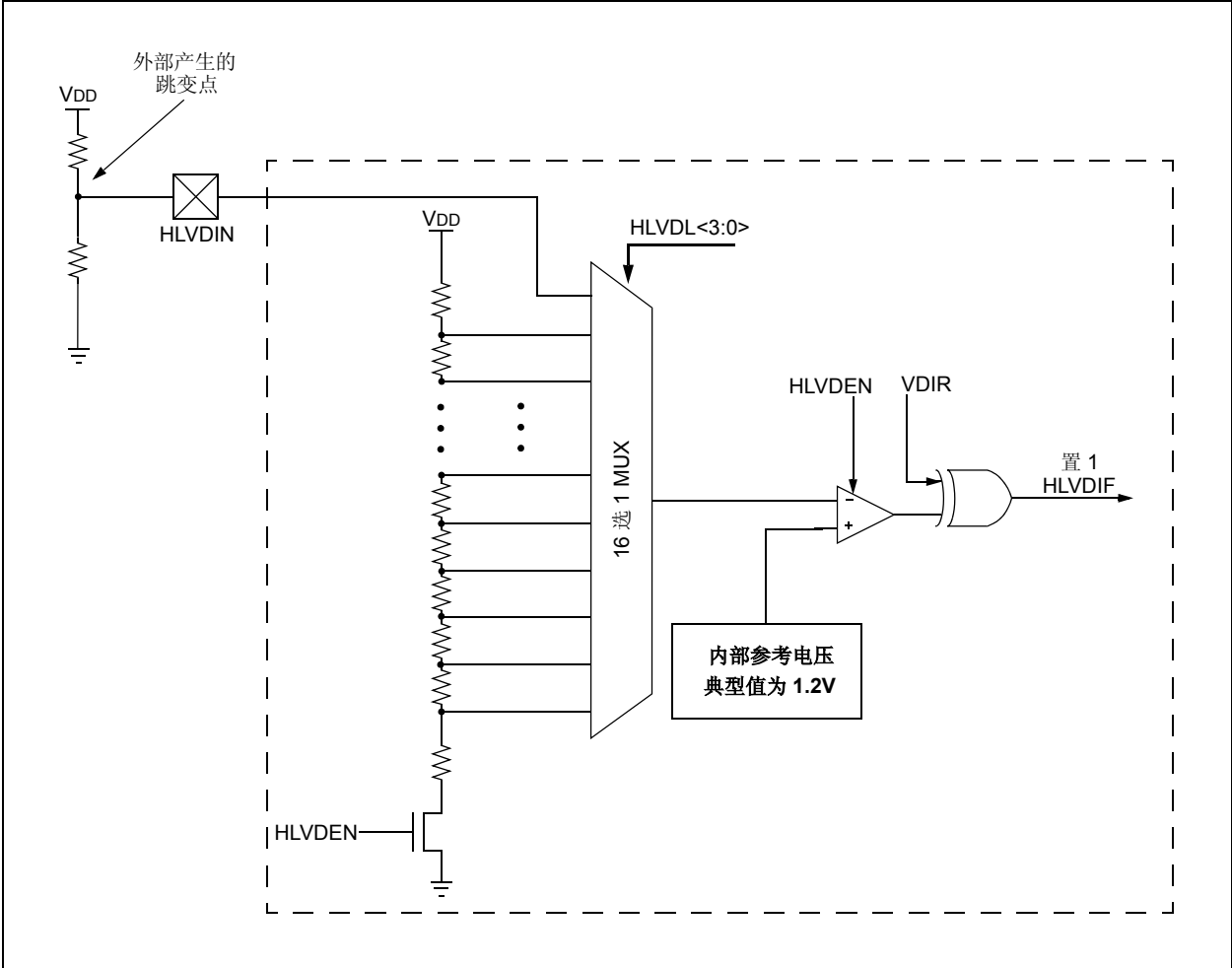
注： 本数据手册总结了PIC24F系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于高 / 低压检测的更多信息，请参见《PIC24F 系列参考手册》的“第36章 高度集成的可编程高 / 低压检测（HLVD）模块”（DS39725A_CN）。

高 / 低压检测（HLVD）模块是一个可编程的电路，它允许用户指定器件的电压跳变点和变化方向。

如果器件电压按照特定的变化方向相对于跳变点发生了偏离，就会将中断标志位置 1。如果允许了中断，程序将跳转到中断向量地址处执行，由软件响应该中断。

HLVD 控制寄存器（见寄存器 20-1）完全控制 HLVD 模块的工作。用户可通过软件控制该寄存器将电路“关闭”，从而使器件的电流消耗降至最低。

图 20-1： 高 / 低压检测（HLVD）模块框图



PIC24F16KA102 系列

寄存器 20-1: HLVDCON: 高 / 低压检测控制寄存器

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
HLVDEN	—	HLSIDL	—	—	—	—	—
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
VDIR	BGVST	IRVST	—	HLVDL3	HLVDL2	HLVDL1	HLVDL0
bit 7							bit 0

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

bit 15	HLVDEN: 高 / 低压检测电源使能位 1 = 使能 HLVD 0 = 禁止 HLVD
bit 14	未实现: 读为 0
bit 13	HLSIDL: HLVD 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 12-8	未实现: 读为 0
bit 7	VDIR: 电压变化方向选择位 1 = 当电压等于或超过跳变点 (HLVDL<3:0>) 时, 事件发生 0 = 当电压等于或低于跳变点 (HLVDL<3:0>) 时, 事件发生
bit 6	BGVST: 带隙电压稳定标志位 1 = 表示带隙电压稳定 0 = 表示带隙电压不稳定
bit 5	IRVST: 内部参考电压稳定标志位 1 = 表示内部参考电压稳定, 高压检测逻辑在检测到指定的电压范围时产生中断标志 0 = 表示内部参考电压不稳定, 高压检测逻辑在检测到指定的电压范围时不产生中断标志, 并且 HLVD 中断不应被允许
bit 4	未实现: 读为 0
bit 3-0	HLVDL<3:0>: 高 / 低压检测限制位 1111 = 使用外部模拟输入 (输入来自于 HLVDIN 引脚) 1110 = 跳变点 1 ⁽¹⁾ 1101 = 跳变点 2 ⁽¹⁾ 1100 = 跳变点 3 ⁽¹⁾ . . . 0000 = 跳变点 15 ⁽¹⁾

注 1: 关于实际跳变点的信息, 请参见第 28.0 节 “电气特性”。

21.0 10 位高速 A/D 转换器

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于 10 位高速 A/D 转换器的更多信息，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（Successive Approximation, SAR）转换
- 转换速度最高可达 500 ksp/s
- 9 个模拟输入引脚
- 外部参考电压输入引脚
- 内部带隙参考输入
- 自动通道扫描模式
- 可选的转换触发源
- 16 字的转换结果缓冲区
- 可选的缓冲区填充模式
- 4 个结果对齐选项
- 可在 CPU 休眠和空闲模式下工作

在所有 PIC24F16KA102 系列器件上，10 位 A/D 转换器都有 9 个模拟输入引脚，指定为 AN0 到 AN5 以及 AN10 到 AN12。此外，还有两个模拟输入引脚用于连接外部参考电压（VREF+ 和 VREF-）。这两个参考电压输入可以与其他模拟输入引脚共用。

图 21-1 给出了 A/D 转换器的框图。

要执行 A/D 转换：

1. 配置 A/D 模块：

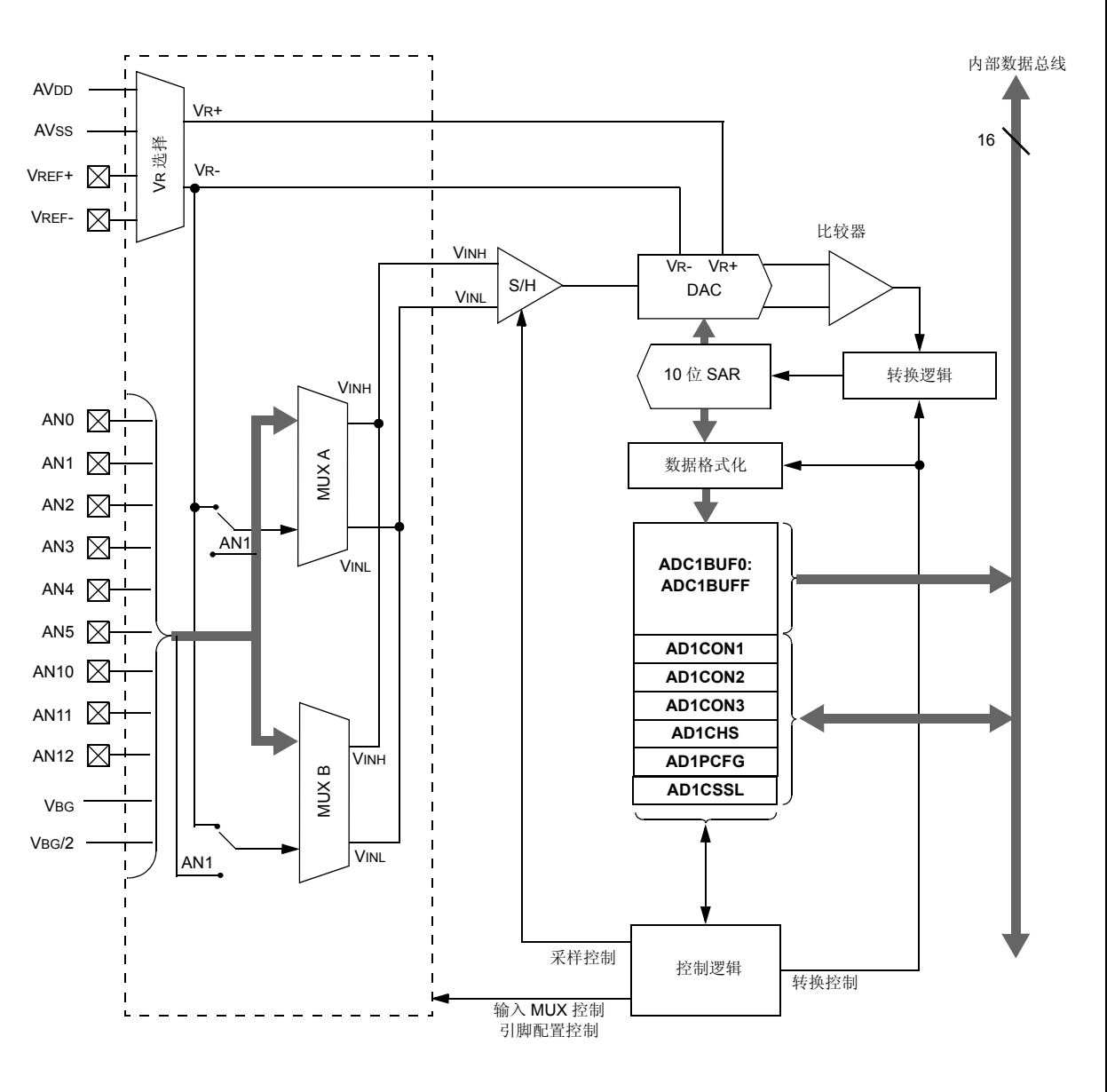
- a) 将端口引脚配置为模拟输入和 / 或选择带隙参考输入（AD1PCFG<15:13>和AD1PCFG<9:6>）。
- b) 选择参考电压源以匹配模拟输入的预期范围（AD1CON2<15:13>）。
- c) 选择模拟转换时钟以使期望的数据速率与处理器时钟匹配（AD1CON3<7:0>）。
- d) 选择适当的采样/转换序列（AD1CON1<7:5>和AD1CON3<12:8>）。
- e) 选择转换结果在缓冲区中的存储方式（AD1CON1<9:8>）。
- f) 选择中断频率（AD1CON2<5:2>）。
- g) 开启 A/D 模块（AD1CON1<15>）。

2. 配置 A/D 中断（如需要）：

- a) 清零 AD1IF 位。
- b) 选择 A/D 中断优先级。

PIC24F16KA102 系列

图 21-1: 10 位高速 A/D 转换器框图



寄存器 21-1: 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 ⁽¹⁾	—	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, HSC	R/W-0, HSC
SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE
bit 7						bit 0	

图注:	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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15 **ADON: A/D 工作模式位⁽¹⁾**
1 = A/D 转换器模块正在工作
0 = A/D 转换器关闭
- 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 **FORM<1:0>: 数据输出格式位**
11 = 有符号小数 (sddd dddd dd00 0000)
10 = 小数 (dddd dddd dd00 0000)
01 = 有符号整数 (ssss sssd dddd dddd)
00 = 整数 (0000 00dd dddd dddd)
- bit 7-5 **SSRC<2:0>: 转换触发源选择位**
111 = 由内部计数器结束采样并启动转换 (自动转换)
110 = 由 CTMU 事件结束采样并启动转换
101 = 保留
100 = 保留
011 = 保留
010 = 由 Timer3 比较结束采样并启动转换
001 = 由 INTO 引脚的有效跳变结束采样并启动转换
000 = 由清零 SAMP 位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 = 最后一次转换结束后立即开始采样; SAMP 位自动置 1
0 = SAMP 位置 1 时开始采样
- bit 1 **SAMP: A/D 采样使能位**
1 = A/D 采样 / 保持放大器正在对输入进行采样
0 = A/D 采样 / 保持放大器保持采样结果
- bit 0 **DONE: A/D 转换状态位**
1 = A/D 转换已完成
0 = A/D 转换未完成

注 1: 在 ADON 位清零后, ADC1BUF_n 寄存器的值将不会被保留。在禁止模块之前, 请先从缓冲区中读出转换值。

PIC24F16KA102 系列

寄存器 21-2: AD1CON2: A/D 控制寄存器 2

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	U-0	U-0
VCFG2	VCFG1	VCFG0	OFFCAL	—	CSCNA	—	—
bit 15							bit 8

R-0, HSC	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: U = 未实现位, 读为 0 HSC = 硬件置 1/ 清零位
R = 可读位 W = 可写位 r = 保留位
-n = POR 时的值 1 = 置 1 0 = 清零 x = 未知

bit 15-13 VCFG<2:0>: 参考电压配置位

VCFG<2:0>	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 OFFCAL: 偏移校准位

1 = 转换为获取偏移校准值
0 = 转换为获取实际输入值

bit 11 未实现: 读为 0

bit 10 CSCNA: MUX A 输入多路开关选择的 CH0+ S/H 输入的扫描输入设置位

1 = 扫描输入
0 = 不扫描输入

bit 9-8 未实现: 读为 0

bit 7 BUFS: 缓冲区填充状态位 (仅当 BUFM = 1 时有效)

1 = A/D 当前正在填充缓冲区 08-0F, 用户应访问 00-07 中的数据
0 = A/D 当前正在填充缓冲区 00-07, 用户应访问 08-0F 中的数据

bit 6 未实现: 读为 0

bit 5-2 SMPI<3:0>: 选择每次中断的采样 / 转换序列数的位

1111 = 每完成 16 个采样 / 转换序列时产生中断
1110 = 每完成 15 个采样 / 转换序列时产生中断

·
·
·

0001 = 每完成 2 个采样 / 转换序列时产生中断
0000 = 每完成 1 个采样 / 转换序列时产生中断

bit 1 BUFM: 缓冲区模式选择位

1 = 缓冲区配置为两个 8 字缓冲区 (ADC1BUF<n<15:8> 和 ADC1BUF<n<7:0>)
0 = 缓冲区配置为一个 16 字缓冲区 (ADC1BUF<n<15:0>)

bit 0 ALTS: 备用输入采样模式选择位

1 = 对于第一次采样, 使用 MUX A 输入多路开关设置, 然后对于所有后续采样, 在 MUX B 和 MUX A 输入多路开关设置之间交替
0 = 始终使用 MUX A 输入多路开关设置

寄存器 21-3: 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 = 保留位			
R = 可读位		W = 可写位		U = 未实现位, 读为 0			
-n = POR 时的值		1 = 置 1		0 = 清零		x = 未知	

- bit 15

ADRC: A/D 转换时钟源位
1 = A/D 内部的 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

SAMC<4:0>: 自动采样时间位
11111 = 31 TAD
.
.
.
00001 = 1 TAD
00000 = 0 TAD (不推荐)
- bit 7-0

ADCS<7:0>: A/D 转换时钟选择位
11111111 = 256 • Tcy
.
.
.
00000001 = 2 • Tcy
00000000 = Tcy

PIC24F16KA102 系列

寄存器 21-4: AD1CHS: A/D 输入选择寄存器

R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NB	—	—	—	CH0SB3	CH0SB2	CH0SB1	CH0SB0
bit 15				bit 8			

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NA	—	—	CH0SA4	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 = POR 时的值	1 = 置 1	0 = 清零
		x = 未知

bit 15	CH0NB: MUX B 多路开关设置的通道 0 反向输入选择位 1 = 通道 0 的反向输入为 AN1 0 = 通道 0 的反向输入为 VR-
bit 14-12	未实现: 读为 0
bit 11-8	CH0SB<3:0>: MUX B 多路开关设置的通道 0 正向输入选择位 1111 = 通道 0 的正向输入为带隙参考电压 (VBG) 1110 = 通道 0 的正向输入为带隙参考电压的一半 (VBG/2) 1101 = 未连接通道 (实际 ADC MUX 开关激活但输入悬空); 用于 CTMU 1100 = 通道 0 的正向输入为 AN12 1011 = 通道 0 的正向输入为 AN11 1010 = 通道 0 的正向输入为 AN10 1001 = 保留 1000 = 保留 0110 = AVDD 0110 = AVSS 0101 = 通道 0 的正向输入为 AN5 0100 = 通道 0 的正向输入为 AN4 0010 = 通道 0 的正向输入为 AN3 0010 = 通道 0 的正向输入为 AN2 0001 = 通道 0 的正向输入为 AN1 0000 = 通道 0 的正向输入为 AN0
bit 7	CH0NA: MUX A 多路开关设置的通道 0 反向输入选择位 1 = 通道 0 的反向输入为 AN1 0 = 通道 0 的反向输入为 VR-
bit 6-5	未实现: 读为 0
bit 4-0	CH0SA<4:0>: 采样多路开关 A 的通道 0 正向输入选择位 1111 = 通道 0 的正向输入为带隙参考电压 (VBG) 1110 = 通道 0 的正向输入为带隙参考电压的一半 (VBG/2) 1101 = 未连接通道 (实际 ADC MUX 开关激活但输入悬空); 用于 CTMU 1100 = 通道 0 的正向输入为 AN12 1011 = 通道 0 的正向输入为 AN11 1010 = 通道 0 的正向输入为 AN10 1001 = 保留 1000 = 保留 0110 = AVDD 0110 = AVSS 0101 = 通道 0 的正向输入为 AN5 0100 = 通道 0 的正向输入为 AN4 0010 = 通道 0 的正向输入为 AN3 0010 = 通道 0 的正向输入为 AN2 0001 = 通道 0 的正向输入为 AN1 0000 = 通道 0 的正向输入为 AN0

寄存器 21-5: AD1PCFG: A/D 端口配置寄存器

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	U-0	U-0
—	—	—	PCFG12	PCFG11	PCFG10	—	—
bit 15						bit 8	
U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	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 = POR 时的值 1 = 置 1 0 = 清零 x = 未知

- bit 15-13 **未实现:** 读为 0
- bit 12-10 **PCFG<12:10>:** 模拟输入引脚配置控制位
1 = 对应模拟通道的引脚配置为数字模式; 使能 I/O 端口读操作
0 = 引脚配置为模拟模式; 禁止 I/O 端口读操作; A/D 采样引脚电压
- bit 9-6 **未实现:** 读为 0
- bit 5-0 **PCFG<5:0>:** 模拟输入引脚配置控制位
1 = 对应模拟通道的引脚配置为数字模式; 使能 I/O 端口读操作
0 = 引脚配置为模拟模式; 禁止 I/O 端口读操作; A/D 采样引脚电压

寄存器 21-6: AD1CSSL: A/D 输入扫描选择寄存器 (低位字)

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	CSSL11	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 = POR 时的值 1 = 置 1 0 = 清零 x = 未知

- bit 15-0 **CSSL<15:0>:** A/D 输入引脚扫描选择位
1 = 为输入扫描选择的对应模拟通道
0 = 被输入扫描忽略的模拟通道

PIC24F16KA102 系列

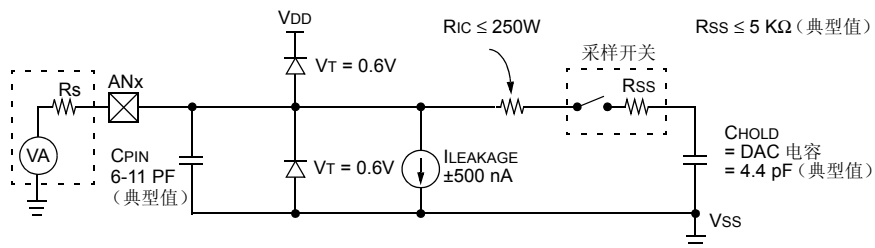
公式 21-1: A/D 转换时钟周期⁽¹⁾

$$ADCS = \frac{T_{AD}}{T_{CY}} - 1$$

$$T_{AD} = T_{CY} \cdot (ADCS + 1)$$

注 1: 基于 $T_{CY} = 2 \cdot T_{OSC}$; 打盹模式和 PLL 被禁止。

图 21-2: 10 位 A/D 转换器模拟输入模型

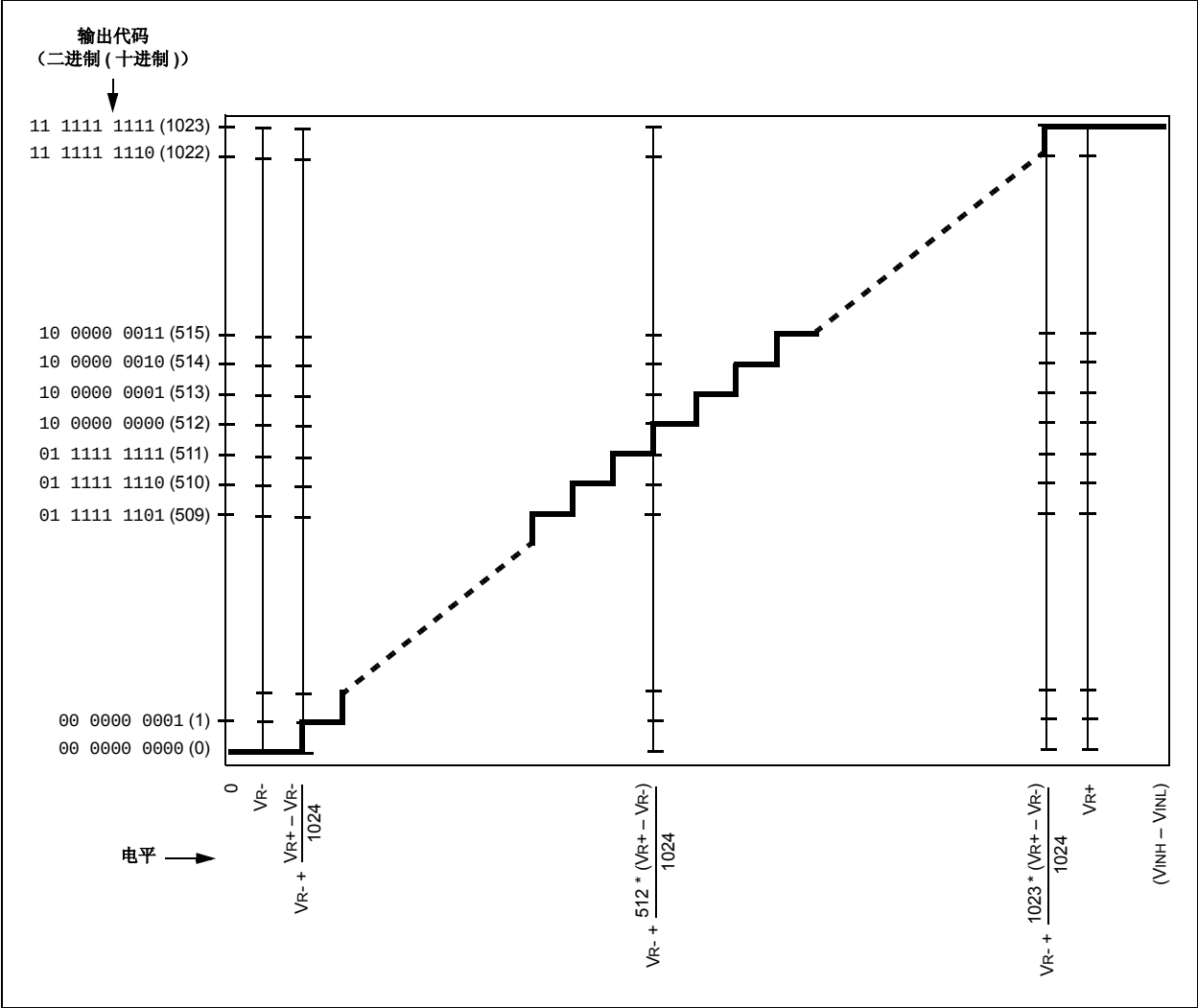


图注:

