



---

# **PIC24FJ64GB004 系列 数据手册**

**采用 nanoWatt XLP 技术  
带 USB On-The-Go ( OTG ) 的  
28/44 引脚  
16 位闪存单片机**

---

请注意以下有关 Microchip 器件代码保护功能的要点：

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

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

---

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

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

#### 商标

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

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

Analog-for-the-Digital Age、Application Maestro、CodeGuard、dsPICDEM、dsPICDEM.net、dsPICworks、dsSPEAK、ECAN、ECONOMONITOR、FanSense、HI-TIDE、In-Circuit Serial Programming、ICSP、Mindi、MiWi、MPASM、MPLAB Certified 徽标、MPLIB、MLINK、mTouch、Octopus、Omniscient Code Generation、PICC、PICC-18、PICDEM、PICDEM.net、PICkit、PICtail、PIC<sup>32</sup> 徽标、REAL ICE、rfLAB、Select Mode、Total Endurance、TSHARC、UniWinDriver、WiperLock 和 ZENA 均为 Microchip Technology Inc. 在美国和其他国家或地区的商标。

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

在此提及的所有其他商标均为各持有公司所有。

© 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<sup>®</sup> MCU 与 dsPIC<sup>®</sup> DSC、KEELOQ<sup>®</sup> 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。



**MICROCHIP**

**PIC24FJ64GB004 系列**

## 采用 nanoWatt XLP 技术带 USB On-The-Go ( OTG ) 的 28/44 引脚 16 位闪存单片机

### 通用串行总线特性：

- 符合 USB v2.0 On-The-Go ( OTG ) 规范
- 双角色能力——可充当主机或外设
- 主机模式下的低速 ( 1.5 Mb/s ) 和全速 ( 12 Mb/s ) USB 操作
- 器件模式下的全速 USB 操作
- USB 的高精度 PLL
- 使用内部振荡器时，精度可达 0.25%——无需外部晶振
- 用于产生 USB 总线电压的内部升压辅助电路
- 用于产生 USB 总线电压的片外电荷泵的接口
- 最多支持 32 个端点 ( 16 个双向端点 ) :
  - USB 模块可将器件上的任意 RAM 地址单元用作 USB 端点缓冲区
- 片内 USB 收发器
- 片外 USB 收发器的接口
- 支持控制、中断、同步和批量传输
- 片内上拉和下拉电阻

### 高性能 CPU :

- 改进型哈佛架构
- 最高运行速度可达 16 MIPS ( @ 32 MHz )
- 8 MHz 内部振荡器 ( 典型精度为 0.25% ) :
  - 96 MHz PLL
  - 多个分频选项
- 17 位 x 17 位单周期硬件小数 / 整数乘法器
- 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 :
  - 76 条基本指令
  - 灵活的寻址模式
- 可寻址最大 12 MB 的线性程序存储器
- 可寻址最大 64 KB 的线性数据存储器
- 2 个地址发生单元，分别用于数据存储器的读 / 写寻址

### 功耗管理模式：

- 采用 nanoWatt XLP 技术的可选功耗管理模式 ( 功耗极低 ) :
  - 深度休眠模式使得系统近似完全掉电 ( 典型电流值为 25 nA , 如果 RTCC 或 WDT 运行，则为 500 nA ) , 且可由外部触发信号唤醒或者在发生可编程 WDT 或 RTCC 闹钟事件时自唤醒
  - 深度休眠模式下的极低功耗 DSBOR , 所有其他模式下的 LPBOR
  - 休眠模式关闭外设和内核以显著节省功耗，在该模式下可快速唤醒
  - 空闲模式关闭 CPU 和外设以节省大量功耗，典型电流消耗降至 4.5 μA
  - 打盹模式使得 CPU 时钟比外设时钟运行缓慢
  - 备用时钟模式允许动态切换到较低的时钟速度，以便有选择地降低功耗，运行模式时消耗的典型电流低至 15 μA

### 单片机特性：

- 工作电压范围为 2.0V 至 3.6V
- 软件控制下可自行再编程
- 可承受 5.5V 输入电压 ( 仅数字引脚 )
- 所有 I/O 引脚上的高拉 / 灌电流 ( 18 mA/18 mA )
- 闪存程序存储器 :
  - 至少可耐受 10,000 次擦写
  - 数据保存时间最短 20 年
  - 可选的写保护边界
- 故障保护时钟监视器操作 :
  - 检测时钟故障并切换至片上 FRC 振荡器
- 片上 2.5V 稳压器
- 上电复位 ( Power-on Reset , POR ) 、上电延时定时器 ( Power-up Timer , PWRT ) 和振荡器起振定时器 ( Oscillator Start-up Timer , OST )
- 两个灵活的看门狗定时器 ( Watchdog Timer , WDT ) , 以确保可靠地工作 :
  - 用于正常工作的标准可编程 WDT
  - 用于深度休眠模式的极低功耗 WDT , 可编程周期为 2 ms 至 26 天
- 通过 2 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 )
- 支持 JTAG 边界扫描

PIC24FJ 器件	引脚	程序存储器 ( 字节数 )	SRAM ( 字节数 )	可重映射的外设						I <sup>2</sup> C™	10 位 A/D ( 道道数 )	比较器	PMP/PSP	RTCC	CTMU	USB OTG
				可重映射 引脚数	16 位 定时器	捕捉 输入	比较 / PWM 输出	带 IrDA® 的 UART	SPI							
32GB002	28	32K	8K	15	5	5	5	2	2	2	9	3	有	有	有	有
64GB002	28	64K	8K	15	5	5	5	2	2	2	9	3	有	有	有	有
32GB004	44	32K	8K	25	5	5	5	2	2	2	13	3	有	有	有	有
64GB004	44	64K	8K	25	5	5	5	2	2	2	13	3	有	有	有	有

# PIC24FJ64GB004 系列

## 模拟特性：

- 最多 13 路通道的 10 位模数 (Analog-to-Digital , A/D) 转换器：
  - 500 kspS 的转换速率
  - 可在休眠和空闲模式下进行转换
- 三个带可编程输入 / 输出配置的模拟比较器
- 充电时间测量单元 (Charge Time Measurement Unit , CTMU)：
  - 支持触摸屏和电容开关的容性触摸传感
  - 提供高分辨率的时间测量和简单的温度传感

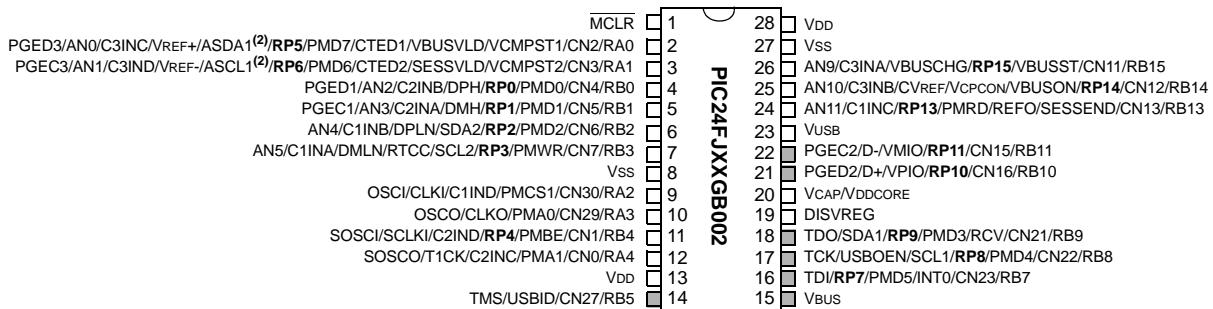
## 外设特性：

- 外设引脚选择：
  - 允许对许多外设进行独立的 I/O 映射
  - 最多 25 个可用引脚 (44 引脚器件)
  - 连续的硬件完整性检查和安全互锁以防止无意中更改配置
- 8 位并行主 / 从端口 (Parallel Master/Slave Port , PMP/PSP)：
  - 44 引脚器件上具有最多 11 个专用的地址引脚，可实现最多 16 位的多路寻址
  - 控制线上的可编程优先级
  - 支持传统的并行从端口

- 硬件实时时钟 / 日历 (Real-Time Clock/Calendar , RTCC)：
  - 提供时钟、日历和闹钟功能
  - 在深度休眠模式下仍能工作
- 2 个 3 线 /4 线 SPI 模块 (支持 4 帧模式)，带 8 级深 FIFO 缓冲区
- 2 个 I<sup>2</sup>C<sup>TM</sup> 模块，支持多主器件 / 从模式和 7 位 /10 位寻址
- 2 个 UART 模块：
  - 支持 RS-485、RS-232 和 LIN/J2602
  - 片上 IrDA<sup>®</sup> 硬件编码器 / 解码器
  - 遇到起始位自动唤醒
  - 自动波特率检测 (Auto-Baud Detect , ABD)
  - 4 级深 FIFO 缓冲区
- 5 个带可编程预分频器的 16 位定时器 / 计数器
- 5 路 16 位捕捉输入，每路都具有专用时基
- 5 路 16 位比较 /PWM 输出，每路都具有专用时基
- 可编程的 32 位循环冗余校验 (Cyclic Redundancy Check , CRC) 发生器
- 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

## 引脚框图

### 28 引脚 SPDIP 和 SOIC<sup>(1)</sup>



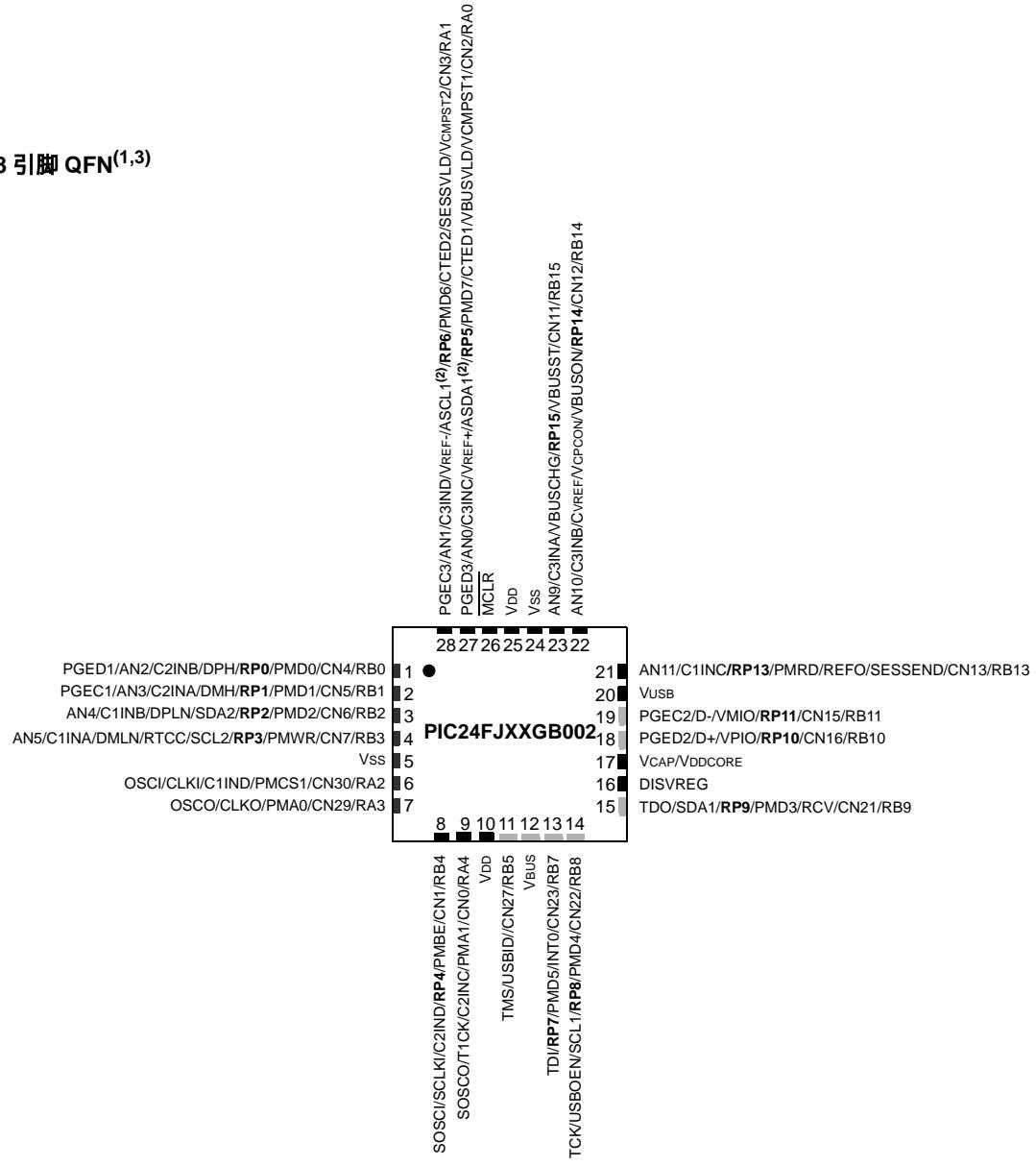
图注： RPn 表示可重映射的外设引脚。

注 1： 灰色阴影表示可承受 5.5V 电压的输入引脚。

2： 当 I<sup>2</sup>C1SEL 位置 1 时，可与 SDA1 和 SCL1 复用。

## 引脚框图

28 引脚 QFN<sup>(1,3)</sup>

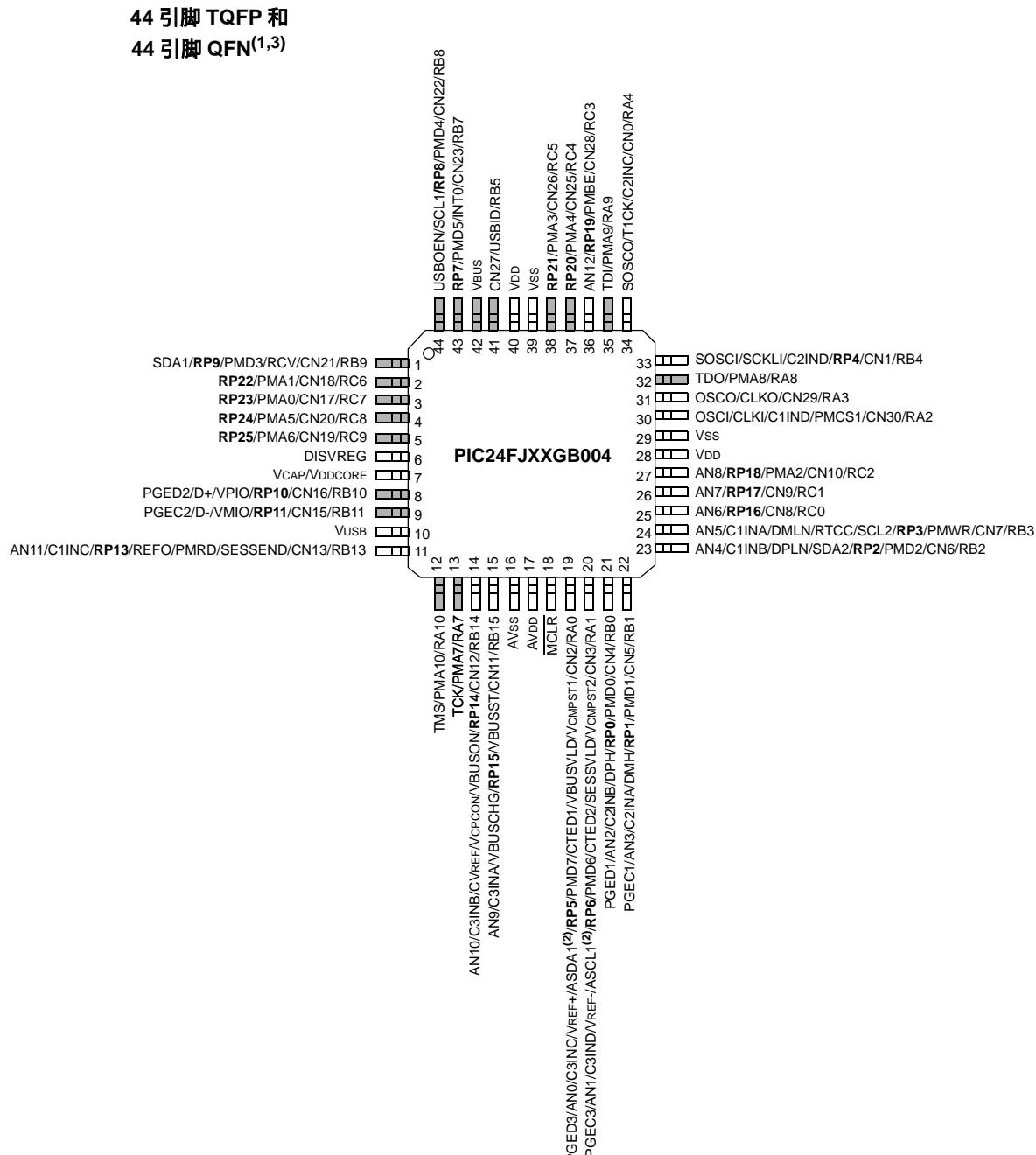


图注： RPn 表示可重映射的外设引脚。

- 注 1：灰色阴影表示可承受 5.5V 电压的输入引脚。
- 2：当 I2C1SEL 位置 1 时，可与 SDA1 和 SCL1 复用。
- 3：QFN 器件背面的焊盘应连接到 Vss。

# PIC24FJ64GB004 系列

## 引脚框图



图注： RP<sub>n</sub> 表示可重映射的外设引脚。

- 注 1： 灰色阴影表示可承受 5.5V 电压的输入引脚。  
2： 当 I2C1SEL 位置 1 时，可与 SDA1 和 SCL1 复用。  
3： QFN 器件背面的焊盘应连接到 VSS。

## 目录

1.0 器件概述	7
2.0 16 位单片机入门指南	17
3.0 CPU	21
4.0 存储器构成	27
5.0 闪存程序存储器	51
6.0 复位	59
7.0 中断控制器	65
8.0 振荡器配置	103
9.0 节能特性	113
10.0 I/O 端口	123
11.0 Timer1	145
12.0 Timer2/3 和 Timer4/5	147
13.0 带专用定时器的输入捕捉	153
14.0 带专用定时器的输出比较	157
15.0 串行外设接口 (SPI)	167
16.0 I <sup>2</sup> C <sup>TM</sup>	177
17.0 通用异步收发器 (UART)	185
18.0 带 On-The-Go 支持的通用串行总线 (USB OTG)	193
19.0 并行主端口 (PMP)	227
20.0 实时时钟和日历 (RTCC)	237
21.0 32 位可编程循环冗余校验 (CRC) 发生器	249
22.0 10 位高速 A/D 转换器	255
23.0 三比较器模块	265
24.0 比较器参考电压	269
25.0 充电时间测量单元 (CTMU)	271
26.0 特殊功能	275
27.0 开发支持	289
28.0 指令集汇总	293
29.0 电气特性	301
30.0 封装信息	319
附录 A : 版本历史	329
Microchip 网站	337
变更通知客户服务	337
客户支持	337
读者反馈表	338
产品标识体系	339

# PIC24FJ64GB004 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](http://www.microchip.com) 上注册。

## 1.0 器件概述

此文档包含针对以下器件的具体信息：

- PIC24FJ32GB002      • PIC24FJ32GB004
- PIC24FJ64GB002      • PIC24FJ64GB004

此系列是对 Microchip 现有的 16 位单片机产品线的扩展，结合了扩展的外设功能集和增强的计算性能，并提供了新的连接选项：USB On-The-Go (OTG)。PIC24FJ64GB004 系列为高性能 USB 应用提供了新的平台，这些应用可能需要高于 8 位的平台，但却无需使用数字信号处理器功能。

## 1.1 内核特性

### 1.1.1 16 位架构

所有 PIC24F 器件都采用了 16 位改进型哈佛架构，该架构最初是在 Microchip 的 dsPIC® 数字信号控制器中引入的。PIC24F CPU 内核提供了大量增强功能，例如：

- 16 位数据路径和 24 位地址路径，可在数据空间和其他存储空间之间传送信息
- 线性寻址能力最高可达 12 MB (程序空间) 和 64 KB (数据空间)
- 利用内建软件堆栈支持 16 元工作寄存器阵列
- 支持整数算术运算的 17 位 x17 位硬件乘法器
- 支持 32 位 /16 位除法运算的硬件
- 支持多种寻址模式并为高级语言（如 C 语言）而优化的指令集
- 工作性能最高可达 16 MIPS

### 1.1.2 节能技术

PIC24FJ64GB004 系列的所有器件具有许多能在工作时显著降低功耗的功能。主要包括以下几项：

- **动态时钟切换**：在器件工作过程中，器件时钟可在软件控制下切换为 Timer1 时钟源或内部低功耗 RC 振荡器，允许用户把节能理念融入到软件设计中去。
- **打盹模式操作**：当那些对时序要求很高的应用（如串行通信）要求外设不间断地工作时，该模式可以适当降低 CPU 时钟速度，从而可在不丢失任一脉冲的前提下进一步节约功耗。

- **基于指令的节能模式**：提供了三种基于指令的节能模式：

- 空闲模式——内核关闭而外设保持活动状态。
- 休眠模式——内核以及需要系统时钟的外设关闭，而使用自己的时钟或来自其他器件的时钟的外设保持活动状态。
- 深度休眠模式——内核、外设（RTCC 和 DSWDT 除外）、闪存和 SRAM 关闭，以最大程度地节省电流来延长便携性应用的电池使用寿命。

### 1.1.3 振荡器选项和特性

PIC24FJ64GB004 系列中的所有器件提供 5 个不同的振荡器选项，使用户在开发应用硬件时有很大的选择范围。这些选项包括：

- 使用晶振或陶瓷谐振器的 2 种晶振模式。
- 提供 2 分频时钟输出选项的 2 种外部时钟模式。
- 标称输出值为 8 MHz 的快速内部 RC 振荡器（Fast Internal Oscillator，FRC），可以在软件控制下分频，从而提供低至 31 kHz 的时钟速度。
- 锁相环（Phase Lock Loop，PLL）倍频器，可在外部振荡器模式和 FRC 振荡器下使用，可使时钟速度最高达 32 MHz。
- 具有固定的 31 kHz 输出的独立低功耗内部 RC 振荡器（LPRC），可为对时序要求很高的应用提供低功耗选项。

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

### 1.1.4 简便移植

无论存储器容量如何，所有器件均共享同一组外设，从而可在应用扩展和改进时，方便地进行移植。整个系列使用相同的引脚配置方案也有助于从一个器件向下一代更大型器件的移植。

PIC24F 系列器件的引脚同 dsPIC33 系列器件的引脚是兼容的，并与 PIC18 和 dsPIC30 器件的引脚配置方案部分兼容。这样仍然可以使用 Microchip 器件而将相对简单的应用顺利移植为强大而复杂的应用。

## 1.2 USB On-The-Go

PIC24FJ64GB004 系列器件将单芯片上的 USB On-The-Go 功能引入到低引脚数 Microchip 器件中。此模块提供了与符合 USB 2.0 标准的目标器件相同的片上功能，以及与 USB 嵌入式主机相同的受限的独立功能。通过实现 USB 主机协商协议（Host Negotiation Protocol , HNP），模块还可以在器件操作和主机操作之间动态切换，从而可以在单片机平台上实现更多使能了 USB 的应用。

除了 USB 主机功能之外，PIC24FJ64GB004 系列器件还提供了一个真正的单芯片 USB 解决方案，包括一个片内收发器以及一个在主机操作期间为总线供电的升压发生器。

## 1.3 其他特性

- 外设引脚选择：**外设引脚选择功能允许大部分的数字外设通过一组固定的数字 I/O 引脚映射。用户可将许多数字外设之一的输入和 / 或输出独立地映射到其中的任一 I/O 引脚。
- 通信：**PIC24FJ64GB004 系列器件集成了许多串行通信外设以处理各种应用需求。提供了两个同时支持主从工作模式的独立 I<sup>2</sup>C<sup>TM</sup> 模块。通过外设引脚选择（Peripheral Pin Select , PPS）功能，器件还具有两个包含内置 IrDA<sup>®</sup> 编码器 / 解码器的独立 UART 以及两个 SPI 模块。
- 模拟特性：**PIC24FJ64GB004 系列的所有器件都包含一个 10 位 A/D 转换器模块和一个三比较器模块。A/D 模块具有可编程的采集时间，从而不必在选择通道和启动转换之间等待一个采样周期，并加快了采样速度。比较器模块包含三个模拟比较器，它们可针对多种操作进行配置。
- CTMU 接口：**此模块为精确时间测量和脉冲生成提供了一个便利的方法，并可用作容性传感器的接口。

- 并行主 / 增强型并行从端口：**可以将一个通用 I/O 端口重新配置为用于增强型并行数据通信。在这种模式下，可以将端口配置为工作在主或从模式下。在主模式下支持 8 位和 16 位数据传输并具有最多 12 条外部地址线。

- 实时时钟 / 日历：**此模块通过硬件实现带有闹钟功能的全功能时钟和日历，从而释放了定时器资源和程序存储空间供内核应用使用。

## 1.4 系列中各产品的具体信息

PIC24FJ64GB004 系列中的器件具有 28 引脚和 44 引脚两类封装形式，图 1-1 给出了所有器件的一般框图。

这些器件在以下几个方面存在差异：

- 闪存程序存储器：**
  - PIC24FJ32GB0 器件——32 KB
  - PIC24FJ64GB0 器件——64 KB
- 可用的 I/O 引脚和端口数：**
  - 28 引脚器件——2 个端口 19 个引脚
  - 44 引脚器件——3 个端口 33 个引脚
- 可用的电平变化中断通知（Interrupt-on-Change Notification , ICN）输入：**
  - 28 引脚器件——19
  - 44 引脚器件——29
- 可用的可重映射引脚数：**
  - 28 引脚器件——15 个引脚
  - 44 引脚器件——25 个引脚
- 可用的 PMP 地址引脚数：**
  - 28 引脚器件——3 个引脚
  - 44 引脚器件——12 个引脚
- 可用的 A/D 输入通道数：**
  - 28 引脚器件——9 个引脚
  - 44 引脚器件——12 个引脚

该系列器件的其他特性都是相同的。表 1-1 对此进行了总结。

表 1-2 给出了 PIC24FJ64GB004 系列器件上可用的引脚功能列表，按功能排序。注意此表只显示了各个外设功能所使用的引脚单元，而没有显示同一引脚上的多种功能的复用方式。在本数据手册开始部分的引脚框图中提供了有关引脚复用的信息。复用的功能按功能的优先级排序，最前面的是优先级最高的外设功能。

# PIC24FJ64GB004 系列

表 1-1 : PIC24FJ64GB004 系列器件的特性

特性	PIC24FJ32GB002	PIC24FJ64GB002	PIC24FJ32GB004	PIC24FJ64GB004
工作频率	DC—32 MHz			
程序存储器 (字节数)	32K	64K	32K	64K
程序存储器 (指令数)	11,008	22,016	11,008	22,016
数据存储器 (字节数)	8,192			
中断源 (软向量 / NMI 陷阱)	45 (41/4)			
I/O 端口	端口 A 和 B		端口 A、B 和 C	
I/O 引脚总数	19		33	
可重映射的引脚数	15		25	
定时器 :				
总数 (16 位)	5 <sup>(1)</sup>			
32 位 (由一对 16 位定时器组成)	2			
输入捕捉通道数	5 <sup>(1)</sup>			
输出比较 /PWM 通道数	5 <sup>(1)</sup>			
输入电平变化通知中断	19		29	
串行通信 :				
UART	2 <sup>(1)</sup>			
SPI (3 线 /4 线)	2 <sup>(1)</sup>			
I <sup>2</sup> C <sup>TM</sup>	2			
并行通信 (PMP/PSP)	有			
JTAG 边界扫描	有			
10 位模数转换器模块 (输入通道)	9		13	
模拟比较器	3			
CTMU 接口	有			
复位 (和延时)	POR、BOR、RESET 指令、MCLR、WDT；非法操作码、REPEAT 指令、硬件陷阱和配置字失配 (PWRT、OST 和 PLL 锁定)			
指令集	76 条基本指令和多种寻址模式			
封装	28 引脚 QFN、SOIC 和 SPDIP		44 引脚 QFN 和 TQFP	

注 1：外设可通过可重映射的引脚访问。

# PIC24FJ64GB004 系列

图 1-1 : PIC24FJ64GB004 系列的一般框图

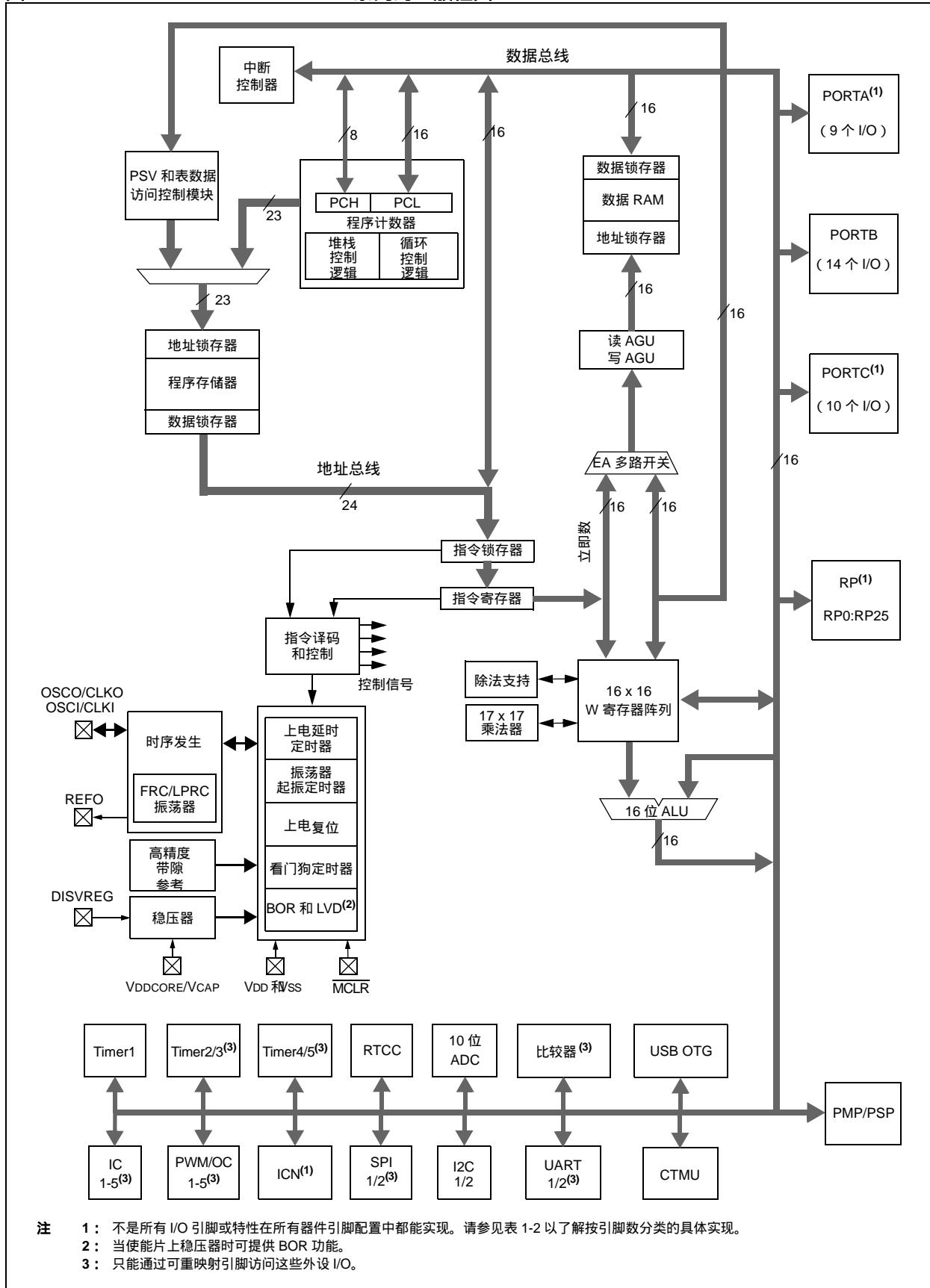


表 1-2 : PIC24FJ64GB004 系列引脚配置说明

功能	引脚数			I/O	输入缓冲器	说明
	28 引脚 SPDIP/ SOIC	28 引脚 QFN	44 引脚 QFN/TQFP			
AN0	2	27	19	I	ANA	A/D 模拟输入。
AN1	3	28	20	I	ANA	
AN2	4	1	21	I	ANA	
AN3	5	2	22	I	ANA	
AN4	6	3	23	I	ANA	
AN5	7	4	24	I	ANA	
AN6	—	—	25	I	ANA	
AN7	—	—	26	I	ANA	
AN8	—	—	27	I	ANA	
AN9	26	23	15	I	ANA	
AN10	25	22	14	I	ANA	
AN11	24	21	11	I	ANA	
AN12	—	—	36	I	ANA	
ASCL1	3	28	20	I/O	I <sup>2</sup> C	备用的 I <sup>2</sup> C1 同步串行时钟输入 / 输出。
ASDA1	2	27	19	I/O	I <sup>2</sup> C	备用的 I <sup>2</sup> C1 同步串行数据输入 / 输出。
AVDD	—	—	17	P	—	模拟模块的正电源。
AVSS	—	—	16	P	—	模拟模块的参考地。
C1INA	7	4	24	I	ANA	比较器 1 输入 A。
C1INB	6	3	23	I	ANA	比较器 1 输入 B。
C1INC	24	21	11	I	ANA	比较器 1 输入 C。
C1IND	9	6	30	I	ANA	比较器 1 输入 D。
C2INA	5	2	22	I	ANA	比较器 2 输入 A。
C2INB	4	1	21	I	ANA	比较器 2 输入 B。
C2INC	12	9	34	I	ANA	比较器 2 输入 C。
C2IND	11	8	33	I	ANA	比较器 2 输入 D。
C3INA	26	23	15	I	ANA	比较器 3 输入 A。
C3INB	25	22	14	I	ANA	比较器 3 输入 B。
C3INC	2	27	19	I	ANA	比较器 3 输入 C。
C3IND	3	28	20	I	ANA	比较器 3 输入 D。
CLKI	9	6	30	I	ANA	主时钟输入连接。
CLKO	10	7	31	O	—	系统时钟输出。

图注 : TTL = TTL 输入缓冲器  
 ANA = 模拟电平输入 / 输出

ST = 施密特触发器输入缓冲器  
 I<sup>2</sup>C<sup>TM</sup> = I<sup>2</sup>C/SMBus 输入缓冲器

# PIC24FJ64GB004 系列

表 1-2 : PIC24FJ64GB004 系列引脚配置说明 (续)

功能	引脚数			I/O	输入 缓冲器	说明
	28 引脚 SPDIP/ SOIC	28 引脚 QFN	44 引脚 QFN/TQFP			
CN0	12	9	34	I	ST	电平变化中断输入。
CN1	11	8	33	I	ST	
CN2	2	27	19	I	ST	
CN3	3	28	20	I	ST	
CN4	4	1	21	I	ST	
CN5	5	2	22	I	ST	
CN6	6	3	23	I	ST	
CN7	7	4	24	I	ST	
CN8	—	—	25	I	ST	
CN9	—	—	26	I	ST	
CN10	—	—	27	I	ST	
CN11	26	23	15	I	ST	
CN12	25	22	14	I	ST	
CN13	24	21	11	I	ST	
CN15	22	19	9	I	ST	
CN16	21	18	8	I	ST	
CN17	—	—	3	I	ST	
CN18	—	—	2	I	ST	
CN19	—	—	5	I	ST	
CN20	—	—	4	I	ST	
CN21	18	15	1	I	ST	
CN22	17	14	44	I	ST	
CN23	16	13	43	I	ST	
CN25	—	—	37	I	ST	
CN26	—	—	38	I	ST	
CN27	14	11	41	I	ST	
CN28	—	—	36	I	ST	
CN29	10	7	31	I	ST	
CN30	9	6	30	I	ST	
CTED1	2	27	19	I	ANA	CTMU 外部边沿输入 1。
CTED2	3	28	20	I	ANA	CTMU 外部边沿输入 2。
CVREF	25	22	14	O	—	比较器参考电压输出。
D+	21	18	8	I/O	—	USB 差分正信号线 (内部收发器)。
D-	22	19	9	I/O	—	USB 差分负信号线 (内部收发器)。
DMH	5	2	22	O	—	D- 外部上拉控制输出。
DMLN	7	4	24	O	—	D- 外部下拉控制输出。
DPH	4	1	21	O	—	D+ 外部上拉控制输出。
DPLN	6	3	23	O	—	D+ 外部下拉控制输出。
DISVREG	19	16	6	I	ST	禁止稳压器。

图注 : TTL = TTL 输入缓冲器  
ANA = 模拟电平输入 / 输出

ST = 施密特触发器输入缓冲器  
 $I^2C^{\text{TM}}$  = I<sup>2</sup>C/SMBus 输入缓冲器

# PIC24FJ64GB004 系列

表 1-2 : PIC24FJ64GB004 系列引脚配置说明 (续)

功能	引脚数			I/O	输入 缓冲器	说明
	28 引脚 SPDIP/ SOIC	28 引脚 QFN	44 引脚 QFN/TQFP			
INT0	16	13	43	I	ST	外部中断输入。
MCLR	1	26	18	I	ST	主复位 (器件复位) 输入。将此引脚拉为低电平可导致复位。
OSCI	9	6	30	I	ANA	主振荡器输入连接。
OSCO	10	7	31	O	ANA	主振荡器输出连接。
PGEC1	5	2	22	I/O	ST	在线调试器 / 仿真器 /ICSP™ 编程时钟。
PGED1	4	1	21	I/O	ST	在线调试器 / 仿真器 /ICSP 编程数据。
PGEC2	22	19	9	I/O	ST	在线调试器 / 仿真器 /ICSP 编程时钟。
PGED2	21	18	8	I/O	ST	在线调试器 / 仿真器 /ICSP 编程数据。
PGEC3	3	28	20	I/O	ST	在线调试器 / 仿真器 /ICSP 编程时钟。
PGED3	2	27	19	I/O	ST	在线调试器 / 仿真器 /ICSP 编程数据。
PMA0	10	7	3	I/O	ST	并行主端口地址 bit 0 输入 (缓冲从模式) 和输出 (主模式)。
PMA1	12	9	2	I/O	ST	并行主端口地址 bit 1 输入 (缓冲从模式) 和输出 (主模式)。
PMA2	—	—	27	O	—	并行主端口地址 (解复用主模式)。
PMA3	—	—	38	O	—	
PMA4	—	—	37	O	—	
PMA5	—	—	4	O	—	
PMA6	—	—	5	O	—	
PMA7	—	—	13	O	—	
PMA8	—	—	32	O	—	
PMA9	—	—	35	O	—	
PMA10	—	—	12	O	—	
PMCS1	9	6	30	I/O	ST/TTL	并行主端口片选 1 选通 / 地址 bit 15。
PMBE	11	8	36	O	—	并行主端口字节使能选通。
PMD0	4	1	21	I/O	ST/TTL	并行主端口数据 (解复用主模式) 或地址 / 数据 (复用主模式)。
PMD1	5	2	22	I/O	ST/TTL	
PMD2	6	3	23	I/O	ST/TTL	
PMD3	18	15	1	I/O	ST/TTL	
PMD4	17	14	44	I/O	ST/TTL	
PMD5	16	13	43	I/O	ST/TTL	
PMD6	3	28	20	I/O	ST/TTL	
PMD7	2	27	19	I/O	ST/TTL	并行主端口读选通。
PMRD	24	21	11	O	—	并行主端口写选通。
PMWR	7	4	24	O	—	并行主端口写选通。

图注 : TTL = TTL 输入缓冲器  
ANA = 模拟电平输入 / 输出

ST = 施密特触发器输入缓冲器  
 $I^2C$ ™ =  $I^2C$ /SMBus 输入缓冲器

# PIC24FJ64GB004 系列

表 1-2 : PIC24FJ64GB004 系列引脚配置说明 (续)

功能	引脚数			I/O	输入 缓冲器	说明
	28 引脚 SPDIP/ SOIC	28 引脚 QFN	44 引脚 QFN/TQFP			
RA0	2	27	19	I/O	ST	PORTA 数字 I/O。
RA1	3	28	20	I/O	ST	
RA2	9	6	30	I/O	ST	
RA3	10	7	31	I/O	ST	
RA4	12	9	34	I/O	ST	
RA7	—	—	13	I/O	ST	
RA8	—	—	32	I/O	ST	
RA9	—	—	35	I/O	ST	
RA10	—	—	12	I/O	ST	
RB0	4	1	21	I/O	ST	
RB1	5	2	22	I/O	ST	PORTB 数字 I/O。
RB2	6	3	23	I/O	ST	
RB3	7	4	24	I/O	ST	
RB4	11	8	33	I/O	ST	
RB5	14	11	41	I/O	ST	
RB7	16	13	43	I/O	ST	
RB8	17	14	44	I/O	ST	
RB9	18	15	1	I/O	ST	
RB10	21	18	8	I/O	ST	
RB11	22	19	9	I/O	ST	
RB13	24	21	11	I/O	ST	PORTC 数字 I/O。
RB14	25	22	14	I/O	ST	
RB15	26	23	15	I/O	ST	
RC0	—	—	25	I/O	ST	
RC1	—	—	26	I/O	ST	
RC2	—	—	27	I/O	ST	
RC3	—	—	36	I/O	ST	
RC4	—	—	37	I/O	ST	
RC5	—	—	38	I/O	ST	
RC6	—	—	2	I/O	ST	
RC7	—	—	3	I/O	ST	
RC8	—	—	4	I/O	ST	
RC9	—	—	5	I/O	ST	
RCV	18	15	1	I	ST	USB 接收输入 (来自外部收发器)。
REFO	24	21	11	O	—	参考时钟输出。

图注 : TTL = TTL 输入缓冲器  
ANA = 模拟电平输入 / 输出

ST = 施密特触发器输入缓冲器  
 $I^2C^{\text{TM}}$  =  $I^2C$ /SMBus 输入缓冲器

表 1-2 : PIC24FJ64GB004 系列引脚配置说明 (续)

功能	引脚数			I/O	输入 缓冲器	说明
	28 引脚 SPDIP/ SOIC	28 引脚 QFN	44 引脚 QFN/TQFP			
RP0	4	1	21	I/O	ST	可重映射外设 (输入或输出)。
RP1	5	2	22	I/O	ST	
RP2	6	3	23	I/O	ST	
RP3	7	4	24	I/O	ST	
RP4	11	8	33	I/O	ST	
RP5	2	27	19	I/O	ST	
RP6	3	28	20	I/O	ST	
RP7	16	13	43	I/O	ST	
RP8	17	14	44	I/O	ST	
RP9	18	15	1	I/O	ST	
RP10	21	18	8	I/O	ST	
RP11	22	19	9	I/O	ST	
RP13	24	21	11	I/O	ST	
RP14	25	22	14	I/O	ST	
RP15	26	23	15	I/O	ST	
RP16	—	—	25	I/O	ST	
RP17	—	—	26	I/O	ST	
RP18	—	—	27	I/O	ST	
RP19	—	—	36	I/O	ST	
RP20	—	—	37	I/O	ST	
RP21	—	—	38	I/O	ST	
RP22	—	—	2	I/O	ST	
RP23	—	—	3	I/O	ST	
RP24	—	—	4	I/O	ST	
RP25	—	—	5	I/O	ST	
RTCC	7	4	24	O	—	实时时钟闹钟 / 秒脉冲输出。
SESEND	24	21	11	I	ST	USB VBUS 会话结束状态输入。
SESSVLD	3	28	20	I	ST	USB VBUS 会话有效状态输入。
SCL1	17	14	44	I/O	I <sup>2</sup> C	I <sup>2</sup> C1 同步串行时钟输入 / 输出。
SCL2	7	4	24	I/O	I <sup>2</sup> C	I <sup>2</sup> C2 同步串行时钟输入 / 输出。
SDA1	18	15	1	I/O	I <sup>2</sup> C	I <sup>2</sup> C1 数据输入 / 输出。
SDA2	6	3	23	I/O	I <sup>2</sup> C	I <sup>2</sup> C2 数据输入 / 输出。
SOSCI	11	8	33	I	ANA	辅助振荡器 / Timer1 时钟输入。
SOSCO	12	9	34	O	ANA	辅助振荡器 / Timer1 时钟输出。
T1CK	12	9	34	I	ST	Timer1 时钟输入。
TCK	17	14	13	I	ST	JTAG 测试时钟 / 编程时钟输入。
TDI	16	13	35	I	ST	JTAG 测试数据 / 编程数据输入。
TDO	18	15	32	O	—	JTAG 测试数据输出。
TMS	14	11	12	I	ST	JTAG 测试模式选择输入。
USBID	14	11	41	I	ST	USB OTG ID (仅 OTG 模式)。
USBOEN	17	14	44	O	—	USB 输出使能控制 (用于外部收发器)。

图注 : TTL = TTL 输入缓冲器  
 ANA = 模拟电平输入 / 输出

ST = 施密特触发器输入缓冲器  
 I<sup>2</sup>C<sup>TM</sup> = I<sup>2</sup>C/SMBus 输入缓冲器

# PIC24FJ64GB004 系列

表 1-2 : PIC24FJ64GB004 系列引脚配置说明 (续)

功能	引脚数			I/O	输入缓冲器	说明
	28 引脚 SPDIP/ SOIC	28 引脚 QFN	44 引脚 QFN/TQFP			
VBUS	15	12	42	P	—	USB 电压, 主机模式 (5V)。
VBUSCHG	26	23	15	O	—	USB 外部 VBUS 控制输出。
VBUSON	25	22	14	O	—	USB OTG 外部电荷泵控制。
VBUSST	26	23	15	I	ANA	USB OTG 内部电荷泵反馈控制。
VBUSVLD	2	27	19	I	ST	USB VBUS 有效状态输入。
VCAP	20	17	7	P	—	外部滤波电容连接 (使能稳压器)。
VCPCON	25	22	14	O	—	USB OTG VBUS PWM/ 电荷输出。
VDD	13, 28	10, 25	28, 40	P	—	外设数字逻辑和 I/O 引脚的正电源。
VDDCORE	20	17	7	P	—	单片机内核逻辑的正电源 (禁止稳压器)。
VMIO	22	19	9	I/O	ST	USB 差分负输入 / 输出 (外部收发器)。
VPIO	21	18	8	I/O	ST	USB 差分正输入 / 输出 (外部收发器)。
VREF-	3	28	20	I	ANA	A/D 和比较器参考电压 (低电压) 输入。
VREF+	2	27	19	I	ANA	A/D 和比较器参考电压 (高电压) 输入。
VSS	8, 27	5, 24	29, 39	P	—	逻辑和 I/O 引脚的参考地。
VUSB	23	20	10	P	—	USB 电压 (3.3V)。

图注 : TTL = TTL 输入缓冲器

ANA = 模拟电平输入 / 输出

ST = 施密特触发器输入缓冲器

$I^2C^{\text{TM}}$  =  $I^2C/\text{SMBus}$  输入缓冲器

## 2.0 16 位单片机入门指南

### 2.1 基本连接要求

在着手使用 16 位单片机 PIC24FJ64GB004 系列进行开发之前必须至少注意一些器件引脚的连接。

以下引脚必须始终连接：

- 所有 VDD 和 Vss 引脚（见第 2.2 节“电源引脚”）
- 所有 AVDD 和 AVss 引脚（不管是否使用模拟器件功能）（见第 2.2 节“电源引脚”）
- MCLR 引脚（见第 2.3 节“主复位（MCLR）引脚”）
- ENVREG/DISVREG 和 VCAP/VDDCORE 引脚（仅 PIC24FJ 器件）（见第 2.4 节“稳压器引脚（ENVREG/DISVREG 和 VCAP/VDDCORE）”）

如果在最终应用中使用了以下引脚，那么也必须连接这些引脚：

- 用于在线串行编程（ICSP™）和调试的 PGECx/PGEDx 引脚（见第 2.5 节“ICSP 引脚”）
- 使用外部振荡器源时用到的 OSC1 和 OSCO 引脚（见第 2.6 节“外部振荡器引脚”）

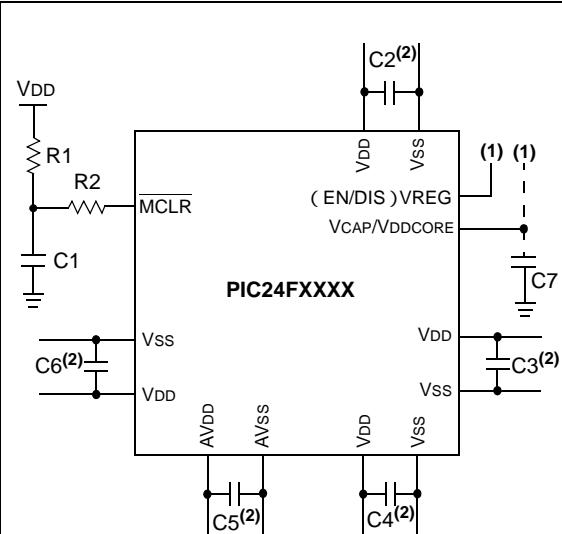
此外，还可能需要以下引脚：

- 模拟模块使用外部参考电压时需要的 VREF+/VREF- 引脚

**Note:** 不管是否使用了任何模拟模块，AVDD 和 AVss 引脚必须始终连接。

图 2-1 给出了最起码的连接要求。

图 2-1：建议的最少连接



要点（所有值都是建议值）：

C1 至 C6 : 0.1  $\mu$ F、20V 的陶瓷电容

C7 : 10  $\mu$ F、16V 的钽电容或陶瓷电容

R1 : 10 k $\Omega$

R2 : 100 $\Omega$  至 470 $\Omega$

**注 1：**请参见第 2.4 节“稳压器引脚（ENVREG/DISVREG 和 VCAP/VDDCORE）”中关于 ENVREG/DISVREG 引脚连接的说明。

**2：**在此给出的示例针对的是具有 5 个 VDD/Vss 和 AVDD/AVss 引脚对的 PIC24F 器件。其他器件具有的引脚对数可能稍有不同；应相应的调整去耦电容数。

## 2.2 电源引脚

### 2.2.1 去耦电容

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

使用去耦电容时需考虑以下条件：

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

### 2.2.2 槽路电容

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

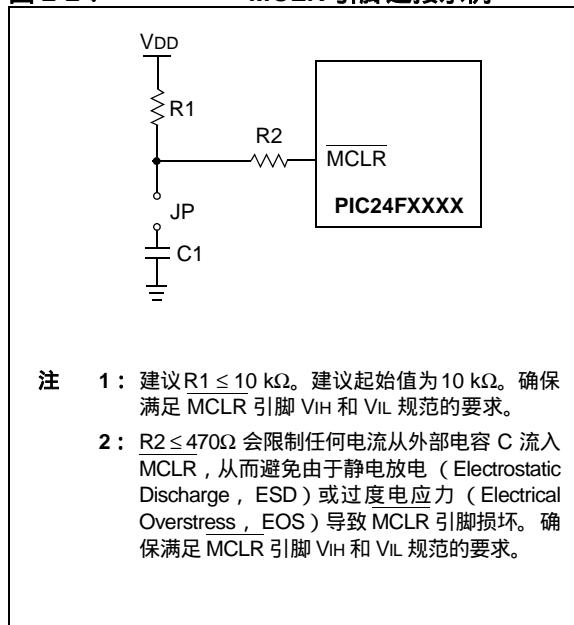
## 2.3 主复位 (MCLR) 引脚

MCLR 引脚提供了两个特殊的器件功能：器件复位以及器件编程和调试功能。如果最终应用中不需要编程和调试功能，只需要将该引脚直接连接到 VDD。为帮助增加应用的电阻以避免因电压骤降而意外复位，可能需要添加其他元件。图 2-1 给出了典型的配置。可根据应用的要求实现其他电路设计。

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

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

图 2-2：MCLR 引脚连接示例



## 2.4 稳压器引脚 (ENVREG/DISVREG 和 VCAP/VDDCORE)

**Note:** 本节仅适用于具有片内稳压器的 PIC24FJ 器件。

片内稳压器使能 / 禁止引脚 (ENVREG 或 DISVREG, 具体取决于器件系列) 必须始终直接连接电源电压引脚或接地引脚。具体的连接取决于是否使用稳压器：

- 对于 ENVREG, 将其连接到 VDD 可使能稳压器, 接地可禁止稳压器
- 对于 DISVREG, 将其接地可使能稳压器, 连接到 VDD 可禁止稳压器

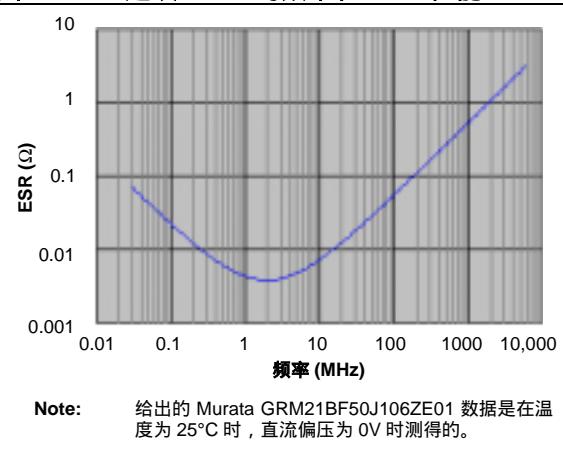
关于连接和使用片内稳压器的详细信息, 请参见第 26.2 节“片内稳压器”。

使能了稳压器时, 需要在 VCAP/VDDCORE 引脚上连接一个低 ESR (<5Ω) 的电容以稳定稳压器输出电压。VCAP/VDDCORE 引脚不得与 VDD 相连, 而是必须使用一个 10 μF 的电容接地。电容类型可以是陶瓷电容或钽电容。Murata GRM21BF50J106ZE01 (10 μF, 6.3 V) 或同等规格的电容就很合适。设计人员可使用图 2-3 来评估候选器件的 ESR。

此电容应靠近 VCAP/VDDCORE 放置。建议走线长度不要超过 0.25 英寸 (6 mm)。更多信息, 请参见第 29.0 节“电气特性”。

禁止稳压器时, VCAP/VDDCORE 引脚必须与电压为 VDDCORE 的电源相连。关于 VDD 和 VDDCORE 的信息, 请参见第 29.0 节“电气特性”。

图 2-3：建议 VCAP 的频率和 ESR 性能



## 2.5 ICSP 引脚

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

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

对于器件仿真, 请确保编程到器件中的“通信通道选择”(即 PGECx/PGEDx 引脚) 与 ICSP 到 MPLAB® ICD 2、MPLAB® ICD 3 或 MPLAB REAL ICE™ 仿真器的物理连接一致。

更多关于 ICD 2、ICD 3 和 REAL ICE 仿真器连接要求的信息, 请参见 Microchip 网站上提供的以下文档。

- 《MPLAB® ICD 2 在线调试器用户指南》(DS51331C\_CN)
- “Using MPLAB® ICD 2”(海报)(DS51265)
- “MPLAB® ICD 2 Design Advisory”(DS51566)
- “Using MPLAB® ICD 2”(海报)(DS51265)
- “MPLAB® ICD 3 Design Advisory”(DS51764)
- 《MPLAB® REAL ICE™ 在线仿真器用户指南》(DS51616A\_CN)
- “Using MPLAB® REAL ICE™ In-Circuit Emulator”(海报)(DS51749)

## 2.6 外部振荡器引脚

许多单片机可以至少有两个振荡器：一个高频主振荡器和一个低频辅助振荡器（详细信息，请参见第 8.0 节“振荡器配置”）。

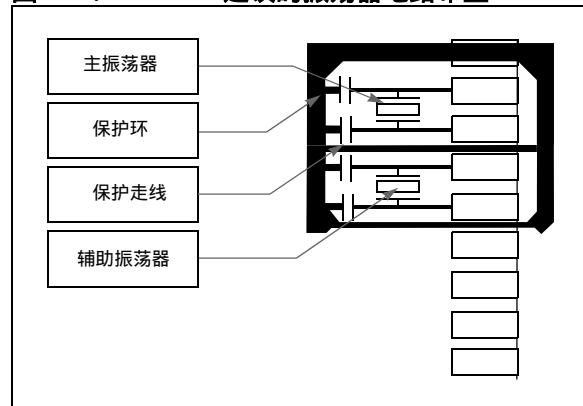
振荡器电路应放到电路板上与器件相同的一侧。将振荡器电路靠近相关振荡器引脚放置，且电路元件与引脚之间的距离不要超过 0.5 英寸（12 mm）。负载电容应在电路板的同一侧挨着振荡器本身放置。

在振荡器电路周围设置接地灌铜区，将其与周围电路隔离。接地灌铜区应直接连接到 MCU 地。不要在接地灌铜区内使用任何信号线或电源线。而且，如果使用双面的电路板，请避免在电路板上放置晶振的位置的另一面布线。图 2-4 给出了建议的电路板布局。

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

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

图 2-4：建议的振荡器电路布置



## 2.7 ICSP 工作期间的模拟和数字引脚配置

如果选择 MPLAB ICD 2、ICD 3 或 REAL ICE 仿真器作为调试器，它会通过将 AD1PCFGL 寄存器中的所有位置 1 自动初始化所有 A/D 输入引脚(ANx)为数字引脚。

用户应用固件不得清零此寄存器中与 MPLAB ICD 2、ICD 3 或 REAL ICE 仿真器初始化的 A/D 引脚相对应的位；否则，将导致调试器和器件之间发生通信错误。

如果在调试会话期间应用需要使用某些 A/D 引脚作为模拟输入引脚，那么用户应用必须在 ADC 模块初始化期间清零 AD1PCFGL 寄存器中的相应位。

当 MPLAB ICD 2、ICD 3 或 REAL ICE 仿真器用作编程器时，用户应用固件必须正确配置AD1PCFGL寄存器。仅在调试器操作期间才会自动初始化此寄存器。如果未能正确配置该寄存器，将导致所有 A/D 引脚被识别为模拟输入引脚，以致端口值被读为逻辑 0，从而可能影响用户应用的功能。

## 2.8 未使用的 I/O

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

## 3.0 CPU

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 2 章“CPU”(DS39703A\_CN)。

PIC24F CPU 采用 16 位（数据）的改进型哈佛架构，具有增强的指令集和带有长度可变的操作码字段的 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 指令可以支持无开销的程序循环结构，在任何时候都可被中断。

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）用作软件堆栈指针，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（Program Space Visibility Page，PSVPAG）寄存器定义的任何 16K 程序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）与 PIC18 相比有了显著的增强，但仍保持了一定程度的向后兼容性。该架构直接或通过简单的宏支持所有的 PIC18 指令和寻址模式。对编译器执行效率的需求促使了对 ISA 的许多改进。

内核支持固有（无操作数）寻址、相对寻址、立即数寻址、存储器直接寻址及其他三组寻址模式。所有模式都支持寄存器直接寻址和各种寄存器间接寻址模式。每组都提供了最多 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。

对于大多数指令，在每个指令周期内，内核能执行一次数据（或程序数据）存储器读操作、一次工作寄存器（数据）读操作、一次数据存储器写操作和一次程序（指令）存储器读操作。因此可以支持 3 个操作数的指令，使 3 操作数运算（即， $A + B = C$ ）能够在单周期内执行。

包含了一个高速 17 位  $\times$  17 位乘法器，显著提高了内核的运算能力和吞吐量。此乘法器支持有符号、无符号和混合模式的 16 位  $\times$  16 位或 8 位  $\times$  8 位整数乘法。所有的乘法指令都在单周期内执行。

已对 16 位 ALU 进行了改进使其具备一个支持整数除法的硬件，该硬件支持迭代的不可撤销的除法算法。它可以和 REPEAT 指令循环机制和迭代除法指令（可选）一起工作，支持 32 位（或 16 位）除以 16 位有符号和无符号整数的除法运算。所有除法运算都需要 19 个周期来完成，但是可在任何周期边界被中断。

PIC24F 具有向量异常机制，具有最多 8 个不可屏蔽的陷阱源和最多 118 个中断源。可以为每个中断源分配 7 个优先级之一。

CPU 的框图如图 3-1 所示。

### 3.1 编程模型

图 3-2 中所示为 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，并且可以由指令直接控制。表 3-1 中提供了对每个寄存器的描述。所有与编程模型相关联的寄存器都是存储器映射的。

# PIC24FJ64GB004 系列

图 3-1 : PIC24F CPU 内核框图

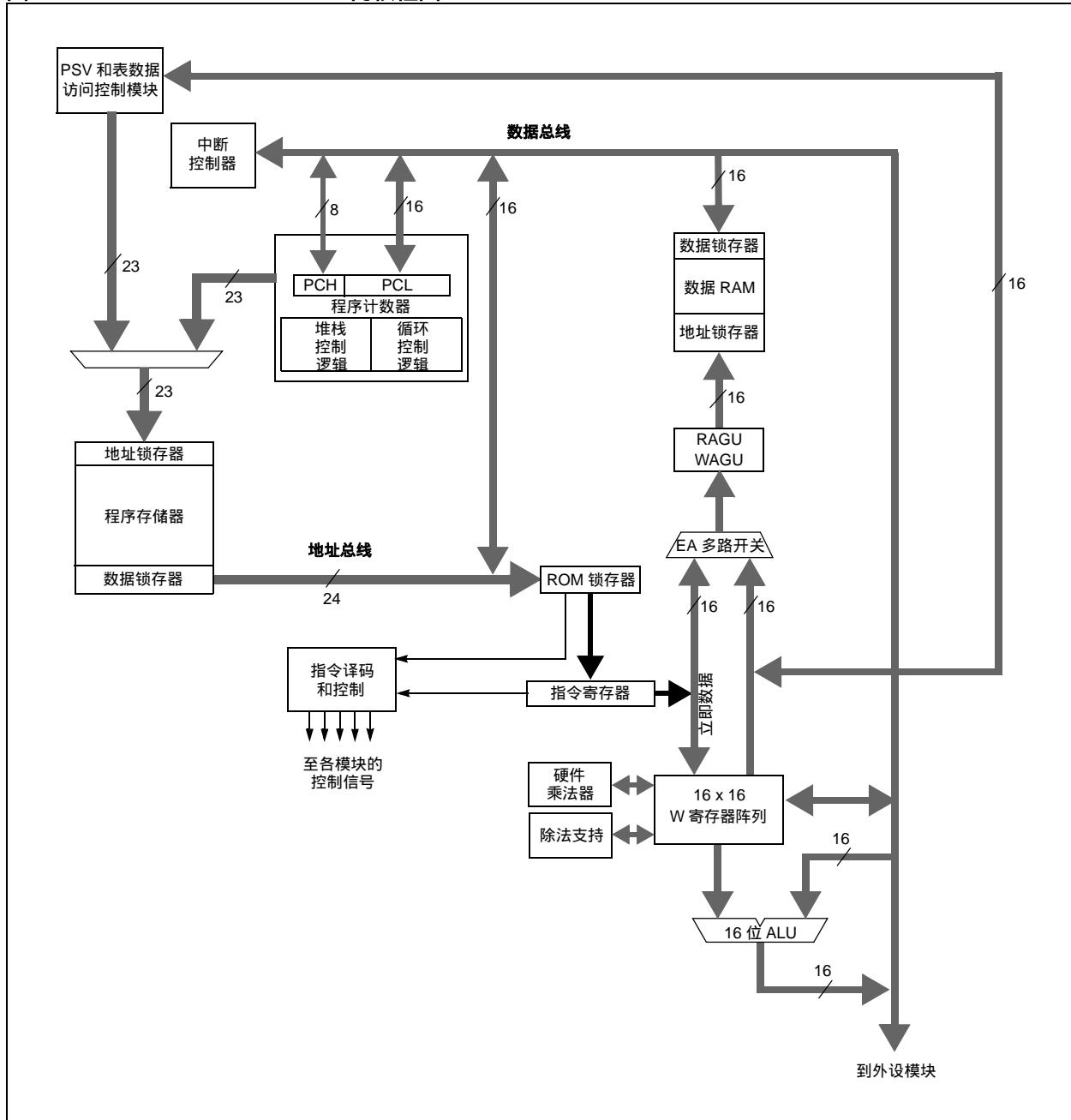
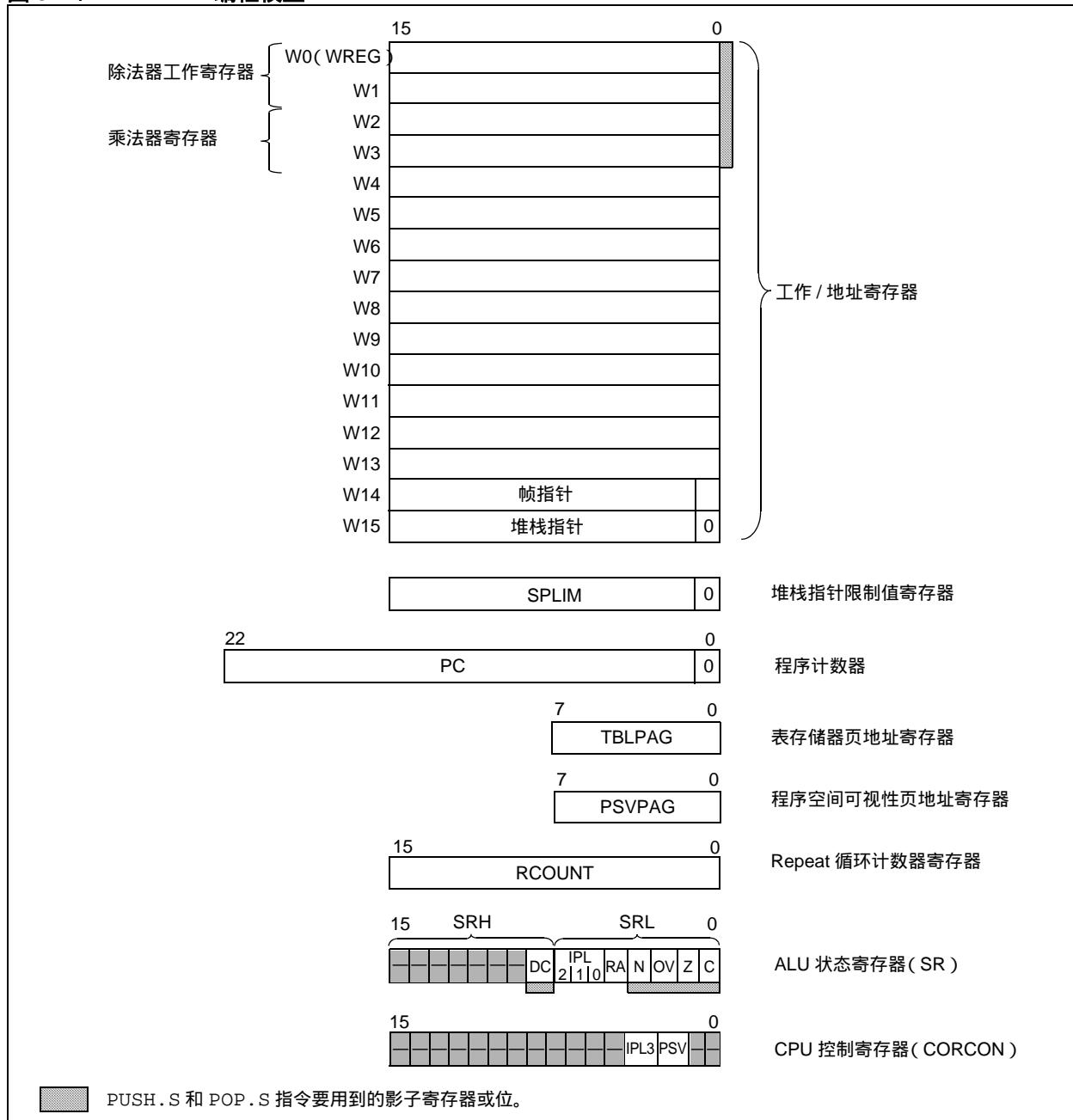


表 3-1 : CPU 内核寄存器

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 3-2 : 编程模型



# PIC24FJ64GB004 系列

## 3.2 CPU 控制寄存器

寄存器 3-1 : SR : ALU 状态寄存器

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

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

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15-9

**未实现** : 读为 0

bit 8

**DC** : ALU 半进位 / 借位标志位

1 = 结果的第 4 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)发生了向高位的进位  
0 = 结果的第 4 个低位或第 8 个低位未发生向高位的进位

bit 7-5

**IPL<2:0>** : CPU 中断优先级状态位<sup>(1,2)</sup>

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 = 未进行 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 = 在本次算术运算中有符号 (二进制补码) 运算发生了溢出  
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经将该位清零 (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 (MSb) 发生了进位  
0 = 结果的最高位未发生进位

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

2 : IPL 状态位与 IPL3 位 (CORCON<3>) 共同决定 CPU 的中断优先级 (Interrupt Priority Level , IPL)。如果 IPL3 = 1 , 则括号中的值表示 IPL。

## 寄存器 3-2 : 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	R/W-0	U-0	U-0
—	—	—	—	IPL3 <sup>(1)</sup>	PSV	—	—
bit 7							bit 0

图注 :

C = 可清零位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-4 未实现 : 读为 0

bit 3 IPL3 : CPU 中断优先级状态位<sup>(1)</sup>

1 = CPU 中断优先级大于 7

0 = CPU 中断优先级为 7 或更小

bit 2 PSV : 数据空间的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。

## 3.3 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。除非另外声明 , 否则算术运算一般采用二进制补码。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和半借位位。

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

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

### 3.3.1 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 :

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

### 3.3.2 除法器

除法模块支持具有下列数据长度的有符号和无符号整数除法运算：

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

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。

**表 3-2： 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一位或多位。
SL	将源寄存器左移一位或多位。
LSR	将源寄存器逻辑右移一位或多位。

### 3.3.3 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操作。由一个移位寄存器电路实现多位移位，在单个周期内最多可将数据算术右移或左移 15 位。所有的多位移位指令仅支持源操作数和目标结果的寄存器直接寻址模式。

在下面的表 3-2 中汇总了所有使用移位操作的指令。

## 4.0 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独立的程序和数据总线。此架构还允许在代码执行过程中直接通过数据空间访问程序空间。

### 4.1 程序地址空间

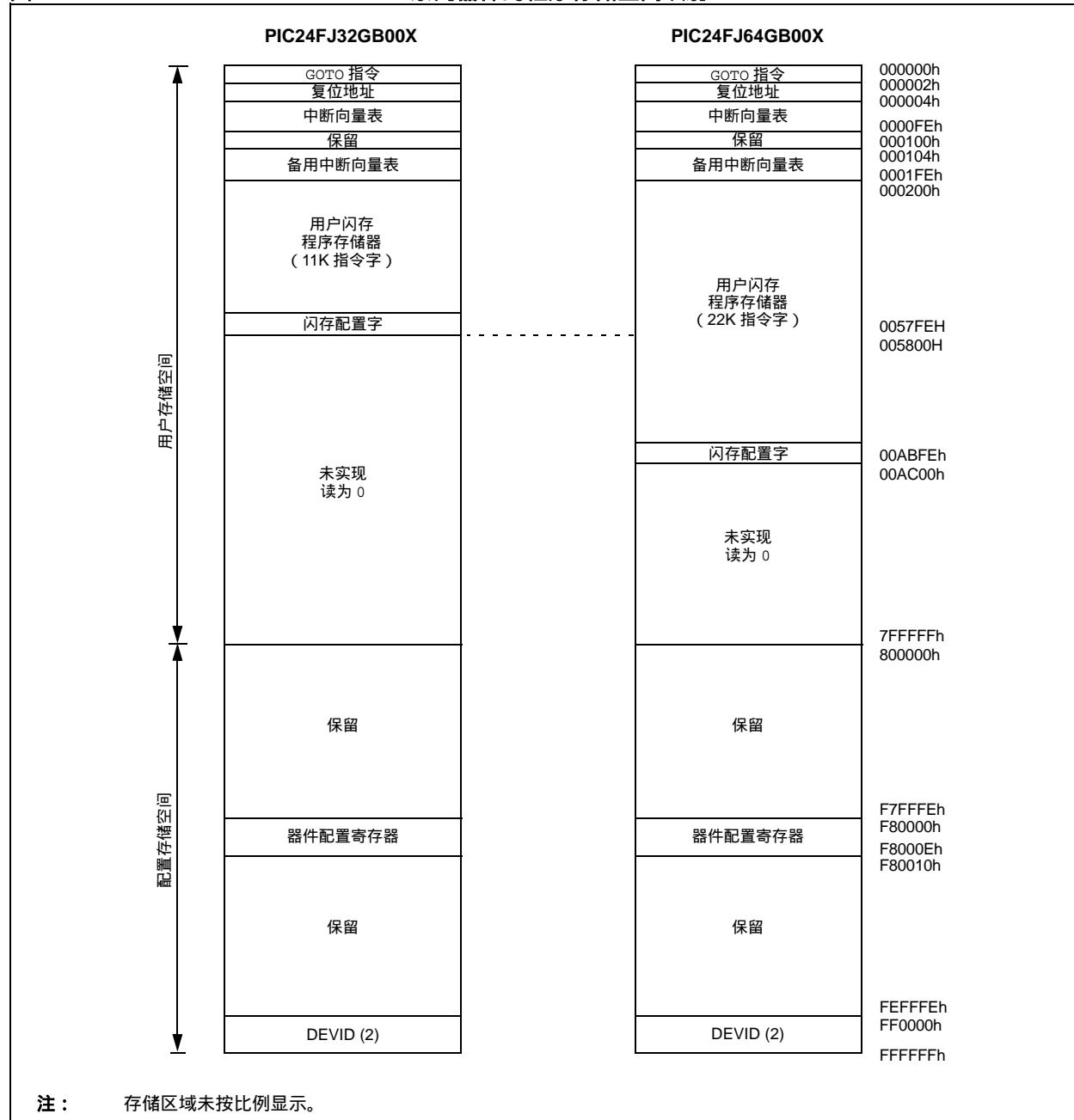
PIC24FJ64GB004 系列器件的程序地址存储空间可存储 4M 指令字。可通过由程序执行过程中 23 位程序计数

器 (PC) 和第 4.3 节 “程序存储空间与数据存储空间的接口”所述的表操作或数据空间重映射得到的 24 位值寻址这一空间。

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，这两条指令使用 TBLPAG<7> 位以允许访问配置存储空间中的配置位和器件 ID。

图 4-1 给出了 PIC24FJ64GB004 系列器件的存储器映射情况。

图 4-1： PIC24FJ64GB004 系列器件的程序存储空间映射



# PIC24FJ64GB004 系列

## 4.1.1 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。低位字的地址总是偶数，而高位字的地址为奇数（图 4-2）。

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。这种对齐方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。

## 4.1.2 存储器硬编码向量

所有 PIC24F 器件中从 000000h 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开始处。用户可在 000000h 地址编写一条 GOTO 指令以将代码的实际起始处设置为 000002h。

PIC24F 器件也具有 2 个中断向量表，地址范围分别为 000004h 至 000FFh 和 000100h 至 0001FFh。这两个中断向量表允许使用不同的 ISR 处理每个器件中断源。**第 7.1 节“中断向量表”** 提供了有关中断向量表的更多详细信息。

## 4.1.3 闪存配置字

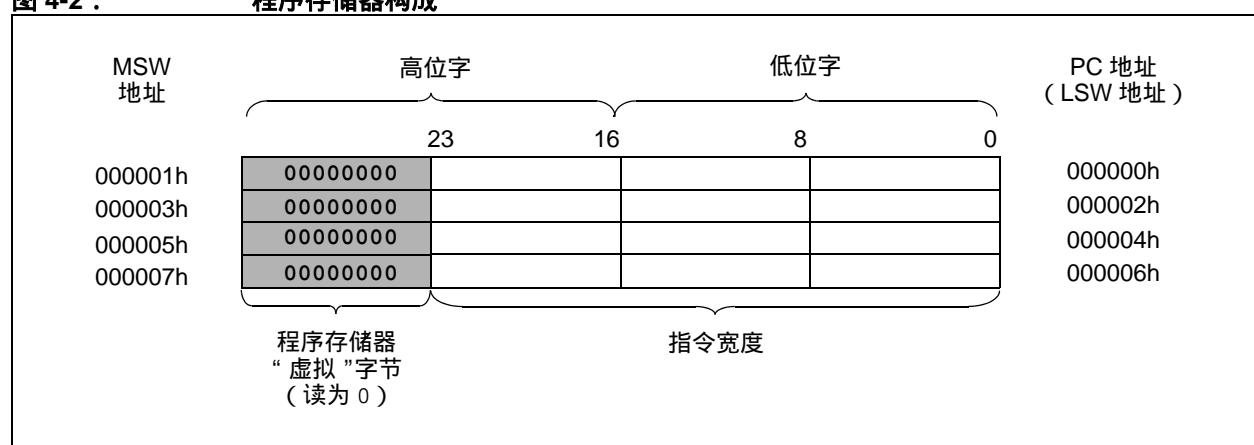
在 PIC24FJ64GB004 系列器件中，保留片上程序存储器的高 4 个字来存储配置信息。器件复位时，将配置信息复制到相应的配置寄存器中。PIC24FJ64GB004 系列器件的闪存配置字的地址如表 4-1 所示。图 4-1 中给出了闪存配置字以及其他存储器向量在存储器映射中的位置。

程序存储器中的配置字为紧凑的格式。实际配置位被映射到配置存储空间的几个不同的寄存器中。它们在闪存配置字中的顺序并不反映它们在配置空间中的相应顺序。**第 26.1 节“配置位”** 给出了器件配置字的更多详细信息。

**表 4-1： PIC24FJ64GB004 系列器件的闪存配置字**

器件	程序存储器 (字)	配置字地址
PIC24FJ32GB0	11,008	0057FCh: 0057FEh
PIC24FJ64GB0	22,016	00ABF8h: 00ABFEh

**图 4-2：** 程序存储器构成



## 4.2 数据地址空间

PIC24F 内核具有一个独立的 16 位宽的数据存储空间，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）对数据空间进行寻址，分别用于读/写操作。图 4-3 给出了数据存储空间映射。

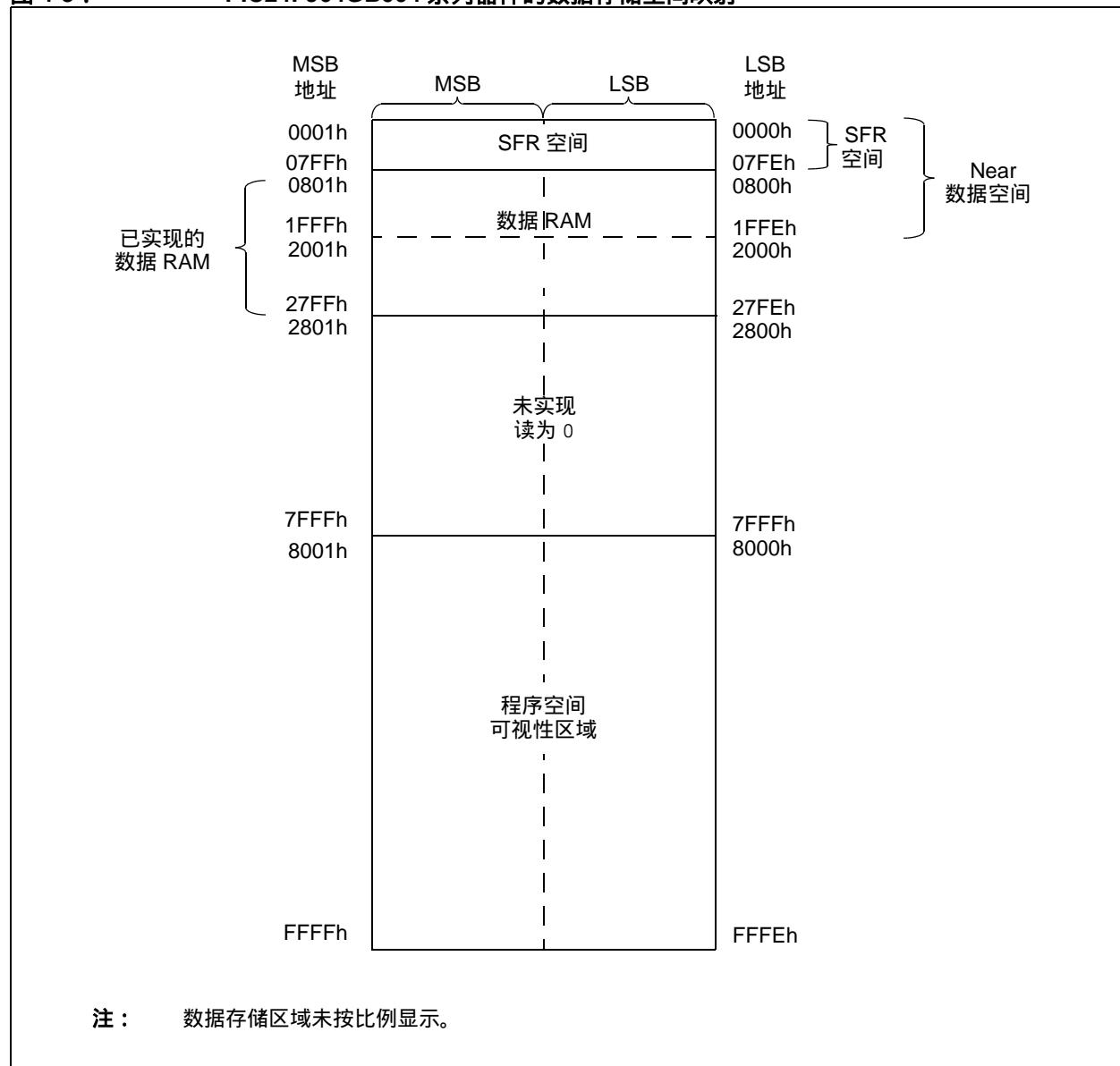
数据存储空间中的所有有效地址（Effective Address，EA）都是 16 位宽，并且指向数据空间中的字节。这种构成方式使得数据空间的地址范围为 64 KB 或 32K 字。数据存储空间的低半部分（即 EA<15> = 0）用作实现的存储器地址，而高半部分（EA<15> = 1）则保留为程序空间可视性区域（见第 4.3.3 节“使用程序空间可视性读程序存储器中的数据”）。

PIC24FJ64GB004 系列器件共实现了 16 KB 的数据存储空间。如果 EA 指向此区域以外的存储单元，将返回一个全零的字或字节。

### 4.2.1 数据空间宽度

数据存储空间组织为可字节寻址的 16 位宽的块。在数据存储器和寄存器中的数据是以 16 位字为单位对齐的，但所有数据空间的 EA 都被解析为字节。每个字的低字节（LSB）部分具有偶地址，而高字节（MSB）部分则具有奇地址。

图 4-3：PIC24FJ64GB004 系列器件的数据存储空间映射



## 4.2.2 数据存储器构成和对齐方式

为了维持与 PIC<sup>®</sup> 器件向后兼容性和提高数据存储空间的使用效率，PIC24F 指令集既支持字操作，也支持字节操作。字节访问会在内部对按字对齐的存储空间的所有有效地址计算进行调整。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 Ws + 2。

使用任何有效地址的最低位 (LSb) 决定要选择哪个字节，数据字节读操作将读取包含此字节的整个字。选中的字节将被放在数据总线的低字节处。也就是说，数据存储器和寄存器被组织为两个字节宽的并行实体，它们共享 (字) 地址译码，而写入线相互独立。数据字节写操作仅写入存储阵列或寄存器中与字节地址匹配的相应部分。

所有字访问都必须按偶地址对齐。不支持不对齐的字数据取操作。因此当混合字节和字操作或从 8 位 MCU 代码移植到此系列器件时，必须要小心。若试图进行这种不对齐的读或写操作，则会产生地址错误陷阱。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发生陷阱，从而允许系统和 / 或用户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。W 寄存器的高字节不变。

提供了一条符号扩展 (SE) 指令，允许用户把 8 位的有符号数据转换为 16 位的有符号值。或者，对于 16 位无符号数据，用户可以清零任何 W 寄存器的 MSB，方法是在相应的地址处执行一条零扩展 (ZE) 指令。

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

## 4.2.3 NEAR 数据空间

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地址单元。剩余的数据空间通过间接寻址访问。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模式。

## 4.2.4 SFR 空间

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

SFR 分布在受其控制的大量模块中，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单元，它们读为 0。表 4-2 给出了 SFR 空间的布局表，显示了实际实现 SFR 的位置。每个已实现的区域表示一个 32 字节区域，其中至少有一个地址实现为 SFR。表 4-3 至表 4-27 给出了所有实现的 SFR 及其地址的完整列表。

表 4-2：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 <sup>2</sup> C <sup>TM</sup>	UART	SPI	—	—	—	I/O	
300h	A/D	A/D/CTMU	—	—	—	—	—	—
400h	—	—	—	—	USB			—
500h	—	—	—	—	—	—	—	—
600h	PMP	RTCC	CRC/ 比较	比较器	PPS			—
700h	—	—	系统 /DS	NVM/PMD	—	—	—	—

图注：— = 此存储块中未实现的 SFR

表 4-3：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																0000	
WREG1	0002																0000	
WREG2	0004																0000	
WREG3	0006																0000	
WREG4	0008																0000	
WREG5	000A																0000	
WREG6	000C																0000	
WREG7	000E																0000	
WREG8	0010																0000	
WREG9	0012																0000	
WREG10	0014																0000	
WREG11	0016																0000	
WREG12	0018																0000	
WREG13	001A																0000	
WREG14	001C																0000	
WREG15	001E																0800	
SPLIM	0020																xxxx	
PCL	002E																0000	
PCH	0030	—	—	—	—	—	—	—	—								0000	
TBLPAG	0032	—	—	—	—	—	—	—	—								0000	
PSVPAG	0034	—	—	—	—	—	—	—	—								0000	
RCOUNT	0036																xxxx	
SR	0042	—	—	—	—	—	—	—	DC	IPL2	IPL1	IPL0	RA	N	OV	Z	C	0000
CORCON	0044	—	—	—	—	—	—	—	—	—	—	—	—	IPL3	PSV	—	—	0000
DISICNT	0052	—	—														xxxx	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-4： 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	—	CN13IE	CN12IE	CN11IE	CN10IE <sup>(1)</sup>	CN9IE <sup>(1)</sup>	CN8IE <sup>(1)</sup>	CN7IE	CN6IE	CN5IE	CN4IE	CN3IE	CN2IE	CN1IE	CNOIE	0000
CNEN2	0062	—	CN30IE	CN29IE	CN28IE <sup>(1)</sup>	CN27IE	CN26IE <sup>(1)</sup>	CN25IE <sup>(1)</sup>	—	CN23IE	CN22IE	CN21IE	CN20IE <sup>(1)</sup>	CN19IE <sup>(1)</sup>	CN18IE <sup>(1)</sup>	CN17IE <sup>(1)</sup>	CN16IE	0000
CNPUI	0068	CN15PUE	—	CN13PUE	CN12PUE	CN11PUE	CN10PUE <sup>(1)</sup>	CN9PUE <sup>(1)</sup>	CN8PUE <sup>(1)</sup>	CN7PUE	CN6PUE	CN5PUE	CN4PUE	CN3PUE	CN2PUE	CN1PUE	CNO PUE	0000
CNPUP2	006A	—	CN30PUE	CN29PUE	CN28PUE <sup>(1)</sup>	CN27PUE	CN26PUE <sup>(1)</sup>	CN25PUE <sup>(1)</sup>	—	CN23PUE	CN22PUE	CN21PUE	CN20PUE <sup>(1)</sup>	CN19PUE <sup>(1)</sup>	CN18PUE <sup>(1)</sup>	CN17PUE <sup>(1)</sup>	CN16PUE	0000

图注： — = 未实现，读为 0。复位值以十六进制格式显示。

注 1： 在 28 引脚器件上未实现，读为 0。

表 4-5：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	ALТИVT	DISI	—	—	—	—	—	—	—	—	—	—	INT2EP	INT1EP	INT0EP	0000	
IFS0	0084	—	—	AD1IF	U1TXIF	U1RXIF	SPI1IF	SPF1IF	T3IF	T2IF	OC2IF	IC2IF	—	T1IF	OC1IF	IC1IF	INT0IF	0000
IFS1	0086	U2TXIF	U2RXIF	INT2IF	T5IF	T4IF	OC4IF	OC3IF	—	—	—	—	INT1IF	CNIF	CMIF	MI2C1IF	SI2C1IF	0000
IFS2	0088	—	—	PMP1IF	—	—	—	OC5IF	—	IC5IF	IC4IF	IC3IF	—	—	—	SPI2IF	SPF2IF	0000
IFS3	008A	—	—	RTCIF	—	—	—	—	—	—	—	—	—	MI2C2IF	SI2C2IF	—	0000	
IFS4	008C	—	—	CTMUIF	—	—	—	—	LVDIF	—	—	—	—	CRCIF	U2ERIF	U1ERIF	—	0000
IFS5	008E	—	—	—	—	—	—	—	—	—	USB1IF	—	—	—	—	—	—	0000
IEC0	0094	—	—	AD1IE	U1TXIE	U1RXIE	SPI1IE	SPF1IE	T3IE	T2IE	OC2IE	IC2IE	—	T1IE	OC1IE	IC1IE	INT0IE	0000
IEC1	0096	U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	—	—	—	—	INT1IE	CNIE	CMIE	MI2C1IE	SI2C1IE	0000
IEC2	0098	—	—	PMP1IE	—	—	—	OC5IE	—	IC5IE	IC4IE	IC3IE	—	—	SPI2IE	SPF2IE	—	0000
IEC3	009A	—	—	RTCIE	—	—	—	—	—	—	—	—	—	MI2C2IE	SI2C2IE	—	0000	
IEC4	009C	—	—	CTMUIE	—	—	—	—	LVDIE	—	—	—	—	CRCIE	U2ERIE	U1ERIE	—	0000
IEC5	009E	—	—	—	—	—	—	—	—	—	USB1IE	—	—	—	—	—	—	0000
IPC0	00A4	—	T1IP2	T1IP1	T1IP0	—	OC1IP2	OC1IP1	OC1IP0	—	IC1IP2	IC1IP1	IC1IP0	—	INT0IP2	INT0IP1	INT0IP0	4444
IPC1	00A6	—	T2IP2	T2IP1	T2IP0	—	OC2IP2	OC2IP1	OC2IP0	—	IC2IP2	IC2IP1	IC2IP0	—	—	—	—	4440
IPC2	00A8	—	U1RXIP2	U1RXIP1	U1RXIP0	—	SPI1IP2	SPI1IP1	SPI1IP0	—	SPF1IP2	SPF1IP1	SPF1IP0	—	T3IP2	T3IP1	T3IP0	4444
IPC3	00AA	—	—	—	—	—	—	—	—	—	AD1IP2	AD1IP1	AD1IP0	—	U1TXIP2	U1TXIP1	U1TXIP0	0044
IPC4	00AC	—	CNIP2	CNIP1	CNIP0	—	CMIP2	CMIP1	CMIP0	—	MI2C1IP2	MI2C1IP1	MI2C1IP0	—	SI2C1IP2	SI2C1IP1	SI2C1IP0	4444
IPC5	00AE	—	—	—	—	—	—	—	—	—	—	—	—	INT1IP2	INT1IP1	INT1IP0	0004	
IPC6	00B0	—	T4IP2	T4IP1	T4IP0	—	OC4IP2	OC4IP1	OC4IP0	—	OC3IP2	OC3IP1	OC3IP0	—	—	—	—	4440
IPC7	00B2	—	U2TXIP2	U2TXIP1	U2TXIP0	—	U2RXIP2	U2RXIP1	U2RXIP0	—	INT2IP2	INT2IP1	INT2IP0	—	T5IP2	T5IP1	T5IP0	4444
IPC8	00B4	—	—	—	—	—	—	—	—	—	SPI2IP2	SPI2IP1	SPI2IP0	—	SPF2IP2	SPF2IP1	SPF2IP0	0044
IPC9	00B6	—	IC5IP2	IC5IP1	IC5IP0	—	IC4IP2	IC4IP1	IC4IP0	—	IC3IP2	IC3IP1	IC3IP0	—	—	—	—	4440
IPC10	00B8	—	—	—	—	—	—	—	—	—	OC5IP2	OC5IP1	OC5IP0	—	—	—	—	0040
IPC11	00BA	—	—	—	—	—	—	—	—	—	PMP1IP2	PMP1IP1	PMP1IP0	—	—	—	—	0040
IPC12	00BC	—	—	—	—	—	—	MI2C2IP2	MI2C2IP1	MI2C2IP0	—	SI2C2IP2	SI2C2IP1	SI2C2IP0	—	—	—	0440
IPC15	00C2	—	—	—	—	—	—	RTCIP2	RTCIP1	RTCIP0	—	—	—	—	—	—	0400	
IPC16	00C4	—	CRCIP2	CRCIP1	CRCIP0	—	U2ERIP2	U2ERIP1	U2ERIP0	—	U1ERIP2	U1ERIP1	U1ERIP0	—	—	—	—	4440
IPC18	00C8	—	—	—	—	—	—	—	—	—	—	—	—	—	LVDIP2	LVDIP1	LVDIP0	0004
IPC19	00CA	—	—	—	—	—	—	—	—	—	CTMUIP2	CTMUIP1	CTMUIP0	—	—	—	—	0040
IPC21	00CE	—	—	—	—	—	—	USB1IP2	USB1IP1	USB1IP0	—	—	—	—	—	—	0400	
INTTREG	00E0	CPUIRQ	—	VHOLD	—	ILR3	ILR2	ILR1	ILR0	—	VECNUM6	VECNUM5	VECNUM4	VECNUM3	VECNUM2	VECNUM1	VECNUM0	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-6 : 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																0000	
PR1	0102																FFFF	
T1CON	0104	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	TSYNC	TCS	—	
TMR2	0106																0000	
TMR3HLD	0108																0000	
TMR3	010A																0000	
PR2	010C																FFFF	
PR3	010E																FFFF	
T2CON	0110	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	T32	—	TCS	—	
T3CON	0112	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	—	TCS	—	
TMR4	0114																0000	
TMR5HLD	0116																0000	
TMR5	0118																0000	
PR4	011A																FFFF	
PR5	011C																FFFF	
T4CON	011E	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	T32	—	TCS	—	
T5CON	0120	TON	—	TSIDL	—	—	—	—	—	—	TGATE	TCKPS1	TCKPS0	—	—	TCS	—	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

表 4-7：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
IC1CON1	0140	—	—	ICSIDL	ICTSEL2	ICTSEL1	ICTSEL0	—	—	—	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC1CON2	0142	—	—	—	—	—	—	—	IC32	ICTRIG	TRIGSTAT	—	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000D
IC1BUF	0144	输入捕捉 1 缓冲寄存器														0000		
IC1TMR	0146	定时器值 1 寄存器														xxxx		
IC2CON1	0148	—	—	ICSIDL	ICTSEL2	ICTSEL1	ICTSEL0	—	—	—	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC2CON2	014A	—	—	—	—	—	—	—	IC32	ICTRIG	TRIGSTAT	—	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000D
IC2BUF	014C	输入捕捉 2 缓冲寄存器														0000		
IC2TMR	014E	定时器值 2 寄存器														xxxx		
IC3CON1	0150	—	—	ICSIDL	ICTSEL2	ICTSEL1	ICTSEL0	—	—	—	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC3CON2	0152	—	—	—	—	—	—	—	IC32	ICTRIG	TRIGSTAT	—	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000D
IC3BUF	0154	输入捕捉 3 缓冲寄存器														0000		
IC3TMR	0156	定时器值 3 寄存器														xxxx		
IC4CON1	0158	—	—	ICSIDL	ICTSEL2	ICTSEL1	ICTSEL0	—	—	—	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC4CON2	015A	—	—	—	—	—	—	—	IC32	ICTRIG	TRIGSTAT	—	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000D
IC4BUF	015C	输入捕捉 4 缓冲寄存器														0000		
IC4TMR	015E	定时器值 4 寄存器														xxxx		
IC5CON1	0160	—	—	ICSIDL	ICTSEL2	ICTSEL1	ICTSEL0	—	—	—	ICI1	ICI0	ICOV	ICBNE	ICM2	ICM1	ICM0	0000
IC5CON2	0162	—	—	—	—	—	—	—	IC32	ICTRIG	TRIGSTAT	—	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000D
IC5BUF	0164	输入捕捉 5 缓冲寄存器														0000		
IC5TMR	0166	定时器值 5 寄存器														xxxx		

