PICmicro®
中档单片机系列
参考手册

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 或任何其它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它受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

商标

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

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

目录

页码

第 1 章 简介 **1-1**

简介 1-2
本手册的宗旨 1-3
器件结构 1-4
开发支持 1-6
器件种类 1-7
格式和符号的约定 1-12
相关文档 1-14
相关应用笔记 1-17
版本历史 1-18

第 2 章 振荡器 **2-1**

简介 2-2
振荡器配置 2-2
晶体振荡器 / 陶瓷谐振器 2-4
外部 RC 振荡器 2-12
4MHz 内部 RC 振荡器 2-13
休眠模式对片内振荡器的影响 2-17
器件复位对片内振荡器的影响 2-17
设计技巧 2-18
相关应用笔记 2-19
版本历史 2-20

第 3 章 复位 **3-1**

简介 3-2
上电复位、上电延时定时器、起振定时器、欠压复位和奇偶校验错误复位 3-4
寄存器和状态位的值 3-10
设计技巧 3-16
相关应用笔记 3-17
版本历史 3-18

第 4 章 架构 **4-1**

简介 4-2
时序图 / 指令周期 4-5
指令流 / 流水线 4-6
I/O 端口描述 4-7
设计技巧 4-12
相关应用笔记 4-13
版本历史 4-14

目录

页码

第 5 章 CPU 和 ALU **5-1**

简介 5-2
指令的一般格式 5-4
中央处理单元（CPU） 5-4
指令时钟 5-4
算术逻辑单元（ALU） 5-5
状态寄存器 5-6
OPTION_REG 寄存器 5-8
电源控制寄存器 5-9
设计技巧 5-10
相关应用笔记 5-11
版本历史 5-12

第 6 章 存储器构成 **6-1**

简介 6-2
程序存储器构成 6-2
数据存储器构成 6-8
初始化 6-14
设计技巧 6-16
相关应用笔记 6-17
版本历史 6-18

第 7 章 数据 EEPROM **7-1**

简介 7-2
控制寄存器 7-3
EEADR 7-4
EECON1 和 EECON2 寄存器 7-4
从 EEPROM 数据存储器中读数据 7-5
向 EEPROM 数据存储器中写数据 7-5
写校验 7-6
误写操作保护 7-7
代码保护配置下的数据 EEPROM 操作 7-7
初始化 7-7
设计技巧 7-8
相关应用笔记 7-9
版本历史 7-10

第 8 章 中断 **8-1**

简介 8-2
控制寄存器 8-5
中断响应延时 8-10
INT 和外部中断 8-10
中断的现场保护 8-11
初始化 8-14
设计技巧 8-16
相关应用笔记 8-17
版本历史 8-18

目录

页码

第 9 章 I/O 端口 **9-1**

简介 9-2
PORTA 和 TRISA 寄存器 9-4
PORTB 和 TRISB 寄存器 9-6
PORTC 和 TRISC 寄存器 9-8
PORTD 和 TRISD 寄存器 9-9
PORTE 和 TRISE 寄存器 9-10
PORTF 和 TRISF 寄存器 9-11
PORTG 和 TRISG 寄存器 9-12
GPIO 和 TRISGP 寄存器 9-13
I/O 编程注意事项 9-14
初始化 9-16
设计技巧 9-17
相关应用笔记 9-19
版本历史 9-20

第 10 章 并行从动端口 **10-1**

简介 10-2
控制寄存器 10-3
操作 10-4
休眠模式下的操作 10-5
复位的影响 10-5
PSP 波形 10-5
设计技巧 10-6
相关应用笔记 10-7
版本历史 10-8

目录

页码

第 11 章 TIMER0 **11-1**

简介 11-2
控制寄存器 11-3
操作 11-4
TMR0 中断 11-5
Timer0 外部时钟的使用 11-6
TMR0 的预分频器 11-7
设计技巧 11-10
相关应用笔记 11-11
版本历史 11-12

第 12 章 TIMER1 **12-1**

简介 12-2
控制寄存器 12-3
Timer1 工作在定时器模式 12-4
Timer1 工作在同步计数器模式 12-4
Timer1 工作在异步计数器模式 12-5
Timer1 振荡器 12-7
休眠操作 12-9
用 CCP 触发器的输出将 Timer1 复位 12-9
Timer1 寄存器（TMR1H:TMR1L）的复位 12-9
Timer1 预分频器 12-9
初始化 12-10
设计技巧 12-12
相关应用笔记 12-13
版本历史 12-14

第 13 章 TIMER2 **13-1**

简介 13-2
控制寄存器 13-3
定时器时钟源 13-4
定时器 TMR2 和 PR2 周期寄存器 13-4
TMR2 匹配输出 13-4
将 Timer2 的预分频器和后分频器清零 13-4
休眠操作 13-4
初始化 13-5
设计技巧 13-6
相关应用笔记 13-7
版本历史 13-8

第 14 章 比较 / 捕捉 / 脉宽调制 (CCP) **14-1**

简介 14-2
控制寄存器 14-3
捕捉模式 14-4
比较模式 14-6
PWM 模式 14-8
初始化 14-12
设计技巧 14-15
相关应用笔记 14-17
版本历史 14-18

目录

页码

第 15 章 同步串行口 （SSP） **15-1**

简介 15-2
控制寄存器 15-3
SPI™ 模式 15-6
SSP 模块的 I²C™ 操作 15-16
初始化 15-26
设计技巧 15-28
相关应用笔记 15-29
版本历史 15-30

第 16 章 基本同步串行口 （BSSP） **16-1**

简介 16-2
控制寄存器 16-3
SPI™ 模式 16-6
SSP 模块 的 I²C™ 操作 16-15
初始化 16-23
设计技巧 16-24
相关应用笔记 16-25
版本历史 16-26

第 17 章 主同步串行口 （MSSP） **17-1**

简介 17-2
控制寄存器 17-4
SPI™ 模式 17-9
SSP 模块的 I²C™ 操作 17-18
I²C™ 总线的连接注意事项 17-56
初始化 17-57
设计技巧 17-58
相关应用笔记 17-59
版本历史 17-60

第 18 章 USART **18-1**

简介 18-2
控制寄存器 18-3
USART 波特率发生器 （BRG） 18-5
USART 异步工作模式 18-8
USART 同步主控模式 18-15
USART 同步从动模式 18-19
初始化 18-21
设计技巧 18-22
相关应用笔记 18-23
版本历史 18-24

目录

页码

第 19 章 参考电压模块 **19-1**

简介 19-2
控制寄存器 19-3
配置参考电压 19-4
参考电压精度 19-5
休眠模式下的操作 19-5
复位的影响 19-5
连接注意事项 19-6
初始化 19-7
设计技巧 19-8
相关应用笔记 19-9
版本历史 19-10

第 20 章 比较器 **20-1**

简介 20-2
控制寄存器 20-3
设置比较器模式 20-4
比较器工作原理 20-6
比较器参考源 20-6
比较器的响应时间 20-8
比较器输出 20-8
比较器中断 20-9
休眠状态下比较器的操作 20-9
复位的影响 20-9
模拟输入连接方式注意事项 20-10
初始化 20-11
设计技巧 20-12
相关应用笔记 20-13
版本历史 20-14

第 21 章 8 位 A/D 转换器 **21-1**

简介 21-2
控制寄存器 21-3
操作 21-5
A/D 采集时间要求 21-6
A/D 转换时钟的选择 21-8
配置模拟输入端口 21-9
A/D 转换 21-10
休眠期间的 A/D 转换 21-12
A/D 精度 / 误差 21-13
复位对 A/D 转换的影响 21-13
CCP 触发器的使用 21-14
连接注意事项 21-14
传递函数 21-14
初始化 21-15
设计技巧 21-16
相关应用笔记 21-17
版本历史 21-18

目录

页码

第 22 章 基本型 8 位 A/D 转换器 22-1

简介 22-2
控制寄存器 22-3
A/D 采集时间要求 22-6
A/D 转换时钟的选择 22-8
配置模拟输入端口 22-10
A/D 转换 22-11
休眠期间的 A/D 转换 22-14
A/D 转换精度 / 误差 22-15
复位对 A/D 转换的影响 22-16
连接时的考虑事项 22-16
传递函数 22-16
初始化 22-17
设计技巧 22-18
相关应用笔记 22-19
版本历史 22-20

第 23 章 10 位 A/D 转换器 23-1

简介 23-2
控制寄存器 23-3
操作 23-5
A/D 采集时间要求 23-6
A/D 转换时钟的选择 23-8
模拟输入引脚的设置 23-9
A/D 转换的编程举例 23-10
休眠期间的 A/D 转换 23-14
复位对 A/D 转换的影响 23-14
A/D 转换精度与误差 23-15
连接时的考虑事项 23-16
传递函数 23-16
初始化 23-17
设计技巧 23-18
相关应用笔记 23-19
版本历史 23-20

第 24 章 积分型 A/D 转换器 24-1

简介 24-2
控制寄存器 24-3
转换过程 24-6
其它模拟模块 24-12
校准参数 24-13
设计技巧 24-14
相关应用笔记 24-15
版本历史 24-16

目录

页码

第 25 章 LCD **25-1**

简介 25-2
控制寄存器 25-3
LCD 定时 25-6
LCD 中断 25-12
像素控制 25-13
电压发生器 25-15
休眠模式下的操作 25-16
复位的影响 25-17
LCD 模块的设置 25-17
判别比 25-18
LCD 电压发生器 25-20
对比度 25-22
LCD 玻璃基板 25-22
初始化 25-23
设计技巧 25-24
相关应用笔记 25-25
版本历史 25-26

第 26 章 看门狗定时器与休眠模式 **26-1**

简介 26-2
控制寄存器 26-3
看门狗定时器 （WDT）的操作 26-4
休眠省电模式 26-7
初始化 26-9
设计技巧 26-10
相关应用笔记 26-11
版本历史 26-12

第 27 章 器件配置位 **27-1**

简介 27-2
配置字位 27-4
编程校验 / 代码保护 27-8
识别码 ID 的位置 27-9
设计技巧 27-10
相关应用笔记 27-11
版本历史 27-12

第 28 章 在线串行编程 **28-1**

简介 28-2
进入在线串行编程模式 28-3
应用电路 28-4
编程器 28-6
编程环境 28-6
其它优点 28-7
PICmicro® OTP 型单片机的现场编程 28-8
FLASH 型 PICmicro® 单片机的现场编程 28-10
设计技巧 28-12
相关应用笔记 28-13
版本历史 28-14

目录

页码

第 29 章 指令集 **29-1**

简介 29-2
指令格式 29-4
作为源 / 目标寄存器的特殊功能寄存器 29-6
Q 周期操作 29-7
指令描述 29-8
设计技巧 29-45
相关应用笔记 29-47
版本历史 29-48

第 30 章 电气规范 **30-1**

简介 30-2
绝对最大值 30-3
器件选型表 30-4
器件电压规范 30-5
器件电流特性 30-6
输入阈值电平 30-9
I/O 电流特性 30-10
输出驱动电压 30-11
I/O 引脚的容性负载 30-12
数据 EEPROM / 闪存 30-13
LCD 30-14
比较器和参考电压 30-15
时序参数符号 30-16
外部时钟时序波形图和时序要求示例 30-17
上电和复位时序波形图及要求示例 30-19
定时器 Timer0 和 Timer1 时序波形图及要求示例 30-20
CCP 的时序图及要求 30-21
并行从动端口 (PSP) 时序图及要求 30-22
SSP 和 MSSP SPI™ 模式时序波形图及要求示例 30-23
SSP I²C™ 模式时序波形图及要求示例 30-27
MSSP I²C™ 模式时序波形图及要求示例 30-30
USART/SCI 时序波形图及要求示例 30-32
8 位 A/D 时序波形图及要求示例 30-34
10 位 A/D 时序波形图及要求示例 30-36
积分型 A/D 时序波形图及要求示例 30-38
LCD 时序波形图及要求示例 30-40
相关应用笔记 30-41
版本历史 30-42

第 31 章 器件特性 **31-1**

简介 31-2
特性和电气规范 31-2
DC 和 AC 特性图表 31-2
版本历史 31-22

目录

页码

第 32 章 开发工具 **32-1**

简介 32-2
集成开发环境（IDE） 32-3
MPLAB® 软件语言支持 32-6
MPLAB® SIM 软件模拟器 32-8
MPLAB® 硬件仿真器支持 32-9
MPLAB® 编程器支持 32-10
辅助工具 32-11
开发板 32-12
针对其它 Microchip 产品的开发工具 32-14
相关应用笔记 32-15
版本历史 32-16

第 33 章 代码开发 **33-1**

版本历史 33-2

第 34 章 附录 **34-1**

I^2C^{TM} 概述 34-2
LCD 玻璃基板生产商 34-11
改进的器件特性 34-13
版本历史 34-19

第 35 章 术语表 **35-1**

版本历史 35-14

第 1 章 简介

目录

本章包括下面一些主要内容：

1.1 简介 1-2
1.2 本手册的宗旨 1-3
1.3 器件结构 1-4
1.4 开发支持 1-6
1.5 器件种类 1-7
1.6 格式和符号的约定 1-12
1.7 相关文档 1-14
1.8 相关应用笔记 1-17
1.9 版本历史 1-18

1.1 简介

Microchip 公司是 The Embedded Control Solutions Company® （嵌入式控制系统解决方案公司），其产品主要满足嵌入式控制市场的需求。我们是以下产品的领先供应商：

- 8 位通用单片机 (PICmicro® 单片机)
- 专用和标准的非易失性存储器件
- 安防器件 (KEELOQ®)
- 专用标准产品

欲获得您所感兴趣的产品列表，请申请一份 Microchip 产品线目录。该文献可从各地的 Microchip 销售办事处获得，或者直接从 Microchip 的网站 (www.microchip.com) 上下载。

以往，8 位单片机的用户只选择传统的 MCU 类型，即 ROM 器件，用于生产。Microchip 率先改变了这种传统观念，向人们展示了 OTP(一次性编程) 器件比 ROM 器件在其寿命周期内具有更低的产品成本。

Microchip 具备 EPROM 技术优势，从而使 EPROM 成为 PICmicro 单片机程序存储器的不二选择。Microchip 尽可能地缩小了 EPROM 和 ROM 存储器技术之间的成本差距，并使顾客从中受益。其他 MCU 供应商无法作到这一点，这从他们的 EPROM 和 ROM 版本之间的价格差异便可以看出。

Microchip 的 8 位单片机市场份额的增长证明了 PICmicro 单片机能够满足大多数人的需要。这也使 PICmicro 单片机架构成为了当今通用市场上应用最广泛的三大体系之一。 Microchip 的低成本 OTP 解决方案所带来的效益是这一增长的助推剂。用户能够从以下各方面受益：

- 快速的产品上市时间
- 允许生产过程中对产品进行代码修改
- 无需掩膜产品所需的一次性工程费用 (NRE)
- 能够轻松为产品进行连续编号
- 无需额外增加硬件即可存储校准数据
- 可最大限度地增加 PICmicro 单片机的库存
- 由于在开发和生产中使用同一器件，从而降低了风险

Microchip 的 8 位 PICmicro 单片机具备很好的性价比，可成为任何传统的 8 位应用和某些 4 位应用 (低档系列)、专用逻辑的替代品以及低端 DSP 应用 (高档系列) 的选择。这些特点及其良好的性价比使 PICmicro 单片机在大多数应用场合极具吸引力。

1.2 本手册的宗旨

PICmicro 单片机根据其指令长度来划分，目前的三个 PICmicro 单片机系列是：

1. 低档： 12 位指令字长度
2. 中档： 14 位指令字长度
3. 高档： 16 位指令字长度

本手册重点介绍中档系列器件，即 PIC16CXXX 单片机系列。

本手册介绍了 PIC16CXXX 系列单片机的架构和外设模块的操作，但并不涉及每个器件的具体细节。因此，本手册并不取代器件数据手册，而是对它作了补充。也就是说，本手册提供了 PICmicro 系列单片机的架构和外设模块的一般特点和操作，而数据手册则给出了具体细节，如存储器映射等。

本手册给出了初始化例子。这些例子有时是针对特定器件，而有别于整个系列的一般属性，尽管对于大多数其他器件来说，它们都是可行的。对寄存器文件映射有所不同的器件，可能需要作一些修改。

注：	少数早期的中档系列产品与本手册中的简介有细微的不同。本手册尽量对这些不同进行了描述。如果需要某个器件的详细信息，请参阅该器件的数据手册。

1.3 器件结构

可将器件划分为以下三个部分：

1. 内核
2. 外设
3. 特殊功能部件

1.3.1 内核

内核是使器件运行的基本部件。包括：

1. 振荡器	版本 "DS31002A”
2. 复位逻辑	版本 "DS31003A”
3. CPU (中央处理单元) 的操作	版本 "DS31005A”
4. ALU (算术逻辑单元) 的操作	版本 "DS31005A”
5. 器件的存储器构成	版本 "DS31006A”
6. 中断操作	版本 "DS31008A”
7. 指令集	版本 "DS31029A”

1.3.2 外设

外设是在单片机上添加的一些特殊功能。这些功能方便了单片机与外部世界进行联系 (例如通用 I/O、 LCD 驱动器、 A/D 输入和 PWM 输出)，并可执行内部任务，如保存不同的时基 （如定时器）。本手册对以下外设进行了介绍：

1. I/O 口	版本 "DS31009A”
2. 定时器 Timer0	版本 "DS31011A”
3. 定时器 Timer1	版本 "DS31012A”
4. 定时器 Timer2	版本 "DS31013A”
5. 捕捉、比较和脉宽调制 (CCP)	版本 "DS31014A”
6. 同步串行口 (SSP)	版本 "DS31015A”
7. 基本同步串行口 (BSSP)	版本 "DS31016A”
8. 主同步串行口 (MSSP)	版本 "DS31017A”
9. 通用同步异步收发器 USART (SCI)	版本 "DS31018A”
10. 参考电压模块	版本 "DS31019A”
11. 比较器	版本 "DS31020A”
12. 8 位 A/D 转换器	版本 "DS31021A”
13. 基本型 8 位 A/D 转换器	版本 "DS31022A”
14. 10 位 A/D 转换器	版本 "DS31023A”
15. 带热敏电阻的积分型 A/D 转换器	版本 "DS31024A”
16. LCD 驱动器	版本 "DS31025A”
17. 并行从动端口 (PSP)	版本 "DS31010A”

1.3.3 特殊功能部件

特殊功能部件是有助于达到以下一个或多个目的的独特的部件：

- 降低系统成本
- 提高系统可靠性
- 增加设计灵活性

中档系列 PICmicro 单片机提供了一些能达到这些目的的特殊部件。本手册对以下特殊功能部件进行了介绍：

1. 器件配置位 版本 "DS31027A”
2. 片内上电复位 (POR) 版本 "DS31003A”
3. 欠压复位 (BOR) 逻辑 版本 "DS31003A”
4. 看门狗定时器 版本 "DS31026A”
5. 低功耗模式 (休眠) 版本 "DS31026A”
6. 内部 RC 振荡器 版本 "DS31002A”
7. 在线串行编程 （In-Circuit Serial Programming™， ICSP™）版本 "DS31028A”

1.4 开发支持

Microchip 提供了大量的开发工具，使用户可以高效地开发和调试应用代码。Microchip 的开发工具可以分为四类：

1. 代码生成
2. 软件调试
3. 器件编程器
4. 产品评估板

所有 Microchip 开发工具都在 MPLAB® 集成开发环境下运行，但某些第三方工具则不一定。代码生成工具包括：

- MPASM™
- MPLAB-C
- MP-DriveWay™

这些软件开发程序包括器件的头文件。每个头文件都将寄存器名称（如器件数据手册所示）定义到具体地址或位的位置。使用头文件便于代码移植，并减少了记忆寄存器地址或寄存器中某位的位置的烦琐程度。

注：	Microchip 强烈建议用户在程序的源代码中使用所提供的头文件。这样做便于代码移植，并有助于 Microchip 提供更加优质而深入的技术支持。

便于软件调试的工具如下：

- PICMASTER® 在线仿真器
- ICEPIC™ 在线仿真器
- MPLAB-SIM 软件模拟器

产生并调试了应用软件后，需要对器件进行编程。Microchip 提供以下两种编程器：

1. PICSTART® Plus 编程器
2. PROMATE® II 编程器

演示板可供软件代码的开发者评估单片机在应用中的性能和适用性。提供的演示板有：

- PICDEM™-1
- PICDEM™-2
- PICDEM™-3
- PICDEM™-14A

“开发工具”一章给出了所有 Microchip 开发工具的完整描述。当出现新开发工具时，其产品简介和用户指南可通过 Microchip 的网页 (www.microchip.com) 或从当地的 Microchip 销售办事处获得。

代码开发的建议和技巧将在 **“代码开发”一章**中介绍。

Microchip 还提供其它辅助工具来加速开发，它们是：

- 应用笔记
- 参考设计
- Microchip 网站
- Microchip 论坛
- 当地销售办事处提供的现场应用支持
- 公司的技术支持热线

网站上的用户兴趣小组如 MIT reflector PICIist 等还提供了额外的帮助。 Microchip 的网站列出了其它一些有用的链接。

1.5 器件种类

一旦器件的功能确定后，还需要考虑其它一些特性，包括：

- 存储器类型
- 工作电压
- 工作温度范围
- 工作频率
- 封装

Microchip 提供了大量的选择及选择组合，其中定有一种可以满足您的需要。

1.5.1 存储器种类

存储器类型对器件的逻辑操作没有影响。由于所需的工艺步骤不同，具有同样功能集 / 引脚排列而存储器类型不同的器件，其电气特性会有所不同。例如电气特性中的 VIL(输入低电压)，对于典型的 EPROM 器件和典型的 ROM 器件会有所不同。

每种单片机都有多种频率和封装选择。根据应用和生产要求，可以通过器件数据手册最末的产品选型章节来选择适当的器件。当订购器件时，请使用 “产品识别体系”订购正确的器件号。

器件的功能与存储器类型和工作电压范围无关。 Microchip 提供三种程序存储器类型。器件编号中产品系列指定符后的第一个 （或多个）字母指明了存储器类型。

1. PIC16**C**XXX 中的 **C** 表示这些器件具备 EPROM 型存储器。
2. PIC16**CR**XXX 中的 **CR** 表示这些器件具备 ROM 型存储器。
3. PIC16**F**XXX 中的 **F** 表示这些器件具备 FLASH 型存储器。

1.5.1.1 EPROM

Microchip 着重于可擦除可编程只读存储器 (EPROM) 技术的生产，为用户的整个设计开发过程提供了灵活性。 Microchip 在此技术的基础上提供了多种封装形式。

1.5.1.2 只读存储器 (ROM)

Microchip 提供几种大容量掩膜只读存储器 (ROM) 器件，从而为用户提供大容量、成熟产品的低成本方案。

ROM 器件的程序存储空间中不能写入序列信息。

欲了解如何提交 ROM 代码的信息，请联系当地 Microchip 销售办事处。

1.5.1.3 闪存存储器

这种器件是电可擦除的，可提供低成本的塑料封装。电可擦除特性使这些器件无需从电路板上拆下，即可擦除和再编程。无论是样机开发、试用，还是产品生产，均使用同样规格的器件。

1.5.2 工作电压范围

所有中档系列的 PICmicro 单片机均可在标准电压范围下运行。我们还提供扩展电压范围（频率范围缩小）的器件。表 1-1 列出了 PIC16CXXX 系列单片机的所有存储器类型和电压范围指定符。指定符用**粗**体显示。

表 1-1: 器件存储器类型和电压范围指定符

存储器类型	电压范围	
	标准	扩展
EPROM	PIC16**C**XXX	PIC16**LC**XXX
ROM	PIC16**CR**XXX	PIC16**LCR**XXX
Flash	PIC16**F**XXX	PIC16**LF**XXX

注：对某一器件不一定提供所有存储器类型。

如表 1-2 所示，在未对器件的特性指标做出标定前， Microchip 的扩展电压范围规范是较为保守的。

表 1-2: 每个器件类型对应的典型电压范围

典型电压范围 (1)		EPROM		ROM		Flash	
标准		**C**	4.5 - 6.0V	**CR**	4.5 - 6.0V	**F**	4.5 - 6.0V
扩展	器件特性指标标定前	**LC**	3.0 - 6.0V	**LCR**	3.0 - 6.0V	**LF**	3.0 - 6.0V
	最后规范	**LC**	2.5 - 6.0V	**LCR**	2.5 - 6.0V	**LF**	2.0 - 6.0V

注 1： Microchip 的 120K 工艺生产的器件，其 VDD 的上限为 5.5V。新的数据手册将明确标明这一点。
2： 工作电压范围由器件特性决定。

1.5.3 封装类型

在产品开发的不同阶段，有三种封装类型可供使用：

第一种带有一个可擦除窗口，一般是陶瓷体封装。该类器件的程序存储器可以被多次擦除和编程，因此一般用于开发阶段。

第二种是低成本塑料封装，这种封装类型的器件一般用于批量生产，以最大限度降低成本。

最后一种是 DIE （管芯），它是一种经过测试的无封装器件。 DIE 通常用于低成本设计以及将电路板空间保持最小的设计中。上述内容可小结如表 1-3 所示。

表 1-3: 典型封装的应用

封装类型	典型应用
窗口型	开发模式
塑封	生产
DIE	特殊应用，如需要最小电路板空间的应用

1.5.3.4 紫外线 (UV) 可擦除器件

UV 可擦除 EPROM 程序存储器是开发样机和试用器件的最佳选择。

这种器件可以被擦除和再编程为任何配置模式。还提供第三方编程器对该类器件进行编程，请参考 Microchip 的 *Third Party Guide* (DS00104)。

彻底擦除该类器件所需的时间与光波长、强度、到紫外线源的距离和器件的制造工艺技术 (存储单元的大小）有关。

注 :	由于荧光灯和日光所发出的光波长均可擦除器件，因此将窗口型器件的窗口裸露一段时间后，器件存储单元的内容会被擦除。荧光灯的擦除时间大约是三年，而日光则只需一周左右。为了防止存储单元的内容丢失，可在擦除窗口上帖一张不透明的标签。

1.5.3.5 一次性可编程 (OTP) 器件

一次性可编程器件对于需要对代码进行修改和更新的用户特别有用。

用户可对塑料封装的 OTP 器件进行一次编程。除了程序和数据 EPROM 存储器外，配置字也必须被编程。

1.5.3.6 闪存器件

闪存器件的存储器可以进行电改写。这意味着系统可以被设计成允许在线编程。因为不需要擦除窗口，器件可以使用低成本的塑料封装。

1.5.3.7 EEPROM 器件

EEPROM 型器件的存储器可以进行电改写。这意味着系统可以被设计成允许在线擦除和再编程。因为不需要擦除窗口，器件可以使用低成本的塑料封装。

1.5.3.8 ROM 器件

ROM 器件的程序存储器在硅加工时就固化了。由于程序存储器的内容不能被改写，这类器件可使用低成本的塑料封装。

1.5.3.9 DIE(管芯)

DIE 可使电路板尽可能缩小。DIE 技术文档 (DS30258) 介绍了 DIE 的使用与设计。我们还提供关于 DIE 的详细规格表。在制造中采用 DIE 需要专门的知识和设备，这说明支持 DIE 的生产商数量有限。如果您决定使用 DIE，请确认您的生产商能够满足使用 DIE 的专业要求。

1.5.3.10 专门服务

对于已有固定代码的 OTP 用户， Microchip 提供两种专门服务：快速批量编程 （Quick Turn Production Programming）和带序列号的快速批量编程 （ Serialized Quick Turn Production Programming），以缩短用户的制造周期。

1.5.3.11 快速批量编程 (QTP)

Microchip 批量生产订单提供这种出厂前的编程服务。这种服务适用于那些不想对中到大批量单片机编程，并且代码已经相对稳定的用户。这种器件与 OTP 器件相同，只是所有 EPROM 的位置和配置已在出厂前设定，并对代码进行了必要的校验。欲了解更详细信息，请联系当地 Microchip 销售办事处。

1.5.3.12 带序列号的快速批量编程 (SQTP^{SM})

Microchip 向用户提供这种独特的编程服务，可将每个器件中的几个用户指定位置编程为各自不同的序列号。该序列号可以是随机数、伪随机数或连续编号。

这种串行编程使每个器件具有唯一的序列号，可以作为登录码、口令或用户识别码。

1.6 格式和符号的约定

本文档采用了特定的字体格式。大多数字体变化表示其与正文的区别。单片机行业中有许多符号和非常规字词定义和缩写。表 1-4 给出了许多本文档中所包含的约定。**“术语表”一章**中提供了一个术语表，其中包含了更多在本手册中出现的字词和缩写的定义。

1.6.1 文档约定

表 1-4 给出了本手册中使用的一些符号和术语。

表 1-4: 文档约定

符号或术语	说明
置 1	强制某一位 / 寄存器的值为逻辑 1。
清零	强制某一位 / 寄存器的值为逻辑 0。
复位	1) 强制某一寄存器 / 位回到默认状态。 2) 复位后器件的状态。某些位将被强制为 0 (如中断允许位)，而其它位被置为 1 (如 I/O 数据方向位)。
0xnn 或 nnh	指定数据 ‘nn’ 为十六进制数。这种约定用于代码实例中。
B’bbbbbbbb’	指定数据 ‘bbbbbbbb’ 为二进制数。这种约定用于文本以及图表中。
R-M-W	读 - 修改 - 写。这表示寄存器或端口值被读取，修改后再写回寄存器或端口。单条指令 (如置位 BSF) 或一个指令序列可执行读 - 修改 - 写操作。
: (冒号)	用来指定范围，或寄存器 / 位 / 引脚的组合。 如 TMR1H:TMR1L 表示用两个 8 位寄存器组成一个 16 位定时器，而 SSPM3:SSPM0 是用来指定 SSP 模块工作模式的 4 位数据。组合顺序 (从左到右) 通常表示一种位置关系 (MSb 到 LSb，高位到低位)。
< >	在特定寄存器中指定位的位置。 如 SSPCON<SSPM3:SSPM0> (或 SSPCON<3:0>) 指定了寄存器和相关的位或位的位置。
`Courier` 字体	用于代码示例、二进制数以及文本中的指令助记符。
Times 字体	用于公式和变量。
Times, 黑体, 斜体	用于图表 / 公式 / 示例中的说明文本。
注	“注”表示需要强调的信息，可以帮助您避免常见的陷阱，或提醒您注意同一系列器件间的操作区别。“注”总是以阴影的方框出现 (如下)，除非用于表格中，这时它位于表格的下方 (如本表格)。 **注：** 这是一个 “注”方框中的 “注”。
小心 (1)	小心描述了一种可能潜在破坏软件或设备的情况。
警告 (1)	警告描述了一种可能潜在导致人身伤害的情况。

注 1： 我们提供小心或警告信息是为了保护您的人身安全。请仔细阅读每一条小心和警告信息。

1.6.2 电气特性

本手册中有一些对电气规范参数编号的引用。尽管不同器件的实际参数值可能不同，但参数编号代表了同一组特性或条件，不同数据手册的参数编号是一致的。

“电气规范”一章中列出了文档中涉及的所有器件的所有参数。一种器件不具备所有这些参数。这一章旨在向您说明 Microchip 对哪些参数作了规定。每个参数的实际值均取决于具体器件，尽管我们对保持所有器件参数的一致性作出了很大的努力。

表 1-5: 电气规范参数命名约定

参数格式	说明
Dxxx	直流规范
Axxx	模拟外设的直流规范
xxx	时序 (交流) 规范
PDxxx	器件编程直流规范
Pxxx	器件编程时序 (交流) 规范

注 1: xxx：代表一个数。

1.7 相关文档

Microchip 及其它合作伙伴提供了其它文档来帮助用户使用 PICmicro 单片机进行开发。下面列出了最常用的文档，当然还有其它文档可供参考。请浏览 Microchip 网站 (www.microchip.com) 查阅最新发布的技术文档。

1.7.1 Microchip 文档

Microchip 提供下列文档。其中许多文档都提供了具体的应用信息，并给出了 PICmicro 单片机的使用、编程和设计实例。

1. **MPASM™ User's Guide (DS33014)**
   该文档介绍了如何使用 Microchip 的 MPASM 汇编器。
2. **MPLAB®-C Compiler User's Guide (DS51014)**
   该文档介绍了如何使用 Microchip 的 MPLAB-C C 编译器。
3. **MPLAB® User's Guide (DS51025)**
   该文档 介绍了如何使用 Microchip 的 MPLAB 集成开发环境。
4. **MPLAB® Editor User's Guide (DS30420)**
   该文档介绍了如何使用 MPLAB 的内置编辑器。
5. **PICMASTER® User's Guide (DS30421)**
   该文档介绍了如何使用 Microchip 的 PICMASTER 在线仿真器。
6. **MPSIM™ User's Guide (DS30027)**
   该文档介绍了如何使用 Microchip 的 MPLAB 模拟器。
7. **PRO MATE® User's Guide (DS30082)**
   该文档介绍了如何使用 Microchip 的 PRO MATE 通用编程器。
8. **PICSTART®-Plus User's Guide (DS51028)**
   该文档介绍了如何使用 Microchip 的 PICSTART-Plus 低成本通用编程器。
9. ***fuzzy*TECH®-MP User's Guide (DS30389)**
   该文档介绍了如何使用 *fuzzy*TECH-MP 模糊逻辑代码生成器。
10. **MP-DriveWay™ User's Guide (DS51027)**
    该文档介绍了如何使用 MP-DriveWay 代码生成器。
11. ***fuzzy*TECH-MP Fuzzy Logic Handbook (DS30238)**
    该文档介绍了 *fuzzy*TECH-MP 模糊技术的基本概念。
12. **Embedded Control Handbook Volume I (DS00092)**
    该文档包含了大量应用笔记。文档中的各种程序代码对于深入了解器件的使用 (或其中的一部分)，以及着手进行一项应用设计是非常有用的。
13. **Embedded Control Handbook Volume II (DS00167)**
    该文档介绍了 PICmicro 单片机的数学库。
14. **In-Circuit Serial Programming™ Guide (DS30277)**
    该文档讨论了如何实现在线串行编程。
15. **PICDEM™-1 User's Guide (DS351079)**
    该文档介绍了如何使用 Microchip 的 PICDEM-1 演示板。
16. **PICDEM™-2 User's Guide (DS30374)**
    该文档介绍了如何使用 Microchip 的 PICDEM-2 演示板。
17. **PICDEM™-3 User's Guide (DS33015)**
    该文档介绍了如何使用 Microchip 的 PICDEM-3 演示板。
18. **Third Party Guide (DS00104)**
    该文档列出了 Microchip 的第三方合作伙伴及顾问。
19. **DIE Support (DS30258)**
    该文档给出了如何使用 Microchip 的 DIE （管芯）产品的相关信息。

1.7.2 第三方文档

我们的全球第三方合作伙伴提供了一些文档。 Microchip 并没有验证这些文档的技术准确性，然而，这些文档有助于理解 Microchip 单片机的操作。下面列出了一些在本手册付印时我们所知的文档，可能并不完整。欲了解如何与这些合作伙伴取得联系，以及我们新增的文档，请访问 Microchip 的网站。

文档	语言
The PIC16C5X Microcontroller: A Practical Approach to Embedded Control Bill Rigby/ Terry Dalby, Tecksystems Inc. 0-9654740-0-3	英语
Easy PIC'n David Benson, Square 1 Electronics 0-9654162-0-8	英语
A Beginners Guide to the Microchip PIC® Nigel Gardner, Bluebird Electronics 1-899013-01-6	英语
PIC® Microcontroller Operation and Applications DN de Beer, Cape Technikon	英语
Digital Systems and Programmable Interface Controllers WP Verburg, Pretoria Technikon	英语
Mikroprozessor PIC16C5X Michael Rose, Hüthig 3-7785-2169-1	德语
Mikroprozessor PIC17C42 Michael Rose, Hüthig 3-7785-2170-5	德语
Les Microcontrolleurs PIC® et mise en oeuvre Christian Tavernier, Dunod 2-10-002647-X	法语
Micontrolleurs PIC® a structure RISC C.F. Urbain, Publitronic 2-86661-058-X	法语
New Possibilities with the Microchip PIC® RIGA	俄语

文档	语言
PIC16C5X/71/84 Development and Design, Part 1 United Tech Electronic Co. Ltd 957-21-0807-7	中文
PIC16C5X/71/84 Development and Design, Part 2 United Tech Electronic Co. Ltd 957-21-1152-3	中文
PIC16C5X/71/84 Development and Design, Part 3 United Tech Electronic Co. Ltd 957-21-1187-6	中文
PIC16C5X/71/84 Development and Design, Part 4 United Tech Electronic Co. Ltd 957-21-1251-1	中文
PIC16C5X/71/84 Development and Design, Part 5 United Tech Electronic Co. Ltd 957-21-1257-0	中文
PIC16C84 MCU Architecture and Software Development ICC Company 957-8716-79-6	中文

1.8 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但是其概念是相近的，通过适当修改并受到一定限制即可使用。目前介绍 Microchip PICmicro 单片机的应用笔记有：

标题	应用笔记 #
A Comparison of Low End 8-bit Microcontrollers	AN520
PIC16C54A EMI Results	AN577
Continuous Improvement	AN503
Improving the Susceptibility of an Application to ESD	AN595
Plastic Packaging and the Effects of Surface Mount Soldering Techniques	AN598

1.9 版本历史

版本 A

这是 Microchip PICmicro 单片机简介的初始发行版。

第 2 章 振荡器

目录

本章包括下面一些主要内容：

2.1 简介 2-2
2.2 振荡器配置 2-2
2.3 晶体振荡器 / 陶瓷谐振器 2-4
2.4 外部 RC 振荡器 2-12
2.5 4MHz 内部 RC 振荡器 2-13
2.6 休眠模式对片内振荡器的影响 2-17
2.7 器件复位对片内振荡器的影响 2-17
2.8 设计技巧 2-18
2.9 相关应用笔记 2-19
2.10 版本历史 2-20

2.1 简介

内部振荡器电路用于产生器件时钟。执行指令和实现外设功能都要用到时钟信号。每四个时钟周期为一个内部指令周期 （TCY）。

振荡器共有 8 种工作模式。其中两种模式允许将内部 RC 振荡器的时钟输出（CLKOUT）在一个 I/O 引脚上输出，或允许将此 I/O 引脚用于一般输入输出功能。振荡器模式通过器件配置位来选择。器件配置位均为非易失性存储器，振荡器的工作模式由编程器件时所写进器件配置位的值决定。振荡模式如下：

- LP 低频 （低功耗）晶体
- XT 晶体 / 谐振器
- HS 高速晶体 / 谐振器
- RC 外部电阻 / 电容振荡模式 （与带 CLKOUT 的 EXTRC 模式相同）
- EXTRC 外部电阻 / 电容振荡模式
- EXTRC 外部电阻 / 电容振荡模式，带 CLKOUT 功能
- INTRC 4MHz 的内部电阻 / 电容振荡模式
- INTRC 4MHz 的内部电阻 / 电容振荡模式，带 CLKOUT 功能

这些振荡模式可满足根据不同应用选择不同振荡器模式的要求。选择 RC 振荡器模式可节省系统的成本，而 LP 晶体模式可降低系统功耗。配置位用来选择不同的选项。更多有关器件配置位的详细内容，请参阅 **“器件特性”一章**。

2.2 振荡器配置

2.2.1 振荡器类型

中档器件可以有 8 种不同的振荡模式。用户可通过编程 3 个器件配置位 （FOSC2， FOSC1 和 FOSC0）选择其中的一种：

- LP 低频 （低功耗）晶体
- XT 晶体 / 谐振器
- HS 高速晶体 / 谐振器
- RC 外部电阻 / 电容振荡模式 （与带 CLKOUT 的 EXTRC 模式相同）
- EXTRC 外部电阻 / 电容振荡模式
- EXTRC 外部电阻 / 电容振荡模式，带 CLKOUT 功能
- INTRC 4MHz 的内部电阻 / 电容振荡模式
- INTRC 4MHz 的内部电阻 / 电容振荡模式，带 CLKOUT 功能

LP、 XT 和 HS 模式的主要区别是振荡器的片内反相器增益不同，这使得振荡器有不同的工作频率范围。表 2-1 和 表 2-2 给出了选择振荡器模式的一些参考信息。通常情况下，选择能够满足技术要求且具有最低增益的模式，这样能够降低动态电流（IDD）。所给出的不同振荡器模式的频率范围值均为我们建议的并经过测试的振荡器截止频率。但若经过彻底验证（电压、温度、元件差异 （包括电阻、电容和单片机的内部振荡器电路）），也可选择不同的增益模式。

RC 模式和带 CLKOUT 的外部电阻 / 电容模式 （EXTRC）具有相同的功能。这样命名是为了便于描述它有别于其它振荡模式的操作。

表 2-1: 通过 FOSC1:FOSC0 位选择器件振荡器模式

配置位 FOSC1:FOSC0	OSC 模式	OSC 反馈 反相放大器 增益	备注
1 1	RC	—	最经济的振荡解决方案 （只需一个外部电阻和电容）。时基变化最大。 器件默认模式。
1 0	HS	高增益	高频应用。 在三种晶体振荡模式中，电流消耗最大。
0 1	XT	中等增益	标准晶体 / 谐振器。 在三种晶体振荡模式中，电流消耗较大。
0 0	LP	低增益	低功耗 / 低频应用 在三种晶体振荡模式中，电流消耗最小。

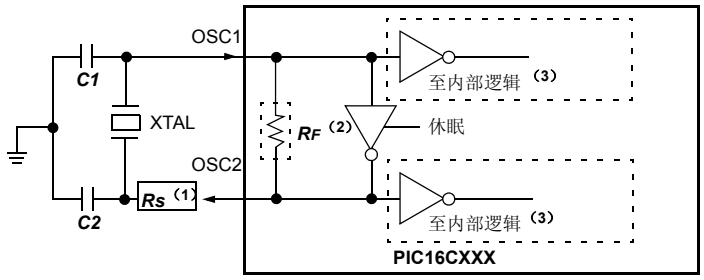
表 2-2: 通过 FOSC2:FOSC0 位选择器件振荡器模式

配置位 FOSC2:FOSC0	OSC 模式	OSC 反馈 反相放大器 增益	备注
1 1 1	带 CLKOUT 的 EXTRC	—	经济的器件振荡解决方案。时基变化最大。引脚的 CLKOUT 被使能。器件默认模式。
1 1 0	EXTRC	—	经济的器件振荡解决方案。时基变化最大。 引脚的 CLKOUT 被禁止 （作为 I/O）。
1 0 1	带 CLKOUT 的 INTRC	—	最经济的器件振荡解决方案。 4MHz 振荡器，振荡频率可微调。 引脚的 CLKOUT 被使能。
1 0 0	INTRC	—	最经济的器件振荡解决方案。 4MHz 振荡器，振荡频率可微调。 引脚的 CLKOUT 被禁止 （作为 I/O）。
0 1 1	—	—	保留
0 1 0	HS	高增益	高频应用。 在三种晶体振荡模式中，电流消耗最大。
0 0 1	XT	中等增益	标准晶体 / 谐振器。 在三种晶体振荡模式中，电流消耗较大。
0 0 0	LP	低增益	低功耗 / 低频应用。 在三种晶体振荡模式中，电流消耗最小。

2.3 晶体振荡器 / 陶瓷谐振器

在XT、LP或HS模式下，OSC1和OSC2引脚连接一个晶体或陶瓷谐振器以产生振荡，（图 2-1）。PICmicro® 单片机振荡器的设计要求使用一个平行切割的晶体。而使用顺序切割的晶体，可能使振荡器产生的频率超出晶体制造厂商所给的参数范围。在 XT、 LP 或 HS 模式下，器件可用一个外部时钟源来驱动 OSC1 引脚 （图 2-3）。

图 2-1: 晶体或陶瓷谐振器的运行 （HS， XT 或 LP 振荡器模式）



注 1: 对于 AT 条形切割 （AT Strip Cut）的晶体可能需要一个串联电阻 ***Rs***。

2: 反馈电阻，***RF***，典型值为 2-10MΩ。

3: 根据器件不同，内部逻辑的缓冲器可置于反相器的输入端或输出端。

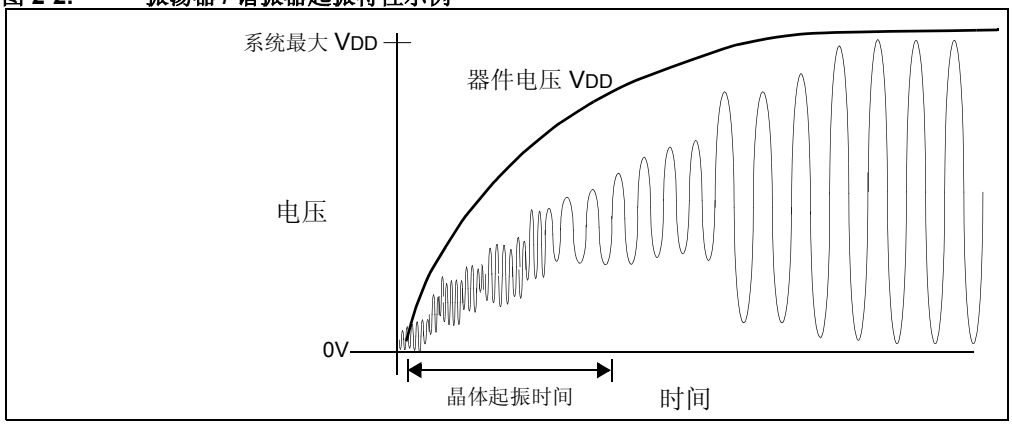
2.3.1 振荡器 / 谐振器起振

随着器件电压从 VSS 电平开始增加，振荡器将起振。起振时间取决于多种因素，包括：

- 振荡器 / 谐振器频率
- 所使用的电容容量 （图 2-1 中的 ***C1*** 和 ***C2***）
- 器件电源电压 V_{DD} 的上升时间
- 系统温度
- 如果使用了串联电阻 （图 2-1 中的 ***Rs***），还包括其阻值和类型
- 器件的振荡模式选择 （决定振荡器内部反相器的增益）
- 晶体的品质
- 振荡器电路布局
- 系统噪声

图 2-2 为一个振荡器 / 谐振器起振的示例。振荡器波形中的峰峰值电压可以很小（小于器件 VDD 的 50%），其波形以 VDD/2 为中心波动。（请参阅 **“电气规范”一章**中的参数 D033 和 D043）。

图 2-2: 振荡器 / 谐振器起振特性示例



2.3.2 元件选择

图 2-1 是晶体或陶瓷谐振器的电路图。反馈电阻 ***RF*** 的典型值在 2 到 10MΩ，其阻值随着器件电压、温度和制造工艺的变化而变化。如果使用 AT 条形切割晶体，可能需要串联电阻 ***Rs***。在确定电阻的需求时，一定要将器件的工作电压和制造工艺考虑进去。如图 2-1 所示，与器件内部逻辑的连接取决于器件本身。关于器件参数，请参阅相关的器件数据手册。表 2-3 和表 2-4 给出了电容的典型值 （***C1***， ***C2***），各器件的数据手册中均给出了经 Microchip 测试的具体数值。

表 2-3: 陶瓷谐振器的典型电容选择

测试范围		
模式	**频率**	***C1*** / ***C2***(1)
XT	455 kHz 2.0 MHz 4.0 MHz	22 - 100 pF 15 - 68 pF 15 - 68 pF
HS	8.0 MHz 16.0 MHz 20.0 MHz	10 - 68 pF 10 - 22 pF TBD
所使用的谐振器		
455 kHz	Panasonic EFO-A455K04B	±0.3%
2.0 MHz	Murata Erie CSA2.00MG	±0.5%
4.0 MHz	Murata Erie CSA4.00MG	±0.5%
8.0 MHz	Murata Erie CSA8.00MT	±0.5%
16.0 MHz	Murata Erie CSA16.00MX	±0.5%
20.0 MHz	待定	待定

注 1: C1 和 C2 的建议值与上述测试范围内的值相同。
采用较大的电容值有利于提高振荡器的稳定性，但同时也延长了起振时间。
这些值仅供设计参考。由于每个谐振器都有其自身的特性，用户应向谐振器厂商咨询外部元件的适当值，或验证谐振器的实际性能。
2: 所有测试的谐振器均需要外接电容。

表 2-4: 晶体振荡器的典型电容选择

模式	频率	*C1*(1)	*C2*(1)
LP	32 kHz 200 kHz	68 - 100 pF 15 - 30 pF	68 - 100 pF 15 - 30 pF
XT	100 kHz 2 MHz 4 MHz	68 - 150 pF 15 - 30 pF 15 - 30 pF	150 - 200 pF 15 - 30 pF 15 - 30 pF
HS	8 MHz 10 MHz 20 MHz	15 - 30 pF 15 - 30 pF 15 - 30 pF	15 - 30 pF 15 - 30 pF 15 - 30 pF

所使用的晶体		
32.768 kHz	Epson C-001R32.768K-A	± 20 PPM
100 kHz	Epson C-2 100.00 KC-P	± 20 PPM
200 kHz	STD XTL 200.000 kHz	± 20 PPM
2.0 MHz	ECS ECS-20-S-2	± 50 PPM
4.0 MHz	ECS ECS-40-S-4	± 50 PPM
10.0 MHz	ECS ECS-100-S-4	± 50 PPM
20.0 MHz	ECS ECS-200-S-4	± 50 PPM

注 1: 采用较大的电容值有利于提高晶体振荡器的稳定性，但同时也延长了起振时间。这些值仅供设计参考。为避免对低驱动规格的晶体造成过驱动，在 HS 和 XT 模式下，可能需要串联电阻 ***Rs***。由于每个谐振器都有其自身的特性，用户应向谐振器厂商咨询外部元件的适当值，或验证谐振器的实际性能。

2.3.3 振荡器电路的调整

由于 Microchip 器件具有宽工作范围 （频率，电压和温度；取决于订购的具体器件和版本 ），加上各种外部元件 （晶体，电容等），它们的质量和生产工艺也不相同，因此需要进行运行验证，以保证元件的选择符合应用要求。

在选择和安装外部元件时，要考虑多种因素，它们是：

- 放大器增益
- 所需频率
- 晶体谐振频率
- 工作温度
- 电源电压范围
- 起振时间
- 稳定性
- 晶体寿命
- 功耗
- 简化电路
- 标准元器件的使用
- 使用最少元件的元件组合形式

2.3.3.1 晶体，时钟模式，*C1*，*C2*，和 *Rs* 的最佳取值

选择元件的最好方法是应用相关知识和进行大量的试验、测量和测试。

通常，**晶体**的选择只考虑其并联谐振频率，但是，其它参数对您的设计可能也很重要，如温度或频率允差。如果想了解更多关于晶体工作原理和订购信息，应用笔记 AN588 是个很不错的参考。

PICmicro 单片机内部振荡器电路是一个并联振荡器电路，要求选择一个并联谐振器。负载电容通常取值为 20 pF 到 32 pF。电容在这个范围内，晶体的振荡频率最接近设计频率。正如后面将要描述的那样，为了提高其他方面的性能，有时稍微更改一下负载电容值也是很有必要的。

时钟模式主要是根据频率，采用器件数据手册中的 FOSC 参数 （参数 1A）来选择的。选择时钟模式（除 RC 模式外）就是选择不同的增益。 低频选低增益，高频选高增益。如果设计需要，也可根据振荡器电路的特殊要求选择高一些或低一些的增益。

C1* 和 *C2 可先根据晶体厂商建议的和器件数据手册上提供的负载电容来选择。由于晶体厂商不同、电源电压的差异和其它已提及的因素，可导致您的电路与出厂前确定特性参数过程中所使用的电路不尽相同，因此 Microchip 数据手册只能作为一个初始参考。

理想情况下，应在建议的晶体负载范围内选择电容，以便电路处于最高工作温度和最低 VDD 电压下时，晶体均能振荡。高温和低 VDD 电压均会对环路增益起限制作用，所以如果振荡器能工作在这些极端情况下，设计者更确信振荡器在其它温度和电源电压下，都能正常工作。在最高增益情况（最高 VDD 电压和最低工作温度）下，输出正弦波应不被限幅，而在最低增益情况（最低 VDD 电压和最高工作温度）下，其幅度应该足够大，以满足器件数据手册上所列的时钟输入逻辑的要求。

一种缩短起振时间的方法是采用一个大于 ***C1*** *的* ***C2*** 值。这使得在上电时，对通过晶体的振荡信号会产生较大的相移，加速振荡器的起振。

外接电容除了使晶体振荡作出适当的频率响应外，如果增加其容量，还能降低环路增益。通过选择 ***C2*** 可影响回路的总增益。如果晶体过驱动，增大 ***C2*** 可降低增益 （请参阅关于 ***Rs*** 的讨论）。若电容值过大，晶体会储存与释放过量的电流， 因此 ***C1*** 和 ***C2*** 不能过大。然而，测量晶体的功耗（瓦特数）不是一件容易的事情，但如果你所选择的电容值并未偏离建议值太远，就不必担心这个问题。

如果其他外部元件都选好之后，发现晶体仍然过驱动，此时可以接入串联电阻 ***Rs***。晶体是否过驱可通过示波器观察 OSC2 引脚 （驱动引脚）来判断。当示波器探头连接到 OSC1 引脚时，会使引脚负载过重，对系统性能产生不利的影响。注意，示波器探头会将其自身的负载电容加到被测电路中，因此在设计时应考虑到此问题。例如：如果一个电路在 ***C2*** 为 20pF 时正常工作，示波器探头负载电容为 10pF，当探头接入 ***C2*** 端时，实际上 ***C2*** 端的电容变为了 30pF。振荡输出信号不应出现限幅或畸变。过驱动晶体则可导致电路振荡频率跃变到一个高次谐波上，甚至会损坏晶体。

OSC2 引脚的信号应为一个平滑的正弦波，且在时钟输入引脚的最大和最小电平值下轻松保持其平滑度（VDD 为 5V 时，时钟输入引脚的峰峰值为 4V 到 5V 通常即可满足要求）。确保正弦波平滑的一个简单方法，还是使系统工作在设计要求的最低温度和最高 VDD 电压下，检测引脚输出波形。此时，时钟输出应是最大振幅。如果在靠近 VDD 和 VSS 的正弦波顶部和底部出现限幅或畸变，而增加负载电容又可能使流过晶体的电流过大或过于偏离厂商的规定值，则可在输出引脚和 ***C2*** 之间增加一个可调电位器，调整电位器直至正弦波恢复平滑。在低温和高 VDD 电压条件下，将输出的平滑正弦波调整至尽量接近最大幅值，这样即可保证该幅值是晶体的最大工作振幅，而防止晶体过驱动。然后用一个最接近标准阻值的串联电阻 ***Rs*** 代替可调电位器。如果 ***Rs*** 过大，如超过 20kΩ，则输出与输入端的隔离度过大，使得时钟易受到噪声信号干扰。如果必须要用这么大的电阻来防止晶体过驱动，可以尝试增大 ***C2*** 来补偿 ***Rs*** 过大造成的不利影响。尽量找到一个结合点，使 ***Rs*** 在 10kΩ 左右或更小，而负载电容也在厂商指定的 20pF 或 32pF 附近。

2.3.3.1.1 起振

振荡器从休眠状态唤醒时是最难起振的，这是因为两个负载电容都被充电到某个静态值，而相位差在唤醒时又最小。因此需要更多的时间来获得稳定振荡。同时不要忘记低电压、高温和低频时钟模式会降低环路增益，这反过来又会影响到起振。下面的每一种因素都会使情况恶化：

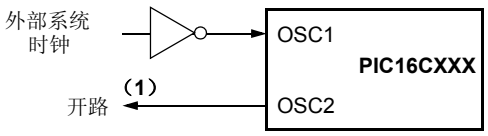
- 低频设计 （低增益时钟模式）
- 低噪声环境 （如电池驱动的器件）
- 在嘈杂的 RF 射频环境之外工作 （如在屏蔽盒内）
- 低电压
- 高温
- 从休眠状态唤醒

实际上，由于噪声有助于激励振荡器，噪声有利于振荡器的起振。

2.3.4 外部时钟输入

如果不使用 PICmicro 单片机的内部振荡器，而通过外部时钟驱动器件，此时应确保将振荡器模式设定在三种晶体模式之一（LP，XT 或 HS），即除 RC 模式之外的模式，这是由于 RC 模式和外部输入时钟信号会产生冲突。理想情况下，应选择与外部输入频率相匹配的模式，但在此这并不重要，因为时钟只是驱动内部逻辑电路而非晶体振荡器环路。可以选择一个频率低于振荡器电路所需频率的时钟模式，从而可节省原本用于反相放大器的那部分功耗。应确保 OSC2 信号幅值可以满足器件要求的逻辑阈值。

图 2-3: 外部器件的时钟输入运行 （HS，XT 或 LP 振荡器模式）



注 1: 为降低系统噪声，可连接一个接地电阻。
这将使系统工作电流升高。

2.3.5 外部晶体振荡器电路

有时单个晶体需要为多个器件提供时钟信号。由于 Microchip 不推荐在 PICmicro 的内部振荡器电路上连接其它逻辑电路，这就需要一个外部晶体振荡器电路。这样每个器件都有一个外部时钟源，外部时钟源能驱动的器件数量取决于缓冲器的驱动能力。这个电路对于多个 PICmicro 单片机需要彼此同步操作同样有用。

外部时钟源既可使用现成的振荡器，也可由带 TTL 门电路的简单的振荡环路构建。现成的振荡器有较宽的工作范围和更好的稳定性。带有 TTL 门电路的设计合理和晶体振荡器可提供良好的性能。可使用两种类型的晶体振荡器电路：串联谐振电路和并联谐振电路。

图 2-4 为外部并联谐振晶体振荡器电路。该电路设计使用晶体的基频。74AS04 反相器提供并联振荡所需的 180 度相移。4.7kΩ 电阻为电路的稳定工作提供一个负反馈，10kΩ 电位器将 74AS04 偏置在线性工作区。

图 2-4: 外部并联谐振的晶体振荡器电路

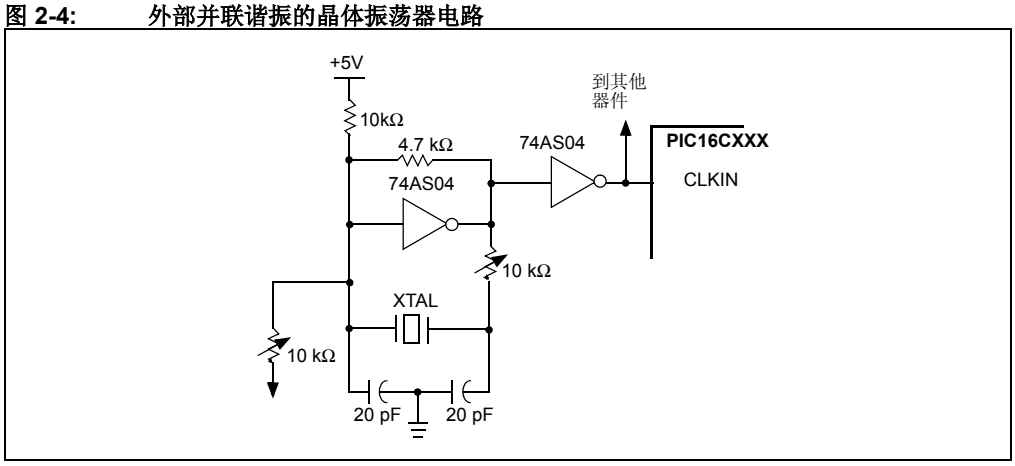
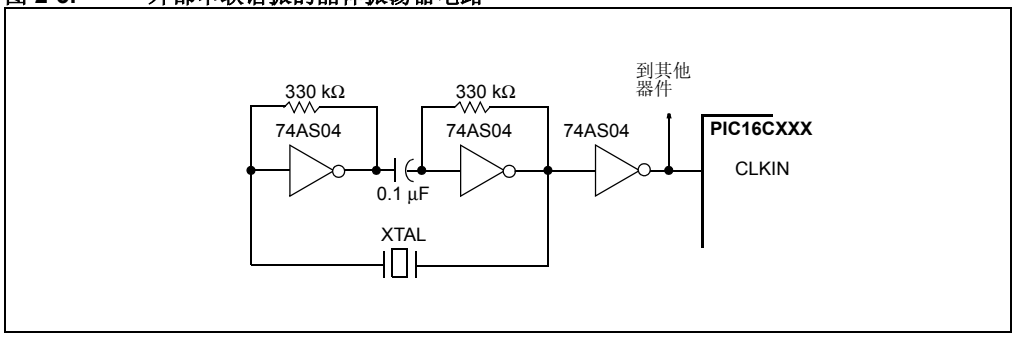


图 2-5 为外部串联谐振振荡器电路。该电路同样设计使用晶体的基频。每个反相器在串联谐振电路中都提供 180 度的相移。 330kΩ 电阻通过负反馈将反相器偏置在线性工作区。

图 2-5: 外部串联谐振的晶体振荡器电路

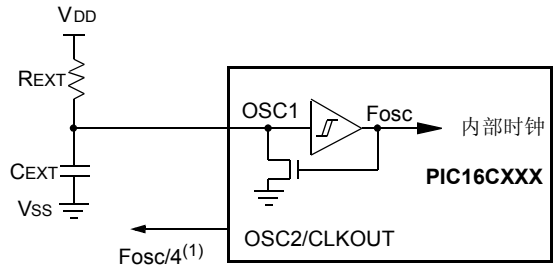


当器件由外部时钟源驱动时（如图 2-4 或图 2-5），单片机的振荡模式必须配置成 LP、XT 或 HS 模式 （图 2-3）。

2.4 外部 RC 振荡器

对于对定时要求不高的应用，“XTRC”器件选项额外降低了成本。 RC 振荡器频率是电源电压、电阻 （REXT）、电容 （CEXT）和工作温度的函数。另外，由于正常的制造工艺参数的差异，每个器件的振荡频率也会有所不同。而不同封装的引线电容不同，也会影响振荡频率，特别是 CEXT 值较小时。用户还需要考虑由于外接电阻 REXT 和电容 CEXT 的公差所带来的影响。图 2-6 显示了如何为 PIC16CXXX 外接 RC 电路。当 REXT 的阻值低于 2.2kΩ 时，振荡器工作可能变得不稳定，或完全停止振荡。 REXT 值很高时 （如 1 MΩ），振荡器则易受噪声、湿度和漏电流的干扰。因此，我们建议将 REXT 保持在 3kΩ 和 100kΩ 之间。

图 2-6: EXTRC 振荡器模式



注 1: 该输出引脚也可以被设置成通用 I/O 引脚。

尽管在没有外部电容 （CEXT = 0 pF）的情况下，振荡器仍可工作，但考虑到噪声和稳定性等因素，我们仍建议使用一个大于 20pF 的电容。在没有外部电容或外部电容很小的情况下，由于外部电容的变化，如 PCB 上走线的电容和封装的引线电容，振荡频率会发生很大的变化。

由于正常制造工艺参数的差异，每个器件的 RC 频率不同，具体可查阅描述 RC 频率特性数据。外接电阻越大，频率偏差越大（因为电阻越大，漏电流的变化对 RC 频率影响越大）；外接电容越小，频率偏差也越大 （因为外接电容越小，输入电容容量的偏差对频率影响越大）。

给定 REXT/CEXT 值，由于 VDD 引起的振荡器频率变化，可参阅相关的特性数据。同样，对给定 REXT、 CEXT 和 VDD 值，由于工作温度引起的振荡器频率变化也请参阅特性数据。

振荡器频率的 4 分频信号可由 OSC2/CLKOUT 引脚输出，可用作测试或同步其它逻辑单元。（请参阅 **“架构”一章**中**图 4-3：“时钟 / 指令周期”**的相关波形）。

2.4.1 RC 起振

随着器件工作电压的上升，当与 RC 相连的引脚电压达到输入阈值（见**“电气规范”一章**中的参数 D032 和 D042）后， RC 将立即起振。 RC 的起振时间取决于许多因素，如：

- 所使用的电阻值
- 所使用的电容值
- 器件 VDD 电压的上升时间
- 系统温度

2.5 4MHz 内部 RC 振荡器

在 VDD 为 5V，25°C 条件下，内部 RC 振荡器（不是所有的器件都有）提供了一个固定的 4MHz 系统时钟，其振荡频率随工作电压和温度变化，请参阅器件数据手册的 **“电气规范”一章**。

OSCCAL 寄存器的值用于调整内部 RC 振荡器的频率。 Microchip 烧写进器件的校准值可对内部振荡器频率进行微调，以消除因生产工艺偏差引起的频率偏差。 OSCCAL 寄存器的 CAL3:CAL0 位可用于在一个频率范围内精确调校振荡频率。 CAL3:CAL0（从 0000 到 1111）的值越大，将振荡频率调校得越高。

当 4 MHz 的内部 RC 振荡器频率无法单独通过 CAL3:CAL0 位精确调校得到时，可将 RC 振荡器频率增加或减小一个偏移频率。 CALFST 和 CALSLW 作为正负偏移频率的使能位，用来将内部 RC 频率偏移到 CAL3:CAL0 的调整范围内。

设置 CALFST 位可将振荡频率向较高频率偏移，而设置 CALSLW 位可将振荡频率向较低频率偏移。

当器件复位时，OSCCAL 寄存器被强制为中间值（CAL3:CAL0 = 7h，CALFST 和 CALSLW 不提供偏置）。

寄存器 2-1: OSCCAL 寄存器

R/W-0	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0	U-0	U-0
CAL3	CAL2	CAL1	CAL0	CALFST	CALSLW	—	—
bit 7							bit 0

bit 7:4 **CAL3:CAL0:** 内部 RC 振荡器的校准位
0000= 在调整范围内的最低时钟频率
•
•
•
1111= 在调整范围内的最高时钟频率

bit 3 **CALFST:** 振荡器频率偏移位
1 = 将内部 RC 振荡器频率提高到 CAL3:CAL0 调整范围内
0 = 不提供偏移

bit 2 **CALSLW：**振荡器频率偏移位
1 = 将内部 RC 振荡器频率降低到 CAL3:CAL0 调整范围内
0 = 不提供偏移

注： 当 **CALFST、 CALSLW** 两位均为 1 时， CALFST 位将超越 CALSLW 位。

bit 1:0 **未使用位：**读作 0

注： 为了与将来的器件兼容，修改 OSCCAL 寄存器时应将这些位写为 ’0’

符号		
R = 可读	W = 可写	
U = 未用，读为 0		- n = 上电复位值

注： OSCCAL 用来消除由于制造工艺偏差引起的内部 RC 振荡频率的变化。不得修改 Microchip提供的OSCCAL值；对于所有定时要求高的功能应根据具体应用软件进行调整。

图 2-7 所示为未校准时器件可能的频率（VDD = 5V，25°C 且 OSCCAL = 70h），以及 OSCCAL 寄存器对频率的调校范围。

图 2-7: 理想的内部 RC 振荡器频率 与 OSCCAL 寄存器值的关系

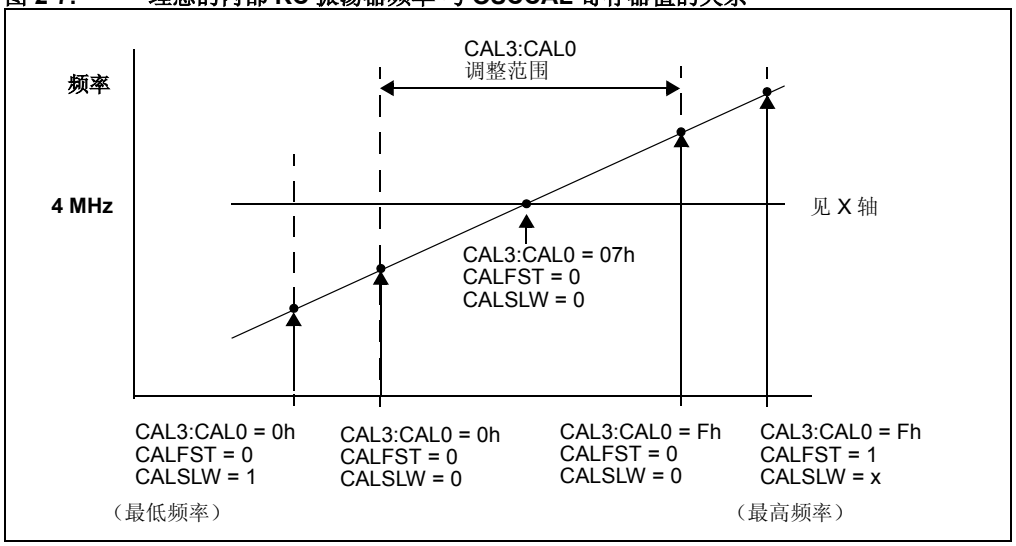


图 2-8 所示为通过选择 CAL3:CAL0 其中一个值，将振荡频率调校到 4MHz 的例子。这些 CAL3:CAL0 位可视为精确调校的频率。有时非校准点的频率不能仅仅通过 CAL3:CAL0 位的值精确调校到 4MHz。因此增加了两个附加位产生一个大的正向或负向频率偏移，将非校准点的频率偏移到能精确调整的范围内。这些附加位是 CALSLW 和 CALFST，它们可对内部 RC 频率产生偏移。 CALSLW 和 CALFST 位的作用如图 2-9 和图 2-10 所示。

图 2-8: CAL3:CAL0 位对内部 RC 振荡器频率的调整

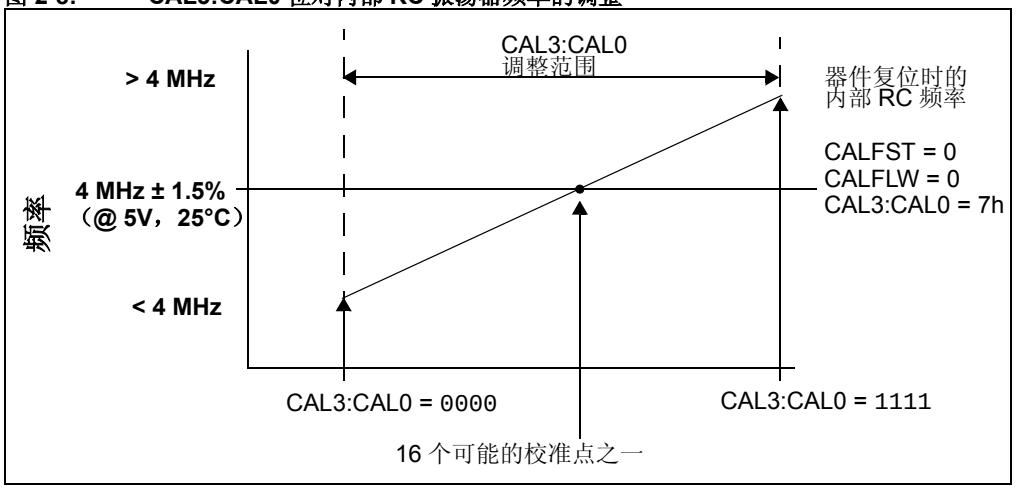


图 2-9: CALFST 对内部 RC 频率的正向偏移

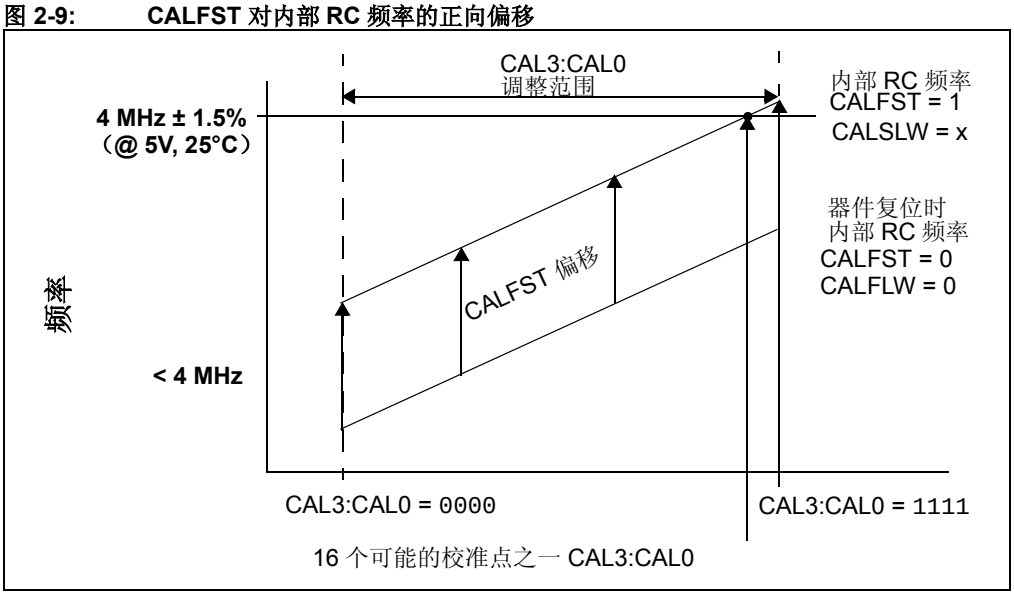
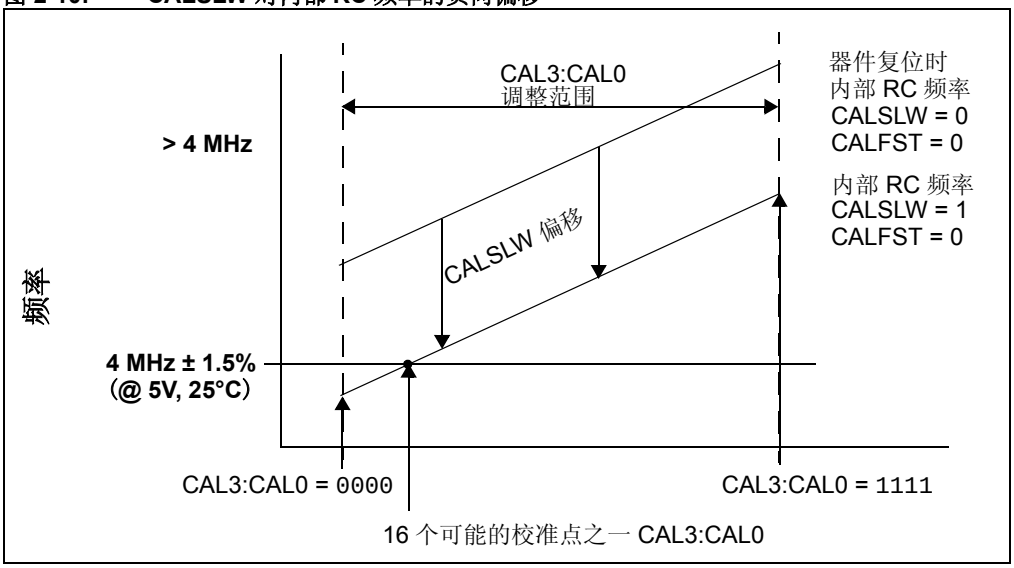


图 2-10: CALSLW 对内部 RC 频率的负向偏移



校准指令被烧写在程序存储器的末地址处。该指令中包含内部 RC 振荡器的校准值，以 RETLW XX 指令的形式出现，其中 XX 就是校准值。为了获得校准值，可调用 CALL YY 指令，YY 是器件内用户可存取程序存储器的末地址 （即 RETLW XX 指令的地址）。校准值被装载到 W 寄存器后，程序可执行一条 MOVWF OSCCAL 指令，把校准值装载到内部 RC 振荡器的校准寄存器中。表 2-5 所示为校准值的存储位置，其位置取决于程序存储器的大小。

表 2-5: 校准值存储位置

程序存储器大小 （字）	校准值存储位置
512	1FFh
1K	3FFh
2K	7FFh
4K	FFFh
8K	1FFFh

注 1: 擦除器件 （窗口型）将擦除厂家为内部振荡器所烧写的校准值。

擦除窗口型器件之前，应先保存内部振荡器的校准值。为了确保校准值不意外丢失，可将校准值写在器件的封装上。
在烧写器件之前，应先将保存的校准值恢复到程序存储器的校准区中。

注 2: OSCCAL<1:0> 未定义，应写作 ‘0’。这将保证其与未来器件兼容。

2.5.1 时钟输出

当配置字地址 （2007h）的 FOSC2、 FOSC1、 FOSC0 位烧写为 “101”（内部 RC 振荡器模式），或 “111”（外部 RC 振荡器模式）时，内部 RC 振荡器可配置为在 CLKOUT 引脚输出一个时钟信号。此时钟信号为振荡信号的 4 分频，可用于测试或同步其它逻辑电路。

当内部 RC 振荡器的校准值被意外擦除后，时钟输出功能可帮助用户决定校准值的大小。用户可编写一段程序，该程序修改 （增加 / 减少） OSCCAL 寄存器中的值。当 CLKOUT 引脚处于 5V 和 25°C 环境下，输出频率为 1MHz （±1.5%）时， OSCCAL 寄存器的值即是正确的校准值。须将该值写入一个端口或通过串行口移位输出，而后将该校准值记下并烧写到校准区。

2.6 休眠模式对片内振荡器的影响

当器件执行一条 SLEEP 指令后，片内时钟和振荡器均被关闭，器件状态保持为指令周期的起始状态 （Q1 状态）。随着振荡器的关闭，OSC1 和 OSC2 引脚的信号将会停止振荡。由于没有了晶体管的开关电流，使休眠模式达到器件的最低电流消耗 （仅有泄漏电流）。使能任何在休眠状态下仍能工作的片内功能，都将增加休眠状态的电流消耗。用户可通过外部复位、看门狗定时器复位或中断将器件从休眠状态唤醒。

表 2-6: 休眠模式下 OSC1 和 OSC2 引脚状态

OSC 模式	OSC1 引脚	OSC2 引脚
EXTRC	悬空，经外部电阻上拉为高电平	逻辑低电平
INTRC	无效	无效
LP、 XT 和 HS	反馈反相放大器被禁止，处于静态电平	反馈反相放大器被禁止，处于静态电平

关于 $\overline{\text{MCLR}}$ 外部复位和休眠的相关延时，请参阅 **“复位”一章**中的表 3-1。

2.7 器件复位对片内振荡器的影响

器件复位对片内晶体振荡器电路并无影响。复位时，振荡器同未复位时一样正常工作。复位期间，器件逻辑操作被保持在 Q1 状态，因此当器件退出复位时，总是处于指令周期的起始状态。

OSC2 引脚被用于外部时钟输出时 （EXTRC 模式）时，其在复位期间保持为低电平。一旦引脚 $\overline{\text{MCLR}}$ 处于 V_{IH} 电平 （输入高电平）， RC 振荡器将开始起振。关于 $\overline{\text{MCLR}}$ 外部复位和休眠的相关延时，请参阅 **“复位”一章**中的表 3-1。

2.7.1 上电延时

有两个定时器为上电过程提供必要的延时。一个是振荡器起振定时器 OST，确保在晶体振荡器的振荡达到稳定前，器件始终处于复位状态。另一个是上电延时定时器 PWRT，它只为上电过程（由上电复位 POR 和欠压复位 BOR 引起的）提供一个固定的 72ms（标称值）延时。PWRT 的作用是确保在电源电压稳定前，器件始终处于复位状态。有这两种片内定时器，大部分的应用无需外接复位电路。更多关于复位的信息，请参阅 **“复位”一章**。

2.8 设计技巧

问 1: *上电后通过示波器观察 OSC2 引脚，没有时钟信号。这是什么原因？*

答 1:

1. 运行了 SLEEP 指令，却没有通过唤醒源 （如 WDT、MCLR 或中断）唤醒器件。检测是否代码将器件置于休眠状态而没有提供唤醒。可能的话，为 MCLR 引脚输入一个低电平脉冲来唤醒器件。上电时将 MCLR 引脚保持低电平将使晶体振荡器有更多的时间起振，但 MCLR 引脚变为高电平之前，程序计数器不会计数。
2. 为设计的工作频率选择了错误的时钟模式。对于一个空白器件，默认的振荡器模式为 EXTRC （外部 RC）。大部分器件的时钟均被选择为默认的 RC 模式，这种模式在外接晶体或谐振器时，是不能起振的。检查时钟模式是否被正确设置。
3. 没有按照正常顺序上电。如果一个 CMOS 电路通过某个 I/O 引脚先于上电过程而上电，就可能发生问题 （硬件闭锁，异常启动等）。发生欠压复位、启动时的电源线干扰、VDD 电压的上升速率过慢等均可能引发问题。试着在器件上电时，断开 I/O 引脚与外部器件的连接，并使用一个已知的性能良好且电压上升速率快的电源。这个问题并没有所说的那样严重，但是有可能存在。关于欠压和上电顺序，请参考器件数据手册中关于上电的介绍。
4. 电容 ***C1*** 、***C2*** 与晶体的连接不正确或者电容的容量有误。请确保所有的硬件连接都是正确的。按器件数据手册上提供的 C1、C2 值连接，振荡器一般均可运行，只是对于你的具体设计，它们可能不是最佳值。

问 2: *PICmicro 单片机已运行，但是工作频率比晶体谐振频率高很多。*

答 2:

振荡器电路的增益太高。请参阅 **2.3 “晶体振荡器 / 陶瓷谐振器”** ，帮助您正确选择 ***C2***（可能要选得大一些）、***Rs***（可能要接此电阻）和***时钟模式***（可能选错了时钟模式）。这种情况尤其可能发生于低频晶体，如常用的 32.768 kHz 晶体。

问 3: *设计的系统运行良好，只是频率稍有偏移，应该如何调整它？*

答 3:

改变 ***C1*** 的值可以影响振荡器的频率。如果使用串联谐振晶体，其谐振频率与并联同一频率的谐振晶体不同。

问 4: *电路板工作良好，然而有时突然停止工作或者计时丢失。*

答 4:

仔细检查你的程序，是否是程序原因引起计时丢失。此外，还有可能是因为振荡器的输出幅度不够高而不能可靠地触发振荡输入。

问 5: *我使用的是带内部 RC 振荡器的器件，不小心擦除了校准值。我该怎么办？*

答 5:

如果对你来说，器件振荡频率不是太重要，你可继续使用它。

如果器件频率的确很关键，你可以买一片新的窗口型器件，或者按 **2.5.1 “ 时钟输出 ”** 中所述的方法恢复校准值。

2.9 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与振荡器相关的应用笔记有：

标题	应用笔记 #
PIC16/17 Oscillator Design Guide	AN588
Low Power Design using PIC16/17	AN606

2.10 版本历史

版本 A

这是描述 PICmicro 单片机振荡器的初始发行版。

第 3 章 复位

目录

本章包括以下一些主要内容：

3.1 简介 3-2
3.2 上电复位、上电延时定时器、起振定时器、欠压复位和奇偶校验错误复位 3-4
3.3 寄存器和状态位的值 3-10
3.4 设计技巧 3-16
3.5 相关应用笔记 3-17
3.6 版本历史 3-18

3.1 简介

复位逻辑用来让器件进入一个已知状态。产生复位的原因可由相关的状态位来判断。该复位逻辑的设计特点在于降低了系统成本，提高了系统可靠性。

器件有以下几种复位方式：

a) 上电复位 （POR）
b) 正常工作状态下的 $\overline{\text{MCLR}}$ 复位
c) 休眠状态下的 $\overline{\text{MCLR}}$ 复位
d) 正常工作状态下 WDT 复位
e) 欠压复位 （BOR）
f) 奇偶校验错误复位 （PER）

大多数寄存器不受复位的影响；在上电复位（POR）时，它们的状态未知，且不会被其它任何复位所改变。而其它寄存器在上电复位 （POR）、 $\overline{\text{MCLR}}$、 $\overline{\text{WDT}}$ （看门狗定时器）复位、欠压复位 （BOR）、奇偶校验错误复位 （PER）和休眠状态下的 $\overline{\text{MCLR}}$ 复位时，被强行置于 “复位状态”。

器件的奇偶校验位可用于验证程序存储器的内容。

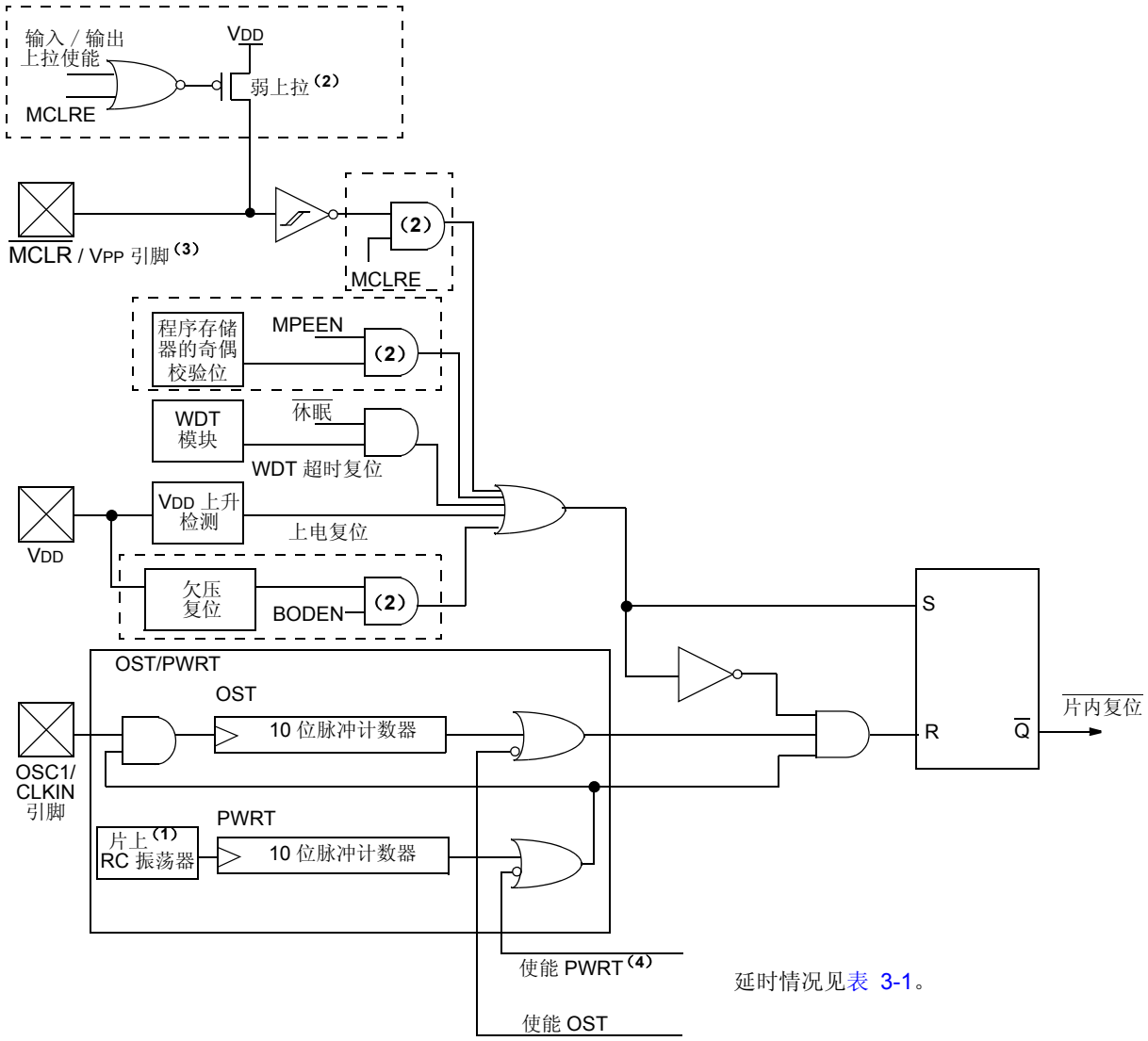
大多数寄存器不受 WDT （看门狗定时器）唤醒复位的影响，这是因为 WDT 唤醒复位被看作是恢复正常运行。如表 3-2 所示，状态位 $\overline{\text{TO}}$，$\overline{\text{PD}}$，$\overline{\text{POR}}$，$\overline{\text{BOR}}$ 和 $\overline{\text{PER}}$ 在不同的复位情况下有着不同的置 “1” 和清 “0” 状态。这些状态位在软件中用于判断复位的类型。表 3-4 给出了所有寄存器的复位状态的完整说明。

图 3-1 给出了一个器件复位电路的简化框图。该框图是复位功能的超集。如果想了解特定器件上具体有哪些复位电路，请参见器件的数据手册。

注：	当 PICmicro® 单片机处在复位状态时，内部时钟相位停留在 Q1 （即一个指令周期的起始处）。

所有的新器件在 $\overline{\text{MCLR}}$ 复位信号的传输路径中均有一个 $\overline{\text{MCLR}}$ 噪声滤波器，它可以检测和滤除小的尖脉冲信号。请参见 **“电气规范”一章**中，有关脉冲宽度规范中参数 30 的说明。

图 3-1: 片内复位系统的简化框图



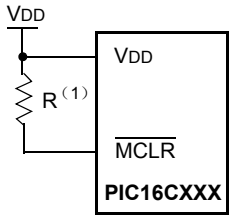
注 1：该振荡器与 CLKIN 引脚上的 RC 振荡器或 INTRC 振荡器不相关。
2：不是所有器件都具备虚线方框中的电路，参见具体器件的数据手册。
3：在某些器件中，该引脚可能被配置为通用输入引脚。
4：早期的 PICmicro 单片机中，配置位 PWRTE = 1 表示使能；而所有其它器件则是 $\overline{\text{PWRTE}}$ = 0 表示使能。

3.2 上电复位、上电延时定时器、起振定时器、欠压复位和奇偶校验错误复位

3.2.1 上电复位 （POR）

器件检测到 VDD 电压上升时，会产生一个上电复位脉冲。要使用上电复位功能，可直接（也可通过一个电阻）将 MCLR 引脚与电源 VDD 相连，如图 3-2 所示。这样可以节省一般上电复位电路用于产生一个上电复位所需的外接 RC 元件。VDD 电压的上升速率应大于最小上升速率。详见**“电气规范”一章**中关于参数 D003 和参数 D004 的说明。

图 3-2: 上电复位的使用

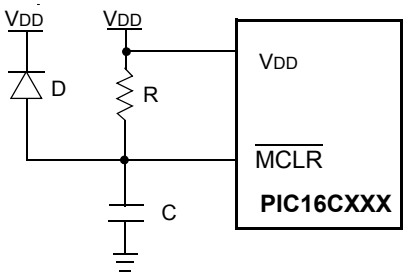


注： 该电阻是可选的。

当器件退出复位状态（开始正常工作）时，其工作参数（电压，频率，温度等）都应在相应的工作范围之内，否则器件将不能正常工作。应保证足够长的延时以使所有工作参数达到规定值。

图 3-3 所示为针对上升速率缓慢的电源的一种上电复位电路。只有在电源 VDD 的上升速率过慢时，才需要这一外部上电复位电路。当电源 VDD 掉电时，二极管 ***D*** 可使电容迅速放电。

图 3-3: 外部上电复位电路（用于 VDD 上升速率过慢时）



注： 为保证电阻 R 上的压降不超过 0.2V，建议 R < 40 kΩ。过大的电压降会降低 MCLR/VP 引脚上高电平 VIH 的值。

3.2.2 上电延时定时器 （PWRT）

上电复位 （POR）或欠压复位 （BOR）时，上电延时定时器提供固定的 72 ms 延时，参见 **“电气规范”一章**中关于参数 33 的说明。上电延时定时器工作在专用的内部 RC 振荡器上。只要 PWRT 有效，器件就保持在复位状态。 PWRT 的延时使 VDD 上升到一个适当电压。上电延时定时器的使能配置位可使能 / 禁止上电定时器。若欠压复位被使能，上电延时定时器应总是被使能。现在，上电定时器的配置位 PWRTE = 0 表示使能，而在早期的器件中配置位 PWRTE = 1 表示使能。由于所有新器件都用 PWRTE = 0 表示使能，因此在后续章节中，我们只介绍这些新器件的使能方式。参见具体数据手册以确保正确设置该位的极性。

由于电源电压、环境温度和制造工艺的影响，不同器件的上电延时时间也会有所不同。详细情况请参看 DC 参数。

3.2.3 起振定时器 （OST）

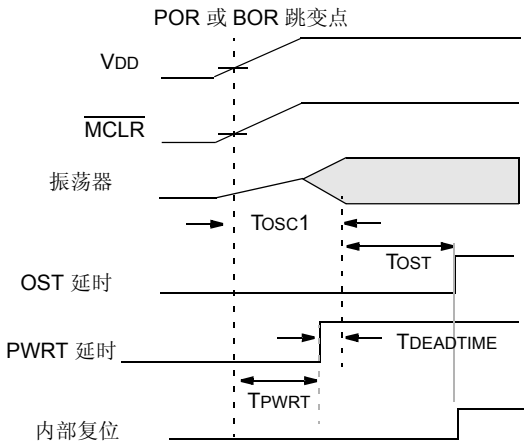
当 PWRT 延时结束后，起振定时器 （OST）提供了一个 1024 个振荡周期的延时 （从 OSC1 输入）。这是为了保证晶体或陶瓷谐振器起振并建立稳定的振荡。

只有在 XT、 LP 和 HS 振荡方式下，并且只有在上电复位、欠压复位或器件从休眠状态中被唤醒时， OST 定时器才启动。

起振定时器对 OSC1/CLKIN 引脚上的振荡脉冲计数，但只有振荡信号的幅值达到振荡器的输入阈值时，起振定时器才开始递增计数。该延时使晶体振荡器和谐振器在器件退出 OST 延时之前就已经达到稳定状态。延时的时间长度是晶体振荡器 / 谐振器工作频率的函数。

图 3-4 给出了起振定时器和上电定时器的运行。对于低频晶体，起振时间可能很长，这是由于低频振荡器的起振时间比上电定时器的延时长。从上电定时器延时结束到振荡器起振的时间叫死区时间，这一死区时间 （***TDEADTIME***）没有最大值和最小值的范围限制。

图 3-4: 振荡器起振时间



Tosc1 = 晶振的振荡电平达到振荡器起振定时器 （OST）可检测到的电平的时间

TOST = 1024TOSC。

3

复位

3.2.4 上电顺序

上电时延时顺序为：首先检测 POR，随后，如果 PWRT 被使能，进入 PWRT 延时。 PWRT 延时时间到之后， OST 被激活。总延时时间取决于振荡器的配置和 PWRTE 位的状态。例如， RC 模式下 PWRTE 位置“1”（PWRT 禁止）的情况下，将不会出现延时。图 3-5，图 3-6 和图 3-7 描述了延时时序。

由于延时发生于内部 POR 脉冲，因此若 $\overline{\text{MCLR}}$ 引脚保持足够长时间的低电平，延时将结束。将 $\overline{\text{MCLR}}$ 电平拉高后程序将立即执行（图 3-7）。这对于测试或同步并行工作的多个器件非常有用。

如果延时结束后，器件的工作电压还未达到电气规范的要求， $\overline{\text{MCLR}}$/VPP 引脚必须保持低电平，直到电压满足电气规范的要求为止。使用外部 RC 延时电路可以满足这类应用要求。

表 3-1 给出了不同情况下的延时情况，而图 3-5 到图 3-8 所示为器件上电时的四种情况：

表 3-1: 不同情况下的延时

振荡器配置	上电定时器		欠压复位	从休眠状态唤醒
	使能	禁止		
XT，HS，LP	72 ms + 1024TOSC	1024TOSC	72 ms + 1024TOSC	1024TOSC
RC	72 ms	—（1）	72 ms	—（1）