- CPIN = 输入电容
- VT = 门限电压
- ILEAKAGE = 由各连接点在引脚上产生的泄漏电流
- RIC = 片内走线等效电阻
- RSS = 采样开关电阻
- CHOLD = 采样/保持电容 (来自 DAC)

注: CPIN 值取决于器件封装, 未经测试。如果 $R_s \leq 5 \text{ k}\Omega$, 可忽略 CPIN 的影响。

图 21-3: A/D 传递函数



PIC24F16KA102 系列

注:

22.0 比较器模块

注： 本数据手册总结了PIC24F系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于比较器模块的更多信息，请参见《PIC24F 系列参考手册》的“第19章 比较器模块”（DS39710A_CN）。

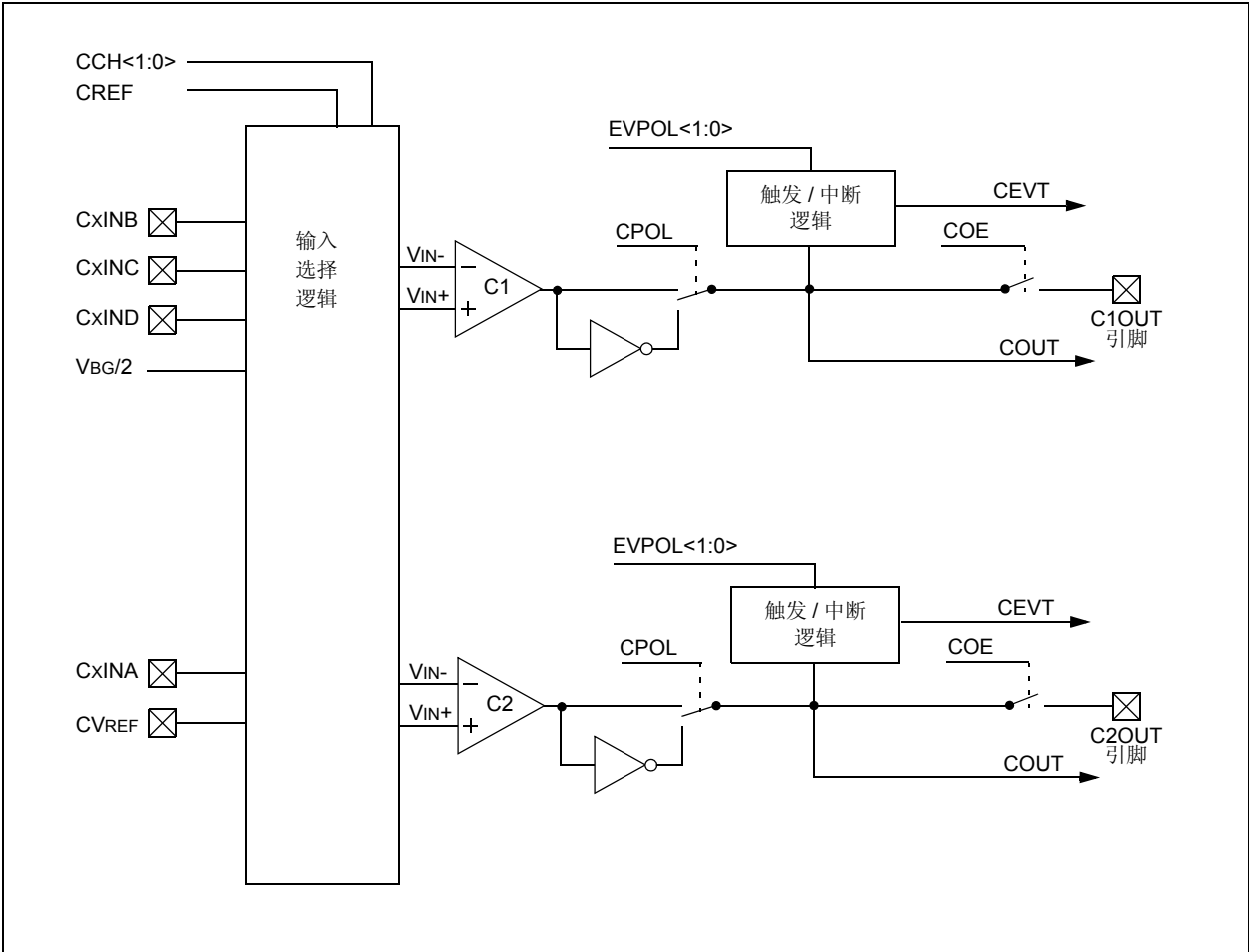
比较器模块提供两个双输入比较器。至比较器的输入可以配置为从4个外部模拟输入中任意选择，参考电压输入可以来自内部带隙参考电压幅值减半（V_{BG}/2）的信号或比较器参考电压发生器。

比较器输出可以直接连接到 CxOUT 引脚。当相应的 COE 等于 1 时，I/O 引脚逻辑使比较器的未同步输出在引脚上可用。

图 22-1 给出了此模块的简化框图。图 22-2 给出了各种可能的比较器配置。

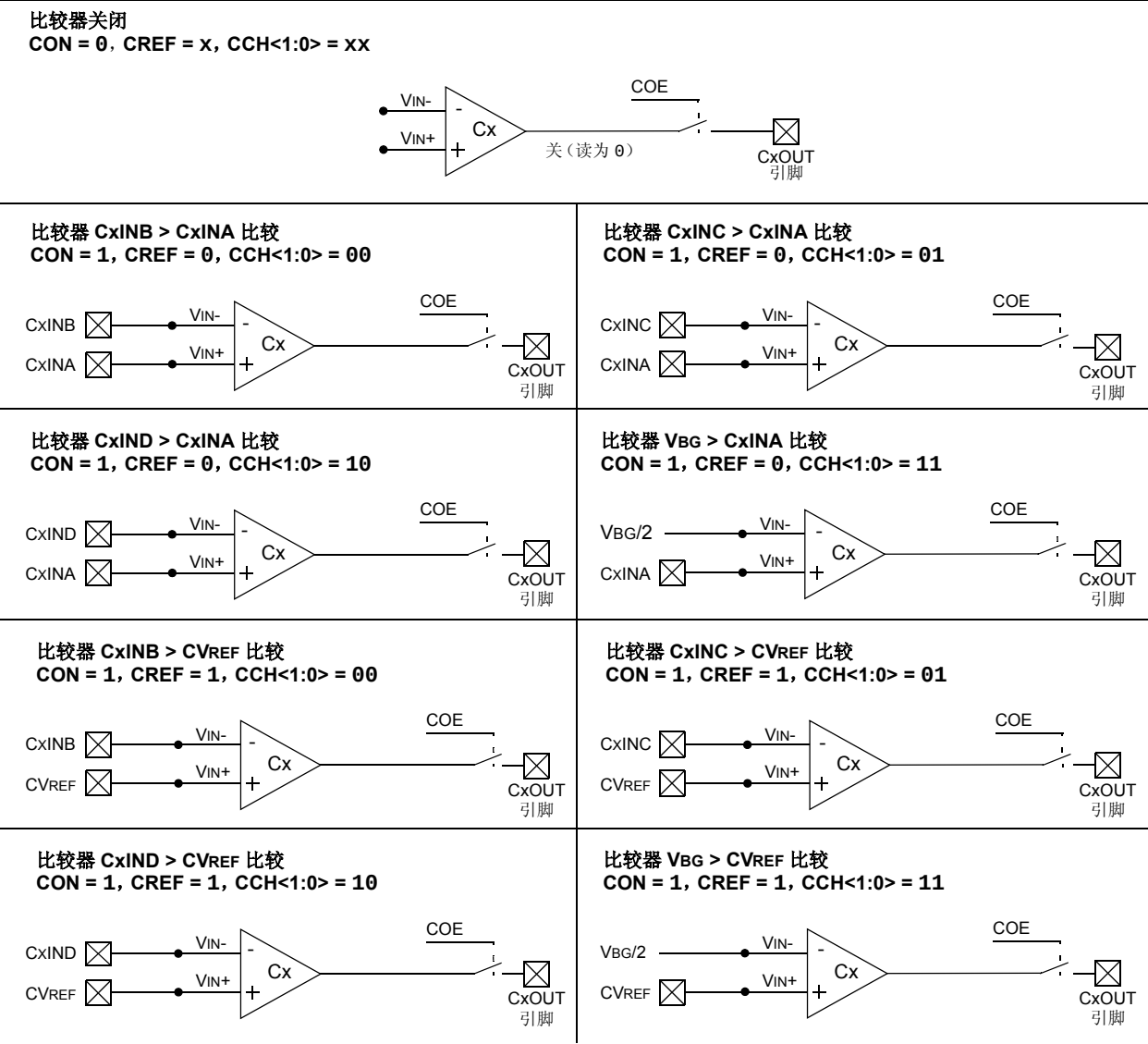
每个比较器都有自己的控制寄存器CMxCON（寄存器22-1），用于使能和配置其操作。所有 3 个比较器的输出和事件状态在 CMSTAT 寄存器（寄存器 22-2）中提供。

图 22-1： 比较器模块框图



PIC24F16KA102 系列

图 22-2: 各种比较器配置



寄存器 22-1: CMxCON: 比较器 x 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R-0
CON	COE	CPOL	CLPWR	—	—	CEVT	COUT
bit 15							bit 8
R/W-0	R/W-0	U-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EVPOL1	EVPOL0	—	CREF	—	—	CCH1	CCH0
bit 7							bit 0

图注:

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

- bit 15 **CON:** 比较器使能位
1 = 使能比较器
0 = 禁止比较器
- bit 14 **COE:** 比较器输出使能位
1 = 比较器输出出现在 CxOUT 引脚上
0 = 比较器输出仅在内部有效
- bit 13 **CPOL:** 比较器输出极性选择位
1 = 比较器输出反相
0 = 比较器输出不反相
- bit 12 **CLPWR:** 比较器低功耗模式选择位
1 = 比较器在低功耗模式下工作
0 = 比较器不在低功耗模式下工作
- bit 11-10 **未实现:** 读为 0
- bit 9 **CEVT:** 比较器事件位
1 = 已发生由 **EVPOL<1:0>** 定义的比较器事件; 后续的触发和中断被禁止, 直到该位被清零
0 = 未发生比较器事件
- bit 8 **COUT:** 比较器输出位
当 **CPOL = 0** 时:
1 = $V_{IN+} > V_{IN-}$
0 = $V_{IN+} < V_{IN-}$
当 **CPOL = 1** 时:
1 = $V_{IN+} < V_{IN-}$
0 = $V_{IN+} > V_{IN-}$
- bit 7-6 **EVPOL<1:0>:** 触发 / 事件 / 中断极性选择位
11 = 在比较器输出发生任何变化时产生触发 / 事件 / 中断 (当 **CEVT = 0** 时)
10 = 在比较器输出发生跳变时产生触发 / 事件 / 中断:
 如果 **CPOL = 0** (极性不反相):
 仅限从高至低跳变。
 如果 **CPOL = 1** (极性反相):
 仅限从低至高跳变。
01 = 在比较器输出发生跳变时产生触发 / 事件 / 中断:
 如果 **CPOL = 0** (极性不反相):
 仅限从低至高跳变。
 如果 **CPOL = 1** (极性反相):
 仅限从高至低跳变。
00 = 禁止产生触发 / 事件 / 中断信号
- bit 5 **未实现:** 读为 0
- bit 4 **CREF:** 比较器参考电压选择位 (同相输入)
1 = 同相输入连接到内部 **CVREF** 电压
0 = 同相输入连接到 CxINA 引脚
- bit 3-2 **未实现:** 读为 0
- bit 1-0 **CCH<1:0>:** 比较器通道选择位
11 = 比较器的反相输入连接到 VBG/2
10 = 比较器的反相输入连接到 CxIND 引脚
01 = 比较器的反相输入连接到 CxINC 引脚
00 = 比较器的反相输入连接至 CxINB 引脚

PIC24F16KA102 系列

寄存器 22-2: CMSTAT: 比较器模块状态寄存器

R/W-0	U-0	U-0	U-0	U-0	U-0	R-0, HSC	R-0, HSC
CMIDL	—	—	—	—	—	C2EVT	C1EVT
bit 15						bit 8	
U-0	U-0	U-0	U-0	U-0	U-0	R-0, HSC	R-0, HSC
—	—	—	—	—	—	C2OUT	C1OUT
bit 7						bit 0	

图注:	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 = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 15	CMIDL: 比较器空闲模式停止位 1 = 器件进入空闲模式后所有比较器停止工作 0 = 在空闲模式下所有已使能的比较器继续工作
bit 14-10	未实现: 读为 0
bit 9	C2EVT: 比较器 2 事件状态位 (只读) 显示比较器 2 当前的事件状态 (CM2CON<9>)。
bit 8	C1EVT: 比较器 1 事件状态位 (只读) 显示比较器 1 当前的事件状态 (CM1CON<9>)。
bit 7-2	未实现: 读为 0
bit 1	C2OUT: 比较器 2 输出状态位 (只读) 显示比较器 2 当前的输出状态 (CM2CON<8>)。
bit 0	C1OUT: 比较器 1 输出状态位 (只读) 显示比较器 1 当前的输出状态 (CM1CON<8>)。

23.0 比较器参考电压

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于比较器参考电压的更多信息，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）。

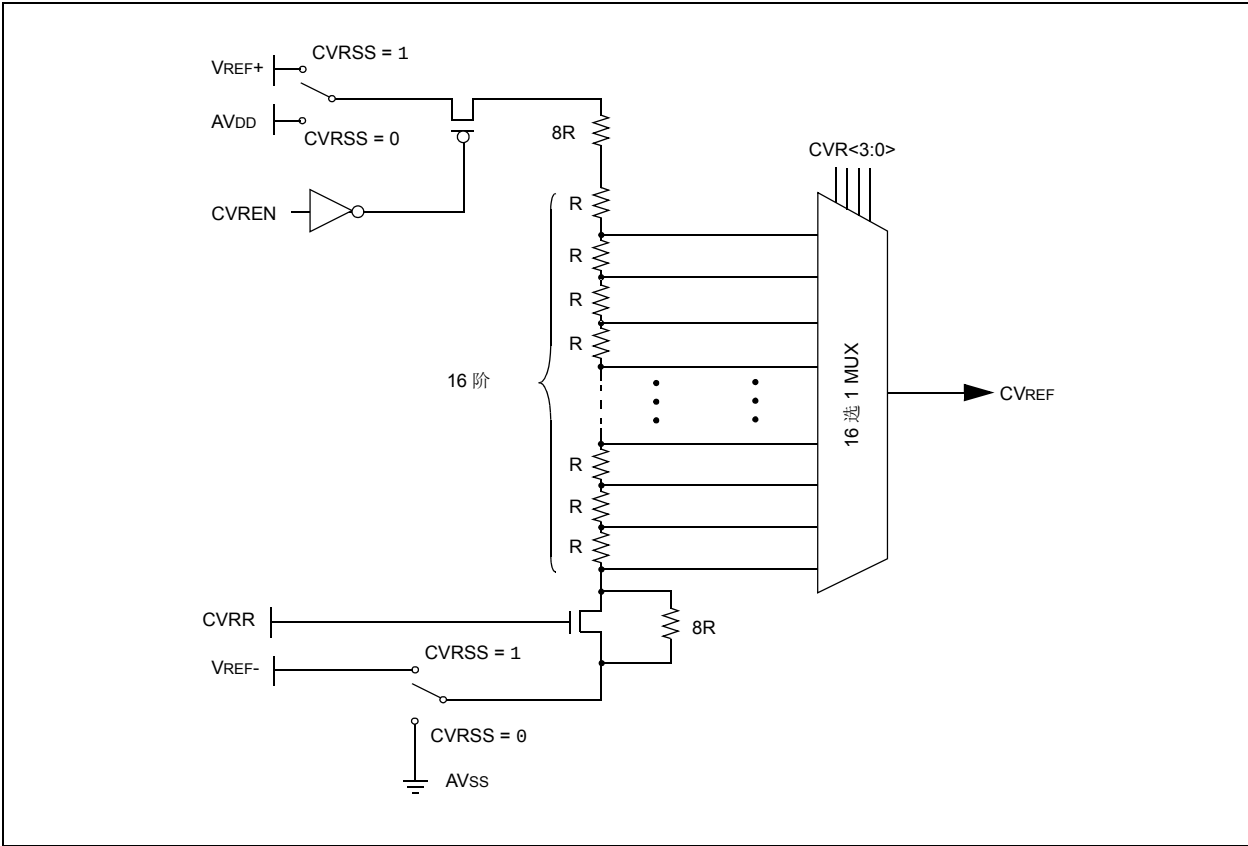
23.1 配置比较器参考电压

比较器参考电压模块是通过 CVRCON 寄存器（寄存器 23-1）来控制的。比较器参考电压模块提供两种范围的输出电压，每种范围都具有 16 个不同的电平。通过 CVRR 位（CVRCON<5>）选择输出电压的范围。这两种范围的主要区别在于其电压值之间的步长不同（其中一种范围可提供较高的分辨率），该步长通过 CVREF 选择位（CVR<3:0>）选择。

比较器参考电压模块的电压源可以来自 VDD 和 VSS，也可以来自外部 VREF+ 和 VREF-。电压源通过 CVRSS 位（CVRCON<4>）选择。

在更改 CVREF 输出值时，必须考虑比较器参考电压的稳定时间。

图 23-1： 比较器参考电压框图



PIC24F16KA102 系列

寄存器 23-1: 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 = POR 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 = CVREF 电路上电 0 = CVREF 电路断电
bit 6	CVROE: 比较器 VREF 输出使能位 1 = CVREF 电平从 CVREF 引脚输出 0 = CVREF 电平从 CVREF 引脚断开
bit 5	CVRR: 比较器 VREF 范围选择位 1 = CVRSRC 范围应从 0 到 0.625 CVRSRC, 步长为 CVRSRC/24 0 = CVRSRC 范围应从 0.25 到 0.719 CVRSRC, 步长为 CVRSRC/32
bit 4	CVRSS: 比较器 VREF 源选择位 1 = 比较器参考电压源, CVRSRC = VREF+ – VREF- 0 = 比较器参考电压源, CVRSRC = AVDD – AVSS
bit 3-0	CVR3:CVR0: 比较器 VREF 值选择位, $0 \leq \text{CVR}<3:0> \leq 15$ 当 CVRR = 1 且 CVRSS = 0 时: $\text{CVREF} = (\text{CVR}<3:0>/24) * (\text{CVRSRC})$ 当 CVRR = 0 且 CVRSS = 0 时: $\text{CVREF} = 1/4 (\text{CVRSRC}) + (\text{CVR}<3:0>/32) * (\text{CVRSRC})$ 当 CVRR = 1 且 CVRSS = 1 时: $\text{CVREF} = ((\text{CVR}<3:0>/24) * (\text{CVRSRC})) + \text{VREF-}$ 当 CVRR = 0 且 CVRSS = 1 时: $\text{CVREF} = (1/4 (\text{CVRSRC}) + (\text{CVR}<3:0>/32) * (\text{CVRSRC})) + \text{VREF-}$

24.0 充电时间测量单元（CTMU）

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于充电时间测量单元的更多信息，请参见《PIC24F 系列参考手册》的“第 11 章 CTMU”（DS39724A_CN）。

充电时间测量单元（CTMU）是一个灵活的模拟模块，它提供电荷测量、脉冲源之间的精确时间差测量，以及异步脉冲生成。它的主要特性包括：

- 4 个边沿输入触发源
- 每个边沿源的极性控制
- 边沿序列控制
- 控制对边沿的响应
- 1 纳秒的时间测量分辨率
- 适合测量电容的精确电流源

CTMU 可与其他片上模拟模块一起，用于精确测量时间、电容、电容的相对变化，或生成独立于系统时钟的输出脉冲。CTMU 模块是与电容式触摸传感器接口的理想选择。

CTMU 通过两个寄存器 CTMUCON 和 CTMUICON 进行控制。CTMUCON 使能该模块，控制边沿源选择、边沿源极性选择和边沿序列的组成。CTMUICON 寄存器选择电流源范围并微调电流。

24.1 测量电容

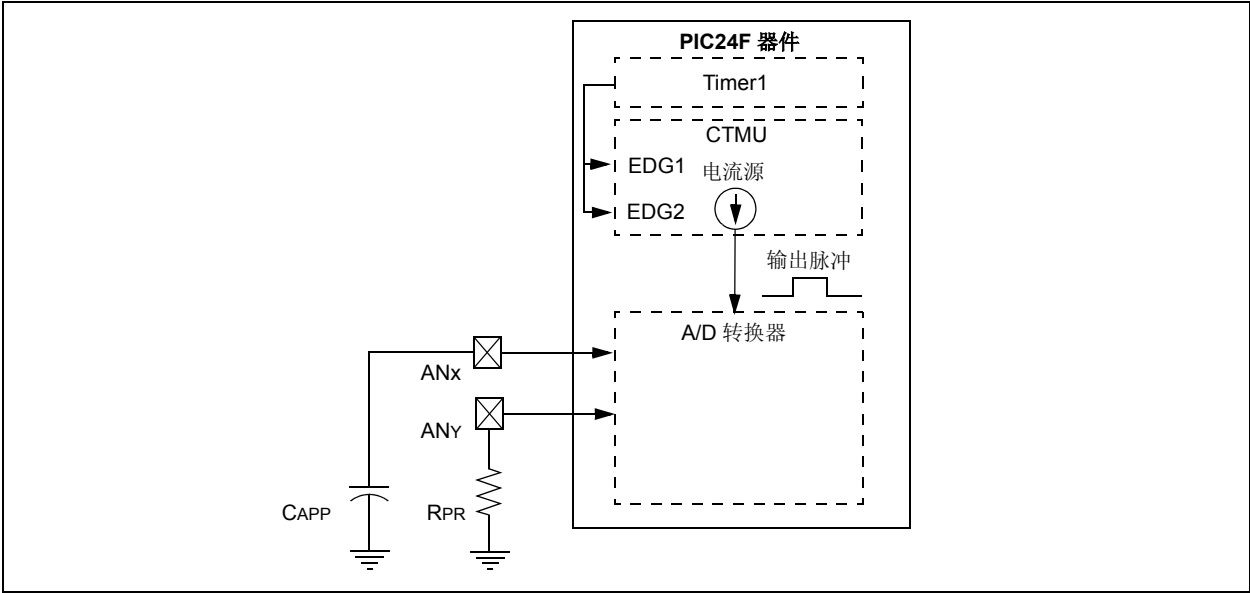
CTMU 模块通过生成宽度和两个独立输入通道上的边沿事件间的时间差相等的输出脉冲，来测量电容。两个输入通道的脉冲边沿事件都可以从 4 个来源选择：两个内部外设模块（OC1 和 Timer1）和两个外部引脚（CTEDG1 和 CTEDG2）。该脉冲和该模块的精确电流源一起使用，可根据以下关系计算电容：

$$C = I \cdot \frac{dV}{dT}$$

测量电容时，A/D 转换器在 CTMU 输出脉冲结束后对其某个输入通道上的外部电容（CAPP）进行采样。一个高精度电阻（RPR）对第二个 A/D 通道上电流源进行校准。脉冲结束后，转换器将测量电容上的电压。电容的实际计算在软件中由应用程序执行。