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-8：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
OC1CON1	0190	—	—	OCSIDL	OCTSEL2	OCTSEL1	OCTSEL0	ENFLT2	ENFLT1	ENFLT0	OCFLT2	OCFLT1	OCFLT0	TRIGMODE	OCM2	OCM1	OCM0	0000
OC1CON2	0192	FLTMD	FLTOUT	FLTTRIEN	OCINV	—	DCB1	DCB0	OC32	OCTRIG	TRIGSTAT	OCTRIS	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000C
OC1RS	0194	输出比较 1 辅助寄存器															0000	
OC1R	0196	输出比较 1 寄存器															0000	
OC1TMR	0198	定时器值 1 寄存器															xxxx	
OC2CON1	019A	—	—	OCSIDL	OCTSEL2	OCTSEL1	OCTSEL0	ENFLT2	ENFLT1	ENFLT0	OCFLT2	OCFLT1	OCFLT0	TRIGMODE	OCM2	OCM1	OCM0	0000
OC2CON2	019C	FLTMD	FLTOUT	FLTTRIEN	OCINV	—	DCB1	DCB0	OC32	OCTRIG	TRIGSTAT	OCTRIS	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000C
OC2RS	019E	输出比较 2 辅助寄存器															0000	
OC2R	01A0	输出比较 2 寄存器															0000	
OC2TMR	01A2	定时器值 2 寄存器															xxxx	
OC3CON1	01A4	—	—	OCSIDL	OCTSEL2	OCTSEL1	OCTSEL0	ENFLT2	ENFLT1	ENFLT0	OCFLT2	OCFLT1	OCFLT0	TRIGMODE	OCM2	OCM1	OCM0	0000
OC3CON2	01A6	FLTMD	FLTOUT	FLTTRIEN	OCINV	—	DCB1	DCB0	OC32	OCTRIG	TRIGSTAT	OCTRIS	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000C
OC3RS	01A8	输出比较 3 辅助寄存器															0000	
OC3R	01AA	输出比较 3 寄存器															0000	
OC3TMR	01AC	定时器值 3 寄存器															xxxx	
OC4CON1	01AE	—	—	OCSIDL	OCTSEL2	OCTSEL1	OCTSEL0	ENFLT2	ENFLT1	ENFLT0	OCFLT2	OCFLT1	OCFLT0	TRIGMODE	OCM2	OCM1	OCM0	0000
OC4CON2	01B0	FLTMD	FLTOUT	FLTTRIEN	OCINV	—	DCB1	DCB0	OC32	OCTRIG	TRIGSTAT	OCTRIS	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000C
OC4RS	01B2	输出比较 4 辅助寄存器															0000	
OC4R	01B4	输出比较 4 寄存器															0000	
OC4TMR	01B6	定时器值 4 寄存器															xxxx	
OC5CON1	01B8	—	—	OCSIDL	OCTSEL2	OCTSEL1	OCTSEL0	ENFLT2	ENFLT1	ENFLT0	OCFLT2	OCFLT1	OCFLT0	TRIGMODE	OCM2	OCM1	OCM0	0000
OC5CON2	01BA	FLTMD	FLTOUT	FLTTRIEN	OCINV	—	DCB1	DCB0	OC32	OCTRIG	TRIGSTAT	OCTRIS	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0	000C
OC5RS	01BC	输出比较 5 辅助寄存器															0000	
OC5R	01BE	输出比较 5 寄存器															0000	
OC5TMR	01C0	定时器值 5 寄存器															xxxx	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-9 : I<sup>2</sup>C<sup>TM</sup> 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	—	—	—	—	—	—	—	—	接收寄存器							0000	
I2C1TRN	0202	—	—	—	—	—	—	—	—	发送寄存器							00FF	
I2C1BRG	0204	—	—	—	—	—	—	—	—	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/A	P	S	R/W	RBF	TBF	0000
I2C1ADD	020A	—	—	—	—	—	—	—	—	地址寄存器							0000	
I2C1MSK	020C	—	—	—	—	—	—	—	—	地址掩码寄存器							0000	
I2C2RCV	0210	—	—	—	—	—	—	—	—	接收寄存器							0000	
I2C2TRN	0212	—	—	—	—	—	—	—	—	发送寄存器							00FF	
I2C2BRG	0214	—	—	—	—	—	—	—	—	波特率发生器寄存器							0000	
I2C2CON	0216	I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN	GCEN	STREN	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	1000
I2C2STAT	0218	ACKSTAT	TRSTAT	—	—	—	BCL	GCSTAT	ADD10	IWCOL	I2COV	D/A	P	S	R/W	RBF	TBF	0000
I2C2ADD	021A	—	—	—	—	—	—	—	—	地址寄存器							0000	
I2C2MSK	021C	—	—	—	—	—	—	—	—	地址掩码寄存器							0000	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-10 : 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	PDSELO	STSEL	0000
U1STA	0222	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U1TXREG	0224	—	—	—	—	—	—	—	—	发送寄存器							xxxx	
U1RXREG	0226	—	—	—	—	—	—	—	—	接收寄存器							0000	
U1BRG	0228	—	—	—	—	—	—	—	—	波特率发生器预分频器寄存器							0000	
U2MODE	0230	UARTEN	—	USIDL	IREN	RTSMD	—	UEN1	UEN0	WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSELO	STSEL	0000
U2STA	0232	UTXISEL1	UTXINV	UTXISEL0	—	UTXBRK	UTXEN	UTXBF	TRMT	URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA	0110
U2TXREG	0234	—	—	—	—	—	—	—	—	发送寄存器							xxxx	
U2RXREG	0236	—	—	—	—	—	—	—	—	接收寄存器							0000	
U2BRG	0238	—	—	—	—	—	—	—	—	波特率发生器预分频器寄存器							0000	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-11 : 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	—	SPIISDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0	SRMPT	SPIROV	SRXMPt	SISEL2	SISEL1	SISEL0	SPITBF	SPIRBF	0000
SPI1CON1	0242	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0	0000
SPI1CON2	0244	FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—	—	—	—	—	—	—	SPIFE	SPIBEN	0000
SPI1BUF	0248	发送和接收缓冲区																0000
SPI2STAT	0260	SPIEN	—	SPIISDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0	SRMPT	SPIROV	SRXMPt	SISEL2	SISEL1	SISEL0	SPITBF	SPIRBF	0000
SPI2CON1	0262	—	—	—	DISSCK	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0	0000
SPI2CON2	0264	FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—	—	—	—	—	—	—	SPIFE	SPIBEN	0000
SPI2BUF	0268	发送和接收缓冲区																0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-12 : 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 <sup>(1)</sup>	Bit 9 <sup>(1)</sup>	Bit 8 <sup>(1)</sup>	Bit 7 <sup>(1)</sup>	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	所有复位时的状态
TRISA	02C0	—	—	—	—	—	TRISA10	TRISA9	TRISA8	TRISA7	—	—	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	079F
PORTA	02C2	—	—	—	—	—	RA10	RA9	RA8	RA7	—	—	RA4	RA3	RA2	RA1	RA0	xxxx
LATA	02C4	—	—	—	—	—	LATA10	LATA9	LATA8	LATA7	—	—	LATA4	LATA3	LATA2	LATA1	LATA0	xxxx
ODCA	02C6	—	—	—	—	—	ODA10	ODA9	ODA8	ODA7	—	—	ODA4	ODA3	ODA2	ODA1	ODA0	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。显示的复位值适用于 44 引脚器件。

注 1：这些位在 28 引脚器件上未实现，读为 0。

**表 4-13 : 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	—	TRISB11	TRISB10	TRISB9	TRISB8	TRISB7	—	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	EFBF
PORTB	02CA	RB15	RB14	RB13	—	RB11	RB10	RB9	RB8	RB7	—	RB5	RB4	RB3	RB2	RB1	RB0	xxxx
LATB	02CC	LATB15	LATB14	LATB13	—	LATB11	LATB10	LATB9	LATB8	LATB7	—	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxx
ODCB	02CE	ODB15	ODB14	ODB13	—	ODB11	ODB10	ODB9	ODB8	ODB7	—	ODB5	ODB4	ODB3	ODB2	ODB1	ODB0	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-14 : PORTC 寄存器映射**

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9 <sup>(1)</sup>	Bit 8 <sup>(1)</sup>	Bit 7 <sup>(1)</sup>	Bit 6 <sup>(1)</sup>	Bit 5 <sup>(1)</sup>	Bit 4 <sup>(1)</sup>	Bit 3 <sup>(1)</sup>	Bit 2 <sup>(1)</sup>	Bit 1 <sup>(2)</sup>	Bit 0 <sup>(1)</sup>	所有复位时的状态
TRISC	02D0	—	—	—	—	—	—	TRISC9	TRISC8	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	03FF
PORTC	02D2	—	—	—	—	—	—	RC9	RC8	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxx
LATC	02D4	—	—	—	—	—	—	LATC9	LATC8	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxx
ODCC	02D6	—	—	—	—	—	—	ODC9	ODC8	ODC7	ODC6	ODC5	ODC4	ODC3	ODC2	ODC1	ODC0	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。显示的复位值适用于 44 引脚器件。

注 1：这些位在 28 引脚器件上未实现，读为 0。

表 4-15：引脚配置寄存器映射

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	—	—	—	—	—	—	—	—	—	—	—	—	—	RTSECSEL1	RTSECSEL0	PMPTTL	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

表 4-16：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																	xxxx
ADC1BUF1	0302																	xxxx
ADC1BUF2	0304																	xxxx
ADC1BUF3	0306																	xxxx
ADC1BUF4	0308																	xxxx
ADC1BUF5	030A																	xxxx
ADC1BUF6	030C																	xxxx
ADC1BUF7	030E																	xxxx
ADC1BUF8	0310																	xxxx
ADC1BUF9	0312																	xxxx
ADC1BUFA	0314																	xxxx
ADC1BUFB	0316																	xxxx
ADC1BUFC	0318																	xxxx
ADC1BUFD	031A																	xxxx
ADC1BUFE	031C																	xxxx
ADC1BUFF	031E																	xxxx
AD1CON1	0320	ADON	—	ADSIDL	—	—	—	FORM1	FORM0	SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE	0000
AD1CON2	0322	VCFG2	VCFG1	VCFG0	r	—	CSCNA	—	—	BUFS	—	SMPI3	SMPI2	SMPI1	SMPI0	BUFM	ALTS	0000
AD1CON3	0324	ADRC	r	r	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0	ADCS7	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0000
AD1CHS	0328	CH0NB	—	—	CH0SB4	CH0SB3	CH0SB2	CH0SB1	CH0SB0	CH0NA	—	—	CH0SA4	CH0SA3	CH0SA2	CH0SA1	CH0SA0	0000
AD1PCFG	032C	PCFG15	PCFG14	PCFG13	PCFG12 <sup>(1)</sup>	PCFG11	PCFG10	PCFG9	PCFG8 <sup>(1)</sup>	PCFG7 <sup>(1)</sup>	PCFG6 <sup>(1)</sup>	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000
AD1CSSL	0330	CSSL15	CSSL14	CSSL13	CSSL12 <sup>(1)</sup>	CSSL11	CSSL10	CSSL9	CSSL8 <sup>(1)</sup>	CSSL7 <sup>(1)</sup>	CSSL6 <sup>(1)</sup>	CSSL5	CSSL4	CSSL3	CSSL2	CSSL1	CSSL0	0000

图注：— = 未实现，读为 0，r = 保留，保持为 0。复位值以十六进制格式显示。

注 1：这些位在 28 引脚器件上未实现，读为 0。

表 4-17：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
CTMUCON	033E	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0	—	—	—	—	—	—	—	—	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

表 4-18：USB OTG 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	
U1OTGIR	0480	—	—	—	—	—	—	—	IDIF	T1MSECIF	LSTATEIF	ACTVIF	SESVDF	SESENDIF	—	VBUVDIF	0000		
U1OTGIE	0482	—	—	—	—	—	—	—	IDIE	T1MSECIE	LSTATEIE	ACTVIE	SESVDIE	SESENDIE	—	VBUVDIE	0000		
U1OTGSTAT	0484	—	—	—	—	—	—	—	ID	—	LSTATE	—	SESVD	SESEND	—	VBUVD	0000		
U1OTGCON	0486	—	—	—	—	—	—	—	DPPULUP	DMPULUP	DPPULDWN	DMPULDWN	VBUSON	OTGEN	VBUCHG	VBUUDIS	0000		
U1PWRC	0488	—	—	—	—	—	—	—	UACTPND	—	—	USLPGRD	—	—	USUSPND	USBPWR	0000		
U1IR	048A <sup>(1)</sup>	—	—	—	—	—	—	—	STALLIF	—	RESUMEIF	IDLEIF	TRNIF	SOFIF	UERRIF	URSTIF	0000		
		—	—	—	—	—	—	—	STALLIF	ATTACHIF <sup>(1)</sup>	RESUMEIF	IDLEIF	TRNIF	SOFIF	UERRIF	DETACHIF <sup>(1)</sup>	0000		
U1IE	048C <sup>(1)</sup>	—	—	—	—	—	—	—	STALLIE	—	RESUMEIE	IDLEIE	TRNIE	SOFIE	UERRIE	URSTIE	0000		
		—	—	—	—	—	—	—	STALLIE	ATTACHIE <sup>(1)</sup>	RESUMEIE	IDLEIE	TRNIE	SOFIE	UERRIE	DETACHIE <sup>(1)</sup>	0000		
U1EIR	048E <sup>(1)</sup>	—	—	—	—	—	—	—	BTSEF	—	DMAEF	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF	0000		
		—	—	—	—	—	—	—	BTSEF	—	DMAEF	BTOEF	DFN8EF	CRC16EF	EOFER <sup>(1)</sup>	PIDEF	0000		
U1EIE	0490 <sup>(1)</sup>	—	—	—	—	—	—	—	BTSEE	—	DMAEE	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE	0000		
		—	—	—	—	—	—	—	BTSEE	—	DMAEE	BTOEE	DFN8EE	CRC16EE	EOFEE <sup>(1)</sup>	PIDEE	0000		
U1STAT	0492	—	—	—	—	—	—	—	ENDPT3	ENDPT2	ENDPT1	ENDPT0	DIR	PPBI	—	—	0000		
U1CON	0494 <sup>(1)</sup>	—	—	—	—	—	—	—	—	SE0	PKTDIS	—	HOSTEN	RESUME	PPBRST	USBEN	0000		
		—	—	—	—	—	—	—	JSTATE <sup>(1)</sup>	SE0	TOKBUSY <sup>(1)</sup>	RESET <sup>(1)</sup>	HOSTEN	RESUME	PPBRST	SOFEN <sup>(1)</sup>	0000		
U1ADDR	0496	—	—	—	—	—	—	—	LSPDEN <sup>(1)</sup>	USB 器件地址 (DEVADDR) 寄存器								0000	
U1BDTP1	0498	—	—	—	—	—	—	—	缓冲区描述符表基址寄存器								—	0000	
U1FRML	049A	—	—	—	—	—	—	—	帧计数寄存器低字节								0000		
U1FRMH	049C	—	—	—	—	—	—	—	帧计数寄存器高字节								0000		
U1TOK <sup>(2)</sup>	049E	—	—	—	—	—	—	—	PID3	PID2	PID1	PID0	EP3	EP2	EP1	EP0	0000		
U1SOE <sup>(2)</sup>	04A0	—	—	—	—	—	—	—	帧起始计数寄存器								0000		
U1CNFG1	04A6	—	—	—	—	—	—	—	UTEYE	UOEMON	—	USBSIDL	—	—	PPB1	PPBO	0000		
U1CNFG2	04A8	—	—	—	—	—	—	—	—	UVCMPSSEL	PUVBUS	EXTI2CE N	UVBUSDIS	UVCMPDIS	UTRDIS	0000		0000	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

注 1：模块工作在主机模式下时的备用寄存器或位定义。

2：此寄存器仅在主机模式下可用。

表 4-18：USB OTG 寄存器映射（续）

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
U1EP0	04AA	—	—	—	—	—	—	—	LSPD <sup>(1)</sup>	RETRY-DIS <sup>(1)</sup>	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP1	04AC	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP2	04AE	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP3	04B0	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP4	04B2	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP5	04B4	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP6	04B6	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP7	04B8	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP8	04BA	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP9	04BC	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP10	04BE	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP11	04C0	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP12	04C2	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP13	04C4	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP14	04C6	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1EP15	04C8	—	—	—	—	—	—	—	—	—	—	EPOCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK	0000	
U1PWMMRRS	04CC	USB 电源 PWM 占空比寄存器								USB 电源 PWM 周期寄存器								0000
U1PWMCN	04CE	PWMEN	—	—	—	—	PWMPOL	CNTEN	—	—	—	—	—	—	—	—	0000	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

注 1：模块工作在主机模式下时的备用寄存器或位定义。

2：此寄存器仅在主机模式下可用。

表 4-19：并行主 / 从端口寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	所有复位时的状态
PMCON	0600	PPMEN	—	PSIDL	ADRMUX1	ADRMUX0	PTBEEN	PTWREN	PTRDEN	CSF1	CSF0	ALP	—	CS1P	BEP	WRSP	RDSP	0000
PMMODE	0602	BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0	WAITB1	WAITB0	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1	WAITE0	0000
PMADDR	0604	—	CS1	—	—	—	ADDR10 <sup>(1)</sup>	ADDR9 <sup>(1)</sup>	ADDR8 <sup>(1)</sup>	ADDR7 <sup>(1)</sup>	ADDR6 <sup>(1)</sup>	ADDR5 <sup>(1)</sup>	ADDR4 <sup>(1)</sup>	ADDR3 <sup>(1)</sup>	ADDR2 <sup>(1)</sup>	ADDR1	ADDR0	0000
并行端口数据输出寄存器 1 (缓冲区 0 和 1)																		0000
PMDOUT1	0606	并行端口数据输出寄存器 2 (缓冲区 2 和 3)																0000
PMDIN1	0608	并行端口数据输入寄存器 1 (缓冲区 0 和 1)																0000
PMDIN2	060A	并行端口数据输入寄存器 2 (缓冲区 2 和 3)																0000
PMAEN	060C	—	PTEN14	—	—	—	PTEN10 <sup>(1)</sup>	PTEN9 <sup>(1)</sup>	PTEN8 <sup>(1)</sup>	PTEN7 <sup>(1)</sup>	PTEN6 <sup>(1)</sup>	PTEN5 <sup>(1)</sup>	PTEN4 <sup>(1)</sup>	PTEN3 <sup>(1)</sup>	PTEN2 <sup>(1)</sup>	PTEN1	PTEN0	0000
PMSTAT	060E	IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F	OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

注 1：这些位在 28 引脚器件上未实现，读为 0。

**表 4-20：** 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
ALRVAL	0620																	xxxx
ALCFGRPT	0622	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0	0000
RTCVAL	0624																	xxxx
RCFGCAL	0626	RTCCEN	—	RTCWRREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	xxxx

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-21：** 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
CRCCON1	0640	CRCEN	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0	CRCFUL	CRCMPT	CRCISEL	CRCGO	LENDIAN	—	—	—	0000
CRCCON2	0642	—	—	—	DWIDTH4	DWIDTH3	DWIDTH2	DWIDTH1	DWIDTH0	—	—	—	PLEN4	PLEN3	PLEN2	PLEN1	PLEN0	0000
CRCXORL	0644	X15	X14	X13	X12	X11	X10	X9	X8	X7	X6	X5	X4	X3	X2	X1	—	0000
CRCXORH	0646	X31	X30	X29	X28	X27	X26	X25	X24	X23	X22	X21	X20	X19	X19	X17	X16	0000
CRCDATL	0648																	xxxx
CRCDATH	064A																	xxxx
CRCWDATL	064C																	xxxx
CRCWDATH	064E																	xxxx

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

**表 4-22：** 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	0650	CMIDL	—	—	—	—	C3EVT	C2EVT	C1EVT	—	—	—	—	C3OUT	C2OUT	C1OUT	0000	
CVRCON	0652	—	—	—	—	—	CVREFP	CVREFM1	CVREFM0	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000
CM1CON	0654	CEN	COE	CPOL	—	—	—	CEVT	COUT	EVPOL1	EVPOLO	—	CREF	—	—	CCH1	CCH0	0000
CM2CON	065C	CEN	COE	CPOL	—	—	—	CEVT	COUT	EVPOL1	EVPOLO	—	CREF	—	—	CCH1	CCH0	0000
CM3CON	0664	CEN	COE	CPOL	—	—	—	CEVT	COUT	EVPOL1	EVPOLO	—	CREF	—	—	CCH1	CCH0	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

表 4-23：外设引脚选择寄存器映射

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
RPINR0	0680	—	—	—	INT1R4	INT1R3	INT1R2	INT1R1	INT1R0	—	—	—	—	—	—	—	1F00	
RPINR1	0682	—	—	—	—	—	—	—	—	—	—	—	INT2R4	INT2R3	INT2R2	INT2R1	INT2R0	001F
RPINR3	0686	—	—	—	T3CKR4	T3CKR3	T3CKR2	T3CKR1	T3CKR0	—	—	—	T2CKR4	T2CKR3	T2CKR2	T2CKR1	T2CKR0	1F1F
RPINR4	0688	—	—	—	T5CKR4	T5CKR3	T5CKR2	T5CKR1	T5CKR0	—	—	—	T4CKR4	T4CKR3	T4CKR2	T4CKR1	T4CKR0	1F1F
RPINR7	068E	—	—	—	IC2R4	IC2R3	IC2R2	IC2R1	IC2R0	—	—	—	IC1R4	IC1R3	IC1R2	IC1R1	IC1R0	1F1F
RPINR8	0690	—	—	—	IC4R4	IC4R3	IC4R2	IC4R1	IC4R0	—	—	—	IC3R4	IC3R3	IC3R2	IC3R1	IC3R0	1F1F
RPINR9	0692	—	—	—	—	—	—	—	—	—	—	—	IC5R4	IC5R3	IC5R2	IC5R1	IC5R0	001F
RPINR11	0696	—	—	—	OCFBR4	OCFBR3	OCFBR2	OCFBR1	OCFBR0	—	—	—	OCFAR4	OCFAR3	OCFAR2	OCFAR1	OCFAR0	1F1F
RPINR18	06A4	—	—	—	U1CTSR4	U1CTSR3	U1CTSR2	U1CTSR1	U1CTSR0	—	—	—	U1RXR4	U1RXR3	U1RXR2	U1RXR1	U1RXR0	1F1F
RPINR19	06A6	—	—	—	U2CTSR4	U2CTSR3	U2CTSR2	U2CTSR1	U2CTSR0	—	—	—	U2RXR4	U2RXR3	U2RXR2	U2RXR1	U2RXR0	1F1F
RPINR20	06A8	—	—	—	SCK1R4	SCK1R3	SCK1R2	SCK1R1	SCK1R0	—	—	—	SDI1R4	SDI1R3	SDI1R2	SDI1R1	SDI1R0	1F1F
RPINR21	06AA	—	—	—	—	—	—	—	—	—	—	—	SS1R4	SS1R3	SS1R2	SS1R1	SS1R0	001F
RPINR22	06AC	—	—	—	SCK2R4	SCK2R3	SCK2R2	SCK2R1	SCK2R0	—	—	—	SDI2R4	SDI2R3	SDI2R2	SDI2R1	SDI2R0	1F1F
RPINR23	06AE	—	—	—	—	—	—	—	—	—	—	—	SS2R4	SS2R3	SS2R2	SS2R1	SS2R0	001F
RPOR0	06C0	—	—	—	RP1R4	RP1R3	RP1R2	RP1R1	RP1R0	—	—	—	RP0R4	RP0R3	RP0R2	RP0R1	RP0R0	0000
RPOR1	06C2	—	—	—	RP3R4	RP3R3	RP3R2	RP3R1	RP3R0	—	—	—	RP2R4	RP2R3	RP2R2	RP2R1	RP2R0	0000
RPOR2	06C4	—	—	—	RP5R4	RP5R3	RP5R2	RP5R1	RP5R0	—	—	—	RP4R4	RP4R3	RP4R2	RP4R1	RP4R0	0000
RPOR3	06C6	—	—	—	RP7R4	RP7R3	RP7R2	RP7R1	RP7R0	—	—	—	RP6R4	RP6R3	RP6R2	RP6R1	RP6R0	0000
RPOR4	06C8	—	—	—	RP9R4	RP9R3	RP9R2	RP9R1	RP9R0	—	—	—	RP8R4	RP8R3	RP8R2	RP8R1	RP8R0	0000
RPOR5	06CA	—	—	—	RP11R4	RP11R3	RP11R2	RP11R1	RP11R0	—	—	—	RP10R4	RP10R3	RP10R2	RP10R1	RP10R0	0000
RPOR6	06CC	—	—	—	RP13R4	RP13R3	RP13R2	RP13R1	RP13R0	—	—	—	—	—	—	—	0000	
RPOR7	06CE	—	—	—	RP15R4	RP15R3	RP15R2	RP15R1	RP15R0	—	—	—	RP14R4	RP14R3	RP14R2	RP14R1	RP14R0	0000
RPOR8 <sup>(1)</sup>	06D0	—	—	—	RP17R4	RP17R3	RP17R2	RP17R1	RP17R0	—	—	—	RP16R4	RP16R3	RP16R2	RP16R1	RP16R0	0000
RPOR9 <sup>(1)</sup>	06D2	—	—	—	RP19R4	RP19R3	RP19R2	RP19R1	RP19R0	—	—	—	RP18R4	RP18R3	RP18R2	RP18R1	RP18R0	0000
RPOR10 <sup>(1)</sup>	06D4	—	—	—	RP21R4	RP21R3	RP21R2	RP21R1	RP21R0	—	—	—	RP20R4	RP20R3	RP20R2	RP20R1	RP20R0	0000
RPOR11 <sup>(1)</sup>	06D6	—	—	—	RP23R4	RP23R3	RP23R2	RP23R1	RP23R0	—	—	—	RP22R4	RP22R3	RP22R2	RP22R1	RP22R0	0000
RPOR12 <sup>(1)</sup>	06D8	—	—	—	RP25R4	RP25R3	RP25R2	RP25R1	RP25R0	—	—	—	RP24R4	RP24R3	RP24R2	RP24R1	RP24R0	0000

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

注 1：这些寄存器在 28 引脚器件上未实现，读为 0。

表 4-24：系统寄存器映射

寄存器名称	地址	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	所有复位时的状态
RCON	0740	TRAPR	IOPUWR	—	—	—	DPSLP	CM	PMSLP	EXTR	SWR	SWDTEN	WDTO	SLEEP	IDLE	BOR	POR	注 1
OSCCON	0742	—	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0	CLKLOCK	IOLOCK	LOCK	—	CF	POSCEN	SOSCEN	OSWEN	注 2
CLKDIV	0744	ROI	DOZE2	DOZE1	DOZE0	DOZEN	RCDIV2	RCDIV1	RCDIV0	CPDIV1	CPDIV0	PLLEN	—	—	—	—	—	0100
OSCTUN	0748	—	—	—	—	—	—	—	—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	0000	
REFOCON	074E	ROEN	—	ROSSLP	ROSEL	RODIV3	RODIV2	RODIV1	RODIV0	—	—	—	—	—	—	—	0000	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

注 1：RCON 寄存器的复位值与复位事件的类型有关。更多信息，请参见第 6.0 节“复位”。

2：OSCCON 寄存器的复位值与复位事件的类型以及器件配置有关。更多信息，请参见第 8.0 节“振荡器配置”。

表 4-25：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 <sup>(1)</sup>
DSCON	758	DSEN	—	—	—	—	—	—	—	—	—	—	—	—	—	DSBOR	RELEASE	0000
DSWAKE	075A	—	—	—	—	—	—	—	DSINT0	DSFLT	—	—	DSWDT	DSRTC	DSMCLR	—	DSPOR	0001
DSGPRO	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。复位值以十六进制格式显示。

注 1：仅在发生 VDD 上电复位事件时才复位深度休眠寄存器。

表 4-26：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	—	—	—	—	—	—	ERASE	—	—	NVMOP3	NVMOP2	NVMOP1	NVMOP0	0000 <sup>(1)</sup>
NVMKEY	0766	—	—	—	—	—	—	—	—	—	NVMKEY 寄存器 <7:0>						0000	

图注：— = 未实现，读为 0。复位值以十六进制格式显示。

注 1：显示的复位值仅针对上电复位。其他复位状态的值取决于复位时存储器写操作或擦除操作的状态。

**表 4-27 : 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	T5MD	T4MD	T3MD	T2MD	T1MD	—	—	—	I2C1MD	U2MD	U1MD	SPI2MD	SPI1MD	—	—	ADC1MD	0000
PMD2	0772	—	—	—	IC5MD	IC4MD	IC3MD	IC2MD	IC1MD	—	—	—	OC5MD	OC4MD	OC3MD	OC2MD	OC1MD	0000
PMD3	0774	—	—	—	—	—	CMPMD	RTCCMD	PMPMD	CRCMD	—	—	—	—	—	I2C2MD	—	0000
PMD4	0776	—	—	—	—	—	—	—	—	UPWMMD	—	—	REFOMD	CTMUMD	LVDMD	USB1MD	—	0000

图注： — = 未实现，读为 0。复位值以十六进制格式显示。

## 4.2.5 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。此指针总是指向第一个可用的空字，从低地址向高地址方向增长。它在弹出堆栈之前递减，在压入堆栈后递增，如图 4-4 所示。对于执行任何 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 始终清零。

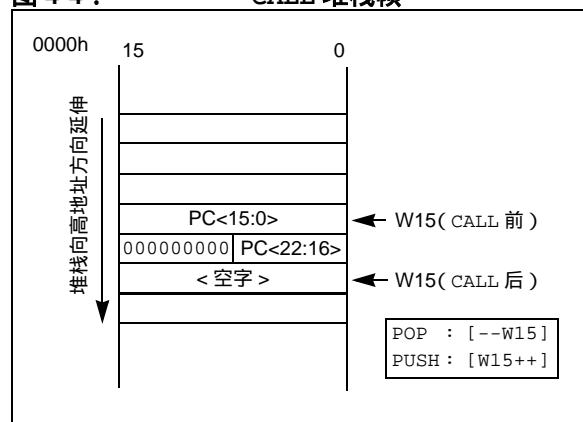
**注：** 在异常处理期间，在 PC 压入堆栈之前，要先将 PC 的 MSB 与 SRL 寄存器组合在一起。

堆栈指针限制值寄存器（SPLIM）与堆栈指针相关联，它设置堆栈上边界的地址。SPLIM 在复位时不会被初始化。与堆栈指针一样，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有效地址时，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。因此，例如如果想要在堆栈超过 RAM 中的地址 2000h 时产生堆栈错误陷阱，则需用值 1FFEh 来初始化 SPLIM。

同理，当堆栈指针地址小于 0080h 时，将产生堆栈指针下溢（堆栈错误）陷阱。这可防止堆栈进入特殊功能寄存器（SFR）空间。

在对 SPLIM 寄存器进行写操作后，不应紧跟使用 W15 进行间接读操作的指令。

图 4-4： CALL 堆栈帧



## 4.3 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新映射到数据空间（程序空间可视性）

表指令允许应用程序读写程序存储器中的一小块区域。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。重映射方式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。这一方式只能访问程序字的低位字。

### 4.3.1 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）定义程序空间中一个 32K 字的区域。TBLPAG 寄存器的 8 位与 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 的最高位用来决定操作是发生在用户存储区（TBLPAG<7> = 0）中还是配置存储区（TBLPAG<7> = 1）中。

对于重映射操作，使用 8 位的程序空间可视性页寄存器（PSVPAG）定义程序空间中的 16K 字页。当 EA 的最高位为 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。

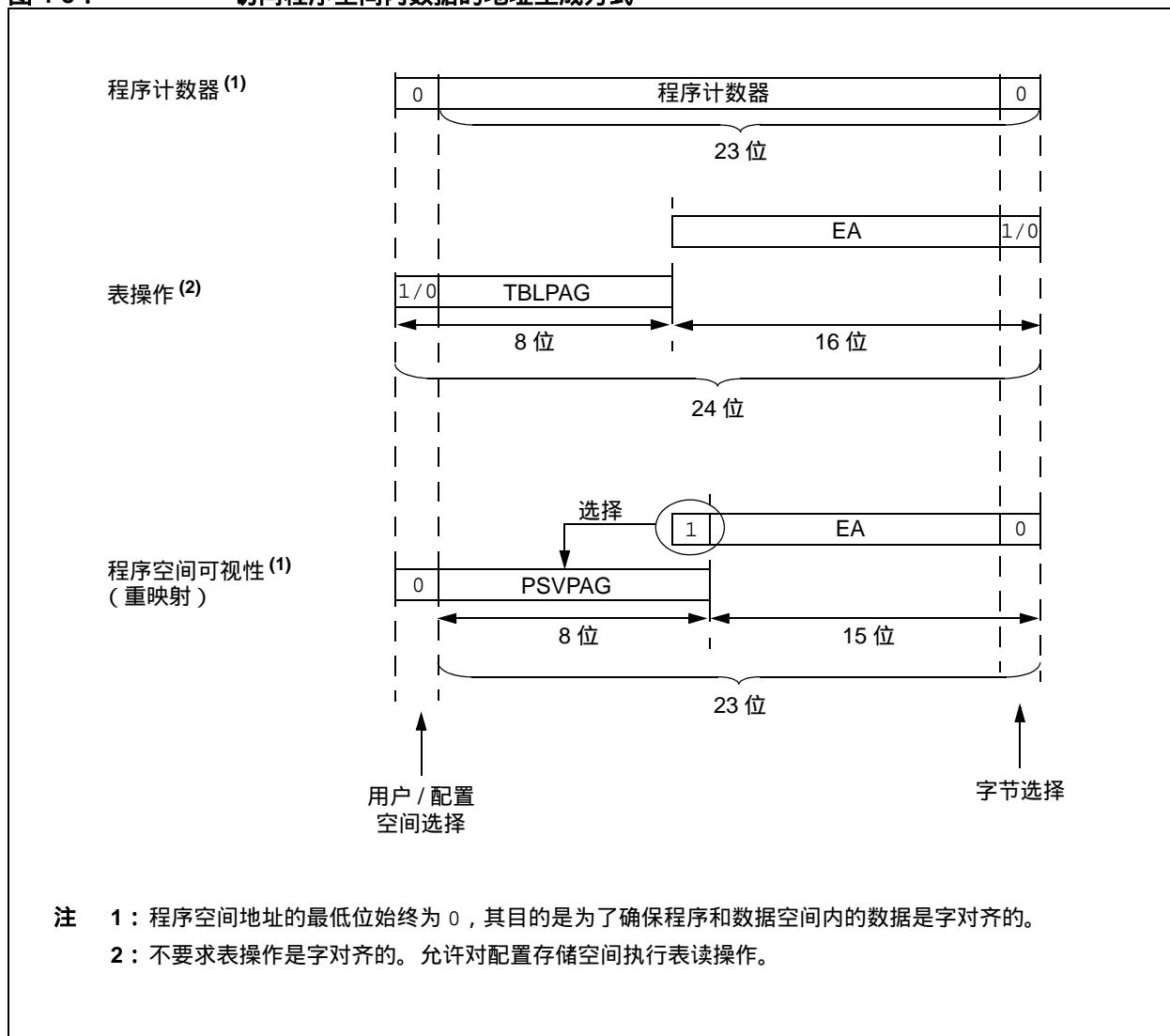
表 4-28 和图 4-5 显示了如何从数据 EA 创建程序 EA 来进行表操作和重映射访问。本文中，P<23:0> 指的是程序空间字，而 D<15:0> 指的是数据空间字。

表 4-28： 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>
通过指令访问 (代码执行)	用户	0	PC<22:1>			0
		0xx xxxx xxxx xxxx xxxx xxxx xxxx 0				
TBLRD/TBLWT (读 / 写字节或字)	用户	TBLPAG<7:0>		数据 EA<15:0>		
		0xxx xxxx		xxxx xxxx xxxx xxxx xxxx		
程序空间可视性 (块重映射 / 读)	配置	TBLPAG<7:0>		数据 EA<15:0>		
		1xxx xxxx		xxxx xxxx xxxx xxxx xxxx		
程序空间可视性 (块重映射 / 读)	用户	0	PSVPAG<7:0>		数据 EA<14:0> <sup>(1)</sup>	
		0	xxxx xxxx	xxx xxxx 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>。

图 4-5： 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始终为 0，其目的是为了确保程序和数据空间内的数据是字对齐的。

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。

## 4.3.2 使用表指令访问程序存储器中的数据

TBLRDL 和 TBLWTL 指令提供了读或写程序空间内任何地址的低位字的直接方法，无需通过数据空间。TBLRDH 和 TBLWTH 指令是可以把一个程序空间字的高 8 位作为数据读写的惟一方法。

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被直接映射到数据空间地址中。于是，程序存储器可以被看作是两个 16 位字宽的地址空间，它们并排放置，具有相同的地址范围。TBLRDL 和 TBLWTL 访问包含数据低位字的空间，TBLRDH 和 TBLWTH 访问包含高数据字节的空间。

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. TBLRDL（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，始终为 0。

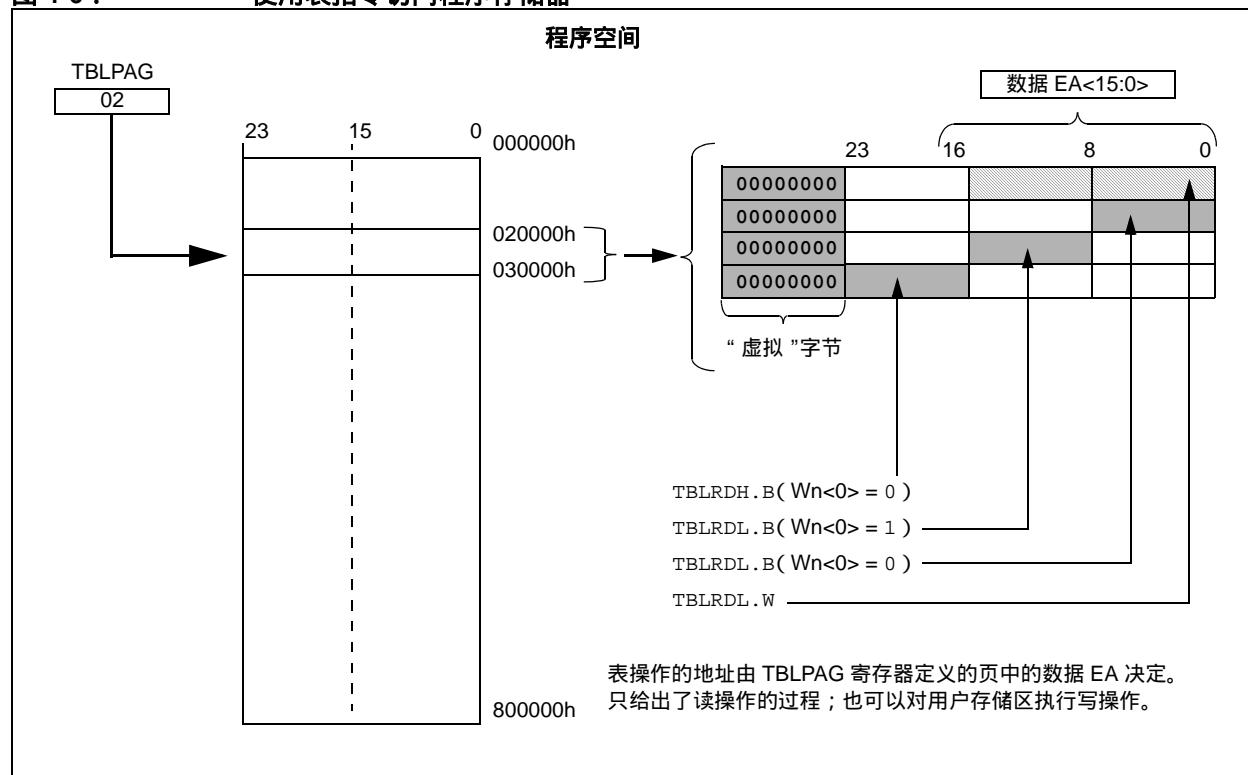
在字节模式下，程序字的高字节或低字节被映射到数据地址的 D<7:0> 中，这和 TBLRDL 指令相同。注意当选择高位“虚拟”字节（字节选择位 = 1）时，数据将始终为 0。

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 5.0 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，包括用户空间和配置空间。当 TBLPAG<7> = 0 时，表页位于用户存储空间中。当 TBLPAG<7> = 1 时，表页位于配置存储空间中。

**注：**仅表读操作可在配置存储空间内执行，且只能在已实现的区域（例如器件 ID）中执行。不允许表写操作。

图 4-6：使用表指令访问程序存储器



### 4.3.3 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 映射到程序空间中的任一 16K 字页中。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 TBLRDI/H 指令）。

如果数据空间 EA 的最高位 (MSb) 为 1，且通过将 CPU 控制寄存器中的 PSV 位 (CORCON<2>) 置 1 使能了程序空间可视性，就可以通过数据空间访问程序空间。要被映射到数据空间的程序存储空间的位置是由程序空间可视性页地址寄存器 (PSVPAG) 决定的。这一 8 位的寄存器定义程序空间中 256 个可能的 16K 字页中的一个。事实上，PSVPAG 作为程序存储器地址的高 8 位，而 EA 的 15 位则作为地址的低位。注意，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。

执行将数据读入该区域的指令，需要一个额外的指令周期，因为这类指令需要对程序存储器执行两次数据取操作。

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 4-7），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 位都应当被设置为 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：** 在执行表读 / 写操作时，PSV 访问被暂时禁止。

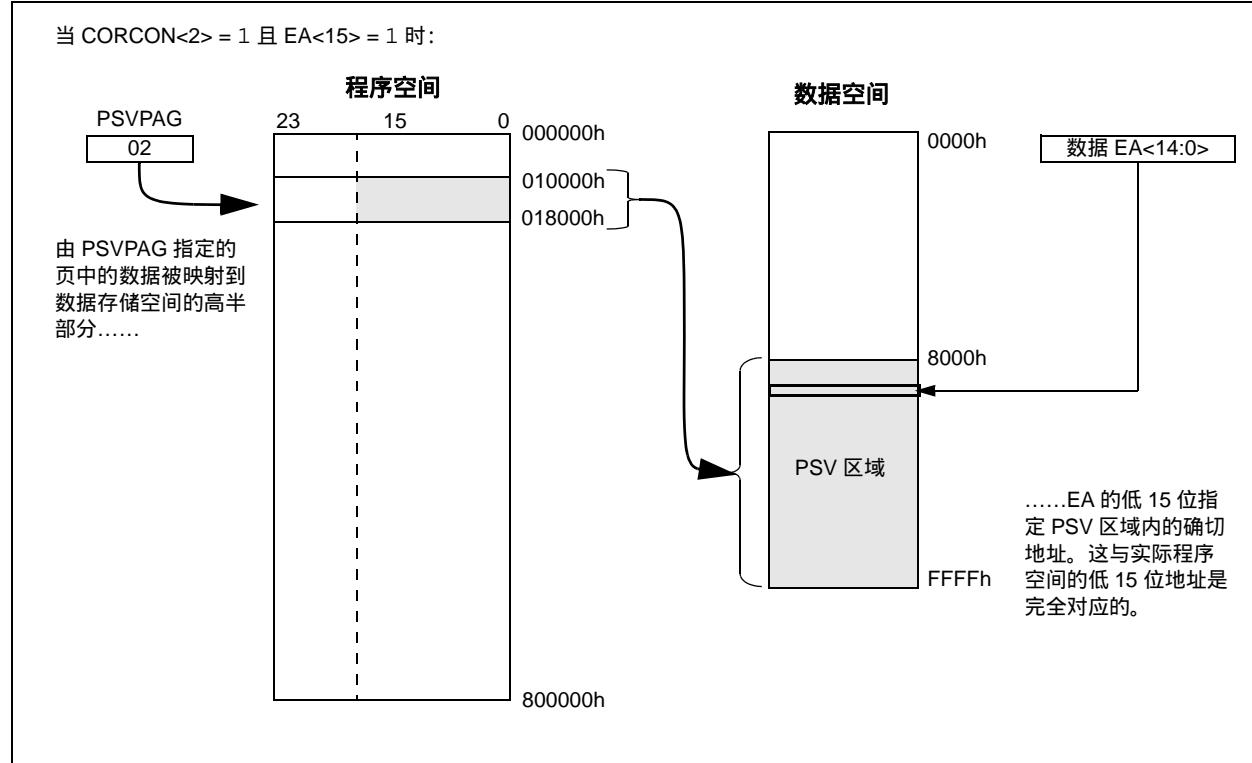
对于使用 PSV 而又在 REPEAT 循环之外执行的指令，MOV 和 MOV.D 指令除了规定的执行时间之外，还需要一个额外的指令周期。其他所有的指令，除了规定的指令执行时间之外，需要两个额外的指令周期。

对于使用 PSV 而又在 REPEAT 循环内执行的指令，下列情况，除了规定的指令执行时间之外，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 4-7： 程序空间可视性操作**



# **PIC24FJ64GB004 系列**

---

---

注：

## 5.0 闪存程序存储器

**注：**本数据手册总结了 PIC24F 系列器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 4 章“程序存储器”(DS39715A\_CN)。

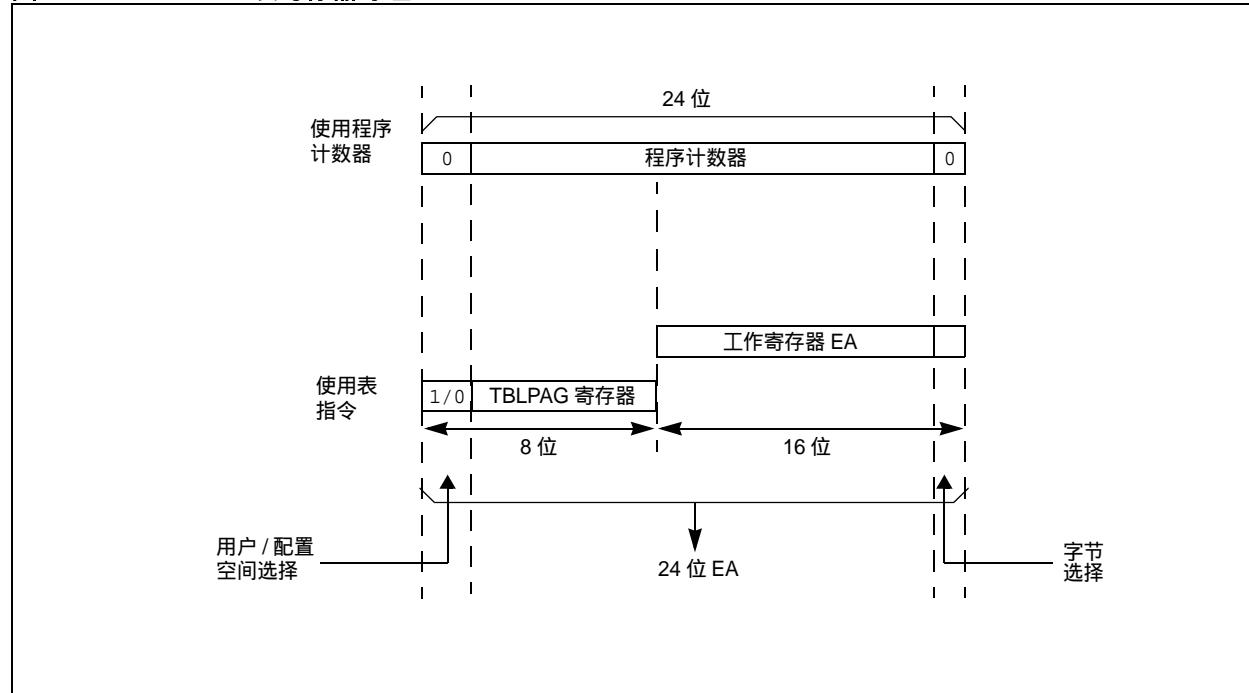
PIC24FJ64GB004 系列器件包含内部闪存程序存储器，用于存储和执行应用代码。当工作电压 VDD 大于 2.35V 时，可对该存储器进行读写和擦除操作。(如果禁止了稳压器，则 VDDCORE 必须大于 2.25V。)

闪存有以下四种编程方式：

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

ICSP 允许在最终应用电路中对 PIC24FJ64GB004 系列器件进行串行编程。只需五根线就可以完成编程，它们分别是编程时钟线 (PGECx)、编程数据线 (PGEDx)、电源线 (VDD)、接地线 (Vss) 和主复位线 (MCLR)。这允许用户使用未编程器件制造电路板，仅在产品交付前才对单片机进行编程。从而可以将最新版本的固件或定制固件烧写到单片机中。

图 5-1：表寄存器寻址



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

## 5.1 表指令和闪存编程

闪存的所有编程都是通过表读和表写指令完成的，与使用的编程方法无关。这些指令允许在器件正常工作模式下通过数据存储器直接读写程序存储空间。24 位程序存储器目标地址由 TBLPAG<7:0> 位和表指令中指定的 W 寄存器中的有效地址 (EA) 组成，如图 5-1 所示。

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

TBLRDH 和 TBLWTH 指令用于读或写程序存储器的 bit<23:16>。TBLRDH 和 TBLWTH 指令也能以字或字节模式访问程序存储器。

## 5.2 RTSP 操作

PIC24F 闪存程序存储器阵列以 64 条指令或 192 字节的行为单位构成。RTSP 允许用户一次擦除 8 行 ( 512 条指令 ) 的块并一次编程一行。它还可以编程单字。

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

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

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

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

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

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

**注：** 不推荐多次写入一个存储单元而不擦除其内容。

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

## 5.3 JTAG 操作

PIC24F 系列支持 JTAG 编程和边界扫描。边界扫描可以通过验证引脚到 PCB 的连通性改进制造工艺。编程是通过支持串行向量格式 ( Serial Vector Format , SVF ) 的工业标准 JTAG 编程器来执行的。

## 5.4 增强型在线串行编程

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

## 5.5 控制寄存器

有两个用于读写闪存程序存储器的 SFR : NVMCON 和 NVMKEY。

NVMCON 寄存器 ( 寄存器 5-1 ) 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时间。

NVMKEY 是只写寄存器，用于写保护。要启动编程或擦除过程，用户必须把 55h 和 AAh 连续写入 NVMKEY 寄存器。更多详细信息，请参见第 5.6 节 “ 编程操作 ” 。

## 5.6 编程操作

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

## 寄存器 5-1： NVMCON : 闪存控制寄存器

R/SO-0, HC <sup>(1)</sup>	R/W-0 <sup>(1)</sup>	R/W-0, HS <sup>(1)</sup>	U-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	—	—	—	—	—
bit 15	bit 8						

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

<b>图注：</b>	SO = 只可置 1 位	HC = 可由硬件清零的位	HS = 可由硬件置 1 的位
R = 可读位	W = 可写位	U = 未实现位，读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

- bit 15      **WR** : 写控制位<sup>(1)</sup>  
               1 = 启动闪存编程或擦除操作。操作是自定时的，一旦操作完成此位即由硬件清零。  
               0 = 编程或擦除操作完成，并处于停止状态
- bit 14      **WREN** : 使能位<sup>(1)</sup>  
               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<sup>(1)</sup>  
               1 = 尝试执行错误的编程或擦除序列或发生终止（任何试图将 WR 位置 1 的操作都会自动置 1 此位）  
               0 = 编程或擦除操作正常完成
- bit 12-7     未实现：读为 0
- bit 6        **ERASE** : 擦除 / 编程使能位<sup>(1)</sup>  
               1 = 在下一个写命令执行由 NVMOP<3:0> 指定的擦除操作  
               0 = 在下一个写命令执行由 NVMOP<3:0> 指定的编程操作
- bit 5-4      未实现：读为 0
- bit 3-0      **NVMOP<3:0>** : NVM 操作选择位<sup>(1,2)</sup>  
               1111 = 存储器批量擦除操作 (ERASE = 1) 或无操作 (ERASE = 0)<sup>(3)</sup>  
               0011 = 存储器字编程操作 (ERASE = 0) 或无操作 (ERASE = 1)  
               0010 = 存储器页擦除操作 (ERASE = 1) 或无操作 (ERASE = 0)  
               0001 = 存储器行编程操作 (ERASE = 0) 或无操作 (ERASE = 1)

- 注**   1：只能在上电复位时复位这些位。  
       2：NVMOP<3:0> 的所有其他组合均未实现。  
       3：仅在 ICSP™ 模式下可用。请参见器件编程规范。

## 5.6.1 闪存程序存储器编程算法

用户一次可编程闪存程序存储器的一行。要实现此操作，必须擦除包含该行在内的一个 8 行擦除块。一般步骤如下：

1. 读程序存储器中的 8 行（512 条指令），并将其存储在数据 RAM 中。
2. 用需要的新数据更新 RAM 中对应的程序数据。
3. 擦除块（见例 5-1）：
  - a) 将 NVMOP 位（NVMCON<3:0>）设置为 0010，以配置为块擦除操作。将 ERASE（NVMCON<6>）和 WREN（NVMCON<14>）位置 1。
  - b) 将要擦除块的起始地址写入 TBLPAG 和 W 寄存器。
  - c) 将 55h 写入 NVMKEY。
  - d) 将 AAh 写入 NVMKEY。
  - e) 将 WR 位（NVMCON<15>）置 1。启动擦除周期，CPU 停止工作等待擦除周期结束。擦除操作完成时，WR 位自动清零。

4. 将数据 RAM 的前 64 条指令写入程序存储器缓冲器（见例 5-1）。

5. 将程序块写入闪存：

- a) 将 NVMOP 位设置为 0001，以配置为行编程操作。将 ERASE 位清零，将 WREN 位置 1。
- b) 将 55h 写入 NVMKEY。
- c) 将 AAh 写入 NVMKEY。
- d) 将 WR 位置 1。启动编程周期，CPU 停止工作等待写周期完成。当写闪存的操作完成时，WR 位自动清零。

6. 将 TBLPAG 的值递增 1，使用数据 RAM 中下一个 64 条指令的块重复步骤 4 和 5，直到将所有 512 条指令写回闪存。

为了防止意外操作，必须向 NVMKEY 写入启动序列从而允许执行任何擦除或编程操作。执行了编程命令以后，用户必须等待一定的编程时间，直到编程操作完成。编程启动序列后紧跟的两条指令必须为 NOP，如例 5-5 所示。

## 例 5-1：擦除程序存储器块——汇编语言代码

```
; Set up NVMCON for block erase operation
    MOV    #0x4042, W0
    MOV    W0, NVMCON
;
; Initialize NVMCON

; Init pointer to row to be ERASED
    MOV    #tblpage(PROG_ADDR), W0
    MOV    W0, TBLPAG
    MOV    #tbloffset(PROG_ADDR), W0
    TBLWTL W0, [W0]
    DISI    #5
;
; Initialize PM Page Boundary SFR
; Initialize in-page EA[15:0] pointer
; Set base address of erase block
; Block all interrupts with priority <7
; for next 5 instructions

    MOV    #0x55, W0
    MOV    W0, NVMKEY
    MOV    #0xAA, W1
    MOV    W1, NVMKEY
    BSET   NVMCON, #WR
    NOP
    NOP
;
; Write the 55 key
;
; Write the AA key
;
; Start the erase sequence
; Insert two NOPs after the erase
; command is asserted
```

## 例 5-2 : 擦除程序存储器块——“C”语言代码

```
// C example using MPLAB C30
unsigned long progAddr = 0xFFFFFFFF;           // Address of row to write
unsigned int offset;

//Set up pointer to the first memory location to be written
TBLPAG = progAddr>>16;                         // 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 = 0x4042;                                 // Initialize NVMCON

asm("DISI #5");                                // Block all interrupts with priority <7
                                                // for next 5 instructions
__builtin_write_NVM();                           // C30 function to perform unlock
                                                // sequence and set WR
```

## 例 5-3 : 装载写缓冲器——汇编语言代码

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

# PIC24FJ64GB004 系列

## 例 5-4：装载写缓冲器——“C”语言代码

```
// C example using MPLAB C30

#define NUM_INSTRUCTION_PER_ROW 64
unsigned int offset;
unsigned int i;
unsigned long progAddr = 0xFFFFFFFF; // Address of row to write
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 = progAddr>>16; // 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 5-5：启动编程序列——汇编语言代码

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

## 例 5-6：启动编程序列——“C”语言代码

```
// C example using MPLAB C30

asm("DISI #5"); // Block all interrupts with priority < 7
; for next 5 instructions

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

## 5.6.2 编程闪存程序存储器的一个字

若已擦除了一个闪存单元，则可用表写指令对此单元进行编程以将一个指令字（24位）写入写锁存器。将闪存地址的8个最高字节装入 TBLPAG 寄存器。TBLWTL 和 TBLWTH 指令将所需数据写入写锁存器，并指定要写

入的程序存储器地址的低16位。要将 NVMCON 寄存器配置为字写操作，应将 NVMOP 位（NVMCON<3:0>）设置为 0011。通过执行解锁序列并将 WR 位置1来执行此写操作（见例 5-7）。

### 例 5-7：编程闪存程序存储器的一个字——汇编语言代码

```
; 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, W2              ;
MOV    #HIGH_BYT, 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
NOP                           ; Insert two NOPs after the erase
NOP                           ; Command is asserted
```

### 例 5-8：编程闪存程序存储器的一个字——“C”语言代码

```
// C example using MPLAB C30

unsigned int offset;
unsigned long progAddr = 0xFFFFFFFF;           // Address of word to program
unsigned int progDataL = 0xFFFF;                 // Data to program lower word
unsigned char progDataH = 0xFF;                  // Data to program upper byte

//Set up NVMCON for word programming
NVMCON = 0x4003;                                // Initialize NVMCON

//Set up pointer to the first memory location to be written
TBLPAG = progAddr>>16;                          // Initialize PM Page Boundary SFR
offset = progAddr & 0xFFFF;                        // Initialize lower word of address

//Perform TBLWT instructions to write latches
__builtin_tblwtl(offset, progDataL);             // Write to address low word
__builtin_tblwth(offset, progDataH);             // Write to upper byte
asm("DISI #5");                                 // Block interrupts with priority < 7
                                                // for next 5 instructions
__builtin_write_NVM();                           // C30 function to perform unlock
                                                // sequence and set WR
```

# **PIC24FJ64GB004 系列**

---

---

注：

## 6.0 复位

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 7 章“复位”(DS39712A\_CN)。

复位模块将所有的复位源组合在一起并控制器件主复位信号 SYSRST。以下所列是器件的复位源：

- POR：上电复位
- MCLR：引脚复位
- SWR：RESET 指令
- WDT：看门狗定时器复位
- BOR：欠压复位
- CM：配置失配复位
- 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

复位模块的简化框图如图 6-1 所示。

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

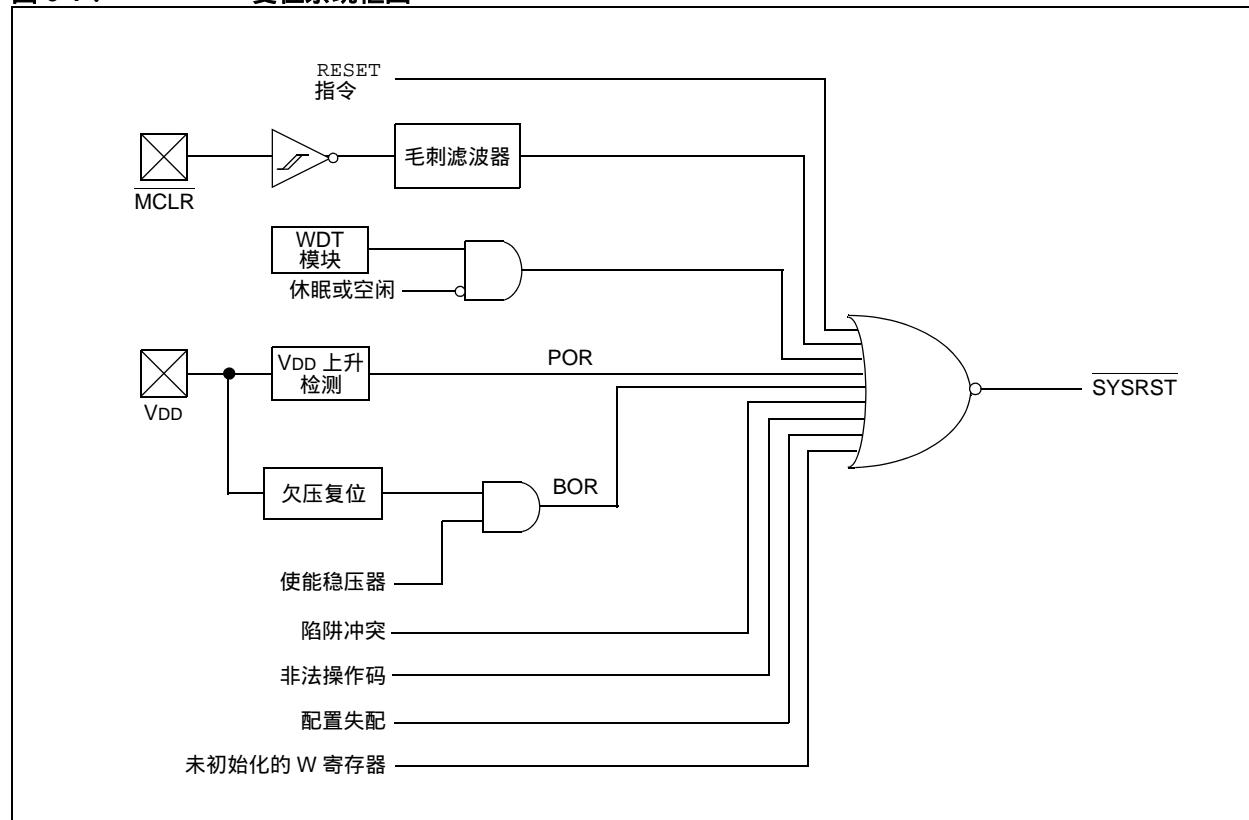
**注：**有关寄存器复位状态的信息，请参见本手册中的特定外设或 CPU 章节。

任何类型的器件复位都会将 RCON 寄存器中相应的位置 1 以表明复位类型（见寄存器 6-1）。上电复位会清零除 BOR 和 POR 位 (RCON<1:0>) 外的所有位，而 BOR 和 POR 位会被置 1。用户可在代码执行过程中的任何时间置 1 或清零任何位。RCON 位仅作为状态位。用软件将特定的复位状态位置 1 不会导致器件发生复位。

RCON 寄存器也有一些与看门狗定时器和器件节能状态相关的位。本数据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在被读取后清零，这样下一次器件复位后的 RCON 寄存器值才会有意义。

图 6-1：复位系统框图



# PIC24FJ64GB004 系列

寄存器 6-1 : RCON : 复位控制寄存器<sup>(1)</sup>

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

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

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

- bit 15      **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 寄存器的复位标志位  
1 = 检测到非法操作码、非法地址模式或未初始化的 W 寄存器用作地址指针而导致的复位  
0 = 未发生由非法操作码或未初始化的 W 寄存器而导致的复位
- bit 13-11    未实现 : 读为 0
- bit 10      **DPSLP** : 深度休眠模式标志位  
1 = 发生深度休眠  
0 = 未发生深度休眠
- bit 9        **CM** : 配置字失配复位标志位  
1 = 发生了配置字失配复位  
0 = 未发生配置字失配复位
- bit 8        **PMSLP** : 休眠期间程序存储器上电位  
1 = 休眠期间继续提供程序存储器偏置电压  
0 = 休眠期间程序存储器偏置电压掉电且稳压器进入待机模式
- 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<sup>(2)</sup>  
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 = 器件未处于休眠模式
- bit 2        **IDLE** : 从空闲模式唤醒标志位  
1 = 器件处于空闲模式  
0 = 器件未处于空闲模式

注 1 : 所有复位状态位都可以用软件置 1 或清零。用软件将任何一位置 1 不会导致器件复位。

2 : 如果 FWDTEN 配置位为 1 (未编程), 则 WDT 总是使能, 而与 SWDTEN 位的设置无关。

## 寄存器 6-1 : RCON : 复位控制寄存器<sup>(1)</sup> (续)

bit 1	<b>BOR</b> : 欠压复位标志位
	1 = 发生了欠压复位。注意 BOR 在上电复位后也被置 1。 0 = 未发生欠压复位
bit 0	<b>POR</b> : 上电复位标志位
	1 = 发生了上电复位 0 = 未发生上电复位

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

表 6-1 : 复位标志位操作

标志位	置 1 事件	清零事件
TRAPR ( RCON<15> )	陷阱冲突事件	POR
IOPUWR ( RCON<14> )	非法操作码或访问未初始化的 W 寄存器	POR
CM ( RCON<9> )	配置失配复位	POR
EXTR ( RCON<7> )	MCLR 复位	POR
SWR ( RCON<6> )	RESET 指令	POR
WDTO ( RCON<4> )	WDT 超时	PWRSAV 指令和 POR
SLEEP ( RCON<3> )	PWRSAV #SLEEP 指令	POR
IDLE ( RCON<2> )	PWRSAV #IDLE 指令	POR
BOR ( RCON<1> )	POR 和 BOR	—
POR ( RCON<0> )	POR	—
DPSLP ( RCON<10> )	PWRSAV #SLEEP 指令且 DSCON <DSEN> 置 1	POR

注：所有复位标志位可由用户软件置 1 或清零。

## 6.1 复位时的时钟源选择

如果使能了时钟切换，器件复位时的系统时钟源可按照表 6-2 进行选择。如果禁止时钟切换，则总是根据振荡器配置位选择系统时钟源。更多详细信息，请参见第 8.0 节“振荡器配置”。

表 6-2 : 振荡器选择与复位类型的关系  
(使能时钟切换)

复位类型	时钟源的确定
POR	FNOSC 配置位 ( CW2<10:8> )
BOR	
MCLR	COSC 控制位 ( OSCCON<14:12> )
WDTO	
SWR	

## 6.2 器件复位时间

表 6-3 总结了各种类型的器件复位所需的时间。注意在 POR 延时和 PWRT 延时结束后会发出系统复位信号 SYSRST。

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

FSCM 延时决定在 SYSRST 信号发出后 FSCM 开始监视系统时钟源的时间。

# PIC24FJ64GB004 系列

表 6-3：各种器件复位的复位延时

复位类型	时钟源	SYSRST 延时	系统时钟 延时	注
POR <sup>(6)</sup>	EC	TPOR + TRST + TPWRT	—	1, 2, 3
	FRC 和 FRCDIV	TPOR + TRST + TPWRT	TFRC	1, 2, 3, 4
	LPRC	TPOR + TRST + TPWRT	TLPRC	1, 2, 3, 4
	ECPLL	TPOR + TRST + TPWRT	TLOCK	1, 2, 3, 5
	FRCPLL	TPOR + TRST + TPWRT	TFRC + TLOCK	1, 2, 3, 4, 5
	XT、HS 和 SOSC	TPOR+ TRST + TPWRT	TOST	1, 2, 3, 6
	XTPLL 和 HSPLL	TPOR + TRST + TPWRT	TOST + TLOCK	1, 2, 3, 5, 6
BOR	EC	TRST + TPWRT	—	2, 3
	FRC 和 FRCDIV	TRST + TPWRT	TFRC	2, 3, 4
	LPRC	TRST + TPWRT	TLPRC	2, 3, 4
	ECPLL	TRST + TPWRT	TLOCK	2, 3, 5
	FRCPLL	TRST + TPWRT	TFRC + TLOCK	2, 3, 4, 5
	XT、HS 和 SOSC	TRST + TPWRT	TOST	2, 3, 6
	XTPLL 和 HSPLL	TRST + TPWRT	TFRC + TLOCK	2, 3, 4, 5
所有其他复位	任何时钟	—	—	2

注 1：TPOR = 上电复位延时。

2：TRST = 内部状态复位时间。

3：TPWRT = 标称值为 64 ms (如果将 DISVREG 与 VDD 相连禁止了稳压器)。

4：TFRC 和 TLPRC = RC 振荡器起振时间。

5：TLOCK = PLL 锁定时间。

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

7：如果使能了双速启动，无论选择了哪种主振荡器，器件都将使用 FRC 启动，在这种情况下，需要考虑 FRC 起振时间。

注：如需了解详细的工作频率和时序规范，请参见第 29.0 节“电气特性”。

## 6.2.1 POR 和长振荡器起振时间

振荡器起振电路及其相关的延时定时器与上电时发生的器件复位延时没有关系。某些晶振电路（尤其是低频晶振）的起振时间会相对较长。因此，在发出 SYSRST 之后，可能会发生以下一个或多个情况：

- 振荡器电路尚未起振。
- 振荡器起振定时器尚未超时（如果使用了晶振）。
- PLL 还未锁定（如果使用了 PLL）。

在有效时钟源供系统使用之前，器件不会开始执行代码。因此，如果必须确定复位延时，必须考虑到振荡器和 PLL 起振延时。

## 6.2.2 故障保护时钟监视器（FSCM）和器件复位

如果使能了 FSCM，它将在 SYSRST 发出时开始监视系统时钟源。如果此时没有可用的有效时钟源，器件将会自动切换到 FRC 振荡器，且用户可在陷阱服务程序（Trap Service Routine，TSR）中将系统时钟源切换到所需的晶振。

## 6.3 特殊功能寄存器的复位状态

大部分与 PIC24F CPU 和外设有关的特殊功能寄存器（SFR）会在器件复位时复位为某个特定值。SFR 是按其外设或 CPU 功能分组的，其复位值在本手册的各章节中有所说明。

除了四个寄存器之外，其他 SFR 的复位值都不受复位类型的影响。复位控制寄存器 RCON 的复位值取决于器件复位的类型。振荡器控制寄存器 OSCCON 的复位值取决于复位的类型和闪存配置字寄存器 2（CW2）中 FNOSC 位的编程值（见表 6-2）。RCFGCAL 和 NVMCON 寄存器只受 POR 影响。

## 6.4 深度休眠 BOR（DSBOR）

深度休眠 BOR 是功耗极低的 BOR 电路，在器件处于深度休眠模式时使用。由于消耗的电流较低，精度可能有所变化。

DSBOR 跳变点在 2.0V 左右。可通过配置 CW4<DSBOREN> = 1 使能 DSBOR。如果 VDD 掉至低于 POR 阈值，则 DSBOR 将重新激活 POR 以确保器件复位。

# **PIC24FJ64GB004 系列**

---

---

注：

## 7.0 中断控制器

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 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到一个到 PIC24F CPU 的中断请求信号。该控制器具有以下特性：

- 最多 8 个处理器异常和软件陷阱
- 7 个用户可选择的优先级
- 具有最多 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惟一的向量
- 指定的用户优先级中具有固定优先级
- 用于支持调试的备用中断向量表 (Alternate Interrupt Vector Table, AIVT)
- 固定的中断进入和返回延时

## 7.1 中断向量表

中断向量表 (IVT) 如图 7-1 所示。IVT 位于程序存储器中，起始单元地址是 000004h。IVT 包含 126 个向量，由 8 个不可屏蔽的陷阱向量和多达 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对应的中断服务程序 (ISR) 的起始地址。

中断向量有一个自然优先级；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。

PIC24FJ64GB004 系列器件实现了不可屏蔽陷阱和惟一中断。表 7-1 和表 7-2 对此做了总结。

### 7.1.1 备用中断向量表

如图 7-1 所示，备用中断向量表 (AIVT) 位于 IVT 之后。ALTIVT 控制位 (INTCON2<15>) 控制对 AIVT 的访问。如果 ALTIVT 位置 1，所有中断和异常处理将使用备用向量而不是默认的向量。备用向量与默认向量的组织方式相同。

AIVT 支持仿真和调试功能，它提供了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。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软件算法。如果不需要 AIVT，应使用 IVT 中使用的地址设置 AIVT。

## 7.2 复位过程

由于复位过程中不涉及到中断控制器，所以器件复位并不是真的异常。作为对复位的响应，PIC24F 器件清零其寄存器，同时强制 PC 为零。然后，单片机从单元 000000h 开始执行程序。用户在复位地址中设置一条 GOTO 指令，该指令会使程序执行重新定位到相应的启动程序。

**注：**应该使用包含 RESET 指令的默认中断处理程序的地址编程 IVT 和 AIVT 中所有未实现或未使用的向量存储单元。

# PIC24FJ64GB004 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
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
—	
—	
—	
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	0001FEh
中断向量 117	000200h
代码起始	

中断向量表(IVT)<sup>(1)</sup>

备用中断向量表(AIVT)<sup>(1)</sup>

注 1：有关中断向量列表的信息，请参见表 7-2。

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 : 实现的中断向量

中断源	向量编号	IVT 地址	AVIT 地址	中断位位置		
				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>	IPC16<14:12>
CTMU 事件	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>
I2C2 主事件	50	000078h	000178h	IFS3<2>	IEC3<2>	IPC12<10:8>
I2C2 从事件	49	000076h	000176h	IFS3<1>	IEC3<1>	IPC12<6:4>
输入捕捉 1	1	000016h	000116h	IFS0<1>	IEC0<1>	IPC0<6:4>
输入捕捉 2	5	00001Eh	00011Eh	IFS0<5>	IEC0<5>	IPC1<6:4>
输入捕捉 3	37	00005Eh	00015Eh	IFS2<5>	IEC2<5>	IPC9<6:4>
输入捕捉 4	38	000060h	000160h	IFS2<6>	IEC2<6>	IPC9<10:8>
输入捕捉 5	39	000062h	000162h	IFS2<7>	IEC2<7>	IPC9<14:12>
输入电平变化通知	19	00003Ah	00013Ah	IFS1<3>	IEC1<3>	IPC4<14:12>
LVD 低电压检测	72	0000A4h	0001A4h	IFS4<8>	IEC4<8>	IPC18<2:0>
输出比较 1	2	000018h	000118h	IFS0<2>	IEC0<2>	IPC0<10:8>
输出比较 2	6	000020h	000120h	IFS0<6>	IEC0<6>	IPC1<10:8>
输出比较 3	25	000046h	000146h	IFS1<9>	IEC1<9>	IPC6<6:4>
输出比较 4	26	000048h	000148h	IFS1<10>	IEC1<10>	IPC6<10:8>
输出比较 5	41	000066h	000166h	IFS2<9>	IEC2<9>	IPC10<6:4>
并行主端口	45	00006Eh	00016Eh	IFS2<13>	IEC2<13>	IPC11<6:4>
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>
SPI2 错误	32	000054h	000154h	IFS2<0>	IEC2<0>	IPC8<2:0>
SPI2 事件	33	000056h	000156h	IFS2<1>	IEC2<1>	IPC8<6:4>
Timer1	3	00001Ah	00011Ah	IFS0<3>	IEC0<3>	IPC0<14:12>
Timer2	7	000022h	000122h	IFS0<7>	IEC0<7>	IPC1<14:12>
Timer3	8	000024h	000124h	IFS0<8>	IEC0<8>	IPC2<2:0>
Timer4	27	00004Ah	00014Ah	IFS1<11>	IEC1<11>	IPC6<14:12>
Timer5	28	00004Ch	00014Ch	IFS1<12>	IEC1<12>	IPC7<2:0>
UART1 错误	65	000096h	000196h	IFS4<1>	IEC4<1>	IPC16<6:4>
UART1 接收器	11	00002Ah	00012Ah	IFS0<11>	IEC0<11>	IPC2<14:12>
UART1 发送器	12	00002Ch	00012Ch	IFS0<12>	IEC0<12>	IPC3<2:0>
UART2 错误	66	000098h	000198h	IFS4<2>	IEC4<2>	IPC16<10:8>
UART2 接收器	30	000050h	000150h	IFS1<14>	IEC1<14>	IPC7<10:8>
UART2 发送器	31	000052h	000152h	IFS1<15>	IEC1<15>	IPC7<14:12>
USB 中断	86	0000C0h	0001C0h	IFS5<6>	IEC5<6>	IPC21<10:8>

## 7.3 中断控制和状态寄存器

PIC24FJ64GB004 系列器件实现了用于中断控制器的以下寄存器：

- INTCON1
- INTCON2
- IFS0 至 IFS5
- IEC0 至 IEC5
- IPC0 至 IPC21 (除 IPC13、IPC14 和 IPC17 之外)
- 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和备用中断向量表的使用。

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，且由软件清零。

IECx 寄存器包含所有的中断允许位。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。可以为每个用户中断源分配 8 个优先级之一。

中断源按其在表 7-2 中的向量编号顺序分配给 IFSx、IECx 和 IPCx 寄存器。例如，INT0 (外部中断 0) 表示向量编号为 0，且自然优先级为 0。所以 INTOIF 状态位在 IFS0<0> 中，INTOIE 允许位在 IEC0<0> 中，INT0P<2:0> 优先级位在 IPC0 最前面的位置 (IPC0<2:0>) 中。

尽管两个 CPU 控制寄存器都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>)；这些位用于指示当前 CPU 的中断优先级。用户可以通过写 IPL 位更改 CPU 的当前优先级。

CORCON 寄存器包含 IPL3 位，此位与 IPL<2:0> 位一起表示当前 CPU 优先级。IPL3 是一个只读位，所以用户软件不能屏蔽陷阱事件。

中断控制器具有显示其状态的中断控制器测试寄存器 (INTTREG)。发生中断请求时，其关联的向量编号和新的中断优先级被锁存到 INTTREG 中。

如果一个通用 ISR 用于多个向量 (例如在自举程序应用中使用 ISR 重映射)，则还可以使用此信息来确定具体的中断源。还可以用它来检查在 ISR 中是否有其他中断在等待处理。

在下面各页中的寄存器 7-1 到寄存器 7-35 说明了所有的中断寄存器。

## 寄存器 7-1 : SR : ALU 状态寄存器 (在 CPU 中)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R-0
—	—	—	—	—	—	—	DC <sup>(1)</sup>
bit 15							bit 8

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

### bit 7-5 IPL<2:0> : CPU 中断优先级状态位 <sup>(2,3)</sup>

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 : 关于那些不是专用于中断控制功能的其他位的描述 , 请参见寄存器 3-1。

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

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

## 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	R/W-0	U-0	U-0
—	—	—	—	IPL3 <sup>(2)</sup>	PSV <sup>(1)</sup>	—	—
bit 7							bit 0

### 图注 :

C = 可清零位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

### bit 3 IPL3 : CPU 中断优先级状态位 <sup>(2)</sup>

1 = CPU 中断优先级大于 7

0 = CPU 中断优先级为 7 或更小

注 1 : 关于那些不是专用于中断控制功能的其他位的描述 , 请参见寄存器 3-2。

2 : IPL3 位与 IPL<2:0> 位 ( SR<7:5> ) 共同决定 CPU 的中断优先级。

# PIC24FJ64GB004 系列

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

**图注 :**

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **NSTDIS** : 中断嵌套禁止位

1 = 禁止中断嵌套

0 = 允许中断嵌套

bit 14-5    **未实现** : 读为 0

bit 4        **MATHERR** : 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

## 寄存器 7-4 : INTCON2 : 中断控制寄存器 2

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **ALTIvt** : 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 = 没有执行 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

# PIC24FJ64GB004 系列

## 寄存器 7-5： IFS0：中断标志状态寄存器 0

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

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

### 图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-14 未实现：读为 0
- bit 13 **AD1IF** : A/D 转换完成中断标志状态位  
1 = 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 **OC2IF** : 输出比较通道 2 中断标志状态位  
1 = 产生了中断请求  
0 = 未产生中断请求
- bit 5 **IC2IF** : 输入捕捉通道 2 中断标志状态位  
1 = 产生了中断请求  
0 = 未产生中断请求
- bit 4 未实现：读为 0
- bit 3 **T1IF** : Timer1 中断标志状态位  
1 = 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

## 寄存器 7-6 : IFS1 : 中断标志状态寄存器 1

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

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

### 图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- |         |  |
|---------|--|
| bit 15  | <b>U2TXIF</b> : UART2 发送器中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求 |
| bit 14  | <b>U2RXIF</b> : UART2 接收器中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求 |
| bit 13  | <b>INT2IF</b> : 外部中断 2 标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求     |
| bit 12  | <b>T5IF</b> : Timer5 中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求     |
| bit 11  | <b>T4IF</b> : Timer4 中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求     |
| bit 10  | <b>OC4IF</b> : 输出比较通道 4 中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求  |
| bit 9   | <b>OC3IF</b> : 输出比较通道 3 中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求  |
| bit 8-5 | <b>未实现</b> : 读为 0  |
| bit 4   | <b>INT1IF</b> : 外部中断 1 标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求     |
| bit 3   | <b>CNIF</b> : 输入电平变化通知中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求    |
| bit 2   | <b>CMIF</b> : 比较器中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求         |
| bit 1   | <b>MI2C1IF</b> : I2C1 主事件中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求 |
| bit 0   | <b>SI2C1IF</b> : I2C1 从事件中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求 |

# PIC24FJ64GB004 系列

## 寄存器 7-7 : IFS2 : 中断标志状态寄存器 2

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-14 未实现 : 读为 0
- bit 13 PMPIF : 并行主端口中断标志状态位  
1 = 产生了中断请求  
0 = 未产生中断请求
- bit 12-10 未实现 : 读为 0
- bit 9 OC5IF : 输出比较通道 5 中断标志状态位  
1 = 产生了中断请求  
0 = 未产生中断请求
- bit 8 未实现 : 读为 0
- bit 7 IC5IF : 输入捕捉通道 5 中断标志状态位  
1 = 产生了中断请求  
0 = 未产生中断请求
- bit 6 IC4IF : 输入捕捉通道 4 中断标志状态位  
1 = 产生了中断请求  
0 = 未产生中断请求
- bit 5 IC3IF : 输入捕捉通道 3 中断标志状态位  
1 = 产生了中断请求  
0 = 未产生中断请求
- bit 4-2 未实现 : 读为 0
- bit 1 SPI2IF : SPI2 事件中断标志状态位  
1 = 产生了中断请求  
0 = 未产生中断请求
- bit 0 SPF2IF : SPI2 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

## 寄存器 7-8 : IFS3 : 中断标志状态寄存器 3

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

U-0	U-0	U-0	U-0	U-0	R/W-0,	R/W-0	U-0
—	—	—	—	—	MI2C2IF	SI2C2IF	—
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- |          |  |
|----------|--|
| bit 15   | 未实现 : 读为 0   |
| bit 14   | <b>RTCIF</b> : 实时时钟 / 日历中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求  |
| bit 13-3 | 未实现 : 读为 0   |
| bit 2    | <b>MI2C2IF</b> : I2C2 主事件中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求 |
| bit 1    | <b>SI2C2IF</b> : I2C2 从事件中断标志状态位<br>1 = 产生了中断请求<br>0 = 未产生中断请求 |
| bit 0    | 未实现 : 读为 0   |

# PIC24FJ64GB004 系列

寄存器 7-9 : IFS4 : 中断标志状态寄存器 4

U-0	U-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
—	—	CTMUIF	—	—	—	—	LVDIF
bit 15	bit 8						

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

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-14 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 LVDIF : 低电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 7-10 : IFS5 : 中断标志状态寄存器 5

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

U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	USB1IF	—	—	—	—	—	—
bit 7							bit 0

### 图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-7 未实现：读为 0

bit 6 USB1IF : USB1 (USB OTG) 中断标志状态位

1 = 产生了中断请求

0 = 未产生中断请求

bit 5-0 未实现：读为 0

# PIC24FJ64GB004 系列

寄存器 7-11 : IEC0 : 中断允许控制寄存器 0

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

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

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-14 未实现 : 读为 0
- bit 13 **AD1IE** : A/D 转换完成中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 12 **U1TXIE** : UART1 发送器中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 11 **U1RXIE** : UART1 接收器中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 10 **SPI1IE** : SPI1 传输完成中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 9 **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 **OC2IE** : 输出比较通道 2 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 5 **IC2IE** : 输入捕捉通道 2 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 4 未实现 : 读为 0
- bit 3 **T1IE** : Timer1 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 2 **OC1IE** : 输出比较通道 1 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 1 **IC1IE** : 输入捕捉通道 1 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 0 **INT0IE** : 外部中断 0 允许位  
1 = 允许中断请求  
0 = 禁止中断请求

## 寄存器 7-12： IEC1：中断允许控制寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0
U2TXIE	U2RXIE	INT2IE <sup>(1)</sup>	T5IE	T4IE	OC4IE	OC3IE	—
bit 15	bit 8						

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT1IE <sup>(1)</sup>	CNIE	CMIE	MI2C1IE	SI2C1IE
bit 7	bit 0						

### 图注：

R = 可读位

-n = 上电复位时的值

W = 可写位

1 = 置 1

U = 未实现位，读为 0

0 = 清零

x = 未知

bit 15	<b>U2TXIE</b> : 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	<b>U2RXIE</b> : 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	<b>INT2IE</b> : 外部中断 2 允许位 <sup>(1)</sup> 1 = 允许中断请求 0 = 禁止中断请求
bit 12	<b>T5IE</b> : Timer5 中断允许位 1 = 允许中断请求 0 = 禁止中断请求
bit 11	<b>T4IE</b> : Timer4 中断允许位 1 = 允许中断请求 0 = 禁止中断请求
bit 10	<b>OC4IE</b> : 输出比较通道 4 中断允许位 1 = 允许中断请求 0 = 禁止中断请求
bit 9	<b>OC3IE</b> : 输出比较通道 3 中断允许位 1 = 允许中断请求 0 = 禁止中断请求
bit 8-5	未实现：读为 0
bit 4	<b>INT1IE</b> : 外部中断 1 允许位 <sup>(1)</sup> 1 = 允许中断请求 0 = 禁止中断请求
bit 3	<b>CNIE</b> : 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	<b>CMIE</b> : 比较器中断允许位 1 = 允许中断请求 0 = 禁止中断请求
bit 1	<b>MI2C1IE</b> : I2C1 主事件中断允许位 1 = 允许中断请求 0 = 禁止中断请求
bit 0	<b>SI2C1IE</b> : I2C1 从事件中断允许位 1 = 允许中断请求 0 = 禁止中断请求

注 1： 如果允许了外部中断，则此中断输入必须配置给可用的 RPn 或 PRlx 引脚。更多信息，请参见第 10.4 节“外设引脚选择（PPS）”。

# PIC24FJ64GB004 系列

## 寄存器 7-13 : IEC2 : 中断允许控制寄存器 2

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-14 未实现 : 读为 0
- bit 13 PMPIE : 并行主端口中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 12-10 未实现 : 读为 0
- bit 9 OC5IE : 输出比较通道 5 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 8 未实现 : 读为 0
- bit 7 IC5IE : 输入捕捉通道 5 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 6 IC4IE : 输入捕捉通道 4 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 5 IC3IE : 输入捕捉通道 3 中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 4-2 未实现 : 读为 0
- bit 1 SPI2IE : SPI2 事件中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求
- bit 0 SPF2IE : SPI2 故障中断允许位  
1 = 允许中断请求  
0 = 禁止中断请求

## 寄存器 7-14 : 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	R/W-0	R/W-0	U-0
—	—	—	—	—	MI2C2IE	SI2C2IE	—
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现 : 读为 0

bit 14 RTCIE : 实时时钟 / 日历中断允许位

1 = 允许中断请求

0 = 禁止中断请求

bit 13-3 未实现 : 读为 0

bit 2 MI2C2IE : I2C2 主事件中断允许位

1 = 允许中断请求

0 = 禁止中断请求

bit 1 SI2C2IE : I2C2 从事件中断允许位

1 = 允许中断请求

0 = 禁止中断请求

bit 0 未实现 : 读为 0

# PIC24FJ64GB004 系列

寄存器 7-15 : 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	—	—	—	—	LVDIE
bit 15	bit 8						

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

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-14 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 **LVDIE** : 低电压检测中断允许位

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 7-16 : IEC5 : 中断允许控制寄存器 5

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

U-0	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0
—	USB1IE	—	—	—	—	—	—
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-7 未实现 : 读为 0

bit 6 USB1IE : USB1 ( USB OTG ) 中断允许位

1 = 允许中断请求

0 = 禁止中断请求

bit 5-0 未实现 : 读为 0

# PIC24FJ64GB004 系列

## 寄存器 7-17 : IPC0 : 中断优先级控制寄存器 0

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

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

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现：读为 0

bit 14-12 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

## 寄存器 7-18 : IPC1 : 中断优先级控制寄存器 1

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现 : 读为 0

bit 14-12 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 未实现 : 读为 0

bit 10-8 OC2IP<2:0> : 输出比较通道 2 中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 7 未实现 : 读为 0

bit 6-4 IC2IP<2:0> : 输入捕捉通道 2 中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 3-0 未实现 : 读为 0

# PIC24FJ64GB004 系列

寄存器 7-19 : IPC2 : 中断优先级控制寄存器 2

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

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

## 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现 : 读为 0

bit 14-12 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

## 寄存器 7-20 : IPC3 : 中断优先级控制寄存器 3

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-7 未实现 : 读为 0

bit 6-4 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

# PIC24FJ64GB004 系列

寄存器 7-21 : IPC4 : 中断优先级控制寄存器 4

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

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

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现 : 读为 0

bit 14-12 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 MI2C1IP<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 SI2C1IP<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

## 寄存器 7-22 : IPC5 : 中断优先级控制寄存器 5

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-3 未实现 : 读为 0

bit 2-0 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

# PIC24FJ64GB004 系列

寄存器 7-23 : IPC6 : 中断优先级控制寄存器 6

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

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

## 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15 未实现 : 读为 0

bit 14-12 T4IP<2:0> : Timer4 中断优先级位

111 = 中断优先级为 7 ( 最高优先级中断 )

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 11 未实现 : 读为 0

bit 10-8 OC4IP<2:0> : 输出比较通道 4 中断优先级位

111 = 中断优先级为 7 ( 最高优先级中断 )

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 7 未实现 : 读为 0

bit 6-4 OC3IP<2:0> : 输出比较通道 3 中断优先级位

111 = 中断优先级为 7 ( 最高优先级中断 )

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 3-0 未实现 : 读为 0

## 寄存器 7-24 : IPC7 : 中断优先级控制寄存器 7

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15      未实现 : 读为 0
- bit 14-12    **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        未实现 : 读为 0
- bit 2-0      **T5IP<2:0>** : Timer5 中断优先级位  
 111 = 中断优先级为 7 (最高优先级中断)  
 •  
 •  
 •  
 001 = 中断优先级为 1  
 000 = 禁止中断源

# PIC24FJ64GB004 系列

寄存器 7-25 : IPC8 : 中断优先级控制寄存器 8

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

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

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-7 未实现 : 读为 0

bit 6-4 SPI2IP<2:0> : SPI2 事件中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 3 未实现 : 读为 0

bit 2-0 SPF2IP<2:0> : SPI2 故障中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

## 寄存器 7-26 : IPC9 : 中断优先级控制寄存器 9

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现 : 读为 0

bit 14-12 IC5IP<2:0> : 输入捕捉通道 5 中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 11 未实现 : 读为 0

bit 10-8 IC4IP<2:0> : 输入捕捉通道 4 中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 7 未实现 : 读为 0

bit 6-4 IC3IP<2:0> : 输入捕捉通道 3 中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 3-0 未实现 : 读为 0

# PIC24FJ64GB004 系列

寄存器 7-27 : IPC10 : 中断优先级控制寄存器 10

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

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

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-7 未实现 : 读为 0

bit 6-4 OC5IP<2:0> : 输出比较通道 5 中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 3-0 未实现 : 读为 0

## 寄存器 7-28 : IPC11 : 中断优先级控制寄存器 11

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-7 未实现 : 读为 0

bit 6-4 PMPIP<2:0> : 并行主端口中断优先级位

111 = 中断优先级为 7 (最高优先级中断)

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 3-0 未实现 : 读为 0

# PIC24FJ64GB004 系列

寄存器 7-29 : IPC12 : 中断优先级控制寄存器 12

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

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

## 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-11 未实现 : 读为 0

bit 10-8 MI2C2IP<2:0> : I2C2 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 7 未实现 : 读为 0

bit 6-4 SI2C2IP<2:0> : I2C2 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

bit 3-0 未实现 : 读为 0

## 寄存器 7-30 : IPC15 : 中断优先级控制寄存器 15

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

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-11 未实现 : 读为 0

bit 10-8 RTCIP<2:0> : 实时时钟 / 日历中断优先级位

111 = 中断优先级为 7 ( 最高优先级中断 )

•

•

•

001 = 中断优先级为 1

000 = 禁止中断源

bit 7-0 未实现 : 读为 0

# PIC24FJ64GB004 系列

寄存器 7-31 : IPC16 : 中断优先级控制寄存器 16

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

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

## 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现 : 读为 0

bit 14-12 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

## 寄存器 7-32 : 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
—	—	—	—	—	LVDIP2	LVDIP1	LVDIP0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-3 未实现 : 读为 0

bit 2-0 LVDIP<2:0> : 低电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 7-33 : 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	CTMUIPO	—	—	—	—
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-7 未实现 : 读为 0

bit 6-4 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

# PIC24FJ64GB004 系列

寄存器 7-34 : IPC21 : 中断优先级控制寄存器 21

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	USB1IP2	USB1IP1	USB1IP0
bit 15	bit 8						

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

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-11 未实现 : 读为 0

bit 10-8 USB1IP<2:0> : USB 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 7-35 : 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	—	ILR3	ILR2	ILR1	ILR0
bit 15	bit 8						

U-0	R-0						
—	VECNUM6	VECNUM5	VECNUM4	VECNUM3	VECNUM2	VECNUM1	VECNUM0
bit 7	bit 0						

### 图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15      **CPUIRQ** : 来自中断控制器 CPU 的中断请求位  
               1 = 产生了中断请求，但 CPU 尚未应答。这发生在 CPU 的优先级高于所请求的中断优先级的情况下  
               0 = 没有未应答的中断请求
- bit 14      未实现：读为 0
- bit 13      **VHOLD** : 向量编号捕捉配置位  
               1 = VECNUM 位保存优先级最高的待处理中断的值  
               0 = VECNUM 位保存上一次应答的中断值（已发生且优先级高于 CPU 的上一个中断，即使有其他待处理中断时也是如此）
- 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 ID 位（待处理中断向量编号为 VECNUM + 8）  
               011111 = 待处理中断向量编号为 135  
               .  
               .  
               .  
               000001 = 待处理中断向量编号为 9  
               000000 = 待处理中断向量编号为 8

## 7.4 中断设置过程

### 7.4.1 初始化

按以下步骤配置中断源：

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 寄存器中的控制位来为中断源选择用户分配的优先级。优先级将取决于具体的应用和中断源的类型。如果不需要多个优先级，可以将所有允许的中断源的 IPCx 寄存器控制位编程为相同的非零值。
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。

**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。

### 7.4.2 中断服务程序

用于声明 ISR 以及使用正确的向量地址初始化 IVT 的方法取决于编程语言（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套件。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 ISR 后会立即再次进入 ISR。如果 ISR 用汇编语言编码，则必须使用 RETFIE 指令结束 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