注 1： 内置 / 外部 RC 模式时，有 250µs （标称值）的延时。

图 3-5: 上电时的延时序列 （$\overline{\text{MCLR}}$ 引脚与 VDD 相连）

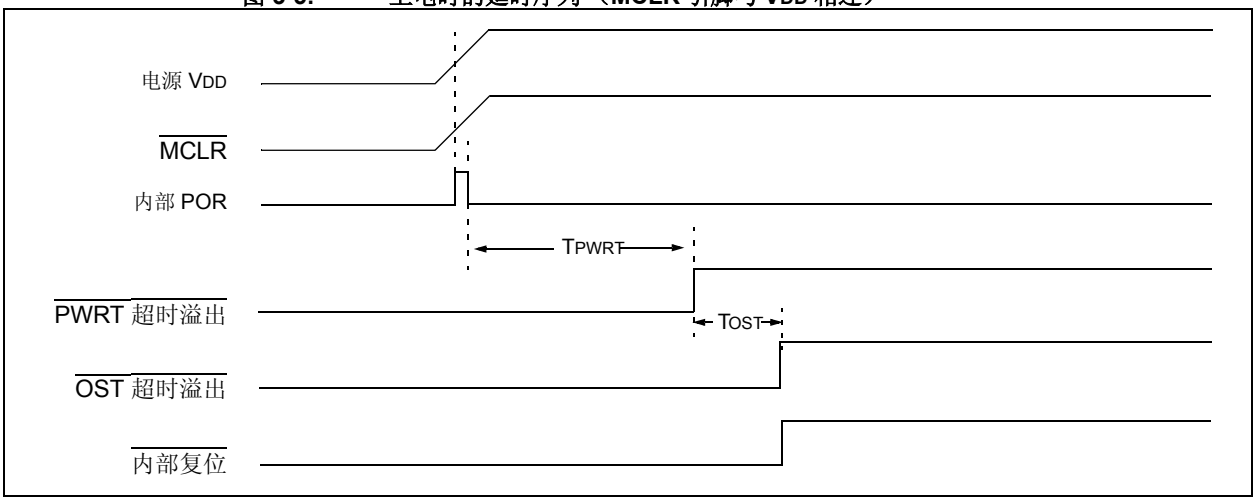


图 3-6: 上电时的延时序列（$\overline{\text{MCLR}}$ 未连接到 VDD）：情形 1

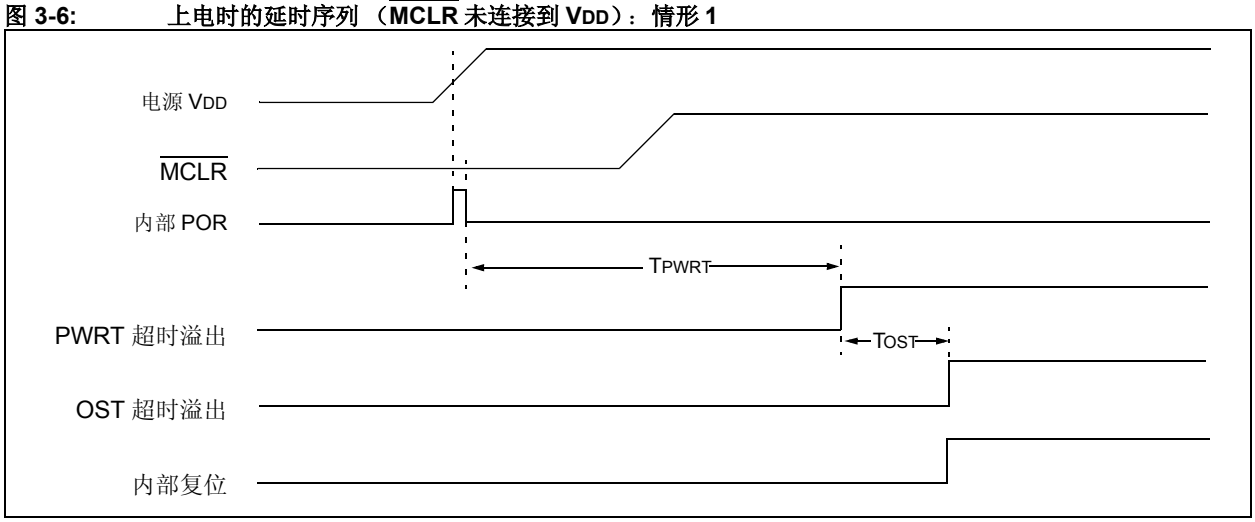


图 3-7: 上电时的延时序列（$\overline{\text{MCLR}}$ 未连接到 VDD）：情形 2

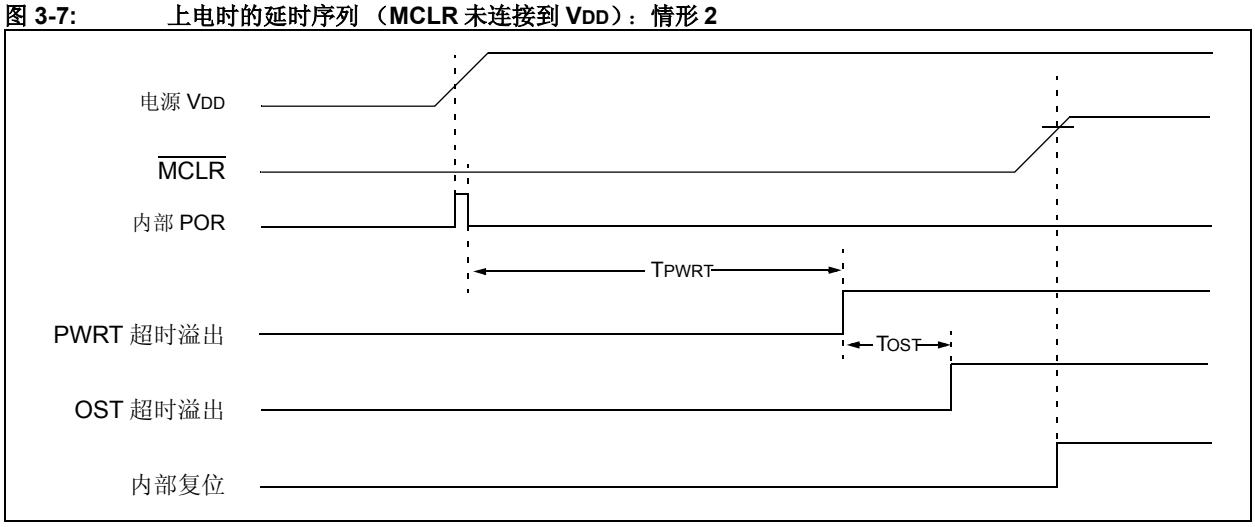
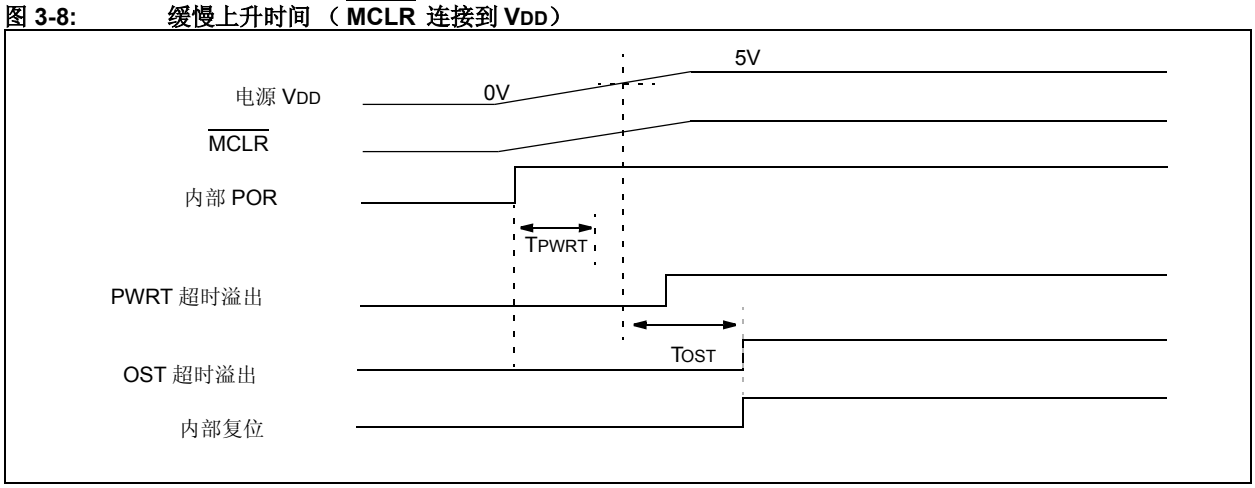


图 3-8: 缓慢上升时间（$\overline{\text{MCLR}}$ 连接到 VDD）



3.2.5 欠压复位 (BOR)

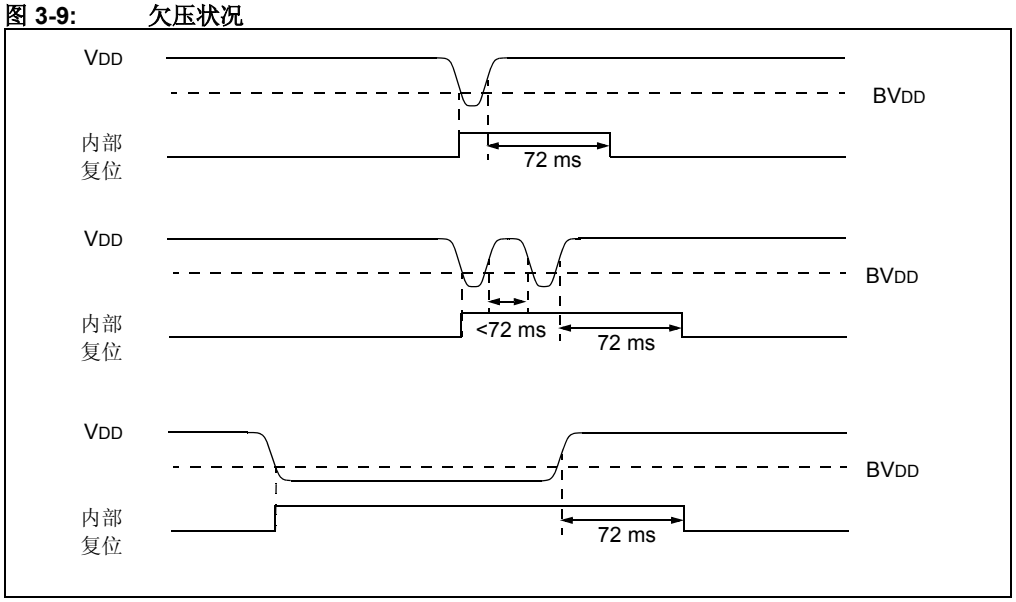
当器件的电压降到跳变点（BVDD）以下时，器件的欠压复位电路使器件处于复位状态。这将确保器件不在有效工作电压之外继续执行程序。在交流电应用或大功率电池应用中，当大型负载（如汽车）接入而导致器件的工作电压暂时降低到规定工作电压以下时，就经常用到欠压复位。

注：	在利用片上欠压复位实现电源电压监控功能（监视电池老化）以前，请查看电气规范，以确保各项参数满足您的要求。

BODEN 配置位可禁止（清除 / 被烧写时）或使能（置位时）欠压复位电路。如果 VDD 电压跌落到 BVDD（典型值为 4.0V。详见 **“电气规范”一章**中的参数 D005）以下，且持续时间超过参数 35，这种欠压状况将使器件复位。如果 VDD 电压跌落到 BVDD 以下，持续时间小于参数 35，就不能保证一定会复位。器件保持欠压复位状态，直至 VDD 电压上升到 BVDD 以上。此时，上电定时器将被激活，并使器件继续保持 72 ms 的复位状态。如果上电定时器运行时，VDD 电压又降到 BVDD 以下，器件将重新回到欠压复位状态，且上电定时器将被重新初始化。一旦 VDD 电压再次上升到 BVDD 以上，上电定时器将重新启动一个 72ms 的延时。图 3-9 显示了典型的欠压复位状况。

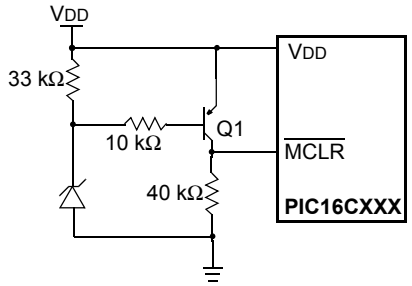
当 BODEN 位置 “1” 时，只要 VDD 电压低于 BVDD，器件就将保持在复位状态。这种状态包括上电时的情况。

图 3-9: 欠压状况



某些器件不具备片内欠压复位电路，而有时应用中的欠压复位跳变点不能满足应用要求时，可采用图 3-10和图 3-11所示的外部欠压复位电路。具体应用时，需要先确定其是否符合应用的要求。

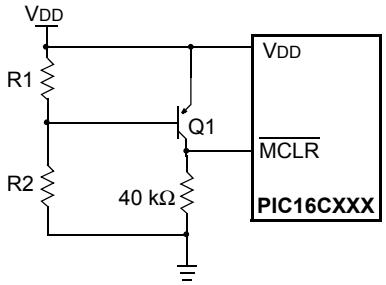
图 3-10: 外部欠压复位电路 1



当 VDD 下降到 （Vz + 0.7V）时，该电路将使单片机复位
其中 Vz = 稳压二极管电压

注 1：使用该电路时，应禁止内部欠压复位功能。
2：应根据晶体管的特性参数来调整电阻值。

图 3-11: 外部欠压复位电路 2



注 1：这种欠压复位电路较为便宜，但精度不高。当 V_{DD} 低于下式计算值时，晶体管 Q1 将关闭。

$$V_{DD} \cdot \frac{R1}{R1 + R2} = 0.7V$$

2：使用该电路时，应禁止内部欠压复位功能。
3：应根据晶体管的特性参数来调整电阻值。

3.3 寄存器和状态位的值

表 3-2: 状态位及其含义

$\overline{POR}$	$\overline{BOR}$(1)	$\overline{TO}$	$\overline{PD}$	条件
0	x	1	1	上电复位
0	x	0	x	非法；在上电复位时， $\overline{TO}$ 置为 1
0	x	x	0	非法；在上电复位时， $\overline{PD}$ 置为 1
1(2)	0	1	1	欠压复位
1(2)	1(2)	0	1	看门狗定时器 （WDT）复位
1(2)	1(2)	0	0	WDT 唤醒复位
1(2)	1(2)	u	u	在正常运行时 $\overline{MCLR}$ 复位
1(2)	1(2)	1	0	休眠时 $\overline{MCLR}$ 复位

图注： u = 未改变， x = 未知， - = 未用，读为 0

注 1：不是每种器件都具备欠压复位 （**BOR**）电路。

2：在给定条件下这些位并不改变，在上电复位 （**POR**）或欠压复位 （**BOR**）后，这些位将初始化 （置位）并读为 1。

表 3-3: 特殊寄存器的初始化条件

条件	程序计数器	状态寄存器	PCON 寄存器
上电复位	000h	0001 1xxx	u--- -10x
正常运行时的 $\overline{\text{MCLR}}$ 复位	000h	000u uuuu	u--- -uuu
休眠时的 $\overline{\text{MCLR}}$ 复位	000h	0001 0uuu	u--- -uuu
WDT 复位	000h	0000 1uuu	u--- -uuu
WDT 唤醒	PC + 1	uuu0 0uuu	u--- -uuu
欠压复位	000h	0001 1uuu	u--- -uu0
休眠时的中断唤醒	PC + 1 (1)	uuu1 0uuu	u--- -uuu

图注： u = 不变 x = 未知， - = 未用，读为 ‘0’

注 1：当器件被中断唤醒且全局允许位 GIE 置 1 时，在执行 PC+1 指令后， PC 指针将指向中断向量 0004h。

2：如果一个状态位未使用，则读为 ‘0’。

表 3-4: 特殊功能寄存器的初始化条件

寄存器	上电复位 欠压复位	- 正常工作 - 休眠或 - WDT 复位时 的 MCLR 复位	- 中断 - WDT 溢出 从休眠状态唤醒
ADCAPL	0000 0000	0000 0000	uuuu uuuu
ADCAPH	0000 0000	0000 0000	uuuu uuuu
ADCON0	0000 00-0	0000 00-0	uuuu uu-u
ADCON1	---- -000	---- -000	---- -uuu
ADRES	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADTMRL	0000 0000	0000 0000	uuuu uuuu
ADMRH	0000 0000	0000 0000	uuuu uuuu
CCP1CON	--00 0000	--00 0000	--uu uuuu
CCP2CON	0000 0000	0000 0000	uuuu uuuu
CCPR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2L	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR2H	xxxx xxxx	uuuu uuuu	uuuu uuuu
CMCON	00-- 0000	00-- 0000	uu-- uuuu
EEADR	xxxx xxxx	uuuu uuuu	uuuu uuuu
EECON1	---0 x000	---0 q000	---0 uuuu
EECON2	-	-	-
EEDATA	xxxx xxxx	uuuu uuuu	uuuu uuuu
FSR	xxxx xxxx	uuuu uuuu	uuuu uuuu
GPIO	--xx xxxx	--uu uuuu	--uu uuuu
I2CADD	0000 0000	0000 0000	uuuu uuuu
I2CBUF	xxxx xxxx	uuuu uuuu	uuuu uuuu
I2CCON	0000 0000	0000 0000	uuuu uuuu
I2CSTAT	--00 0000	--00 0000	--uu uuuu
INDF	-	-	-
INTCON	0000 000x	0000 000u	uuuu uuuu (1)
LCDCON	00-0 0000	00-0 0000	uu-u uuuu
LCDD00 to LCDD15	xxxx xxxx	uuuu uuuu	uuuu uuuu
LCDPS	---- 0000	---- 0000	---- uuuu
LCDSE	1111 1111	1111 1111	uuuu uuuu
OPTION_REG	1111 1111	1111 1111	uuuu uuuu
OSCCAL	0111 00--	uuuu uu--	uuuu uu--
PCL	0000 0000	0000 0000	PC + 1 (2)
PCLATH	---0 0000	---0 0000	---u uuuu
PCON	---- --0u	---- --uu	---- --uu
PIE1	0000 0000	0000 0000	uuuu uuuu
PIE2	---- ---0	---- ---0	---- ---u
PIR1	0000 0000	0000 0000	uuuu uuuu
PIR2	---- ---0	---- ---0	---- ---u

图注： u = 未改变， x = 未知， - = 未用，读为 '0'， q = 根据条件而变化

注 1：INTCON、PIR1 寄存器中的若干位将受到影响（而产生唤醒）。

2：当器件被中断唤醒且全局允许位 GIE 置 1 时，在执行 PC+1 指令后， PC 指针将指向中断向量 0004h。

3：表 3-3 给出了具体条件下的复位值。

表 3-4: 特殊功能寄存器的初始化条件 (续)

寄存器	上电复位 欠压复位	– 正常工作 – 休眠或 – WDT 复位时 的 MCLR 复位	– 中断 – WDT 溢出 从休眠状态唤醒
PORTA	--xx xxxx	--uu uuuu	--uu uuuu
PORTB	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTC	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTD	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTE	---- -xxx	---- -uuu	---- -uuu
PORTF	0000 0000	0000 0000	uuuu uuuu
PORTG	0000 0000	0000 0000	uuuu uuuu
PR2	1111 1111	1111 1111	1111 1111
PREFA	0000 0000	0000 0000	uuuu uuuu
PREFB	0000 0000	0000 0000	uuuu uuuu
RCSTA	0000 -00x	0000 -00x	uuuu -uuu
RCREG	0000 0000	0000 0000	uuuu uuuu
SLPCON	0011 1111	0011 1111	uuuu uuuu
SPBRG	0000 0000	0000 0000	uuuu uuuu
SSPBUF	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSPCON	0000 0000	0000 0000	uuuu uuuu
SSPADD	0000 0000	0000 0000	uuuu uuuu
SSPSTAT	0000 0000	0000 0000	uuuu uuuu
STATUS	0001 1xxx	000q quuu (3)	uuuq quuu (3)
T1CON	--00 0000	--uu uuuu	--uu uuuu
T2CON	-000 0000	-000 0000	-uuu uuuu
TMR0	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR2	0000 0000	0000 0000	uuuu uuuu
TRIS	--11 1111	--11 1111	--uu uuuu
TRISA	--11 1111	--11 1111	--uu uuuu
TRISB	1111 1111	1111 1111	uuuu uuuu
TRISC	1111 1111	1111 1111	uuuu uuuu
TRISD	1111 1111	1111 1111	uuuu uuuu
TRISE	0000 -111	0000 -111	uuuu -uuu
TRISF	1111 1111	1111 1111	uuuu uuuu
TRISG	1111 1111	1111 1111	uuuu uuuu
TXREG	0000 0000	0000 0000	uuuu uuuu
TXSTA	0000 -010	0000 -010	uuuu -uuu
VRCON	000- 0000	000- 0000	uuu- uuuu
W	xxxx xxxx	uuuu uuuu	uuuu uuuu

图注： u = 未改变， x = 未知， - = 未用，读为 ‘0’， q = 根据条件而变化

注 1：INTCON、 PIR1 寄存器中的若干位将受到影响 （而产生唤醒）。

2：当器件被中断唤醒且全局允许位 GIE 置 1 时，在执行 PC+1 指令后， PC 指针将指向中断向量 0004h。

3：表 3-3 给出了具体条件下的复位值。

3.3.1 电源控制寄存器 （PCON）和状态寄存器 （STATUS）

电源控制寄存器 （PCON）中有一个状态位可用于区分上电复位 (BOR)、外部 $\overline{MCLR}$ 复位或是看门狗定时器复位。它还包含另一个状态位，用于判断是否发生了欠压复位 (BOR)。电源控制 / 状态寄存器 PCON 中共有四个位。

上电复位时， $\overline{BOR}$（ 欠压复位 ） 的状态位不确定。用户应先将该位置位，并在复位后查看是否 BOR = 0，即发生了欠压复位。BOR 状态位是一个“无关位”，而且也不一定能够预测欠压电路是否被禁止（将配置字的 BODEN 位清零）。

上电复位时， $\overline{POR}$（上电复位）状态位被清零，而其他复位不影响该位。上电复位后，用户应将该位置 1。此后，当发生复位时，若 $\overline{POR}$ 是 0，则表示发生了一次上电复位。

奇偶校验错误复位时， $\overline{PER}$ (奇偶校验错误复位) 状态位被清零，用户应该用软件将该位置 1。上电复位后，也应将该位置 1。

MPEEN（存储器 奇偶校验错误使能）位反映出配置字的 MPEEN 位的设置，任何复位和中断都不会影响该位的值。

注：	上电复位时， $\overline{BOR}$ 位的状态是未知的，用户必须将其置 1。随后发生复位时，若该位被清零，则表示发生过欠压复位。 $\overline{BOR}$ 状态位是一个“无关位”，而且也不一定能够预测欠压电路是否被禁止（将配置字的 BODEN 位清零）。

寄存器 3-1: PCON 寄存器

R-u	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
MPEEN	—	—	—	—	$\overline{PER}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

bit 7 **MPEEN**：存储器奇偶校验错误位
该位反映了 MPEEN 配置位的值

bit 6:3 **未用位**：读作 '0'

bit 2 **$\overline{PER}$**：存储器奇偶校验错误复位的状态位
1 = 没有发生奇偶校验错误复位
0 = 发生过程序存储器取奇偶校验错误
（应在上电复位或奇偶校验误差复位后，用软件置 1 ）

bit 1 **$\overline{POR}$**：上电复位的状态位
1 = 没有发生上电复位
0 = 发生过上电复位（应在上电复位发生之后用软件置 1）

bit 0 **$\overline{BOR}$**：欠压复位的状态位
1 = 没有发生欠压复位
0 = 发生欠压复位（应在上电复位或欠压复位后，用软件置 1）

图注		
R = 可读位	W = 可写位	u = 未改变
U = 未用，读作 0		- n = 上电复位 （POR）时的值

注：	不一定要用到所有的位。

状态寄存器 STATUS 中含有两个位 （$\overline{TO}$ 和 $\overline{PD}$），当与 PCON 寄存器联用时，可为用户提供足够的信息用以确定复位的原因。

寄存器 3-2: STATUS 寄存器

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP**：寄存器组选择位 （用于间接寻址）
1 = 第 2， 3 组 （100h - 1FFh）
0 = 第 0， 1 组 （00h - FFh）
对于只有 0 和 1 寄存器组的器件， IRP 位是保留位，其值始终为 0。

bit 6:5 **RP1:RP0**：寄存器组选择位 （用于直接寻址）
11 = 第 3 组 (180h - 1FFh)
10 = 第 2 组 （100h - 17Fh）
01 = 第 1 组 （80h - FFh）
00 = 第 0 组 （00h - 7Fh）
每组有 128 字节。对于只有 0 和 1 寄存器组的器件， IRP 位是保留位，其值始终为 0。

bit 4 $\overline{TO}$：延时溢出位
1 = 上电，执行 CLRWDT 指令，或 SLEEP 指令
0 = 发生看门狗定时器 （WDT）超时溢出

bit 3 $\overline{PD}$：掉电位
1 = 上电或是执行 CLRWDT 指令
0 = 执行了 SLEEP 指令

bit 2 **Z**：零位
1 = 运算或逻辑运算的结果是零
0 = 运算或逻辑结果的结果非零

bit 1 **DC**：辅助进位 / $\overline{借位}$位 （ADDWF， ADDLW， SUBLW， SUBWF 指令）（辅助$\overline{借位}$的极性相反）
1 = 有第 3 位向第 4 位进位或无第 3 位向第 4 位借位
0 = 无第 3 位向第 4 位进位或有第 3 位向第 4 位借位

bit 0 **C**：进位 / $\overline{借位}$位 （ADDWF， ADDLW，SUBLW，SUBWF 指令）
1 = 有进位或无借位
0 = 无进位或有借位

注： 借位时极性相反。执行减法是通过加上第二个操作数的补码来完成的。对于移位指令 （RRF， RLF），源寄存器的最高位或最低位移入此位 .

图注		
R = 可读	W = 可写	
U = 未用，读作 0		- n = 上电复位 （POR）时的值

3.4 设计技巧

问 1: ***我的系统处于 ESD （静电放电）和 EMI （电磁干扰）环境下时运行出错。***

如果所使用的器件没有针对片内主复位电路（附录 C）的毛刺滤波器，应在 MCLR 引脚上外接适当的滤波器来滤除小的尖脉冲干扰。电气规范参数 35 规定了的可引起复位的所需脉冲宽度。

问 2: ***使用 JW （窗口）型器件时，我的系统复位和工作都很正常。但使用 OTP （一次性编程）器件，我的系统工作就不正常了。***

答 2:

最常见原因是没有将窗口型器件 （JW）的窗口覆盖住，其背景光使器件上电后的状态与没有光线时不同。在大多数情况下，通用 RAM 和特殊功能寄存器未正确进行初始化。

3.5 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与复位相关的应用笔记有：

标题	应用笔记 #
Power-up Trouble Shooting	AN607
Power-up Considerations	AN522

3.6 版本历史

版本 A

这是描述复位的初始发行版。

第 4 章 架构

目录

本章包括下面一些主要内容：

4.1 简介 4-2
4.2 时序图 / 指令周期 4-5
4.3 指令流 / 流水线 4-6
4.4 I/O 端口描述 4-7
4.5 设计技巧 4-12
4.6 相关应用笔记 4-13
4.7 版本历史 4-14

4.1 简介

高性能的 PICmicro 单片机具有许多 RISC 微处理器的架构特点，如下所示：

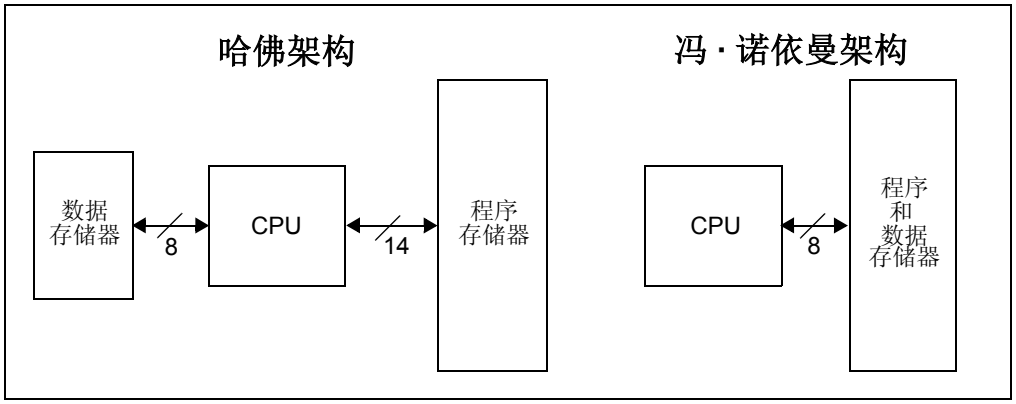
- 哈佛架构
- 长字指令
- 单字指令
- 单周期指令
- 指令流水线
- 精简指令集
- 文件寄存器结构
- 正交 （对称）指令集

图 4-2 所示为中档系列单片机的简化总线结构图。

哈佛架构：

哈佛架构有独立的程序存储器和数据存储器，并可通过各自的独立总线进行存取。与冯 · 诺依曼架构相比，哈佛架构有更宽的数据带宽，因为冯 · 诺依曼架构的程序存储器和数据存储器是合二为一的，且通过同一总线访问。当执行一条指令时，冯 · 诺依曼架构的处理器通常需要通过 8 位总线进行多次操作，才能取得指令，同时也需要对数据进行读取、处理以及写操作，可见数据总线的操作非常繁忙。但哈佛架构则不同，取指令在单周期内完成（中档系列的指令为 14 位），在访问程序存储器中的同时，可通过独立的总线对数据存储器进行读写操作。独立的总线使得在执行一条指令的同时，可以取下一条指令。冯 · 诺依曼架构和哈佛架构的对比，见图 4-1。

图 4-1： 哈佛架构与冯 · 诺依曼架构的对比



长字指令：

长字指令的指令总线宽度比 8 位数据存储器的数据总线宽 （位数更多）。由于使用了相互独立的总线，故能做到这一点。由于指令总线宽度可以不同于 8 位数据宽度，在针对架构对程序存储器做优化后，程序存储器的使用效率会更高。

单字指令：

单字指令为 14 位宽，因此所有指令都可以是单字指令。通过 14 位宽的程序存储器总线可以在单周期内取一条 14 位的指令。由于是单字指令，所以单片机程序存储器的大小 （字数）就等于指令数，这意味着在所有存储单元内都是合法指令。

在典型的冯 · 诺依曼架构中，大多数指令为多字节指令，在 4K 字节的程序存储器中一般只能存放大约 2K 条指令。这 2:1 的比率只是大概情况，实际上与应用代码有关。由于有多字节指令，所以不能保证每个存储单元内都是合法指令。

指令流水线：

指令流水线有两级流水线，可以使取指操作和指令执行重叠进行。取指花费一个 TCY 时间，而该指令在下一个 TCY 时间内执行。由于当前指令的取指操作和前一条指令的执行是重叠的，所以在每一个 TCY 内，进行一条指令的取指和另一条指令的执行。

单周期指令：

程序存储器总线是 14 位宽，因此能在一个机器周期 (TCY) 内完成整条指令的取指操作。指令中包含了所需的所有信息，并能在单周期内执行完毕。如果指令的执行结果要修改程序指针 PC，那么完成指令可能需要 2 个周期 （有一个周期的延迟），因为此时流水线会作废一条指令并重新取指令。

精简指令集：

当指令集设计得很好而且高度正交 （对称）时，完成所有工作只需很少的指令。指令少，就很容易掌握整个指令集。

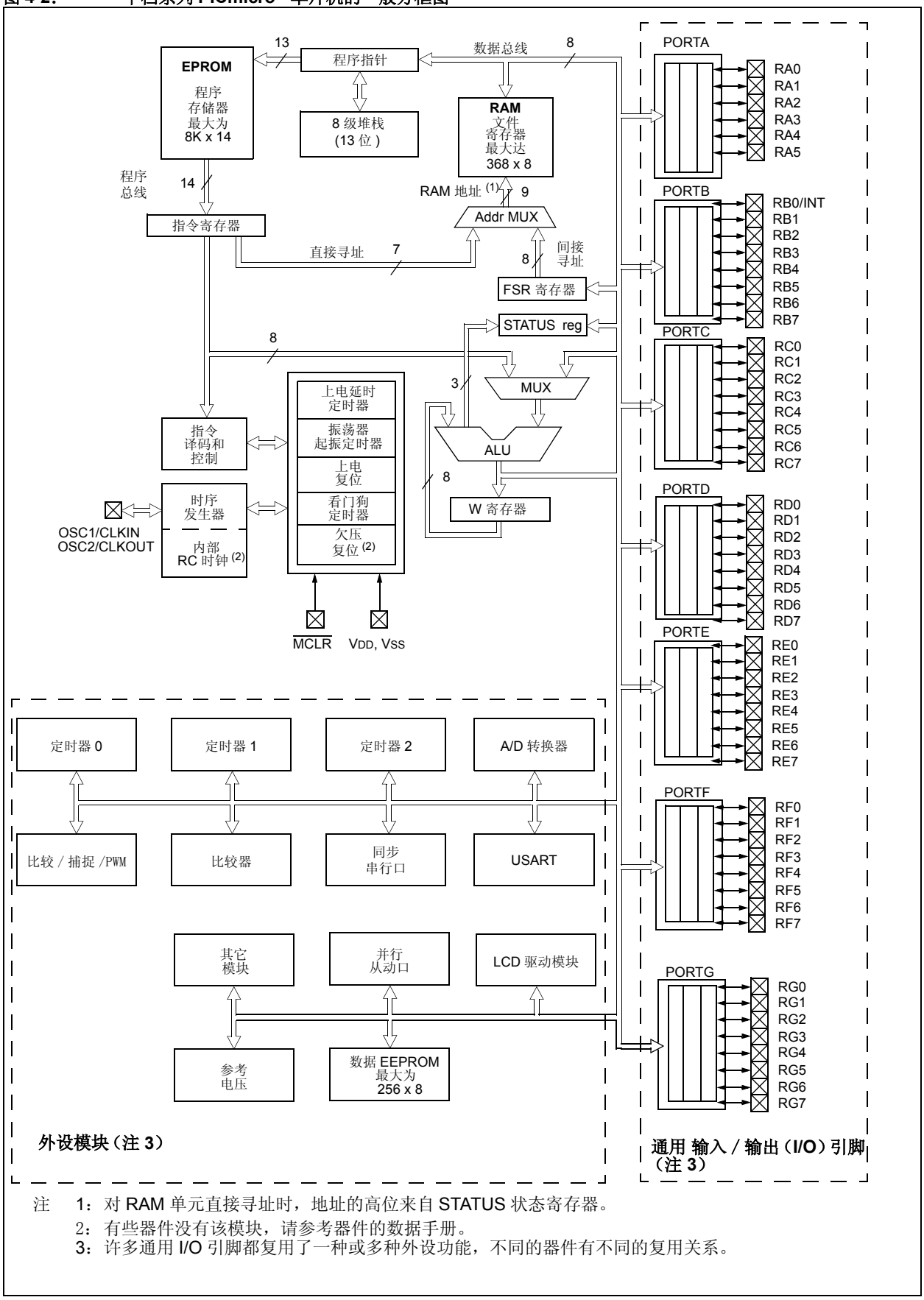
文件寄存器结构：

文件寄存器 / 数据存储器可以通过直接或间接寻址方式来访问。所有特殊功能寄存器，包括程序指针 PC，都映射到数据存储器。

正交 （对称）的指令集：

正交指令有助于实现：对任意寄存器，采用所有的寻址方式完成所有可能的操作。指令集的对称性以及无特殊指令会使编程更简单和有效。此外，所有这些都有助于大大缩短指令的学习周期。中档系列单片机的指令集里，只有两条与寄存器操作无关的指令，它们用于两种与内核相关的操作上。其中一条为 SLEEP 指令，可将器件设置为最低功耗模式。另一条是 CLRWDT 指令，该指令对片内看门狗定时器 (WDT) 清零以防止溢出复位，以检验器件是否正常工作。

图 4-2： 中档系列 PICmicro® 单片机的一般方框图

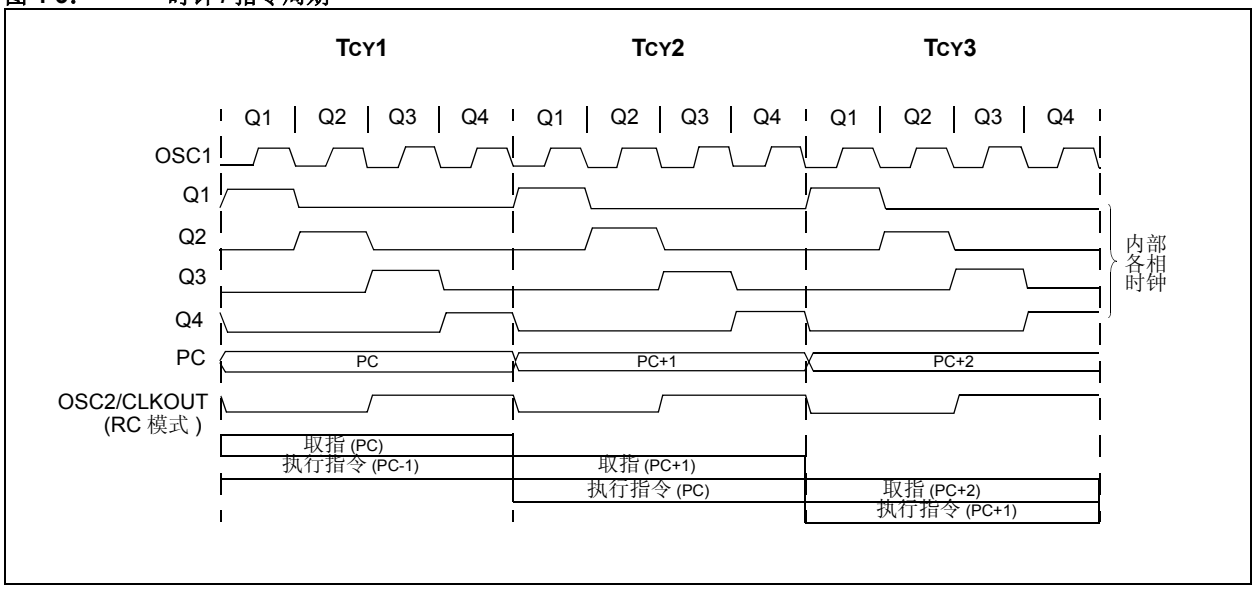


注 1：对 RAM 单元直接寻址时，地址的高位来自 STATUS 状态寄存器。
2：有些器件没有该模块，请参考器件的数据手册。
3：许多通用 I/O 引脚都复用了一种或多种外设功能，不同的器件有不同的复用关系。

4.2 时序图 / 指令周期

由 OSC1 引脚输入的时钟信号，在器件内部经过 4 分频后，产生非重叠的 4 个正交时钟节拍，分别命名为 Q1, Q2, Q3 和 Q4。在每个 Q1 拍，程序指针 PC 加 1；在 Q4 拍，从程序存储器取指，并将指令锁存在指令寄存器中。指令的译码和执行是在下一个 Q1 到 Q4 拍之间完成。 图 4-3 和 例 4-1 所示为时钟和指令流的关系。

图 4-3： 时钟 / 指令周期



4.3 指令流 / 流水线

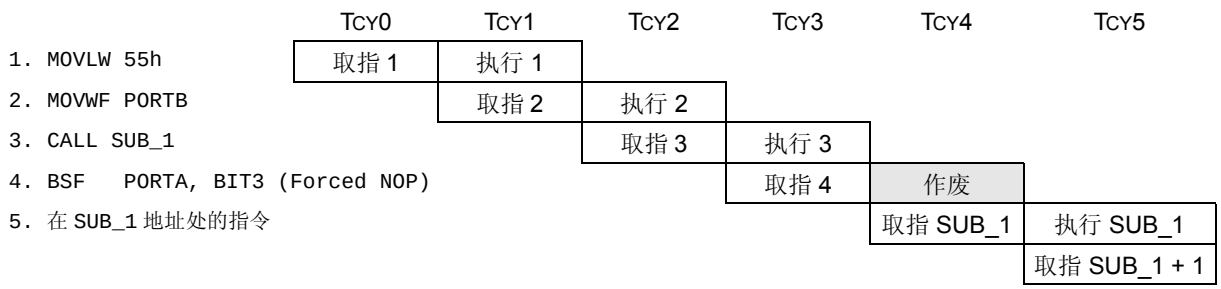
一个指令周期由 4 个 Q 周期组成 (即 Q1, Q2, Q3 和 Q4)。取指需要一个指令周期，而该指令在下一个指令周期内执行。由于流水线的原因，每条指令的等效执行时间为一个指令周期。如果指令改变了程序指针 PC （如 GOTO 指令），则需要多一个指令周期才能完成指令 （见例 4-1）。

在 Q1 周期，开始**取指**操作，程序指针加 1。

指令的**执行**过程：在 Q1 周期中，所取指令被锁存到指令寄存器 (IR) 中。在 Q2、Q3 和 Q4 周期中进行指令的译码和执行，其中读数据存储器 （读操作数）发生在 Q2 周期，写操作发生在 Q4 周期。

例 4-1 示出了对于给出的指令序列两级流水线如何工作。在 TCY0 指令周期，从程序存储器中取第一条指令；在 TCY1 指令周期，执行第一条指令，并取第二条指令；在 TCY2 指令周期，执行第二条指令，并取第三条指令；在 TCY3 指令周期，执行第三条指令 (CALL SUB_1)，并取第四条指令； 在执行完第三条指令后， CPU 将第四条指令的地址压入堆栈，并将程序指针 PC 改变为 SUB_1 的地址，这意味着将作废在 TCY3 指令周期所取的指令。在 TCY4 指令周期，指令 4 作废，将执行一条空操作指令 （NOP），并对 SUB_1 地址单元进行取指。在 TCY5 指令周期，执行第五条指令，并对 SUB_1+ 1 地址单元进行取指。

例 4-1： 指令流水线



除程序转移指令外，所有的指令都是单周期指令；由于程序转移指令将导致流水线的一条取指作废，然后重新取指和执行指令，所以程序转移指令需要两个周期。

4.4 I/O 端口描述

表 4-1 概述了端口引脚可以复用的功能。有些引脚有多种功能。当发生功能复用时，外设模块的功能要求可能使引脚的数据方向设置（TRIS 位）不再起作用（例如用于 A/D 或 LCD 模块时）。

表 4-1： I/O 端口描述

引脚名称	引脚类型	缓冲器类型	描述
			模拟输入通道
AN0	I	模拟	
AN1	I	模拟	
AN2	I	模拟	
AN3	I	模拟	
AN4	I	模拟	
AN5	I	模拟	
AN6	I	模拟	
AN7	I	模拟	
AN8	I	模拟	
AN9	I	模拟	
AN10	I	模拟	
AN11	I	模拟	
AN12	I	模拟	
AN13	I	模拟	
AN14	I	模拟	
AN15	I	模拟	
AVDD	P	P	模拟电源
AVSS	P	P	模拟地
C1	I	模拟	LCD 电压产生
C2	I	模拟	LCD 电压产生
CCP1	I/O	ST	捕捉 1 输入 / 比较器 1 输出 /PWM1 输出
CCP2	I/O	ST	捕捉 2 输入 / 比较器 2 输出 /PWM2 输出
CDAC	O	模拟	A/D 模块的斜坡电流源输出。一般外接电容，以产生线性的斜坡电压。
CK	I/O	ST	通用同步异步收发器的异步时钟，与 TX 引脚复用 (参见有关 TX、RX 和 DT 引脚的信息)。
CLKIN	I	ST/CMOS	外部时钟源输入，与 OSC1 引脚复用 (参见有关 OSC1/CLKIN、OSC2/CLKOUT 引脚的信息)。
CLKOUT	O	—	振荡器的输出。在晶体振荡器模式时，该引脚与晶体或谐振器相连；在 RC 振荡模式时，从 OSC2 引脚输出振荡信号 CLKOUT，该信号是 OSC1 引脚上的振荡信号的 4 分频，等于指令周期。该功能与 OSC2 引脚复用 （参见有关 OSC2 和 OSC1 引脚的信息）
CMPA	O	—	比较器 A 的输出
CMPB	O	—	比较器 B 的输出

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
SM = SMBus 兼容的输入。作为输出端口时，要外接上拉电阻
NPU = N 沟道上拉　　PU = 内部弱上拉
No-P diode = 没有二极管接到 VDD　　AN = 模拟输入或输出
I = 输入　　O = 输出
P = 电源　　L = LCD 驱动

表 4-1： I/O 端口描述 （续）

引脚名称	引脚类型	缓冲器类型	描述
COM0	L	—	LCD 模块的公共驱动端 0
COM1	L	—	LCD 模块的公共驱动端 1
COM2	L	—	LCD 模块的公共驱动端 2
COM3	L	—	LCD 模块的公共驱动端 3
$\overline{CS}$	I	TTL	并行从动端口的片选控制 （参见有关 $\overline{RD}$ 和 $\overline{WR}$ 的信息）。
DT	I/O	ST	通用同步异步收发器的数据口，与 RX 引脚复用 （参见有关 TX、 RX 和 DT 引脚的信息）。
			GP 是双向 I/O 端口。 GP 口的某些引脚可以通过软件编程为内部弱上拉的输入端。
GP0	I/O	TTL/ST	作为通用 I/O 端口时，为 TTL 输入缓冲；在串行编程模式时，为施密特触发器输入缓冲。
GP1	I/O	TTL/ST	作为通用 I/O 端口时，为 TTL 输入缓冲；在串行编程模式时，为施密特触发器输入缓冲。
GP2	I/O	ST	
GP3	I	TTL	
GP4	I/O	TTL	
GP5	I/O	TTL	
INT	I	ST	外部中断
$\overline{MCLR}$/VPP	I/P	ST	主复位输入，或编程电压输入。此引脚为低电平时，单片机复位 （低电平有效）。
NC	—	—	这些引脚应该悬空。
OSC1	I	ST/CMOS	晶振输入或外部时钟源输入。配置为 RC 模式时，该脚带 ST 缓冲，其它模式为 CMOS 缓冲。
OSC2	O	—	晶振输出。在晶振模式时，该引脚与晶体或谐振器相连；在 RC 模式时，从 OSC2 引脚输出振荡信号 CLKOUT，该信号是 OSC1 引脚上振荡信号的 4 分频，等于指令周期。
PBTN	I	ST	带弱上拉电阻的输入口，可用于产生中断。
PSP0	I/O	TTL	并行从动端口，可以和其它微处理器端口相连接。当使能并行从动端口 （PSP）模块时，这些引脚带 TTL 输入缓冲。
PSP1	I/O	TTL	
PSP2	I/O	TTL	
PSP3	I/O	TTL	
PSP4	I/O	TTL	
PSP5	I/O	TTL	
PSP6	I/O	TTL	
PSP7	I/O	TTL	
			PORTA 是双向 I/O 端口
RA0	I/O	TTL	
RA1	I/O	TTL	
RA2	I/O	TTL	
RA3	I/O	TTL	
RA4	I/O	ST	当 RA4 引脚配置输出时，为集电极开路。
RA5	I/O	TTL	

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
SM = SMBus 兼容的输入。作为输出端口时，要外接上拉电阻
NPU = N 沟道上拉　　PU = 内部弱上拉
No-P diode = 没有二极管接到 VDD　　AN = 模拟输入或输出
I = 输入　　O = 输出
P = 电源　　L = LCD 驱动

表 4-1： I/O 端口描述 （续）

引脚名称	引脚类型	缓冲器类型	描述
			PORTB是双向I/O端口。PORTB端口可以通过软件编程为所有引脚都是带内部弱上拉的输入端。
RB0	I/O	TTL	
RB1	I/O	TTL	
RB2	I/O	TTL	
RB3	I/O	TTL	
RB4	I/O	TTL	电平变化中断引脚
RB5	I/O	TTL	电平变化中断引脚
RB6	I/O	TTL/ST	电平变化中断引脚。串行编程的时钟线。作为通用 I/O 口时，带 TTL 输入缓冲；作为串行编程的时钟时，带施密特触发器输入缓冲。
RB7	I/O	TTL/ST	电平变化中断引脚。串行编程的数据线。作为通用 I/O 口时，带 TTL 输入缓冲；作为串行编程的时钟时，带施密特触发器输入缓冲。
			PORTC 是双向 I/O 端口。
RC0	I/O	ST	
RC1	I/O	ST	
RC2	I/O	ST	
RC3	I/O	ST	
RC4	I/O	ST	
RC5	I/O	ST	
RC6	I/O	ST	
RC7	I/O	ST	
$\overline{RD}$	I	TTL	并行从动端口的读控制引脚 (参见有关 $\overline{WR}$ 和 $\overline{CS}$ 引脚的信息)。
			PORTD 是双向 I/O 端口。
RD0	I/O	ST	
RD1	I/O	ST	
RD2	I/O	ST	
RD3	I/O	ST	
RD4	I/O	ST	
RD5	I/O	ST	
RD6	I/O	ST	
RD7	I/O	ST	
			PORTE 是双向 I/O 端口。
RE0	I/O	ST	
RE1	I/O	ST	
RE2	I/O	ST	
RE3	I/O	ST	
RE4	I/O	ST	
RE5	I/O	ST	
RE6	I/O	ST	
RE7	I/O	ST	

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
SM = SMBus 兼容的输入。作为输出端口时，要外接上拉电阻
NPU = N 沟道上拉　　PU = 内部弱上拉
No-P diode = 没有二极管接到 V_{DD}　　AN = 模拟输入或输出
I = 输入　　O = 输出
P = 电源　　L = LCD 驱动

表 4-1： I/O 端口描述 （续）

引脚名称	引脚类型	缓冲器类型	描述
REFA	O	CMOS	可编程参考电压模块 A 的输出 .
REFB	O	CMOS	可编程参考电压模块 B 的输出 .
			PORTF 是数字输入端口或 LCD 的段驱动端口。
RF0	I/O	ST	
RF1	I/O	ST	
RF2	I/O	ST	
RF3	I/O	ST	
RF4	I/O	ST	
RF5	I/O	ST	
RF6	I/O	ST	
RF7	I/O	ST	
			PORTG 是数字输入端口或 LCD 的段驱动端口。
RG0	I/O	ST	
RG1	I/O	ST	
RG2	I/O	ST	
RG3	I/O	ST	
RG4	I/O	ST	
RG5	I/O	ST	
RG6	I/O	ST	
RG7	I/O	ST	
RX	I	ST	工作在异步模式时，通用同步异步收发器的数据接收引脚。
SCL	I/O	ST	I^2C^{TM} 模式时，同步串行时钟的输入 / 输出引脚。
SCLA	I/O	ST	I^2C 接口的同步串行时钟引脚。
SCLB	I/O	ST	I^2C 接口的同步串行时钟引脚。
SDA	I/O	ST	I^2C 的数据 I/O 引脚
SDAA	I/O	ST	I^2C 接口的同步串行数据 I/O 引脚
SDAB	I/O	ST	I^2C 接口的同步串行数据 I/O 引脚
SCK	I/O	ST	SPI^{TM} 模式时，同步串行时钟的输入 / 输出引脚。
SDI	I	ST	SPI 模式的数据输入
SDO	O	—	SPI 模式的数据输出
$\overline{SS}$	I	ST	SPI 同步从模式选择
SEG00 至 SEG31	I/L	ST	LCD 的段驱动 00 至段驱动 31。
SUM	O	AN	AN1 的求和输出。该引脚可外接电容，可对窄脉冲进行平滑。
T0CKI	I	ST	定时器 0 的外部时钟输入引脚
T1CKI	I	ST	定时器 1 的外部时钟输入引脚
T1OSO	O	CMOS	定时器 1 的振荡器输出
T1OSI	I	CMOS	定时器 1 的振荡器输入
TX	O	—	工作在异步模式时，通用同步异步收发器的数据发送引脚 (参见有关 RX 引脚的信息)。

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
SM = SMBus 兼容的输入。作为输出端口时，要外接上拉电阻
NPU = N 沟道上拉　　PU = 内部弱上拉
No-P diode = 没有二极管接到 V_{DD}　　AN = 模拟输入或输出
I = 输入　　O = 输出
P = 电源　　L = LCD 驱动

I^2C 是 Philips 公司的商标。

表 4-1： I/O 端口描述 （续）

引脚名称	引脚类型	缓冲器类型	描述
VLCD1	P	—	LCD 电压
VLCD2	P	—	LCD 电压
VLCD3	P	—	LCD 电压
VLCDADJ	I	模拟	LCD 电压产生
VREF	I	模拟	模拟高参考电压输入 带有比较器的单片机的 DR 参考电压输出。
VREF+	I	模拟	模拟高参考电压输入。 一般与一个模拟引脚复用。
VREF-	I	模拟	模拟低参考电压输入。 一般与一个模拟引脚复用。
VREG	O	—	该引脚为输出引脚，控制外部 N 沟道场效应管的栅极来达到稳压作用。
VSS	P	—	数字逻辑和 I/O 引脚的参考地
VDD	P	—	数字逻辑和 I/O 引脚的正电源
$\overline{WR}$	I	TTL	并行从动端口的写控制引脚 (参见 $\overline{RD}$ 和 $\overline{CS}$ 引脚)

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
SM = SMBus 兼容的输入。作为输出端口时，要外接上拉电阻
NPU = N 沟道上拉　　PU = 内部弱上拉
No-P diode = 没有二极管接到 VDD　　AN = 模拟输入或输出
I = 输入　　O = 输出
P = 电源　　L = LCD 驱动

4.5 设计技巧

目前没有相关的设计技巧。

4.6 相关应用笔记

本节列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一些限制即可使用。目前与架构相关的应用笔记有：

标题	应用笔记 #

目前没有相关的应用笔记。

4.7 版本历史

版本 A

这是描述 PICmicro 单片机架构的初始发行版。

第 5 章 CPU 和 ALU

目录

本章包括下面一些主要内容：

5.1 简介 5-2
5.2 指令的一般格式 5-4
5.3 中央处理单元 （CPU） 5-4
5.4 指令时钟 5-4
5.5 算术逻辑单元 （ALU） 5-5
5.6 状态寄存器 5-6
5.7 OPTION_REG 寄存器 5-8
5.8 电源控制寄存器 5-9
5.9 设计技巧 5-10
5.10 相关应用笔记 5-11
5.11 版本历史 5-12

5.1 简介

中央处理单元 （CPU）通过执行程序存储器中的信息 （指令）来控制器件的运行。其中许多指令是对数据存储器进行操作。对数据存储器的操作需要使用算术逻辑单元 （ALU）。除了执行算术和逻辑操作外，ALU 还控制状态位（在状态寄存器中）。一些指令的执行结果会根据结果的状态而改变状态位。

CPU 的机器码如表 5-1 中所示 （MPASM™ 用来产生这些机器码的指令助记符也列在表中）。

表 5-1:　　中档系列单片机指令集

助记符，操作数		描　述	周期	14 位宽指令字				影响的状态位	备注
				最高位			最低位		
针对字节的数据寄存器操作									
ADDWF	f, d	W 加 f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	f, d	W 和 f 与运算	1	00	0101	dfff	ffff	Z	1,2
CLRF	f	f 清零	1	00	0001	lfff	ffff	Z	2
CLRW	-	W 清零	1	00	0001	0xxx	xxxx	Z	
COMF	f, d	f 取反	1	00	1001	dfff	ffff	Z	1,2
DECF	f, d	f 减 1	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	f, d	f 减 1，为 0 则跳过	1(2)	00	1011	dfff	ffff		1,2,3
INCF	f, d	f 增 1	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	f, d	f 增 1，为 0 则跳过	1(2)	00	1111	dfff	ffff		1,2,3
IORWF	f, d	W 和 f 或运算	1	00	0100	dfff	ffff	Z	1,2
MOVF	f, d	f 送到 d	1	00	1000	dfff	ffff	Z	1,2
MOVWF	f	W 送到 f	1	00	0000	lfff	ffff		
NOP	-	空操作	1	00	0000	0xx0	0000		
RLF	f, d	f 循环左移	1	00	1101	dfff	ffff	C	1,2
RRF	f, d	f 循环右移	1	00	1100	dfff	ffff	C	1,2
SUBWF	f, d	f 减去 W	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	f, d	f 半字节交换	1	00	1110	dfff	ffff		1,2
XORWF	f, d	W 和 f 异或运算	1	00	0110	dfff	ffff	Z	1,2
针对位的数据寄存器操作									
BCF	f, b	清除 f 的 bit b	1	01	00bb	bfff	ffff		1,2
BSF	f, b	置 f 的 bit b	1	01	01bb	bfff	ffff		1,2
BTFSC	f, b	测试 f 的 bit b，为 0 则跳过	1 (2)	01	10bb	bfff-	ffff		3
BTFSS	f, b	测试 f 的 bit b，为 1 则跳过	1 (2)	01	11bb	bfff	ffff		3
立即数操作和控制操作									
ADDLW	k	立即数加 W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	k	立即数和 W 与运算	1	11	1001	kkkk	kkkk	Z	
CALL	k	调用子程序	2	10	0kkk	kkkk	kkkk		
CLRWDT	-	清除看门狗定时器	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	跳转	2	10	1kkk	kkkk	kkkk		
IORLW	k	立即数和 W 或运算	1	11	1000	kkkk	kkkk	Z	
MOVLW	k	立即数送到 W	1	11	00xx	kkkk	kkkk		
RETFIE	-	中断返回	2	00	0000	0000	1001		
RETLW	k	立即数送到 W 的子程序返回	2	11	01xx	kkkk	kkkk		
RETURN	-	子程序返回	2	00	0000	0000	1000		
SLEEP	-	进入休眠状态	1	00	0000	0110	0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	立即数减 W	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	k	立即数和 W 异或运算	1	11	1010	kkkk	kkkk	Z	

注　1：当 I/O 寄存器用自身内容修改自己时 （如 MOVF PORTB， 1），使用的值是引脚上的当前值。例如，某个设置成输入的引脚，其数据锁存器中的值为 “1”，但此时被外部器件拉为低电平，执行指令后，数据锁存器的值变为 “0”。

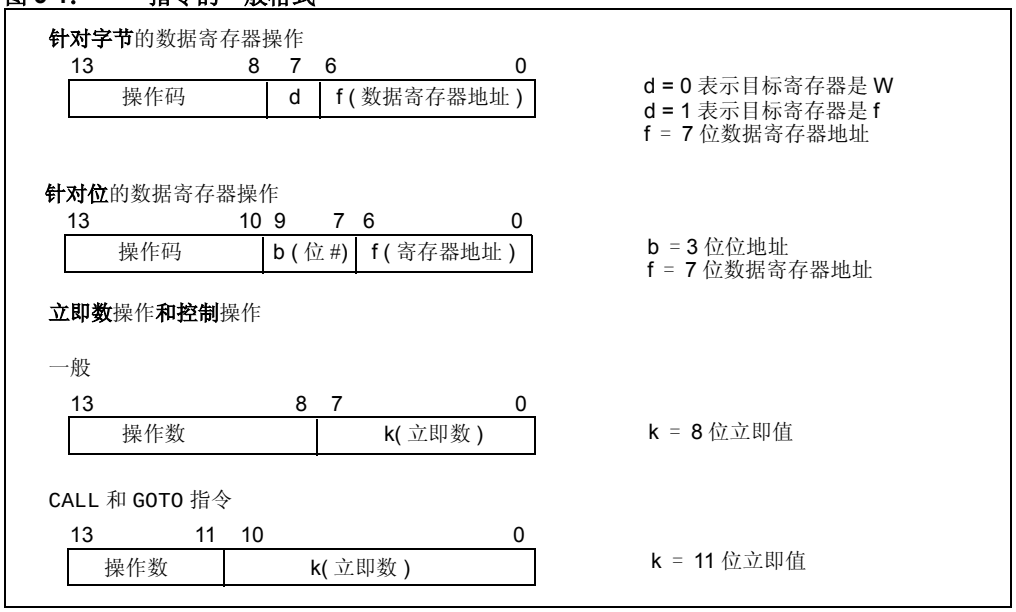
2：如果预分频器分配给 Timer0 模块，对 TMR0 寄存器执行该指令 （且适用时 d=1）时，将清零预分频器。

3：改写程序计数器 （PC）或条件测试为真时，需要 2 个周期执行指令，第二个周期执行空操作指令 NOP。

5.2 指令的一般格式

中档系列单片机的指令有四种一般格式，如图 5-1 所示。指令的操作码从 3 位到 6 位不等。这种可变长度的操作码组成了 35 条指令。

图 5-1： 指令的一般格式



5.3 中央处理单元 （CPU）

CPU 被视为器件的 “大脑”，它负责获取正确的执行指令、译码并且执行该指令。

CPU 有时和 ALU 配合工作来完成指令的执行 （如算术或逻辑操作）。

CPU 控制程序存储器的地址总线、数据存储器的地址总线以及对堆栈的访问。

5.4 指令时钟

每个指令周期 （TCY）由 4 个时钟节拍 （Q1-Q4）构成。时钟节拍和器件振荡周期 （Tosc）相同。在各个时钟节拍分别对指令的译码、读取、处理和写操作等进行计时 / 标示。下图表明了时钟节拍和指令周期之间的关系。

组成指令周期 （Tcy）的 4 个时钟节拍归纳如下：

Q1： 指令的译码周期或强制性空操作

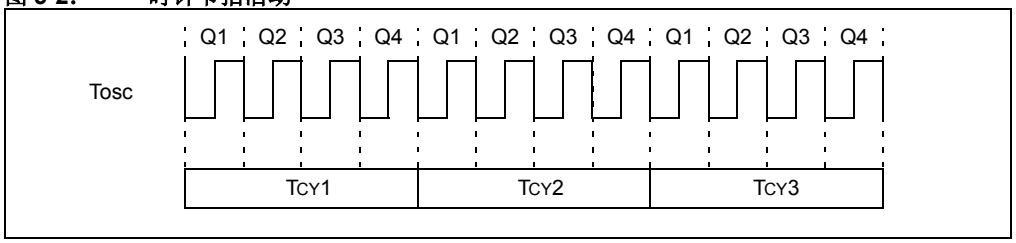
Q2： 指令的读数据周期或空操作

Q3： 处理数据

Q4： 指令的写数据周期或空操作

每条指令都有具体的时钟节拍操作。

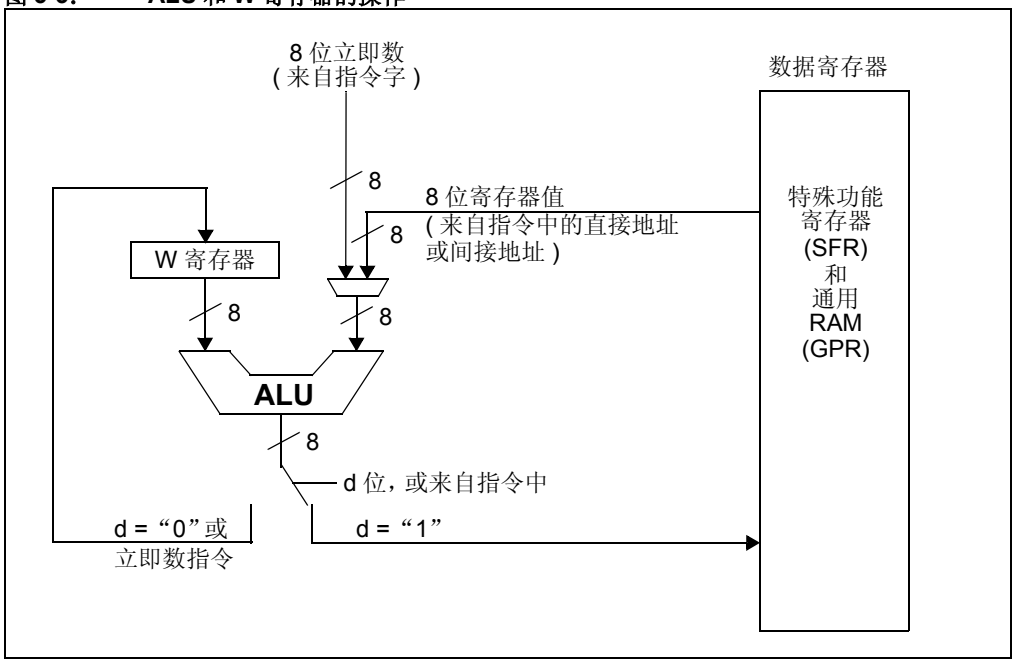
图 5-2： 时钟节拍活动



5.5 算术逻辑单元 （ALU）

PICmicro® 单片机包含一个 8 位 ALU 和一个 8 位工作寄存器。ALU 是一个通用的算术逻辑单元，它对工作寄存器和数据寄存器中的数据进行算术和布尔运算。

图 5-3： ALU 和 W 寄存器的操作



ALU 是 8 位宽，能够进行加、减、移位和逻辑操作。除非特别指明，算术运算一般是以二进制补码形式进行。在 2 个操作数的指令中，典型情况下，其中一个操作数是在工作寄存器 （W 寄存器）中，另一个操作数放在一个数据寄存器中或是一个立即数。在单操作数指令中，操作数放在 W 寄存器中或某个数据寄存器中。

W 寄存器是一个 8 位宽、用于 ALU 运算的工作寄存器，它是一个不可寻址的寄存器。

根据所执行的指令，ALU 可以影响状态寄存器中的进位标志位 C、辅助进位标志位 DC 和全零标志位 Z。在减法操作中，C 和 DC 位就分别作为借位和辅助借位之反。例如指令 SUBLW 和 SUBWF。

5.6 状态寄存器

状态寄存器（如图 5-1 所示）含有 ALU 的算术运算结果状态、复位状态及数据存储区的选择位。因为数据存储区的选择是由状态寄存器控制的，所以各存储区里都有状态寄存器的映射。而且，这些映射在每个存储区的相对位置（偏移位置）都相同（见**“存储器构成”一章**中**图 6-5：“寄存器映射”**）。

状态寄存器和其它寄存器一样，可以作为任何指令的目标寄存器。如果状态寄存器作为一条指令的目标寄存器，而这条指令又影响 Z、 DC 或 C 标志位，那么这三个标志位的状态不能由指令直接写入。这些标志位的状态要根据器件逻辑操作的结果来置 1 或清零。此外，不能对 $\overline{TO}$ 和 $\overline{PD}$ 位进行写操作，所以当执行一条把状态寄存器作为目标寄存器的指令后，状态寄存器的结果可能和预想的不一样。

例如，指令 `CLRF STATUS` 将状态寄存器的高 3 位清零，将 Z 标志位置 1。操作后状态寄存器的结果为 `000u u1uu`（u 表示未变化）。

因此，建议仅使用位操作指令 `BCF`、 `BSF` 或传送指令 `MOVWF` 来改变状态寄存器，因为这些指令不影响该寄存器中的 Z、 C 或 DC 标志位。关于其它不影响任何状态位的指令，请见表 5-1。

注 1：	一些器件不需要 IRP 和 RP1 位（STATUS<7:6>）。本章（CPU 和 ALU）未使用这些位，应该将这些位保持为零。不建议将这些位作为通用读 / 写位，因为这可能会影响代码对未来产品的向上兼容性。
注 2：	在减法运算中，C 和 DC 位分别作为借位和辅助借位之反。

寄存器 5-1： 状态寄存器

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP**：寄存器组选择位 （用于间接寻址）
1：选择 Bank 2， Bank3 （100h - 1FFh）
0：选择 Bank 0， Bank1 （00h - FFh）

对于只有 Bank0 和 Bank1 的器件，保留 IRP 位，且应始终保持为 0。

bit 6:5 **RP1:RP0**：寄存器组选择位 （用于直接寻址）
11：Bank 3 （180h - 1FFh）
10：Bank 2 （100h - 17Fh）
01：Bank 1 （80h - FFh）
00：Bank 0 （00h - 7Fh）

每组 128 个字节。对于只有 Bank 0 和 Bank1 的器件，保留 IRP 位，且应始终保持为 0。

bit 4 **$\overline{TO}$**：超时位
1 = 上电、执行 CLRWDT 或 SLEEP 指令后

0 = 发生看门狗定时器超时

bit 3 **$\overline{PD}$**：低功耗标志位
1 = 上电或执行 CLRWDT 指令后
0 = 执行 SLEEP 指令后

bit 2 **Z**：零标志位
1 = 算术或逻辑运算结果为 0
0 = 算术或逻辑运算结果不为 0

bit 1 **DC**：辅助进位 / 借位标志位（ADDWF、ADDLW、SUBLW 和 SUBWF 指令）（辅助借位极性相反）
1 = 结果的低 4 位向高 4 位进位 / 低 4 位向高 4 位无借位
0 = 结果的低 4 位没有向高 4 位进位 / 低 4 位向高 4 位借位

bit 0 **C**：进位 / 借位标志位 （ADDWF、 ADDLW、 SUBLW 和 SUBWF 指令）
1 = 结果的最高位有进位 / 最高位无借位
0 = 结果的最高位无进位 / 最高位有借位

注： 借位的极性是相反的。减法指令通过加上第二个操作数 2 的补码来实现。对于移位指令 （RRF 和 RLF）， C 位值来自源寄存器的最高位或最低位。

图注		
R = 可读位	W = 可写位	
U = 未用位，读为 “0”		-n = 上电复位时的值

5.7 OPTION_REG 寄存器

OPTION_REG 寄存器是可读写寄存器，它包含配置 TMR0/WDT 的预分频器、外部 INT 中断、TMR0 和 PORTB 弱上拉的各个控制位。

寄存器 5-2： OPTION_REG 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
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

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

bit 6 **INTEDG**：中断触发边沿选择位
1 = INT 引脚的上升沿触发中断
0 = INT 引脚的下降沿触发中断

bit 5 **T0CS**：TMR0 时钟源选择位
1 = T0CKI 引脚上的外部时钟
0 = 内部指令周期时钟 （CLKOUT）

bit 4 **T0SE**：TMR0 计数边沿选择位
1 = T0CKI 引脚上的下降沿递增
0 = T0CKI 引脚上的上升沿递增

bit 3 **PSA**：预分频器分配位
1 = 预分频器分配给 WDT
0 = 预分频器分配给 Timer0 模块

bit 2-0 **PS2:PS0**：预分频比选择位

位值	TMR0 比率	WDT 比率
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

图注
R = 可读位　　W = 可写位
U = 未用位，读为 “0”　　-n = 上电复位时的值

注： 如果需要 TMR0 寄存器得到 1:1 的预分频比，可以把预分频器分配给看门狗定时器 WDT （即 PSA=1）。

5.8 电源控制寄存器

利用电源控制（PCON）寄存器中的标志位和TO及PD位，用户可以区别各种不同的器件复位。

注 1：	$\overline{\text{BOR}}$ 在上电复位时的值是未知的。它必须由用户软件置位，并在以后复位时检查该位是否为零，可判断有无欠压发生。如果禁用欠压电路（通过清除配置字里的 BODEN 位），则不必关心 $\overline{\text{BOR}}$ 状态位，也没必要预测。
注 2：	建议在检测到上电复位时，将 $\overline{\text{POR}}$ 位清零，以便检测后续的上电复位。

寄存器 5-3： 电源控制寄存器

R-u	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
MPEEN	—	—	—	—	$\overline{\text{PER}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$
bit 7							bit 0

bit 7 **MPEEN**：存储器奇偶校验错误状态位
该位反映 MPEEN 配置位的值。

bit 6:3 **未用位**：读为 “0”

bit 2 **$\overline{\text{PER}}$**：存储器奇偶校验错误复位状态位
1 = 没有错误发生
0 = 发生程序存储器取奇偶校验错误
（在上电复位后必须用软件置 1）

bit 1 **$\overline{\text{POR}}$**：上电复位状态位
1 = 未发生上电复位
0 = 发生上电复位（上电复位后，由软件置 1）

bit 0 **$\overline{\text{BOR}}$**：欠压复位状态位
1 = 未发生欠压复位
0 = 发生欠压复位（欠压复位后，由软件置 1）

图注	
R = 可读位	W = 可写位
U = 未用位，读为 “0”	-n = 上电复位时的值

5.9 设计技巧

问 1： *我的程序算法好象工作不正确。*

答 1：

1. 指令的目标寄存器可能被指定为 W 寄存器 （d=0），而不是数据寄存器 （d = 1）。
2. 可能没有正确选择寄存器组选择位（RP1:RP0 或 IRP）。而且如果使用了中断的话，在退出中断处理时，可能没有正确恢复寄存器组选择位。

问 2： *我不能修改状态寄存器的标志位。*

答 2：

如果一条影响 Z、 DC 或 C 位的指令的目标寄存器是状态寄存器，会禁止对这些位进行直接写操作。这些位是根据器件的逻辑操作结果来置位或清零。因此，要修改状态寄存器里的这些位，建议使用指令 BCF 和 BSF。

5.10 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制，即可使用。目前与 CPU 和 ALU 相关的应用笔记有：

标题	应用笔记 #
Fixed Point Routines	AN617
IEEE 754 Compliant Floating Point Routines	AN575
Digital Signal Processing with the PIC16C74	AN616
Math Utility Routines	AN544
Implementing IIR Digital Filters	AN540
Implementation of Fast Fourier Transforms	AN542
Tone Generation	AN543
Servo Control of a DC Brushless Motor	AN532
Implementation of the Data Encryption Standard using the PIC17C42	AN583
PIC16C5X / PIC16CXX Utility Math Routines	AN526
Real Time Operating System for PIC16/17	AN585

5.11 版本历史

版本 A

这是描述 CPU 和 ALU 的初始发行版。

第 6 章 存储器构成

目录

本章包括下面一些主要内容：

6.1 简介 6-2
6.2 程序存储器构成 6-2
6.3 数据存储器构成 6-8
6.4 初始化 6-14
6.5 设计技巧 6-16
6.6 相关应用笔记 6-17
6.7 版本历史 6-18

6.1 简介

本章主要包括两种存储器模块：程序存储器和数据存储器。每一个模块都有自己的总线，所以在同一个振荡周期内可对两种存储器模块同时进行访问。

数据存储器可以更进一步地分成通用寄存器和特殊功能寄存器 （SFR）。控制内核的特殊功能寄存器的操作将在本章介绍；而控制外设模块的特殊功能寄存器将在论述每个外设模块的相应章节中介绍。

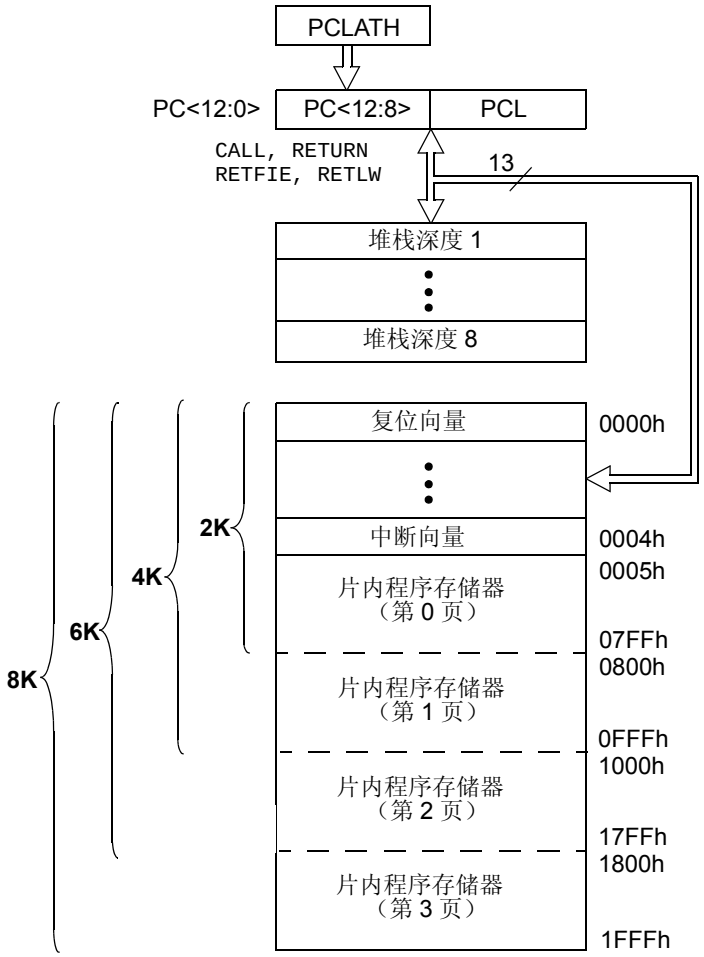
6.2 程序存储器构成

中档系列单片机有一个 13 位的程序计数器，可以寻址 8K × 14 位的程序存储空间。程序存储器总线宽度（指令字）为 14 位。由于所有的指令均为单字指令，所以一个具有 8K × 14 位程序存储器的器件可以存储 8K 条指令。很易于确定是否有充足的程序存储空间来实现应用程序。

中档系列单片机把程序存储器分成 4 页，每页 2K 字（0h - 7FFh,、800h - FFFh、1000h - 17FFh 和 1800h - 1FFFh)。图 6-1 所示为程序存储器映射和一个 8 级深度硬件堆栈。实际上单片机可能只实现了图中所示存储器的一部分，这与器件型号有关。关于单片机所提供的存储器，请查阅单片机的数据手册。

为了能在程序存储器页之间跳转，必须修改程序计数器（PC）的高位。这是通过在 PCLATH（程序计数器高位锁存器）中写入需要的值来完成的。如果指令连续运行，无需任何用户的干预，程序计数器即可以跨页。对于那些程序存储器不足 8K 字的器件，访问超过物理地址空间的存储单元时，会回到有效的程序存储空间。也就是说，在一个有 4K 字存储空间的单片机中，寻址 17FFh 实际就是寻址 7FFh。 2K 字或更少程序存储空间的器件不需要分页。

图 6-1：　　中档系列单片机的程序存储器映射和堆栈



注　1：不是所有的器件都实现了上述全部程序存储空间。
　　2：标定数据可烧写到程序存储器中。

6.2.1 复位向量

对于任何单片机，复位都将使程序计数器指向地址 0h，我们称这个地址为“复位向量地址”，也就是单片机发生复位时，程序执行的入口地址。

任何复位操作都会将 PCLATH 寄存器的内容清零。这表明，复位向量地址 （0h）处的任何转移指令都将跳转到程序存储器的第 0 页 （PAGE0）。

6.2.2 中断向量

当响应中断时， PC 指向地址 0004h，我们称这个地址为 “中断向量地址”。当 PC 指向中断向量时， PCLATH 寄存器的值并不会被修改。这意味着，在中断服务程序中，在改写 PC 实现程序跳转前，应按目的地址所处的实际程序页面先设定 PCLATH 寄存器。在中断服务程序修改PCLATH寄存器前，应将原PCLATH的内容保存起来，以便从中断服务程序返回时恢复PCLATH。

6.2.3 标定信息

某些器件在程序存储器中存储标定信息。在器件最终测试时， Microchip 将标定信息写入程序存储器。应用程序利用这些值可以获得更好的运行结果。标定信息通常放在程序存储器的末尾，并以 RETLW 指令形式实现，该指令所带的立即数就是标定信息。

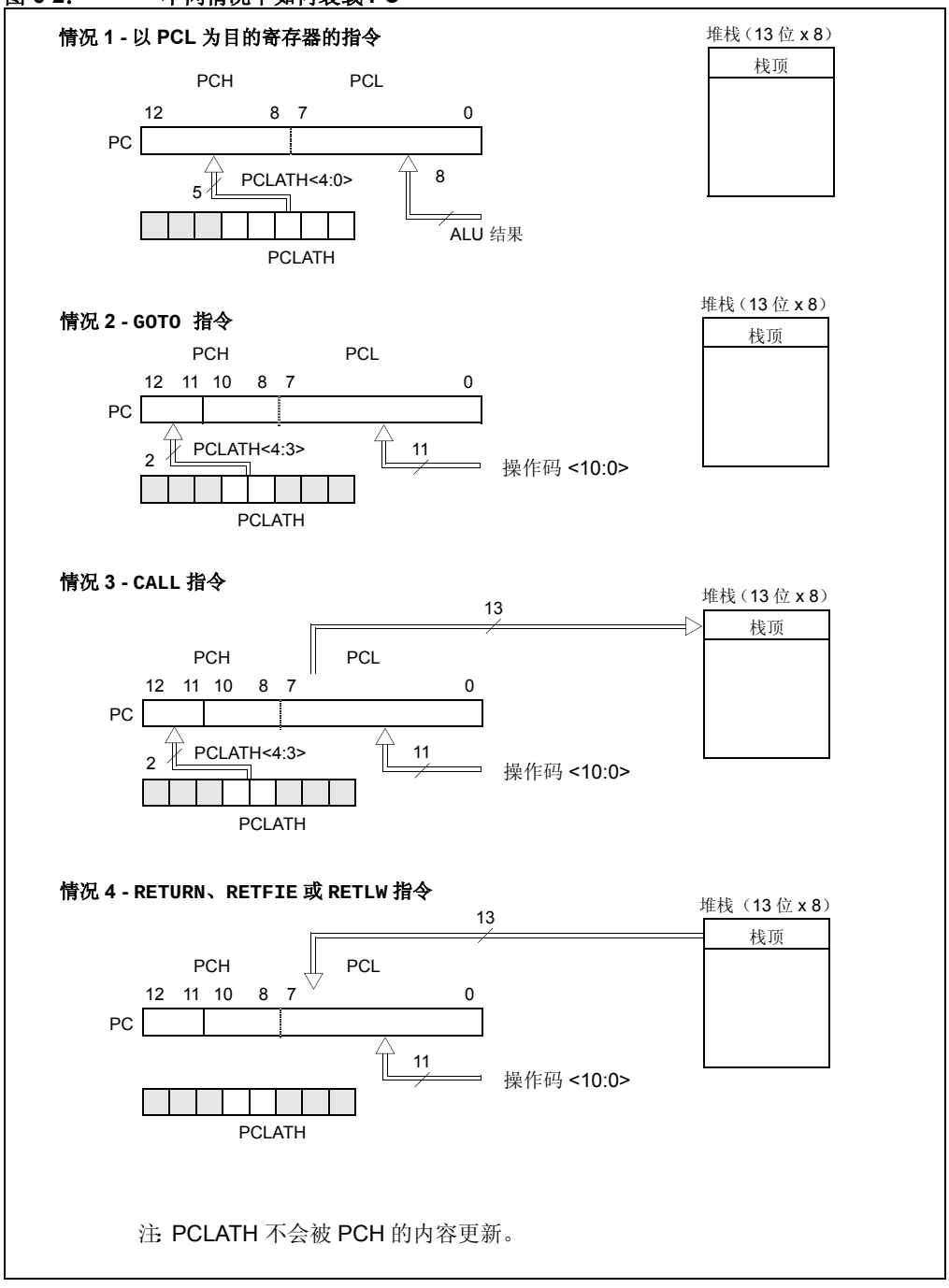
注：	对于窗口型器件，在擦除器件内容**前** （同时会擦除标定信息），务必先记下所有的标定值。这样在重新烧写器件时能恢复标定值。建议将标定值写在封装上。

6.2.4 程序计数器 （PC）

程序计数器指定要取出执行的指令的地址，其宽度为 13 位，其中低 8 位来自 PCL 寄存器，该寄存器可读写的，而高 5 位 （PC<12:8>）来自 PCH 寄存器 （不可直接读写）。 PCH 寄存器的值只能通过 PCLATH 寄存器来更新。

图 6-2 为装载 PC 值的四种情况。情况 1 为写 PCL 时，如何装载 PC（PCLATH<4:0>→PCH）；情况 2 为执行 GOTO 指令时，如何装载 PC (PCLATH<4:3> → PCH)；情况 3 为执行 CALL 指令时，如何装载 PC （PCLATH<4:3> → PCH）以及 PC 值如何压入栈顶；情况 4 为执行返回指令时，如何装载 PC，此时 PC 值从栈顶装载 （弹出）。

图 6-2： 不同情况下如何装载 PC



注：PCLATH 不会被 PCH 的内容更新。

6.2.4.1 相对跳转指令

程序的相对跳转指令是通过向程序计数器加一个偏移量来实现的（ADDWF PCL），当使用相对跳转指令方法对表进行读操作时，要注意表地址是否超过了 PCL 寄存器的寻址范围 （每块 256 个字节）。

注：	对程序计数器 （PCL）的任何写操作，都会使 PCLATH 的低五位装载到 PCH 中。

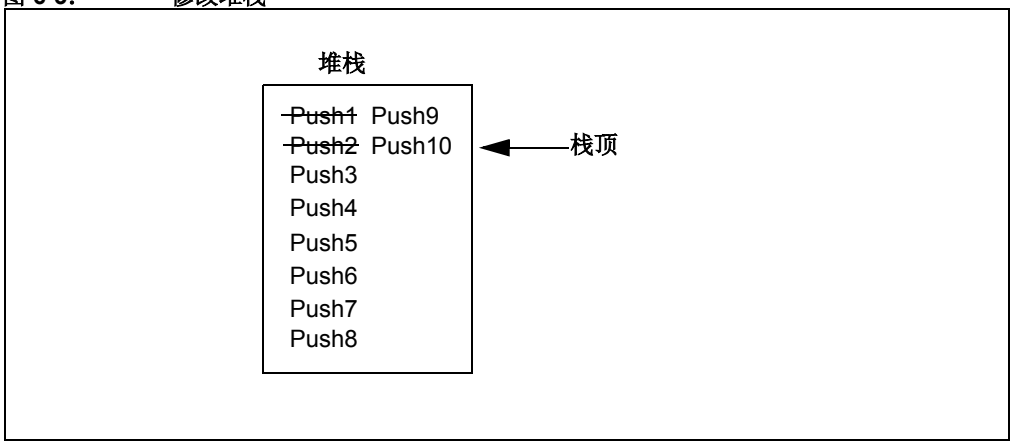
6.2.5 堆栈

堆栈允许 8 级深度的子程序嵌套调用和中断。堆栈包含了程序执行分支的返回地址。

中档系列单片机有一个 8 级深度、13 位宽的硬件堆栈。堆栈既不占用程序存储空间也不占用数据存储空间，栈指针不能读写。当执行 CALL 指令或响应中断发生跳转时， PC 值被压入堆栈（PUSH）。而执行 RETURN、RETLW 或 RETFIE 指令时， PC 值从堆栈弹出（POP）。执行压栈或出栈操作时，不会修改 PCLATH 寄存器。

压栈 （PUSH） 8 次之后，进行第 9 次压栈时，进栈的数据将覆盖第 1 次压栈存储的数据，而第 10次压栈时进栈的数据将覆盖第2次压栈存储的数据，依此类推。一个堆栈被覆盖的例子如图 6-3 所示。

图 6-3： 修改堆栈



注 1：	没有用于表示堆栈溢出或堆栈下溢条件的状态位。
注 2：	没有称为 PUSH 或 POP 的指令或助记符。而实现类似效果的操作是执行 CALL、RETURN、 RETLW 和 RETFIE 指令，或转到中断向量地址。

6.2.6 程序存储器分页

某些器件的程序存储器空间大于 2K 字，但是 CALL 和 GOTO 指令只有 11 位地址范围，这 11 位地址只允许在 2K 存储空间范围内跳转。为了使 CALL 和 GOTO 指令可以访问整个 8K 的程序存储地址范围，必须有另外两位来指定程序存储器页。将PCLATH<4:3>位作为页面选择位（图 6-2）。在执行 CALL 或 GOTO 指令前，用户必须确保正确设置页面选择位 PCLATH<4:3>，以便指向需要的程序存储页面（图 6-2）。当执行一条返回指令时，整个 13 位 PC 地址值都从堆栈弹出，不需要再对 PCLATH<4:3> 位进行设置。

注：	当器件的程序存储器空间小于或等于 2K 字时，可忽略用来存取有多个页面的程序存储器的页面选择位 （PCLATH<4:3>）。但不推荐将 PCLATH<4:3> 位作为一般读写位使用，因为这样做可能影响与将来产品的向上兼容性。 对于程序存储器空间在 2K 到 4K 字之间的器件，可忽略页面选择位 PCLATH<4>，因为它是用来寻址 2、 3 页 （1000h ～ 1FFFh）的。通常也不推荐将 PCLATH<4> 作为一般读写位使用，因为这样做可能影响与将来产品的向上兼容性。

例 6-1 是调用在程序存储器第 1 页上子程序的例子。本例假使 PCLATH 寄存器由中断服务程序保存和恢复 （如果使用了中断）。

例 6-1： 从第 0 页调用第 1 页的子程序

```
     ORG 0x500
     BSF    PCLATH,3  ; Select Page1 (800h-FFFh)
     CALL   SUB1_P1   ; Call subroutine in Page1 (800h-FFFh)
       :              ;
       :              ;
     ORG 0x900        ;
SUB1_P1:              ; called subroutine Page1 (800h-FFFh)
       :              ;
     RETURN           ; return to Call subroutine in Page0 (000h-7FFh)
                      ;
```

6.3 数据存储器构成

数据存储器由特殊功能寄存器（SFR）和通用寄存器（GPR）组成。SFR 控制器件的操作，而 GPR 则是数据存储和改写的通用区域。

SFR 和 GPR 数据存储区分成不同的存储区。GPR 区分成不同的存储区，以实现对超过 96 字节的通用 RAM 的寻址。SFR 是用来控制外设和内核功能的寄存器。STATUS 寄存器的存储区选择控制位（STATUS<7:5>）用于选择存储区。图 6-5 是数据存储器的构成映射，这个映射与器件型号有关。

从一个寄存器向另一个寄存器传送数据时，必须通过 W 寄存器。这意味着所有寄存器之间的数据传送，都需要两个指令周期。

整个数据存储器可以采用直接寻址或间接寻址来存取。直接寻址可能需要使用 RP1、RP0 位，间接寻址需要用到指针寄存器（FSR）。间接寻址数据存储器的存储区 0/ 存储区 1 或存储区 2/ 存储区 3 时，要使用状态寄存器的间接寄存器指针（IRP）位。

6.3.1 通用寄存器（GPR）

某些中档单片机的 GPR 区分成不同的存储区，上电复位并不能初始化 GPR，其它的复位也不能改变 GPR 的值。

寄存器既可以直接寻址，也可以使用指针寄存器 FSR 间接寻址。某些器件具有各数据存储区共享的公用数据存储区，对公用数据存储区的读写不必考虑当前所在存储区，可使用同一个地址单元（值），我们称这个区域为公用 RAM。

6.3.2 特殊功能寄存器（SFR）

特殊功能寄存器由 CPU 和外设使用，用于控制器件的操作，这类寄存器实现为静态 RAM 形式。

特殊功能寄存器可分为两类，一类与内核功能有关，另一类与外设功能有关。本章将讲述与内核功能有关的特殊功能寄存器，另一类与外设功能操作有关的特殊功能寄存器将在相应的外设功能模块章节中讲述。

所有中档单片机的 SFR 寄存器区也分成不同的存储区。在这些存储区间切换时，需要设置状态（STATUS ）寄存器的 RP0、RP1 位来选择所需存储区。某些 SFR 寄存器会被上电复位和其它复位初始化，而有些一些 SFR 寄存器在复位时不会被初始化。

注： 可能有通用寄存器映射到特殊功能寄存器区。

寄存器既可以直接寻址，也可以通过指针寄存器间接寻址。

6.3.3 存储区划分

数据存储器分为 4 个存储区，每个存储区包括特殊功能寄存器和通用寄存器。使用直接寻址时，为在这些存储区之间切换，需要设置状态寄存器的 RP0、RP1 位以选择需要的存储区。状态寄存器的 IRP 位用于间接寻址。

表 6-1： 直接和间接寻址时的存储区控制

存储区	直接寻址（RP1:RP）	间接寻址（IRP）
0	0 0	0
1	0 1	
2	1 0	1
3	1 1	

每个存储区最多可有 128 字节 （7FH）。特殊功能寄存器安排在存储区的低地址单元；通用寄存器安排在高地址单元。所有数据存储器都是用静态 RAM。所有存储区都包括特殊功能寄存器。为了减少程序代码和提高存取速度，存储区 0 中某些使用率高的特殊功能寄存器映射在其它存储区中。

随着产品的发展，其数据存储器的设计布局有一些变化。对于所有新器件来说，标准的数据存储器构成如图 6-5 所示。在这个存储器映射中，所有存储区的最后 16 字节都映射到存储区 0 中，这可以降低用于现场切换的软件开销。用**粗体**表示的寄存器存在于每种单片机中，其它寄存器的有无与外设模块有关。图中没有示出所有的外设寄存器，因为针对不同的器件，在某些文件地址处的寄存器定义与这里所显示的不同。除了使用本手册所提供的所有图、表和说明外，也应参阅特定器件的数据手册来核实细节。

图 6-4： 直接寻址

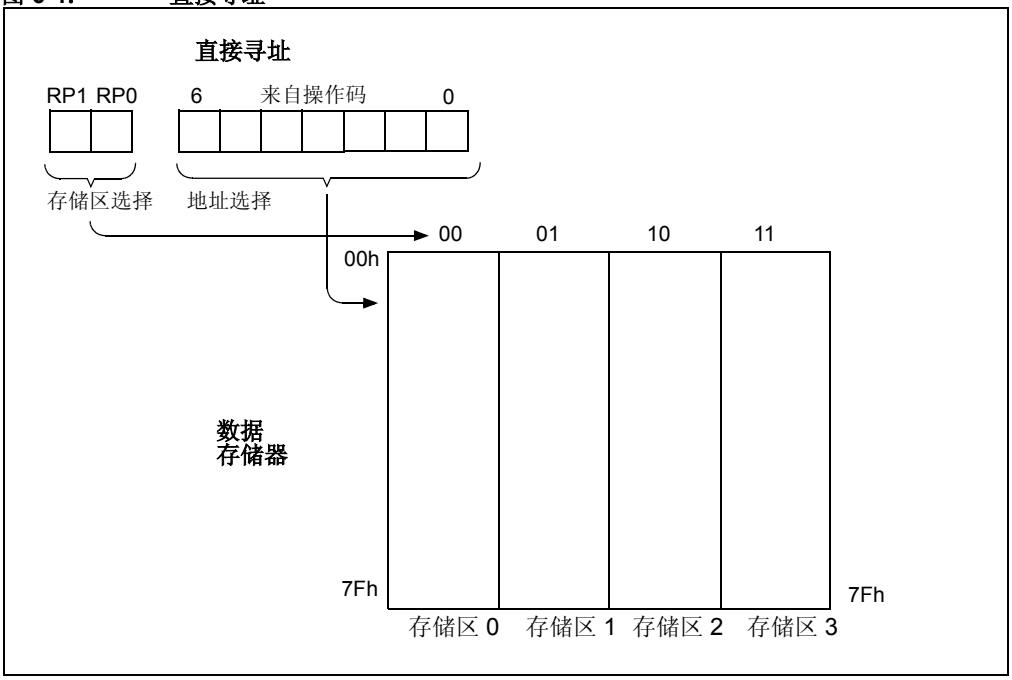


图 6-5： 寄存器映射

寄存器地址	存储区 0	寄存器地址	存储区 1	寄存器地址	存储区 2 (5)	寄存器地址	存储区 3 (5)
00h	**INDF**	80h	**INDF**	100h	**INDF**	180h	**INDF**
01h	**TMR0**	81h	**OPTION_REG**	101h	**TMR0**	181h	**OPTION_REG**
02h	**PCL**	82h	**PCL**	102h	**PCL**	182h	**PCL**
03h	**STATUS**	83h	**STATUS**	103h	**STATUS**	183h	**STATUS**
04h	**FSR**	84h	**FSR**	104h	**FSR**	184h	**FSR**
05h	PORTA	85h	TRISA	105h		185h	
06h	PORTB	86h	TRISB	106h	PORTB	186h	TRISB
07h	PORTC	87h	TRISC	107h	PORTF	187h	TRISF
08h	PORTD	88h	TRISD	108h	PORTG	188h	TRISG
09h	PORTE	89h	TRISE	109h		189h	
0Ah	**PCLATH**	8Ah	**PCLATH**	10Ah	**PCLATH**	18Ah	**PCLATH**
0Bh	**INTCON**	8Bh	**INTCON**	10Bh	**INTCON**	18Bh	**INTCON**
0Ch	**PIR1**	8Ch	**PIE1**	10Ch		18Ch	
0Dh	PIR2	8Dh	PIE2	10Dh		18Dh	
0Eh	TMR1L	8Eh	**PCON**	10Eh		18Eh	
0Fh	TMR1H	8Fh	OSCCAL	10Fh		18Fh	
10h	T1CON	90h		110h		190h	
11h	TMR2	91h		111h		191h	
12h	T2CON	92h	PR2	112h		192h	
13h	SSPBUF	93h	SSPADD	113h		193h	
14h	SSPCON	94h	SSPATAT	114h		194h	
15h	CCPR1L	95h		115h		195h	
16h	CCPR1H	96h		116h		196h	
17h	CCP1CON	97h		117h		197h	
18h	RCSTA	98h	TXSTA	118h		198h	
19h	TXREG	99h	SPBRG	119h		199h	
1Ah	RCREG	9Ah		11Ah		19Ah	
1Bh	CCPR2L	9Bh		11Bh		19Bh	
1Ch	CCPR2H	9Ch		11Ch		19Ch	
1Dh	CCP2CON	9Dh		11Dh		19Dh	
1Eh	ADRES	9Eh		11Eh		19Eh	
1Fh	ADCON0	9Fh	ADCON1	11Fh		19Fh	
20h	通用寄存器 (2)	A0h	通用寄存器 (3)	120h	通用寄存器 (3)	1A0h	通用寄存器 (3)
		EFh		16Fh		1EFh	
F0h	映射到存储区 0 70h - 7Fh (4)			170h	映射到存储区 0 70h - 7Fh (4)	1F0h	映射到存储区 0 70h - 7Fh (4)
7Fh		FFh		17Fh		1FFh	

