



**PIC18F23K20/24K20/25K20/26K20/
43K20/44K20/45K20/46K20**

数据手册

采用 nanoWatt XLP 技术的
28/40/44 引脚闪存单片机

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、PIC³² 徽标、rfPIC 和 UNI/O 均为 Microchip Technology Inc. 在美国和其他国家或地区的注册商标。

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

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

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

在此提及的所有其他商标均为各持有公司所有。

© 2010, Microchip Technology Inc. 版权所有。

ISBN: 978-1-60932-122-2

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洲和印度的设计中心均通过了 ISO/TS-16949:2002 认证。公司在 PIC[®] MCU 与 dsPIC[®] DSC、KEELOQ[®] 跳码器件、串行 EEPROM、单片机外设、非易失性存储器和模拟产品方面的质量体系流程均符合 ISO/TS-16949:2002。此外，Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。



MICROCHIP

PIC18F2XK20/4XK20

采用 nanoWatt XLP 技术的 28/40/44 引脚闪存单片机

高性能 RISC CPU:

- 针对 C 编译器优化的架构:
 - 为优化可重入代码而设计的可选扩展指令集
- 最大 1024 字节的数据 EEPROM
- 可寻址最大 64 KB 的线性程序存储空间
- 可寻址最大 3936 字节的线性数据存储空间
- 最高 16 MIPS 的工作速度
- 16 位宽指令, 8 位宽数据总线
- 中断优先级
- 31 级深、软件可访问的硬件堆栈
- 8 x 8 单周期硬件乘法器

灵活的振荡器结构:

- 高精度 16 MHz 内部振荡器模块:
 - 出厂时精度已校准到 $\pm 1\%$
 - 可通过软件选择频率范围: 31 kHz 到 16 MHz
 - 使用 PLL 时可达 64 MHz 性能——无需外部元件
- 4 种晶振模式, 频率最高为 64 MHz
- 两种外部时钟模式, 频率最高为 64 MHz
- 4 倍频锁相环 (Phase Lock Loop, PLL)
- 辅助振荡器使用 Timer1 (工作频率为 32 kHz)
- 故障保护时钟监视器 (Fail-Safe Clock Monitor, FSCM):
 - 当外设时钟停止时可使器件安全关闭
- 双速振荡器启动

单片机特性:

- 工作电压范围: 1.8V 至 3.6V
- 可在软件控制下自编程
- 可编程 16 级高 / 低压检测 (High/Low-Voltage Detection, HLVD) 模块:
 - 高 / 低压检测中断
- 可编程欠压复位 (Brown-out Reset, BOR):
 - 带软件使能选项
- 扩展型看门狗定时器 (Watchdog Timer, WDT):
 - 可编程周期从 4 ms 到 131s
- 通过两个引脚进行单电源 3V 在线串行编程 (In-Circuit Serial Programming™, ICSP™)
- 通过两个引脚进行在线调试 (In-Circuit Debug, ICD)

采用 nanoWatt XLP 的超低功耗管理:

- 休眠模式: 1.8V 时 $< 100 \text{ nA}$
- 看门狗定时器: 1.8V 时 $< 800 \text{ nA}$
- Timer1 振荡器: 32 kHz、1.8V 时 $< 800 \text{ nA}$

模拟特性:

- 模数转换器 (Analog-to-Digital Converter, ADC) 模块:
 - 10 位分辨率, 13 路外部通道
 - 自动采集功能
 - 可在休眠模式下进行转换
 - 1.2V 固定参考电压 (Fixed Voltage Reference, FVR) 通道
 - 独立的输入多路选择
- 模拟比较器模块:
 - 两个轨到轨模拟比较器
 - 独立的输入多路选择
- 参考电压 (CVREF) 模块
 - 可编程 (VDD 的 %), 16 阶
 - 使用 VREF 引脚的两个 16 级电压范围

外设特点:

- 最多 35 个 I/O 引脚加上 1 个仅用作输入的引脚:
 - 高灌 / 拉电流: 25 mA/25 mA
 - 3 个可编程外部中断
 - 4 个可编程电平变化中断
 - 8 个可编程弱上拉
 - 可编程斜率
- 捕捉/比较/PWM (Capture/Compare/PWM, CCP) 模块
- 增强型 CCP (ECCP) 模块:
 - 1、2 或 4 路 PWM 输出
 - 可选择的极性
 - 可编程的死区
 - 自动关闭和自动重启
- 主同步串行口 (Master Synchronous Serial Port, MSSP) 模块
 - 3 线 SPI (支持所有 4 种模式)
 - I²C™ 主 / 从模式 (带地址掩码)
- 增强型通用同步 / 异步收发器 (Enhanced Universal Synchronous Asynchronous Receiver Transmitter, EUSART) 模块:
 - 支持 RS-485、RS-232 和 LIN
 - 使用内部振荡器的 RS-232 工作
 - 接收到间隔字符时自动唤醒
 - 自动波特率检测

PIC18F2XK20/4XK20

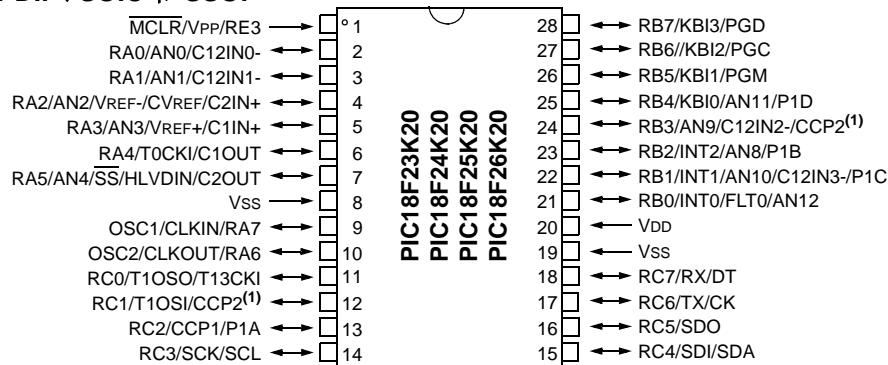
器件	程序存储器		数据存储器		I/O ⁽¹⁾	10 位 A/D (通道) ⁽²⁾	CCP/ ECCP (PWM)	MSSP		USART	比较器	8/16 位 定时器
	闪存 (字节)	单字 指令数	SRAM (字节)	EEPROM (字节)				SPI	主 I ² C TM			
PIC18F23K20	8K	4096	512	256	25	11	1/1	有	有	1	2	1/3
PIC18F24K20	16K	8192	768	256	25	11	1/1	有	有	1	2	1/3
PIC18F25K20	32K	16384	1536	256	25	11	1/1	有	有	1	2	1/3
PIC18F26K20	64K	32768	3936	1024	25	11	1/1	有	有	1	2	1/3
PIC18F43K20	8K	4096	512	256	36	14	1/1	有	有	1	2	1/3
PIC18F44K20	16K	8192	768	256	36	14	1/1	有	有	1	2	1/3
PIC18F45K20	32K	16384	1536	256	36	14	1/1	有	有	1	2	1/3
PIC18F46K20	64K	32768	3936	1024	36	14	1/1	有	有	1	2	1/3

注 1: 其中的一个引脚是仅能用作输入的引脚。

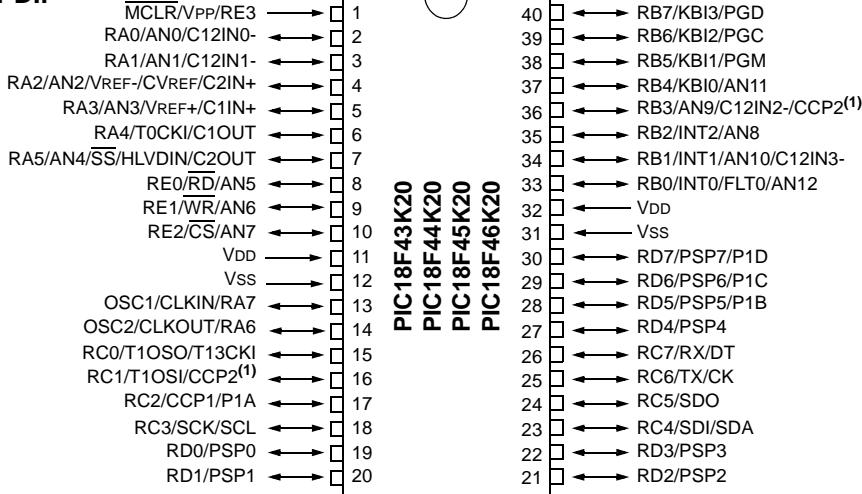
2: 通道数包括内部的固定参考电压通道。

引脚图

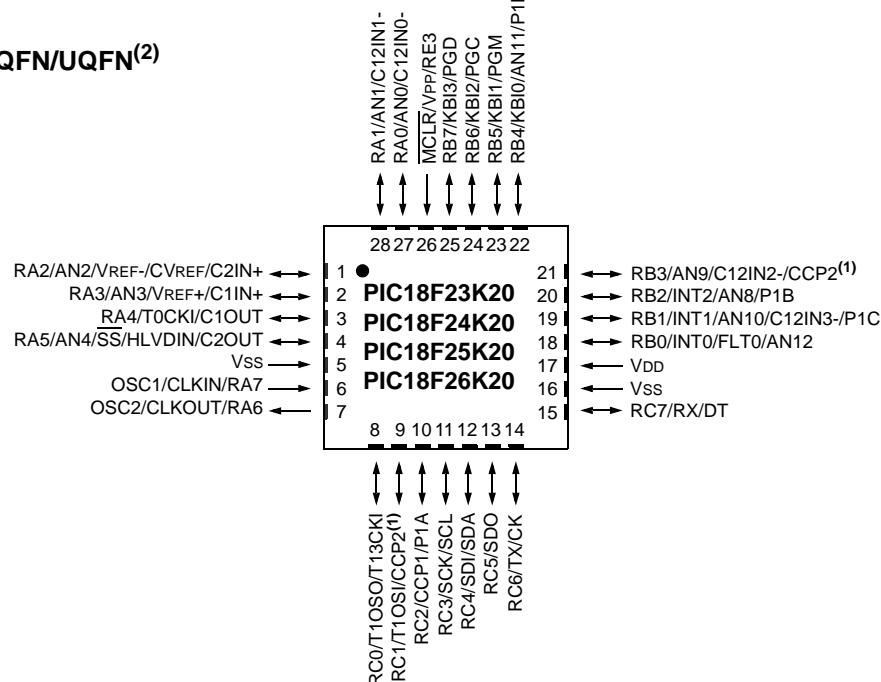
28 引脚 PDIP、SOIC 和 SSOP



40 引脚 PDIP



28 引脚 QFN/UQFN⁽²⁾



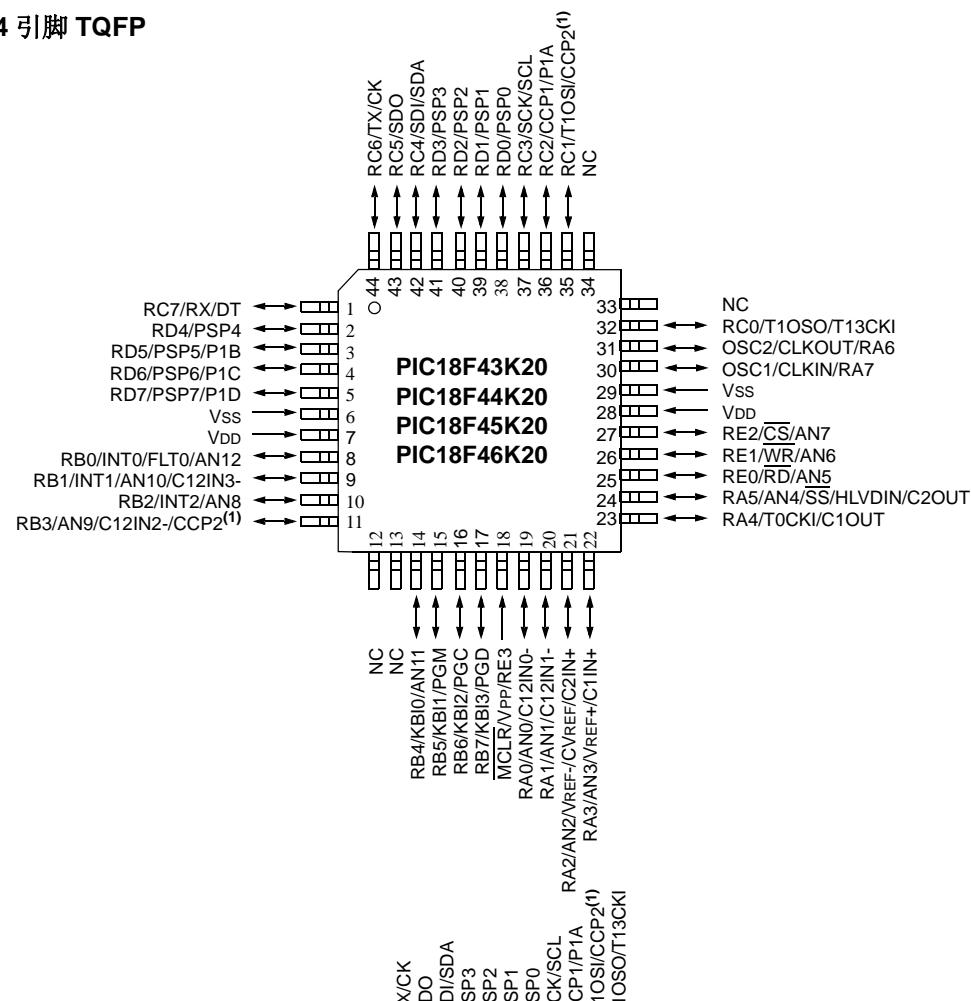
注 1: RB3 是与 CCP2 复用的备用引脚。

2: UQFN 封装仅适用于 PIC18F23K20。

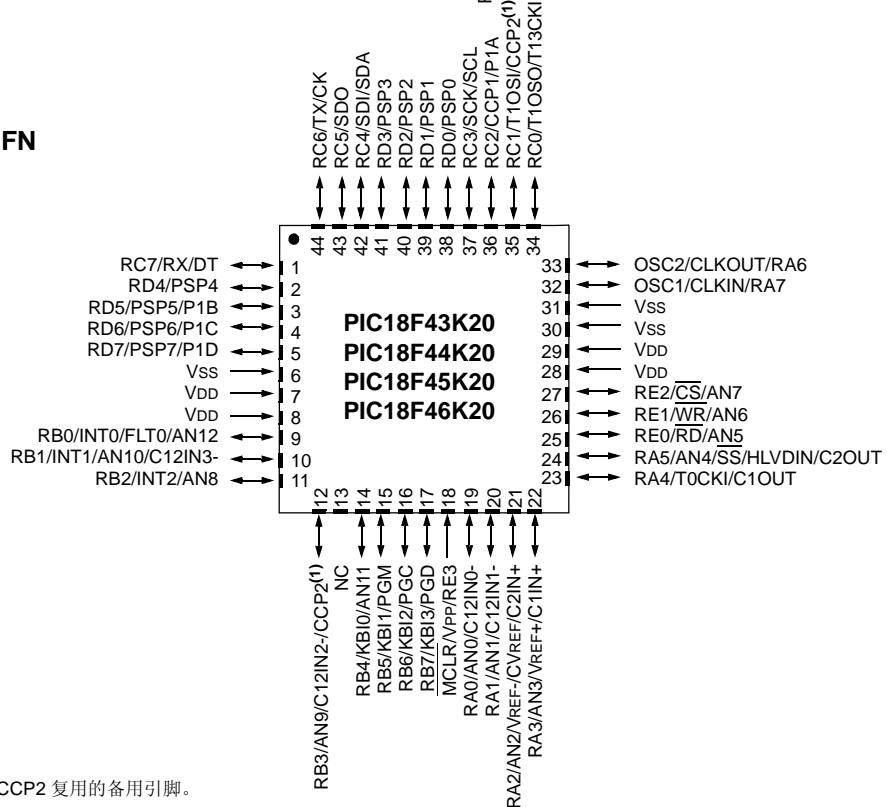
PIC18F2XK20/4XK20

引脚图 (续)

44 引脚 TQFP



44 引脚 QFN



注 1: RB3 是与 CCP2 复用的备用引脚。

表1： PIC18F4XK20 引脚汇总

DIL 引脚	TQFP 引脚	QFN 引脚	I/O	模拟	比较器	参考电压	ECCP	EUSART	MSSP	定时器	从端口	中断	上拉	基本功能
2	19	19	RA0	AN0	C12IN0-	—	—	—	—	—	—	—	—	—
3	20	20	RA1	AN1	C12IN1-	—	—	—	—	—	—	—	—	—
4	21	21	RA2	AN2	C2IN+	VREF-/CVREF	—	—	—	—	—	—	—	—
5	22	22	RA3	AN3	C1IN+	VREF+	—	—	—	—	—	—	—	—
6	23	23	RA4	—	C1OUT	—	—	—	—	T0CKI	—	—	—	—
7	24	24	RA5	AN4	C2OUT	HLVDIN	—	—	SS	—	—	—	—	—
14	31	33	RA6	—	—	—	—	—	—	—	—	—	—	OSC2/CLKOUT
13	30	32	RA7	—	—	—	—	—	—	—	—	—	—	OSC1/CLKIN
33	8	9	RB0	AN12	—	—	FLT0	—	—	—	—	INT0	有	—
34	9	10	RB1	AN10	C12IN3-	—	—	—	—	—	—	INT1	有	—
35	10	11	RB2	AN8	—	—	—	—	—	—	—	INT2	有	—
36	11	12	RB3	AN9	C12IN2-	—	CCP2 ⁽¹⁾	—	—	—	—	—	有	—
37	14	14	RB4	AN11	—	—	—	—	—	—	—	KBI0	有	—
38	15	15	RB5	—	—	—	—	—	—	—	—	KBI1	有	PGM
39	16	16	RB6	—	—	—	—	—	—	—	—	KBI2	有	PGC
40	17	17	RB7	—	—	—	—	—	—	—	—	KBI3	有	PGD
15	32	34	RC0	—	—	—	—	—	—	T1OSO/T13CKI	—	—	—	—
16	35	35	RC1	—	—	—	CCP2 ⁽²⁾	—	—	T1OSI	—	—	—	—
17	36	36	RC2	—	—	—	CCP1/P1A	—	—	—	—	—	—	—
18	37	37	RC3	—	—	—	—	—	SCK/SCL	—	—	—	—	—
23	42	42	RC4	—	—	—	—	—	SDI/SDA	—	—	—	—	—
24	43	43	RC5	—	—	—	—	—	SDO	—	—	—	—	—
25	44	44	RC6	—	—	—	—	TX/CK	—	—	—	—	—	—
26	1	1	RC7	—	—	—	—	RX/DT	—	—	—	—	—	—
19	38	38	RD0	—	—	—	—	—	—	—	PSP0	—	—	—
20	39	39	RD1	—	—	—	—	—	—	—	PSP1	—	—	—
21	40	40	RD2	—	—	—	—	—	—	—	PSP2	—	—	—
22	41	41	RD3	—	—	—	—	—	—	—	PSP3	—	—	—
27	2	2	RD4	—	—	—	—	—	—	—	PSP4	—	—	—
28	3	3	RD5	—	—	—	P1B	—	—	—	PSP5	—	—	—
29	4	4	RD6	—	—	—	P1C	—	—	—	PSP6	—	—	—
30	5	5	RD7	—	—	—	P1D	—	—	—	PSP7	—	—	—
8	25	25	RE0	AN5	—	—	—	—	—	RD	—	—	—	—
9	26	26	RE1	AN6	—	—	—	—	—	WR	—	—	—	—
10	27	27	RE2	AN7	—	—	—	—	—	CS	—	—	—	—
1	18	18	RE3 ⁽³⁾	—	—	—	—	—	—	—	—	—	—	MCLR/VPP
11	7	7	—	—	—	—	—	—	—	—	—	—	—	VDD
32	28	28	—	—	—	—	—	—	—	—	—	—	—	VDD
12	6	6	—	—	—	—	—	—	—	—	—	—	—	VSS
31	29	30	—	—	—	—	—	—	—	—	—	—	—	VSS
—	NC	8	—	—	—	—	—	—	—	—	—	—	—	VDD
—	NC	29	—	—	—	—	—	—	—	—	—	—	—	VDD
—	NC	31	—	—	—	—	—	—	—	—	—	—	—	VSS

注 1: 当 CONFIG3H<0> = 0 时, CCP2 与 RB3 复用

2: 当 CONFIG3H<0> = 1 时, CCP2 与 RC1 复用

3: 仅用作输入

PIC18F2XK20/4XK20

表 2: PIC18F2XK20 引脚汇总

DIL 引脚	QUAD 引脚	I/O	模拟	比较器	参考电压	ECCP	EUSART	MSSP	定时器	从端口	中断	上拉	基本功能
2	27	RA0	AN0	C12IN0-									
3	28	RA1	AN1	C12IN1-									
4	1	RA2	AN2	C2IN+	VREF-/CVREF								
5	2	RA3	AN3	C1IN+	VREF+								
6	3	RA4		C1OUT					T0CKI				
7	4	RA5	AN4	C2OUT	HLVDIN			SS					
10	7	RA6											OSC2/CLKOUT
9	6	RA7											OSC1/CLKIN
21	18	RB0	AN12			FLT0					INT0	有	
22	19	RB1	AN10	C12IN3-		P1C					INT1	有	
23	20	RB2	AN8			P1B					INT2	有	
24	21	RB3	AN9	C12IN2-		CCP2 ⁽¹⁾						有	
25	22	RB4	AN11			P1D					KBI0	有	
26	23	RB5									KBI1	有	PGM
27	24	RB6									KBI2	有	PGC
28	25	RB7									KBI3	有	PGD
11	8	RC0							T1OSO/T13CKI				
12	9	RC1				CCP2 ⁽²⁾			T1OSI				
13	10	RC2				CCP1/P1A							
14	11	RC3						SCK/SCL					
15	12	RC4						SDI/SDA					
16	13	RC5						SDO					
17	14	RC6					TX/CK						
18	15	RC7					RX/DT						
1	26	RE3 ⁽³⁾											MCLR/VPP
8	5												VSS
19	16												VSS
20	17												VDD

注 1: 当 CONFIG3H<0> = 0 时, CCP2 与 RB3 复用

2: 当 CONFIG3H<0> = 1 时, CCP2 与 RC1 复用

3: 仅用作输入

目录

1.0 器件概述	11
2.0 振荡器模块（带故障保护时钟监视器）	27
3.0 功耗管理模式	43
4.0 复位	51
5.0 存储器构成	65
6.0 闪存程序存储器	89
7.0 数据 EEPROM 存储器	99
8.0 8 x 8 硬件乘法器	105
9.0 中断	107
10.0 I/O 端口	121
11.0 捕捉 / 比较 /PWM (CCP) 模块	143
12.0 Timer0 模块	155
13.0 Timer1 模块	159
14.0 Timer2 模块	167
15.0 Timer3 模块	169
16.0 增强型捕捉 / 比较 /PWM (ECCP) 模块	173
17.0 主同步串行口 (MSSP) 模块	193
18.0 增强型通用同步 / 异步收发器 (EUSART)	235
19.0 模数转换器 (ADC) 模块	263
20.0 比较器模块	277
21.0 参考电压	287
22.0 高 / 低压检测 (HLVD)	291
23.0 CPU 的特殊功能	297
24.0 指令集汇总	313
25.0 开发支持	363
26.0 电气特性	367
27.0 直流和交流特性图表	401
28.0 封装信息	423
附录 A: 版本历史	437
附录 B: 器件差异	438
索引	439
Microchip 网站	449
变更通知客户服务	449
客户支持	449
读者反馈表	450
产品标识体系	451

致客户

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

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

最新数据手册

欲获得本数据手册的最新版本，请查询我公司的网站：

<http://www.microchip.com>

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

勘误表

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

欲了解某一器件是否存在勘误表，请通过以下方式之一查询：

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

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

客户通知系统

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

1.0 器件概述

本文档包含以下器件的具体信息：

- PIC18F23K20
- PIC18F24K20
- PIC18F25K20
- PIC18F26K20
- PIC18F43K20
- PIC18F44K20
- PIC18F45K20
- PIC18F46K20

本系列具备所有 PIC18 单片机固有的优点，即以实惠的价格提供出色的计算性能，以及高耐用性的闪存程序存储器。除了这些优点之外，PIC18F2XK20/4XK20 系列还增强了器件设计，使得该系列单片机成为许多高性能、功耗敏感应用的明智选择。

1.1 新的内核特性

1.1.1 纳瓦技术

PIC18F2XK20/4XK20 系列的所有器件具有一系列能在工作时显著降低功耗的功能。主要包括以下几项：

- **备用运行模式：**通过将 Timer1 或内部振荡器模块作为控制器时钟源，可使代码执行时的功耗降低大约 90%。
- **多种空闲模式：**控制器还可在其 CPU 内核禁止而外设仍然工作的情况下工作。处于这些状态时，功耗能降得更低，最低只有正常工作时的 4%。
- **动态模式切换：**在器件工作期间可由用户代码调用功耗管理模式，允许用户将节能的理念融入到其应用软件设计中。
- **关键模块低功耗：**Timer1 和看门狗定时器模块的功耗需求可降至最小。具体数值请参见第 26.0 节“电气特性”。

1.1.2 多个振荡器选项和特性

PIC18F2XK20/4XK20 系列的所有器件可提供 10 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：

- 4 种晶振模式，使用晶振或陶瓷谐振器
- 两种外部时钟模式，提供使用两个引脚（振荡器输入引脚和四分频时钟输出引脚）或一个引脚（振荡器输入引脚，四分频时钟输出引脚重新分配为通用 I/O 引脚）的选项
- 两种外部 RC 振荡器模式，具有与外部时钟模式相同的引脚选项
- 内部振荡器模块包含一个 16 MHz HFINTOSC 振荡器和一个 31 kHz LFINTOSC 振荡器，它们共同提供 8 个可供用户选择的时钟频率，范围从 31 kHz 到 16 MHz。此选项可以空出两个振荡器引脚作为额外的通用 I/O 引脚。
- 一个锁相环（PLL）倍频器，可在高速晶振和内部振荡器模式下使用，可使时钟速度最高达到 64 MHz。PLL 和内部振荡器配合使用，可以向用户提供频率范围从 31 kHz 到 64 MHz 的时钟速度以供选择，而且不需要使用外部晶振或时钟电路。

除了可被用作时钟源外，内部振荡器模块还提供了一个稳定的参考源，增加了以下功能以使器件更安全地工作：

- **故障保护时钟监视器：**该功能的作用是持续监视主时钟源，将其与 LFINTOSC 提供的参考信号作比较。如果时钟发生了故障，单片机会将时钟源切换到内部振荡器模块，使器件可继续工作或安全地关闭应用。
- **双速启动：**该功能允许在上电复位或从休眠模式唤醒时将内部振荡器用作时钟源，直到主时钟源可用为止。

1.2 其他特殊功能

- 存储器耐用性：**程序存储器和数据 EEPROM 的闪存单元经评估，可以耐受数万次擦写，程序存储器最高可达 10,000 次，EEPROM 最高可达 100,000 次。在不刷新的情况下，数据保存时间保守地估计在 40 年以上。
- 自编程能力：**这些器件能在内部软件控制下写入各自的程序存储空间。通过使用受保护的引导块（位于程序存储器的顶端）中的自举程序，可创建能在现场进行自我更新的应用。
- 扩展指令集：**PIC18F2XK20/4XK20 系列在 PIC18 指令集的基础上进行了可选择的扩展，添加了 8 条新指令和一个变址寻址模式。此扩展可以使用一个器件配置选项使能，它是为优化可重入应用程序代码而特别设计的，这些代码原来是使用高级语言（如 C 语言）开发的。
- 增强型 CCP 模块：**在 PWM 模式下，该模块提供用于控制半桥或全桥驱动器的 1、2 或 4 路调制输出。其他特性包括：
 - 自动关闭，用于在中断或其他选定条件出现时关闭 PWM 输出
 - 自动重启，用于在条件清除后重新激活输出
 - 输出转向，用于有选择地使能 4 路输出中的一个或多个提供 PWM 信号
- 增强型可寻址 USART：**该串行通信模块可进行标准的 RS-232 通信并支持 LIN 总线协议。其他增强功能包括自动波特率检测和分辨率更高的 16 位波特率发生器。当单片机使用内部振荡器模块时，USART 为与外界对话的应用程序提供稳定的工作，而无需使用外部晶振也无需额外的功耗。
- 10 位 A/D 转换器：**该模块具备可编程采集时间，从而不必在选择通道和启动转换之间等待一个采样周期，因而减少了代码开销。
- 扩展型看门狗定时器 (WDT)：**该增强型版本加入了一个 16 位后分频器，可以提供在不同工作电压和温度下保持稳定的扩展超时范围。超时周期请参见第 26.0 节 “电气特性”。

1.3 系列中各产品的详细说明

PIC18F2XK20/4XK20 系列器件以 28 引脚和 40/44 引脚封装形式提供。图 1-1 和图 1-2 分别为这两类器件的框图。

这些器件在以下五个方面存在差异：

1. 闪存程序存储器（PIC18F23K20/43K20 器件为 8 KB，PIC18F24K20/44K20 器件为 16 KB，PIC18F25K20/45K20 器件为 32 KB，PIC18F26K20/46K20 器件为 64 KB）。
2. A/D 通道数（28 引脚器件有 11 路，40/44 引脚器件有 14 路）。
3. I/O 端口数（28 引脚器件有 3 个双向端口，40/44 引脚器件有 5 个双向端口）。
4. 并行从端口（仅在 40/44 引脚器件上存在）。

本系列器件的所有其他功能都是相同的。表 1-1 汇总了这些功能。

所有器件的引脚排列如引脚汇总表：表 1 和表 2 以及 I/O 引脚说明表：表 1-2 和表 1-3 所示。

表 1-1: 器件特性

特性	PIC18F23K20	PIC18F24K20	PIC18F25K20	PIC18F26K20	PIC18F43K20	PIC18F44K20	PIC18F45K20	PIC18F46K20
工作频率 ⁽²⁾	DC – 64 MHz	DC – 64 MHz	DC – 64 MHz	DC – 64 MHz	DC – 64 MHz	DC – 64 MHz	DC – 64 MHz	DC – 64 MHz
程序存储器 (字节)	8192	16384	32768	65536	8192	16384	32768	65536
程序存储器 (指令)	4096	8192	16384	32768	4096	8192	16384	32768
数据存储器 (字节)	512	768	1536	3936	512	768	1536	3936
数据 EEPROM 存储器 (字节)	256	256	256	1024	256	256	256	1024
中断源	19	19	19	19	20	20	20	20
I/O 端口	A、B、C 或 (E) ⁽¹⁾	A、B、C 或 (E) ⁽¹⁾	A、B、C 或 (E) ⁽¹⁾	A、B、C 或 (E) ⁽¹⁾	A、B、C、D 和 E	A、B、C、D 和 E	A、B、C、D 和 E	A、B、C、D 和 E
定时器	4	4	44		44		44	
捕捉 / 比较 / PWM 模块	1	1	1	1	1	1	1	1
增强型捕捉 / 比较 / PWM 模块	1	1	11		11		11	
串行通信	MSSP 和增强型 USART	MSSP 和增强型 USART	MSSP 和增强型 USART	MSSP 和增强型 USART	MSSP 和增强型 USART	MSSP 和增强型 USART	MSSP 和增强型 USART	MSSP 和增强型 USART
并行通信 (PSP)	无	无	无	无	有	有	有	有
10 位模数转换模块	1 路内部通道加上 10 路输入通道	1 路内部通道加上 10 路输入通道	1 路内部通道加上 10 路输入通道	1 路内部通道加上 10 路输入通道	1 路内部通道加上 13 路输入通道			
复位 (和延时)	POR、BOR、RESET 指令、堆栈满、堆栈下溢 (PWRT 和 OST)、MCLR (可选) 和 WDT	POR、BOR、RESET 指令、堆栈满、堆栈下溢 (PWRT 和 OST)、MCLR (可选) 和 WDT	POR、BOR、RESET 指令、堆栈满、堆栈下溢 (PWRT 和 OST)、MCLR (可选) 和 WDT	POR、BOR、RESET 指令、堆栈满、堆栈下溢 (PWRT 和 OST)、MCLR (可选) 和 WDT	POR、BOR、RESET 指令、堆栈满、堆栈下溢 (PWRT 和 OST)、MCLR (可选) 和 WDT	POR、BOR、RESET 指令、堆栈满、堆栈下溢 (PWRT 和 OST)、MCLR (可选) 和 WDT	POR、BOR、RESET 指令、堆栈满、堆栈下溢 (PWRT 和 OST)、MCLR (可选) 和 WDT	POR、BOR、RESET 指令、堆栈满、堆栈下溢 (PWRT 和 OST)、MCLR (可选) 和 WDT
可编程高 / 低压检测	有	有	有	有	有	有	有	有
可编程欠压复位	有	有	有	有	有	有	有	有
指令集	75 条指令；使能扩展指令集后总共为 83 条指令	75 条指令；使能扩展指令集后总共为 83 条指令	75 条指令；使能扩展指令集后总共为 83 条指令	75 条指令；使能扩展指令集后总共为 83 条指令	75 条指令；使能扩展指令集后总共为 83 条指令	75 条指令；使能扩展指令集后总共为 83 条指令	75 条指令；使能扩展指令集后总共为 83 条指令	75 条指令；使能扩展指令集后总共为 83 条指令
封装	28 引脚 PDIP 28 引脚 SOIC 28 引脚 QFN 28 引脚 SSOP 28 引脚 UQFN	28 引脚 PDIP 28 引脚 SOIC 28 引脚 QFN 28 引脚 SSOP	28 引脚 PDIP 28 引脚 SOIC 28 引脚 QFN 28 引脚 SSOP	28 引脚 PDIP 28 引脚 SOIC 28 引脚 QFN 28 引脚 SSOP	40 引脚 PDIP 44 引脚 QFN 44 引脚 TQFP			

注 1: PORTE 包含单个 RE3 只读位。LATE 和 TRISE 寄存器未实现。

2: 所示的频率范围仅适用于工业级范围器件。扩展级范围器件的最高频率为 48 MHz。

PIC18F2XK20/4XK20

图 1-1: PIC18F2XK20 (28 引脚) 框图

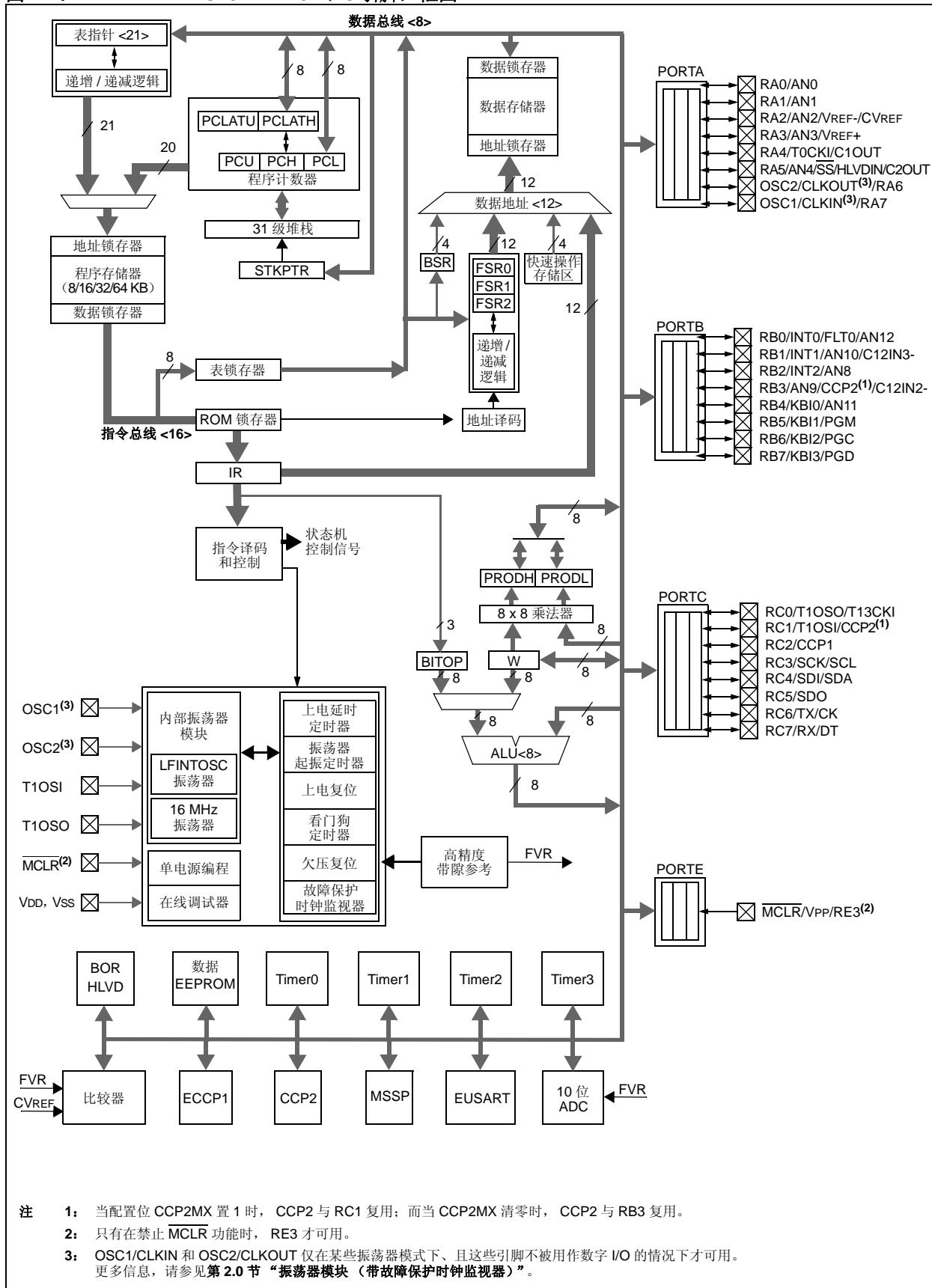
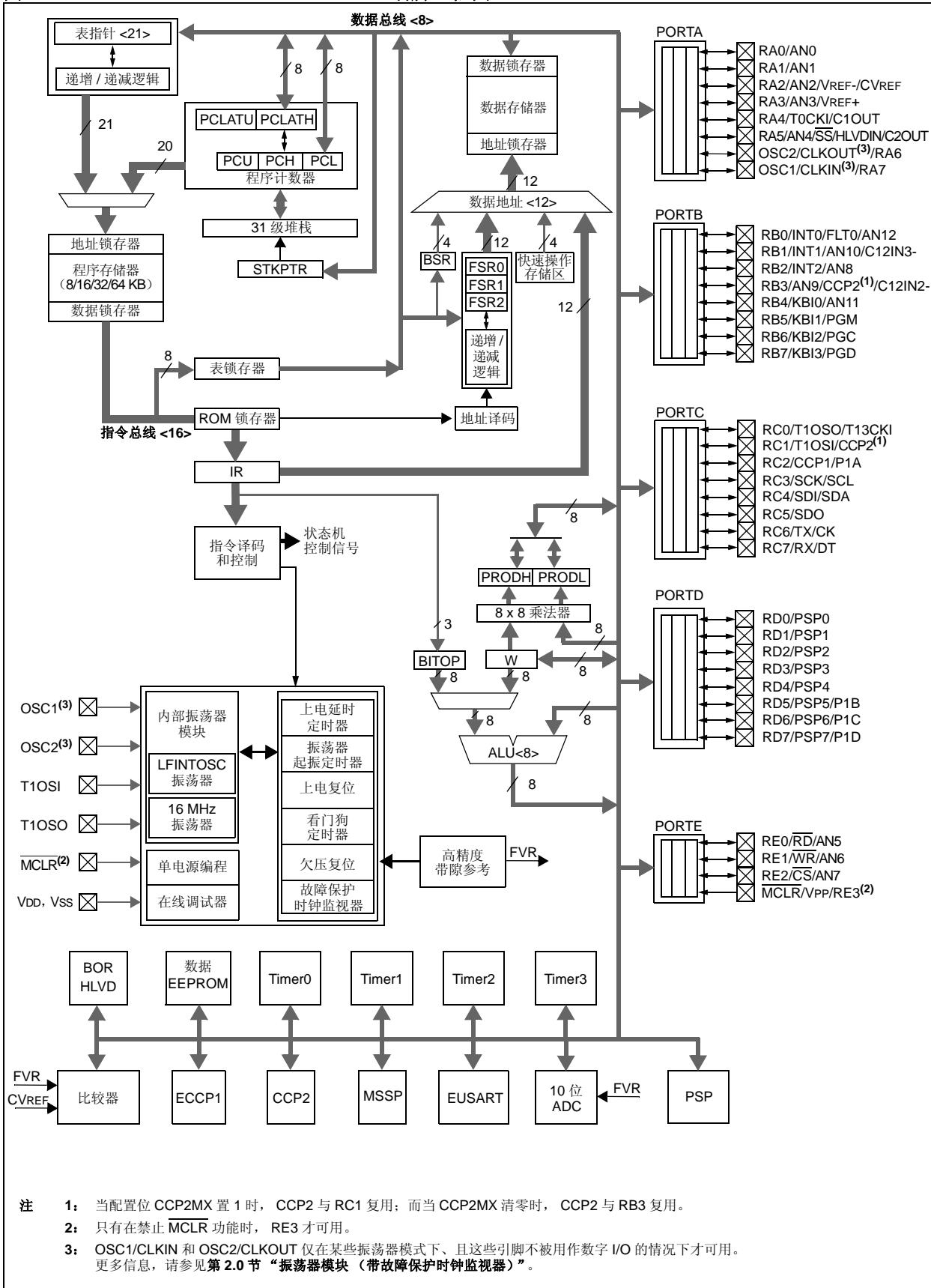


图 1-2: PIC18F4XK20 (40/44 引脚) 框图



PIC18F2XK20/4XK20

表 1-2: PIC18F2XK20 I/O 引脚说明

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP 和 SOIC	QFN			
MCLR/VPP/RE3 MCLR VPP RE3	1	26	I P I	ST ST	主复位（输入）或编程电压（输入） 低电平有效的主复位（器件复位）输入 编程电压输入 数字输入
OSC1/CLKIN/RA7 OSC1 CLKIN RA7	9	6	I I I/O	ST CMOS TTL	晶振或外部时钟输入 晶振输入或外部时钟源输入 配置为 RC 模式时为 ST 缓冲器输入；否则为 CMOS 缓冲器输入 外部时钟源输入。总是与 OSC1 引脚功能相关联。（见相关的 OSC1/CLKIN 和 OSC2/CLKOUT 引脚信息） 通用 I/O 引脚
OSC2/CLKOUT/RA6 OSC2 CLKOUT RA6	10	7	O O I/O	— — TTL	晶振或时钟输出 晶振输出。在晶振模式下，该引脚与晶振或谐振器相连 在 RC 模式下，OSC2 引脚输出 CLKOUT 信号，其频率是 OSC1 引脚上信号频率的 4 分频，该频率等于指令周期的倒数 通用 I/O 引脚

图注: TTL = TTL 兼容输入

ST = CMOS 电平的施密特触发器输入
O = 输出

CMOS = CMOS 兼容输入或输出

I = 输入
P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

表 1-2: PIC18F2XK20 I/O 引脚说明 (续)

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP 和 SOIC	QFN			
					PORTA 是双向 I/O 端口。
RA0/AN0/C12IN0- RA0 AN0 C12IN0-	2	27	I/O 	TTL 模拟 模拟	数字 I/O 模拟输入 0, ADC 通道 0 比较器 C1 和 C2 的反相输入
RA1/AN1/C12IN1- RA1 AN1 C12IN1-	3	28	I/O 	TTL 模拟 模拟	数字 I/O 模拟输入 1, ADC 通道 1 比较器 C1 和 C2 的反相输入
RA2/AN2/VREF-/CVREF/ C2IN+ RA2 AN2 VREF- CVREF C2IN+	4	1	I/O O 	TTL 模拟 模拟 模拟 模拟	数字 I/O 模拟输入 2, ADC 通道 2 A/D 参考电压 (低电压) 输入 比较器参考电压输出 比较器 C2 的同相输入
RA3/AN3/VREF+/C1IN+ RA3 AN3 VREF+ C1IN+	5	2	I/O 	TTL 模拟 模拟 模拟	数字 I/O 模拟输入 3, ADC 通道 3 A/D 参考电压 (高电压) 输入 比较器 C1 的同相输入
RA4/T0CKI/C1OUT RA4 T0CKI C1OUT	6	3	I/O O	ST ST CMOS	数字 I/O Timer0 外部时钟输入 比较器 C1 的输出
RA5/AN4/SS/HLDIN/ C2OUT RA5 AN4 SS HLDIN C2OUT	7	4	I/O O	TTL 模拟 TTL 模拟 CMOS	数字 I/O 模拟输入 4, ADC 通道 4 SPI 从选择输入 高 / 低压检测输入 比较器 C2 的输出
RA6					见 OSC2/CLKOUT/RA6 引脚信息
RA7					见 OSC1/CLKIN/RA7 引脚信息

图注: TTL = TTL 兼容输入

ST = CMOS 电平的施密特触发器输入

O = 输出

CMOS = CMOS 兼容输入或输出

| = 输入

P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

PIC18F2XK20/4XK20

表 1-2: PIC18F2XK20 I/O 引脚说明 (续)

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP 和 SOIC	QFN			
					PORTB 是双向 I/O 端口。 PORTB 的所有输入端都可软件编程为内部弱上拉。
RB0/INT0/FLT0/AN12 RB0 INT0 FLT0 AN12	21	18	I/O I I I	TTL ST ST 模拟	数字 I/O 外部中断 0 CCP1 的 PWM 故障输入 模拟输入 12, ADC 通道 12
RB1/INT1/AN10/C12IN3-/P1C RB1 INT1 AN10 C12IN3-P1C	22	19	I/O I I I O	TTL ST 模拟 模拟 CMOS	数字 I/O 外部中断 1 模拟输入 10, ADC 通道 10 比较器 C1 和 C2 的反相输入 增强型 CCP1 PWM 输出
RB2/INT2/AN8/P1B RB2 INT2 AN8 P1B	23	20	I/O I I O	TTL ST 模拟 CMOS	数字 I/O 外部中断 2 模拟输入 8, ADC 通道 8 增强型 CCP1 PWM 输出
RB3/AN9/C12IN2-/CCP2 RB3 AN9 C12IN2-CCP2 ⁽²⁾	24	21	I/O I I I/O	TTL 模拟 模拟 ST	数字 I/O 模拟输入 9, ADC 通道 9 比较器 C1 和 C2 的反相输入 捕捉 2 输入 / 比较 2 输出 / PWM2 输出
RB4/KBI0/AN11/P1D RB4 KBI0 AN11 P1D	25	22	I/O I I O	TTL TTL 模拟 CMOS	数字 I/O 电平变化中断引脚 模拟输入 11, ADC 通道 11 增强型 CCP1 PWM 输出
RB5/KBI1/PGM RB5 KBI1 PGM	26	23	I/O I I/O	TTL TTL ST	数字 I/O 电平变化中断引脚 低压 ICSP™ 编程使能引脚
RB6/KBI2/PGC RB6 KBI2 PGC	27	24	I/O I I/O	TTL TTL ST	数字 I/O 电平变化中断引脚 在线调试器和 ICSP™ 编程时钟引脚
RB7/KBI3/PGD RB7 KBI3 PGD	28	25	I/O I I/O	TTL TTL ST	数字 I/O 电平变化中断引脚 在线调试器和 ICSP™ 编程数据引脚

图注: TTL = TTL 兼容输入

ST = CMOS 电平的施密特触发器输入

O = 输出

CMOS = CMOS 兼容输入或输出

I = 输入

P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

表 1-2: PIC18F2XK20 I/O 引脚说明 (续)

引脚名称	引脚编号		引脚类型	缓冲器类型	说明
	PDIP 和 SOIC	QFN			
					PORTC 是双向 I/O 端口。
RC0/T1OSO/T13CKI RC0 T1OSO T13CKI	11	8	I/O O I	ST — ST	数字 I/O Timer1 振荡器输出 Timer1/Timer3 外部时钟输入
RC1/T1OSI/CCP2 RC1 T1OSI CCP2 ⁽¹⁾	12	9	I/O I I/O	ST 模拟 ST	数字 I/O Timer1 振荡器输入 捕捉 2 输入 / 比较 2 输出 / PWM2 输出
RC2/CCP1/P1A RC2 CCP1 P1A	13	10	I/O I/O O	ST ST CMOS	数字 I/O 捕捉 1 输入 / 比较 1 输出 增强型 CCP1 PWM 输出
RC3/SCK/SCL RC3 SCK SCL	14	11	I/O I/O I/O	ST ST ST	数字 I/O SPI 模式的同步串行时钟输入 / 输出 I ² C TM 模式的同步串行时钟输入 / 输出
RC4/SDI/SDA RC4 SDI SDA	15	12	I/O I I/O	ST ST ST	数字 I/O SPI 数据输入 I ² C TM 数据 I/O
RC5/SDO RC5 SDO	16	13	I/O O	ST —	数字 I/O SPI 数据输出
RC6/TX/CK RC6 TX CK	17	14	I/O O I/O	ST — ST	数字 I/O EUSART 异步发送 EUSART 同步时钟 (见相关的 RX/DT 引脚信息)
RC7/RX/DT RC7 RX DT	18	15	I/O I I/O	ST ST ST	数字 I/O EUSART 异步接收 EUSART 同步数据 (见相关的 TX/CK 引脚信息)
RE3	—	—	—	—	见 MCLR/VPP/RE3 引脚信息
VSS	8, 19	5, 16	P	—	逻辑和 I/O 引脚的参考地
VDD	20	17	P	—	逻辑和 I/O 引脚的正电源

图注: TTL = TTL 兼容输入

CMOS = CMOS 兼容输入或输出

ST = CMOS 电平的施密特触发器输入

I = 输入

O = 输出

P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

PIC18F2XK20/4XK20

表 1-3: PIC18F4XK20 I/O 引脚说明

引脚名称	引脚编号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
MCLR/VPP/RE3 MCLR VPP RE3	1	18	18	I P I	ST ST	主复位（输入）或编程电压（输入） 低电平有效的主复位（器件复位）输入 编程电压输入 数字输入
OSC1/CLKIN/RA7 OSC1 CLKIN RA7	13	32	30	I I I/O	ST CMOS TTL	晶振或外部时钟输入 晶振输入或外部时钟源输入 配置为RC模式时为ST缓冲器输入；否则为模拟输入 外部时钟源输入。总是与OSC1引脚功能相关联（见相关的OSC1/CLKIN和OSC2/CLKOUT引脚信息） 通用I/O引脚
OSC2/CLKOUT/RA6 OSC2 CLKOUT RA6	14	33	31	O O I/O	— — TTL	晶振或时钟输出 晶振输出。在晶振模式下，该引脚与晶振或谐振器相连 在RC模式下，OSC2引脚输出CLKOUT信号，其频率是OSC1引脚上信号频率的4分频，该频率等于指令周期的倒数 通用I/O引脚

图注: TTL = TTL 兼容输入

ST = CMOS 电平的施密特触发器输入
O = 输出

CMOS = CMOS 兼容输入或输出

I = 输入
P = 电源

- 注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。
2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

PIC18F2XK20/4XK20

表 1-3: PIC18F4XK20 I/O 引脚说明 (续)

引脚名称	引脚编号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
						PORTA 是双向 I/O 端口。
RA0/AN0/C12IN0- RA0 AN0 C12IN0-	2	19	19	I/O 	TTL 模拟 模拟	数字 I/O 模拟输入 0, ADC 通道 0 比较器 C1 和 C2 的反相输入
RA1/AN1/C12IN0- RA1 AN1 C12IN0-	3	20	20	I/O 	TTL 模拟 模拟	数字 I/O 模拟输入 1, ADC 通道 1 比较器 C1 和 C2 的反相输入
RA2/AN2/VREF-/CVREF/ C2IN+ RA2 AN2 VREF- CVREF C2IN+	4	21	21	I/O O 	TTL 模拟 模拟 模拟 模拟	数字 I/O 模拟输入 2, ADC 通道 2 A/D 参考电压 (低电压) 输入 比较器参考电压输出 比较器 C2 的同相输入
RA3/AN3/VREF+/ C1IN+ RA3 AN3 VREF+ C1IN+	5	22	22	I/O 	TTL 模拟 模拟 模拟	数字 I/O 模拟输入 3, ADC 通道 3 A/D 参考电压 (高电压) 输入 比较器 C1 的同相输入
RA4/T0CKI/C1OUT RA4 T0CKI C1OUT	6	23	23	I/O O	ST ST CMOS	数字 I/O Timer0 外部时钟输入 比较器 C1 的输出
RA5/AN4/SS/HLDIN/ C2OUT RA5 AN4 SS HLDIN C2OUT	7	24	24	I/O O	TTL 模拟 TTL 模拟 CMOS	数字 I/O 模拟输入 4, ADC 通道 4 SPI 从选择输入 高 / 低压检测输入 比较器 C2 的输出
RA6						见 OSC2/CLKOUT/RA6 引脚信息
RA7						见 OSC1/CLKIN/RA7 引脚信息

图注: TTL = TTL 兼容输入

CMOS = CMOS 兼容输入或输出

ST = CMOS 电平的施密特触发器输入

| = 输入

O = 输出

P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

PIC18F2XK20/4XK20

表 1-3: PIC18F4XK20 I/O 引脚说明 (续)

引脚名称	引脚编号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
						PORTB 是双向 I/O 端口。 PORTB 的所有输入端都可软件编程为内部弱上拉。
RB0/INT0/FLT0/AN12 RB0 INT0 FLT0 AN12	33	9	8	I/O I I I	TTL ST ST 模拟	数字 I/O 外部中断 0 增强型 CCP1 的 PWM 故障输入 模拟输入 12, ADC 通道 12
RB1/INT1/AN10/ C12IN3- RB1 INT1 AN10 C12IN3-	34	10	9	I/O I I I	TTL ST 模拟 模拟	数字 I/O 外部中断 1 模拟输入 10, ADC 通道 10 比较器 C1 和 C2 的反相输入
RB2/INT2/AN8 RB2 INT2 AN8	35	11	10	I/O I I	TTL ST 模拟	数字 I/O 外部中断 2 模拟输入 8, ADC 通道 8
RB3/AN9/C12IN2-/ CCP2 RB3 AN9 C12IN23- CCP2 ⁽²⁾	36	12	11	I/O I I I/O	TTL 模拟 模拟 ST	数字 I/O 模拟输入 9, ADC 通道 9 比较器 C1 和 C2 的反相输入 捕捉 2 输入 / 比较 2 输出 / PWM2 输出
RB4/KBI0/AN11 RB4 KBI0 AN11	37	14	14	I/O I I	TTL TTL 模拟	数字 I/O 电平变化中断引脚 模拟输入 11, ADC 通道 11
RB5/KBI1/PGM RB5 KBI1 PGM	38	15	15	I/O I I/O	TTL TTL ST	数字 I/O 电平变化中断引脚 低压 ICSP™ 编程使能引脚
RB6/KBI2/PGC RB6 KBI2 PGC	39	16	16	I/O I I/O	TTL TTL ST	数字 I/O 电平变化中断引脚 在线调试器和 ICSP™ 编程时钟引脚
RB7/KBI3/PGD RB7 KBI3 PGD	40	17	17	I/O I I/O	TTL TTL ST	数字 I/O 电平变化中断引脚 在线调试器和 ICSP™ 编程数据引脚

图注: TTL = TTL 兼容输入

ST = CMOS 电平的施密特触发器输入

O = 输出

CMOS = CMOS 兼容输入或输出

I = 输入

P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

表 1-3: PIC18F4XK20 I/O 引脚说明 (续)

引脚名称	引脚编号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
						PORTC 是双向 I/O 端口。
RC0/T1OSO/T13CKI RC0 T1OSO T13CKI	15	34	32	I/O O I	ST — ST	数字 I/O Timer1 振荡器输出 Timer1/Timer3 外部时钟输入
RC1/T1OSI/CCP2 RC1 T1OSI CCP2 ⁽¹⁾	16	35	35	I/O I I/O	ST CMOS ST	数字 I/O Timer1 振荡器输入 捕捉 2 输入 / 比较 2 输出 / PWM2 输出
RC2/CCP1/P1A RC2 CCP1 P1A	17	36	36	I/O I/O O	ST ST —	数字 I/O 捕捉 1 输入 / 比较 1 输出 / PWM1 输出 增强型 CCP1 输出
RC3/SCK/SCL RC3 SCK SCL	18	37	37	I/O I/O I/O	ST ST ST	数字 I/O SPI 模式的同步串行时钟输入 / 输出 I ² C TM 模式的同步串行时钟输入 / 输出
RC4/SDI/SDA RC4 SDI SDA	23	42	42	I/O I I/O	ST ST ST	数字 I/O SPI 数据输入 I ² C TM 数据 I/O
RC5/SDO RC5 SDO	24	43	43	I/O O	ST —	数字 I/O SPI 数据输出
RC6/TX/CK RC6 TX CK	25	44	44	I/O O I/O	ST — ST	数字 I/O EUSART 异步发送 EUSART 同步时钟 (见相关的 RX/DT 引脚信息)
RC7/RX/DT RC7 RX DT	26	1	1	I/O I I/O	ST ST ST	数字 I/O EUSART 异步接收 EUSART 同步数据 (见相关的 TX/CK 引脚信息)

图注: TTL = TTL 兼容输入

ST = CMOS 电平的施密特触发器输入

O = 输出

CMOS = CMOS 兼容输入或输出

I = 输入

P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

PIC18F2XK20/4XK20

表 1-3: PIC18F4XK20 I/O 引脚说明 (续)

引脚名称	引脚编号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
						PORTD 是双向 I/O 端口或与微处理器端口接口的并行从端口 (Parallel Slave Port, PSP)。当使能 PSP 模块时，这些引脚具有 TTL 输入缓冲器。
RD0/PSP0 RD0 PSP0	19	38	38	I/O I/O	ST TTL	数字 I/O 并行从端口数据
RD1/PSP1 RD1 PSP1	20	39	39	I/O I/O	ST TTL	数字 I/O 并行从端口数据
RD2/PSP2 RD2 PSP2	21	40	40	I/O I/O	ST TTL	数字 I/O 并行从端口数据
RD3/PSP3 RD3 PSP3	22	41	41	I/O I/O	ST TTL	数字 I/O 并行从端口数据
RD4/PSP4 RD4 PSP4	27	2	2	I/O I/O	ST TTL	数字 I/O 并行从端口数据
RD5/PSP5/P1B RD5 PSP5 P1B	28	3	3	I/O I/O O	ST TTL —	数字 I/O 并行从端口数据 增强型 CCP1 输出
RD6/PSP6/P1C RD6 PSP6 P1C	29	4	4	I/O I/O O	ST TTL —	数字 I/O 并行从端口数据 增强型 CCP1 输出
RD7/PSP7/P1D RD7 PSP7 P1D	30	5	5	I/O I/O O	ST TTL —	数字 I/O 并行从端口数据 增强型 CCP1 输出

图注: TTL = TTL 兼容输入

ST = CMOS 电平的施密特触发器输入

O = 输出

CMOS = CMOS 兼容输入或输出

I = 输入

P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

表 1-3: PIC18F4XK20 I/O 引脚说明 (续)

引脚名称	引脚编号			引脚类型	缓冲器类型	说明
	PDIP	QFN	TQFP			
						PORTE 是双向 I/O 端口。
RE0/RD/AN5 RE0 RD AN5	8	25	25	I/O I I	ST TTL 模拟	数字 I/O 并行从端口的读控制 (见相关的 WR 和 CS 引脚信息) 模拟输入 5, ADC 通道 5
RE1/WR/AN6 RE1 WR AN6	9	26	26	I/O I I	ST TTL 模拟	数字 I/O 并行从端口的写控制 (见相关的 CS 和 RD 引脚信息) 模拟输入 6, ADC 通道 6
RE2/CS/AN7 RE2 CS AN7	10	27	27	I/O I I	ST TTL 模拟	数字 I/O 并行从端口的片选控制 (见相关的 RD 和 WR 引脚信息) 模拟输入 7, ADC 通道 7
RE3	—	—	—	—	—	见 MCLR/VPP/RE3 引脚信息
VSS	12, 31	6, 30, 31	6, 29	P	—	逻辑和 I/O 引脚的参考地
VDD	11, 32	7, 8, 28, 29	7, 28	P	—	逻辑和 I/O 引脚的正电源
NC	—	13	12, 13, 33, 34	—	—	无连接

图注: TTL = TTL 兼容输入

CMOS = CMOS 兼容输入或输出

ST = CMOS 电平的施密特触发器输入

| = 输入

O = 输出

P = 电源

注 1: 当配置位 CCP2MX 置 1 时 CCP2 的默认分配。

2: 当配置位 CCP2MX 清零时 CCP2 的备用分配。

PIC18F2XK20/4XK20

注:

2.0 振荡器模块（带故障保护时钟监视器）

2.1 概述

振荡器模块具有多种时钟源和选择特性，广泛使用于各种应用中，同时最大限度地发挥性能并降低功耗。图 2-1 给出了振荡器模块的框图。

时钟源可配置为来自外部振荡器、石英晶体谐振器、陶瓷谐振器以及容阻（Resistor-Capacitor, RC）电路。此外，系统时钟源可配置为两个内部振荡器之一，并通过软件来选择速度。其他时钟特性包括：

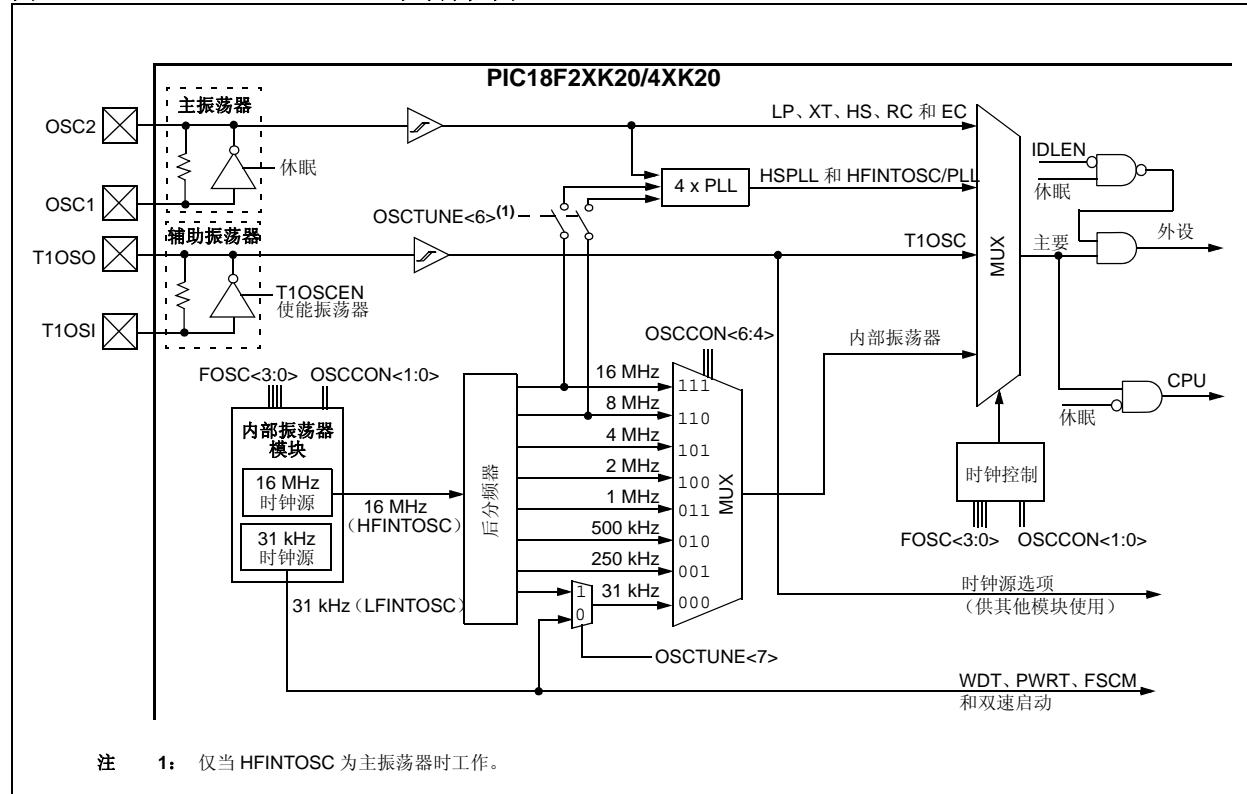
- 可通过软件选择外部或内部系统时钟源。
- 双速启动模式，最大限度地缩短外部振荡器起振与代码执行之间的延时。
- 故障保护时钟监视器（FSCM），用来检测外部时钟源（LP、XT、HS、EC 或 RC 模式）故障并自动切换到内部振荡器。

振荡器模块可配置为以下 10 种主时钟模式之一。

- | | |
|------------|--|
| 1. LP | 低功耗晶振 |
| 2. XT | 晶振 / 谐振器 |
| 3. HS | 高速晶振 / 谐振器 |
| 4. HSPLL | 使能 PLL 的高速晶振 / 谐振器 |
| 5. RC | 外部电阻 / 电容，通过 RA6 引脚输出 Fosc/4 的信号 |
| 6. RCIO | 外部电阻 / 电容，RA6 用作 I/O 引脚 |
| 7. INTOSC | 内部振荡器，通过 RA6 引脚输出 Fosc/4 的信号，RA7 用作 I/O 引脚 |
| 8. INTOSCI | 内部振荡器，RA6 和 RA7 均用作 I/O 引脚 |
| 9. EC | 带 Fosc/4 输出的外部时钟 |
| 10. ECIO | RA6 用作 I/O 引脚的外部时钟 |

通过 CONFIG1H 配置寄存器的 FOSC<3:0> 位来选择主时钟模式。HFINTOSC 和 LFINTOSC 分别为出厂校准的高频和低频振荡器，可用作内部时钟源。

图 2-1：PIC® MCU 时钟源框图



2.2 振荡器控制

OSCCON 寄存器（寄存器 2-1）控制全功耗模式和功耗管理模式下器件时钟工作的多个方面。

- 主系统时钟选择（SCS）
- 内部频率选择位（IRCF）
- 时钟状态位（OSTS 和 IOFS）
- 功耗管理选择（IDLEN）

2.2.1 主系统时钟选择

系统时钟选择位 SCS<1:0> 用于选择主时钟源。可用的时钟源有：

- 由 CONFIG1H 的 FOSC<3:0> 位定义的主时钟。主时钟可以是主振荡器、外部时钟或内部振荡器模块。
- 辅助时钟（Timer1 振荡器）。
- 内部振荡器模块（HFINTOSC 和 LFINTOSC）。

当写入一个或多个位之后，接着是一段很短的时钟转换间隔，然后时钟源会立即改变。在所有形式的复位后 SCS 位都会被清零以选择主时钟。

2.2.2 内部频率选择

内部振荡器频率选择位（IRCF<2:0>）用于选择内部振荡器模块的频率输出。频率输出选择有 LFINTOSC 时钟源（31 kHz）、HFINTOSC 时钟源（16 MHz）或来自 HFINTOSC 后分频器的频率之一（31.25 kHz 至 8 MHz）。如果主时钟由内部振荡器模块提供，改变这些位的状态会使内部振荡器输出立即改变。器件复位时，内部振荡器的输出频率被设置为默认频率 1 MHz。

2.2.3 低频选择

当选取 31 kHz 的标称输出频率（IRCF<2:0> = 000）时，用户可以选择用作时钟源的内部振荡器。这通过使用 OSCTUNE 寄存器的 INTSRC 位实现。将该位置 1 选择 HFINTOSC 作为时钟源，并通过使能 HFINTOSC 后分频器的 512 分频输出，使该时钟源输出 31.25 kHz 的时钟信号。将 INTSRC 清零选择 LFINTOSC（标称值为 31 kHz）作为时钟源。

此选项使用户能选择可调节且更精确的 HFINTOSC 作为时钟源，同时以非常低的时钟速度运行以节省功耗。无论 INTSRC 的设置如何，LFINTOSC 总是作为看门狗定时器和故障保护时钟监视器之类部件的时钟源。

2.2.4 时钟状态

OSCCON 寄存器的 OSTS 和 IOFS 位，以及 T1CON 寄存器的 T1RUN 位，指示当前是哪个时钟源在提供主时钟。OSTS 位置 1 表明振荡器起振定时器已超时且主时钟提供器件时钟。IOFS 位置 1 表明内部振荡器模块已稳定并在 HFINTOSC 时钟模式下提供器件时钟。当 SCS<1:0> = 00，且 HFINTOSC 是主时钟时，IOFS 和 OSTS 状态位都会置 1。T1RUN 位置 1 表明 Timer1 振荡器正在辅助时钟模式下提供器件时钟。当 SCS<1:0> ≠ 00 时，在任意时刻这三个位中将只有一位置 1。如果这些位都没有置 1，则表示当前时钟源是 LFINTOSC，或 HFINTOSC 刚刚起振且尚未稳定。

2.2.5 功耗管理

OSCCON 寄存器的 IDLEN 位决定当执行 SLEEP 指令时器件是进入休眠模式还是某种空闲模式。

第 3.0 节“功耗管理模式”更详细地讨论了 OSCCON 寄存器中标志位和控制位的使用。

- 注 1:** 要选择辅助时钟源，必须使能 Timer1 振荡器。通过将 T1CON 寄存器的 T1OSCEN 位置 1 可使能 Timer1 振荡器。如果 Timer1 振荡器未使能，那么主振荡器将继续使用先前选定的时钟源运行。然后在 T1OSCEN 位置 1 之后，时钟源将切换为辅助振荡器。
- 2:** 建议在 Timer1 振荡器稳定工作之后再选择辅助时钟源，否则当 Timer1 振荡器起振时可能会发生很长的延时。

寄存器 2-1: OSCCON: 振荡器控制寄存器

R/W-0	R/W-0	R/W-1	R/W-1	R-q	R-0	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS ⁽¹⁾	IOFS	SCS1	SCS0
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

q = 值取决于具体条件

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	IDLEN: 空闲使能位 1 = 执行 SLEEP 指令后器件进入空闲模式 0 = 执行 SLEEP 指令后器件进入休眠模式
bit 6-4	IRCF<2:0>: 内部振荡器频率选择位 111 = 16 MHz (由 HFINTOSC 直接驱动时钟) 110 = 8 MHz 101 = 4 MHz 100 = 2 MHz 011 = 1 MHz ⁽³⁾ 010 = 500 kHz 001 = 250 kHz 000 = 31 kHz (来自 HFINTOSC/512 或直接来自 LFINTOSC) ⁽²⁾
bit 3	OSTS: 振荡器起振延时状态位 ⁽¹⁾ 1 = 器件依靠 CONFIG1 寄存器的 FOSC<2:0> 定义的时钟运行 0 = 器件依靠内部振荡器 (HFINTOSC 或 LFINTOSC) 运行
bit 2	IOFS: HFINTOSC 频率稳定位 1 = HFINTOSC 频率稳定 0 = HFINTOSC 频率不稳定
bit 1-0	SCS<1:0>: 系统时钟选择位 1x = 内部振荡器模块 01 = 辅助 (Timer1) 振荡器 00 = 主时钟 (由 CONFIG1H[FOSC<3:0>] 决定)

注 1: 复位状态取决于 IESO 配置位的状态。

2: 时钟源由 OSCTUNE 寄存器的 INTSRC 位选择, 请参见上文。

3: 复位时 HFINTOSC 的默认输出频率。

2.3 时钟源模式

时钟源模式可分为外部和内部模式。

- 外部时钟模式依靠外部电路提供时钟源。例如：时钟模块（EC 模式）、石英晶体谐振器或陶瓷谐振器（LP、XT 和 HS 模式）以及阻容（RC 模式）电路。
- 内部时钟源内置于振荡器模块中。振荡器模块有两个内部振荡器：一个是 16 MHz 高频内部振荡器（HFINTOSC），另一个是 31 kHz 低频内部振荡器（LFINTOSC）。

通过 OSCCON 寄存器的系统时钟选择（SCS<1:0>）位在外部或内部时钟源之间选择系统时钟。更多信息，请参见第 2.9 节“时钟切换”。

2.4 外部时钟模式

2.4.1 振荡器起振定时器（OST）

如果振荡器模块配置为 LP、XT 或 HS 模式，则振荡器起振定时器（Oscillator Start-up Timer, OST）对 OSC1 计数 1024 个振荡周期。这发生在上电复位（Power-on Reset, POR）和上电延时定时器（Power-up Timer, PWRT）延时结束（如果配置了），或从休眠中唤醒时。在此期间，程序计数器不递增，程序执行暂停。OST 确保使用石英晶体谐振器或陶瓷谐振器的振荡器电路已经起振并为振荡器模块提供稳定的系统时钟。当在时钟源之间切换时，需要一定的延时以使新时钟稳定。表 2-1 给出了振荡器延时的例子。

为了使外部振荡器起振和代码执行之间的延时最小，可选择双速时钟启动模式（见第 2.10 节“双速时钟启动模式”）。

表 2-1：振荡器延时示例

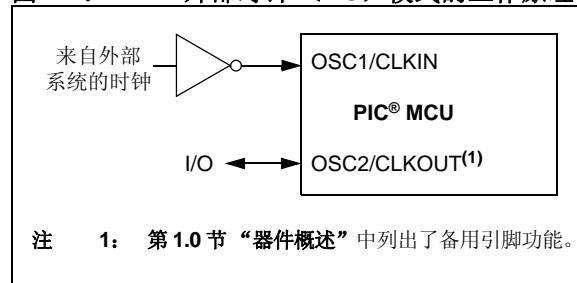
切换自	切换到	频率	振荡器延时
休眠 /POR	LFINTOSC HFINTOSC	31 kHz 250 kHz 到 16 MHz	振荡器预热延时（TWARM）
休眠 /POR	EC 或 RC	DC – 64 MHz	2 个指令周期
LFINTOSC (31 kHz)	EC 或 RC	DC – 64 MHz	每次一周期
休眠 /POR	LP、XT 或 HS	32 kHz 到 40 MHz	1024 个时钟周期（OST）
休眠 /POR	HSPLL	32 MHz 到 64 MHz	1024 个时钟周期（OST）+ 2 ms
LFINTOSC (31 kHz)	HFINTOSC	250 kHz 到 16 MHz	1 μs（近似值）

2.4.2 EC 模式

外部时钟（EC）模式允许外部产生的逻辑电平作为系统时钟源。工作在此模式下时，外部时钟源连接到 OSC1 输入，OSC2 可用作通用 I/O。图 2-2 给出了 EC 模式的引脚连接。

当选取 EC 模式时，振荡器起振定时器（OST）被禁止。因此，上电复位（POR）后或者从休眠中唤醒后的操作不存在延时。因为 PIC® MCU 的设计是完全静态的，停止外部时钟输入将使器件暂停工作并保持所有数据完整。当再次启动外部时钟时，器件恢复工作，就好像没有停止过一样。

图 2-2：外部时钟（EC）模式的工作原理



2.4.3 LP、XT 和 HS 模式

LP、XT 和 HS 模式支持使用连接到 OSC1 和 OSC2 的石英晶体谐振器或陶瓷谐振器（图 2-3）。模式选择内部反相放大器的低、中或高增益设定，以支持各种谐振器类型及速度。

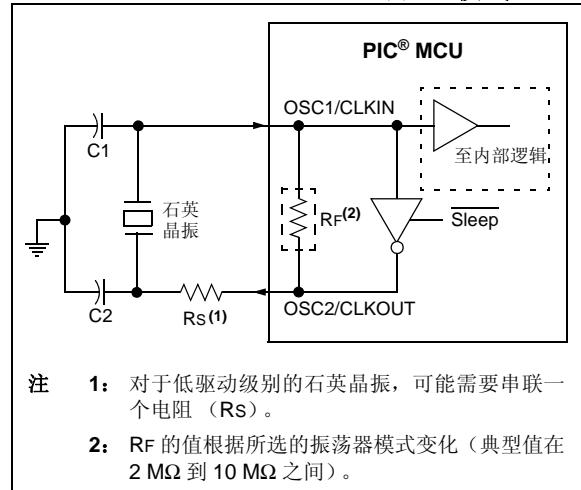
LP 振荡器模式选择内部反相放大器的最低增益设定。LP 模式的电流消耗在三种模式中最小。该模式最适合驱动具备低驱动级别规格要求的谐振器，例如，音叉（Tuning Fork）型晶振。

XT 振荡器模式选择内部反相放大器的中等增益设定。XT 模式的电流消耗在三种模式中居中。该模式最适合驱动具备中等驱动级别规格要求的谐振器。

HS 振荡器模式选择内部反相放大器的最高增益设定。HS 模式的电流消耗在三种模式中最大。该模式最适合驱动需要高驱动设定的谐振器。

图 2-3 和图 2-4 分别给出了石英晶体谐振器和陶瓷谐振器的典型电路。

图 2-3: 石英晶振的工作原理
(LP、XT 或 HS 模式)



注 1: 石英晶振的特性随类型、封装和制造商而变化。要了解规格说明和推荐应用，应查阅制造商提供的数据手册。

2: 应始终验证振荡器在应用要求的 VDD 和温度范围下的性能。

3: 如需振荡器设计帮助，请参见以下 Microchip 应用笔记：

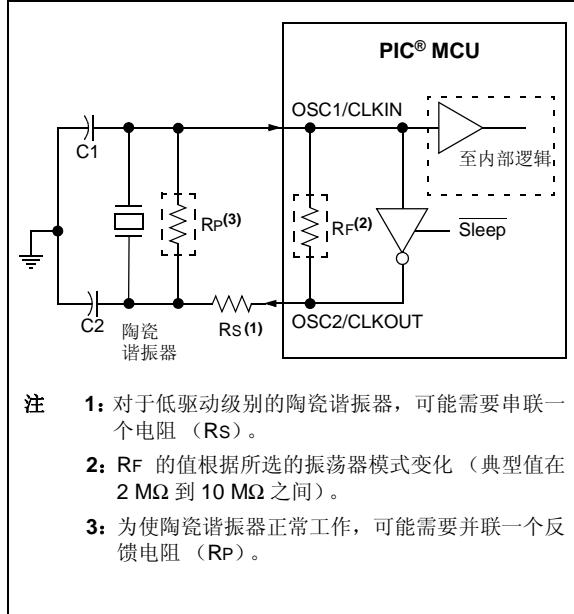
- AN826, “Crystal Oscillator Basics and Crystal Selection for rfPIC® and PIC® Devices” (DS00826)

- AN849, “Basic PIC® Oscillator Design” (DS00849)

- AN943, “Practical PIC® Oscillator Analysis and Design” (DS00943)

- AN949, “Making Your Oscillator Work” (DS00949)

图 2-4: 陶瓷谐振器的工作原理
(XT 或 HS 模式)



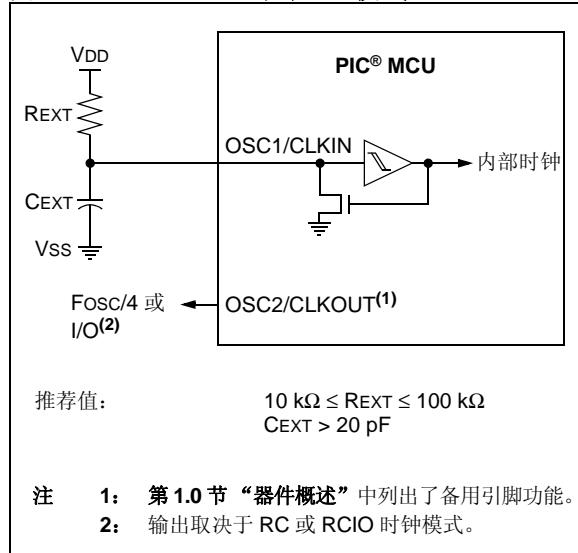
2.4.4 外部 RC 模式

外部阻容 (RC) 模式支持使用外部 RC 电路。对时钟精度要求不高时，这使设计人员有了很大的频率选择空间，且保持成本最低。有以下两种模式：RC 和 RCIO。

2.4.4.1 RC 模式

在 RC 模式下，RC 电路连接到 OSC1。OSC2/CLKOUT 输出 RC 振荡器频率的 4 分频。该信号可用来为外部电路、同步、校准、测试或其他应用需求提供时钟。图 2-5 给出了外部 RC 模式的连接图。

图 2-5：外部 RC 模式



2.4.4.2 RCIO 模式

在 RCIO 模式下，RC 电路连接到 OSC1。OSC2 成为额外的通用 I/O 引脚。

RC 振荡器频率与供电电压、电阻 (**REXT**) 和电容 (**CEXT**) 值以及工作温度有关。影响振荡器频率的其他因素有：

- 输入电压门限值变化
- 元件容差
- 不同封装的电容

用户还应考虑因所使用的外部 RC 元件的容差而导致的差异。

2.5 内部时钟模式

振荡器模块有两个独立的内部振荡器，可配置或选取为系统时钟源。

1. **HFINTOSC**（高频内部振荡器）出厂时已校准，工作频率为 16 MHz。使用 **OSCTUNE** 寄存器（寄存器 2-2），可通过软件调整 HFINTOSC 的频率。
2. **LFINTOSC**（低频内部振荡器）工作频率为 31 kHz。

使用 **OSCCON** 寄存器的内部振荡器频率选择位 **IRCF<2:0>**，可通过软件选择系统时钟速度。

通过 **OSCCON** 寄存器的系统时钟选择 (**SCS<1:0>**) 位在外部或内部时钟源之间选择系统时钟。更多信息，请参见第 2.9 节“时钟切换”。

2.5.1 INTOSC 和 INTOSCIO 模式

INTOSC 和 INTOSCIO 模式可将内部振荡器配置为主时钟源。CONFIG1H 寄存器中的 **FOSC<3:0>** 位决定选取何种模式。更多信息，请参见第 23.0 节“CPU 的特殊功能”。

在 **INTOSC** 模式下，**OSC1/CLKIN** 可用作通用 I/O。**OSC2/CLKOUT** 输出所选内部振荡器频率的 4 分频。**CLKOUT** 信号可用来为外部电路、同步、校准、测试或其他应用需求提供时钟。

在 **INTOSCIO** 模式下，**OSC1/CLKIN** 和 **OSC2/CLKOUT** 可用作通用 I/O。

2.5.2 HFINTOSC

HFINTOSC 的输出连接到后分频器和多路开关（见图 2-1）。使用 **OSCCON** 寄存器的 **IRCF<2:0>** 位，可通过软件选择 8 个频率之一。更多信息，请参见第 2.5.4 节“频率选择位 (IRCF)”。

发生以下情况时，HFINTOSC 被使能：

- **SCS1 = 1** 且 **IRCF<2:0> ≠ 000**
- **SCS1 = 1**、**IRCF<2:0> = 000** 且 **INTSRC = 1**
- **CONFIG1H** 的 **IESO** 位 = 1 使能双速启动
- **CONFIG1H** 的 **FCMEM** 位 = 1 使能双速启动和故障保护模式
- **CONFIG1H** 的 **FOSC<3:0>** 位将内部振荡器选为主时钟

OSCCON 寄存器的 HF 内部振荡器 (**IOFS**) 位用于指示 HFINTOSC 是否稳定。

2.5.2.1 OSCTUNE 寄存器

HFINTOSC 在出厂时已校准，但可通过在软件中写入 OSCTUNE 寄存器（寄存器 2-2）的 TUN<5:0> 位来进行调节。

TUN<5:0> 的默认值为 000000。该值是一个 6 位的二进制补码。

当 OSCTUNE 寄存器被修改时，HFINTOSC 频率将开始转变为新频率。转变期间，代码将继续执行。是否已发生频率转变并无明确的指示。

OSCTUNE 不影响 LFINTOSC 频率。依赖于 LFINTOSC 时钟源频率的功能，如上电延时定时器（PWRT）、看

门狗定时器（WDT）、故障保护时钟监视器（FSCM）以及外设等，其工作不受频率改变的影响。

OSCTUNE 寄存器也有 INTSRC 和 PLLN 位，它们控制内部振荡器模块的某些功能。

当选取 31 kHz 频率时，用户可通过 INTSRC 位选择用作时钟源的内部振荡器。在第 2.2.3 节“低频选择”中对此进行了更详细的说明。

在内部振荡器模式下，PLLEN 位控制倍频器 PLL 的工作模式。关于 PLLEN 位的功能的更多详细信息，请参见第 2.6.2 节“HFINTOSC 模式下的 PLL”。

寄存器 2-2： OSCTUNE：振荡器调节寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INTSRC	PLLEN ⁽¹⁾	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
bit 7							bit 0

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7

INTSRC：内部振荡器低频时钟源选择位

1 = 来自 16 MHz HFINTOSC 时钟源的 31.25 kHz 器件时钟（使能 512 分频）

0 = 直接来自 LFINTOSC 内部振荡器的 31 kHz 器件时钟

bit 6

PLLEN：HFINTOSC 的倍频器 PLL 使能位⁽¹⁾

1 = 为 HFINTOSC 使能 PLL（仅限 8 MHz 和 16 MHz）

0 = 禁止 PLL

bit 5-0

TUN<5:0>：频率调节位

011111 = 最高频率

011110 =

•••

000001 =

000000 = 振荡器模块运行在出厂时已校准的频率下

111111 =

•••

100000 = 最低频率

注 1：只有 HFINTOSC 是主时钟源（FOSC<2:0> = 100x），且选定的频率是 8 MHz 或 16 MHz 时，PLLEN 位才有效。否则，PLLEN 位将不可用，并且始终读为 0。

2.5.3 LFINTOSC

低频内部振荡器 (LFINTOSC) 是 31 kHz 内部时钟源。LFINTOSC 的输出连接到内部振荡器模块频率选择多路开关 (见图 2-1)。使用 OSCCON 寄存器的 IRCF<2:0> 位和 OSCTUNE 寄存器的 INTSRC 位, 可通过软件选择 31 kHz。更多信息, 请参见第 2.5.4 节 “频率选择位 (IRCF)”。LFINTOSC 还是上电延时定时器 (PWRT)、看门狗定时器 (WDT) 以及故障保护时钟监视器 (FSCM) 的时钟源。

当使能以下任一功能时, 将使能 LFINTOSC:

- OSCCON 寄存器的 IRCF<2:0> 位 = 000 且 OSCTUNE 寄存器的 INTSRC 位 = 0
- 上电延时定时器 (PWRT)
- 看门狗定时器 (WDT)
- 故障保护时钟监视器 (FSCM)

2.5.4 频率选择位 (IRCF)

16 MHz HFINTOSC 和 31 kHz LFINTOSC 的输出连接到后分频器和多路开关 (见图 2-1)。OSCCON 寄存器的内部振荡器频率选择位 IRCF<2:0> 用于选择内部振荡器的输出频率。可通过软件选择以下 8 种频率之一:

- 16 MHz
- 8 MHz
- 4 MHz
- 2 MHz
- 1 MHz (复位后的默认值)
- 500 kHz
- 250 kHz
- 31 kHz (LFINTOSC 或 HFINTOSC/512)

注: 任何复位后, OSCCON 寄存器的 IRCF<2:0> 位都被设置为 011, 频率选择被设置为 1 MHz。用户可修改 IRCF 位来选择其他频率。

2.5.5 HFINTOSC 频率漂移

出厂时内部振荡器模块输出 (HFINTOSC) 被校准为 16 MHz。但是, 此频率可能会随着 VDD 或温度的改变而发生漂移, 这一点可能会以各种方式影响控制器的工作。可通过修改 OSCTUNE 寄存器中 TUN<5:0> 位的值来调整 HFINTOSC 频率。这不会对 LFINTOSC 时钟源的频率造成影响。

调节 HFINTOSC 时钟源需要了解何时调节、调节的方向以及在某些情况下需要的调整量。以下几小节讨论了三种可能的补偿技术, 但也可能使用其他技术。

2.5.5.1 用 USART 进行补偿

当 USART 开始产生帧错误, 或者在异步模式下接收数据有错误时可能需要进行调节。帧错误表示器件时钟频率太高; 要对此进行调节, 可以减小 OSCTUNE 中的值来降低时钟频率。另一方面, 数据中有错误可能表明时钟速度太低; 要进行补偿, 可以增大 OSCTUNE 中的值来提高时钟频率。

2.5.5.2 用定时器进行补偿

此技术是将器件时钟的速度与某一个参考时钟进行比较。可能要用到两个定时器; 一个由外设时钟提供时钟源, 而另一个由一个固定的参考源 (如 Timer1 振荡器) 提供时钟源。

两个定时器都被清零, 但由参考源提供时钟信号的定时器产生中断。当中断发生时, 使用内部时钟源的定时器值被读取且两个定时器均被清零。如果使用内部时钟源的定时器的值大于期望值, 则表示内部振荡器模块运行过快。要对此进行调整, 需减小 OSCTUNE 寄存器中的值。

2.5.5.3 在捕捉模式下用 CCP 模块进行补偿

CCP 模块可以使用由内部振荡器模块提供时钟信号的自由运行的 Timer1 (或 Timer3) 和已知周期的外部事件 (即交流电源频率)。在 CCPRxH:CCPRxL 寄存器中捕捉并记录第一个事件的时间以备以后使用。当第二个事件导致捕捉时, 要用第二个事件的时间减去第一个事件的时间。由于外部事件的周期是已知的, 因此可以计算事件之间的时间差。

如果测得的时间比计算得到的时间大很多, 则表示内部振荡器模块运行过快; 要对此进行补偿, 需减小 OSCTUNE 寄存器中的值。如果测得的时间比计算得到的时间小很多, 则表示内部振荡器模块运行过慢; 要对此进行补偿, 需增大 OSCTUNE 寄存器中的值。

2.6 PLL 倍频器

如果用户希望使用低频振荡器电路或通过晶振将器件频率调节至其最高额定频率，可以选择使用锁相环（PLL）电路。对于担心高频晶振引起 EMI 或希望使用内部振荡器来提供高速时钟的用户而言，这样做会有帮助。发生以下三种情况时，可以使用 PLL：

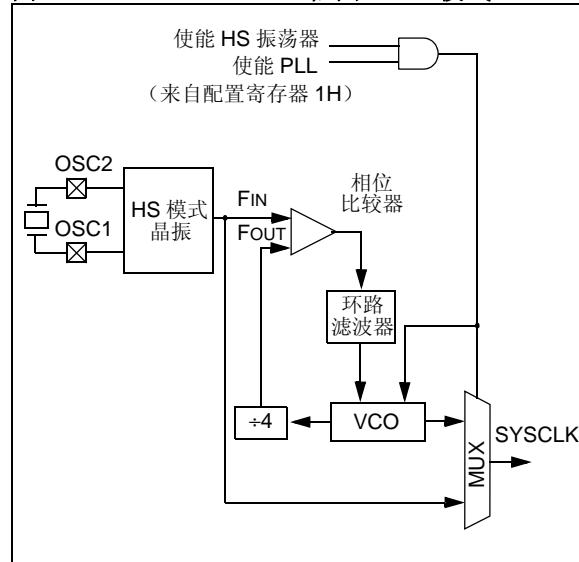
- 当主时钟为 HSPLL
- 当主时钟为 HFINTOSC 且所选频率为 16 MHz
- 当主时钟为 HFINTOSC 且所选频率为 8 MHz

2.6.1 HSPLL 振荡器模式

HSPLL 模式使用 HS 模式振荡器产生最高 16 MHz 的频率。然后 PLL 将振荡器输出频率 4 倍频，从而产生最高 64 MHz 的内部时钟频率。仅当 HFINTOSC 是主时钟，并且在 HSPLL 振荡器模式下不可用时，OSCTUNE 寄存器的 PLLEN 位才有效。

仅当将 FOSC<3:0> 配置位编程为 HSPLL 模式 (= 0110) 时，主振荡器才可以使用 PLL。

图 2-6: PLL 框图 (HS 模式)



2.6.2 HFINTOSC 模式下的 PLL

内部振荡器模块可以通过使用 4x 倍频器来产生比一般内部振荡器所能产生的时钟速度更快的器件时钟速度。当使能时，PLL 最高可产生 64 MHz 的时钟速度。

与 HSPLL 模式不同，PLL 由软件控制。当使用 HFINTOSC 时，OSCTUNE 寄存器的 PLLEN 控制位用于使能或禁止 PLL 操作。

当器件被配置为使用内部振荡器模块作为其主时钟源 (FOSC<3:0> = 1001 或 1000) 时，可以使用 PLL。此外，仅当选定的输出频率为 8 MHz 或 16 MHz (OSCCON<6:4> = 111 或 110) 时，PLL 才会工作。如果两个条件都不满足，则会禁止 PLL。

PLLEN 控制位只有在使用 PLL 的内部振荡器模式下才有效。在所有其他模式下，它被强制为 0 并且无效。

2.7 功耗管理模式对各种时钟源的影响

关于本节讨论的模式的更多信息，请参见第 3.0 节“功耗管理模式”。表 3-1 中也提供了快速参考列表。

当选取 PRI_IDLE 模式时，指定的主振荡器会继续运行而不中断。对于所有其他功耗管理模式，使用 OSC1 引脚的振荡器会被禁止。OSC1 引脚（以及由振荡器使用的 OSC2 引脚）将会停止振荡。

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

在内部振荡器模式（INTOSC_RUN 和 INTOSC_IDLE）下，由内部振荡器模块提供器件时钟源。无论是哪种功耗管理模式，31 kHz 的 LFINTOSC 输出均可被直接用来提供时钟并且可被使能来支持多种特殊的功能部件（关于 WDT、故障保护时钟监视器和双速启动的更多信息，请参见第 23.2 节“看门狗定时器（WDT）”、第 2.10 节“双速时钟启动模式”和第 2.11 节“故障保护时钟监视器”）。16 MHz 的 HFINTOSC 输出可被直接用来为器件提供时钟，或者也可先由后分频器进行分频再用作器件时钟。如果直接由 LFINTOSC 输出提供时钟，则会禁止 HFINTOSC 输出。

如果选择了休眠模式，所有的时钟源都会被停止。因为休眠模式切断了所有晶体管的开关电流，休眠模式能实现最低的器件电流消耗（仅泄漏电流）。

在休眠期间使能任何片上功能都将增加休眠时的电流消耗。要支持 WDT 工作，需要使能 LFINTOSC。Timer1 振荡器可用来为实时时钟提供时钟源。不需要器件时钟源的其他功能部件也可以工作（即，SSP 从器件、PSP 和 INTn 引脚等）。在第 26.8 节“直流特性”中列出了可能显著增加电流消耗的外设。

表 2-2：休眠模式下 OSC1 和 OSC2 引脚的状态

振荡器模式	OSC1 引脚	OSC2 引脚
RC 和 INTOSC	悬空，应通过外部电阻拉高	处于逻辑低电平（时钟 4 分频输出）
RCIO	悬空，应通过外部电阻拉高	配置为 PORTA 的 bit 6
INTOSCI0	配置为 PORTA 的 bit 7	配置为 PORTA 的 bit 6
ECIO	悬空，由外部时钟驱动	配置为 PORTA 的 bit 6
EC	悬空，由外部时钟驱动	处于逻辑低电平（时钟 4 分频输出）
LP、XT、HS 和 HSPLL	反馈反相器被禁止，处于静止电压值	反馈反相器被禁止，处于静止电压值

注：关于由休眠和 MCLR 复位引起的延时，请参见第 4.0 节“复位”中的表 4-2。

2.8 上电延时

由两个定时器控制上电延时，这样大多数应用都无需外接复位电路。上电延时可以确保在器件电源稳定（常规环境下）和主时钟稳定工作之前器件保持在复位状态。关于上电延时的更多信息，请参见第 4.5 节“器件复位定时器”。

第一个定时器是上电延时定时器（PWRT），它在上电时提供固定的延时（表 26-10 中的参数 33）。通过清零（= 0）PWRTEN 配置位可使能它。

第二个定时器是振荡器起振定时器（OST），用于在晶振稳定前使芯片保持在复位状态（LP、XT 和 HS 模式）。OST 在计数 1024 个振荡周期后允许振荡器为器件提供时钟。

当选取 HSPLL 振荡器模式时，器件将在 HS 模式下的 OST 延时之后另外保持 2 ms 的复位状态，这样可使 PLL 锁定到输入时钟频率。

POR之后有一个TCSD间隔的延时（表26-10中的参数38），在延时期间控制器为执行指令做准备。此延时与任何其他延时并行发生。当将 EC、RC 或 INTIO 模式之一用作主时钟源时，这可能是唯一的延时。

当选择 HFINTOSC 作为主时钟时，主系统时钟可以进行延时，直到 HFINTOSC 稳定为止。用户可以通过 CONFIG3H 配置寄存器的 HFOFST 位进行选择。当 HFOFST 位清零时，主系统将进行延时，直到 HFINTOSC 稳定为止。当 HFOFST 位置 1 时，主系统时钟将立即启动。在两种情况下，都可以读取 OSCCON 寄存器的 IOFS 位来确定 HFINTOSC 是否工作和稳定。

2.9 时钟切换

使用 OSCCON 寄存器的系统时钟选择（SCS<1:0>）位，可通过软件在外部和内部时钟源之间切换系统时钟源。

PIC18F2XK20/4XK20 器件包含防止在切换时钟源时发生时钟“毛刺”的电路。在切换时钟时，器件时钟会有短暂的停顿。该停顿的时间长度是旧时钟源的两个周期与新时钟源的三到四个周期的和。此公式假设新时钟源是稳定的。

第 3.1.2 节“进入功耗管理模式”详细讨论了时钟转换。

2.9.1 系统时钟选择（SCS<1:0>）位

OSCCON 寄存器的系统时钟选择（SCS<1:0>）位选择用于 CPU 和外设的系统时钟源。

- 当 SCS<1:0> = 00 时，系统时钟源由 CONFIG1H 配置寄存器中 FOSC<2:0> 位的配置决定。
- 当 SCS<1:0> = 10 时，系统时钟源由内部振荡器频率选择，而内部振荡器频率通过 OSCTUNE 寄存器的 INTSRC 位和 OSCCON 寄存器的 IRCF<2:0> 位选择。
- 当 SCS<1:0> = 01 时，系统时钟源是与 Timer1 共用的 32.768 kHz 辅助振荡器。

复位之后，OSCCON 寄存器的 SCS<1:0> 位始终都清零。

注：任何自动时钟切换（可能产生自双速启动或故障保护时钟监视器）都不会更新 OSCCON 寄存器的 SCS<1:0> 位。用户可以监视 T1CON 寄存器的 T1RUN 位和 OSCCON 寄存器的 IOFS 与 OSTS 位，以确定当前的系统时钟源。

2.9.2 振荡器起振延时状态（OSTS）位

OSCCON 寄存器的振荡器起振延时状态（OSTS）位指明系统时钟是来自外部时钟源（通过 CONFIG1H 配置寄存器中的 FOSC<3:0> 位定义），还是来自内部时钟源。当主振荡器为主时钟源时，OSTS 还特别指明在 LP、XT 或 HS 模式下，振荡器起振定时器（OST）是否已超时。

2.9.3 时钟切换时序

当在一个振荡器和另一个振荡器之间切换时，新的振荡器可能无法工作，这将带来节能（见图 2-7）。如果是这种情况，则在修改 OSCCON 寄存器的 SCS<1:0> 位之后，频率改变之前，存在一定的延时。OSCCON 寄存器的 OSTS 和 IOFS 位将反映外部振荡器和 HFINTOSC 振荡器的当前工作状态。频率选择时序如下：

- 修改 OSCCON 寄存器的 SCS<1:0> 位。
- 旧时钟继续工作，直到新时钟就绪。
- 时钟切换电路在新时钟就绪信号为真后等待两个连续的旧时钟上升沿。
- 从旧时钟的下一个下降沿开始，系统时钟保持低电平。
- 时钟切换电路再等待新时钟的两个上升沿。
- 在新时钟的下一个下降沿，释放系统时钟保持的低电平，并将新时钟切换为系统时钟。
- 时钟切换完成。

更多详细信息，请参见图 2-1。

如果旧频率和新频率的时钟源都是 HFINTOSC，则在新频率生效之前不存在启动延时。这是因为新旧频率都来自经过后分频器和多路开关的 HFINTOSC。

启动延时规范请参见第 26.0 节“电气特性”，在交流规范（振荡器模块）下。

2.10 双速时钟启动模式

双速启动模式通过最大限度地缩短外部振荡器起振与代码执行之间的延时，进一步节省了功耗。对于频繁使用休眠模式的应用，双速启动模式将从器件唤醒的时间中去除外部振荡器的起振时间，从而可降低器件的总体功耗。

该模式使得应用能够从休眠中唤醒，将 HFINTOSC 用作时钟源执行数条指令，然后再返回休眠状态而无需等待主振荡器的稳定。

注： 执行 SLEEP 指令将中止振荡器起振时间，并使 OSCCON 寄存器的 OSTS 位保持清零。

当振荡器模块被配置为 LP、XT 或 HS 模式时，振荡器起振定时器（OST）使能（见第 2.4.1 节“振荡器起振定时器（OST）”）。OST 将暂停程序执行，直到完成 1024 次振荡计数。双速启动模式在 OST 计数时使用内部振荡器进行工作，最大限度地缩短了代码执行的延时。当 OST 计数到 1024 且 OSCCON 寄存器的 OSTS 位置 1 时，程序执行切换至外部振荡器。

2.10.1 双速启动模式配置

当以下所有设置均按说明进行配置时，将使能双速启动模式：

- 通过将 CONFIG1H 配置寄存器的 IESO 位置 1 来使能双速启动模式。默认情况下，故障保护模式（FCMEM = 1）也会使能双速启动。
- SCS<1:0>（在 OSCCON 寄存器中）= 00。
- CONFIG1H 配置寄存器的 FOSC<2:0> 位被配置为 LP、XT 或 HS 模式。

在以下事件之后，双速启动模式变为有效：

- 上电复位（POR）以及在上电延时定时器（PWRT）延时结束（如果使能）后，或者
- 从休眠中唤醒。

如果外部时钟振荡器被配置为除 LP、XT 或 HS 模式以外的任一模式，那么双速启动将被禁止。这是因为 POR 后或从休眠中退出时，外部时钟振荡器不需要稳定时间。

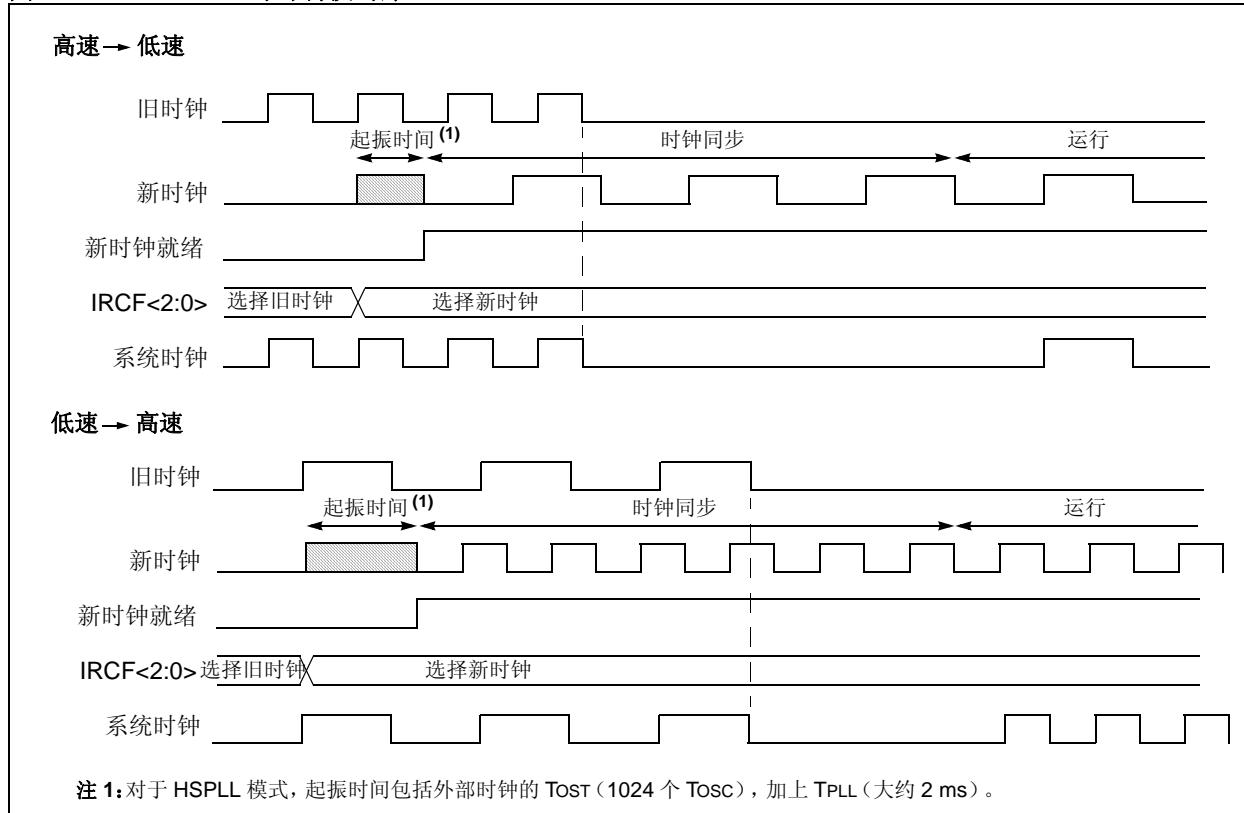
2.10.2 双速启动序列

1. 从上电复位或休眠中唤醒。
2. 使用内部振荡器以 OSCCON 寄存器的 IRCF<2:0> 位设置的频率开始执行指令。
3. OST 使能，计数 1024 个外部时钟周期。
4. OST 超时。外部时钟就绪。
5. OSTS 置 1。
6. 根据图 2-7：“时钟切换时序” 时钟切换结束。

2.10.3 检查双速时钟状态

通过检查 OSCCON 寄存器的 OSTS 位的状态，可以确认单片机是否如 CONFIG1H 配置寄存器中 FOSC<2:0> 位定义的那样依靠外部时钟源运行，还是依靠内部振荡器运行。当外部振荡器未就绪时，OSTS = 0，这指示系统使用内部振荡器运行。

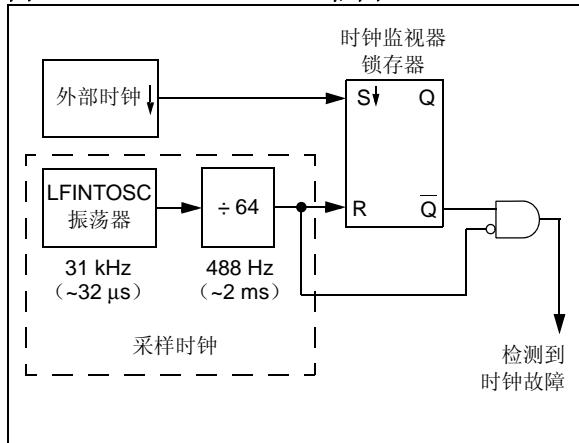
图 2-7: 时钟切换时序



2.11 故障保护时钟监视器

故障保护时钟监视器（FSCM）使得器件在出现外部振荡器故障时仍能继续工作。FSCM 能在振荡器起振延时定时器（OST）延时结束后的任一时刻检测振荡器故障。FSCM 通过将 CONFIG1H 配置寄存器中的 FCMEN 位置 1 来使能。FSCM 可用于所有外部振荡器模式（LP、XT、HS、EC、RC 和 RCIO）。

图 2-8： FSCM 框图



2.11.1 故障保护检测

FSCM 模块通过将外部振荡器与 FSCM 采样时钟比较来检测振荡器故障。将 LFINTOSC 64 分频，就产生了采样时钟。请参见图 2-8。故障检测器内部有一个锁存器。在外部时钟的每个下降沿，锁存器被置 1。在采样时钟的每个上升沿，锁存器被清零。如果已经经过采样时钟的整个半周期，但主时钟仍未变为低电平，则会检测到故障。

2.11.2 故障保护操作

当外部时钟出现故障时，FSCM 将器件时钟切换到内部时钟源，并将 PIR2 寄存器的 OSFIF 标志位置 1。如果 PIE2 寄存器的 OSCFIE 位也置 1，则 OSCFIF 标志置 1 会导致中断发生。器件固件随后会采取措施减轻可能由故障时钟所产生的问题。系统时钟将继续来自内部时钟源，直到器件固件成功重启外部振荡器并切换回外部时钟源进行工作。不会自动转换回发生故障的时钟源。

FSCM 选定的内部时钟源由 OSCCON 寄存器的 IRCF<2:0> 位决定。这使得可以在故障发生前配置内部振荡器。

2.11.3 故障保护条件清除

故障保护条件通过以下事件之一清除：

- 任何复位
- 通过翻转 OSCCON 寄存器的 SCS1 位

这两个条件都会重新启动 OST。OST 运行时，器件将依靠 OSCCON 选定的 INTOSC 工作。OST 超时后，故障保护条件被清除，器件自动切换到外部时钟源。在 OSCFIF 标志清零之前，不需要清除故障保护条件。

2.11.4 复位或从休眠中唤醒

FSCM 设计为能在振荡器起振延时定时器（OST）延时结束后的任一时刻检测振荡器故障。OST 用在从休眠状态唤醒后以及任何类型的复位后。OST 不能在 EC 或 RC 时钟模式下使用，所以一旦复位或唤醒完成，FSCM 就处于活动状态。当 FSCM 被使能时，双速启动也被使能。因此，当 OST 运行时，器件总是处于代码执行阶段。

注： 由于振荡器起振时间范围较大，在振荡器起振期间（即，从复位或休眠中退出后），故障保护电路不处于活动状态。经过一段适当的时间后，用户应检查 OSCCON 寄存器的 OSTS 位，以验证振荡器是否已成功起振以及系统时钟是否切换成功。

图 2-9: FSCM 时序图

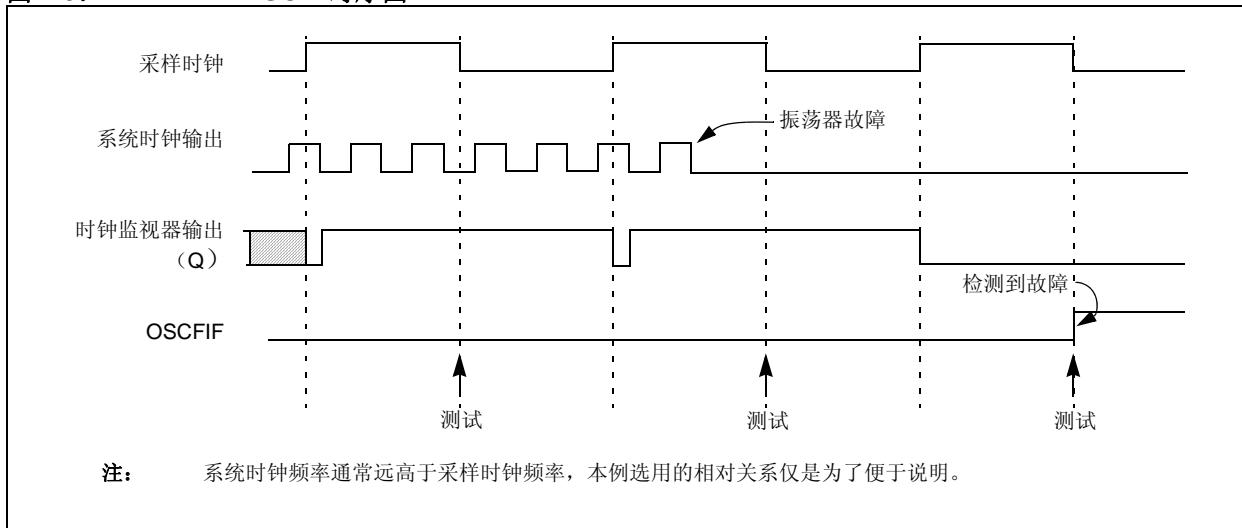


表 2-3: 与时钟源相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	所有其他复位时的值 ⁽¹⁾
CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	—	—
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000x
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0011 q000	0011 q000
OSCTUNE	INTSRC	PLLEN	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	0000 0000	000u uuuu
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	0000 0000	0000 0000
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	0000 0000	0000 0000
IPR2	OSCFIP	—	—	—	—	—	—	—	1111 1111	1111 1111

图注: x = 未知, u = 不变, — = 未实现位 (读为 0)。振荡器模块不使用阴影单元。

注 1: 其他 (非上电) 复位包括在正常工作期间的 MCLR 复位和看门狗定时器复位。

PIC18F2XK20/4XK20

注:

3.0 功耗管理模式

PIC18F2XK20/4XK20 器件总共提供 7 种工作模式，可以更有效地进行功耗管理。这些工作模式提供了多种选择，可在资源受限的应用（即，电池供电的设备）中节省功耗。

功耗管理模式有三种类别：

- 运行模式
- 空闲模式
- 休眠模式

这些类别定义了需要为器件的哪些部分提供时钟，有时还需要定义时钟的速度。运行模式和空闲模式可以使用三种时钟源（主时钟源、辅助时钟源或内部振荡器模块）中的任意一种；而休眠模式则不使用时钟源。

功耗管理模式包括几个由早期的 PIC[®] 单片机器件提供的节省功耗的功能。其中之一就是时钟切换，该功能允许使用 Timer1 振荡器代替主振荡器。节省功耗的功能还包括所有 PIC[®] 单片机都能提供的休眠模式，在该模式下，器件所有的时钟都停止。

3.1 选择功耗管理模式

选择功耗管理模式之前需要先做出两个决定：

- 是否为 CPU 提供时钟源
- 选择何种时钟源

OSCCON 寄存器的 IDLEN 位控制是否为 CPU 提供时钟源，而 SCS<1:0> 位选择时钟源。表 3-1 总结了各个模式下的位设置、时钟源和受影响的模块。

表 3-1：功耗管理模式

模式	OSCCON 位		模块时钟		可用时钟和振荡器源
	IDLEN ⁽¹⁾	SCS<1:0>	CPU	外设	
休眠	0	N/A	关闭	关闭	无——所有时钟被禁止
PRI_RUN	N/A	00	提供时钟	提供时钟	主时钟——LP、XT、HS、HSPLL、RC、EC 和内部振荡器模块 ⁽²⁾ 。 这是正常的全功耗执行模式。
SEC_RUN	N/A	01	提供时钟	提供时钟	辅助时钟——Timer1 振荡器
RC_RUN	N/A	1x	提供时钟	提供时钟	内部振荡器模块 ⁽²⁾
PRI_IDLE	1	00	关闭	提供时钟	主时钟——LP、XT、HS、HSPLL、RC 和 EC
SEC_IDLE	1	01	关闭	提供时钟	辅助时钟——Timer1 振荡器
RC_IDLE	1	1x	关闭	提供时钟	内部振荡器模块 ⁽²⁾

注 1：IDLEN 在执行 SLEEP 指令时反映其值。

2：包含 HFINTOSC 和 HFINTOSC 后分频器以及 LFINTOSC 源。

3.1.1 时钟源

SCS<1:0> 位允许为功耗管理模式在三个时钟源中任选其一。它们是：

- 主时钟，由 FOSC<3:0> 配置位定义
- 辅助时钟（Timer1 振荡器）
- 内部振荡器模块

3.1.2 进入功耗管理模式

可以通过装载 OSCCON 寄存器从一种功耗管理模式切换到另一种功耗管理模式。SCS<1:0> 位选择时钟源并确定使用运行模式还是空闲模式。更改这些位会导致立即切换到一个新的时钟源（假定新时钟源正在运行）。此切换可能会引起时钟转换延时。第 3.1.3 节“时钟转换和状态指示”及其后续章节将会讨论这些问题。

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

更改功耗管理模式并不总是要求设置所有这些位，而是取决于当前的模式和将要切换到的模式。通过在发出 SLEEP 指令之前更改振荡器选择位或更改 IDLEN 位可完成多种模式转换。如果已经正确配置了 IDLEN 位，可能只需执行 SLEEP 指令就可切换到所需的模式。

3.1.3 时钟转换和状态指示

在两个时钟源之间进行转换所需的时间长度是以下三项之和：

- 新时钟的启动时间
- 旧时钟源的 2.5 个周期
- 新时钟源的 2.5 个周期

以下3个标志位用于指明当前的时钟源及其状态。它们是：

- OSTS（在 OSCCON 寄存器中）
- IOFS（在 OSCCON 寄存器中）
- T1RUN（在 T1CON 寄存器中）

通常，当处于给定的功耗管理模式时，这些位中只有一位会置 1。表 3-2 显示了这些标志与工作的主系统时钟源的关系。

表 3-2： 系统时钟指示

OSTS	IOFS	T1RUN	主系统时钟源
1	0	0	主振荡器
0	1	0	HFINTOSC
0	0	1	辅助振荡器
1	1	0	HFINTOSC 作为主时钟
0	0	0	LFINTOSC 或 HFINTOSC 尚未稳定

注 1：执行 SLEEP 指令并不一定会将器件置于休眠模式。它只是作为触发条件，让器件进入休眠模式或一种空闲模式，具体何种模式由 IDLEN 位的设置决定。

3.1.4 SLEEP 命令的多种功能

使用 SLEEP 指令调用功耗管理模式时，具体进入何种模式在该指令执行那一刻由 OSCCON 寄存器的 IDLEN 位的设置决定。当 IDLEN 位清零时，如果执行 SLEEP 指令，所有时钟将停止，功耗达到最低。当 IDLEN 位置 1 时，如果执行 SLEEP 指令，系统时钟会继续向外设提供时钟，但从 CPU 断开。

3.2 运行模式

在运行模式下，内核和外设的时钟均有效。这些运行模式之间的区别就在于时钟源不同。

3.2.1 PRI_RUN 模式

PRI_RUN 模式是单片机的正常全功耗执行模式。除非使能了双速启动（详情请参见第 2.10 节“双速时钟启动模式”），该模式也是器件复位后的默认模式。在此模式下，OSTS 位置 1。如果 HFINTOSC 是主时钟源，并且振荡器稳定，IOFS 位将置 1（见第 2.2 节“振荡器控制”）。

3.2.2 SEC_RUN 模式

SEC_RUN 模式与其他 PIC18 器件提供的“时钟切换”功能兼容。在此模式下，CPU 和外设将 Timer1 振荡器作为时钟源。这允许用户在使用高精度时钟源的情况下仍可获得较低的功耗。

通过将 SCS<1:0> 设置为 01 可以进入 SEC_RUN 模式。当工作于 SEC_RUN 模式时，将发生以下所有情况：

- 主时钟源被切换到 Timer1 振荡器
- 主振荡器被关闭
- T1CON 寄存器的 T1RUN 位被置 1
- OSTS 位被清零

注： Timer1 振荡器应该在进入 SEC_RUN 模式之前就已经运行了。如果 SCS<1:0> 位设置为 01 时，T1OSCEN 位未置 1，则只有 T1OSCEN 位置 1、且 Timer1 振荡器就绪时，才会进入 SEC_RUN 模式。

在从 SEC_RUN 模式转换到 PRI_RUN 模式时，外设和 CPU 继续使用 Timer1 振荡器作为时钟源，直到主时钟启动。当主时钟就绪以后，时钟切换回主时钟（见图 2-7）。当时钟切换完成后，T1RUN 位被清零，OSTS 位被置 1 并且由主时钟提供主系统时钟。只要 T1OSCEN 位置 1，Timer1 振荡器就会继续运行。

3.2.3 RC_RUN 模式

在 RC_RUN 模式下，使用来自 HFINTOSC 多路开关的选择之一，将内部振荡器模块作为 CPU 和外设的时钟源。在此模式下，主振荡器关闭。当 LFINTOSC 是主时钟源时，RC_RUN 模式是所有运行模式中最节省功耗的模式。它非常适用于对定时精度要求不高或者不是一直需要高速时钟的应用。

如果主时钟源为内部振荡器模块（LFINTOSC 或 HFINTOSC），在代码执行期间，PRI_RUN 和 RC_RUN 这两种模式区别不大。但是在进入和退出 RC_RUN 模式时会发生时钟切换延时。因此，如果主时钟源为内部振荡器模块，建议不要使用 RC_RUN 模式。关于时钟切换的详细信息，请参见第 2.9.3 节“时钟切换时序”。

通过将 SCS1 位设为 1 可以进入 RC_RUN 模式。SCS0 位可以为 0 或 1，但应设为 0，以保持与未来器件的软件兼容性。当时钟源从主振荡器切换到 HFINTOSC 多路开关时，主振荡器被关闭并且 OSTS 位被清零。在任何时候修改 IRCF 位可以立即改变时钟速度。

在从 RC_RUN 模式转换到 PRI_RUN 模式时，在主时钟处于启动状态时，器件将继续使用内部振荡器作为时钟源。当主振荡器就绪以后，时钟切换回主时钟。当时钟切换完成后，IOFS 位被清零，OSTS 位被置 1 并且由主振荡器提供主系统时钟。如果满足第 2.5.2 节“HFINTOSC”中描述的任一条件，HFINTOSC 将继续运行。如果满足第 2.5.3 节“LFINTOSC”中描述的任一条件，LFINTOSC 源将继续运行。

3.3 休眠模式

PIC18F2XK20/4XK20 器件的功耗管理休眠模式和所有其他 PIC® 单片机器件提供的传统休眠模式相同。通过清零 IDLEN 位（器件复位时的默认状态）并执行 SLEEP 指令即可进入此模式。这将关闭选定的振荡器（见图 3-1），并将所有的时钟源状态位清零。

从任何其他模式进入休眠模式不需要切换时钟。这是因为单片机一旦进入休眠模式就不需要时钟了。如果选择了 WDT，LFINTOSC 时钟源将继续工作。如果使能了 Timer1 振荡器，它也将继续运行。

当在休眠模式下发生唤醒事件（通过中断、复位或 WDT 超时）时，在时钟源（通过 SCS<1:0> 位选择）就绪之前器件将没有时钟源（见图 3-2），或者如果使能了双速启动或故障保护时钟监视器，它将使用内部振荡器模块作为时钟源（见第 23.0 节“CPU 的特殊功能”）。在这两种情况下，当由主时钟提供器件时钟时，OSTS 位将置 1。这种唤醒不会影响 IDLEN 和 SCS 位。

3.4 空闲模式

空闲模式允许在外设继续工作的同时有选择地关闭控制器的 CPU。选择特定的空闲模式允许用户进一步管理功耗。

如果在执行 SLEEP 指令时，IDLEN 位被置 1，外设将使用由 SCS<1:0> 位选择的时钟源，而 CPU 没有时钟源。时钟源状态位不受影响。将 IDLEN 置 1 并执行 SLEEP 指令可以从给定的运行模式快速切换到相应的空闲模式。

如果选择了 WDT，LFINTOSC 时钟源将继续工作。如果使能了 Timer1 振荡器，它也将继续运行。

由于 CPU 没有执行指令，器件只能通过中断、WDT 超时或复位从空闲模式退出。当发生唤醒事件时，CPU 会在其准备执行代码前延时一个 Tcsd 间隔（表 26-10 中的参数 38）。当 CPU 开始执行代码时，它将沿用当前空闲模式所使用的时钟源。例如，当从 RC_IDLE 模式唤醒时，将使用内部振荡器模块为 CPU 和外设提供时钟（即 RC_RUN 模式）。这种唤醒不会影响 IDLEN 和 SCS 位。

当处于任何空闲模式或休眠模式下时，WDT 超时会导致 WDT 唤醒并进入当前由 SCS<1:0> 位指定的运行模式。

PIC18F2XK20/4XK20

图 3-1: 进入休眠模式的转换时序

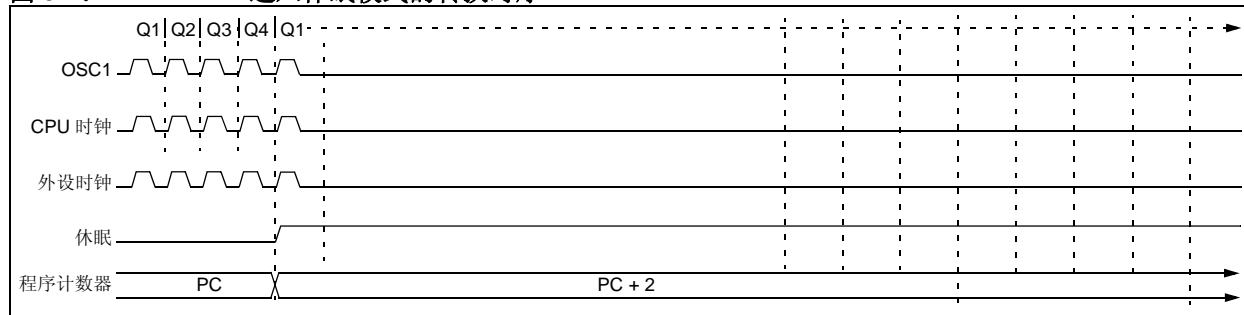
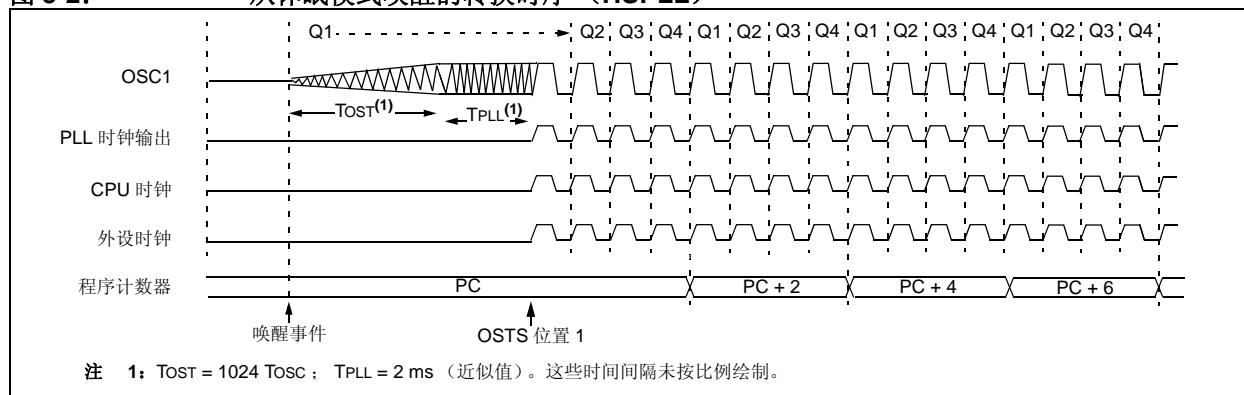


图 3-2: 从休眠模式唤醒的转换时序 (HSPLL)



3.4.1 PRI_IDLE 模式

在三种低功耗空闲模式中，只有该模式不会禁止主器件时钟。由于时钟源不需要“预热”或是从其他振荡器转换过来，对于时序敏感的应用，选用此模式可以使用较精确的主时钟源以最快的速度恢复器件运行。

可以通过将 IDLEN 位置 1 并执行 SLEEP 指令以实现从 PRI_RUN 模式进入 PRI_IDLE 模式。如果器件处于另一种运行模式，首先将 IDLEN 位置 1，然后将 SCS 位清零并执行 SLEEP。虽然 CPU 已被禁止，但外设仍可继续使用由 FOSC<3:0> 配置位指定的主时钟源为其提供时钟信号。OSTS 位保持置 1（见图 3-3）。

当发生唤醒事件时，由主时钟源为 CPU 提供时钟。在唤醒事件和代码执行开始之间需要一个 TCSD 间隔的延时。该延时用来让 CPU 做好执行指令的准备。在唤醒之后，OSTS 位保持置 1 状态。这种唤醒不会影响 IDLEN 和 SCS 位（见图 3-4）。

3.4.2 SEC_IDLE 模式

在 SEC_IDLE 模式下，CPU 被禁止，但外设继续将 Timer1 振荡器作为时钟源。可以通过将 IDLEN 位置 1 并执行 SLEEP 指令从 SEC_RUN 模式进入此模式。如果器件处于另一种运行模式，首先将 IDLEN 位置 1，然后将 SCS<1:0> 位设置为 01 并执行 SLEEP。当时钟源切换到 Timer1 振荡器时，主振荡器被关闭，OSTS 位被清零并且 T1RUN 位被置 1。

当唤醒事件发生时，外设继续将 Timer1 振荡器作为时钟源。唤醒事件发生后经过一个 TCSD 间隔，CPU 开始执行代码并使用 Timer1 振荡器作为其时钟源。这种唤醒不会影响 IDLEN 和 SCS 位。Timer1 振荡器继续运行（见图 3-4）。

注： Timer1 振荡器应该在进入 SEC_IDLE 模式之前就已经运行了。如果执行 SLEEP 指令时，T1OSCEN 位未置 1，那么主系统时钟将继续以先前选定的模式工作，并进入相应的空闲模式（即，PRI_IDLE 或 RC_IDLE）。

图 3-3：进入空闲模式的转换时序

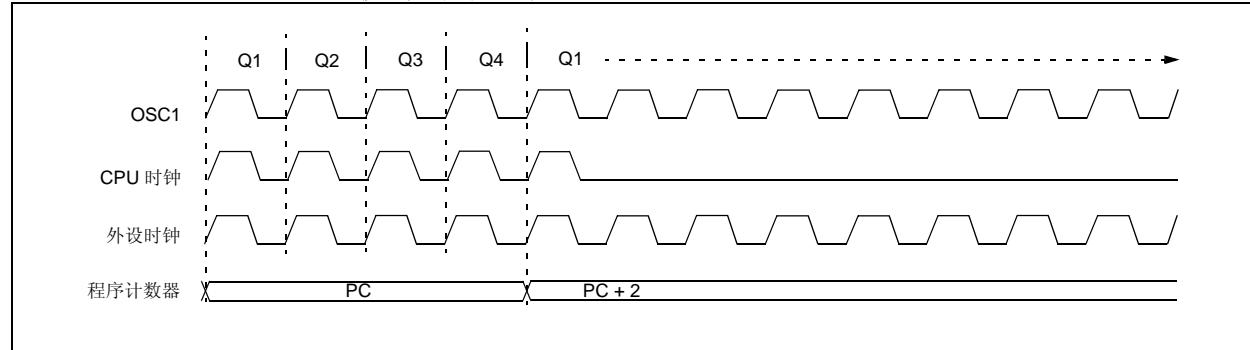
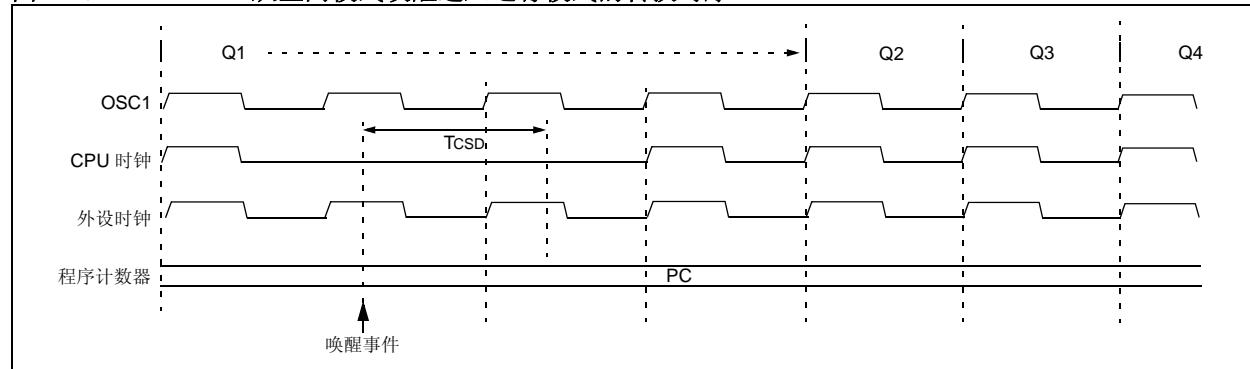


图 3-4：从空闲模式唤醒进入运行模式的转换时序



3.4.3 RC_IDLE 模式

在 RC_IDLE 模式下，CPU 被禁止，但仍继续由使用 HFINTOSC 多路开关输出的内部振荡器模块为外设提供时钟。该模式允许在空闲期间对功耗进行控制。

可以通过将 IDLEN 位置 1 并执行 SLEEP 指令从 RC_RUN 模式进入此模式。如果器件处于另一种运行模式，可以先将 IDLEN 位置 1，然后再将 SCS1 位置 1 并执行 SLEEP。虽然 SCS0 的值常常被忽略，但仍建议将其清零，这将保证与未来器件的软件兼容性。通过在执行 SLEEP 指令之前修改 IRCF 位，可以使用 HFINTOSC 多路开关来选择更高的时钟频率。当时钟源切换到 HFINTOSC 多路开关时，主振荡器被关闭并且 OSTS 位被清零。

如果 IRCF 位被设置为任何非零值，或者 INTSRC 位被置 1，就会使能 HFINTOSC 输出。在一个 TIOBST 间隔（表 26-10 中的参数 39）之后 HFINTOSC 输出将趋于稳定，随后 IOFS 位置 1。外设的时钟继续运行直到 HFINTOSC 时钟源稳定。如果之前 IRCF 位为一个非零值或者在执行 SLEEP 指令之前 INTSRC 已置 1，并且当前 HFINTOSC 时钟源已稳定，IOFS 位将保持置 1 状态。如果 IRCF 和 INTSRC 位全部清零，就不会使能 HFINTOSC 输出，IOFS 位将保持清零状态，此时将不会有当前时钟源的任何指示。

当唤醒事件发生时，外设继续将 HFINTOSC 多路开关输出作为时钟源。在唤醒事件后的 Tcsd 延时之后，CPU 开始执行代码并使用 HFINTOSC 多路开关作为时钟源。唤醒不会影响 IDLEN 和 SCS 位。如果使能了 WDT 或故障保护时钟监视器，LFINTOSC 时钟源将继续运行。

3.5 退出空闲和休眠模式

通过以下任一事件可以触发从休眠模式或任意空闲模式退出：

- 中断
- 复位
- 看门狗定时器超时

本节将讨论从功耗管理模式退出的触发方式。在每种功耗管理模式章节中我们已经讨论过其时钟源子系统的操作（见第 3.2 节“运行模式”、第 3.3 节“休眠模式”和第 3.4 节“空闲模式”）。

3.5.1 通过中断退出

任何可用的中断源都可导致器件从空闲模式或休眠模式退出到运行模式。要使能此功能，必须通过将对应 INTCON 或 PIE 寄存器中的中断源允许位置 1 来允许中断源。如果所需的中断允许位位于 PIE 寄存器中，则还必须将 PEIE 位置 1。当相应的中断标志位置 1 时，触发退出操作。

在通过中断从空闲或休眠模式退出时，将执行紧随在 SLEEP 指令后的指令。然后，如果 INTCON 寄存器的 GIE/GIEH 位置 1，代码将跳转到中断向量处执行，否则代码将继续执行，不进行跳转（见第 9.0 节“中断”）。

唤醒事件之后需要一个固定的 Tcsd 间隔的延时，器件才会退出休眠和空闲模式。CPU 需要此延时来准备执行代码。在延时后的第一个时钟周期重新开始执行指令。

3.5.2 通过 WDT 超时退出

根据 WDT 超时发生时器件所处的不同功耗管理模式会引发不同的操作。

如果器件不在执行代码（所有空闲模式和休眠模式），超时将导致从功耗管理模式退出（见第 3.2 节“运行模式”和第 3.3 节“休眠模式”）。如果器件正在执行代码（所有运行模式），超时将导致 WDT 复位（见第 23.2 节“看门狗定时器（WDT）”）。

WDT 定时器和后分频器可由以下任一事件清零：

- 执行 SLEEP 指令
- 执行 CLRWDT 指令
- 当前选定的时钟源失效（如果使能了故障保护时钟监视器）
- 修改 OSCCON 寄存器中的 IRCF 位（如果内部振荡器模块是器件时钟源）

3.5.3 通过复位退出

通过复位退出休眠和空闲模式会导致代码从地址 0 重新开始执行。更多详细信息，请参见第 4.0 节“复位”。

从复位状态退出到开始执行代码期间的延迟时间由唤醒前后的时钟源以及振荡器的类型决定。表 3-3 中总结了退出延时。

3.5.4 在没有振荡器起振延时的情况下退出
从某些功耗管理模式退出完全不需要 OST 延时。有以下两种情形：

- 主时钟源不停止的 PRI_IDLE 模式
- 主时钟源不是 LP、XT、HS 或 HSPLL 中的任意一种模式

在这些情况下，主时钟源不需要振荡器起振延时，因为它已经在运行（PRI_IDLE），或者它本来就不需要振荡器起振延时（RC、EC、INTOSC 和 INTOSCI 模式）。但是，当器件退出休眠和空闲模式时，在唤醒事件之后仍然需要一个固定的 TCSD 间隔的延时，以便让 CPU 准备好执行代码。在延时后的第一个时钟周期重新开始执行指令。

表 3-3：通过复位从休眠模式或任何空闲模式唤醒的退出延时（按时钟源分类）

唤醒之前的时钟源	唤醒之后的时钟源	退出延时	时钟就绪状态位 (OSCCON)
主器件时钟 (PRI_IDLE 模式)	LP、XT 或 HS	TCSD ⁽¹⁾	OSTS
	HSPLL		
	EC 或 RC		IOFS
	HFINTOSC ⁽²⁾		
T1OSC 或 LFINTOSC ⁽¹⁾	LP、XT 或 HS	TOST ⁽³⁾	OSTS
	HSPLL	TOST + t _{PLL} ⁽³⁾	
	EC 或 RC	TCSD ⁽¹⁾	IOFS
	HFINTOSC ⁽¹⁾	TIOBST ⁽⁴⁾	
HFINTOSC ⁽²⁾	LP、XT 或 HS	TOST ⁽⁴⁾	OSTS
	HSPLL	TOST + t _{PLL} ⁽³⁾	
	EC 或 RC	TCSD ⁽¹⁾	IOFS
	HFINTOSC ⁽¹⁾	无	
无 (休眠模式)	LP、XT 或 HS	TOST ⁽³⁾	OSTS
	HSPLL	TOST + t _{PLL} ⁽³⁾	
	EC 或 RC	TCSD ⁽¹⁾	IOFS
	HFINTOSC ⁽¹⁾	TIOBST ⁽⁴⁾	

注 1: 当从休眠模式和所有空闲模式唤醒时都需要 TCSD（参数 38）延时，该延时与所需的其他延时并行运行（见第 3.4 节“空闲模式”）。复位时，HFINTOSC 默认值为 1 MHz。

2: 包括 HFINTOSC 16 MHz 时钟源和后分频器产生的频率。

3: TOST 是振荡器起振定时器的延迟时间（参数 32）。t_{PLL} 是 PLL 锁定延时定时器的延迟时间（参数 F12）。

4: 在 HFINTOSC 稳定周期 TIOBST（参数 39）延时期间，代码继续执行。

PIC18F2XK20/4XK20

注:

4.0 复位

PIC18F2XK20/4XK20器件有以下几种不同类型的复位：

- 上电复位（POR）
- 正常工作期间的 MCLR 复位
- 功耗管理模式下的 MCLR 复位
- 看门狗定时器（WDT）复位（执行程序期间）
- 可编程欠压复位（BOR）
- RESET 指令
- 堆栈满复位
- 堆栈下溢复位

本节讨论了由 MCLR、POR 和 BOR 产生的复位，并涉及各种起振定时器的工作方式。堆栈复位事件将在第 5.1.2.4 节“堆栈满和下溢复位”中讨论。WDT 复位将在第 23.2 节“看门狗定时器（WDT）”中讨论。

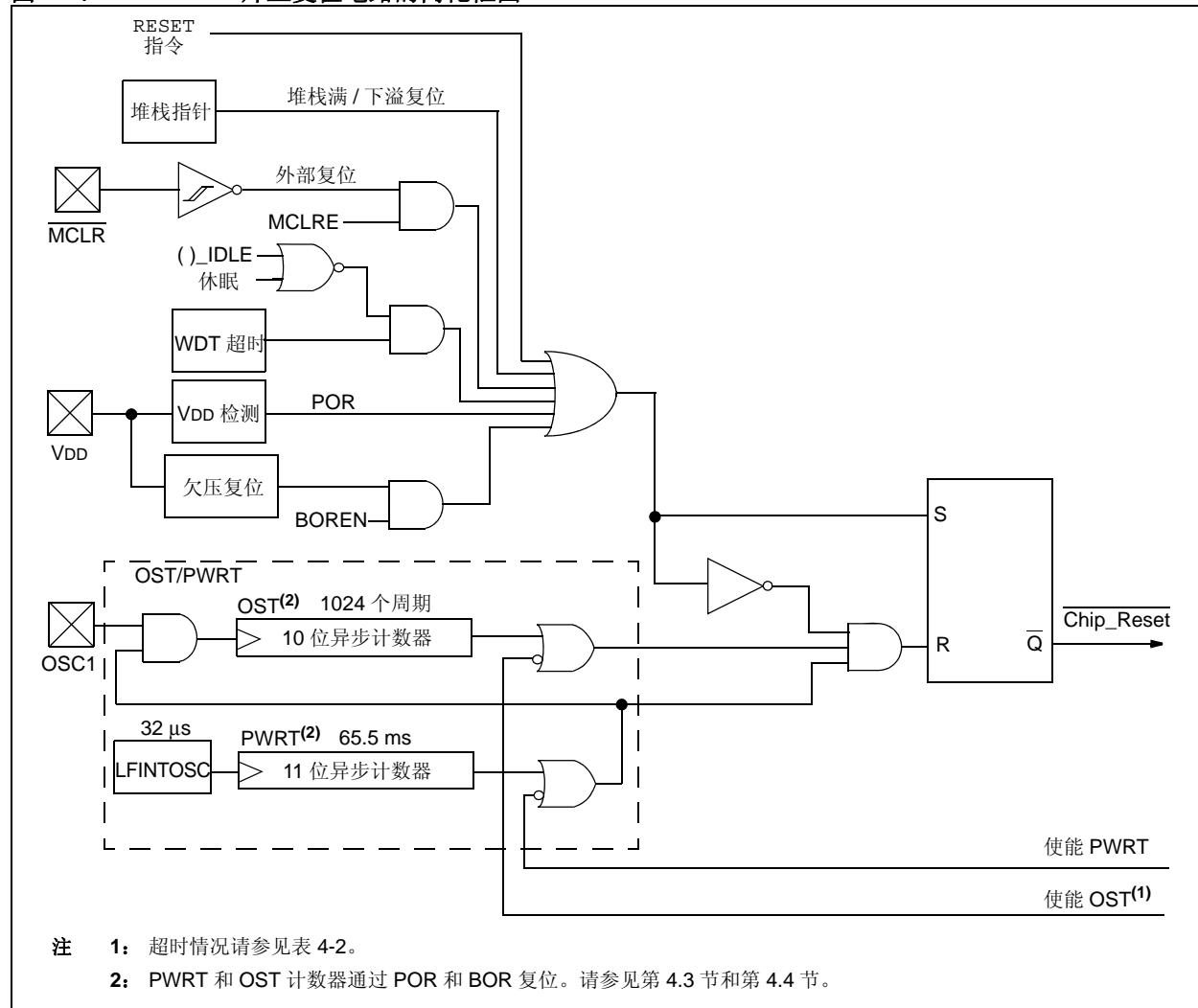
图 4-1 给出了片上复位电路的简化框图。

4.1 RCON 寄存器

通过 RCON 寄存器（寄存器 4-1）跟踪器件复位事件。该寄存器的低 5 位表明是否已经发生了特定的复位事件。在大多数情况下，只能通过事件将这些位清零，而且必须在事件发生后由应用程序将它们置 1。需要读取所有这些标志位来确定刚发生的复位的类型。在第 4.6 节“寄存器的复位状态”中对此进行了更详细的说明。

RCON 寄存器还有设置中断优先级的控制位（IPEN）和对 BOR 进行软件控制的控制位（SBOREN）。在第 9.0 节“中断”中讨论了中断优先级。在第 4.4 节“欠压复位（BOR）”中讨论了 BOR。

图 4-1：片上复位电路的简化框图



PIC18F2XK20/4XK20

寄存器 4-1: RCON: 复位控制寄存器

R/W-0	R/W-1	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN ⁽¹⁾	—	\overline{RI}	\overline{TO}	\overline{PD}	$\overline{POR}^{(2)}$	\overline{BOR}
bit 7	bit 0						

图注:

R = 可读位

-n = POR 时的值

W = 可写位

1 = 置 1

U = 未实现位, 读为 0

0 = 清零

x = 未知

bit 7

IPEN: 中断优先级使能位

1 = 使能中断优先级

0 = 禁止中断优先级 (PIC16CXXX 兼容模式)

bit 6

SBOREN: BOR 软件使能位⁽¹⁾

如果 $\overline{BORN}<1:0> = 01$:

1 = 使能 BOR

0 = 禁止 BOR

如果 $\overline{BORN}<1:0> = 00, 10$ 或 11 :

该位被禁止并读为 0。

bit 5

未实现: 读为 0

bit 4

RI: RESET 指令标志位

1 = 未执行 RESET 指令 (由固件置 1 或在上电复位时置 1)

0 = 执行了 RESET 指令, 导致器件复位 (发生代码执行复位后必须由固件置 1)

bit 3

TO: 看门狗超时标志位

1 = 通过上电、CLRWDAT 指令或 SLEEP 指令置 1

0 = 发生了 WDT 超时

bit 2

PD: 掉电检测标志位

1 = 通过上电或 CLRWDAT 指令置 1

0 = 通过执行 SLEEP 指令置 1

bit 1

POR: 上电复位状态位⁽²⁾

1 = 未发生上电复位

0 = 发生了上电复位 (发生上电复位后必须用软件置 1)

bit 0

BOR: 欠压复位状态位⁽³⁾

1 = 未发生欠压复位 (只能由固件置 1)

0 = 发生了欠压复位 (发生 POR 或欠压复位后必须由固件置 1)

注 1: 当 CONFIG2L[2:1] = 01 时, SBOREN 复位状态为 1; 否则, SBOREN 复位状态为 0。

2: POR 的实际复位值由器件复位的类型决定。更多信息, 请参见该寄存器下方的“注”和第 4.6 节“寄存器的复位状态”。

3: 请参见表 4-3。

注 1: 当 \overline{BOR} 为 0 并且 POR 为 1 时 (假定在 POR 之后立即用固件将 POR 和 BOR 设为 1), 可以说已发生了欠压复位。

2: 建议在检测到上电复位后, 将 POR 置 1, 以便继续检测后续的上电复位。

4.2 主复位 (MCLR)

MCLR 引脚提供了触发器件外部复位的方法。将该引脚拉低可以产生复位信号。这些器件在 MCLR 复位路径上有一个噪声滤波器，该滤波器可以检测并滤除小的干扰脉冲。

任何内部复位，包括 WDT 复位，均不能将 MCLR 引脚驱动为低电平。

在 PIC18F2XK20/4XK20 器件中，可以用 MCLRE 配置位禁止 MCLR 输入。当禁止 MCLR 时，该引脚将成为一个数字输入引脚。更多信息，请参见第 10.6 节“PORTE、TRISE 和 LATE 寄存器”。

4.3 上电复位 (POR)

只要当 VDD 上升到高于某个门限时，就会在片上产生上电复位脉冲。这使得 VDD 达到满足器件正常工作的数值时，器件会以初始化状态启动。

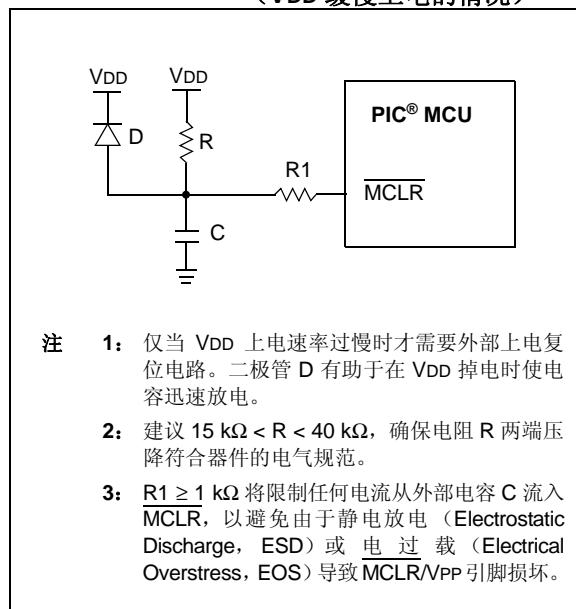
为了利用 POR 电路，需要将 MCLR 引脚通过一个电阻连接到 VDD。这样可以省去产生上电复位延时通常所需的外部 RC 元件。VDD 的最小上升速率已指定（参数 D004）。上升速率缓慢的情况，请参见图 4-2。

当器件开始正常工作（即，退出复位状态）时，器件的工作参数（电压、频率和温度等）必须得到满足，以确保其正常工作。如果不满足这些条件，那么器件必须保持在复位状态，直到满足工作条件为止。

POR 事件由 RCON 寄存器的 POR 位捕捉。每当发生 POR 时，该位的状态就会被设为 0；任何其他复位事件均不能改变它。任何硬件事件均不能将 POR 复位为 1。要捕捉多个事件，用户必须在 POR 之后用软件手动将该位设为 1。

图 4-2:

外部上电复位电路
(VDD 缓慢上电的情况)



- 注
- 1: 仅当 VDD 上电速率过慢时才需要外部上电复位电路。二极管 D 有助于在 VDD 掉电时使电容迅速放电。
 - 2: 建议 $15 \text{ k}\Omega < R < 40 \text{ k}\Omega$ ，确保电阻 R 两端压降符合器件的电气规范。
 - 3: $R1 \geq 1 \text{ k}\Omega$ 将限制任何电流从外部电容 C 流入 MCLR，以避免由于静电放电 (Electrostatic Discharge, ESD) 或 电 过 载 (Electrical Overstress, EOS) 导致 MCLR/VPP 引脚损坏。

4.4 欠压复位 (BOR)

PIC18F2XK20/4XK20 器件带有一个 BOR 电路，它将为用户提供一系列配置和节能选项。BOR 由 CONFIG2L 配置寄存器的 BORV<1:0> 和 BOREN<1:0> 位控制。总共有 4 种 BOR 配置，归纳在表 4-1 中。

BOR 门限值由 BORV<1:0> 位设置。如果使能了 BOR (BOREN<1:0> 为除 00 以外的任何值)，只要 VDD 低于 VBOR (参数 D005) 值的时间大于 TBOR (参数 35) 就会复位器件。如果 VDD 降到 VBOR 以下的时间小于 TBOR，器件是否发生复位不确定。芯片将保持欠压复位状态，直至 VDD 上升到 VBOR 以上。

如果使能了上电延时定时器，则它将在 VDD 上升到超过 VBOR 之后开始工作，并使芯片在延时 TPWRT (参数 33) 期间保持复位。如果在上电延时定时器运行过程中，VDD 电压降到 VBOR 以下，芯片将重新回到欠压复位状态并且初始化上电延时定时器。一旦 VDD 电压上升到 VBOR 以上，上电延时定时器将重新执行延时。

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

BOR 电路有输出送入 POR 电路，并在 BOR 的工作电压范围内重新激活 POR。POR 的提前重新激活可确保在 VDD 低于 BOR 电路的工作电压范围时，器件将保持在复位状态。

4.4.1 检测 BOR

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

4.4.2 用软件使能 BOR

当 BOREN<1:0> = 01 时，用户可以用软件使能或禁止 BOR。这通过使用 RCON 寄存器的 SBOREN 控制位实现。如前所述，将 SBOREN 置 1 可使能 BOR。清零 SBOREN 将完全禁止 BOR。SBOREN 位只在该模式下工作；否则读为 0。

用软件控制 BOR 位可使用户能更灵活地定制应用程序以使其适应环境，而无需通过对器件再编程来更改 BOR 配置。它还允许用户通过减少 BOR 消耗的电流，用软件调节器件的功耗。虽然 BOR 的电流通常很小，但是它可能对低功耗应用有一些影响。

注： 即使当 BOR 受软件控制时，BOR 复位电压仍将由 BORV<1:0> 配置位设置。该值不能用软件更改。

4.4.3 在休眠模式下禁止 BOR

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

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

4.4.4 最小 BOR 使能时间

如果没有任何需要固定参考电压 (FVR) 的外设处于工作状态，则使能 BOR 的同时也将使能 FVR。只有在 FVR 稳定后，BOR 才变为有效。因此，要确保 BOR 保护，当用软件使能 BOR 或从休眠模式唤醒后自动使能 BOR 时，必须考虑 FVR 稳定时间。如果在 FVR 稳定前用软件或通过重新进入休眠模式禁止了 BOR，则 BOR 电路将不会检测 BOR 条件。CVRCON2 寄存器的 FVRST 位可用于确定 FVR 稳定性。

表 4-1：BOR 配置

BOR 配置		SBOREN 的状态 (RCON<6>)	BOR 操作
BOREN1	BORENO		
0	0	不可用	禁止 BOR；必须通过对配置位再编程来使能 BOR。
0	1	可用	用软件使能 BOR；工作模式由 SBOREN 控制。
1	0	不可用	用硬件在运行和空闲模式下使能 BOR，在休眠模式下禁止 BOR。
1	1	不可用	用硬件使能 BOR；必须通过对配置位再编程来禁止 BOR。

4.5 器件复位定时器

PIC18F2XK20/4XK20 器件包含了三个独立的片上定时器，有助于调节上电复位过程。它们的主要功能是确保在代码执行之前器件时钟稳定。这些定时器是：

- 上电延时定时器（PWRT）
- 振荡器起振定时器（OST）
- PLL 锁定延时定时器

4.5.1 上电延时定时器（PWRT）

PIC18F2XK20/4XK20 器件的上电延时定时器（PWRT）是一个 11 位计数器，它使用 LFINTOSC 时钟源作为时钟输入。该定时器可产生大约 $2048 \times 32 \mu\text{s} = 65.6 \text{ ms}$ 的时间间隔。PWRT 计数期间，器件保持在复位状态。

上电延时时间取决于 LFINTOSC 时钟，并且由于温度和工艺的不同，不同器件的延迟时间也将各不相同。详情请参见直流参数 33。

通过清零 PWRTE_N 配置位可使能 PWRT。

4.5.2 振荡器起振定时器（OST）

在 PWRT 延时（参数 33）结束以后，由振荡器起振定时器（OST）提供一个 1024 振荡周期（来自 OSC1 输入）的延时，从而确保晶振或谐振器的起振和稳定工作。

只有在 XT、LP、HS 和 HSPLL 模式下，并且仅当发生上电复位或从所有功耗管理模式退出（停止外部振荡器）时，才启动 OST 延时。

4.5.3 PLL 锁定延时定时器

当在 PLL 模式下使能 PLL 时，上电复位后的延时时序与其他振荡器模式略有不同。在 PLL 模式下需要使用一个独立的定时器来提供一段足够让 PLL 锁定主振荡器频率的固定延时。PLL 锁定延时（TPLL）通常为 2 ms，且在振荡器起振延时后发生。

4.5.4 延时时序

上电延时时序如下：

1. POR 脉冲清零后，启动 PWRT 延时（如果使能）。
2. 然后，OST 被激活。

总延迟时间取决于振荡器配置和 PWRT 的状态。图 4-3、图 4-4、图 4-5、图 4-6 和图 4-7 各自描述了不同的上电延时时序，其中上电延时定时器被使能，并且器件工作在 HS 振荡器模式下。图 4-3 到图 4-6 也适用于在 XT 或 LP 模式下工作的器件。对于工作在 RC 模式下且禁止了 PWRT 的器件，将根本没有延时。

由于延时是由 POR 脉冲触发的，因此如果 MCLR 保持足够长时间的低电平，所有延时将结束，之后将 MCLR 电平拉高后器件将立即开始执行程序（图 4-5）。这对于测试或同步多个并行工作的 PIC18FXXK20 器件来说是非常有用的。

表 4-2：不同情形下的延时

振荡器配置	上电复位 ⁽²⁾ 和 欠压复位		从功耗管理模式退出
	PWRTE _N = 0	PWRTE _N = 1	
HSPLL	66 ms ⁽¹⁾ + 1024 Tosc + 2 ms ⁽²⁾	1024 Tosc + 2 ms ⁽²⁾	1024 Tosc + 2 ms ⁽²⁾
HS、XT 和 LP	66 ms ⁽¹⁾ + 1024 Tosc	1024 Tosc	1024 Tosc
EC 和 ECIO	66 ms ⁽¹⁾	—	—
RC 和 RCIO	66 ms ⁽¹⁾	—	—
INTIO1 和 INTIO2	66 ms ⁽¹⁾	—	—

注 1: 66 ms (65.5 ms) 是上电延时定时器（PWRT）延迟时间的标称值。

2: 2 ms 是 PLL 锁定所需的标称时间。

PIC18F2XK20/4XK20

图 4-3: 上电延时时序 (MCLR 连接到 VDD, VDD 电压上升时间 < TPWRT)

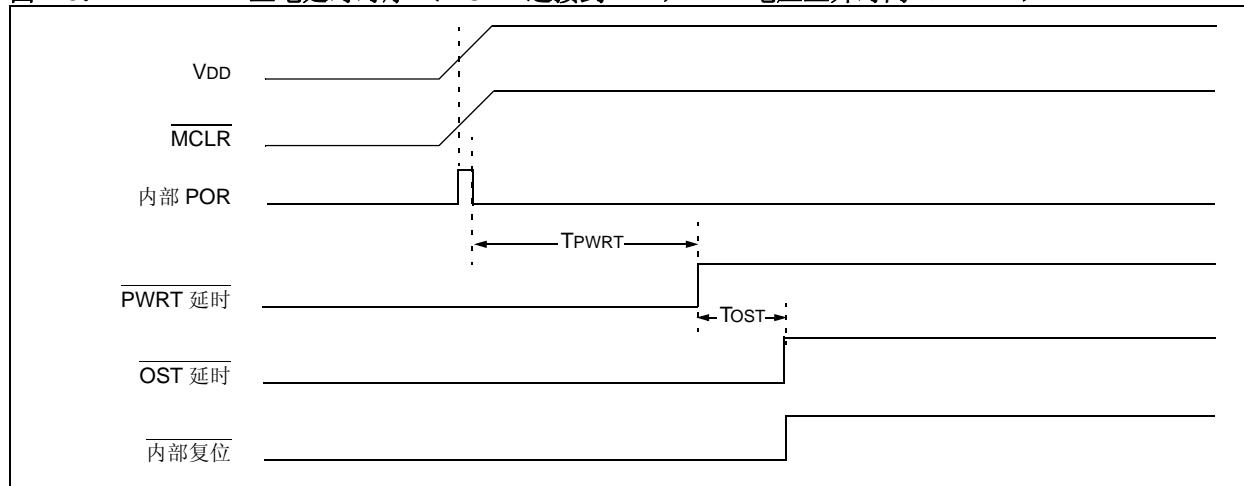


图 4-4: 上电延时时序 (MCLR 未连接到 VDD): 情形 1

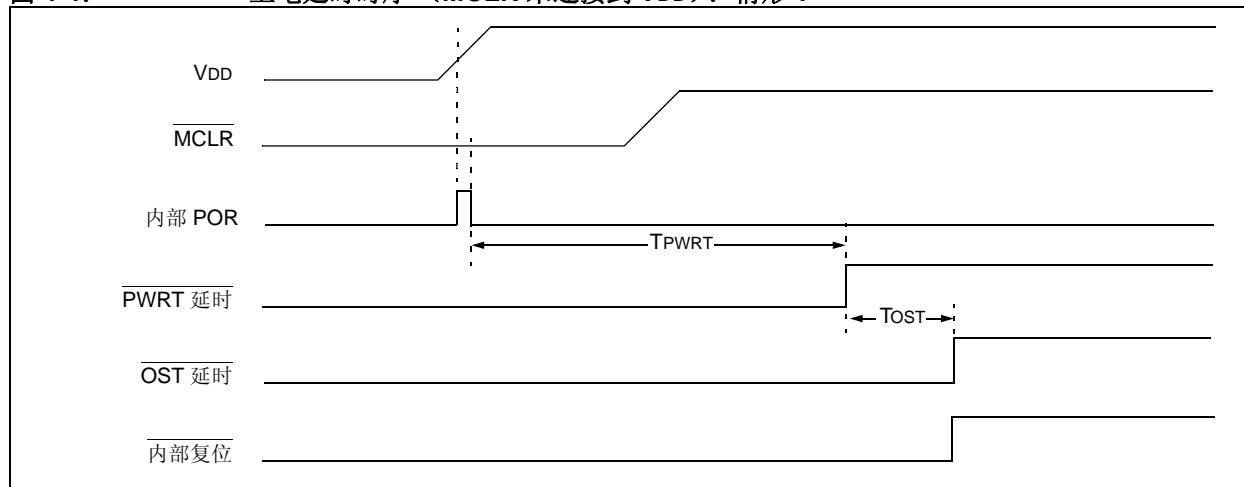


图 4-5: 上电延时时序 (MCLR 未连接到 VDD): 情形 2

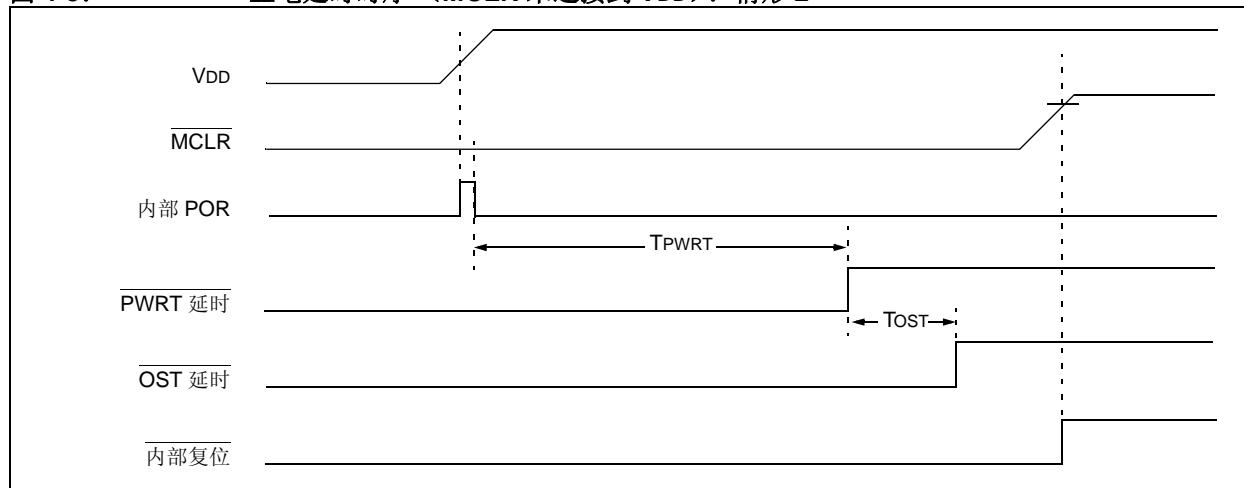


图 4-6:

缓慢上升时间 (MCLR 连接到 VDD, VDD 电压上升时间 > TPWRT)