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

### 7.4.4 禁止中断

可以通过以下步骤禁止所有用户中断：

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

要允许用户中断，可以使用 POP 指令恢复先前的 SR 值。

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

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

## 8.0 振荡器配置

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 6 章“振荡器”(DS39700A\_CN)。

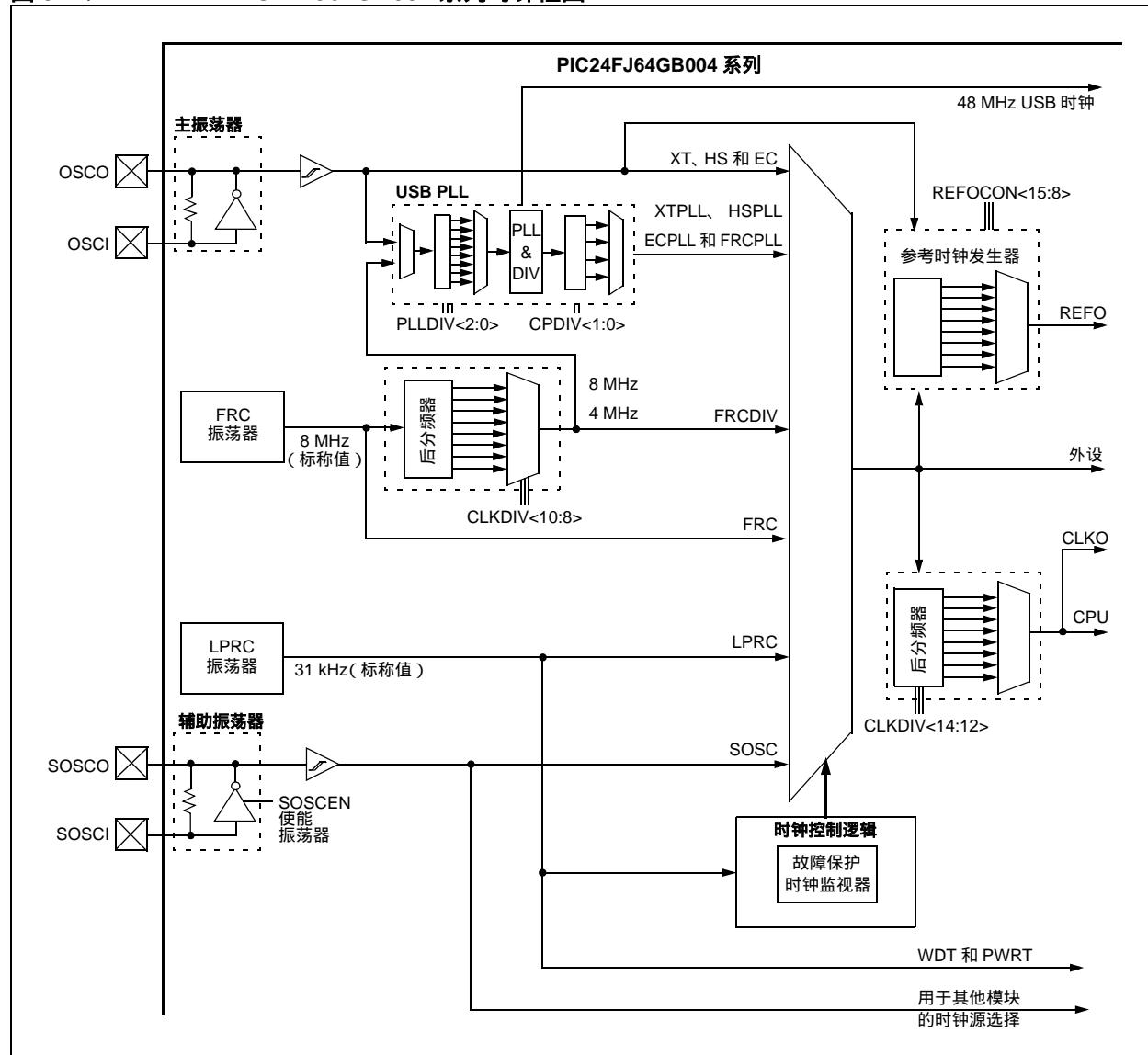
PIC24FJ64GB004 系列器件的振荡器系统具有以下特性：

- 共有四个外部和内部振荡器可选作为时钟源，提供 11 种不同的时钟模式

- 一个片内 USB PLL 模块，用于为 USB 模块提供稳定的 48 MHz 时钟，并为系统时钟提供较大范围的频率选择
- 可通过软件控制在多个时钟源之间切换
- 可通过软件控制后分频器有选择地为 CPU 提供时钟以节省系统功耗
- 具有故障保护时钟监视器 (FSCM)，可检测时钟故障，以便应用可安全地恢复或关闭
- 可单独使用且独立配置的系统时钟输出，以同步外部硬件

图 8-1 所示为振荡器系统的简化框图。

图 8-1：PIC24FJ64GB004 系列时钟框图



## 8.1 CPU 时钟机制

系统时钟源可由以下四种时钟源之一提供：

- 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 )
- 内部快速 RC ( FRC ) 振荡器
- 内部低功耗 RC ( LPRC ) 振荡器

主振荡器和 FRC 源可以选择使用内部 USB PLL 模块，用它来通过 96 MHz PLL 产生 USB 模块时钟和单独的系统时钟。其他信息请参见第 8.5 节“振荡器模式和 USB 工作”。

内部快速 FRC 提供 8 MHz 的时钟源。可以选择用可编程时钟分频器降低其频率，从而提供一系列系统时钟频率。

选择的时钟源提供处理器和外设的时钟。将处理器时钟源二分频可以产生内部指令周期时钟 FCY。在本文档中，指令周期时钟由 Fosc/2 表示。在 OSCO I/O 引脚上可以提供内部指令周期时钟 Fosc/2，用于主振荡器的某些工作模式。

## 8.2 上电复位时的初始配置

通过使用配置位设置选择器件发生上电复位事件时使用的振荡器源（和工作模式）。振荡器配置位设置位于程序存储器中的配置寄存器中（更多详细信息请参见第 26.1 节“配置位”）。主振荡器配置位 POSCMD<1:0>（配置字 2<1:0>）和初始振荡器选择配置位 FNOSC<2:0>（配置字 2<10:8>）用于选择上电复位时使用的振荡器源。带有后分频器的 FRC 主振荡器 (FRCDIV) 是默认（未编程）的选择。对这些位单元进行编程可以选择辅助振荡器或其中一个内部振荡器。

配置位允许用户在多个时钟模式中进行选择，如表 8-1 所示。

### 8.2.1 时钟切换模式配置位

FCKSM 配置位（配置字 2<7:6>）用于联合配置器件时钟切换和故障保护时钟监视器（FSCM）。只有将 FCKSM1 编程为 0 时才可以使能时钟切换功能。只有将 FCKSM<1:0> 编程为 00 时才可以使能 FSCM。

表 8-1：用于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
带有后分频器的快速 RC 振荡器 (FRCDIV)	内部	11	111	1, 2
(保留)	内部	xx	110	1
低功耗 RC 振荡器 (LPRC)	内部	11	101	1
辅助 (Timer1) 振荡器 (SOSC)	辅助	11	100	1
带有 PLL 模块的主振荡器 (XTPLL)	主	01	011	
带有 PLL 模块的主振荡器 (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	
带有 PLL 模块的快速 RC 振荡器 (FRCPPLL)	内部	11	001	1
快速 RC 振荡器 (FRC)	内部	11	000	1

注 1：OSCO 引脚功能由 OSCIOFCN 配置位决定。

2：对于未编程（已擦除）器件，这是默认的振荡器模式。

## 8.3 控制寄存器

振荡器的操作由三个特殊功能寄存器（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。它控制时钟源的切换并允许监视时钟源。

CLKDIV 寄存器（寄存器 8-2）控制打盹模式的相关特性，以及 FRC 振荡器的后分频器。OSCTUN 寄存器（寄存器 8-3）允许用户在大约  $\pm 12\%$  的范围内精确地调整 FRC 振荡器。

### 寄存器 8-1：OSCCON：振荡器控制寄存器

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

R/SO-0	R/W-0	R-0 <sup>(3)</sup>	U-0	R/CO-0	R/W-0	R/W-0	R/W-0
CLKLOCK	IOLOCK <sup>(2)</sup>	LOCK	—	CF	POSCEN	SOSCEN	OSWEN
bit 7	bit 0						

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

bit 15 未实现：读为 0

bit 14-12 **COSC<2:0>**：当前振荡器选择位

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

bit 11 未实现：读为 0

bit 10-8 **NOSC<2:0>**：新振荡器选择位<sup>(1)</sup>

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

**注 1：**这些位的复位值由 FNOSC 配置位决定。

**2：**只能在执行解锁序列后更改 IOLOCK 位的状态。另外，如果 IOL1WAY 配置位为 1，一旦 IOLOCK 位置 1，它就不能清零。

**3：**在进行有效的时钟切换或选择非 PLL 时钟模式时，也复位为 0。

# PIC24FJ64GB004 系列

---

## 寄存器 8-1 : OSCCON : 振荡器控制寄存器 (续)

bit 7	<b>CLKLOCK</b> : 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，可通过将 OSWEN 位置 1 来进行修改 <u>如果禁止 FSCM ( FCKSM1 = 0 ) :</u> 时钟和 PLL 选择始终未锁定，可通过将 OSWEN 位置 1 来进行修改。
bit 6	<b>IOLOCK</b> : I/O 锁定使能位 <sup>(2)</sup> 1 = I/O 锁定有效 0 = I/O 锁定无效
bit 5	<b>LOCK</b> : PLL 锁定状态位 <sup>(3)</sup> 1 = PLL 模块处于锁定状态或 PLL 模块的起振定时器延时结束 0 = PLL 模块不处于锁定状态，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	<b>CF</b> : 时钟故障检测位 1 = FSCM 检测到一个时钟故障 0 = 未检测到时钟故障
bit 2	<b>POSCEN</b> : 主振荡器休眠使能位 1 = 主振荡器在休眠模式期间继续工作 0 = 主振荡器在休眠模式期间被禁止
bit 1	<b>SOSCEN</b> : 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	<b>OSWEN</b> : 振荡器切换使能位 1 = 启动振荡器切换，切换到由 NOSC<2:0> 位指定的时钟源 0 = 完成振荡器切换

注 1：这些位的复位值由 FNOSC 配置位决定。

2：只能在执行解锁序列后更改 IOLOCK 位的状态。另外，如果 IOL1WAY 配置位为 1，一旦 IOLOCK 位置 1，它就不能清零。

3：在进行有效的时钟切换或选择非 PLL 时钟模式时，也复位为 0。

## 寄存器 8-2 : CLKDIV : 时钟分频器寄存器

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

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
CPDIV1	CPDIV0	PLLEN	—	—	—	—	—
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **ROI** : 中断时恢复位  
 1 = 中断时清零 DOZEN 位 , 并将 CPU/ 外设时钟比复位为 1:1  
 0 = 中断不影响 DOZEN 位

bit 14-12    **DOZE<2:0>** : CPU/ 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<sup>(1)</sup>  
 1 = DOZE<2:0> 位指定 CPU/ 外设时钟比  
 0 = 将 CPU/ 外设时钟比设置为 1:1

bit 10-8     **RCDIV<2:0>** : FRC 后分频比选择位  
 111 = 31.25 kHz ( 256 分频 )  
 110 = 125 kHz ( 64 分频 )  
 101 = 250 kHz ( 32 分频 )  
 100 = 500 kHz ( 16 分频 )  
 011 = 1 MHz ( 8 分频 )  
 010 = 2 MHz ( 4 分频 )  
 001 = 4 MHz ( 2 分频 )  
 000 = 8 MHz ( 1 分频 )

bit 7-6      **CPDIV<1:0>** : USB 系统时钟选择位 ( 通过将 32 MHz 时钟进行后分频 )  
 11 = 4 MHz ( 8 分频 )<sup>(2)</sup>  
 10 = 8 MHz ( 4 分频 )<sup>(2)</sup>  
 01 = 16 MHz ( 2 分频 )  
 00 = 32 MHz ( 1 分频 )

bit 5        **PLLEN** : 96 MHz PLL 使能位  
 1 = 使能 PLL  
 0 = 禁止 PLL

bit 4-0      未实现 : 读为 0

注 1 : 当 ROI 位置 1 并发生中断时 , 该位会自动清零。

2 : 使能 USB 模块时不允许此设置。

# PIC24FJ64GB004 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 <sup>(1)</sup>	TUN4 <sup>(1)</sup>	TUN3 <sup>(1)</sup>	TUN2 <sup>(1)</sup>	TUN1 <sup>(1)</sup>	TUN0 <sup>(1)</sup>
bit 7							bit 0

## 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15-6 未实现 : 读为 0

bit 5-0 TUN<5:0> : FRC 振荡器调节位<sup>(1)</sup>

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> 的增加或减少不能在 FRC 调节范围内同步更改 FRC 频率 , 且频率的变化也可能不是单调的。

## 8.4 时钟切换操作

应用可在软件控制下随时在四个时钟源 ( POSC、SOSC、FRC 和 LPRC ) 间自由切换 , 几乎没有什么限制。为限制该灵活性可能带来的负面影响 , PIC24F 器件在时钟切换过程中采用了安全锁定。

注 : 主振荡器模式有三种不同的子模式 ( XT、HS 和 EC ), 这三个子模式由 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重新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 , 必须将 CW2 的 FCKSM 配置位编程为 00。(更多详细信息请参见第 26.1 节 “ 配置位 ” 。) 如果没有对 FCKSM 配置位编程 ( 即 FCKSM 配置位为 1x ) , 则时钟切换功能和故障保护时钟监视器功能都将被禁止 ; 这是默认设置。

在时钟切换被禁止的情况下 , NOSCx 控制位 ( OSCCON<10:8> ) 不控制时钟选择。但是 , COSC<sub>x</sub> 位 ( OSCCON<14:12> ) 将会反映 FNOSC<sub>x</sub> 配置位选择的时钟源。

禁止了时钟切换功能时 , OSWEN 控制位 ( OSCCON<0> ) 不起作用 ; 它始终保持为 0 。

## 8.4.2 振荡器切换序列

执行时钟切换至少需要下列基本序列：

1. 如果需要，读 COSCx 位 ( OSCCON<14:12>) 确定当前振荡器源。
2. 执行解锁序列以允许写入 OSCCON 寄存器的高字节。
3. 将新振荡器源的相关值写入 NOSCx 位 ( OSCCON<10:8>)。
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 来启动振荡器切换。

一旦基本序列完成，系统时钟硬件将自动进行如下响应：

1. 时钟切换硬件将 NOSCx 控制位的新值和 COSCx 位做比较。如果相等，时钟切换为冗余操作。在这种情况下，OSWEN 位被自动清零且时钟切换被中止。
2. 如果启动了有效时钟切换，则 LOCK ( OSCCON<5>) 和 CF ( OSCCON<3>) 位被清零。
3. 如果新振荡器现在不运行，硬件会将其开启。如果必须开启晶振，硬件将等待直至 OST 超时。如果新振荡源正在使用 PLL，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 位以表示时钟切换成功。此外，NOSCx 位的值被传送到 COSCx 位中。
6. 此时旧时钟源被关闭，但 LPRC (如果使能了 WDT 或 FSCM) 或 SOSC (如果 SOSCEN 位保持置 1 状态) 除外。

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

**2：**不允许直接在带 PLL 的任何主振荡器模式和 FRCPLL 模式之间进行时钟切换。这适用于任一方向的时钟切换。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。

时钟切换的建议代码序列通常包括：

1. 在 OSCCON 寄存器解锁和写序列过程中禁止中断。
2. 通过两条连续的指令将 78h 和 9Ah 写入 OSCCON<15:8> 执行 OSCCON 高字节的解锁序列。
3. 执行解锁序列之后，立即使用指令将新振荡器源写入 NOSCx 位。
4. 通过两条连续的指令将 46h 和 57h 写入 OSCCON<7:0> 执行 OSCCON 低字节的解锁序列。
5. 执行解锁序列之后，立即使用指令将 OSWEN 位置 1。
6. 继续执行对时钟要求不高的代码（可选）。
7. 调用适当数量的软件延时（周期计数），允许选定的振荡器和 / 或 PLL 启动并稳定下来。
8. 检查 OSWEN 位是否为 0。如果是，则切换成功。如果 OSWEN 位仍为置 1 状态，则检查 LOCK 位以确定故障原因。

解锁 OSCCON 寄存器并启动时钟切换的核心序列如例 8-1 所示。

### 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
```

# PIC24FJ64GB004 系列

## 8.5 振荡器模式和 USB 工作

由于 USB 特有的时序要求，所以使能 USB 模块时始终需要一个 48 MHz 的内部时钟。由于这远远超出了 CPU 最高时钟速度，因此提供了一个方法，从单一振荡器源内部生成 USB 时钟和系统时钟。PIC24FJ64GB004 系列器件使用了与其他 PIC24FJ 器件相同的时钟结构，但包含一个有两个分支的 PLL 系统，用于生成两个时钟信号。

USB PLL 模块如图 8-2 所示。在这个系统中，主振荡器的输入被 PLL 预分频器分频，产生 4 MHz 的输出。这可以用来驱动片内 96 MHz PLL 倍频器，从而驱动两路时钟。一路使用固定的 2 分频分频器来产生 48 MHz 的 USB 时钟。另一路使用固定的 3 分频分频器和可配置的 PLL 预分频器 / 倍频器产生一组系统时钟频率。CPDIV 位选择系统时钟速度；表 8-2 列出了可用时钟选项。

USB PLL 预分频器不会自动检测传入的振荡器频率。用户必须使用 PLLDIV<2:0> 配置位手动配置 PLL 分频器，以产生所需的 4 MHz 输出。这就将主振荡器频率选择限制为 8 种，如表 8-3 所示。

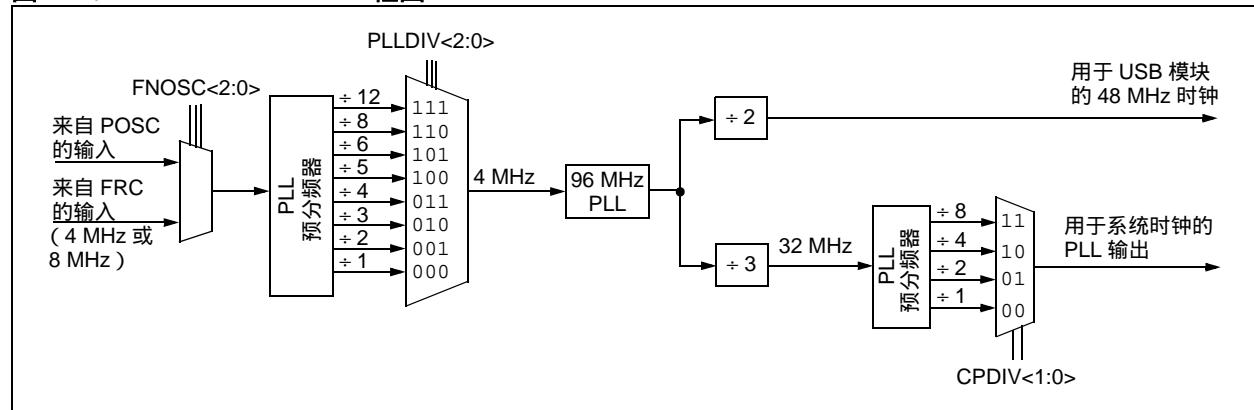
表 8-2：USB 工作期间的系统时钟选项

MCU 时钟分频 (CPDIV<1:0>)	单片机 时钟频率
无 (00)	32 MHz
÷2 (01)	16 MHz
÷4 (10)	8 MHz
÷8 (11)	4 MHz

表 8-3：确保 USB 正常工作的有效主振荡器配置

振荡器输入频率	时钟模式	PLL 分频 (PLLDIV<2:0>)
48 MHz	ECPLL	÷12 (111)
32 MHz	ECPLL	÷8 (110)
24 MHz	HSPLL 和 ECPLL	÷6 (101)
20 MHz	HSPLL 和 ECPLL	÷5 (100)
16 MHz	HSPLL 和 ECPLL	÷4 (011)
12 MHz	HSPLL 和 ECPLL	÷3 (010)
8 MHz	XTPLL 和 ECPLL	÷2 (001)
4 MHz	XTPLL 和 ECPLL	÷1 (000)

图 8-2：USB PLL 框图



### 8.5.1 USB 工作的注意事项

使用 PIC24FJ64GB004 系列器件的 USB On-The-Go 模块时，用户在配置系统时钟时必须始终遵守以下这些规则：

- 要确保 USB 正常工作，所选的时钟源（EC、HS 或 XT）必须满足 USB 时钟容差的要求。

- 主振荡器 /PLL 模式是允许 USB 工作的唯一振荡器配置。没有为 USB 模块提供单独的外部时钟源。
- 所有振荡器模式都是可用的；但是，选择这些模式时 USB 无法工作。但是这些模式仍然可以用在需要其他工作功耗级别且无需 USB 模块的应用中（例如，应用处于休眠模式下并等待总线连接）。

## 8.6 辅助振荡器 (SOSC)

### 8.6.1 基本 SOSC 工作

PIC24FJ64GB004 系列器件不必置 1 SOSCEN 位来使用辅助振荡器。任何需要 SOSC 的模块（例如 RTCC、Timer1 或 DSWDT）在需要时钟信号时将自动开启 SOSC。但是，SOSC 的起振时间较长。为避免外设启动时产生延迟，可使用 SOSCEN 位手动启动 SOSC。

要使用辅助振荡器，必须将  $SOSCSEL<1:0>$  位 ( $CW3<9:8>$ ) 配置为下面两种振荡器模式之一：11 或 01。 $SOSCSEL<1:0>$  设置为 00 时可配置 SOSC 引脚为数字模式，从而使能该引脚上的数字 I/O 功能。如果 SOSC 配置为任一振荡器模式，则数字功能将不可用。

### 8.6.2 低功耗 SOSC 工作

辅助振荡器可根据器件配置的要求工作在两种不同的功耗级别下。在低功耗模式下，振荡器工作在低驱动强度、低功耗状态下。默认情况下，振荡器使用较高的驱动强度，因而需要更多的功耗。辅助振荡器模式配置位  $SOSCSEL<1:0>$  ( $CW3<9:8>$ ) 可决定辅助振荡器的功耗模式。将 SOSCSEL 编程为 01 选择低功耗工作模式。由于此模式具有低驱动强度，使得 SOSC 对噪声更加敏感，从而需要较长的起振时间。使用低功耗模式时，在 SOSC 电路的设计和布线过程中必须小心，以确保振荡器正确起振和运行。

### 8.6.3 外部（数字）时钟模式（SCLKI）

SOSC 也可配置为使用外部 32 kHz 时钟源（而不是内部振荡器）运行。在此模式（也称作数字模式）下，可使用 SCLKI 引脚上提供的时钟源给配置为使用辅助振荡器的任何模块提供时钟。在此模式下，晶振驱动电路被禁止，SOSCEN 位 ( $OSCCON<1>$ ) 不起作用。

### 8.6.4 SOSC 布线注意事项

低引脚数器件（例如 PIC24FJ64GB004 系列中的低引脚数器件）的引脚排列限制使得 SOSC 比其他 PIC24F 器件对噪声更加敏感。除非在 SOSC 电路的设计和布线过程中非常小心，否则这种外部噪声会使振荡器的周期不精确。

通常情况下，晶振电路的连接应尽可能的短。在晶振电路周围布置一个接地回路或接地板也是一种好的做法。关于晶振电路设计的更多信息，请参见《PIC24F 系列参考手册》的第 6 章“振荡器”(DS39700A\_CN)。还可参见以下这些 Microchip 应用笔记了解更多信息：

- AN826，“Crystal Oscillator Basics and Crystal Selection for rfPIC® and PICmicro® Devices” (DS00826)
- AN849，“Basic PICmicro® Oscillator Design” (DS00849)。

## 8.7 参考时钟输出

除了某些振荡器模式中使用的 CLKO 输出 (Fosc/2) 外，还可对 PIC24FJ64GB004 系列器件中的器件时钟进行配置，以为端口引脚提供参考时钟输出信号。此特性可用于所有的振荡器配置，用户可通过它选择更大范围的时钟分频比以驱动应用中的外部器件。

REFOCON 寄存器控制此参考时钟输出（寄存器 8-4）。置 1 ROEN 位 (REFOCON<15>) 可使 REFO 引脚上的时钟信号可用。RODIV 位 (REFOCON<11:8>) 允许选择 16 种不同的时钟分频比选项。

ROSSLP 和 ROSEL 位 (REFOCON<13:12>) 控制休眠模式期间参考输出是否可用。ROSEL 位决定是由 OSC1 和 OSC2 上的振荡器，还是由当前系统时钟源提供参考时钟输出。ROSSLP 位决定当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，必须将 ROSSLP 和 ROSEL 位都置 1。器件时钟也必须配置为其中一种主模式 (EC、HS 或 XT)；否则，如果 POSCEN 位也没有置 1，那么当器件进入休眠模式时，OSC1 和 OSC2 上的振荡器会掉电。任何时钟切换期间清零 ROSEL 位都会允许参考输出频率随着系统时钟的变化而变化。

# PIC24FJ64GB004 系列

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 = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15      **ROEN** : 参考振荡器输出使能位  
1 = 使能 REFO 引脚上的参考振荡器  
0 = 禁止参考振荡器
- bit 14      未实现 : 读为 0
- bit 13      **ROSSLP** : 休眠模式下参考振荡器输出停止位  
1 = 休眠模式下参考振荡器继续工作  
0 = 休眠模式下禁止参考振荡器
- bit 12      **ROSEL** : 参考振荡器源选择位  
1 = 主振荡器用作基本时钟。注意 , 必须使用 FOSC<2:0> 位使能晶振 ; 晶振在休眠模式下继续工作。  
0 = 系统时钟用作基本时钟 ; 时基反映了器件的任何时钟切换
- bit 11-8     **RODIV<3:0>** : 参考振荡器分频比选择位  
1111 = 基本时钟值的 32,768 分频  
1110 = 基本时钟值的 16,384 分频  
1101 = 基本时钟值的 8,192 分频  
1100 = 基本时钟值的 4,096 分频  
1011 = 基本时钟值的 2,048 分频  
1010 = 基本时钟值的 1,024 分频  
1001 = 基本时钟值的 512 分频  
1000 = 基本时钟值的 256 分频  
0111 = 基本时钟值的 128 分频  
0110 = 基本时钟值的 64 分频  
0101 = 基本时钟值的 32 分频  
0100 = 基本时钟值的 16 分频  
0011 = 基本时钟值的 8 分频  
0010 = 基本时钟值的 4 分频  
0001 = 基本时钟值的 2 分频  
0000 = 基本时钟值
- bit 7-0      未实现 : 读为 0

## 9.0 节能特性

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中第 39 章“带深度休眠的节能特性”(DS39727A\_CN)。

PIC24FJ64GB004 系列器件提供了管理功耗的功能，该功能是通过选择性地管理 CPU 和外设的时钟来实现的。一般而言，较低的时钟频率和减少时钟源驱动电路的数目会使功耗降低。所有 PIC24F 器件通过以下四种方法管理功耗：

- 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
- 通过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特性（如对于时序要求高的通信）的情况下有选择地调节应用的功耗。

### 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”中进行了详细的讨论。

### 9.2 基于指令的节能模式

PIC24F 器件有两种特殊的节能模式，通过执行特殊的 PWRS AV 指令可以进入这两种模式。休眠模式下时钟停止运行并停止所有代码执行；空闲模式下 CPU 停止工作，代码停止执行，但是允许外设模块继续工作。深度休眠模式下时钟停止运行，代码停止执行，且除 RTCC 和 DSWDT 之外的所有外设都停止运行，还会冻结 I/O 状态并切断 SRAM 和闪存的电源。

PWRS AV 指令的汇编语法如例 9-1 所示。

**注：**SLEEP\_MODE 和 IDLE\_MODE 是在所选器件的汇编器头文件中定义的常量。

允许的中断、WDT 超时或器件复位会导致器件从休眠和空闲模式退出。器件退出这两种模式的过程称为“唤醒”。

#### 9.2.1 休眠模式

休眠模式有以下这些特性：

- 系统时钟源关闭。如果使用了片上振荡器，也要关闭它。
- 如果 I/O 引脚上不消耗电流，则器件电流消耗将降至最低。
- 冻结 I/O 引脚方向和状态。
- 由于禁止了系统时钟源，所以故障保护时钟监视器在休眠模式下不工作。
- 如果 WDT 或 LPRC 作为时钟源的 RTCC 被使能，LPRC 时钟将继续在休眠模式下运行。
- 若使能了 WDT，它将在进入休眠模式之前自动清零。
- 某些器件功能或外设可在休眠模式下继续工作。这包括 I/O 端口上的输入电平变化通知功能或使用外部时钟输入的外设等。任何需要系统时钟源工作的外设在休眠模式下将被禁止。

当发生以下任何事件时，器件将从休眠模式唤醒：

- 产生任何被单独允许的中断源
- 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启动。

#### 例 9-1：PWRS AV 指令语法

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

## 9.2.2 空闲模式

空闲模式有以下这些特性：

- CPU 将停止执行指令。
- WDT 自动清零。
- 系统时钟源保持工作状态。默认情况下，所有外设模块将使用系统时钟源继续正常工作，但也可被有选择地禁止（见第 9.4 节“选择性外设模块控制”）。
- 若使能了 WDT 或 FSCM，LPRC 也将保持工作状态。

当发生以下任何事件时，器件将从空闲模式唤醒：

- 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，为 CPU 重新提供时钟，且立即从 PWRS AV 指令之后的下一条指令或 ISR 中的第一条指令开始执行。

## 9.2.3 在节能指令执行期间的中断

在 PWRS AV 指令执行期间发生的中断都将延迟到进入休眠或空闲模式后才产生（深度休眠模式除外），并导致器件从休眠或空闲模式唤醒。

## 9.2.4 深度休眠模式

在 PIC24FJ64GB004 系列器件中，深度休眠模式用于提供可用的最低功耗级别，无需使用外部开关就可切断器件的所有电源。深度休眠模式的进入是完全受软件控制的。发生以下任何事件时可触发器件退出深度休眠模式：

- POR 事件
- MCLR 事件
- RTCC 闹钟（如果使用了 RTCC 的话）
- 外部中断 0
- 深度休眠看门狗定时器（DSWDT）超时

在深度休眠模式下，可以保持器件的实时时钟和日历（RTCC）继续工作且不会丢失时钟周期。

器件使用专用的深度休眠欠压复位（Deep Sleep Brown-out Reset，DSBOR）和深度休眠看门狗定时器复位（Deep Sleep Watchdog Timer Reset，DSWDT）监视电压和超时事件。DSBOR 和 DSWDT 与其他功耗管理模式（休眠、空闲和打盹）下使用的标准 BOR 和 WDT 是相互独立的。

**注：** 由于深度休眠模式下关闭片上 VDDCORE 稳压器可使单片机掉电，因此仅当器件工作在使能内部稳压器的情况下，深度休眠功能才可用。

### 9.2.4.1 进入深度休眠模式

进入深度休眠模式的方法是：将 DSCON 寄存器中的 DSEN 位置 1，然后在一至三个指令周期内执行 SLEEP 指令（PWRS AV #SLEEP\_MODE）以降低假进入深度休眠模式的可能性。

如果 PWRS AV 命令不是在三个指令周期内发出的，DSEN 位将由硬件清零，且必须由软件再次置 1，然后才能进入深度休眠模式。退出深度休眠模式时，DSEN 位也将自动清零。

**注：** 要在深度休眠唤醒之后再次进入深度休眠模式，允许在清零 RELEASE 位之后至少有一个 3 TCY 的延迟。

进入深度休眠模式的步骤是：

1. 如果应用需要深度休眠 WDT，请使能它并配置其时钟源（详细信息请参见第 9.2.4.7 节“深度休眠 WDT”）。
2. 如果应用需要深度休眠 BOR，请通过编程 DSBOREN 配置位（CW4<6>）使能它。
3. 如果应用需要在发生 RTCC 闹钟事件时从深度休眠模式唤醒，请使能并配置 RTCC 模块（更多信息请参见第 20.0 节“实时时钟和日历（RTCC）”）。
4. 如有需要，请通过把应用程序的任何现场数据写入 DSGPR0 和 DSGPR1 寄存器来保存这些数据（可选）。
5. 通过置 1 DSEN 位（DSCON<15>）使能深度休眠模式。
6. 通过立即发出 PWRS AV #0 指令进入深度休眠模式。

任何时候将 DSEN 位置 1，DSWAKE 寄存器中的所有位都将自动清零。

## 9.2.4.2 进入深度休眠模式的特殊情形

进入深度休眠模式时，在某些情况下，需要在 DSEN 位置 1 和执行 PWRSAV 指令之间有一个延时。通常有以下三种情形：

1. 情形（1）：使用外部唤醒源（INT0）或 RTCC
2. 情形（2）：具有可临时禁止的应用级中断
3. 情形（3）：具有必须要监视的中断

在第一种情形下，应用需要在 INT0 引脚有效或发生 RTCC 中断时从深度休眠模式唤醒。在此情形下，必须插入三条 NOP 指令，以在器件进入深度休眠模式后能够正确同步检测异步 INT0 中断。如果应用不使用在 INT0 引脚有效或发生 RTCC 中断时唤醒功能，那么是否添加 NOP 指令无关紧要。

在第二种情形下，应用还使用了可暂时忽略的中断。在这些应用中，如果执行 NOP 指令期间发生了中断事件，那么可能会导致执行 ISR。这意味着在返回代码之前将经过三个以上的指令周期，并且 DSEN 位将被清零。为防止误进入深度休眠模式，需在进入深度休眠模式之前临时禁止中断。调用 DISI 指令四个指令周期，足以防止中断导致器件误进入深度休眠模式。

在第三种情形下，中断不能被忽略，即使暂时忽略也不可以；需要一直进行中断检测，即使在 DSEN 置 1 和执行 PWRSAV 指令之间的时间间隔内也是如此。对于这些情形，可以禁止中断并检测中断条件，必要的话还可跳过 PWRSAV 指令。可通过检查 CPUIRQ 位（INTTREG<15>）的状态来实现中断检测；如果有未处理的中断被挂起，此位将置 1。如果在执行 PWRSAV 指令之前 CPUIRQ 位置 1，那么将跳过该指令。此时，DISI 指令的周期已到（从它被执行起已过了 4 个指令周期以上），且应用程序跳转到相应的 ISR。当应用程序返回时，它将尝试再次进入深度休眠模式或执行某个其他的系统功能。无论哪种情况，都必须在 PWRSAV 指令后放置一些功能代码，以使跳过 PWRSAV 指令时，器件不会进入深度休眠模式。

例 9-2 给出了实现这些情况的示例。建议在这些情况下使用汇编程序或内联 C 程序，以确保代码在所需的周期数内执行。

### 例 9-2：实现进入深度休眠的特殊情形

```
// Case 1: simplest delay scenario
//
asm("bset DSCON, #15");
asm("nop");
asm("nop");
asm("nop");
asm("pwrssav #0");
//
// Case 2: interrupts disabled
//
asm("disi #4");
asm("bset DSCON, #15");
asm("nop");
asm("nop");
asm("nop");
asm("pwrssav #0");
//
// Case 3: interrupts disabled with
// interrupt testing
//
asm("disi #4");
asm("bset DSCON, #15");
asm("nop");
asm("nop");
asm("btss INTTREG, #15");
asm("pwrssav #0");
// continue with application code here
//
```

## 9.2.4.3 退出深度休眠模式

发生以下任一事件时将退出深度休眠模式：

- VDD 电源发生 POR 事件。如果没有用 DSBOR 电路重新激活 VDD 电源 POR 电路，那么必须把外部 VDD 电源降低到 POR 电路的自然激活电压。
- DSWDT 超时。DSWDT 定时器超时后，器件退出深度休眠模式。
- RTCC 闹钟（如果 RTCEN = 1）。
- MCLR 引脚有效（0）。
- INT0 引脚有效（如果在进入深度休眠模式之前允许此中断的话）。极性配置用于确定导致器件退出深度休眠模式的引脚的有效电平（0 或 1）。深度休眠模式期间，INT0 引脚上的电平变化会导致器件退出深度休眠模式。

**注：** 进入深度休眠模式时，所有待处理的中断都将被清除。

退出深度休眠模式通常不会保持器件的状态，它等同于器件的上电复位（POR）。但 RTCC（如果有，它在唤醒后继续保持工作）、DSGPRx 寄存器和 DSWDT 为例外情况。

忽略从退出深度休眠模式到 POR 序列完成过程中发生的唤醒事件，且不会被捕获到 DSWAKE 寄存器中。

退出深度休眠模式的步骤是：

1. 发生唤醒事件后，器件退出深度休眠模式并执行 POR。DSEN 位自动清零。代码将从复位向量处继续执行。
2. 要确定器件是否已退出深度休眠模式，应读深度休眠位 DPSLP（RCON<10>）。如果已退出深度休眠模式，该位将置 1。如果该位置 1，请将其清零。
3. 通过读 DSWAKE 寄存器确定唤醒源。
4. 通过读 DSBOR 位（DSCON<1>）确定在深度休眠模式期间是否发生了 DSBOR 事件。
5. 如果已保存了应用的现场数据，请将其从 DSGPR0 和 DSGPR1 寄存器读回。
6. 清零 RELEASE 位（DSCON<0>）。

#### 9.2.4.4 深度休眠唤醒时间

由于从深度休眠唤醒会导致上电复位，所以从深度休眠唤醒的时间与器件上电复位的时间相同。而且，因为内部稳压器被关闭，VCAP 上的电压下降量可能会根据器件休眠时间的长短而有所不同。如果 VCAP 降至 2V 以下，那么唤醒时间还将延长以使稳压器对 VCAP 充电。

在第 29.0 节“电气特性”中将深度休眠唤醒时间指定为 TDSWU。此规范指定最差情况下的唤醒时间（包括完全上电复位时间（包括 TPOR 和 TRST））以及对 VCAP 上已放电至 0V 的 10 μF 电容完全充电的时间。如果 VCAP 未放电，唤醒可能要快得多。

#### 9.2.4.5 使用 DSGPR0/DSGPR1 寄存器保存现场数据

由于退出深度休眠模式会导致 POR，大部分特殊功能寄存器都将复位为默认的 POR 值。而且，由于在深度休眠模式下不提供 VDDCORE 电源，所以退出此模式时有可能会丢失数据 RAM 中的信息。

需要在进入深度休眠模式之前保存关键数据的应用可使用深度休眠通用寄存器 DSGPR0 和 DSGPR1 或数据 EEPROM（如果可用的话）。与其他 SFR 不同，当器件处于深度休眠模式时这些寄存器的内容被保留。退出深度休眠模式后，可用软件通过读这些寄存器并清零 RELEASE 位（DSCON<0>）来恢复数据。

#### 9.2.4.6 深度休眠模式期间的 I/O 引脚

深度休眠模式期间，通用 I/O 引脚保留其之前的状态且辅助振荡器（SOSC）将继续运行（如果使能的话）。在进入深度休眠模式之前配置为输入（TRIS 位置 1）的引脚在深度休眠模式期间保持高阻态。在进入深度休眠模式之前配置为输出（TRIS 位清零）的引脚在深度休眠模式期间保持为输出引脚。在此模式期间，它们继续驱动在进入深度休眠模式时其对应的 LAT 位所确定的输出电平。

一旦器件被唤醒，所有 I/O 引脚将继续保持其之前的状态，即使在器件完成了 POR 序列并再次执行应用代码后也是如此。配置为输入的引脚在深度休眠模式期间保持高阻态，配置为输出的引脚继续驱动其之前的电平值。唤醒后，TRIS 和 LAT 寄存器以及 SOSCEN 位 (OSCCON<1>) 都将复位。如果固件修改了这些位或寄存器中的任何一个，I/O 将不会立即进入新配置的状态。固件清零 RELEASE 位 (DSCON<0>) 将立即“释放”I/O 引脚。从而导致 I/O 引脚处于其对应的 TRIS 和 LAT 位的值所配置的状态。

这意味着唤醒后保持 SOSC 运行需要在清零 RELEASE 之前将 SOSCEN 位置 1。

如果使能了深度休眠 BOR (DSBOR)，且深度休眠期间发生了 DSBOR 或真正的 POR 事件，那么 I/O 引脚将立即被释放，这与清零 RELEASE 位有些类似。所有之前的状态信息都将丢失，包括通用 DSGPRO 和 DSGPR1 的内容。

如果深度休眠期间发生了 MCLR 复位事件，那么 DSGPRx、DSCON 和 DSWARE 寄存器将保持有效且 RELEASE 位将保持置 1 状态。SOSC 的状态也将保持。但是，I/O 引脚将复位为 MCLR 复位状态。由于 RELEASE 仍然置 1，所以对 SOSCEN 位 (OSCCON<1>) 的更改直到 RELEASE 位清零之后才会生效。

在所有其他深度休眠唤醒情况下，应用固件必须清零 RELEASE 位才能重新配置 I/O 引脚。

## 9.2.4.7 深度休眠 WDT

要使能深度休眠模式下的 DSWDT，请编程配置位 DSWDTEN (CW4<7>)。无需为了 DSWDT 正常工作而使能器件的看门狗定时器 (WDT)。进入深度休眠模式将自动复位 DSWDT。

通过 DSWDTOSC 配置位 (CW4<4>) 选择 DSWDT 的时钟源。通过 DSWDTPS<3:0> 配置位 (CW4<3:0>) 编程后分频比选项。可实现的最小超时周期为 2.1 ms，最大为 25.7 天。更多关于 CW4 配置寄存器和 DSWDT 配置选项的信息，请参见第 26.0 节“特殊功能”。

## 9.2.4.8 深度休眠模式下切换时钟

RTCC 和 DSWDT 可使用 SOSC 或 LPRC 时钟源提供的时钟运行。这使得 RTCC 和 DSWDT 可以在不需要同时使能 LPRC 和 SOSC 的情况下运行，从而降低功耗。

RTCC 使用 LPRC 提供的时钟运行将导致 RTCC 的精度损失大约 5% 至 10%。如果需要精确的 RTCC，必须使用 SOSC 时钟源提供的时钟运行 RTCC。通过 RTCOSC 配置位 (CW4<5>) 选择 RTCC 的时钟源。

在某些环境下，可以在进入深度休眠模式时关闭 DSWDT 的时钟源。在这种情况下，无需软件干预，就可自动开启时钟源（如果使能了 DSWDT）。但是，这会导致 DSWDT 计数器启动时有一段延时。在 SOSC 用作时钟源时，为了避免出现延时，应用可在进入深度休眠模式之前激活 SOSC。

## 9.2.4.9 检查并清零深度休眠的状态位

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

上电时，软件应读取该状态位以确定复位是否是由退出深度休眠模式引起的，如果该位置 1 的话，则将其清零。在 DPSLP 和 POR 位状态的四种可能组合中，需注意以下三种情况：

- DPSLP 和 POR 位都清零。在这种情况下，复位是由其他事件（而不是退出深度休眠模式）引起的。
- DPSLP 位清零，但是 POR 位置 1。这是正常的上电复位。
- DPSLP 和 POR 位都置 1。这表示发生了以下情况：进入了深度休眠模式，器件掉电，然后退出深度休眠模式。

## 9.2.4.10 上电复位 (POR)

监视 V<sub>DD</sub> 电压以产生 POR。由于退出深度休眠模式在功能上与 POR 类似，应使用第 9.2.4.9 节“**检查并清零深度休眠的状态位**”中描述的技巧来区别退出深度休眠和真正的 POR 事件。

发生真正的 POR 时，整个器件包括所有深度休眠逻辑（深度休眠寄存器、RTCC 和 DSWDT 等）在内都将复位。

## 9.2.4.11 深度休眠步骤总结

下面回顾了进入和退出深度休眠模式的必需步骤：

1. 器件退出复位并开始执行其应用代码。
2. 如果需要 DSWDT 功能，编程其相应的配置位。
3. 为 DSWDT 和 RTCC 选择合适的时钟（可选）。
4. 使能和配置 RTCC（可选）。
5. 将现场数据写入 DSGPRx 寄存器（可选）。
6. 允许 INT0 中断（可选）。
7. 将 DSCON 寄存器中的 DSEN 位置 1。
8. 通过发出 PWRSV #SLEEP\_MODE 命令进入深度休眠模式。
9. 发生唤醒事件时器件退出深度休眠模式。
10. DSEN 位自动清零。
11. 读取并清零 RCON 中的 DPSLP 状态位以及 DSWARE 状态位。
12. 读取 DSGPRx 寄存器（可选）。
13. 完成所有状态相关配置后，清零 RELEASE 位。
14. 应用恢复正常工作。

## 寄存器 9-1 : DSCON : 深度休眠控制寄存器

R/W-0, HC	U-0	U-0	U-0	U-0	U-0	U-0	U-0
DSEN <sup>(1)</sup>	—	—	—	—	—	—	—
bit 15	bit 8						

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0, HCS	R/C-0, HS
—	—	—	—	—	—	DSBOR <sup>(1,2,3)</sup>	RELEASE <sup>(1,2)</sup>
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

C = 可清零位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

HC = 可由硬件清零的位

HS = 可由硬件置 1 的位

HCS = 可由硬件清零 / 置 1 的位

bit 15

#### DSEN : 深度休眠使能位<sup>(1)</sup>

1 = 在下一条指令执行 PWRSAV #0 指令后 , 器件进入深度休眠模式

0 = 执行 PWRSAV #0 指令后器件进入正常的休眠模式

bit 14-2

#### 未实现 : 读为 0

bit 1

#### DSBOR : 深度休眠 BOR 事件状态位<sup>(1,2,3)</sup>

1 = DSBOR 处于活动状态 , 且在深度休眠模式期间检测到 BOR 事件

0 = DSBOR 被禁止或处于活动状态 , 但在深度休眠模式期间未检测到 BOR 事件

bit 0

#### RELEASE : 深度休眠时 I/O 引脚状态释放位<sup>(1,2)</sup>

1 = I/O 引脚和 SOSC 在退出深度休眠模式后保持其状态 ( 不管其 LAT 和 TRIS 配置如何 )。

0 = I/O 引脚和 SOSC 从其深度休眠模式状态释放。引脚状态由 LAT 和 TRIS 配置以及 SOSCEN 位控制。

注 1 : 这些位仅在发生深度休眠模式以外的 POR 事件时才复位。

2 : 复位值仅在首次上电 POR 时为 0 , 深度休眠 POR 时为 1。

3 : 这只是状态位 ; DSBOR 事件不会导致从深度休眠模式唤醒。

# PIC24FJ64GB004 系列

寄存器 9-2 : DSWAKE : 深度休眠唤醒源寄存器

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0, HS
—	—	—	—	—	—	—	DSINT0 <sup>(1)</sup>
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 <sup>(1)</sup>	—	—	DSWDT <sup>(1)</sup>	DSRTC <sup>(1)</sup>	DSMCLR <sup>(1)</sup>	—	DSPOR <sup>(2)</sup>
bit 7							bit 0

图注 : HS = 可由硬件置 1 的位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15-9 未实现 : 读为 0

bit 8 DSINT0 : 电平变化中断位<sup>(1)</sup>

1 = 深度休眠期间 , 外部中断 0 有效

0 = 深度休眠期间 , 外部中断 0 无效

bit 7 DSFLT : 深度休眠故障检测位<sup>(1)</sup>

1 = 深度休眠期间发生了故障 , 一些深度休眠配置设置可能遭到破坏

0 = 深度休眠期间未检测到任何故障

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<sup>(1)</sup>

1 = 深度休眠期间 , 深度休眠看门狗定时器超时

0 = 深度休眠期间 , 深度休眠看门狗定时器未超时

bit 3 DSRTC : 实时时钟和日历闹钟位<sup>(1)</sup>

1 = 深度休眠期间实时时钟和日历触发了闹钟

0 = 深度休眠期间实时时钟和日历未触发闹钟

bit 2 DSMCLR : 深度休眠 MCLR 事件位<sup>(1)</sup>

1 = 深度休眠期间 , MCLR 引脚有效

0 = 深度休眠期间 , MCLR 引脚无效

bit 1 未实现 : 读为 0

bit 0 DSPOR : 上电复位事件位<sup>(2)</sup>

1 = VDD 电源 POR 电路处于活动状态且检测到 POR 事件

0 = VDD 电源 POR 电路处于不活动状态 , 或者处于活动状态 , 但是未检测到 POR 事件

注 1 : 仅在器件处于深度休眠模式时此位置 1。

2 : 不处于深度休眠模式时此位置 1。

## 9.3 打盹模式

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

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，继续以相同的时钟源和相同的速度驱动系统时钟。外设模块时钟速度保持不变，但 CPU 时钟的速度降低了。保持两个时钟域同步，以允许外设在 CPU 以较慢的速率执行代码时访问 SFR。

通过将 DOZEN 位 (CLKDIV<11>) 置 1 使能打盹模式。外设和内核时钟速度之比由 DOZE<2:0> 位 (CLKDIV<14:12>) 决定。有 8 种可能的配置，从 1:1 到 1:128，其中 1:1 是默认设置。

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

## 9.4 选择性外设模块控制

空闲和打盹模式允许用户通过降低 CPU 时钟速度或停止 CPU 时钟大幅降低功耗。即使如此，外设模块仍然使用时钟源，因此会有功耗。可能在有些情况下应用需要这些模式无法提供的功能，比如将绝大部分能源分配给 CPU 处理工作，而只为外设提供最低的功耗。

PIC24F 器件允许有选择地禁止外设模块，从而降低或消除它们的功耗，以此满足上述需求。可通过两个控制位完成此操作：

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

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

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

要节省更多的能耗，也可在器件进入空闲模式时有选择地禁止外设模块。使用通用名格式为“XXXIDL”的控制位可以执行此操作。默认情况下，可以在空闲模式下工作的所有模块都可以执行此操作。使用“在空闲模式下禁止”功能可以进一步降低空闲模式下的功耗，从而增强了对功耗要求极高的应用的节能功能。

# **PIC24FJ64GB004 系列**

---

---

注：

## 10.0 I/O 端口

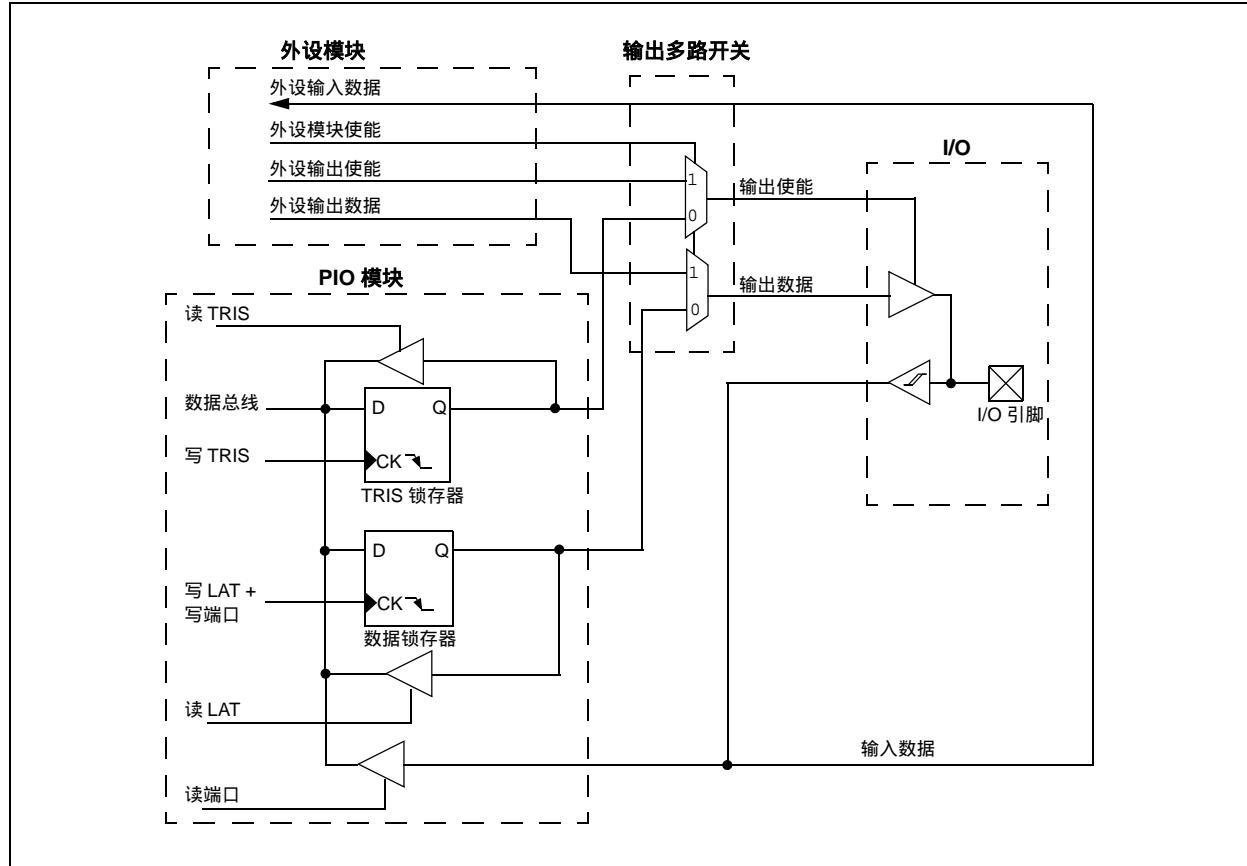
**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 12 章“带外设引脚选择 (PPS) 的 I/O 端口”(DS39711A\_CN)。

所有器件引脚（除了 VDD、VSS、MCLR 和 OSC1/CLK1 以外）均由外设和并行 I/O 端口共用。所有 I/O 输入端口都为施密特触发器输入，以便提高噪声抑制能力。

### 10.1 并行 I/O (PIO) 端口

通常，与某个外设共用一个引脚的并行 I/O 端口总是服从于该外设。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用于外设还是相应的端口。该逻辑同时会阻止“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输入。图 10-1 中显示端口是如何与其他外设复用的，及其连接的关联 I/O 引脚。

图 10-1：共用端口的典型结构框图



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

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

对于某个特定器件，无效的位及其相关的数据和控制寄存器都将被禁止。这意味着相应的 LAT 和 TRIS 寄存器以及该端口引脚将读为 0。

当一个定义为只用作输入的引脚与另一个外设或功能复用时，由于没有其他竞争的输出源，它将被视为专用端口。

## 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对应的漏极开路控制寄存器 ODCx 控制的。将其中的任何位置 1 即可将相应的引脚配置为漏极开路输出。

这种开漏特性允许通过使用外部上拉电阻在任何只能用作数字功能的引脚上产生高于 VDD（如 5V）的输出电平。允许的最大开漏电压与最大 VIH 规范相同。

## 10.2 配置模拟端口引脚

AD1PCFG1 和 TRIS 寄存器用于控制 A/D 端口引脚的操作。若希望端口引脚设置为模拟输入引脚，则必须将其对应的 TRIS 位置 1。若将 TRIS 位清零（输出），则该引脚的数字输出电平（VOH 或 VOL）将被转换。

当对 PORT 寄存器进行读操作时，所有配置为模拟输入通道的引脚将被读为零（低电平）。

配置为数字输入的引脚将不会对模拟输入进行转换。任何定义为数字输入的引脚（包括 ANx 引脚）上的模拟电平可能导致输入缓冲器消耗的电流超过器件规范限值。

## 10.2.1 I/O 端口写 / 读时序

在改变端口方向或对端口执行写操作，与对同一端口执行读操作之间需要间隔一个指令周期。通常在两者之间插一条 NOP 指令（例 10-1）。

## 10.2.2 模拟输入引脚和输入电压注意事项

用作器件输入的引脚可承受的电压与该引脚的输入功能有关。只用作数字输入的引脚能够承受最高 5.5V 的直流电压，这是数字逻辑电路的典型电平值。而同时具有任何类型模拟输入功能的引脚只能承受最高 VDD 的电压。应避免使这些引脚上的电压超出 VDD。

表 10-1 总结了输入电压能力。更多详细信息，请参见第 29.0 节“电气特性”。

表 10-1： 可承受的输入电压

端口或引脚	可承受的输入电压	说明
PORTA<4:0>	VDD	只能承受 VDD 输入电平。
PORTB<15:13>		
PORTB<4:0>		
PORTC<3:0> <sup>(1)</sup>		
PORTA<10:7> <sup>(1)</sup>	5.5V	可承受高于 VDD 的输入电平，这对于大部分标准逻辑有用。
PORTB<11:7>		
PORTB<5>		
PORTC<9:4> <sup>(1)</sup>		

注 1： 在 28 引脚器件上未提供。

### 例 10-1： 端口写 / 读示例

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

## 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PIC24FJ64GB004系列器件在选定输入引脚的电平状态变化时向处理器发出中断请求。即便是在时钟被禁止的休眠模式下，该特性也可检测到输入电平状态变化（Change-Of-States，COS）事件。根据器件的引脚数，最多可以选择（允许）29个外部引脚在输入电平状态发生变化时产生中断请求。

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电路。弱上拉电路充当连接到该引脚的电流源。这样的话，当连接按钮或键盘设备时，不再需要使用外部电阻。使用CNPU1 和CNPU2 寄存器可单独使能上拉电路。每个CN 引脚都为其上拉电路提供了独立的控制位。将某个控制位置1 可使能其对应引脚的弱上拉电路。

选择内部上拉电路时，引脚将拉至 VDD – 0.7V（典型值）。确保在使能了内部上拉电路时无外部上拉源，因为电压差会导致电流通路。

**注：** 只要端口引脚被配置为数字输出引脚，电平变化通知引脚上的弱上拉电路将始终被禁止。

## 10.4 外设引脚选择 (PPS)

通用器件的最大挑战是在最小化 I/O 引脚上功能冲突的同时，提供最大可能的外设功能集。在需要使用复用一个引脚的多个外设的应用中，对应用代码进行繁琐的更改或彻底重新设计可能是惟一的选择。

外设引脚选择功能通过使能用户外设集选择并将外设功能放置到大量的 I/O 引脚中提供了一个替代这两种选择的方案。通过增加特定器件上的引脚配置选项，用户可以更好地调节单片机以满足整个应用的需要，而不是调整应用来满足器件。

外设引脚选择功能通过固定数量的数字 I/O 引脚进行操作。用户可将任一数字外设的输入和 / 或输出独立映射到这些 I/O 引脚之一。外设引脚选择通过软件执行，通

常不需要器件重新编程。器件带有保护硬件，可防止在建立了外设映射后意外或误更改此映射。

### 10.4.1 可用引脚

外设引脚选择功能可在最多 25 个引脚的范围内使用；可用的引脚数取决于特定器件及其引脚数。支持外设引脚选择功能的引脚在其完整引脚名称中有“R<sub>Pn</sub>”标识，其中“n”指的是可重映射的引脚的编号。

请参见表 1-2，了解各封装提供的引脚排列选项。

### 10.4.2 可用外设

外设引脚选择管理的外设都是仅数字外设，包括通用串行通信（UART 和 SPI）、通用定时器时钟输入、定时器相关外设（输入捕捉和输出比较）以及外部中断输入。由于比较器模块的输出为离散数字信号，因此也包括在内。

外设引脚选择不适用于 I<sup>2</sup>C<sup>TM</sup>、电平变化通知输入、RTCC 闹钟输出或具有模拟输入的外设。

有引脚选择和无引脚选择的外设的最大区别是有引脚选择的外设与默认 I/O 引脚之间无关联。必须在使用外设前将其分配给指定的 I/O 引脚。相反，假设无引脚选择的外设处于激活状态且未与其他外设发生冲突，则此外设可始终通过默认引脚使用。

#### 10.4.2.1 外设引脚选择功能优先级

可选引脚的外设的输出（例如 OC 和 UART 发送）的优先级高于与该引脚永久连接的所有通用数字功能（例如 PMP 和端口 I/O）。特殊的数字输出（例如 USB 功能）的优先级高于同一引脚上的 PPS 输出。在本数据手册开头部分的引脚框图按照优先级顺序列出了外设输出。请参见这些引脚框图以了解关于某个特定引脚的优先级信息。

与具有固定外设的器件不同，可选引脚的外设输入从不具有该引脚的所有权。引脚的输出缓冲器由该引脚的 TRIS 位设置控制，或由该引脚上的固定外设控制。如果引脚配置为数字模式，则 PPS 输入正常工作，读到的是输入值。如果使能了同一引脚上的模拟功能，则禁止可选引脚输入。

## 10.4.3 控制外设引脚选择

外设引脚选择功能是通过以下两组特殊功能寄存器控制的：一组用于映射外设输入，另一组用于映射输出。由于输入和输出是单独控制的，因此特定外设的输入和输出（若该外设都有）均可施加到任何可选的功能引脚上，而没有限制。

根据映射的是输入还是输出，有两种方法可处理外设与外设可选引脚之间的关联。

### 10.4.3.1 输入映射

外设引脚选择选项的输入根据外设进行映射；即，与外设相关的控制寄存器指示要映射到的引脚。RPINRx 寄存器用于配置外设输入映射（见寄存器 10-1 至寄存器 10-14）。各寄存器最多包含两组 5 位位域，每组都与一个可选引脚的外设相关。给指定外设的位域赋上正确的 6 位值，会将具有此值的 RPin 引脚映射到该外设。对于任何给定器件，任何位域值的有效范围对应于此器件所支持的外设引脚选择的最大值。

表 10-2： 可选输入源（将输入映射到功能）<sup>(1)</sup>

输入名称	功能名称	寄存器	功能映射位
外部中断 1	INT1	RPINR0	INT1R<5:0>
外部中断 2	INT2	RPINR1	INT2R<5:0>
输入捕捉 1	IC1	RPINR7	IC1R<5:0>
输入捕捉 2	IC2	RPINR7	IC2R<5:0>
输入捕捉 3	IC3	RPINR8	IC3R<5:0>
输入捕捉 4	IC4	RPINR8	IC4R<5:0>
输入捕捉 5	IC5	RPINR9	IC5R<5:0>
输出比较故障 A	OCFA	RPINR11	OCFAR<5:0>
输出比较故障 B	OCFB	RPINR11	OCFBR<5:0>
SPI1 时钟输入	SCK1IN	RPINR20	SCK1R<5:0>
SPI1 数据输入	SDI1	RPINR20	SDI1R<5:0>
SPI1 从选择输入	SS1IN	RPINR21	SS1R<5:0>
SPI2 时钟输入	SCK2IN	RPINR22	SCK2R<5:0>
SPI2 数据输入	SDI2	RPINR22	SDI2R<5:0>
SPI2 从选择输入	SS2IN	RPINR23	SS2R<5:0>
Timer2 外部时钟	T2CK	RPINR3	T2CKR<5:0>
Timer3 外部时钟	T3CK	RPINR3	T3CKR<5:0>
Timer4 外部时钟	T4CK	RPINR4	T4CKR<5:0>
Timer5 外部时钟	T5CK	RPINR4	T5CKR<5:0>
UART1 允许发送	U1CTS	RPINR18	U1CTSR<5:0>
UART1 接收	U1RX	RPINR18	U1RXR<5:0>
UART2 允许发送	U2CTS	RPINR19	U2CTSR<5:0>
UART2 接收	U2RX	RPINR19	U2RXR<5:0>

注 1：除非另外声明，否则所有输入均使用施密特触发器输入缓冲器。

### 10.4.3.2 输出映射

与输入相反，外设引脚选择选项的输出根据引脚进行映射。这种情况下，与特定引脚相关的控制寄存器指示要映射的外设输出。RPORx 寄存器用于控制输出映射。各寄存器最多包含两组 5 位位域；各位域都与一个 RPn 引脚相关（见寄存器 10-15 至寄存器 10-27）。位域的值与一个外设相对应，该外设的输出映射到 RPn 引脚（见表 10-3）。

由于采用的映射技术，输出映射的外设列表中还包含一个空值 000000。此值允许任何给定引脚始终与所有可选引脚的外设的输出断开。

**表 10-3： 可选输出源（将功能映射到输出）**

输出功能编号 <sup>(1)</sup>	功能	输出名称
0	NULL <sup>(2)</sup>	空
1	C1OUT	比较器 1 输出
2	C2OUT	比较器 2 输出
3	U1TX	UART1 发送
4	U1RTS <sup>(3)</sup>	UART1 请求发送
5	U2TX	UART2 发送
6	U2RTS <sup>(3)</sup>	UART2 请求发送
7	SDO1	SPI1 数据输出
8	SCK1OUT	SPI1 时钟输出
9	SS1OUT	SPI1 从选择输出
10	SDO2	SPI2 数据输出
11	SCK2OUT	SPI2 时钟输出
12	SS2OUT	SPI2 从选择输出
18	OC1	输出比较 1
19	OC2	输出比较 2
20	OC3	输出比较 3
21	OC4	输出比较 4
22	OC5	输出比较 5
23-28	(未用)	NC
29	CTPLS	CTMU 输出脉冲
30	C3OUT	比较器 3 输出
31	(未用)	NC

**注 1：** 使用列出的值设置 RPORx 寄存器会将输出功能指定到相应的 RPn 引脚。

**2：** 在器件复位时将 NULL 功能分配给所有的 RPn 输出，并禁止 RPn 输出功能。

**3：** IrDA® BCLK 功能使用此输出。

### 10.4.3.3 映射限制

外设引脚选择的控制模式相当灵活。除阻止由两个配置为相同输入功能的物理引脚或配置给同一引脚的两个输出功能引起的信号冲突的系统电路外，无其他依靠硬件的输出锁定电路。此灵活性可扩展到允许一个输入驱动多个外设或一个功能输出驱动多个输出引脚。

### 10.4.3.4 PIC24FJ64GB0 系列器件的 PPS 映射例外

尽管 PPS 寄存器可具有最多 32 个可重映射引脚，但不是所有这些引脚在所有器件中都实现。在表 10-4 中列出了可重映射引脚例外和未实现的 RPn 引脚。

**表 10-4：PIC24FJ64GB004 系列器件的可重映射引脚例外**

器件引脚数	RP 引脚数 (I/O)	
	总共	未实现
28 引脚	15	RP12 和 RP16-RP25
44 引脚	25	RP12

### 10.4.4 控制配置更改

由于可在运行时更改外设重映射，因此需要对外设重映射设置一些限制条件以阻止意外更改配置。PIC24F 器件有以下三种用于阻止更改外设映射的功能：

- 控制寄存器锁定序列
- 连续状态监视
- 配置位重映射锁定

### 10.4.4.1 控制寄存器锁定

正常工作状态下，不允许写 RPINRx 和 RPORx 寄存器。尝试的写操作看似正常执行，但寄存器的内容并没有发生变化。要更改这些寄存器的内容，寄存器必须用硬件解锁。寄存器锁定由 IOLOCK 位 (OSCCON<6>) 控制。将 IOLOCK 置 1 将阻止写入控制寄存器；而将 IOLOCK 清零将允许写入。

要置 1 或清零 IOLOCK，必须执行以下指定命令序列：

1. 将 46h 写入 OSCCON<7:0>。
2. 将 57h 写入 OSCCON<7:0>。
3. 通过一次操作清零（或置 1）IOLOCK。

与振荡器 LOCK 位的类似序列不同，IOLOCK 在更改前一直保持一种状态。这允许使用一个解锁序列对所有外设引脚选择进行配置，然后对所有控制寄存器进行更新，最后用第二个锁定序列锁定。

### 10.4.4.2 连续状态监视

除了阻止直接写入外，RPINRx 和 RPORx 寄存器的内容还由影子寄存器通过硬件不停地进行监视。如果任何寄存器发生了不希望的更改（例如，由 ESD 或其他外部事件引起的电池干扰），则将触发配置失配复位 (Configuration Mismatch Reset)。

### 10.4.4.3 配置位引脚选择锁定

作为进一步保护，可配置器件以阻止对 RPINRx 和 RPORx 寄存器执行多次写会话。IOL1WAY(CW2<4>) 配置位会阻止 IOLOCK 位在置 1 后被清零。若 IOLOCK 保持置 1 状态，寄存器解锁过程将不会执行，且不能写入外设引脚选择控制寄存器。清零该位并重新使能外设重映射的唯一方法是执行器件复位。

默认（未编程）状态下，IOL1WAY 置 1，限制用户只能进行一次写会话。编程 IOL1WAY 允许用户对外设引脚选择寄存器进行不受限制的访问（通过正确地使用解锁序列）。

## 10.4.5 外设引脚选择注意事项

控制外设引脚选择的功能需要注意应用设计中几个容易被忽视的事项。对于仅可用作可重映射外设的几个常用外设尤其如此。

主要问题是在器件默认（复位）状态下外设引脚选择功能在默认引脚上不可用。由于所有 RPINRx 寄存器复位为 11111，且所有 RPORx 寄存器复位为 00000，因此所有外设引脚选择输入与 Vss 相连，而所有外设引脚选择输出断开连接。

**注：** 器件上不必具有 RP31，这是因为寄存器复位会实现这一设置，外设引脚输出会连接到 RP31。

此情景需要用户在执行任何其他应用代码前使用正确的外设配置初始化器件。由于 IOLOCK 位复位为解锁状态，所以不必在器件退出复位后执行解锁序列。但是，考虑到应用的安全性，最好在写入控制寄存器之后将 IOLOCK 位置 1 并锁定配置。

由于解锁序列对时序有严格要求，因此必须作为汇编语言程序以与更改振荡器配置相同的方式执行。若应用程序是用 C 或其他高级语言编写的，则仍应通过编写行内汇编代码执行解锁序列。

选择配置需要查看所有外设引脚选择及其引脚分配，尤其是那些未在应用中使用的引脚。在所有情况下，未用的可选引脚的外设应被完全禁止。未用外设的输入应分配给未用的 RPn 引脚功能。具有未用的 RPn 功能的 I/O 引脚应配置为空外设输出。

将外设分配给特定引脚的操作不能自动执行引脚 I/O 电路的任何其他配置。理论上，也就是说将可选引脚的输出添加到某个引脚意味着在驱动输出时可能会无意识地驱动现有的外设输入。用户必须熟悉其他共用一个可重映射引脚的固定外设的行为，了解何时使能或禁止这些外设。为了安全起见，共用一个引脚的固定数字外设在不用时应禁止。

遵照这些方针，对特定外设的可重映射的引脚进行配置并不会自动使能对应的外设。必须特别配置外设实现特定的操作并使能外设，如同将外设连接到固定引脚时一样。此操作在应用代码中的位置（紧跟在器件复位和外设配置之后或在主应用程序内）取决于外设及其在应用中的使用。

最后一个注意事项是外设引脚选择功能既不改写模拟输入也不会将具有模拟功能的引脚重新配置为数字 I/O。若某个引脚在器件复位时配置为模拟输入，则在使用该引脚的外设引脚选择功能时必须将此引脚明确配置为数字 I/O。

例 10-2 所示为使用 UART1 实现带有流控制的双向通信的配置。使用了以下输入和输出功能：

- 输入功能：U1RX 和 U1CTS
- 输出功能：U1TX 和 U1RTS

## 例 10-2：配置 UART1 输入和输出功能

```
// Unlock Registers
asm volatile ("MOV #OSCCON, w1\n"
              "MOV #0x46, w2\n"
              "MOV #0x57, w3\n"
              "MOV.b w2, [w1]\n"
              "MOV.b w3, [w1]\n"
              "BCLR OSCCON,#6");

// Configure Input Functions (Table 9-1)
// Assign U1RX To Pin RP0
RPINR18bits.U1RXR = 0;

// Assign U1CTS To Pin RP1
RPINR18bits.U1CTSR = 1;

// Configure Output Functions (Table 9-2)
// Assign U1TX To Pin RP2
RPOR1bits.RP2R = 3;

// Assign U1RTS To Pin RP3
RPOR1bits.RP3R = 4;