注 1：**黑体**标注的寄存器在各种单片机中都存在。
2：有些地址单元可能没有实现，读为 ‘0’。
3：这些地址单元可能没有实现。根据所使用器件的不同，对这些未实现地址单元的访问会有不同，详细信息请参考具体的器件数据手册。
4：有些器件没有将这些寄存器映射到存储区 0，此时这些寄存器就作为通用 RAM。
5：有些器件可能没有这些存储区，未实现存储区中的地址单元读为 ‘0’。
6：通用寄存器也可以位于特殊功能寄存器区中。

图 6-6 所示为某些 18 引脚器件的寄存器存储区映射。未实现的寄存器读为 ‘0’。

图 6-6： 寄存器映射

	寄存器地址		寄存器地址
INDF	00h	**INDF**	80h
TMR0	01h	**OPTION_REG**	81h
PCL	02h	**PCL**	82h
STATUS	03h	**STATUS**	83h
FSR	04h	**FSR**	84h
PORTA	05h	**TRISA**	85h
PORTB	06h	**TRISB**	86h
	07h	**PCON**	87h
ADCON0 / EEDATA (2)	08h	ADCON1 / EECON1 (2)	88h
ADRES / EEADR (2)	09h	ADRES / EECON2 (2)	89h
PCLATH	0Ah	**PCLATH**	8Ah
INTCON	0Bh	**INTCON**	8Bh
通用寄存器 (3)	0Ch – 7Fh	通用寄存器 (4)	8Ch – FFh
存储区 0		存储区 1	

注 1：**黑体**标注的寄存器在每种单片机中都存在。
2：这些寄存器可能没有实现，或在一些器件中实现为其它的寄存器。
3：有些地址单元可能没有实现，读为 ‘0’。
4：这些地址单元在存储区 1 中没有实现。访问这些没有实现的地址单元时，将访问存储区 0 中相应的寄存器。

6.3.4 间接寻址、INDF 和 FSR 寄存器

间接寻址是寻址数据存储器的一种方式，间接寻址时指令中的数据存储器地址不是固定的，而是使用一个 SFR （指针寄存器）作为指向要读写的数据存储器地址的指针。由于该指针位于 RAM 中，其内容可以通过程序修改。这对于操作数据存储器中的数据表格很有用。图 6-7 所示为间接寻址的工作原理，实现了将数据传送到 FSR 寄存器的值指定的数据存储单元。

使用 INDF 寄存器可以实现间接寻址。任何使用 INDF 寄存器的指令实际上访问的是由指针寄存器 （FSR）所指向的寄存器。若使用间接寻址方式对 INDF 寄存器进行读操作 （FSR= '0'），读的结果将为 00h ；而使用间接寻址对 INDF 寄存器进行写操作，实际执行的是空操作，但可能会影响状态位。 8 位 FSR 寄存器与状态寄存器的 IRP 位 （STATUS<7>）相组合，可以得到一个 9 位有效地址。如图 6-8 所示：

图 6-7： 间接寻址

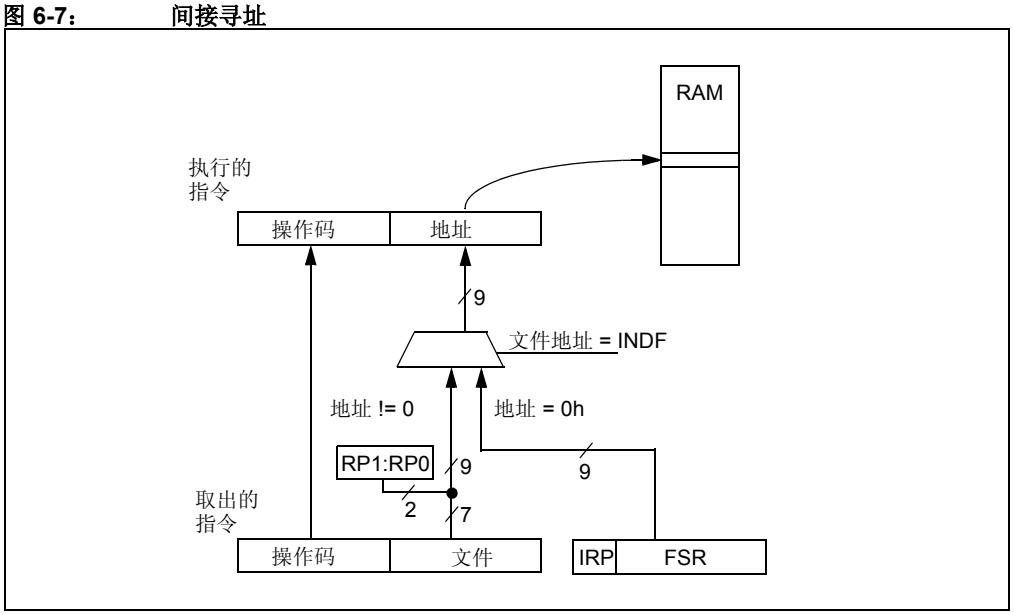


图 6-8： 间接寻址

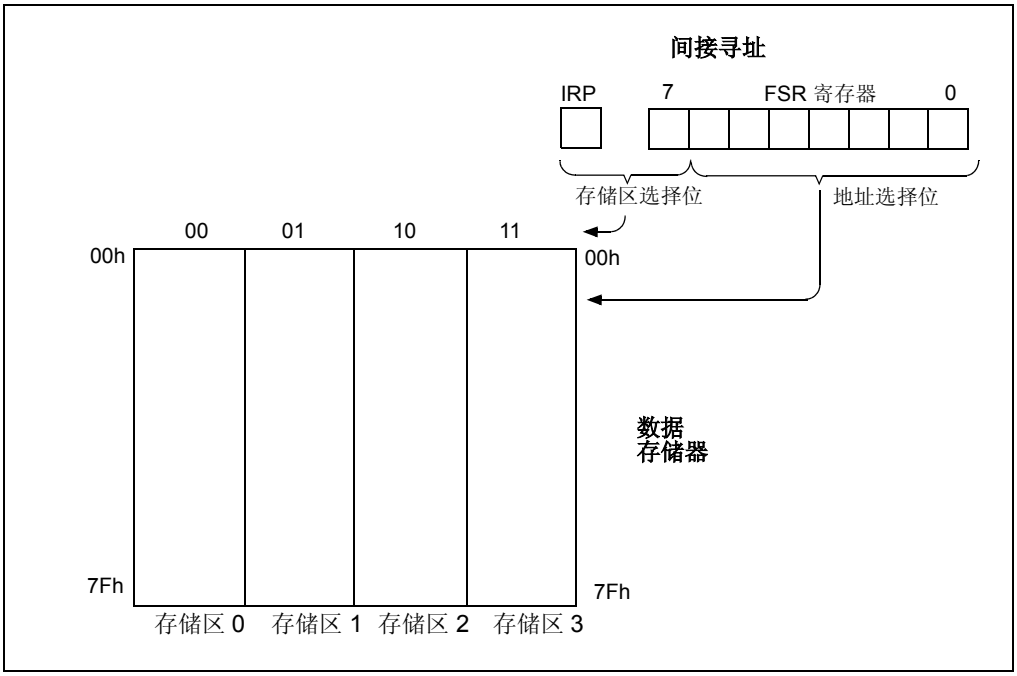


图 6-2 用最少的指令，采用间接寻址方法实现了对 20h-2Fh RAM 单元的清零。采用与此相同的方法，可以将已定义字节数的数据 （数据块）传送到 USART 的发送寄存器 （TXREG）。要发送数据块的起始地址很容易通过程序来修改。

例 6-2： 间接寻址

```
        BCF     STATUS, IRP  ; Indirect addressing Bank0/1
         MOVLW    0x20          ; Initialize pointer to RAM
         MOVWF    FSR           ;
NEXT     CLRF     INDF          ; Clear INDF register
         INCF     FSR,F         ; Inc pointer
         BTFSS    FSR,4         ; All done?
         GOTO     NEXT          ; NO, clear next
CONTINUE                        ;
         :                    ; YES, continue
```

6.4 初始化

例 6-3 举例说明了直接寻址时如何进行存储区切换。例 6-4 为初始化（清零）通用 RAM 的部分代码。

例 6-3： 存储区选择

```
    CLRF   STATUS         ; Clear STATUS register (Bank0)
    :                     ;
    BSF    STATUS, RP0    ; Bank1
    :                     ;
    BCF    STATUS, RP0    ; Bank0
    :                     ;
    MOVLW  0x60           ; Set RP0 and RP1 in STATUS register, other
    XORWF  STATUS, F      ;   bits unchanged (Bank3)
    :                     ;
    BCF    STATUS, RP0    ; Bank2
    :                     ;
    BCF    STATUS, RP1    ; Bank0
```

例 6-4： RAM 初始化

```
       CLRF   STATUS       ; Clear STATUS register (Bank0)
       MOVLW  0x20         ; 1st address (in bank) of GPR area
       MOVWF  FSR          ; Move it to Indirect address register
Bank0_LP
       CLRF   INDF0        ; Clear GPR at address pointed to by FSR
       INCF   FSR          ; Next GPR (RAM) address
       BTFSS  FSR, 7       ; End of current bank ? (FSR = 80h, C = 0)
       GOTO   Bank0_LP     ; NO, clear next location
;
; Next Bank (Bank1)
; (** ONLY REQUIRED IF DEVICE HAS A BANK1 **)
;
       MOVLW  0xA0         ; 1st address (in bank) of GPR area
       MOVWF  FSR          ; Move it to Indirect address register
Bank1_LP
       CLRF   INDF0        ; Clear GPR at address pointed to by FSR
       INCF   FSR          ; Next GPR (RAM) address
       BTFSS  STATUS, C    ; End of current bank? (FSR = 00h, C = 1)
       GOTO   Bank1_LP     ; NO, clear next location
;
; Next Bank (Bank2)
; (** ONLY REQUIRED IF DEVICE HAS A BANK2 **)
;
       BSF    STATUS, IRP  ; Select Bank2 and Bank3
                           ;    for Indirect addressing
       MOVLW  0x20         ; 1st address (in bank) of GPR area
       MOVWF  FSR          ; Move it to Indirect address register
Bank2_LP
       CLRF   INDF0        ; Clear GPR at address pointed to by FSR
       INCF   FSR          ; Next GPR (RAM) address
       BTFSS  FSR, 7       ; End of current bank? (FSR = 80h, C = 0)
       GOTO   Bank2_LP     ; NO, clear next location
;
; Next Bank (Bank3)
; (** ONLY REQUIRED IF DEVICE HAS A BANK3 **)
;
       MOVLW  0xA0         ; 1st address (in bank) of GPR area
       MOVWF  FSR          ; Move it to Indirect address register
Bank3_LP
       CLRF   INDF0        ; Clear GPR at address pointed to by FSR
       INCF   FSR          ; Next GPR (RAM) address
       BTFSS  STATUS, C    ; End of current bank? (FSR = 00h, C = 1)
       GOTO   Bank3_LP     ; NO, clear next location
    :                      ; YES, All GPRs (RAM) is cleared
```

6.5 设计技巧

问 1： *程序执行时似乎跑飞了。*

答 1：

当所使用器件的程序存储器超过 2K 字时，子程序调用可能需要在执行 CALL （或 GOTO）指令前，装载 PCLATH 寄存器，以正确指定子程序所在的程序存储器页面。无论标号 SUB_1 在程序存储器中的哪个地址，下列程序都将正确地装载 PCLATH 寄存器。

```
          MOVLW    HIGH (SUB_1)    ; Select Program Memory Page of
          MOVWF    PCLATH          ;     Routine.
          CALL     SUB_1           ; Call the desired routine
          :
          :
SUB_1     :                        ; Start of routine
          :
          RETURN                   ; Return from routine
```

问 2： *我需要将 RAM 初始化成 0，有什么简便的方法吗？*

答 2：

参见例 6-4。如果你所使用的器件没有 4 个数据存储区，可以删去一些代码。

6.6 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但其概念是相近的，通过适当修改并受到一定限制即可使用。目前与存储器构成相关的应用笔记有：

标题	应用笔记 #
Implementing a Table Read	AN556

6.7 版本历史

版本 A

这是描述存储器构成的初始发行版。

第 7 章 数据 EEPROM

目录

本章主要包括以下一些主要内容：

7.1 简介 7-2
7.2 控制寄存器 7-3
7.3 EEADR 7-4
7.4 EECON1 和 EECON2 寄存器 7-4
7.5 从 EEPROM 数据存储器中读数据 7-5
7.6 向 EEPROM 数据存储器中写数据 7-5
7.7 写校验 7-6
7.8 误写操作保护 7-7
7.9 代码保护配置下的数据 EEPROM 操作 7-7
7.10 初始化 7-7
7.11 设计技巧 7-8
7.12 相关应用笔记 7-9
7.13 版本历史 7-10

7.1 简介

在整个 VDD 范围内的正常运行期间， EEPROM 数据存储器是可以读写的。该存储器并不直接映射到寄存器文件空间，而是通过特殊功能寄存器来间接寻址。用四个特殊功能寄存器控制 EEPROM 存储器的读写，它们是：

- EECON1
- EECON2 (非实际存在的寄存器)
- EEDATA
- EEADR

EEDATA寄存器存放8位读写数据，而 EEADR寄存器存放EEPROM被访问过的地址。8位EEADR寄存器最多能够访问数据 EEPROM 的 256 个地址。 EEADR 寄存器可视为 EEPROM 存储器的间接寻址寄存器。 EECON1 包含相关的控制位，而 EECON2 是用于初始化读 / 写的寄存器。

一些器件未达到整个存储器映射空间，但地址范围都是从 0h 开始，到 EEPROM 空间的末地址。表 7-1 给出了一些常用器件的存储区可能的大小及地址范围。

表 7-1: 可能的 EEPROM 数据存储器大小

EEPROM 数据存储器大小 (1)	地址范围
64	0h - 3Fh
128	0h - 7Fh
256	0h - FFh

注 1: 目前，提供的器件只带有 64 字节的数据 EEPROM。

EEPROM 数据存储器可按字节进行读和写。一个字节的写操作将自动擦除并写入新的值 （即先擦除后写入）。EEPROM 是一种具有高擦 / 写周期的存储器。写入的时间由片内定时器控制，它随着电压、温度以及器件的不同而不同，请参见 AC 规范中的具体限制值。

当器件处于代码保护的情况下，CPU 可以继续对 EEPROM 存储器进行读写操作，而器件编程器将不再能读取这些存储器。

7.2 控制寄存器

寄存器 7-1: EECON1 寄存器

U-0	U-0	U-0	R/W-1	R/W-1	R/W-x	R/W-0	R/W-x
—	—	—	EEIF (1)	WRERR	WREN	WR	RD
bit 7							bit 0

bit 7:5 **未用 :** 读为 ‘0’

bit 4 **EEIF**：EEPROM 写操作中断标志位
1 = 写操作完成 (必须用软件清零)
0 = 写操作未完成或还未开始

bit 3 **WRERR**：EEPROM 错误标志位
1 = 写操作过早终止
(指正常操作期间出现 $\overline{\text{MCLR}}$ 或 WDT 复位)
0 = 写操作已完成

bit 2 **WREN**：EEPROM 写使能位
1 = 允许写入
0 = 禁止写入 EEPROM

bit 1 **WR**：写操作控制位
1 = 启动写周期。一旦写入完成，该位将被硬件清零。
写操作控制位只能用软件置 1 （不是清零）。
0 = EEPROM 的写周期已完成

bit 0 **RD**：读控制位
1 = 启动 EEPROM 读操作。读需要一个周期， RD 由硬件清零。
RD 只能用软件置 1 （不是清零）。
0 = 未启动 EEPROM 读操作

图注		
R = 可读	W = 可写	S = 可设置
U = 未用，读为 ‘0’		- n = 上电复位值

注 1： 对于将来的新器件，该位将包含在 PIR 寄存器中。

7.3 EEADR

EEADR 寄存器最多可寻址 256 字节的数据 EEPROM 存储器。

未使用的地址位都参加了译码。这意味着这些地址位必须始终为 '0'，以确保译码地址都在 EEPROM 的存储器空间内。

7.4 EECON1 和 EECON2 寄存器

EECON1 控制寄存器的低 5 位被实际使用，而高三位未使用且读为 '0'。

控制位 RD 和 WR 分别用于启动读和写操作。这些位不能被清零，只能用软件置 “1”。在读或写操作完成后，它们被硬件清零。由于软件无法对 WR 位清零，使写操作不会意外地过早终止。

WREN 位置 “1”后将允许一次写入操作。上电时， WREN 位被清零。正常运行时当写操作被 MCLR 或 WDT 复位中断时，WRERR 位置“1”。这些情况发生时，用户可在复位后检查 WRERR 位，判断是否需要进行重写。这些情况下， EEDATA 和 EEADR 寄存器中的数据和地址都不会发生变化。

当写操作完成时，中断标志位 EEIF 会置 “1”，该位必须用软件清零。

EECON2 不是一个物理存在的寄存器。对 EECON2 读的结果为 '0'。EECON2 寄存器专门用在 EEPROM 写操作的时序上。

7.5 从 EEPROM 数据存储器中读数据

要读出 EEPROM 存储器的值，用户应先把读地址写到 EEADR 寄存器中，并将控制位 RD (EECON1<0>) 置“1”。下一个指令周期，EEDATA 寄存器中的内容即为所读结果； EEDATA 将该值一直保留到下一次读操作或用户对该寄存器进行写操作为止。

例 7-1: 数据 EEPROM 的读操作

```
BCF     STATUS, RP0  ; Bank0
MOVLW   CONFIG_ADDR  ; Any location in Data EEPROM memory space
MOVWF   EEADR        ; Address to read
BSF     STATUS, RP0  ; Bank1
BSF     EECON1, RD   ; EE Read
BCF     STATUS, RP0  ; Bank0
MOVF    EEDATA, W    ; W = EEDATA
```

7.6 向 EEPROM 数据存储器中写数据

要向 EEPROM 数据区写入数据，用户首先必须将地址写入 EEADR 寄存器中，再将数据写入 EEDATA 寄存器中，然后必须按照特定顺序逐个字节地写入 EEPROM。

例 7-2: 数据 EEPROM 的写操作

必须按照这个顺序

```
BSF     STATUS, RP0  ; Bank1
BCF     INTCON, GIE  ; Disable INTs.
BSF     EECON1, WREN ; Enable Write
MOVLW   55h          ;
MOVWF   EECON2       ; 55h must be written to EECON2
MOVLW   AAh          ;   to start write sequence
MOVWF   EECON2       ; Write AAh
BSF     EECON1,WR    ; Set WR bit begin write
BSF     INTCON, GIE  ; Enable INTs.
```

如果未完全按照以上顺序（将 55h 写入 EECON2，将 AAh 写入 EECON2，然后将 WR 位置“1”）逐个字节写入，写操作将不会开始。我们强烈建议在该段代码中禁止中断。

此外， EECON1 的 WREN 位应置“1”，写操作才会使能。这种机制可防止由于意外执行错误代码（例如程序跑飞），造成对 EEPROM 的错误写入。除了 EEPROM 更新以外， WREN 位应始终保持为“0”。 WREN 位不会被硬件清零。

写操作启动后，将 WREN 位清零并不会对写周期产生影响。除非将 WREN 置“1”，否则 WR 位将不执行置“1”操作。

写操作完成后， WR 位将被硬件清零，同时 EEPROM 写入完成中断标志位 (EEIF) 被置“1”。用户可以使能中断或查询该位。 EEIF 必须用软件清零。

7.7 写校验

根据应用的需要，编程时一般要求对已写入EEPROM的数据与所需值相校验（例 7-3）。应用中，当 EEPROM 的数据位擦写次数接近其极限值时，就应进行写校验。 Total Endurance™ 磁盘将有助于您确定放心程度。

例 7-3: 写校验

```
       BCF   STATUS, RP0 ; Bank0
       :                 ; Any code can go here
       :                 ;
       MOVF  EEDATA, W   ; Must be in Bank0
       BSF   STATUS, RP0 ; Bank1
READ
       BSF   EECON1, RD  ; YES, Read the value written
       BCF   STATUS, RP0 ; Bank0
;
; Is the value written (in W reg) and read (in EEDATA) the same?
;
       SUBWF EEDATA, W   ;
       BTFSS STATUS, Z   ; Is difference 0?
       GOTO  WRITE_ERR   ; NO, Write error
       :                 ; YES, Good write
       :                 ; Continue program
```

7.8 误写操作保护

有些情况下，器件不宜向数据 EEPROM 存储器中写入数据，为了防止误写操作，器件内建了各种保护机制。上电时， WREN 位被清零。同时，上电定时器 （持续时间 72 ms ）也可防止误写 EEPROM。

在欠压、电源毛刺或软件故障期间，写操作启动顺序和 WREN 位可共同防止意外误写操作的发生。

7.9 代码保护配置下的数据 EEPROM 操作

当器件受代码保护时， CPU 能够对数据 EEPROM 进行读写操作。

ROM 器件有两个代码保护位：一个用于 ROM 程序存储器，另一个用于数据 EEPROM 存储器。欲获取更多有关代码保护位的信息，请参见器件的编程说明。

7.10 初始化

与其它模块不同， EEPROM 存储器模块没有初始化序列。对于 EEPROM 存储器的读操作请参见例 7-1。对于 EEPROM 存储器的写操作请参见例 7-2，要验证写操作是否成功完成请参见例 7-3。

与通用 RAM 类似，最好先将所有数据 EEPROM 的存储单元都初始化为已知状态。该初始化可在器件编程时或应用诊断模式下进行，因为用户一般不希望在复位时将数据 EEPROM 清零。

应用诊断模式可以是上电后器件检测到 I/O 引脚的状态。通过此模式，器件可以实现一些诊断功能。 I/O 引脚的状态必须是只有借助电平输入，强制进入该诊断模式才能达到的状态。

7.11 设计技巧

问 1: *为什么数据 EEPROM 的存储单元内不是我写入的数据？*

答 1:

产生这样的情况有几种可能，但最有可能的原因是您没有完全按照例 7-2 给出的写入顺序进行操作。如果您是按照例 7-2 给出的写入顺序进行的，请确认写操作时所有的中断均已禁止。

问 2: *为什么数据 EEPROM 中的数据被改动了？*

答 2:

只有在发生数据 EEPROM 存储器写操作时，数据才会发生改变。在发生欠压时 (已超出了正常的工作范围)，而器件尚未进入复位状态的情况下，可能发生偶发性的写入。在欠压期间，可通过使能内部欠压复位电路 (若存在)，也可使用外部复位电路对 PICmicro® 单片机进行复位，以保证在器件超出正常工作范围时，不发生数据 EEPROM 的写操作。

7.12 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与 EEPROM 存储器相关的应用笔记是：

标题	应用笔记 #
EEPROM Endurance Tutorial	AN601
How to get 10 Million Cycles out of your Microchip Serial EEPROM	AN602
Basic Serial EEPROM Operation	AN536
Everything a System Engineer needs to know about Serial EEPROM Endurance	AN537
Using the Microchip Endurance Predictive Software	AN562

7.13 版本历史

版本 A

这是描述数据 EEPROM 的初始发行版。

第 8 章 中断

目录

本章包括下面一些主要内容：

8.1 简介 8-2
8.2 控制寄存器 8-5
8.3 中断响应延时 8-10
8.4 INT 和外部中断 8-10
8.5 中断的现场保护 8-11
8.6 初始化 8-14
8.7 设计技巧 8-16
8.8 相关应用笔记 8-17
8.9 版本历史 8-18

8.1 简介

PICmicro® 单片机有丰富的中断源。虽然某些外设模块可能产生多个中断（比如 USART 模块），但一般情况下是一个外设模块只有一个中断源。目前的中断包括：

- INT 引脚中断 （外部中断）
- TMR0 溢出中断
- PORTB 电平变化中断 （引脚 RB7:RB4）
- 比较器变化中断
- 并行从动端口中断
- USART 中断
- 接收中断
- 发送中断
- A/D 转换完成中断
- LCD 中断
- 向 EEPROM 写数据完成中断
- Timer1 溢出中断
- Timer2 溢出中断
- CCP 中断
- SSP 中断

中断的控制和中断状态的表示至少需要一个寄存器。该寄存器是：

- INTCON

此外，如果器件有外设中断，则会有允许外设中断的寄存器和保存中断标志位的寄存器。根据器件的具体型号，这些寄存器为：

- PIE1
- PIR1
- PIE2
- PIR2

我们通常称这些寄存器为 PIR 和 PIE。如果将来的器件拥有更多的中断源，则会增加新的寄存器对 （PIR3 和 PIE3）。

中断控制寄存器 INTCON 记录请求内核中断的各个中断标志位、允许位以及全局中断允许位。

全局中断允许位 GIE（INTCON<7>）置位时，允许所有未屏蔽的中断；清零时，禁止所有中断。通过 INTCON 寄存器中的允许位也能禁止各相应的中断。 GIE 位在复位时被清零。

执行 “中断返回”指令 RETFIE 将退出中断服务程序，同时将 GIE 位置 “1”，从而可响应任何暂挂的中断。

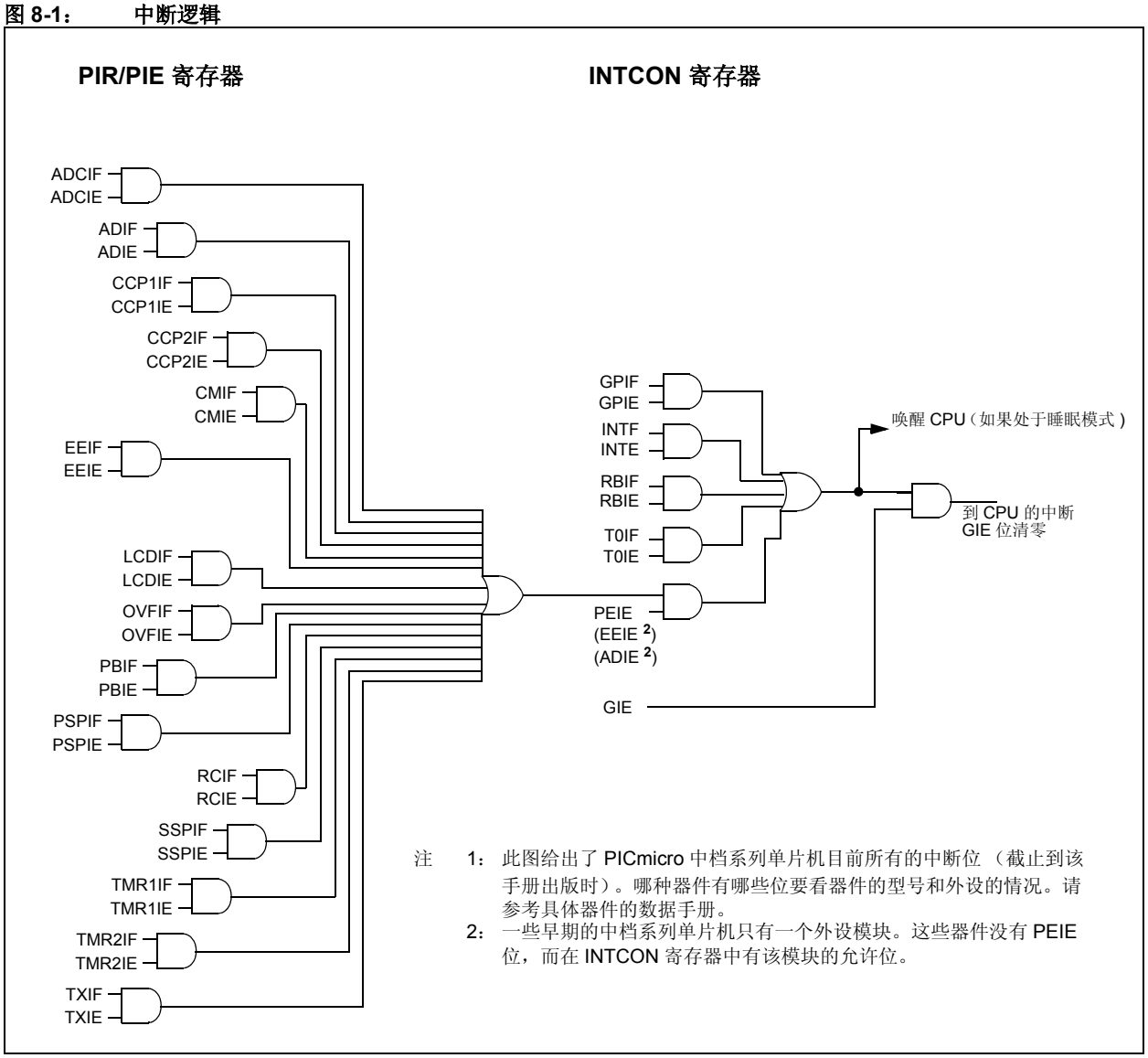
INTCON 寄存器包含以下中断的标志位和允许位：INT 引脚中断、RB 端口电平变化中断和 TMR0 溢出中断。 INTCON 寄存器还包含外设中断允许位 PEIE。当 PEIE 位置 “1”／清零时，将允许／禁止内核响应外设中断请求。

当一个中断被响应时， GIE 位被清零以禁止其它中断，返回地址压入堆栈， PC 中装入 0004H。在中断服务程序中，通过检测中断标志位可判断中断源。通常，中断标志位应在重新允许全局中断允许位 GIE 之前通过软件清零，以避免重复响应该中断。

在中断服务程序中，通过检测中断标志位可以判断中断源。各中断标志位的置位不受相应的中断屏蔽位和 GIE 位的状态影响。

注 1：	各中断标志位的置位不受对应的中断屏蔽位和 GIE 位的状态影响。
注 2：	当一条指令将 GIE 位清零时，会忽略原本将在其紧跟着的一个指令周期内等待响应的中断。CPU 将在紧跟着该指令的指令周期内执行一个空操作。被忽略的中断将继续等待，一直到 GIE 位再次被置位。

图 8-1： 中断逻辑



注 1：此图给出了 PICmicro 中档系列单片机目前所有的中断位（截止到该手册出版时）。哪种器件有哪些位要看器件的型号和外设的情况。请参考具体器件的数据手册。

2：一些早期的中档系列单片机只有一个外设模块。这些器件没有 PEIE 位，而在 INTCON 寄存器中有该模块的允许位。

8.2 控制寄存器

通常，单片机至少有三个和中断有关的寄存器。 INTCON 寄存器包含全局中断允许位 GIE 和外设中断允许位 PEIE。 PIE / PIR 寄存器对分别用于允许外设中断和显示中断标志状态。

8.2.1 INTCON 寄存器

INTCON 寄存器是一个可读写的寄存器，包含了多个允许位和标志位。

注：	当一个中断条件发生时，不管相应的中断允许位或全局允许位 GIE（INTCON<7>）的状态如何，中断标志位都将置 “1”。故中断标志位可以用于软件查询。

寄存器 8-1： INTCON 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
GIE	PEIE (3)	T0IE	INTE (2)	RBIE (1, 2)	T0IF	INTF (2)	RBIF (1, 2)
bit 7							bit 0

bit 7 **GIE**：全局中断允许位
1 = 允许所有未屏蔽的中断
0 = 禁止所有中断

bit 6 **PEIE**：外设中断允许位
1 = 允许所有未屏蔽的外设中断
0 = 禁止所有的外设中断

bit 5 **T0IE**：TMR0 溢出中断允许位
1 = 允许 TMR0 溢出中断
0 = 禁止 TMR0 溢出中断

bit 4 **INTE**：INT 外部引脚中断允许位
1 = 允许 INT 外部引脚中断
0 = 禁止 INT 外部引脚中断

bit 3 **RBIE (1)**：RB 端口电平变化中断允许位
1 = 允许 RB 端口电平变化中断
0 = 禁止 RB 端口电平变化中断

bit 2 **T0IF**：TMR0 溢出中断标志位
1 = TMR0 寄存器已经溢出 （必须用软件清零）
0 = TMR0 寄存器尚未发生溢出

bit 1 **INTF**：INT 外部引脚中断标志位
1 = 发生 INT 外部中断 （必须用软件清零）
0 = 未发生 INT 外部中断

bit 0 **RBIF (1)**：RB 端口电平变化中断标志位
1 = RB7:RB4 引脚中至少有一位的状态发生了变化 （必须用软件清零）
0 = RB7:RB4 引脚没有发生状态变化

图注		
R = 可读位	W = 可写位	
U = 未用位，读作 “0”		- n = 上电复位时的值

注 1：	某些型号单片机里， RBIE 位也称作 GPIE，而 RBIF 位称为 GPIF。
注 2：	某些型号单片机可能没有该功能。对于那些单片机，该位是保留的。
注 3：	对那些只有一个外设中断的单片机，这个位可能是 EEIE 或 ADIE。

8.2.2 PIE 寄存器

根据外设中断源的数量，可以有多个外设中断允许寄存器 （PIE1， PIE2） 。这些寄存器包含各外设中断的允许位。这些寄存器通常被称为 PIE。如果器件有 PIE 寄存器，那么要允许任何一个外设中断，必须将 PEIE 位置 “1”。

注： 要允许任何一个外设中断，必须将 PEIE （INTCON<6>）位置 “1”。

PIE 寄存器中各位的定义一般是固定的，但不保证未来的新器件也采用相同的定义。如果用户使用 Microchip 在头文件中提供的符号表示这些位，则可以忽略不同型号在位定义上的差别。这使得汇编器 / 编译器通过正确的寄存器和位名称能够自动查找这些位的位置。

寄存器 8-2： PIE 寄存器

R/W-0
（注 1）
bit 7 ... bit 0

bit **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

bit **TMR2IE**：TMR2 对 PR2 匹配中断允许位
1 = 允许 TMR2 对 PR2 匹配中断
0 = 禁止 TMR2 对 PR2 匹配中断

bit **CCP1IE**：CCP1 中断允许位
1 = 允许 CCP1 中断
0 = 禁止 CCP1 中断

bit **CCP2IE**：CCP2 中断允许位
1 = 允许 CCP2 中断
0 = 禁止 CCP2 中断

bit **SSPIE**：同步串行口中断允许位
1 = 允许 SSP 中断
0 = 禁止 SSP 中断

bit **RCIE**：USART 接收中断允许位
1 = 允许 USART 接收中断
0 = 禁止 USART 接收中断

bit **TXIE**：USART 发送中断允许位
1 = 允许 USART 发送中断
0 = 禁止 USART 发送中断

bit **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 **ADCIE**：积分型 A/D 转换器比较器翻转中断允许位
1 = 允许积分型 A/D 转换器中断
0 = 禁止积分型 A/D 转换器中断

bit **OVFIE**：积分型 A/D TMR 溢出中断允许位
1 = 允许积分型 A/D TMR 溢出中断
0 = 禁止积分型 A/D TMR 溢出中断

bit **PSPIE**：并行从动端口的读 / 写中断允许位
1 = 允许 PSP 的读 / 写中断
0 = 禁止 PSP 的读 / 写中断

bit **EEIE**：EEPROM 写操作完成中断允许位
1 = 允许 EEPROM 写操作完成中断
0 = 禁止 EEPROM 写操作完成中断

bit **LCDIE**：LCD 中断允许位
1 = 允许 LCD 中断
0 = 禁止 LCD 中断

bit **CMIE**：比较器中断允许位
1 = 允许比较器中断
0 = 禁止比较器中断

图注	
R = 可读位	W = 可写位
U = 未用位，读作 “0”	- n = 上电复位时的值

注 1： 各允许位的位置依具体型号而不同。请参考相关器件的数据手册，了解各位的位置。

8.2.3 PIR 寄存器

根据外设中断源的数量，可以有多个外设中断标志寄存器 （PIR1， PIR2）。这些寄存器包含各外设中断的标志位。这些寄存器通常被称为 PIR。

注 1：	当一个中断条件发生时，不管相应的中断允许位和全局允许位 GIE (INTCON<7>) 的状态如何，中断标志位都将置位。
注 2：	用户软件应在允许一个中断之前，先将该中断标志位清零；同时在响应该中断后，也应将该中断标志位清零。

PIR 寄存器中各位的定义一般是固定的，但不保证未来的新器件也采用相同的定义。如果用户使用 Microchip 在头文件中提供的符号表示这些位，则可以忽略不同型号在位定义上的差别。这使得汇编器 / 编译器能够自动查找指定寄存器内这些位的位置。

寄存器 8-3： PIR 寄存器

R/W-0
（注 1）
bit 7 ... bit 0

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

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

bit **CCP1IF**：CCP1 中断标志位

输入捕捉模式
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

输出比较模式
1 = 发生了 TMR1 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TRM1 寄存器的比较匹配

脉宽调制模式下
此模式下未定义

bit **CCP2IF**：CCP2 中断标志位

输入捕捉模式
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

输出比较模式
1 = 发生了 TMR1 寄存器的比较匹配 （必须用软件清零）
0 = 未发生 TRM1 寄存器的比较匹配

脉宽调制模式下
此模式下未定义

bit **SSPIF**：同步串行口中断标志位
1 = 完成发送 / 接收
0 = 等待发送 / 接收完成

bit **RCIF**：USART 接收中断标志位
1 = USART 接收缓冲器 RCREG 满 （当读取 RCREG 时清零）
0 = USART 接收缓冲器为空

bit **TXIF**：USART 发送中断标志位
1 = USART 发送缓冲器 TXREG 为空 （当写入 TXREG 时清零）
0 = USART 发送缓冲器满

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

寄存器 8-3： PIR 寄存器 （续）

bit **ADCIF**：积分型 A/D 转换器的比较器翻转中断标志位
1 = 完成 A/D 转换 （必须用软件清零）
0 = 未完成 A/D 转换

bit **OVFIF**：积分型 A/D TMR 溢出中断标志位
1 = 积分型 A/D TMR 发生了溢出 （必须用软件清零）
0 = 积分型 A/D TMR 没有溢出

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

bit **EEIF**：EEPROM 写操作完成中断标志位
1 = 数据 EEPROM 写操作已完成 （必须用软件清零）
0 = 数据 EEPROM 写操作未完成

bit **LCDIF**：LCD 中断标志位
1 = 发生 LCD 中断 （必须用软件清零）
0 = 未发生 LCD 中断

bit **CMIF**：比较器中断标志位
1 = 比较器输入发生变化 （必须用软件清零）
0 = 比较器输入未发生变化

图注		
R = 可读位	W = 可写位	
U = 未用位，读作 “0”		- n = 上电复位时的值

注 1： .

8.3 中断响应延时

中断响应延时被定义为从中断事件发生（中断标志位被置位）到地址 0004h 的指令开始执行（该中断是允许时）之间的这段时间。

对同步中断 （一般是器件的内部中断），响应延时为 3 个指令周期。

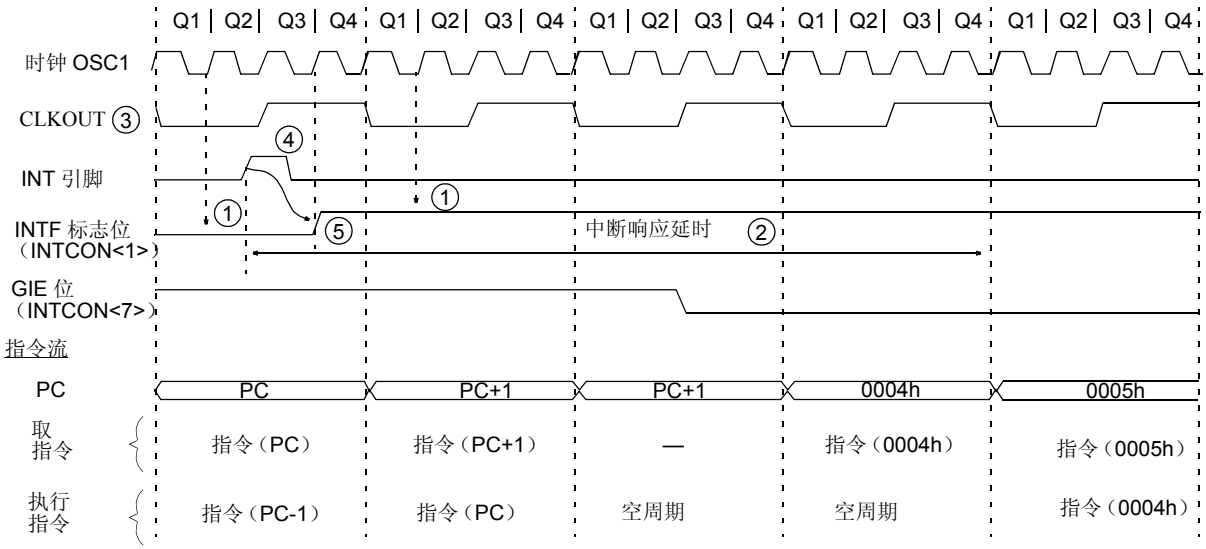
对异步中断 （一般是器件的外部中断，如 INT 引脚中断或 RB 端口的电平变化中断），中断响应延时将是3 - 3.75个指令周期。确切的延时取决于中断事件在指令周期的哪个时刻发生（图 8-2）。

对于单周期或双周期指令，中断响应延时是一样的。

8.4 INT 和外部中断

INT 引脚的外部中断是边沿触发的：如果 INTEDG 位（OPTION<6>）被置位，为上升沿触发；若清零，则为下降沿触发。当 INT 引脚上出现一个有效边沿时， INTF 标志位 （INTCON<1>）被置位。通过对 INTE 允许位 （INTCON<4>）置位 / 清零，该中断可以被允许 / 禁止。在重新允许该中断前，必须在中断服务程序中先用软件将 INTF 位清零。如果 INTE 位在进入休眠状态前被置位，则 INT 中断能把处理器从休眠状态中唤醒。GIE 位的状态取决于处理器在唤醒之后是否进入中断服务程序。请参见 **“看门狗定时器与休眠模式”一章**，了解休眠的详细信息和 INT 中断唤醒处理器的时序。

图 8-2： INT 引脚中断和其它外部中断的时序



注 1: INTF 标志在每个 Q1 时钟节拍采样。
2: 中断响应延时 = 3-4 TCY（这里 TCY = 指令周期）。
不管指令（PC）是单周期还是双周期，响应延时是一样的。
3: CLKOUT 在 RC 振荡器模式下才存在。
4: 关于 INT 脉冲的最小宽度要求，请参考 AC 规范。
5: 在 Q4-Q1 时钟节拍期间，随时可以对 INTF 位置位。

注： 任何外部信号（如定时器、捕捉和端口电平变化）引起的中断都有以上相似的时序。

8.5 中断的现场保护

在中断期间，仅将返回的 PC 地址压入堆栈。而用户通常可能还希望保存中断期间的一些重要寄存器值，如 W 寄存器和 STATUS 寄存器。这要通过软件来实现。

保存信息的操作通常被称作 “PUSHing”（压入），而恢复信息的操作被称作 “POPing”（弹出）。 PUSH 和 POP 不是指令符，而是概念性的操作。该操作通过一串指令序列实现。为了使代码易于移植，这些代码段可被写成宏形式（请参看 MPASM Assembler User’s Guide，了解宏创建的详细信息）。

例 8-1 说明了对于具有公共 RAM 的器件（PIC16C77），如何保存和恢复 STATUS 和 W 寄存器的值。用户定义的寄存器 W_TEMP，必须在各个存储区上都有定义，且应定义在各存储区内相同的偏移地址处（例如，在 Bank0 的 0x70 - 0x7F 范围内定义 W_TEMP）。用户定义的寄存器 STATUS_TEMP 应定义在 Bank0 中，本例即是如此。

例 8-1 的程序流程：

1. 保存 W 寄存器内容，不必考虑当前操作的是第几个存储区。
2. 在 Bank0 中保存 STATUS 寄存器内容。
3. 执行中断服务程序（ISR）代码。
4. 恢复 STATUS 寄存器（和存储区选择位寄存器）。
5. 恢复 W 寄存器。

如果在执行中断服务程序（ISR）代码前，还需保存其它内容，则应在保存了 STATUS 寄存器内容后（第 2 步）保存这些内容，在恢复 STATUS 寄存器内容前（第 4 步）恢复这些内容。

例 8-1： 在 RAM 内保存 STATUS 和 W 寄存器（适用于有公共 RAM 的器件）

```
    MOVWF   W_TEMP          ; Copy W to a Temporary Register
                            ;    regardless of current bank
    SWAPF   STATUS,W        ; Swap STATUS nibbles and place
                            ;    into W register
    MOVWF   STATUS_TEMP     ; Save STATUS to a Temporary register
                            ;    in Bank0
    :
    : (Interrupt Service Routine (ISR) )
    :
    SWAPF   STATUS_TEMP,W   ; Swap original STATUS register value
                            ;    into W (restores original bank)
    MOVWF   STATUS          ; Restore STATUS register from
                            ;    W register
    SWAPF   W_TEMP,F        ; Swap W_Temp nibbles and return
                            ;    value to W_Temp
    SWAPF   W_TEMP,W        ; Swap W_Temp to W to restore original
                            ;    W value without affecting STATUS
```

例 8-2 说明了对于没有公共 RAM 的器件（如 PIC16C74A），如何保存和恢复 STATUS 和 W 寄存器的值。用户定义的寄存器 W_TEMP，必须在各存储区上都有定义，且应定义在各存储区内相同的偏移地址处（例如，在 Bank0 的 0x70 - 0x7F 范围内定义 W_TEMP）。用户定义的寄存器 STATUS_TEMP 应定义在 Bank0 中。

不论 W_TEMP 定义在 Bank0 的 70h - 7Fh 范围内的何处，其它存储区内的相同位置都应作为该 W 寄存器的备份单元。

例 8-2 的程序流程：

1. 保存 W 寄存器内容，不必考虑当前操作的是第几个存储区。
2. 在 Bank0 中保存 STATUS 寄存器内容。
3. 执行中断服务程序（ISR）代码。
4. 恢复 STATUS 寄存器（和存储区选择位寄存器）。
5. 恢复 W 寄存器。

如果在执行中断服务程序（ISR）代码前，还需保存其它内容，则应在保存了 STATUS 寄存器内容后（第 2 步）保存这些内容，在恢复 STATUS 寄存器内容前（第 4 步）恢复这些内容。

例 8-2：　在 RAM 内保存 STATUS 和 W 寄存器（适用于无公共 RAM 的器件）

```
    MOVWF   W_TEMP          ; Copy W to a Temporary Register
                            ;    regardless of current bank
    SWAPF   STATUS,W        ; Swap STATUS nibbles and place
                            ;    into W register
    BCF     STATUS,RP0      ; Change to Bank0 regardless of
                            ;    current bank
    MOVWF   STATUS_TEMP     ; Save STATUS to a Temporary register
                            ;    in Bank0
    :
    : (Interrupt Service Routine (ISR) )
    :
    SWAPF   STATUS_TEMP,W   ; Swap original STATUS register value
                            ;    into W (restores original bank)
    MOVWF   STATUS          ; Restore STATUS register from
                            ;    W register
    SWAPF   W_TEMP,F        ; Swap W_Temp nibbles and return
                            ;    value to W_Temp
    SWAPF   W_TEMP,W        ; Swap W_Temp to W to restore original
                            ;    W value without affecting STATUS
```

例 8-3 说明了对于仅在Bank0内有通用RAM的器件（如PIC16C620），如何保存和恢复STATUS和 W 寄存器的值。在保存任何用户定义的寄存器前，必须检测存储区。用户定义的寄存器W_TEMP，必须在各存储区上都有定义，且应定义在各存储区内相同的偏移地址处。用户寄存器STATUS_TEMP 应定义在 Bank0 中。

例 8-3 的程序流程：

1. 检测当前操作的存储区。
2. 保存 W 寄存器内容，不必考虑当前操作的是第几个存储区。
3. 在 Bank0 中保存 STATUS 寄存器内容。
4. 执行中断服务程序 （ISR）代码。
5. 恢复 STATUS 寄存器 （和存储区选择位寄存器）。
6. 恢复 W 寄存器。

如果在执行中断服务程序 （ISR）代码前，还需保存其它内容，则应在保存了 STATUS 寄存器内容后 （第 2 步）保存这些内容，在恢复 STATUS 寄存器内容前 （第 4 步）恢复这些内容。

例 8-3： 在 RAM 内保存 STATUS 和 W 寄存器
（适用于仅在 Bank0 内有通用 RAM 的器件）

```
Push
      BTFSS   STATUS, RP0         ; In Bank 0?
      GOTO    RP0CLEAR            ; YES,
      BCF     STATUS, RP0         ; NO, Force to Bank 0
      MOVWF   W_TEMP              ; Store W register
      SWAPF   STATUS, W           ; Swap STATUS register and
      MOVWF   STATUS_TEMP         ;   store in STATUS_TEMP
      BSF     STATUS_TEMP, 1      ; Set the bit that corresponds to RP0
      GOTO    ISR_Code            ; Push completed
RP0CLEAR
      MOVWF   W_TEMP              ; Store W register
      SWAPF   STATUS, W           ; Swap STATUS register and
      MOVWF   STATUS_TEMP         ;   store in STATUS_TEMP
;
ISR_Code
      :
      : (Interrupt Service Routine (ISR) )
      :
;
Pop
      SWAPF   STATUS_TEMP, W      ; Restore Status register
      MOVWF   STATUS              ;
      BTFSS   STATUS, RP0         ; In Bank 1?
      GOTO    Restore_WREG        ; NO,
      BCF     STATUS, RP0         ; YES, Force Bank 0
      SWAPF   W_TEMP, F           ; Restore W register
      SWAPF   W_TEMP, W           ;
      BSF     STATUS, RP0         ; Back to Bank 1
      RETFIE                      ; POP completed
Restore_WREG
      SWAPF   W_TEMP, F           ; Restore W register
      SWAPF   W_TEMP, W           ;
      RETFIE                      ; POP completed
```

8.6 初始化

例 8-4 给出了初始化和中断允许的示例，其中 PIE1_MASK1 值就是将要写入中断允许寄存器的值。

例 8-5 给出了如何创建函数的宏定义。宏必须在使用前定义。为了调试方便，可将宏定义写在其它文件中并包含进来，这会使源代码看起来不那么混乱。所有需包含进来的宏定义文件必须在源程序前实现，这样简化了程序的调试，如例 8-6 所示。

例 8-7 给出了一个典型的中断服务程序结构。在执行中断代码前，该 ISR 用宏命令来保存和恢复寄存器。

例 8-4： 初始化和中断允许

```
PIE1_MASK1   EQU B'01101010'   ; This is the Interrupt Enable
      :                        ;    Register mask value
      :
      CLRF   STATUS         ; Bank0
      CLRF   INTCON         ; Disable interrupts and clear some flags
      CLRF   PIR1           ; Clear all flag bits
      BSF    STATUS, RP0    ; Bank1
      MOVLW  PIE1_MASK1     ; This is the initial masking for PIE1
      MOVWF  PIE1           ;
      BCF    STATUS, RP0    ; Bank0
      BSF    INTCON, GIE    ; Enable Interrupts
```

例 8-5： 用宏命令来保存 / 恢复寄存器

```
PUSH_MACRO    MACRO           ; This Macro Saves register contents
     MOVWF    W_TEMP          ; Copy W to a Temporary Register
                              ;    regardless of current bank
     SWAPF    STATUS,W        ; Swap STATUS nibbles and place
                              ;    into W register
     MOVWF    STATUS_TEMP     ; Save STATUS to a Temporary register
                              ;    in Bank0
     ENDM                     ; End this Macro
;
POP_MACRO     MACRO           ; This Macro Restores register contents
     SWAPF    STATUS_TEMP,W   ; Swap original STATUS register value
                              ;    into W (restores original bank)
     MOVWF    STATUS          ; Restore STATUS register from
                              ;    W register
     SWAPF    W_TEMP,F        ; Swap W_Temp nibbles and return
                              ;    value to W_Temp
     SWAPF    W_TEMP,W        ; Swap W_Temp to W to restore original
                              ;    W value without affecting STATUS
     ENDM                     ; End this Macro
```

例 8-6： 源文件模板

```
     LIST   p = p16C77      ; List Directive,
;    Revision History
;
     #INCLUDE    <P16C77.INC>    ; Microchip Device Header File
;
     #INCLUDE    <MY_STD.MAC>    ; Include my standard macros
     #INCLUDE    <APP.MAC>       ; File which includes macros specific
                                 ;   to this application
; Specify Device Configuration Bits
     __CONFIG    _XT_OSC & _PWRTE_ON & _BODEN_OFF & _CP_OFF & _WDT_ON
;
     org   0x00            ; Start of Program Memory
RESET_ADDR   :             ; First instruction to execute after a reset


     end
```

例 8-7： 典型的中断服务程序（ISR）

```
    org  ISR_ADDR           ;
      PUSH_MACRO            ; MACRO that saves required context registers,
                            ;   or in-line code
      CLRF   STATUS         ; Bank0
      BTFSC  PIR1, TMR1IF   ; Timer1 overflow interrupt?
      GOTO   T1_INT         ; YES
      BTFSC  PIR1, ADIF     ; NO, A/D interrupt?
      GOTO   AD_INT         ; YES, do A/D thing
      :                     ; NO, do this for all sources
      :                     ;
      BTFSC  PIR1, LCDIF    ; NO, LCD interrupt
      GOTO   LCD_INT        ; YES, do LCD thing
      BTFSC  INTCON, RBIF   ; NO, Change on PORTB interrupt?
      GOTO   PORTB_INT      ; YES, Do PortB Change thing
INT_ERROR_LP1               ; NO, do error recovery
      GOTO   INT_ERROR_LP1  ; This is the trap if you enter the ISR
                            ;    but there were no expected
                            ;    interrupts
T1_INT                      ; Routine when the Timer1 overflows
      :                     ;
      BCF     PIR1, TMR1IF  ; Clear the Timer1 overflow interrupt flag
      GOTO    END_ISR       ; Ready to leave ISR (for this request)
AD_INT                      ; Routine when the A/D completes
      :                     ;
      BCF     PIR1, ADIF    ; Clear the A/D interrupt flag
      GOTO    END_ISR       ; Ready to leave ISR (for this request)
LCD_INT                     ; Routine when the LCD Frame begins
      :                     ;
      BCF     PIR1, LCDIF   ; Clear the LCD interrupt flag
      GOTO    END_ISR       ; Ready to leave ISR (for this request)
PORTB_INT                   ; Routine when PortB has a change
      :                     ;
END_ISR                     ;
        POP_MACRO           ; MACRO that restores required registers,
                            ;   or in-line code
        RETFIE              ; Return and enable interrupts
```

8.7 设计技巧

问 1： *为什么算法没有给出正确结果？*

答 1：

假设算法是正确的，并且在算法执行期间中断已允许，对于那些既在算法中，又在中断服务程序中使用的寄存器，应确保有正确的中断现场保存和恢复。如果没有正确的中断现场保存和恢复，一些寄存器值可能会因 ISR 的执行而遭破坏。

问 2： *我的系统好象锁死了。*

答 2：

如果使用了中断，要确保在提供了中断服务后，执行 RETFIE 指令前，中断标志位被清零。在 RETFIE 指令执行后，如果中断标志位仍然保持为 1，会被误认为又是一次允许中断，程序执行立即再次进入中断服务。

8.8 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制，即可使用。目前与本章内容相关的应用笔记有：

标题	应用笔记 #
Using the PortB Interrupt On Change as an External Interrupt	AN566

8.9 版本历史

版本 A

这是描述中断的初始发行版。

第 9 章 I/O 端口

目录

本章包括下面一些主要内容：

9.1 简介 9-2
9.2 PORTA 和 TRISA 寄存器 9-4
9.3 PORTB 和 TRISB 寄存器 9-6
9.4 PORTC 和 TRISC 寄存器 9-8
9.5 PORTD 和 TRISD 寄存器 9-9
9.6 PORTE 和 TRISE 寄存器 9-10
9.7 PORTF 和 TRISF 寄存器 9-11
9.8 PORTG 和 TRISG 寄存器 9-12
9.9 GPIO 和 TRISGP 寄存器 9-13
9.10 I/O 编程注意事项 9-14
9.11 初始化 9-16
9.12 设计技巧 9-17
9.13 相关应用笔记 9-19
9.14 版本历史 9-20

9.1 简介

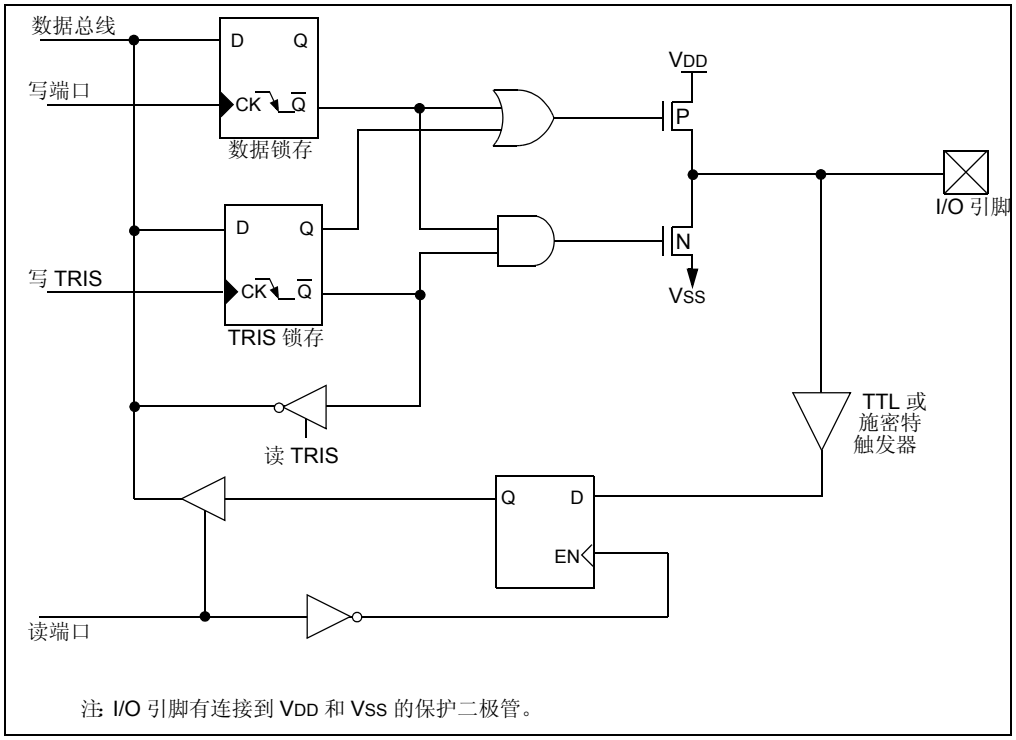
通用 I/O 引脚可看作是最简单的外设，PICmicro® 单片机通过 I/O 端口监视和控制其它设备。为了增强器件的灵活性和功能，一些引脚被定义为多功能复用引脚。这些功能由器件上相应外设的特点决定。一般来说，当相应的外设使能时，其对应的引脚不能作为通用 I/O 引脚使用。

对于大多数端口，I/O 引脚的输入输出方向，由数据方向寄存器（TRIS 寄存器）来控制。数据方向寄存器 TRIS<x> 控制 PORT<x> 的方向。当 TRIS 寄存器的某位置 “1” 时，相应引脚便为输入；当置 “0” 时，其引脚便为输出。这很好记，因为 1 很像 I（input，输入），0 很像 o（output，输出）。

PORT 寄存器可以锁存输出数据。当读 PORT 寄存器时，器件直接读 I/O 引脚上的电平（而不是内部的锁存器）。因此，在对端口执行读 - 修改 - 写入命令或引脚由输入变为输出时，应该特别小心。

图 9-1 是一个典型的 I/O 端口。它没有画出 I/O 引脚复用的外设功能。读 PORT 寄存器是读取引脚上的电平状态，而写 PORT 寄存器是将数据写入端口的数据锁存器。所有的写操作 （如 BSF 和 BCF 指令）都是读 - 修改 - 写入操作。因此，对一个端口进行写入操作意味着总是先读取端口引脚电平，再修改这个值，然后再写入端口的数据锁存器。

图 9-1： 典型的 I/O 端口



注：I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

当通用 I/O 引脚作为外设功能引脚时，其功能由外设模块决定。例如当器件复位时，模 - 数转换器 （A/D）或 LCD 驱动模块的相应多功能 I/O 引脚就变为外设功能引脚。对于模 - 数转换器（A/D），这样可以降低复位后由于 A/D 引脚上的模拟输入电压所引起的过多电流损耗。

当使能某些外设时，其相应 I/O 引脚的 TRIS 位设置将被忽略。因此，应避免使用以 TRIS 寄存器为目的寄存器的读 - 修改 - 写指令 （BSF、 BCF 和 XORWF）。用户应该查阅相应的外设章节，以确保 TRIS 位的正确设置。

PORT 引脚的多功能可能是模拟输入和模拟参考电压 VREF 输入。这些引脚是作为模拟输入还是数字 I/O，可通过对 ADCON1 寄存器（A/D 控制寄存器 1）中相应控制位的设置来决定。当设置为模拟输入时，读这些端口，其结果为 “0”。

即使将这类端口设置为模拟输入，端口引脚的输入输出方向仍然由 TRIS 寄存器控制。引脚当作模拟输入时，用户必须确保 TRIS 位置 “1”。

注 1： 如果引脚的复用功能是模拟输入，则上电复位时，这些引脚会受 ADCON1 寄存器的控制而设置为模拟输入。此时读端口的结果为 “0”。

注 2： 如果引脚的复用功能是比较器的模拟输入，则上电复位时这些引脚会受 CMCON 寄存器的控制而设置为模拟输入。此时读端口的结果为 “0”。

注 3： 如果引脚的复用功能是 LCD 显示段驱动，则上电复位时这些引脚会受 LCDSE 寄存器的控制而设置为 LCD 显示段驱动。要将引脚设置为数字端口， LCDSE 寄存器中的相应位必须清零。 LCDSE 寄存器中的任一位置 “1”，就会屏蔽该引脚 TRIS 位设置。

注 4： 引脚的复用功能也可以是并行从动端口 （PSP）。要设置为 PSP 功能， I/O 引脚必须设置为数字输入，且 PSPMODE 位必须置 “1”。

注 5： 目前，仅 PORTD 和 PORTE 可复用并行从动端口 （PSP）。当 PSPMODE 位置 “1”时，将使能 PSP。在此模式下，必须确保 TRISE 的相应位置 “1”（引脚设置为数字输入），且 PORTE 设置为数字 I/O。而此时 TRISD 寄存器的设置不会影响 PORTD 的输入输出方向。在此模式下， PORTD 和 PORTE 是 TTL 型输入缓冲。PSP 的使能位在 TRISE 寄存器中。

9.2 PORTA 和 TRISA 寄存器

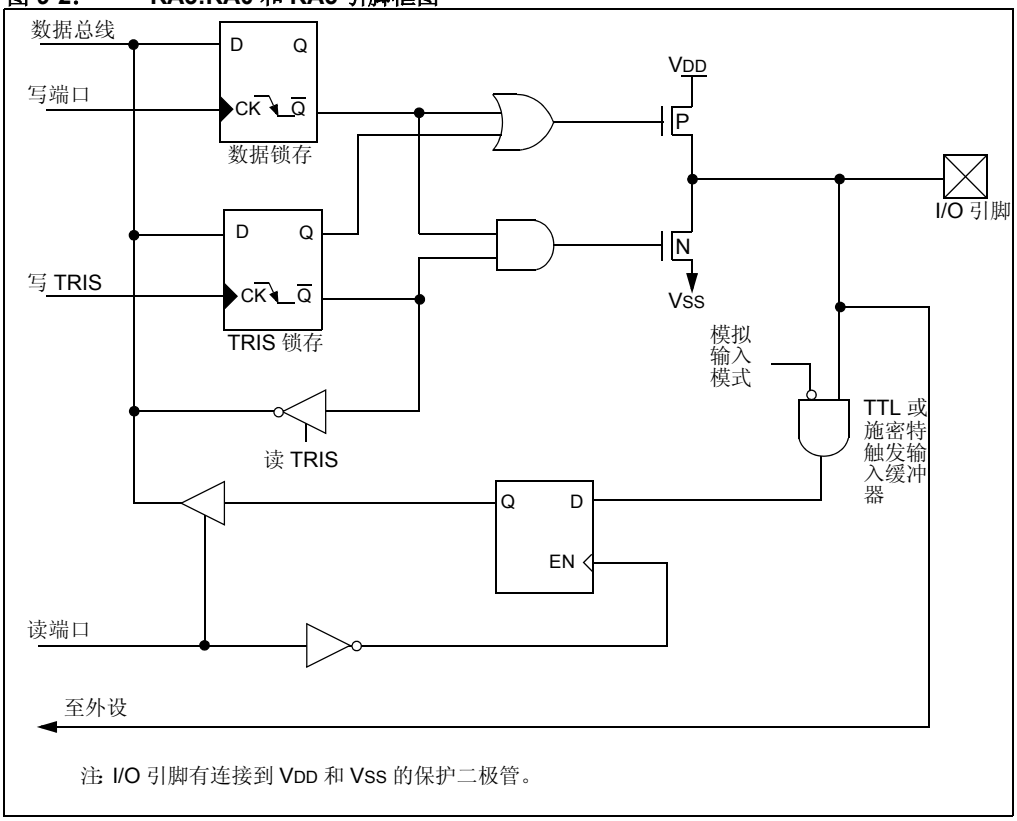
RA4引脚是施密特触发缓冲输入和漏极开路输出，而其它的RA端口引脚都是TTL逻辑电平输入和CMOS 驱动输出。所有的引脚都有数据方向位 （TRIS 寄存器）来设置输入 / 输出方向。

当 TRISA 寄存器某位置 “1” 时，其相应位的输出驱动呈高阻态；当 TRISA 寄存器某位清零，则输出锁存器中的数据就从相应引脚输出。

例 9-1： 初始化 PORTA

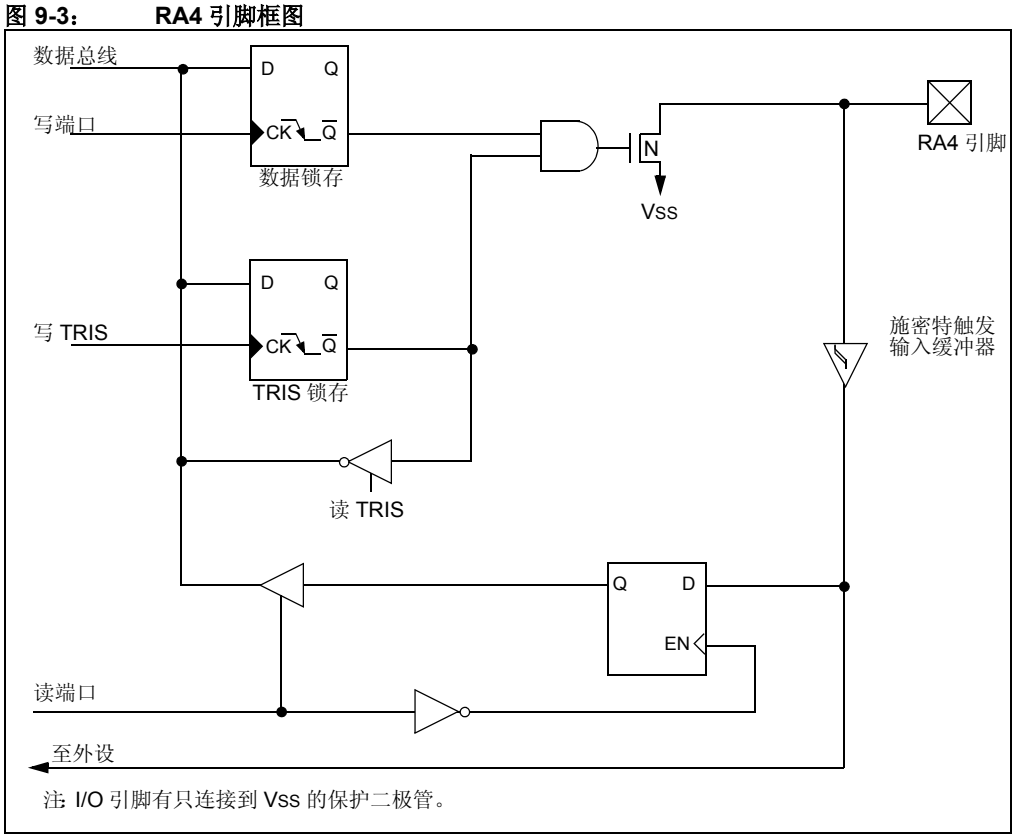
```
    CLRF    STATUS        ; Bank0
    CLRF    PORTA         ; Initialize PORTA by clearing output
                          ;    data latches
    BSF     STATUS, RP0   ; Select Bank1
    MOVLW   0xCF          ; Value used to initialize data direction
    MOVWF   TRISA         ; PORTA<3:0> = inputs PORTA<5:4> = outputs
                          ;    TRISA<7:6> always read as '0'
```

图 9-2： RA3:RA0 和 RA5 引脚框图



注：I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

图 9-3： RA4 引脚框图



注：I/O 引脚有只连接到 Vss 的保护二极管。

9.3 PORTB 和 TRISB 寄存器

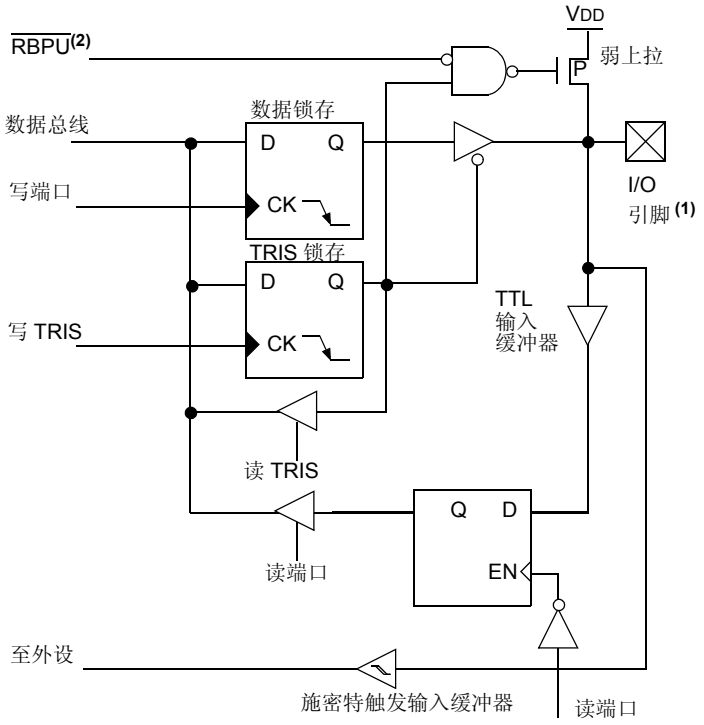
PORTB 是一个 8 位的双向端口，TRISB 是 PORTB 对应的数据方向寄存器。当 TRISB 寄存器的某一位置为“1”时， PORTB 相应引脚的输出驱动呈高阻态输入模式。当 TRISB 寄存器的某一位清零，则输出锁存器上的数据就从相应引脚输出。

例 9-2： 初始化 PORTB

```
    CLRF    STATUS       ; Bank0
    CLRF    PORTB        ; Initialize PORTB by clearing output
                         ;    data latches
    BSF     STATUS, RP0  ; Select Bank1
    MOVLW   0xCF         ; Value used to initialize data direction
    MOVWF   TRISB        ; PORTB<3:0> = inputs, PORTB<5:4> = outputs
                         ;    PORTB<7:6> = inputs
```

PORTB 的每个引脚都有内部弱上拉电路。只要对控制位 $\overline{\text{RBPU}}$（OPTION<7>）清零，就可以开启所有引脚的弱上拉功能。当 PORTB 端口的引脚设置为输出时，其弱上拉电路会自动切断。在上电复位后，会禁止弱上拉功能。

图 9-4： RB3:RB0 引脚框图



注 1：I/O 引脚有连接到 VDD 和 VSS 的保护二极管。
2：若要使能某引脚的弱上拉功能，需将相应的 TRIS 位置“1”，并把 $\overline{\text{RBPU}}$ 位清零（OPTION<7>）。

PORTB的RB7:RB4引脚被设置为输入时，若引脚电平有变化，会可以产生中断（即，当RB7:RB4的任何一个引脚被设置为输出时，该引脚不再具有电平变化的中断功能）。该功能的实现过程为：当前 RB7:RB4 引脚上的输入电平与前次从 PORTB 读入锁存器的旧值进行比较（异或运算），若有变化，则将 RBIF 标志位 （INTCON<0>）置 “1”，产生 RB 端口电平变化中断。

该中断可以唤醒器件。在中断服务程序中，用户可以采用以下方式来清除中断请求：

a) 对 PORTB 进行读 / 写操作。这将结束引脚电平变化的情况。

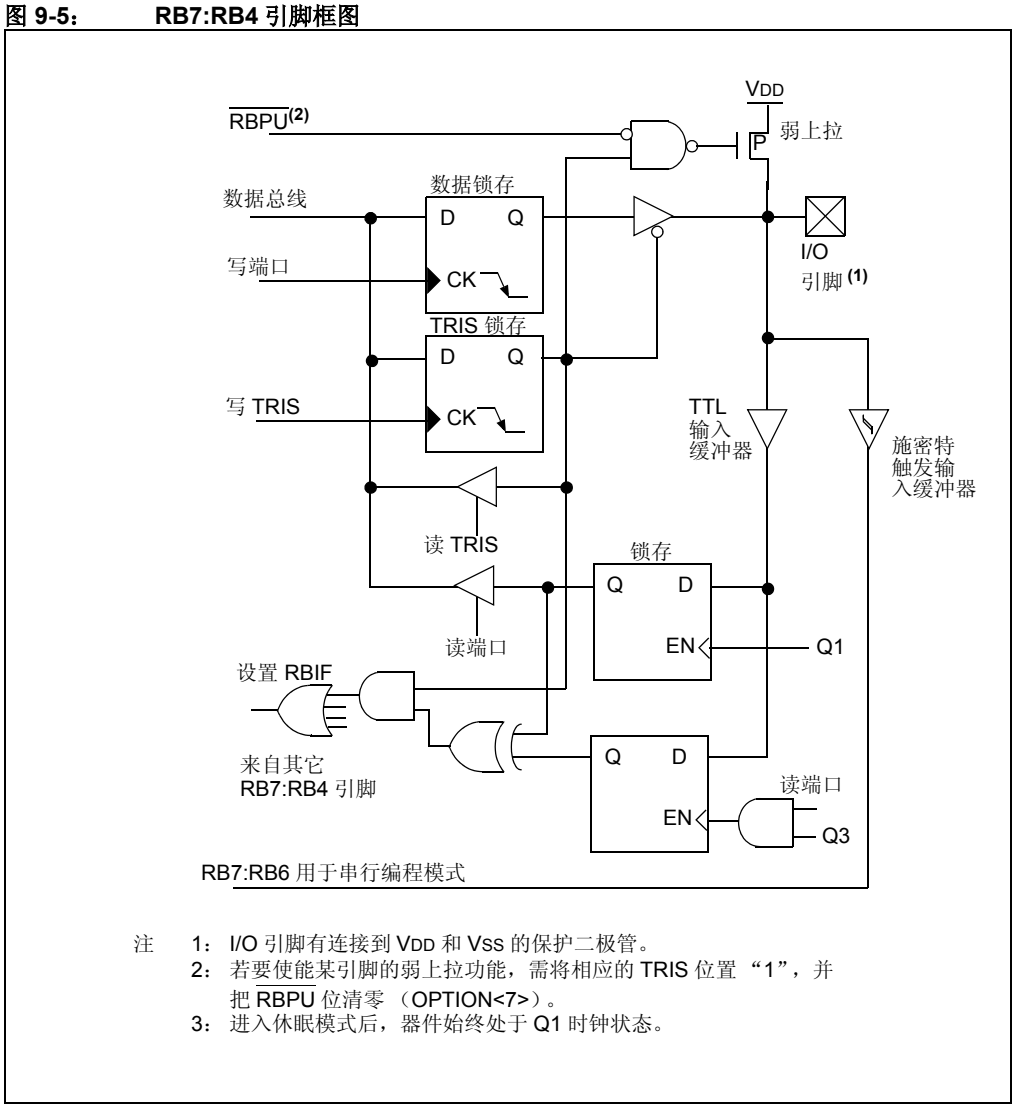
b) RBIF 标志位清零。

引脚上电平变化的情况会一直不断地将 RBIF 标志位置 “1”。而对 PORTB 进行读操作，将结束引脚电平变化的情况，才可以真正将 RBIF 标志位清零。

利用 RB7:RB4 引脚的电平变化中断功能和软件可配置的上拉功能，可以很容易地与键盘连接，实现按键唤醒功能。

对于按键唤醒以及其它需要用到 PORTB 电平变化中断功能的操作，都可以用该电平变化中断来实现。在使用电平变化中断功能时，不需要软件不断地查询 PORTB 的状态。

图 9-5： RB7:RB4 引脚框图



注 1： I/O 引脚有连接到 VDD 和 VSS 的保护二极管。
2： 若要使能某引脚的弱上拉功能，需将相应的 TRIS 位置 “1”，并把 RBPU 位清零 （OPTION<7>）。
3： 进入休眠模式后，器件始终处于 Q1 时钟状态。

9.4 PORTC 和 TRISC 寄存器

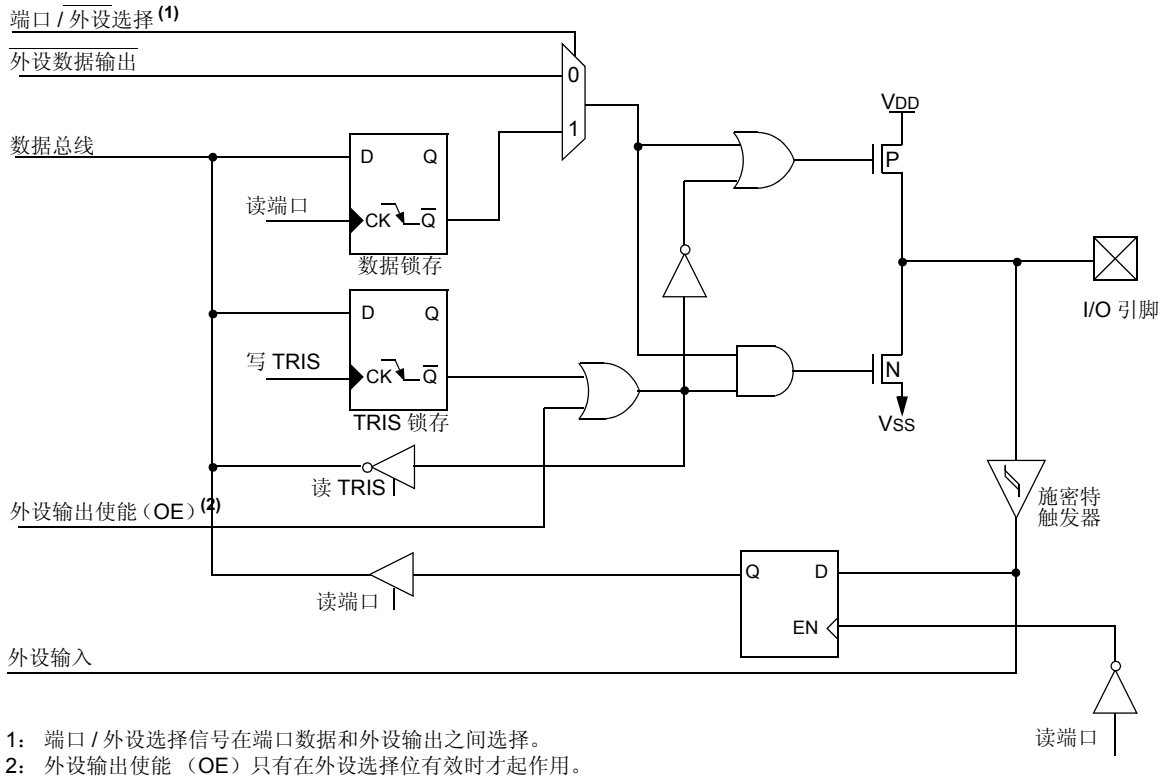
PORTC 是一个 8 位的双向端口。利用 TRISC 寄存器可将各引脚分别设置为输入或输出。PORTC 引脚都有施密特触发输入缓冲器。

当外设功能使能时，应考虑到每个 PORTC 引脚的 TRIS 位方向设置。有些外设使能时，会超越相应引脚的 TRIS 位方向设置而将引脚直接定义为输出；而另一些外设使能时，也会超越相应引脚的 TRIS 方向设置，但却将引脚直接定义为输入。

例 9-3： 初始化 PORTC

```
    CLRF    STATUS       ; Bank0
    CLRF    PORTC        ; Initialize PORTC by clearing output
                         ;    data latches
    BSF     STATUS, RP0  ; Select Bank1
    MOVLW   0xCF         ; Value used to initialize data direction
    MOVWF   TRISC        ; PORTC<3:0> = inputs, PORTC<5:4> = outputs
                         ;    PORTC<7:6> = inputs
```

图 9-6： PORTC 框图 （外设输出时，可超越 TRIS 位设置）



注 1：端口 / 外设选择信号在端口数据和外设输出之间选择。
2：外设输出使能 （OE）只有在外设选择位有效时才起作用。
3：I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

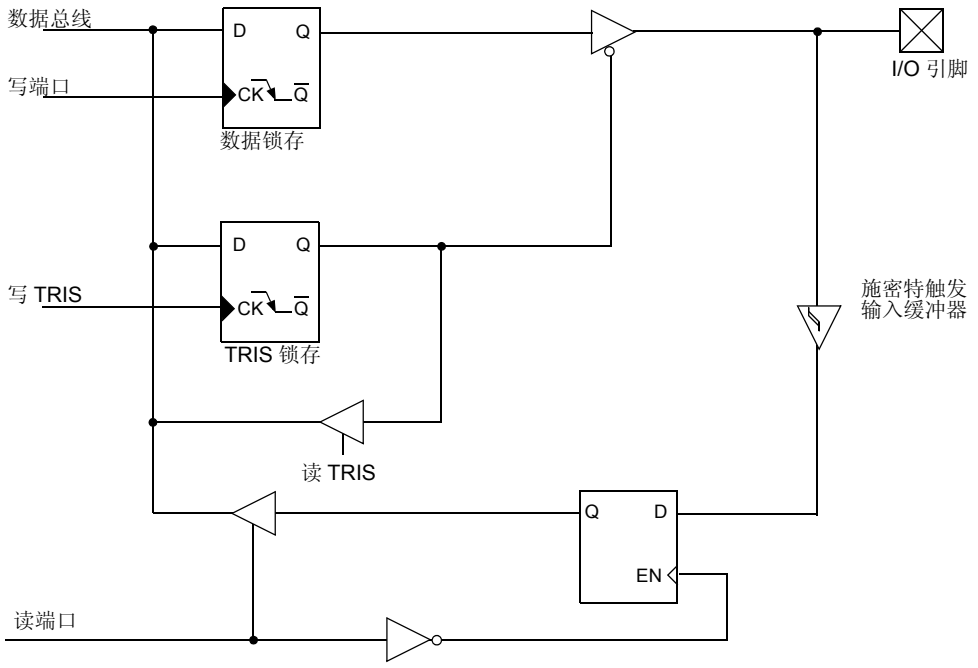
9.5 PORTD 和 TRISD 寄存器

PORTD 是一个带有施密特触发输入缓冲的 8 位双向端口。各引脚都可以被分别设置为输入或输出。

例 9-4： 初始化 PORTD

```
    CLRF    STATUS       ; Bank0
    CLRF    PORTD        ; Initialize PORTD by clearing output
                         ;    data latches
    BSF     STATUS, RP0  ; Select Bank1
    MOVLW   0xCF         ; Value used to initialize data direction
    MOVWF   TRISD        ; PORTD<3:0> = inputs, PORTD<5:4> = outputs
                         ;    PORTD<7:6> = inputs
```

图 9-7： 典型的 PORTD 框图 （I/O 端口模式）



注：I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

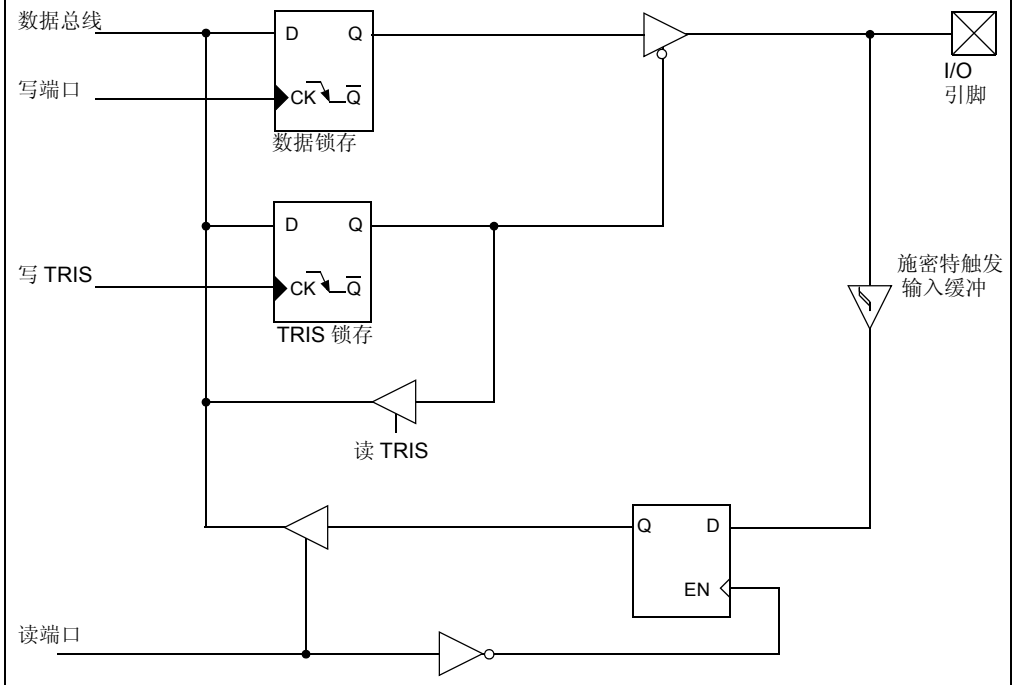
9.6 PORTE 和 TRISE 寄存器

PORTE 是一个带有施密特触发输入缓冲的端口，其引脚数最多可以有 8 个。各引脚都可以被分别设置为输入或输出。

例 9-5： 初始化 PORTE

```
    CLRF    STATUS       ; Bank0
    CLRF    PORTE        ; Initialize PORTE by clearing output
                         ;    data latches
    BSF     STATUS, RP0  ; Select Bank1
    MOVLW   0x03         ; Value used to initialize data direction
    MOVWF   TRISE        ; PORTE<1:0> = inputs, PORTE<7:2> = outputs
```

图 9-8： 典型的 PORTE 框图 （I/O 端口模式）



注：I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

注： 对于带 PORTE 的器件，TRISE 寄存器的高位用于并行从动端口的控制和状态表示。

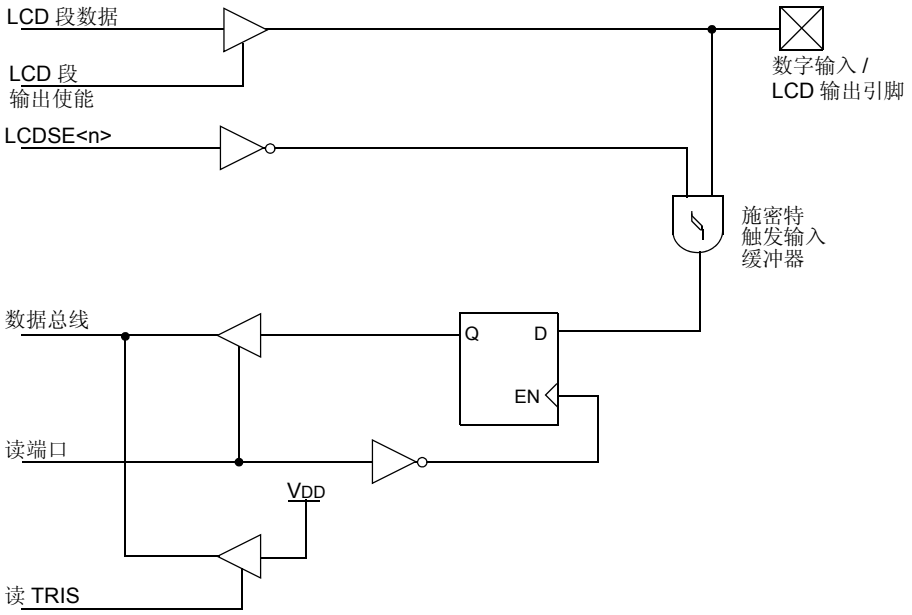
9.7 PORTF 和 TRISF 寄存器

PORTF只是数字输入端口，而且每个引脚都和LCD段驱动功能复用。这些引脚都有施密特触发输入缓冲器。

例 9-6： 初始化 PORTF

```
    BCF   STATUS, RP0    ; Select Bank2
    BSF   STATUS, RP1    ;
    BCF   LCDSE, SE16    ; Make all PORTF
    BCF   LCDSE, SE12    ;    digital inputs
```

图 9-9： PORTF LCD 框图



注：I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

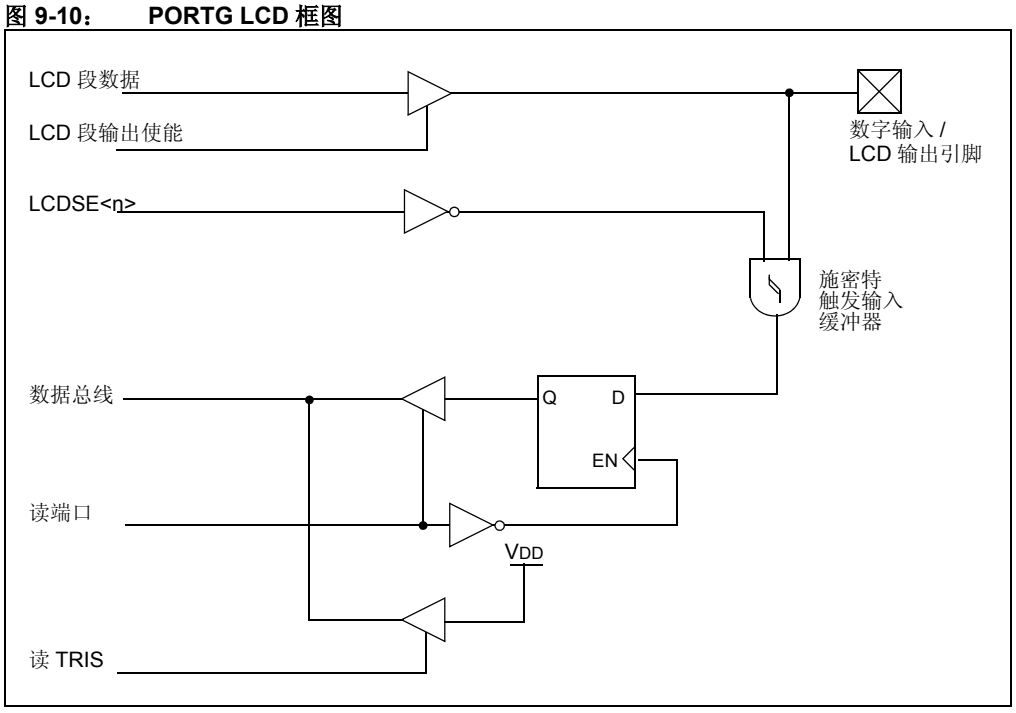
9.8 PORTG 和 TRISG 寄存器

PORTG 只是数字输入端口，而且每个引脚都和 LCD 段驱动功能复用。这些引脚都有施密特触发输入缓冲器。

例 9-7： 初始化 PORTG

```
    BCF   STATUS, RP0    ; Select Bank2
    BSF   STATUS, RP1    ;
    BCF   LCDSE, SE27    ; Make all PORTG
    BCF   LCDSE, SE20    ;    and PORTE<7> digital inputs
```

图 9-10： PORTG LCD 框图



9.9 GPIO 和 TRISGP 寄存器

GPIO 是一个 8 位 I/O 寄存器。只有低 6 位（GP5:GP0）可用，第六和第七位未用且读为“0”。各 GPIO 引脚（除了 GP3 外）都可以通过程序分别设置为输入或者输出， GP3 引脚只能是输入引脚。

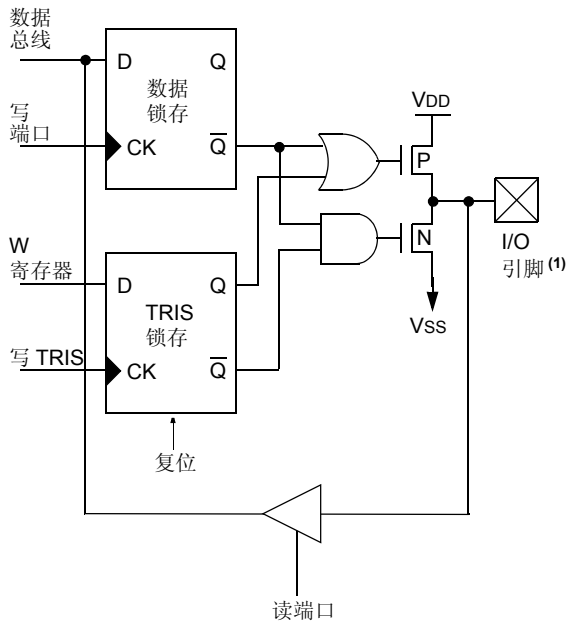
TRISGP 寄存器控制 GPIO 引脚的数据输入输出方向。TRISGP 寄存器的某位置“1”，使相应的输出驱动变成高阻态，而置“0”，则输出数据锁存器中的数据就从相应引脚输出。 GP3 引脚是特殊情况，它只能作为输入引脚，它的 TRIS 位始终读为“1”。复位时， TRISGP 寄存器的所有位均为“1”，使所有引脚都是输入引脚。

读 GPIO 端口实际是读引脚电平而非输出数据锁存器。输入数据应一直有效，直到执行了输入指令（如 MOVF GPIO,W）。输出将被锁存并且保持不变，直到重新写输出锁存器。

例 9-8： 初始化 GPIO

```
    CLRF    STATUS          ; Bank0
    CLRF    GPIO            ; Initialize GPIO by clearing output
                            ;   data latches
    BSF     STATUS, RP0     ; Select Bank1
    MOVLW   0xCF            ; Value used to initialize data direction
    MOVWF   TRISGP          ; GP<3:0> = inputs GP<5:4> = outputs
                            ;   TRISGP<7:6> always read as '0'
```

图 9-11： GP5:GP0 引脚框图



注 1: I/O 引脚有连接到 VDD 和 VSS 的保护二极管。
GP3 仅作为输入，没有数据锁存器和输出驱动器。

通过配置字可以选择 I/O 端口的复用功能。当引脚设为复用功能时，读端口的结果为“0”。GP0、GP1 和 GP3 引脚有弱上拉功能和电平变化中断功能。电平变化中断功能和弱上拉功能不能针对引脚单独使能或禁止。INTCON<3> 位置“1”，可使能电平变化中断功能。如果器件配置位选择了一种外部振荡器模式，GP4和GP5引脚的GPIO功能就被禁止，这两个引脚被用作振荡器引脚。

9.10 I/O 编程注意事项

使用端口及 GPIO 作为 I/O 时，为确保操作按预想的进行，需要考虑一些设计注意事项。

9.10.1 双向 I/O 口

任何写操作指令实际上都是先执行一个读操作，再执行一个写操作。例如，BCF 和 BSF 指令，先读寄存器的值到 CPU，然后执行位操作，最后将结果写回寄存器。当一个端口既有输入引脚又有输出引脚时，对其的操作必须加倍小心。例如，对 PORTB 的 bit5 执行 BSF 操作时，先将 PORTB 的全部 8 位数值读入 CPU，然后将 bit5 位置 “1”，最后将 PORTB 的结果写回输出锁存器。如果另一个 PORTB 的引脚是作为双向 I/O 引脚（如：bit0），而且此时引脚定义为输入，则该引脚当前的输入信号被读入 CPU，然后将值重新写入该特定引脚的数据锁存器，覆盖先前的内容。只要该引脚一直是输入模式，就毫无问题。然而，如果 bit0 随后变成输出引脚，其数据锁存器的内容已经改变而无法得知。

对端口寄存器的读操作，是读端口引脚的电平值。对端口寄存器的写操作是向端口锁存器写值。对一个端口使用读 - 修改 - 写指令（如：BCF 和 BSF 等）时，首先读入端口引脚的数值，然后对读入值执行指定的操作，最后将操作结果写入端口锁存器。

例 9-9 显示了对一个 I/O 端口执行两个连续的读 – 修改 – 写指令的情况。

例 9-9： 对 I/O 端口执行读 - 修改 - 写指令

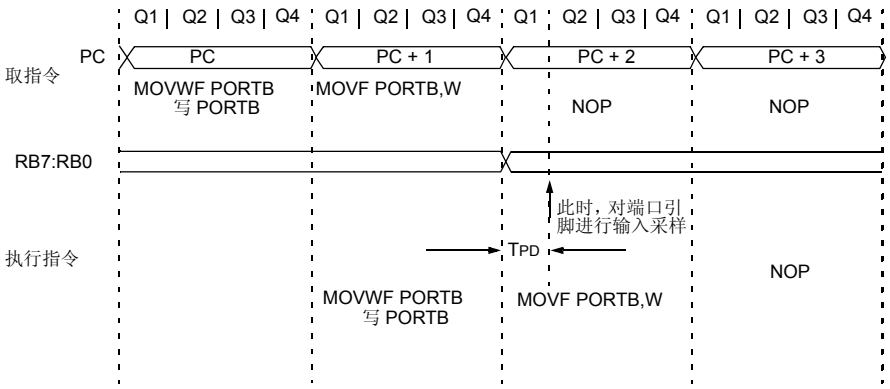
```
; Initial PORT settings: PORTB<7:4> Inputs
;                        PORTB<3:0> Outputs
; PORTB<7:6> have external pull-ups and are not connected to other circuitry
;
;                       PORT latch  PORT pins
;                       ----------  ---------
    BCF  PORTB, 7     ; 01pp pppp    11pp pppp
    BCF  PORTB, 6     ; 10pp pppp    11pp pppp
    BSF  STATUS, RP0  ;
    BCF  TRISB, 7     ; 10pp pppp    11pp pppp
    BCF  TRISB, 6     ; 10pp pppp    10pp pppp
;
; Note that the user may have expected the pin values to be 00pp ppp.
; The 2nd BCF caused RB7 to be latched as the pin value (high).
```

当某引脚设置为输出时，不应通过外部器件另加电平来驱动该引脚的电平高低，以达到改变其电平（“线或”或“线与”）的目的。不然，短路的高输出电流有可能损坏器件。

9.10.2 I/O 端口的连续操作

对 I/O 端口的写操作实际发生在指令周期的末尾时刻，但对于读操作，在指令周期的开始处，所读的数据就必须有效（图 9-12）。因此，如果对同一个 I/O 端口进行写操作之后，接着执行读操作，此时要特别注意。指令的执行顺序应该是：等引脚电压达到稳定（与负载有关）后，才执行下一个指令将端口值读入 CPU。否则，读入的可能是引脚的前一个状态而不是新状态。当状态不确定时，最好用一个 NOP 指令或者其它不访问该 I/O 端口的指令隔开这些指令。

图 9-12： 连续 I/O 端口操作



此例为对 PORTB 写操作后，接着读 PORTB。

注： 数据建立时间 = (0.25TCY - TPD)
其中 TCY = 指令周期
TPD = 传输延迟

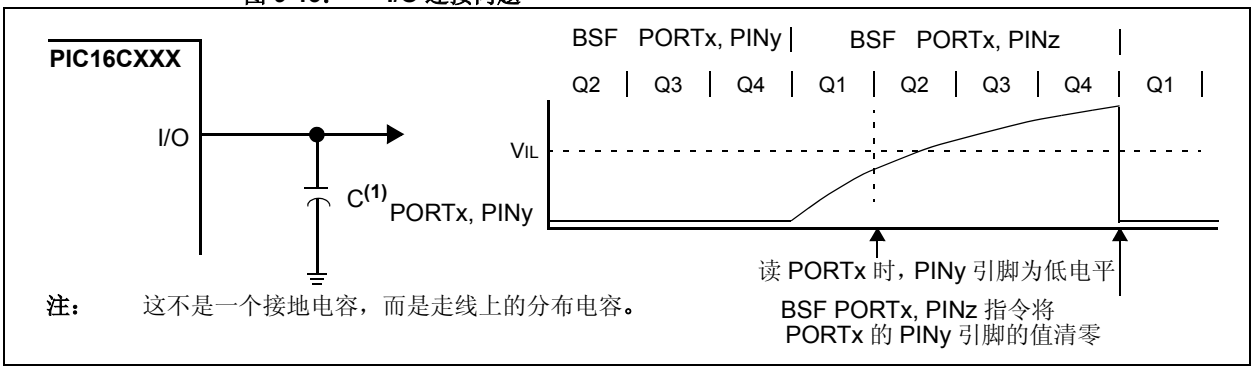
因此，当时钟频率很高时，由于外部电容的影响，写端口后马上再读端口可能会出现问题。

图 9-13 所示为这类情况的 I/O 模型。随着等效电容（C）变大，I/O 引脚的上升或下降时间也将延长。随着器件工作频率或等效电容的增加，对 PORTx 端口连续执行读 – 修改 – 写指令出现问题的可能性也随之增加。该等效电容包括印制板走线的分布电容。

解决该问题的最好办法就是在 I/O 引脚上串连一个电阻。该电阻可以使 I/O 引脚在下一条指令执行前很快达到预期值。

在对 PORTx 连续执行的读 – 修改 – 写指令之间加入 NOP 指令，是解决该问题的一种低成本方法。由等效电容 C 和器件的工作频率决定 NOP 指令的数量。

图 9-13： I/O 连接问题



注： 这不是一个接地电容，而是走线上的分布电容。

9.11 初始化

参见前述各端口的初始化例子。

注：	在初始化端口时，建议首先初始化数据锁存器 （PORT 寄存器），然后是数据方向寄存器 （TRIS）。这样会消除由于上电时端口数据锁存器的状态不确定性，所引起的误操作。

9.12 设计技巧

问 1： *振荡器工作正常，但程序不能改变任何 I/O 端口设置。是哪里出错了？*

答 1：

1. 是否正确初始化了方向寄存器 TRIS？在 BANK1 中可以直接对方向寄存器进行设置。但是大部分情况下，用户在对 TRIS 寄存器写零前，并没有切换到 BANK1。BSF STATUS,RP0 指令，切换到 BANK1。
2. 如果您在 BANK1（即 RP0 = 1）中正确地设置了 TRIS 寄存器，那么在对端口进行写操作前，您可能并没有返回 BANK0。返回 BANK0，可使用 BCF STATUS,RP0 指令。
3. 是否使能了引脚的外设复用功能？
4. 是否在器件烧写编程时，使能了看门狗定时器？如果是，应至少每隔 9 ms 使用 CLRWDT 指令将看门狗定时器清零一次。如果看门狗定时器使用了预分频器，那么时间间隔可适当加长。
5. 您是否使用了正确的指令来写端口？当应该使用 MOVWF 指令时，不止一个人都用成了 MOVF 指令。
6. 如果有中断处理，是否禁止了中断响应？如果没有，可以先禁止中断响应来验证一下是不是中断响应造成的影响。

问 2： *当程序读一个端口时，得到的值与端口寄存器的值不同。这是什么原因？*

答 2：

1. 无论端口是设置成输入还是输出，读端口都是读端口的引脚值。所以，如果一个引脚被设置成输入，无论寄存器的值为多少，读的都是引脚的值而不是寄存器的值。
2. 一个引脚被设置成输出时，假设相应的数据锁存器值为 1。如果该引脚接地短路，那么读引脚的值始终为零。这对于建立容错系统非常有用，也利于处理 I^2C 总线冲突：在实现 I^2C 协议时，对 I^2C 总线只驱动输出低电平，而要输出高电平时，将引脚置为三态。如果引脚为低电平而您却没有驱动引脚输出，此时其它器件就会试图占用总线。
3. 所有中档系列的单片机都至少有一个漏极开路（或集电极开路）引脚。这类引脚只能输出一个低电平或三态。对大多数中档单片机来说，该引脚是 RA4。漏极开路引脚需要外接上拉电阻，才能输出高电平。这个引脚对于驱动需非标准电源的负载非常有用。上拉电阻可以连接到一个电源（其电压一般小于 V_{DD}），此时引脚就可以输出高电平。

问 3： *PIC16CXXX 的 RB0 引脚被设置成中断输入，但却无法产生中断。当我改变程序来查询该引脚时，读到值为高电平而且运行良好。这是什么原因？*

答 3：

PORTB 在多数情况下是 TTL 电平输入的，所以当有一个 3V 电平输入（V_{DD} = 5V）时，读端口的结果将为 1。然而 RB0 引脚的中断缓冲器输入是施密特触发输入，它的高电平输入要求电压比 TTL 的高电平输入要求电压要高。所以读 RB0 引脚为 1，但却可能无法产生中断。RB0 引脚中断采用具有滞后效应的施密特触发器输入，可以降低噪声。引脚上输入的短噪声脉冲可能使数据产生错误，这与噪声引起中断不是一回事。

I^2C is a trademark of Philips Corporation.