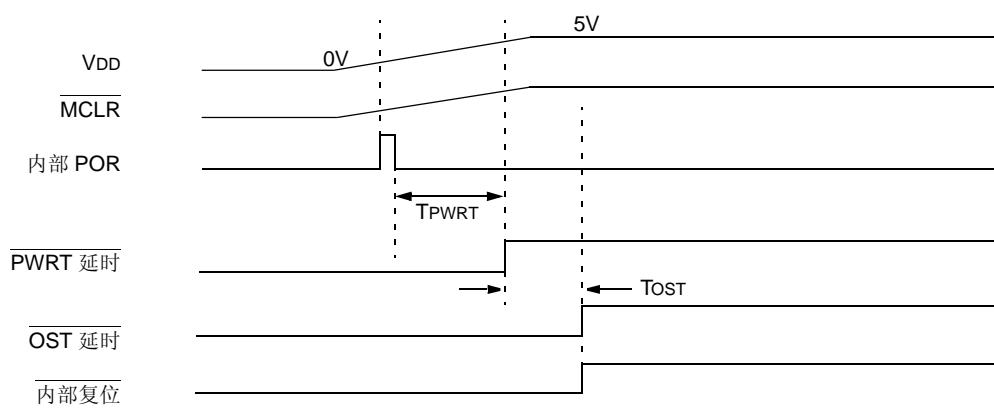
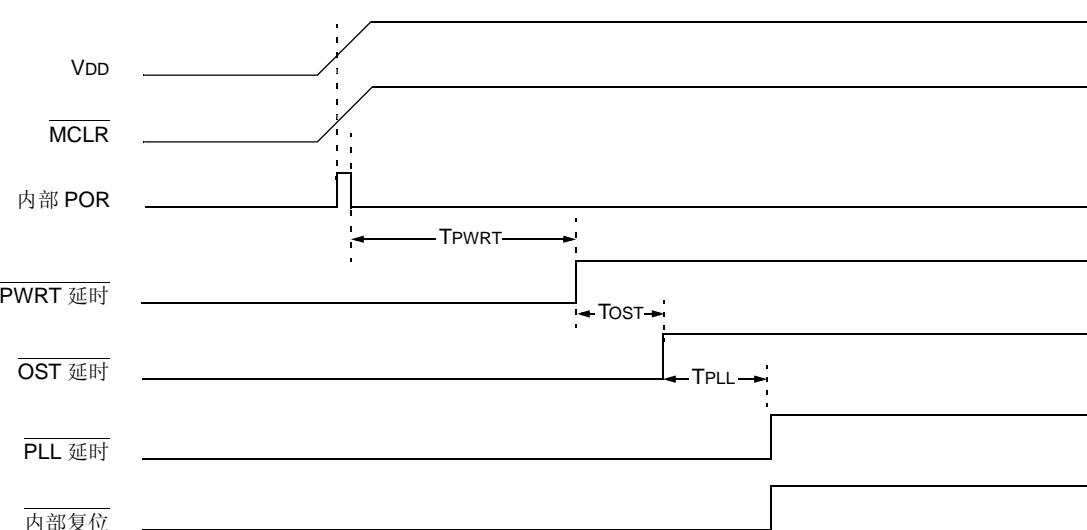


图 4-7:

在 PLL 使能时 POR 的延时时序 (MCLR 连接到 VDD)



注:

$T_{OST} = 1024$ 个时钟周期。
 $T_{PLL} \approx 2 \text{ ms}$ (最大), PWRT 定时器延时的前三个阶段。

4.6 寄存器的复位状态

一些寄存器不受复位的影响。在 POR 时这些寄存器的状态不确定，而在其他复位时它们的状态不变。而所有其他寄存器则根据不同的复位类型被强制为“复位状态”。

大多数寄存器不受 WDT 唤醒的影响，因为这被视为恢复正常工作。如表 4-3 所示，RCON 寄存器中的状态位 (\overline{RI} 、 \overline{TO} 、 \overline{PD} 、 \overline{POR} 和 \overline{BOR}) 在不同的复位情形下会分别被置 1 或清零。可在软件中使用这些位判断复位的性质。

表 4-3: RCON 寄存器的状态位、含义以及初始化状态

条件	程序计数器	RCON 寄存器						STKPTR 寄存器	
		SBOREN	\overline{RI}	\overline{TO}	\overline{PD}	\overline{POR}	\overline{BOR}	STKFUL	STKUNF
上电复位	0000h	1	1	1	1	0	0	0	0
RESET 指令	0000h	u ⁽²⁾	0	u	u	u	u	u	u
欠压复位	0000h	u ⁽²⁾	1	1	1	u	0	u	u
功耗管理运行模式下的 MCLR 复位	0000h	u ⁽²⁾	u	1	u	u	u	u	u
功耗管理空闲和休眠模式下的 MCLR 复位	0000h	u ⁽²⁾	u	1	0	u	u	u	u
全功耗或功耗管理运行模式下的 WDT 超时	0000h	u ⁽²⁾	u	0	u	u	u	u	u
全功耗执行期间的 MCLR 复位	0000h	u ⁽²⁾	u	u	u	u	u	u	u
堆栈满复位 (STVREN = 1)	0000h	u ⁽²⁾	u	u	u	u	u	1	u
堆栈下溢复位 (STVREN = 1)	0000h	u ⁽²⁾	u	u	u	u	u	u	1
堆栈下溢错误 (不是真正的复位, STVREN = 0)	0000h	u ⁽²⁾	u	u	u	u	u	u	1
功耗管理空闲或休眠模式下的 WDT 超时	PC + 2	u ⁽²⁾	u	0	0	u	u	u	u
通过中断从功耗管理模式退出	PC + 2 ⁽¹⁾	u ⁽²⁾	u	u	0	u	u	u	u

图注: u = 不变

注 1: 当器件被中断唤醒且 GIEH 或 GIEL 置 1 时, PC 装入中断向量 (008h 或 0018h)。

2: 当软件使能 BOR (BOREN<1:0> 配置位 = 01) 时, SBOREN 的复位状态为 1 且所有其他复位不能改变该状态。否则, 其复位状态为 0。

表 4-4 描述了所有特殊功能寄存器的复位状态。可以将这些复位状态分类为上电和欠压复位、主复位、WDT 复位以及 WDT 唤醒。

表 4-4: 所有寄存器的初始化状态

寄存器	适用器件		上电复位, 欠压复位	MCLR 复位, WDT 复位, RESET 指令, 堆栈复位	通过 WDT 或 中断唤醒器件
TOSU	PIC18F2XK20	PIC18F4XK20	---0 0000	---0 0000	---0 uuuu ⁽³⁾
TOSH	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu ⁽³⁾
TOSL	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu ⁽³⁾
STKPTR	PIC18F2XK20	PIC18F4XK20	00-0 0000	uu-0 0000	uu-u uuuu ⁽³⁾
PCLATU	PIC18F2XK20	PIC18F4XK20	---0 0000	---0 0000	---u uuuu
PCLATH	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
PCL	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	PC + ⁽²⁾
TBLPTRU	PIC18F2XK20	PIC18F4XK20	--00 0000	--00 0000	--uu uuuu
TBLPTRH	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
TABLAT	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
PRODH	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	PIC18F2XK20	PIC18F4XK20	0000 000x	0000 000u	uuuu uuuu ⁽¹⁾
INTCON2	PIC18F2XK20	PIC18F4XK20	1111 -1-1	1111 -1-1	uuuu -u-u ⁽¹⁾
INTCON3	PIC18F2XK20	PIC18F4XK20	11-0 0-00	11-0 0-00	uu-u u-uu ⁽¹⁾
INDF0	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
POSTINC0	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
POSTDEC0	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
PREINC0	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
PLUSW0	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
FSR0H	PIC18F2XK20	PIC18F4XK20	---- 0000	---- 0000	---- uuuu
FSR0L	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
POSTINC1	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
POSTDEC1	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
PREINC1	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
PLUSW1	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A

图注: u = 不变, x = 未知, - = 未实现位 (读为 0), q = 值取决于具体条件。

阴影单元表示不适用于指定器件的状态。

注 1: INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 (引起唤醒)。

2: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, PC 装入中断向量 (0008h 或 0018h)。

3: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, 用 PC 的当前值更新 TOSU、TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。

4: 具体条件下的复位值, 请参见表 4-3。

5: 根据所选的振荡器模式使能 PORTA、LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚, 则它们将被禁止并读为 0。

6: 如果 CONFIG3H 的 PBADEN 位为 0, 则 ANSELH 寄存器的所有位初始化为 0。

PIC18F2XK20/4XK20

表 4-4: 所有寄存器的初始化状态 (续)

寄存器	适用器件		上电复位, 欠压复位	MCLR 复位, WDT 复位, RESET 指令, 堆栈复位	通过 WDT 或 中断唤醒器件
FSR1H	PIC18F2XK20	PIC18F4XK20	---- 0000	---- 0000	---- uuuu
FSR1L	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	PIC18F2XK20	PIC18F4XK20	---- 0000	---- 0000	---- uuuu
INDF2	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
POSTINC2	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
POSTDEC2	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
PREINC2	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
PLUSW2	PIC18F2XK20	PIC18F4XK20	N/A	N/A	N/A
FSR2H	PIC18F2XK20	PIC18F4XK20	---- 0000	---- 0000	---- uuuu
FSR2L	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	PIC18F2XK20	PIC18F4XK20	--x xxxx	--u uuuu	--u uuuu
TMR0H	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
TMR0L	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu
OSCCON	PIC18F2XK20	PIC18F4XK20	0011 qq00	0011 qq00	uuuu uuuu
HLVDCON	PIC18F2XK20	PIC18F4XK20	0-00 0101	0-00 0101	u-uu uuuu
WDTCON	PIC18F2XK20	PIC18F4XK20	---- --0	---- --0	---- --u
RCON ⁽⁴⁾	PIC18F2XK20	PIC18F4XK20	0q-1 11q0	0u-q qquu	uu-u qquu
TMR1H	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	PIC18F2XK20	PIC18F4XK20	0000 0000	u0uu uuuu	uuuu uuuu
TMR2	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
PR2	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	1111 1111
T2CON	PIC18F2XK20	PIC18F4XK20	-000 0000	-000 0000	-uuu uuuu
SSPBUF	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPADD	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
SSPCON1	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
SSPCON2	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu

图注: u = 不变, x = 未知, - = 未实现位 (读为 0), q = 值取决于具体条件。

阴影单元表示不适用于指定器件的状态。

- 注 1: INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 (引起唤醒)。
 2: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, PC 装入中断向量 (0008h 或 0018h)。
 3: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, 用 PC 的当前值更新 TOSU、TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。
 4: 具体条件下的复位值, 请参见表 4-3。
 5: 根据所选的振荡器模式使能 PORTA、LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚, 则它们将被禁止并读为 0。
 6: 如果 CONFIG3H 的 PBADEN 位为 0, 则 ANSELH 寄存器的所有位初始化为 0。

表 4-4: 所有寄存器的初始化状态 (续)

寄存器	适用器件		上电复位, 欠压复位	MCLR 复位, WDT 复位, RESET 指令, 堆栈复位	通过 WDT 或 中断唤醒器件
ADRESH	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADRESL	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON0	PIC18F2XK20	PIC18F4XK20	--00 0000	--00 0000	--uu uuuu
ADCON1	PIC18F2XK20	PIC18F4XK20	--00 0qqq	--00 0qqq	--uu uuuu
ADCON2	PIC18F2XK20	PIC18F4XK20	0-00 0000	0-00 0000	u-uu uuuu
CCPR1H	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
CCPR2H	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP2CON	PIC18F2XK20	PIC18F4XK20	--00 0000	--00 0000	--uu uuuu
PSTRCON	PIC18F2XK20	PIC18F4XK20	---0 0001	---0 0001	---u uuuu
BAUDCON	PIC18F2XK20	PIC18F4XK20	0100 0-00	0100 0-00	uuuu u-uu
PWM1CON	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
ECCP1AS	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
CVRCON	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
CVRCON2	PIC18F2XK20	PIC18F4XK20	00-- ----	00-- ----	uu-- -----
TMR3H	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	PIC18F2XK20	PIC18F4XK20	0000 0000	uuuu uuuu	uuuu uuuu
SPBRGH	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
SPBRG	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
RCREG	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
TXREG	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
TXSTA	PIC18F2XK20	PIC18F4XK20	0000 0010	0000 0010	uuuu uuuu
RCSTA	PIC18F2XK20	PIC18F4XK20	0000 000x	0000 000x	uuuu uuuu
EEADR	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
EEADRH	PIC18F26K20	PIC18F46K20	---- --00	---- --00	---- --uu
EEDATA	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
EECON2	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	0000 0000
EECON1	PIC18F2XK20	PIC18F4XK20	xx-0 x000	uu-0 u000	uu-0 u000

图注: u = 不变, x = 未知, - = 未实现位 (读为 0), q = 值取决于具体条件。
阴影单元表示不适用于指定器件的状态。

- 注 1: INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 (引起唤醒)。
 2: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, PC 装入中断向量 (0008h 或 0018h)。
 3: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, 用 PC 的当前值更新 TOSU、TOSH 和 Tosl。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。
 4: 具体条件下的复位值, 请参见表 4-3。
 5: 根据所选的振荡器模式使能 PORTA、LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚, 则它们将被禁止并读为 0。
 6: 如果 CONFIG3H 的 PBADEN 位为 0, 则 ANSELH 寄存器的所有位初始化为 0。

PIC18F2XK20/4XK20

表 4-4: 所有寄存器的初始化状态 (续)

寄存器	适用器件		上电复位, 欠压复位	MCLR 复位, WDT 复位, RESET 指令, 堆栈复位	通过 WDT 或 中断唤醒器件
IPR2	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu
PIR2	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu ⁽¹⁾
PIE2	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
IPR1	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu
	PIC18F2XK20	PIC18F4XK20	-111 1111	-111 1111	-uuu uuuu
PIR1	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu ⁽¹⁾
	PIC18F2XK20	PIC18F4XK20	-000 0000	-000 0000	-uuu uuuu ⁽¹⁾
PIE1	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
	PIC18F2XK20	PIC18F4XK20	-000 0000	-000 0000	-uuu uuuu
OSCTUNE	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
TRISE	PIC18F2XK20	PIC18F4XK20	---- -111	---- -111	---- -uuu
TRISD	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu
TRISC	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu
TRISB	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu
TRISA ⁽⁵⁾	PIC18F2XK20	PIC18F4XK20	1111 1111 ⁽⁵⁾	1111 1111 ⁽⁵⁾	uuuu uuuu ⁽⁵⁾
LATE	PIC18F2XK20	PIC18F4XK20	---- -xxxx	---- -uuu	---- -uuu
LATD	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA ⁽⁵⁾	PIC18F2XK20	PIC18F4XK20	xxxx xxxx ⁽⁵⁾	uuuu uuuu ⁽⁵⁾	uuuu uuuu ⁽⁵⁾
PORTE	PIC18F2XK20	PIC18F4XK20	---- x000	---- u000	---- uuuu
	PIC18F2XK20	PIC18F4XK20	---- x---	---- u---	---- u---
PORTD	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	PIC18F2XK20	PIC18F4XK20	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTB	PIC18F2XK20	PIC18F4XK20	xxx0 0000	uuu0 0000	uuuu uuuu
PORTA ⁽⁵⁾	PIC18F2XK20	PIC18F4XK20	xx0x 0000 ⁽⁵⁾	uu0u 0000 ⁽⁵⁾	uuuu uuuu ⁽⁵⁾
ANSELH ⁽⁶⁾	PIC18F2XK20	PIC18F4XK20	--1 1111	--1 1111	--u uuuu
ANSEL	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu
IOCB	PIC18F2XK20	PIC18F4XK20	0000 ----	0000 ----	uuuu ----
WPUB	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu
CM1CON0	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu
CM2CON0	PIC18F2XK20	PIC18F4XK20	0000 0000	0000 0000	uuuu uuuu

图注: u = 不变, x = 未知, - = 未实现位 (读为 0), q = 值取决于具体条件。

阴影单元表示不适用于指定器件的状态。

- 注 1: INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 (引起唤醒)。
 2: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, PC 装入中断向量 (0008h 或 0018h)。
 3: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, 用 PC 的当前值更新 TOSU、TOSH 和 TOSL。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。
 4: 具体条件下的复位值, 请参见表 4-3。
 5: 根据所选的振荡器模式使能 PORTA、LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚, 则它们将被禁止并读为 0。
 6: 如果 CONFIG3H 的 PBADEN 位为 0, 则 ANSELH 寄存器的所有位初始化为 0。

表 4-4: 所有寄存器的初始化状态 (续)

寄存器	适用器件		上电复位, 欠压复位	MCLR 复位, WDT 复位, RESET 指令, 堆栈复位	通过 WDT 或 中断唤醒器件
CM2CON1	PIC18F2XK20	PIC18F4XK20	0000 ----	0000 ----	uuuu ----
SLRCON	PIC18F2XK20	PIC18F4XK20	---1 1111	---1 1111	---u uuuu
SSPMSK	PIC18F2XK20	PIC18F4XK20	1111 1111	1111 1111	uuuu uuuu

图注: u = 不变, x = 未知, - = 未实现位 (读为 0), q = 值取决于具体条件。

阴影单元表示不适用于指定器件的状态。

- 注 1: INTCONx 或 PIRx 寄存器中的一位或多位会受到影响 (引起唤醒)。
- 2: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, PC 装入中断向量 (0008h 或 0018h)。
- 3: 当器件被中断唤醒且 GIEL 或 GIEH 位被置 1 时, 用 PC 的当前值更新 TOSU、TOSH 和 Tosl。将 STKPTR 修改为指向硬件堆栈的下一个存储单元。
- 4: 具体条件下的复位值, 请参见表 4-3。
- 5: 根据所选的振荡器模式使能 PORTA、LATA 和 TRISA 中的 bit 6 和 bit 7。如果未被使能为 PORTA 引脚, 则它们将被禁止并读为 0。
- 6: 如果 CONFIG3H 的 PBADEN 位为 0, 则 ANSELH 寄存器的所有位初始化为 0。

PIC18F2XK20/4XK20

注:

5.0 存储器构成

PIC18 增强型单片机器件有三种类型的存储器：

- 程序存储器
- 数据 RAM
- 数据 EEPROM

由于是哈佛架构的器件，数据和程序存储器使用不同的总线，因而可同时访问这两种存储空间。实际使用时，可将数据 EEPROM 当作外设，因为它可以通过一组控制寄存器进行寻址和访问。

第 6.0 节“闪存程序存储器” 提供了关于闪存程序存储器操作的更多详细信息。数据 EEPROM 将单独在**第 7.0 节“数据 EEPROM 存储器”**中讨论。

5.1 程序存储器构成

PIC18 单片机具有一个 21 位程序计数器，可以对 2 MB 的程序存储空间进行寻址。访问物理实现存储器的上边界和这个 2 MB 地址之间的存储单元会返回全 0 (NOP 指令)。

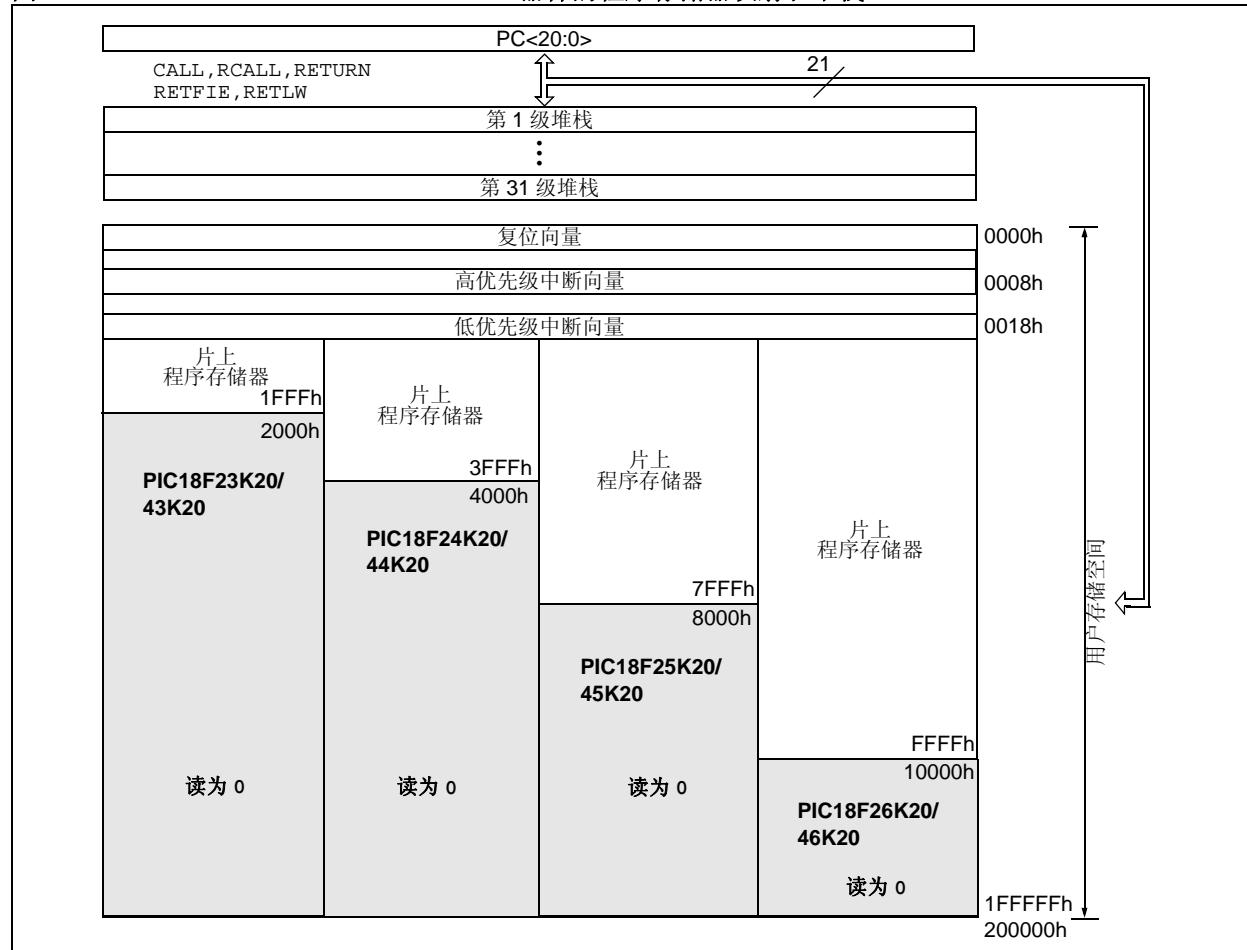
本系列器件包含的闪存如下：

- PIC18F23K20 和 PIC18F43K20: 8 KB 闪存，最多 4,096 条单字指令
- PIC18F24K20 和 PIC18F44K20: 16 KB 闪存，最多 8,192 条单字指令
- PIC18F25K20 和 PIC18F45K20: 32 KB 闪存，最多 16,384 条单字指令
- PIC18F26K20 和 PIC18F46K20: 64 KB 闪存，最多 37,768 条单字指令

PIC18 器件有两个中断向量。复位向量地址为 0000h，中断向量地址为 0008h 和 0018h。

图 5-1 给出了 PIC18F2XK20/4XK20 器件的程序存储器映射。存储器框图的详细信息如图 23-2 所示。

图 5-1: PIC18F2XK20/4XK20 器件的程序存储器映射和堆栈



5.1.1 程序计数器

程序计数器（Program Counter, PC）指定欲取出执行的指令的地址。PC 为 21 位宽，保存在三个不同的 8 位寄存器中。存储低字节的寄存器称为 PCL 寄存器，该寄存器可读写。存储高字节的寄存器，即 PCH 寄存器，存储 $PC<15:8>$ 位；该寄存器不可直接读写。更新 PCH 寄存器的操作是通过 PCLATH 寄存器实现的。存储最高字节的寄存器称为 PCU。该寄存器存储 $PC<20:16>$ 位；它也不能直接读写。更新 PCU 寄存器的操作是通过 PCLATU 寄存器实现的。

PCLATH 和 PCLATU 的内容通过执行写 PCL 的任何操作被传送到程序计数器。同样，程序计数器的两个高字节通过读 PCL 的操作被传送到 PCLATH 和 PCLATU。这对于 PC 的计算偏移量很有用处（见第 5.1.4.1 节“计算 GOTO”）。

PC 是按字节寻址程序存储器的。为了防止 PC 不能正确获取字指令，需要将 PCL 的最低有效位固定取值为 0。PC 每次加 2 来寻址程序存储器中的顺序指令。

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

5.1.2 返回地址堆栈

返回地址堆栈允许最多 31 个程序调用和中断的任意组合。当执行 CALL、RCALL 指令或响应中断时，PC 值会被压入该堆栈。而在执行 RETURN、RETLW 或 RETFIE 指令时，PC 值会从堆栈弹出。PCLATU 和 PCLATH 不受 RETURN 或 CALL 指令的影响。

通过 21 位的 RAM 和一个 5 位的堆栈指针 STKPTR 来实现 31 字的堆栈操作。堆栈既不占用程序存储空间，也不占用数据存储空间。堆栈指针是可读写的，并且通过栈顶（Top-of-Stack, TOS）特殊文件寄存器可以读写栈顶地址。也可以使用这些寄存器将数据压入堆栈或者从堆栈弹出。

执行 CALL 类型的指令时，产生压栈操作：首先堆栈指针加 1，并且将 PC（PC 已经指向 CALL 后下一条指令）的内容写入堆栈指针所指向的地址单元。执行 RETURN 类型的指令时，产生出栈操作：STKPTR 寄存器所指向的地址单元的内容会被传送给 PC，然后堆栈指针减 1。

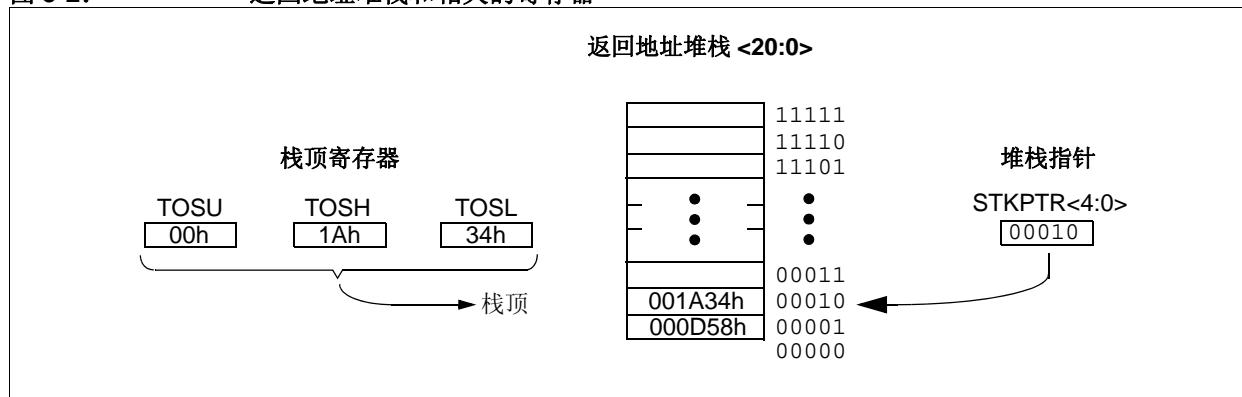
所有复位后，堆栈指针均会初始化为 00000。堆栈指针值 00000 不指向任何 RAM 单元；它仅仅是一个复位值。状态位表明堆栈是已满、上溢还是下溢。

5.1.2.1 访问栈顶

只可读写返回地址堆栈的栈顶（TOS）。有三个寄存器 TOSU:TOSH:TOSL 用于保存由 STKPTR 寄存器所指向的堆栈单元的内容（图 5-2）。这可以让用户在必要时实现软件堆栈。在 CALL、RCALL 或中断后，软件可以通过读取 TOSU:TOSH:TOSL 寄存器来读取压入堆栈的值。这些值可以被存放到由用户定义的软件堆栈。返回时，软件将这些值存回 TOSU:TOSH:TOSL 并执行返回。

为防止意外的堆栈操作，访问堆栈时用户必须禁止全局中断允许位。

图 5-2： 返回地址堆栈和相关的寄存器



5.1.2.2 返回堆栈指针 (STKPTR)

STKPTR 寄存器（寄存器 5-1）包含堆栈指针值、**STKFUL**（堆栈满）状态位和 **STKUNF**（堆栈下溢）状态位。堆栈指针值可为 0 到 31 范围内的值。向堆栈压入值前，堆栈指针加 1；而从堆栈弹出值后，堆栈指针减 1。复位时，堆栈指针值为零。用户可以读写堆栈指针的值。实时操作系统（Real-Time Operating System, RTOS）可以利用此特性对返回堆栈进行维护。

向堆栈压入 PC 值 31 次（且没有值从堆栈弹出）后，**STKFUL** 位置 1。通过软件或 POR 将 **STKFUL** 位清零。

由 **STVREN**（堆栈溢出复位使能）配置位的状态决定堆栈满时将执行的操作。（关于器件配置位的说明，请参见第 23.1 节“配置位”。）如果 **STVREN** 置 1（默认），第 31 次压栈将把 (PC+2) 值压入堆栈，从而将 **STKFUL** 位置 1 并复位器件。**STKFUL** 位将保持置 1，而堆栈指针将被清零。

如果 **STVREN** 清零，第 31 次压栈时 **STKFUL** 位将会置 1，堆栈指针递增到 31。任何其他压栈操作都不会覆盖第 31 次压栈的值，并且 **STKPTR** 将保持为 31。

当堆栈弹出次数足够卸空堆栈时，下一次出栈会向 PC 返回一个零值，并将 **STKUNF** 位置 1，而堆栈指针则保持为零。**STKUNF** 位将保持置 1，直到由软件清零或发生 POR 为止。

注： 下溢时，将零值返回给 PC，会使程序指向复位向量，此时可以验证堆栈状态并采取相应的操作。这与复位不同，因为下溢时 SFR 的内容不受影响。

5.1.2.3 PUSH 和 POP 指令

由于栈顶是可以读写的，因此将值压入堆栈或从堆栈弹出而不影响程序的正常执行是非常理想的。PIC18 指令集包含两条指令 **PUSH** 和 **POP**，使用这两条指令可在软件控制下对 **TOS** 执行操作。然后就可以修改 **TOSU**、**TOSH** 和 **TOSL**，将数据或返回地址压入堆栈。

PUSH 指令将当前的 PC 值压入堆栈。执行该指令会使堆栈指针加 1 并将当前的 PC 值装入堆栈。

POP 指令通过将堆栈指针减 1 来放弃当前的 **TOS** 值。然后前一个入栈的值就成为了 **TOS** 值。

寄存器 5-1： STKPTR：堆栈指针寄存器

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL⁽¹⁾	STKUNF⁽¹⁾	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

图注：

R = 可读位

W = 可写位

U = 未实现

C = 只可清零位

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **STKFUL:** 堆栈满标志位⁽¹⁾

1 = 堆栈满或溢出

0 = 堆栈未满也未溢出

bit 6 **STKUNF:** 堆栈下溢标志位⁽¹⁾

1 = 发生了堆栈下溢

0 = 未发生堆栈下溢

bit 5 **未实现:** 读为 0

bit 4-0 **SP<4:0>:** 堆栈指针地址位

注 1： 通过用户软件或 POR 清零 bit 7 和 bit 6。

5.1.2.4 堆栈满和下溢复位

通过将配置寄存器 4L 中的 STVREN 位置 1，来使能在堆栈溢出或下溢时的器件复位。当 STVREN 置 1 时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，然后使器件复位。当 STVREN 清零时，堆栈满或堆栈下溢状态会将相应的 STKFUL 或 STKUNF 位置 1，但不会使器件复位。通过用户软件或上电复位将 STKFUL 或 STKUNF 位清零。

5.1.3 快速寄存器堆栈

为 STATUS、WREG 和 BSR 寄存器提供的快速寄存器堆栈具有从中断“快速返回”的功能。每个堆栈只有一级且不可读写。当处理器转入中断向量处执行时，它装入对应寄存器的当前值。所有中断源都会将值压入堆栈寄存器。如果使用 RETFIE, FAST 指令从中断返回，这些寄存器中的值就会被装回相关的寄存器。

如果同时允许低优先级中断和高优先级中断，从低优先级中断返回时，无法可靠地使用堆栈寄存器。如果在为低优先级中断提供服务时，发生了高优先级中断，则低优先级中断存储在堆栈寄存器中的值将被覆盖。在为低优先级中断提供服务时，用户必须用软件保存关键寄存器的值。

如果未使用中断优先级，所有中断都可以使用快速寄存器堆栈从中断返回。如果没有使用中断，快速寄存器堆栈可以用于在子程序调用结束后恢复 STATUS、WREG 和 BSR 寄存器。要在子程序调用中使用快速寄存器堆栈，必须执行 CALL label, FAST 指令将 STATUS、WREG 和 BSR 寄存器的内容存入快速寄存器堆栈。然后执行 RETURN, FAST 指令，从快速寄存器堆栈恢复这些寄存器。

例 5-1 给出了一个在子程序调用和返回期间使用快速寄存器堆栈的源代码示例。

例 5-1： 快速寄存器堆栈代码示例

```
CALL SUB1, FAST      ; STATUS, WREG, BSR
                      ; SAVED IN FAST REGISTER
                      ; STACK
.
.
.
SUB1
.
.
.
RETURN, FAST        ; RESTORE VALUES SAVED
                      ; IN FAST REGISTER STACK
```

5.1.4 程序存储器中的查找表

有的编程场合可能需要在程序存储器中创建数据结构或查找表。对于 PIC18 器件，可以用两种方式实现查找表：

- 计算 GOTO
- 表读

5.1.4.1 计算 GOTO

计算 GOTO 是通过向程序计数器加一个偏移量来实现的。例 5-2 中给出了一个示例。

可以使用 ADDWF PCL 指令和一组 RETLW nn 指令创建一个查找表。在调用该表前，会先将查找表中的偏移量装入 W 寄存器。被调用子程序的第一条指令应该是 ADDWF PCL 指令。接下去执行的一条是 RETLW nn 指令，它将值 nn 返回给调用函数。

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

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

例 5-2： 使用偏移量值的计算 GOTO

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

5.1.4.2 表读与表写

有一种更好的方法可以将数据存储在程序存储器中，这种方法允许在每个指令单元存储 2 个字节的数据。

使用表读和表写，每个程序字可以存储 2 个字节的查找表数据。表指针（TBLPTR）寄存器指定字节地址，而表锁存寄存器（TABLAT）则存储从程序存储器中读取或写入的数据。一次只能向程序存储器或从程序存储器传送一个字节。

第 6.1 节“表读与表写”将进一步讨论表读和表写操作。

5.2 PIC18 指令周期

5.2.1 时钟机制

来自内部或外部时钟源的单片机时钟输入都将在内部被四分频以产生四个互不重叠的正交时钟信号（Q1、Q2、Q3 和 Q4）。程序计数器在每个 Q1 递增；在 Q4 期间，从程序存储器取指令并将指令锁存到指令寄存器中。指令的译码和执行在下一个 Q1 到 Q4 周期完成。图 5-3 所示为时钟和指令执行流程。

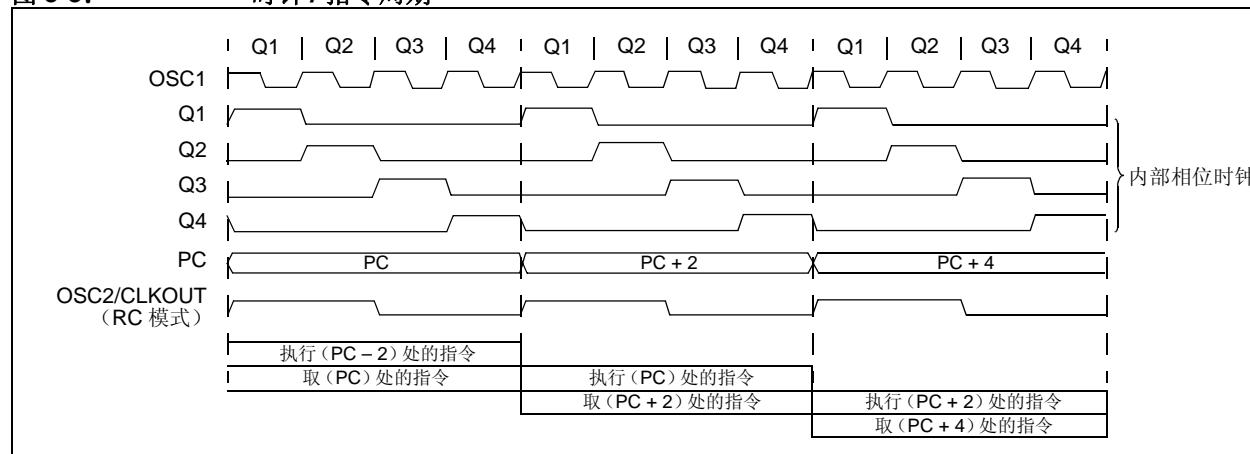
5.2.2 指令流 / 流水线

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

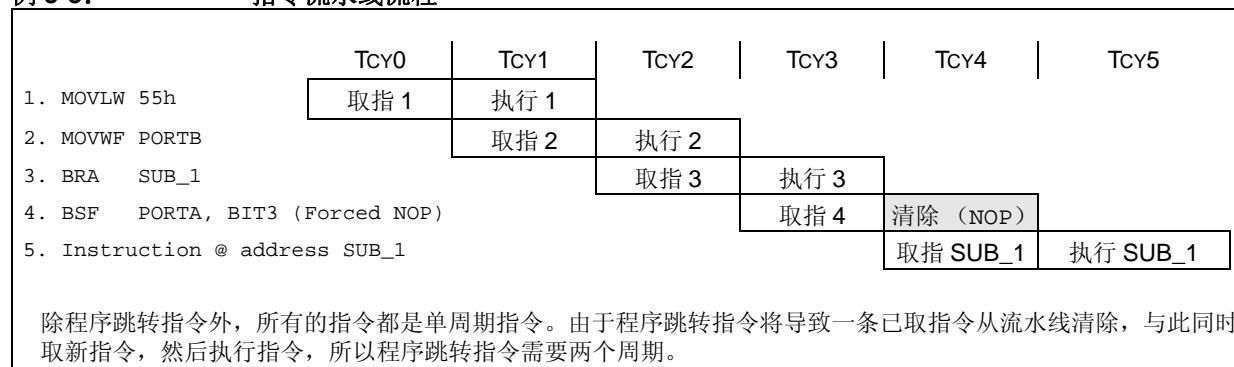
取指周期中：程序计数器（PC）在 Q1 周期递增，开始取指令。

指令执行周期中：在 Q1 周期，将所取指令锁存到指令寄存器（Instruction Register, IR）。在随后的 Q2、Q3 和 Q4 周期中译码并执行该指令。其中读数据存储器（读操作数）发生在 Q2 周期，写操作（写目标寄存器）发生在 Q4 周期。

图 5-3：时钟 / 指令周期



例 5-3：指令流水线流程



5.2.3 程序存储器中的指令

程序存储器按字节寻址。指令以 2 字节或 4 字节的形式存储在程序存储器中。指令字的最低有效字节始终存储在地址为偶数的程序存储单元中 ($LSb = 0$)。要保证与指令边界对齐, PC 必须以 2 为单位递增, 并且 LSb 总是读为 0 (见第 5.1.1 节 “程序计数器”)。

图 5-4 给出了指令字存储在程序存储器中的一个示例。

CALL 和 GOTO 指令在指令中嵌入了程序存储器的绝对地址。由于指令总是按字边界存储, 因而指令所包含的数据为一个字地址。字地址会写入 $PC<20:1>$, 用于访问程序存储器中的目标字节。图 5-4 中的指令 2 给出了指令 GOTO 0006h 在程序存储器中的译码过程。程序跳转指令也采取同样的方式对相对地址偏移量进行译码。存储在跳转指令中的偏移量代表单字指令数, PC 将以此作为偏移量跳转到指定的地址单元。第 24.0 节 “指令集汇总” 提供了指令集的更多详细信息。

图 5-4: 程序存储器中的指令

		$LSb = 1$	$LSb = 0$	字地址 ↓
	程序存储器 字节单元 →			000000h
				000002h
				000004h
				000006h
指令 1:	MOVlw	055h		000008h
指令 2:	GOTO	0006h		0000Ah
指令 3:	MOVff	123h, 456h	F0h	0000Ch
			C1h	0000Eh
			F4h	000010h
				000012h
				000014h

5.2.4 双字指令

标准的 PIC18 指令集有 4 条双字指令: CALL、MOVFF、GOTO 和 LSFR。这些指令第二个字的高 4 位均为 1111; 其他 12 位是立即数数据, 通常为一个数据存储器地址。

指令的高 4 位为 1111, 用于指定一条特殊形式的 NOP 指令。指令顺序执行的正确顺序为: 执行完第一个字之后立即按顺序访问并使用第二个字中的数据。如果由于

某些原因跳过了第一个字而自动执行指令的第二个字, 那么将作为一条 NOP 指令执行。如果双字指令跟在修改 PC 的条件指令后, 就有必要执行此操作。例 5-4 给出了它的执行过程。

注: 关于扩展指令集中的双字指令信息, 请参见第 5.6 节 “PIC18 指令的执行和扩展指令集”。

例 5-4: 双字指令

情形 1:	
目标代码	源代码
0110 0110 0000 0000	TSTFSZ REG1 ; is RAM location 0?
1100 0001 0010 0011	MOVFF REG1, REG2 ; No, skip this word
1111 0100 0101 0110	;
0010 0100 0000 0000	ADDWF REG3 ; continue code
情形 2:	
目标代码	源代码
0110 0110 0000 0000	TSTFSZ REG1 ; is RAM location 0?
1100 0001 0010 0011	MOVFF REG1, REG2 ; Yes, execute this word
1111 0100 0101 0110	;
0010 0100 0000 0000	ADDWF REG3 ; continue code

5.3 数据存储器构成

注: 当使能了 PIC18 扩展指令集时, 数据存储器某些方面的操作会有所改变。更多信息, 请参见第 5.5 节“数据存储器和扩展指令集”。

PIC18 器件中的数据存储器是用静态 RAM 实现的。在数据存储器中, 每个寄存器都有 12 位地址, 允许数据存储器实现为最大 4096 个字节。存储空间最多被分为 16 个存储区, 每个存储区包含 256 个字节。图 5-5 到 5-7 给出了 PIC18F2XK20/4XK20 器件的数据存储器构成。

数据存储器由特殊功能寄存器 (Special Function Register, SFR) 和通用寄存器 (General Purpose Register, GPR) 组成。SFR 用于单片机和外设功能模块的控制和状态指示, GPR 则用于用户应用程序的数据存储和中间结果暂存。任何未实现存储单元均读为 0。

这样的指令集和架构支持跨所有存储区的操作。可以通过直接、间接或变址寻址模式访问整个数据存储器。本章后面的部分将讨论寻址模式。

为确保能在周期中访问常用寄存器 (SFR 和某些 GPR), PIC18 器件实现了一个快速操作存储区。该存储区是一个 256 字节的存储空间, 它可实现对 SFR 和 GPR Bank 0 的低地址单元的快速访问, 而无需使用存储区选择寄存器 (Bank Select Register, BSR)。第 5.3.2 节“快速操作存储区”提供了对于快速操作 RAM 的详细说明。

5.3.1 存储区选择寄存器 (BSR)

容量较大的数据存储器需要高效的寻址机制, 以便对所有地址进行快速访问。理想状况下, 这意味着不必为每次读写操作提供完整地址。PIC18 器件是使用 RAM 存储区分区机制实现快速访问的。这种机制将存储空间分成连续的 16 个 256 字节的存储区。根据不同的指令, 可以通过完整的 12 位地址, 或通过 8 位的低字节地址和 4 位存储区指针直接寻址每个存储单元。

PIC18 指令集中的大部分指令都使用存储区指针, 也就是存储区选择寄存器 (BSR)。SFR 保存单元地址的高 4 位, 而指令本身则包括单元地址的低 8 位。只使用 BSR 的低 4 位 ($BSR<3:0>$), 不使用高 4 位; 高 4 位始终读为 0 且不能被写入。可以通过使用 MOVLB 指令直接装入 BSR。

BSR 的值代表数据存储器中的存储区; 指令中的 8 位指向存储区中的存储单元, 可以将它看作距离存储区下边界的偏移量。图 5-5 到 5-7 显示了 BSR 的值与数据存储器中的存储区之间的关系。

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

当选择存储区时, 只有已实现的存储区才可以读写。对未实现存储区进行的写操作将被忽略, 而读这些存储区会返回 0。虽然是这样, STATUS 寄存器仍然会受到影响, 好像操作是成功的。图 5-5 到 5-7 中的数据存储器映射指出了已实现的存储区。

在核心 PIC18 指令集中, 只有 MOVFF 指令指定源寄存器和目标寄存器的完整 12 位地址。该指令在执行时完全忽略 BSR。所有其他指令仅包含作为操作数的低位地址, 而且必须使用 BSR 或快速操作存储区来寻址目标寄存器。

PIC18F2XK20/4XK20

图 5-5: PIC18F23K20/43K20 器件的数据存储器映射

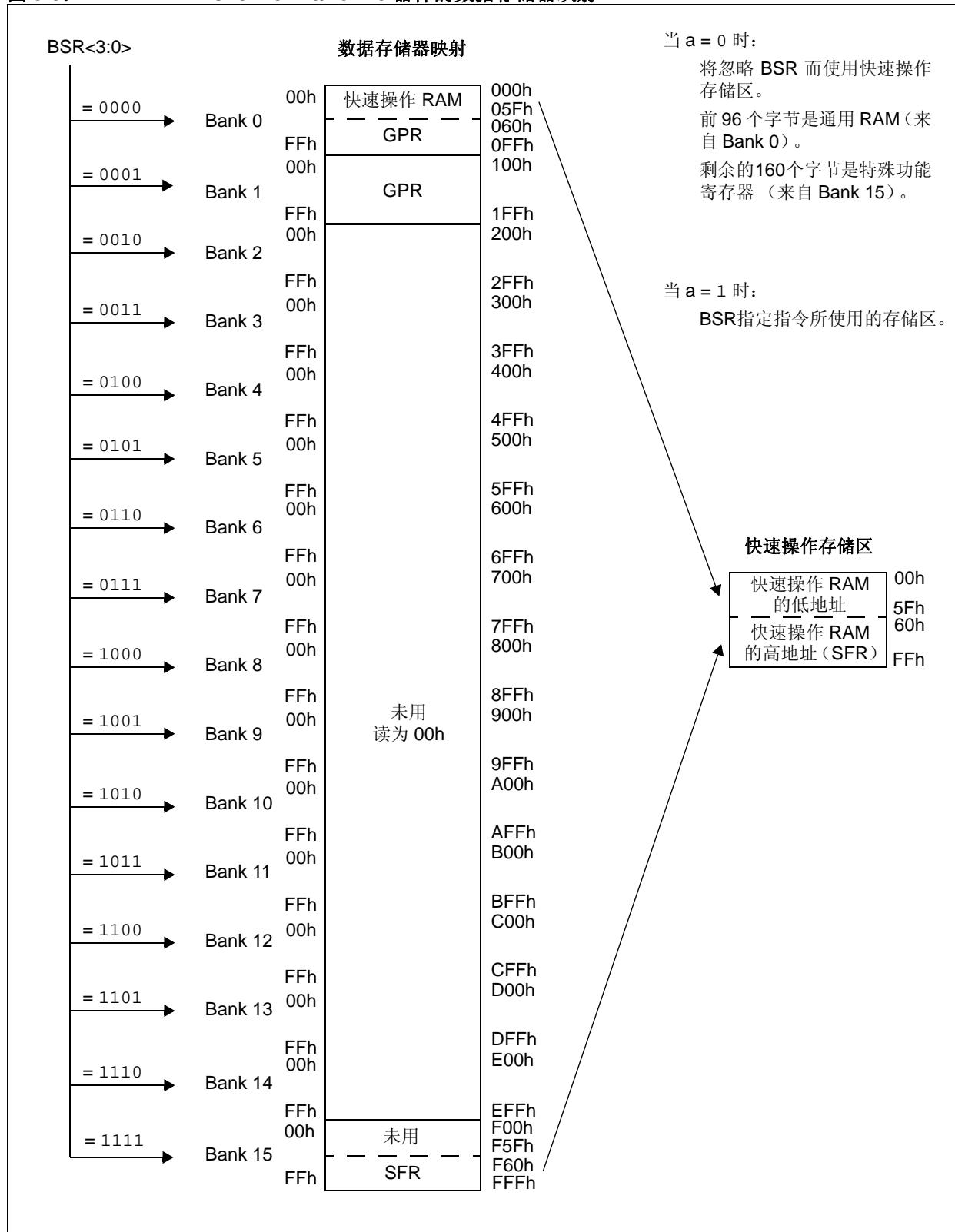
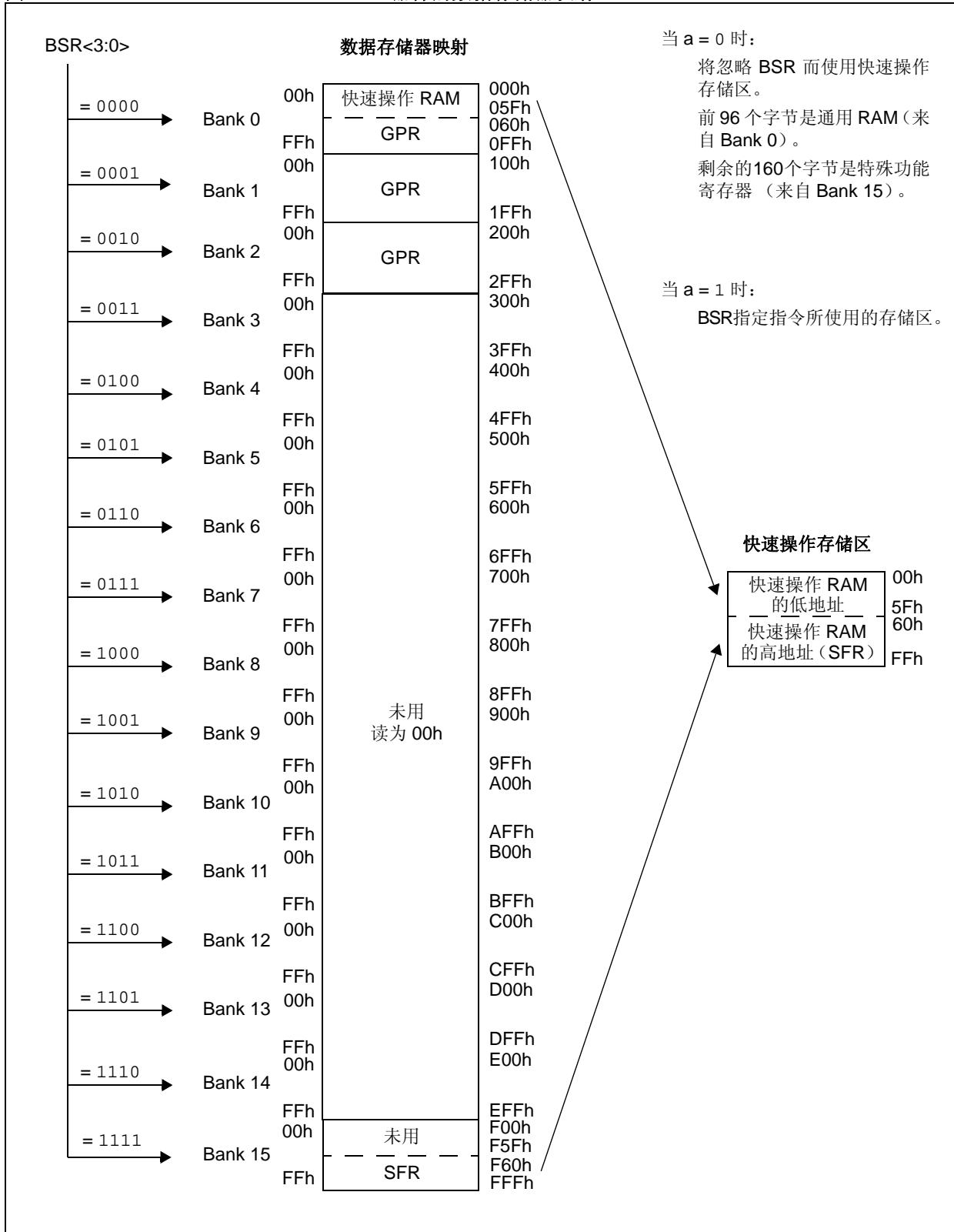


图 5-6: PIC18F24K20/44K20 器件的数据存储器映射



PIC18F2XK20/4XK20

图 5-7: PIC18F25K20/45K20 器件的数据存储器映射

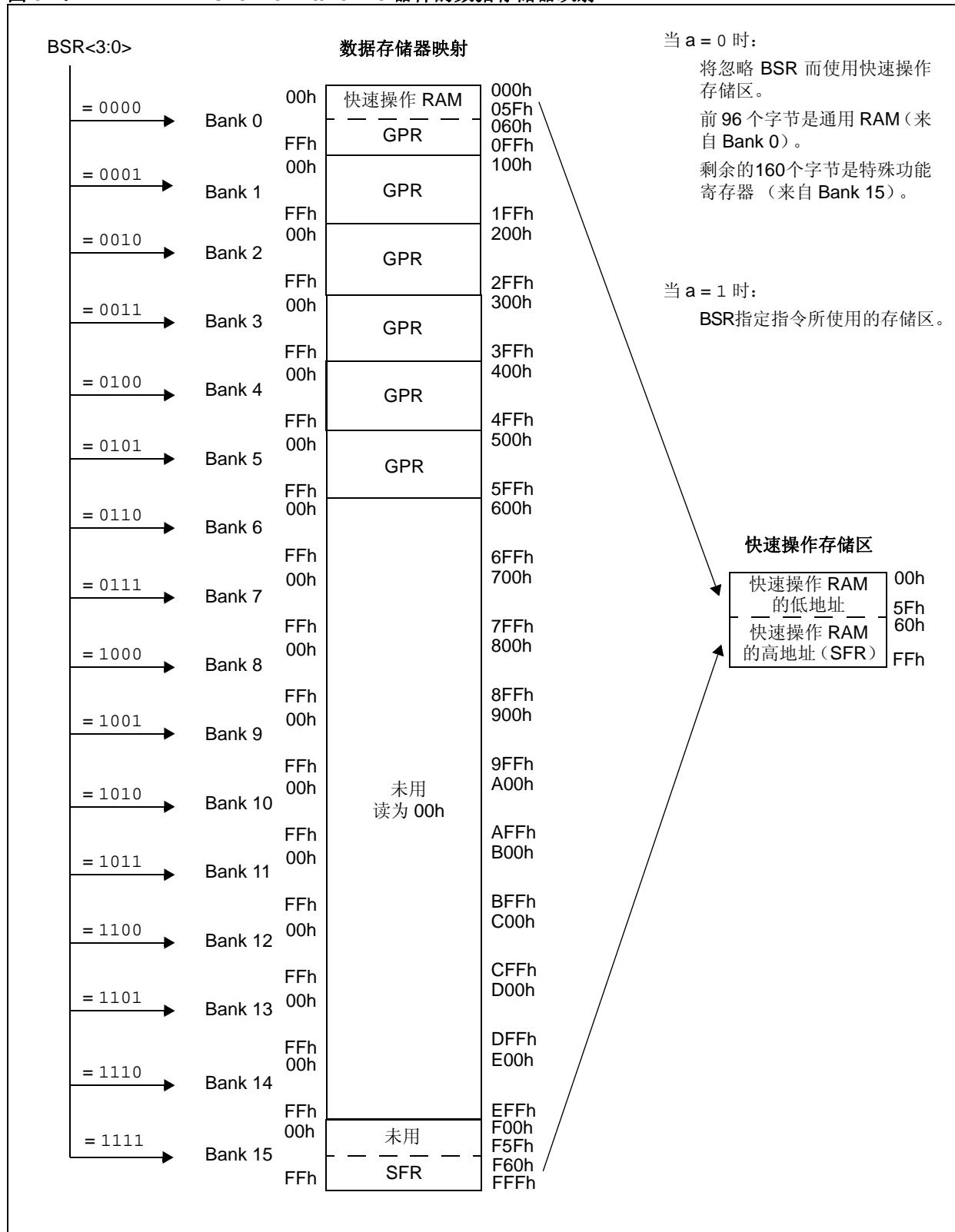
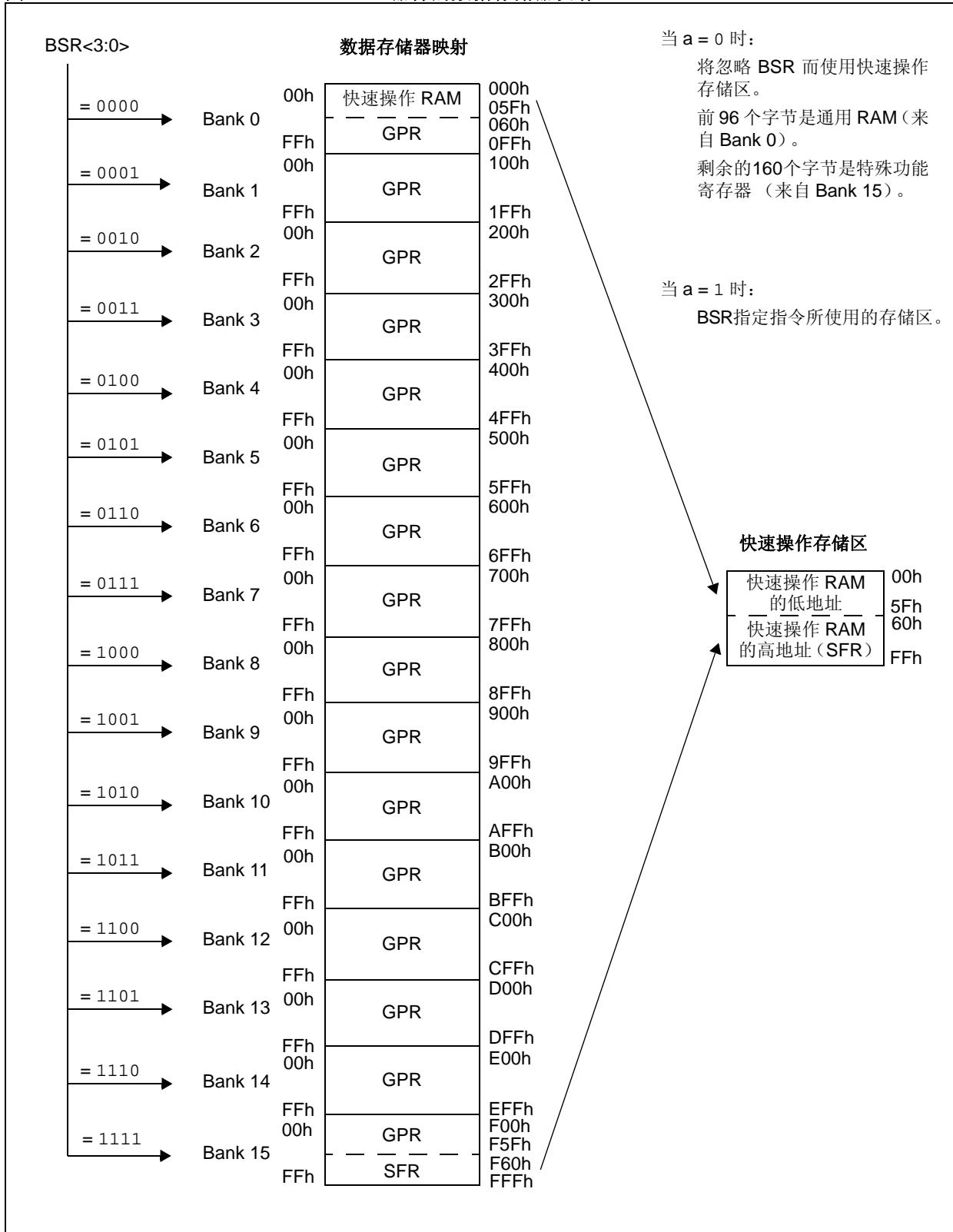
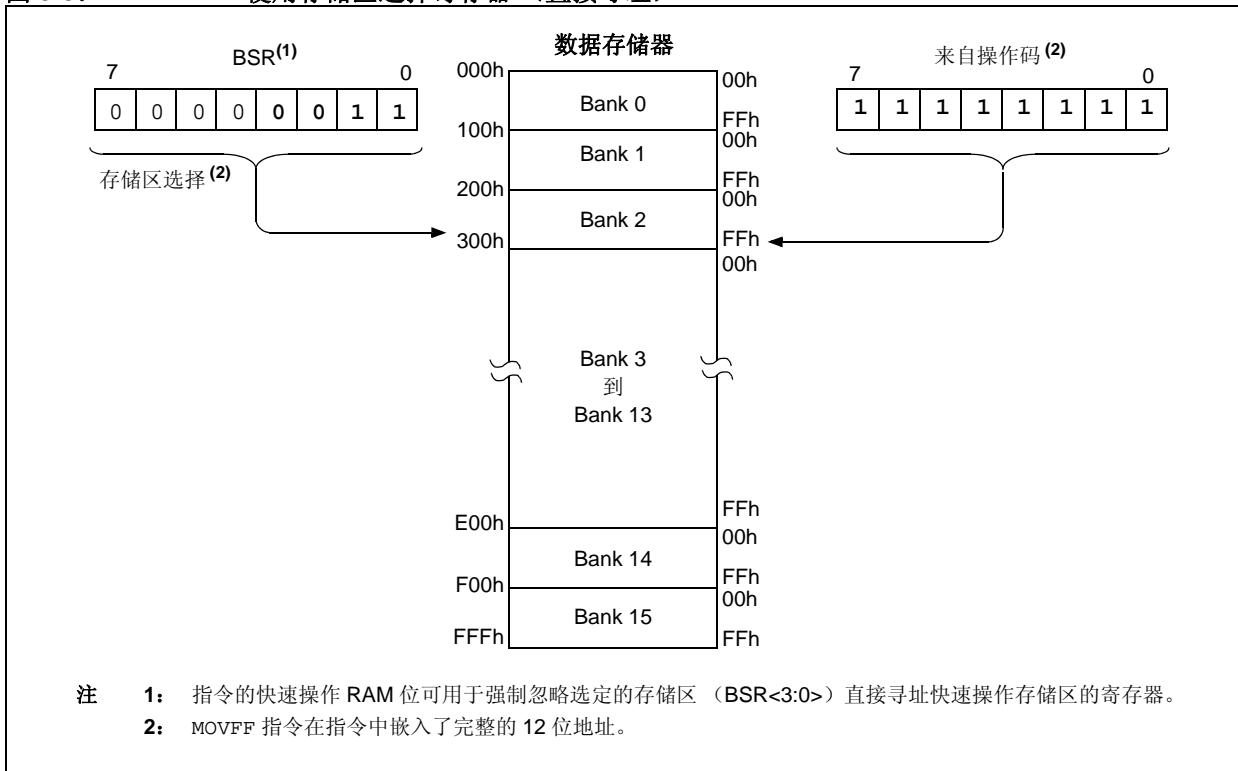


图 5-8: PIC18F26K20/46K20 器件的数据存储器映射



PIC18F2XK20/4XK20

图 5-9: 使用存储区选择寄存器（直接寻址）



5.3.2 快速操作存储区

使用 **BSR** 和嵌入的 8 位地址，用户可以寻址数据存储器的整个空间，但这同时也意味着用户必须始终确保选择了正确的存储区。否则，可能会从错误的单元读取数据或将数据写入错误的单元。如果本来是向 **GPR** 进行写操作，却将结果写入了 **SFR**，后果是非常严重的。但是在每次对数据存储器进行读或写操作时验证和 / 或更改 **BSR** 会严重影响工作效率。

为了提高访问大多数常用数据存储单元的效率，现为数据存储器配置了快速操作存储区，这样可以允许用户访问被映射的存储区而无需指定 **BSR**。快速操作存储区由 **Bank 0** 的前 96 个字节（00h-5Fh）和 **Bank 15** 的后 160 个字节（60h-FFh）组成。地址较低的部分被称为“快速操作 RAM”，由 **GPR** 组成。地址较高的部分则被映射为器件的 **SFR**。这两个区域被连续地映射到快速操作存储区并且可以用一个 8 位地址进行线性寻址（图 5-5 到 5-7）。

包括快速操作 RAM 位（指令中的“*a*”参数）的核心 **PIC18** 指令使用快速操作存储区。当“*a*”等于 1 时，指令使用 **BSR** 和包含在操作码中的 8 位地址对数据存储器寻址。当“*a*”为 0 时，强制指令使用快速操作存储区地址映射，此时完全忽略 **BSR** 的当前值。

此“强制”寻址模式可使指令在一个周期内对数据地址进行操作，而不需要首先更新 **BSR**。这意味着用户可以更高效地对 8 位地址为 60h 或以上的 **SFR** 进行取值和操作。地址为 60h 以下的快速操作 RAM 非常适合于存储那些用户可能需要快速访问的数据值，如直接计算结果或常用程序变量。快速操作 RAM 也可实现更加快速和高效的现场保护和变量切换代码。

使能扩展指令集（**XINST** 配置位 = 1）时的快速操作存储区的映射略有不同。在第 5.5.3 节“在立即数变址模式下映射快速操作存储区”中对此进行了更详细的讨论。

5.3.3 通用寄存器文件

PIC18 器件在 **GPR** 区中划分了一部分存储区。这部分存储区为数据 **RAM**，所有指令均可访问它。**GPR** 区从 **Bank 0** 的底部（地址 000h）开始向上延伸直到 **SFR** 区的底部。上电复位不会初始化 **GPR**，并且其他复位也不会改变其内容。

5.3.4 特殊功能寄存器

特殊功能寄存器（**SFR**）是 **CPU** 和外设模块用来控制所需器件操作的寄存器。这些寄存器以静态 **RAM** 的形式实现。**SFR** 从数据存储器的顶部（FFFh）开始向下，它占据了 **Bank 15** 的高地址部分空间（F60h 到 FFFh）。表 5-1 和表 5-2 列出了这些寄存器。

可以将 **SFR** 归类为两组：与“内核”器件功能（**ALU**、复位和中断）相关的寄存器和与外设功能相关的寄存器。复位和中断寄存器在相应章节中进行讨论，本章后面的部分将对 **ALU** 的 **STATUS** 寄存器进行说明。与外设操作相关的寄存器将在该外设的章节中进行说明。

SFR 通常分布在功能受其控制的外设中。未使用的 **SFR** 单元是未实现的，读为 0。

PIC18F2XK20/4XK20

表 5-1： PIC18F2XK20/4XK20 器件的特殊功能寄存器映射

地址	名称	地址	名称	地址	名称	地址	名称
FFFh	TOSU	FD7h	TMR0H	FAFh	SPBRG	F87h	__(2)
FFEh	TOSH	FD6h	TMR0L	FAEh	RCREG	F86h	__(2)
FFDh	TOSL	FD5h	T0CON	FADh	TXREG	F85h	__(2)
FFCh	STKPTR	FD4h	__(2)	FACh	TXSTA	F84h	PORTE
FFBh	PCLATU	FD3h	OSCCON	FABh	RCSTA	F83h	PORTD ⁽³⁾
FFAh	PCLATH	FD2h	HLVDCON	FAAh	EEADR ⁽⁴⁾	F82h	PORTC
FF9h	PCL	FD1h	WDTCON	FA9h	EEADR	F81h	PORTB
FF8h	TBLPTRU	FD0h	RCON	FA8h	EEDATA	F80h	PORTA
FF7h	TBLPTRH	FCFh	TMR1H	FA7h	EECON2 ⁽¹⁾	F7Fh	ANSELH
FF6h	TBLPTRL	FCEh	TMR1L	FA6h	EECON1	F7Eh	ANSEL
FF5h	TABLAT	FCDh	T1CON	FA5h	__(2)	F7Dh	IOCB
FF4h	PRODH	FCCh	TMR2	FA4h	__(2)	F7Ch	WPUB
FF3h	PRODL	FCBh	PR2	FA3h	__(2)	F7Bh	CM1CON0
FF2h	INTCON	FCAh	T2CON	FA2h	IPR2	F7Ah	CM2CON0
FF1h	INTCON2	FC9h	SSPBUF	FA1h	PIR2	F79h	CM2CON1
FF0h	INTCON3	FC8h	SSPADD	FA0h	PIE2	F78h	SLRCON
FEFh	INDF0 ⁽¹⁾	FC7h	SSPSTAT	F9Fh	IPR1	F77h	SSPMSK
FEEh	POSTINC0 ⁽¹⁾	FC6h	SSPCON1	F9Eh	PIR1	F76h	__(2)
FEDh	POSTDEC0 ⁽¹⁾	FC5h	SSPCON2	F9Dh	PIE1	F75h	__(2)
FECh	PREINC0 ⁽¹⁾	FC4h	ADRESH	F9Ch	__(2)	F74h	__(2)
FEBh	PLUSW0 ⁽¹⁾	FC3h	ADRESL	F9Bh	OSCTUNE	F73h	__(2)
FEAh	FSR0H	FC2h	ADCON0	F9Ah	__(2)	F72h	__(2)
FE9h	FSR0L	FC1h	ADCON1	F99h	__(2)	F71h	__(2)
FE8h	WREG	FC0h	ADCON2	F98h	__(2)	F70h	__(2)
FE7h	INDF1 ⁽¹⁾	FBFh	CCPR1H	F97h	__(2)	F6Fh	__(2)
FE6h	POSTINC1 ⁽¹⁾	FBEh	CCPR1L	F96h	TRISE ⁽³⁾	F6Eh	__(2)
FE5h	POSTDEC1 ⁽¹⁾	FBDh	CCP1CON	F95h	TRISD ⁽³⁾	F6Dh	__(2)
FE4h	PREINC1 ⁽¹⁾	FBCh	CCPR2H	F94h	TRISC	F6Ch	__(2)
FE3h	PLUSW1 ⁽¹⁾	FBBh	CCPR2L	F93h	TRISB	F6Bh	__(2)
FE2h	FSR1H	FBAh	CCP2CON	F92h	TRISA	F6Ah	__(2)
FE1h	FSR1L	FB9h	PSTRCON	F91h	__(2)	F69h	__(2)
FE0h	BSR	FB8h	BAUDCON	F90h	__(2)	F68h	__(2)
FDFh	INDF2 ⁽¹⁾	FB7h	PWM1CON	F8Fh	__(2)	F67h	__(2)
FDEh	POSTINC2 ⁽¹⁾	FB6h	ECCP1AS	F8Eh	__(2)	F66h	__(2)
FDDh	POSTDEC2 ⁽¹⁾	FB5h	CVRC CON	F8Dh	LATE ⁽³⁾	F65h	__(2)
FDCh	PREINC2 ⁽¹⁾	FB4h	CVRC CON2	F8Ch	LATD ⁽³⁾	F64h	__(2)
FDBh	PLUSW2 ⁽¹⁾	FB3h	TMR3H	F8Bh	LATC	F63h	__(2)
FDAh	FSR2H	FB2h	TMR3L	F8Ah	LATB	F62h	__(2)
FD9h	FSR2L	FB1h	T3CON	F89h	LATA	F61h	__(2)
FD8h	STATUS	FB0h	SPBRGH	F88h	__(2)	F60h	__(2)

注 1: 这不是实际存在的寄存器。

2: 未实现的寄存器，读为 0。

3: 该寄存器在 PIC18F2XK20 器件上不可用。

4: 该寄存器仅在 PIC18F46K20 和 PIC18F26K20 器件上实现。

表 5-2: 寄存器文件汇总 (PIC18F2XK20/4XK20)

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见 (页):
TOSU	—	—	—	栈顶最高字节 (TOS<20:16>)					---0 0000	59, 66
TOSH	栈顶高字节 (TOS<15:8>)								0000 0000	59, 66
TOSL	栈顶低字节 (TOS<7:0>)								0000 0000	59, 66
STKPTR	STKFUL	STKUNF	—	SP4	SP3	SP2	SP1	SP0	00-0 0000	59, 67
PCLATU	—	—	—	PC<20:16> 的保持寄存器					---0 0000	59, 66
PCLATH	PC<15:8> 的保持寄存器								0000 0000	59, 66
PCL	PC 低字节 (PC<7:0>)								0000 0000	59, 66
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 (TBLPTR<20:16>)					--00 0000	59, 92
TBLPTRH	程序存储器表指针高字节 (TBLPTR<15:8>)								0000 0000	59, 92
TBLPTRL	程序存储器表指针低字节 (TBLPTR<7:0>)								0000 0000	59, 92
TABLAT	程序存储器表锁存器								0000 0000	59, 92
PRODH	乘积寄存器的高字节								xxxx xxxx	59, 105
PRODL	乘积寄存器的低字节								xxxx xxxx	59, 105
INTCON	GIE/GIEH	PEIE/GIEL	TMROIE	INT0IE	RBIE	TMROIF	INT0IF	RBIF	0000 000x	59, 109
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	—	TMROIP	—	RBIP	1111 -1-1	59, 110
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	11-0 0-00	59, 111
INDF0	使用 FSR0 的内容寻址数据存储器——FSR0 的值不变 (不是实际存在的寄存器)								N/A	59, 84
POSTINC0	使用 FSR0 的内容寻址数据存储器——FSR0 的值后递增 (不是实际存在的寄存器)								N/A	59, 84
POSTDEC0	使用 FSR0 的内容寻址数据存储器——FSR0 的值后递减 (不是实际存在的寄存器)								N/A	59, 84
PREINC0	使用 FSR0 的内容寻址数据存储器——FSR0 的值预递增 (不是实际存在的寄存器)								N/A	59, 84
PLUSW0	使用 FSR0 的内容寻址数据存储器——FSR0 偏移量的值由 W 寄存器提供 (不是实际存在的寄存器)								N/A	59, 84
FSR0H	—	—	—	—	间接数据存储器地址指针 0 的高字节				---- 0000	59, 84
FSR0L	间接数据存储器地址指针 0 的低字节								xxxx xxxx	59, 84
WREG	工作寄存器								xxxx xxxx	59
INDF1	使用 FSR1 的内容寻址数据存储器——FSR1 的值不变 (不是实际存在的寄存器)								N/A	59, 84
POSTINC1	使用 FSR1 的内容寻址数据存储器——FSR1 的值后递增 (不是实际存在的寄存器)								N/A	59, 84
POSTDEC1	使用 FSR1 的内容寻址数据存储器——FSR1 的值后递减 (不是实际存在的寄存器)								N/A	59, 84
PREINC1	使用 FSR1 的内容寻址数据存储器——FSR1 的值预递增 (不是实际存在的寄存器)								N/A	59, 84
PLUSW1	使用 FSR1 的内容寻址数据存储器——FSR1 偏移量的值由 W 寄存器提供 (不是实际存在的寄存器)								N/A	59, 84
FSR1H	—	—	—	—	间接数据存储器地址指针 1 的高字节				---- 0000	60, 84
FSR1L	间接数据存储器地址指针 1 的低字节								xxxx xxxx	60, 84
BSR	—	—	—	—	存储区选择寄存器				---- 0000	60, 71
INDF2	使用 FSR2 的内容寻址数据存储器——FSR2 的值不变 (不是实际存在的寄存器)								N/A	60, 84
POSTINC2	使用 FSR2 的内容寻址数据存储器——FSR2 的值后递增 (不是实际存在的寄存器)								N/A	60, 84
POSTDEC2	使用 FSR2 的内容寻址数据存储器——FSR2 的值后递减 (不是实际存在的寄存器)								N/A	60, 84
PREINC2	使用 FSR2 的内容寻址数据存储器——FSR2 的值预递增 (不是实际存在的寄存器)								N/A	60, 84
PLUSW2	使用 FSR2 的内容寻址数据存储器——FSR2 偏移量的值由 W 寄存器提供 (不是实际存在的寄存器)								N/A	60, 84
FSR2H	—	—	—	—	间接数据存储器地址指针 2 的高字节				---- 0000	60, 84
FSR2L	间接数据存储器地址指针 2 的低字节								xxxx xxxx	60, 84
STATUS	—	—	—	N	OV	Z	DC	C	---x xxxx	60, 82

图注: x = 未知, u = 不变, — = 未实现, q = 值取决于具体条件

注 1: SBOREN 位仅在 BOREN<1:0> 配置位 = 01 时可用; 否则, 它被禁止且读为 0。请参见第 4.4 节 “欠压复位 (BOR)”。

2: 这些寄存器和 / 或位在 28 引脚器件上未实现, 读为 0。给出了 40/44 引脚器件的复位值; 各未实现位表示为 —。

3: PLLEN 位仅在特定的振荡器配置中可用; 否则, 它被禁止且读为 0。请参见第 2.6.2 节 “HFINTOSC 模式下的 PLL”。

4: RE3 位仅在主复位被禁止 (MCLRE 配置位 = 0) 时可用。否则, RE3 读为 0。该位是只读的。

5: 根据不同的主振荡器模式, 可将 RA6/RA7 及其相关的锁存及方向位分别配置为端口引脚。这些位在禁止时读为 0。

6: 如果 CONFIG3H 的 PBADEN 位为 0, 则 ANSELH 寄存器的所有位初始化为 0。

7: 该寄存器仅在 PIC18F46K20 和 PIC18F26K20 器件上实现。

PIC18F2XK20/4XK20

表 5-2: 寄存器文件汇总 (PIC18F2XK20/4XK20) (续)

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见 (页):
TMR0H	Timer0 寄存器的高字节								0000 0000	60, 157
TMR0L	Timer0 寄存器的低字节								xxxx xxxx	60, 157
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	60, 155
OSCCON	IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0	0011 qq00	29, 60
HLVDCON	VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	0-00 0101	60, 291
WDTCON	—	—	—	—	—	—	—	SWDTEN	--- ---0	60, 307
RCON	IPEN	SBOREN ⁽¹⁾	—	RI	TO	PD	POR	BOR	0q-1 11q0	51, 58, 118
TMR1H	Timer1 寄存器的高字节								xxxx xxxx	60, 165
TMR1L	Timer1 寄存器的低字节								xxxx xxxx	60, 165
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	—	TMR1CS	TMR1ON	0000 0000	60, 159
TMR2	Timer2 寄存器								0000 0000	60, 168
PR2	Timer2 周期寄存器								1111 1111	60, 168
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	60, 167
SSPBUF	SSP 接收缓冲 / 发送寄存器								xxxx xxxx	60, 201, 202
SSPADD	I ² C TM 从模式下的 SSP 地址寄存器。 I ² C 主模式下的 SSP 波特率重载寄存器。								0000 0000	60, 202
SSPSTAT	SMP	CKE	D/A	P	S	R/W	UA	BF	0000 0000	60, 194, 204
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	60, 195, 205
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	60, 206
ADRESH	A/D 结果寄存器的高字节								xxxx xxxx	61, 275
ADRESL	A/D 结果寄存器的低字节								xxxx xxxx	61, 275
ADCON0	—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON	--00 0000	61, 269
ADCON1	—	—	VCFG1	VCFG0	—	—	—	—	--00 ----	59, 270
ADCON2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	0-00 0000	61, 271
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								xxxx xxxx	61, 144
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								xxxx xxxx	61, 144
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	0000 0000	61, 173
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								xxxx xxxx	61, 144
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								xxxx xxxx	61, 144
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	--00 0000	61, 143
PSTRCON	—	—	—	STRSYNC	STRD	STRC	STRB	STRA	--0 0001	61, 187
BAUDCON	ABDOVF	RCIDL	DTRXP	CKTXP	BRG16	—	WUE	ABDEN	0100 0-00	61, 246
PWM1CON	PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0	0000 0000	61, 186
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0	0000 0000	61, 183
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	0000 0000	61, 289
CVRCON2	FVREN	FVRST	—	—	—	—	—	—	00-- ----	61, 290
TMR3H	Timer3 寄存器的高字节								xxxx xxxx	61, 172
TMR3L	Timer3 寄存器的低字节								xxxx xxxx	61, 172
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	—	TMR3CS	TMR3ON	0000 0000	61, 169

图注: x = 未知, u = 不变, — = 未实现, q = 值取决于具体条件

注 1: SBOREN 位仅在 BOREN<1:0> 配置位 = 01 时可用; 否则, 它被禁止且读为 0。请参见第 4.4 节 “欠压复位 (BOR)”。

2: 这些寄存器和 / 或位在 28 引脚器件上未实现, 读为 0。给出了 40/44 引脚器件的复位值; 各未实现位表示为 —。

3: PLLEN 位仅在特定的振荡器配置中可用; 否则, 它被禁止且读为 0。请参见第 2.6.2 节 “HFINTOSC 模式下的 PLL”。

4: RE3 位仅在主复位被禁止 (MCLRE 配置位 = 0) 时可用。否则, RE3 读为 0。该位是只读的。

5: 根据不同的主振荡器模式, 可将 RA6/RA7 及其相关的锁存及方向位分别配置为端口引脚。这些位在禁止时读为 0。

6: 如果 CONFIG3H 的 PBADEN 位为 0, 则 ANSELH 寄存器的所有位初始化为 0。

7: 该寄存器仅在 PIC18F46K20 和 PIC18F26K20 器件上实现。

表 5-2: 寄存器文件汇总 (PIC18F2XK20/4XK20) (续)

寄存器名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	POR/BOR 时的值	详情请见 (页):
SPBRGH	EUSART 波特率发生器寄存器的高字节								0000 0000	61, 239
SPBRG	EUSART 波特率发生器寄存器的低字节								0000 0000	61, 239
RCREG	EUSART 接收寄存器								0000 0000	61, 236
TXREG	EUSART 发送寄存器								0000 0000	61, 235
TXSTA	CSRC	TX9	TXEN	SYNC	SEND _B	BRGH	TRMT	TX9D	0000 0010	61, 244
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	61, 245
EEADR	EEADR7	EEADR6	EEADR5	EEADR4	EEADR3	EEADR2	EEADR1	EEADRO	0000 0000	61, 90, 99
EEADRH ⁽⁷⁾	—	—	—	—	—	—	EEADR9	EEADR8	---- --00	61, 90, 99
EEDATA	EEPROM 数据寄存器								0000 0000	61, 90, 99
EECON2	EEPROM 控制寄存器 2 (不是实际存在的寄存器)								0000 0000	61, 90, 99
EECON1	EEPGD	CFG _S	—	FREE	WRERR	WREN	WR	RD	xx-0 x000	61, 91, 99
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	1111 1111	62, 117
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	0000 0000	62, 113
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	0000 0000	62, 115
IPR1	PSPIP ⁽²⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	1111 1111	62, 116
PIR1	PSPIF ⁽²⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	62, 112
PIE1	PSPIE ⁽²⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	62, 114
OSCTUNE	INTSRC	PLLEN ⁽³⁾	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0	0q00 0000	33, 62
TRISE ⁽²⁾	IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0	0000 -111	62, 134
TRISD ⁽²⁾	PORTD 数据方向控制寄存器								1111 1111	62, 130
TRISC	PORTC 数据方向控制寄存器								1111 1111	62, 127
TRISB	PORTB 数据方向控制寄存器								1111 1111	62, 124
TRISA	TRISA7 ⁽⁵⁾	TRISA6 ⁽⁵⁾	PORTA 数据方向控制寄存器						1111 1111	62, 121
LATE ⁽²⁾	—	—	—	—	—	PORTE 数据锁存寄存器 (读和写数据锁存器)			---- -xxxx	62, 133
LATD ⁽²⁾	PORTD 数据锁存寄存器 (读和写数据锁存器)								xxxxx xxxx	62, 130
LATC	PORTC 数据锁存寄存器 (读和写数据锁存器)								xxxxx xxxx	62, 127
LATB	PORTB 数据锁存寄存器 (读和写数据锁存器)								xxxxx xxxx	62, 124
LATA	LATA7 ⁽⁵⁾	LATA6 ⁽⁵⁾	PORTA 数据锁存寄存器 (读和写数据锁存器)						xxxxx xxxx	62, 121
PORTE	—	—	—	—	RE3 ⁽⁴⁾	RE2 ⁽²⁾	RE1 ⁽²⁾	RE0 ⁽²⁾	---- x000	62, 133
PORTD ⁽²⁾	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	xxxxx xxxx	62, 130
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxxx xxxx	62, 127
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxx0 0000	62, 124
PORTA	RA7 ⁽⁵⁾	RA6 ⁽⁵⁾	RA5	RA4	RA3	RA2	RA1	RA0	xx0x 0000	62, 121
ANSELH ⁽⁶⁾	—	—	—	ANS12	ANS11	ANS10	ANS9	ANS8	---1 1111	62, 137
ANSEL	ANS7 ⁽²⁾	ANS6 ⁽²⁾	ANS5 ⁽²⁾	ANS4	ANS3	ANS2	ANS1	ANS0	1111 1111	62, 136
IOCB	IOCB7	IOCB6	IOCB5	IOCB4	—	—	—	—	0000 ----	62, 124
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	1111 1111	62, 124
CM1CON0	C1ON	C1OUT	C1OE	C1POL	C1SP	C1R	C1CH1	C1CH0	0000 0000	62, 282
CM2CON0	C2ON	C2OUT	C2OE	C2POL	C2SP	C2R	C2CH1	C2CH0	0000 0000	62, 283
CM2CON1	MC1OUT	MC2OUT	C1RSEL	C2RSEL	—	—	—	—	0000 ----	63, 285
SLRCON	—	—	—	SLRE ⁽²⁾	SLRD ⁽²⁾	SLRC	SLRB	SLRA	---1 1111	63, 138
SSPM _S K	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	1111 1111	63, 213

图注: x = 未知, u = 不变, — = 未实现, q = 值取决于具体条件

注 1: SBOREN 位仅在 BOREN<1:0> 配置位 = 01 时可用; 否则, 它被禁止且读为 0。请参见第 4.4 节 “欠压复位 (BOR)”。

2: 这些寄存器和 / 或位在 28 引脚器件上未实现, 读为 0。给出了 40/44 引脚器件的复位值; 各未实现位表示为 —。

3: PLLN 位仅在特定的振荡器配置中可用; 否则, 它被禁止且读为 0。请参见第 2.6.2 节 “HFINTOSC 模式下的 PLL”。

4: RE3 位仅在主复位被禁止 (MCLRE 配置位 = 0) 时可用。否则, RE3 读为 0。该位是只读的。

5: 根据不同的主振荡器模式, 可将 RA6/RA7 及其相关的锁存及方向位分别配置为端口引脚。这些位在禁止时读为 0。

6: 如果 CONFIG3H 的 PBADEM 位为 0, 则 ANSELH 寄存器的所有位初始化为 0。

7: 该寄存器仅在 PIC18F46K20 和 PIC18F26K20 器件上实现。

5.3.5 STATUS 寄存器

如寄存器 5-2 所示, STATUS 寄存器包含 ALU 的算术运算状态。和任何其他 SFR 一样, 它也可以作为任何指令的操作数。

如果一条影响 Z、DC、C、OV 或 N 位的指令以 STATUS 寄存器作为目标寄存器, 将不会直接写入结果, 而是根据指令的执行更新 STATUS 寄存器。因此, 当执行一条把 STATUS 寄存器作为目标寄存器的指令后, 运行结果可能与预想的不同。例如, CLRF STATUS 将 Z 位置 1 并保持其余状态位不变 (000u uluu)。

因此, 建议仅使用 BCF、BSF、SWAPF、MOVFF 和 MOVWF 指令来改变 STATUS 寄存器, 因为这些指令不会影响 STATUS 寄存器中的 Z、C、DC、OV 或 N 位。

关于其他不会影响状态位的指令, 请参见表 24-2 和表 24-3 中的指令集汇总。

注: 在减法运算中, C 和 DC 位分别作为借位位和半借位位。

寄存器 5-2: STATUS: 状态寄存器

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC ⁽¹⁾	C ⁽¹⁾
bit 7	bit 0						

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-5 未实现: 读为 0

bit 4 **N:** 负标志位

此位用于有符号的算术运算 (以二进制补码方式进行)。它可以表示结果是否为负 (ALU MSB = 1)。

1 = 结果为负

0 = 结果为正

bit 3 **OV:** 溢出标志位

此位用于有符号的算术运算 (以二进制补码方式进行)。它表明运算结果溢出了 7 位二进制数的范围, 溢出导致符号位 (bit 7) 发生改变。

1 = 有符号算术运算中发生溢出 (本次算术运算)

0 = 未发生溢出

bit 2 **Z:** 全零标志位

1 = 算术运算或逻辑运算的结果为零

0 = 算术运算或逻辑运算的结果不为零

bit 1 **DC:** 半进位 / 借位位 (ADDWF、ADDLW、SUBLW 和 SUBWF 指令) ⁽¹⁾

1 = 结果的第 4 个低位发生了进位

0 = 结果的第 4 个低位未发生进位

bit 0 **C:** 进位 / 借位位 (ADDWF、ADDLW、SUBLW 和 SUBWF 指令) ⁽¹⁾

1 = 结果的最高有效位发生了进位

0 = 结果的最高有效位未发生进位

注 1: 对于借位, 极性是相反的。减法是通过加上第二个操作数的二进制补码来执行的。对于移位指令 (RRF 和 RLF), 此位来自源寄存器的最高位或最低位。

5.4 数据寻址模式

注: 当使能 PIC18 扩展指令集时, 核心 PIC18 指令集中某些指令的执行方式会发生改变。更多信息, 请参见第 5.5 节“数据存储器和扩展指令集”。

程序存储器只能用一种方式寻址(通过程序计数器), 而数据存储空间可用多种方式寻址。大部分指令的寻址模式都是固定的。其他指令可能使用最多三种模式, 根据它们所使用的操作数和是否使能了扩展指令集而定。

这些寻址模式为:

- 固有寻址
- 立即数寻址
- 直接寻址
- 间接寻址

当使能了扩展指令集(XINST 配置位 = 1)时, 还可使用另外一种寻址模式, 即立即数变址寻址模式。第 5.5.1 节“使用立即数偏移量进行变址寻址”将更详细讨论它的操作。

5.4.1 固有寻址和立即数寻址

很多 PIC18 控制指令根本不需要任何参数; 执行这些指令要么对整个器件造成影响, 要么仅隐式地针对一个寄存器进行操作。此寻址模式就是固有寻址。例如指令 SLEEP、RESET 和 DAW。

其他指令的工作方式与此类似, 但需要操作码中有其他显式的参数。由于需要一些立即数作为参数, 这种寻址模式被称为立即数寻址。例如 ADDLW 和 MOVLW, 它们分别向 W 寄存器加或移入立即数值。其他立即数寻址指令, 例如 CALL 和 GOTO, 它们包括一个 20 位的程序存储器地址。

5.4.2 直接寻址

直接寻址在操作码中指定操作的全部或部分源地址和 / 或目标地址。这些选项由指令附带的参数指定。

在核心 PIC18 指令集中, 针对位和针对字节的指令默认情况下使用直接寻址。所有这些指令都包含某个 8 位的立即数地址作为其最低有效字节。此地址指定数据 RAM 的某个存储区中寄存器的地址(第 5.3.3 节“通用寄存器文件”)或快速操作存储区(第 5.3.2 节“快速操作存储区”)中作为指令数据源的单元地址。

快速操作 RAM 位“a”决定地址的解析方式。当“a”为 1 时, BSR(第 5.3.1 节“存储区选择寄存器(BSR)”)的内容将和指令中的直接地址一起用于确定寄存器的完整 12 位地址。当“a”为 0 时, 此直接地址将被解析为快速操作存储区中的一个寄存器。使用快速操作 RAM 的寻址模式有时也被称为直接强制寻址模式。

有几条指令, 例如 MOVFF, 在操作码中包含完整的 12 位地址(源地址或目标地址)。在这些情况下, BSR 被完全忽略。

保存操作结果的目标寄存器由目标位“d”确定。当“d”为 1 时, 结果被存回源寄存器并覆盖原来的内容。当“d”为 0 时, 结果被存储在 W 寄存器中。没有“d”参数的指令的目标寄存器隐含在指令中, 这些指令的目标寄存器是正在操作的目标寄存器或 W 寄存器。

5.4.3 间接寻址

间接寻址允许用户访问数据存储器中的单元而无需在指令中给出一个固定的地址。这种寻址模式是通过使用文件选择寄存器(File Select Register, FSR)作为指向被读写单元的指针实现的。由于 FSR 本身作为特殊文件寄存器位于 RAM 中, 因此也可在程序控制中直接操作它们。这使得 FSR 对于在数据存储器中实现诸如表和数组等数据结构非常有用。

也可以使用间接文件操作数(Indirect File Operand, INDF)进行寄存器间接寻址。这种操作允许自动递增、递减或偏移指针, 从而自动操作指针的值。它通过使用循环提高代码执行效率, 如例 5-5 所示的清零整个 RAM 存储区的操作。

例 5-5: 如何使用间接寻址清零 RAM (BANK 1)

```
LFSR    FSRO, 100h ;  
NEXT    CLRF    POSTINCO ; Clear INDF  
          ; register then  
          ; inc pointer  
          ;  
          ; BTFS S FSROH, 1 ; All done with  
          ; Bank1?  
          ;  
          ; BRA     NEXT    ; NO, clear next  
CONTINUE                      ; YES, continue
```

5.4.3.1 FSR 寄存器和 INDF 操作数

间接寻址的核心是三组寄存器 FSR0、FSR1 和 FSR2。每组寄存器都含有一对 8 位寄存器：FSRnH 和 FSRnL。每对 FSR 保存一个 12 位值，因此 FSRnH 寄存器的高 4 位未使用。12 位 FSR 值可以线性寻址数据存储器的整个空间。因此，FSR 寄存器对被用作数据存储器的地址指针。

间接寻址是通过一组间接文件操作数 (INDF0 到 INDF2) 完成的。这些操作数可被看作“虚拟”寄存器：它们被映射到 SFR 空间而不是物理实现的。对特定的 INDF 寄存器执行读或写操作实际上访问的是与之对应的一对 FSR 寄存器。例如，读 INDF1 就是读 FSR1H:FSR1L 指向地址中的数据。使用 INDF 寄存器作为操作数的指令实际上使用相应 FSR 的内容作为指向指令目标地址的指针。INDF 操作数只是使用指针的一种简便方法。

由于间接寻址使用完整的 12 位地址，因此没有必要进行数据 RAM 分区。所以 BSR 的当前内容和快速操作 RAM 位对于确定目标地址没有影响。

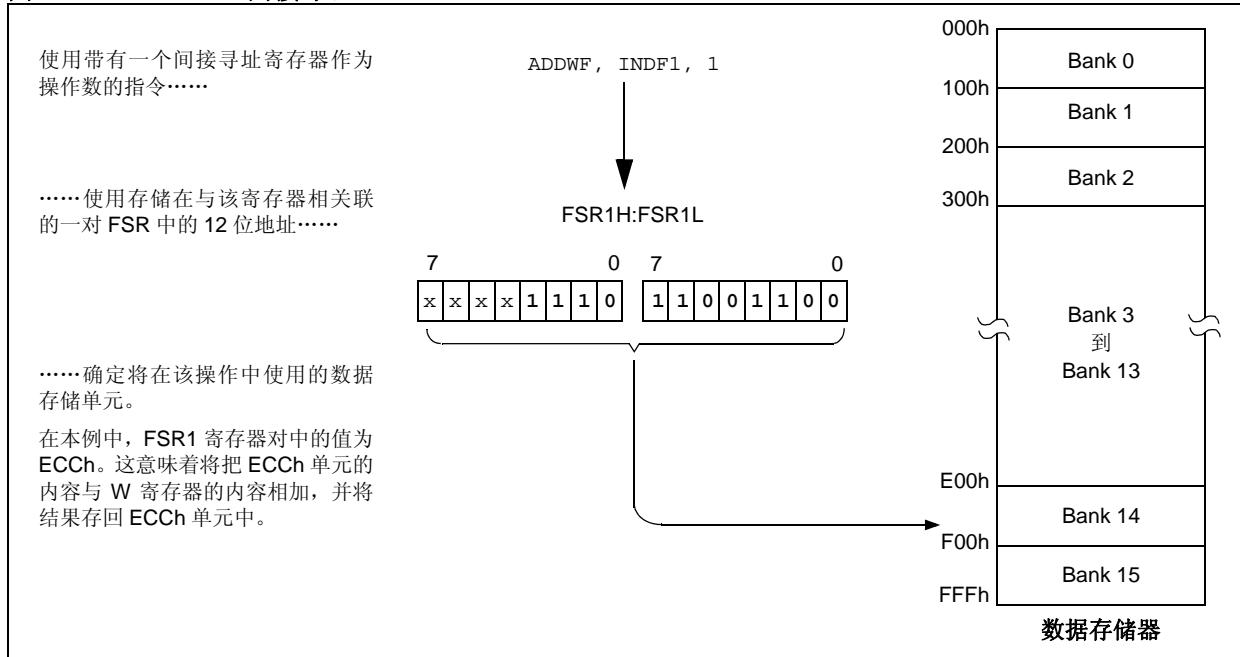
5.4.3.2 FSR 寄存器和 POSTINC、POSTDEC、PREINC 以及 PLUSW

除了 INDF 操作数之外，每对 FSR 寄存器还有 4 个额外的间接操作数。和 INDF 一样，它们也都是不能直接读写的“虚拟”寄存器。访问这些寄存器实际上访问的是与之相关的一对 FSR 寄存器所指向的地址单元，并对 FSR 值进行特定的操作。这些寄存器是：

- POSTDEC：访问 FSR 指向的地址单元，然后将 FSR 的值自动减 1
- POSTINC：访问 FSR 指向的地址单元，然后将 FSR 的值自动加 1
- PREINC：将 FSR 的值自动加 1，然后在操作中使用 FSR 指向的地址单元
- PLUSW：将 W 寄存器中有符号的值（从 -127 到 128）与 FSR 中的值相加，并在操作中使用结果指向的地址单元

在本文中，访问 INDF 寄存器使用相关 FSR 寄存器中的值（不会更改此值）。同样，访问 PLUSW 寄存器是将 W 寄存器中的值作为 FSR 的偏移量；该操作不会改变 W 或 FSR 中的值。访问其他虚拟寄存器均会更改 FSR 寄存器的值。

图 5-10：间接寻址



使用 POSTDEC、POSTINC 和 PREINC 对 FSR 进行操作会影响整对寄存器：即，**FSRnL** 寄存器从 FFh 溢出到 00h 并向 **FSRnH** 寄存器进位。但这些操作的结果不会更改 STATUS 寄存器中的任何标志位（如 Z、N 和 OV 等）。

PLUSW 寄存器可用于在数据存储空间实现变址寻址。通过操作 W 寄存器中的值，用户可以访问相对当前指针地址有固定偏移量的地址单元。在某些应用中，该功能可用于在数据存储器内部实现某些强大的程序控制结构，如软件堆栈。

5.4.3.3 通过 FSR 对其他 FSR 进行操作

在某些特殊情况下，间接寻址操作以其他 FSR 或虚拟寄存器作为目标。例如，使用 FSR 指向一个虚拟寄存器会导致操作不成功。假设如下特殊情况：**FSR0H:FSR0L** 保存的是 **INDF1** 的地址 **FE7h**。尝试使用 **INDF0** 作为操作数读取 **INDF1** 的值，将返回 **00h**。尝试使用 **INDF0** 作为操作数写入 **INDF1**，将会导致执行一条 NOP 指令。

另一方面，使用虚拟寄存器对一对 FSR 寄存器进行写操作可能会产生与预期不同的结果。在这些情况下，会将值写入一对 FSR 寄存器，但 FSR 不会递增或递减。因此，写入 **INDF2** 或 **POSTDEC2** 寄存器时会把同样的值写入 **FSR2H:FSR2L**。

由于 FSR 是映射到 SFR 空间中的物理寄存器，所以可以通过所有直接寻址来操作它们。用户在使用这些寄存器时应该特别小心，尤其是在代码使用间接寻址时。

同样，通常允许通过间接寻址对所有其他 SFR 进行操作。用户在进行此类操作时应该特别小心，以免不小心更改设置从而影响器件操作。

5.5 数据存储器和扩展指令集

使能 PIC18 扩展指令集（**XINST** 配置位 = 1）显著改变了数据存储器及其寻址的某些方面。特别是，许多核心 PIC18 指令使用快速操作存储区的方式有所不同。这是由于扩展指令集引入了对数据存储空间的新的寻址模式。

同样需要了解哪些部分保持不变。数据存储空间的大小及其线性寻址模式都不会改变。SFR 映射也保持不变。核心 PIC18 指令也仍然以直接和间接寻址模式进行操作；固有和立即数寻址指令操作照旧。**FSR0** 和 **FSR1** 的间接寻址模式也保持不变。

5.5.1 使用立即数偏移量进行变址寻址

使能 PIC18 扩展指令集将更改快速操作 RAM 中使用 **FSR2** 寄存器对进行间接寻址的方式。在适当的条件下，使用快速操作存储区的指令（即绝大多数针对位和针对字节的指令）可以利用指令中的偏移量来执行变址寻址。这种特殊的寻址模式被称为使用立即数偏移量的变址寻址或立即数变址寻址模式。

使用扩展指令集时，这种寻址模式有如下要求：

- 强制使用快速操作存储区 (**a** = 0)；且
- 文件地址参数要小于或等于 **5Fh**。

在这些条件下，指令的文件地址不会被解析为地址的低字节（在直接寻址中和 **BSR** 一起使用），或快速操作存储区中的 8 位地址，而是被解析为由 **FSR2** 指定的地址指针的偏移量。将该偏移量与 **FSR2** 的内容相加以获取操作的目标地址。

5.5.2 受立即数变址寻址模式影响的指令

任何使用直接寻址模式的核心 PIC18 指令均会受到立即数变址寻址模式的潜在影响，包括所有针对字节和针对位的指令，即核心 PIC18 指令集中几乎一半的指令。只有使用固有寻址或立即数寻址模式的指令不受影响。

此外，如果针对字节和针对位的指令使用快速操作存储区（快速操作 RAM 位为 1）或包含 60h 以上的文件地址，它们也不受影响。符合这些条件的指令会像以前一样执行。图 5-11 给出了当使能扩展指令集时，各种寻址模式之间的对比。

那些想要在立即数变址寻址模式中使用针对位或针对字节的指令的用户，应该注意此模式下汇编语法的改变。在第 24.2.1 节“扩展指令的语法”中对此进行了更详细的说明。

PIC18F2XK20/4XK20

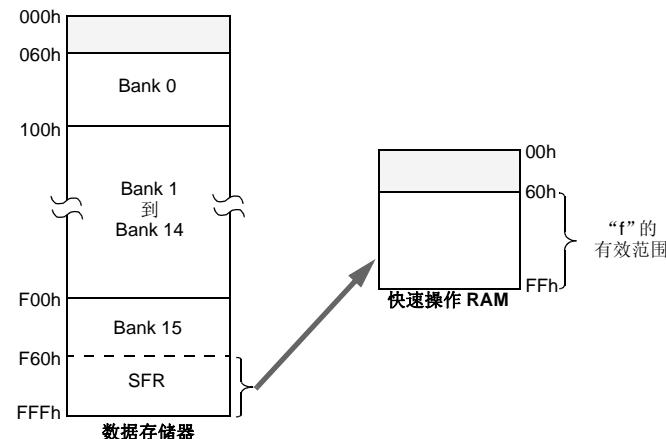
图 5-11：针对位和针对字节的指令的寻址模式对比（使能了扩展指令集）

示例指令: ADDWF, f, d, a (操作码: 0010 01da ffff ffff)

当 $a = 0$ 且 $f \geq 60h$ 时:

此指令以直接强制模式执行。“f”被解析为快速操作 RAM 中 060h 到 OFFh 之间的单元地址，该地址也是数据存储器的 F60h 到 FFFh (Bank 15)。

不可用此模式寻址地址低于 60h 的单元。



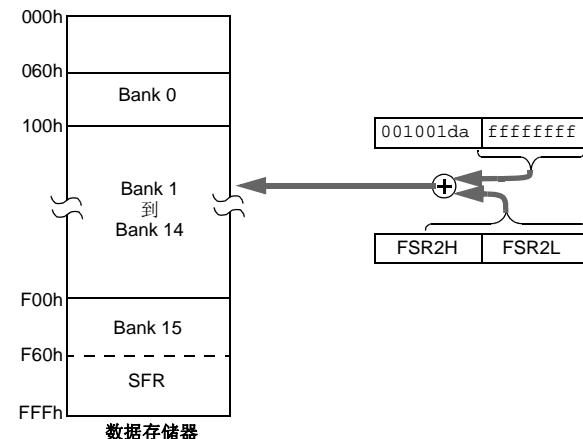
当 $a = 0$ 且 $f \leq 5Fh$ 时:

此指令以立即数变址寻址模式执行。“f”被解析为 FSR2 中地址值的偏移量。将这两个值相加可以得到指令的目标寄存器的地址。此地址可以在数据存储空间的任何地方。

注意在此模式中，正确的语法如下：

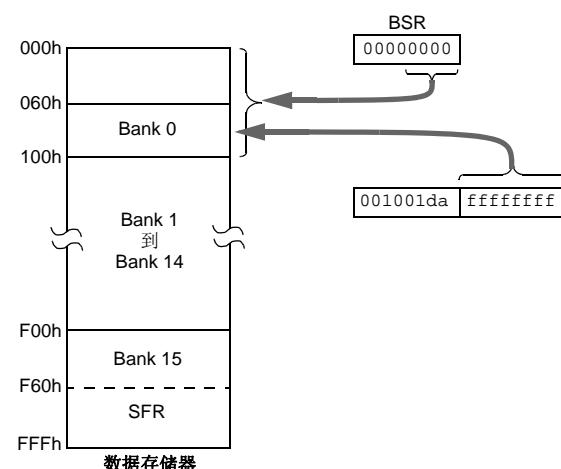
ADDWF [k], d

其中 “k” 就是 “f”。



当 $a = 1$ (f 可为任何值) 时:

指令以直接寻址模式（也称为直接长地址寻址模式）执行。“f”被解析为数据存储空间的 16 个存储区之一中的一个单元地址。存储区由存储区选择寄存器（BSR）指定。此地址可以位于数据存储空间的任何已实现存储区中。



5.5.3 在立即数变址模式下映射快速操作存储区

使用立即数变址寻址模式能有效改变快速操作 RAM 前 96 个地址单元（00h 到 5Fh）的映射方式。此模式映射由用户定义的、可位于数据存储空间中任何地方的“窗口”内容，而不仅仅映射 Bank 0 底部的内容。FSR2 的值定义映射到窗口的地址的下边界，而上边界则由 FSR2 加 95（5Fh）决定。地址大于 5Fh 的快速操作 RAM 的映射方法如前所述（见第 5.3.2 节“快速操作存储区”）。图 5-12 给出了在此寻址模式下重映射快速操作存储区的示例。

快速操作存储区的重映射仅适用于立即数变址寻址模式。使用 BSR（快速操作 RAM 位为 1）的操作和以前一样继续使用直接寻址模式。

5.6 PIC18 指令的执行和扩展指令集

使能扩展指令集将 8 条其他命令添加到现有的 PIC18 指令集中。这些指令如第 24.2 节“扩展指令集”中所述执行。

图 5-12： 使用立即数变址寻址模式重映射快速操作存储区

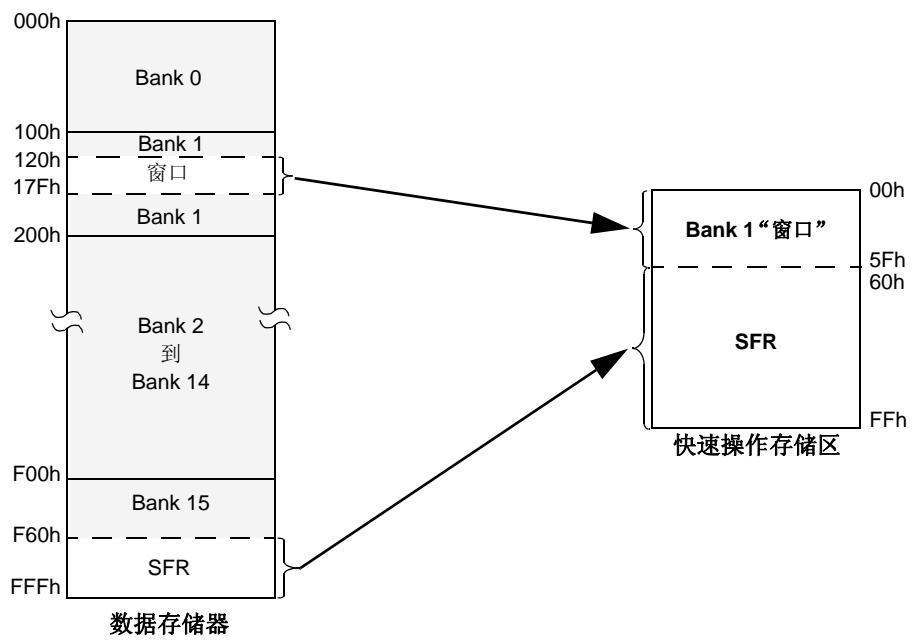
示例：

```
ADDWF f, d, a
FSR2H:FSR2L = 120h
```

从 FSR2 指针（120h）到 FSR2 指针加 05Fh（17Fh）区域内的存储单元被映射到快速操作 RAM 的底部（00h-05Fh）。

从 F60h 到 FFFh 的特殊功能寄存器被映射到 60h 到 FFh，和常规寻址一样。

Bank 0 中低于 5Fh 的地址仍然可以使用 BSR 对它们进行寻址。



PIC18F2XK20/4XK20

注:

6.0 闪存程序存储器

在整个 VDD 范围内，正常操作期间，闪存程序存储器都是可读写、可擦除的。

读程序存储器时，每次读取一个字节。写程序存储器时，根据具体的器件，每次写一个 64、32 或 16 字节的块（见表 6-1）。擦除程序存储器时，每次擦除一个 64 字节的块。由于写和擦除块大小的差别，所以需要 1 至 4 次块写操作来恢复单次块擦除操作的内容。用户代码不能执行批量擦除操作。

表 6-1：写 / 擦除操作块大小

器件	写操作块大小 (字节)	擦除操作块大小 (字节)
PIC18F43K20 和 PIC18F23K20	16	64
PIC18F24K20、 PIC18F25K20、 PIC18F44K20 和 PIC18F45K20	32	64
PIC18F26K20 和 PIC18F46K20	64	64

在擦写程序存储器时，系统会停止取指令直到操作完成。擦写期间不能访问程序存储器，因此也就无法执行代码。由内部编程定时器来终止程序存储器的擦写操作。

写入程序存储器的值不一定非要是有效指令。执行存储无效指令的程序存储单元会导致执行 NOP。

6.1 表读与表写

为了读写程序存储器，有两个操作可供处理器在程序存储空间和数据 RAM 之间传送字节：

- 表读（TBLRD）
- 表写（TBLWT）

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

表读操作从程序存储器直接取一个字节的数据，然后将其放入 TABLAT 寄存器。图 6-1 显示了表读操作。

表写操作将一个字节的数据从 TABLAT 寄存器存储到写操作块保持寄存器中。第 6.5 节“写闪存程序存储器”详细介绍了将保持寄存器的内容写入程序存储器的过程。图 6-2 显示了程序存储器和数据 RAM 之间的一次表写操作。

表操作以字节为单位。包含数据而非程序指令的表不需要按字对齐。因此，表可以在任何字节地址开始和结束。如果使用表写操作向程序存储器写入可执行代码，程序指令必须按字对齐。

图 6-1：表读操作

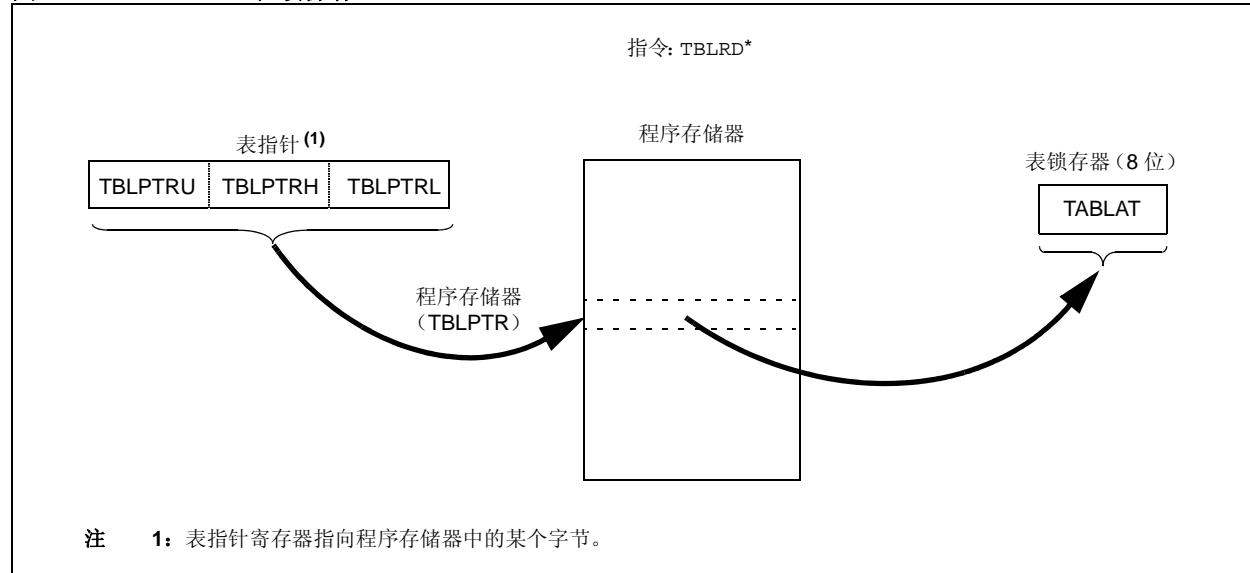
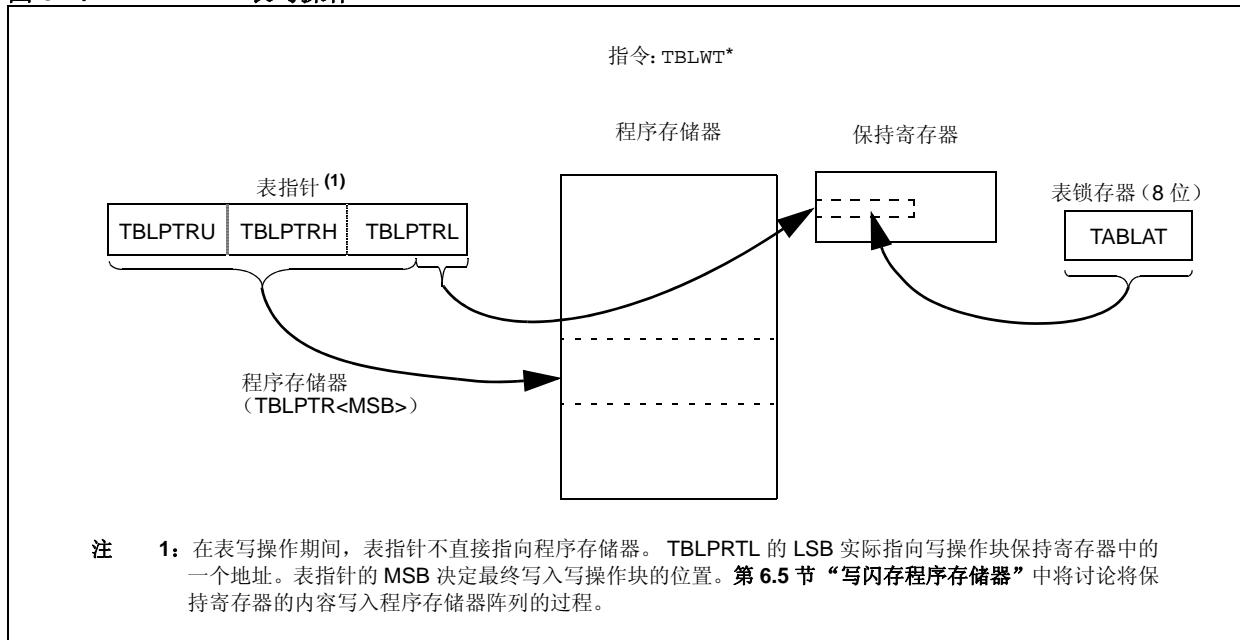


图 6-2: 表写操作



6.2 控制寄存器

TBLRD 和 TBLWT 指令要用到几个控制寄存器。这些寄存器包括:

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

6.2.1 EECON1 和 EECON2 寄存器

EECON1 寄存器 (寄存器 6-1) 是存储器访问的控制寄存器。EECON2 寄存器不是实际存在的寄存器, 专用于存储器的擦写操作。读 EECON2 将得到全 0。

EPPGD 控制位决定访问的是程序存储器还是数据 EEPROM 存储器。当 EPPGD 清零时, 任何后续操作都将针对数据 EEPROM 存储器进行。当 EPPGD 置 1 时, 任何后续操作都将针对程序存储器进行。

CFGs 控制位决定访问的是配置 / 校准寄存器还是程序存储器 / 数据 EEPROM 存储器。当 CFGs 置 1 时, 不管 EPPGD 的值如何, 后续操作将针对配置寄存器进行 (见第 23.0 节“CPU 的特殊功能”)。当 CFGs 清零时, 则由 EPPGD 来选择访问的存储器。

FREE 位允许进行程序存储器的擦除操作。当 FREE 置 1 时, 擦除操作由下一条 WR 命令启动。当 FREE 清零时, 则仅使能写操作。

当 WREN 位置 1 时, 允许进行写操作。上电时, WREN 位被清零。

WRERR 位在 WR 位置 1 时由硬件置 1, 在内部编程定时器超时、写操作结束时被清零。

注: 在正常操作期间, WRERR 读为 1。这表明写操作被复位提早终止或进行了不合法的写操作。

WR 控制位用于启动写操作。用固件只能将 WR 位置 1 而无法清零。写操作完成时, 由硬件将 WR 位清零。

注: 当写操作完成时, PIR2 寄存器的 EEIF 中断标志位被置 1。EIF 标志保持置 1 状态直到被固件清零为止。

寄存器 6-1: EECON1: 数据 EEPROM 控制寄存器 1

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

图注:

R = 可读位

W = 可写位

S = 该位可用软件置 1, 但不能清零

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	EEPGD: 闪存程序存储器或数据 EEPROM 存储器选择位 1 = 访问闪存程序存储器 0 = 访问数据 EEPROM 存储器
bit 6	CFGs: 闪存程序存储器 / 数据 EEPROM 存储器或配置寄存器选择位 1 = 访问配置寄存器 0 = 访问闪存程序存储器或数据 EEPROM 存储器
bit 5	未实现: 读为 0
bit 4	FREE: 闪存行 (块) 擦除使能位 1 = 在下一条 WR 命令时擦除 TBLPTR 指定的程序存储器块 (擦除操作完成后清零) 0 = 仅执行写操作
bit 3	WRERR: 闪存程序存储器 / 数据 EEPROM 存储器错误标志位 ⁽¹⁾ 1 = 写操作提早终止 (由于正常操作中自定时编程期间的任何复位, 或不合法的写操作) 0 = 写操作完成
bit 2	WREN: 闪存程序存储器 / 数据 EEPROM 存储器写使能位 1 = 允许对闪存程序存储器 / 数据 EEPROM 存储器的写周期 0 = 禁止对闪存程序存储器 / 数据 EEPROM 存储器的写周期
bit 1	WR: 写控制位 1 = 启动数据 EEPROM 擦除 / 写周期或者程序存储器的擦除周期或写周期。 (操作是自定时的, 一旦写操作完成, 该位即由硬件清零。用软件只能将 WR 置 1, 但不能清零。) 0 = EEPROM 写周期完成
bit 0	RD: 读控制位 1 = 启动 EEPROM 读操作 (读操作需要一个指令周期。RD 由硬件清零。用软件只能将 RD 置 1, 但不能清零。EEP GD = 1 或 CFGs = 1 时, RD 位无法置 1。) 0 = 不启动 EEPROM 读操作

注 1: 当发生 WRERR 时, EEPGD 和 CFGs 位不会被清零。这样可以跟踪错误情况。

6.2.2 TABLAT——表锁存寄存器

表锁存器 (TABLAT) 是映射到 SFR 空间的一个 8 位寄存器。表锁存器用于在程序存储器和数据 RAM 之间传输数据时保存 8 位数据。

6.2.3 TBLPTR——表指针寄存器

表指针 (TBLPTR) 寄存器在程序存储器中以字节为单位进行寻址。TBLPTR 由 3 个 SFR 寄存器组成：表指针最高字节、表指针高字节和表指针低字节 (TBLPTRU:TBLPTRH:TBLPTRL)。这 3 个寄存器合起来组成一个 22 位宽的指针。其中低 21 位允许器件寻址最大 2 MB 的程序存储空间。第 22 位则允许访问器件 ID、用户 ID 和配置位。

TBLRD 和 TBLWT 指令要使用表指针寄存器 TBLPTR。这些指令可以基于表操作以 4 种方法之一更新 TBLPTR。表 6-2 列出了这些操作。这些操作只会影响 TBLPTR 的低 21 位。

6.2.4 表指针边界

TBLPTR 用于读、写和擦除闪存程序存储器。

当执行 TBLRD 时，TBLPTR 的所有 22 位决定将程序存储器的哪个字节直接读入 TABLAT 寄存器。

当执行 TBLWT 时，TBLPTR 寄存器中的字节将写入保持寄存器（而不是闪存），以准备进行程序存储器写操作。保持寄存器包含一个写操作块，写操作块因器件而异（见表 6-1）。TBLPTRL 寄存器的 3、4 或 5 个 LSb 决定数据写入保持寄存器块中的哪个具体地址。在 TBLWT 操作期间，表指针的 MSB 没有作用。

当执行程序存储器写操作时，整个保持寄存器块的内容写入到闪存中，写入地址由 TBLPTR 的 MSb 决定。在闪存写操作期间，3、4 或 5 个 LSb 被忽略。更多详细信息，请参见第 6.5 节“写闪存程序存储器”。

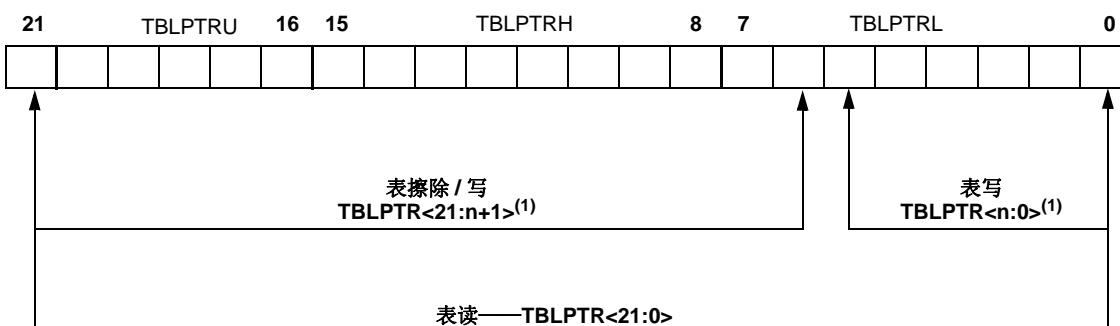
当执行擦除程序存储器时，表指针寄存器的高 16 位 (TBLPTR<21:6>) 指向将要擦除的 64 字节块。最低有效位 (TBLPTR<5:0>) 被忽略。

图 6-3 说明了基于闪存程序存储器操作的 TBLPTR 相关边界。

表 6-2： 执行 TBLRD 和 TBLWT 指令的表指针操作

示例	表指针操作
TBLRD*	
TBLWT*	不修改 TBLPTR
TBLRD*+ TBLWT*+	TBLPTR 在读 / 写后递增
TBLRD*- TBLWT*-	TBLPTR 在读 / 写后递减
TBLRD+* TBLWT+*	TBLPTR 在读 / 写前递增

图 6-3： 基于操作的表指针边界



注 1: n = 3、4、5 或 6，分别对应于 8、16、32 或 64 字节的块大小。

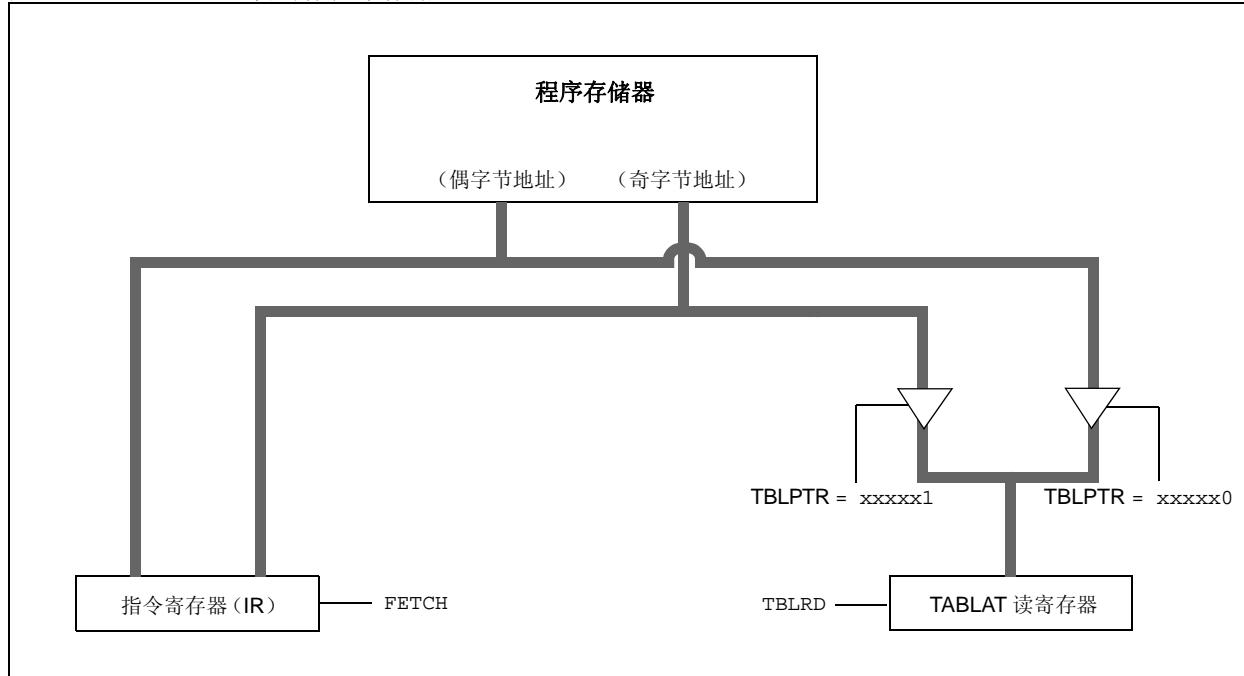
6.3 读闪存程序存储器

TBLRD 指令用于从程序存储器读取数据并放入数据 RAM。表读操作每次从程序存储器读取一个字节。

TBLPTR 指向程序存储空间的某个字节地址。执行 TBLRD 指令将把指向的字节装入 TABLAT。此外，还可以自动修改 TBLPTR 以进行下一次表读操作。

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

图 6-4：读闪存程序存储器



例 6-1：读闪存程序存储器的一个字

```

MOVLW  CODE_ADDR_UPPER      ; Load TBLPTR with the base
MOVWF  TBLPTRU               ; address of the word
MOVLW  CODE_ADDR_HIGH
MOVWF  TBLPTRH
MOVLW  CODE_ADDR_LOW
MOVWF  TBLPTRL

READ_WORD
TBLRD*+
MOVF   TABLAT, W             ; read into TABLAT and increment
MOVWF WORD_EVEN              ; get data
TBLRD*+
MOVFW TABLAT, W             ; read into TABLAT and increment
MOVF   WORD_ODD              ; get data
    
```

6.4 擦除闪存程序存储器

最小擦除块大小为 32 个字或 64 字节。只有通过使用外部编程器，或通过 ICSP™ 控制，才能够批量擦除更大的程序存储器块。闪存阵列不支持字擦除。

当单片机自身启动一个擦除序列时，会擦除一个 64 字节的程序存储器块。高 16 位 TBLPTR<21:6> 指向要擦除的块。TBLPTR<5:0> 被忽略。

擦除操作由 EECON1 寄存器控制。必须将 EEPGD 位置 1 以指向闪存程序存储器。WREN 位必须被置 1 以便能写操作。FREE 位被置 1 以选择擦除操作。

EECON2 的写操作启动序列（在第 6.4.1 节“闪存程序存储器擦除序列”中显示为步骤 4 至 6）用于防止意外的写操作。这有时称为长写操作。

擦除内部闪存必须执行长写操作。在长写周期中，指令暂停执行。由内部编程定时器终止长写操作。

6.4.1 闪存程序存储器擦除序列

擦除内部程序存储器块的过程如下：

1. 将要擦除的块地址装入表指针寄存器。
2. 设置 EECON1 寄存器来执行擦除操作：
 - 将 EEPGD 位置 1 以指向程序存储器；
 - 将 CFGS 位清零以访问程序存储器；
 - 将 WREN 位置 1 以便能写操作；
 - 将 FREE 位置 1 以便能擦除操作。
3. 禁止中断。
4. 将 55h 写入 EECON2。
5. 将 0AAh 写入 EECON2。
6. 将 WR 位置 1。这将开始块擦除周期。
7. CPU 在擦除期间（使用内部定时器约为 2 ms）将会停止工作。
8. 重新允许中断。

例 6-2：擦除闪存程序存储器块

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

必需的序列

6.5 写闪存程序存储器

编程块大小为 8、32 或 64 字节，这取决于器件（见表 6-1）。不支持字或字节编程。

在内部使用表写指令将需要写入闪存的内容装入保持寄存器中。保持寄存器的数量与写操作块中的字节数相同（见表 6-1）。

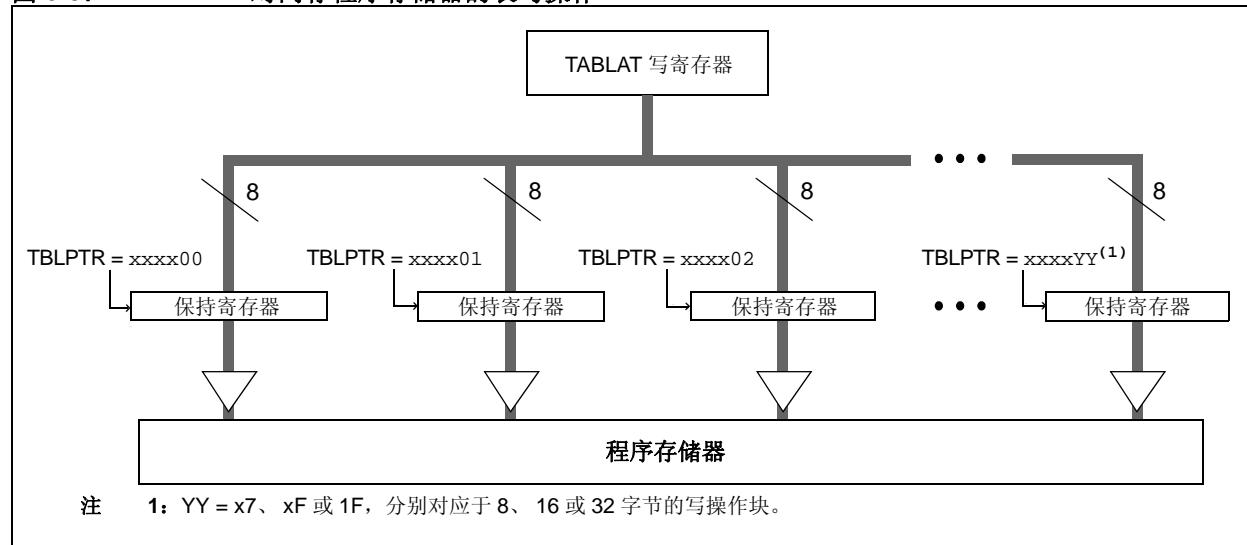
由于表锁存器（TABLAT）只是单字节寄存器，所以每次编程操作 TBLWT 指令可能需要执行 8、32 或 64 次（取决于具体器件）。因为只写保持寄存器，所以所有的表写操作实际上都是短写。在数据写入所有保持寄存器之后，该存储器块的编程操作通过以下操作启动：先配置 EECON1 寄存器来进行程序存储器写操作，然后执行长写序列。

对内部闪存编程要求使用长写操作。在长写周期中，指令暂停执行。由内部编程定时器终止长写操作。

由 EEPROM 片上定时器控制写入的时间。写入 / 擦除电压由片上的电荷泵产生，该电荷泵可以在器件的电压范围内工作。

注： 器件复位和写操作完成后保持寄存器的默认值为 FFh。将 FFh 写入保持寄存器不会修改其中的数值。这意味着可以修改程序存储器的各字节（假如不是想把任何位从 0 更改为 1）。当修改各字节时，无需在执行长写操作前装入所有保持寄存器。

图 6-5：对闪存程序存储器的表写操作



6.5.1 闪存程序存储器写序列

对内部程序存储单元编程的过程如下：

1. 将 64 字节读入 RAM。
2. 必要时更新 RAM 中的数据值。
3. 将要擦除的地址装入表指针寄存器。
4. 执行块擦除过程。
5. 将要写入的第一个字节的地址装入表指针寄存器。
6. 通过自动递增将 8、32 或 64 个字节块写入保持寄存器。
7. 设置 EECON1 寄存器来执行写操作：
 - 将 EEPGD 位置 1 以指向程序存储器；
 - 将 CFGS 位清零以访问程序存储器；
 - 将 WREN 位置 1 以便能字节写操作。

8. 禁止中断。
9. 将 55h 写入 EECON2。
10. 将 0AAh 写入 EECON2。
11. 将 WR 位置 1。这将开始写周期。
12. CPU 在写入期间（使用内部定时器约为 2 ms）将会停止工作。
13. 重新允许中断。
14. 对每个块重复步骤 6 至 13，直到全部 64 个字节都已写入。
15. 校验存储器（表读）。

该过程需要大约 6 ms 的时间来更新存储器的每个写操作块。例 6-3 给出了所需代码的示例。

注： 在将 WR 位置 1 前，表指针地址必须处于保持寄存器中的字节将要被写入的地址范围内。

PIC18F2XK20/4XK20

例 6-3：写闪存程序存储器

```
        MOVLW    D'64'                      ; number of bytes in erase block
        MOVWF    COUNTER
        MOVLW    BUFFER_ADDR_HIGH          ; point to buffer
        MOVWF    FSR0H
        MOVLW    BUFFER_ADDR_LOW
        MOVWF    FSR0L
        MOVLW    CODE_ADDR_UPPER          ; Load TBLPTR with the base
        MOVWF    TBLPTRU                 ; address of the memory block
        MOVLW    CODE_ADDR_HIGH
        MOVWF    TBLPTRH
        MOVLW    CODE_ADDR_LOW
        MOVWF    TBLPTRL
READ_BLOCK
        TBLRD*+
        MOVF     TABLAT, W                ; read into TABLAT, and inc
        MOVWF   POSTINC0                 ; get data
        DECFSZ  COUNTER                 ; store data
        BRA     READ_BLOCK               ; done?
MODIFY_WORD
        MOVLW    BUFFER_ADDR_HIGH          ; point to buffer
        MOVWF    FSR0H
        MOVLW    BUFFER_ADDR_LOW
        MOVWF    FSR0L
        MOVLW    NEW_DATA_LOW             ; update buffer word
        MOVWF   POSTINC0
        MOVLW    NEW_DATA_HIGH
        MOVWF    INDF0
ERASE_BLOCK
        MOVLW    CODE_ADDR_UPPER          ; load TBLPTR with the base
        MOVWF    TBLPTRU                 ; address of the memory block
        MOVLW    CODE_ADDR_HIGH
        MOVWF    TBLPTRH
        MOVLW    CODE_ADDR_LOW
        MOVWF    TBLPTRL
        BSF      EECON1, EEPGD            ; point to Flash program memory
        BCF      EECON1, CFGS
        BSF      EECON1, WREN
        BSF      EECON1, FREE
        BCF      INTCON, GIE              ; enable write to memory
                                         ; enable Erase operation
                                         ; disable interrupts
必需的序列
        MOVLW    55h
        MOVWF    EECON2                  ; write 55h
        MOVLW    0AAh
        MOVWF    EECON2                  ; write 0AAh
        BSF      EECON1, WR              ; start erase (CPU stall)
        BSF      INTCON, GIE             ; re-enable interrupts
        TBLRD*-
        MOVLW    BUFFER_ADDR_HIGH          ; dummy read decrement
        MOVWF    FSR0H
        MOVLW    BUFFER_ADDR_LOW
        MOVWF    FSR0L
WRITE_BUFFER_BACK
        MOVLW    BlockSize                ; number of bytes in holding register
        MOVWF    COUNTER
        MOVLW    D'64'/BlockSize          ; number of write blocks in 64 bytes
        MOVWF    COUNTER2
WRITE_BYTE_TO_HREGS
        MOVF     POSTINC0, W              ; get low byte of buffer data
        MOVWF    TABLAT                  ; present data to table latch
        TBLWT+*                           ; write data, perform a short write
                                         ; to internal TBLWT holding register.
```

例 6-3：写闪存程序存储器（续）

PROGRAM_MEMORY	DECFSZ COUNTER	; loop until holding registers are full	
	BRA WRITE_WORD_TO_HREGS		
	BSF EECON1, EEPGD		; point to Flash program memory
	BCF EECON1, CFGS		; access Flash program memory
	BSF EECON1, WREN		; enable write to memory
	BCF INTCON, GIE		; disable interrupts
	MOVLW 55h		
	MOVWF EECON2		; write 55h
	MOVLW 0AAh		
	MOVWF EECON2		; write 0AAh
必需的序列	BSF EECON1, WR		; start program (CPU stall)
	DCFSZ COUNTER2		; repeat for remaining write blocks
	BRA WRITE_BYTE_TO_HREGS		
	BSF INTCON, GIE		; re-enable interrupts
	BCF EECON1, WREN		; disable write to memory

6.5.2 写校验

根据具体应用，将写入存储器的值对照原始值进行校验是一个很好的编程习惯。在应用中，如果某些位的写次数接近规定极限值，就应该进行写校验。

6.5.3 意外终止写操作

如果由于意外事件（如掉电或意外复位）终止了写操作，应该对刚刚编程的存储单元进行校验，如有必要，还要重新进行编程。如果写操作在正常操作期间因 MCLR 复位或 WDT 超时复位而中断，则 WRERR 位将被置 1，用户可以检查该位以确定是否需要重写该单元。

6.5.4 防止误写操作的保护措施

为防止对闪存程序存储器的误写操作，必须遵循写操作的启动顺序。更多详细信息，请参见第 23.0 节“CPU 的特殊功能”。

6.6 代码保护期间闪存程序存储器的操作

关于闪存程序存储器代码保护的详细信息，请参见第 23.3 节“程序校验和代码保护”。

表 6-3：与闪存程序存储器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TBLPTRU	—	—	bit 21	程序存储器表指针最高字节 (TBLPTR<20:16>)					59
TBPLTRH	程序存储器表指针高字节 (TBLPTR<15:8>)								59
TBLPTRL	程序存储器表指针低字节 (TBLPTR<7:0>)								59
TABLAT	程序存储器表锁存器								59
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
EECON2	EEPROM 控制寄存器 2 (不是实际存在的寄存器)								61
EECON1	EEPGD	CFGs	—	FREE	WRERR	WREN	WR	RD	61
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62

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

PIC18F2XK20/4XK20

注:

7.0 数据 EEPROM 存储器

数据 EEPROM 是非易失性的存储器阵列，独立于数据 RAM 和程序存储器，用于程序数据的长期存储。它并不直接映射到寄存器文件或程序存储空间，而是通过特殊功能寄存器（SFR）来间接寻址。在整个 VDD 范围内的正常运行期间，EEPROM 是可读写的。

有 5 个 SFR 用于读写数据 EEPROM 以及程序存储器。它们是：

- EECON1
- EECON2
- EEDATA
- EEADR
- EEADRH

数据 EEPROM 允许以字节为单位读写。当与数据存储器模块接口时，EEDATA 存放 8 位读写数据，而 EEADR:EEADRH 寄存器对存放被访问的 EEPROM 存储单元的地址。

EEPROM 数据存储器具有高耐擦写次数。字节写操作会自动擦除目标存储单元并写入新数据（在写入前擦除）。写入时间由片上定时器控制，其值根据电压、温度和不同的芯片而不同。具体的限制值，请参见 D122（第 26.0 节“电气特性”中的表 26.10）。

7.1 EEADR 和 EEADRH 寄存器

EEADR 寄存器用于寻址数据 EEPROM 以进行读写操作。8 位的寄存器可寻址 256 字节（00h 至 FFh）的存储器范围。通过增加 2 个额外的地址位，EEADRH 寄存器将范围扩展到 1024 字节。

7.2 EECON1 和 EECON2 寄存器

对数据 EEPROM 的访问由 EECON1 和 EECON2 两个寄存器控制。它们也用来控制对程序存储器的访问，使用方法与访问数据 EEPROM 类似。

EECON1 寄存器（寄存器 7-1）是数据和程序存储器访问的控制寄存器。控制位 EEPGD 决定访问的是程序存储器还是数据 EEPROM 存储器。当 EEPGD 位清零时，操作将访问数据 EEPROM 存储器。当 EEPGD 位置 1 时，将访问程序存储器。

控制位 CFGS 决定访问的是配置寄存器还是程序存储器 / 数据 EEPROM 存储器。当 CFGS 位置 1 时，后续操作将访问配置寄存器。当 CFGS 位清零时，则由 EEPGD 位来选择具体访问闪存程序存储器还是数据 EEPROM 存储器。

当 WREN 位置 1 时，允许进行写操作。上电时，WREN 位被清零。

WRERR 位在 WR 位置 1 时由硬件置 1，在内部编程定时器超时、写操作结束时被清零。

注： 在正常操作期间，WRERR 读为 1。这表明写操作被复位提早终止或进行了不合法的写操作。

WR 控制位用于启动写操作。用软件只能将该位置 1 而无法清零。在写操作完成后，由硬件将其清零。

注： 当写操作完成时，PIR2 寄存器的 EIF 中断标志位被置 1。它必须用软件清零。

控制位 RD 和 WR 分别启动读和擦写操作。这些位由固件置 1，并在操作完成时由硬件清零。

当访问程序存储器（EEPGD = 1）时，RD 位无法置 1。程序存储器是通过表读指令读取的。关于表读的信息，请参见第 6.1 节“表读与表写”。

EECON2 寄存器不是实际存在的寄存器。它专用于存储器写和擦除过程。读 EECON2 将得到全 0。

PIC18F2XK20/4XK20

寄存器 7-1: EECON1: 数据 EEPROM 控制寄存器 1

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

图注:

R = 可读位 W = 可写位

S = 该位可用软件置 1, 但不能清零

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

- bit 7 **EEPGD:** 闪存程序存储器或数据 EEPROM 存储器选择位
1 = 访问闪存程序存储器
0 = 访问数据 EEPROM 存储器
- bit 6 **CFGs:** 闪存程序存储器 / 数据 EEPROM 存储器或配置寄存器选择位
1 = 访问配置寄存器
0 = 访问闪存程序存储器或数据 EEPROM 存储器
- bit 5 **未实现:** 读为 0
- bit 4 **FREE:** 闪存行 (块) 擦除使能位
1 = 在下一条 WR 命令时擦除 TBLPTR 指定的程序存储器块 (擦除操作完成后清零)
0 = 仅执行写操作
- bit 3 **WRERR:** 闪存程序存储器 / 数据 EEPROM 存储器错误标志位 ⁽¹⁾
1 = 写操作提早终止 (由于正常操作中自定时编程期间的任何复位, 或不合法的写操作)
0 = 写操作完成
- bit 2 **WREN:** 闪存程序存储器 / 数据 EEPROM 存储器写使能位
1 = 允许对闪存程序存储器 / 数据 EEPROM 存储器的写周期
0 = 禁止对闪存程序存储器 / 数据 EEPROM 存储器的写周期
- bit 1 **WR:** 写控制位
1 = 启动数据 EEPROM 擦除 / 写周期或者程序存储器的擦除周期或写周期。
(操作是自定时的, 一旦写操作完成, 该位即由硬件清零。用软件只能将 WR 位置 1, 但不能清零。)
0 = EEPROM 写周期完成
- bit 0 **RD:** 读控制位
1 = 启动 EEPROM 读操作 (读操作需要一个指令周期。RD 由硬件清零。用软件只能将 RD 位置 1, 但不能清零。EEPGD = 1 或 CFGs = 1 时, RD 位无法置 1。)
0 = 不启动 EEPROM 读操作

注 1: 当发生 WRERR 时, EEPGD 和 CFGS 位不会被清零。这样可以跟踪错误情况。

7.3 读数据 EEPROM 存储器

要读取数据存储单元，用户必须将地址写入 EEADR 寄存器，清零 EECON1 寄存器的 EEPGD 控制位，然后将控制位 RD 置 1。可在下一个指令周期访问该数据；因此，EEDATA 寄存器可由下一条指令读取。EEDATA 将把此值保存至下一次读取或用户向该单元写入数据时（写操作）为止。

基本过程如例 7-1 中所示。

7.4 写数据 EEPROM 存储器

要向 EEPROM 数据存储单元写入数据，必须首先将地址写入 EEADR 寄存器，并将数据写入 EEDATA 寄存器。必须遵循例 7-2 中的序列启动写周期。

如果没有完全遵循该指令序列（即首先将 55h 写入 EECON2，随后将 0AAh 写入 EECON2，最后将 WR 位置 1）写每个字节，将不会启动写操作。强烈建议在这段代码执行期间禁止中断。

例 7-1：读数据 EEPROM

```
MOVLW DATA_EE_ADDR      ;  
MOVWF EEADR              ; Data Memory Address to read  
BCF EECON1, EEPGD         ; Point to DATA memory  
BCF EECON1, CFGS          ; Access EEPROM  
BSF EECON1, RD             ; EEPROM Read  
MOVF EEDATA, W            ; W = EEDATA
```

例 7-2：写数据 EEPROM

必需的序列	MOVLW DATA_EE_ADDR_LOW ;
	MOVWF EEADR ; Data Memory Address to write
	MOVLW DATA_EE_ADDR_HI ;
	MOVWF EEADRH ;
	MOVLW DATA_EE_DATA ;
	MOVWF EEDATA ; Data Memory Value to write
	BCF EECON1, EEPGD ; Point to DATA memory
	BCF EECON1, CFGS ; Access EEPROM
	BSF EECON1, WREN ; Enable writes
	BCF INTCON, GIE ; Disable Interrupts
	MOVLW 55h ;
	MOVWF EECON2 ; Write 55h
	MOVLW 0AAh ;
	MOVWF EECON2 ; Write 0AAh
	BSF EECON1, WR ; Set WR bit to begin write
BSF INTCON, GIE ; Enable Interrupts	
; User code execution	
BCF EECON1, WREN ; Disable writes on write complete (EEIF set)	

此外，必须将 EECON1 中的 WREN 位置 1 以便能写操作。这种机制可防止由于意外执行代码（即程序失控）导致误写数据 EEPROM。除了更新 EEPROM 时以外，WREN 位应始终保持清零。WREN 位不会被硬件清零。

一个写序列启动后，EECON1、EEADR 和 EEDATA 不能被修改。除非 WREN 位置 1，否则 WR 位将禁止置 1。WR 和 WREN 不能被同一条指令置 1。

写周期完成后，WR 位由硬件清零并且 EEPROM 中断标志位 EEIF 被置 1。用户可以允许此中断或查询此位。EEIF 必须用软件清零。

7.5 写校验

根据具体应用，将写入存储器的值对照原始值进行校验是一个很好的编程习惯。在应用中，如果某些位的写次数接近规定极限值，就应该进行写校验。

7.6 代码保护期间的操作

数据 EEPROM 存储器在配置字中有它自己的代码保护位。如果代码保护机制被使能，外部读写操作就被禁止。

单片机本身可以读写内部数据 EEPROM，与代码保护配置位的状态无关。更多信息，请参见第 23.0 节“CPU 的特殊功能”。

7.7 防止误写操作的保护措施

有些情况下，用户并不希望写入数据 EEPROM 存储器。为了防止 EEPROM 误写操作，器件实现了各种保护机制。上电时，WREN 位被清零。而且，上电延时期间（TPWRT，参数 33）也会阻止对 EEPROM 进行写操作。

在欠压、电源故障或软件故障期间，写操作的启动序列以及 WREN 位可共同防止意外写操作的发生。

7.8 使用数据 EEPROM

数据 EEPROM 是高耐用性可字节寻址的阵列，已将其优化以便存储频繁变动的信息（例如，程序变量或其他经常更新的数据）。如果一个段中的变量经常发生改变，而另一个段中的变量不发生改变，就可能造成超出对 EEPROM 的总写次数（规范 D124），而不超出对某个字节的总写次数（规范 D120）。如果这样，必须执行一次阵列刷新。出于此原因，不经常更新的变量（如常量、ID 和校准值等）应存放在闪存程序存储器中。

简单的数据 EEPROM 刷新程序如例 7-3 中所示。

注： 如果数据 EEPROM 仅用于存储常量和/或很少改变的数据，没有必要执行阵列刷新。请参见规范。

例 7-3： 数据 EEPROM 刷新程序

```
CLRF    EEADR          ; Start at address 0
BCF     EECON1, CFGS      ; Set for memory
BCF     EECON1, EEPGD      ; Set for Data EEPROM
BCF     INTCON, GIE        ; Disable interrupts
BSF     EECON1, WREN        ; Enable writes
Loop
  BSF    EECON1, RD        ; Loop to refresh array
  MOVLW  55h
  MOVWF  EECON2            ; Read current address
  MOVWF  0AAh
  MOVWF  EECON2            ; Write 55h
  BSF    EECON1, WR        ; Write 0AAh
  BTFSC  EECON1, WR        ; Set WR bit to begin write
  BRA    $-2
  INCFSZ EEADR, F          ; Wait for write to complete
  BRA    LOOP              ; Increment address
                           ; Not zero, do it again

  BCF    EECON1, WREN        ; Disable writes
  BSF    INTCON, GIE        ; Enable interrupts
```

表 7-1：与数据 EEPROM 存储器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
EEADR	EEADR7	EEADR6	EEADR5	EEADR4	EEADR3	EEADR2	EEADR1	EEADR0	61
EEADRH ⁽¹⁾	—	—	—	—	—	—	EEADR9	EEADR8	61
EEDATA	EEPROM 数据寄存器								61
EECON2	EEPROM 控制寄存器 2 (不是实际存在的寄存器)								61
EECON1	EEPGD	CFGs	—	FREE	WRERR	WREN	WR	RD	61
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62

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

注 1：仅限 PIC18F26K20/PIC18F46K20。

PIC18F2XK20/4XK20

注:

8.0 8 x 8 硬件乘法器

8.1 简介

所有 PIC18 器件均包含一个 8 x 8 硬件乘法器（是 ALU 的一部分）。该乘法器可执行无符号运算并产生一个 16 位运算结果，该结果存储在一对乘积寄存器 PRODH:PRODL 中。该乘法器执行的运算不会影响 STATUS 寄存器中的任何标志。

通过硬件执行乘法运算只需要一个指令周期。硬件乘法器具有更高的计算吞吐量并缩短了乘法算法的代码长度，从而可在许多先前仅能使用数字信号处理器的应用中使用 PIC18 器件。表 8-1 给出了硬件和软件乘法运算的比较，包括所需存储空间和执行时间。

8.2 工作原理

例 8-1 给出了一个 8 x 8 无符号乘法运算的指令序列。当已在 WREG 寄存器中装入了一个参数时，实现该运算仅需一条指令。

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

例 8-1: 8 x 8 无符号乘法程序

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

例 8-2: 8 x 8 有符号乘法程序

```
MOVF ARG1, W      ;  
MULWF ARG2        ; ARG1 * ARG2 ->  
                  ; PRODH:PRODL  
BTFSC ARG2, SB    ; Test Sign Bit  
SUBWF PRODH, F    ; PRODH = PRODH  
                  ; - ARG1  
MOVF ARG2, W      ;  
BTFSC ARG1, SB    ; Test Sign Bit  
SUBWF PRODH, F    ; PRODH = PRODH  
                  ; - ARG2
```

表 8-1：各种乘法运算的性能比较

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

例 8-3 给出了一个 16×16 无符号乘法运算的指令序列。公式 8-1 为所使用的算法。32 位结果存储在 4 个寄存器 ($\text{RES}_{<3:0>}$) 中。

公式 8-1: 16×16 无符号乘法算法

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

例 8-3: 16×16 无符号乘法程序

```

MOVF ARG1L, W
MULWF ARG2L      ; ARG1L * ARG2L->
                  ; PRODH:PRODL
MOVFF PRODH, RES1 ;
MOVFF PRODL, RES0 ;
;

MOVF ARG1H, W
MULWF ARG2H      ; ARG1H * ARG2H->
                  ; PRODH:PRODL
MOVFF PRODH, RES3 ;
MOVFF PRODL, RES2 ;
;

MOVF ARG1L, W
MULWF ARG2H      ; ARG1L * ARG2H->
                  ; PRODH:PRODL
MOVF PRODL, W    ;
ADDWF RES1, F    ; Add cross
MOVF PRODH, W    ; products
ADDWFC RES2, F   ;
CLRF WREG        ;
ADDWFC RES3, F   ;
;

MOVF ARG1H, W
MULWF ARG2L      ; ARG1H * ARG2L->
                  ; PRODH:PRODL
MOVF PRODL, W    ;
ADDWF RES1, F    ; Add cross
MOVF PRODH, W    ; products
ADDWFC RES2, F   ;
CLRF WREG        ;
ADDWFC RES3, F   ;
;
```

例 8-4 给出了一个 16×16 有符号乘法运算的指令序列。公式 8-2 为所使用的算法。32 位结果存储在 4 个寄存器 ($\text{RES}_{<3:0>}$) 中。要弄清参数的符号位，必须检查每个参数对的 MSb，并做相应的减法。

公式 8-2: 16×16 有符号乘法算法

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

例 8-4: 16×16 有符号乘法程序

```

MOVF ARG1L, W
MULWF ARG2L      ; ARG1L * ARG2L ->
                  ; PRODH:PRODL
MOVFF PRODH, RES1 ;
MOVFF PRODL, RES0 ;
;

MOVF ARG1H, W
MULWF ARG2H      ; ARG1H * ARG2H ->
                  ; PRODH:PRODL
MOVFF PRODH, RES3 ;
MOVFF PRODL, RES2 ;
;

MOVF ARG1L, W
MULWF ARG2H      ; ARG1L * ARG2H ->
                  ; PRODH:PRODL
MOVF PRODL, W    ;
ADDWF RES1, F    ; Add cross
MOVF PRODH, W    ; products
ADDWFC RES2, F   ;
CLRF WREG        ;
ADDWFC RES3, F   ;
;

MOVF ARG1H, W
MULWF ARG2L      ; ARG1H * ARG2L ->
                  ; PRODH:PRODL
MOVF PRODL, W    ;
ADDWF RES1, F    ; Add cross
MOVF PRODH, W    ; products
ADDWFC RES2, F   ;
CLRF WREG        ;
ADDWFC RES3, F   ;
;

BTFS ARG2H, 7    ; ARG2H:ARG2L neg?
BRA SIGN_ARG1    ; no, check ARG1
MOVF ARG1L, W    ;
SUBWF RES2        ;
MOVF ARG1H, W    ;
SUBWFB RES3        ;
;

SIGN_ARG1
BTFS ARG1H, 7    ; ARG1H:ARG1L neg?
BRA CONT_CODE    ; no, done
MOVF ARG2L, W    ;
SUBWF RES2        ;
MOVF ARG2H, W    ;
SUBWFB RES3        ;
;

CONT_CODE
:
```

9.0 中断

PIC18F2XK20/4XK20 器件具有多个中断源及一个中断优先级功能，该功能可以给大多数中断源分配高优先级或者低优先级。高优先级中断向量位于 0008h，低优先级中断向量位于 0018h。高优先级中断事件可以中断正在处理的低优先级中断。

有 10 个寄存器用于控制中断操作。这些寄存器是：

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1 和 PIR2
- PIE1 和 PIE2
- IPR1 和 IPR2

建议使用 MPLAB® IDE 提供的 Microchip 头文件命名这些寄存器中的位。这使得汇编器 / 编译器能够自动存放指定寄存器中的这些位。

通常，中断源有 3 个位用于控制其操作。这些位分别是：

- **标志位** 表明发生了中断事件
- **允许位** 允许程序跳转到中断向量地址处执行（当标志位置 1 时）
- **优先级位** 用于选择高优先级还是低优先级

9.1 与中档器件的兼容性

当 IPEN 位清零（默认状态）时，便会禁止中断优先级功能，此时中断是与 PIC® 单片机中档器件兼容的。在兼容模式下，IPRx 寄存器的中断优先级位不起作用。INTCON 寄存器的 PEIE 位是外设的全局中断允许位。PEIE 位只能禁止外设中断源，在 GIE 位也置 1 时可以允许外设中断源。INTCON 寄存器的 GIE 位是全局中断允许位，它可以允许所有非外设中断源，可以禁止包括外设在内的所有中断源。在兼容模式下，所有中断均跳转到地址 0008h。

9.2 中断优先级

通过将 RCON 寄存器的 IPEN 位置 1，可使能中断优先级功能。当使能中断优先级时，兼容模式的 GIE 和 PEIE 全局中断允许位被 GIEH（高优先级）和 GIEL（低优先级）全局中断允许位替代。当 INTCON 寄存器的 GIEH 位置 1 时，可以允许所有相应的 IPRx 寄存器或 INTCONx 寄存器优先级位置 1（高优先级）的中断。当清零时，GIEH 位可以禁止包括低优先级中断源在内的所有中断源。INTCON 寄存器的 GIEL 位在清零时，只能禁止相对应的优先级位清零（低优先级）的中断。当置 1 时，GIEL 位可以在 GIEH 位也置 1 时允许低优先级中断源。

当中断标志位、允许位及相应的全局中断允许位均被置 1 时，中断将根据中断源的优先级位设置的级别立即跳转到地址 0008h（高优先级）或 0018h（低优先级）。也可以通过设置相应的中断允许位来禁止单个中断。

9.3 中断响应

当响应中断时，全局中断允许位被清零以禁止后续中断。当 IPEN 位清零时，GIE 位是全局中断允许位。当 IPEN 位置 1 时（即使能中断优先级），GIEH 位是高优先级全局中断允许位，GIEL 位是低优先级全局中断允许位。高优先级中断源会中断低优先级中断。在处理高优先级中断时，低优先级中断将不被处理。

返回地址被压入堆栈，中断向量地址（0008h 或 0018h）被装入 PC。只要在中断服务程序中，就可以通过查询 INTCONx 和 PIRx 寄存器中的中断标志位来确定中断源。在重新允许中断前，必须用软件将中断标志位清零，以避免重复响应同一中断。

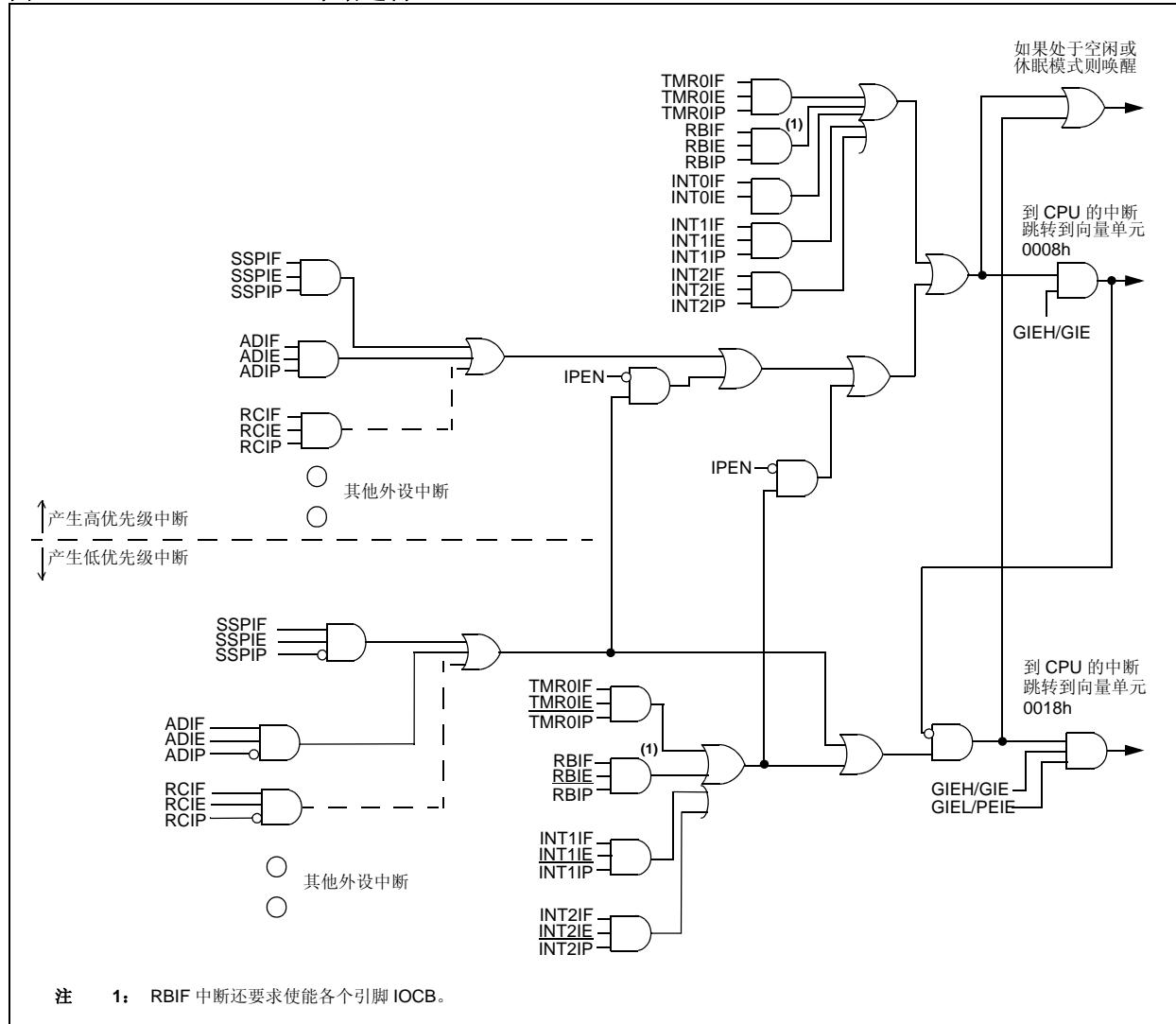
执行“从中断返回”指令 RETFIE 将退出中断服务程序，同时将 GIE 位（若使用中断优先级则为 GIEH 或 GIEL 位）置 1，从而重新允许中断。

对于外部中断事件，例如 INT 引脚中断或者 PORTB 电平变化中断，中断响应延时将会是 3 到 4 个指令周期。对于单周期或双周期指令，中断响应延时完全相同。各中断标志位的置 1 不受对应的中断允许位和全局中断允许位状态的影响。

注： 当允许任何中断时，不要使用 MOVFF 指令修改任何中断控制寄存器。否则可能导致单片机操作出错。

PIC18F2XK20/4XK20

图 9-1: PIC18 中断逻辑



9.4 INTCON 寄存器

INTCON 寄存器是可读写的寄存器，包含各个中断允许位、优先级位和标志位。

注：当中断条件产生时，不管相应的中断允许位或全局中断允许位的状态如何，中断标志位都将置 1。用户软件应在允许一个中断前，先将相应的中断标志位清零。这样做允许用软件查询中断标志位。

寄存器 9-1： INTCON： 中断控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMROIE	INT0IE	RBIE	TMROIF	INT0IF	RBIF
bit 7							

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	GIE/GIEH: 全局中断允许位 <u>当 IPEN = 0 时：</u> 1 = 允许所有未被屏蔽的中断 0 = 禁止所有中断（包括外设中断） <u>当 IPEN = 1 时：</u> 1 = 允许所有高优先级中断 0 = 禁止所有中断（包括低优先级中断）
bit 6	PEIE/GIEL: 外设中断允许位 <u>当 IPEN = 0 时：</u> 1 = 允许所有未被屏蔽的外设中断 0 = 禁止所有外设中断 <u>当 IPEN = 1 时：</u> 1 = 允许所有低优先级的中断 0 = 禁止所有低优先级的中断
bit 5	TMROIE: TMRO 溢出中断允许位 1 = 允许 TMRO 溢出中断 0 = 禁止 TMRO 溢出中断
bit 4	INT0IE: INT0 外部中断允许位 1 = 允许 INT0 外部中断 0 = 禁止 INT0 外部中断
bit 3	RBIE: RB 端口电平变化中断允许位 (2) 1 = 允许 RB 端口电平变化中断 0 = 禁止 RB 端口电平变化中断
bit 2	TMROIF: TMRO 溢出中断标志位 1 = TMRO 寄存器已溢出（必须用软件清零） 0 = TMRO 寄存器未溢出
bit 1	INT0IF: INT0 外部中断标志位 1 = 发生了 INT0 外部中断（必须用软件清零） 0 = 未发生 INT0 外部中断
bit 0	RBIF: RB 端口电平变化中断标志位 (1) 1 = RB<7:4> 引脚中至少有一个引脚的电平状态发生了改变（必须用软件清零） 0 = 所有 RB<7:4> 引脚的电平状态都没有改变