图 24-1 显示了用于电容测量的外部连接，以及该应用中 CTMU 和 A/D 模块关系如何。该示例还显示了来自 Timer1 的边沿事件，但使用外部边沿源的其他配置也是可能的。用 CTMU 模块测量电容和时间的详细讨论在《PIC24F 系列参考手册》中提供。

图 24-1： 电容测量的典型连接和内部配置



PIC24F16KA102 系列

24.2 测量时间

对脉冲宽度的时间测量也可类似地执行，需要使用 A/D 模块的内部电容（CAD）和用于校准电流的高精度电阻。图 24-2 显示了用于时间测量的外部连接，以及该应用中 CTMU 和 A/D 模块关系如何。该示例还显示了来自外部 CTEDG 引脚的两个边沿事件，但使用内部边沿源的其他配置也是可能的。

24.3 脉冲产生和延时

CTMU 模块也可产生边沿与器件的系统时钟异步的输出脉冲。更明确地说，它可以产生距离边沿事件输入到该模块的延时可编程的脉冲。

通过将 TGEN 位（CTMUCON<12>）置 1 配置该模块在产生脉冲前应用延时，内部电流源连接到比较器 2 的 B 输入。电容（CDELAY）连接到比较器 2 的引脚 C2INB，比较器参考电压 CVREF 连接到 C2INA 将其设置为特定跳变点。检测到边沿事件时，该模块开始对 CDELAY 充电。CDELAY 充电达到 CVREF 跳变点时，在 CTPLS 上输出脉冲。脉冲延时的长度由 CDELAY 和 CVREF 跳变点的值决定。

图 24-3 显示了脉冲产生的外部连接，以及所需的不同模拟模块间的关系。CTEDG1 显示为输入脉冲源时，其他选项可用。用 CTMU 模块产生脉冲的详细讨论在《PIC24F 系列参考手册》中提供。

图 24-2: 测量时间的典型连接和内部配置

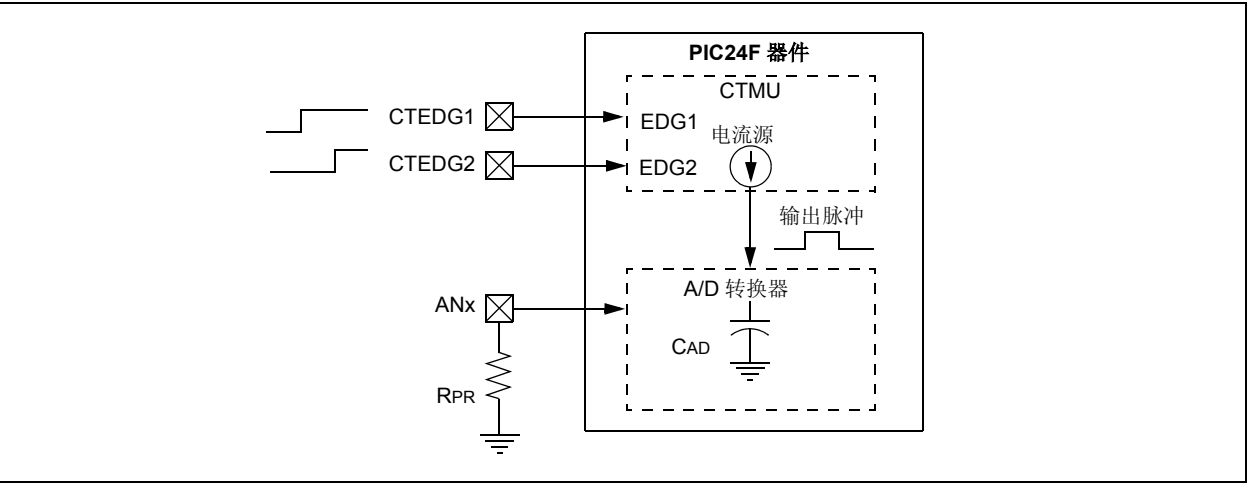
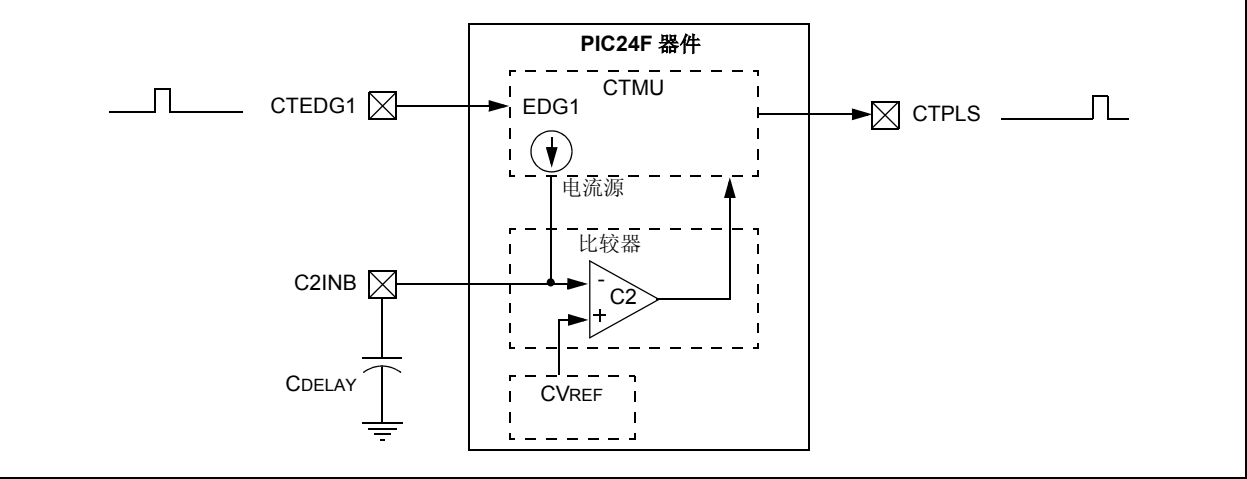


图 24-3: 产生脉冲延时的典型连接和内部配置



寄存器 24-1: CTMUCON: CTMU 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
CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG
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
EDG2POL	EDG2SEL1	EDG2SEL0	EDG1POL	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT
bit 7							bit 0

图注:

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

- bit 15 **CTMUEN:** CTMU 使能位
1 = 使能模块
0 = 禁止模块
- bit 14 **未实现:** 读为 0
- bit 13 **CTMUSIDL:** 空闲模式停止位
1 = 当器件进入空闲模式时, 模块停止工作
0 = 在空闲模式下模块继续工作
- bit 12 **TGEN:** 延时产生使能位
1 = 使能边沿延时产生
0 = 禁止边沿延时产生
- bit 10 **EDGEN:** 边沿使能位
1 = 未阻止边沿
0 = 阻止边沿
- bit 10 **EDGSEQEN:** 边沿序列使能位
1 = 边沿 1 事件必须在边沿 2 事件发生前发生
0 = 无需边沿序列
- bit 9 **IDISSEN:** 模拟电流源控制位
1 = 模拟电流源输出接地
0 = 模拟电流源输出未接地
- bit 8 **CTTRIG:** 触发器控制位
1 = 使能触发器输出
0 = 禁止触发器输出
- bit 7 **EDG2POL:** 边沿 2 极性选择位
1 = 边沿 2 设定为正边沿响应
0 = 边沿 2 设定为负边沿响应
- bit 6-5 **EDG2SEL<1:0>:** 边沿 2 源选择位
11 = CTED1 引脚
10 = CTED2 引脚
01 = OC1 模块
00 = Timer1 模块
- bit 4 **EDG1POL:** 边沿 1 极性选择位
1 = 边沿 1 设定为正边沿响应
0 = 边沿 1 设定为负边沿响应

PIC24F16KA102 系列

寄存器 24-1: CTMUCON: CTMU 控制寄存器 (续)

bit 3-2 **EDG1SEL<1:0>**: 边沿 1 源选择位

11 = CTED1 引脚

10 = CTED2 引脚

01 = OC1 模块

00 = Timer1 模块

bit 1 **EDG2STAT**: 边沿 2 状态位

1 = 已发生边沿 2 事件

0 = 未发生边沿 2 事件

bit 0 **EDG1STAT**: 边沿 1 状态位

1 = 已发生边沿 1 事件

0 = 未发生边沿 1 事件

寄存器 24-2: CTMUICON: CTMU 电流控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0
bit 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 = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-10 **ITRIM<5:0>**: 电流源微调位

011111 = 对标称电流的最大正向调整

011110

.

.

.

000001 = 对标称电流的最小正向调整

000000 = IRNG<1:0> 指定的标称电流输出

111111 = 对标称电流的最小负向调整

.

.

.

100010

100001 = 对标称电流的最大负向调整

bit 9-8 **IRNG<1:0>**: 电流源范围选择位

11 = 100 × 基本电流

10 = 10 × 基本电流

01 = 基本电流 (标称值为 0.55 μA)

00 = 禁止电流源

bit 7-0 **未实现**: 读为 0

25.0 特殊功能

注： 本数据手册总结了 PIC24F 系列器件的特性。但是不应把本手册当作无所不包的参考手册来使用。关于看门狗定时器、高级器件集成和编程诊断的更多信息，请参见《PIC24F 系列参考手册》的以下各章节：

- 第 9 章 看门狗定时器（WDT）（DS39697A_CN）
- 第 36 章 高度集成的可编程高 / 低电压检测（HLVD）模块（DS39725A_CN）
- 第 33 章 编程和诊断（DS39716A_CN）

PIC24F16KA102 系列器件具有几项特殊的功能旨在最大限度地提高应用的灵活性和可靠性，并通过减少外部元件的使用将成本降至最低。提供的特殊功能包括：

- 灵活的配置
- 看门狗定时器（WDT）
- 代码保护
- 在线串行编程（ICSP™）
- 在线仿真

25.1 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。这些位被映射到程序存储器以 F80000h 开始的单元中。表 25-1 给出了完整的列表。从寄存器 25-1 到寄存器 25-8 详细解释了各配置位的不同功能。

地址 F80000h 超出了用户程序存储空间。事实上，它属于只能使用表读和表写访问的配置存储空间（800000h-FFFFFFh）。

表 25-1: 配置寄存器地址

配置寄存器	地址
FBS	F80000
FGS	F80004
FOSCSEL	F80006
FOSC	F80008
FWDT	F8000A
FPOR	F8000C
FICD	F8000E
FDS	F80010

寄存器 25-1: FBS: 引导段配置寄存器

U-0	U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	—	BSS2	BSS1	BSS0	BWRP
bit 7				bit 0			

图注：

R = 可读位

-n = POR 时的值

W = 可写位

1 = 置 1

U = 未实现位，读为 0

0 = 清零

x = 未知

- bit 7-4

未实现：读为 0
- bit 3-1

BSS<2:0>: 引导段程序闪存代码保护位

111 = 无引导程序闪存段

011 = 保留

110 = 标准安全性，引导程序闪存段开始于 200h，结束于 000AFEh

010 = 高安全性，引导程序闪存段开始于 200h，结束于 000AFEh

101 = 标准安全性，引导程序闪存段开始于 200h，结束于 0015FEh⁽¹⁾

001 = 高安全性，引导程序闪存段开始于 200h，结束于 0015FEh⁽¹⁾

100 = 保留

000 = 保留
- bit 0

BWRP: 引导段程序闪存写保护位

1 = 引导段可写

0 = 引导段被写保护

注 1: 此选择不应在 PIC24F08KA1XX 器件中使用。

PIC24F16KA102 系列

寄存器 25-2: FGS: 通用段配置寄存器

U-0	U-0	U-0	U-0	U-0	U-0	R/C-1	R/C-1
—	—	—	—	—	—	GSS0	GWRP
bit 7						bit 0	

图注:

R = 可读位
-n = POR 时的值
C = 可清零位
1 = 置 1
U = 未实现位, 读为 0
0 = 清零
x = 未知

bit 7-2 未实现: 读为 0
bit 1 **GSS0**: 通用段代码闪存代码保护位
1 = 无保护
0 = 使能标准安全性
bit 0 **GWRP**: 通用段代码闪存写保护位
1 = 通用段可写
0 = 通用段被写保护

寄存器 25-3: FOSCSEL: 振荡器选择配置寄存器

R/P-1	U-0	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1
IESO	—	—	—	—	FNOSC2	FNOSC1	FNOSC0
bit 7						bit 0	

图注:

R = 可读位
-n = POR 时的值
P = 可编程位
1 = 置 1
U = 未实现位, 读为 0
0 = 清零
x = 未知

bit 7 **IESO**: 内 / 外部切换位
1 = 使能内 / 外部切换模式 (使能双速启动)
0 = 禁止内 / 外部切换模式 (禁止双速启动)
bit 6-3 未实现: 读为 0
bit 2-0 **FNOSC<2:0>**: 振荡器选择位
000 = 快速 RC 振荡器 (FRC)
001 = 带 PLL 模块、N 分频的快速 RC 振荡器 (FRCDIV+PLL)
010 = 主振荡器 (XT、HS 和 EC)
011 = 带 PLL 模块的主振荡器 (HS+PLL 和 EC+PLL)
100 = 辅助振荡器 (SOSC)
101 = 低功耗 RC 振荡器 (LPRC)
110 = 带 N 分频的 500 kHz 低功耗 FRC 振荡器 (LPFRCDIV)
111 = 带 N 分频的 8 MHz FRC 振荡器 (FRCDIV)

寄存器 25-4: FOSC: 振荡器配置寄存器

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
FCKSM1	FCKSM0	SOSCSEL	POSCFREQ1	POSCFREQ0	OSCIOFNC	POSCMD1	POSCMD0
bit 7							bit 0

图注:			
R = 可读位	P = 可编程位	U = 未实现位, 读为 0	
-n = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 7-6	FCKSM<1:0> : 时钟切换和监视器选择配置位 1x = 禁止时钟切换, 禁止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	SOSCSEL : 辅助振荡器选择位 1 = 辅助振荡器配置为高功耗工作 0 = 辅助振荡器配置为低功耗工作
bit 4-3	POSCFREQ<1:0> : 主振荡器频率范围配置位 11 = 主振荡器 / 外部时钟输入频率大于 8 MHz 10 = 主振荡器 / 外部时钟输入频率介于 100 kHz 和 8 MHz 之间 01 = 主振荡器 / 外部时钟输入频率小于 100 kHz 00 = 保留; 不要使用
bit 2	OSCIOFNC : CLKO 使能配置位 1 = CLKO 输出信号在 OSCO 引脚上有效; 必须禁止主振荡器或配置为外部时钟模式 (EC) 以使 CLKO 有效 (POSCMD<1:0> = 11 或 00) 0 = 禁止 CLKO 输出
bit 1-0	POSCMD<1:0> : 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 = 选择外部时钟模式

PIC24F16KA102 系列

寄存器 25-5: FWDTP: 看门狗定时器配置寄存器

R/P-1	R/P-1	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
FWDTEN	WINDIS	—	FWPSA	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

图注:			
R = 可读位	P = 可编程位	U = 未实现位, 读为 0	
-n = POR 时的值	1 = 置 1	0 = 清零	x = 未知

bit 7	FWDTEN: 看门狗定时器使能位 1 = 使能 WDT 0 = 禁止 WDT (由 SWDTEN 位控制)
bit 6	WINDIS: 窗口看门狗定时器禁止位 1 = 选择标准 WDT; 禁止窗口 WDT 0 = 使能窗口 WDT
bit 5	未实现: 读为 0
bit 4	FWPSA: WDT 预分频比位 1 = WDT 预分频比 1:128 0 = WDT 预分频比 1:32
bit 3-0	WDTPS<3:0>: 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

寄存器 25-6: FPOR: 复位配置寄存器

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	U-0	R/P-1	R/P-1
MCLRE ⁽²⁾	BORV1 ⁽³⁾	BORV0 ⁽³⁾	I2C1SEL ⁽¹⁾	PWRTEN	—	BOREN1	BOREN0
bit 7							bit 0

图注:

R = 可读位 P = 可编程位 U = 未实现位, 读为 0
 -n = POR 时的值 1 = 置 1 0 = 清零 x = 未知

- bit 7 **MCLRE:** $\overline{\text{MCLR}}$ 引脚使能位 ⁽²⁾
 1 = 使能 MCLR 引脚; 禁止 RA5 输入引脚
 0 = 使能 RA5 输入引脚; 禁止 MCLR
- bit 6-5 **BORV<1:0>:** 欠压复位使能位 ⁽³⁾
 11 = 欠压复位设置为最低电压
 10 = 欠压复位
 01 = 欠压复位设置为最高电压
 00 = 低功耗欠压复位发生在大约 2.0V
- bit 4 **I2C1SEL:** 备用 I2C1 引脚映射位 ⁽¹⁾
 0 = SCL1/SDA1 引脚的备用位置
 1 = SCL1/SDA1 引脚的默认位置
- bit 3 **PWRTEN:** 上电延时定时器使能位
 0 = 禁止 PWRT
 1 = 使能 PWRT
- bit 2 **未实现:** 读为 0
- bit 1-0 **BOREN<1:0>:** 欠压复位使能位
 11 = 用硬件使能欠压复位; 禁止 SBOREN 位
 10 = 仅当器件工作时使能欠压复位, 休眠模式下则禁止; 禁止 SBOREN 位
 01 = 欠压复位由 SBOREN 位设置控制
 00 = 用硬件禁止欠压复位; 禁止 SBOREN 位

- 注 1: 仅适用于 28 引脚器件。
 2: 只有使用基于 VPP 的方式进入 ICSP™ 模式时, 才能更改 MCLRE 熔丝。这可以防止用户意外地锁定器件, 无法进入低压测试。
 3: 请参见 BOR 电压的电气规范。

寄存器 25-7: FICD: 在线调试器配置寄存器

R/P-1	U-0	U-0	U-0	U-0	U-0	R/P-1	R/P-1
DEBUG	—	—	—	—	—	FICD1	FICD0
bit 7							bit 0

图注:

R = 可读位 P = 可编程位 U = 未实现位, 读为 0
 -n = POR 时的值 1 = 置 1 0 = 清零 x = 未知

- bit 7 **DEBUG:** 后台调试器使能位
 1 = 禁止后台调试器
 0 = 使能后台调试器功能
- bit 6-2 **未实现:** 读为 0
- bit 1-0 **FICD<1:0>:** ICD 引脚选择位
 11 = PGC1/PGD1 用于编程和调试器件
 10 = PGC2/PGD2 用于编程和调试器件
 01 = PGC3/PGD3 用于编程和调试器件
 00 = 保留; 不要使用

PIC24F16KA102 系列

寄存器 25-8: FDS: 深度休眠配置寄存器

R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
DSWDTEN	DSLBPBOR	RTCKSEL	DSWCKSEL	DSWDTPS3	DSWDTPS2	DSWDTPS1	DSWDTPS0
bit 7							bit 0

图注:

R = 可读位	P = 可编程位	U = 未实现位, 读为 0
-n = POR 时的值	1 = 置 1	0 = 清零
		x = 未知

bit 7	DSWDTEN: 深度休眠看门狗定时器使能位 1 = 使能 DSWDT 0 = 禁止 DSWDT
bit 6	DSLBPBOR: 深度休眠 / 低功耗 BOR 使能位 (不影响非深度休眠模式下的工作) 1 = 在深度休眠模式下使能深度休眠 BOR 0 = 在深度休眠模式下禁止深度休眠 BOR
bit 5	RTCKSEL: RTCC 参考时钟选择位 1 = RTCC 使用 SOSC 作为参考时钟 0 = RTCC 使用 LPRC 作为参考时钟
bit 4	DSWCKSEL: DSWDT 参考时钟选择位 1 = DSWDT 使用 LPRC 作为参考时钟 0 = DSWDT 使用 SOSC 作为参考时钟
bit 3-0	DSWDTPS<3:0>: 深度休眠看门狗定时器后分频比选择位 DSWDT 预分频比为 32; 这产生一个约为 1 ms 的基本时间单位。 1111 = 1:2,147,483,648 (25.7 天) 标称值 1110 = 1:536,870,912 (6.4 天) 标称值 1101 = 1:134,217,728 (38.5 小时) 标称值 1100 = 1:33,554,432 (9.6 小时) 标称值 1011 = 1:8,388,608 (2.4 小时) 标称值 1010 = 1:2,097,152 (36 分钟) 标称值 1001 = 1:524,288 (9 分钟) 标称值 1000 = 1:131,072 (135 秒) 标称值 0111 = 1:32,768 (34 秒) 标称值 0110 = 1:8,192 (8.5 秒) 标称值 0101 = 1:2,048 (2.1 秒) 标称值 0100 = 1:512 (528 ms) 标称值 0011 = 1:128 (132 ms) 标称值 0010 = 1:32 (33 ms) 标称值 0001 = 1:8 (8.3 ms) 标称值 0000 = 1:2 (2.1 ms) 标称值

PIC24F16KA102 系列

寄存器 25-9: DEVID: 器件 ID 寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 23				bit 16			
R	R	R	R	R	R	R	R
FAMID7	FAMID6	FAMID5	FAMID4	FAMID3	FAMID2	FAMID1	FAMID0
bit 15				bit 8			
R	R	R	R	R	R	R	R
DEV7	DEV6	DEV5	DEV4	DEV3	DEV2	DEV1	DEV0
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 23-16 未实现: 读为 0

bit 15-8 **FAMID<7:0>**: 器件系列标识符位
00001011 = PIC24F16KA102 系列

bit 7-0 **DEV<7:0>**: 各个器件的标识符位
00000011 = PIC24F16KA102
00001010 = PIC24F08KA102
00000001 = PIC24F16KA101
00001000 = PIC24F08KA101

寄存器 25-10: DEVREV: 器件版本寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 23				bit 16			
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	R	R	R
—	—	—	—	REV3	REV2	REV1	REV0
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 23-4 未实现: 读为 0

bit 3-0 **REV<3:0>**: 次要版本标识符位

PIC24F16KA102 系列

25.2 看门狗定时器 (WDT)

对于 PIC24F16KA102 系列器件, WDT 由 LPRC 振荡器驱动。当使能 WDT 时, 时钟源也将被使能。

由 LPRC 提供的 WDT 时钟源的频率标称值为 31 kHz。此信号输入给可配置为 5 位 (32 分频) 或 7 位 (128 分频) 工作的预分频器。预分频比由 FWPSA 配置位设置。使用 31 kHz 输入, 预分频器在 5 位模式下将产生 1 ms 的标称 WDT 超时周期 (TWDT), 在 7 位模式下产生的超时周期为 4 ms。

分频比可变的后分频器对 WDT 预分频器的输出进行分频, 并扩展超时周期范围。后分频比由配置位 WDTPS<3:0> (FWDT<3:0>) 控制, 该配置位允许选择 16 种设置, 从 1:1 到 1:32,768。使用预分频器和后分频器, 可以使超时周期的范围扩展到 1 ms 至 131 秒。

WDT、预分频器和后分频器在以下条件下复位:

- 任何器件复位时
- 在完成时钟切换时, 无论时钟切换是由软件 (即, 在更改 NOSC 位后将 OSWEN 位置 1) 引起还是由硬件 (即, 故障保护时钟监视器) 引起
- 当执行 PWRSAV 指令时 (即, 进入休眠或空闲模式)
- 当器件退出休眠或空闲模式恢复正常工作时
- 当在正常执行过程中执行 CLRWDT 指令时

如果使能了 WDT, 它将在休眠或空闲模式下继续运行。当发生 WDT 超时, 将唤醒器件并且将从执行 PWRSAV

指令处继续执行代码。唤醒器件后, 需要用软件将对应的 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。要检测后面的 WDT 事件, 必须用软件将该标志位清零。