问 4： *当我执行一条 BCF 指令时，该端口的其它引脚都清零了，为什么？*

答 4：

1. 另举一个读-修改-写指令意外改变其它引脚值的例子：假设PORTC全部为输出，而且引脚输出为低电平。在每一个端口引脚上有一接地的 LED，这样引脚输出高电平时将点亮 LED。 LED 旁并联了一个 100 µF 的电容。同样假设单片机的运行速度非常快，比方说 20 MHz。现在依次将各个引脚的输出置“1”：BSF PORTC,0，然后 BSF PORTC,1，再 BSF PORTC,2 等等，此时只有最后一个引脚被置“1”而且仅点亮最后一个 LED。这是因为电容充电需要时间。当某个引脚置 “1” 时，其前一个引脚并没有完成充电，所以对前一个引脚的读结果为零。这个零被写回端口锁存器（记住：执行的是读－修改－写指令），而使之前要置 “1” 的位清零。对端口进行高速连续地操作时，这通常只是一种可能，但是它也会发生，所以也应将其考虑在内。
2. 如果使用的器件是 PIC16C7XX，那么您可能没有在 ADCON1 寄存器中正确地设置 I/O 引脚。如果一个引脚设置成模拟输入，那么无论引脚的电压为多少，读该引脚的结果均为零。一般情况下，读的都是引脚上的输入状态，但这是一个特例。您可以通过 TRIS 寄存器将模拟输入引脚设置成输出，然后对其执行写操作来驱动该引脚输出高或低电平，但是读该引脚时结果始终为零。因此执行读 - 修改 - 写指令 （参见前面的问题）时，所有模拟引脚均读为零，这些未被指令修改的读入值可以在端口锁存器中重新写为零。对于设置成模拟输入的引脚，其引脚输入值可能既不是逻辑高电平也不是逻辑低电平，也不是浮动的。数字引脚应禁止浮动输入，因为它可能导致输入缓冲器较大的工作电流，因此此时数字电平的输入缓冲器被禁止了。

9.13 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制，即可使用。目前与 I/O 端口相关的应用笔记有：

标题	应用笔记 #
Improving the Susceptibility of an Application to ESD	AN595
Clock Design using Low Power/Cost Techniques	AN615
Implementing Wake-up on Keystroke	AN528
Interfacing to AC Power Lines	AN521
Multiplexing LED Drive and a 4 x 4 Keypad Sampling	AN529
Using PIC16C5X as an LCD Drivers	AN563
Serial Port Routines Without Using TMR0	AN593
Implementation of an Asynchronous Serial I/O	AN510
Using the PORTB Interrupt on Change Feature as an External Interrupt	AN566
Implementing Wake-up on Keystroke	AN522
Apple Desktop Bus	AN591
Software Implementation of Asynchronous Serial I/O	AN555
Communicating with the I^2C^{TM} Bus using the PIC16C5X	AN515
Interfacing 93CX6 Serial EEPROMs to the PIC16C5X Microcontrollers	AN530
Logic Powered Serial EEPROMs	AN535
Interfacing 24LCXXB Serial EEPROMs to the PIC16C54	AN567
Using the 24XX65 and 24XX32 with Stand-alone PIC16C54 Code	AN558

9.14 版本历史

版本 A

这是描述 I/O 端口的初始发行版。

第 10 章 并行从动端口

目录

本章包括以下一些主要内容：

10.1 简介 10-2
10.2 控制寄存器 10-3
10.3 操作 10-4
10.4 休眠模式下的操作 10-5
10.5 复位的影响 10-5
10.6 PSP 波形 10-5
10.7 设计技巧 10-6
10.8 相关应用笔记 10-7
10.9 版本历史 10-8

10.1 简介

有些器件具有 8 位的并行从动端口 （PSP），它与器件的一个 I/O 端口复用。该端口可作为一个 8 位并行从动端口，而当控制位 PSPMODE 置 “1” 时，可作为一个普通微处理器端口。此时，输入缓冲器是 TTL 型。

在并行从动模式时，通过 $\overline{RD}$ 和 $\overline{WR}$ 控制输入引脚可由外部实现异步读写。

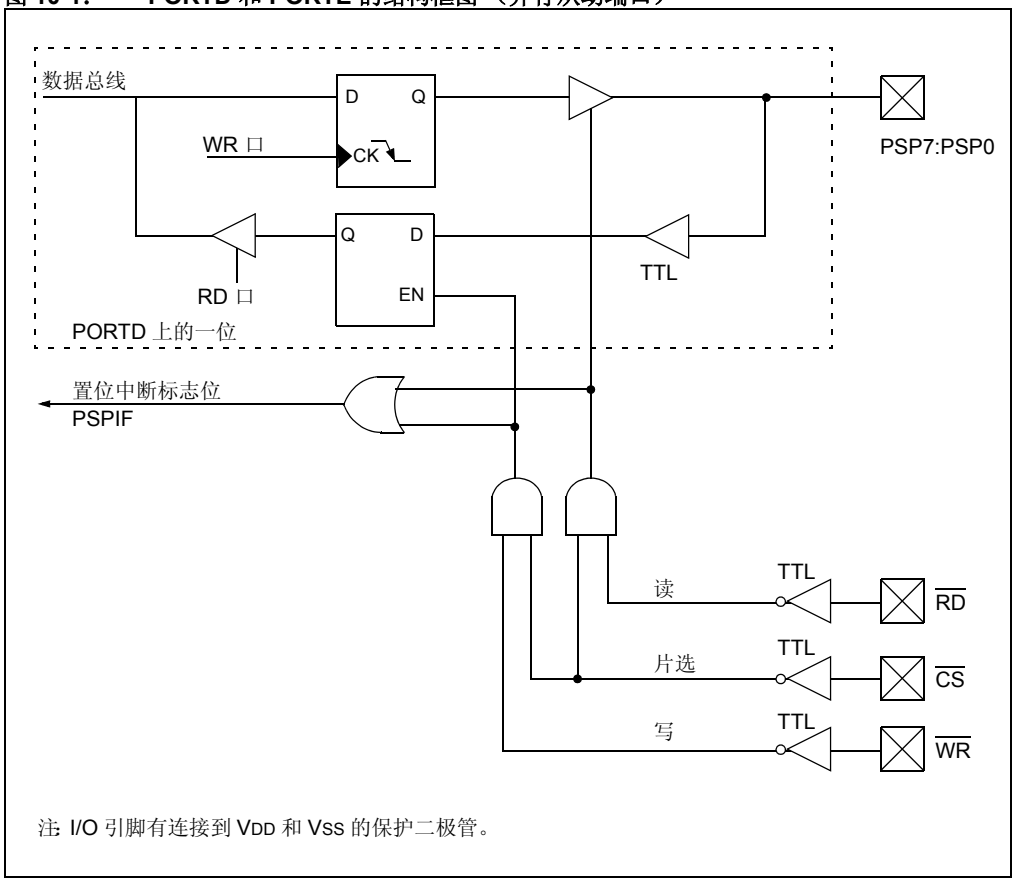
并行从动端口可直接与 8 位的微处理器数据总线相连。外部处理器能够读写 8 位端口锁存器。将 PSPMODE 位置 “1”，使端口相应引脚分别成为 $\overline{RD}$ 输入、 $\overline{WR}$ 输入和 $\overline{CS}$ （片选）输入。

注 1：	目前只有 PORTD 和 PORTE 能复用为并行从动端口 （PSP）。 PSPMODE 位置 “1” 时，端口即被使能为微处理器端口。在这种模式下，用户必须确保 PORTD 和 PORTE 被配置为数字 I/O 口。也就是说，如果存在与 PSP 功能复用的外设模块（如 A/D），会被禁止。 当 PORTE 配置为数字 I/O 口时， PORTD 将忽略 TRISD 寄存器的设置。
注 2：	此时，PORTD 和 PORTE 的输入缓冲器是 TTL 型。TRISE 寄存器控制 PSP 的操作。

并行从动端口实际上有两个 8 位锁存器，一个用作数据输出（来自 PICmicro® 单片机），而另一个用作数据输入。用户将 8 位数据写入端口的数据锁存器，而从动口的引脚锁存器读取数据（注意：这两个锁存器地址相同）。在从动端口模式下，忽略 TRIS 寄存器，因为微处理器在控制着数据流的方向。

图 10-1 是 PSP 的结构框图。

图 10-1： PORTD 和 PORTE 的结构框图 （并行从动端口）



注: I/O 引脚有连接到 VDD 和 VSS 的保护二极管。

10.2 控制寄存器

寄存器 10-1： TRISE 寄存器

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

bit 7 **IBF**：输入缓冲器满状态位

1 = 接收到一个数据，等待 CPU 读取
0 = 未接收到任何数据

bit 6 **OBF**：输出缓冲器满状态位

1 = 输出缓冲器仍保存着上一次写入的数据
0 = 已读取输出缓冲器

bit 5 **IBOV**：输入缓冲器溢出检测位 （在微处理器模式）

1 = 在尚未读取上一次输入数据前发生了一次写入
（必须用软件清零）

0 = 无溢出发生

bit 4 **PSPMODE**：并行从动端口模式选择位

1 = 并行从动端口模式
0 = 通用 I/O 口模式

bit 3 **未用位**：读作 “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

图注	
R = 可读位	W = 可写位
U = 未用位，读作 “0”	- n = 上电复位时的值

10.3 操作

当检测到 $\overline{CS}$ 和 $\overline{WR}$ 都为低电平时，从外部系统向 PSP 进行写入操作。当 $\overline{CS}$ 或 $\overline{WR}$ 变成高电平（边沿触发）时，输入缓冲器满标志位 IBF（TRISE<7>）在下一个 Q2 时钟节拍之后的 Q4 时钟节拍置“1”。中断标志位 PSPIF 在同一个 Q4 时钟节拍也置为“1”。IBF 标志位置“1”后，不能马上清零，要延迟几个指令周期（参见参数 66）。如果 IBF 标志位是通过读取 PORTD 输入锁存器清零的，那么只能用只读指令（如 MOVF），而不是读 - 修改 - 写指令来读取。如果在上次写入的数据还未从缓冲器中被读出的情况下，试图向并行从动端口进行第二次写操作，输入缓冲溢出标志位 IBOV（TRISE<5>）就置为“1”。

当检测到 $\overline{CS}$ 和 $\overline{RD}$ 为低电平时，由外部系统向 PSP 进行读取操作。输出缓冲满标志位 OBF（TRISE<6>）被立即清零，表明外部总线读取了 PORTD 锁存器。当 $\overline{CS}$ 或 $\overline{RD}$ 引脚变成高电平（边沿触发）时，中断标志位 PSPIF 在下一个 Q2 时钟节拍之后的 Q4 时钟节拍置“1”，表明读操作完成。OBF 位将保持低电平，直至由用户固件将数据写入 PORTD 锁存器。

当接收到的数据等待 CPU 读取时，输入缓冲满标志位 IBF 置为“1”。一旦读取了端口输入锁存器，就将 IBF 位清零。当写入端口锁存器的数据等待外部总线读取时，输出缓冲满标志位 OBF 置为“1”。一旦微处理器读取了 PORTD 输出锁存器，就将 OBF 位清零。如果在上次写入的数据还未被读取的情况下，试图向微处理器端口进行第二次写操作，输入缓冲溢出状态标志位 IBOV 就置为“1”。

当不在并行从动端口模式时，IBF 和 OBF 位应保持清零状态。不过，如果标志位 IBOV 先前被置“1”，必须用软件将其清零。

完成一个读写操作后，会产生一个中断并将标志位 PSPIF 置“1”。中断标志位 PSPIF 必须通过用户软件清零。将中断使能位 PSPIE 清零，便禁止了 PSP 中断响应。

表 10-1： PORTE 引脚功能

名称	功能
$\overline{RD}$	并行从动端口模式下读控制输入位： $\overline{RD}$ 1 = 无读操作 0 = 读操作。读 PORTD 寄存器（若片选有效）
$\overline{WR}$	并行从动端口模式下写控制输入位： $\overline{WR}$ 1 = 无写操作 0 = 写操作。写入 PORTD 寄存器（若片选有效）
$\overline{CS}$	并行从动端口模式下片选控制输入位： $\overline{CS}$ 1 = 器件没有被选择 0 = 器件被选择

注： 具有 PSP 功能的引脚可能还复用其它功能。在 PSP 操作时，必须将引脚配置为数字 I/O。

10.4 休眠模式下的操作

在休眠模式下，微处理器仍可读写并行从动端口。读写操作会将 PSPIF 位置“1”。如果 PSP 中断被使能，读写操作还会唤醒处理器，从而可以读取 PSP 数据锁存器或将下一个值写入锁存器中，以供微处理器使用。

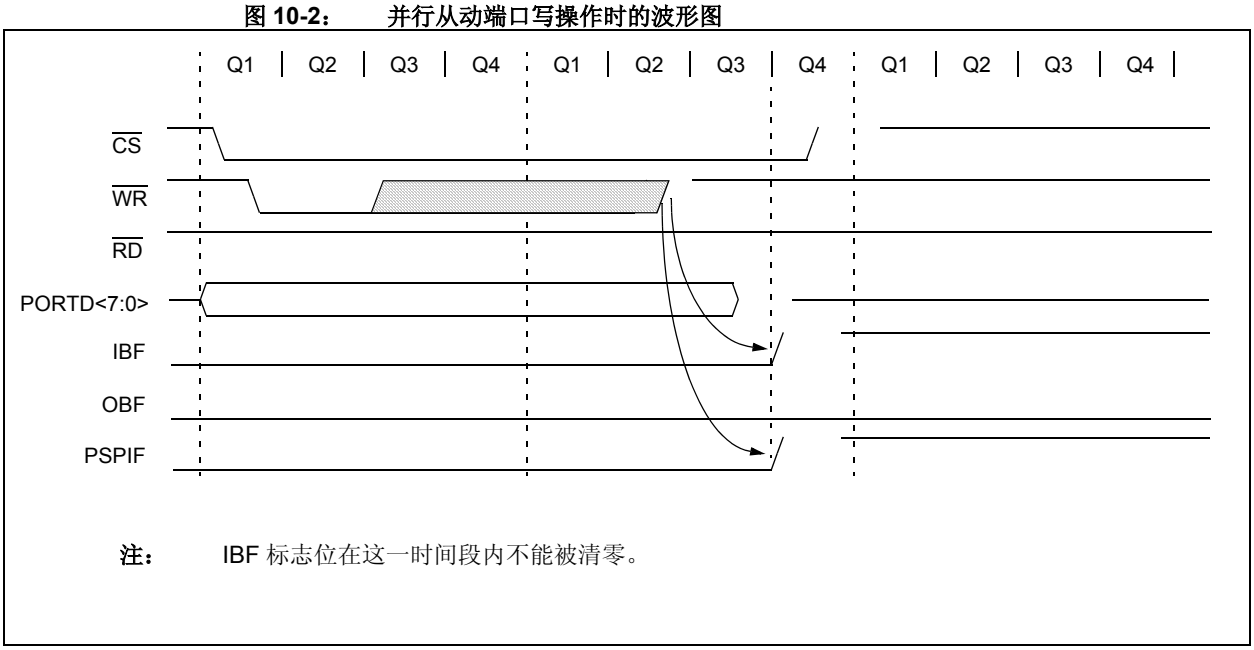
10.5 复位的影响

复位后， PSP 被禁止，并强制 PORTD 和 PORTE 为缺省模式。

10.6 PSP 波形

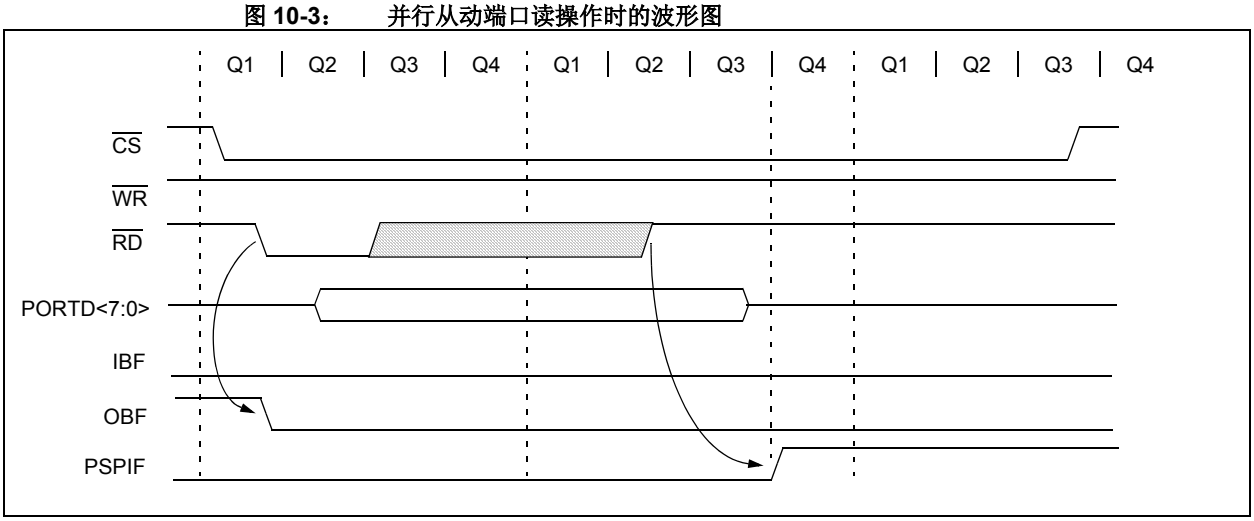
图 10-2 是微处理器向 PSP 写入数据时的波形图，而图 10-3 是微处理器从 PSP 读取数据时的波形图。

图 10-2： 并行从动端口写操作时的波形图



注： IBF 标志位在这一时间段内不能被清零。

图 10-3： 并行从动端口读操作时的波形图



10.7 设计技巧

问 1： ***对于器件 PIC16C74 和 PIC16C74A， PSP 的操作好像不同。***

答 1：

是的，因设计更改， PIC16C74A 是边沿触发型，而 PIC16C74 是电平触发型。具体信息，请参见附录 C.9。

10.8 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 （即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制，即可使用。目前与并行从动端口相关的应用笔记有：

标题	应用笔记 #
Using the 8-bit Parallel Slave Port	AN579

10.9 版本历史

版本 A

这是描述并行从动端口的初始发行版。

第 11 章 Timer0

目录

本章包括以下一些主要内容：

11.1 简介 11-2
11.2 控制寄存器 11-3
11.3 操作 11-4
11.4 TMR0 中断 11-5
11.5 Timer0 外部时钟的使用 11-6
11.6 TMR0 的预分频器 11-7
11.7 设计技巧 11-10
11.8 相关应用笔记 11-11
11.9 版本历史 11-12

11.1 简介

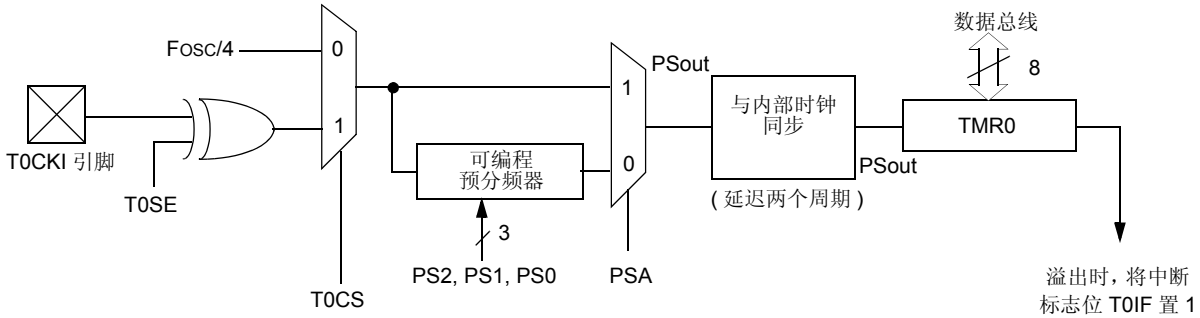
Timer0 模块有以下特性：

- 8 位定时器 / 计数器
- 可读写
- 软件可编程的 8 位预分频器
- 可选择内部或外部时钟信号
- 从 FFh 计数到 00h 时，发生溢出中断
- 外部时钟边沿选择

注： 要使 TMR0 寄存器得到 1:1 的预分频比，可将预分频器分配给看门狗定时器。

图 11-1 是 Timer0 模块的简化框图。

图 11-1：Timer0 结构框图



注 1: T0CS, T0SE, PSA, PS2:PS0 (OPTION_REG<5:0>)。
2: 该预分频器与看门狗定时器共用 (具体框图参见图 11-6)。

11.2 控制寄存器

OPTION_REG 寄存器是一个可读写寄存器，它含有各种控制位，用来设置 TMR0/WDT 预分频器、外部 INT 中断、TMR0 和 PORTB 的弱上拉等。

注： 要使 TMR0 寄存器得到 1:1 的预分频比，可将预分频器分配给看门狗定时器。

寄存器 11-1： OPTION_REG 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
$\overline{\text{RBPU}}$ (1)	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$ (1)**：弱上拉使能位
1 = 禁止弱上拉
0 = 使能弱上拉

bit 6 **INTEDG**：中断信号边沿选择位
1 = INT 引脚上升沿中断
0 = INT 引脚下降沿中断

bit 5 **T0CS**：TMR0 时钟源选择位
1 = T0CKI 引脚输入时钟
0 = 内部指令周期时钟 (CLKOUT)

bit 4 **T0SE**：TMR0 时钟源边沿选择位
1 = T0CKI 引脚电平由高到低转变时递增
0 = T0CKI 引脚电平由低到高转变时递增

bit 3 **PSA**：预分频器分配位
1 = 预分频器分配给 WDT
0 = 预分频器分配给 Timer0 模块

bit 2:0 **PS2:PS0**：预分频比选择位

选择位的值	TMR0 分频比	WDT 分频比
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

图注
R = 可读位　　W = 可写位
U = 未用位，读为 0　　- n = 上电复位值

注 1： 有些器件称此位为 $\overline{\text{GPPU}}$。带有 $\overline{\text{RBPU}}$ 位的器件在 PORTB 上有弱上拉，带有 $\overline{\text{GPPU}}$ 位的器件在 GPIO 端口有弱上拉。

11.3 操作

将 T0CS 位 (OPTION<5>) 清零可选择 TMR0 模式。在定时器模式下，Timer0 模块在每个指令周期递增 (不使用预分频器)。 TMR0 寄存器被写入时，在接下来的两个指令周期禁止递增 （见图 11-2 和 图 11-3)。用户可通过将校正值写入 TMR0 寄存器避开这一问题。

将 T0CS 位 (OPTION<5>) 置 1 可选择计数器模式。在计数器模式下， Timer0 可在 T0CKI 引脚的每个上升沿或下降沿递增计数。具体是上升沿还是下降沿由 Timer0 的时钟源边沿选择位 T0SE 位 (OPTION<4>) 决定。将 T0SE 位清零选择上升沿。外部时钟输入的限制在 **11.5 “Timer0 外部时钟的使用”小节**中详细讨论。

预分频器由 Timer0 和看门狗定时器共用，并在分配使用时相互排斥。预分频器的分配在软件中由 PSA 控制位 (OPTION<3>) 控制。将 PSA 清零将预分频器分配给 Timer0。预分频器是不可读写的。当预分频器分配给 Timer0 时，可选择的预分频比有 1:2, 1:4,..., 1:256。预分频器的具体操作参见 **11.6 “TMR0 的预分频器”小节**。

向 TMR0 寄存器写入数据将导致其禁止两个指令周期 (2TCY)。即，在 TMR0 寄存器中写入新的数据后，TMR0 在第三个指令周期后才递增计数 (图 11-2)。当预分频器被分配给 Timer0 模块时，任何向 TMR0 寄存器的写操作都将立即更新 TMR0 寄存器，并将预分频器清零。Timer0（TMR0 和预分频器）的递增计数也将在两个指令周期 (TCY) 中禁止。因此当预分频比配置为 2 时，向 TMR0 寄存器写入后，TMR0 寄存器将在 4 个 Timer0 时钟周期后才开始递增计数 (图 11-3)。此后， TMR0 将在预分频器中配置的时钟周期数之后进行递增计数。

图 11-2：Timer0 时序图：内部时钟 / 无预分频器

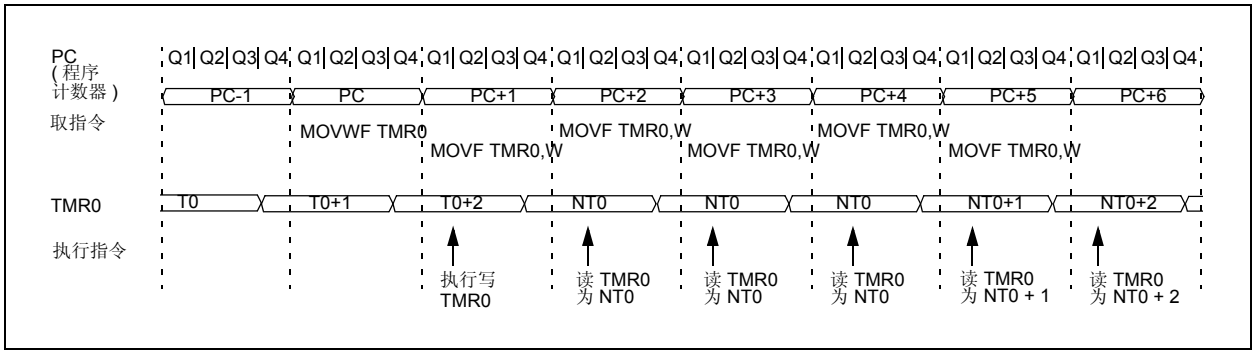
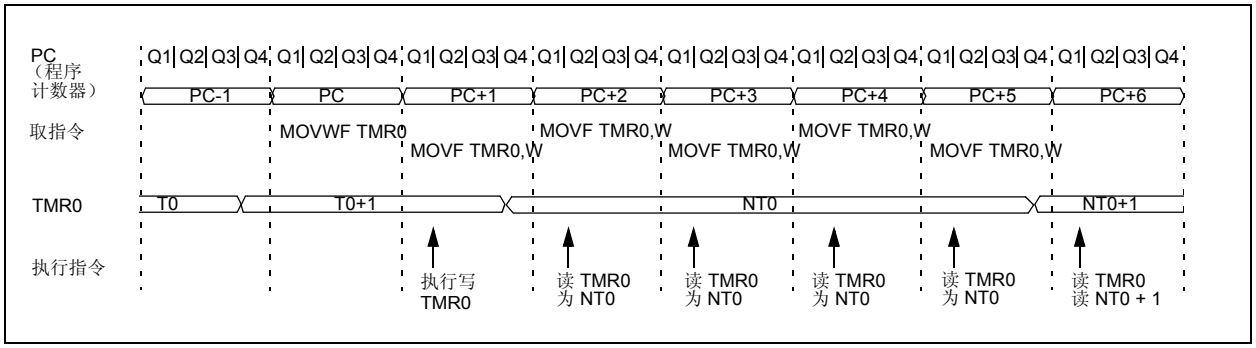


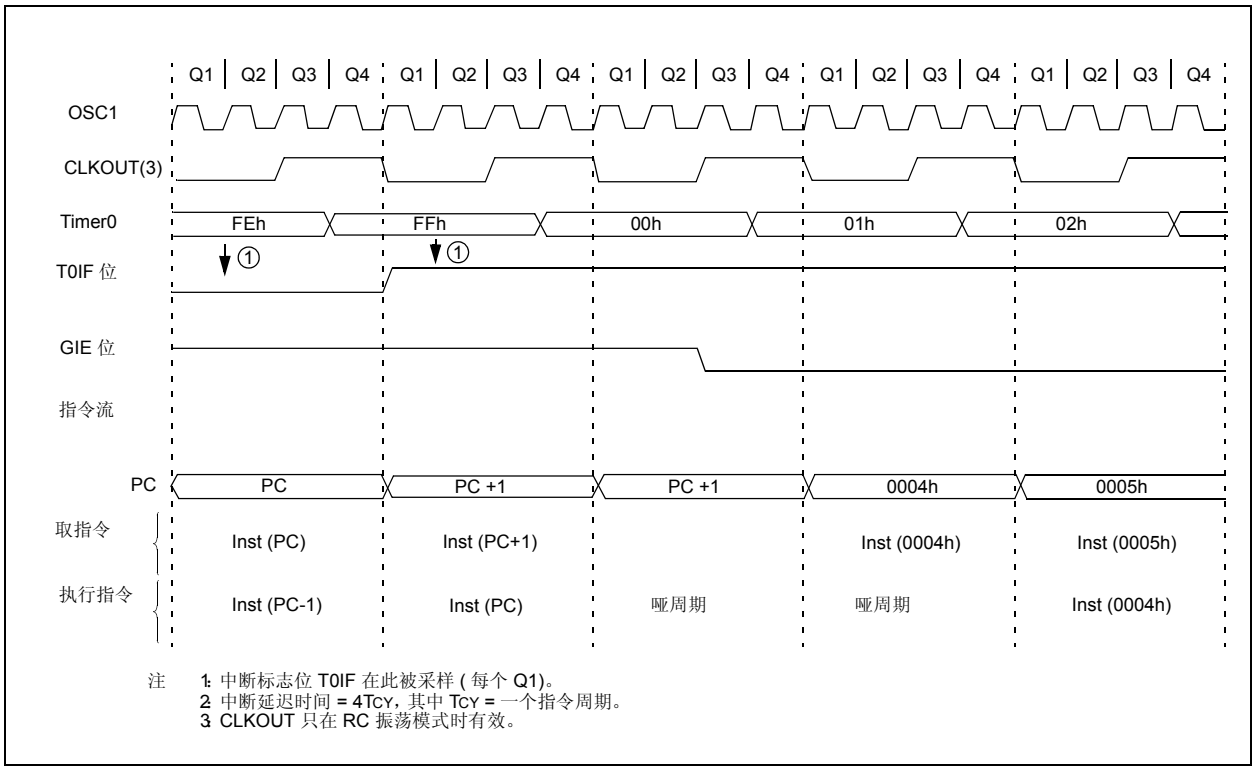
图 11-3：Timer0 时序图：内部时钟 / 预分频比 1:2



11.4 TMR0 中断

当 TMR0 寄存器从 FFh 到 00h 发生计数溢出时，即产生 TMR0 中断。该溢出将 T0IF 位 (INTCON<2>) 置 1。中断请求可以通过清零 T0IE (INTCON<5>) 来屏蔽。在重新允许中断前，必须在 Timer0 中断服务子程序中用软件将 T0IF 位清零。休眠状态时，由于 TMR0 被关闭，所以 TMR0 中断无法唤醒单片机。图 11-4 是 Timer0 的中断时序图。

图 11-4：TMR0 中断时序图



注 1：中断标志位 T0IF 在此被采样 (每个 Q1)。
2：中断延迟时间 = 4TCY，其中 TCY = 一个指令周期。
3：CLKOUT 只在 RC 振荡模式时有效。

11.5 Timer0 外部时钟的使用

当 Timer0 使用外部时钟输入时，必须满足一定要求，详见 **11.5.1“外部时钟的同步”** 这些要求确保外部时钟和内部相位时钟 (Tosc) 保持同步。同步后 Timer0 的实际递增计数也将有一定延时。

11.5.1 外部时钟的同步

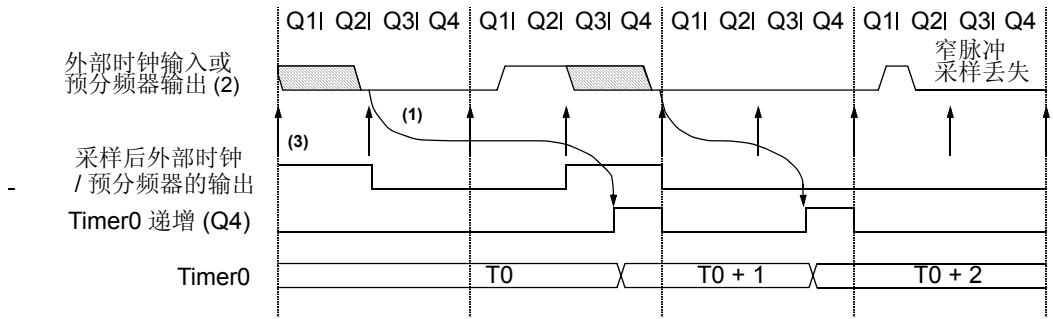
当不使用预分频器时，外部时钟输入与预分频器输出相同。在内部时钟的 Q2 和 Q4 周期对预分频器输出进行采样可实现 T0CKI 与内部时钟的同步 (图 11-5)。因此，要求 T0CKI 的高电平至少保持 2Tosc （加上 20 ns 的一小段 RC 延时）并且其低电平也至少保持 2Tosc（加上 20 ns 的一小段 RC 延时）。具体请参考所用器件的电气特性中的参数 40、41 和 42。

当使用预分频器时，外部时钟输入被预分频器分频，故预分频器的输出是对称的。为了使外部时钟满足采样要求，必须考虑计数器。因此， T0CKI 的周期必须至少保持 4Tosc（加上 20 ns 的一小段 RC 延时）除以预分频比。对 T0CKI 的高、低电平持续时间的唯一要求是不小于 10 ns 的最小脉宽要求。具体请参考所用器件的电气特性中的参数 40、41 和 42。

11.5.2 TMR0 递增计数的延时

由于预分频器的输出与内部时钟同步，所以从外部时钟沿发生的时间到 Timer0 模块实际递增计数有一小段延时。图 11-5 显示了从外部时钟沿到计时器递增计数的延时。

图 11-5：Timer0 与外部时钟时序图



注 1: 从时钟输入变化到 Timer0 递增有 3Tosc 到 7Tosc 的延时 (持续时间 Q=Tosc)。
因此，测量 Timer0 输入的相邻脉冲边沿间隔时，其最大误差为 ±4Tosc。
2: 如果未选择预分频器，则为外部时钟，否则为预分频器输出。
3: 箭头所指为采样时刻。

11.6 TMR0 的预分频器

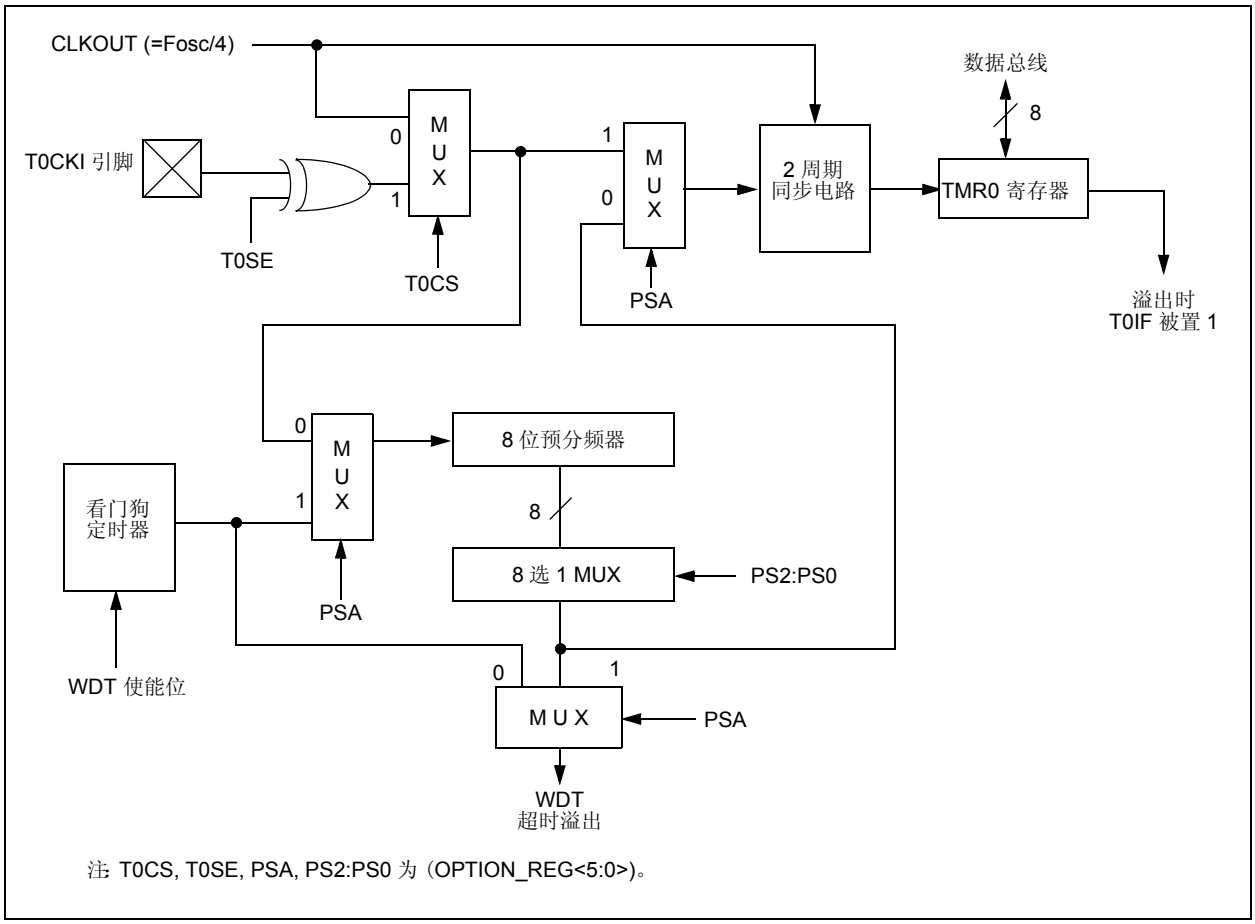
一个 8 位的计数器可作为 Timer0 模块的预分频器，或作为 WDT 的后分频器 (图 11-6)。为简化起见，在 Timer0 说明中称该计数器为 “预分频器”。因此，如果把预分频器分配给 TMR0 就意味着 WDT 无后分频器可用，反之亦然。

注： 预分频器要么分配给 Timer0 模块，要么分配给看门狗定时器。

PSA 和 PS2:PS0 位 (OPTION<3:0>) 决定预分频器的分配和分频比。

当预分频器分配给 Timer0 模块时，对 TMR0 寄存器执行的所有指令写入操作（如 `CLRF TMR0`, `MOVWF TMR0`, `BSF TMR0,x`.... 等) 都将对预分频器清零。当预分频器分配给 WDT 时，执行 `CLRWDT` 指令将同时将预分频器和 WDT 清零。预分频器是不可读写的。

图 11-6：Timer0/WDT 预分频器框图



注：T0CS, T0SE, PSA, PS2:PS0 为（OPTION_REG<5:0>)。

11.6.1 预分频器分配的切换

预分频器的分配完全由软件控制，即，它可在程序执行期间 “随时”被改变。

注：	为避免器件意外复位，当把预分频器从 Timer0 分配给 WDT 时，必须执行下列指令序列 (见例 11-1)。即使 WDT 被禁止也要执行该指令序列。

在例 11-1 中，如果所需的最终预分频比不是 1:1，就不需要对 OPTION_REG 进行初始修改。如果所需的最终预分频比是 1:1，就需先设置一个非 1:1 的临时预分频比，并将所需的最终预分频比在最后一次修改 OPTION_REG 时予以设置。

例 11-1: 改变预分频器 (Timer0→WDT)

如果所需的最终预分频比不是 1:1，则无需第 2 行和第 3 行。如果所需的最终预分频比是 1:1，则在第 2 行和第 3 行先设定临时预分频比，而所需的最终预分频比在第 10 行和第 11 行设置。

```
1)  BSF    STATUS, RP0   ;Bank1
2)  MOVLW  b'xx0x0xxx'   ;Select clock source and prescale value of
3)  MOVWF  OPTION_REG    ;other than 1:1
4)  BCF    STATUS, RP0   ;Bank0
5)  CLRF   TMR0          ;Clear TMR0 and prescaler
6)  BSF    STATUS, RP1   ;Bank1
7)  MOVLW  b'xxxx1xxx'   ;Select WDT, do not change prescale value
8)  MOVWF  OPTION_REG    ;
9)  CLRWDT               ;Clears WDT and prescaler
10) MOVLW  b'xxxx1xxx'   ;Select new prescale value and WDT
11) MOVWF  OPTION_REG    ;
12) BCF    STATUS, RP0   ;Bank0
```

要将预分频器从 WDT 分配给 Timer0 模块，使用例 11-2 所示的指令序列。

例 11-2: 改变预分频器 (WDT→Timer0)

```
    CLRWDT                  ; Clear WDT and prescaler
    BSF      STATUS, RP0    ; Bank1
    MOVLW    b'xxxx0xxx'    ; Select TMR0, new prescale
    MOVWF    OPTION_REG     ; value and clock source
    BCF      STATUS, RP0    ; Bank0
```

11.6.2 初始化

由于 Timer0 有一个软件可编程的时钟源，下面给出 2 个例子说明 Timer0 使用不同时钟源时的初始化。例 11-3 是使用内部时钟源 (定时器模式) 时的初始化，例 11-4 是使用外部时钟源 (计数器模式) 时的初始化。

例 11-3: Timer0 的初始化 (内部时钟源)

```
        CLRF    TMR0            ; Clear Timer0 register
        CLRF    INTCON          ; Disable interrupts and clear T0IF
        BSF     STATUS, RP0     ; Bank1
        MOVLW   0xC3            ; PortB pull-ups are disabled,
        MOVWF   OPTION_REG      ;   Interrupt on rising edge of RB0
                                ;   Timer0 increment from internal clock
                                ;   with a prescaler of 1:16.
        BCF     STATUS, RP0     ; Bank0
;**     BSF     INTCON, T0IE    ; Enable TMR0 interrupt
;**     BSF     INTCON, GIE     ; Enable all interrupts
;
; The TMR0 interrupt is disabled, do polling on the overflow bit
;
T0_OVFL_WAIT
        BTFSS   INTCON, T0IF
        GOTO    T0_OVFL_WAIT
; Timer has overflowed
```

例 11-4: Timer0 的初始化 (外部时钟源)

```
    CLRF    TMR0            ; Clear Timer0 register
    CLRF    INTCON          ; Disable interrupts and clear T0IF
    BSF     STATUS, RP0     ; Bank1
    MOVLW   0x37            ; PortB pull-ups are enabled,
    MOVWF   OPTION_REG      ;   Interrupt on falling edge of RB0
                            ;   Timer0 increment from external clock
                            ;   on the high-to-low transition of T0CKI
                            ;   with a prescaler of 1:256.
    BCF     STATUS, RP0     ; Bank0
;** BSF     INTCON, T0IE    ; Enable TMR0 interrupt
;** BSF     INTCON, GIE     ; Enable all interrupts
;
; The TMR0 interrupt is disabled, do polling on the overflow bit
;
T0_OVFL_WAIT
    BTFSS   INTCON, T0IF
    GOTO    T0_OVFL_WAIT
; Timer has overflowed
```

11.7 设计技巧

问 1: *在我的计数器 / 时钟应用中，为什么时钟会丢失时间或不准确？*

答 1:

如果您通过查询 TMR0 来查看它是否计满回零，可以执行：

```
wait   MOVF    TMR0,W     ; read the timer into W
       BTFSS   STATUS,Z   ; see if it was zero, if so,
                          ;    break from loop
       GOTO     wait      ; if not zero yet, keep waiting
```

丢失时钟周期的两种可能的情况为：

1. 当 TMR0 的递增计数是来自内部指令时钟，或与内部时钟几乎同样快的外部时钟源时，那么在执行两周期的 GOTO 指令时，可能会发生溢出，使您的时钟周期丢失。在这种情况下，应将 TMR0 进行预分频。
   或者您也可做一个测试，通过检测小于标称值的值来查看 TMR0 是否计满回零：

   ```
   Wait   movlw  3
          subwf  TMR0,W
          btfsc  STATUS,C
          goto   Wait
   ```

2. 当向 TMR0 写入数据时，会丢失两个时钟周期。通常您希望对一个特定的时间段计数，如十进制数 100。此时您会将 156 写入 TMR0(256–100 = 156)。然而，由于向 TMR0 写入数据时会丢失两个指令周期（用于内部逻辑电路的同步），因此实际上应向定时器写入 158。

11.8 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，通过适当修改并受一定限制即可使用。目前与 Timer0 相关的应用笔记有：

标题	应用笔记 #
Frequency Counter Using PIC16C5X	AN592
A Clock Design using the PIC16C54 for LED Display and Switch Inputs	AN590

11.9 版本历史

版本 A

这是描述 Timer0 模块的初始发行版。

第 12 章 Timer1

目录：

本章包括以下一些主要内容：

12.1 简介 12-2
12.2 控制寄存器 12-3
12.3 Timer1 工作在定时器模式 12-4
12.4 Timer1 工作在同步计数器模式 12-4
12.5 Timer1 工作在异步计数器模式 12-5
12.6 Timer1 振荡器 12-7
12.7 休眠操作 12-9
12.8 用 CCP 触发器的输出将 Timer1 复位 12-9
12.9 Timer1 寄存器 （TMR1H:TMR1L）的复位 12-9
12.10 Timer1 预分频器 12-9
12.11 初始化 12-10
12.12 设计技巧 12-12
12.13 相关应用笔记 12-13
12.14 版本历史 12-14

12.1 简介

Timer1 模块是由两个可读写的 8 位寄存器（TMR1H 和 TMR1L）组成的 16 位定时器 / 计数器。TMR1 寄存器对(TMR1H:TMR1L) 从 0000h 递增到 FFFFh 后，计满回零到 0000h。如果允许 Timer1 中断，则溢出时会产生 Timer1 中断。该中断可通过置位/清零 TMR1IE 位来允许/禁止。

Timer1 可以有三种工作模式：

- 同步定时器模式
- 同步计数器模式
- 异步计数器模式

工作模式是由时钟选择位TMR1CS (T1CON<1>)和同步控制位$\overline{\text{T1SYNC}}$来决定的（如图 12-1）。

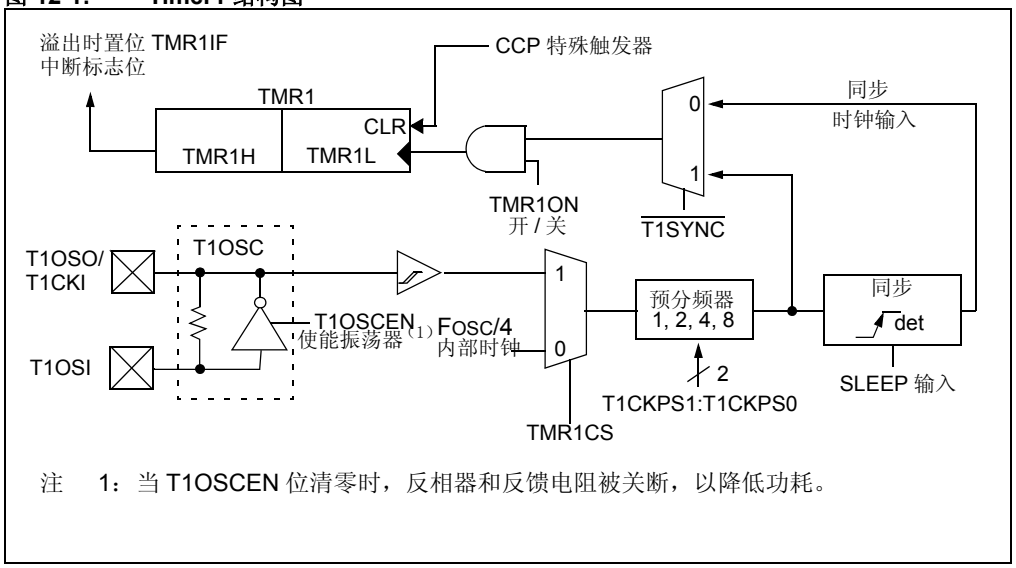
在定时器模式下，Timer1 在每个指令周期递增。而在计数器模式下，Timer1 在 T1CKI 引脚上外部时钟的每个上升沿递增。

Timer1 可以通过 TMR1ON(T1CON<0>) 控制位来打开和关闭。

Timer1 还有一个内部 “复位输入”，可由一个 CCP 模块产生。

Timer1 可以外接晶体振荡器，当 Timer1 的振荡器被使能 (T1OSCEN 位置 1) 时， T1OSI 和 T1OSO 引脚设定为输入引脚。这就是说，其相应的 TRIS 值被忽略。

图 12-1: Timer1 结构图



注 1：当 T1OSCEN 位清零时，反相器和反馈电阻被关断，以降低功耗。

12.2 控制寄存器

寄存器 12-1 显示了 Timer1 控制寄存器

寄存器 12-1: T1CON:Timer1 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
—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7:6 **未用 :** 始终读为 '0'

bit 5:4 **T1CKPS1:T1CKPS0**: Timer1 输入时钟预分频比选择位

11 = 1:8 预分频比
10 = 1:4 预分频比
01 = 1:2 预分频比
00 = 1:1 预分频比

bit 3 **T1OSCEN**: Timer1 振荡器使能位

1 = 振荡器被使能
0 = 振荡器被关闭。振荡器的反相器和反馈电阻被关断，以降低功耗

bit 2 **$\overline{\text{T1SYNC}}$**: Timer1 外部时钟输入同步控制位

当 TMR1CS = 1 时 :

1 = 不同步外部时钟
0 = 同步外部时钟

当 TMR1CS = 0 时 :

此位被忽略。TMR1CS = 0 时 Timer1 使用内部时钟。

bit 1 **TMR1CS**: Timer1 时钟源选择位

1 ： 选择 T1OSO/T1CKI 引脚的外部时钟 (上升沿计数)
0 ： 选择内部时钟 (FOSC/4)

bit 0 **TMR1ON**: Timer1 使能位

1 ： 使能 Timer1
0 ： 关闭 Timer1

符号：	
R = 可读	W = 可写
U = 未用， 读为 0	- n = 上电复位值

12.3 Timer1 工作在定时器模式

将 TMR1CS (T1CON<1>) 清 0，选择 TMR1 工作在定时器模式。在这种模式下，定时器的输入时钟是内部时钟频率的 4 分频（FOSC/4）。因为内部时钟总是同步的，所以同步控制位 T1SYNC (T1CON<2>) 此时不起作用。

12.4 Timer1 工作在同步计数器模式

将 TMR1CS(T1CON<1>) 置 1，选择 TMR1 工作在计数器模式。在这种模式下，计数器在引脚 T1OSI（T1OSCEN 置 1 时）或 T1OSO/T1CK（T1OSCEN 清 0 时）输入时钟的每个上升沿递增。

如果 $\overline{\text{T1SYNC}}$ 位清 0，那么外部时钟输入与内部相位时钟同步，同步是在预分频器后完成的。预分频器是一个异步脉动计数器。

在同步计数器模式下，当工作于休眠方式时，即使使用的是外部时钟，Timer1 也不会递增，因为同步电路已被关闭。但是预分频器继续递增。

12.4.1 同步计数器模式下的外部时钟输入

当 Timer1 工作在同步计数器模式时，外部输入的时钟信号必须满足一定的要求，这主要是因为要与内部相位时钟 (TOSC) 同步。同步后， Timer1 的实际递增计数与外部时钟沿之间会产生一定的延时。

当预分频器的分频比为 1:1 时，外部输入时钟和预分频器的输出相同。T1CKI 与内部相位时钟的同步是通过在相邻的两个 Tosc 内部相位时钟下对预分频器的输出进行采样来实现的。因此，要求 T1CKI 引脚上的信号高、低电平分别至少保持 2Tosc (加上一小段 RC 延时)。请参考 **“电气规范”一章**中的参数 45、 46 和 47。

当预分频器的分频比是除 1:1 外的其它情况时，外部输入时钟信号要先经过异步脉动计数器预分频器的分频，而使预分频器的输出对称。为了使外部时钟满足采样要求，必须将脉动计数器考虑在内。因此，要求 T1CKI 引脚上的信号至少保持 4Tosc(加上一小段 RC 延时) 供预分频器分频。此外，T1CKI 引脚上的时钟信号还必须满足高低电平的最小脉宽要求。参见 **“电气规范”一章**中的参数 40、 42、 45、 46 和 47。

12.5 Timer1 工作在异步计数器模式

当 $\overline{\text{T1SYNC}}$ (T1CON<2>) 位置 1 时，外部时钟输入就不同步。 Timer1 继续进行异步于内部相位时钟的递增计数。在休眠状态下， Timer1 将继续运行，并在计满溢出时产生中断，唤醒处理器。但在软件中应特别注意对 Timer1 的读写 （参见 **12.5.2 “异步计数工作模式下对 Timer1 的读写操作”小节**）。异步计数器能在器件休眠时工作，因此 Timer1 可用于实现一个实时时钟。

在异步计数器模式时， Timer1 不能用作捕捉器或比较器的工作时基。

12.5.1 异步时钟的外部时钟时序

$\overline{\text{T1SYNC}}$ 控制位置 1 时， Timer1 将完全异步递增计数。输入时钟必须满足最小高电平和低电平的最小脉宽要求。参见 **“电气规范”一章**的参数 45、 46 和 47。

12.5.2 异步计数工作模式下对 Timer1 的读写操作

当定时器工作在外部异步时钟下时，对 TMR1H 或 TMR1L 寄存器进行读操作能够保证有效的读取 （由硬件完成）。但用户应牢记，用读两个 8 位值来读一个 16 位值本身就存在问题，这是因为定时器在两次读操作之间可能会溢出。

对于写操作，建议用户关闭定时器后再写入数据；而计数器正在计数时，向 Timer1 的寄存器写入数据可能会产生冲突，这可导致定时器寄存器中产生无法预知的值。

由于是分两次读出整个 16 位数据，因此在读 16 位值时应非常注意。例 12-1 显示了为何不能直接读 16 位寄存器的原因。

例 12-1: 读 16 位寄存器的问题

TMR1 值	顺序 1		顺序 2	
	动作	TMPH:TMPL	动作	TMPH:TMPL
04FFh	读 TMR1L 内容	xxxxh	读 TMR1H 内容	xxxxh
0500h	存入 TMPL	xxFFh	存入 TMPH	04xxh
0501h	读 TMR1H 内容	xxFFh	读 TMR1L 内容	04xxh
0502h	存入 TMPH	05FFh	存入 TMPL	0401h

遇到例 12-1 的问题时，可用例 12-2 给出的这段程序读取 16 位定时器的值。在定时器无法停止时，该程序将很有帮助。

例 12-2: 读取运行中的定时器的 16 位值

```
; All interrupts are disabled
       MOVF   TMR1H, W    ; Read high byte
       MOVWF  TMPH        ;
       MOVF   TMR1L, W    ; Read low byte
       MOVWF  TMPL        ;
       MOVF   TMR1H, W    ; Read high byte
       SUBWF  TMPH,  W    ; Sub 1st read with 2nd read
       BTFSC  STATUS,Z    ; Is result = 0
       GOTO   CONTINUE    ; Good 16-bit read
;
; TMR1L may have rolled over between the read of the high and low bytes.
; Reading the high and low bytes now will read a good value.
;
       MOVF   TMR1H, W    ; Read high byte
       MOVWF  TMPH        ;
       MOVF   TMR1L, W    ; Read low byte
       MOVWF  TMPL        ;
; Re-enable the Interrupt (if required)
CONTINUE                  ; Continue with your code
```

可直接将 16 位值写入 TMR1 寄存器。先将 TMR1L 清 0，以确保 TMR1L 向 TMR1H 溢出前还可进行多个 Timer1 时钟/振荡器周期计数。然后写入 TMR1H，最后写入 TMR1L。该过程见例 12-3：

例 12-3: 向运行中的定时器写入 16 位值

```
; All interrupts are disabled
       CLRF   TMR1L       ; Clear Low byte, Ensures no
                          ;   rollover into TMR1H
       MOVLW  HI_BYTE     ; Value to load into TMR1H
       MOVWF  TMR1H, F    ; Write High byte
       MOVLW  LO_BYTE     ; Value to load into TMR1L
       MOVWF  TMR1H, F    ; Write Low byte
; Re-enable the Interrupt (if required)
CONTINUE                  ; Continue with your code
```

12.6 Timer1 振荡器

在 T1OSI (放大器输入) 和 T1OSO (放大器输出) 引脚之间内接有晶体振荡器电路，通过将 T1OSCEN 控制位 (T1CON<3>) 置位使能该电路。该振荡器是一个低功耗的振荡器，频率可达 200 kHz。它在休眠状态下可以继续工作。一般建议的使用频率为 32 kHz，这是一个产生实时时钟的理想频率。表 12-1 所示为不同频率的晶体振荡器所需的外接电容。

Timer1 的振荡器是一种低速低功耗 （LP) 振荡器，用户应在 Timer1 的振荡器刚开始工作时，使用一定的软件延时，以确保振荡器先可靠起振。

注：	这使得计数器在休眠模式下仍然继续工作 （递增），因此 Timer1 可用于产生一个实时时钟。

表 12-1: Timer1 振荡器的电容器选择表

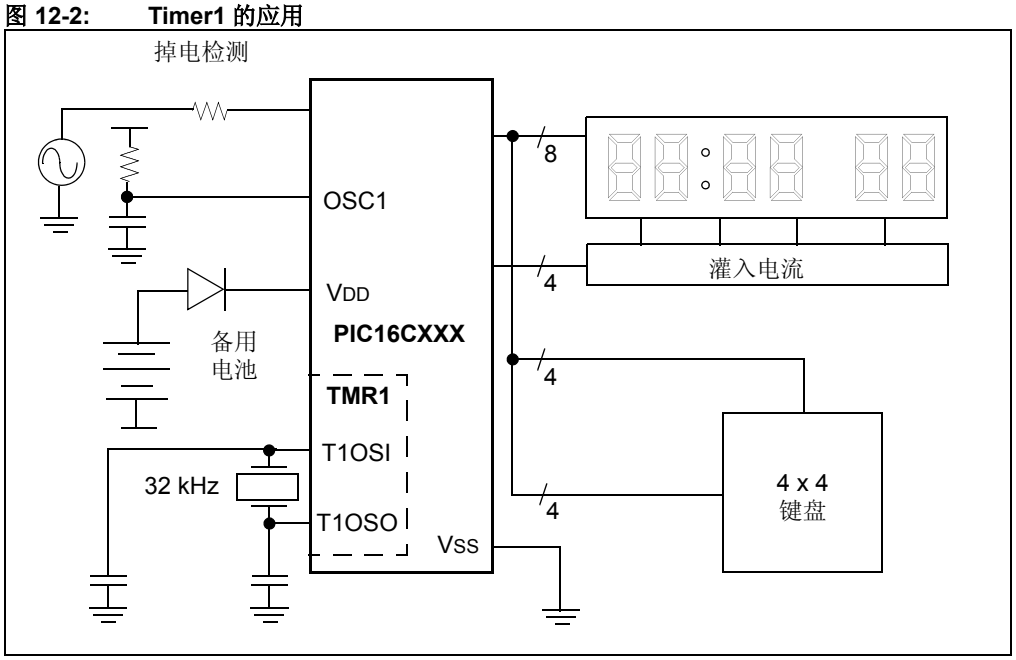
振荡类型	频率	C1	C2
LP	32 kHz	33 pF	33 pF
	100 kHz	15 pF	15 pF
	200 kHz	15 pF	15 pF
经测试的晶体：			
32.768 kHz	Epson C-001R32.768K-A		± 20 PPM
100 kHz	Epson C-2 100.00 KC-P		± 20 PPM
200 kHz	STD XTL 200.000 kHz		± 20 PPM

注 1：增加电容容量可提高振荡器的稳定性，但是同时也延长了振荡器的起振时间。
2：由于每个谐振器 / 晶体均有其自身的特性，用户应向谐振器 / 晶体厂商咨询适当的外部元件值。

12.6.1 典型应用

该功能一般适用于既需要实时时钟、又需要低功耗的应用场合。Timer1 振荡器允许器件进入休眠模式时定时器继续递增计数。当 Timer1 溢出中断时会唤醒器件，从而更新相关寄存器的值。

图 12-2: Timer1 的应用



12.7 休眠操作

当 Timer1 被配置为异步工作模式时， TMR1 寄存器将继续对每个输入时钟进行递增计数 （或对多个时钟进行预分频）。当 TMR1 寄存器溢出时， TMR1IF 被置位，此时如果中断使能，会将处理器从休眠状态中唤醒。

由于该电路的运行，Timer1 振荡器会增加一个额外的电流消耗。也就是说，休眠时的电流将不再仅仅是器件的漏电流，而是包括了 Timer1 振荡器和其他 Timer1 电路的工作电流。

12.8 用 CCP 触发器的输出将 Timer1 复位

如果 CCP 模块被设置为比较器模式以产生 “特殊事件触发”信号 （即 CCP1M3:CCP1M0 = 1011），则该信号会将 Timer1 复位。

注：	CCP 模块产生的特殊事件触发信号不会将中断标志位 TMR1IF 置位。

为了利用这个功能， Timer1 必须被设置为定时器或同步计数器模式。如果 Timer1 在异步计数器模式下运行，该复位操作不工作，用户也不应使用这一功能。

当 Timer1 的写操作和 CCP 模块的特殊事件触发复位同时发生时，则写操作优先。

在这种操作方式下， CCPRxH:CCPRxL 这对寄存器实际上变成了 Timer1 的周期寄存器。

12.9 Timer1 寄存器 （TMR1H:TMR1L）的复位

除了 CCP 特殊事件触发器， POR 或其它复位并不会将 TMR1H 和 TMR1L 复位。

在上电复位或欠压复位， T1CON 寄存器被复位为 00h，而其它复位并不影响寄存器。

12.10 Timer1 预分频器

当对寄存器 TMR1H 或 TMR1L 进行写操作时，预分频器将被清零。

表 12-2: Timer1 作为定时器 / 计数器时与之相关的寄存器

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	其他复位时的值
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	TMR1IF (1)								0	0
PIE	TMR1IE (1)								0	0
TMR1L	保存 16 位 TMR1 寄存器低字节的寄存器								xxxx xxxx	uuuu uuuu
TMR1H	保存 16 位 TMR1 寄存器高字节的寄存器								xxxx xxxx	uuuu uuuu
T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON	--00 0000	--uu uuuu

图注： x = 未知， u = 不变， - = 未定义，读为 0。
阴影部分与 Timer1 模块无关。

注 1：该位的配置与具体器件有关。
2：也可将这些位命名为 GPIE 和 GPIF。

12.11 初始化

由于 Timer1 具备一个软件可编程的时钟源，以下给出了 3 种工作模式下初始化的例子。例 12-4 是内部时钟源的初始化；例 12-5是外部时钟源的初始化；例 12-6是外部振荡器模式的初始化。

例 12-4: Timer1 的初始化 (内部时钟源)

```
    CLRF   T1CON         ; Stop Timer1, Internal Clock Source,
                         ;   T1 oscillator disabled, prescaler = 1:1
    CLRF   TMR1H         ; Clear Timer1 High byte register
    CLRF   TMR1L         ; Clear Timer1 Low byte register
    CLRF   INTCON        ; Disable interrupts
    BSF    STATUS, RP0   ; Bank1
    CLRF   PIE1          ; Disable peripheral interrupts
    BCF    STATUS, RP0   ; Bank0
    CLRF   PIR1          ; Clear peripheral interrupts Flags
    MOVLW  0x30          ; Internal Clock source with 1:8 prescaler
    MOVWF  T1CON         ;   Timer1 is stopped and T1 osc is disabled
    BSF    T1CON, TMR1ON ; Timer1 starts to increment
;
; The Timer1 interrupt is disabled, do polling on the overflow bit
;
T1_OVFL_WAIT
    BTFSS  PIR1, TMR1IF
    GOTO   T1_OVFL_WAIT
;
; Timer has overflowed
;
    BCF    PIR1, TMR1IF
```

例 12-5: Timer1 的初始化 (外部时钟源)

```
    CLRF   T1CON         ; Stop Timer1, Internal Clock Source,
                         ;   T1 oscillator disabled, prescaler = 1:1
    CLRF   TMR1H         ; Clear Timer1 High byte register
    CLRF   TMR1L         ; Clear Timer1 Low byte register
    CLRF   INTCON        ; Disable interrupts
    BSF    STATUS, RP0   ; Bank1
    CLRF   PIE1          ; Disable peripheral interrupts
    BCF    STATUS, RP0   ; Bank0
    CLRF   PIR1          ; Clear peripheral interrupts Flags
    MOVLW  0x32          ; External Clock source with 1:8 prescaler
    MOVWF  T1CON         ;   Clock source is synchronized to device
                         ;   Timer1 is stopped and T1 osc is disabled
    BSF    T1CON, TMR1ON ; Timer1 starts to increment
;
; The Timer1 interrupt is disabled, do polling on the overflow bit
;
T1_OVFL_WAIT
    BTFSS  PIR1, TMR1IF
    GOTO   T1_OVFL_WAIT
;
; Timer has overflowed
;
    BCF    PIR1, TMR1IF
```

例 12-6: Timer1 的初始化 (外部振荡器时钟源)

```
    CLRF   T1CON         ; Stop Timer1, Internal Clock Source,
                         ;   T1 oscillator disabled, prescaler = 1:1
    CLRF   TMR1H         ; Clear Timer1 High byte register
    CLRF   TMR1L         ; Clear Timer1 Low byte register
    CLRF   INTCON        ; Disable interrupts
    BSF    STATUS, RP0   ; Bank1
    CLRF   PIE1          ; Disable peripheral interrupts
    BCF    STATUS, RP0   ; Bank0
    CLRF   PIR1          ; Clear peripheral interrupts Flags
    MOVLW  0x3E          ; External Clock source with oscillator
    MOVWF  T1CON         ;   circuitry, 1:8 prescaler, Clock source
                         ;   is asynchronous to device
                         ;   Timer1 is stopped
    BSF    T1CON, TMR1ON ; Timer1 starts to increment
;
; The Timer1 interrupt is disabled, do polling on the overflow bit
;
T1_OVFL_WAIT
    BTFSS  PIR1, TMR1IF
    GOTO   T1_OVFL_WAIT
;
; Timer has overflowed
;
    BCF    PIR1, TMR1IF
```

12.12 设计技巧

问 1: ***Timer1 定时时间不准确。***

答 1:

以下原因可能造成上述问题：

1. 您不应向Timer1执行写操作，否则会导致时间误差。大多数情况这意味着您不应向TMR1L执行写操作，但若条件允许，您也可向 TMR1H 寄存器写入数据。通常，当您希望 Timer1 溢出中断早于 16 位延时溢出时，才可向 TMR1H 寄存器写入数据。
2. 应确保您的 PCB 采用了合理的布局，从而使噪声无法耦合到 Timer1 的振荡器线路上。

12.13 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，经过适当修改并受到一定的限制即可使用。目前与 Timer1 相关的应用笔记有：

标题	应用笔记 #
Using Timer1 in Asynchronous Clock Mode	AN580
Low Power Real Time Clock	AN582
Yet another Clock using the PIC16C92X	AN649

12.14 版本历史

版本 A

这是描述 Timer1 模块的初始发行版。

第 13 章 Timer2

目录

本章包括以下一些主要内容：

13.1 简介 13-2
13.2 控制寄存器 13-3
13.3 定时器时钟源 13-4
13.4 定时器 TMR2 和 PR2 周期寄存器 13-4
13.5 TMR2 匹配输出 13-4
13.6 将 Timer2 的预分频器和后分频器清零 13-4
13.7 休眠操作 13-4
13.8 初始化 13-5
13.9 设计技巧 13-6
13.10 相关应用笔记 13-7
13.11 版本历史 13-8

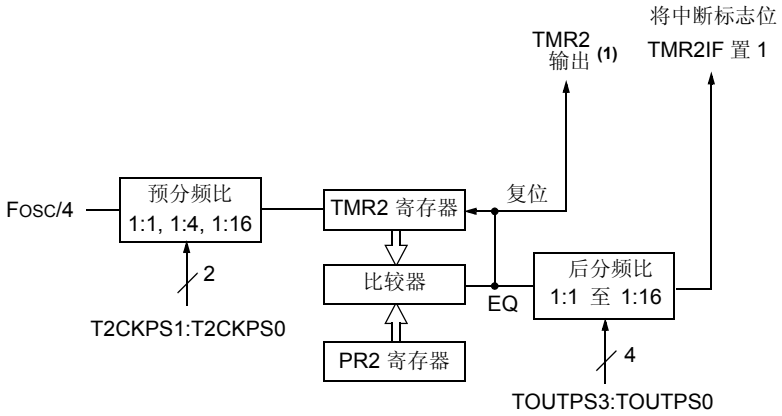
13.1 简介

Timer2 是一个 8 位定时器，带有一个预分频器、一个后分频器和一个周期寄存器。当将预分频器和后分频器设置为最大值时，其溢出时间与 16 位定时器的相同。

在 PWM 模式下使用 CCP 模块时， Timer2 为 PWM 时基。

图 13-1 显示了 Timer2 的结构框图。后分频器对 TMR2 寄存器和 PR2 寄存器的匹配次数进行计数。这样有助于减少中断服务程序的调用频率，优化 CPU 性能。

图 13-1：Timer2 结构框图



注： TMR2 寄存器输出可在软件中由 SSP 模块选择作为波特率时钟信号。

13.2 控制寄存器

寄存器 13-1 为 Timer2 控制寄存器。

寄存器 13-1: T2CON: Timer2 控制寄存器

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7 **未用：**读为 '0'

bit 6:3 **TOUTPS3:TOUTPS0**: 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 允许位

1 = Timer2 允许
0 = Timer2 关闭

bit 1:0 **T2CKPS1:T2CKPS0**: Timer2 时钟预分频比选择位

00 = 预分频比为 1
01 = 预分频比为 4
1x = 预分频比为 16

图注		
R = 可读位	W = 可写位	
U = 未用，读为 0		- n = 上电复位值

13.3 定时器时钟源

Timer2 模块有一个输入时钟源，即器件时钟 (Fosc/4)。通过控制位 T2CKPS1:T2CKPS0 (T2CON<1:0>) 可选择预分频比为 1:1、1:4 或 1:16。

13.4 定时器 TMR2 和 PR2 周期寄存器

TMR2 寄存器是可读写的，在器件的所有复位时清零。 Timer2 从 00h 开始递增计数直到与 PR2 匹配，然后在下一个周期复位并重新从 00h 开始计数。 PR2 为可读写寄存器。

当发生 WDT、POR、$\overline{\text{MCLR}}$ 或 BOR 复位时， TMR2 清零，而 PR2 寄存器被置 1。

将 TMR2ON 控制位 (T2CON<2>) 清零可关闭 Timer2(禁止递增计数)，这样可最大限度地减小模块的功耗。

13.5 TMR2 匹配输出

TMR2 匹配输出信号可输出到以下两个时钟源：

1. Timer2 后分频器
2. SSP 时钟输入

通过四个控制位可以选择从 1:1 到 1:16(含) 的后分频比。后分频器溢出后将使 TMR2 中断标志位 TMR2IF 置 1，表明 Timer2 溢出。这样有助于降低 Timer2 中断服务程序的软件开销，这是因为程序只在后分频器匹配时才会执行一次。

TMR2 的匹配输出还被输出到同步串行口模块，可在软件中选择它作为移位时钟的时钟源。

13.6 将 Timer2 的预分频器和后分频器清零

当下面的任一种情况发生时，预分频器和后分频器被清零。

- 向 TMR2 寄存器执行写操作
- 向 T2CON 寄存器执行写操作

注： 当向 T2CON 写入数据时， TMR2 并不清零。

- 器件的任何复位 (包括上电复位、$\overline{\text{MCLR}}$ 复位、看门狗定时器复位、欠压复位或奇偶校验错误复位)。

13.7 休眠操作

在休眠期间，TMR2 并不递增计数。预分频器将保留最末次的预分频器计数值，并随时作好器件从休眠状态唤醒时恢复运行的准备。

表 13-1: 与 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	POR、BOR 和 PER 时的复位值	其他复位时的值
INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
PIR	TMR2IF (1)								0	0
PIE	TMR2IE (1)								0	0
TMR2	Timer2 模块的寄存器								0000 0000	0000 0000
T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	-000 0000
PR2	Timer2 周期寄存器								1111 1111	1111 1111

图注： x = 未知， u = 不变 , - = 未用，读为 '0'。
阴影部分与 Timer2 模块无关。

注 1: 该位的位置与具体器件有关。

13.8 初始化

例 13-1 显示了如何初始化 Timer2 模块，包括指定 Timer2 的预分频器和后分频器。

例 13-1: Timer2 的初始化

```
       CLRF    T2CON          ; Stop Timer2, Prescaler = 1:1,
                              ;   Postscaler = 1:1
       CLRF    TMR2           ; Clear Timer2 register
       CLRF    INTCON         ; Disable interrupts
       BSF     STATUS, RP0    ; Bank1
       CLRF    PIE1           ; Disable peripheral interrupts
       BCF     STATUS, RP0    ; Bank0
       CLRF    PIR1           ; Clear peripheral interrupts Flags
       MOVLW   0x72           ; Postscaler = 1:15, Prescaler = 1:16
       MOVWF   T2CON          ;   Timer2 is off
       BSF     T2CON, TMR2ON  ; Timer2 starts to increment
;
; The Timer2 interrupt is disabled, do polling on the overflow bit
;
T2_OVFL_WAIT
       BTFSS   PIR1, TMR2IF   ; Has TMR2 interrupt occurred?
       GOTO    T2_OVFL_WAIT   ; NO, continue loop
;
; Timer has overflowed
;
       BCF     PIR1, TMR2IF   ; YES, clear flag and continue.
```

13.9 设计技巧

当前没有相关的设计技巧。

13.10 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与 Timer2 模块相关的应用笔记有：

标题	应用笔记 #
Using the CCP Module	AN594
Air Flow Control using Fuzzy Logic	AN600
Adaptive Differential Pulse Code Modulation using PICmicros	AN643

13.11 版本历史

版本 A

这是描述 Timer2 模块的初始发行版。

第 14 章 比较 / 捕捉 / 脉宽调制 (CCP)

目录

本章包括以下一些主要内容：

14.1 简介 14-2
14.2 控制寄存器 14-3
14.3 捕捉模式 14-4
14.4 比较模式 14-6
14.5 PWM 模式 14-8
14.6 初始化 14-12
14.7 设计技巧 14-15
14.8 相关应用笔记 14-17
14.9 版本历史 14-18

14.1 简介

每个 CCP(捕捉 / 比较 / 脉宽调制) 模块有一个 16 位寄存器，它可以用作 16 位捕捉寄存器、 16 位比较寄存器或 10 位 PWM 主 / 从占空比寄存器。除了特殊事件触发器之外，各 CCP 模块的操作是相同的。

每个 CCP 模块含有 3 个寄存器。一个器件中可能有多个 CCP 模块。在本章中我们用通用名来表示这些 CCP 寄存器。 这些通用名见表 14-1。

表 14-1: 通用 CCP 的命名

通用名	CCP1	CCP2	注释
CCPxCON	CCP1CON	CCP2CON	CCP 控制寄存器
CCPRxH	CCPR1H	CCPR2H	CCP 高字节
CCPRxL	CCPR1L	CCPR2L	CCP 低字节
CCPx	CCP1	CCP2	CCP 引脚

表 14-2 列出了各种模式下 CCP 模块的定时器资源。表 14-3 显示了 CCP 模块间的相互关系，其中， CCPx 是一个 CCP 模块，而 CCPy 是另一个 CCP 模块。

表 14-2: CCP 模式与定时器

CCP 模式	定时器
捕捉	Timer1
比较	Timer1
PWM	Timer2

表 14-3: CCP 模块间的相互关系

CCPx 模式	CCPy 模式	相互关系
捕捉	捕捉	使用相同的 TMR1 时基
捕捉	比较	比较模式应被设置成特殊事件触发器，这将使 TMR1 清零。
比较	比较	比较模式应被设置成特殊事件触发器，这将使 TMR1 清零。
PWM	PWM	PWM 将具有相同的频率和更新速率 (TMR2 中断)
PWM	捕捉	无
PWM	比较	无

14.2 控制寄存器

寄存器 14-1: CCPxCON 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
—	—	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit 7:6 **未用：**读为 '0'

bit 5:4 **DCxB1:DCxB0**：PWM 占空比的 bit1 和 bit0

捕捉模式：
未用

比较模式：
未用

PWM 模式：

它们是 10 位 PWM 占空比的两个 LSB (bit1 和 bit0)。占空比的高 8 位 (DCx9:DCx2) 在 CCPRxL 中。

bit 3:0 **CCPxM3:CCPxM0**: CCPx 模式选择位

0000 = 捕捉 / 比较 /PWM 关闭 (即复位 CCPx 模块)
0100 = 捕捉模式，每个下降沿发生
0101 = 捕捉模式，每个上升沿发生
0110 = 捕捉模式，每 4 个上升沿发生
0111 = 捕捉模式，每 16 个上升沿发生
1000 = 比较模式，
CCP 引脚初始为低电平，比较相符时，强制 CCP 引脚为高电平 (CCPIF 置 1)
1001 = 比较模式，
CCP 引脚初始为高电平，比较相符时，强制 CCP 引脚为低电平 (CCPIF 置 1)
1010 = 比较模式，
比较相符时，产生软件中断
(CCPIF 置 1， CCP 引脚不受影响)
1011 = 比较模式，
比较相符时，产生特殊触发事件
(CCPIF 置 1， CCP 引脚不受影响)
11xx = PWM 模式

图注	
R = 可读位	W = 可写位
U = 未用，读为 '0'	- n = 上电复位值

14.3 捕捉模式

在捕捉模式下，当 CCPx 的引脚发生以下事件时，CCPRxH:CCPRxL 即捕捉 TMR1 寄存器的 16 位计数值。

- 每个脉冲的下降沿
- 每个脉冲的上升沿
- 每 4 个脉冲的上升沿
- 每 16 个脉冲的上升沿

由控制位 CCPxM3:CCPxM0 (CCPxCON<3:0>) 来选择上述 4 种事件之一。当一个捕捉发生时，中断请求标志位 CCPxIF 置 1。该位必须用软件清零。如果寄存器 CCPRx 中的值被读出之前发生另一个捕捉，那么之前捕捉的数据将会丢失。

注： 欲使 CCP 模块使用捕捉功能， Timer1 必须工作在定时器或同步计数器模式。在异步计数器模式下，可能无法进行捕捉操作。

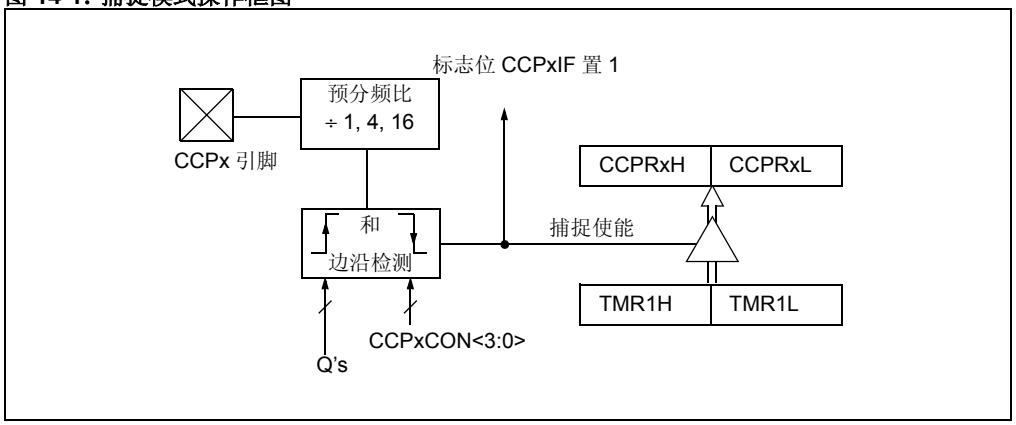
如图 14-1 所示，捕捉并不使 16 位 TMR1 寄存器复位。因此 Timer1 可被用做其他操作的时基。根据前后两次捕捉值的差异，可以很容易地计算出这两次捕捉的时间间隔。如果 Timer1 溢出，TMR1IF 位被置 1，中断使能时会产生中断，使时基扩展到大于 16 位的数。

14.3.1 CCP 引脚配置

捕捉模式下，应通过将相应的 TRIS 位置 1 将 CCPx 引脚设置为输入。

注： 如果 CCPx 引脚被设置为输出，对该端口的写操作可能引发一个捕捉事件。

图 14-1：捕捉模式操作框图



输入频率不变时通过使用预分频器可以获得极佳的平均分辨率。例如，输入频率稳定时，将预分频比设置为 1:16，那么这 16 个周期的总误差为 1 个 TCY。其有效分辨率为 TCY/16，即在 20MHz 时为 12.5 ns。只有在输入频率在 16 个采样周期均“稳定”的情况下，这一技巧才有效。不使用预分频器 （1:1）时，每个采样分辨率为 TCY。

14.3.2 捕捉模式的转换

当捕捉模式改变时，可能会产生一个捕捉中断。用户应保持 CCPxIE 位清零以禁止这种中断，用户还应在发生这种运行模式的改变后将 CCPxIF 位清零。

14.3.2.1 CCP 预分频器

通过设置 CCPxM3:CCPxM0 可以选择 4 种预分频比。只要 CCP 模块被关闭或不是设置为捕捉模式，预分频器的计数器值都被清零。也就是说任何复位都将使预分频器清零。

从一种捕捉预分频设置切换为另一种设置时可产生一个中断，且预分频器的计数器不会被清零，因此第一次捕捉可能是从一个非零的预分频比开始的。例 14-1 是切换捕捉预分设置时建议采用的方法。该例也使预分频器的计数器清零且不会产生中断。

例 14-1: 改变捕捉预分频器

```
CLRF    CCP1CON         ; Turn CCP module off
MOVLW   NEW_CAPT_PS     ; Load the W reg with the new prescaler
                        ;   mode value and CCP ON
MOVWF   CCP1CON         ; Load CCP1CON with this value
```

如果要对捕捉预分频器计数器清零， CCP 模块必须设置为非捕捉模式 (比较、PWM 或 CCP 关闭模式)。

14.3.3 休眠状态下的操作

器件处于休眠模式时， Timer1 不再递增计数 （因为 Timer1 处于同步模式），但预分频器仍然对事件进行计数 (非同步)。当指定捕捉事件发生时， CCPxIF 置 1，但捕捉寄存器不会更新。如果 CCP 中断被使能，器件将从休眠中被唤醒。 16 位 TMR1 寄存器中的值**不会**被发送到 16 位捕捉寄存器中，但由于定时器并不递增计数，这个值也没有意义。这将使 CCP 引脚被用作另一个外部中断。