- 注**
- 1: 不匹配条件将继续把 RBIF 位置 1。读取 PORTB 将结束不匹配条件，并允许将该位清零。
 - 2: RB 端口电平变化中断还要求使能各个引脚 IOCB。

PIC18F2XK20/4XK20

寄存器 9-2: INTCON2: 中断控制寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	U-0	R/W-1	U-0	R/W-1
RBPU	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP
bit 7	bit 0						

图注:

R = 可读位

-n = POR 时的值

W = 可写位

1 = 置 1

U = 未实现位, 读为 0

0 = 清零

x = 未知

bit 7 **RBPU:** PORTB 上拉使能位

1 = 禁止所有 PORTB 上拉

0 = 如果引脚是输入引脚, 且对应的 WPUB 位置 1, 则使能 PORTB 上拉

bit 6 **INTEDG0:** 外部中断 0 边沿选择位

1 = 上升沿触发中断

0 = 下降沿触发中断

bit 5 **INTEDG1:** 外部中断 1 边沿选择位

1 = 上升沿触发中断

0 = 下降沿触发中断

bit 4 **INTEDG2:** 外部中断 2 边沿选择位

1 = 上升沿触发中断

0 = 下降沿触发中断

bit 3 未实现: 读为 0

bit 2 **TMR0IP:** TMR0 溢出中断优先级位

1 = 高优先级

0 = 低优先级

bit 1 未实现: 读为 0

bit 0 **RBIP:** RB 端口电平变化中断优先级位

1 = 高优先级

0 = 低优先级

注: 当中断条件产生时, 不管相应的中断允许位或全局中断允许位的状态如何, 中断标志位都将置 1。用户软件应在允许一个中断前, 先将相应的中断标志位清零。这样做允许用软件查询中断标志位。

寄存器 9-3: INTCON3: 中断控制寄存器 3

R/W-1	R/W-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF
bit 7	bit 0						

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **INT2IP: INT2 外部中断优先级位**

1 = 高优先级

0 = 低优先级

bit 6 **INT1IP: INT1 外部中断优先级位**

1 = 高优先级

0 = 低优先级

bit 5 未实现: 读为 0

bit 4 **INT2IE: INT2 外部中断允许位**

1 = 允许 INT2 外部中断

0 = 禁止 INT2 外部中断

bit 3 **INT1IE: INT1 外部中断允许位**

1 = 允许 INT1 外部中断

0 = 禁止 INT1 外部中断

bit 2 未实现: 读为 0

bit 1 **INT2IF: INT2 外部中断标志位**

1 = 发生了 INT2 外部中断 (必须用软件清零)

0 = 未发生 INT2 外部中断

bit 0 **INT1IF: INT1 外部中断标志位**

1 = 发生了 INT1 外部中断 (必须用软件清零)

0 = 未发生 INT1 外部中断

注: 当中断条件产生时, 不管相应的中断允许位或全局中断允许位的状态如何, 中断标志位都将置 1。用户软件应在允许一个中断前, 先将相应的中断标志位清零。这样做允许用软件查询中断标志位。

PIC18F2XK20/4XK20

9.5 PIR 寄存器

PIR 寄存器包含各外设中断的标志位。根据外设中断源的数量，有两个外设中断请求标志寄存器（PIR1 和 PIR2）。

- 注 1:** 当中断条件产生时，不管相应的中断允许位或全局中断允许位 GIE（在 INTCON 寄存器中）的状态如何，中断标志位都将置 1。
- 2:** 用户软件应在允许中断前和处理完中断后，将相应的中断标志位清零。

寄存器 9-4: PIR1: 外设中断请求（标志）寄存器 1

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **PSPIF:** 并行从端口读 / 写中断标志位⁽¹⁾

1 = 发生了读或写操作（必须用软件清零）

0 = 未发生读或写操作

bit 6 **ADIF:** A/D 转换器中断标志位

1 = 一次 A/D 转换已完成（必须用软件清零）

0 = A/D 转换未完成或尚未开始

bit 5 **RCIF:** EUSART 接收中断标志位

1 = EUSART 接收缓冲区 RCREG 已满（读取 RCREG 时清零）

0 = EUSART 接收缓冲区为空

bit 4 **TXIF:** EUSART 发送中断标志位

1 = EUSART 发送缓冲区 TXREG 为空（写入 TXREG 时清零）

0 = EUSART 发送缓冲区已满

bit 3 **SSPIF:** 主同步串口中断标志位

1 = 发送 / 接收已完成（必须用软件清零）

0 = 等待发送 / 接收

bit 2 **CCP1IF:** CCP1 中断标志位

捕捉模式:

1 = 发生了 TMR1 寄存器捕捉（必须用软件清零）

0 = 未发生 TMR1 寄存器捕捉

比较模式:

1 = 发生了 TMR1 寄存器的比较匹配（必须用软件清零）

0 = 未发生 TMR1 寄存器的比较匹配

PWM 模式:

在此模式下未使用

bit 1 **TMR2IF:** TMR2 与 PR2 匹配中断标志位

1 = TMR2 与 PR2 发生匹配（必须用软件清零）

0 = TMR2 与 PR2 未发生匹配

bit 0 **TMR1IF:** TMR1 溢出中断标志位

1 = TMR1 寄存器已溢出（必须用软件清零）

0 = TMR1 寄存器未溢出

注 1: PSPIF 位在 28 引脚器件上未实现，读为 0。

寄存器 9-5: PIR2: 外设中断请求 (标志) 寄存器 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

- | | |
|-------|---|
| bit 7 | OSCFIF: 振荡器故障中断标志位
1 = 器件振荡器发生故障, 改由 HFINTOSC 作为时钟输入 (必须用软件清零)
0 = 器件时钟正常工作 |
| bit 6 | C1IF: 比较器 C1 中断标志位
1 = 比较器 C1 输出已改变 (必须用软件清零)
0 = 比较器 C1 输出未改变 |
| bit 5 | C2IF: 比较器 C2 中断标志位
1 = 比较器 C2 输出已改变 (必须用软件清零)
0 = 比较器 C2 输出未改变 |
| bit 4 | EEIF: 数据 EEPROM/ 闪存写操作中断标志位
1 = 写操作完成 (必须用软件清零)
0 = 写操作未完成或尚未开始 |
| bit 3 | BCLIF: 总线冲突中断标志位
1 = 发生了总线冲突 (必须用软件清零)
0 = 未发生总线冲突 |
| bit 2 | HLVDIF: 低压检测中断标志位
1 = 发生了低压条件 (方向由 HLVDCON 寄存器的 VDIRMAG 位确定)
0 = 未发生低压条件 |
| bit 1 | TMR3IF: TMR3 溢出中断标志位
1 = TMR3 寄存器已溢出 (必须用软件清零)
0 = TMR3 寄存器未溢出 |
| bit 0 | CCP2IF: CCP2 中断标志位
<u>捕捉模式:</u>
1 = 发生了 TMR1 寄存器捕捉 (必须用软件清零)
0 = 未发生 TMR1 寄存器捕捉
<u>比较模式:</u>
1 = 发生了 TMR1 寄存器的比较匹配 (必须用软件清零)
0 = 未发生 TMR1 寄存器的比较匹配
<u>PWM 模式:</u>
在此模式下未使用 |

PIC18F2XK20/4XK20

9.6 PIE 寄存器

PIE 寄存器包含各外设中断的允许位。根据外设中断源的数量，有两个外设中断允许寄存器（PIE1 和 PIE2）。当 IPEN = 0 时，要允许任一外设中断，必须将 PEIE 位置 1。

寄存器 9-6： PIE1：外设中断允许（标志）寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **PSPIE**: 并行从端口读 / 写中断允许位⁽¹⁾

1 = 允许 PSP 读 / 写中断

0 = 禁止 PSP 读 / 写中断

bit 6 **ADIE**: A/D 转换器中断允许位

1 = 允许 A/D 中断

0 = 禁止 A/D 中断

bit 5 **RCIE**: EUSART 接收中断允许位

1 = 允许 EUSART 接收中断

0 = 禁止 EUSART 接收中断

bit 4 **TXIE**: EUSART 发送中断允许位

1 = 允许 EUSART 发送中断

0 = 禁止 EUSART 发送中断

bit 3 **SSPIE**: 主同步串口中断允许位

1 = 允许 MSSP 中断

0 = 禁止 MSSP 中断

bit 2 **CCP1IE**: CCP1 中断允许位

1 = 允许 CCP1 中断

0 = 禁止 CCP1 中断

bit 1 **TMR2IE**: TMR2 与 PR2 匹配中断允许位

1 = 允许 TMR2 与 PR2 匹配中断

0 = 禁止 TMR2 与 PR2 匹配中断

bit 0 **TMR1IE**: TMR1 溢出中断允许位

1 = 允许 TMR1 溢出中断

0 = 禁止 TMR1 溢出中断

注 1: PSPIE 位在 28 引脚器件上未实现，读为 0。

寄存器 9-7: PIE2: 外设中断允许 (标志) 寄存器 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **OSCFIE:** 振荡器故障中断允许位

1 = 允许

0 = 禁止

bit 6 **C1IE:** 比较器 C1 中断允许位

1 = 允许

0 = 禁止

bit 5 **C2IE:** 比较器 C2 中断允许位

1 = 允许

0 = 禁止

bit 4 **EEIE:** 数据 EEPROM/ 闪存写操作中断允许位

1 = 允许

0 = 禁止

bit 3 **BCLIE:** 总线冲突中断允许位

1 = 允许

0 = 禁止

bit 2 **HLVDIE:** 低压检测中断允许位

1 = 允许

0 = 禁止

bit 1 **TMR3IE:** TMR3 溢出中断允许位

1 = 允许

0 = 禁止

bit 0 **CCP2IE:** CCP2 中断允许位

1 = 允许

0 = 禁止

PIC18F2XK20/4XK20

9.7 IPR 寄存器

IPR 寄存器包含各外设中断的优先级位。根据外设中断源的数量，有两个外设中断优先级寄存器（IPR1 和 IPR2）。使用优先级位时，要求将中断优先级使能（IPEN）位置 1。

寄存器 9-8： IPR1：外设中断优先级寄存器 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	
PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	
bit 7								bit 0

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **PSPIP:** 并行从端口读 / 写中断优先级位 ⁽¹⁾

1 = 高优先级

0 = 低优先级

bit 6 **ADIP:** A/D 转换器中断优先级位

1 = 高优先级

0 = 低优先级

bit 5 **RCIP:** EUSART 接收中断优先级位

1 = 高优先级

0 = 低优先级

bit 4 **TXIP:** EUSART 发送中断优先级位

1 = 高优先级

0 = 低优先级

bit 3 **SSPIP:** 主同步串口中断优先级位

1 = 高优先级

0 = 低优先级

bit 2 **CCP1IP:** CCP1 中断优先级位

1 = 高优先级

0 = 低优先级

bit 1 **TMR2IP:** TMR2 与 PR2 匹配中断优先级位

1 = 高优先级

0 = 低优先级

bit 0 **TMR1IP:** TMR1 溢出中断优先级位

1 = 高优先级

0 = 低优先级

注 1：PSPIF 位在 28 引脚器件上未实现，读为 0。

寄存器 9-9: IPR2: 外设中断优先级寄存器 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **OSCFIP:** 振荡器故障中断优先级位

1 = 高优先级

0 = 低优先级

bit 6 **C1IP:** 比较器 C1 中断优先级位

1 = 高优先级

0 = 低优先级

bit 5 **C2IP:** 比较器 C2 中断优先级位

1 = 高优先级

0 = 低优先级

bit 4 **EEIP:** 数据 EEPROM/ 闪存写操作中断优先级位

1 = 高优先级

0 = 低优先级

bit 3 **BCLIP:** 总线冲突中断优先级位

1 = 高优先级

0 = 低优先级

bit 2 **HLVDIP:** 低压检测中断优先级位

1 = 高优先级

0 = 低优先级

bit 1 **TMR3IP:** TMR3 溢出中断优先级位

1 = 高优先级

0 = 低优先级

bit 0 **CCP2IP:** CCP2 中断优先级位

1 = 高优先级

0 = 低优先级

9.8 RCON 寄存器

RCON寄存器中包含的标志位可用来确定器件上次复位或者从空闲模式或休眠模式唤醒的原因。RCON 还包含一个可使能中断优先级的 IPEN 位。

SBOREN 位和复位标志位的操作已在第 4.1 节“RCON 寄存器”中详细讨论。

寄存器 9-10: RCON: 复位控制寄存器

R/W-0	R/W-1	U-0	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	SBOREN ⁽¹⁾	—	<u>RI</u>	<u>TO</u>	<u>PD</u>	<u>POR⁽¹⁾</u>	<u>BOR</u>
bit 7	bit 0						

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **IPEN:** 中断优先级使能位

1 = 使能中断优先级

0 = 禁止中断优先级 (中档器件兼容模式)

bit 6 **SBOREN:** 软件 BOR 使能位⁽¹⁾

位操作的详细信息, 请参见寄存器 4-1。

bit 5 **未实现:** 读为 0

bit 4 **RI:** RESET 指令标志位

位操作的详细信息, 请参见寄存器 4-1。

bit 3 **TO:** 看门狗超时标志位

位操作的详细信息, 请参见寄存器 4-1。

bit 2 **PD:** 掉电检测标志位

位操作的详细信息, 请参见寄存器 4-1。

bit 1 **POR:** 上电复位状态位

位操作的详细信息, 请参见寄存器 4-1。

bit 0 **BOR:** 欠压复位状态位

位操作的详细信息, 请参见寄存器 4-1。

注 1: 实际复位值由器件配置和器件复位的特性决定。更多信息, 请参见寄存器 4-1。

9.9 INTn 引脚中断

RB0/INT0、RB1/INT1 和 RB2/INT2 引脚上的外部中断都是边沿触发的。如果 INTCON2 寄存器中相应的 INTEDG_x 位被置 1 (= 1)，则为上升沿触发；如果该位被清零，则为下降沿触发。当 RB_x/INT_x 引脚上出现一个有效边沿时，相应的标志位 INTxF 被置 1。通过清零相应的中断允许位 INTxE，可禁止该中断。在重新允许该中断前，必须在中断服务程序中先用软件将中断标志位 INTxF 清零。

如果 INTxE 位在进入空闲或休眠模式前被置 1，则所有的外部中断 (INT0、INT2 和 INT2) 均能将处理器从空闲或休眠模式唤醒。如果全局中断允许位 GIE 被置 1，则处理器将在被唤醒之后跳转到中断向量处执行程序。

INT1 和 INT2 的中断优先级由 INTCON3 寄存器的中断优先级位 INT1IP 和 INT2IP 的值决定。没有与 INT0 相关的优先级位。INT0 始终是一个高优先级的中断源。

9.10 TMR0 中断

在 8 位模式（默认模式）下，TMR0 寄存器的溢出 (FFh → 00h) 会将 TMR0IF 标志位置 1。在 16 位模式下，TMR0H:TMR0L 寄存器对的溢出 (FFFFh → 0000h) 会将 TMR0IF 标志位置 1。可以通过置 1/清零 INTCON 寄存器中的中断允许位 TMR0IE 来允许 / 禁止该中断。Timer0 的中断优先级由 INTCON2 寄存器的中断优先级位 TMR0IP 的值决定。欲进一步了解 Timer0 模块的详细信息，请参见第 12.0 节“Timer0 模块”。

9.11 PORTB 电平变化中断

PORTB<7:4> 上的输入电平变化会将 INTCON 寄存器的标志位 RBIF 置 1。可以通过置 1/清零 INTCON 寄存器中的中断允许位 RBIE 来允许 / 禁止该中断。各个引脚还必须使用 IOCB 寄存器分别使能。PORTB 电平变化中断的优先级由 INTCON2 寄存器的中断优先级位 RBIP 中的值决定。

9.12 中断现场保护

在中断期间，PC 的返回地址被保存在堆栈中。另外，WREG、STATUS 和 BSR 寄存器的值被压入快速返回堆栈。如果未使用从中断快速返回功能（见第 5.1.3 节“快速寄存器堆栈”），那么用户可能需要在进入中断服务程序前，保存 WREG、STATUS 和 BSR 寄存器的值。根据用户的具体应用，还可能需要保存其他寄存器的值。例 9-1 在执行中断服务程序期间，保存并恢复 WREG、STATUS 和 BSR 寄存器的值。

例 9-1：将 STATUS、WREG 和 BSR 寄存器的值保存在 RAM 中

```
MOVWF    W_TEMP           ; W_TEMP is in virtual bank
MOVFF    STATUS, STATUS_TEMP ; STATUS_TEMP located anywhere
MOVFF    BSR, BSR_TEMP     ; BSR_TEMP located anywhere
;
; USER ISR CODE
;
MOVFF    BSR_TEMP, BSR      ; Restore BSR
MOVF     W_TEMP, W          ; Restore WREG
MOVFF    STATUS_TEMP, STATUS ; Restore STATUS
```

PIC18F2XK20/4XK20

注:

10.0 I/O 端口

根据选定的器件和使能的功能，最多有 5 个端口可供使用。I/O 端口的一些引脚与器件上外设功能复用。通常而言，当某个外设使能时，其相关引脚可能不能用作通用 I/O 引脚。

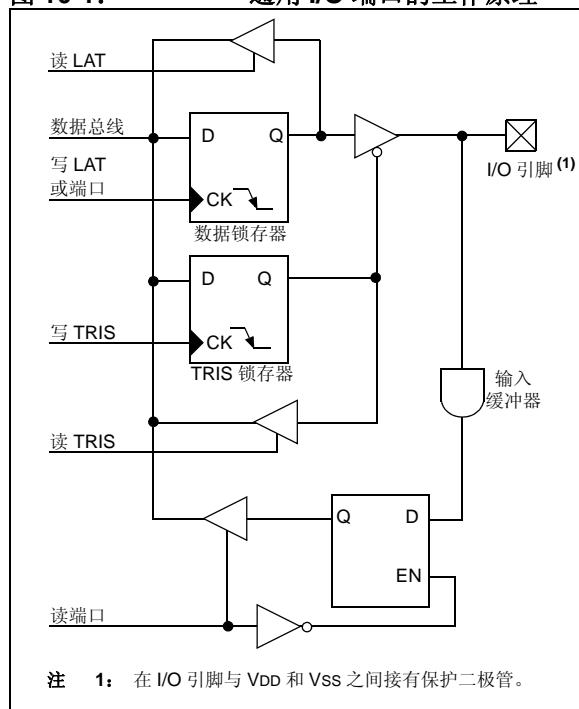
每个端口都有三个工作寄存器。这些寄存器是：

- TRIS 寄存器（数据方向寄存器）
- 端口寄存器（读取器件引脚的电平）
- LAT 寄存器（输出锁存器）

在对 I/O 引脚驱动值进行读-修改-写操作时会用到数据锁存器（LAT 寄存器）。

图 10-1 给出了通用 I/O 端口的简化模型，没有给出与其他外设的接口。

图 10-1：通用 I/O 端口的工作原理



10.1 PORTA、TRISA 和 LATA 寄存器

PORTA 是一个 8 位宽的双向端口，对应的数据方向寄存器是 TRISA。将 TRISA 某位置 1 (= 1) 时，会将 PORTA 的相应引脚设为输入（即，禁止输出驱动器）。将 TRISA 某位清零 (= 0) 时，会将 PORTA 的相应引脚设为输出（即，使能输出驱动器并将输出锁存器中的内容输出到选中引脚）。

读 PORTA 寄存器将读出相应引脚的状态，而对其进行写操作则是将数据写入端口锁存器。

数据锁存寄存器（LATA）也是存储器映射的。对 LATA 寄存器执行读-修改-写操作将读写 PORTA 的锁存输出值。

RA4 引脚与 Timer0 模块的时钟输入以及比较器输出之一复用，成为 RA4/T0CKI/C1OUT 引脚。RA6 和 RA7 引脚与主振荡器引脚复用，通过在配置寄存器（详情请参见第 23.1 节“配置位”）中对主振荡器进行配置可将这两个引脚使能为振荡器或 I/O 引脚。当没被用作端口引脚时，RA6 和 RA7 及其相关的 TRIS 和 LAT 位均读为 0。

其他 PORTA 引脚与模拟输入、模拟 VREF+、VREF- 输入和比较器参考电压输出复用。通过设置 ANSEL 寄存器中的 ANS<4:0> 位，可以选择引脚 RA<3:0> 和 RA5 作为模拟引脚工作；这是上电复位后的默认设置。

通过设置 CM1CON0 和 CM2CON0 寄存器中相应的位还可以将 RA0 到 RA5 引脚用作比较器输入或输出。

注： 上电复位时，RA5 和 RA<3:0> 被配置为模拟输入并读为 0。RA4 则被配置为数字输入。

RA4/T0CKI/C1OUT 引脚是施密特触发器输入引脚。所有其他 PORTA 引脚都是 TTL 输入电平和全 CMOS 输出驱动器。

TRISA 寄存器控制着 PORTA 引脚的方向，即使它们被用作模拟输入。当引脚用于模拟输入时，用户应确保 TRISA 寄存器中的各位保持置 1。

例 10-1：初始化 PORTA

```

CLRF    PORTA    ; Initialize PORTA by
                  ; clearing output
                  ; data latches
CLRF    LATA     ; Alternate method
                  ; to clear output
                  ; data latches
MOVLW  E0h      ; Configure I/O
MOVWF   ANSEL   ; for digital inputs
MOVLW  0CFh    ; Value used to
                  ; initialize data
                  ; direction
MOVWF   TRISA   ; Set RA<3:0> as inputs
                  ; RA<5:4> as outputs

```

PIC18F2XK20/4XK20

表 10-1: PORTA I/O 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RA0/AN0/C12IN0-	RA0	0	O	DIG	LATA<0> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<0> 数据输入；当使能模拟输入时被禁止。
	AN0	1	I	ANA	A/D 输入通道 0。POR 时的默认输入配置；不影响数字输出。
	C12IN0-	1	I	ANA	比较器 C1 和 C2 的反相输入，通道 0。模拟选择与 ADC 共用。
RA1/AN1/C12IN1-	RA1	0	O	DIG	LATA<1> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<1> 数据输入；当使能模拟输入时被禁止。
	AN1	1	I	ANA	A/D 输入通道 1。POR 时的默认输入配置；不影响数字输出。
	C12IN1-	1	I	ANA	比较器 C1 和 C2 的反相输入，通道 1。模拟选择与 ADC 共用。
RA2/AN2/C2IN+ VREF-/CVREF	RA2	0	O	DIG	LATA<2> 数据输出；不受模拟输入影响。当使能 CVREF 输出时被禁止。
		1	I	TTL	PORTA<2> 数据输入。当使能模拟功能时被禁止；当使能 CVREF 输出时被禁止。
	AN2	1	I	ANA	A/D 输入通道 2。POR 时的默认输入配置；不受模拟输出影响。
	C2IN+	1	I	ANA	比较器 C2 的同相输入。模拟选择与 ADC 共用。
	VREF-	1	I	ANA	ADC 和比较器低参考电压输入。
	CVREF	x	O	ANA	比较器参考电压输出。使能该功能将禁止数字 I/O。
RA3/AN3/C1IN+/ VREF+	RA3	0	O	DIG	LATA<3> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<3> 数据输入；当使能模拟输入时被禁止。
	AN3	1	I	ANA	A/D 输入通道 3。POR 时的默认输入配置。
	C1IN+	1	I	ANA	比较器 C1 的同相输入。模拟选择与 ADC 共用。
	VREF+	1	I	ANA	ADC 和比较器高参考电压输入。
RA4/T0CKI/C1OUT	RA4	0	O	DIG	LATA<4> 数据输出。
		1	I	ST	PORTA<4> 数据输入；POR 时的默认配置。
	T0CKI	1	I	ST	Timer0 的时钟输入。
	C1OUT	0	O	DIG	比较器 1 的输出；优先于端口数据。
RA5/AN4/SS/ HLVDIN/C2OUT	RA5	0	O	DIG	LATA<5> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTA<5> 数据输入；当使能模拟输入时被禁止。
	AN4	1	I	ANA	A/D 输入通道 4。POR 时的默认配置。
	SS	1	I	TTL	SSP 的从选择输入（MSSP 模块）。
	HLVDIN	1	I	ANA	低压检测外部跳变点输入。
	C2OUT	0	O	DIG	比较器 2 的输出；优先于端口数据。
OSC2/CLKOUT/ RA6	RA6	0	O	DIG	LATA<6> 数据输出。仅在 RCIO、INTIO2 和 ECIO 模式下使能。
		1	I	TTL	PORTA<6> 数据输入。仅在 RCIO、INTIO2 和 ECIO 模式下使能。
	OSC2	x	O	ANA	主振荡器反馈输出连接（XT、HS 和 LP 模式）。
	CLKOUT	x	O	DIG	RC、INTIO1 和 EC 振荡器模式下的系统周期时钟输出（Fosc/4）。

图注: DIG = 数字电平输出; TTL = TTL 输入缓冲器; ST = 施密特触发器输入缓冲器; ANA = 模拟电平输入 / 输出;
x = 无关位 (TRIS 位不影响端口方向或在此可忽略)。

表 10-1: PORTA I/O 汇总 (续)

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
OSC1/CLKIN/RA7	RA7	0	O	DIG	LATA<7> 数据输出。在外部振荡器模式下被禁止。
		1	I	TTL	PORTA<7> 数据输入。在外部振荡器模式下被禁止。
	OSC1	x	I	ANA	主振荡器输入连接。
	CLKIN	x	I	ANA	主时钟输入连接。

图注: DIG = 数字电平输出; TTL = TTL 输入缓冲器; ST = 施密特触发器输入缓冲器; ANA = 模拟电平输入 / 输出;
x = 无关位 (TRIS 位不影响端口方向或在此可忽略)。

表 10-2: 与 PORTA 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值 所在页
PORTA	RA7 ⁽¹⁾	RA6 ⁽¹⁾	RA5	RA4	RA3	RA2	RA1	RA0	62
LATA	LATA7 ⁽¹⁾	LATA6 ⁽¹⁾	PORTA 数据锁存寄存器 (读和写数据锁存器)						62
TRISA	TRISA7 ⁽¹⁾	TRISA6 ⁽¹⁾	PORTA 数据方向控制寄存器						62
ANSEL	ANS7 ⁽²⁾	ANS6 ⁽²⁾	ANS5 ⁽²⁾	ANS4	ANS3	ANS2	ANS1	ANS0	62
SLRCON	—	—	—	SLRE ⁽²⁾	SLRD ⁽²⁾	SLRC	SLRB	SLRA	63
CM1CON0	C1ON	C1OUT	C1OE	C1POL	C1SP	C1R	C1CH1	C1CH0	62
CM2CON0	C2ON	C2OUT	C2OE	C2POL	C2SP	C2R	C2CH1	C2CH0	62
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	61

图注: — = 未实现, 读为 0。PORTA 不使用阴影单元。

注 1: RA<7:6> 及其相关的锁存器和数据方向位根据振荡器配置使能为 I/O 引脚; 否则, 它们将读为 0。

2: 在 PIC18F2XK20 器件上未实现。

10.2 PORTB、TRISB 和 LATB 寄存器

PORTB 是一个 8 位宽的双向端口，对应的数据方向寄存器是 TRISB。将 TRISB 某位置 1 (= 1) 时，会将 PORTB 的相应引脚设为输入（即，禁止输出驱动器）。将 TRISB 某位清零 (= 0) 时，会将 PORTB 的相应引脚设为输出（即，使能输出驱动器并将输出锁存器中的内容输出到选中引脚）。

数据锁存寄存器（LATB）也是存储器映射的。对 LATB 寄存器执行读 - 修改 - 写操作将读写 PORTB 的锁存输出值。

例 10-2： 初始化 PORTB

```
CLRF    PORTB    ; Initialize PORTB by
                  ; clearing output
                  ;data latches
CLRF    LATB     ; Alternate method
                  ; to clear output
                  ;data latches
CLRF    ANSELH   ; Set RB<4:0> as
                  ; digital I/O pins
                  ;(required if config bit
                  ; PBADEN is set)
MOVlw    0CFh    ; Value used to
                  ;initialize data
                  ;direction
MOVwf    TRISB   ; Set RB<3:0> as inputs
                  ; RB<5:4> as outputs
                  ; RB<7:6> as inputs
```

10.3 PORTB 引脚的其他功能

PORTB 的引脚 RB<7:4> 具有电平变化中断选项。所有的 PORTB 引脚都具有弱上拉选项。RB3 上提供备用的 CCP2 外设选项。

10.3.1 弱上拉

每个 PORTB 引脚都具有单独控制的内部弱上拉功能。当置 1 时，WPUB 寄存器的每个位可以使能相应的引脚上拉功能。当清零时，INTCON2 寄存器的 RBPU 位可以使能将对应的 WPUB 位置 1 的所有引脚上的上拉功能。当置 1 时，RBPU 位可以禁止所有弱上拉功能。当端口引脚被配置为输出时，其弱上拉功能会自动关闭。上电复位会禁止上拉功能。

注： 上电复位时，默认情况下 RB<4:0> 被配置为模拟输入且读为 0；RB<7:5> 则被配置为数字输入。
当 PBADEN 配置位设为 1 时，RB<4:0> 可以在 POR 时被配置为数字输入。

10.3.2 电平变化中断

PORTB 的 4 个引脚（RB<7:4>）被单独配置为具有电平变化中断功能的引脚。IOCB 寄存器中的控制位可以允许（置 1 时）或禁止（清零时）每个引脚的中断功能。

当置 1 时，INTCON 寄存器的 RBIE 位可以允许将对应的 IOCB 位置 1 的所有引脚上的中断。当清零时，RBIE 位可以禁止所有电平变化中断。

仅当将这些引脚配置为输入时，才可使用此中断功能（即当 RB<7:4> 中的任何一个引脚被配置为输出时，该引脚将不再具有电平变化中断功能）。

对于允许了电平变化中断功能的引脚，其值将与上次读取的 PORTB 的旧锁存值相比较。所有与上次读取值不匹配的输出进行逻辑或运算，运算结果用来设置 INTCON 寄存器中的 PORTB 电平变化中断标志位（RBIF）。

该中断可将器件从休眠模式或任何空闲模式唤醒。用户可用以下方式在中断服务程序中清除该中断：

- 通过读或写 PORTB 来清除不匹配条件（当 PORTB 是 MOVFF 指令的源或目标时除外）。
- 清零标志位 RBIF。

不匹配条件将继续把 RBIF 标志位置 1。读或写 PORTB 将结束不匹配条件并允许将 RBIF 位清零。保存上一次读取值的锁存器不受 MCLR 或欠压复位的影响。在这些复位之后，如果存在不匹配情况，RBIF 标志位还将继续被置 1。

注： 当读操作正在执行时发生了 I/O 引脚电平变化（Q2 周期的起始时刻），则 RBIF 中断标志位可能不会被置 1。此外，由于端口上的读或写操作会影响该端口的所有位，所以在电平变化中断模式下使用多个引脚时必须特别小心。在处理一个引脚上的电平变化时，可能不会注意到另一个引脚上的电平变化。

建议使用电平变化中断功能实现按键唤醒操作，以及那些仅用到 PORTB 的电平变化中断功能的操作。在使用电平变化中断功能时，建议不要查询 PORTB 的状态。

10.3.3 备用 CCP2 选项

RB3 可以配置为 CCP2 模块的备用外设引脚，方法是通过清零 CONFIG3H 的 CCP2MX 配置位。CCP2MX 配置位的默认状态是 1，该状态选择 RC1 作为 CCP2 外设引脚。

表 10-3: PORTB I/O 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RB0/INT0/FLT0/ AN12	RB0	0	O	DIG	LATB<0> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTB<0> 数据输入；可编程的弱上拉。当使能模拟输入时被禁止。 ⁽¹⁾
	INT0	1	I	ST	外部中断 0 输入。
	FLT0	1	I	ST	增强型 PWM 故障输入（ECCP1 模块）；通过软件使能。
	AN12	1	I	ANA	A/D 输入通道 12。 ⁽¹⁾
RB1/INT1/AN10/ C12IN3-/P1C	RB1	0	O	DIG	LATB<1> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTB<1> 数据输入；可编程的弱上拉。当使能模拟输入时被禁止。 ⁽¹⁾
	INT1	1	I	ST	外部中断 1 输入。
	AN10	1	I	ANA	A/D 输入通道 10。 ⁽¹⁾
	C12IN3-	1	I	ANA	比较器 C1 和 C2 的反相输入，通道 3。模拟选择与 ADC 共用。
	P1C	0	O	DIG	ECCP PWM 输出（仅限 28 引脚器件）。
RB2/INT2/AN8/ P1B	RB2	0	O	DIG	LATB<2> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTB<2> 数据输入；可编程的弱上拉。当使能模拟输入时被禁止。 ⁽¹⁾
	INT2	1	I	ST	外部中断 2 输入。
	AN8	1	I	ANA	A/D 输入通道 8。 ⁽¹⁾
	P1B	0	O	DIG	ECCP PWM 输出（仅限 28 引脚器件）。
RB3/AN9/C12IN2/- CCP2	RB3	0	O	DIG	LATB<3> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTB<3> 数据输入；可编程的弱上拉。当使能模拟输入时被禁止。 ⁽¹⁾
	AN9	1	I	ANA	A/D 输入通道 9。 ⁽¹⁾
	C12IN2-	1	I	ANA	比较器 C1 和 C2 的反相输入，通道 2。模拟选择与 ADC 共用。
	CCP2 ⁽²⁾	0	O	DIG	CCP2 比较输出和 PWM 输出。
		1	I	ST	CCP2 捕捉输入。
RB4/KBI0/AN11/ P1D	RB4	0	O	DIG	LATB<4> 数据输出；不受模拟输入影响。
		1	I	TTL	PORTB<4> 数据输入；可编程的弱上拉。当使能模拟输入时被禁止。 ⁽¹⁾
	KBI0	1	I	TTL	引脚电平变化中断。
	AN11	1	I	ANA	A/D 输入通道 11。 ⁽¹⁾
	P1D	0	O	DIG	ECCP PWM 输出（仅限 28 引脚器件）。
RB5/KBI1/PGM	RB5	0	O	DIG	LATB<5> 数据输出。
		1	I	TTL	PORTB<5> 数据输入；可编程的弱上拉。
	KBI1	1	I	TTL	引脚电平变化中断。
	PGM	x	I	ST	单电源供电编程模式进入（ICSP TM ）。由 LVP 配置位使能；所有其他引脚功能被禁止。

图注: DIG = 数字电平输出; TTL = TTL 输入缓冲器; ST = 施密特触发器输入缓冲器; ANA = 模拟电平输入 / 输出;
x = 无关位 (TRIS 位不影响端口方向或在此可忽略)。

注 1: POR 时的配置由 PBADEN 配置位决定。默认情况下, 当 PBADEN 置 1 时, 引脚被配置为模拟输入; 当 PBADEN 清零时, 则被配置为数字输入。

2: 当 CCP2MX 配置位为 0 时 CCP2 的备用设置。默认设置为 RC1。

3: 当使能 ICSP 或 ICD 时, 禁止所有其他引脚功能。

PIC18F2XK20/4XK20

表 10-3: PORTB I/O 汇总 (续)

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RB6/KBI2/PGC	RB6	0	O	DIG	LATB<6> 数据输出。
		1	I	TTL	PORTB<6> 数据输入; 可编程的弱上拉。
	KBI2	1	I	TTL	引脚电平变化中断。
	PGC	x	I	ST	供 ICSP 和 ICD 工作使用的串行执行 (ICSP) 时钟输入。 ⁽³⁾
RB7/KBI3/PGD	RB7	0	O	DIG	LATB<7> 数据输出。
		1	I	TTL	PORTB<7> 数据输入; 可编程的弱上拉。
	KBI3	1	I	TTL	引脚电平变化中断。
	PGD	x	O	DIG	供 ICSP 和 ICD 工作使用的串行执行数据输出。 ⁽³⁾
		x	I	ST	供 ICSP 和 ICD 工作使用的串行执行数据输入。 ⁽³⁾

图注: DIG = 数字电平输出; TTL = TTL 输入缓冲器; ST = 施密特触发器输入缓冲器; ANA = 模拟电平输入 / 输出;
x = 无关位 (TRIS 位不影响端口方向或在此可忽略)。

- 注 1: POR 时的配置由 PBADEN 配置位决定。默认情况下, 当 PBADEN 置 1 时, 引脚被配置为模拟输入; 当 PBADEN 清零时, 则被配置为数字输入。
2: 当 CCP2MX 配置位为 0 时 CCP2 的备用设置。默认设置为 RC1。
3: 当使能 ICSP 或 ICD 时, 禁止所有其他引脚功能。

表 10-4: 与 PORTB 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值 所在页
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	62
LATB	PORTB 数据锁存寄存器 (读和写数据锁存器)								62
TRISB	PORTB 数据方向控制寄存器								62
WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	62
IOCB	IOCB7	IOCB6	IOCB5	IOCB4	—	—	—	—	62
SLRCON	—	—	—	SLRE ⁽¹⁾	SLRD ⁽¹⁾	SLRC	SLRB	SLRA	63
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
INTCON2	RBPU	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	59
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	59
ANSELH	—	—	—	ANS12	ANS11	ANS10	ANS9	ANS8	62

图注: — = 未实现, 读为 0。PORTB 不使用阴影单元。

- 注 1: 在 PIC18F2XK20 器件上未实现。

10.4 PORTC、TRISC 和 LATC 寄存器

PORTC 是一个 8 位宽的双向端口，对应的数据方向寄存器是 TRISC。将 TRISC 某位置 1 (= 1) 时，会将 PORTC 的相应引脚设为输入（即，禁止输出驱动器）。将 TRISC 某位清零 (= 0) 时，会将 PORTC 的相应引脚设为输出（即，使能输出驱动器并将输出锁存器中的内容输出到选中引脚）。

数据锁存寄存器 (LATC) 也是存储器映射的。对 LATC 寄存器执行读 - 修改 - 写操作将读写 PORTC 的锁存输出值。

PORTC 与几种外设功能复用（表 10-5）。这些引脚配有施密特触发器输入缓冲器。RC1 是 CCP2 外设引脚的默认配置。通过将配置字 CONFIG3H 的 CCP2MX 位清零，可以将 CCP2 功能重新分配到 RB3 引脚。CCP2MX 配置位的默认状态是 1。

当使能外设功能时，应小心定义每个 PORTC 引脚的 TRIS 位。EUSART 和 MSSP 外设会改写 TRIS 位的设置，将引脚定义为输出或输入引脚（取决于外设配置）。更多信息，请参见相应的外设小节。

注： 上电复位时，这些引脚被配置为数字输入。

外设对引脚的改写会影响 TRISC 寄存器的内容。尽管外设可能会改写一个或多个引脚，读 TRISC 总是会返回其当前的内容。

例 10-3： 初始化 PORTC

```
CLRF    PORTC ; Initialize PORTC by
; clearing output
;data latches
CLRF    LATC ; Alternate method
; to clear output
;data latches
MOVLW   0CFh ; Value used to
; initialize data
; direction
MOVWF   TRISC ; Set RC<3:0> as inputs
; RC<5:4> as outputs
; RC<7:6> as inputs
```

PIC18F2XK20/4XK20

表 10-5: PORTC I/O 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RC0/T1OSO/ T13CKI	RC0	0	O	DIG	LATC<0> 数据输出。
		1	I	ST	PORTC<0> 数据输入。
	T1OSO	x	O	ANA	Timer1 振荡器输出；当使能 Timer1 振荡器时被使能。禁止数字 I/O。
	T13CKI	1	I	ST	Timer1/Timer3 计数器输入。
RC1/T1OSI/CCP2	RC1	0	O	DIG	LATC<1> 数据输出。
		1	I	ST	PORTC<1> 数据输入。
	T1OSI	x	I	ANA	Timer1 振荡器输入；当使能 Timer1 振荡器时被使能。禁止数字 I/O。
	CCP2 ⁽¹⁾	0	O	DIG	CCP2 比较输出和 PWM 输出；优先于端口数据。
		1	I	ST	CCP2 捕捉输入。
RC2/CCP1/P1A	RC2	0	O	DIG	LATC<2> 数据输出。
		1	I	ST	PORTC<2> 数据输入。
	CCP1	0	O	DIG	ECCP1 比较输出或 PWM 输出；优先于端口数据。
		1	I	ST	ECCP1 捕捉输入。
	P1A	0	O	DIG	ECCP1 增强型 PWM 输出，通道 A。可以在增强型 PWM 关闭期间被配置为三态。优先于端口数据。
RC3/SCK/SCL	RC3	0	O	DIG	LATC<3> 数据输出。
		1	I	ST	PORTC<3> 数据输入。
	SCK	0	O	DIG	SPI 时钟输出（MSSP 模块）；优先于端口数据。
		1	I	ST	SPI 时钟输入（MSSP 模块）。
	SCL	0	O	DIG	I ² C TM 时钟输出（MSSP 模块）；优先于端口数据。
		1	I	I ² C/SMB	I ² C 时钟输入（MSSP 模块）；输入类型取决于模块设置。
RC4/SDI/SDA	RC4	0	O	DIG	LATC<4> 数据输出。
		1	I	ST	PORTC<4> 数据输入。
	SDI	1	I	ST	SPI 数据输入（MSSP 模块）。
	SDA	1	O	DIG	I ² C 数据输出（MSSP 模块）；优先于端口数据。
		1	I	I ² C/SMB	I ² C 数据输入（MSSP 模块）；输入类型取决于模块设置。
RC5/SDO	RC5	0	O	DIG	LATC<5> 数据输出。
		1	I	ST	PORTC<5> 数据输入。
	SDO	0	O	DIG	SPI 数据输出（MSSP 模块）；优先于端口数据。
RC6/TX/CK	RC6	0	O	DIG	LATC<6> 数据输出。
		1	I	ST	PORTC<6> 数据输入。
	TX	1	O	DIG	异步串行发送数据输出（USART 模块）；优先于端口数据。用户必须将其配置为输出。
	CK	1	O	DIG	同步串行时钟输出（USART 模块）；优先于端口数据。
		1	I	ST	同步串行时钟输入（USART 模块）。
RC7/RX/DT	RC7	0	O	DIG	LATC<7> 数据输出。
		1	I	ST	PORTC<7> 数据输入。
	RX	1	I	ST	异步串行接收数据输入（USART 模块）。
	DT	1	O	DIG	同步串行数据输出（USART 模块）；优先于端口数据。
		1	I	ST	同步串行数据输入（USART 模块）。用户必须将其配置为输入。

图注： DIG = 数字电平输出； TTL = TTL 输入缓冲器； ST = 施密特触发器输入缓冲器； ANA = 模拟电平输入 / 输出； I²C/SMB = I²C/SMBus 输入缓冲器； x = 无关位（TRIS 位不影响端口方向或在此可忽略）。

注 1： 当 CCP2MX 配置位置 1 时 CCP2 的默认分配。备用分配为 RB3。

表 10-6：与 PORTC 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	62
LATC	PORTC 数据锁存寄存器（读和写数据锁存器）								62
TRISC	PORTC 数据方向控制寄存器								62
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	60
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON	61
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	61
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0	61
SLRCON	—	—	—	SLRE ⁽¹⁾	SLRD ⁽¹⁾	SLRC	SLRB	SLRA	63

图注：— = 未实现，读为 0。PORTC 不使用阴影单元。

注 1：在 PIC18F2XK20 器件上未实现。

10.5 PORTD、TRISD 和 LATD 寄存器

注: PORTD 仅在 40/44 引脚器件上可用。

PORTD 是一个 8 位宽的双向端口，对应的数据方向寄存器是 TRISD。将 TRISD 某位置 1 (= 1) 时，会将 PORTD 的相应引脚设为输入（即，禁止输出驱动器）。将 TRISD 某位清零 (= 0) 时，会将 PORTD 的相应引脚设为输出（即，使能输出驱动器并将输出锁存器中的内容输出到选中引脚）。

数据锁存寄存器 (LATD) 也是存储器映射的。对 LATD 寄存器执行读 - 修改 - 写操作将读写 PORTD 的锁存输出值。

PORTD 上的所有引脚都配有施密特触发器输入缓冲器。每个引脚都可被单独配置为输入或输出。

PORTD 的 3 个引脚与增强型 CCP 模块的 P1B、P1C 和 P1D 输出复用。这些额外的 PWM 输出引脚的操作在第 16.0 节 “增强型捕捉 / 比较 /PWM (ECCP) 模块” 中有更详细的阐述。

注: 上电复位时，这些引脚被配置为数字输入。

还可通过将控制位 PSPMODE (TRISE<4>) 置 1，将 PORTD 配置为 8 位宽的微处理器端口 (并行从端口)。在此模式下，输入缓冲器是 TTL。关于并行从端口 (PSP) 的更多信息，请参见第 10.9 节 “并行从端口”。

注: 当增强型 PWM 模式使用双输出或四输出时，PORTD 的 PSP 功能被自动禁止。

例 10-4: 初始化 PORTD

```
CLRF    PORTD    ; Initialize PORTD by
                  ; clearing output
                  ; data latches
CLRF    LATD     ; Alternate method
                  ; to clear output
                  ; data latches
MOVlw   0CFh    ; Value used to
                  ; initialize data
                  ; direction
MOVwf   TRISD    ; Set RD<3:0> as inputs
                  ; RD<5:4> as outputs
                  ; RD<7:6> as inputs
```

表 10-7: PORTD I/O 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RD0/PSP0	RD0	0	O	DIG	LATD<0> 数据输出。
		1	I	ST	PORTD<0> 数据输入。
	PSP0	x	O	DIG	PSP 读数据输出 (LATD<0>)；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
RD1/PSP1	RD1	0	O	DIG	LATD<1> 数据输出。
		1	I	ST	PORTD<1> 数据输入。
	PSP1	x	O	DIG	PSP 读数据输出 (LATD<1>)；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
RD2/PSP2	RD2	0	O	DIG	LATD<2> 数据输出。
		1	I	ST	PORTD<2> 数据输入。
	PSP2	x	O	DIG	PSP 读数据输出 (LATD<2>)；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
RD3/PSP3	RD3	0	O	DIG	LATD<3> 数据输出。
		1	I	ST	PORTD<3> 数据输入。
	PSP3	x	O	DIG	PSP 读数据输出 (LATD<3>)；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
RD4/PSP4	RD4	0	O	DIG	LATD<4> 数据输出。
		1	I	ST	PORTD<4> 数据输入。
	PSP4	x	O	DIG	PSP 读数据输出 (LATD<4>)；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
RD5/PSP5/P1B	RD5	0	O	DIG	LATD<5> 数据输出。
		1	I	ST	PORTD<5> 数据输入。
	PSP5	x	O	DIG	PSP 读数据输出 (LATD<5>)；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	P1B	0	O	DIG	ECCP1 增强型 PWM 输出，通道 B；优先于端口数据和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RD6/PSP6/P1C	RD6	0	O	DIG	LATD<6> 数据输出。
		1	I	ST	PORTD<6> 数据输入。
	PSP6	x	O	DIG	PSP 读数据输出 (LATD<6>)；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	P1C	0	O	DIG	ECCP1 增强型 PWM 输出，通道 C；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。
RD7/PSP7/P1D	RD7	0	O	DIG	LATD<7> 数据输出。
		1	I	ST	PORTD<7> 数据输入。
	PSP7	x	O	DIG	PSP 读数据输出 (LATD<7>)；优先于端口数据。
		x	I	TTL	PSP 写数据输入。
	P1D	0	O	DIG	ECCP1 增强型 PWM 输出，通道 D；优先于端口和 PSP 数据。可以在增强型 PWM 关闭期间被配置为三态。

图注: DIG = 数字电平输出; TTL = TTL 输入缓冲器; ST = 施密特触发器输入缓冲器; x = 无关位 (TRIS 位不影响端口方向或在此可忽略)。

PIC18F2XK20/4XK20

表 10-8：与 PORTD 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTD ⁽¹⁾	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	62
LATD ⁽¹⁾	PORTD 数据锁存寄存器（读和写数据锁存器）								62
TRISD ⁽¹⁾	PORTD 数据方向控制寄存器								62
TRISE ⁽¹⁾	IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0	62
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
SLRCON	—	—	—	SLRE ⁽¹⁾	SLRD ⁽¹⁾	SLRC	SLRB	SLRA	63

图注：— = 未实现，读为 0。PORTD 不使用阴影单元。

注 1：在 PIC18F2XK20 器件上未实现。

10.6 PORTE、TRISE 和 LATE 寄存器

根据选定的特定 PIC18F2XK20/4XK20 器件，PORTE 可通过两种不同的方式实现。

10.6.1 PIC18F4XK20 器件上的 PORTE

对于 PIC18F4XK20 器件，PORTE 是一个 4 位宽的端口。3 个引脚（RE0/RD/AN5、RE1/WR/AN6 和 RE2/CS/AN7）可单独配置为输入或输出。这些引脚配有施密特触发器输入缓冲器。当被选为模拟输入时，这些引脚将读为 0。

对应的数据方向寄存器是 TRISE。将 TRISE 某位置 1 (= 1) 时，会将 PORTE 的相应引脚设为输入（即，禁止输出驱动器）。将 TRISE 某位清零 (= 0) 时，会将 PORTE 的相应引脚设为输出（即，使能输出驱动器并将输出锁存器中的内容输出到选中引脚）。

TRISE 控制着 RE 引脚的方向，即使它们被用作模拟输入。用户在将这些引脚用作模拟输入时，必须确保将它们配置为输入。

注： 上电复位时，RE<2:0>被配置为模拟输入。

TRISE 寄存器的高 4 位也控制着并行从端口的操作。它们的操作在寄存器 10-1 中进行解释。

数据锁存寄存器（LATE）也是存储器映射的。对 LATE 寄存器执行读 - 修改 - 写操作将读写 PORTE 的锁存输出值。

PORTE 的第 4 个引脚（MCLR/VPP/RE3）是仅输入引脚。其操作由 MCLRE 配置位控制。当被选为端口引脚（MCLRE = 0）时，它仅用作数字输入引脚；这样，它不具备与操作相关的 TRIS 或 LAT 位。否则，它用作器件的主复位输入。在任何一种配置中，RE3 都用作编程期间的编程电压输入。

注： 上电复位时，仅当主复位功能禁止时，才能将 RE3 使能为数字输入。

例 10-5：初始化 PORTE

```
CLRF    PORTE    ; Initialize PORTE by
                  ; clearing output
                  ; data latches
CLRF    LATE     ; Alternate method
                  ; to clear output
                  ; data latches
MOVLW  1Fh      ; Configure analog pins
ANDWF  ANSEL,w ; for digital only
MOVLW  05h      ; Value used to
                  ; initialize data
                  ; direction
MOVWF  TRISE    ; Set RE<0> as input
                  ; RE<1> as output
                  ; RE<2> as input
```

10.6.2 PIC18F2XK20 器件上的 PORTE

对于 PIC18F2XK20 器件，仅当主复位功能禁止（MCLR = 0）时，才能使用 PORTE。在这些情况下，PORTE 是仅由 RE3 组成的仅输入端口，只包含一位。引脚操作如前所述。

PIC18F2XK20/4XK20

寄存器 10-1: TRISE: PORTE/PSP 控制寄存器 (仅限 PIC18F4XK20 器件)

R-0	R-0	R/W-0	R/W-0	U-0	R/W-1	R/W-1	R/W-1
IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0
bit 7	bit 0						

图注:

R = 可读位

-n = POR 时的值

W = 可写位

1 = 置 1

U = 未实现位, 读为 0

0 = 清零

x = 未知

bit 7 **IBF:** 输入缓冲区满状态位
1 = 已接收一个字, 等待 CPU 读取
0 = 未接收到任何字

bit 6 **OBF:** 输出缓冲区满状态位
1 = 输出缓冲区仍保存着之前写入的字
0 = 输出缓冲区已被读取

bit 5 **IBOV:** 输入缓冲区溢出检测位 (处于微处理器模式下)
1 = 之前输入的字尚未被读取时发生写操作 (必须用软件清零)
0 = 未发生溢出

bit 4 **PSPMODE:** 并行从端口模式选择位
1 = 并行从端口模式
0 = 通用 I/O 模式

bit 3 未实现: 读为 0

bit 2 **TRISE2:** RE2 方向控制位
1 = 输入
0 = 输出

bit 1 **TRISE1:** RE1 方向控制位
1 = 输入
0 = 输出

bit 0 **TRISE0:** RE0 方向控制位
1 = 输入
0 = 输出

表 10-9: PORTE I/O 汇总

引脚	功能	TRIS 设置	I/O	I/O 类型	说明
RE0/RD/AN5	RE0	0	O	DIG	LATE<0> 数据输出；不受模拟输入影响。
		1	I	ST	PORTE<0> 数据输入；当使能模拟输入时被禁止。
	RD	1	I	TTL	PSP 读使能输入（PSP 被使能）。
	AN5	1	I	ANA	A/D 输入通道 5；POR 时的默认输入配置。
RE1/WR/AN6	RE1	0	O	DIG	LATE<1> 数据输出；不受模拟输入影响。
		1	I	ST	PORTE<1> 数据输入；当使能模拟输入时被禁止。
	WR	1	I	TTL	PSP 写使能输入（PSP 被使能）。
	AN6	1	I	ANA	A/D 输入通道 6；POR 时的默认输入配置。
RE2/CS/AN7	RE2	0	O	DIG	LATE<2> 数据输出；不受模拟输入影响。
		1	I	ST	PORTE<2> 数据输入；当使能模拟输入时被禁止。
	CS	1	I	TTL	PSP 写使能输入（PSP 被使能）。
	AN7	1	I	ANA	A/D 输入通道 7；POR 时的默认输入配置。
MCLR/VPP/ RE3 ^(1,2)	MCLR	—	I	ST	外部主复位输入；当 MCLRE 配置位置 1 时被使能。
	VPP	—	I	ANA	高压检测；用于 ICSP™ 模式进入检测。始终可用，与引脚模式无关。
	RE3	— ⁽²⁾	I	ST	PORTE<3> 数据输入；当 MCLRE 配置位清零时被使能。

图注: DIG = 数字电平输出; TTL = TTL 输入缓冲器; ST = 施密特触发器输入缓冲器; ANA = 模拟电平输入 / 输出;
x = 无关位 (TRIS 位不影响端口方向或在此可忽略)。

注 1: RE3 仅在 PIC18F2XK20 和 PIC18F4XK20 器件上可用。所有其他 PORTE 引脚仅在 PIC18F4XK20 器件上实现。

2: RE3 没有相应的 TRIS 位来控制数据方向。

表 10-10: 与 PORTE 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值 所在页
PORTE	—	—	—	—	RE3 ^(1,2)	RE2	RE1	RE0	62
LATE ⁽²⁾	—	—	—	—	—	LATE 数据输出寄存器			62
TRISE ⁽³⁾	IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0	62
SLRCON	—	—	—	SLRE ⁽³⁾	SLRD ⁽³⁾	SLRC	SLRB	SLRA	63
ANSEL	ANS7 ⁽³⁾	ANS6 ⁽³⁾	ANS5 ⁽³⁾	ANS4	ANS3	ANS2	ANS1	ANS0	62

图注: — = 未实现, 读为 0。PORTE 不使用阴影单元。

注 1: 仅当主复位功能被禁止 (MCLRE 配置位 = 0) 时实现。

2: RE3 是在 PIC18F2XK20 和 PIC18F4XK20 器件上都实现的唯一 PORTE 位。所有其他位仅在 PORTE 实现时才实现 (即 PIC18F4XK20 器件)。

3: 在 PIC18F2XK20 器件上未实现。

10.7 端口模拟控制

一些端口引脚与模拟功能（例如模数转换器和比较器）复用。当这些 I/O 引脚要用作模拟输入时，必须禁止数字输入缓冲器，以避免数字输入的错误偏置导致过大的电流。通过 ANSEL 和 ANSELH 寄存器，可以对共用模拟功能的引脚上的数字输入缓冲器进行单独控制。将

ANS_x 位设为高电平可以禁止关联的数字输入缓冲器，并导致对该引脚的所有读操作返回 0，同时允许该引脚的模拟功能正确工作。

ANS_x 位的状态不会影响数字输出功能。相关 TRIS_x 位清零且 ANS_x 位置 1 的引脚将仍作为数字输出工作，但输入模式将变为模拟。当在受影响的端口上执行读 - 修改 - 写操作时，这会引起意外行为。

寄存器 10-2： ANSEL： 模拟选择寄存器 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANS7 ⁽¹⁾	ANS6 ⁽¹⁾	ANS5 ⁽¹⁾	ANS4	ANS3	ANS2	ANS1	ANS0
bit 7				bit 0			

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	ANS7: RE2 模拟选择控制位 ⁽¹⁾
	1 = 禁止 RE2 的数字输入缓冲器
	0 = 使能 RE2 的数字输入缓冲器
bit 6	ANS6: RE1 模拟选择控制位 ⁽¹⁾
	1 = 禁止 RE1 的数字输入缓冲器
	0 = 使能 RE1 的数字输入缓冲器
bit 5	ANS5: RE0 模拟选择控制位 ⁽¹⁾
	1 = 禁止 RE0 的数字输入缓冲器
	0 = 使能 RE0 的数字输入缓冲器
bit 4	ANS4: RA5 模拟选择控制位
	1 = 禁止 RA5 的数字输入缓冲器
	0 = 使能 RA5 的数字输入缓冲器
bit 3	ANS3: RA3 模拟选择控制位
	1 = 禁止 RA3 的数字输入缓冲器
	0 = 使能 RA3 的数字输入缓冲器
bit 2	ANS2: RA2 模拟选择控制位
	1 = 禁止 RA2 的数字输入缓冲器
	0 = 使能 RA2 的数字输入缓冲器
bit 1	ANS1: RA1 模拟选择控制位
	1 = 禁止 RA1 的数字输入缓冲器
	0 = 使能 RA1 的数字输入缓冲器
bit 0	ANS0: RA0 模拟选择控制位
	1 = 禁止 RA0 的数字输入缓冲器
	0 = 使能 RA0 的数字输入缓冲器

注 1： 这些位在 PIC18F2XK20 器件上未实现。

寄存器 10-3: ANSELH: 模拟选择寄存器 2

U-0	U-0	U-0	R/W-1 ⁽¹⁾				
—	—	—	ANS12	ANS11	ANS10	ANS9	ANS8
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

- | | |
|---------|---|
| bit 7-5 | 未实现: 读为 0 |
| bit 4 | ANS12: RB0 模拟选择控制位
1 = 禁止 RB0 的数字输入缓冲器
0 = 使能 RB0 的数字输入缓冲器 |
| bit 3 | ANS11: RB4 模拟选择控制位
1 = 禁止 RB4 的数字输入缓冲器
0 = 使能 RB4 的数字输入缓冲器 |
| bit 2 | ANS10: RB1 模拟选择控制位
1 = 禁止 RB1 的数字输入缓冲器
0 = 使能 RB1 的数字输入缓冲器 |
| bit 1 | ANS9: RB3 模拟选择控制位
1 = 禁止 RB3 的数字输入缓冲器
0 = 使能 RB3 的数字输入缓冲器 |
| bit 0 | ANS8: RB2 模拟选择控制位
1 = 禁止 RB2 的数字输入缓冲器
0 = 使能 RB2 的数字输入缓冲器 |

注 1: 默认状态由 CONFIG3H 的 PBADEN 位决定。当 PBADEN = 0 时, 默认状态为 0。

10.8 端口斜率控制

可对每个端口的输出斜率进行编程，以选择标准变化速率或降低的变化速率（标准变化速率的 0.1 倍，最大程度降低 EMI）。对于所有端口，默认斜率是降低的变化速率。

寄存器 10-4: **SLRCON: 斜率控制寄存器**

U-0	U-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	
—	—	—	SLRE ⁽¹⁾	SLRD ⁽¹⁾	SLRC	SLRB	SLRA	
bit 7								bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-5 未实现：读为 0

SLRE: PORTE 斜率控制位⁽¹⁾

1 = PORTE 上的所有输出的斜率选择限制速率

0 = PORTE 上的所有输出的斜率选择标准速率

bit 3 **SLRD: PORTD 斜率控制位⁽¹⁾**

1 = PORTD 上的所有输出的斜率选择限制速率

0 = PORTD 上的所有输出的斜率选择标准速率

bit 2 **SLRC: PORTC 斜率控制位**

1 = PORTC 上的所有输出的斜率选择限制速率

0 = PORTC 上的所有输出的斜率选择标准速率

bit 1 **SLRB: PORTB 斜率控制位**

1 = PORTB 上的所有输出的斜率选择限制速率

0 = PORTB 上的所有输出的斜率选择标准速率

bit 0 **SLRA: PORTA 斜率控制位**

1 = PORTA 上的所有输出的斜率选择限制速率⁽²⁾

0 = PORTA 上的所有输出的斜率选择标准速率

注 1: 这些位在 PIC18F2XK20 器件上未实现。

2: 当引脚用作 CLKOUT 时， RA6 的斜率默认设为标准速率。

10.9 并行从端口

注：并行从端口仅在PIC18F4XK20器件上可用。

除了作为通用 I/O 端口，PORTD 还可用作一个 8 位宽的并行从端口（PSP）或微处理器端口。PSP 操作由 TRISE 寄存器（寄存器 10-1）的高 4 位控制。只要增强型 CCP 模块不是工作在双输出或四输出 PWM 模式下，将控制位 PSPMODE（TRISE<4>）置 1 可使能 PSP 操作。在从模式下，可从外部异步地读写端口。

PSP 可以直接与 8 位微处理器数据总线接口。外部微处理器可以读或写 PORTD 8 位锁存值。将控制位 PSPMODE 置 1 可使能 PORTE I/O 引脚，使之成为微处理器端口的控制输入。当置 1 时，端口引脚 RE0 为 RD 输入，RE1 为 WR 输入，RE2 为 CS（片选）输入。要实现此功能，TRISE 寄存器（TRISE<2:0>）对应的数据方向位必须配置为输入（置 1）且 ANSEL<7:5> 位必须清零。

当第一次检测到 CS 和 WR 线均为低电平时发生对 PSP 的写操作，当检测到任何一根线为高电平时结束操作。写操作结束后，PSPIF 和 IBF 标志位均置 1。

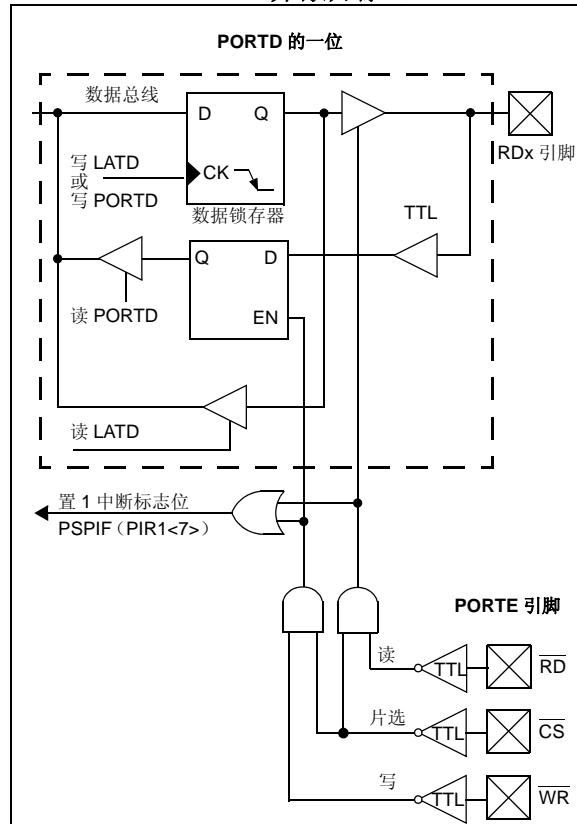
当第一次检测到 CS 和 RD 线均为低电平时发生对 PSP 的读操作。PORTD 中的数据被读出且 OBF 位被清零。如果用户通过将新数据写入 PORTD 而试图将 OBF 置 1，该数据会立即被读出；但 OBF 位不会被置 1。

当 CS 或 RD 线被检测到高电平时，PORTD 引脚返回到输入状态且 PSPIF 位被置 1。用户应用程序在处理 PSP 之前应该等待 PSPIF 被置 1；发生这种情况时，可以查询 IBF 和 OBF 位并进行相应的操作。

写和读模式下控制信号的时序分别如图 10-3 和图 10-4 所示。

图 10-2：

**PORTE 和 PORTD 框图
(并行从端口)**



注： I/O 引脚与 VDD 和 Vss 之间接有保护二极管。

PIC18F2XK20/4XK20

图 10-3: 并行从端口写波形图

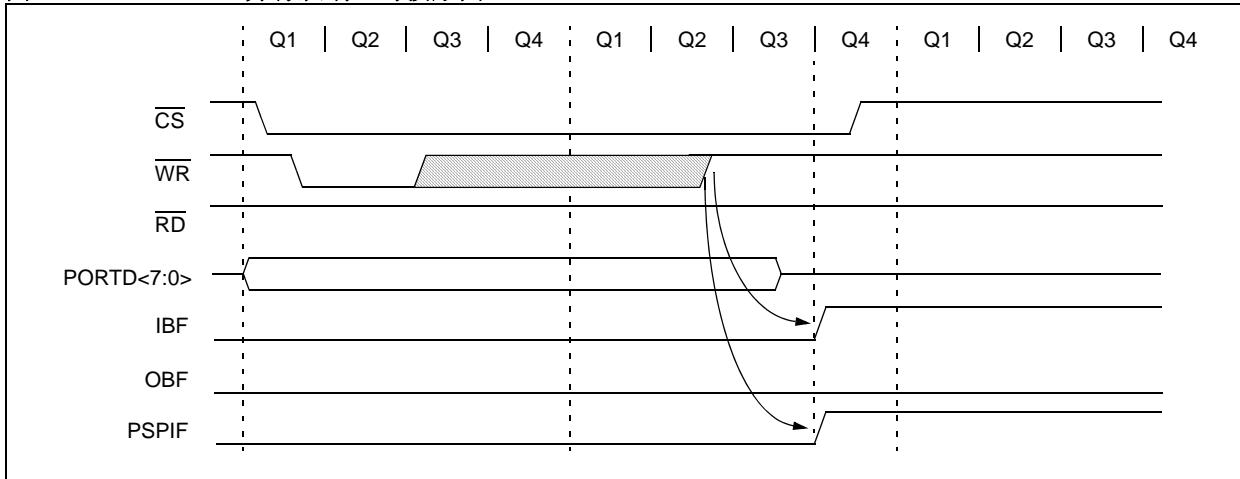


图 10-4: 并行从端口读波形图

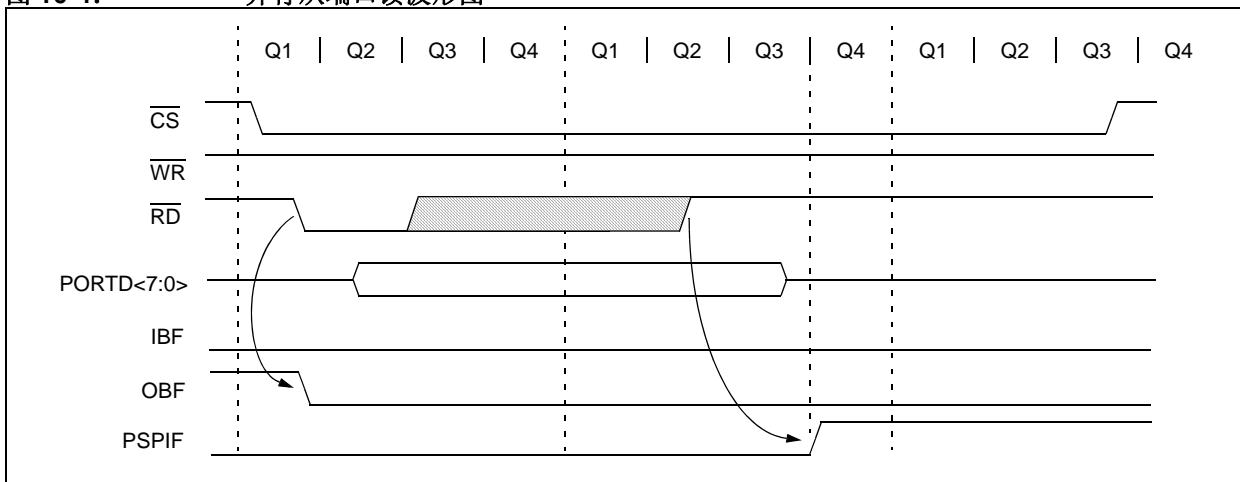


表 10-11：与并行从端口相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
PORTD ⁽¹⁾	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0	62
LATD ⁽¹⁾	PORTD 数据锁存寄存器（读和写数据锁存器）								62
TRISD ⁽¹⁾	PORTD 数据方向控制寄存器								62
PORTE	—	—	—	—	RE3	RE2 ⁽¹⁾	RE1 ⁽¹⁾	RE0 ⁽¹⁾	62
LATE ⁽¹⁾	—	—	—	—	—	LATE 数据输出位			62
TRISE ⁽¹⁾	IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0	62
SLRCON	—	—	—	SLRE ⁽¹⁾	SLRD ⁽¹⁾	SLRC	SLRB	SLRA	63
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
ANSEL	ANS7 ⁽¹⁾	ANS6 ⁽¹⁾	ANS5 ⁽¹⁾	ANS4	ANS3	ANS2	ANS1	ANS0	62

图注：— = 未实现，读为 0。并行从端口不使用阴影单元。

注 1：在 PIC18F2XK20 器件上未实现。

PIC18F2XK20/4XK20

注:

11.0 捕捉 / 比较 /PWM (CCP) 模块

PIC18F2XK20/4XK20 器件都有两个 CCP (捕捉 / 比较 /PWM) 模块。每个模块包含一个 16 位寄存器，可用作 16 位捕捉寄存器、16 位比较寄存器或 PWM 主 / 从占空比寄存器。

CCP1 实现为增强型 CCP 模块，具有标准捕捉和比较模式以及增强型 PWM 模式。ECCP 实现在第 16.0 节“增强型捕捉 / 比较 /PWM (ECCP) 模块”中进行讨论。CCP2 实现为标准 CCP 模块，但不具有增强功能。

本章中描述的捕捉和比较操作适用于标准和增强型 CCP 模块。

注： 在本节和第 16.0 节“增强型捕捉 / 比较 /PWM (ECCP) 模块”中，在提到与 CCP 模块相关的寄存器和位名称时，一般会使用“x”或“y”代替特定的模块编号。因此，“CCPxCON”可能指 CCP1、CCP2 或 ECCP1 的控制寄存器。“CCPxCON”在这些章节中用来指代模块控制寄存器，与 CCP 模块是标准还是增强型实现无关。

寄存器 11-1： CCP2CON：标准捕捉 / 比较 /PWM 控制寄存器

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-6

未实现： 读为 0

bit 5-4

DC2B<1:0>： CCP2 模块的 PWM 占空比 bit 1 和 bit 0

捕捉模式：

未使用。

比较模式：

未使用。

PWM 模式：

这两位是 10 位 PWM 占空比的低 2 位 (bit 1 和 bit 0)。占空比的高 8 位 (DC2B<9:2>) 在 CCPR2L 中。

bit 3-0

CCP2M<3:0>： CCP2 模式选择位

0000 = 禁止捕捉 / 比较 /PWM (复位 CCP2 模块)

0001 = 保留

0010 = 比较模式，匹配时输出电平翻转 (CCP2IF 位置 1)

0011 = 保留

0100 = 捕捉模式，每个下降沿

0101 = 捕捉模式，每个上升沿

0110 = 捕捉模式，每 4 个上升沿

0111 = 捕捉模式，每 16 个上升沿

1000 = 比较模式：初始化 CCP2 引脚为低电平；比较匹配时强制 CCP2 引脚为高电平 (CCP2IF 位置 1)

1001 = 比较模式：初始化 CCP2 引脚为高电平；比较匹配时强制 CCP2 引脚为低电平 (CCP2IF 位置 1)

1010 = 比较模式：比较匹配时产生软件中断 (CCP2IF 位置 1, CCP2 引脚反映 I/O 状态)

1011 = 比较模式：当 CCP2 发生匹配时触发特殊事件信号、复位定时器或启动 A/D 转换 (CCP2IF 位置 1)

11xx = PWM 模式

11.1 CCP 模块配置

每个捕捉 / 比较 / PWM 模块均与一个控制寄存器（通常为 CCPxCON）和一个数据寄存器（CCPRx）相关联。数据寄存器由两个 8 位寄存器组成：CCPRxL（低字节）和 CCPRxH（高字节）。所有寄存器都是可读写的。

11.1.1 CCP 模块和定时器资源

CCP 模块根据选定的模式使用 Timer1、Timer2 或 Timer3。Timer1 和 Timer3 适用于工作在捕捉或比较模式下的模块，而 Timer2 适用于工作在 PWM 模式下的模块。

表 11-1： CCP 模式——定时器资源

CCP/ECCP 模式	定时器资源
捕捉	Timer1 或 Timer3
比较	Timer1 或 Timer3
PWM	Timer2

表 11-2： CCP1 和 CCP2 在使用定时器资源方面的相互关系

CCP1 模式	CCP2 模式	相互关系
捕捉	捕捉	每个模块都可用 TMR1 或 TMR3 作为时基。每个 CCP 的时基也可以各不相同。
捕捉	比较	可将 CCP2 配置为特殊事件触发器用以复位 TMR1 或 TMR3（取决于所使用的时基）。也可用于在发生触发事件时自动触发 A/D 转换。如果 CCP1 使用与 CCP2 相同的定时器作为时基，上述操作可能会对 CCP1 产生影响。
比较	捕捉	可将 CCP1 配置为特殊事件触发器用以复位 TMR1 或 TMR3（取决于所使用的时基）。如果 CCP2 使用与 CCP1 相同的定时器作为时基，上述操作可能会对 CCP2 产生影响。
比较	比较	每个模块均可配置为特殊事件触发器用以复位时基。CCP2 触发事件还可自动触发 A/D 转换。如果两个模块使用相同的时基，可能会发生冲突。
捕捉	PWM	无
比较	PWM	无
PWM ⁽¹⁾	捕捉	无
PWM ⁽¹⁾	比较	无
PWM ⁽¹⁾	PWM	两个 PWM 具有相同的频率和更新速率（TMR2 中断）。

注 1： 包括标准和增强型 PWM 操作。

要将哪个特定的定时器分配给 CCP 模块是由 T3CON 寄存器（寄存器 15-1）中的“Timer-to-CCP（将定时器分配给 CCP）”使能位决定的。如果将两个 CCP 模块配置为工作在相同的模式（捕捉 / 比较或 PWM）下，那么这两个模块可同时被激活并可共享相同的定时器资源。图 11-1 和图 11-2 总结了这两个模块间的相互关系。在异步计数器模式下，可能无法进行捕捉操作。

11.1.2 CCP2 引脚分配

可根据器件配置改变 CCP2（捕捉输入、比较和 PWM 输出）的引脚分配。CCP2MX 配置位决定哪个引脚将与 CCP2 复用。默认情况下，CCP2 引脚被分配给 RC1（CCP2MX = 1）。如果清零该配置位，CCP2 将与 RB3 复用。

改变 CCP2 的引脚分配并不会自动改变对端口引脚的配置。无论其引脚的分配如何，用户必须始终确保与 CCP2 操作相对应的 TRIS 寄存器配置正确。

11.2 捕捉模式

在捕捉模式下，当相应的 CCP_x 引脚发生以下事件时，CCPR_{xH}:CCPR_{xL} 寄存器对捕捉 TMR1 或 TMR3 寄存器的 16 位值。事件定义为以下情况之一：

- 每个下降沿
- 每个上升沿
- 每 4 个上升沿
- 每 16 个上升沿

事件由 CCP_{xCON} 寄存器的模式选择位 CCP_{xM<3:0>} 选择。当完成一次捕捉时，中断请求标志位 CCP_{xIF} 置 1；它必须用软件清零。如果在读取寄存器 CCPR_x 值之前发生了另一次捕捉，那么原来的捕捉值会被新的捕捉值覆盖。

11.2.1 CCP 引脚配置

在捕捉模式下，应通过将相应的 TRIS 方向位置 1 将 CCP_x 引脚配置为输入。

注： 如果 CCP_x 引脚被配置为输出，则写端口将产生一次捕捉条件。

11.2.2 TIMER1/TIMER3 模式选择

用于捕捉功能的定时器（Timer1 和 / 或 Timer3）必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，可能无法进行捕捉操作。可在 T3CON 寄存器中选择用于每个 CCP 模块的定时器（见第 11.1.1 节“**CCP 模块和定时器资源**”）。

11.2.3 软件中断

当捕捉模式改变时，可能会产生错误的捕捉中断。用户应该保持 CCP_{xIE} 中断允许位清零以避免错误中断。应在工作模式改变后清零中断标志位 CCP_{xIF}。

11.2.4 CCP 预分频器

在捕捉模式下有 4 种预分频比设置；它们作为工作模式的一部分由模式选择位 (CCPxM<3:0>) 指定。只要关闭 CCP 模块或禁止捕捉模式，预分频器计数器就会被清零。这意味着任何复位都会将预分频器计数器清零。

在两个捕捉预分频比之间切换可能会产生中断。而且，预分频器计数器不会被清零；因此，第一次捕捉可能来自于一个非零的预分频器。例 11-1 给出了切换捕捉预分频比时建议采用的方法。这个示例使预分频器计数器清零且不会产生错误中断。

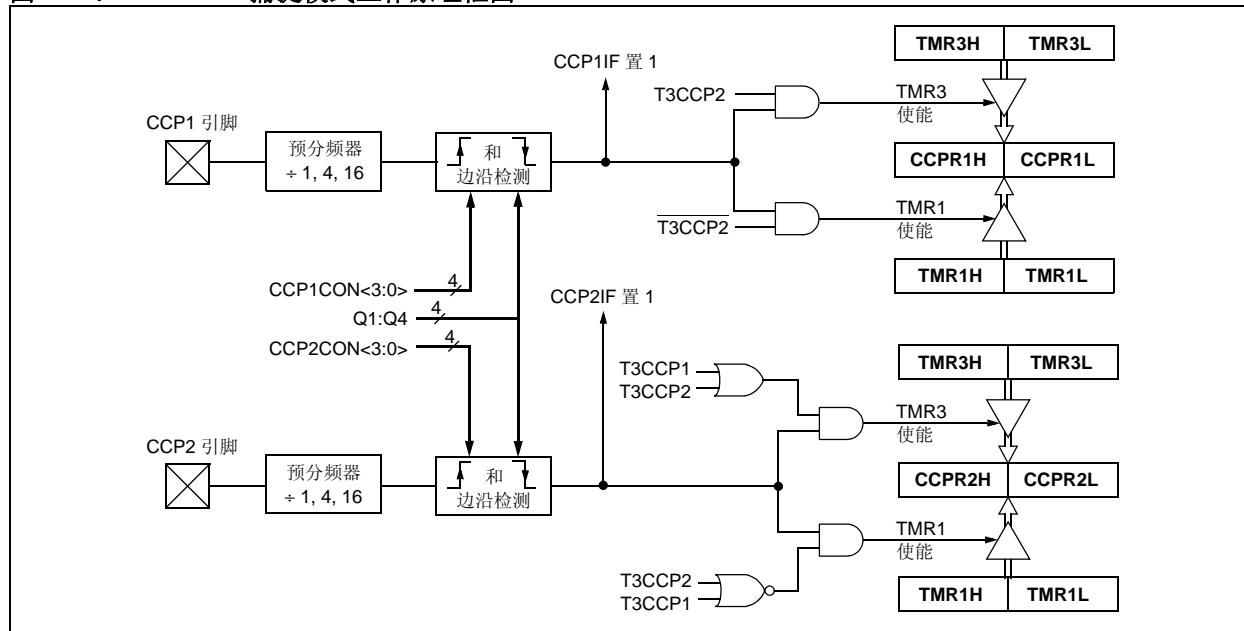
例 11-1:

改变捕捉预分频比 (以 CCP2 为例)

```
CLRF  CCP2CON      ; Turn CCP module off
MOVLW NEW_CAPT_PS ; Load WREG with the
                    ; new prescaler mode
                    ; value and CCP ON
MOVWF CCP2CON      ; Load CCP2CON with
                    ; this value
```

PIC18F2XK20/4XK20

图 11-1： 捕捉模式工作原理框图



11.3 比较模式

在比较模式下，16位CCPRx寄存器的值不断与TMR1或TMR3寄存器对的值作比较。当两者匹配时，CCPx引脚将会：

- 驱动为高电平
- 驱动为低电平
- 电平翻转（高电平变为低电平或低电平变为高电平）
- 保持不变（即反映I/O锁存器的状态）

引脚动作取决于模式选择位（CCPxM<3:0>）的值。同时，中断标志位CCPxIF置1。

11.3.1 CCP引脚配置

用户必须通过将相应的TRIS位清零，将CCPx引脚配置为输出。

注： 清零CCPxCON寄存器会将CCPx比较输出锁存器（取决于器件配置）强制为默认的低电平。这不是PORTB或PORTC I/O数据锁存器。

11.3.2 TIMER1/TIMER3模式选择

如果CCP模块使用比较功能，则Timer1和/或Timer3必须运行在定时器模式或同步计数器模式下。在异步计数器模式下，可能无法进行比较操作。

11.3.3 软件中断模式

当选择了“产生软件中断模式”（CCPxM<3:0> = 1010）时，相应的CCPx引脚不受影响。只会影响CCPxIF中断标志。

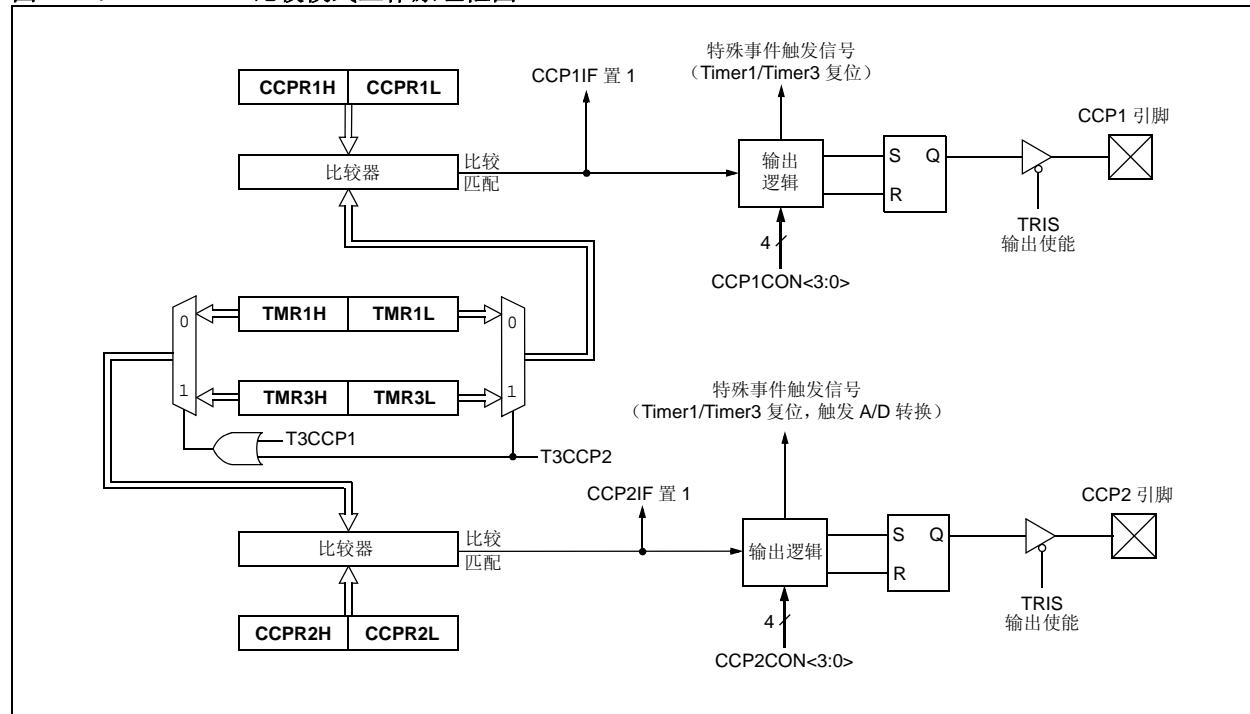
11.3.4 特殊事件触发器

两个CCP模块均配备了一个特殊事件触发器。在比较模式下可产生内部硬件信号以触发其他模块动作。通过选择比较特殊事件触发模式（CCPxM<3:0> = 1011），使能特殊事件触发器。

对于任何一个CCP模块，无论当前使用哪个定时器资源作为模块的时基，特殊事件触发信号将把定时器寄存器对复位。这样CCPRx寄存器可用作两个定时器中任一定时器的可编程周期寄存器。

CCP2的特殊事件触发信号还能启动A/D转换。要实现此功能，必须首先使能A/D转换器。

图 11-2： 比较模式工作原理框图



PIC18F2XK20/4XK20

表 11-3: 与捕捉、比较、TIMER1 和 TIMER3 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	SBOREN	—	RI	TO	PD	POR	BOR	58
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62
TRISB	PORTB 数据方向控制寄存器								62
TRISC	PORTC 数据方向控制寄存器								62
TMR1L	Timer1 寄存器的低字节								60
TMR1H	Timer1 寄存器的高字节								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	60
TMR3H	Timer3 寄存器的高字节								61
TMR3L	Timer3 寄存器的低字节								61
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON	61
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								61
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								61
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								61
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								61
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	61

图注: — = 未实现, 读为 0。捕捉 / 比较、Timer1 或 Timer3 不使用阴影单元。

注 1: 在 PIC18F2XK20 器件上未实现。

11.4 PWM 模式

PWM 模式在 CCP2 引脚上（对于 CCP 模块）以及在 P1A 到 P1D 引脚上（对于 ECCP 模块）产生脉宽调制信号。此后，调制输出引脚被称为 CCPx 引脚。占空比、周期和分辨率由以下寄存器决定：

- PR2
- T2CON
- CCPRxL
- CCPxCON

在脉宽调制（Pulse-Width Modulation, PWM）模式下，CCP 模块会在 CCPx 引脚上产生最大 10 位分辨率的 PWM 输出信号。由于 CCPx 引脚与端口数据锁存器复用，必须清零相应的 TRIS 位以使能 CCPx 引脚输出驱动器。

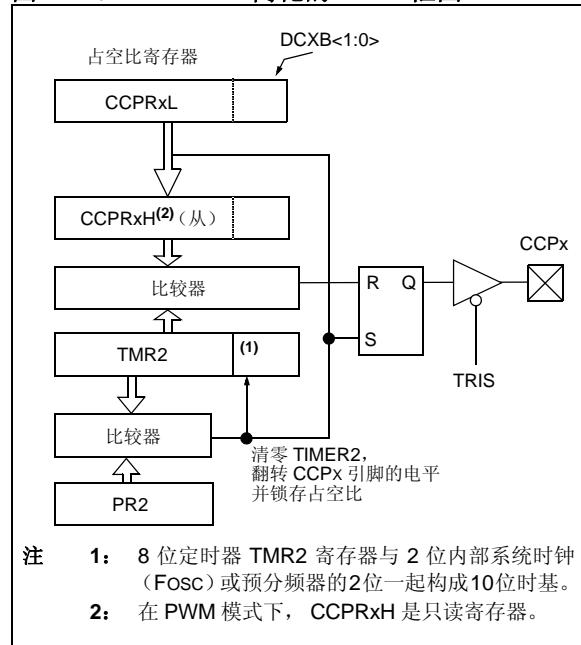
注： 清零 CCPxCON 寄存器将放弃对 CCPx 引脚的 CCPx 控制。

图 11-3 给出了 PWM 工作原理的简化框图。

图 11-4 给出了 PWM 信号的典型波形。

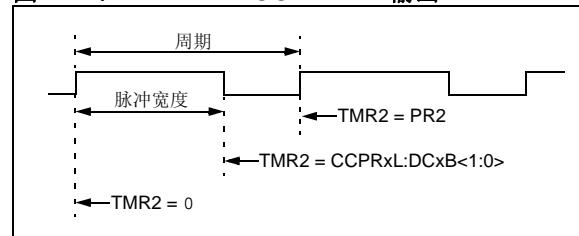
关于如何设置 CCP 模块使之工作于 PWM 模式的详细步骤，请参见第 11.4.7 节“设置 PWM 操作”。

图 11-3： 简化的 PWM 框图



PWM 输出（图 11-4）有一个时基（周期）和一段输出保持为高电平的时间（占空比）。

图 11-4： CCP PWM 输出



11.4.1 PWM 周期

PWM 周期可通过 Timer2 的 PR2 寄存器来指定。PWM 周期可由公式 11-1 计算。

公式 11-1: PWM 周期

$$\text{PWM 周期} = [(PR2) + I] \cdot 4 \cdot Tosc \cdot \\ (\text{TMR2 预分频值})$$

注: $TOSC = 1/FOSC$ 。

当 TMR2 中的值与 PR2 中的值相等时, 在下一个递增周期将发生以下 3 个事件:

- TMR2 被清零。
- CCPx 引脚被置 1。(例外情况: 如果 PWM 占空比 = 0%, 引脚将不会被置 1。)
- PWM 占空比从 CCPRxL 锁存到 CCPRxH。

注: 在确定 PWM 频率时不会用到 Timer2 后分频比(见第 14.1 节“Timer2 工作原理”)。

11.4.2 PWM 占空比

通过将 10 位值写入多个寄存器来指定 PWM 占空比: CCPRxL 寄存器和 CCPxCON 寄存器的 DCxB<1:0>位。CCPRxL 包含高 8 位而 CCPxCON 寄存器的 DCxB<1:0>位包含低 2 位。可以在任何时候写入 CCPRxL 和 CCPxCON 寄存器的 DCxB<1:0>位。在周期结束(即 PR2 和 TMR2 寄存器发生匹配)前占空比值不会被锁存到 CCPRxH 中。使用 PWM 时, CCPRxH 寄存器是只读的。

公式 11-2 用于计算 PWM 脉冲宽度。

公式 11-3 用于计算 PWM 占空比。

公式 11-2: 脉冲宽度

$$\text{脉冲宽度} = (CCPRxL:DCxB<1:0>) \cdot \\ Tosc \cdot (\text{TMR2 预分频值})$$

公式 11-3: 占空比

$$\text{占空比} = \frac{(CCPRxL:DCxB<1:0>)}{4(PR2 + I)}$$

CCPRxH 寄存器和一个 2 位的内部锁存器用于给 PWM 占空比提供双重缓冲。这种双重缓冲结构非常重要, 它可以避免在 PWM 操作中产生毛刺。

8 位定时器 TMR2 寄存器与 2 位内部系统时钟 (Fosc) 或预分频器的 2 位一起构成 10 位时基。如果 Timer2 预分频比设置为 1:1, 则使用系统时钟。

当 10 位时基与 CCPRxH 和 2 位锁存值匹配时, CCPx 引脚被清零(见图 11-3)。

11.4.3 PWM 分辨率

分辨率决定给定周期的可用占空比数。例如，10位分辨率将可得到1024个不连续的占空比，而8位分辨率将可得到256个不连续的占空比。

当PR2为255时，PWM最大分辨率为10位。分辨率是PR2寄存器值的函数，如公式11-4所示。

公式 11-4: PWM 分辨率

$$\text{分辨率} = \frac{\log[4(PR2 + 1)]}{\log(2)} \text{ 位}$$

注：如果脉冲宽度值比周期长，则指定的PWM引脚将保持不变。

表 11-4: PWM 频率和分辨率示例 (Fosc = 40 MHz)

PWM 频率	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
定时器预分频值 (1、4 和 16)	16	4	1	1	1	1
PR2 值	FFh	FFh	FFh	3Fh	1Fh	17h
最大分辨率 (位)	10	10	10	8	7	6.58

表 11-5: PWM 频率和分辨率示例 (Fosc = 20 MHz)

PWM 频率	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
定时器预分频值 (1、4 和 16)	16	4	1	1	1	1
PR2 值	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
最大分辨率 (位)	10	10	10	8	7	6.6

表 11-6: PWM 频率和分辨率示例 (Fosc = 8 MHz)

PWM 频率	1.22 kHz	4.90 kHz	19.61 kHz	76.92 kHz	153.85 kHz	200.0 kHz
定时器预分频值 (1、4 和 16)	16	4	1	1	1	1
PR2 值	0x65	0x65	0x65	0x19	0x0C	0x09
最大分辨率 (位)	8	8	8	6	5	5

11.4.4 在功耗管理模式下的操作

在休眠模式下，TMR2 寄存器将不会递增，模块状态也不会改变。如果 CCPx 引脚正在驱动一个值，则会继续驱动该值。当器件被唤醒时，TMR2 将从先前状态继续。

在 PRI_IDLE 模式下，主时钟将继续作为 CCP 模块的时钟源，保持不变。在所有其他功耗管理模式下，选定的功耗管理模式时钟将作为 Timer2 的时钟源。其他功耗管理模式时钟很可能与主时钟频率不同。

11.4.5 改变系统时钟频率

PWM 频率来自于系统时钟频率。系统时钟频率的任何改变将导致 PWM 频率的改变。更多详细信息，请参见第 2.0 节“振荡器模块（带故障保护时钟监视器）”。

11.4.6 复位的影响

任何复位都将强制所有端口为输入模式，并强制 CCP 寄存器为其复位状态。

11.4.7 设置 PWM 操作

当配置 CCP 模块的 PWM 操作时，可采用以下步骤：

1. 通过将相关的 TRIS 位置 1，禁止 PWM 引脚 (CCPx) 输出驱动器。
2. 仅针对 ECCP 模块：通过将 PSTRCON 寄存器的相应转向位置 1，选择所需的 PWM 输出 (P1A 到 P1D)。
3. 通过装入 PR2 寄存器设置 PWM 周期。
4. 通过将适当的值装入 CCPxCON 寄存器，将 CCP 模块配置为 PWM 模式。
5. 通过装入 CCPRxL 寄存器和 CCPxCON 寄存器的 CCPx 位设置 PWM 占空比。
6. 配置和启动 Timer2：
 - 清零 PIR1 寄存器的 TMR2IF 中断标志位。
 - 通过装入 T2CON 寄存器的 T2CKPS 位设置 Timer2 预分频值。
 - 通过将 T2CON 寄存器的 TMR2ON 位置 1 使能 Timer2。
7. 在新的 PWM 周期开始后使能 PWM 输出：
 - 等待直到 Timer2 溢出 (PIR1 寄存器的 TMR2IF 位置 1)。
 - 通过清零相关的 TRIS 位使能 CCPx 引脚输出驱动器。

表 11-7：与 PWM 和 TIMER2 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值 所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMROIE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	SBOREN	—	RI	TO	PD	POR	BOR	58
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
TRISB	PORTB 数据方向控制寄存器								62
TRISC	PORTC 数据方向控制寄存器								62
TMR2	Timer2 寄存器								60
PR2	Timer2 周期寄存器								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								61
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								61
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
CCPR2L	捕捉 / 比较 /PWM 寄存器 2 的低字节								61
CCPR2H	捕捉 / 比较 /PWM 寄存器 2 的高字节								61
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	61
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0	61
PWM1CON	PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC	PDC0	61

图注：— = 未实现，读为 0。PWM 或 Timer2 不使用阴影单元。

注 1：在 PIC18F2XK20 器件上未实现。

PIC18F2XK20/4XK20

注:

12.0 TIMER0 模块

Timer0 模块具有以下特性：

- 可由软件选择作为 8 位或 16 位定时器 / 计数器
- 可读写寄存器
- 专用的 8 位软件可编程预分频器
- 可选的时钟源（内部或外部）
- 外部时钟的边沿选择
- 溢出时产生中断

T0CON 寄存器（寄存器 12-1）控制该模块操作的所有方面，包括预分频比的选择。它是可读写的。

图 12-1 给出了 8 位模式下 Timer0 模块的简化框图。图 12-2 给出了 16 位模式下 Timer0 模块的简化框图。

寄存器 12-1： T0CON： TIMER0 控制寄存器

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **TMR0ON:** Timer0 开 / 关控制位

1 = 使能 Timer0

0 = 停止 Timer0

bit 6 **T08BIT:** Timer0 8 位 /16 位控制位

1 = Timer0 被配置为 8 位定时器 / 计数器

0 = Timer0 被配置为 16 位定时器 / 计数器

bit 5 **T0CS:** Timer0 时钟源选择位

1 = T0CKI 引脚上的电平跳变

0 = 内部指令周期时钟（CLKOUT）

bit 4 **T0SE:** Timer0 时钟源边沿选择位

1 = 在 T0CKI 引脚信号从高至低跳变时，递增计数

0 = 在 T0CKI 引脚信号从低至高跳变时，递增计数

bit 3 **PSA:** Timer0 预分频器分配位

1 = 未分配 Timer0 预分频器。Timer0 时钟输入不经预分频器分频。

0 = 已分配 Timer0 预分频器。Timer0 时钟输入来自预分频器的输出。

bit 2-0 **T0PS<2:0>:** Timer0 预分频比选择位

111 = 1:256 预分频比

110 = 1:128 预分频比

101 = 1:64 预分频比

100 = 1:32 预分频比

011 = 1:16 预分频比

010 = 1:8 预分频比

001 = 1:4 预分频比

000 = 1:2 预分频比

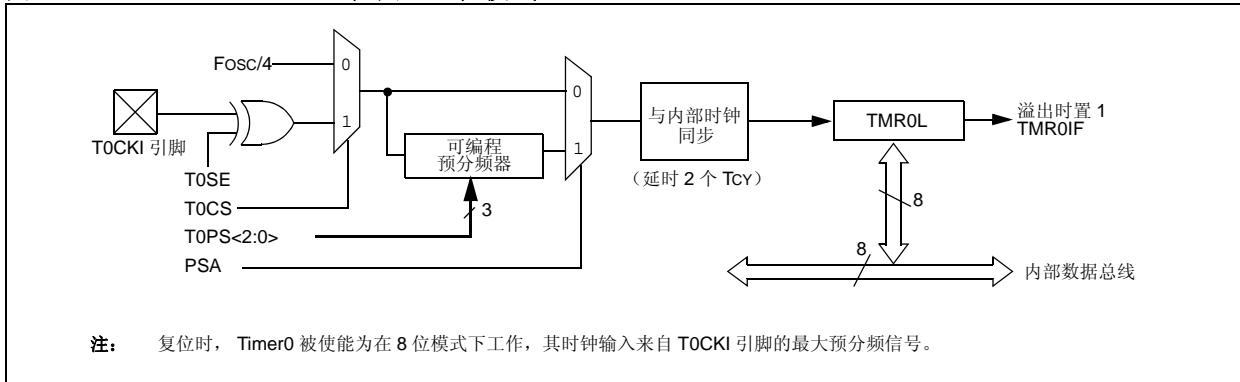
12.1 Timer0 工作原理

Timer0 既可用作定时器也可用作计数器；可以通过 T0CON 寄存器的 T0CS 位来选择模式。在定时器模式 (T0CS = 0) 下，该模块在每个时钟周期都会递增（默认情况下），除非选择了其他预分频值（见第 12.3 节“预分频器”）。在对 TMR0 寄存器执行写操作之后的两个指令周期内禁止 Timer0 递增。通过调整写入 TMR0 寄存器的值来补偿这两个指令周期内错过的递增，用户可以解决这一问题。

通过将 T0CS 位置 1 (= 1) 选择计数器模式。在该模式下，Timer0 可在 RA4/T0CKI 引脚信号的每个上升沿或下降沿递增。递增边沿由 T0CON 寄存器的 Timer0 时钟源边沿选择位 T0SE 决定，清零该位即选择上升沿。下面讨论外部时钟输入的限制条件。

可以使用外部时钟源来驱动 Timer0；但是，必须满足一定要求（见表 26-11），以确保外部时钟和内部相位时钟 (TOSC) 保持同步。在同步之后，定时器 / 计数器需要一定的延时才开始递增。

图 12-1: TIMER0 框图 (8 位模式)

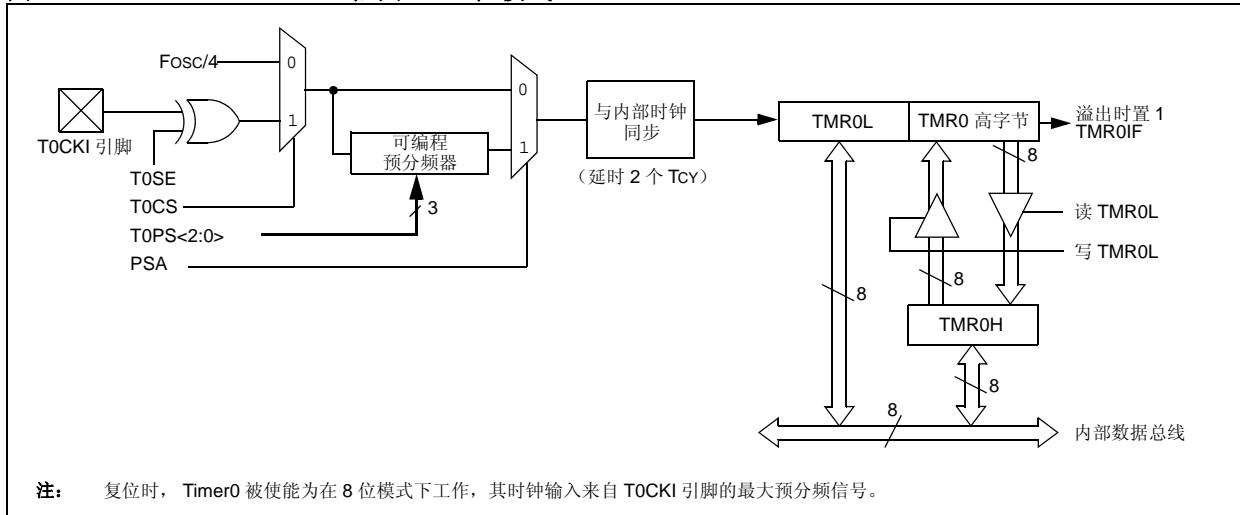


12.2 16 位模式下 Timer0 的读写操作

TMR0H 并不是 16 位模式下 Timer0 的实际高字节，而是 Timer0 实际高字节的缓存形式，不可以被直接读写（见图 12-2）。在读 TMR0L 时使用 Timer0 高字节的内容更新 TMR0H。这种方式使用户可以读取 Timer0 的全部 16 位，而不需要验证高字节和低字节读取的有效性。由于高字节和低字节连续读取之间的计满返回，可能会产生无效读取。

同样，写入 Timer0 的高字节也必须通过 TMR0H 缓冲寄存器来操作。写入 TMR0H 不会直接影响 Timer0。而是在写入 TMR0H 的同时，使用 TMR0H 的内容更新 Timer0 的高字节。这样一次就可以完成 Timer0 全部 16 位的更新。

图 12-2: TIMER0 框图 (16 位模式)



12.3 预分频器

Timer0 模块的预分频器为一个 8 位计数器。该预分频器不可直接读写；通过 PSA 和 T0CON 寄存器的 T0PS<2:0> 位进行预分频器的分配和设定预分频比。

将 PSA 位清零可将预分频器分配给 Timer0 模块。如果已经分配了预分频器，预分频值可在 1:2 到 1:256 之间进行选择，以 2 的整数次幂递增。

如果将预分频器分配给 Timer0 模块，所有写入 TMR0 寄存器的指令（例如，CLRF TMR0、MOVWF TMR0 和 BSF TMR0 等）都会将预分频器的计数值清零。

注：如果将预分频器分配给 Timer0，写入 TMR0 会将预分频器的计数值清零，但不会改变预分频器的分配。

12.3.1 切换预分频器的分配

预分频器的分配完全由软件控制，并且在程序执行期间可以随时更改。

12.4 Timer0 中断

8 位模式下 TMR0 寄存器从 FFh 溢出到 00h，或 16 位模式下 TMR0 从 FFFFh 溢出到 0000h 时，将产生 TMR0 中断。这种溢出会将 TMR0IF 标志位置 1。可以通过清零 INTCON 寄存器的 TMR0IE 位来屏蔽该中断。在重新允许该中断前，必须在中断服务程序中用软件清零 TMR0IF 位。

由于 Timer0 在休眠模式下是关闭的，所以 TMR0 中断无法将处理器从休眠状态唤醒。

表 12-1: 与 TIMER0 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TMR0L	Timer0 寄存器的低字节								60
TMR0H	Timer0 寄存器的高字节								60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	60
TRISA	RA7 ⁽¹⁾	RA6 ⁽¹⁾	RA5	RA4	RA3	RA2	RA1	RA0	62

图注：Timer0 模块不使用阴影单元。

注 1：PORTA<7:6> 及其方向位根据不同的主振荡器模式被单独配置为端口引脚。当被禁止时，这些位读为 0。

PIC18F2XK20/4XK20

注:

13.0 TIMER1 模块

Timer1 定时器 / 计数器模块具有以下特性：

- 可由软件选择作为 16 位定时器或计数器
- 可读写的 8 位寄存器（TMR1H 和 TMR1L）
- 可选择的内部或外部时钟源以及 Timer1 振荡器选项
- 溢出时产生中断
- CCP 特殊事件触发复位
- 器件时钟状态标志位（T1RUN）

图 13-1 给出了 Timer1 模块的简化框图。图 13-2 给出了此模块在读 / 写模式下的工作原理框图。

此模块自身带有低功耗振荡器，可提供额外的时钟选项。Timer1 振荡器也可作为单片机处于节能状态时的低功耗时钟源。

仅需极少外部元件和代码开销，Timer1 就可为应用提供实时时钟（Real-Time Clock, RTC）功能。

Timer1 由 T1CON 控制寄存器（寄存器 13-1）控制。该寄存器还包含 Timer1 振荡器使能位（T1OSCEN）。可以通过将 T1CON 寄存器的控制位 TMR1ON 置 1 或清零来使能或禁止 Timer1。

寄存器 13-1： T1CON: TIMER1 控制寄存器

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON
bit 7							
bit 0							

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	RD16: 16 位读 / 写模式使能位 1 = 使能 Timer1 通过一次 16 位操作进行寄存器读 / 写 0 = 使能 Timer1 通过两次 8 位操作进行寄存器读 / 写
bit 6	T1RUN: Timer1 系统时钟状态位 1 = 主系统时钟由 Timer1 振荡器产生 0 = 主系统时钟由另一个时钟源产生
bit 5-4	T1CKPS<1:0>: Timer1 输入时钟预分频比选择位 11 = 1:8 预分频比 10 = 1:4 预分频比 01 = 1:2 预分频比 00 = 1:1 预分频比
bit 3	T1OSCEN: Timer1 振荡器使能位 1 = 使能 Timer1 振荡器 0 = 关闭 Timer1 振荡器 关闭振荡器的反相器和反馈电阻以减少功耗。
bit 2	T1SYNC: Timer1 外部时钟输入同步选择位 <u>当 TMR1CS = 1 时：</u> 1 = 不同步外部时钟输入 0 = 同步外部时钟输入 <u>当 TMR1CS = 0 时：</u> 该位为无关位。当 TMR1CS = 0 时， Timer1 使用内部时钟。
bit 1	TMR1CS: Timer1 时钟源选择位 1 = 使用 RC0/T1OSO/T13CKI 引脚上的外部时钟（上升沿计数） 0 = 内部时钟（Fosc/4）
bit 0	TMR1ON: Timer1 使能位 1 = 使能 Timer1 0 = 停止 Timer1

13.1 Timer1 工作原理

Timer1 可工作在以下模式之一：

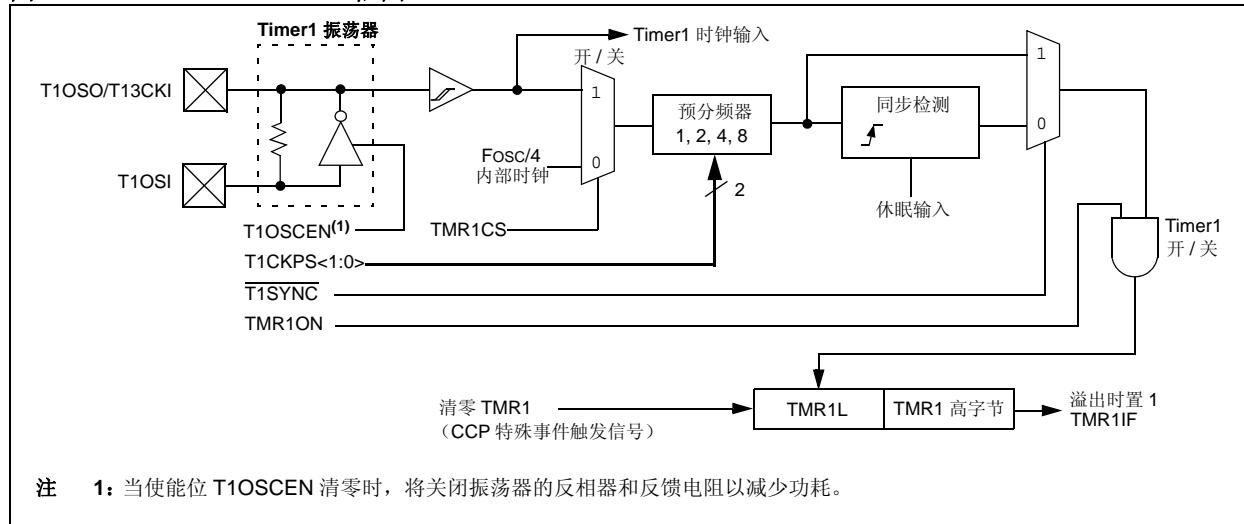
- 定时器
- 同步计数器
- 异步计数器

工作模式由 T1CON 寄存器的时钟选择位 TMR1CS 决定。当 TMR1CS 清零 (= 0) 时，Timer1 在每个内部

指令周期 ($F_{osc}/4$) 递增。当该位置 1 时，Timer1 在 Timer1 外部时钟输入信号或 Timer1 振荡器输出信号（如果使能）的每个上升沿递增。

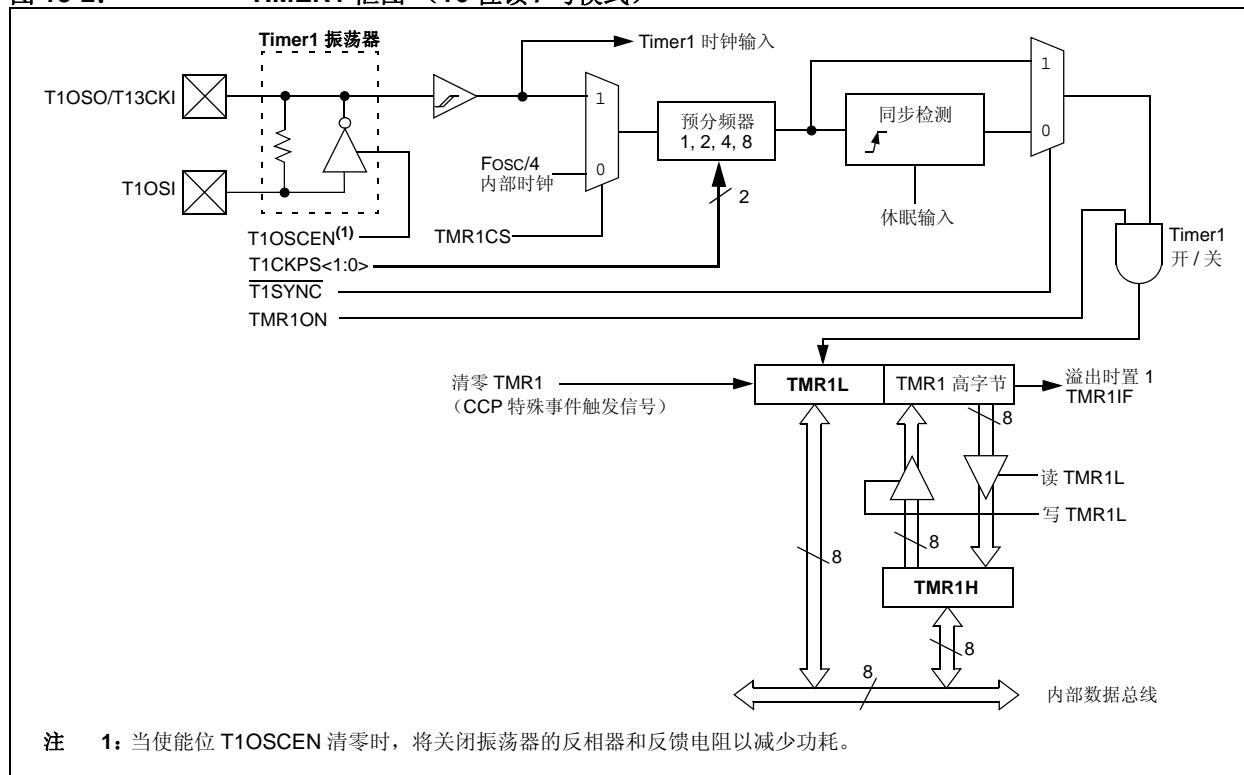
当使能 Timer1 振荡器时，与 RC1/T1OSI 和 RC0/T1OSO/T13CKI 引脚相关的数字电路被禁止。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

图 13-1： TIMER1 框图



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

图 13-2： TIMER1 框图 (16 位读 / 写模式)



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

13.2 时钟源选择

T1CON 寄存器的 TMR1CS 位用于选择时钟源。当 TMR1CS = 0 时，时钟源的频率为 Fosc/4。当 TMR1CS = 1 时，时钟源由外部提供。

13.2.1 内部时钟源

当选择内部时钟源时，TMR1H:TMR1L 寄存器对将在 TCY 的整数倍（由 Timer1 预分频器决定）处递增。

13.2.2 外部时钟源

当选择外部时钟源时，Timer1 模块可以作为定时器或计数器工作。

计数时，Timer1 在外部时钟输入 T1CKI 的上升沿递增。此外，计数器模式时钟可以与单片机系统时钟同步，也可以异步工作。

如果需要使用外部时钟振荡器（并且单片机正在使用不带 CLKOUT 的 INTOSC），则 Timer1 可以将 LP 振荡器用作时钟源。

- 注：** 在计数器模式下，发生以下任何一个或多个情况后，计数器在首个上升沿递增前，必须先经过一个下降沿（见图 13-3）：
- Timer1 在 POR 或 BOR 复位后被使能
 - 写 TMR1H 或 TMR1L
 - T1CKI 为高电平时 Timer1 被禁止（TMR1ON = 0），然后在 T1CKI 为低电平时 Timer1 被使能（TMR1ON = 1）。

13.2.3 在异步计数器模式下读写 TIMER1

当定时器采用外部异步时钟工作时，对 TMR1H 或 TMR1L 的读操作将确保有效（由硬件实现）。但是，应该注意的是，通过读两个 8 位值来读取 16 位定时器本身就会产生某些问题，这是因为定时器可能在两次读操作之间产生溢出。

对于写操作，建议用户直接停止定时器，然后写入需要的值。如果定时器寄存器正进行递增计数，对定时器寄存器进行写操作，可能会导致写入竞争，从而可能在 TMR1H:TTMR1L 寄存器对中产生不可预测的值。

13.3 Timer1 预分频器

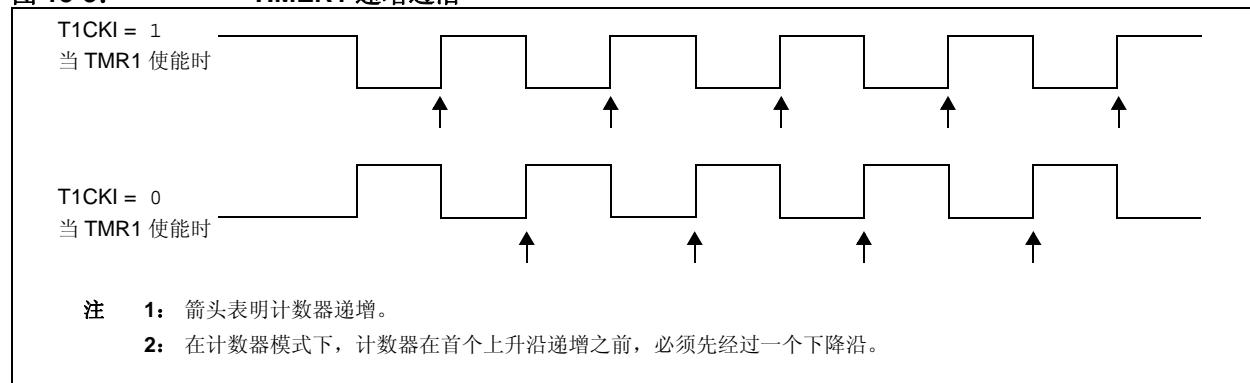
Timer1 有 4 种预分频比选择，允许对时钟输入进行 1、2、4 或 8 分频。T1CON 寄存器的 T1CKPS 位控制预分频器计数器。对预分频器计数器不能直接进行读写操作；但是，通过写入 TMR1H 或 TMR1L 可将预分频器计数器清零。

13.4 Timer1 在异步计数器模式下的工作原理

如果 T1CON 寄存器的控制位 T1SYNC 置 1，外部时钟输入将不同步。定时器继续异步于内部相位时钟进行递增计数。在休眠期间定时器将继续运行，并在溢出时产生中断，溢出中断将唤醒处理器。但是，用软件对定时器进行读 / 写操作时，要特别当心（见第 13.2.3 节“在异步计数器模式下读写 Timer1”）。

- 注 1：** 当从同步切换到异步操作时，可能会跳过一次递增。当从异步切换到同步操作时，可能会产生一次额外递增。

图 13-3：TIMER1 递增边沿



13.5 Timer1 的 16 位读 / 写模式

可将 Timer1 配置为 16 位读写模式（见图 13-2）。当 T1CON 寄存器的 RD16 控制位置 1 时，TMR1H 的地址被映射到 Timer1 的高字节缓冲寄存器。读 TMR1L 将把 Timer1 的高字节的内容装入 Timer1 高字节缓冲寄存器。这种方式使用户可以精确地读取 Timer1 的全部 16 位，而不需要像先读高字节再读低字节那样，由于两次读取之间可能存在计满返回或进位，而不得不验证读取的有效性。

写入 TMR1H 不会直接影响 Timer1。而是在写入 TMR1L 的同时，使用 TMR1H 的内容更新 Timer1 的高字节。这样一次就可以完成 Timer1 全部 16 位的更新。

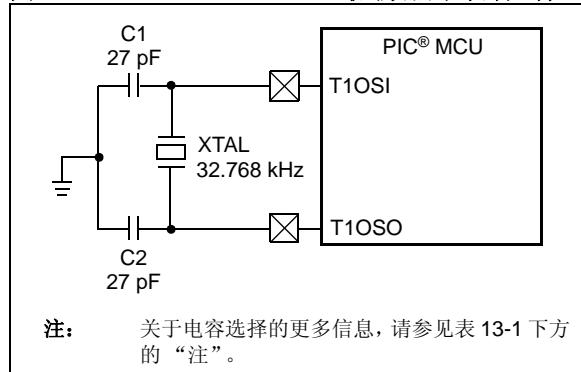
在该模式下不能直接读写 Timer1 的高字节。所有读写都必须通过 Timer1 高字节缓冲寄存器来进行。写入 TMR1H 不会清零 Timer1 预分频器。只有在写 TMR1L 时才会清零该预分频器。

13.6 Timer1 振荡器

片上晶振电路连接在 T1OSI (输入) 引脚和 T1OSO (放大器输出) 引脚之间。可以通过将 T1CON 寄存器的 Timer1 振荡器使能位 T1OSCEN 置 1 来使能该振荡器电路。该振荡器电路是一种低功耗电路，它采用了额定振荡频率为 32 kHz 的晶振。在所有功耗管理模式下都可继续运行。图 13-4 所示为典型的 LP 振荡器电路。表 13-1 给出了供 Timer1 振荡器选择的电容值。

用户必须提供软件延时来确保 Timer1 振荡器的正常起振。

图 13-4: TIMER1 LP 振荡器的外部元件



注：关于电容选择的更多信息，请参见表 13-1 下方的“注”。

表 13-1: 定时器振荡器的电容选择

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

注 1: Microchip 建议仅将这些值作为验证振荡器电路的起始点。

2: 电容越大，振荡器越稳定，但起振时间越长。

3: 因为每种谐振器 / 晶振都有其自身特性，用户应当向谐振器 / 晶振制造厂商询问外部元件的适当值。

4: 上述电容值仅供设计参考。

13.6.1 使用 TIMER1 作为时钟源

在功耗管理模式下也可以将 Timer1 振荡器用作时钟源。通过将 OSCCON 寄存器的时钟选择位 SCS<1:0> 设置为 01，器件可以切换到 SEC_RUN 模式；CPU 和外设都可以用 Timer1 振荡器作为时钟源。如果 OSCCON 寄存器的 IDLEN 位被清零并且执行了 SLEEP 指令，器件将进入 SEC_IDLE 模式。更多详细信息，请参见第 3.0 节“功耗管理模式”。

无论何时将 Timer1 振荡器用作时钟源，T1CON 寄存器的 Timer1 系统时钟状态标志位 T1RUN 均会置 1。这可用于确定控制器的当前时钟模式。该位也可指示故障保护时钟监视器当前正使用的时钟源。如果使能了时钟监视器并且 Timer1 振荡器在提供时钟信号时发生了故障，查询 T1RUN 位可以确定时钟源是 Timer1 振荡器还是其他时钟源。

13.6.2 低功耗 TIMER1 选项

根据器件配置，Timer1 振荡器可以在两种不同的功耗级别下工作。当 CONFIG3H 寄存器的 LPT1OSC 配置位置 1 时，Timer1 振荡器在低功耗模式下工作。当 LPT1OSC 清零时，Timer1 在高功耗模式下工作。不管器件工作在什么模式下，特定模式的功耗都是相对固定的。默认将 Timer1 配置为工作在功耗较高的模式下。

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

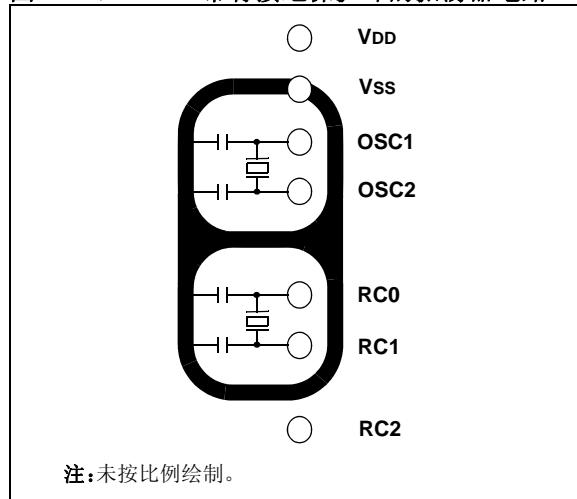
13.6.3 TIMER1 振荡器布线注意事项

Timer1 振荡器电路在工作期间仅消耗极小的电流。鉴于此振荡器的低功耗特性，它对附近变化较快的信号比较敏感。

如图 13-4 所示，振荡器电路应该尽可能靠近单片机。除了 Vss 或 VDD 外，在该振荡器电路边界内不应有其他电路通过。

对于单面 PCB，如果必须要在该振荡器附近布高速电路（如输出比较模式或 PWM 模式下的 CCP1 引脚，或使用 OSC2 引脚的主振荡器），那么在该振荡器电路周围布接地保护环（如图 13-5 所示），或再加一个地平面可能会有帮助。

图 13-5：带有接地保护环的振荡器电路



13.7 Timer1 中断

TMR1 寄存器对 (TMR1H:TMR1L) 从 0000h 递增到 FFFFh，然后计满返回到 0000h 重新开始计数。如果允许了 Timer1 中断，则溢出时会产生 Timer1 中断，锁存在 PIR1 寄存器的 TMR1IF 中断标志位。可以通过将 PIE1 寄存器的 TMR1IE 中断允许位置 1 或清零来允许或禁止该中断。

13.8 使用 CCP 特殊事件触发信号复位 Timer1

如果 CCP 模块配置为在比较模式 (CCP1M<3:0> 或 CCP2M<3:0> = 1011) 下使用 Timer1 并产生特殊事件触发信号，该信号将复位 Timer1。如果使能了 A/D 模块，来自 CCP2 的触发信号还将启动 A/D 转换（更多信息，请参见第 11.3.4 节“特殊事件触发器”）。

要使用这一功能，必须将模块配置为定时器或同步计数器。在这种情况下，CCPRH:CCPRL 寄存器对实际上变成了 Timer1 的周期寄存器。

如果 Timer1 在异步计数器模式下运行，复位操作可能不起作用。

如果对 Timer1 的写操作和特殊事件触发信号同时发生，则写操作优先。

注：CCP2 模块产生的特殊事件触发信号不会将 PIR1 寄存器的 TMR1IF 中断标志位置 1。

13.9 使用 Timer1 作为实时时钟

为 Timer1 外接一个 LP 振荡器（如第 13.6 节“Timer1 振荡器”中所述），可以允许用户在他们的应用中包括 RTC 功能。只需通过一个提供精确时基的廉价时钟晶振以及几行计算时间的应用代码就可实现这一功能。当器件在休眠模式下工作并使用电池或超大容量电容作为电源时，可省去另外的 RTC 器件和备用电池。

应用代码程序 RTCisr（如例 13-1 所示），演示了使用中断服务程序以 1 秒的间隔递增计数器的简单方法。将 TMR1 寄存器对的值递增至溢出将触发中断并调用中断服务程序，该程序会使秒计数器递增 1；其他的分钟和小时计数器则会在前面的计数器溢出时递增 1。

由于寄存器对为 16 位宽，32.768 kHz 时钟源需要 2 秒递增计数到溢出。要强制在所需的 1 秒时间间隔溢出，必须预先加载它；最简单的方法是用 BSF 指令将 TMR1H 的最高有效位置 1。请注意决不要预先加载或改变 TMR1L 寄存器，这样做可能会引起多个周期的累积错误。

要使此方法精确，Timer1 必须工作于异步模式且必须允许 Timer1 溢出中断 (PIE1<0> = 1)，如程序 RTCinit 所示。同时 Timer1 振荡器也必须被使能并始终运行。

例 13-1：使用 TIMER1 中断服务程序实现实时时钟

```
RTCinit
    MOVLW   80h           ; Preload TMR1 register pair
    MOVWF   TMR1H          ; for 1 second overflow
    CLRF    TMR1L
    MOVLW   b'00001111'    ; Configure for external clock,
    MOVWF   T1CON          ; Asynchronous operation, external oscillator
    CLRF    secs            ; Initialize timekeeping registers
    CLRF    mins            ;
    MOVLW   .12             ;
    MOVWF   hours           ;
    BSF    PIE1, TMR1IE     ; Enable Timer1 interrupt
    RETURN

RTCisr
    BSF    TMR1H, 7         ; Preload for 1 sec overflow
    BCF    PIR1, TMR1IF      ; Clear interrupt flag
    INCF   secs, F           ; Increment seconds
    MOVLW   .59              ; 60 seconds elapsed?
    CPFSGT secs
    RETURN                  ; No, done
    CLRF   secs              ; Clear seconds
    INCF   mins, F           ; Increment minutes
    MOVLW   .59              ; 60 minutes elapsed?
    CPFSGT mins
    RETURN                  ; No, done
    CLRF   mins              ; clear minutes
    INCF   hours, F          ; Increment hours
    MOVLW   .23              ; 24 hours elapsed?
    CPFSGT hours
    RETURN                  ; No, done
    CLRF   hours             ; Reset hours
    RETURN                  ; Done
```

表 13-2：与 TIMER1 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
TMR1L	Timer1 寄存器的低字节								60
TMR1H	Timer1 寄存器的高字节								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	60

图注： Timer1 模块不使用阴影单元。

注 1： 这些位在 28 引脚器件上未实现；始终保持这些位清零。

PIC18F2XK20/4XK20

注:

14.0 TIMER2 模块

Timer2 模块定时器具有以下特性：

- 8 位定时器和周期寄存器（分别为 TMR2 和 PR2）
- 可读写（以上两个寄存器）
- 可软件编程的预分频比（分频比为 1:1、1:4 和 1:16）
- 可软件编程的后分频比（分频比为 1:1 到 1:16）
- TMR2 与 PR2 匹配时产生中断
- 可选择用作 MSSP 模块的移位时钟

此模块由 T2CON 寄存器（寄存器 14-1）控制，此寄存器使能或禁止定时器并配置预分频比和后分频比。可以通过清零 T2CON 寄存器的控制位 TMR2ON 关闭 Timer2，以实现功耗最小。

图 14-1 给出了此模块的简化框图。

14.1 Timer2 工作原理

在正常工作模式下，TMR2 从 00h 开始，每个时钟周期 ($F_{osc}/4$) 递增 1。4 位计数器 / 预分频器提供了对时钟输入不分频、4 分频和 16 分频三种预分频选项；可通过 T2CON 寄存器的预分频比控制位 $T2CKPS<1:0>$ 进行选择。在每个时钟周期，TMR2 的值都会与周期寄存器 PR2 中的值进行比较。当两个值匹配时，由比较器产生匹配信号作为定时器的输出。此信号也会将 TMR2 的值在下一个周期复位为 00h，并驱动输出计数器 / 后分频器（见第 14.2 节“Timer2 中断”）。

TMR2 和 PR2 寄存器均可直接读写。在任何器件复位时，TMR2 寄存器都会清零，而 PR2 寄存器则初始化为 FFh。预分频器和后分频器计数器均会在发生以下事件时清零：

- 对 TMR2 寄存器进行写操作
- 对 T2CON 寄存器进行写操作
- 任何器件复位（上电复位、MCLR 复位、看门狗定时器复位或欠压复位）

写 T2CON 时 TMR2 不会清零。

寄存器 14-1： T2CON: TIMER2 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 未实现：读为 0

bit 6-3 **T2OUTPS<3:0>:** Timer2 输出后分频比选择位

0000 = 1:1 后分频比

0001 = 1:2 后分频比

•

•

•

1111 = 1:16 后分频比

bit 2 **TMR2ON:** Timer2 使能位

1 = 使能 Timer2

0 = 关闭 Timer2

bit 1-0 **T2CKPS<1:0>:** Timer2 时钟预分频比选择位

00 = 预分频比为 1

01 = 预分频比为 4

1x = 预分频比为 16

14.2 Timer2 中断

Timer2 也可以产生可选的器件中断。Timer2 输出信号（TMR2 与 PR2 匹配时）为 4 位输出计数器 / 后分频器提供输入。此计数器产生 TMR2 匹配中断，对应的中断标志位为 PIR1 寄存器的 TMR2IF 位。可以通过将 PIE1 寄存器的 TMR2 匹配中断允许位 TMR2IE 置 1 来允许此中断。

可以通过 T2CON 寄存器的后分频比控制位 T2OUTPS<3:0> 在 16 个后分频比选项（从 1:1 到 1:16）中选择其一。

14.3 Timer2 输出

TMR2 的不经分频的输出主要用于 CCP 模块，它用作 CCP 模块在 PWM 模式下工作时的时基。

还可选择将 Timer2 用作 MSSP 模块在 SPI 模式下工作时的移位时钟源。第 17.0 节“主同步串行口（MSSP）模块”中提供了更多信息。

图 14-1: TIMER2 框图

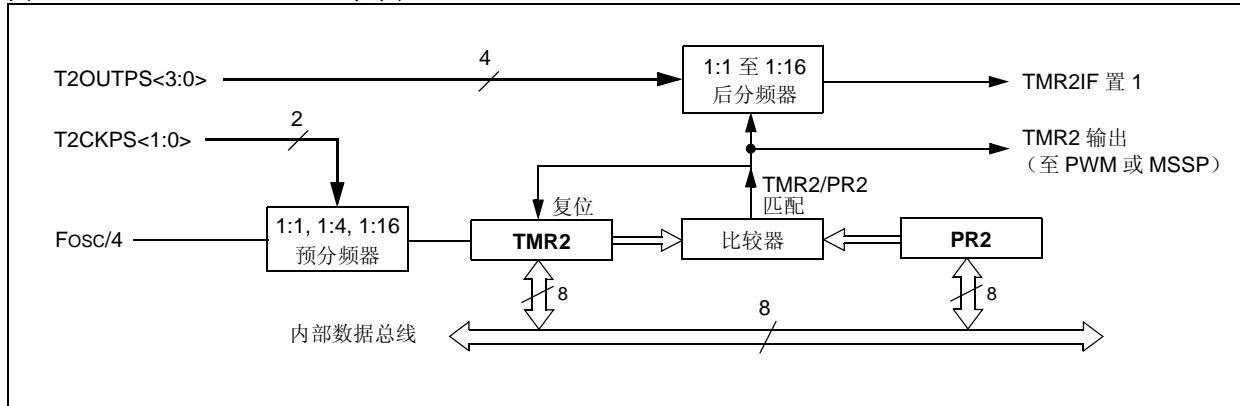


表 14-1: 与 TIMER2 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSP1F ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSP1E ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSP1P ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
TMR2	Timer2 寄存器								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
PR2	Timer2 周期寄存器								60

图注：— = 未实现，读为 0。Timer2 模块不使用阴影单元。

注 1：这些位在 28 引脚器件上未实现；始终保持这些位清零。

15.0 TIMER3 模块

Timer3 定时器 / 计数器模块具有以下特性：

- 可由软件选择作为 16 位定时器或计数器
- 可读写的 8 位寄存器（TMR3H 和 TMR3L）
- 可选择器件时钟或 Timer1 内部振荡器作为时钟源（内部或外部）
- 溢出时产生中断
- CCP 特殊事件触发模块复位

图 15-1 给出了 Timer3 模块的简化框图。图 15-2 给出了此模块在读 / 写模式下的工作原理框图。

Timer3 模块是通过 T3CON 寄存器（寄存器 15-1）来控制的。它还可以为 CCP 模块选择时钟源（更多信息，请参见第 11.1.1 节“CCP 模块和定时器资源”）。

寄存器 15-1： T3CON： TIMER3 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	RD16: 16 位读 / 写模式使能位 1 = 使能 Timer3 通过一次 16 位操作进行寄存器读 / 写 0 = 使能 Timer3 通过两次 8 位操作进行寄存器读 / 写
bit 6,3	T3CCP<2:1>: CCPx 的 Timer3 和 Timer1 使能位 1x = Timer3 是 CCP1 和 CCP2 的捕捉 / 比较时钟源 01 = Timer3 是 CCP2 的捕捉 / 比较时钟源； Timer1 是 CCP1 的捕捉 / 比较时钟源 00 = Timer1 是 CCP1 和 CCP2 的捕捉 / 比较时钟源
bit 5-4	T3CKPS<1:0>: Timer3 输入时钟预分频比选择位 11 = 1:8 预分频比 10 = 1:4 预分频比 01 = 1:2 预分频比 00 = 1:1 预分频比
bit 2	T3SYNC: Timer3 外部时钟输入同步控制位 (不适用于器件时钟来自 Timer1/Timer3 的场合。) <u>当 TMR3CS = 1 时：</u> 1 = 不同步外部时钟输入 0 = 同步外部时钟输入 <u>当 TMR3CS = 0 时：</u> 该位为无关位。当 TMR3CS = 0 时， Timer3 使用内部时钟。
bit 1	TMR3CS: Timer3 时钟源选择位 1 = 使用 Timer1 振荡器或 T13CKI 引脚信号作为外部时钟输入（在第一个下降沿之后的上升沿开始计数） 0 = 内部时钟（Fosc/4）
bit 0	TMR3ON: Timer3 使能位 1 = 使能 Timer3 0 = 停止 Timer3

15.1 Timer3 工作原理

Timer3 可工作在以下三种模式之一：

- 定时器
- 同步计数器
- 异步计数器

工作模式由 T3CON 寄存器的时钟选择位 TMR3CS 决定。当 TMR3CS 清零 (= 0) 时，Timer3 在每个内部指令周期 ($F_{osc}/4$) 递增。当该位置 1 时，Timer3 在 Timer1 外部时钟输入信号或 Timer1 振荡器输出信号（如果使能）的每个上升沿递增。

当使能 Timer1 振荡器时，与 RC1/T1OSI 和 RC0/T1OSO/T13CKI 引脚相关的数字电路被禁止。这意味着 TRISC<1:0> 的值被忽略并且这些引脚将读为 0。

图 15-1：TIMER3 框图

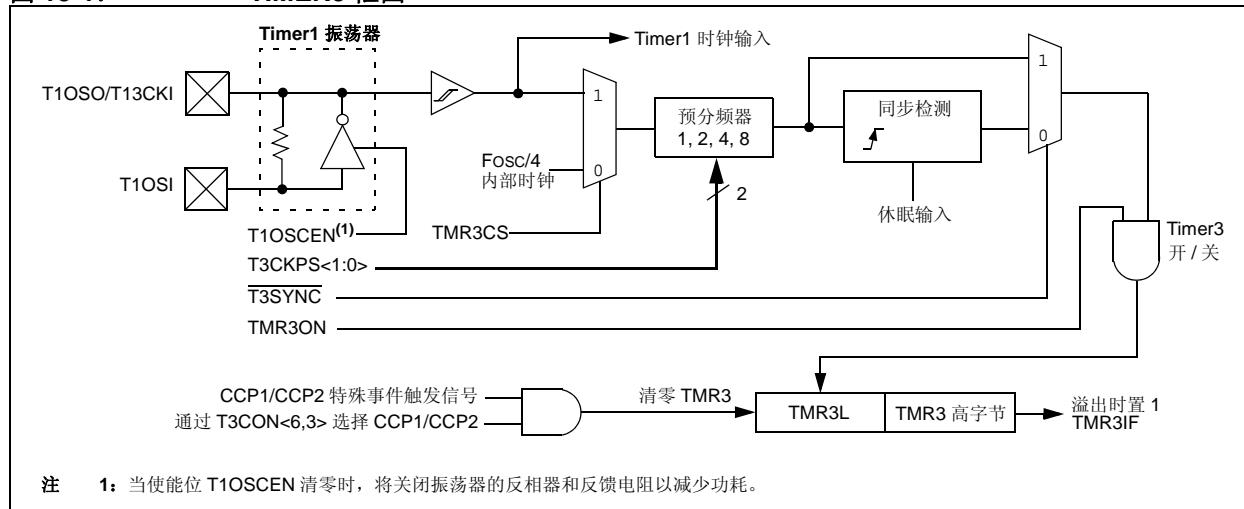
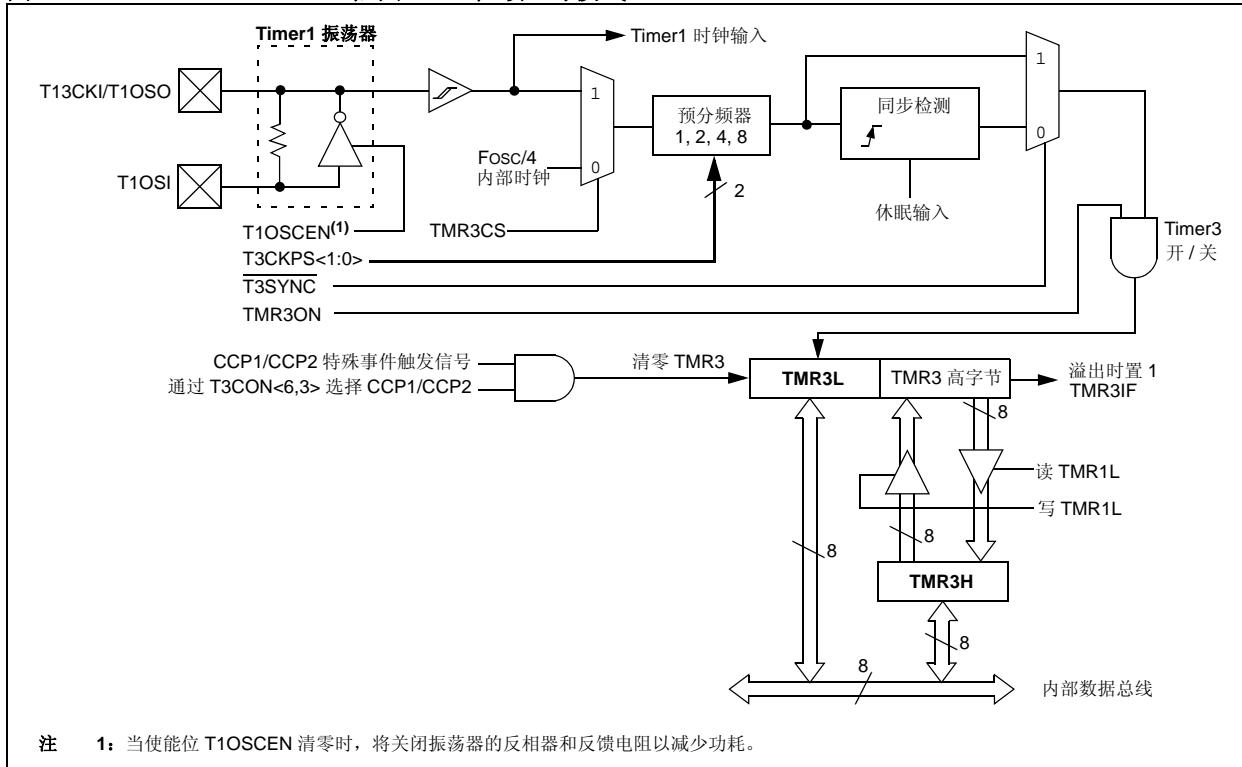


图 15-2: TIMER3 框图 (16 位读 / 写模式)



15.2 Timer3 的 16 位读 / 写模式

可将 Timer3 配置为 16 位读写模式 (见图 15-2)。当 T3CON 寄存器的 RD16 控制位置 1 时, TMR3H 的地址被映射到 Timer3 的高字节缓冲寄存器。读 TMR3L 将把 Timer3 的高字节的内容装入 Timer3 高字节缓冲寄存器。这种方式使用户可以精确地读取 Timer3 的全部 16 位, 而不需要像先读高字节再读低字节那样, 由于两次读取之间可能存在计满返回, 而不得不验证读取的有效性。

写 Timer3 的高字节也必须通过 TMR3H 缓冲寄存器进行。在写入 TMR3L 的同时, 使用 TMR3H 的内容更新 Timer3 的高字节。这样允许用户将所有 16 位一次写入 Timer3 的高字节和低字节。

在该模式下不能直接读写 Timer3 的高字节。所有读写都必须通过 Timer3 高字节缓冲寄存器来进行。

写入 TMR3H 不会清零 Timer3 预分频器。只有在写 TMR3L 时才会清零该预分频器。

15.3 使用 Timer1 振荡器作为 Timer3 的时钟源

Timer1 内部振荡器可用作 Timer3 的时钟源。通过将 T1CON 寄存器的 T1OSCEN 位置 1 可使能 Timer1 振荡器。要将它用作 Timer3 的时钟源, 还必须将 TMR3CS 位置 1。如前文所述, 这样做也会将 Timer3 配置为在振荡器源的每个上升沿递增。

在第 13.0 节 “Timer1 模块” 中对 Timer1 振荡器进行了说明。

15.4 Timer3 中断

TMR3 寄存器对 (TMR3H:TMR3L) 从 0000h 递增到 FFFFh, 然后计满返回到 0000h 重新开始计数。如果允许了 Timer3 中断, 则溢出时会产生 Timer3 中断, 锁存到 PIR2 寄存器的中断标志位 TMR3IF 中。可以通过 PIE2 寄存器的 Timer3 中断允许位 TMR3IE 置 1 或清零来允许或禁止该中断。

15.5 使用 CCP 特殊事件触发信号复位 Timer3

如果 CCP 模块配置为在比较模式（CCP1M<3:0> 或 CCP2M<3:0> = 1011）下使用 Timer3 并产生特殊事件触发信号，该信号将复位 Timer3。如果使能了 A/D 模块，还将启动 A/D 转换（更多信息，请参见第 11.3.4 节“特殊事件触发器”）。

要使用这一功能，必须将模块配置为定时器或同步计数器。在这种情况下，CCPR2H:CCPR2L 寄存器对实际上变成了 Timer3 的周期寄存器。

如果 Timer3 在异步计数器模式下运行，复位操作可能不起作用。

如果 Timer3 的写操作和特殊事件触发同时发生，则写操作优先。

注： CCP2 模块产生的特殊事件触发信号不会将 PIR3 寄存器的 TMR3IF 中断标志位置 1。

表 15-1：与 TIMER3 作为定时器 / 计数器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62
TMR3L	Timer3 寄存器的低字节								61
TMR3H	Timer3 寄存器的高字节								61
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	60
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON	61

图注：— = 未实现，读为 0。Timer3 模块不使用阴影单元。

16.0 增强型捕捉 / 比较 /PWM (ECCP) 模块

CCP1 实现为具有增强型 PWM 功能的标准 CCP 模块。这些功能包括：

- 提供 2 路或 4 路输出通道
- 输出转向
- 可编程极性
- 可编程死区控制
- 自动关闭和重启

第 16.4 节 “PWM (增强型模式)” 将详细讨论增强功能。ECCP 模块的捕捉、比较和单输出 PWM 功能与标准 CCP 模块的相同。

增强型 CCP 模块的控制寄存器如寄存器 16-1 所示。它与 CCP2CON 寄存器的不同之处在于，它的高 2 位用来控制 PWM 功能。

寄存器 16-1： CCP1CON：增强型捕捉 / 比较 /PWM 控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-6

P1M<1:0>：增强型 PWM 输出配置位

如果 CCP1M<3:2> = 00、01 和 10：

xx = P1A 配置为捕捉 / 比较输入 / 输出； P1B、P1C 和 P1D 配置为端口引脚

如果 CCP1M<3:2> = 11：

00 = 单输出：P1A、P1B、P1C 和 P1D 通过转向控制（见第 16.4.7 节“脉冲转向模式”）

01 = 全桥正向输出：P1D 被调制；P1A 有效；P1B 和 P1C 无效

10 = 半桥输出：P1A 和 P1B 被调制，带有死区控制；P1C 和 P1D 配置为端口引脚

11 = 全桥反向输出：P1B 被调制；P1C 有效；P1A 和 P1D 无效

bit 5-4

DC1B<1:0>： PWM 占空比 bit 1 和 bit 0

捕捉模式：

未使用。

比较模式：

未使用。

PWM 模式：

这两位是 10 位 PWM 占空比的低 2 位。占空比的高 8 位在 CCP1L 中。

bit 3-0

CCP1M<3:0>：增强型 CCP 模式选择位

0000 = 捕捉 / 比较 /PWM 关闭（复位 ECCP 模块）

0001 = 保留

0010 = 比较模式，匹配时输出电平翻转

0011 = 保留

0100 = 捕捉模式，每个下降沿

0101 = 捕捉模式，每个上升沿

0110 = 捕捉模式，每 4 个上升沿

0111 = 捕捉模式，每 16 个上升沿

1000 = 比较模式，初始化 CCP1 引脚为低电平，比较匹配时输出置 1（CCP1IF 置 1）

1001 = 比较模式，初始化 CCP1 引脚为高电平，比较匹配时输出清零（CCP1IF 置 1）

1010 = 比较模式，仅产生软件中断，CCP1 引脚恢复到 I/O 状态

1011 = 比较模式，触发特殊事件（ECCP 复位 TMR1 或 TMR3, CC1IF 位置 1）

1100 = PWM 模式：P1A 和 P1C 高电平有效；P1B 和 P1D 高电平有效

1101 = PWM 模式：P1A 和 P1C 高电平有效；P1B 和 P1D 低电平有效

1110 = PWM 模式：P1A 和 P1C 低电平有效；P1B 和 P1D 高电平有效

1111 = PWM 模式：P1A 和 P1C 低电平有效；P1B 和 P1D 低电平有效

除了可使用 CCP1CON 寄存器和 ECCP1AS 寄存器提供的扩展模式外，ECCP 模块还有两个与增强型 PWM 操作和自动关闭功能相关的寄存器。它们是：

- PWM1CON（死区延时）
- PSTRCON（输出转向）

16.1 ECCP 输出和配置

取决于所选定的工作模式，增强型 CCP 模块最多有 4 路 PWM 输出。这些指定为 P1A 到 P1D 的输出，可以与 PORTC 和 PORTD（对于 PIC18F4XK20 器件）或 PORTB（对于 PIC18F2XK20 器件）的 I/O 引脚复用。输出根据选定的 CCP 工作模式被激活。表 16-1 中总结了引脚分配。

若要将 I/O 引脚配置为 PWM 输出，必须通过设置 P1M<1:0> 和 CCP1M<3:0> 位来选择适当的 PWM 模式。端口引脚相应的 TRISC 和 TRISD 方向位也必须设置为输出。

16.1.1 ECCP 模块和定时器资源

与标准的 CCP 模块一样，ECCP 模块根据选定的模式使用 Timer1、Timer2 或 Timer3。该模块在捕捉或比较模式下使用 Timer1 和 Timer3，而在 PWM 模式下使用 Timer2。标准和增强型 CCP 模块之间的相互关系与标准 CCP 模块之间的相同。**第 11.1.1 节“CCP 模块和定时器资源”** 给出了定时器资源的更多详细信息。

16.2 捕捉和比较模式

除了下面讨论的特殊事件触发器的操作，ECCP 模块的捕捉和比较模式与 CCP2 的操作是相同的。这些已在**第 11.2 节“捕捉模式”** 和 **第 11.3 节“比较模式”** 中有详细讨论。当在 28 引脚和 40/44 引脚器件之间移植时不需要任何更改。

16.2.1 特殊事件触发器

ECCP1 的特殊事件触发器输出会复位 TMR1 或 TMR3 寄存器对，具体复位哪一对寄存器，视当前选定的定时器资源而定。这使得 CCPR1 寄存器实际上成为 Timer1 或 Timer3 的 16 位可编程周期寄存器。

16.3 标准 PWM 模式

当配置为单输出模式时，ECCP 模块的功能与 PWM 模式下的标准 CCP 模块相同，如**第 11.4 节“PWM 模式”** 中所述。有时也称为“单 CCP”模式，如表 16-1 所示。

16.4 PWM (增强型模式)

增强型 PWM 模式可以在最多 4 个输出引脚上产生 PWM 信号，最高可达 10 位分辨率。可通过四种不同的 PWM 输出模式实现：

- 单 PWM
- 半桥 PWM
- 全桥 PWM, 正向模式
- 全桥 PWM, 反向模式

要选择增强型 PWM 模式，CCP1CON 寄存器的 P1M 位必须适当置 1。

注： PWM 增强型模式仅在增强型捕捉 / 比较 / PWM 模块 (CCP1) 上可用。

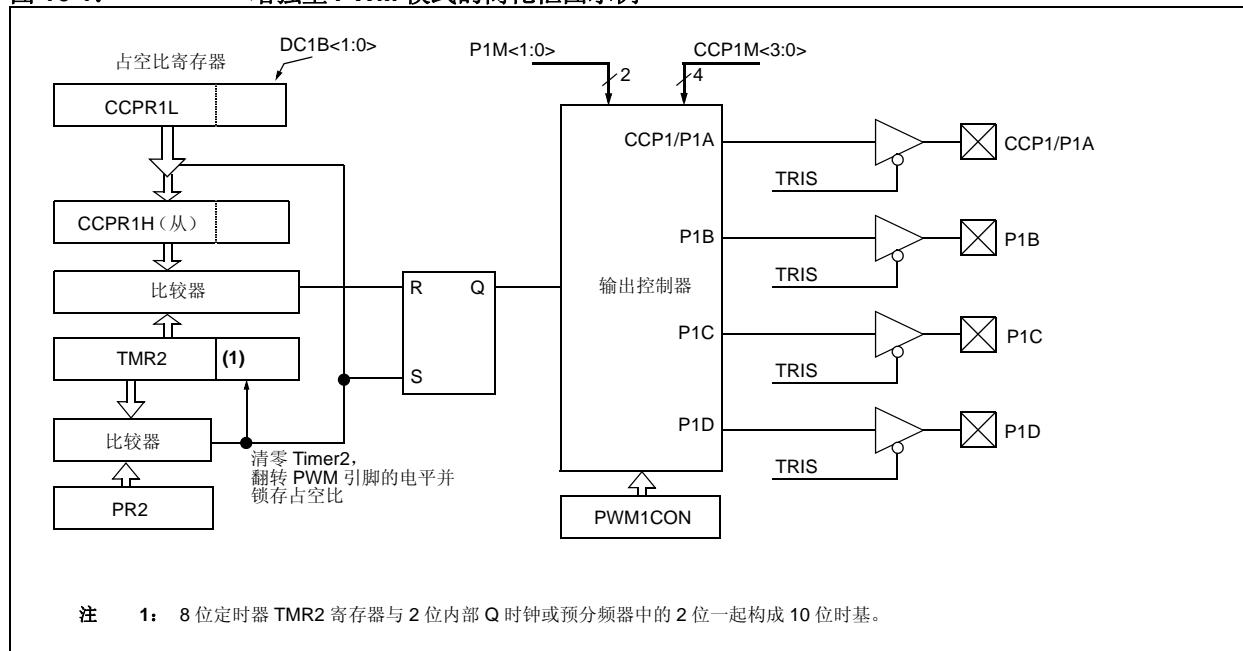
PWM 输出与 I/O 引脚复用，指定为 P1A、P1B、P1C 和 P1D。PWM 引脚的极性是可配置的，通过将 CCP1CON 寄存器中的 CCP1M 位适当置 1 来选择。

表 16-1 给出了每种增强型 PWM 模式的引脚分配。

图 16-1 给出了增强型 PWM 模块的简化框图的示例。

注： 为防止在最开始使能 PWM 时产生不完整的波形，ECCP 模块在产生 PWM 信号前会等待直到新的 PWM 周期开始。

图 16-1：增强型 PWM 模式的简化框图示例



注 1： 每个 PWM 输出的 TRIS 寄存器值必须适当配置。

2： 清零 CCPxCON 寄存器将放弃对所有 PWM 输出引脚的 ECCP 控制。

3： 增强型 PWM 模式没有使用的任何引脚均可用于备用引脚功能。

表 16-1：各种 PWM 增强型模式的引脚分配示例

ECCP 模式	P1M<1:0>	CCP1/P1A	P1B	P1C	P1D
单 PWM	00	使用 (1)	使用 (1)	使用 (1)	使用 (1)
半桥	10	使用	使用	不使用	不使用
全桥, 正向	01	使用	使用	使用	使用
全桥, 反向	11	使用	使用	使用	使用

注 1： 在单输出模式下，输出通过脉冲转向使能。请参见寄存器 16-4。

PIC18F2XK20/4XK20

图 16-2: PWM (增强型模式) 输出关系示例 (高电平有效状态)

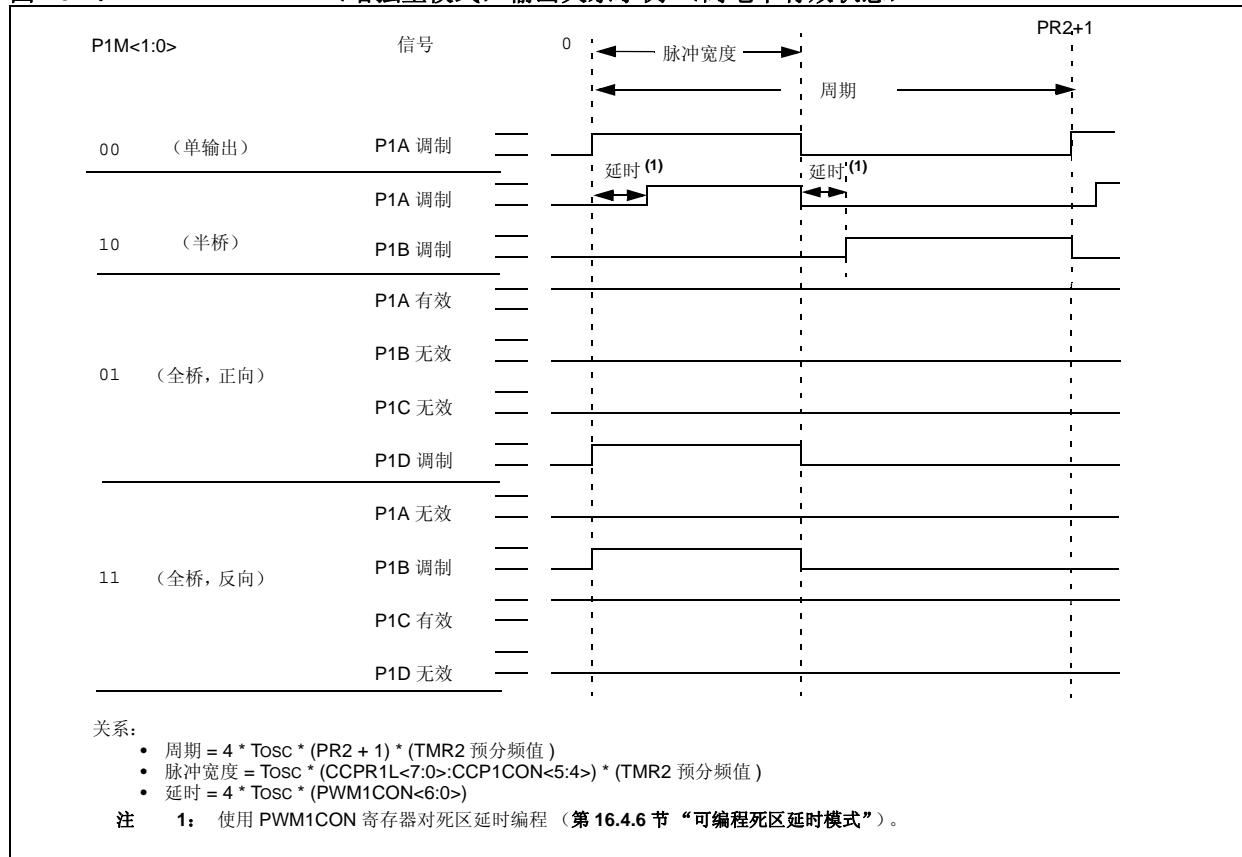
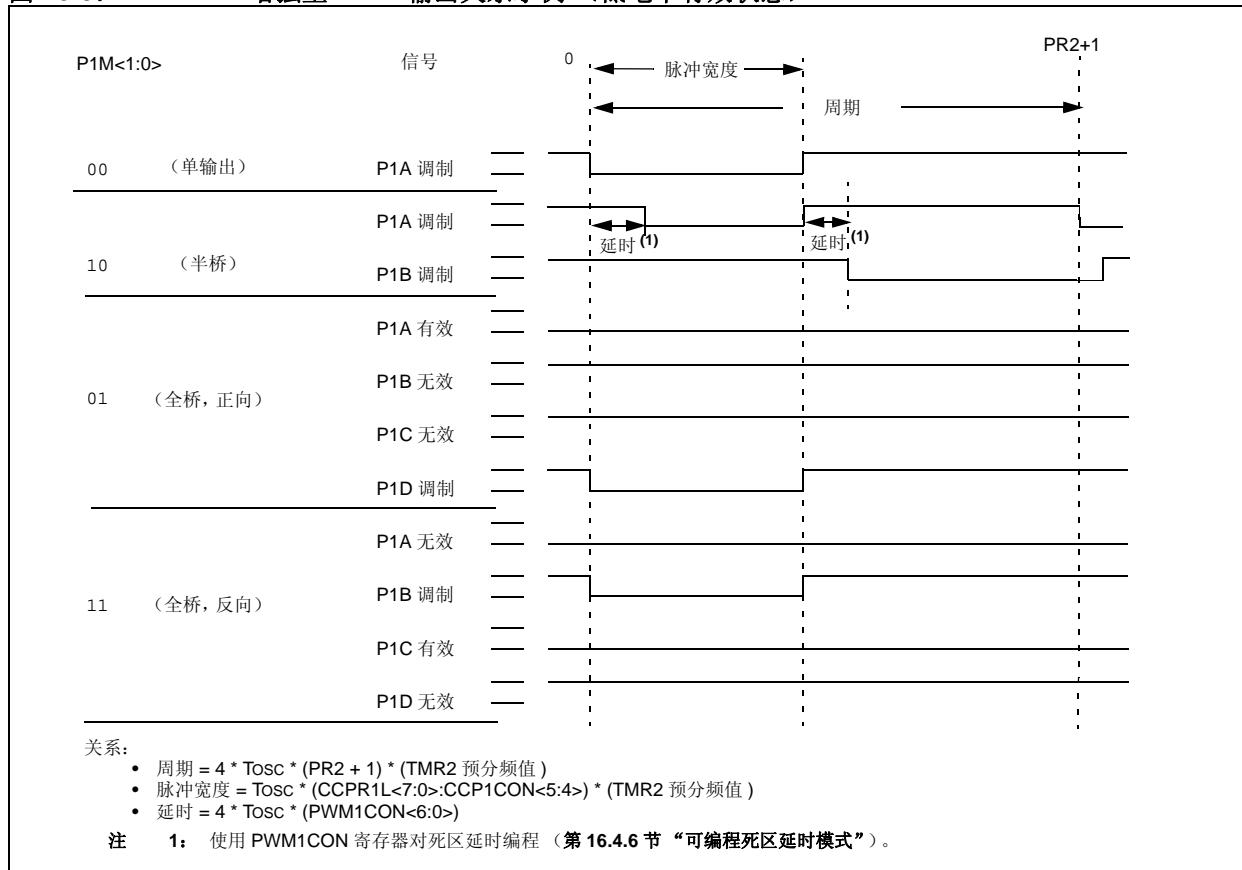


图 16-3：增强型 PWM 输出关系示例（低电平有效状态）



16.4.1 半桥模式

在半桥模式下，有两个引脚用作输出驱动推挽式负载。CCPx/P1A 引脚输出 PWM 输出信号，P1B 引脚输出互补的 PWM 输出信号（见图 16-5）。这种模式可用于半桥应用（如图 16-5 所示），或者用于全桥应用，在全桥应用中使用两个 PWM 信号调制 4 个功率开关。

在半桥模式下，可编程死区延时可用来防止半桥功率器件中流过直通（Shoot-through）电流。PWM1CON 寄存器的 PDC<6:0> 位的值设置在输出被驱动为有效之前的指令周期数。如果这个值比占空比大，则在整个周期中相应的输出保持为无效。关于死区延时操作的更多详细信息，请参见第 16.4.6 节“可编程死区延时模式”。