// Lock Registers
asm volatile ("MOV #OSCCON, w1\n"
              "MOV #0x46, w2\n"
              "MOV #0x57, w3\n"
              "MOV.b w2, [w1]\n"
              "MOV.b w3, [w1]\n"
              "BSET OSCCON, #6");
```

# PIC24FJ64GB004 系列

## 10.4.6 外设引脚选择寄存器

PIC24FJ64GB004 系列器件共实现了 27 个寄存器用于配置可重映射的外设：

- 输入可重映射外设寄存器 (14)
- 输出可重映射外设寄存器 (13)

**注：** 仅在 IOLOCK (OSCCON<6>) = 0 时才能更改输入和输出寄存器的值。请参见第 10.4.4.1 节“控制寄存器锁定”了解特定的命令序列。

### 寄存器 10-1：RPINR0：外设引脚选择输入寄存器 0

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	INT1R4	INT1R3	INT1R2	INT1R1	INT1R0
bit 15							bit 8

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

#### 图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现：读为 0

bit 12-8 INT1R<4:0>：将外部中断 1 (INT1) 分配给相应的 RPn 或 RPIn 引脚的位

bit 7-0 未实现：读为 0

### 寄存器 10-2：RPINR1：外设引脚选择输入寄存器 1

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-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	INT2R4	INT2R3	INT2R2	INT2R1	INT2R0
bit 7							bit 0

#### 图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-5 未实现：读为 0

bit 4-0 INT2R<4:0>：将外部中断 2 (INT2) 分配给相应的 RPn 或 RPIn 引脚的位

# PIC24FJ64GB004 系列

## 寄存器 10-3 : RPINR3 : 外设引脚选择输入寄存器 3

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T3CKR4	T3CKR3	T3CKR2	T3CKR1	T3CKR0
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T2CKR4	T2CKR3	T2CKR2	T2CKR1	T2CKR0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 T3CKR<4:0> : 将 Timer3 外部时钟 (T3CK) 分配给相应的 RPn 或 RPIn 引脚的位

bit 7-5 未实现 : 读为 0

bit 4-0 T2CKR<4:0> : 将 Timer2 外部时钟 (T2CK) 分配给相应的 RPn 或 RPIn 引脚的位

## 寄存器 10-4 : RPINR4 : 外设引脚选择输入寄存器 4

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T5CKR4	T5CKR3	T5CKR2	T5CKR1	T5CKR0
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	T4CKR4	T4CKR3	T4CKR2	T4CKR1	T4CKR0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 T5CKR<4:0> : 将 Timer5 外部时钟 (T5CK) 分配给相应的 RPn 或 RPIn 引脚的位

bit 7-5 未实现 : 读为 0

bit 4-0 T4CKR<4:0> : 将 Timer4 外部时钟 (T4CK) 分配给相应的 RPn 或 RPIn 引脚的位

# PIC24FJ64GB004 系列

## 寄存器 10-5 : RPINR7 : 外设引脚选择输入寄存器 7

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC2R4	IC2R3	IC2R2	IC2R1	IC2R0
bit 15				bit 8			

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC1R4	IC1R3	IC1R2	IC1R1	IC1R0
bit 7				bit 0			

### 图注 :

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

- bit 15-13 未实现 : 读为 0  
bit 12-8 IC2R<4:0> : 将输入捕捉 2 ( IC2 ) 分配给相应的 RPn 或 RPIn 引脚的位  
bit 7-5 未实现 : 读为 0  
bit 4-0 IC1R<4:0> : 将输入捕捉 1 ( IC1 ) 分配给相应的 RPn 或 RPIn 引脚的位

## 寄存器 10-6 : RPINR8 : 外设引脚选择输入寄存器 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC4R4	IC4R3	IC4R2	IC4R1	IC4R0
bit 15				bit 8			

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC3R4	IC3R3	IC3R2	IC3R1	IC3R0
bit 7				bit 0			

### 图注 :

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

- bit 15-13 未实现 : 读为 0  
bit 12-8 IC4R<4:0> : 将输入捕捉 4 ( IC4 ) 分配给相应的 RPn 或 RPIn 引脚的位  
bit 7-5 未实现 : 读为 0  
bit 4-0 IC3R<4:0> : 将输入捕捉 3 ( IC3 ) 分配给相应的 RPn 或 RPIn 引脚的位

## 寄存器 10-7 : RPINR9 : 外设引脚选择输入寄存器 9

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-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	IC5R4	IC5R3	IC5R2	IC5R1	IC5R0
bit 7							bit 0

### 图注 :

R = 可读位

-n = 上电复位时的值

W = 可写位

1 = 置 1

U = 未实现位 , 读为 0

0 = 清零

x = 未知

bit 15-5 未实现 : 读为 0

bit 4-0 IC5R<4:0> : 将输入捕捉 5 ( IC5 ) 分配给相应的 RPn 或 RPIn 引脚的位

## 寄存器 10-8 : RPINR11 : 外设引脚选择输入寄存器 11

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	OCFB4	OCFB3	OCFB2	OCFB1	OCFB0
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	OCFA4	OCFA3	OCFA2	OCFA1	OCFA0
bit 7							bit 0

### 图注 :

R = 可读位

-n = 上电复位时的值

W = 可写位

1 = 置 1

U = 未实现位 , 读为 0

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 OCFBR<4:0> : 将输出比较故障 B ( OCFB ) 分配给相应的 RPn 或 RPIn 引脚的位

bit 7-5 未实现 : 读为 0

bit 4-0 OCFA<4:0> : 将输出比较故障 A ( OCFA ) 分配给相应的 RPn 或 RPIn 引脚的位

# PIC24FJ64GB004 系列

寄存器 10-9 : RPINR18 : 外设引脚选择输入寄存器 18

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	U1CTSR4	U1CTSR3	U1CTSR2	U1CTSR1	U1CTSR0
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	U1RXR4	U1RXR3	U1RXR2	U1RXR1	U1RXR0
bit 7							bit 0

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现：读为 0

bit 12-8 U1CTSR<4:0> : 将 UART1 允许发送 ( $\overline{U1CTS}$ ) 分配给相应的 RPn 或 RPIn 引脚的位

bit 7-5 未实现：读为 0

bit 4-0 U1RXR<4:0> : 将 UART1 接收 (U1RX) 分配给相应的 RPn 或 RPIn 引脚的位

寄存器 10-10 : RPINR19 : 外设引脚选择输入寄存器 19

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	U2CTSR4	U2CTSR3	U2CTSR2	U2CTSR1	U2CTSR0
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	U2RXR4	U2RXR3	U2RXR2	U2RXR1	U2RXR0
bit 7							bit 0

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现：读为 0

bit 12-8 U2CTSR<4:0> : 将 UART2 允许发送 ( $\overline{U2CTS}$ ) 分配给相应的 RPn 或 RPIn 引脚的位

bit 7-5 未实现：读为 0

bit 4-0 U2RXR<4:0> : 将 UART2 接收 (U2RX) 分配给相应的 RPn 或 RPIn 引脚的位

## 寄存器 10-11 : RPINR20 : 外设引脚选择输入寄存器 20

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SCK1R4	SCK1R3	SCK1R2	SCK1R1	SCK1R0
bit 15						bit 8	

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SDI1R4	SDI1R3	SDI1R2	SDI1R1	SDI1R0
bit 7						bit 0	

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 SCK1R<4:0> : 将 SPI1 时钟输入 ( SCK1IN ) 分配给相应的 RPn 或 RPIn 引脚的位

bit 7-5 未实现 : 读为 0

bit 4-0 SDI1R<4:0> : 将 SPI1 数据输入 ( SDI1 ) 分配给相应的 RPn 或 RPIn 引脚的位

## 寄存器 10-12 : RPINR21 : 外设引脚选择输入寄存器 21

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-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SS1R4	SS1R3	SS1R2	SS1R1	SS1R0
bit 7						bit 0	

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-5 未实现 : 读为 0

bit 4-0 SS1R<4:0> : 将 SPI1 从选择输入 ( SS1IN ) 分配给相应的 RPn 或 RPIn 引脚的位

# PIC24FJ64GB004 系列

## 寄存器 10-13 : RPINR22 : 外设引脚选择输入寄存器 22

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SCK2R4	SCK2R3	SCK2R2	SCK2R1	SCK2R0
bit 15							bit 8

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SDI2R4	SDI2R3	SDI2R2	SDI2R1	SDI2R0
bit 7							bit 0

### 图注 :

R = 可读位                    W = 可写位                    U = 未实现位 , 读为 0

-n = 上电复位时的值        1 = 置 1                    0 = 清零                    x = 未知

bit 15-13      未实现 : 读为 0

bit 12-8        SCK2R<4:0> : 将 SPI2 时钟输入 ( SCK2IN ) 分配给相应的 RPn 或 RPIn 引脚的位

bit 7-5        未实现 : 读为 0

bit 4-0        SDI2R<4:0> : 将 SPI2 数据输入 ( SDI2 ) 分配给相应的 RPn 或 RPIn 引脚的位

## 寄存器 10-14 : RPINR23 : 外设引脚选择输入寄存器 23

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-1	R/W-1	R/W-1	R/W-1	R/W-1
—	—	—	SS2R4	SS2R3	SS2R2	SS2R1	SS2R0
bit 7							bit 0

### 图注 :

R = 可读位                    W = 可写位                    U = 未实现位 , 读为 0

-n = 上电复位时的值        1 = 置 1                    0 = 清零                    x = 未知

bit 15-5      未实现 : 读为 0

bit 4-0        SS2R<4:0> : 将 SPI2 从选择输入 ( SS2IN ) 分配给相应的 RPn 或 RPIn 引脚的位

## 寄存器 10-15 : RPOR0 : 外设引脚选择输出寄存器 0

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP1R4	RP1R3	RP1R2	RP1R1	RP1R0
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
—	—	—	RP0R4	RP0R3	RP0R2	RP0R1	RP0R0
bit 7							bit 0

**图注 :**

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP1R<4:0> : RP1 输出引脚映射位

将外设输出编号 n 分配给引脚 RP1 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP0R<4:0> : RP0 输出引脚映射位

将外设输出编号 n 分配给引脚 RP0 (请参见表 10-3 了解外设功能编号)

## 寄存器 10-16 : RPOR1 : 外设引脚选择输出寄存器 1

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP3R4	RP3R3	RP3R2	RP3R1	RP3R0
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
—	—	—	RP2R4	RP2R3	RP2R2	RP2R1	RP2R0
bit 7							bit 0

**图注 :**

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP3R<4:0> : RP3 输出引脚映射位

将外设输出编号 n 分配给引脚 RP3 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP2R<4:0> : RP2 输出引脚映射位

将外设输出编号 n 分配给引脚 RP2 (请参见表 10-3 了解外设功能编号)

# PIC24FJ64GB004 系列

## 寄存器 10-17 : RPOR2 : 外设引脚选择输出寄存器 2

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP5R4	RP5R3	RP5R2	RP5R1	RP5R0
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
—	—	—	RP4R4	RP4R3	RP4R2	RP4R1	RP4R0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP5R<4:0> : RP5 输出引脚映射位<sup>(1)</sup>

将外设输出编号 n 分配给引脚 RP5 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP4R<4:0> : RP4 输出引脚映射位

将外设输出编号 n 分配给引脚 RP4 (请参见表 10-3 了解外设功能编号)

## 寄存器 10-18 : RPOR3 : 外设引脚选择输出寄存器 3

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP7R4	RP7R3	RP7R2	RP7R1	RP7R0
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
—	—	—	RP6R4	RP6R3	RP6R2	RP6R1	RP6R0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

X = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP7R<4:0> : RP7 输出引脚映射位

将外设输出编号 n 分配给引脚 RP7 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP6R<4:0> : RP6 输出引脚映射位

将外设输出编号 n 分配给引脚 RP6 (请参见表 10-3 了解外设功能编号)

## 寄存器 10-19 : RPOR4 : 外设引脚选择输出寄存器 4

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP9R4	RP9R3	RP9R2	RP9R1	RP9R0
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
—	—	—	RP8R4	RP8R3	RP8R2	RP8R1	RP8R0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP9R<4:0> : RP9 输出引脚映射位

将外设输出编号 n 分配给引脚 RP9 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP8R<4:0> : RP8 输出引脚映射位

将外设输出编号 n 分配给引脚 RP8 (请参见表 10-3 了解外设功能编号)

## 寄存器 10-20 : RPOR5 : 外设引脚选择输出寄存器 5

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP11R4	RP11R3	RP11R2	RP11R1	RP11R0
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
—	—	—	RP10R4	RP10R3	RP10R2	RP10R1	RP10R0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP11R<4:0> : RP11 输出引脚映射位

将外设输出编号 n 分配给引脚 RP11 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP10R<4:0> : RP10 输出引脚映射位

将外设输出编号 n 分配给引脚 RP10 (请参见表 10-3 了解外设功能编号)

# PIC24FJ64GB004 系列

## 寄存器 10-21 : RPOR6 : 外设引脚选择输出寄存器 6

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP13R4	RP13R3	RP13R2	RP13R1	RP13R0
bit 15	bit 8						

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

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP13R<4:0> : RP13 输出引脚映射位

将外设输出编号 n 分配给引脚 RP13 (请参见表 10-3 了解外设功能编号)

bit 7-0 未实现 : 读为 0

## 寄存器 10-22 : RPOR7 : 外设引脚选择输出寄存器 7

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP15R4	RP15R3	RP15R2	RP15R1	RP15R0
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
—	—	—	RP14R4	RP14R3	RP14R2	RP14R1	RP14R0
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

X = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP15R<4:0> : RP15 输出引脚映射位 (1)

将外设输出编号 n 分配给引脚 RP15 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP14R<4:0> : RP14 输出引脚映射位

将外设输出编号 n 分配给引脚 RP14 (请参见表 10-3 了解外设功能编号)

## 寄存器 10-23 : RPOR8 : 外设引脚选择输出寄存器 8<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP17R4	RP17R3	RP17R2	RP17R1	RP17R0
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
—	—	—	RP16R4	RP16R3	RP16R2	RP16R1	RP16R0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP17R<4:0> : RP17 输出引脚映射位

将外设输出编号 n 分配给引脚 RP17 ( 请参见表 10-3 了解外设功能编号 )

bit 7-5 未实现 : 读为 0

bit 4-0 RP16R<4:0> : RP16 输出引脚映射位

将外设输出编号 n 分配给引脚 RP16 ( 请参见表 10-3 了解外设功能编号 )

注 1 : 该寄存器在 28 引脚器件中未实现 ; 所有位读为 0。

## 寄存器 10-24 : RPOR9 : 外设引脚选择输出寄存器 9<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP19R4	RP19R3	RP19R2	RP19R1	RP19R0
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
—	—	—	RP18R4	RP18R3	RP18R2	RP18R1	RP18R0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP19R<4:0> : RP19 输出引脚映射位

将外设输出编号 n 分配给引脚 RP19 ( 请参见表 10-3 了解外设功能编号 )

bit 7-5 未实现 : 读为 0

bit 4-0 RP18R<4:0> : RP18 输出引脚映射位

将外设输出编号 n 分配给引脚 RP18 ( 请参见表 10-3 了解外设功能编号 )

注 1 : 该寄存器在 28 引脚器件中未实现 ; 所有位读为 0。

# PIC24FJ64GB004 系列

寄存器 10-25 : RPOR10 : 外设引脚选择输出寄存器 10<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP21R4	RP21R3	RP21R2	RP21R1	RP21R0
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
—	—	—	RP20R4	RP20R3	RP20R2	RP20R1	RP20R0
bit 7							bit 0

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP21R<4:0> : RP21 输出引脚映射位

将外设输出编号 n 分配给引脚 RP21 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP20R<4:0> : RP20 输出引脚映射位

将外设输出编号 n 分配给引脚 RP20 (请参见表 10-3 了解外设功能编号)

注 1 : 该寄存器在 28 引脚器件中未实现 ; 所有位读为 0。

寄存器 10-26 : RPOR11 : 外设引脚选择输出寄存器 11<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP23R4	RP23R3	RP23R2	RP23R1	RP23R0
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
—	—	—	RP22R4	RP22R3	RP22R2	RP22R1	RP22R0
bit 7							bit 0

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

X = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP23R<4:0> : RP23 输出引脚映射位

将外设输出编号 n 分配给引脚 RP23 (请参见表 10-3 了解外设功能编号)

bit 7-5 未实现 : 读为 0

bit 4-0 RP22R<4:0> : RP22 输出引脚映射位

将外设输出编号 n 分配给引脚 RP22 (请参见表 10-3 了解外设功能编号)

注 1 : 该寄存器在 28 引脚器件中未实现 ; 所有位读为 0。

## 寄存器 10-27 : RPOR12 : 外设引脚选择输出寄存器 12<sup>(1)</sup>

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	RP25R4	RP25R3	RP25R2	RP25R1	RP25R0
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
—	—	—	RP24R4	RP24R3	RP24R2	RP24R1	RP24R0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 RP25R<5:0> : RP25 输出引脚映射位

将外设输出编号 n 分配给引脚 RP25 ( 请参见表 10-3 了解外设功能编号 )

bit 7-5 未实现 : 读为 0

bit 4-0 RP24R<5:0> : RP24 输出引脚映射位

将外设输出编号 n 分配给引脚 RP24 ( 请参见表 10-3 了解外设功能编号 )

注 1 : 该寄存器在 28 引脚器件中未实现 ; 所有位读为 0。

# **PIC24FJ64GB004 系列**

---

---

注：

## 11.0 TIMER1

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 14 章 “Timers” (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 还支持下列特性：

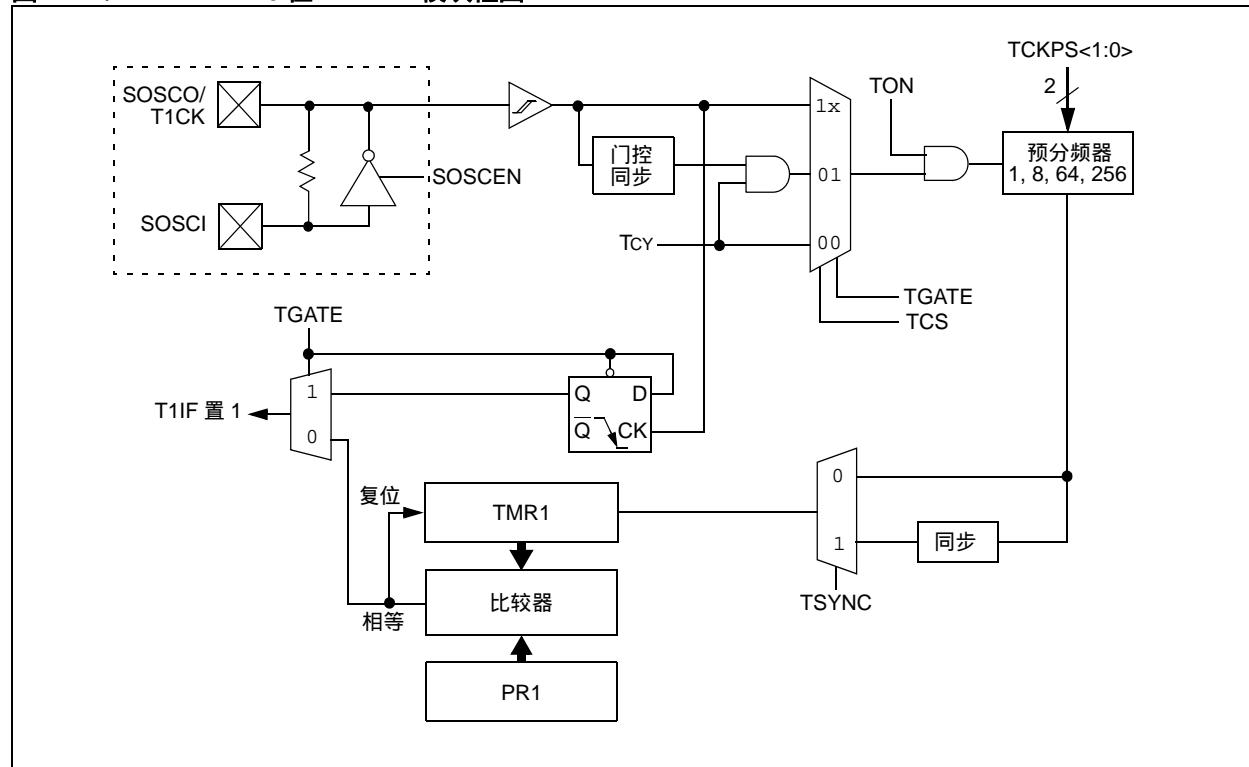
- 定时器门控操作
- 可选的预分频比设置
- CPU 空闲和休眠模式下的定时器操作
- 在 16 位周期寄存器匹配时或外部门控信号的下降沿产生中断

图 11-1 给出了 16 位定时器模块的框图。

配置 Timer1 的步骤：

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 或清零以配置同步或异步操作。
5. 将定时器周期值装入 PR1 寄存器。
6. 如果需要中断，则将中断允许位 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



# PIC24FJ64GB004 系列

寄存器 11-1 : T1CON : TIMER1 控制寄存器<sup>(1)</sup>

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

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

**图注 :**

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15      **TON** : Timer1 启动位  
1 = 启动 16 位 Timer1  
0 = 停止 16 位 Timer1
- bit 14      未实现 : 读为 0
- bit 13      **TSIDL** : 空闲模式停止位  
1 = 当器件进入空闲模式后 , 模块停止工作  
0 = 模块在空闲模式下继续工作
- bit 12-7     未实现 : 读为 0
- bit 6        **TGATE** : Timer1 门控时间累加使能位  
当 TCS = 1 时 :  
此位为无关位。  
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 :  
此位为无关位。
- bit 1        **TCS** : Timer1 时钟源选择位  
1 = T1CK 引脚的外部时钟上升沿  
0 = 内部时钟 ( Fosc/2 )
- bit 0        未实现 : 读为 0

**注 1 :** 定时器运行时 ( TON = 1 ) 更改 TxCON 的值会导致定时器预分频计数器复位 , 不推荐这么做。

## 12.0 TIMER2/3 和 TIMER4/5

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 14 章“定时器”(DS39704A\_CN)。

Timer2/3 和 Timer4/5 模块各为一个 32 位定时器，它们也可以被配置为 4 个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 和 Timer4/5 可以在以下三种模式中工作：

- 两个独立的 16 位定时器，可实现所有 16 位工作模式（异步计数器模式除外）
- 一个 32 位定时器
- 一个 32 位同步计数器

它们还支持以下功能：

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

在作为 16 位定时器时，所有 4 个定时器都可以单独用作同步定时器或计数器。它们也可以提供上面列出的功能，但触发 ADC 事件除外，它只能由 Timer5 实现。工作模式和使能的功能由在 T2CON、T3CON、T4CON 和 T5CON 寄存器中相应位的设置决定。寄存器 12-1 给出了 T2CON 和 T4CON 的一般形式；寄存器 12-2 给出了 T3CON 和 T5CON 的一般形式。

当工作在 32 位定时器/计数器模式时，Timer2 和 Timer4 作为 32 定时器的低位字，而 Timer3 和 Timer5 则作为高位字。

**注：**当定时器工作在 32 位模式时，T3CON 和 T5CON 的控制位将被忽略。只有 T2CON 和 T4CON 的控制位用于设置和控制。32 位定时器模块使用 Timer2 和 Timer4 时钟和门控输入，但中断由 Timer3 或 Timer5 的中断标志产生。

要将 Timer2/3 或 Timer4/5 配置为 32 位工作模式：

1. 将 T32 位置 1 (T2CON<3> 或 T4CON<3> = 1)。
2. 使用 TCKPS<1:0> 位为 Timer2 或 Timer4 选择预分频比。
3. 使用 TCS 和 TGATE 位设置时钟和门控模式。如果 TCS 被设置为外部时钟，则必须将 RPINRx (TxCK) 配置给可用的 RPn 引脚。更多信息，请参见第 10.4 节“外设引脚选择 (PPS)”。
4. 装入定时器周期值。PR3 (或 PR5) 将包含值的高字，而 PR2 (或 PR4) 包含低字。
5. 如果需要中断，将中断允许位 T3IE 或 T5IE 置 1，并使用优先级位 T3IP<2:0> 或 T5IP<2:0> 设置中断优先级。请注意虽然 Timer2 或 Timer4 控制定时器，但其中断却表现为 Timer3 或 Timer5 中断。
6. 将 TON 位置 1 (= 1)。

定时器的值可以随时被存储到寄存器对 TMR3:TMR2 (或 TMR5:TMR4) 中。TMR3 (TMR5) 始终存放计数值的高字，而 TMR2 (TMR4) 始终存放低字。

要将任何定时器配置为独立的 16 位工作模式：

1. 将与该定时器对应的 T32 位清零（对于 Timer2 和 Timer3 来说是 T2CON<3>；对于 Timer4 和 Timer5 来说是 T4CON<3>）。
2. 使用 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。更多信息，请参见第 10.4 节“外设引脚选择 (PPS)”。
4. 将定时器周期值装入 PRx 寄存器。
5. 如果需要中断，将中断允许位 TxIE 置 1，并使用优先级位 TxIP<2:0> 设置中断优先级。
6. 将 TON 位置 1 (TxCON<15> = 1)。

# PIC24FJ64GB004 系列

图 12-1：TIMER2/3 和 TIMER4/5 (32 位) 框图

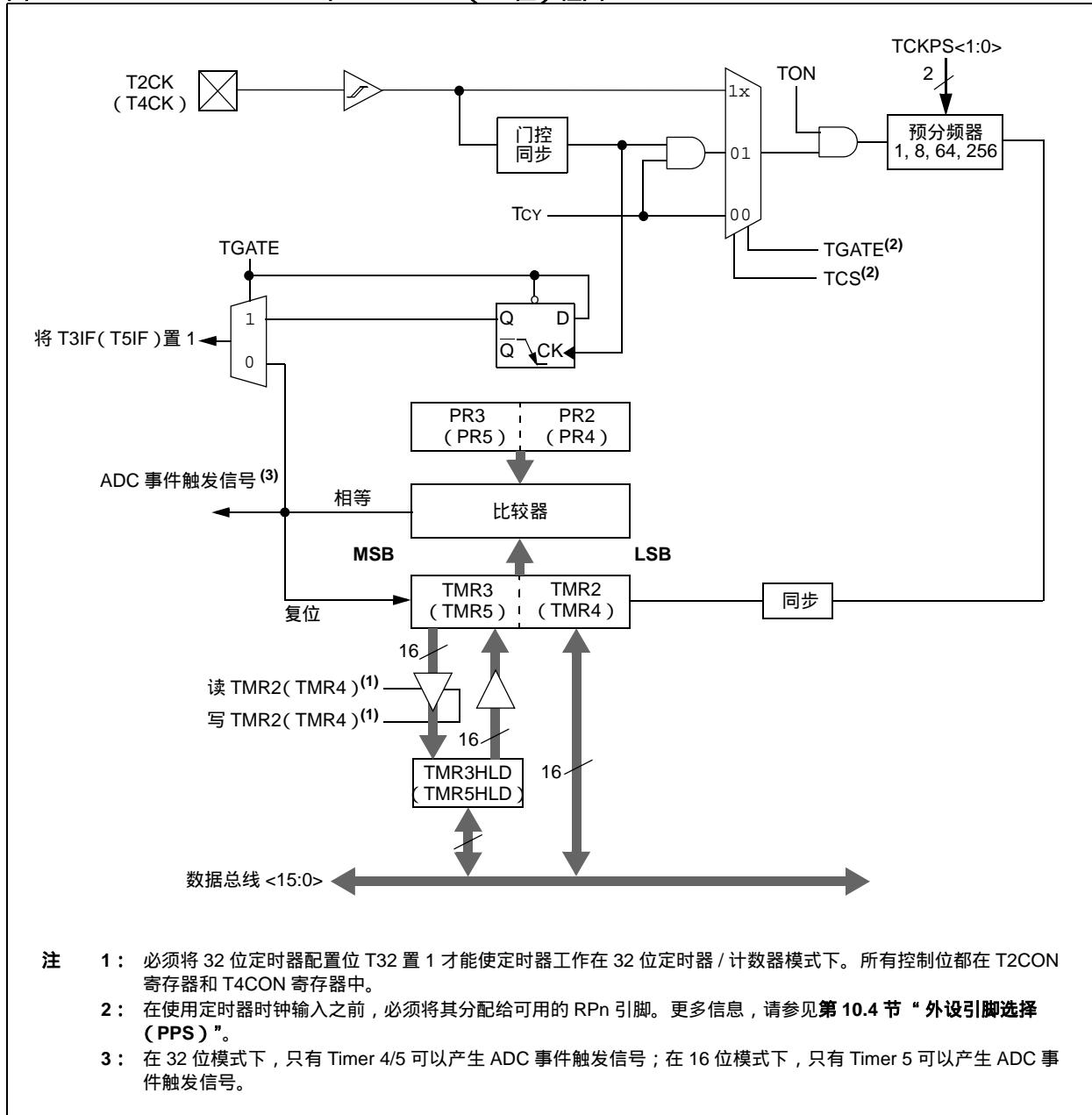


图 12-2： TIMER2 和 TIMER4 (16 位同步) 框图

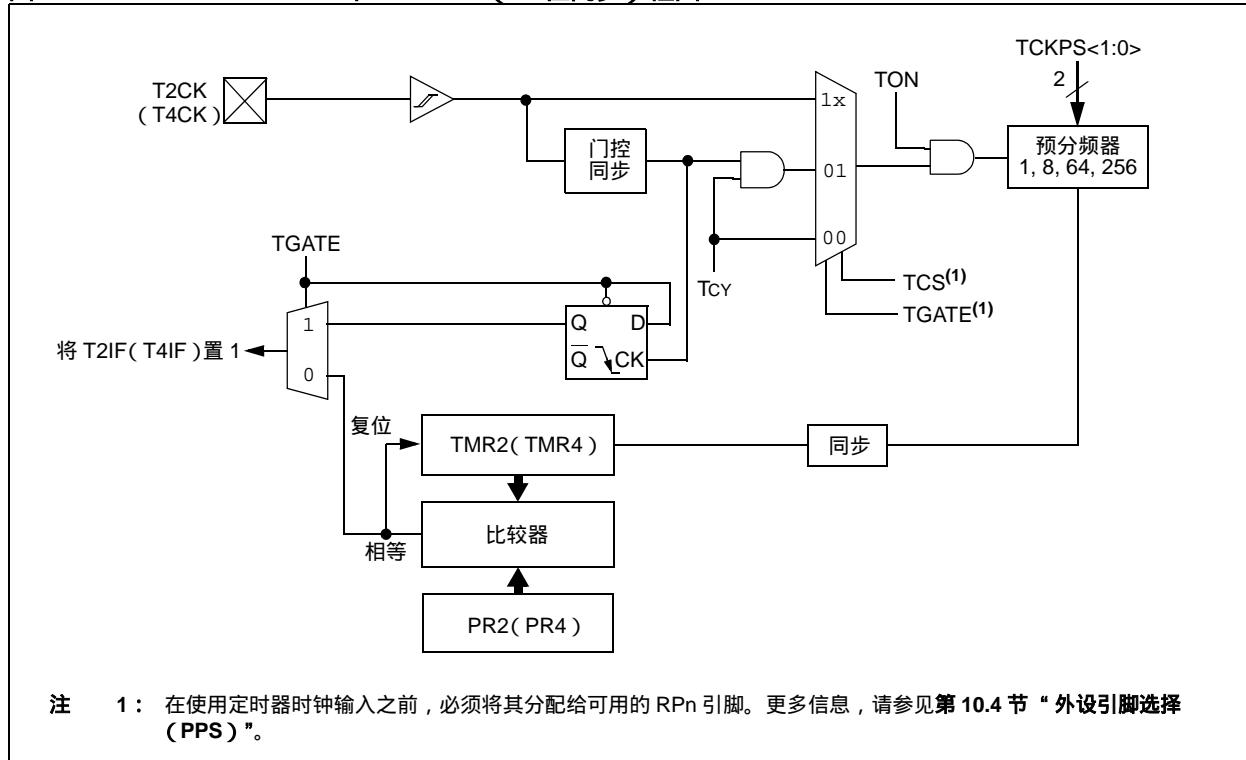
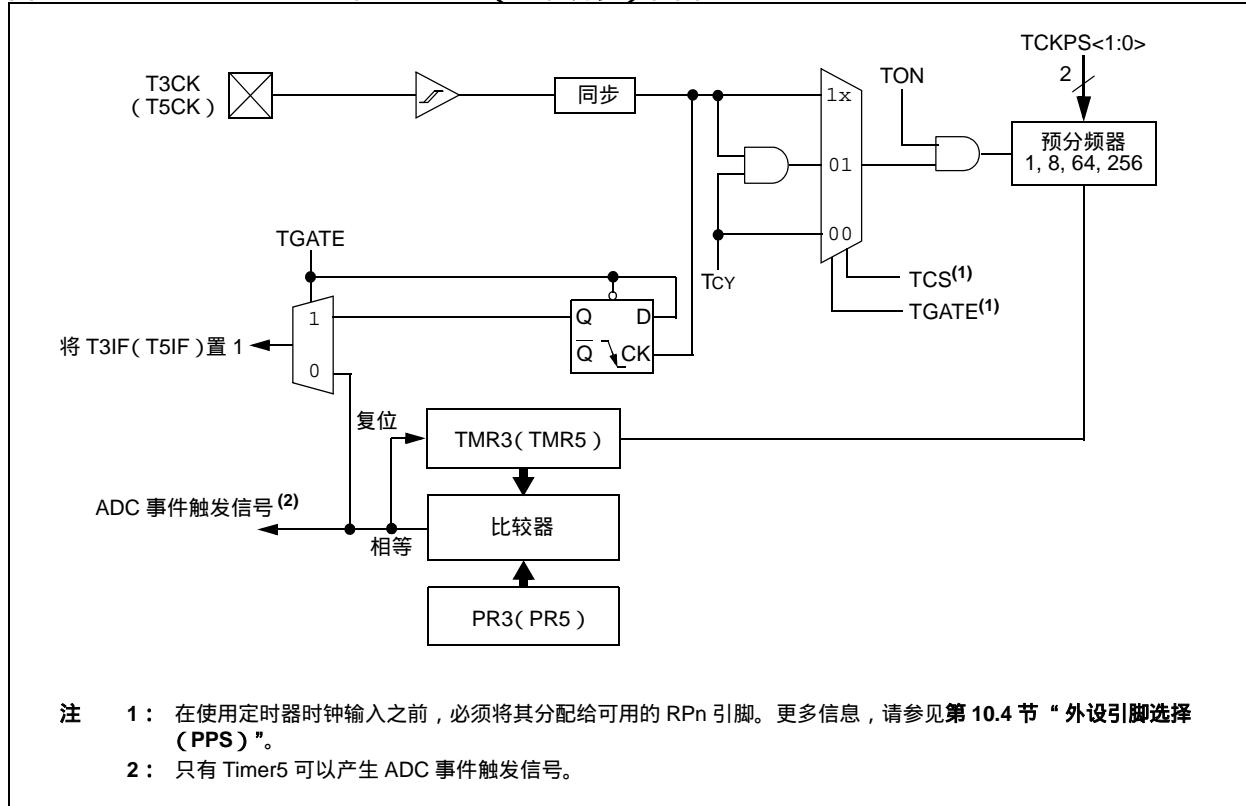


图 12-3： TIMER3 和 TIMER5 (16 位异步) 框图



# PIC24FJ64GB004 系列

寄存器 12-1 : TxCON : TIMER2 和 TIMER4 控制寄存器<sup>(3)</sup>

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 <sup>(1)</sup>	—	TCS <sup>(2)</sup>	—
bit 7	bit 0						

## 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **TON** : Timerx 使能位

当 TxCON<3> = 1 时 :

1 = 启动 32 位 Timerx/y

0 = 停止 32 位 Timerx/y

当 TxCON<3> = 0 时 :

1 = 启动 16 位 Timerx

0 = 停止 16 位 Timerx

bit 14      未实现 : 读为 0

bit 13      **TSIDL** : 空闲模式停止位

1 = 当器件进入空闲模式后 , 模块停止工作

0 = 模块在空闲模式下继续工作

bit 12-7      未实现 : 读为 0

bit 6      **TGATE** : Timerx 门控时间累加使能位

当 TCS = 1 时 :

此位为无关位。

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>** : Timerx 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<sup>(1)</sup>

1 = Timerx 和 Timery 构成一个 32 位定时器

0 = Timerx 和 Timery 作为两个 16 位定时器

在 32 位模式下 , T3CON 控制位不影响 32 位定时器的工作。

bit 2      未实现 : 读为 0

bit 1      **TCS** : Timerx 时钟源选择位<sup>(2)</sup>

1 = 来自 TxCK 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 或 T5CON 控制位不影响 32 位定时器的工作。

**2 :** 如果 TCS = 1 , 必须将 RPINRx (TxCK) 配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS)”。

**3 :** 定时器运行 (TON = 1) 时改变 TxCON 的值会导致定时器预分频计数器复位 , 不推荐这么做。

## 寄存器 12-2 : TyCON : TIMER3 和 TIMER5 控制寄存器<sup>(3)</sup>

R/W-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
TON <sup>(1)</sup>	—	TSIDL <sup>(1)</sup>	—	—	—	—	—
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 <sup>(1)</sup>	TCKPS1 <sup>(1)</sup>	TCKPS0 <sup>(1)</sup>	—	—	TCS <sup>(1,2)</sup>	—
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **TON** : Timery 使能位<sup>(1)</sup>

1 = 启动 16 位 Timery

0 = 停止 16 位 Timery

bit 14      未实现 : 读为 0

bit 13      **TSIDL** : 空闲模式停止位<sup>(1)</sup>

1 = 当器件进入空闲模式后 , 模块停止工作

0 = 模块在空闲模式下继续工作

bit 12-7    未实现 : 读为 0

bit 6        **TGATE** : Timery 门控时间累加使能位<sup>(1)</sup>

当 TCS = 1 时 :

此位为无关位。

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>** : Timery 输入时钟预分频比选择位<sup>(1)</sup>

11 = 1:256

10 = 1:64

01 = 1:8

00 = 1:1

bit 3-2      未实现 : 读为 0

bit 1        **TCS** : Timery 时钟源选择位<sup>(1,2)</sup>

1 = 来自 TyCK 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 :** 当使能 32 位工作模式时 (T3CON<3> 或 T5CON<3> = 1) , 这些位不会对 Timery 的工作产生影响。定时器的所有功能都通过 T2CON 和 T4CON 寄存器设置。

**2 :** 如果 TCS = 1 , 必须将 RPINRx (TxCK) 配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

**3 :** 定时器运行 (TON = 1) 时改变 TyCON 的值会导致定时器预分频计数器复位 , 不推荐这么做。

# **PIC24FJ64GB004 系列**

---

---

注：

## 13.0 带专用定时器的输入捕捉

**注：**本数据手册总结了该组PIC24F器件具有的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F系列参考手册》中的第34章“带专用定时器的输入捕捉”(DS39722A\_CN)。

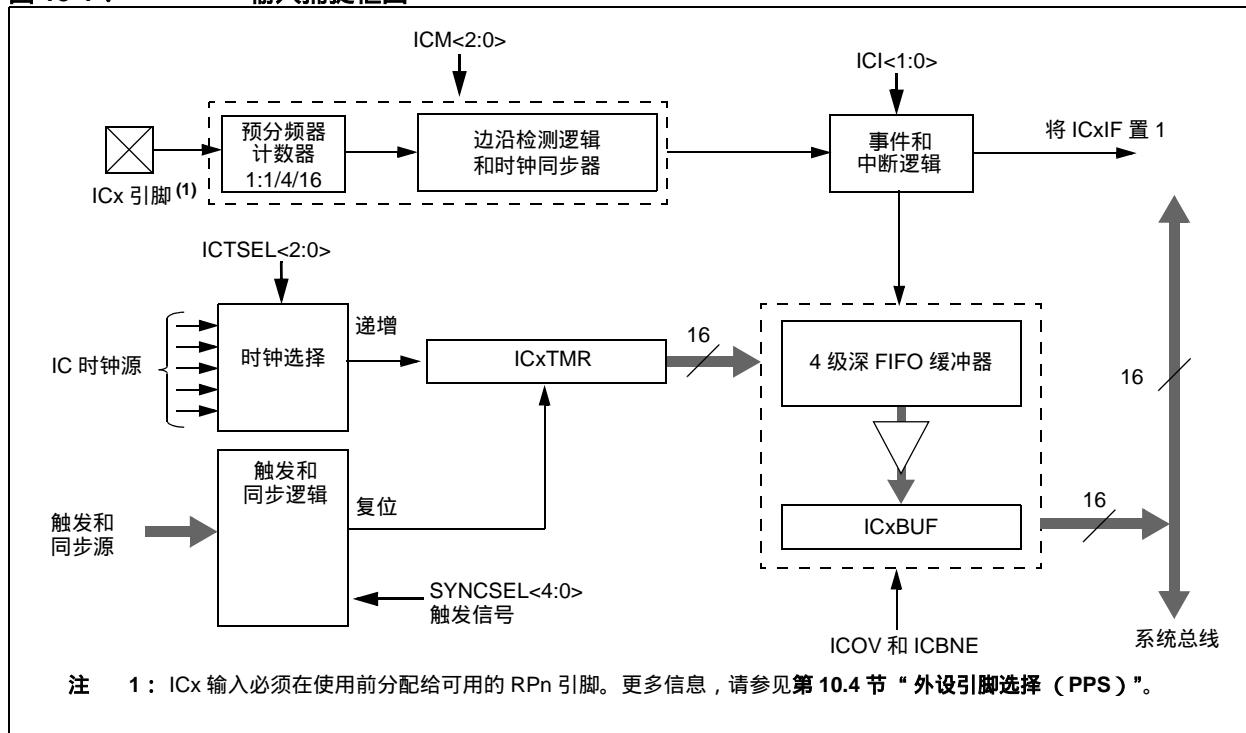
PIC24FJ64GB004系列中的器件都具有9个独立的输入捕捉模块。每个模块都为捕捉外部脉冲事件和产生中断提供了各种配置和工作选项。

输入捕捉模块的主要特性包括：

- 通过级联两个相邻的模块，可采用硬件方式配置为所有的32位工作模式
- 输入捕捉操作有同步和触发两种模式，最多具有30个用户可选的触发/同步源
- 一个4级深的FIFO缓冲器，用于捕捉并保存多个事件的定时器值
- 可配置中断产生
- 为每个模块提供了最多6个时钟源，驱动独立的内部16位计数器

此模块可通过两个寄存器进行控制：ICxCON1（寄存器13-1）和ICxCON2（寄存器13-2）。模块的一般框图如图13-1所示。

图13-1：输入捕捉框图



## 13.1 一般工作模式

### 13.1.1 同步和触发模式

默认情况下，输入捕捉模块工作在自由运行模式下。内部16位计数器ICxTMR可连续向上计数，并在每次溢出时从FFFFh折回0000h，其周期与所选的外部时钟源同步。发生捕捉事件时，将内部计数器的当前16位值写入FIFO缓冲器。

在同步模式下，一旦使能了所选的时钟源，模块就开始捕捉ICx引脚上的事件。只要所选的同步源发生事件，内部计数器就会复位。在触发模式下，模块等待来自其他内部模块的同步事件发生后，才允许内部计数器运行。

通过把SYNCSEL位设置为00000并清零ICTRIG位(ICxCON2<7>)选择标准的自由运行工作模式。当SYNCSEL位设置为除00000以外的其他值时，将选择同步和触发模式。ICTRIG位选择同步或触发模式；置1该位将选择触发工作模式。在两种模式下，SYNCSEL位都可决定同步/触发源。

当SYNCSEL位设置为00000且ICTRIG置1时，模块工作在软件触发模式下。在此情况下，可通过手动将TRIGSTAT位(ICxCON2<6>)置1来启动捕捉操作。

## 13.1.2 级联 (32 位) 模式

默认情况下，每个模块都可以使用自己的 16 位定时器独立运行。要提高分辨率，可将相邻的偶数编号的模块和奇数编号的模块配置为一个 32 位模块。（例如，模块 1 和 2 配对，模块 3 和 4 配对等）。奇数编号的模块 (ICx) 提供 32 位寄存器对中的低 16 位，而偶数编号的模块 (ICy) 提供高 16 位。ICx 寄存器的折回操作会导致其相应的 ICy 寄存器递增。

通过将两个模块的 IC32 位 (ICxCON2<8>) 置 1 使用硬件来配置级联操作。

## 13.2 捕捉操作

输入捕捉模块可配置为在 ICx 信号的上升沿或所有跳变沿处捕捉定时器值并产生中断。捕捉可配置在所有上升沿上发生，或仅在某些上升沿（例如每 4 个上升沿或每 16 个上升沿）发生。可单独配置中断以在每个事件或一组事件发生时产生中断。

将模块设置为捕捉操作：

1. 将 ICx 输入配置为可用的外设引脚选择引脚之一。
2. 如果要使用同步模式，应在继续前禁止同步源。
3. 确保 ICBNE 位 (ICxCON1<3>) 清零之前已通过读 ICxBUF 从 FIFO 中除去了所有以前的数据。
4. 设置 SYNCSEL 位 (ICxCON2<4:0>) 以得到所需的同步 / 触发源。
5. 设置 ICTSEL 位 (ICxCON1<12:10>) 以得到所需的时钟源。
6. 设置 ICI 位 (ICxCON1<6:5>) 以得到所需的中断频率。
7. 选择同步或触发工作模式：
  - a) 确认 SYNCSEL 位未设置为 00000。
  - b) 对于同步模式，清零 ITRIG 位 (ICxCON2<7>)。
  - c) 对于触发模式，将 ITRIG 位置 1 并清零 TRIGSTAT 位 (ICxCON2<6>)。
8. 设置 ICM 位 (ICxCON1<2:0>) 以得到所需的工作模式。
9. 使能所选的触发 / 同步源。

对于 32 位级联操作，设置步骤稍有不同：

1. 将两个模块的 IC32 位 (ICyCON2<8> 和 ICxCON2<8>) 置 1，首先使能偶数编号的模块。这可以确保模块一起启动。
2. 分别设置两个模块的 ICTSEL 和 SYNCSEL 位以选择相同的同步 / 触发源和时基源。首先设置偶数编号的模块，然后设置奇数编号的模块。两个模块必须使用相同的ICTSEL和SYNCSEL设置。
3. 清零偶数编号的模块的 ITRIG 位 (ICyCON2<7>)；这可以强制该模块与奇数编号的模块运行在同步模式下（不管其触发设置如何）。
4. 使用奇数编号的模块的 ICI 位 (ICxCON1<6:5>) 来得到所需的中断频率。
5. 使用奇数编号的模块的 ITRIG 位 (ICxCON2<7>) 配置触发或同步工作模式。

**注：** 对于同步工作模式，最后一个步骤是使能同步源。使能同步源之前，两个输入捕捉模块都保持复位状态。

6. 使用奇数编号的模块的 ICM 位 (ICxCON1<2:0>) 设置所需的捕捉模式。

使能了时基和触发 / 同步源之后，模块准备就绪，可以捕捉事件了。ICBNE 位 (ICxCON1<3>) 置 1 时，FIFO 中至少有一个捕捉值。从 FIFO 读取输入捕捉值，直到 ICBNE 清零。

对于 32 位操作，读取 ICxBUF 和 ICyBUF 以得到完整的 32 位定时器值 (ICxBUF 用于低 16 位，ICyBUF 用于高 16 位)。当奇数编号的模块的 ICBNE 位 (ICxCON1<3>) 置 1 时，FIFO 缓冲器中至少有一个捕捉值。继续读缓冲寄存器直到 ICBNE 清零（由硬件自动执行）。

## 寄存器 13-1 : ICxCON1 : 输入捕捉 x 控制寄存器 1

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
—	—	ICSDL	ICTSEL2	ICTSEL1	ICTSEL0	—	—
bit 15							bit 8

U-0	R/W-0	R/W-0	R-0, HCS	R-0, HCS	R/W-0	R/W-0	R/W-0
—	ICI1	ICI0	ICOV	ICBNE	ICM2 <sup>(1)</sup>	ICM1 <sup>(1)</sup>	ICM0 <sup>(1)</sup>
bit 7							bit 0

**图注 :** HCS = 可由硬件清零 / 置 1 的位

R = 可读位                            W = 可写位                            U = 未实现位 , 读为 0

-n = 上电复位时的值                1 = 置 1                            0 = 清零                            x = 未知

bit 15-14      未实现 : 读为 0

bit 13          **ICSDL** : 空闲时输入捕捉 x 模块停止控制位

1 = 输入捕捉模块将在 CPU 空闲模式下停止工作

0 = 输入捕捉模块将在 CPU 空闲模式下继续工作

bit 12-10       **ICTSEL<2:0>** : 输入捕捉定时器选择位

111 = 系统时钟 ( Fosc/2 )

110 = 保留

101 = 保留

100 = Timer1

011 = Timer5

010 = Timer4

001 = Timer2

000 = Timer3

bit 9-7          未实现 : 读为 0

bit 6-5          **ICI<1:0>** : 选择每次中断发生的捕捉次数的位

11 = 每四次捕捉事件中断一次

10 = 每三次捕捉事件中断一次

01 = 每两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 x 溢出状态标志位 ( 只读 )

1 = 已发生输入捕捉溢出

0 = 未发生输入捕捉溢出

bit 3            **ICBNE** : 输入捕捉 x 缓冲器空状态位 ( 只读 )

1 = 输入捕捉缓冲器不为空 , 至少可以读取一个以上的捕捉值

0 = 输入捕捉缓冲器为空

bit 2-0          **ICM<2:0>** : 输入捕捉模式选择位<sup>(1)</sup>

111 = 中断模式 : 只有当器件处于休眠或空闲模式时输入捕捉才可作为中断引脚 ( 只检测上升沿 , 所有其他控制位不适用 )

110 = 未使用 ( 禁止模块 )

101 = 预分频器捕捉模式 : 每 16 个上升沿捕捉一次

100 = 预分频器捕捉模式 : 每 4 个上升沿捕捉一次

011 = 简单捕捉模式 : 每个上升沿捕捉一次

010 = 简单捕捉模式 : 每个下降沿捕捉一次

001 = 边沿检测捕捉模式 : 每个沿 ( 上升和下降 ) 捕捉一次 , ICI<1:0> 位不控制此模式下中断的产生

000 = 输入捕捉模块关闭

**注 1 :** ICx 输入也必须配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “ 外设引脚选择 ( PPS ) ”。

# PIC24FJ64GB004 系列

## 寄存器 13-2 : ICxCON2 : 输入捕捉 x 控制寄存器 2

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	IC32
bit 15							bit 8

R/W-0	R/W-0, HS	U-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-1
ICTRIG	TRIGSTAT	—	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0
bit 7							bit 0

图注 : HS = 可由硬件置 1 的位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-9 未实现 : 读为 0

bit 8 IC32 : 级联两个 IC 模块使能位 (32 位操作)

1 = ICx 和 ICy 以级联的形式作为 32 位模块使用 (两个模块中的此位都必须置 1)

0 = ICx 作为 16 位模块独立工作

bit 7 ICTRIG : ICx 触发 / 同步选择位

1 = 用 SYNCSELx 位指定的源触发 ICx

0 = 使 ICx 与 SYNCSELx 位指定的源同步

bit 6 TRIGSTAT : 定时器触发状态位

1 = 定时器源被触发且正在运行 (该位可用硬件置 1, 也可用软件置 1)

0 = 定时器源未被触发, 保持清零状态

bit 5 未实现 : 读为 0

bit 4-0 SYNCSEL<4:0> : 触发 / 同步源选择位

11111 = 保留

11110 = 保留

11101 = 保留

11100 = CTMU<sup>(1)</sup>

11011 = A/D<sup>(1)</sup>

11010 = 比较器 3<sup>(1)</sup>

11001 = 比较器 2<sup>(1)</sup>

11000 = 比较器 1<sup>(1)</sup>

10111 = 输入捕捉 4

10110 = 输入捕捉 3

10101 = 输入捕捉 2

10100 = 输入捕捉 1

10011 = 保留

10010 = 保留

1000x = 保留

01111 = Timer5

01110 = Timer4

01101 = Timer3

01100 = Timer2

01011 = Timer1

01010 = 输入捕捉 5

01001 = 保留

01000 = 保留

00111 = 保留

00110 = 保留

00101 = 输出比较 5

00100 = 输出比较 4

00011 = 输出比较 3

00010 = 输出比较 2

00001 = 输出比较 1

00000 = 不与任何其他模块同步

注 1 : 仅将这些输入用作触发源, 不能用作同步源。

## 14.0 带专用定时器的输出比较

**注：** 本数据手册总结了该组PIC24F器件具有的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F系列参考手册》中的第35章“带专用定时器的输出比较”(DS39723A\_CN)。

PIC24FJ64GB004系列中的器件都具有9个独立的输出比较模块。每个模块都为发生器件内部事件时产生连续脉冲信号提供了各种配置和工作选项，还可产生脉宽调制(Pulse-Width Modulated, PWM)波形以驱动功率应用。

输出比较模块的主要特性包括：

- 通过级联两个相邻的模块，可采用硬件方式配置为所有的32位工作模式
- 输出比较操作有同步和触发两种模式，最多具有30个用户可选的触发/同步源
- 两个独立的周期寄存器（一个主寄存器OCxR和一个辅助寄存器OCxRS），可更加灵活地产生不同宽度的脉冲
- 可配置为在发生输出事件时产生单脉冲信号或连续脉冲信号，或产生连续的PWM波形
- 为每个模块提供了最多6个时钟源，驱动独立的内部16位计数器

### 14.1 一般工作模式

#### 14.1.1 同步和触发模式

默认情况下，输出比较模块工作在自由运行模式下。内部16位计数器OCxTMR可连续向上计数，并在每次溢出时从FFFFh折回0000h，其周期与所选的外部时钟源同步。当内部计数器与其中一个周期寄存器发生匹配时，产生一个比较或PWM事件。

在同步模式下，一旦使能了所选的时钟源，模块就开始执行比较或PWM操作。只要所选的同步源发生事件，模块的内部计数器就会复位。在触发模式下，模块等待来自其他内部模块的同步事件发生后，才允许计数器运行。

默认情况下或SYNCSEL位(OCxCON2<4:0>)设置为00000时，选择自由运行模式。当SYNCSEL位设置为00000之外的其他值时，将选择同步或触发模式。OCTRIG位(OCxCON2<7>)用于选择同步或触发模式；置1该位将选择触发工作模式。在两种模式下，SYNCSEL位都可决定同步/触发源。

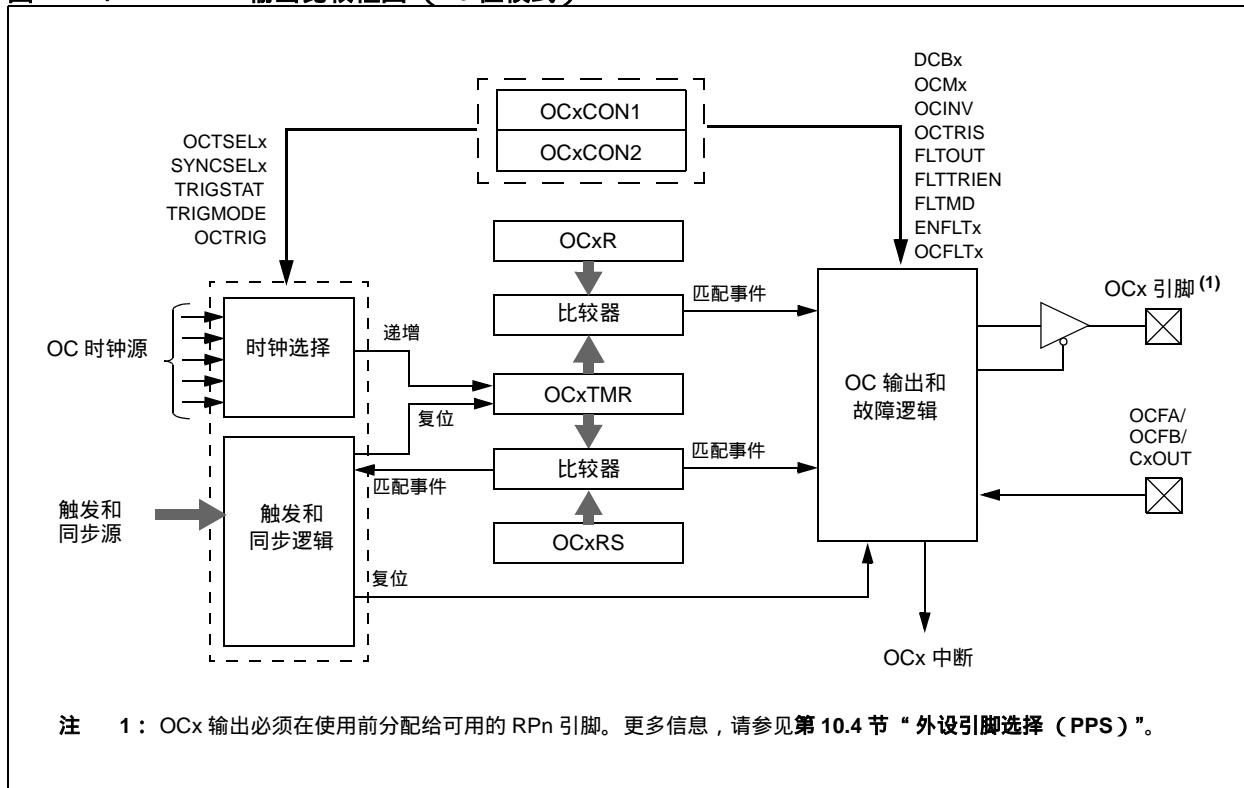
#### 14.1.2 级联(32位)模式

默认情况下，每个模块都可以使用自己的16位定时器和占空比寄存器独立运行。要提高分辨率，可将相邻的偶数编号的模块和奇数编号的模块配置为一个32位模块。(例如，模块1和2配对，模块3和4配对等)。奇数编号的模块(OCx)提供32位寄存器对中的低16位，而偶数编号的模块(OCy)提供高16位。OCx寄存器的折回操作会导致其相应的OCy寄存器递增。

通过将两个模块的OC32位(OCxCON2<8>)置1使用硬件来配置级联操作。

# PIC24FJ64GB004 系列

图 14-1：输出比较框图（16 位模式）



## 14.2 比较操作

在比较模式下（图 14-1），输出比较模块可配置为产生单脉冲信号或产生连续脉冲信号；它还可在发生每个定时器事件时重复翻转输出引脚。

将模块设置为比较操作：

1. 将 OC<sub>x</sub> 输出配置为可用的外设引脚选择引脚之一。
2. 计算 OC<sub>xR</sub> 和（对于双比较模式）OC<sub>xRS</sub> 占空比寄存器所需的值：
  - a) 确定指令时钟周期。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。
  - b) 计算从定时器起始值（0000h）到输出脉冲的上升沿所需的时间。
  - c) 根据所需的脉冲宽度和到脉冲上升沿的时间计算出现脉冲下降沿的时间。
3. 将上升沿的值写入 OC<sub>xR</sub>，下降沿的值写入 OC<sub>xRS</sub>。
4. 将定时器周期寄存器 PRy 的值设置为大于或等于 OC<sub>xRS</sub> 中的值。
5. 对于触发工作模式，将 OCTRIG 置 1 以使能触发模式。置 1 或清零 TRIGMODE 可配置触发工作模式，置 1 或清零 TRIGSTAT 可选择硬件触发或软件触发。对于同步模式，清零 OCTRIG。
6. 设置 SYNCSEL<4:0> 位可配置触发或同步源。如需定时器工作在自由运行模式下，请把 SYNCSEL 位设置为 00000（无同步/触发源）。
7. 使用 OCTSEL<2:0> 位选择时基源。如有必要，将使能比较时基进行计数的选定定时器的 TON 位置 1。使能了时基后，启动同步工作模式；发生触发源事件后，启动触发工作模式。
8. 设置 OCM<2:0> 位（= 0xx）以选择合适的比较操作。

对于 32 位级联操作，还包含以下必需步骤：

1. 将两个寄存器的 OC32 位（即 OC<sub>yCON2<8></sub> 和 OC<sub>xCON2<8></sub>）置 1。首先使能偶数编号的模块，以确保模块一起启动。
2. 清零偶数编号的模块的 OCTRIG 位（OC<sub>yCON2</sub>），使其在同步模式下运行。
3. 为 OC<sub>y</sub> 配置所需的输出和故障设置。
4. 通过清零 OCTRIS 位强制 OC<sub>x</sub> 的输出引脚为输出状态。
5. 如果需要触发工作模式，请使用 OCTRIG（OC<sub>xCON2<7></sub>）、TRIGSTAT（OC<sub>xCON2<6></sub>）和 SYNCSEL（OC<sub>xCON2<4:0></sub>）位配置 OC<sub>x</sub> 中的触发选项。
6. 首先为 OC<sub>y</sub> 配置所需的比较或 PWM 工作模式（通过设置 OCM<2:0>），然后再对 OC<sub>x</sub> 进行配置。

根据选择的输出模式，模块保持 OC<sub>x</sub> 引脚处于默认状态，并在 OC<sub>xR</sub> 与定时器匹配时强制翻转该引脚的状态。在双比较模式下，当定时器与 OC<sub>xRS</sub> 匹配时，OC<sub>x</sub> 被强制返回其默认状态。在单比较模式下定时器与 OC<sub>xR</sub> 匹配，以及在双比较模式下每次定时器与 OC<sub>xRS</sub> 匹配后，OC<sub>xIF</sub> 中断标志都会置 1。

单脉冲事件仅发生一次，但是可通过简单地重写 OC<sub>yCON1</sub> 寄存器的值使其重复发生。连续脉冲事件在终止之前可无限次地发生。

## 14.3 脉宽调制 (PWM) 模式

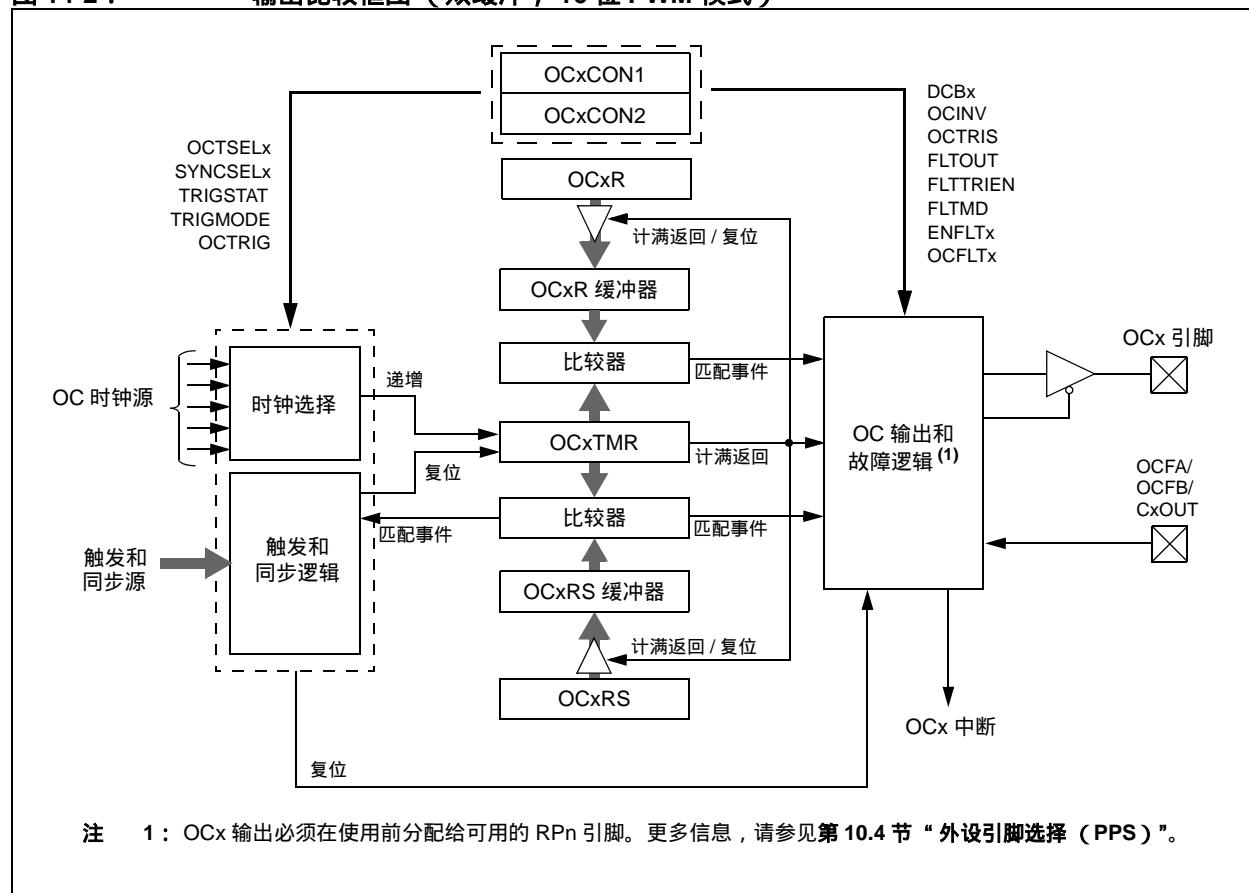
在 PWM 模式下，输出比较模块可配置为产生边沿对齐或中心对齐的脉冲波形。所有 PWM 操作都是双缓冲的（缓冲寄存器在模块内部，没有将其映射到 SFR 空间）。将输出比较模块配置为 PWM 模式，需要以下步骤：

1. 将 OCx 输出配置为可用的外设引脚选择引脚之一。
2. 计算所需的占空比并将计算结果装入 OCxR 寄存器。
3. 计算所需的周期并将计算结果装入 OCxRS 寄存器。
4. 选择当前的 OCx 作为同步源，方法是把 0x1F 写入 SYNCSEL<4:0> (OCxCON2<4:0>)，把 0 写入 OCTRIG (OCxCON2<7>)。

5. 通过写 OCTSEL2<2:0> 位 (OCxCON<12:10>) 选择时钟源。
6. 如果需要的话，允许定时器和输出比较模块中断。使用 PWM 故障引脚时需要输出比较中断。
7. 通过 OCM<2:0> (OCxCON1<2:0>) 位选择所需的 PWM 模式。
8. 如果定时器被选为时钟源，请设置 TMRy 预分频值，并通过将 TON (TxCON<15>) 位置 1 来使能时基。

**注：** 本外设包含的输入和输出功能可能需要通过外设引脚选择功能进行配置。更多信息，请参见第 10.4 节“外设引脚选择 (PPS)”。

图 14-2：输出比较框图（双缓冲，16 位 PWM 模式）



### 14.3.1 PWM 周期

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

#### 公式 14-1：计算 PWM 周期<sup>(1)</sup>

$$\text{PWM 周期} = [(PRy) + 1] \cdot TCY \cdot (\text{定时器预分频值})$$

$$\text{其中: } \text{PWM 频率} = 1/\text{[PWM 周期]}$$

**注 1：**基于  $TCY = Tosc * 2$ ；禁止打盹模式和 PLL。

**注：**若 PRy 的值为 N，则会使 PWM 周期成为  $N + 1$  个时基计数周期。例如，写入 PRy 寄存器的值为 7 将产生由 8 个时基周期组成的 PWM 周期。

### 公式 14-2：计算最大 PWM 分辨率<sup>(1)</sup>

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

**注 1：**基于  $FCY = Fosc/2$ ；禁止打盹模式和 PLL。

### 例 14-1：计算 PWM 周期和占空比<sup>(1)</sup>

- 在所需的 PWM 频率为 52.08 kHz、FOSC = 8 MHz 并带 PLL (32 MHz 器件时钟速率) 且 Timer2 预分频比为 1:1 的条件下计算定时器周期寄存器的值。

$$TCY = 2 * Tosc = 62.5 \text{ ns}$$

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

$$\text{PWM 周期} = (PR2 + 1) \cdot TCY \cdot (\text{Timer2 预分频值})$$

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

$$PR2 = 306$$

- 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：

$$\text{PWM 分辨率} = \log_{10}(FCY/FPWM)/\log_{10}2 \text{ 位}$$

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

$$= 8.3 \text{ 位}$$

**注 1：**基于  $TCY = 2 * Tosc$ ，禁止打盹模式和 PLL。

### 14.3.2 PWM 占空比

通过写 OCxRS 和 OCxR 寄存器指定 PWM 占空比。在任何时间都可以写入 OCxRS 和 OCxR 寄存器，但是在 PRy 和 TMry 发生匹配（即周期完成）前占空比值不会被锁存。这可以为 PWM 占空比提供双缓冲，对于 PWM 的无毛刺操作极其重要。

以下是 PWM 占空比的部分重要边界参数：

- 如果 OCxR、OCxRS 和 PRy 均装入 0000h，那么 OCx 引脚将保持低电平（占空比为 0%）。
- 如果 OCxRS 大于 PRy，则引脚将保持高电平（占空比为 100%）。

请参见例 14-1 了解 PWM 模式时序的详细信息。表 14-1 和表 14-2 给出了当器件分别工作在 4 MIPS 和 10 MIPS 时，所对应的 PWM 频率和分辨率的示例。

## 14.4 小于单指令周期的分辨率

DCB 位 ( $OCxCON2<10:9>$ ) 提供了比一个指令周期更好的分辨率。使用这些分辨率时，会使由匹配事件产生的下降沿延迟出现小于一个指令周期的时间。

例如，设置  $DCB<1:0> = 10$  时，将导致下降沿在发生匹配事件的指令周期的中间出现，而不是在开头出现。当  $OCM<2:0> = 001$  时，无法使用这些位。模块工作在 PWM 模式 ( $OCM<2:0> = 110$  或  $111$ ) 下时，DCB 位将是双缓冲的。

DCB 位用于与提供系统时钟的时钟源一起使用。使用已使能预分频器的定时器时，由 DCB 位引起的下降沿延迟将以系统时钟周期（而不是定时器的周期）作为参考。

表 14-1：器件工作在 4 MIPS ( $F_{CY} = 4 \text{ MHz}$ ) 时的 PWM 频率和分辨率示例<sup>(1)</sup>

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

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

表 14-2：器件工作在 16 MIPS ( $F_{CY} = 16 \text{ MHz}$ ) 时的 PWM 频率和分辨率示例<sup>(1)</sup>

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

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

## 寄存器 14-1 : OCxCON1 : 输出比较 x 控制寄存器 1

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	OCSIDL	OCTSEL2	OCTSEL1	OCTSEL0	ENFLT2 <sup>(2)</sup>	ENFLT1
bit 15	bit 8						

R/W-0	R/W-0, HCS	R/W-0, HCS	R/W-0, HCS	R/W-0	R/W-0	R/W-0	R/W-0
ENFLT0	OCFLT2	OCFLT1	OCFLT0	TRIGMODE	OCM2 <sup>(1)</sup>	OCM1 <sup>(1)</sup>	OCM0 <sup>(1)</sup>
bit 7	bit 0						

### 图注 :

HCS = 可由硬件清零 / 置 1 的位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-14 未实现 : 读为 0
- bit 13 **OCSIDL** : 在空闲模式下停止输出比较 x 控制位  
 1 = 输出比较 x 在 CPU 空闲模式下停止  
 0 = 输出比较 x 在 CPU 空闲模式下继续工作
- bit 12-10 **OCTSEL<2:0>** : 输出比较 x 定时器选择位  
 111 = 系统时钟  
 110 = 保留  
 101 = 保留  
 100 = Timer1  
 011 = Timer5  
 010 = Timer4  
 001 = Timer3  
 000 = Timer2
- bit 9 **ENFLT2** : 比较器故障输入使能位<sup>(2)</sup>  
 1 = 使能比较器故障输入  
 0 = 禁止比较器故障输入
- bit 8 **ENFLT1** : OCFB 故障输入使能位  
 1 = 使能 OCFB 故障输入  
 0 = 禁止 OCFB 故障输入
- bit 7 **ENFLT0** : OCFA 故障输入使能位  
 1 = 使能 OCFA 故障输入  
 0 = 禁止 OCFA 故障输入
- bit 6 **OCFLT2** : PWM 比较器故障条件状态位<sup>(2)</sup>  
 1 = 已经产生 PWM 比较器故障条件 (仅由硬件清零)  
 0 = 没有产生 PWM 比较器故障条件 (只有在 OCM<2:0> = 111 时才使用该位)
- bit 5 **OCFLT1** : PWM OCFB 故障输入使能位  
 1 = 已经产生 PWM OCFB 故障条件 (仅由硬件清零)  
 0 = 没有产生 PWM OCFB 故障条件 (只有在 OCM<2:0> = 111 时才使用该位)
- bit 4 **OCFLT0** : PWM OCFA 故障条件状态位  
 1 = 已经产生 PWM OCFA 故障条件 (仅由硬件清零)  
 0 = 没有产生 PWM OCFA 故障条件 (只有在 OCM<2:0> = 111 时才使用该位)
- bit 3 **TRIGMODE** : 触发状态模式选择位  
 1 = OCxRS = OCxTMR 时清零 TRIGSTAT (OCxCON2<6>) , 或用软件清零  
 0 = TRIGSTAT 只能由软件清零

注 1 : OCx 输出也必须配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

2 : 用于故障输入的比较器模块因 OCx 模块而异。OC1 和 OC2 使用比较器 1。OC3 和 OC4 使用比较器 2。OC5 使用比较器 3。

# PIC24FJ64GB004 系列

---

## 寄存器 14-1 : OCxCON1 : 输出比较 x 控制寄存器 1 (续)

bit 2-0      **OCM<2:0>** : 输出比较 x 模式选择位<sup>(1)</sup>

111 = OCx 处于中心对齐 PWM 模式

110 = OCx 处于边沿对齐 PWM 模式

101 = 双比较连续脉冲模式: 初始化OCx引脚为低电平, 当定时器值与OCxR和OCxRS交替匹配时连续翻转 OCx 的状态

100 = 双比较单脉冲模式: 初始化OCx引脚为低电平, 在一个周期内与OCxR和OCxRS匹配时交替翻转 OCx 的状态

011 = 单比较连续脉冲模式: 比较事件连续翻转 OCx 引脚

010 = 单比较单脉冲模式: 初始化 OCx 引脚为高电平, 比较事件强制 OCx 引脚为低电平

001 = 单比较单脉冲模式: 初始化 OCx 引脚为低电平, 比较事件强制 OCx 引脚为高电平

000 = 禁止输出比较通道

**注 1 :** OCx 输出也必须配置给可用的 RPn 引脚。更多信息, 请参见第 10.4 节“外设引脚选择 (PPS)”。

**2 :** 用于故障输入的比较器模块因 OCx 模块而异。OC1 和 OC2 使用比较器 1。OC3 和 OC4 使用比较器 2。OC5 使用比较器 3。

## 寄存器 14-2 : OCxCON2 : 输出比较 x 控制寄存器 2

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
FLTMD	FLTOUT	FLTTRIEN	OCINV	—	DCB1 <sup>(3)</sup>	DCB0 <sup>(3)</sup>	OC32
bit 15	bit 8						

R/W-0	R/W-0, HS	R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0
OCTRIG	TRIGSTAT	OCTRIS	SYNCSEL4	SYNCSEL3	SYNCSEL2	SYNCSEL1	SYNCSEL0
bit 7	bit 0						

图注 : HS = 可由硬件置 1 的位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 **FLTMD** : 故障模式选择位

1 = 故障模式保持 , 直到消除故障源并且相应的 OCFLT0 位由软件清零  
0 = 故障模式保持 , 直到消除故障源并且启动了新的 PWM 周期

bit 14 **FLTOUT** : 故障输出位

1 = 发生故障时 PWM 输出被驱动为高电平  
0 = 发生故障时 PWM 输出被驱动为低电平

bit 13 **FLTTRIEN** : 故障输出状态选择位

1 = 发生故障条件时引脚被强制为输出状态  
0 = 引脚 I/O 状态不受故障影响

bit 12 **OCINV** : OCMP 反相位

1 = OCx 输出反相  
0 = OCx 输出不反相

bit 11 未实现 : 读为 0

bit 10-9 **DCB<1:0>** : OC 脉宽最低有效位<sup>(3)</sup>

11 = OCx 下降沿延迟 3/4 指令周期  
10 = OCx 下降沿延迟 1/2 指令周期  
01 = OCx 下降沿延迟 1/4 指令周期  
00 = OCx 下降沿在指令周期一开始就出现

bit 8 **OC32** : 级联两个 OC 模块使能位 (32 位操作)

1 = 使能级联模块操作  
0 = 禁止级联模块操作

bit 7 **OCTRIG** : OCx 触发 / 同步选择位

1 = 用 SYNCSELx 位指定的源触发 OCx  
0 = 用 SYNCSELx 位指定的源同步 OCx

bit 6 **TRIGSTAT** : 定时器触发状态位

1 = 定时器源被触发且正在运行  
0 = 定时器源未被触发 , 保持清零状态

bit 5 **OCTRIS** : OCx 输出引脚方向选择位

1 = OCx 引脚是三态引脚  
0 = OCx 引脚上连接了输出比较外设 x

注 1 : 不要通过以下方式将 OC 模块用作自己的触发源 : 选择这种模式或其他同等 SYNCSEL 设置。

2 : 仅将这些输入用作触发源 , 不能用作同步源。

3 : OCINV = 1 时 , 这些位会影响上升沿。OCM ( OCxCON1<1:0>) = 001 时 , 这些位不起作用。

# PIC24FJ64GB004 系列

---

## 寄存器 14-2 : OCxCON2 : 输出比较 x 控制寄存器 2 (续)

bit 4-0      **SYNCSEL<4:0>** : 触发 / 同步源选择位

11111 = 此 OC 模块<sup>(1)</sup>

11110 = 保留

11101 = 保留

11100 = CTMU<sup>(2)</sup>

11011 = A/D<sup>(2)</sup>

11010 = 比较器 3<sup>(2)</sup>

11001 = 比较器 2<sup>(2)</sup>

11000 = 比较器 1<sup>(2)</sup>

10111 = 输入捕捉 4<sup>(2)</sup>

10110 = 输入捕捉 3<sup>(2)</sup>

10101 = 输入捕捉 2<sup>(2)</sup>

10100 = 输入捕捉 1<sup>(2)</sup>

100xx = 保留

01111 = Timer5

01110 = Timer4

01101 = Timer3

01100 = Timer2

01011 = Timer1

01010 = 输入捕捉 5<sup>(2)</sup>

01001 = 保留

01000 = 保留

00111 = 保留

00110 = 保留

00101 = 输出比较 5<sup>(1)</sup>

00100 = 输出比较 4<sup>(1)</sup>

00011 = 输出比较 3<sup>(1)</sup>

00010 = 输出比较 2<sup>(1)</sup>

00001 = 输出比较 1<sup>(1)</sup>

00000 = 不与任何其他模块同步

**注** 1 : 不要通过以下方式将 OC 模块用作自己的触发源 : 选择这种模式或其他同等 SYNCSEL 设置。

2 : 仅将这些输入用作触发源 , 不能用作同步源。

3 : OCINV = 1 时 , 这些位会影响上升沿。OCM ( OCxCON1<1:0> ) = 001 时 , 这些位不起作用。

## 15.0 串行外设接口（SPI）

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 23 章“串行外设接口（SPI）”(DS39699A\_CN)。

串行外设接口（Serial Peripheral Interface，SPI）模块是一个同步串行接口，可用于与其他外设或者单片机进行通信。这些外设器件可以是串行 EEPROM、移位寄存器、显示驱动器和 A/D 转换器等。SPI 模块与 Motorola® 的 SPI 和 SIOP 接口兼容。PIC24FJ64GB004 系列的所有器件都包含三个 SPI 模块。

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

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

工作在主模式或从模式下时，该模块还支持基本的帧 SPI 协议。共支持四种帧 SPI 配置。

SPI 串行接口由以下四个引脚组成：

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

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

图 15-1 和图 15-2 所示为标准模式和增强型模式下该模块的框图。

**注：**在本章中，SPI 模块统称为 SPIx，或分别称为 SPI1、SPI2 和 SPI3。特殊功能寄存器也使用类似的符号表示。例如，SPIxCON1 和 SPIxCON2 表示这 3 个 SPI 模块中任何一个的控制寄存器。

# PIC24FJ64GB004 系列

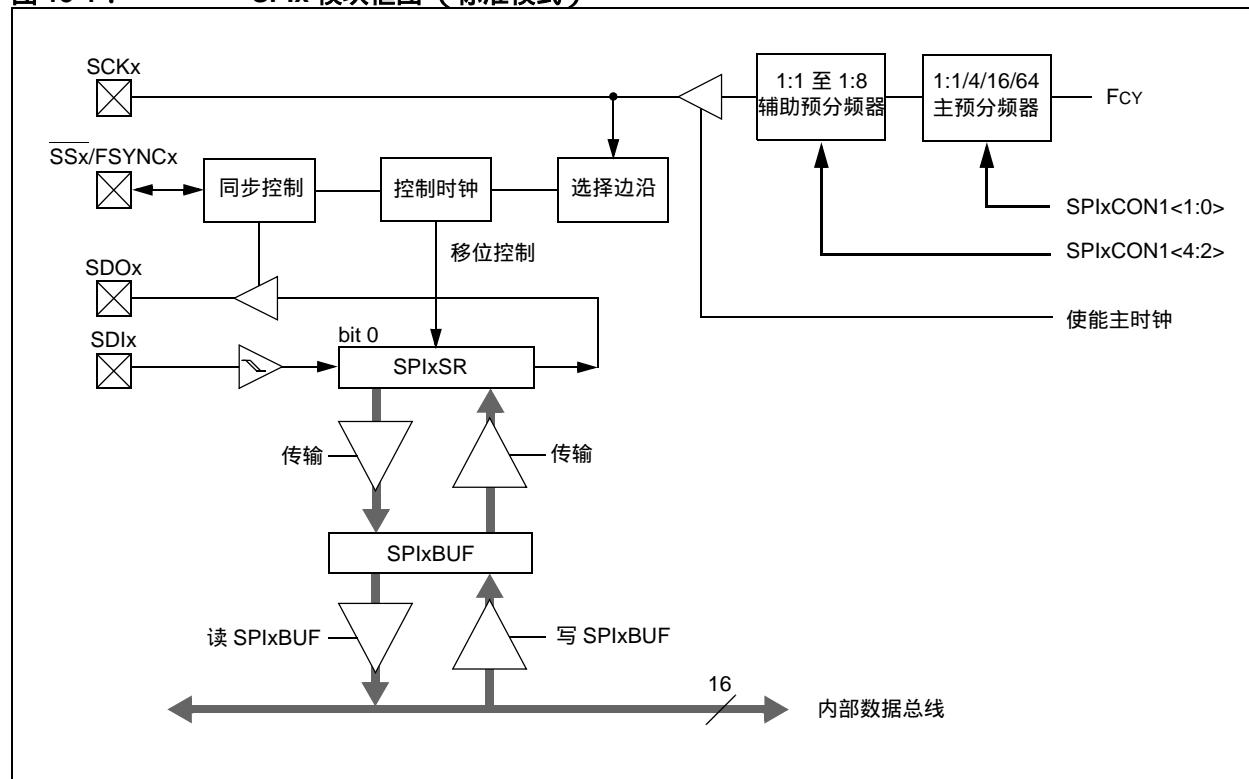
要将 SPI 模块设置为工作在标准主模式下，请遵循以下步骤：

1. 如果使用中断：
  - a) 将相应 IFS 寄存器中的 SPIxIF 位清零。
  - b) 将相应 IEC 寄存器中的 SPIxIE 位置 1。
  - c) 通过写相应 IPC 寄存器中的 SPIxIP 位来设置中断的优先级。
2. 将所需设置写入 SPIxCON1 和 SPIxCON2 寄存器，且 MSTEN (SPIxCON1<5>) = 1。
3. 将 SPIROV 位 (SPIxSTAT<6>) 清零。
4. 通过将 SPIEN 位 (SPIxSTAT<15>) 置 1 使能 SPI 工作。
5. 将待发送数据写入 SPIxBUF 寄存器。数据一写入 SPIxBUF 寄存器发送 (和接收) 就会立即开始。

要将 SPI 模块设置为工作在标准从模式下，请遵循以下步骤：

1. 将 SPIxBUF 寄存器清零。
2. 如果使用中断：
  - a) 将相应 IFS 寄存器中的 SPIxIF 位清零。
  - b) 将相应 IEC 寄存器中的 SPIxIE 位置 1。
  - c) 通过写相应 IPC 寄存器中的 SPIxIP 位来设置中断的优先级。
3. 将所需设置写入 SPIxCON1 和 SPIxCON2 寄存器，且 MSTEN (SPIxCON1<5>) = 0。
4. 将 SMP 位清零。
5. 如果将 CKE 位 (SPIxCON1<8>) 置 1，然后必须将 SSEN 位 (SPIxCON1<7>) 置 1 来使能 SS<sub>x</sub> 引脚。
6. 将 SPIROV 位 (SPIxSTAT<6>) 清零。
7. 通过将 SPIEN 位 (SPIxSTAT<15>) 置 1 使能 SPI 工作。

图 15-1： SPIx 模块框图（标准模式）



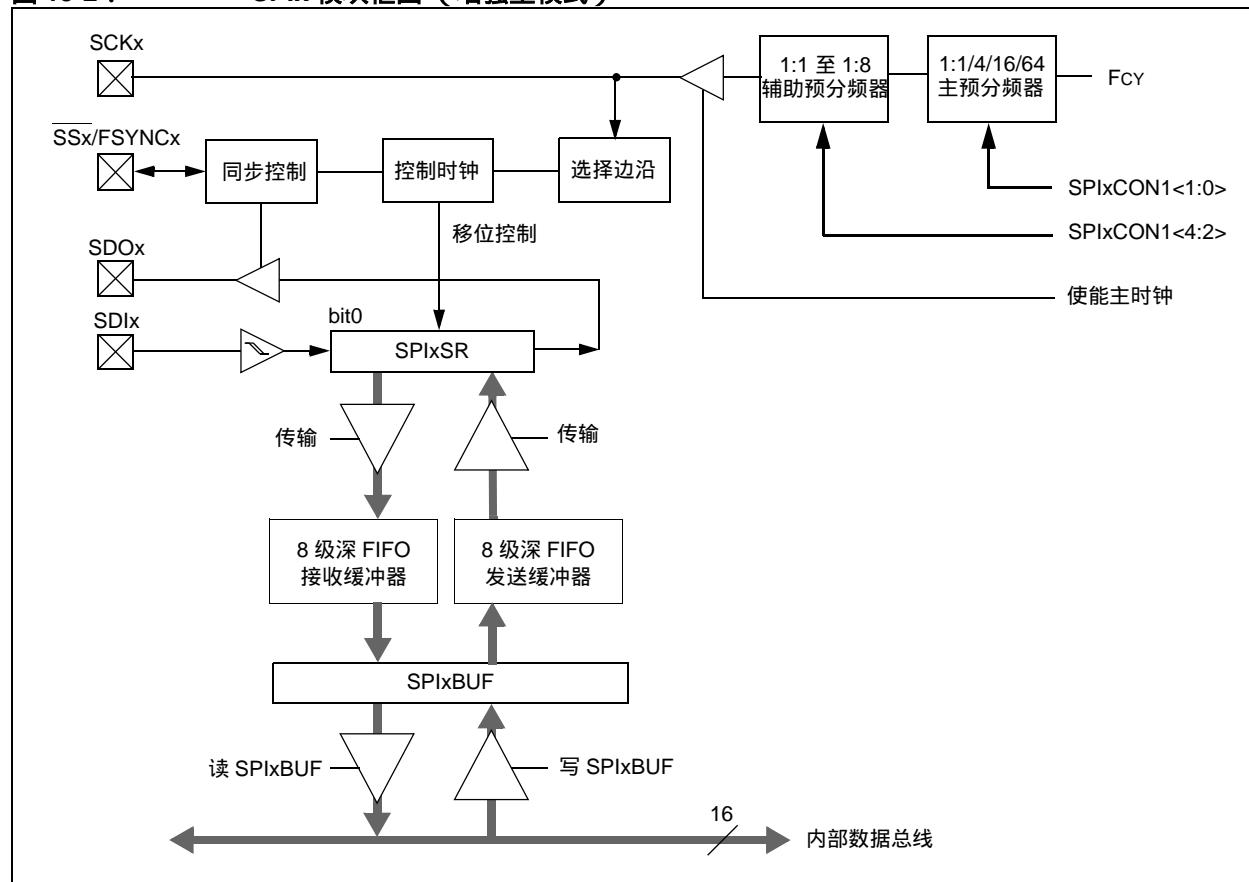
要将 SPI 模块设置为工作在增强型缓冲主模式下，请遵循以下步骤：

1. 如果使用中断：
  - a) 将相应 IFS 寄存器中的 SPIxIF 位清零。
  - b) 将相应 IEC 寄存器中的 SPIxIE 位置 1。
  - c) 写相应 IPC 寄存器中的 SPIxIP 位。
2. 将所需设置写入 SPIxCON1 和 SPIxCON2 寄存器，且 MSTEN ( SPIxCON1<5>) = 1。
3. 将 SPIROV 位 ( SPIxSTAT<6>) 清零。
4. 通过将 SPIBEN 位 ( SPIxCON2<0>) 置 1 选择增强型缓冲模式。
5. 通过将 SPIEN 位 ( SPIxSTAT<15>) 置 1 使能 SPI 工作。
6. 将待发送数据写入 SPIxBUF 寄存器。数据一写入 SPIxBUF 寄存器发送 (和接收) 就会立即开始。

要将 SPI 模块设置为工作在增强型缓冲从模式下，请遵循以下步骤：

1. 将 SPIxBUF 寄存器清零。
2. 如果使用中断：
  - a) 将相应 IFS 寄存器中的 SPIxIF 位清零。
  - b) 将相应 IEC 寄存器中的 SPIxIE 位置 1。
  - c) 通过写相应 IPC 寄存器中的 SPIxIP 位来设置中断的优先级。
3. 将所需设置写入 SPIxCON1 和 SPIxCON2 寄存器，且 MSTEN ( SPIxCON1<5>) = 0。
4. 将 SMP 位清零。
5. 如果 CKE 位置 1，则 SSEN 位也必须置 1 以使能 SS<sub>x</sub> 引脚。
6. 将 SPIROV 位 ( SPIxSTAT<6>) 清零。
7. 通过将 SPIBEN 位 ( SPIxCON2<0>) 置 1 选择增强型缓冲模式。
8. 通过将 SPIEN 位 ( SPIxSTAT<15>) 置 1 使能 SPI 工作。

图 15-2： SPIx 模块框图（增强型模式）



# PIC24FJ64GB004 系列

寄存器 15-1 : SPIxSTAT : SPIx 状态和控制寄存器

R/W-0	U-0	R/W-0	U-0	U-0	R-0	R-0	R-0
SPIEN <sup>(1)</sup>	—	SPISIDL	—	—	SPIBEC2	SPIBEC1	SPIBEC0
bit 15	bit 8						

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

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

bit 15	<b>SPIEN</b> : SPIx 使能位 <sup>(1)</sup>
	1 = 使能模块并将 SCKx、SDOx、SDIx 和 SSx 配置为串行端口引脚
	0 = 禁止模块
bit 14	<b>未实现</b> : 读为 0
bit 13	<b>SPISIDL</b> : 空闲模式停止位
	1 = 当器件进入空闲模式时 , 模块停止工作
	0 = 模块在空闲模式下继续工作
bit 12-11	<b>未实现</b> : 读为 0
bit 10-8	<b>SPIBEC&lt;2:0&gt;</b> : SPIx 缓冲器元素计数位 ( 在增强型缓冲模式下有效 )
	<u>主模式</u> :
	等待中的 SPI 传输数。
	<u>从模式</u> :
	未读取的 SPI 传输数。
bit 7	<b>SRMPT</b> : 移位寄存器 ( SPIxSR ) 空位 ( 在增强型缓冲模式下有效 )
	1 = SPIx 移位寄存器为空并已做好发送或接收准备
	0 = SPIx 移位寄存器不为空
bit 6	<b>SPIROV</b> : 接收溢出标志位
	1 = 一个新字节 / 字已被完全接收并丢弃。用户软件尚未读取 SPIxBUF 寄存器中先前接收到的数据。
	0 = 未发生溢出
bit 5	<b>SRXMPT</b> : 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 不为空
bit 4-2	<b>SISEL&lt;2:0&gt;</b> : SPIx 缓冲器中断模式位 ( 在增强型缓冲模式下有效 )
	111 = 当 SPIx 发送缓冲器满时产生中断 ( SPITBF 位置 1 )
	110 = 最后一位移入 SPIxSR 导致发送 FIFO 为空时产生中断
	101 = 最后一位被移出 SPIxSR 时产生中断 , 此时发送完成
	100 = 有数据移入 SPIxSR 导致发送 FIFO 有待发送数据时产生中断
	011 = 当 SPIx 接收缓冲器满时产生中断 ( SPIRBF 位置 1 )
	010 = 当 SPIx 接收缓冲器被填满 3/4 或更多时产生中断
	001 = 接收缓冲器中有数据时产生中断 ( SRMPT 位置 1 )
	000 = 读取接收缓冲器中的最后一个数据 , 导致缓冲器为空时产生中断 ( SRXMPT 位置 1 )

注 1 : 如果 SPIEN = 1 , 这些功能在使用之前必须分配给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “ 外设引脚选择 ( PPS ) ”。

## 寄存器 15-1 : SPIxSTAT : SPIx 状态和控制寄存器 (续)

bit 1      **SPITBF** : SPIx 发送缓冲器满状态位

1 = 未开始发送 , SPIxTXB 已满  
0 = 已开始发送 , SPIxTXB 为空

在标准缓冲模式下 :

当 CPU 通过写 SPIxBUF 地址单元装载 SPIxTXB 时 , 该位由硬件自动置 1。  
当 SPIx 模块将数据从 SPIxTXB 传输到 SPIxSR 时 , 该位由硬件自动清零。

在增强型缓冲模式下 :

当 CPU 通过写 SPIxBUF 地址单元装载最后一个可用缓冲单元时 , 该位由硬件自动置 1。  
当有空的缓冲器单元可由 CPU 写入时 , 该位由硬件自动清零。

bit 0      **SPIRBF** : SPIx 接收缓冲器满状态位

1 = 接收完成 , SPIxRXB 已满  
0 = 接收未完成 , SPIxRXB 为空

在标准缓冲模式下 :

当 SPIx 将数据从 SPIxSR 传输到 SPIxRXB 时 , 该位由硬件自动置 1。  
当内核通过读 SPIxBUF 地址单元读 SPIxRXB 时 , 该位由硬件自动清零。

在增强型缓冲模式下 :

当 SPIx 将数据从 SPIxSR 传输到缓冲器填充了最后一个未读的缓冲单元时 , 该位由硬件自动置 1。  
当有空的缓冲器单元可接收来自 SPIxSR 的传输数据时 , 该位由硬件自动清零。

**注 1:** 如果 SPIEN = 1 , 这些功能在使用之前必须分配给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

# PIC24FJ64GB004 系列

## 寄存器 15-2 : SPIxCON1 : SPIx 控制寄存器 1

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	DISSCK <sup>(1)</sup>	DISSDO <sup>(2)</sup>	MODE16	SMP	CKE <sup>(3)</sup>
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 <sup>(4)</sup>	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13	未实现 : 读为 0
bit 12	<b>DISSCK</b> : 禁止 SCKx 引脚位 (仅限 SPI 主模式) <sup>(1)</sup> 1 = 禁止内部 SPI 时钟 ; 引脚用作 I/O 0 = 使能内部 SPI 时钟
bit 11	<b>DISSDO</b> : 禁止 SDOx 引脚位 <sup>(2)</sup> 1 = 模块不使用 SDOx 引脚 , 引脚用作 I/O 0 = SDOx 引脚由模块控制
bit 10	<b>MODE16</b> : 字 / 字节通信选择位 1 = 通信为字宽 (16 位) 0 = 通信为字节宽 (8 位)
bit 9	<b>SMP</b> : SPIx 数据输入采样阶段位 <u>主模式</u> : 1 = 在数据输出时间的结尾采样输入数据 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 SPIx 时 , 必须将 SMP 清零。
bit 8	<b>CKE</b> : SPIx 时钟边沿选择位 <sup>(3)</sup> 1 = 串行输出数据在时钟由有效状态变为空闲状态时改变 (见 bit 6) 0 = 串行输出数据在时钟由空闲状态变为有效状态时改变 (见 bit 6)
bit 7	<b>SSEN</b> : 从选择使能 (从模式) 位 <sup>(4)</sup> 1 = SSx 引脚用于从模式 0 = 模块不使用 SSx 引脚 , 引脚由端口功能控制
bit 6	<b>CKP</b> : 时钟极性选择位 1 = 时钟信号空闲状态为高电平 ; 有效状态为低电平 0 = 时钟信号空闲状态为低电平 ; 有效状态为高电平
bit 5	<b>MSTEN</b> : 主模式使能位 1 = 主模式 0 = 从模式

- 注**
- 1 : 如果 DISSCK = 0 , 必须将 SCKx 配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。
  - 2 : 如果 DISSDO = 0 , 必须将 SDOx 配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。
  - 3 : 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。
  - 4 : 如果 SSEN = 1 , 必须将 SSx 配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

## 寄存器 15-2 : SPIxCON1 : SPIx 控制寄存器 1 (续)

bit 4-2      **SPRE<2:0>** : 辅助预分频比位 (主模式)

111 = 辅助预分频比为 1:1

110 = 辅助预分频比为 2:1

...

000 = 辅助预分频比为 8:1

bit 1-0      **PPRE<1:0>** : 主预分频比位 (主模式)

11 = 主预分频比为 1:1

10 = 主预分频比为 4:1

01 = 主预分频比为 16:1

00 = 主预分频比为 64:1