注: 执行 CLRWDT 和 PWRSAV 指令会将预分频器和后分频器的计数值清零。

25.2.1 窗口操作

看门狗定时器有一种可选的固定窗口工作模式。在该窗口模式下, CLRWDT 指令只能在设定的 WDT 周期的后 1/4 复位 WDT。在该窗口前执行的 CLRWDT 指令会导致 WDT 复位, 类似于 WDT 超时。

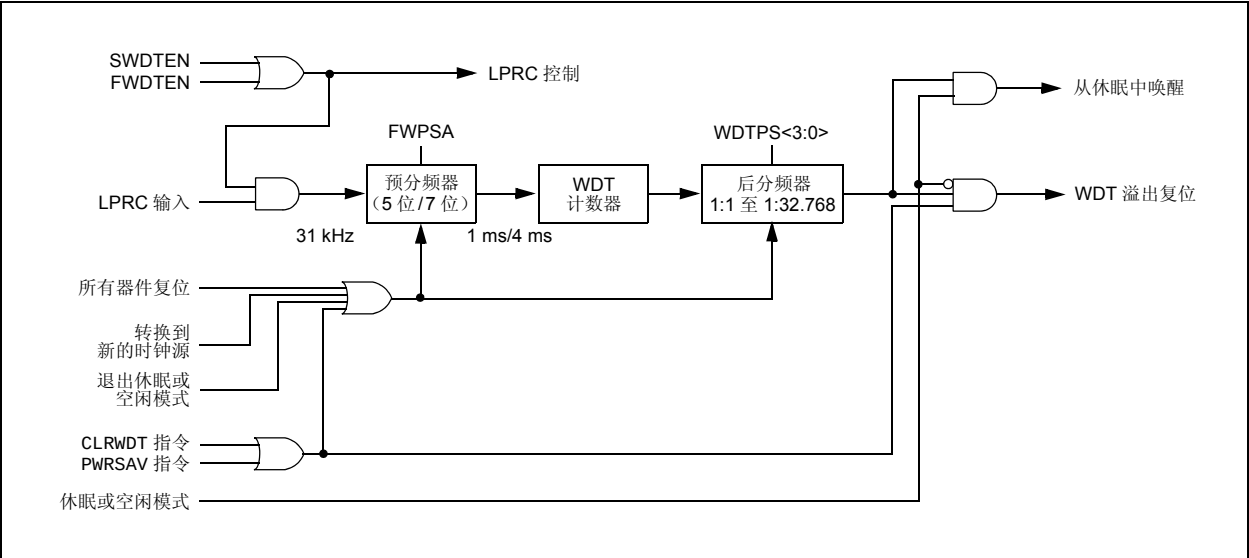
窗口 WDT 模式通过将配置位 WINDIS (FWDT<6>) 编程为 0 来使能。

25.2.2 控制寄存器

WDT 通过 FWDTEN 配置位使能或禁止。当 FWDTEN 配置位置 1 时, WDT 始终是使能的。

当 FWDTEN 配置位被编程为 0 时, 可以选择用软件控制 WDT。用软件通过将 SWDTEN 控制位 (RCON<5>) 置 1 来使能 WDT。SWDTEN 控制位在任何器件复位时清零。软件 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, 从而最大限度地降低功耗。

图 25-1: WDT 框图



25.3 深度休眠看门狗定时器 (DSWDT)

在 PIC24F16KA102 系列器件中，除了 WDT 模块之外，还具有 DSWDT 模块，该模块在器件处于深度休眠模式时运行（如果使能）。它由 SOSC 或 LPRC 振荡器驱动。时钟源通过配置位 DSWCKSEL（FDS<4>）进行选择。

DSWDT 的超时周期可以配置为 2.1 ms 至 25.7 天，配置方法是选择相应的后分频比。后分频比可以通过配置位 DSWDTPS<3:0>（FDS<3:0>）进行选择。当使能 DSWDT 时，时钟源也将被使能。

DSWDT 是可以将器件从深度休眠模式中唤醒的唤醒源之一。

25.4 程序校验和代码保护

对于 PIC24F16KA102 系列中的所有器件，引导段的代码保护由配置位 BSS0 控制，通用段的代码保护由配置位 GSS0 控制。这些位禁止外部对程序存储空间的读和写操作；但对正常执行模式没有直接影响。

写保护由配置字中的 BWRP 位（用于引导段）和 GWRP 位（用于通用段）控制。当这些位被编程为 0 时，阻止内部对程序存储器的写和擦除操作。

25.5 在线串行编程

可以在最终的应用电路中对 PIC24F16KA102 系列单片机进行串行编程。只需要 5 根线即可实现这一操作，其中时钟线（PGCx）、数据线（PGDx）各一根，其余 3 根分别是电源线、接地线和编程电压线。这允许用户在生产电路板时使用未编程器件，仅在产品交付之前才对单片机进行编程，从而可以使用最新版本的固件或者定制固件进行编程。

25.6 在线调试器

当选择 MPLAB® ICD 2 作为调试器时，使能在线调试功能。该功能允许与 MPLAB IDE 配合使用来进行简单的调试。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，就必须在设计中对 MCLR、VDD、VSS、PGCx、PGDx 和 EMUDx/EMUCx 引脚对进行正确的 ICSP 连接。此外，当使能该功能时，某些资源就不能用于一般用途了。这些资源包括数据 RAM 的前 80 字节和两个 I/O 引脚。

PIC24F16KA102 系列

注:

26.0 开发支持

一系列硬件及软件开发工具对 PIC® 单片机提供支持：

- 集成开发环境
 - MPLAB® IDE 软件
- 汇编器 / 编译器 / 链接器
 - MPASM™ 汇编器
 - MPLAB C18 和 MPLAB C30 C 编译器
 - MPLINK™ 目标链接器 / MPLIB™ 目标库管理器
 - MPLAB ASM30 汇编器 / 链接器 / 库
- 模拟器
 - MPLAB SIM 软件模拟器
- 仿真器
 - MPLAB ICE 2000 在线仿真器
 - MPLAB REAL ICE™ 在线仿真器
- 在线调试器
 - MPLAB ICD 2
- 器件编程器
 - PICSTART® Plus 开发编程器
 - MPLAB PM3 器件编程器
 - PICKit™ 2 开发编程器
- 低成本演示和开发板及评估工具包

26.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16 位单片机市场提供了前所未有的易于使用的软件开发平台。MPLAB IDE 是基于 Windows® 操作系统的应用软件，包括：

- 一个包含所有调试工具的图形界面
 - 模拟器
 - 编程器（单独销售）
 - 仿真器（单独销售）
 - 在线调试器（单独销售）
- 具有彩色上下文代码显示的全功能编辑器
- 多项目管理器
- 内容可直接编辑的可定制式数据窗口
- 高级源代码调试
- 可视化器件初始化程序，便于进行寄存器的初始化
- 鼠标停留在变量上进行查看的功能
- 通过拖放把变量从源代码窗口拉到观察窗口
- 丰富的在线帮助
- 集成了可选的第三方工具，如 HI-TECH 软件 C 编译器和 IAR C 编译器

MPLAB IDE 可以让您：

- 编辑源文件（汇编语言或 C 语言）
- 点击一次即可完成汇编（或编译）并将代码下载到 PIC MCU 仿真器和模拟器工具中（自动更新所有项目信息）
- 可使用如下各项进行调试：
 - 源文件（汇编语言或 C 语言）
 - 混合汇编语言和 C 语言
 - 机器码

MPLAB IDE 在单个开发范例中支持使用多种调试工具，包括从成本效益高的模拟器到低成本的在线调试器，再到全功能的仿真器。这样缩短了用户升级到更加灵活而功能更强大的工具时的学习时间。

26.2 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于所有的 PIC MCU。

MPASM 汇编器可生成用于 MPLINK 目标链接器的可重定位目标文件、Intel® 标准 HEX 文件、详细描述存储器使用状况和符号参考的 MAP 文件、包含源代码行及生成机器码的绝对 LST 文件以及用于调试的 COFF 文件。

MPASM 汇编器具有如下特征：

- 集成在 MPLAB IDE 项目中
- 用户定义的宏可简化汇编代码
- 对多用途源文件进行条件汇编
- 允许完全控制汇编过程的指令

26.3 MPLAB C18 和 MPLAB C30 C 编译器

MPLAB C18 和 MPLAB C30 代码开发系统是完全的 ANSI C 编译器，分别适用于 Microchip 的 PIC18 和 PIC24 系列单片机及 dsPIC30F 和 dsPIC33 系列数字信号控制器。这些编译器可提供其他编译器并不具备的强大的集成功能和出众的代码优化能力，且使用方便。

为便于源代码调试，编译器提供了针对 MPLAB IDE 调试器的优化符号信息。

26.4 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器包含了由 MPASM 汇编器、MPLAB C18 C 编译器产生的可重定位目标。通过使用链接器脚本中的指令，它还可链接预编译库中的可重定位目标。

MPLIB 目标库管理器管理预编译代码库文件的创建和修改。当从源文件调用库中的一段子程序时，只有包含此子程序的模块被链接到应用中。这样可使大型库在许多不同应用中被高效地利用。

目标链接器 / 库管理器具有如下特征：

- 高效地连接单个的库而不是许多小文件
- 通过将相关的模块组合在一起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

26.5 MPLAB ASM30 汇编器、链接器和库管理器

MPLAB ASM30 汇编器为 dsPIC30F 器件提供转换自符号汇编语言的可重定位机器码。MPLAB C30 C 编译器使用该汇编器生成目标文件。汇编器产生可重定位目标文件之后，可将这些目标文件存档，或与其他可重定位目标文件和存档链接以生成可执行文件。该汇编器有如下显著特征：

- 支持整个 dsPIC30F 指令集
- 支持定点数据和浮点数据
- 命令行界面
- 丰富的指令集
- 灵活的宏语言
- MPLAB IDE 兼容性

26.6 MPLAB SIM 软件模拟器

MPLAB SIM 软件模拟器在指令级对 PIC MCU 和 dsPIC® DSC 进行模拟，使得用户可以在 PC 主机的环境下进行代码开发。对于任何给定的指令，用户均可对数据区进行检查或修改，并通过各种触发机制来产生激励。可以将各寄存器的情况记录在文件中，以便进行进一步地运行时分析。跟踪缓冲器和逻辑分析器的显示使模拟器还能记录和跟踪程序的执行、I/O 的动作、大部分的外设及内部寄存器的状况。

MPLAB SIM 软件模拟器完全支持使用 MPLAB C18 和 MPLAB C30 C 编译器以及 MPASM 和 MPLAB ASM30 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。

26.7 MPLAB ICE 2000 高性能在线仿真器

MPLAB ICE 2000 在线仿真器旨在为产品开发工程师提供一整套用于 PIC 单片机的设计工具。MPLAB ICE 2000 在线仿真器的软件控制由 MPLAB 集成开发环境平台提供，它允许在单一环境下进行编辑、编译、下载以及源代码调试。

MPLAB ICE 2000 是全功能仿真器系统，它具有增强的跟踪、触发和数据监控功能。处理器模块可插拔，使系统可轻松进行重新配置以适应各种不同处理器的仿真需要。MPLAB ICE 2000 在线仿真器的架构允许对其进行扩展以支持新的 PIC 单片机。

MPLAB ICE 2000 在线仿真器系统设计为一款实时仿真系统，该仿真系统具备通常只有昂贵的开发工具中才有的高级功能。选择 PC 平台和 Microsoft® Windows® 32 位操作系统可使这些功能在一个简单而统一的应用中得到很好的利用。

26.8 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® DSC 进行调试和编程。IDE 是随每个工具包一起提供的。

MPLAB REAL ICE 探针通过高速 USB 2.0 接口与设计工程师的 PC 相连，并利用与常用 MPLAB ICD 2 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 MPLAB REAL ICE 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，如软件断点和汇编代码跟踪等。在同类仿真器中，MPLAB REAL ICE 的优势十分明显：低成本、高速仿真、实时变量监视、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。

26.9 MPLAB ICD 2 在线调试器

Microchip 的在线调试器 MPLAB ICD 2 是一款功能强大而成本低廉的运行时开发工具，通过 RS-232 或高速 USB 接口与 PC 主机相连。该工具基于闪存 PIC MCU，可用于开发本系列及其他 PIC MCU 和 dsPIC DSC。MPLAB ICD 2 使用了闪存器件中内建的在线调试功能。该功能结合 Microchip 的在线串行编程 (In-Circuit Serial Programming™, ICSP™) 协议，可在 MPLAB 集成开发环境的图形用户界面上提供成本效益很高的在线闪存调试。这使设计人员可通过设置断点、单步运行以及对变量、CPU 状态以及外设寄存器进行监视的方法实现源代码的开发和调试。其全速运行特性可对硬件和应用进行实时测试。MPLAB ICD 2 还可用作某些 PIC 器件的开发编程器。

26.10 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款通用的、符合 CE 规范的器件编程器，其可编程电压设置在 VDDMIN 和 VDDMAX 之间时可靠性最高。它有一个用来显示菜单和错误信息的大 LCD 显示器 (128 x 64)，以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 ICSP™ 电缆。在单机模式下，MPLAB PM3 器件编程器不必与 PC 相连即可对 PIC 器件进行读取、验证和编程。在该模式下它还可设置代码保护。MPLAB PM3 通过 RS-232 或 USB 电缆连接到 PC 主机上。MPLAB PM3 具备高速通信能力以及优化算法，可对存储器很大的器件进行快速编程，它还采用 SD/MMC 卡用作文件存储及数据安全应用。

PIC24F16KA102 系列

26.11 PICSTART Plus 开发编程器

PICSTART Plus 开发编程器是一款易于使用而成本低廉的原型编程器。它通过 COM (RS-232) 端口与 PC 相连。MPLAB 集成开发环境软件使得该编程器的使用简便、高效。PICSTART Plus 开发编程器支持采用 DIP 封装的大部分 PIC 器件，其引脚数最多可达 40 个。引脚数更多的器件，如 PIC16C92X 和 PIC17C76X，可通过连接一个转接插槽来获得支持。PICSTART Plus 开发编程器符合 CE 规范。

26.12 PICKit 2 开发编程器

PICKit™ 2 开发编程器是一个低成本编程器；对于某些选定闪存器件，它也是一个调试器，通过其易于使用的接口可对众多 Microchip 的低档、中档和 PIC18F 系列闪存单片机进行编程。PICKit 2 入门工具包中包含一个有实验布线区的开发板、十二堂系列课程、软件和 HI-TECH 的 PICC™ Lite C 编译器，有助于用户快速掌握 PIC® 单片机的使用。这一工具包为使用 Microchip 功能强大的中档闪存系列单片机进行编程、评估和应用开发，提供了所需的一切。

26.13 演示、开发和评估板

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，用于测试和修改。

这些板支持多种功能部件，包括 LED、温度传感器、开关、扬声器、RS-232 接口、LCD 显示器、电位计和附加 EEPROM 存储器。

演示和开发板可用于教学环境，在实验布线区设计定制电路，从而掌握各种单片机应用。

除了 PICDEM™ 和 dsPICDEM™ 演示 / 开发板系列电路外，Microchip 还有一系列评估工具包和演示软件，适用于模拟滤波器设计、KEELOQ® 数据安全产品 IC、CAN、IrDA®、PowerSmart 电池管理、SEEVAL® 评估系统、 Σ - Δ ADC、流速传感器，等等。

有关演示、开发和评估工具包的完整列表，请查阅 Microchip 公司网页 (www.microchip.com)。

27.0 指令集汇总

注： 本章简要概述了 PIC24F 指令集架构，但是不应将其当作无所不包的参考手册来使用。

与以前的 PIC® MCU 指令集相比，PIC24F 指令集添加了许多增强功能，同时保持了易于从以前 PIC MCU 指令集移植的特点。大部分指令的长度为一个程序存储字。只有三条指令需要两个程序存储单元。

每一条单字指令长 24 位，分为一个指定指令类型的 8 位操作码和进一步指定指令操作的一个或多个操作数。指令集是高度正交的，分为 4 个基本类别：

- 针对字或字节的操作
- 针对位的操作
- 立即数操作
- 控制操作

表 27-1 给出了在说明指令时使用的通用符号。表 27-2 是 PIC24F 指令集的汇总，它列出了所有指令，以及每条指令影响的状态标志位。

大多数针对字或字节的 W 寄存器指令（包括桶形移位指令）有三个操作数：

- 第一个源操作数，通常是寄存器 Wb，不带任何地址修改量
- 第二个源操作数，通常是寄存器 Ws，带或不带地址修改量
- 运算结果的目标寄存器，通常是寄存器 Wd，带或不带地址修改量

不过，针对字或字节的文件寄存器指令只有两个操作数：

- 文件寄存器，由 f 值指定
- 目标寄存器，可以是文件寄存器 f 或 W0 寄存器（用 WREG 表示）

大多数位操作类指令（包括简单的循环 / 移位指令）有两个操作数：

- W 寄存器（带或不带地址修改量）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（由立即数值指定，或者由寄存器 Wb 的内容间接指定）

涉及数据传送的立即数指令，可以使用下列操作数：

- 要被装入到 W 寄存器或文件寄存器中的立即数（由 k 值指定）
- 要装入立即数的 W 寄存器或文件寄存器（由 Wb 或 f 指定）

然而，涉及算术或逻辑运算的立即数指令，使用如下的操作数：

- 第一个源操作数是寄存器 Wb，不带任何地址修改量
- 第二个源操作数是立即数
- 操作结果的目标寄存器（仅在与第一个源操作数不同时）通常是寄存器 Wd（带或不带地址修改量）

控制指令可以使用下列操作数：

- 程序存储器中的地址
- 表读和表写指令的模式

除了某些双字指令外，所有指令都是单字指令；双字指令之所以是双字长的（48 位），是因为要用 48 位来提供所需信息。在第二个字中，8 个 MSb 全为 0。如果指令自身把第二个字当作一条指令来执行的话，它将作为一条 NOP 指令来执行。

大多数单字长指令都在一个指令周期内执行，除非条件测试结果为 true 或者指令执行结果改变了程序计数器（PC）。对于上述两种特殊情况，指令执行需要两个指令周期，在第二个指令周期中执行一条 NOP 指令。值得注意的例外是 BRA（无条件/计算转移）、间接 CALL/GOTO、所有的表读和表写以及 RETURN/RETFIE 指令，它们是单字长指令，但执行需要两个或三个周期。

某些可能涉及到跳过后续指令的指令，如果要执行跳过的话，可能需要两个或三个周期，这取决于被跳过的指令是单字还是双字指令。此外，双字传送需要两个周期。双字指令执行需要两个指令周期。

PIC24F16KA102 系列

表 27-1: 操作码说明中使用的符号

字段	说明
#text	表示由 text 定义的立即数
(text)	表示 text 的内容
[text]	表示由 text 寻址的存储单元
{ }	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） $\in \{0...15\}$
C, DC, N, OV, Z	MCU 状态位：进位、半进位、负、溢出和全零标志位
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 $\in \{0000h...1FFFh\}$
lit1	1 位无符号立即数 $\in \{0,1\}$
lit4	4 位无符号立即数 $\in \{0...15\}$
lit5	5 位无符号立即数 $\in \{0...31\}$
lit8	8 位无符号立即数 $\in \{0...255\}$
lit10	10 位无符号立即数，对于字节模式， $\in \{0...255\}$ ；对于字模式， $\in \{0:1023\}$
lit14	14 位无符号立即数 $\in \{0...16384\}$
lit16	16 位无符号立即数 $\in \{0...65535\}$
lit23	23 位无符号立即数 $\in \{0...8388608\}$ ；LSB 必须为 0
None	字段无需内容，可为空
PC	程序计数器
Slit10	10 位有符号立即数 $\in \{-512...511\}$
Slit16	16 位有符号立即数 $\in \{-32768...32767\}$
Slit6	6 位有符号立即数 $\in \{-16...16\}$
Wb	基本 W 寄存器 $\in \{W0..W15\}$
Wd	目标 W 寄存器 $\in \{Wd, [Wd], [Wd++] , [Wd--], [++Wd], [--Wd] \}$
Wdo	目标 W 寄存器 $\in \{Wnd, [Wnd], [Wnd++] , [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] \}$
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 $\in \{W0..W15\}$
Wnd	16 个目标工作寄存器之一 $\in \{W0..W15\}$
Wns	16 个源工作寄存器之一 $\in \{W0..W15\}$
WREG	W0（文件寄存器指令中使用的工作寄存器）
Ws	源 W 寄存器 $\in \{Ws, [Ws], [Ws++] , [Ws--], [++Ws], [--Ws] \}$
Wso	源 W 寄存器 $\in \{Wns, [Wns], [Wns++] , [Wns--], [++Wns], [--Wns], [Wns+Wb] \}$

表 27-2: 指令集概述

汇编助记符	汇编语法	说明	字数	周期数	影响的状态标志
ADD	ADD f	$f = f + WREG$	1	1	C, DC, N, OV, Z
	ADD f, WREG	$WREG = f + WREG$	1	1	C, DC, N, OV, Z
	ADD #lit10, Wn	$Wd = lit10 + Wd$	1	1	C, DC, N, OV, Z
	ADD Wb, Ws, Wd	$Wd = Wb + Ws$	1	1	C, DC, N, OV, Z
	ADD Wb, #lit5, Wd	$Wd = Wb + lit5$	1	1	C, DC, N, OV, Z
ADDC	ADDC f	$f = f + WREG + (C)$	1	1	C, DC, N, OV, Z
	ADDC f, WREG	$WREG = f + WREG + (C)$	1	1	C, DC, N, OV, Z
	ADDC #lit10, Wn	$Wd = lit10 + Wd + (C)$	1	1	C, DC, N, OV, Z
	ADDC Wb, Ws, Wd	$Wd = Wb + Ws + (C)$	1	1	C, DC, N, OV, Z
	ADDC Wb, #lit5, Wd	$Wd = Wb + lit5 + (C)$	1	1	C, DC, N, OV, Z
AND	AND f	$f = f .AND. WREG$	1	1	N, Z
	AND f, WREG	$WREG = f .AND. WREG$	1	1	N, Z
	AND #lit10, Wn	$Wd = lit10 .AND. Wd$	1	1	N, Z
	AND Wb, Ws, Wd	$Wd = Wb .AND. Ws$	1	1	N, Z
	AND Wb, #lit5, Wd	$Wd = Wb .AND. lit5$	1	1	N, Z
ASR	ASR f	f = 算术右移 f	1	1	C, N, OV, Z
	ASR f, WREG	WREG = 算术右移 f	1	1	C, N, OV, Z
	ASR Ws, Wd	Wd = 算术右移 Ws	1	1	C, N, OV, Z
	ASR Wb, Wns, Wnd	Wnd = 将 Wb 算术右移 Wns 位	1	1	N, Z
	ASR Wb, #lit5, Wnd	Wnd = 将 Wb 算术右移 lit5 位	1	1	N, Z
BCLR	BCLR f, #bit4	将 f 中的指定位清零	1	1	无
	BCLR Ws, #bit4	将 Ws 中的指定位清零	1	1	无
BRA	BRA C, Expr	如果有进位则转移	1	1 (2)	无
	BRA GE, Expr	如果大于或等于则转移	1	1 (2)	无
	BRA GEU, Expr	如果无符号大于或等于则转移	1	1 (2)	无
	BRA GT, Expr	如果大于则转移	1	1 (2)	无
	BRA GTU, Expr	如果无符号大于则转移	1	1 (2)	无
	BRA LE, Expr	如果小于或等于则转移	1	1 (2)	无
	BRA LEU, Expr	如果无符号小于或等于则转移	1	1 (2)	无
	BRA LT, Expr	如果小于则转移	1	1 (2)	无
	BRA LTU, Expr	如果无符号小于则转移	1	1 (2)	无
	BRA N, Expr	如果为负则转移	1	1 (2)	无
	BRA NC, Expr	如果没有进位则转移	1	1 (2)	无
	BRA NN, Expr	如果不为负则转移	1	1 (2)	无
	BRA NOV, Expr	如果不溢出则转移	1	1 (2)	无
	BRA NZ, Expr	如果不为零则转移	1	1 (2)	无
	BRA OV, Expr	如果溢出则转移	1	1 (2)	无
	BRA Expr	无条件转移	1	2	无
	BRA Z, Expr	如果为零则转移	1	1 (2)	无
	BRA Wn	计算转移	1	2	无
BSET	BSET f, #bit4	将 f 中的指定位置 1	1	1	无
	BSET Ws, #bit4	将 Ws 中的指定位置 1	1	1	无
BSW	BSW.C Ws, Wb	将 C 位内容写入 Ws<Wb>	1	1	无
	BSW.Z Ws, Wb	将 Z 位内容写入 Ws<Wb>	1	1	无
BTG	BTG f, #bit4	将 f 中的指定位翻转	1	1	无
	BTG Ws, #bit4	将 Ws 中的指定位翻转	1	1	无
BTSC	BTSC f, #bit4	对 f 中的指定位进行测试, 如果为零则跳过	1	1 (2 或 3)	无
	BTSC Ws, #bit4	对 Ws 中的指定位进行测试, 如果为零则跳过	1	1 (2 或 3)	无