由于 P1A 和 P1B 输出与端口数据锁存器是复用的，相关的 TRIS 位必须清零，从而将 P1A 和 P1B 配置为输出。

图 16-4：半桥 PWM 输出示例

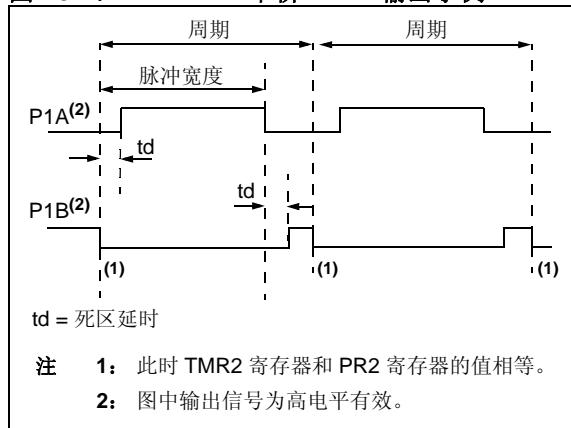
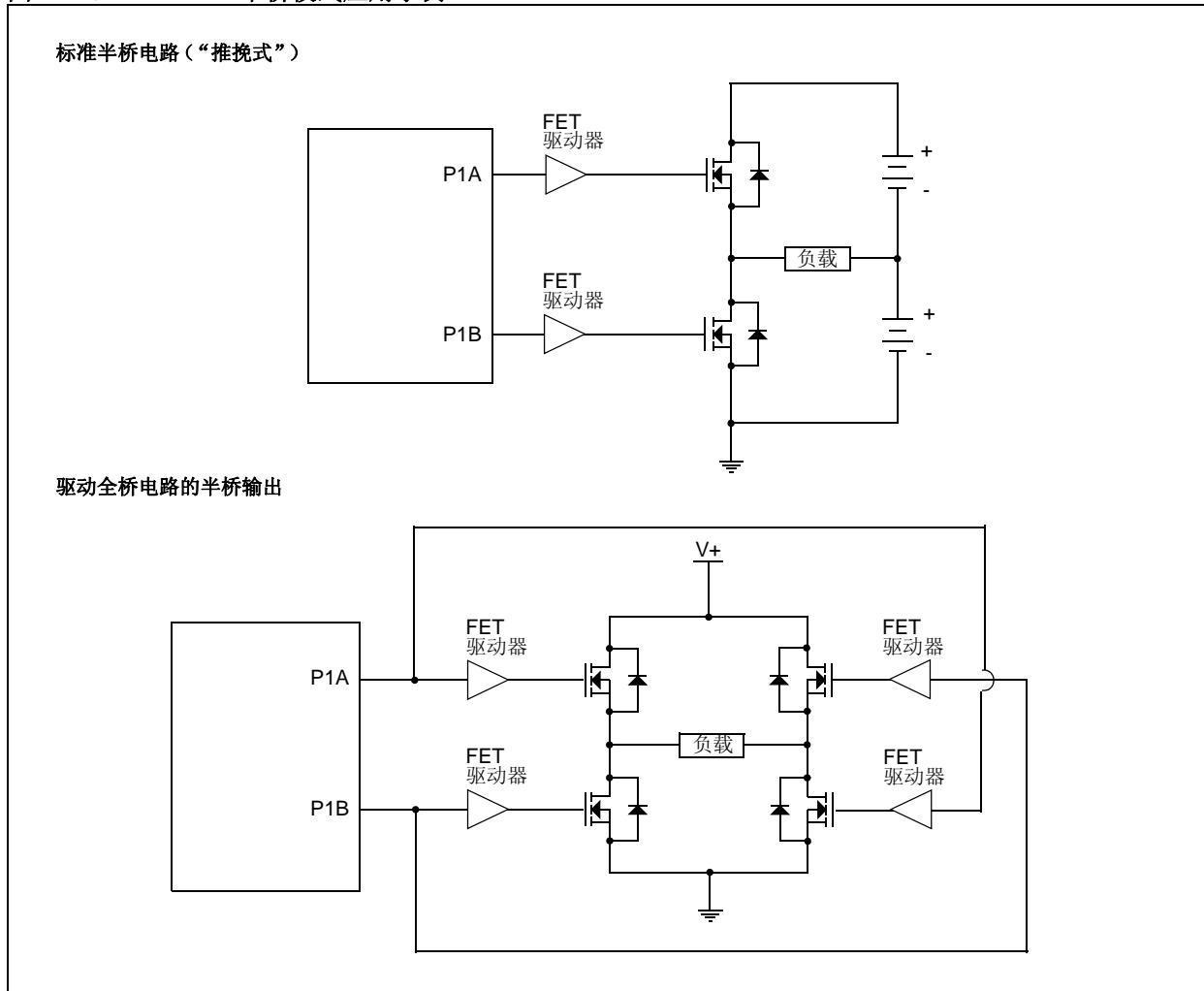


图 16-5：半桥模式应用示例



16.4.2 全桥模式

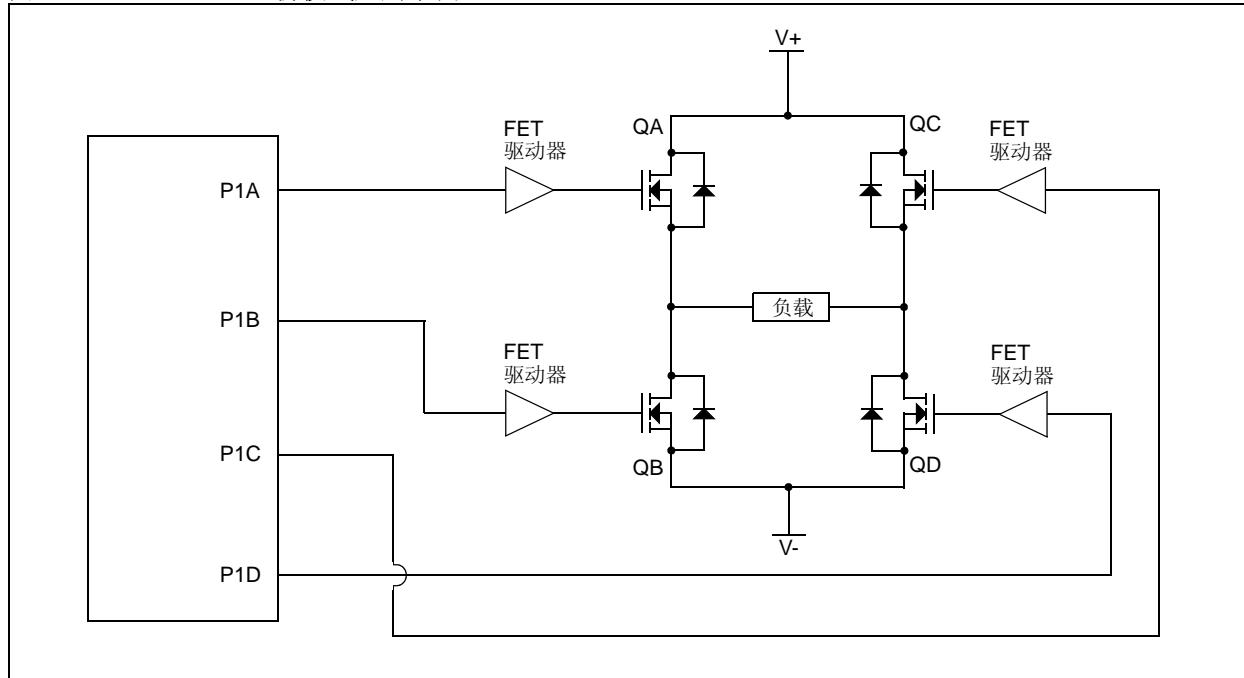
在全桥模式下，4个引脚都用作输出。全桥应用的示例如图16-6所示。

在正向模式下，引脚 CCP1/P1A 被驱动为有效状态，引脚 P1D 被调制，而 P1B 和 P1C 将被驱动为无效状态，如图16-7所示。

在反向模式下，P1C 被驱动为有效状态，引脚 P1B 被调制，而 P1A 和 P1D 将被驱动为无效状态，如图16-7所示。

P1A、P1B、P1C 和 P1D 输出与端口数据锁存器复用。相关的 TRIS 位必须清零，从而将 P1A、P1B、P1C 和 P1D 引脚配置为输出。

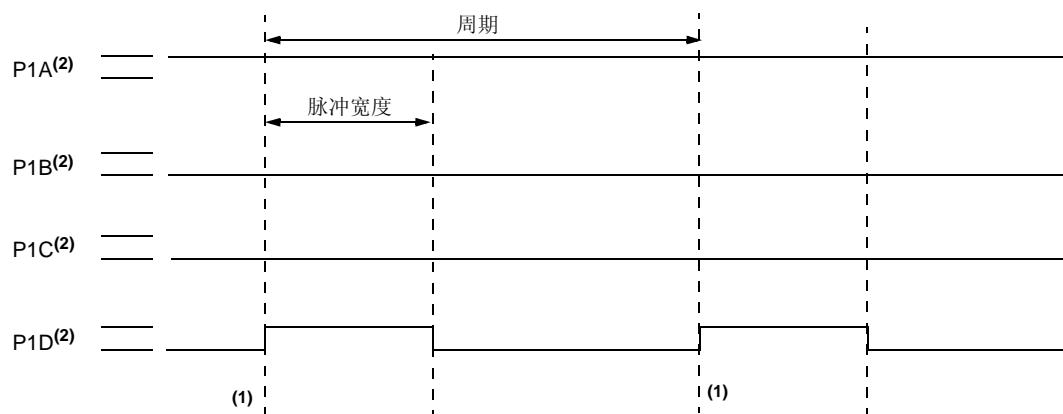
图 16-6：全桥模式应用示例



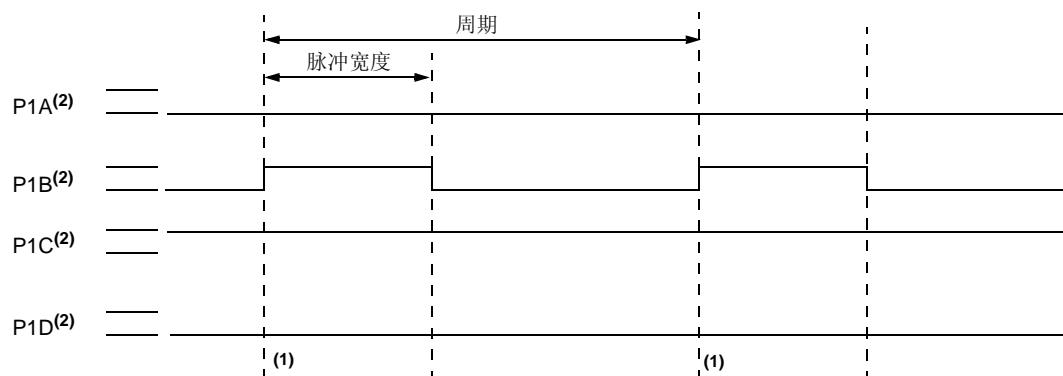
PIC18F2XK20/4XK20

图 16-7：全桥 PWM 输出示例

正向模式



反向模式



注 1: 此时 TMR2 寄存器和 PR2 寄存器的值相等。

2: 图中输出信号为高电平有效。

16.4.2.1 全桥模式中的方向改变

在全桥模式下，CCP1CON 寄存器中的 P1M1 位允许用户控制正/反方向。当应用软件改变这个方向控制位时，模块将在下一个 PWM 周期改新的方向。

通过改变 CCP1CON 寄存器的 P1M1 位，可以用软件启动方向改变。以下序列在当前 PWM 周期结束前发生：

- 调制输出（P1B 和 P1D）进入无效状态。
- 相关的未调制输出（P1A 和 P1C）被切换到以相反的方向驱动。
- PWM 调制在下一个周期开始继续。

关于该序列的说明，请参见图 16-8。

全桥模式不提供死区延时。因为一次只有一个输出被调制，所以一般不需要死区延时。有一种情况需要死区延时。这一情况发生在以下两个条件同时满足时：

1. 当输出的占空比达到或者接近 100%，PWM 输出方向改变。
2. 功率开关（包括功率器件和驱动电路）的关断时间比导通时间要长。

在图 16-9 所示的示例中，在占空比接近 100% 时，PWM 方向从正向改变到反向。在这个示例中，在时间 t1，输出 P1A 和 P1D 变为无效，而输出 P1C 变为有效。因为功率器件的关断时间比导通时间要长，在“t”时间内，功率器件 QC 和 QD 中可能流过直通电流（见图 16-6）。当 PWM 方向从反向改变到正向时，功率器件 QA 和 QB 也将出现相同的现象。

如果应用中需要在高占空比时改变 PWM 方向，避免直通电流可采用以下两种方法：

1. 在改变方向之前的一个 PWM 周期降低 PWM 占空比。
2. 使用开关驱动电路，使开关的关断时间比导通时间短。

也可能存在其他避免直通电流的方案。

图 16-8： PWM 方向改变的示例

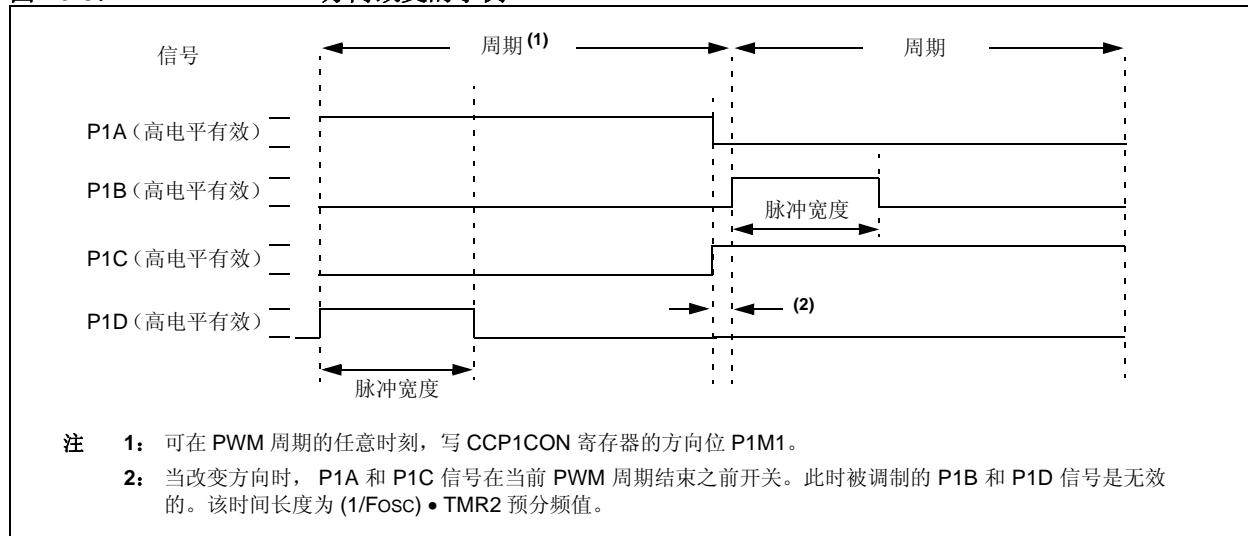
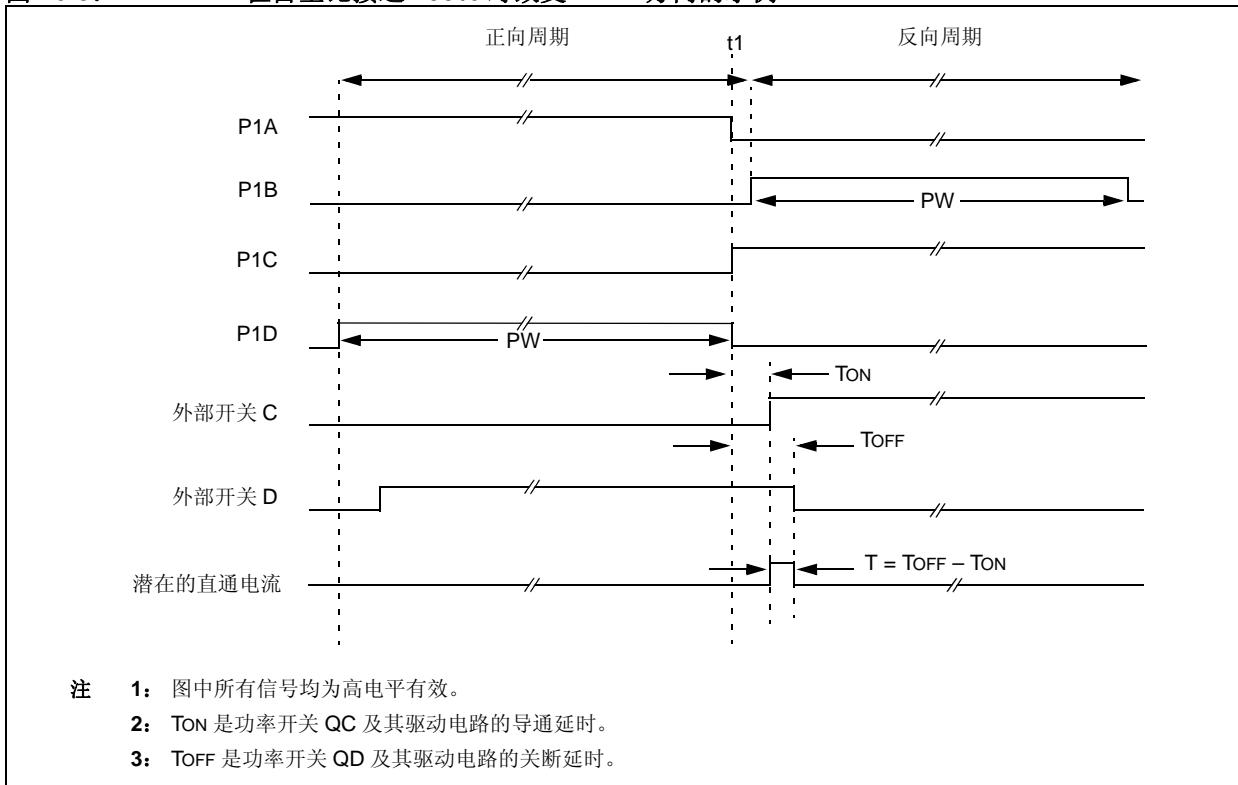


图 16-9：在占空比接近 100% 时改变 PWM 方向的示例



16.4.3 启动注意事项

当使用任何 PWM 模式时，应用硬件必须在 PWM 输出引脚上外接适当的上拉和 / 或下拉电阻。

注：当单片机退出复位状态时，所有 I/O 引脚呈高阻态。外部电路必须保持功率开关器件处于截止状态，直到单片机将 I/O 引脚驱动为适当的信号电平，或者激活 PWM 输出为止。

CCP1CON 寄存器的 CCP1M<1:0> 位允许用户为每一对 PWM 输出引脚（P1A/P1C 和 P1B/P1D）选择 PWM 输出信号是高电平有效还是低电平有效。PWM 输出极性必须在使能 PWM 引脚输出驱动器之前选择。不推荐在使能 PWM 引脚输出驱动器时改变极性配置，因为这样做可能导致应用电路损坏。

当 PWM 模块初始化时，P1A、P1B、P1C 和 P1D 输出锁存器可能不为正确的状态。这样在配置为增强型 PWM 模式的同时使能 PWM 引脚输出驱动器，可能损坏应用电路。应首先将增强型 PWM 模式配置为正确的输出模式并经过一个完整的 PWM 周期之后，再使能 PWM 引脚输出驱动器。当第二个 PWM 周期开始时，PIR1 寄存器的 TMR2IF 位置 1 表明一个完整的 PWM 周期结束了。

16.4.4 增强型 PWM 自动关闭模式

PWM模式支持自动关闭模式，当外部关闭事件发生时将禁止 PWM 输出。自动关闭模式将 PWM 输出引脚置于预先确定的状态。该模式用于防止过大的 PWM 脉冲损坏应用。

通过使用 ECCP1AS 寄存器的 ECCPAS<2:0> 位来选择自动关闭源。关闭事件由以下条件产生：

- FLTO 引脚上的逻辑 0
- 比较器 C1
- 比较器 C2
- 用固件将 ECCPASE 位置 1

关闭条件由 ECCP1AS 寄存器的 ECCPASE（自动关闭事件状态）位指示。如果该位为 0， PWM 引脚正常工作。如果该位为 1， PWM 输出处于关闭状态。

当关闭事件发生时，会发生以下两个事件：

ECCPASE 位被设为 1。ECCPASE 将保持置 1 直到由固件清零或发生自动重启（见第 16.4.5 节“自动重启模式”）。

能使的 PWM 引脚被陆续置为其关闭状态。PWM 输出引脚被分组为 [P1A/P1C] 和 [P1B/P1D] 对。每对引脚的状态由 ECCP1AS 寄存器的 PSSAC 和 PSSBD 位决定。每对引脚可以设置为以下 3 种状态之一：

- 驱动逻辑 1
- 驱动逻辑 0
- 三态（高阻态）

寄存器 16-2： ECCP1AS：增强型捕捉 / 比较 /PWM 自动关闭控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7

ECCPASE：ECCP 自动关闭事件状态位

1 = 发生了关闭事件；ECCP 输出为关闭状态
0 = ECCP 输出正常工作

bit 6-4

ECCPAS<2:0>：ECCP 自动关闭源选择位

000 = 禁止自动关闭
001 = 比较器 C1OUT 输出为高电平
010 = 比较器 C2OUT 输出为高电平
011 = 比较器 C1OUT 或 C2OUT 为高电平
100 = FLTO 引脚电压为 VIL
101 = FLTO 引脚电压为 VIL 或比较器 C1OUT 输出为高电平
110 = FLTO 引脚电压为 VIL 或比较器 C2OUT 输出为高电平
111 = FLTO 引脚电压为 VIL，或者比较器 C1OUT 或比较器 C2OUT 为高电平

bit 3-2

PSSACn：引脚 P1A 和 P1C 关闭状态控制位

00 = 驱动引脚 P1A 和 P1C 为 0
01 = 驱动引脚 P1A 和 P1C 为 1
1x = 引脚 P1A 和 P1C 为三态

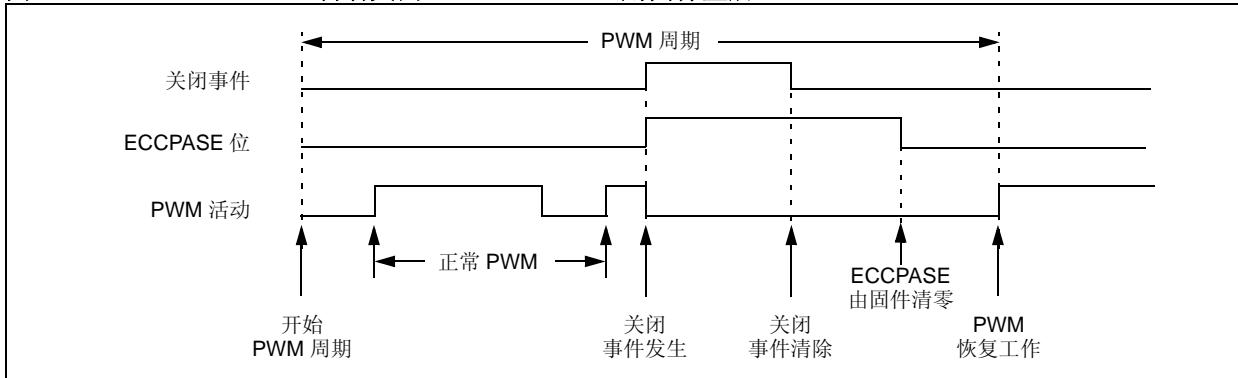
bit 1-0

PSSBDn：引脚 P1B 和 P1D 关闭状态控制位

00 = 驱动引脚 P1B 和 P1D 为 0
01 = 驱动引脚 P1B 和 P1D 为 1
1x = 引脚 P1B 和 P1D 为三态

- 注 1:** 自动关闭条件是基于电平的信号，而不是基于边沿的信号。只要电平存在，自动关闭就将持续。
- 2:** 当自动关闭条件持续时，禁止写 ECCPASE 位。
- 3:** 一旦自动关闭条件被移除并且发生 PWM 重启（通过固件或自动重启），PWM 信号将总是在下一个 PWM 周期重启。

图 16-10: PWM 自动关闭 (PRSEN = 0, 用固件重启)

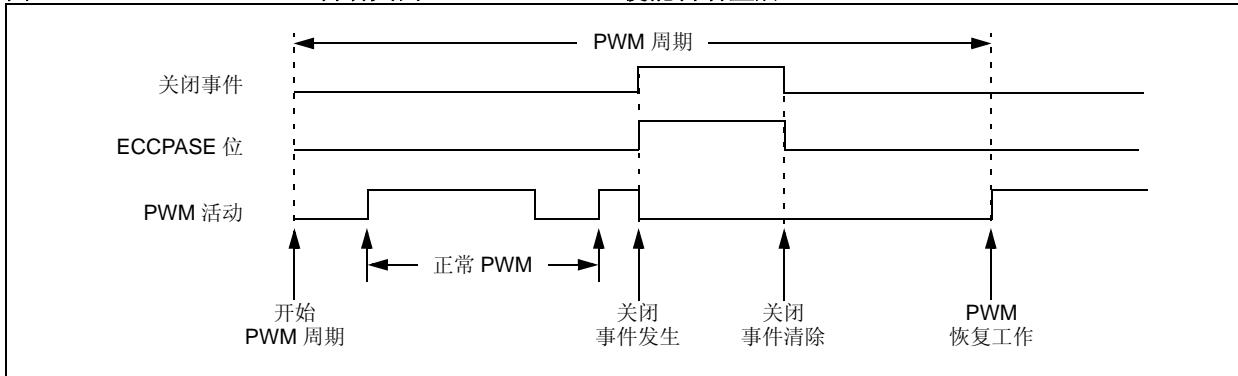


16.4.5 自动重启模式

一旦自动关闭条件被移除，增强型 PWM 可被配置为自动重启 PWM 信号。通过将 PWM1CON 寄存器中的 PRSEN 位置 1 使能自动重启。

如果使能了自动重启，只要自动关闭条件有效，ECCPASE 位将保持置 1。当自动关闭条件被移除时，ECCPASE 位将由硬件清零并恢复正常工作。

图 16-11: PWM 自动关闭 (PRSEN = 1, 使能自动重启)



16.4.6 可编程死区延时模式

在所有功率开关都以 PWM 频率调制的半桥应用中，功率开关关断通常比导通需要更多的时间。如果上下两个功率开关同时开关（一个导通，另一个关断），那么在一段很短的时间里，两个开关可能同时导通，直到其中一个开关完全关断为止。在这短暂的时间中，两个功率开关中可能流过较高的电流（直通电流），将逆变桥的电源与地短路。为避免开关过程中可能会出现的破坏性直通电流，通常需要延迟功率开关的导通，保证在一个开关完全关断之后，再导通相应的功率开关。

在半桥模式下，可采用数字可编程死区延时来避免出现损坏逆变桥功率开关的直通电流。在信号从无效状态切换到有效状态时增加延时。请参见图 16-12。PWM1CON 寄存器（寄存器 16-3）的低 7 位以单片机指令周期（T_{CY} 或 4 T_{Osc}）为单位设置延时。

图 16-12: 半桥 PWM 输出示例

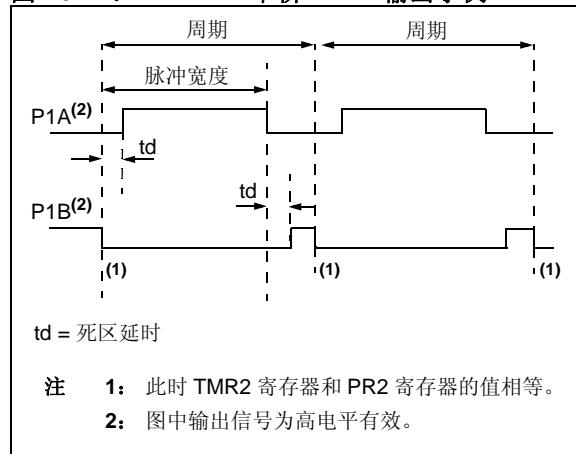
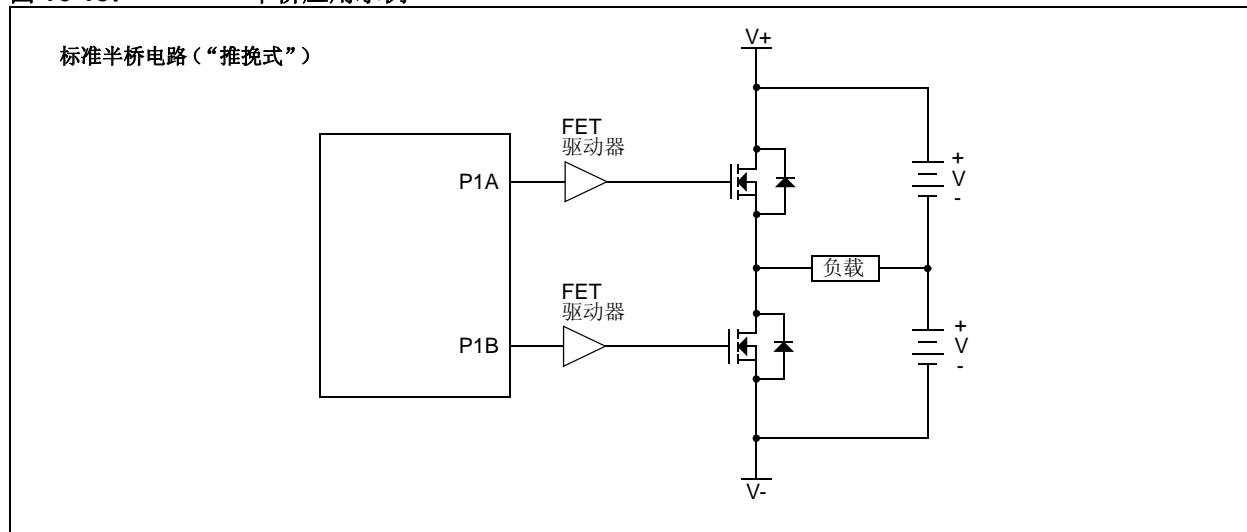


图 16-13: 半桥应用示例



PIC18F2XK20/4XK20

寄存器 16-3: **PWM1CON:** 增强型 PWM 控制寄存器

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| PRSEN | PDC6 | PDC5 | PDC4 | PDC3 | PDC2 | PDC1 | PDC0 |
| bit 7 | bit 0 | | | | | | |

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7

PRSEN: PWM 重启使能位

1 = 自动关闭时, 一旦关闭事件消失, ECCPASE 位自动清零; PWM 自动重启

0 = 自动关闭时, ECCPASE 必须用软件清零以重启 PWM

bit 6-0

PDC<6:0>: PWM 延时计数位

PDCn = 在 PWM 信号应该转换为有效的预定时间和转换为有效的实际时间之间的 Fosc/4 (4 * Tosc) 周期数

16.4.7 脉冲转向模式

在单输出模式下，脉冲转向允许任何 PWM 引脚为调制信号。此外，多个引脚上可以同时使用同一 PWM 信号。

一旦选择了单输出模式（CCP1CON 寄存器的 CCP1M<3:2> = 11 且 P1M<1:0> = 00），通过将 PSTRCON 寄存器的相应 STR<D:A>位置 1，用户固件可将同一 PWM 信号加到 1、2、3 或 4 个输出引脚，如表 16-1 所示。

注： 必须将相关的 TRIS 位设为输出（0）以便能引脚输出驱动器，从而在引脚上看到 PWM 信号。

当 PWM 转向模式有效时，CCP1CON 寄存器的 CCP1M<1:0>位将为 P1<D:A>引脚选择 PWM 输出极性。

PWM 自动关闭操作也适用于 PWM 转向模式，如第 16.4.4 节“增强型 PWM 自动关闭模式”中所述。自动关闭事件只对使能 PWM 输出的引脚有影响。

寄存器 16-4： PSTRCON：脉冲转向控制寄存器⁽¹⁾

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
—	—	—	STRSYNC	STRD	STRC	STRB	STRA
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-5 未实现：读为 0

bit 4 **STRSYNC：** 转向同步位

1 = 在下一个 PWM 周期发生输出转向更新

0 = 在指令周期边界的开始发生输出转向更新

bit 3 **STRD：** 转向使能位 D

1 = P1D 引脚的 PWM 波形极性受 CCPxM<1:0> 控制

0 = P1D 引脚被分配为端口引脚

bit 2 **STRC：** 转向使能位 C

1 = P1C 引脚的 PWM 波形极性受 CCPxM<1:0> 控制

0 = P1C 引脚被分配为端口引脚

bit 1 **STRB：** 转向使能位 B

1 = P1B 引脚的 PWM 波形极性受 CCPxM<1:0> 控制

0 = P1B 引脚被分配为端口引脚

bit 0 **STRA：** 转向使能位 A

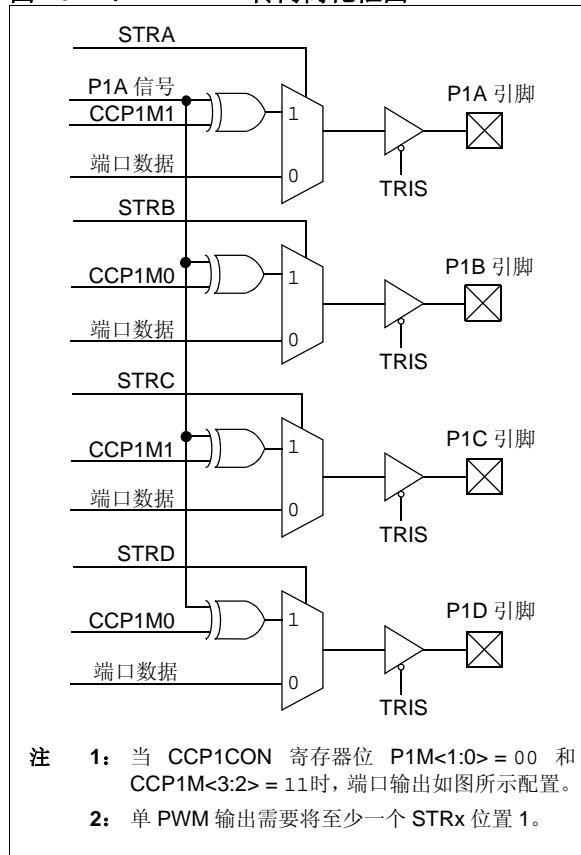
1 = P1A 引脚的 PWM 波形极性受 CCPxM<1:0> 控制

0 = P1A 引脚被分配为端口引脚

注 1： PWM 转向模式仅在 CCP1CON 寄存器位 CCP1M<3:2> = 11 和 P1M<1:0> = 00 时可用。

PIC18F2XK20/4XK20

图 16-14: 转向简化框图



16.4.7.1 转向同步

当转向事件发生时, PSTRCON 寄存器的 STRSYNC 位向用户提供两种选择。当 STRSYNC 位为 0 时, 转向事件将发生在写 PSTRCON 寄存器指令结束时。在这种情况下, P1<D:A> 引脚的输出信号可能是一个不完整的 PWM 波形。用户固件需要立即停止从引脚输出 PWM 信号时, 该操作非常有用。

当 STRSYNC 位为 1 时, 在下一个 PWM 周期的开始将发生有效的转向更新。此时, 转向开 / 关 PWM 输出将始终产生一个完整的 PWM 波形。

图 16-15 和 16-16 是根据 STRSYNC 设置的 PWM 转向时序图。

图 16-15: 指令结束时发生的转向事件的示例 (STRSYNC = 0)

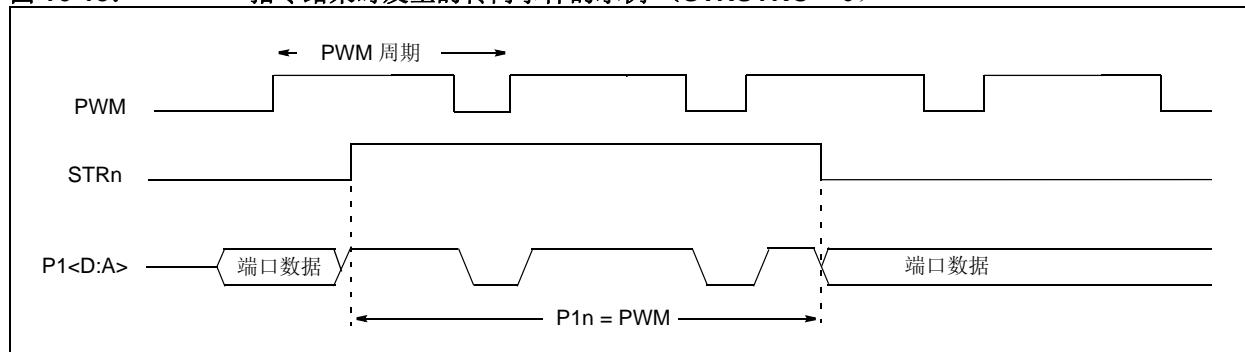
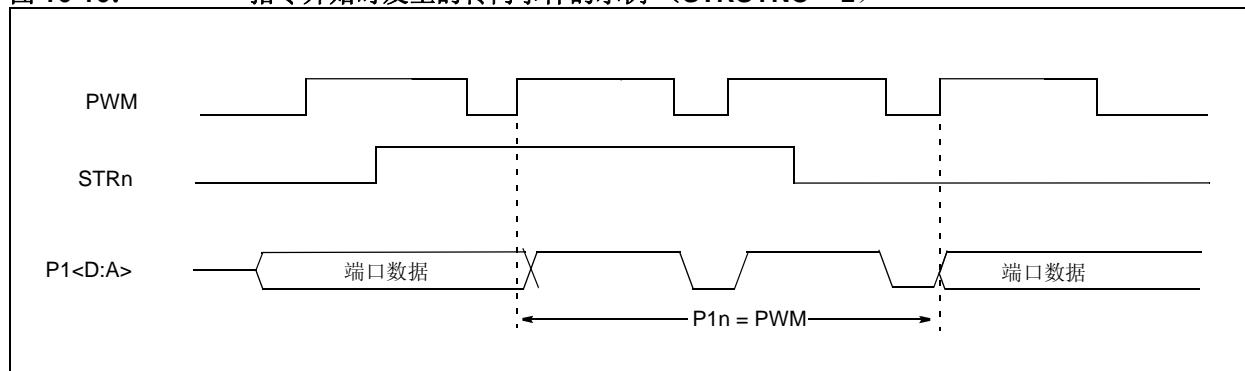


图 16-16: 指令开始时发生的转向事件的示例 (STRSYNC = 1)



16.4.8 在功耗管理模式下的操作

在休眠模式下，所有时钟源都被禁止。Timer2 不再递增，模块的状态也不会改变。如果 ECCP 引脚正在驱动一个值，则会继续驱动该值。当器件被唤醒时，将从该状态继续。如果使能了双速启动，来自 HFINTOSC 和后分频器的初始启动频率可能不会立即稳定。

在 PRI_IDLE 模式下，主时钟将继续作为 ECCP 模块的时钟源，保持不变。在所有其他功耗管理模式下，选定的功耗管理模式时钟将作为 Timer2 的时钟源。功耗管理模式时钟很可能与主时钟频率不同。

16.4.8.1 故障保护时钟监视器相关操作

如果使能了故障保护时钟监视器，时钟故障将强制器件进入 RC_RUN 功耗管理模式，并将 PIR2 寄存器的 OSCFIF 位置 1。ECCP 将从内部振荡器时钟源获取时钟信号，该时钟信号的频率可能与主时钟的时钟频率不同。

更多详细信息，请参见前面的章节。

16.4.9 复位的影响

上电复位及后续的复位都将强制所有端口为输入模式，并强制 CCP 寄存器为复位状态。

这将强制增强型 CCP 模块复位到与标准 CCP 模块兼容的状态。

表 16-2：与 ECCP1 模块和 TIMER1 到 TIMER3 相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMROIE	INTOIE	RBIE	TMR0IF	INT0IF	RBIF	59
RCON	IPEN	SBOREN	—	RI	TO	PD	POR	BOR	58
PIR1	PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62
TRISB	PORTB 数据方向控制寄存器								62
TRISC	PORTC 数据方向控制寄存器								62
TRISD	PORTD 数据方向控制寄存器								62
TMR1L	Timer1 寄存器的低字节								60
TMR1H	Timer1 寄存器的高字节								60
T1CON	RD16	T1RUN	T1CKPS1	T1CKPS0	T1OSCEN	T1SYNC	TMR1CS	TMR1ON	60
TMR2	Timer2 寄存器								60
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	60
PR2	Timer2 周期寄存器								60
TMR3L	Timer3 寄存器的低字节								61
TMR3H	Timer3 寄存器的高字节								61
T3CON	RD16	T3CCP2	T3CKPS1	T3CKPS0	T3CCP1	T3SYNC	TMR3CS	TMR3ON	61
CCPR1L	捕捉 / 比较 /PWM 寄存器 1 的低字节								61
CCPR1H	捕捉 / 比较 /PWM 寄存器 1 的高字节								61
CCP1CON	P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	61
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1	PSSBD0	61
PWM1CON	PRSEN	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0	61

图注：— = 未实现，读为 0。ECCP 操作期间不使用阴影单元。

PIC18F2XK20/4XK20

注:

17.0 主同步串行口（MSSP）模块

17.1 主 SSP（MSSP）模块概述

主同步串行口（MSSP）模块是用于同其他外设或单片机进行通信的串行接口。这些外设可以是串行 EEPROM、移位寄存器、显示驱动器和 A/D 转换器等。MSSP 模块有以下两种工作模式：

- 串行外设接口（Serial Peripheral Interface, SPI）
- I²C
 - 完全的主模式
 - 从模式（支持广播地址呼叫）

I²C 接口硬件上支持以下模式：

- 主模式
- 多主模式
- 从模式

17.2 控制寄存器

MSSP 模块有 7 个相关的寄存器。这些寄存器包括：

- SSPSTA—STATUS 寄存器
- SSPCON1—第一个控制寄存器
- SSPCON2—第二个控制寄存器
- SSPBUF—发送 / 接收缓冲器
- SSPSR—移位寄存器（不可直接访问）
- SSPADD—地址寄存器
- SSPMSK—地址掩码寄存器

根据 MSSP 模块是在 SPI 模式还是 I²C 模式下工作，这些寄存器的用途及它们各自的配置位将完全不同。

下面各节会提供更多详细信息。

17.3 SPI 模式

SPI 模式允许同时同步发送和接收 8 位数据。器件支持 SPI 的所有四种模式。通常使用以下 3 个引脚来实现通信：

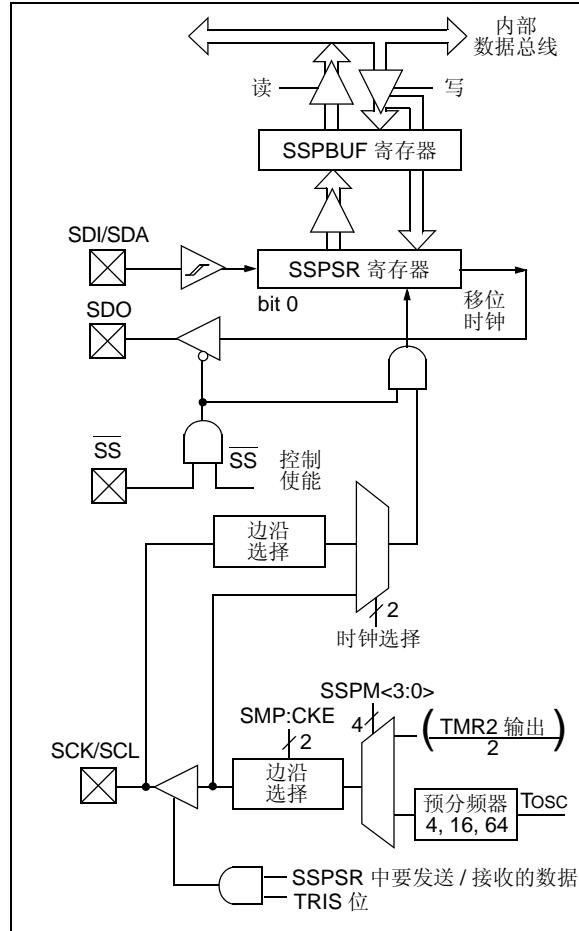
- 串行数据输出——SDO
- 串行数据输入——SDI/SDA
- 串行时钟——SCK/SCL

此外，当处于从工作模式时要使用第 4 个引脚：

- 从选择——SS

图 17-1 给出了 MSSP 模块在 SPI 模式下的工作原理框图。

图 17-1: MSSP 框图（SPI 模式）



17.3.1 寄存器

MSSP 模块有四个寄存器用于 SPI 工作模式。这些寄存器包括：

- SSPCON1——控制寄存器
- SSPSTAT——状态寄存器
- SSPBUF——串行接收 / 发送缓冲寄存器
- SSPSR——移位寄存器（不可直接访问）

SSPCON1 和 SSPSTAT 是 SPI 工作模式下的控制寄存器和状态寄存器。SSPCON1 寄存器是可读写的。SSPSTAT 的低 6 位是只读的，而高 2 位是可读写的。

SSPSR 是用来将数据移入和移出的移位寄存器。SSPBUF 提供对 SSPSR 寄存器的间接访问。SSPBUF 是缓冲寄存器，可用于数据字节的写入或读出。

接收数据时，SSPSR 和 SSPBUF 共同构成一个双重缓冲接收器。当 SSPSR 接收到一个完整的字节之后，该字节会被送入 SSPBUF，同时将中断标志位 SSPIF 置 1。

在数据发送过程中，SSPBUF 不是双重缓冲的，对 SSPBUF 的写操作将同时写入 SSPBUF 和 SSPSR。

寄存器 17-1： SSPSTAT：MSSP 状态寄存器（SPI 模式）

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/A	P	S	R/W	UA	BF
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7

SMP: 采样位

SPI 主模式:

1 = 在数据输出时间的末端采样输入数据

0 = 在数据输出时间的中间采样输入数据

SPI 从模式:

当 SPI 工作在从模式时，必须将 SMP 清零。

bit 6

CKE: SPI 时钟选择位 ⁽¹⁾

1 = 时钟状态从有效转换到空闲时输出数据

0 = 时钟状态从空闲转换到有效时输出数据

bit 5

D/A: 数据 / 地址位

仅在 I²C 模式下使用。

bit 4

P: 停止位

仅在 I²C 模式下使用。当禁止 MSSP 模块（SSPEN 清零）时，该位被清零。

bit 3

S: 启动位

仅在 I²C 模式下使用。

bit 2

R/W: 读 / 写信息位

仅在 I²C 模式下使用。

bit 1

UA: 更新地址位

仅在 I²C 模式下使用。

bit 0

BF: 缓冲区满状态位（仅用于接收模式）

1 = 接收完成，SSPBUF 已满

0 = 接收未完成，SSPBUF 为空

注 1：时钟状态的极性由 SSPCON1 寄存器的 CKP 位设置。

寄存器 17-2: SSPCON1: MSSP 控制寄存器 1 (SPI 模式)

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 |
| bit 7 | bit 0 | | | | | | |

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	WCOL: 写冲突检测位 (仅用于发送模式) 1 = 正在发送前一个字时, 又有数据写入 SSPBUF 寄存器 (必须用软件清零) 0 = 未发生冲突
bit 6	SSPOV: 接收溢出指示位 (1) <u>SPI 从模式:</u> 1 = SSPBUF 寄存器中仍保存前一数据时, 又接收到一个新的字节。如果发生溢出, SSPSR 中的数据会丢失。溢出只会在从模式下发生。即使只是发送数据, 用户也必须读 SSPBUF, 以避免将溢出标志位置 1 (该位必须用软件清零)。 0 = 无溢出
bit 5	SSPEN: 同步串口使能位 (2) 1 = 使能串口并将 SCK、SDO、SDI 和 SS 配置为串口引脚。当使能串口时, 必须将 SDA 和 SCL 引脚配置为输入引脚。 0 = 禁止串口并将上述引脚配置为 I/O 端口引脚
bit 4	CKP: 时钟极性选择位 1 = 空闲状态时, 时钟为高电平 0 = 空闲状态时, 时钟为低电平
bit 3-0	SSPM<3:0>: 同步串口模式选择位 (3) 0101 = SPI 从模式, 时钟 = SCK 引脚, 禁止 SS 引脚控制, 可将 SS 用作 I/O 引脚 0100 = SPI 从模式, 时钟 = SCK 引脚, 使能 SS 引脚控制 0011 = SPI 主模式, 时钟 = TMR2 输出 /2 0010 = SPI 主模式, 时钟 = Fosc/64 0001 = SPI 主模式, 时钟 = Fosc/16 0000 = SPI 主模式, 时钟 = Fosc/4

注 1: 在主模式下, 溢出位不会被置 1, 因为每次接收 (和发送) 新数据都是通过写入 SSPBUF 寄存器启动的。

2: 当使能时, 必须将这些引脚正确地配置为输入或输出。

3: 在此未列出的位组合被保留或仅在 I²C 模式下实现。

17.3.2 工作原理

初始化 SPI 时需要指定几个选项。可以通过编程相应的控制位（SSPCON1<5:0> 和 SSPSTAT<7:6>）来指定这些选项。这些控制位用于指定以下选项：

- 主模式（SCK 作为时钟输出）
- 从模式（SCK 作为时钟输入）
- 时钟极性（SCK 的空闲状态）
- 数据输入采样阶段（数据输出时间的中间或末尾）
- 时钟边沿（在 SCK 的上升沿 / 下降沿输出数据）
- 时钟速率（仅用于主模式）
- 从选择模式（仅用于从模式）

MSSP 模块由一个发送 / 接收移位寄存器（SSPSR）和一个缓冲寄存器（SSPBUF）组成。SSPSR 将数据移入 / 移出器件，先移位 MSb。在新数据接收完毕前，SSPBUF 保存上次写入 SSPSR 的数据。一旦 8 位数据接收完毕，该字节就被移入 SSPBUF 寄存器。然后，SSPSTAT 寄存器的缓冲区满检测位 BF 和中断标志位 SSPIF 被置 1。这种双重缓冲数据接收方式（SSPBUF），允许在 CPU 读取刚接收的数据之前，就开始接收下一个字节。当 SSPBUF 寄存器正在发送 / 接收数据时，对它写入的任何数据都将被忽略，同时 SSPCON1 寄存器的写冲突检测位 WCOL 被置 1。用户必须用软件将 WCOL 位清零才能判断以后对 SSPBUF 寄存器的写入是否成功。

例 17-1：装载 SSPBUF（SSPSR）寄存器

```
LOOP    BTFSS   SSPSTAT, BF      ;Has data been received (transmit complete)?
        BRA     LOOP             ;No
        MOVF    SSPBUF, W       ;WREG reg = contents of SSPBUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSPBUF          ;New data to xmit
```

为确保应用软件能接收有效数据，在下一个要发送的数据字节写入 SSPBUF 之前，读取 SSPBUF 中现有的数据。SSPSTAT 寄存器的缓冲区满位 BF 用于表示何时 SSPBUF 装入了接收到的数据（发送完成）。当 SSPBUF 中的数据被读取后，BF 位即被清零。如果 SPI 仅作为一个发送器，则不必理会该数据。通常，可用 MSSP 中断来判断发送 / 接收是否已完成。必须读取和 / 或写入 SSPBUF。如果不打算使用中断，用软件查询的方法同样可确保不会发生写冲突。例 17-1 举例说明了装载 SSPBUF（SSPSR）进行数据发送的过程。

不能直接读写 SSPSR 寄存器，只能通过寻址 SSPBUF 寄存器来访问。此外，MSSP 状态寄存器（SSPSTAT）用于指示各种状态条件。

17.3.3 使能 SPI I/O

要使能串口，SSPCON1 寄存器的 SSP 使能位 SSPEN 必须置 1。要复位或重新配置 SPI 模式，要先将 SSPEN 位清零，重新初始化 SSPCON 寄存器，然后将 SSPEN 位置 1。这将把 SDI、SDO、SCK 和 SS 引脚配置为串口引脚。要将引脚用作串口功能，必须正确设置引脚的数据方向位（在 TRIS 寄存器中）：

- SDI 由 SPI 模块自动控制
- SDO 必须将对应的 TRIS 位清零
- SCK（主模式）必须将对应的 TRIS 位清零
- SCK（从模式）必须将对应的 TRIS 位置 1
- SS 必须将对应的 TRIS 位置 1

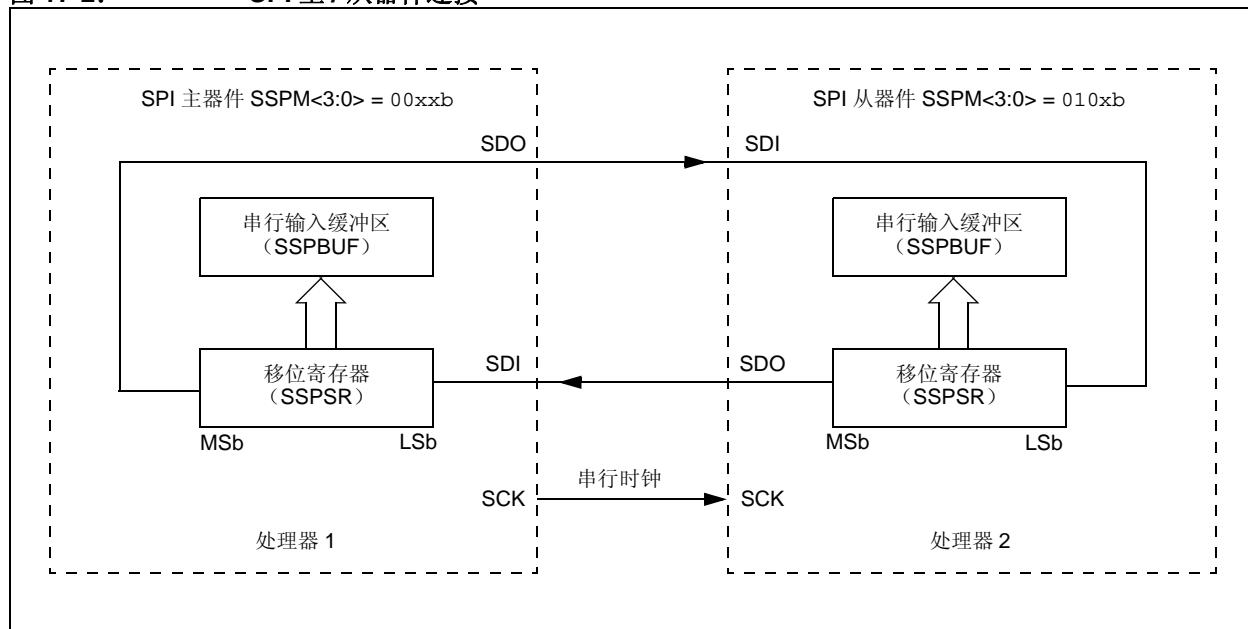
对于不需要的串口功能，可通过将对应的数据方向寄存器（TRIS）设置为相反值来改写。

17.3.4 典型连接

图 17-2 给出了两个单片机之间的典型连接。主控制器（处理器 1）通过发送 SCK 信号来启动数据传输。数据在编程设定的时钟边沿从两个移位寄存器移出，并在相反的时钟边沿锁存。必须将两个处理器的时钟极性（CKP）设置为相同，这样就可以同时收发数据。数据是否有意义（或无效数据），取决于应用软件。这就导致以下三种数据传输情形：

- 主器件发送数据——从器件发送无效（Dummy）数据
- 主器件发送数据——从器件发送数据
- 主器件发送无效数据——从器件发送数据

图 17-2: SPI 主 / 从器件连接



17.3.5 主模式

因为由主器件控制 SCK 信号，所以它可以在任意时刻启动数据传输。主器件根据软件协议确定从器件（图 17-2 中的处理器 2）在何时广播数据。

在主模式下，一写入 SSPBUF 寄存器就发送或接收数据。如果只打算将 SPI 作为接收器，则可以禁止 SDO 输出（将其编程设置为输入）。SSPSR 寄存器按设置的时钟速率，连续移入 SDI 引脚上的信号。每接收到一个字节，就将其装入 SSPBUF 寄存器，就像接收到普通字节一样（中断和状态位相应置 1）。这在以“线路活动监控”（Line Activity Monitor）方式工作的接收器应用中很有用。

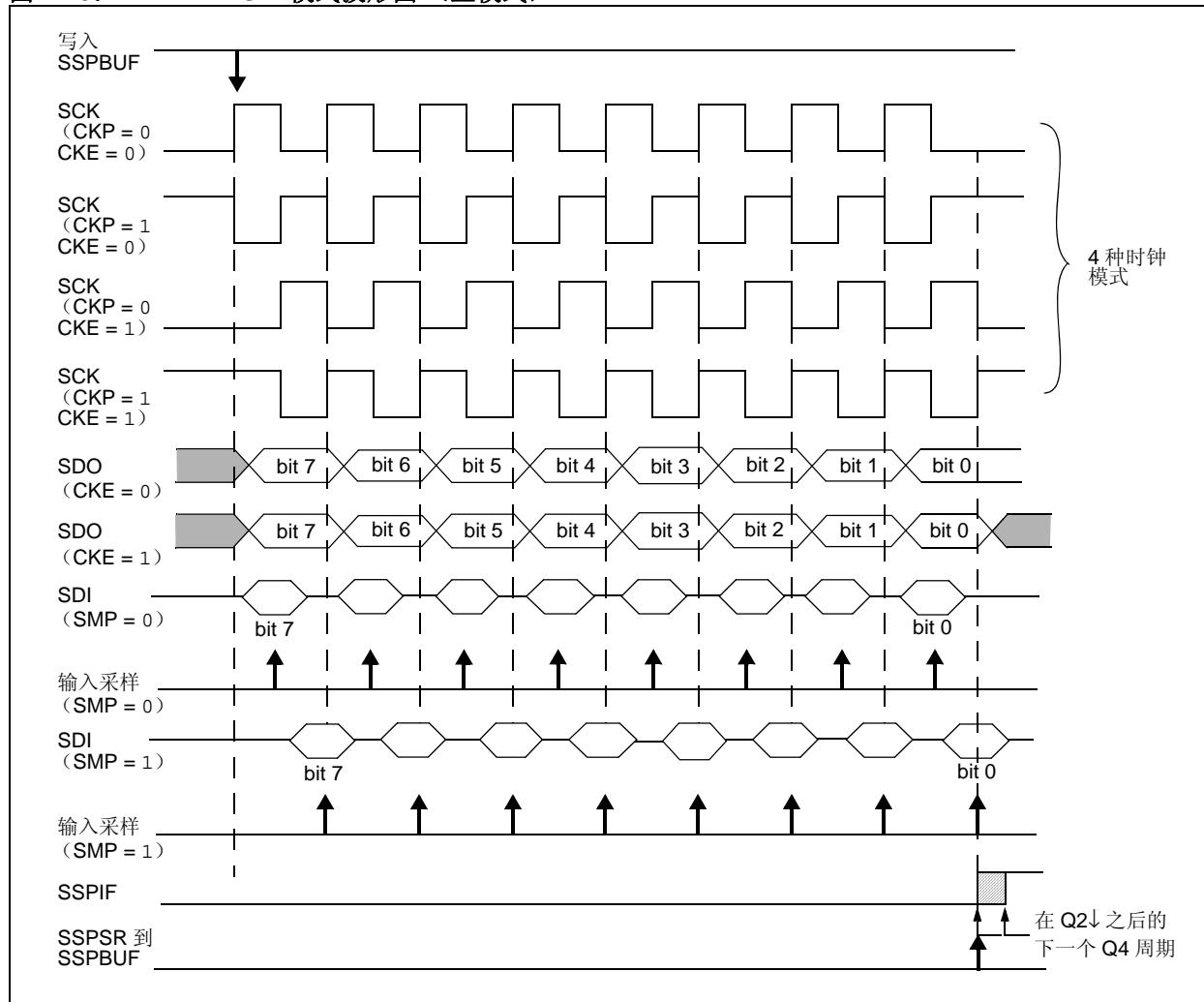
可通过对 SSPCON1 寄存器的 CKP 位进行适当的编程来选择时钟极性。图 17-3、图 17-5 和图 17-6 给出了 SPI 通信的波形图，其中 MSB 先发送。在主模式下，SPI 时钟速率（比特率）可由用户编程设定为以下几种之一：

- Fosc/4（或 Tcy）
- Fosc/16（或 4 • Tcy）
- Fosc/64（或 16 • Tcy）
- Timer2 输出 /2

这样可使数据速率最高达到 16.00 Mbps（时钟频率为 64 MHz）。

图 17-3 给出了主模式的波形图。当 CKE 位置 1 时，SDO 数据在 SCK 上出现时钟边沿前一直有效。图中所示的输入采样的变化由 SMP 位的状态反映。图中给出了将接收到的数据装入 SSPBUF 的时刻。

图 17-3： SPI 模式波形图（主模式）



17.3.6 从模式

在从模式下，当 SCK 引脚上出现外部时钟脉冲时发送和接收数据。当最后一位数据被锁存后，中断标志位 SSPIF 置 1。

在 SPI 从模式下使能该模块前，时钟线必须处于相应的空闲状态。时钟线可通过读 SCK 引脚来查看。空闲状态由 SSPCON1 寄存器的 CKP 位决定。

在从模式下，外部时钟由 SCK 引脚上的外部时钟源提供。外部时钟必须满足电气规范中规定的高电平和低电平的最短时间要求。

在休眠模式下，从器件仍可发送 / 接收数据。当接收到一个字节时，器件从休眠状态唤醒。

17.3.7 从选择同步

SS 引脚允许器件工作于同步从模式。SPI 必须处于从模式，并使能 SS 引脚控制（SSPCON1<3:0> = 04h）。要使 SS 引脚作为输入，不得将该引脚驱动为低电平。

数据锁存器必须为高电平。当 SS 引脚为低电平时，使能数据的发送和接收，同时 SDO 引脚被驱动。当 SS 引脚变为高电平时，即使是在字节的发送过程中，也不再驱动 SDO 引脚，而是变成悬空输出状态。可根据应用需要，在 SDO 引脚上外接上拉 / 下拉电阻。

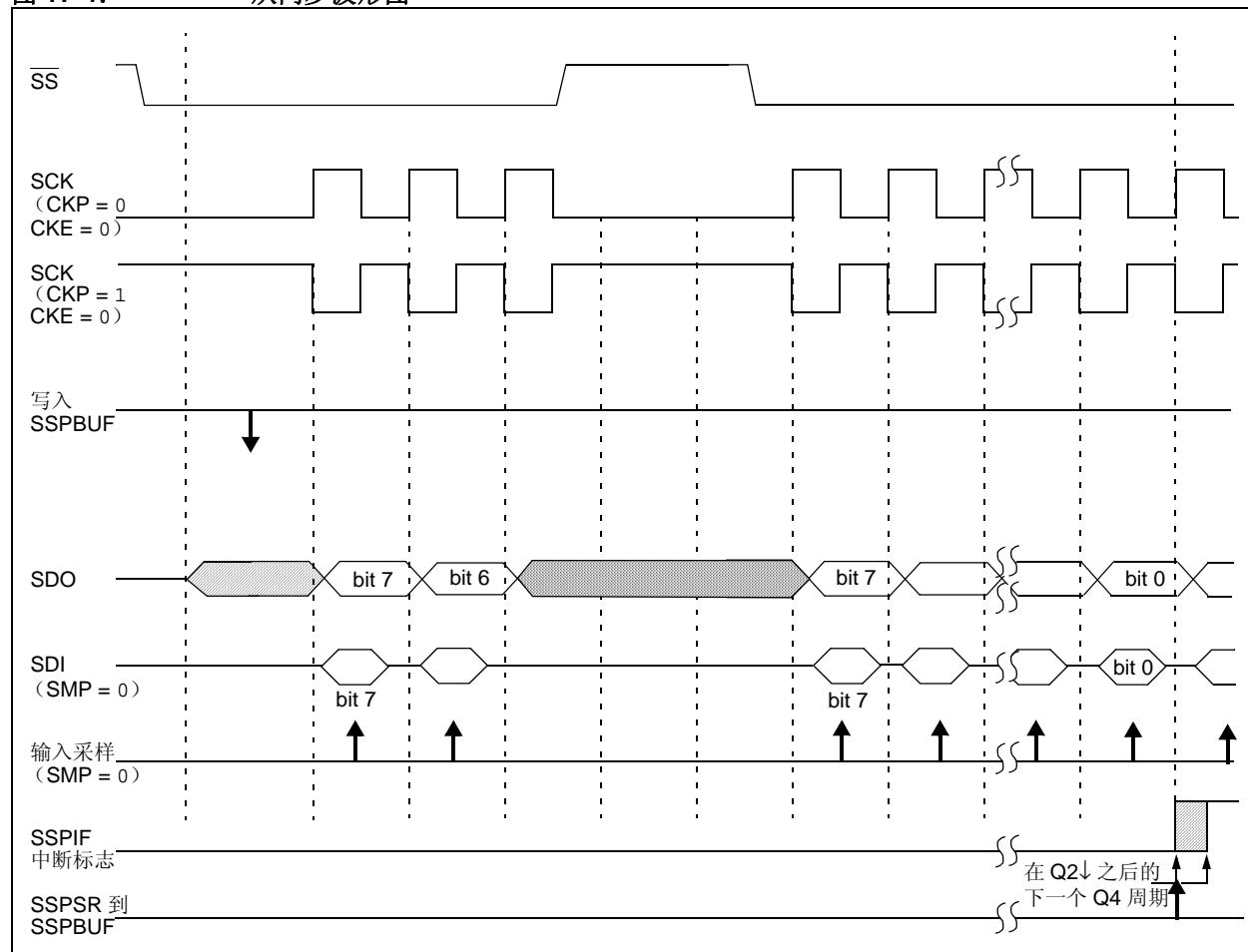
注 1: 当 SPI 处于从模式且 SS 引脚控制使能 (SSPCON<3:0> = 0100) 时，如果 SS 引脚设置为 VDD，SPI 模块将会复位。

2: 如果 SPI 工作在从模式下并且 CKE 置 1，则必须使能 SS 引脚控制。

当 SPI 模块复位后，位计数器被强制为 0。这是通过强制将 SS 引脚拉为高电平或将 SSPEN 位清零来实现的。

可将 SDO 引脚和 SDI 引脚相连，来仿真二线制通信。当 SPI 需要作为接收器工作时，SDO 引脚可被配置为输入端。这样就禁止了从 SDO 发送数据。因为 SDI 不会引起总线冲突，所以可以一直将其保留为输入（SDI 功能）。

图 17-4: 从同步波形图



PIC18F2XK20/4XK20

图 17-5: SPI 模式波形图 (从模式, CKE = 0)

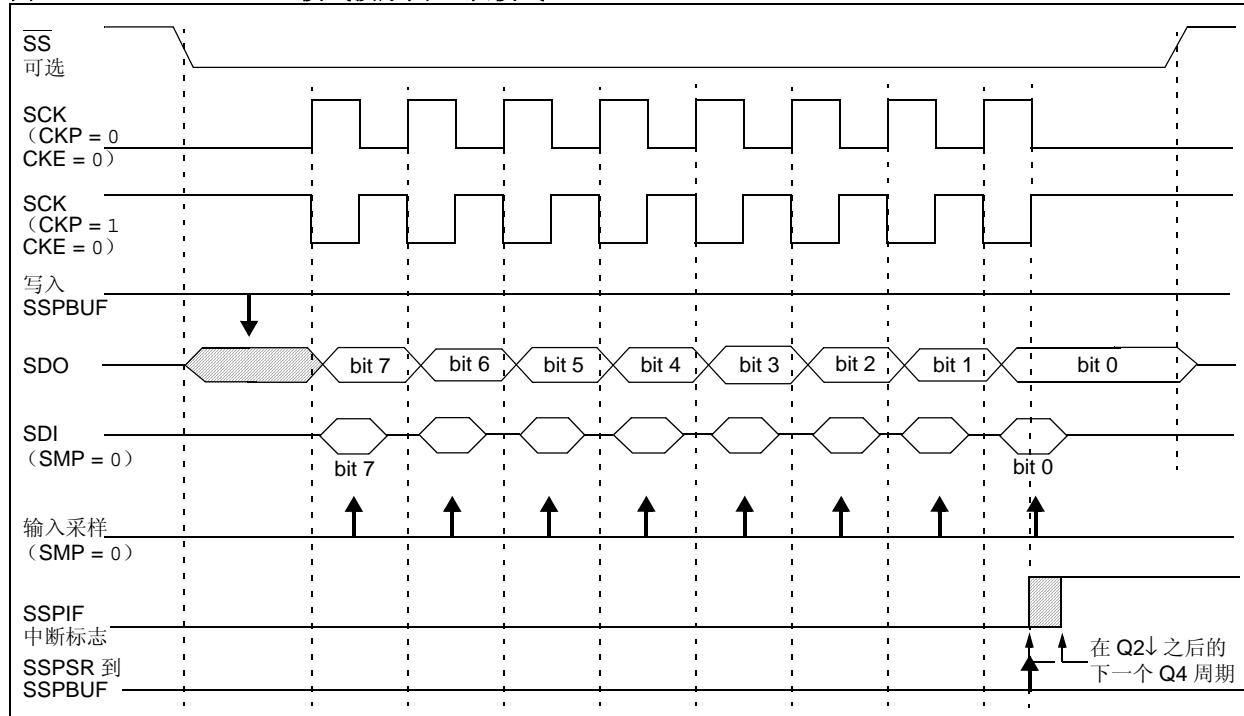
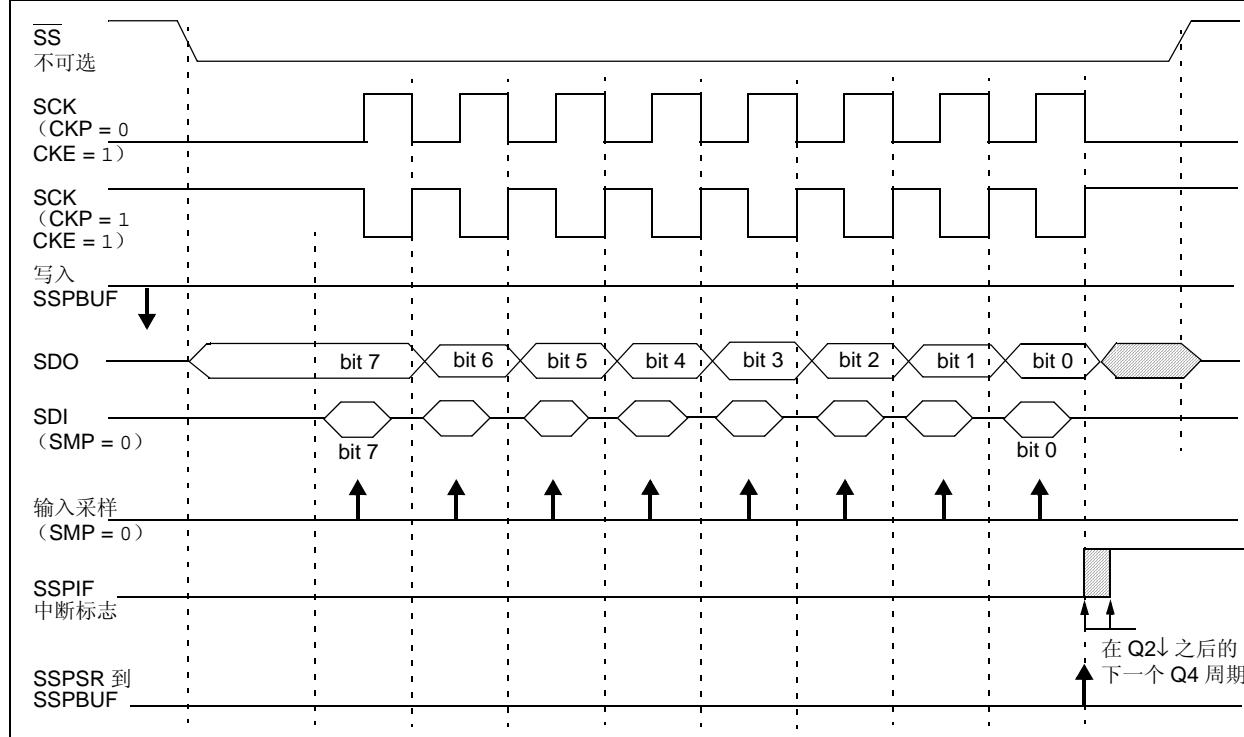


图 17-6: SPI 模式波形图 (从模式, CKE = 1)



17.3.8 在功耗管理模式下的操作

在 SPI 主模式下，模块时钟速度与全功耗模式下的不同；处于休眠模式时，所有时钟都停止。

在所有空闲模式下，需要为外设提供一个时钟。该时钟可能来自主时钟源、辅助时钟源（32.768 kHz 的 Timer1 振荡器）或 INTOSC 时钟源。更多信息，请参见第 3.0 节“**功耗管理模式**”。

在大多数情况下，主器件为 SPI 数据提供的时钟速度并不重要；但是，每个系统都应该评估此因素。

如果允许了 MSSP 中断，那么当主器件发送完数据时，MSSP 中断将唤醒控制器：

- 从休眠模式唤醒（在从模式下）
- 从空闲模式唤醒（在从模式或主器件模式下）

如果不从休眠或空闲模式退出，应该禁止 MSSP 中断。

在 SPI 主模式下，如果选择了休眠模式，所有模块的时钟都将停止，并且在器件被唤醒前，发送 / 接收将保持此停止状态。当器件返回到运行模式后，该模块将恢复发送和接收数据。

在 SPI 从模式下，SPI 发送 / 接收移位寄存器与器件异步工作。这可使器件处于任何功耗管理模式下，而且数据仍可被移入 SPI 发送 / 接收移位寄存器。当 8 位数据

全部接收到后，MSSP 中断标志位将置 1，并且如果允许中断的话，器件被唤醒。

17.3.9 复位的影响

复位会禁止 MSSP 模块并终止当前的数据传输。

17.3.10 总线模式兼容性

表 17-1 给出了标准 SPI 模式与 CKP 和 CKE 控制位状态之间的兼容性。

表 17-1： SPI 总线模式

标准 SPI 模式术语	控制位状态	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

还有一个 SMP 位用来控制数据何时被采样。

表 17-2： 与 SPI 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMROIE	INT0IE	RBIE	TMROIF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
TRISA	TRISA7 ⁽²⁾	TRISA6 ⁽²⁾	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	62
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISCO	62
SSPBUF	SSP 接收缓冲 / 发送寄存器								60
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
SSPSTAT	SMP	CKE	D/A	P	S	R/W	UA	BF	60

图注： SPI 模式下的 MSSP 模块不使用阴影单元。

注 1： 这些位在 28 引脚器件上未实现；始终保持这些位清零。

2： PORTA<7:6> 及其方向位根据不同的主振荡器模式被单独配置为端口引脚。当被禁止时，这些位读为 0。

17.4 I²C 模式

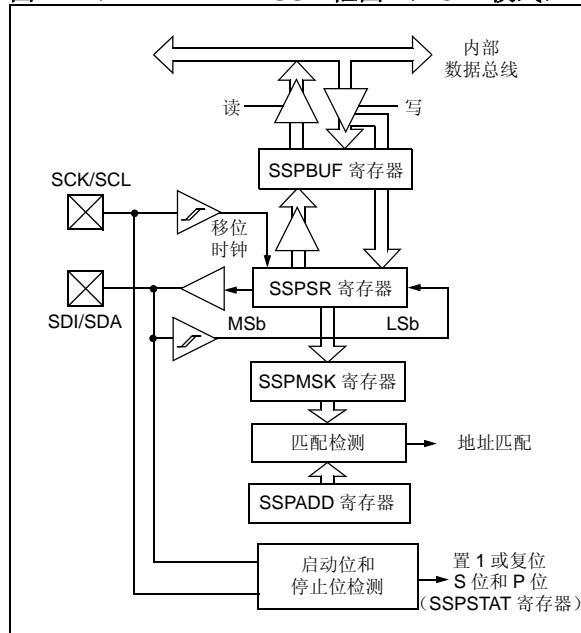
MSSP 模块工作在 I²C 模式时，可以实现所有的主和从功能（包括广播呼叫支持），并且由硬件提供遇到启动位和停止位时产生中断来判断总线何时空闲（多主器件功能）。MSSP 模块实现了标准模式规范以及 7 位和 10 位寻址。

有两个引脚用于数据传输：

- 串行时钟（SCL）——SCK/SCL
- 串行数据（SDA）——SDI/SDA

用户必须通过设置对应的 TRIS 位将这些引脚配置为输入引脚。

图 17-7：MSSP 框图（I²C™ 模式）



17.4.1 寄存器

MSSP 模块有 7 个寄存器用于 I²C 操作。这些寄存器包括：

- MSSP 控制寄存器 1（SSPCON1）
- MSSP 控制寄存器 2（SSPCON2）
- MSSP 状态寄存器（SSPSTAT）
- 串行接收 / 发送缓冲寄存器（SSPBUF）
- MSSP 移位寄存器（SSPSR）——不可直接访问
- MSSP 地址寄存器（SSPADD）
- MSSP 地址掩码寄存器（SSPMSK）

SSPCON1、SSPCON2 和 SSPSTAT 是在 I²C 工作模式下的控制寄存器和状态寄存器。SSPCON1 和 SSPCON2 寄存器是可读写的。SSPSTAT 的低 6 位是只读的，而高 2 位是可读写的。

SSPSR 是用来将数据移入或移出的移位寄存器。SSPBUF 是缓冲寄存器，可用于数据字节的写入或读出。

当 SSP 被配置为工作在主模式下时，SSPADD 的低 7 位用作波特率发生器的重载值。当 SSP 被配置为工作在 I²C 从模式下时，SSPADD 寄存器保存从器件的地址。通过使用 SSPMSK 寄存器检验地址寄存器的选定位，SSP 可被配置为响应一个地址范围。

接收数据时，SSPSR 和 SSPBUF 共同构成一个双重缓冲接收器。当 SSPSR 接收到一个完整的字节之后，该字节会被送入 SSPBUF，同时将中断标志位 SSPIF 置 1。

在数据发送过程中，SSPBUF 不是双重缓冲的，对 SSPBUF 的写操作将同时写入 SSPBUF 和 SSPSR。

寄存器 17-3: SSPADD: MSSP 地址和波特率寄存器 (I²C 模式)

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| ADD7 | ADD6 | ADD5 | ADD4 | ADD3 | ADD2 | ADD1 | ADD0 |
| bit 7 | | | | | | | bit 0 |

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

主模式

bit 7-0 **ADD<7:0>**: 波特率时钟分频比位

SCL 引脚时钟周期 = ((ADD<7:0> + 1) *4)/FOSC

10 位从模式: 高位地址字节

bit 7-3 未使用: 对高位地址字节不使用。该寄存器的位状态为无关。主器件发送的位格式由 I²C 规范确定, 必须等于 11110。但是, 那些位通过硬件进行比较, 并且不受该寄存器中的值影响。

bit 2-1 **ADD<9:8>**: 10 位地址的高 2 位

bit 0 未使用: 在此模式下未使用。位状态为无关。

10 位从模式: 低位地址字节

bit 7-0 **ADD<7:0>**: 10 位地址的低 8 位

7 位从模式

bit 7-1 **ADD<7:1>**: 7 位地址

bit 0 未使用: 在此模式下未使用。位状态为无关。

PIC18F2XK20/4XK20

寄存器 17-4: SSPSTAT: MSSP 状态寄存器 (I²C 模式)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/A	P ⁽¹⁾	S ⁽¹⁾	R/W ^(2, 3)	UA	BF
bit 7	bit 0						

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7

SMP: 边沿斜率控制位

在主模式或从模式下:

1 = 标准速度模式下禁止边沿斜率控制 (100 kHz 和 1 MHz)

0 = 高速模式下使能边沿斜率控制 (400 kHz)

bit 6

CKE: SMBus 选择位

在主模式或从模式下:

1 = 使能 SMBus 特定输入

0 = 禁止 SMBus 特定输入

bit 5

D/A: 数据 / 地址位

在主模式下:

保留。

在从模式下:

1 = 表示上一个接收或发送的字节是数据

0 = 表示上一个接收或发送的字节是地址

bit 4

P: 停止位 ⁽¹⁾

1 = 表示上次检测到停止位

0 = 上次未检测到停止位

bit 3

S: 启动位 ⁽¹⁾

1 = 表示上次检测到启动位

0 = 上次未检测到启动位

bit 2

R/W: 读 / 写信息位 (仅用于 I²C 模式) ^(2, 3)

在从模式下:

1 = 读

0 = 写

在主模式下:

1 = 正在进行发送

0 = 不在进行发送

bit 1

UA: 更新地址位 (仅用于 10 位从模式)

1 = 表示用户需要更新 SSPADD 寄存器中的地址

0 = 不需要更新地址

bit 0

BF: 缓冲区满状态位

在发送模式下:

1 = SSPBUF 已满

0 = SSPBUF 为空

在接收模式下:

1 = SSPBUF 已满 (不包括 ACK 位和停止位)

0 = SSPBUF 为空 (不包括 ACK 位和停止位)

注 1: 该位在复位及 SSPEN 清零时被清零。

2: 该位保存上一次地址匹配后的 R/W 位信息。该位仅在从地址匹配到出现下一个启动位、停止位或非 ACK 位之间有效。

3: 将该位与 SEN、RSEN、PEN、RCEN 或 ACKEN 进行逻辑或运算将指示 MSSP 是否处于有效模式。

寄存器 17-5: SSPCON1: MSSP 控制寄存器 1 (I²C 模式)

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| WCOL | SSPOV | SSPEN | CKP | SSPM3 | SSPM2 | SSPM1 | SSPM0 |
| bit 7 | bit 0 | | | | | | |

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

- bit 7 **WCOL:** 写冲突检测位
在主发送模式下:
 1 = 当 I²C 不满足启动发送数据的条件时, 试图向 SSPBUF 寄存器写入数据 (必须用软件清零)
 0 = 未发生冲突
在从发送模式下:
 1 = 正在发送前一个字时, 又有数据写入 SSPBUF 寄存器 (必须用软件清零)
 0 = 未发生冲突
在接收模式 (主或从模式) 下:
 该位是无关位。
- bit 6 **SSPOV:** 接收溢出指示位
在接收模式下:
 1 = SSPBUF 寄存器仍保存前一字节时, 接收到一个新的字节 (必须用软件清零)
 0 = 无溢出
在发送模式下:
 在发送模式下, 该位是无关位。
- bit 5 **SSPEN:** 同步串口使能位
 1 = 使能串口并将 SDA 和 SCL 引脚配置为串口引脚。当使能串口时, 必须将 SDA 和 SCL 引脚配置为输入引脚。
 0 = 禁止串口并将上述引脚配置为 I/O 端口引脚
- bit 4 **CKP:** SCK 释放控制位
在从模式下:
 1 = 释放时钟
 0 = 保持时钟低电平 (时钟延长), 用来确保数据建立时间
在主模式下:
 在此模式下未使用。
- bit 3-0 **SSPM<3:0>:** 同步串口模式选择位
 1111 = I²C 从模式, 10 位地址, 并允许启动位和停止位中断
 1110 = I²C 从模式, 7 位地址, 并允许启动位和停止位中断
 1011 = I²C 固件控制的主模式 (从器件空闲)
 1000 = I²C 主模式, 时钟 = FOSC/(4 * (SSPADD + 1))
 0111 = I²C 从模式, 10 位地址
 0110 = I²C 从模式, 7 位地址
 此处未列出的位组合被保留或仅在 SPI 模式下实现。

寄存器 17-6: SSPCON2: MSSP 控制寄存器 2 (I²C 模式)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT ⁽²⁾	ACKEN ⁽¹⁾	RCEN ⁽¹⁾	PEN ⁽¹⁾	RSEN ⁽¹⁾	SEN ⁽¹⁾
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **GCEN:** 广播呼叫使能位 (仅用于从模式)
 1 = 当 SSPSR 接收到广播呼叫地址 (0000h) 时产生中断
 0 = 禁止广播呼叫地址

bit 6 **ACKSTAT:** 应答状态位 (仅用于主发送模式)
 1 = 未收到来自从器件的应答
 0 = 收到来自从器件的应答

bit 5 **ACKDT:** 应答数据位 (仅用于主接收模式) ⁽²⁾
 1 = 无应答
 0 = 应答

bit 4 **ACKEN:** 应答序列使能位 (仅用于主接收模式) ⁽¹⁾
 1 = 在 SDA 和 SCL 引脚上发出应答序列, 并发送 ACKDT 数据位。由硬件自动清零。
 0 = 应答序列空闲

bit 3 **RCEN:** 接收使能位 (仅用于主模式) ⁽¹⁾
 1 = 使能 I²C 接收模式
 0 = 接收空闲

bit 2 **PEN:** 停止条件使能位 (仅用于主模式) ⁽¹⁾
 1 = 在 SDA 和 SCL 引脚上发出停止条件。由硬件自动清零。
 0 = 停止条件空闲

bit 1 **RSEN:** 重复启动条件使能位 (仅用于主模式) ⁽¹⁾
 1 = 在 SDA 和 SCL 引脚上发出重复启动条件。由硬件自动清零。
 0 = 重复启动条件空闲

bit 0 **SEN:** 启动条件使能 / 延长使能位 ⁽¹⁾
在主模式下:
 1 = 在 SDA 和 SCL 引脚上发出启动条件。由硬件自动清零。
 0 = 启动条件空闲
在从模式下:
 1 = 为从发送和从接收 (已使能时钟延长) 使能时钟延长
 0 = 禁止从接收的时钟延长。仍然使能从发送的时钟延长。

- 注 1: 对于 ACKEN、RCEN、PEN、RSEN 和 SEN 位: 如果 I²C 模块不处于空闲模式, 可能这些位不会被置 1 (无并行工作), 并且可能不会写入 SSPBUF (或禁止写 SSPBUF)。
 2: 当用户在接收结束时发出一个应答序列时, 发送该值。

17.4.2 工作原理

MSSP 模块功能通过将 SSPCON1 寄存器的 SSPEN 位置 1 使能。

SSPCON1 寄存器用于控制 I²C 工作模式。可通过设置 SSPCON1 寄存器的模式选择位选择以下 I²C 模式之一：

- I²C 主模式，时钟 = (Fosc/(4 x (SSPADD + 1)))
- I²C 从模式（7 位地址）
- I²C 从模式（10 位地址）
- I²C 从模式（7 位地址），允许启动位和停止位中断
- I²C 从模式（10 位地址），允许启动位和停止位中断
- I²C 固件控制的主模式，从器件空闲

如果通过将相应的 TRIS 位置 1，将 SCL 和 SDA 引脚编程为输入引脚，则在 SSPEN 位置 1 时选择任何 I²C 模式，将强制上述引脚漏极开路。要确保此模块正常工作，必须为 SCL 和 SDA 引脚外接上拉电阻。

17.4.3 从模式

在从模式下，SCL 和 SDA 引脚必须被配置为输入。必要时 MSSP 模块将用输出数据改写输入状态（从发送器）。

I²C 从模式硬件总是在地址匹配时产生中断。用户也可以通过模式选择位，选择在遇到启动位或停止位时产生中断。

当地址匹配或在地址匹配后发送的数据被接收时，硬件会自动产生一个应答（ACK）脉冲，并把当前 SSPSR 寄存器中接收到的值装入 SSPBUF 寄存器。

只要满足以下条件之一，MSSP 模块就不会产生此 ACK 脉冲：

- 在接收到数据前，SSPSTAT 寄存器的缓冲区满位 BF 被置 1。
- 在接收到数据前，SSPCON1 寄存器的溢出位 SSPOV 被置 1。

在上述情况下，SSPSR 寄存器的值不会装入 SSPBUF，但 PIR1 寄存器的 SSPIF 位会置 1。BF 位是通过读取 SSPBUF 寄存器清零的，而 SSPOV 位是通过软件清零的。

为确保正常工作，SCL 时钟输入必须满足最小高电平和最小低电平时间要求。在时序参数 100 和参数 101（见表 26-19）中给出了 I²C 规范的高低电平时间和对 MSSP 模块的具体要求。

17.4.3.1 寻址

一旦 MSSP 模块被使能，它就会等待启动条件出现。启动条件出现后，8 位数据被移入 SSPSR 寄存器。在时钟（SCL）线的上升沿采样所有的输入位。寄存器 SSPSR<7:1> 的值会和 SSPADD 寄存器的值比较，该比较是在第 8 个时钟（SCL）脉冲的下降沿进行的。如果地址匹配，并且 BF 位和 SSPOV 位都被清零，会发生以下事件：

1. SSPSR 寄存器的值被装入 SSPBUF 寄存器。
2. 缓冲区满标志位 BF 被置 1。
3. 产生 ACK 脉冲。
4. 在第 9 个 SCL 脉冲的下降沿，PIR1 寄存器的 MSSP 中断标志位 SSPIF 被置 1（如果允许中断，则产生中断）。

在 10 位地址模式下，从器件需要接收两个地址字节。第一个地址字节的高 5 位（MSb）将指定这是否是一个 10 位地址。SSPSTAT 寄存器的 R/W 位必须指定写操作，这样从器件才能接收到第二个地址字节。对于 10 位地址，第一个字节应该是“11110 A9 A8 0”，其中“A9”和“A8”是该地址的高 2 位。10 位地址模式的操作步骤如下，其中 7-9 步是针对从发送器而言的。

1. 接收地址的第一个（高）字节（SSPIF 位、SSPSTAT 寄存器的 BF 和 UA 位置 1）。
2. 用地址的第二个（低）字节更新 SSPADD 寄存器（UA 位清零并释放 SCL 线）。
3. 读 SSPBUF 寄存器（BF 位清零）并将标志位 SSPIF 清零。
4. 接收地址的第二个（低）字节（SSPIF、BF 和 UA 位置 1）。如果地址匹配，则 SCL 将保留直到下一步。否则 SCL 线不会保留。
5. 使用地址的第一个（高）字节更新 SSPADD 寄存器。（这将清零 UA 位并释放保留的 SCL 线。）
6. 读 SSPBUF 寄存器（BF 位清零）并将标志位 SSPIF 清零。
7. 接收重复启动条件。
8. 接收地址的第一个（高）字节（SSPIF 位和 BF 位置 1）。
9. 读 SSPBUF 寄存器（BF 位清零）并将标志位 SSPIF 清零。

17.4.3.2 接收

当地址字节的 R/W 位清零并发生地址匹配时，**SSPSTAT** 寄存器的 R/W 位清零。接收的地址被装入 **SSPBUF** 寄存器，且 **SDA** 线保持低电平（ACK）。

当发生地址字节溢出时，则不会产生应答（ACK）脉冲。溢出条件定义为 **SSPSTAT** 寄存器的 **BF** 位被置 1，或 **SSPCON1** 寄存器的 **SSPOV** 位被置 1。

每传输一个数据字节都会产生一个 **MSSP** 中断。**PIR1** 寄存器的标志位 **SSPIF** 必须用软件清零。通过 **SSPSTAT** 寄存器可以确定该字节的状态。

当 **SSPCON2** 寄存器的 **SEN** 位被置 1 时，**SCK/SCL** 将在每次数据传输后保持低电平（时钟延长）。必须通过将 **SSPCON1** 寄存器的 **CKP** 位置 1 来释放时钟。更多详细信息，请参见第 17.4.4 节“时钟延长”。

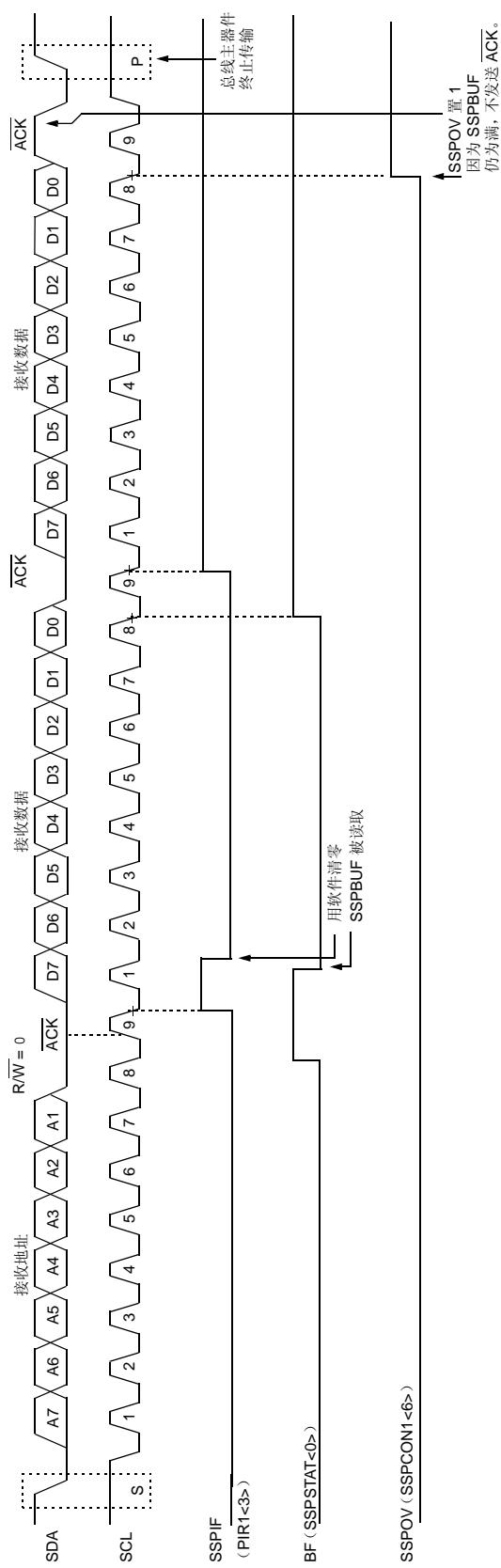
17.4.3.3 发送

当输入的地址字节的 R/W 位置 1 并发生地址匹配时，**SSPSTAT** 寄存器的 R/W 位置 1。接收到的地址被装入 **SSPBUF** 寄存器。ACK 脉冲在第 9 位发送，同时不管 **SEN** 的值如何，**SCK/SCL** 引脚保持低电平（更多详细信息，请参见第 17.4.4 节“时钟延长”）。通过延长时钟，主器件只有在从器件准备好发送数据时，才发出另一个时钟脉冲。发送的数据必须被装入 **SSPBUF** 寄存器，同时也被装入 **SSPSR** 寄存器。然后，应通过将 **SSPCON1** 寄存器的 **CKP** 位置 1 来使能 **SCK/SCL** 引脚。8 个数据位在 **SCL** 输入的下降沿被移出。这可确保在 **SCL** 为高电平期间 **SDA** 信号是有效的（图 17-9）。

来自主接收器的 ACK 脉冲将在第 9 个 **SCL** 输入脉冲的上升沿锁存。如果 **SDA** 线为高电平（无 ACK 应答），那么表示数据传输已完成。在这种情况下，如果从器件锁存了 ACK，将复位从逻辑（复位 **SSPSTAT** 寄存器），同时从器件监视下一个启动位的出现。如果 **SDA** 线为低电平（ACK），则必须将下一个要发送的数据装入 **SSPBUF** 寄存器。同样，必须通过将 **CKP** 位置 1 来使能 **SCK/SCL** 引脚。

每传输一个数据字节都会产生一个 **MSSP** 中断。**SSPIF** 位必须用软件清零，**SSPSTAT** 寄存器用于确定字节的状态。**SSPIF** 位在第 9 个时钟脉冲的下降沿被置 1。

图 17-8: I²CTM 从模式接收时序 (SEN = 0, 7 位地址)



PIC18F2XK20/4XK20

图 17-9: I²C™ 从模式发送时序 (7 位地址)

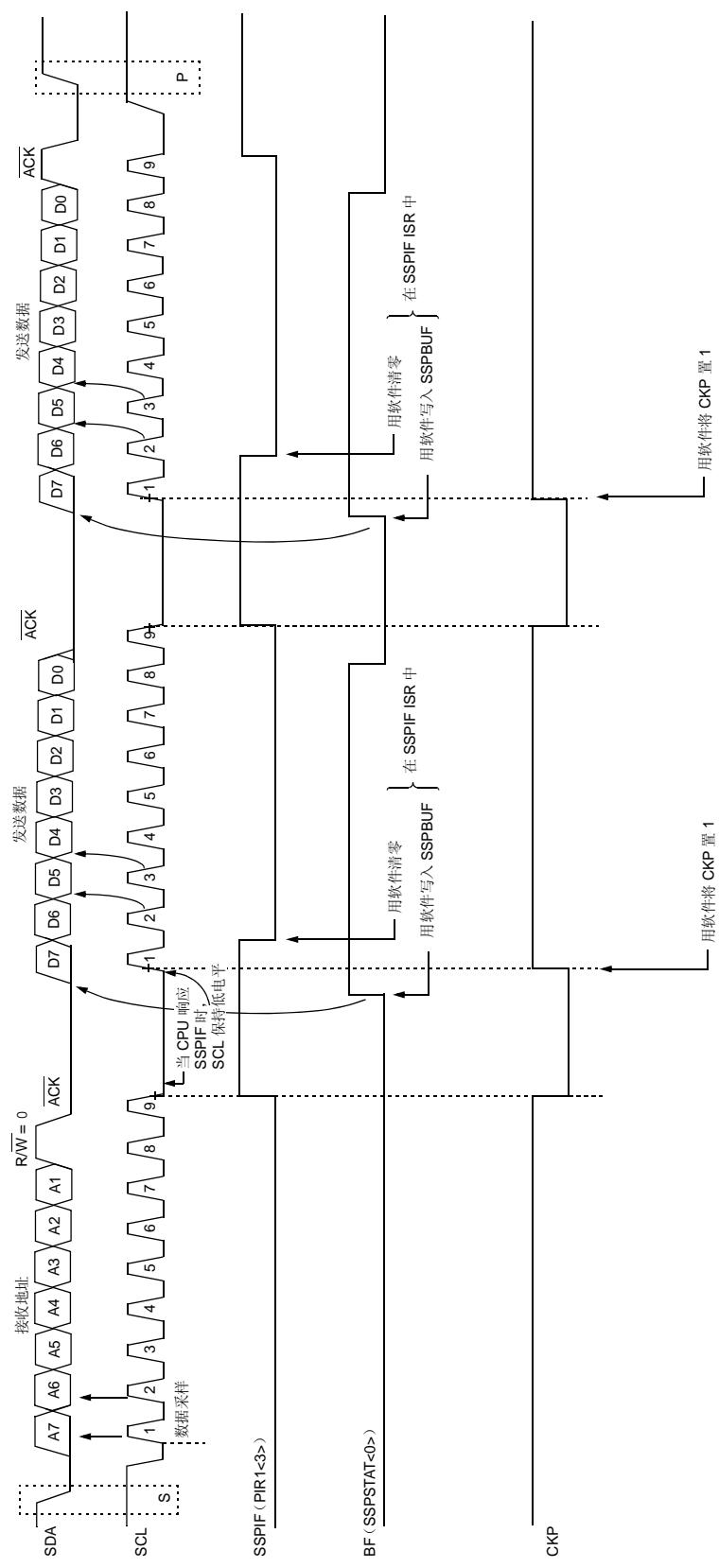
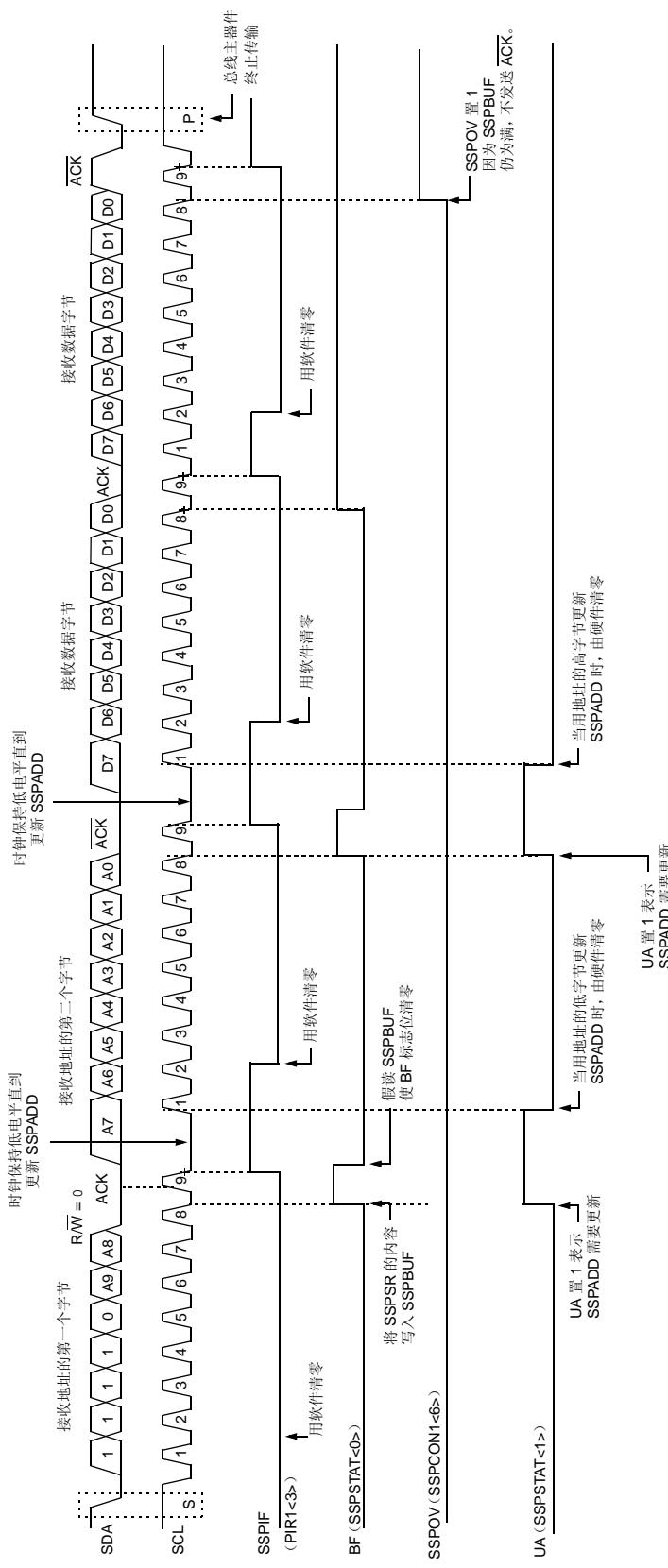
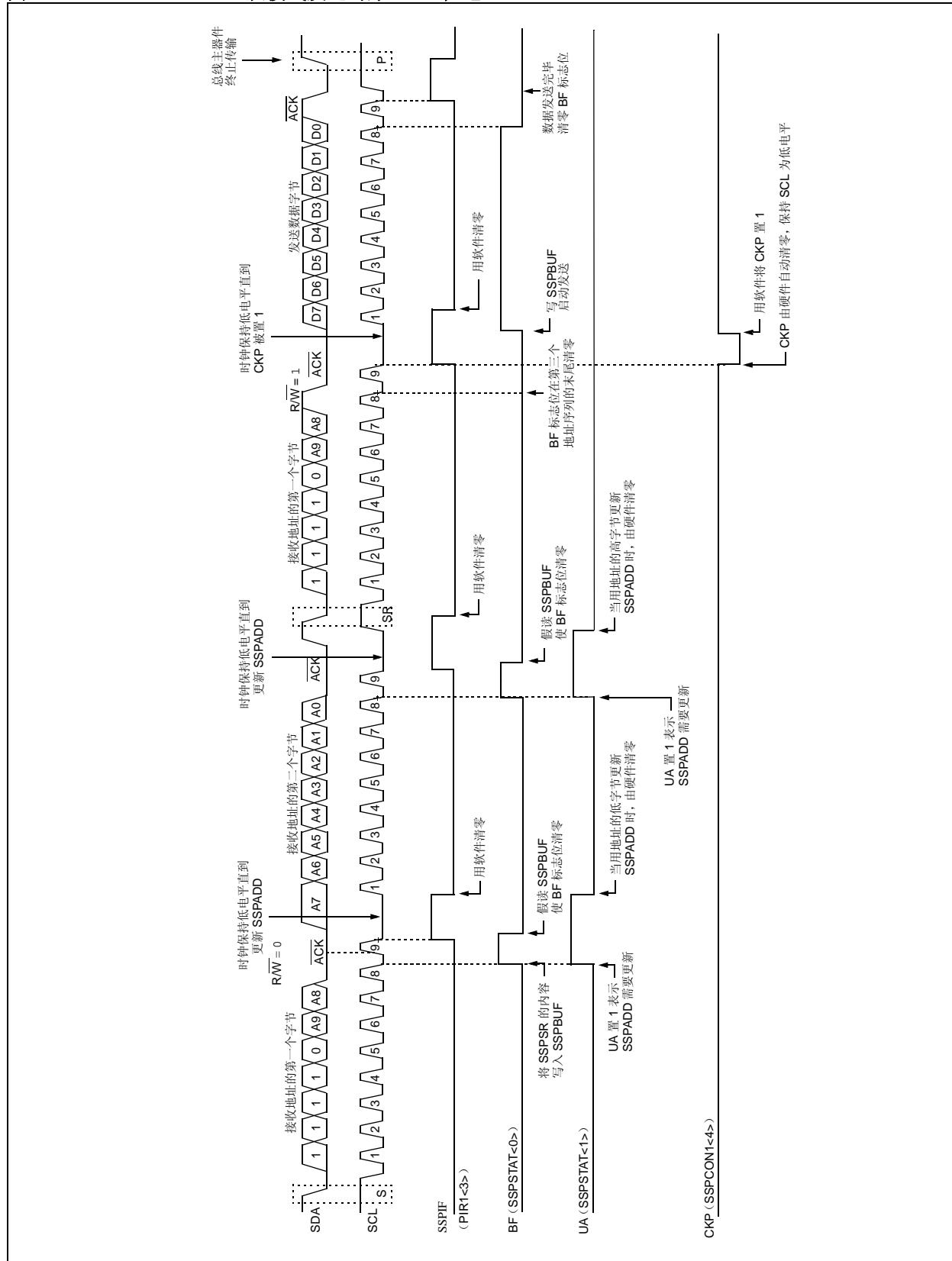


图 17-10: I²CTM 从模式接收时序 (SEN = 0, 10 位地址)



PIC18F2XK20/4XK20

图 17-11: I²C™ 从模式发送时序 (10 位地址)



17.4.3.4 SSP 掩码寄存器

SSP 掩码 (SSPMASK) 寄存器在 I²C 从模式下可用，用作地址比较操作期间 SSPSR 寄存器中保存的值的掩码。SSPMASK 寄存器中的零 (0) 位会影响对 SSPSR 寄存器中相应位的忽略。

任何复位后，该寄存器都会复位到全 1 状态，因此，在写入掩码值之前对标准 SSP 操作没有影响。

在将 SSPM<3:0> 位置 1 前必须初始化该寄存器，以便选择 I²C 从模式（7 位或 10 位地址）。

SSP 掩码寄存器在以下期间保持活动状态：

- 7 位地址模式：A<7:1> 的地址比较。
- 10 位地址模式：仅针对 A<7:0> 的地址比较。在接收地址的第一个（高）字节期间，SSP 掩码没有影响。

寄存器 17-7：SSPMASK：SSP 掩码寄存器

| R/W-1 |
|-------|-------|-------|-------|-------|-------|-------|---------------------|
| MSK7 | MSK6 | MSK5 | MSK4 | MSK3 | MSK2 | MSK1 | MSK0 ⁽¹⁾ |
| bit 7 | | | | | | | bit 0 |

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-1

MSK<7:1>：掩码位

1 = 接收到的地址 bit n 与 SSPADD<n> 相比较来检测 I²C 模式下地址是否匹配
0 = 接收到的地址 bit n 不用于检测 I²C 模式下地址是否匹配

bit 0

MSK<0>：掩码位用于 I²C 从模式，10 位地址⁽¹⁾

I²C 从模式，10 位地址 (SSPM<3:0> = 0111)：
1 = 接收到的地址 bit 0 与 SSPADD<0> 相比较来检测 I²C 模式下地址是否匹配
0 = 接收到的地址 bit 0 不用于检测 I²C 模式下地址是否匹配

注 1：MSK0 位仅在 10 位从模式下使用。在所有其他模式下，该位不起作用。

17.4.4 时钟延长

7位和10位从模式均能在发送序列期间实现自动时钟延长。

SSPCON2寄存器的SEN位允许在接收期间使能时钟延长。将SEN置1将使SCL引脚在每个数据接收序列的末尾保持低电平。

17.4.4.1 7位从接收模式(SEN = 1)的时钟延长

在7位从接收模式下，如果在ACK序列末尾的第9个时钟的下降沿将BF位置1，则SSPCON1寄存器的CKP位就会自动清零，强制SCL输出保持在低电平。CKP被清零会将SCL线拉为低电平。在允许继续接收之前，必须在用户的ISR中将CKP位置1。保持SCL线为低电平，用户可以在主器件发起另一个数据传输序列之前，有时间执行ISR并读取SSPBUF的内容。这将防止发生缓冲区溢出（见图17-13）。

- 注 1:** 如果用户在第9个时钟的下降沿到来之前读取了SSPBUF的内容，使得BF位被清零，那么CKP位就不会被清零，也不会发生时钟延长。
- 2:** 不管BF位的状态如何，CKP位都可以用软件置1。为避免溢出，在下一个接收序列开始之前，用户要注意在ISR中清零BF位。

17.4.4.2 10位从接收模式(SEN = 1)的时钟延长

在10位从接收模式下，在地址序列中会自动发生时钟延长，但是CKP位不会被清零。在这期间，如果UA位在第9个时钟之后置1，将启动时钟延长。UA位在接收到10位地址的高字节后被置1，然后接收10位地址的第二个字节并清零R/W位。在更新SSPADD时释放时钟线。如同7位模式一样，在每个数据接收序列中均会发生时钟延长。

- 注:** 如果用户在第9个时钟的下降沿出现之前查询UA位，并通过更新SSPADD寄存器清零UA位，而且在此之前用户没有读取SSPBUF寄存器使BF位清零，则CKP位的电平仍然不会被拉低。基于BF位状态的时钟延长仅在数据序列中出现，不会出现在地址序列中。

17.4.4.3 7位从发送模式的时钟延长

如果BF位被清零，7位从发送模式将通过在第9个时钟的下降沿出现后清零CKP位，以实现时钟延长。上述情形与SEN位的状态无关。

用户的ISR必须先将CKP位置1才可以继续发送。在保持SCL线为低电平期间，用户在主器件发起另一个数据传输序列之前，将有时间执行ISR并装入SSPBUF的内容（见图17-9）。

- 注 1:** 如果用户在第9个时钟的下降沿之前就装入SSPBUF的内容，使得BF位被置1，那么CKP位就不会被清零，也不会发生时钟延长。
- 2:** 不管BF位的状态如何，CKP位都可以用软件置1。

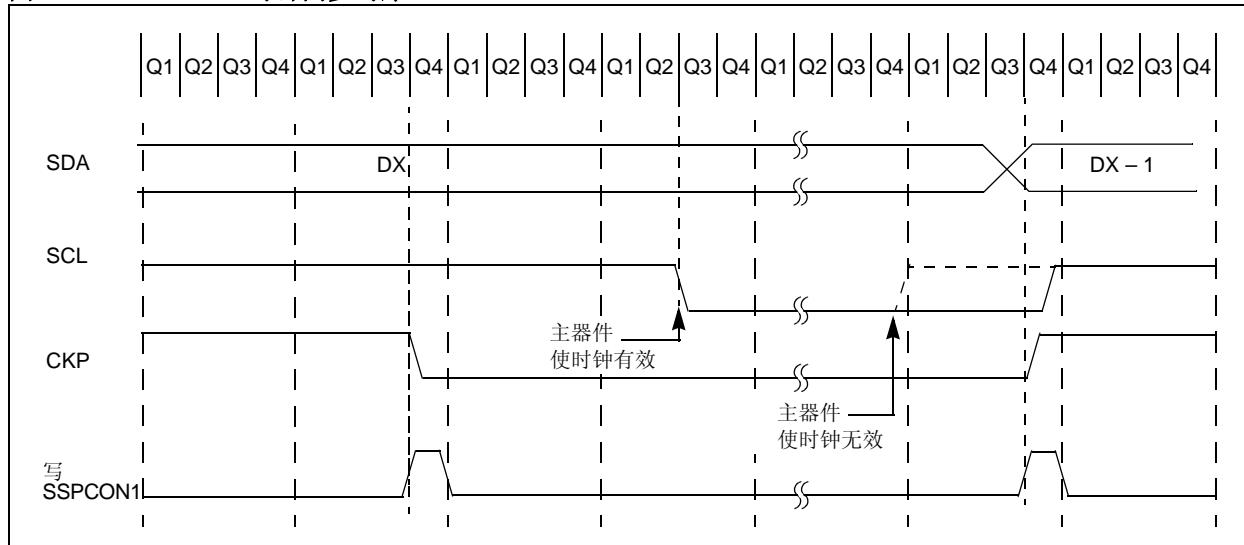
17.4.4.4 10位从发送模式的时钟延长

在10位从发送模式下，在前两个地址序列中由UA位的状态来控制时钟延长，正如同10位从接收模式一样。头两个地址后跟着第三个地址序列，该地址序列包含10位地址的高位和被置为1的R/W位。在执行完第三个地址序列后，UA位不置1，此时模块配置为发送模式，BF标志位控制时钟延长，正如7位从发送模式一样（见图17-11）。

17.4.4.5 时钟同步和 CKP 位

当 CKP 位被清零时，SCL 输出被强制为 0。然而，将 CKP 位清零不会将 SCL 输出拉为低电平，除非已经采样到 SCL 输出为低电平。因此，CKP 位不会将 SCL 线拉为低电平，除非外部 I²C 主器件将 SCL 线拉低。SCL 输出将保持低电平，直到 CKP 位置 1 且 I²C 总线上的所有其他器件将 SCL 电平拉高为止。这可以确保对 CKP 位的写操作不会违反 SCL 的最小高电平时间要求（见图 17-12）。

图 17-12：时钟同步时序



PIC18F2XK20/4XK20

图 17-13: I²C™ 从模式接收时序 (SEN = 1, 7 位地址)

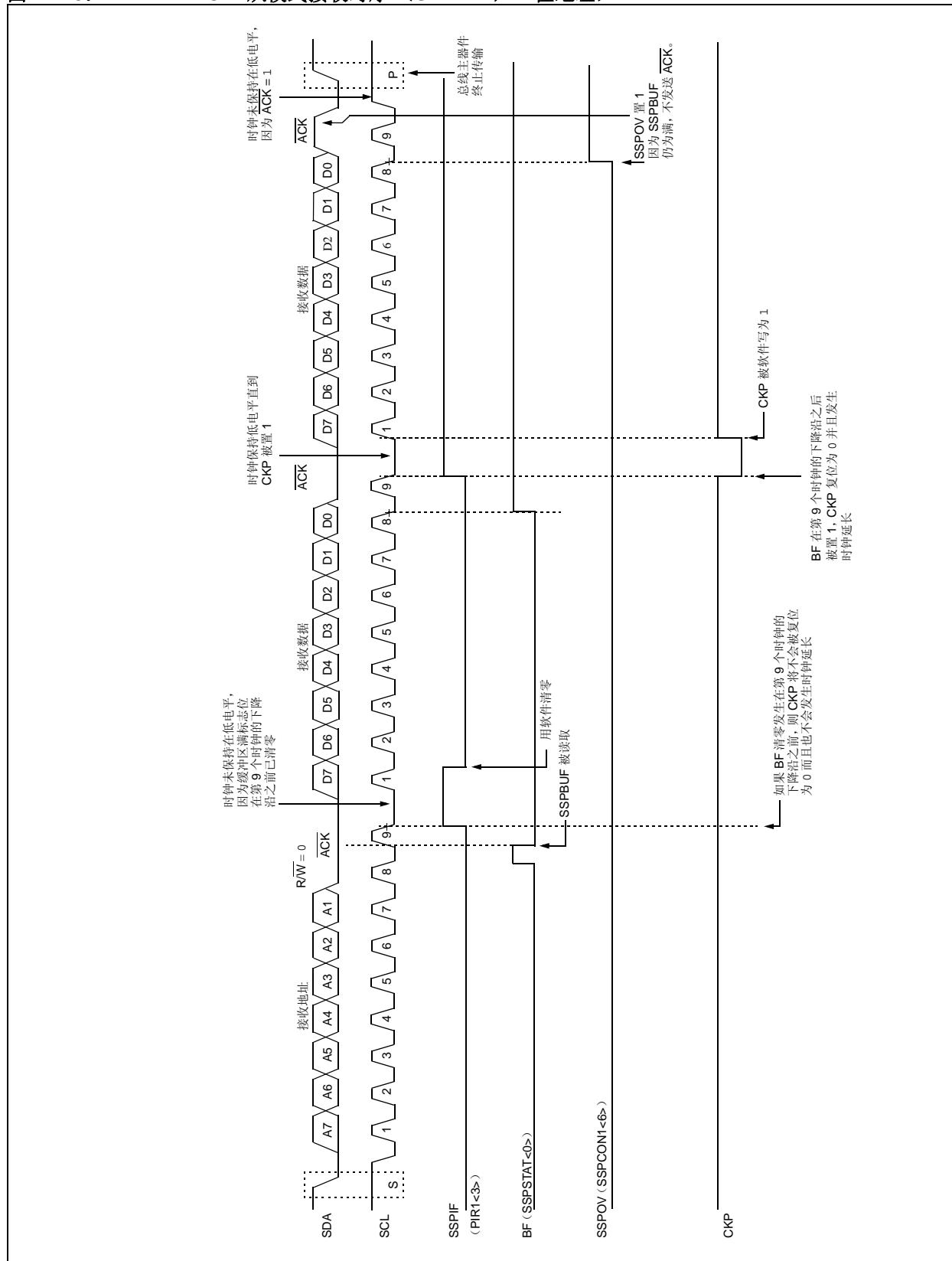
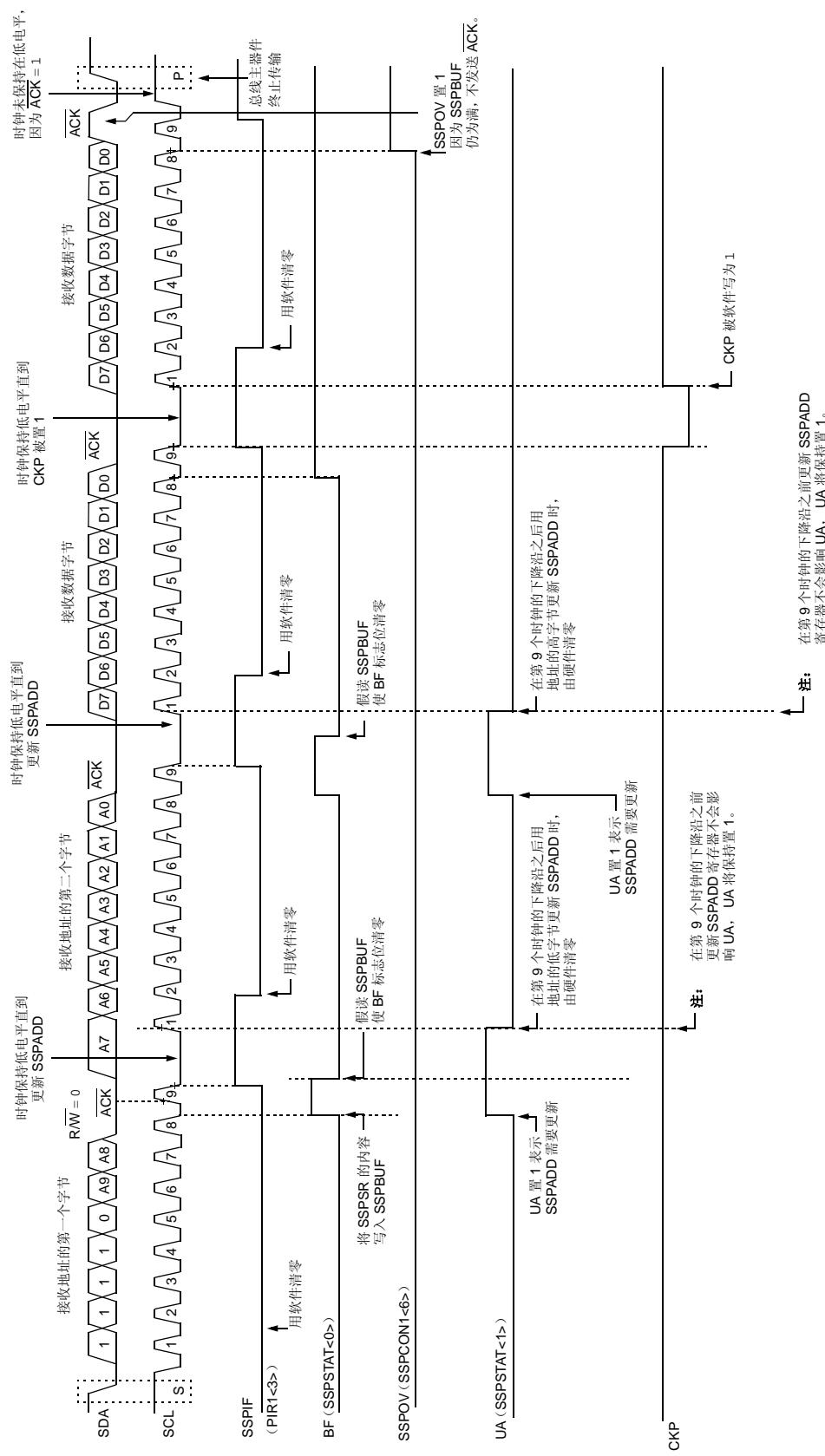


图 17-14: I²CTM 从模式接收时序 (SEN = 1, 10 位地址)



17.4.5 广播呼叫地址支持

在 I²C 总线的寻址过程中，通常由启动条件后的第一个字节决定主器件将寻址哪个从器件。但广播呼叫地址例外，它能寻址所有器件。当使用这个地址时，理论上所有的器件都应该发送一个应答信号来响应。

广播呼叫地址是 I²C 协议为特定目的保留的 8 个地址之一。它由全 0 组成，且 R/W = 0。

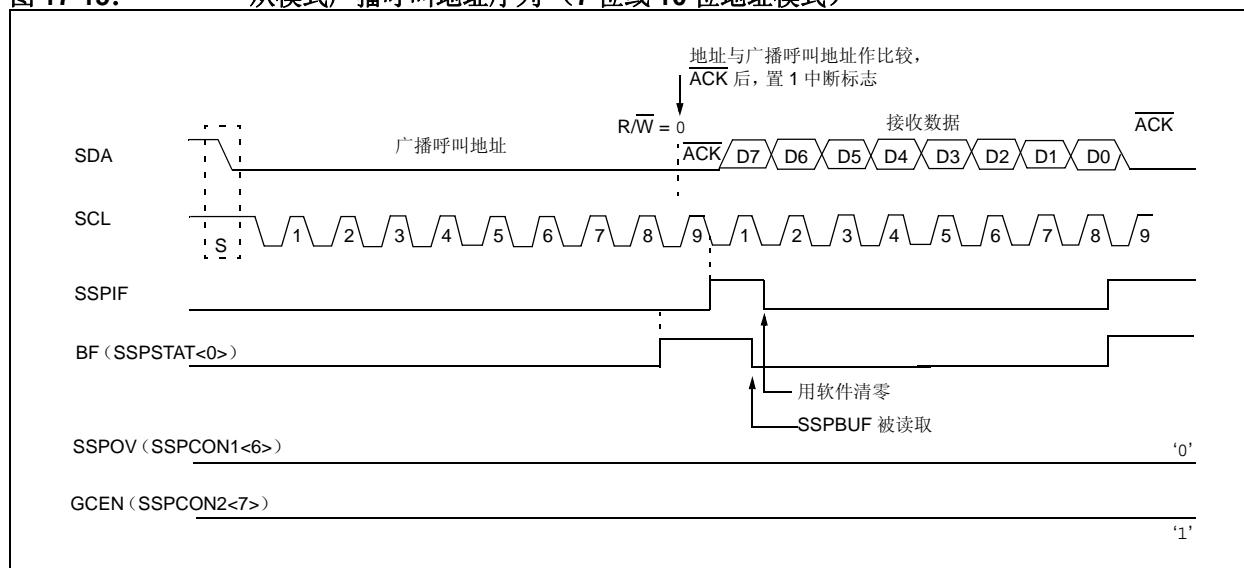
当 SSPCON2 的 GCEN 位置 1 时，可识别广播呼叫地址。检测到启动位后，8 位数据会被移入 SSPSR，同时将该地址与 SSPADD 进行比较。它还会与广播呼叫地址进行比较并用硬件设定。

如果与广播呼叫地址匹配，SSPSR 的值将被传输到 SSPBUF，BF 标志位（第 8 位）置 1，并且 SSPIF 中断标志位在第 9 位（ACK 位）的下降沿置 1。

当中断得到响应时，可以通过读取 SSPBUF 的内容来检查中断源。该值可用于判断是特定器件的地址还是一个广播呼叫地址。

在 10 位模式下，需要更新 SSPADD 用来匹配地址的后半部分，同时 SSPSTAT 寄存器的 UA 位置 1。如果 GCEN 位置 1 时采样到广播呼叫地址，同时从器件被配置为 10 位地址模式，则不再需要地址的后半部分，也不会将 UA 位置 1，从器件将在应答后开始接收数据（图 17-15）。

图 17-15：从模式广播呼叫地址序列（7 位或 10 位地址模式）



17.4.6 主模式

通过将 SSPCON1 中的相应 SSPM 位置 1 和清零，同时将 SSPEN 位置 1，可以使能主模式。在主模式下，SCL 和 SDA 线由 MSSP 硬件控制。

主模式通过在检测到启动条件和停止条件时产生中断来工作。停止 (P) 位和启动 (S) 位在复位或禁止 MSSP 模块时清零。当 P 位置 1 时，可以取得 I²C 总线的控制权；或者，总线处于空闲状态，S 位和 P 位都清零。

在固件控制的主模式下，用户代码根据启动位和停止位条件执行所有的 I²C 总线操作。

一旦使能主模式，用户即可选择以下 6 项操作。

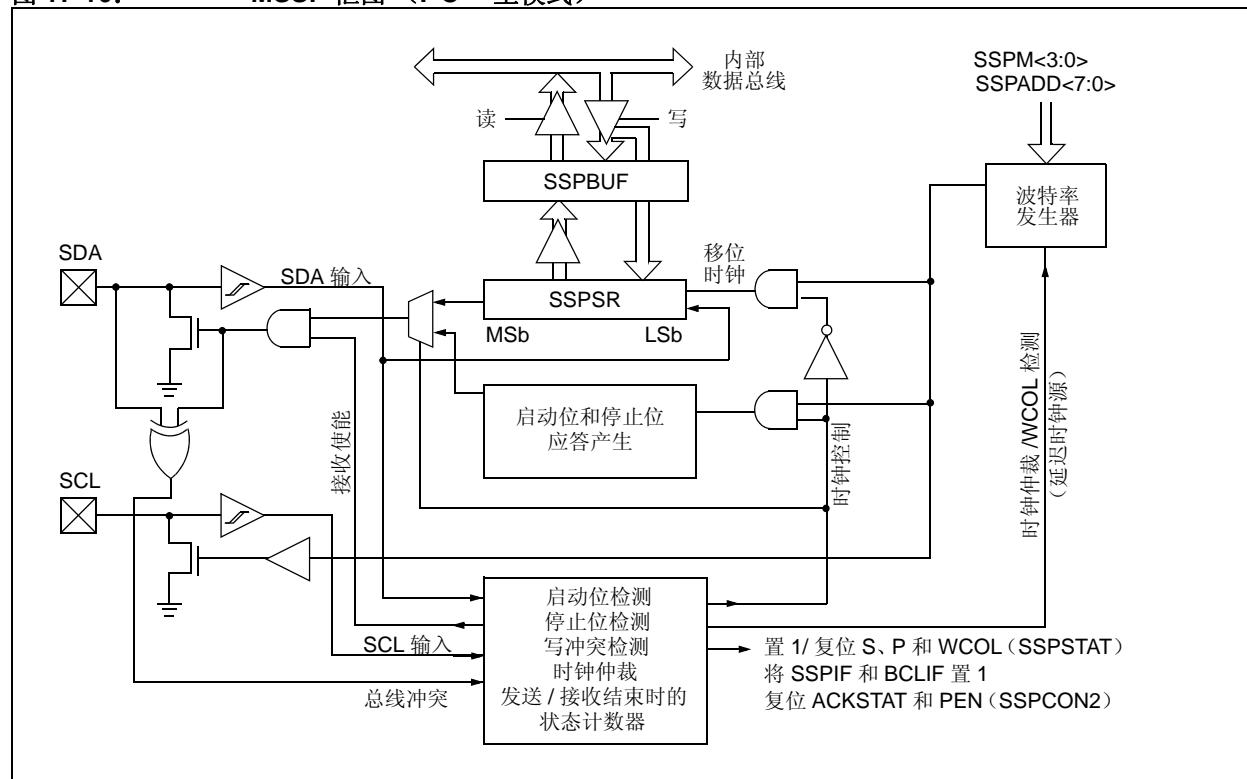
1. 在 SDA 和 SCL 上发出一个启动条件。
2. 在 SDA 和 SCL 上发出一个重复启动条件。
3. 写入 SSPBUF 寄存器，启动数据 / 地址的发送。
4. 配置 I²C 端口用于接收数据。
5. 在接收数据字节末尾产生应答信号。
6. 在 SDA 和 SCL 上产生一个停止条件。

注：当配置为 I²C 主模式时，MSSP 模块不允许事件排队。例如，在启动条件结束前，不允许用户立即写 SSPBUF 寄存器以启动传输。在这种情况下，将不会执行写 SSPBUF，WCOL 位将被置 1，这表明没有发生对 SSPBUF 的写操作。

以下事件会使 SSP 中断标志位 SSPIF 置 1（如果允许 SSP 中断，则产生中断）：

- 启动条件
- 停止条件
- 数据传输字节发送 / 接收
- 应答发送
- 重复启动

图 17-16：MSSP 框图 (I²C™ 主模式)



17.4.6.1 I²C 主模式工作原理

主器件产生所有的串行时钟脉冲、启动条件和停止条件。以停止条件或重复启动条件结束传输过程。因为重复启动条件也是下一次串行传输的开始，因此 I²C 总线不会被释放。

在主发送器模式下，串行数据通过 SDA 输出，而串行时钟由 SCL 输出。发送的第一个字节包括作为接收方的从器件地址（7 位）和读 / 写（R/W）位。在这种情况下，R/W 位将是逻辑 0。一次发送 8 位串行数据。每发送一个字节，会收到一个应答位。输出启动和停止条件，表明串行传输的开始和结束。

在主接收模式下，发送的第一个字节包括作为发送方的从器件地址（7 位）和 R/W 位。在这种情况下，R/W 将是逻辑 1。因此，发送的第一个字节是一个 7 位从器件地址，后面跟 1 表示接收。串行数据通过 SDA 接收，而串行时钟由 SCL 输出。一次接收 8 位串行数据。每接收到一个字节，都会发送一个应答位。启动和停止条件分别表明发送的开始和结束。

在 I²C 模式下，将使用 SPI 模式中的波特率发生器将 SCL 时钟频率设置为 100 kHz、400 kHz 或 1 MHz。更多详细信息，请参见第 17.4.7 节“波特率”。

下面是一个典型的发送序列：

1. 用户通过将 SSPCON2 寄存器的 SEN 位置 1，产生启动条件。
2. SSPIF 置 1。在进行任何其他操作前，MSSP 模块将等待所需的启动时间。
3. 用户将从器件地址装入 SSPBUF 进行发送。
4. 器件地址从 SDA 引脚移出，直到发送完所有 8 位地址数据。
5. MSSP 模块移入来自从器件的 $\overline{\text{ACK}}$ 位，并将它的值写入 SSPCON2 寄存器的 ACKSTAT 位。
6. MSSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
7. 用户将 8 位数据装入 SSPBUF。
8. 数据从 SDA 引脚移出，直到发送完所有 8 位数据。
9. MSSP 模块移入来自从器件的 $\overline{\text{ACK}}$ 位，并将它的值写入 SSPCON2 寄存器的 ACKSTAT 位。
10. MSSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
11. 用户通过将 SSPCON2 寄存器的 PEN 位置 1，产生停止条件。
12. 一旦停止条件完成，将产生一个中断。

17.4.7 波特率

在 I²C 主模式下，波特率发生器 (Baud Rate Generator, BRG) 的重载值存放在 SSPADD 寄存器中（图 17-17）。当发生对 SSPBUF 的写操作时，波特率发生器将自动开始计数。BRG 会递减计数至 0，然后停止直到再次发生重载。BRG 计数器会在每个指令周期 (TCY) 中的 Q2 和 Q4 时钟周期上进行两次递减计数。在 I²C 主模式下，会自动重载 BRG。SCL 周期的一半等于 $[(SSPADD+1) \cdot 2]/FOSC$ 。因此 $SSPADD = (FCY/FOSC) - 1$ 。

如果指定操作完成（即，在传输的最后一个数据位后面跟着 ACK），内部时钟将自动停止计数，SCL 引脚将保持在其最后的状态。

表 17-3 给出了不同的指令周期下的时钟频率以及装入 SSPADD 的 BRG 值。

波特率发生器的最小 SSPADD 值为 0x03。

图 17-17： 波特率发生器框图

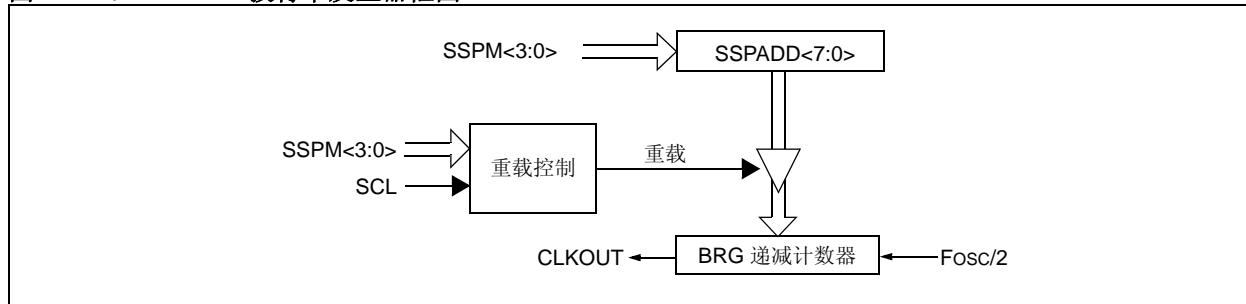


表 17-3： 使用 BRG 的 I²CTM 时钟频率

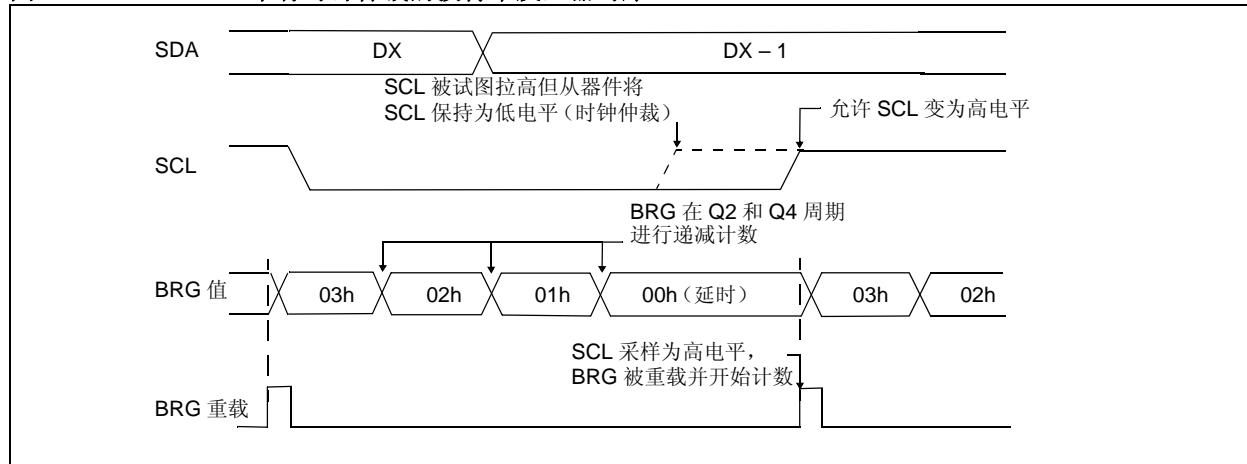
Fosc	FCY	BRG 值	Fscl (两次 BRG 计满返回)
64 MHz	16 MHz	27h	400 kHz ⁽¹⁾
64 MHz	16 MHz	32h	313.7 kHz
64 MHz	16 MHz	3Fh	250 kHz
40 MHz	10 MHz	18h	400 kHz ⁽¹⁾
40 MHz	10 MHz	1Fh	312.5 kHz
40 MHz	10 MHz	63h	100 kHz
16 MHz	4 MHz	09h	400 kHz ⁽¹⁾
16 MHz	4 MHz	0Ch	308 kHz
16 MHz	4 MHz	27h	100 kHz
4 MHz	1 MHz	09h	100 kHz

注 1： 虽然 I²C 接口各方面都不符合 400 kHz I²C 规范（该规范适用于大于 100 kHz 的频率），但在需要较高频率的应用场合可以慎重使用。

17.4.7.1 时钟仲裁

如果在任何接收、发送或重复启动 / 停止条件期间，主器件拉高了 SCL 引脚（允许 SCL 引脚悬空为高电平），就会发生时钟仲裁。当允许 SCL 引脚悬空为高电平时，波特率发生器（BRG）暂停计数，直到 SCL 引脚被实际采样到高电平为止。然后波特率发生器将被重新装入 SSPADD<7:0> 的内容并开始计数。这可以保证当外部器件将时钟拉低时，SCL 在至少一个 BRG 计满返回计数周期内保持高电平（图 17-18）。

图 17-18：带有时钟仲裁的波特率发生器时序



17.4.8 I²C 主模式启动条件时序

要发出启动条件，用户应将 SSPCON2 寄存器的启动使能位 SEN 置 1。当 SDA 和 SCL 引脚采样为高电平时，波特率发生器重新装入 SSPADD<7:0> 的内容并开始计数。如果波特率发生器发生超时（TBRG）时，SCL 和 SDA 都采样为高电平，则 SDA 引脚被驱动为低电平。当 SCL 为高电平时，将 SDA 驱动为低电平将产生启动条件，并使 SSPSTAT1 寄存器的 S 位置 1。随后波特率发生器重新装入 SSPADD<7:0> 的内容并恢复计数。当波特率发生器再次超时（TBRG）时，SSPCON2 寄存器的 SEN 位将自动被硬件清零，波特率发生器暂停工作，SDA 线保持低电平，启动条件结束。

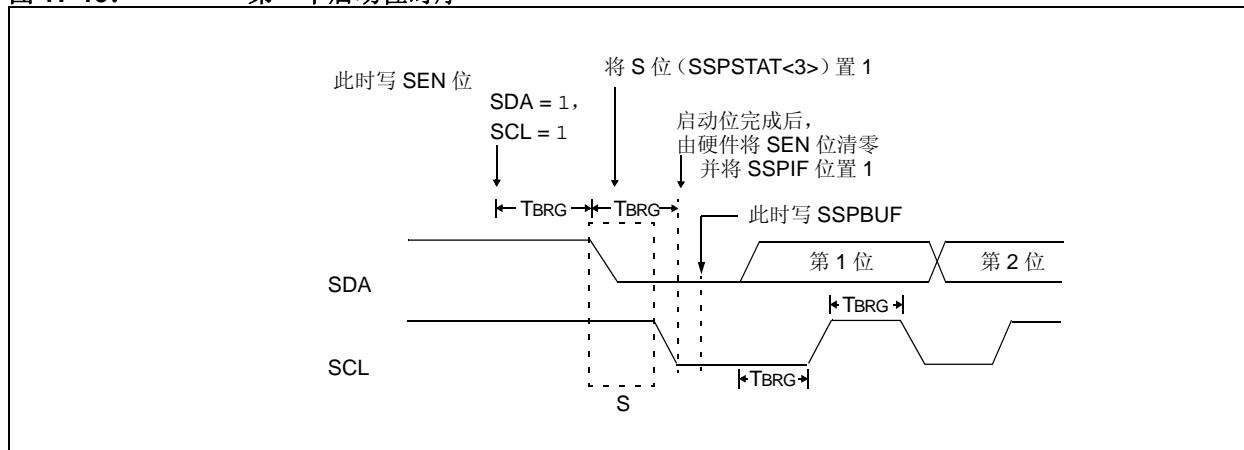
注： 如果在启动条件开始时，SDA 和 SCL 引脚已经采样为低电平，或者在启动条件期间，SCL 在 SDA 线被驱动为低电平之前已经采样为低电平，则会发生总线冲突。总线冲突中断标志位 BCLIF 置 1，启动条件中止，I²C 模块复位到空闲状态。

17.4.8.1 WCOL 状态标志

如果用户在启动序列过程中写 SSPBUF，则 WCOL 被置 1，同时缓冲区内容不变（写操作无效）。

注： 由于不允许事件排队，在启动条件结束之前，不能写 SSPCON2 的低 5 位。

图 17-19：第一个启动位时序



17.4.9 I²C 主模式重复启动条件时序

将 SSPCON2 寄存器的 RSEN 位编程为高电平，并且 I²C 逻辑模块处于空闲状态时，就会产生重复启动条件。当 RSEN 位置 1 时，SCL 引脚被拉为低电平。当 SCL 引脚采样为低电平时，波特率发生器装入 SSPADD<7:0> 的值并开始计数。在该波特率发生器计数周期 (TBRG) 内 SDA 引脚被释放（被拉高）。当波特率发生器超时时，如果 SDA 采样为高电平，SCL 引脚将被拉高。当 SCL 被采样为高电平时，波特率发生器重新装入 SSPADD<7:0> 的内容并开始计数。SDA 和 SCL 必须在一个计数周期 TBRG 内采样为高电平。接下来，在一个 TBRG 中，将 SDA 引脚驱动为低电平 (SDA = 0)，同时 SCL 保持高电平。随后 SSPCON2 寄存器的 RSEN 位将自动清零，这次波特率发生器不会重载，SDA 引脚保持低电平。一旦在 SDA 和 SCL 引脚上检测到启动条件，SSPSTAT 寄存器的 S 位将被置 1。直到波特率发生器发生超时后，SSPIF 位才会置 1。

注 1: 有任何其他事件在进行时，编程设置 RSEN 无效。

2: 在重复启动条件期间，以下事件将会导致总线冲突：

- 当 SCL 由低电平变为高电平时，SDA 采样为低电平。
- 在 SDA 被拉低之前，SCL 变为低电平。这表明另一个主器件正试图发送一个数据 1。

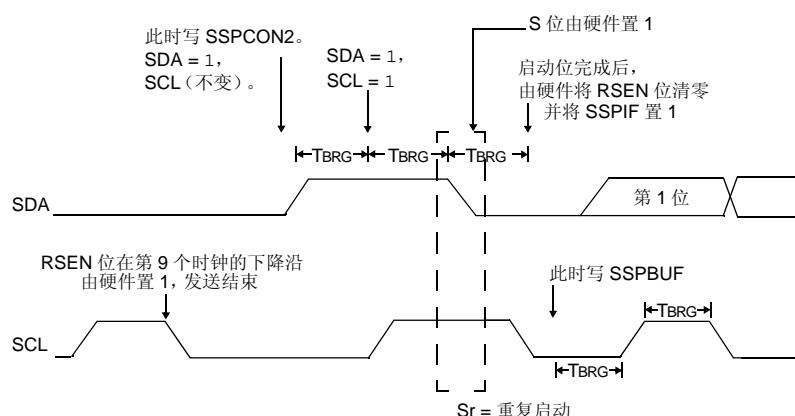
一旦 SSPIF 位被置 1，用户便可以在 7 位地址模式下将 7 位地址，或者在 10 位地址模式下将默认的第一个地址字节写入 SSPBUF。当发送完第一个 8 位数据并接收到一个 ACK 后，用户可以发送另外 8 位地址 (10 位模式) 或 8 位数据 (7 位模式)。

17.4.9.1 WCOL 状态标志

如果用户在重复启动序列过程中写 SSPBUF，则 WCOL 被置 1，同时缓冲区内容不变（写操作无效）。

注： 由于不允许事件排队，在重复启动条件结束之前，不能写 SSPCON2 的低 5 位。

图 17-20：重复启动条件波形图



17.4.10 I²C 主模式发送

发送一个数据字节、一个 7 位地址或一个 10 位地址的一半，都是通过写一个值到 SSPBUF 寄存器来实现的。该操作将使缓冲区满标志位 BF 置 1，波特率发生器开始计数，同时开始下一次发送。在 SCL 的下降沿有效后（见数据保持时间规范参数 106），地址 / 数据的每一位被移出至 SDA 引脚。在一个波特率发生器计满返回计数周期（TBRG）内，SCL 保持低电平。数据应该在 SCL 释放为高电平前保持有效（见数据建立时间规范参数 107）。当 SCL 引脚释放为高电平时，它将在一个 TBRG 内保持高电平状态。在此期间以及 SCL 的下一个下降沿之后的一段时间内，SDA 引脚上的数据必须保持稳定。在第 8 位数据被移出（第 8 个时钟周期的下降沿）之后，BF 标志位被清零，同时主器件释放 SDA。此时如果发生地址匹配或是数据被正确接收，被寻址的从器件将在第 9 个时钟周期发出一个 ACK 位作为响应。ACK 的状态在第 9 个时钟周期的下降沿写入 ACKDT 位。主器件接收到应答之后，应答状态位 ACKSTAT 会被清零。如果未收到应答，则该位被置 1。第 9 个时钟周期之后，SSPIF 位会置 1，主时钟（波特率发生器）暂停，直到下一个数据字节装入 SSPBUF，SCL 引脚保持低电平，SDA 保持不变（图 17-21）。

在写 SSPBUF 之后，地址的每一位在 SCL 的下降沿被移出，直到所有 7 个地址位和 R/W 位都被移出。在第 8 个时钟的下降沿，主器件将 SDA 引脚拉为高电平，以允许从器件发出一个应答响应。在第 9 个时钟的下降沿，主器件通过采样 SDA 引脚来判断地址是否被从器件识别。ACK 位的状态被装入 SSPCON2 寄存器的 ACKSTAT 状态位。在发送地址的第 9 个时钟下降沿之后，SSPIF 置 1，BF 标志位清零，波特率发生器关闭直到下一次写 SSPBUF，且 SCL 引脚保持低电平，允许 SDA 引脚悬空。

17.4.10.1 BF 状态标志

在发送模式下，SSPSTAT 寄存器的 BF 位在 CPU 写 SSPBUF 时置 1，在所有 8 位数据移出后清零。

17.4.10.2 WCOL 状态标志

如果用户在发送过程中（即，SSPSR 仍在移出数据字节时）写 SSPBUF，则 WCOL 被置 1，同时缓冲区内容不变（写操作无效）。

WCOL 必须用软件清零。

17.4.10.3 ACKSTAT 状态标志

在发送模式下，当从器件发送应答响应（ACK = 0）时，SSPCON2 寄存器的 ACKSTAT 位清零；当从器件没有应答（ACK = 1）时，该位置 1。从器件在识别出其地址（包括广播呼叫地址）或正确接收数据后，会发送一个应答。

17.4.11 I²C 主模式接收

通过编程 SSPCON2 寄存器的接收使能位 RCEN 使能主模式接收。

注： 将 RCEN 位置 1 前，MSSP 模块必须处于空闲状态，否则对 RCEN 位置 1 将无效。

波特率发生器开始计数，每次计满返回时，SCL 引脚的状态发生改变（由高变低或由低变高），数据被移入 SSPSR。第 8 个时钟的下降沿之后，接收使能标志位自动清零，SSPSR 的内容装入 SSPBUF，BF 标志位置 1，SSPIF 标志位置 1，波特率发生器暂停计数，且 SCL 保持为低电平。此时 MSSP 处于空闲状态，等待下一条命令。当 CPU 读缓冲区时，BF 标志位将自动清零。通过将 SSPCON2 寄存器的应答序列使能位 ACKEN 置 1，用户可以在接收结束后发送应答位。

17.4.11.1 BF 状态标志

接收数据过程中，把地址或数据字节从 SSPSR 装入 SSPBUF 时，BF 位置 1。在读 SSPBUF 寄存器时将其清零。

17.4.11.2 SSPOV 状态标志

接收数据过程中，当 SSPSR 接收到 8 位数据时，SSPOV 位置 1，BF 标志位已经在上一次接收时置 1。

17.4.11.3 WCOL 状态标志

如果用户在接收过程中（即，SSPSR 仍在移入数据字节时）写 SSPBUF，则 WCOL 位被置 1，同时缓冲区内容不变（写操作无效）。

PIC18F2XK20/4XK20

图 17-21: I²C™ 主模式发送波形图 (7 位或 10 位地址)

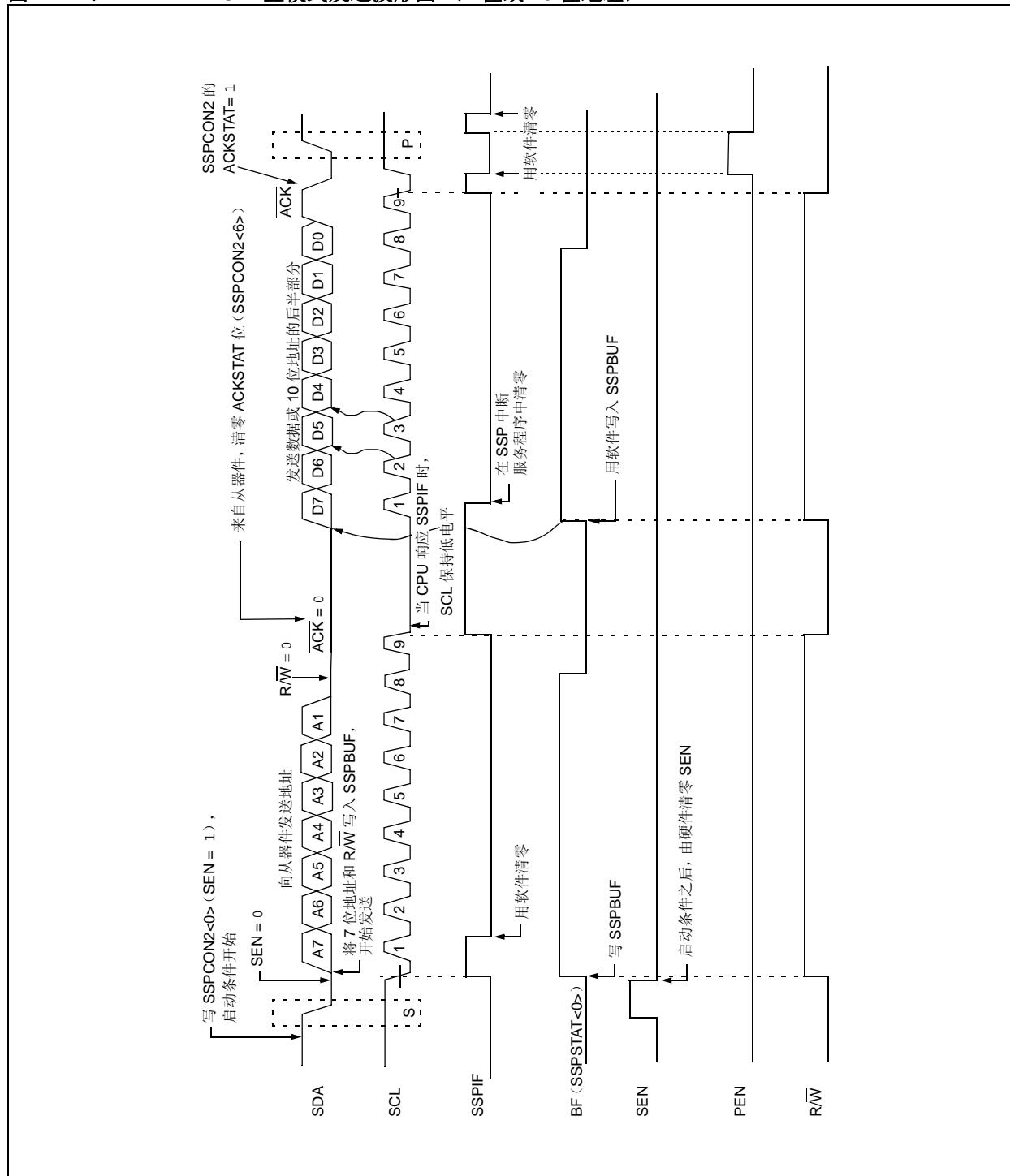
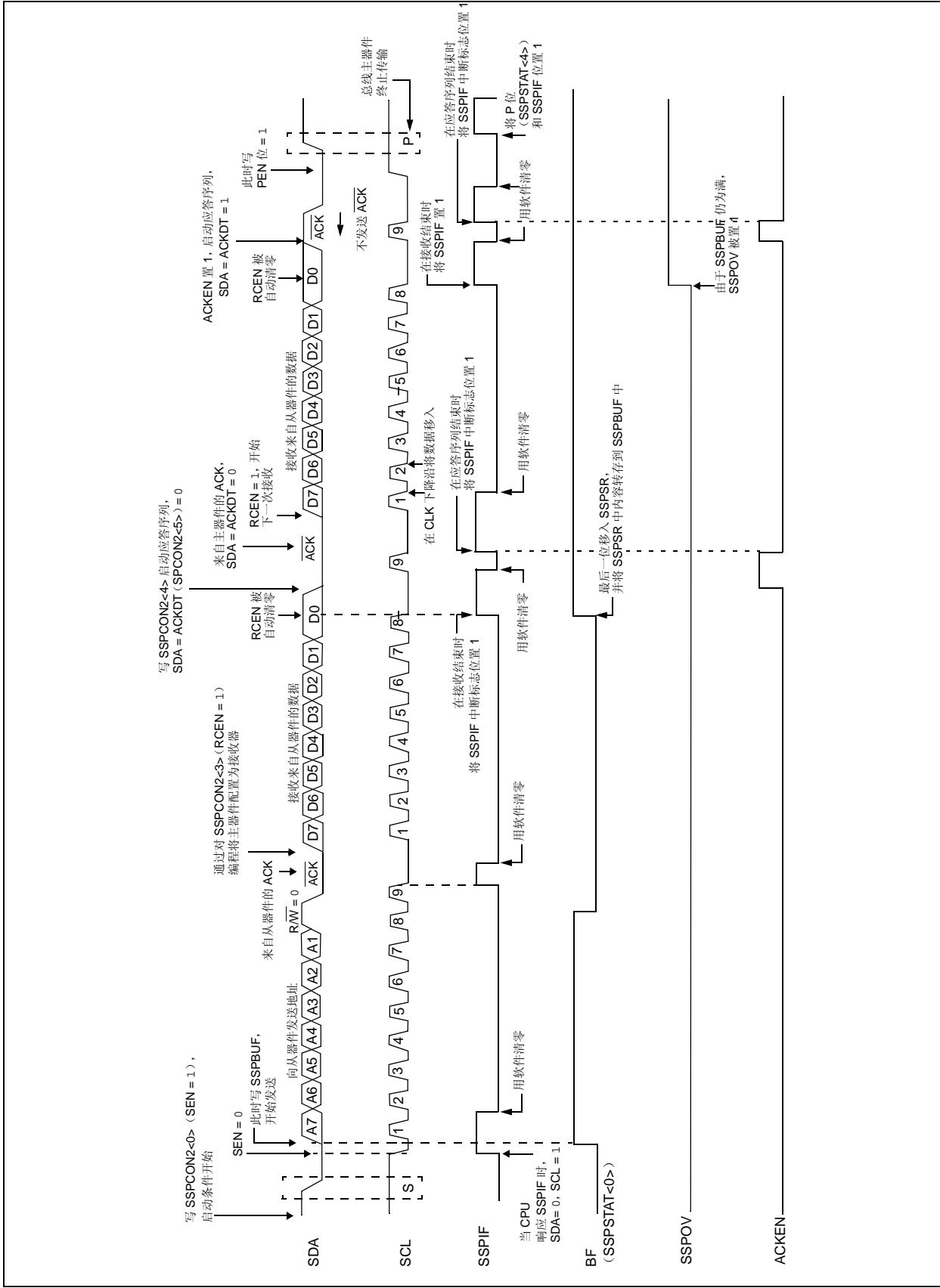


图 17-22: I²CTM 主模式接收波形图 (7 位地址)



17.4.12 应答序列时序

将 SSPCON2 寄存器的应答序列使能位 ACKEN 置 1 即可使能应答序列。当该位被置 1 时，SCL 引脚被拉低，应答数据位的内容输出到 SDA 引脚上。如果用户希望产生一个应答，则应该将 ACKDT 位清零。否则，用户要在应答序列开始前将 ACKDT 位置 1。然后波特率发生器进行一个计满返回周期 (TBRG) 的计数，随后 SCL 引脚电平被拉高。当 SCL 引脚采样为高电平（时钟仲裁）时，波特率发生器再进行一个 TBRG 周期的计数。然后 SCL 引脚被拉低。在这之后，ACKEN 位自动清零，波特率发生器关闭，MSSP 模块进入空闲模式（图 17-23）。

17.4.12.1 WCOL 状态标志

如果用户在应答序列进行过程中写 SSPBUF，则 WCOL 被置 1，同时缓冲区内容不变（写操作无效）。

图 17-23：应答序列波形图

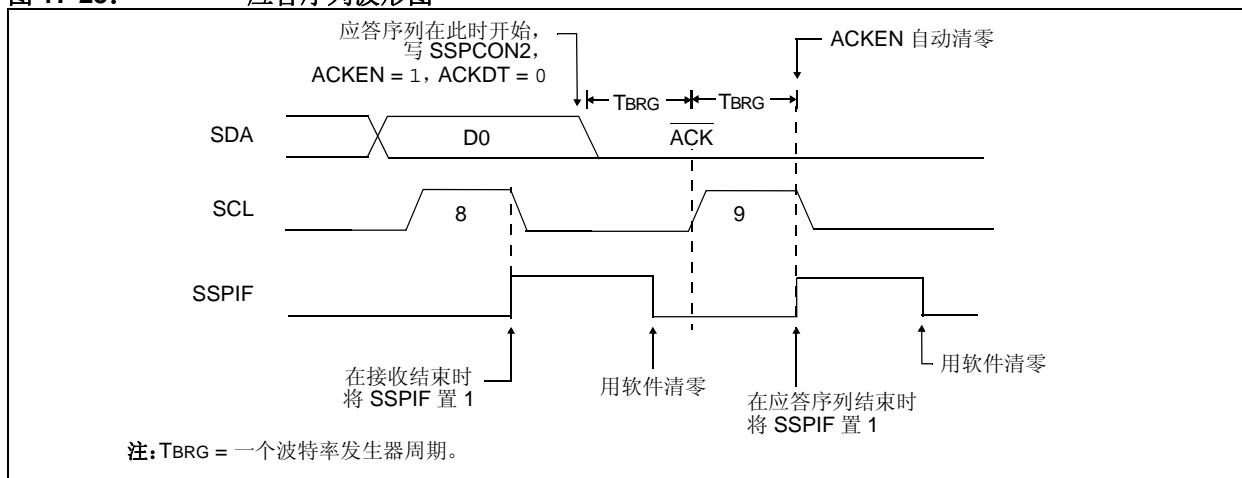
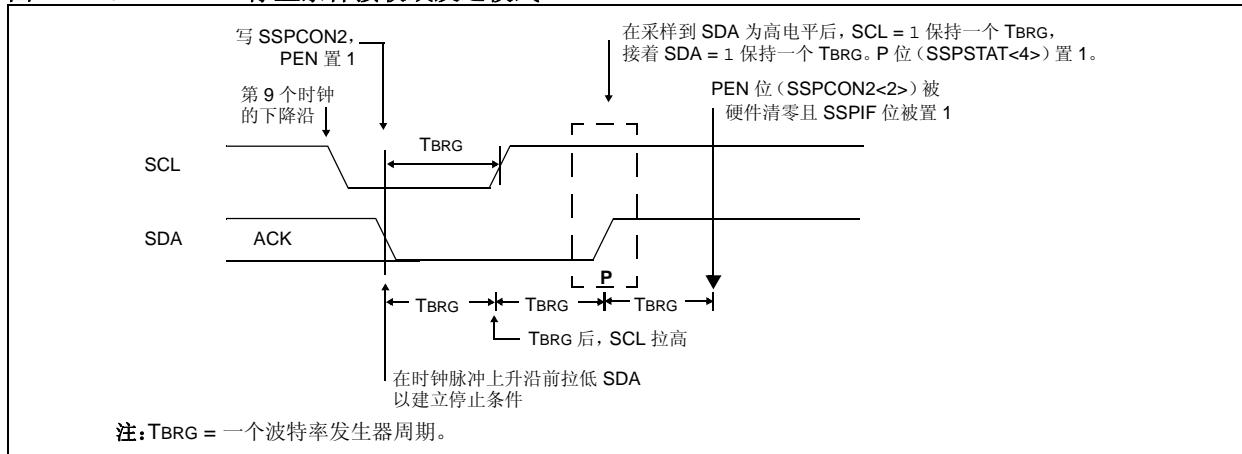


图 17-24：停止条件接收或发送模式



17.4.14 休眠模式下的操作

在休眠模式下, I²C 模块能够接收地址或数据, 并且在地址匹配或字节传输完成后, 如果允许 MSSP 中断, 将唤醒处理器。

17.4.15 复位的影响

复位会禁止 MSSP 模块并终止当前的数据传输。

17.4.16 多主模式

在多主模式下, 在检测到启动和停止条件时将产生中断, 这可用于判断总线是否空闲。停止 (P) 位和启动 (S) 位在复位或禁止 MSSP 模块时清零。当 SSPSTAT 寄存器的 P 位置 1 时, 可以取得 I²C 总线的控制权; 或者, 总线处于空闲状态, S 位和 P 位都清零。当总线忙时, 当发生停止条件时, 将产生 SSP 中断。

在多主模式下, 必须一直监视 SDA 线来进行仲裁, 查看信号电平是否为期望的输出电平。此操作由硬件实现, 其结果保存在 BCLIF 位中。

可能导致仲裁失败的情况是:

- 地址传输
- 数据传输
- 启动条件
- 重复启动条件
- 应答条件

17.4.17 多主器件通信、总线冲突和总线仲裁

多主模式是通过总线仲裁来支持的。当主器件将地址 / 数据位输出到 SDA 引脚时, 如果一个主器件在 SDA 上输出 1 (将 SDA 引脚悬空为高电平), 而另一个主器件输出 0, 就会发生总线仲裁。如果 SDA 引脚上期望的数据是 1, 而实际采样到的数据是 0, 则发生了总线冲突。主器件将把总线冲突中断标志位 BCLIF 置 1, 并将 I²C 端口复位到空闲状态 (图 17-25)。

如果在发送过程中发生总线冲突, 则发送操作停止, BF 标志位被清零, SDA 和 SCL 线被拉高, 并且可写入 SSPBUF。当执行完总线冲突中断服务程序时, 如果 I²C 总线空闲, 用户可通过发出启动条件恢复通信。

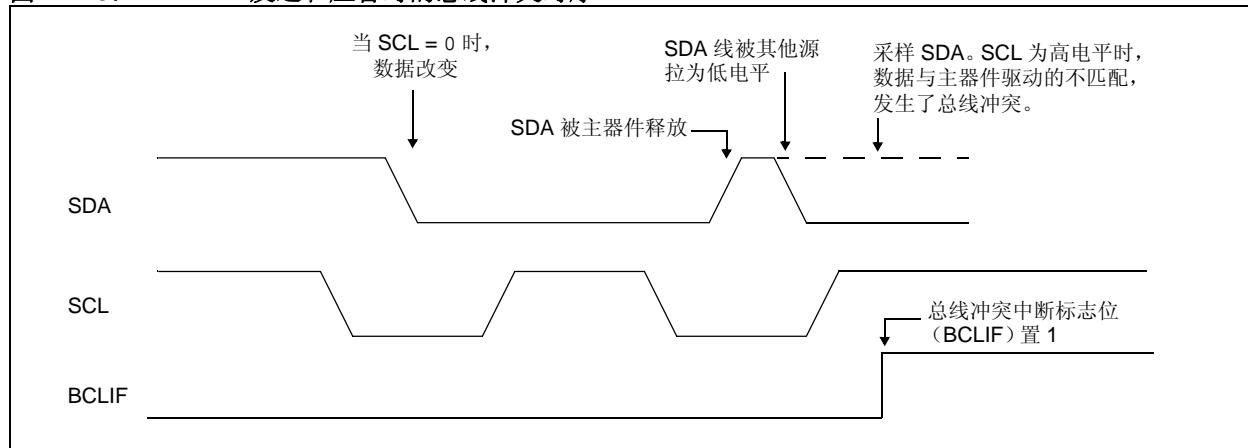
如果在启动、重复启动、停止或应答信号的执行过程中发生总线冲突, 则这种状态被中止, SDA 和 SCL 线被拉高, SSPCON2 寄存器中的对应控制位清零。当执行完总线冲突中断服务程序时, 如果 I²C 总线空闲, 用户可通过发出启动条件恢复通信。

主器件将继续监视 SDA 和 SCL 引脚。一旦出现停止条件, SSPIF 位将被置 1。

发生总线冲突时无论发送的进度如何, 写入 SSPBUF 都会从第一个数据位开始发送数据。

在多主模式下, 通过在检测到启动条件和停止条件时产生中断可以确定总线何时空闲。SSPSTAT 寄存器中的 P 位置 1 时, 可以取得 I²C 总线的控制权; 或者, 总线处于空闲状态, S 位和 P 位都清零。