14.3.4 复位的影响

复位时， CCP 模块关闭，捕捉预分频器中的值被强制清零。

14.4 比较模式

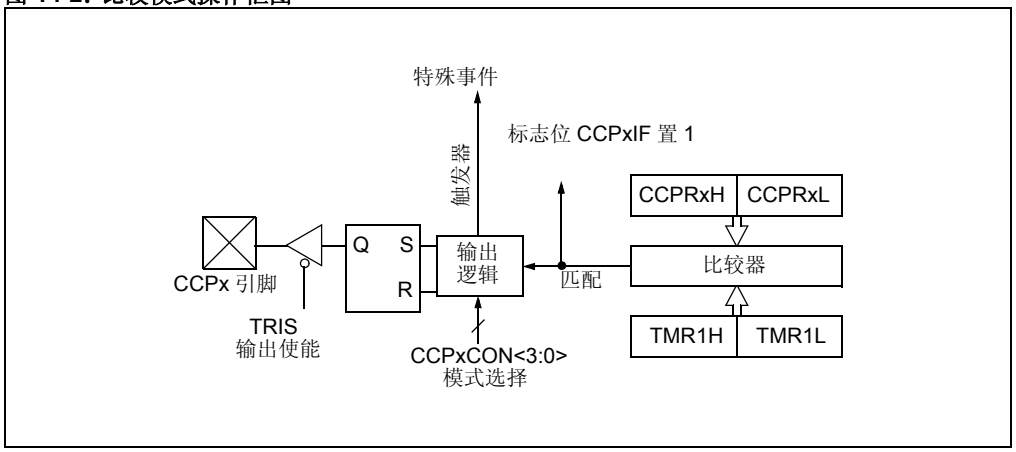
比较模式下，16 位 CCPRx 寄存器的值随时与 TMR1 寄存器对的值相比较。当两者相符时，CCPx 引脚将出现以下三种情况中的一种：

- 变为高
- 变为低
- 保持不变

引脚的状态取决于控制位 CCPxM3:CCPxM0 (CCPxCON<3:0>) 的值。同时还将产生一个比较中断。

注：	欲使 CCP 模块使用比较功能，Timer1 必须工作在定时器或同步计数器模式。在异步计数器模式下，可能无法进行比较操作。

图 14-2：比较模式操作框图



14.4.1 比较模式下 CCP 引脚的配置

比较模式下，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 寄存器将强制 CCPx 比较输出锁存器为低电平。这不是 I/O 端口的数据锁存器。

选择比较输出模式将强制 CCP 引脚为与匹配状态相反的状态。因此若将比较模式选择为匹配时强制输出引脚为低电平，则匹配条件发生前，输出引脚被强制为高电平。

14.4.2 软件中断模式

当选择了发生软件中断模式时，CCPx 引脚上的电平不受影响。只会产生一个 CCP 中断（中断使能时）。

14.4.3 特殊事件触发器

在这一模式下，将产生一个内部硬件触发信号，可用来触发一个操作：

CCPx 的特殊事件触发器输出使 TMR1 寄存器对复位。这将使 CCPRx 寄存器被用作 Timer1 的 16 位可编程周期寄存器。

对于某些器件， CCP 模块的特殊触发器输出使 TMR1 寄存器对复位，并启动 A/D 转换 (若 A/D 模块被使能)。

注： 特殊事件触发信号不会将 Timer1 的中断标志位 TMR1IF 置 1。

14.4.4 休眠模式下的操作

器件处于休眠模式时，Timer1 不再递增计数（因为 Timer1 处于同步模式），且 CCP 模块的状态保持不变。如果 CCP 引脚有输出，在休眠模式下将继续保持该输出值不变。当器件被唤醒后，输出仍然保持原来设置的状态。

14.4.5 复位的影响

复位将关闭 CCP 模块。

14.5 PWM 模式

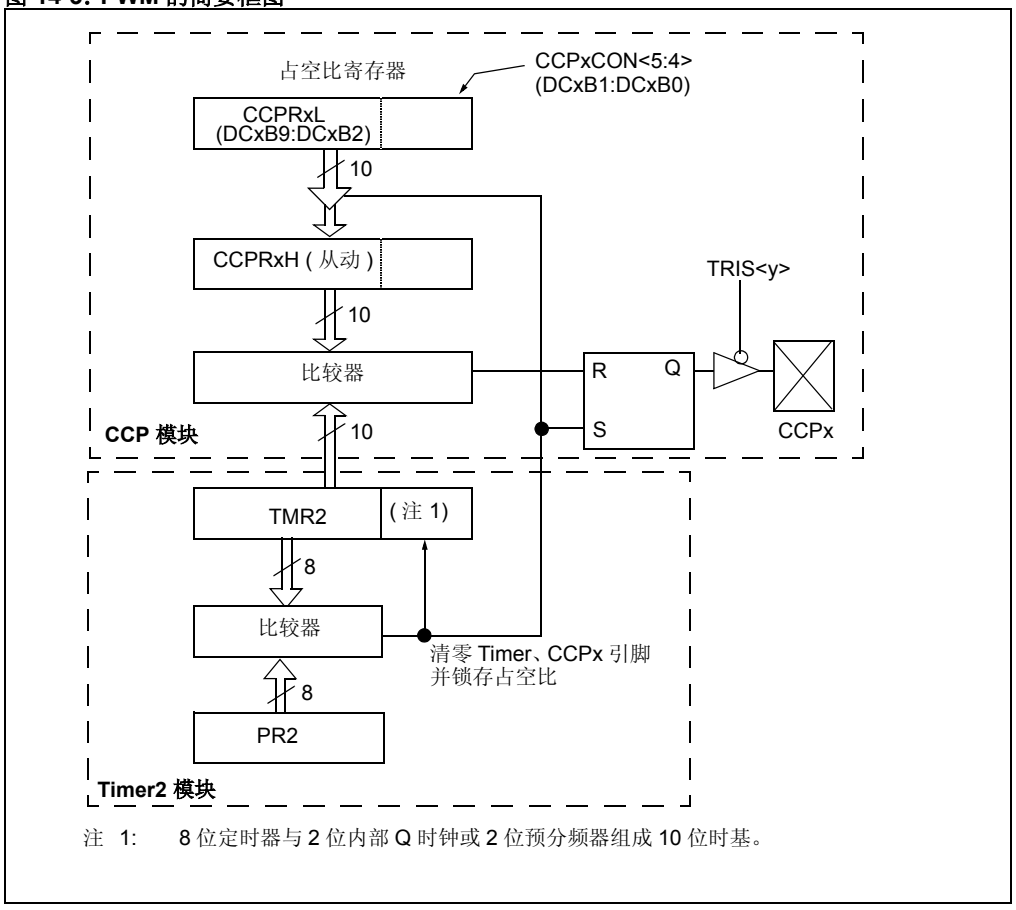
在脉冲宽度调制 (PWM) 模式下， CCPx 引脚可输出分辨率高达 10 位的 PWM 输出。因为 CCPx 引脚与端口数据锁存器是复用的，所以相应的 TRIS 位必须清零以使 CCPx 引脚为输出状态。

注：	清零 CCPxCON 寄存器将强制 CCPx PWM 的输出为低电平，而不是 I/O 端口的数据锁存器的值。

图 14-3 是工作在 PWM 模式下的 CCP 模块的简要框图。

将 CCP 模块设置成 PWM 模式的详细步骤参见 **14.5.3“设置 PWM 操作”**。

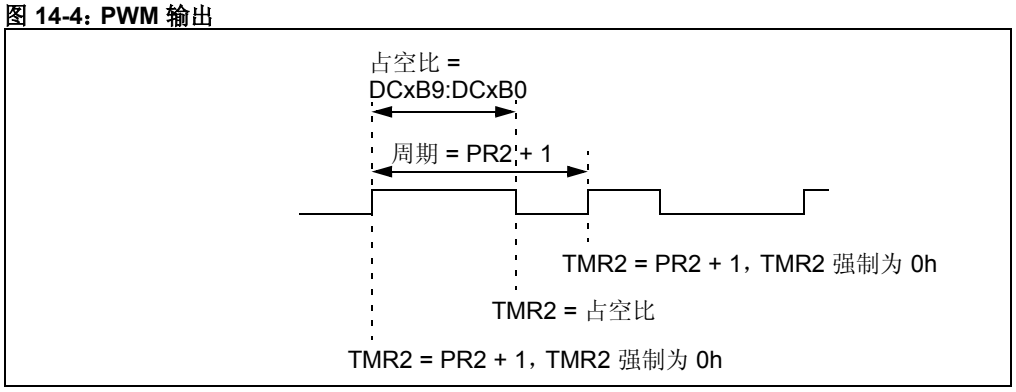
图 14-3：PWM 的简要框图



注 1: 8 位定时器与 2 位内部 Q 时钟或 2 位预分频器组成 10 位时基。

一个 PWM 输出 (如图 14-4) 包含一个时基 (周期) 和一段输出高电平的时间 (占空比)。PWM 的频率是周期的倒数 (1/ 周期)。

图 14-4：PWM 输出



14.5.1 PWM 周期

PWM 周期可通过写入 PR2 寄存器来规定，可用以下公式计算：

PWM 周期 = [(PR2) + 1] • 4 • TOSC • (TMR2 预分频比)，用时间单位表示

PWM 频率 (FPWM) 定义为 1 / [PWM 周期]。

当 TMR2 等于 PR2 时，在下一递增计数周期中将产生下面三个事件：

- TMR2 被清零
- CCPx 引脚被置 1(例外情况：如果 PWM 占空比 = 0%， CCPx 不被置 1)
- PWM 占空比从 CCPRxL 被锁定为 CCPRxH

注：	Timer2 的后分频器值与 PWM 频率无关。使用后分频器，可以用和 PWM 频率不同的速率进行相应的数据更新。

14.5.2 PWM 占空比

PWM 占空比可通过写入 CCPRxL 寄存器和 DCxB1:DCxB0 (CCPxCON<5:4>) 位来规定。最高分辨率可达 10 位：由 CCPRxL 中的高 8 位和 CCPxCON<5:4> 中的低 2 位组成。这一 10 位值由 DCxB9:DCxB0 来表征。计算 PWM 占空比的公式如下：

PWM 占空比 = (DCxB9:DCxB0) • Tosc • (TMR2 预分频比)，用时间单位表示

DCxB9:DCxB0 的值可以在任何时候写入，但直到 PR2 与 TMR2 中的值相符 (当前周期结束) 时，占空比的值才被锁存到 CCPRxH。在 PWM 模式下， CCPRxH 是只读寄存器。

CCPRxH 寄存器和一个 2 位的内部锁存器用于为 PWM 占空比提供双重缓冲。双重缓冲对 PWM 的无毛刺操作是极其重要的。

当 CCPRxH 和 2 位锁存器的值与 TMR2 和内部 2 位 Q 时钟 (或 TMR2 预分频器的 2 位) 串接值相符时， CCPx 引脚被清零。此时占空比结束。

对于给定的 PWM 频率，其最大分辨率 (位) 为：

$$= \frac{\log\left(\frac{FOSC}{FPWM}\right)}{\log(2)} \text{ 位}$$

注：	如果 PWM 占空比值大于 PWM 周期，CCPx 引脚将不会被清零。这时占空比将达到 100%。

14.5.2.2 最小分辨率

PWM 占空比中每位的最小分辨率 （时间）取决于 Timer2 的预分频器。

表 14-4: 最小占空比位时间

预分频比	T2CKPS1:T2CKPS0	最小分辨率 (时间)
1	0 0	TOSC
4	0 1	TCY
16	1 x	4 TCY

例 14-2: PWM 周期和占空比的计算

所需 PWM 频率为 78.125 kHz，
Fosc = 20 MHz
TMR2 预分频比 = 1

1/78.125 kHz = [(PR2) + 1] • 4 • 1/20 MHz • 1
12.8 µs = [(PR2) + 1] • 4 • 50 ns • 1
PR2 = 63

求可在 78.125kHz 频率和 20MHz 振荡器下使用的占空比的最大分辨率：

1/78.125 kHz = $2^{\text{PWM 分辨率}}$ • 1/20 MHz • 1
12.8 µs = $2^{\text{PWM 分辨率}}$ • 50 ns • 1
256 = $2^{\text{PWM 分辨率}}$
log(256) = (PWM 分辨率) • log(2)
8.0 = PWM 分辨率

频率为 78.125kHz 和 20MHz 的振荡器时可获得最大 8 位分辨率的占空比，即 0 ≤ DCxB9:DCxB0 ≤ 255，任何大于 255 的值将使占空比为 100%。

要得到更高分辨率，必须降低 PWM 频率。要得到更高 PWM 频率，必须降低分辨率。

表 14-5 列出了 Fosc = 20 MHz 时的 PWM 频率和分辨率的值，同时也列出了 TMR2 预分频比和 PR2 的值。

表 14-5: PWM 的频率与分辨率的关系举例 （ 20 MHz）

PWM 频率	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	208.3 kHz
定时器预分频比 (1, 4, 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	5.5

14.5.3 设置 PWM 操作

通过以下步骤将 CCP 模块配置为 PWM 模式：

1. 写入 PR2 寄存器以设定 PWM 周期。
2. 写入 DCxB9:DCxB0 位以设置 PWM 占空比。
3. 将相应的 TRIS 位清零以将 CCPx 引脚设为输出。
4. 写入 T2CON 以设置 TMR2 预分频比并使能 Timer2。
5. 将 CCP 模块配置为 PWM 模式。

14.5.4 休眠模式下的操作

器件处于休眠模式时， Timer2 停止递增计数， CCP 模块的状态保持不变。如果 CCP 引脚有输出，在休眠模式下将继续保持该输出值不变；当单片机被唤醒后，将从唤醒时的状态开始继续工作。

14.5.5 复位的影响

复位将关闭 CCP 模块。

14.6 初始化

CCP 模块有 3 种工作模式。例 14-3、例 14-4 和例 14-5 分别显示了捕捉、比较和 PWM 模式的初始化程序。

例 14-3: 捕捉模式的初始化

```
    CLRF   CCP1CON       ; CCP Module is off
    CLRF   TMR1H         ; Clear Timer1 High byte
    CLRF   TMR1L         ; Clear Timer1 Low byte
    CLRF   INTCON        ; Disable interrupts and clear T0IF
    BSF    STATUS, RP0   ; Bank1
    BSF    TRISC, CCP1   ; Make CCP pin input
    CLRF   PIE1          ; Disable peripheral interrupts
    BCF    STATUS, RP0   ; Bank0
    CLRF   PIR1          ; Clear peripheral interrupts Flags
    MOVLW  0x06          ; Capture mode, every 4th rising edge
    MOVWF  CCP1CON       ;
    BSF    T1CON, TMR1ON ; Timer1 starts to increment
;
; The CCP1 interrupt is disabled,
; do polling on the CCP Interrupt flag bit
;
Capture_Event
    BTFSS  PIR1, CCP1IF
    GOTO   Capture_Event
;
; Capture has occurred
;
    BCF    PIR1, CCP1IF  ; This needs to be done before next compare
```

例 14-4:　　比较模式的初始化

```
    CLRF   CCP1CON       ; CCP Module is off
    CLRF   TMR1H         ; Clear Timer1 High byte
    CLRF   TMR1L         ; Clear Timer1 Low byte
    CLRF   INTCON        ; Disable interrupts and clear T0IF
    BSF    STATUS, RP0   ; Bank1
    BCF    TRISC, CCP1   ; Make CCP pin output if controlling state of pin
    CLRF   PIE1          ; Disable peripheral interrupts
    BCF    STATUS, RP0   ; Bank0
    CLRF   PIR1          ; Clear peripheral interrupts Flags
    MOVLW  0x08          ; Compare mode, set CCP1 pin on match
    MOVWF  CCP1CON       ;
    BSF    T1CON, TMR1ON ; Timer1 starts to increment
;
; The CCP1 interrupt is disabled,
; do polling on the CCP Interrupt flag bit
;
Compare_Event
    BTFSS  PIR1, CCP1IF
    GOTO   Compare_Event
;
; Compare has occurred
;
    BCF    PIR1, CCP1IF   ; This needs to be done before next compare
```

例 14-5: PWM 的初始化

```
    CLRF   CCP1CON       ; CCP Module is off
    CLRF   TMR2          ; Clear Timer2
    MOVLW  0x7F          ;
    MOVWF  PR2           ;
    MOVLW  0x1F          ;
    MOVWF  CCPR1L        ; Duty Cycle is 25% of PWM Period
    CLRF   INTCON        ; Disable interrupts and clear T0IF
    BSF    STATUS, RP0   ; Bank1
    BCF    TRISC, PWM1   ; Make pin output
    CLRF   PIE1          ; Disable peripheral interrupts
    BCF    STATUS, RP0   ; Bank0
    CLRF   PIR1          ; Clear peripheral interrupts Flags
    MOVLW  0x2C          ; PWM mode, 2 LSbs of Duty cycle = 10
    MOVWF  CCP1CON       ;
    BSF    T2CON, TMR2ON ; Timer2 starts to increment
;
; The CCP1 interrupt is disabled,
; do polling on the TMR2 Interrupt flag bit
;
PWM_Period_Match
    BTFSS  PIR1, TMR2IF
    GOTO   PWM_Period_Match
;
; Update this PWM period and the following PWM Duty cycle
;
    BCF    PIR1, TMR2IF
```

14.7 设计技巧

问 1: *捕捉和比较模式可以使用哪些定时器？*

答 1:

捕捉和比较模式是基于 Timer1 设计的，所以只能使用这一定时器。并且如果（器件中带有）多个 CCP 模块工作在捕捉或比较模式下，它们将共用一个定时器。

问 2: *PWM 模式可以使用哪些定时器？*

答 2:

PWM 模式是基于 Timer2 设计的，所以只能使用这一定时器。（这是唯一带有周期寄存器的定时器。）如果（器件中带有）多个 CCP 模块工作在 PWM 模式，它们将共用一个定时器，即，它们将具有相同的 PWM 周期和频率。

问 3: *既然 CCP 的 3 种工作模式使用不同的定时器作为基准，那么 CCP 模块能否同时进行捕捉 (或比较) 和 PWM 操作？*

答 3:

不能。尽管定时器不同，但其他逻辑电路是共用的。然而，从一种模式切换到另一种模式是可以的。对于带有两个 CCP 模块的器件，可以将 CCP1 设定为 PWM 模式而 CCP2 设定为捕捉或比较模式 (或相反)，因为它们是互为独立的模块。

问 4: *复位对 CCP 模块有何影响？*

答 4:

任何一种复位都将关闭 CCP 模块。参阅有关复位的章节查看相应复位值。

问 5: *将 CCP1CON 设为 “ 比较模式，触发特殊事件 ”(1011) 并使 TMR1 复位。当比较达到匹配时，是否会同时发生 TMR1 和 CCP1 中断请求 (即 TMR1IF 和 CCP1IF 均置 1)?*

答 5:

达到匹配状态时，CCP1IF 标志将被置 1。当 Timer1 溢出时，TMR1IF 被置 1，而 Timer1 的特殊触发复位并不视为溢出。然而，如果寄存器 CCPR1L 和 CCPR1H 在 FFh 时被置 1，那么在匹配的同时将产生溢出，这将使 CCP1IF 和 TMR1IF 都被置 1。

问 6: *怎样将 Timer2 用作通用定时器，并使其在计满回零时产生中断？*

答 6:

当 Timer2 和 PR2 相等时，它将总是被复位成 0，这时标志位 TMR2IF 也被置 1。例如将 FFh 写入 PR2，那么与 Timer0 相似，Timer2 计数到 FFh 时会产生一个溢出中断。很多时候我们希望某个事件能够周期性的出现，例如中断驱动事件。通常可以将一个初始值写入定时器，以使溢出在我们期望时发生。每次溢出后，这个值将被重新写入定时器，使中断周期性地产生。Timer2 的优点在于可以将一个值写入 PR2，从而使其在您期望的时间间隔发生复位。这样，由于 PR2 中保存着这个值，便不用在每次溢出后都重新输入。

问 7: ***我在使用 CCP 模块的 PWM 模式时，占空比几乎总是 100%，甚至当我在程序中将7Fh 写入占空比寄存器时也是如此，而此时占空比应为50%。哪里出现了错误?***

答 7:

1. CCPRxL 的值高于 PR2 会造成这种情况。当用户想要获得高速的 PWM 输出频率而把一个较小的数值写入 PR2 时经常会发生这种情况。这时，如果把 7Eh 写入 PR2，把 7Fh 写入 CCPRxL 就会产生 100% 的占空比。
2. 如果您所使用的 CCP 输出引脚所对应的 TRIS 位被配置成输入，则 PWM 无法驱动该引脚。这种情况下引脚值不确定，从而占空比可能为 0%、 100% 或其他随机值。

问 8: ***我想用 CCP 模块的捕捉模式测量信号周期从而计算频率。我在第一个时钟沿将 Timer1 复位，在第二个时钟沿将捕捉寄存器里的值作为信号的周期。问题是由于中断延迟和中断时寄存器的现场保护，将定时器清零的语句在第一个捕捉边沿之后 12 个指令周期才执行，因此无法测量很高的频率。有没有更好的方法？***

答 8:

其实并不需要清零计数器来计算两个脉冲边沿的时间差。只需将第一个捕捉值存入另一组寄存器，然后当第二个捕捉发生时，用第二次的捕获值减第一次的捕获值。如果你的脉冲边沿相隔不太远，计数器不产生循环并超越上一次的捕捉值，那么结果将始终是正确的。通过以下例子可以说明：

1. 第一次捕捉值为 FFFEh。存入两个寄存器。
2. 第二次捕捉值为 0001h (计数器递增计数 3 次)。
3. 0001h - FFFEh = 0003，这与你清零 Timer1 然后让它计数到 3 的结果是相同的。(理论如此，除非执行代码前发生了延时将 Timer1 清零，这时实际值将有所不同)。

因为捕捉是自动完成的，因此这时中断的开销并不太重要。对于频率更快的输入信号，不要使能中断，只须在循环中检测中断标志位即可。如果需要捕捉周期很长的信号，即定时器循环并超越了上一次捕捉周期，则应考虑采用自动定标法，即加大 Timer1 的初始预分频比，然后接近准确频率时减小预分频比。

14.8 相关应用笔记

这里列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，通过适当修改并受到一定限制即可使用。目前与 CCP 模块相关的应用笔记有：

标题	应用笔记 #
Using the CCP Modules	AN594
Implementing Ultrasonic Ranging	AN597
Air Flow Control Using Fuzzy Logic	AN600
Adaptive Differential Pulse Code Modulation	AN643

14.9 版本历史

版本 A

这是描述 CCP 模块的初始发行版。

第 15 章 同步串行口 （SSP）

目录

本章包括下面一些主要内容：

15.1 简介 15-2
15.2 控制寄存器 15-3
15.3 SPITM 模式 15-6
15.4 SSP 模块的 I^2CTM 操作 15-16
15.5 初始化 15-26
15.6 设计技巧 15-28
15.7 相关应用笔记 15-29
15.8 版本历史 15-30

注： 关于哪些型号的器件包含此模块，请参阅附录 C.2 或器件数据手册。

15

SSP

I^2C 是 Philips 公司的商标。

15.1 简介

同步串行口（SSP）模块是一个串行接口，可用于和其它外设模块（如串行 EEPROM、移位寄存器、显示驱动器、A/D 转换器等）或者单片机之间的通信。 SSP 模块有下列两种工作模式：

- 串行外设接口（SPI™）
- 内部互联（I²C™）
  - 从动模式
  - I/O 端口输出电压斜率控制；启动位和停止位，便于用软件实现主控模式和多主机模式。

SPI 是 Motorola 公司的商标。.
I²C 是 Philips 公司的商标。

15.2 控制寄存器

寄存器 15-1： SSPSTAT：同步串行口状态寄存器

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7 **SMP**：SPI™ 输入数据的采样相位

SPI 主控模式

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 从动模式

当 SPI 为从动模式时， SMP 必须清零

bit 6 **CKE**：SPI 时钟沿选择位 （见图 15-3、图 15-4 和图 15-5）

CKP = 0 （SSPCON<4>）

1 = 在 SCK 上升沿发送数据
0 = 在 SCK 下降沿发送数据

CKP = 1 （SSPCON<4>）

1 = 在 SCK 下降沿发送数据
0 = 在 SCK 上升沿发送数据

bit 5 **D/$\overline{A}$**：数据 / 地址位 （仅用于 I^2C 模式）

1 = 表示最后接收或发送的字节是数据
0 = 表示最后接收或发送的字节是地址

bit 4 **P**：停止位
（仅用于 I^2C 模式。当 SSP 模块被禁止时该位被清零）

1 = 表示检测到停止位 （复位时该位为 '0' ）
0 = 表示未检测到停止位

bit 3 **S**：启动位
（仅用于 I^2C 模式。当 SSP 模块被禁止时该位被清零）

1 = 表示检测到启动位 （复位时该位为 '0' ）
0 = 表示未检测到启动位

bit 2 **R/$\overline{W}$**：读 / 写位信息 （仅用于 I^2C 模式）

该位用来记录在最后一次地址匹配后接收到的读 / 写信息。从本机地址与接收地址匹配开始，到下一个启动位、停止位或无应答位 （$\overline{ACK}$）时，该位有效。

1 = 读
0 = 写

bit 1 **UA**：地址更新 （仅用于 10 位 I^2C 模式）

1 = 表示需要更新 SSPADD 寄存器中的地址
0 = 表示地址不需要更新

bit 0 **BF**：缓冲区满状态位

接收时 （SPI 和 I^2C 模式）

1 = 表示接收完成，SSPBUF 满
0 = 表示接收未完成，SSPBUF 空

发送时 （I^2C 模式时）

1 = 表示发送正在进行， SSPBUF 满
0 = 表示发送已经完成， SSPBUF 空

图注：		
R = 可读位	W = 可写位	
U ： 未用位，读为 ‘0’		- n = 上电复位时的值

寄存器 15-2： SSPCON：同步串行口控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位

1 = 正在发送前一个字时，又有数据写入 SSPBUF 寄存器
（该位必须用软件清零）
0 = 表示未发生冲突

bit 6 **SSPOV**：接收溢出指示位

在 SPI 模式下：

1 = SSPBUF 中仍保持前一个数据时又收到新的字节。在溢出时， SSPSR 中的数据会丢失，而且 SSPBUF 不能再被更新。溢出只会发生在从动模式下。即使只是发送数据，用户也必须读 SSPBUF，以避免产生溢出。在主控模式下，溢出位不会被置位，因为每次接收或发送新数据，都要通过写 SSPBUF 来启动。

0 = 没有溢出

在 I^2C 模式下：

1 = SSPBUF 中仍保持前一个字节时又收到新的数据。

在发送方式下 SSPOV 位无效，在 SPI 和 I^2C 模式下，该位都必须用软件清零。

0 = 没有溢出

bit 5 **SSPEN**：同步串行口使能位
在 SPI 和 I^2C 两种模式下，当该位为 1 而使能时，应正确定义相应引脚的输入输出方向。

在 SPI 模式下：
1 = 使能串行口，并定义 SCK、 SDO、 SDI 和 SS 为串行口引脚
0 = 禁止串行口，并定义 SCK、SDO、 SDI 和 SS 引脚为一般 I/O 端口引脚

在 I^2C 模式下：
1 = 使能串行口，并定义 SDA 和 SCL 为串行口引脚
0 = 禁止串行口，并定义 SDA 和 SCL 引脚为一般 I/O 端口引脚

bit 4 **CKP**：时钟极性选择位

在 SPI 模式下：
1 = 空闲状态时，时钟为高电平
0 = 空闲状态时，时钟为低电平

在 I^2C 模式下：
SCK 释放控制
1 = 使能时钟
0 = 保持时钟线为低电平 （用于保证数据的建立时间）

寄存器 15-2： SSPCON：同步串行口控制寄存器 （续）

bit 3:0 **SSPM3:SSPM0**：同步串行口模式选择位

0000 = SPI 主控模式 , 时钟 = FOSC/4
0001 = SPI 主控模式 , 时钟 = FOSC/16
0010 = SPI 主控模式 , 时钟 = FOSC/64
0011 = SPI 主控模式 , 时钟 = TMR2 输出 /2
0100 = SPI 从动模式 , 时钟 = SCK 引脚，使能 $\overline{SS}$ 引脚控制
0101 = SPI 从动模式 , 时钟 = SCK 引脚，禁止 $\overline{SS}$ 引脚控制，$\overline{SS}$ 可用作 I/O 引脚
0110 = I^2C 7 位地址的从动模式
0111 = I^2C 10 位地址的从动模式
1000 = 保留
1001 = 保留
1010 = 保留
1011 = I^2C 固件控制的主控模式 （从动模式空闲）
1100 = 保留
1101 = 保留
1110 = I^2C 从动模式
7 位地址，允许启动位和停止位中断功能
1111 = I^2C 从动模式
10 位地址，允许启动位和停止位中断功能

图注：	
R = 可读位	W = 可写位
U = 未用位，读为 ‘0’	- n = 上电复位时的值

15.3 SPI™ 模式

SPI 模式允许同时同步发送和接收 8 位数据。支持 SPI 的四种工作模式，当 SPI 处于主控模式时还支持 Microwire™ （边沿采样）。

以下三个引脚来完成通信：

- 串行数据输出 （SDO）
- 串行数据输入 （SDI）
- 串行时钟 （SCK）

当工作在从动模式时，可能还需要第 4 个引脚：

- 从动选择 （$\overline{SS}$）

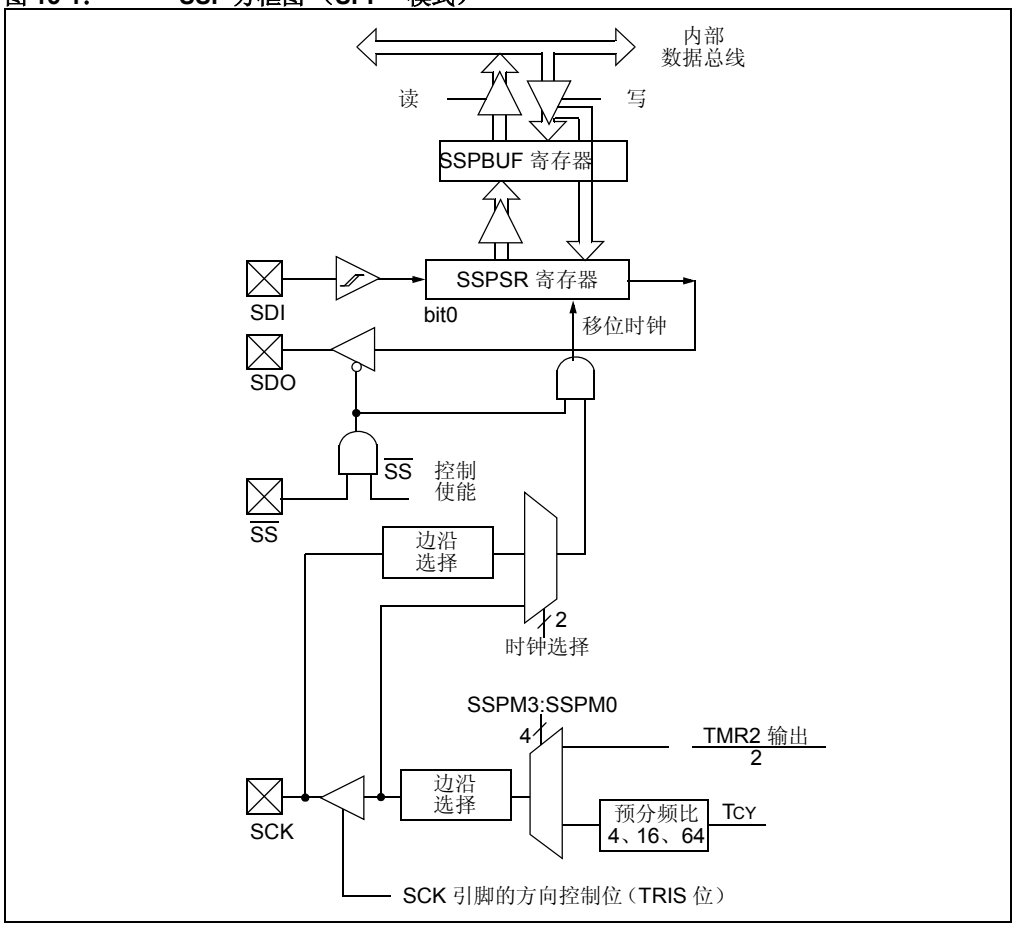
15.3.1 操作

初始化 SPI 时，必须通过设置 SSPCON 寄存器中的相应控制位 （SSPCON<5:0>）和 SSPSTAT<7:6> 来指定以下各项：

- 主控模式 （SCK 作为时钟输出）
- 从动模式 （SCK 作为时钟输入）
- 时钟极性 （空闲时 SCK 的状态）
- 时钟边沿 （决定是在 SCK 的上升沿还是下降沿输出数据）
- 输入数据的采样相位
- 时钟速率 （仅用于主控模式）
- 从动选择模式 （仅用于从动模式）

图 15-1 给出了工作在 SPI 模式下 SSP 模块的方框图。

图 15-1： SSP 方框图 （SPI™ 模式）



SSP 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 用于器件输入和输出数据的移位，最高有效位在前。在新的数据接收完毕前，SSPBUF 保存写入 SSPSR 的数据。一旦 8 位新数据接收完毕，该字节被送入 SSPBUF 寄存器。同时缓冲区满标志位 BF（SSPSTAT <0>）和中断标志位 SSPIF 置 1。这种双重缓冲接收方式，允许接收的数据被 CPU 读取之前，开始接收下一个数据。在数据发送 / 接收期间，任何试图写 SSPBUF 寄存器的操作都无效，会将冲突检测位 WCOL （SSPCON<7>）置 1。此时用户必须用软件将 WCOL 位清零，否则无法判别下一次对 SSPBUF 的写操作是否成功。当应用软件要接收一个有效数据时，应该在下一个要传送的数据写入 SSPBUF 之前，将 SSPBUF 中的前一个数据读出。缓冲器满标志位 BF（SSPSTAT<0>）用于表示何时把接收到的数据送入 SSPBUF 寄存器（传输完成）。当 SSPBUF 中的数据被读出后，BF 位即被清零。如果 SPI 仅仅作为一个发送器，则不必理会接收的数据。通常可用 SSP 中断来判断发送或接收是否完成。必须读并 / 或写 SSPBUF。如果不使用中断来处理数据的收发，用软件查询方法同样可确保不会发生写冲突。例 15-2 举例说明了在发送数据过程中如何写 SSPBUF （SSPSR），阴影部分的指令仅在需要接收数据时才使用 （而某些 SPI 应用只发送数据）。

例 15-1： 对 SSPBUF （SSPSR）寄存器的写操作

```
     BCF   STATUS, RP1    ;Specify Bank1
     BSF   STATUS, RP0    ;
LOOP BTFSS SSPSTAT, BF    ;Has data been received (transmit complete)?
     GOTO  LOOP           ;No
     BCF   STATUS, RP0    ;Specify Bank0
     MOVF  SSPBUF, W      ;W 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
```

SSPSR 寄存器不能直接读写，只能通过对 SSPBUF 寄存器寻址来访问。SSP 模块的状态寄存器 SSPSTAT 可用来指示各种状态条件。

Microwire 是 National Semiconductor 公司的商标。

15.3.2 使能 SPI™ 的 I/O 引脚

要使能 SSP 串行口，必须将 SSP 使能位 SSPEN 位 （SSPCON<5>）置位。要复位或重新配置 SPI模式，先将 SSPEN 位清零，对 SSPCON 重新初始化，然后把 SSPEN 位置 1。这将设定 SDI、SDO、SCK 和 SS 引脚为 SSP 串行口引脚。要将这些引脚用于串行口功能，还必须通过 TRIS 寄存器设置正确的方向，即：

- SDI 定义成输入
- SDO 定义成输出
- 主控模式时， SCK 定义成输出
- 从动模式时， SCK 定义成输入
- SS 定义成输入

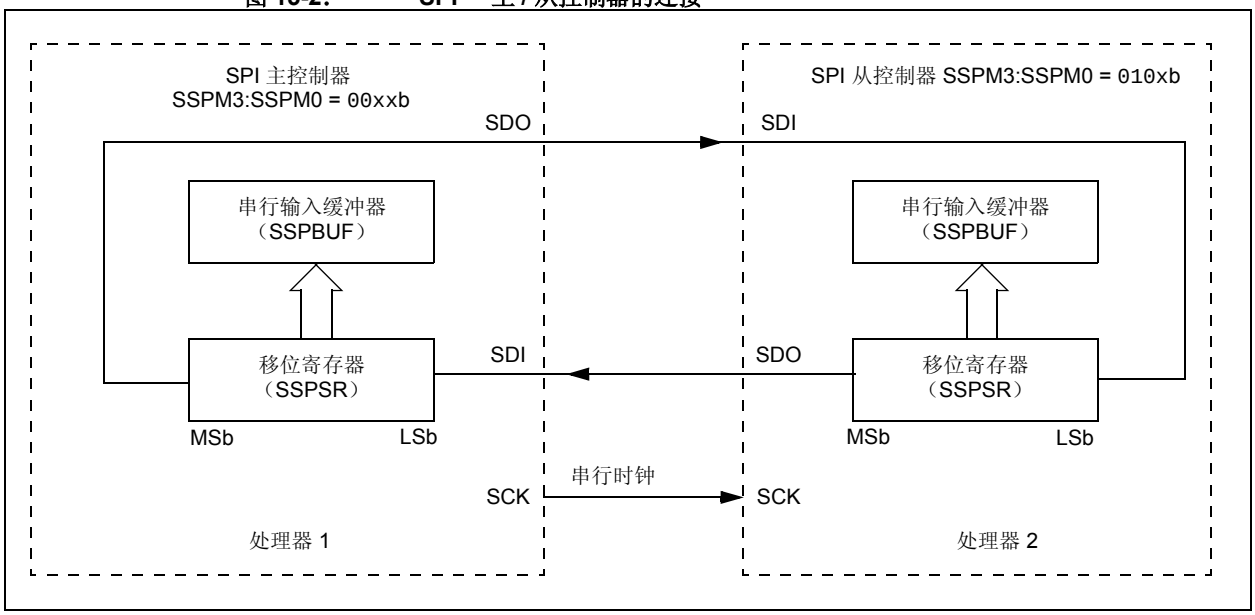
对于不需要的同步串行口功能，可以通过把相应的方向寄存器 （TRIC）设置为上述的相反值而另作它用。例如在主控模式下，如果只发送数据 （如发送到显示驱动电路），那么通过将 SDI 和 SS 引脚的相应 TRIS 寄存器方向位清零，就可以把这两个引脚作为通用的输出口使用。

15.3.3 典型连接

图 15-2 给出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。根据程序设定的时钟边沿，分别位于两个处理器里的移位寄存器中的数据同时被移出，并在 SMP 位指定的时钟边沿被锁存。两个处理器的时钟极性 （CKP）必须编程设定为相同，这样两个处理器就可以同时收发数据。至于数据是否有意义（或是无效“哑”数据）则取决于应用软件，这就导致以下三种数据发送方式：

- 主控制器发送数据 — 从控制器发送无效数据 （“哑”数据）。
- 主控制器发送数据 — 从控制器发送数据
- 主控制器发送无效数据 — 从控制器发送数据

图 15-2： SPI™ 主 / 从控制器的连接



15.3.4 主控模式操作

因为主控制器控制着 SCK 信号，所以它可以在任何时候启动数据传输，同时主控制器通过软件协议来决定从控制器 （处理器 2）何时传送数据。

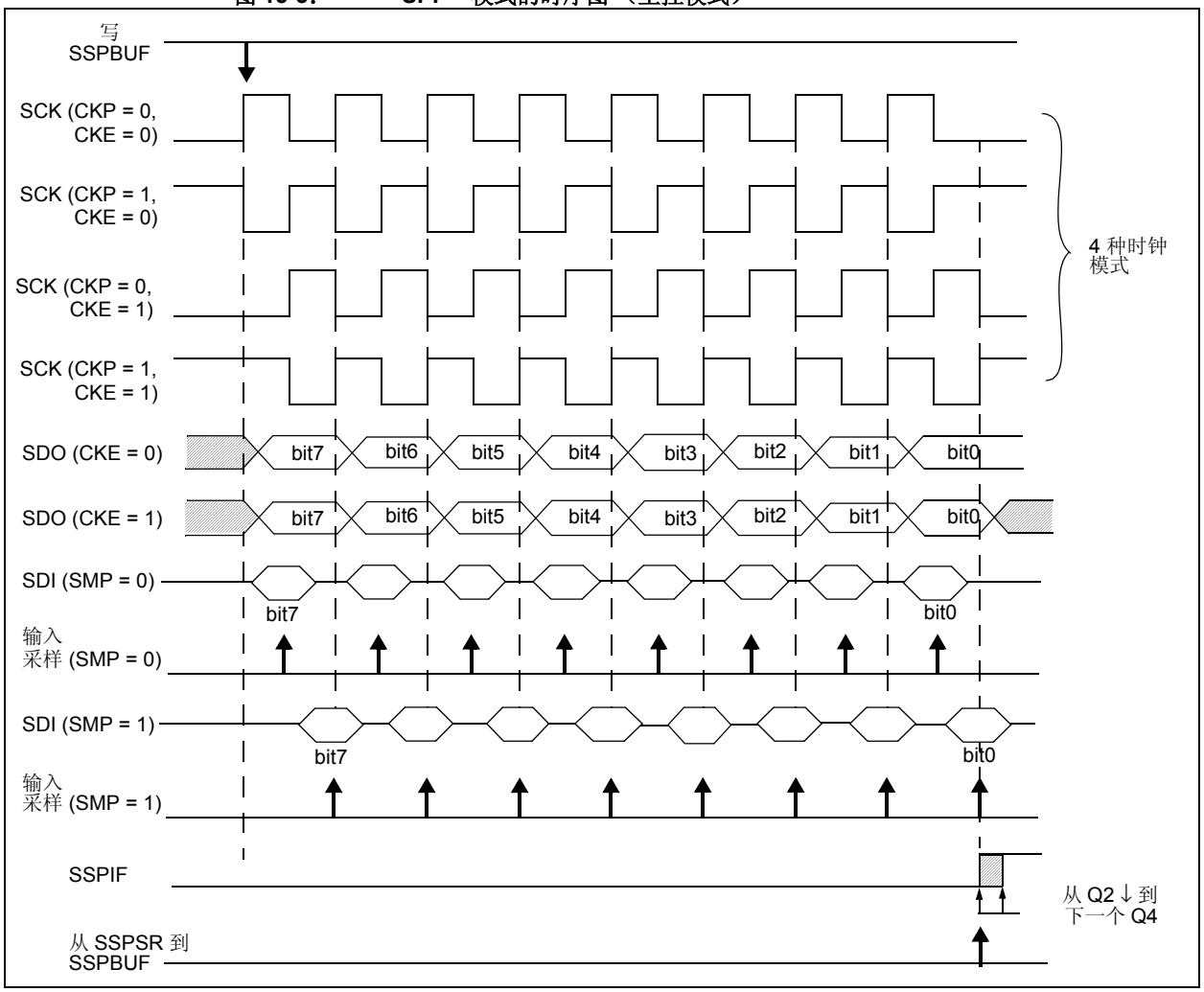
在主控模式下，数据一旦写入 SSPBUF 就开始发送或接收。如果 SPI 仅作为接收器，则可以禁止 SDO 输出 （将其设置为输入端口）。 SSPSR 寄存器按设置的时钟速率，对 SDI 引脚上的信号进行连续地移位输入。每接收完一个字节，都把其送入 SSPBUF 寄存器，就象普通的接收字节一样 （相应的中断和状态位置 1）。这在 “线路主动监控” 方式的接收器应用中可能是很有用的。

时钟极性可通过对 SSPCON 寄存器的 CKP 位（SSPCON<4>）编程来设定。图 15-3、图 15-4 和图 15-5 是 SPI 通信的时序图，最高位首先发送。在主控模式下， SPI 时钟速率（位速率）可由用户编程设定为下面几种方式之一 ：

- FOSC/4 （或 TCY）
- FOSC/16 （或 4 • TCY）
- FOSC/64 （或 16 • TCY）
- 定时器 2 输出速率 /2

最大数据通信速率是 5 Mbps （当晶振为 20 MHz 时）。

图 15-3： SPI™ 模式的时序图 （主控模式）



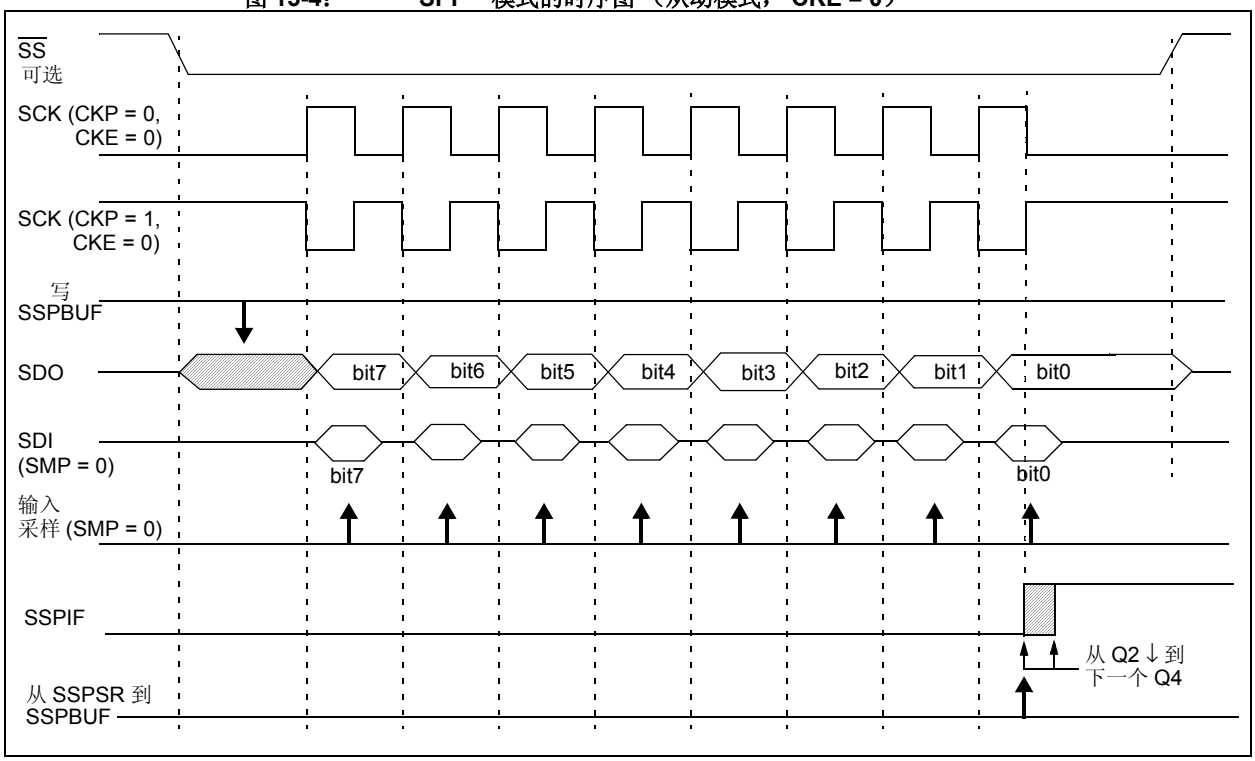
15.3.5 从动模式的操作

在从动模式下，当 SCK 引脚上出现外部时钟脉冲时，发送 / 接收数据。当最后一位数据锁存后，中断标志位 SSPIF 置 “1”。

时钟极性通过对 SSPCON 寄存器的 CKP 位（SSPCON<4>）编程来设定。图 15-3、图 15-4 和图 15-5 是 SPI 通信的时序图，其中最高位先被发送。在从动模式下，外部时钟必须满足最短高电平和低电平的脉宽要求。

在休眠模式下，从控制器仍可发送和接收数据。如果允许中断，接收到数据时还可唤醒单片机。

图 15-4： SPI™ 模式的时序图 （从动模式， CKE = 0）



15.3.6 从动选择模式

在从动选择模式下，通过 $\overline{SS}$ 引脚可以将多个从动模式器件和一个主控模式器件连接在一起工作。要将 SPI 设置成从动选择模式， SPI 必须工作在从动模式 （SSPCON<3:0> = 04h），并将 $\overline{SS}$ 引脚的 TRIS 位置位。当 $\overline{SS}$ 引脚为低电平时，允许数据的发送和接收，同时 SDO 引脚被驱动为高电平或低电平。当 $\overline{SS}$ 引脚为高电平时，即使是在数据的发送过程中，SDO 引脚也不再被驱动，而是变成高阻悬浮状态。根据应用的需要，可在 SDO 引脚上外接上拉或下拉电阻。

如果 SPI 工作在从动模式且使能 $\overline{SS}$ 引脚控制 （SSPCON<3:0> = 0100），如果 $\overline{SS}$ 引脚置成 V_{DD} 电平将复位 SPI 模块。在 SPI 从动模式时，如果 CKE 位置“1”，那么 $\overline{SS}$ 引脚控制必须使能。

当 SPI 模块复位时，位计数器被强制为零。通过将 $\overline{SS}$ 引脚置 1 或者 SSPEN 位清零，也可将位计数器强制清零 （如图 15-6）。

将 SDO 引脚和 SDI 引脚相连，可以仿真二线制通信。当 SPI 作为接收器时，可将 SDO 引脚定义为输入，这样就禁止 SDO 引脚发送数据。而 SDI 总是定义为输入，因为它不会引起总线冲突。

图 15-5： SPI™ 模式时序 （CKE = 1 时的从动选择模式）

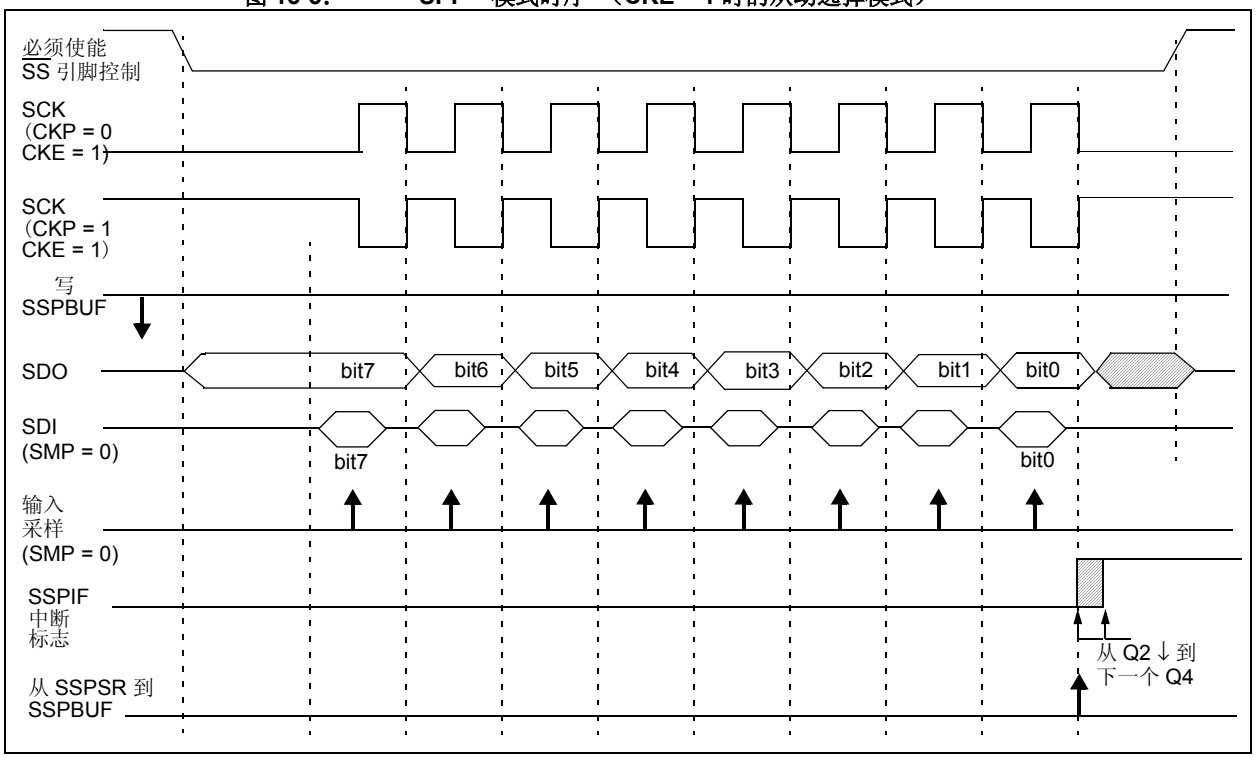
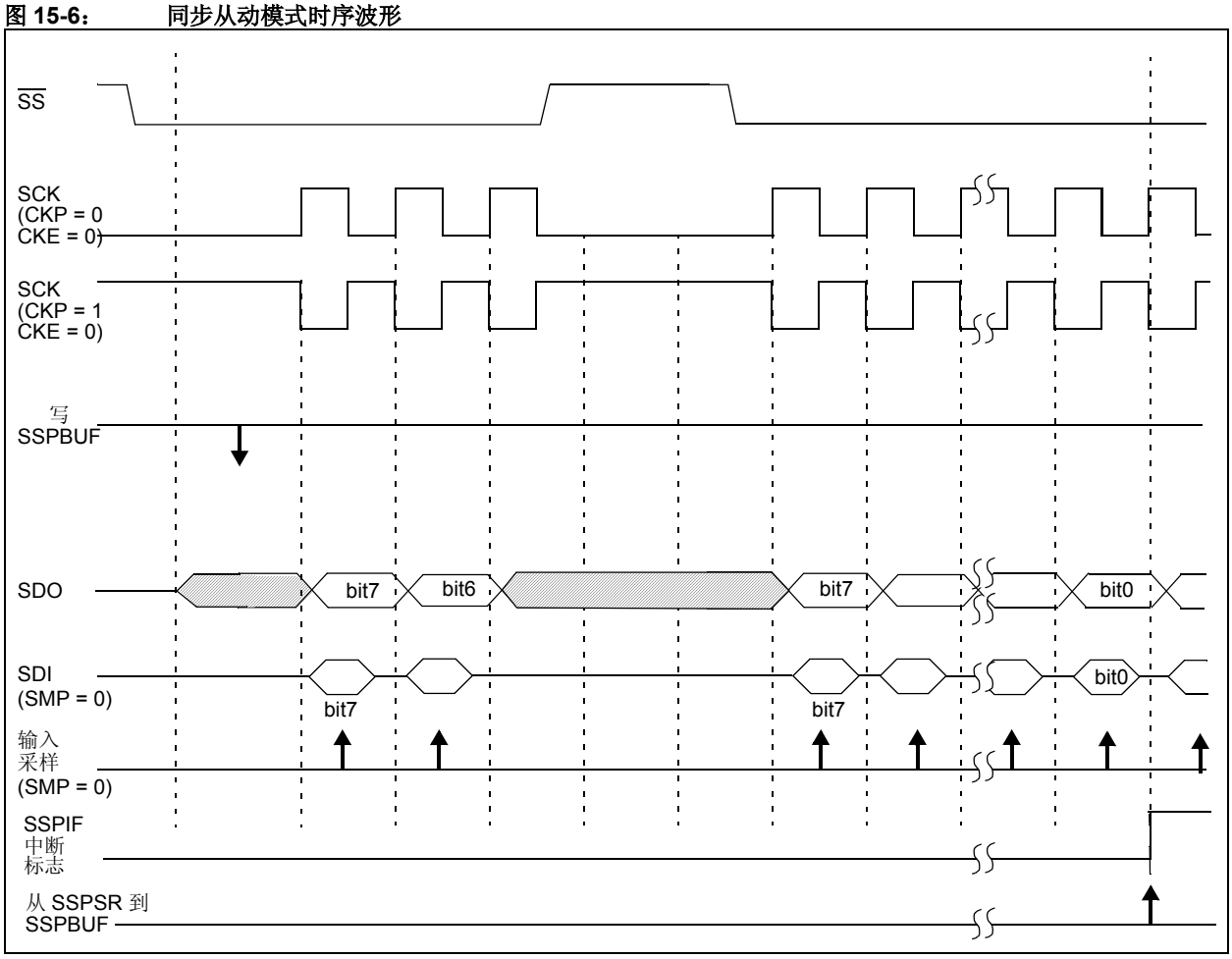


图 15-6：　同步从动模式时序波形



15.3.7 休眠状态下的操作

在主控模式下，此时所有模块的时钟都停止了，在器件被唤醒前，发送 / 接收也处于停滞状态。在器件恢复正常工作状态后，模块将继续数据的发送 / 接收。

在从动模式下，SPI 发送 / 接收移位寄存器与器件异步工作，所以在休眠状态时，数据仍可被移入 SPI 发送 / 接收移位寄存器。当接收完 8 位数据后，SSP 中断标志位将置 1，如果此时该中断是使能的，将唤醒器件。

15.3.8 复位的影响

复位会禁止 SSP 模块并停止当前的数据传输。

表 15-1： 和 SPI™ 操作有关的寄存器

寄存器名	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位、欠压复位时的值	其它复位状态时的值
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF (1)								0	0
PIE	SSPIE (1)								0	0
SSPBUF	同步串行口接收缓冲器 / 发送寄存器								xxxx xxxx	uuuu uuuu
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
TRISA	—	—	PORTA 数据方向寄存器						--11 1111	--11 1111
TRISC	PORTC 数据方向控制寄存器								1111 1111	1111 1111
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000

其中： x = 未知， u = 不变， - = 未用，读为 ‘0’，阴影部分在 SPI 模式下的 SSP 中未用。

注 1：该位的位置和器件的具体型号有关。

2：这些位也可被称为 GPIE 和 GPIF。

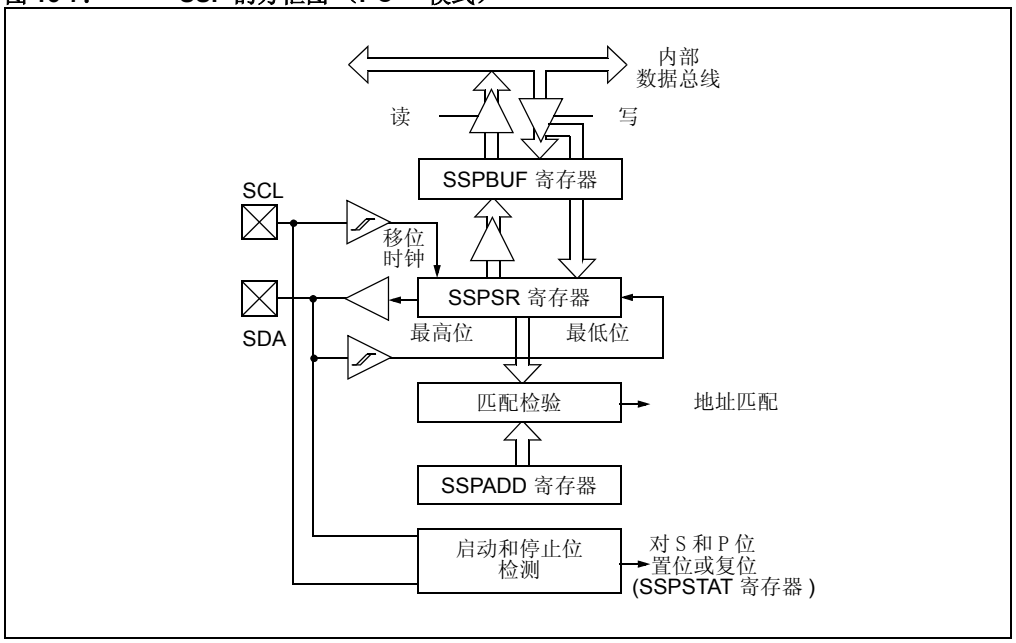
15.4 SSP 模块的 I²C™ 操作

SSP 模块工作在 I²C 模式时，除了不支持全局呼叫外，可以完成所有的从动模式功能，并且硬件上提供启动位和停止位中断，以便软件实现主控功能。SSP 模块实现了标准和快速模式规范，并支持对 7 位和 10 位地址的寻址。附录 A 给出了 I²C 总线规范的概述。

数据传输使用两个引脚：一个是作为时钟线的 SCL 引脚，另一个是作为数据线的 SDA 引脚。用户应通过 TRIC 寄存器把这些引脚设置成输入。将 SSP 使能位 SSPEN （SSPCON<5>）置位，可使能 SSP 模块功能。

当 SCL 和 SDA 引脚为输入时，引脚上有窄脉冲（毛刺）滤波器，该滤波器可以工作在 100 KHz 和 400 KHz 两种模式下。在 100 KHz 模式下， 当这些引脚作为输出时，可对引脚进行压摆率控制，此控制和器件频率无关。

图 15-7： SSP 的方框图 （I²C™ 模式）



SSP 模块有 5 个寄存器用于 I^2C 操作，它们是：

- SSP 控制寄存器 （SSPCON）
- SSP 状态寄存器 （SSPSTAT）
- 串行接收 / 发送缓冲器 （SSPBUF）
- SSP 移位寄存器 （SSPSR）— 不可直接访问
- SSP 地址寄存器 （SSPADD）

SSPCON 寄存器用于控制 I^2C的工作模式。可通过设置四个模式选择位 (SSPCON<3:0>) 来选择以下几种 I^2C 模式：

- I^2C 从动模式 （7 位地址）
- I^2C 从动模式 （10 位地址）
- I^2C 固件控制的多主机模式 （允许启动位和停止位中断）
- I^2C 固件控制的多主机模式 （允许启动位和停止位中断）
- I^2C 固件控制的主控模式，从动模式空闲

在选择 I^2C 工作模式前，通过置位相应的 TRIS 位，将 SCL 和 SDA 引脚定义为输入。然后选择I^2C工作模式，通过将SSPEN 位置1，使能SCL和 SDA引脚分别作为 I^2C模式的时钟线和数据线。

SSPSTAT 寄存器提供数据传输的状态，其中包括启动位和停止位的检测、确定接收的字节是数据还是地址、下一个字节是否已完成 10 位地址的传送，以及是数据传输是读操作还是写操作。

SSPBUF 寄存器存放要读 / 写的传输数据， SSPSR 寄存器将数据移入或移出器件。接收时，SSPBUF和SSPSR构成了一个双重缓冲接收器，允许在前一个数据读出前即可接收下一个数据。当接收完字节后，将其送入 SSPBUF 寄存器，同时置位中断请求标志位 SSPIF。如果在 SSPBUF 寄存器中的前一个数据被读走前，又接收到一个新数据， 就会发生接收器溢出， SSPOV 位（SSPCON<6>）将置 1，且 SSPSR 中的数据将丢失。

SSPADD 寄存器用于保存从机地址。在 10 位地址模式时，用户需要写入地址的高字节（1111 0 A9 A8 0）。在高字节地址匹配后，再写入地址的低字节 （A7:A0）。

15.4.1 从动模式

在从动模式下， SCL 和 SDA 引脚必须设置为输入 （相应的 TRIS 位置 1）。在需要时 （如在从动发送器情况下）， SSP 模块会强行将输入状态改变为输出状态输出数据。

当地址匹配时或在地址匹配后传送的数据被接收时，硬件会自动产生一个应答（$\overline{ACK}$）脉冲，并把在 SSPSR 中接收到的数据装入 SSPBUF 缓冲区。

满足下列条件之一， SSP 模块不会产生应答脉冲 $\overline{ACK}$：

a) 在传送的数据被接收之前，缓冲区满标志位 BF （SSPSTAT<0>）已被置 1。

b) 在传送的数据被接收之前，溢出标志位 SSPOV （SSPCON<6>）已被置 1。

此时，移位寄存器 SSPSR 的值不会装入缓冲区 SSPBUF 中，但是中断标志位 SSPIF 和 SSPOV 位仍会置 1。表 15-2 列出了在给定 BF 和 SSPOV 位状态时，数据的接收情况。阴影部分表示用户软件没有对溢出状态进行适当清零的情况。标志位 BF 是通过读取 SSPBUF 进行清零的，而 SSPOV 位必须通过软件来清零。

SCL 时钟输入的高电平和低电平必须满足最小脉宽的要求才能正常工作。关于 I^2C 规范所规定的高低电平脉宽以及对 SSP 模块的具体要求，请参见器件数据手册 **“电气规范”一章**的参数 100 和 101。

15.4.1.1 寻址

一旦 SSP 模块被使能，它就等待 START （启动）信号出现。在启动信号出现后， 8 位数据被移入 SSPSR 寄存器。所有移入的位都在时钟线 SCL 的上升沿采样。在 SCL 时钟的第 8 个脉冲下降沿，SSPSR<7:1> 的值与地址寄存器 SSPADD 的值作比较，如果地址匹配， BF 和 SSPOV 位就被清零，并完成下列操作：

a) 在第 8 个 SCL 脉冲的下降沿，把 SSPSR 寄存器的值装入 SSPBUF 寄存器。
b) 缓冲器满标志位 BF 在第 8 个 SCL 脉冲的下降沿被置 1。
c) 产生 $\overline{\text{ACK}}$ 脉冲。
d) 在第 9 个 SCL 脉冲的下降沿，SSP 中断标志位 SSPIF 置 1（如果允许中断，则产生中断）。

在 10 位地址模式时，从动器件需要接收两个地址字节。第一个地址字节的高 5 位指定这是否是一个 10 位地址。R/$\overline{\text{W}}$ 位 （SSPSTAT<2>）必须指定为写操作，这样从动器件就会接收第二个地址字节。对于 10 位地址，高字节应该是 ‘1111 0 A9 A8 0’，其中 A9 和 A8 是 10 位地址的两个最高位。10 位地址的工作步骤如下，其中 7- 9 步是针对从动发送器而言的：

1. 接收地址的第一个 （高）字节 （SSPIF、 BF 和 UA （SSPSTAT<1>）位被置 1）。
2. 用地址的第二个字节（10 位地址的低 8 位）更新 SSPADD 寄存器（对 UA 位清零同时释放 SCL 时钟线）。
3. 读 SSPBUF 寄存器 （BF 位被清零）并清零中断标志位 SSPIF。
4. 接收地址第二个字节 （SSPIF、 BF 和 UA 被置 1）。
5. 用地址的高字节更新 SSPADD 寄存器，这将使 UA 位清零并释放 SCL 时钟线。
6. 读 SSPBUF 寄存器 （BF 位清零）并清零中断标志位 SSPIF。
7. 接收再次出现的启动 （START）信号。
8. 接收地址的第一 个 （高 ）字节 (SSPIF 和 BF 位被置 1)。
9. 读 SSPBUF 寄存器 （BF 位清零）并清零中断标志位 SSPIF。

注：	10 位地址模式下，在重复启动 （RESTART）条件 （第 7 步）出现后，用户只需要匹配开始的 7 位地址，不需要更新 SSPADD 寄存器以匹配另一半地址。

表 15-2： 传输接收数据字节的情况

数据接收时的状态位				
BF	**SSPOV**	**SSPSR → SSPBUF**	**产生 $\overline{\text{ACK}}$ 脉冲**	**置位 SSPIF 位 （如允许中断，则产生 SSP 中断）**
0	0	是	是	是
1	0	否	否	是
1	1	否	否	是
0	1	是	否	是

注：阴影部分表示用户软件没有正确清除溢出条件时的情况。

15.4.1.2 接收

当地址字节的 R/$\overline{W}$ 位是零且地址匹配时， SSPSTAT 寄存器中的 R/$\overline{W}$ 位被清零，同时把接收的地址装入缓冲器 SSPBUF。

当发生地址字节接收溢出时，则不会产生应答信号（$\overline{ACK}$）。溢出条件是指 BF 位（SSPSTAT<0>）置 1 或 SSPOV 位 （SSPCON<6>）置 1。在这种情况下，接收完一个字节并试图从 SSPSR 寄存器移位到 SSPBUF 寄存器时，将不会产生应答信号 （$\overline{ACK}$）。

每个数据传输字节都会产生一个 SSP 中断，SSPIF 标志位必须用软件清零。状态寄存器 SSPSTAT 用于确定接收字节的状态。

图 15-8： I^2C^{TM} 模式接收时序图 （7 位地址）

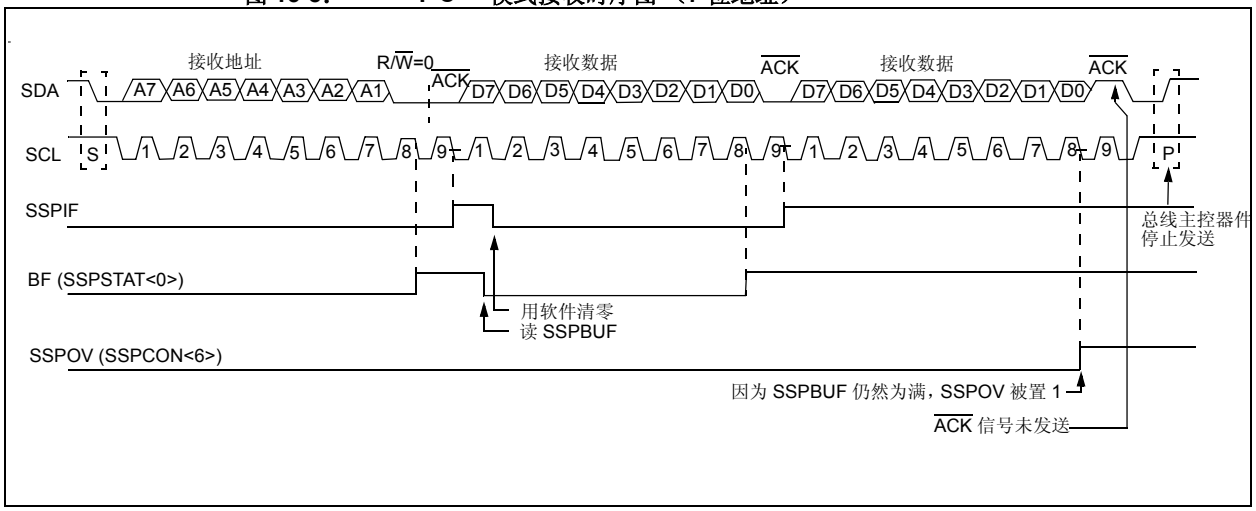
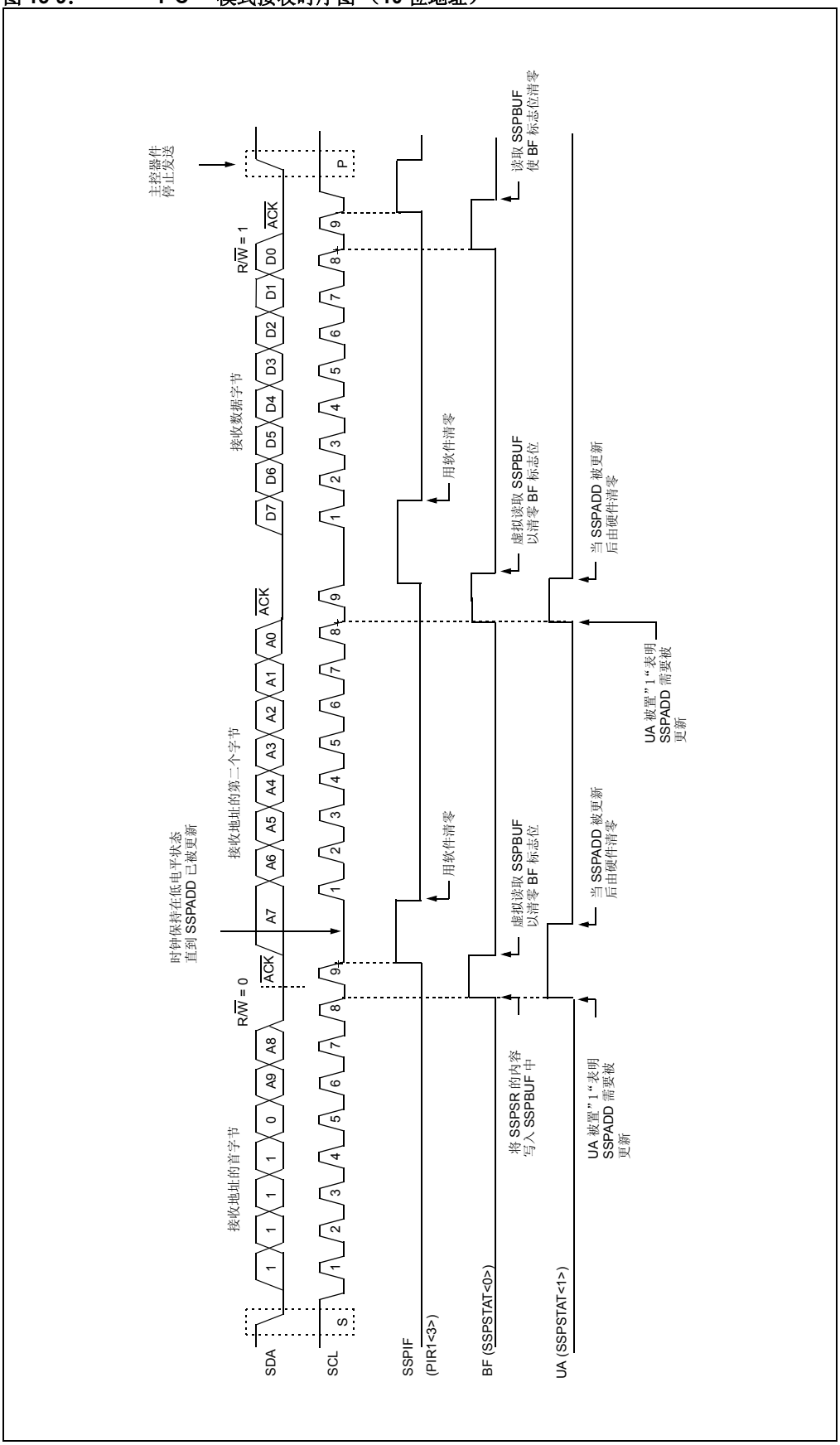


图 15-9：　　I^2C^{TM} 模式接收时序图 （10 位地址）



15.4.1.3 发送

当输入地址字节的 R/$\overline{W}$ 位置 1 且地址匹配时，状态寄存器 SSPSTAT 的 R/$\overline{W}$ 位被置 1。接收到的地址被装入缓冲器 SSPBUF。应答信号 $\overline{ACK}$ 在 SCL 的第 9 个脉冲时发送，同时 SCL 引脚保持低电平。发送的数据必须送入 SSPBUF 缓冲器，同时也送入 SSPSR 寄存器。然后通过把 CKP 位 （SSPCON<4>）置 1，使能 SCL 引脚。主控器件必须监视 SCL 引脚脉冲信号。从动器件可以通过延长 SCL 时钟的低电平，而使主控器件处于数据等待状态。8 位数据在 SCL 时钟的下降沿被移位输出。这可确保在 SCL 为高电平期间 SDA 信号是有效的 （如图 15-10)。

每个数据发送字节都会产生一个 SSP 中断， SSPIF 标志位必须通过软件清零，状态寄存器 SSPSTAT 用于确定字节传输的状态。SSPIF 标志位是在第 9 个脉冲下降沿被置 1。

作为从动发送器，在第 9 个 SCL 时钟脉冲的上升沿锁存从主控接收器发出的 $\overline{ACK}$ 信号。若 SDA 线为高电平 （无 $\overline{ACK}$ 应答信号 ），那么表示数据传输已完成。当无 $\overline{ACK}$ 应答被从动器件锁存时，从动逻辑被复位，并再监测下一个 START 信号的发生。如果 SDA 线是低电平 （有 $\overline{ACK}$ 应答信号），发送数据应装入 SSPBUF 缓冲器，并装入 SSPSR 寄存器，然后将 CKP 置 1，使能 SCL （即释放 SCL）。

图 15-10： I^2C^{TM} 模式的发送时序图 （7 位地址）

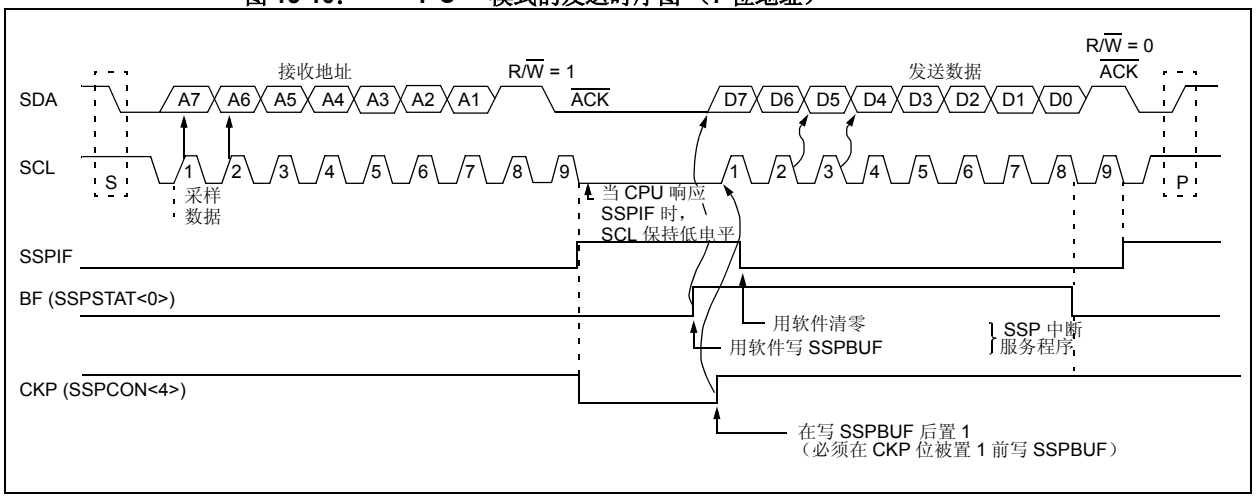
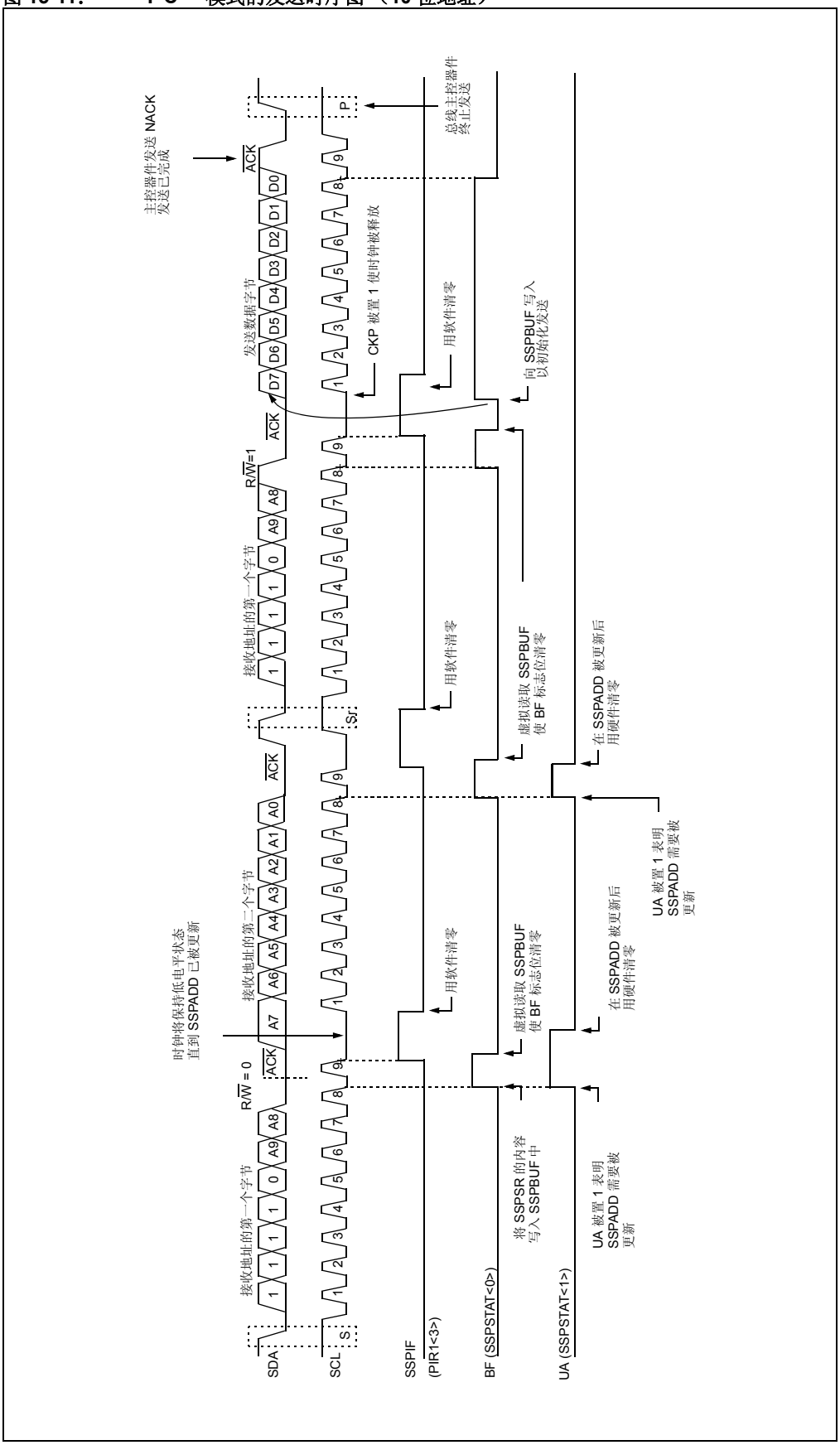


图 15-11： I^2C^{TM} 模式的发送时序图 （10 位地址）



15.4.1.4 时钟仲裁

时钟仲裁是指在 SCL 引脚上抑制主控器件发送下一个时钟脉冲。当 CPU 需要响应 SSP 中断时（SSPIF 位置 1，CKP 位清零），处在 I^2C 从动模式的 SSP 模块将保持 SCL 引脚为低电平。从动器件需要将要发送的数据写入 SSPBUF 寄存器，然后将 CKP 位置 1，以允许主控器件发出所需的时钟信号。

15.4.2 主控模式（固件）

主控模式是通过检测启动（START）和停止（STOP）条件产生中断来工作的。STOP（P）和 START（S）位在复位或禁止 SSP 模块时被清零。当 P 位被置 1，或 P 位和 S 位都清零而总线是空闲时，可以获得对 I^2C 总线的控制权。

在主控模式下，SCL 和 SDA 线是通过改变相应的 TRIS 方向控制位来控制的。不管 PORT 寄存器中的值是什么，其输出电平总是为低电平。所以在发送数据时，发送 '1' 时必须把 TRIS 的相应位置 1（设为输入），发送 '0' 时把该位清零（设为输出）。对于 SCL 线，可采用同样的方法对相应的 TRIS 位进行控制。

下列事件会引起中断标志位 SSPIF 置 1（如果允许中断，便产生中断）：

- 启动条件（START）
- 停止条件（STOP）
- 数据字节的发送 / 接收

在从动模式空闲状态（SSPM3:SSPM0 = 1011）或从动模式有效状态（SSPM3:SSP0 = 1110 或 1111），都可以运行主控模式。当从动模式有效时，需要用软件来判断中断源。

15.4.3 多主机模式（固件）

在多主机模式下，利用检测启动条件（START）和停止条件（STOP）产生中断的功能，可以判断总线何时空闲。在复位或禁止 SSP 模块时，停止位（P）和启动位（S）位清零。当停止位 P 位置 1 或 P 位和 S 位都为零而总线是空闲时，可以对 I^2C 总线进行控制操作。当总线处于忙状态且允许 SSP 中断时，一旦检测到停止条件便产生中断。

在多主机模式中，SDA 线必须一直被检测以判断信号电平是否是所期望的输出电平。如果期望高电平输出，但检测的是低电平，器件就需要释放 SDA 和 SCL 线（把 TRIS 的相应位置 1）。有两种情况可能会丢失对总线的控制：

- 地址传输
- 数据传输

当允许从动模式时，从动器件将连续接收。如果在地址传输阶段失去总线控制机会，器件仍可被其它主机寻址，若被其它主机寻址，将产生应答信号 $\overline{ACK}$ 脉冲。如果在数据传输期间失去总线控制机会，则器件需要在以后重新传输数据。

15.4.4 休眠操作

在休眠模式下，I^2C 模块能够接收地址或数据。并且地址匹配或字节传输完成后，如果允许 SSP 中断，将唤醒处理器。

15.4.5 复位的影响

复位会禁止 SSP 模块并停止当前的传输。

表 15-3： 与 I^2C^{TM} 操作有关的寄存器

寄存器名	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位、欠压复位时的值	其它复位时的值
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF (1)								0	0
PIE	SSPIE (1)								0	0
SSPBUF	同步串行口接收缓冲器 / 发送寄存器								xxxx xxxx	uuuu uuuu
SSPADD	同步串行口（I^2C 模式）地址寄存器								0000 0000	0000 0000
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000

其中： x = 未知， u = 不变， - = 未用，读作 ‘0’。
阴影部分在 I^2C 模式下的 SSP 中未用。

注 1：该位的位置和器件的具体型号有关。
2：这些位也可以称为 GPIE 和 GPIF。

15.5 初始化

例 15-2： SPI™ 主控模式的初始化

```
    CLRF   STATUS     ; Bank 0
    CLRF   SSPSTAT    ; SMP = 0, CKE = 0, and clear status bits
    BSF    SSPSTAT, CKE ; CKE = 1
    MOVLW  0x31       ; Set up SPI port, Master mode, CLK/16,
    MOVWF  SSPCON     ;   Data xmit on falling edge (CKE=1 & CKP=1)
                      ;   Data sampled in middle (SMP=0 & Master mode)
    BSF    STATUS, RP0  ; Bank 1
    BSF    PIE, SSPIE   ; Enable SSP interrupt
    BCF    STATUS, RP0  ; Bank 0
    BSF    INTCON, GIE  ; Enable, enabled interrupts
    MOVLW  DataByte     ; Data to be Transmitted
                      ;   Could move data from RAM location
    MOVWF  SSPBUF     ; Start Transmission
```

15.5.1 SSP 模块 / 基本 SSP 模块的兼容性

当从基本 SSP 模块升级到 SSP 模块时， SSPSTAT 寄存器还另外包含两个控制位，这两个控制位仅在 SPI 模式下使用，它们是：

- SMP， SPI 输入数据的采样相位
- CKE， SPI 时钟沿选择位

为了使 SSP 模块与基本 SSP 模块的 SPI 模式相兼容，这些位必须合理设置。如果不按表 15-4 进行设置，可能会发生 SPI 通信错误。

表 15-4： 保持兼容的位状态设置

基本 SSP 模块	SSP 模块		
CKP	CKP	CKE	SMP
1	1	0	0
0	0	0	0

15.6 设计技巧

问 1： *在 SPI 模式下，为什么无法和 SPI 器件通信？*

答 1：

确保你使用了该器件的正确 SPI 模式。该 SPI 支持所有 4 种 SPI 模式，所以要确认所使用器件支持的 SPI 模式。检查时钟的极性和相位。

问 2： *使用 I^2C 模式时，为什么主控模式无法工作？*

答 2：

SSP 模块不支持硬件完全自动实现的主控模式，请参看应用笔记 AN578 介绍的如何用软件实现 SSP 模块的主控模式。如果你需要硬件完全自动实现的 I^2C 主控模式，请通过 Microchip 产品目录卡，查阅具有主控 SSP 模块的器件。

注：	在该手册印制时，只有某些高档器件（PIC17CXXX 和 PIC18F/CXXX）具有完全实现的 I^2C 主控模式。

问 3： *使用 I^2C 模式时，将数据写入 SSPBUF 寄存器，但数据并未发送，为什么？*

答 3：

确保 CKP 位置 1，以释放 I^2C 时钟。

15.7 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但其概念是相近的，通过适当修改并受到一定限制即可使用。目前和 SSP 模块相关的应用笔记有：

标题	应用笔记 #
Use of the SSP Module in the I ^{2}C™ Multi-Master Environment.	AN578
Using Microchip 93 Series Serial EEPROMs with Microcontroller SPI™ Ports	AN613
Software Implementation of I^{2}C™ Bus Master	AN554
Use of the SSP module in the Multi-master Environment	AN578
Interfacing PIC16C64/74 to Microchip SPI™ Serial EEPROM	AN647
Interfacing a Microchip PIC16C92x to Microchip SPI™ Serial EEPROM	AN668

15.8 版本历史

版本 A

这是描述 SSP 模块的初始发行版。

第 16 章 基本同步串行口 （BSSP）

目录

本章包括下面一些主要内容：

16.1 简介 16-2
16.2 控制寄存器 16-3
16.3 SPITM 模式 16-6
16.4 SSP 模块的 I^2CTM 操作 16-15
16.5 初始化 16-23
16.6 设计技巧 16-24
16.7 相关应用笔记 16-25
16.8 版本历史 16-26

注：	关于哪些型号的器件包含此模块，请参照附录 C.2 或器件数据手册。

SPI 是 Motorola 公司的商标。
I^2C 是 Philips 公司的商标。

16.1 简介

基本同步串行口模块 (BSSP) 是用于同其它外设模块或单片机进行通信的串行接口。这些外设模块可以是串行 EEPROM、移位寄存器、显示驱动器或 A/D 转换器等。BSSP 模块有以下两种工作模式：

- 串行外设接口 (SPI™)
- 内部互联 (I^2C™)
  - 从动模式
  - I/O 口输出斜率控制，启动位和停止位，便于用软件实现主控和多主机模式。

I^2C 是 Philips 公司的商标。

16.2 控制寄存器

寄存器 16-1： SSPSTAT: 同步串行口状态寄存器

U-0	U-0	R-0	R-0	R-0	R-0	R-0	R-0
—	—	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7:6 **未用**：读为 0

bit 5 **D/$\overline{A}$**：数据 / 地址位 (仅用于 I^2C 模式）

1 = 最后接收或发送的字节是数据
0 = 最后接收或发送的字节是地址

bit 4 **P**：停止位
(仅用于 I^2C 模式。当 SSP 模块被禁止时该位被清零)

1 = 检测到停止位 (复位时该位为 0)
0 = 未检测到停止位

bit 3 **S**：启动位
(仅用于 I^2C 模式。当 SSP 模块被禁止时该位被清零)

1 = 检测到启动位 (复位时该位为 0)
0 = 未检测到启动位

bit 2 **R/$\overline{W}$**：读 / 写位信息 (仅用于 I^2C 模式)

该位用来记录在最后一次地址匹配后接收到的读 / 写信息。从本机地址与接收地址匹配开始，到下一个启动位、停止位或无应答位 （$\overline{ACK}$）时，该位有效。

1 = 读
0 = 写

bit 1 **UA**：地址更新 (仅用于 10 位 I^2C 模式)

1 = 需要更新 SSPADD 寄存器中的地址
0 = 地址不需要更新

bit 0 **BF**：缓冲区满状态位

接收时 (SPI 和 I^2C 模式)

1 = 接收完成，SSPBUF 满
0 = 接收未完成，SSPBUF 空

发送时 (I^2C 模式时)

1 = 发送正在进行， SSPBUF 满
0 = 发送已经完成， SSPBUF 空

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

寄存器 16-2： SSPCON: 同步串行口控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位

1 = 正在发送前一个数据时，又有数据写入 SSPB 寄存器
(该位必须用软件清零)
0 = 表示未发生冲突

bit 6 **SSPOV**：接收溢出标志位

在 SPI 模式下：

1 = SSPBUF 寄存器中仍保持前一个数据时又收到新的数据。在溢出时，SSPSR 中的数据会丢失。溢出只会发生在从动模式下。即使只是发送数据，用户也必须读 SSPBUF，以避免产生溢出。在主控模式下，溢出位不会被置 “1”，因为每次接收或发送新数据，都要通过写 SSPBUF 来启动。

0 = 没有溢出

在 I^2C 模式下：

1 = 表示 SSPBUF 中仍保持前一个数据时又收到新的数据。

在发送方式下 SSPOV 位无效，在 SPI、 I^2C 模式下，该位都必须用软件清零。

0 = 没有溢出

bit 5 **SSPEN**：同步串行口的使能位

在 SPI 和 I^2C 两种模式下，当该位为 1 而使能时，应正确配置相应引脚的输入输出方向。

在 SPI 模式下：

1 = 使能串行口，并定义 SCK、 SDO、 SDI 和 SS 引脚为串行口引脚
0 = 禁止串行口，并定义 SCK、 SDO、 SDI 和 SS 引脚为一般 I/O 端口引脚

在 I^2C 模式下：

1 = 使能串行口，并定义 SDA 和 SCL 引脚为串行口引脚
0 = 禁止串行口，并定义 SDA 和 SCL 引脚为一般 I/O 端口引脚

bit 4 **CKP**：时钟极性选择：

在 SPI 模式下：

1 = 空闲状态时，时钟为高电平
0 = 空闲状态时，时钟为低电平

在 I^2C 模式下：

SCK 释放控制
1 = 使能时钟
0 = 保持时钟线为低电平 (用于保证数据的建立时间)

寄存器 16-2： SSPCON: 同步串行口控制寄存器 （续）

bit 3:0 **SSPM3:SSPM0**：同步串行口的工作模式选择位

0000 = SPI 主控模式，时钟 = FOSC/4
0001 = SPI 主控模式，时钟 = FOSC/16
0010 = SPI 主控模式，时钟 = FOSC/64
0011 = SPI 主控模式，时钟 = TMR2 输出 /2
0100 = SPI 从动模式，时钟 = SCK 引脚。使能 $\overline{SS}$ 引脚控制。
0101 = SPI 从动模式，时钟 = SCK 引脚。禁止 SS 引脚控制。$\overline{SS}$ 可用作 I/O 引脚。
0110 = I^2C 7 位地址的从动模式
0111 = I^2C 10 位地址的从动模式
1000 = 保留
1001 = 保留
1010 = 保留
1011 = I^2C 固件控制的主控模式 (从动模式空闲)
1100 = 保留
1101 = 保留
1110 = I^2C 固件控制的多主机模式
7 位地址，允许启动位和停止位中断功能
1111 = I^2C 固件控制的主控模式
10 位地址，允许启动位和停止位中断功能

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

16.3 SPI™ 模式

SPI 模式允许同步发送或接收 8 位数据，一般用以下三个引脚来完成通信：

- 串行数据输出 (SDO)
- 串行数据输入 (SDI)
- 串行时钟 (SCK)

当在从动模式下工作时，可能还需要第 4 个引脚：

- 从动选择 ($\overline{SS}$)