**注 1 :** 如果 DISSCK = 0 , 必须将 SCKx 配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

**2 :** 如果 DISSDO = 0 , 必须将 SDOx 配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

**3 :** 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。

**4 :** 如果 SSEN = 1 , 必须将 SSx 配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

## 寄存器 15-3 : SPIxCON2 : SPIx 控制寄存器 2

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—
bit 15	bit 8						
U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	SPIFE	SPIBEN
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **FRMEN** : 帧 SPIx 支持位

1 = 使能帧 SPIx 支持

0 = 禁止帧 SPIx 支持

bit 14      **SPIFSD** : SSx 引脚上的帧同步脉冲方向控制位

1 = 帧同步脉冲输入 (从模式)

0 = 帧同步脉冲输出 (主模式)

bit 13      **SPIFPOL** : 帧同步脉冲极性位 (仅用于帧模式)

1 = 帧同步脉冲高电平有效

0 = 帧同步脉冲低电平有效

bit 12-2      未实现 : 读为 0

bit 1      **SPIFE** : 帧同步脉冲边沿选择位

1 = 帧同步脉冲与第一个位时钟同步

0 = 帧同步脉冲比第一个位时钟超前

bit 0      **SPIBEN** : 增强型缓冲器使能位

1 = 使能增强型缓冲器

0 = 禁止增强型缓冲器 (传统模式)

# PIC24FJ64GB004 系列

图 15-3： SPI 主 / 从连接（标准模式）

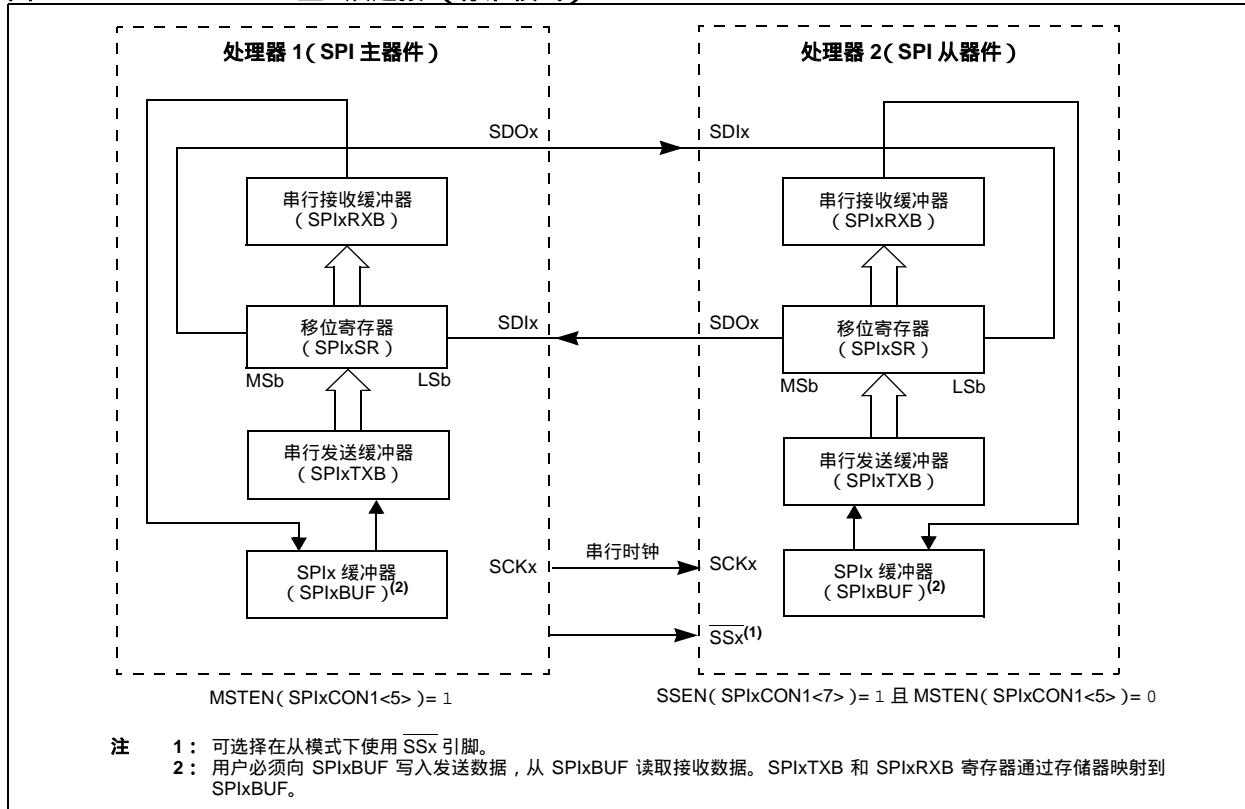


图 15-4： SPI 主 / 从连接（增强型缓冲模式）

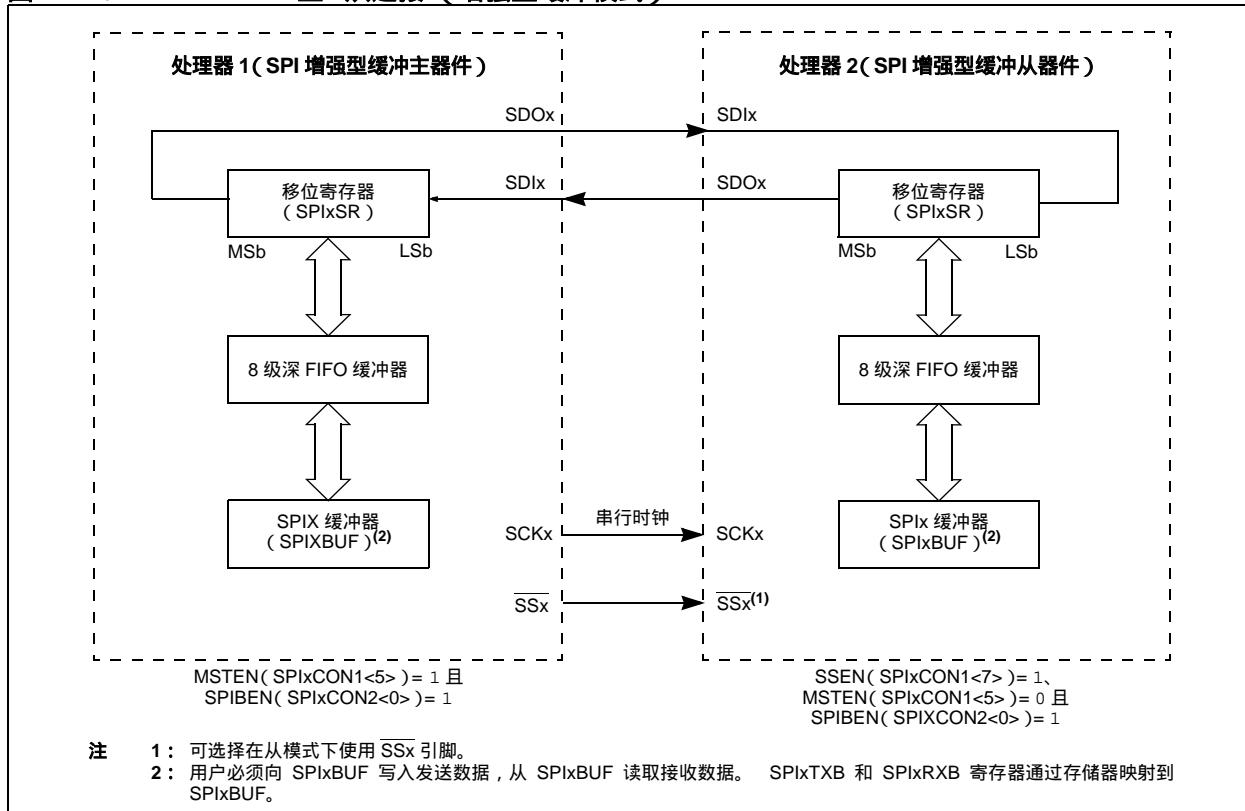


图 15-5： SPI 主器件、帧主器件连接图

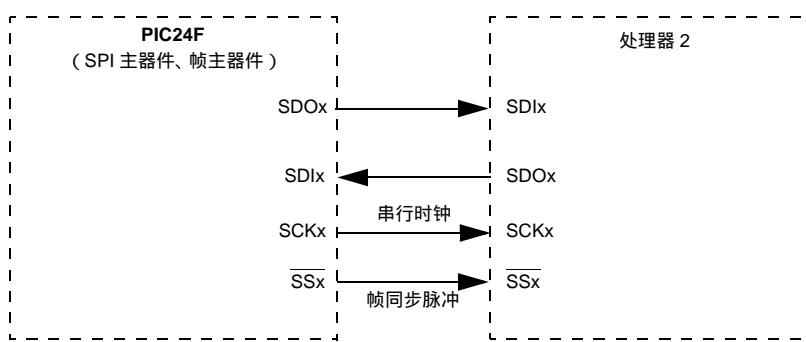


图 15-6： SPI 主器件、帧从器件连接图

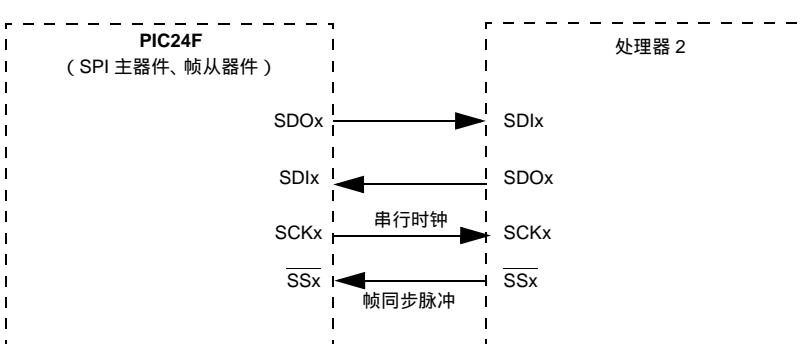


图 15-7： SPI 从器件、帧主器件连接图

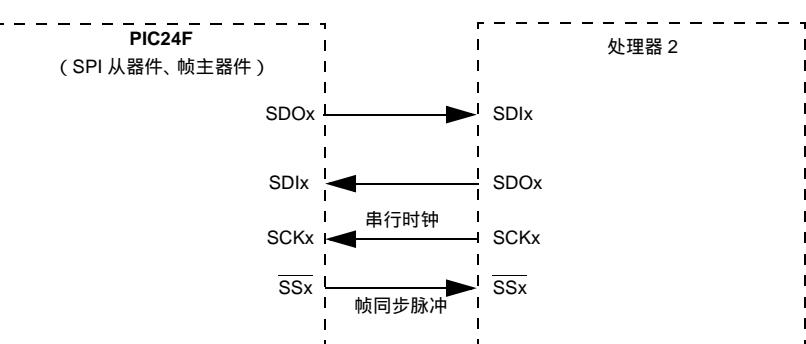
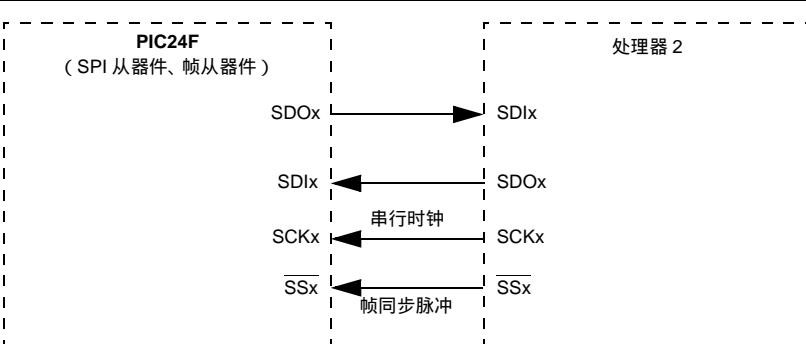


图 15-8： SPI 从器件、帧从器件连接图



# PIC24FJ64GB004 系列

公式 15-1： 器件速率和 SPI 时钟速度之间的关系<sup>(1)</sup>

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

注 1：基于  $FCY = FOSC/2$ ；禁止打盹模式和 PLL。

表 15-1： 采样 SCK 频率<sup>(1,2)</sup>

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  $FCY = FOSC/2$ ；禁止打盹模式和 PLL。

2：表中 SCKx 频率的单位为 kHz。

## 16.0 I<sup>2</sup>C<sup>TM</sup>

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 24 章“I<sup>2</sup>C<sup>TM</sup>”(DS39702A\_CN)。

I<sup>2</sup>C 模块是用来与其他外设或单片机通信的串行接口。这些外设器件可以是串行 EEPROM、显示驱动器和 A/D 转换器等。

I<sup>2</sup>C 模块支持以下特性：

- 独立的主和从逻辑
- 7 位和 10 位器件地址
- I<sup>2</sup>C 协议中定义的广播呼叫地址
- 时钟延长，可为处理器提供响应从器件数据请求的延时
- 支持 100 kHz 和 400 kHz 两种总线规范
- 可配置的地址掩码
- 多主器件模式以防止仲裁时报文丢失
- 总线重发器模式，允许作为从器件接收来自所有地址的所有报文
- 自动 SCL

模块的框图如图 16-1 所示。

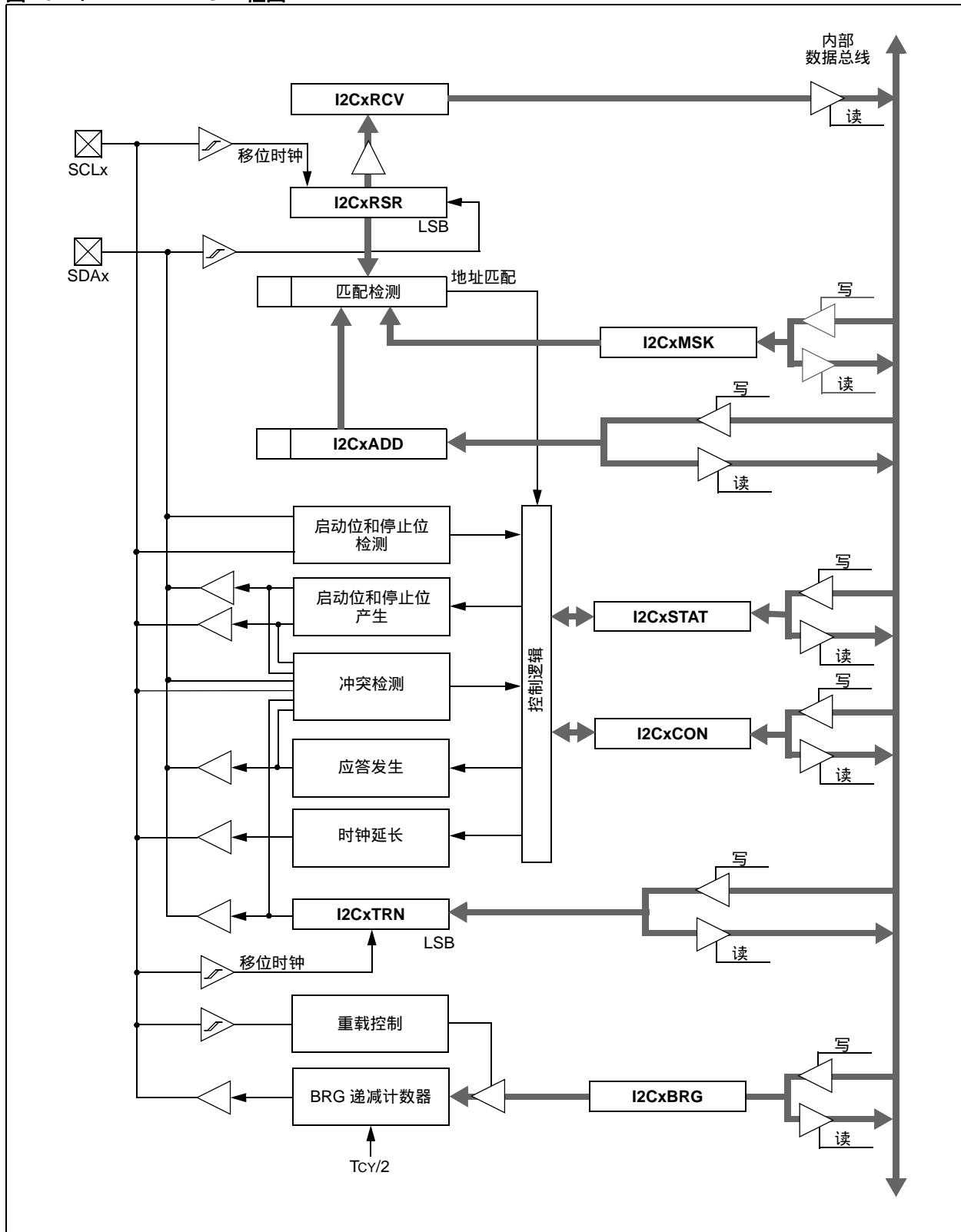
### 16.1 作为主器件在单主器件环境中通信

在主模式下发送报文的细节取决于要与主器件进行通信的器件的通信协议。通常，事件序列如下：

1. 在 SDAx 和 SCLx 上发出一个启动条件。
2. 发送一个 I<sup>2</sup>C 器件地址字节到从器件，表明要执行写操作。
3. 等待并验证从器件的应答。
4. 发送第一个数据字节（有时是命令）到从器件。
5. 等待并验证从器件的应答。
6. 发送串行存储器地址低字节到从器件。
7. 重复步骤 4 和步骤 5，直至所有数据字节发送结束。
8. 在 SDAx 和 SCLx 上发出一个重复启动条件
9. 发送一个器件地址字节到从器件，表明要执行读操作。
10. 等待并验证从器件的应答。
11. 使能主器件接收以接收串行存储器数据。
12. 在数据字节接收完毕时，产生 ACK 或 NACK 条件。
13. 在 SDAx 和 SCLx 上产生一个停止条件。

# PIC24FJ64GB004 系列

图 16-1 : I<sup>2</sup>C<sup>TM</sup> 框图



## 16.2 设置用作总线主器件时的波特率

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

**公式 16-1 : 计算波特率重载值<sup>(1,2)</sup>**

$$Fscl = \frac{FcY}{I2CxBRG + 1 + \frac{FcY}{10,000,000}}$$

或

$$I2CxBRG = \left( \frac{FcY}{Fscl} - \frac{FcY}{10,000,000} \right) - 1$$

**注 1 :** 基于  $FcY = Fosc/2$  ; 禁止打盹模式和 PLL。

**2 :** 这些时钟速率值仅供参考。实际的时钟速率与各个系统级参数有关。应在其目标应用中测量实际的时钟速率。

## 16.3 从地址掩码

I2CxMSK 寄存器 (见寄存器 16-3) 指定了 7 位和 10 位地址模式下某些“无关”地址位的位置。将 I2CxMSK 寄存器中某个特定位置 1 (= 1) , 不论相应的地址位是 0 还是 1 , 从模块都会做出响应。例如 , 当将 I2CxMSK 设置为 00100000 时 , 从模块将检测两个地址 0000000 和 0100000。

为了使能地址掩码 , 必须通过将 IPMIEN 位 (I2CxCON<11>) 清零来禁止智能外设管理接口 (Intelligent Peripheral Management Interface , IPMI)。

**注 :** 由于修改了 I<sup>2</sup>C<sup>TM</sup> 协议 , 表 16-2 中的地址将被保留 , 并且在从模式下不会被应答。这同样适用于包含任意这些地址的所有地址掩码设置。

**表 16-1 : I<sup>2</sup>C<sup>TM</sup> 时钟速率<sup>(1,2)</sup>**

所需的系统 Fscl	FcY	I2CxBRG 值		实际 Fscl
		(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  $FcY = Fosc/2$  ; 禁止打盹模式和 PLL。

**2 :** 这些时钟速率值仅供参考。实际的时钟速率与各个系统级参数有关。应在其目标应用中测量实际的时钟速率。

**表 16-2 : I<sup>2</sup>C<sup>TM</sup> 保留地址<sup>(1)</sup>**

从地址	R/W 位	说明
0000 000	0	广播呼叫地址 <sup>(2)</sup>
0000 000	1	启动字节
0000 001	x	Cbus 地址
0000 010	x	保留
0000 011	x	保留
0000 1xx	x	HS 模式主代码
1111 1xx	x	保留
1111 0xx	x	10 位从地址高字节 <sup>(3)</sup>

**注 1 :** 上述地址位永远不会导致地址匹配 , 无论地址掩码的设置如何。

**2 :** 仅当 GCEN = 1 时 , 才会应答地址。

**3 :** 仅在 10 位寻址模式下才会与地址的高字节匹配。

# PIC24FJ64GB004 系列

## 寄存器 16-1 : I2CxCON : I2Cx 控制寄存器

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

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

图注 : HC = 可由硬件清零的位

R = 可读位 W = 可写位 U = 未实现位 , 读为 0

-n = 上电复位时的值 1 = 置 1 0 = 清零 X = 未知

- bit 15 **I2CEN** : I2Cx 使能位  
1 = 使能 I2Cx 模块并将 SDAx 和 SCLx 引脚配置为串行端口引脚  
0 = 禁止 I2Cx 模块。所有 I<sup>2</sup>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 = 模块在空闲模式下继续工作
- bit 12 **SCLREL** : SCLx 释放控制位 (作为 I<sup>2</sup>C 从器件工作时 )  
1 = 释放 SCLx 时钟  
0 = 保持 SCLx 时钟为低电平 (时钟延长)  
**如果 STREN = 1 :**  
该位为 R/W (即软件可写入 0 来启动时钟延长或写入 1 来释放时钟)。在从器件发送开始或接收结束时 , 由硬件清零。  
**如果 STREN = 0 :**  
该位为 R/S (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 模式
- bit 10 **A10M** : 10 位从器件寻址位  
1 = I2CxADD 是一个 10 位从地址  
0 = I2CxADD 是一个 7 位从地址
- bit 9 **DISSLW** : 禁止压摆率控制位  
1 = 禁止压摆率控制  
0 = 使能压摆率控制
- 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<sup>2</sup>C 从器件工作时 )  
1 = 允许在 I2CxRSR 接收到广播呼叫地址时产生中断 (已使能模块接收 )  
0 = 禁止广播呼叫地址
- bit 6 **STREN** : SCLx 时钟延长使能位 (作为 I<sup>2</sup>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 : I2CxCON : I<sup>2</sup>Cx 控制寄存器 (续)

bit 5	<b>ACKDT</b> : 应答数据位 (作为 I <sup>2</sup> C 主器件工作时, 适用于主器件接收操作。) 当软件启动应答序列时将发送的值。 1 = 在应答时发送 NACK 0 = 在应答时发送 ACK
bit 4	<b>ACKEN</b> : 应答序列使能位 (作为 I <sup>2</sup> C 主器件工作时, 适用于主器件接收操作。) 1 = 在 SDAx 和 SCLx 引脚上启动应答序列, 并发送 ACKDT 数据位。在主器件应答序列结束时由硬件清零。 0 = 没有启动应答序列
bit 3	<b>RCEN</b> : 接收使能位 (作为 I <sup>2</sup> C 主器件工作时) 1 = 使能 I <sup>2</sup> C 接收模式。在主器件接收完数据字节的 8 位后由硬件清零。 0 = 没有启动接收序列
bit 2	<b>PEN</b> : 停止条件使能位 (作为 I <sup>2</sup> C 主器件工作时) 1 = 在 SDAx 和 SCLx 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 = 没有发起停止条件
bit 1	<b>RSEN</b> : 重复启动条件使能位 (作为 I <sup>2</sup> C 主器件工作时) 1 = 在 SDAx 和 SCLx 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 = 没有发起重复启动条件
bit 0	<b>SEN</b> : 启动条件使能位 (作为 I <sup>2</sup> C 主器件工作时) 1 = 在 SDAx 和 SCLx 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 = 没有发起启动条件

# PIC24FJ64GB004 系列

## 寄存器 16-2 : I2CxSTAT : I2Cx 状态寄存器

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

R/C-0, HS	R/C-0, HS	R-0, HSC	R/C-0, HSC	R/C-0, HSC	R-0, HSC	R-0, HSC	R-0, HSC
IWCOL	I2COV	D/A	P	S	R/W	RBF	TBF
bit 7							bit 0

图注 :	C = 可清零位	HS = 可由硬件置 1 的位	HSC = 可由硬件置 1/ 清零的位
R = 可读位	W = 可写位	U = 未实现位 , 读为 0	
-n = 上电复位时的值	1 = 置 1	0 = 清零	x = 未知

bit 15 **ACKSTAT** : 应答状态位

1 = 上次检测到 NACK

0 = 上次检测到 ACK

在从器件应答结束时由硬件置 1 或清零。

bit 14 **TRSTAT** : 发送状态位

(作为 I<sup>2</sup>C 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

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。当检测到停止条件时由硬件清零。

bit 8 **ADD10** : 10 位地址状态位

1 = 10 位地址匹配

0 = 10 位地址不匹配

当与匹配的 10 位地址的第 2 个字节匹配时由硬件置 1。当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 I<sup>2</sup>C 模块忙 , 尝试写 I2CxTRN 寄存器的操作失败。

0 = 未发生冲突

当总线忙时写 I2CxTRN 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 I2CxRCV 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 I2CxRSR 传输到 I2CxRCV 时由硬件置 1 (由软件清零)。

bit 5 **D/A** : 数据 / 地址位 (作为 I<sup>2</sup>C 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

器件地址匹配时由硬件清零。在发送完成后或接收从器件字节时由硬件置 1。

## 寄存器 16-2 : I2CxSTAT : I2Cx 状态寄存器 (续)

- bit 4      **P** : 停止位  
1 = 表示上次检测到停止位  
0 = 表示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。
- bit 3      **S** : 启动位  
1 = 表示上次检测到启动位 (或重复启动位)  
0 = 表示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 I<sup>2</sup>C 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 I<sup>2</sup>C 器件地址字节后由硬件置 1 或清零。
- bit 1      **RBF** : 接收缓冲器满状态位  
1 = 接收完成, I2CxRCV 为满  
0 = 接收未完成, I2CxRCV 为空  
用接收到的字节写 I2CxRCV 时由硬件置 1。当用软件读 I2CxRCV 时由硬件清零。
- bit 0      **TFB** : 发送缓冲器满状态位  
1 = 发送正在进行, I2CxTRN 为满  
0 = 发送完成, I2CxTRN 为空  
用软件写 I2CxTRN 时由硬件置 1。数据发送完成时由硬件清零。

# PIC24FJ64GB004 系列

寄存器 16-3 : I2CxMSK : I2Cx 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 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| AMSK7 | AMSK6 | AMSK5 | AMSK4 | AMSK3 | AMSK2 | AMSK1 | AMSK0 |
| bit 7 |       |       |       |       |       |       | bit 0 |

**图注 :**

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-10 未实现 : 读为 0

bit 9-0 AMSK<9:0> : 地址位 x 的掩码选择位

1 = 使能输入报文地址 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

## 17.0 通用异步收发器 (UART)

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 21 章 “UART” (DS39708A\_CN)。

通用异步收发器 (Universal Asynchronous Receiver Transmitter, UART) 模块是 PIC24F 系列器件提供的串行 I/O 模块之一。UART 是可以和外设 (例如个人电脑、LIN/J2602、RS-232 和 RS-485 接口) 通信的全双工异步系统。UART 模块还通过 UxCTS 和 UxRTS 引脚支持硬件流控制，该模块中还包含有 IrDA® 编码器和解码器。

UART 模块的主要特性有：

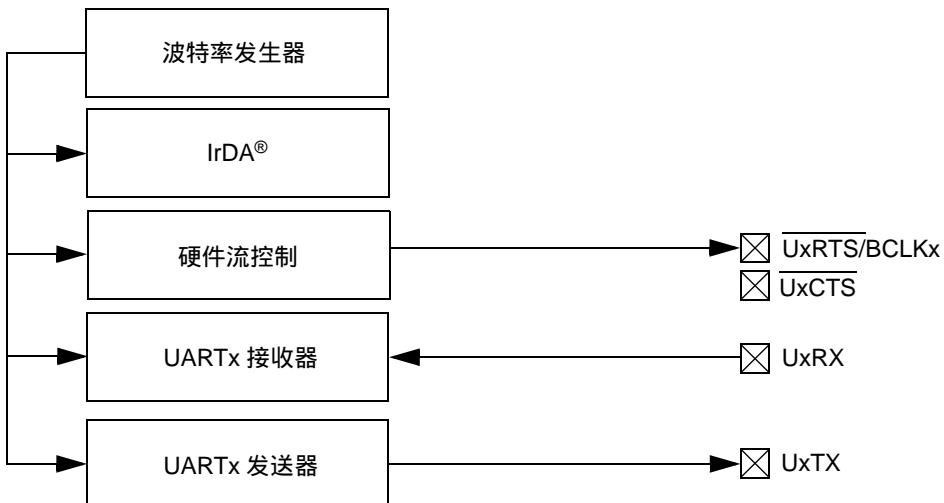
- 通过 UxTX 和 UxRX 引脚进行全双工 8 位或 9 位数据传输
- 偶校验、奇校验或无奇偶校验选项 (对于 8 位数据)
- 一个或两个停止位
- 通过 UxCTS 和 UxRTS 引脚实现硬件流控制

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

图 17-1 显示了 UART 的简化框图。UART 模块由以下重要硬件组成：

- 波特率发生器
- 异步发送器
- 异步接收器

图 17-1：UART 简化框图



**注：**UART 输入和输出必须在使用之前分配给可用的 RPn 引脚。更多信息，请参见第 10.4 节 “外设引脚选择 (PPS)”。

## 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。UxBRG 寄存器控制一个独立运行的 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 : **BRGH = 0 时的 UART 波特率<sup>(1,2)</sup>**

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

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

- 注 1: FCY 表示指令周期的时钟频率 (FOSC/2)。  
2: 基于 FCY = FOSC/2 ; 禁止打盹模式和 PLL。

例 17-1 给出了下列条件下的波特率误差计算 :

- FCY = 4 MHz
- 目标波特率 = 9600

### 例 17-1 : 波特率误差计算 (BRGH = 0)<sup>(1)</sup>

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

UxBRG 值的计算方法 :

$$\begin{aligned}\text{UxBRG} &= ((\text{FCY}/\text{目标波特率})/16) - 1 \\ \text{UxBRG} &= ((4000000/9600)/16) - 1 \\ \text{UxBRG} &= 25\end{aligned}$$

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

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

- 注 1: 基于 FCY = FOSC/2 , 禁止打盹模式和 PLL。

可能的最大波特率 (BRGH = 0) 是 FCY/16 (当 UxBRG = 0 时), 可能的最小波特率是 FCY/(16 \* 65536)。公式 17-2 给出了 BRGH = 1 时计算波特率的公式。

公式 17-2 : **BRGH = 1 时的 UART 波特率<sup>(1,2)</sup>**

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

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

- 注 1: FCY 表示指令周期的时钟频率。  
2: 基于 FCY = FOSC/2 ; 禁止打盹模式和 PLL。

可能的最大波特率 (BRGH = 1) 是 FCY/4 (当 UxBRG = 0 时), 可能的最小波特率是 FCY/(4 \* 65536)。

向 UxBRG 寄存器中写入新值会导致 BRG 定时器复位 (清零)。这保证了 BRG 在产生新的波特率之前不需要等待定时器溢出。

## 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 置 1 (置 1 后两个周期内产生发送中断)。
4. 将数据字节写入 UxTXREG 字的低字节。该数据字节将被立即传送给发送移位寄存器 (Transmit Shift Register, 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. 另外，数据字节也可在 UTXEN = 0 时被传送，且随后用户可将 UTXEN 置 1。由于波特率时钟是从清零状态启动的，所以该行为能立即开始发送串行比特流。
6. 将根据中断控制位 UTXISELx 的设置产生发送中断。

## 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” 所描述的那样)。
2. 使能 UART。
3. 将 UTXEN 置 1 (产生发送中断)。
4. 只将 16 位值写入 UxTXREG。
5. 向 UxTXREG 写入一个字可触发 9 位数据向 TSR 传送。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. 将根据中断控制位 UTXISELx 的设置产生发送中断。

## 17.4 间隔和同步发送序列

下面的序列将发送一个由间隔字符和其后的自动波特率同步字节组成的报文帧头。

1. 将 UART 配置为所需的模式。
2. 将 UTXEN 和 UTXBRK 置 1，以设置间隔字符。
3. 将一个“虚拟”字符装入 UxTXREG 寄存器中以启动发送 (该值会被忽略)。
4. 向 UxTXREG 写入 55h，这会将同步字符装入发送 FIFO 中。
5. 当间隔字符发送完成后，由硬件将 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” 所描述的那样)。
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。必须用软件将 OERR 位复位。
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 UARTx 请求发送 (UxRTS) 是两个与 UART 模块有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。这两个引脚控制数据终端设备 (Data Terminal Equipment, DTE) 之间的数据发送和接收。通过 UxMODE 寄存器的 UEN<1:0> 位来配置这两个引脚。

## 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，支持外部 IrDA 编码器和解码器 (支持传统模块)；另一种是完全实现的 IrDA 编码器和解码器。注意，因为 IrDA 模式需要 16 倍频的波特率时钟，所以它们仅在 BRGH 位 (UxMODE<3>) 为 0 时才能工作。

### 17.7.1 支持外部 IRDA 的 IRDA 时钟输出

为了支持外部 IrDA 编码器和解码器，可将 BCLKx 引脚 (与 UxRTS 引脚相同) 配置为产生 16 倍频的波特率时钟。当使能了 UART 模块且 UEN<1:0> = 11 时，BCLKx 引脚将输出 16 倍频的波特率时钟，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 (即 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。

# PIC24FJ64GB004 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	R/W-0
UARTEN <sup>(1)</sup>	—	USIDL	IREN <sup>(2)</sup>	RTSMD	—	UEN1	UENO
bit 15	bit 8						

R/W-0, HC	R/W-0	R/W-0, HC	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAKE	LPBACK	ABAUD	RXINV	BRGH	PDSEL1	PDSEL0	STSEL
bit 7	bit 0						

图注 : HC = 可由硬件清零的位

R = 可读位 W = 可写位 U = 未实现位 , 读为 0

-n = 上电复位时的值 1 = 置 1 0 = 清零 x = 未知

bit 15 **UARTEN** : UARTx 使能位<sup>(1)</sup>

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 ; 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 = 模块在空闲模式下继续工作

bit 12 **IREN** : IrDA® 编码器和解码器使能位<sup>(2)</sup>

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** : UxRTS 引脚模式选择位

1 = UxRTS 引脚处于单工模式  
0 = UxRTS 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

11 = 使能并使用 UxTX、UxRX 和 BCLKx 引脚 ; UxCTS 引脚由端口锁存器控制  
10 = 使能并使用 UxTX、UxRX、UxCTS 和 UxRTS 引脚  
01 = 使能并使用 UxTX、UxRX 和 UxRTS 引脚 ; UxCTS 引脚由端口锁存器控制  
00 = 使能并使用 UxTX 和 UxRX 引脚 ; UxCTS 和 UxRTS/BCLKx 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时唤醒的使能位

1 = UARTx 将继续采样 UxRX 引脚 ; 在出现下降沿时产生中断 ; 在随后的上升沿由硬件清零  
0 = 未使能唤醒

bit 6 **LPBACK** : UARTx 环回模式选择位

1 = 使能环回模式  
0 = 禁止环回模式

bit 5 **ABAUD** : 自动波特率使能位

1 = 使能在下一个字符传输过程中对波特率进行测量——需要接收到同步字段 ( 55h ) ; 完成时由硬件清零  
0 = 禁止或已完成波特率测量

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 :** 如果 UARTEN = 1 , 外设的输入和输出必须配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

**2 :** 此功能仅适用于 16 倍 BRG 模式 ( BRGH = 0 )。

## 寄存器 17-1 : UxMODE : UARTx 模式寄存器 (续)

bit 3	<b>BRGH</b> : 高波特率使能位 1 = 高速模式 (每个位周期内产生 4 个 BRG 时钟信号) 0 = 标准模式 (每个位周期内产生 16 个 BRG 时钟信号)
bit 2-1	<b>PDSEL&lt;1:0&gt;</b> : 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	<b>STSEL</b> : 停止位选择位 1 = 2 个停止位 0 = 1 个停止位

注 1：如果  $\text{UARTEN} = 1$ ，外设的输入和输出必须配置给可用的 RPn 引脚。更多信息，请参见第 10.4 节“外设引脚选择 (PPS)”。  
2：此功能仅适用于 16 倍 BRG 模式 ( $\text{BRGH} = 0$ )。

# PIC24FJ64GB004 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	R-1
UTXISEL1	UTXINV <sup>(1)</sup>	UTXISEL0	—	UTXBRK	UTXEN <sup>(2)</sup>	UTXBF	TRMT
bit 15	bit 8						

R/W-0	R/W-0	R/W-0	R-1	R-0	R-0	R/C-0	R-0
URXISEL1	URXISEL0	ADDEN	RIDLE	PERR	FERR	OERR	URXDA
bit 7	bit 0						

图注 :	C = 可清零位	HC = 可由硬件清零的位
R = 可读位	W = 可写位	U = 未实现位 , 读为 0
-n = 上电复位时的值	1 = 置 1	0 = 清零
		x = 未知

- bit 15,13      **UTXISEL<1:0>** : 发送中断模式选择位  
11 = 保留 ; 不要使用  
10 = 当一个字符被传送到发送移位寄存器 ( TSR ) 导致发送缓冲器为空时 , 产生中断  
01 = 当最后一个字符被移出发送移位寄存器时产生中断 ; 发送操作全部完成  
00 = 当一个字符被传送到发送移位寄存器 ( 前提是发送缓冲器中至少还有一个字符可供发送 ) 时产生中断
- bit 14      **UTXINV** : IrDA® 编码器发送极性翻转位<sup>(1)</sup>  
IREN = 0 :  
1 = UxTX 空闲状态为 0  
0 = UxTX 空闲状态为 1  
IREN = 1 :  
1 = UxTX 空闲状态为 1  
0 = UxTX 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——起始位 , 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<sup>(2)</sup>  
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 , 缓冲器复位 ; 端口控制 UxTX 引脚
- bit 9      **UTXBF** : 发送缓冲器满状态位 ( 只读 )  
1 = 发送缓冲器满  
0 = 发送缓冲器未满 , 至少还可写入一个字符
- bit 8      **TRMT** : 发送移位寄存器空位 ( 只读 )  
1 = 发送移位寄存器为空 , 同时发送缓冲器为空 ( 上一次发送已完成 )  
0 = 发送移位寄存器非空 , 发送正在进行或等待发送
- bit 7-6      **URXISEL<1:0>** : 接收中断模式选择位  
11 = 当 RSR 传输使接收缓冲器为满时 ( 即 , 有 4 个数据字符 ) , 中断标志位置 1  
10 = 当 RSR 传输使接收缓冲器为 3/4 满时 ( 即 , 有 3 个数据字符 ) , 中断标志位置 1  
0x = 当接收缓冲器接收到来自 RSR 的任何字符时 , 中断标志位置 1 。接收缓冲器有一个或多个字符

- 注 1 : 仅当使能了 IrDA 编码器 ( IREN = 1 ) 时 , 该位的值才影响模块的发送属性。  
2 : 如果 UARTEN = 1 , 外设的输入和输出必须配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “ 外设引脚选择 ( PPS ) ”。

## 寄存器 17-2 : UxSTA : UARTx 状态和控制寄存器 (续)

bit 5	<b>ADDEN</b> : 地址字符检测位 (接收数据的第 8 位 = 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	<b>RIDLE</b> : 接收器空闲位 (只读) 1 = 接收器空闲 0 = 接收器工作
bit 3	<b>PERR</b> : 奇偶校验错误状态位 (只读) 1 = 检测到当前字符 (在接收 FIFO 顶部的字符) 的奇偶校验错误 0 = 没有检测到奇偶校验错误
bit 2	<b>FERR</b> : 帧错误状态位 (只读) 1 = 检测到当前字符 (在接收 FIFO 顶部的字符) 的帧错误 0 = 没有检测到帧错误
bit 1	<b>OERR</b> : 接收缓冲器溢出错误状态位 (只读 / 清零) 1 = 接收缓冲器已经溢出 0 = 接收缓冲器没有溢出。清零先前置 1 的 OERR 位 ( $1 \rightarrow 0$ 的跳变) 会使接收缓冲器和 RSR 复位为空状态
bit 0	<b>URXDA</b> : 接收缓冲器中数据可用位 (只读) 1 = 接收缓冲器中有数据，有至少一个字符可被读取 0 = 接收缓冲器为空

- 注 1：仅当使能了 IrDA 编码器 (IREN = 1) 时，该位的值才影响模块的发送属性。  
2：如果 UARTEN = 1，外设的输入和输出必须配置给可用的 RPn 引脚。更多信息，请参见第 10.4 节“外设引脚选择 (PPS)”。

# **PIC24FJ64GB004 系列**

---

---

注：

## 18.0 带 ON-THE-GO 支持的通用串行总线 (USB OTG)

**注：** 本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 27 章“USB On-The-Go (OTG)”(DS39721A\_CN)。

PIC24FJ64GB004 系列器件包含一个兼有全速和低速两种工作模式的 On-The-Go (OTG) USB 串行接口引擎 (Serial Interface Engine, SIE)。OTG 功能使得器件可以用作 USB 外设或具有有限主机功能的 USB 嵌入式主机。OTG 功能还可使器件通过 OTG 的主机协商协议 (Host Negotiation Protocol, HNP) 从设备动态地切换到主机。

更多关于 OTG 操作的详细信息，请参见 USB-IF 发布的“On-The-Go Supplement to the USB 2.0 Specification”(USB 2.0 规范 On-The-Go 补充信息)。更多关于 USB 操作的详细信息，请参见“Universal Serial Bus Specification v2.0”(通用串行总线规范 2.0 版)。

USB OTG 模块提供以下特性：

- 设备和主机模式下的 USB 功能与应用程序控制模式下的 OTG 功能切换
- 使用内部振荡器时精度为 0.25%—无需外部晶振
- 软件可选的模块速度：全速 (12 Mbps) 或低速 (1.5 Mbps，仅在主机模式下可用)
- 支持所有四种 USB 传输类型：控制、中断、批量和同步
- 共 32 个单向端点可构成 16 个双向端点
- 用于数据 RAM 快速操作的 DMA 接口
- 最多 16 个单向端点传输队列，无需服务
- 集成的片内 USB 收发器，通过数字接口支持片外收发器
- 使用片内比较器的集成 VBUS 生成和升压生成，通过数字接口支持外部 VBUS 比较器和稳压器
- 片上总线上拉和下拉电阻配置

USB OTG 模块的简化框图如图 18-1 所示。

USB OTG 模块可用作 USB 外设或 USB 主机，并可在软件控制下在设备和主机模式间动态切换。两种模式使用同一数据路径和缓冲区描述符来收发数据。

讨论 USB 操作时，本章使用以控制器为中心的命名法说明单片机和 USB 之间的数据传输方向。Rx (接收) 用于说明将数据从 USB 移动到单片机的传输，Tx (发送) 用于说明将数据从单片机移动到 USB 的传输。表 18-1 显示了这种命名法的数据方向和 USB 令牌交换之间的关系。

**表 18-1：USB 主机或目标设备以控制器为中心的数据方向**

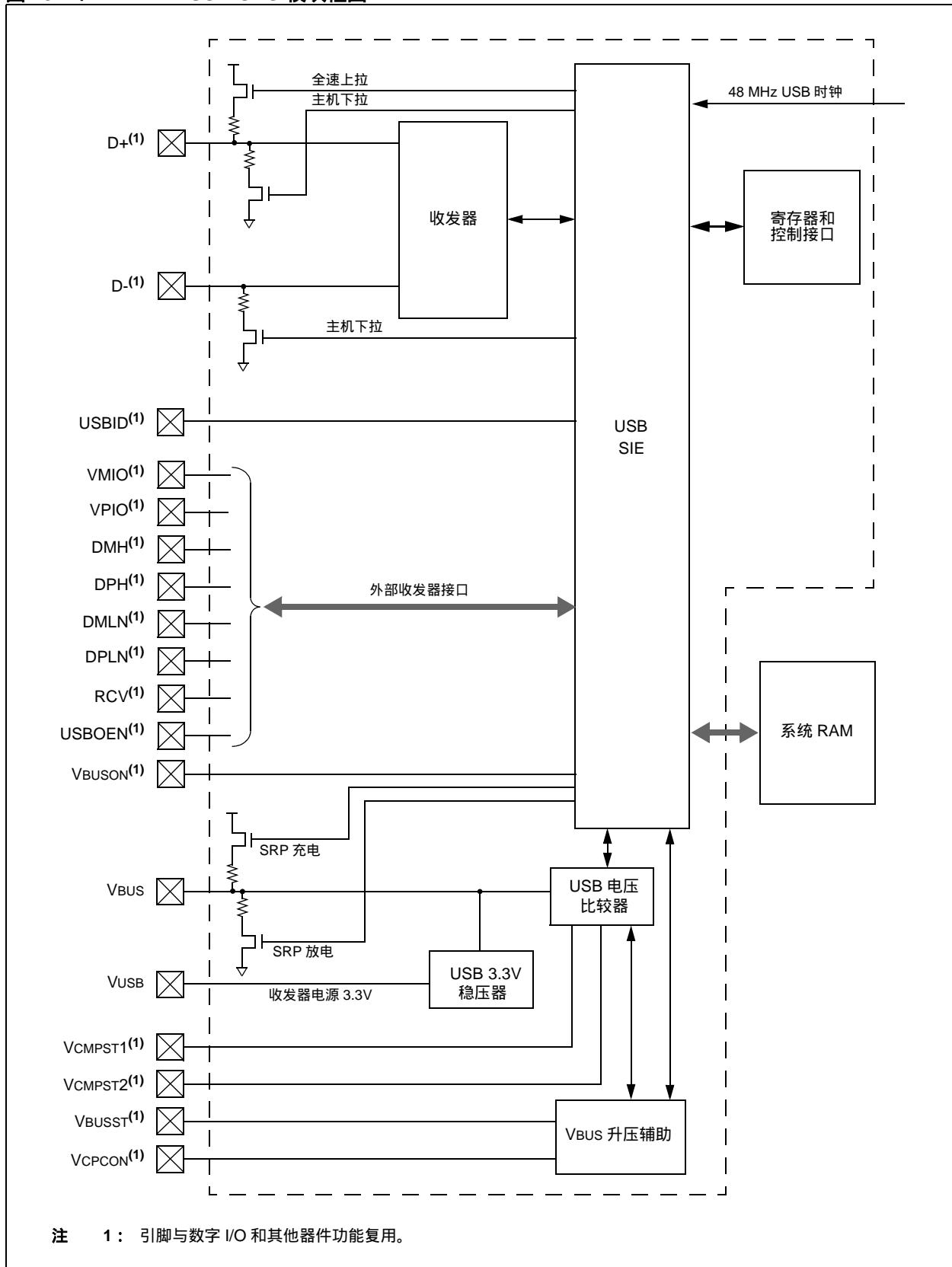
USB 模式	方向	
	Rx	Tx
设备	输出或设置	输入
主机	输入	输出或设置

本章将介绍在应用中实现 USB OTG 功能所需的最基本的操作。对 USB 协议及其 OTG 补充信息的完整详细说明不在本数据手册的讨论范围内。本数据手册建立在用户对 USB 架构以及协议最新版本已有了基本了解的基础上。

这里并没有给出正确 USB 操作（例如设备枚举）所需的全部步骤。建议应用开发人员使用适当的设备驱动程序来实现所有的必需功能。Microchip 提供了许多特定于应用的资源，例如 USB 固件和驱动程序支持。可访问 [www.microchip.com](http://www.microchip.com) 以获取最新的固件和驱动程序支持。

# PIC24FJ64GB004 系列

图 18-1：USB OTG 模块框图



## 18.1 硬件配置

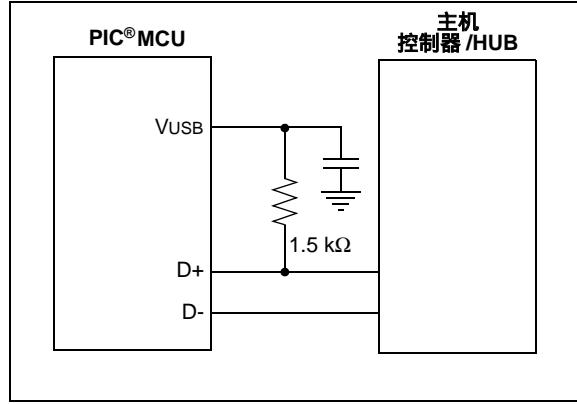
### 18.1.1 器件模式

#### 18.1.1.1 D+ 上拉电阻

PIC24FJ64GB004 系列器件在 D+ 线上有一个内置的  $1.5\text{ k}\Omega$  电阻，当单片机工作在设备模式下时可用。这用于以信号形式告知外部主机：器件工作在全速设备模式。将 USBEN (U1CON<0>) 位置 1 可使能全速设备模式。如果 OTGEN (U1OTGCON<2>) 位置 1，那么可通过 DPPULUP (U1OTGCON<7>) 位使能 D+ 上拉。

或者，也可以在 D+ 上使用外部电阻，如图 18-2 所示。

**图 18-2：** 全速器件模式下的外部上拉



#### 18.1.1.2 电源模式

许多 USB 应用可能有多组不同的电源要求和配置。最常遇见的电源模式有：

- 仅总线电源
- 仅自供电
- 双电源且自供电为主

仅总线电源模式（图 18-3）是最简单的方法。应用的所有电源都从 USB 汲取。

要满足 USB 2.0 规范的涌流要求，VBUS 和地之间的总有效电容不得大于  $10\text{ }\mu\text{F}$ 。

在 USB 暂挂模式下，器件从 USB 电缆的 5V VBUS 线消耗的电流不得大于  $2.5\text{ mA}$ 。在 USB 暂挂模式期间，D+ 或 D- 上拉电阻必须保持有效，其消耗的电流为允许的暂挂电流的一部分。

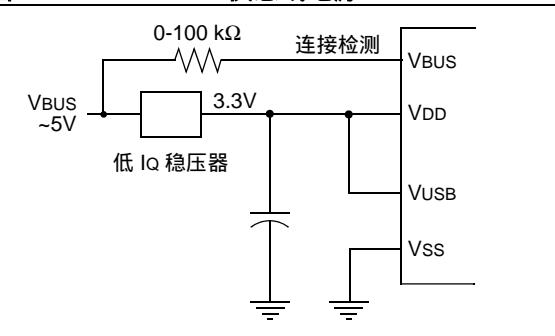
在仅自供电模式（图 18-4）下，USB 应用有自己的电源，从 USB 消耗的电源极小。注意，将添加一个连接指示，表示何时连接 USB 以及主机何时有效地为 VBUS 供电。

要满足规范要求，在主机将 VBUS 有效地驱动为高电平之前不应使能 USB 模块以及 D+ 或 D- 上拉电阻。可使用一个 5.5V 容差的 I/O 引脚来实现此目的。

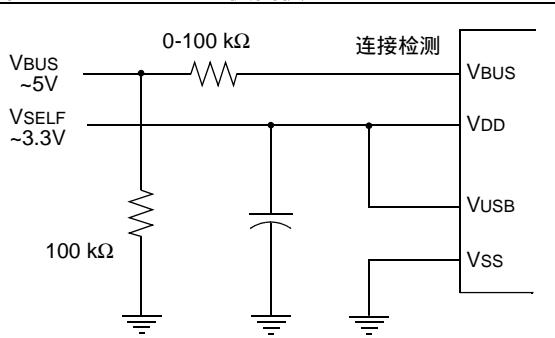
应用不应向 USB 电缆的 5V VBUS 引脚汲取任何电流。

双电源且自供电为主的选项（图 18-5）允许应用主要使用内部电源，但是在内部电源不可用的情况下可切换到从 USB 供电。双电源器件还必须满足前面描述的涌流和暂挂模式电流的特殊要求，并且在 VBUS 驱动为高电平之前不得使能 USB 模块。

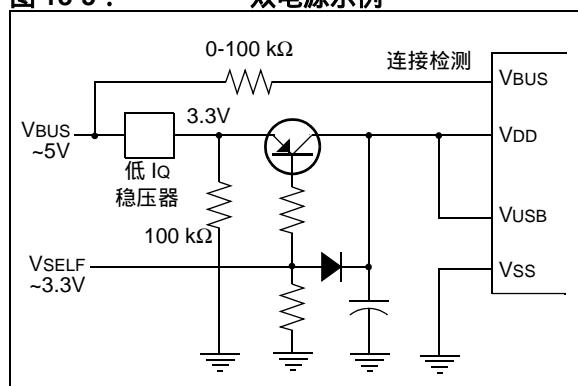
**图 18-3：** 仅总线电源



**图 18-4：** 仅自供电



**图 18-5：** 双电源示例



# PIC24FJ64GB004 系列

## 18.1.2 主机和 OTG 模式

### 18.1.2.1 D+ 和 D- 下拉电阻

PIC24FJ64GB004 系列器件在 D+ 和 D- 线上有内置的  $15\text{ k}\Omega$  下拉电阻。这两个电阻串联起来，以信号形式告知总线：单片机工作在主机模式下。将 HOSTEN (U1CON<3>) 位置 1 可使能主机模式。如果 OTGEN (U1OTGCON<2>) 位置 1，那么将 DPPULDWN 和 DMPULDWN (U1OTGCON<5:4>) 位置 1 可使能这两个下拉电阻。

### 18.1.2.2 电源配置

在主机模式下以及主机模式下的 On-the-Go 操作，USB 2.0 规范要求主机应用在 VBUS 上提供电源。由于单片机

以低于 VBUS 的电压运行，而且无法提供足够的电流，因此必须提供单独的电源。

当应用始终在主机模式下工作时，可使用一个简单的电路在总线上为 VBUS 供电并调节总线上的电流（图 18-6）。对于 OTG 操作，必须能够根据需要开启或关闭 VBUS，因为单片机会在器件模式和主机模式之间切换。图 18-7 给出了使用外部电荷泵的典型示例。

图 18-6： 主机接口示例

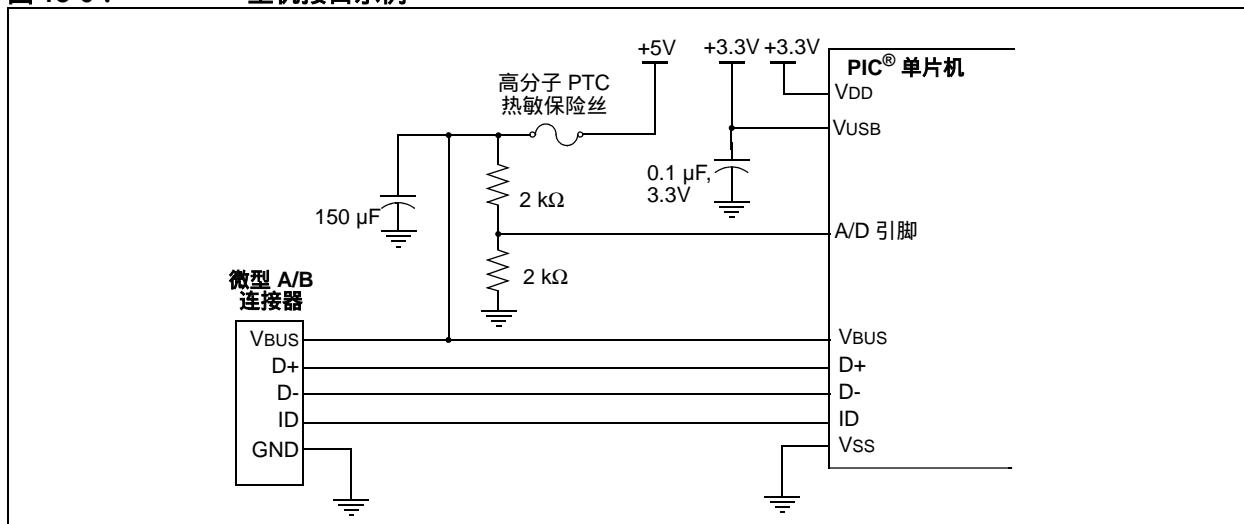
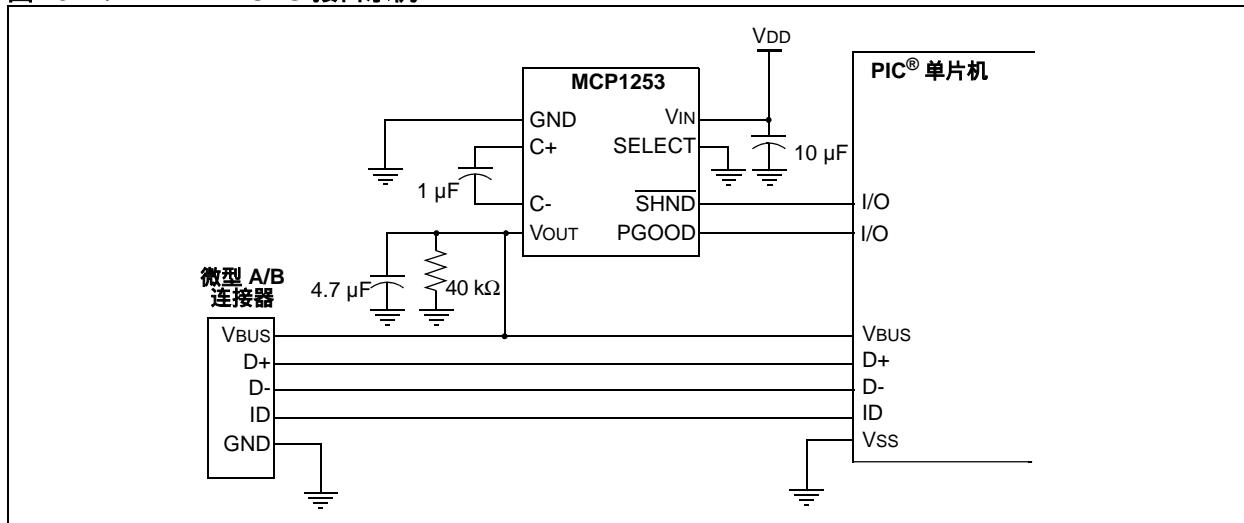


图 18-7： OTG 接口示例



### 18.1.2.3 使用外部器件生成 VBUS 电压

作为 USB 主机（即作为 OTG 配置中的 A 设备或嵌入式主机）工作时，必须为连接的器件提供 VBUS。PIC24FJ64GB004 系列器件具有一个内部 VBUS 升压辅助工具，帮助从电路板上的可用电压产生所需的 5V VBUS。该辅助工具由一个简单的 PWM 输出（用于控制开关电源）和内置比较器（用于监视输出电压和限制电流）组成。

要使能电压生成，请执行以下操作：

1. 验证 USB 模块是否上电 (U1PWRC<0> = 1) 以及 VBUS 放电是否被禁止 (U1OTGCON<0> = 0)。
2. 根据需要设置 PWM 周期 (U1PWMRRS<7:0>) 和占空比 (U1PWMRRS<15:8>)。
3. 根据外部电路的配置使用 PWMPOL 位 (U1PWMCON<9>) 选择输出信号的所需极性。
4. 使用 VBUSCHG 位 (U1OTGCON<1>) 选择所需的目标电压。
5. 通过将 CNTEN 位 (U1PWMCON<8>) 设置为 1 使能 PWM 计数器。
6. 通过将 PWMEN 位 (U1PWMCON<15>) 设置为 1 使能 PWM 模块。
7. 使能 VBUS 发生电路 (U1OTGCON<3> = 1)。

**注：**本节描述了 VBUS 电压发生和控制的一般过程。更多示例请参阅《PIC24F 系列参考手册》。

### 公式 18-1：估算 USB 收发器的电流消耗

$$I_{XCVR} = \frac{(40 \text{ mA} \cdot V_{USB} \cdot P_{ZERO} \cdot PIN \cdot L_{CABLE})}{(3.3V \cdot 5m)} + I_{PULLUP}$$

**图注：** V<sub>USB</sub>——施加到 V<sub>USB</sub> 引脚上的电压（单位为 V，3.0V 至 3.6V）。

P<sub>ZERO</sub>——PIC<sup>®</sup> 单片机发送的 IN（输入）通信中值为 0 的位所占的百分比（单位为 %）。

PIN——用于 IN 通信的总线带宽占总总线带宽的百分比（单位为 %）。

L<sub>CABLE</sub>——USB 电缆的长度（单位为 m）。USB 2.0 规范要求全速应用不得使用长度超过 5m 的电缆。

I<sub>PULLUP</sub>——1.5 kΩ 的标称上拉电阻（被使能时）必须为 USB 电缆提供的电流。

### 18.1.3 使用外部接口

某些应用可能需要将 USB 接口与系统的其余部分隔离。PIC24FJ64GB004 系列器件包含一个可与外部 USB 收发器通信的完整接口，并通过该接口控制外部 USB 收发器，包括数据线上拉和下拉的控制。也可针对不同的 VBUS 发生拓扑来配置 VBUS 电压发生控制电路。

有关使用外部接口的信息，请参见《PIC24F 系列参考手册》的第 27 章“USB On-The-Go (OTG)”(DS39721A\_CN)。

### 18.1.4 计算收发器功耗要求

根据 USB 电缆的特性阻抗、电缆长度、V<sub>USB</sub> 供电电压以及 USB 电缆上的实际数据传输模式，USB 收发器消耗的电流量也有所不同。电缆越长，电容就越大，切换到输出状态时消耗的总能耗也就越多。对于不同的应用，收发器的总电流消耗也各不相同。公式 18-1 可帮助估算在全速应用中可能需要的实际电流量。

关于收发器功耗的全面讨论，请参见《PIC24F 系列参考手册》的第 27 章“USB On-The-Go (OTG)”(DS39721A\_CN)。

## 18.2 USB 缓冲区描述符和 BDT

可通过一个称为缓冲区描述符表 (Buffer Descriptor Table, BDT) 的结构来控制端点缓冲区。用户可使用这个方法灵活地构造和控制各种长度以及各种配置的端点缓冲区。

BDT 可位于数据 RAM 中任何可用的 512 字节对齐的块中。BDT 指针 (U1BDTP1) 包含 BDT 的高地址字节，用于设置 BDT 在 RAM 中的位置。用户必须设置该指针，以指示该表的位置。

BDT 由定义和控制 USB RAM 空间中的实际缓冲区的缓冲区描述符 (Buffer Descriptor, BD) 组成。每个 BD 都包含两个16位“软”(地址不固定)寄存器:BDnSTAT 和 BDnADR，其中 n 表示 64 个可能的 BD (范围为 0 至 63) 之一。BDnSTAT 是 BDn 的状态寄存器，而 BDnADR 指定与 BDn 相关的缓冲区的起始地址。

根据使用的端点缓冲配置，对于总共 256 字节的 BDT 空间，最多有 64 组缓冲区描述符。根据最低配置的要求，BDT 必须至少是 8 字节长。这是因为 USB 规范强制每个设备必须提供具有输入 / 输出的端点 0，用于初始化设置。

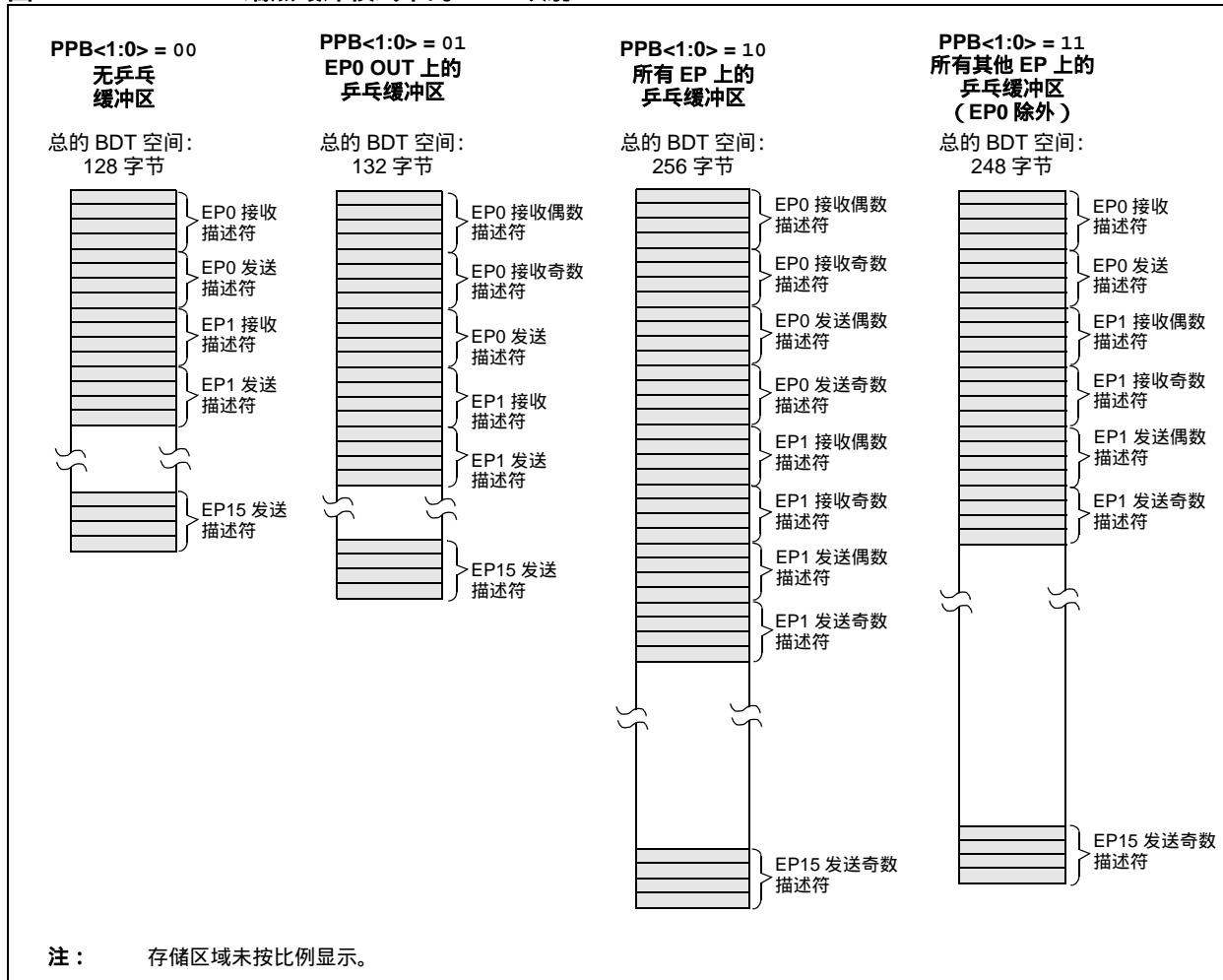
BDT 中的端点映射取决于三个变量：

- 端点号 (0 至 15)
- 端点方向 (Rx 或 Tx)
- 乒乓设置 (U1CNFG1<1:0>)

图 18-8 演示了如何使用这些变量映射 BDT 中的端点。

在主机模式下，仅使用了端点 0 的缓冲区描述符。所有传输操作都使用了端点 0 的缓冲区描述符和端点控制寄存器 (U1EP0)。接收数据包时，可通过 USB 状态寄存器 (U1STAT<7:4>) 中 ENDPT<3:0> 的值指示所连接设备的源端点。发送数据包时，通过写入令牌寄存器 (U1TOK) 的值指示所连接设备的目标端点。

**图 18-8： 端点缓冲模式下的 BDT 映射**



根据缓冲区配置，BD 与特定端点有固定关系。表 18-2 提供了 BD 到端点的映射。这种关系还意味着如果没有连续使能端点，BDT 中可能会存在间隙。从理论上而言，这表示用于已禁止端点的 BD 能够用作缓冲空间。但实际上，用户会避免在 BDT 中使用这类空间，除非执行验证 BD 地址操作。

## 18.2.1 缓冲区所有权

由于缓冲区及其 BD 是由 CPU 和 USB 模块共用的，所以可将 UOWN 位用作简单的信号机制，用以区分允许哪个模块更新 BD 及其相关的缓冲存储区。UOWN 是 BDnSTAT 的两个配置之间唯一共用的位。

清零 UOWN 时，BD 条目归单片机内核“所有”。UOWN 位置 1 时，BD 条目和缓冲存储器归 USB 外设“所有”。在此期间内，内核不应修改 BD 或其相应的数据缓冲区。注意，SIE 拥有缓冲区时，单片机内核仍可读取 BDnSTAT，反之亦然。

根据寄存器更新源不同，缓冲区描述符具有不同的含义。寄存器 18-1 和寄存器 18-2 显示了 BDnSTAT 当前“所有权”不同时的差异。

当 UOWN 置 1 时，用户不再信赖于写入 BD 的值。从此时起，USB 模块只在需要时更新 BD，改写原来的 BD 值。SIE 使用令牌 PID 更新 BDnSTAT 寄存器，同时也将更新传输计数。

## 18.2.2 DMA 接口

USB OTG 模块使用专用的 DMA 访问 BDT 和端点数据缓冲区。因为 DMA 的部分地址空间专用于缓冲区描述符，所以与 DMA 连接的存储器的一部分必须是正确映射的连续地址空间，以供模块访问。

**表 18-2：不同缓冲模式下缓冲区描述符的分配**

端点	分配给端点的 BD							
	模式 0 (无乒乓缓冲区)		模式 1 (EP0 OUT 上的乒乓缓冲区)		模式 2 (所有 EP 上的乒乓缓冲区)		模式 3 (所有其他 EP 上的乒乓缓冲区，EP0 除外)	
	输出	输入	输出	输入	输出	输入	输出	输入
0	0	1	0 (E), 1 (O)	2	0 (E), 1 (O)	2 (E), 3 (O)	0	1
1	2	3	3	4	4 (E), 5 (O)	6 (E), 7 (O)	2 (E), 3 (O)	4 (E), 5 (O)
2	4	5	5	6	8 (E), 9 (O)	10 (E), 11 (O)	6 (E), 7 (O)	8 (E), 9 (O)
3	6	7	7	8	12 (E), 13 (O)	14 (E), 15 (O)	10 (E), 11 (O)	12 (E), 13 (O)
4	8	9	9	10	16 (E), 17 (O)	18 (E), 19 (O)	14 (E), 15 (O)	16 (E), 17 (O)
5	10	11	11	12	20 (E), 21 (O)	22 (E), 23 (O)	18 (E), 19 (O)	20 (E), 21 (O)
6	12	13	13	14	24 (E), 25 (O)	26 (E), 27 (O)	22 (E), 23 (O)	24 (E), 25 (O)
7	14	15	15	16	28 (E), 29 (O)	30 (E), 31 (O)	26 (E), 27 (O)	28 (E), 29 (O)
8	16	17	17	18	32 (E), 33 (O)	34 (E), 35 (O)	30 (E), 31 (O)	32 (E), 33 (O)
9	18	19	19	20	36 (E), 37 (O)	38 (E), 39 (O)	34 (E), 35 (O)	36 (E), 37 (O)
10	20	21	21	22	40 (E), 41 (O)	42 (E), 43 (O)	38 (E), 39 (O)	40 (E), 41 (O)
11	22	23	23	24	44 (E), 45 (O)	46 (E), 47 (O)	42 (E), 43 (O)	44 (E), 45 (O)
12	24	25	25	26	48 (E), 49 (O)	50 (E), 51 (O)	46 (E), 47 (O)	48 (E), 49 (O)
13	26	27	27	28	52 (E), 53 (O)	54 (E), 55 (O)	50 (E), 51 (O)	52 (E), 53 (O)
14	28	29	29	30	56 (E), 57 (O)	58 (E), 59 (O)	54 (E), 55 (O)	56 (E), 57 (O)
15	30	31	31	32	60 (E), 61 (O)	62 (E), 63 (O)	58 (E), 59 (O)	60 (E), 61 (O)

图注：(E) = 偶数事务缓冲区，(O) = 奇数事务缓冲区

# PIC24FJ64GB004 系列

寄存器 18-1 : BDnSTAT : 缓冲区描述符 n 状态寄存器原型 , USB 模式 (BD0STAT 至 BD63STAT)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
UOWN	DTS	PID3	PID2	PID1	PID0	BC9	BC8
bit 15							bit 8

| R/W-x |
|-------|-------|-------|-------|-------|-------|-------|-------|
| BC7   | BC6   | BC5   | BC4   | BC3   | BC2   | BC1   | BC0   |
| bit 7 |       |       |       |       |       |       | bit 0 |

## 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15      **UOWN** : USB 所有权位  
1 = USB 模块拥有 BD 及其对应的缓冲区 ; CPU 不得修改 BD 或缓冲区
- bit 14      **DTS** : 数据翻转数据包位  
1 = 数据 1 数据包  
0 = 数据 0 数据包
- bit 13-10    **PID<3:0>** : 数据包标识符位 (由 USB 模块写入)  
在设备模式下 : 表示上一次传输期间接收到的令牌的 PID。  
在主机模式下 : 表示上一次返回的 PID 或传输状态指示。
- bit 9-0      **BC<9:0>** : 字节计数位  
这表示传输期间要发送的字节数或要接收的最大字节数。完成后 , 由 USB 模块使用实际发送或接收的字节数更新该字节计数。

## 寄存器 18-2 : BDnSTAT : 缓冲区描述符 n 状态寄存器原型 , CPU 模式 (BD0STAT 至 BD63STAT)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
UOWN	DTS <sup>(1)</sup>	0	0	DTSEN	BSTALL	BC9	BC8
bit 15	bit 8						

| R/W-x |
|-------|-------|-------|-------|-------|-------|-------|-------|
| BC7   | BC6   | BC5   | BC4   | BC3   | BC2   | BC1   | BC0   |
| bit 7 | bit 0 |       |       |       |       |       |       |

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15      **UOWN** : USB 所有权位  
0 = 单片机内核拥有 BD 及其对应的缓冲区。USB 模块忽略 BD 中的所有其他字段。
- bit 14      **DTS** : 数据翻转数据包位<sup>(1)</sup>  
1 = 数据 1 数据包  
0 = 数据 0 数据包
- bit 13-12    **保留功能** : 保持为 0
- bit 11      **DTSEN** : 数据翻转同步使能位  
1 = 使能数据翻转同步 ; 忽略具有错误同步值的数据包  
0 = 不执行数据翻转同步
- bit 10      **BSTALL** : 缓冲区停止使能位  
1 = 使能缓冲区停止 ; 如果接收到使用给定存储单元 BD 的令牌 , 发出停止 (STALL) 握手 (UOWN 位保持置 1 , BD 值不变 ) ; 任何停止握手都将导致相应的 EPSTALL 位置 1  
0 = 禁止缓冲区停止
- bit 9-0      **BC<9:0>** : 字节计数位  
这表示传输期间要发送的字节数或要接收的最大字节数。完成后 , 由 USB 模块使用实际发送或接收的字节数更新该字节计数。

注 1 : 除非 DTSEN = 1 , 否则该位将被忽略。

## 18.3 USB 中断

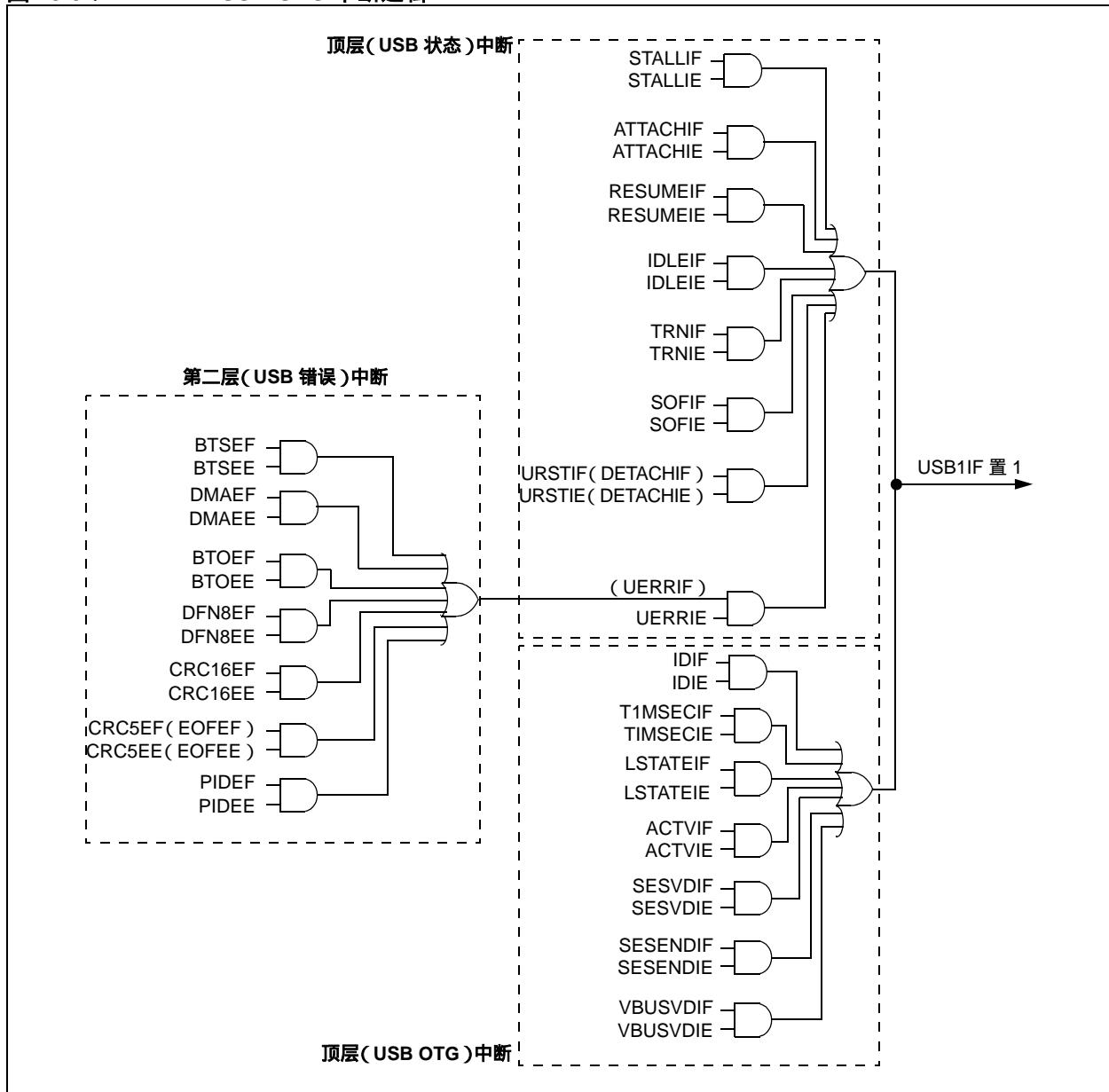
USB OTG 模块在许多条件下可以配置为产生中断。所有中断源都使用同一中断向量。

图 18-10 给出了 USB 模块的中断逻辑。在 USB 模块中有两层中断寄存器。顶层由所有 USB 状态中断组成；这些中断分别在 U1IE 和 U1IR 寄存器中被允许和标记。

第二层由 USB 错误条件组成，它们分别在 U1EIE 和 U1EIR 寄存器中被允许和标记。任意一个中断条件都可以触发顶层的 USB 错误中断标志（UERRIF）。

中断可用于在 USB 事务中产生陷阱。图 18-10 提供了一些 USB 帧中常见的事件及其相应的中断。

图 18-9：USB OTG 中断逻辑

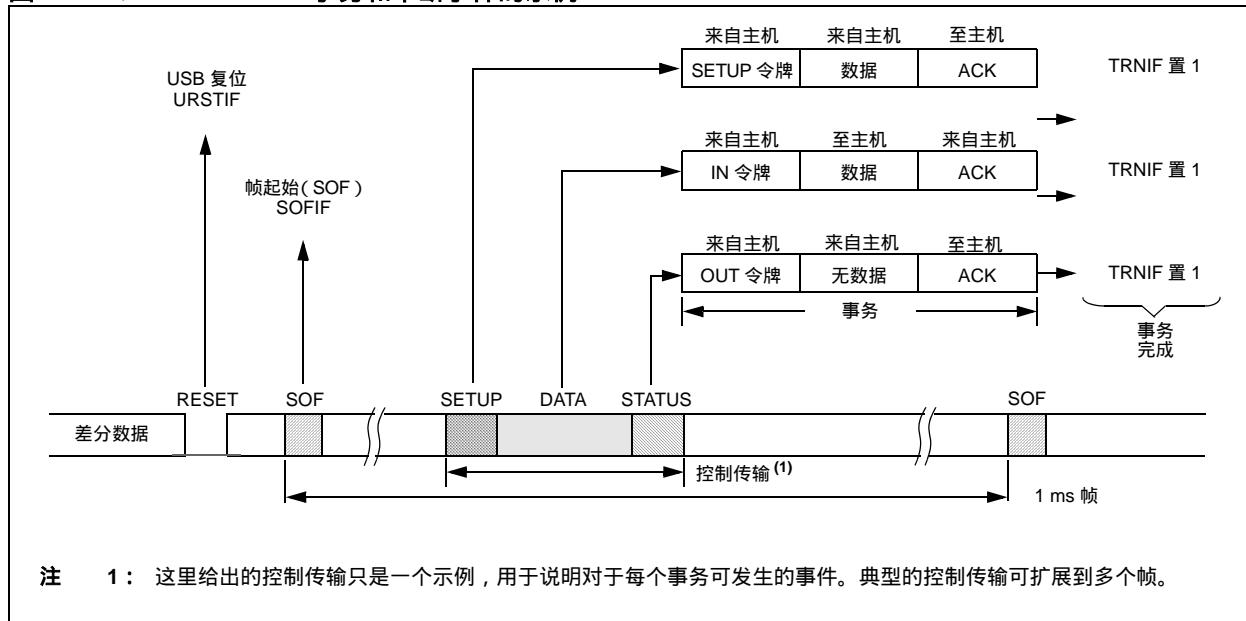


### 18.3.1 清除 USB OTG 中断

与器件级别的中断不同，USB OTG 中断状态标志不能用软件自由写入。所有 USB OTG 标志位都实现为只能由硬件置 1 位。而且，只能通过用软件写入 1 到其存储单元（即执行 MOV 类型的指令）来清零这些位。写入 0 到标志位（即 BCLR 指令）的操作无效。

**注：** 在本数据手册中，只能通过写入 1 到其存储单元来进行清零的位被称为“写 1 清除”位。在寄存器描述中，此功能用描述符“K”表示。

**图 18-10：USB 事务和中断事件的示例**



## 18.4 设备模式的操作

接下来的章节描述如何执行常见的设备模式任务。在设备模式下，在传输层执行 USB 传输。USB 模块自动执行传输的状态阶段。

### 18.4.1 使能设备模式

1. 通过先置 1 后清零乒乓缓冲区复位位 PPBRST (U1CON<1>) 来复位乒乓缓冲区指针。
2. 禁止所有中断 (U1IE 和 U1EIE = 00h)。
3. 通过将 FFh 写入 U1IR 和 U1EIR 来清零任何现有的中断标志。
4. 确认 VBUS 已提供 (仅限非 OTG 设备)。

5. 通过将 USBEN 位 (U1CON<0>) 置 1 使能 USB 模块。
6. 通过将 OTGEN 位 (U1OTGCON<2>) 置 1 使能 OTG 操作。
7. 通过将端点 0 的 EPRXEN 和 EPHSHK 位置 1 (U1EP0<3,0> = 1) 使能端点 0 缓冲区接收第一个设置数据包。
8. 通过将 USBPWR 位 (U1PWRC<0>) 置 1 使 USB 模块上电。
9. 通过将 DPPULUP (U1OTGCON<7>) 置 1 使能 D+ 上拉电阻发出连接信号。

## 18.4.2 在设备模式下接收 IN 令牌

1. 按 USB 2.0 规范第 9 章中的说明连接到 USB 主机并执行枚举。
2. 创建数据缓冲区，并用要发送到主机的数据进行填充。
3. 在所需端点的适当（偶数或奇数）发送 BD 中：
  - a) 使用正确的数据翻转（DATA0/1）值和数据缓冲区的字节计数设置状态寄存器（BDnSTAT）。
  - b) 使用数据缓冲区的起始地址设置地址寄存器（BDnADR）。
  - c) 将状态寄存器的 UOWN 位置 1。
4. 当 USB 模块接收到 IN 令牌时，会自动发送缓冲区中的数据。完成时，模块更新状态寄存器（BDnSTAT）并将传输完成中断标志 TRNIF（U1IR<3>）置 1。

## 18.4.3 在设备模式下接收 OUT 令牌

1. 按 USB 2.0 规范第 9 章中的说明连接到 USB 主机并执行枚举。
2. 使用期望从主机获得的数据量创建数据缓冲区。
3. 在所需端点的适当（偶数或奇数）发送 BD 中：
  - a) 使用正确的数据翻转（DATA0/1）值和数据缓冲区的字节计数设置状态寄存器（BDnSTAT）。
  - b) 使用数据缓冲区的起始地址设置地址寄存器（BDnADR）。
  - c) 将状态寄存器的 UOWN 位置 1。
4. 当 USB 模块接收到 OUT 令牌时，会自动接收主机发送给缓冲区的数据。完成时，模块更新状态寄存器（BDnSTAT）并将传输完成中断标志 TRNIF（U1IR<3>）置 1。

## 18.5 主机模式的操作

接下来的章节描述如何执行常见的主机模式任务。在主机模式下，USB 传输由主机软件明确调用。主机软件负责传输的应答部分。而且，所有传输都是使用端点 0 控制寄存器（U1EP0）和缓冲区描述符执行的。

### 18.5.1 使能主机模式并发现连接的设备

1. 通过将 HOSTEN 位（U1CON<3>）置 1 使能主机模式。这使得其他 USB OTG 寄存器中的主机模式控制位可用。
2. 通过将 DPPULDWN 和 DMPULDWN（U1OTGCON<5:4>）置 1 使能 D+ 和 D- 下拉电阻。通过清零 DPPULUP 和 DMPULUP（U1OTGCON<7:6>）禁止 D+ 和 D- 上拉电阻。
3. 此时，SOF 计数器装入 12,000 开始 SOF 生成。通过清零 SOFEN 位（U1CON<0>）禁止帧起始（Start-Of-Frame，SOF）数据包生成，从而消除 USB 上的噪声。
4. 通过将 ATTACHIE（U1IE<6>）置 1 允许所连接设备的中断。
5. 等待所连接设备的中断（U1IR<6> = 1）。这可通过发出以下信号来实现：USB 设备将 D+ 或 D- 的状态从 0 更改为 1（SE0 变为 J 状态）。在此之后，等待 100 ms 的时间，以使设备电源稳定下来。
6. 检查 U1CON 中 JSTATE 和 SE0 位的状态。如果 JSTATE 位（U1CON<7>）为 0，则连接的设备为低速。如果连接的设备为低速，则将低速 LSPDEN 和 LSPD 位（U1ADDR<7> 和 U1EP0<7>）置 1 使能低速操作。
7. 通过将 USBRST 位（U1CON<4>）置 1 至少 50 ms 在总线上发出复位信号，来复位 USB 设备。50 ms 后，通过清零 USBRST 终止复位。
8. 要避免所连接设备进入暂挂状态，可通过将 SOFEN 位置 1 使能 SOF 数据包生成。
9. 等待 10 ms，使设备从复位中恢复。
10. 按 USB 2.0 规范第 9 章中的说明执行枚举。

## 18.5.2 完成对所连接设备的控制事务

1. 按照第 18.5.1 节 “使能主机模式并发现连接的设备” 中描述的步骤发现设备。
2. 通过将 0Dh 写入 U1EP0 (这会将 EPCONDIS、EPTXEN 和 EPHSHK 位置 1) 将端点控制寄存器设置为双向控制传输。
3. 在存储缓冲区中存储设备框架设置命令的副本。关于设备框架命令集的信息 , 请参见 USB 2.0 规范第 9 章。
4. 初始化当前 (偶数或奇数) 发送 EP0 的缓冲区描述符 (BD) , 以传输设备框架命令的 8 字节命令数据 (即 GET DEVICE DESCRIPTOR) :
  - a) 将 BD 数据缓冲区地址 (BD0ADR) 设置为包含此命令的 8 字节存储缓冲区的起始地址。
  - b) 将 8008h 写入 BD0STAT (这会将 UOWN 位置 1 和将字节计数设置为 8 )。
5. 在地址寄存器 (U1ADDR<6:0>) 中设置目标设备的 USB 设备地址。在 USB 总线复位后 , 设备 USB 地址将为零。在枚举操作后 , 它将设置为一个 1 至 127 之间的值。
6. 将 D0h 写入 U1TOK ; 这是一个到目标设备的默认控制管道端点 0 的 SETUP 令牌。这将在总线上启动 SETUP 令牌 , 后跟数据包。在数据包传输完成后 , 将返回设备握手到 BD0STAT 的 PID 字段中。当 USB 模块更新了 BD0STAT 时 , 将产生一个传输完成中断 (TRNIF 标志置 1)。该操作将完成设置事务的设置阶段 , 如 USB 规范第 9 章中所述。
7. 要启动设置事务的数据阶段 (即获取 GET DEVICE 描述符命令所需数据) , 请在存储器中设置缓冲区以存储接收到的数据。
8. 初始化当前 (偶数或奇数) 接收或发送 (对于 IN 是接收 , 对于 OUT 是发送) EP0 BD 以传输数据。
  - a) 将 C040h 写入 BD0STAT。这会将 UOWN 置 1、将数据翻转同步 (Data Toggle Synchronization , DTS) 配置为 DATA1 , 并将字节计数设置为数据缓冲区的长度 (这里是 64 即 40h )。
  - b) 将 BD0ADR 设置为数据缓冲区的起始地址。
9. 用对于目标设备的默认控制管道端点 0 适当的 IN 或 OUT 令牌写令牌寄存器 (例如 , 对于 GET DEVICE DESCRIPTOR 命令的 IN 令牌 , 写入 90h 到 U1TOK)。这会在总线上启动 IN 令牌 , 后跟从设备发送到主机的数据包。当数据包完成时 , 将写入 BD0STAT 并产生一个传输完成中断 (即 TRNIF 标志置 1)。对于单数据包数据阶段的控制传输 , 该操作将完成设置事务的数据阶段 , 如 USB 规范第 9 章中所述。如果需要传输更多数据 , 请返回步骤 8。
10. 要启动设置事务的状态阶段 , 请在存储器中建立一个缓冲区以接收或发送零长度状态阶段数据包。
11. 初始化当前 (偶数或奇数) 发送 EP0 BD 以传输状态数据 :
  - a) 将 BDT 缓冲区地址字段设置为数据缓冲区的起始地址
  - b) 将 8000h 写入 BD0STAT (置 1 UOWN 位、将 DTS 配置为 DATA0 , 并将字节计数设置为 0)。
12. 用对于目标设备的默认控制管道端点 0 适当的 IN 或 OUT 令牌写令牌寄存器 (例如 , 对于 GET DEVICE DESCRIPTOR 命令的 OUT 令牌 , 写入 01h 到 U1TOK)。这会在总线上启动 OUT 令牌 , 后跟从主机发送到设备的零长度数据包。当数据包完成时 , 将用来自设备的握手更新 BD , 并产生一个传输完成中断 (即 TRNIF 标志置 1)。该操作将完成设置事务的状态阶段 , 如 USB 规范第 9 章中所述。

**注 :** 对于每个帧 , 只能执行一个控制事务。

## 18.5.3 向目标设备发送全速批量数据传输

1. 按照第 18.5.1 节“使能主机模式并发现连接的设备”和第 18.5.2 节“完成对所连接设备的控制事务”中描述的步骤发现和配置设备。
2. 要在使能握手时使能发送和接收传输，请将 1Dh 写入 U1EP0。如果目标设备为低速设备，还需要将 LSPD 位 (U1EP0<7>) 置 1。如果希望硬件在目标设备在传输中产生 NAK 时无限期自动重试，请清零重试禁止位 RETRYDIS (U1EP0<6>)。
3. 设置当前（偶数或奇数）发送 EP0 的 BD，以传输最多 64 字节。
4. 在地址寄存器 (U1ADDR<6:0>) 中设置目标设备的 USB 设备地址。
5. 将到目标端点的 OUT 令牌写入 U1TOK。该操作会触发该模块的发送状态机，开始发送令牌和数据。
6. 等待传输完成中断标志 TRNIF。这表示 BD 所有权已释放给微处理器且传输完成。如果重试禁止位置 1，则返回握手 (ACK、NAK、STALL 或 ERROR (0Fh)) 到 BD PID 字段。如果发生暂停中断，则从队列中除去暂挂数据包，并清除目标设备中的错误条件。如果发生断开连接中断（保持 SE0 状态 2.5 μs 以上），则目标已断开连接 (U1IR<0> 置 1)。
7. 发生传输完成中断 (TRNIF 置 1) 后，可返回步骤 2 检查 BD 并将下一个数据包入队。

**注：** 只能在模块设置阶段配置 USB 速度、收发器和上拉。建议不要在模块使能后更改这些设置。

## 18.6 OTG 工作

### 18.6.1 会话请求协议 (SRP)

OTG A 设备可在未使用 USB 链路时，通过会话请求协议 (Session Request Protocol, SRP) 关闭 VBUS 电源。软件可通过清零 VBUSON (U1OTGCON<3>) 完成此操作。当 VBUS 电源掉电后，说明 A 器件已结束 USB 会话。

OTG A 设备或嵌入式主机可以随时重新让 VBUS 电源上电（发起新的会话）。OTG B 设备也可以请求 OTG A 设备重新让 VBUS 电源上电（发起新的会话）。这可以通过会话请求协议 (SRP) 来完成。

在请求新的会话之前，B 设备必须检查上一个会话是否确实已结束。要完成此操作，B 设备必须检查两个条件：

1. VBUS 电源低于会话有效电压，且
2. D+ 和 D- 都至少保持低电平状态 2 ms。

SESENDIF (U1OTGIR<2>) 中断会向 B 设备通知条件 1。软件将必须手动检查条件 2。

**注：** 当 A 设备使 VBUS 电源掉电时，B 设备必须断开其上拉电阻与电源的连接。如果设备是自供电的，它会通过清零 DPPULUP (U1OTGCON<7>) 和 DMPULUP (U1OTGCON<6>) 来执行此操作。

B 设备可以通过一个电阻让 VBUS 电源放电来协助达到条件 1。用软件通过将 VBUSDIS (U1OTGCON<0>) 置 1 可完成此操作。

满足这些初始条件后，B 设备可开始请求新的会话。B 设备通过在 D+ 数据线上发送脉冲信号开始请求。用软件通过将 DPPULUP (U1OTGCON<7>) 置 1 来完成此操作。数据线应保持高电平 5 至 10 ms。

B 设备随后在 VBUS 电源上发送脉冲信号。用软件通过将 PUVBUS ( U1CNFG2<4> ) 置 1 来完成此操作。当 A 设备检测到 SRP 信号 ( 通过 ATTACHIF ( U1IR<6> ) 中断或通过 SESVDIF ( U1OTGIR<3> ) 中断 ) 时 , A 设备必须通过将 VBUSON ( U1OTGCON<3> ) 置 1 或将控制外部电源的 I/O 端口设为高电平来恢复 VBUS 电源。

B 设备在发送 VBUS 电源脉冲信号时不会监视 VBUS 电源的状态。当 B 设备确实检测到 VBUS 电源已恢复 ( 通过 SESVDIF ( U1OTGIR<3> ) 中断 ) 时 , 它必须通过上拉 D+ 或 D- ( 通过 DPPULUP 或 DMPULUP 位 ) 重新连接到 USB 链路。

A 设备必须通过驱动 USB 复位信号来完成 SRP。

### 18.6.2 主机协商协议 ( HNP )

在 USB OTG 应用中 , 双角色设备 ( Dual Role Device , DRD ) 是能够用作主机或外设的设备。所有 OTG DRD 都必须支持主机协商协议 ( HNP )。

HNP 允许 OTG B 设备临时用作 USB 主机。A 设备必须先使能 B 设备 , 以遵守 HNP 。更多关于 HNP 的信息 , 请参见 “ USB 2.0 规范 On-The-Go 补充信息 ” 。 HNP 只能在全速时启动。

由 A 设备使能以支持 HNP 后 , B 设备只需指示断开连接即可请求在 USB 链路处于暂挂状态的任何时刻作为主机。用软件通过清零 DPPULUP 和 DMPULUP 可完

成此操作。当 A 设备检测到断开连接条件 ( 通过 URSTIF ( U1IR<0> ) 中断 ) 时 , A 设备可允许 B 设备作为主机接管。A 设备通过发出全速连接信号完成该操作。用软件将 DPPULUP 置 1 实现该操作。

如果 A 设备改为使用恢复信号作为响应 , 则 A 设备仍用作主机。当 B 设备检测到连接条件 ( 通过 ATTACHIF ( U1IR<6> ) ) 时 , 它成为主机。B 设备在使用总线之前驱动复位信号。

当 B 设备完成主机角色后 , 它停止所有总线活动并通过将 DPPULUP 置 1 来使其 D+ 上拉电阻。当 A 设备检测到暂挂条件 ( 保持空闲 3 ms ) 时 , A 设备关闭其 D+ 上拉。A 设备还可让 VBUS 电源掉电以结束会话。当 A 设备检测到连接条件 ( 通过 ATTACHIF ) 时 , 它恢复主机操作并驱动复位信号。

### 18.6.3 外部 VBUS 比较器

通过将 UVCMPDIS 位 ( U1CNFG2<1> ) 置 1 使能外部 VBUS 比较器。这会禁止内部 VBUS 比较器 , 无需将 VBUS 连接到单片机的 VBUS 引脚。

外部比较器接口根据 UVCMPSEL 位 ( U1CNFG2<5> ) 设置的不同 , 使用 VCMPST1 和 VCMPST2 引脚或 VBUSVLD 、 SESSVLD 和 SESSEND 引脚。这些引脚是数字输入引脚 , 应根据 VBUS 电压的当前电平 , 设置为以下几种模式 ( 见表 18-3 )。

**表 18-3 : 外部 VBUS 比较器状态**

如果 UVCMPSEL = 0			
VCMPST1	VCMPST2	总线条件	
0	0	VBUS < VB_SESS_END	
1	0	VB_SESS_END < VBUS < VA_SESS_VLD	
0	1	VA_SESS_VLD < VBUS < VA_VBUS_VLD	
1	1	VBUS > VVBUS_VLD	

如果 UVCMPSEL = 1			
VBUSVLD	SESSVLD	SESSEND	总线条件
0	0	1	VBUS < VB_SESS_END
0	0	0	VB_SESS_END < VBUS < VA_SESS_VLD
0	1	0	VA_SESS_VLD < VBUS < VA_VBUS_VLD
1	1	0	VBUS > VVBUS_VLD

## 18.7 USB OTG 模块寄存器

共有 37 个存储器映射的寄存器与 USB OTG 模块关联。  
可划分为四个通用类：

- USB OTG 模块控制 (12)
- USB 中断 (7)
- USB 端点管理 (16)
- USB VBUS 电源控制 (2)

该总数不包括 BDT 中的 (最多) 128 个 BD 寄存器。其原型寄存器 (如寄存器 18-1 和寄存器 18-2 所示) 分别在第 18.2 节 “USB 缓冲区描述符和 BDT” 中进行了说明。

除了 U1PWMCON 和 U1PWMRRS 之外，所有 USB OTG 寄存器都只实现了寄存器的低字节。高字节中的位未实现，不起作用。注意，有些寄存器仅在主机模式下有意义，而其他寄存器在设备和主机模式下具有不同的位含义和功能。

下面章节描述的寄存器包含那些具有特殊控制和配置功能的位。以下寄存器仅用于数据或地址值：

- U1BDTP1：在数据 RAM 中指定 256 字的页用于 BDT；8 位值 (bit 0 固定为 0) 用于边界对齐
- U1FRML 和 U1FRMH：包含用于当前数据帧的 11 位字节计数器
- U1PWMRRS：包含用于 VBUS 升压辅助 PWM 模块的表示 PWM 占空比的 8 位值 (bit<15:8>) 和 PWM 周期 (bit<7:0>)

## 18.7.1 USB OTG 模块控制寄存器

**寄存器 18-3 : U1OTGSTAT : USB OTG 状态寄存器 (仅主机模式)**

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

R-0, HSC	U-0	R-0, HSC	U-0	R-0, HSC	R-0, HSC	U-0	R-0, HSC
ID	—	LSTATE	—	SESVD	SESEND	—	VBUSVD
bit 7							bit 0

**图注 :** U = 未实现位 , 读为 0

R = 可读位

W = 可写位

HSC = 可由硬件置 1/ 清零的位

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-8 未实现 : 读为 0

bit 7 **ID** : ID 引脚状态指示位

1 = 未连接插头或 B 型电缆已插入 USB 插座

0 = A 型插头已插入 USB 插座

bit 6 未实现 : 读为 0

bit 5 **LSTATE** : 线路状态稳定指示位

1 = USB 线路状态 (由 SE0 和 JSTATE 定义) 已在上 1 ms 达到稳定

0 = USB 线路状态在上 1 ms 未达到稳定

bit 4 未实现 : 读为 0

bit 3 **SESVD** : 会话有效指示位

1 = VBUS 电压高于 A 或 B 设备上的 VA\_SESS\_VLD (如 USB OTG 规范所定义)

0 = VBUS 电压低于 A 或 B 设备上的 VA\_SESS\_VLD

bit 2 **SESEND** : B 会话结束指示位

1 = VBUS 电压低于 B 设备上的 VB\_SESS\_END (如 USB OTG 规范所定义)

0 = VBUS 电压高于 B 设备上的 VB\_SESS\_END

bit 1 未实现 : 读为 0

bit 0 **VBUSVD** : A-VBUS 有效指示位

1 = VBUS 电压高于 A 设备上的 VA\_VBUS\_VLD (如 USB OTG 规范所定义)

0 = VBUS 电压低于 A 设备上的 VA\_VBUS\_VLD

# PIC24FJ64GB004 系列

寄存器 18-4 : U1OTGCON : USB ON-THE-GO 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
DPPULUP	DMPULUP	DPPULDWN <sup>(1)</sup>	DMPULDWN <sup>(1)</sup>	VBUSON <sup>(1)</sup>	OTGEN <sup>(1)</sup>	VBUSCHG <sup>(1)</sup>	VBUUSDIS <sup>(1)</sup>
bit 7							bit 0

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-8	未实现 : 读为 0
bit 7	<b>DPPULUP</b> : D+ 上拉使能位 1 = 使能 D+ 数据线上拉电阻 0 = 禁止 D+ 数据线上拉电阻
bit 6	<b>DMPULUP</b> : D- 上拉使能位 1 = 使能 D- 数据线上拉电阻 0 = 禁止 D- 数据线上拉电阻
bit 5	<b>DPPULDWN</b> : D+ 下拉使能位 <sup>(1)</sup> 1 = 使能 D+ 数据线下拉电阻 0 = 禁止 D+ 数据线下拉电阻
bit 4	<b>DMPULDWN</b> : D- 下拉使能位 <sup>(1)</sup> 1 = 使能 D- 数据线下拉电阻 0 = 禁止 D- 数据线下拉电阻
bit 3	<b>VBUSON</b> : VBUS 上电位 <sup>(1)</sup> 1 = VBUS 线路上电 0 = VBUS 线路未上电
bit 2	<b>OTGEN</b> : OTG 功能使能位 <sup>(1)</sup> 1 = 使能 USB OTG 以及所有 D+/D- 上拉和下拉位 0 = 禁止 USB OTG ; 通过设置 HOSTEN 和 USBEN 位 ( U1CON<3,0> ) 用硬件控制 D+/D- 上拉和下拉
bit 1	<b>VBUSCHG</b> : VBUS 充电选择位 <sup>(1)</sup> 1 = VBUS 线路设置为充电到 3.3V 0 = VBUS 线路设置为充电到 5V
bit 0	<b>VBUUSDIS</b> : VBUS 放电使能位 <sup>(1)</sup> 1 = VBUS 线路通过电阻放电 0 = VBUS 线路不放电

注 1 : 这些位仅在主机模式下使用 ; 不可在设备模式下使用。

## 寄存器 18-5 : U1PWRC : USB 电源控制寄存器

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

R/W-0, HS	U-0	U-0	R/W-0	U-0	U-0	R/W-0, HC	R/W-0
UACTPND	—	—	USLPGRD	—	—	USUSPND	USBPWR
bit 7							bit 0

### 图注 :

R = 可读位

-n = 上电复位时的值

HS = 可由硬件置 1 的位

W = 可写位

1 = 置 1

HC = 可由硬件清零的位

U = 未实现位 , 读为 0

0 = 清零

x = 未知

bit 15-8 未实现 : 读为 0

bit 7 **UACTPND** : USB 活动暂挂位

1 = 此时不应暂挂模块 ( 需要 USLPGRD 位置 1 )

0 = 模块可以暂挂或掉电

bit 6-5 未实现 : 读为 0

bit 4 **USLPGRD** : 休眠 / 暂挂保护位

1 = 向 USB 模块指示它即将被暂挂或掉电

0 = 未暂挂

bit 3-2 未实现 : 读为 0

bit 1 **USUSPND** : USB 暂挂模式使能位

1 = USB OTG 模块处于暂挂模式 ; USB 时钟是门控的 , 且收发器处于低功耗状态

0 = 正常的 USB OTG 操作

bit 0 **USBPWR** : USB 操作使能位

1 = 使能 USB OTG 模块

0 = 禁止 USB OTG 模块<sup>(1)</sup>

注 1: 不要清零该位 , 除非 HOSTEN、 USBEN 和 OTGEN 位 ( U1CON<3,0> 和 U1OTGCON<2> ) 都被清零。

# PIC24FJ64GB004 系列

寄存器 18-6 : U1STAT : USB 状态寄存器

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

R-0, HSC	U-0	U-0					
ENDPT3	ENDPT2	ENDPT1	ENDPT0	DIR	PPBI <sup>(1)</sup>	—	—
bit 7							bit 0

图注 : U = 未实现位 , 读为 0

R = 可读位 W = 可写位

HSC = 可由硬件置 1/ 清零的位

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15-8 未实现 : 读为 0

bit 7-4 ENDPT<3:0> : 上一个活动端点编号位  
( 表示上一次 USB 传输更新的 BDT 编号 )

1111 = 端点 15

1110 = 端点 14

...

0001 = 端点 1

0000 = 端点 0

bit 3 DIR : 上一个 BD 方向指示位

1 = 上一个事务是发送传输 ( Tx )

0 = 上一个事务是接收传输 ( Rx )

bit 2 PPBI : 乒乓 BD 指针指示位<sup>(1)</sup>

1 = 上一个事务针对奇数 BD 存储区

0 = 上一个事务针对偶数 BD 存储区

bit 1-0 未实现 : 读为 0

注 1 : 此位仅对具有可用偶数和奇数 BD 寄存器的端点有效。

## 寄存器 18-7 : U1CON : USB 控制寄存器 (设备模式)

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

U-0	R-x, HSC	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	SE0	PKTDIS	—	HOSTEN	RESUME	PPBRST	USBEN
bit 7	bit 0						

图注 :

U = 未实现位 , 读为 0

R = 可读位

W = 可写位

HSC = 可由硬件置 1/ 清零的位

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-7 未实现 : 读为 0

bit 6 **SE0** : 有效单端零标志位

1 = USB 总线上检测到有效的单端零

0 = 未检测到单端零

bit 5 **PKTDIS** : 数据包传输禁止位

1 = 禁止处理 SIE 令牌和数据包 ; 接收到 SETUP 令牌时自动置 1

0 = 使能处理 SIE 令牌和数据包

bit 4 未实现 : 读为 0

bit 3 **HOSTEN** : 主机模式使能位

1 = 使能 USB 主机功能 ; 用硬件激活 D+ 和 D- 上的下拉

0 = 禁止 USB 主机功能

bit 2 **RESUME** : 恢复信号使能位

1 = 激活恢复信号

0 = 禁止恢复信号

bit 1 **PPBRST** : 乒乓缓冲区复位位

1 = 将所有乒乓缓冲区指针复位到偶数 BD 存储区

0 = 不复位乒乓缓冲区指针

bit 0 **USBEN** : USB 模块使能位

1 = 使能 USB 模块和支持电路 (设备已连接) ; 用硬件激活 D+ 上拉

0 = 禁止 USB 模块和支持电路 (设备已断开连接)

# PIC24FJ64GB004 系列

寄存器 18-8 : U1CON : USB 控制寄存器 (仅主机模式)

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

R-x, HSC	R-x, HSC	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
JSTATE	SE0	TOKBUSY	USBRST	HOSTEN	RESUME	PPBRST	SOFEN
bit 7							

图注 : U = 未实现位 , 读为 0

R = 可读位  
-n = 上电复位时的值

W = 可写位  
1 = 置 1

HSC = 可由硬件置 1/ 清零的位  
0 = 清零

x = 未知

bit 15-8 未实现 : 读为 0

bit 7 **JSTATE** : 有效差分接收器 J 状态标志位

1 = 在 USB 上检测到 J 状态 (低速模式下为差分 0 , 全速模式下为差分 1 )  
0 = 未检测到 J 状态

bit 6 **SE0** : 有效单端零标志位

1 = USB 总线上检测到有效的单端零  
0 = 未检测到单端零

bit 5 **TOKBUSY** : 令牌忙状态位

1 = USB 模块在 On-The-Go 状态下执行令牌  
0 = 不执行令牌

bit 4 **USBRST** : 模块复位位

1 = 产生了 USB 复位 ; 对于软件复位 , 应用程序必须置 1 该位 50 ms 然后清零  
0 = USB 复位已终止

bit 3 **HOSTEN** : 主机模式使能位

1 = 使能 USB 主机功能 ; 用硬件激活 D+ 和 D- 上的下拉  
0 = 禁止 USB 主机功能

bit 2 **RESUME** : 恢复信号使能位

1 = 激活恢复信号 ; 软件必须置 1 该位 10 ms 然后清零 , 以使能远程唤醒  
0 = 禁止恢复信号

bit 1 **PPBRST** : 乒乓缓冲区复位位

1 = 将所有乒乓缓冲区指针复位到偶数 BD 存储区  
0 = 不复位乒乓缓冲区指针

bit 0 **SOFEN** : 帧起始使能位

1 = 每隔 1 ms 发送一次帧起始令牌  
0 = 禁止帧起始令牌

## 寄存器 18-9 : U1ADDR : USB 地址寄存器

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
LSPDEN <sup>(1)</sup>	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-8 未实现 : 读为 0

bit 7 LSPDEN : 低速使能指示位<sup>(1)</sup>

1 = USB 模块工作在低速模式下

0 = USB 模块工作在全速模式下

bit 6-0 ADDR<6:0> : USB 设备地址位

注 1 : 仅主机模式。在设备模式下 , 该位未实现 , 读为 0。

## 寄存器 18-10 : U1TOK : USB 令牌寄存器 (仅主机模式)

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

R/W-0							
PID3	PID2	PID1	PID0	EP3	EP2	EP1	EP0
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-8 未实现 : 读为 0

bit 7-4 PID<3:0> : 令牌类型标识位

1101 = SETUP (Tx) 令牌类型事务<sup>(1)</sup>

1001 = IN (Rx) 令牌类型事务<sup>(1)</sup>

0001 = OUT (Tx) 令牌类型事务<sup>(1)</sup>

bit 3-0 EP<3:0> : 令牌命令端点地址位

该值必须指定所连接设备上的有效端点。

注 1 : 所有其他组合都被保留不用。

# PIC24FJ64GB004 系列

## 寄存器 18-11 : U1SOF : USB OTG 令牌起始门限值寄存器 (仅主机模式)

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

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| CNT7  | CNT6  | CNT5  | CNT4  | CNT3  | CNT2  | CNT1  | CNT0  |
| bit 7 |       |       |       |       |       |       | bit 0 |

### 图注 :

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

- bit 15-8      未实现 : 读为 0  
bit 7-0      CNT<7:0> : 帧起始大小位 : 值表示  $10 + (n \text{ 字节的数据包大小})$   
例如 :  
0100 1010 = 64 字节数据包  
0010 1010 = 32 字节数据包  
0001 0010 = 8 字节数据包

## 寄存器 18-12 : U1CNFG1 : USB 配置寄存器 1

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