图 17-25: 发送和应答时的总线冲突时序



17.4.17.1 启动条件期间的总线冲突

启动条件期间，以下事件将导致总线冲突：

- a) 在启动条件开始时，SDA 或 SCL 被采样为低电平（图 17-26）。
- b) SDA 被拉低之前，SCL 采样为低电平（图 17-27）。

在启动条件期间，SDA 和 SCL 引脚都会被监视。

如果 SDA 引脚已经是低电平，或 SCL 引脚已经是低电平，则：

- 中止启动条件，
- BCLIF 标志位置 1，并且
- MSSP 模块复位为空闲状态（图 17-26）。

启动条件从 SDA 和 SCL 引脚被拉高开始。当 SDA 引脚采样为高电平时，波特率发生器装入 SSPADD<7:0> 的内容并递减计数至 0。如果在 SDA 为高电平时，SCL 引脚采样为低电平，则发生总线冲突，因为这表示另一个主器件在启动条件期间试图驱动一个数据 1。

如果 SDA 引脚在该计数周期内采样为低电平，则 BRG 复位，同时 SDA 线保持原值（图 17-28）。但是，如果 SDA 引脚采样为 1，则在 BRG 计数结束时该引脚将被置为低电平。接着，波特率发生器被重载并递减计数至 0；在此期间，如果 SCL 引脚采样到 0，则不会发生总线冲突。在 BRG 计数结束时，SCL 引脚被拉为低电平。

注： 在启动条件期间不太可能发生总线冲突，因为两个总线主器件不可能精确地在同一时刻发出启动条件。因此一个主器件将总是先于另一个主器件将 SDA 拉低。但是上述情况不会引起总线冲突，因为两个主器件一定会对启动条件后的第一个地址进行仲裁。如果地址是相同的，必须继续对数据部分、重复启动条件或停止条件进行仲裁。

图 17-26：启动条件期间的总线冲突（仅用于 SDA）

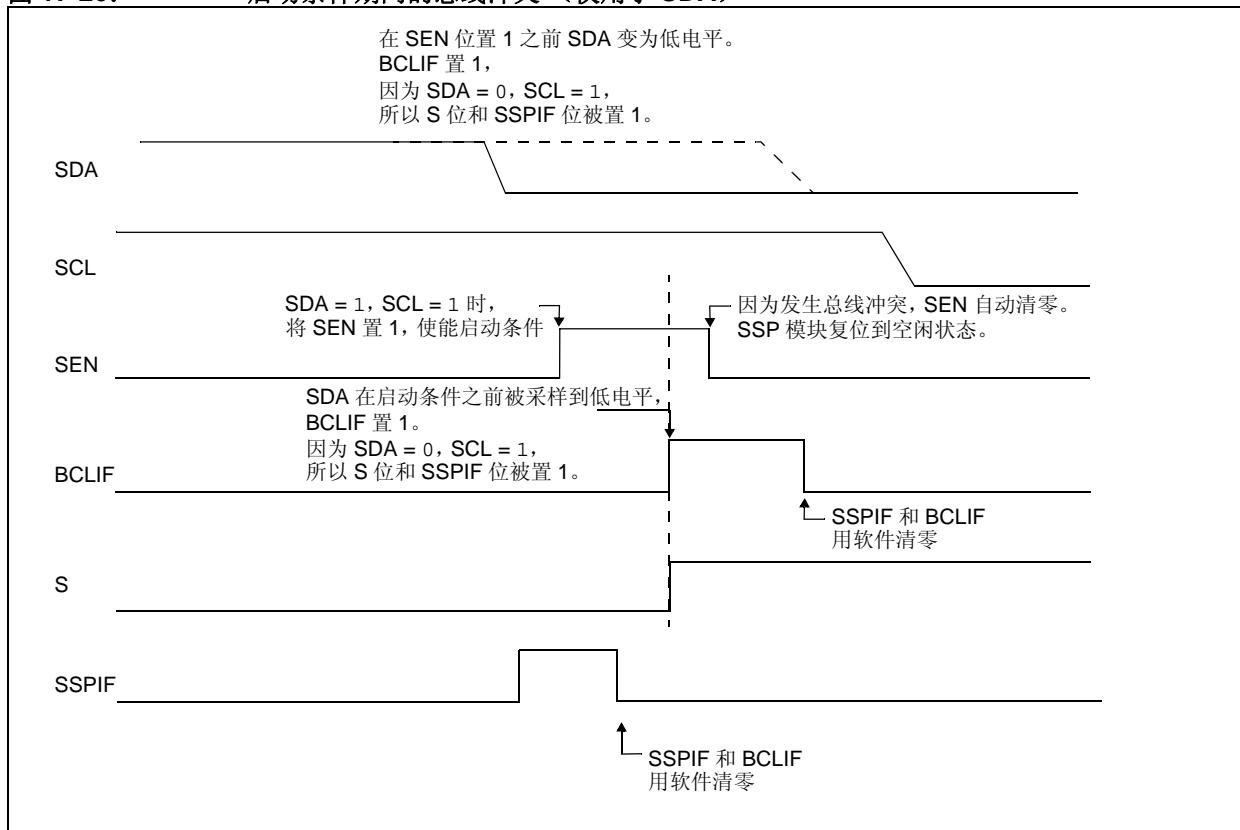


图 17-27：启动条件期间的总线冲突 ($SCL = 0$)

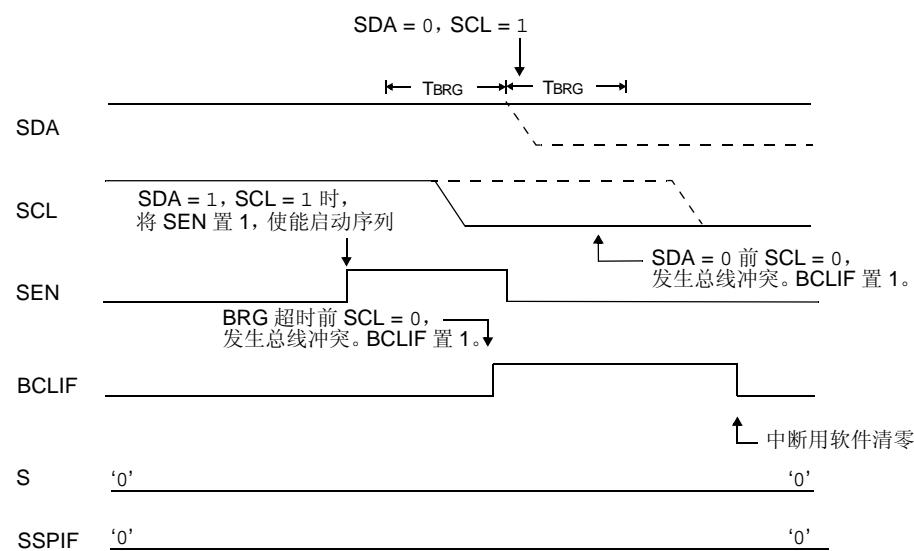
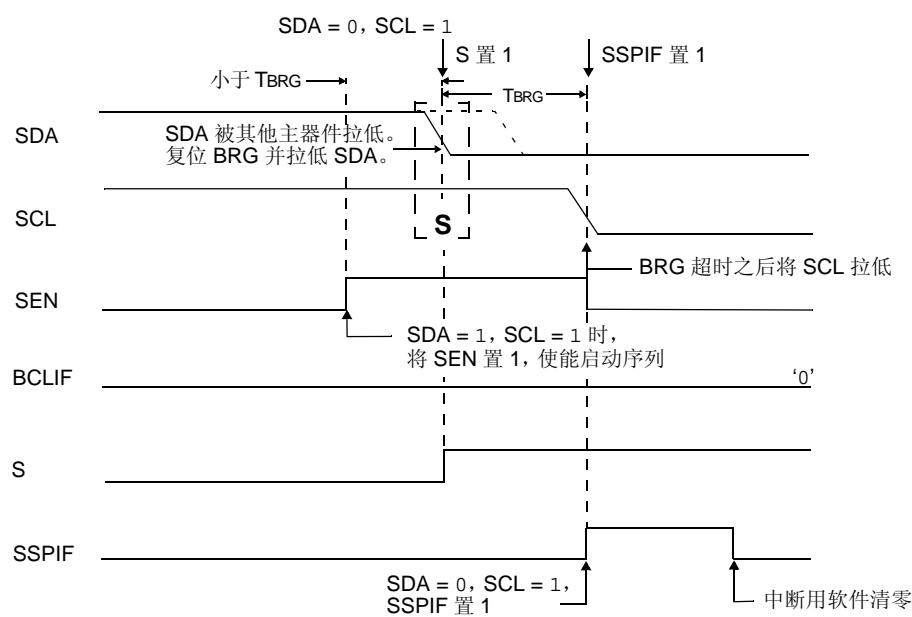


图 17-28：启动条件期间由 SDA 仲裁引起的 BRG 复位



17.4.17.2 重复启动条件期间的总线冲突

在以下情况中，重复启动条件期间会发生总线冲突：

- 在 SCL 由低电平变为高电平期间，在 SDA 上采样到低电平。
- 在 SDA 被拉为低电平之前，SCL 变为低电平，表示另一个主器件正试图发送一个数据 1。

当用户拉高 SDA 并允许该引脚悬空时，BRG 装入 SSPADD<7:0> 中的值并递减计数至 0，接着 SCL 引脚被拉高，当 SCL 引脚采样到高电平时，对 SDA 引脚进行采样。

如果 SDA 为低电平，则已发生了总线冲突（即，另一个主器件正试图发送一个数据 0，见图 17-29）。如果 SDA 被采样到高电平，则 BRG 被重新装入值并开始计数。如果 SDA 在 BRG 超时之前从高电平变为低电平，则不会发生总线冲突，因为两个主器件不可能精确地在同一时刻将 SDA 拉低。

如果 SCL 在 BRG 超时之前从高电平变为低电平，且 SDA 尚未被拉低，那么将发生总线冲突。在此情况下，另一个主器件在重复启动条件期间正试图发送一个数据 1（见图 17-30）。

如果在 BRG 超时结束时 SCL 和 SDA 都仍然是高电平，则 SDA 引脚被拉低，BRG 重新装入值并开始计数。在计数结束时，不管 SCL 引脚的状态如何，SCL 引脚都被拉低，重复启动条件结束。

图 17-29：重复启动条件期间的总线冲突（情形 1）

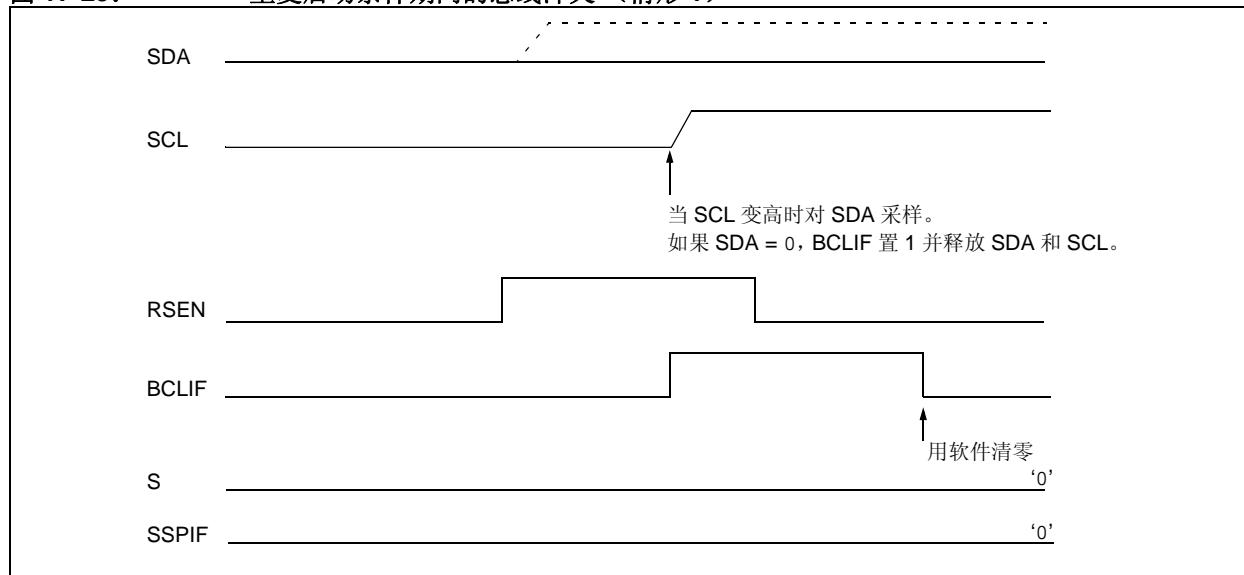
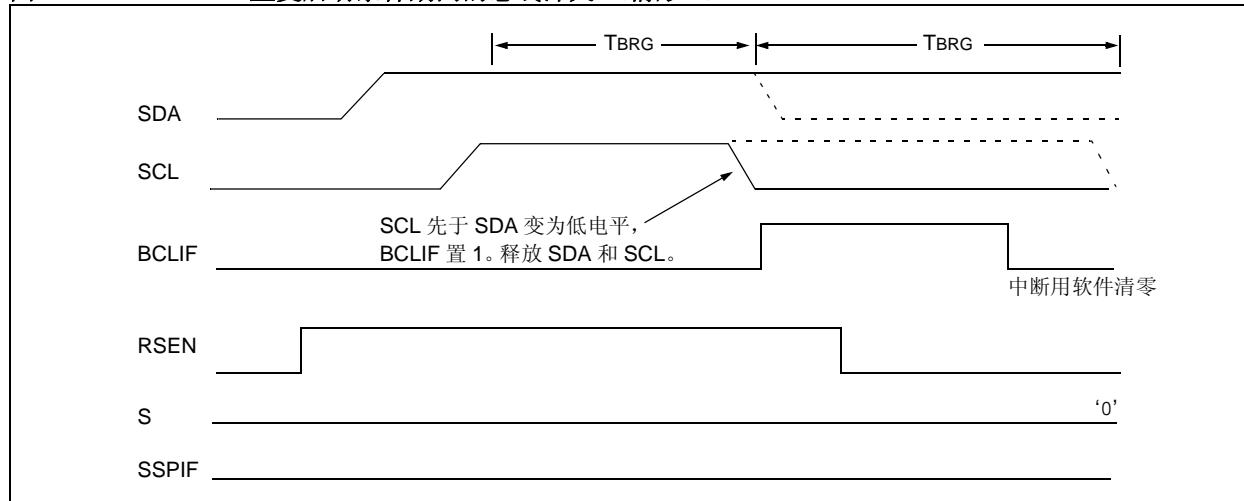


图 17-30：重复启动条件期间的总线冲突（情形 2）



17.4.17.3 停止条件期间的总线冲突

以下事件会导致停止条件期间发生总线冲突：

- SDA已被拉高并允许悬空为高电平之后，SDA在BRG超时后被采样到低电平。
- SCL引脚被拉高之后，SCL在SDA变成高电平之前被采样到低电平。

停止条件从 SDA 被置成低电平开始。当 SDA 采样为低电平时，SCL 引脚被允许悬空。当 SDA 被采样到高电平（时钟仲裁）时，波特率发生器装入 $SSPADD<7:0>$ 的值并递减计数至 0。BRG 超时后，SDA 被采样。如果 SDA 采样为低电平，则已发生总线冲突。这是因为另一个主器件正试图发送一个数据 0（图 17-31）。如果 SCL 引脚在允许 SDA 悬空为高电平前被采样到低电平，也会发生总线冲突。这是另一个主器件正试图发送一个数据 0 的另外一种情况（图 17-32）。

图 17-31：停止条件期间的总线冲突（情形 1）

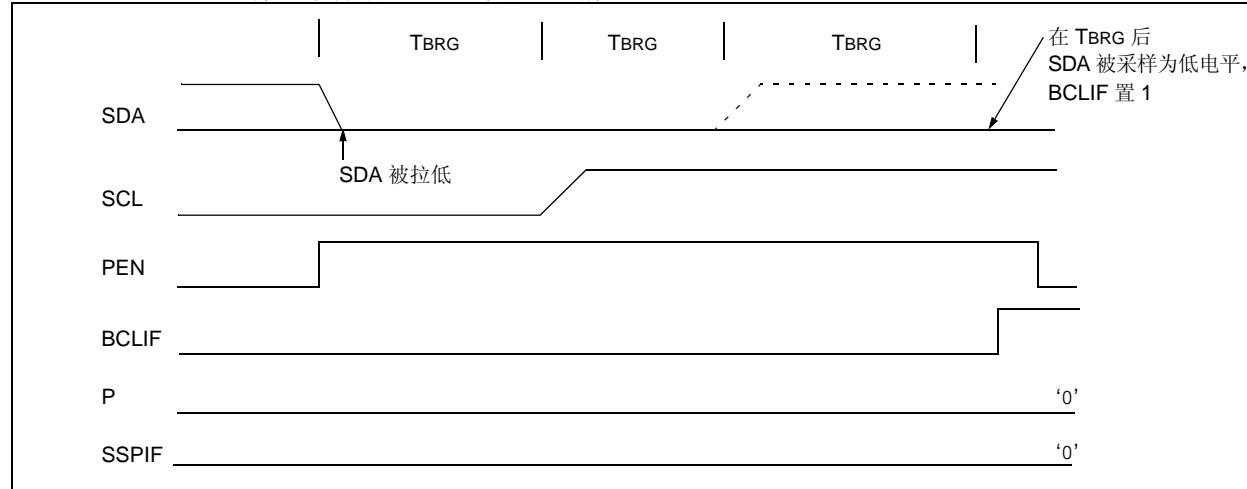
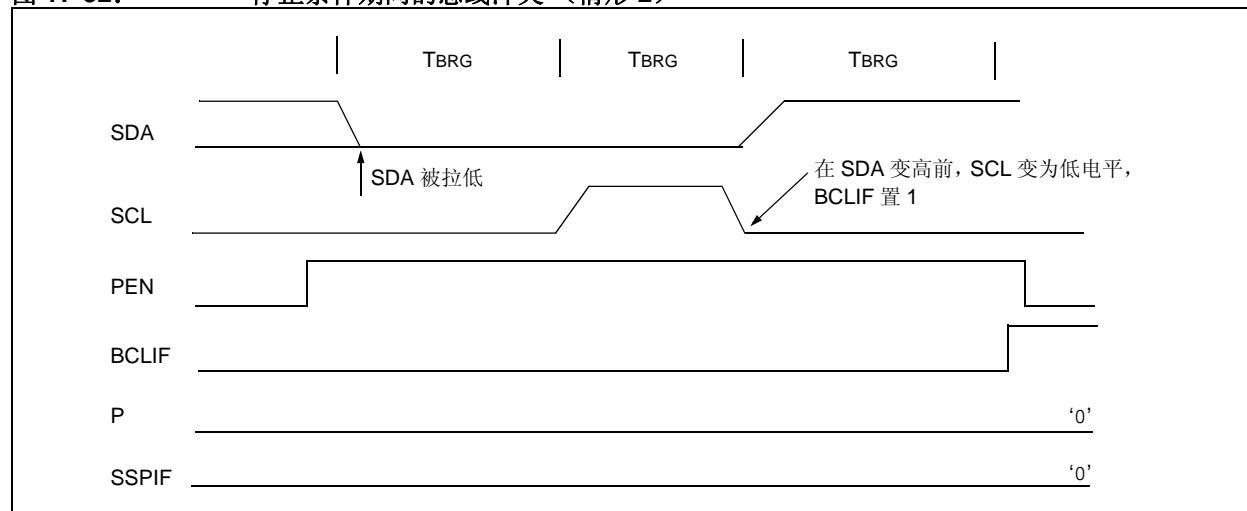


图 17-32：停止条件期间的总线冲突（情形 2）



PIC18F2XK20/4XK20

表 17-4: 与 I²CTM 相关的寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62
SSPADD	I ² C TM 从模式下的 SSP 地址寄存器。 I ² C 主模式下的 SSP 波特率重载寄存器。								60
SSPBUF	SSP 接收缓冲 / 发送寄存器								60
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	60
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	60
SSPMSK	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0	63
SSPSTAT	SMP	CKE	D/A	P	S	R/W	UA	BF	60
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	62

图注: — = 未实现, 读为 0。 I²CTM 模式下的 MSSP 模块不使用阴影单元。

注 1: 在 PIC18F2XK20 器件上未实现。

18.0 增强型通用同步 / 异步收发器 (EUSART)

增强型通用同步 / 异步收发器 (EUSART) 模块是一种串行 I/O 通信外设。它包含用来完成与器件程序执行无关的输入或输出串行数据传输所需的所有时钟发生器、移位寄存器和数据缓冲区等。EUSART 也是一种串行通信接口 (Serial Communications Interface, SCI)，可配置为全双工异步系统或半双工同步系统。全双工模式可用来与外设系统通信，如 CRT 终端和个人计算机。半双工同步模式用于与外设通信，如 A/D 或 D/A 集成电路、串行 EEPROM 或其他单片机。这些器件通常不具备用以产生波特率的内部时钟，并需要由主同步器件提供外部时钟信号。

EUSART 模块具备以下功能：

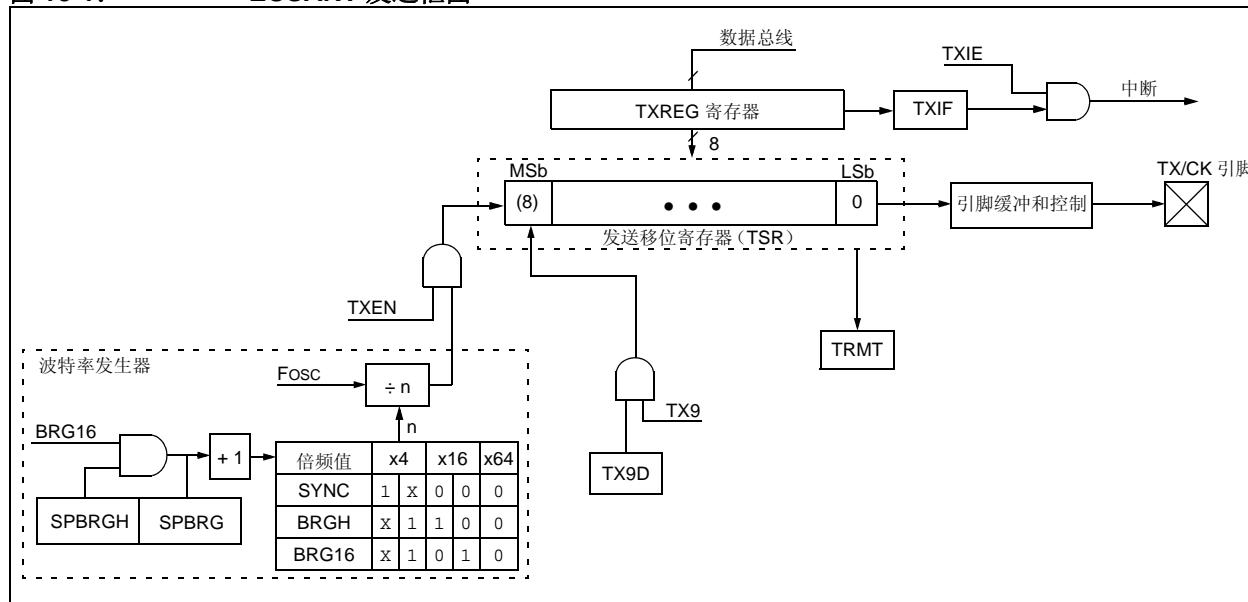
- 全双工异步收发
- 双字符输入缓冲区
- 单字符输出缓冲区
- 可编程 8 位或 9 位字符长度
- 9 位模式下的地址检测
- 输入缓冲区溢出错误检测
- 接收字符帧错误检测
- 半双工同步主模式
- 半双工同步从模式
- 可编程的时钟和数据极性

EUSART 模块还具备以下特性，使其成为局域互联网 (Local Interconnect Network, LIN) 总线系统的理想选择：

- 自动检测和波特率校准
- 接收到间隔字符 (Break) 时唤醒
- 13 位间隔字符发送

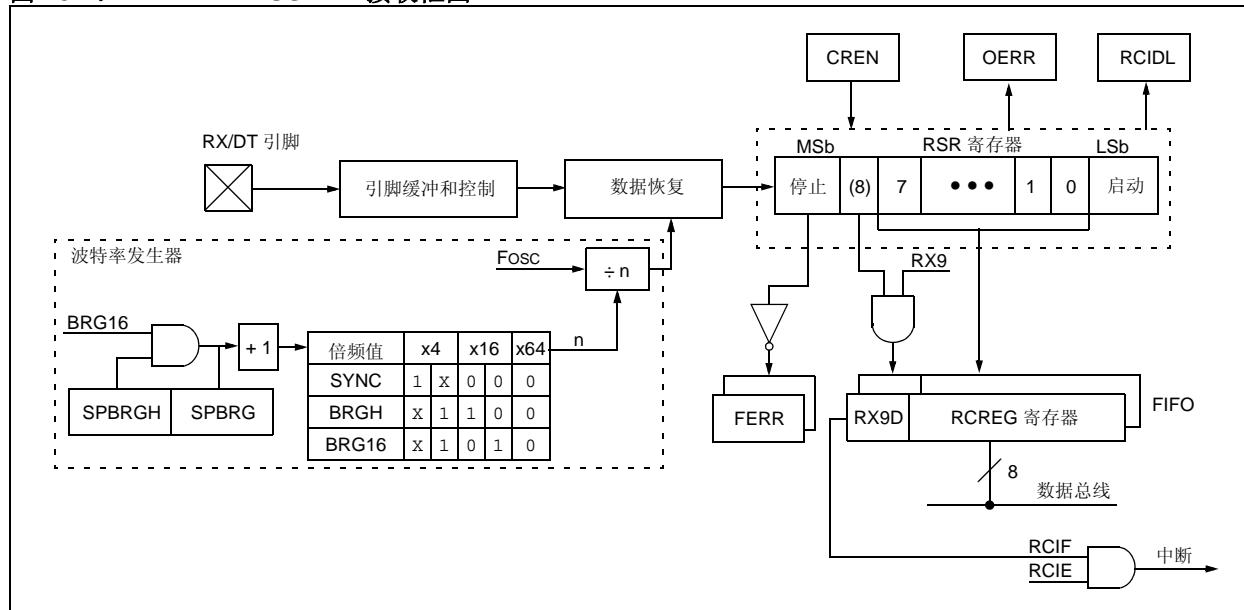
EUSART 收发器的框图如图 18-1 和图 18-2 所示。

图 18-1：EUSART 发送框图



PIC18F2XK20/4XK20

图 18-2：EUSART 接收框图



EUSART 模块的操作由以下 3 个寄存器控制：

- 发送状态和控制（TXSTA）
- 接收状态和控制（RCSTA）
- 波特率控制（BAUDCON）

这些寄存器的详细信息请分别参见寄存器 18-1、寄存器 18-2 和寄存器 18-3。

对于 EUSART 操作的所有模式，与 RX/DT 和 TX/CK 引脚对应的 TRIS 控制位应被置 1。EUSART 控制将根据需要自动将引脚从输入重新配置为输出。

当未使能接收器或发送器部分时，对应的 RX 或 TX 引脚可用于通用输入和输出。

18.1 EUSART 异步模式

EUSART 采用标准不归零 (non-return-to-zero, NRZ) 格式发送和接收数据。NRZ 实现为两种电平: VOH 标记状态 (mark state) 代表 “1” 数据位, 而 VOL 空格状态 (space state) 代表 “0” 数据位。NRZ 指的是当具有相同值的连续数据位被发送时, 它们保持在该位的输出电平不变, 而不会在每个位之间回到中间电平。NRZ 发送端口在标记状态空闲。每个字符发送包含 1 个启动位及随后的 8 个或 9 个数据位, 并始终由 1 个或多个停止位终止。启动位始终是一个空格, 停止位始终是标记。最常见的数据格式为 8 位。每个发送位保持 1/(波特率) 个周期。使用片上专用 8 位 /16 位波特率发生器从系统振荡器产生标准波特率频率。波特率配置示例请参见表 18-5。

EUSART 先发送和接收 LSb。EUSART 的发送器和接收器在功能上是相互独立的, 但它们的数据格式和波特率相同。硬件不支持奇偶校验, 但可通过软件实现并作为第 9 个数据位存储。

18.1.1 EUSART 异步发送器

图 18-1 给出了 EUSART 发送器框图。发送器的核心是串行发送移位寄存器 (Transmit Shift Register, TSR), 该寄存器不可用软件直接访问。TSR 从发送缓冲区 (即 TXREG 寄存器) 取得数据。

18.1.1.1 使能发送器

EUSART 发送器可通过配置以下 3 个控制位使能为异步操作:

- TXEN = 1
- SYNC = 0
- SPEN = 1

假定所有其他 EUSART 控制位均处于其默认状态。

将 TXSTA 寄存器的 TXEN 位置 1 使能 EUSART 的发送器电路。清零 TXSTA 寄存器的 SYNC 位将 EUSART 配置为异步操作。将 RCSTA 寄存器的 SPEN 位置 1 使能 EUSART 并自动将 TX/CK I/O 引脚配置为输出。如果 TX/CK 引脚与模拟外设共用, 那么必须通过清零对应的 ANSEL 位禁止模拟 I/O 功能。

注: TXEN 中断允许位置 1 时, TXIF 发送器中断标志位置 1。

18.1.1.2 发送数据

向 TXREG 寄存器写入一个字符时启动发送。如果这是首字符, 或前一个字符被完全从 TSR 中送出, TXREG 中的数据就立即被传送到 TSR 寄存器。如果 TSR 中仍保存前一个字符的全部或部分, 则新字符被保存在 TXREG 中, 直到前一个字符的停止位被发送。之后, 在 TXREG 中等待的字符在停止位发送后 1 个 TCY 内被传送到 TSR 中。TXREG 中的数据被传送到 TSR 后, 启动位、数据位和停止位的发送序列立即开始。

18.1.1.3 发送数据极性

可通过 BAUDCON 寄存器的 CKTYP 位来控制发送数据的极性。该位的默认状态为 0, 选择高电平有效发送空闲和数据位。将 CKTYP 位设为 1 将发送数据的极性取反, 从而选择低电平有效空闲和数据位。CKTYP 位仅在异步模式下控制发送数据的极性。在同步模式下, CKTYP 位有不同的功能。

18.1.1.4 发送中断标志

只要 EUSART 发送器被使能, 而且 TXREG 中没有等待发送的字符, PIR1 寄存器的 TXIF 中断标志位就被置 1。换句话说, 只有在 TSR 中有字符, 并且 TXREG 中还有一个排队等待发送的新字符时, TXIF 位才被清零。写入 TXREG 后并不立即清零 TXIF 标志位, 而是在之后的第一个指令周期将其清零。写入 TXREG 后立即查询 TXIF 位将返回无效结果。TXIF 位是只读的, 不能用软件置 1 或清零。

将 PIE1 寄存器的 TXIE 中断允许位置 1 可允许 TXIF 中断。但是, 只要 TXREG 为空, TXIF 标志位就会被置 1, 无论 TXIE 中断允许位的状态如何。

要在发送数据时使用中断, 应只在仍有数据要发送时才将 TXIE 位置 1。在将发送的最后一个字符写入 TXREG 后应清零 TXIE 中断允许位。

18.1.1.5 TSR 状态

TXSTA 寄存器的 TRMT 位指示 TSR 寄存器的状态。该位是只读位。TSR 寄存器为空时，TRMT 位置 1，而当一个字符从 TXREG 传送到 TSR 寄存器中时，该位清零。TRMT 位将保持清零，直到所有位移出 TSR 寄存器。该位不与任何中断逻辑关联，因此用户需要查询该位以确定 TSR 的状态。

注： TSR 寄存器不映射到数据存储器中，因此用户无法使用。

18.1.1.6 发送 9 位字符

EUSART 支持 9 位字符发送。当 TXSTA 寄存器的 TX9 位置 1 时，EUSART 将在发送每个字符时移出 9 位。TXSTA 寄存器的 TX9D 位是第 9 个数据位，也是最高有效位。发送 9 位数据时，TX9D 数据位必须先于低 8 位写入 TXREG。写入 TXREG 后，所有 9 个位将被立即传送到 TSR 移位寄存器中。

有多个接收器时，可使用一种特殊的 9 位地址模式。关于地址模式的更多信息，请参见第 18.1.2.8 节“地址检测”。

18.1.1.7 异步发送设置

1. 初始化 SPBRGH:SPBRG 寄存器对以及 BRGH 和 BRG16 位，获得所需的波特率（见第 18.3 节“EUSART 波特率发生器（BRG）”）。
2. 将 RX/DT 和 TX/CK 的 TRIS 控制位置 1。
3. 清零 SYNC 位并将 SPEN 位置 1，使能异步串口。
4. 如果需要 9 位发送，将 TX9 控制位置 1。如果第 9 个数据位置 1，则表示发送器置于检测地址时，低 8 位为地址。
5. 如果需要将发送数据的极性取反，将 CKTXP 控制位置 1。
6. 将 TXEN 控制位置 1 使能发送。这将导致 TXIF 中断标志位置 1。
7. 如果需要中断，将 TXIE 中断允许位置 1。如果 INTCON 寄存器的 GIE 和 PEIE 位也置 1，则立即产生中断。
8. 如果选择了 9 位发送，应将第 9 位装入 TX9D 数据位。
9. 将 8 位数据装入 TXREG 寄存器。这将启动发送。

图 18-3： 异步发送

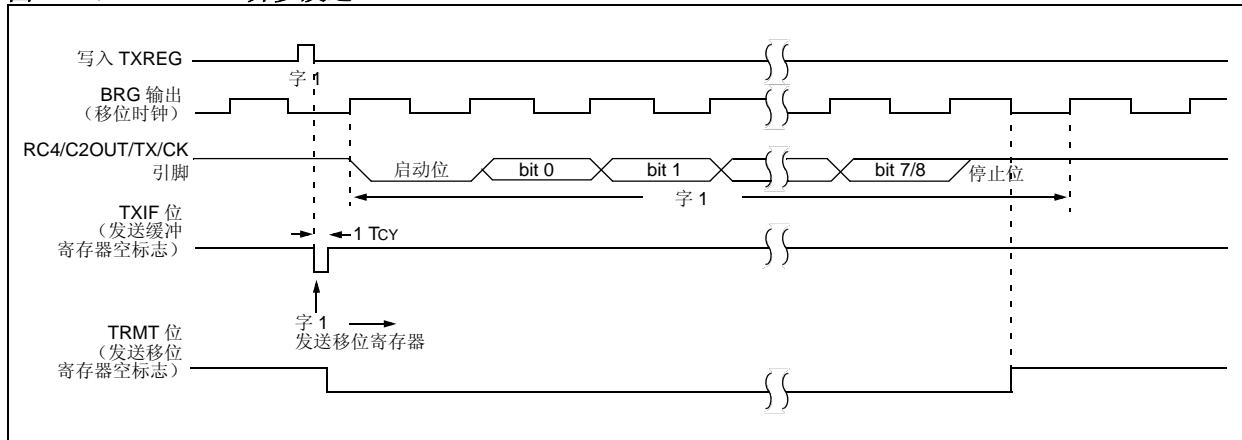


图 18-4: 异步发送 (背对背)

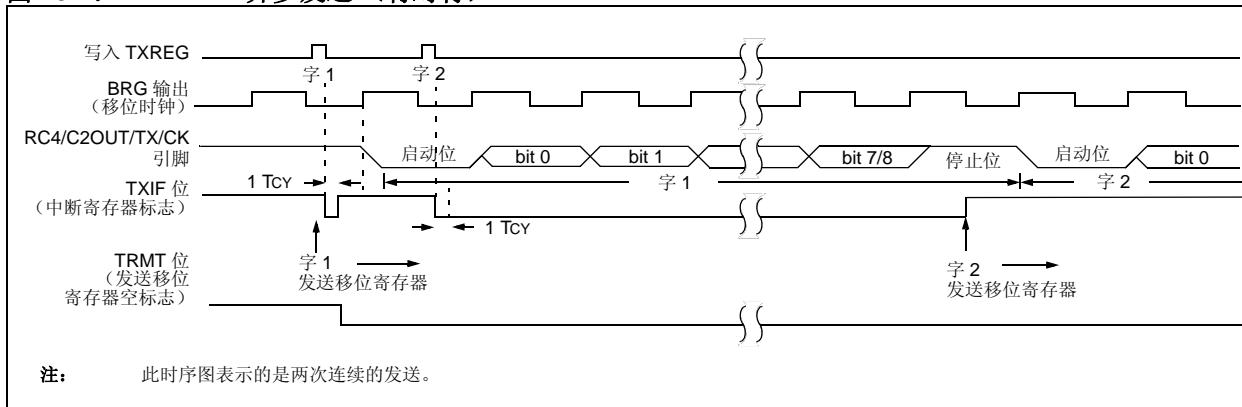


表 18-1: 与异步发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TXREG	EUSART 发送寄存器								61
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON	ABDOVF	RCIDL	DTRXP	CKTXP	BRG16	—	WUE	ABDEN	61
SPBRGH	EUSART 波特率发生器寄存器的高字节								61
SPBRG	EUSART 波特率发生器寄存器的低字节								61

图注: — = 未实现位, 读为 0。异步发送不使用阴影单元。

注 1: 在 PIC18F2XK20 器件中保留; 始终保持这些位清零。

18.1.2 EUSART 异步接收器

异步模式一般用于 RS-232 系统中。图 18-2 给出了接收器框图。数据在 RX/DT 引脚上接收并驱动数据恢复模块。数据恢复模块实际上是一个高速移位器，工作频率为 16 倍波特率，而串行接收移位寄存器（Receive Shift Register, RSR）工作频率为比特率。所有 8 位或 9 位字符移入后被立即传送到双字符的先进先出（First-In-First-Out, FIFO）存储区中。FIFO 缓冲区允许先接收两个完整字符和第三个字符的开始部分后，再启动软件服务 EUSART 接收器。FIFO 和 RSR 寄存器不能直接用软件访问。通过 RCREG 寄存器访问接收数据。

18.1.2.1 使能接收器

EUSART 接收器可通过配置以下 3 个控制位使能为异步操作：

- CREN = 1
- SYNC = 0
- SPEN = 1

假定所有其他 EUSART 控制位均处于其默认状态。

将 RCSTA 寄存器的 CREN 位置 1 使能 EUSART 的接收器电路。清零 TXSTA 寄存器的 SYNC 位将 EUSART 配置为异步操作。将 RCSTA 寄存器的 SPEN 位置 1 可使能 EUSART。必须通过将对应的 TRIS 控制位置 1 将 RX/DT I/O 引脚配置为输入。如果 RX/DT 引脚与模拟外设共用，那么必须通过清零对应的 ANSEL 位禁止模拟 I/O 功能。

18.1.2.2 接收数据

接收器的数据恢复电路在第一位的下降沿启动字符接收。第一位也称启动（Start）位，始终为零。数据恢复电路计数传输半个位的时间至启动位的中点并验证该位是否仍为零。如果该位非零则数据恢复电路中止字符接收，不产生错误，并恢复寻找启动位的下降沿。如果启动位被验证为零，则数据恢复电路计数一个位时间至下个位的中点。该位被一个择多检测电路采样，其结果（0 或 1）被移入 RSR。重复此过程直到所有数据位均被采样并移入 RSR。最后一个位时间被测量且其电平被采样。此为停止（Stop）位，始终为 1。如果数据恢复电路在停止位处采样到 0，则置 1 此字符的帧错误标志位，否则清零此字符的帧错误标志位。关于帧错误的更多信息，请参见第 18.1.2.5 节“接收帧错误”。

所有数据位和停止位被接收后，RSR 中的字符就被立即传送到 EUSART 接收 FIFO，且 PIR1 寄存器的 RCIF 中断标志位被置 1。读取 RCREG 寄存器时，FIFO 中顶部的字符被送出 FIFO。

注： 如果接收 FIFO 溢出，在溢出条件被清除前不会接收更多字符。关于溢出错误的更多信息，请参见第 18.1.2.6 节“接收溢出错误”。

18.1.2.3 接收数据极性

可通过 BAUDCON 寄存器的 DTRXP 位来控制接收数据的极性。该位的默认状态为 0，选择高电平有效接收空闲和数据位。将 DTRXP 位设为 1 将接收数据的极性取反，从而选择低电平有效空闲和数据位。DTRXP 位仅在异步模式下控制接收数据的极性。在同步模式下，DTRXP 位有不同的功能。

18.1.2.4 接收中断

只要 EUSART 接收器被使能且接收 FIFO 中存在未被读取的字符，PIR1 寄存器的 RCIF 中断标志位就会被置 1。RCIF 中断标志位是只读位，不能用软件置 1 或清零。

将以下位置 1 可允许 RCIF 中断：

- PIE1 寄存器的 RCIE 中断允许位
- INTCON 寄存器的 PEIE 外设中断允许位
- INTCON 寄存器的 GIE 全局中断允许位

当 FIFO 中存在未被读取的字符时，无论中断允许位的状态如何，RCIF 中断标志位均会被置 1。

18.1.2.5 接收帧错误

接收 FIFO 缓冲区中的每个字符都有相应的帧错误状态位。帧错误表明在预期时间内未见到停止位。通过 RCSTA 寄存器的 FERR 位可访问帧错误状态。FERR 位表示接收 FIFO 中顶部的未读字符的状态。因此，在读取 RCREG 前必须读出 FERR 位。

FERR 位是只读位，只用于接收 FIFO 中顶部的未读字符。帧错误（FERR = 1）并不会禁止接收更多字符。此时不必将 FERR 位清零。从 FIFO 缓冲区读出下一个字符将使 FIFO 进入下一个字符和下一个相应的帧错误。

将 RCSTA 寄存器的 SPEN 位清零可复位 EUSART，这样就可将 FERR 位强制清零。将 RCSTA 寄存器的 CREN 位清零不影响 FERR 位。自身产生的帧错误不会产生中断。

注：如果接收 FIFO 中的所有接收字符均有帧错误，反复读取 RCREG 不会将 FERR 位清零。

18.1.2.6 接收溢出错误

接收 FIFO 缓冲区可容纳两个字符。在访问 FIFO 前接收到完整的第三个字符时会产生溢出错误。此时，RCSTA 寄存器的 OERR 位置 1。FIFO 缓冲区中已有的字符可被读出，但溢出错误被清除前不能再接收其他字符。将 RCSTA 寄存器的 CREN 位清零或通过将 RCSTA 寄存器的 SPEN 位清零复位 EUSART，可清除该错误。

18.1.2.7 接收 9 位字符

EUSART 支持 9 位字符接收。当 RCSTA 寄存器的 RX9 位置 1 时，EUSART 将在接收每个字符时将 9 个位移入 RSR。RCSTA 寄存器的 RX9D 位是第 9 位，也是接收 FIFO 顶部未读字符的最高有效位。从接收 FIFO 缓冲区读取 9 位数据时，在读取 RCREG 的低 8 位前必须先读取 RX9D 数据位。

18.1.2.8 地址检测

当多个接收器共用同一条发送线时，如在 RS-485 系统中，有一个特殊的地址检测模式可供使用。将 RCSTA 寄存器的 ADDEN 位置 1 可使能地址检测。

地址检测要求接收 9 位字符。使能地址检测时，只有第 9 个数据位置 1 的字符会被传递到接收 FIFO 缓冲区，并将 RCIF 中断标志位置 1。所有其他字符均被忽略。

接收到地址字符后，用户软件可决定地址是否与自身匹配。地址匹配时，发生下一个停止位前，用户软件必须通过清零 ADDEN 位禁止地址检测。当用户软件根据所使用的报文协议检测到报文的末尾时，软件将 ADDEN 位置 1，将接收器重新置于地址检测模式。

18.1.2.9 异步接收设置

1. 初始化 SPBRGH:SPBRG 寄存器对以及 BRGH 和 BRG16 位，获得所需的波特率（见第 18.3 节“**EUSART 波特率发生器（BRG）**”）。
2. 将 RX/DT 和 TX/CK 的 TRIS 控制位置 1。
3. 将 SPEN 位和 RX/DT 引脚的 TRIS 位置 1 使能串口。SYNC 位必须清零才能进行异步操作。
4. 如果需要中断，将 RCIE 中断允许位置 1，并将 INTCON 寄存器的 GIE 和 PEIE 位置 1。
5. 如果需要接收 9 位数据，将 RX9 位置 1。
6. 如果需要将接收极性取反，将 DTRXP 置 1。
7. 将 CREN 位置 1 使能接收。
8. 当字符从 RSR 被移入接收缓冲区时，RCIF 中断标志位将被置 1。如果 RCIE 中断允许位也置 1，则产生中断。
9. 读取 RCSTA 寄存器取得错误标志，以及第 9 个数据位（9 位数据接收使能时）。
10. 读取 RCREG 寄存器从接收缓冲区取得接收数据的低 8 位。
11. 发生溢出时，通过清零 CREN 接收器使能位清零 OERR 标志位。

18.1.2.10 9 位地址检测模式设置

此模式通常用于 RS-485 系统中。设置使能地址检测的异步接收的步骤如下：

1. 初始化 SPBRGH:SPBRG 寄存器对以及 BRGH 和 BRG16 位，获得所需的波特率（见第 18.3 节“**EUSART 波特率发生器（BRG）**”）。
2. 将 RX/DT 和 TX/CK 的 TRIS 控制位置 1。
3. 将 SPEN 位置 1 使能串口。SYNC 位必须清零才能进行异步操作。
4. 如果需要中断，将 RCIE 中断允许位置 1，并将 INTCON 寄存器的 GIE 和 PEIE 位置 1。
5. 将 RX9 位置 1 使能 9 位接收。
6. 将 ADDEN 位置 1 使能地址检测。
7. 如果需要将接收极性取反，将 DTRXP 置 1。
8. 将 CREN 位置 1 使能接收。
9. 当第 9 位置 1 的字符从 RSR 被移入接收缓冲区时，RCIF 中断标志位将被置 1。如果 RCIE 中断允许位也置 1，则产生中断。
10. 读取 RCSTA 寄存器取得错误标志。第 9 个数据位将始终置 1。
11. 读取 RCREG 寄存器从接收缓冲区取得接收数据的低 8 位。软件将决定此地址是否是器件地址。
12. 发生溢出时，通过清零 CREN 接收器使能位清零 OERR 标志位。
13. 如果器件被寻址，将 ADDEN 位清零以允许所有接收到的数据被送入接收缓冲区并产生中断。

图 18-5: 异步接收

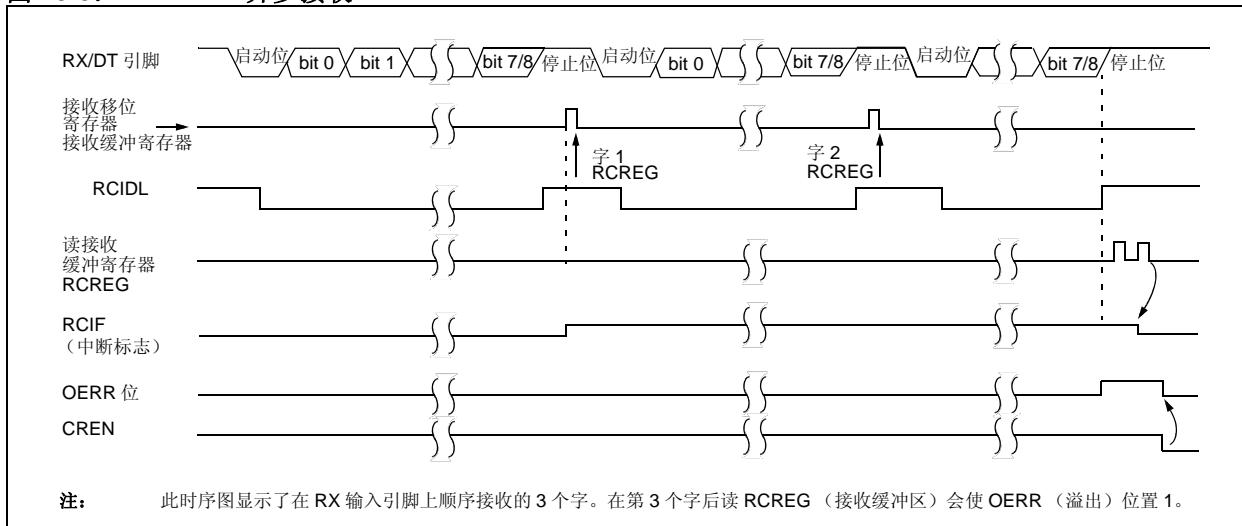


表 18-2: 与异步接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG	EUSART 接收寄存器								61
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	62
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON	ABDOVF	RCIDL	DTRXP	CKTYP	BRG16	—	WUE	ABDEN	61
SPBRGH	EUSART 波特率发生器寄存器的高字节								61
SPBRG	EUSART 波特率发生器寄存器的低字节								61

图注: — = 未实现位, 读为 0。异步接收不使用阴影单元。

注 1: 在 PIC18F2XK20 器件中保留; 始终保持这些位清零。

18.2 异步操作的时钟精确性

内部振荡器模块输出（HFINTOSC）在出厂时做了校准。但是，VDD 或温度变化时 HFINTOSC 频率有可能漂移，这将直接影响异步波特率。有两种方法可用来调整波特率时钟，但它们都需要某种参考时钟源。

第一种（推荐）方法使用 OSCTUNE 寄存器调整 HFINTOSC 输出。调整 OSCTUNE 寄存器的值可对系统时钟源的分辨率进行微调。更多信息，请参见第 2.5 节“内部时钟模式”。

另一种方法调整波特率发生器的值。自动波特率检测可自动完成这种调整（见第 18.3.1 节“自动波特率检测”）。通过调整波特率发生器来补偿外设时钟频率的逐渐变化时，这种方法的分辨率可能不够。

寄存器 18-1： TXSTA：发送状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN ⁽¹⁾	SYNC	SENDDB	BRGH	TRMT	TX9D
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **CSRC：** 时钟源选择位

异步模式：

无关位

同步模式：

1 = 主模式（时钟由内部 BRG 产生）

0 = 从模式（时钟来自外部时钟源）

bit 6 **TX9：** 9 位发送使能位

1 = 选择 9 位发送

0 = 选择 8 位发送

bit 5 **TXEN：** 发送使能位⁽¹⁾

1 = 使能发送

0 = 禁止发送

bit 4 **SYNC：** EUSART 模式选择位

1 = 同步模式

0 = 异步模式

bit 3 **SENDDB：** 发送间隔字符位

异步模式：

1 = 在下一次发送时发送同步间隔字符（完成时由硬件清零）

0 = 同步间隔字符发送完成

同步模式：

无关位

bit 2 **BRGH：** 高波特率选择位

异步模式：

1 = 高速

0 = 低速

同步模式：

在此模式下未使用

bit 1 **TRMT：** 发送移位寄存器状态位

1 = TSR 空

0 = TSR 满

bit 0 **TX9D：** 发送数据的第 9 位

可以是地址 / 数据位或奇偶校验位。

注 1： 在同步模式下，SREN/CREN 可改写 TXEN。

寄存器 18-2: RCSTA: 接收状态和控制寄存器 (1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **SPEN:** 串口使能位
 1 = 使能串口 (将 RX/DT 和 TX/CK 引脚配置为串口引脚)
 0 = 禁止串口 (保持在复位状态)

bit 6 **RX9:** 9 位接收使能位
 1 = 选择 9 位接收
 0 = 选择 8 位接收

bit 5 **SREN:** 单字节接收使能位
异步模式:
 无关位
同步主模式:
 1 = 使能单字节接收
 0 = 禁止单字节接收
 此位在接收完成后清零。
同步从模式:
 无关位

bit 4 **CREN:** 连续接收使能位
异步模式:
 1 = 使能接收器
 0 = 禁止接收器
同步模式:
 1 = 使能连续接收, 直到使能位 CREN 清零 (CREN 的优先级高于 SREN)
 0 = 禁止连续接收

bit 3 **ADDEN:** 地址检测使能位
9 位异步模式 (RX9 = 1):
 1 = 当 RSR<8> 置 1 时, 使能地址检测, 允许中断并装入接收缓冲区
 0 = 禁止地址检测, 接收所有字节并且第 9 位可作为奇偶校验位
8 位异步模式 (RX9 = 0):
 无关位

bit 2 **FERR:** 帧错误位
 1 = 帧错误 (可以通过读 RCREG 寄存器更新该位并接收下一个有效字节)
 0 = 无帧错误

bit 1 **OERR:** 溢出错误位
 1 = 溢出错误 (可以通过清零 CREN 位来清零该位)
 0 = 无溢出错误

bit 0 **RX9D:** 接收数据的第 9 位
 该位可以是地址 / 数据位或奇偶校验位, 并且必须由用户固件计算得到。

PIC18F2XK20/4XK20

寄存器 18-3: BAUDCON: 波特率控制寄存器

R/W-0	R-1	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	DTRXP	CKTXP	BRG16	—	WUE	ABDEN
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **ABDOVF:** 自动波特率检测溢出位

异步模式:

1 = 自动波特率定时器溢出

0 = 自动波特率定时器未溢出

同步模式:

无关位

bit 6 **RCIDL:** 接收空闲标志位

异步模式:

1 = 接收器空闲

0 = 已检测到启动位且接收器处于活动状态

同步模式:

无关位

bit 5 **DTRXP:** 数据 / 接收极性选择位

异步模式:

1 = 接收数据 (RX) 反相 (低电平有效)

0 = 接收数据 (RX) 未反相 (高电平有效)

同步模式:

1 = 数据 (DT) 反相 (低电平有效)

0 = 数据 (DT) 未反相 (高电平有效)

bit 4 **CKTXP:** 时钟 / 发送极性选择位

异步模式:

1 = 发送 (TX) 的空闲状态为低电平

0 = 发送 (TX) 的空闲状态为高电平

同步模式:

1 = 在时钟的下降沿改变数据, 在时钟的上升沿采样数据

0 = 在时钟的上升沿改变数据, 在时钟的下降沿采样数据

bit 3 **BRG16:** 16 位波特率发生器位

1 = 使用 16 位波特率发生器 (SPBRGH:SPBRG)

0 = 使用 8 位波特率发生器 (SPBRG)

bit 2 **未实现:** 读为 0

WUE: 唤醒使能位

异步模式:

1 = 接收器正在等待下降沿。不会接收到任何字符, 但 RCIF 在下降沿将被置 1。WUE 在上升沿将自动清零。

0 = 接收器正常工作

同步模式:

无关位

bit 0 **ABDEN:** 自动波特率检测使能位

异步模式:

1 = 使能自动波特率模式 (完成自动波特率后清零)

0 = 禁止自动波特率模式

同步模式:

无关位

18.3 EUSART 波特率发生器 (BRG)

波特率发生器 (BRG) 是 8 位或 16 位定时器，专用于支持异步和同步 EUSART 操作。默认情况下，BRG 工作在 8 位模式下。将 BAUDCON 寄存器的 BRG16 位置 1 可选择 16 位模式。

SPBRGH:SPBRG 寄存器对决定自由运行波特率定时器的周期。在异步模式下，波特率周期的倍数由 TXSTA 寄存器的 BRGH 位和 BAUDCON 寄存器的 BRG16 位决定。在同步模式下，BRGH 位被忽略。

表 18-3 提供了确定波特率的公式。例 18-1 提供了确定波特率和波特率误差的计算示例。

为便于您使用，各种异步模式的典型波特率和误差值已经计算出来，如表 18-5 所示。使用高波特率 (BRGH = 1) 或 16 位 BRG (BRG16 = 1) 有助于降低波特率误差。16 位 BRG 模式用于在高速振荡器频率下取得较缓慢的波特率。

将新值写入 SPBRGH:SPBRG 寄存器对将导致 BRG 定时器复位（或清零）。这可以确保 BRG 无需等待定时器溢出就可以输出新的波特率。

如果系统时钟在主动接收操作过程中发生变化，可能会导致接收错误或数据丢失。为避免此问题，应检查 RCIDL 位的状态，以确保在改变系统时钟前接收操作处于空闲状态。

例 18-1:

计算波特率误差

器件工作在 $F_{OSC} = 16 \text{ MHz}$, 目标波特率 = 9600, 异步模式, 8 位 BRG:

$$\text{目标波特率} = \frac{F_{OSC}}{64(\text{SPBRGH:SPBRG}) + 1}$$

求解 SPBRGH:SPBRG:

$$\begin{aligned} X &= \frac{\frac{F_{OSC}}{\text{目标波特率}}}{64} - 1 \\ &= \frac{\frac{16000000}{9600}}{64} - 1 \\ &= [25.042] = 25 \end{aligned}$$

$$\begin{aligned} \text{计算波特率} &= \frac{16000000}{64(25 + 1)} \\ &= 9615 \end{aligned}$$

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

表 18-3: 波特率公式

配置位			BRG/EUSART 模式	波特率公式
SYNC	BRG16	BRGH		
0	0	0	8 位 / 异步	$F_{osc}/[64(n+1)]$
0	0	1	8 位 / 异步	$F_{osc}/[16(n+1)]$
0	1	0	16 位 / 异步	
0	1	1	16 位 / 异步	$F_{osc}/[4(n+1)]$
1	0	x	8 位 / 同步	
1	1	x	16 位 / 同步	

图注: x = 无关位, n = SPBRGH:SPBRG 寄存器对的值

表 18-4: 与波特率发生器相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
BAUDCON	ABDOVF	RCIDL	DTRXP	CKTXP	BRG16	—	WUE	ABDEN	61
SPBRGH	EUSART 波特率发生器寄存器的高字节								61
SPBRG	EUSART 波特率发生器寄存器的低字节								61

图注: — = 未实现, 读为 0。BRG 不使用阴影单元。

PIC18F2XK20/4XK20

表 18-5: 异步模式下的波特率

波特率	SYNC = 0, BRGH = 0, BRG16 = 0											
	Fosc = 64.000 MHz			Fosc = 18.432 MHz			Fosc = 16.000 MHz			Fosc = 11.0592 MHz		
	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)
300	—	—	—	—	—	—	—	—	—	—	—	—
1200	—	—	—	1200	0.00	239	1202	0.16	207	1200	0.00	143
2400	—	—	—	2400	0.00	119	2404	0.16	103	2400	0.00	71
9600	9615	0.16	103	9600	0.00	29	9615	0.16	25	9600	0.00	17
10417	10417	0.00	95	10286	-1.26	27	10417	0.00	23	10165	-2.42	16
19.2k	19.23k	0.16	51	19.20k	0.00	14	19.23k	0.16	12	19.20k	0.00	8
57.6k	58.82k	2.12	16	57.60k	0.00	7	—	—	—	57.60k	0.00	2
115.2k	111.11k	-3.55	8	—	—	—	—	—	—	—	—	—

波特率	SYNC = 0, BRGH = 0, BRG16 = 0											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 1.000 MHz		
	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)
300	—	—	—	300	0.16	207	300	0.00	191	300	0.16	51
1200	1202	0.16	103	1202	0.16	51	1200	0.00	47	1202	0.16	12
2400	2404	0.16	51	2404	0.16	25	2400	0.00	23	—	—	—
9600	9615	0.16	12	—	—	—	9600	0.00	5	—	—	—
10417	10417	0.00	11	10417	0.00	5	—	—	—	—	—	—
19.2k	—	—	—	—	—	—	19.20k	0.00	2	—	—	—
57.6k	—	—	—	—	—	—	57.60k	0.00	0	—	—	—
115.2k	—	—	—	—	—	—	—	—	—	—	—	—

波特率	SYNC = 0, BRGH = 1, BRG16 = 0											
	Fosc = 64.000 MHz			Fosc = 18.432 MHz			Fosc = 16.000 MHz			Fosc = 11.0592 MHz		
	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)
300	—	—	—	—	—	—	—	—	—	—	—	—
1200	—	—	—	—	—	—	—	—	—	—	—	—
2400	—	—	—	—	—	—	—	—	—	—	—	—
9600	—	—	—	9600	0.00	119	9615	0.16	103	9600	0.00	71
10417	—	—	—	10378	-0.37	110	10417	0.00	95	10473	0.53	65
19.2k	19.23k	0.16	207	19.20k	0.00	59	19.23k	0.16	51	19.20k	0.00	35
57.6k	57.97k	0.64	68	57.60k	0.00	19	58.82k	2.12	16	57.60k	0.00	11
115.2k	114.29k	-0.79	34	115.2k	0.00	9	111.1k	-3.55	8	115.2k	0.00	5

表 18-5：异步模式下的波特率（续）

波特率	SYNC = 0, BRGH = 1, BRG16 = 0											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 1.000 MHz		
	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)	实际 波特率	% 误差	SPBRG 值 (十进制)
300	—	—	—	—	—	—	—	—	—	300	0.16	207
1200	—	—	—	1202	0.16	207	1200	0.00	191	1202	0.16	51
2400	2404	0.16	207	2404	0.16	103	2400	0.00	95	2404	0.16	25
9600	9615	0.16	51	9615	0.16	25	9600	0.00	23	—	—	—
10417	10417	0.00	47	10417	0.00	23	10473	0.53	21	10417	0.00	5
19.2k	19231	0.16	25	19.23k	0.16	12	19.2k	0.00	11	—	—	—
57.6k	55556	-3.55	8	—	—	—	57.60k	0.00	3	—	—	—
115.2k	—	—	—	—	—	—	115.2k	0.00	1	—	—	—

波特率	SYNC = 0, BRGH = 0, BRG16 = 1											
	Fosc = 64.000 MHz			Fosc = 18.432 MHz			Fosc = 16.000 MHz			Fosc = 11.0592 MHz		
	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)
300	300.0	0.00	13332	300.0	0.00	3839	300.03	0.01	3332	300.0	0.00	2303
1200	1200.1	0.01	3332	1200	0.00	959	1200.5	0.04	832	1200	0.00	575
2400	2399	-0.02	1666	2400	0.00	479	2398	-0.08	416	2400	0.00	287
9600	9592	-0.08	416	9600	0.00	119	9615	0.16	103	9600	0.00	71
10417	10417	0.00	383	10378	-0.37	110	10417	0.00	95	10473	0.53	65
19.2k	19.23k	0.16	207	19.20k	0.00	59	19.23k	0.16	51	19.20k	0.00	35
57.6k	57.97k	0.64	68	57.60k	0.00	19	58.82k	2.12	16	57.60k	0.00	11
115.2k	114.29k	-0.79	34	115.2k	0.00	9	111k	-3.55	8	115.2k	0.00	5

波特率	SYNC = 0, BRGH = 0, BRG16 = 1											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 1.000 MHz		
	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)
300	299.9	-0.02	1666	300.1	0.04	832	300.0	0.00	767	300.5	0.16	207
1200	1199	-0.08	416	1202	0.16	207	1200	0.00	191	1202	0.16	51
2400	2404	0.16	207	2404	0.16	103	2400	0.00	95	2404	0.16	25
9600	9615	0.16	51	9615	0.16	25	9600	0.00	23	—	—	—
10417	10417	0.00	47	10417	0.00	23	10473	0.53	21	10417	0.00	5
19.2k	19.23k	0.16	25	19.23k	0.16	12	19.20k	0.00	11	—	—	—
57.6k	55556	-3.55	8	—	—	—	57.60k	0.00	3	—	—	—
115.2k	—	—	—	—	—	—	115.2k	0.00	1	—	—	—

PIC18F2XK20/4XK20

表 18-5： 异步模式下的波特率（续）

波特率	SYNC = 0, BRGH = 1, BRG16 = 1 或 SYNC = 1, BRG16 = 1											
	Fosc = 64.000 MHz			Fosc = 18.432 MHz			Fosc = 16.000 MHz			Fosc = 11.0592 MHz		
	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)
300	300	0.00	53332	300.0	0.00	15359	300.0	0.00	13332	300.0	0.00	9215
1200	1200	0.00	13332	1200	0.00	3839	1200.1	0.01	3332	1200	0.00	2303
2400	2400	0.00	6666	2400	0.00	1919	2399.5	-0.02	1666	2400	0.00	1151
9600	9598.1	-0.02	1666	9600	0.00	479	9592	-0.08	416	9600	0.00	287
10417	10417	0.00	1535	10425	0.08	441	10417	0.00	383	10433	0.16	264
19.2k	19.21k	0.04	832	19.20k	0.00	239	19.23k	0.16	207	19.20k	0.00	143
57.6k	57.55k	-0.08	277	57.60k	0.00	79	57.97k	0.64	68	57.60k	0.00	47
115.2k	115.11k	-0.08	138	115.2k	0.00	39	114.29k	-0.79	34	115.2k	0.00	23

波特率	SYNC = 0, BRGH = 1, BRG16 = 1 或 SYNC = 1, BRG16 = 1											
	Fosc = 8.000 MHz			Fosc = 4.000 MHz			Fosc = 3.6864 MHz			Fosc = 1.000 MHz		
	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)	实际 波特率	% 误差	SPBRGH :SPBRG (十进制)
300	300.0	0.00	6666	300.0	0.01	3332	300.0	0.00	3071	300.1	0.04	832
1200	1200	-0.02	1666	1200	0.04	832	1200	0.00	767	1202	0.16	207
2400	2401	0.04	832	2398	0.08	416	2400	0.00	383	2404	0.16	103
9600	9615	0.16	207	9615	0.16	103	9600	0.00	95	9615	0.16	25
10417	10417	0.00	191	10417	0.00	95	10473	0.53	87	10417	0.00	23
19.2k	19.23k	0.16	103	19.23k	0.16	51	19.20k	0.00	47	19.23k	0.16	12
57.6k	57.14k	-0.79	34	58.82k	2.12	16	57.60k	0.00	15	—	—	—
115.2k	117.6k	2.12	16	111.1k	-3.55	8	115.2k	0.00	7	—	—	—

18.3.1 自动波特率检测

EUSART 模块支持波特率自动检测和校准。

在自动波特率检测（Auto-Baud Rate Detect, ABD）模式下，BRG 的时钟信号反向。BRG 并不为进入的 RX 信号提供时钟信号，而是由 RX 信号为 BRG 定时。波特率发生器用于为接收的 55h (ASCII “U”) 定时，这是 LIN 总线的同步字符。此字符的特殊之处在于它具有包括停止位边沿在内的 5 个上升沿。

将 BAUDCON 寄存器的 ABDEN 位置 1 将启动自动波特率校验序列（图 18-6）。当发生 ABD 序列时，EUSART 状态机保持在空闲状态。在接收线的第一个上升沿（启动位之后），SPBRG 使用 BRG 计数器时钟递增计数，如表 18-6 所示。在第 8 位周期的末尾将在 RX 引脚上出现第 5 个上升沿。此时，累计数据即正确的 BRG 周期总数被留在 SPBRGH:SPBRG 寄存器对中，ABDEN 位被自动清零而 RCIF 中断标志被置 1。要清除 RCIF 中断，需要执行对 RCREG 的读操作。RCREG 的内容应该被丢弃。校准不使用 SPBRGH 寄存器的模式时，用户可通过查询 SPBRGH 寄存器的值是否为 00h 来验证 SPBRG 寄存器是否溢出。

BRG 自动波特率时钟由 BRG16 和 BRGH 位决定，如表 18-6 所示。在 ABD 期间，SPBRGH 和 SPBRG 寄存器共同用作 16 位计数器，这与 BRG16 位的设置

无关。在校准波特率周期时，SPBRGH 和 SPBRG 寄存器的时钟频率为 BRG 基时钟频率的 1/8。得到的字节测量结果为全速下的平均位时间。

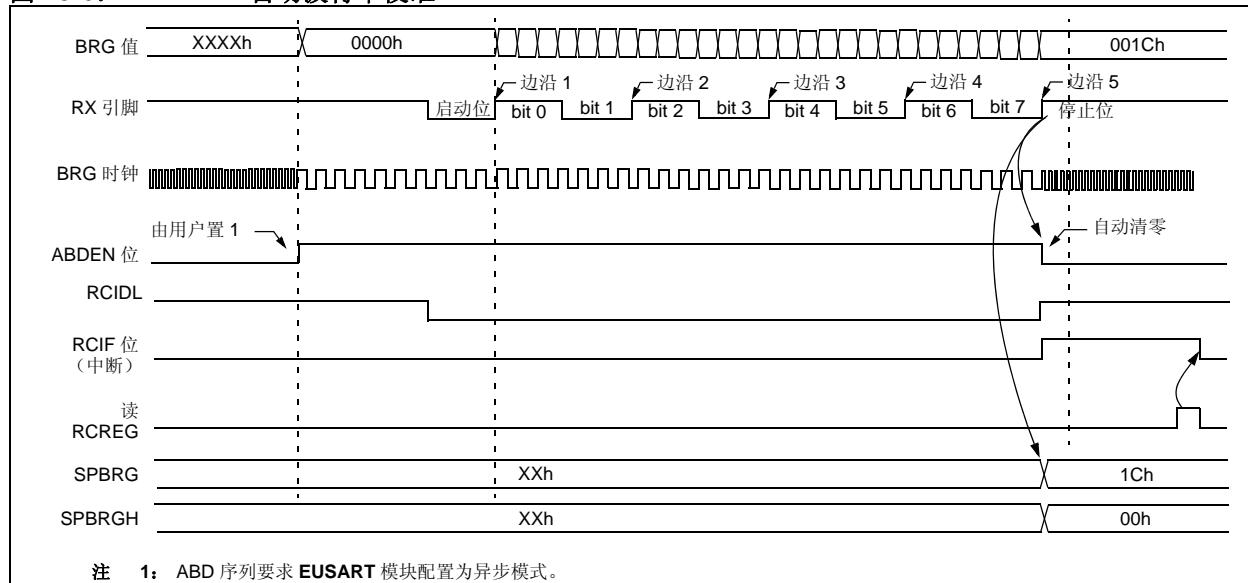
- 注**
- 1:** 如果 WUE 位和 ABDEN 位都置 1，自动波特率检测将发生在间隔字符之后的字节开始（见第 18.3.3 节“接收到间隔字符时自动唤醒”）。
 - 2:** 需要由用户来判断输入字符的波特率是否处于所选 BRG 时钟源范围内。某些振荡器频率和 EUSART 波特率组合不可能实现。
 - 3:** 在自动波特率过程中，自动波特率计数器从 1 开始计数。自动波特率序列完成后，为了得到最准确的结果，应从 SPBRGH:SPBRG 寄存器对的值中减去 1。

表 18-6： BRG 计数器时钟速率

BRG16	BRGH	BRG 基时钟	BRG ABD 时钟
0	0	Fosc/64	Fosc/512
0	1	Fosc/16	Fosc/128
1	0	Fosc/16	Fosc/128
1	1	Fosc/4	Fosc/32

注： 在 ABD 序列期间，SPBRG 和 SPBRGH 寄存器同时用作 16 位计数器，与 BRG16 的设置无关。

图 18-6： 自动波特率校准



18.3.2 自动波特率溢出

在自动波特率检测过程中，如果在 RX 引脚上检测到第 5 个上升沿之前波特率计数器溢出，则 BAUDCON 寄存器的 ABDOVF 位将被置 1。ABDOVF 位指示计数器已超出适合 SPBRGH:SPBRG 寄存器对的 16 位的最大计数。在 ABDOVF 置 1 后，计数器将继续计数，直到在 RX 引脚上检测到第 5 个上升沿为止。一旦检测到第 5 个 RX 边沿，硬件将把 RCIF 中断标志置 1，并将 BAUDCON 寄存器的 ABDEN 位清零。随后可以通过读取 RCREG 将 RCIF 标志清零。ABDOVF 标志可直接用软件清零。

若要在 RCIF 标志置 1 前终止自动波特率过程，请先将 ABDEN 位清零，然后将 ABDOVF 位清零。如果没有先将 ABDEN 位清零，ABDOVF 位将保持置 1 状态。

18.3.3 接收到间隔字符时自动唤醒

在休眠模式下，EUSART 的所有时钟都会暂停。因此，波特率发生器处于不工作状态，不能正常进行字符接收。自动唤醒功能使控制器可被 RX/DT 线上的活动唤醒。该功能只在异步模式下可用。

自动唤醒功能可通过将 BAUDCON 寄存器的 WUE 位置 1 来使能。一旦置 1，RX/DT 上的正常接收序列就被禁止，EUSART 保持在空闲状态，监视与 CPU 模式无关的唤醒事件。唤醒事件包含 RX/DT 线上电平由高至低的跳变。（这与同步间隔字符的开始或 LIN 协议的唤醒信号字符一致。）

EUSART 模块产生的 RCIF 中断与唤醒事件同步。在正常 CPU 工作模式下，中断产生与 Q 时钟同步（图 18-7），而器件处于休眠模式时则异步发生（图 18-8）。通过读 RCREG 寄存器可清除中断条件。

在间隔字符末尾 RX 线由低至高的跳变将自动清零 WUE 位。这向用户表明间隔事件结束。此时，EUSART 模块处于空闲模式，等待接收下一个字符。

18.3.3.1 特殊注意事项

间隔字符

在发生唤醒事件期间为了避免字符错误或字符碎片，唤醒字符必须为全零。

唤醒被使能时，其工作与数据流的低电平时间无关。如果 WUE 位置 1 并接收到了有效的非零字符，则从启动位至第一个上升沿的低电平时间将被解读为唤醒事件。字符的其余位将作为碎片字符接收，后续字符有可能产生帧错误或溢出错误。

因此，发送的首字符必须为全 0。这必须持续 10 个或更长的位时间，对于 LIN 总线建议持续 13 个位时间，而标准 RS-232 器件可为任意个位时间。

振荡器起振时间

必须考虑振荡器起振时间，特别在使用起振时间较长的振荡器（即，LP、XT 或 HS/PLL 模式）的应用中。同步间隔（或唤醒信号）字符必须足够长，并随后有一个足够长的间隔时间，以使所选的振荡器有足够的时间起振并在这段时间对 EUSART 进行适当初始化。

WUE 位

唤醒事件会通过将 RCIF 位置 1 产生一个接收中断。WUE 位在 RX/DT 的上升沿由硬件清零。之后软件通过读取 RCREG 寄存器并丢弃其内容将中断条件清除。

要确保不丢失实际数据，应在将 WUE 位置 1 前检查 RCIDL 位，验证没有接收操作在进行。如果未发生接收操作，可在进入休眠模式前将 WUE 位置 1。

图 18-7：正常操作时的自动唤醒位（WUE）时序

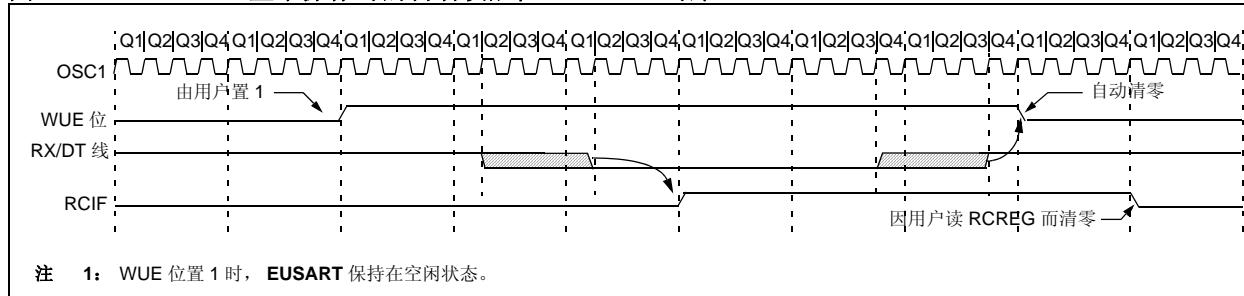
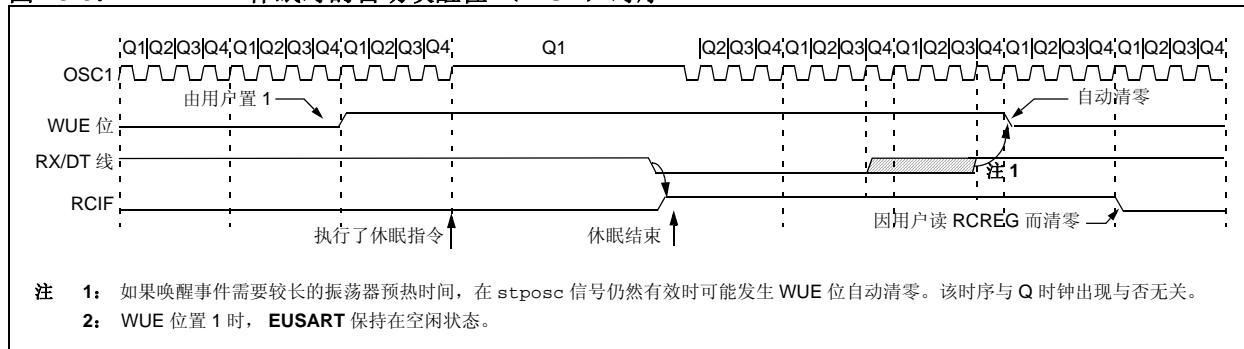


图 18-8：休眠时的自动唤醒位（WUE）时序



18.3.4 间隔字符序列

EUSART模块能够发送符合LIN总线标准的特殊间隔字符序列。间隔字符包含1个启动位，随后的12个0位和1个停止位。

要发送间隔字符，应将TXSTA寄存器的SENDB和TXEN位置1。随后对TXREG执行写操作可启动间隔字符发送。写入TXREG的数据值会被忽略并发送全0。

在发送了相应的停止位后，硬件会自动将SENDB位复位。这样用户可以在间隔字符（在LIN规范中通常是同步字符）后预先将下一个要发送字节装入发送FIFO。

TXSTA寄存器的TRMT位表明发送操作何时处于激活或空闲状态，这与正常发送时相同。图18-9给出了发送间隔字符的时序。

18.3.4.1 间隔和同步发送序列

以下序列将启动报文帧头，它由间隔字符和其后的自动波特率同步字节组成。这是LIN总线主器件的典型序列。

1. 将EUSART配置为所需的模式。
2. 将TXEN和SENDB位置1使能间隔序列。
3. 将无效字符装入TXREG，启动发送（该值会被忽略）。
4. 将“55h”写入TXREG，以便将同步字符装入发送FIFO缓冲区。
5. 发送间隔字符后，SENDB位被硬件复位，同步字符随后被发送。

当TXREG为空时（由TXIF指出），下一个数据字节会写入TXREG。

18.3.5 接收间隔字符

增强型EUSART模块接收间隔字符有两种方法。

第一种检测间隔字符的方法采用RCSTA寄存器的FERR位和如RCREG所指示的接收数据。假定波特率发生器已初始化为所需的波特率。

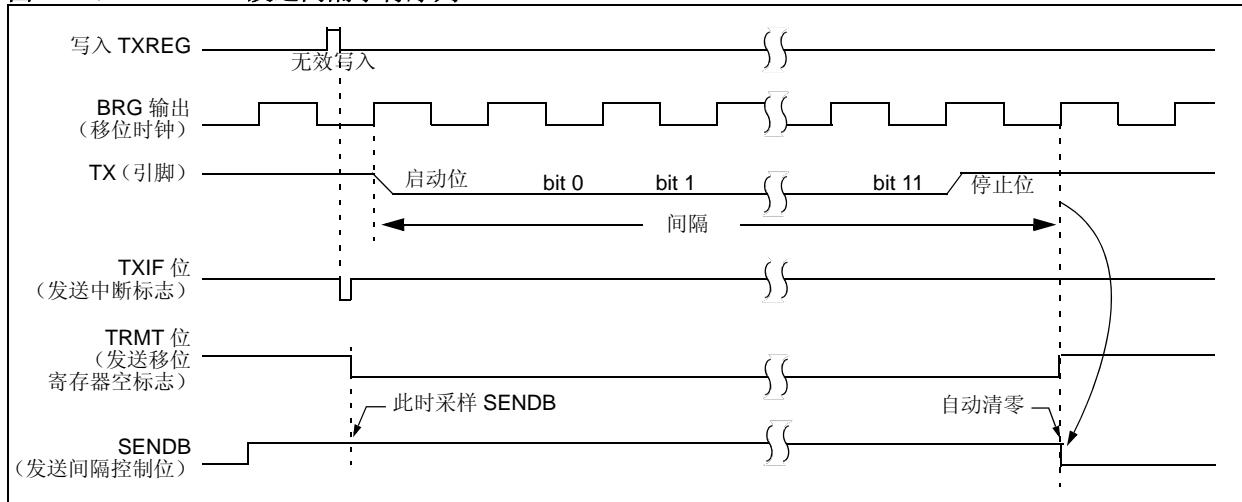
以下成立时，表明接收到间隔字符：

- RCIF位被置1
- FERR位被置1
- RCREG = 00h

第二种方法采用第18.3.3节“接收到间隔字符时自动唤醒”中所述的自动唤醒功能。通过使能此功能，EUSART将采样RX/DT上的下两次跳变，产生RCIF中断，并接收下一个数据字节并再产生一次中断。

请注意在间隔字符后，用户通常希望使能自动波特率检测功能。采用这两种方法时，用户均可在EUSART进入休眠模式前将BAUDCON寄存器的ABDEN位置1。

图 18-9：发送间隔字符序列



18.4 EUSART 同步模式

同步串行通信通常用于具有一个主器件和一个或多个从器件的系统中。主器件包含生成波特率所需的电路，可将时钟提供给系统中的所有器件。从器件使用主时钟，可不再需要内部时钟生成电路。

同步模式下有两条信号线：双向数据线和时钟线。从器件使用主器件提供的外部时钟将串行数据移入或移出相应的接收和发送移位寄存器。由于数据线是双向的，同步操作只能是半双工的。半双工指从器件能够接收和发送数据，但不能同时进行。EUSART 可作为主器件，也可作为从器件。

同步发送时不使用启动位和停止位。

18.4.1 同步主模式

使用以下位将 EUSART 配置为同步主操作：

- SYNC = 1
- CSRC = 1
- SREN = 0 (用于发送) ; SREN = 1 (用于接收)
- CREN = 0 (用于发送) ; CREN = 1 (用于接收)
- SPEN = 1

将 TXSTA 寄存器的 SYNC 位置 1 将器件配置为同步操作。将 TXSTA 寄存器的 CSRC 位置 1 可将器件配置为主器件。将 RCSTA 寄存器的 SREN 和 CREN 位清零可确保器件处于发送模式，否则器件将被配置为接收。将 RCSTA 寄存器的 SPEN 位置 1 可使能 EUSART。如果 RX/DT 或 TX/CK 引脚与模拟外设共用，那么必须通过清零对应的 ANSEL 位禁止模拟 I/O 功能。

与 RX/DT 和 TX/CK 引脚对应的 TRIS 位应置 1。

18.4.1.1 主时钟

同步数据传送使用独立的时钟线，时钟与数据同步。配置为主器件的器件将时钟信号发送到 TX/CK 线上。EUSART 配置为同步发送或接收操作时，自动使能 TX/CK 引脚输出驱动器。串行数据位在时钟前沿改变，以确保其在时钟的后续边沿有效。为每个数据位产生一个时钟周期。数据位有多少，就产生多少个时钟周期。

18.4.1.2 时钟极性

提供了时钟极性选项以与 Microwire 兼容。时钟极性通过 BAUDCON 寄存器的 CKTYP 位选择。将 CKTYP 位置 1 将时钟空闲状态设置为高电平。当 CKTYP 位置 1 时，在每个时钟的下降沿改变数据，在每个时钟的上升沿采样数据。将 CKTYP 位清零将时钟空闲状态设置为低电平。当 CKTYP 位清零时，在每个时钟的上升沿改变数据，在每个时钟的下降沿采样数据。

18.4.1.3 同步主发送

从器件的 RX/DT 引脚输出数据。EUSART 配置为同步主发送操作时，RX/DT 和 TX/CK 引脚的输出驱动器被自动使能。

向 TXREG 寄存器写入一个字符时启动发送。如果 TSR 中仍保存前一个字符的全部或部分，则新字符被保存在 TXREG 中，直到前一个字符的最后一一位被发送。如果这是首字符，或前一个字符被完全从 TSR 中送出，TXREG 中的数据就立即被传送到 TSR。字符发送在数据从 TXREG 送入 TSR 后立即开始。

每个数据位在主时钟的时钟前沿改变，并在下一个时钟前沿到来前保持有效。

注： TSR 寄存器不映射到数据存储器中，因此用户无法使用。

18.4.1.4 数据极性

可通过 BAUDCON 寄存器的 DTRXP 位来控制发送和接收数据的极性。该位的默认状态为 0，选择高电平有效发送和接收数据。将 DTRXP 位置 1 将数据极性取反，从而选择低电平有效发送和接收数据。

PIC18F2XK20/4XK20

18.4.1.5 同步主发送设置

1. 初始化 SPBRGH:SPBRG 寄存器对以及 BRGH 和 BRG16 位，获得所需的波特率（见第 18.3 节“**EUSART 波特率发生器 (BRG)**”）。
2. 将 RX/DT 和 TX/CK 的 TRIS 控制位置 1。
3. 将 SYNC、SPEN 和 CSRC 位置 1 使能同步主串口。将与 RX/DT 和 TX/CK I/O 引脚对应的 TRIS 位置 1。
4. 将 SREN 和 CREN 位清零禁止接收模式。
5. 将 TXEN 位置 1 使能发送模式。
6. 如果需要 9 位发送，将 TX9 位置 1。
7. 如果需要中断，将 TXIE、GIE 和 PEIE 中断允许位置 1。
8. 如果选择了 9 位发送，应将第 9 位装入 TX9D 位。
9. 将数据装入 TXREG 寄存器，启动发送。

图 18-10： 同步发送

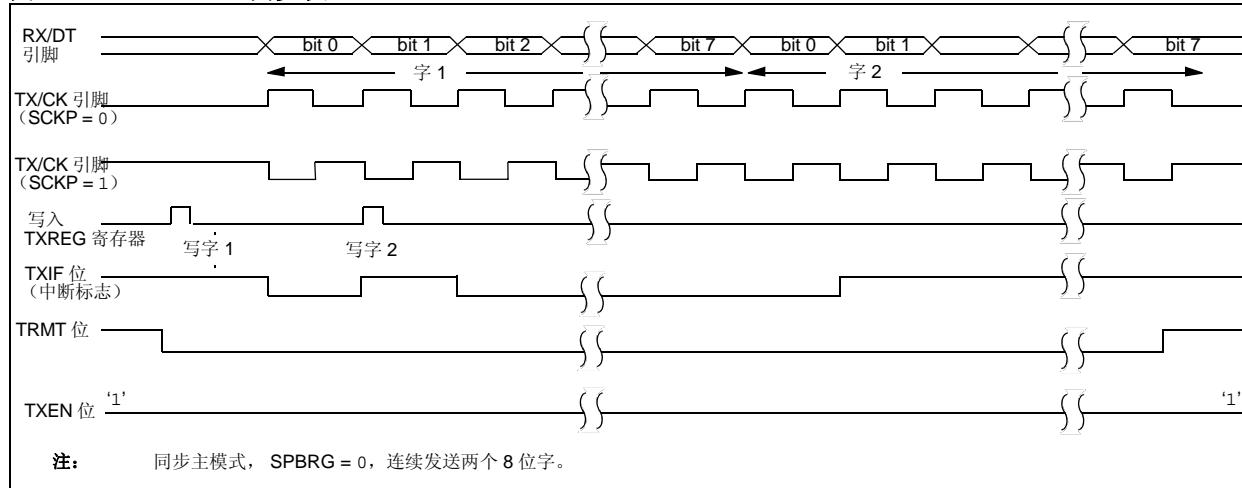


图 18-11： 同步发送（由 TXEN 位控制）

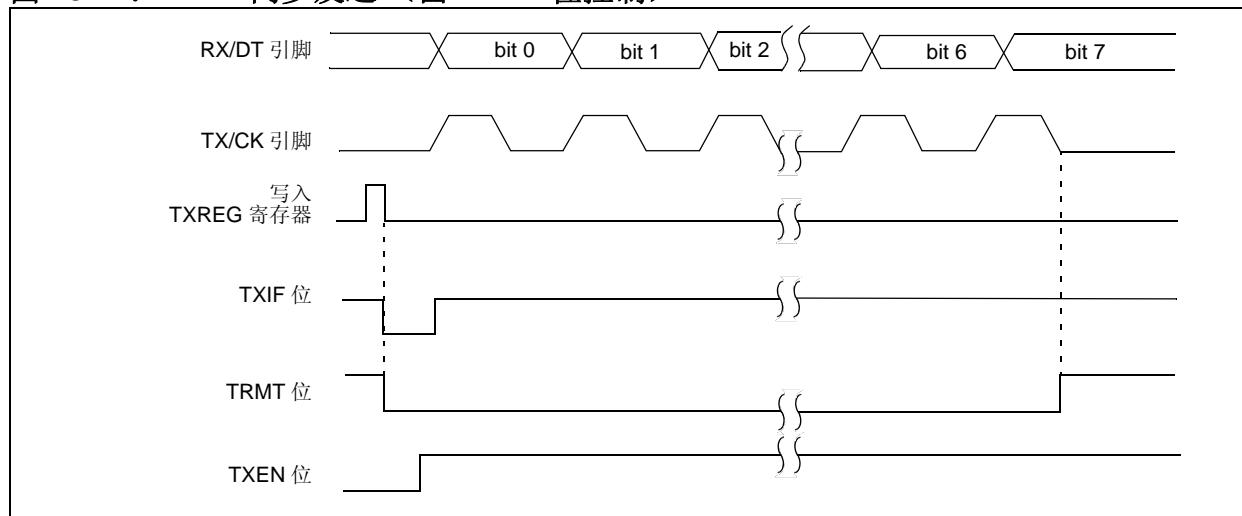


表 18-7：与同步主发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMROIE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	62
TXREG	EUSART 发送寄存器								61
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON	ABDOVF	RCIDL	DTRXP	CKTXP	BRG16	—	WUE	ABDEN	61
SPBRGH	EUSART 波特率发生器寄存器的高字节								61
SPBRG	EUSART 波特率发生器寄存器的低字节								61

图注：— = 未实现，读为 0。同步主发送不使用阴影单元。

注 1：在 PIC18F2XK20 器件中保留；始终保持这些位清零。

18.4.1.6 同步主接收

数据在 RX/DT 引脚上接收。将 EUSART 配置为同步主接收操作时，必须通过将对应的 TRIS 位置 1 来禁止 RX/DT 引脚输出驱动器。

在同步模式下，可通过将单字节接收使能位（RCSTA 寄存器的 SREN）或连续接收使能位（RCSTA 寄存器的 CREN）置 1 使能接收。

SREN 置 1 且 CREN 清零时，一个字符中有多少数据位就产生多少个时钟周期。一个字符接收完成后 SREN 位被自动清零。CREN 置 1 时，将连续产生时钟直到 CREN 被清零。如果 CREN 在字符接收过程中被清零，则 CK 时钟立即停止，接收到的部分字符被丢弃。如果 SREN 和 CREN 同时置 1，则首字符接收完成时 SREN 被清零，CREN 优先。

要启动接收，将 SREN 或 CREN 置 1。在 TX/CK 时钟引脚的后续边沿对 RX/DT 引脚上的数据进行采样，并移入接收移位寄存器（RSR）。当完整的字符被接收进 RSR 后，RCIF 位置 1 且该字符被自动送入两个字符的接收 FIFO。接收 FIFO 中顶部字符的低 8 位在 RCREG 中。只要接收 FIFO 中有未读字符，RCIF 位就保持置 1。

18.4.1.7 从时钟

同步数据传送使用独立的时钟线，时钟与数据同步。配置为从器件的器件在 TX/CK 线上接收时钟信号。将器件配置为同步从发送或接收操作时，必须通过将相关的 TRIS 位置 1 来禁止 TX/CK 引脚输出驱动器。串行数据位在时钟前沿改变，以确保其在时钟的后续边沿有效。每个时钟周期传送一个数据位。数据位有多少，就产生多少个接收时钟周期。

18.4.1.8 接收溢出错误

接收 FIFO 缓冲区可容纳两个字符。在 RCREG 被读取以访问 FIFO 前，接收到完整的第三个字符时会产生溢出错误。此时，RCSTA 寄存器的 OERR 位置 1。FIFO 中的前一个数据不会被覆盖。FIFO 缓冲区中的两个字符可被读出，但错误被清除前不能再接收其他字符。只有清除了溢出条件才可将 OERR 位清零。如果 SREN 位置 1 且 CREN 清零时发生溢出错误，则读取 RCREG 可清除错误。如果 CREN 位置 1 时发生溢出，则可通过清零 RCSTA 寄存器的 CREN 位或清零可将 EUSART 复位的 SPEN 位清除错误条件。

18.4.1.9 接收 9 位字符

EUSART 支持 9 位字符接收。当 RCSTA 寄存器的 RX9 位置 1 时，EUSART 将在接收每个字符时将 9 个位移入 RSR。RCSTA 寄存器的 RX9D 位是第 9 位，也是接收 FIFO 顶部未读字符的最高有效位。从接收 FIFO 缓冲区读取 9 位数据时，在读取 RCREG 的低 8 位前必须先读取 RX9D 数据位。

18.4.1.10 同步主接收设置

1. 初始化 SPBRGH:SPBRG 寄存器对，获得所需的波特率。按需要将 BRGH 和 BRG16 位置 1 或清零，获得所需的波特率。
2. 将 RX/DT 和 TX/CK 的 TRIS 控制位置 1。
3. 将 SYNC、SPEN 和 CSRC 位置 1 使能同步主串口。将对应的 TRIS 位置 1 禁止 RX/DT 和 TX/CK 输出驱动器。
4. 确保将 CREN 和 SREN 位清零。
5. 如果使用中断，将 INTCON 寄存器的 GIE 和 PEIE 位置 1 并将 RCIE 置 1。
6. 如果需要接收 9 位数据，将 RX9 位置 1。
7. 将 SREN 位置 1 启动接收，或将 CREN 位置 1 使能连续接收。
8. 字符接收完成时中断标志位 RCIF 将被置 1。如果中断允许位 RCIE 已置 1，则产生中断。
9. 读取 RCSTA 寄存器取得第 9 位（如果已使能），并确定接收时是否发生了错误。
10. 通过读取 RCREG 寄存器来读取接收到的 8 位数据。
11. 如果发生了溢出错误，可通过清零 RCSTA 寄存器的 CREN 位或清零可将 EUSART 复位的 SPEN 位清除错误。

图 18-12: 同步接收（主模式， SREN）

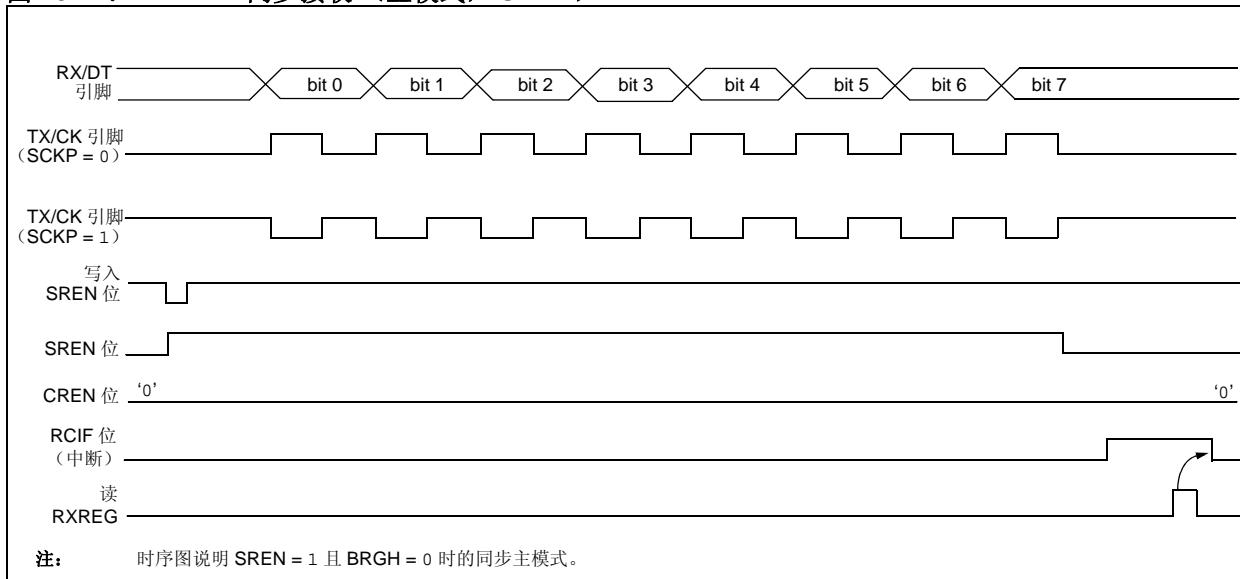


表 18-8: 与同步主接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG	EUSART 接收寄存器								61
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON	ABDOVF	RCIDL	DTRXP	CKTYP	BRG16	—	WUE	ABDEN	61
SPBRGH	EUSART 波特率发生器寄存器的高字节								61
SPBRG	EUSART 波特率发生器寄存器的低字节								61

图注：— = 未实现，读为 0。同步主接收不使用阴影单元。

注 1：在 28 引脚器件中保留；始终保持这些位清零。

18.4.2 同步从模式

使用以下位将 EUSART 配置为同步从操作：

- SYNC = 1
- CSRC = 0
- SREN = 0 (用于发送) ; SREN = 1 (用于接收)
- CREN = 0 (用于发送) ; CREN = 1 (用于接收)
- SPEN = 1

将 TXSTA 寄存器的 SYNC 位置 1 将器件配置为同步操作。将 TXSTA 寄存器的 CSRC 位清零将器件配置为从器件。将 RCSTA 寄存器的 SREN 和 CREN 位清零可确保器件处于发送模式，否则器件将被配置为接收。将 RCSTA 寄存器的 SPEN 位置 1 可使能 EUSART。如果 RX/DT 或 TX/CK 引脚与模拟外设共用，那么必须通过清零对应的 ANSEL 位禁止模拟 I/O 功能。

必须通过将对应的 TRIS 位置 1 来禁止 RX/DT 和 TX/CK 输出驱动器。

18.4.2.1 EUSART 同步从发送

除了休眠模式以外，同步主模式和从模式的工作原理是相同的（见第 18.4.1.3 节“同步主发送”）。

如果向 TXREG 写入两个字，然后执行 SLEEP 指令，则会发生以下事件：

1. 第一个字符将立即传送到 TSR 寄存器并发送。
2. 第二个字将保留在 TXREG 寄存器中。
3. TXIF 位不会被置 1。
4. 第一个字符移出 TSR 后，TXREG 寄存器会将第二个字符传送到 TSR，此时 TXIF 位将置 1。
5. 如果 PEIE 和 TXIE 位均置 1，则发生中断将器件从休眠唤醒，并执行下一条指令。如果 GIE 位也置 1，程序将调用中断服务程序。

18.4.2.2 同步从发送设置

1. 将 SYNC 和 SPEN 位置 1 并清零 CSRC 位。
2. 将 RX/DT 和 TX/CK 的 TRIS 控制位置 1。
3. 清零 CREN 和 SREN 位。
4. 如果使用中断，应确保 INTCON 寄存器的 GIE 和 PEIE 位置 1 并将 TXIE 位置 1。
5. 如果需要 9 位发送，将 TX9 位置 1。
6. 将 TXEN 位置 1 使能发送。
7. 如果选择了 9 位发送，将最高有效位写入 TX9D 位。
8. 将低 8 位写入 TXREG 寄存器，启动发送。

表 18-9：与同步从发送相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	62
TXREG	EUSART 发送寄存器								61
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON	ABDOVF	RCIDL	DTRXP	CKTXP	BRG16	—	WUE	ABDEN	61
SPBRGH	EUSART 波特率发生器寄存器的高字节								61
SPBRG	EUSART 波特率发生器寄存器的低字节								61

图注：— = 未实现，读为 0。同步从发送不使用阴影单元。

注 1：在 PIC18F2XK20 器件中保留；始终保持这些位清零。

18.4.2.3 EUSART 同步从接收

除下列各项外，同步主模式和从模式的工作原理是相同的（第 18.4.1.6 节“同步主接收”）：

- 休眠
- CREN 位始终置 1，因此接收器从不空闲
- SREN 位在从模式下为“无关位”

进入休眠前将 CREN 位置 1，可在休眠模式下接收一个字符。接收到该字后，RSR 寄存器将把数据发送到 RCREG 寄存器。如果 RCIE 中断允许位置 1，产生的中断会将器件从休眠唤醒并执行下一条指令。如果 GIE 位也置 1，程序将跳转到中断向量。

18.4.2.4 同步从接收设置

1. 将 SYNC 和 SPEN 位置 1 并清零 CSRC 位。
2. 将 RX/DT 和 TX/CK 的 TRIS 控制位置 1。
3. 如果使用中断，应确保将 INTCON 寄存器的 GIE 和 PEIE 位置 1 并将 RCIE 位置 1。
4. 如果需要接收 9 位数据，将 RX9 位置 1。
5. 将 CREN 位置 1 使能接收。
6. 接收完成时 RCIF 位将被置 1。如果 RCIE 位已置 1，则产生中断。
7. 如果使能了 9 位模式，从 RCSTA 寄存器的 RX9D 位取出最高有效位。
8. 读取 RCREG 寄存器，从接收 FIFO 取出低 8 位。
9. 如果发生了溢出错误，可通过清零 RCSTA 寄存器的 CREN 位或清零可将 EUSART 复位的 SPEN 位清除错误。

表 18-10：与同步从接收相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	61
RCREG	EUSART 接收寄存器								61
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	61
BAUDCON	ABDOVF	RCIDL	DTRXP	CKTXP	BRG16	—	WUE	ABDEN	61
SPBRGH	EUSART 波特率发生器寄存器的高字节								61
SPBRG	EUSART 波特率发生器寄存器的低字节								61

图注：— = 未实现，读为 0。同步从接收不使用阴影单元。

注 1：在 28 引脚器件中保留；始终保持这些位清零。

PIC18F2XK20/4XK20

注:

19.0 模数转换器（ADC）模块

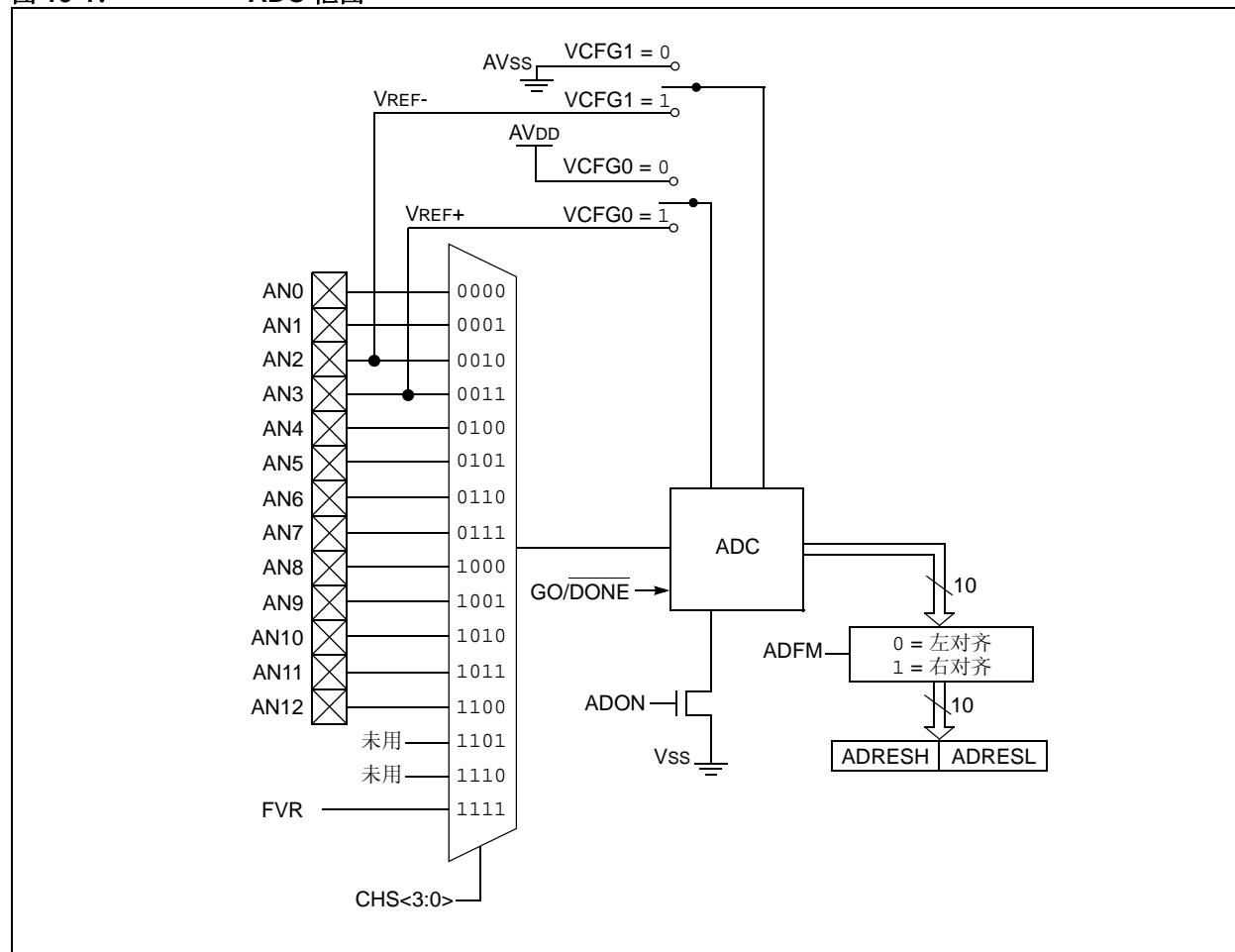
模数转换器（ADC）可将模拟输入信号转换为信号的10位二进制表示。该模块使用模拟输入，这些输入通过多路开关连接到同一个采样和保持电路。采样和保持电路的输出与转换器的输入相连接。转换器通过逐次逼近法产生10位二进制结果，并将转换结果存储在ADC结果寄存器（ADRESL和ADRESH）中。

可通过软件选择VDD或施加在外部参考引脚上的电压作为ADC参考电压。

ADC可在转换完成时产生中断。该中断可用于将器件从休眠状态唤醒。

图19-1给出了ADC的框图。

图 19-1: ADC 框图



19.1 ADC 配置

配置和使用 ADC 时必须考虑以下功能：

- 端口配置
- 通道选择
- ADC 参考电压选择
- ADC 转换时钟源
- 中断控制
- 结果格式

19.1.1 端口配置

ANSEL、**ANSELH**、**TRISA**、**TRISB** 和 **TRISE** 寄存器均可用于配置 A/D 端口引脚。任何需要用到模拟输入的端口引脚都应将其对应的 **ANSx** 位置 1 以禁止数字输入缓冲器，并将 **TRISx** 位置 1 以禁止数字输出驱动器。如果 **TRISx** 位清零，则数字输出电平（**V_{OH}** 或 **V_{OL}**）将被转换。

A/D 转换操作与 **ANSx** 位和 **TRIS** 位的状态无关。

- 注 1:** 当读取端口寄存器时，对应 **ANSx** 位置 1 的所有引脚均读为零（低电平）。但是，对于配置为数字输入的引脚（**ANSx** 位清零，**TRISx** 位置 1），将可以正确进行模拟转换。
- 2:** 对应 **ANSx** 位清零的任何引脚上的模拟电平可能会导致数字输入缓冲器消耗的电流超出器件规范。
- 3:** 通过控制 **ANSELH** 中的位的复位方式，配置寄存器 3H 中的 **PBADEN** 位可将 **PORTB** 引脚配置成复位时为模拟或数字引脚。

19.1.2 通道选择

ADCON0 寄存器的 **CHS** 位决定与采样和保持电路相连接的通道。

改变通道时，在开始下一次转换前需要一段延时。更多信息，请参见第 19.2 节“ADC 工作原理”。

19.1.3 ADC 参考电压

ADCON1 寄存器的 **VCFG** 位提供对正负参考电压的独立控制。正参考电压可以是 **V_{DD}** 或外部电压源。同样，负参考电压可以是 **V_{SS}** 或外部电压源。

19.1.4 选择和配置采集时间

ADCON2 寄存器允许用户选择采集时间，该时间在每当 **GO/DONE** 位置 1 时发生。

采集时间使用 **ADCON2** 寄存器的 **ACQT<2:0>** 位设置。采集延时的范围为 2 至 20 个 **T_{AD}**。当 **GO/DONE** 位被置 1 时，A/D 模块会继续在选定采集时间内采样输入通道，然后自动启动一次转换。由于采集时间已被编程，因此在选择通道和 **GO/DONE** 位置 1 之间无需另外等待一个采集时间。

如果 **ACQT<2:0>** = 000，则表示选择手动采集。当 **GO/DONE** 位被置 1 时，采样停止并启动转换。用户必须确保在选择所需的输入通道和将 **GO/DONE** 位置 1 之间已经过了所需的采集时间。此选项也是 **ACQT<2:0>** 位的默认复位状态，并且与不提供可编程采集时间的器件相兼容。

在这两种情况下，当转换完成时，**GO/DONE** 位均被清零，**ADIF** 标志位均被置 1 并且 A/D 开始再次对当前选定的通道进行采样。当采集时间被编程时，不会有关于何时采集时间结束、转换开始的指示。

19.1.5 转换时钟

可通过软件设置 **ADCON2** 寄存器的 **ADCS** 位来选择转换时钟源。有以下 7 种时钟频率可供选择：

- **Fosc/2**
- **Fosc/4**
- **Fosc/8**
- **Fosc/16**
- **Fosc/32**
- **Fosc/64**
- **FRC**（专用内部振荡器）

完成一个位转换所需的时间定义为 **T_{AD}**。一次完整的 10 位转换需要 11 个 **T_{AD}** 周期，如图 19-3 所示。

为正确转换，必须满足适当的 **T_{AD}** 规范。更多信息，请参见表 26-25 中的 A/D 转换要求。表 19-1 给出了适当的 ADC 时钟选择的示例。

注: 除非使用 **FRC**，否则系统时钟频率的任何改变都会改变 ADC 时钟频率，从而对 ADC 结果产生不利影响。

19.1.6 中断

ADC 模块可在模数转换完成时产生中断。ADC 中断标志位是 PIR1 寄存器中的 ADIF 位。ADC 中断允许位是 PIE1 寄存器中的 ADIE 位。ADIF 位必须用软件清零。

注： ADIF 位在每次转换完成时置 1，与是否允许 ADC 中断无关。

器件工作或休眠时都可产生该中断。如果器件处于休眠状态，该中断会唤醒器件。从休眠状态唤醒时，总是执行紧跟 SLEEP 指令后的下一条指令。如果用户试图从休眠状态唤醒器件并恢复主代码执行，必须禁止全局中断。如果允许了全局中断，执行将切换到中断服务程序。更多信息，请参见第 19.1.6 节“中断”。

表 19-1：ADC 时钟周期 (TAD) 与器件工作频率关系表

ADC 时钟周期 (TAD)		器件频率 (Fosc)			
ADC 时钟源	ADCS<2:0>	64 MHz	16 MHz	4 MHz	1 MHz
Fosc/2	000	31.25 ns ⁽²⁾	125 ns ⁽²⁾	500 ns ⁽²⁾	2.0 μs
Fosc/4	100	62.5 ns ⁽²⁾	250 ns ⁽²⁾	1.0 μs	4.0 μs ⁽³⁾
Fosc/8	001	400 ns ⁽²⁾	500 ns ⁽²⁾	2.0 μs	8.0 μs ⁽³⁾
Fosc/16	101	250 ns ⁽²⁾	1.0 μs	4.0 μs ⁽³⁾	16.0 μs ⁽³⁾
Fosc/32	010	500 ns ⁽²⁾	2.0 μs	8.0 μs ⁽³⁾	32.0 μs ⁽³⁾
Fosc/64	110	1.0 μs	4.0 μs ⁽³⁾	16.0 μs ⁽³⁾	64.0 μs ⁽³⁾
FRC	x11	1-4 μs ^(1,4)	1-4 μs ^(1,4)	1-4 μs ^(1,4)	1-4 μs ^(1,4)

图注： 阴影单元表示超出了建议范围。

注 1： FRC 时钟源具有 1.7 μs 的典型 TAD 时间。

2： 这些值均违反了所需的最小 TAD 时间。

3： 为了加快转换速度，建议选用其他时钟源。

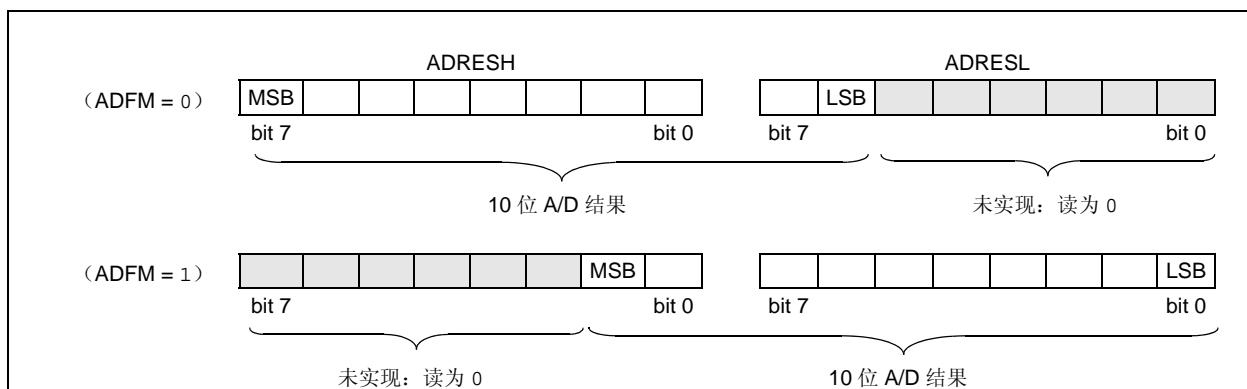
4： 当器件频率高于 1 MHz 时，仅当在休眠状态下进行转换时才推荐使用 FRC 时钟源。

19.1.7 结果格式

10 位 A/D 转换结果可以两种格式提供：左对齐或右对齐。ADCON2 寄存器的 ADFM 位控制输出格式。

图 19-2 给出了两种输出格式。

图 19-2：10 位 A/D 转换结果格式



19.2 ADC 工作原理

19.2.1 启动转换

要使能 ADC 模块, ADCON0 寄存器的 ADON 位必须设为 1。将 ADCON0 寄存器的 GO/DONE 位设为 1, 根据 ADCON2 寄存器的 ACQT 位的状态, 将立即启动模数转换或在模数转换后启动采集延时。

图 19-3 显示了在 GO 位置 1 且 ACQT<2:0> 位被清零后 A/D 转换器的工作状态。转换在下一条指令执行之后开始, 以允许器件在转换开始之前进入休眠模式。

图 19-4 显示了在 GO 位置 1, ACQT<2:0> 位被设置为 010, 且在转换开始之前选择 4 TAD 采集时间后 A/D 转换器的工作状态。

注: 不应在启动 ADC 的同一条指令中将 GO/DONE 位置 1。请参见第 19.2.9 节 “A/D 转换步骤”。

图 19-3: A/D 转换 TAD 周期 (ACQT<2:0> = 000, TACQ = 0)

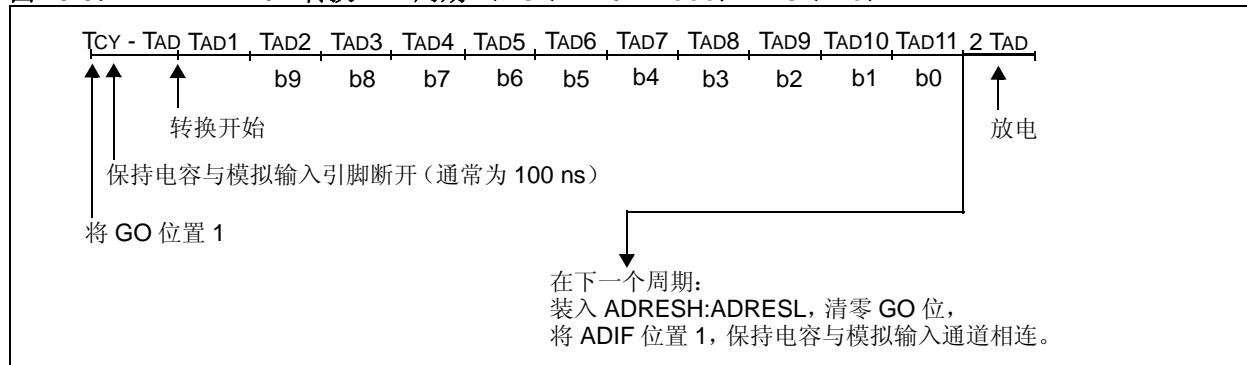
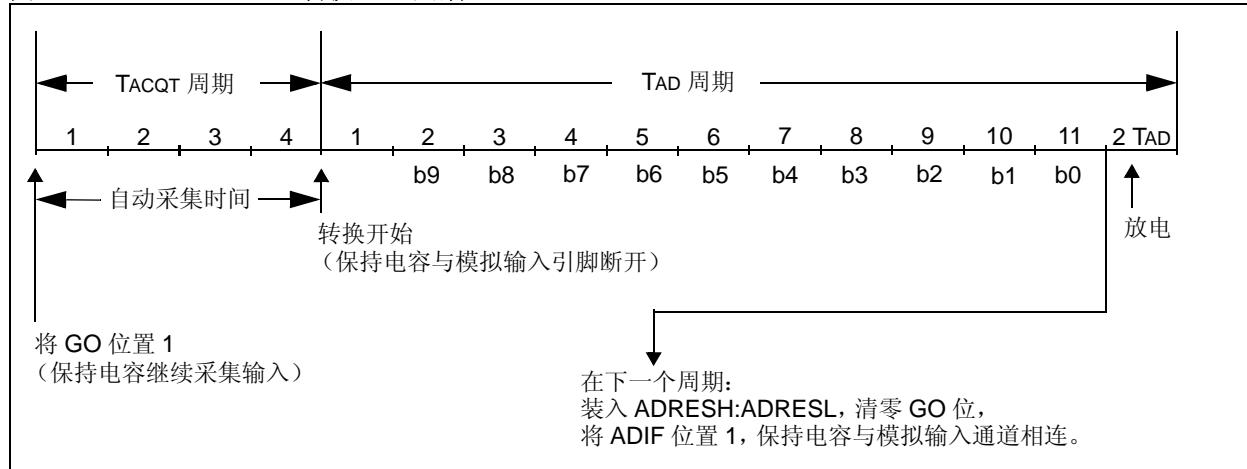


图 19-4: A/D 转换 TAD 周期 (ACQT<2:0> = 010, TACQ = 4 TAD)



19.2.2 转换完成

转换完成时，ADC 模块将：

- 清零 GO/DONE 位
- 将 ADIF 标志位置 1
- 用新的转换结果更新 ADRESH:ADRESL 寄存器

19.2.3 放电

放电过程用于对电容阵列的值进行初始化。在每次采样之后都会对此阵列放电。这一特性有助于优化单位增益放大器，因为每次需要重新为电容阵列充电，而不是根据以前测量的值进行充放电。

19.2.4 终止转换

如果必须在转换完成前终止转换，可用软件将 GO/DONE 位清零。不会用部分完成的模数转换结果更新 ADRESH 和 ADRESL 寄存器，ADRESH:ADRESL 寄存器对仍然保留前一次转换的值。

注： 器件复位将强制所有寄存器为复位状态。
因此，ADC 模块被关闭，任何进行中的转换操作被终止。

19.2.5 转换之间的延时

在 A/D 转换完成或中止后，需要等待 2 个 TAD 才能开始下一次采集。等待该时间之后，当前选定的通道会重新连接到充电保持电容，开始下一次采集。

19.2.6 在功耗管理模式下的 ADC 操作

在功耗管理模式下，自动采集时间和 A/D 转换时钟的选择一定程度上可由时钟源和频率决定。

如果希望器件处于功耗管理模式时进行 A/D 采集转换，就应该根据该模式下使用的时钟对 ADCON2 中的 ACQT<2:0> 和 ADCS<2:0> 位进行更新。在进入功耗管理模式之后，就可以开始 A/D 采集或转换。采集或转换开始以后，器件应继续使用相同的时钟源直到转换完成。

如果需要，器件也可以在转换过程中被置于相应的空闲模式。如果器件时钟频率小于 1 MHz，就应该选择 A/D FRC 时钟源。

19.2.7 休眠期间的 ADC 操作

ADC 模块可以在休眠模式下工作。这需要将 ADC 时钟源设置为 FRC 选项。当选择 FRC 时钟源时，ADC 需等待一个额外的指令周期后才能启动转换。这使得可以执行 SLEEP 指令，以降低转换期间的系统噪声。如果允许了 ADC 中断，转换完成时器件将从休眠状态唤醒。如果禁止了 ADC 中断，尽管 ADON 位仍保持置 1，转换完成后 ADC 模块将关闭。

ADC 时钟源不是 FRC 时，尽管 ADON 位仍保持置 1，SLEEP 指令会导致当前转换中止，ADC 模块关闭。

19.2.8 特殊事件触发器

CCP2 特殊事件触发器允许定期 ADC 测量而无需软件干预。当出现触发信号后，GO/DONE 位由硬件置 1，Timer1 或 Timer3 计数器复位为零。

使用特殊事件触发器不能确保正确的 ADC 时序。用户需负责确保 ADC 时序要求得到满足。

更多信息，请参见第 11.3.4 节 “特殊事件触发器”。

19.2.9 A/D 转换步骤

以下是使用 ADC 执行模数转换的示例步骤：

1. 配置端口：
 - 禁止引脚输出驱动器（见 TRIS 寄存器）
 - 将引脚配置为模拟
2. 配置 ADC 模块：
 - 选择 ADC 转换时钟
 - 配置参考电压
 - 选择 ADC 输入通道
 - 选择结果格式
 - 选择采集延时
 - 开启 ADC 模块
3. 配置 ADC 中断（可选）：
 - 清零 ADC 中断标志
 - 允许 ADC 中断
 - 允许外设中断
 - 允许全局中断⁽¹⁾
4. 等待所需采集时间⁽²⁾。
5. 通过将 GO/DONE 位置 1 启动转换。
6. 通过以下方式之一等待 ADC 转换完成：
 - 查询 GO/DONE 位
 - 等待 ADC 中断（已允许中断）
7. 读取 ADC 结果。
8. 清零 ADC 中断标志（如果已允许中断则需要）。

注 1: 如果用户试图从休眠状态唤醒器件并恢复主代码执行，必须禁止全局中断。
2: 如果 ACQT 位设置为零延时，则需要软件延时。请参见第 19.3 节“**A/D 采集要求**”。

例 19-1：A/D 转换

```
;This code block configures the ADC  
;for polling, Vdd and Vss as reference, Frc  
clock and AN0 input.  
;  
;Conversion start & polling for completion  
; are included.  
;  
MOVlw    B'10101111' ;right justify, Frc,  
MOVwf    ADCON2       ; & 12 TAD ACQ time  
MOVlw    B'00000000' ;ADC ref = Vdd,Vss  
MOVwf    ADCON1       ;  
BSF      TRISA,0      ;Set RA0 to input  
BSF      ANSEL,0      ;Set RA0 to analog  
MOVlw    B'00000001' ;AN0, ADC on  
MOVwf    ADCON0       ;  
BSF      ADCON0,GO    ;Start conversion  
ADCPoll:  
BTFSR   ADCON0,GO    ;Is conversion done?  
BRA     ADCPoll      ;No, test again  
; Result is complete - store 2 MSbits in  
; RESULTHI and 8 LSbits in RESULTLO  
MOVff    ADRESH,RESULTHI  
MOVff    ADRESL,RESULTLO
```

19.2.10 ADC 寄存器定义

以下寄存器用于控制 ADC 的操作。

注: 模拟引脚控制由 ANSEL 和 ANSELH 寄存器执行。关于 ANSEL 和 ANSELH 寄存器的信息，请分别参见寄存器 10-2 和寄存器 10-3。

寄存器 19-1: ADCON0: A/D 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
—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON
bit 7	bit 0						

图注:

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-6 未实现：读为 0

bit 5-2 **CHS<3:0>**: 模拟通道选择位

0000 = AN0

0001 = AN1

0010 = AN2

0011 = AN3

0100 = AN4

0101 = AN5⁽¹⁾

0110 = AN6⁽¹⁾

0111 = AN7⁽¹⁾

1000 = AN8

1001 = AN9

1010 = AN10

1011 = AN11

1100 = AN12

1101 = 保留

1110 = 保留

1111 = FVR (1.2V 固定参考电压) ⁽²⁾

bit 1 **GO/DONE**: A/D 转换状态位

1 = A/D 转换正在进行。将该位置 1 可启动 A/D 转换周期。

A/D 转换完成后，该位由硬件自动清零。

0 = A/D 转换已完成 / 未进行

bit 0 **ADON**: ADC 使能位

1 = 使能 ADC

0 = 禁止 ADC，不消耗工作电流

注 1: 这些通道在 PIC18F2XK20 器件上未实现。

2: 测量固定参考电压时，允许大于 15 μs 的采集时间。

PIC18F2XK20/4XK20

寄存器 19-2: ADCON1: A/D 控制寄存器 1

U-0	U-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	—	VCFG1	VCFG0	—	—	—	—
bit 7	bit 0						

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-6 未实现: 读为 0

bit 5 **VCFG1:** 负参考电压选择位

1 = 负参考电压通过 VREF- 引脚从外部提供。

0 = 负参考电压由 VSS 在内部提供。

bit 4 **VCFG0:** 正参考电压选择位

1 = 正参考电压通过 VREF+ 引脚从外部提供。

0 = 正参考电压由 VDD 在内部提供。

bit 3-0 未实现: 读为 0

寄存器 19-3: ADCON2: A/D 控制寄存器 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7	bit 0						

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **ADFM:** A/D 转换结果格式选择位

1 = 右对齐
0 = 左对齐

bit 6 未实现: 读为 0

bit 5-3 **ACQT<2:0>:** A/D 采集时间选择位。采集时间是从 GO/DONE 位置 1 的时刻到转换开始时, A/D 充电保持电容保持连接到 A/D 通道的持续时间。

000 = 0⁽¹⁾
001 = 2 TAD
010 = 4 TAD
011 = 6 TAD
100 = 8 TAD
101 = 12 TAD
110 = 16 TAD
111 = 20 TAD

bit 2-0 **ADCS<2:0>:** A/D 转换时钟选择位

000 = Fosc/2
001 = Fosc/8
010 = Fosc/32
011 = FRC⁽¹⁾ (由专用内部振荡器产生的时钟, 其频率的标称值为 600 kHz)
100 = FOSC/4
101 = FOSC/16
110 = FOSC/64
111 = FRC⁽¹⁾ (由专用内部振荡器产生的时钟, 其频率的标称值为 600 kHz)

注 1: 当 A/D 时钟源选择 FRC 时, 在 GO/DONE 位置 1 后, 转换开始时间会延迟 1 个指令周期, 以允许执行 SLEEP 指令。

PIC18F2XK20/4XK20

寄存器 19-4: ADRESH: ADC 结果寄存器的高字节 (ADRESH) ADFM = 0

| R/W-x |
|--------|--------|--------|--------|--------|--------|--------|--------|
| ADRES9 | ADRES8 | ADRES7 | ADRES6 | ADRES5 | ADRES4 | ADRES3 | ADRES2 |
| bit 7 | | | | | | | bit 0 |

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-0 **ADRES<9:2>**: ADC 结果寄存器位
10 位转换结果的高 8 位

寄存器 19-5: ADRESL: ADC 结果寄存器的低字节 (ADRESL) ADFM = 0

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
ADRES1	ADRES0	—	—	—	—	—	—
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-6 **ADRES<1:0>**: ADC 结果寄存器位
10 位转换结果的低 2 位

bit 5-0 保留: 不要使用。

寄存器 19-6: ADRESH: ADC 结果寄存器的高字节 (ADRESH) ADFM = 1

| R/W-x |
|-------|-------|-------|-------|-------|-------|-------|--------|
| — | — | — | — | — | — | — | ADRES9 |
| bit 7 | | | | | | | bit 0 |

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-2 保留: 不要使用。

bit 1-0 **ADRES<9:8>**: ADC 结果寄存器位
10 位转换结果的高 2 位

寄存器 19-7: ADRESL: ADC 结果寄存器的低字节 (ADRESL) ADFM = 1

| R/W-x |
|--------|--------|--------|--------|--------|--------|--------|--------|
| ADRES7 | ADRES6 | ADRES5 | ADRES4 | ADRES3 | ADRES2 | ADRES1 | ADRES0 |
| bit 7 | | | | | | | bit 0 |

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-0 **ADRES<7:0>**: ADC 结果寄存器位
10 位转换结果的低 8 位

19.3 A/D 采集要求

为了使 ADC 达到规定的精度，必须使充电保持电容 (CHOLD) 完全充电至输入通道的电压。模拟输入模型见图 19-5。模拟信号源阻抗 (R_s) 和内部采样开关阻抗 (R_{SS}) 直接影响电容 CHOLD 的充电时间。采样开关阻抗 (R_{SS}) 随器件电压 (V_{DD}) 的变化而变化，请参见图 19-5。**模拟信号源的最大阻抗推荐值为 10 k Ω** 。采集时间随着源阻抗的降低而缩短。在选择（或改变）模拟输入通道后，必须在开始转换前完成 A/D 采集。可以使用公式 19-1 来

计算最小采集时间。该公式假设误差为 1/2 LSB (ADC 转换需要 1024 步)。1/2 LSB 误差是 ADC 达到规定分辨率所允许的最大误差。

公式 19-1：采集时间示例

假设：温度 = 50°C，外部阻抗为 10 k Ω ，VDD 为 3.0V

$$\begin{aligned} T_{ACQ} &= \text{放大器稳定时间} + \text{保持电容充电时间} + \text{温度系数} \\ &= T_{AMP} + T_C + T_{COFF} \\ &= 5 \mu\text{s} + T_C + [(\text{温度} - 25^\circ\text{C}) (0.05 \mu\text{s}/^\circ\text{C})] \end{aligned}$$

T_C 值可以用以下公式近似计算：

$$\begin{aligned} V_{APPLIED} \left(1 - \frac{1}{2047} \right) &= V_{CHOLD} && ; [1] \text{ 充电到 } V_{CHOLD} \text{ (1/2 LSB 误差范围)} \\ V_{APPLIED} \left(1 - e^{-\frac{T_C}{RC}} \right) &= V_{CHOLD} && ; [2] \text{ 响应 } V_{APPLIED} \text{ 充电到 } V_{CHOLD} \\ V_{APPLIED} \left(1 - e^{-\frac{T_C}{RC}} \right) &= V_{APPLIED} \left(1 - \frac{1}{2047} \right) && ; \text{结合 [1] 和 [2]} \end{aligned}$$

求解 T_C ：

$$\begin{aligned} T_C &= -C_{HOLD}(R_{IC} + R_{SS} + R_S) \ln(1/2047) \\ &= -13.5 \mu\text{F} (1k\Omega + 7k\Omega + 10k\Omega) \ln(0.0004885) \\ &= 1.20 \mu\text{s} \end{aligned}$$

因此：

$$\begin{aligned} T_{ACQ} &= 5 \mu\text{s} + 1.20 \mu\text{s} + [(50^\circ\text{C} - 25^\circ\text{C}) (0.05 \mu\text{s}/^\circ\text{C})] \\ &= 7.45 \mu\text{s} \end{aligned}$$

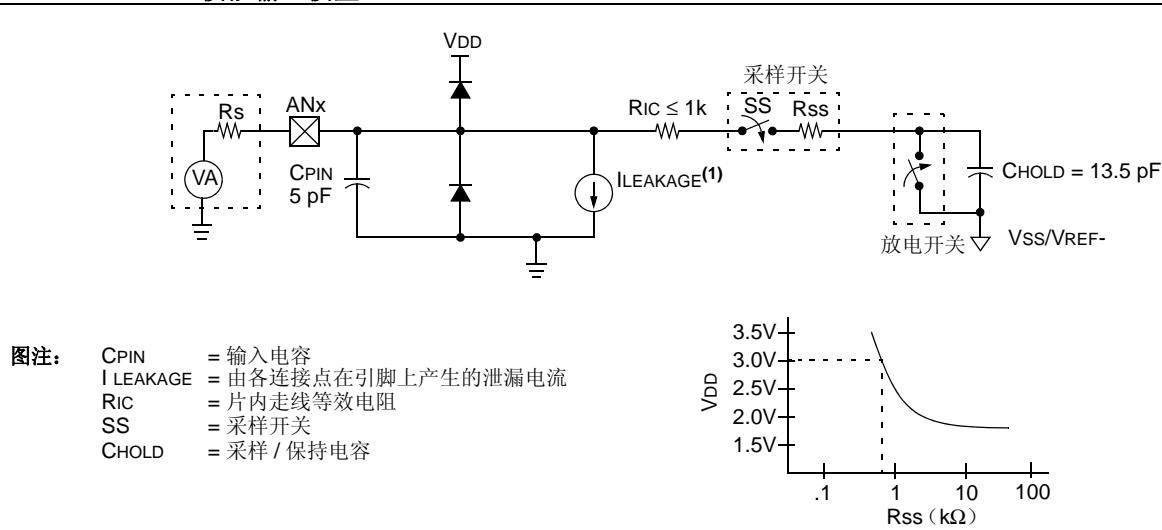
注 1：因为参考电压 (V_{REF}) 自行抵消，因此它对该公式没有影响。

2：充电保持电容 (CHOLD) 在每次转换后放电。

3：模拟信号源的最大阻抗推荐值为 10 k Ω 。此要求是为了符合引脚泄漏电流规范。

PIC18F2XK20/4XK20

图 19-5: 模拟输入模型



注 1：请参见第 26.0 节“电气特性”。

图 19-6: ADC 传递函数

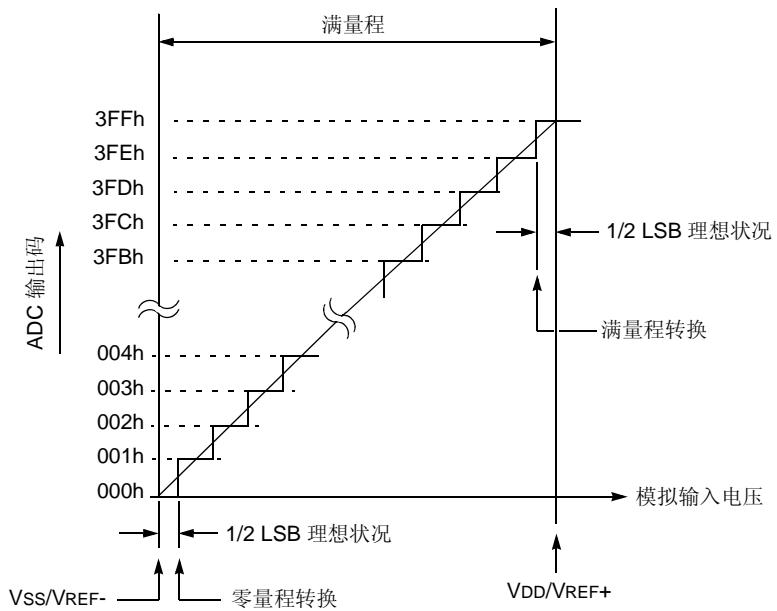


表 19-2: 与 A/D 操作相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR1	PSPIF ⁽¹⁾	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	62
PIE1	PSPIE ⁽¹⁾	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	62
IPR1	PSPIP ⁽¹⁾	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	62
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62
ADRESH	A/D 结果寄存器的高字节								61
ADRESL	A/D 结果寄存器的低字节								61
ADC0N0	—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON	61
ADC0N1	—	—	VCFG1	VCFG0	—	—	—	—	61
ADC0N2	ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0	61
ANSEL	ANS7 ⁽¹⁾	ANS6 ⁽¹⁾	ANS5 ⁽¹⁾	ANS4	ANS3	ANS2	ANS1	ANS0	62
ANSELH	—	—	—	ANS12	ANS11	ANS10	ANS9	ANS8	62
PORTA	RA7 ⁽²⁾	RA6 ⁽²⁾	RA5	RA4	RA3	RA2	RA1	RA0	62
TRISA	TRISA7 ⁽²⁾	TRISA6 ⁽²⁾	PORTA 数据方向控制寄存器						62
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	62
TRISB	PORTB 数据方向控制寄存器								62
LATB	PORTB 数据锁存寄存器 (读和写数据锁存器)								62
PORTE ⁽⁴⁾	—	—	—	—	RE3 ⁽³⁾	RE2	RE1	RE0	62
TRISE ⁽⁴⁾	IBF	OBF	IBOV	PSPMODE	—	TRISE2	TRISE1	TRISE0	62
LATE ⁽⁴⁾	—	—	—	—	—	PORTE 数据锁存寄存器			62

图注: — = 未实现, 读为 0。A/D 转换不使用阴影单元。

注 1: 这些位在 PIC18F2XK20 器件上未实现; 始终保持这些位清零。

2: PORTA<7:6> 及其方向位根据不同的主振荡器模式被单独配置为端口引脚。当被禁止时, 这些位读为 0。

3: 当 MCLRE 配置位为 0 时, RE3 端口位只能作为输入。

4: 这些寄存器在 PIC18F2XK20 器件上未实现。

PIC18F2XK20/4XK20

注:

20.0 比较器模块

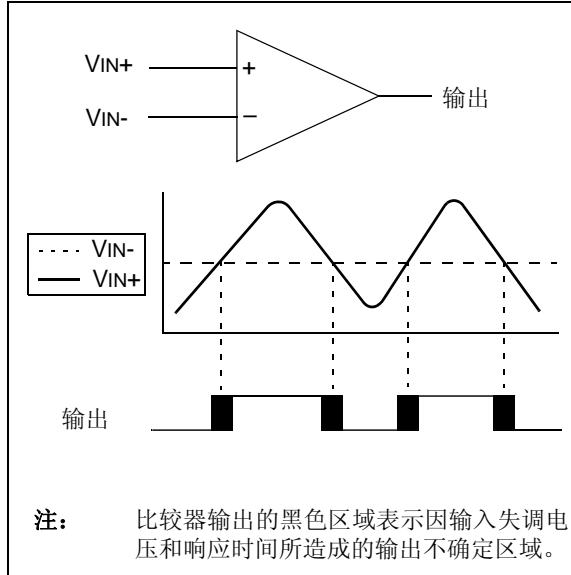
比较器模块通过比较两个模拟电压并提供其相对幅值的数字表示，用于建立模拟电路与数字电路的接口。比较器是非常有用的混合信号模块，因为它提供了与程序执行相独立的模拟功能。模拟比较器模块包括以下特性：

- 独立的比较器控制
- 可编程输入选择
- 比较器可提供内部 / 外部输出
- 可编程输出极性
- 电平变化中断
- 从休眠状态唤醒
- 可编程的速度 / 功耗优化
- PWM 关闭
- 可编程和固定参考电压

20.1 比较器概述

图20-1所示为单比较器以及模拟输入电平与数字输出之间的关系。当 V_{IN+} 上的模拟电压小于 V_{IN-} 上的模拟电压值时，比较器输出为数字低电平。当 V_{IN+} 上的模拟电压大于 V_{IN-} 上的模拟电压值时，比较器输出为数字高电平。

图 20-1： 单比较器



PIC18F2XK20/4XK20

图 20-2: 比较器 C1 的简化框图

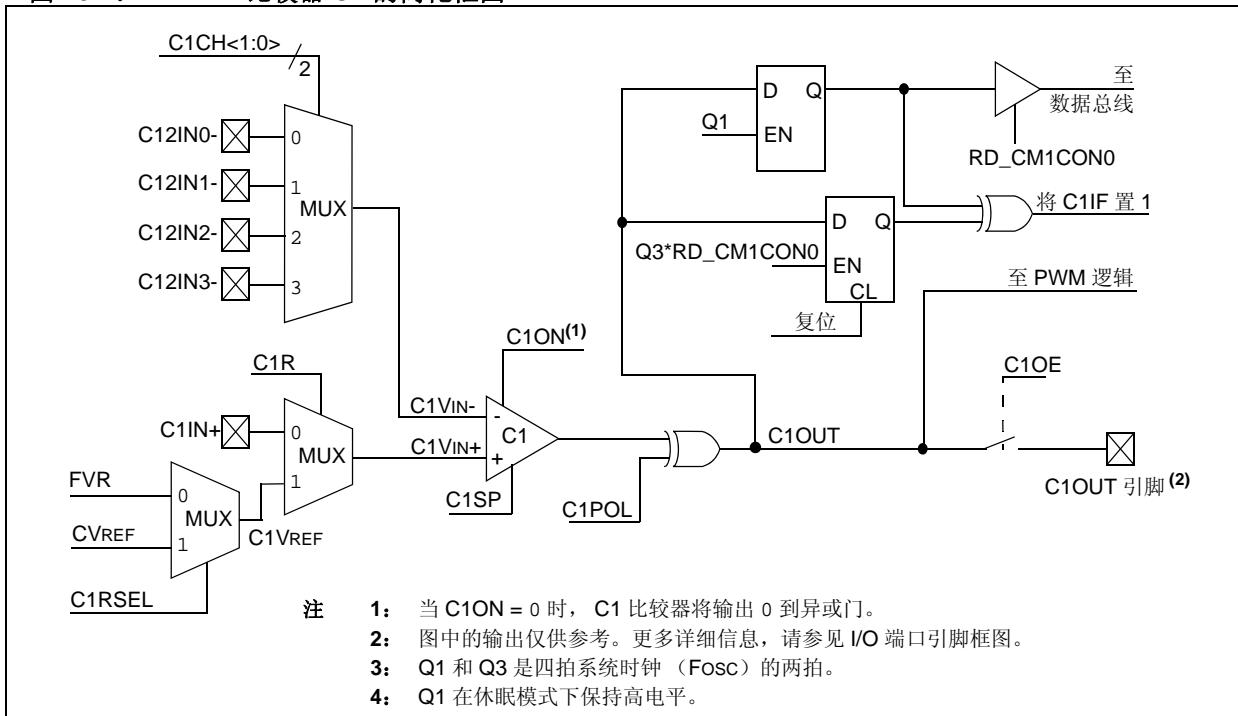
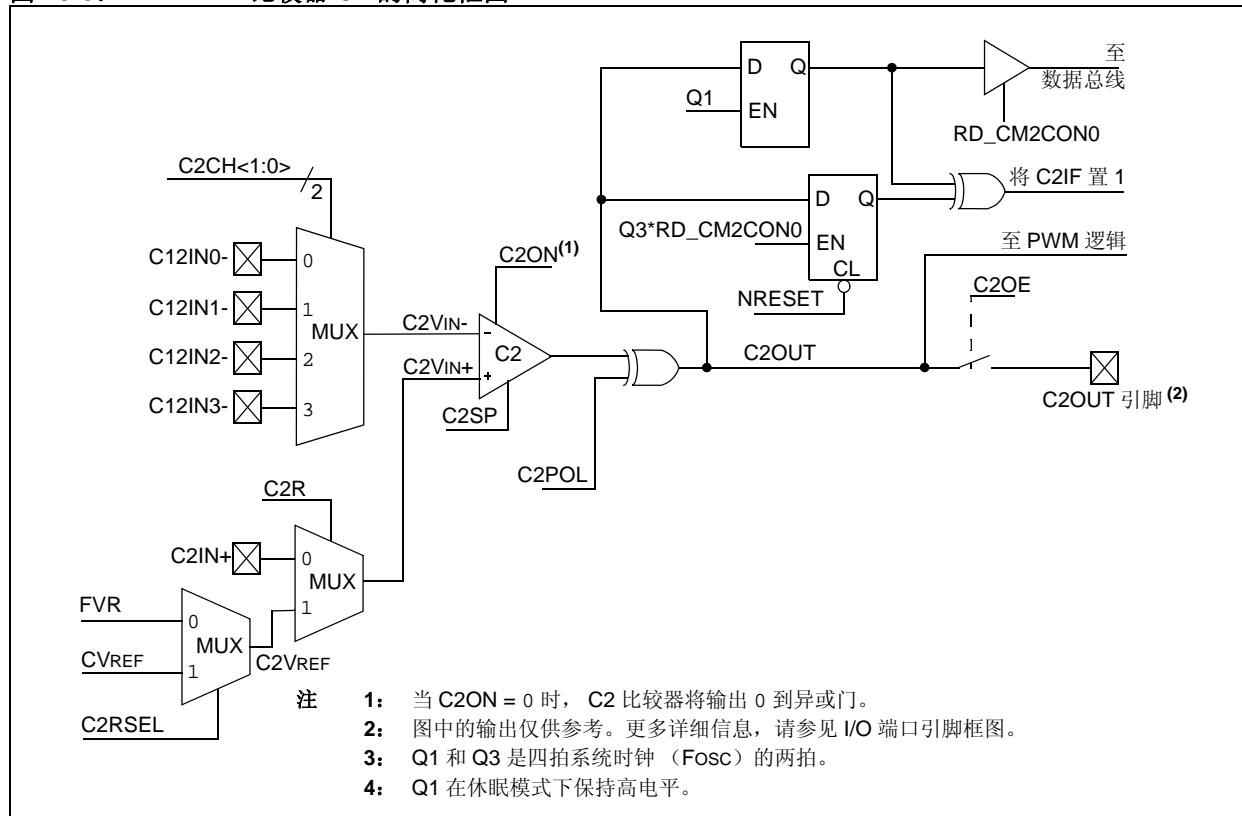


图 20-3: 比较器 C2 的简化框图



20.2 比较器控制

每个比较器都有独立的控制和配置寄存器：比较器 C1 为 CM1CON0，比较器 C2 为 CM2CON0。此外，比较器 C2 还有另外一个控制寄存器 CM2CON1，用于控制与 Timer1 的交互和两个比较器输出的同时读取。

CM1CON0 和 CM2CON0 寄存器（分别见寄存器 20-1 和 20-2）包含以下控制和状态位：

- 使能
- 输入选择
- 参考电压选择
- 输出选择
- 输出极性
- 速度选择

20.2.1 比较器使能

将 CMxCON0 寄存器的 CxON 位置 1 可以使能比较器操作。清零 CxON 位可以禁止比较器，以使电流消耗降至最低。

20.2.2 比较器输入选择

CMxCON0 寄存器的 CxCH<1:0> 位指示 4 个模拟输入引脚之一连接到比较器的反相输入。

注：要将 CxIN+ 和 C12INx- 引脚用作模拟输入，必须将 ANSEL 寄存器中相应的位置 1，同时也必须置 1 相应的 TRIS 位来禁止输出驱动器。

20.2.3 比较器参考电压选择

通过将 CMxCON0 寄存器的 CxR 位置 1，将内部参考电压或模拟输入引脚连接到比较器的同相输入。关于内部参考电压模块的更多信息，请参见第 21.0 节“参考电压”。

20.2.4 比较器输出选择

可以通过读 CMxCON0 寄存器的 CxOUT 位或 CM2CON1 寄存器的 MCxOUT 位监视比较器的输出。为了使输出可用于外部连接，必须满足以下条件：

- 必须将 CMxCON0 寄存器的 CxOE 位置 1
- 必须清零相应的 TRIS 位
- 必须将 CMxCON0 寄存器的 CxON 位置 1

注 1：CxOE 位改写端口数据锁存器。将 CxON 置 1 对端口改写没有影响。

2：比较器的内部输出在每个指令周期被锁存。除非另外指定，否则不锁存外部输出。

20.2.5 比较器输出极性

反相比较器的输出在功能上等同于交换比较器输入。可以通过将 CMxCON0 寄存器的 CxPOL 位置 1 来使比较器输出的极性反相。清零 CxPOL 位得到的是未反相的输出信号。

表 20-1 给出了输出状态与输入条件的关系（包括极性控制）。

表 20-1：比较器输出状态与输入条件

输入条件	CxPOL	CxOUT
CxVIN- > CxVIN+	0	0
CxVIN- < CxVIN+	0	1
CxVIN- > CxVIN+	1	1
CxVIN- < CxVIN+	1	0

20.2.6 比较器速度选择

在程序执行期间通过 CxSP 控制位可以最佳地在速度与功耗之间做出权衡。该位的默认状态为 1，选择正常速度模式。可以通过将 CxSP 位清零对器件功耗进行优化，代价是比较器传输延时变长。

20.3 比较器响应时间

在改变输入源或选择新的参考电压后，一段时间内比较器的输出状态都是不确定的。这段时间被称为响应时间。比较器的响应时间不同于参考电压的稳定时间。因此，在确定比较器输入改变的总响应时间时，必须考虑这两个时间。更多详细信息，请参见第 26.0 节“电气特性”中的比较器和参考电压规范。

20.4 比较器中断操作

只要比较器的输出值发生变化，相应的比较器中断标志位就会置 1。比较器输出值的变化由不匹配电路识别，该电路由两个锁存器和一个异或门组成（见图 20-2 和图 20-3）。当读 CMxCON0 寄存器时，一个锁存器更新为比较器的输出电平。该锁存器将保持该值，直到下次读 CMxCON0 寄存器，或者发生复位。不匹配电路的另一个锁存器在每个 Q1 系统时钟更新。当比较器的输出变化在 Q1 时钟周期输入到第二个锁存器时，即产生不匹配条件。这时两个不匹配锁存器具有相反的输出电平，由异或门检测并被送到中断电路。不匹配条件持续直到读 CMxCON0 寄存器或者比较器输出返回先前的状态。

- 注 1:** 对 CMxCON0 寄存器的写操作也将清除不匹配条件，因为所有写操作的写周期开始都包含一个读操作。
- 2:** 比较器中断的正常工作与 CxOE 的状态无关。

比较器中断是根据不匹配边沿而不是不匹配电平置 1 的。这意味着不需要额外的读取或写入 CMxCON0 寄存器来将不匹配寄存器清零这一步骤，即可将中断标志复位。当不匹配寄存器清零时，比较器输出返回前一个状态将发生中断，否则不会发生中断。

需要用软件保存比较器输出状态的信息（从 CMxCON0 寄存器或 CM2CON1 寄存器读取），以确定实际发生的变化。请参见图 20-4 和 20-5。

PIR2 寄存器的 CxIF 位是比较器中断标志。必须用软件将该位清零以将其复位。由于可以对该寄存器写入 1，因此可产生中断。

在中档器件兼容模式下，必须将 PIE2 寄存器的 CxIE 位以及 INTCON 寄存器的 PEIE 和 GIE 位都置 1 以允许比较器中断。如果上述位中有任何一位被清零，则无法允许中断，尽管中断条件发生时仍会将 PIR2 寄存器的 CxIF 位置 1。

20.4.1 预设不匹配锁存器

比较器不匹配锁存器可以在使能比较器之前预设为所需的状态。当比较器关闭时，CxPOL 位控制 CxOUT 电平。在 CxON 位清零时，将 CxPOL 位设置为所需的 CxOUT 非中断电平。然后，在 CxON 位置 1 的相同指令中配置所需的 CxPOL 电平。因为所有的寄存器写操作都按读 - 修改 - 写的方式执行，所以不匹配锁存器将在指令“读”阶段被清零，实际配置 CxON 和 CxPOL 位将在最后的“写”阶段进行。

图 20-4: 比较器中断时序
(不带 CMxCON0 读取)

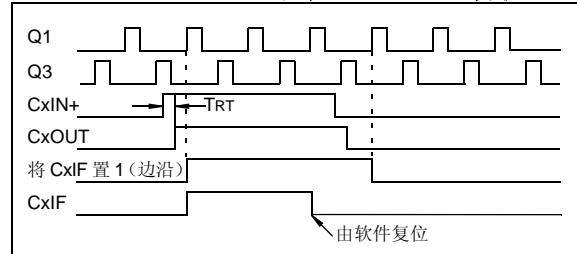
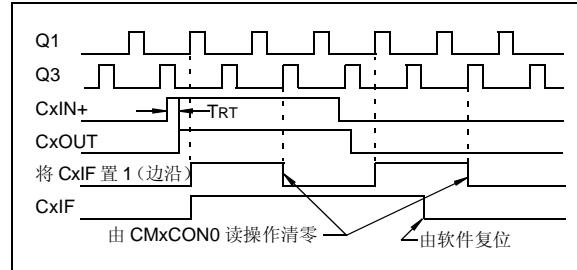


图 20-5: 比较器中断时序
(带 CMxCON0 读取)



- 注 1:** 在读操作执行过程中（Q2 周期的起始时刻），如果 CMxCON0 寄存器（CxOUT）发生变化，那么 PIR2 寄存器的 CxIF 中断标志位可能不会被置 1。

- 2:** 当两个比较器之一先使能时，比较器模块中的偏置电路在稳定前可能产生比较器的无效输出。应允许偏置电路有 1 μs 的稳定时间，然后在允许比较器中断前将不匹配条件和中断标志清除。

20.5 休眠期间的操作

如果在进入休眠模式之前比较器使能，则它在休眠期间仍将处于运行状态。**第 26.0 节“电气特性”**中单独给出了比较器消耗的额外电流。如果不使用比较器来唤醒器件，则在休眠模式下可通过关闭比较器使功耗降至最低。可以通过清零 CMxCON0 寄存器的 CxON 位来关闭每个比较器。

比较器输出变化可以将器件从休眠状态唤醒。要使比较器能将器件从休眠状态唤醒，必须将 PIE2 寄存器的 CxIE 位和 INTCON 寄存器的 PEIE 位置 1。紧跟 SLEEP 指令后的指令总是在从休眠状态唤醒之后执行。如果也将 INTCON 寄存器的 GIE 位置 1，则器件将执行中断服务程序。

20.6 复位的影响

器件复位强制 CMxCON0 和 CM2CON1 寄存器为复位状态。这使比较器和参考电压被强制为“关闭”状态。

PIC18F2XK20/4XK20

寄存器 20-1： CM1CON0： 比较器 1 的控制寄存器 0

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C1ON	C1OUT	C1OE	C1POL	C1SP	C1R	C1CH1	C1CH0
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **C1ON:** 比较器 C1 使能位

1 = 使能比较器 C1

0 = 禁止比较器 C1

bit 6 **C1OUT:** 比较器 C1 输出位

如果 C1POL = 1 (极性反相)：

当 C1VIN+ > C1VIN- 时, C1OUT = 0

当 C1VIN+ < C1VIN- 时, C1OUT = 1

如果 C1POL = 0 (极性不反相)：

当 C1VIN+ > C1VIN- 时, C1OUT = 1

当 C1VIN+ < C1VIN- 时, C1OUT = 0

bit 5 **C1OE:** 比较器 C1 输出使能位

1 = C1OUT 出现在 C1OUT 引脚上 ⁽¹⁾

0 = C1OUT 仅在内部有效

bit 4 **C1POL:** 比较器 C1 输出极性选择位

1 = C1OUT 逻辑反相

0 = C1OUT 逻辑不反相

bit 3 **C1SP:** 比较器 C1 速度 / 功耗选择位

1 = C1 工作在正常功耗、高速模式下

0 = C1 工作在低功耗、低速模式下

bit 2 **C1R:** 比较器 C1 参考电压选择位 (同相输入)

1 = C1VIN+ 连接至 C1VREF 输出

0 = C1VIN+ 连接至 C1IN+ 引脚

bit 1-0 **C1CH<1:0>:** 比较器 C1 通道选择位

00 = C1 的 C12IN0- 引脚连接至 C1VIN-

01 = C1 的 C12IN1- 引脚连接至 C1VIN-

10 = C1 的 C12IN2- 引脚连接至 C1VIN-

11 = C1 的 C12IN3- 引脚连接至 C1VIN-

注 1： 比较器输出需要以下 3 个条件： C1OE = 1, C1ON = 1 且相应的端口 TRIS 位 = 0。

寄存器 20-2: CM2CON0: 比较器 2 的控制寄存器 0

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2ON	C2OUT	C2OE	C2POL	C2SP	C2R	C2CH1	C2CH0
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	C2ON: 比较器 C2 使能位 1 = 使能比较器 C2 0 = 禁止比较器 C2
bit 6	C2OUT: 比较器 C2 输出位 <u>如果 C2POL = 1 (极性反相):</u> 当 C2VIN+ > C2VIN- 时, C2OUT = 0 当 C2VIN+ < C2VIN- 时, C2OUT = 1 <u>如果 C2POL = 0 (极性不反相):</u> 当 C2VIN+ > C2VIN- 时, C2OUT = 1 当 C2VIN+ < C2VIN- 时, C2OUT = 0
bit 5	C2OE: 比较器 C2 输出使能位 1 = C2OUT 出现在 C2OUT 引脚上 ⁽¹⁾ 0 = C2OUT 仅在内部有效
bit 4	C2POL: 比较器 C2 输出极性选择位 1 = C2OUT 逻辑反相 0 = C2OUT 逻辑不反相
bit 3	C2SP: 比较器 C2 速度 / 功耗选择位 1 = C2 工作在正常功耗、高速模式下 0 = C2 工作在低功耗、低速模式下
bit 2	C2R: 比较器 C2 参考电压选择位 (同相输入) 1 = C2VIN+ 连接至 C2VREF 0 = C2VIN+ 连接至 C2IN+ 引脚
bit 1-0	C2CH<1:0>: 比较器 C2 通道选择位 00 = C2 的 C12IN0- 引脚连接至 C2VIN- 01 = C2 的 C12IN1- 引脚连接至 C2VIN- 10 = C2 的 C12IN2- 引脚连接至 C2VIN- 11 = C2 的 C12IN3- 引脚连接至 C2VIN-

注 1: 比较器输出需要以下 3 个条件: C2OE = 1, C2ON = 1 且相应的端口 TRIS 位 = 0。

20.7 模拟输入连接注意事项

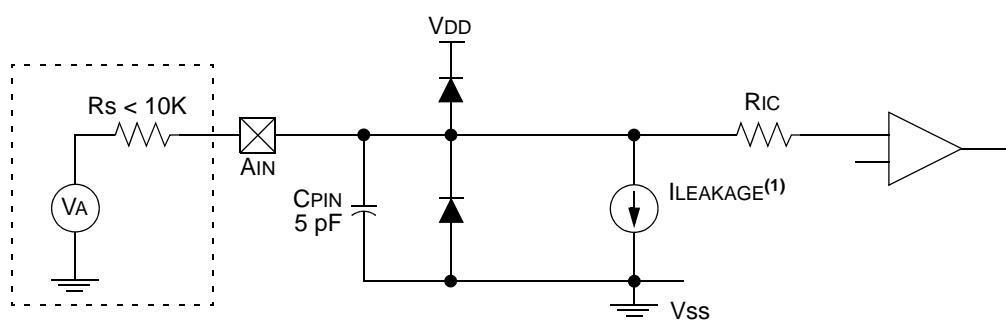
模拟输入的简化电路如图 20-6 所示。由于模拟输入引脚与数字输入共用连接，它们在 VDD 和 VSS 之间连有反向偏置的 ESD 保护二极管。因此，模拟输入必须在 VSS 和 VDD 之间。如果输入电压与这一范围偏离的绝对值超过 0.6V，就可能发生一个二极管正向导通，从而可能导致锁死发生。

模拟信号源的最大阻抗推荐值为 $10\text{ k}\Omega$ 。任何连接到模拟输入引脚的外部元件（如电容或齐纳二极管），应保证其泄漏电流极小以使引入的误差降至最低。

注 1: 读端口寄存器时，所有配置为模拟输入的引脚均读为 0。配置为数字输入的引脚将根据输入规范转换为模拟输入。

2: 定义为数字输入引脚上的模拟电平可能会使输入缓冲器的电流消耗超过规定值。

图 20-6： 模拟输入模型



图注：
CPIN = 输入电容
ILEAKAGE = 由各连接点在引脚上产生的泄漏电流
RIC = 片内走线等效电阻
RS = 信号源阻抗
VA = 模拟电压

注 1：请参见第 26.0 节 “电气特性”。

20.8 其他比较器特性

共有 2 个额外的比较器特性：

- 比较器输出的同时读取
- 内部参考电压选择

20.8.1 比较器输出的同时读取

CM2CON1 寄存器的 MC1OUT 和 MC2OUT 位是两个比较器输出的镜像副本。从同一个寄存器中同时读取两个输出的功能，可以消除从不同寄存器读取时存在的时序错位。

注 1：通过读 CM2CON1 获取 C1OUT 或 C2OUT 的状态并不影响比较器中断不匹配寄存器。

20.8.2 内部参考电压选择

有两个内部参考电压可供每个比较器的同相输入使用。其中一个是 1.2V 固定参考电压 (FVR)，另一个是可变比较器参考电压 (CVREF)。CM2CON 寄存器的 CxRSEL 位决定其中的哪一个参考电压连接到比较器参考电压输出 (CxVREF)。参考电压与比较器的具体连接通过 CMxCON0 寄存器的 CxR 位控制。更多详细信息，请参见第21.1节“比较器参考电压”、图20-2和图20-3。

寄存器 20-3： CM2CON1：比较器 2 的控制寄存器 1

R-0	R-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
MC1OUT	MC2OUT	C1RSEL	C2RSEL	—	—	—	—
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现位，读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **MC1OUT:** C1OUT 的镜像副本位

bit 6 **MC2OUT:** C2OUT 的镜像副本位

bit 5 **C1RSEL:** 比较器 C1 参考电压选择位

1 = CVREF 连接至 C1VREF 输入

0 = FVR (1.2V 固定参考电压) 连接至 C1VREF 输入

bit 4 **C2RSEL:** 比较器 C2 参考电压选择位

1 = CVREF 连接至 C2VREF 输入

0 = FVR (1.2V 固定参考电压) 连接至 C2VREF 输入

bit 3-0 未实现：读为 0

PIC18F2XK20/4XK20

表 20-2：与比较器模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
CM1CON0	C1ON	C1OUT	C1OE	C1POL	C1SP	C1R	C1CH1	C1CH0	62
CM2CON0	C2ON	C2OUT	C2OE	C2POL	C2SP	C2R	C2CH1	C2CH0	62
CM2CON1	MC1OUT	MC2OUT	C1RSEL	C2RSEL	—	—	—	—	63
CVRCN	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	61
CVRCN2	FVREN	FVRST	—	—	—	—	—	—	61
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62
PORTA	RA7 ⁽¹⁾	RA6 ⁽¹⁾	RA5	RA4	RA3	RA2	RA1	RA0	62
LATA	LATA7 ⁽¹⁾	LATA6 ⁽¹⁾	PORTA 数据锁存寄存器（读和写数据锁存器）						62
TRISA	TRISA7 ⁽¹⁾	TRISA6 ⁽¹⁾	PORTA 数据方向控制寄存器						62

图注：— = 未实现，读为 0。比较器模块不使用阴影单元。

注 1：PORTA<7:6> 及其方向和锁存位根据不同的主振荡器模式被单独配置为端口引脚。当被禁止时，这些位读为 0。

21.0 参考电压

有两种独立的参考电压：

- 可编程的比较器参考电压
- 1.2V 固定参考电压

21.1 比较器参考电压

比较器参考电压模块为比较器提供了内部产生的参考电压。它具有以下特性：

- 与比较器操作独立
- 两个 16 级电压范围
- 输出钳位到 V_{SS}
- 与 V_{DD} 成比例
- 1.2V 固定参考电压 (FVR)

CVRCON 寄存器（寄存器 21-1）控制参考电压模块，如图 21-1 所示。

21.1.1 独立操作

比较器参考电压独立于比较器配置。将 CVRCON 寄存器的 CVREN 位置 1，允许电流流入 CVREF 分压器，将使能参考电压。当同时清零两个 CVREN 位时，禁止电流流入 CVREF 分压器，从而将参考电压外设的功耗降至最低。

21.1.2 输出电压选择

CVREF 参考电压有 2 个范围，每个范围有 16 级电压。范围的选择由 CVRCON 寄存器的 CVRR 位控制。16 级电压通过 CVRCON 寄存器的 CVR<3:0> 位进行设置。