PIC24F16KA102 系列

表 27-2: 指令集概述 (续)

汇编助记符	汇编语法	说明	字数	周期数	影响的状态标志
BTSS	BTSS f, #bit4	对 f 中的指定位进行测试, 如果为 1 则跳过	1	1 (2 或 3)	无
	BTSS Ws, #bit4	对 Ws 中的指定位进行测试, 如果为 1 则跳过	1	1 (2 或 3)	无
BTST	BTST f, #bit4	对 f 中的指定位进行测试	1	1	Z
	BTST.C Ws, #bit4	对 Ws 中的指定位进行测试, 并将其值存储到 C	1	1	C
	BTST.Z Ws, #bit4	对 Ws 中的指定位进行测试, 并将其反码存储到 Z	1	1	Z
	BTST.C Ws, Wb	对 Ws<Wb> 位进行测试, 并将其值存储到 C	1	1	C
	BTST.Z Ws, Wb	对 Ws<Wb> 位进行测试, 并将其反码存储到 Z	1	1	Z
	BTST.Z Ws, Wb	对 Ws<Wb> 位进行测试, 并将其反码存储到 Z	1	1	Z
BTSTS	BTSTS f, #bit4	对 f 中的指定位进行测试, 并将 f 中的该位置 1	1	1	Z
	BTSTS.C Ws, #bit4	对 Ws 中的指定位进行测试, 并将其值存储到 C, 然后将 Ws 中的该位置 1	1	1	C
	BTSTS.Z Ws, #bit4	对 Ws 中的指定位进行测试, 并将其反码存储到 Z, 然后将 Ws 中的该位置 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
CLRWDI	CLRWDI	将看门狗定时器清零	1	1	WDT0, Sleep
COM	COM f	f = $\bar{f}$	1	1	N, Z
	COM f, WREG	WREG = $\bar{f}$	1	1	N, Z
	COM Ws, Wd	Wd = $\bar{Ws}$	1	1	N, Z
CP	CP f	比较 f 和 WREG	1	1	C, DC, N, OV, Z
	CP Wb, #lit5	比较 Wb 和 lit5	1	1	C, DC, N, OV, Z
	CP Wb, Ws	比较 Wb 和 Ws (Wb - Ws)	1	1	C, DC, N, OV, Z
CP0	CP0 f	比较 f 和 0x0000	1	1	C, DC, N, OV, Z
	CP0 Ws	比较 Ws 和 0x0000	1	1	C, DC, N, OV, Z
CPB	CPB f	带借位比较 f 和 WREG	1	1	C, DC, N, OV, Z
	CPB Wb, #lit5	带借位比较 Wb 和 lit5	1	1	C, DC, N, OV, Z
	CPB Wb, Ws	带借位比较 Wb 和 Ws (Wb - Ws - C)	1	1	C, DC, N, OV, Z
CPSEQ	CPSEQ Wb, Wn	比较 Wb 和 Wn, 如果相等则跳过	1	1 (2 或 3)	无
CPSGT	CPSGT Wb, Wn	比较 Wb 和 Wn, 如果大于则跳过	1	1 (2 或 3)	无
CPSLT	CPSLT Wb, Wn	比较 Wb 和 Wn, 如果小于则跳过	1	1 (2 或 3)	无
CPSNE	CPSNE Wb, Wn	比较 Wb 和 Wn, 如果不相等则跳过	1	1 (2 或 3)	无
DAW	DAW Wn	Wn = 十进制调整 Wn	1	1	C
DEC	DEC f	f = f - 1	1	1	C, DC, N, OV, Z
	DEC f, WREG	WREG = f - 1	1	1	C, DC, N, OV, Z
	DEC Ws, Wd	Wd = Ws - 1	1	1	C, DC, N, OV, Z
DEC2	DEC2 f	f = f - 2	1	1	C, DC, N, OV, Z
	DEC2 f, WREG	WREG = f - 2	1	1	C, DC, N, OV, Z
	DEC2 Ws, Wd	Wd = Ws - 2	1	1	C, DC, N, OV, 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, OV
	DIV.SD Wm, Wn	有符号 32/16 位整数除法	1	18	N, Z, C, OV
	DIV.UW Wm, Wn	无符号 16/16 位整数除法	1	18	N, Z, C, OV
	DIV.UD Wm, Wn	无符号 32/16 位整数除法	1	18	N, Z, C, OV
EXCH	EXCH Wns, Wnd	交换 Wns 和 Wnd 的内容	1	1	无
FF1L	FF1L Ws, Wnd	从左边 (MSb) 查找第一个 1	1	1	C
FF1R	FF1R Ws, Wnd	从右边 (LSb) 查找第一个 1	1	1	C

表 27-2: 指令集概述 (续)

汇编助记符	汇编语法	说明	字数	周期数	影响的状态标志
GOTO	GOTO Expr	转移到地址	2	2	无
	GOTO Wn	间接转移到地址	1	2	无
INC	INC f	$f = f + 1$	1	1	C, DC, N, OV, Z
	INC f, WREG	$WREG = f + 1$	1	1	C, DC, N, OV, Z
	INC Ws, Wd	$Wd = Ws + 1$	1	1	C, DC, N, OV, Z
INC2	INC2 f	$f = f + 2$	1	1	C, DC, N, OV, Z
	INC2 f, WREG	$WREG = f + 2$	1	1	C, DC, N, OV, Z
	INC2 Ws, Wd	$Wd = Ws + 2$	1	1	C, DC, N, OV, Z
IOR	IOR f	$f = f .IOR.WREG$	1	1	N, Z
	IOR f, WREG	$WREG = f .IOR.WREG$	1	1	N, Z
	IOR #lit10, Wn	$Wd = lit10 .IOR.Wd$	1	1	N, Z
	IOR Wb, Ws, Wd	$Wd = Wb .IOR.Ws$	1	1	N, Z
	IOR Wb, #lit5, Wd	$Wd = Wb .IOR.lit5$	1	1	N, Z
LNK	LNK #lit14	分配堆栈帧	1	1	无
LSR	LSR f	$f =$ 逻辑右移 f	1	1	C, N, OV, Z
	LSR f, WREG	$WREG =$ 逻辑右移 f	1	1	C, N, OV, Z
	LSR Ws, Wd	$Wd =$ 逻辑右移 Ws	1	1	C, N, OV, Z
	LSR Wb, Wns, Wnd	$Wnd =$ 将 Wb 逻辑右移 Wns 位	1	1	N, Z
	LSR Wb, #lit5, Wnd	$Wnd =$ 将 Wb 逻辑右移 $lit5$ 位	1	1	N, Z
MOV	MOV f, Wn	将 f 中的内容送入 Wn	1	1	无
	MOV [Wns+Slit10], Wnd	将 [Wns+Slit10] 中的内容送入 Wnd	1	1	无
	MOV f	将 f 中的内容送入目标寄存器	1	1	N, Z
	MOV f, WREG	将 f 中的内容送入 WREG	1	1	N, Z
	MOV #lit16, Wn	将 16 位立即数送入 Wn	1	1	无
	MOV.b #lit8, Wn	将 8 位立即数送入 Wn	1	1	无
	MOV Wn, f	将 Wn 中的内容送入 f	1	1	无
	MOV Wns, [Wns+Slit10]	将 Wns 中的内容送入 [Wns+Slit10]	1	1	无
	MOV Wso, Wdo	将 Ws 中的内容送入 Wd	1	1	无
	MOV WREG, f	将 WREG 中的内容送入 f	1	1	N, Z
	MOV.D Wns, Wd	将 $W(ns):W(ns+1)$ 中的双字内容送入 Wd	1	2	无
	MOV.D Ws, Wnd	将 Ws 中的双字内容送入 $W(nd+1):W(nd)$	1	2	无
MUL	MUL.SS Wb, Ws, Wnd	$\{Wnd+1, Wnd\} = \text{Signed}(Wb) * \text{Signed}(Ws)$	1	1	无
	MUL.SU Wb, Ws, Wnd	$\{Wnd+1, Wnd\} = \text{Signed}(Wb) * \text{Unsigned}(Ws)$	1	1	无
	MUL.US Wb, Ws, Wnd	$\{Wnd+1, Wnd\} = \text{Unsigned}(Wb) * \text{Signed}(Ws)$	1	1	无
	MUL.UU Wb, Ws, Wnd	$\{Wnd+1, Wnd\} = \text{Unsigned}(Wb) * \text{Unsigned}(Ws)$	1	1	无
	MUL.SU Wb, #lit5, Wnd	$\{Wnd+1, Wnd\} = \text{Signed}(Wb) * \text{Unsigned}(lit5)$	1	1	无
	MUL.UU Wb, #lit5, Wnd	$\{Wnd+1, Wnd\} = \text{Unsigned}(Wb) * \text{Unsigned}(lit5)$	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, Z
	NEG f, WREG	$WREG = \bar{f} + 1$	1	1	C, DC, N, OV, Z
	NEG Ws, Wd	$Wd = \overline{Ws} + 1$	1	1	C, DC, N, OV, Z
NOP	NOP	空操作	1	1	无
	NOPR	空操作	1	1	无
POP	POP f	将栈顶 (TOS) 的内容弹出到 f	1	1	无
	POP Wdo	将栈顶 (TOS) 的内容弹出到 Wdo	1	1	无
	POP.D Wnd	将栈顶 (TOS) 的内容弹出到 $W(nd):W(nd+1)$	1	2	无
	POP.S	将影子寄存器的内容弹出到主寄存器	1	1	全部
PUSH	PUSH f	将 f 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 $W(ns):W(ns+1)$ 的双字内容压入栈顶 (TOS)	1	2	无
	PUSH.S	将主寄存器中的内容压入影子寄存器	1	1	无

PIC24F16KA102 系列

表 27-2: 指令集概述 (续)

汇编助记符	汇编语法	说明	字数	周期数	影响的状态标志
PWRSV	PWRSV #lit1	进入休眠或空闲模式	1	1	WDTO, Sleep
RCALL	RCALL Expr	相对调用	1	2	无
	RCALL Wn	计算调用	1	2	无
REPEAT	REPEAT #lit14	将下一条指令重复执行 lit14 + 1 次	1	1	无
	REPEAT Wn	将下一条指令重复执行 (Wn) + 1 次	1	1	无
RESET	RESET	软件器件复位	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 执行带进位的循环左移	1	1	C, N, Z
	RLC f, WREG	WREG = 对 f 执行带进位的循环左移	1	1	C, N, Z
	RLC Ws, Wd	Wd = 对 Ws 执行带进位的循环左移	1	1	C, N, Z
RLNC	RLNC f	f = 循环左移 f (不带进位)	1	1	N, Z
	RLNC f, WREG	WREG = 循环左移 f (不带进位)	1	1	N, Z
	RLNC Ws, Wd	Wd = 循环左移 Ws (不带进位)	1	1	N, Z
RRC	RRC f	f = 对 f 执行带进位的循环右移	1	1	C, N, Z
	RRC f, WREG	WREG = 对 f 执行带进位的循环右移	1	1	C, N, Z
	RRC Ws, Wd	Wd = 对 Ws 执行带进位的循环右移	1	1	C, N, Z
RRNC	RRNC f	f = 循环右移 f (不带进位)	1	1	N, Z
	RRNC f, WREG	WREG = 循环右移 f (不带进位)	1	1	N, Z
	RRNC Ws, Wd	Wd = 循环右移 Ws (不带进位)	1	1	N, Z
SE	SE Ws, Wnd	Wnd = 符号扩展后的 Ws	1	1	C, N, 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, Z
	SL f, WREG	WREG = 左移 f	1	1	C, N, OV, Z
	SL Ws, Wd	Wd = 左移 Ws	1	1	C, N, OV, Z
	SL Wb, Wns, Wnd	Wnd = 将 Wb 左移 Wns 位	1	1	N, Z
	SL Wb, #lit5, Wnd	Wnd = 将 Wb 左移 lit5 位	1	1	N, Z
SUB	SUB f	f = f - WREG	1	1	C, DC, N, OV, Z
	SUB f, WREG	WREG = f - WREG	1	1	C, DC, N, OV, Z
	SUB #lit10, Wn	Wn = Wn - lit10	1	1	C, DC, N, OV, Z
	SUB Wb, Ws, Wd	Wd = Wb - Ws	1	1	C, DC, N, OV, Z
	SUB Wb, #lit5, Wd	Wd = Wb - lit5	1	1	C, DC, N, OV, Z
SUBB	SUBB f	f = f - WREG - ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB f, WREG	WREG = f - WREG - ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB #lit10, Wn	Wn = Wn - lit10 - ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB Wb, Ws, Wd	Wd = Wb - Ws - ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBB Wb, #lit5, Wd	Wd = Wb - lit5 - ($\overline{C}$)	1	1	C, DC, N, OV, Z
SUBR	SUBR f	f = WREG - f	1	1	C, DC, N, OV, Z
	SUBR f, WREG	WREG = WREG - f	1	1	C, DC, N, OV, Z
	SUBR Wb, Ws, Wd	Wd = Ws - Wb	1	1	C, DC, N, OV, Z
	SUBR Wb, #lit5, Wd	Wd = lit5 - Wb	1	1	C, DC, N, OV, Z
SUBBR	SUBBR f	f = WREG - f - ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR f, WREG	WREG = WREG - f - ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR Wb, Ws, Wd	Wd = Ws - Wb - ($\overline{C}$)	1	1	C, DC, N, OV, Z
	SUBBR Wb, #lit5, Wd	Wd = lit5 - Wb - ($\overline{C}$)	1	1	C, DC, N, OV, Z
SWAP	SWAP.b Wn	Wn = 将 Wn 的前后两个半字节交换	1	1	无
	SWAP Wn	Wn = 将 Wn 的两个字节相交换	1	1	无
TBLRDH	TBLRDH Ws, Wd	将程序存储器中某个单元的 bit<23:16> 读入 Wd<7:0>	1	2	无

表 27-2: 指令集概述 (续)

汇编助记符	汇编语法	说明	字数	周期数	影响的状态标志
TBLRDL	TBLRDL Ws, Wd	将程序存储器中某个单元的 bit<15:0> 读入 Wd	1	2	无
TBLWTH	TBLWTH Ws, Wd	将 Ws<7:0> 写入程序存储器中某个单元的 bit<23:16>	1	2	无
TBLWTL	TBLWTL Ws, Wd	将 Ws 写入程序存储器中某个单元的 bit<15:0>	1	2	无
ULNK	ULNK	释放堆栈帧	1	1	无
XOR	XOR f	f = f .XOR.WREG	1	1	N, Z
	XOR f, WREG	WREG = f .XOR.WREG	1	1	N, Z
	XOR #lit10, Wn	Wd = lit10 .XOR.Wd	1	1	N, Z
	XOR Wb, Ws, Wd	Wd = Wb .XOR.Ws	1	1	N, Z
	XOR Wb, #lit5, Wd	Wd = Wb .XOR. lit5	1	1	N, Z
ZE	ZE Ws, Wnd	Wnd = 零扩展后的 Ws	1	1	C, Z, N

PIC24F16KA102 系列

注:

28.0 电气特性

本章将对 PIC24F16KA102 系列的电气特性进行概括介绍。其余信息在本文档的将来版本中提供。

下面列出了 PIC24F16KA102 器件的绝对最大值。器件长时间工作在最大值条件下可能会影响其可靠性。我们不建议使器件在或超过本规范指定的最大值条件下运行。

绝对最大值 (†)

环境温度.....	-40°C 至 +125°C
储存温度.....	-65°C 至 +150°C
VDD 引脚相对于 VSS 的电压.....	-0.3V 至 +4.0V
任一模拟 / 数字引脚相对于 Vss 的电压.....	-0.3V 至 (VDD + 0.3V)
任一仅用作数字功能的引脚相对于 Vss 的电压.....	-0.3V 至 (VDD + 0.3V)
MCLR/VPP 引脚相对于 VSS 的电压.....	-0.3V 至 +9.0V
流出 VSS 引脚的最大电流.....	300 mA
流入 VDD 引脚的最大电流 (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 (1).....	200 mA

注 1： 允许的最大电流由器件最大功耗决定（见表 28-1）。

† 注： 如果器件工作条件超过上述“绝对最大值”，可能引起器件永久性损坏。这仅是极限参数，我们不建议器件工作在极限值甚至超过上述极限值。器件长时间工作在极限条件下可能会影响其可靠性。

PIC24F16KA102 系列

28.1 直流特性

图 28-1: PIC24F16KA102 系列电压—频率关系图（工业级）

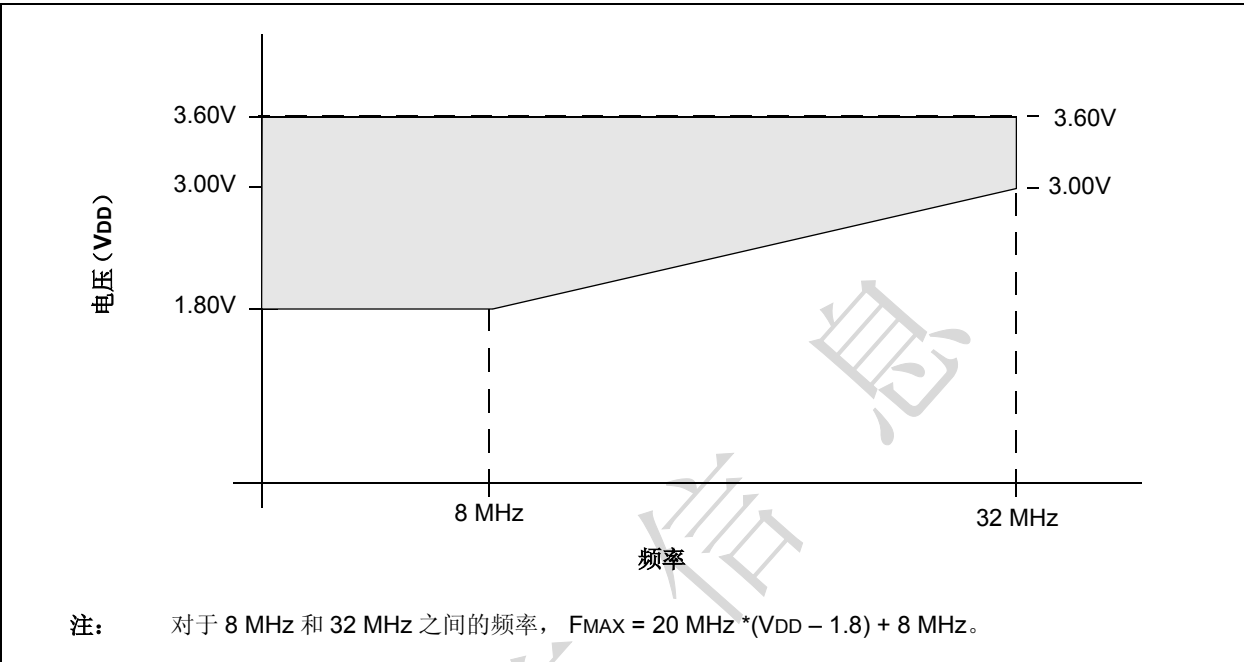


表 28-1: 温度工作条件

参数	符号	最小值	典型值	最大值	单位
工作结温范围	TJ	-40	—	+125	°C
工作环境温度范围	TA	-40	—	+85	°C
功耗: 芯片内部功耗: $P_{INT} = V_{DD} \times (I_{DD} - \sum I_{OH})$ I/O 引脚功耗: $P_{I/O} = \sum (\{V_{DD} - V_{OH}\} \times I_{OH}) + \sum (V_{OL} \times I_{OL})$	PD	PINT + PI/O			W
最大允许功耗	PDMAX	$(T_J - T_A)/\theta_{JA}$			W

表 28-2: 温度封装特性

特性	符号	典型值	最大值	单位	注
封装热阻，20 引脚 PDIP	θ_{JA}	62.4	—	°C/W	1
封装热阻，28 引脚 SPDIP	θ_{JA}	60	—	°C/W	1
封装热阻，20 引脚 SSOP	θ_{JA}	108	—	°C/W	1
封装热阻，28 引脚 SSOP	θ_{JA}	71	—	°C/W	1
封装热阻，20 引脚 SOIC	θ_{JA}	75	—	°C/W	1
封装热阻，28 引脚 SOIC	θ_{JA}	80.2	—	°C/W	1
封装热阻，20 引脚 QFN	θ_{JA}	43	—	°C/W	1
封装热阻，28 引脚 QFN	θ_{JA}	32	—	°C/W	1

注 1: 通过封装模拟获得结点与环境的热阻值 θ_{JA} 。

表 28-3: 直流特性: 温度和电压规范

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (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
DC18	VDD	供电电压	1.8	—	3.6	V	
DC12	VDR	RAM 数据保持电压 ⁽²⁾	1.5	—	—	V	
DC16	VPOR	VDD 启动电压 (确保内部上电复位信号)	—	—	VSS	V	
DC17	SVDD	VDD 上升速率 (确保内部上电复位信号)	0.05	—	—	V/ms	0-3.3V/0.1s 0-2.5V/60 ms

- 注 1: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。
- 2: 这是在不丢失 RAM 数据的前提下, VDD 的下限值。

表 28-4: 高 / 低压检测特性

标准工作条件 (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (工业级)							
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
DC18	VHLVD	VDD 跳变时的 HLVD 电压	HLVDL<3:0> = 0000	TBD	1.85	TBD	V
			HLVDL<3:0> = 0001	TBD	1.90	TBD	V
			HLVDL<3:0> = 0010	TBD	1.95	TBD	V
			HLVDL<3:0> = 0011	TBD	2.00	TBD	V
			HLVDL<3:0> = 0100	TBD	2.05	TBD	V
			HLVDL<3:0> = 0101	TBD	2.17	TBD	V
			HLVDL<3:0> = 0110	TBD	2.23	TBD	V
			HLVDL<3:0> = 0111	TBD	2.36	TBD	V
			HLVDL<3:0> = 1000	TBD	2.43	TBD	V
			HLVDL<3:0> = 1001	TBD	2.60	TBD	V
			HLVDL<3:0> = 1010	TBD	2.78	TBD	V
			HLVDL<3:0> = 1011	TBD	2.88	TBD	V
			HLVDL<3:0> = 1100	TBD	3.00	TBD	V
			HLVDL<3:0> = 1101	TBD	3.12	TBD	V
			HLVDL<3:0> = 1110	TBD	3.39	TBD	V

图注: TBD = 待定

PIC24F16KA102 系列

表 28-5: BOR 跳变点

标准工作条件（除非另外声明） 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ （工业级）							
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
DC19		VDD 跳变时的 BOR 电压	BOR = 00	—	2	—	V
			BOR = 01	—	3	—	V
			BOR = 10	—	2.7	—	V
			BOR = 11	—	2	—	V

表 28-6: 直流特性: 工作电流 (IDD)