R/W-0	R/W-0	U-0	R/W-0	U-0	U-0	R/W-0	R/W-0
UTEYE	UOEMON <sup>(1)</sup>	—	USBSIDL	—	—	PPB1	PPB0
bit 7							bit 0

### 图注 :

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

- bit 15-8      未实现 : 读为 0  
bit 7      UTEYE : USB 眼图测试使能位  
1 = 使能眼图测试  
0 = 禁止眼图测试  
bit 6      UOEMON : USB  $\overline{\text{OE}}$  监视器使能位<sup>(1)</sup>  
1 =  $\overline{\text{OE}}$  信号有效 ; 它表示驱动 D+/D- 线的时间间隔  
0 =  $\overline{\text{OE}}$  信号无效  
bit 5      未实现 : 读为 0  
bit 4      USBSIDL : 空闲模式下 USB OTG 停止位  
1 = 当器件进入空闲模式时 , 模块停止工作  
0 = 模块在空闲模式下继续工作  
bit 3-2      未实现 : 读为 0

注 1 : 此位仅在 UTRDIS 位 ( U1CNFG2<0> ) 置 1 时有效。

## 寄存器 18-12 : U1CNFG1 : USB 配置寄存器 1 (续)

bit 1-0 **PPB<1:0>** : 乒乓缓冲区配置位

- 11 = 为端点 1 至 15 使能偶数 / 奇数乒乓缓冲区
- 10 = 为所有端点使能偶数 / 奇数乒乓缓冲区
- 01 = 为 OUT 端点 0 使能偶数 / 奇数乒乓缓冲区
- 00 = 禁止偶数 / 奇数乒乓缓冲区

注 1：此位仅在 UTRDIS 位 (U1CNFG2<0>) 置 1 时有效。

## 寄存器 18-13 : U1CNFG2 : USB 配置寄存器 2

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

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	UVCMPSSEL	PUVBUS	EXTI2CEN	UVBUSDIS <sup>(1)</sup>	UVCMPPDIS <sup>(1)</sup>	UTRDIS <sup>(1)</sup>
bit 7							

### 图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-6 未实现：读为 0

bit 5 **UVCMPSSEL**：外部比较器输入模式选择位 (见表 18-3)

当 UVCMPPDIS 置 1 时：

- 1 = 使用 3 引脚输入连接外部比较器
- 0 = 使用 2 引脚输入连接外部比较器

bit 4 **PUVBUS**：VBUS 上拉使能位

- 1 = 使能 VBUS 引脚上的上拉
- 0 = 禁止 VBUS 引脚上的上拉

bit 3 **EXTI2CEN**：外部模块的 I<sup>2</sup>C™ 接口控制使能位

- 1 = 通过 I<sup>2</sup>C 接口控制外部模块
- 0 = 通过专用引脚控制外部模块

bit 2 **UVBUSDIS**：片内 5V 升压稳压器构建器禁止位<sup>(1)</sup>

- 1 = 禁止片内升压稳压器构建器；使能数字输出控制接口
- 0 = 激活片内升压稳压器构建器

bit 1 **UVCMPPDIS**：片内 VBUS 比较器禁止位<sup>(1)</sup>

- 1 = 禁止片内充电 VBUS 比较器；使能数字输入状态接口
- 0 = 激活片内充电 VBUS 比较器

bit 0 **UTRDIS**：片内收发器禁止位<sup>(1)</sup>

- 1 = 禁止片内收发器；使能数字收发器接口
- 0 = 激活片内收发器

注 1：USBPWR 位置 1 (U1PWRC<0> = 1) 时，绝不要更改这些位。

# PIC24FJ64GB004 系列

## 18.7.2 USB 中断寄存器

寄存器 18-14 : U1OTGIR : USB OTG 中断状态寄存器 (仅主机模式)

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

R/K-0, HS	U-0	R/K-0, HS					
IDIF	T1MSECIF	LSTATEIF	ACTVIF	SESVdif	SESENDIF	—	VBUSVDIF
bit 7							bit 0

图注 :	U = 未实现位 , 读为 0
R = 可读位	K = 写 “ 1 ” 以清零位
-n = 上电复位时的值	HS = 可由硬件置 1 的位
	1 = 置 1
	0 = 清零
	X = 未知

bit 15-8	未实现 : 读为 0
bit 7	<b>IDIF</b> : ID 状态更改指示位 1 = 检测到 ID 状态发生了更改 0 = ID 状态未发生更改
bit 6	<b>T1MSECIF</b> : 1 ms 定时器位 1 = 1 ms 定时器已超时 0 = 1 ms 定时器未超时
bit 5	<b>LSTATEIF</b> : 线路状态稳定指示位 1 = USB 线路状态 (由 SE0 和 JSTATE 位定义) 已稳定 1 ms , 但与上次不同 0 = USB 线路状态稳定未达到 1 ms
bit 4	<b>ACTVIF</b> : 总线活动指示位 1 = 检测到 D+/D- 线路或 VBUS 上有活动 0 = 未检测到 D+/D- 线路或 VBUS 上有活动
bit 3	<b>SESVdif</b> : 会话有效更改指示位 1 = VBUS 电压已超过 VA_SESS_END (如 USB OTG 规范所定义) <sup>(1)</sup> 0 = VBUS 电压未超过 VA_SESS_END
bit 2	<b>SESENDIF</b> : B 设备 VBUS 变化指示位 1 = 检测到 B 设备上的 VBUS 发生变化 ; VBUS 电压已超过 VB_SESS_END( 如 USB OTG 规范所定义) <sup>(1)</sup> 0 = VBUS 电压未超过 VA_SESS_END
bit 1	未实现 : 读为 0
bit 0	<b>VBUSVDIF</b> : A 设备 VBUS 变化指示位 1 = 检测到 A 设备上的 VBUS 发生变化 ; VBUS 电压已超过 VA_VBUS_VLD( 如 USB OTG 规范所定义) <sup>(1)</sup> 0 = 未检测到 A 设备上的 VBUS 发生变化

注 1 : 穿越 VBUS 门限值的事件可以发生在电压上升过程也可以发生在电压下降过程。

注 :	各个位只能通过向该位所在的单元写入 1 (作为对整个寄存器的字写操作的一部分) 来清零。使用布尔指令或位操作指令写单个位所在的单元会导致写过程中所有置 1 的位都被清零。
-----	---

## 寄存器 18-15 : U1OTGIE : USB OTG 中断允许寄存器 (仅主机模式)

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	U-0	R/W-0
IDIE	T1MSECIE	LSTATEIE	ACTVIE	SESVDIE	SESENDIE	—	VBUSVDIE
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-8 未实现：读为 0
- bit 7 **IDIE** : ID 中断允许位
  - 1 = 允许中断
  - 0 = 禁止中断
- bit 6 **T1MSECIE** : 1 ms 定时器中断允许位
  - 1 = 允许中断
  - 0 = 禁止中断
- bit 5 **LSTATEIE** : 线路状态稳定中断允许位
  - 1 = 允许中断
  - 0 = 禁止中断
- bit 4 **ACTVIE** : 总线活动中断允许位
  - 1 = 允许中断
  - 0 = 禁止中断
- bit 3 **SESVDIE** : 会话有效中断允许位
  - 1 = 允许中断
  - 0 = 禁止中断
- bit 2 **SESENDIE** : B 设备会话结束中断允许位
  - 1 = 允许中断
  - 0 = 禁止中断
- bit 1 未实现：读为 0
- bit 0 **VBUSSVDIE** : A 设备 VBUS 有效中断允许位
  - 1 = 允许中断
  - 0 = 禁止中断

# PIC24FJ64GB004 系列

## 寄存器 18-16 : U1IR : USB 中断状态寄存器 (仅设备模式)

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

R/K-0, HS	U-0	R/K-0, HS	R/K-0, HS	R/K-0, HS	R/K-0, HS	R-0	R/K-0, HS
STALLIF	—	RESUMEIF	IDLEIF	TRNIF	SOFIF	UERRIF	URSTIF
bit 7							bit 0

图注 :	U = 未实现位 , 读为 0	
R = 可读位	K = 写入 1 清零该位	HS = 可由硬件置 1 的位
-n = 上电复位时的值	1 = 置 1	0 = 清零

bit 15-8	未实现 : 读为 0
bit 7	<b>STALLIF</b> : STALL 握手中断位 1 = 在设备模式下 , 外设在事务的握手阶段发送 STALL 握手 0 = 未发送 STALL 握手
bit 6	未实现 : 读为 0
bit 5	<b>RESUMEIF</b> : 恢复中断位 1 = 观察到 D+ 或 D- 引脚上保持 K 状态 2.5 μs ( 低速模式下为差分 1 , 全速模式下为差分 0 ) 0 = 未观察到 K 状态
bit 4	<b>IDLEIF</b> : 空闲检测中断位 1 = 检测到空闲条件 ( 保持连续的空闲状态 3 ms 或更长时间 ) 0 = 未检测到空闲条件
bit 3	<b>TRNIF</b> : 令牌处理完成中断位 1 = 完成当前令牌的处理 ; 从 U1STAT 寄存器中读取端点信息 0 = 未完成当前令牌的处理 ; 清零 U1STAT 寄存器或从 STAT 装入下个令牌 ( 清零此位将导致 STAT FIFO 递增 )
bit 2	<b>SOFIF</b> : 帧起始令牌中断位 1 = 外设接收到帧起始令牌或主机到达帧起始门限值 0 = 未接收到帧起始令牌或未到达帧起始门限值
bit 1	<b>UERRIF</b> : USB 错误条件中断位 ( 只读 ) 1 = 发生未屏蔽的错误条件 ; 仅 U1EIE 寄存器中允许的错误状态才能将此位置 1 0 = 未发生未屏蔽的错误条件
bit 0	<b>URSTIF</b> : USB 复位中断位 1 = 有效的 USB 复位已发生至少 2.5 μs ; 在此位失效之前必须清零复位状态 0 = 未发生 USB 复位。各个位只能通过向该位所在的单元写入 1 ( 作为对整个寄存器的字写操作的一部分 ) 来清零。使用布尔指令或位操作指令写单个位所在的单元会导致写过程中所有置 1 的位都被清零。

注 :	各个位只能通过向该位所在的单元写入 1 ( 作为对整个寄存器的字写操作的一部分 ) 来清零。使用布尔指令或位操作指令写单个位所在的单元会导致写过程中所有置 1 的位都被清零。
-----	---

## 寄存器 18-17 : U1IR : USB 中断状态寄存器 (仅主机模式)

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

R/K-0, HS	R-0	R/K-0, HS					
STALLIF	ATTACHIF	RESUMEIF	IDLEIF	TRNIF	SOFIF	UERRIF	DETACHIF
bit 7							bit 0

**图注 :**

R = 可读位

-n = 上电复位时的值

U = 未实现位 , 读为 0

K = 写入 1 清零该位

1 = 置 1

HS = 可由硬件置 1 的位

0 = 清零

x = 未知

bit 15-8

**未实现** : 读为 0

bit 7

**STALLIF** : STALL 握手中断位

1 = 在设备模式下 , 外设在事务的握手阶段发送 STALL 握手

0 = 未发送 STALL 握手

bit 6

**ATTACHIF** : 外设连接中断位

1 = 模块已检测到外设连接 ; 如果总线状态不是 SE0 且在 2.5 μs 内没有总线活动则置 1 该位  
0 = 未检测到外设连接

bit 5

**RESUMEIF** : 恢复中断位

1 = 观察到 D+ 或 D- 引脚上保持 K 状态 2.5 μs ( 低速模式下为差分 1 , 全速模式下为差分 0 )  
0 = 未观察到 K 状态

bit 4

**IDLEIF** : 空闲检测中断位

1 = 检测到空闲条件 ( 保持连续的空闲状态 3 ms 或更长时间 )  
0 = 未检测到空闲条件

bit 3

**TRNIF** : 令牌处理完成中断位

1 = 完成当前令牌的处理 ; 从 U1STAT 寄存器中读取端点信息  
0 = 未完成当前令牌的处理 ; 清零 U1STAT 寄存器或从 U1STAT 装入下一个令牌

bit 2

**SOFIF** : 帧起始令牌中断位

1 = 外设接收到帧起始令牌或主机到达帧起始门限值  
0 = 未接收到帧起始令牌或未到达帧起始门限值

bit 1

**UERRIF** : USB 错误条件中断位

1 = 发生未屏蔽的错误条件 ; 仅 U1EIE 寄存器中允许的错误状态才能将此位置 1  
0 = 未发生未屏蔽的错误条件

bit 0

**DETACHIF** : 断开连接中断位

1 = 模块检测到外设断开连接 ; 在此位失效之前必须清零复位状态  
0 = 未检测到外设断开连接。各个位只能通过向该位所在的单元写入 1 ( 作为对整个寄存器的字写操作的一部分 ) 来清零。使用布尔指令或位操作指令写单个位所在的单元会导致写过程中所有置 1 的位都被清零。

**注 :**

各个位只能通过向该位所在的单元写入 1 ( 作为对整个寄存器的字写操作的一部分 ) 来清零。使用布尔指令或位操作指令写单个位所在的单元会导致写过程中所有置 1 的位都被清零。

# PIC24FJ64GB004 系列

寄存器 18-18 : U1IE : USB 中断允许寄存器 (所有 USB 模式)

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
STALLIE	ATTACHIE <sup>(1)</sup>	RESUMEIE	IDLEIE	TRNIE	SOFIE	UERRIE	URSTIE DETACHIE <sup>(1)</sup>
bit 7							bit 0

图注 :

R = 可读位                    W = 可写位                    U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-8      未实现 : 读为 0
- bit 7      **STALLIE** : STALL 握手中断允许位  
1 = 允许中断  
0 = 禁止中断
- bit 6      **ATTACHIE** : 外设连接中断位 (仅主机模式)<sup>(1)</sup>  
1 = 允许中断  
0 = 禁止中断
- bit 5      **RESUMEIE** : 恢复中断位  
1 = 允许中断  
0 = 禁止中断
- bit 4      **IDLEIE** : 空闲检测中断位  
1 = 允许中断  
0 = 禁止中断
- bit 3      **TRNIE** : 令牌处理完成中断位  
1 = 允许中断  
0 = 禁止中断
- bit 2      **SOFIE** : 帧起始令牌中断位  
1 = 允许中断  
0 = 禁止中断
- bit 1      **UERRIE** : USB 错误条件中断位  
1 = 允许中断  
0 = 禁止中断
- bit 0      **URSTIE 或 DETACHIE** : USB 复位中断 (设备模式) 或 USB 断开连接中断 (主机模式) 允许位  
1 = 允许中断  
0 = 禁止中断

注 1 : 在设备模式下未实现 , 读为 0。

## 寄存器 18-19 : U1EIR : USB 错误中断状态寄存器 (所有 USB 模式)

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

R/K-0, HS	U-0	R/K-0, HS	R/K-0, HS				
BTSEF	—	DMAEF	BTOEF	DFN8EF	CRC16EF	CRC5EF EOFEF	PIDEF
bit 7							bit 0

图注 :

U = 未实现位 , 读为 0

R = 可读位

K = 写入 1 清零该位

HS = 可由硬件置 1 的位

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-8 未实现 : 读为 0
- bit 7 **BTSEF** : 位填充错误标志位  
 1 = 检测到位填充错误  
 0 = 没有位填充错误
- bit 6 未实现 : 读为 0
- bit 5 **DMAEF** : DMA 错误标志位  
 1 = 检测到 USB DMA 错误条件 ; BD 字节计数字段指示的数据大小小于接收到的字节数。接收到的数据被截短。  
 0 = 没有 DMA 错误
- bit 4 **BTOEF** : 总线周转 ( Turnaround ) 超时错误标志位  
 1 = 发生总线周转超时  
 0 = 未发生总线周转超时
- bit 3 **DFN8EF** : 数据字段大小错误标志位  
 1 = 数据字段的字节数不是整数  
 0 = 数据字段的字节数是整数
- bit 2 **CRC16EF** : CRC16 失败标志位  
 1 = CRC16 失败  
 0 = CRC16 通过
- bit 1 在设备模式下 :  
**CRC5EF** : CRC5 主机错误标志位  
 1 = 令牌数据包由于 CRC5 错误而被拒绝  
 0 = 接受令牌数据包 ( 无 CRC5 错误 )
- 在主机模式下 :  
**EOFEF** : 帧结束 ( End-Of-Frame , EOF ) 错误标志位  
 1 = 发生了帧结束错误  
 0 = 禁止帧结束中断
- bit 0 **PIDEF** : PID 检查失败标志位  
 1 = PID 检查失败  
 0 = PID 检查通过。各个位只能通过向该位所在的单元写入 1 ( 作为对整个寄存器的字写操作的一部分 ) 来清零。使用布尔指令或位操作指令写单个位所在的单元会导致写过程中所有置 1 的位都被清零。

注 : 各个位只能通过向该位所在的单元写入 1 ( 作为对整个寄存器的字写操作的一部分 ) 来清零。使用布尔指令或位操作指令写单个位所在的单元会导致写过程中所有置 1 的位都被清零。

# PIC24FJ64GB004 系列

寄存器 18-20 : U1EIE : USB 错误中断允许寄存器

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

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BTSEE	—	DMAEE	BTOEE	DFN8EE	CRC16EE	CRC5EE EOFEE	PIDEE
bit 7							

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15-8 未实现：读为 0

bit 7 BTSEE : 位填充错误中断允许位

1 = 允许中断

0 = 禁止中断

bit 6 未实现：读为 0

bit 5 DMAEE : DMA 错误中断允许位

1 = 允许中断

0 = 禁止中断

bit 4 BTOEE : 总线周转超时错误中断允许位

1 = 允许中断

0 = 禁止中断

bit 3 DFN8EE : 数据字段大小错误中断允许位

1 = 允许中断

0 = 禁止中断

bit 2 CRC16EE : CRC16 失败中断允许位

1 = 允许中断

0 = 禁止中断

bit 1 在设备模式下：

CRC5EE : CRC5 主机错误中断允许位

1 = 允许中断

0 = 禁止中断

在主机模式下：

EOFEE : 帧结束错误中断允许位 (1)

1 = 允许中断

0 = 禁止中断

bit 0 PIDEE : PID 检查失败中断允许位

1 = 允许中断

0 = 禁止中断

## 18.7.3 USB 端点管理寄存器

**寄存器 18-21 : U1EPn : USB 端点控制寄存器 (n = 0 至 15)**

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	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LSPD <sup>(1)</sup>	RETRYDIS <sup>(1)</sup>	—	EPCONDIS	EPRXEN	EPTXEN	EPSTALL	EPHSHK
bit 7	bit 0						

**图注 :**

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-8      **未实现** : 读为 0
- bit 7      **LSPD** : 低速直接连接使能位 (仅限 U1EP0) <sup>(1)</sup>  
 1 = 使能直接连接到低速设备  
 0 = 禁止直接连接到低速设备
- bit 6      **RETRYDIS** : 重试禁止位 (仅限 U1EP0) <sup>(1)</sup>  
 1 = 禁止重试 NAK 事务  
 0 = 允许重试 NAK 事务 ; 用硬件完成重试
- bit 5      **未实现** : 读为 0
- bit 4      **EPCONDIS** : 双向端点控制位  
如果 EPTXEN 和 EPRXEN = 1 :  
 1 = 禁止端点 n 进行控制传输 ; 仅允许发送和接收传输  
 0 = 允许端点 n 进行控制 (SETUP) 传输 ; 也允许发送和接收传输  
对于 EPTXEN 和 EPRXEN 的所有其他组合 :  
 此位为无关位。
- bit 3      **EPRXEN** : 端点接收使能位  
 1 = 使能端点 n 接收  
 0 = 禁止端点 n 接收
- bit 2      **EPTXEN** : 端点发送使能位  
 1 = 使能端点 n 发送  
 0 = 禁止端点 n 发送
- bit 1      **EPSTALL** : 端点停止状态位  
 1 = 端点 n 已停止  
 0 = 端点 n 未停止
- bit 0      **EPHSHK** : 端点握手使能位  
 1 = 使能端点握手  
 0 = 禁止端点握手 (通常用于同步端点)

**注 1 :** 这些位仅在主机模式下对 U1EP0 可用。对于所有其他 U1EPn 寄存器 , 这些位始终未实现 , 读为 0。

# PIC24FJ64GB004 系列

## 18.7.4 USB VBUS 电源控制寄存器

寄存器 18-22 : U1PWMCON : USB VBUS PWM 发生器控制寄存器

R/W-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
PWMEN	—	—	—	—	—	PWMPOL	CNTEN
bit 15							bit 8

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

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 **PWMEN** : PWM 使能位

1 = 使能 PWM 发生器

0 = 禁止 PWM 发生器；输出保持 PWMPOL 指定的复位状态

bit 14-10 **未实现** : 读为 0

bit 9 **PWMPOL** : PWM 极性位

1 = PWM 输出低电平有效，复位状态为高电平

0 = PWM 输出高电平有效，复位状态为低电平

bit 8 **CNTEN** : PWM 计数器使能位

1 = 使能计数器

0 = 禁止计数器

bit 7-0 **未实现** : 读为 0

## 19.0 并行主端口 (PMP)

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 13 章“并行主端口 (PMP)”(DS39713A\_CN)。

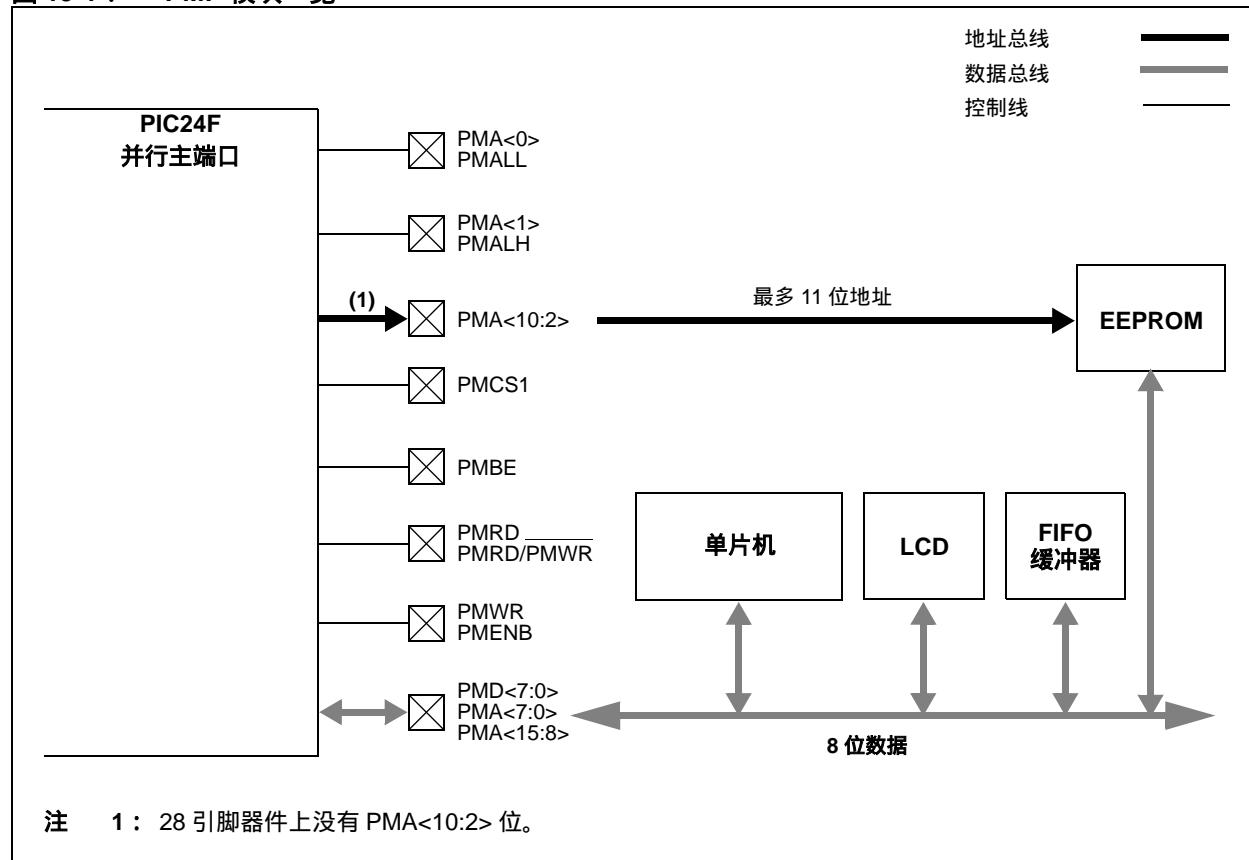
并行主端口 (Parallel Master Port, PMP) 模块是一个 8 位并行 I/O 模块，专用于与通信外设、LCD、外部存储器件以及单片机等多种并行器件进行通信。由于并行外设的接口差异很大，因此 PMP 具有很强的可配置能力。

**注：**某些 PMP 引脚在 PIC24FJ64GB0 系列器件上不存在。请参见特定器件的引脚排列以确定可用的引脚。

PMP 模块的主要特性包括：

- 最多 16 条可编程地址线
- 一条片选信号线
- 可编程选通选项：
  - 独立的读和写选通或；
  - 带使能端的读 / 写选通
- 地址自动增减 / 自动递减
- 可编程地址 / 数据复用
- 控制信号的可编程极性
- 支持传统的并行从端口
- 支持增强型并行从端口：
  - 地址支持
  - 4 字节深自动递增缓冲器
- 可编程等待状态
- 可选择的输入电平

图 19-1：PMP 模块一览



# PIC24FJ64GB004 系列

## 寄存器 19-1 : PMCON : 并行端口控制寄存器

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
MPEN	—	PSIDL	ADRMUX1 <sup>(1)</sup>	ADRMUX0 <sup>(1)</sup>	PTBEN	PTWREN	PTRDEN
bit 15	bit 8						

R/W-0	R/W-0	R/W-0 <sup>(2)</sup>	U-0	R/W-0 <sup>(2)</sup>	R/W-0	R/W-0	R/W-0
CSF1	CSF0	ALP	—	CS1P	BEP	WRSP	RDSP
bit 7	bit 0						

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15      **MPEN** : 并行主端口使能位  
1 = 使能 PMP  
0 = 禁止 PMP , 不执行任何片外访问
- bit 14      未实现 : 读为 0
- bit 13      **PSIDL** : 空闲模式停止位  
1 = 当器件进入空闲模式时 , 模块停止工作  
0 = 模块在空闲模式下继续工作
- bit 12-11    **ADRMUX<1:0>** : 地址 / 数据复用选择位<sup>(1)</sup>  
11 = 保留  
10 = 所有 16 位地址与 PMD<7:0> 引脚复用  
01 = 地址的低 8 位与 PMD<7:0> 引脚复用 , 高 3 位与 PMA<10:8> 引脚复用  
00 = 地址和数据使用独立的引脚
- bit 10      **PTBEN** : 字节使能端口使能位 (16 位主模式 )  
1 = 使能 PMBE 端口  
0 = 禁止 PMBE 端口
- bit 9      **PTWREN** : 写使能选通端口使能位  
1 = 使能 PMWR/PMENB 端口  
0 = 禁止 PMWR/PMENB 端口
- bit 8      **PTRDEN** : 读 / 写选通端口使能位  
1 = 使能 PMRD/PMWR 端口  
0 = 禁止 PMRD/PMWR 端口
- bit 7-6     **CSF<1:0>** : 片选功能位  
11 = 保留  
10 = PMCS1 用作片选功能  
01 = 保留  
00 = 保留
- bit 5      **ALP** : 地址锁存极性位<sup>(2)</sup>  
1 = 高电平有效 ( PMALL 和 PMALH )  
0 = 低电平有效 ( PMALL 和 PMALH )
- bit 4      未实现 : 读为 0
- bit 3      **CS1P** : 片选 1 极性位<sup>(2)</sup>  
1 = 高电平有效 ( PMCS1/PMCS1 )  
0 = 低电平有效 ( PMCS1/PMCS1 )

注 1 : 28 引脚器件上没有 PMA<10:2> 位。

2 : 这些位在相应引脚用作地址线时无效。

## 寄存器 19-1 : PMCON : 并行端口控制寄存器 (续)

bit 2	<b>BEP</b> : 字节使能极性位 1 = 字节使能高电平有效 ( PMBE ) 0 = 字节使能低电平有效 ( PMBE )
bit 1	<b>WRSP</b> : 写选通极性位 <u>对于从模式和主模式 2 ( PMMODE&lt;9:8&gt; = 00、01 或 10 ) :</u> 1 = 写选通高电平有效 ( PMWR ) 0 = 写选通低电平有效 ( PMWR ) <u>对于主模式 1 ( PMMODE&lt;9:8&gt; = 11 ) :</u> 1 = 使能选通高电平有效 ( PMENB ) 0 = 使能选通低电平有效 ( PMENB )
bit 0	<b>RDSP</b> : 读选通极性位 <u>对于从模式和主模式 2 ( PMMODE&lt;9:8&gt; = 00、01 或 10 ) :</u> 1 = 读选通高电平有效 ( PMRD ) 0 = 读选通低电平有效 ( PMRD ) <u>对于主模式 1 ( PMMODE&lt;9:8&gt; = 11 ) :</u> 1 = 读 / 写选通高电平有效 ( PMRD/PMWR ) 0 = 读 / 写选通低电平有效 ( PMRD/PMWR )

注 1：28 引脚器件上没有 PMA<10:2> 位。  
2：这些位在相应引脚用作地址线时无效。

# PIC24FJ64GB004 系列

## 寄存器 19-2 : PMMODE : 并行端口模式寄存器

R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUSY	IRQM1	IRQM0	INCM1	INCM0	MODE16	MODE1	MODE0
bit 15							

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WAITB1 <sup>(1)</sup>	WAITB0 <sup>(1)</sup>	WAITM3	WAITM2	WAITM1	WAITM0	WAITE1 <sup>(1)</sup>	WAITE0 <sup>(1)</sup>
bit 7							

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15	<b>BUSY</b> : 忙位 (仅用于主模式) 1 = 端口忙 (当处理器停止工作时无用) 0 = 端口不忙
bit 14-13	<b>IRQM&lt;1:0&gt;</b> : 中断请求模式位 11 = 当读取读缓冲器 3 或写入写缓冲器 3 时产生中断 (缓冲的 PSP 模式) 或在 PMA<1:0> = 11 (仅可寻址 PSP 模式) 时执行读或写操作时产生中断 10 = 不产生中断 , 处理器停止工作 01 = 在读 / 写周期结束时产生中断 00 = 不产生中断
bit 12-11	<b>INCM&lt;1:0&gt;</b> : 递增模式位 11 = PSP 读和写缓冲器自动递增 (仅传统 PSP 模式) 10 = 每个读 / 写周期 ADDR<10:0> 减 1 01 = 每个读 / 写周期 ADDR<10:0> 加 1 00 = 地址不发生增减
bit 10	<b>MODE16</b> : 8/16 位模式位 1 = 16 位模式 : 数据寄存器为 16 位 , 对数据寄存器执行读或写操作将调用两次 8 位传输 0 = 8 位模式 : 数据寄存器为 8 位 , 对数据寄存器执行读或写操作将调用一次 8 位传输
bit 9-8	<b>MODE&lt;1:0&gt;</b> : 并行端口模式选择位 11 = 主模式 1 ( PMCS1、PMRD/PMWR、PMENB、PMBE、PMA<x:0> 和 PMD<7:0> ) 10 = 主模式 2 ( PMCS1、PMRD、PMWR、PMBE、PMA<x:0> 和 PMD<7:0> ) 01 = 增强型 PSP 控制信号 ( PMRD、PMWR、PMCS1、PMD<7:0> 和 PMA<1:0> ) 00 = 传统并行从端口控制信号 ( PMRD、PMWR、PMCS1 和 PMD<7:0> )
bit 7-6	<b>WAITB&lt;1:0&gt;</b> : 从数据建立到执行读 / 写的等待状态配置位 <sup>(1)</sup> 11 = 数据等待时间为 4 个 TCY ; 地址复用时间为 4 个 TCY 10 = 数据等待时间为 3 个 TCY ; 地址复用时间为 3 个 TCY 01 = 数据等待时间为 2 个 TCY ; 地址复用时间为 2 个 TCY 00 = 数据等待时间为 1 个 TCY ; 地址复用时间为 1 个 TCY
bit 5-2	<b>WAITM&lt;3:0&gt;</b> : 从执行读操作到字节使能选通的等待状态配置位 1111 = 额外等待 15 个 TCY ... 0001 = 额外等待 1 个 TCY 0000 = 无额外等待周期 ( 强制操作在 1 个 TCY 内完成 )
bit 1-0	<b>WAITE&lt;1:0&gt;</b> : 选通后数据保持的等待状态配置位 <sup>(1)</sup> 11 = 等待 4 个 TCY 10 = 等待 3 个 TCY 01 = 等待 2 个 TCY 00 = 等待 1 个 TCY

注 1 : 只要 WAITM<3:0> = 0000 , WAITB 和 WAITE 位就被忽略。

## 寄存器 19-3 : PMADDR : 并行端口地址寄存器

U-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	CS1	—	—	—	ADDR10 <sup>(1)</sup>	ADDR9 <sup>(1)</sup>	ADDR8 <sup>(1)</sup>
bit 15							bit 8

R/W-0							
ADDR7 <sup>(1)</sup>	ADDR6 <sup>(1)</sup>	ADDR5 <sup>(1)</sup>	ADDR4 <sup>(1)</sup>	ADDR3 <sup>(1)</sup>	ADDR2 <sup>(1)</sup>	ADDR1 <sup>(1)</sup>	ADDR0 <sup>(1)</sup>
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现 : 读为 0

bit 14 CS1 : 片选 1 位

1 = 片选 1 有效

0 = 片选 1 无效

bit 13-11 未实现 : 读为 0

bit 10-0 ADDR<10:0> : 并行端口目标地址位<sup>(1)</sup>

注 1 : 28 引脚器件上没有 PMA<10:2> 位。

## 寄存器 19-4 : PMAEN : 并行端口使能寄存器

U-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	PTEN14	—	—	—	PTEN10 <sup>(1)</sup>	PTEN9 <sup>(1)</sup>	PTEN8 <sup>(1)</sup>
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTEN7 <sup>(1)</sup>	PTEN6 <sup>(1)</sup>	PTEN5 <sup>(1)</sup>	PTEN4 <sup>(1)</sup>	PTEN3 <sup>(1)</sup>	PTEN2 <sup>(1)</sup>	PTEN1	PTEN0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 未实现 : 读为 0

bit 14 PTEN14 : PMCS1 选通使能位

1 = PMCS1 用作片选功能

0 = PMCS1 引脚用作端口 I/O

bit 13-11 未实现 : 读为 0

bit 10-2 PTEN<10:2> : PMP 地址端口使能位<sup>(1)</sup>

1 = PMA<10:2> 用作 PMP 地址线

0 = PMA<10:2> 用作端口 I/O

bit 1-0 PTEN<1:0> : PMALH/PMALL 选通使能位

1 = PMA1 和 PMA0 用作 PMA<1:0> 或分别用作 PMALH 和 PMALL

0 = PMA1 和 PMA0 引脚用作端口 I/O

注 1 : 28 引脚器件上没有 PMA<10:2> 位。

# PIC24FJ64GB004 系列

## 寄存器 19-5 : PMSTAT : 并行端口状态寄存器

R-0	R/W-0, HS	U-0	U-0	R-0	R-0	R-0	R-0
IBF	IBOV	—	—	IB3F	IB2F	IB1F	IB0F
bit 15	bit 8						

R-1	R/W-0, HS	U-0	U-0	R-1	R-1	R-1	R-1
OBE	OBUF	—	—	OB3E	OB2E	OB1E	OB0E
bit 7	bit 0						

图注 : HS = 可由硬件置 1 的位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

X = 未知

bit 15 **IBF** : 输入缓冲器满状态位

1 = 所有可写的输入缓冲寄存器已满

0 = 部分或所有可写的输入缓冲寄存器为空

bit 14 **IBOV** : 输入缓冲器溢出状态位

1 = 尝试对已满的输入字节寄存器执行写操作 ( 必须由软件清零 )

0 = 未发生溢出

bit 13-12 未实现 : 读为 0

bit 11-8 **IB3F:IB0F** : 输入缓冲器 x 满状态位

1 = 输入缓冲器包含尚未读取的数据 ( 读缓冲器将清零此位 )

0 = 输入缓冲器不包含任何未读取的数据

bit 7 **OBE** : 输出缓冲器空状态位

1 = 所有可读的输出缓冲寄存器均为空

0 = 部分或所有可读的输出缓冲寄存器已满

bit 6 **OBUF** : 输出缓冲器下溢状态位

1 = 对空的输出字节寄存器执行了读操作 ( 必须由软件清零 )

0 = 未发生下溢

bit 5-4 未实现 : 读为 0

bit 3-0 **OB3E:OB0E** : 输出缓冲器 x 空状态位

1 = 输出缓冲器为空 ( 将数据写入该缓冲器会将该位清零 )

0 = 输出缓冲器中包含尚未发送的数据

## 寄存器 19-6 : PADCFG1 : 填充配置控制寄存器

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

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	RTSECSEL1 <sup>(1)</sup>	RTSECSEL0 <sup>(1)</sup>	PMPTTL
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-3 未实现 : 读为 0

bit 2-1 RTSECSEL<1:0> : RTCC 秒时钟输出选择位<sup>(1)</sup>

11 = 保留 ; 不要使用

10 = 选择从 RTCC 引脚输出 RTCC 时钟源 (时钟可以是 LPRC 或 SOSC , 具体取决于闪存配置位 RTCOSC (CW4<5>) 的设置 )

01 = 选择从 RTCC 引脚输出 RTCC 秒时钟

00 = 选择从 RTCC 引脚输出 RTCC 闹钟脉冲

bit 0 PMPTTL : PMP 模块 TTL 输入缓冲器选择位

1 = PMP 模块使用 TTL 输入缓冲器

0 = PMP 模块使用施密特触发器输入缓冲器

注 1 : 要使能实际 RTCC 输出 , 需将 RTCOE (RCFGCAL<10>) 位置 1。

# PIC24FJ64GB004 系列

图 19-2：传统并行从端口示例

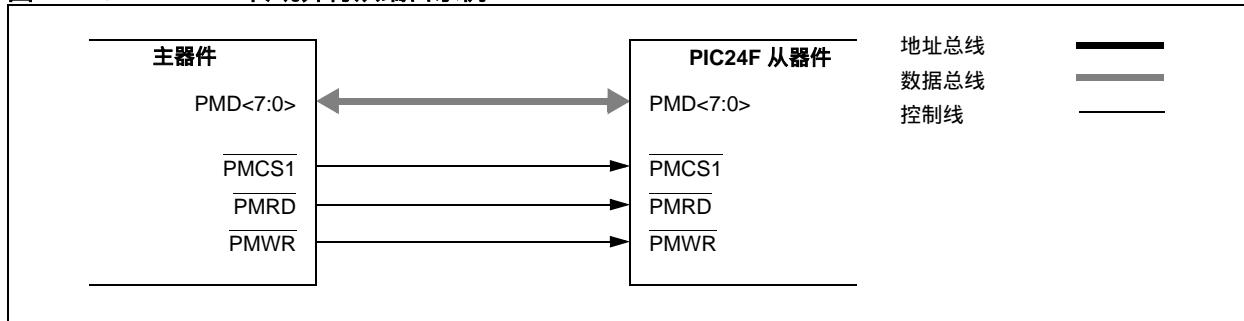


图 19-3：可寻址的并行从端口示例

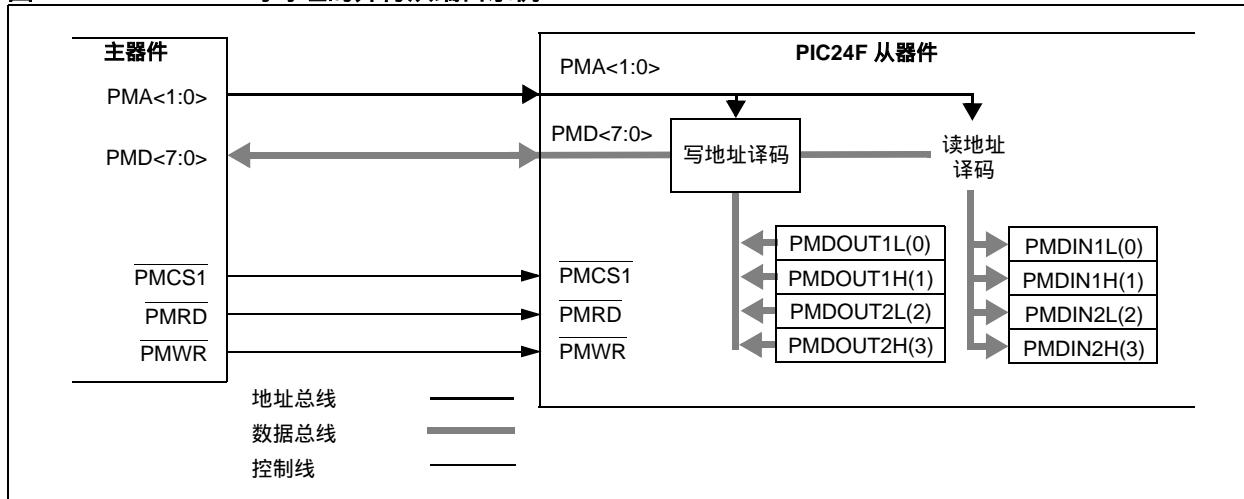


表 19-1：从模式地址解析

PMA<1:0>	输出寄存器（缓冲器）	输入寄存器（缓冲器）
00	PMDOUT1<7:0> (0)	PMDIN1<7:0> (0)
01	PMDOUT1<15:8> (1)	PMDIN1<15:8> (1)
10	PMDOUT2<7:0> (2)	PMDIN2<7:0> (2)
11	PMDOUT2<15:8> (3)	PMDIN2<15:8> (3)

图 19-4：主模式下的不复用寻址（独立的读和写选通、一个片选信号）

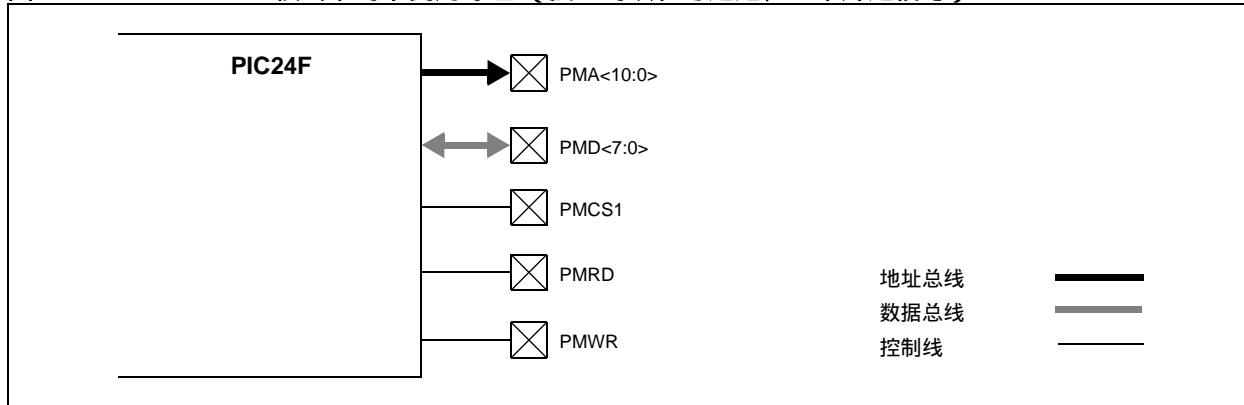


图 19-5： 主模式下的部分复用寻址（独立的读和写选通、一个片选信号）

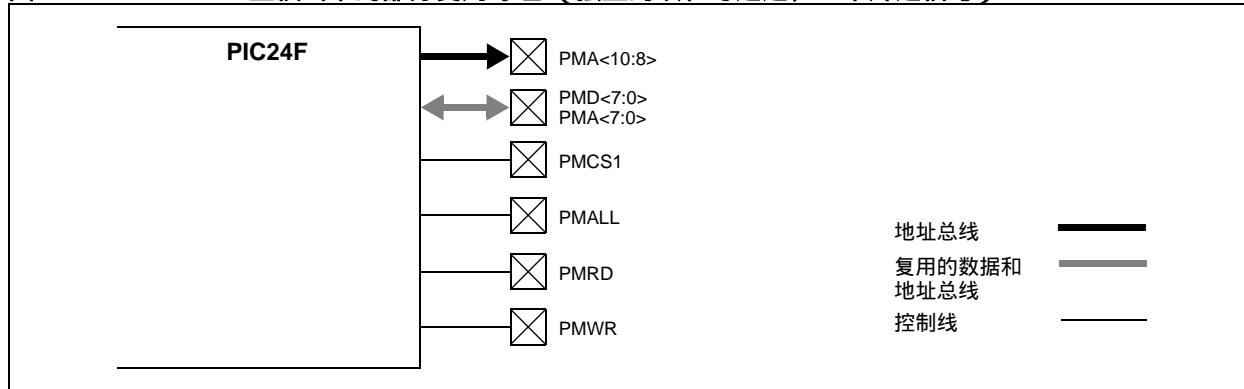


图 19-6： 主模式下的完全复用寻址（独立的读和写选通、一个片选信号）

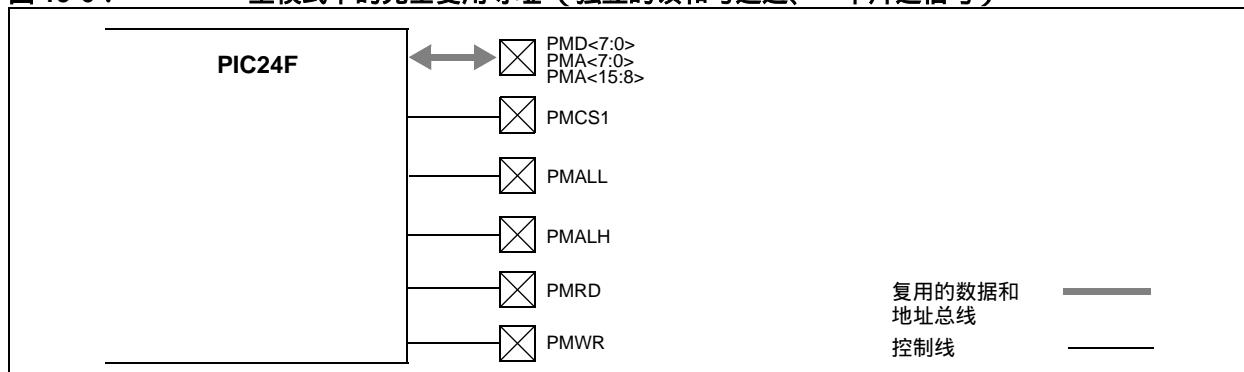


图 19-7： 复用寻址应用示例

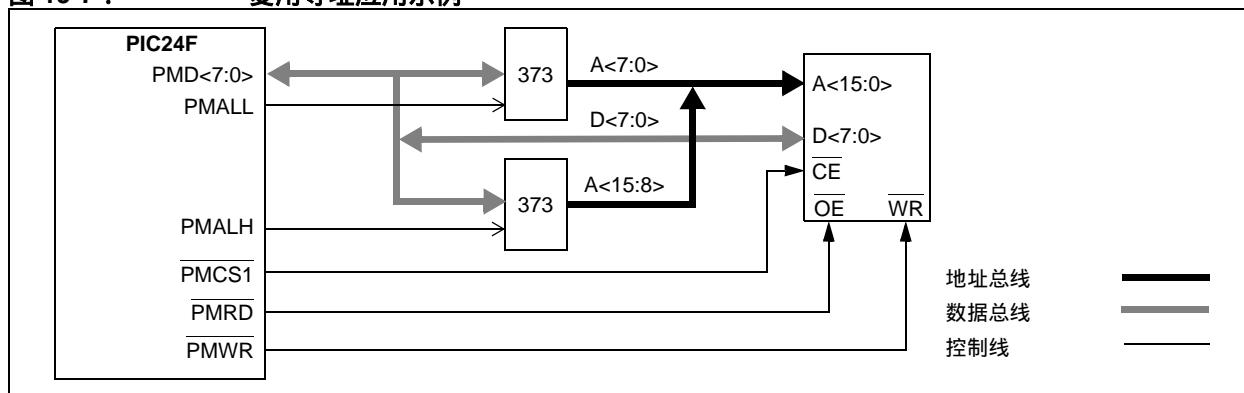
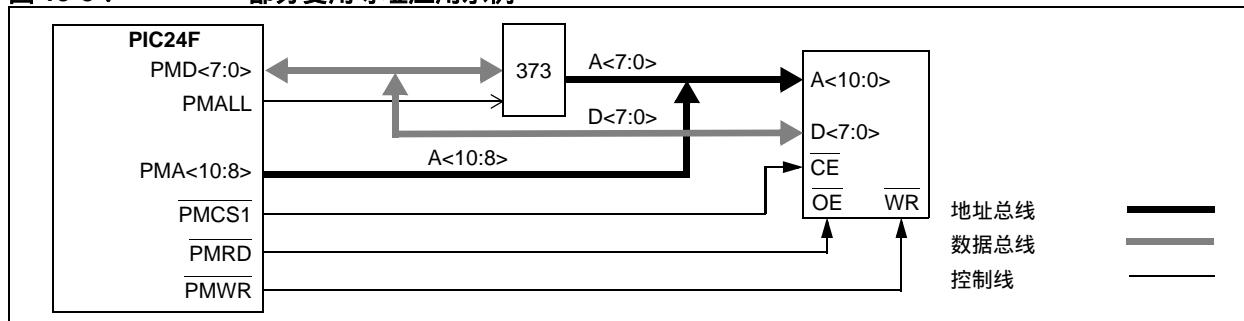


图 19-8： 部分复用寻址应用示例



# PIC24FJ64GB004 系列

图 19-9： 8 位复用地址和数据应用示例

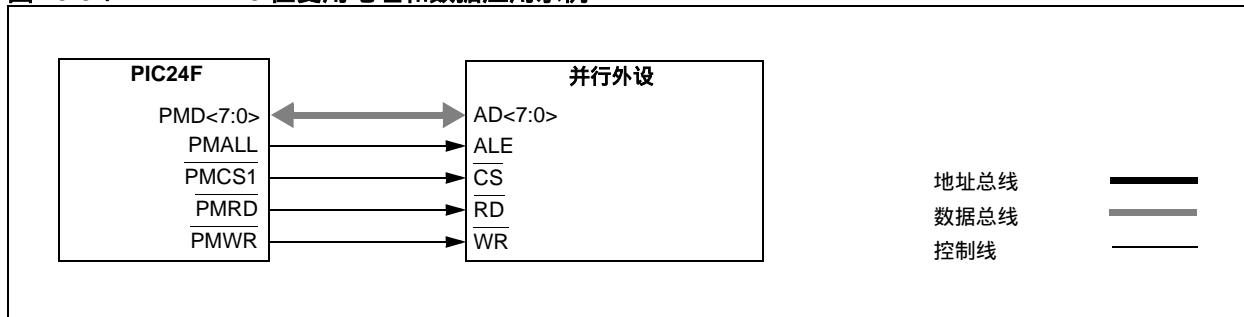


图 19-10： 并行 EEPROM 示例（最多 11 位地址、 8 位数据）

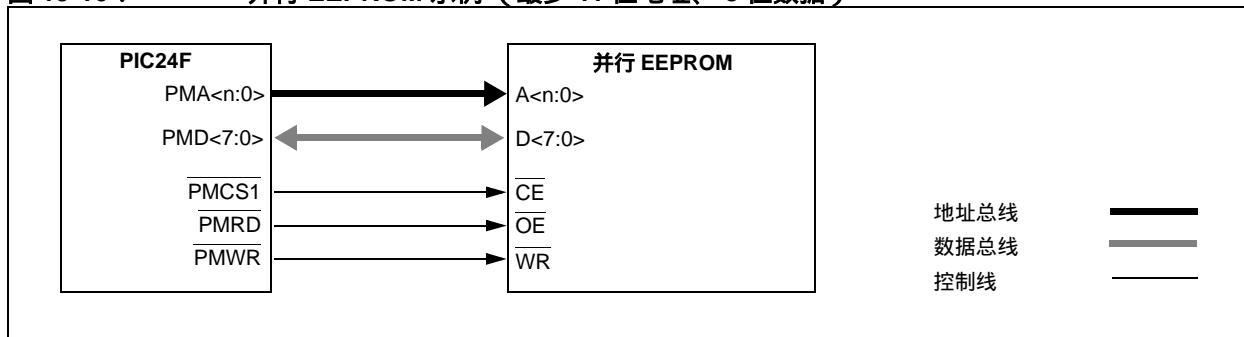


图 19-11： 并行 EEPROM 示例（最多 11 位地址、 16 位数据）

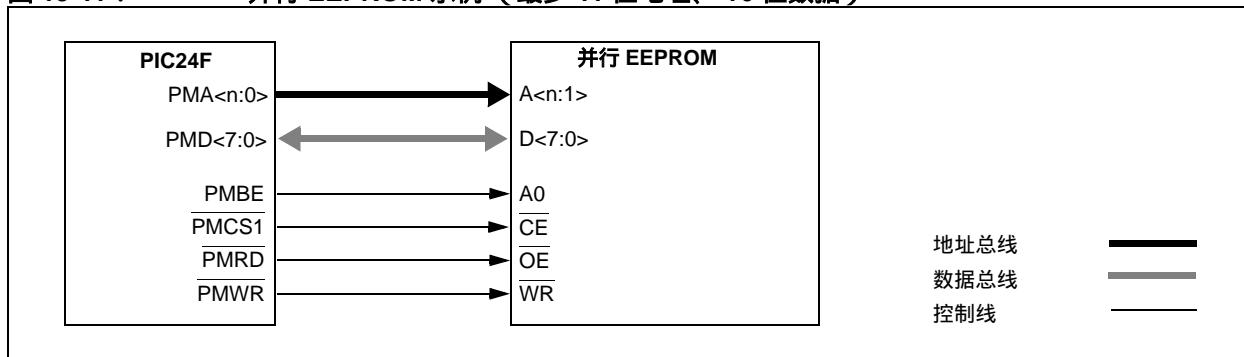
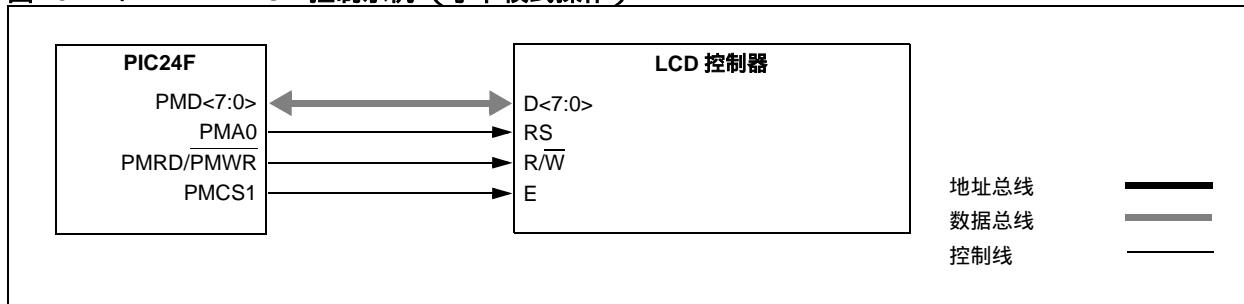


图 19-12： LCD 控制示例（字节模式操作）



## 20.0 实时时钟和日历 (RTCC)

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 29 章“实时时钟和日历 (RTCC)”(DS39696A\_CN)。

RTCC 为用户提供了可校准的实时时钟和日历 (Real-Time Clock and Calendar, RTCC) 功能。

RTCC 模块的主要特性包括：

- 可在深度休眠模式下工作
- 可选的时钟源
- 采用 24 小时格式提供时、分和秒
- 可分辨半秒的时长
- 提供日历——星期、日期、月和年

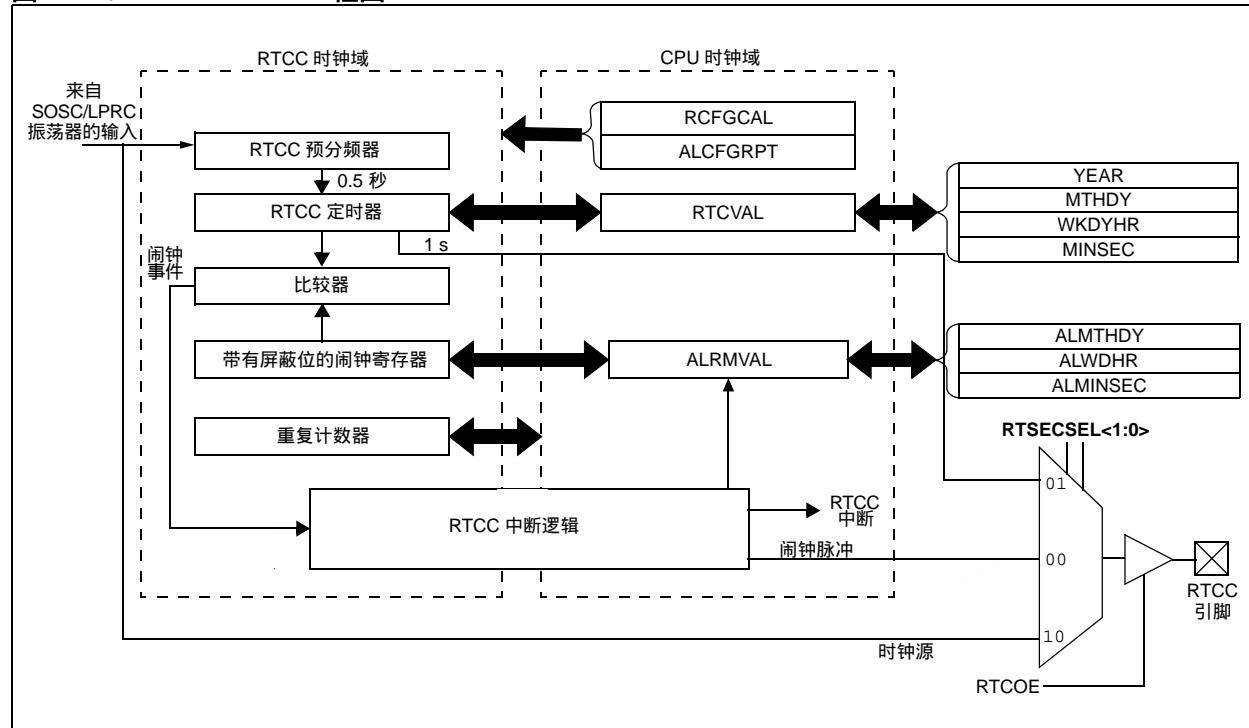
- 闹钟可配置半秒、1 秒、10 秒、1 分钟、10 分钟、1 小时、1 天、1 周、1 个月或 1 年
- 闹钟使用递减计数器进行重复
- 闹钟具有无限重复响铃
- 年份为 2000 至 2099，带闰年修正
- 采用 BCD 格式，以减小软件开销
- 为长期电池工作进行了优化
- 32.768 kHz 时钟晶振的用户校准 / 带周期性自动调整的 32K INTRC 频率

### 20.1 RTCC 源时钟

用户可选择 SOSC 晶振或 LPRC 低功耗内部 RC 振荡器作为 RTCC 模块的参考时钟。可使用 RTCOSC (CW4<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。

当器件通过 MCLR 复位时，SOSC 和 RTCC 都将保持运行，并且器件在释放 MCLR 后还将继续运行。

图 20-1： RTCC 框图



## 20.2 RTCC 模块寄存器

RTCC 模块寄存器分为三类：

- RTCC 控制寄存器
- RTCC 值寄存器
- 闹钟值寄存器

### 20.2.1 寄存器映射

要限制寄存器接口，则需通过相应的寄存器指针访问 RTCC 定时器和闹钟时间寄存器。RTCC 值寄存器窗口 (RTCVALH 和 RTCVALL) 使用 RTCPTR 位 (RCFGCAL<9:8>) 来选择需要的定时器寄存器对 (见表 20-1)。

通过写 RTCVALH 字节，使 RTCC 指针值 (RTCPTR<1:0> 位) 逐次递减 1 直至 00。一旦其达到 00，在手动更改指针值前，可通过 RTCVALH 和 RTCVALL 来访问分钟和秒值。

**表 20-1： RTCVAL 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 ALRMMVAL) 使用 ALRMPTR 位 (ALCFGRPT<9:8>) 来选择所需的闹钟寄存器对 (见表 20-2)。

通过写 ALRMVALH 字节，使闹钟指针值 (ALRMPTR<1:0> 位) 逐次递减直至 00。一旦其达到 00，在手动更改指针值前，可通过 ALRMMIN 和 ALRMSEC 来访问 ALRMMIN 和 ALRMSEC 的值。

### 例 20-1： 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 20-2： ALRMVAL 寄存器映射**

ALRMPTR <1:0>	闹钟值寄存器窗口	
	ALRMVAL<15:8>	ALRMVAL<7:0>
00	ALRMMIN	ALRMSEC
01	ALRMWD	ALRMHR
10	ALRMMNTH	ALRMDAY
11	—	—

考虑到 16 位内核不能区分 8 位和 16 位读操作，用户必须注意读 ALRMVALH 或 ALRMMVAL 字节会递减 ALRMPTR<1:0> 的值。同样，读 RTCVALH 或 RTCVALL 字节会使 RTCPTR<1:0> 的值递减。

**注：** 这仅适用于读操作，而不适用于写操作。

### 20.2.2 写锁定

为了对任何 RTCC 定时器寄存器执行写操作，必须将 RTCWREN 位 (RCFGCAL<13>) 置 1 (见例 20-1)。

**注：** 为避免意外写入定时器，建议在除要特意执行写操作之外的任何时候都保持 RTCWREN 位 (RCFGCAL<13>) 清零。对于要使 RTCWREN 位置 1 的操作，由于在 55h/Ah 序列和 RTCWREN 置 1 之间仅允许有一个指令周期的时间；因此，建议按照例 20-1 中的过程执行代码。

### 20.2.3 选择 RTCC 时钟源

使用闪存配置位 RTCOSC (CW4<5>) 位选择 RTCC 模块的时钟源。该位设置为 1 时，辅助振荡器 (SOSC) 用作参考时钟；该位设置为 0 时，LPRC 用作参考时钟。

## 20.2.4 RTCC 控制寄存器

**寄存器 20-1 : RCFGCAL : RTCC 校准和配置寄存器<sup>(1)</sup>**

R/W-0	U-0	R/W-0	R-0, HSC	R-0, HSC	R/W-0	R/W-0	R/W-0
RTCEN <sup>(2)</sup>	—	RTCWREN	RTCSYNC	HALFSEC <sup>(3)</sup>	RTCOE	RTCPTR1	RTCPTR0
bit 15							bit 8

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| CAL7  | CAL6  | CAL5  | CAL4  | CAL3  | CAL2  | CAL1  | CAL0  |
| bit 7 |       |       |       |       |       |       | bit 0 |

**图注 :**

HSC = 可由硬件置 1/ 清零的位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15	<b>RTCEN</b> : RTCC 使能位 <sup>(2)</sup>
	1 = 使能 RTCC 模块
	0 = 禁止 RTCC 模块
bit 14	<b>未实现</b> : 读为 0
bit 13	<b>RTCWREN</b> : RTCC 值寄存器写使能位
	1 = RTCVALH 和 RTCVALL 寄存器可被用户写入
	0 = RTCVALH 和 RTCVALL 寄存器被锁定不允许用户写入
bit 12	<b>RTCSYNC</b> : RTCC 值寄存器读同步位
	1 = 由于计满返回 RTCVALH、RTCVALL 和 ALCFGRPT 寄存器可能会在读取期间变化 , 从而导致读到的数据无效。如果两次读取寄存器得到的数据相同 , 则认为数据有效。
	0 = RTCVALH、RTCVALL 或 ALCFGRPT 寄存器可被读取 , 而不用担心计满返回
bit 11	<b>HALFSEC</b> : 半秒状态位 <sup>(3)</sup>
	1 = 一秒的后半周期
	0 = 一秒的前半周期
bit 10	<b>RTCOE</b> : RTCC 输出使能位
	1 = 使能 RTCC 输出
	0 = 禁止 RTCC 输出
bit 9-8	<b>RTCPTR&lt;1:0&gt;</b> : RTCC 值寄存器窗口指针位
	当读 RTCVALH 和 RTCVALL 寄存器时 , 指向相应的 RTCC 值寄存器 ; 每次读或写 RTCVALH 时 , RTCPTR<1:0> 的值都会减 1 , 直至 00 。
	<b>RTCVAL&lt;15:8&gt;</b> :
	00 = 分钟数
	01 = 星期
	10 = 月
	11 = 保留
	<b>RTCVAL&lt;7:0&gt;</b> :
	00 = 秒数
	01 = 小时数
	10 = 日
	11 = 年

**注 1** : RCFGCAL 寄存器仅受到 POR 影响。

**2** : 仅当 RTCWREN = 1 时才允许写入 RTCEN 位。

**3** : 该位是只读位。当写入 MINSEC 寄存器的低半部分时 , 该位清 0 。

# PIC24FJ64GB004 系列

## 寄存器 20-1 : RCFGCAL : RTCC 校准和配置寄存器<sup>(1)</sup> (续)

bit 7-0	<b>CAL&lt;7:0&gt;</b> : 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
	.
	.
	00000001 = 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 : RCFGCAL 寄存器仅受到 POR 影响。  
2 : 仅当 RTCWREN = 1 时才允许写入 RTCEN 位。  
3 : 该位是只读位。当写入 MINSEC 寄存器的低半部分时 , 该位清 0。

## 寄存器 20-2 : 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	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	RTSECSEL1 <sup>(1)</sup>	RTSECSEL0 <sup>(1)</sup>	PMPTTL
bit 7							bit 0

### 图注 :

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

bit 15-3	未实现 : 读为 0
bit 2-1	<b>RTSECSEL&lt;1:0&gt;</b> : RTCC 秒时钟输出选择位 <sup>(1)</sup>
	11 = 保留 ; 不要使用
	10 = 选择从 RTCC 引脚输出 RTCC 时钟源 ( 时钟可以是 LPRC 或 SOSC , 具体取决于 RTCOSC 位 ( CW4<5> ) 的设置 )
	01 = 选择从 RTCC 引脚输出 RTCC 秒时钟
	00 = 选择从 RTCC 引脚输出 RTCC 闹钟脉冲
bit 0	<b>PMPTTL</b> : PMP 模块 TTL 输入缓冲器选择位
	1 = PMP 模块使用 TTL 输入缓冲器
	0 = PMP 模块使用施密特触发器输入缓冲器

- 注 1 : 要使能实际 RTCC 输出 , 需将 RTCOE ( RCFGCAL<10> ) 位置 1。

## 寄存器 20-3 : 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 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| ARPT7 | ARPT6 | ARPT5 | ARPT4 | ARPT3 | ARPT2 | ARPT1 | ARPT0 |
| bit 7 | bit 0 |       |       |       |       |       |       |

### 图注 :

R = 可读位

-n = 上电复位时的值

W = 可写位

1 = 置 1

U = 未实现位 , 读为 0

0 = 清零

x = 未知

bit 15      **ALRMEN** : 闹钟使能位  
 1 = 使能闹钟 (只要 ARPT<7:0> = 00h 且 CHIME = 0 , 就会在闹钟事件后自动清零 )  
 0 = 禁止闹钟

bit 14      **CHIME** : 响铃使能位  
 1 = 使能响铃 ; 允许 ARPT<7:0> 位从 00h 返回到 FFh  
 0 = 禁止响铃 ; ARPT<7:0> 位到达 00h 时停止

bit 13-10    **AMASK<3:0>** : 闹钟屏蔽配置位  
 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 (除了配置在 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  
 当读 ALRMVALH 和 ALRMVALL 寄存器时指向相应的闹钟值寄存器 ; 每次读或写 ALRMVALH 时 ,  
 ALRMPTR<1:0> 的值都会减 1 , 直至 00 。

#### ALRMVAL<15:8> :

00 = ALRMMIN  
 01 = ALRMWD  
 10 = ALRMMNTH  
 11 = 未实现

#### ALRMVAL<7:0> :

00 = ALRMSEC  
 01 = ALRMHR  
 10 = ALRMDAY  
 11 = 未实现

bit 7-0      **ARPT<7:0>** : 闹钟重复计数器值位  
 11111111 = 闹钟将重复 255 次

00000000 = 闹钟不重复  
 每发生一次闹钟事件 , 计数器就减 1 ; 除非 CHIME = 1 , 否则计数器不能从 00h 返回到 FFh 。

# PIC24FJ64GB004 系列

## 20.2.5 RTCVAL 寄存器映射

### 寄存器 20-4 : YEAR : 年值寄存器<sup>(1)</sup>

| U-0, HSC   |
|------------|------------|------------|------------|------------|------------|------------|------------|------------|
| —          | —          | —          | —          | —          | —          | —          | —          | —          |
| bit 15     |            |            |            |            |            |            |            | bit 8      |
| R/W-x, HSC |
| YRTEN3     | YRTEN2     | YRTEN1     | YRTEN0     | YRONE3     | YRONE2     | YRONE1     | YRONE0     |            |
| bit 7      |            |            |            |            |            |            |            | bit 0      |

图注：

HSC = 可由硬件置 1/ 清零的位

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-8

未实现：读为 0

bit 7-4

YRTEN<3:0>：年的十位数字的二 – 十进制码（Binary Coded Decimal，BCD）值  
包含从 0 到 9 的值。

bit 3-0

YRONE<3:0>：年的个位数字的 BCD 值  
包含从 0 到 9 的值。

注 1：仅当 RTCWREN = 1 时才允许写入 YEAR 寄存器。

### 寄存器 20-5 : MTHDY : 月和日值寄存器<sup>(1)</sup>

U-0, HSC	U-0, HSC	U-0, HSC	R/W-x, HSC	R/W-x, HSC	R/W-x, HSC	R/W-x, HSC	R/W-x, HSC	R/W-x, HSC
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0	
bit 15								bit 8
R-0, HSC	U-0, HSC	R/W-x, HSC						
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0	
bit 7								bit 0

图注：

HSC = 可由硬件置 1/ 清零的位

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

bit 15-13

未实现：读为 0

bit 12

MTHTEN0：月的十位数字的 BCD 值  
包含为 0 或 1 的值。

bit 11-8

MTHONE<3:0>：月的个位数字的 BCD 值  
包含从 0 到 9 的值。

bit 7-6

未实现：读为 0

bit 5-4

DAYTEN<1:0>：日的十位数字的 BCD 值  
包含从 0 到 3 的值。

bit 3-0

DAYONE<3:0>：日的个位数字的 BCD 值  
包含从 0 到 9 的值。

注 1：仅当 RTCWREN = 1 时才允许写入该寄存器。

## 寄存器 20-6 : WKDYHR : 星期和小时值寄存器<sup>(1)</sup>

U-0, HSC	R/W-x, HSC	R/W-x, HSC	R/W-x, HSC				
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 15							

U-0, HSC	U-0, HSC	R/W-x, HSC					
—	—	HRTEN1	HRTENO	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							

图注 : HSC = 可由硬件置 1/ 清零的位

R = 可读位 W = 可写位 U = 未实现位 , 读为 0

-n = 上电复位时的值 1 = 置 1 0 = 清零 x = 未知

bit 15-11 未实现 : 读为 0

bit 10-8 WDAY<2:0> : 星期数字的 BCD 值  
包含从 0 到 6 的值。

bit 7-6 未实现 : 读为 0

bit 5-4 HRTEN<1:0> : 小时的十位数字的 BCD 值  
包含从 0 到 2 的值。

bit 3-0 HRONE<3:0> : 小时的个位数字的 BCD 值  
包含从 0 到 9 的值。

注 1 : 仅当 RTCWREN = 1 时才允许写入该寄存器。

## 寄存器 20-7 : MINSEC : 分钟和秒值寄存器

U-0, HSC	R/W-x, HSC						
—	MINTEN2	MINTEN1	MINTENO	MINONE3	MINONE2	MINONE1	MINONE0
bit 15							

U-0, HSC	R/W-x, HSC						
—	SECTEN2	SECTEN1	SECTENO	SECOME3	SECOME2	SECOME1	SECOME0
bit 7							

图注 : HSC = 可由硬件置 1/ 清零的位

R = 可读位 W = 可写位 U = 未实现位 , 读为 0

-n = 上电复位时的值 1 = 置 1 0 = 清零 x = 未知

bit 15 未实现 : 读为 0

bit 14-12 MINTEN<2:0> : 分钟的十位数字的 BCD 值  
包含从 0 到 5 的值。

bit 11-8 MINONE<3:0> : 分钟的个位数字的 BCD 值  
包含从 0 到 9 的值。

bit 7 未实现 : 读为 0

bit 6-4 SECTEN<2:0> : 秒的十位数字的 BCD 值  
包含从 0 到 5 的值。

bit 3-0 SECOME<3:0> : 秒的个位数字的 BCD 值  
包含从 0 到 9 的值。

# PIC24FJ64GB004 系列

## 20.2.6 ALRMVAL 寄存器映射

### 寄存器 20-8 : ALMTHDY : 闹钟月和日值寄存器<sup>(1)</sup>

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 15							bit 8

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

#### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-13 未实现 : 读为 0  
bit 12 MTHTEN0 : 月的十位数字的 BCD 值  
包含为 0 或 1 的值。  
bit 11-8 MTHONE<3:0> : 月的个位数字的 BCD 值  
包含从 0 到 9 的值。  
bit 7-6 未实现 : 读为 0  
bit 5-4 DAYTEN<1:0> : 日的十位数字的 BCD 值  
包含从 0 到 3 的值。  
bit 3-0 DAYONE<3:0> : 日的个位数字的 BCD 值  
包含从 0 到 9 的值。

注 1 : 仅当 RTCWREN = 1 时才允许写入该寄存器。

### 寄存器 20-9 : ALWDHHR : 闹钟星期和小时值寄存器<sup>(1)</sup>

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 15							bit 8

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

#### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-11 未实现 : 读为 0  
bit 10-8 WDAY<2:0> : 星期数字的 BCD 值  
包含从 0 到 6 的值。  
bit 7-6 未实现 : 读为 0  
bit 5-4 HRTEN<1:0> : 小时的十位数字的 BCD 值  
包含从 0 到 2 的值。  
bit 3-0 HRONE<3:0> : 小时的个位数字的 BCD 值  
包含从 0 到 9 的值。

注 1 : 仅当 RTCWREN = 1 时才允许写入该寄存器。

## 寄存器 20-10 : ALMINSEC : 闹钟分钟和秒值寄存器

U-0	R/W-x						
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 15	bit 8						

U-0	R/W-x						
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- |           |  |
|-----------|--|
| bit 15    | 未实现 : 读为 0   |
| bit 14-12 | <b>MINTEN&lt;2:0&gt;</b> : 分钟的十位数字的 BCD 值<br>包含从 0 到 5 的值。 |
| bit 11-8  | <b>MINONE&lt;3:0&gt;</b> : 分钟的个位数字的 BCD 值<br>包含从 0 到 9 的值。 |
| bit 7     | 未实现 : 读为 0   |
| bit 6-4   | <b>SECTEN&lt;2:0&gt;</b> : 秒的十位数字的 BCD 值<br>包含从 0 到 5 的值。  |
| bit 3-0   | <b>SECONE&lt;3:0&gt;</b> : 秒的个位数字的 BCD 值<br>包含从 0 到 9 的值。  |

## 20.3 校准

实时晶振输入可使用周期性自动调整功能来校准。正确校准后，RTCC 可实现每月小于 3 秒的误差。这可通过确定误差时钟脉冲数并将其存储到 RCFGCAL 寄存器的低半部分完成。装入到 RCFGCAL 低半部分的 8 位有符号值与 4 相乘，然后每分钟与 RTCC 定时器的值相加或相减一次。请参见以下步骤校准 RTCC：

1. 使用器件上的其他定时器资源，用户必须确定 32.768 kHz 晶振的误差。
2. 知道误差后，必须将其转换为每分钟的误差时钟脉冲数。
3. a) 如果振荡器频率高于理想值（步骤 2 中的计算结果为负），那么 RCFGCAL 寄存器的值就必须为负。这样每分钟会从定时器计数器中减去指定的时钟脉冲数。  
b) 如果振荡器频率低于理想值（步骤 2 中的计算结果为正），那么 RCFGCAL 寄存器的值就必须为正。这样每分钟会从定时器计数器中加上指定的时钟脉冲数。

将每分钟的误差时钟数除以 4 得到正确的校准值并将该值装入 RCFGCAL 寄存器。（校准值的每 1 位递增 1 都会加上或减去 4 个脉冲）。

### 公式 20-1：

$$(\text{理想频率 } \dagger - \text{测得频率}) * 60 = \text{每分钟的时钟数}$$

$$\dagger \text{ 理想频率} = 32,768 \text{ Hz}$$

仅当定时器关闭或者紧跟在秒脉冲的上升沿后才可写入 RCFGCAL 寄存器的低半部分。

**注：**是否在误差值中包含由于温漂和晶振老化造成的误差由用户自行决定。

## 20.4 闹钟

- 闹钟的时间间隔可配置为从半秒到一年
- 使用 ALRMEN 位 (ALCFGRPT<15>) 使能
- 可选择一次性闹钟和重复闹钟

### 20.4.1 配置闹钟

使用 ALRMEN 位使能闹钟功能。闹钟事件发生后该位清零。只有在 ALRMEN = 0 时才允许写入 ALRMVAL。

如图 20-2 所示，可通过 AMASK 位 (ALCFGRPT<13:10>) 配置闹钟时间间隔。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。

也可配置闹钟根据预配置的时间间隔重复。一旦使能了闹钟，闹钟发生的总次数就会被存储到 ARPT<7:0> 位 (ALCFGRPT<7:0>)。当 ARPT 位的值等于 00h 且 CHIME 位 (ALCFGRPT<14>) 清零时，禁止重复功能，仅发生单次闹钟。通过将 FFh 装入 ARPT<7:0>，可使闹钟重复最多 255 次。

每发生一次闹钟事件，ARPT 位的值都会减 1。值达到 00h，将最后一次发出闹钟，然后 ALRMEN 位自动清零，关闭闹钟。

如果 CHIME 位 = 1，闹钟可无限次重复发生。在这种情况下，当 ARPT 位的值达到 00h 时，不会禁止闹钟，而是返回 FFh 并继续无限次地计数。

### 20.4.2 闹钟中断

每一次闹钟事件都会产生一个中断。此外会提供闹钟脉冲输出，其频率是闹钟频率的一半。该输出与 RTCC 时钟完全同步，且可用作其他外设的触发时钟。

**注：**当使能闹钟 (ALRMEN = 1) 时，更改除 RCFGCAL 和 ALCFGRPT 寄存器外的任何寄存器以及 CHIME 位都将导致错误的闹钟事件，从而引起错误的闹钟中断。为了避免错误闹钟事件，只能在禁止闹钟 (ALRMEN = 0) 时更改定时器和闹钟值。建议在 RTCSYNC = 0 时更改 ALCFGRPT 寄存器和 CHIME 位。

图 20-2：闹钟屏蔽设置

闹钟屏蔽设置 (AMASK<3:0>)	星期	月	日	小时	分钟	秒
0000——每半秒	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
0001——每秒	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
0010——每 10 秒	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> s
0011——每分钟	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> s s
0100——每 10 分钟	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> m	<input type="checkbox"/> s s
0101——每小时	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> m	<input type="checkbox"/> m	<input type="checkbox"/> s s
0110——每日	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	h h	<input type="checkbox"/> m m	<input type="checkbox"/> s s
0111——每周	d	<input type="checkbox"/>	<input type="checkbox"/>	h h	<input type="checkbox"/> m m	<input type="checkbox"/> s s
1000——每月	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/> d d	h h	<input type="checkbox"/> m m	<input type="checkbox"/> s s
1001——每年 <sup>(1)</sup>	<input type="checkbox"/>	m m	/ d d	h h	<input type="checkbox"/> m m	<input type="checkbox"/> s s

注 1：每年，除非配置为 2 月 29 日。

# **PIC24FJ64GB004 系列**

---

---

注：

## 21.0 32 位可编程循环冗余校验 (CRC) 发生器

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 30 章“可编程循环冗余校验 (CRC)”(DS39714A\_CN)。

可编程 CRC 发生器为各种网络和安防应用提供快速生成校验和的硬件实现方法。它提供以下特性：

- 用户可编程的 CRC 多项式方程（最多 32 位）
- 可编程移位方向（小尾数或大尾数格式）
- 独立的数据和多项式长度
- 可配置中断输出
- 数据 FIFO

CRC 发生器的简化框图如图 21-1 所示。图 21-2 中给出了 CRC 移位引擎的简化形式。

图 21-1：CRC 框图

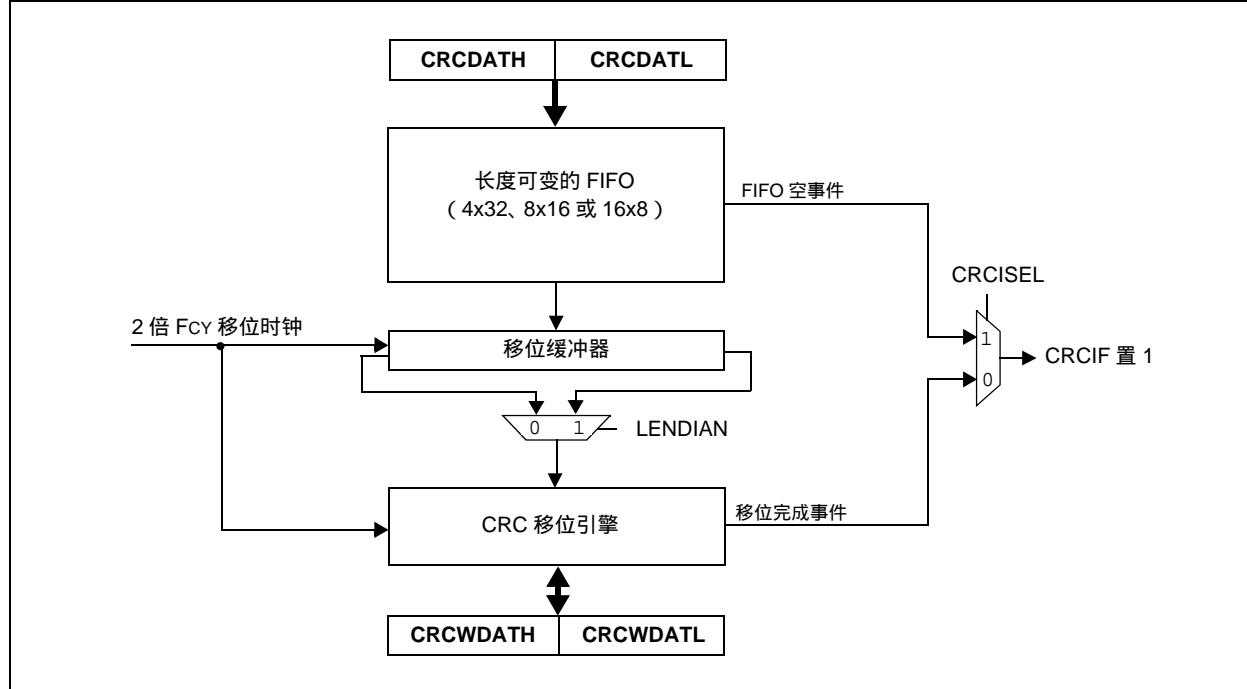
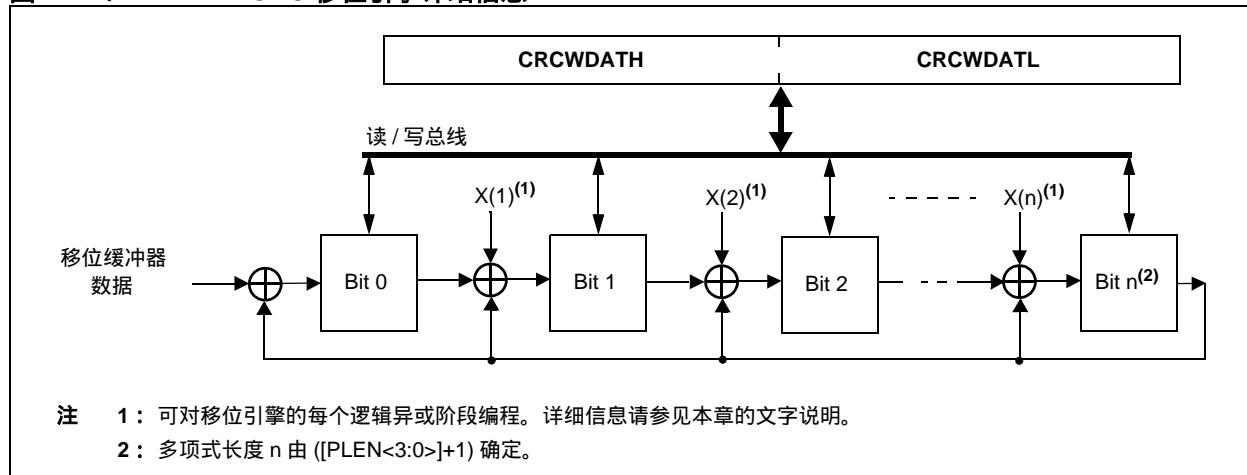


图 21-2：CRC 移位引擎详细信息



## 21.1 用户接口

### 21.1.1 多项式接口

可使用最多 32 位，将 CRC 模块编程为生成多达 32 阶的 CRC 多项式。通过 PLEN<4:0> 位 (CRCCON2<4:0>) 选择反映方程中最高次幂的多项式长度。

CRCXORL 和 CRCXORH 寄存器控制方程中包含的幂项。将寄存器中的某个位置 1 会在方程中包含相应的幂项，在功能上相当于包含在 CRC 引擎的相应位上执行的逻辑异或操作。该位清零会禁止此异或操作。

例如，考虑两个 CRC 多项式，一个是 16 位方程，另一个是 32 位方程：

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

和

$$\begin{aligned} &x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 \\ &+ x^5 + x^4 + x^2 + x + 1 \end{aligned}$$

要将这些多项式编程到 CRC 发生器中，请设置表 21-1 中的寄存器位。

注意，请将相应的存储单元设置为 1 以指示在方程中使用了它们（例如 X26 和 X23）。方程所需的 0 位始终为异或操作，X0 表示无关位。对于长度为 N 的多项式，假设始终使用第 N 位，而与该位的设置无关。因此，对于长度为 32 的多项式，在 CRCxOR 寄存器中没有第 32 位。

### 21.1.2 数据接口

CRC 模块具有适合各种数据宽度的 FIFO。可使用 DWIDTH<4:0> 位 (CRCCON2<12:8>) 将输入数据宽度配置为 1 到 32 位之间的任何数值。当数据宽度大于 15 时，FIFO 为 4 字深。当 DWIDTH 值在 8 至 15 之间时，FIFO 为 8 字深。当 DWIDTH 值小于 8 时，FIFO 为 16 字深。

表 21-1：CRC 设置示例（16 和 32 位多项式）

CRC 控制位	位值	
	16 位多项式	32 位多项式
PLEN<4:0>	01111	11111
X31:X16	0000 0000 0000 000x	0000 0100 1100 0001
X15:X0	0001 0000 0010 000X	0001 1101 1011 011X

首先把要进行 CRC 计算所需的数据写入 FIFO。即使数据宽度小于 8，可写入 FIFO 的最小数据元素也是 1 个字节。例如，如果 DWIDTH 值为 5，那么数据大小为 DWIDTH + 1，即 6。在这里，数据是作为整个字节写入的，CRC 模块忽略了该字节中两个未使用的高位。

数据一旦被写入 CRCDAT 寄存器的最高位（由数据宽度定义），VWORD<4:0> 位 (CRCCON1<12:8>) 的值将递增 1。例如，如果 DWIDTH 值为 24，则写入 CRCDATH 的 bit 7 时，VWORD 位递增 1。因此，CRCDAT 必须始终先于 CRCDATH 写入。

当 CRCGO 位置 1 且 VWORD 的值大于零时，CRC 引擎开始移动数据。每个字都从 FIFO 复制到缓冲寄存器（这会导致 VWORD 递减 1）。然后从缓冲寄存器中移出数据。CRC 引擎会以每指令周期两个位的速率继续移动数据，直到 VWORD 值达到 0。因此对于给定的数据宽度，每个字完成计算所需的指令周期数为数据宽度的一半。对于，完成 32 位单字数据的 CRC 计算需要 16 个周期。

当 VWORD 值达到所配置 DWIDTH 值（4、8 或 16）的最大值时，CRCFUL 位置 1。当 VWORD 值达到零时，CRCMPT 位置 1。当 CRCEN 为 0 时，FIFO 为空且 VWORD<4:0> 设置为 00000。

在写入 CRCDAT 之后，必须至少经过一个指令周期才可以读 VWORD 位。

### 21.1.3 数据移位方向

LENDIAN 位 (CRCCON1<3>) 用于控制移位方向。默认情况下，CRC 通过引擎从最高有效位开始移动数据。将 LENDIAN 置 1 (= 1) 可导致 CRC 从最低有效位开始移动数据。此设置可使 CRC 与各种通信机制更好地集成，并省去了用软件使位反转的开销。注意，这只是改变了将数据移入引擎的方向。CRC 计算的结果仍然是正常的 CRC 结果，不是反转的 CRC 结果。

### 21.1.4 中断操作

用户可以配置该模块在以下两种条件下产生中断。

如果 CRCISEL 为 0，当 VWORD<4:0> 位的值从 1 跳变到 0 时产生中断。如果 CRCISEL 为 1，在 CRC 操作完成且模块将 CRCGO 位设置为 0 时产生中断。手动将 CRCGO 设置为 0 不会产生中断。

### 21.1.5 典型操作

要使模块执行典型 CRC 计算，请执行以下步骤：

1. 将 CRCEN 位置 1 以使能模块。
2. 配置模块以实现所需的操作：
  - c) 使用 CRCXORL 和 CRCXORH 寄存器以及 PLEN<4:0> 位设置所需的多项式
  - d) 使用 DWIDTH 和 LENDIAN 位配置数据宽度和移位方向
  - e) 使用 CRCISEL 位选择所需的中断模式
3. 在 CRCFUL 位置 1 之前，通过写入 CRCDATL 和 CRCDATH 寄存器来预装载 FIFO，否则将无用于移位的数据。
4. 通过将 00h 写入 CRCWDATL 和 CRCWDATH 来清除旧数据。CRCWDAT 也可以保持不变以继续上一个暂停的计算。
5. 将 CRCGO 位置 1 以启动计算。
6. 将剩余数据写入 FIFO 中（当 FIFO 有可用空间时）。
7. 计算完成时，CRCGO 自动清零。CRCISEL = 1 时，将产生中断。
8. 读取 CRCWDATL 和 CRCWDATH 以获得计算结果。

### 21.2 寄存器

有 8 个与本模块关联的寄存器：

- CRCCON1
- CRCCON2
- CRCXORL
- CRCXORH
- CRCDATL
- CRCDATH
- CRCWDATL
- CRCWDATH

CRCCON1 和 CRCCON2 寄存器（寄存器 21-1 和寄存器 21-2）用于控制模块的操作和配置各种设置。CRCXOR 寄存器（寄存器 21-3 和寄存器 21-4）用于选择 CRC 方程中要使用的多项式项。CRCDAT 和 CRCWDAT 寄存器是分别用作双字输入数据和 CRC 处理结果输出的缓冲器的寄存器对。

# PIC24FJ64GB004 系列

寄存器 21-1 : CRCCON1 : CRC 控制寄存器 1

R/W-0	U-0	R/W-0	R-0	R-0	R-0	R-0	R-0
CRCEN	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0
bit 15	bit 8						

R-0, HCS	R-1, HCS	R/W-0	R/W-0, HC	R/W-0	U-0	U-0	U-0
CRCFUL	CRCMPT	CRCISEL	CRCGO	LENDIAN	—	—	—
bit 7	bit 0						

图注 :	HC = 可由硬件清零的位	HCS = 可由硬件清零 / 置 1 的位
R = 可读位	W = 可写位	U = 未实现位 , 读为 0
-n = 上电复位时的值	1 = 置 1	0 = 清零

- bit 15      **CRCEN** : CRC 使能位  
    1 = 使能模块  
    0 = 禁止模块。所有状态机、指针和 CRCWDAT/CRCDAT 都复位；其他 SFR 不复位
- bit 14      **未实现** : 读为 0
- bit 13      **CSIDL** : 空闲模式 CRC 停止位  
    1 = 当器件进入空闲模式时，模块停止工作  
    0 = 模块在空闲模式下继续工作
- bit 12-8     **VWORD<4:0>** : 指针值位  
    指出 FIFO 中有效字的个数。当 PLEN<3:0> > 7 时，最大值为 8，当 PLEN<3:0> ≤ 7 时，最大值为 16。
- bit 7        **CRCFUL** : FIFO 满位  
    1 = FIFO 为满  
    0 = FIFO 未满
- bit 6        **CRCMPT** : FIFO 空位  
    1 = FIFO 为空  
    0 = FIFO 非空
- bit 5        **CRCISEL** : CRC 中断选择位  
    1 = FIFO 空时中断；CRC 计算未完成  
    0 = 移位完成时中断且 CRCWDAT 结果已计算好
- bit 4        **CRCGO** : 启动 CRC 位  
    1 = 启动 CRC 串行移位器  
    0 = 关闭 CRC 串行移位器
- bit 3        **LENDIAN** : 数据移位方向选择位  
    1 = 数据字从最低有效位开始移入 CRC (小尾数格式)  
    0 = 数据字从最高有效位开始移入 CRC (大尾数格式)
- bit 2-0      **未实现** : 读为 0

## 寄存器 21-2 : CRCCON2 : CRC 控制寄存器 2

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	DWIDTH4	DWIDTH3	DWIDTH2	DWIDTH1	DWIDTH0
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
—	—	—	PLEN4	PLEN3	PLEN2	PLEN1	PLEN0
bit 7							bit 0

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 未实现 : 读为 0

bit 12-8 DWIDTH<4:0> : 数据宽度选择位

定义数据字的宽度 ( 数据字宽度 = (DWIDHT<4:0>) + 1 )

bit 7-5 未实现 : 读为 0

bit 4-0 PLEN<4:0> : 多项式长度选择位

定义 CRC 多项式的长度 ( 多项式长度 = (PLEN<4:0>) + 1 )

## 寄存器 21-3 : CRCXORL : CRC 多项式异或操作寄存器 ( 低字节 )

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
X15	X14	X13	X12	X11	X10	X9	X8
bit 15							bit 8

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 = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-1 X<15:1> : 多项式的项  $X^n$  的异或操作使能位

bit 0 未实现 : 读为 0

# PIC24FJ64GB004 系列

寄存器 21-4 : CRCXORH : CRC 多项式异或操作寄存器 (高字节)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
X31	X30	X29	X28	X27	X26	X25	X24
bit 15							bit 8

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| X23   | X22   | X21   | X20   | X19   | X18   | X17   | X16   |
| bit 7 |       |       |       |       |       |       | bit 0 |

## 图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-0 X<31:16> : 多项式的项  $X^n$  的异或操作使能位

## 22.0 10 位高速 A/D 转换器

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 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 ksps
- 13 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
- 可选转换触发源
- 16 字转换结果缓冲区
- 可选缓冲区填充模式
- 四种结果对齐选项
- 可在 CPU 休眠和空闲模式下继续工作

在所有 PIC24FJ64GB004 系列器件上，10 位 A/D 转换器都具有 13 个模拟输入引脚，标记为 AN0 至 AN12。此外，还有两个可连接外部参考电压的模拟输入引脚 (VREF+ 和 VREF-)。这两个参考电压输入引脚可以和其他模拟输入引脚复用。

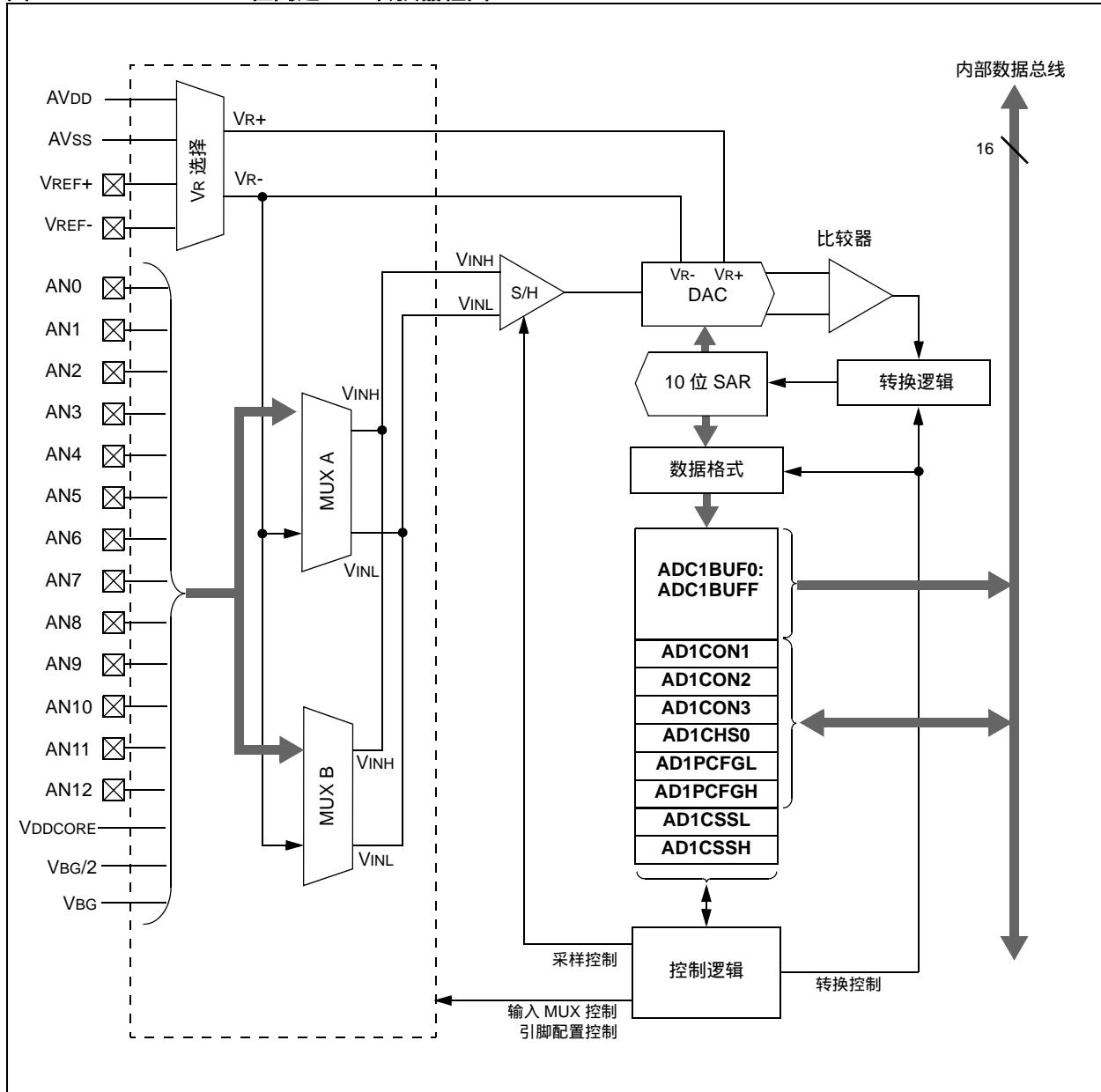
图 22-1 为 A/D 转换器的框图。

要执行一个 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 (AD1PCFGL<15:0> 和 AD1PCFGH<1:0>)。
  - 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便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) 选择中断发生的频率 (AD1CON2<5:2>)。
  - g) 启动 A/D 模块 (AD1CON1<15>)。
2. 配置 A/D 中断 (如果需要)：
  - a) 清零 AD1IF 位。
  - b) 选择 A/D 中断优先级。

# PIC24FJ64GB004 系列

图 22-1：10 位高速 A/D 转换器框图



## 寄存器 22-1 : 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 <sup>(1)</sup>	—	ADSIDL	—	—	—	FORM1	FORM0
bit 15	bit 8						

R/W-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0, HCS	R/C-0, HCS
SSRC2	SSRC1	SSRC0	—	—	ASAM	SAMP	DONE
bit 7	bit 0						

### 图注 :

C = 可清零位

HCS = 可由硬件清零 / 置 1 的位

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **ADON** : A/D 工作模式位<sup>(1)</sup>

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 = 模块在空闲模式下继续工作

bit 12-10    未实现 : 读为 0

bit 9-8      **FORM<1:0>** : 数据输出格式位

11 = 有符号的小数 ( sddd dddd dd00 0000 )  
10 = 小数 ( dddd dddd dd00 0000 )  
01 = 有符号的整数 ( 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 = 内部计数器结束采样并启动转换 ( 自动转换 )  
110 = CTMU 事件结束采样并启动转换  
101 = 保留  
100 = Timer5 比较结束采样并启动转换  
011 = 保留  
010 = Timer3 比较结束采样并启动转换  
001 = 由 INT0 引脚的有效跳变沿结束采样并启动转换  
000 = 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 = 采样在上次转换完成后立即开始。 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 / 保持放大器正在采样输入  
0 = A/D 采样 / 保持放大器正在保持采样结果

bit 0      **DONE** : A/D 转换状态位

1 = A/D 转换完成  
0 = A/D 转换未完成

注 1 : 清零 ADON 位后 , 不会保持 ADC1BUFx 寄存器的值。请在禁止 A/D 模块之前从缓冲区中读出转换值。

# PIC24FJ64GB004 系列

## 寄存器 22-2 : AD1CON2 : A/D 控制寄存器 2

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

R-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUFS	—	SMP13	SMP12	SMP11	SMP10	BUFM	ALTS
bit 7	bit 0						

图注 :	r = 保留位
R = 可读位	W = 可写位
-n = 上电复位时的值	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 保留 : 保持为 0

bit 11 未实现 : 读为 0

bit 10 CSCNA : MUX A 输入多路选择的 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 (只在 BUFM = 1 时有效)

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

0 = A/D 当前在填充缓冲区 00-07 , 用户应该访问 08-0F 中的数据

bit 6 未实现 : 读为 0

bit 5-2 SMP<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 / 转换序列产生一次中断

1110 = 每完成 15 个采样 / 转换序列产生一次中断

.....

0001 = 每完成 2 个采样 / 转换序列产生一次中断

0000 = 每完成 1 个采样 / 转换序列产生一次中断

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 (ADC1BUFn<15:8> 和 ADC1BUFn<7:0> )

0 = 缓冲区配置为一个 16 字缓冲区 (ADC1BUFn<15:0> )

bit 0 ALTS : 交替输入采样模式选择位

1 = 第一次采样时 , 使用 MUX A 输入多路开关设置 , 然后交替使用 MUX B 和 MUX A 输入多路  
开关设置进行后续采样

0 = 总是使用 MUX A 输入多路开关设置