CVREF 输出电压由以下公式确定：

公式 21-1： CVREF 输出电压

CVRR = 1 (低电压范围)：

$$CVREF = (CVRSRC/24) \times CVR<3:0> + VREF-$$

CVRR = 0 (高电压范围)：

$$CVREF = (CVRSRC/32) \times (8 + CVR<3:0>) + VREF-$$

$$CVRSRC = VDD \text{ 或 } [(VREF+) - (VREF-)]$$

注：当 CVRSS = 0 时，VREF- 为 0。

由于模块的构造所限，无法实现 V_{SS} 到 V_{DD} 的满量程。请参见图 21-1。

21.1.3 输出钳位到 V_{SS}

可以将 CVREF 输出电压设为 V_{SS} 且不产生功耗，方法是将 CVRCON 配置如下：

- CVREN = 0
- CVRR = 1
- CVR<3:0> = 0000

这使得比较器可以检测到过零点，且不额外消耗 CVREF 模块电流。

21.1.4 输出与 V_{DD} 成比例

比较器参考电压来自 V_{DD}，因此，CVREF 输出会随着 V_{DD} 的变化而变化。经过测试的比较器参考电压的绝对精度，请参见第 26.0 节“电气特性”。

21.1.5 参考电压输出

可以通过将 CVRCON 寄存器的 CVROE 位设为 1，可将 CVREF 参考电压输出到器件 CVREF 引脚。选择将参考电压输出到 CVREF 引脚会自动改写数字输出缓冲器和数字输入门限值检测器功能。当 CVREF 引脚已被配置为参考电压输出时，读取该引脚始终返回 0。

要提高电流驱动能力，CVREF 参考电压输出端必须外接缓冲器。图 21-2 举例说明了这一缓冲技术。

21.1.6 休眠期间的操作

如果因中断或看门狗定时器超时将器件从休眠模式唤醒，CVRCON 寄存器的内容将不受影响。为了降低休眠模式下的电流消耗，应禁止参考电压模块。

21.1.7 复位的影响

器件复位会产生以下影响：

- 禁止比较器参考电压
- 禁止固定参考电压
- CVREF 从 CVREF 引脚移除
- 选择高电压范围
- CVR<3:0> 范围选择位被清零

21.2 FVR 参考电压模块

FVR 参考电压是稳定的固定参考电压，独立于 VDD，输出电压标称值为 1.2V。可以通过将 CVRCON2 寄存器的 FVREN 位设为 1 使能该参考电压。当 HFINTOSC、HLVD 或 BOR 功能中的一个或多个使能时，FVR 默认为使能。FVR 参考电压可以连接至比较器或 ADC 输入通道。

21.2.1 FVR 稳定周期

当固定参考电压模块使能时，需要一段时间使参考电压及其放大电路达到稳定。用户程序必须包含允许模块稳定的延时子程序。CVRCON2 寄存器的 FVRST 稳定位还用于指示 FVR 参考电压是否已工作足够长时间达到稳定。关于最小延时要求，请参见第 26.0 节“电气特性”。

图 21-1：参考电压框图

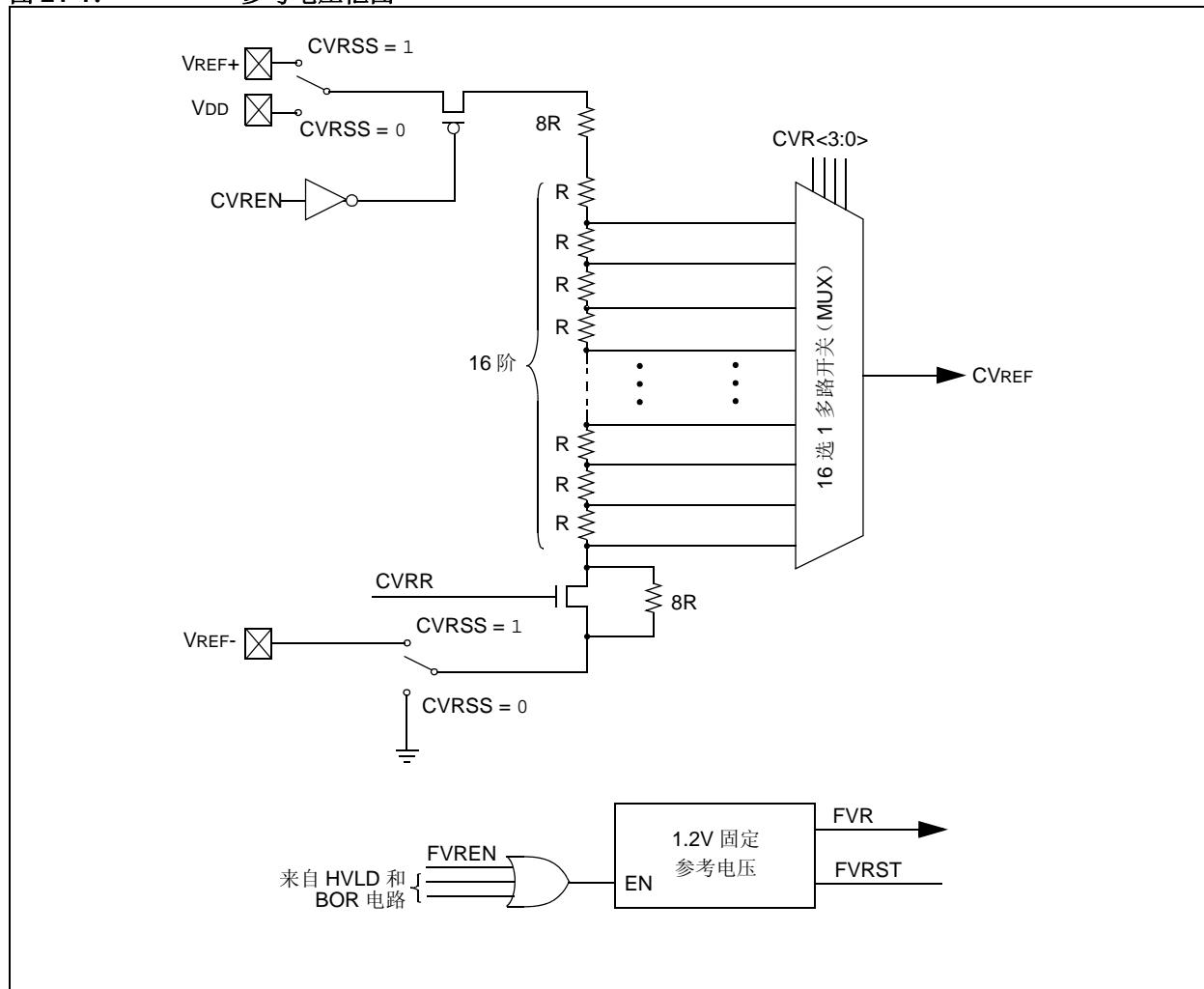
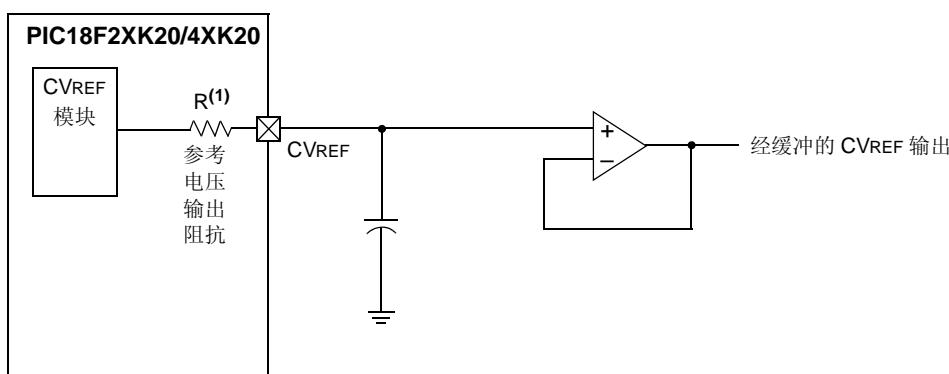


图 21-2: 参考电压输出缓冲示例



注 1: R 的值取决于参考电压配置位 CVR<3:0> 和 CVRR。

寄存器 21-1: CVRCON: 比较器参考电压控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVREN	CVROE ⁽¹⁾	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **CVREN:** 比较器参考电压使能位

1 = CVREF 电路上电

0 = CVREF 电路断电

bit 6 **CVROE:** 比较器 VREF 输出使能位⁽¹⁾

1 = CVREF 电平也从 CVREF 引脚输出

0 = CVREF 电压从 CVREF 引脚断开

bit 5 **CVRR:** 比较器 VREF 范围选择位

1 = 0 到 0.667 CVRSRC, 步长为 CVRSRC/24 (低电压范围)

0 = 0.25 CVRSRC 到 0.75 CVRSRC, 步长为 CVRSRC/32 (高电压范围)

bit 4 **CVRSS:** 比较器 VREF 源选择位

1 = 比较器参考电压源, CVRSRC = (VREF+) - (VREF-)

0 = 比较器参考电压源, CVRSRC = VDD - VSS

bit 3-0 **CVR<3:0>:** 比较器 VREF 值选择位 ($0 \leq (\text{CVR}<3:0>) \leq 15$)

当 CVRR = 1 时:

$\text{CVREF} = ((\text{CVR}<3:0>)/24) \bullet (\text{CVRSRC}) + \text{VREF}-$

当 CVRR = 0 时:

$\text{CVREF} = (\text{CVRSRC}/4) + ((\text{CVR}<3:0>)/32) \bullet (\text{CVRSRC}) + \text{VREF}-$

注 1: CVROE 改写 TRISA<2> 位设置。

PIC18F2XK20/4XK20

寄存器 21-2: CVRCON2: 比较器参考电压控制寄存器 2

R/W-0	R-0	U-0						
FVREN	FVRST	—	—	—	—	—	—	—
bit 7	bit 0							

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7 **FVREN:** 固定参考电压使能位

1 = FVR 电路上电

0 = FVR 电路不能通过 FVREN 使能。其他外设可使能 FVR。

bit 6 **FVRST:** 固定电压稳定状态位

1 = FVR 稳定, 可以使用。

0 = FVR 不稳定, 不应使用。

bit 5-0 未实现: 读为 0。

表 21-1: 与比较器参考电压相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
CVRCON	CVREN	CVROE	CVRR	CVRSS	CVR3	CVR2	CVR1	CVR0	62
CVRCON2	FVREN	FVRST	—	—	—	—	—	—	61
CM1CON0	C1ON	C1OUT	C1OE	C1POL	C1SP	C1R	C1CH1	C1CH0	62
CM2CON0	C2ON	C2OUT	C2OE	C2POL	C2SP	C2R	C2CH1	C2CH0	62
CM2CON1	MC1OUT	MC2OUT	C1RSEL	C2RSEL	—	—	—	—	63
TRISA	TRISA7 ⁽¹⁾	TRISA6 ⁽¹⁾	PORTA 数据方向控制寄存器						62

图注: 比较器参考电压模块不使用阴影单元。

注 1: PORTA 引脚根据振荡器配置情况使能。

22.0 高 / 低压检测 (HLVD)

PIC18F2XK20/4XK20 器件具有一个高 / 低压检测 (HLVD) 模块。该模块是一个可编程的电路，它允许用户指定器件的电压跳变点和变化方向。如果器件电压按照指定的方向相对于该跳变点发生了偏离，就会将中断标志位置 1。如果允许了中断，程序将跳转到中断向量地址处执行，由软件响应该中断。

高 / 低压检测控制寄存器（寄存器 22-1）完全控制 HLVD 模块的工作。用户可通过软件控制该寄存器将电路“关闭”，从而使器件的电流消耗降至最低。

图 22-1 给出了 HLVD 模块的框图。

通过将 **HLVDEN** 位置 1 使能该模块。每次使能 HLVD 模块时，电路需要一定时间才能稳定下来。**IRVST** 位是一个只读位，用于指示电路何时稳定。仅当电路稳定且 **IRVST** 位置 1 后，该模块才能产生中断。

VDIRMAG 位决定该模块的整体工作状态。当 **VDIRMAG** 清零时，模块监视 **VDD** 看它是否降到预先确定的设置点以下。当该位置 1 时，模块监视 **VDD** 看它是否上升到设置点以上。

寄存器 22-1： **HLVDCON**: 高 / 低压检测控制寄存器

R/W-0	U-0	R-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1
VDIRMAG	—	IRVST	HLVDEN	HLVDL3 ⁽¹⁾	HLVDL2 ⁽¹⁾	HLVDL1 ⁽¹⁾	HLVDL0 ⁽¹⁾
bit 7	bit 0						

图注：

R = 可读位

W = 可写位

U = 未实现

C = 只可清零位

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7	VDIRMAG : 电压方向大小选择位 1 = 当电压等于或超过跳变点 (HLVDL<3:0>) 时，事件发生 0 = 当电压等于或低于跳变点 (HLVDL<3:0>) 时，事件发生
bit 6	未实现：读为 0
bit 5	IRVST : 内部参考电压稳定标志位 1 = 表示电压检测逻辑在检测到指定的电压范围时，产生中断标志 0 = 表示电压检测逻辑在检测到指定的电压范围时，不会产生中断标志，并且 HLVD 中断不被允许
bit 4	HLVDEN : 高 / 低压检测模块使能位 1 = 使能 HLVD 0 = 禁止 HLVD
bit 3-0	HLVDL<3:0> : 电压检测限制位 ⁽¹⁾ 1111 = 使用外部模拟输入（输入来自于 HLVDIN 引脚） 1110 = 最大设置 • • • 0000 = 最小设置

注 1: 具体规范请参见表 26-4。

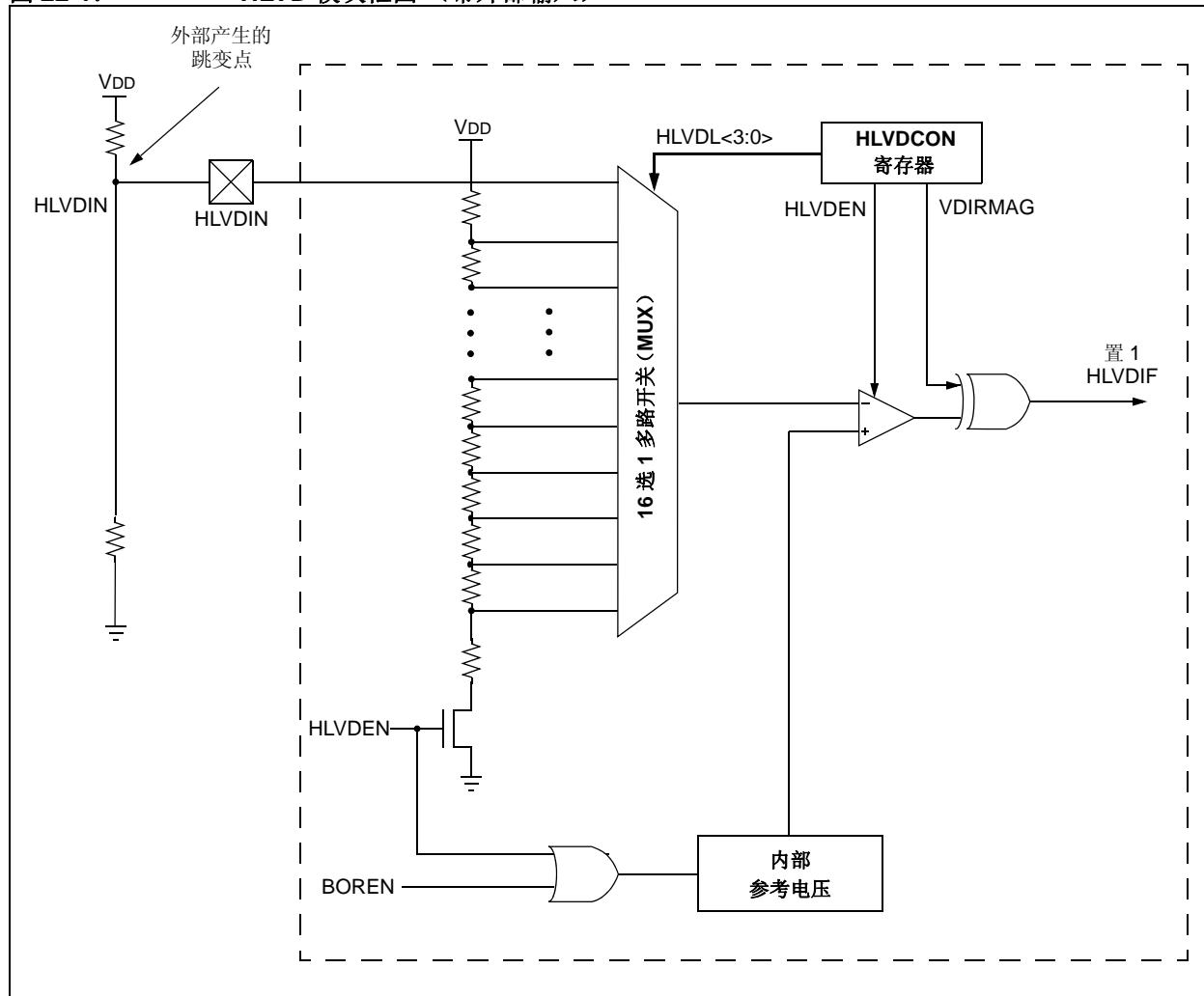
22.1 工作原理

当使能了 HLVD 模块时，比较器使用内部产生的参考电压作为设置点。将设置点的电压与跳变点电压作比较，其中电阻分压器中的每个节点均代表一个电压跳变点。“跳变点”电压是被检测到的高压或低压事件的电平，它取决于该模块的配置。当供电电压等于跳变点电压时，电阻阵列的节点电压输出值等于由参考电压模块产生的内部参考电压。然后比较器通过将 HLVDIF 位置 1 产生一个中断信号。

可用软件编程设定跳变点电压为 16 个值中的任何一个。通过对 HLVDCON 寄存器的 HLVDL<3:0> 位进行编程可以选择跳变点。

HLVD 模块还有一个额外的功能，允许用户通过外部电源向模块提供跳变电压。当 HLVDL<3:0> 位被设置为 1111 时，使能该模块。在此状态下，比较器输入与外部输入引脚 HLVDIN 复用。因此用户可以灵活地配置高/低压检测中断，使之可以在有效工作范围内的任何电压点上产生。

图 22-1： HLVD 模块框图（带外部输入）



22.2 HLVD 设置

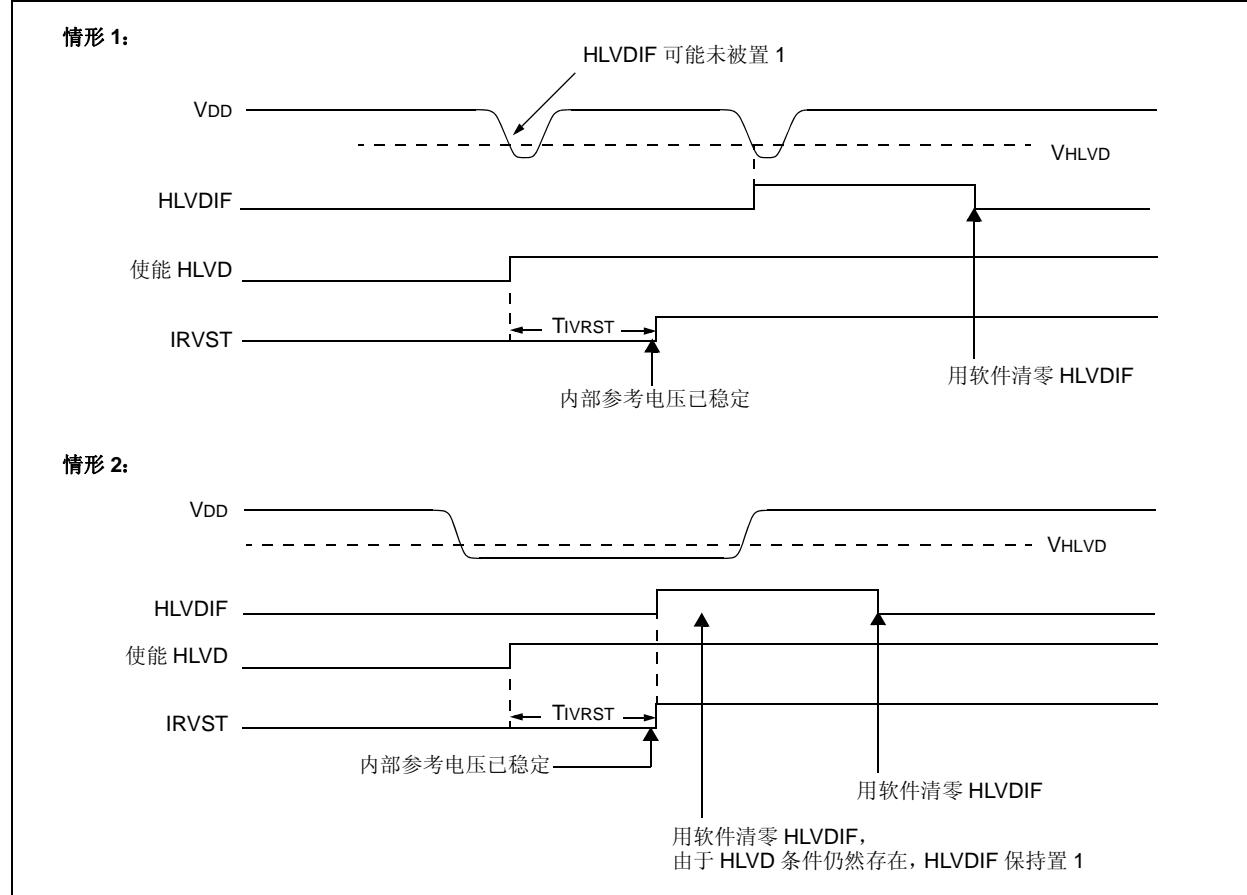
要设置 HLVD 模块，需要遵循以下步骤：

1. 将值写入 $\text{HLVDL}<3:0>$ 位，选择所需的 HLVD 跳变点。
2. 将 VDIRMAG 位设置为检测高压 ($\text{VDIRMAG} = 1$) 或低压 ($\text{VDIRMAG} = 0$)。
3. 通过将 HLVDEN 位置 1，使能 HLVD 模块。
4. 清零 PIR2 寄存器的 HLVD 中断标志位，该位可能被上次中断置 1。
5. 如果需要中断，通过将 PIE2 寄存器的 HLVDIE 位、 INTCON 寄存器的 GIE 和 PEIE 位置 1，允许 HLVD 中断。直到 IRVST 位也置 1 时才会发生中断。

22.3 电流消耗

使能了该模块就使能了 HLVD 比较器和分压器，并将消耗静态电流。电气规范中的参数 D024B 给出了使能该模块时的电流总消耗。

图 22-2：低压检测工作原理 ($\text{VDIRMAG} = 0$)



HLVD 模块无需一直工作，工作与否取决于具体的应用。要降低电流消耗，只需要在检测电压时，短时间地使能 HLVD 电路，在检测完成之后马上禁止 HLVD 模块。

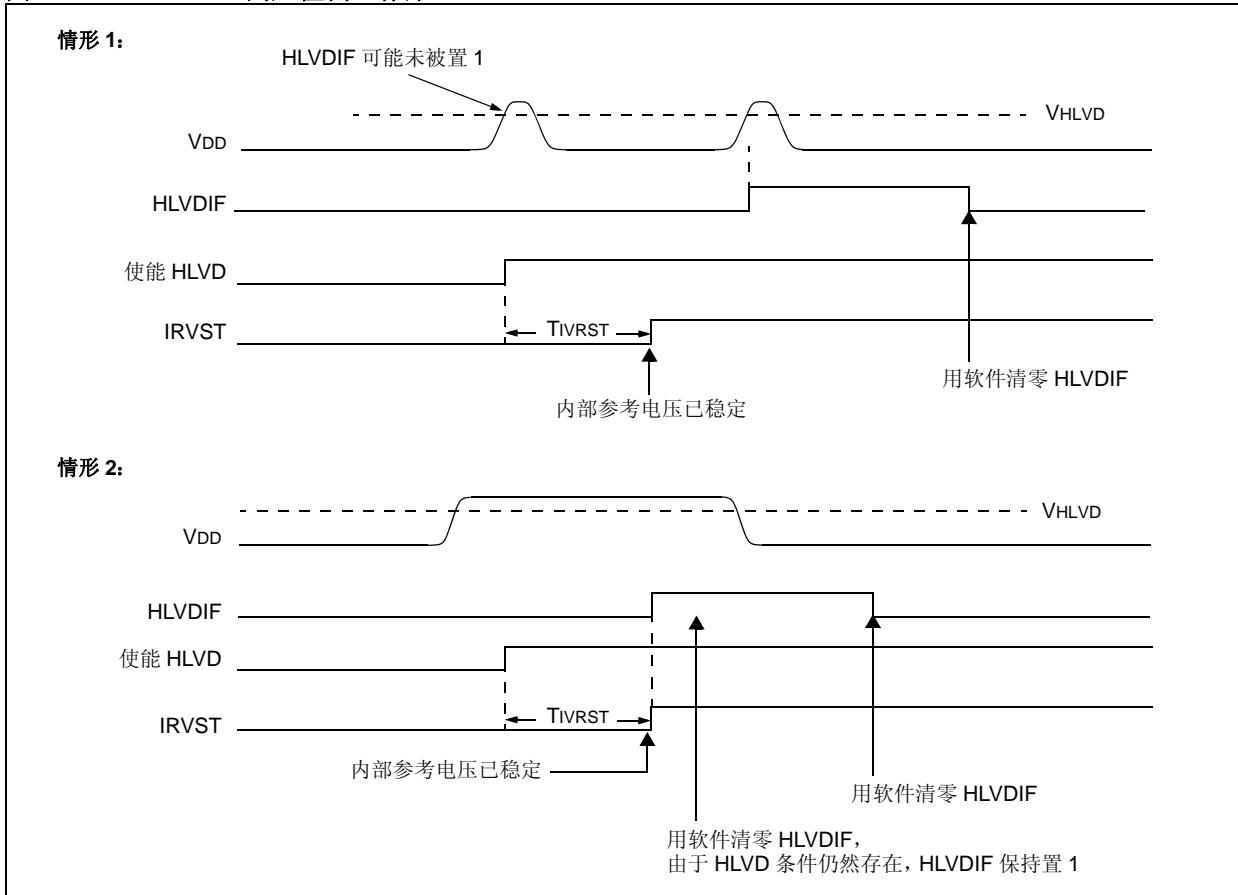
22.4 HLVD 启动时间

电气规范中的参数 D420 规定了 HLVD 模块的内部参考电压，该参考电压也可供其他内部电路（如可编程欠压复位电路）使用。如果禁止了 HLVD 或其他使用参考电压的电路以降低器件的电流消耗，则参考电压电路将需要一段时间稳定下来以后才能可靠地检测低压或高压条件。HLVD 启动时间 TIRVST 与器件时钟速度无关。它由电气规范中的参数 36 规定。

直到 TIRVST 结束并且参考电压达到稳定后才会允许 HLVD 中断标志。基于此原因，在此时间间隔期间，超出设置点的短暂偏离可能不会被检测到。请参见图 22-2 或图 22-3。

PIC18F2XK20/4XK20

图 22-3: 高压检测工作原理 ($VDIRMAG = 1$)

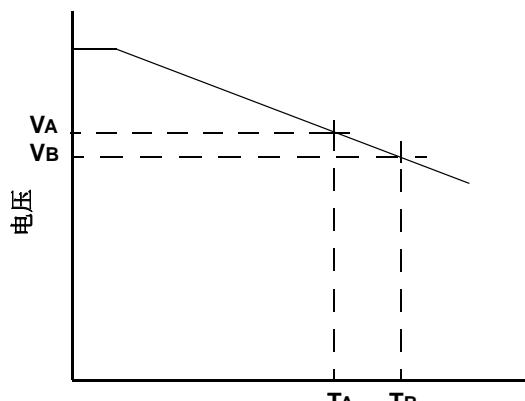


22.5 应用

在许多应用中，当电压低于或高于某个门限值时，系统希望可以检测到该事件。例如，可以定期使能 HLVD 模块来检测是否连接通用串行总线（Universal Serial Bus, USB）。这里假设断开连接时器件的供电电压低于 USB 电压。若连接了 USB，将检测到 3.3V 到 5V 的高压（USB 上的电压）；如果断开连接，情况正好相反。此功能可以省去一些额外的元件和连接信号（输入引脚）。

对于一般的电池应用，图 22-4 给出了一个近似的电压曲线。器件电压会随时间逐渐下降。当器件电压达到电压 V_A 时，HLVD 逻辑电路会在时间 T_A 产生中断。该中断将导致执行中断服务子程序，从而使应用程序能在器件电压退出有效工作范围（对应的时间为 T_B ）之前执行“日常任务”，并执行受控关闭。因此，HLVD 将会提供一个时间窗（表示为 T_A 和 T_B 的时间差）使应用程序能安全地退出。

图 22-4：典型低压检测应用



图注：
 V_A = HLVD 跳变点
 V_B = 器件的最低有效工作电压

表 22-1：与高 / 低压检测模块相关的寄存器

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
HLVDCON	VDIRMAG	—	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0	60
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	59
PIR2	OSCFIF	C1IF	C2IF	EEIF	BCLIF	HLVDIF	TMR3IF	CCP2IF	62
PIE2	OSCFIE	C1IE	C2IE	EEIE	BCLIE	HLVDIE	TMR3IE	CCP2IE	62
IPR2	OSCFIP	C1IP	C2IP	EEIP	BCLIP	HLVDIP	TMR3IP	CCP2IP	62

图注：— = 未实现，读为 0。HLVD 模块不使用阴影单元。

PIC18F2XK20/4XK20

注:

23.0 CPU 的特殊功能

PIC18F2XK20/4XK20器件包含的功能旨在最大限度地提高系统可靠性，并通过减少外部元件把系统成本降到最低。这些功能包括：

- 振荡器选择
- 复位：
 - 上电复位（POR）
 - 上电延时定时器（PWRT）
 - 振荡器起振定时器（OST）
 - 欠压复位（BOR）
- 中断
- 看门狗定时器（WDT）
- 代码保护
- ID 存储单元
- 在线串行编程

要根据具体应用对频率、功耗、精度和成本的要求来选择振荡器。在第 2.0 节“振荡器模块（带故障保护时钟监视器）”中详细讨论了所有的选项。

在本数据手册的前面几章中已完整地讨论了器件的复位和中断。

除了为复位提供了上电延时定时器和振荡器起振定时器之外，PIC18F2XK20/4XK20 器件还提供了一个看门狗定时器，该定时器可通过配置位永久使能或用软件控制（如果看门狗定时器使能位配置为禁止的话）。

器件自带的内部 RC 振荡器还提供了故障保护时钟监视器（FSCM）和双速启动这两个额外的功能。FSCM 对外设时钟进行后台监视，并在外设时钟发生故障时自动切换时钟源。双速启动使得几乎可在启动发生那一刻立即执行代码，同时主时钟源继续其起振延时。

通过设置相应的配置寄存器位可以使能和配置所有这些功能。

23.1 配置位

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以 300000h 开始的单元中。

用户会注意到地址 300000h 超出了用户程序存储空间范围。事实上，它属于配置存储空间（300000h-3FFFFh），这一空间仅能通过表读和表写进行访问。

对配置寄存器进行编程的方式与对闪存进行编程的方式类似。EECON1 寄存器中的 WR 位启动自定时写入配置寄存器。在正常操作模式下，TBLPTR 指向配置寄存器的 TBLWLT 指令会设置写配置寄存器要用到的地址和数据。将 WR 位置 1 会启动对配置寄存器的长写操作。配置寄存器一次被写入一个字节。TBLWT 指令可以将 1 或 0 写入单元来改写配置单元的内容。关于闪存编程的更多详细信息，请参见第 6.5 节“写闪存程序存储器”。

表 23-1：配置位和器件 ID

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	默认 / 未编程值
300001h	CONFIG1H	IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0	00-- 0111
300002h	CONFIG2L	—	—	—	BORV1	BORV0	BOREN1	BOREN0	PWRREN	---1 1111
300003h	CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	---1 1111
300005h	CONFIG3H	MCLRE	—	—	—	HFOFST	LPT1OSC	PBADEN	CCP2MX	1--- 1011
300006h	CONFIG4L	DEBUG	XINST	—	—	—	LVP	—	STVREN	10-- -1-1
300008h	CONFIG5L	—	—	—	—	CP3 ⁽¹⁾	CP2 ⁽¹⁾	CP1	CP0	---- 1111
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah	CONFIG6L	—	—	—	—	WRT3 ⁽¹⁾	WRT2 ⁽¹⁾	WRT1	WRT0	---- 1111
30000Bh	CONFIG6H	WRTD	WRTB	WRTE	—	—	—	—	—	111- ----
30000Ch	CONFIG7L	—	—	—	—	EBTR3 ⁽¹⁾	EBTR2 ⁽¹⁾	EBTR1	EBTR0	---- 1111
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh	DEVID1 ⁽²⁾	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	qqqq qqqq ⁽²⁾
3FFFFFh	DEVID2 ⁽²⁾	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0000 1100

图注：x = 未知，u = 不变，— = 未实现，q = 值取决于具体条件。

阴影单元未实现，读为 0。

注 1：在 PIC18FX3K20 和 PIC18FX4K20 器件上实现了但未使用；保持该位置 1。

2：DEVID1 值请参见寄存器 23-12。DEVID 寄存器为只读寄存器，用户不能对其进行编程。

寄存器 23-1: CONFIG1H: 配置寄存器 1 的高字节

R/P-0	R/P-0	U-0	U-0	R/P-0	R/P-1	R/P-1	R/P-1
IESO	FCMEN	—	—	FOSC3	FOSC2	FOSC1	FOSC0
bit 7							bit 0

图注:

R = 可读位

P = 可编程位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

x = 未知

bit 7 **IESO:** 内部 / 外部振荡器切换位

1 = 使能振荡器切换模式

0 = 禁止振荡器切换模式

bit 6 **FCMEN:** 故障保护时钟监视器使能位

1 = 使能故障保护时钟监视器

0 = 禁止故障保护时钟监视器

bit 5-4 未实现: 读为 0

bit 3-0 **FOSC<3:0>:** 振荡器选择位

11xx = 外部 RC 振荡器, RA6 用作 CLK0 引脚

101x = 外部 RC 振荡器, RA6 用作 CLKOUT 引脚

1001 = 内部振荡器模块, RA6 用作 CLKOUT 引脚, RA7 用作端口引脚

1000 = 内部振荡器模块, RA6 和 RA7 均用作端口引脚

0111 = 外部 RC 振荡器, RA6 用作端口引脚

0110 = 使能 PLL 的 HS 振荡器 (时钟频率 = 4 x FOSC1)

0101 = EC 振荡器, RA6 用作端口引脚

0100 = EC 振荡器, RA6 用作 CLKOUT 引脚

0011 = 外部 RC 振荡器, RA6 用作 CLKOUT 引脚

0010 = HS 振荡器

0001 = XT 振荡器

0000 = LP 振荡器

PIC18F2XK20/4XK20

寄存器 23-2: CONFIG2L: 配置寄存器 2 的低字节

U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	BORV1 ⁽¹⁾	BORV0 ⁽¹⁾	BOREN1 ⁽²⁾	BOREN0 ⁽²⁾	PWRTE ⁽²⁾
bit 7	bit 0						

图注:

R = 可读位

P = 可编程位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

x = 未知

bit 7-5 未实现: 读为 0

bit 4-3 **BORV<1:0>**: 欠压复位电压位⁽¹⁾

11 = VBOR 设置为 1.8V 标称值

10 = VBOR 设置为 2.2V 标称值

01 = VBOR 设置为 2.7V 标称值

00 = VBOR 设置为 3.0V 标称值

bit 2-1 **BOREN<1:0>**: 欠压复位使能位⁽²⁾

11 = 只能由硬件使能欠压复位 (禁止 SBOREN)

10 = 只能由硬件使能欠压复位, 休眠模式下被禁止 (禁止 SBOREN)

01 = 由软件使能和控制欠压复位 (使能 SBOREN)

00 = 用硬件和软件禁止欠压复位

bit 0 **PWRTE**: 上电延时定时器使能位⁽²⁾

1 = 禁止 PWRT

0 = 使能 PWRT

注 1: 请参见第 26.1 节 “直流特性: 供电电压” 获取规范信息。

2: 上电延时定时器与欠压复位是相互独立的, 可以分别控制两者操作。

寄存器 23-3: CONFIG2H: 配置寄存器 2 的高字节

U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN
bit 7	bit 0						

图注:

R = 可读位

P = 可编程位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

x = 未知

bit 7-5 未实现: 读为 0

bit 4-1 **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

bit 0 **WDTEN**: 看门狗定时器使能位

1 = WDT 总是使能的。 SWDTEN 位不起作用。

0 = WDT 由 WDTCON 寄存器的 SWDTEN 位控制

寄存器 23-4: CONFIG3H: 配置寄存器 3 的高字节

R/P-1	U-0	U-0	U-0	R/P-1	R/P-0	R/P-1	R/P-1
MCLRE	—	—	—	HFOFST	LPT1OSC	PBADEN	CCP2MX
bit 7							bit 0

图注:

R = 可读位

P = 可编程位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

X = 未知

bit 7	MCLRE: <u>MCLR</u> 引脚使能位 1 = 使能 MCLR 引脚; 禁止 RE3 输入引脚 0 = 使能 RE3 输入引脚; 禁止 MCLR
bit 6-4	未实现: 读为 0
bit 3	HFOFST: HFINTOSC 快速起振位 1 = HFINTOSC 开始为 CPU 提供时钟而无需等待振荡器稳定下来 0 = 系统时钟关闭直到 HFINTOSC 稳定为止
bit 2	LPT1OSC: 低功耗 Timer1 振荡器使能位 1 = Timer1 配置为低功耗运行 0 = Timer1 配置为高功耗运行
bit 1	PBADEN: PORTB A/D 使能位 (影响 ANSELH 复位状态。ANSELH 控制 PORTB<4:0> 引脚配置。) 1 = PORTB<4:0> 引脚在复位时被配置为模拟输入通道 0 = PORTB<4:0> 引脚在复位时被配置为数字 I/O
bit 0	CCP2MX: CCP2 多路复用位 1 = CCP2 输入 / 输出与 RC1 复用 0 = CCP2 输入 / 输出与 RB3 复用

寄存器 23-5: CONFIG4L: 配置寄存器 4 的低字节

R/P-1	R/P-0	U-0	U-0	U-0	R/P-1	U-0	R/P-1
DEBUG	XINST	—	—	—	LVP ⁽¹⁾	—	STVREN
bit 7							bit 0

图注:

R = 可读位

P = 可编程位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

X = 未知

bit 7	DEBUG: 后台调试器使能位 1 = 禁止后台调试器, RB6 和 RB7 被配置为通用 I/O 引脚 0 = 使能后台调试器, RB6 和 RB7 专用于在线调试
bit 6	XINST: 扩展指令集使能位 1 = 使能指令集扩展和变址寻址模式 0 = 禁止指令集扩展和变址寻址模式 (传统模式)
bit 5-3	未实现: 读为 0
bit 2	LVP: 单电源 ICSP 使能位 1 = 使能单电源 ICSP 0 = 禁止单电源 ICSP
bit 1	未实现: 读为 0
bit 0	STVREN: 堆栈满 / 下溢复位使能位 1 = 堆栈满 / 下溢导致复位 0 = 堆栈满 / 下溢不会导致复位

注 1: 该位只能在高压编程模式下由编程器更改。

PIC18F2XK20/4XK20

寄存器 23-6: CONFIG5L: 配置寄存器 5 的低字节

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	CP3 ⁽¹⁾	CP2 ⁽¹⁾	CP1	CP0
bit 7	bit 0						

图注:

R = 可读位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

C = 只可清零位

bit 7-4	未实现: 读为 0
bit 3	CP3: 代码保护位 ⁽¹⁾ 1 = Block 3 不受代码保护 0 = Block 3 受代码保护
bit 2	CP2: 代码保护位 ⁽¹⁾ 1 = Block 2 不受代码保护 0 = Block 2 受代码保护
bit 1	CP1: 代码保护位 1 = Block 1 不受代码保护 0 = Block 1 受代码保护
bit 0	CP0: 代码保护位 1 = Block 0 不受代码保护 0 = Block 0 受代码保护

注 1: 在 PIC18FX3K20 和 PIC18FX4K20 器件上已实现, 但未使用。

寄存器 23-7: CONFIG5H: 配置寄存器 5 的高字节

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD	CPB	—	—	—	—	—	—
bit 7	bit 0						

图注:

R = 可读位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

C = 只可清零位

bit 7	CPD: 数据 EEPROM 代码保护位 1 = 数据 EEPROM 不受代码保护 0 = 数据 EEPROM 受代码保护
bit 6	CPB: 引导块代码保护位 1 = 引导块不受代码保护 0 = 引导块受代码保护
bit 5-0	未实现: 读为 0

寄存器 23-8: CONFIG6L: 配置寄存器 6 的低字节

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	WRT3 ⁽¹⁾	WRT2 ⁽¹⁾	WRT1	WRT0
bit 7							bit 0

图注:

R = 可读位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

C = 只可清零位

bit 7-4	未实现: 读为 0
bit 3	WRT3: 写保护位 ⁽¹⁾ 1 = Block 3 不受写保护 0 = Block 3 受写保护
bit 2	WRT2: 写保护位 ⁽¹⁾ 1 = Block 2 不受写保护 0 = Block 2 受写保护
bit 1	WRT1: 写保护位 1 = Block 1 不受写保护 0 = Block 1 受写保护
bit 0	WRT0: 写保护位 1 = Block 0 不受写保护 0 = Block 0 受写保护

注 1: 在 PIC18FX3K20 和 PIC18FX4K20 器件上已实现, 但未使用。

寄存器 23-9: CONFIG6H: 配置寄存器 6 的高字节

R/C-1	R/C-1	R-1	U-0	U-0	U-0	U-0	U-0
WRTD	WRTB	WR _{TC} ⁽¹⁾	—	—	—	—	—
bit 7							bit 0

图注:

R = 可读位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

C = 只可清零位

bit 7	WRTD: 数据 EEPROM 写保护位 1 = 数据 EEPROM 不受写保护 0 = 数据 EEPROM 受写保护
bit 6	WRTB: 引导块写保护位 1 = 引导块不受写保护 0 = 引导块受写保护
bit 5	WR_{TC}: 配置寄存器写保护位 ⁽¹⁾ 1 = 配置寄存器不受写保护 0 = 配置寄存器受写保护
bit 4-0	未实现: 读为 0

注 1: 在正常执行模式下, 该位是只读的; 该位仅在编程模式下可写入。

PIC18F2XK20/4XK20

寄存器 23-10: CONFIG7L: 配置寄存器 7 的低字节

U-0	U-0	U-0	U-0	R/C-1	R/C-1	R/C-1	R/C-1
—	—	—	—	EBTR3 ⁽¹⁾	EBTR2 ⁽¹⁾	EBTR1	EBTR0
bit 7	bit 0						

图注:

R = 可读位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

C = 只可清零位

bit 7-4 未实现: 读为 0

bit 3 **EBTR3:** 表读保护位⁽¹⁾

1 = 其他块可对 Block 3 执行表读操作
0 = 禁止其他块对 Block 3 执行表读操作

bit 2 **EBTR2:** 表读保护位⁽¹⁾

1 = 其他块可对 Block 2 执行表读操作
0 = 禁止其他块对 Block 2 执行表读操作

bit 1 **EBTR1:** 表读保护位

1 = 其他块可对 Block 1 执行表读操作
0 = 禁止其他块对 Block 1 执行表读操作

bit 0 **EBTR0:** 表读保护位

1 = 其他块可对 Block 0 执行表读操作
0 = 禁止其他块对 Block 0 执行表读操作

注 1: 在 PIC18FX3K20 和 PIC18FX4K20 器件上已实现, 但未使用。

寄存器 23-11: CONFIG7H: 配置寄存器 7 的高字节

U-0	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
—	EBTRB	—	—	—	—	—	—
bit 7	bit 0						

图注:

R = 可读位

U = 未实现位, 读为 0

-n = 未对器件编程时的值

C = 只可清零位

bit 7 未实现: 读为 0

bit 6 **EBTRB:** 引导块表读保护位

1 = 其他块可对引导块执行表读操作
0 = 禁止其他块对引导块执行表读操作

bit 5-0 未实现: 读为 0

PIC18F2XK20/4XK20

寄存器 23-12: DEVID1: PIC18F2XK20/4XK20 器件的器件 ID 寄存器 1

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							

图注:

R = 可读位

-n = 未对器件编程时的值

U = 未实现位, 读为 0

C = 只可清零位

bit 7-5 **DEV<2:0>**: 器件 ID 位

000 = PIC18F46K20

001 = PIC18F26K20

010 = PIC18F45K20

011 = PIC18F25K20

100 = PIC18F44K20

101 = PIC18F24K20

110 = PIC18F43K20

111 = PIC18F23K20

bit 4-0 **REV<4:0>**: 版本 ID 位

这些位用于表明器件版本。

寄存器 23-13: DEVID2: PIC18F2XK20/4XK20 器件的器件 ID 寄存器 2

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							

图注:

R = 可读位

-n = 未对器件编程时的值

U = 未实现位, 读为 0

C = 只可清零位

bit 7-0 **DEV<10:3>**: 器件 ID 位

这些位与器件 ID 寄存器 1 中的 DEV<2:0> 位一起用于标识器件编号。

0010 0000 = PIC18F2XK20/4XK20 器件

注 1: DEV<10:3> 的值可能会用于其他器件。特定器件总是通过使用整个 DEV<10:0> 位序列来标识的。

PIC18F2XK20/4XK20

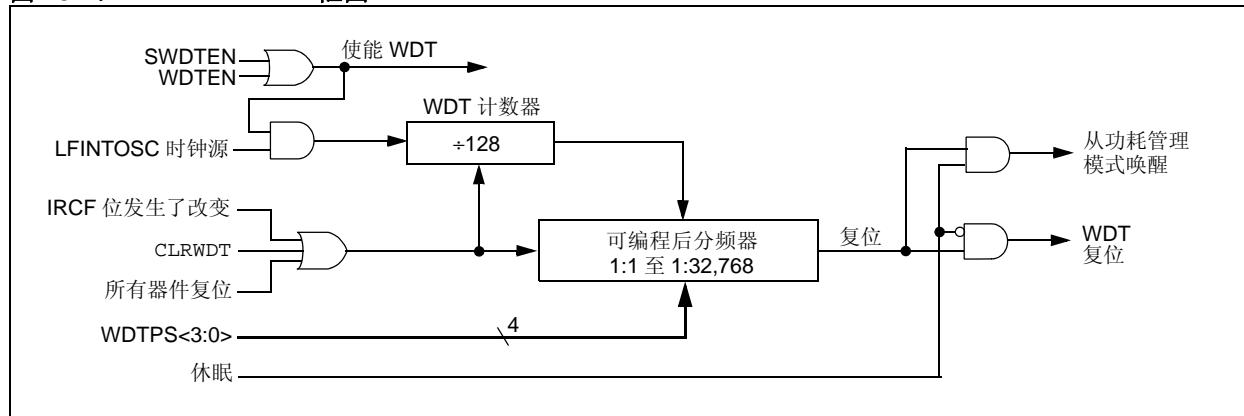
23.2 看门狗定时器 (WDT)

PIC18F2XK20/4XK20 器件的 WDT 是由 LFINTOSC 时钟源驱动的。当使能 WDT 时，时钟源也将同时使能。WDT 定时周期的标称值为 4 ms，其稳定性与 LFINTOSC 振荡器相同。

4 ms 的 WDT 定时周期将与 16 位后分频器的值相乘来得到更长的时间周期。通过配置寄存器 2H 中的位来控制一个多路开关以对 WDT 后分频器的输出进行选择。因此可获得的定时周期范围为 4 ms 至 131.072 秒 (2.18 分钟)。当发生以下任一事件时，WDT 和后分频器将被清零，这些事件包括：执行了 SLEEP 或 CLRWDAT 指令、改变了 OSCCON 寄存器的 IRCF 位或发生了时钟故障。

- 注 1: 当执行 CLRWDAT 和 SLEEP 指令时，WDT 和后分频器的计数值将被清零。
- 2: 更改 OSCCON 寄存器的 IRCF 位的设置会清零 WDT 和后分频器的计数值。
- 3: 当执行 CLRWDAT 指令时，后分频器的计数值将被清零。

图 23-1: WDT 框图



23.2.1 控制寄存器

寄存器 23-14 所示为 WDTCON 寄存器。它是可读写寄存器并包含一个控制位，仅当用配置位禁止 WDT 时，该控制位才允许使用软件改写 WDT 使能配置位。

寄存器 23-14: WDTCON: 看门狗定时器控制寄存器

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

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 7-1 未实现: 读为 0

bit 0 **SWDTEN:** 软件使能或禁止看门狗定时器位⁽¹⁾

1 = WDT 开启

0 = WDT 关闭 (复位值)

注 1: 当使能 WDTEN 配置位时该位不起作用。

表 23-2: 看门狗定时器寄存器汇总

名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	复位值所在页
RCON	IPEN	SBOREN	—	RI	TO	PD	POR	BOR	58
WDTCON	—	—	—	—	—	—	—	SWDTEN	60
CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDTEN	300

图注: — = 未实现, 读为 0。看门狗定时器不使用阴影单元。

PIC18F2XK20/4XK20

23.3 程序校验和代码保护

PIC18闪存器件的整个代码保护结构与以往的PIC®单片机器件完全不同。

用户程序存储器根据具体器件被分为3个或5个存储块。其中一个为0.5 KB或2 KB的引导块，取决于具体器件。存储器的剩余部分按二进制边界被分为单独的存储块。

每个存储块都有与其相关的3个代码保护位。它们是：

- 代码保护位 (CPn)
- 写保护位 (WRTn)
- 外部存储块表读位 (EBTRn)

图23-2给出了8 KB、16 KB和32 KB器件的程序存储器构成以及与每个存储块相关的特定代码保护位。表23-3中总结了这些位的实际地址。

图 23-2： PIC18F2XK20/4XK20 的受代码保护的程序存储器

存储容量 / 器件				存储块的代码保护 受控于：
8 KB (PIC18FX3K20)	16 KB (PIC18FX4K20)	32 KB (PIC18FX5K20)	64 KB (PIC18FX6K20)	
引导块 (000h-1FFh)	引导块 (000h-7FFh)	引导块 (000h-7FFh)	引导块 (000h-7FFh)	CPB, WRTB, EBTRB
Block 0 (200h-FFFh)	Block 0 (800h-1FFFh)	Block 0 (800h-1FFFh)	Block 0 (800h-3FFFh)	CP0, WRT0, EBTR0
Block 1 (1000h-1FFFh)	Block 1 (2000h-3FFFh)	Block 1 (2000h-3FFFh)	Block 1 (4000h-7FFFh)	CP1, WRT1, EBTR1
未实现 读为0 (2000h-1FFFFh)	未实现 读为0 (4000h-1FFFFh)	Block 2 (4000h-5FFFh)	Block 2 (8000h-BFFFh)	CP2, WRT2, EBTR2
		Block 3 (6000h-7FFFh)	Block 3 (C000h-FFFFh)	CP3, WRT3, EBTR3
未实现 读为0 (8000h-1FFFFh)				(未实现的存储空间)

表 23-3： 代码保护寄存器汇总

寄存器名称		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h	CONFIG5L	—	—	—	—	CP3 ⁽¹⁾	CP2 ⁽¹⁾	CP1	CP0
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—
30000Ah	CONFIG6L	—	—	—	—	WRT3 ⁽¹⁾	WRT2 ⁽¹⁾	WRT1	WRT0
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—
30000Ch	CONFIG7L	—	—	—	—	EBTR3 ⁽¹⁾	EBTR2 ⁽¹⁾	EBTR1	EBTR0
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—

图注： 阴影单元未实现。

注 1： 在 PIC18FX3K20 和 PIC18FX4K20 器件上已实现，但未使用。

23.3.1 程序存储器代码保护

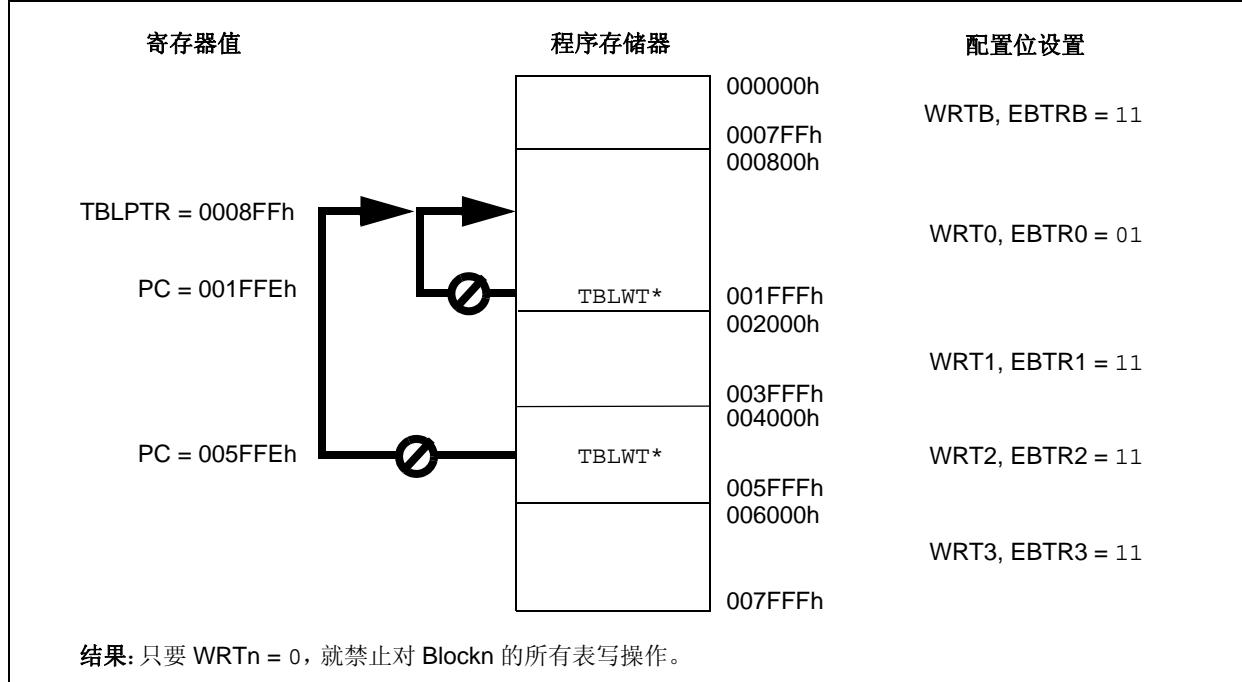
可使用表读和表写指令从任何存储单元读写程序存储器。器件 ID 可以通过表读指令进行读取。配置寄存器可以通过表读和表写指令进行读写操作。

在正常执行模式下，CPn 位不起直接作用。CPn 位禁止外部读写。如果 WRTn 配置位为 0，则用户存储区的存储块可被保护不受表写指令的影响。EBTRn 位控制表读操作。对于 EBTRn 位清零的用户存储区中的存储块，允

许在该存储块内执行表读指令。该存储块以外的存储单元执行的表读指令则不被允许，将导致读为 0。图 23-3 到 23-5 给出了表写和表读保护的图示。

注： 代码保护位只能从 1 改写到 0 状态。不可能将处于 0 状态的位改写为 1。只有通过整片擦除或块擦除功能才能将代码保护位设为 1。整片擦除和块擦除功能只能通过 ICSP 或外部编程器启动。

图 23-3：不允许表写（WRTn）



PIC18F2XK20/4XK20

图 23-4: 不允许外部存储块的表读 (EBTRn)

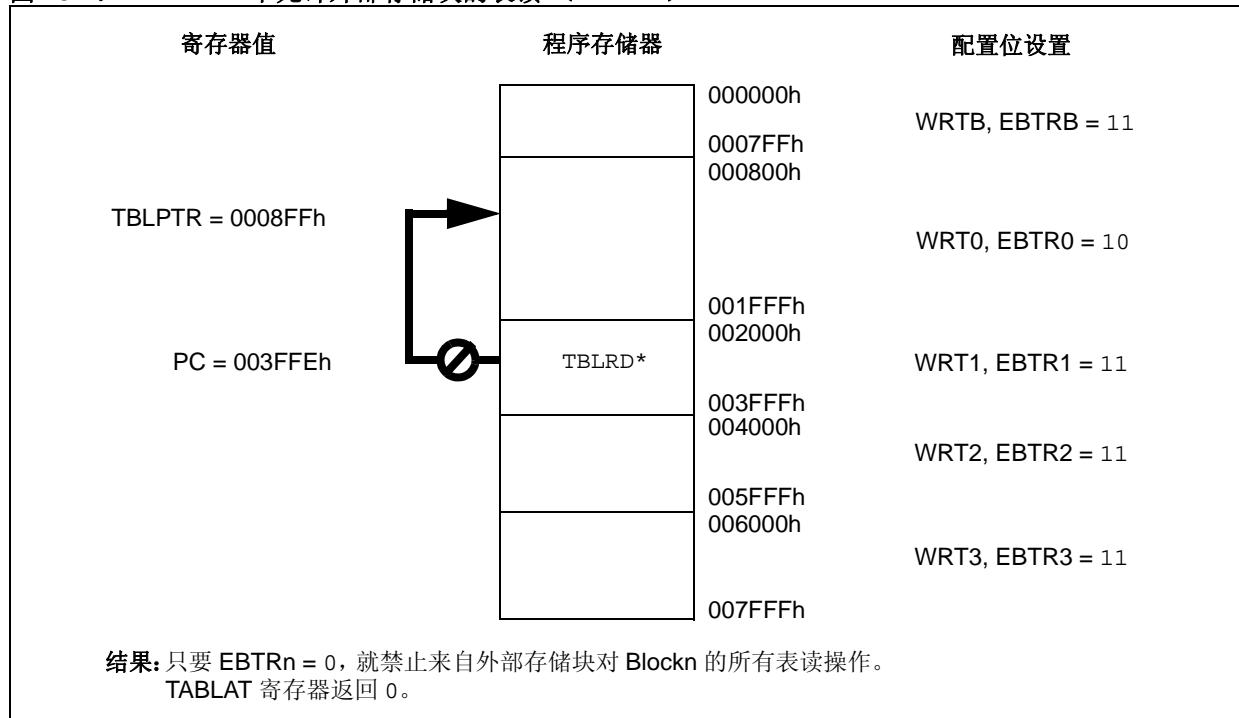
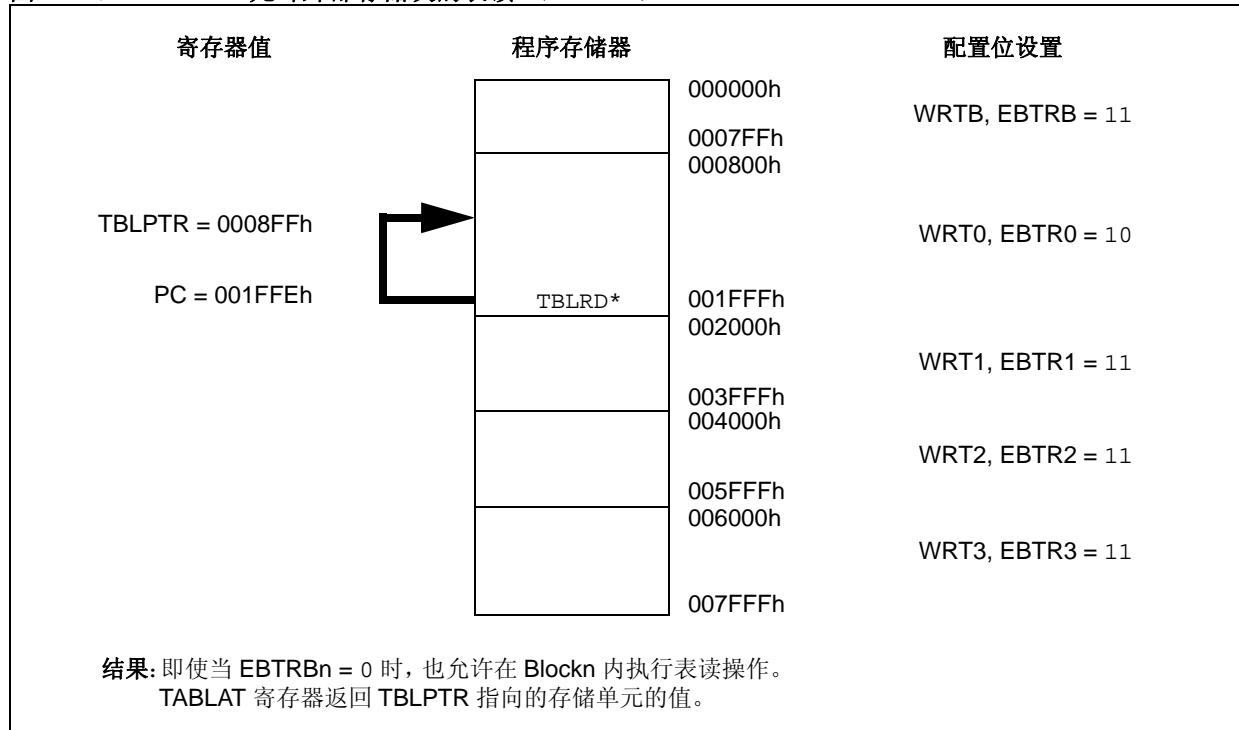


图 23-5: 允许外部存储块的表读 (EBTRn)



23.3.2 数据 EEPROM 代码保护

CPD 和 WRTD 两个位可保护整个数据 EEPROM 不被从外部读写。CPD 禁止数据 EEPROM 的外部读写。WRTD 禁止从内部和外部写数据 EEPROM。在正常操作下，CPU 可以始终读数据 EEPROM，与保护位的设置无关。

23.3.3 配置寄存器保护

配置寄存器可以是写保护的。WRTC 位控制配置寄存器的保护。在正常执行模式下，WRTC 位是只读的。WRTC 只能通过 ICSP 或外部编程器写入。

23.4 ID 存储单元

有 8 个存储单元（200000h-200007h）被指定为 ID 单元，供用户存储校验和或其他代码标识编号。在执行程序时可通过 TBLRD 和 TBLWT 指令读写这些单元；在编程 / 验证时，也可读写这些单元。当器件有代码保护时，也可读取 ID 单元。

23.5 在线串行编程

PIC18F2XK20/4XK20 器件可以在最终的应用电路中进行串行编程。只需要 5 根线即可实现这一操作，其中时钟线、数据线各一根，其余 3 根分别是电源线、接地线和编程电压线。这允许用户在生产电路板时使用未编程器件，仅在产品交付之前才对单片机进行编程，从而可以使用最新版本的固件或者定制固件进行编程。

23.6 在线调试器

将 DEBUG 配置位编程为 0，可使能在线调试功能。该功能允许与 MPLAB® IDE 配合使用进行简单的调试。当使能了单片机的这项功能时，有些资源就不再是通用的了。表 23-4 给出了后台调试器所需的资源。

表 23-4： 调试器资源

I/O 引脚：	RB6 和 RB7
堆栈：	2 级深度
程序存储器：	512 字节
数据存储器：	10 字节

要使用单片机的在线调试功能，设计必须实现以下引脚的在线串行编程连接：

- MCLR/VPP/RE3
- VDD
- VSS
- RB7
- RB6

从而为 Microchip 或第三方开发工具公司提供的在线调试器模块提供交互的接口。

23.7 单电源 ICSP 编程

LVP 配置位使能单电源 ICSP 编程（原来称为低电压 ICSP 编程或 LVP）。当使能单电源编程时，单片机可以在无需对 MCLR/VPP/RE3 引脚施加高电压的情况下进行编程，但 RB5/KB1/PGM 引脚专用于控制编程模式选择，不能再用作通用 I/O 引脚。

使用单电源编程时，VDD 被施加到 MCLR/VPP/RE3 引脚，如同在正常执行模式下一样。要进入编程模式，VDD 被施加到 PGM 引脚。

- 注**
- 1:** 通过将 VIHH 施加到 MCLR 引脚，就可以进行高电压编程，与 LVP 位或 PGM 引脚的状态无关。
 - 2:** 默认情况下，使能对未编程器件（如 Microchip 提供的）和已擦除器件进行单电源 ICSP 编程。
 - 3:** 当使能单电源编程时，RB5 引脚不能再用作通用 I/O 引脚。
 - 4:** 当使能 LVP 时，在外部将 PGM 引脚拉至 Vss 以允许正常程序执行。

如果不再使用单电源 ICSP 编程模式，则 LVP 位可以被清零。RB5/KB1/PGM 随后可作为数字 I/O 引脚 RB5。LVP 位仅可在使用标准高电压编程时被置 1 或清零（VIHH 被施加到 MCLR/VPP/RE3 引脚）。一旦 LVP 被禁止，只能使用标准高电压编程来对器件进行编程。

不受代码保护的存储器可以使用块擦除或逐行擦除进行擦除，然后在任何指定的 VDD 下进行写入。如果要擦除受代码保护的存储器，需要进行块擦除。

PIC18F2XK20/4XK20

注:

24.0 指令集汇总

PIC18F2XK20/4XK20 器件具有一个包含 75 条 PIC18 核心指令的标准指令集，和一个包含 8 条新指令的扩展指令集，扩展指令集用于优化递归代码或使用软件堆栈的代码。本章后面的部分将讨论扩展指令集。

24.1 标准指令集

标准的 PIC18 指令集与以前的 PIC® MCU 指令集相比，添加了很多增强功能，并保持了易于从这些 PIC® MCU 指令集移植的特点。大部分指令为单程序存储字指令（16 位），只有 4 条指令需要两个程序存储单元。

每个单字指令都是一个 16 位字，由操作码（指明指令类型）和一个或多个操作数（指定指令操作）组成。

整个指令集具有高度的正交性，可以分为以下 4 种基本类型：

- **字节**操作类指令
- **位**操作类指令
- **立即数**操作类指令
- **控制**操作类指令

表 24-2 为 PIC18 指令集汇总，列出了上述四类指令。表 24-1 给出了操作码字段的说明。

大部分**字节**操作类的指令都含有三种操作数：

1. 文件寄存器（由 “f” 指定）
2. 保存结果的目标寄存器（由 “d” 指定）
3. 被访问存储区（由 “a” 指定）

文件寄存器标识符 “f” 指定了指令将会使用哪一个文件寄存器。目标寄存器标识符 “d” 指定了操作结果的存放位置。如果 “d” 为 0，操作结果存入 WREG 寄存器中。如果 “d” 为 1，操作结果存入指令指定的文件寄存器中。

所有**位**操作类指令都含有三种操作数：

1. 文件寄存器（由 “f” 指定）
2. 文件寄存器中的位（由 “b” 指定）
3. 被访问存储区（由 “a” 指定）

位域标识符 “b” 选择操作所影响的位的编号，而文件寄存器标识符 “f” 则代表这些位所在的文件寄存器地址。

立即数操作类指令使用以下操作数：

- 要装入到文件寄存器中的立即数（由 “k” 指定）
- 要装入立即数的 FSR 寄存器（由 “f” 指定）
- 不需要操作数（由 “—” 指定）

控制类指令可以使用以下操作数：

- 程序存储器地址（由 “n” 指定）
- CALL 或 RETURN 指令的模式（由 “s” 指定）
- 表读和表写指令的模式（由 “m” 指定）
- 不需要操作数（由 “—” 指定）

除了 4 条双字指令外，其他所有的指令都是单字指令。双字指令将所需的信息保存在 32 位中。第二个字的高 4 位都是 1。如果第二个字作为一条指令执行，它会执行行为 NOP 指令。

除非条件测试结果为真或者指令执行改变了程序计数器的值，否则执行所有的单字指令都只需要一个指令周期。对于上述两种特殊情况，指令执行需要两个指令周期，在第二个指令周期中执行一条 NOP 指令。

执行双字指令需要两个指令周期。

每个指令周期由 4 个振荡周期组成。因此，如果振荡器频率为 4 MHz，正常的指令执行时间为 1 μs。如果条件测试结果为真或者指令执行改变了程序计数器的值，则该指令的执行时间为 2 μs。双字跳转指令（如果为真）的执行则需要 3 μs。

图 24-1 给出了指令的几种通用格式。所有示例均使用 “nnh” 来表示十六进制数。

指令集汇总（见表 24-2）列出了可被 Microchip MPASM™ 汇编器识别的标准指令。

第 24.1.1 节“**标准指令集**”中对每条指令进行了介绍。

PIC18F2XK20/4XK20

表 24-1：操作码字段说明

字段	说明
a	快速操作 RAM 位 a = 0: 快速操作 RAM 内的 RAM 存储单元 (BSR 寄存器被忽略) a = 1: 由 BSR 寄存器指定 RAM 存储区
bbb	8 位文件寄存器内的位地址 (0 到 7)。
BSR	存储区选择寄存器。用于选择当前的 RAM 存储区。
C、DC、Z、OV 和 N	ALU 状态位: 进位、半进位、全零、溢出和负标志位。
d	目标寄存器选择位 d = 0: 结果保存至 WREG 寄存器 d = 1: 结果保存至文件寄存器 f
dest	目标寄存器: 可以是 WREG 寄存器或指定的文件寄存器地址。
f	8 位文件寄存器地址 (00h 到 FFh)，或 2 位 FSR 标识符 (0h 到 3h)。
f _s	12 位文件寄存器地址 (000h 到 FFFh)。这是源地址。
f _d	12 位文件寄存器地址 (000h 到 FFFh)。这是目标地址。
GIE	全局中断允许位。
k	立即数、常数或者标号 (可能是 8 位、12 位或 20 位的值)。
label	标号名称。
mm	表读和表写指令的 TBLPTR 寄存器模式。 只与表读和表写指令一起使用: * 不改变寄存器 (如用于表读和表写的 TBLPTR) *+ 后递增寄存器 (如用于表读和表写的 TBLPTR) *- 后递减寄存器 (如用于表读和表写的 TBLPTR) +* 预递增寄存器 (如用于表读和表写的 TBLPTR)
n	相对跳转指令的相对地址 (二进制补码形式)，或 CALL/ 跳转和 RETURN 指令的直接地址。
PC	程序计数器。
PCL	程序计数器低字节。
PCH	程序计数器高字节。
PCLATH	程序计数器高字节锁存器。
PCLATU	程序计数器最高字节锁存器。
PD	掉电位。
PRODH	乘积的高字节。
PRODL	乘积的低字节。
s	快速调用 / 返回模式选择位 s = 0: 不对影子寄存器进行更新，也不用影子寄存器的内容更新其他寄存器 s = 1: 将寄存器的值装入影子寄存器或把影子寄存器中的值装入寄存器 (快速模式)
TBLPTR	21 位表指针 (指向程序存储器地址)。
TABLAT	8 位表锁存器。
TO	超时位。
TOS	栈顶。
u	未使用或未改变。
WDT	看门狗定时器。
WREG	工作寄存器 (累加器)。
x	无关位 (0 或 1)。汇编器将生成 x = 0 的代码。为了与所有的 Microchip 软件工具兼容，建议使用这种形式。
z _s	对寄存器 (源) 进行间接寻址的 7 位偏移量。
z _d	对寄存器 (目标) 进行间接寻址的 7 位偏移量。
{ }	可选参数。
[text]	表示变址地址。
(text)	text 的内容。
[expr]<n>	表示由指针 expr 指向的寄存器中的 bit n。
→	赋值。
< >	寄存器位域。
ε	表示属于某个集合。
斜体文字	用户定义项 (字体为 Courier)。

图 24-1: 指令的通用格式

针对字节的文件寄存器操作指令	指令示例																
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>15</td><td>10</td><td>9</td><td>8</td><td>7</td><td>0</td></tr> <tr> <td colspan="2">操作码</td><td>d</td><td>a</td><td colspan="2">f(寄存器地址)</td></tr> </table> <p> d = 0, 表示结果存入 WREG 寄存器 d = 1, 表示结果存入文件寄存器 (f) a = 0, 强制使用快速操作存储区 a = 1, 根据 BSR 选择存储区 f = 8 位文件寄存器地址 </p>	15	10	9	8	7	0	操作码		d	a	f(寄存器地址)		ADDWF MYREG, W, B				
15	10	9	8	7	0												
操作码		d	a	f(寄存器地址)													
字节到字节的传送操作 (双字) 指令																	
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>15</td><td>12</td><td>11</td><td>0</td></tr> <tr> <td colspan="3">操作码</td><td>f(源寄存器地址)</td></tr> <tr> <td>15</td><td>12</td><td>11</td><td>0</td></tr> <tr> <td colspan="3">1111</td><td>f(目标寄存器地址)</td></tr> </table> <p>f = 12 位文件寄存器地址</p>	15	12	11	0	操作码			f(源寄存器地址)	15	12	11	0	1111			f(目标寄存器地址)	MOVFF MYREG1, MYREG2
15	12	11	0														
操作码			f(源寄存器地址)														
15	12	11	0														
1111			f(目标寄存器地址)														
针对位的文件寄存器操作指令																	
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>15</td><td>12</td><td>11</td><td>9</td><td>8</td><td>7</td><td>0</td></tr> <tr> <td colspan="2">操作码</td><td>b(位号)</td><td>a</td><td colspan="3">f(寄存器地址)</td></tr> </table> <p> b = 占 3 位, 表示文件寄存器 (f) 中位的位置 a = 0, 强制使用快速操作存储区 a = 1, 根据 BSR 选择存储区 f = 8 位文件寄存器地址 </p>	15	12	11	9	8	7	0	操作码		b(位号)	a	f(寄存器地址)			BSF MYREG, bit, B		
15	12	11	9	8	7	0											
操作码		b(位号)	a	f(寄存器地址)													
立即数操作指令																	
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>15</td><td>8</td><td>7</td><td>0</td></tr> <tr> <td colspan="3">操作码</td><td>k(立即数)</td></tr> </table> <p>k = 8 位立即数的值</p>	15	8	7	0	操作码			k(立即数)	MOVLW 7Fh								
15	8	7	0														
操作码			k(立即数)														
控制操作指令																	
CALL、GOTO 和跳转操作类指令																	
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>15</td><td>8</td><td>7</td><td>0</td></tr> <tr> <td colspan="3">操作码</td><td>n<7:0> (立即数)</td></tr> <tr> <td>15</td><td>12</td><td>11</td><td>0</td></tr> <tr> <td colspan="3">1111</td><td>n<19:8> (立即数)</td></tr> </table> <p>n = 20 位立即数的值</p>	15	8	7	0	操作码			n<7:0> (立即数)	15	12	11	0	1111			n<19:8> (立即数)	GOTO Label
15	8	7	0														
操作码			n<7:0> (立即数)														
15	12	11	0														
1111			n<19:8> (立即数)														
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>15</td><td>8</td><td>7</td><td>0</td></tr> <tr> <td colspan="2">操作码</td><td>S</td><td>n<7:0> (立即数)</td></tr> <tr> <td>15</td><td>12</td><td>11</td><td>0</td></tr> <tr> <td colspan="3">1111</td><td>n<19:8> (立即数)</td></tr> </table> <p>S = 快速位</p>	15	8	7	0	操作码		S	n<7:0> (立即数)	15	12	11	0	1111			n<19:8> (立即数)	CALL MYFUNC
15	8	7	0														
操作码		S	n<7:0> (立即数)														
15	12	11	0														
1111			n<19:8> (立即数)														
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>15</td><td>11</td><td>10</td><td>0</td></tr> <tr> <td colspan="3">操作码</td><td>n<10:0> (立即数)</td></tr> </table>	15	11	10	0	操作码			n<10:0> (立即数)	BRA MYFUNC								
15	11	10	0														
操作码			n<10:0> (立即数)														
<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>15</td><td>8</td><td>7</td><td>0</td></tr> <tr> <td colspan="3">操作码</td><td>n<7:0> (立即数)</td></tr> </table>	15	8	7	0	操作码			n<7:0> (立即数)	BC MYFUNC								
15	8	7	0														
操作码			n<7:0> (立即数)														

表 24-2: PIC18FXXXX 指令集

助记符, 操作数	说明	周期数	16 位指令字				受影响的 状态位	注	
			MSb	LSb					
针对字节的操作类指令									
ADDWF	f, d, a	WREG 与 f 相加	1	0010	01da	ffff	ffff	C, DC, Z, OV, N 1, 2	
ADDWFC	f, d, a	WREG 与 f 带进位相加	1	0010	00da	ffff	ffff	C, DC, Z, OV, N 1, 2	
ANDWF	f, d, a	WREG 和 f 作逻辑与运算	1	0001	01da	ffff	ffff	Z, N 1,2	
CLRF	f, a	将 f 清零	1	0110	101a	ffff	ffff	Z 2	
COMF	f, d, a	对 f 取反	1	0001	11da	ffff	ffff	Z, N 1, 2	
CPFSEQ	f, a	将 f 与 WREG 作比较, 相等则跳过	1 (2 或 3)	0110	001a	ffff	ffff	无 4	
CPFSGT	f, a	将 f 与 WREG 作比较, 大于则跳过	1 (2 或 3)	0110	010a	ffff	ffff	无 4	
CPFSLT	f, a	将 f 与 WREG 作比较, 小于则跳过	1 (2 或 3)	0110	000a	ffff	ffff	无 1, 2	
DECFSZ	f, d, a	f 递减 1	1	0000	01da	ffff	ffff	C, DC, Z, OV, N 1, 2, 3, 4	
DECFSZ	f, d, a	f 递减 1, 为 0 则跳过	1 (2 或 3)	0010	11da	ffff	ffff	无 1, 2, 3, 4	
DCFSNZ	f, d, a	f 递减 1, 非 0 则跳过	1 (2 或 3)	0100	11da	ffff	ffff	无 1, 2	
INCFSZ	f, d, a	f 递增 1	1	0010	10da	ffff	ffff	C, DC, Z, OV, N 1, 2, 3, 4	
INCFSZ	f, d, a	f 递增 1, 为 0 则跳过	1 (2 或 3)	0011	11da	ffff	ffff	无 4	
INFSNZ	f, d, a	f 递增 1, 非 0 则跳过	1 (2 或 3)	0100	10da	ffff	ffff	无 1, 2	
IORWF	f, d, a	WREG 和 f 作逻辑或运算	1	0001	00da	ffff	ffff	Z, N 1, 2	
MOVF	f, d, a	传送 f	1	0101	00da	ffff	ffff	Z, N 1	
MOVFF	f _s , f _d	从 f _s (源) 送到 第一个字 f _d (目标) 第二个字	2	1100	ffff	ffff	ffff	无	
				1111	ffff	ffff	ffff		
MOVWF	f, a	将 WREG 内容传送到 f	1	0110	111a	ffff	ffff	无	
MULWF	f, a	WREG 与 f 相乘	1	0000	001a	ffff	ffff	无 1, 2	
NEGF	f, a	对 f 取补	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	
RLCF	f, d, a	f 带进位循环左移	1	0011	01da	ffff	ffff	C, Z, N 1, 2	
RLNCF	f, d, a	f 循环左移 (不带进位)	1	0100	01da	ffff	ffff	Z, N	
RRCF	f, d, a	f 带进位循环右移	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	f 循环右移 (不带进位)	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	将 f 的内容置为全 1	1	0110	100a	ffff	ffff	无 1, 2	
SUBFWB	f, d, a	WREG 减去 f (带借位)	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	f 减去 WREG	1	0101	11da	ffff	ffff	C, DC, Z, OV, N 1, 2	
SUBWFB	f, d, a	f 减去 WREG (带借位)	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	将 f 中的两个半字节进行交换	1	0011	10da	ffff	ffff	无 4	
TSTFSZ	f, a	测试 f, 为 0 则跳过	1 (2 或 3)	0110	011a	ffff	ffff	无 1, 2	
XORWF	f, d, a	WREG 和 f 作逻辑异或运算	1	0001	10da	ffff	ffff	Z, N	

注 1: 当端口寄存器修改自身时 (例如, MOVF PORTB, 1, 0), 修改时使用的值是引脚上的当前值。例如, 如果将一引脚配置为输入, 其对应数据锁存器中的值将为 1, 但此时若有外部器件将该引脚驱动为低电平, 则被写回数据锁存器的数据值将是 0。

2: 当对 TMRO 寄存器执行该指令 (并且 d = 1) 时, 如果已为其分配了预分频器, 则将该预分频器清零。

3: 如果程序计数器 (PC) 被修改或者条件测试为真, 则该指令需要两个周期。第二个周期执行一条 NOP 指令。

4: 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息, 否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

表 24-2: PIC18FXXXX 指令集 (续)

助记符, 操作数	说明	周期数	16 位指令字				受影响的 状态位	注
			MSb		Lsb			
针对位的操作类指令								
BCF	f, b, a	将 f 中的某位清零	1	1001	bbbb	ffff	ffff	无
BSF	f, b, a	将 f 中的某位置 1	1	1000	bbbb	ffff	ffff	无
BTFS	f, b, a	测试 f 中的某位, 为 0 则跳过	1 (2 或 3)	1011	bbbb	ffff	ffff	无
BTFS	f, b, a	测试 f 中的某位, 为 1 则跳过	1 (2 或 3)	1010	bbbb	ffff	ffff	无
BTG	f, d, a	将 f 中的某位取反	1	0111	bbbb	ffff	ffff	无
控制类指令								
BC	n	进位则跳转	1 (2)	1110	0010	nnnn	nnnn	无
BN	n	为负则跳转	1 (2)	1110	0110	nnnn	nnnn	无
BNC	n	无进位则跳转	1 (2)	1110	0011	nnnn	nnnn	无
BNN	n	不为负则跳转	1 (2)	1110	0111	nnnn	nnnn	无
BNOV	n	不溢出则跳转	1 (2)	1110	0101	nnnn	nnnn	无
BNZ	n	不为零则跳转	1 (2)	1110	0001	nnnn	nnnn	无
BOV	n	溢出则跳转	1 (2)	1110	0100	nnnn	nnnn	无
BRA	n	无条件跳转	2	1101	0nnn	nnnn	nnnn	无
BZ	n	为零则跳转	1 (2)	1110	0000	nnnn	nnnn	无
CALL	n, s	调用子程序	第一个字	2	1110	110s	kkkk	kkkk
			第二个字		1111	kkkk	kkkk	无
CLRWDT	—	将看门狗定时器清零	1	0000	0000	0000	0100	TO, PD
DAW	—	对 WREG 进行十进制调整	1	0000	0000	0000	0111	C
GOTO	n	跳转到地址	第一个字	2	1110	1111	kkkk	无
			第二个字		1111	kkkk	kkkk	
NOP	—	空操作	1	0000	0000	0000	0000	无
NOP	—	空操作	1	1111	xxxx	xxxx	xxxx	无
POP	—	弹出返回堆栈栈顶 (TOS) 的内容	1	0000	0000	0000	0110	无
PUSH	—	将数据压入返回堆栈栈顶 (TOS)	1	0000	0000	0000	0101	无
RCALL	n	相对调用	2	1101	1nnn	nnnn	nnnn	无
RESET		软件器件复位	1	0000	0000	1111	1111	全部
RETFIE	s	中断返回允许	2	0000	0000	0001	000s	GIE/GIEH, PEIE/GIEL
RETLW	k	返回时将立即数送入 WREG	2	0000	1100	kkkk	kkkk	无
RETURN	s	从子程序返回	2	0000	0000	0001	001s	无
SLEEP	—	进入待机模式	1	0000	0000	0000	0011	TO, PD

注 1: 当端口寄存器修改自身时 (例如, MOVF PORTB, 1, 0), 修改时使用的值是引脚上的当前值。例如, 如果将一引脚配置为输入, 其对应数据锁存器中的值将为 1, 但此时若有外部器件将该引脚驱动为低电平, 则被写回数据锁存器的数据值将是 0。

2: 当对 TMRO 寄存器执行该指令 (并且 d = 1) 时, 如果已为其分配了预分频器, 则将该预分频器清零。

3: 如果程序计数器 (PC) 被修改或者条件测试为真, 则该指令需要两个周期。第二个周期执行一条 NOP 指令。

4: 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息, 否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

PIC18F2XK20/4XK20

表 24-2: PIC18FXXXX 指令集 (续)

助记符, 操作数	说明	周期数	16 位指令字				受影响的 状态位	注
			MSb		Lsb			
立即数操作类指令								
ADDLW k	WREG 与立即数相加	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW k	WREG 和立即数进行逻辑与运算	1	0000	1011	kkkk	kkkk	Z, N	
IORLW k	WREG 和立即数进行逻辑或运算	1	0000	1001	kkkk	kkkk	Z, N	
LFSR f, k	传送立即数 (12 位) 第二个字到 FSR (f) 第一个字	2	1110	1110	00ff	kkkk	无	
MOVLB k	将立即数送入 BSR<3:0>	1	0000	0001	0000	kkkk	无	
MOVLW k	将立即数送入 WREG	1	0000	1110	kkkk	kkkk	无	
MULLW k	WREG 和立即数相乘	1	0000	1101	kkkk	kkkk	无	
RETLW k	返回时将立即数送入 WREG	2	0000	1100	kkkk	kkkk	无	
SUBLW k	立即数减去 WREG	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW k	WREG 和立即数进行逻辑异或运算	1	0000	1010	kkkk	kkkk	Z, N	
数据存储器 ↔ 程序存储器操作								
TBLRD*	表读	2	0000	0000	0000	1000	无	
TBLRD*+	后递增表读		0000	0000	0000	1001	无	
TBLRD*-	后递减表读		0000	0000	0000	1010	无	
TBLRD+*	预递增表读		0000	0000	0000	1011	无	
TBLWT*	表写	2	0000	0000	0000	1100	无	
TBLWT*+	后递增表写		0000	0000	0000	1101	无	
TBLWT*-	后递减表写		0000	0000	0000	1110	无	
TBLWT+*	预递增表写		0000	0000	0000	1111	无	

注 1: 当端口寄存器修改自身时 (例如, MOVF PORTB, 1, 0), 修改时使用的值是引脚上的当前值。例如, 如果将一引脚配置为输入, 其对应数据锁存器中的值将为 1, 但此时若有外部器件将该引脚驱动为低电平, 则被写回数据锁存器的数据值将是 0。

- 2: 当对 TMRO 寄存器执行该指令 (并且 d = 1) 时, 如果已为其分配了预分频器, 则将该预分频器清零。
- 3: 如果程序计数器 (PC) 被修改或者条件测试为真, 则该指令需要两个周期。第二个周期执行一条 NOP 指令。
- 4: 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息, 否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

24.1.1 标准指令集

ADDLW	W 与立即数相加			
语法:	ADDLW k			
操作数:	$0 \leq k \leq 255$			
操作:	$(W) + k \rightarrow W$			
受影响的状态位:	N、OV、C、DC 和 Z			
机器码:	0000	1111	kkkk	kkkk
说明:	将 W 寄存器的内容与 8 位立即数 k 相加，结果存储在 W 寄存器中。			
指令字数:	1			
指令周期数:	1			
Q 周期操作:	Q1	Q2	Q3	Q4
	译码	读立即数 k	处理数据	写入 W

示例: ADDLW 15h

执行指令前
W = 10h
执行指令后
W = 25h

ADDWF	W 与 f 相加			
语法:	ADDWF f {,d {,a}}			
操作数:	$0 \leq f \leq 255$			
	$d \in [0,1]$			
	$a \in [0,1]$			
操作:	$(W) + (f) \rightarrow dest$			
受影响的状态位:	N、OV、C、DC 和 Z			
机器码:	0010	01da	ffff	ffff
说明:	将 W 的内容与 f 寄存器的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。			
指令字数:	1			
指令周期数:	1			
Q 周期操作:	Q1	Q2	Q3	Q4
	译码	读寄存器 f	处理数据	写入目标寄存器

示例: ADDWF REG, 0, 0

执行指令前
W = 17h
REG = 0C2h
执行指令后
W = 0D9h
REG = 0C2h

注: 所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数，用于符号寻址。如果使用了标号，那么指令格式将变为: {label} 指令参数。

PIC18F2XK20/4XK20

ADDWFC

W 与 f 带进位相加

语法:	ADDWFC f {,d {,a}}								
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$								
操作:	$(W) + (f) + (C) \rightarrow dest$								
受影响的状态位:	N、 OV、 C、 DC 和 Z								
机器码:	0010 00da ffff ffff								
说明:	将 W 的内容、进位标志位与数据存储单元 f 的内容相加。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存储在数据存储单元 f 中。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。								
指令字数:	1								
指令周期数:	1								
Q 周期操作:	<table border="1"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>译码</td> <td>读寄存器 f</td> <td>处理数据</td> <td>写入目标寄存器</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	写入目标寄存器
Q1	Q2	Q3	Q4						
译码	读寄存器 f	处理数据	写入目标寄存器						

示例: ADDWFC REG, 0, 1

执行指令前

进位标志位	= 1
REG	= 02h
W	= 4Dh

执行指令后

进位标志位	= 0
REG	= 02h
W	= 50h

ANDLW

立即数和 W 寄存器作逻辑与运算

语法:	ANDLW k								
操作数:	$0 \leq k \leq 255$								
操作:	$(W) .AND. k \rightarrow W$								
受影响的状态位:	N 和 Z								
机器码:	0000 1011 kkkk kkkk								
说明:	将 W 的内容与 8 位立即数 k 进行逻辑与运算。结果存储在 W 寄存器中。								
指令字数:	1								
指令周期数:	1								
Q 周期操作:	<table border="1"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>译码</td> <td>读立即数 k</td> <td>处理数据</td> <td>写入 W</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读立即数 k	处理数据	写入 W
Q1	Q2	Q3	Q4						
译码	读立即数 k	处理数据	写入 W						

示例: ANDLW 05Fh

执行指令前	
W	= A3h
执行指令后	
W	= 03h

ANDWF

将 W 和 f 作逻辑与运算

语法:	ANDWF f {,d {,a}}		
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$		
操作:	$(W) .AND.(f) \rightarrow dest$		
受影响的状态位:	N 和 Z		
机器码:	0001 01da ffff ffff		
说明:	将 W 的内容与寄存器 f 的内容进行逻辑与运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f(默认)。如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例:

执行指令前	ANDWF	REG, 0, 0
W	= 17h	
REG	= C2h	
执行指令后		
W	= 02h	
REG	= C2h	

BC

进位则跳转

语法:	BC n		
操作数:	$-128 \leq n \leq 127$		
操作:	如果进位标志位为 1 $(PC) + 2 + 2n \rightarrow PC$		
受影响的状态位:	无		
机器码:	1110 0010 nnnn nnnn		
说明:	如果进位标志位为 1，程序将跳转。 “ $2n$ ”(以二进制补码表示)与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 $PC + 2 + 2n$ 。该指令为一条双周期指令。		
指令字数:	1		
指令周期数:	1 (2)		
Q 周期操作:			
如果跳转:			
Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作
如果不跳转:			
Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例:

执行指令前	HERE	BC 5
PC	= 地址 (HERE)	
执行指令后		
如果进位标志位 = 1;	PC = 地址 (HERE + 12)	
如果进位标志位 = 0;	PC = 地址 (HERE + 2)	

PIC18F2XK20/4XK20

BCF

将 f 中的某位清零

语法:	BCF f, b {,a}		
操作数:	$0 \leq f \leq 255$ $0 \leq b \leq 7$ $a \in [0,1]$		
操作:	$0 \rightarrow f$		
受影响的状态位:	无		
机器码:	1001 bbba ffff ffff		
说明:	将寄存器 f 中的位 b 清零。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例: BCF FLAG_REG, 7, 0

执行指令前
FLAG_REG = C7h
执行指令后
FLAG_REG = 47h

BN

为负则跳转

语法:	BN n		
操作数:	$-128 \leq n \leq 127$		
操作:	如果负标志位为 1 $(PC) + 2 + 2n \rightarrow PC$		
受影响的状态位:	无		
机器码:	1110 0110 nnnn nnnn		
说明:	如果负标志位为 1, 程序将跳转。 “2n”(以二进制补码表示)与 PC 相加。 由于 PC 将递增以便取出下一条指令, 所以新地址将为 $PC + 2 + 2n$ 。该指令为一条双周期指令。		
指令字数:	1		
指令周期数:	1 (2)		
Q 周期操作:			
如果跳转:			
Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	写入 PC
空操作	空操作	空操作	空操作
如果不跳转:			
Q1	Q2	Q3	Q4
译码	读立即数 n	处理数据	空操作

示例: HERE BN Jump

执行指令前
PC = 地址 (HERE)
执行指令后
如果负标志位 = 1:
PC = 地址 (Jump)
如果负标志位 = 0:
PC = 地址 (HERE + 2)

BNC

无进位则跳转

语法:	BNC n			
操作数:	$-128 \leq n \leq 127$			
操作:	如果进位标志位为 0 $(PC) + 2 + 2n \rightarrow PC$			
受影响的状态位:	无			
机器码:	1110	0011	nnnn	nnnn
说明:	如果进位标志位为 0, 程序将跳转。 “2n”(以二进制补码表示)与 PC 相加。由于 PC 将递增以便取出下一条指令, 所以新地址将为 $PC + 2 + 2n$ 。该指令为一条双周期指令。			
指令字数:	1			
指令周期数:	1 (2)			
Q 周期操作:				
如果跳转:				
Q1	Q2	Q3	Q4	
译码	读立即数 n	处理数据	写入 PC	
空操作	空操作	空操作	空操作	
如果不跳转:				
Q1	Q2	Q3	Q4	
译码	读立即数 n	处理数据	空操作	

示例: HERE BNC Jump

执行指令前

PC = 地址 (HERE)

执行指令后

如果进位标志位 = 0;

PC = 地址 (Jump)

如果进位标志位 = 1;

PC = 地址 (HERE + 2)

BNN

不为负则跳转

语法:	BNN n			
操作数:	$-128 \leq n \leq 127$			
操作:	如果负标志位为 0 $(PC) + 2 + 2n \rightarrow PC$			
受影响的状态位:	无			
机器码:	1110	0111	nnnn	nnnn
说明:	如果负标志位为 0, 程序将跳转。 “2n”(以二进制补码表示)与 PC 相加。由于 PC 将递增以便取出下一条指令, 所以新地址将为 $PC + 2 + 2n$ 。该指令为一条双周期指令。			
指令字数:	1			
指令周期数:	1 (2)			
Q 周期操作:				
如果跳转:				
Q1	Q2	Q3	Q4	
译码	读立即数 n	处理数据	写入 PC	
空操作	空操作	空操作	空操作	
如果不跳转:				
Q1	Q2	Q3	Q4	
译码	读立即数 n	处理数据	空操作	

示例: HERE BNN Jump

执行指令前

PC = 地址 (HERE)

执行指令后

如果负标志位 = 0;

PC = 地址 (Jump)

如果负标志位 = 1;

PC = 地址 (HERE + 2)

PIC18F2XK20/4XK20

BNOV

不溢出则跳转

语法:	BNOV n			
操作数:	-128 ≤ n ≤ 127			
操作:	如果溢出标志位为 0 (PC) + 2 + 2n → PC			
受影响的状态位:	无			
机器码:	1110	0101	nnnn	nnnn
说明:	如果溢出标志位为 0, 程序将跳转。 “2n”(以二进制补码表示)与 PC 相加。 由于 PC 将递增以便取出下一条指令, 所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。			
指令字数:	1			
指令周期数:	1 (2)			
Q 周期操作:				
如果跳转:	Q1	Q2	Q3	Q4
	译码	读立即数 n	处理数据	写入 PC
如果不跳转:	空操作	空操作	空操作	空操作
	Q1	Q2	Q3	Q4
	译码	读立即数 n	处理数据	空操作

示例: HERE BNOV Jump

执行指令前
PC = 地址 (HERE)
执行指令后
如果溢出标志位 = 0;
PC = 地址 (Jump)
如果溢出标志位 = 1;
PC = 地址 (HERE + 2)

BNZ

不为零则跳转

语法:	BNZ n			
操作数:	-128 ≤ n ≤ 127			
操作:	如果全零标志位为 0 (PC) + 2 + 2n → PC			
受影响的状态位:	无			
机器码:	1110	0001	nnnn	nnnn
说明:	如果全零标志位为 0, 程序将跳转。 “2n”(以二进制补码表示)与 PC 相加。 由于 PC 将递增以便取出下一条指令, 所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。			
指令字数:	1			
指令周期数:	1 (2)			
Q 周期操作:				
如果跳转:	Q1	Q2	Q3	Q4
	译码	读立即数 n	处理数据	写入 PC
如果不跳转:	空操作	空操作	空操作	空操作
	Q1	Q2	Q3	Q4
	译码	读立即数 n	处理数据	空操作

示例: HERE BNZ Jump

执行指令前
PC = 地址 (HERE)
执行指令后
如果全零标志位 = 0;
PC = 地址 (Jump)
如果全零标志位 = 1;
PC = 地址 (HERE + 2)

BRA 无条件跳转

语法:	BRA n			
操作数:	$-1024 \leq n \leq 1023$			
操作:	$(PC) + 2 + 2n \rightarrow PC$			
受影响的状态位:	无			
机器码:	1101	0nnn	nnnn	nnnn
说明:	<p>“2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 $PC + 2 + 2n$。该指令为一条双周期指令。</p>			
指令字数:	1			
指令周期数:	2			
Q 周期操作:				
	Q1	Q2	Q3	Q4
	译码	读立即数 n	处理数据	写入 PC
	空操作	空操作	空操作	空操作

示例: HERE BRA Jump

执行指令前
PC = 地址 (HERE)
执行指令后
PC = 地址 (Jump)

BSF 将 f 中的某位置 1

语法:	BSF f, b {,a}			
操作数:	$0 \leq f \leq 255$			
	$0 \leq b \leq 7$			
	$a \in [0,1]$			
操作:	$1 \rightarrow f$			
受影响的状态位:	无			
机器码:	1000	bbba	ffff	ffff
说明:	<p>将寄存器 f 的位 b 置 1。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。</p>			
指令字数:	1			
指令周期数:	1			
Q 周期操作:				
	Q1	Q2	Q3	Q4
	译码	读寄存器 f	处理数据	写寄存器 f

示例: BSF FLAG_REG, 7, 1

执行指令前
FLAG_REG = 0Ah
执行指令后
FLAG_REG = 8Ah

BTFSC		测试文件寄存器中的某位，为 0 则跳过	
语法:	BTFSC f, b {,a}		
操作数:	0 ≤ f ≤ 255 0 ≤ b ≤ 7 a ∈ [0,1]		
操作:	如果 (f) = 0，则跳过		
受影响的状态位:	无		
机器码:	1011 bbba ffff ffff		
说明:	如果寄存器 f 的位 b 为 0，则跳过下一条指令。即在 b 位为 0 时，丢弃执行当前指令过程中已取的下一条指令，转而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。		
指令字数:	1		
指令周期数:	1 (2)		
注:	如果跳过，且后面跟有 2 字指令，则执行 BTFSC 需要 3 个周期。		
Q 周期操作:			
	Q1 Q2 Q3 Q4		
	译码 读寄存器 f 处理数据 空操作		
如果跳过:			
	Q1 Q2 Q3 Q4		
	空操作 空操作 空操作 空操作		
如果跳过，且后面跟有 2 字指令:			
	Q1 Q2 Q3 Q4		
	空操作 空操作 空操作 空操作		
	空操作 空操作 空操作 空操作		

示例:

```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE   :
```

执行指令前
PC = 地址 (HERE)

执行指令后
如果 FLAG<1> = 0;
 PC = 地址 (TRUE)
如果 FLAG<1> = 1;
 PC = 地址 (FALSE)

BTFSS		测试文件寄存器中的某位，为 1 则跳过	
语法:	BTFSS f, b {,a}		
操作数:	0 ≤ f ≤ 255 0 ≤ b < 7 a ∈ [0,1]		
操作:	如果 (f) = 1，则跳过		
受影响的状态位:	无		
机器码:	1010 bbba ffff ffff		
说明:	如果寄存器 f 的位 b 为 1，则跳过下一条指令。即在 b 位为 1 时，丢弃执行当前指令过程中已取的下一条指令，转而执行一条 NOP 指令，使该指令成为双周期指令。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 f ≤ 95 (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。		
指令字数:	1		
指令周期数:	1 (2)		
注:	如果跳过，且后面跟有 2 字指令，则执行 BTFSS 需要 3 个周期。		
Q 周期操作:			
	Q1 Q2 Q3 Q4		
	译码 读寄存器 f 处理数据 空操作		
如果跳过:			
	Q1 Q2 Q3 Q4		
	空操作 空操作 空操作 空操作		
如果跳过，且后面跟有 2 字指令:			
	Q1 Q2 Q3 Q4		
	空操作 空操作 空操作 空操作		
	空操作 空操作 空操作 空操作		

示例:

```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE   :
```

执行指令前
PC = 地址 (HERE)

执行指令后
如果 FLAG<1> = 0;
 PC = 地址 (FALSE)
如果 FLAG<1> = 1;
 PC = 地址 (TRUE)

BTG 将 f 中的某位取反

语法:	BTG f, b {,a}								
操作数:	$0 \leq f \leq 255$ $0 \leq b < 7$ $a \in [0,1]$								
操作:	$(f) \rightarrow f$								
受影响的状态位:	无								
机器码:	0111 bbba ffff ffff								
说明:	将数据存储单元 f 中的位 b 取反。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。 指令字数: 1 指令周期数: 1 Q 周期操作: <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>译码</td> <td>读寄存器 f</td> <td>处理数据</td> <td>写寄存器 f</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	写寄存器 f
Q1	Q2	Q3	Q4						
译码	读寄存器 f	处理数据	写寄存器 f						

示例: BTG PORTC, 4, 0

执行指令前:
PORTC = 0111 0101 [75h]

执行指令后:
PORTC = 0110 0101 [65h]

BOV 溢出则跳转

语法:	BOV n												
操作数:	$-128 \leq n \leq 127$												
操作:	如果溢出标志位为 1 $(PC) + 2 + 2n \rightarrow PC$												
受影响的状态位:	无												
机器码:	1110 0100 nnnn nnnn												
说明:	如果溢出标志位为 1, 程序将跳转。 “2n”(以二进制补码表示)与 PC 相加。 由于 PC 将递增以便取出下一条指令, 所以新地址将为 $PC + 2 + 2n$ 。该指令为一条双周期指令。												
指令字数:	1												
指令周期数:	1 (2)												
Q 周期操作:													
如果跳转:	<table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>译码</td> <td>读立即数 n</td> <td>处理数据</td> <td>写入 PC</td> </tr> <tr> <td>空操作</td> <td>空操作</td> <td>空操作</td> <td>空操作</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读立即数 n	处理数据	写入 PC	空操作	空操作	空操作	空操作
Q1	Q2	Q3	Q4										
译码	读立即数 n	处理数据	写入 PC										
空操作	空操作	空操作	空操作										
如果不跳转:	<table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>译码</td> <td>读立即数 n</td> <td>处理数据</td> <td>空操作</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读立即数 n	处理数据	空操作				
Q1	Q2	Q3	Q4										
译码	读立即数 n	处理数据	空操作										

示例: HERE BOV JUMP

执行指令前
PC = 地址 (HERE)
执行指令后
如果溢出标志位 = 1;
PC = 地址 (JUMP)
如果溢出标志位 = 0;
PC = 地址 (HERE + 2)

PIC18F2XK20/4XK20

BZ 为零则跳转

语法:	BZ n			
操作数:	$-128 \leq n \leq 127$			
操作:	如果全零标志位为 1 $(PC) + 2 + 2n \rightarrow PC$			
受影响的状态位:	无			
机器码:	1110	0000	nnnn	nnnn
说明:	如果全零标志位为 1, 程序将跳转。 “2n”(以二进制补码表示)与 PC 相加。由于 PC 将递增以便取出下一条指令, 所以新地址将为 $PC + 2 + 2n$ 。该指令为一条双周期指令。			
指令字数:	1			
指令周期数:	1 (2)			
Q 周期操作:				
如果跳转:				
	Q1	Q2	Q3	Q4
	译码	读立即数 n	处理数据	写入 PC
	空操作	空操作	空操作	空操作
如果不跳转:				
	Q1	Q2	Q3	Q4
	译码	读立即数 n	处理数据	空操作

示例: HERE BZ Jump

执行指令前
PC = 地址 (HERE)
执行指令后
如果全零标志位 = 1:
PC = 地址 (Jump)
如果全零标志位 = 0:
PC = 地址 (HERE + 2)

CALL 调用子程序

语法:	CALL k {s}			
操作数:	$0 \leq k \leq 1048575$ $s \in [0,1]$			
操作:	$(PC) + 4 \rightarrow TOS$, $k \rightarrow PC<20:1>$, 如果 $s = 1$ $(W) \rightarrow WS$, $(STATUS) \rightarrow STATUS$, $(BSR) \rightarrow BSRS$			
受影响的状态位:	无			
机器码:	1110	110s	k_7k_{15}	kkk_0
第一个字 (k<7:0>)	1111	$k_{19}k_{18}$	kkkk	kkk_8
第二个字 (k<19:8>)				
说明:	可在整个 2 MB 的存储器范围内进行子程序调用。首先, 将返回地址 (PC + 4) 压入返回堆栈。如果 s = 1, 还会将 W、STATUS 和 BSR 寄存器的内容存入它们各自的影子寄存器 WS、STATUS 和 BSRS。如果 s = 0, 将不会进行任何更新 (默认)。然后, 将 20 位的值 k 装入 PC<20:1>。CALL 是一条双周期指令。			
指令字数:	2			
指令周期数:	2			
Q 周期操作:				
	Q1	Q2	Q3	Q4
	译码	读立即数 k<7:0>	将 PC 压入堆栈	读立即数 k<19:8>, 写入 PC
	空操作	空操作	空操作	空操作

示例: HERE CALL THERE, 1

执行指令前
PC = 地址 (HERE)
执行指令后
PC = 地址 (THERE)
TOS = 地址 (HERE + 4)
WS = W
BSRS = BSR
STATUS = STATUS

CLRF	将 f 清零		
语法:	CLRF f {,a}		
操作数:	$0 \leq f \leq 255$ $a \in [0,1]$		
操作:	$000h \rightarrow f$ $1 \rightarrow Z$		
受影响的状态位:	Z		
机器码:	0110 101a ffff ffff		
说明:	清零指定寄存器的内容。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例: CLRF FLAG_REG, 1

执行指令前
FLAG_REG = 5Ah

执行指令后
FLAG_REG = 00h

CLRWDT	将看门狗定时器清零		
语法:	CLRWDT		
操作数:	无		
操作:	$000h \rightarrow WDT$, $000h \rightarrow WDT$ 后分频器, $1 \rightarrow TO$, $1 \rightarrow PD$		
受影响的状态位:	TO 和 PD		
机器码:	0000 0000 0000 0100		
说明:	CLRWDT 指令复位看门狗定时器及其后分频器。状态位 TO 和 PD 置 1。		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	空操作	处理数据	空操作

示例: CLRWDT

执行指令前	
WDT 计数器	= ?
执行指令后	
WDT 计数器	= 00h
WDT 后分频器	= 0
TO	= 1
PD	= 1

COMF	对 f 取反				
语法:	COMF f {,d {,a}}				
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$				
操作:	$(f) \rightarrow dest$				
受影响的状态位:	N 和 Z				
机器码:	<table border="1"><tr><td>0001</td><td>11da</td><td>ffff</td><td>ffff</td></tr></table>	0001	11da	ffff	ffff
0001	11da	ffff	ffff		
说明:	将寄存器 f 的内容取反。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
	Q1 Q2 Q3 Q4				
	译码	读寄存器 f	处理数据	写入 目标寄存器	

示例:	COMF	REG, 0, 0
执行指令前		
REG	= 13h	
执行指令后		
REG	= 13h	
W	= ECh	

CPFSEQ	比较 f 和 W, 如果 f = W 则跳过				
语法:	CPFSEQ f {,a}				
操作数:	$0 \leq f \leq 255$ $a \in [0,1]$				
操作:	$(f) - (W)$, 如果 $(f) = (W)$, 则跳过 (无符号比较)				
受影响的状态位:	无				
机器码:	<table border="1"><tr><td>0110</td><td>001a</td><td>ffff</td><td>ffff</td></tr></table>	0110	001a	ffff	ffff
0110	001a	ffff	ffff		
说明:	通过执行无符号的减法, 将数据存储单元 f 的内容与 W 的内容作比较。 如果 f = W, 则丢弃已取的指令转而执行一条执行一条 NOP 指令, 使该指令成为双周期指令。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。				
指令字数:	1				
指令周期数:	1 (2)				
Q 周期操作:					
	Q1 Q2 Q3 Q4				
	译码	读寄存器 f	处理数据	空操作	
如果跳过:					
	Q1 Q2 Q3 Q4				
	空操作	空操作	空操作	空操作	
如果跳过, 且后面跟有 2 字指令:					
	Q1 Q2 Q3 Q4				
	空操作	空操作	空操作	空操作	
	空操作	空操作	空操作	空操作	

示例:	HERE	CPFSEQ REG, 0
	NEQUAL	:
	EQUAL	:

执行指令前

PC 地址	=	HERE
W	=	?
REG	=	?

执行指令后

如果 REG	=	W;
PC	=	地址 (EQUAL)
如果 REG	≠	W;
PC	=	地址 (NEQUAL)

CPFSGT	比较 f 和 W, 如果 f > W 则跳过												
语法:	CPFSGT f {,a}												
操作数:	$0 \leq f \leq 255$ $a \in [0,1]$												
操作:	$(f) - (W)$, 如果 $(f) > (W)$, 则跳过 (无符号比较)												
受影响的状态位:	无												
机器码:	<table border="1"><tr><td>0110</td><td>010a</td><td>ffff</td><td>ffff</td></tr></table>	0110	010a	ffff	ffff								
0110	010a	ffff	ffff										
说明:	通过执行无符号的减法, 将数据存储单元 f 的内容与 W 的内容作比较。 如果 $f > W$, 则丢弃已取的指令转而执行一条 NOP 指令, 使该指令成为双周期指令。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。												
指令字数:	1												
指令周期数:	1 (2) 注: 如果跳过, 且后面跟有 2 字指令, 则执行 CPFSGT 需要 3 个周期。												
Q 周期操作:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>译码</td><td>读寄存器 f</td><td>处理数据</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	空操作				
Q1	Q2	Q3	Q4										
译码	读寄存器 f	处理数据	空操作										
如果跳过:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作				
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
如果跳过, 且后面跟有 2 字指令:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作	空操作	空操作	空操作	空操作
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
空操作	空操作	空操作	空操作										
示例:	HERE CPFSGT REG, 0 NGREATERT : GREATER :												
执行指令前	PC = 地址 (HERE) W = ?												
执行指令后	如果 REG > W ; PC = 地址 (GREATER) 如果 REG ≤ W ; PC = 地址 (NGREATERT)												

CPFSLT	比较 f 和 W, 如果 f < W 则跳过												
语法:	CPFSLT f {,a}												
操作数:	$0 \leq f \leq 255$ $a \in [0,1]$												
操作:	$(f) - (W)$, 如果 $(f) < (W)$, 则跳过 (无符号比较)												
受影响的状态位:	无												
机器码:	<table border="1"><tr><td>0110</td><td>000a</td><td>ffff</td><td>ffff</td></tr></table>	0110	000a	ffff	ffff								
0110	000a	ffff	ffff										
说明:	通过执行无符号的减法, 将数据存储单元 f 的内容与 W 的内容作比较。 如果 $f < W$, 则丢弃已取的指令转而执行一条 NOP 指令, 使该指令成为双周期指令。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。												
指令字数:	1												
指令周期数:	1 (2) 注: 如果跳过, 且后面跟有 2 字指令, 则执行 CPFSLT 需要 3 个周期。												
Q 周期操作:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>译码</td><td>读寄存器 f</td><td>处理数据</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	空操作				
Q1	Q2	Q3	Q4										
译码	读寄存器 f	处理数据	空操作										
如果跳过:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作				
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
如果跳过, 且后面跟有 2 字指令:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作	空操作	空操作	空操作	空操作
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
空操作	空操作	空操作	空操作										

示例:
 HERE CPFSLT REG, 1
 NLESS :
 LESS :
 执行指令前
 PC = 地址 (HERE)
 W = ?
 执行指令后
 如果 REG < W ;
 PC = 地址 (LESS)
 如果 REG ≥ W ;
 PC = 地址 (NLESS)

DAW 对 W 寄存器进行十进制调整					
语法:	DAW				
操作数:	无				
操作:	<p>如果 $[W<3:0> > 9]$ 或 $[DC = 1]$, 则 $(W<3:0>) + 6 \rightarrow W<3:0>;$ 否则 $(W<3:0>) \rightarrow W<3:0>;$</p> <p>如果 $[W<7:4> + DC > 9]$ 或 $[C = 1]$, 则 $(W<7:4>) + 6 + DC \rightarrow W<7:4>;$ 否则 $(W<7:4>) + DC \rightarrow W<7:4>$</p>				
受影响的状态位:	C				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>0000</td><td>0000</td><td>0000</td><td>0111</td></tr> </table>	0000	0000	0000	0111
0000	0000	0000	0111		
说明:	DAW 指令调整 W 寄存器内的 8 位值, 即之前两个压缩 BCD 格式的变量之和, 并产生一个正确的压缩 BCD 格式结果。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1 Q2 Q3 Q4					
译码 读寄存器 W 处理数据 写 W					

<u>例 1:</u>	DAW
执行指令前	
W	= A5h
C	= 0
DC	= 0
执行指令后	
W	= 05h
C	= 1
DC	= 0
<u>例 2:</u>	
执行指令前	
W	= CEh
C	= 0
DC	= 0
执行指令后	
W	= 34h
C	= 1
DC	= 0

DECF f 递减 1					
语法:	DECF f {d {,a}}				
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$				
操作:	$(f) - 1 \rightarrow dest$				
受影响的状态位:	C、DC、N、OV 和 Z				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>0000</td><td>01da</td><td>ffff</td><td>ffff</td></tr> </table>	0000	01da	ffff	ffff
0000	01da	ffff	ffff		
说明:	<p>将寄存器 f 的内容递减 1。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。</p> <p>如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。</p> <p>如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。</p>				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1 Q2 Q3 Q4					
译码 读寄存器 f 处理数据 写入 目标寄存器					

<u>示例:</u>	DECF CNT, 1, 0
执行指令前	
CNT	= 01h
Z	= 0
执行指令后	
CNT	= 00h
Z	= 1

DECFSZ	f 递减 1, 为 0 则跳过												
语法:	DECFSZ f {,d {,a}}												
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$												
操作:	$(f) - 1 \rightarrow dest$, 如果结果 = 0 则跳过												
受影响的状态位:	无												
机器码:	<table border="1"><tr><td>0010</td><td>11da</td><td>ffff</td><td>ffff</td></tr></table>	0010	11da	ffff	ffff								
0010	11da	ffff	ffff										
说明:	将寄存器 f 的内容递减 1。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。 如果结果为 0, 则丢弃已取的指令转而执行一条 NOP 指令, 使该指令成为双周期指令。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。												
指令字数:	1												
指令周期数:	1 (2)												
注:	如果跳过, 且后面跟有 2 字指令, 则执行 DECFSZ 需要 3 个周期。												
Q 周期操作:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>译码</td><td>读寄存器 f</td><td>处理数据</td><td>写入目标寄存器</td></tr></table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	写入目标寄存器				
Q1	Q2	Q3	Q4										
译码	读寄存器 f	处理数据	写入目标寄存器										
如果跳过:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作				
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
如果跳过, 且后面跟有 2 字指令:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作	空操作	空操作	空操作	空操作
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
空操作	空操作	空操作	空操作										
示例:	HERE DECFSZ CNT, 1, 1 GOTO LOOP CONTINUE												
执行指令前	PC = 地址 (HERE)												
执行指令后	CNT = CNT - 1 如果 CNT = 0: PC = 地址 (CONTINUE) 如果 CNT ≠ 0: PC = 地址 (HERE + 2)												

DCFSNZ	f 递减 1, 非 0 则跳过												
语法:	DCFSNZ f {,d {,a}}												
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$												
操作:	$(f) - 1 \rightarrow dest$, 如果结果 ≠ 0 则跳过												
受影响的状态位:	无												
机器码:	<table border="1"><tr><td>0100</td><td>11da</td><td>ffff</td><td>ffff</td></tr></table>	0100	11da	ffff	ffff								
0100	11da	ffff	ffff										
说明:	将寄存器 f 的内容递减 1。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。 如果结果不为 0, 则丢弃已取的指令转而执行一条 NOP 指令, 使该指令成为双周期指令。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。												
指令字数:	1												
指令周期数:	1 (2)												
注:	如果跳过, 且后面跟有 2 字指令, 则执行 DCFSNZ 需要 3 个周期。												
Q 周期操作:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>译码</td><td>读寄存器 f</td><td>处理数据</td><td>写入目标寄存器</td></tr></table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	写入目标寄存器				
Q1	Q2	Q3	Q4										
译码	读寄存器 f	处理数据	写入目标寄存器										
如果跳过:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作				
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
如果跳过, 且后面跟有 2 字指令:	<table border="1"><tr><th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr><tr><td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr></table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作	空操作	空操作	空操作	空操作
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
空操作	空操作	空操作	空操作										
示例:	HERE DCFSNZ TEMP, 1, 0 ZERO : NZERO :												
执行指令前	TEMP = ?												
执行指令后	TEMP = TEMP - 1, 如果 TEMP = 0: PC = 地址 (ZERO) 如果 TEMP ≠ 0: PC = 地址 (NZERO)												

PIC18F2XK20/4XK20

GOTO

无条件跳转

语法:	GOTO k											
操作数:	$0 \leq k \leq 1048575$											
操作:	$k \rightarrow PC<20:1>$											
受影响的状态位:	无											
机器码:	<table border="1" style="width: 100%; text-align: center;"> <tr> <td>1110</td> <td>1111</td> <td>k_7k_{15}</td> <td>$k_{15}k_0$</td> </tr> <tr> <td>1111</td> <td>$k_{19}k_{15}$</td> <td>kkkk</td> <td>kkk_8</td> </tr> </table>				1110	1111	k_7k_{15}	$k_{15}k_0$	1111	$k_{19}k_{15}$	kkkk	kkk_8
1110	1111	k_7k_{15}	$k_{15}k_0$									
1111	$k_{19}k_{15}$	kkkk	kkk_8									
说明:	<p>GOTO 指令允许无条件跳转到整个 2 MB 存储器范围中的任何位置。将 20 位值 k 装入 $PC<20:1>$。GOTO 始终为双周期指令。</p>											
指令字数:	2											
指令周期数:	2											
Q 周期操作:												
	Q1	Q2	Q3	Q4								
	译码	读立即数 $k<7:0>$	空操作	读立即数 $k<19:8>$, 写入 PC								
	空操作	空操作	空操作	空操作								

示例:

GOTO THERE

执行指令后

PC = 地址 (THERE)

INCF

f 递增 1

语法:	INCF f {,d {,a}}			
操作数:	$0 \leq f \leq 255$			
	$d \in [0,1]$			
	$a \in [0,1]$			
操作:	$(f) + 1 \rightarrow dest$			

受影响的状态位:

C、DC、N、OV 和 Z

机器码:

0010 10da ffff ffff

说明:

将寄存器 f 的内容递增 1。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f (默认)。

如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。

如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针
对位的指令”。

指令字数:

1

指令周期数:

1

Q 周期操作:

Q1	译码	读寄存器 f	处理数据	写入 目标寄存器
----	----	--------	------	-------------

示例:

INCF CNT, 1, 0

执行指令前

CNT	=	FFh
Z	=	0
C	=	?
DC	=	?

执行指令后

CNT	=	00h
Z	=	1
C	=	1
DC	=	1

INCFSZ	f 递增 1, 为 0 则跳过												
语法:	INCFSZ f {,d {,a}}												
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$												
操作:	$(f) + 1 \rightarrow dest$, 如果结果 = 0 则跳过												
受影响的状态位:	无												
机器码:	0011 11da ffff ffff												
说明:	将寄存器 f 的内容递增 1。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。 如果结果为 0, 则丢弃已取的指令转而执行一条 NOP 指令, 使该指令成为双周期指令。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。												
指令字数:	1												
指令周期数:	1 (2) 注: 如果跳过, 且后面跟有 2 字指令, 则执行 INCFSZ 需要 3 个周期。												
Q 周期操作:	<table border="1"> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> <tr> <td>译码</td><td>读寄存器 f</td><td>处理数据</td><td>写入目标寄存器</td></tr> </table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	写入目标寄存器				
Q1	Q2	Q3	Q4										
译码	读寄存器 f	处理数据	写入目标寄存器										
如果跳过:	<table border="1"> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> <tr> <td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr> </table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作				
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
如果跳过, 且后面跟有 2 字指令:	<table border="1"> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> <tr> <td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr> <tr> <td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr> </table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作	空操作	空操作	空操作	空操作
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
空操作	空操作	空操作	空操作										
示例:	HERE INCFSZ CNT, 1, 0 NZERO : ZERO :												
执行指令前	PC = 地址 (HERE)												
执行指令后	CNT = CNT + 1 如果 CNT = 0: PC = 地址 (ZERO) 如果 CNT ≠ 0: PC = 地址 (NZERO)												

INFSNZ	f 递增 1, 非 0 则跳过												
语法:	INFSNZ f {,d {,a}}												
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$												
操作:	$(f) + 1 \rightarrow dest$, 如果结果 ≠ 0 则跳过												
受影响的状态位:	无												
机器码:	0100 10da ffff ffff												
说明:	将寄存器 f 的内容递增 1。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。 如果结果不为 0, 则丢弃已取的指令转而执行一条 NOP 指令, 使该指令成为双周期指令。 如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。												
指令字数:	1												
指令周期数:	1 (2) 注: 如果跳过, 且后面跟有 2 字指令, 则执行 INFSNZ 需要 3 个周期。												
Q 周期操作:	<table border="1"> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> <tr> <td>译码</td><td>读寄存器 f</td><td>处理数据</td><td>写入目标寄存器</td></tr> </table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	写入目标寄存器				
Q1	Q2	Q3	Q4										
译码	读寄存器 f	处理数据	写入目标寄存器										
如果跳过:	<table border="1"> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> <tr> <td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr> </table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作				
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
如果跳过, 且后面跟有 2 字指令:	<table border="1"> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> <tr> <td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr> <tr> <td>空操作</td><td>空操作</td><td>空操作</td><td>空操作</td></tr> </table>	Q1	Q2	Q3	Q4	空操作	空操作	空操作	空操作	空操作	空操作	空操作	空操作
Q1	Q2	Q3	Q4										
空操作	空操作	空操作	空操作										
空操作	空操作	空操作	空操作										
示例:	HERE INFSNZ REG, 1, 0 ZERO : NZERO :												
执行指令前	PC = 地址 (HERE)												
执行指令后	REG = REG + 1 如果 REG ≠ 0: PC = 地址 (NZERO) 如果 REG = 0: PC = 地址 (ZERO)												

IORLW 将立即数与 W 作逻辑或运算

语法:	IORLW k				
操作数:	$0 \leq k \leq 255$				
操作:	$(W) .OR. k \rightarrow W$				
受影响的状态位:	N 和 Z				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td>0000</td> <td>1001</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	0000	1001	kkkk	kkkk
0000	1001	kkkk	kkkk		
说明:	将 W 的内容与 8 位立即数 k 进行逻辑或运算。结果存储在 W 寄存器中。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1 Q2 Q3 Q4					
译码 读立即数 k 处理数据 写入 W					

示例: IORLW 35h

执行指令前
W = 9Ah
执行指令后
W = BFh

IORWF 将 W 与 f 作逻辑或运算

语法:	IORWF f {,d {,a}}				
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$				
操作:	$(W) .OR.(f) \rightarrow dest$				
受影响的状态位:	N 和 Z				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td>0001</td> <td>00da</td> <td>ffff</td> <td>ffff</td> </tr> </table>	0001	00da	ffff	ffff
0001	00da	ffff	ffff		
说明:	将 W 的内容与寄存器 f 的内容进行逻辑或运算。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令将以立即数变址寻址模式进行操作。详情请参见第24.2.3节“立即数变址寻址模式中针对字节和针对位的指令”。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1 Q2 Q3 Q4					
译码 读寄存器 f 处理数据 写入目标寄存器					

示例: IORWF RESULT, 0, 1

执行指令前
RESULT = 13h
W = 91h
执行指令后
RESULT = 13h
W = 93h

LFSR	装入 FSR															
语法:	LFSR f, k															
操作数:	0 ≤ f ≤ 2 0 ≤ k ≤ 4095															
操作:	$k \rightarrow \text{FSR}_f$															
受影响的状态位:	无															
机器码:	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>1110</td><td>1110</td><td>00ff</td><td>$k_{11}k_{10}k_9k_8$</td></tr> <tr> <td>1111</td><td>0000</td><td>$k_7k_6k_5k_4$</td><td>kkkk</td></tr> </table>				1110	1110	00ff	$k_{11}k_{10}k_9k_8$	1111	0000	$k_7k_6k_5k_4$	kkkk				
1110	1110	00ff	$k_{11}k_{10}k_9k_8$													
1111	0000	$k_7k_6k_5k_4$	kkkk													
说明:	将 12 位立即数 k 装入 f 所指向的文件选择寄存器。															
指令字数:	2															
指令周期数:	2															
Q 周期操作:	<table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> </thead> <tbody> <tr> <td>译码</td><td>读立即数 k 的 MSB</td><td>处理数据</td><td>将立即数 k 的 MSB 写入 FSR_fH</td></tr> <tr> <td>译码</td><td>读立即数 k 的 LSB</td><td>处理数据</td><td>将立即数 k 的 LSB 写入 FSR_fL</td></tr> </tbody> </table>				Q1	Q2	Q3	Q4	译码	读立即数 k 的 MSB	处理数据	将立即数 k 的 MSB 写入 FSR_fH	译码	读立即数 k 的 LSB	处理数据	将立即数 k 的 LSB 写入 FSR_fL
Q1	Q2	Q3	Q4													
译码	读立即数 k 的 MSB	处理数据	将立即数 k 的 MSB 写入 FSR_fH													
译码	读立即数 k 的 LSB	处理数据	将立即数 k 的 LSB 写入 FSR_fL													

示例: LFSR 2, 3ABh

执行指令后
 FSR2H = 03h
 FSR2L = ABh

MOVF	传送 f											
语法:	MOVF f {,d {,a}}											
操作数:	0 ≤ f ≤ 255 $d \in [0,1]$ $a \in [0,1]$											
操作:	$f \rightarrow \text{dest}$											
受影响的状态位:	N 和 Z											
机器码:	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>0101</td><td>00da</td><td>ffff</td><td>ffff</td></tr> </table>				0101	00da	ffff	ffff				
0101	00da	ffff	ffff									
说明:	<p>根据 d 的状态, 将寄存器 f 的内容送入目标寄存器。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。f 可以为 256 字节存储区中的任何地址单元。</p> <p>如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。</p> <p>如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。</p>											
指令字数:	1											
指令周期数:	1											
Q 周期操作:	<table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> </thead> <tbody> <tr> <td>译码</td><td>读寄存器 f</td><td>处理数据</td><td>写 W</td></tr> </tbody> </table>				Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	写 W
Q1	Q2	Q3	Q4									
译码	读寄存器 f	处理数据	写 W									

示例: MOVF REG, 0, 0

执行指令前
 REG = 22h
 W = FFh

执行指令后
 REG = 22h
 W = 22h

PIC18F2XK20/4XK20

MOVFF 将源寄存器的内容送入目标寄存器

语法:	MOVFF f _s ,f _d												
操作数:	0 ≤ f _s ≤ 4095 0 ≤ f _d ≤ 4095												
操作:	(f _s) → f _d												
受影响的状态位:	无												
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td>1100</td> <td>ffff</td> <td>ffff</td> <td>fffff_s</td> </tr> <tr> <td>1111</td> <td>ffff</td> <td>ffff</td> <td>fffff_d</td> </tr> </table>	1100	ffff	ffff	fffff _s	1111	ffff	ffff	fffff _d				
1100	ffff	ffff	fffff _s										
1111	ffff	ffff	fffff _d										
说明:	<p>将源寄存器 f_s 的内容送入目标寄存器 f_d。源寄存器 f_s 可以是 4096 字节数据空间 (000h 到 FFFh) 中的任何单元，目标寄存器 f_d 也可以是 000h 到 FFFh 中的任何单元。</p> <p>源或目标寄存器都可以是 W (这是个有用的特点)。</p> <p>MOVFF 指令对于将数据存储单元中的内容送入外设寄存器 (如发送缓冲区或 I/O 端口) 的场合非常有用。</p> <p>MOVFF 指令不能使用 PCL、TOSU、TOSH 或 TOSL 作为目标寄存器。</p>												
指令字数:	2												
指令周期数:	2 (3)												
Q 周期操作:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> <tr> <td>译码</td> <td>读寄存器 f (源寄存器)</td> <td>处理数据</td> <td>空操作</td> </tr> <tr> <td>译码</td> <td>空操作 无假读</td> <td>空操作</td> <td>写寄存器 f (目标寄存器)</td> </tr> </table>	Q1	Q2	Q3	Q4	译码	读寄存器 f (源寄存器)	处理数据	空操作	译码	空操作 无假读	空操作	写寄存器 f (目标寄存器)
Q1	Q2	Q3	Q4										
译码	读寄存器 f (源寄存器)	处理数据	空操作										
译码	空操作 无假读	空操作	写寄存器 f (目标寄存器)										

示例: MOVFF REG1, REG2

执行指令前	
REG1	= 33h
REG2	= 11h
执行指令后	
REG1	= 33h
REG2	= 33h

MOVLB 将立即数送入 BSR 的低半字节

语法:	MOVLW k								
操作数:	0 ≤ k ≤ 255								
操作:	k → BSR								
受影响的状态位:	无								
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td>0000</td> <td>0001</td> <td>kkkk</td> <td>kkkk</td> </tr> </table>	0000	0001	kkkk	kkkk				
0000	0001	kkkk	kkkk						
说明:	将 8 位立即数 k 装入存储区选择寄存器 (BSR)。不管 k ₇ :k ₄ 的值如何，BSR<7:4> 的值将始终保持为 0。								
指令字数:	1								
指令周期数:	1								
Q 周期操作:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> <tr> <td>译码</td> <td>读立即数 k</td> <td>处理数据</td> <td>将立即数 k 写入 BSR</td> </tr> </table>	Q1	Q2	Q3	Q4	译码	读立即数 k	处理数据	将立即数 k 写入 BSR
Q1	Q2	Q3	Q4						
译码	读立即数 k	处理数据	将立即数 k 写入 BSR						

示例: MOVLB 5

执行指令前	BSR 寄存器 = 02h
执行指令后	BSR 寄存器 = 05h

MOVLW

将立即数送入 W

语法: MOVLW k
 操作数: $0 \leq k \leq 255$
 操作: $k \rightarrow W$
 受影响的状态位: 无
 机器码:

0000	1110	kkkk	kkkk
------	------	------	------

 说明: 将 8 位立即数 k 装入 W。
 指令字数: 1
 指令周期数: 1
 Q 周期操作:

Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

示例: MOVLW 5Ah

执行指令后
 $W = 5Ah$

MOVWF

将 W 的内容送入 f

语法: MOVWF f {,a}
 操作数: $0 \leq f \leq 255$
 $a \in [0,1]$
 操作: $(W) \rightarrow f$
 受影响的状态位: 无
 机器码:

0110	111a	ffff	ffff
------	------	------	------

 说明: 将 W 寄存器中的数据送入寄存器 f。f 可以是 256 字节存储区中的任何地址单元。如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针位的指令”。

指令字数: 1
 指令周期数: 1
 Q 周期操作:

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例: MOVWF REG, 0

执行指令前
 $W = 4Fh$
 $REG = FFh$
 执行指令后
 $W = 4Fh$
 $REG = 4Fh$

PIC18F2XK20/4XK20

MULLW

将立即数与 W 中的内容相乘

语法:	MULLW k				
操作数:	$0 \leq k \leq 255$				
操作:	$(W) \times k \rightarrow PRODH:PRODL$				
受影响的状态位:	无				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>0000</td><td>1101</td><td>kkkk</td><td>kkkk</td></tr></table>	0000	1101	kkkk	kkkk
0000	1101	kkkk	kkkk		
说明:	将 W 的内容与 8 位立即数 k 进行无符号的乘法运算。16 位的结果存储在 PRODH:PRODL 寄存器对中，其中 PRODH 用于存储高字节。W 的内容不改变。所有状态标志位都不受影响。请注意此操作不可能发生溢出或进位。结果有可能为全零，但不会被检测到。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	读立即数 k	处理数据	写寄存器 PRODH: PRODL		

示例: MULLW 0C4h

执行指令前		
W	=	E2h
PRODH	=	?
PRODL	=	?
执行指令后		
W	=	E2h
PRODH	=	ADh
PRODL	=	08h

MULWF

将 W 与 f 的内容相乘

语法:	MULWF f {,a}				
操作数:	$0 \leq f \leq 255$ $a \in [0,1]$				
操作:	$(W) \times (f) \rightarrow PRODH:PRODL$				
受影响的状态位:	无				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>0000</td><td>001a</td><td>ffff</td><td>ffff</td></tr></table>	0000	001a	ffff	ffff
0000	001a	ffff	ffff		
说明:	将 W 的内容与寄存器单元 f 的内容执行无符号的乘法运算。运算的 16 位结果保存在 PRODH:PRODL 寄存器对中，其中 PRODH 用于存储高字节。W 和 f 的内容都不改变。所有状态标志位都不受影响。请注意此操作不可能发生溢出或进位。结果有可能为全零，但不会被检测到。如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和对齐位的指令”。				

示例: MULWF REG, 1

执行指令前		
W	=	C4h
REG	=	B5h
PRODH	=	?
PRODL	=	?
执行指令后		
W	=	C4h
REG	=	B5h
PRODH	=	8Ah
PRODL	=	94h

NEGF	对 f 取补				
语法:	NEGF f {,a}				
操作数:	$0 \leq f \leq 255$ $a \in [0,1]$				
操作:	$(\bar{f}) + 1 \rightarrow f$				
受影响的状态位:	N、OV、C、DC 和 Z				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td>0110</td> <td>110a</td> <td>ffff</td> <td>ffff</td> </tr> </table>	0110	110a	ffff	ffff
0110	110a	ffff	ffff		
说明:	用二进制补码对存储单元 f 取补，结果存储在数据存储单元 f 中。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	读寄存器 f	处理数据	写寄存器 f		

<u>示例:</u>	NEGF REG, 1
执行指令前	
REG	= 0011 1010 [3Ah]
执行指令后	
REG	= 1100 0110 [C6h]

NOP	空操作								
语法:	NOP								
操作数:	无								
操作:	空操作								
受影响的状态位:	无								
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td>0000</td> <td>0000</td> <td>0000</td> <td>0000</td> </tr> <tr> <td>1111</td> <td>xxxx</td> <td>xxxx</td> <td>xxxx</td> </tr> </table>	0000	0000	0000	0000	1111	xxxx	xxxx	xxxx
0000	0000	0000	0000						
1111	xxxx	xxxx	xxxx						
说明:	不执行任何操作。								
指令字数:	1								
指令周期数:	1								
Q 周期操作:									
Q1	Q2	Q3	Q4						
译码	空操作	空操作	空操作						

示例:
无。

PIC18F2XK20/4XK20

POP 弹出返回堆栈栈顶的内容

语法:	POP				
操作数:	无				
操作:	(TOS) → 位桶 (即丢弃)				
受影响的状态位:	无				
机器码:	<table border="1"><tr><td>0000</td><td>0000</td><td>0000</td><td>0110</td></tr></table>	0000	0000	0000	0110
0000	0000	0000	0110		
说明:	从返回堆栈弹出 TOS 值并丢弃。然后，前一个压入返回堆栈的值成为 TOS 值。此指令可以让用户正确管理返回堆栈，从而实现软件堆栈。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	空操作	弹出 TOS 值	空操作		

示例: POP
GOTO NEW

执行指令前
TOS = 0031A2h
堆栈 (下一级) = 014332h

执行指令后
TOS = 014332h
PC = NEW

PUSH 将数据压入返回堆栈栈顶

语法:	PUSH				
操作数:	无				
操作:	(PC + 2) → TOS				
受影响的状态位:	无				
机器码:	<table border="1"><tr><td>0000</td><td>0000</td><td>0000</td><td>0101</td></tr></table>	0000	0000	0000	0101
0000	0000	0000	0101		
说明:	PC + 2 的值被压入返回堆栈的栈顶。原先的 TOS 值被压入堆栈的下一级。 此指令允许通过修改 TOS 并将其压入返回堆栈来实现软件堆栈。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	将 PC + 2 压入返回堆栈	空操作	空操作		

示例: PUSH

执行指令前
TOS = 345Ah
PC = 0124h

执行指令后
PC = 0126h
TOS = 0126h
堆栈 (下一级) = 345Ah

RCALL

相对调用

语法:	RCALL n		
操作数:	-1024 ≤ n ≤ 1023		
操作:	(PC) + 2 → TOS, (PC) + 2 + 2n → PC		
受影响的状态位:	无		
机器码:	1101 1nnn nnnn nnnn		
说明:	从当前地址跳转（最多 1K）来调用子程序。首先，将返回地址 (PC + 2) 压入返回堆栈。然后，将 “2n”（以二进制补码表示）与 PC 相加。由于 PC 将递增以便取出下一条指令，所以新地址将为 PC + 2 + 2n。该指令为一条双周期指令。		
指令字数:	1		
指令周期数:	2		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	读立即数 n 将 PC 压入堆栈	处理数据	写入 PC
空操作	空操作	空操作	空操作

示例: HERE RCALL Jump

执行指令前

PC = 地址 (HERE)

执行指令后

PC = 地址 (Jump)
TOS = 地址 (HERE + 2)

RESET

复位

语法:	RESET		
操作数:	无		
操作:	将所有受 MCLR 复位影响的寄存器和标志位复位。		
受影响的状态位:	全部		
机器码:	0000 0000 1111 1111		
说明:	此指令可实现用软件执行 MCLR 复位。		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	开始复位	空操作	空操作

示例: RESET

执行指令后
寄存器 = 复位值
标志位 * = 复位值

RETFIE 从中断返回					
语法:	RETFIE {s}				
操作数:	$s \in [0,1]$				
操作:	$(TOS) \rightarrow PC$, $1 \rightarrow GIE/GIEH$ 或 $PEIE/GIEL$, 如果 $s = 1$ $(WS) \rightarrow W$, $(STATUSS) \rightarrow STATUS$, $(BSRS) \rightarrow BSR$, $PCLATU$ 和 $PCLATH$ 保持不变				
受影响的状态位:	GIE/GIEH 和 PEIE/GIEL				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>0000</td><td>0000</td><td>0001</td><td>000s</td></tr> </table>	0000	0000	0001	000s
0000	0000	0001	000s		
说明:	从中断返回。执行出栈操作，将栈顶 (TOS) 的内容装入 PC。通过将高或低优先级全局中断允许位置 1，来允许中断。如果 $s = 1$ ，则影子寄存器 WS、STATUSS 和 BSRS 的内容将被装入对应的寄存器 W、STATUS 和 BSR。如果 $s = 0$ ，则不更新这些寄存器（默认）。				
指令字数:	1				
指令周期数:	2				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	空操作	空操作	从堆栈弹出 PC 值 将 GIEH 或 GIEL 置 1		
空操作	空操作	空操作	空操作		

示例: RETFIE 1

中断后

PC	= TOS
W	= WS
BSR	= BSRS
Status	= STATUSS
GIE/GIEH, PEIE/GIEL	= 1

RETLW 将立即数返回到 W					
语法:	RETLW k				
操作数:	$0 \leq k \leq 255$				
操作:	$k \rightarrow W$, $(TOS) \rightarrow PC$, $PCLATU$ 和 $PCLATH$ 保持不变				
受影响的状态位:	无				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>0000</td><td>1100</td><td>kkkk</td><td>kkkk</td></tr> </table>	0000	1100	kkkk	kkkk
0000	1100	kkkk	kkkk		
说明:	将 8 位立即数 k 装入 W。将栈顶内容（返回地址）装入程序计数器。高字节地址锁存器（PCLATH）内容保持不变。				
指令字数:	1				
指令周期数:	2				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	读立即数 k	处理数据	从堆栈弹出 PC 值, 写入 W		
空操作	空操作	空操作	空操作		

示例:

```

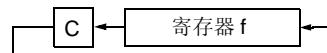
CALL TABLE ; W contains table
; offset value
; W now has
; table value
:
TABLE
  ADDWF PCL ; W = offset
  RETLW k0 ; Begin table
  RETLW k1 ;
  :
  RETLW kn ; End of table

  执行指令前
  W = 07h
  执行指令后
  W = kn 的值

```

RETURN	从子程序返回		
语法:	RETURN {s}		
操作数:	$s \in [0,1]$		
操作:	$(TOS) \rightarrow PC$, 如果 $s = 1$ $(WS) \rightarrow W$, $(STATUS) \rightarrow STATUS$, $(BSRS) \rightarrow BSR$, PCLATU 和 PCLATH 保持不变		
受影响的状态位:	无		
机器码:	0000 0000 0001 001s		
说明:	从子程序返回。执行出栈操作，将栈顶（TOS）的内容装入程序计数器。如果 $s = 1$ ，则影子寄存器 WS、STATUS 和 BSRS 的内容将被装入对应的寄存器 W、STATUS 和 BSR。如果 $s = 0$ ，则不更新这些寄存器（默认）。		
指令字数:	1		
指令周期数:	2		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	空操作	处理数据	从堆栈弹出 PC 值
空操作	空操作	空操作	空操作

示例: RETURN
执行指令后:
PC = TOS

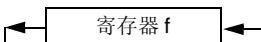
RLCF	f 带进位循环左移		
语法:	RLCF f {d {a}}		
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$		
操作:	$(f < n >) \rightarrow dest < n + 1 >$, $(f < 7 >) \rightarrow C$, $(C) \rightarrow dest < 0 >$		
受影响的状态位:	C、N 和 Z		
机器码:	0011 01da ffff ffff		
说明:	将寄存器 f 的内容连同进位标志位一起循环左移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。 		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入 目标寄存器

示例: RLCF REG, 0, 0
执行指令前
REG = 1110 0110
C = 0
执行指令后
REG = 1110 0110
W = 1100 1100
C = 1

PIC18F2XK20/4XK20

RLNCF

f 循环左移（不带进位）

语法:	RLNCF f {,d {,a}}
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$
操作:	$(f < n>) \rightarrow \text{dest} < n + 1>$, $(f < 7>) \rightarrow \text{dest} < 0>$
受影响的状态位:	N 和 Z
机器码:	0100 01da ffff ffff
说明:	将寄存器 f 的内容循环左移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。
	
指令字数:	1
指令周期数:	1
Q 周期操作:	
Q1 Q2 Q3 Q4	
译码 读寄存器 f 处理数据 写入 目标寄存器	

示例: RLNCF REG, 1, 0

执行指令前
REG = 1010 1011
执行指令后
REG = 0101 0111

RRCF

f 带进位循环右移

语法:	RRCF f {,d {,a}}
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$
操作:	$(f < n>) \rightarrow \text{dest} < n - 1>$, $(f < 0>) \rightarrow C$, $(C) \rightarrow \text{dest} < 7>$
受影响的状态位:	C、N 和 Z
机器码:	0011 00da ffff ffff
说明:	将寄存器 f 的内容连同进位标志位一起循环右移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。
	
指令字数:	1
指令周期数:	1
Q 周期操作:	
Q1 Q2 Q3 Q4	
译码 读寄存器 f 处理数据 写入 目标寄存器	

示例: RRCF REG, 0, 0

执行指令前
REG = 1110 0110
C = 0
执行指令后
REG = 1110 0110
W = 0111 0011
C = 0

RRNCF

f 循环右移（不带进位）

语法:	RRNCF f {,d ,a}		
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$		
操作:	$(f < n>) \rightarrow dest <n - 1>$, $(f < 0>) \rightarrow dest <7>$		
受影响的状态位:	N 和 Z		
机器码:	0100 00da ffff ffff		
说明:	将寄存器 f 的内容循环右移 1 位。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。 如果 a 为 0，选择快速操作存储区（默认），忽略 BSR 的值。如果 a 为 1，则根据 BSR 值选择存储区。 如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1:

RRNCF REG, 1, 0

执行指令前
REG = 1101 0111
执行指令后
REG = 1110 1011

例 2:

RRNCF REG, 0, 0

执行指令前
W = ?
REG = 1101 0111
执行指令后
W = 1110 1011
REG = 1101 0111

SETF

将 f 的内容置为全 1

语法:	SETF f {,a}		
操作数:	$0 \leq f \leq 255$ $a \in [0,1]$		
操作:	FFh \rightarrow f		
受影响的状态位:	无		
机器码:	0110 100a ffff ffff		
说明:	将指定寄存器的内容置为 FFh。 如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。 如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针 对位的指令”。		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写寄存器 f

示例:

执行指令前 REG	=	5Ah
执行指令后 REG	=	FFh

PIC18F2XK20/4XK20

SLEEP	进入休眠模式				
语法:	SLEEP				
操作数:	无				
操作:	00h → WDT, 0 → WDT 后分频器, 1 → \overline{TO} , 0 → PD				
受影响的状态位:	TO 和 PD				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>0000</td><td>0000</td><td>0000</td><td>0011</td></tr></table>	0000	0000	0000	0011
0000	0000	0000	0011		
说明:	掉电状态位 (PD) 清零。超时状态位 (TO) 置 1。看门狗定时器及其后分频器清零。 振荡器停振，处理器进入休眠模式。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1 Q2 Q3 Q4					
译码 空操作 处理数据 进入休眠模式					

示例: SLEEP

执行指令前
 $\overline{TO} = ?$
 $PD = ?$

执行指令后
 $\overline{TO} = 1 \dagger$
 $PD = 0$

† 如果由 WDT 引起唤醒，则该位将被清零。

SUBFWB W 减去 f (带借位)

语法:	SUBFWB f {,d {,a}}				
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$				
操作:	$(W) - (f) - (\overline{C}) \rightarrow dest$				
受影响的状态位:	N、OV、C、DC 和 Z				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td>0101</td><td>01da</td><td>ffff</td><td>ffff</td></tr></table>	0101	01da	ffff	ffff
0101	01da	ffff	ffff		
说明:	将 W 的内容减去 f 寄存器的内容和进位标志位 (借位) (通过二进制补码方式进行运算)。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第24.2.3节“立即数变址寻址模式中针对字节和针对位的指令”。				

指令字数:	1
指令周期数:	1
Q 周期操作:	
Q1 Q2 Q3 Q4	
译码 读寄存器 f 处理数据 写入目标寄存器	

例 1: SUBFWB REG, 1, 0

执行指令前
REG = 3
W = 2
C = 1

执行指令后
REG = FF
W = 2
C = 0
Z = 0
N = 1 ; 结果为负

例 2: SUBFWB REG, 0, 0

执行指令前
REG = 2
W = 5
C = 1

执行指令后
REG = 2
W = 3
C = 1
Z = 0
N = 0 ; 结果为正

例 3: SUBFWB REG, 1, 0

执行指令前
REG = 1
W = 2
C = 0

执行指令后
REG = 0
W = 2
C = 1
Z = 1
N = 0 ; 结果为全零

SUBLW	立即数减去 W 的内容		
语法:	SUBLW k		
操作数:	$0 \leq k \leq 255$		
操作:	$k - (W) \rightarrow W$		
受影响的状态位:	N、OV、C、DC 和 Z		
机器码:	0000 1000 kkkk kkkk		
说明	用 8 位立即数 k 减去 W。结果存储在 W 寄存器中。		
指令字数:	1		
指令周期数:	1		
Q 周期操作:			
Q1	Q2	Q3	Q4
译码	读立即数 k	处理数据	写入 W

例 1: SUBLW 02h

执行指令前

W	=	01h
C	=	?

执行指令后

W	=	01h
C	=	1
Z	=	0
N	=	0

; 结果为正

例 2: SUBLW 02h

执行指令前

W	=	02h
C	=	?

执行指令后

W	=	00h
C	=	1
Z	=	1
N	=	0

; 结果为全零

例 3: SUBLW 02h

执行指令前

W	=	03h
C	=	?

执行指令后

W	=	FFh
C	=	0
Z	=	0
N	=	1

; (二进制补码)

; 结果为负

SUBWF	f 减去 W		
语法:	SUBWF f {,d {,a}}		
操作数:	$0 \leq f \leq 255$		
	$d \in [0,1]$		
	$a \in [0,1]$		
操作:	$(f) - (W) \rightarrow dest$		
受影响的状态位:	N、OV、C、DC 和 Z		
机器码:	0101 11da ffff ffff		
说明:	用寄存器 f 中的内容减去 W 寄存器的内容（通过二进制补码方式进行运算）。如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f（默认）。如果 a 为 0，选择快速操作存储区。如果 a 为 1，使用 BSR 选择 GPR 存储区。如果 a 为 0 且使能了扩展指令集，只要 $f \leq 95$ (5Fh)，指令就将以立即数变址寻址模式进行操作。详情请参见第24.2.3节“立即数变址寻址模式中针对字节和针对应的指令”。		
Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

例 1: SUBWF REG, 1, 0

执行指令前

REG	=	3
W	=	2
C	=	?

执行指令后

REG	=	1
W	=	2
C	=	1
Z	=	0
N	=	0

; 结果为正

例 2: SUBWF REG, 0, 0

执行指令前

REG	=	2
W	=	2
C	=	?

执行指令后

REG	=	2
W	=	0
C	=	1
Z	=	1
N	=	0

; 结果为全零

例 3: SUBWF REG, 1, 0

执行指令前

REG	=	1
W	=	2
C	=	?