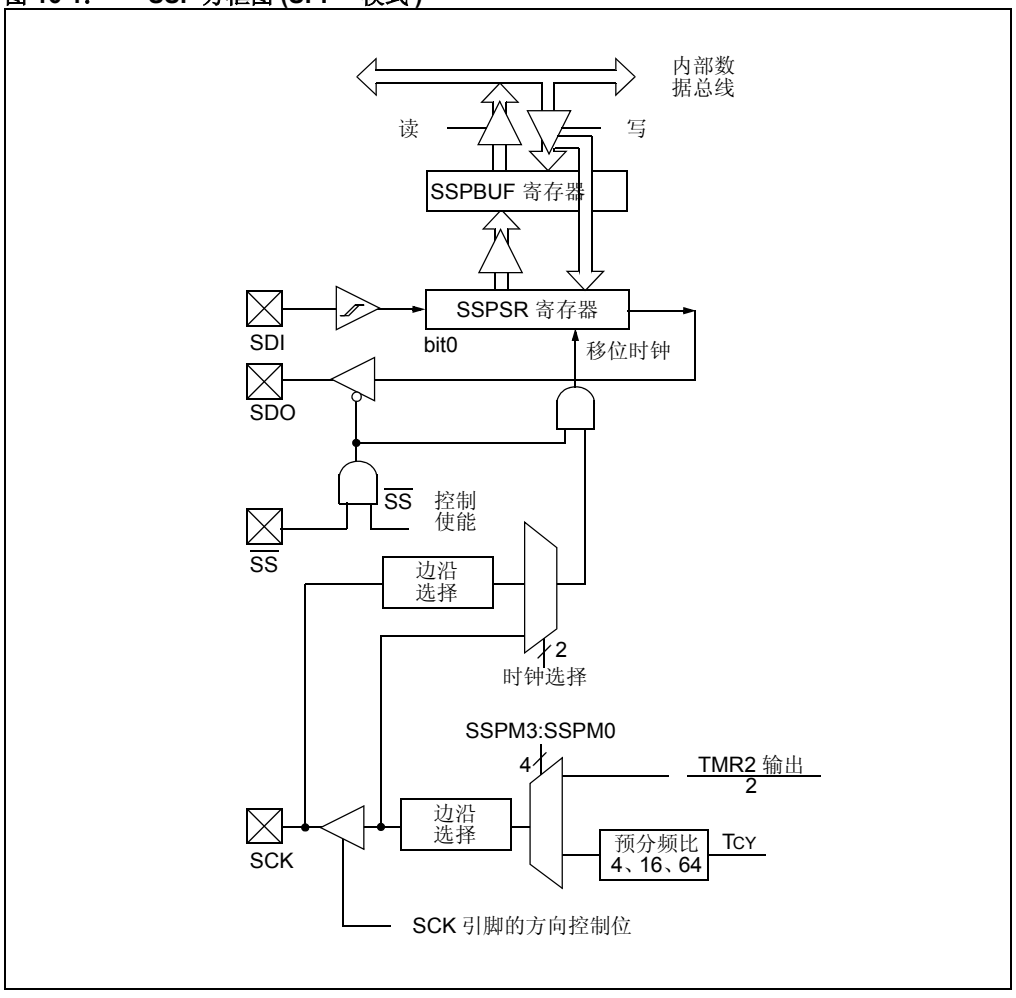
16.3.1 操作

初始化SPI时，必须通过设置SSPCON寄存器中的相应控制位 (SSPCON<5:0>)来指定以下各项：

- 主控模式 (SCK 作为时钟输出)
- 从动模式 (SCK 作为时钟输入)
- 时钟极性 (选择在 SCK 的上升沿还是下降沿输出 / 输入数据)
- 时钟速率 (仅用于主控模式)
- 从动选择 (仅用于从动模式)

图 16-1 给出了工作在 SPI 模式下 SSP 模块的方框图。

图 16-1： SSP 方框图 (SPI™ 模式)



SPI 是 Motorola 公司的商标。

SSP 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 用于器件输入和输出数据的移位，最高有效位在前。在新的数据接收完毕前，SSPBUF 保存写入 SSPSR 的数据。一旦 8 位新数据接收完毕，该数据被送入 SSPBUF 寄存器。同时缓冲区满标志位 BF (SSPSTAT <0>) 和中断标志位 SSPIF 置 1。这种双重缓冲接收方式，允许接收的数据被 CPU 读取之前，开始接收下一个数据。在数据发送 / 接收期间，任何试图写 SSPBUF 寄存器的操作都无效，会将冲突检测位 WCOL (SSPCON<7>) 置 1。此时用户必须用软件将 WCOL 位清零，否则无法判别下一次对 SSPBUF 的写操作是否成功。当应用软件要接收一个有效数据时，应该在下一个要传送的数据写入 SSPBUF 之前，将 SSPBUF 中的前一个数据读出。缓冲器满标志位 BF (SSPSTAT<0>) 用于表示何时把接收到的数据送入 SSPBUF 寄存器 （传输完成）。当 SSPBUF 中的数据被读出后， BF 位即被清零。如果 SPI 仅仅作为一个发送器，则不必理会接收的数据。通常可用 SSP 中断来判断发送或接收是否完成。 如果数据有效，可读出 SSPBUF 中的数据并 / 或对 SSPBUF (SSPSR) 进行写操作。如果不使用中断来处理数据的收发，用软件查询方法同样可确保不会发生写冲突。例 16-1 举例说明了在数据发送时如何写 SPBUF，阴影部分的指令仅在需要接收数据时才使用 (而某些 SPI 应用只发送数据）。

例 16-1： 对 SSPBUF (SSPSR) 寄存器的写操作

```
     BCF   STATUS, RP1    ;Specify Bank1
     BSF   STATUS, RP0    ;
LOOP BTFSS SSPSTAT, BF    ;Has data been received (transmit complete)?
     GOTO  LOOP           ;No
     BCF   STATUS, RP0    ;Specify Bank0
     MOVF  SSPBUF, W      ;W 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
```

SSPSR 寄存器不能直接读写，只能通过对 SSPBUF 寄存器寻址来访问。SSP 模块的状态寄存器 SSPSTAT 可用来指示各种状态条件。

16.3.2 使能 SPI™ 的 I/O 口

要使能 SSP 串行口，SSP 使能位 SSPEN (SSPCON<5>) 必须被置 1。要复位或重新配置 SPI 模式，先将 SSPEN 位清零，对 SSPCON 重新初始化，然后把 SSPEN 位置 1。这将设定 SDI、SDO、SCK 和 SS 引脚为 SSP 串行口引脚。要使这些引脚用作串行口功能，还必须通过 TRIS 寄存器正确设置这些引脚的方向，即：

- SDI 定义成输入
- SDO 定义成输出
- 主控模式时， SCK 定义成输出
- 从动模式时， SCK 定义成输入
- $\overline{SS}$ 定义成输入

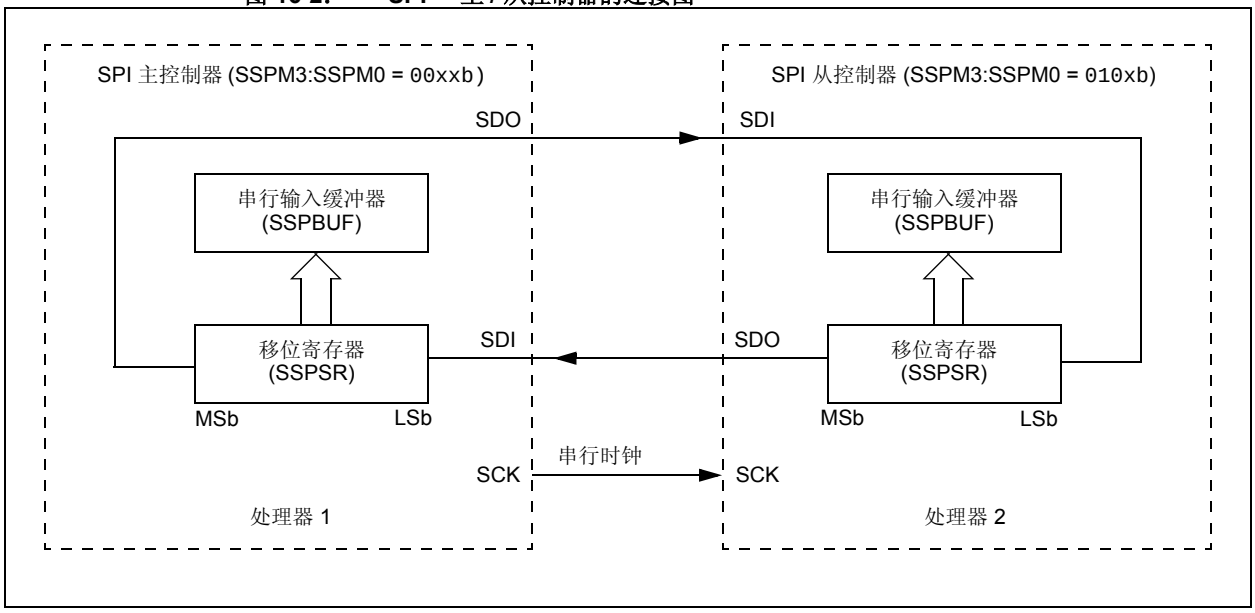
对于不需要的串行口引脚，可以通过把相应的数据方向寄存器 （TRIS）设置为上述的相反值而另作它用。例如在主控模式下，如果只发送数据（如发送到显示驱动器），那么通过将 SDI 和 SS 引脚的 TRIS 方向位清零，就可以把这两个引脚作为通用的输出口使用。

16.3.3 典型连接

图 16-2 给出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。根据程序设定的时钟边沿，分别位于两个处理器里的移位寄存器中的数据同时被移出，并在相反的时钟边沿被锁存。两个处理器的时钟极性 （CKP) 必须编程设定为相同，这样两个处理器就可以同时收发数据。至于数据是否有意义（或是无效“哑”数据）则取决于应用软件，这就导致以下三种数据发送方式：

- 主控制器发送数据 — 从控制器发送无效数据 （“哑”数据）。
- 主控制器发送数据 — 从控制器发送数据
- 主控制器发送无效数据 — 从控制器发送数据

图 16-2： SPI™ 主 / 从控制器的连接图



16.3.4 主控模式的操作

因为主控制器控制着 SCK 信号，所以它可以在任何时候启动数据传输，同时主控制器通过软件协议来决定从控制器 (处理器 2) 何时要传送数据。

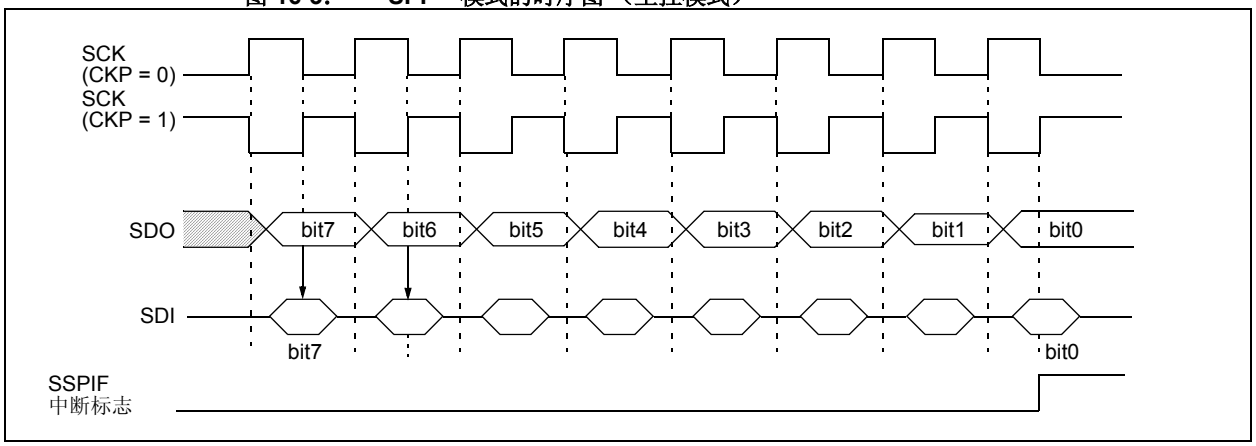
在主控模式下，数据一旦写入 SSPBUF 就开始发送或接收。如果 SPI 仅作为接收器，则可以禁止 SDO 输出 (将其设置为输入端口)。SSPSR 寄存器按设置的时钟速率，对 SDI 引脚上的信号进行连续地移位输入。每接收完一个字节，都把其送入 SSPBUF 寄存器，就象普通的接收字节一样（相应的中断和状态位置 1）。这在“线路主动监控”方式的接收器应用中可能是很有用的。

时钟极性可通过对 SSPCON 寄存器的 CKP 位 (SSPCON<4>) 编程来设定。图 16-3 是主控模式下 SPI 通信的时序图，最高位首先发送。在主控模式下， SPI 时钟速率 （位速率）可由用户编程设定为下面几种方式之一：

- FOSC/4 (或 TCY)
- FOSC/16 (或 4 • TCY)
- FOSC/64 (或 16 • TCY)
- 定时器 2 输出速率 /2

最大数据通信速率是 5 Mbps （当晶振为 20 MHz 时）。

图 16-3： SPI™ 模式的时序图 （主控模式）



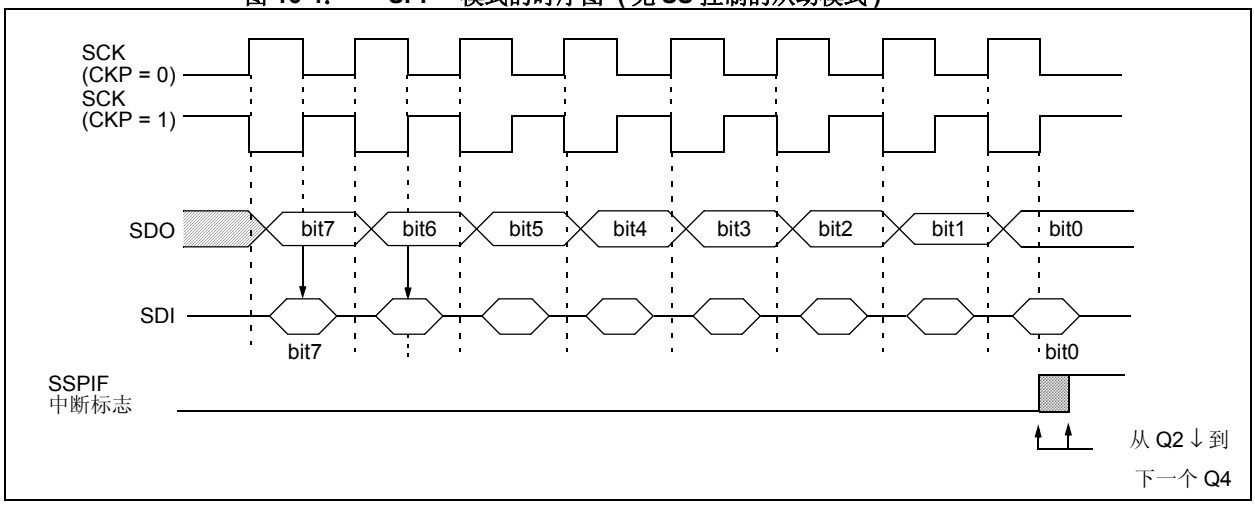
16.3.5 从动模式的操作

在从动模式下，当 SCK 引脚上出现外部时钟脉冲时，发送 / 接收数据。当最后一位数据锁存后，中断标志位 SSPIF 置 “1”。

时钟极性通过对 SSPCON 寄存器的 CKP 位 (SSPCON<4>) 编程来设定。图 16-4 是 SPI 通信的时序图，最高位先被发送。在从动模式下，外部时钟必须满足最短高电平和低电平的脉宽要求。

在休眠状态下，从控制器仍可发送和接收数据，如果允许中断，还可唤醒单片机。

图 16-4： SPI™ 模式的时序图 (无 $\overline{SS}$ 控制的从动模式)



16.3.6 从动选择模式

$\overline{SS}$ 引脚还可用于同步从动模式。要将 SPI 设置成同步从动模式，SPI 必须工作在从动模式（SSPCON<3:0> = 04h），并将 $\overline{SS}$ 引脚的 TRIS 位置位。当 $\overline{SS}$ 引脚为低电平时，允许数据的发送和接收，同时 SDO 引脚被驱动为高电平或低电平。当 $\overline{SS}$ 引脚变为高电平时，即使是在数据的发送过程中，SDO 引脚也不再被驱动，而是变成高阻悬浮状态。当 $\overline{SS}$ 引脚再次变成低电平时，如果此前没有重新设置 SPI 的工作模式，则数据将接着上次变为高电平时的中断点继续发送。为清零位计数器，必须先禁止然后再重新使能基本 SSP 模块。根据应用的需要，可在 SDO 引脚上外接上拉或下拉电阻。

将 SDO 引脚和 SDI 引脚相连，可以仿真二线制通信。当 SPI 作为接收器时，可将 SDO 引脚定义为输入，这样就禁止 SDO 引脚发送数据。而 SDI 总是定义为输入，因为它不会引起总线冲突。

图 16-5： SPI™ 模式的时序图 (带 $\overline{SS}$ 控制的从动选择模式)

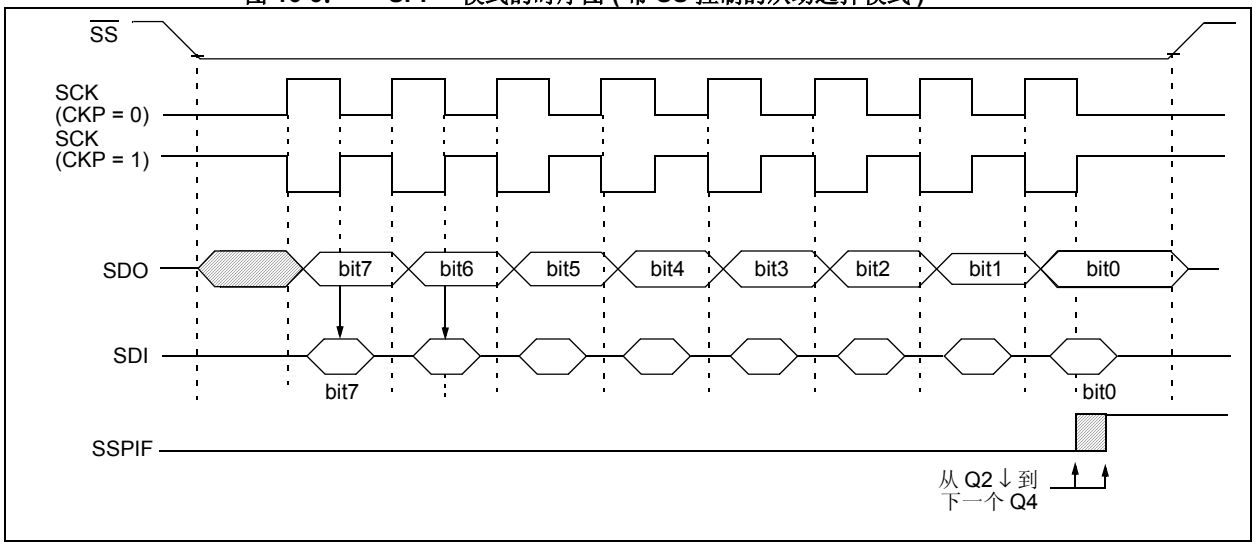
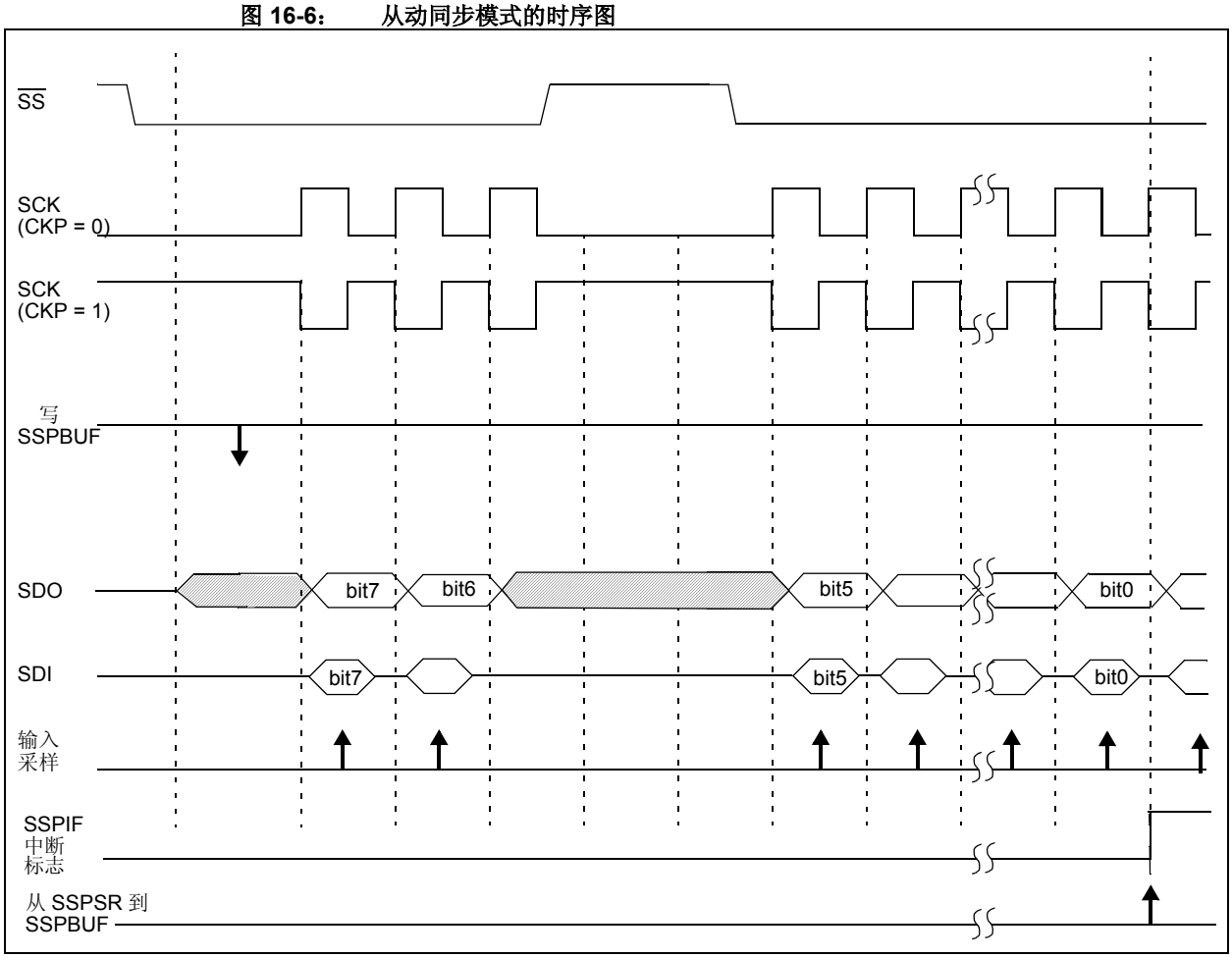


图 16-6： 从动同步模式的时序图



16.3.7 休眠状态下的操作

在主控模式下，此时所有模块的时钟都停止了，在器件被唤醒前，发送 / 接收也处于停滞状态。在器件恢复正常工作状态后，模块将继续数据的发送 / 接收。

在从动模式下，SPI 发送 / 接收移位寄存器与器件异步工作，所以在休眠状态时，数据仍可被移入 SPI 发送 / 接收移位寄存器。当接收完 8 位数据后，SSP 中断标志位将置 1，如果此时该中断是使能的，将唤醒器件。

16.3.8 复位的影响

复位会禁止 SSP 模块并停止当前的数据传输。

表 16-1： 和 SPI™ 操作有关的寄存器

寄存器名	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位、欠压复位时的值	其它复位时的值
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF (1)								0	0
PIE	SSPIE (1)								0	0
SSPBUF	同步串行口的接收缓冲器 / 发送寄存器								xxxx xxxx	uuuu uuuu
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	—	—	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	--00 0000	--00 0000

其中： x = 未知， u = 不变， - = 未用，读为 0。
阴影部分在 SPI 模式下的 SSP 中未用。

注 1：该位的位置和器件的具体型号有关。
2：这些位也可称为 GPIE 和 GPIF。

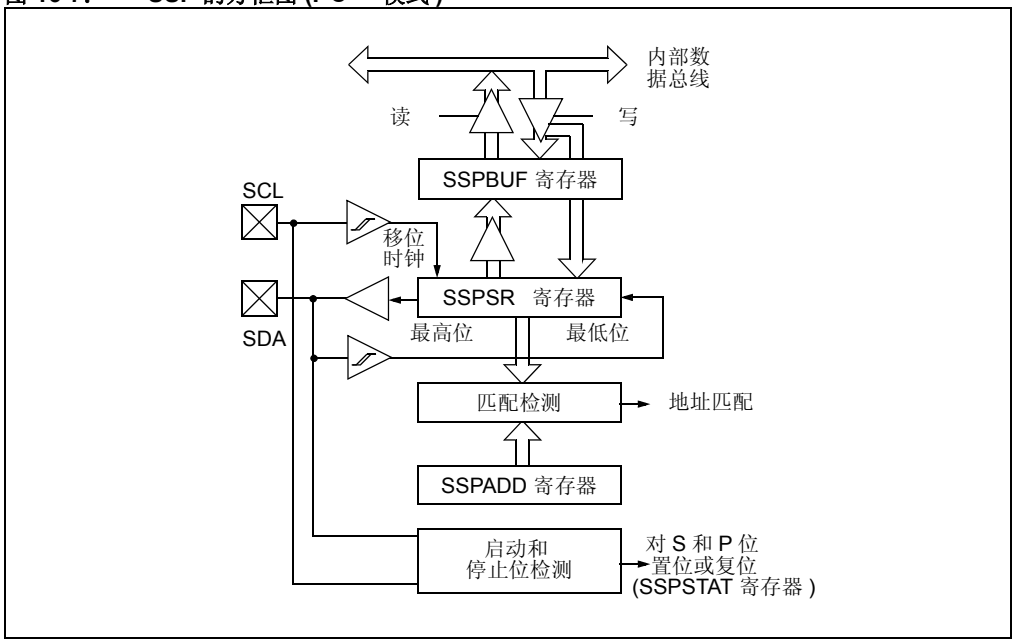
16.4 SSP 模块的 I²C™ 操作

SSP 模块工作在 I²C 模式时，除了不支持全局呼叫外，可以完成所有的从动模式功能，并且硬件上提供启动位和停止位中断，以便软件实现主控功能。SSP 模块实现了标准模式规范，并支持对 7 位和 10 位地址的寻址。附录 A 给出了 I²C 总线规范的概述。

数据传输使用两个引脚：一个是作为时钟线的 SCL 引脚，另一个是作为数据线的 SDA 引脚。用户应通过 TRIC 寄存器把这些引脚设置成输入。将 SSP 使能位 SSPEN （SSPCON<5>）置位，可使能 SSP 模块功能。

当 SCL 和 SDA 引脚为输入时，引脚上有窄脉冲（毛刺）滤波器，该滤波器可以工作在 100 KHz 和 400 KHz 两种模式下。在 100 KHz 模式下， 当这些引脚作为输出时，可对引脚进行压摆率控制，此控制和器件频率无关。

图 16-7： SSP 的方框图 (I²C™ 模式)



SSP 模块有 5 个寄存器用于 I^2C 操作，它们是：

- SSP 控制寄存器 (SSPCON)
- SSP 状态寄存器 (SSPSTAT)
- 串行接收 / 发送缓冲器 (SSPBUF)
- SSP 移位寄存器 (SSPSR) - 不可直接访问
- SSP 地址寄存器 (SSPADD)

SSPCON 寄存器用于控制 I^2C 的工作模式。可通过设置四个模式选择位 (SSPCON<3:0>) 来选择以下几种 I^2C 模式：

- I^2C 从动模式 (7 位地址)
- I^2C 从动模式 (10 位地址)
- I^2C 固件控制的多主机模式， 7 位地址 （允许启动位和停止位中断）
- I^2C 固件控制的多主机模式，10 位地址 （允许启动位和停止位中断)
- I^2C 固件控制的主控模式，从动模式空闲

在选择 I^2C 工作模式前，通过置位相应的 TRIS 位，将 SCL 和 SDA 引脚定义为输入。然后选择 I^2C工作模式，通过将SSPEN 位置1，使能SCL和 SDA引脚分别作为 I^2C模式的时钟线和数据线。

SSPSTAT 寄存器提供数据传输的状态，其中包括启动位和停止位的检测、确定接收的字节是数据还是地址、下一个字节是否已完成 10 位地址的传送，以及是数据传输是读操作还是写操作。SSPSTAT 是一个只读寄存器。

SSPBUF 寄存器存放要读 / 写的传输数据， SSPSR 寄存器将数据移入或移出器件。接收时，SSPBUF 和 SSPSR 构成了一个双重缓冲接收器，允许在前一个数据读出前即可接收下一个数据。当接收完字节后，将其送入 SSPBUF 寄存器，同时置位中断请求标志位 SSPIF。如果在 SSPBUF 寄存器中的前一个数据被读走前，又接收到一个新数据，就会发生接收器溢出，SSPOV 位 (SSPCON<6>) 将置 1。

SSPADD 寄存器用于保存从机地址。在 10 位地址模式时，用户需要写入地址的高字节 (1111 0 A9 A8 0)。在高字节地址匹配后，再写入地址的低字节 (A7:A0)。

16.4.1 从动模式

在从动模式下， SCL 和 SDA 引脚必须设置为输入 (相应的 TRIS 位置 1)。在需要时 （如在从动发送器情况下）， SSP 模块会将输入状态改变为输出状态输出数据。

当地址匹配时或在地址匹配后传送的数据被接收时，硬件会自动产生一个应答 ($\overline{ACK}$) 脉冲，并把在 SSPSR 中接收到的数据装入 SSPBUF 缓冲区。

满足下列条件之一， SSP 模块不会产生应答脉冲 $\overline{ACK}$：

a) 在传送的数据被接收之前，缓冲区满标志位 BF (SSPSTAT<0>) 已被置为 1。

b) 在传送的数据被接收之前，溢出标志位 SSPOV (SSPCON<6>) 已被置为 1

此时，移位寄存器 SSPSR 的值不会装入缓冲区 SSPBUF 中，但是中断标志位 SSPIF 和 SSPOV 位仍会置 1。表 16-2 列出了在给定 BF 和 SSPOV 位状态时，数据的接收情况。阴影部分表示用户软件没有对溢出状态进行适当清零的情况。标志位 BF 是通过读取 SSPBUF 进行清零的，而 SSPOV 位必须通过软件来清零。

SCL 时钟输入的高电平和低电平必须满足最小脉宽的要求才能正常工作。关于 I^2C 规范所规定的高低电平脉宽以及对 SSP 模块的具体要求，请参见 **“电气规范”一章**参数 100 和 101。

16.4.1.1 寻址

一旦 SSP 模块被使能，它就等待 START（启动）信号出现。在启动信号出现后， 8 位数据被移入 SSPSR 寄存器。所有移入的位都在时钟线 SCL 的上升沿采样。在 SCL 时钟的第 8 个脉冲下降沿，SSPSR<7:1> 的值与地址寄存器 SSPADD 的值作比较，如果地址匹配， BF 和 SSPOV 位就被清零，并完成下列操作：

a) 在第 8 个 SCL 脉冲的下降沿把 SSPSR 寄存器的值装入 SSPBUF 寄存器。

b) 缓冲器满标志位 BF 在第 8 个 SCL 脉冲的下降沿被置 1。

c) 产生 $\overline{\text{ACK}}$ 脉冲。

d) 在第 9 个 SCL 脉冲的下降沿，SSP 中断标志位 SSPIF 置 1 (如果允许中断，则产生中断）。

在 10 位地址模式时， 从动器件需要接收两个地址字节。第一个地址字节的高 5 位指定这是否是一个 10 位地址。R/W 位 (SSPSTAT<2>) 必须指定为写操作，这样从动器件就会接收第二个地址字节。对于 10 位地址，高字节应该是 ‘1111 0 A9 A8 0’，其中 A9 和 A8 是 10 位地址的两个最高位。10 位地址的工作步骤如下，其中 7- 9 步是针对从动发送器而言的：

1. 接收地址的第一个 (高) 字节 (SSPIF、 BF 和 UA (SSPSTAT<1>) 位被置 1)。
2. 用地址的第二个字节（10 位地址的低 8 位）更新 SSPADD 寄存器 (对 UA 位清零同时释放 SCL 时钟线)。
3. 读 SSPBUF 寄存器 (BF 位被清零) 并清零中断标志位 SSPIF。
4. 接收地址第二个字节 (SSPIF、 BF 和 UA 被置 1)。
5. 用地址的高字节更新 SSPADD 寄存器，这将使 UA 位清零并释放 SCL 时钟线。
6. 读 SSPBUF 寄存器 (BF 位清零) 并清零中断标志位 SSPIF。
7. 接收再次出现的启动 （START）信号。
8. 接收地址的第一 个 (高) 字节 (SSPIF 和 BF 位被置 1)。
9. 读 SSPBUF 寄存器 (BF 位清零) 并清零中断标志位 SSPIF。

注：	10 位地址模式下，在重复启动 （RESTART）条件 （第 7 步）出现后，用户只需要匹配开始的 7 位地址，不需要更新 SSPADD 寄存器以匹配另一半地址。

表 16-2： 传输接收数据字节的情况

数据接收时的状态位				
BF	**SSPOV**	**SSPSR → SSPBUF**	**产生 $\overline{\text{ACK}}$ 脉冲**	**置位 SSPIF 位（如允许中断，则产生 SSP 中断）**
0	0	是	是	是
1	0	否	否	是
1	1	否	否	是
0	1	是	否	是

注：阴影部分表示用户软件没有正确清除溢出条件时的情况。

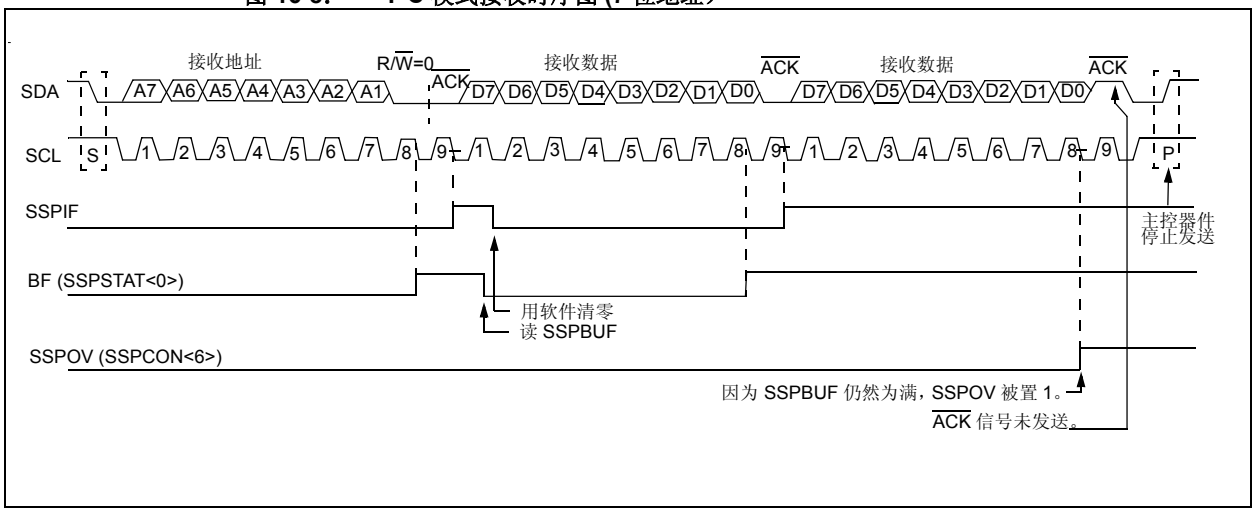
16.4.1.2 接收

当地址字节的 R/$\overline{W}$ 位是零且地址匹配时， SSPSTAT 寄存器中的 R/$\overline{W}$ 位被清零，同时把接收的地址装入缓冲器 SSPBUF。

当发生地址字节接收溢出时，则不会产生应答信号 ($\overline{ACK}$)。溢出条件是指 BF 位 (SSPSTAT<0>) 置 1 或 SSPOV 位 (SSPCON<6>) 置 1。

每个数据传输字节都会产生一个SSP 中断。SSPIF标志位必须用软件清零，状态寄存器SSPSTAT用于确定字节的状态。

图 16-8： I^2C 模式接收时序图 (7 位地址）



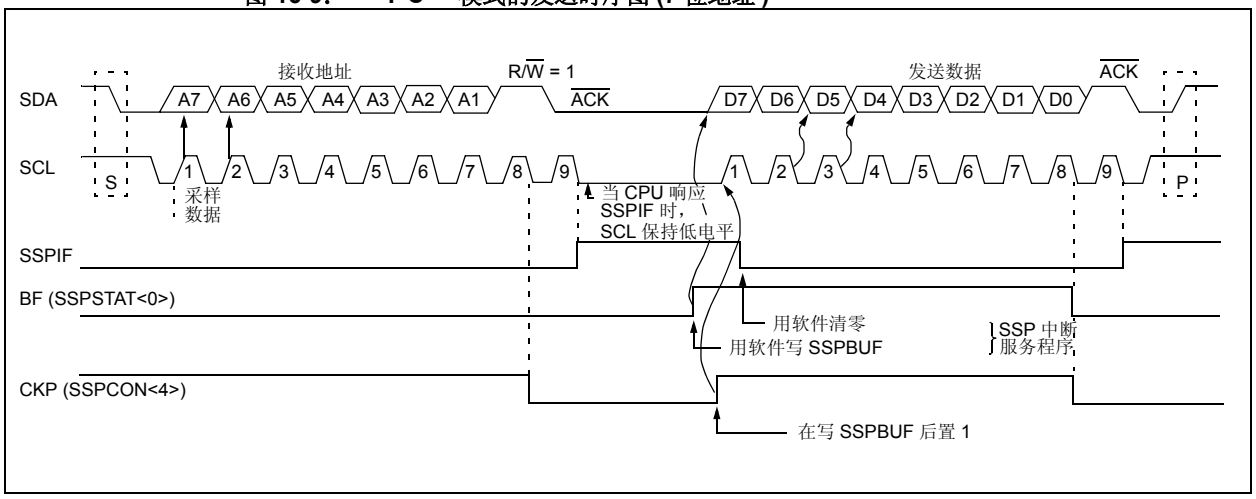
16.4.1.3 发送

当输入地址字节的 R/$\overline{W}$ 位置 1 且地址匹配时，状态寄存器 SSPSTAT 的 R/$\overline{W}$ 位被置 1。接收到的地址被装入缓冲器 SSPBUF。应答信号 $\overline{ACK}$ 在 SCL 的第 9 个脉冲时发送，同时 SCL 引脚保持低电平。发送的数据必须送入 SSPBUF 缓冲器，同时也送入 SSPSR 寄存器。然后通过把 CKP 位 （SSPCON<4>）置 1，使能 SCL 引脚。主控器件必须监视 SCL 引脚脉冲信号。从动器件可以通过延长 SCL 时钟的低电平，而使主控器件处于数据等待状态。8 位数据在 SCL 时钟的下降沿被移位输出。这可确保在 SCL 为高电平期间 SDA 信号是有效的 (如图 16-9)。

每个数据发送字节都会产生一个 SSP 中断， SSPIF 标志位必须通过软件清零，状态寄存器 SSPSTAT 用于确定字节传输的状态。SSPIF 标志位是在第 9 个脉冲下降沿被置 1。

作为从动发送器，在第 9 个 SCL 时钟脉冲的上升沿锁存从主控接收器发出的 $\overline{ACK}$ 信号。若 SDA 线为高电平（无 $\overline{ACK}$ 应答信号)，那么表示数据传输已完成。当无 $\overline{ACK}$ 应答被从动器件锁存时，从动逻辑被复位，并再监测下一个 START 信号的发生。如果 SDA 线是低电平 （有 $\overline{ACK}$ 应答信号），发送数据应装入 SSPBUF 缓冲器，并装入 SSPSR 寄存器，然后将 CKP 置 1，使能 SCL （即释放 SCL）。

图 16-9： I^2C^{TM} 模式的发送时序图 (7 位地址)



16.4.1.4 时钟仲裁

时钟仲裁是指在 SCL 引脚上抑制主控器件发送下一个时钟脉冲。当 CPU 需要响应 SSP 中断时 (SSPIF 位置 1，CKP 位清零)，处在 I^2C 从动模式的 SSP 模块将保持 SCL 引脚为低电平。从动器件需要将要发送的数据写入 SSPBUF 寄存器，然后将 CKP 位置 1，以允许主控器件发出所需的时钟信号。

16.4.2 主控模式 （固件）

主控模式是通过检测启动 （START） 和停止 （ STOP） 条件产生中断来工作的。STOP (P) 和 START (S) 位在复位或禁止 SSP 模块时被清零。当 P 位被置 1，或 P 位和 S 位都清零而总线是空闲时，可以获得对 I^2C 总线的控制权。

在主控模式下，SCL 和 SDA 线是通过改变相应的 TRIS 方向控制位来控制的。不管 PORT 寄存器中的值是什么，其输出电平总是为低电平。所以在发送数据时，发送 1 时必须把 TRIS 的相应位置 1 (设为输入)，发送 0 时把该位清零 （设为输出）。对于 SCL 线，可采用同样的方法对相应的 TRIS 位进行控制。

下列事件会引起中断标志位 SSPIF 置 1 (如果允许中断，便产生中断)：

- 启动条件 （START）
- 停止条件 （STOP）
- 数据字节的发送 / 接收

在从动模式空闲状态 (SSPM3:SSPM0 = 1011) 或从动模式有效状态 (SSPM3:SSP0 = 1110 或 1111)，都可以运行主控模式。当从动模式有效时，需要用软件来判断中断源。

16.4.3 多主控模式 （固件）

在多主机模式下，利用检测启动条件 （START） 和停止条件 （STOP） 产生中断的功能，可以判断总线何时空闲。在复位或禁止 SSP 模块时，停止位 (P) 和启动位 (S) 位清零。当停止位 P 位置 1 或 P 位和 S 位都为零而总线是空闲时，可以对 I^2C 总线进行控制操作。当总线处于忙状态且允许 SSP 中断时，一旦检测到停止条件便产生中断。

在多主机模式中，SDA 线必须一直被检测以判断信号电平是否是所期望的输出电平。如果期望高电平输出，但检测的是低电平，器件就需要释放 SDA 和 SCL 线 (把 TRIS 的相应位置 1)。有两种情况可能会丢失对总线的控制：

- 地址传输
- 数据传输

当允许从动模式时，从动器件将连续接收。如果在地址传输阶段失去总线控制机会，器件仍可被其它主机寻址，若被其它主机寻址，将产生应答信号 ACK 脉冲。如果在数据传输期间失去总线控制机会，则器件需要在以后重新传输数据。

16.4.4 休眠操作

在休眠模式下，I^2C 模块能够接收地址或数据。并且地址匹配或字节传输完成后，如果允许 SSP 中断，将唤醒处理器。

16.4.5 复位的影响

复位会禁止 SSP 模块并停止当前的传输。

表 16-3： 与 I^2C^{TM} 操作有关的寄存器

寄存器名	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位和欠压复位时的值	所有其它复位时的值
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000u
PIR	SSPIF (1)								0	0
PIE	SSPIE (1)								0	0
SSPBUF	同步串行口的接收缓冲器 / 发送寄存器								xxxx xxxx	uuuu uuuu
SSPADD	同步串行口 (I^2C 模式) 的地址寄存器								0000 0000	0000 0000
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	—	—	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	--00 0000	--00 0000

其中： x = 未知， u = 不变， - = 未用，读作 0。
阴影部分在 I^2C 模式下未用。

注 1：这些位的位置和器件的具体型号有关。
2：这些位也可称为 GPIE 和 GPIF。

16.5 初始化

例 16-2： SPI™ 主控模式的初始化

```
    CLRF    STATUS        ; Bank 0
    CLRF    SSPSTAT       ; Clear status bits
    MOVLW   0x31          ; Set up SPI port, Master mode, CLK/16,
    MOVWF   SSPCON        ;   Data xmit on rising edge
                          ;   Data sampled in middle
    BSF     STATUS, RP0   ; Bank 1
    BSF     PIE1, SSPIE   ; Enable SSP interrupt
    BCF     STATUS, RP0   ; Bank 0
    BSF     INTCON, GIE   ; Enable, enabled interrupts
    MOVLW   DataByte      ; Data to be Transmitted
                          ;   Could move data from RAM location
    MOVWF   SSPBUF        ; Start Transmission
```

16.5.1 SSP 模块 / 基本 SSP 模块的兼容性

与基本 SSP 模块相比， SSP 模块的 SSPSTAT 寄存器还另外包含两个控制位，它们是：

- SMP， SPI 输入数据的采样相位
- CKE， SPI 时钟沿选择位

为了使SSP模块与基本 SSP模块的SPI通信模式相兼容，这些位必须合理设置。如果不按表 16-4 进行设置，可能会发生 SPI 通信错误。如果 SSP 模块的配置与表 16-4 中所列不同，则基本 SSP 模块不能用于实现 SPI 模式。可以通过软件实现该模式。

表 16-4： 保持兼容的位状态设置

基本 SSP 模块	SSP 模块		
CKP	CKP	CKE	SMP
1	1	0	0
0	0	0	0

16.6 设计技巧

问 1： *在 SPI 模式下，为什么无法和 SPI™ 器件通信？*

答 1：

确保你使用了该器件的正确 SPI 模式。该 SPI 支持 4 种 SPI 模式中的两种，所以要确保你使用的 SPI 器件与这两种模式之一兼容。检查时钟的极性和相位。

如果你所使用的器件与这两种模式都不兼容，请选择使用 Microchip 具有 SSP 模块的器件。

问 2： *使用 I^2C 模式时，为什么主控模式无法工作？*

答 2：

SSP 模块不支持硬件完全自动实现的主控模式，请参看应用笔记 AN578 介绍的如何用软件实现 SSP 模块的主控模式。如果你需要硬件完全自动实现的 I^2C 主控模式，请通过 Microchip 产品目录卡，查阅具有主控 SSP 模块的器件。

注：	在该手册印制时，只有某些高档器件 (PIC17CXXX 和 PIC18F/CXXX) 具有完全实现的 I^2C 主控模式。

问 3： *使用 I^2C 模式时，将数据写入 SSPBUF 寄存器，但数据并未发送，为什么？*

答 3:

确保 CKP 位置 1，以释放 I^2C 时钟。

16.7 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但其概念是相近的，通过适当修改并受到一定限制即可使用。目前与本章相关的应用笔记有：

标题	应用笔记 #
Use of the SSP Module in the I $^2C^{TM}$ Multi-Master Environment.	AN578
Using Microchip 93 Series Serial EEPROMs with Microcontroller SPI^{TM} Ports	AN613
Software Implementation of I$^2C^{TM}$ Bus Master	AN554
Use of the SSP module in the Multi-Master Environment	AN578
Interfacing PIC16C64/74 to Microchip SPI^{TM} Serial EEPROM	AN647
Interfacing a Microchip PIC16C92x to Microchip SPI^{TM} Serial EEPROM	AN668

16.8 版本历史

版本 A

这是描述基本 SSP 模块的初始发行版。

第 17 章 主同步串行口 （MSSP）

目录

本章包括下面一些主要内容：

17.1 简介 17-2
17.2 控制寄存器 17-4
17.3 SPITM 模式 17-9
17.4 SSP 模块的 I²C™ 操作 17-18
17.5 I²CTM 总线的连接注意事项 17-56
17.6 初始化 17-57
17.7 设计技巧 17-58
17.8 相关应用笔记 17-59
17.9 版本历史 17-60

注：	目前中档系列单片机还不带有该模块。新的器件正在研制当中，但还没有相应的供货时间表。请到 Microchip 的网站或 BBS 上查看产品简介，以及新器件的详细情况和特点。 **Microchip 的高档单片机系列 (PIC17CXXX 和 PIC18F/CXXX) 带有该模块。请参考 Microchip 的网站、BBS, 或联系地区销售办事处或厂商代表。**

I²C 是 Philips 公司的商标。

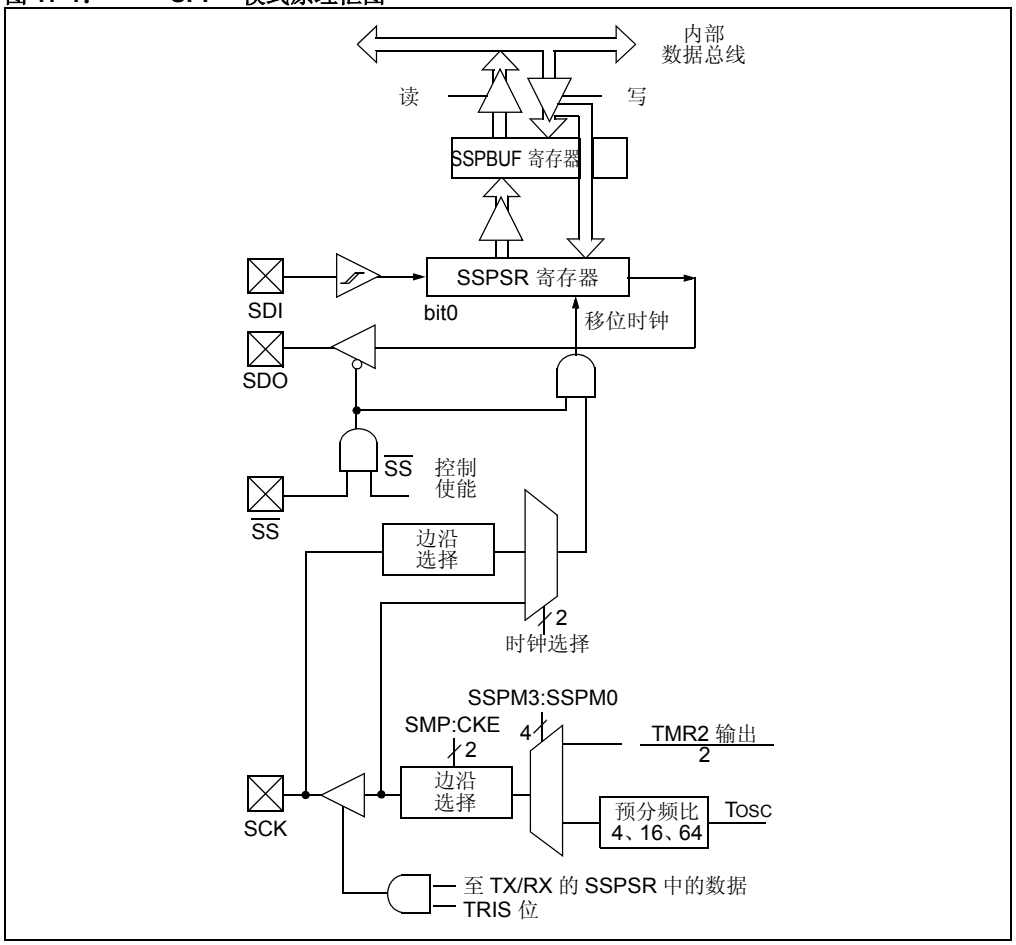
17.1 简介

主同步串行口 (MSSP) 模块是用于同其它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、显示驱动器或 A/D 转换器等等。 MSSP 模块有以下两种工作模式：

- 串行外设接口 (SPI™)
- 内部互联 (I^2C™)
  - 全主控模式
  - 从动模式 (全局地址呼叫)

图 17-1 给出了 SPI™ 模式的框图，图 17-2 和图 17-3 是两种不同 I^2C™ 模式的工作框图。

图 17-1： SPI™ 模式原理框图



SPI 是 Motorola 公司的商标。
I^2C 是 Philips 公司的商标。

图 17-2：　I²C™ 从动模式原理框图

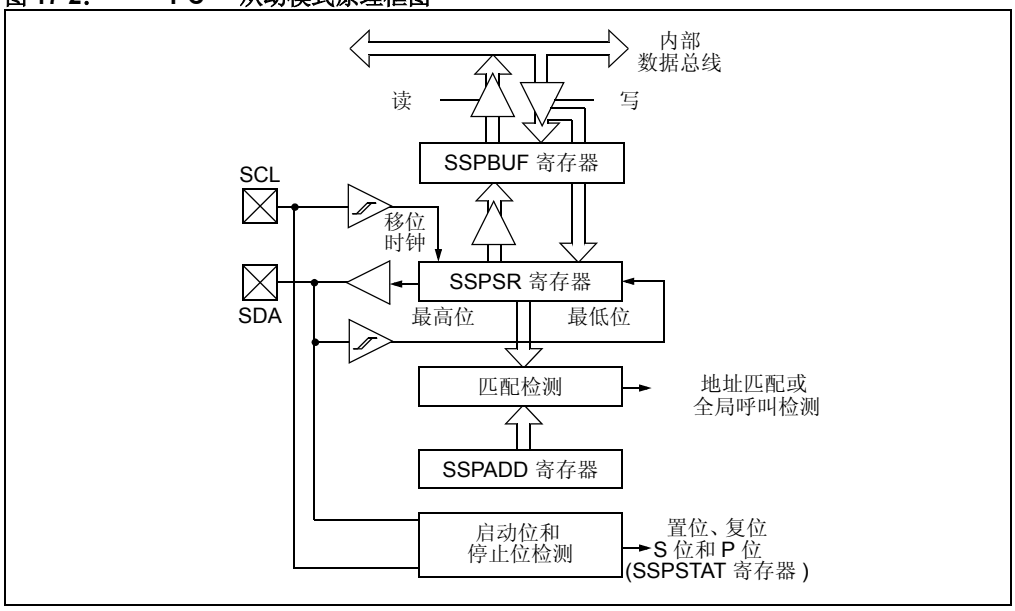
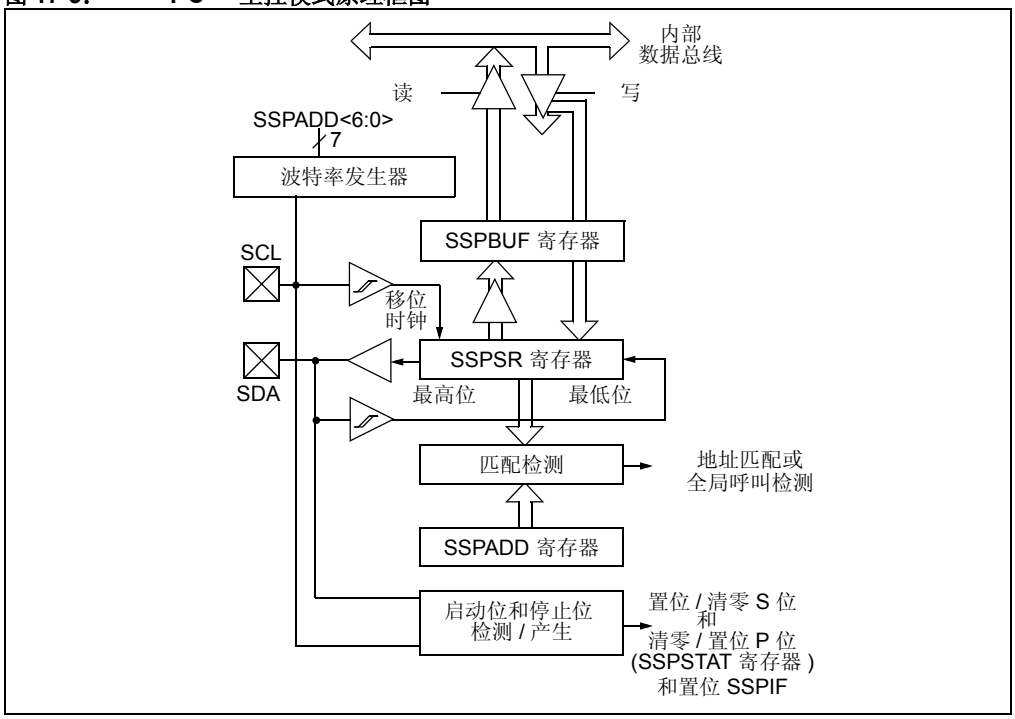


图 17-3：　I²C™ 主控模式原理框图



17.2 控制寄存器

寄存器 17-1： SSPSTAT: 同步串行口的状态寄存器

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

bit 7 **SMP:** 采样位

SPI 主控模式

1 = 在数据输出时间的末端采样输入数据
0 = 在数据输出时间的中间采样输入数据

SPI 从动模式

在该模式下， SMP 必须被清零。

I^2C 主控或从动模式：

1= 禁止标准速度模式 (100 kHz 和 1 MHz) 的压摆率控制
0= 使能高速模式 (400 kHz) 的压摆率控制

bit 6 **CKE**: SPI 时钟边沿选择

CKP = 0
1 = 在 SCK 上升沿传输数据
0 = 在 SCK 下降沿传输数据

CKP = 1
1 = 在 SCK 下降沿传输数据
0 = 在 SCK 上升沿传输数据

bit 5 **D/$\overline{A}$**: 数据 / 地址 位 (仅用于 I^2C 方式)

1 = 表示最后接收或发送的字节是数据
0 = 表示最后接收或发送的字节是地址

bit 4 **P**: 停止位
(仅用于 I^2C 方式。当 SSP 模块被禁止 （SSPEN 清零）时该位被清零。)

1 = 表示检测到停止位 (复位时该位为 '0')
0 = 表示未检测到停止位

bit 3 **S**: 启动位
(仅用于 I^2C 方式。当 SSP 模块被禁止 （SSPEN 清零）时该位被清零。)

1 = 表示检测到启动位 (复位时该位为 '0')
0 = 表示未检测到启动位

bit 2 **R/$\overline{W}$**: 读 / 写位信息 (仅用于 I^2C 模式)

该位用来记录在最后一次地址匹配时，接收到的读 / 写信息。仅在从本机地址与接收地址匹配开始，到下一个启动位、停止位或无 ACK 位，该位有效。

I^2C 从动模式：
1 = 读
0 = 写

I^2C 主控模式：
1 = 正在进行传送
0 = 不在进行传送
该位与 SEN、 RSEN、 PEN、 RCEN 或 ACKEN 相或的结果表示 SSP 是否处在空闲状态。

bit 1 **UA**：地址更新 (仅用于 10 位 I^2C 模式)

1 = 表示需要更新 SSPADD 寄存器中的地址
0 = 表示地址不需要更新

寄存器 17-1： SSPSTAT: 同步串行口的状态寄存器 （续）

bit 0 **BF：**缓冲区满状态位

接收时 (SPI 和 I^2C 模式)
1 = 表示接收完成，SSPBUF 满
0 = 表示接收未完成， SSPBUF 空

发送时 (仅 I^2C 模式)
1 = 数据正在发送 (不包括 $\overline{ACK}$ 和停止位)，SSPBUF 满
0 = 数据发送完成 (不包括 ACK 和停止位)，SSPBUF 空

图注：		
R = 可读位	W = 可写位	
U = 未用位，读为 ‘0’		- n = 上电复位时的值

寄存器 17-2： SSPCON1：SSP 控制寄存器 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL**：写冲突检测位

主控模式：

1 = 当 I^2C 不满足开始发送数据的条件时，有数据要写入 SSPBUF 寄存器
0 = 未发生冲突

从动模式：

1 = 正在发送前一个数据时，又有数据写入 SSPBUF 寄存器
(该位必须用软件清零)
0 = 未发生冲突

bit 6 **SSPOV**：接收溢出标志位

在 SPI 模式下：

1 = SSPBUF 中仍保存前一个数据时又收到新的数据。在溢出时， SSPSR 中的数据会丢失。溢出只会发生在从动模式下。在从动模式下，即使只是发送数据，用户也必须读 SSPBUF，以避免产生溢出。在主控模式下，溢出位不会被置 “1”，因为每次接收或发送新数据，都要通过写 SSPBUF 来启动。

0 = 没有溢出

在 I^2C 模式下：

1 = SSPBUF 中仍保存前一个数据时又收到新的数据。

在发送方式下， SSPOV 位无效，在 SPI 或 I^2C 模式下，该位都必须用软件清零。

0 = 没有溢出

bit 5 **SSPEN**：同步串行口的使能位

在 SPI 和 I^2C 两种模式下，当该位为 1 而使能时，应正确定义相应的引脚的输入输出方向。

在 SPI 模式下：

1 = 使能串行口，并定义 SCK、 SDO、 SDI 和 SS 为串行口引脚
0 = 禁止串行口，并定义 SCK、 SDO、 SDI 和 SS 引脚为一般 I/O 口引脚

在 I^2C 模式下：

1 = 使能串行口，并定义 SDA 和 SCL 为串行口引脚
0 = 禁止串行口，并定义 SDA 和 SCL 引脚为一般 I/O 口引脚

bit 4 **CKP**：时钟极性选择位

在 SPI 模式下：

1 = 空闲状态时，时钟为高电平
0 = 空闲状态时，时钟为低电平

在 I^2C 从动模式下：

SCK 释放控制

1 = 使能时钟
0 = 保持钟线为低电平 (用于保证数据的建立时间)

在 I^2C 主控模式下：

未使用

寄存器 17-2： SSPCON1：SSP 控制寄存器 1（续）

bit 3 - 0 **SSPM3:SSPM0**：同步串行口的工作模式选择位

0000 = SPI 主控模式，时钟 = FOSC/4
0001 = SPI 主控模式，时钟 = FOSC/16
0010 = SPI 主控模式，时钟 = FOSC/64
0011 = SPI 主控模式，时钟 = TMR2 输出 /2
0100 = SPI 从动模式，时钟 = SCK 引 脚，SS 引脚控制使能。
0101 = SPI 从动模式，时钟 = SCK 引脚。 SS 引脚控制禁止， SS 可用作一般 I/O 引脚。
0110 = I^2C 7 位地址的从动模式
0111 = I^2C 10 位地址的从动模式
1000 = I^2C 主控模式，时钟 = FOSC / (4 * (SSPADD+1))
1xx1 = 保留
1x1x = 保留

图注：		
R = 可读位	W = 可写位	
U = 未用位，读为 ‘0’		- n = 上电复位时的值

寄存器 17-3： SSPCON2: SSP 控制寄存器 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
GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
bit 7							bit 0

bit 7 **GCEN**: 全局呼叫使能位 (仅在 I^2C 从动模式下)
1 = SSPSR 接收到全局呼叫地址 (0000h) 时允许中断
0 = 禁止全局呼叫地址

bit 6 **ACKSTAT**：应答状态位 (仅在 I^2C 主控模式下)
在主控发送模式下：
1 = 未收到来自从动器件的应答
0 = 收到来自从动器件的应答

bit 5 **ACKDT**: 应答数据位 (仅在 I^2C 主控模式下)
在主控接收模式下：
用户在接收结束启动一个应答序列时，该值被发送出去。
1 = 不应答
0 = 应答

bit 4 **ACKEN**: 应答序列使能位 (仅在 I^2C 主控模式下)
在主控接收模式下：
1 = 在 SDA 和 SCL 引脚启动应答序列，发送 AKDT 数据位。
硬件自动清零
0 = 应答序列空闲

注： 如果 I^2C 模块不处在空闲模式，该位不能被置位(不支持并行操作)，不能对 SSPBUF 进行写操作 (或者禁止写 SSPBUF)。

bit 3 **RCEN**: 接收使能位 (仅在 I^2C 主控模式下)
1 = 使能 I^2C 接收模式
0 = 接收空闲

注： 如果 I^2C 模块不处在空闲模式，该位不能被置位(不支持并行操作)，不能对 SSPBUF 进行写操作 (或者禁止写 SSPBUF)。

bit 2 **PEN**：停止条件使能位 (仅在 I^2C 主控模式下)
SCK 释放控制
1 = 在 SDA 和 SCL 引脚启动停止条件，由硬件自动清零。
0 = 停止条件空闲

注： 如果 I^2C 模块不处在空闲模式，该位不能被置位(不支持并行操作)，不能对 SSPBUF 进行写操作 (或者禁止写 SSPBUF)。

bit 1 **RSEN**: 重复启动条件使能位 (仅在 I^2C 主控模式下)
1 = 在 SDA 和 SCL 引脚启动重复启动条件，由硬件自动清零。
0 = 重复启动条件空闲

注： 如果 I^2C 模块不处在空闲模式，该位不能被置位(不支持并行操作)，不能对 SSPBUF 进行写操作 (或者禁止写 SSPBUF)。

bit 0 **SEN**: 启动条件使能位 (仅在 I^2C 主控模式下)
1 = 在 SDA 和 SCL 引脚启动条件，由硬件自动清零。
0 = 启动条件空闲

注： 如果 I^2C 模块不处在空闲模式，该位不能被置位(不支持并行操作)，不能对 SSPBUF 进行写操作 (或者禁止写 SSPBUF)。

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

17.3 SPI™ 模式

SPI 模式允许同时、同步发送和接收 8 位数据。支持 SPI 的所有四种模式。一般用以下三个引脚来完成通信：

- 串行数据输出 (SDO)
- 串行数据输入 (SDI)
- 串行时钟 (SCK)

当工作在从动模式时，可能还需要第 4 个引脚：

- 从动选择 ($\overline{SS}$)

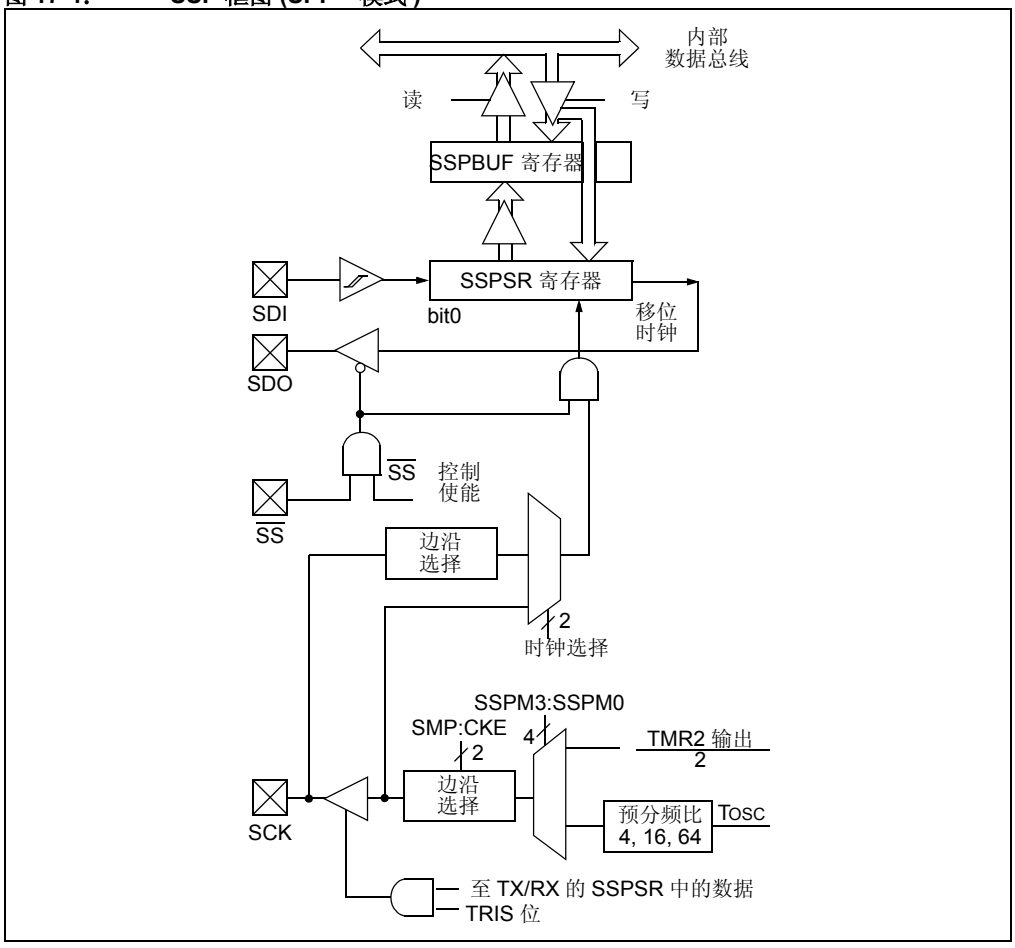
17.3.1 操作

初始化 SPI 时，必须通过编程来设置控制寄存器 SSPCON1 中的相应控制位 (SSPCON<5:0>) 和 SSPSTAT<7:6> 来确定以下工作模式：

- 主控模式 (SCK 作为时钟输出)
- 从动模式 (SCK 作为时钟输入)
- 时钟极性 (选择 SCK 的空闲状态)
- 数据输入采样相位 (数据输出时间的中间或末端)
- 时钟边沿 (在 SCK 的上升沿 / 下降沿输出数据)
- 时钟速率 (仅在主控模式下)
- 从动选择模式 (仅在从动模式下)

图 17-4 给出了 SPI 模式下 SSP 模块的框图。

图 17-4： SSP 框图 (SPI™ 模式)



SSP 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 用于器件输入和输出数据的移位，高位在前。在新的数据接收完毕前， SSPBUF 保存上次写入 SSPSR 的数据。一旦 8 位新数据接收完毕，该字节被送入 SSPBUF 寄存器。同时缓冲区满标志位 BF (SSPSTAT <0>) 和中断标志位 SSPIF 置 1。这种双重缓冲接收方式，允许接收数据被读走之前，开始接收下一个数据。在数据发送 / 接收期间，任何试图写 SSPBUF 寄存器的操作都无效，却会将写冲突检测位 WCOL (SSPCON<7>) 置 1。此时用户必须用软件将 WCOL 位清零，以判别下一次对 SSPBUF 的写操作是否成功完成。

为确保应用软件有效地接收数据，应该在新数据写入 SSPBUF 之前，将 SSPBUF 中的数据读走。缓冲区满标志位 BF (SSPSTAT<0>) 用于表示 SSPBUF 是否已经载入了接收的数据（发送完成）。当 SSPBUF 中的数据被读取后，BF 位即被清零。如果 SPI 仅仅作为一个发送器，则不必理会这一位。通常可用 SSP 中断来判断发送或接收是否完成。如果需要接收数据，可从 SSPBUF 中读取。SSP 中断一般用来确定发送 / 接收何时完成。必须对 SSPBUF 进行读和 / 或写。如果不打算使用中断方法，用软件查询方法同样可确保不会发生写冲突。例 17-1 给出了在发送数据时，如何写 SSPBUF (SSPSR) 的例子。

例 17-1： 对 SSPBUF (SSPSR) 寄存器的写操作

```
     BCF   STATUS, RP1    ;Specify Bank1
     BSF   STATUS, RP0    ;
LOOP BTFSS SSPSTAT, BF    ;Has data been received (transmit complete)?
     GOTO  LOOP           ;No
     BCF   STATUS, RP0    ;Specify Bank0
     MOVF  SSPBUF, W      ;W 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
```

SSPSR 寄存器不能直接读写，只能通过 SSPBUF 寄存器进行间接读写。另外，SSP 模块的状态寄存器 SSPSTAT 可用来表示各种状态条件。

17.3.2 使能 SPI™ 的 I/O 口

要使能 SSP 串行端口，SSP 使能位 SSPEN 位 (SSPCON<5>) 必须被置“1”。要复位或重新设置 SPI 模式，要先将 SSPEN 位清零，对 SSPCON 寄存器重新初始化，然后把 SSPEN 位置 1。这将设定 SDI、SDO、SCK 和 $\overline{SS}$ 引脚为 SSP 串行口引脚。要将这些引脚用作串行口功能，有些引脚还必须通过 TRIS 寄存器设置正确的方向，即：

- SDI 由 SPI 模块自动控制
- SDO 定义成输出
- 主控模式时， SCK 定义成输出
- 从动模式时， SCK 定义成输入
- $\overline{SS}$ 定义成输入

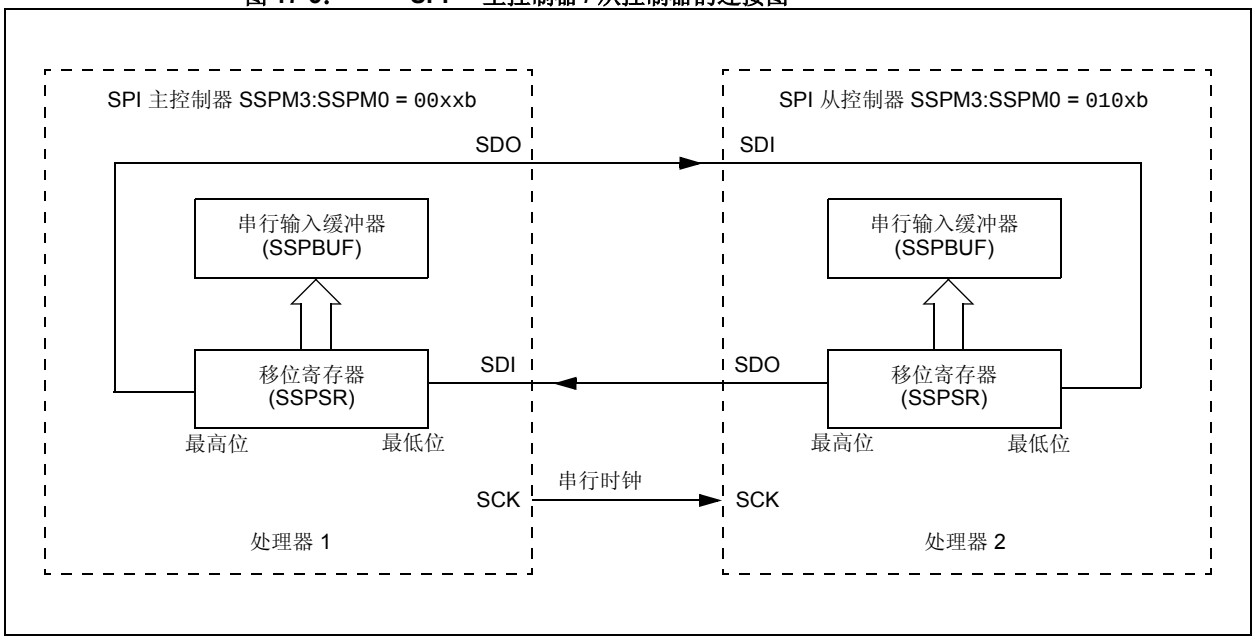
对于不需要的串行口功能，可以通过把相应的方向寄存器 （TRIS）设置为上述的相反值而另作它用。

17.3.3 典型连接

图 17-5 给出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 信号来启动数据传送。在两个移位寄存器之间，数据在编程规定的时钟边沿上传送，并在下一个相反的时钟边沿锁存。两个处理器的时钟极性 （CKP) 必须编程为相同，这样就可以同时收发数据。至于数据是否有意义 （或是无效 “哑”数据）则取决于应用软件，这就导致以下三种数据传送方式：

- 主控制器发送数据 — 从控制器发送无效数据
- 主控制器发送数据 — 从控制器发送数据
- 主控制器发送无效数据 — 从控制器发送数据

图 17-5： SPI™ 主控制器 / 从控制器的连接图



17.3.4 主控模式的操作

因为主控制器控制着 SCK 信号，所以它可以在任何时候启动数据传输，同时主控制器通过软件协议来决定从控制器 (图 17-5 中的处理器 2) 何时发送数据。

在主控模式下，数据一旦写入 SSPBUF 就开始发送或接收。如果 SPI 仅作为接收器，则 SDO 输出可以禁止 (将其设置为输入口)。SSPSR 寄存器按编程设置的时钟速率，对 SDI 引脚上的信号进行连续地移位输入，每接收完一个字节，都把其送入 SSPBUF，就象普通的接收字节一样（相应的中断和状态位置 1）。这在作为 “在线主动监控” 方式的接收器应用中是很有用的。

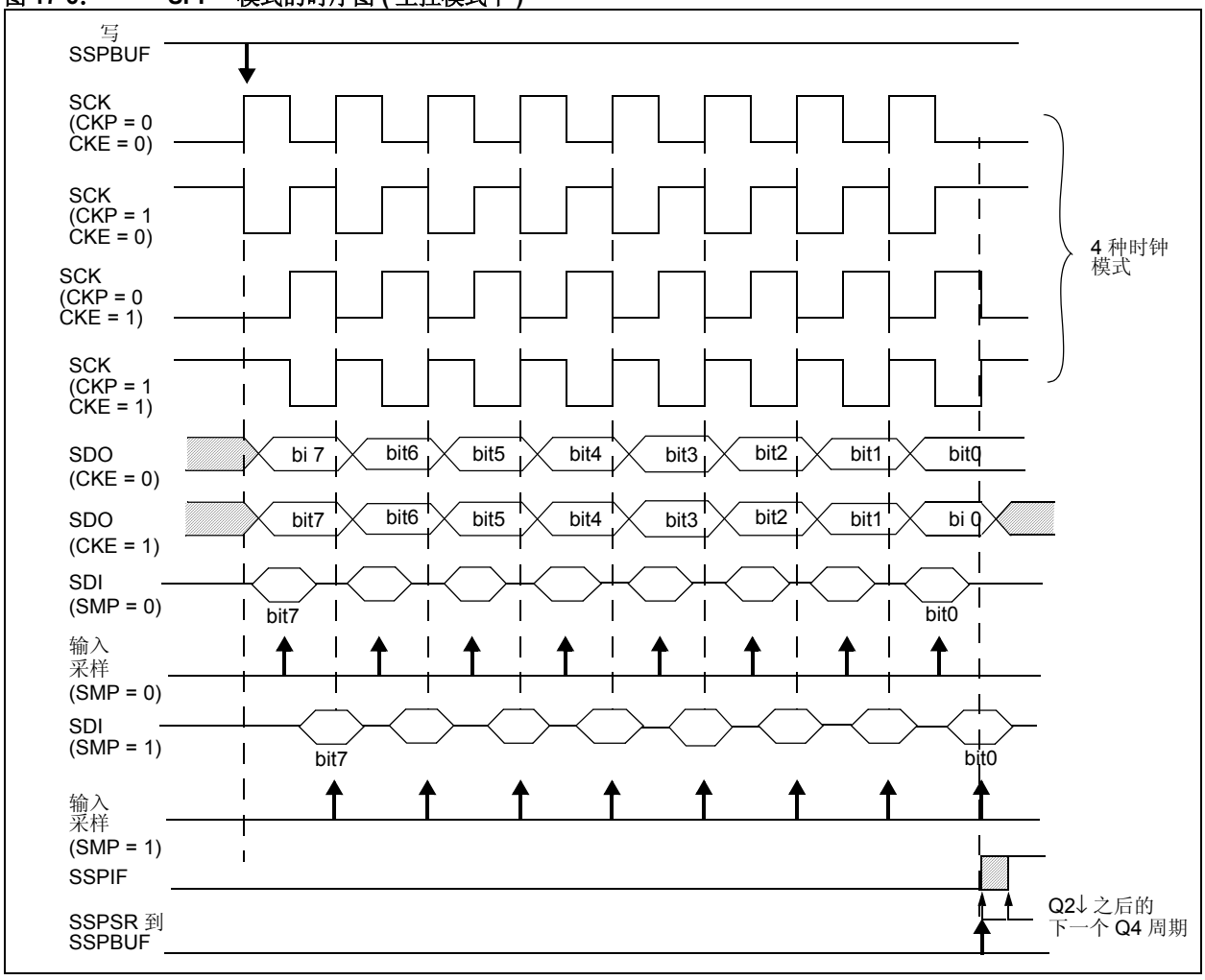
时钟极性可通过对 SSPCON 寄存器的 CKP 位 (SSPCON<4>) 进行适当的编程来设定。图 17-6、图 17-8 和 图 17-9 是 SPI 通信的时序图，最高位首先发送。在主控模式下， SPI 时钟速率（位速率）可由用户编程设定为下面几种方式之一 ：

- FOSC/4 (或 TCY)
- FOSC/16 (或 4 • TCY)
- FOSC/64 (或 16 • TCY)
- 定时器 2 输出 /2

这里允许的最大数据速率是 8.25 Mbps(当晶振为 20 MHz 时)。

图 17-6 给出了主控模式的时序图。当 CKE 位置 1 时，SDO 数据在 SCK 的第一个时钟边沿前就有效。图中所示输入采样的变化则由 SMP 位的状态决定。图中同时给出了何时将接收到的数据写入 SSPBUF。

图 17-6： SPI™ 模式的时序图 (主控模式下)



17.3.5 从动模式的操作

在从动模式下，当 SCK 引脚上的有外部时钟脉冲时发送 / 接收数据。当最后一位数据锁存后，中断标志位 SSPIF 置 1。

在从动模式下，外部时钟来自于 SCK 引脚上的外部时钟源。外部时钟必须满足电气规范中规定的最短高电平和最短低电平时间。

在休眠状态下，从动器件仍可发送 / 接收数据，当收到一个字节时将唤醒 CPU。

17.3.6 从动选择同步

$\overline{SS}$ 引脚用于同步从动模式。要将 SPI 设置成从动选择模式， SPI 必须工作在从动模式下，并使能 $\overline{SS}$ 引脚控制 （SSPCON<3:0> = 04h）。为使 $\overline{SS}$ 引脚作为输入端，不能将其驱动为低电平。数据锁存必须为高电平。当 $\overline{SS}$ 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 $\overline{SS}$ 引脚为高电平时，即使是在数据的发送过程中，SDO 引脚也不再被驱动，而是变成高阻悬浮状态。根据应用的需要，可在 SDO 引脚上外接上拉或下拉电阻。

注 1:	当 SPI 工作在从动模式下并且 $\overline{SS}$ 引脚控制使能 (SSPCON<3:0> = 0100) 时，如果 $\overline{SS}$ 引脚置为 V_{DD} 电平将使 SPI 复位。
注 2:	如果 SPI 工作在从动模式下并且 CKE 置 1，则必须使能 $\overline{SS}$ 引脚控制。

SPI 复位时，位计数器将被清零。这可由强制 $\overline{SS}$ 引脚为高电平或清零 SSPEN 位来实现。

将 SDO 引脚和 SDI 引脚相连，可以仿真二线制通信。当需要 SPI 作为接收时，可将 SDO 引脚定义为输入，这样 SDO 引脚就不会发送数据。而 SDI 一直作为输入线 （SDI 功能），因为它不会引起总线冲突。

图 17-7：　从动同步模式的时序图

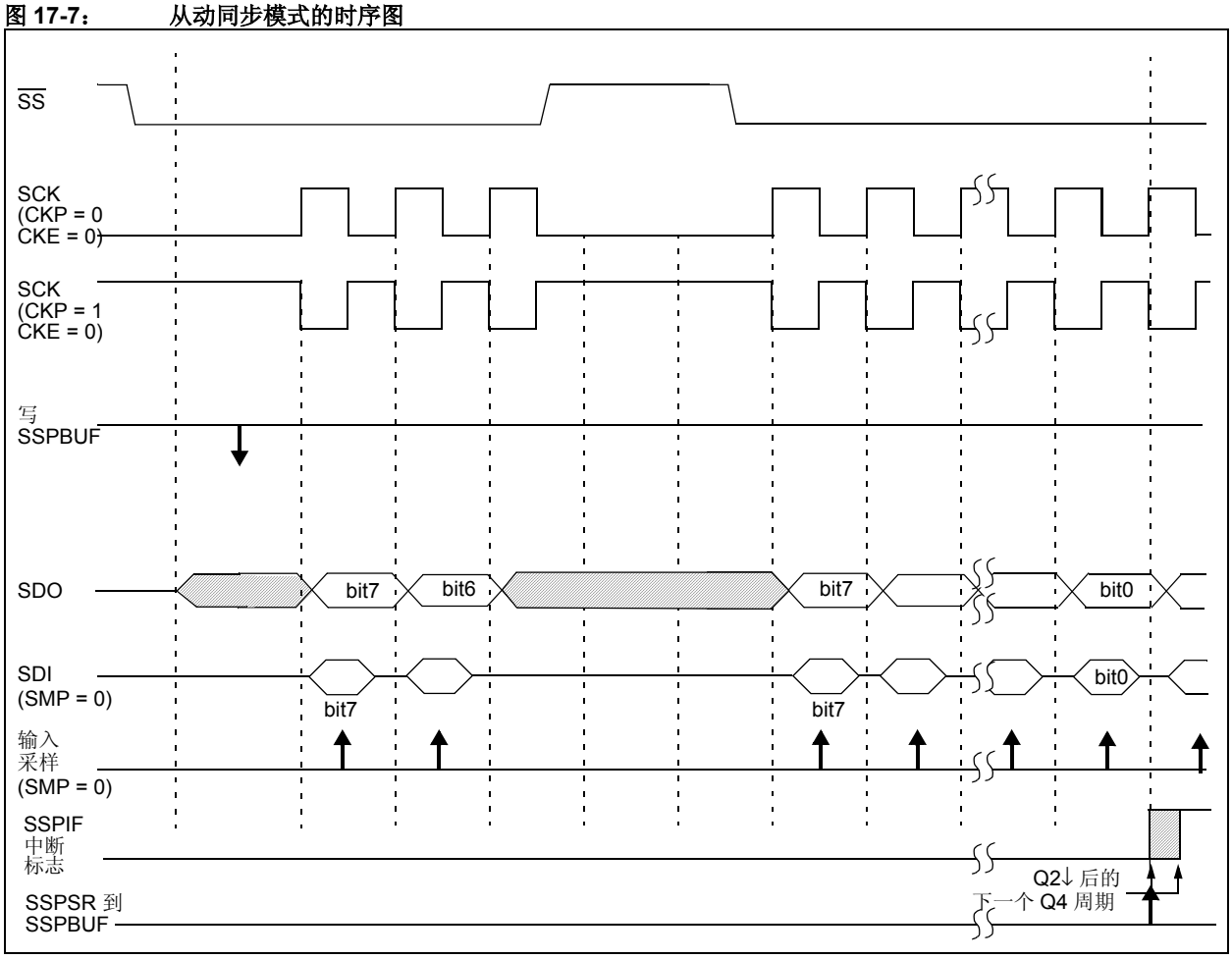


图 17-8：　SPI™ 模式的时序图 (从动模式， CKE = 0)

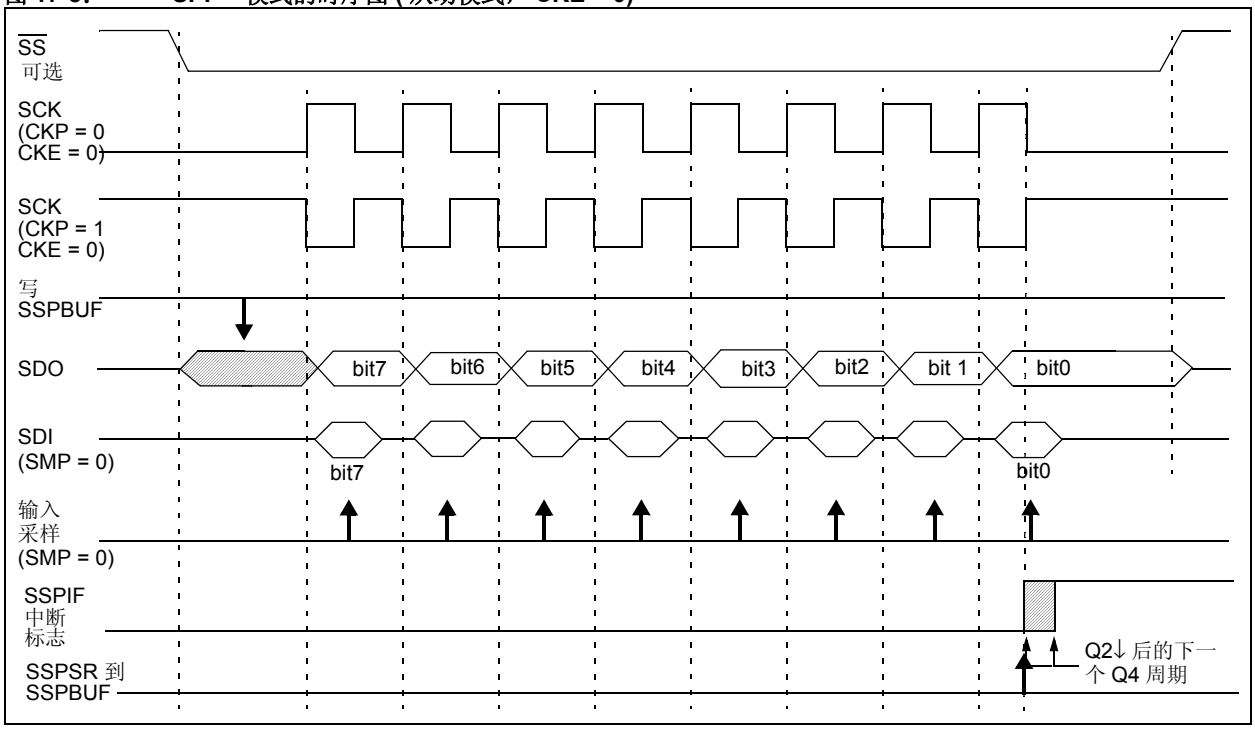
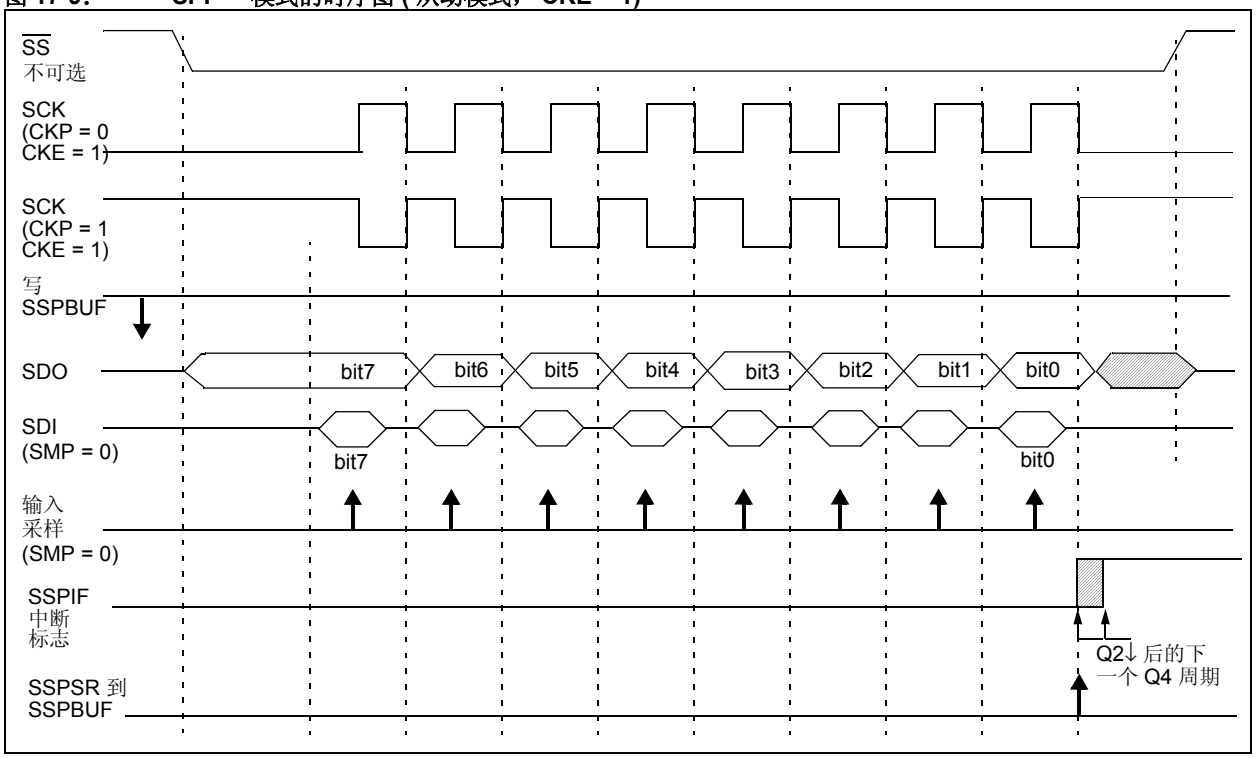


图 17-9： SPI ™ 模式的时序图 (从动模式， CKE = 1)



17.3.7 休眠状态下的操作

在主控模式下，此时所有模块的时钟都停止了，在器件被唤醒前，发送 / 接收也处于停滞状态。在器件恢复正常工作状态后，模块将继续数据的发送 / 接收。

在从动模式下，SPI™ 发送 / 接收移位寄存器与器件异步工作。所以在休眠状态，数据仍可被移入 SPI 发送 / 接收移位寄存器。当接收完 8 位数据后， MSSP 中断标志位将置 1，并且如果此时 MSSP 中断为允许状态，将唤醒器件。

17.3.8 复位的影响

复位操作会禁止 MSSP 模块并停止当前的数据发送。

表 17-1： 和 SPI 模块有关的寄存器

寄存器名	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位、欠压复位时的值	其它复位时的值
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 0000	0000 0000
PIR	SSPIF (1)								0	0
PIE	SSPIE (1)								0	0
SSPBUF	同步串行口的接收缓冲器 / 发送寄存器								xxxx xxxx	uuuu uuuu
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000

其中： x = 未知 , u = 不变， - = 未使用，读为 0。
阴影部分与 SPI 模式无关。

注 1：该位的位置和器件的具体型号有关。
2：这些位也称为 GPIE 和 GPIF。

17.4 SSP 模块的 I²C™ 操作

MSSP 模块工作在 I²C 模式时，可以完成所有的主控和从动功能 (包括全局呼叫支持)，并且硬件上提供启动位和停止位的中断来判断总线何时空闲 (多主机方式)。 SSP 模块实现标准模式规范，以及 7 位和 10 位寻址。附录 A 给出了 I²C 总线规范的概述。

当 SCL 和 SDA 引脚作为输入时，引脚上有窄脉冲滤波器，该滤波器可以工作在 100 KHz 和 400 KHz 两种模式下。在 100 KHz 模式下，当这些引脚作为输出时，引脚上附有与器件频率无关的压摆率控制特性。

图 17-10： I²C™ 从动模式框图

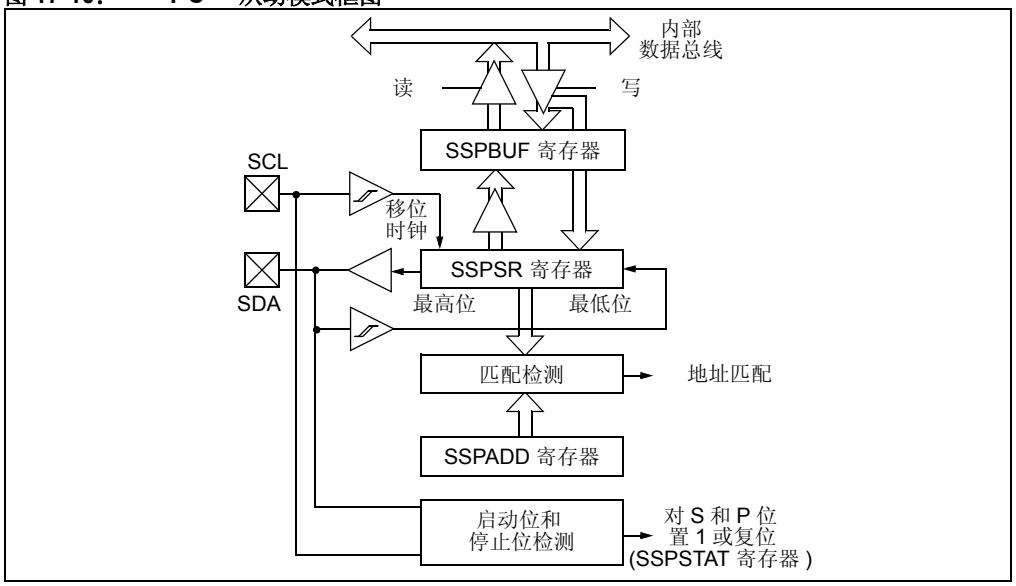
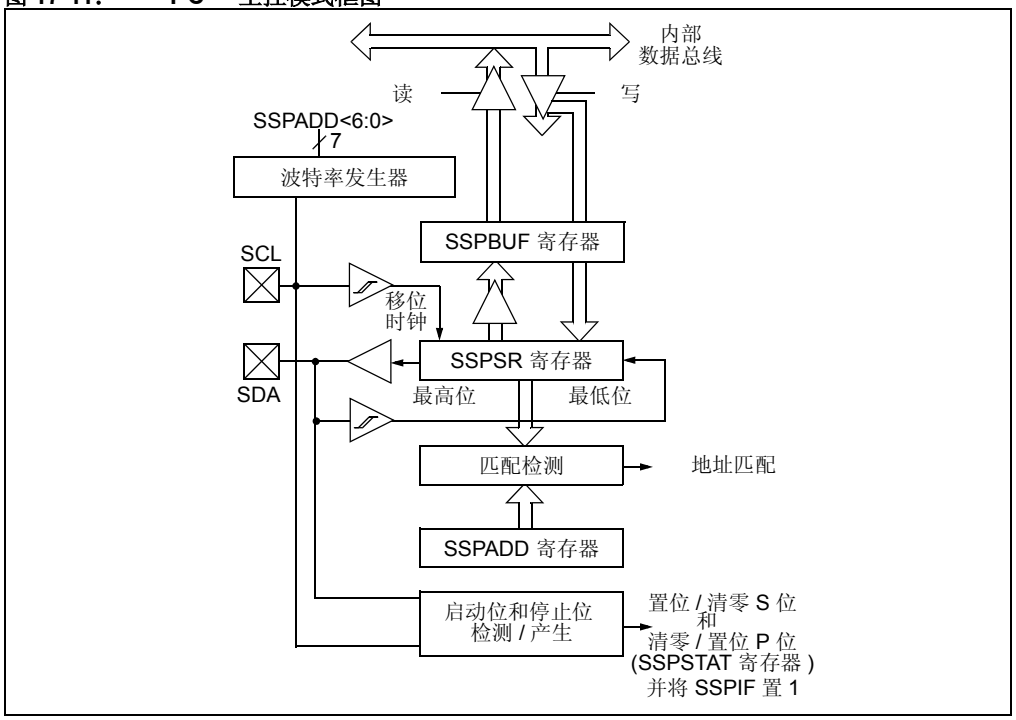