## 寄存器 22-3 : AD1CON3 : A/D 控制寄存器 3

R/W-0	r-0	r-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADRC	r	r	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0
bit 15	bit 8						

| 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 = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **ADRC** : A/D 转换时钟源位

1 = A/D 内部的 RC 时钟

0 = 时钟由系统时钟产生

bit 14-13    保留 : 保持为 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 至 01000000 = 保留

.....

00111111 = 64 • TCY

.....

00000001 = 2 • TCY

00000000 = TCY

# PIC24FJ64GB004 系列

## 寄存器 22-4 : AD1CHS : A/D 输入选择寄存器

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NB	—	—	CH0SB4 <sup>(1,2)</sup>	CH0SB3 <sup>(1,2)</sup>	CH0SB2 <sup>(1,2)</sup>	CH0SB1 <sup>(1,2)</sup>	CH0SB0 <sup>(1,2)</sup>
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 = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **CH0NB** : MUX B 多路开关选择的通道 0 负输入设置位

1 = 通道 0 的负输入是 AN1

0 = 通道 0 的负输入是 VR-

bit 14-13    **未实现** : 读为 0

bit 12-8     **CH0SB<4:0>** : MUX B 多路开关选择的通道 0 正输入设置位<sup>(1,2)</sup>

11111 = 通道 0 正输入 , 保留仅供 CTMU 使用<sup>(3)</sup>

1xxxx = 未实现 ; 不要使用。

01111 = 通道 0 的正输入是内部带隙参考 ( VBG )

01110 = 通道 0 的正输入是 VBG/2

01101 = 通道 0 的正输入是稳压器输出 ( VDDCORE )

01100 = 通道 0 的正输入是 AN12

01011 = 通道 0 的正输入是 AN11

01010 = 通道 0 的正输入是 AN10

01001 = 通道 0 的正输入是 AN9

01000 = 通道 0 的正输入是 AN8

00111 = 通道 0 的正输入是 AN7

00110 = 通道 0 的正输入是 AN6

00101 = 通道 0 的正输入是 AN5

00100 = 通道 0 的正输入是 AN4

00011 = 通道 0 的正输入是 AN3

00010 = 通道 0 的正输入是 AN2

00001 = 通道 0 的正输入是 AN1

00000 = 通道 0 的正输入是 AN0

bit 7        **CH0NA** : MUX A 多路开关选择的通道 0 负输入设置位

1 = 通道 0 的负输入是 AN1

0 = 通道 0 的负输入是 VR-

bit 6-5     **未实现** : 读为 0

bit 4-0     **CH0SA<4:0>** : MUX A 多路开关选择的通道 0 正输入设置位

实现的组合与上面的 CH0SB<4:0> 的组合相同。

**注** 1 : 此处未显示的组合未实现 , 不要使用。

2 : 模拟通道 AN6、 AN7、 AN8 和 AN12 在 28 引脚器件上未提供 , 请不要使用。

3 : 选择该内部通道 , 允许 CTMU 模块使用 A/D 转换器的采样和保持电容 ( CAD ) 测量最短的时间。

## 寄存器 22-5 : AD1PCFG : A/D 端口配置寄存器

R/W-0	R/W-0	R/W-0	R/W-0 <sup>(1)</sup>	R/W-0	R/W-0	R/W-0	R/W-0 <sup>(1)</sup>
PCFG15	PCFG14	PCFG13	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8
bit 15	bit 8						

R/W-0 <sup>(1)</sup>	R/W-0 <sup>(1)</sup>	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
bit 7	bit 0						

### 图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **PCFG15** : A/D 输入带隙参考使能位

1 = 禁止内部带隙 ( VBG ) 参考通道  
0 = 使能内部带隙参考通道

bit 14      **PCFG14** : A/D 输入半带隙参考使能位

1 = 禁止内部半带隙 ( VBG/2 ) 参考通道  
0 = 使能内部半带隙参考通道

bit 13      **PCFG13** : A/D 输入稳压器输出参考使能位

1 = 禁止内部稳压器输出 ( VDDCORE ) 参考通道  
0 = 使能内部稳压器输出参考通道

bit 12-0     **PCFG<12:0>** : 模拟输入引脚配置控制位<sup>(1)</sup>

1 = 相应模拟通道的引脚被配置为数字模式 ; 使能 I/O 端口读操作  
0 = 引脚被配置为模拟模式 ; 禁止 I/O 端口读操作 , A/D 对引脚电压进行采样

注 1 : 模拟通道 AN6、AN7、AN8 和 AN12 在 28 引脚器件上未提供 , 保留这些通道对应的位置 1。

# PIC24FJ64GB004 系列

寄存器 22-6 : AD1CSSL : A/D 输入扫描选择寄存器

R/W-0	R/W-0	R/W-0	R/W-0 <sup>(1)</sup>	R/W-0	R/W-0	R/W-0	R/W-0
CSSL15	CSSL14	CSSL13	CSSL12	CSSL11	CSSL10	CSSL9	CSSL8 <sup>(1)</sup>
bit 15							bit 8

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| CSSL7 | CSSL6 | CSSL5 | CSSL4 | CSSL3 | CSSL2 | CSSL1 | CSSL0 |
| bit 7 |       |       |       |       |       |       | bit 0 |

图注 :

R = 可读位

W = 可写位

U = 未实现位 , 读为 0

-n = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **CSSL15** : A/D 输入带隙扫描使能位

1 = 对内部带隙 ( VBG ) 通道使能输入扫描  
0 = 对模拟通道禁止输入扫描

bit 14      **CSSL14** : A/D 输入半带隙扫描使能位

1 = 对内部半带隙 ( VBG/2 ) 通道使能输入扫描  
0 = 对模拟通道禁止输入扫描

bit 13      **CSSL13** : A/D 输入稳压器输出扫描使能位

1 = 对内部稳压器输出 ( VDDCORE ) 使能输入扫描  
0 = 对模拟通道禁止输入扫描

bit 12-0      **CSSL<12:0>** : A/D 输入引脚扫描选择位

1 = 选择对应的模拟通道进行输入扫描  
0 = 输入扫描时跳过模拟通道

注 1 : 模拟通道 AN6、AN7、AN8 和 AN12 在 28 引脚器件上未提供 , 保留这些通道对应的位清零。

公式 22-1 :

A/D 转换时钟周期<sup>(1)</sup>

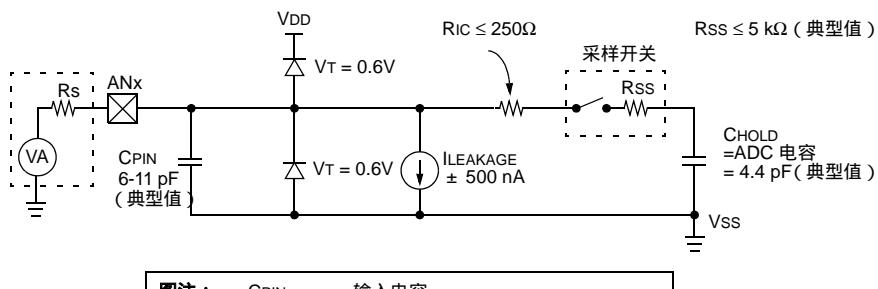
$$ADCS = \frac{T_{AD}}{T_{CY}} - 1$$

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

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

图 22-2：

10 位 A/D 转换器模拟输入模型

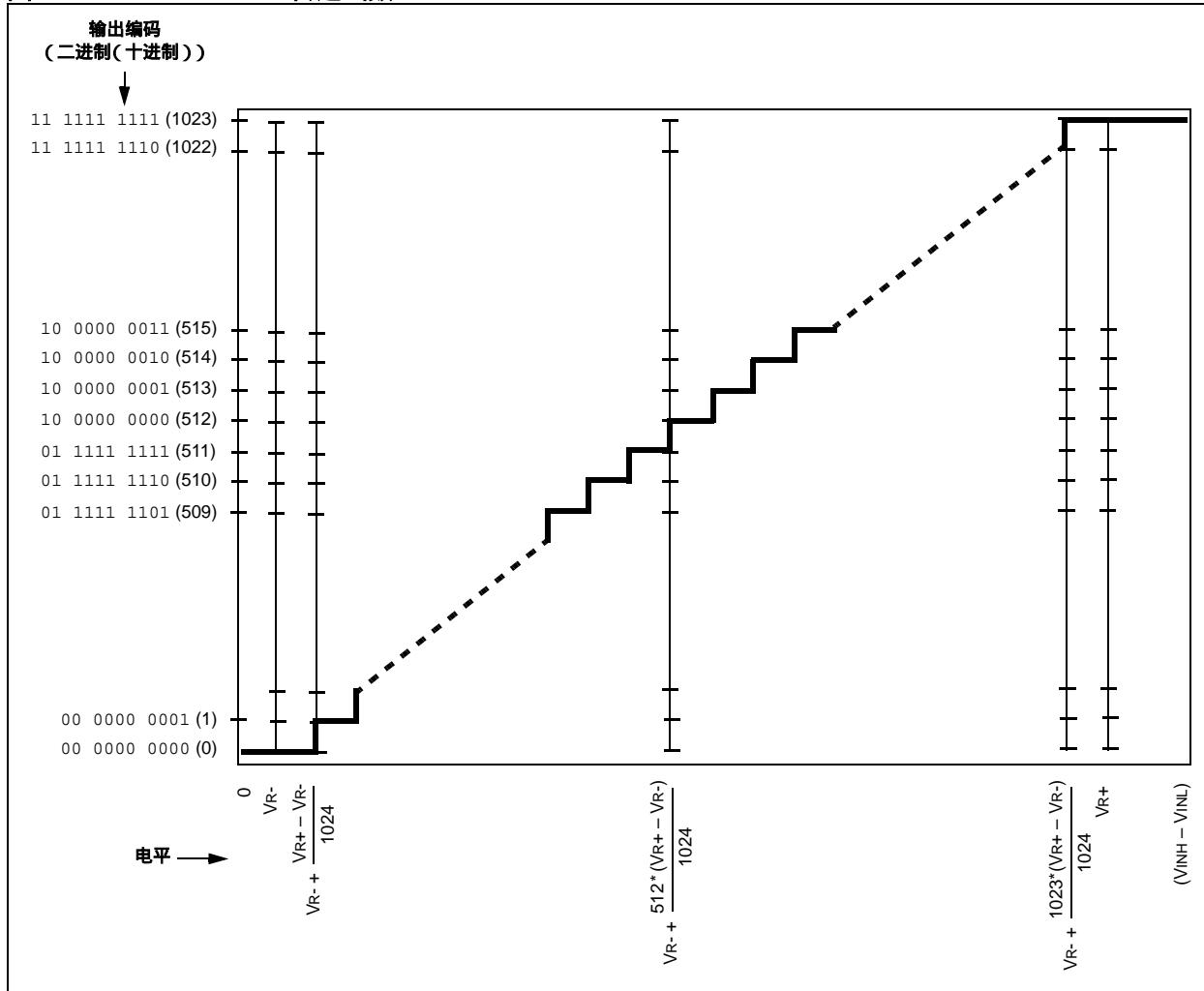


图注：	CPIN	= 输入电容
	VT	= 门限电压
	ILEAKAGE	= 各个连接点在引脚上产生的泄漏电流
	RIC	= 内部走线等效阻抗
	RSS	= 采样开关阻抗
	CHOLD	= 采样 / 保持电容(来自 ADC )

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

# PIC24FJ64GB004 系列

图 22-3： A/D 传递函数



## 23.0 三比较器模块

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本参考手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 19 章“比较器模块”(DS39710A\_CN)。

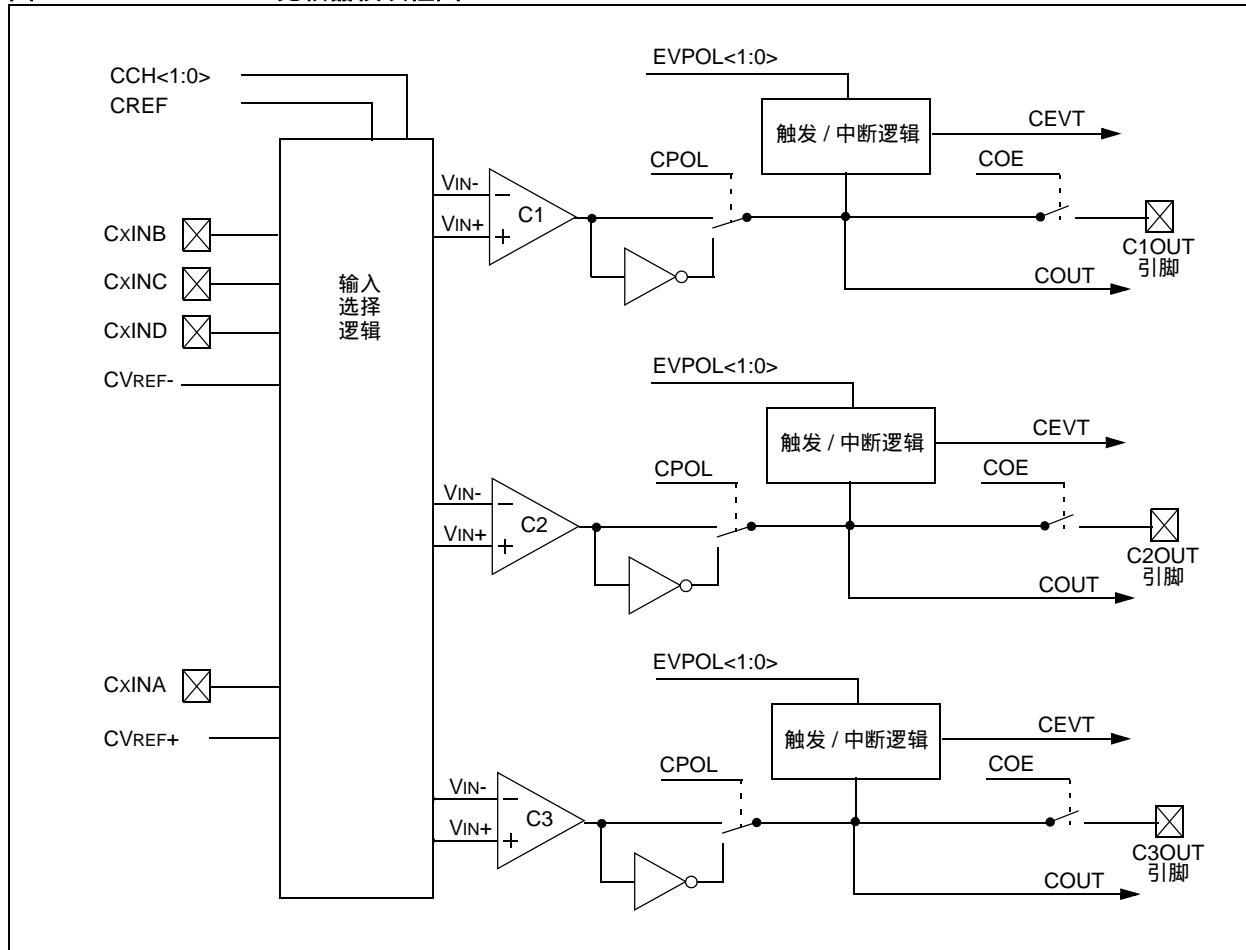
三比较器模块提供三个双输入比较器。至比较器的输入可配置为使用四个外部模拟输入之一，且参考电压输入可以来自参考电压发生器和带隙参考。

比较器输出可直接连接到 CxOUT 引脚。当对应的 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。

模块的简化框图如图 23-1 所示。图 23-2 中给出了各种可能的比较器配置图。

每个比较器都有自己的控制寄存器 CMxCON (寄存器 23-1)，用于使能和配置其操作。所有三个比较器的输出和事件状态都在 CMSTAT 寄存器 (寄存器 23-2) 中给出。

图 23-1：三比较器模块框图



# PIC24FJ64GB004 系列

图 23-2：各个比较器配置

<b>比较器关闭</b> $CEN = 0, CREF = x, CCH<1:0> = xx$	
<b>比较器 <math>CxINB &gt; CxINA</math> 比较</b> $CEN = 1, CREF = 0, CCH<1:0> = 00$	<b>比较器 <math>CxINC &gt; CxINA</math> 比较</b> $CEN = 1, CREF = 0, CCH<1:0> = 01$
<b>比较器 <math>CxIND &gt; CxINA</math> 比较</b> $CEN = 1, CREF = 0, CCH<1:0> = 10$	<b>比较器 <math>CVREF- &gt; CxINA</math> 比较</b> $CEN = 1, CREF = 0, CCH<1:0> = 11$
<b>比较器 <math>CxINB &gt; CVREF+</math> 比较</b> $CEN = 1, CREF = 1, CCH<1:0> = 00$	<b>比较器 <math>CxINC &gt; CVREF+</math> 比较</b> $CEN = 1, CREF = 1, CCH<1:0> = 01$
<b>比较器 <math>CxIND &gt; CVREF+</math> 比较</b> $CEN = 1, CREF = 1, CCH<1:0> = 10$	<b>比较器 <math>VBG &gt; CVREF+</math> 比较</b> $CEN = 1, CREF = 1, CCH<1:0> = 11$

## 寄存器 23-1 : CMxCON : 比较器 x 控制寄存器 (比较器 1 - 3)

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R-0
CEN	COE	CPOL	—	—	—	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 = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15      **CEN** : 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 = 在 CxOUT 引脚上提供比较器输出  
               0 = 比较器输出仅在内部进行
- 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 12-10    未实现 : 读为 0
- bit 9        **CEVT** : 比较器事件位  
               1 = 发生由 EVPOL<1:0> 定义的比较器事件 ; 该位清零之前禁止后续触发和中断  
               0 = 未发生比较器事件
- bit 8        **COUT** : 比较器输出位  
当 CPOL = 0 时 :  
               1 = VIN+ > VIN-  
               0 = VIN+ < VIN-  
当 CPOL = 1 时 :  
               1 = VIN+ < VIN-  
               0 = VIN+ > VIN-
- bit 7-6      **EVPOL<1:0>** : 触发 / 事件 / 中断极性选择位  
               11 = 比较器输出的任何变化导致发生触发 / 事件 / 中断 (当 CEVT = 0 时 )  
               10 = 比较器输出的以下跳变导致发生触发 / 事件 / 中断 :  
                   如果 CPOL = 0 (同相极性) :  
                   仅从高电平到低电平的跳变。  
                   如果 CPOL = 1 (反相极性) :  
                   仅从低电平到高电平的跳变。  
               01 = 比较器输出的以下跳变导致发生触发 / 事件 / 中断 :  
                   如果 CPOL = 0 (同相极性) :  
                   仅从低电平到高电平的跳变。  
                   如果 CPOL = 1 (反相极性) :  
                   仅从高电平到低电平的跳变。  
               00 = 禁止发生触发 / 事件 / 中断
- bit 5        未实现 : 读为 0

# PIC24FJ64GB004 系列

## 寄存器 23-1 : CMxCON : 比较器 x 控制寄存器 (比较器 1 - 3) (续)

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+ 输入参考电压  
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 = 比较器反相输入连接到 CVREF+ 输入参考电压  
10 = 比较器反相输入连接到 CxIND 引脚  
01 = 比较器反相输入连接到 CxINC 引脚  
00 = 比较器反相输入连接到 CxINB 引脚

## 寄存器 23-2 : CMSTAT : 比较器模块状态寄存器

R/W-0	U-0	U-0	U-0	U-0	R-0	R-0	R-0
CMIDL	—	—	—	—	C3EVT	C2EVT	C1EVT
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	R-0	R-0	R-0
—	—	—	—	—	C3OUT	C2OUT	C1OUT
bit 7							bit 0

### 图注 :

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

bit 15      **CMIDL** : 比较器空闲模式停止位  
1 = 当器件进入空闲模式时 , 所有比较器停止工作  
0 = 所有使能的比较器在空闲模式下继续工作

bit 14-11    未实现 : 读为 0

bit 10       **C3EVT** : 比较器 3 事件状态位 (只读)  
显示比较器 3 的当前事件状态 (CM3CON<9>)。

bit 9        **C2EVT** : 比较器 2 事件状态位 (只读)  
显示比较器 2 的当前事件状态 (CM2CON<9>)。

bit 8        **C1EVT** : 比较器 1 事件状态位 (只读)  
显示比较器 1 的当前事件状态 (CM1CON<9>)。

bit 7-3      未实现 : 读为 0

bit 2        **C3OUT** : 比较器 3 输出状态位 (只读)  
显示比较器 3 的当前输出 (CM3CON<8>)。

bit 1        **C2OUT** : 比较器 2 输出状态位 (只读)  
显示比较器 2 的当前输出 (CM2CON<8>)。

bit 0        **C1OUT** : 比较器 1 输出状态位 (只读)  
显示比较器 1 的当前输出 (CM1CON<8>)。

## 24.0 比较器参考电压

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 20 章“比较器参考电压模块”(DS39709A\_CN)。

的电压范围。这两种电压范围的主要区别在于 CVREF 选择位 (CVR<3:0>) 所选的步长不同，其中一个范围提供更高的分辨率。

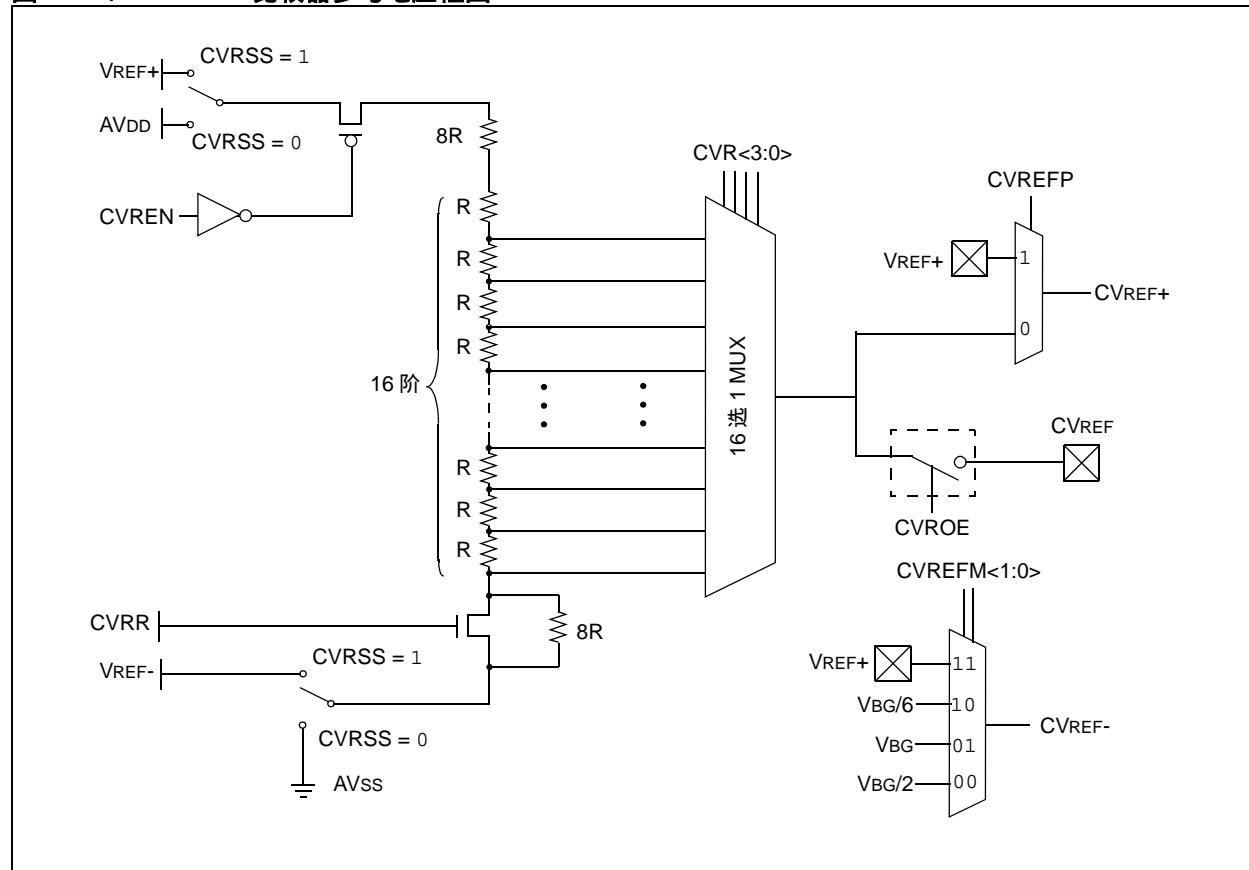
比较器参考电压模块的电源可以来自 VDD 和 Vss 或外部 VREF+ 和 VREF-。由 CVRSS 位 (CVRCON<4>) 选择电压源。

在改变 CVREF 输出值时，必须考虑到比较器参考电压的稳定时间。

## 24.1 配置比较器参考电压

参考电压模块由 CVRCON 寄存器 (寄存器 24-1) 控制，它能提供两种范围的输出电压，每种范围都具有 16 种不同的电平。CVRR 位 (CVRCON<5>) 选择要使用

图 24-1：比较器参考电压框图



# PIC24FJ64GB004 系列

寄存器 24-1 : CVRCON : 比较器参考电压控制寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	CVREFP	CVREFM1	CVREFM0
bit 15							bit 8

| 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 = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

- bit 15-11 未实现 : 读为 0
- bit 10 **CVREFP** : CVREF+ 参考输出选择位  
1 = VREF+ 输入引脚用作到比较器的 CVREF+ 参考输出  
0 = 比较器参考电压模块生成的输出用作到比较器的 CVREF+ 参考输出
- bit 9-8 **CVREFM<1:0>** : CVREF- 参考输出选择位  
11 = VREF+ 输入引脚用作到比较器的 CVREF- 参考输出  
10 = VBG/6 用作到比较器的 CVREF- 参考输出  
01 = VBG 用作到比较器的 CVREF- 参考输出  
00 = VBG/2 用作到比较器的 CVREF- 参考输出
- bit 7 **CVREN** : 比较器参考电压使能位  
1 = CVREF 电路上电  
0 = CVREF 电路掉电
- bit 6 **CVROE** : 比较器 VREF 输出使能位  
1 = CVREF 电压从 CVREF 引脚输出  
0 = CVREF 电压与 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 = 比较器参考源 CVRSRC = VREF+ - VREF-  
0 = 比较器参考源 CVRSRC = AVDD - AVSS
- bit 3-0 **CVR<3:0>** : 比较器 VREF 值选择位 ( 0 ≤ CVR<3:0> ≤ 15 )  
当 CVRR = 1 时 :  
CVREF = (CVR<3:0>/ 24) • (CVRSRC)  
当 CVRR = 0 时 :  
CVREF = 1/4 • (CVRSRC) + (CVR<3:0>/32) • (CVRSRC)

## 25.0 充电时间测量单元 (CTMU)

**注：**本数据手册总结了该组 PIC24F 器件的功能。但是不应把本参考手册当作无所不包的参考手册来使用。如需了解更多信息，请参见《PIC24F 系列参考手册》中的第 11 章“充电时间测量单元 (CTMU)”(DS39724A\_CN)。

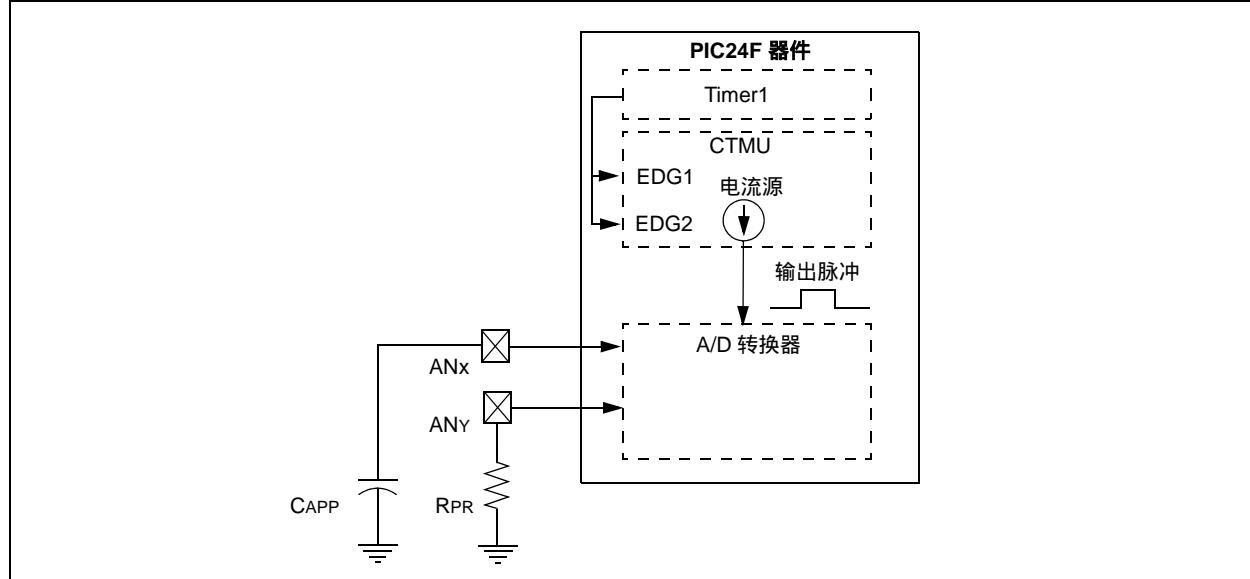
充电时间测量单元是一个灵活的模拟模块，它提供对脉冲源输出脉冲时间差的精确测量，以及异步脉冲生成。它的主要功能包括：

- 四个边沿输入触发源
- 每个边沿源的极性控制
- 边沿序列控制
- 控制对边沿的响应
- 1 ns 的时间测量分辨率
- 适合测量电容的精确电流源

CTMU 可与其他片内模拟模块一起使用，以精确地测量时间、电容以及电容的相对变化或生成不依赖于系统时钟的输出脉冲。CTMU 模块是与电容式传感器接口的理想选择。

CTMU 可通过两个寄存器进行控制：CTMUCON 和 CTMUICON。CTMUCON 用于使能模块和控制边沿源选择、边沿源极性选择以及边沿序列。CTMUICON 寄存器用于控制电流源的选择和调节。

图 25-1：电容测量的典型连接和内部配置



## 25.1 测量电容

CTMU 模块通过生成脉冲宽度和两个独立输入通道上的边沿事件间的时间宽度相等的输出脉冲，来测量电容。这两个输入通道上的脉冲边沿事件源可选择由以下 4 种源触发：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：

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

如需测量电容，在 CTMU 的脉冲输出信号之后，外部电容（CAPP）会对 A/D 转换器的其中一个输入通道进行采样。由第二个通道上的高精度电阻（RPR）对电流源进行校准。脉冲信号结束后，转换器确定电容上的电压。电容的实际计算在软件中由应用程序执行。

图 25-1 给出了电容测量使用的外部连接以及此应用中 CTMU 和 A/D 模块的关系。此示例还显示了来自 Timer1 的边沿事件，但使用外部边沿源的其他配置也是可能的。关于使用 CTMU 模块测量电容和时间的详细说明在《PIC24F 系列参考手册》中提供。

## 25.2 测量时间

对脉冲宽度的时间测量也可类似地执行，用 A/D 模块的内部电容（CAD）和高精度电阻校准电流。图 25-2 给出了时间测量使用的外部连接以及此应用中 CTMU 和 A/D 模块的关系。此示例还显示了来自外部 CTEDG 引脚的边沿事件，但使用内部边沿源的其他配置也是可能的。要实现最短时间测量，选择内部 A/D 通道 31，配置 CHOS<4:0>= 1 1 1 1 1。这可使输入引脚引入的寄生电容最小，而使总电容仅为 A/D 转换器本身的电容（4-5 pF）。关于使用 CTMU 模块测量电容和时间的详细说明在《PIC24F 系列参考手册》中提供。

## 25.3 脉冲生成和延时

CTMU 模块也可生成边沿和器件的系统时钟异步的输出脉冲。更明确地说，它可以生成边沿事件输入到该模块的延时可编程的脉冲。

当模块通过将 TGEN 位（CTMUCON<12>）置 1 配置为脉冲生成延时时，内部电流源连接到比较器 2 的 B 输入。将电容（CDELAY）连接到比较器 2 引脚 C2INB，且比较器参考电压 CVREF 连接到 C2INA。CVREF 随后被配置为特定跳变点。当检测到边沿事件时，模块开始对 CDELAY 充电。当 CDELAY 充电到超过 CVREF 跳变点时，在 CTPLS 上输出脉冲信号。脉冲延时的时间长度由 CDELAY 和 CVREF 跳变点的值决定。

图 25-3 给出了脉冲生成的外部连接，以及所需的不同模拟模块之间的关系。CTEDG1 显示为输入脉冲源时，其他选项可用。关于使用 CTMU 模块生成脉冲的详细说明在《PIC24F 系列参考手册》中提供。

图 25-2： 测量时间的典型连接和内部配置

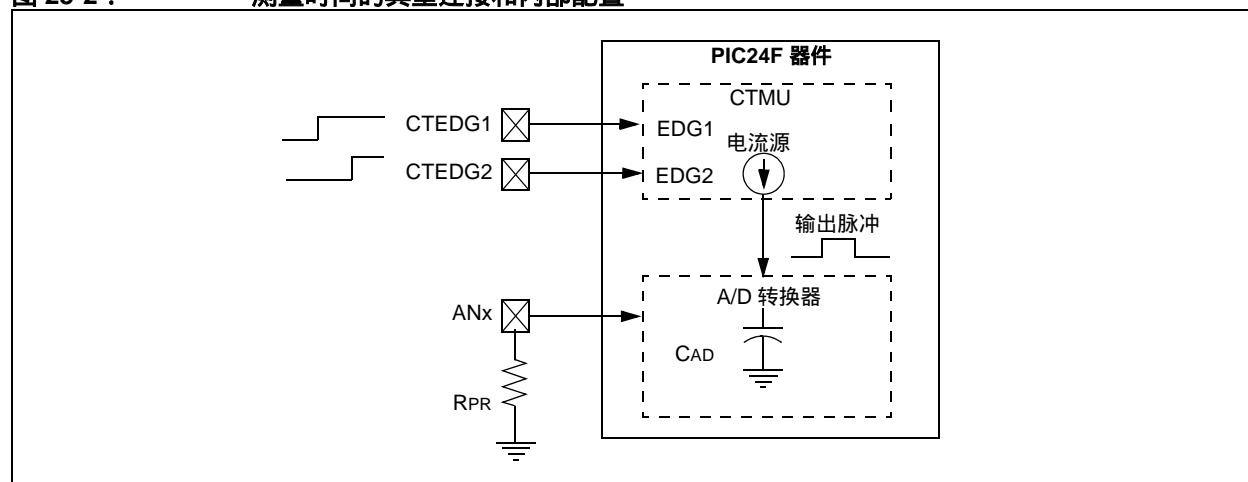
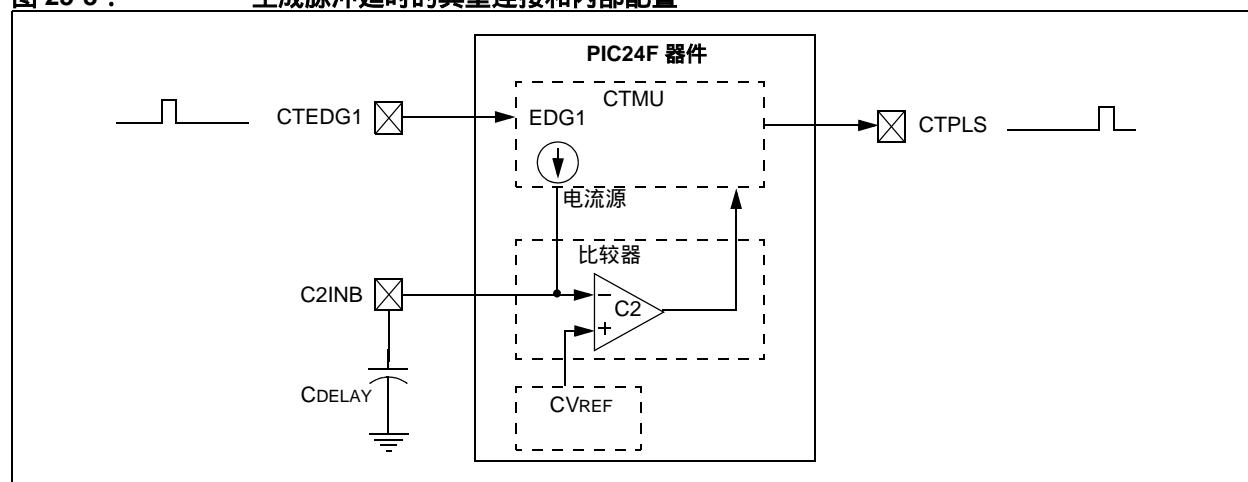


图 25-3： 生成脉冲延时的典型连接和内部配置



## 寄存器 25-1 : 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
CTMUEEN	—	CTMUSIDL	TGEN <sup>(1)</sup>	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 = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15      **CTMUEEN** : 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 = 模块在空闲模式下继续工作

bit 12      **TGEN** : 时间生成使能位<sup>(1)</sup>

1 = 使能生成边沿延时

0 = 禁止生成边沿延时

bit 11      **EDGEN** : 边沿使能位

1 = 不阻止边沿

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 事件之前发生

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 = 模拟电流源输出不接地

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 编程为正边沿响应

0 = 边沿 2 编程为负边沿响应

bit 6-5      **EDG2SEL<1:0>** : 边沿 2 源选择位

11 = CTED1 引脚

10 = CTED2 引脚

01 = OC1 模块

00 = Timer1 模块

bit 4      **EDG1POL** : 边沿 1 极性选择位

1 = 边沿 1 编程为正边沿响应

0 = 边沿 1 编程为负边沿响应

注 1 : 如果 TGEN = 1 , 外设的输入和输出必须配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

# PIC24FJ64GB004 系列

## 寄存器 25-1 : 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

**注 1 :** 如果 TGEN = 1 , 外设的输入和输出必须配置给可用的 RPn 引脚。更多信息 , 请参见第 10.4 节 “外设引脚选择 (PPS) ”。

## 寄存器 25-2 : 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 = 上电复位时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-10 **ITRIM<5:0>** : 电流源调节位

011111 = 标称电流的最大正向调整  
011110  
.....

000001 = 标称电流的最小正向调整  
000000 = IRNG<1:0> 指定的标称电流输出  
111111 = 标称电流的最小负向调整

.....  
100010  
100001 = 标称电流的最大负向调整

bit 9-8 **IRNG<1:0>** : 电流源范围选择位

11 = 100 倍基本电流  
10 = 10 倍基本电流  
01 = 基本电流级别 ( 标称值为 0.55 μA )  
00 = 禁止电流源

bit 7-0 **未实现** : 读为 0

## 26.0 特殊功能

<b>注：</b>	本数据手册总结了该组 PIC24F 器件的功能。但是不应把本数据手册当作无所不包的参考手册来使用。更多信息，请参见《PIC24F 系列参考手册》的以下章节：
	• 第 9 章 “看门狗定时器 (WDT)” ( DS39697A_CN )
	• 第 32 章 “高级器件集成” ( DS39719A_CN )
	• 第 33 章 “编程和诊断” ( DS39716A_CN )

PIC24FJ64GB004 系列器件具有几项特殊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。这些特殊功能包括：

- 灵活的配置
- 看门狗定时器 (WDT)
- 代码保护
- JTAG 边界扫描接口
- 在线串行编程
- 在线仿真

### 26.1 配置位

可以通过对配置位编程（读为 0）或不编程（读为 1）来选择不同的器件配置。这些配置位被映射到程序存储器中从 F80000h 开始的单元中。寄存器 26-1 到寄存器 26-6 详细说明了各配置位的不同功能。

注意地址 F80000h 超出了用户程序存储空间的范围。事实上，它属于配置存储空间（800000h-FFFFFFh），这一空间仅能通过表读和表写进行访问。

**表 26-1： PIC24FJ64GB004 系列器件的闪存配置字位置**

器件	配置字地址			
	1	2	3	4
PIC24FJ32GB00X	57FEh	57FCh	57FAh	57F8h
PIC24FJ64GB00X	ABFEh	ABFCh	ABFAh	ABF8h

#### 26.1.1 配置 PIC24FJ64GB004 系列器件的注意事项

在 PIC24FJ64GB004 系列器件中，配置字节以易失性存储方式实现。这就意味着在器件每次上电时都必须对配置数据进行编程。配置数据存储在片上程序存储空间顶部的 3 个字中，这些字被称为闪存配置字。表 26-1 给出了它们的具体位置。这些字是实际器件配置位的紧凑表现形式，这些配置位实际上散布在配置空间的几个单元中。器件复位时，配置数据会被自动从闪存配置字装入相应的配置寄存器中。

<b>注：</b>	所有类型的器件复位都会重新装载配置数据。
-----------	----------------------

当为这些器件创建应用程序时，用户应始终为配置数据分配特定的闪存配置字单元。此操作是为了确保在编译代码时不会把程序代码存储在该地址单元中。

程序存储器中的所有闪存配置字的高字节应始终为 1111 1111。这样当这些配置字被远程事件意外执行时，会被当作一条 NOP 指令。由于配置位并未真正保存在对应的单元内，因此向这些单元写 1 不会影响器件工作。

<b>注：</b>	在程序存储器末页中执行页擦除操作会清除闪存配置字，从而使能代码保护。因此，用户应避免在程序存储器末页中执行页擦除操作。
-----------	---

# PIC24FJ64GB004 系列

寄存器 26-1 : CW1 : 闪存配置字 1

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

r-x	R/PO-1	R/PO-1	R/PO-1	R/PO-1	U-1	R/PO-1	R/PO-1
r	JTAGEN <sup>(1)</sup>	GCP	GWRP	DEBUG	—	ICS1	ICS0
bit 15							bit 8

R/PO-1	R/PO-1	U-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1
FWDTEN	WINDIS	—	FWPSA	WDTPS3	WDTPS2	WDTPS1	WDTPS0
bit 7							bit 0

图注 :	r = 保留位
R = 可读位	PO = 一次编程位
-n = 器件未编程时的值	U = 未实现 , 读为 0 1 = 置 1 0 = 清零

- bit 23-16 未实现 : 读为 1
- bit 15 保留 : 该值未知 ; 编程为 0
- bit 14 **JTAGEN** : JTAG 端口使能位<sup>(1)</sup>  
1 = 使能 JTAG 端口  
0 = 禁止 JTAG 端口
- bit 13 **GCP** : 通用段程序存储器代码保护位  
1 = 禁止代码保护  
0 = 使能对整个程序存储空间的代码保护
- bit 12 **GWRP** : 通用段代码闪存写保护位  
1 = 允许写程序存储器  
0 = 禁止写程序存储器
- bit 11 **DEBUG** : 后台调试器使能位  
1 = 器件复位至正常工作模式  
0 = 器件复位至调试模式
- bit 10 未实现 : 读为 1
- bit 9-8 **ICS<1:0>** : 仿真器引脚位置选择位  
11 = 仿真器功能与 PGEC1/PGED1 复用  
10 = 仿真器功能与 PGEC2/PGED2 复用  
01 = 仿真器功能与 PGEC3/PGED3 复用  
00 = 保留 ; 不要使用
- bit 7 **FWDTEN** : 看门狗定时器使能位  
1 = 使能看门狗定时器  
0 = 禁止看门狗定时器
- bit 6 **WINDIS** : 窗口看门狗定时器禁止位  
1 = 使能标准看门狗定时器  
0 = 使能窗口看门狗定时器 ; FWDTEN 必须为 1
- bit 5 未实现 : 读为 1
- bit 4 **FWPSA** : WDT 预分频比选择位  
1 = 预分频比为 1:128  
0 = 预分频比为 1:32

注 1 : 只能使用在线串行编程 (ICSP™) 修改 JTAGEN 位。通过 JTAG 接口对器件编程时无法进行修改。

## 寄存器 26-1 : CW1 : 闪存配置字 1 (续)

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

注 1：只能使用在线串行编程 (ICSP™) 修改 JTAGEN 位。通过 JTAG 接口对器件编程时无法进行修改。

# PIC24FJ64GB004 系列

## 寄存器 26-2 : CW2 : 闪存配置字 2

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1
IESO	PLLDIV2	PLLDIV1	PLLDIV0	PLL96MHZ	FNOSC2	FNOSC1	FNOSC0
bit 15							bit 8

R/PO-1	R/PO-1	R/PO-1	R/PO-1	U-1	R/PO-1	R/PO-1	R/PO-1
FCKSM1	FCKSM0	OSCIOFCN	IOL1WAY	—	I2C1SEL	POSCMD1	POSCMD0
bit 7							bit 0

### 图注 :

R = 可读位

PO = 只可编程一次的位

U = 未实现 , 读为 0

-n = 器件未编程时的值

1 = 置 1

0 = 清零

bit 23-16

**未实现** : 读为 1

bit 15

**IESO** : 内部 / 外部切换位

1 = 使能 IESO 模式 ( 双速启动 )

0 = 禁止 IESO 模式 ( 双速启动 )

bit 14-12

**PLLDIV<2:0>** : USB 96 MHz PLL 预分频比选择位

111 = 振荡器输入的 12 分频 ( 48 MHz 输入 )

110 = 振荡器输入的 8 分频 ( 32 MHz 输入 )

101 = 振荡器输入的 6 分频 ( 24 MHz 输入 )

100 = 振荡器输入的 5 分频 ( 20 MHz 输入 )

011 = 振荡器输入的 4 分频 ( 16 MHz 输入 )

010 = 振荡器输入的 3 分频 ( 12 MHz 输入 )

001 = 振荡器输入的 2 分频 ( 8 MHz 输入 )

000 = 直接使用振荡器输入 ( 4 MHz 输入 )

bit 11

**PLL96MHZ** : USB 96 MHz PLL 启动使能位

1 = 启动时自动使能 96 MHz PLL

0 = 由用户通过软件使能 96 MHz PLL ( 使用 PLLEN 位 ( 即 CLKDIV<5> ) 进行控制 )

bit 10-8

**FNOSC<2:0>** : 初始振荡器选择位

111 = 带后分频器的快速 RC 振荡器 ( FRC DIV )

110 = 保留

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 模块的快速 RC 振荡器 ( FRCPLL )

000 = 快速 RC 振荡器 ( FRC )

bit 7-6

**FCKSM<1:0>** : 时钟切换和故障保护时钟监视器配置位

1x = 禁止时钟切换和故障保护时钟监视器

01 = 使能时钟切换 , 禁止故障保护时钟监视器

00 = 使能时钟切换和故障保护时钟监视器

bit 5

**OSCIOFCN** : OSCO 引脚配置位

如果 POSCMD<1:0> = 11 或 00 :

1 = OSCO/CLKO/RA3 用作 CLKO ( Fosc/2 )

0 = OSCO/CLKO/RA3 用作端口 I/O ( RC15 )

如果 POSCMD1:POSCMD0 = 10 或 01 :

OSCIOFCN 对 OSCO/CLKO/RA3 没有影响。

## 寄存器 26-2 : CW2 : 闪存配置字 2 (续)

bit 4	<b>IOL1WAY</b> : IOLOCK 单次置 1 使能位 1 = 解锁序列完成后，可将 IOLOCK 位 (OSCCON<6>) 置 1 一次。一旦置 1，就不能再次写入外设引脚选择寄存器。 0 = 解锁序列完成后，可根据需要将 IOLOCK 位置 1 或清零。
bit 3	<b>未实现</b> : 读为 1
bit 2	<b>I2C1SEL</b> : I2C1 引脚选择位 1 = 使用默认的 SCL1/SDA1 引脚 0 = 使用备用的 SCL1/SDA1 引脚
bit 1-0	<b>POSCMD&lt;1:0&gt;</b> : 主振荡器配置位 11 = 禁止主振荡器 10 = 选择 HS 振荡器模式 01 = 选择 XT 振荡器模式 00 = 选择 EC 振荡器模式

# PIC24FJ64GB004 系列

寄存器 26-3 : CW3 : 闪存配置字 3

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

R/PO-1	R/PO-1	R/PO-1	U-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1
WPEND	WPCFG	WPDIS	—	WUTSEL1	WUTSEL0	SOSCSEL1 <sup>(1)</sup>	SOSCSEL0 <sup>(1)</sup>
bit 15							bit 8

U-1	U-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1
—	—	WPFP5	WPFP4	WPFP3	WPFP2	WPFP1	WPFP0
bit 7							bit 0

**图注 :**

R = 可读位

PO = 只可编程一次的位

U = 未实现 , 读为 0

-n = 器件未编程时的值

1 = 置 1

0 = 清零

bit 23-16

**未实现** : 读为 1

bit 15

**WPEND** : 段写保护结束页选择位

1 = 受保护代码段下边界位于程序存储器的末尾 ( 000000h ) ; 上边界是 WPFP<8:0> 指定的代码页  
0 = 受保护代码段上边界是程序存储器的最后一页 ; 下边界是 WPFP<8:0> 指定的代码页

bit 14

**WPCFG** : 配置字代码页保护选择位

1 = 最后一页 ( 位于程序存储器开头 ) 和闪存配置字未受保护  
0 = 对最后一页和闪存配置字进行代码保护

bit 13

**WPDIS** : 段写保护禁止位

1 = 禁止段代码保护  
0 = 使能段代码保护 ; 受保护的段由 WPEND、 WPCFG 和 WPFPx 配置位指定

bit 12

**未实现** : 读为 1

bit 11-10

**WUTSEL<1:0>** : 稳压器待机模式唤醒时间选择位

11 = 使用默认稳压器启动时间  
01 = 使用快速稳压器启动时间  
x0 = 保留 ; 不要使用

bit 9-8

**SOSCSEL<1:0>** : 辅助振荡器电源模式选择位<sup>(1)</sup>

11 = SOSC 引脚工作在默认 ( 高驱动强度 ) 振荡器模式下  
01 = SOSC 引脚工作在低功耗 ( 低驱动强度 ) 振荡器模式下  
00 = SOSC 引脚具有数字 I/O 功能 ( RA4 和 RB4 ) ; 使用 SCLKI  
10 = 保留

bit 7-6

**未实现** : 读为 1

bit 5-0

**WPFP<5:0>** : 受保护代码段边界页位

指定受保护代码段的边界为从程序存储器结尾的页 9 开始的 512 条指令的页。

如果 WPEND = 1 :

指定代码页的最后一个地址是段的上边界。

如果 WPEND = 0 :

指定代码页的第一个地址是段的下边界。

**注 1 :** 仅当配置为数字 I/O 模式 ( 00 ) 时 , SOSCI 和 SOSCO 引脚上的数字功能才可用。

## 寄存器 26-4 : CW4 : 闪存配置字 4

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 23							bit 16

U-1	U-1	U-1	U-1	U-1	U-1	U-1	U-1
—	—	—	—	—	—	—	—
bit 15							bit 8

R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1	R/PO-1
DSWDTEN	DSBOREN	RTOSC	DSWDTOSC	DSWDTPS3	DSWDTPS2	DSWDTPS1	DSWDTPS0
bit 7							bit 0

### 图注 :

R = 可读位

PO = 只可编程一次的位

U = 未实现 , 读为 0

-n = 器件未编程时的值

1 = 置 1

0 = 清零

bit 23-8

**未实现** : 读为 1

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

**DSBOREN** : 深度休眠 BOR 使能位

1 = 深度休眠模式下使能 BOR

0 = 深度休眠模式下禁止 BOR ( 不影响休眠模式 )

bit 5

**RTOSC** : 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

**DSWDTOSC** : 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>** : DSWDT 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 )

1110 = 1:536,870,912 ( 6.4 天 )

1101 = 1:134,217,728 ( 38.5 小时 )

1100 = 1:33,554,432 ( 9.6 小时 )

1011 = 1:8,388,608 ( 2.4 小时 )

1010 = 1:2,097,152 ( 36 分钟 )

1001 = 1:524,288 ( 9 分钟 )

1000 = 1:131,072 ( 135 秒 )

0111 = 1:32,768 ( 34 秒 )

0110 = 1:8,192 ( 8.5 秒 )

0101 = 1:2,048 ( 2.1 秒 )

0100 = 1:512 ( 528 ms )

0011 = 1:128 ( 132 ms )

0010 = 1:32 ( 33 ms )

0001 = 1:8 ( 8.3 ms )

0000 = 1:2 ( 2.1 ms )

# PIC24FJ64GB004 系列

## 寄存器 26-5 : DEVID : 器件 ID 寄存器

U	U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—	—
bit 23								bit 16

R	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	R
DEV7	DEV6	DEV5	DEV4	DEV3	DEV2	DEV1	DEV0	
bit 7								bit 0

图注 : R = 只读位	U = 未实现位
--------------	----------

bit 23-16 未实现 : 读为 1

bit 15-8 FAMID<7:0> : 器件系列标识符位

01000010 = PIC24FJ64GB004 系列

bit 7-0 DEV<7:0> : 各个器件标识符位

00000011 = PIC24FJ32GB002

00000111 = PIC24FJ64GB002

00001011 = PIC24FJ32GB004

00001111 = PIC24FJ64GB004

## 寄存器 26-6 : DEVREV : 器件版本寄存器

U	U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—	—
bit 23								bit 16

U	U	U	U	U	U	U	U	U
—	—	—	—	—	—	—	—	—
bit 15								bit 8

U	U	U	U	R	R	R	R	R
—	—	—	—	REV3	REV2	REV1	REV0	
bit 7								bit 0

图注 : R = 只读位	U = 未实现位
--------------	----------

bit 23-4 未实现 : 读为 0

bit 3-0 REV<3:0> : 次要版本标识符位

对器件的版本号编码 (仅序列号 ; 无主 / 次字段)。

## 26.2 片内稳压器

所有 PIC24FJ64GB004 系列器件使用标称值为 2.5V 的电源为其内核数字逻辑供电。对于需要工作在一个更高的典型电压值（如 3.3V）的设计来讲，这可能会带来问题。为简化系统设计，PIC24FJ64GB004 系列中的所有器件均包含一个片内稳压器，可使器件内核逻辑工作在 VDD 下。

DISVREG 引脚控制该稳压器。把 VSS 连到该引脚将使能稳压器，然后稳压器通过其他 VDD 引脚向内核供电。当使能稳压器时，必须在 VDDCORE/VCAP 引脚连接低 ESR 电容（如陶瓷电容）（见图 26-1）。这有利于保持稳压器的稳定性。第 29.1 节“直流特性”中提供了该滤波电容的推荐值（CEFC）。

如果 DISVREG 与 VDD 连接，则禁止稳压器。在这种情况下，必须通过 VDDCORE/VCAP 引脚对器件内核逻辑单独提供标称值为 2.5V 的电压，从而使 I/O 引脚可以具有较高的电压（通常为 3.3V）。另外，VDDCORE/VCAP 和 VDD 引脚可以连在一起，使器件工作在较低的标称电压下。请参见图 26-1 了解可能的配置。

### 26.2.1 稳压器跟踪模式和低电压检测

当使能时，片内稳压器为数字内核逻辑提供标称值为 2.5V 的恒定电压。

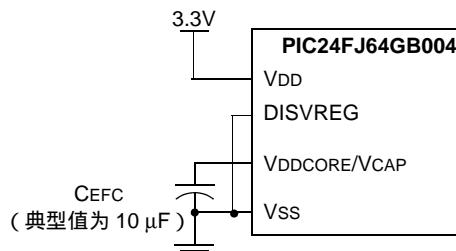
稳压器可提供从约为 2.5V 直至器件所能承受的最高电压 VDDMAX 的 VDD 电压范围。该稳压器无法将低于 2.5V 的 VDD 电压提高。为防止用于稳压器的电压降得过低时产生“欠压”条件，稳压器进入跟踪模式。在跟踪模式下，稳压器输出跟随 VDD，通常比 Vdd 小 100 mV。

器件进入跟踪模式后，不可能再继续以全速工作。为了确定器件何时进入跟踪模式的信息，片内稳压器中包含了一个简单的低电压检测电路。当 VDD 降低到全速工作电压以下时，该电路将低电压检测中断标志位 LVDIF (IFS4<8>) 置 1。这可被用于产生中断并使应用进入到低功耗工作模式，或触发正常关闭。

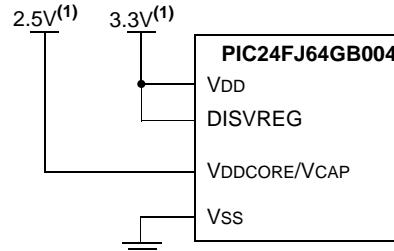
低电压检测仅在稳压器使能的情况下可用。

图 26-1：片内稳压器的连接

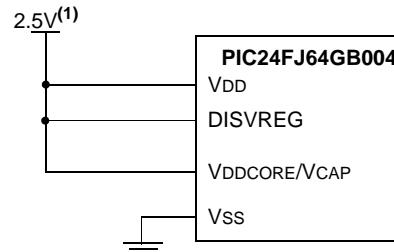
使能稳压器 (DISVREG 连接到 VSS)：



禁止稳压器 (DISVREG 连接到 VDD)：



禁止稳压器 (VDD 连接到 VDDCORE)：



注 1：这些为典型的工作电压值。请参见第 29.1 节“直流特性”了解 VDD 和 VDDCORE 的全部工作范围。

### 26.2.2 片内稳压器和 POR

使能稳压器后，需等待大约 10 μs 才能产生输出。在这段称为 TPM 的时间内，禁止代码执行。每次器件在掉电（包括休眠模式）后恢复工作时都需要 TPM。TPM 由 PMSLP 位 (RCON<8>) 和 WUTSEL 配置位 (CW3<11:10>) 的设置决定。

注：更多关于 TPM 的信息，请参见第 29.0 节“电气特性”。

如果禁止稳压器，将自动使能独立的上电延时定时器 (PWRT)。在器件启动 (仅 POR 或 BOR) 时，PWRT 会增加一段固定的 64 ms (标称值) 的延时。

在禁止稳压器的情况下从休眠模式唤醒时，器件使用 TPM 确定唤醒时间。若要减少器件在禁止稳压器的情况下唤醒时间，请将 PMSLP 位置 1。

## 26.2.3 片内稳压器和 BOR

当使能片内稳压器时，PIC24FJ64GB004 系列器件还具有简单的欠压复位功能。如果向稳压器提供的电压不足以维持跟踪电平，那么稳压器复位电路将产生欠压复位。BOR 标志位 (RCON<1>) 将捕捉该事件。《PIC24F 系列参考手册》的第 7 章“复位”(DS39712A\_CN) 中提供了欠压电压规范。

## 26.2.4 上电要求

片内稳压器是为了满足器件的上电要求而设计的。如果应用不使用稳压器，那就必须严格遵守上电条件。在上电时，VDDCORE 决不能超出 VDD 电平 0.3V 以上。

**注：** 更多信息，请参见第 29.0 节“电气特性”。

## 26.2.5 稳压器待机模式

当使能片内稳压器时，它除消耗 IDD/IPD 外，还总是会额外消耗一个大的电流，器件工作在休眠模式下也是如此，尽管此时内核数字逻辑并不需要耗能。为了在供电紧张的应用中节省更多的功耗，可设置稳压器在器件进入休眠模式时通过给闪存断电自动进入待机模式。PMSLP (RCON<8>) 位控制此特性。默认情况下，该位清零，使能待机模式。

对于 PIC24FJ64GB004 系列器件，WUTSEL<1:0> 配置位 (CW3<11:10>) 控制稳压器从待机模式唤醒所需的时间。所有器件的默认唤醒时间是 190 μs，这是为匹配旧的 PIC24F 器件唤醒时间而提供的传统模式。

WUTSEL 配置位的实现提供了一个快速唤醒选项。当 WUTSEL<1:0> = 01 时，稳压器唤醒时间为 TPM，即 10 μs。

当关闭稳压器待机模式 (PMSLP = 1) 时，闪存程序存储器在休眠模式下处于上电状态。这使得器件无需等待 TPM 时间即可唤醒。但是，如果 PMSLP 置 1，休眠模式下消耗的电流约比允许稳压器进入待机模式时高 40 μA。

## 26.3 看门狗定时器 (WDT)

PIC24FJ64GB004 系列器件的 WDT 是由 LPRC 振荡器驱动的。当使能 WDT 时，也将同时使能该时钟源。

由 LPRC 提供的 WDT 时钟源的频率标称值为 31 kHz。将此时钟源提供给可配置为 5 位 (32 分频) 或 7 位 (128 分频) 工作模式的预分频器。分频比由 FWPSA 配置位设置。31 kHz 的输入使预分频器生成标称的 WDT 超时周期 (TWDT) —— 5 位模式为 1 ms，7 位模式为 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，从而获得更大范围的超时周期。后分频比由 WDTPS<3:0> 配置位 (CW1<3:0>) 控制，该配置位共允许选择 16 种设置，从 1:1 到 1:32,768。使用预分频器和后分频器后，可获得 1 ms 到 131 秒的超时周期。

WDT、预分频器和后分频器在以下条件下复位：

- 任何类型的器件复位
- 在时钟切换完成时，无论时钟切换是由软件（即改变 NOSC 位后将 OSWEN 位置 1）或硬件（即故障保护时钟监视器）引起
- 执行 PWRSAV 指令时（即进入休眠模式或空闲模式）
- 当器件退出休眠模式或空闲模式恢复正常工作时
- 在正常执行过程中，执行 CLRWDT 指令

如果使能 WDT，它将在休眠或空闲模式下继续运行。当发生 WDT 超时时，将唤醒器件并且代码将从 PWRSAV 指令处继续执行。器件被唤醒后，需要用软件将相应的 SLEEP 或 IDLE 位 (RCON<3:2>) 清零。

WDT 标志位 WTO (RCON<4>) 不会在 WDT 超时后自动清零。要检测后续的 WDT 事件，必须用软件将该标志清零。

**注：** 当执行 CLRWDT 和 PWRSAV 指令时，预分频器和后分频器的计数值将被清零。

### 26.3.1 窗口操作

看门狗定时器具有可选的固定窗口工作模式。在此窗口模式下，CLRWDAT 指令只能在编程的 WDT 周期的后 1/4 周期复位 WDT。在该窗口前执行的 CLRWDAT 指令会导致 WDT 复位，这与 WDT 超时类似。

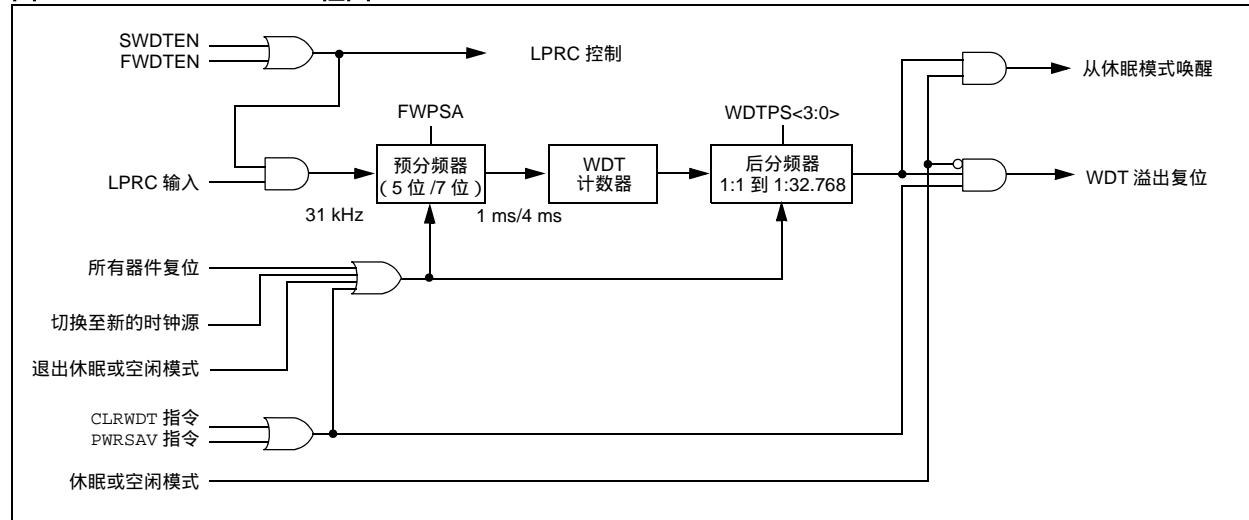
将 WINDIS 配置位 (CW1<6>) 编程为 0 使能 WDT 窗口模式。

### 26.3.2 控制寄存器

通过 FWDTEN 配置位使能或禁止 WDT。当 FWDTEN 配置位置 1 时，始终使能 WDT。

FWDTEN 配置位被编程为 0 后，也可以使用软件来控制 WDT。通过在软件中将 SWDTEN 控制位 (RCON<5>) 置 1 使能 WDT。任何类型的器件复位都会使 SWDTEN 控制位清零。软件 WDT 选项允许用户在关键代码段使能 WDT 并在非关键代码段禁止 WDT，以最大限度地降低功耗。

**图 26-2：WDT 框图**



### 26.4 深度休眠看门狗定时器 (DSWDT)

PIC24FJ64GB004 系列器件具有 WDT 模块和 DSWDT 模块。后者被使能后可在器件处于深度休眠模式时运行，并由 SOSC 或 LPRC 振荡器驱动。通过 DSWDTONOSC 配置位 (CW4<4>) 选择 DSWDT 的时钟源。

可通过配置位 DSWDTPS<3:0> (CW4<3:0>) 选择相应的后分频比，将 DSWDT 配置为产生 2.1 ms 至 25.7 天的延时。当使能 DSWDT 时，也将同时使能该时钟源。DSWDT 是可以将器件从深度休眠模式中唤醒的唤醒源之一。

### 26.5 程序校验和代码保护

PIC24FJ64GB004 系列器件提供了两种补充方法防止应用代码被改写和擦除。这两种方法也有助于在运行时防止器件配置被意外更改。

#### 26.5.1 通用段保护

PIC24FJ64GB004 系列中所有器件的片内程序存储空间被视作一个存储区，即通用段 (General Segment, GS)。配置位 GCP 控制该存储区的代码保护。该位阻止外部对程序存储空间的读写操作。但对正常的执行模式没有直接影响。

写保护是由配置字中的 GWRP 位控制的。当 GWRP 被编程为 0 时，还禁止在内部对程序存储器执行写和擦除操作。

## 26.5.2 代码段保护

除了全局通用段保护外，程序存储空间的独立子范围可以单独地被保护以防止写和擦除操作。该区域可在需要写和擦除受保护代码的单独存储块中有多种用途，例如自举程序。与常见的引导区实现不同，PIC24FJ64GB004 系列器件中特定的受保护段可被用户放在程序空间的任何位置并配置为各种大小。

代码段保护还增加了对程序存储器的指定区域的新保护级别，这是通过在写或擦除地址发生在指定范围内时禁止 NVM 安全互锁实现的。它不会改写 GCP 或 GWRP 位控制的通用段保护。例如，如果使能了 GCP 和 GWRP，则使能程序存储器下半部分的段代码保护不会撤销对上半部分的通用段保护。

段代码保护范围的大小和类型由配置字 3 的 WPFPx、WPEND、WPCFG 和 WPDIS 位配置。将 WPDIS 位编程为 0 使能代码段保护。WPFP 位通过指定 512 字代码页（即受保护段的开始或结束位置）指定受保护的段大小。因为包含指定的存储区，所以，此页也将受到保护。

WPEND 位决定受保护段是否使用程序空间的开头或末尾作为边界。将 WPEND 编程为 0 可设置程序存储器的末尾 (000000h) 作为受保护段的下边界。保留 WPEND 未编程（即 = 1）将保护指定页到已实现程序存储器的最后一页（包含配置字单元）。

独立位 WPCFG 用于单独保护程序空间的最后一页（包含闪存配置字）。将 WPCFG 编程为 0 可保护最后一页（不管其他位的设置如何）。这可用在需要对存储器末尾的代码段以及闪存配置字进行写保护的情况下。

表 26-2 中给出了段代码保护的各种选项。

## 26.5.3 保护配置寄存器

有两种方法保护配置寄存器使其免遭意外的或不想要的更改或读取。主要的保护方法与保护 RP 寄存器的方法相同——影子寄存器中包含了一个基准值，持续将该值与实际的值进行比较。

出于从防范不可预见事件方面的考虑，由于电池故障（如 ESD 事件）引起的配置位更改将导致奇偶校验错误并触发器件复位。

配置寄存器的数据来自于程序存储器中的闪存配置字。当 GCP 位置 1 时，也将保护器件配置的源数据。即使未使能通用段保护，使用适当的代码段保护设置也可以保护器件配置。

表 26-2：段代码保护配置选项

段配置位			代码段的写 / 擦除保护
WPDIS	WPEND	WPCFG	
1	x	1	未使能其他的保护；所有程序存储器保护都由 GCP 和 GWRP 配置
1	x	0	保护最后一个代码页（包含闪存配置字）
0	1	0	保护范围从 WPFP<5:0> 定义的代码页的第一个地址到已实现程序存储器的末尾（包含末尾），包含闪存配置字
0	0	0	保护范围为地址 000000h 到 WPFP<5:0> 定义的代码页的最后一个地址（包含该地址）
0	1	1	保护范围从 WPFP<5:0> 定义的代码页的第一个地址到已实现程序存储器的末尾（包含末尾），包含闪存配置字
0	0	1	保护范围从 WPFP<5:0> 定义的代码页的第一个地址到已实现程序存储器的末尾（包含末尾）

## 26.6 JTAG 接口

PIC24FJ64GB004 系列器件实现了 JTAG 接口，该接口支持边界扫描器件测试。

## 26.7 在线串行编程

可以在最终应用电路中对 PIC24FJ64GB004 系列单片机进行串行编程。只需要 5 根线即可完成这一操作，其中时钟线 (PGECx) 和数据线 (PGEDx) 各一根，其余 3 根分别是电源线、接地线和编程电压线。这允许用户使用未编程器件制造电路板，仅在产品交付前才对单片机进行编程，从而可以将最新版本的固件或定制固件烧写到单片机中。

## 26.8 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。这一功能允许结合 MPLAB IDE 进行一些简单的调试。通过 PGECx (仿真 / 调试时钟) 和 PGEDx (仿真 / 调试数据) 引脚控制调试功能。

要使用器件的在线调试功能，在设计中必须实现至 MCLR、VDD、Vss 和 ICS 配置位指定的 PGECx/PGEDx 引脚对的 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 个字节和两个 I/O 引脚。

# **PIC24FJ64GB004 系列**

---

---

注：

## 27.0 开发支持

一系列软件及硬件开发工具对 PIC<sup>®</sup> 单片机和 dsPIC<sup>®</sup> 数字信号控制器提供支持：

- 集成开发环境
  - MPLAB<sup>®</sup> IDE 软件
- 编译器 / 汇编器 / 链接器
  - 适用于各种器件系列的 MPLAB C 编译器
  - 适用于各种器件系列的 HI-TECH C 编译器
  - MPASM<sup>™</sup> 汇编器
  - MPLINK<sup>™</sup> 目标链接器 /  
MPLIB<sup>™</sup> 目标库管理器
  - 适用于各种器件系列的 MPLAB 汇编器 / 链接器 / 库管理器
- 模拟器
  - MPLAB SIM 软件模拟器
- 仿真器
  - MPLAB REAL ICE<sup>™</sup> 在线仿真器
- 在线调试器
  - MPLAB ICD 3
  - PICkit<sup>™</sup> 3 Debug Express
- 器件编程器
  - PICkit<sup>™</sup> 2 编程器
  - MPLAB PM3 器件编程器
- 低成本演示 / 开发板、评估工具包及入门工具包

## 27.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16/32 位单片机市场提供了前所未有的易于使用的软件开发平台。MPLAB IDE 是基于 Windows<sup>®</sup> 操作系统的应用软件，包括：

- 一个包含所有调试工具的图形界面
    - 模拟器
    - 编程器（单独销售）
    - 在线仿真器（单独销售）
    - 在线调试器（单独销售）
  - 具有彩色上下文代码显示的全功能编辑器
  - 多项目管理器
  - 内容可直接编辑的可定制式数据窗口
  - 高级源代码调试
  - 鼠标停留在变量上进行查看的功能
  - 将变量从源代码窗口拖放到 Watch（观察）窗口
  - 丰富的在线帮助
  - 集成了可选的第三方工具，如 IAR C 编译器
- MPLAB IDE 可以让您：
- 编辑源文件（C 语言或汇编语言）
  - 点击一次即可完成编译或汇编，并将代码下载到仿真器和模拟器工具中（自动更新所有项目信息）
  - 可使用如下各项进行调试：
    - 源文件（C 语言或汇编语言）
    - 混合 C 语言和汇编语言
    - 机器码

MPLAB IDE 在单个开发范例中支持使用多种调试工具，包括从成本效益高的模拟器到低成本的在线调试器，再到全功能的仿真器。这样缩短了用户升级到更加灵活而功能强大的工具时的学习时间。

## 27.2 适用于各种器件系列的 MPLAB C 编译器

MPLAB C 编译器代码开发系统是完全的 ANSI C 编译器，适用于 Microchip 的 PIC18、PIC24 和 PIC32 系列单片机及 dsPIC30 和 dsPIC33 系列数字信号控制器。这些编译器提供强大的集成功能和出众的代码优化能力，且使用方便。

为便于源代码调试，编译器提供针对 MPLAB IDE 调试器优化的符号信息。

## 27.3 适用于各种器件系列的 HI-TECH C 编译器

HI-TECH C 编译器代码开发系统是完全的 ANSI C 编译器，适用于 Microchip 的 PIC 系列单片机及 dsPIC 系列数字信号控制器。这些编译器提供强大的集成功能和全知代码生成能力，且使用方便。

为便于源代码调试，编译器提供针对 MPLAB IDE 调试器优化的符号信息。

编译器包括一个宏汇编器、链接器、预处理程序和单步驱动程序，可以在多种平台上运行。

## 27.4 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于 PIC10/12/16/18 MCU。

MPASM 汇编器可生成用于 MPLINK 目标链接器的可重定位目标文件、Intel® 标准 HEX 文件、详细描述存储器使用状况和符号参考的 MAP 文件、包含源代码行及生成机器码的绝对 LST 文件以及用于调试的 COFF 文件。

MPASM 汇编器具有如下特性：

- 集成在 MPLAB IDE 项目中
- 用户定义的宏可简化汇编代码
- 对多用途源文件进行条件汇编
- 允许完全控制汇编过程的指令

## 27.5 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器包含了由 MPASM 汇编器、MPLAB C18 C 编译器产生的可重定位目标。通过使用链接器脚本中的指令，它还可链接预编译库中的可重定位目标。

MPLIB 目标库管理器管理预编译代码库文件的创建和修改。当从源文件调用库中的一段子程序时，只有包含此子程序的模块被链接到应用程序。这样可使大型库在许多不同应用中被高效地利用。

目标链接器 / 库管理器具有如下特性：

- 高效地连接单个的库而不是许多小文件
- 通过将相关的模块组合在一起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

## 27.6 适用于各种器件系列的 MPLAB 汇编器、链接器和库管理器

MPLAB 汇编器为 PIC24、PIC32 和 dsPIC 器件从符号汇编语言生成可重定位机器码。MPLAB C 编译器使用该汇编器生成目标文件。汇编器产生可重定位目标文件之后，可将这些目标文件存档，或与其他可重定位目标文件和存档链接以生成可执行文件。该汇编器有如下显著特性：

- 支持整个器件指令集
- 支持定点数据和浮点数据
- 命令行界面
- 丰富的指令集
- 灵活的宏语言
- MPLAB IDE 兼容性

## 27.7 MPLAB SIM 软件模拟器

MPLAB SIM 软件模拟器通过在指令级对 PIC MCU 和 dsPIC® DSC 进行模拟，可在 PC 主机环境下进行代码开发。对于任何给定的指令，都可以对数据区进行检查或修改，并通过一个全面的激励控制器来施加激励。可以将各寄存器记录在文件中，以便进行进一步的运行时分析。跟踪缓冲区和逻辑分析器的显示使软件模拟器还能记录和跟踪程序的执行、I/O 的动作、大部分的外设及内部寄存器。

MPLAB SIM 软件模拟器完全支持使用 MPLAB C 编译器以及 MPASM 和 MPLAB 汇编器的符号调试。该软件模拟器可用于在硬件实验室环境外灵活地开发和调试代码，是一款完美且经济的软件开发工具。

## 27.8 MPLAB REAL ICE 在线仿真器系统

MPLAB REAL ICE 在线仿真器系统是 Microchip 针对其闪存 DSC 和 MCU 器件而推出的新一代高速仿真器。结合 MPLAB 集成开发环境（IDE）所具有的易于使用且功能强大的图形用户界面，该仿真器可对 PIC® 闪存 MCU 和 dsPIC® 闪存 DSC 进行调试和编程。IDE 是随每个工具包一起提供的。

该仿真器通过高速 USB 2.0 接口与设计工程师的 PC 相连，并利用与在线调试器系统兼容的连接器（RJ11）或新型抗噪声、高速低压差分信号（LVDS）互连电缆（CAT5）与目标板相连。

可通过 MPLAB IDE 下载将来版本的固件，对该仿真器进行现场升级。在即将推出的 MPLAB IDE 版本中，会支持许多新器件，还将增加一些新特性。在同类仿真器中，MPLAB REAL ICE 的优势十分明显：低成本、全速仿真、运行时变量查看、跟踪分析、复杂断点、耐用的探针接口及较长（长达 3 米）的互连电缆。

## 27.9 MPLAB ICD 3 在线调试器系统

MPLAB ICD 3 在线调试器系统是 Microchip 成本效益最高的高速硬件调试器 / 编程器，适用于 Microchip 闪存数字信号控制器（DSC）和单片机（MCU）器件。结合 MPLAB 集成开发环境（IDE）所具有的功能强大但易于使用的图形用户界面，该调试器可对 PIC® 闪存单片机和 dsPIC® DSC 进行调试和编程。

MPLAB ICD 3 在线调试器通过高速 USB 2.0 接口与设计工程师的 PC 相连，并利用与 MPLAB ICD 2 或 MPLAB REAL ICE 系统兼容的连接器（RJ-11）与目标板相连。MPLAB ICD 3 支持所有 MPLAB ICD 2 转接器。

## 27.10 PICkit 3 在线调试器 / 编程器及 PICkit 3 Debug Express

结合 MPLAB 集成开发环境（IDE）所具有的功能强大的图形用户界面，MPLAB PICkit 3 可对 PIC® 闪存单片机和 dsPIC® 数字信号控制器进行调试和编程，且价位较低。MPLAB PICkit 3 通过全速 USB 接口与设计工程师的 PC 相连，并利用 Microchip 调试（RJ-11）连接器（与 MPLAB ICD 3 和 MPLAB REAL ICE 兼容）与目标板相连。连接器使用两个器件 I/O 引脚和复位线来实现在线调试和在线串行编程。

PICkit 3 Debug Express 包括 PICkit 3、演示板和单片机、连接电缆和光盘（内含用户指南、课程、教程、编译器和 MPLAB IDE 软件）。

# PIC24FJ64GB004 系列

## 27.11 PICkit 2 开发编程器 / 调试器及 PICkit 2 Debug Express

PICkit™ 2 开发编程器 / 调试器是一款低成本开发工具，具有易于使用的界面，适用于对 Microchip 的闪存系列单片机进行编程和调试。这一全功能的 Windows® 编程界面支持低档 (PIC10F、PIC12F5xx 和 PIC16F5xx)、中档 (PIC12F6xx 和 PIC16F)、PIC18F、PIC24、dsPIC30、dsPIC33 和 PIC32 系列的 8 位、16 位及 32 位单片机，以及许多 Microchip 串行 EEPROM 产品。结合 Microchip 功能强大的 MPLAB 集成开发环境 (IDE)，PICkit 2 可对大多数 PIC® 单片机进行在线调试。即使 PIC 单片机已嵌入应用，在线调试功能仍可以运行、暂停和单步执行程序。在断点处暂停时，可以检查和修改文件寄存器。

PICkit 2 Debug Express 包括 PICkit 2、演示板和单片机、连接电缆和光盘（内含用户指南、课程、教程、编译器和 MPLAB IDE 软件）。

## 27.12 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款符合 CE 规范的通用器件编程器，在 VDDMIN 和 VDDMAX 点对其可编程电压进行校验以确保可靠性最高。它有一个用来显示菜单和错误消息的大 LCD 显示器 (128 x 64)，以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 ICSPTM 电缆。在单机模式下，MPLAB PM3 器件编程器不必与 PC 相连即可对 PIC 器件进行读取、校验和编程。在该模式下它还可设置代码保护。MPLAB PM3 通过 RS-232 或 USB 电缆连接到 PC 主机上。MPLAB PM3 具备高速通信能力以及优化算法，可对具有大存储器的器件进行快速编程。它还包含了 MMC 卡，用于文件存储及数据应用。

## 27.13 演示 / 开发板、评估工具包及入门工具包

有许多演示、开发和评估板可用于各种 PIC MCU 和 dsPIC DSC，实现对全功能系统的快速应用开发。大多数的演示、开发和评估板都有实验布线区，供用户添加定制电路；还有应用固件和源代码，用于检查和修改。

这些板支持多种功能部件，包括 LED、温度传感器、开关、扬声器、RS-232 接口、LCD 显示器、电位计和附加 EEPROM 存储器。

演示和开发板可用于教学环境，在实验布线区设计定制电路，从而掌握各种单片机应用。

除了 PICDEM™ 和 dsPICDEM™ 演示 / 开发板系列电路外，Microchip 还有一系列评估工具包和演示软件，适用于模拟滤波器设计、KEELOQ® 数据安全产品 IC、CAN、IrDA®、PowerSmart 电池管理、SEEVAL® 评估系统、Σ-Δ ADC、流速传感器，等等。

同时还提供入门工具包，其中包含体验指定器件功能所需的所有软硬件。通常提供单个应用以及调试功能，都包含在一块电路板上。

有关演示、开发和评估工具包的完整列表，请访问 Microchip 网站 ([www.microchip.com](http://www.microchip.com))。

## 28.0 指令集汇总

**注：** 本章是 PIC24F 指令集架构的简要汇总，并不能作为详尽的参考资料使用。

PIC24F 指令集与以前的 PIC® MCU 指令集相比，添加了许多增强功能，并保持了易于从以前的 PIC MCU 指令集移植的特点。大部分指令只占用一个程序存储字。只有 3 条指令需要两个程序存储单元。

每条单字指令都是一个 24 位字，由一个 8 位的操作码（指明指令类型）和一个或多个操作数（指定指令具体操作）组成。整个指令集具有高度的正交性，分为以下 4 种基本类型：

- 面向字或字节的操作类指令
- 面向位的操作类指令
- 立即数操作类指令
- 控制操作类指令

表28-1给出了在说明指令时要用到的通用符号。表28-2 中的 PIC24F 指令集汇总列出了所有指令及每条指令影响的状态标志。

大部分面向字或字节的 W 寄存器指令（包含桶形移位寄存器指令）含有三个操作数：

- 第一个源操作数通常是不带地址修改符的“Wb”寄存器
- 第二个源操作数通常是带或不带地址修改符的“Ws”寄存器
- 结果的目标地址通常是带或不带地址修改符的“Wd”寄存器

而面向字或字节的文件寄存器指令具有两个操作数：

- 文件寄存器（由“f”值指定）
- 目标寄存器（可以是文件寄存器“f”，也可以是记作“WREG”的W0寄存器）

大部分面向位的操作类指令（包括简单循环/移位指令）具有两个操作数：

- W 寄存器（带或不带地址修改符）或文件寄存器（由“Ws”或“f”的值指定）
- W 寄存器或文件寄存器中的位（由立即数直接指定或由“Wb”寄存器中的内容间接指定）

涉及数据传送的立即数指令可能使用以下操作数：

- 将被装载到 W 寄存器或文件寄存器的立即数值（由“k”的值指定）
- 将要装载立即数值的 W 寄存器或文件寄存器（由“Wb”或“f”指定）

而涉及算术或逻辑运算的立即数指令使用以下操作数：

- 第一个源操作数是不带地址修改符的“Wb”寄存器
- 第二个源操作数是立即数
- 结果的目标地址（只有在与第一个源操作数不同的情况下）通常为带或不带地址修改符的“Wd”寄存器

控制操作类指令使用以下操作数：

- 程序存储器地址
- 表读和表写指令的模式

除某些双字指令外所有指令都是单字指令。双字指令中所有必需的信息都在这 48 位中，第二个字的高 8 位全为 0。如果第二个字作为一条指令（由自身）执行，它将会被当作 NOP 指令执行。

除非条件测试结果为 true 或者指令执行后改变了程序计数器的值，否则执行所有的单字指令都只需要一个指令周期。对于上述两种特殊情况，指令执行需要两个指令周期，第二个指令周期执行一条 NOP 指令。值得注意的特殊指令有：BRA（无条件 / 计算转移指令）、间接 CALL/GOTO 指令、所有表读和表写指令以及 RETURN/RETFIE 指令，这些指令都是单字指令，但执行起来需要 2 或 3 个指令周期。

某些涉及跳过后续指令的指令，在执行跳过时需要两或三个指令周期，具体周期数取决于被跳过的指令是单字指令还是双字指令。此外需要传送两个字的指令需要两个周期。执行双字指令需要两个指令周期。

# PIC24FJ64GB004 系列

表 28-1：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“地址为text的单元”
{ }	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（默认情况）
bit4	4位位选择字段（用于字寻址指令） $\in \{0\dots15\}$
C、DC、N、OV 和 Z	MCU 状态位：进位、半进位、负标志、溢出标志和全零标志
Expr	绝对地址、标号或表达式（由链接器解析）
f	文件寄存器地址 $\in \{0000h\dots1FFFh\}$
lit1	1位无符号立即数 $\in \{0,1\}$
lit4	4位无符号立即数 $\in \{0\dots15\}$
lit5	5位无符号立即数 $\in \{0\dots31\}$
lit8	8位无符号立即数 $\in \{0\dots255\}$
lit10	10位无符号立即数，字节模式下， $\in \{0\dots255\}$ ，字模式下， $\in \{0:1023\}$
lit14	14位无符号立即数 $\in \{0\dots16383\}$
lit16	16位无符号立即数 $\in \{0\dots65535\}$
lit23	23位无符号立即数 $\in \{0\dots8388607\}$ ；LSB 必须为 0
None	该字段不必有输入项，可以为空白
PC	程序计数器
Slit10	10位有符号立即数 $\in \{-512\dots511\}$
Slit16	16位有符号立即数 $\in \{-32768\dots32767\}$
Slit6	6位有符号立即数 $\in \{-16\dots16\}$
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 28-2： 指令集概述

汇编指令 助记符	汇编语法	说明	字数	周期数	受影响的 状态标志
ADD	ADD f	f = f + WREG	1	1	C、DC、N、OV 和 Z
	ADD f, WREG	WREG = f + WREG	1	1	C、DC、N、OV 和 Z
	ADD #lit10, Wn	Wd = lit10 + Wd	1	1	C、DC、N、OV 和 Z
	ADD Wb, Ws, Wd	Wd = Wb + Ws	1	1	C、DC、N、OV 和 Z
	ADD Wb, #lit5, Wd	Wd = Wb + lit5	1	1	C、DC、N、OV 和 Z
ADDC	ADDC f	f = f + WREG + (C)	1	1	C、DC、N、OV 和 Z
	ADDC f, WREG	WREG = f + WREG + (C)	1	1	C、DC、N、OV 和 Z
	ADDC #lit10, Wn	Wd = lit10 + Wd + (C)	1	1	C、DC、N、OV 和 Z
	ADDC Wb, Ws, Wd	Wd = Wb + Ws + (C)	1	1	C、DC、N、OV 和 Z
	ADDC Wb, #lit5, Wd	Wd = Wb + lit5 + (C)	1	1	C、DC、N、OV 和 Z
AND	AND f	f = f .AND. WREG	1	1	N 和 Z
	AND f, WREG	WREG = f .AND. WREG	1	1	N 和 Z
	AND #lit10, Wn	Wd = lit10 .AND. Wd	1	1	N 和 Z
	AND Wb, Ws, Wd	Wd = Wb .AND. Ws	1	1	N 和 Z
	AND Wb, #lit5, Wd	Wd = Wb .AND. lit5	1	1	N 和 Z
ASR	ASR f	f = 算术右移 f	1	1	C、N、OV 和 Z
	ASR f, WREG	WREG = 算术右移 f	1	1	C、N、OV 和 Z
	ASR Ws, Wd	Wd = 算术右移 Ws	1	1	C、N、OV 和 Z
	ASR Wb, Wns, Wnd	Wnd = 将 Wb 算术右移 Wns 位	1	1	N 和 Z
	ASR Wb, #lit5, Wnd	Wnd = 将 Wb 算术右移 lit5 位	1	1	N 和 Z
BCLR	BCLR f, #bit4	将 f 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	如果进位位为 1 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	如果进位位为 0 则转移	1	1 (2)	无
	BRA NN, Expr	如果非负则转移	1	1 (2)	无
	BRA NOV, Expr	如果未溢出则转移	1	1 (2)	无
	BRA NZ, Expr	如果非零则转移	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	相对转移	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 中的某位取反	1	1	无
	BTG Ws, #bit4	将 Ws 中的某位取反	1	1	无
BTSC	BTSC f, #bit4	对 f 中的指定位进行检测，如果为零则跳过	1	1 (2 或 3)	无
	BTSC Ws, #bit4	对 Ws 中的指定位进行检测，如果为零则跳过	1	1 (2 或 3)	无

# PIC24FJ64GB004 系列

表 28-2： 指令集概述（续）

汇编指令 助记符	汇编语法	说明	字数	周期数	受影响的 状态标志
BTSS	BTSS f,#bit4	对 f 中的指定位进行检测，如果为 1 则跳过	1	1 (2 或 3)	无
	BTSS Ws,#bit4	对 Ws 中的指定位进行检测，如果为 1 则跳过	1	1 (2 或 3)	无
BTST	BTST f,#bit4	对 f 中的指定位进行检测	1	1	Z
	BTST.C Ws,#bit4	对 Ws 中的指定位进行检测，并将结果存储到进位标志位 C 中	1	1	C
	BTST.Z Ws,#bit4	对 Ws 中的指定位进行检测，并将结果存储到全零标志位 Z 中	1	1	Z
	BTST.C Ws,Wb	对 Ws<Wb>位进行检测，并将结果存储到进位标志位 C 中	1	1	C
	BTST.Z Ws,Wb	对 Ws<Wb>位进行检测，并将结果存储到全零标志位 Z 中	1	1	Z
BTSTS	BTSTS f,#bit4	对 f 中的指定位进行检测，并将该位置 1	1	1	Z
	BTSTS.C Ws,#bit4	对 Ws 中的指定位进行检测，并将结果存储到进位标志位 C 中，然后将 Ws 中的该位置 1	1	1	C
	BTSTS.Z Ws,#bit4	对 Ws 中的指定位进行检测，并将结果存储到全零标志位 Z 中，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
CLRWDT	CLRWDT	将看门狗定时器清零	1	1	WDTO 和 Sleep
COM	COM f	f = $\bar{f}$	1	1	N 和 Z
	COM f,WREG	WREG = $\bar{f}$	1	1	N 和 Z
	COM Ws,Wd	Wd = $\bar{Ws}$	1	1	N 和 Z
CP	CP f	比较 f 和 WREG	1	1	C、DC、N、OV 和 Z
	CP Wb,#lit5	比较 Wb 和 lit5	1	1	C、DC、N、OV 和 Z
	CP Wb,Ws	比较 Wb 和 Ws (Wb - Ws)	1	1	C、DC、N、OV 和 Z
CP0	CP0 f	比较 f 和 0x0000	1	1	C、DC、N、OV 和 Z
	CP0 Ws	比较 Ws 和 0x0000	1	1	C、DC、N、OV 和 Z
CPB	CPB f	比较 f 和 WREG (通过带借位减法实现)	1	1	C、DC、N、OV 和 Z
	CPB Wb,#lit5	比较 Wb 和 lit5 (通过带借位减法实现)	1	1	C、DC、N、OV 和 Z
	CPB Wb,Ws	比较 Wb 和 Ws (通过带借位减法实现) (Wb - Ws - C)	1	1	C、DC、N、OV 和 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.B Wn	Wn = 对 Wn 进行十进制调整	1	1	C
DEC	DEC f	f = f - 1	1	1	C、DC、N、OV 和 Z
	DEC f,WREG	WREG = f - 1	1	1	C、DC、N、OV 和 Z
	DEC Ws,Wd	Wd = Ws - 1	1	1	C、DC、N、OV 和 Z
DEC2	DEC2 f	f = f - 2	1	1	C、DC、N、OV 和 Z
	DEC2 f,WREG	WREG = f - 2	1	1	C、DC、N、OV 和 Z
	DEC2 Ws,Wd	Wd = Ws - 2	1	1	C、DC、N、OV 和 Z
DISI	DISI #lit14	在 k 个指令周期内禁止中断	1	1	无
DIV	DIV.SW Wm,Wn	有符号 16/16 位整数除法	1	18	N、Z、C 和 OV
	DIV.SD Wm,Wn	有符号 32/16 位整数除法	1	18	N、Z、C 和 OV
	DIV.UW Wm,Wn	无符号 16/16 位整数除法	1	18	N、Z、C 和 OV
	DIV.UD Wm,Wn	无符号 32/16 位整数除法	1	18	N、Z、C 和 OV
EXCH	EXCH Wns,Wnd	将 Wns 和 Wnd 交换	1	1	无
FF1L	FF1L Ws,Wnd	从左边 (MSb) 查找第一个 1	1	1	C
FF1R	FF1R Ws,Wnd	从右边 (MSb) 查找第一个 1	1	1	C

表 28-2：指令集概述（续）

汇编指令 助记符	汇编语法	说明	字数	周期数	受影响的 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 和 Z
	INC f,WREG	WREG = f + 1	1	1	C、DC、N、OV 和 Z
	INC Ws,Wd	Wd = Ws + 1	1	1	C、DC、N、OV 和 Z
INC2	INC2 f	f = f + 2	1	1	C、DC、N、OV 和 Z
	INC2 f,WREG	WREG = f + 2	1	1	C、DC、N、OV 和 Z
	INC2 Ws,Wd	Wd = Ws + 2	1	1	C、DC、N、OV 和 Z
IOR	IOR f	f = f .IOR. WREG	1	1	N 和 Z
	IOR f,WREG	WREG = f .IOR. WREG	1	1	N 和 Z
	IOR #lit10,Wn	Wd = lit10 .IOR. Wd	1	1	N 和 Z
	IOR Wb,Ws,Wd	Wd = Wb .IOR. Ws	1	1	N 和 Z
	IOR Wb,#lit5,Wd	Wd = Wb .IOR. lit5	1	1	N 和 Z
LINK	LINK #lit14	链接帧指针	1	1	无
LSR	LSR f	f = 逻辑右移 f	1	1	C、N、OV 和 Z
	LSR f,WREG	WREG = 逻辑右移 f	1	1	C、N、OV 和 Z
	LSR Ws,Wd	Wd = 逻辑右移 Ws	1	1	C、N、OV 和 Z
	LSR Wb,Wns,Wnd	Wnd = 将 Wb 逻辑右移 Wns 位	1	1	N 和 Z
	LSR Wb,#lit5,Wnd	Wnd = 将 Wb 逻辑右移 lit5 位	1	1	N 和 Z
MOV	MOV f,Wn	将 f 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 和 Z
	MOV f,WREG	将 f 中的内容送入 WREG	1	1	N 和 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	
	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 和 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} = Signed(Wb) * Signed(Ws)	1	1	无
	MUL.SU Wb,Ws,Wnd	{Wnd + 1, Wnd} = Signed(Wb) * Unsigned(Ws)	1	1	无
	MUL.US Wb,Ws,Wnd	{Wnd + 1, Wnd} = Unsigned(Wb) * Signed(Ws)	1	1	无
	MUL.UU Wb,Ws,Wnd	{Wnd + 1, Wnd} = Unsigned(Wb) * Unsigned(Ws)	1	1	无
	MUL.SU Wb,#lit5,Wnd	{Wnd + 1, Wnd} = Signed(Wb) * Unsigned(lit5)	1	1	无
	MUL.UU Wb,#lit5,Wnd	{Wnd + 1, Wnd} = Unsigned(Wb) * Unsigned(lit5)	1	1	无
	MUL f	W3:W2 = f * WREG	1	1	无
NEG	NEG f	f = -f + 1	1	1	C、DC、N、OV 和 Z
	NEG f,WREG	WREG = -f + 1	1	1	C、DC、N、OV 和 Z
	NEG Ws,Wd	Wd = Ws + 1	1	1	C、DC、N、OV 和 Z
NOP	NOP	空操作	1	1	无
	NOPR	空操作	1	1	无
POP	POP f	从栈顶 (TOS) 弹出 f 寄存器的内容	1	1	无
	POP Wdo	将栈顶 (TOS) 的内容弹出到 Wdo 中	1	1	无
	POP.D Wnd	将栈顶 (TOS) 的内容弹出到 W(nd):W(nd+1) 中	1	2	无
	POP.S	将影子寄存器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) 中的内容压入栈顶 (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

# PIC24FJ64GB004 系列

表 28-2： 指令集概述（续）

汇编指令助记符	汇编语法	说明	字数	周期数	受影响的状态标志
PWRSAV	PWRSAV #lit1	进入休眠或空闲模式	1	1	WDTO 和 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	用软件使器件复位	1	1	无
RETFIE	RETFIE	从中断返回	1	3 (2)	无
RETLW	RETLW #lit10,Wn	返回并将立即数存入 Wn	1	3 (2)	无
RETURN	RETURN	从子程序返回	1	3 (2)	无
RLC	RLC f	f = 对 f 执行带进位的循环左移	1	1	C、N 和 Z
	RLC f,WREG	WREG = 对 f 执行带进位的循环左移	1	1	C、N 和 Z
	RLC Ws,Wd	Wd = 对 Ws 执行带进位的循环左移	1	1	C、N 和 Z
RLNC	RLNC f	f = 循环左移 f (不带进位)	1	1	N 和 Z
	RLNC f,WREG	WREG = 循环左移 f (不带进位)	1	1	N 和 Z
	RLNC Ws,Wd	Wd = 循环左移 Ws (不带进位)	1	1	N 和 Z
RRC	RRC f	f = 对 f 执行带进位的循环右移	1	1	C、N 和 Z
	RRC f,WREG	WREG = 对 f 执行带进位的循环右移	1	1	C、N 和 Z
	RRC Ws,Wd	Wd = 对 Ws 执行带进位的循环右移	1	1	C、N 和 Z
RRNC	RRNC f	f = 循环右移 f (不带进位)	1	1	N 和 Z
	RRNC f,WREG	WREG = 循环右移 f (不带进位)	1	1	N 和 Z
	RRNC Ws,Wd	Wd = 循环右移 Ws (不带进位)	1	1	N 和 Z
SE	SE Ws,Wnd	Wnd = 对 Ws 进行符号扩展	1	1	C、N 和 Z