执行指令后

REG	=	FFh
W	=	2
C	=	0
Z	=	0
N	=	1

; (二进制补码)

; 结果为负

SUBWFB f 减去 W (带借位)

语法:	SUBWFB f {,d {,a}}			
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$			
操作:	$(f) - (W) - (\bar{C}) \rightarrow dest$			
受影响的状态位:	N、OV、C、DC 和 Z			
机器码:	0101	10da	ffff	ffff
说明:	<p>用 f 寄存器的内容减去 W 的内容和进位标志位 (借位) (通过二进制补码方式进行运算)。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。</p> <p>如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。</p> <p>如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。</p>			
指令字数:	1			
指令周期数:	1			
Q 周期操作:				
	Q1	Q2	Q3	Q4
	译码	读寄存器 f	处理数据	写入目标寄存器

例 1: SUBWFB REG, 1, 0

执行指令前	REG	=	19h	(0001 1001)
	W	=	0Dh	(0000 1101)
	C	=	1	

执行指令后	REG	=	0Ch	(0000 1100)
	W	=	0Dh	(0000 1101)
	C	=	1	
	Z	=	0	
	N	=	0	; 结果为正

例 2: SUBWFB REG, 0, 0

执行指令前	REG	=	1Bh	(0001 1011)
	W	=	1Ah	(0001 1010)
	C	=	0	

执行指令后	REG	=	1Bh	(0001 1011)
	W	=	00h	
	C	=	1	
	Z	=	1	; 结果为全零
	N	=	0	

例 3: SUBWFB REG, 1, 0

执行指令前	REG	=	03h	(0000 0011)
	W	=	0Eh	(0000 1110)
	C	=	1	

执行指令后	REG	=	F5h	(1111 0101)
	W	=	0Eh	; [二进制补码] (0000 1110)
	C	=	0	
	Z	=	0	
	N	=	1	; 结果为负

SWAPF 将 f 的高半字节和低半字节交换

语法:	SWAPF f {,d {,a}}			
操作数:	$0 \leq f \leq 255$ $d \in [0,1]$ $a \in [0,1]$			
操作:	$(f<3:0>) \rightarrow dest<7:4>$, $(f<7:4>) \rightarrow dest<3:0>$			
受影响的状态位:	无			
机器码:	0011	10da	ffff	ffff
说明:	<p>寄存器 f 的高半字节和低半字节相互交换。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f (默认)。</p> <p>如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。</p> <p>如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。</p>			
指令字数:	1			
指令周期数:	1			
Q 周期操作:				
	Q1	Q2	Q3	Q4
	译码	读寄存器 f	处理数据	写入目标寄存器

示例: SWAPF REG, 1, 0

执行指令前	REG	=	53h	
执行指令后	REG	=	35h	

TBLRD	表读				
语法:	TBLRD (*; *+; *-; +*)				
操作数:	无				
操作:	如果执行 TBLRD *, (程序存储单元 (TBLPTR)) → TABLAT ; TBLPTR 不改变; 如果执行 TBLRD *+, (程序存储单元 (TBLPTR)) → TABLAT ; (TBLPTR) + 1 → TBLPTR ; 如果执行 TBLRD *-, (程序存储单元 (TBLPTR)) → TABLAT ; (TBLPTR) - 1 → TBLPTR ; 如果执行 TBLRD +*, (TBLPTR) + 1 → TBLPTR ; (程序存储单元 (TBLPTR)) → TABLAT ;				
受影响的状态位:	无				
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>0000</td><td>0000</td><td>0000</td><td>10nn nn=0 * =1 *+ =2 *- =3 +*</td></tr> </table>	0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*
0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*		
说明:	<p>该指令用于读取程序存储单元 (P.M.) 的内容。使用表指针 (TBLPTR) 对程序存储单元进行寻址。</p> <p>TBLPTR (一个 21 位指针) 指向程序存储器中的每个字节。TBLPTR 的寻址范围为 2 MB。</p> <p>TBLPTR[0] = 0: 程序存储字的最低有效字节 TBLPTR[0] = 1: 程序存储字的最高有效字节</p> <p>TBLRD 指令可用如下方法修改 TBLPTR 的值:</p> <ul style="list-style-type: none"> 不变 后递增 后递减 预递增 				
指令字数:	1				
指令周期数:	2				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	空操作	空操作	空操作		
空操作	空操作 (读程序存储器)	空操作	空操作 (写 TABLAT)		

TBLRD	表读 (续)
例 1:	TBLRD *+ ;
执行指令前	
TABLAT	= 55h
TBLPTR	= 00A356h
存储单元 (00A356h)	= 34h
执行指令后	
TABLAT	= 34h
TBLPTR	= 00A357h
例 2:	TBLRD +* ;
执行指令前	
TABLAT	= AAh
TBLPTR	= 01A357h
存储单元 (01A357h)	= 12h
存储单元 (01A358h)	= 34h
执行指令后	
TABLAT	= 34h
TBLPTR	= 01A358h

PIC18F2XK20/4XK20

TBLWT

表写

语法: TBLWT (*, *+; *-; +*)
操作数: 无
操作:
 如果执行 TBLWT*,
 (TABLAT) → 保持寄存器;
 TBLPTR 不改变;
 如果执行 TBLWT*+,
 (TABLAT) → 保持寄存器;
 (TBLPTR) + 1 → TBLPTR ;
 如果执行 TBLWT*-,
 (TABLAT) → 保持寄存器;
 (TBLPTR) - 1 → TBLPTR ;
 如果执行 TBLWT+*,
 (TBLPTR) + 1 → TBLPTR ;
 (TABLAT) → 保持寄存器;
受影响的状态位: 无
机器码:

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*
------	------	------	---

说明: 此指令使用 TBLPTR 的低 3 位来确定要将 TABLAT 中的内容写入 8 个保持寄存器中的哪一个。该保持寄存器用于对程序存储单元 (P.M.) 的内容编程。(关于对闪存程序存储器编程的更多详细信息, 请参见第6.0节“闪存程序存储器”。)
TBLPTR (一个 21 位指针) 指向程序存储器中的每个字节。TBLPTR 的寻址范围为 2 MB。TBLPTR 的 Lsb 选择要访问程序存储单元的那个字节。

TBLPTR[0] = 0: 程序存储字的最低有效字节
TBLPTR[0] = 1: 程序存储字的最高有效字节

TBLWT 指令可用如下方法修改 TBLPTR 的值:

- 不变
- 后递增
- 后递减
- 预递增

指令字数: 1

指令周期数: 2

Q 周期操作:

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作

空操作	空操作 (读 TABLAT)	空操作	空操作 (写保持 寄存器)
-----	----------------------	-----	---------------------

TBLWT

表写 (续)

例 1:	TBLWT *+;
执行指令前	
TABLAT = 55h	
TBLPTR = 00A356h	
保持寄存器 (00A356h) = FFh	
执行指令后 (表写操作完成)	
TABLAT = 55h	
TBLPTR = 00A357h	
保持寄存器 (00A356h) = 55h	
例 2:	TBLWT +*;
执行指令前	
TABLAT = 34h	
TBLPTR = 01389Ah	
保持寄存器 (01389Ah) = FFh	
保持寄存器 (01389Bh) = FFh	
执行指令后 (表写操作完成)	
TABLAT = 34h	
TBLPTR = 01389Bh	
保持寄存器 (01389Ah) = FFh	
保持寄存器 (01389Bh) = 34h	

TSTFSZ

测试 f, 为 0 则跳过

语法:	TSTFSZ f {,a}			
操作数:	$0 \leq f \leq 255$ $a \in [0,1]$			
操作:	$f = 0$ 则跳过			
受影响的状态位:	无			
机器码:	0110	011a	ffff	ffff
说明:	<p>如果 $f = 0$, 丢弃执行当前指令过程中已取的下一条指令并执行一条 NOP 指令, 使该指令成为双周期指令。</p> <p>如果 $a = 0$, 选择快速操作存储区。如果 $a = 1$, 使用 BSR 选择 GPR 存储区。</p> <p>如果 $a = 0$ 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。</p>			
指令字数:	1			
指令周期数:	1 (2)			
注:	如果跳过, 且后面跟有2字指令, 则执行 TSTFSZ 需要 3 个周期。			
Q 周期操作:	Q1	Q2	Q3	Q4
	译码	读寄存器 f	处理数据	空操作

如果跳过:

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

如果跳过, 且后面跟有 2 字指令:

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作
空操作	空操作	空操作	空操作

示例:

```

HERE      TSTFSZ    CNT, 1
NZERO    :
ZERO     :

```

执行指令前

PC	=	地址 (HERE)
----	---	-----------

执行指令后

如果 CNT	=	00h,
PC	=	地址 (ZERO)
如果 CNT	≠	00h,
PC	=	地址 (NZERO)

XORLW

将立即数与 W 作逻辑异或运算

语法:	XORLW k			
操作数:	$0 \leq k \leq 255$			
操作:	$(W) .XOR. k \rightarrow W$			
受影响的状态位:	N 和 Z			
机器码:	0000	1010	kkkk	kkkk
说明:	将 W 的内容与 8 位立即数 k 进行逻辑异或运算。结果存储在 W 寄存器中。			
指令字数:	1			
指令周期数:	1			
Q 周期操作:	Q1	Q2	Q3	Q4
	译码	读立即数 k	处理数据	写入 W

示例: XORLW 0AFh

执行指令前
W = B5h
执行指令后
W = 1Ah

PIC18F2XK20/4XK20

XORWF

将 W 与 f 作逻辑异或运算

语法: XORWF f {,d {,a}}

操作数: $0 \leq f \leq 255$
 $d \in [0,1]$
 $a \in [0,1]$

操作: (W) .XOR.(f) \rightarrow dest

受影响的状态位: N 和 Z

机器码: 0001 10da ffff ffff

说明: 将 W 的内容与寄存器 f 的内容进行逻辑异或运算。如果 d 为 0, 结果存储在 W 中。如果 d 为 1, 结果存回寄存器 f(默认)。如果 a 为 0, 选择快速操作存储区。如果 a 为 1, 使用 BSR 选择 GPR 存储区。如果 a 为 0 且使能了扩展指令集, 只要 $f \leq 95$ (5Fh), 指令就将以立即数变址寻址模式进行操作。详情请参见第 24.2.3 节“立即数变址寻址模式中针对字节和针对位的指令”。

指令字数: 1

指令周期数: 1

Q 周期操作:

Q1	Q2	Q3	Q4
译码	读寄存器 f	处理数据	写入目标寄存器

示例: XORWF REG, 1, 0

执行指令前

REG = AFh
W = B5h

执行指令后

REG = 1Ah
W = B5h

24.2 扩展指令集

除了 PIC18 指令集的 75 条标准指令之外, PIC18F2XK20/4XK20 器件还提供了针对核心 CPU 功能的可选扩展指令。这些新增的功能包括 8 条额外的指令, 增加了间接和变址寻址操作, 并使得许多标准 PIC18 指令可以实现立即数变址寻址。

扩展指令集的额外功能在默认情况下是禁止的。用户必须通过将 XINST 配置位置 1, 才能使能它们。

扩展指令集中的指令可以全部被归为立即数操作类指令, 它们既可以对文件选择寄存器进行操作, 也可以使用这些寄存器进行变址寻址。还为其中两条指令 ADDFSR 和 SUBFSR 提供了使用 FSR2 的特例形式, 即 ADDULNK 和 SUBULNK, 这两条指令允许在执行后自动返回。

这些扩展的指令专门用于优化用高级语言特别是 C 语言编写的可重入程序代码 (也就是递归或使用软件堆栈的代码)。此外, 它们使用户能更高效地用高级语言对数据结构执行某些操作。这些操作包括:

- 在进入和退出子程序时对软件堆栈空间进行动态分配和释放
- 函数指针调用
- 对软件堆栈指针进行操作
- 对软件堆栈中的变量进行操作

表 24-3 提供了扩展指令集中的指令汇总。第 24.2.2 节“**扩展指令集**”对这些指令进行了详细说明。表 24-1 提供了标准和扩展的 PIC18 指令集的操作码字段说明。

注: 扩展指令集和立即数变址寻址模式是专为优化用 C 语言编写的应用程序而设计的, 用户可能不会在汇编程序中直接使用这些指令。对于那些需要查看编译器生成代码的用户, 这些命令的语法可作为参考。

24.2.1 扩展指令的语法

大部分扩展指令都使用变址参数, 使用一个文件选择寄存器和某一偏移量来指定源寄存器或目标寄存器。当指令的参数作为变址寻址的一部分时, 会用方括号 (“[]”) 把它括起来。这用于表示此参数用作变址或偏移量。如果 MPASM™ 汇编器发现一个变址或偏移量没有被括起来, 它就会给出出错信息。

当使能扩展指令集时, 方括号也用于表示针对字节和针对位的指令中的变址参数。这是对指令语法的额外更改。更多详细信息, 请参见第 24.2.3.1 节“**标准 PIC18 命令的扩展指令语法**”。

注: 以前, 在 PIC18 和早期的指令集中使用方括号来表示可选参数。在此文本和以后的文本中, 可选参数将用大括号 (“{ }”) 表示。

表 24-3: PIC18 指令集的扩展

助记符, 操作数	说明	周期数	16 位指令字				受影响的 状态位
			MSb	LSb			
ADDFSR f, k	将立即数加到 FSR	1	1110	1000	ffkk	kkkk	无
ADDULNK k	将立即数加到 FSR2 并返回	2	1110	1000	11kk	kkkk	无
CALLW	使用 WREG 调用子程序	2	0000	0000	0001	0100	无
MOVSF z _s , f _d	将 z _s (源) 移入 第一个字 f _d (目标) 第二个字	2	1110	1011	0zzz	zzzz	无
MOVSS z _s , z _d	将 z _s (源) 移入 第一个字 z _d (目标) 第二个字	2	1111	ffff	ffff	ffff	无
PUSHL k	将立即数保存到 FSR2, FSR2 递减 1	1	1110	1011	1zzz	zzzz	无
SUBFSR f, k	FSR 减去立即数	1	1110	1001	ffkk	kkkk	无
SUBULNK k	FSR2 减去立即数并返回	2	1110	1001	11kk	kkkk	无

PIC18F2XK20/4XK20

24.2.2 扩展指令集

ADDFSR	将立即数加到 FSR				
语法:	ADDFSR f, k				
操作数:	$0 \leq k \leq 63$ $f \in [0, 1, 2]$				
操作:	$FSR(f) + k \rightarrow FSR(f)$				
受影响的状态位:	无				
机器码:	<table border="1"><tr><td>1110</td><td>1000</td><td>ffkk</td><td>kkkk</td></tr></table>	1110	1000	ffkk	kkkk
1110	1000	ffkk	kkkk		
说明:	将 6 位立即数 k 加到由 f 指定的 FSR 的内容。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	读立即数 k	处理数据	写入 FSR		

示例: ADDFSR 2, 23h

执行指令前
FSR2 = 03FFh
执行指令后
FSR2 = 0422h

ADDULNK	将立即数加到 FSR2 并返回				
语法:	ADDULNK k				
操作数:	$0 \leq k \leq 63$				
操作:	$FSR2 + k \rightarrow FSR2$, (TOS) \rightarrow PC				
受影响的状态位:	无				
机器码:	<table border="1"><tr><td>1110</td><td>1000</td><td>11kk</td><td>kkkk</td></tr></table>	1110	1000	11kk	kkkk
1110	1000	11kk	kkkk		
说明:	将 6 位立即数 k 加到 FSR2 的内容。然后通过将 TOS 的值装入 PC, 执行 RETURN。 执行该指令需要两个周期; 在第二个周期执行一条 NOP 指令。 该指令可以被认为是 ADDFSR 指令的特例, 其中 f = 3 (二进制 “11”), 它仅针对 FSR2 进行操作。				
指令字数:	1				
指令周期数:	2				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	读立即数 k	处理数据	写入 FSR		
空操作	空操作	空操作	空操作		

示例: ADDULNK 23h

执行指令前
FSR2 = 03FFh
PC = 0100h
执行指令后
FSR2 = 0422h
PC = (TOS)

注: 所有的 PIC18 指令都可能在其指令助记符之前使用可选的标号参数, 用于符号寻址。如果使用了标号, 那么指令格式将变为: {label} 指令参数。

CALLW	使用 WREG 调用子程序												
语法:	CALLW												
操作数:	无												
操作:	(PC + 2) → TOS, (W) → PCL, (PCLATH) → PCH, (PCLATU) → PCU												
受影响的状态位:	无												
机器码:	0000 0000 0001 0100												
说明	<p>首先，返回地址 (PC + 2) 被压入返回堆栈。接下来，将 W 寄存器的内容写入 PCL、PCL 现有的值被丢弃。然后，PCLATH 和 PCLATU 的内容被分别锁存到 PCH 和 PCU。第二个周期执行一条 NOP 指令，并同时取下一条新指令。</p> <p>和 CALL 不一样，该指令没有更新 W、STATUS 或 BSR 寄存器的选项。</p>												
指令字数:	1												
指令周期数:	2												
Q 周期操作:	<table border="1"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>译码</td> <td>读 WREG</td> <td>将 PC 压入堆栈</td> <td>空操作</td> </tr> <tr> <td>空操作</td> <td>空操作</td> <td>空操作</td> <td>空操作</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读 WREG	将 PC 压入堆栈	空操作	空操作	空操作	空操作	空操作
Q1	Q2	Q3	Q4										
译码	读 WREG	将 PC 压入堆栈	空操作										
空操作	空操作	空操作	空操作										

示例: HERE CALLW

执行指令前	
PC	= 地址 (HERE)
PCLATH	= 10h
PCLATU	= 00h
W	= 06h
执行指令后	
PC	= 001006h
TOS	= 地址 (HERE + 2)
PCLATH	= 10h
PCLATU	= 00h
W	= 06h

MOVSF	将变址寻址单元内容送入 f												
语法:	MOVSF [z _s], f _d												
操作数:	0 ≤ z _s ≤ 127 0 ≤ f _d ≤ 4095												
操作:	((FSR2) + z _s) → f _d												
受影响的状态位:	无												
机器码:	<table border="1"> <tr> <td>第一个字 (源)</td> <td>0zzz</td> <td>zzzz_s</td> </tr> <tr> <td>第二个字 (目标)</td> <td>ffff</td> <td>ffff_d</td> </tr> </table>	第一个字 (源)	0zzz	zzzz _s	第二个字 (目标)	ffff	ffff _d						
第一个字 (源)	0zzz	zzzz _s											
第二个字 (目标)	ffff	ffff _d											
说明:	<p>将源寄存器的内容送入目标寄存器 f_d。通过将第一个字中的 7 位立即数偏移量 z_s 与 FSR2 的值相加来确定源寄存器的实际地址。第二个字中的 12 位立即数 f_d 指向目标寄存器的地址。两个地址均可以是 4096 字节的数据空间 (000h 到 FFFh) 中的任何存储单元。</p> <p>MOVSF 指令中的目标寄存器不能是 PCL、TOSU、TOSH 或 TOSL。</p> <p>如果计算得到的源地址指向间接寻址寄存器，将返回 00h。</p>												
指令字数:	2												
指令周期数:	2												
Q 周期操作:	<table border="1"> <thead> <tr> <th>Q1</th> <th>Q2</th> <th>Q3</th> <th>Q4</th> </tr> </thead> <tbody> <tr> <td>译码</td> <td>确定源地址</td> <td>确定源地址</td> <td>读源寄存器</td> </tr> <tr> <td>译码</td> <td>空操作</td> <td>空操作</td> <td>写寄存器 f (目标寄存器)</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	确定源地址	确定源地址	读源寄存器	译码	空操作	空操作	写寄存器 f (目标寄存器)
Q1	Q2	Q3	Q4										
译码	确定源地址	确定源地址	读源寄存器										
译码	空操作	空操作	写寄存器 f (目标寄存器)										

示例: MOVSF [05h], REG2

执行指令前	
FSR2	= 80h
85h 单元的内容	= 33h
REG2	= 11h
执行指令后	
FSR2	= 80h
85h 单元的内容	= 33h
REG2	= 33h

PIC18F2XK20/4XK20

MOVSS	在变址寻址单元之间传送数据												
语法:	MOVSS [z _s], [z _d]												
操作数:	0 ≤ z _s ≤ 127 0 ≤ z _d ≤ 127												
操作:	((FSR2) + z _s) → ((FSR2) + z _d)												
受影响的状态位:	无												
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>1110</td><td>1011</td><td>1zzz</td><td>zzzz_s</td></tr> <tr><td>1111</td><td>xxxx</td><td>xzzz</td><td>zzzz_d</td></tr> </table>	1110	1011	1zzz	zzzz _s	1111	xxxx	xzzz	zzzz _d				
1110	1011	1zzz	zzzz _s										
1111	xxxx	xzzz	zzzz _d										
说明	<p>将源寄存器的内容送入目标寄存器。通过将 FSR2 中的值分别加上 7 位立即数偏移量 z_s 和 z_d 来确定源寄存器和目标寄存器的地址。两个寄存器都可以是 4096 字节数据存储空间（000h 到 FFFh）中的任意存储单元。</p> <p>MOVSS 指令不能使用 PCL、TOSU、TOSH 或 TOSL 作为目标寄存器。</p> <p>如果计算得到的源地址指向间接寻址寄存器，将返回 00h。如果计算得到的目标地址指向间接寻址寄存器，将执行一条 NOP 指令。</p>												
指令字数:	2												
指令周期数:	2												
Q 周期操作:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> <tr> <td>译码</td><td>确定源地址</td><td>确定源地址</td><td>读源寄存器</td></tr> <tr> <td>译码</td><td>确定 目标地址</td><td>确定 目标地址</td><td>写 目标寄存器</td></tr> </table>	Q1	Q2	Q3	Q4	译码	确定源地址	确定源地址	读源寄存器	译码	确定 目标地址	确定 目标地址	写 目标寄存器
Q1	Q2	Q3	Q4										
译码	确定源地址	确定源地址	读源寄存器										
译码	确定 目标地址	确定 目标地址	写 目标寄存器										

示例: MOVSS [05h], [06h]

执行指令前	
FSR2	= 80h
85h 单元的内容	= 33h
86h 单元的内容	= 11h
执行指令后	
FSR2	= 80h
85h 单元的内容	= 33h
86h 单元的内容	= 33h

PUSHL	将立即数保存到 FSR2, FSR2 递减 1								
语法:	PUSHL k								
操作数:	0 ≤ k ≤ 255								
操作:	k → (FSR2), FSR2 - 1 → FSR2								
受影响的状态位:	无								
机器码:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>1111</td><td>1010</td><td>kkkk</td><td>kkkk</td></tr> </table>	1111	1010	kkkk	kkkk				
1111	1010	kkkk	kkkk						
说明:	<p>8 位立即数 k 被写入由 FSR2 指定的数据存储单元。操作完后 FSR2 递减 1。</p> <p>此指令允许用户将值压入软件堆栈。</p>								
指令字数:	1								
指令周期数:	1								
Q 周期操作:	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <th>Q1</th><th>Q2</th><th>Q3</th><th>Q4</th></tr> <tr> <td>译码</td><td>读取 k</td><td>处理数据</td><td>写入 目标寄存器</td></tr> </table>	Q1	Q2	Q3	Q4	译码	读取 k	处理数据	写入 目标寄存器
Q1	Q2	Q3	Q4						
译码	读取 k	处理数据	写入 目标寄存器						

示例: PUSHL 08h

执行指令前	
FSR2H:FSR2L	= 01ECh
存储单元 (01ECh)	= 00h
执行指令后	
FSR2H:FSR2L	= 01EBh
存储单元 (01ECh)	= 08h

SUBFSR	FSR 减去立即数				
语法:	SUBFSR f, k				
操作数:	$0 \leq k \leq 63$				
	$f \in [0, 1, 2]$				
操作:	$FSR(f) - k \rightarrow FSRf$				
受影响的状态位:	无				
机器码:	<table border="1"><tr><td>1110</td><td>1001</td><td>ffffkk</td><td>kkkk</td></tr></table>	1110	1001	ffffkk	kkkk
1110	1001	ffffkk	kkkk		
说明:	用 f 指定的 FSR 的内容减去 6 位立即数 k。				
指令字数:	1				
指令周期数:	1				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	读寄存器 f	处理数据	写入目标寄存器		

示例: SUBFSR 2, 23h

执行指令前
FSR2 = 03FFh

执行指令后
FSR2 = 03DCh

SUBULNK	FSR2 减去立即数并返回				
语法:	SUBULNK k				
操作数:	$0 \leq k \leq 63$				
操作:	$FSR2 - k \rightarrow FSR2$				
	$(TOS) \rightarrow PC$				
受影响的状态位:	无				
机器码:	<table border="1"><tr><td>1110</td><td>1001</td><td>11kk</td><td>kkkk</td></tr></table>	1110	1001	11kk	kkkk
1110	1001	11kk	kkkk		
说明:	用 FSR 的内容减去 6 位立即数 k, 然后通过将 TOS 的值装入 PC, 执行 RETURN。 执行该指令需要两个指令周期, 第二个指令周期执行一条 NOP 指令。 该指令可以被认为是 SUBFSR 指令的特例, 其中 $f = 3$ (二进制 "11") ; 它仅针对 FSR2 进行操作。				
指令字数:	1				
指令周期数:	2				
Q 周期操作:					
Q1	Q2	Q3	Q4		
译码	读寄存器 f	处理数据	写入目标寄存器		
空操作	空操作	空操作	空操作		

示例: SUBULNK 23h

执行指令前
FSR2 = 03FFh
PC = 0100h

执行指令后
FSR2 = 03DCh
PC = (TOS)

24.2.3 立即数变址寻址模式中针对字节和针对位的指令

注：使能 PIC18 扩展指令集可能导致常规应用程序运行不正常或完全失败。

一旦使能扩展指令集，除了可以使用扩展指令集中的 8 条新命令之外，还将使能立即数变址寻址模式（**第 5.5.1 节“使用立即数偏移量进行变址寻址”**）。这将导致标准 PIC18 指令集中大部分指令的地址解析方法有很大变化。

当禁止扩展指令集时，嵌入在操作码中的地址被视作立即数存储单元。可以是快速操作存储区中的存储单元 ($a = 0$)，或由 BSR 指定的 GPR 存储区中的存储单元 ($a = 1$)。当使能扩展指令集且 $a = 0$ 时，地址小于或等于 5Fh 的文件寄存器参数被解析为 FSR2 中的指针值的偏移量，而不是一个立即数地址。对于实际应用来说，这意味着所有使用快速操作 RAM 位作为参数的指令，即所有针对字节或针对位的指令，或者几乎半数的核心 PIC18 指令，在使能了扩展指令集时操作都会有所不同。

当 FSR2 的内容为 00h 时，快速操作 RAM 的边界会被重新映射到它们的原始值。这对于编写向下兼容的代码很有用处。如果使用此技术，有必要在 C 程序调用汇编子程序时保存 FSR2 的值并在返回时将它恢复，这样做的目的是保护堆栈指针。用户还必须记住扩展指令集的语法要求（见**第 24.2.3.1 节“标准 PIC18 命令的扩展指令语法”**）。

虽然立即数变址寻址模式对于动态堆栈和指针操作很有用处，但是如果不小心对错误的寄存器进行了简单的算术运算也会非常麻烦。已经习惯使用 PIC18 编程的用户必须记住，在使能了扩展指令集后，地址小于或等于 5Fh 的寄存器用于立即数变址寻址。

下页提供了在立即数变址寻址模式中，一些针对字节和位的指令的表示例，通过这些示例可以看出指令执行如何受到影响。示例中的操作数条件适用于所有这些类型的指令。

24.2.3.1 标准 PIC18 命令的扩展指令语法

当使能了扩展指令集时，立即数偏移量 “k” 被用来替换标准的针对字节和位的命令中的文件寄存器参数 “f”。如前所述，只有在 “f” 小于或等于 5Fh 时才会发生这种情况。当使用偏移量时，偏移量必须用方括号 “[]” 标出。因为在扩展指令集中，编译器将方括号中的值解析为变址地址或偏移量。省略方括号，或在方括号内使用大于 5Fh 的值会在 MPASM 汇编器中产生错误。

如果变址参数已被正确加上了方括号，那么就不再需要指定快速操作 RAM 参数；此参数被自动假定为 0。这与标准操作（禁止扩展指令集时）刚好相反，在标准操作中，“a” 基于目标地址被置 1。在变址寻址模式中，声明快速操作 RAM 位也将在 MPASM 汇编器中产生错误。

目标参数 “d”的操作和以前一样。

在 MPASM™ 汇编器的最新版本中，必须明确启用对扩展指令集的语言支持。可以通过命令行选项 /Y 或在源代码中加入 PE 伪指令进行启用。

24.2.4 使能扩展指令集时的注意事项

需要注意的是，并非所有用户都有必要使用扩展指令集，尤其是那些不使用软件堆栈的用户。

此外，立即数变址寻址模式可能会给写入 PIC18 汇编器的常规应用程序带来问题。这是因为常规的指令会尝试寻址快速操作存储区中地址低于 5Fh 的寄存器。当使能了扩展指令集时，这些地址被解析为相对于 FSR2 的立即数偏移量，所以应用程序会读或写错误的地址。

将应用程序移植到 PIC18F2XK20/4XK20 器件时，考虑代码的类型是非常重要的。在使用扩展指令集时，用 C 语言编写的代码较长的可重入应用程序会运行得很好，而大量使用快速操作存储区的常规应用程序不会获得任何益处。

ADDWF 将 W 与变址寻址单元的内容相加 (立即数变址寻址模式)

语法:	ADDWF [k] {d}								
操作数:	$0 \leq k \leq 95$ $d \in [0,1]$								
操作:	$(W) + ((FSR2) + k) \rightarrow dest$								
受影响的状态位:	N、OV、C、DC 和 Z								
机器码:	0010 01d0 kkkk kkkk								
说明:	将 W 的内容与由 FSR2 加上偏移量 k 指定的寄存器的内容相加。 如果 d 为 0，结果存储在 W 中。如果 d 为 1，结果存回寄存器 f (默认)。								
指令字数:	1								
指令周期数:	1								
Q 周期操作:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center; width: 25%;">Q1</th> <th style="text-align: center; width: 25%;">Q2</th> <th style="text-align: center; width: 25%;">Q3</th> <th style="text-align: center; width: 25%;">Q4</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">译码</td> <td style="text-align: center;">读取 k</td> <td style="text-align: center;">处理数据</td> <td style="text-align: center;">写入目标寄存器</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读取 k	处理数据	写入目标寄存器
Q1	Q2	Q3	Q4						
译码	读取 k	处理数据	写入目标寄存器						

示例: ADDWF [OFST], 0

执行指令前

W	=	17h
OFST	=	2Ch
FSR2	=	0A00h
0A2Ch 单元的内容	=	20h

执行指令后

W	=	37h
0A2Ch 单元的内容	=	20h

BSF 将变址寻址单元相应位置 1 (立即数变址寻址模式)

语法:	BSF [k], b								
操作数:	$0 \leq f \leq 95$ $0 \leq b \leq 7$								
操作:	$1 \rightarrow ((FSR2) + k)$								
受影响的状态位:	无								
机器码:	1000 bbb0 kkkk kkkk								
说明:	将由 FSR2 加上偏移量 k 指定的寄存器中的位 b 置 1。								
指令字数:	1								
指令周期数:	1								
Q 周期操作:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center; width: 25%;">Q1</th> <th style="text-align: center; width: 25%;">Q2</th> <th style="text-align: center; width: 25%;">Q3</th> <th style="text-align: center; width: 25%;">Q4</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">译码</td> <td style="text-align: center;">读寄存器 f</td> <td style="text-align: center;">处理数据</td> <td style="text-align: center;">写入目标寄存器</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读寄存器 f	处理数据	写入目标寄存器
Q1	Q2	Q3	Q4						
译码	读寄存器 f	处理数据	写入目标寄存器						

示例: BSF [FLAG_OFST], 7

执行指令前
 FLAG_OFST = 0Ah
 FSR2 = 0A00h
 0A0Ah 单元的内容 = 55h
 执行指令后
 0A0Ah 单元的内容 = D5h

SETF 将变址寻址单元置全 1 (立即数变址寻址模式)

语法:	SETF [k]								
操作数:	$0 \leq k \leq 95$								
操作:	$FFh \rightarrow ((FSR2) + k)$								
受影响的状态位:	无								
机器码:	0110 1000 kkkk kkkk								
说明:	将由 FSR2 加上偏移量 k 指定的寄存器的内容置为 FFh。								
指令字数:	1								
指令周期数:	1								
Q 周期操作:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center; width: 25%;">Q1</th> <th style="text-align: center; width: 25%;">Q2</th> <th style="text-align: center; width: 25%;">Q3</th> <th style="text-align: center; width: 25%;">Q4</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">译码</td> <td style="text-align: center;">读取 k</td> <td style="text-align: center;">处理数据</td> <td style="text-align: center;">写寄存器</td> </tr> </tbody> </table>	Q1	Q2	Q3	Q4	译码	读取 k	处理数据	写寄存器
Q1	Q2	Q3	Q4						
译码	读取 k	处理数据	写寄存器						

示例: SETF [OFST]

执行指令前
 OFST = 2Ch
 FSR2 = 0A00h
 0A2Ch 单元的内容 = 00h
 执行指令后
 0A2Ch 单元的内容 = FFh

24.2.5 使用 MICROCHIP MPLAB® IDE 工具的注意事项

最新版本的Microchip软件工具完全支持PIC18F2XK20/4XK20系列器件的扩展指令集。软件工具包括MPLAB C18 C语言编译器、MPASM汇编语言和MPLAB集成开发环境（Integrated Development Environment, IDE）。

在选择了目标器件进行软件开发后，MPLAB IDE 将自动按默认模式设置该器件的配置位。**XINST** 配置位的默认设置是 0，禁用扩展指令集和立即数变址寻址模式。在编程时必须将**XINST**位置 1 才能确保使用扩展指令集开发的应用程序能够正确执行。

要使用扩展指令集开发软件，用户必须设置他们的语言工具以实现对扩展指令和变址寻址模式的支持。根据所使用的环境，可以通过以下几种方法：

- 开发环境中的菜单选项或对话框，允许用户配置项目的语言工具及其设置
- 命令行选项
- 源代码中的伪指令

这些选项在不同的编译器、汇编器和开发环境中将有所不同。建议用户在其开发系统所附带的文档中查询相应的文档。

25.0 开发支持

一系列软件及硬件开发工具对 PIC® 单片机和 dsPIC® 数字信号控制器提供支持：

- 集成开发环境
 - MPLAB® IDE 软件
- 编译器 / 汇编器 / 链接器
 - 适用于各种器件系列的 MPLAB C 编译器
 - 适用于各种器件系列的 HI-TECH C 编译器
- MPASM™ 汇编器
- MPLINK™ 目标链接器 /
MPLIB™ 目标库管理器
- 适用于各种器件系列的 MPLAB 汇编器 / 链接器 / 库管理器
- 模拟器
 - MPLAB SIM 软件模拟器
- 仿真器
 - MPLAB REAL ICE™ 在线仿真器
- 在线调试器
 - MPLAB ICD 3
 - PICkit™ 3 Debug Express
- 器件编程器
 - PICkit™ 2 编程器
 - MPLAB PM3 器件编程器
- 低成本演示 / 开发板、评估工具包及入门工具包

25.1 MPLAB 集成开发环境软件

MPLAB IDE 软件为 8/16/32 位单片机市场提供了前所未有的易于使用的软件开发平台。MPLAB IDE 是基于 Windows® 操作系统的应用软件，包括：

- 一个包含所有调试工具的图形界面
 - 模拟器
 - 编程器（单独销售）
 - 在线仿真器（单独销售）
 - 在线调试器（单独销售）
 - 具有彩色上下文代码显示的全功能编辑器
 - 多项目管理器
 - 内容可直接编辑的可定制式数据窗口
 - 高级源代码调试
 - 鼠标停留在变量上进行查看的功能
 - 将变量从源代码窗口拖放到 Watch（观察）窗口
 - 丰富的在线帮助
 - 集成了可选的第三方工具，如 IAR C 编译器
- MPLAB IDE 可以让您：
- 编辑源文件（C 语言或汇编语言）
 - 点击一次即可完成编译或汇编，并将代码下载到仿真器和模拟器工具中（自动更新所有项目信息）
 - 可使用如下各项进行调试：
 - 源文件（C 语言或汇编语言）
 - 混合 C 语言和汇编语言
 - 机器码

MPLAB IDE 在单个开发范例中支持使用多种调试工具，包括从成本效益高的模拟器到低成本的在线调试器，再到全功能的仿真器。这样缩短了用户升级到更加灵活而功能强大的工具时的学习时间。

25.2 适用于各种器件系列的 MPLAB C 编译器

MPLAB C 编译器代码开发系统是完全的 ANSI C 编译器，适用于 Microchip 的 PIC18、PIC24 和 PIC32 系列单片机及 dsPIC30 和 dsPIC33 系列数字信号控制器。这些编译器提供强大的集成功能和出众的代码优化能力，且使用方便。

为便于源代码调试，编译器提供针对 MPLAB IDE 调试器优化的符号信息。

25.3 适用于各种器件系列的 HI-TECH C 编译器

HI-TECH C 编译器代码开发系统是完全的 ANSI C 编译器，适用于 Microchip 的 PIC 系列单片机及 dsPIC 系列数字信号控制器。这些编译器提供强大的集成功能和全知代码生成能力，且使用方便。

为便于源代码调试，编译器提供针对 MPLAB IDE 调试器优化的符号信息。

编译器包括一个宏汇编器、链接器、预处理程序和单步驱动程序，可以在多种平台上运行。

25.4 MPASM 汇编器

MPASM 汇编器是全功能通用宏汇编器，适用于 PIC10/12/16/18 MCU。

MPASM 汇编器可生成用于 MPLINK 目标链接器的可重定位目标文件、Intel® 标准 HEX 文件、详细描述存储器使用状况和符号参考的 MAP 文件、包含源代码行及生成机器码的绝对 LST 文件以及用于调试的 COFF 文件。

MPASM 汇编器具有如下特性：

- 集成在 MPLAB IDE 项目中
- 用户定义的宏可简化汇编代码
- 对多用途源文件进行条件汇编
- 允许完全控制汇编过程的指令

25.5 MPLINK 目标链接器 / MPLIB 目标库管理器

MPLINK 目标链接器包含了由 MPASM 汇编器、MPLAB C18 C 编译器产生的可重定位目标。通过使用链接器脚本中的指令，它还可链接预编译库中的可重定位目标。

MPLIB 目标库管理器管理预编译代码库文件的创建和修改。当从源文件调用库中的一段子程序时，只有包含此子程序的模块被链接到应用程序。这样可使大型库在许多不同应用中被高效地利用。

目标链接器 / 库管理器具有如下特性：

- 高效地连接单个的库而不是许多小文件
- 通过将相关的模块组合在一起增强代码的可维护性
- 只要列出、替换、删除和抽取模块，便可灵活地创建库

25.6 适用于各种器件系列的 MPLAB 汇编器、链接器和库管理器

MPLAB 汇编器为 PIC24、PIC32 和 dsPIC 器件从符号汇编语言生成可重定位机器码。MPLAB C 编译器使用该汇编器生成目标文件。汇编器产生可重定位目标文件之后，可将这些目标文件存档，或与其他可重定位目标文件和存档链接以生成可执行文件。该汇编器有如下显著特性：

- 支持整个器件指令集
- 支持定点数据和浮点数据
- 命令行界面
- 丰富的指令集
- 灵活的宏语言
- MPLAB IDE 兼容性

25.7 MPLAB SIM 软件模拟器

MPLAB SIM 软件模拟器通过在指令级对 PIC MCU 和 dsPIC® DSC 进行模拟，可在 PC 主机环境下进行代码开发。对于任何给定的指令，都可以对数据区进行检查或修改，并通过一个全面的激励控制器来施加激励。可以将各寄存器记录在文件中，以便进行进一步的运行时分析。跟踪缓冲区和逻辑分析器的显示使软件模拟器还能记录和跟踪程序的执行、I/O 的动作、大部分的外设及内部寄存器。

MPLAB SIM 软件模拟器完全支持使用 MPLAB C 编译器以及 MPASM 和 MPLAB 汇编器的符号调试。该软件模拟器可用于在硬件实验室环境外灵活地开发和调试代码，是一款完美且经济的软件开发工具。

25.8 MPLAB REAL ICE 在线仿真器系统

MPLAB REAL ICE 在线仿真器系统是 Microchip 针对其闪存 DSC 和 MCU 器件而推出的新一代高速仿真器。结合 MPLAB 集成开发环境（IDE）所具有的易于使用且功能强大的图形用户界面，该仿真器可对 PIC® 闪存 MCU 和 dsPIC® 闪存 DSC 进行调试和编程。IDE 是随每个工具包一起提供的。

该仿真器通过高速 USB 2.0 接口与设计工程师的 PC 相连，并利用与在线调试器系统兼容的连接器（RJ11）或新型抗噪声、高速低压差分信号（LVDS）互连电缆（CAT5）与目标板相连。

可通过 MPLAB IDE 下载将来版本的固件，对该仿真器进行现场升级。在即将推出的 MPLAB IDE 版本中，会支持许多新器件，还将增加一些新特性。在同类仿真器中，MPLAB REAL ICE 的优势十分明显：低成本、全速仿真、运行时变量查看、跟踪分析、复杂断点、耐用的探针接口及较长（长达 3 米）的互连电缆。

25.9 MPLAB ICD 3 在线调试器系统

MPLAB ICD 3 在线调试器系统是 Microchip 成本效益最高的高速硬件调试器 / 编程器，适用于 Microchip 闪存数字信号控制器（DSC）和单片机（MCU）器件。结合 MPLAB 集成开发环境（IDE）所具有的功能强大但易于使用的图形用户界面，该调试器可对 PIC® 闪存单片机和 dsPIC® DSC 进行调试和编程。

MPLAB ICD 3 在线调试器通过高速 USB 2.0 接口与设计工程师的 PC 相连，并利用与 MPLAB ICD 2 或 MPLAB REAL ICE 系统兼容的连接器（RJ-11）与目标板相连。MPLAB ICD 3 支持所有 MPLAB ICD 2 转接器。

25.10 PICkit 3 在线调试器 / 编程器及 PICkit 3 Debug Express

结合 MPLAB 集成开发环境（IDE）所具有的功能强大的图形用户界面，MPLAB PICkit 3 可对 PIC® 闪存单片机和 dsPIC® 数字信号控制器进行调试和编程，且价位较低。MPLAB PICkit 3 通过全速 USB 接口与设计工程师的 PC 相连，并利用 Microchip 调试（RJ-11）连接器（与 MPLAB ICD 3 和 MPLAB REAL ICE 兼容）与目标板相连。连接器使用两个器件 I/O 引脚和复位线来实现在线调试和在线串行编程。

PICkit 3 Debug Express 包括 PICkit 3、演示板和单片机、连接电缆和光盘（内含用户指南、课程、教程、编译器和 MPLAB IDE 软件）。

25.11 PICkit 2 开发编程器 / 调试器及 PICkit 2 Debug Express

PICkit™ 2 开发编程器 / 调试器是一款低成本开发工具，具有易于使用的界面，适用于对 Microchip 的闪存系列单片机进行编程和调试。这一全功能的 Windows® 编程界面支持低档 (PIC10F、PIC12F5xx 和 PIC16F5xx)、中档 (PIC12F6xx 和 PIC16F)、PIC18F、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）。

25.12 MPLAB PM3 器件编程器

MPLAB PM3 器件编程器是一款符合 CE 规范的通用器件编程器，在 VDDMIN 和 VDDMAX 点对其可编程电压进行校验以确保可靠性最高。它有一个用来显示菜单和错误消息的大 LCD 显示器 (128 x 64)，以及一个支持各种封装类型的可拆卸模块化插槽装置。编程器标准配置中带有一根 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。

25.13 演示 / 开发板、评估工具包及入门工具包

有许多演示、开发和评估板可用于各种 PIC MCU 和 dsPIC DSC，实现对全功能系统的快速应用开发。大多数的演示、开发和评估板都有实验布线区，供用户添加定制电路；还有应用固件和源代码，用于检查和修改。

这些板支持多种功能部件，包括 LED、温度传感器、开关、扬声器、RS-232 接口、LCD 显示器、电位计和附加 EEPROM 存储器。

演示和开发板可用于教学环境，在实验布线区设计定制电路，从而掌握各种单片机应用。

除了 PICDEM™ 和 dsPICDEM™ 演示 / 开发板系列电路外，Microchip 还有一系列评估工具包和演示软件，适用于模拟滤波器设计、KEELOQ® 数据安全产品 IC、CAN、IrDA®、PowerSmart 电池管理、SEEVVAL® 评估系统、Σ-Δ ADC、流速传感器，等等。

同时还提供入门工具包，其中包含体验指定器件功能所需的所有软硬件。通常提供单个应用以及调试功能，都包含在一块电路板上。

有关演示、开发和评估工具包的完整列表，请访问 Microchip 网站 (www.microchip.com)。

26.0 电气特性

绝对最大值 (†)

环境温度.....	-40°C 至 +125°C
储存温度.....	-65°C 至 +150°C
任一引脚（除 V _{DD} 和 MCLR 外）相对于 V _{SS} 的电压.....	-0.3V 至 (V _{DD} + 0.3V)
V _{DD} 引脚相对于 V _{SS} 的电压.....	-0.3V 至 +4.5V
MCLR 引脚相对于 V _{SS} 的电压（注 2）.....	0V 至 +11.0V
总功耗（注 1）.....	1.0W
流出 V _{SS} 引脚的最大电流（-40°C 至 +85°C）.....	300 mA
流出 V _{SS} 引脚的最大电流（+85°C 至 +125°C）.....	125 mA
流入 V _{DD} 引脚的最大电流（-40°C 至 +85°C）.....	200 mA
流入 V _{DD} 引脚的最大电流（+85°C 至 +125°C）.....	85 mA
输入钳位电流 I _{IK} （V _I < 0 或 V _I > V _{DD} ）.....	±20 mA
输出钳位电流 I _{OK} （V _O < 0 或 V _O > V _{DD} ）.....	±20 mA
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（-40°C 至 +85°C）.....	200 mA
所有端口的最大灌电流（+85°C 至 +125°C）.....	110 mA
所有端口的最大拉电流（-40°C 至 +85°C）.....	185 mA
所有端口的最大拉电流（+85°C 至 +125°C）.....	70 mA

注 1：功耗按如下公式计算：

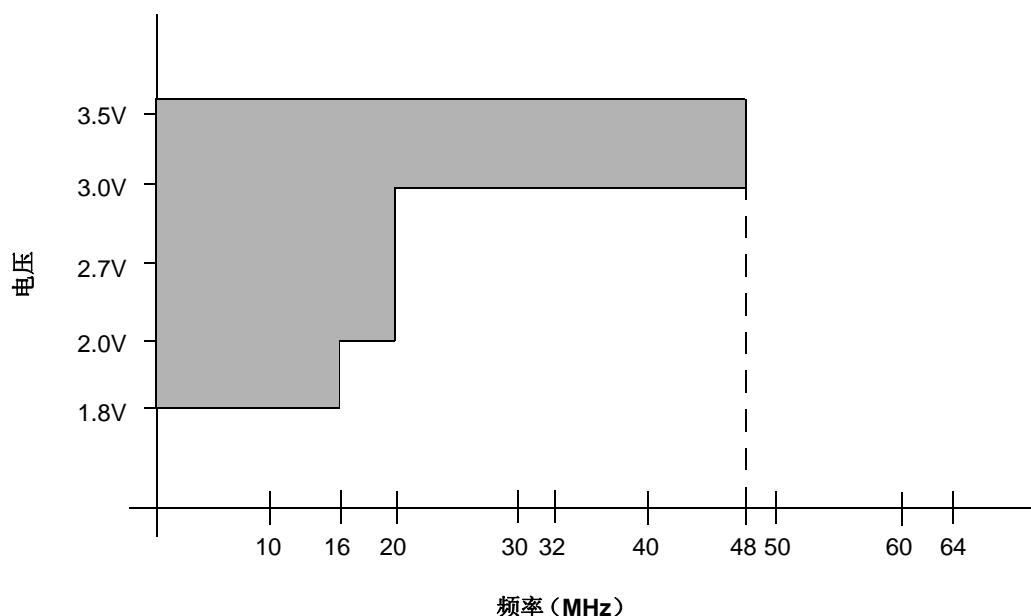
$$P_{dis} = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$$

2：如果 MCLR/VPP/RE3 引脚上的尖峰电压低于 V_{SS}，感应电流大于 80 mA，可能会引起器件锁死。因此当 MCLR/VPP/RE3 引脚驱动为低电平时，应该串联一个 50-100Ω 的电阻，而不是直接把该引脚连接到 V_{SS}。

†注：如果运行条件超过了上述“绝对最大值”，即可能对器件造成永久性损坏。这仅是极限参数，我们不建议器件工作在极限值甚至超过上述极限值。器件长时间工作在极限条件下可能会影响其可靠性。

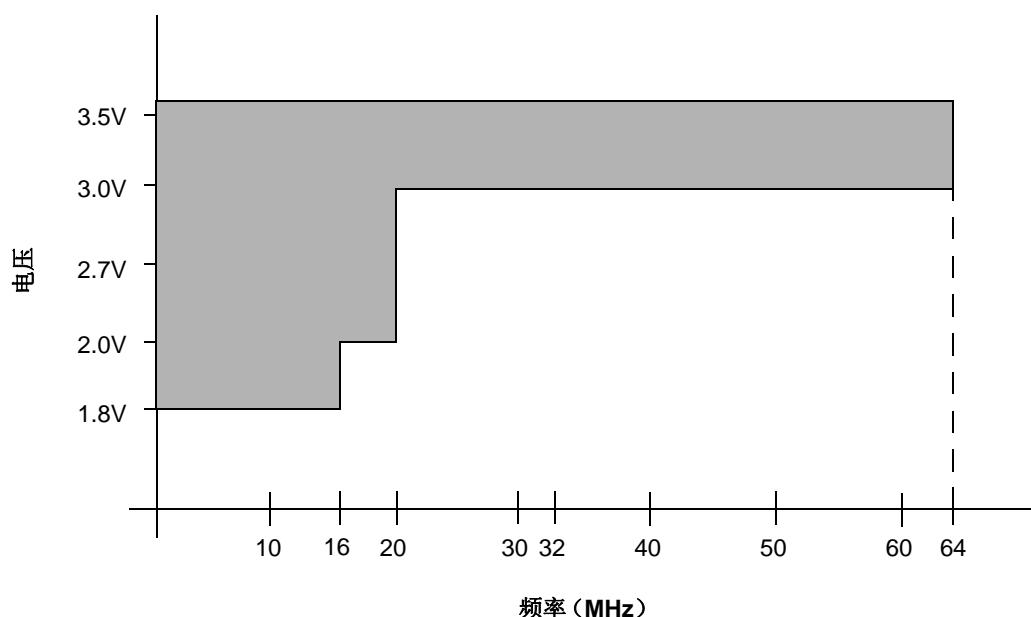
PIC18F2XK20/4XK20

图 26-1: PIC18F2XK20/4XK20 电压-频率关系图 (扩展级)



注：
最高频率 16 MHz, 1.8V 至 2.0V, -40°C 至 +125°C
最高频率 20 MHz, 2.0V 至 3.0V, -40°C 至 +125°C
最高频率 48 MHz, 3.0V 至 3.6V, -40°C 至 +125°C

图 26-2: PIC18F2XK20/4XK20 电压-频率关系图 (工业级)



注：
最高频率 16 MHz, 1.8V 至 2.0V, -40°C 至 +85°C
最高频率 20 MHz, 2.0V 至 3.0V, -40°C 至 +85°C
最高频率 64 MHz, 3.0V 至 3.6V, -40°C 至 +85°C

26.1 直流特性:

供电电压, PIC18F2XK20/4XK20

PIC18F2XK20/4XK20			标准工作条件 (除非另外声明) 工作温度 -40°C ≤ TA ≤ +125°C				
参数编号	符号	特性	最小值	典型值	最大值	单位	条件
D001	VDD	供电电压	1.8	—	3.6	V	
D002	VDR	RAM 数据保持电压 ⁽¹⁾	1.5	—	—	V	
D003	VPOR	确保内部上电复位信号的 VDD 启动电压	—	—	0.7	V	详情请参见“上电复位”章节
D004	SVDD	确保内部上电复位信号的 VDD 上升速率	0.05	—	—	V/ms	详情请参见“上电复位”章节
D005	VBOR	欠压复位电压					
BORV<1:0> = 11 ⁽²⁾			1.72	1.82	1.95	V	
BORV<1:0> = 10			2.15	2.27	2.40	V	
BORV<1:0> = 01			2.65	2.75	2.90	V	
BORV<1:0> = 00 ⁽³⁾			2.98	3.08	3.25	V	

注 1: 这是在不丢失 RAM 数据的前提下, 休眠模式或器件复位期间 VDD 所能降到的最小电压值。

2: 当 BOR 使能时, 在 BOR 发生之前支持继续工作。尽管 VDD 可能低于最小额定供电电压, 这仍是有效的。

3: 当 BOR 使能时, 在 BOR 发生之前支持全速工作 ($\text{FOSC} = 64 \text{ MHz}$)。尽管 VDD 可能低于该频率所需的小电压, 这仍是有效的。

26.2 直流特性:

掉电电流, PIC18F2XK20/4XK20

PIC18F2XK20/4XK20			标准工作条件 (除非另外声明) 工作温度 -40°C ≤ TA ≤ +125°C				
参数编号	器件特性	典型值	最大值	单位	条件		
D006	掉电电流 (IPD) ⁽¹⁾	0.05	1.0	μA	-40°C	VDD = 1.8V (休眠模式)	
		0.05	1.0	μA	+25°C		
		0.6	3.0	μA	+85°C		
		4	20	μA	+125°C		
D007		0.1	1.0	μA	-40°C	VDD = 3.0V (休眠模式)	
		0.1	1.0	μA	+25°C		
		0.7	3.0	μA	+85°C		
		5	20	μA	+125°C		

注 1: 在休眠模式下, 掉电电流与振荡器类型无关。掉电电流是在器件处于休眠模式、所有 I/O 引脚处于高阻状态并且连接到 VDD 或 VSS, 禁止所有会带来新增电流的功能部件 (如 WDT、Timer1 振荡器和 BOR 等) 时测得的。

26.3 直流特性：

RC 运行供电电流， PIC18F2XK20/4XK20

PIC18F2XK20/4XK20		标准工作条件（除非另外声明） 工作温度 -40°C ≤ TA ≤ +125°C				
参数编号	器件特性	典型值	最大值	单位	条件	
D008	供电电流 (I_{DD}) (1, 2)	5.5	9	μA	-40°C	$V_{DD} = 1.8V$ $F_{OSC} = 31\text{ kHz}$ (RC_RUN 模式, LFINTOSC 时钟源)
		6.0	10	μA	+25°C	
		6.5	14	μA	+85°C	
		9.0	30	μA	+125°C	
D008A		10.0	15	μA	-40°C	$V_{DD} = 3.0V$ $F_{OSC} = 1\text{ MHz}$ (RC_RUN 模式, HFINTOSC 时钟源)
		10.5	16	μA	+25°C	
		11.0	20	μA	+85°C	
		14.0	40	μA	+125°C	
D009		0.40	0.50	mA	-40°C 至 +125°C	$V_{DD} = 1.8V$
D009A		0.60	0.80	mA	-40°C 至 +125°C	$V_{DD} = 3.0V$
D010		2.2	3.0	mA	-40°C 至 +125°C	$V_{DD} = 1.8V$
D010A		3.8	4.4	mA	-40°C 至 +125°C	$V_{DD} = 3.0V$

注 1: 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。

2: 有效工作模式下，所有 I_{DD} 测量值的测试条件为：

所有 I/O 引脚设置为输出，驱动为 V_{SS} ；

$MCLR = V_{DD}$ ；

$OSC1 = \text{外部方波，轨到轨满幅}$ （仅限 **PRI_RUN** 和 **PRI_IDLE**）。

26.4 直流特性：

RC 空闲供电电流， PIC18F2XK20/4XK20

PIC18F2XK20/4XK20		标准工作条件（除非另外声明） 工作温度 -40°C ≤ TA ≤ +125°C					
参数编号	器件特性	典型值	最大值	单位	条件		
D011	供电电流 (IDD) (1, 2)	2.0	5	μA	-40°C	VDD = 1.8V	FOSC = 31 kHz (RC_IDLE 模式, LFINTOSC 时钟源)
		2.0	5	μA	+25°C		
		2.5	9	μA	+85°C		
		5.0	25	μA	+125°C		
D011A		3.5	8	μA	-40°C	VDD = 3.0V	
		3.5	8	μA	+25°C		
		4.0	12	μA	+85°C		
		7.0	30	μA	+125°C		
D012		0.30	0.40	mA	-40°C 至 +125°C	VDD = 1.8V	FOSC = 1 MHz (RC_IDLE 模式, HFINTOSC 时钟源)
D012A		0.40	0.60	mA	-40°C 至 +125°C	VDD = 3.0V	
D013		1.0	1.2	mA	-40°C 至 +125°C	VDD = 1.8V	FOSC = 16 MHz (RC_IDLE 模式, HFINTOSC 时钟源)
D013A		1.6	2.0	mA	-40°C 至 +125°C	VDD = 3.0V	

注 1： 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。

2： 有效工作模式下，所有 IDD 测量值的测试条件为：

所有 I/O 引脚设置为输出，驱动为 Vss；

MCLR = VDD；

OSC1 = 外部方波，轨到轨满幅（仅限 PRI_RUN 和 PRI_IDLE）。

PIC18F2XK20/4XK20

26.5 直流特性:

主运行供电电流, PIC18F2XK20/4XK20

PIC18F2XK20/4XK20		标准工作条件 (除非另外声明) 工作温度 -40°C ≤ TA ≤ +125°C					
参数编号	器件特性	典型值	最大值	单位	条件		
D014	供电电流 (IDD) (1, 2)	0.25	0.45	mA	-40°C 至 +125°C	VDD = 1.8V	FOSC = 1 MHz (PRI_RUN, EC 振荡器)
D014A		0.50	0.75	mA	-40°C 至 +125°C	VDD = 3.0V	
D015		2.7	3.2	mA	-40°C 至 +125°C	VDD = 2V	FOSC = 20 MHz (PRI_RUN, EC 振荡器)
D015A		4.3	5.0	mA	-40°C 至 +125°C	VDD = 3.0V	
D016		12.2	14.0	mA	-40°C 至 +85°C	VDD = 3.0V	FOSC = 64 MHz (PRI_RUN, EC 振荡器)
D017		2.1	2.9	mA	-40°C 至 +125°C	VDD = 1.8V	FOSC = 4 MHz 16 MHz 内部 (PRI_RUN HS+PLL)
D017A		4.2	5.0	mA	-40°C 至 +125°C	VDD = 3.0V	
D018		12.2	15.0	mA	-40°C 至 +85°C	VDD = 3.0V	FOSC = 16 MHz 64 MHz 内部 (PRI_RUN HS+PLL)

注 1: 供电电流主要受工作电压、频率和模式的影响。其他因素, 如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。

2: 有效工作模式下, 所有 IDD 测量值的测试条件为:

所有 I/O 引脚设置为输出, 驱动为 Vss ;

MCLR = VDD ;

OSC1 = 外部方波, 轨到轨满幅 (仅限 PRI_RUN 和 PRI_IDLE)。

26.6 直流特性:

主空闲供电电流, PIC18F2XK20/4XK20

PIC18F2XK20/4XK20		标准工作条件 (除非另外声明) 工作温度 -40°C ≤ TA ≤ +125°C					
参数编号	器件特性	典型值	最大值	单位	条件		
D019	供电电流 (IDD) (1, 2)	0.05	0.07	mA	-40°C 至 +125°C	VDD = 1.8V	FOSC = 1 MHz (PRI_IDLE 模式, EC 振荡器)
D019A		0.09	0.15	mA	-40°C 至 +125°C	VDD = 3.0V	
D020		1.2	1.6	mA	-40°C 至 +125°C	VDD = 2.0V	FOSC = 20 MHz (PRI_IDLE 模式, EC 振荡器)
D020A		1.8	2.5	mA	-40°C 至 +125°C	VDD = 3.0V	
D021		5.6	7.0	mA	-40°C 至 +85°C	VDD = 3.0V	FOSC = 64 MHz (PRI_IDLE 模式, EC 振荡器)

注 1: 供电电流主要受工作电压、频率和模式的影响。其他因素, 如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。

2: 有效工作模式下, 所有 IDD 测量值的测试条件为:

所有 I/O 引脚设置为输出, 驱动为 Vss ;

MCLR = VDD ;

OSC1 = 外部方波, 轨到轨满幅 (仅限 PRI_RUN 和 PRI_IDLE)。

26.7 直流特性：

辅助振荡器供电电流， PIC18F2XK20/4XK20

PIC18F2XK20/4XK20		标准工作条件（除非另外声明） 工作温度 -40°C ≤ TA ≤ +125°C					
参数编号	器件特性	典型值	最大值	单位	条件		
D022	供电电流 (IDD) (1, 2)	5.5	9	μA	-40°C	VDD = 1.8V	Fosc = 32 kHz ⁽³⁾ (SEC_RUN 模式, Timer1 作为时钟源)
		5.5	10	μA	+25°C		
		6.5	14	μA	+85°C		
D022A		10.0	15	μA	-40°C	VDD = 3.0V	
		10.0	16	μA	+25°C		
		11.0	20	μA	+85°C		
D023		2.0	5	μA	-40°C	VDD = 1.8V	Fosc = 32 kHz ⁽³⁾ (SEC_IDLE 模式, Timer1 作为时钟源)
		2.0	5	μA	+25°C		
		2.5	9	μA	+85°C		
D023A		3.5	8	μA	-40°C	VDD = 3.0V	
		3.5	8	μA	+25°C		
		4.0	12	μA	+85°C		

注 1: 供电电流主要受工作电压、频率和模式的影响。其他因素，如 I/O 引脚负载和开关速率、振荡器类型和电路、内部代码执行模式和温度也会对电流消耗产生影响。

2: 有效工作模式下，所有 IDD 测量值的测试条件为：

所有 I/O 引脚设置为输出，驱动为 Vss；

MCLR = VDD；

OSC1 = 外部方波，轨到轨满幅（仅限 PRI_RUN 和 PRI_IDLE）。

3: T1 振荡器的低功耗模式。低功耗模式最高温度限制为 85°C。

PIC18F2XK20/4XK20

26.8 直流特性： 外设供电电流， PIC18F2XK20/4XK20

PIC18F2XK20/4XK20		标准工作条件（除非另外声明） 工作温度 -40°C ≤ TA ≤ +125°C					
参数编号	器件特性	典型值	最大值	单位	条件		
模块增加电流							
D024 (ΔI _{WDT})	看门狗定时器	0.7	2.0	μA	-40°C 至 +125°C	V _{DD} = 1.8V	
		1.1	3.0	μA	-40°C 至 +125°C	V _{DD} = 3.0V	
D024A (ΔI _{BOR})	欠压复位 ⁽²⁾	21	50	μA	-40°C 至 +125°C	V _{DD} = 2.0V	
		25	60	μA	-40°C 至 +125°C	V _{DD} = 3.3V	
		0	—	μA	-40°C 至 +125°C	V _{DD} = 3.3V	休眠模式, BOREN<1:0> = 10
D024B (ΔI _{HLVD})	高 / 低压检测 ⁽²⁾	13	30	μA	-40°C 至 +125°C	V _{DD} = 1.8-3.0V	
D025 (ΔI _{OSCB}) LP	Timer1 振荡器	0.5	2.0	μA	-40°C	V _{DD} = 1.8V	Timer1 为 32 kHz ⁽¹⁾
		0.5	2.0	μA	+25°C		
		0.7	2.0	μA	+85°C		
		0.7	3.0	μA	-40°C	V _{DD} = 3.0V	Timer1 为 32 kHz ⁽¹⁾
		0.7	3.0	μA	+25°C		
		0.9	3.0	μA	+85°C		
D025A (ΔI _{OSCB}) HP	Timer1 振荡器	11	30	μA	-40°C	V _{DD} = 1.8V	Timer1 为 32 kHz ⁽³⁾
		13	33	μA	+25°C		
		15	35	μA	+85°C		
		14	33	μA	-40°C	V _{DD} = 3.0V	Timer1 为 32 kHz ⁽³⁾
		17	37	μA	+25°C		
		19	40	μA	+85°C		
D026 (ΔI _{AD}) ΔI _{FRC}	A/D 转换器 ⁽⁴⁾	200	290	μA	-40°C 至 +125°C	V _{DD} = 1.8V	A/D 开启, 但不进行转换 FRC 增加电流
		260	425	μA	-40°C 至 +125°C	V _{DD} = 3.0V	
		2	5	μA	-40°C 至 +125°C	V _{DD} = 1.8V	
		11	18	μA	-40°C 至 +125°C	V _{DD} = 3.0V	
D027 (ΔI _{COMP})	比较器	5	15	μA	-40°C 至 +125°C	V _{DD} = 1.8-3.0V	LP 模式
		40	90	μA	-40°C 至 +125°C	V _{DD} = 1.8-3.0V	HP 模式
D028 (ΔI _{CVREF})	CV _{REF}	18	40	μA	-40°C 至 +125°C	V _{DD} = 1.8V	
		32	60	μA	-40°C 至 +125°C	V _{DD} = 3.0V	

注 1: T1 振荡器为低功耗模式。低功耗模式最高温度限制为 85°C。

2: BOR 和 HLVD 使能内部带隙参考源。当同时使能两个模块时，电流消耗将少于两个规范值的和。

3: T1 振荡器为高功耗模式。

4: A/D 转换器增加电流仅适用于运行模式。在休眠或空闲模式下，只要转换（如果有）完成，ADC 和 FRC 就会关闭。

26.9 直流特性： 输入 / 输出特性， PIC18F2XK20/4XK20

直流特性			标准工作条件（除非另外声明） 工作温度 -40°C ≤ TA ≤ +125°C				
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
D030 D031 D032 D033 D033A D033B D034	V _{IL}	输入低电压 I/O 端口： 带 TTL 缓冲器 带施密特触发器	V _{SS}	—	0.15 V _D _D	V	
		MCLR	V _{SS}	—	0.2 V _D _D	V	
		OSC1	V _{SS}	—	0.3 V _D _D	V	HS 和 HSPLL 模式
		OSC1	V _{SS}	—	0.2 V _D _D	V	RC 和 EC 模式 ⁽¹⁾
		OSC1	V _{SS}	—	0.3 V _D _D	V	XT 和 LP 模式
		T13CKI	V _{SS}	—	0.3 V _D _D	V	
D040 D041 D042 D043 D043A D043B D043C D044	V _{IH}	输入高电压 I/O 端口： 带 TTL 缓冲器 带施密特触发器	0.25 V _D _D + 0.8V	—	V _D _D	V	
		MCLR	0.8 V _D _D	—	V _D _D	V	2.4V ≤ V _D _D ≤ 3.6V
			0.9 V _D _D	—	V _D _D	V	V _D _D < 2.4V
		OSC1	0.8 V _D _D	—	V _D _D	V	2.4V ≤ V _D _D ≤ 3.6V
			0.9 V _D _D	—	V _D _D	V	V _D _D < 2.4V
		OSC1	0.7 V _D _D	—	V _D _D	V	HS 和 HSPLL 模式
			0.8 V _D _D	—	V _D _D	V	EC 模式
D060 D061 D062	I _{IL}	I/O 和 MCLR 上的输入泄漏电流 ^(2,3) I/O 端口	—	5	50	nA	V _{SS} ≤ V _{PIN} ≤ V _D _D , 引脚处于高阻态
			—	10	100	nA	≤ +25°C
			—	30	200	nA	+60°C
			—	100	1000	nA	+85°C
		RA2 上的输入泄漏电流	—	10	100	nA	+125°C
			—	35	250	nA	≤ +25°C
			—	200	750	nA	+60°C
	I _{IL}	RA3 上的输入泄漏电流	—	400	2000	nA	+85°C
			—	10	80	nA	+125°C
			—	25	200	nA	≤ +25°C
			—	70	500	nA	+60°C
			—	300	1500	nA	+85°C
			—				+125°C

- 注 1: 在 RC 振荡器配置中，OSC1/CLKIN 引脚为施密特触发器输入。在 RC 模式下，建议不要使用外部时钟驱动 PIC® 器件。
- 2: MCLR 引脚上的泄漏电流主要取决于所施加的电压。规定电压为正常工作条件下的电压。在不同的输入电压下可测得更高的泄漏电流。
- 3: 负电流定义为引脚的拉电流。
- 4: 参数为特性值，未经测试。

PIC18F2XK20/4XK20

26.9 直流特性： 输入 / 输出特性， PIC18F2XK20/4XK20 (续)

直流特性			标准工作条件（除非另外声明） 工作温度 $-40^{\circ}\text{C} \leq T_{\text{A}} \leq +125^{\circ}\text{C}$				
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
D070	IPU IPURB	弱上拉电流 PORTB 弱上拉电流	50	90	400	μA	$V_{\text{DD}} = 3.0\text{V}$, $V_{\text{PIN}} = V_{\text{SS}}$

- 注 1: 在 RC 振荡器配置中, OSC1/CLKIN 引脚为施密特触发器输入。在 RC 模式下, 建议不要使用外部时钟驱动 PIC® 器件。
- 2: MCLR 引脚上的泄漏电流主要取决于所施加的电压。规定电压为正常工作条件下的电压。在不同的输入电压下可测得更高的泄漏电流。
- 3: 负电流定义为引脚的拉电流。
- 4: 参数为特性值, 未经测试。

26.9 直流特性： 输入 / 输出特性， PIC18F2XK20/4XK20（续）

直流特性			标准工作条件（除非另外声明） 工作温度 -40°C ≤ TA ≤ +125°C				
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
D080	VOL	输出低电压 I/O 端口	—	—	0.6	V	IOL = 8.5 mA, VDD = 3.0V, -40°C 至 +85°C
		OSC2/CLKOUT (RC、RCIO、EC和ECIO 模式)	—	—	0.6	V	IOL = 1.6 mA, VDD = 3.0V, -40°C 至 +85°C
D090	VOH	输出高电压 ⁽³⁾ I/O 端口	VDD - 0.7	—	—	V	IOH = -3.0 mA, VDD = 3.0V, -40°C 至 +85°C
		OSC2/CLKOUT (RC、RCIO、EC和ECIO 模式)	VDD - 0.7	—	—	V	IOH = -1.3 mA, VDD = 3.0V, -40°C 至 +85°C
D100 ⁽⁴⁾	Cosc2	输出引脚上的容性负载 规范 OSC2 引脚	—	—	15	pF	当外部时钟用于驱 动OSC1时处于XT、 HS 和 LP 模式下 满足交流时序规范
D101	CIO	所有 I/O 引脚和 OSC2 (在 RC 模式下)	—	—	50	pF	
D102	CB	SCL 和 SDA	—	—	400	pF	I ² C TM 规范

注 1: 在 RC 振荡器配置中，OSC1/CLKIN 引脚为施密特触发器输入。在 RC 模式下，建议不要使用外部时钟驱动 PIC[®] 器件。

2: MCLR 引脚上的泄漏电流主要取决于所施加的电压。规定电压为正常工作条件下的电压。在不同的输入电压下可测得更高的泄漏电流。

3: 负电流定义为引脚的拉电流。

4: 参数为特性值，未经测试。

26.10 存储器编程要求

直流特性			标准工作条件（除非另外声明） 工作温度 -40°C ≤ TA ≤ +125°C				
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
D110	VPP	内部程序存储器编程规范 ⁽¹⁾ MCLR/VPP/RE3 引脚上的电压	VDD + 4.5	—	9	V	(注 3, 注 4)
D113	IDDP	编程时的供电电流	—	—	10	mA	
D120	ED	数据 EEPROM 存储器	100K	—	—	E/W	-40°C 至 +85°C
D121	VDRW	字节耐擦写能力	1.8	—	3.6	V	使用 EECON 读 / 写
D122	TDEW	读 / 写操作时的 VDD	—	4	—	ms	
D123	TRETD	擦除 / 写周期时间	40	—	—	年	假设没有违反其他规范
D124	TREF	特性保持时间	1M	10M	—	E/W	-40°C 至 +85°C
D130	EP	刷新前的总擦除 / 写次数 ⁽²⁾	10K	—	—	E/W	-40°C 至 +85°C (注 5)
D131	VPR	单元耐擦写能力	1.8	—	3.6	V	
D132	VIW	读操作时的 VDD	2.2	—	3.6	V	
D133	TIW	行擦除或写操作时的 VDD	—	2	—	ms	
D134	TRETD	自定时写周期时间	40	—	—	年	假设没有违反其他规范

† 除非另外声明，否则“典型值”栏中的数据均为 3.0V 和 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 这些规范用于通过使用表写指令对片上程序存储器进行编程。

2: 关于数据 EEPROM 耐擦写能力的详细讨论，请参见第 7.8 节“使用数据 EEPROM”。

3: 仅当禁止单电源编程时才需要。

4: MPLAB ICD 2 不支持可变 VPP 输出。当使用 ICD 2 进行编程或调试时，用于限制 ICD 2 VPP 电压的电路必须置于 ICD 2 和目标系统之间。

5: 自写和块擦除。

26.11 模拟特性

表 26-1: 比较器规范

工作条件: $1.8V < VDD < 3.6V$, $-40^{\circ}C < TA < +125^{\circ}C$ (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
CM01	VIOFF	输入失调电压	—	12	50	mV	高功耗模式
			—	18	80	mV	低功耗模式
CM02	VICM	输入共模电压	VSS	—	VDD	V	
CM04	TRESP	响应时间	—	200	400	ns	高功耗模式
			—	300	600	ns	低功耗模式
CM05	TMC2OV	比较器模式改变到输出有效的时间*	—	—	10	μs	

* 这些参数为特性值, 未经测试。

注 1: 响应时间是在比较器的一个输入端电压为 $VDD/2$, 而另一个输入端从 VSS 跳变到 VDD 时测得的。

表 26-2: CVREF 参考电压规范

工作条件: $1.8V < VDD < 3.6V$, $-40^{\circ}C < TA < +125^{\circ}C$ (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
CV01*	CLSB	步长 (2)	— —	$VDD/24$ $VDD/32$	— —	V V	低电压范围 ($VRR = 1$) 高电压范围 ($VRR = 0$)
CV02*	CACC	绝对精度	—	—	$\pm 1/2$	LSb	
CV03*	CR	单位电阻值 (R)	—	3k	—	Ω	
CV04*	CST	稳定时间 (1)	—	7.5	10	μs	

* 这些参数为特性值, 未经测试。

注 1: 稳定时间是在 $CVRR = 1$ 并且 $CVR3:CVR0$ 从 0000 跳变到 1111 时测得的。

2: 更多信息, 请参见第 21.1 节 “比较器参考电压”。

表 26-3: 固定参考电压 (FVR) 规范

工作条件: $1.8V < VDD < 3.6V$, $-40^{\circ}C < TA < +125^{\circ}C$ (除非另外声明)。							
VR 参考电压规范			标准工作条件 (除非另外声明) 工作温度 $-40^{\circ}C \leq TA \leq +125^{\circ}C$				
参数编号	符号	特性	最小值	典型值	最大值	单位	备注
VR01	VR01	VR 电压输出	1.15	1.20	1.25	V	$-40^{\circ}C$ 至 $+85^{\circ}C$
			1.10	1.20	1.30	V	$+85^{\circ}C$ 至 $+125^{\circ}C$
VR02	TCVOUT	电压漂移温度系数	—	<50	—	ppm/°C	$-40^{\circ}C$ 至 $+40^{\circ}C$ (见图 27-30)
VR03	$\Delta VR01/\Delta VDD$	相对于 VDD 稳压值的电压漂移	—	<2000	—	μV/V	$25^{\circ}C$, 2.0 至 3.3V (见图 27-29)
VR04	TSTABLE	稳定时间	—	25	35	μs	

* 这些参数为特性值, 未经测试。

PIC18F2XK20/4XK20

图 26-3: 高 / 低压检测特性

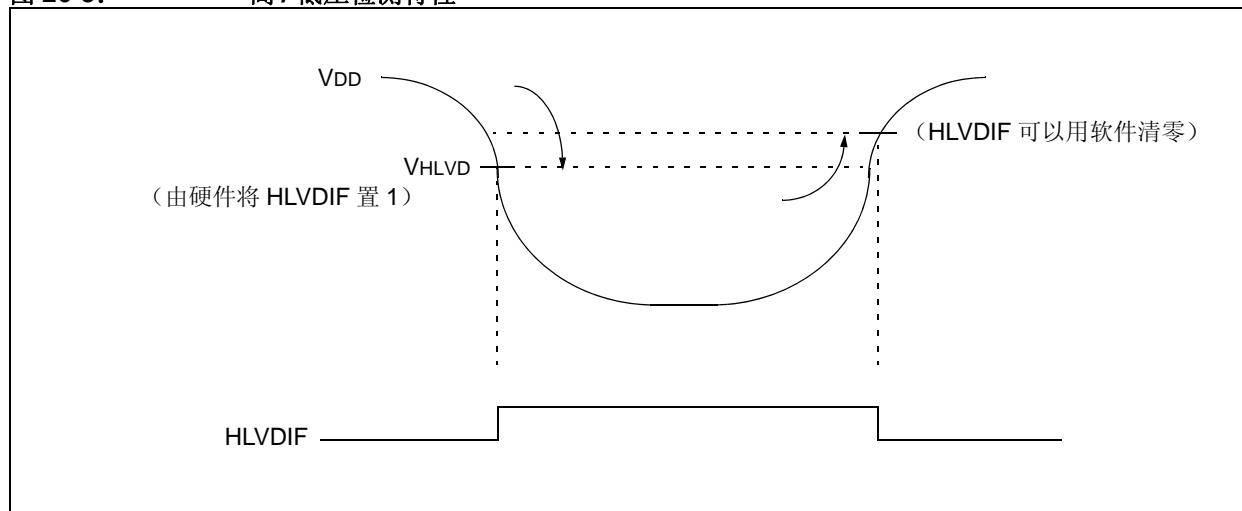


表 26-4: 高 / 低压检测特性

标准工作条件 (除非另外声明)							
工作温度 -40°C ≤ TA ≤ +125°C							
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
D420		VDD 由高转变为低时的 HLVd 电压	LVV = 0000	1.70	1.85	2.00	V
			LVV = 0001	1.80	1.95	2.10	V
			LVV = 0010	1.91	2.06	2.21	V
			LVV = 0011	2.02	2.17	2.32	V
			LVV = 0100	2.15	2.30	2.45	V
			LVV = 0101	2.22	2.37	2.52	V
			LVV = 0110	2.38	2.53	2.68	V
			LVV = 0111	2.46	2.61	2.76	V
			LVV = 1000	2.55	2.70	2.85	V
			LVV = 1001	2.65	2.80	2.95	V
			LVV = 1010	2.75	2.90	3.05	V
			LVV = 1011	2.87	3.02	3.17	V
			LVV = 1100	2.98	3.13	3.28	V
			LVV = 1101	3.26	3.41	3.56	V
			LVV = 1110	3.42	3.57	3.72	V

† TAMB = 25°C 时的生产测试。超过温度限制的规范由器件特性保证。

26.12 交流（时序）特性

26.12.1 时序参数符号体系

时序参数符号采用以下格式之一进行创建：

1. T_{ppS}2T_{ppS}

2. T_{ppS}

3. T_{CC:ST} (仅用于 I²CTM 规范)

4. T_s (仅用于 I²C 规范)

T	
F 频率	T 时间

小写字母 (pp) 及其含义：

pp			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	<u>RD</u>
cs	<u>CS</u>	rw	<u>RD</u> 或 <u>WR</u>
di	SDI	sc	SCK
do	SDO	ss	SS
dt	数据输入	t0	T0CKI
io	I/O 端口	t1	T13CKI
mc	MCLR	wr	<u>WR</u>

大写字母及其含义：

S		P	
F	下降	R	上升
H	高	V	有效
I	无效 (高阻)	Z	高阻
L	低	High	高
仅用于 I ² C		Low	低
AA	输出访问		
BUF	总线空闲		

T_{CC:ST} (仅用于 I²C 规范)

CC		SU	
HD	保持		
ST			
DAT	数据输入保持	STO	停止条件
STA	启动条件		

PIC18F2XK20/4XK20

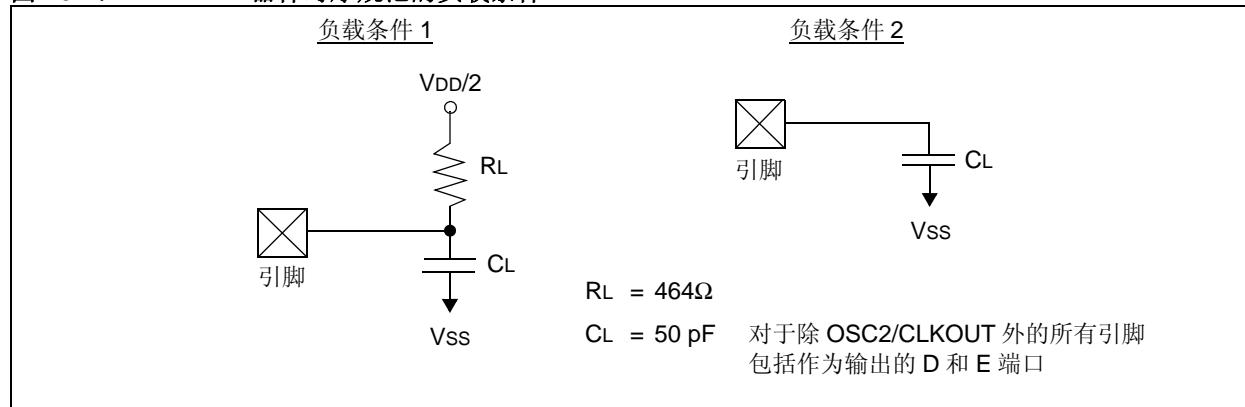
26.12.2 时序条件

表 26-5 中指定的温度和电压适用于所有的时序规范（除非另外声明）。图 26-4 规定了时序规范的负载条件。

表 26-5: 温度和电压规范——交流

交流特性	标准工作条件（除非另外声明）	
	工作温度	-40°C ≤ TA ≤ +125°C
	工作电压 VDD	范围如直流规范第 26.1 节和第 26.9 节中所述。

图 26-4: 器件时序规范的负载条件



26.12.3 时序图和规范

图 26-5: 外部时钟时序（除 PLL 外的所有模式）

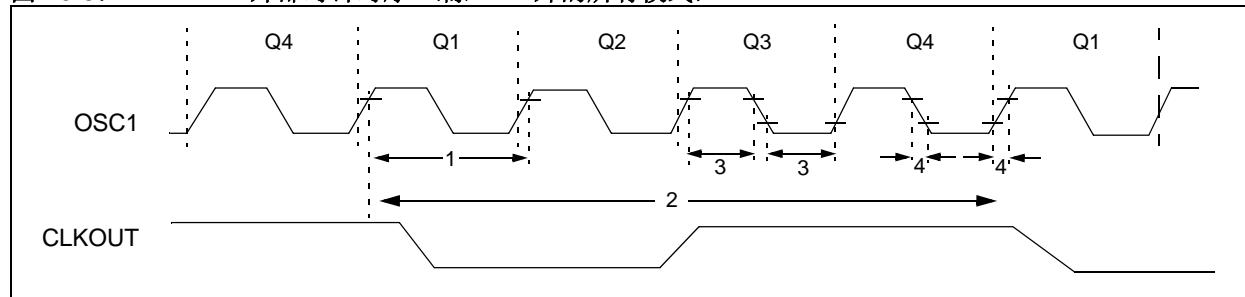


表 26-6：外部时钟时序要求

参数编号	符号	特性	最小值	最大值	单位	条件
1A	FOSC	外部 CLKIN 频率 ⁽¹⁾	DC	48	MHz	EC 和 ECIO 振荡器模式 (扩展级范围器件)
			DC	64	MHz	EC 和 ECIO 振荡器模式 (工业级范围器件)
		振荡器频率 ⁽¹⁾	DC	4	MHz	RC 振荡器模式
			0.1	4	MHz	XT 振荡器模式
			4	25	MHz	HS 振荡器模式
			4	16	MHz	HS + PLL 振荡器模式 (工业级范围器件)
			4	12	MHz	HS + PLL 振荡器模式 (扩展级范围器件)
			5	200	kHz	LP 振荡器模式
1	TOSC	外部 CLKIN 周期 ⁽¹⁾	20.8	—	ns	EC 和 ECIO 振荡器模式 (扩展级范围器件)
			15.6	—	ns	EC 和 ECIO 振荡器模式 (工业级范围器件)
		振荡周期 ⁽¹⁾	250	—	ns	RC 振荡器模式
			250	10,000	ns	XT 振荡器模式
			40	250	ns	HS 振荡器模式
			62.5	250	ns	HS + PLL 振荡器模式 (工业级范围器件)
			83.3	250	ns	HS + PLL 振荡器模式 (扩展级范围器件)
			5	200	μs	LP 振荡器模式
2	TCY	指令周期 ⁽¹⁾	62.5	—	ns	TCY = 4/FOSC
3	TosL, TosH	外部时钟输入 (OSC1) 的高电平或低电平时间	30	—	ns	XT 振荡器模式
			2.5	—	μs	LP 振荡器模式
			10	—	ns	HS 振荡器模式
4	TosR, TosF	外部时钟输入 (OSC1) 的上升或下降时间	—	20	ns	XT 振荡器模式
			—	50	ns	LP 振荡器模式
			—	7.5	ns	HS 振荡器模式

注 1：对于除 PLL 外的所有配置，指令周期 (TCY) 等于输入振荡器时基周期的四倍。所有规定值均为基于针对特定振荡器类型，器件在标准工作条件下执行代码时的特性数据。超出这些规定的限定值，可能导致振荡器运行不稳定和 / 或导致电流消耗超出预期值。所有器件在测试“最小值”时，都在 OSC1/CLKIN 引脚连接了外部时钟。当使用了外部时钟输入时，所有器件的“最大值”周期时间限制为“DC”（无时钟）。

PIC18F2XK20/4XK20

表 26-7: PLL 时钟时序规范 ($V_{DD} = 1.8V$ 至 $3.6V$)

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
F10	Fosc	振荡器频率范围	4	—	4	MHz	$V_{DD} = 1.8\text{-}2.0V$
			4	—	5	MHz	$V_{DD} = 2.0\text{-}3.0V$
			4	—	16	MHz	$V_{DD} = 3.0\text{-}3.6V$, 工业级范围器件
			4	—	12	MHz	$V_{DD} = 3.0\text{-}3.6V$, 扩展级范围器件
F11	Fsys	片上 VCO 系统频率	16	—	16	MHz	$V_{DD} = 1.8\text{-}2.0V$
			16	—	20	MHz	$V_{DD} = 2.0\text{-}3.0V$
			16	—	64	MHz	$V_{DD} = 3.0\text{-}3.6V$, 工业级范围器件
			16	—	48	MHz	$V_{DD} = 3.0\text{-}3.6V$, 扩展级范围器件
F12	t_{rc}	PLL 起振时间 (锁定时间)	—	—	2	ms	
F13	ΔCLK	CLKOUT 稳定性 (抗抖动性)	-2	—	+2	%	

表 26-8: 交流特性: PIC18F2XK20/4XK20 内部振荡器精度

PIC18F2XK20/4XK20		标准工作条件 (除非另外声明) 工作温度 $-40^{\circ}\text{C} \leq T_A \leq +125^{\circ}\text{C}$					
参数编号		最小值	典型值	最大值	单位	条件	
OA1	在频率为 16 MHz、8 MHz、4 MHz、2 MHz、1 MHz、500 kHz 和 250 kHz 时的 HFINTOSC 精度 ⁽¹⁾		-2	0	+2	%	+0°C 至 +85°C $V_{DD} = 1.8\text{-}3.6V$
	-5	—	+5	%	-40°C 至 0°C 和 +85°C 至 125°C	$V_{DD} = 1.8\text{-}3.6V$	
	在频率为 31.25 kHz 时的 LFINTOSC 精度		-15	—	+15	%	-40°C 至 +125°C $V_{DD} = 1.8\text{-}3.6V$

注 1: 频率校准温度为 25°C。OSCTUNE 寄存器可用于补偿温度漂移。

图 26-6: CLKOUT 和 I/O 时序

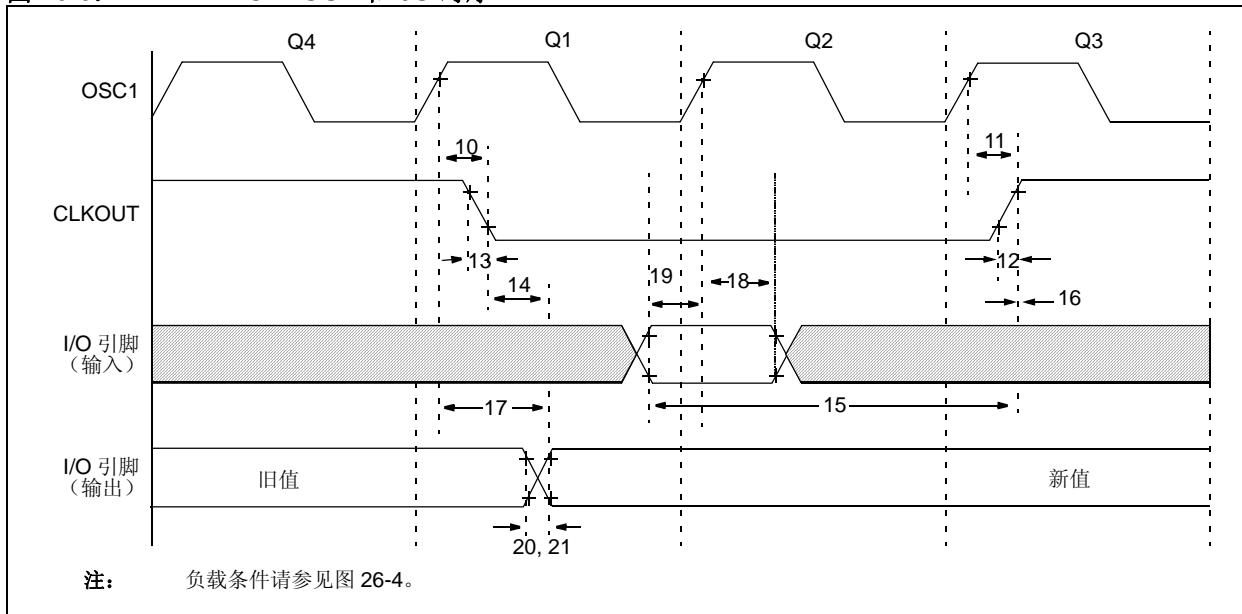


表 26-9: CLKOUT 和 I/O 时序要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
10	TosH2ckL	OSC1↑ 到 CLKOUT↓ 的时间	—	75	200	ns	(注 1)
11	TosH2ckH	OSC1↑ 到 CLKOUT↑ 的时间	—	75	200	ns	(注 1)
12	TckR	CLKOUT 上升时间	—	35	100	ns	(注 1)
13	TckF	CLKOUT 下降时间	—	35	100	ns	(注 1)
14	TckL2ioV	CLKOUT ↓ 到端口输出有效的时间	—	—	0.5 TCY + 20	ns	(注 1)
15	TioV2ckH	CLKOUT ↑ 之前端口输入有效的时间	0.25 TCY + 25	—	—	ns	(注 1)
16	TckH2iol	CLKOUT ↑ 之后端口输入保持的时间	0	—	—	ns	(注 1)
17	TosH2ioV	OSC1 ↑ (Q1 周期) 到端口输出有效的时间	—	50	150	ns	
18	TosH2iol	OSC1 ↑ (Q2 周期) 到端口输入无效的时间 (I/O 输入保持时间)	100	—	—	ns	
19	TioV2osH	端口输入有效到 OSC1 ↑ 的时间 (I/O 输入建立时间)	0	—	—	ns	
20	TioR	端口输出上升时间	—	10	25	ns	
21	TioF	端口输出下降时间	—	10	25	ns	
22†	TiNP	INTx 引脚高电平或低电平时间	20	—	—	ns	
23†	TRBP	RB<7:4> 电平变化中断 KBIx 高电平或低电平时间	TCY	—	—	ns	

† 这些参数是与任何内部时钟边沿无关的异步事件。

注 1: 测量是在 RC 模式下进行的，其中 CLKOUT 输出为 $4 \times Tosc$ 。

PIC18F2XK20/4XK20

图 26-7：复位、看门狗定时器、振荡器起振定时器和上电延时定时器时序

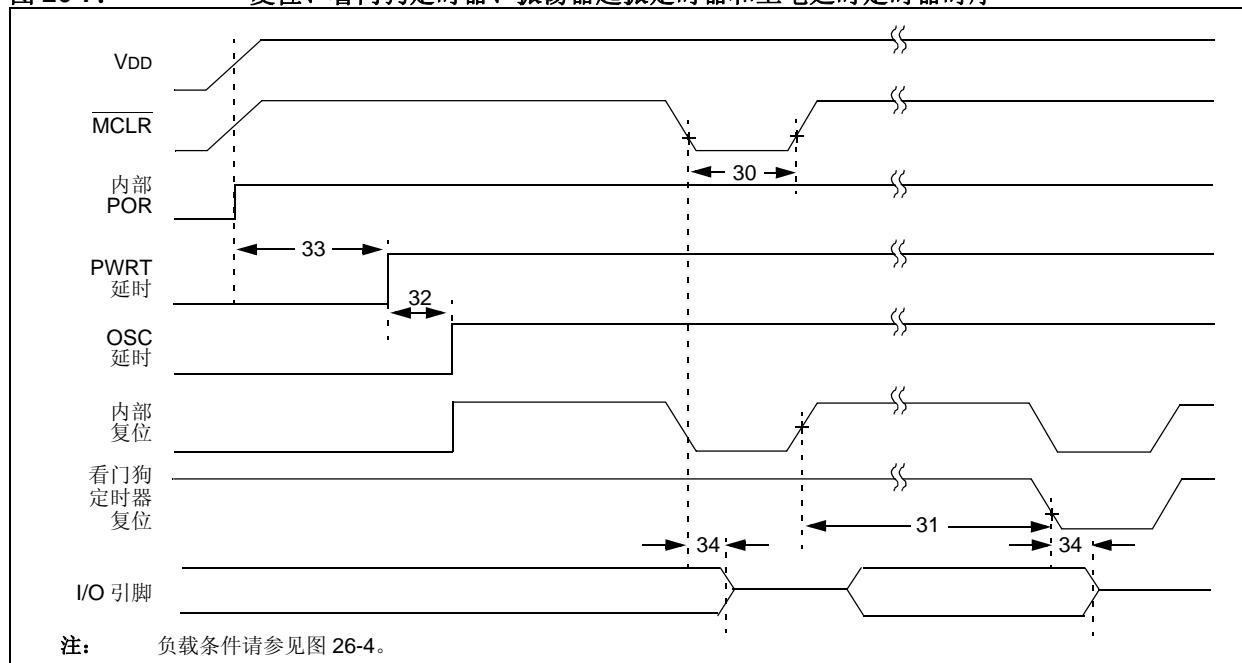


图 26-8：欠压复位时序

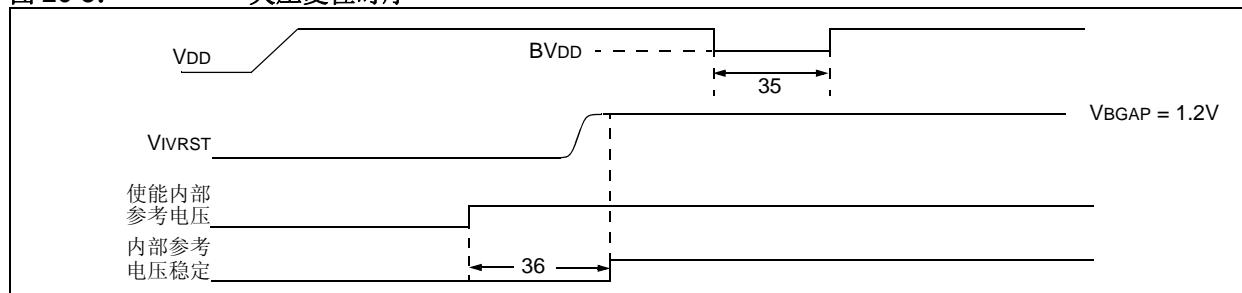


表 26-10：复位、看门狗定时器、振荡器起振定时器、上电延时定时器和欠压复位要求

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
30	T _{mclL}	MCLR 脉冲宽度 (低电平)	2	—	—	μs	
31	T _{WDT}	看门狗定时器超时周期 (无后分频器)	3.5	4.1	4.7	ms	1:1 预分频比
32	T _{OSEN}	振荡器起振定时器周期	1024 T _{osc}	—	1024 T _{osc}	—	T _{osc} = OSC1 周期
33	T _{PWRT}	上电延时定时器周期	54.8	64.4	74.1	ms	
34	T _{IOZ}	自 MCLR 低电平或看门狗定时器复位起 I/O 处于高阻态的时间	—	2	—	μs	
35	T _{BOR}	欠压复位脉冲宽度	200	—	—	μs	$VDD \leq BVDD$ (见 D005)
36	T _{IVRST}	内部参考电压稳定时间	—	25	35	μs	
37	T _{HLVD}	高 / 低压检测脉冲宽度	200	—	—	μs	$VDD \leq VHLVD$
38	T _{CSD}	CPU 启动时间	5	—	10	μs	
39	T _{IOBOST}	HF-INTOSC 稳定时间	—	0.25	1	ms	

图 26-9: TIMER0 和 TIMER1 外部时钟时序

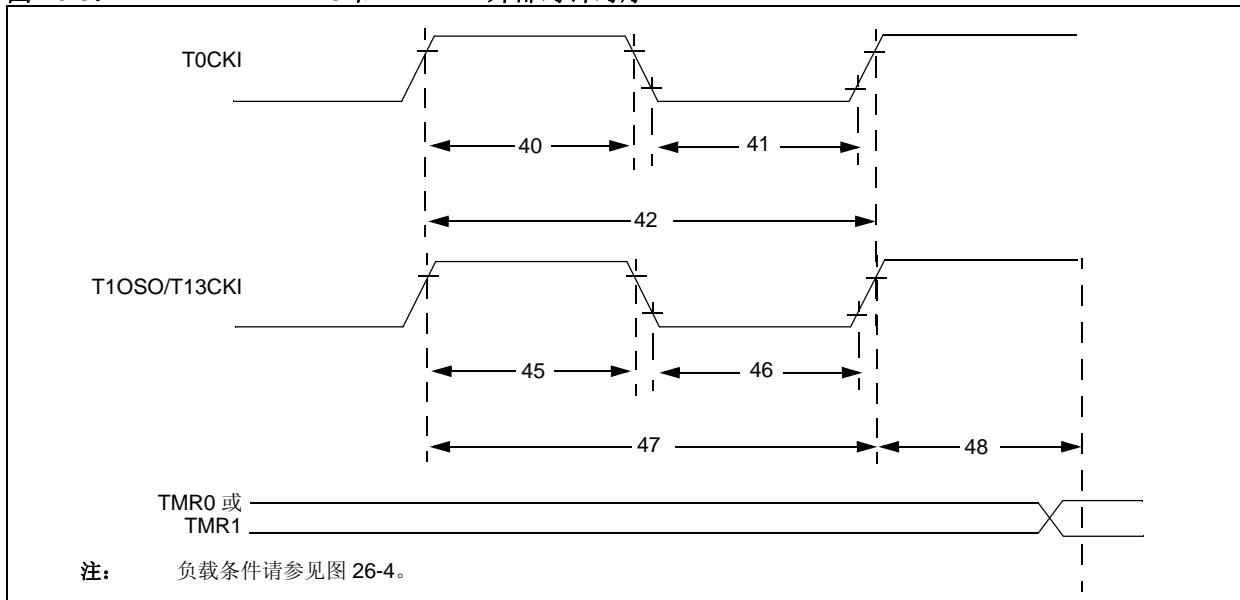
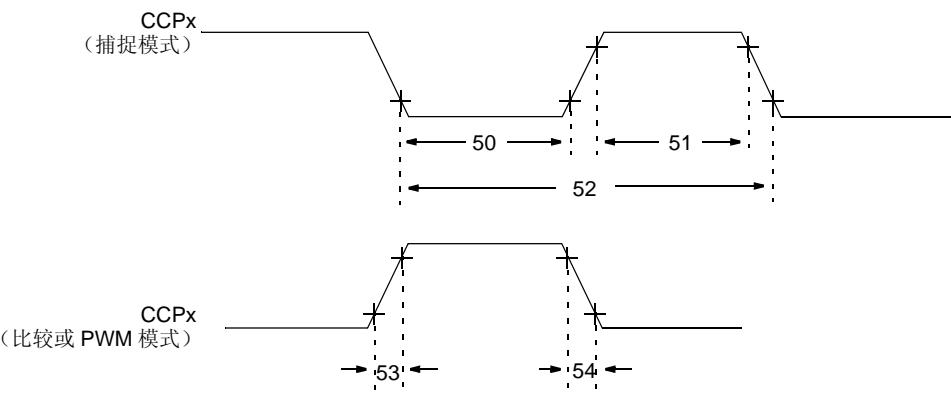


表 26-11: TIMER0 和 TIMER1 外部时钟要求

参数编号	符号	特性		最小值	最大值	单位	条件
40	Tt0H	T0CKI 高电平脉冲宽度	无预分频器	0.5 TCY + 20	—	ns	
			带预分频器	10	—	ns	
41	Tt0L	T0CKI 低电平脉冲宽度	无预分频器	0.5 TCY + 20	—	ns	
			带预分频器	10	—	ns	
42	Tt0P	T0CKI 周期	无预分频器	TCY + 10	—	ns	$N = \text{预分频值}$ (1, 2, 4,..., 256)
			带预分频器	取如下二者中较大值: 20 ns 或 $(TCY + 40)/N$	—	ns	
45	Tt1H	T13CKI 高电平时间	同步, 无预分频器	0.5 TCY + 20	—	ns	
			同步, 带预分频器	10	—	ns	
			异步	30	—	ns	
46	Tt1L	T13CKI 低电平时间	同步, 无预分频器	0.5 TCY + 5	—	ns	
			同步, 带预分频器	10	—	ns	
			异步	30	—	ns	
47	Tt1P	T13CKI 输入周期	同步	取如下二者中较大值: 20 ns 或 $(TCY + 40)/N$	—	ns	$N = \text{预分频值}$ (1, 2, 4 和 8)
			异步	60	—	ns	
	Ft1	T13CKI 振荡器输入频率范围	DC	50	kHz		
48	Tcke2tmrl	从外部 T13CKI 时钟边沿到定时器递增的延时	2Tosc	7Tosc	—		

PIC18F2XK20/4XK20

图 26-10: 捕捉 / 比较 /PWM 时序 (所有 CCP 模块)



注：负载条件请参见图 26-4。

表 26-12: 捕捉 / 比较 /PWM 要求 (所有 CCP 模块)

参数编号	符号	特性	最小值	最大值	单位	条件
50	TccL	CCPx 输入低电平时间	无预分频器 0.5 TCY + 20	—	ns	
		带预分频器	10	—	ns	
51	TccH	CCPx 输入高电平时间	无预分频器 0.5 TCY + 20	—	ns	
		带预分频器	10	—	ns	
52	TccP	CCPx 输入周期	$\frac{3 \text{ TCY} + 40}{N}$	—	ns	$N = \text{预分频值}$ (1、4 或 16)
53	TccR	CCPx 输出上升时间	—	25	ns	
54	TccF	CCPx 输出下降时间	—	25	ns	

图 26-11: 并行从端口时序 (PIC18F4XK20)

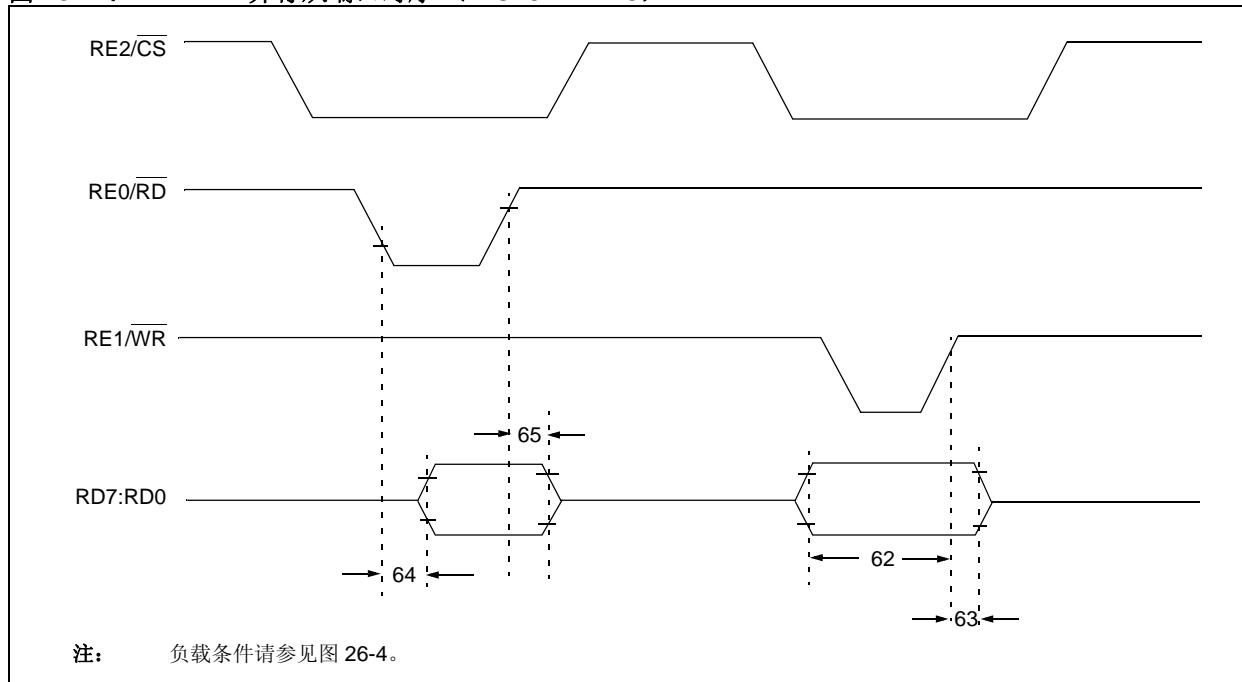


表 26-13: 并行从端口要求 (PIC18F4XK20)

参数编号	符号	特性	最小值	最大值	单位	条件
62	TdtV2wrH	WR↑或CS↑之前的数据输入有效时间（建立时间）	20	—	ns	
63	TwrH2dtl	WR↑或CS↑到数据输入无效的时间（保持时间）	20	—	ns	
64	TrdL2dtV	RD↓和CS↓到数据输出有效的时间	—	80	ns	
65	TrdH2dtl	RD↑或CS↓到数据输出无效的时间	10	30	ns	
66	TibfINH	禁止IBF标志位被WR↑或CS↑清零的时间	—	3 TCY		

PIC18F2XK20/4XK20

图 26-12: SPI 主模式时序示例 (CKE = 0)

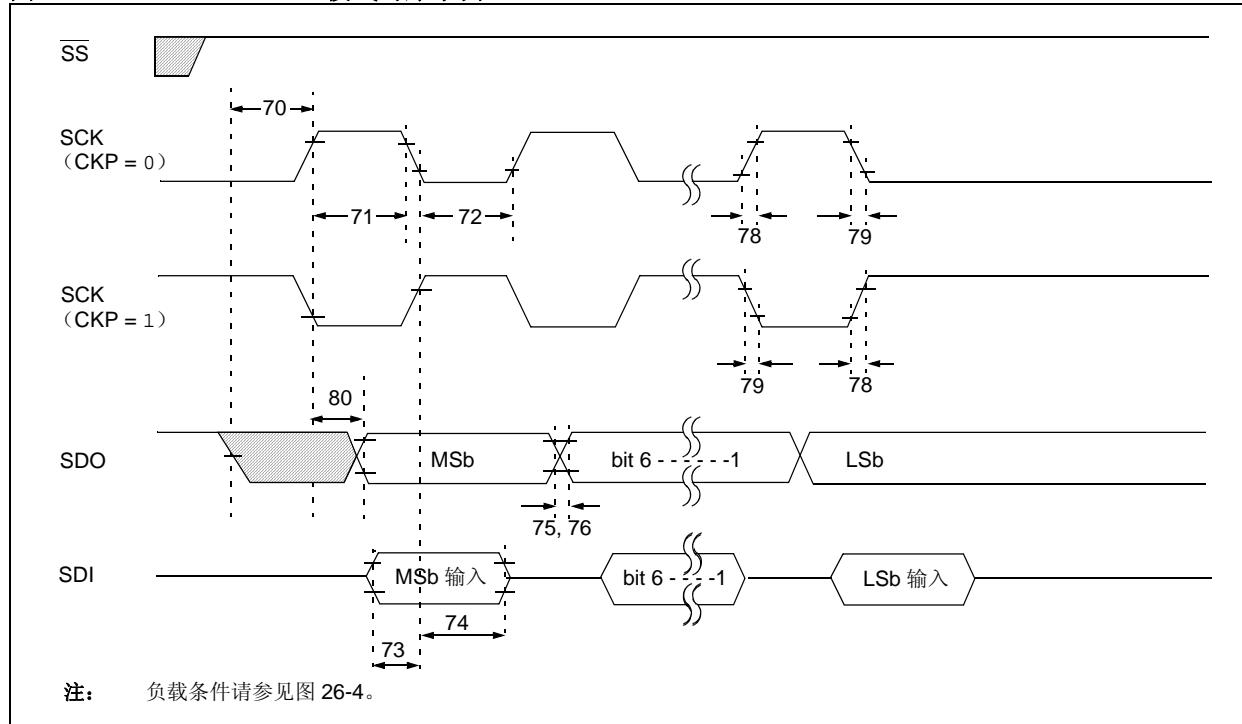


表 26-14: SPI 模式要求示例 (主模式, CKE = 0)

参数编号	符号	特性		最小值	最大值	单位	条件
70	TssL2scH, TssL2scL	\overline{SS} ↓ 到 SCK ↓ 或 SCK ↑ 输入的时间		TCY	—	ns	
71 71A	TscH	SCK 输入高电平时间 (从模式)	连续	1.25 TCY + 30	—	ns	
			单字节	40	—	ns	(注 1)
72 72A	TscL	SCK 输入低电平时间 (从模式)	连续	1.25 TCY + 30	—	ns	
			单字节	40	—	ns	(注 1)
73	TdiV2scH, TdiV2scL	SDI 数据输入到 SCK 边沿的建立时间		100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间		1.5 TCY + 40	—	ns	(注 2)
74	TscH2diL, TscL2diL	SDI 数据输入到 SCK 边沿的保持时间		100	—	ns	
75	TdoR	SDO 数据输出上升时间		—	25	ns	
76	TdoF	SDO 数据输出下降时间		—	25	ns	
78	TscR	SCK 输出上升时间 (主模式)		—	25	ns	
79	TscF	SCK 输出下降时间 (主模式)		—	25	ns	
80	TscH2doV, TscL2doV	SCK 边沿之后 SDO 数据输出有效的时间		—	50	ns	

注 1: 要求使用参数 73A。

2: 仅当使用参数 71A 和 72A 时。

图 26-13: SPI 主模式时序示例 (CKE = 1)

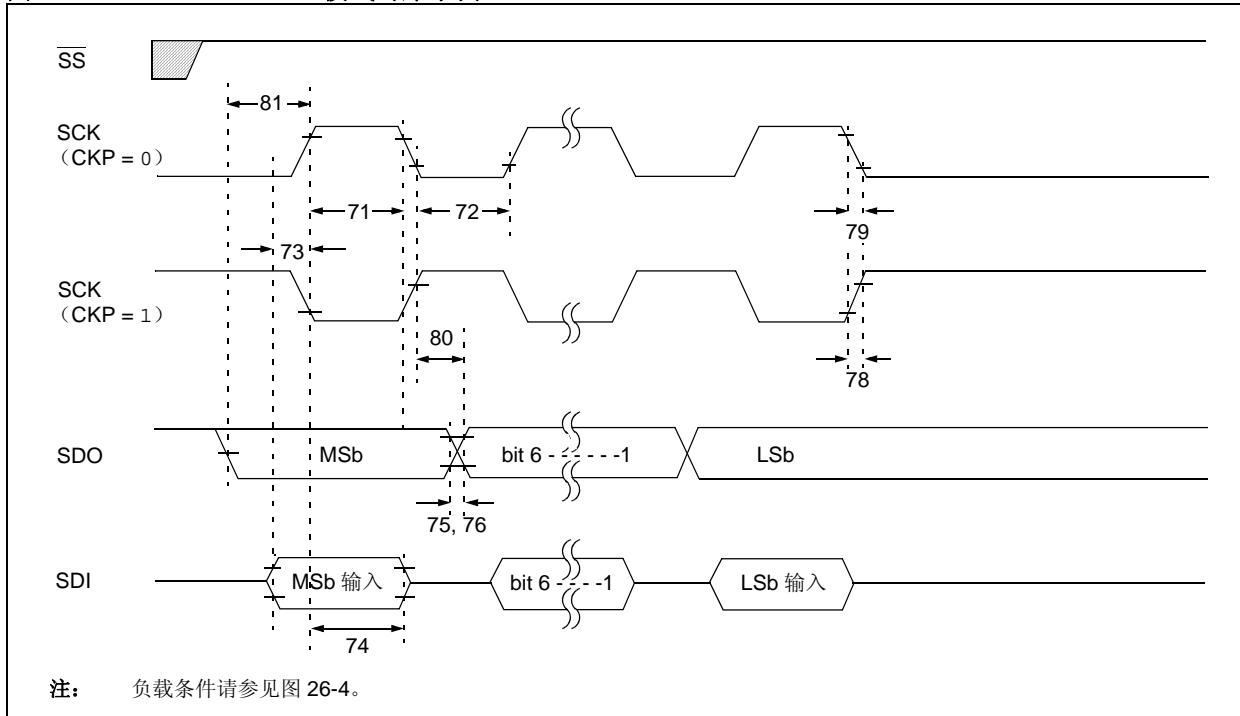


表 26-15: SPI 模式要求示例 (主模式, CKE = 1)

参数编号	符号	特性	最小值	最大值	单位	条件
71 71A	Tsch	SCK 输入高电平时间 (从模式)	连续	1.25 Tcy + 30	—	ns
			单字节	40	—	ns (注 1)
72 72A	TscL	SCK 输入低电平时间 (从模式)	连续	1.25 Tcy + 30	—	ns
			单字节	40	—	ns (注 1)
73	TdiV2scH, TdiV2scL	SDI 数据输入到 SCK 边沿的建立时间	100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间	1.5 Tcy + 40	—	ns	(注 2)
74	Tsch2diL, TscL2diL	SDI 数据输入到 SCK 边沿的保持时间	100	—	ns	
75	TdoR	SDO 数据输出上升时间	—	25	ns	
76	TdoF	SDO 数据输出下降时间	—	25	ns	
78	TscR	SCK 输出上升时间 (主模式)	—	25	ns	
79	TscF	SCK 输出下降时间 (主模式)	—	25	ns	
80	Tsch2doV, TscL2doV	SCK 边沿之后 SDO 数据输出有效的时间	—	50	ns	
81	TdoV2scH, TdoV2scL	SDO 数据输出建立到出现 SCK 边沿的时间	Tcy	—	ns	

注 1: 要求使用参数 73A。

2: 仅当使用参数 71A 和 72A 时。

PIC18F2XK20/4XK20

图 26-14: SPI 从模式时序示例 ($CKE = 0$)

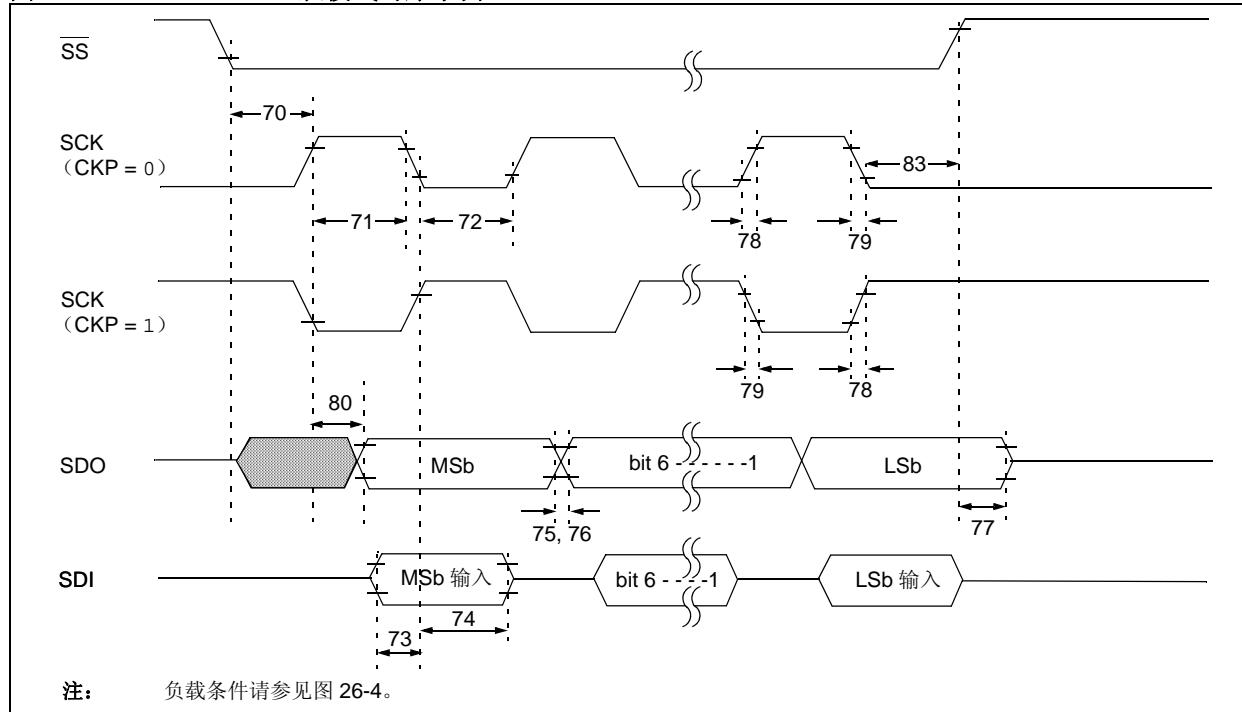


表 26-16: SPI 模式要求示例 (从模式时序, $CKE = 0$)

参数编号	符号	特性	最小值	最大值	单位	条件
70	TssL2scH, TssL2scL	SS ↓ 到 SCK ↓ 或 SCK ↑ 输入的时间	TCY	—	ns	
71 71A	TscH	SCK 输入高电平时间 (从模式)	连续	1.25 TCY + 30	—	ns
			单字节	40	—	ns (注 1)
72 72A	TscL	SCK 输入低电平时间 (从模式)	连续	1.25 TCY + 30	—	ns
			单字节	40	—	ns (注 1)
73	TdiV2scH, TdiV2scL	SDI 数据输入到 SCK 边沿的建立时间	100	—	ns	
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿之间的时间	1.5 TCY + 40	—	ns	(注 2)
74	TscH2diL, TscL2diL	SDI 数据输入到 SCK 边沿的保持时间	100	—	ns	
75	TdoR	SDO 数据输出上升时间	—	25	ns	
76	TdoF	SDO 数据输出下降时间	—	25	ns	
77	TssH2doZ	SS↑ 到 SDO 输出高阻态的时间	10	50	ns	
78	TscR	SCK 输出上升时间 (主模式)	—	25	ns	
79	TscF	SCK 输出下降时间 (主模式)	—	25	ns	
80	TscH2doV, TscL2doV	SCK 边沿之后 SDO 数据输出有效的时间	—	50	ns	
83	TscH2ssH, TscL2ssH	SCK 边沿之后出现 SS↑ 的时间	1.5 TCY + 40	—	ns	

注 1: 要求使用参数 73A。

2: 仅当使用参数 71A 和 72A 时。

图 26-15: SPI 从模式时序示例 (CKE = 1)

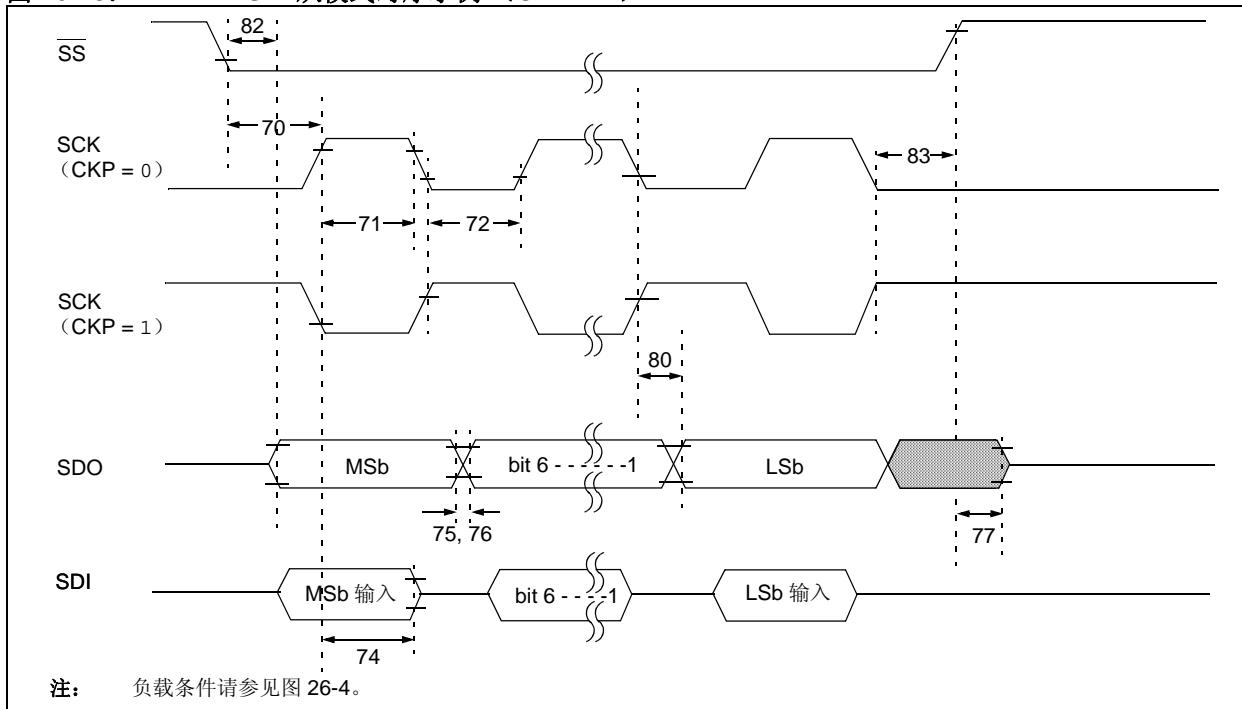


表 26-17: SPI 从模式要求示例 (CKE = 1)

参数编号	符号	特性	最小值	最大值	单位	条件
70	TssL2scH, TssL2scL	SS ↓ 到 SCK ↓ 或 SCK ↑ 输入的时间	TCY	—	ns	
71 71A	TscH	SCK 输入高电平时间 (从模式)	连续	1.25 TCY + 30	—	ns
		单字节	40	—	ns	(注 1)
72 72A	TscL	SCK 输入低电平时间 (从模式)	连续	1.25 TCY + 30	—	ns
		单字节	40	—	ns	(注 1)
73A	Tb2b	字节 1 的最后一个时钟边沿到字节 2 的第一个时钟边沿 之间的时间	1.5 TCY + 40	—	ns	(注 2)
74	TscH2diL, TscL2diL	SDI 数据输入到 SCK 边沿的保持时间	100	—	ns	
75	TdoR	SDO 数据输出上升时间	—	25	ns	
76	TdoF	SDO 数据输出下降时间	—	25	ns	
77	TssH2doZ	SS↑ 到 SDO 输出高阻态的时间	10	50	ns	
78	TscR	SCK 输出上升时间 (主模式)	—	25	ns	
79	TscF	SCK 输出下降时间 (主模式)	—	25	ns	
80	TscH2doV, TscL2doV	SCK 边沿之后 SDO 数据输出有效的时间	—	50	ns	
82	TssL2doV	SS ↓ 边沿之后 SDO 数据输出有效的时间	—	50	ns	
83	TscH2ssH, TscL2ssH	SCK 边沿之后出现 SS↑ 的时间	1.5 TCY + 40	—	ns	

注 1: 要求使用参数 73A。

2: 仅当使用参数 71A 和 72A 时。

PIC18F2XK20/4XK20

图 26-16: I²CTM 总线启动位 / 停止位时序

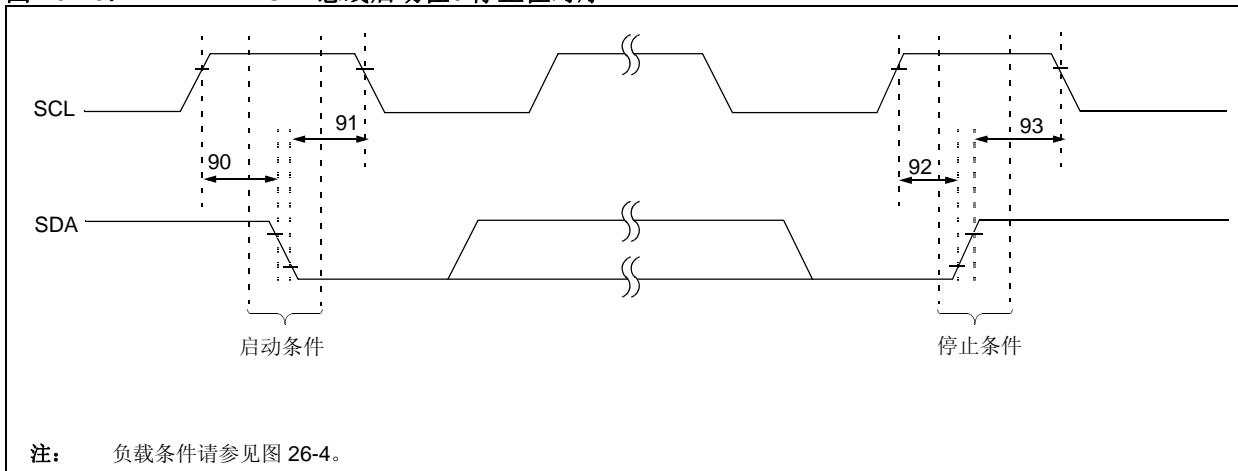


表 26-18: I²CTM 总线启动位 / 停止位要求（从模式）

参数编号	符号	特性	最小值	最大值	单位	条件
90	TSU:STA	启动条件	100 kHz 模式	4700	—	ns 仅与重复启动条件相关
		建立时间	400 kHz 模式	600	—	
91	THD:STA	启动条件	100 kHz 模式	4000	—	ns 这个周期后产生第一个时钟脉冲
		保持时间	400 kHz 模式	600	—	
92	TSU:STO	停止条件	100 kHz 模式	4700	—	ns
		建立时间	400 kHz 模式	600	—	
93	THD:STO	停止条件	100 kHz 模式	4000	—	ns
		保持时间	400 kHz 模式	600	—	

图 26-17: I²CTM 总线数据时序

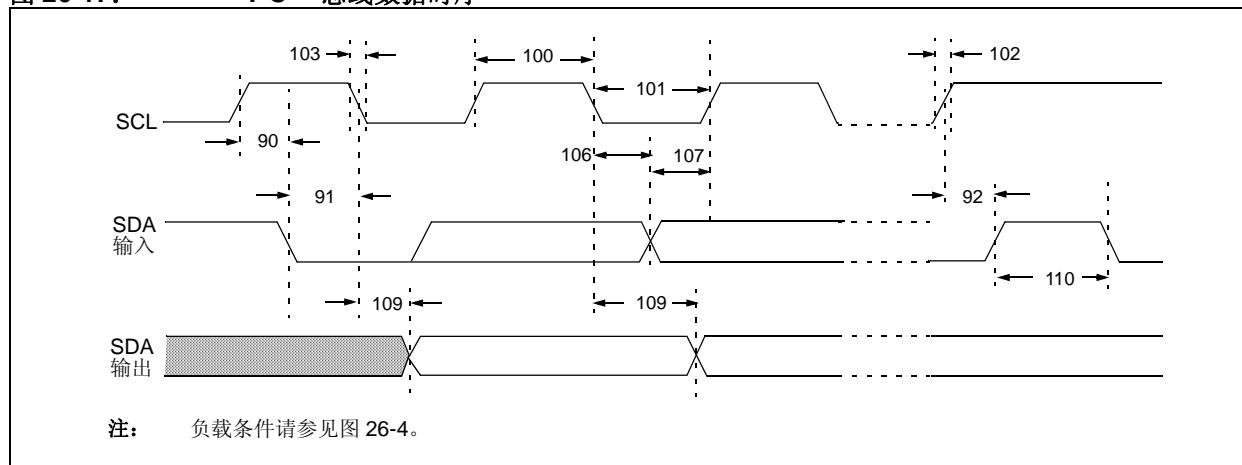


表 26-19: I²C™ 总线数据要求 (从模式)

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	4.0	—	μs	PIC18FXXXX 的工作频率不得低于 1.5 MHz
			400 kHz 模式	0.6	—	μs	PIC18FXXXX 的工作频率不得低于 10 MHz
			SSP 模块	1.5 TCY	—		
101	TLOW	时钟低电平时间	100 kHz 模式	4.7	—	μs	PIC18FXXXX 的工作频率不得低于 1.5 MHz
			400 kHz 模式	1.3	—	μs	PIC18FXXXX 的工作频率不得低于 10 MHz
			SSP 模块	1.5 TCY	—		
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值规定在 10 至 400 pF 之间
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20 + 0.1 CB	300	ns	CB 值规定在 10 至 400 pF 之间
90	TSU:STA	启动条件建立时间	100 kHz 模式	4.7	—	μs	仅与重复启动条件相关
			400 kHz 模式	0.6	—	μs	
91	THD:STA	启动条件保持时间	100 kHz 模式	4.0	—	μs	这个周期后产生第一个时钟脉冲
			400 kHz 模式	0.6	—	μs	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	μs	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	(注 2)
			400 kHz 模式	100	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	4.7	—	μs	
			400 kHz 模式	0.6	—	μs	
109	TAA	从时钟有效到输出有效的时间	100 kHz 模式	—	3500	ns	(注 1)
			400 kHz 模式	—	—	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	μs	在启动一个新的传输前总线必须保持空闲的时间
			400 kHz 模式	1.3	—	μs	
D102	CB	总线容性负载		—	400	pF	

注 1: 为避免产生意外的启动或停止条件, 作为发送器的器件必须提供这个内部最小延时以补偿 SCL 下降沿的未定义区域 (最小值 300 ns)。

2: 快速模式的 I²C 总线器件也可在标准模式的 I²C 总线系统中使用, 但必须满足 $TSU:DAT \geq 250$ ns 的要求。如果快速模式器件没有延长 SCL 信号的低电平周期, 则必然满足此条件。如果该器件延长了 SCL 信号的低电平周期, 它必须将下一个数据位输出到 SDA 线。SCL 线被释放前, 根据标准模式 I²C 总线规范, $TR_{max.} + TSU:DAT = 1000 + 250 = 1250$ ns。

PIC18F2XK20/4XK20

图 26-18: 主 SSP I²CTM 总线启动位 / 停止位时序波形

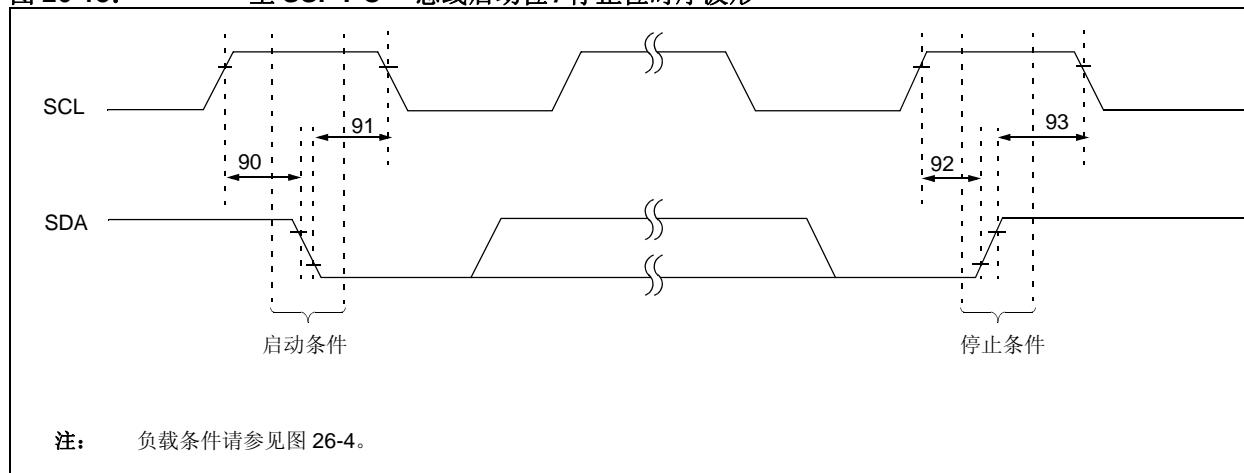


表 26-20: 主 SSP I²CTM 总线启动位 / 停止位要求

参数编号	符号	特性	最小值	最大值	单位	条件
90	TSU:STA	启动条件 建立时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	仅与重复启动条件相关
			400 kHz 模式	2(Tosc)(BRG + 1)	—	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	
91	THD:STA	启动条件 保持时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(Tosc)(BRG + 1)	—	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	
92	TSU:STO	停止条件 建立时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ns
			400 kHz 模式	2(Tosc)(BRG + 1)	—	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	
93	THD:STO	停止条件 保持时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ns
			400 kHz 模式	2(Tosc)(BRG + 1)	—	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	

注 1：对于所有 I²C 引脚，最小引脚电容均为 10 pF。

图 26-19: 主 SSP I²CTM 总线数据时序

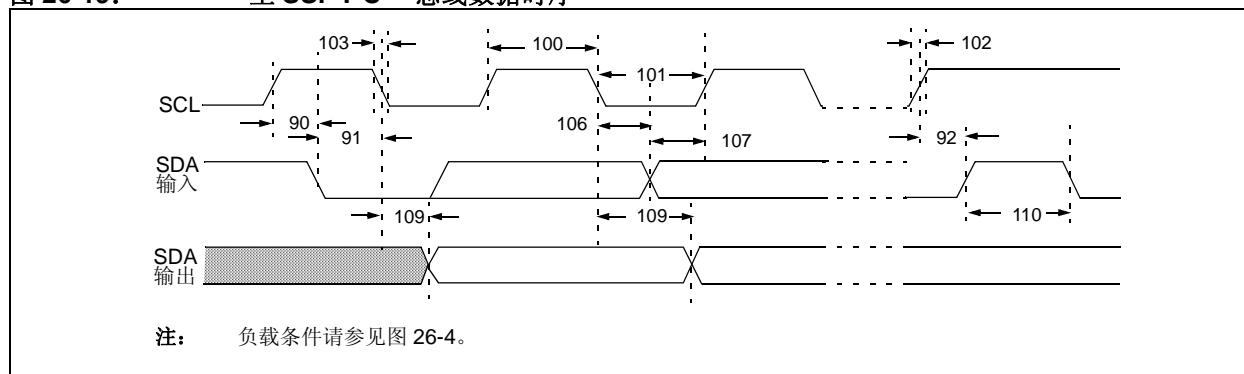


表 26-21：主 SSP I²CTM 总线数据要求

参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ms	
			400 kHz 模式	2(Tosc)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	ms	
101	TLOW	时钟低电平时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ms	
			400 kHz 模式	2(Tosc)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	ms	
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	CB 值规定在 10 至 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	300	ns	
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	CB 值规定在 10 至 400 pF 之间
			400 kHz 模式	20 + 0.1 CB	300	ns	
			1 MHz 模式 (1)	—	100	ns	
90	TSU:STA	启动条件建立时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ms	仅与重复启动条件相关
			400 kHz 模式	2(Tosc)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	ms	
91	THD:STA	启动条件保持时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ms	这个周期后产生第一个时钟脉冲
			400 kHz 模式	2(Tosc)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	ms	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	ms	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	(注 2)
			400 kHz 模式	100	—	ns	
92	TSU:STO	停止条件建立时间	100 kHz 模式	2(Tosc)(BRG + 1)	—	ms	
			400 kHz 模式	2(Tosc)(BRG + 1)	—	ms	
			1 MHz 模式 (1)	2(Tosc)(BRG + 1)	—	ms	
109	TAA	从时钟有效到输出有效的时间	100 kHz 模式	—	3500	ns	
			400 kHz 模式	—	1000	ns	
			1 MHz 模式 (1)	—	—	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	ms	在启动一个新的传输前总线必须保持空闲的时间
			400 kHz 模式	1.3	—	ms	
D102	CB	总线容性负载		—	400	pF	

注 1: 对于所有 I²C 引脚，最小引脚电容均为 10 pF。

2: 快速模式的 I²C 总线器件也可在标准模式的 I²C 总线系统中使用，但必须满足参数 $107 \geq 250$ ns 的要求。如果快速模式器件没有延长 SCL 信号的低电平周期，则必然满足此条件。如果该器件延长了 SCL 信号的低电平周期，它必须将下一个数据位输出到 SDA 线。SCL 线被释放前，在 100 kHz 模式下，参数 $102 +$ 参数 $107 = 1000 + 250 = 1250$ ns。

PIC18F2XK20/4XK20

图 26-20: EUSART 同步发送（主 / 从）时序

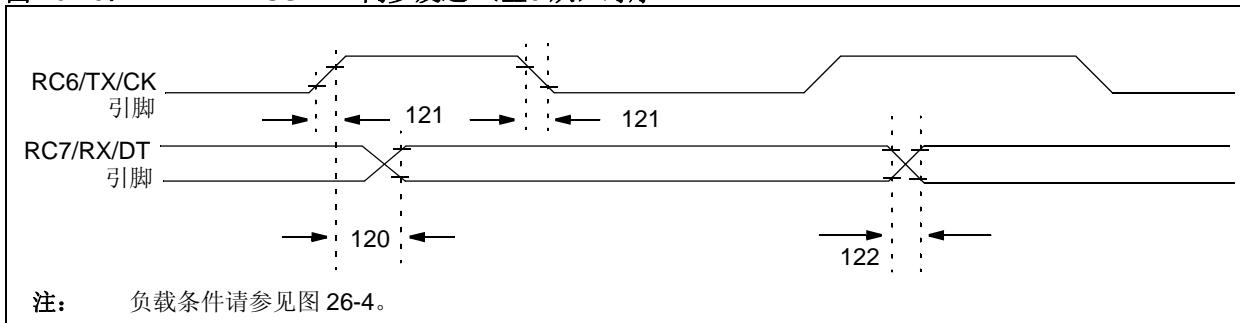


表 26-22: EUSART 同步发送要求

参数编号	符号	特性	最小值	最大值	单位	条件
120	TckH2dtV	同步发送（主 / 从） 时钟高电平到数据输出有效的时间	—	40	ns	
121	Tckrf	时钟输出上升时间和下降时间（主模式）	—	20	ns	
122	Tdtrf	数据输出上升时间和下降时间	—	20	ns	

图 26-21: EUSART 同步接收（主 / 从）时序

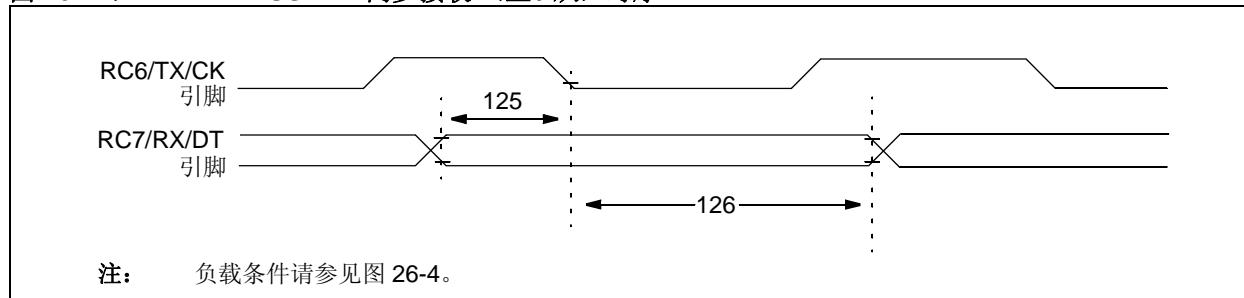


表 26-23: EUSART 同步接收要求

参数编号	符号	特性	最小值	最大值	单位	条件
125	TdtV2ckl	同步接收（主 / 从） CK ↓之前数据的建立时间（DT 建立时间）	10	—	ns	
126	TckL2dtl	CK ↓之后数据的保持时间（DT 保持时间）	15	—	ns	

PIC18F2XK20/4XK20

表 26-24: A/D 转换器特性: PIC18F2XK20/4XK20

参数编号	符号	特性	最小值	典型值	最大值	单位	条件
A01	NR	分辨率	—	—	10	位	-40°C 至 +85°C, $\Delta V_{REF} \geq 2.0V$
A03	EIL	积分线性误差	—	± 0.5	± 1	LSb	-40°C 至 +85°C, $\Delta V_{REF} \geq 2.0V$
A04	EDL	微分线性误差	—	± 0.4	± 1	LSb	-40°C 至 +85°C, $\Delta V_{REF} \geq 2.0V$
A06	E _{OFF}	失调误差	—	0.4	± 2	LSb	-40°C 至 +85°C, $\Delta V_{REF} \geq 2.0V$
A07	E _{GN}	增益误差	—	0.3	± 2	LSb	-40°C 至 +85°C, $\Delta V_{REF} \geq 2.0V$
A08	E _{TOTAL}	总误差	—	1	± 3	LSb	-40°C 至 +85°C, $\Delta V_{REF} \geq 2.0V$
A20	ΔV_{REF}	参考电压范围 ($V_{REFH} - V_{REFL}$)	1.8 2.0	— —	— —	V V	绝对最小值 确保1 LSb精度的最小值
A21	V _{REFH}	参考电压高电压	V _{DD} /2	—	V _{DD} + 0.3	V	
A22	V _{REFL}	参考电压低电压	V _{SS} - 0.3V	—	V _{DD} /2	V	
A25	V _{AIN}	模拟输入电压	V _{REFL}	—	V _{REFH}	V	
A30	Z _{AIN}	模拟信号源的推荐阻抗	—	—	3	kΩ	-40°C 至 +85°C

注 1: A/D 转换结果不会因输入电压的增加而减小，并且不会丢失编码。

2: V_{REFH} 电流来自选择作为 V_{REFH} 源的 RA3/AN3/V_{REF+} 引脚或 V_{DD}。

V_{REFL} 电流来自选择作为 V_{REFL} 源的 RA2/AN2/V_{REF-/CVREF} 引脚或 V_{SS}。

PIC18F2XK20/4XK20

图 26-22: A/D 转换时序

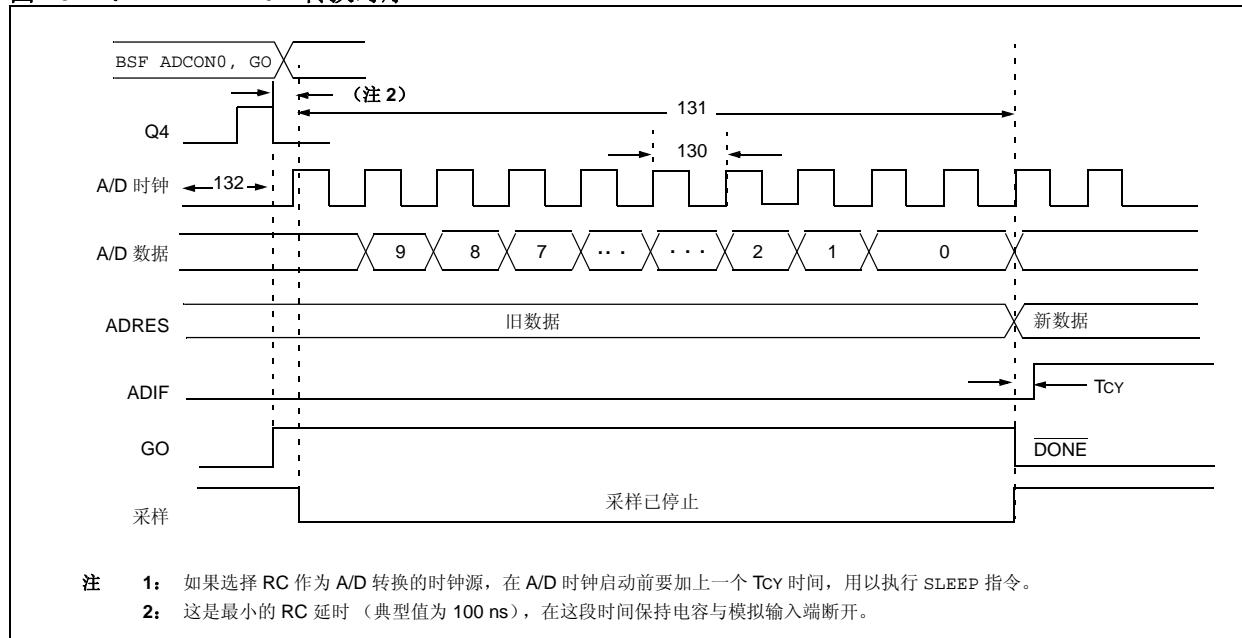


表 26-25: A/D 转换要求

参数编号	符号	特性	最小值	最大值	单位	条件
130	TAD	A/D 时钟周期	0.7	25.0 ⁽¹⁾	μs	基于 Tosc, -40°C 至 +85°C
			0.7	4.0 ⁽¹⁾	μs	基于 Tosc, +85°C 至 +125°C
			1.0	4.0	μs	FRC 模式, VDD ≥ 2.0V
131	TCNV	转换时间（不包括采集时间） ^(注 2)	12	12	TAD	
132	TACQ	采集时间 ^(注 3)	1.4	—	ms	VDD = 3V, RS = 50Ω
135	TSWC	转换 → 采样的切换时间	—	(注 4)		
136	TDIS	电容放电时间	2	2	TAD	

图注：TBD = 待定

- 注
- A/D 时钟周期取决于器件频率和 TAD 时钟分频比。
 - ADRES 寄存器可在下一个 TCY 周期被读取。
 - 转换完成后当电压满量程变化时（VDD 至 VSS 或 VSS 至 VDD），保持电容采集一个“新”输入电压所需的时间。在输入通道上的信号源阻抗（RS）为 50Ω。
 - 在器件时钟的下一个周期。

27.0 直流和交流特性图表

图 27-1: PIC18F4XK20/PIC18F2XK20 的基本 IPD 典型值曲线

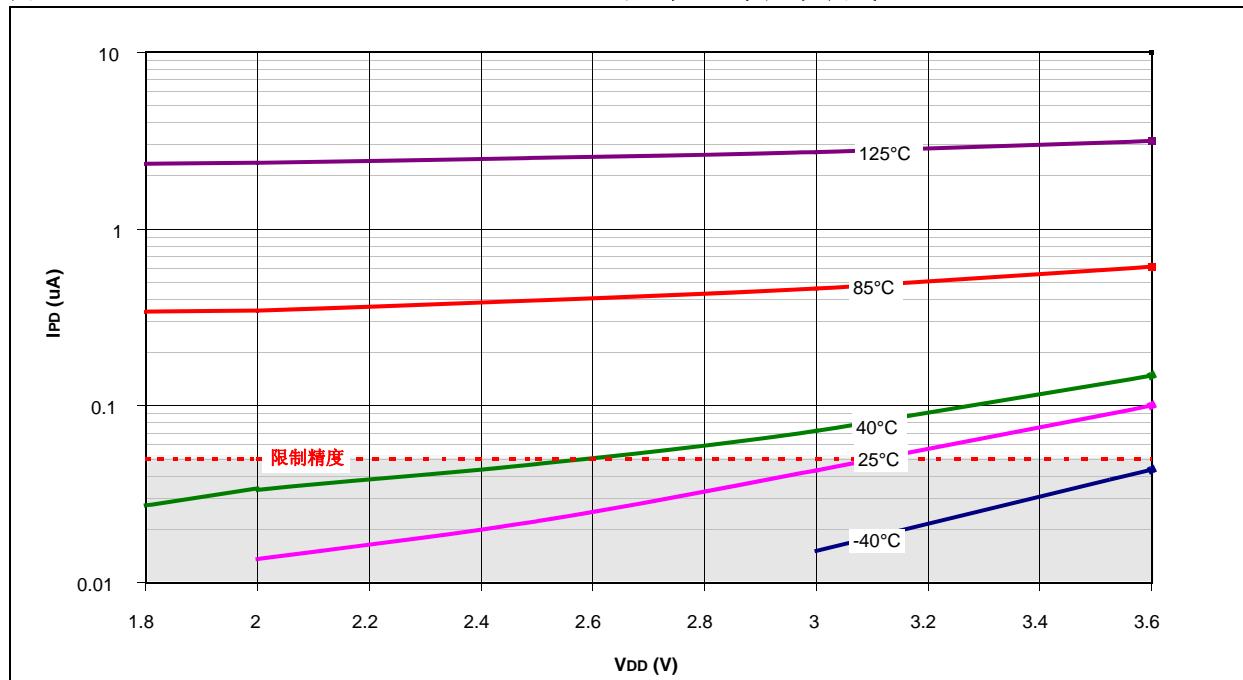
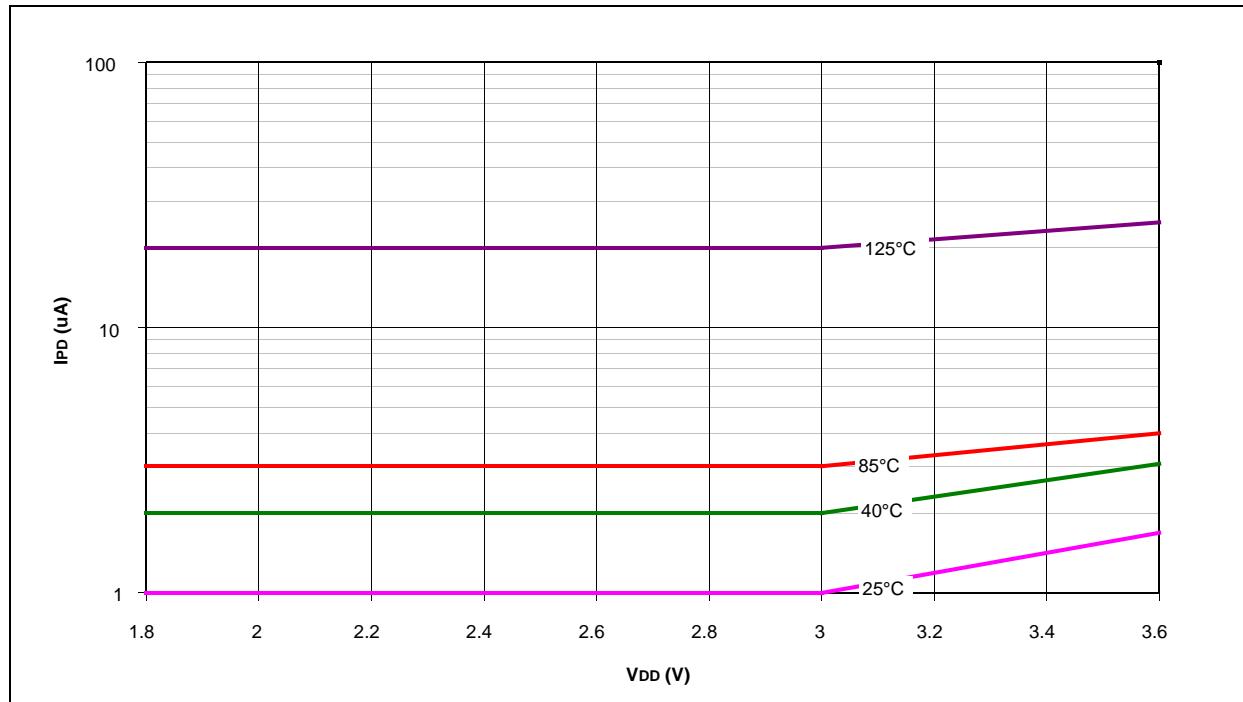


图 27-2: PIC184XK20/PIC18F2XK20 的基本 IPD 最大值曲线



PIC18F2XK20/4XK20

图 27-3: PIC18F4XK20/PIC18F2XK20 RC_RUN 模式、31 KHZ 时的 IDD 典型值曲线

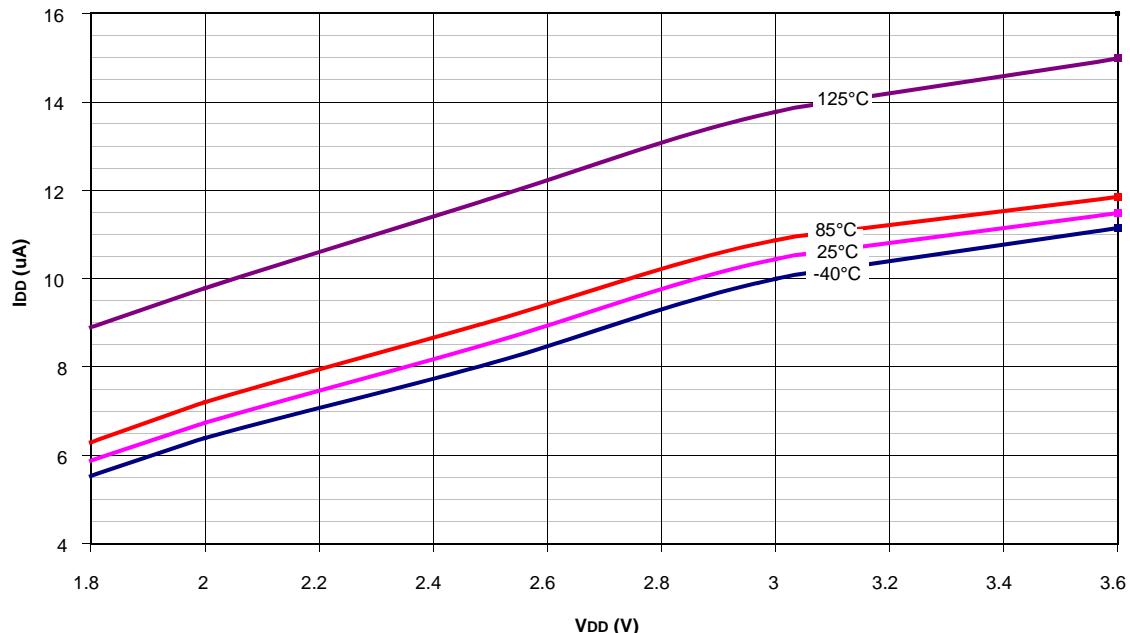
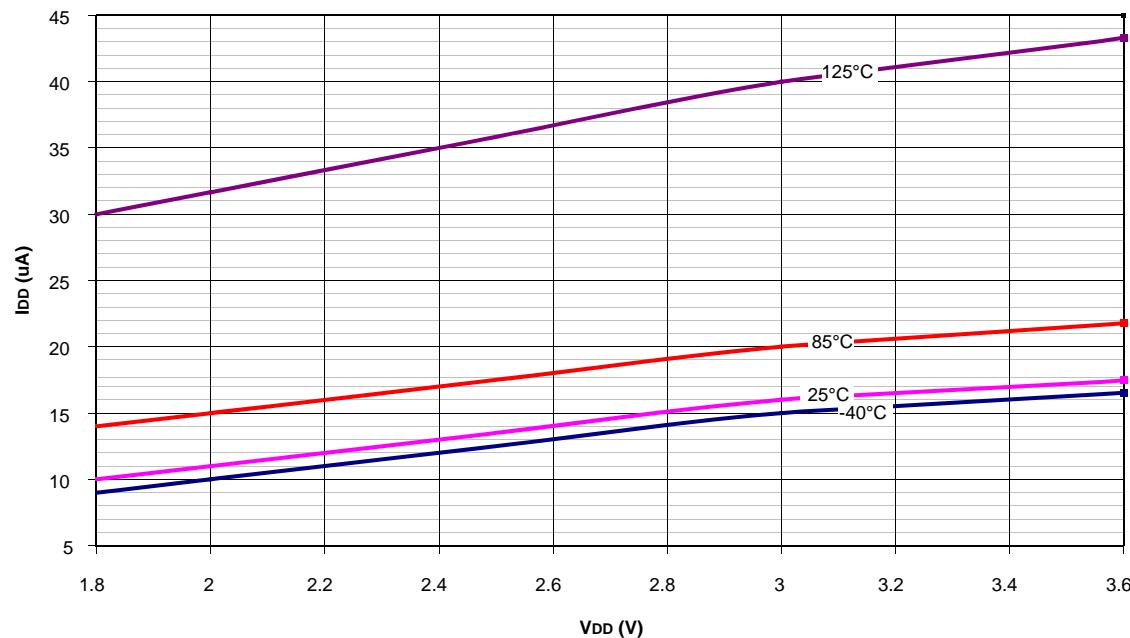


图 27-4: PIC18F4XK20/PIC18F2XK20 RC_RUN 模式、31 KHZ 时的 IDD 最大值曲线



PIC18F2XK20/4XK20

图 27-5: PIC18F4XK20/PIC18F2XK20 RC_RUN 模式时的 IDD 典型值曲线

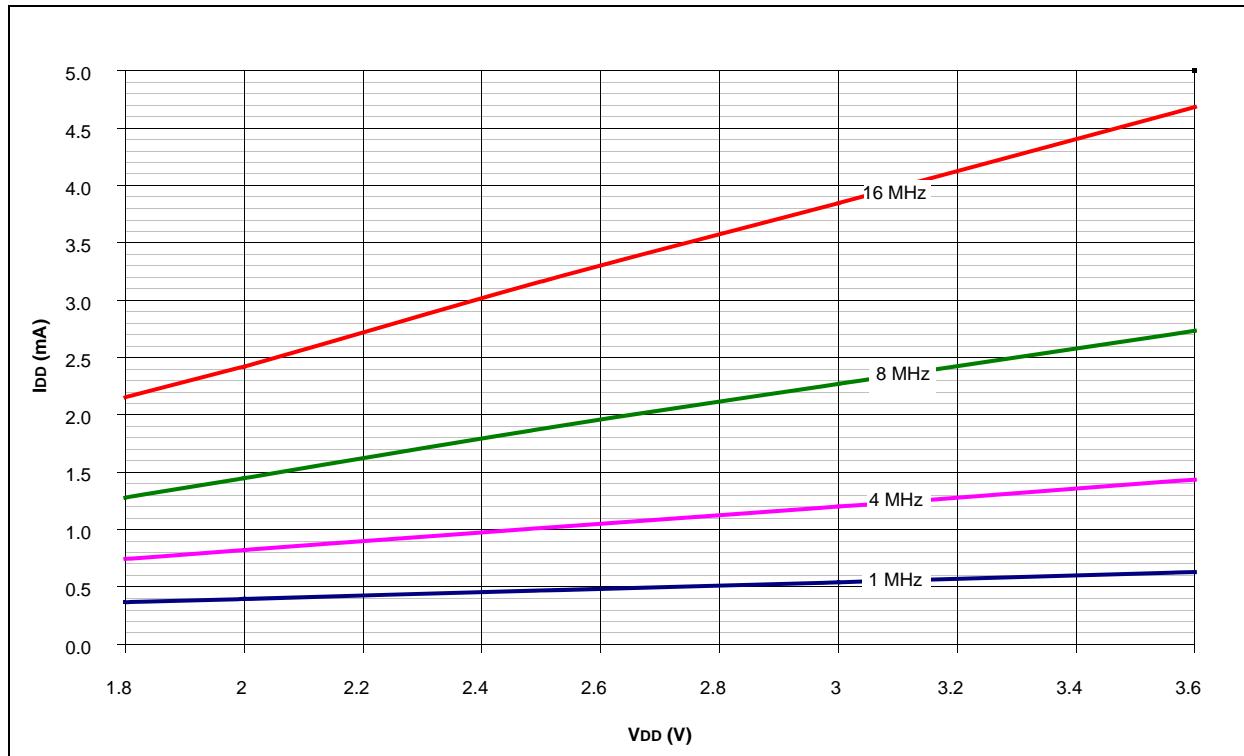
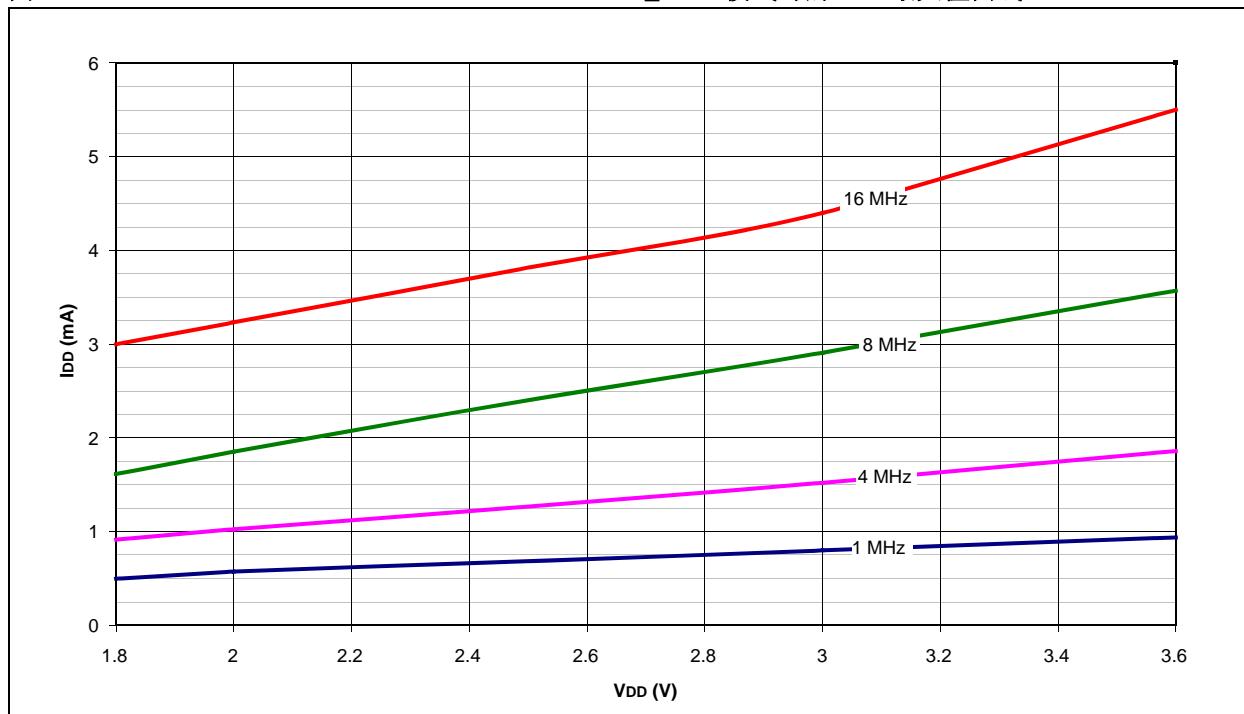


图 27-6: PIC18F4XK20/PIC18F2XK20 RC_RUN 模式时的 IDD 最大值曲线



PIC18F2XK20/4XK20

图 27-7: PIC18F4XK20/PIC18F2XK20 RC_IDLE 模式、31 KHZ 时的 IDD 典型值曲线

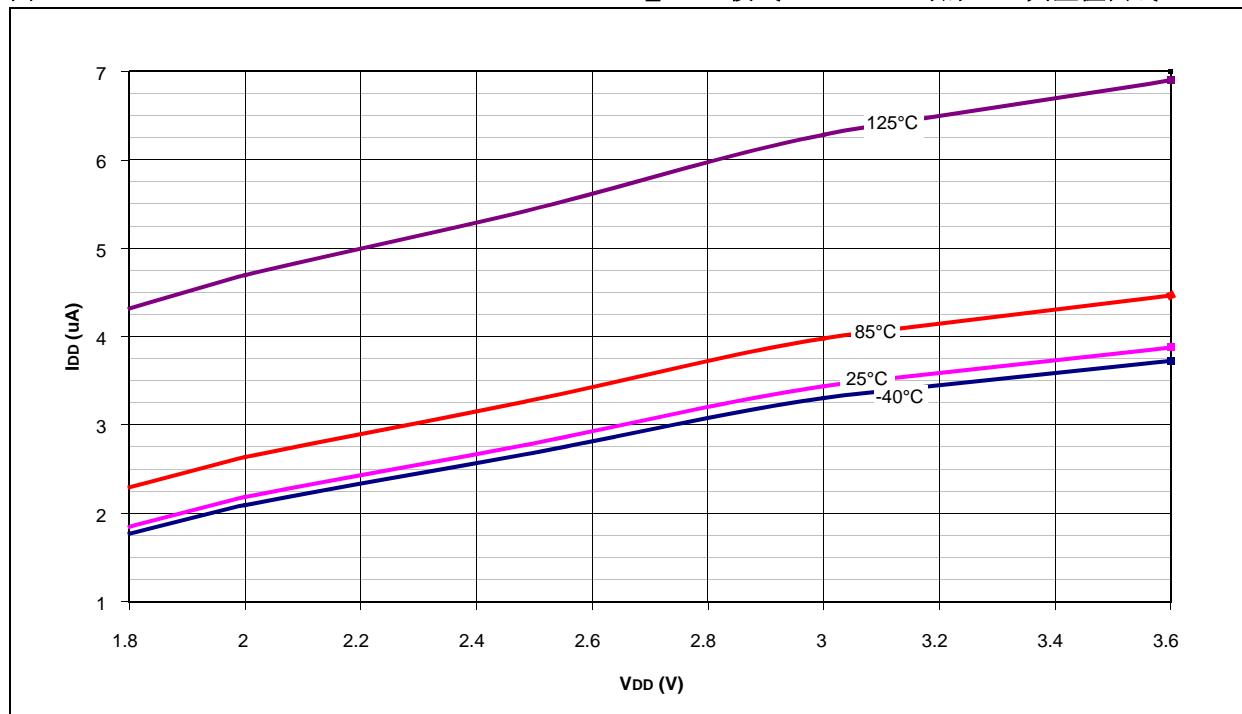


图 27-8: PIC18F4XK20/PIC18F2XK20 RC_IDLE 模式、31 KHZ 时的 IDD 最大值曲线

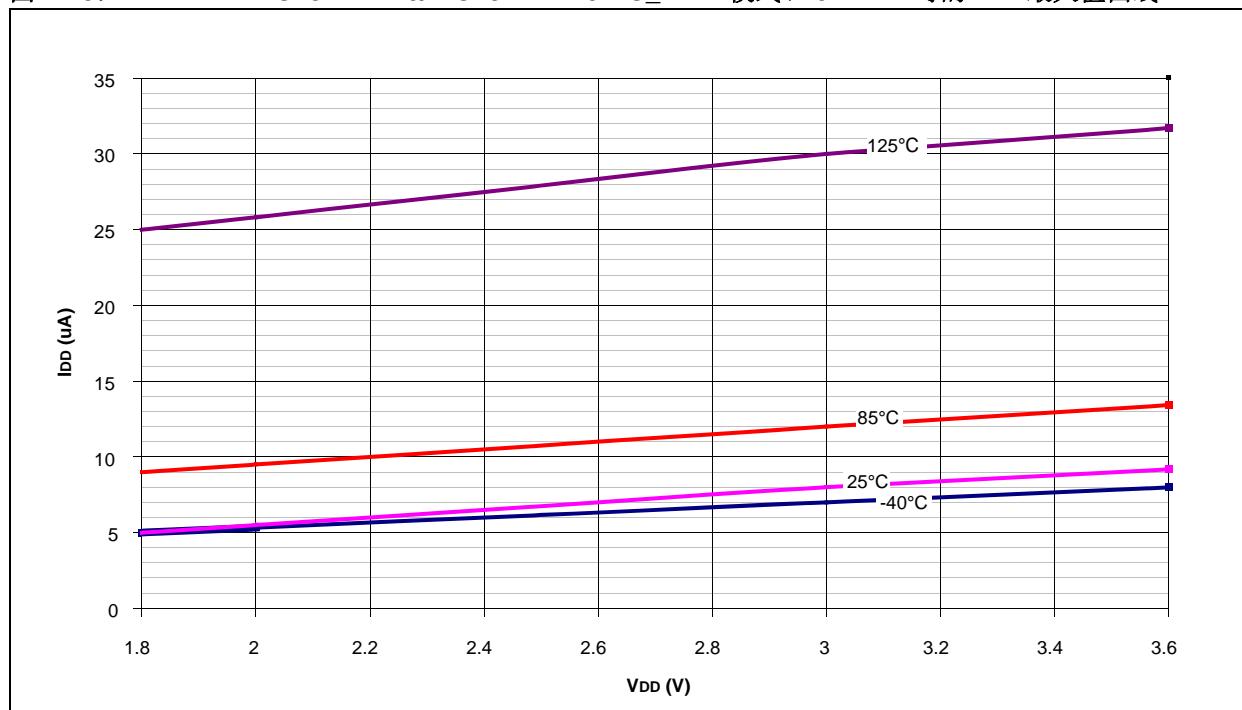


图 27-9: PIC18F4XK20/PIC18F2XK20 RC_IDLE 模式时的 IDD 典型值曲线

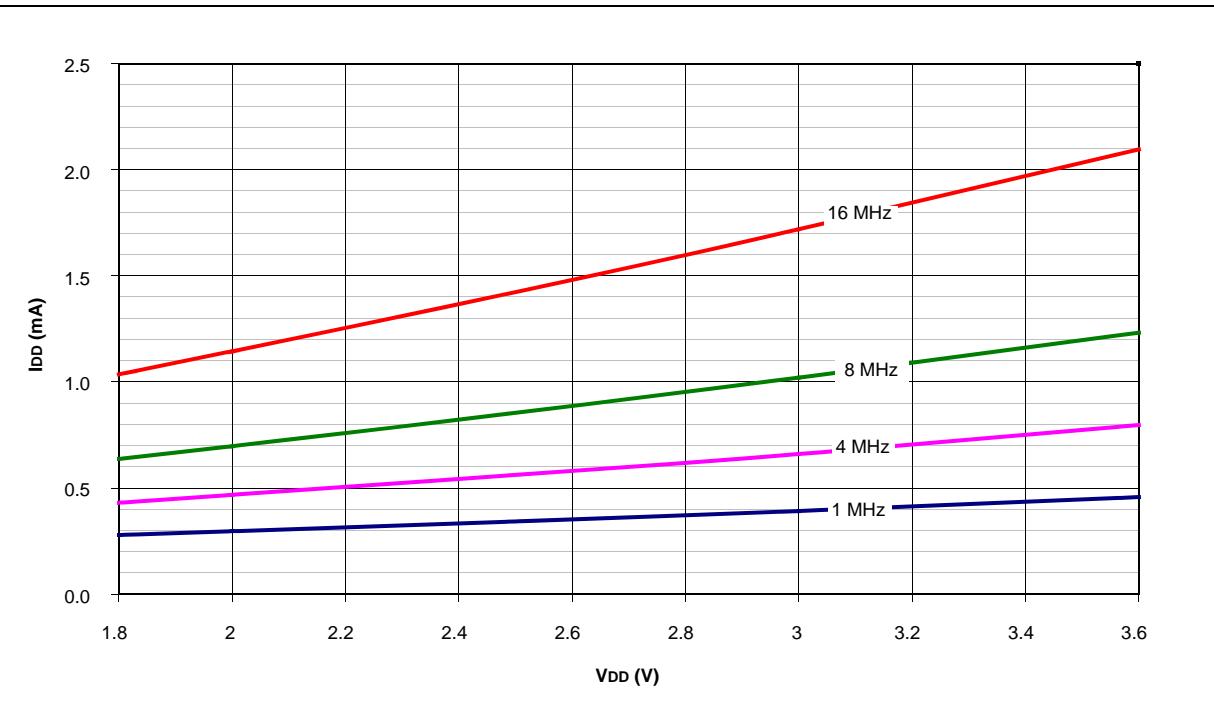
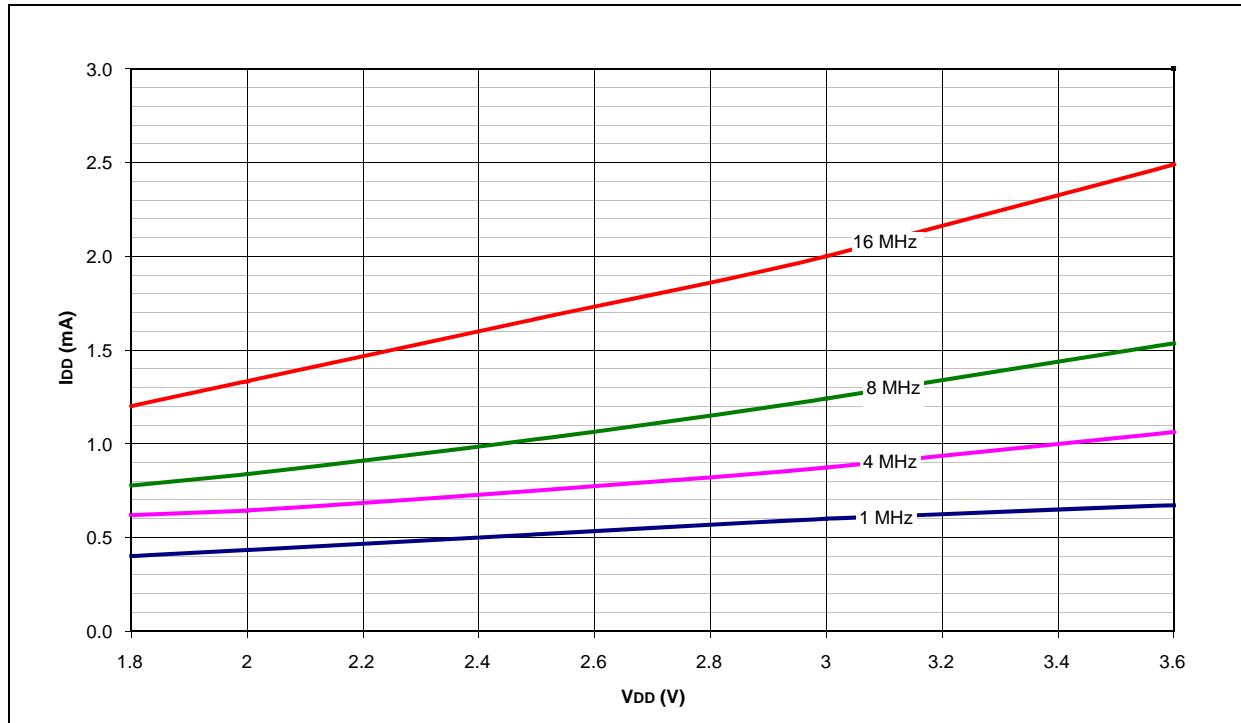


图 27-10: PIC18F4XK20/PIC18F2XK20 RC_IDLE 模式时的 IDD 最大值曲线



PIC18F2XK20/4XK20

图 27-11: PIC18F4XK20/PIC18F2XK20 PRI_RUN 模式时的 IDD 典型值曲线 (EC)

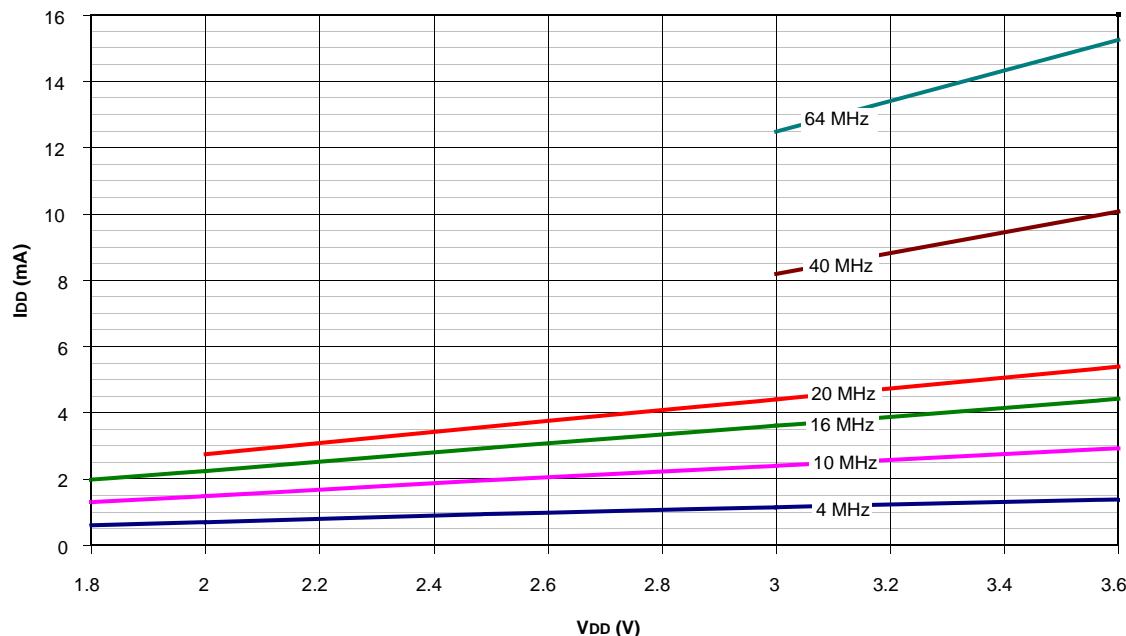
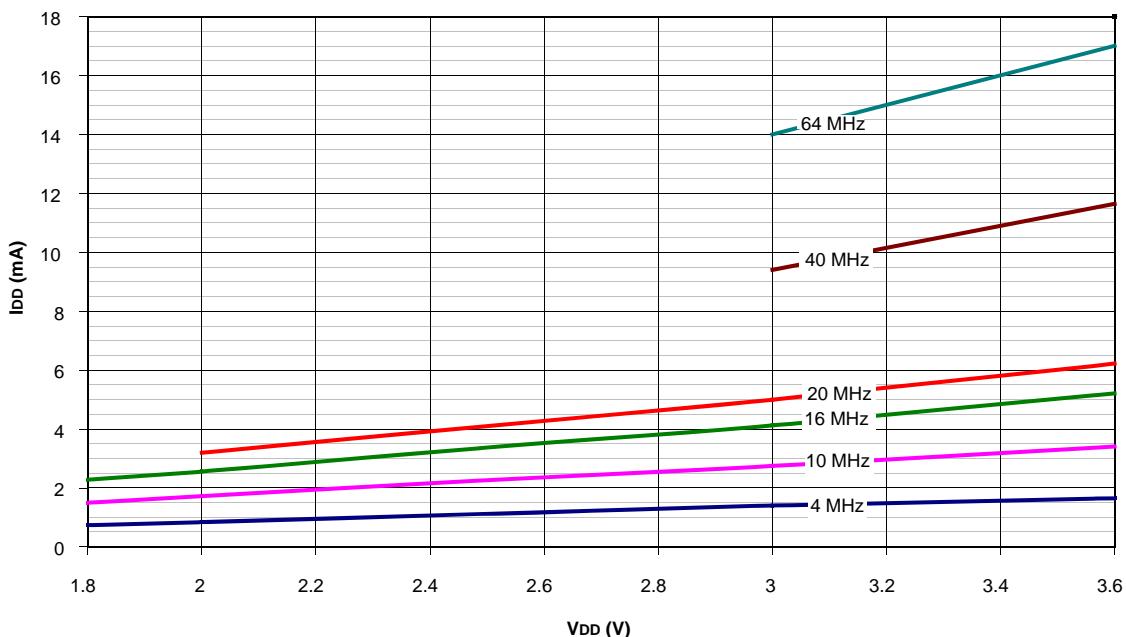