图 17-11： I²C™ 主控模式框图



有两个引脚用于数据传送。它们是时钟引脚 SCL 和数据引脚 SDA。当使能 I^2C 模式时，这两个引脚自动配置。通过置位 SSP 使能位 SSPEN (SSPCON1<5>) 可使能 SSP 模块功能。

SSP 模块有 6 个寄存器用于 I^2C 操作，它们是 ：

- SSP 控制寄存器 1 (SSPCON1)
- SSP 控制寄存器 2 (SSPCON2)
- SSP 状态寄存器 (SSPSTAT)
- 串行接收 / 发送缓冲器 (SSPBUF)
- SSP 移位寄存器 (SSPSR) — 不可直接访问
- SSP 地址寄存器 (SSPADD)

SSPCON1 寄存器用于控制 I^2C 的工作模式。可通过设置 (SSPCON<3:0>) 选择以下几种 I^2C 模式：

- I^2C 从动模式 (7 位地址)
- I^2C 从动模式 (10 位地址)
- I^2C 主控模式，时钟 = OSC/4 (SSPADD +1)

在选择 I^2C 工作模式前，必须通过置位相应的 TRIS 位，将 SCL 和 SDA 引脚定义为输入。然后选择 I^2C 工作模式，通过将 SSPEN 位置 1，使能 SCL 和 SDA 引脚分别用作 I^2C 模式的时钟线和数据线。

SSPSTAT 寄存器提供数据传输的状态，其中包括启动位和停止位的检测、确定接收的字节是数据还是地址、下一个字节是否已完成 10 位地址的传送，以及是数据传输是读操作还是写操作。

SSPBUF 寄存器存放要读 / 写的传输数据， SSPSR 寄存器将数据移入或移出器件。接收时，SSPBUF 和 SSPSR 构成了一个双重缓冲接收器，允许在前一个数据读出前即可接收下一个数据。当接收完字节后，将其送入 SSPBUF 寄存器，同时置位中断请求标志位 SSPIF。如果在 SSPBUF 寄存器中的前一个数据被读走前，又接收到一个新数据， 就会发生接收器溢出，SSPOV 位 (SSPCON<6>) 将置 1，且 SSPSR 中的数据将丢失。

SSPADD 寄存器用于保存从机地址。在 10 位地址模式时，用户需要写入地址的高字节 (1111 0 A9 A8 0)。在高字节地址匹配后，再装入地址的低字节 (A7:A0)。

17.4.1 从动模式

在从动模式下，SCL 和 SDA 引脚必须配置为输入。在需要时（如作为从动发送器），SSP 模块会强行将输入状态改变为输出状态输出数据。

当地址匹配时或在地址匹配后传送的数据被接收时，硬件会自动产生一个应答 ($\overline{ACK}$) 脉冲，并把在 SSPSR 中接收到的数据装入 SSPBUF 缓冲区。

满足下列条件之一， SSP 模块不会产生应答脉冲 $\overline{ACK}$：

a) 在传送的数据被接收之前，缓冲区满标志位 BF (SSPSTAT<0>) 已被置 1。

b) 在传送的数据被接收之前，溢出标志位 SSPOV (SSPCON<6>) 已被置 1。

如果 BF 位置 1，移位寄存器 SSPSR 的值不会装入缓冲器 SSPBUF 中，但是中断标志位 SSPIF 和 SSPOV 位仍会置 1。表 17-2 列出了在给定 BF 和 SSPOV 位状态时，数据的接收情况。阴影部分表示用户软件没有对溢出状态进行适当清零的情况。BF 标志位通过读 SSPBUF 寄存器可清零，SSPOV 溢出标志位则必须通过软件来清零。

SCL 时钟输入的高电平和低电平必须满足最小脉宽的要求才能正常工作。关于 I^2C 规范所规定的高低电平脉宽以及对 SSP 模块的具体要求，请参见 **“电气规范”一章**的参数 100 和 101。

17.4.1.1 寻址

一旦 SSP 模块被使能，它就等待 START（启动）信号出现。在启动信号出现后， 8 位数据被移入 SSPSR 寄存器。所有移入的位都在时钟线 SCL 的上升沿采样。在 SCL 时钟的第 8 个脉冲下降沿，SSPSR<7:1> 的值与地址寄存器 SSPADD 的值作比较，如果地址匹配， BF 和 SSPOV 位就被清零，并完成下列操作：

a) 在第 8 个 SCL 脉冲的下降沿，把 SSPSR 寄存器的值装入 SSPBUF 寄存器。
b) 缓冲器满标志位 BF 在第 8 个 SCL 脉冲的下降沿被置 1。
c) 产生 $\overline{\text{ACK}}$ 脉冲。
d) 在第 9 个 SCL 脉冲的下降沿，SSP 中断标志位 SSPIF 置 1 (如果允许中断，则产生中断）。

在 10 位地址模式时，从动器件需要接收两个地址字节。第一个地址字节的高 5 位指定这是否是一个 10 位地址。R/$\overline{\text{W}}$ 位 (SSPSTAT<2>) 必须指定为写操作，这样从动器件就会接收第二个地址字节。对于 10 位地址，第一个 （高）字节应该是 '1111 0 A9 A8 0'，其中 A9 和 A8 是 10 位地址的两个最高位。 10 位地址的工作步骤如下，其中 7- 9 步是针对从动发送器而言的：

1. 接收地址的第一个 (高) 字节 (SSPIF、 BF 和 UA (SSPSTAT<1>) 位被置 1)。
2. 用地址的第二个字节（10 位地址的低 8 位）更新 SSPADD 寄存器 (对 UA 位清零同时释放 SCL 时钟线)。
3. 读 SSPBUF 寄存器 (BF 位被清零) 并清零中断标志位 SSPIF。
4. 接收地址第二个字节 (SSPIF、 BF 和 UA 被置 1)。
5. 用地址的高字节更新 SSPADD 寄存器，这将使 UA 位清零并释放 SCL 时钟线。
6. 读 SSPBUF 寄存器 (BF 位清零) 并清零中断标志位 SSPIF。
7. 接收重复的启动信号。
8. 接收地址的第一 个 (高) 字节 (SSPIF 和 BF 位被置 1)。
9. 读 SSPBUF 寄存器 (BF 位清零) 并清零中断标志位 SSPIF。

注：	10 位地址模式下，在重复启动条件（第 7 步）出现后，用户只需要匹配开始的 7 位地址，不需要更新 SSPADD 寄存器以匹配另一半地址。

表 17-2： 传输接收数据字节的情况

数据接收时的状态位		SSPSR → SSPBUF	产生 $\overline{\text{ACK}}$ 脉冲	置位 SSPIF (如允许中断，则产生 SSP 中断)
BF	SSPOV			
0	0	是	是	是
1	0	否	否	是
1	1	否	否	是
0	1	是	否	是

注： 阴影部分表示用户软件没有正确清除溢出状态时的情况。

17.4.1.2 从动接收

当地址字节的 R/$\overline{W}$ 位是零且地址匹配时， SSPSTAT 寄存器中的 R/$\overline{W}$ 位被清零，同时把接收的地址装入 SSPBUF。

当发生地址字节接收溢出时，则不会产生应答信号 ($\overline{ACK}$)。溢出条件是指 BF 置 1 或 SSPOV 位 (SSPCON<6>) 置 1。

每个数据传输字节都会产生一个 SSP 中断，SSPIF 标志位必须用软件清零。 状态寄存器 SSPSTAT 用于确定接收数据字节的状态。

注：	如果 SSPOV 位被置 1 且 BF 标志位被清零，则会对 SSPBUF 进行写操作。如果对 SSPBUF 进行读操作，但用户在下一次接收前不对 SSPOV 位清零，则从动器件将不发送 $\overline{ACK}$，并且 SSPBUF 被更新。

17.4.1.3 从动发送

当输入地址字节的 R/$\overline{W}$ 位置 1 且地址匹配时，状态寄存器 SSPSTAT 的 R/$\overline{W}$ 位被置 1。接收到的地址被装入缓冲器 SSPBUF。应答信号 $\overline{ACK}$ 在 SCL 的第 9 个脉冲时发送，同时 SCL 引脚保持低电平。发送的数据必须送入 SSPBUF 缓冲器，同时也送入 SSPSR 寄存器。然后通过把 CKP 位 (SSPCON<4>) 置 1，使能 SCL 引脚。主控器件必须监控 SCL 引脚脉冲信号。从动器件可以通过延长 SCL 时钟的低电平，而使主控器件处于数据等待状态。 8 位数据在 SCL 时钟的下降沿被移位输出。这可确保在 SCL 为高电平期间 SDA 信号是有效的 (如图 17-13)。

每个数据发送字节都会产生一个 SSP 中断， SSPIF 标志位必须通过软件清零，状态寄存器 SSPSTAT 用于确定字节传输的状态。SSPIF 标志位是在第 9 个脉冲下降沿被置 1。

作为从动发送器，在第 9 个 SCL 时钟脉冲的上升沿锁存从主控接收器发出的 $\overline{ACK}$ 信号。若 SDA 线为高电平（无 $\overline{ACK}$ 应答信号），那么表示数据传输已完成。当无 $\overline{ACK}$ 被从动器件锁存时，从动逻辑被复位，并再监测下一个 START 信号的发生。如果 SDA 线是低电平 （有 $\overline{ACK}$ 应答信号），发送数据应装入 SSPBUF 缓冲器，并装入 SSPSR 寄存器，然后将 CKP 置 1，使能 SCL。

图 17-12： I^2C^{TM} 从动模式的接收时序图（7 位地址）

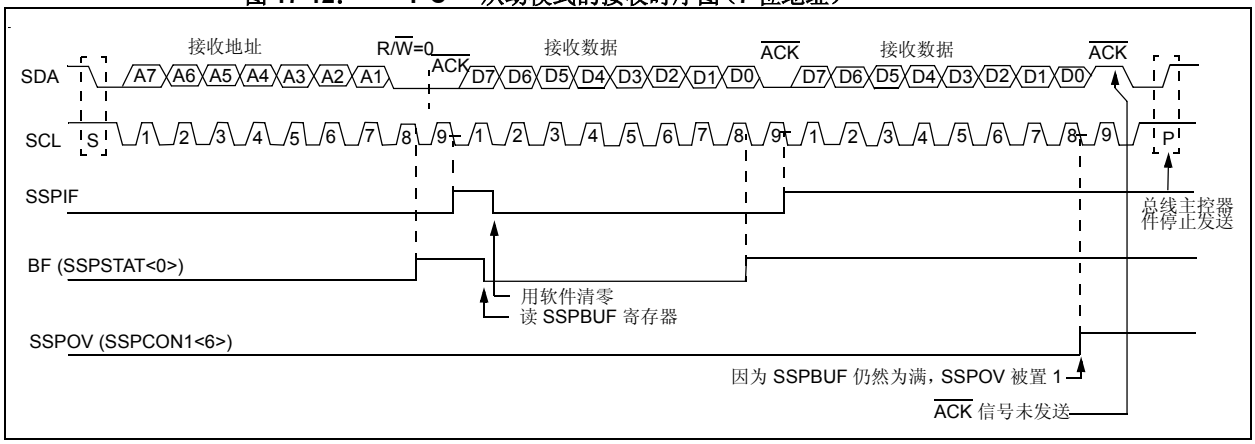


图 17-13： I^2C^{TM} 从动模式的发送时序图 (7 位地址)

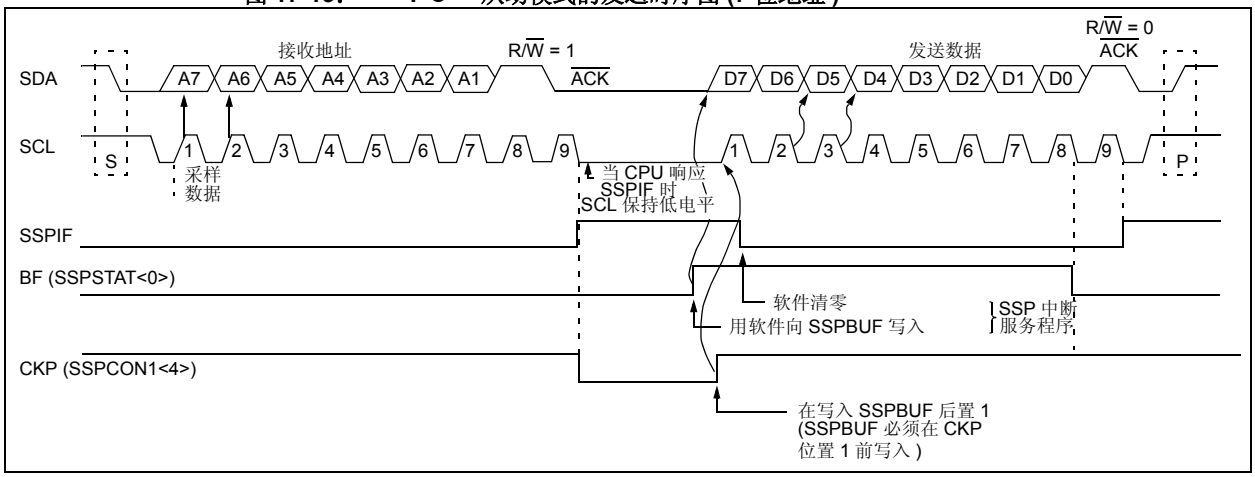


图 17-14： I^2C^{TM} 从动模式时序图 (发送， 10 位地址)

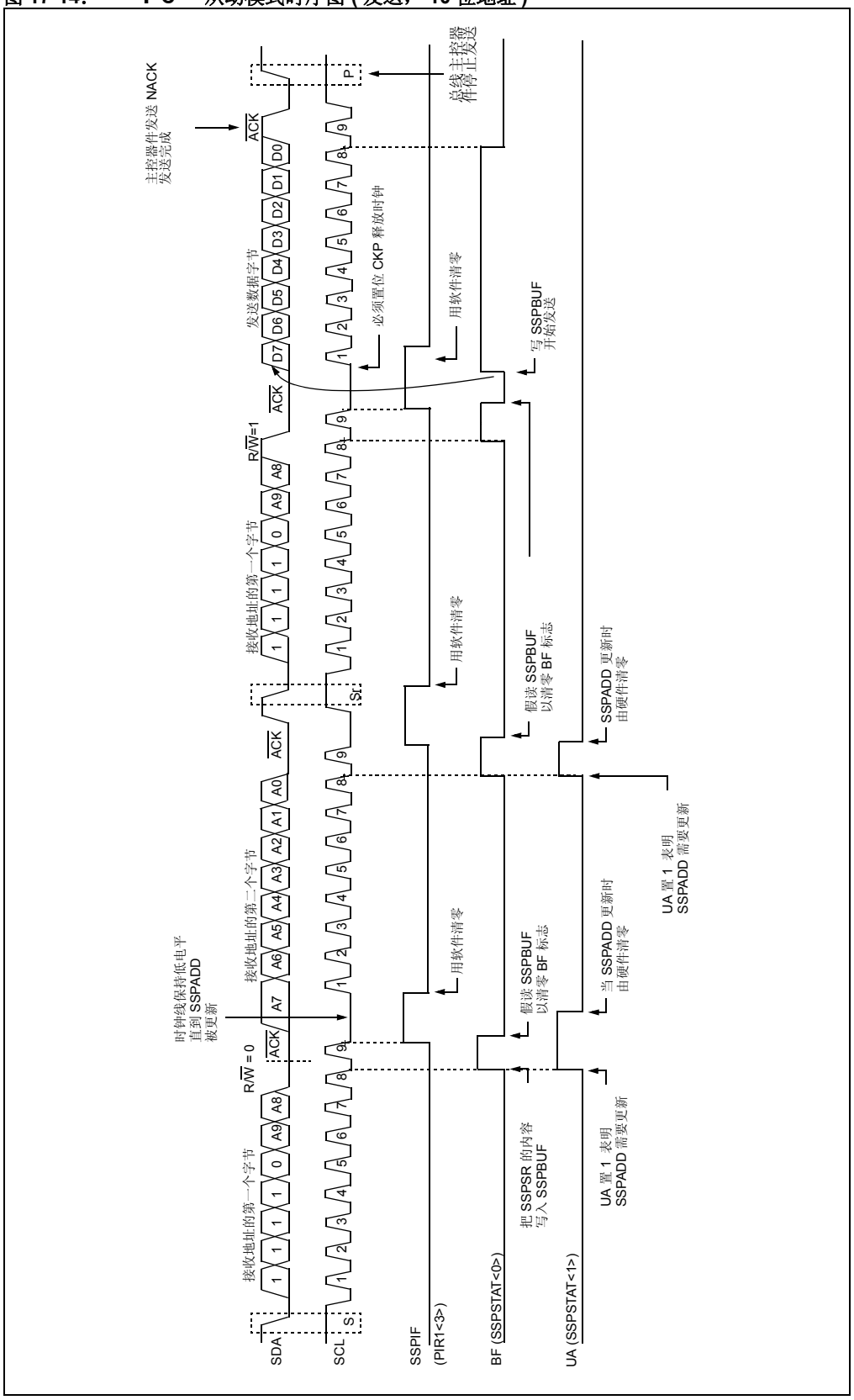
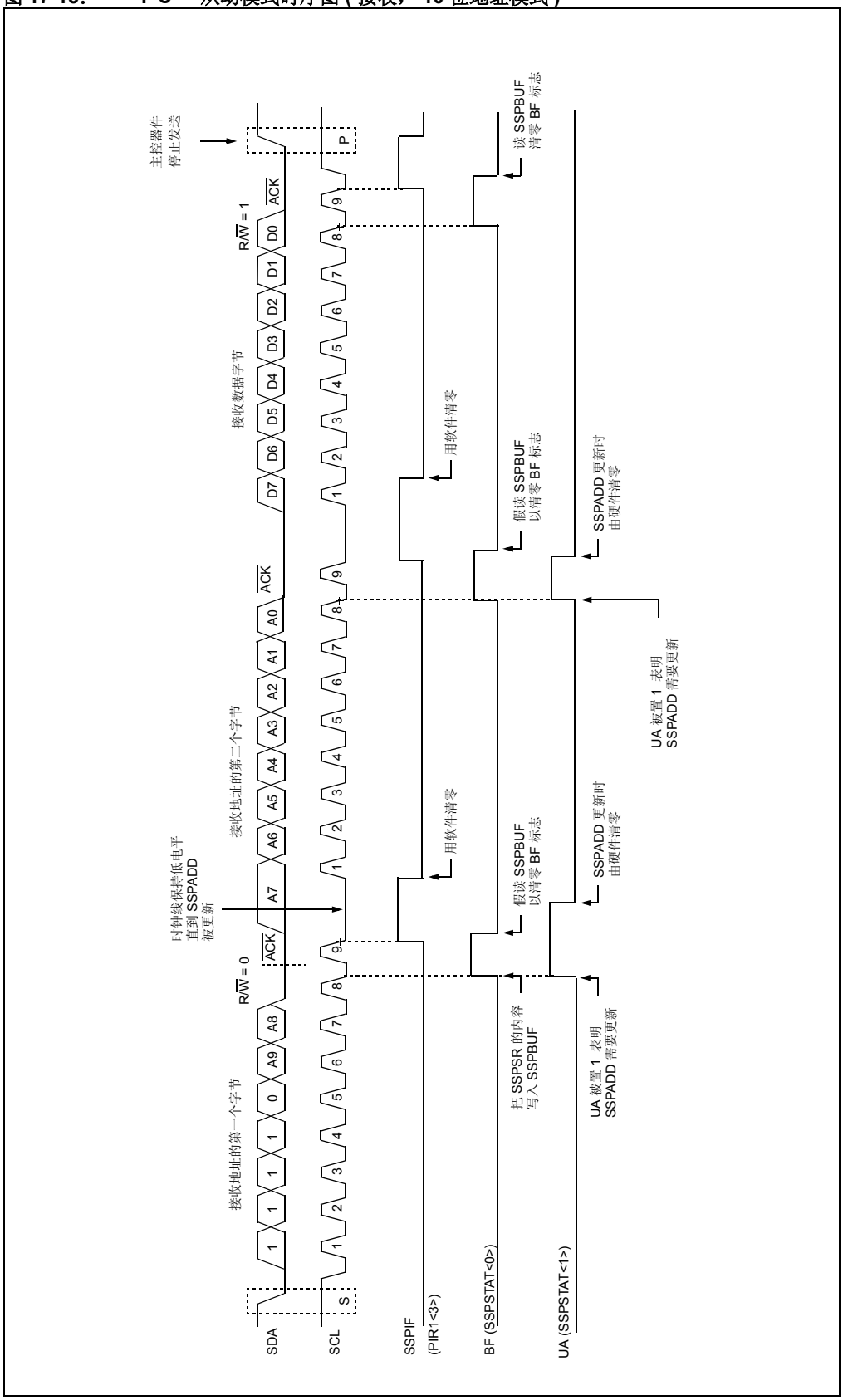


图 17-15： I^2C^{TM} 从动模式时序图 (接收， 10 位地址模式)



17.4.2 全局呼叫地址支持

对于 I²C 总线，是由启动条件后的第一个字节来决定哪个从机被主机寻址。而全局呼叫地址是个例外，它能够寻址所有的从机。理论上当使用这一地址时，所有的从机都应该发送一个应答响应。

全局呼叫地址是由 I²C 协议为特定用途保留的 8 个地址之一。 R/$\overline{W}$ = 0 时，这个地址的的所有位都为 0。

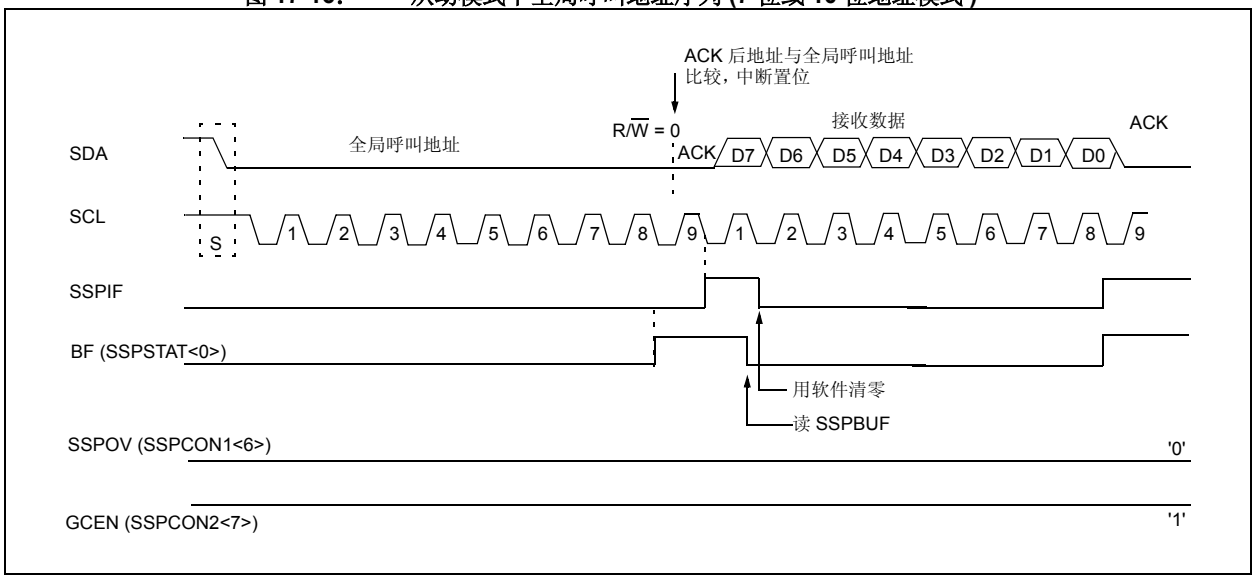
使能全局呼叫使能位 GCEN(SSPCON2<7> 置位) 时，即可识别全局呼叫地址。在检测到一个启动位之后，将 8 位数据移入 SSPSR，将地址和 SSPADD 作比较，同时也将地址和硬件设定的全局呼叫地址比较。

如果地址与全局呼叫地址匹配，则 SSPSR 中数据送入 SSPBUF，BF 标志位被置 1（第 8 位），在第 9 位 (ACK 位) 的下降沿， SSPIF 中断标志位被置 1。

当中断被响应时，通过读 SSPBUF 的内容检查中断源，判断是特定从机的地址还是全局呼叫地址。

在 10 位地址模式下，需要更新 SSPADD 以进行地址第二字节的匹配，并将 UA 位 (SSPSTAT<1>) 置 1。如果从机工作在 10 位地址模式下，当 GCEN 置 1 时采样到全局呼叫地址，那么将不需要进行地址第二字节的匹配，UA 位不会被置 1，并且从机在应答之后开始接收数据（图 17-16）。

图 17-16： 从动模式下全局呼叫地址序列 (7 位或 10 位地址模式)



17.4.3 休眠操作

在休眠方式下，I^2C模块能够接收地址或数据。并且地址匹配或字节传输完成时，如果允许 MSSP 中断，将唤醒 CPU。

17.4.4 复位的影响

复位操作会禁止 SSP 模块并停止当前的传输。

表 17-3： 与 I^2C^{TM} 操作有关的寄存器

寄存器名	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位 欠压复位时 的值	所有其 它复位时 的值
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 0000	0000 0000
PIR	SSPIF, BCLIF (1)								0, 0	0, 0
PIE	SSPIE, BCLIF (1)								0, 0	0, 0
SSPADD	同步串行口 (I^2C 模式) 地址寄存器 (从动模式)/ 波特率发生器 (主控模式)								0000 0000	0000 0000
SSPBUF	同步串行口的接收缓冲器 / 发送寄存器								xxxx xxxx	uuuu uuuu
SSPCON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	0000 0000	0000 0000
SSPSTAT	SMP	CKE	D/A	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000

其中： x = 未知 , u = 不变 , - = 未使用，读作 ‘0’。
阴影部分在 I^2C 模式下未用。

注 1：这些位的位置和器件的具体型号有关。
2：这些位也可称为 GPIE 和 GPIF。

17.4.5 主控模式

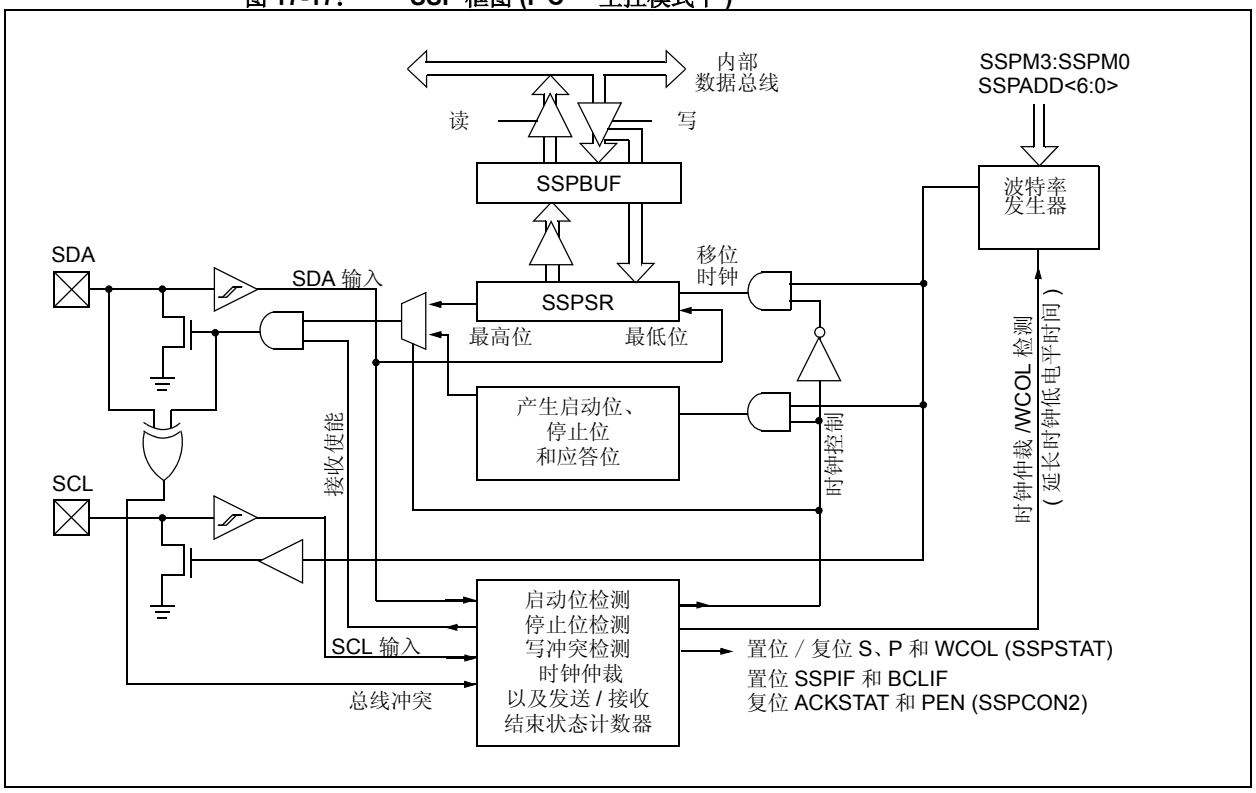
主控模式是通过检测 START 和 STOP 信号并引发中断来工作的。STOP (P) 和 START (S) 位在复位或禁止 SSP 模块时被清零。当 P 位被置 1，或 P 位和 S 位都清零而总线是空闲时，可以获得对 I^2C 总 线的控制权。

在主控模式下 SCL 和 SDA 线由 SSP 硬件控制。

下列事件会引起中断标志位 SSPIF 置 1 (如果允许中断，便产生 SSP 中断)：

- 启动条件
- 停止条件
- 数据传送字节的发送 / 接收
- 应答发送
- 重复启动

图 17-17： SSP 框图 (I^2C^{TM} 主控模式下)



17.4.6 多主机模式

在多主机模式下，利用检测启动条件（START）和停止条件（STOP）产生中断的功能，可以判断总线何时空闲。在复位或禁止 SSP 模块时，停止位 (P) 和启动位 (S) 位都被清零。当停止位 P 置 1 或 P 位和 S 位都为零而总线空闲时，获得对 I^2C 总线的控制权。当总线处于忙状态且 SSP 中断使能时，一旦检测到停止条件便产生中断。

工作在多主机模式时，SDA 线必须一直被监测，以判断信号电平是否是所期望的输出电平。该检测由硬件完成，结果保存在 BCLIF 位中。

以下几种情况可能会失去控制总线的机会：

- 地址传输
- 数据传输
- 启动条件
- 重复启动条件
- 应答条件

17.4.7 I^2C^{TM} 主控模式支持

通过对 SSPCON1 中的相应 SSPM 位置 1 或清零，并对 SSPEN 位置 1，可以使能主控模式。一旦主控模式被使能，用户有 6 种选择。

1. 在 SDA 和 SCL 线上送一个启动信号。
2. 在 SDA 和 SCL 线上送重复一个启动信号。
3. 写 SSPBUF 寄存器，启动数据 / 地址的发送。
4. 在 SDA 和 SCL 线上产生一个停止信号。
5. 设置 I^2C 端口以接收数据。
6. 在接收的数据字节末尾产生一个应答条件。

注：	SSP 模块的 I^2C 主控模式不允许事件排队。例如 : 不允许发出启动条件后 , 在启动条件完成前立即写 SSPBUF 寄存器以启动数据传送。在这种情况下，将不能写 SSPBUF，WCOL 将被置 1，表明没有发生对 SSPBUF 的写操作。

17.4.7.1 I^2C^{TM} 主控模式操作

主机产生所有串行时钟脉冲和启动 / 停止信号。当停止信号或重复启动信号到来时中止传送。因为重复启动信号也是下一个串行传送的开始，因此不会释放 I^2C 总线。

在主发送器模式下，通过 SDA 线输出串行数据，而 SCL 线输出串行时钟。发送的第一个字节包括接收器件的从机地址 （7 位）和读 / 写位。在这种情况下 R/$\overline{W}$ 为逻辑 '0'。每次发送 8 位串行数据。在发送每个字节后，会接收到一个应答位。启动和停止条件分别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发送器件的从机地址和读 / $\overline{写位}$。这时 R/$\overline{W}$ 为逻辑 ‘1‘。因此发送的第一个字节是一个 7 位的从机地址，和一个表明接收的位 '1'。通过 SDA 接收串行数据，SCL 输出串行时钟。每次接收 8 位串行数据。接收每个字节后，都发送一个应答位。启动和停止条件分别表明传输的开始和结束。

用于 SPI 模式操作的波特率发生器，在 I^2C 模式下用来将 SCL 时钟频率设置为 100kHz、400kHz 或 1MHz。波特率发生器的重载值保存在 SSPADD 寄存器的低 7 位中。当对 SSPBUF 进行写操作时，波特率发生器自动开始计数。一旦指定的操作完成 （即最后一个数据位发送后紧跟一个 ACK），内部时钟将自动停止计数，SCL 引脚保持在最后的状态。

下面是一个典型的发送序列：

a) 用户通过将启动使能位 SEN (SSPCON2<0>) 置 1 来产生启动条件。
b) SSPIF 置 1。在进行任何其它操作前，SSP 模块将等待所需的启动时间。
c) 用户将地址装入 SSPBUF 进行发送。
d) 地址从 SDA 引脚移出，直到发送完所有 8 位。
e) SSP 模块移入来自从机的 ACK 位，并将它的值写入 SSPCON2 寄存器 (SSPCON2<6>)。
f) SSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
g) 用户将 8 位数据装入 SSPBUF。
h) 数据从 SDA 引脚移出，直到发送完所有 8 位。
i) SSP 模块移入来自从机的 ACK 位，并将它的值写入 SSPCON2 寄存器 (SSPCON2<6>)。
j) SSP 模块在第 9 个时钟周期的末尾将 SSPIF 置 1，产生一个中断。
k) 用户通过将停止使能位 PEN (SSPCON2<2>) 置 1 来产生停止条件。
l) 一旦停止条件完成，将产生一个中断。

17.4.8 波特率发生器

在 I²C 主控模式下，波特率发生器 BRG 的重载值位于 SSPADD 寄存器的低 7 位（图 17-18）。当 BRG 装入该值后，BRG 递减计数直至 0，然后停止，等待再次装入。在 I²C 主控模式下，BRG 自动重新装入值。如果发生了时钟仲裁， BRG 将在 SCL 引脚采样到高电平时重新装入值（图 17-19）。

图 17-18： 波特率发生器框图

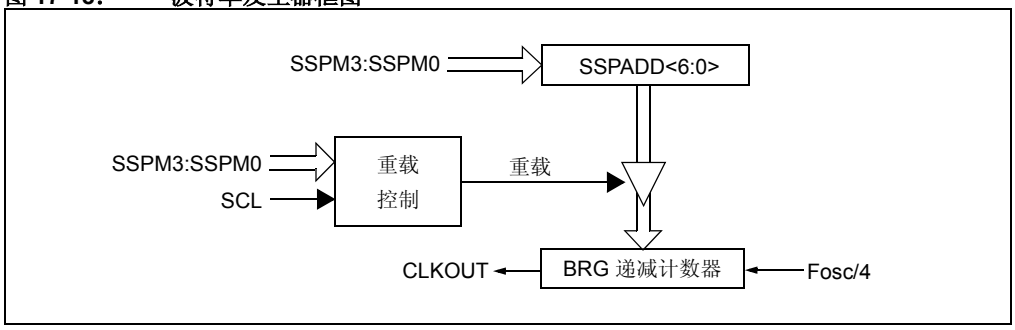
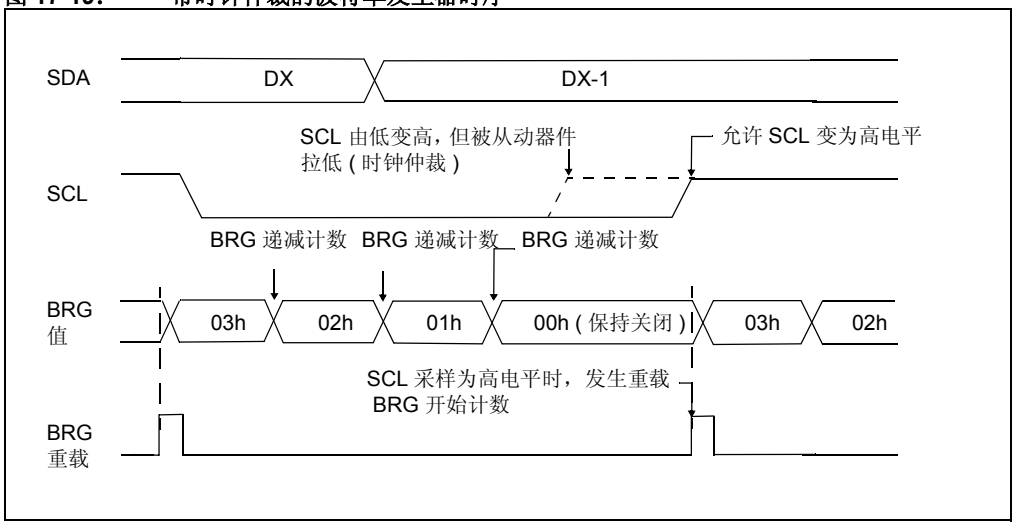


图 17-19： 带时钟仲裁的波特率发生器时序



17.4.9 I^2C^{TM} 主控模式启动条件时序

为产生启动条件，用户应将启动条件使能位 SEN (SSPCON2<0>) 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<6:0> 的内容并开始递减计数。当波特率发生器超时 （T_{BRG}）溢出时，如果 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 由高电平拉到低电平将产生一个启动条件，并使 S 位 (SSPSTAT<3>) 置 1。随后波特率发生器重新装入 SSPADD<6:0> 的内容并重新开始计数。当波特率发生器超时（T_{BRG}）溢出时，SEN 位 (SSPCON2<0>) 将自动被硬件清零，波特率发生器暂停工作，SDA 保持低电平，启动条件完成。

注：	如果在启动条件的开始，SDA 和 SCL 引脚已经采样为低电平，或者启动条件期间，在 SDA 线被拉低之前，SCL 线已经采样为低电平，则会产生总线冲突，总线冲突标志位 BCLIF 被置 1，启动条件中止， I^2C 模块复位到空闲状态。

17.4.9.1 WCOL 状态标志位

当启动条件正在进行时，如果用户写 SSPBUF，则 WCOL 被置 1，同时缓冲器内容不变 (写操作无效)。

注：	由于不允许事件排队，在启动条件完成之前，不能对 SSPCON2 的低 5 位进行写操作。

图 17-20： 第一个启动位的时序图

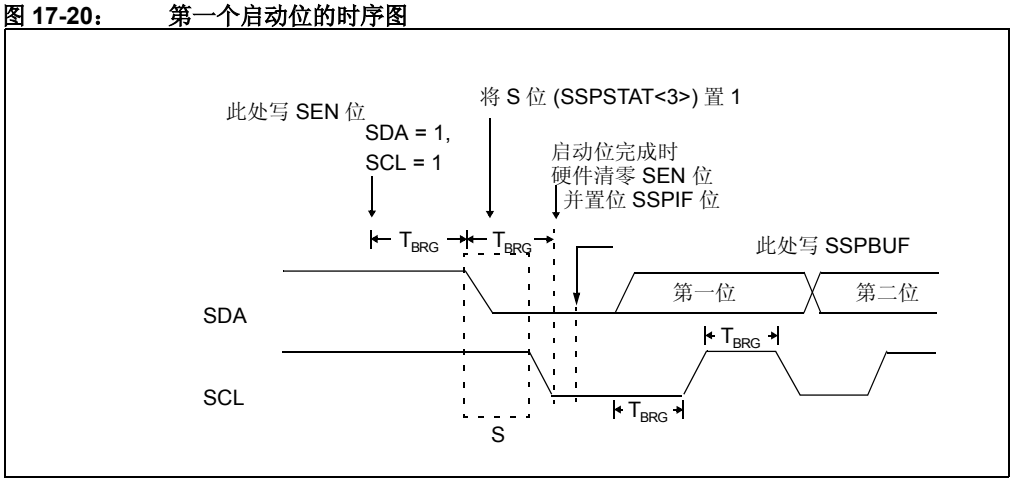
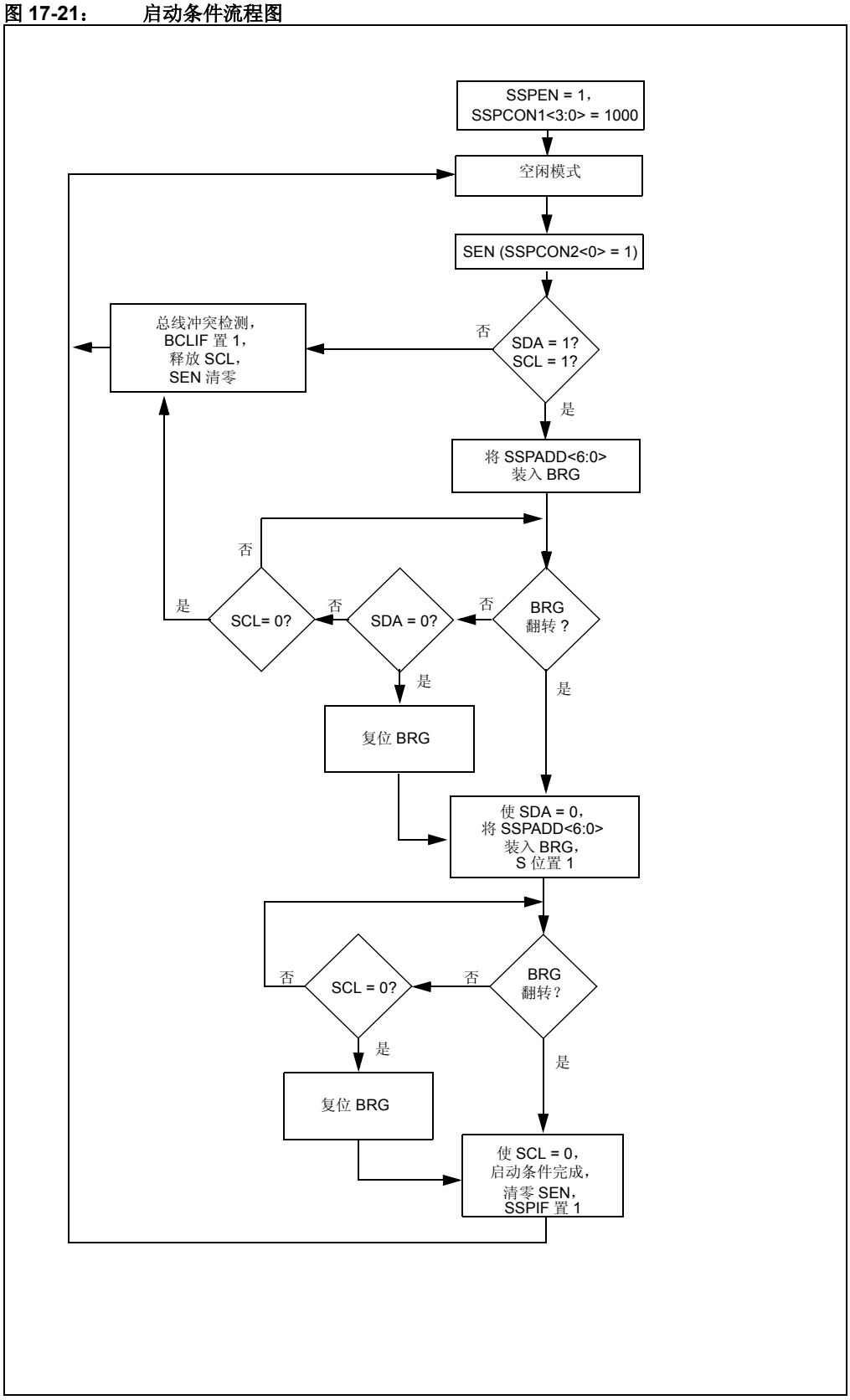


图 17-21： 启动条件流程图



17.4.10 I^2C^{TM} 主控模式重复启动条件时序

当 RSEN 位 (SSPCON2<1>) 设置为高电平，并且 I^2C 逻辑模块处于空闲状态时，产生重复启动条件。当 RSEN 位置 1 时， SCL 引脚为低电平有效。当 SCL 引脚采样为低电平时，波特率发生器装入 SSPADD<5:0> 的内容，并开始计数。在一个波特率发生器计数周期 (T_{BRG}>) 内 SDA 引脚被释放，其引脚电平被拉高。当波特率发生器超时溢出时，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<6:0> 的内容并开始计数。在一个计数周期 T_{BRG} 内， SDA 和 SCL 必须保持为高电平。接着在下一个 T_{BRG} 中，当 SCL 为高电平时，将 SDA 引脚拉为低电平。然后 RSEN 位 (SSPCON2<1>) 将自动清零，波特率发生器不再被重新装入值， SDA 引脚保持低电平。一旦在 SDA 和 SCL 引脚上检测到启动条件，启动标志位 S (SSPSTAT<3>) 将被置 1。直到波特率发生器超时溢出，SSPIF 位才会置 1。

注 1: 有任何其它事件在进行时，对 RSEN 的设置无效。

注 2: 在重复启动条件期间，下列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。这表明另一个主机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 位地址写入 SSPBUF，或者在 10 位地址模式下写入默认的第一个地址字节。当第一个 8 位数据发送完并接收到一个 $\overline{ACK}$ 后，用户可以发送另一个 8 位地址 (10 位地址模式下) 或 8 位数据 (7 位地址模式下)。

17.4.10.1 WCOL 状态标志位

当重复启动序列进行时，如果用户写 SSPBUF，则 WCOL 被置 1，同时缓冲器内容不变 (写操作无效)。

注：	由于不允许事件排队，在重复启动条件结束之前，不能对 SSPCON2 的低 5 位进行写操作。

图 17-22： 重复启动条件的时序图

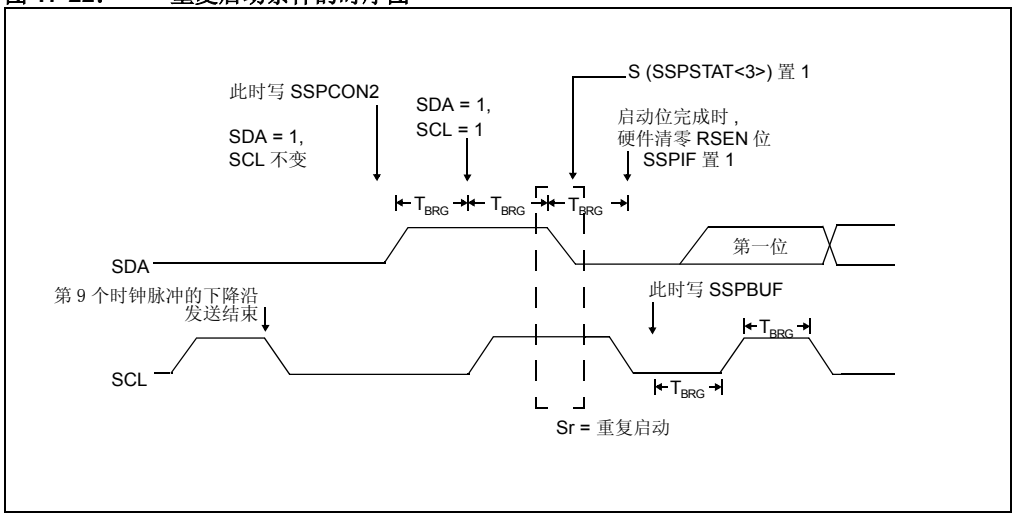


图 17-23： 重复启动条件的流程图 （1）

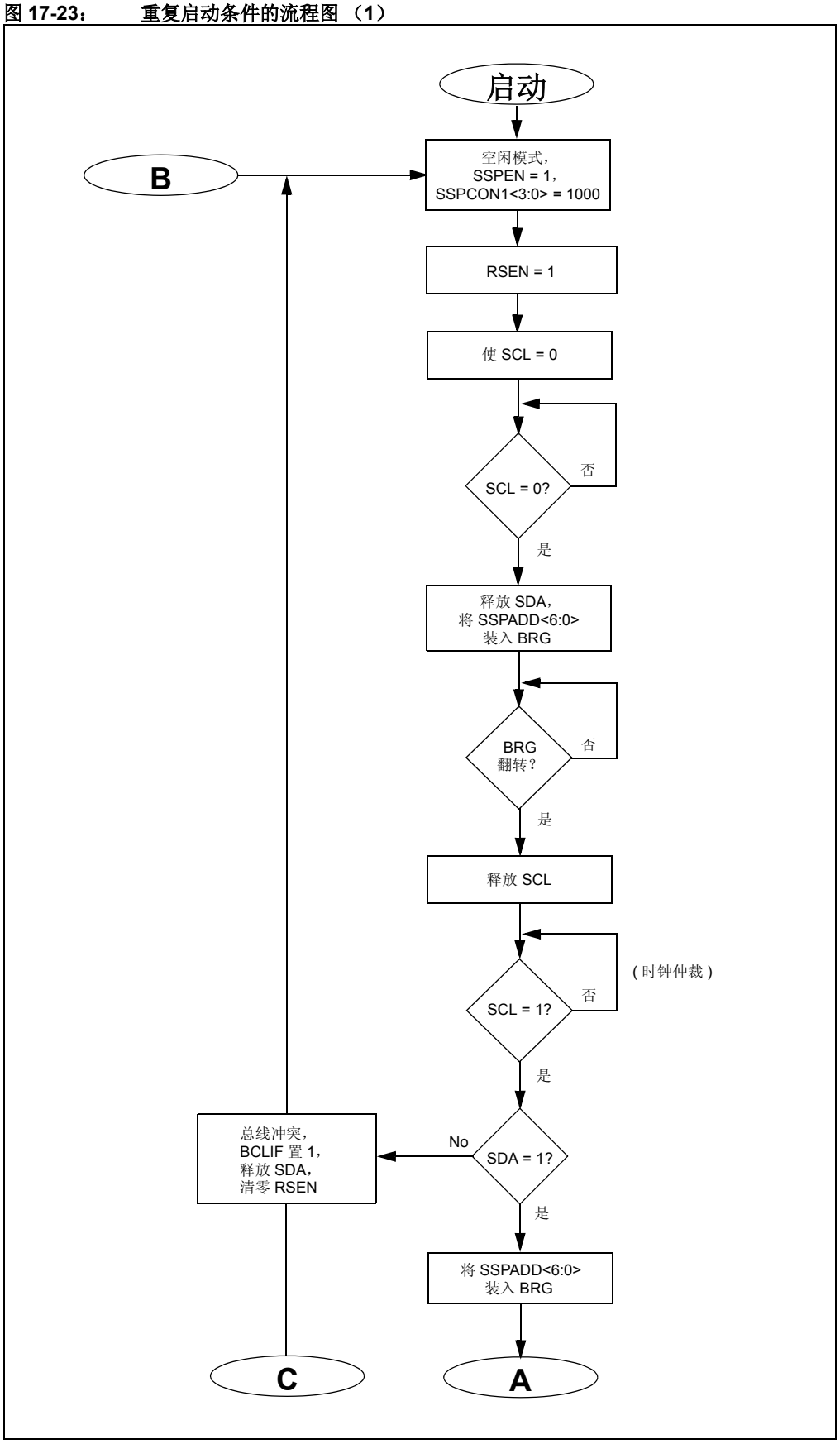
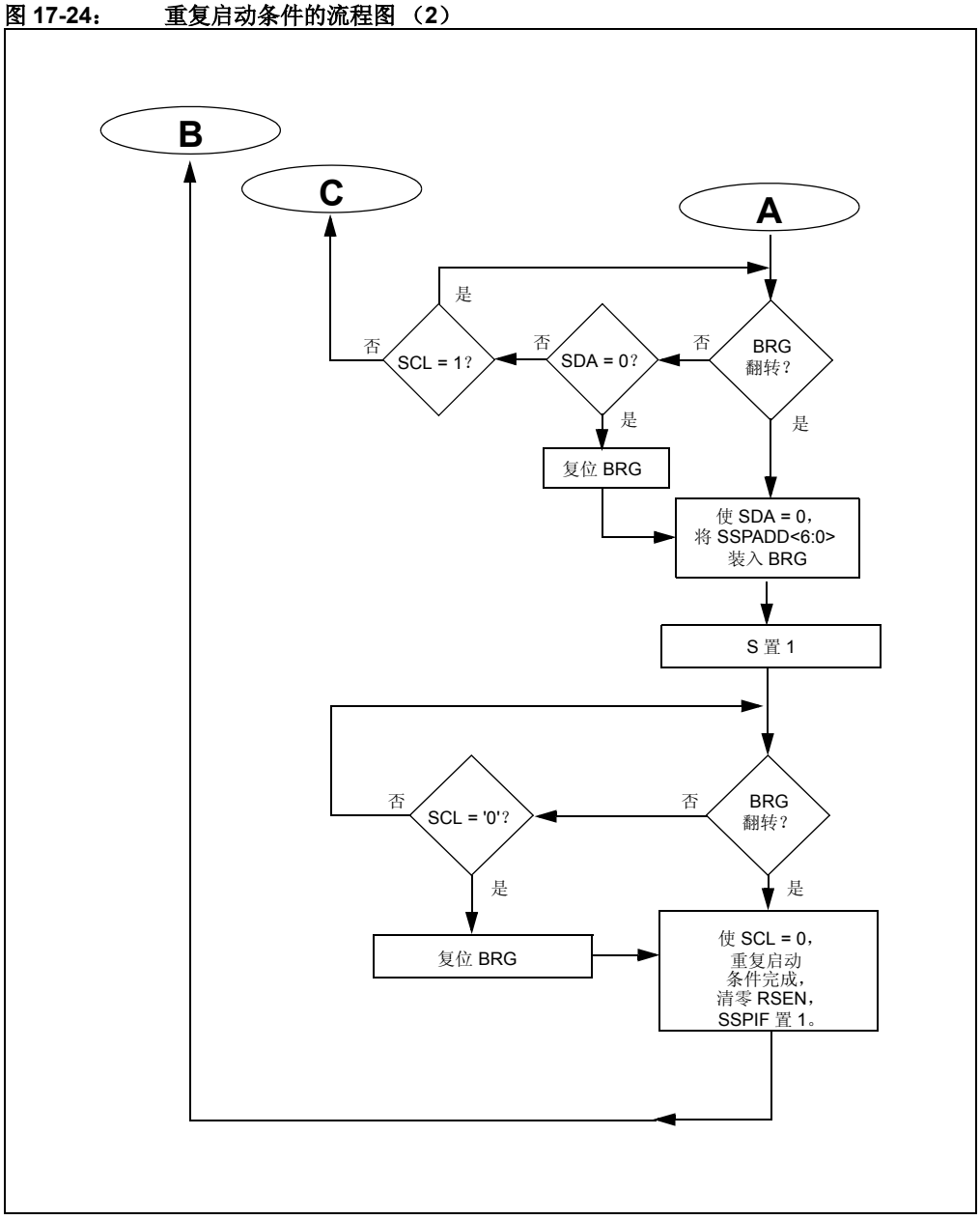


图 17-24： 重复启动条件的流程图 （2）



17.4.11 I^2C^{TM} 主控模式下的发送

发送一个数据字节、一个 7 位地址或一个 10 位地址的某个字节，都可以通过写一个值到 SSPBUF 寄存器来实现。该操作将使缓冲器满标志位 BF 置 1，并且波特率发生器开始计数，同时启动下一次发送。在 SCL 的下降沿后 (参见数据保持时间规范参数 106)，地址 / 数据的每一位都从 SDA 引脚移出。在一个波特率发生器翻转计数周期 (T_{BRG}) 内，SCL 保持低电平。数据应该在 SCL 上升到高电平前有效 (参见数据建立时间规范参数 107)。当 SCL 引脚被释放为高电平时 （保持高电平的时间为 T_{BRG}），在 SCL 引脚保持为为高电平的这个 T_{BRG} 内，以及下一个 SCL 时钟脉冲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 8 位都移出之后 (第 8 个时钟的下降沿)，BF 标志位清零，同时主机释放 SDA，此时如果发生地址匹配或是数据被正确接收，被寻址的从机将在第 9 个时钟脉冲时响应一个 $\overline{ACK}$ 位。 $\overline{ACK}$ 的状态在第 9 个时钟的下降沿写入 ACKDT。主机收到应答响应之后，应答状态位 ACKSTAT 清零。如果未收到响应，该位被置 1。第 9 个时钟脉冲之后， SSPIF 位置 1，主时钟 (波特率发生器) 暂停，直到下一个数据装入 SSPBUF， SCL 引脚保持低电平，同时 SDA 引脚电平不变 (图 17-26)。

在写 SSPBUF 之后，地址的每一位在 SCL 的下降沿被移出，直至所有 7 位地址和 R/$\overline{W}$ 位都移出。在第 8 个时钟的下降沿，主机将 SDA 引脚上拉为高电平，以允许从机发出一个应答响应。在第 9 个时钟下降沿，主机通过采样 SDA 引脚来判断地址是否被从机识别。 $\overline{ACK}$ 位状态被装入 ACKSTAT 状态位 (SSPCON2<6>)。在地址发送的第 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。

17.4.11.1 缓冲器满状态标志位 BF

在发送模式下，BF 位 (SSPSTAT<0>) 在 CPU 写 SSPBUF 时置 1，在所有 8 位数据移出后清零。

17.4.11.2 写冲突状态标志位 WCOL

如果用户在发送过程中 (即 SSPSR 仍在移出数据字节) 写 SSPBUF，则 WCOL 置 1，缓冲器内容不变 (写操作无效)。

WCOL 必须用软件清零。

17.4.11.3 应答状态标志位 ACKSTAT

在发送模式下，当从机发送应答信号位 ($\overline{ACK}$ = 0) 时，ACKSTAT 位 (SSPCON2<6>) 清零，当从机不应答 ($\overline{ACK}$ = 1) 时，ACKSTAT 位 (SSPCON2<6>) 置 1。从机在识别出其地址 (包括全局呼叫地址) 或正确接收数据后，发送一个应答信号。

图 17-25： 主控模式发送的流程图

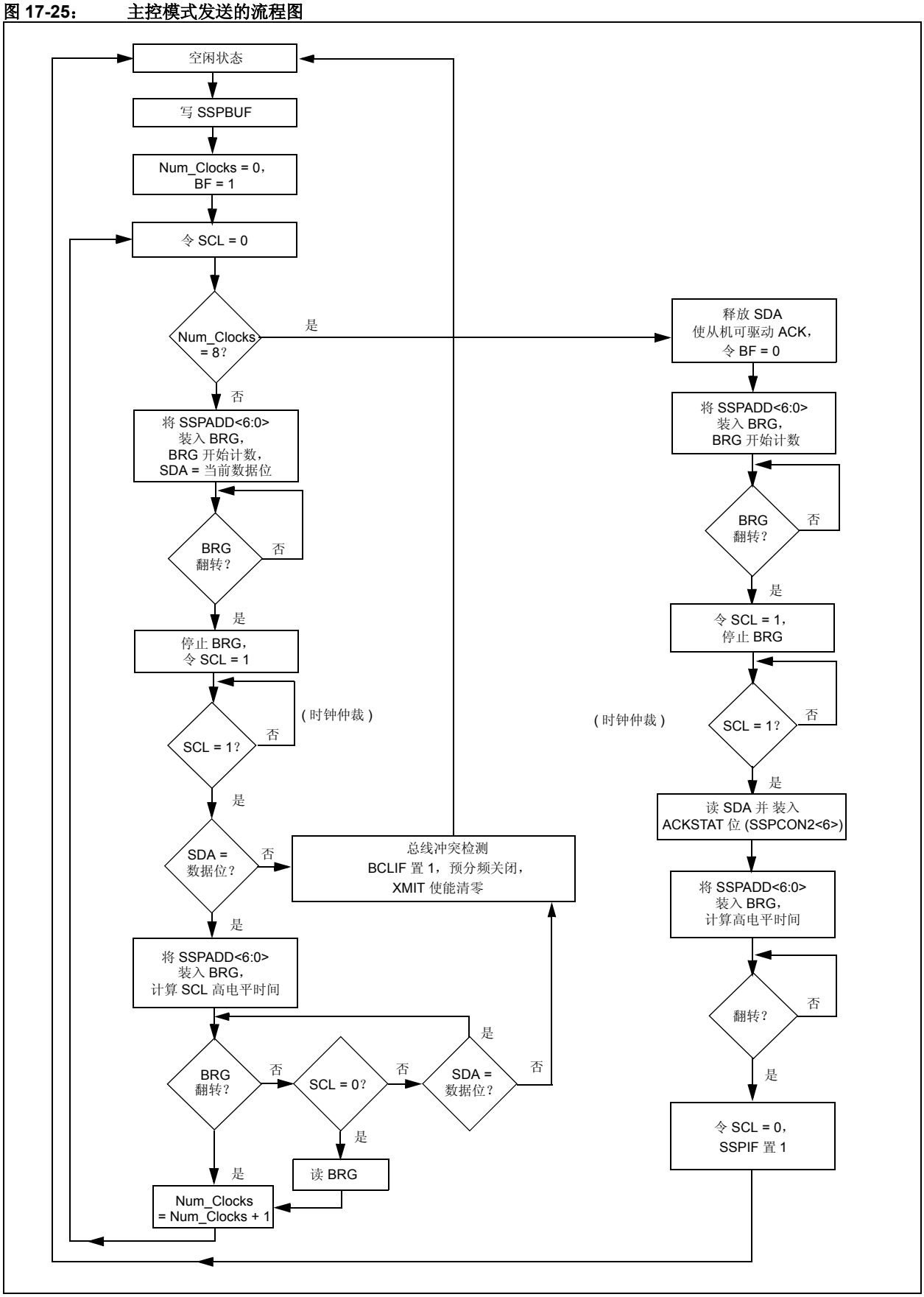
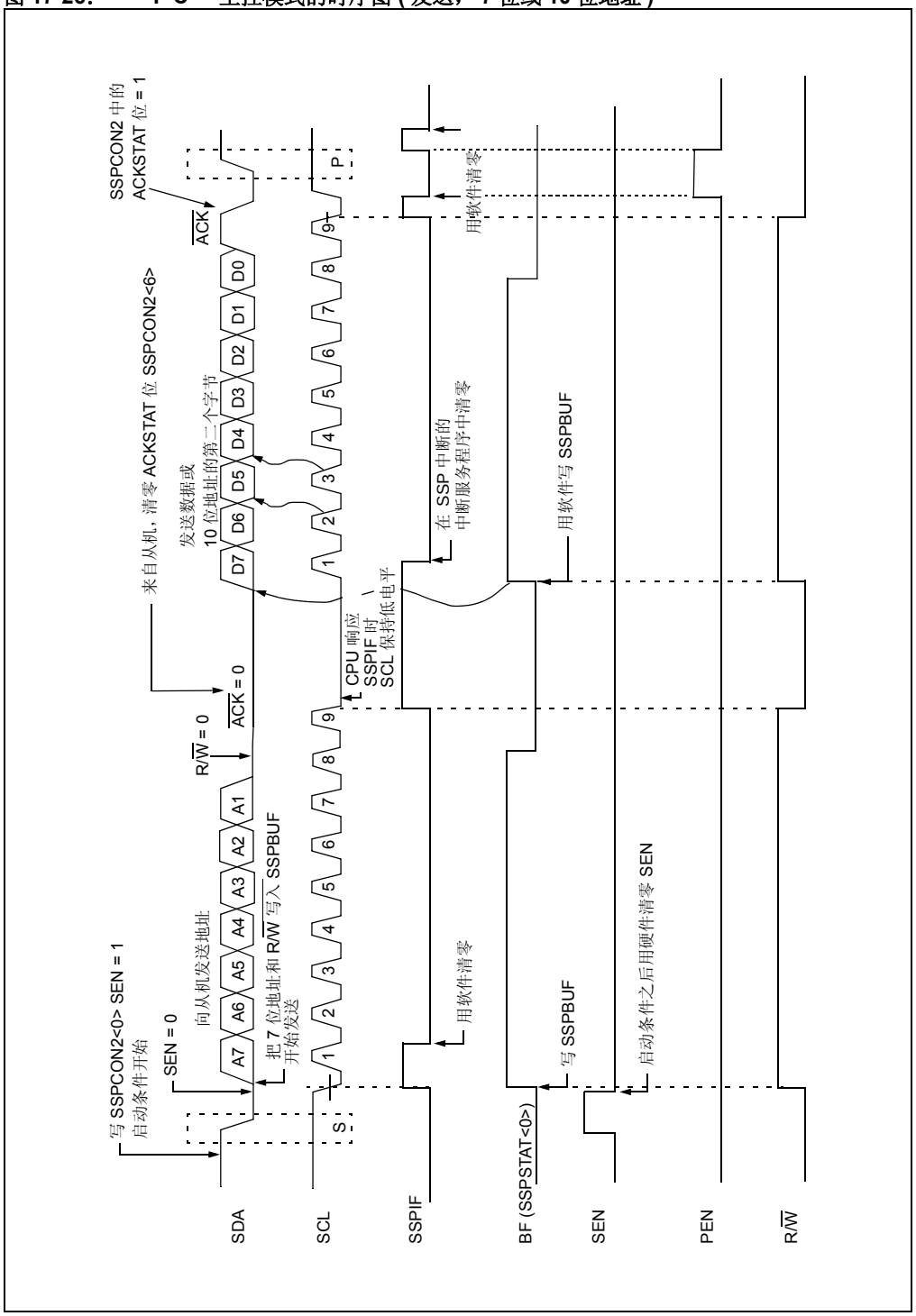


图 17-26： I^2C^{TM} 主控模式的时序图 (发送， 7 位或 10 位地址)



17.4.12 I^2C^{TM} 主控模式接收

将接收使能位 RCEN (SSPCON2<3>) 置 1 便进入主控模式接收状态。

注：	RCEN 位置 1 前， SSP 必须 处于空闲状态，否则对 RCEN 位的置位将无效。

波特率发生器开始计数，每次翻转时， SCL 引脚的状态改变 (由高变低或由低变高)，数据被移入 SSPSR。第 8 个时钟下降沿之后，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，保持 SCL 为低电平。SSP 进入空闲状态，等待下一条命令。当 CPU 读缓冲器时， BF 标志位自动清零。通过将应答序列使能位 ACKEN (SSPCON2<4>) 置 1，用户可以在接收结束后发出应答信号。

17.4.12.1 缓冲器满状态标志位 BF

接收时，BF 位在将 SSPSR 中的地址或数据字节装入 SSPBUF 时置 1，在读 SSPBUF 寄存器时清零。

17.4.12.2 SSP 溢出状态标志位 SSPOV

接收时，如果在 SSPSR 接收到 8 位数据时，BF 标志位已经在上一次接收时置 1，则 SSPOV 位被置 1。

17.4.12.3 写冲突状态标志位 WCOL

如果在接收过程中 (即 SSPSR 仍在移位数据字节) 又写 SSPBUF，则 WCOL 置 1，缓冲器内容不变 (写操作无效)。

图 17-27： 主控模式接收的流程图

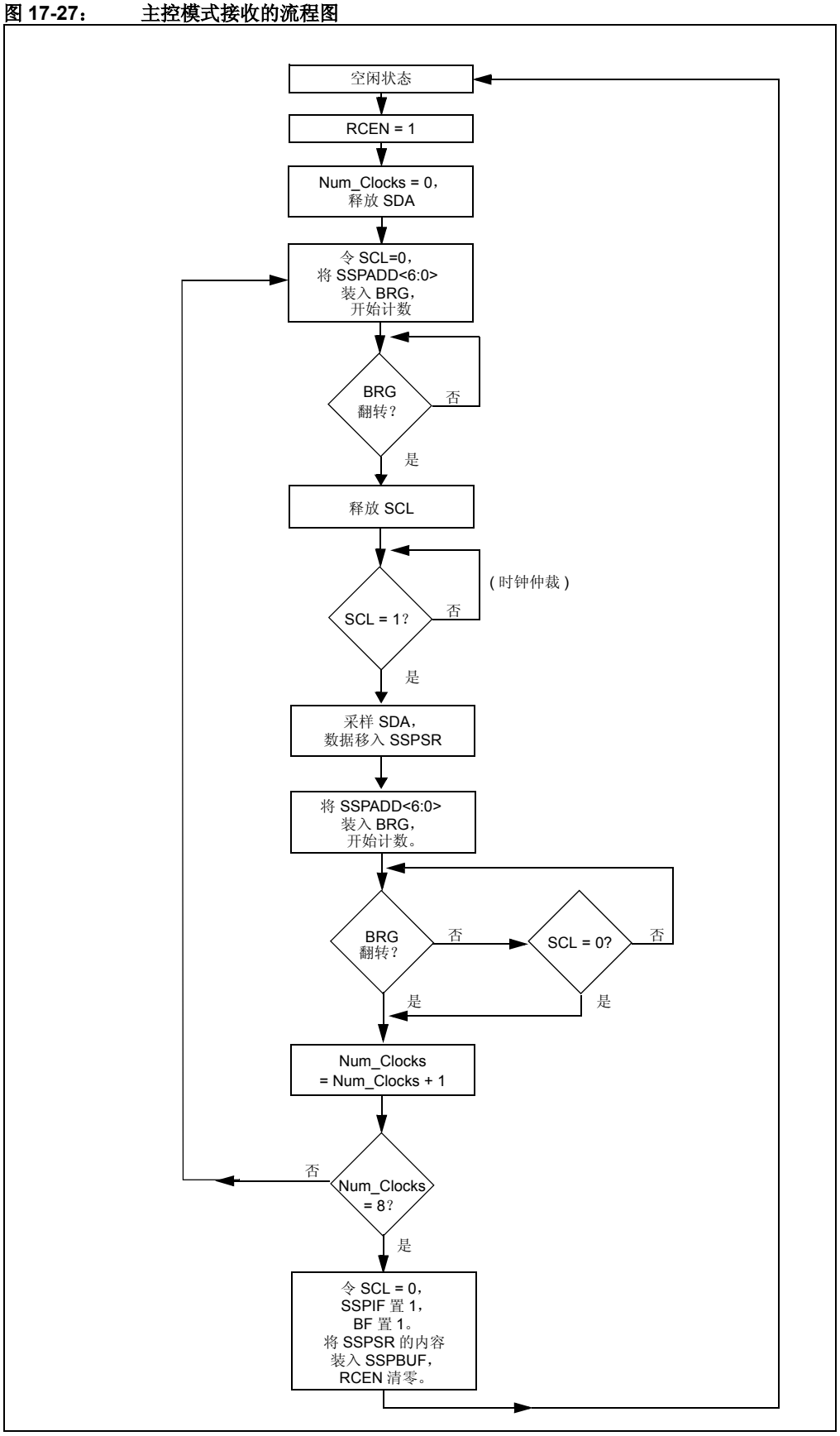
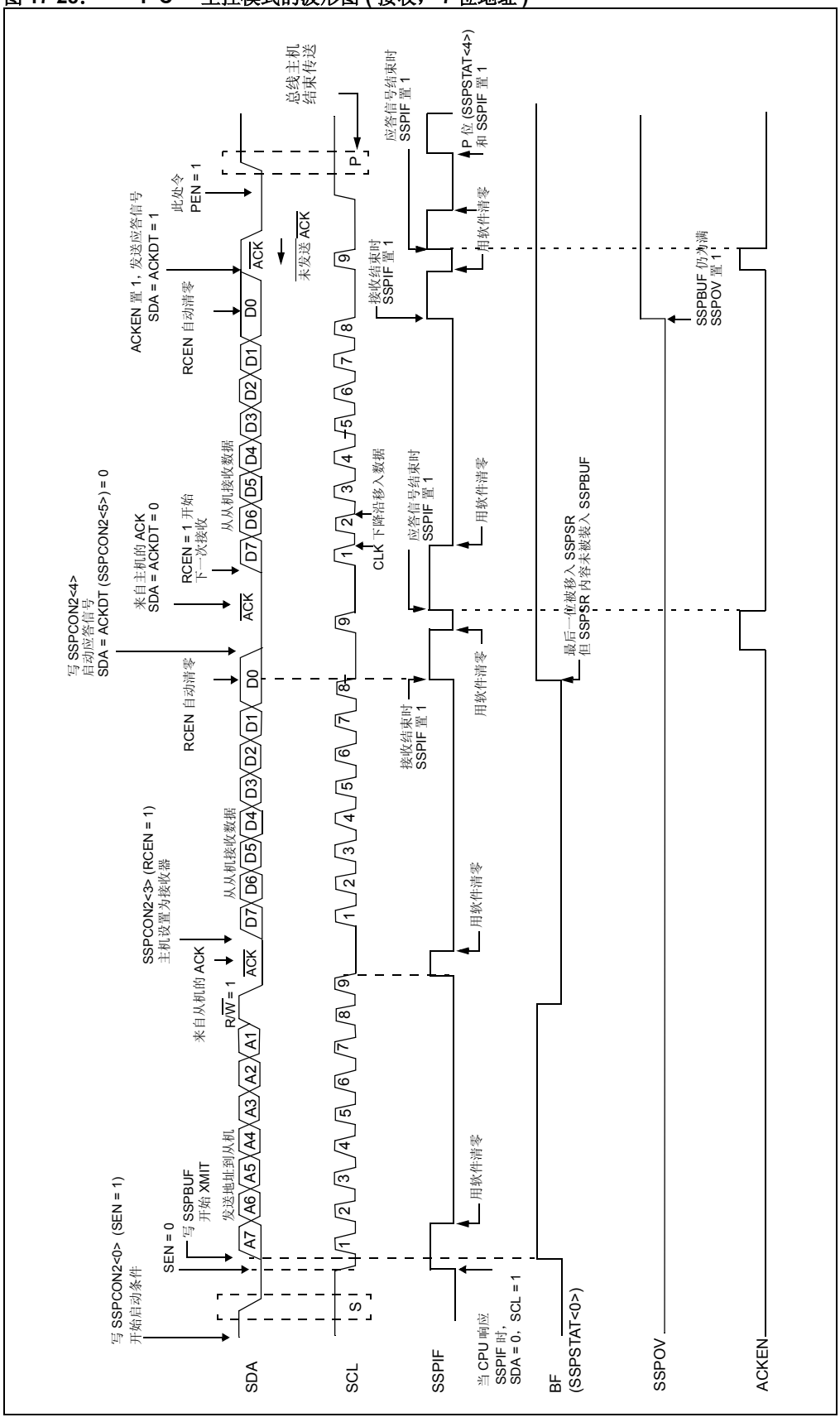


图 17-28： I^2C^{TM} 主控模式的波形图 (接收， 7 位地址)



17.4.13 应答时序

将应答使能位 ACKEN (SSPCON2<4>) 置 1，就允许产生应答信号。当该位被置 1 时，SCL 引脚被拉低，应答数据位的内容出现在 SDA 引脚上。如果用户希望产生一个应答，ACKDT 位应清零。否则应在应答信号开始前将 ACKDT 位置 1。波特率发生器进行一个翻转周期 (T_{BRG}) 的计数，T_{BRG} 到时，SCL 引脚被释放 (拉高)。当 SCL 引脚采样为高电平时 (时钟仲裁)，波特率发生器进行一个 T_{BRG} 周期的计数，SCL 引脚被拉低。在这之后，ACKEN 位自动清零，波特率发生器关闭，SSP 模块进入空闲模式 (图 17-29)。

17.4.13.1 写冲突状态标志 WCOL

在发送应答信号过程中，如果写 SSPBUF，WCOL 将被置 1，并且缓冲器的内容不变 (写操作无效)。

图 17-29： 应答序列时序图

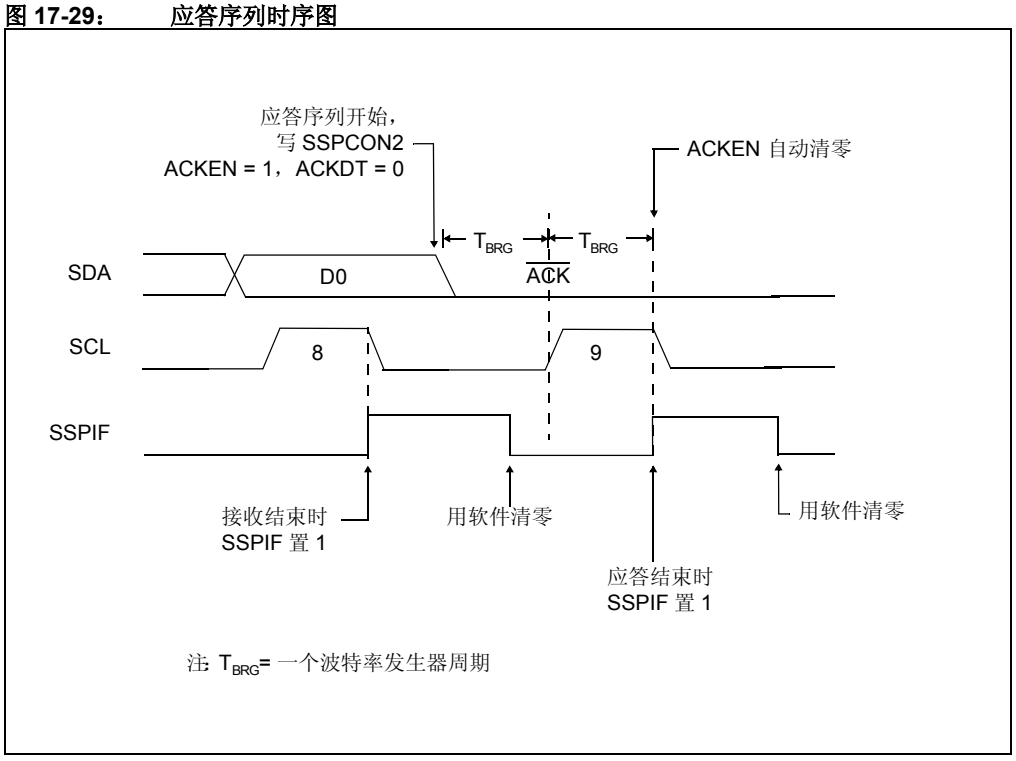
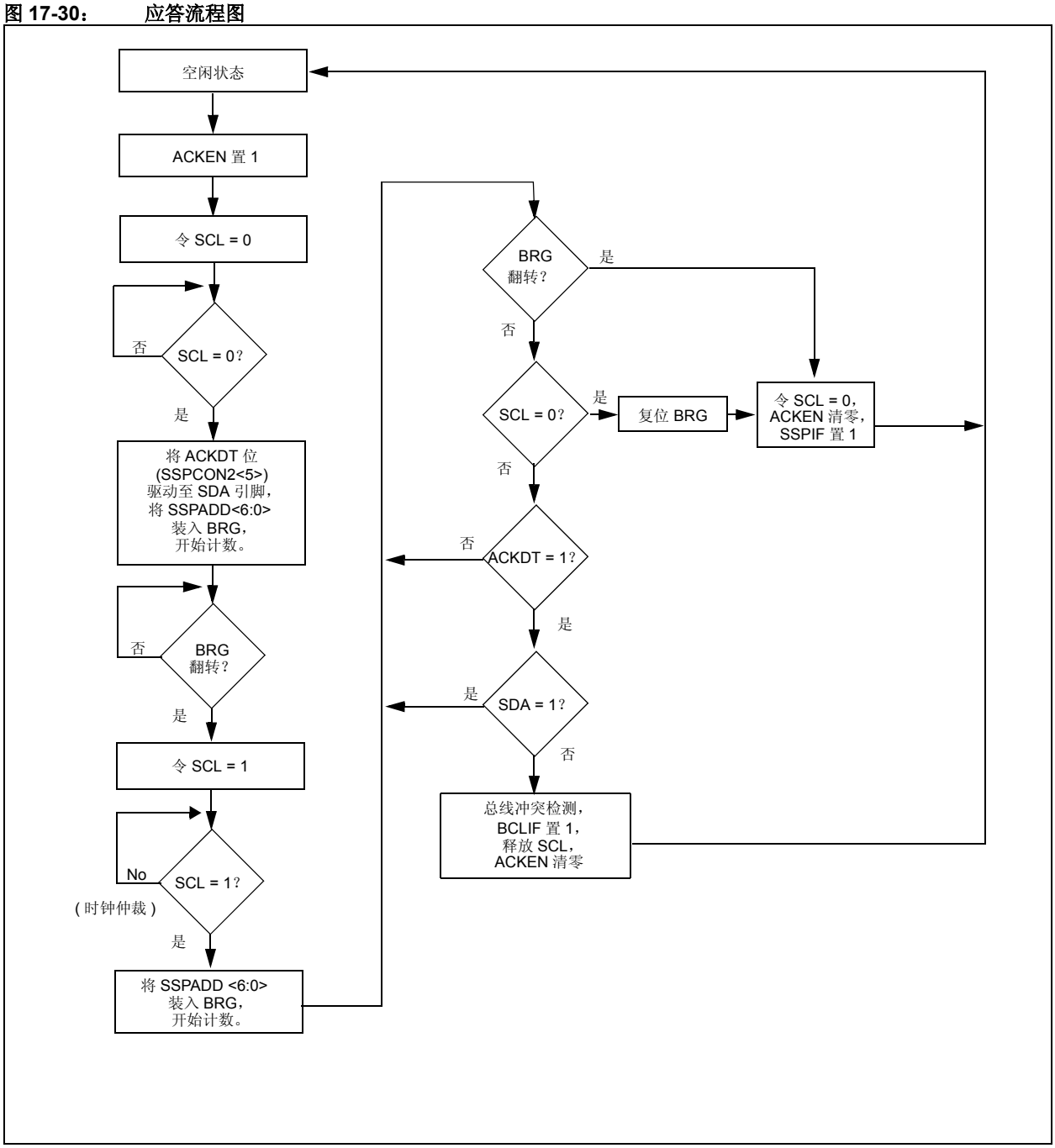


图 17-30： 应答流程图


空闲状态
ACKEN 置 1
令 SCL = 0
否
SCL = 0?
是
将 ACKDT 位
(SSPCON2<5>)
驱动至 SDA 引脚，
将 SSPADD<6:0>
装入 BRG，
开始计数。
否
BRG
翻转?
是
令 SCL = 1
No
SCL = 1?
(时钟仲裁)
是
将 SSPADD <6:0>
装入 BRG，
开始计数。
BRG
翻转?
是
否
SCL = 0?
是
复位 BRG
令 SCL = 0，
ACKEN 清零，
SSPIF 置 1
否
否
ACKDT = 1?
是
是
SDA = 1?
否
总线冲突检测，
BCLIF 置 1，
释放 SCL，
ACKEN 清零

17.4.14 停止条件时序

将停止序列使能位 PEN (SSPCON2<2>) 置 1，在接收 / 发送结束后，将通过 SDA 引脚的电平变化产生停止位。在接收 / 发送结束后， SCL 在第 9 个时钟下降沿之后保持低电平。当 PEN 位置 1 时，主机将 SDA 线置为低电平。当 SDA 采样为低电平时，波特率发生器被重新装入值并递减计数至 0。当波特率发生器溢出时， SCL 引脚被拉到高电平。一个 T_{BRG}（波特率发生器翻转周期）之后， SDA 引脚被重新拉高。当 SDA 引脚采样为高电平且 SCL 也是高电平时， P 位 (SSPSTAT<4>) 置 1。一个 TBRG 之后， PEN 位清零，同时 SSPIF 位置 1 (图 17-31)。

无论什么时候固件要取得总线控制权，首先都要检查 SSPSTAT 寄存器中的 S 和 P 位，以确定总线是否处于忙状态。如果总线忙，那么当检测到一个停止位时 (即总线空闲)，则向 CPU 发出 中断请求 （或通知）。

17.4.14.1 写冲突状态标志位 WCOL

当处于停止条件过程中，如果用户写 SSPBUF，WCOL 将被置 1，并且缓冲器的内容不变 (写操作无效)。

图 17-31： 停止条件接收或发送模式

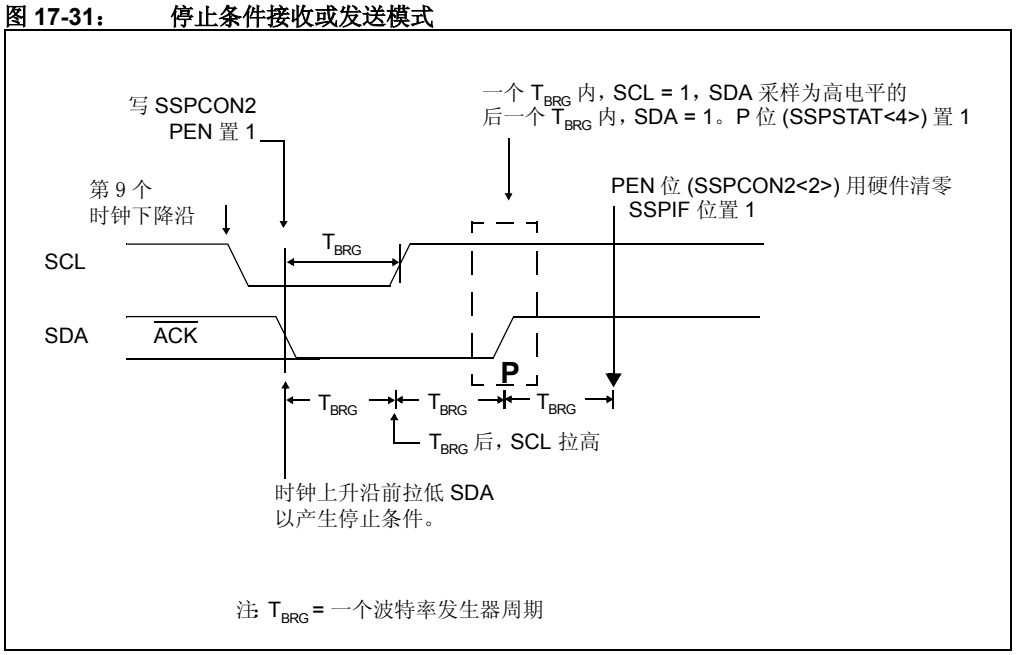
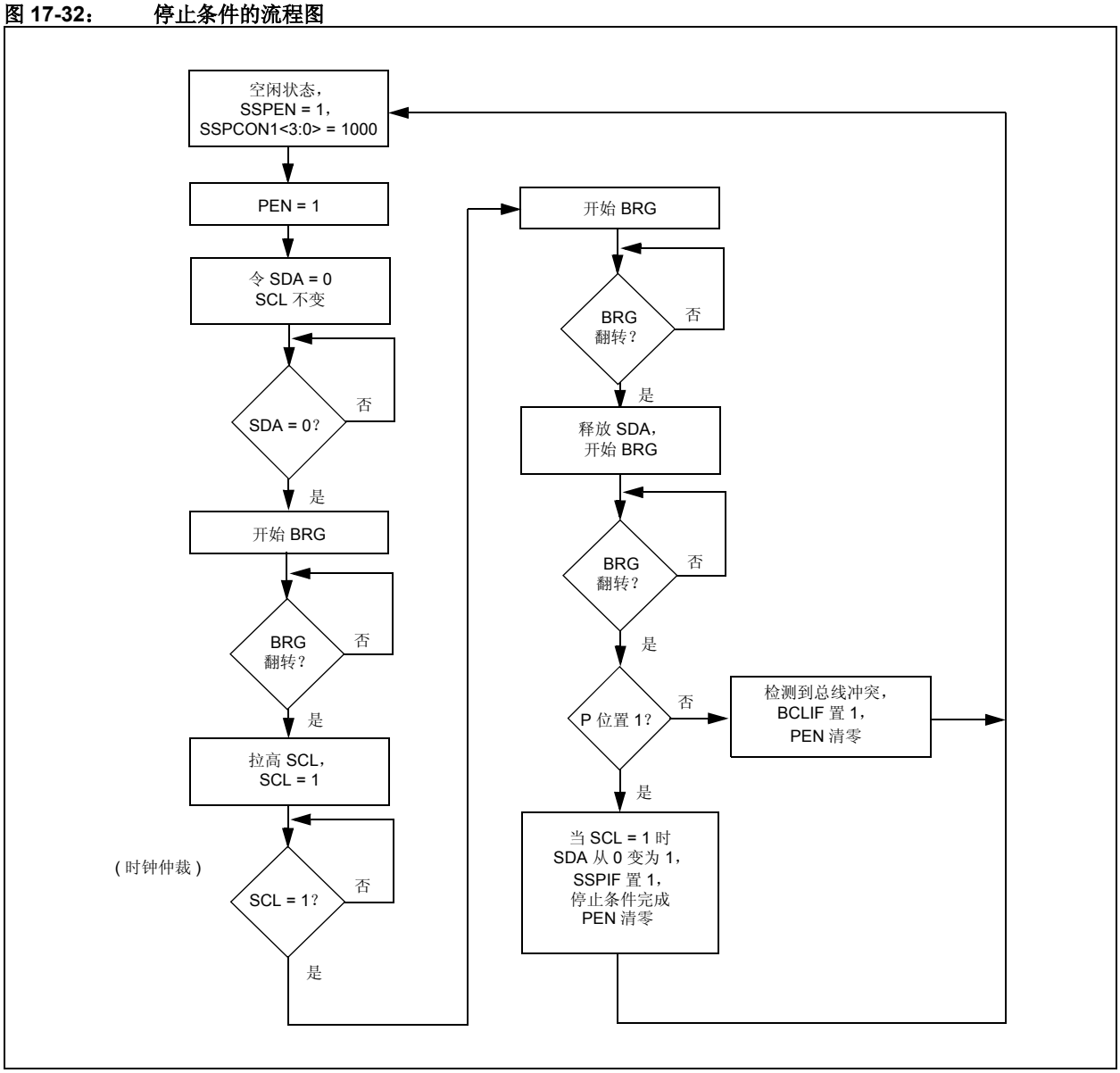


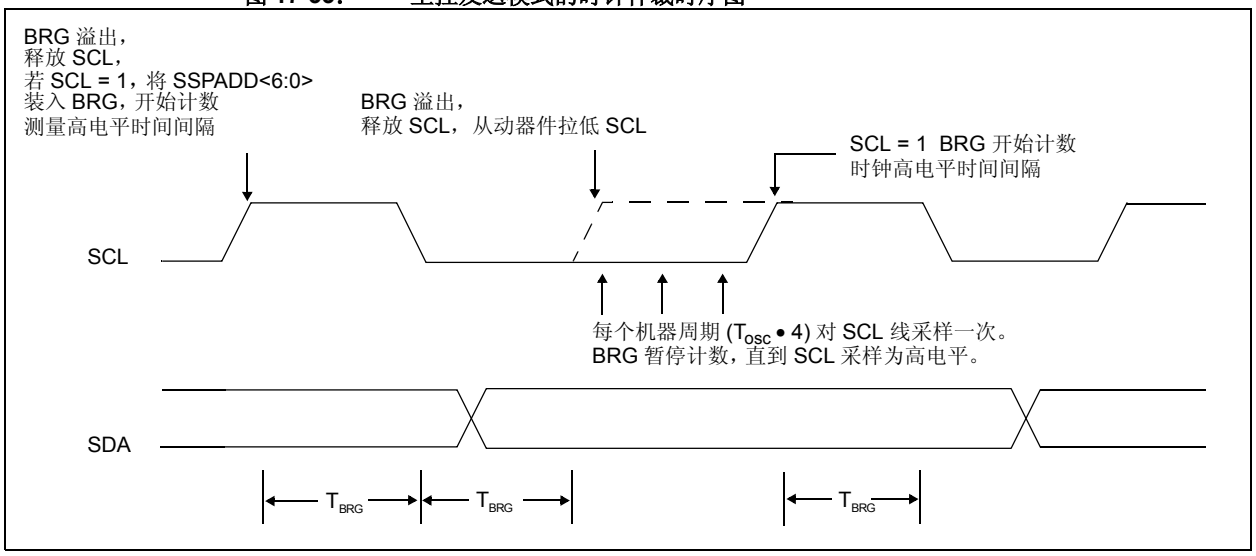
图 17-32： 停止条件的流程图



17.4.15 时钟仲裁

当主机在任何接收、发送或重复启动 / 停止条件期间把 SCL 引脚拉高 (SCL 允许悬空为高电平) 时，将发生时钟仲裁。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 引脚实际采样为高电平。当 SCL 采样为高电平时，波特率发生器被重新装入 SSPADD<6:0> 的内容并开始计数。这就保证在外部器件将时钟拉低时，SCL 始终至少保持一个 BRG 翻转计数周期 T_{BRG} 的高电平 (图 17-33)。

图 17-33： 主控发送模式的时钟仲裁时序图



17.4.16 休眠操作

在休眠方式下，I^2C 模块可以接收地址或数据。并且地址匹配或字节传输完成后，如果允许 MSSP 中断，将唤醒 CPU。

17.4.17 复位的影响

复位操作会禁止 MSSP 模块并停止当前的数据传输。

17.4.18 多主机通讯、总线冲突和总线仲裁

多主机模式支持是通过总线仲裁来实现的。当主机输出地址 / 数据到 SDA 引脚时，如果主机通过将 SDA 引脚悬空为高电平来输出 ‘1’，而另一个主机输出 ‘0’ 到 SDA 线上，就会发生总线仲裁。当 SCL 引脚悬空为高电平时，SDA 线上的数据电平应保持稳定。如果 SDA 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。主机将置位总线冲突中断标志位 BCLIF，并将 I^2C 端口复位到空闲状态 (图 17-34)。

如果在发送过程中发生总线冲突，发送将被中止，BF 标志位清零，SDA 和 SCL 被拉高，SSPBUF 可被写入。当执行完总线冲突中断服务程序之后，如果 I^2C 总线处于空闲状态，用户可通过发送一个启动信号恢复通信。