SETM	SETM f	f = FFFFh	1	1	无
	SETM WREG	WREG = FFFFh	1	1	无
	SETM Ws	Ws = FFFFh	1	1	无
SL	SL f	f = 左移 f	1	1	C、N、OV 和 Z
	SL f,WREG	WREG = 左移 f	1	1	C、N、OV 和 Z
	SL Ws,Wd	Wd = 左移 Ws	1	1	C、N、OV 和 Z
	SL Wb,Wns,Wnd	Wnd = 将 Wb 左移 Wns 位	1	1	N 和 Z
	SL Wb,#lit5,Wnd	Wnd = 将 Wb 左移 lit5 位	1	1	N 和 Z
SUB	SUB f	f = f - WREG	1	1	C、DC、N、OV 和 Z
	SUB f,WREG	WREG = f - WREG	1	1	C、DC、N、OV 和 Z
	SUB #lit10,Wn	Wn = Wn - lit10	1	1	C、DC、N、OV 和 Z
	SUB Wb,Ws,Wd	Wd = Wb - Ws	1	1	C、DC、N、OV 和 Z
	SUB Wb,#lit5,Wd	Wd = Wb - lit5	1	1	C、DC、N、OV 和 Z
SUBB	SUBB f	f = f - WREG - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
	SUBB f,WREG	WREG = f - WREG - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
	SUBB #lit10,Wn	Wn = Wn - lit10 - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
	SUBB Wb,Ws,Wd	Wd = Wb - Ws - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
	SUBB Wb,#lit5,Wd	Wd = Wb - lit5 - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
SUBR	SUBR f	f = WREG - f	1	1	C、DC、N、OV 和 Z
	SUBR f,WREG	WREG = WREG - f	1	1	C、DC、N、OV 和 Z
	SUBR Wb,Ws,Wd	Wd = Ws - Wb	1	1	C、DC、N、OV 和 Z
	SUBR Wb,#lit5,Wd	Wd = lit5 - Wb	1	1	C、DC、N、OV 和 Z
SUBBR	SUBBR f	f = WREG - f - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
	SUBBR f,WREG	WREG = WREG - f - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
	SUBBR Wb,Ws,Wd	Wd = Ws - Wb - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
	SUBBR Wb,#lit5,Wd	Wd = lit5 - Wb - ( $\bar{C}$ )	1	1	C、DC、N、OV 和 Z
SWAP	SWAP.b Wn	Wn = 将 Wn 的两个半字节相交换	1	1	无
	SWAP Wn	Wn = 将 Wn 的两个字节相交换	1	1	无

**表 28-2： 指令集概述（续）**

汇编指令 助记符	汇编语法	说明	字数	周期数	受影响的 状态标志
TBLRDH	TBLRDH Ws,Wd	将程序计数器的 <23:16> 读入 Wd<7:0>	1	2	无
TBLRDL	TBLRDL Ws,Wd	将程序计数器的 <15:0> 读入 Wd	1	2	无
TBLWTH	TBLWTH Ws,Wd	将 Ws<7:0> 写入程序计数器的 <23:16>	1	2	无
TBLWTL	TBLWTL Ws,Wd	将 Ws 写入程序计数器的 <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 和 Z
	XOR f,WREG	WREG = f .XOR. WREG	1	1	N 和 Z
	XOR #lit10,Wn	Wd = lit10 .XOR. Wd	1	1	N 和 Z
	XOR Wb,Ws,Wd	Wd = Wb .XOR. Ws	1	1	N 和 Z
	XOR Wb,#lit5,Wd	Wd = Wb .XOR. lit5	1	1	N 和 Z
ZE	ZE Ws,Wnd	Wnd = 对 Ws 进行零扩展	1	1	C、N 和 Z

# **PIC24FJ64GB004 系列**

---

---

注：

## 29.0 电气特性

本章提供 PIC24FJ64GB004 系列电气特性的概述。在文档后续版本中会添加其他信息。

下面列出了 PIC24FJ64GB004 系列的绝对最大值。器件长时间工作在最大值条件下，其稳定性会受到影响。我们建议不要使器件在该规范规定的参数范围外工作。

### 绝对最大值<sup>(†)</sup>

环境温度.....	-40°C 至 +100°C
存储温度.....	-65°C 至 +150°C
VDD 引脚相对于 Vss 的电压.....	-0.3V 至 +4.0V
模拟数字组合引脚和 MCLR 引脚相对于 Vss 的电压.....	-0.3V 至 (VDD + 0.3V)
只能用作数字功能的引脚相对于 Vss 的电压.....	-0.3V 至 +6.0V
VDDCORE 引脚相对于 Vss 的电压 .....	-0.3V 至 +3.0V
Vss 引脚的最大输出电流.....	300 mA
VDD 引脚的最大输入电流（注 1）.....	250 mA
任一 I/O 引脚的最大输出灌电流 .....	25 mA
任一 I/O 引脚的最大输出拉电流 .....	25 mA
所有端口的最大灌电流 .....	200 mA
所有端口的最大拉电流（注 1）.....	200 mA

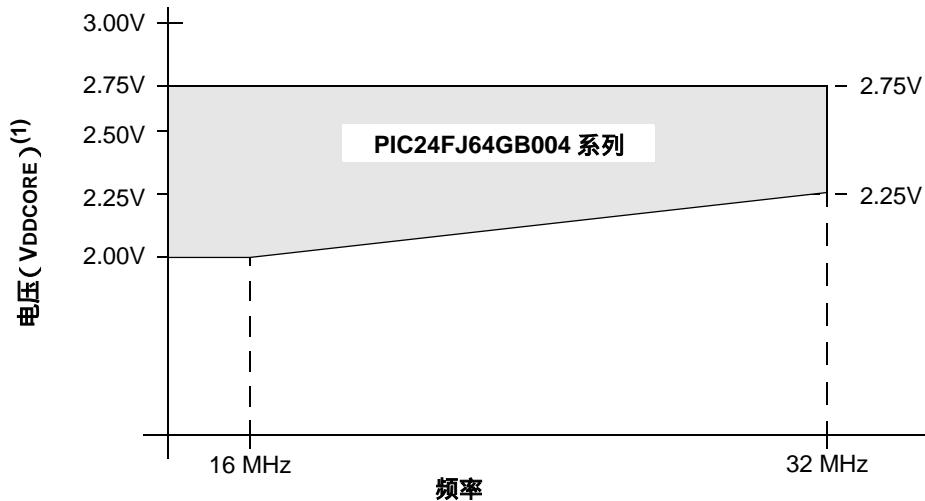
注 1：允许的最大电流由器件的最大功耗决定（见表 29-1）。

†注：如果器件的工作条件超过上述“绝对最大值”，可能会对器件造成永久性损坏。上述值仅为运行条件极大值，我们建议不要使器件在该规范规定的范围以外运行。器件长时间工作在最大值条件下，其稳定性会受到影响。

# PIC24FJ64GB004 系列

## 29.1 直流特性

图 29-1 : PIC24FJ64GB004 系列电压—频率关系图 (工业级)



当频率在 16 MHz 和 32 MHz 之间时， $F_{MAX} = (64 \text{ MHz}/V) * (V_{DDCORE} - 2\text{V}) + 16 \text{ MHz}$ 。

注 1：当禁止稳压器时，VDD 和 VDDCORE 必须满足以下条件：  
 $V_{DDCORE} \leq VDD \leq 3.6\text{V}$ 。

表 29-1 : 热工作条件

额定值	符号	最小值	典型值	最大值	单位
PIC24FJ64GB004 系列：					
工作结温范围	T <sub>J</sub>	-40	—	+125	°C
工作环境温度范围	T <sub>A</sub>	-40	—	+85	°C
功耗：					
内部芯片功耗： $P_{INT} = V_{DD} \times (I_{DD} - \sum I_{OH})$	P <sub>D</sub>	$P_{INT} + P_{I/O}$			W
I/O 引脚功耗： $P_{I/O} = \sum (\{V_{DD} - V_{OH}\} \times I_{OH}) + \sum (V_{OL} \times I_{OL})$					
允许的最大功耗	P <sub>DMAX</sub>	$(T_J - T_A)/\theta_{JA}$			W

表 29-2 : 封装热阻特性

特性	符号	典型值	最大值	单位	注
封装热阻，300 mil SOIC	θ <sub>JA</sub>	49	—	°C/W	(注 1)
封装热阻，6x6x0.9 mm QFN	θ <sub>JA</sub>	33.7	—	°C/W	(注 1)
封装热阻，8x8x1 mm QFN	θ <sub>JA</sub>	28	—	°C/W	(注 1)
封装热阻，10x10x1 mm TQFP	θ <sub>JA</sub>	39.3	—	°C/W	(注 1)

注 1：通过封装模拟获得结点与环境的热阻值 θ<sub>JA</sub>。

表 29-3 : 直流特性 : 温度和电压规范

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°C ≤ TA ≤ +85°C (工业级)				
参数 编号	符号	特性	最小值	典型值 <sup>(1)</sup>	最大值	单位	条件
<b>工作电压</b>							
DC10	<b>电源电压</b>						
	VDD		2.2	—	3.6	V	使能稳压器
	VDD		VDDCORE	—	3.6	V	禁止稳压器
	VDDCORE		2.0	—	2.75	V	禁止稳压器
DC12	VDR	<b>RAM 数据保持电压<sup>(2)</sup></b>	1.5	—	—	V	
DC16	VPOR	确保内部上电复位信号的 <b>VDD 起始电压</b>	—	VSS	—	V	
DC17	SVDD	确保内部上电复位信号的 <b>VDD 上升速率</b>	0.05	—	—	V/ms	0.1s 内电压变化范围为 0-3.3V 60 ms 内电压变化范围为 0-2.5V
DC18	VBOR	<b>欠压复位电压</b>	TBD	2.05	TBD	V	

图注 : TBD = 待定

注 1 : 除非另外声明 , 否则 “典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。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。

# PIC24FJ64GB004 系列

表 29-4 : 直流特性 : 工作电流 ( IDD )

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°C ≤ TA ≤ +85°C (工业级)			
参数 编号	典型值 <sup>(1)</sup>	最大值	单位	条件		
<b>工作电流 ( IDD )<sup>(2)</sup></b>						
DC21	0.24	0.395	mA	-40°C	2.0V <sup>(3)</sup>	0.5 MIPS
DC21a	0.25	0.395	mA	+25°C		
DC21b	0.25	0.395	mA	+85°C		
DC21c	0.44	0.78	mA	-40°C		3.3V <sup>(4)</sup>
DC21d	0.41	0.78	mA	+25°C		
DC21e	0.41	0.78	mA	+85°C		
DC20	0.5	0.75	mA	-40°C	2.0V <sup>(3)</sup>	1 MIPS
DC20a	0.5	0.75	mA	+25°C		
DC20b	0.5	0.75	mA	+85°C		
DC20d	0.75	1.4	mA	-40°C	3.3V <sup>(4)</sup>	4 MIPS
DC20e	0.75	1.4	mA	+25°C		
DC20f	0.75	1.4	mA	+85°C		
DC23	2.0	3.0	mA	-40°C	2.0V <sup>(3)</sup>	16 MIPS
DC23a	2.0	3.0	mA	+25°C		
DC23b	2.0	3.0	mA	+85°C		
DC23d	2.9	4.2	mA	-40°C	3.3V <sup>(4)</sup>	LPRC (31 kHz)
DC23e	2.9	4.2	mA	+25°C		
DC23f	2.9	4.2	mA	+85°C		
DC24	10.5	15.5	mA	-40°C	2.5V <sup>(3)</sup>	LPRC (31 kHz)
DC24a	10.5	15.5	mA	+25°C		
DC24b	10.5	15.5	mA	+85°C		
DC24d	11.3	15.5	mA	-40°C	3.3V <sup>(4)</sup>	LPRC (31 kHz)
DC24e	11.3	15.5	mA	+25°C		
DC24f	11.3	15.5	mA	+85°C		
DC31	15.0	18.0	μA	-40°C	2.0V <sup>(3)</sup>	LPRC (31 kHz)
DC31a	15.0	19.0	μA	+25°C		
DC31b	20.0	36.0	μA	+85°C		
DC31d	57.0	120.0	μA	-40°C	3.3V <sup>(4)</sup>	LPRC (31 kHz)
DC31e	57.0	125.0	μA	+25°C		
DC31f	95.0	160.0	μA	+85°C		

注 1 : 除非另外声明 , 否则 “典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考 , 未经测试。

2 : 供电电流主要受工作电压和频率的影响。其他因素如 I/O 引脚负载和开关频率、振荡器类型、内部代码执行模式以及温度也对电流消耗有影响。所有 IDD 测量的测试条件为 : 使用满幅的外部方波驱动 OSC1。所有 I/O 引脚配置为输入且被拉到 VDD。

MCLR = VDD , WDT 和 FSCM 被禁止。CPU、SRAM、程序存储器和数据存储器正常工作。外设模块均不工作且所有的外设模块禁止 (PMD) 位均置 1。

3 : 禁止片上稳压器 (DISVREG 连接到 Vss)。

4 : 使能片上稳压器 (DISVREG 连接到 VDD)。使能了低电压检测 (Low-Voltage Detect , LVD) 和欠压检测 (Brown-out Detect , BOD)。

表 29-5 : 直流特性 : 空闲电流 (I<sub>IDLE</sub>)

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°C ≤ TA ≤ +85°C (工业级)			
参数 编号	典型值 <sup>(1)</sup>	最大值	单位	条件		
<b>空闲电流 (I<sub>IDLE</sub>)<sup>(2)</sup></b>						
DC41	67	100	μA	-40°C	2.0V <sup>(3)</sup>	0.5 MIPS
DC41a	68	100	μA	+25°C		
DC41b	74	100	μA	+85°C		
DC41c	166	265	μA	-40°C		
DC41d	167	265	μA	+25°C		
DC41e	177	265	μA	+85°C		
DC40	125	180	μA	-40°C		1 MIPS
DC40a	125	180	μA	+25°C		
DC40b	125	180	μA	+85°C		
DC40d	210	350	μA	-40°C		
DC40e	210	350	μA	+25°C		
DC40f	210	350	μA	+85°C		
DC43	0.5	0.6	mA	-40°C	2.0V <sup>(3)</sup>	4 MIPS
DC43a	0.5	0.6	mA	+25°C		
DC43b	0.5	0.6	mA	+85°C		
DC43d	0.75	0.95	mA	-40°C		
DC43e	0.75	0.95	mA	+25°C		
DC43f	0.75	0.95	mA	+85°C		
DC47	2.6	3.3	mA	-40°C	2.5V <sup>(3)</sup>	16 MIPS
DC47a	2.6	3.3	mA	+25°C		
DC47b	2.6	3.3	mA	+85°C		
DC47c	2.9	3.5	mA	-40°C		
DC47d	2.9	3.5	mA	+25°C		
DC47e	2.9	3.5	mA	+85°C		
DC50	0.8	1.0	mA	-40°C	2.0V <sup>(3)</sup>	FRC ( 4 MIPS )
DC50a	0.8	1.0	mA	+25°C		
DC50b	0.8	1.0	mA	+85°C		
DC50d	1.1	1.3	mA	-40°C		
DC50e	1.1	1.3	mA	+25°C		
DC50f	1.1	1.3	mA	+85°C		
DC51	2.4	8.0	μA	-40°C	2.0V <sup>(3)</sup>	LPRC ( 31 kHz )
DC51a	2.2	8.0	μA	+25°C		
DC51b	7.2	21.0	μA	+85°C		
DC51d	38	55	μA	-40°C		
DC51e	44	60	μA	+25°C		
DC51f	70	100	μA	+85°C		

注 1：除非另外声明，否则“典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考，未经测试。

2：基本 I<sub>IDLE</sub> 电流是在内核关闭且使用满幅的外部方波驱动 OSCI 的条件下测得的。所有 I/O 引脚配置为输入且被拉到 V<sub>DD</sub>。MCLR = V<sub>DD</sub>；WDT 和 FSCM 被禁止。外设模块均不工作且所有的外设模块禁止 (PMD) 位均置 1。

3：禁止片上稳压器 (DISVREG 连接到 V<sub>DD</sub>)。

4：使能片上稳压器 (DISVREG 连接到 V<sub>SS</sub>)。使能了低电压检测 (LVD) 和欠压检测 (BOD)。

# PIC24FJ64GB004 系列

表 29-6： 直流特性：基本掉电电流 (IPD)

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°C ≤ TA ≤ +85°C (工业级)			
参数 编号	典型值 <sup>(1)</sup>	最大值	单位	条件		
<b>掉电电流 (IPD) <sup>(2)</sup></b>						
DC60	0.05	1.0	μA	-40°C	2.0V <sup>(3)</sup>	基本掉电电流 <sup>(5)</sup>
DC60a	0.2	1.0	μA	+25°C		
DC60i	2.0	6.5	μA	+60°C		
DC60b	3.5	12.0	μA	+85°C		
DC60c	0.1	1.0	μA	-40°C		
DC60d	0.4	1.0	μA	+25°C		
DC60j	2.5	15	μA	+60°C		
DC60e	4.2	25	μA	+85°C		
DC60f	3.3	9.0	μA	-40°C	2.5V <sup>(3)</sup>	基本深度休眠电流
DC60g	3.3	10.0	μA	+25°C		
DC60k	5.0	20.0	μA	+60°C		
DC60h	7.0	30.0	μA	+85°C		
DC70c	.003	0.2	μA	-40°C	2.5V <sup>(4)</sup>	基本深度休眠电流
DC70d	0.02	0.2	μA	+25°C		
DC70j	0.2	0.35	μA	+60°C		
DC70e	.51	1.5	μA	+85°C		
DC70f	.01	0.3	μA	-40°C	3.3V <sup>(4)</sup>	基本深度休眠电流
DC70g	0.04	0.3	μA	+25°C		
DC70k	0.2	0.5	μA	+60°C		
DC70h	.71	2.0	μA	+85°C		

注 1：除非另外声明，否则“典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考，未经测试。

2：基本 IPD 是在器件处于休眠模式（即关闭所有外设和时钟）的条件下测得的。所有 I/O 引脚都配置为输入且被拉至高电平。WDT 等外设均关闭，PMSLP 位被清零，且用于所有未使用外设的外设模块禁止 (PMD) 位置 1。

3：禁止片上稳压器 (DISVREG 连接到 VDD)。

4：使能片上稳压器 (DISVREG 连接到 Vss)。使能了低电压检测 (LVD) 和欠压检测 (BOD)。

5：电流为模块使能时额外消耗的电流。掉电时外设模块的电流消耗是这一电流与基本 IPD 电流之和。

表 29-7 : 直流特性 : 掉电外设模块  $\Delta$  电流 (IPD)

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°C ≤ TA ≤ +85°C (工业级)				
参数 编号	典型值 <sup>(1)</sup>	最大值	单位	条件			
<b>掉电电流 (IPD) : PMD 位置 1 , PMSLP 位为 0<sup>(2)</sup></b>							
DC61	0.2	0.7	μA	-40°C	2.0V <sup>(3)</sup>	使与 31 kHz LPRC 振荡器配合使用的 RTCC、WDT、DSWDT 或 Timer 1 : $\Delta$ ILPRC <sup>(5)</sup>	
DC61a	0.2	0.7	μA	+25°C			
DC61i	0.2	0.7	μA	+60°C			
DC61b	0.23	0.7	μA	+85°C			
DC61c	0.25	0.9	μA	-40°C	2.5V <sup>(3)</sup>		
DC61d	0.25	0.9	μA	+25°C			
DC61j	0.25	0.9	μA	+60°C			
DC61e	0.28	0.9	μA	+85°C			
DC61f	0.6	1.5	μA	-40°C	3.3V <sup>(4)</sup>		
DC61g	0.6	1.5	μA	+25°C			
DC61k	0.6	1.5	μA	+60°C			
DC61h	0.8	1.5	μA	+85°C			
DC62	0.5	1.0	μA	-40°C	2.0V <sup>(3)</sup>	低驱动强度，与 32 kHz 晶振配合使用的 RTCC、DSWDT 或 Timer1 : $\Delta$ ISOSC ; SOSCSEL = 01 <sup>(5)</sup>	
DC62a	0.5	1.0	μA	+25°C			
DC62i	0.5	1.0	μA	+60°C			
DC62b	0.5	1.3	μA	+85°C			
DC62c	0.7	1.5	μA	-40°C	2.5V <sup>(3)</sup>		
DC62d	0.7	1.5	μA	+25°C			
DC62j	0.7	1.5	μA	+60°C			
DC62e	0.7	1.8	μA	+85°C			
DC62f	1.5	2.0	μA	-40°C	3.3V <sup>(4)</sup>		
DC62g	1.5	2.0	μA	+25°C			
DC62k	1.5	2.0	μA	+60°C			
DC62h	1.5	2.5	μA	+85°C			

注 1 : 除非另外声明，否则“典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考，未经测试。

2 : 外设 IPD 增量是在器件处于休眠模式（即关闭所有外设和时钟）的条件下测得的。所有 I/O 引脚都配置为输入且被拉至高电平。只使能了要测量的外设或时钟。PMSLP 位清零，且用于所有未使用外设的外设模块禁止位 (PMD) 置 1。

3 : 禁止片上稳压器 (DISVREG 连接到 VDD)。

4 : 使能片上稳压器 (DISVREG 连接到 Vss)。使能了低电压检测 (LVD) 和欠压检测 (BOD)。

5 : 电流为模块使能时额外消耗的电流。掉电时外设模块的电流消耗是这一电流与基本 IPD 电流之和。

# PIC24FJ64GB004 系列

表 29-7： 直流特性：掉电外设模块  $\Delta$  电流 (IPD) (续)

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°C ≤ TA ≤ +85°C (工业级)		
参数 编号	典型值 <sup>(1)</sup>	最大值	单位	条件	
<b>掉电电流 (IPD) : PMD 位置 1 , PMSLP 位为 0<sup>(2)</sup></b>					
DC63	1.8	2.3	μA	-40°C	2.0V <sup>(3)</sup>  使用与 32-kHz 晶振配合使用的 RTCC、DSWDT 或 Timer1 : ΔIsosc ; SOSCSEL = 11 <sup>(5)</sup>
DC63a	1.8	2.7	μA	+25°C	
DC63i	1.8	3.0	μA	+60°C	
DC63b	1.8	3.0	μA	+85°C	
DC63c	2	2.7	μA	-40°C	
DC63d	2	2.9	μA	+25°C	
DC63j	2	3.2	μA	+60°C	
DC63e	2	3.5	μA	+85°C	
DC63f	2.25	3.0	μA	-40°C	
DC63g	2.25	3.0	μA	+25°C	
DC63k	2.25	3.3	μA	+60°C	3.3V <sup>(4)</sup>  深度休眠 BOR : ΔIDSBOR
DC63h	2.25	3.5	μA	+85°C	
DC71c	.001	0.25	μA	-40°C	
DC71d	.03	0.25	μA	+25°C	
DC71j	0.05	0.60	μA	+60°C	
DC71e	.08	2.0	μA	+85°C	
DC71f	.001	0.50	μA	-40°C	
DC71g	.03	0.50	μA	+25°C	
DC71k	0.05	0.75	μA	+60°C	3.3V <sup>(4)</sup>
DC71h	.08	2.50	μA	+85°C	

注 1：除非另外声明，否则“典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考，未经测试。

2：外设 IPD 增量是在器件处于休眠模式（即关闭所有外设和时钟）的条件下测得的。所有 I/O 引脚都配置为输入且被拉至高电平。只使能了要测量的外设或时钟。PMSLP 位清零，且用于所有未使用外设的外设模块禁止位 (PMD) 置 1。

3：禁止片上稳压器 (DISVREG 连接到 VDD)。

4：使能片上稳压器 (DISVREG 连接到 Vss)。使能了低电压检测 (LVD) 和欠压检测 (BOD)。

5：电流为模块使能时额外消耗的电流。掉电时外设模块的电流消耗是这一电流与基本 IPD 电流之和。

表 29-8 : 直流特性 : I/O 引脚输入规范

直流特性			标准工作条件 : 2.0V 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 <sup>(1)</sup>	最大值	单位	条件
DI10 DI11  DI15 DI16 DI17 DI18 DI19	VIL	输入低电压 <sup>(4)</sup> 带 ST 缓冲器的 I/O 引脚 带 TTL 缓冲器的 I/O 引脚	Vss	—	0.2 VDD	V	
		MCLR	Vss	—	0.15 VDD	V	
		OSC1 (XT 模式)	Vss	—	0.2 VDD	V	
		OSC1 (HS 模式)	Vss	—	0.2 VDD	V	
		带 I <sup>2</sup> C <sup>TM</sup> 缓冲器的 I/O 引脚 :	Vss	—	0.3 VDD	V	
		带 SMBus 缓冲器的 I/O 引脚 :	Vss	—	0.8	V	使能 SMBus
DI20 DI21  DI25 DI26 DI27 DI28 DI29	VIH	输入高电压 <sup>(4)</sup> 带 ST 缓冲器的 I/O 引脚 : 具有模拟功能 , 仅有数字功能	0.8 VDD 0.8 VDD	— —	VDD 5.5	V	
		带 TTL 缓冲器的 I/O 引脚 : 具有模拟功能 , 仅有数字功能	0.25 VDD + 0.8 0.25 VDD + 0.8	— —	VDD 5.5	V	
		MCLR	0.8 VDD	—	VDD	V	
		OSC1 (XT 模式)	0.7 VDD	—	VDD	V	
		OSC1 (HS 模式)	0.7 VDD	—	VDD	V	
		带 I <sup>2</sup> C 缓冲器的 I/O 引脚 : 具有模拟功能 , 仅有数字功能	0.7 VDD 0.7 VDD	— —	VDD 5.5	V	
		带 SMBus 缓冲器的 I/O 引脚 : 具有模拟功能 , 仅有数字功能	2.1 2.1		VDD 5.5	V	2.5V ≤ VPIN ≤ VDD
DI30	ICNPU	CNx 上拉电流	50	250	400	μA	VDD = 3.3V , VPIN = VSS
DI50 DI51  DI52 DI55 DI56	IIL	输入泄漏电流 <sup>(2,3)</sup> I/O 端口	—	—	±50	μA	VSS ≤ VPIN ≤ VDD , 引脚处于高阻态
		模拟输入引脚	—	—	±50	μA	VSS ≤ VPIN ≤ VDD , 引脚处于高阻态
		USB 差分引脚 (D+ 和 D-)	—	—	±50	μA	V <sub>USB</sub> ≥ VDD
		MCLR	—	—	±50	μA	VSS ≤ VPIN ≤ VDD
		OSC1	—	—	±50	μA	VSS ≤ VPIN ≤ VDD, XT 和 HS 模式

注 1: 除非另外声明, 否则 “典型值” 栏中的数据都是在 3.3V、25°C 的条件下给出的。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加电平。表中给出的电平表示正常工作条件下的电平。在不同的输入电压条件下可能测得更高的泄漏电流。

3: 负电流定义为自引脚流出的电流。

4: I/O 引脚缓冲器类型请参见表 1-2。

# PIC24FJ64GB004 系列

表 29-9 : 直流特性 : I/O 引脚输出规范

直流特性			标准工作条件 : 2.0V 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 <sup>(1)</sup>	最大值	单位	条件
DO10	VOL	输出低电压 I/O 端口	—	—	0.4	V	IOL = 8.5 mA , VDD = 3.6V
			—	—	0.4	V	IOL = 6.0 mA , VDD = 2.0V
DO20	VOH	输出高电压 I/O 端口	3.0	—	—	V	IOH = -3.0 mA , VDD = 3.6V
			2.4	—	—	V	IOH = -6.0 mA , VDD = 3.6V
			1.65	—	—	V	IOH = -1.0 mA , VDD = 2.0V
			1.4	—	—	V	IOH = -3.0 mA , VDD = 2.0V

注 1 : 除非另外声明, 否则 “典型值” 栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考, 未经测试。

表 29-10 : 直流特性 : 程序存储器

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°C ≤ TA ≤ +85°C (工业级)				
参数编号	符号	特性	最小值	典型值 <sup>(1)</sup>	最大值	单位	条件
D130	EP	单元耐擦写能力	10,000	—	—	E/W	-40°C 至 +85°C
D131	VPR	读操作使用的 VDD	VMIN	—	3.6	V	VMIN = 最小工作电压
D132A D132B	VPEW	自定时写操作使用的供电电压 VDDCORE VDD	2.25	—	3.6	V	
			2.35	—	3.6	V	
D133A	TIW	自定时写周期时间	—	3	—	ms	
D133B	TIE	自定时页擦除时间	40	—	—	ms	
D134	TRETD	特性保持时间	20	—	—	年	假定未违反其他规范
D135	IDDP	编程时的供电电流	—	7	—	mA	

注 1 : 除非另外声明, 否则 “典型值” 栏中的数据都是在 3.3V、25°C 的条件下给出的。

**表 29-11 : 比较器规范**

工作条件 : $2.0V < VDD < 3.6V$ , $-40^{\circ}C < TA < +85^{\circ}C$ (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	VIOFF	输入失调电压 *	—	20	40	mV	
D301	VICM	输入共模电压 *	0	—	VDD	V	
D302	CMRR	共模抑制比 *	55	—	—	dB	
300	TRESP	响应时间 *(1)	—	150	400	ns	
301	TMC2OV	比较器模式更改到输出有效的时间 *	—	—	10	μs	

\* 参数仅为特性值，未经测试。

注 1：响应时间是在比较器的一个输入端为  $(VDD - 1.5)/2$ ，而另一输入端从 Vss 变化到 VDD 的情况下测得的。

**表 29-12 : 比较器参考电压规范**

工作条件 : $2.0V < VDD < 3.6V$ , $-40^{\circ}C < TA < +85^{\circ}C$ (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	VDD/24	—	VDD/32	LSb	
VRD311	CVRAA	绝对精度	—	—	AVDD - 1.5	LSb	
VRD312	CVRUR	单元电阻值 (R)	—	2k	—	Ω	
VR310	TSET	稳定时间 *(1)	—	—	10	μs	

注 1：稳定时间是在 CVRR = 1 且 CVR<3:0> 位从 0000 变化到 1111 的情况下测得的。

**表 29-13 : 内部稳压器规范**

工作条件 : $-40^{\circ}C < TA < +85^{\circ}C$ (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
	VBG	带隙参考电压	1.14	1.2	1.26	V	
	TBG	带隙参考启动时间	—	1	—	ms	
	VRGOUT	稳压器输出电压	2.35	2.5	2.75	V	
	CEFC	外部滤波器电容值	4.7	10	—	μF	建议串联一个阻抗 $< 3\Omega$ 的电阻；要求串联电阻的阻抗 $< 5\Omega$ 。

# PIC24FJ64GB004 系列

## 29.2 交流特性和时序参数

本节包含的信息说明了 PIC24FJ64GB004 系列器件的交流特性和时序参数。

表 29-14 : 温度和电压规范——交流

交流特性	标准工作条件 : 2.0V 至 3.6V (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq \text{TA} \leq +85^{\circ}\text{C}$ (工业级) 工作电压 VDD 范围如第 29.1 节 “直流特性” 所示。
------	--

图 29-2 : 器件时序规范的负载条件

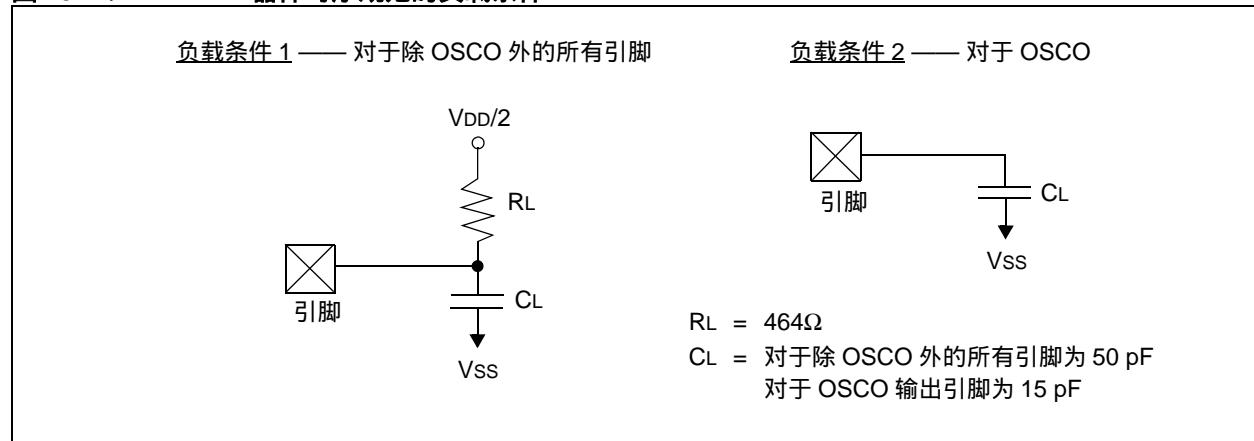


表 29-15 : 输出引脚上的容性负载要求

参数编号	符号	特性	最小值	典型值 <sup>(1)</sup>	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。
DO56	CIO	所有 I/O 引脚和 OSCO	—	—	50	pF	EC 模式。
DO58	CB	SCLx 和 SDAx	—	—	400	pF	在 I <sup>2</sup> C <sup>TM</sup> 模式下。

注 1 : 除非另外声明，否则“典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考，未经测试。

图 29-3 : 外部时钟时序

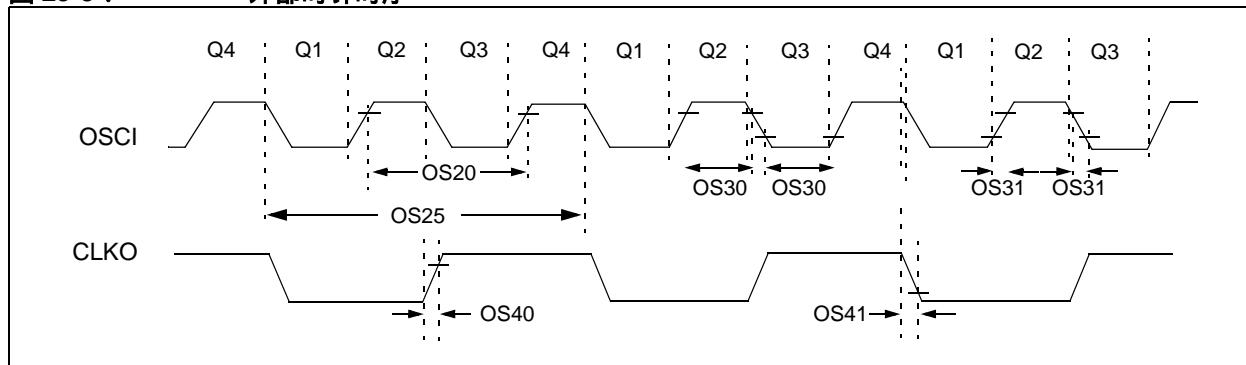


表 29-16 : 外部时钟时序要求

交流特性			标准工作条件 : 2.50 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 <sup>(1)</sup>	最大值	单位	条件
OS10	Fosc	外部 CLKI 频率 (仅在 EC 模式下允许使用外部时钟)	DC 4	—	32 MHz	MHz	EC
		振荡器频率	3	—	48 MHz	MHz	ECPLL
			4	—	10 MHz	MHz	XT
			10	—	8 MHz	MHz	XTPLL
			12	—	32 MHz	MHz	HS
			31	—	32 MHz	MHz	HSPLL
OS20	Tosc	$T_{osc} = 1/F_{osc}$	—	—	—	ns	关于 Fosc 的值 , 见参数 OS10
OS25	Tcy	指令周期 <sup>(2)</sup>	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 x Tosc	—	—	ns	EC
OS31	TosR, TosF	外部时钟输入 (OSCI) 上升或下降时间	—	—	20 ns	ns	EC
OS40	TckR	CLKO 上升时间 <sup>(3)</sup>	—	6	10 ns	ns	
OS41	TckF	CLKO 下降时间 <sup>(3)</sup>	—	6	10 ns	ns	

注 1 : 除非另外声明 , 否则 “典型值” 栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考 , 未经测试。

2 : 指令周期 (TCY) 等于输入振荡器时基周期的 2 倍。所有规范值均基于标准工作条件下 , 器件执行代码时对应特定振荡器类型的特性数据。超过规范值可导致振荡器运行不稳定和 / 或使电流消耗超过预期值。所有器件在测试 “最小值” 时 , 均在 OSCI/CLKI 引脚接入了外部时钟。当使用外部时钟输入时 , 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。CLKO 信号是在 OSCO 引脚上测得的。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。

# PIC24FJ64GB004 系列

表 29-17 : PLL 时钟时序规范 (VDD = 2.0V 至 3.6V)

交流特性			标准工作条件 : 2.0V 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 <sup>(1)</sup>	最小值	典型值 <sup>(2)</sup>	最大值	单位	条件
OS50	FPLL1	PLL 输入频率范围 <sup>(2)</sup>	4	—	32	MHz	ECPLL、HSPLL 和 XTPLL 模式
OS51	FSYS	PLL 输出频率范围	95.76	—	96.24	MHz	
OS52	TLOCK	PLL 启动时间 (锁定时间)	—	180	—	μs	
OS53	DCLK	CLKO 稳定性 (去抖动性能)	-0.25	—	0.25	%	

注 1 : 这些参数为特征值，但未经生产测试。

2 : 除非另外声明，否则“典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。这些参数仅供设计参考，未经测试。

表 29-18 : 内部 RC 振荡器规范

交流特性			标准工作条件 : 2.0V 至 3.6V (除非另外声明) 工作温度 -40°C ≤ TA ≤ +85°C (工业级)				
参数编号	符号	特性 <sup>(1)</sup>	最小值	典型值	最大值	单位	条件
	TFRC	FRC 启动时间	—	15	—	μs	
	TLPRC	LPRC 启动时间	—	500	—	μs	

表 29-19 : 内部 RC 振荡器精度

交流特性		标准工作条件 : 2.0V 至 3.6V (除非另外声明) 工作温度 -40°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 精度 @ 8 MHz <sup>(1,3)</sup>	-1.25	±0.25	1.0	%	-40°C ≤ TA ≤ +85°C , 3.0V ≤ VDD ≤ 3.6V	
F21	LPRC 精度 @ 31 kHz <sup>(2)</sup>	-15	—	15	%	-40°C ≤ TA ≤ +85°C , 3.0V ≤ VDD ≤ 3.6V	

注 1 : 已在 25°C、3.3V 条件下对频率进行了校准。OSCTUN 位可用来补偿温度漂移。

2 : LPRC 频率将随 VDD 的变化而变化。

3 : 要达到这一精度，单片机封装所受到的物理应力（例如：弯曲 PCB）必须保持最小。

图 29-4 : CLKO 和 I/O 时序特性

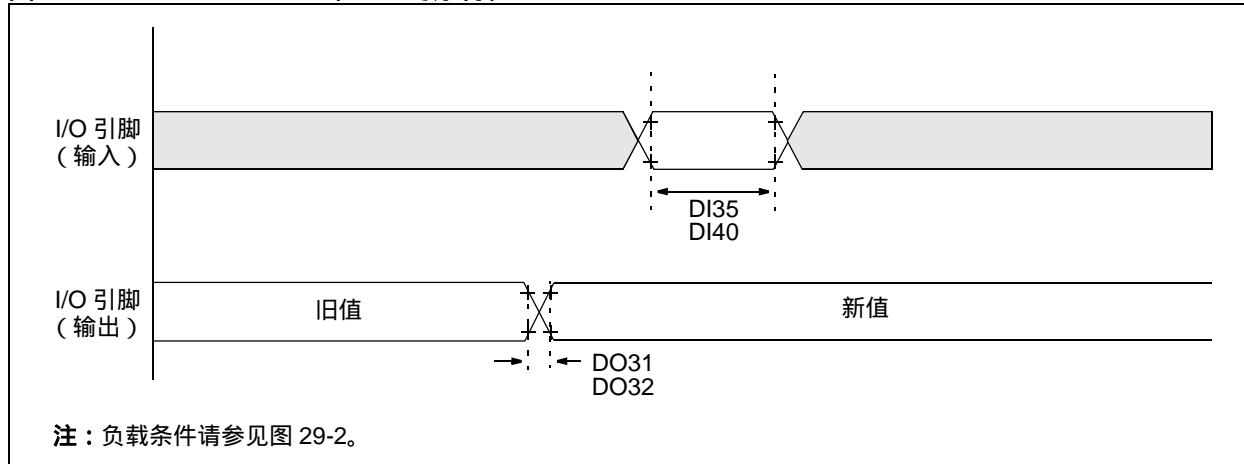


表 29-20 : CLKO 和 I/O 时序要求

交流特性			标准工作条件 : 2.0V 至 3.6V (除非另外声明) 工作温度 -40°C ≤ TA ≤ +85°C (工业级)				
参数编号	符号	特性	最小值	典型值 <sup>(1)</sup>	最大值	单位	条件
DO31	T <sub>IO</sub> R	端口输出上升时间	—	10	25	ns	
DO32	T <sub>IO</sub> F	端口输出下降时间	—	10	25	ns	
DI35	T <sub>INP</sub>	INTx 引脚高电平或低电平时间 (输出)	20	—	—	ns	
DI40	T <sub>RB</sub> P	CNx 高电平或低电平时间 (输入)	2	—	—	T <sub>CY</sub>	

注 1：除非另外声明，否则“典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。

表 29-21 : 复位、上电延时定时器和欠压复位时序要求

交流特性			标准工作条件 : 2.0V 至 3.6V (除非另外声明) 工作温度 -40°C ≤ TA ≤ +85°C (工业级)				
参数编号	符号	特性	最小值	典型值 <sup>(1)</sup>	最大值	单位	条件
SY10	T <sub>mCL</sub>	MCLR 脉冲宽度 (低电平)	2	—	—	μs	
SY11	T <sub>PWRT</sub>	上电延时定时器周期	—	64	—	ms	
SY12	T <sub>POR</sub>	上电复位延时	—	2	—	μs	
SY13	T <sub>IOZ</sub>	在 MCLR 低电平或看门狗定时器复位之后 I/O 处于高阻态的时间	—	—	100	ns	
SY25	T <sub>BOR</sub>	欠压复位脉冲宽度	1	—	—	μs	V <sub>DD</sub> ≤ V <sub>BOR</sub>
	T <sub>RST</sub>	内部状态复位时间	—	50	—	μs	
	T <sub>DSWU</sub>	从深度休眠唤醒时间	—	200	—	μs	基于 V <sub>CAP</sub> 上 10 μF 电容的完全放电。包含 T <sub>POR</sub> 和 T <sub>RST</sub> 。
	T <sub>PM</sub>		—	10 — 190	—	μs μs	PMSLP = 0 且 WUTSEL<1:0> = 11 时的休眠唤醒

注 1：除非另外声明，否则“典型值”栏中的数据都是在 3.3V、25°C 的条件下给出的。

# PIC24FJ64GB004 系列

表 29-22 : ADC 模块规范

交流特性			标准工作条件：2.0V 至 3.6V (除非另外声明) 工作温度 -40°C ≤ TA ≤ +85°C (工业级)				
参数 编号	符号	特性	最小值	典型值	最大值	单位	条件
<b>器件电源</b>							
AD01	AVDD	模块电源 VDD	取 VDD - 0.3 或 2.0 中的较大值	—	取 VDD + 0.3 或 3.6 中的较小值	V	
AD02	AVSS	模块电源 VSS	VSS - 0.3	—	VSS + 0.3	V	
<b>参考输入</b>							
AD05	VREFH	参考电压高电平	AVSS + 1.7	—	AVDD	V	
AD06	VREFL	参考电压低电平	AVSS	—	AVDD - 1.7	V	
AD07	VREF	绝对参考电压	AVSS - 0.3	—	AVDD + 0.3	V	
<b>模拟输入</b>							
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	
AD13	—	泄漏电流	—	±0.001	±0.610	µA	VINL = AVSS = VREFL = 0V , AVDD = VREFH = 3V , 源阻抗 = 2.5 kΩ
AD17	RIN	模拟信号源的推荐阻抗	—	—	2.5K	Ω	10 位
<b>ADC 精度</b>							
AD20b	NR	分辨率	—	10	—	位	
AD21b	INL	积分非线性误差	—	±1	≤ ±2	LSb	VINL = AVSS = VREFL = 0V , AVDD = VREFH = 3V
AD22b	DNL	微分非线性误差	—	±0.5	< ±1.25	LSb	VINL = AVSS = VREFL = 0V , AVDD = VREFH = 3V
AD23b	GERR	增益误差	—	±1	±3	LSb	VINL = AVSS = VREFL = 0V , AVDD = VREFH = 3V
AD24b	EOFF	失调误差	—	±1	±2	LSb	VINL = AVSS = VREFL = 0V , AVDD = VREFH = 3V
AD25b	—	单调性 (1)	—	—	—	—	保证

注 1 : A/D 转换结果将不会随着输入电压的增加而减小，而且不会丢失编码。

2 : 测量是在使用外部 VREF+ 和 VREF- 作为 ADC 参考电压的情况下进行的。

表 29-23 : ADC 转换时序要求<sup>(1)</sup>

交流特性			标准工作条件 : 2.0V 至 3.6V (除非另外声明) 工作温度 -40°C ≤ TA ≤ +85°C (工业级)				
参数 编号	符号	特性	最小值	典型值	最大值	单位	条件
时钟参数							
AD50	TAD	ADC 时钟周期	75	—	—	ns	TCY = 75 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	FCONV	吞吐率	—	—	500	ksps	AVDD > 2.7V
AD57	tsAMP	采样时间	—	1	—	TAD	
时钟参数							
AD61	tPSS	从采样位 (SAMP) 置 1 到采样 启动的延时	2	—	3	TAD	

注 1：因为采样电容最终将释放电荷，因此低于 10 kHz 的时钟频率可能会影响线性性能，尤其在温度较高时。

# **PIC24FJ64GB004 系列**

---

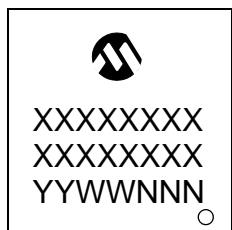
---

注：

## 30.0 封装信息

### 30.1 封装标识信息

28 引脚 QFN



示例



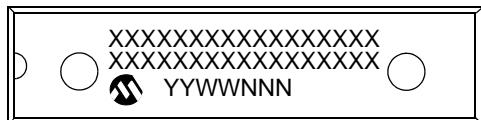
28 引脚 SOIC (.300")



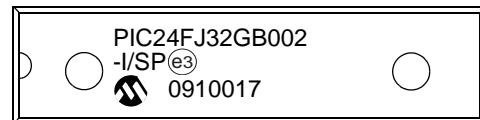
示例



28 引脚 SPDIP



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）
	Y	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
		雾锡（Matte Tin, Sn）的 JEDEC 无铅标志
	*	表示无铅封装。JEDEC 无铅标志（e3） 标示于此种封装的外包装上。

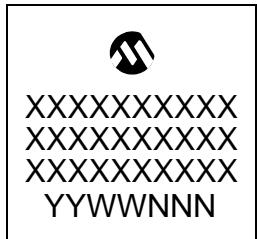
注：Microchip 元器件编号如果无法在同一行内完整标注，将换行标出，因此会限制表示客户指定信息的可用字符数。

# PIC24FJ64GB004 系列

---

---

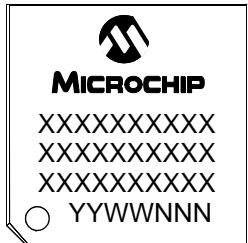
44 引脚 QFN



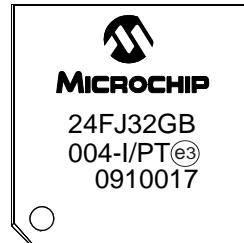
示例



44 引脚 TQFP



示例

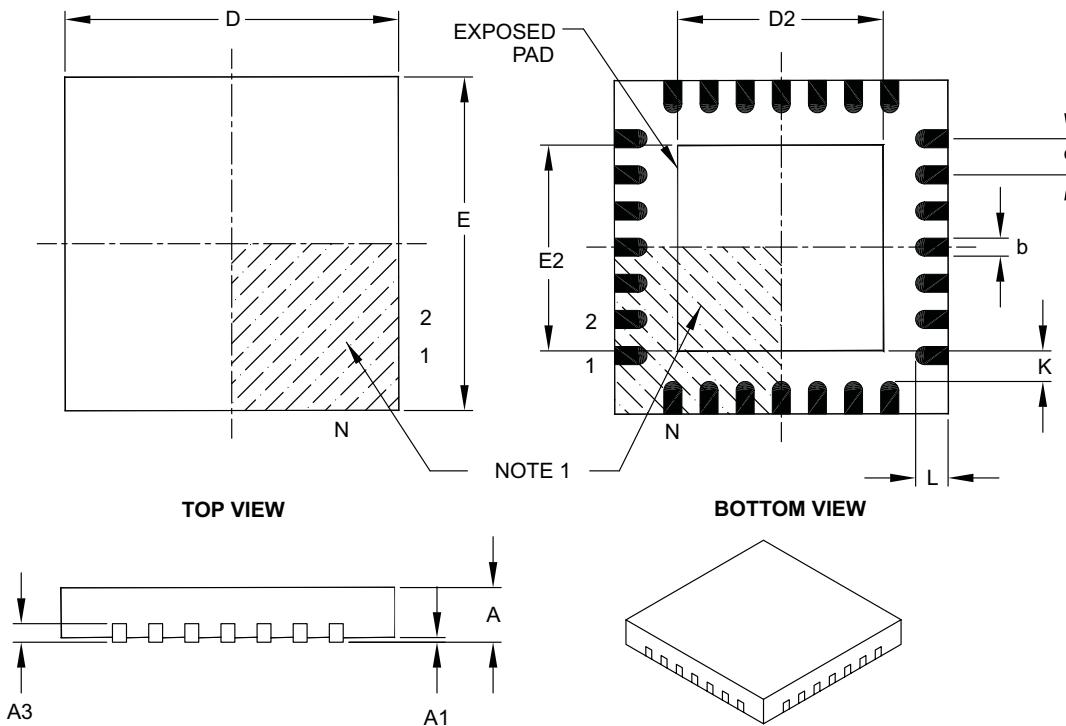


## 30.2 封装详细信息

以下部分将介绍各种封装技术的细节。

### 28 引脚塑封四方扁平无引脚封装 (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:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

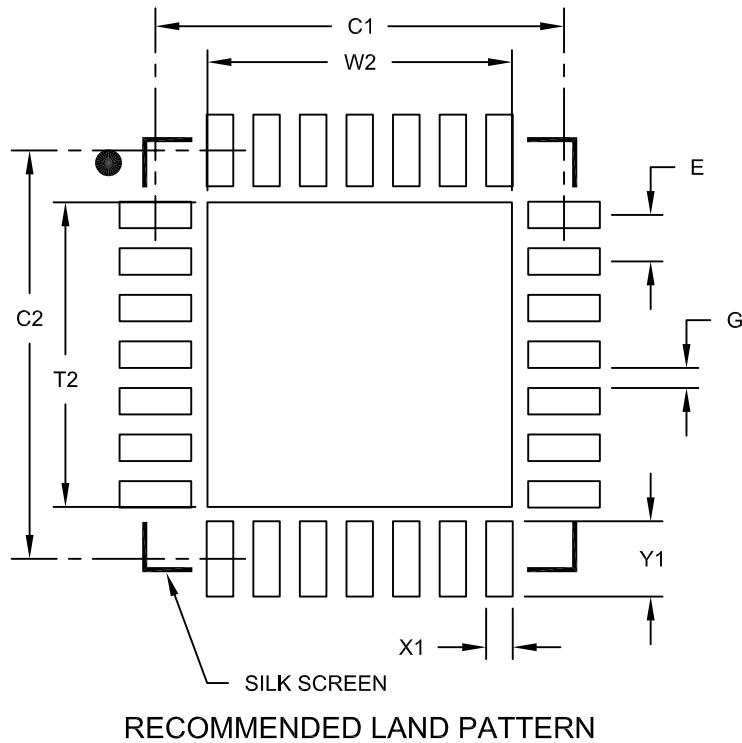
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

# PIC24FJ64GB004 系列

## 28 引脚塑封四方扁平无引脚封装 (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。



Dimension	Limits	Units		
		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:

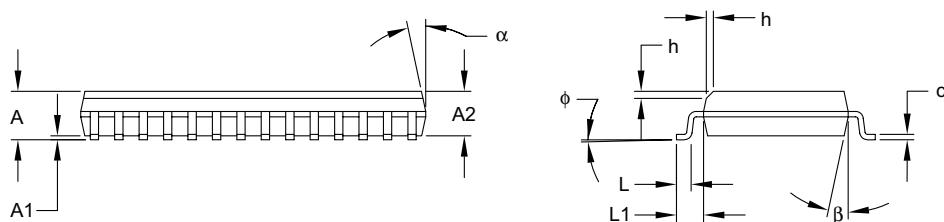
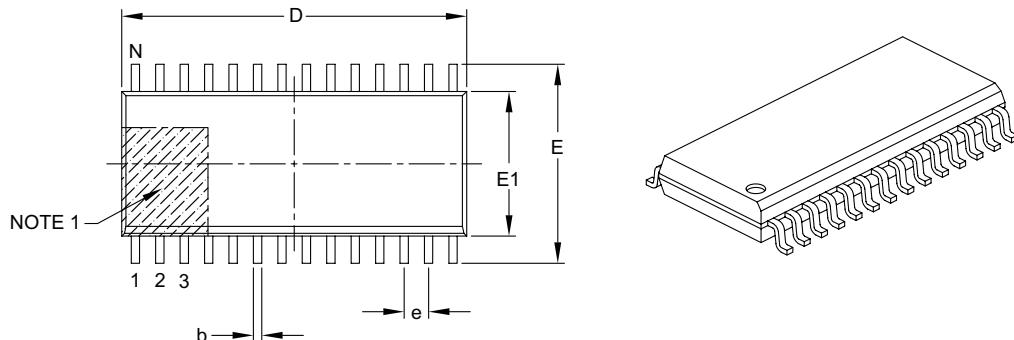
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2105A

## 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	—
Overall Width	E	10.30 BSC	
Molded Package Width	E1	7.50 BSC	
Overall Length	D	17.90 BSC	
Chamfer (optional)	h	0.25	—
Foot Length	L	0.40	—
Footprint	L1	1.40 REF	
Foot Angle Top	ϕ	0°	—
Lead Thickness	c	0.18	—
Lead Width	b	0.31	—
Mold Draft Angle Top	α	5°	—
Mold Draft Angle Bottom	β	5°	—

### Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.15 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

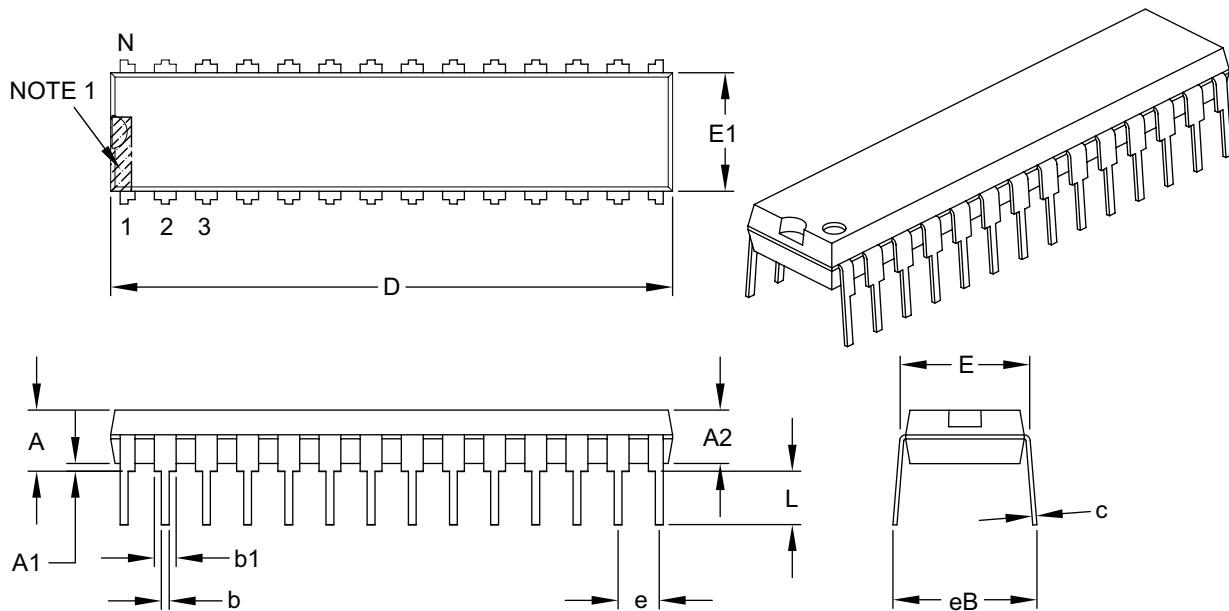
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-052B

# PIC24FJ64GB004 系列

## 28 引脚窄条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。



Dimension Limits	Units			INCHES		
	MIN	NOM	MAX	MIN	NOM	MAX
Number of Pins	N		28			
Pitch	e		.100 BSC			
Top to Seating Plane	A	—	—	.290	.310	.335
Molded Package Thickness	A2	.120	.135	.150		
Base to Seating Plane	A1	.015	—	—		
Shoulder to Shoulder Width	E	.240	.285	.295		
Molded Package Width	E1	.110	.130	.150		
Overall Length	D	1.345	1.365	1.400		
Tip to Seating Plane	L	.008	.010	.015		
Lead Thickness	c	.040	.050	.070		
Upper Lead Width	b1	.014	.018	.022		
Lower Lead Width	b	—	—	.430		
Overall Row Spacing §	eB					

### Notes:

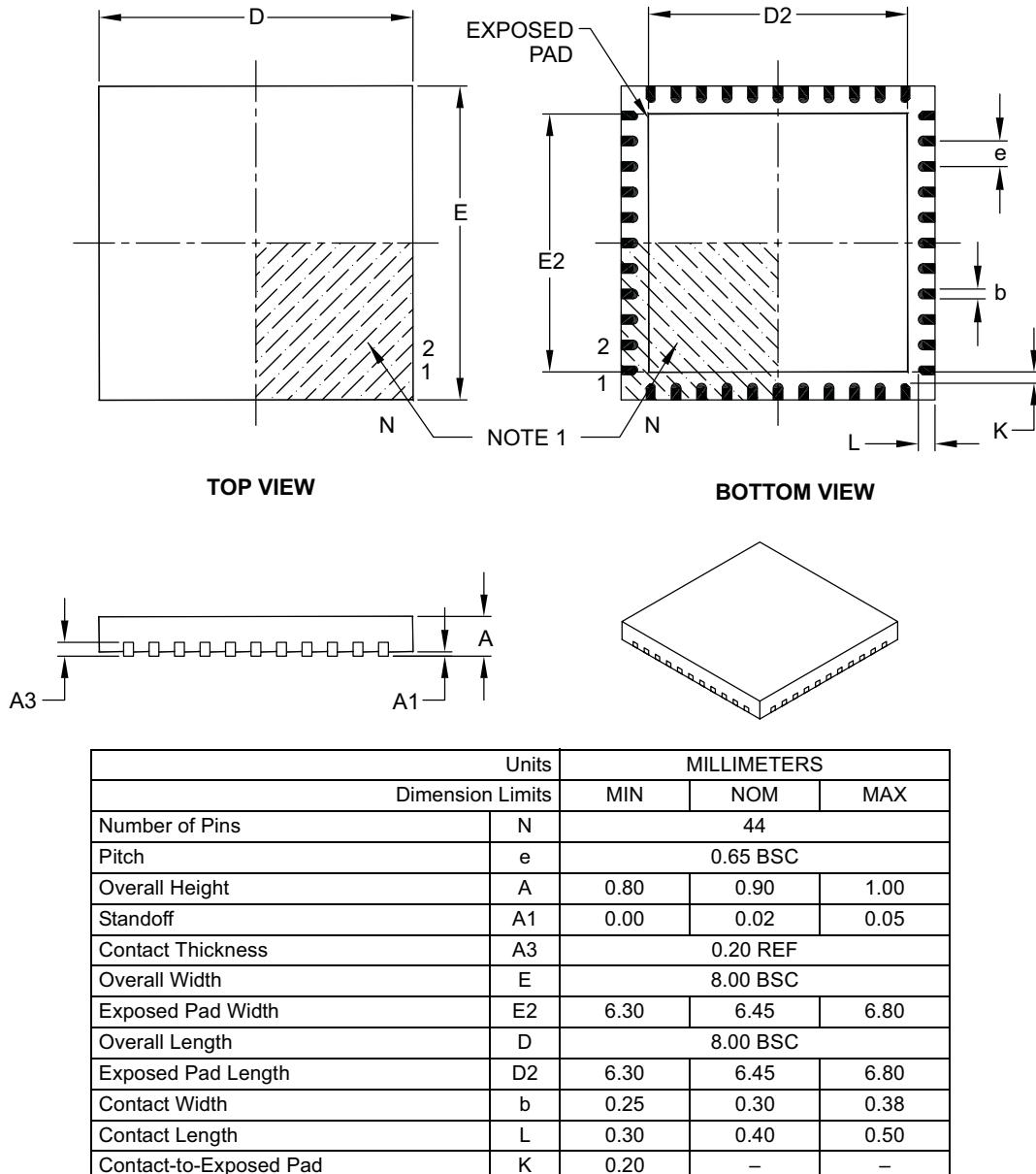
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

## 44 引脚塑封四方扁平无引脚封装 (ML) —— 主体 8x8 mm[QFN]

注： 最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



**Notes:**

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

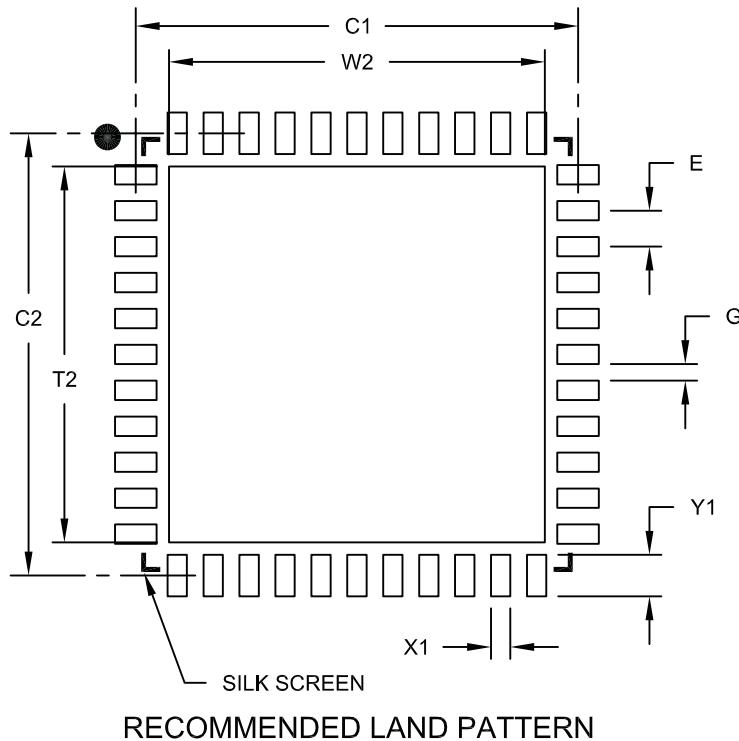
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-103B

# PIC24FJ64GB004 系列

## 44 引脚塑封四方扁平无引脚封装 (ML) —— 主体 8x8 mm[QFN]

注： 最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



Dimension	Limits	UNITS MILLIMETERS		
		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W2			6.80
Optional Center Pad Length	T2			6.80
Contact Pad Spacing	C1		8.00	
Contact Pad Spacing	C2		8.00	
Contact Pad Width (X44)	X1			0.35
Contact Pad Length (X44)	Y1			0.80
Distance Between Pads	G	0.25		

Notes:

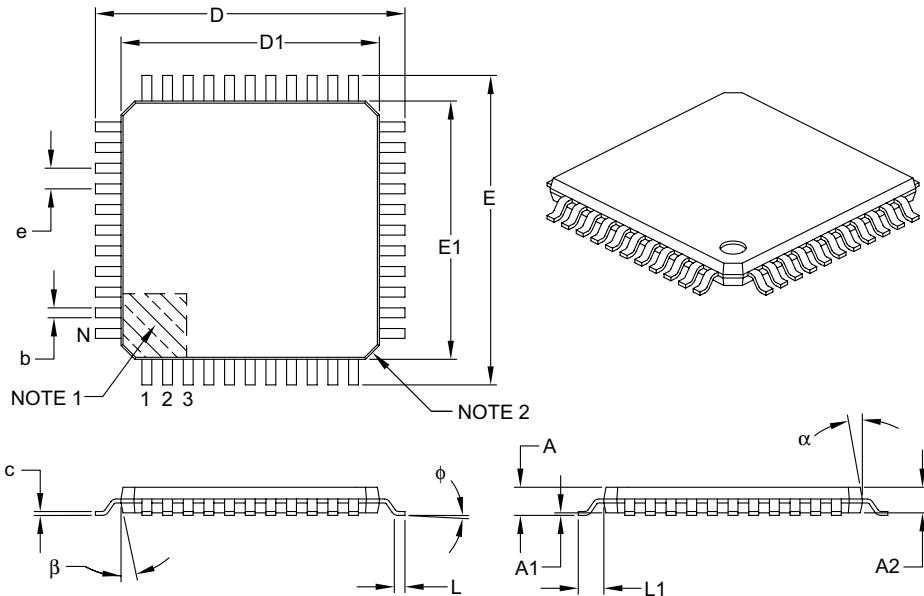
- Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2103A

## 44 引脚塑封薄型四方扁平封装 (PT)——主体 10x10x1 mm , 2.00 mm[TQFP]

注： 最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads		N		
Lead Pitch		e		
Overall Height		A		
Molded Package Thickness		A2		
Standoff		A1		
Foot Length		L		
Footprint		L1		
Foot Angle		0°		
Overall Width		E		
Overall Length		D		
Molded Package Width		E1		
Molded Package Length		D1		
Lead Thickness		c		
Lead Width		b		
Mold Draft Angle Top		11°		
Mold Draft Angle Bottom		11°		
		12°		
		12°		
		13°		

**Notes:**

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

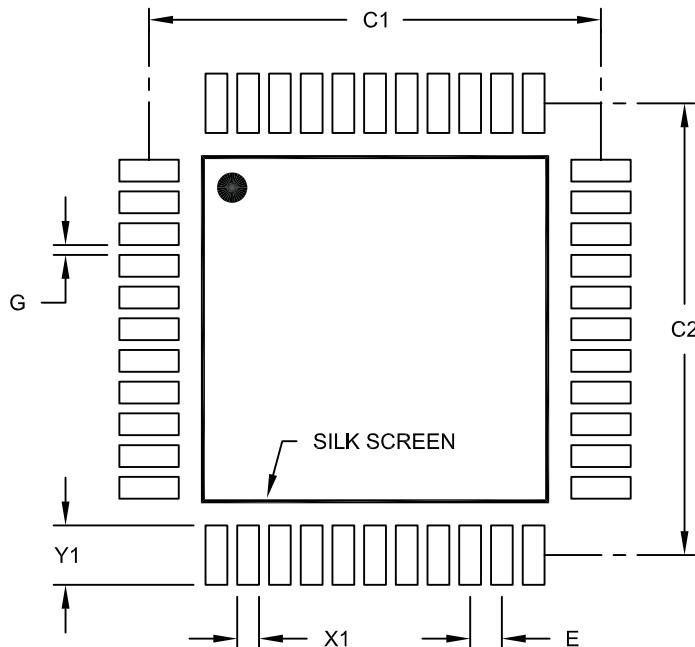
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-076B

# PIC24FJ64GB004 系列

44 引脚塑封薄型四方扁平封装 (PT) —— 主体 10x10x1 mm , 2.00 mm[TQFP]

注： 最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E	0.80 BSC		
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width (X44)	X1			0.55
Contact Pad Length (X44)	Y1			1.50
Distance Between Pads	G	0.25		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2076A

## 附录 A : 版本历史

### 版本 A (2009 年 4 月)

PIC24FJ64GB004 系列器件的初始数据手册。

### 版本 B (2009 年 7 月)

在表 4-4 中删除了未实现的 CNPD1 和 CNPD2 寄存器。  
纠正了同一表中 CNPU1 和 CNPU2 寄存器的地址。

更新了表 6-2 (复位延时时间), 在所有表条目中都增加了 TRSRT。

更新了寄存器 7-35 (INTTREG), 使之更具有说明性。

更新了 第 9.2.4 节 “深度休眠模式”, 添加了系列特定信息, 还进一步讨论了进入深度休眠模式的特殊情形。

更新了 第 29.1 节 “直流特性”, 具体如下:

- 在表 29-4、表 29-5、表 29-6 和表 29-7 中添加了最大值。
- 更新了表 29-3 和表 29-8 中的规范。
- 添加了新的表 29-11 (比较器规范) 和表 29-12 (比较器参考电压规范), 并对这两个表后面的所有表格重新编号。
- 删除了表 29-6、表 29-7 和表 29-12 中的冗余或过时的规范。

更新了 第 29.2 节 “交流特性和时序参数”, 具体如下:

- 更新了表 29-17 和表 29-19 中的规范。
- 添加了新的表 29-21 (复位、上电延时定时器和欠压复位时序要求), 并对该表后面的所有表格重新编号。

还对整个文档的排版进行了细微修订。

### 版本 C (2009 年 10 月)

纠正了第 10.3 节 “输入电平变化通知” 中 ICN 输入的数目, 以及去掉了“下拉”文字。

删除了表 10-2 中有关 “Timer 1 时钟输入”的项, 更新了第 10.4.2 节 “可用外设”。

对第 29.1 节 “直流特性” 作了如下更新:

- 向表 29-4 和表 29-5 添加了新的参数, 指定 0.5 MIPS 运行速度下的 IDD 和 I<sub>IDLE</sub>。
- 修改了 1 MIPS 和 4 MIPS 下的最大 IDD 参数, 更新了表 29-4。

修改了  $\Delta$  IPD 参数的编号 (32 kHz, SOSCEL = 11), 将其从 DC62n 更改为 DC63n。

# **PIC24FJ64GB004 系列**

---

---

注：

## 索引

### A

- A/D 转换器
  - 传递函数 ..... 264
  - 模拟输入模型 ..... 263

### B

- Block Diagrams
- 版本历史 ..... 329
- 备用中断向量表 (AIVT) ..... 65
- 比较器参考电压 ..... 269
- 并行主端口。见 PMP。 ..... 227
- 变更通知客户服务 ..... 336

### C

- CPU
  - 编程模型 ..... 21
  - 控制寄存器 ..... 24
  - 内核寄存器 ..... 23
  - 时钟机制 ..... 104
  - 算术逻辑单元 (ALU) ..... 25

### CRC

- 典型操作 ..... 251
- 寄存器 ..... 251
- 用户接口
  - 多项式 ..... 250
  - 数据 ..... 250

### CTMU

- 测量电容 ..... 271
- 测量时间 ..... 272
- 脉冲产生和延时 ..... 272

### C 编译器

- MPLAB C18 ..... 290
- 参考时钟输出 ..... 111
- 产品标识体系 ..... 338

### 程序存储器

- 程序空间可视性 ..... 49
- 存储器映射 ..... 27
- 地址构成 ..... 46
- 地址空间 ..... 27
- 构成 ..... 28
- 闪存配置字 ..... 28
- 使用表指令访问 ..... 48

### 程序校验

- 285

### 程序空间可视性 (PSV)

- 49

充电时间测量单元。见 CTMU。

串行外设接口。见 SPI。

### D

- DISVREG 引脚 ..... 283
- 打盹模式 ..... 121
- 代码保护
  - 保护配置寄存器 ..... 286
  - 代码段保护 ..... 286
  - 配置选项 ..... 286

### 代码示例

- I/O 端口读 / 写 ..... 124
- PWRSAV 指令语法 ..... 113
- 编程闪存的一个单字，“C” ..... 57
- 编程闪存的一个单字，汇编 ..... 57
- 擦除程序存储器块，“C” ..... 55
- 擦除程序存储器块，汇编 ..... 54
- 基本时钟切换序列 ..... 109
- 将 RTCWREN 位置 1 ..... 238

- 配置 UART1 输入和输出功能 (PPS) ..... 129
- 启动编程序列，“C” ..... 56
- 启动编程序列，汇编 ..... 56
- 装载写缓冲器，“C” ..... 56
- 装载写缓冲器，汇编 ..... 55

### 电气特性

- V/F 图 ..... 302
- 温度条件 ..... 302
- 绝对最大值 ..... 301
- 读者反馈表 ..... 337

### F

- 封装 ..... 319
- 标识 ..... 319
- 详细信息 ..... 321

### 复位

- BOR (欠压复位) ..... 59
- CM (配置不匹配复位) ..... 59
- IOPUWR (非法操作码复位) ..... 59
- MCLR (引脚复位) ..... 59
- POR (上电复位) ..... 59
- RCON 标志操作 ..... 61
- SFR 状态 ..... 63
- SWR (RESET 指令) ..... 59
- TRAPR (陷阱冲突复位) ..... 59
- UWR (未初始化的 W 寄存器复位) ..... 59
- WDT (看门狗定时器复位) ..... 59
- 器件时间 ..... 61
- 时钟源选择 ..... 61
- 延时时间 ..... 62

### G

#### 公式

- A/D 转换时钟周期 ..... 263
- $\text{BRGH} = 0$  时的 UART 波特率 ..... 186
- $\text{BRGH} = 1$  时的 UART 波特率 ..... 186
- 波特率重载值 ..... 179
- 计算 PWM 周期 ..... 161
- 计算最大 PWM 分辨率 ..... 161
- 器件速率和 SPI 时钟速度之间的关系 ..... 176

### H

#### 汇编器

- MPASM 汇编器 ..... 290

### I

#### I/O 端口

- 并行 (PIO) ..... 123
- 模拟端口引脚配置 ..... 124
- 漏极开路配置 ..... 124
- 上拉和下拉 ..... 125
- 输入电压注意事项 ..... 124
- 输入电平变化通知 ..... 125
- 外设引脚选择 ..... 125

### I<sup>2</sup>C

- 保留地址 ..... 179
- 从地址屏蔽 ..... 179
- 时钟速率 ..... 179
- 用作总线主器件时设置波特率 ..... 179
- 作为主器件在单主器件环境中通信 ..... 177

### J

- JTAG 接口 ..... 287
- 寄存器

# PIC24FJ64GB004 系列

AD1CHS (A/D 输入选择) .....	260
AD1CON1 (A/D 控制 1) .....	257
AD1CON2 (A/D 控制 2) .....	258
AD1CON3 (A/D 控制 3) .....	259
AD1CSSL (A/D 输入扫描选择) .....	262
AD1PCFG (A/D 端口配置) .....	261
ALCFGRPT (闹钟配置) .....	241
ALMINSEC (闹钟分钟和秒值) .....	245
ALMTHDY (闹钟月和日值) .....	244
ALWDHRS (闹钟星期和小时值) .....	244
BdnSTAT 样本 (缓冲器描述符 n 状态, CPU 模式) .....	201
BdnSTAT 样本 (缓冲器描述符 n 状态, USB 模式) .....	200
CLKDIV (时钟分频器) .....	107
CMSTAT (比较器模块状态) .....	268
CMxCON (比较器 x 控制) .....	267
CORCON (CPU 控制) .....	25
CORCON (CPU 内核控制) .....	69
CRCCON1 (CRC 控制 1) .....	252
CRCCON2 (CRC 控制 2) .....	253
CRCXORH (CRC 多项式异或操作, 高字节) .....	254
CRCXORL (CRC 多项式异或操作, 低字节) .....	253
CTMUCON (CTMU 控制) .....	273
CTMUICON (CTMU 电流控制) .....	274
CVRCON (比较器参考电压控制) .....	270
CW1 (闪存配置字 1) .....	276
CW2 (闪存配置字 2) .....	278
CW3 (闪存配置字 3) .....	280
DEVID (器件 ID) .....	282
DEVREV (器件版本) .....	282
DSCON (深度休眠控制) .....	119
DSWAKE (深度休眠唤醒源) .....	120
I2CxCON (I2Cx 控制) .....	180
I2CxMSK (I2C 从模式地址屏蔽) .....	184
I2CxSTAT (I2Cx 状态) .....	182
ICxCON1 (输入捕捉 x 控制 1) .....	155
ICxCON2 (输入捕捉 x 控制 2) .....	156
IEC0 (中断允许控制 0) .....	78
IEC1 (中断允许控制 1) .....	79
IEC2 (中断允许控制 2) .....	80
IEC3 (中断允许控制 3) .....	81
IEC4 (中断允许控制 4) .....	82
IEC5 (中断允许控制 5) .....	83
IFS0 (中断标志状态 0) .....	72
IFS1 (中断标志状态 1) .....	73
IFS2 (中断标志状态 2) .....	74
IFS3 (中断标志状态 3) .....	75
IFS4 (中断标志状态 4) .....	76
IFS5 (中断标志状态 5) .....	77
INTCON1 (中断控制 1) .....	70
INTCON2 (中断控制 2) .....	71
INTTREG (中断控制和状态) .....	101
IPC0 (中断优先级控制 0) .....	84
IPC1 (中断优先级控制 1) .....	85
IPC10 (中断优先级控制 10) .....	94
IPC11 (中断优先级控制 11) .....	95
IPC12 (中断优先级控制 12) .....	96
IPC15 (中断优先级控制 15) .....	97
IPC16 (中断优先级控制 16) .....	98
IPC18 (中断优先级控制 18) .....	99
IPC19 (中断优先级控制 19) .....	99
IPC2 (中断优先级控制 2) .....	86
IPC21 (中断优先级控制 21) .....	100
IPC3 (中断优先级控制 3) .....	87
IPC4 (中断优先级控制 4) .....	88
IPC5 (中断优先级控制 5) .....	89
IPC6 (中断优先级控制 6) .....	90
IPC7 (中断优先级控制 7) .....	91
IPC8 (中断优先级控制寄存器 8) .....	92
IPC9 (中断优先级控制 9) .....	93
MINSEC (RTCC 分钟和秒值) .....	243
MTHDY (RTCC 月和日值) .....	242
NVMCON (闪存控制) .....	53
OCxCON1 (输出比较 x 控制 1) .....	163
OCxCON2 (输出比较 x 控制 2) .....	165
OSCCON (振荡器控制) .....	105
OSCTUN (FRC 振荡器调节) .....	108
PADCFG1 (填充配置控制) .....	233, 240
PMADDR (并行端口地址) .....	231
PMAEN (并行端口使能) .....	231
PMCON (并行端口控制) .....	228
PMMODE (并行端口模式) .....	230
PMSTAT (并行端口状态) .....	232
RCFGCAL (RTCC 校准和配置) .....	239
RCON (复位控制) .....	60
REFOCON (参考振荡器控制) .....	112
RPINR0 (外设引脚选择输入 0) .....	130
RPINR1 (外设引脚选择输入 1) .....	130
RPINR11 (外设引脚选择输入 11) .....	133
RPINR18 (外设引脚选择输入 18) .....	134
RPINR19 (外设引脚选择输入 19) .....	134
RPINR20 (外设引脚选择输入 20) .....	135
RPINR21 (外设引脚选择输入 21) .....	135
RPINR22 (外设引脚选择输入 22) .....	136
RPINR23 (外设引脚选择输入 23) .....	136
RPINR3 (外设引脚选择输入 3) .....	131
RPINR4 (外设引脚选择输入 4) .....	131
RPINR7 (外设引脚选择输入 7) .....	132
RPINR8 (外设引脚选择输入 8) .....	132
RPINR9 (外设引脚选择输入 9) .....	133
RPOR0 (外设引脚选择输出 0) .....	137
RPOR1 (外设引脚选择输出 1) .....	137
RPOR10 (外设引脚选择输出 10) .....	142
RPOR11 (外设引脚选择输出 11) .....	142
RPOR12 (外设引脚选择输出 12) .....	143
RPOR2 (外设引脚选择输出 2) .....	138
RPOR3 (外设引脚选择输出 3) .....	138
RPOR4 (外设引脚选择输出 4) .....	139
RPOR5 (外设引脚选择输出 5) .....	139
RPOR6 (外设引脚选择输出 6) .....	140
RPOR7 (外设引脚选择输出 7) .....	140
RPOR8 (外设引脚选择输出 8) .....	141
RPOR9 (外设引脚选择输出 9) .....	141
SPIxCON1 (SPIx 控制 1) .....	172
SPIxCON2 (SPIx 控制 2) .....	173
SPIxSTAT (SPIx 状态和控制) .....	170
SR (ALU 状态) .....	24
SR (ALU 状态, 在 CPU 中) .....	69
T1CON (Timer1 门控) .....	146
TxCON (Timer2 和 Timer4 控制) .....	150
TyCON (Timer3 和 Timer5 控制) .....	151
U1ADDR (USB 地址) .....	215
U1CNFG1 (USB 配置 1) .....	216
U1CNFG2 (USB 配置 2) .....	217
U1CON (USB 控制, 器件模式) .....	213
U1CON (USB 控制, 主机模式) .....	214
U1EIE (USB 错误中断允许) .....	224
U1EIR (USB 错误中断状态) .....	223
U1EPn (USB 端点 n 控制) .....	225
U1IE (USB 中断允许) .....	222
U1IR (USB 中断状态, 仅器件模式) .....	220

U1IR ( USB 中断状态 , 仅主机模式 ) .....	221	客户支持 .....	336
U1OTGCON ( USB OTG 控制 ) .....	210	客户通知服务 .....	336
U1OTGIE ( USB OTG 中断允许 , 仅主机模式 ) .....	219	框图	
U1OTGIR ( USB OTG 中断状态 , 仅主机模式 ) .....	218	10 位高速 A/D 转换器 .....	256
U1OTGSTAT ( USB OTG 状态 , 仅主机模式 ) .....	209	16 位 Timer1 模块 .....	145
U1PWMCON ( USB VBUS PWM 发生器控制 ) .....	226	16 位同步 Timer2 和 Timer4 .....	149
U1PWRC ( USB 电源控制 ) .....	211	16 位异步 Timer3 和 Timer5 .....	149
U1SOF ( USB OTG 标记起始阈值 , 仅主机模式 ) .....	216	32 位 Timer2/3 和 Timer4/5 .....	148
U1STAT ( USB 状态 ) .....	212	8 位地址和数据复用的应用示例 .....	236
U1TOK ( USB 标记 , 仅主机模式 ) .....	215	CALL 堆栈帧 .....	46
UxMODE ( UARTx 模式 ) .....	188	CPU 编程模型 .....	23
UxSTA ( UARTx 状态和控制 ) .....	190	CRC 模块 .....	249
WKDYHR ( RTCC 星期和小时值 ) .....	243	CRC 移位引擎 .....	249
YEAR ( RTCC 年值 ) .....	242	I <sup>2</sup> C 模块 .....	178
<b>寄存器映射</b>		LCD 控制示例 , 字节模式 .....	236
A/D 转换器 .....	39	PIC24F CPU 内核 .....	22
CPU 内核 .....	31	PIC24FJ64GB004 系列 ( 通用 ) .....	10
CRC .....	42	PMP 模块一览 .....	227
CTMU .....	39	PSV 操作 .....	49
I <sup>2</sup> C .....	37	RTCC .....	237
ICN .....	32	SPIx 模块 ( 标准框图 ) .....	168
NVM .....	44	SPIx 模块 ( 增强型模式 ) .....	169
PMD .....	45	SPI 主 / 从连接 ( 标准模式 ) .....	174
PORTA .....	38	SPI 主 / 从连接 ( 增强型缓冲器模式 ) .....	174
PORTB .....	38	SPI 从器件、帧从器件连接 .....	175
PORTC .....	38	SPI 从器件、帧主器件连接 .....	175
RTCC .....	42	SPI 主器件、帧从器件连接 .....	175
SPI .....	38	SPI 主器件、帧主器件连接 .....	175
UART .....	37	UART ( 简化 ) .....	185
USB OTG .....	40	USB OTG	
比较器 .....	42	仅自供电 .....	195
并行主 / 从端口 .....	41	仅总线电源 .....	195
定时器 .....	34	双电源 .....	195
深度休眠 .....	44	USB OTG 模块 .....	194
输出比较 .....	36	USB OTG 中断逻辑 .....	202
输入捕捉 .....	35	USB PLL .....	110
外设引脚选择 .....	43	比较器参考电压 .....	269
系统 .....	44	表寄存器寻址 .....	51
引脚配置 .....	39	并行 EEPROM 示例 , 16 位数据 .....	236
中断控制器 .....	33	并行 EEPROM 示例 , 8 位数据 .....	236
<b>基本连接要求</b> .....	17	部分复用寻址应用示例 .....	235
<b>交流特性</b>		产生脉冲延时的 CTMU 典型连接和内部配置 .....	272
A/D 规范 .....	316	传统 PSP 示例 .....	234
ADC 转换要求 .....	317	单独的比较器配置 .....	266
CLKO 和 I/O 时序 .....	315	电容测量的 CTMU 连接和内部配置 .....	271
PLL 时钟规范 .....	314	端点缓冲模式下的 BDT 映射 .....	198
复位、上电延时定时器和欠压复位时序 .....	315	访问程序空间内的数据的地址生成方式 .....	47
内部 RC 振荡器规范 .....	314	复位系统 .....	59
内部 RC 振荡器精度 .....	314	复用寻址应用示例 .....	235
时序规范的负载条件和要求 .....	312	共用 I/O 端口的结构 .....	123
输出引脚上的容性负载要求 .....	312	仅总线电源 .....	195
外部时钟时序 .....	313	看门狗定时器 ( WDT ) .....	285
温度和电压规范 .....	312	可寻址的 PSP 示例 .....	234
<b>基于指令的节能模式</b> .....	113	片内稳压器的连接 .....	283
深度休眠 .....	114	三比较器模块 .....	265
休眠 .....	113	时间测量的 CTMU 典型连接和内部配置 .....	272
空闲 .....	114	使用表指令访问程序存储器 .....	48
<b>节能特性</b> .....	113	输出比较 ( 16 位模式 ) .....	158
时钟频率和时钟切换 .....	113	输出比较 ( 双缓冲 , 16 位 PWM 模式 ) .....	160
<b>K</b>		输入捕捉 .....	153
开发支持 .....	289	系统时钟 .....	103
看门狗定时器 ( WDT ) .....	284	主模式下的部分复用寻址 .....	235
窗口操作 .....	285	主模式下的解复用寻址 .....	234
控制寄存器 .....	285	主模式下的完全复用寻址 .....	235
勘误表 .....	6		

# PIC24FJ64GB004 系列

## M

Microchip 因特网网站 .....	336
MPLAB ASM30 汇编器、链接器和库管理器 .....	290
MPLAB ICE 2000 高性能通用在线仿真器 .....	291
MPLAB 集成开发环境软件 .....	289
MPLINK 目标链接器 /MPLIB 目标库管理器 .....	290
脉宽调制 ( PWM ) 模式 .....	160
脉宽调制。见 PWM.	

## N

Near 数据空间 .....	30
内部集成电路。见 I <sup>2</sup> C。	177
内核特性 .....	7

## P

PICSTART 2 开发编程器 .....	292
PICSTART Plus 开发编程器 .....	292
PWM	
占空比和周期 .....	161
配置位 .....	275
片内稳压器 .....	283
待机模式 .....	284
跟踪模式 .....	283
和 BOR .....	284
和 POR .....	283
上电要求 .....	284

## R

复位、上电延时定时器和欠压复位时序要求 .....	315
RTCC	
寄存器映射 .....	238
校准 .....	246
闹钟配置 .....	246
闹钟屏蔽设置 ( 图 ) .....	247
写锁定 .....	238
选择时钟源 .....	238
源时钟 .....	237
软件堆栈 .....	46
软件模拟器 ( MPLAB SIM ) .....	290

## S

SFR 空间 .....	30
SPI	
三比较器 .....	265
闪存程序存储器 .....	51
JTAG 操作 .....	52
RTSP 操作 .....	52
编程单字 .....	57
编程算法 .....	54
和表指令 .....	51
增强型 ICSP 操作 .....	52
闪存配置字 .....	28, 275 – 281
深度休眠 BOR ( DSBOR ) .....	63
深度休眠看门狗定时器 ( DSWDT ) .....	285
示例	
波特率误差计算 ( BRGH = 0 ) .....	186
时序图	
CLKO 和 I/O 时序 .....	315
外部时钟 .....	313
使用专用定时器的输出比较 .....	157
使用专用定时器的输入捕捉 .....	153
输出比较	
32 位模式 .....	157
操作 .....	159
单指令周期的分辨率 .....	162

同步和触发模式 .....	157
---------------	-----

数据存储器	
Near 数据空间 .....	30
SFR 空间 .....	30
存储器映射 .....	29
地址空间 .....	29
空间构成 .....	30
软件堆栈 .....	46

输入捕捉	
32 位模式 .....	154
操作 .....	154
同步和触发模式 .....	153

## T

Timer1 .....	145
Timer2/3 和 Timer4/5 .....	147
特性 .....	8
通用串行总线。见 USB OTG.	
通用串行总线	
不同缓冲模式下缓冲区描述符的分配 .....	199
中断和 USB 事务 .....	203

通用异步收发器。见 UART.	
-----------------	--

## U

UART .....	185
IrDA 支持 .....	187
UxCTS 和 UxRTS 控制引脚的操作 .....	187
按 8 位或 9 位数据模式接收 .....	187
波特率发生器 ( BRG ) .....	186
发送	
8 位数据模式 .....	187
9 位数据模式 .....	187
间隔或同步序列 .....	187
USB On-The-Go ( OTG ) .....	8
USB OTG	
DMA 接口 .....	199
OTG 工作 .....	206
Vbus 电压生成 .....	197
缓冲器描述符和 BDT .....	198
寄存器 .....	208 – 226
器件模式工作 .....	203
硬件配置	
器件模式 .....	195
收发器功耗要求 .....	197
Vbus 电压生成 .....	197
外部接口 .....	197
主机和 OTG 模式 .....	196
中断 .....	202
主机模式工作 .....	204

## V

VDDCORE/VCAP 引脚 .....	283
-----------------------	-----

## W

WWW 在线技术支持 .....	6
WWW 地址 .....	336
外设模块禁止位 .....	121
外设使能位 .....	121
外设引脚选择 ( PPS ) .....	125
可用外设和引脚 .....	125
配置控制 .....	128
使用注意事项 .....	129
输入映射 .....	126
输出映射 .....	127
外设优先级 .....	125

未使用的 I/O .....	20
<b>Y</b>	
引脚配置说明 .....	11
引脚框图 .....	2, 3, 4
引脚	
ICSP .....	19
ICSP 工作期间的模拟和数字引脚配置 .....	20
电源 .....	18
外部振荡器 .....	20
稳压器引脚 .....	19
主复位 ( MCLR ) .....	18
因特网地址 .....	336
有选择的外设模块控制 .....	121
<b>Z</b>	
振荡器配置	
USB 工作 .....	110
特殊注意事项 .....	110
上电复位时的初始配置 .....	104
时钟切换 .....	108
序列 .....	109
时钟选择 .....	104
辅助振荡器 ( SOSC ) .....	111
指令集	
操作码说明中使用的符号 .....	294
概述 .....	295
汇总 .....	293
直流特性	
I/O 引脚输入规范 .....	309
I/O 引脚输出规范 .....	310
比较器规范 .....	311
比较器电压规范 .....	311
程序存储器 .....	310
掉电外设模块电流 ( IPD ) .....	307
基本掉电电流 .....	306
工作电流 .....	304
空闲电流 .....	305
内部稳压器规范 .....	311
温度和电压规范 .....	303
中断	
和复位过程 .....	65
控制和状态寄存器 .....	68
设置和服务过程 .....	102
实现的向量 .....	67
陷阱向量 .....	66
向量表 .....	66
中断向量表 ( IVT ) .....	65
中断服务程序 ( ISR ) .....	102

# **PIC24FJ64GB004 系列**

---

---

注:

## MICROCHIP 网站

Microchip 网站 ([www.microchip.com](http://www.microchip.com)) 为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的因特网浏览器即可访问。网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和样本程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及存档软件
- **一般技术支持**——常见问题 (FAQ)、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

## 变更通知客户服务

Microchip 的变更通知客户服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 [www.microchip.com](http://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。

# PIC24FJ64GB004 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： PIC24FJ64GB004 系列

文献编号：

DS39940C\_CN

问题：

1. 本文档中哪些部分最有特色？

\_\_\_\_\_

2. 本文档是否满足了您的软硬件开发要求？如何满足的？

\_\_\_\_\_

3. 您认为本文档的组织结构便于理解吗？如果不便于理解，那么问题何在？

\_\_\_\_\_

4. 您认为本文档应该添加哪些内容以改善其结构和主题？

\_\_\_\_\_

5. 您认为本文档中可以删减哪些内容，而又不会影响整体使用效果？

\_\_\_\_\_

6. 本文档中是否存在错误或误导信息？如果存在，请指出是什么信息及其具体页数。

\_\_\_\_\_

7. 您认为本文档还有哪些方面有待改进？

\_\_\_\_\_

## 产品标识体系

欲订货或获取价格、交货等信息，请与我公司生产厂或各销售办事处联系。

PIC 24 FJ 64 GB0 04 I - 1 / PT - XXX		示例：
Microchip 商标	—	a) PIC24FJ64GB004-I/PT : 具有 USB On-The-Go 功能、64 KB 程序存储器、44 引脚、工业级温度以及 TQFP 封装的 PIC24F 器件。
架构	—	b) PIC24FJ32GB002-I/ML : 具有 USB On-The-Go 功能、32 KB 程序存储器、28 引脚、工业级温度以及 QFN 封装的 PIC24F 器件。
闪存系列	—	
程序存储器容量 (KB)	—	
产品组	—	
引脚数	—	
卷带标志 (如果适用)	—	
温度范围	—	
封装	—	
模式	—	
架构	24 = 不带 DSP 的 16 位改进型哈佛架构	
闪存系列	FJ = 闪存程序存储器	
产品组	GA0 = 具有 USB On-The-Go 功能的通用单片机	
引脚数	02 = 28 引脚 04 = 44 引脚	
温度范围	I = -40°C 到 +85°C (工业级)	
封装	ML = 28 引脚 (6x6 mm) 或 44 引脚 (8x8 mm) QFN (四方扁平) PT = 44 引脚 (10x10x1 mm) TQFP (薄型四方扁平) SO = 28 引脚 (7.50 mm 宽) SOIC (小外形) SP = 28 引脚 (300 mil) SPDIP (窄条塑封双列直插)	
模式	3 位 QTP、SQTP、代码或特殊要求 (空白为其他情况) ES = 工程样片	



# MICROCHIP

## 全球销售及服务网点

### 美洲

#### 公司总部 Corporate Office

2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 1-480-792-7200  
Fax: 1-480-792-7277

技术支持：  
<http://support.microchip.com>  
网址：[www.microchip.com](http://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

#### 克里夫兰 Cleveland

Independence, OH  
Tel: 216-447-0464  
Fax: 216-447-0643

#### 达拉斯 Dallas

Addison, TX  
Tel: 1-972-818-7423  
Fax: 1-972-818-2924

#### 底特律 Detroit

Farmington Hills, MI  
Tel: 1-248-538-2250  
Fax: 1-248-538-2260

#### 科科莫 Kokomo

Kokomo, IN  
Tel: 1-765-864-8360  
Fax: 1-765-864-8387

#### 洛杉矶 Los Angeles

Mission Viejo, CA  
Tel: 1-949-462-9523  
Fax: 1-949-462-9608

#### 圣克拉拉 Santa Clara

Santa Clara, CA  
Tel: 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: 886-3-6578-300  
Fax: 886-3-6578-370

### 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: 91-80-3090-4444  
Fax: 91-80-3090-4080

#### 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