图 27-12: PIC18F4XK20/PIC18F2XK20 PRI_RUN 模式时的 IDD 最大值曲线 (EC)



PIC18F2XK20/4XK20

图 27-13: PIC18F4XK20/PIC18F2XK20 PRI_RUN 模式时的 IDD 典型值曲线 (HS + PLL)

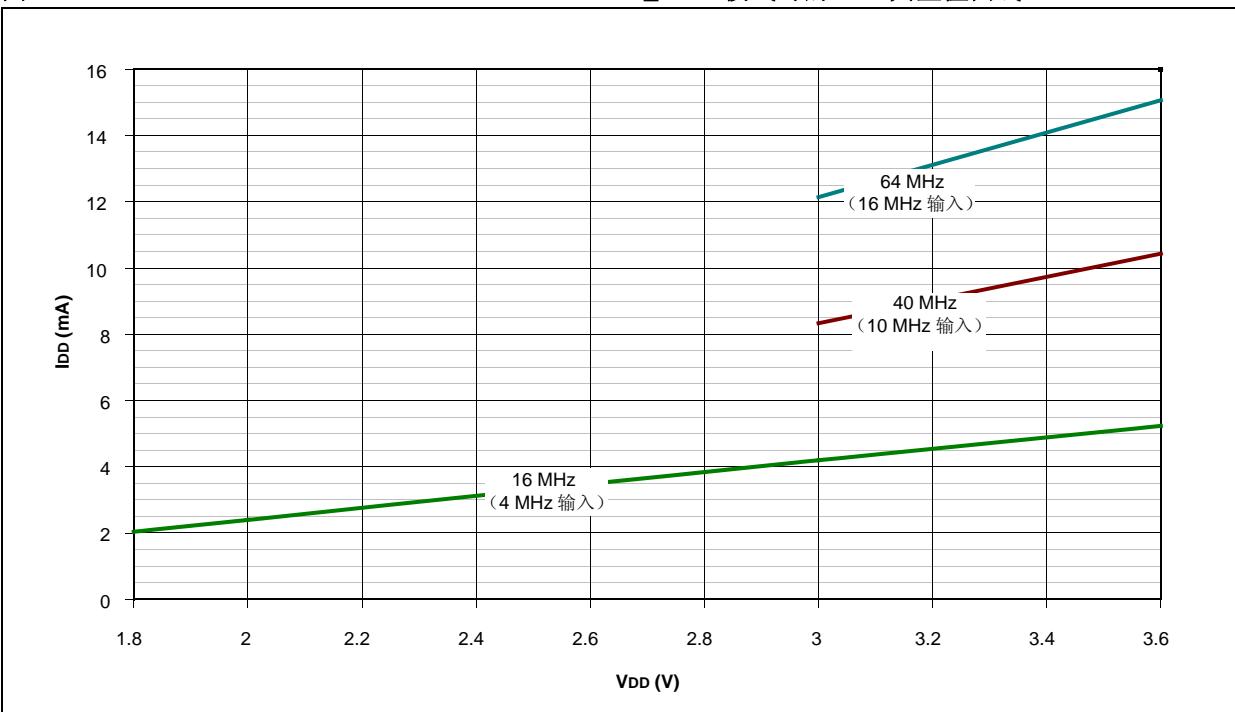
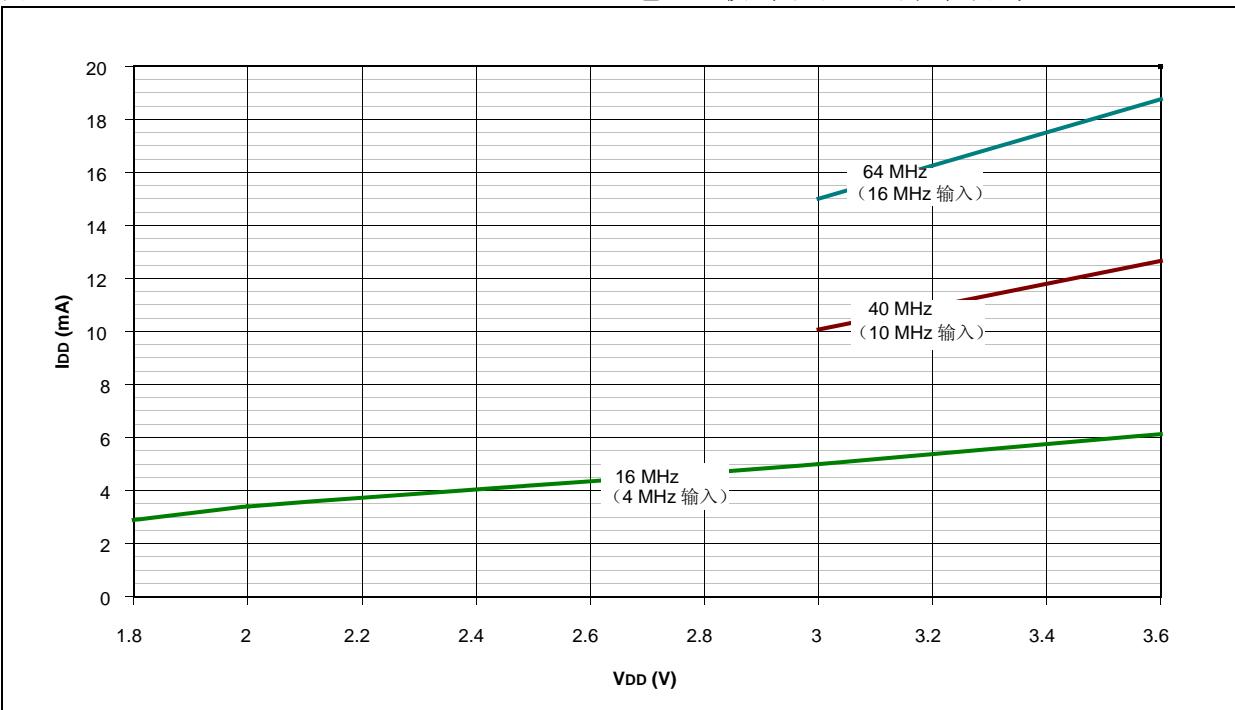


图 27-14: PIC18F4XK20/PIC18F2XK20 PRI_RUN 模式时的 IDD 最大值曲线 (HS + PLL)



PIC18F2XK20/4XK20

图 27-15: PIC18F4XK20/PIC18F2XK20 PRI_IDLE 模式时的 IDD 典型值曲线 (EC)

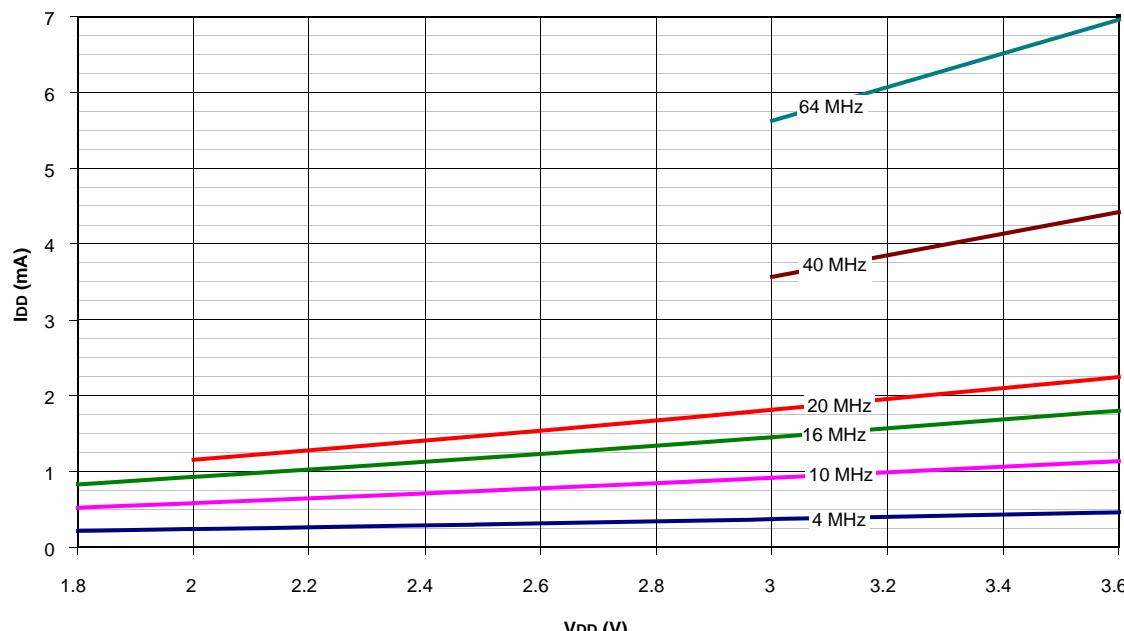
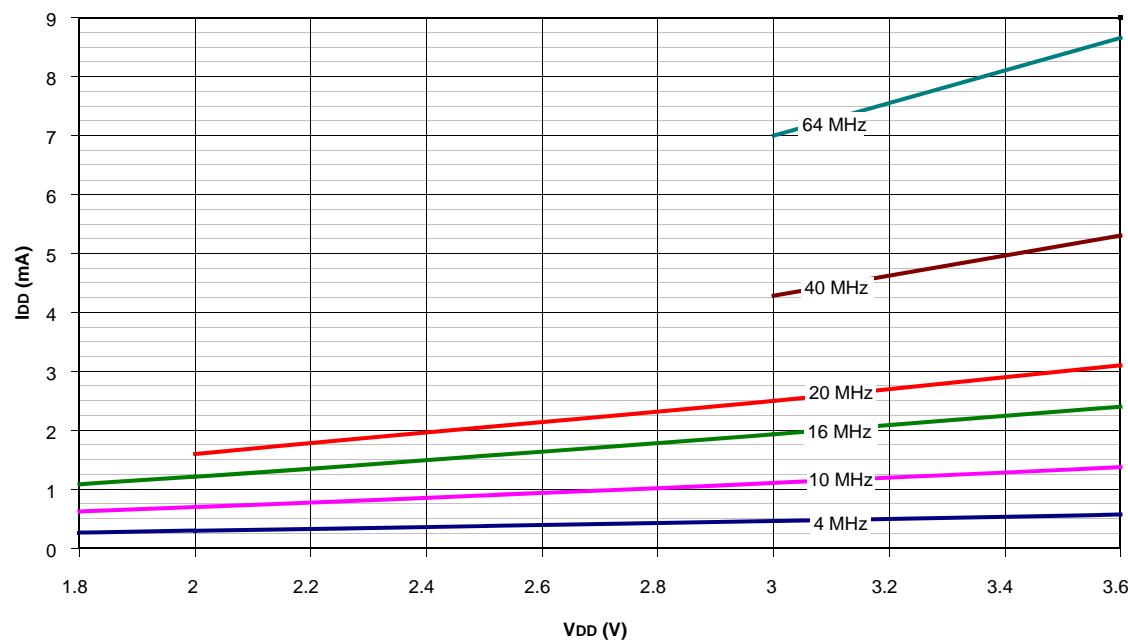


图 27-16: PIC18F4XK20/PIC18F2XK20 PRI_IDLE 模式时的 IDD 最大值曲线 (EC)



PIC18F2XK20/4XK20

图 27-17:

PIC18F4XK20/PIC18F2XK20 的 ΔI_{WDT} —看门狗定时器的 ΔI_{PD} 曲线, -40°C 至 +125°C

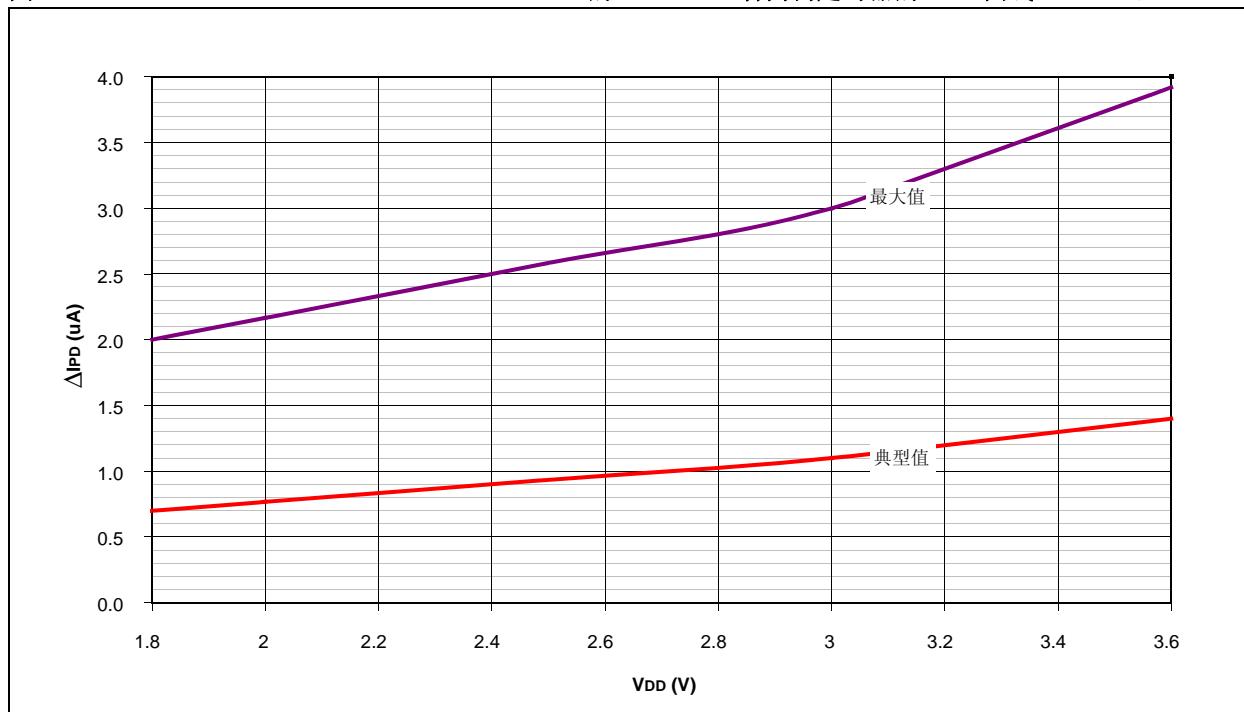
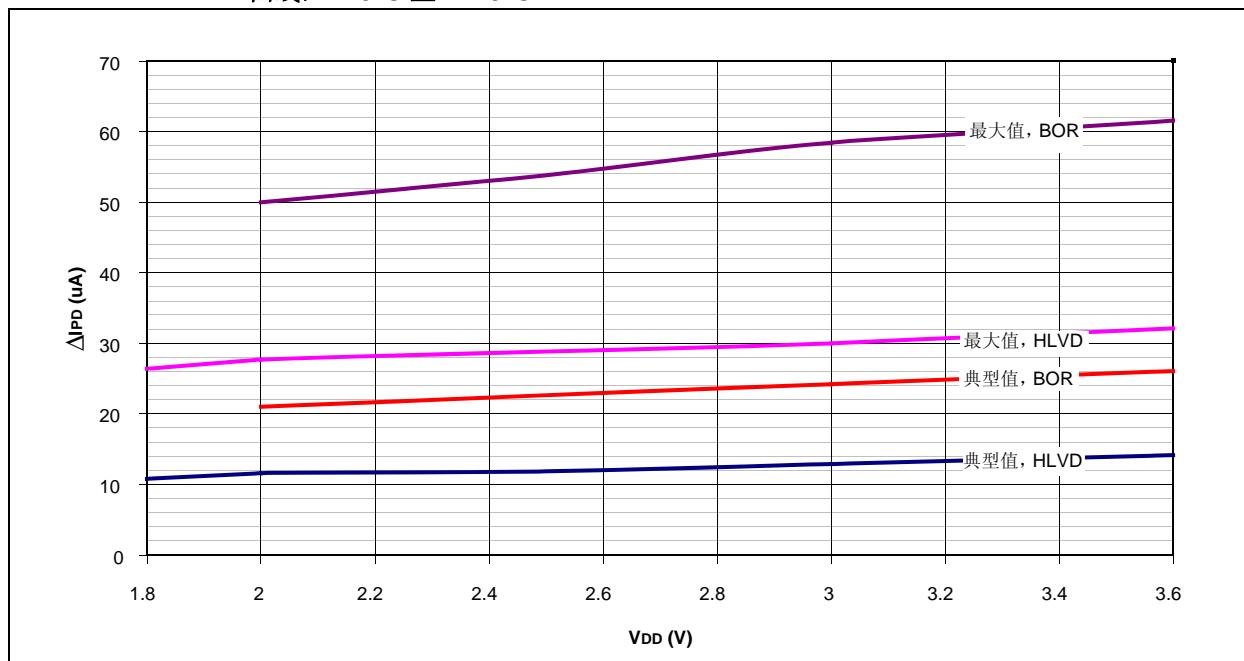


图 27-18:

PIC18F4XK20/PIC18F2XK20 的 ΔI_{BOR} 和 ΔI_{HLVD} —欠压复位和高 / 低压检测的 ΔI_{PD} 曲线, -40°C 至 +125°C



PIC18F2XK20/4XK20

图 27-19: PIC18F4XK20/PIC18F2XK20 的 ΔI_{OCSB} —低功耗 Timer1 振荡器的 ΔI_{PD} 曲线

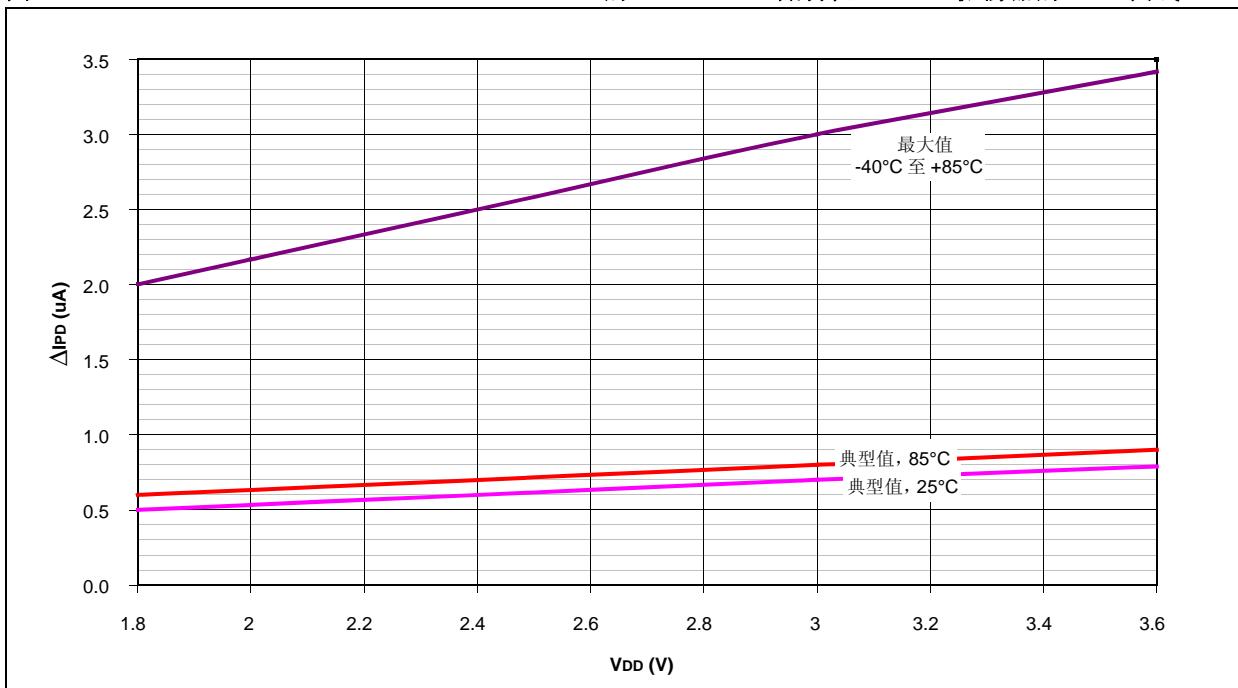
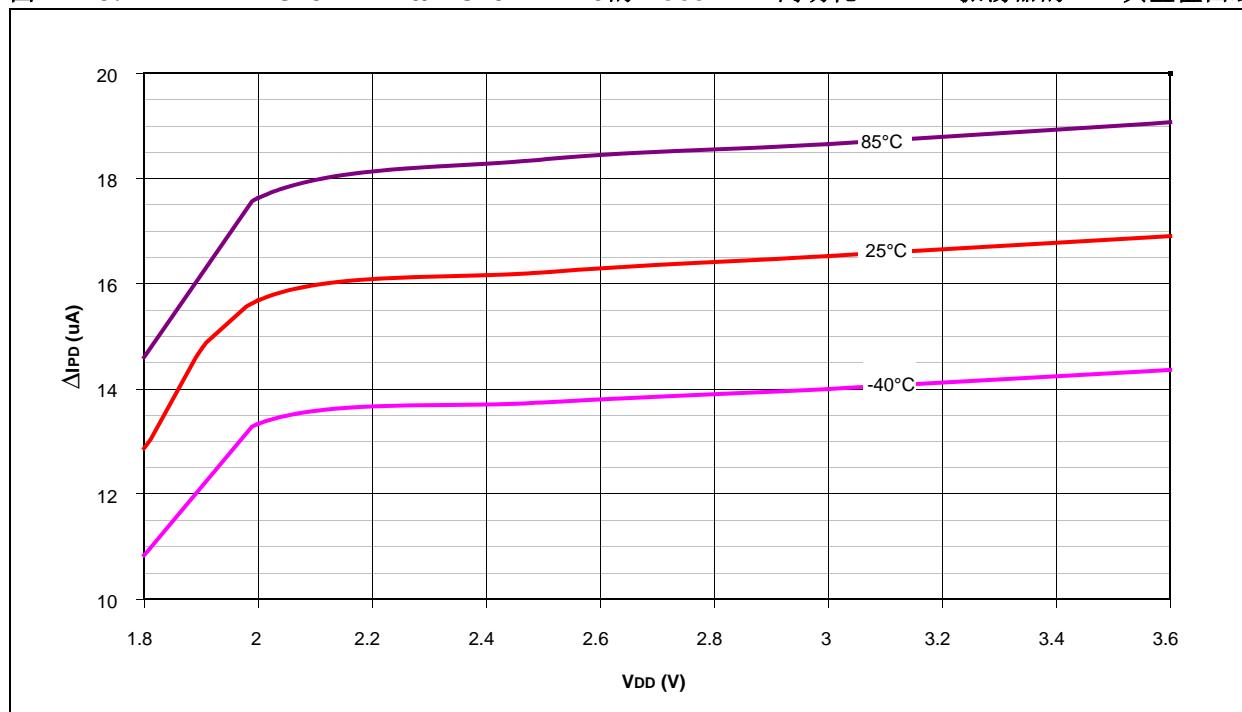


图 27-20: PIC18F4XK20/PIC18F2XK20 的 ΔI_{OCSB} —高功耗 TIMER1 振荡器的 ΔI_{PD} 典型值曲线



PIC18F2XK20/4XK20

图 27-21：

PIC18F4XK20/PIC18F2XK20 的 ΔI_{PD} ——高功耗 TIMER1 振荡器的 ΔI_{PD} 最大值曲线

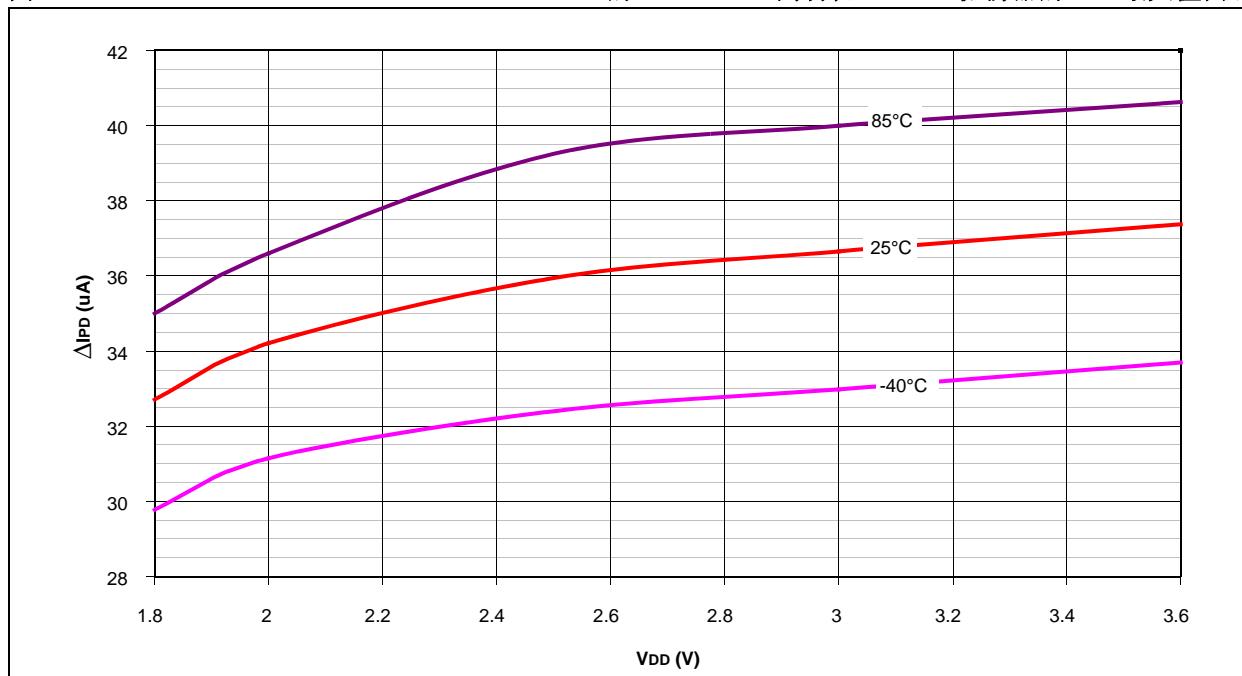
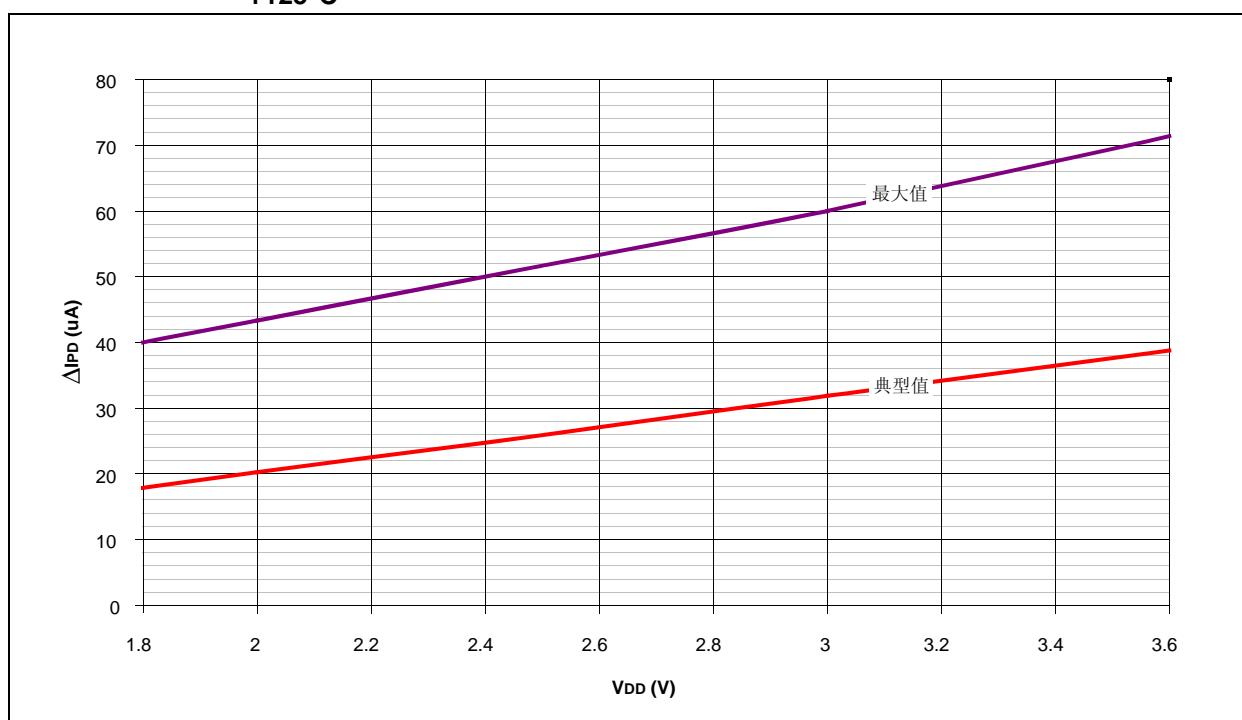


图 27-22：

PIC18F4XK20/PIC18F2XK20 的 ΔI_{CVREF} ——比较器参考电压的 ΔI_{PD} 曲线，-40°C 至 +125°C



PIC18F2XK20/4XK20

图 27-23: PIC18F4XK20/PIC18F2XK20 的 ΔI_{AD} —ADC 的 ΔI_{DD} 典型值曲线, 25°C 至 +125°C
(运行模式, ADC 启动, 但不进行转换)

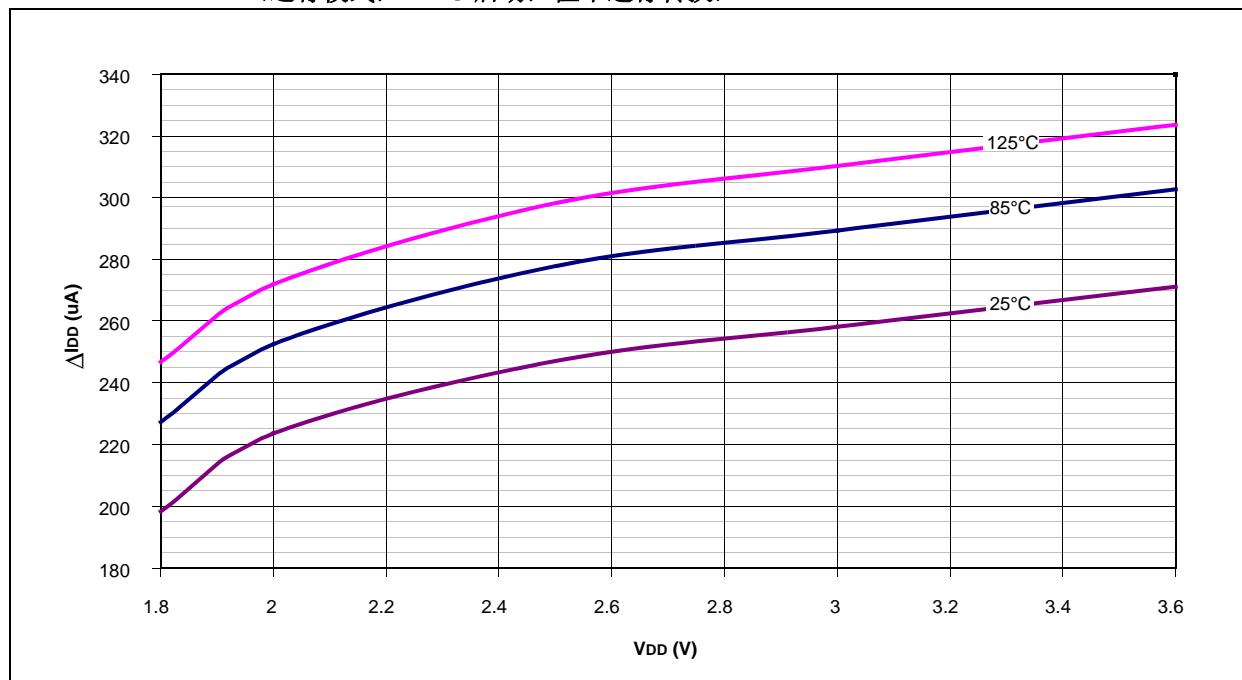


图 27-24: PIC18F4XK20/PIC18F2XK20 的 ΔI_{AD} —ADC 的 ΔI_{DD} 最大值曲线, 25°C 至 +125°C
(运行模式, ADC 启动, 但不进行转换)

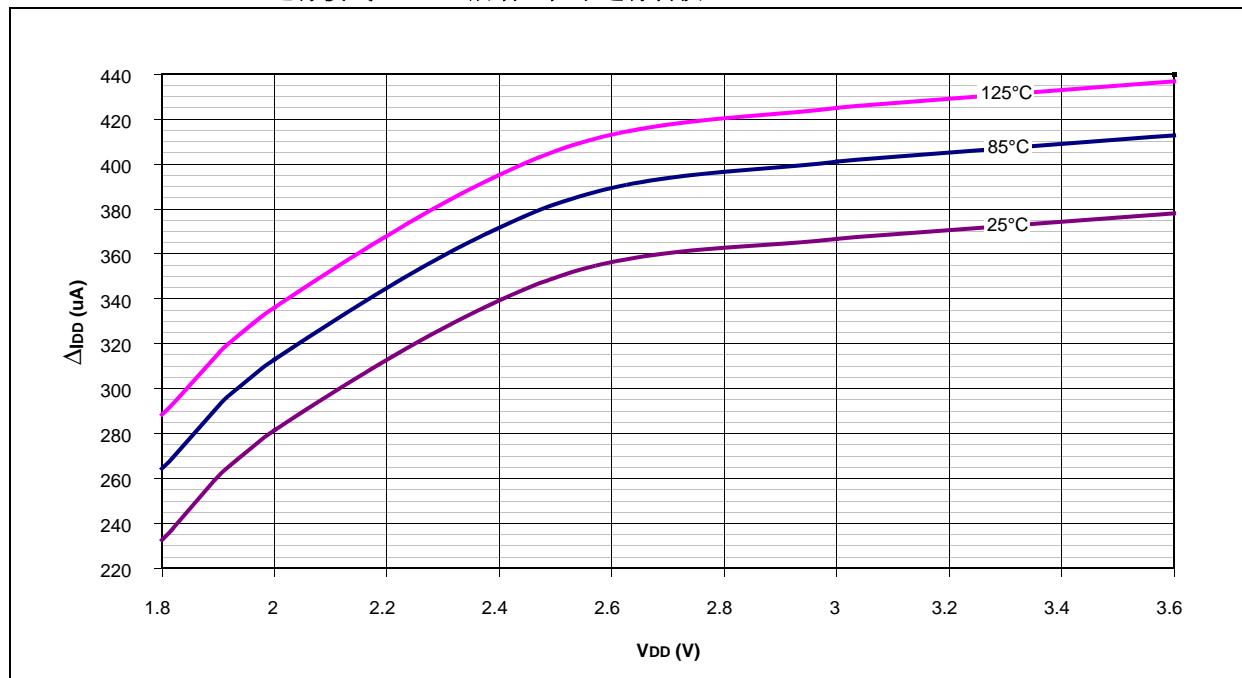


图 27-25: PIC18F4XK20/PIC18F2XK20 的 ΔI_{COMP} ——低功耗模式下比较器的 ΔI_{PD} 典型值曲线,
 -40°C 至 $+125^{\circ}\text{C}$

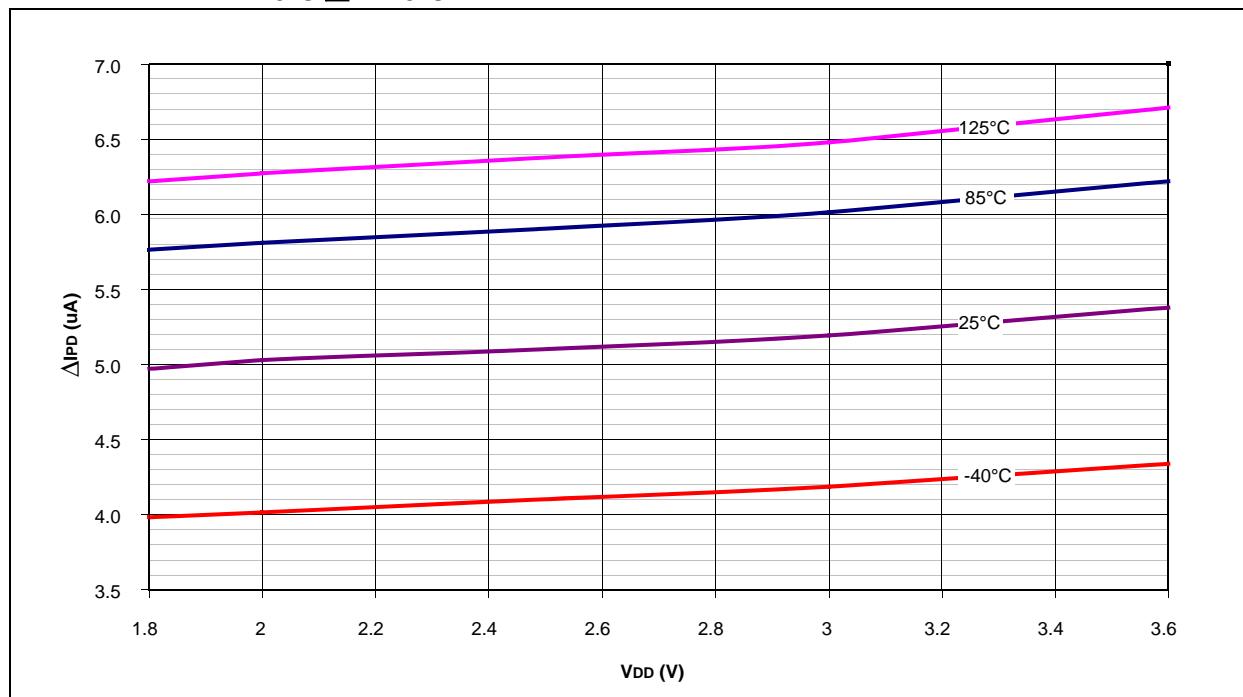
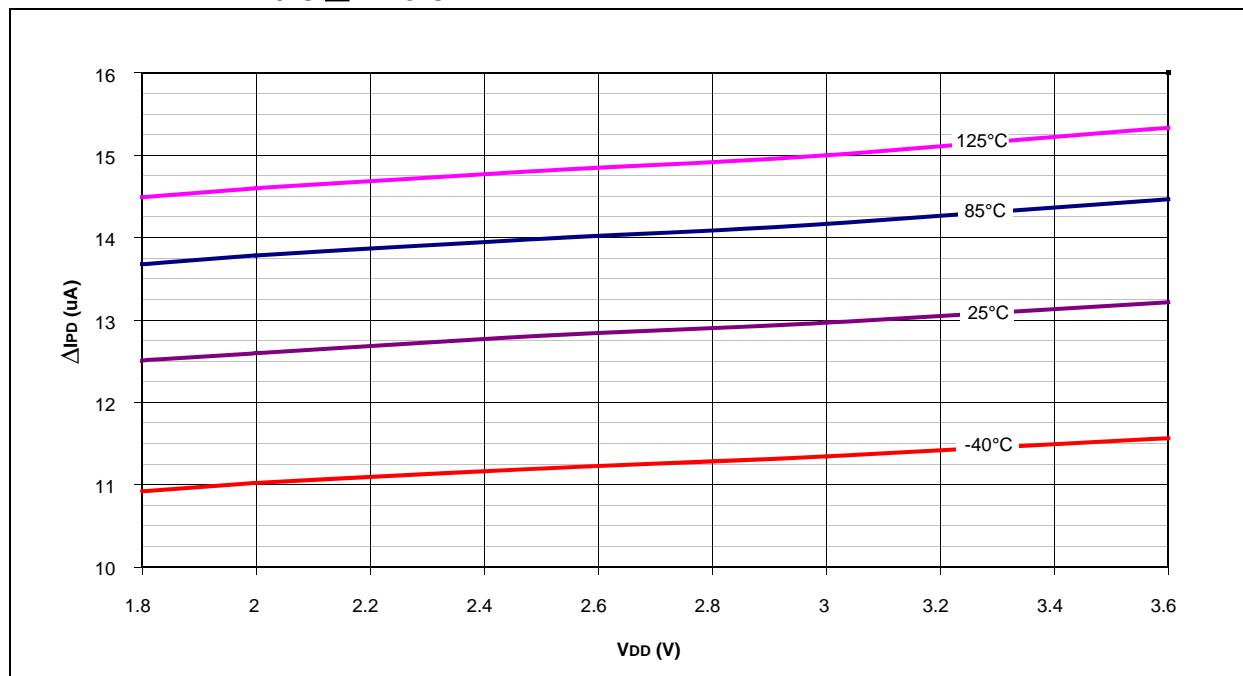


图 27-26: PIC18F4XK20/PIC18F2XK20 的 ΔI_{COMP} ——低功耗模式下比较器的 ΔI_{PD} 最大值曲线,
 -40°C 至 $+125^{\circ}\text{C}$



PIC18F2XK20/4XK20

图 27-27: PIC18F4XK20/PIC18F2XK20 的 ΔI_{COMP} ——高功耗模式下比较器的 ΔI_{PD} 典型值曲线,
-40°C 至 +125°C

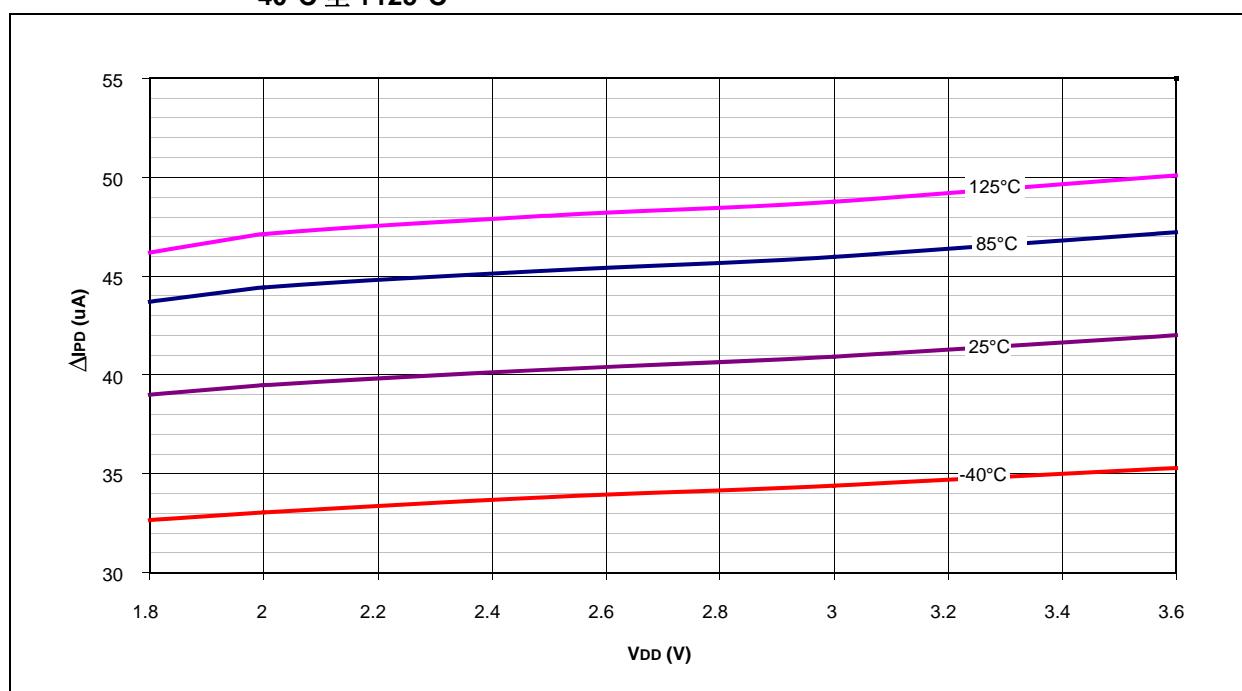


图 27-28: PIC18F4XK20/PIC18F2XK20 的 ΔI_{COMP} ——高功耗模式下比较器的 ΔI_{PD} 最大值曲线,
-40°C 至 +125°C

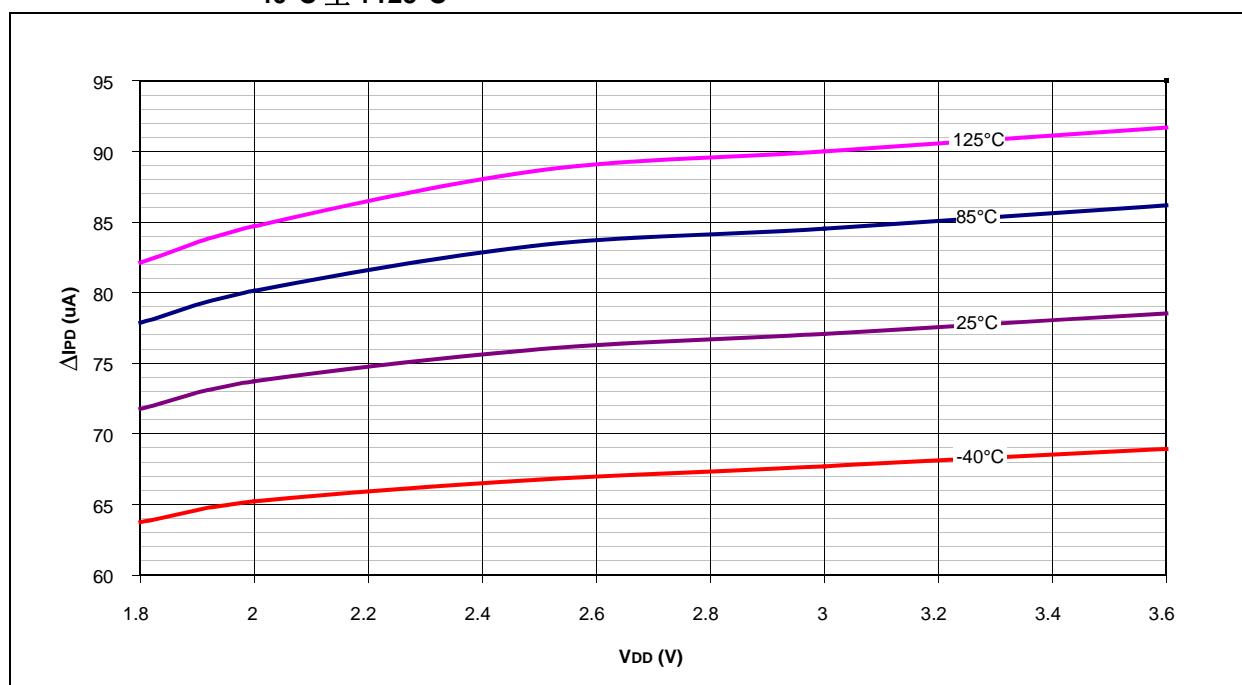


图 27-29: PIC18F4XK20/PIC18F2XK20 的固定参考电压典型值曲线

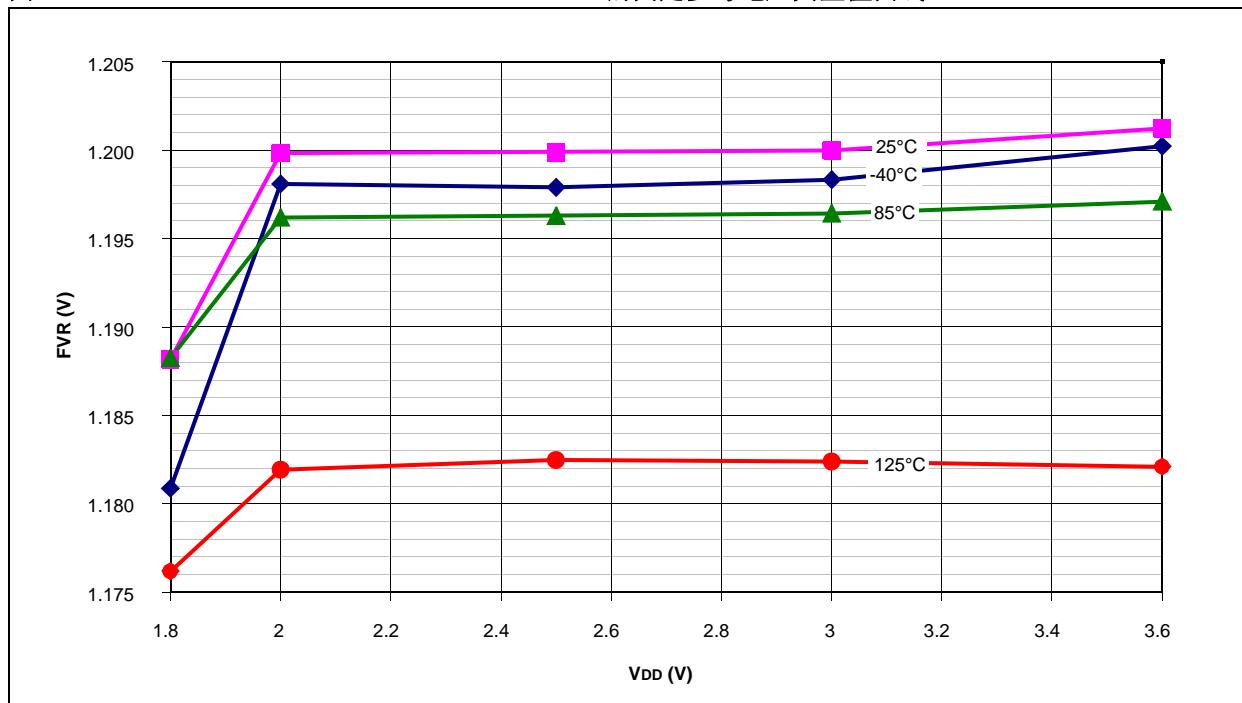
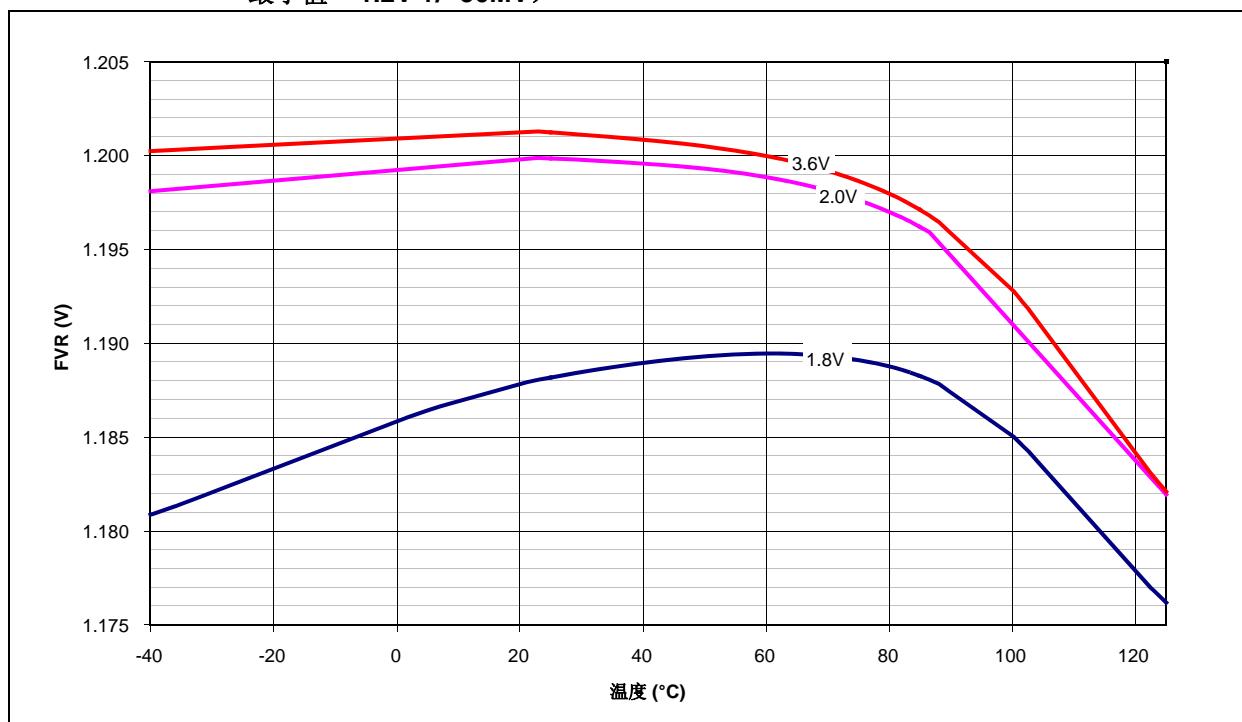


图 27-30: PIC18F4XK20/PIC18F2XK20 的固定参考电压典型值曲线 (-40°C 至 +85°C 时最大值 / 最小值 = 1.2V +/- 50mV)



PIC18F2XK20/4XK20

图 27-31: PIC18F4XK20/PIC18F2XK20 的 TTL 缓冲器 V_{IH} 曲线

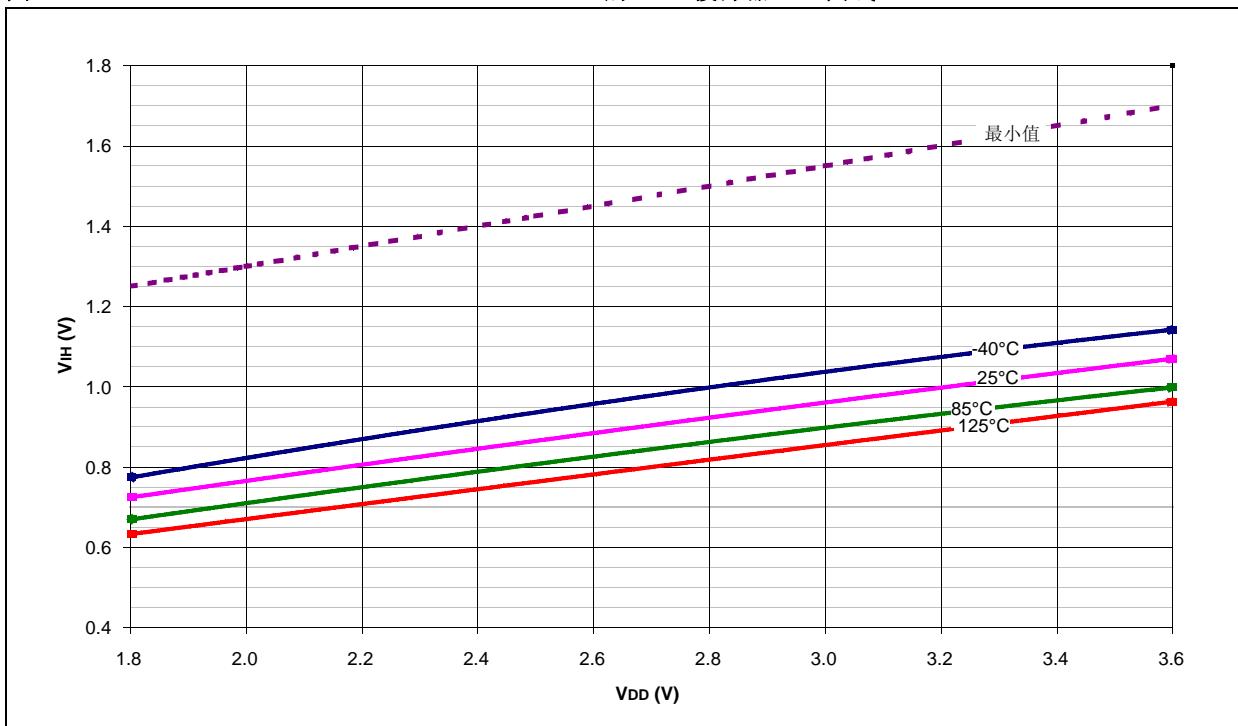


图 27-32: PIC18F4XK20/PIC18F2XK20 的施密特触发器缓冲器 V_{IH} 曲线

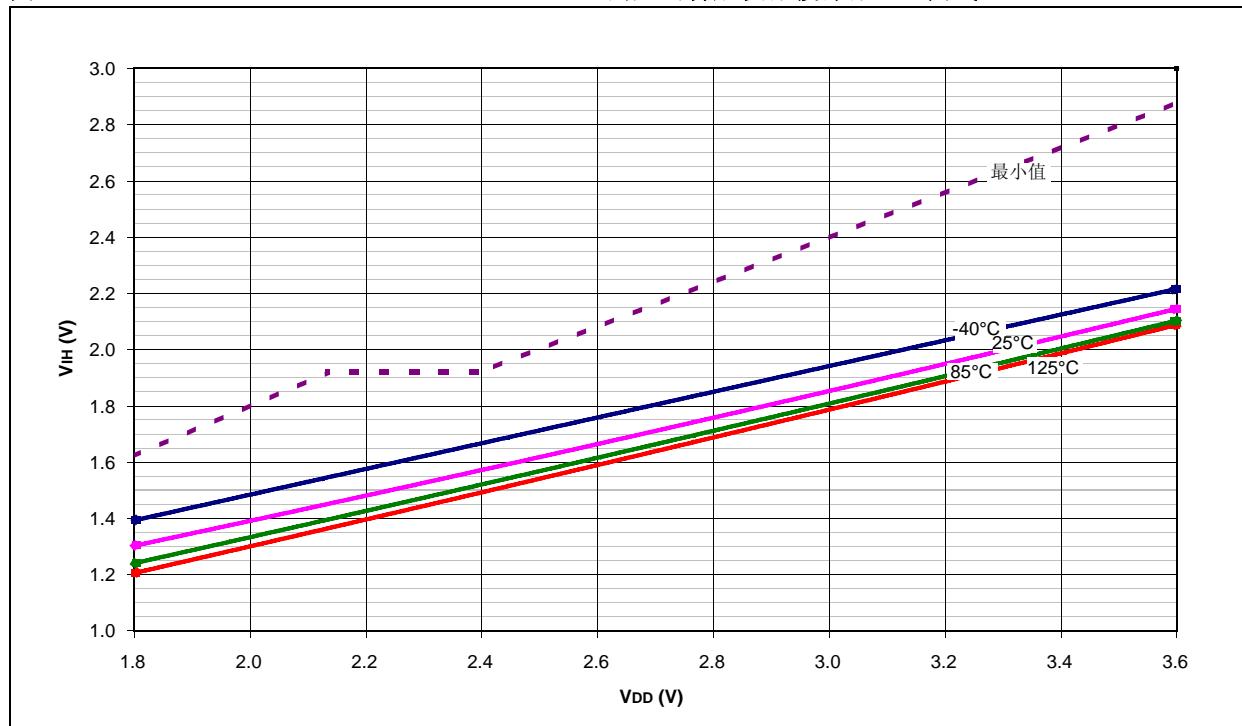


图 27-33: PIC18F4XK20/PIC18F2XK20 的 TTL 缓冲器 V_{IL} 曲线

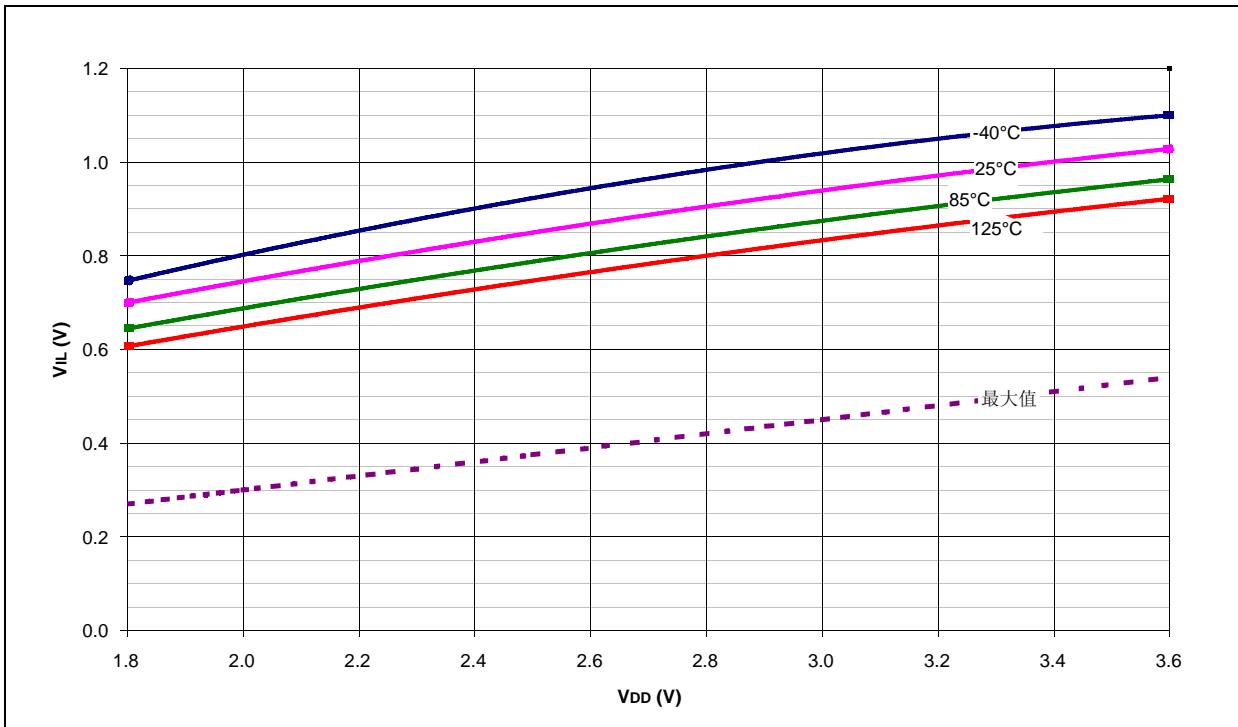
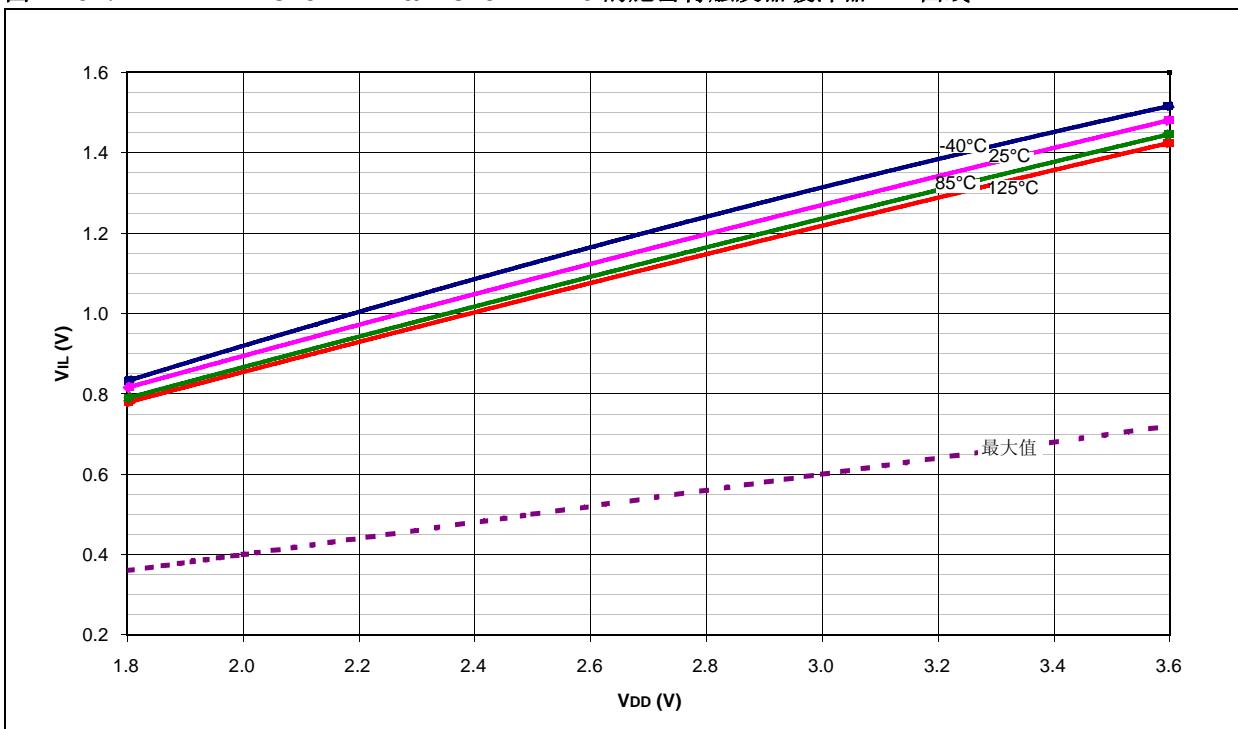


图 27-34: PIC18F4XK20/PIC18F2XK20 的施密特触发器缓冲器 V_{IL} 曲线



PIC18F2XK20/4XK20

图 27-35: PIC18F4XK20/PIC18F2XK20 的 V_{OH} — I_{OH} 曲线 (-40°C 至 +125°C)

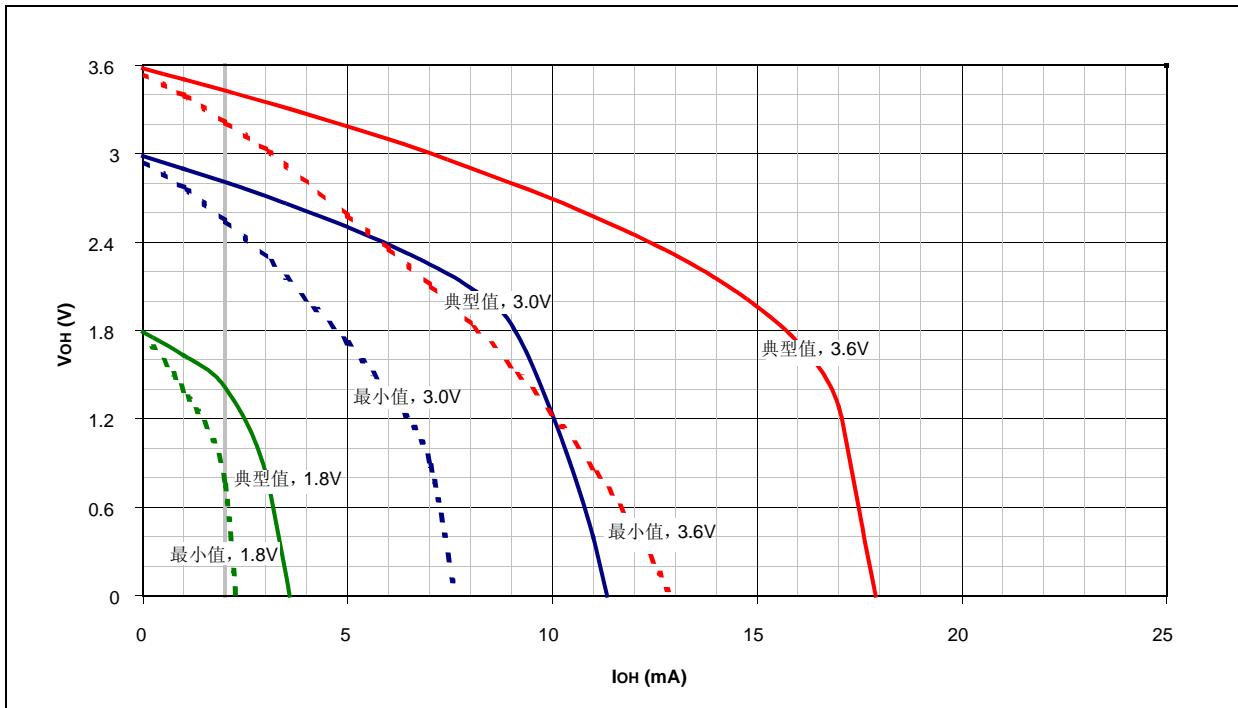


图 27-36: PIC18F4XK20/PIC18F2XK20 的 V_{OL} — I_{OL} 曲线 (-40°C 至 +125°C)

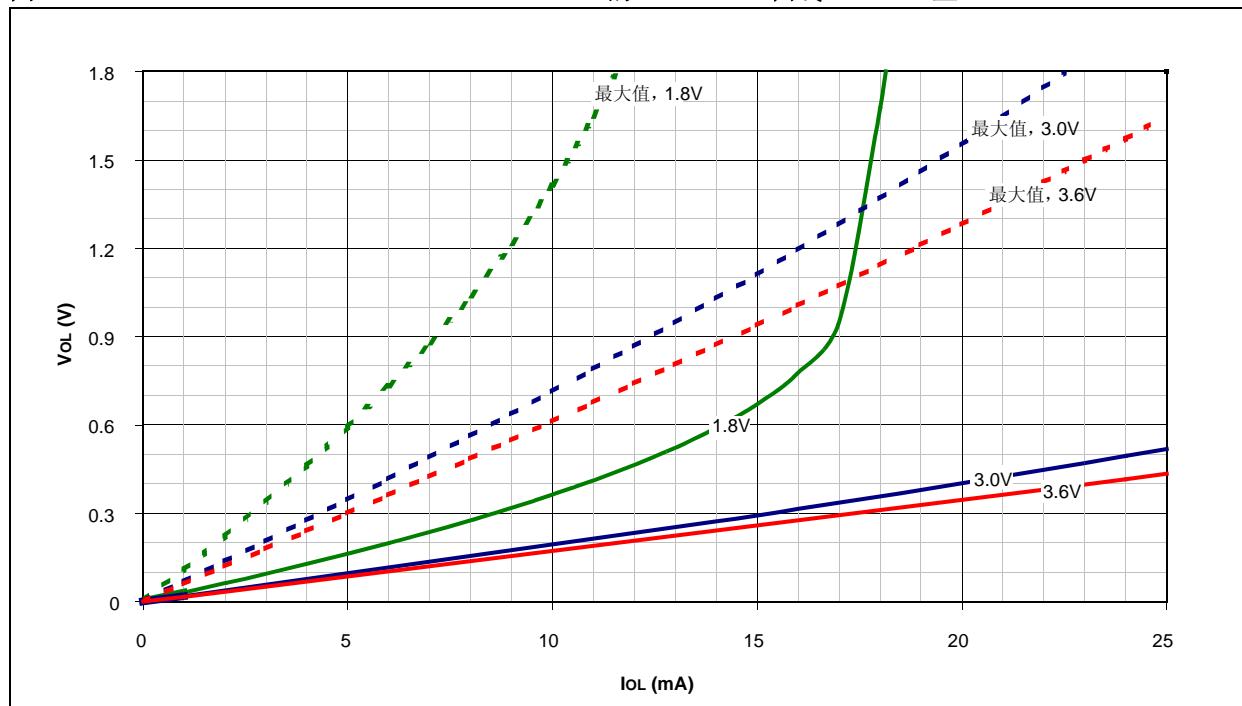
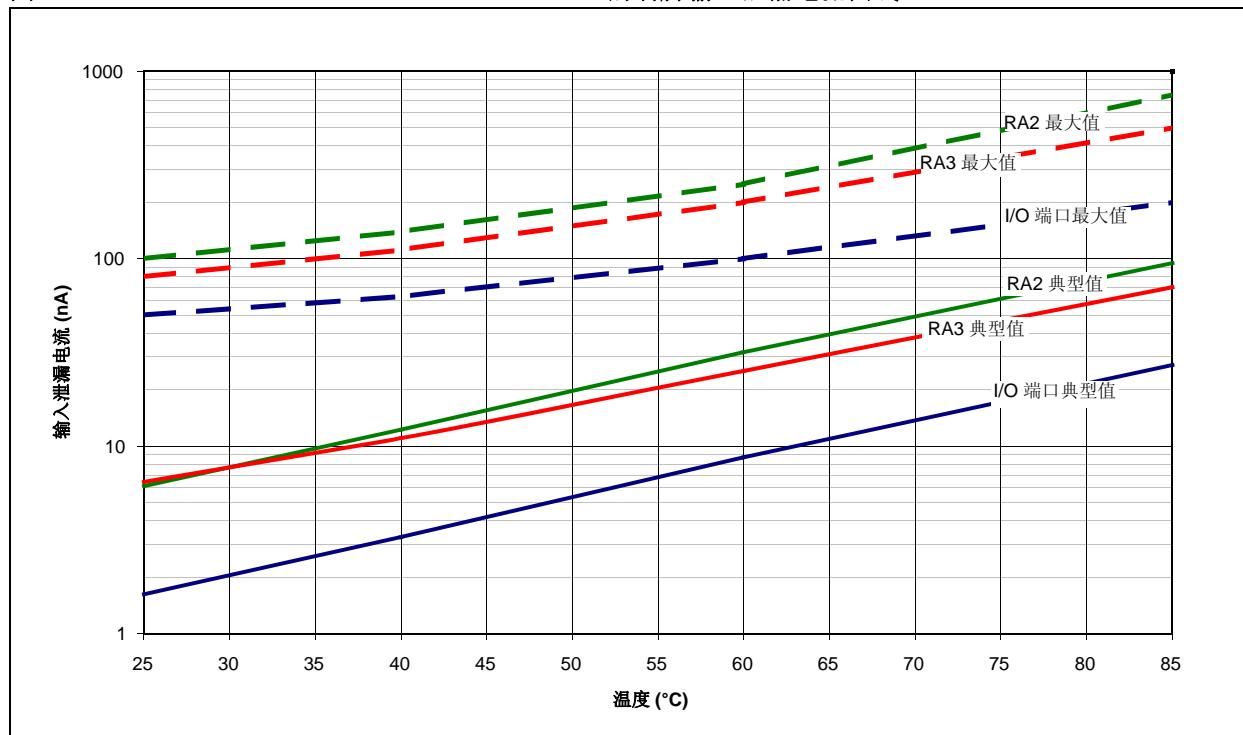


图 27-37: PIC18F4XK20/PIC18F2XK20 的引脚输入泄漏电流曲线



PIC18F2XK20/4XK20

图 27-38: PIC18F4XK20/PIC18F2XK20 的 HF-INTOSC 频率典型值曲线

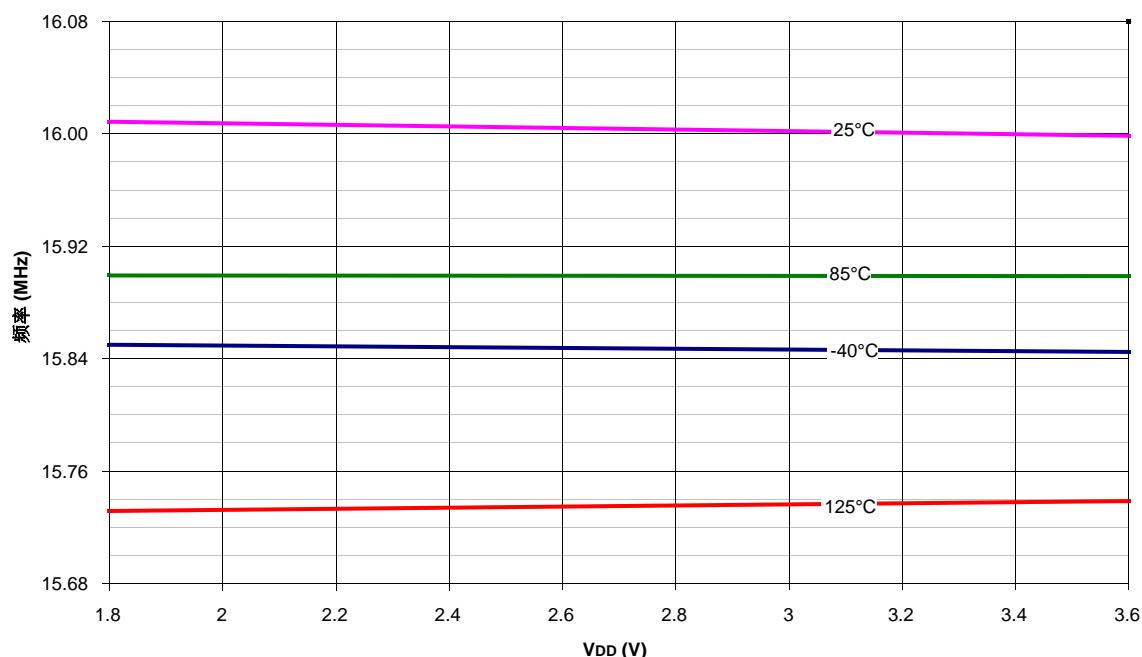


图 27-39: PIC18F4XK20/PIC18F2XK20 的 HF-INTOSC 频率典型值曲线

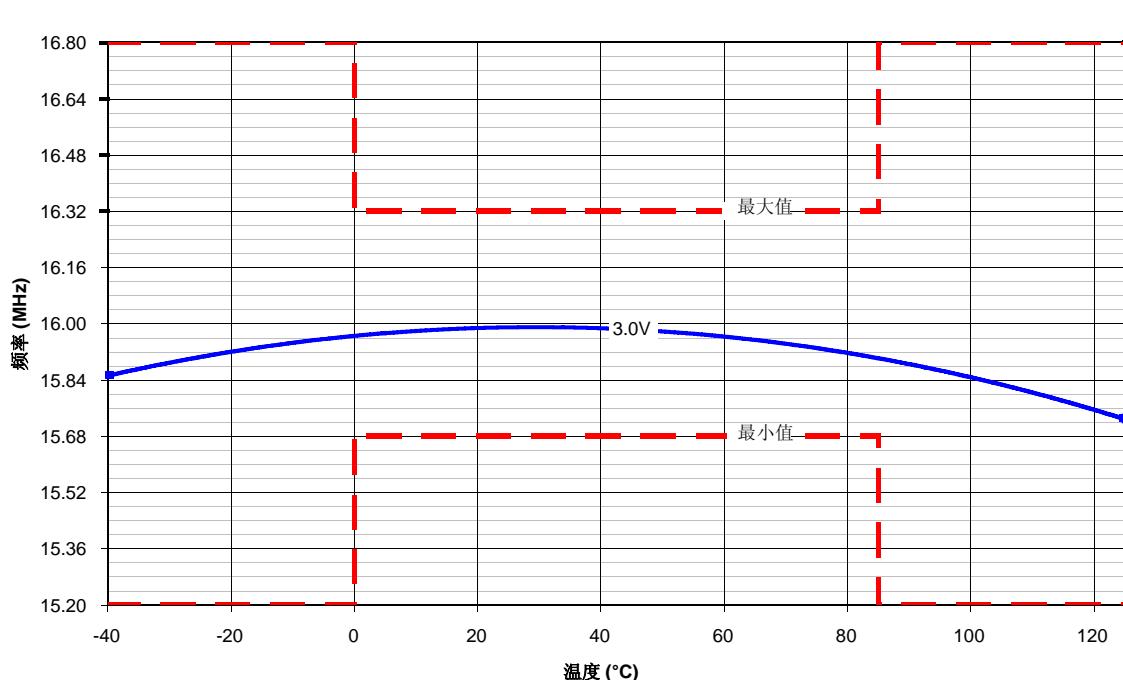


图 27-40:

PIC18F4XK20/PIC18F2XK20 的 LF-INTOSC 频率典型值曲线（最大值 / 最小值 = 31.25 KHZ +/-15%）

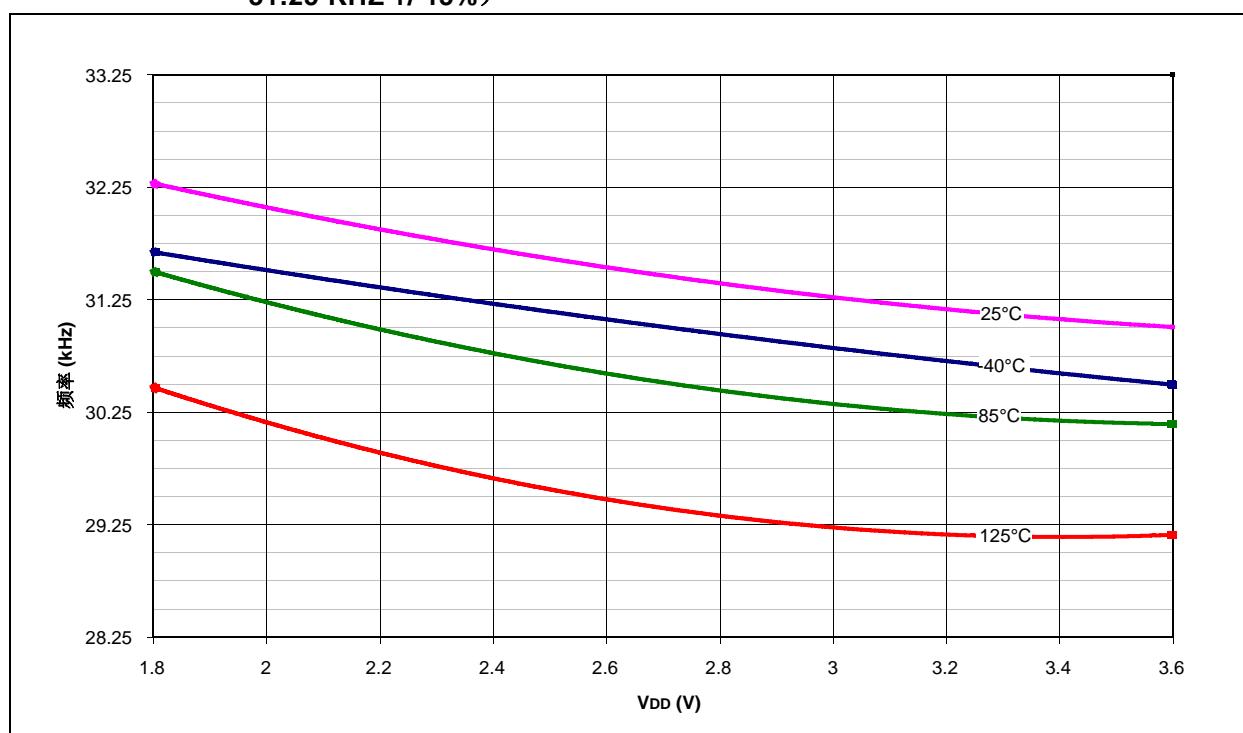
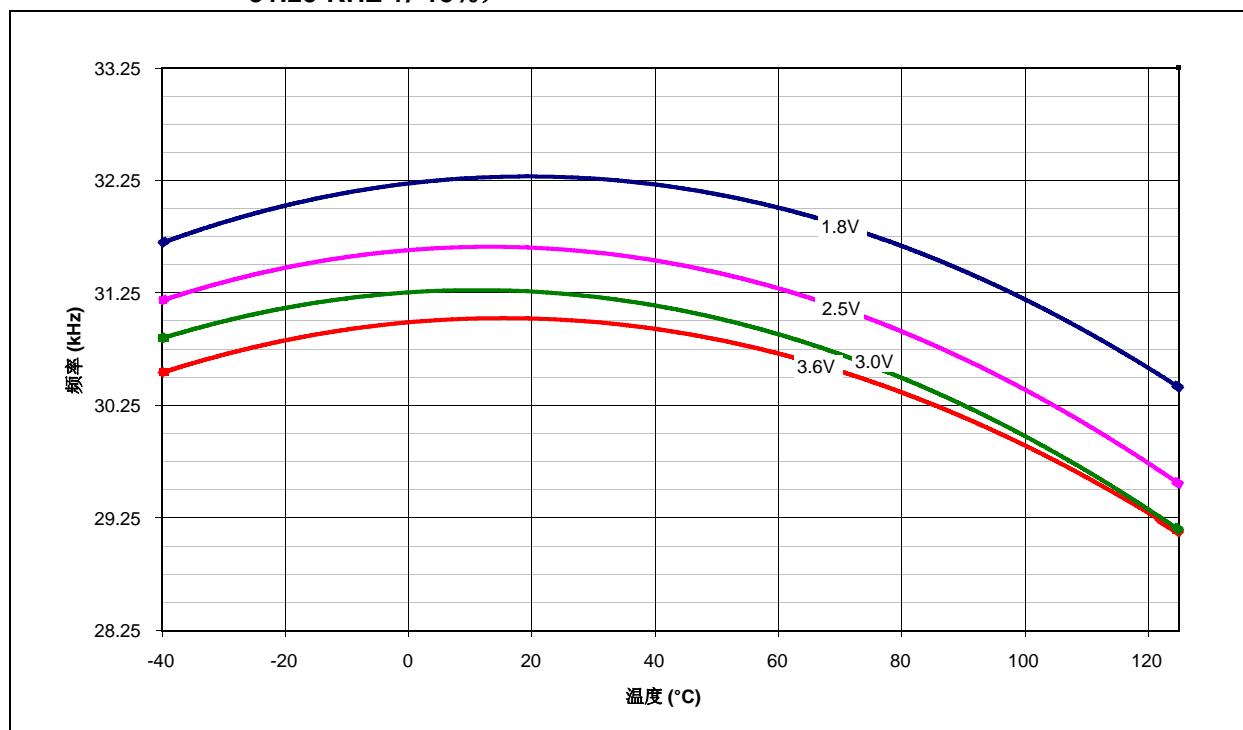


图 27-41:

PIC18F4XK20/PIC18F2XK20 的 LF-INTOSC 频率典型值曲线（最大值 / 最小值 = 31.25 KHZ +/-15%）



PIC18F2XK20/4XK20

注:

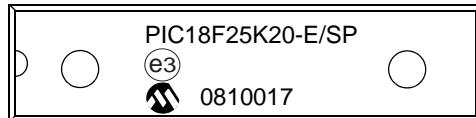
28.0 封装信息

28.1 封装标识信息

28 引脚 PDIP



示例



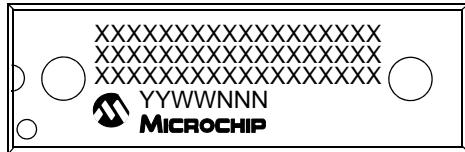
28 引脚 SOIC (7.50 mm)



示例



40 引脚 PDIP



示例

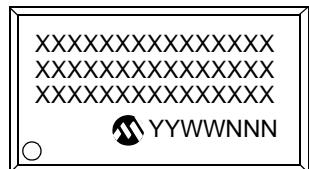


图注:	XX...X 客户信息 Y 年份代码 (日历年的最后一位数字) YY 年份代码 (日历年的最后两位数字) WW 星期代码 (一月一日的星期代码为“01”) NNN 以字母数字排序的追踪代码 (e3) 雾锡 (Matte Tin, Sn) 的 JEDEC 无铅标志 * 本封装为无铅封装。JEDEC 无铅标志 (e3) 标示于此种封装的外包装上。
注:	Microchip 元器件编号如果无法在同一行内完整标注, 将换行标出, 因此会限制表示客户信息的字符数。

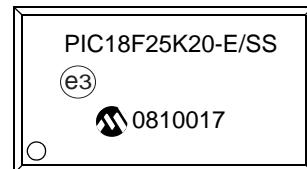
PIC18F2XK20/4XK20

封装标识信息（续）

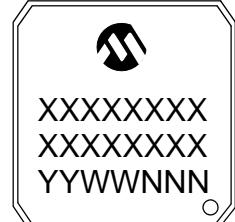
28 引脚 SSOP



示例



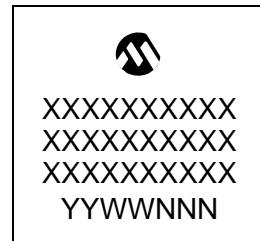
28 引脚 QFN



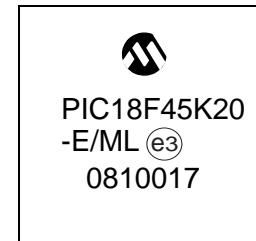
示例



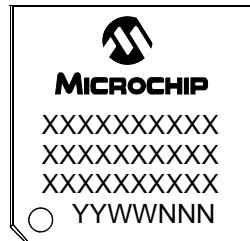
44 引脚 QFN



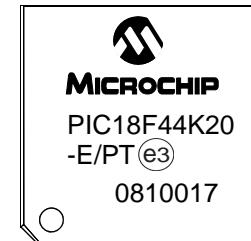
示例



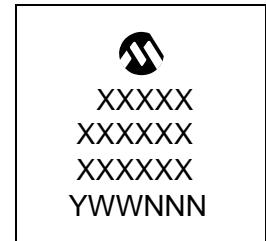
44 引脚 TQFP



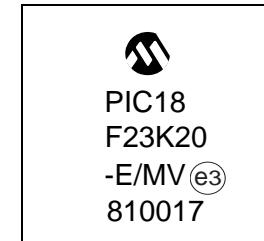
示例



28 引脚 UQFN



示例

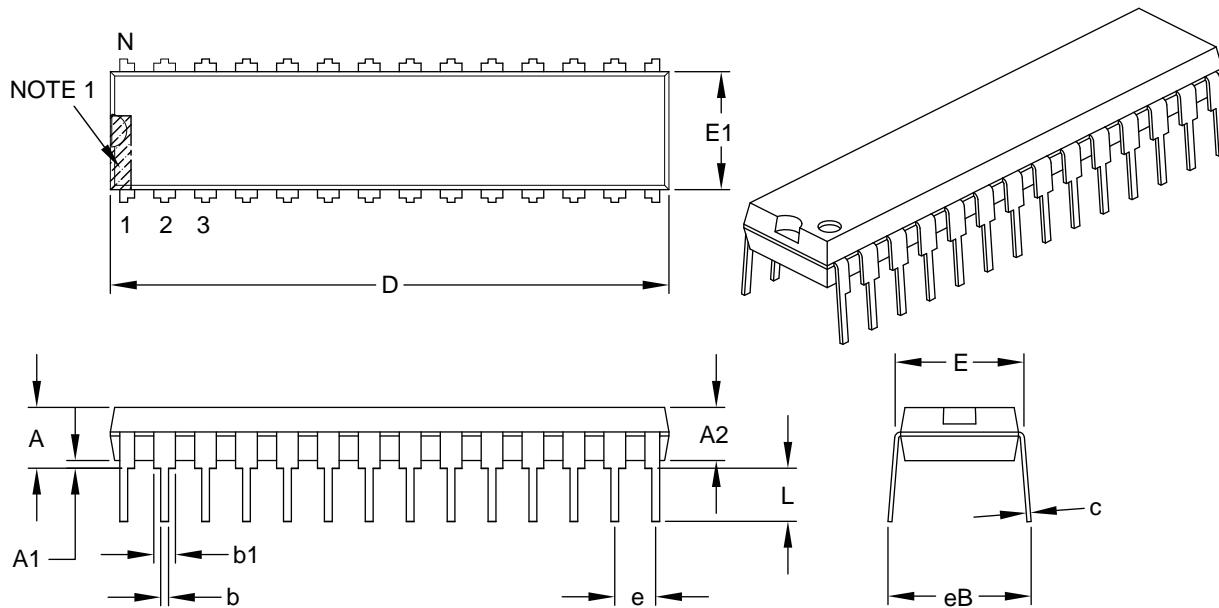


28.2 封装详细信息

以下部分将介绍各种封装的技术细节。

28 引脚窄型塑封双列直插式封装 (SP) —— 主体 300 mil [SPDIP]

注：最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



	Units	INCHES		
	Dimension Limits	MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		.100 BSC	
Top to Seating Plane	A	—	—	.200
Molded Package Thickness	A2	.120	.135	.150
Base to Seating Plane	A1	.015	—	—
Shoulder to Shoulder Width	E	.290	.310	.335
Molded Package Width	E1	.240	.285	.295
Overall Length	D	1.345	1.365	1.400
Tip to Seating Plane	L	.110	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.050	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	—	—	.430

Notes:

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.

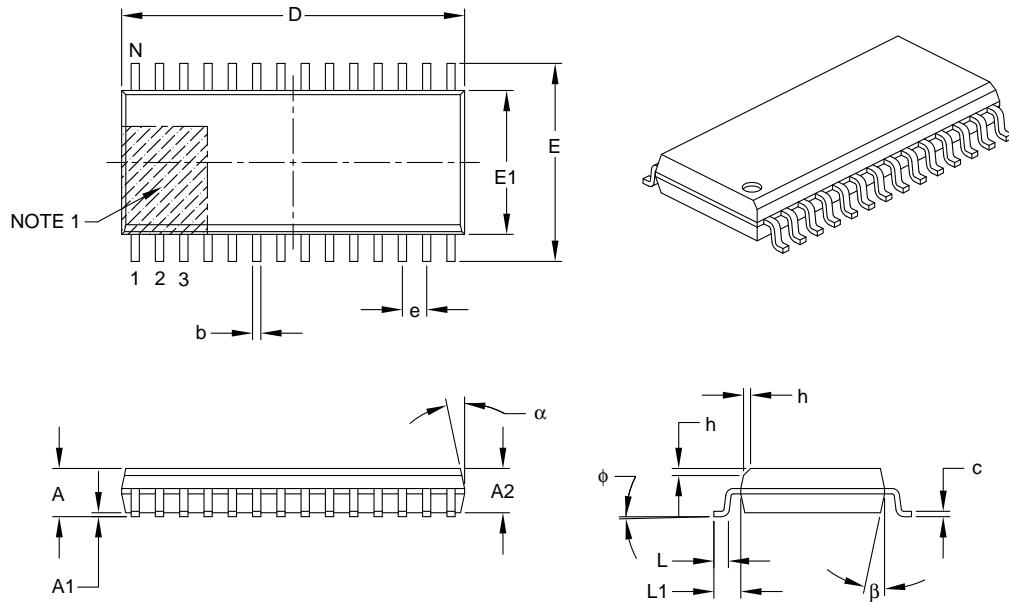
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-070B

PIC18F2XK20/4XK20

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。



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e	1.27 BSC		
Overall Height	A	—	—	2.65
Molded Package Thickness	A2	2.05	—	—
Standoff §	A1	0.10	—	0.30
Overall Width	E	10.30 BSC		
Molded Package Width	E1	7.50 BSC		
Overall Length	D	17.90 BSC		
Chamfer (optional)	h	0.25	—	0.75
Foot Length	L	0.40	—	1.27
Footprint	L1	1.40 REF		
Foot Angle Top	phi	0°	—	8°
Lead Thickness	c	0.18	—	0.33
Lead Width	b	0.31	—	0.51
Mold Draft Angle Top	alpha	5°	—	15°
Mold Draft Angle Bottom	beta	5°	—	15°

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.

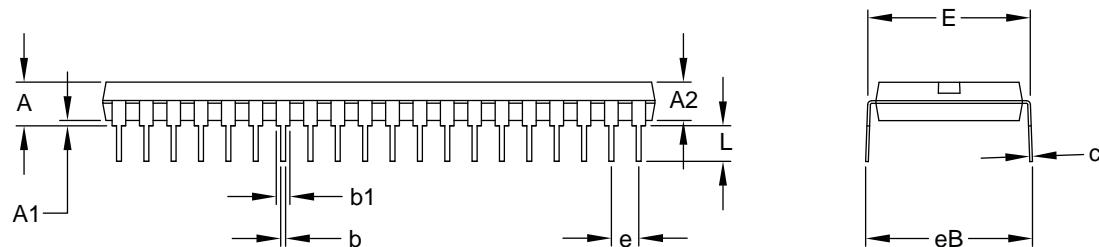
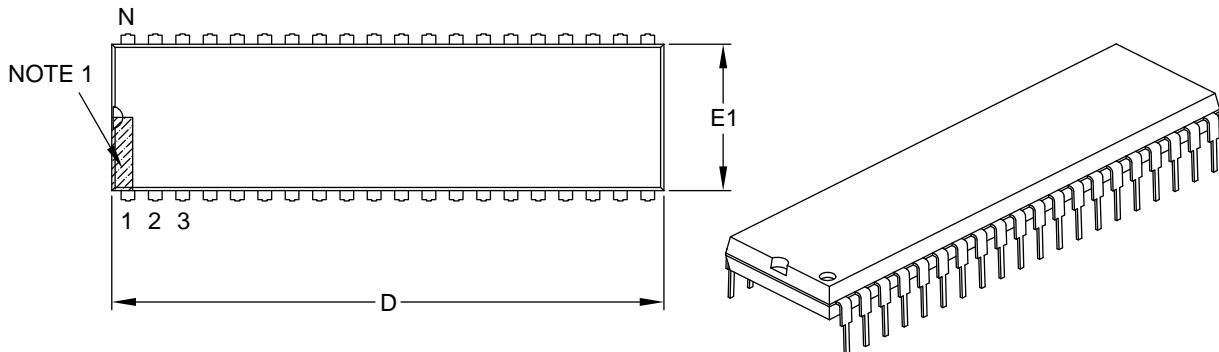
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-052B

PIC18F2XK20/4XK20

40 引脚塑封双列直插式封装 (P) —— 主体 600 mil [PDIP]

注：最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



Dimension Limits		Units	INCHES		
			MIN	NOM	MAX
Number of Pins	N			40	
Pitch	e			.100 BSC	
Top to Seating Plane	A	—	—	.250	
Molded Package Thickness	A2	.125	—	.195	
Base to Seating Plane	A1	.015	—	—	
Shoulder to Shoulder Width	E	.590	—	.625	
Molded Package Width	E1	.485	—	.580	
Overall Length	D	1.980	—	2.095	
Tip to Seating Plane	L	.115	—	.200	
Lead Thickness	c	.008	—	.015	
Upper Lead Width	b1	.030	—	.070	
Lower Lead Width	b	.014	—	.023	
Overall Row Spacing §	eB	—	—	.700	

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.

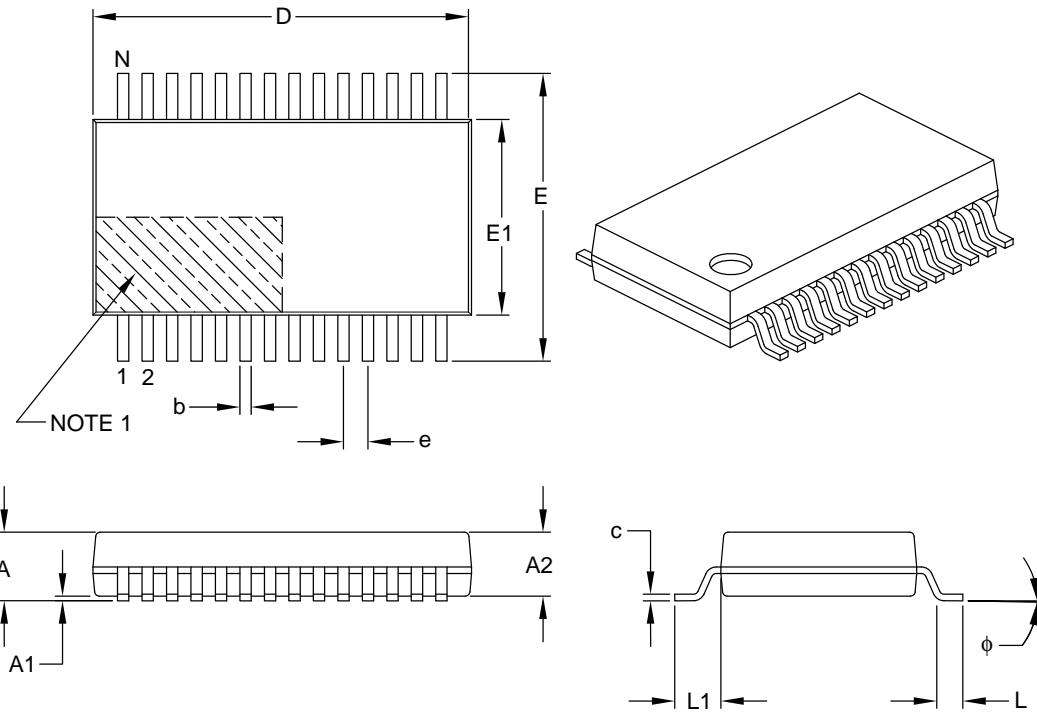
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-016B

PIC18F2XK20/4XK20

28 引脚塑封缩小型小外形封装 (SS) —— 主体 5.30 mm [SSOP]

注：最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



Dimension Limits		MILLIMETERS		
		MIN	NOM	MAX
Number of Pins	N	28		
Pitch	e		0.65 BSC	
Overall Height	A	—	—	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	—	—
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	9.90	10.20	10.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1	1.25 REF		
Lead Thickness	c	0.09	—	0.25
Foot Angle	phi	0°	4°	8°
Lead Width	b	0.22	—	0.38

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

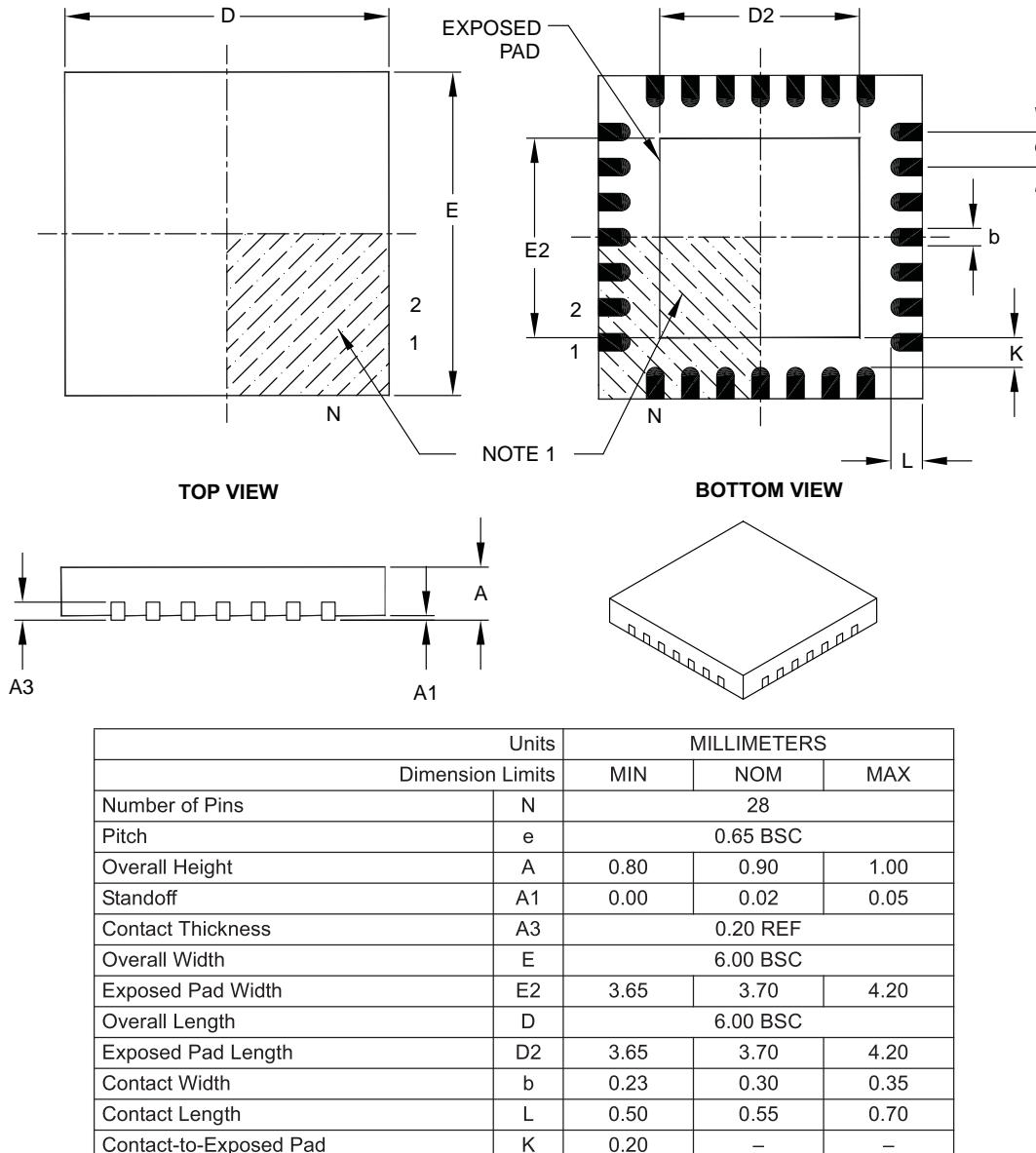
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

28 引脚塑封正方扁平无脚封装 (ML) —— 主体 6x6 mm [QFN], 触点长度为 0.55 mm

注：最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



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.

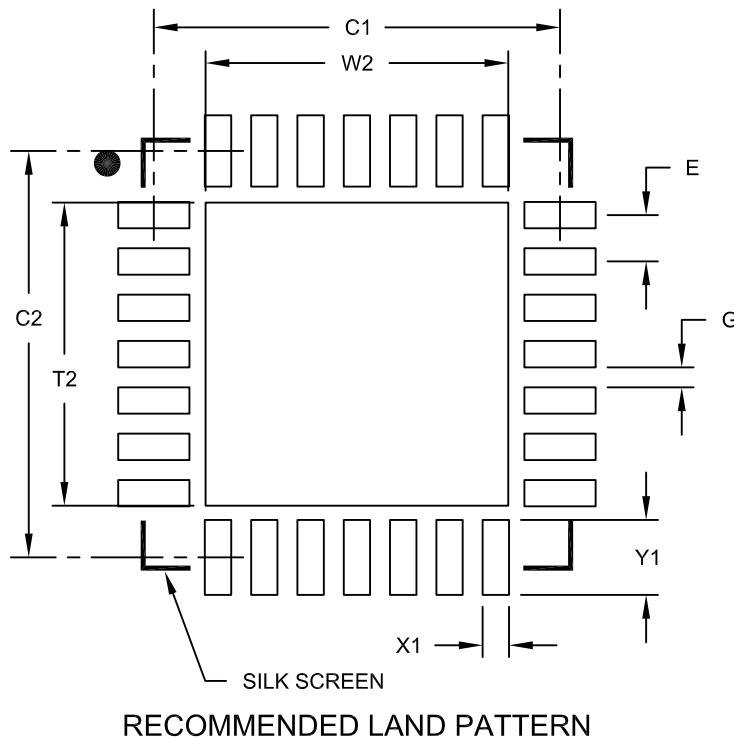
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

PIC18F2XK20/4XK20

28 引脚塑封正方扁平无脚封装 (ML) —— 主体 6x6 mm [QFN], 触点长度为 0.55 mm

注：最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension	Limits	MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
Distance Between Pads	G	0.20		

Notes:

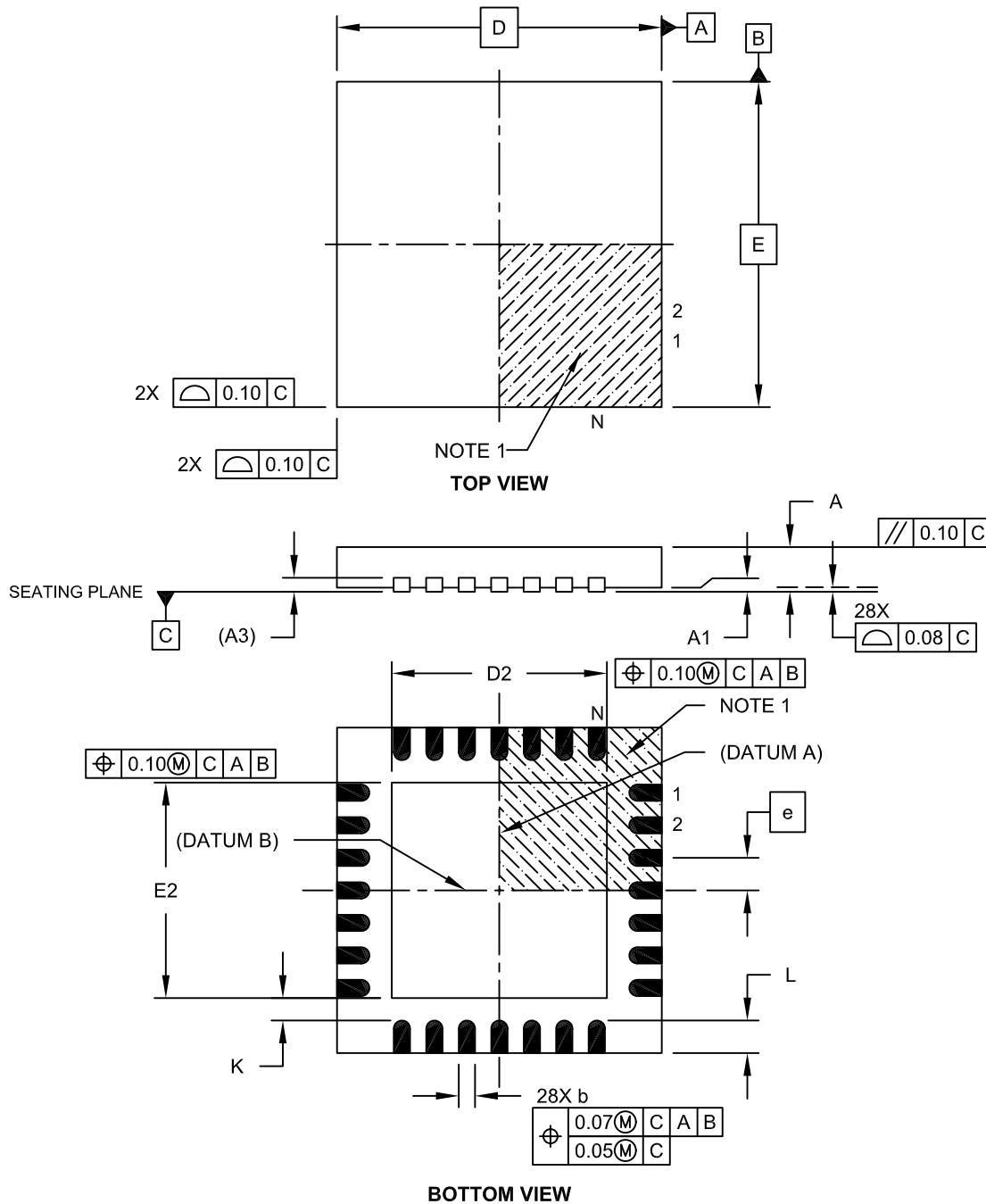
- Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2105A

28 引脚塑封超薄正方扁平无脚封装 (MV) —— 主体 4x4x0.5 mm [UQFN]

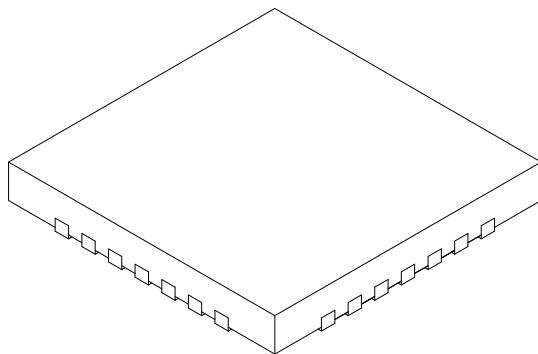
注：最新封装图请至 <http://www.microchip.com/packaging> 查看 Microchip 封装规范。



PIC18F2XK20/4XK20

28 引脚塑封超薄正方扁平无脚封装 (MV) —— 主体 4x4x0.5 mm [UQFN]

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			28				
Pitch	e	0.40 BSC			0.40 BSC				
Overall Height	A	0.45		0.50		0.55			
Standoff	A1	0.00		0.02		0.05			
Contact Thickness	A3	0.127 REF			0.127 REF				
Overall Width	E	4.00 BSC			4.00 BSC				
Exposed Pad Width	E2	2.55		2.65		2.75			
Overall Length	D	4.00 BSC			4.00 BSC				
Exposed Pad Length	D2	2.55		2.65		2.75			
Contact Width	b	0.15		0.20		0.25			
Contact Length	L	0.30		0.40		0.50			
Contact-to-Exposed Pad	K	0.20		-		-			

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

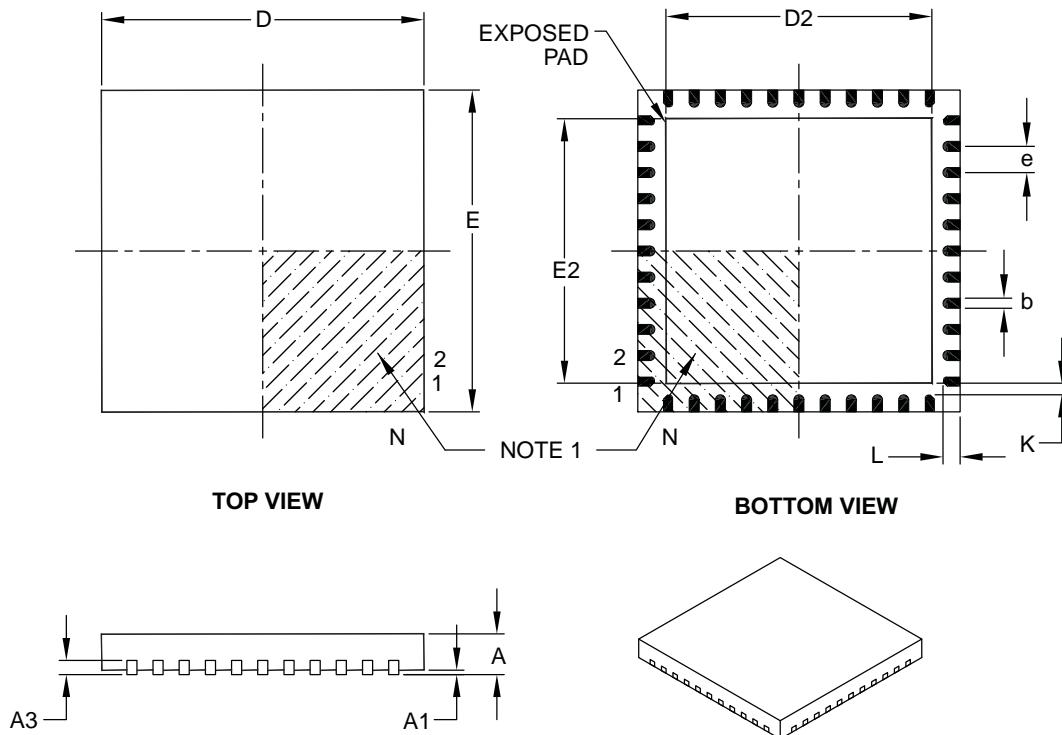
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-152A Sheet 2 of 2

44 引脚塑封正方扁平无脚封装 (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。



		Units	MILLIMETERS		
Dimension Limits			MIN	NOM	MAX
Number of Pins	N		44		
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	8.00	BSC		
Exposed Pad Width	E2	6.30	6.45	6.80	
Overall Length	D	8.00	BSC		
Exposed Pad Length	D2	6.30	6.45	6.80	
Contact Width	b	0.25	0.30	0.38	
Contact Length	L	0.30	0.40	0.50	
Contact-to-Exposed Pad	K	0.20	—	—	

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.

2. Package is saw singulated.

3. Dimensioning and tolerancing per ASME Y14.5M.

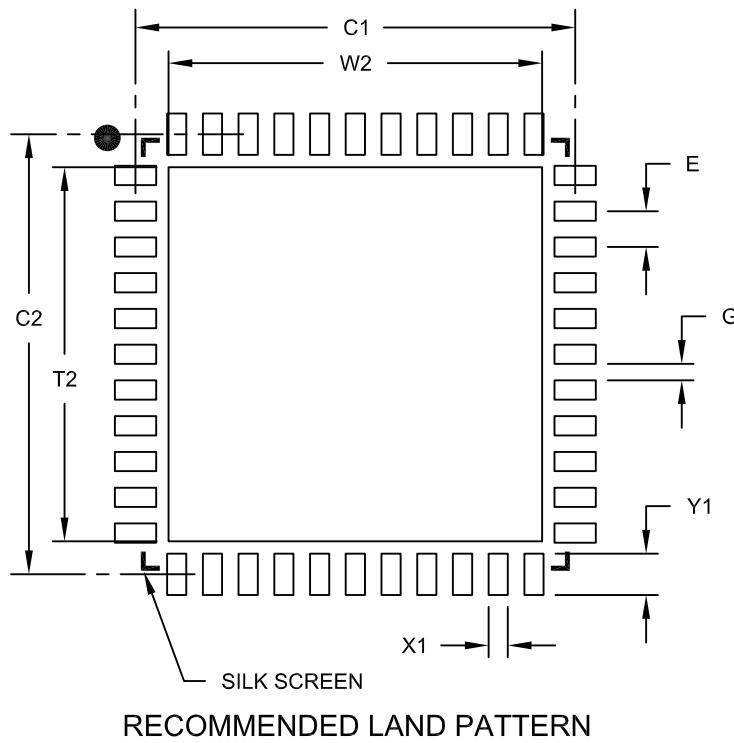
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

PIC18F2XK20/4XK20

44 引脚塑封正方扁平无脚封装 (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。



Units		MILLIMETERS		
Dimension Limits		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:

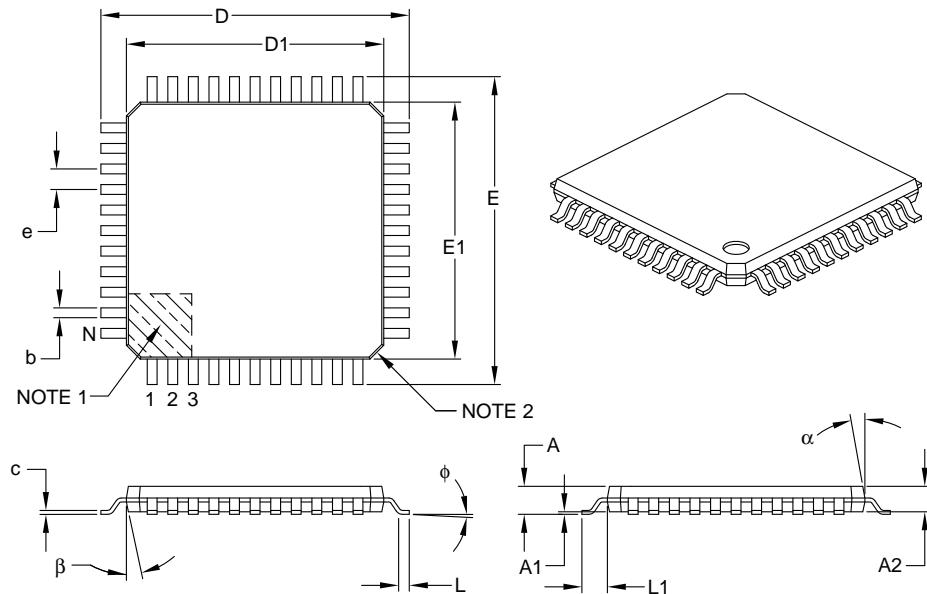
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2103A

44 引脚塑封薄型正方扁平封装 (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			44	
Lead Pitch	e			0.80 BSC	
Overall Height	A	—	—	1.20	
Molded Package Thickness	A2	0.95	1.00	1.05	
Standoff	A1	0.05	—	0.15	
Foot Length	L	0.45	0.60	0.75	
Footprint	L1		1.00 REF		
Foot Angle	phi	0°	3.5°	7°	
Overall Width	E		12.00 BSC		
Overall Length	D		12.00 BSC		
Molded Package Width	E1		10.00 BSC		
Molded Package Length	D1		10.00 BSC		
Lead Thickness	c	0.09	—	0.20	
Lead Width	b	0.30	0.37	0.45	
Mold Draft Angle Top	alpha	11°	12°	13°	
Mold Draft Angle Bottom	beta	11°	12°	13°	

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

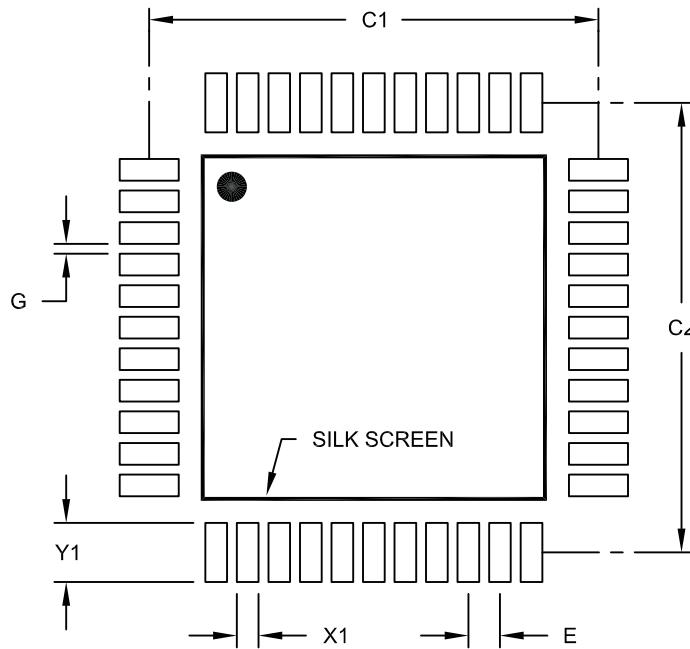
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-076B

PIC18F2XK20/4XK20

44 引脚塑封薄型正方扁平封装 (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

Dimension Limits		MILLIMETERS		
		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:

- 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 (2006 年 7 月)

PIC18F2XK20/4XK20 器件的原始数据手册。

版本 B (2007 年 3 月)

增加了器件编号 PIC18F26K20 和 PIC18F46K20；更换了“开发支持”章节；更换了封装图。

版本 C (2007 年 10 月)

修改了表 1 的 DIL 引脚 34 和 35；表 2 的引脚 22 和 24；表 1-2 的引脚 RB1 和 RB3；表 1-3 的引脚 RB1 和 RB3；修改了第 4.3 节、第 4.4 节、第 4.4.1 节、第 4.4.2 节和第 4.4.4 节；修改了表 4-3 的注 2；修改了表 6-1；修改了第 7.8 节；修改了第 9.2 节；修改了例 10-1 和 10-2；修改了表 10-3 的引脚 RB1 和 RB3；修改了第 12.2 节到第 12.5 节；修改了寄存器 16-1 的 bit3-0；修改了第 16.1 节、第 16.2 节和第 16.4.4 节；修改了寄存器 16-2 的 bit 6-4；修改了表 16-2 的注 2；修改了寄存器 17-1 的 bit 6；修改了寄存器 17-3；修改了表 17-4；修改了寄存器 19-1，增加了注 2；修改了寄存器 20-3 的 bit 5 和 bit 4；修改了寄存器 23-4 的 bit 1；修改了寄存器 23-12 的 bit 7-5；修改了第 23.3 节；修改了第 24.1.1 节的指令集说明；修改了第 26 章 MCLR 上的电压；修改了直流特性 26.2、26.3、26.4、26.5、26.6、26.7、26.8 和 26.10；修改了表 26-1、26-6、26-7、26-9 和 26-23。

版本 D (2008 年 8 月)

更新了外设特点（USART 模块）；删除了第 2.2.6 节（振荡器转换）；修改了第 2.5.3 节和第 2.9 节；增加了第 2.9.3 节（时钟切换时序）；删除了第 2.10.4 节（时钟切换时序）；通篇将 BAUDCTL 替换为 BAUDCON；修改了表 5-2（PLUSW0、PLUSW1 和 PLUSW2）；在表 7-1 中增加了注 1（EEADRH）；修改了第 6.4.4 节和寄存器 16-2（FLTO 引脚）；修改了寄存器 17-2 和 17-5（SSPEN）；修改了寄存器 17-6（SEN）；在图 18-2 之后新增了一个段落；修改了第 18.1.1 节中的“注”；删除了第 18.1.2 节中的“注”；新增了注 2、第 18.1.2.9 节和第 18.1.2.10 节；修改了第 18.3.1 节中的注 1；增加了第 18.3.2 节；修改了第 18.3.5 节；新增了注 2、第 18.4.1.5 节、第 18.4.1.10 节、第 18.4.2.2 节和第 18.4.2.4 节；修改了寄存器 21-1（CVR）；修改了注 1、寄存器 23-6、23-8 和 23-10 以及表 23-3；新增了图 26-1；修改了第 26.2 节、第 26.6 节、第 26.7 节（注 3）、第 26.8 节、第 26.9 节和第 26.10 节；修改了表 26-1、26-2、26-3、26-6、26-7、26-8 和 26-25；更新了封装图。

版本 E (2009 年 4 月)

修改了数据手册标题；修改了功耗管理模式、外设特点和模拟特性；修改了 26.2 直流特性表。

版本 F (2009 年 9 月)

更改了“采用 nanoWatt XLP 的超低功耗管理”章节中的值；在引脚图中新增了注 2；更新了“电气特性”章节；在“DS 特性”章节中增加了图；删除了“初稿”标识；在引脚图中增加了 UQFN；在表 3-1 中增加了 28 引脚 UQFN；更新了 MSSP 章节（寄存器 17-3；将 SSPADD<6:0> 更改为 SSPADD<7:0>）；通过删除第 25.7 节更新了“开发支持”章节；在“封装信息”章节中增加了 28 引脚 UQFN 封装标识图和 28 引脚塑封超薄正方扁平无脚封装（MV）——主体 4X4X0.5 mm（UQFN）封装图；其他少量修正。

PIC18F2XK20/4XK20

附录 B: 器件差异

表 B-1 给出了本数据手册中所列器件之间的差异。

表 B-1: 器件差异

特性	PIC18F23K20	PIC18F24K20	PIC18F25K20	PIC18F26K20	PIC18F43K20	PIC18F44K20	PIC18F45K20	PIC18F46K20
程序存储器 (字节)	8192	16384	32768	65536	8192	16384	32768	65536
程序存储器 (指令)	4096	8192	16384	32768	4096	8192	16384	32768
中断源	19	19	19	19	20	20	20	20
I/O 端口	端口 A、B、C 或 (E)	端口 A、B、C 或 (E)	端口 A、B、C 或 (E)	端口 A、B、C 或 (E)	端口 A、B、C、D 和 E			
捕捉 / 比较 / PWM 模块	1	1	1	1	1	1	1	1
增强型捕捉 / 比较 / PWM 模块	1	1	1	1	1	1	1	1
并行通信 (PSP)	无	无	无	无	有	有	有	有
10 位模数转换模块	11 路输入通道	11 路输入通道	11 路输入通道	11 路输入通道	14 路输入通道	14 路输入通道	14 路输入通道	14 路输入通道
封装	28 引脚 PDIP 28 引脚 SOIC 28 引脚 SSOP 28 引脚 QFN 28 引脚 UQFN	28 引脚 PDIP 28 引脚 SOIC 28 引脚 SSOP 28 引脚 QFN	28 引脚 PDIP 28 引脚 SOIC 28 引脚 SSOP 28 引脚 QFN	28 引脚 PDIP 28 引脚 SOIC 28 引脚 SSOP 28 引脚 QFN	40 引脚 PDIP 44 引脚 TQFP 44 引脚 QFN			

索引

A

A/D

放电	267
模拟端口引脚, 配置	275
特殊事件触发信号 (ECCP)	174
相关的寄存器	275
选择和配置采集时间	264
转换次数	266
转换器特性	399
ACKSTAT	225
ACKSTAT 状态标志	225
ADC	263
采集要求	273
参考电压 (VREF)	264
端口配置	264
功耗管理	267
工作原理	266
计算采集时间	273
结果格式	265
框图	263
模拟信号源阻抗	273
内部采样开关阻抗 (Rss)	273
配置	264
启动 A/D 转换	265
特殊事件触发器	267
通道选择	264
休眠期间的操作	267
中断	265
转换步骤	268
转换时钟	264
ADCON0 寄存器	269
ADCON1 寄存器	270
ADCON2 寄存器	271
ADDFSR	356
ADDLW	319
ADDWF	319
ADDWFC	320
ADDULNK	356
ADRESH 寄存器 (ADFM = 0)	272
ADRESH 寄存器 (ADFM = 1)	272
ADRESL 寄存器 (ADFM = 0)	272
ADRESL 寄存器 (ADFM = 1)	272
ANDLW	320
ANDWF	321
ANSELH 寄存器	137
ANSEL 寄存器	136
ANSEL (端口模拟控制)	136

B

BAUDCON 寄存器	246
BC	321
BCF	322
BF	225
BF 状态标志	225
BN	322
BNC	323
BNN	323
BNOV	324
BNZ	324
BOR。请参见欠压复位。	
BOV	327
BRA	325
BRG。请参见波特率发生器。	

BSF	325
BTFSC	326
BTFSS	326
BTG	327
BZ	328
版本历史	437
比较 (CCP 模块)	147
CCPRx 寄存器	147
Timer1/Timer3 模式选择	147
软件中断	147
特殊事件触发器	147, 172
相关的寄存器	148
引脚配置	147
比较 (ECCP 模块)	174
特殊事件触发器	174
比较器	
复位的影响	281
工作原理	277
相关的寄存器	286
响应时间	279
休眠期间的操作	281
比较器参考电压 (CVREF)	
复位的影响	281, 287
概述	287
相关的寄存器	290
响应时间	279
休眠期间的操作	287
比较器规范	379
比较器模块	277
C1 输出状态与输入条件	279
编程, 器件指令	313
变更通知客户服务	15
表读 / 表写	68
表指针操作 (表)	92
标准指令	313
并行从端口 (PSP)	130, 139
CS (片选)	139
PORTD	139
RD (读输入)	139
WR (写输入)	139
相关的寄存器	141
选择 (PSPMODE 位)	130, 139
波特率发生器	221
不同情形下的延时 (表)	55
捕捉 / 比较 / PWM (CCP)	143
CCPRxH 寄存器	144
CCPRxL 寄存器	144
CCP 模式和定时器资源	144
PWM 模式	149
PWM 频率和分辨率示例	
Fosc = 20 MHZ	151
Fosc = 40 MHZ	151
Fosc = 8 MHZ	151
复位的影响	152
设置操作	152
系统时钟频率改变	152
休眠模式下的操作	152
占空比	150
PWM 周期	150
设置 PWM 操作	152
比较模式。请参见比较。	
捕捉模式。请参见捕捉。	
两个 CCP 模块的相互关系	144

PIC18F2XK20/4XK20

模块配置	144
捕捉 (CCP 模块)	145
CCPRxH:CCPRxL 寄存器	145
CCP 引脚配置	145
Timer1/Timer3 模式选择	145
软件中断	145
相关的寄存器	148
预分频器	145
捕捉 (ECCP 模块)	174
C	
CALL	328
CALLW	357
CCP1CON 寄存器	173
CCP2CON 寄存器	143
CLRF	329
CLRWDT	329
CM1CON0 寄存器	282
CM2CON0 寄存器	283
CM2CON1 寄存器	285
COMF	330
CONFIG1H 寄存器	299
CONFIG2H 寄存器	300
CONFIG2L 寄存器	300
CONFIG3H 寄存器	301
CONFIG4L 寄存器	301
CONFIG5H 寄存器	302
CONFIG5L 寄存器	302
CONFIG6H 寄存器	303
CONFIG6L 寄存器	303
CONFIG7H 寄存器	304
CONFIG7L 寄存器	304
CPFSEQ	330
CPFGT	331
CPFSLT	331
CPU 的特殊功能	297
CVREF 参考电压规范	379
C 编译器	
MPLAB C18	364
参考电压	
VR 稳定	288
固定参考电压 (FVR)	288
参考电压。请参见比较器参考电压 (CVREF)	
参考电压 (VR)	
规范	379
程序存储器	
查找表	68
代码保护	309
复位向量	65
和扩展指令集	87
映射和堆栈 (图)	65
指令	70
双字	70
中断向量	65
程序计数器	66
PCLATH 和 PCLATU 寄存器	66
PCL、PCH 和 PCU 寄存器	66
程序校验和代码保护	308
相关的寄存器	308
串行时钟, SCK	193
串行数据输出 (SDO)	193
串行数据输入 (SDI)	193
串行外设接口。请参见 SPI 模式。	
从选择同步	199
从选择 (SS)	193
存储器构成	65
程序存储器	65
数据存储器	71
存储区选择寄存器 (BSR)	71
D	
DAW	332
DCFNSZ	333
DECFL	332
DECFSZ	333
DEVID1 寄存器	305
DEVID2 寄存器	305
代码保护	297
代码示例	
16 x 16 无符号乘法程序	106
16 x 16 有符号乘法程序	106
8 x 8 无符号乘法程序	105
8 x 8 有符号乘法程序	105
A/D 转换	268
擦除闪存程序存储器的一行	94
初始化 PORTA	121
初始化 PORTB	124
初始化 PORTC	127
初始化 PORTD	130
初始化 PORTE	133
读闪存程序存储器的一个字	93
读数据 EEPROM	101
改变捕捉预分频比	145
将 STATUS、WREG 和 BSR 寄存器的值保存在	
RAM 中	119
快速寄存器堆栈	68
实现 Timer1 实时时钟	164
使用间接寻址清零 RAM	83
使用偏移量值的计算 GOTO	68
数据 EEPROM 刷新程序	102
写闪存程序存储器	96 - 97
写数据 EEPROM	101
装载 SSPBUF (SSPSR) 寄存器	196
单电源 ICSP 编程。	
低电压 ICSP 编程。请参见单电源 ICSP 编程	
电气特性	367
读者反馈	16
对标准 PIC 的影响	360
堆栈满 / 下溢复位	68
E	
ECCPAS 寄存器	183
EECON1 寄存器	91, 100
EUSART	235
波特率发生器 (BRG)	
波特率误差, 计算	247
波特率, 异步模式	248
高波特率选择 (BRGH 位)	247
公式	247
相关的寄存器	247
自动波特率检测	251
时钟极性	
同步模式	255
数据极性	
同步模式	255
异步发送	237
异步接收	240
同步从模式	
发送	260
接收	261
相关的寄存器, 接收	261
同步主模式	255, 260

发送	255	运行模式	44
接收	258	PRI_RUN	44
相关的寄存器, 发送	257, 260	SEC_RUN	44
相关的寄存器, 接收	259	功耗管理模式对各种时钟源的影响	36
异步模式	237	固件指令	313
12 位间隔字符发送和接收	254	故障保护时钟监视器	40, 297
波特率发生器 (BRG)	247	复位或从休眠中唤醒	40
发送器	237	故障保护操作	40
接收到间隔字符时自动唤醒	252	故障保护检测	40
接收器	240	故障保护条件清除	40
设置带地址检测的 9 位模式	242	广播呼叫地址支持	218
时钟精确性	244		
相关的寄存器, 发送	239	H	
相关的寄存器, 接收	243	HLVDCON 寄存器	291
中断		HLVD。请参见高 / 低压检测。	291
异步发送	237	汇编器	
异步接收	241	MPASM 汇编器	364
F			
返回地址堆栈	66	I	
返回堆栈指针 (STKPTR)	67	I/O 端口	121
访问栈顶	66	I/O 引脚说明	
封装信息	423	PIC18F2XK20	16
标识	423	PIC18F4XK20	20
复位	51, 297	I²C	
欠压复位 (BOR)	297	相关的寄存器	234
上电复位 (POR)	297	I²C 模式 (MSSP)	
上电延时定时器 (PWRT)	297	波特率发生器	221
振荡器起振定时器 (OST)	297	串行时钟 (RC3/SCK/SCL)	208
复位的影响		从模式	207
PWM 模式	152	发送	208
G		接收	208
GOTO	334	寻址	207
高 / 低压检测	291	读 / 写位信息 (R/W 位)	207, 208
操作		多主模式	229
休眠期间	295	多主器件通信、总线冲突和仲裁	229
电流消耗	293	复位的影响	229
典型应用	295	工作原理	207
复位的影响	295	广播呼叫地址支持	218
工作原理	292	寄存器	202
启动时间	293	使用 BRG 的 I ² C 时钟频率	221
设置	293	时钟同步和 CKP 位 (SEN = 1)	215
特性	380	时钟延长	214
相关的寄存器	295	10 位从发送模式	214
应用	295	10 位从接收模式 (SEN = 1)	214
功耗管理模式	43	7 位从发送模式	214
对时钟源的影响	36	7 位从接收模式 (SEN = 1)	214
多种休眠功能	44	时钟仲裁	222
和 A/D 操作	267	停止条件时序	228
和 PWM 操作	190	休眠模式下的操作	229
和 SPI 操作	201	应答序列时序	228
汇总 (表)	43	主模式	219
进入	43	重复启动条件时序	224
空闲模式	45	发送	225
PRI_IDLE	47	工作原理	220
RC_IDLE	48	接收	225
SEC_IDLE	47	启动条件时序	223
时钟转换和状态指示	44	总线冲突	
退出空闲和休眠模式	48	重复启动条件期间	232
没有起振延时	49	停止条件期间	233
通过复位	48	ID 存储单元	297, 311
通过 WDT 超时	48	INCF	334
通过中断	48	INCFSZ	335
休眠模式	45	INFSNZ	335
选择	43	INTCON2 寄存器	110
		INTCON3 寄存器	111
		INTCON 寄存器	109 - 111

PIC18F2XK20/4XK20

IORLW	336
IORWF	336
IPR1 寄存器	116
IPR2 寄存器	117
IPR 寄存器	116
J	
寄存器	
ADCON0 (ADC 控制 0)	269
ADCON1 (ADC 控制 1)	270
ADCON2 (ADC 控制 2)	271
ADRESH (ADC 结果高字节, ADFM = 0)	272
ADRESH (ADC 结果高字节, ADFM = 1)	272
ADRESL (ADC 结果低字节, ADFM = 0)	272
ADRESL (ADC 结果低字节, ADFM = 1)	272
ANSELH (端口模拟控制)	137
ANSELH (模拟选择 2)	137
ANSEL (端口模拟控制)	136
ANSEL (模拟选择 1)	136
BAUDCON (波特率控制)	246
BAUDCON (EUSART 波特率控制)	246
CCP1CON (增强型捕捉 / 比较 /PWM 控制)	173
CCP2CON (标准捕捉 / 比较 /PWM 控制)	143
CM1CON0 (C1 控制)	282
CM2CON0 (C2 控制)	283
CM2CON1 (C2 控制)	285
CONFIG1H (配置 1 高字节)	299
CONFIG2H (配置 2 高字节)	300
CONFIG2L (配置 2 低字节)	300
CONFIG3H (配置 3 高字节)	301
CONFIG4L (配置 4 低字节)	301
CONFIG5H (配置 5 高字节)	302
CONFIG5L (配置 5 低字节)	302
CONFIG6H (配置 6 高字节)	303
CONFIG6L (配置 6 低字节)	303
CONFIG7 (配置 7 低字节)	304
CONFIG7 (配置 7 高字节)	304
CVRCN2 (比较器参考电压控制 2)	
CVRCN2 寄存器	290
CVRCN (比较器参考电压控制)	
CVRCN 寄存器	289
DEVID1 (器件 ID 1)	305
DEVID2 (器件 ID 2)	305
ECCPAS (增强型 CCP 自动关闭控制)	183
EECON1 (数据 EEPROM 控制 1)	91, 100
HLVDCON (高 / 低压检测控制)	291
INTCON2 (中断控制 2)	110
INTCON3 (中断控制 3)	111
INTCON (中断控制)	109
IPR1 (外设中断优先级 1)	116
IPR2 (外设中断优先级 2)	117
OSCCON (振荡器控制)	29
OSCTUNE (振荡器调节)	33
PIE1 (外设中断允许 1)	114
PIE2 (外设中断允许 2)	115
PIR1 (外设中断请求 1)	112
PIR2 (外设中断请求 2)	113
PSTRCON (脉冲转向控制)	187
PWM1CON (增强型 PWM 控制)	186
RCON (复位控制)	52, 118
RCREG 寄存器	251
RCSTA (接收状态和控制寄存器)	245
SLRCON (端口斜率控制)	138
SSPADD (MSSP 地址和波特率, SPI 模式)	203
SSPCON1 (MSSP 控制 1, I ² C 模式)	205
SSPCON1 (MSSP 控制 1, SPI 模式)	195
SSPCON2 (MSSP 控制 2, I ² C 模式)	206
SSPMASK (SSP 掩码)	213
SSPSTAT (MSSP 状态, SPI 模式)	194, 204
STATUS	82
STKPTR (堆栈指针)	67
T0CON (Timer0 控制)	155
T1CON (Timer1 控制)	159
T2CON (Timer2 控制)	167
T3CON (Timer3 控制)	169
TRISE (PORTE/PSP 控制)	134
TXSTA (发送状态和控制寄存器)	244
WDTCON (看门狗定时器控制)	307
寄存器的复位状态	58
寄存器文件	77
寄存器文件汇总	79 - 81
计算 GOTO	68
间隔字符 (12 位) 发送和接收	254
间接寻址	84
交流特性	
内部 RC 精度	384
交流 (时序) 特性	381
参数符号体系	381
器件时序规范的负载条件	382
时序条件	382
温度和电压规范	382
接收到间隔字符时唤醒	252
绝对最大值	367
K	
开发支持	363
看门狗定时器 (WDT)	297, 306
编程注意事项	306
控制寄存器	307
相关的寄存器	307
勘误表	10
客户服务	15
客户支持	15
快速操作存储区	
在立即数变址寻址模式下映射	87
快速寄存器堆栈	68
框图	
ADC	263
ADC 传递函数	274
CCP PWM	149
EUSART 发送	235
EUSART 接收	236
MSSP (I ² C 模式)	202
MSSP (I ² C 主模式)	219
MSSP (SPI 模式)	193
PIC18F2XK20	14
PIC18F4XK20	15
PLL (HS 模式)	35
PORTD 和 PORTE (并行从端口)	139
PWM (增强型)	175
Timer0 (16 位模式)	157
Timer0 (8 位模式)	156
Timer1	160
Timer1 (16 位读 / 写模式)	160
Timer2	168
Timer3	170
Timer3 (16 位读 / 写模式)	171
比较模式工作原理	147
比较器 1	278
比较器 2	278
比较器参考电压	288
表读操作	89

表写操作	90
波特率发生器	221
捕捉模式工作原理	146
参考电压输出缓冲示例	289
带外部输入的高 / 低压检测	292
读闪存程序存储器	93
对闪存程序存储器的表写操作	95
故障保护时钟监视器 (FSCM)	40
晶振的工作原理	31
看门狗定时器	306
模拟输入模型	274, 284
片上复位电路	51
时钟源	27
通用 I/O 端口	121
外部 POR 电路 (VDD 缓慢上电的情况)	53
外部 RC 模式	32
谐振器的工作原理	31
中断逻辑	108
扩展指令集	
ADDFSR	356
ADDULNK	356
CALLW	357
MOVSF	357
MOVSS	358
PUSHL	358
SUBFSR	359
SUBULNK	359
和使用 MPLAB 工具	362
使用注意事项	360
语法	355
L	
LFSR	337
立即数变址模式	360
立即数变址寻址 和标准 PIC18 指令	360
M	
Microchip 因特网网站	15
MOVF	337
MOVFF	338
MOVLB	338
MOVLW	339
MOVSF	357
MOVSS	358
MOVWF	339
MPLAB ASM30 汇编器、链接器和库管理器	364
MPLAB PM3 器件编程器	366
MPLAB REAL ICE 在线仿真器系统	365
MPLAB 集成开发环境软件	363
MPLINK 目标链接器 /MPLIB 目标库管理器	364
MSSP	
ACK 脉冲	207, 208
I ² C 模式。请参见 I ² C 模式。	
SPI 模式。请参见 SPI 模式。	
SPI 主 / 从器件连接	197
SSPBUF 寄存器	198
SSPSR 寄存器	198
控制寄存器 (通用)	193
模块概述	193
MULLW	340
MULWF	340
脉冲转向	187
模拟输入连接注意事项	284
模数转换器。请参见 ADC	
N	
NEGF	341
NOP	341
内部采样开关阻抗 (RSS)	273
内部集成电路。请参见 I ² C。	
内部 RC 振荡器 与 WDT 一起使用	306
内部振荡器模块	
HFINTOSC 模式下的 PLL	35
HFINTOSC 频率漂移	34
O	
OSCCON 寄存器	29
OSCTUNE 寄存器	33
P	
P1A/P1B/P1C/P1D。请参见 增强型捕捉 / 比较 /PWM (ECCP)	175
PIE1 寄存器	114
PIE2 寄存器	115
PIE 寄存器	114
PIR1 寄存器	112
PIR2 寄存器	113
PIR 寄存器	112
PLL 倍频器	35
HSPLL 振荡器模式	35
POP	342
PORTA	
LATA 寄存器	121
PORTA 寄存器	121
TRISA 寄存器	121
相关的寄存器	123
PORTB	
LATB 寄存器	124
PORTB 寄存器	124
TRISB 寄存器	124
相关的寄存器	126
PORTC	
LATC 寄存器	127
PORTC 寄存器	127
RC3/SCK/SCL 引脚	208
TRISC 寄存器	127
相关的寄存器	129
PORTD	
LATD 寄存器	130
PORTD 寄存器	130
TRISD 寄存器	130
并行从端口 (PSP) 功能	130
相关的寄存器	132
PORTE	
LATE 寄存器	133
PORTE 寄存器	133
PSP 模式选择 (PSPMODE 位)	130
TRISE 寄存器	133
相关的寄存器	135
POR。请参见上电复位。	
PRI_IDLE 模式	47
PRI_RUN 模式	44
PSP。请参见并行从端口。	
PSTRCON 寄存器	187
PWM1CON 寄存器	186
PWM 模式。请参见增强型捕捉 / 比较 /PWM	175
PWM (CCP 模块) 相关的寄存器	153
PWM (ECCP 模块)	

PIC18F2XK20/4XK20

复位的影响	190
故障保护时钟监视器相关操作	190
脉冲转向	187
在功耗管理模式下的操作	190
转向同步	189
PUSH	342
PUSH 和 POP 指令	67
PUSHL	358
配置寄存器保护	311
配置位	298
Q	
器件差异	438
器件复位定时器	55
PLL 锁定延时定时器	55
上电延时定时器 (PWRT)	55
延时时序	55
器件概述	11
其他特殊功能	12
系列中各器件的详细说明	12
新的内核特性	11
欠压复位 (BOR)	54
检测	54
软件使能	54
在休眠模式下禁止	54
最小使能时间	54
R	
RAM。请参见数据存储器。	
RCALL	343
RC_IDLE 模式	48
RCON 寄存器	52, 118
初始化时位的状态	58
RCREG	242
RCSTA 寄存器	245
RESET	343
RETFIE	344
RETLW	344
RETURN	345
RLCF	345
RLNCF	346
RRCF	346
RRNCF	347
软件模拟器 (MPLAB SIM)	365
S	
SCK	193
SDI	193
SDO	193
SEC_IDLE 模式	47
SEC_RUN 模式	44
SETF	347
SLEEP	348
SLRCON 寄存器	138
SPBRG	247
SPBRGH	247
SPI 模式 (MSSP)	
SPI 时钟	198
串行时钟	193
串行数据输出	193
串行数据输入	193
从模式	199
从选择	193
从选择同步	199
典型连接	197
复位的影响	201
工作原理	196
使能 SPI I/O	197
相关的寄存器	201
在功耗管理模式下的操作	201
主 / 从器件连接	197
主模式	198
总线模式兼容性	201
SS	193
SSPADD 寄存器	203
SSPCON1 寄存器	195, 205
SSPCON2 寄存器	206
SSPMASK 寄存器	213
SSPOV	225
SSPOV 状态标志	225
SSPSTAT 寄存器	194, 204
R/W 位	207, 208
STATUS 寄存器	82
STKPTR 寄存器	67
SWAPF	350
SUBFSR	359
SUBFWB	348
SUBLW	349
SUBWF	349
SUBWFB	350
SUBULNK	359
闪存程序存储器	89
表读与表写	89
表指针	
基于操作的边界	92
表指针边界	92
擦除	94
擦除序列	94
代码保护期间的操作	97
读取	93
控制寄存器	90
EECON1 和 EECON2	90
TABLAT (表锁存) 寄存器	92
TBLPTR (表指针) 寄存器	92
相关的寄存器	97
写入	95
防止误写操作的保护措施	97
写校验	97
意外终止	97
写序列	95
上电复位 (POR)	53
上电延时定时器 (PWRT)	55
延时时序	55
上电延时	36
上电延时定时器 (PWRT)	36
时序图	
A/D 转换	400
CLKO 和 I/O	385
I ² C 从模式广播呼叫地址序列	
(7 位或 10 位地址模式)	218
I ² C 从模式 (10 位发送)	212
I ² C 从模式 (10 位接收, SEN = 0)	211
I ² C 从模式 (10 位接收, SEN = 1)	217
I ² C 从模式 (7 位发送)	210
I ² C 从模式 (7 位接收, SEN = 0)	209
I ² C 从模式 (7 位接收, SEN = 1)	216
I ² C 停止条件接收或发送模式	228
I ² C 主模式 (7 位或 10 位发送)	226
I ² C 主模式 (7 位接收)	227
I ² C 总线启动位 / 停止位	394
I ² C 总线数据	394

PWM 方向改变	181
PWM 输出（低电平有效）	177
PWM 输出（高电平有效）	176
PWM 自动关闭	
固件重启	184
使能自动重启	184
SPI 从模式示例（CKE = 0）	392
SPI 从模式示例（CKE = 1）	393
SPI 模式（从模式， CKE = 0）	200
SPI 模式（从模式， CKE = 1）	200
SPI 模式（主模式）	198
SPI 主模式示例（CKE = 0）	390
SPI 主模式示例（CKE = 1）	391
Timer0 和 Timer1 外部时钟	387
Timer1 递增边沿	161
USART 同步发送（主 / 从）	398
USART 同步接收（主 / 从）	398
半桥 PWM 输出	178, 185
比较器输出	277
并行从端口（PIC18F4XK20）	389
并行从端口（PSP）读	140
并行从端口（PSP）写	140
捕捉 / 比较 / PWM（CCP）	388
重复启动条件	224
重复启动条件期间的总线冲突（情形 1）	232
重复启动条件期间的总线冲突（情形 2）	232
从空闲模式唤醒进入运行模式的转换时序	47
从同步	199
从休眠模式唤醒的转换（HSPLL）	46
带有时钟仲裁的波特率发生器	222
第一个启动位时序	223
发送和应答时的总线冲突	229
发送间隔字符序列	254
复位、看门狗定时器（WDT）、振荡器起振定时器 （OST）和上电延时定时器（PWRT）	386
高 / 低压检测工作原理（VDIRMAG = 0）	293
高 / 低压检测工作原理（VDIRMAG = 1）	294
高 / 低压检测特性	380
故障保护时钟监视器（FSCM）	41
缓慢上升时间（MCLR 连接到 VDD, VDD 电压 上升时间 > TPWRT）	57
进入空闲模式的转换时序	47
进入休眠模式的转换	46
内部振荡器切换时序	39
启动条件期间的总线冲突（仅用于 SDA）	230
启动条件期间的总线冲突（SCL = 0）	231
启动条件期间由 SDA 仲裁引起的 BRG 复位	231
欠压复位（BOR）	386
全桥 PWM 输出	180
上电延时时序（MCLR 连接到 VDD, VDD 电压 上升时间 < TPWRT）	56
上电延时时序（MCLR 未连接到 VDD, 情形 1）	56
上电延时时序（MCLR 未连接到 VDD, 情形 2）	56
时钟 / 指令周期	69
时钟同步	215
停止条件期间的总线冲突（情形 1）	233
停止条件期间的总线冲突（情形 2）	233
同步发送	256
同步发送（由 TXEN 位控制）	256
同步接收（主模式, SREN）	259
外部时钟（除 PLL 外的所有模式）	382
休眠时的自动唤醒位（WUE）	253
异步发送	238
异步发送（背对背）	239
异步接收	243
应答序列	228
在 PLL 使能时 POR 的延时时序 （MCLR 连接到 VDD）	57
在占空比接近 100% 时改变 PWM 方向	182
正常操作时的自动唤醒位（WUE）	253
主 SSP I ² C 总线启动位 / 停止位	396
主 SSP I ² C 总线数据	396
自动波特率计算器	252
时序图和规范	382
A/D 转换要求	400
CLKO 和 I/O 要求	385
I ² C 总线启动位 / 停止位要求（从模式）	394
I ² C 总线数据要求（从模式）	395
PLL 时钟	384
SPI 模式要求示例	
（从模式, CKE = 0）	392
（从模式, CKE = 1）	393
（主模式, CKE = 0）	390
（主模式, CKE = 1）	391
Timer0 和 Timer1 外部时钟要求	387
USART 同步发送要求	398
USART 同步接收要求	398
并行从端口要求（PIC18F4X20）	389
捕捉 / 比较 / PWM 要求	388
复位、看门狗定时器、振荡器起振定时器、上电延时 定时器和欠压复位要求	386
外部时钟要求	383
主 SSP I ² C 总线启动位 / 停止位要求	396
主 SSP I ² C 总线数据要求	397
时钟切换	37
时钟源	
内部模式	32
HFINTOSC	32
INTOSC	32
INTOSCIO	32
LFINTOSC	34
频率选择	34
使用 OSCCON 寄存器选择	28
外部模式	30
EC	30
HS	31
LP	31
OST	30
RC	32
XT	31
相关的寄存器	41
选择 31 kHz 时钟源	28
数据存储器	71
PIC18F23K20/43K20 的映射	72
PIC18F24K20/44K20 的映射	73
PIC18F25K20/45K20 的映射	74, 75
存储区选择寄存器（BSR）	71
和扩展指令集	85
快速操作存储区	77
特殊功能寄存器	77
通用寄存器	77
数据 EEPROM	
代码保护	311
数据 EEPROM 存储器	99
EEADR 和 EEADRH 寄存器	99
EECON1 和 EECON2 寄存器	99
代码保护期间的操作	102
读	101
防止误写操作的保护措施	102
使用	102

PIC18F2XK20/4XK20

相关的寄存器	103
写	101
写校验	101
数据寻址模式	83
固有和立即数	83
间接	83
立即数变址寻址	85
受影响的指令	85
使能了扩展指令集的寻址模式对比	86
直接寻址	83
双速启动	297
双速时钟启动模式	38
双字指令	
示例情形	70
所有寄存器的初始化状态	59 - 62
T	
T0CON 寄存器	155
T1CON 寄存器	159
T2CON 寄存器	167
T3CON 寄存器	169
TBLRD	351
TBLWT	352
Timer0	155
16 位模式下的读写操作	156
工作原理	156
切换预分频器的分配	157
时钟源边沿选择 (T0SE 位)	156
时钟源选择 (T0CS 位)	156
相关的寄存器	157
溢出中断	157
预分频比选择 (T0PS2:T0PS0 位)	157
预分频器	157
预分频器分配 (PSA 位)	157
预分频器。请参见预分频器, Timer0。	
Timer1	159
16 位读 / 写模式	162
TMR1H 寄存器	159
TMR1L 寄存器	159
复位, 使用 CCP 特殊事件触发信号	163
工作原理	160
特殊事件触发信号 (ECCP)	174
相关的寄存器	165
异步计数器模式	161
读写	161
溢出中断	159
用作实时时钟	164
预分频器	161
振荡器	159, 162
振荡器布线注意事项	163
中断	163
Timer2	167
工作原理	167
输出	168
相关的寄存器	168
中断	168
Timer3	169
16 位读 / 写模式	171
TMR3H 寄存器	169
TMR3L 寄存器	169
工作原理	170
特殊事件触发信号 (CCP)	172
相关的寄存器	172
溢出中断	169, 171
振荡器	169, 171
TRISE 寄存器	134
PSPMODE 位	130
TSTFSZ	353
TXREG	237
TXSTA 寄存器	244
BRGH 位	247
特殊功能寄存器	77
映射	78
特殊事件触发器	267
特殊事件触发器。请参见比较 (ECCP 模块)。	
特殊事件触发器。请参见比较 (ECCP 模式)。	
V	
VREF。请参见 ADC 参考电压	
W	
WCOL	223, 224, 225, 228
WCOL 状态标志	223, 224, 225, 228
WDTCON 寄存器	307
WWW 地址	15
WWW, 在线支持	10
X	
XORLW	353
XORWF	354
斜率	138
休眠	
OSC1 和 OSC2 引脚的状态	36
休眠模式	45
Y	
异步操作的时钟精确性	244
引脚功能	
MCLR/VPP/RE3	16, 20
OSC1/CLK1/RA7	16, 20
OSC2/CLK0/RA6	16, 20
RA0/AN0/C12IN0-	17, 21
RA1/AN1/C12IN0-	21
RA1/AN1/C12IN1-	17
RA2/AN2/VREF-/CVREF/C2IN+	17, 21
RA3/AN3/VREF+/C1IN+	17, 21
RA4/T0CKI/C1OUT	17, 21
RA5/AN4/SS/HLDIN/C2OUT	17, 21
RB0/INT0/FLT0/AN12	18, 22
RB1/INT1/AN10/C12IN3-	22
RB1/INT1/AN10/P1C/C12IN3-	18
RB2/INT2/AN8	22
RB2/INT2/AN8/P1B	18
RB3/AN9/CCP2/C12IN2-	18, 22
RB4/KBI0/AN11	22
RB4/KBI0/AN11/P1D	18
RB5/KBI1/PGM	18, 22
RB6/KBI2/PGC	18, 22
RB7/KBI3/PGD	18, 22
RC0/T1OSO/T13CKI	19, 23
RC1/T1OSI/CCP2	19, 23
RC2/CCP1/P1A	19, 23
RC3/SCK/SCL	19, 23
RC4/SDI/SDA	19, 23
RC5/SDO	19, 23
RC6/TX/CK	19, 23
RC7/RX/DT	19, 23
RD0/PSP0	24
RD1/PSP1	24
RD2/PSP2	24
RD3/PSP3	24
RD4/PSP4	24

RD5/PSP5/P1B	24
RD6/PSP6/P1C	24
RD7/PSP7/P1D	24
RE0/RD/AN5	25
RE1/WR/AN6	25
RE2/CS/AN7	25
VDD	19, 25
Vss	19, 25
因特网地址	15
硬件乘法器	105
工作原理	105
简介	105
性能比较	105
预分频器, Timer0	157
Z	
在线串行编程 (ICSP)	297, 311
在线调试器	311
增强型捕捉 / 比较 / PWM (ECCP)	173
标准 PWM 模式	174
捕捉和比较模式	174
捕捉模式。请参见捕捉 (ECCP 模块)。	
定时器资源	174
输出和配置	174
相关的寄存器	191
增强型 PWM 模式	175
半桥模式	178
半桥应用	178
半桥应用示例	185
可编程死区延时	185
启动注意事项	182
全桥模式	179
全桥输出模式中的方向改变	181
全桥应用	179
输出关系图	177
输出关系 (高电平有效和低电平有效)	176
直通电流	185
自动关闭	183
自动重启	184
增强型通用同步 / 异步收发器 (EUSART)	235
振荡器模块	27
HFINTOSC	27
LFINTOSC	27
振荡器配置	
EC	27
ECIO	27
HS	27
HSPLL	27
INTOSC	27
INTOSCIO	27
LP	27
RC	27
RCIO	27
XT	27
振荡器起振定时器 (OST)	36, 55
振荡器切换	
故障保护时钟监视器	40
双速时钟启动	38
振荡器选择	297
振荡器, Timer1	159, 171
振荡器, Timer3	169
直接寻址	84
指令集	313
ADDLW	319
ADDWF	319
ADDWFC	320
ADDWF (立即数变址寻址模式)	361
ANDLW	320
ANDWF	321
BC	321
BCF	322
BN	322
BNC	323
BNN	323
BNOV	324
BNZ	324
BOV	327
BRA	325
BSF	325
BSF (立即数变址寻址模式)	361
BTFSC	326
BTFSS	326
BTG	327
BZ	328
CALL	328
CLRF	329
CLRWD	329
COMF	330
CPFSEQ	330
CPFGT	331
CPFSLT	331
DAW	332
DCFSNZ	333
DECFL	332
DECFSZ	333
GOTO	334
INCF	334
INCFSZ	335
INFSNZ	335
IORLW	336
IORWF	336
LFSR	337
MOVFL	337
MOVFF	338
MOVLB	338
MOV LW	339
MOVWF	339
MULLW	340
MULWF	340
NEGF	341
NOP	341
POP	342
PUSH	342
RCALL	343
RESET	343
RETFIE	344
RETLW	344
RETURN	345
RLCF	345
RLNCF	346
RRCF	346
RRNCF	347
SETF	347
SETF (立即数变址寻址模式)	361
SLEEP	348
SWAPF	350
SUBFWB	348
SUBLW	349
SUBWF	349
SUBWFB	350
TBLRD	351

PIC18F2XK20/4XK20

TBLWT	352
TSTFSZ	353
XORLW	353
XORWF	354
操作码字段说明	314
扩展指令集	355
通用格式	315
指令流 / 流水线	69
指令周期	69
时钟机制	69
直流和交流特性	
图表	401
直流特性	
RC 空闲供电电流	371
RC 运行供电电流	370
掉电电流	369
辅助振荡器供电电流	373
供电电压	369
输入 / 输出	375
外设供电电流	374
主空闲供电电流	372
主运行供电电流	372
直通电流	185
中断	107
中断现场保护	119
中断源	297
ADC	265
INTn 引脚	119
PORTB, 电平变化中断	119
TMR0	119
TMR0 溢出	157
TMR1 溢出	159
TMR3 溢出	169, 171
比较完成 (CCP)	147
捕捉完成 (CCP)	145
电平变化中断 (RB7:RB4)	124
主复位 (MCLR)	53
主同步串行口 (MSSP)。请参见 MSSP。	

MICROCHIP 网站

Microchip 网站 (www.microchip.com) 为客户提供在线支持。客户可通过该网站方便地获取文件和信息。只要使用常用的因特网浏览器即可访问。网站提供以下信息：

- **产品支持**——数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及存档软件
- **一般技术支持**——常见问题 (FAQ)、技术支持请求、在线讨论组以及 Microchip 顾问计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

变更通知客户服务

Microchip 的客户通知服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请登录 Microchip 网站 www.microchip.com，点击“变更通知客户 (Customer Change Notification)”服务后按照注册说明完成注册。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师 (FAE)
- 技术支持

客户应联系其代理商、代表或应用工程师 (FAE) 寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过<http://support.microchip.com>获得网上技术支持。

PIC18F2XK20/4XK20

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： PIC18F2XK20/4XK20

文献编号： DS41303F_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。

器件编号	X	IXX	XXX
器件	温度范围	封装	模式
器件: PIC18F23K20 ⁽¹⁾ 、PIC18F24K20 ⁽¹⁾ 、PIC18F25K20 ⁽¹⁾ 、 PIC18F26K20 ⁽¹⁾ 、PIC18F43K20 ⁽¹⁾ 、PIC18F44K20 ⁽¹⁾ 、 PIC18F45K20 ⁽¹⁾ 和 PIC18F46K20 ⁽¹⁾			
温度范围: E = -40°C 至 +125°C (扩展级) I = -40°C 至 +85°C (工业级)			
封装: ML = QFN MV = UQFN P = PDIP PT = TQFP (薄型正方扁平封装) SO = SOIC SP = 窄型塑封 DIP SS = SSOP			
模式: QTP、SQTP、代码或特殊要求 (其他情况空白)			
示例: a) PIC18F45K20-E/P 301 = 扩展级温度, PDIP 封装, 扩展 VDD 范围, QTP 模式 #301。 b) PIC18F23K20-I/SO = 工业级温度, SOIC 封装。 c) PIC18F44K20-E/P = 扩展级温度, PDIP 封装。			
注 1: T = 卷带式 PLCC, 仅限 TQFP 封装。			



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

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 - 重庆
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

中国 - 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 - 南京
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

中国 - 青岛
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 - 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 - 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 - 深圳
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

中国 - 武汉
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 - 西安
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

中国 - 厦门
Tel: 86-592-238-8138
Fax: 86-592-238-8130

中国 - 珠海
Tel: 86-756-321-0040
Fax: 86-756-321-0049

台湾地区 - 高雄
Tel: 886-7-536-4818
Fax: 886-7-536-4803

台湾地区 - 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

亚太地区

台湾地区 - 新竹

Tel: 886-3-6578-300
Fax: 886-3-6578-370

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4080

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

12/30/09