直流特性			标准工作条件: 1.8V 至 3.6V（除非另外声明） 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		
IDD 电流						
DC20	195	—	μA	-40°C	1.8V	0.5 MIPS, Fosc = 1 MHz
DS20a				+25°C		
DC20b				+85°C		
DC20c	365	—	μA	-40°C	3.3V	
DC20d				+25°C		
DC20e				+85°C		
DC22	363	—	μA	-40°C	1.8V	1 MIPS, Fosc = 2 MHz
DC22a				+25°C		
DC22b				+85°C		
DC22c	695	—	μA	-40°C	3.3V	
DC22d				+25°C		
DC22e				+85°C		
DC23	8.15	—	mA	-40°C	2.5V	16 MIPS, Fosc = 32 MHz
DC23a				+25°C		
DC23b				+85°C		
DC23c	11	—	mA	-40°C	3.3V	
DC23d				+25°C		
DC23e				+85°C		
DC27	2.25	—	mA	-40°C	2.5V	FRC（4 MIPS）， Fosc = 8 MHz
DC27a				+25°C		
DC27b				+85°C		
DC27c	3.05	—	mA	-40°C	3.3V	
DC27d				+25°C		
DC27e				+85°C		

注 1: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。

注 2: 工作参数:

- EC 模式, 时钟输入使用轨到轨方波驱动
- I/O 配置为输出, 驱动为低电平
- MCLR – VDD
- 禁止 WDT 和 FSCM
- SRAM、程序和数据存储有效
- 对应于正被测量的模块的 PMD 位除外, 所有其他 PMD 位均置 1

表 28-6: 直流特性: 工作电流 (IDD) (续)

直流特性			标准工作条件: 1.8V 至 3.6V（除非另外声明） 工作温度 -40℃ ≤ TA ≤ +85℃（工业级）			
参数编号	典型值 ⁽¹⁾	最大值	单位	条件		
IDD 电流						
DC31	8	—	μA	-40℃	1.8V	LPRC（31 kHz）
DC31a				+25℃		
DC31b				+85℃		
DC31c	15	—	μA	-40℃	3.3V	
DC31d				+25℃		
DC31e				+85℃		

注 1: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。

2: 工作参数:

- EC 模式, 时钟输入使用轨到轨方波驱动
- I/O 配置为输出, 驱动为低电平
- MCLR – VDD
- 禁止 WDT 和 FSCM
- SRAM、程序和数据存储器有效
- 对应于正被测量的模块的 PMD 位除外, 所有其他 PMD 位均置 1

表 28-7: 直流特性: 空闲电流 (IDLE)

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 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 (IDLE): 内核不工作、时钟工作时的基本电流, PMD 位置 1 ⁽²⁾						
DC40	48	—	μA	-40°C	1.8V	0.5 MIPS, Fosc = 1 MHz
DC40a				+25°C		
DC40c				+85°C		
DC40d	106	—	μA	-40°C	3.3V	
DC40e				+25°C		
DC40g				+85°C		
DC42	94	—	μA	-40°C	1.8V	1 MIPS, Fosc = 2 MHz
DC42a				+25°C		
DC42c				+85°C		
DC42d	160	—	μA	-40°C	3.3V	
DC42e				+25°C		
DC43g				+85°C		

注 1: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。

2: 工作参数:

- 内核不工作
- EC 模式, 时钟输入使用轨到轨方波驱动
- I/O 配置为输出, 驱动为低电平
- MCLR – VDD
- 禁止 WDT 和 FSCM
- SRAM、程序和数据存储器有效
- 对应于正被测量的模块的 PMD 位除外, 所有其他 PMD 位均置 1

PIC24F16KA102 系列

表 28-7: 直流特性: 空闲电流 (IDLE) (续)

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 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 (IDLE): 内核不工作、时钟工作时的基本电流, PMD 位置 1 ⁽²⁾						
DC43	2.3	—	mA	-40°C	2.5V	16 MIPS, FOSC = 32 MHz
DC43a				+25°C		
DC43c				+85°C		
DC43d	3.1	—	mA	-40°C	3.3V	
DC43e				+25°C		
DC43g				+85°C		
DC47	0.73	—	mA	-40°C	2.5V	FRC (4 MIPS), FOSC = 8 MHz
DC47a				+25°C		
DC47c				+85°C		
DC47d	0.95	—	mA	-40°C	3.3V	
DC47e				+25°C		
DC47g				+85°C		
DC50	2	—	μA	-40°C	1.8V	LPRC (31 kHz)
DC50a				+25°C		
DC50c				+85°C		
DC50d	4	—	μA	-40°C	3.3V	
DC50e				+25°C		
DC50g				+85°C		

注 1: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。

2: 工作参数:

- 内核不工作
- EC 模式, 时钟输入使用轨到轨方波驱动
- I/O 配置为输出, 驱动为低电平
- MCLR – VDD
- 禁止 WDT 和 FSCM
- SRAM、程序和数据存储器有效
- 对应于正被测量的模块的 PMD 位除外, 所有其他 PMD 位均置 1

表 28-8: 直流特性: 掉电电流 (IPD)

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明)			
			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): PMD 位置 1, PMSLP 位为 0 ⁽²⁾						
DC60	0.025	—	μA	-40°C	1.8V	基本掉电电流 (休眠) ⁽³⁾
DC60a				+25°C		
DC60b				+85°C		
DC60c	0.040	—	μA	-40°C	2.5V	
DC60d				+25°C		
DC60e				+85°C		
DC60f	0.105	—	μA	-40°C	3.3V	
DC60g				+25°C		
DC60h				+85°C		
DC70	0.020	—	μA	-40°C	1.8V	深度休眠时的基本电流
DC70a				+25°C		
DC70b				+85°C		
DC70c	0.024	—	μA	-40°C	2.5V	
DC70d				+25°C		
DC70e				+85°C		
DC70f	0.035	—	μA	-40°C	3.3V	
DC70g				+25°C		
DC70h				+85°C		
DC61	1.20	—	μA	-40°C	1.8V	看门狗定时器电流: WDT ⁽⁴⁾
DC61a				+25°C		
DC61b				+85°C		
DC61c	1.30	—	μA	-40°C	2.5V	
DC61d				+25°C		
DC61e				+85°C		
DC61f	1.75	—	μA	-40°C	3.3V	
DC61g				+25°C		
DC61h				+85°C		

- 注 1:** 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。
- 2:** 基本 IPD 是在所有外设和时钟都关闭的条件下进行测量的。所有 I/O 引脚配置为输入且被上拉到高电平。WDT 等外设也都被关闭。
- 3:** Δ 电流为当模块使能时额外消耗的电流。此电流应被加到基本 IPD 电流。
- 4:** 电流仅适用于休眠模式。
- 5:** 电流适用于休眠和深度休眠模式。
- 6:** 电流仅适用于深度休眠模式。

PIC24F16KA102 系列

表 28-8: 直流特性: 掉电电流 (IPD) (续)

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明)			
			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): PMD 位置 1, PMSLP 位为 0 ⁽²⁾						
DC62	0.450	—	μA	-40°C	1.8V	Timer1, 带 32 kHz 晶振: T132 (SOSC – LP)
DC62a				+25°C		
DC62b				+85°C		
DC62c	0.570	—	μA	-40°C	2.5V	
DC62d				+25°C		
DC62e				+85°C		
DC62f	0.730	—	μA	-40°C	3.3V	
DC62g				+25°C		
DC62h				+85°C		
DC64	5.5	—	μA	-40°C	1.8V	HLVD ⁽⁴⁾
DC64a				+25°C		
DC64b				+85°C		
DC64c	5.9	—	μA	-40°C	2.5V	
DC64d				+25°C		
DC64e				+85°C		
DC64f	6.2	—	μA	-40°C	3.3V	
DC64g				+25°C		
DC64h				+85°C		
DC63	4.1	—	μA	-40°C	2.5V	BOR ⁽⁴⁾
DC63a				+25°C		
DC63b				+85°C		
DC63c	4.5	—	μA	-40°C	3.3V	
DC63d				+25°C		
DC63e				+85°C		
DC62	0.49	—	μA	-40°C	1.8V	RTCC ⁽⁵⁾
DC62a				+25°C		
DC62b				+85°C		
DC62c	0.62	—	μA	-40°C	2.5V	
DC62d				+25°C		
DC62e				+85°C		
DC62f	0.80	—	μA	-40°C	3.3V	
DC62g				+25°C		
DC62h				+85°C		

- 注 1: 除非另外声明, 否则 “典型值” 栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。
- 2: 基本 IPD 是在所有外设和时钟都关闭的条件下进行测量的。所有 I/O 引脚配置为输入且被上拉到高电平。WDT 等外设也都被关闭。
- 3: Δ 电流为当模块使能时额外消耗的电流。此电流应被加到基本 IPD 电流。
- 4: 电流仅适用于休眠模式。
- 5: 电流适用于休眠和深度休眠模式。
- 6: 电流仅适用于深度休眠模式。

表 28-8: 直流特性: 掉电电流 (IPD) (续)

直流特性			标准工作条件: 1.8V 至 3.6V（除非另外声明） 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）：PMD 位置 1，PMSLP 位为 0 ⁽²⁾						
DC70	0.045	—	μA	-40°C	1.8V	LPBOR ⁽⁴⁾
DC70a				+25°C		
DC70b				+85°C		
DC70c	0.050	—	μA	-40°C	2.5V	
DC70d				+25°C		
DC70e				+85°C		
DC70f	0.095	—	μA	-40°C	3.3V	
DC70g				+25°C		
DC70h				+85°C		
DC71	0.40	—	μA	-40°C	1.8V	深度休眠看门狗定时器： DSWDT（SOSC – LP） ⁽⁶⁾
DC71a				+25°C		
DC71b				+85°C		
DC71c	0.50	—	μA	-40°C	2.5V	
DC71d				+25°C		
DC71e				+85°C		
DC71f	0.65	—	μA	-40°C	3.3V	
DC71g				+25°C		
DC71h				+85°C		
DC72	0.008	—	μA	-40°C	1.8V	深度休眠 BOR：DSBOR ⁽⁶⁾
DC72a				+25°C		
DC72b				+85°C		
DC72c	0.02	—	μA	-40°C	2.5V	
DC72d				+25°C		
DC72e				+85°C		
DC72f	0.011	—	μA	-40°C	3.3V	
DC72g				+25°C		
DC72h				+85°C		

- 注 1:** 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。
- 2:** 基本 IPD 是在所有外设和时钟都关闭的条件下进行测量的。所有 I/O 引脚配置为输入且被上拉到高电平。WDT 等外设也都被关闭。
- 3:** Δ 电流为当模块使能时额外消耗的电流。此电流应被加到基本 IPD 电流。
- 4:** 电流仅适用于休眠模式。
- 5:** 电流适用于休眠和深度休眠模式。
- 6:** 电流仅适用于深度休眠模式。

PIC24F16KA102 系列

表 28-9: 直流特性: I/O 引脚输入规范

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (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
DI10	V _{IL}	输入低电压 ⁽⁴⁾	—	—	—	—	
DI15		I/O 引脚	V _{SS}	—	0.2 V _{DD}	V	
DI16		MCLR	V _{SS}	—	0.2 V _{DD}	V	
DI17		OSCI (XT 模式)	V _{SS}	—	0.2 V _{DD}	V	
DI17		OSCI (HS 模式)	V _{SS}	—	0.2 V _{DD}	V	
DI18		I/O 引脚, 带 I ² C™ 缓冲器	V _{SS}	—	0.3 V _{DD}	V	禁止 SMBus
DI19		I/O 引脚, 带 SMBus 缓冲器	V _{SS}	—	0.8	V	使能 SMBus
DI20	V _{IH}	输入高电压 ⁽⁴⁾	—	—	—	—	
		I/O 引脚:					
		带模拟功能	0.8 V _{DD}	—	V _{DD}	V	
		仅数字功能	0.8 V _{DD}	—	V _{DD}	V	
DI25		MCLR	0.8 V _{DD}	—	V _{DD}	V	
DI26		OSCI (XT 模式)	0.7 V _{DD}	—	V _{DD}	V	
DI27		OSCI (HS 模式)	0.7 V _{DD}	—	V _{DD}	V	
DI28		I/O 引脚, 带 I ² C 缓冲器:					
		带模拟功能	0.7 V _{DD}	—	V _{DD}	V	
		仅数字功能	0.7 V _{DD}	—	V _{DD}	V	
DI29		I/O 引脚, 带 SMBus 缓冲器	2.1	—	V _{DD}	V	2.5V ≤ V _{PIN} ≤ V _{DD}
DI30	ICNPU	CNx 上拉电流	50	250	400	μA	V _{DD} = 3.3V, V _{PIN} = V _{SS}
DI50	I _{IL}	输入泄漏电流 ^(2,3)	—	TBD	TBD	μA	V _{SS} ≤ V _{PIN} ≤ V _{DD} , 引脚处于高阻态
		I/O 端口	—	TBD	TBD	μA	
DI51		模拟输入引脚	—	TBD	TBD	μA	V _{SS} ≤ V _{PIN} ≤ V _{DD} , 引脚处于高阻态
DI55		MCLR	—	TBD	TBD	μA	V _{SS} ≤ V _{PIN} ≤ V _{DD}
DI56		OSCI	—	TBD	TBD	μA	V _{SS} ≤ V _{PIN} ≤ V _{DD} , XT 和 HS 模式

图注: TBD = 待定

- 注 1: 除非另外声明, 否则“典型值”栏中的数据均为3.3V和25°C条件下的值。这些参数仅供设计参考, 未经测试。
- 2: MCLR 引脚上的泄漏电流主要取决于所施加电压。规定电压为正常工作条件下的电压。在不同的输入电压下可能测得更高的泄漏电流。
- 3: 负电流定义为引脚的拉电流。
- 4: 请参见表 1-2 了解 I/O 引脚缓冲器类型。

表 28-10: 直流特性: I/O 引脚输出规范

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 -40°C ≤ TA ≤ +85°C (工业级)				
参数编号	符号	特性	最小值	典型值 ⁽¹⁾	最大值	单位	条件
DO10	VOL	输出低电压 所有 I/O 引脚	—	—	0.4	V	IOL = 6.5 mA, VDD = 3.6V IOL = 3.5 mA, VDD = 2.0V IOL = 8.0 mA, VDD = 3.6V IOL = 4.5 mA, VDD = 1.8V
DO16			—	—	0.4	V	
DO16		OSC2/CLKO	—	—	0.4	V	
			—	—	0.4	V	
DO20	VOH	输出高电压 所有 I/O 引脚	3	—	—	V	IOH = -3.0 mA, VDD = 3.6V IOH = -1.0 mA, VDD = 2.0V IOH = -2.5 mA, VDD = 3.6V IOH = -1.0 mA, VDD = 2.0V
DO26			1.8	—	—	V	
DO26		OSC2/CLKO	3	—	—	V	
			1.8	—	—	V	

注 1: 除非另外声明, 否则“典型值”栏中的数据均为 25°C 条件下的值。这些参数仅供设计参考, 未经测试。

表 28-11: 直流特性: 程序存储器

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (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
D130	EP	闪存程序存储器 单元耐擦写能力	1000 ⁽²⁾	10000	—	E/W	V _{MIN} = 最小工作电压 假设没有违反其他规范
D131	VPR	读操作时的 VDD	V _{MIN}	—	3.6	V	
D133A	TIW	自定时写周期时间	—	2	—	ms	
D134	TRETD	特性保持时间	40	—	—	年	
D135	IDDP	编程时的供电电流	—	10	—	mA	

注 1: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。

2: 自写和块擦除。

表 28-12: 直流特性: 数据 EEPROM 存储器

直流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (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
D140	EPD	数据 EEPROM 存储器 单元耐擦写能力	100000	1000000	—	E/W	V _{MIN} = 最小工作电压 假设没有违反其他规范
D141	VPRD	读操作时的 VDD	V _{MIN}	—	3.6	V	
D143A	TIWD	自定时写周期时间	—	4	—	ms	
D143B	TREF	刷新前写 / 擦除的总周期数	—	10M	—	E/W	
D144	TRETD	特性保持时间	40	—	—	年	
D145	IDDPD	编程时的供电电流	—	7	—	mA	

注 1: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。

PIC24F16KA102 系列

表 28-13: 比较器直流规范

工作条件: $2.0V < V_{DD} < 3.6V$, $-40^{\circ}C < T_A < +85^{\circ}C$ (除非另外声明)							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
D300	V_{IOFF}	输入失调电压 *	—	TBD	TBD	mV	
D301	V_{ICM}	输入共模电压 *	TBD	TBD	TBD	V	
D302	CMRR	共模抑制比 *	TBD	TBD	—	dB	

图注: TBD = 待定

* 参数为特性值, 未经测试。

表 28-14: 比较器参考电压直流规范

工作条件: $2.0V < V_{DD} < 3.6V$, $-40^{\circ}C < T_A < +85^{\circ}C$ (除非另外声明)							
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
VRD310	CVRES	分辨率	TBD	TBD	TBD	LSb	
VRD311	CVRAA	绝对精度	TBD	TBD	TBD	LSb	
VRD312	CVRUR	单位电阻值 (R)	—	2k	—	Ω	

图注: TBD = 待定

28.2 交流特性和时序参数

本节包含的信息说明了 PIC24F16KA102 系列器件的交流特性和时序参数。

表 28-15: 温度和电压规范——交流

交流特性	标准工作条件: 1.8V 至 3.6V (除非另外声明)
	工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (工业级)
	工作电压 V_{DD} 范围如第 28.1 节“直流特性”中所述。

图 28-2: 器件时序规范的负载条件

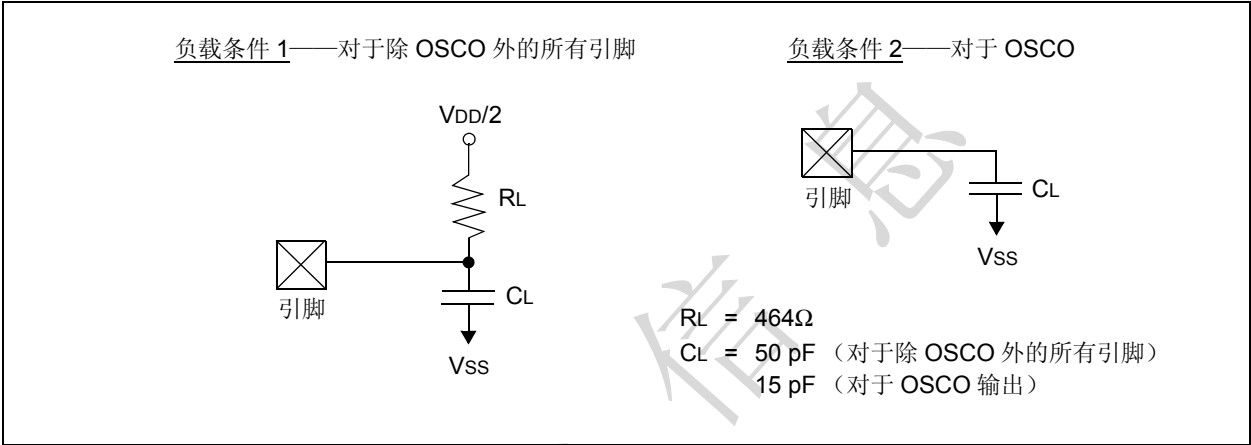


表 28-16: 输出引脚上的容性负载要求

参数编号	符号	特性	最小值	典型值 ⁽¹⁾	最大值	单位	条件
DO50	Cosc2	OSCO/CLKO 引脚	—	—	15	pF	当外部时钟用于驱动 OSCI 时处于 XT 和 HS 模式下
DO56	CIO	所有 I/O 引脚和 OSCO	—	—	50	pF	EC 模式
DO58	CB	SCLx 和 SDAx	—	—	400	pF	在 I ² C™ 模式下

注 1: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。

PIC24F16KA102 系列

图 28-3: 外部时钟时序

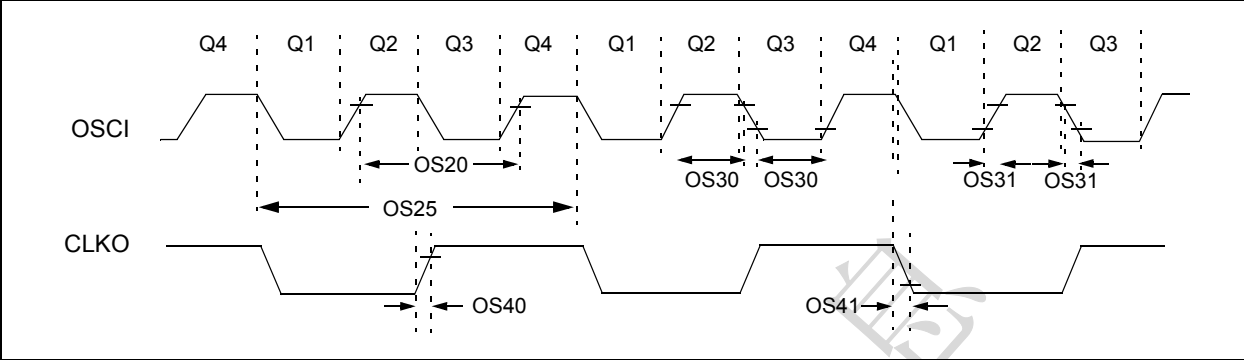


表 28-17: 外部时钟时序要求

交流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (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 (外部时钟仅允许运行于 EC 模式)	DC 4	— —	32 8	MHz MHz	EC ECPLL
		振荡器频率	0.2 4 4 31	— — — —	4 25 8 33	MHz MHz MHz kHz	XT HS HSPLL SOSC
OS20	Tosc	$T_{osc} = 1/F_{osc}$	—	—	—	—	Fosc 值见参数 OS10
OS25	Tcy	指令周期 ⁽²⁾	62.5	—	DC	ns	
OS30	TosL, TosH	外部时钟输入 (OSCI) 高电平或低电平时间	$0.45 \times T_{osc}$	—	—	ns	EC
OS31	TosR, TosF	外部时钟输入 (OSCI) 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: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。
- 2: 指令周期 (Tcy) 等于输入振荡器时钟周期的两倍。所有规定值均为基于针对特定振荡器类型, 器件在标准工作条件下执行代码时的特性数据。超出这些规定的限定值, 可能导致振荡器运行不稳定和 / 或导致电流消耗超出预期值。所有器件在测试“最小”值时, 都在 OSCI/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”(无时钟)。
- 3: 测量在 EC 模式下进行。在 OSC0 引脚上测量 CLKO 信号。CLKO 在 Q1-Q2 周期 (1/2 Tcy) 中为低电平, 在 Q3-Q4 周期 (1/2 Tcy) 中为高电平。

表 28-18: PLL 时钟时序规范 (VDD = 1.8V 至 3.6V)

交流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 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	4	—	8	MHz	ECPLL 和 HSPLL 模式, -40°C ≤ TA ≤ +85°C
OS51	Fsys	PLL 输出频率范围	16	—	32	MHz	-40°C ≤ TA ≤ +85°C
OS52	TLOCK	PLL 起振时间 (锁定时间)	—	—	2	ms	—
OS53	DCLK	CLKO 稳定性 (抗抖动性)	-2	1	2	%	在 100 ms 时间段内测量

注 1: 这些参数为特性值, 但生产时未经测试。

2: 除非另外声明, 否则“典型值”栏中的数据均为 3.3V 和 25°C 条件下的值。这些参数仅供设计参考, 未经测试。

表 28-19: 交流特性: 内部 RC 精度

交流特性		标准工作条件: 1.8V 至 3.6V (除非另外声明) 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	FRC 频率为 8 MHz 时的内部 FRC 精度 ⁽¹⁾						
	FRC	-2	—	2	%	+25°C	3.0V ≤ VDD ≤ 3.6V
		-5	—	5	%	-40°C ≤ TA ≤ +85°C	

注 1: 频率在 25°C 和 3.3V 条件下校准。OSCTUN 位可用来补偿温度漂移。

表 28-20: 交流特性: 内部 RC 精度

交流特性		标准工作条件: 1.8V 至 3.6V (除非另外声明) 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	
F21	LPRC 频率为 31 kHz ⁽¹⁾						
		-15	—	15	%	+25°C	3.0V ≤ VDD ≤ 3.6V
		-15	—	15	%	-40°C ≤ Ta ≤ +85°C	

注 1: LPRC 频率将随 VDD 的变化而变化。

PIC24F16KA102 系列

图 28-4: CLKO 和 I/O 时序特性

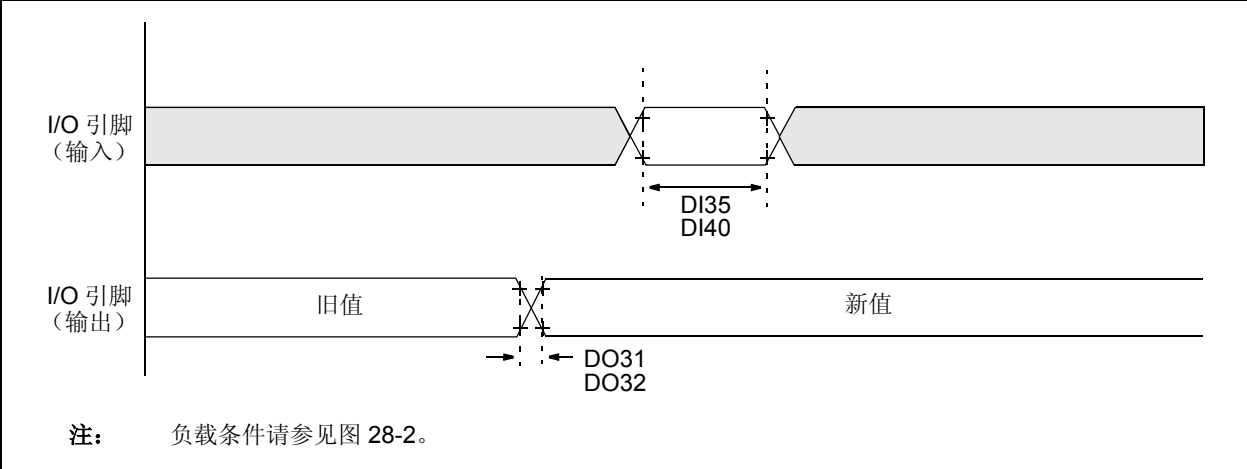


表 28-21: CLKO 和 I/O 时序要求

交流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (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
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: 除非另外声明, 否则 “典型值” 栏中的数据均为 3.3V 和 25°C 条件下的值。

表 28-22: 比较器时序

参数编号	符号	特性	最小值	典型值	最大值	单位	备注
300	TRESP	响应时间 ⁽¹⁾	TBD	150	TBD	ns	
301	TMC2OV	比较器模式变为输出有效的时间 [*]	TBD	—	TBD	μs	

图注: TBD = 待定

* 参数为特性值, 未经测试。

注 1: 响应时间是在比较器的一个输入端电压为 $(V_{DD} - 1.5)/2$, 而另一个输入端从 V_{SS} 跳变到 V_{DD} 时测得的。

表 28-23: 比较器参考电压稳定时间规范

参数编号	符号	特性	最小值	典型值	最大值	单位	备注
VR310	TSET	稳定时间 ⁽¹⁾	TBD	—	TBD	μs	

图注: TBD = 待定

注 1: 稳定时间是在 $CVRR = 1$ 并且 $CVR<3:0>$ 位从 0000 跳变到 1111 时测得的。

表 28-24: CTMU 电流源规范

直流特性			标准工作条件: 2.0V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (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
	IOUT1	CTMU 电流源, 基本范围	TBD	550	TBD	nA	CTMUICON<1:0> = 01
	IOUT2	CTMU 电流源, 10x 范围	TBD	5.5	TBD	μA	CTMUICON<1:0> = 10
	IOUT3	CTMU 电流源, 100x 范围	TBD	55	TBD	μA	CTMUICON<1:0> = 11

图注: TBD = 待定

注 1: 电流微调范围的中点 (CTMUICON<7:2> = 000000) 为标称值。

PIC24F16KA102 系列

表 28-25: ADC 模块规范

交流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (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	VDD - 0.3 和 1.8 中的 较大值	—	VDD + 0.3 和 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	参考电压高电压	AVSS + 1.7	—	AVDD	V	
AD06	VREFL	参考电压低电压	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	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	10 位
ADC 精度							
AD20b	NR	分辨率	TBD	10	TBD	位	
AD21b	INL	积分非线性误差	TBD	± 1	TBD	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD22b	DNL	微分非线性误差	TBD	± 1	TBD	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD23b	GERR	增益误差	TBD	± 1	TBD	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD24b	Eoff	失调误差	TBD	± 1	TBD	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD25b		单调性 ⁽¹⁾	TBD	—	TBD	—	保证

图注: TBD = 待定

注 1: ADC 转换结果不会因输入电压的增加而减小, 并且不会丢失编码。

2: 测量采用外部 VREF+ 和 VREF- 用作 ADC 参考电压。

表 28-26: ADC 转换时序要求⁽¹⁾

交流特性			标准工作条件: 1.8V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +85^{\circ}\text{C}$ (工业级)				
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
时钟参数							
AD50	TAD	ADC 时钟周期	75	—	—	ns	$T_{CY} = 75 \text{ ns}$, AD1CON3 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	$A_{VDD} \geq 2.7V$
AD57	TSAMP	采样时间	—	1	—	TAD	
AD58	TACQ	采集时间	750	—	—	ns	(注 2)
AD59	TSWC	转换到采样的切换时间	—	—	(注 3)		
AD60	TDIS	电容放电时间	0.5	—	—	TAD	
时钟参数							
AD61	TPSS	从采样位 (SAMP) 置 1 到采样启动的时间	2	—	3	TAD	

- 注 1: 因为采样电容最终将释放电荷, 因此低于 10 kHz 的时钟频率可能影响线性性能, 尤其是在温度较高时。
- 2: 转换完成后当电压满量程变化时 (V_{DD} 至 V_{SS} 或 V_{SS} 至 V_{DD}), 保持电容采集一个“新”输入电压所需的时间。
- 3: 在器件时钟的下一个周期。

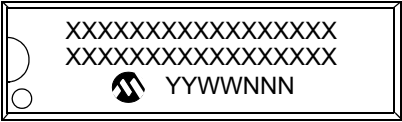
PIC24F16KA102 系列

注:

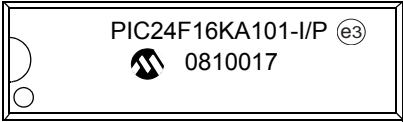
29.0 封装信息

29.1 封装标识信息

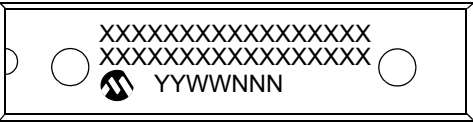
20 引脚 PDIP



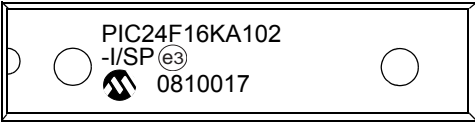
示例



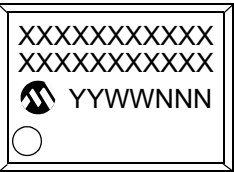
28 引脚 SPDIP



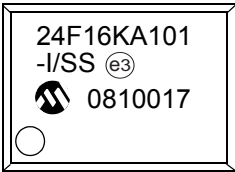
示例



20 引脚 SSOP



示例



28 引脚 SSOP



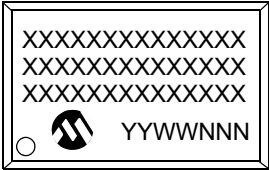
示例



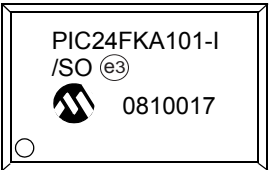
图注:	XX...X	客户指定信息
	Y	年份代码（日历年的最后一位数字）
	YY	年份代码（日历年的最后两位数字）
	WW	星期代码（一月一日的星期代码为“01”）
	NNN	以字母数字排序的追踪代码
	(e3)	雾锡（Matte Tin, Sn）的 JEDEC 无铅标志
	*	本封装为无铅封装。JEDEC 无铅标志 ((e3)) 标示于此种封装的外包装上。
注:	Microchip 元器件编号如果无法在同一行内完整标注，将换行标出，因此会限制表示客户信息的字符数。	

PIC24F16KA102 系列

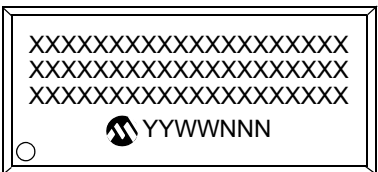
20 引脚 SOIC (0.300 英寸)



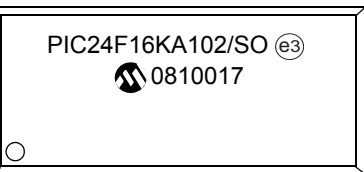
示例



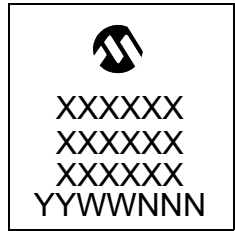
28 引脚 SOIC (0.300 英寸)



示例



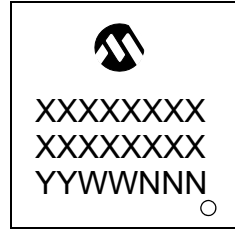
20 引脚 QFN



示例



28 引脚 QFN



示例

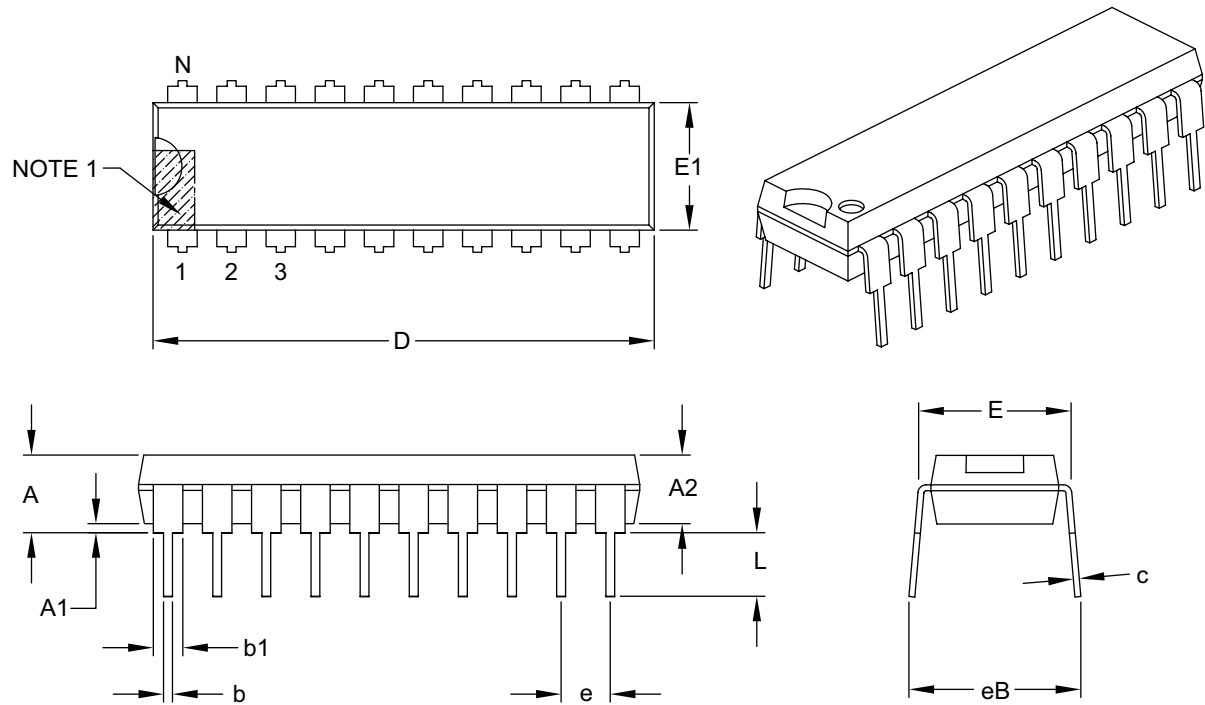


29.2 封装详细信息

以下部分将介绍各种封装的技术细节。

20引脚塑封双列直插式封装（P）——主体300 mil [PDIP]

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



Dimension Limits	Units	INCHES		
		MIN	NOM	MAX
Number of Pins	N	20		
Pitch	e	.100 BSC		
Top to Seating Plane	A	—	—	.210
Molded Package Thickness	A2	.115	.130	.195
Base to Seating Plane	A1	.015	—	—
Shoulder to Shoulder Width	E	.300	.310	.325
Molded Package Width	E1	.240	.250	.280
Overall Length	D	.980	1.030	1.060
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.045	.060	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	—	—	.430

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
- Dimensioning and tolerancing per ASME Y14.5M.

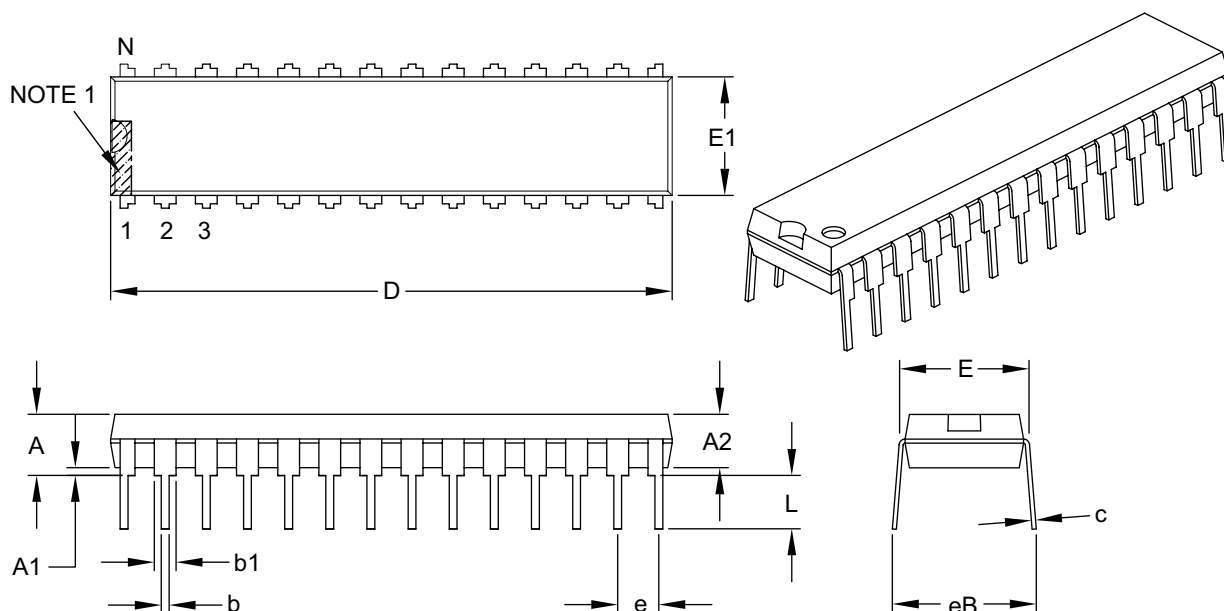
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-019B

PIC24F16KA102 系列

28引脚窄型塑封双列直插式封装（SP）——主体300 mil [SPDIP]

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	.100 BSC		
Top to Seating Plane	A	—	—	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	—	—
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	—	—	.430

Notes:

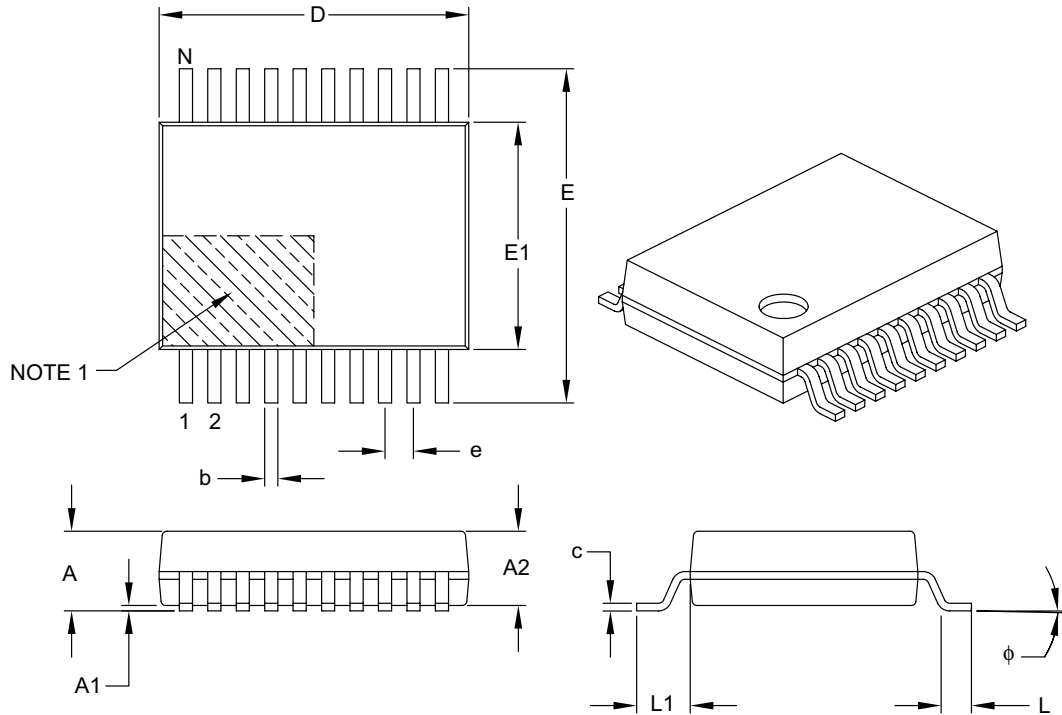
- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

20引脚塑封缩小型小外形封装（SS）——主体5.30 mm [SSOP]

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



		Units	MILLIMETERS		
Dimension Limits			MIN	NOM	MAX
Number of Pins	N		20		
Pitch	e		0.65 BSC		
Overall Height	A		—	—	2.00
Molded Package Thickness	A2		1.65	1.75	1.85
Standoff	A1		0.05	—	—
Overall Width	E		7.40	7.80	8.20
Molded Package Width	E1		5.00	5.30	5.60
Overall Length	D		6.90	7.20	7.50
Foot Length	L		0.55	0.75	0.95
Footprint	L1		1.25 REF		
Lead Thickness	c		0.09	—	0.25
Foot Angle	φ		0°	4°	8°
Lead Width	b		0.22	—	0.38

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

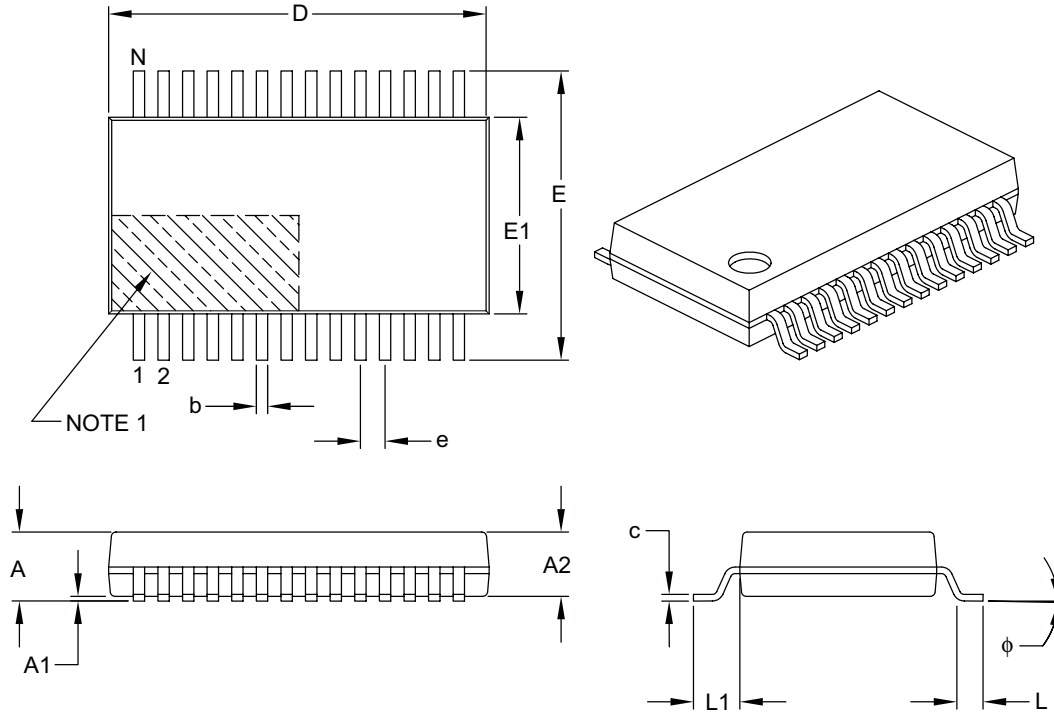
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-072B

PIC24F16KA102 系列

28引脚塑封缩小型小外形封装（SS）——主体5.30 mm [SSOP]

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	0.65 BSC		
Overall Height	A	—	—	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	—	—
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	9.90	10.20	10.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1	1.25 REF		
Lead Thickness	c	0.09	—	0.25
Foot Angle	φ	0°	4°	8°
Lead Width	b	0.22	—	0.38

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

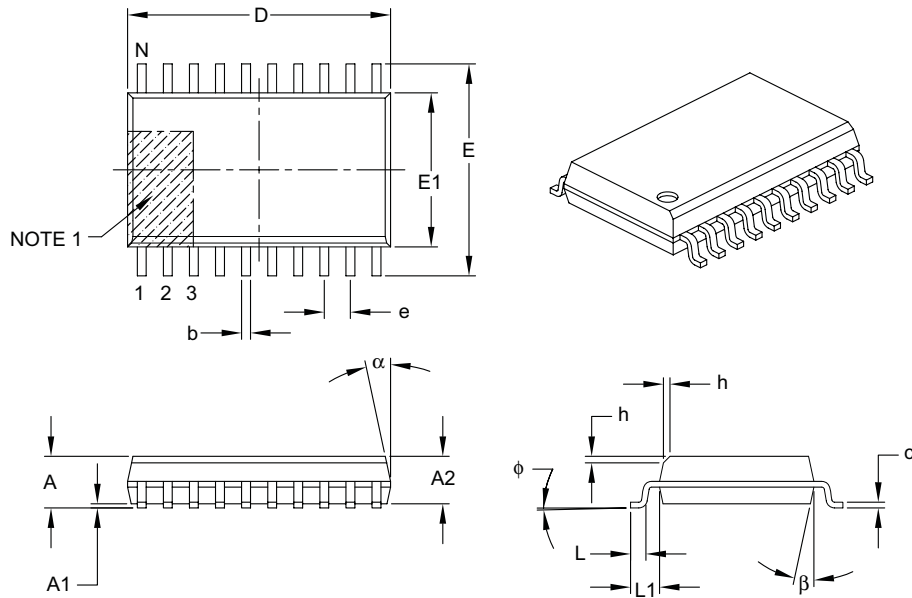
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

20引脚塑封宽条小外形封装（SO）——主体7.50 mm [SOIC]

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	20		
Pitch	e	1.27 BSC		
Overall Height	A	—	—	2.65
Molded Package Thickness	A2	2.05	—	—
Standoff §	A1	0.10	—	0.30
Overall Width	E	10.30 BSC		
Molded Package Width	E1	7.50 BSC		
Overall Length	D	12.80 BSC		
Chamfer (optional)	h	0.25	—	0.75
Foot Length	L	0.40	—	1.27
Footprint	L1	1.40 REF		
Foot Angle	φ	0°	—	8°
Lead Thickness	c	0.20	—	0.33
Lead Width	b	0.31	—	0.51
Mold Draft Angle Top	α	5°	—	15°
Mold Draft Angle Bottom	β	5°	—	15°

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

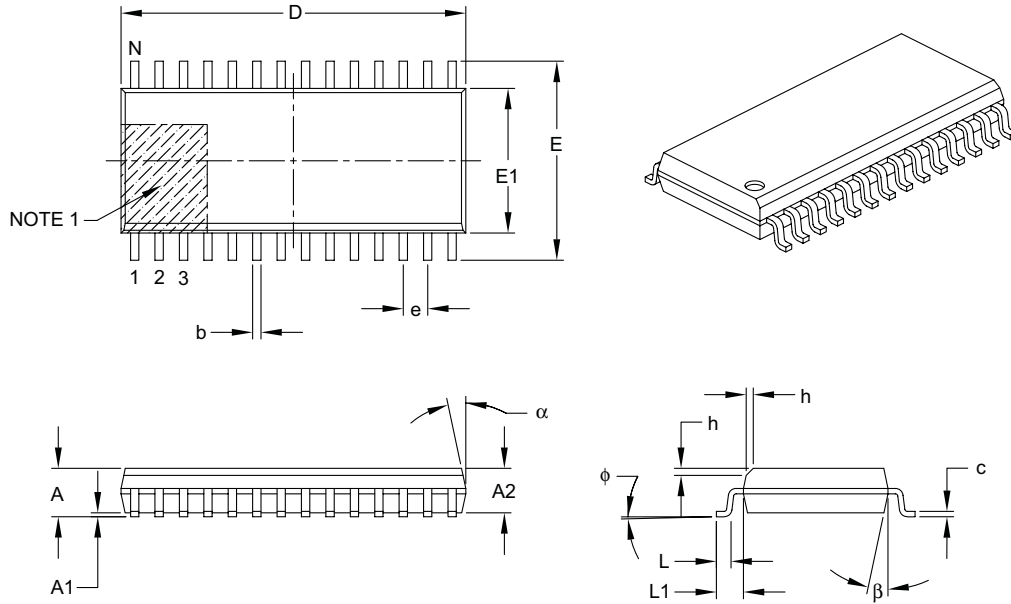
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-094B

PIC24F16KA102 系列

28引脚塑封宽条小外形封装（SO）——主体7.50 mm [SOIC]

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



Dimension Limits	Units	MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	1.27 BSC		
Overall Height	A	—	—	2.65
Molded Package Thickness	A2	2.05	—	—
Standoff §	A1	0.10	—	0.30
Overall Width	E	10.30 BSC		
Molded Package Width	E1	7.50 BSC		
Overall Length	D	17.90 BSC		
Chamfer (optional)	h	0.25	—	0.75
Foot Length	L	0.40	—	1.27
Footprint	L1	1.40 REF		
Foot Angle Top	φ	0°	—	8°
Lead Thickness	c	0.18	—	0.33
Lead Width	b	0.31	—	0.51
Mold Draft Angle Top	α	5°	—	15°
Mold Draft Angle Bottom	β	5°	—	15°

Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- § Significant Characteristic.
- Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
- Dimensioning and tolerancing per ASME Y14.5M.

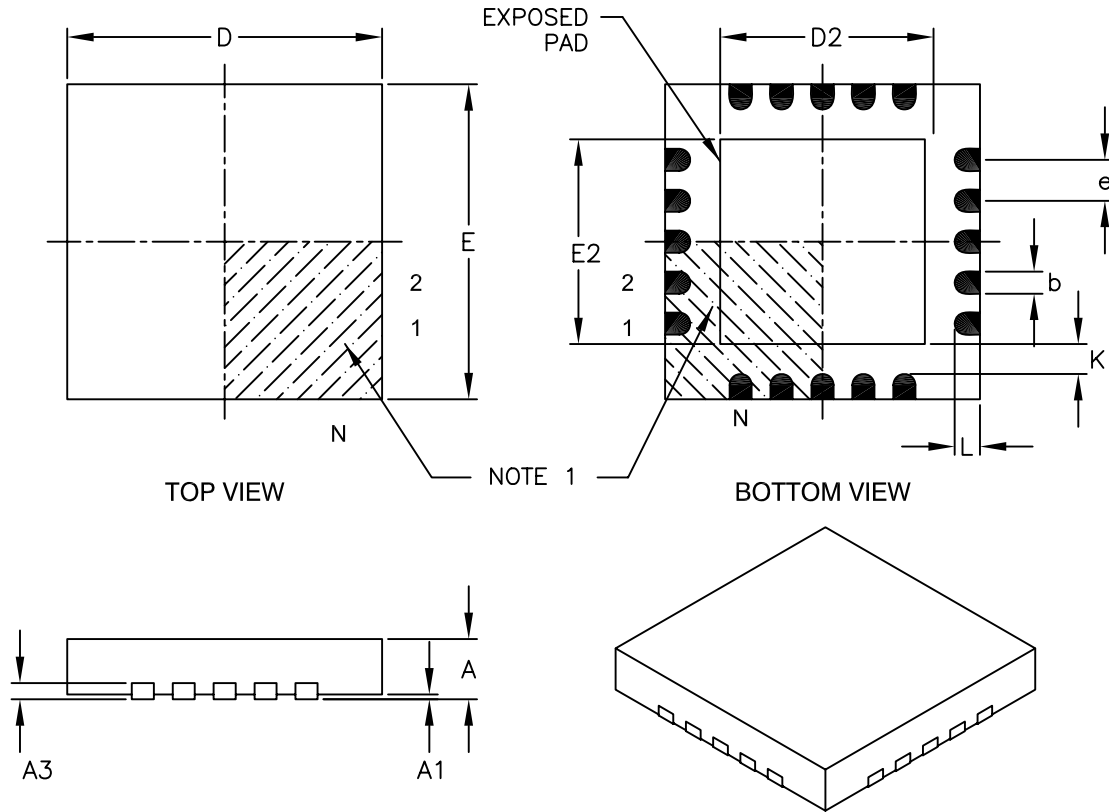
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-052B

20引脚塑封正方扁平无脚封装（MQ）——主体5x5x0.9 mm [QFN]

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	20		
Pitch	e	0.65 BSC		
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.20 REF		
Overall Width	E	5.00 BSC		
Exposed Pad Width	E2	3.15	3.25	3.35
Overall Length	D	5.00 BSC		
Exposed Pad Length	D2	3.15	3.25	3.35
Contact Width	b	0.25	0.30	0.35
Contact Length	L	0.35	0.40	0.45
Contact-to-Exposed Pad	K	0.20	-	-

Notes:

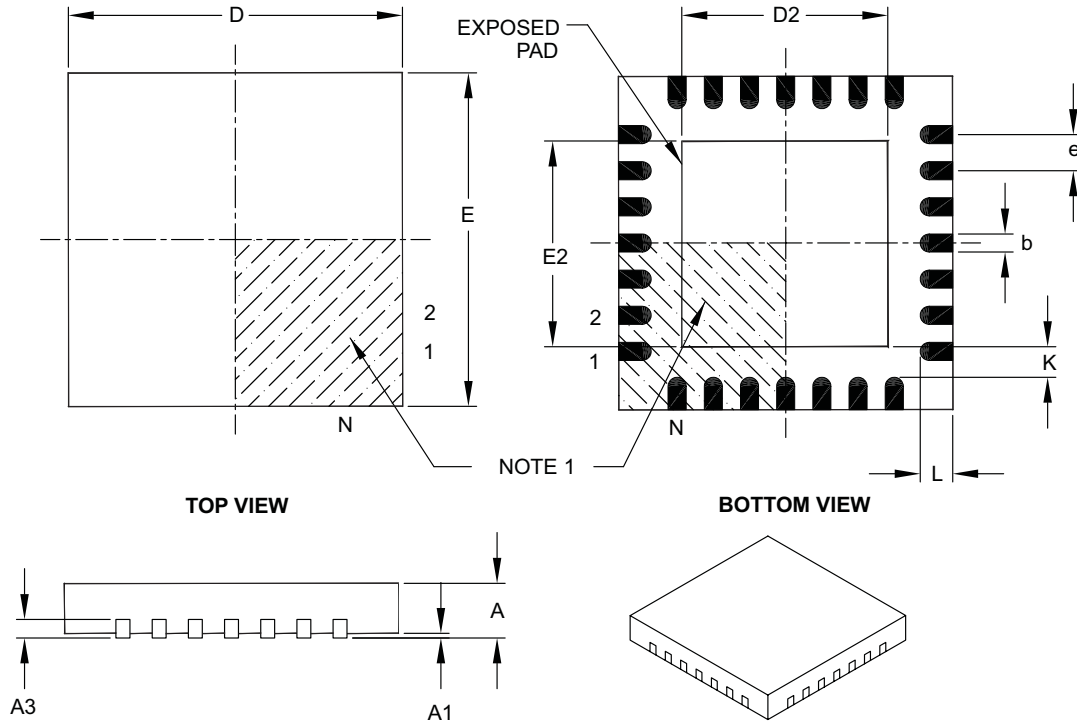
- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Package is saw singulated.
- Dimensioning and tolerancing per ASME Y14.5M.
BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-139B

PIC24F16KA102 系列

28引脚塑封正方扁平无脚封装（ML）——主体6x6 mm [QFN]，触点长度为0.55 mm

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	0.65 BSC		
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3	0.20 REF		
Overall Width	E	6.00 BSC		
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D	6.00 BSC		
Exposed Pad Length	D2	3.65	3.70	4.20
Contact Width	b	0.23	0.30	0.35
Contact Length	L	0.50	0.55	0.70
Contact-to-Exposed Pad	K	0.20	—	—

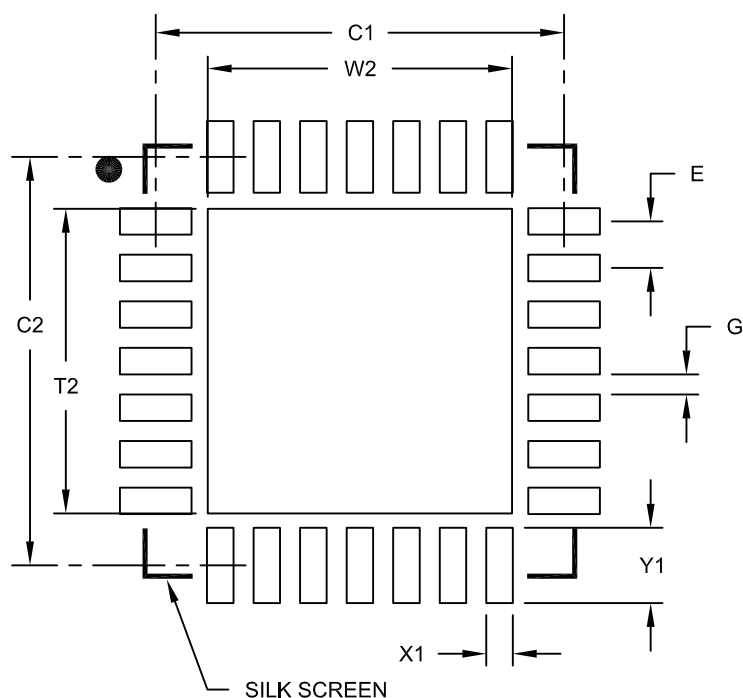
Notes:

- Pin 1 visual index feature may vary, but must be located within the hatched area.
- Package is saw singulated.
- 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 C04-105B

28引脚塑封正方扁平无脚封装（ML）——主体6x6 mm [QFN]，触点长度为0.55 mm

注：最新封装图请至<http://www.microchip.com/packaging>查看Microchip封装规范。



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.65 BSC		
Optional Center Pad Width	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2105A

PIC24F16KA102 系列

注:

附录 A: 版本历史

版本 A (2008 年 11 月)

PIC24F16KA102 系列器件的初始数据手册。

PIC24F16KA102 系列

注:

索引

A

A/D

10 位高速 A/D 转换器	163
模块规范	222
转换时序要求	223

A/D 转换器

模拟输入模型	170
传递函数	171

B

版本历史	237
比较器	173
比较器参考电压	177
变更通知客户服务	10
波特率发生器	
设置作为总线主器件	131

C

CPU

ALU	19
编程模型	15
控制寄存器	18
内核寄存器	16

CRC

节能模式下的操作	158
用户接口	158

CTMU

测量电容	179
测量时间	180
脉冲延时和产生	180

C 编译器

MPLAB C18	194
MPLAB C30	194

参考时钟输出	92
产品标识体系	12

程序存储器

存储器映射	21
地址空间	21

程序和数据存储空间	
接口	33

程序和数据存储器	
程序空间可视性	36
使用表指令访问	35

程序校验	191
------------	-----

充电时间测量单元。请参见 CTMU。

串行外设接口。请参见 SPI。

D

打盹模式	101
代码保护	191

代码示例

I/O 端口写 / 读	104
PWRSVAV 指令语法	96
编程闪存程序存储器的一个字	45
擦除程序存储器的一行，C 语言代码	43
擦除程序存储器的一行，汇编语言代码	42
单字擦除	50
将 RTCWREN 位置 1	146
启动编程序列，C 语言代码	45
启动编程序列，汇编语言代码	45
使用 TBLRD 命令读取数据 EEPROM	52
时钟切换的序列	92
数据 EEPROM 解锁序列	47

数据 EEPROM 批量擦除	51
写数据 EEPROM 的单个字	51
装载写缓冲区，C 语言代码	44
装载写缓冲区，汇编语言代码	44

电气特性

V/F 图	206
绝对最大额定值	205
温度工作条件	206
读者反馈	11

F

封装

标识	225
详细信息	227

G

高 / 低压检测

特性	207
高 / 低压检测 (HLVD)	161

公式

A/D 转换时钟周期	170
UART 波特率 (BRGH = 0)	138
UART 波特率 (BRGH = 1)	138
波特率重载计算	131
计算 PWM 周期	116
计算最大 PWM 分辨率	116
器件工作频率和 SPI 时钟速度之间的关系	128

H

汇编器

MPASM 汇编器	194
-----------------	-----

I

I/O 端口

并行 (PIO)	103
漏极开路配置	104
模拟端口引脚	104
输入电平变化通知	104

I²C

保留的地址	131
从地址掩码	131
时钟速率	131
引脚重映射选项	129
在单主机环境下作为主器件进行通信	129

J

寄存器

AD1CHS (A/D 输入选择)	168
AD1CON1 (A/D 控制 1)	165
AD1CON2 (A/D 控制 2)	166
AD1CON3 (A/D 控制 3)	167
AD1CSSL (A/D 输入扫描选择，低位字)	169
AD1PCFG (A/D 端口配置)	169
ALCFGRPT (闹钟配置)	149
ALMINSEC (闹钟分钟和秒值)	153
ALMTHDY (闹钟月和日值)	152
ALWDHR (闹钟星期和小时值)	152
CLKDIV (时钟分频)	89
CMSTAT (比较器状态)	176
CMXCON (比较器 x 控制)	175
CORCON (CPU 控制)	19
CORCON (内核控制)	62
CRCCON (CRC 控制)	159
CRCXOR (CRC XOR 多项式)	160

PIC24F16KA102 系列

CTMUCON (CTMU 控制)	181
CTMUICON (CTMU 电流控制)	182
CVRCON (比较器参考电压控制)	178
DEVID (器件 ID)	189
DEVREV (器件版本)	189
DSCON (深度休眠控制)	99
DSWSRC (深度休眠唤醒源)	100
FBS (引导段配置)	183
FDS (深度休眠配置)	188
FGS (通用段配置)	184
FICD (在线调试器配置)	187
FOSCSSEL (振荡器选择配置)	184
FOSC (振荡器配置)	185
FPOR (复位配置)	187
FWDT (看门狗定时器配置)	186
HLVDCON (高/低压检测控制)	162
I2C1CON (I2C1 控制)	132
I2C1MSK (I2C1 从模式地址掩码)	136
I2C1STAT (I2C1 状态)	134
IC1CON (输入捕捉 1 控制)	114
IEC0 (中断允许控制 0)	69
IEC1 (中断允许控制 1)	70
IEC3 (中断允许控制 3)	71
IEC4 (中断允许控制 4)	72
IFS0 (中断标志状态 0)	65
IFS1 (中断标志状态 1)	66
IFS3 (中断标志状态 3)	67
IFS4 (中断标志状态 4)	68
INTCON1 (中断控制 1)	63
INTTREG 中断控制和状态	83
IPC0 (中断优先级控制 0)	73
IPC1 (中断优先级控制 1)	74
IPC15 (中断优先级控制 15)	80
IPC16 (中断优先级控制 16)	81
IPC18 (中断优先级控制 18)	82
IPC19 (中断优先级控制 19)	82
IPC2 (中断优先级控制 2)	75
IPC3 (中断优先级控制 3)	76
IPC4 (中断优先级控制 4)	77
IPC5 (中断优先级控制 5)	78
IPC7 (中断优先级控制 7)	79
MINSEC (RTCC 分钟和秒值)	151
MTHDY (RTCC 月和日值)	150
NVMCON (非易失性存储器控制)	48
NVMCON (闪存控制)	41
OC1CON (输出比较 1 控制)	119
OSCCON (振荡器控制)	87
OSCTUN (FRC 振荡器调节)	90
PADCFG1 (焊盘配置控制)	120, 136, 148
RCFGCAL (RTCC 校准和配置)	147
RCON (复位控制)	54
REFOCON (参考振荡器控制)	93
SPI1CON1 (SPI1 控制 1)	126
SPI1CON2 (SPI1 控制 2)	127
SPI1STAT (SPI1 状态和控制)	124
SR (ALU 状态)	18, 61
T1CON (Timer1 控制)	106
T2CON (Timer2 控制)	110
T3CON (Timer3 控制)	111
UxMODE (UARTx 模式)	140
UxRXREG (UARTx 接收)	144
UxSTA (UARTx 状态和控制)	142
UxTXREG (UARTx 发送)	144
WKDYHR (RTCC 星期和小时值)	151
YEAR (RTCC 年值)	150

寄存器映射	
A/D 转换器 (ADC)	30
CPU 内核	25
CRC	31
CTMU	30
I ² C	28
ICN	26
NVM	32
PMD	32
PORTA	29
PORTB	29
SPI	28
UART	28
定时器	27
焊盘配置	29
深度休眠	32
实时时钟和日历 (RTCC)	31
时钟控制	32
输出比较	27
输入捕捉	27
双比较器	31
中断控制器	26
基于指令的节能模式	95
交流特性	
CTMU 电流源	221
比较器	221
比较器参考电压稳定时间	221
负载条件和要求	217
内部 RC 精度	219
输出引脚上的容性负载要求	217
温度和电压规范	217
节能特性	95

K

开发支持	193
看门狗定时器	
深度休眠 (DSWDT)	191
看门狗定时器 (WDT)	190
窗口操作	190
勘误表	6
可编程循环冗余校验 (CRC) 发生器	157
客户通知服务	10
客户支持	10
空闲模式	95
框图	
10 位高速 A/D 转换器	164
16 位 Timer1	105
CALL 堆栈帧	33
CPU 编程模型	17
CRC 根据多项式进行了重新配置	158
CRC 移位寄存器详细信息	157
I ² C 模块	130
PIC24F CPU 内核	16
PIC24F16KA102 系列 (通用)	10
PSV 操作	37
RTCC	145
SPI1 模块 (标准缓冲模式)	122
SPI1 模块 (增强型缓冲模式)	123
Timer2 (16 位同步模式)	109
Timer2/3 (32 位模式)	108
Timer3 (16 位同步模式)	109
比较器参考电压	177
比较器模块	173
测量电容的 CTMU 连接和内部配置	179
测量时间的 CTMU 典型连接和内部配置	180
产生脉冲延时的 CTMU 典型连接和内部配置	180

访问程序空间内数据的地址生成方式	34
复位系统	53
高 / 低压检测 (HLVD)	161
各个比较器的配置	174
共用 I/O 端口结构	103
简化 UART	137
看门狗定时器 (WDT)	190
使用表指令访问程序存储器	36
输出比较	118
输入捕捉	113
系统时钟	85

M

Microchip 因特网网站	10
MPLAB ASM30 汇编器、链接器和库管理器	194
MPLAB ICD 2 在线调试器	195
MPLAB ICE 2000 高性能通用在线仿真器	195
MPLAB PM3 器件编程器	195
MPLAB REAL ICE 在线仿真器系统	195
MPLAB 集成开发环境软件	193
MPLINK 目标链接器 /MPLIB 目标库管理器	194
脉宽调制。请参见 PWM。	

N

Near 数据空间	24
内核特性	7

P

PICSTART 2 开发编程器	196
PICSTART Plus 开发编程器	196
配置位	183

Q

器件特性 (汇总)	9
欠压复位	
跳变点	208
欠压复位 (BOR)	55

R

RTCC	145
寄存器映射	146
校准	154
闹钟配置	154
闹钟时标设置 (图)	155
写锁定	146
选择时钟源	146
源时钟	145
软件堆栈	33
软件模拟器 (MPLAB SIM)	194

S

SFR 空间	24
闪存程序存储器	
RTSP 工作原理	40
编程操作	40
编程算法	42
表指令	39
控制寄存器	40
增强型 ICSP 操作	40
闪存和数据 EEPROM 编程	
控制寄存器	47
NVMADR	49
NVMCON	47
NVMKEY	47
深度休眠模式	96

时序图

CLKO 和 I/O 时序	220
外部时钟	218

时序要求

CLKO 和 I/O	220
PLL 时钟规范	219
外部时钟	218

时钟频率, 时钟切换	95
------------------	----

输出比较

PWM 模式	116
周期和占空比计算	117
产生单个输出脉冲	115
产生连续输出脉冲	115

数据存储

Near 数据空间	24
SFR 空间	24
存储器映射	23
地址空间	23
构成	24
空间宽度	23
软件堆栈	33

数据 EEPROM

编程	
单字写	51
读数据 EEPROM	52
数据 EEPROM 批量擦除	51
擦除	50
操作	49
输入捕捉	113

T

Timer1	105
Timer2/3	107

U

UART	137
8 位或 9 位数据模式下的接收	139
8 位数据模式下的发送	139
9 位数据模式下的发送	139
IrDA 支持	139
UxCTS 和 UxRTS 控制引脚的操作	139
波特率发生器 (BRG)	138
间隔和同步发送序列	139

W

WWW, 在线支持	6
WWW 地址	10

X

休眠模式	95
选择性外设功耗控制	101

Y

引脚说明	11–14
因特网地址	10

Z

在线串行编程	191
在线调试器	191
振荡器配置	
CPU 时钟机制	86
POR 时的初始配置	86
时钟切换	91
序列	91
时钟选择的配置值	86

PIC24F16KA102 系列

指令集

操作码符号	198
概述	199
汇总	197

直流特性

I/O 引脚输出规范	215
I/O 引脚输入规范	214
比较器	216
比较器参考电压	216
程序存储器	215
掉电电流 I_{PD}	211
工作电流 I_{DD}	208
空闲电流 I_{IDLE}	209
数据 EEPROM 存储器	215
温度和电压规范	207

中断

备用中断向量表 (AIVT)	57
复位过程	57
设置和服务过程	84
陷阱向量	59
向量表	58
已实现的向量	59
中断向量表 (IVT)	57

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 的客户通知服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时, 收到电子邮件通知。

欲注册, 请登录 Microchip 网站 www.microchip.com, 点击 “变更通知客户 (Customer Change Notification)” 服务后按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助:

- 代理商或代表
- 当地销售办事处
- 应用工程师 (FAE)
- 技术支持

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。

PIC24F16KA102 系列

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： PIC24F16KA102 系列

文献编号： DS39927A_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？



MICROCHIP

全球销售及服务中心

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: 678-957-9614
Fax: 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

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: 408-961-6444
Fax: 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 - 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 - 厦门
Tel: 86-592-238-8138
Fax: 86-592-238-8130

中国 - 西安
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

中国 - 珠海
Tel: 86-756-321-0040
Fax: 86-756-321-0049

台湾地区 - 高雄
Tel: 886-7-536-4818
Fax: 886-7-536-4803

台湾地区 - 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

台湾地区 - 新竹
Tel: 886-3-572-9526
Fax: 886-3-572-6459

亚太地区

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-4182-8400
Fax: 91-80-4182-8422

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

01/02/08