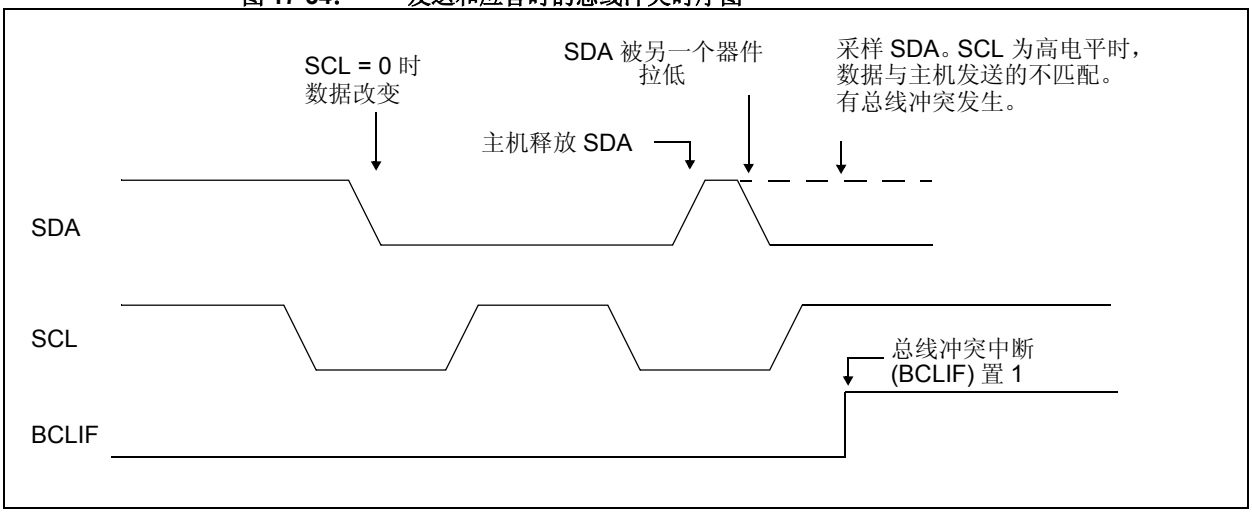
如果在启动、重复启动、停止或应答条件的执行过程中发生总线冲突，则这些条件被中止，SDA 和 SCL 被拉高，SSPCON2 寄存器中的相应控制位被清零。当执行完总线冲突中断服务程序之后，如果 I^2C 总线处于空闲状态，用户可通过发送一个启动信号恢复通信。

主机将继续监控 SDA 和 SCL 引脚，如果出现停止信号，SSPIF 位将被置 1。

无论发生总线冲突时发送的进度如何，写 SSPBUF 都会启动从第一个数据位开始发送数据。

在多主机模式下，通过检测启动和停止条件可以产生中断的功能可以判断总线何时空闲。SSPSTAT 寄存器中的 P 位置 1，或总线空闲而 S 和 P 位清零时，可以对 I^2C 总线进行控制操作。

图 17-34： 发送和应答时的总线冲突时序图



17.4.18.1 启动条件过程中的总线冲突

启动条件过程中，以下事件将导致总线冲突：

a) 在启动条件的开始， SDA 或 SCL 采样为低电平 (图 17-35)。

b) SDA 被拉低之前， SCL 采样为低电平 (图 17-36)。

在启动条件过程中，要监控 SDA 和 SCL 引脚。

如果：

SDA 引脚已经是低电平
或 SCL 引脚已经是低电平，

那么：

启动条件被中止，
BCLIF 标志置 1，
SSP 模块复位到空闲状态 (图 17-35)。

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<6:0> 的值并递减计数至 0。如果在 SDA 为高电平时，SCL 采样为低电平，则发生总线冲突，因为这表示另一台主机在启动状态期间试图发送一个数据 '1'。

如果 SDA 在该计数期间内采样为低电平，则 BRG 复位，同时 SDA 线保持原值 (图 17-37)。如果 SDA 引脚采样为 '1'， SDA 引脚将在 BRG 计数结束时被置为低电平。接着，波特率发生器被重新装入值并递减计数至 0。在这期间，如果 SCL 引脚采样为 '0'，则不会发生总线冲突。在 BRG 计数结束时，SCL 引脚被置为低电平。

注：	在启动条件期间不会发生总线冲突是因为两个主机不可能精确地在同一时刻发出启动信号，因此一个主机总在另一个主机之前将 SDA 置低。但是这一情况不会引起总线冲突，因为两个主机必须对启动条件之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 17-35： 启动条件期间的总线冲突 (仅 SDA)

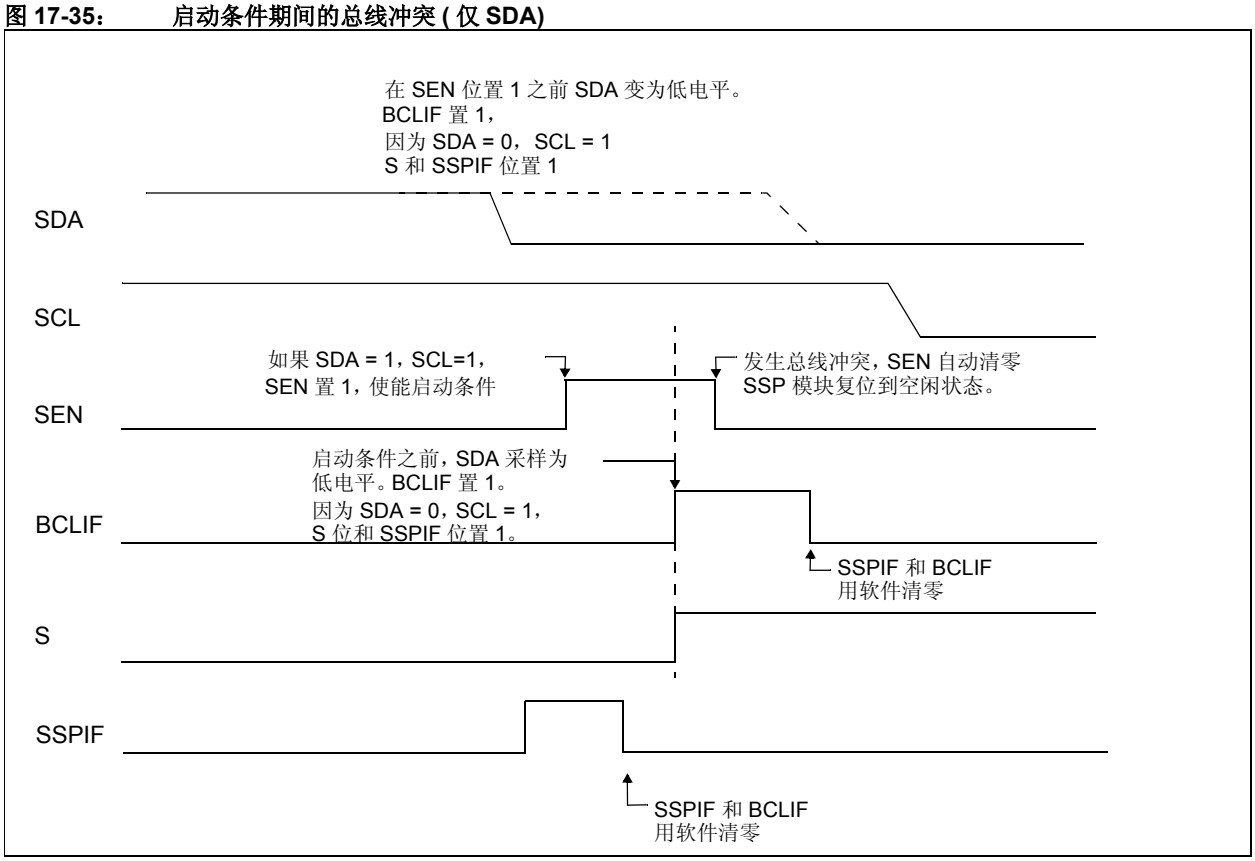


图 17-36： 启动条件期间的总线冲突 (SCL = 0)

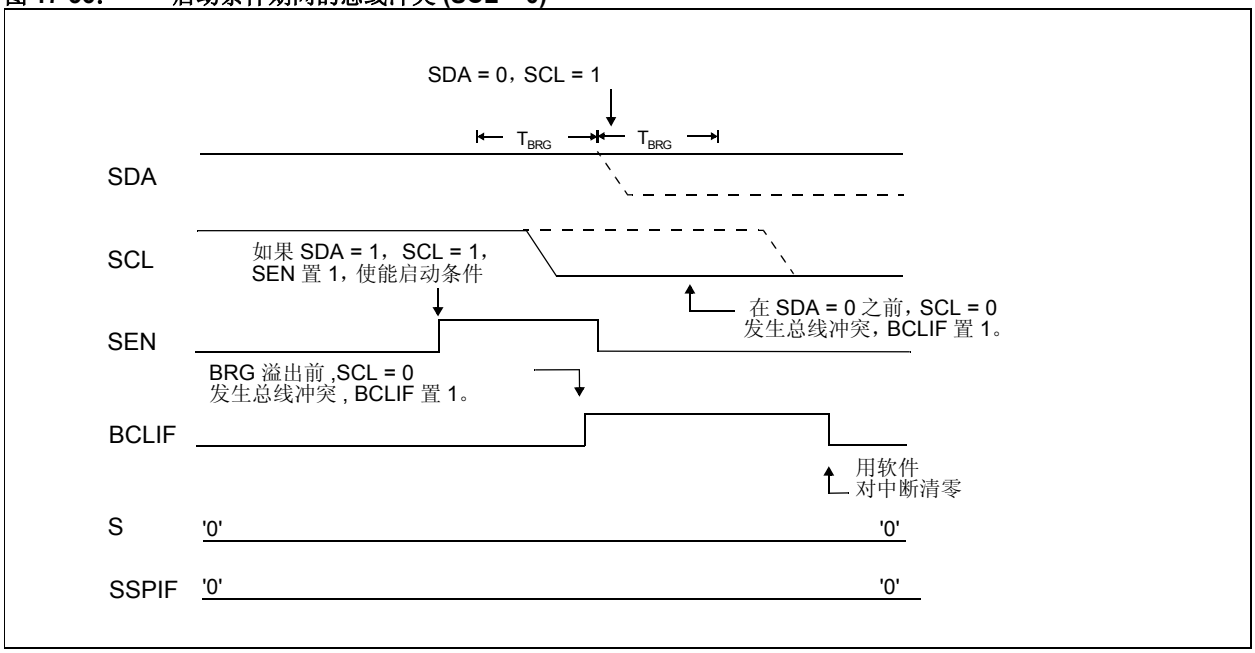
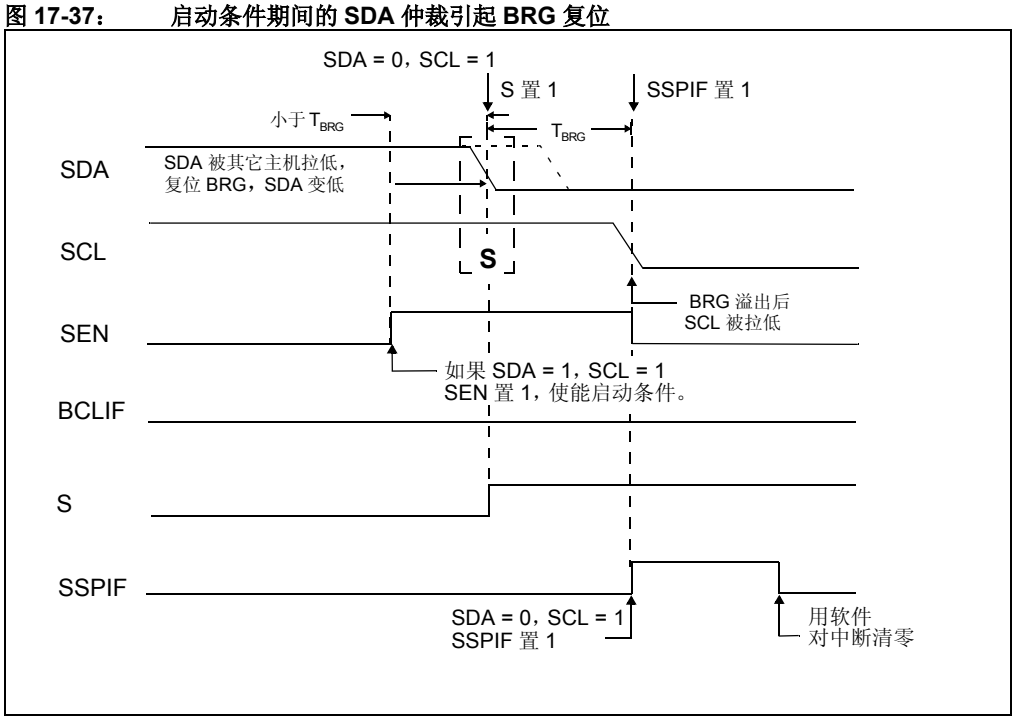


图 17-37： 启动条件期间的 SDA 仲裁引起 BRG 复位



17.4.18.2 重复启动条件过程中的总线冲突

重复启动条件期间，以下事件将导致总线冲突：

a) 在 SCL 由低电平变为高电平的过程中，在 SDA 上采样到低电平。

b) 在 SDA 被置为低电平之前， SCL 变为低电平，表示另一个主机正试图发送一个数据 '1'。

当 SDA 置高且允许该引脚悬空为高电平时，波特率发生器装入 SSPADD<6:0> 中的值并递减计数至零。接着 SCL 引脚被置为高电平，当 SCL 采样为高电平时，对 SDA 引脚进行采样。如果 SDA 为低电平，则发生总线冲突 (即另一个主机，如图 17-38，正试图发送一个数据 '0')。但是，如果 SDA 采样为高电平， 则波特率发生器被重新装入值并开始计数。如果 SDA 在 BRG 溢出之前从高电平变为低电平，则不会发生总线冲突，因为两个主机不可能精确地在同一时刻将 SDA 置低。

如果 SCL 在 BRG 溢出和 SDA 变低之前就从高电平变为低电平，那么将发生总线冲突。这种情况表示另一个主机在重复启动条件过程中正试图发送一个数据 ‘1’，如 图 17-39。

如果在 BRG 溢出结束时 SCL 和 SDA 都是高电平，那么 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 引脚状态如何， SCL 引脚都被拉低，重复启动状态结束。

图 17-38： 重复启动状态时总线冲突的时序图 (情况 1)

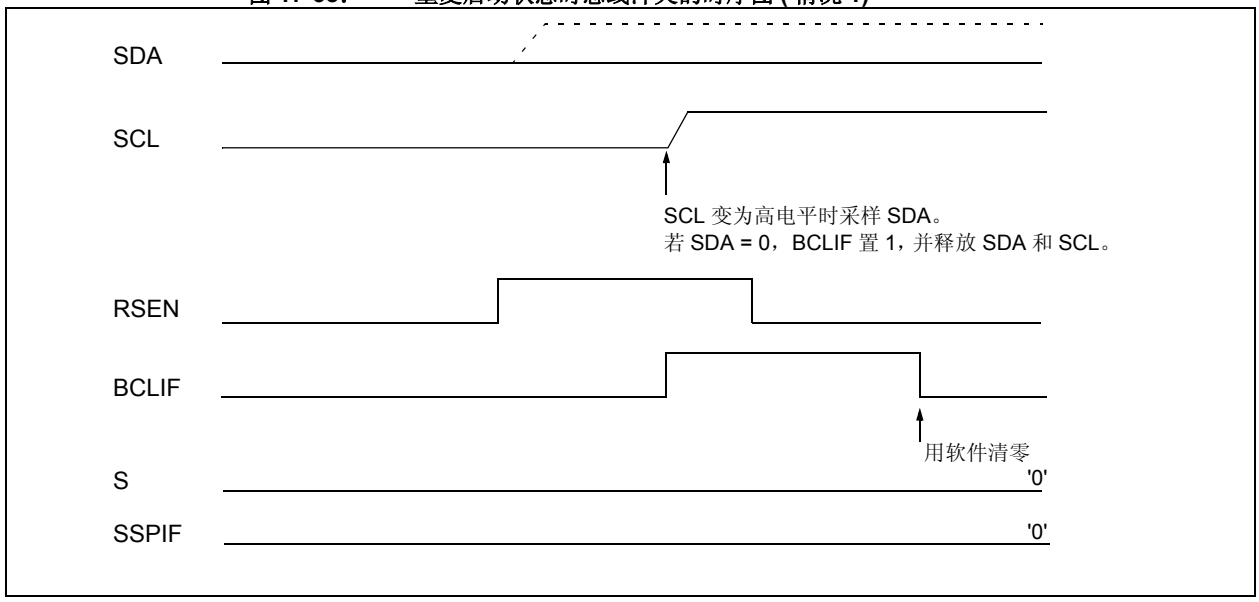
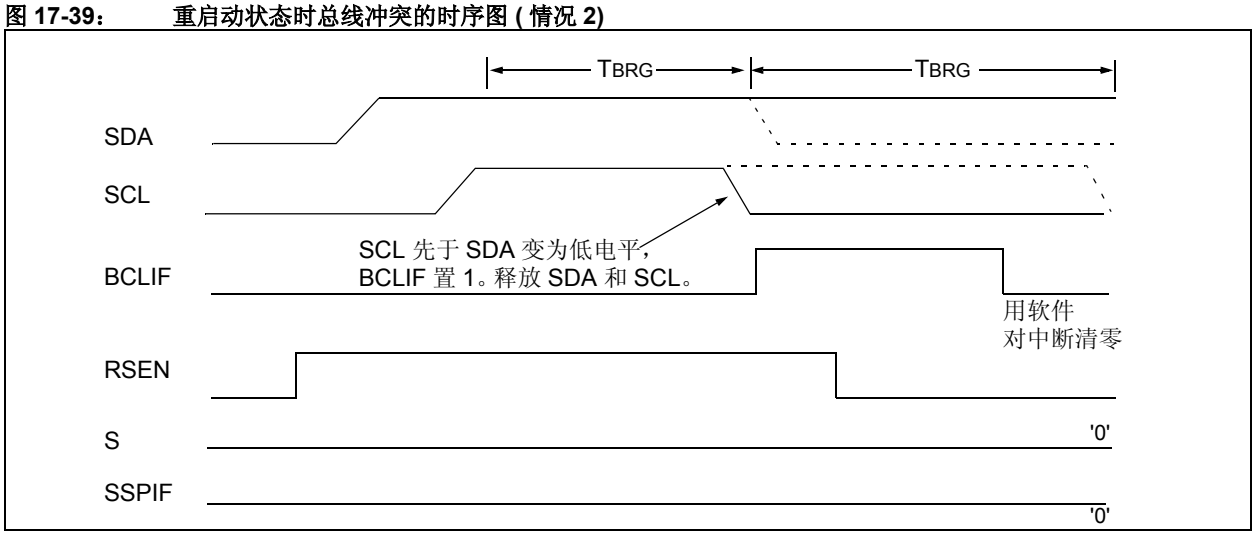


图 17-39： 重启动状态时总线冲突的时序图 (情况 2)



17.4.18.3 停止条件期间的总线冲突

停止条件期间以下事件将导致总线冲突：

a) SDA 已经被置高和允许悬空为高电平之后， SDA 在 BRG 溢出之后被采样到低电平。

b) SCL 引脚被置高之后， SCL 在 SDA 变成高电平之前被采样到低电平。

停止条件从 SDA 被置为低电平时开始。当 SDA 采样为低电平时，SCL 引脚允许悬空。当引脚被采样为高电平时 (时钟仲裁)，波特率发生器装入 SSPADD<6:0> 的值并递减计数到 0。在 BRG 溢出后，对 SDA 采样。如果 SDA 采样为低电平，则已发生总线冲突。这是因为另一个主机正试图发送一个数据 '0' (图 17-40)。如果 SCL 引脚在允许 SDA 悬空为高电平前采样为低电平，也会发生总线冲突。 这是另一个主机正试图发送一个数据 '0' 的另一种情况 (图 17-41)。

图 17-40： 停止条件期间的总线冲突 (情况 1)

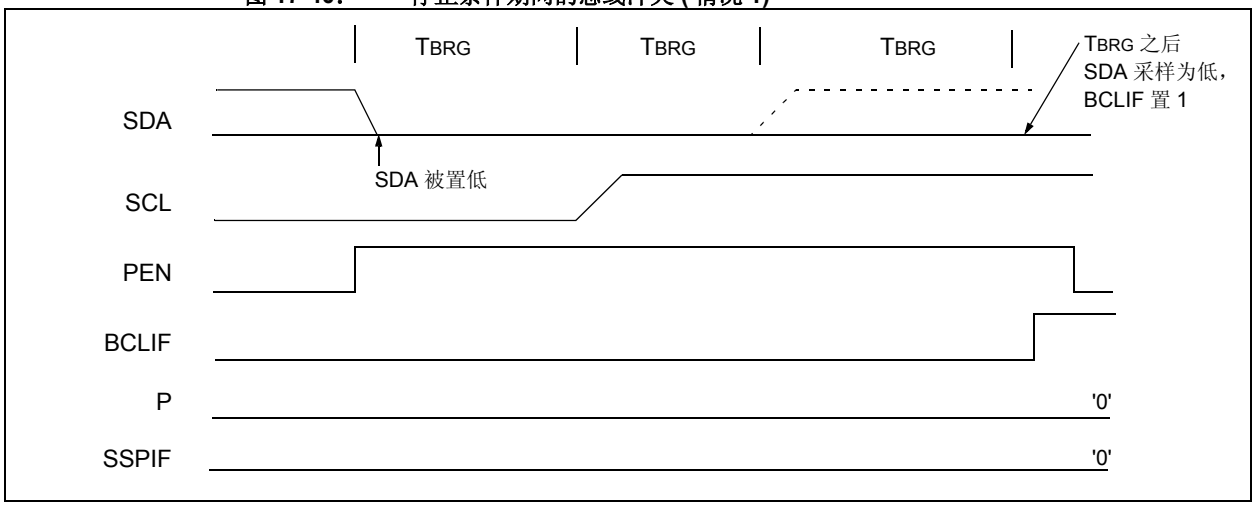
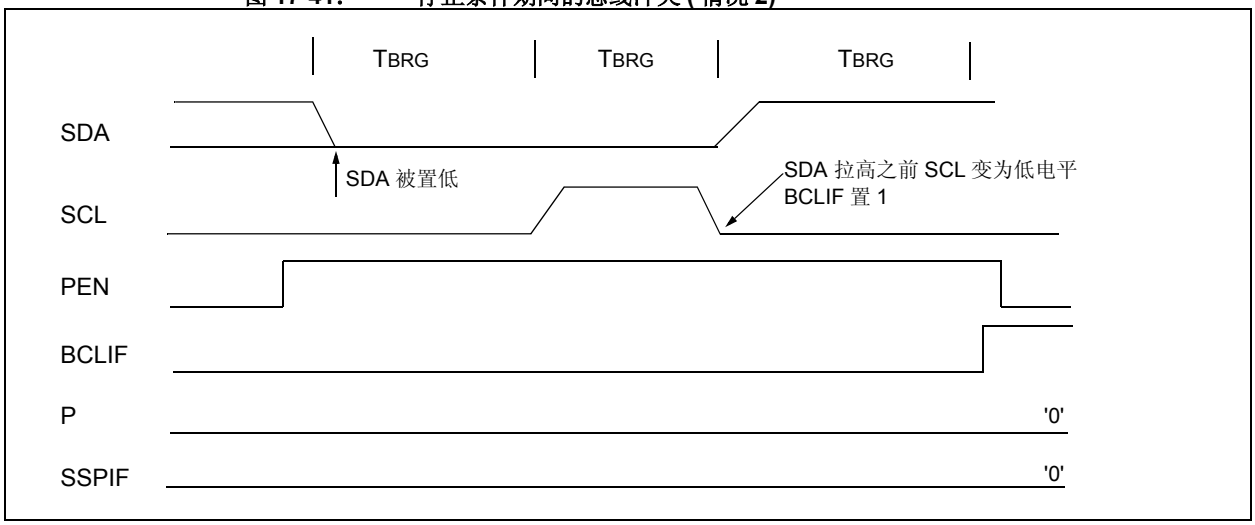


图 17-41： 停止条件期间的总线冲突 (情况 2)



17.5 I^2C^{TM} 总线的连接注意事项

对于标准模式的 I^2C 总线器件，图 17-42 中电阻 ***RP*** 和 ***RS*** 的取值取决于以下参数：

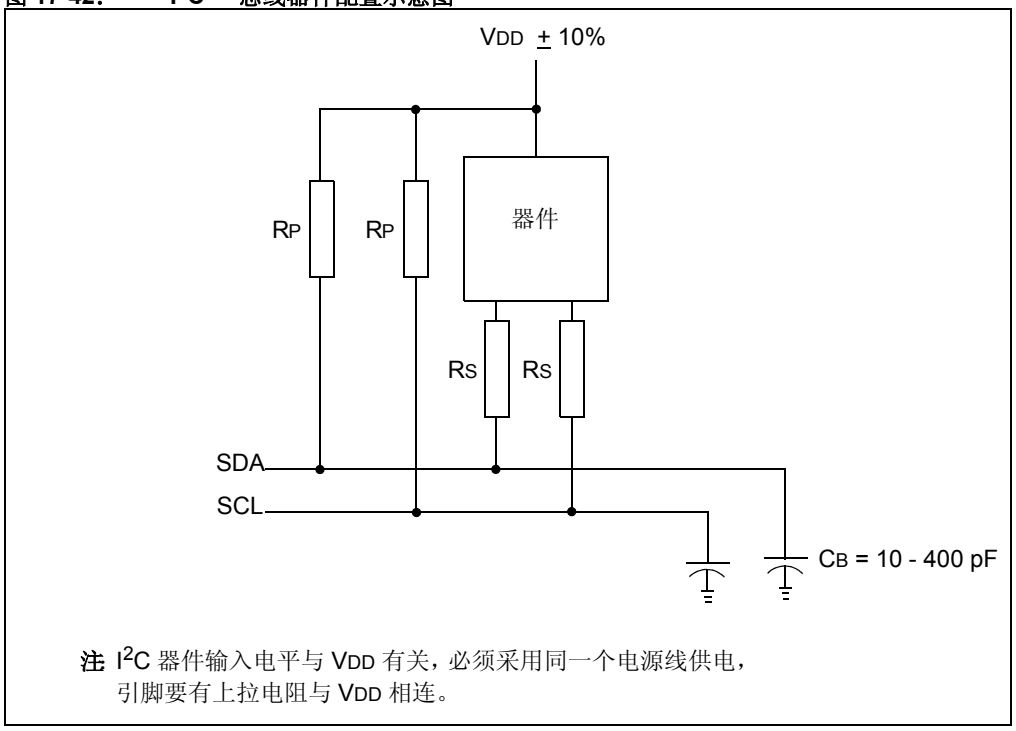
- 电源电压
- 总线容量
- 所连接器件数 (输入电流 + 漏电流)

电源电压限制了电阻 ***RP*** 的最小值，这是因为对于所指定的输出级，在 VOLMAX = 0.4V 时，所规定的最小吸入电流为 3 mA。例如，电源电压 VDD = 5V±10% 时，在 3 mA 时 VOLMAX = 0.4V，RPMIN = (5.5-0.4)/0.003 = 1.7 kΩ。VDD 为 ***RP*** 的函数，见图 17-42。对于低电平，期望的噪声极限为 0.1VDD，这限制了 ***RS*** 的最大值。可以选择串联电阻来提高抗 ESD 的能力。

总线电容是指线路、连接和引脚的总电容。由于规定的上升时间，总线电容限制了 ***RP*** 的最大值 (图 17-42)。

SMP 位是压摆率控制使能位。该位在 SSPSTAT 寄存器中，用于控制 I^2C 模式下 (主控或从动模式)I/O 引脚的压摆率。

图 17-42： I^2C^{TM} 总线器件配置示意图



注： I^2C 器件输入电平与 VDD 有关，必须采用同一个电源线供电，引脚要有上拉电阻与 VDD 相连。

17.6 初始化

例 17-2： SPI™ 主控模式初始化

```
    CLRF    STATUS        ; Bank 0
    CLRF    SSPSTAT       ; SMP = 0, CKE = 0, and clear status bits
    BSF     SSPSTAT, CKE  ; CKE = 1
    MOVLW   0x31          ; Set up SPI port, Master mode, CLK/16,
    MOVWF   SSPCON        ;   Data xmit on falling edge (CKE=1 & CKP=1)
                          ;   Data sampled in middle (SMP=0 & Master mode)
    BSF     STATUS, RP0   ; Bank 1
    BSF     PIE, SSPIE    ; Enable SSP interrupt
    BCF     STATUS, RP0   ; Bank 0
    BSF     INTCON, GIE   ; Enable, enabled interrupts
    MOVLW   DataByte      ; Data to be Transmitted
                          ;   Could move data from RAM location
    MOVWF   SSPBUF        ; Start Transmission
```

17.6.1 主 SSP 模块 / 基本 SSP 模块的兼容性

当从基本 SSP 模块的 SPI 升级到主 SSP 模块时，SSPSTAT 寄存器还另外包含两个控制位，它们是：

- SMP， SPI 输入数据采样控制位
- CKE， SPI 时钟边沿选择位

为了使主 SSP 模块与基本 SSP 模块的 SPI 通信模式相兼容，这些位必须合理设置。如果不按表 17-4 进行设置，可能会发生 SPI 通信错误。

表 17-4： 保持兼容的位状态设置

基本 SSP 模块	主 SSP 模块		
CKP	CKP	CKE	SMP
1	1	0	0
0	0	0	0

17.7 设计技巧

问 1: *在 SPI 模式下时，为什么无法和 SPI™ 器件通信？*

答 1:

确认你使用了该器件支持的正确 SPI 模式。该 SPI 支持所有 4 种 SPI 模式。为使其正常工作，应先检查时钟的极性和相位。

问 2: *使用 I²C 模式时，为什么写入 SSPBUF 寄存器的值不能传送？*

答 2:

确认将 CKP 位置 1 以释放 I²C 时钟。

17.8 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但是其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与主同步串行口模块相关的应用笔记有：

标题	应用笔记 #
Use of the SSP Module in the I $^2C^{TM}$ Multi-Master Environment	AN578
Using Microchip 93 Series Serial EEPROMs with Microcontroller SPI^{TM} Ports	AN613
Interfacing PIC16C64/74 to Microchip SPI^{TM} Serial EEPROM	AN647
Interfacing a Microchip PIC16C92x to Microchip SPI^{TM} Serial EEPROM	AN668

17.9 版本历史

版本 A

这是描述主 SSP 模块的初始发行版。

第 18 章 USART

目录

本章包括下面一些主要内容：

18.1 简介 18-2
18.2 控制寄存器 18-3
18.3 USART 波特率发生器 （BRG） 18-5
18.4 USART 异步工作模式 18-8
18.5 USART 同步主控模式 18-15
18.6 USART 同步从动模式 18-19
18.7 初始化 18-21
18.8 设计技巧 18-22
18.9 相关应用笔记 18-23
18.10 版本历史 18-24

18.1 简介

通用同步 / 异步收发器（USART）模块是两个串行 I/O 模块之一（另一个是 SSP 模块）。USART 也称为串行通信接口或 SCI。USART 可以配置为全双工异步系统，可与 CRT 终端和个人计算机等外设进行通信；也可配置为半双工同步系统，可与 A/D 或 D/A 集成电路，以及串行 EEPROM 等外设器件进行通信。

USART 可配置为以下几种工作模式：

- 全双工异步模式
- 半双工同步主控模式
- 半双工同步从动模式

为将 TX/CK 和 RX/DT 引脚配置为用于 USART，必须把 SPEN 位（RCSTA<7>）和相应的 TRIS 位置 1。

18.2 控制寄存器

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	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC**：时钟源选择位
异步模式
此位未用

同步模式
1 = 主控模式（由内部波特率发生器产生时钟）
0 = 从动模式（由外部时钟源提供时钟信号）

bit 6 **TX9**：9 位发送使能位
1 = 选择 9 位数据发送
0 = 选择 8 位数据发送

bit 5 **TXEN**：发送使能位
1 = 允许发送
0 = 禁止发送

注： 在 SYNC 模式下， SREN/CREN 位比 TXEN 位优先级高。

bit 4 **SYNC**：USART 模式选择位
1 = 同步模式
0 = 异步模式

bit 3 **未用位**：读为 0

bit 2 **BRGH**：高速波特率使能位。
异步模式
1 = 高速
0 = 低速

同步模式
此位未用

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，可作为奇偶校验位。

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

寄存器 18-2： RCSTA：接收状态和控制寄存器

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R-0	R-0	R-0
SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN**：串口使能位
1 = 允许串口工作（把 RX/DT 和 TX/CK 引脚配置为串口引脚 ）
0 = 禁止串口工作

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
此位未用

同步主控模式
1 = 允许接收单字节
0 = 禁止接收单字节
在接收完成后该位被清零。

同步从动模式
此位未用

bit 4 **CREN**：连续接收使能位
异步模式
1 = 允许连续接收
0 = 禁止连续接收

同步模式
1 = 允许连续接收直到 CREN 位被清零（CREN 位比 SREN 位优先级高 ）
0 = 禁止连续接收

bit 3 **未用位**：读为 0

bit 2 **FERR**：帧出错标志位
1 = 帧出错 （读 RCREG 寄存器可更新该位，并接收下一个有效字节）
0 = 无帧错误。

bit 1 **OERR**：溢出错误位
1 = 有溢出错误 （清零 CREN 位可将此位清零）
0 = 无溢出错误。

bit 0 **RX9D**：接收数据的第 9 位，可作为奇偶校验位。

其中：		
R = 可读位	W = 可写位	
U = 未用位，读为 ‘0’		- n = 上电复位时的值

18.3 USART 波特率发生器 （BRG）

USART 有一个专用的 8 位波特率发生器，它支持 USART 的同步模式和异步模式。SPBRG 寄存器控制着独立的 8 位定时器的周期。在异步方式下， BRGH（TXSTA<2>）位也用来控制波特率。在同步模式下不使用 BRGH。表 18-1 是在主控方式 （内部时钟）下，不同 USART 工作模式的波特率的计算公式。

给出目标波特率值和时钟频率 Fosc，就可用表 18-1 中的公式计算出 SPBRG 寄存器的最接近的整数值，表中 X 表示 SPBRG 寄存器中的值（0 到 255）。由此可计算出波特率的误差。

表 18-1： 波特率计算公式

SYNC	BRGH = 0 （低速）	BRGH = 1（高速）
0	（异步）波特率 = FOSC/（64(X+1))	波特率 = FOSC/(16(X+1))
1	（同步）波特率 = FOSC/(4(X+1))	无

X 为 SPBRG 寄存器中的值（0 到 255）

例 18-1 给出了在下列情况下波特率误差的计算：

FOSC = 16 MHz
目标波特率 = 9600
BRGH = 0
SYNC = 0

例 18-1： 计算波特率误差

目标波特率 = Fosc / (64 (X + 1))
9600 = 16000000 / (64 (X + 1))
X = ⌊25.042⌋ = 25

计算波特率 = 16000000 / (64 (25 + 1))
= 9615

误差 = (计算波特率 - 目标波特率) / 目标波特率
= (9615 - 9600) / 9600
= 0.16%

即使对于低波特率时钟，使用高波特率公式 （BRGH = 1）也是有利的，因为公式 FOSC / (16(X + 1)) 在某些情况下可以减小波特率误差。

向波特率寄存器 SPBRG 写入一个新值会使 BRG 定时器复位 （或清零），这可确保波特率发生器 BRG 不需要等到定时器溢出就可以输出新的波特率值。

表 18-2： 与波特率发生器有关的寄存器

寄存器	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位和欠压复位时的值	其它复位时的值
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
SPBRG	波特率发生器寄存器								0000 0000	0000 0000

其中： x = 未知， - = 未用位，读为 ‘0’， 阴影部分 BRG 未使用。

表 18-3： 同步模式下的波特率

目标波特率 (Kbps)	Fosc = 20MHz			16MHz			10MHz			7.15909MHz		
	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	NA	-	-	NA	-	-	NA	-	-	NA	-	-
2.4	NA	-	-	NA	-	-	NA	-	-	NA	-	-
9.6	NA	-	-	NA	-	-	9.766	+1.73	255	9.622	+0.23	185
19.2	19.53	+1.73	255	19.23	+0.16	207	19.23	+0.16	129	19.24	+0.23	92
76.8	76.92	+0.16	64	76.92	+0.16	51	75.76	-1.36	32	77.82	+1.32	22
96	96.15	+0.16	51	95.24	-0.79	41	96.15	+0.16	25	94.20	-1.88	18
300	294.1	-1.96	16	307.69	+2.56	12	312.5	+4.17	7	298.3	-0.57	5
500	500	0	9	500	0	7	500	0	4	无	-	-
高	5000	-	0	4000	-	0	2500	-	0	1789.8	-	0
低	19.53	-	255	15.625	-	255	9.766	-	255	6.991	-	255

目标波特率 (Kbps)	Fosc = 5.0688MHz			4MHz			3.579545MHz			1MHz			32.768kHz		
	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)	计算波特率 (Kbps)	误差 (%)	SPBRG 值 (十进制)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-	0.303	+1.14	26
1.2	NA	-	-	NA	-	-	NA	-	-	1.202	+0.16	207	1.170	-2.48	6
2.4	NA	-	-	NA	-	-	NA	-	-	2.404	+0.16	103	NA	-	-
9.6	9.6	0	131	9.615	+0.16	103	9.622	+0.23	92	9.615	+0.16	25	NA	-	-
19.2	19.2	0	65	19.231	+0.16	51	19.04	-0.83	46	19.24	+0.16	12	NA	-	-
76.8	79.2	+3.13	15	76.923	+0.16	12	74.57	-2.90	11	83.34	+8.51	2	NA	-	-
96	97.48	+1.54	12	1000	+4.17	9	99.43	+3.57	8	NA	-	-	NA	-	-
300	316.8	+5.60	3	NA	-	-	298.3	-0.57	2	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
高	1267	-	0	100	-	0	894.9	-	0	250	-	0	8.192	-	0
低	4.950	-	255	3.906	-	255	3.496	-	255	0.9766	-	255	0.032	-	255

表 18-4： 异步模式下的波特率 （BRGH = 0）

目标波特率 (Kbps)	FOSC = 20 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	16 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	10 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	7.15909 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)
0.3	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1.2	1.221	+1.73	255	1.202	+0.16	207	1.202	+0.16	129	1.203	+0.23	92
2.4	2.404	+0.16	129	2.404	+0.16	103	2.404	+0.16	64	2.380	-0.83	46
9.6	9.469	-1.36	32	9.615	+0.16	25	9.766	+1.73	15	9.322	-2.90	11
19.2	19.53	+1.73	15	19.23	+0.16	12	19.53	+1.73	7	18.64	-2.90	5
76.8	78.13	+1.73	3	83.33	+8.51	2	78.13	+1.73	1	NA	-	-
96	104.2	+8.51	2	NA	-	-	NA	-	-	NA	-	-
300	312.5	+4.17	0	NA	-	-	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-
高	312.5	-	0	250	-	0	156.3	-	0	111.9	-	0
低	1.221	-	255	0.977	-	255	0.6104	-	255	0.437	-	255

目标波特率 (Kbps)	FOSC = 5.0688 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	4 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	3.579545 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	1 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	32.768 kHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)
0.3	0.31	+3.13	255	0.3005	-0.17	207	0.301	+0.23	185	0.300	+0.16	51	0.256	-14.67	1
1.2	1.2	0	65	1.202	+1.67	51	1.190	-0.83	46	1.202	+0.16	12	NA	-	-
2.4	2.4	0	32	2.404	+1.67	25	2.432	+1.32	22	2.232	-6.99	6	NA	-	-
9.6	9.9	+3.13	7	NA	-	-	9.322	-2.90	5	NA	-	-	NA	-	-
19.2	19.8	+3.13	3	NA	-	-	18.64	-2.90	2	NA	-	-	NA	-	-
76.8	79.2	+3.13	0	NA	-	-	NA	-	-	NA	-	-	NA	-	-
96	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
300	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
500	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
高	79.2	-	0	62.500	-	0	55.93	-	0	15.63	-	0	0.512	-	0
低	0.3094	-	255	3.906	-	255	0.2185	-	255	0.0610	-	255	0.0020	-	255

表 18-5： 异步模式下的波特率 （BRGH = 1）

目标波特率 (Kbps)	FOSC = 20 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	16 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	10 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	7.15909 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)
9.6	9.615	+0.16	129	9.615	+0.16	103	9.615	+0.16	64	9.520	-0.83	46
19.2	19.230	+0.16	64	19.230	+0.16	51	18.939	-1.36	32	19.454	+1.32	22
38.4	37.878	-1.36	32	38.461	+0.16	25	39.062	+1.7	15	37.286	-2.90	11
57.6	56.818	-1.36	21	58.823	+2.12	16	56.818	-1.36	10	55.930	-2.90	7
115.2	113.636	-1.36	10	111.111	-3.55	8	125	+8.51	4	111.860	-2.90	3
250	250	0	4	250	0	3	NA	-	-	NA	-	-
625	625	0	1	NA	-	-	625	0	0	NA	-	-
1250	1250	0	0	NA	-	-	NA	-	-	NA	-	-

目标波特率 (Kbps)	FOSC = 5.0688 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	4 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	3.579545 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	1 MHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)	32.768 kHz 计算波特率 (Kbps)	误差 （%）	SPBRG 值 (十进制)
9.6	9.6	0	32	NA	-	-	9.727	+1.32	22	8.928	-6.99	6	NA	-	-
19.2	18.645	-2.94	16	1.202	+0.17	207	18.643	-2.90	11	20.833	+8.51	2	NA	-	-
38.4	39.6	+3.12	7	2.403	+0.13	103	37.286	-2.90	5	31.25	-18.61	1	NA	-	-
57.6	52.8	-8.33	5	9.615	+0.16	25	55.930	-2.90	3	62.5	+8.51	0	NA	-	-
115.2	105.6	-8.33	2	19.231	+0.16	12	111.860	-2.90	1	NA	-	-	NA	-	-
250	NA	-	-	NA	-	-	223.721	-10.51	0	NA	-	-	NA	-	-
625	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-
1250	NA	-	-	NA	-	-	NA	-	-	NA	-	-	NA	-	-

18.4 USART 异步工作模式

在异步工作模式下，USART 采用的是标准非归零（NRZ）编码格式（一位起始位、8 位或 9 位数据位和一位停止位）。最常用的数据格式是 8 位。片内专用的 8 位波特率发生器可用于由振荡器产生标准的波特率频率。USART 首先发送和接收最低有效位。USART 的发送器和接收器在功能上是独立的，但采用相同的数据格式和波特率。波特率发生器可以根据 BRGH 位（TXSTA<2>）的状态产生两种不同的移位速率：对系统时钟 16 分频或 64 分频的波特率时钟。 USART 硬件不支持奇偶校验，但可以用软件实现（奇偶校验位是第 9 个数据位）。在休眠状态下， USART 不能在异步模式下工作。

通过对 SYNC 位（TXSTA<4>）清零，可选择 USART 异步工作模式。

USART 异步工作模式包括如下重要组成部分：

- 波特率发生器
- 采样电路
- 异步发送器
- 异步接收器

18.4.1 USART 异步发送器

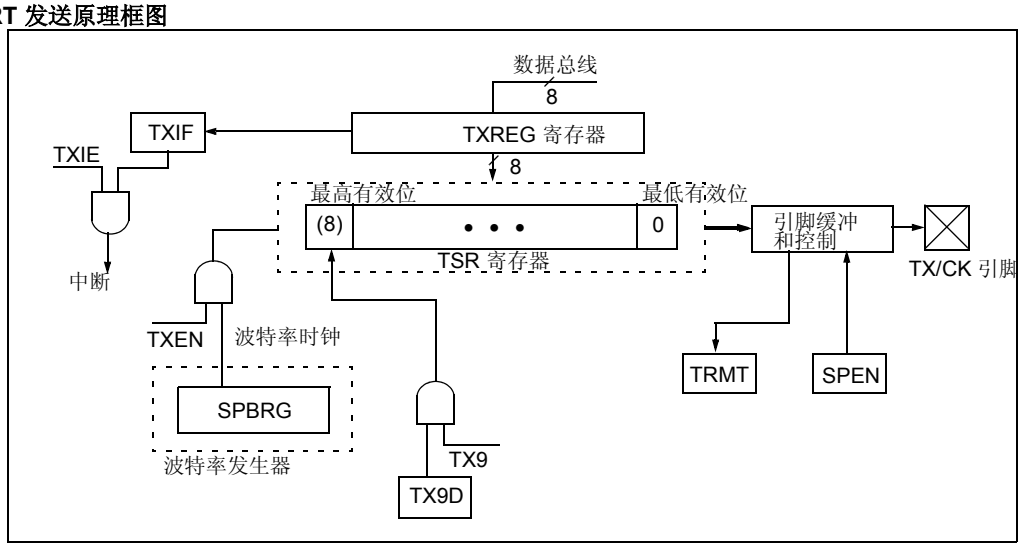
图 18-1 所示为 USART 发送器的原理框图。发送器的核心是发送（串行）移位寄存器（TSR）。发送移位寄存器从读 / 写发送缓冲器 TXREG 获得要发送的数据。 TXREG 寄存器由软件装入数据。前一次装入的停止位发送出去后，TSR 寄存器才装入数据。停止位一发送出去，TSR 就装入 TXREG 中的新数据（如果有的话）。一旦把 TXREG 中的数据送入 TSR 寄存器（在一个 TCY 周期内），则 TXREG 寄存器就变为空，同时发送中断标志位 TXIF 被置 1。可以通过置 1 或清零 TXIE 使能位来允许或屏蔽这个中断。不管 TXIE 位的状态如何，中断标志位 TXIF 都被置 1，且不能用软件清零。只有当新的数据装入 TXREG 寄存器时，TXIF 位才能被复位。用 TXIE 标志位表示 TXREG 寄存器的状态，而用另一位 TRMT（TXSTA<1>）表示 TSR 寄存器的状态。 TRMT 状态位是一个只读位，当 TSR 寄存器为空时被置 1。TRMT 位与任何中断逻辑都没有联系，所以要确定 TSR 寄存器是否为空，用户就必须对这一位进行查询。

注 1:	TSR 寄存器并未映射到数据存储器中，因此用户程序不能直接访问它。
注 2:	当 TXEN 位置 1 时， TXIF 标志位也会被置 1，这是因为发送缓冲器还未满 （还可以发送数据到 TXREG 寄存器）。

将 TXEN 位（TXSTA<5>）置 1 就可以使能发送，但是只有在 TXREG 寄存器装入数据和波特率发生器（BRG）产生移位时钟之后才能实际进行数据发送（图 18-1)。也可以先给 TXREG 寄存器装入数据，再将 TXEN 位置 1 来启动发送。通常，当发送先开始而 TSR 寄存器为空时，发送到 TXREG 寄存器的数据就立即被传送到 TSR，从而导致 TXREG 寄存器为空，因此可以实现连续发送 （图 18-3）。在发送过程中将 TXEN 位清零会导致发送终止，并复位发送器，同时 TX/CK 引脚会恢复到高阻状态。

为选择 9 位数据发送方式，应将发送位 TX9 位（TXSTA<6>）置 1，并且第 9 位应写入 TX9D 位（TXSTA<0>）。应先写第 9 位数据，然后将低 8 位数据写入 TXREG 寄存器，因为如果 TSR 寄存器为空，向 TXREG 寄存器写数据会导致数据立即送入 TSR 寄存器。在这种情况下，装入 TSR 寄存器的第 9 位数据可能是错误的。

图 18-1：　USART 发送原理框图



设置异步发送模式应遵循以下一些步骤：

1. 选择合适的波特率，对 SPBRG 寄存器进行初始化。如果需要高速波特率，将 BRGH 位置 1；（参见 **18.3 “USART 波特率发生器（BRG）”小节**）
2. 将 SYNC 位清零、 SPEN 位置 1，使能异步串行端口；
3. 若需要中断，将 TXIE、 GIE 和 PEIE 位置 1；
4. 若需要发送 9 位数据，将 TX9 位置 1；
5. 将 TXEN 位置 1，使能发送，这也将置位 TXIF 位；
6. 若选择发送 9 位数据，第 9 位数据应该先写入 TX9D 位；
7. 把数据送入 TXREG 寄存器（启动发送）。

图 18-2：　异步主控发送时序图

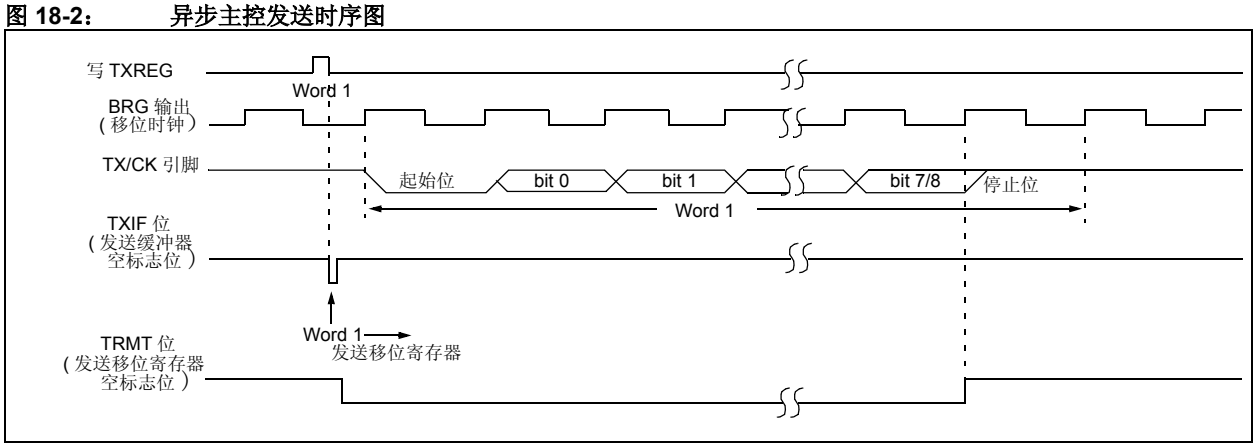
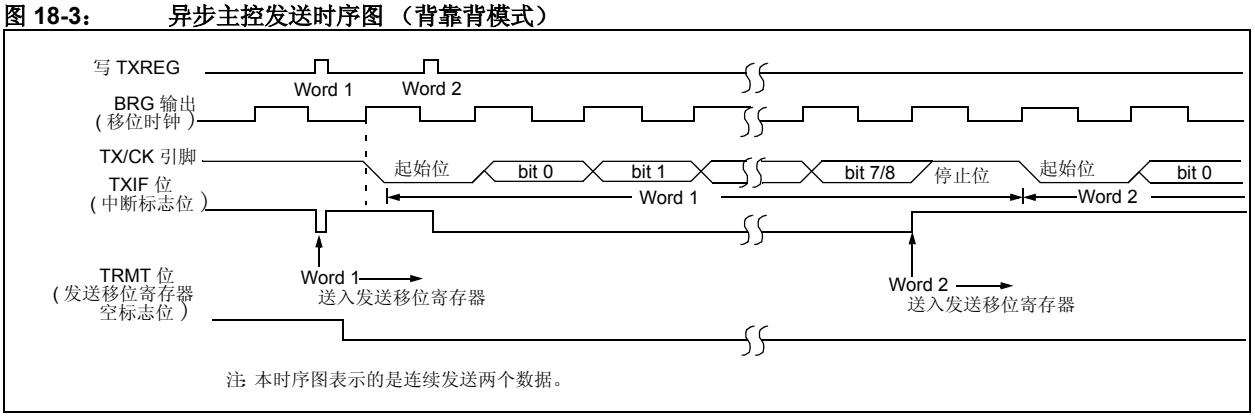


图 18-3： 异步主控发送时序图 （背靠背模式）



注: 本时序图表示的是连续发送两个数据。

表 18-6： 与异步发送有关的寄存器

寄存器	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位、欠压复位时的值	其它复位时的值
PIR	TXIF (1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
TXREG	TX7	TX6	TX5	TX4	TX3	TX2	TX1	TX0	0000 0000	0000 0000
PIE	TXIE (1)								0	0
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRG	波特率发生器寄存器								0000 0000	0000 0000

其中： x = 未知， - = 未用位，读为 ‘0’。
阴影部分异步发送未使用。
注 1：该位的位置和器件的具体型号有关。

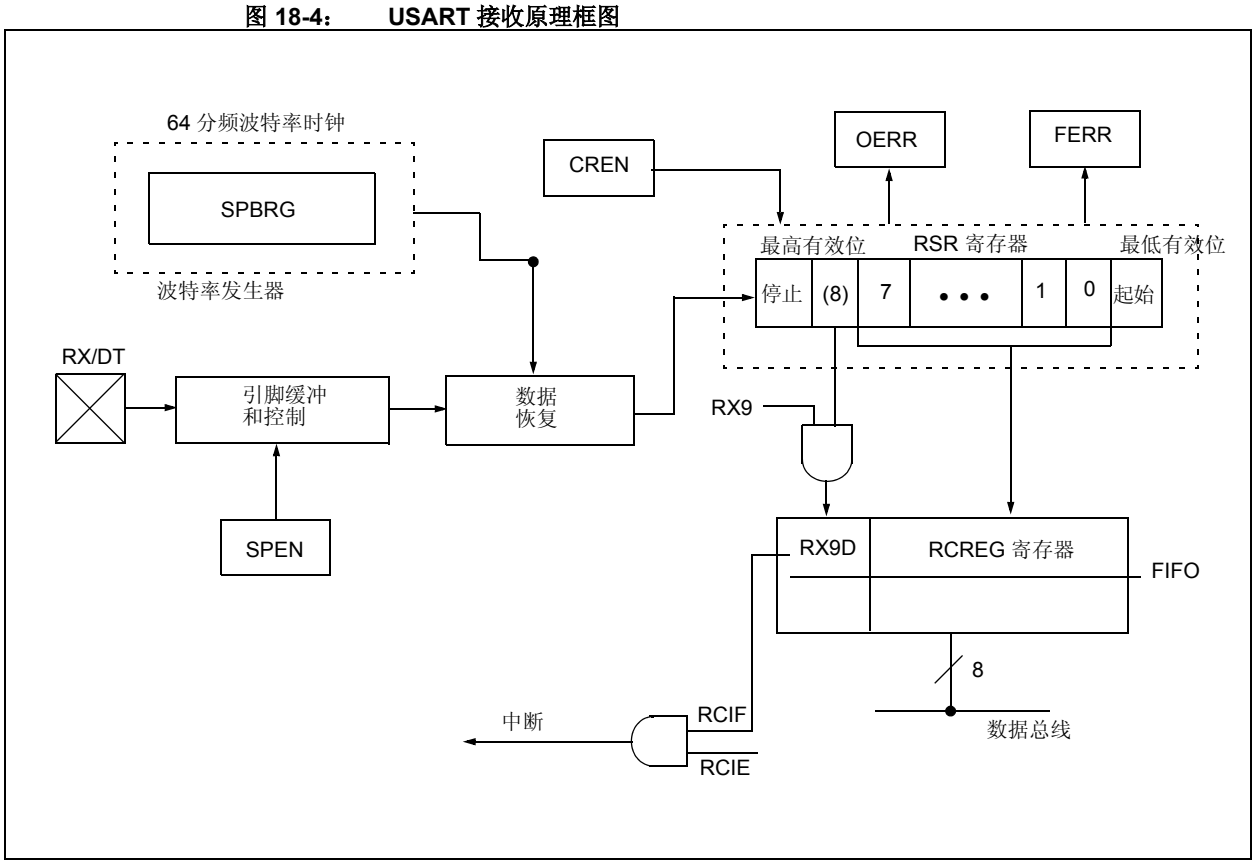
18.4.2 USART 异步接收器

图 18-4 是接收器的原理框图。在 RX/DT 引脚上接收数据，并驱动数据恢复模块。数据恢复模块实际上是一个以 16 倍波特率工作的高速移位寄存器，而主接收串行移位寄存器以位速率或 Fosc 频率工作。

选择异步模式后，将 CREN 位（RCSTA<4>）置 1 使能异步接收。

接收器的核心部件是接收（串行）移位寄存器（RSR）。在 RX/TX 引脚上采样到停止位之后，RSR 中接收到的数据被送到 RCREG 寄存器（如果 RCREG 寄存器为空）。数据传送完后，RCIF 标志位被置 1。将 RCIE 位置 1（或清零）可允许（或屏蔽）接收中断。RCIF 标志位是只读位，由硬件清零，它在 RCREG 寄存器被读之后或 RCREG 寄存器为空时被硬件清零。RCREG 寄存器是一个双缓冲寄存器（即两级深度的 FIFO），因此可以实现接收两个字节的数据并传送到 RCREG FIFO，然后第三个字节开始移位到 RSR 寄存器。在检测到第三个字节的停止位后，如果 RCREG FIFO 仍然是满的，则溢出错误标志位 OERR（RCSTA<1>）会被置 1，RSR 寄存器中的数据被丢失。可以对 RCREG 寄存器读两次重新获得 FIFO 中的两个字节。OERR 位必须由软件清零，这可以通过复位接收逻辑（将 CREN 位清零后再置 1）实现。如果 OERR 位被置 1，则禁止将 RSR 中的数据传送到 RCREG 寄存器，因此如果 OERR 位被置 1，必须将它清零。如果停止位检测为零电平，帧出错标志位 FERR（RCSTA<2>）将被置 1。FERR 位和接收到的第 9 位数据以和接收数据同样的方式被缓冲。读 RCREG 寄存器将会给 RX9D 和 FERR 位装入新值，因此为了不丢失 FERR 和 RX9D 位原来的信息，用户必须在读 RCREG 寄存器之前读 RCSTA 寄存器。

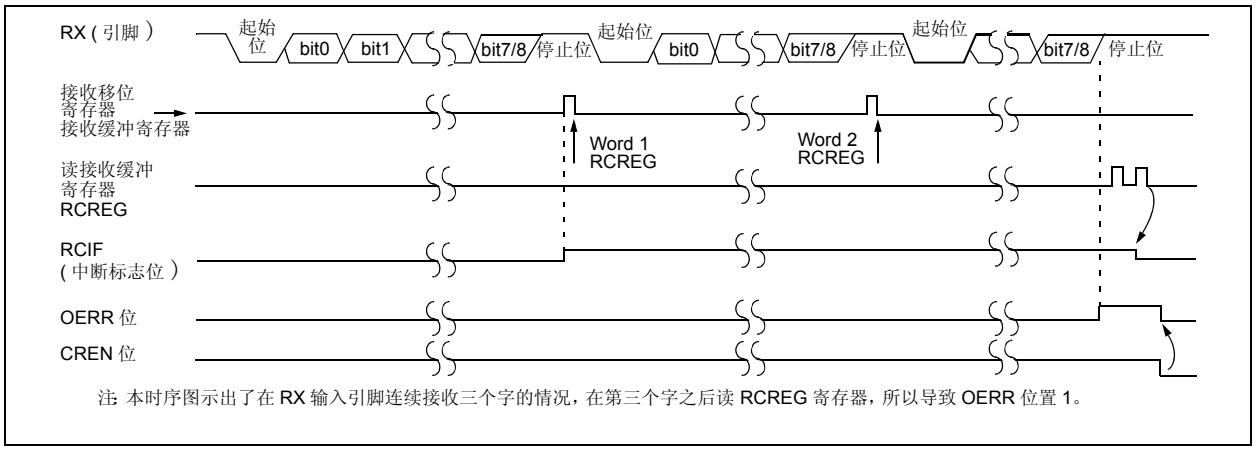
图 18-4： USART 接收原理框图



设置异步接收模式时必须遵循以下步骤：

1. 选择合适的波特率对 SPBRG 进行初始化，如果需要高速波特率，将 BRGH 置 1（参见 **18.3 “USART 波特率发生器（BRG）”小节**）；
2. 将 SYNC 清零， SPEN 置 1，使能异步串口；
3. 若需要中断，将 RCIE、GIE 和 PEIE 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 USART 工作在接收方式；
6. 当接收完成后，中断标志位 RCIF 被置 1，如果此时 RCIE 已被置 1，便产生中断；
7. 读 RCSTA 寄存器获取第 9 位数据（如果已使能），并判断在接收操作中是否发生错误；
8. 读 RCREG 寄存器来读取 8 位接收到的数据；
9. 如果发生错误，通过将 CREN 清零来清除错误。

图 18-5： 异步接收时序图

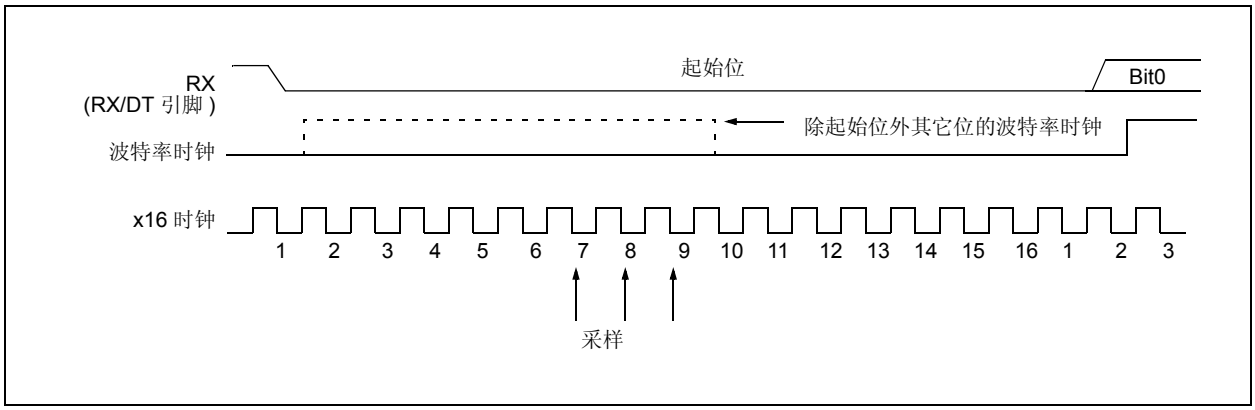


注：本时序图示出了在 RX 输入引脚连续接收三个字的情况，在第三个字之后读 RCREG 寄存器，所以导致 OERR 位置 1。

18.4.3 采样

检测电路对 RX/DT 引脚采样三次，以判定 RX 引脚上出现的是高电平还是低电平。图 18-6 是采样电路的波形图。无论 BRGH 位的状态如何，采样操作都一样，只是 16 分频时钟源有所不同。

图 18-6： RX 引脚的采样波形图， BRGH = 0 或 BRGH = 1



18.4.3.1 不同采样模式的单片机

除了下列单片机外，所有新型单片机都按照图 18-6 进行采样操作：

- PIC16C63
- PIC16C65
- PIC16C65A
- PIC16C73
- PIC16C73A
- PIC16C74
- PIC16C74A

上述单片机的采样电路按以下方式工作：如果 BRGH 位（TXSTA<2>）被清零（即在低波特率状态），采样在 16 分频时钟的第 7、8、9 个下降沿进行 (图 18-7)。如果 BRGH 被置 1(即在高波特率状态），采样发生在 4 分频时钟的第一个下降沿和它之后的第二个上升沿之间的 3 个时钟边沿 （图 18-8 和图 18-9）。

图 18-7： RX 引脚的采样时序图 （BRGH = 0）

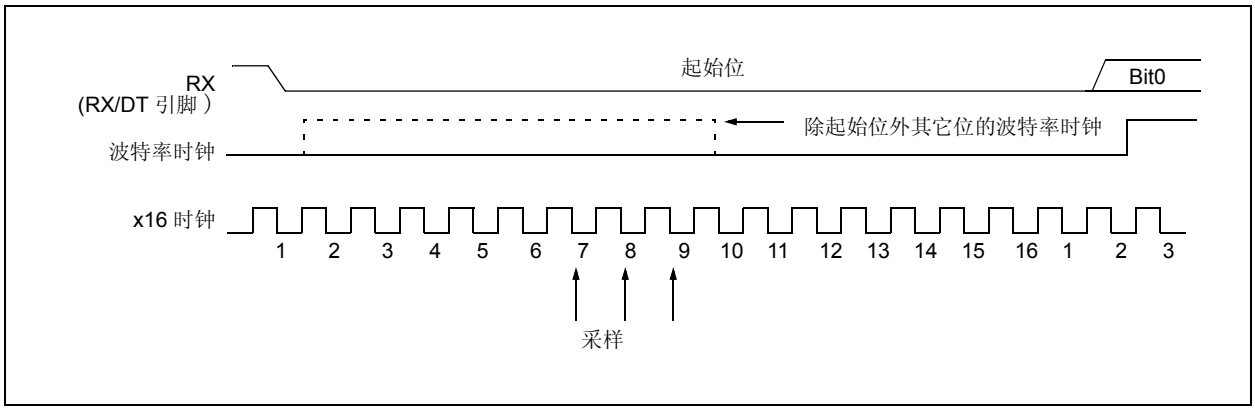


图 18-8： RX 引脚的采样时序图 （BRGH = 1）

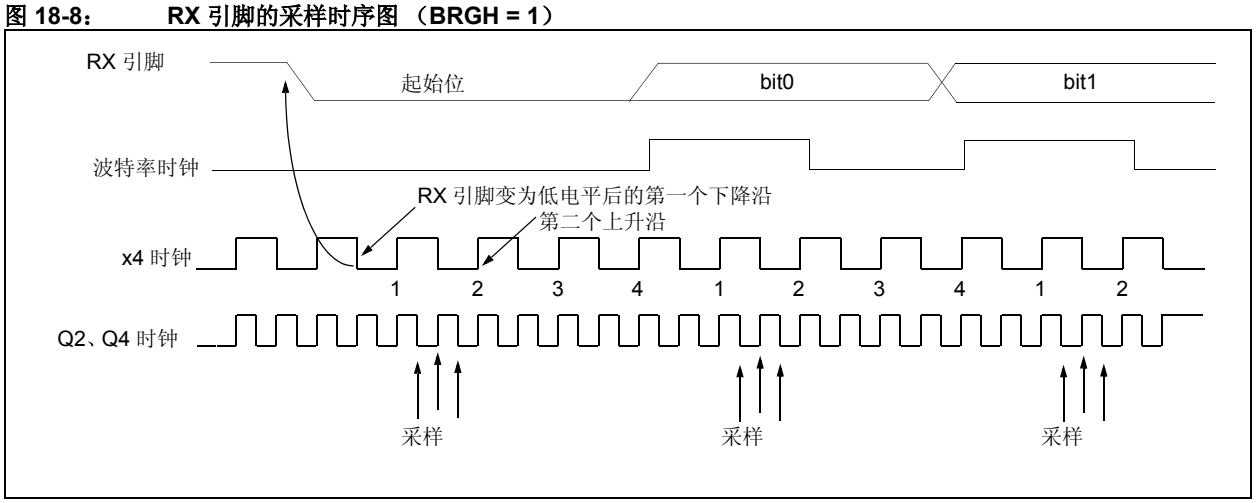


图 18-9： RX 引脚的采样时序图 （BRGH = 1）

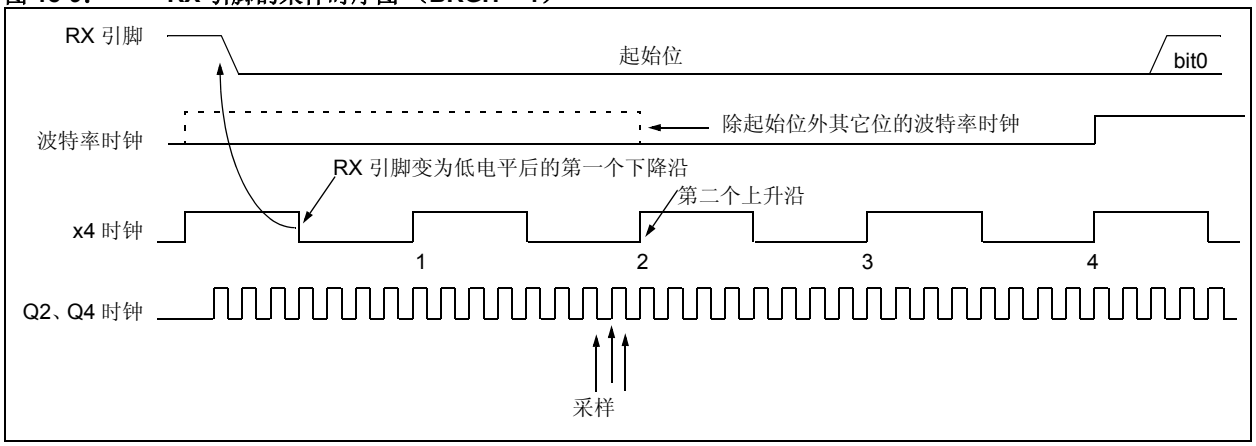


表 18-7： 与异步接收有关的寄存器

寄存器	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位、欠压复位时的值	其它复位时的值
PIR	RCIF (1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
RCREG	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0	0000 0000	0000 0000
PIE	RCIE (1)								0	0
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRG	波特率发生器寄存器								0000 0000	0000 0000

其中： x = 未知， - = 未用位，读为 ‘0’。
阴影异步接收模式未使用。

注 1：该位的位置和器件的具体型号有关。

18.5 USART 同步主控模式

在同步主控模式下，数据的传输是以半双工的方式进行，即发送和接收不同时进行。在发送数据时禁止接收数据，反之亦然。把 SYNC 位（TXSTA<4>）置 1 就可以进入同步工作方式。另外还应把 SPEN 使能位 （RCSTA<7>）置 1，将 TX/CK 和 RX/DT I/O 引脚分别配置为时钟线 CK 和数据线 DT。把 CSRC 位（TXSTA<7>）置 1 可进入主控模式，这意味着处理器在 CK 线上发送主控时钟信号。

18.5.1 USART 同步主控发送

图 18-1 是 USART 发送器的原理框图。发送器的核心是串行发送移位寄存器（TSR）。发送移位寄存器 TSR 从读 / 写发送缓冲器 TXREG 获得要发送的数据。TXREG 寄存器中的数据由软件控制写入。要等待 TSR 发送完上一个数据的最后一位，才能将 TXREG 中的数据送入 TSR 中。当 TSR 中的最后一位被发送完后， TSR 就从 TXREG 中装入新的数据（ 如果有数据的话 ）。当 TXREG 把数据送入 TSR 后（在一个时钟周期内完成），TXREG 变为空，同时中断标志位 TXIF 被置 1。置位 / 清零中断允许位 TXIE，可允许 / 禁止中断。不管 TXIE 位状态如何， TXIF 位都会被置 1，并且 TXIF 位不能用软件清零。只有当新数据写入 TXREG 寄存器时， TXIF 位才会被复位。 TXIF 标志位表示 TXREG 寄存器的状态，而 TRMT 位（TXSTA<1>）表示 TSR 寄存器的状态。 TRMT 位是一个只读位，当 TSR 为空时， TRMT 置 1。没有任何中断逻辑与 TRMT 位有联系，所以为确定 TSR 寄存器是否为空，只能通过对 TRMT 位查询进行判断。TSR 并未映射到数据存储器中，所以用户程序不能对它直接访问。

将 TXEN 位（TXSTA<5>）置 1 可使能发送，但是实际的发送是在 TXREG 寄存器装入数据之后才开始的。第一位数据在时钟线的下一个有效的时钟上升沿被移出，数据在同步时钟下降沿前后是稳定的 (图 18-10)。也可以通过先把发送数据送入 TXREG 寄存器，再将 TXEN 位置 1 来启动发送。这对于选择低速波特率时是有利的，因为当 TXEN、CREN 和 SREN 位清零时 BRG 保持复位状态。将 TXEN 位置 1 将启动 BRG，使其立即产生移位时钟。通常，在第一次开始发送时，TSR 寄存器为空，因此送到 TXREG 寄存器的数据会被立即送到 TSR 寄存器，导致 TXREG 寄存器为空，从而进行连续的发送 （背靠背发送）。

在发送过程中将 TXEN 位清零会导致发送中止，同时发送器被复位。 DT 和 CK 引脚将恢复到高阻状态。在发送过程中，CREN 或 SREN 中的任一位置 1 都会中止发送，同时 DT 引脚恢复到高阻状态 （准备接收）。如果 CSRC 位被置 1 （内部时钟）， CK 引脚将保持输出状态。虽然发送器的逻辑没有和这两个引脚连接，但是它不会被复位，必须通过对 TXEN 位清零来使发送器复位。如果 SREN 位被置 1 （将中断正在进行的发送，而开始接收一个字） ，在接收一个字后，SREN 位被清零，由于 TXEN 位仍然为 1，串口将恢复到发送模式。DT 线将立即从高阻抗接收模式切换到发送模式并开始启动发送。为避免这种情况的产生，必须将 TXEN 位清零。

为选择 9 位发送模式， TX9 位（TXSTA<6>）应被置 1，而且第 9 位数据应当写入 TX9D 位（TXSTA<0>）。必须在向 TXREG 寄存器写入 8 位数据之前将第 9 位数据写入 TX9D。这是因为当 TSR 为空时，向 TXREG 寄存器写数据会导致数据立即送入 TSR 寄存器。向 TX9D 位写新值之前向 TXREG 寄存器写数据，此时送入 TSR 的第 9 位数据是 TX9D 的 “当前”值，而不是将要写入的第 9 位数据。

设置同步主控方式应遵循以下步骤：

1. 选择合适的波特率对 SPBRG 进行初始化（见 **18.3 “USART 波特率发生器（BRG）”小节**）。
2. 将 SYNC、 SPEN 和 CSRC 位置 1，使能同步主控串行口。
3. 若需要中断，将 TXIE 位置 1；
4. 若需要传送 9 位数据，将 TX9 位置 1；
5. 将 TXEN 位置 1，使 USART 工作在发送模式；
6. 若选择发送 9 位数据，第 9 位数据应该写入 TX9D；
7. 把数据送入 TXREG 寄存器来启动发送。

表 18-8： 与同步主控发送有关的寄存器

寄存器	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位和欠压复位时的值	其它复位时的值
PIR	TXIF (1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
TXREG	TX7	TX6	TX5	TX4	TX3	TX2	TX1	TX0	0000 0000	0000 0000
PIE	TXIE (1)								0	0
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRG	波特率发生器寄存器								0000 0000	0000 0000

其中： x = 未知， - = 未用位，读为 ‘0’。
阴影部分同步接收模式未使用。
注 1：该位的位置和器件的具体型号有关。

图 18-10： 同步发送时序图

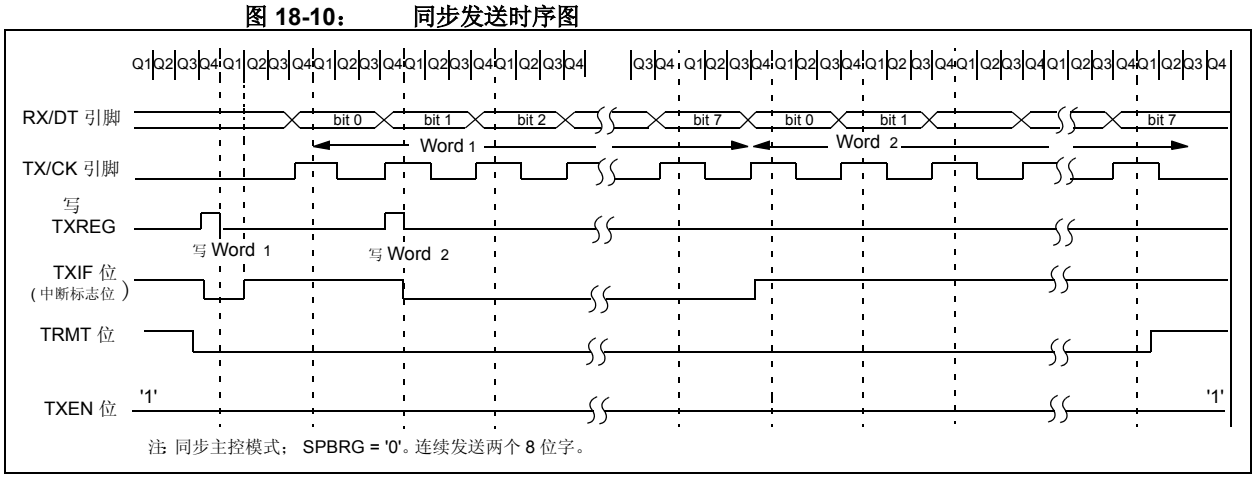
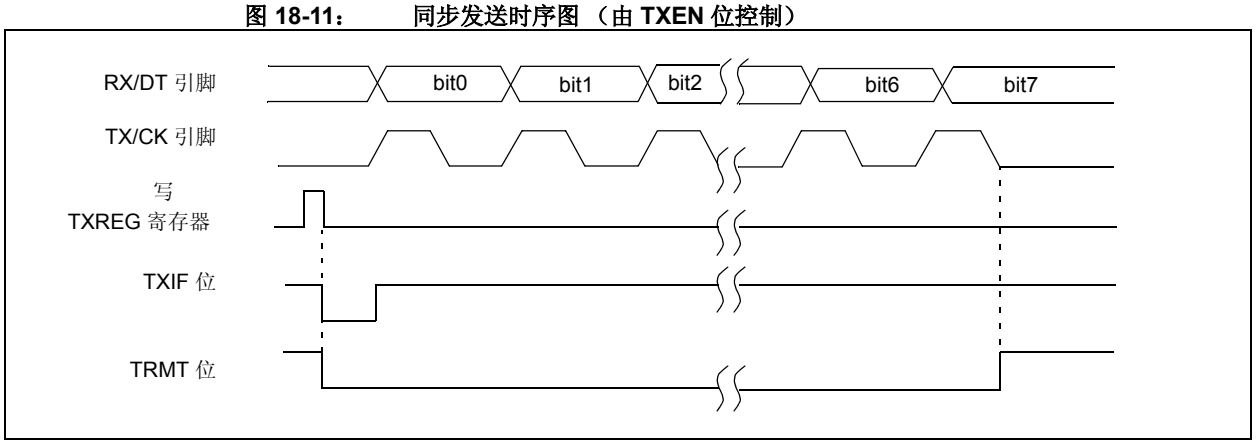


图 18-11： 同步发送时序图 （由 TXEN 位控制）



18.5.2 USART 同步主控接收

一旦选择同步模式后，只要把 SREN 位（RCSTA<5>）或 CREN 位（RCSTA<4>）置 1，即可进入同步主控接收模式。在时钟的下降沿采样 RX/DT 引脚上的数据。如果 SREN =1，仅接收一个字；如果 CREN = 1，则可连续地接收数据，直到 CREN 位被清零。如果两个位都被置 1，则 CREN 位优先而进行连续接收。在最后一位的时钟到来之后， RSR 中接收到的数据被送到 RCREG 寄存器（如果该寄存器为空）。数据传送完之后，中断标志位 RCIF 被置 1。实际的中断可以通过将 RCIE 位置 1（或清零）来使能（或禁止）。标志位 RCIF 是一个只读位，由硬件清零，它是在 RCREG 寄存器被读之后或 RCREG 寄存器为空时清零。由于 RCREG 寄存器是一个双缓冲的寄存器（即两级深度的 FIFO），因此允许在两个字节的数据接收到并传送给 RCREG FIFO 后，第三个字节再移位到 RSR 寄存器。在检测到第三个字节的最后一位后，如果 RCREG 寄存器仍然是满的，则溢出错误标志位 OERR（RCSTA<1>）被置 1， RSR 寄存器中的字会丢失。可以对 RCREG 寄存器读两次以重新获得 FIFO 中的两个字节。OERR 位必须由软件清零（将 CREN 位清零）。如果 OERR 位置 1，则禁止 RSR 中的数据传送到 RCREG 寄存器，因此如果 OERR 位被置 1，必须将它清零。第 9 位接收到的数据以与接收数据同样的方式被缓冲。读 RCREG 寄存器将会给 RX9D 位装入新值，因此为了不丢失 RX9D 位原来的信息，用户必须在读 RCREG 寄存器之前读 RCSTA 寄存器。

在设置同步主控接收模式时，应遵循以下步骤：

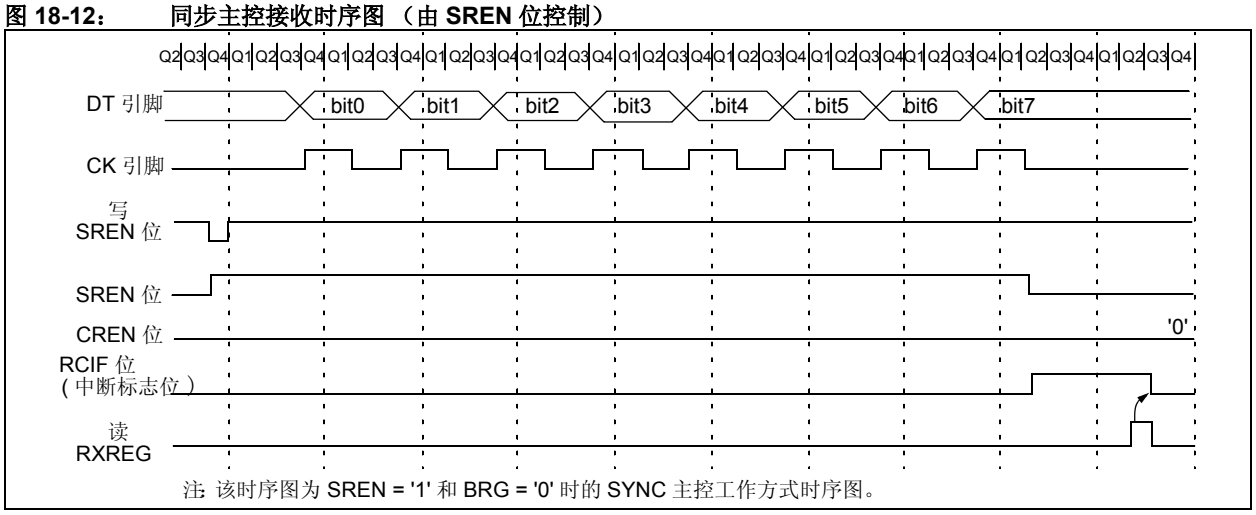
1. 选择合适的波特率对 SPBRG 进行初始化 （见 **18.3 “USART 波特率发生器 （BRG）” 小节** ）。
2. 将 SYNC、 SPEN 和 CSRC 位置 1，使能同步主控串行口；
3. 确保 CREN 和 SREN 位清零；
4. 若需要中断，清零 RCIE 位；
5. 若要接收 9 位数据，将 RX9 位置 1 ；
6. 若需要单字节接收，将 SREN 位置 1 ；若需要连续接收，将 CREN 位置 1 ；
7. 当接收完成后，中断标志位 RCIF 被置 1，如果此时 RCIE 已被置 1，便产生中断；
8. 如果设定接收 9 位数据，读 RCSTA 寄存器获取第 9 位数据，并判断在接收操作中是否发生错误；
9. 读 RCREG 寄存器来读取 8 位接收到数据；
10. 如果发生某种错误，通过将 CREN 清零来清除错误。

表 18-9： 与同步主控接收模式有关的寄存器

寄存器	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位和欠压复位时的值	其它复位时的值
PIR	RCIF (1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
RCREG	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0	0000 0000	0000 0000
PIE	RCIE (1)								0	0
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRG	波特率发生器寄存器								0000 0000	0000 0000

其中： x = 未知， - = 未用位，读为 ‘0’。
阴影部分同步接收模式未使用。
注 1：该位的位置和器件的具体型号有关。

图 18-12： 同步主控接收时序图 （由 SREN 位控制）



注: 该时序图为 SREN = '1' 和 BRG = '0' 时的 SYNC 主控工作方式时序图。

18.6 USART 同步从动模式

同步从动模式与主控模式不同，其移位时钟信号是在 TX/CK 引脚上由外部提供（主控模式是由内部提供移位时钟）。这样就允许器件在休眠状态下发送或接收数据。把 CSRC 位（TXSTA<7>）清零即可进入从动模式。

18.6.1 USART 同步从动发送

除了休眠状态外，同步主控模式和从动模式的操作是一样的。

如果向缓冲器 TXREG 写入 2 个字，然后执行 SLEEP 指令，则会发生以下事件：

a) 第一个字立即传送到移位寄存器 TSR 进行发送；
b) 第二个字仍保存在 TXREG 寄存器中；
c) TXIF 中断标志位不会被置 1；
d) 当第一个字已移出 TSR 后，TXREG 将第二个字送入 TSR，TXIF 标志位此时被置 1；
e) 如果中断允许位 TXIE 为 1，中断将把器件从休眠状态唤醒，如果允许全局中断，那么程序就跳转到中断向量（0004h）。

在设置同步从动发送模式时，应遵循以下步骤：

1. 置位 SYNC 和 SPEN 位，清零 CSRC 位，来使能同步从动串口；
2. 清零 CREN 和 SREN 位；
3. 如果需要中断， TXIE 置 1；
4. 若要发送 9 位数据，置位 TX9 位；
5. 置位 TXEN 位，使能发送模式；
6. 若选择发送 9 位数据，第 9 位应该先写入 TX9D；
7. 把数据送入 TXREG 寄存器启动发送。

表 18-10： 与同步从动发送有关的寄存器

寄存器	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位和欠压复位时的值	其它复位时的值
PIR	TXIF (1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
TXREG	TX7	TX6	TX5	TX4	TX3	TX2	TX1	TX0	0000 0000	0000 0000
PIE	TXIE (1)								0	0
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRG	波特率发生器寄存器								0000 0000	0000 0000

其中： x = 未知，- = 未用位，读为 ‘0’。
阴影部分同步从动接收模式未使用。
注 1：该位的位置和器件的具体型号有关。

18.6.2 USART 同步从动接收

除了休眠状态外，同步主控模式和从动模式的操作基本是一样的。另外，在从动模式下没有使用 SREN 位。

如果在执行 SLEEP 指令之前已使能接收模式 （即置位 CREN 位），那么在休眠状态下仍可以接收数据。当接收完数据字后，RSR 寄存器将把数据送入 RCREG 寄存器，并且如果 RCIE 使能位被置 1，则产生的中断将唤醒器件。如果全局中断允许，那么程序就跳转到中断向量 (0004h)。

在设置同步从动接收模式时，应遵循以下步骤：

1. 通过将 SYNC 和 SPEN 位置 1，将 CSRC 位清零，使能同步主控串口。
2. 如果需要中断，置位 RCIE 位；
3. 如果要接收 9 位数据，置位 RX9 位；
4. 置位 CREN 位，使 USART 工作在接收模式；
5. 接收完成后， RCIF 置 1 ；如果此时 RCIE 置 1，将产生中断；
6. 若选择接收 9 位数据，从 RCSTA 中读出第 9 位数据，并判断在接收操作中是否发生错误；
7. 读 RCREG 寄存器以读取 8 位接收到的数据；
8. 如果发生任何错误，通过将 CREN 清零来清除错误。

表 18-11： 与同步从动接收有关的寄存器

寄存器	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	上电复位和欠压复位时的值	其它复位时的值
PIR	RCIF (1)								0	0
RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00x
RCREG	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0	0000 0000	0000 0000
PIE	RCIE (1)								0	0
TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
SPBRG	波特率发生器寄存器								0000 0000	0000 0000

其中： x = 未知， - = 未用位，读为 ‘0’。
阴影部分同步接收模式未使用。
注 1：该位的位置和器件的具体型号有关。

18.7 初始化

例 18-2 是异步发送器 / 接收器模式的初始化程序。例 18-3 用于同步模式。在这两个例子中，数据都是 8 位的，要装入 SPBRG 寄存器的值取决于所需的波特率和器件频率。

例 18-2： 异步发送器 / 接收器

```
    BSF    STATUS,RP0     ; Go to Bank1
    MOVLW  <baudrate>     ; Set Baud rate
    MOVWF  SPBRG
    MOVLW  0x40           ; 8-bit transmit, transmitter enabled,
    MOVWF  TXSTA          ; asynchronous mode, low speed mode
    BSF    PIE1,TXIE      ; Enable transmit interrupts
    BSF    PIE1,RCIE      ; Enable receive interrupts
    BCF    STATUS,RP0     ; Go to Bank 0
    MOVLW  0x90           ; 8-bit receive, receiver enabled,
    MOVWF  RCSTA          ; serial port enabled
```

例 18-3： 同步发送器 / 接收器

```
    BSF    STATUS,RP0      ; Go to Bank 1
    MOVLW  <baudrate>      ; Set Baud Rate
    MOVWF  SPBRG
    MOVLW  0xB0            ; Synchronous Master,8-bit transmit,
    MOVWF  TXSTA           ; transmitter enabled, low speed mode
    BSF    PIE1,TXIE       ; Enable transmit interrupts
    BSF    PIE1,RCIE       ; Enable receive interrupts
    BCF    STATUS,RP0      ; Go to Bank 0
    MOVLW  0x90            ; 8-bit receive, receiver enabled,
    MOVWF  RCSTA           ; continuous receive, serial port enabled
```

18.8 设计技巧

问 1： *为什么在使用异步模式时，出现了很多传输错误？*

答 1：

最常见的原因是：

1. 对 PIC16C65/65A/73/73A/74/74A 单片机使用高速模式（BRGH 置 1），而这些单片机在异步高速模式时，其采样电路的工作方式与其它型号单片机不同。
2. 没有正确计算出要装入 SPBRG 寄存器的值。
3. 发送和接收的波特率误差总和太大。

18.9 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与本章相关的应用笔记有：

标题	应用笔记 #
Serial Port Utilities	AN547
Servo Control of a DC Brushless Motor	AN543

18.10 版本历史

版本 A

这是描述 USART 模块的初始发行版。

第 19 章 参考电压模块

目录

本章包括下面一些主要内容：

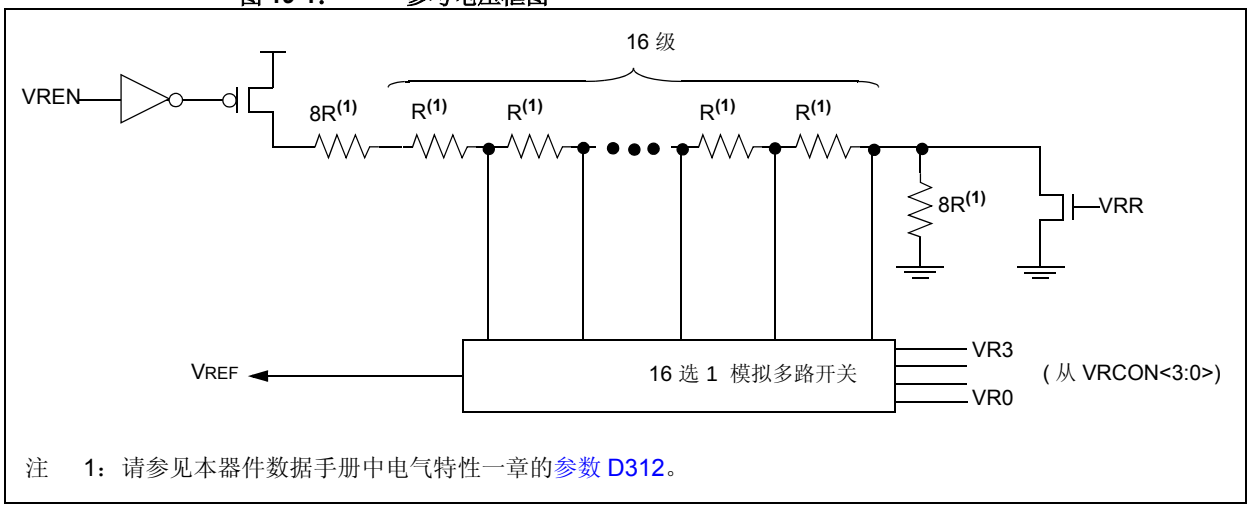
19.1 简介 19-2
19.2 控制寄存器 19-3
19.3 配置参考电压 19-4
19.4 参考电压精度 19-5
19.5 休眠模式下的操作 19-5
19.6 复位的影响 19-5
19.7 连接注意事项 19-6
19.8 初始化 19-7
19.9 设计技巧 19-8
19.10 相关应用笔记 19-9
19.11 版本历史 19-10

19.1 简介

参考电压模块通常和比较器模块一起工作。因为比较器模块并不需要很高的输入驱动，所以参考电压的驱动能力不是很强。

参考电压模块有一个 16 阶梯形电阻网络，可以改变参考电压模块的输出电压。分割梯形电阻可提供两个不同量程范围的参考电压，并且在不需要使用参考电压时可以将其关闭以降低功耗。VRCON 寄存器对参考电压进行控制，如图 19-1 所示。图 19-1 为参考电压框图。在每个量程范围内， 16 阶都是单调的 （即，每组渐增的编码会得到渐增的输出）。

图 19-1： 参考电压框图



注 1：请参见本器件数据手册中电气特性一章的参数 D312。

表 19-1： 典型参考电压 （VDD = 5.0V）

VR3:VR0	VREF	
	VRR = 1	VRR = 0
0000	0.00 V	1.25 V
0001	0.21 V	1.41 V
0010	0.42 V	1.56 V
0011	0.63 V	1.72 V
0100	0.83 V	1.88 V
0101	1.04 V	2.03 V
0110	1.25 V	2.19 V
0111	1.46 V	2.34 V
1000	1.67 V	2.50 V
1001	1.88 V	2.66 V
1010	2.08 V	2.81 V
1011	2.29 V	2.97 V
1100	2.50 V	3.13 V
1101	2.71 V	3.28 V
1110	2.92 V	3.44 V
1111	3.13 V	3.59 V

19.2 控制寄存器

寄存器 19-1： VRCON 寄存器

R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
VREN	VROE	VRR	—	VR3	VR2	VR1	VR0
bit 7							bit 0

bit 7 **VREN**：VREF 使能
1 = 开启 VREF 电路
0 = 关闭 VREF 电路

bit 6 **VROE**：VREF 输出使能
1 = VREF 与比较器模块的 VREF 相连。电压值也输出到 VREF 引脚
0 = VREF 与比较器模块不相连。电压值与 VREF 引脚无关

bit 5 **VRR**：VREF 范围选择
1 = 0V 至 0.75 VDD，间隔为 VDD/24 的步长
0 = 0.25 至 0.75 VDD，间隔为 VDD/32 的步长

bit 4 **未用位**：读为 “0”

bit 3:0 **VR3:VR0**：VREF 值选择 0 ≤ VR3:VR0 ≤ 15

若 VRR = 1：

$V_{REF} = (VR<3:0>/ 24) \bullet V_{DD}$

若 VRR = 0：

$V_{REF} = 1/4 * V_{DD} + (VR3:VR0/ 32) \bullet V_{DD}$

图注	
R = 可读位	W = 可写位
U = 未用位，读为 “0”	- n = 上电复位时的值

19.3 配置参考电压

在每个量程范围内，参考电压模块可输出 16 种不同的电压值。

下面是计算参考电压输出值的公式：

若 VRR = 1：VREF = (VR3:VR0/24) x VDD

若 VRR = 0：VREF = (VDD x 1/4) + (VR3:VR0/32) x VDD

当改变 VREF 输出值时，需要考虑参考电压的稳定时间。例 19-1 说明了如何在 VDD = 5.0V 时输出一个 1.25V 的参考电压。

通常系统的 VREF 和 VDD 是已知的，只需确定装入 VR3:VR0 的值。公式 19-1 说明了如何计算 VR3:VR0 的值。因为 VR3:VR0 的值只能是整数，所以可能会出现一些误差。同时为了使结果不大于 15，必须正确选择 VREF 和 VDD 的值。

公式 19-1： 计算 VR3:VR0

若 VRR = 1

$$VR3:VR0 = \frac{VREF}{VDD} \quad X\ 24$$

若 VRR = 0

$$VR3:VR0 = \frac{VREF - VDD/4}{VDD} \quad X\ 32$$

19.4 参考电压精度

由于参考电压模块结构上的限制，不能实现从VSS到VDD的满量程输出。梯形电阻网络（图 19-1）顶部和底部的晶体管使 VREF 的值无法达到 VSS 或 VDD。由于参考电压是由 VDD 供电的，因此 VREF 的输出随 VDD 变化。参考电压的绝对精度可查看数据手册电气规范的参数 D311。

19.5 休眠模式下的操作

如果是因中断或看门狗定时器超时将器件从休眠模式下唤醒， VRCON 寄存器将不受影响。为了降低休眠模式下的电流消耗，应关闭参考电压模块。

19.6 复位的影响

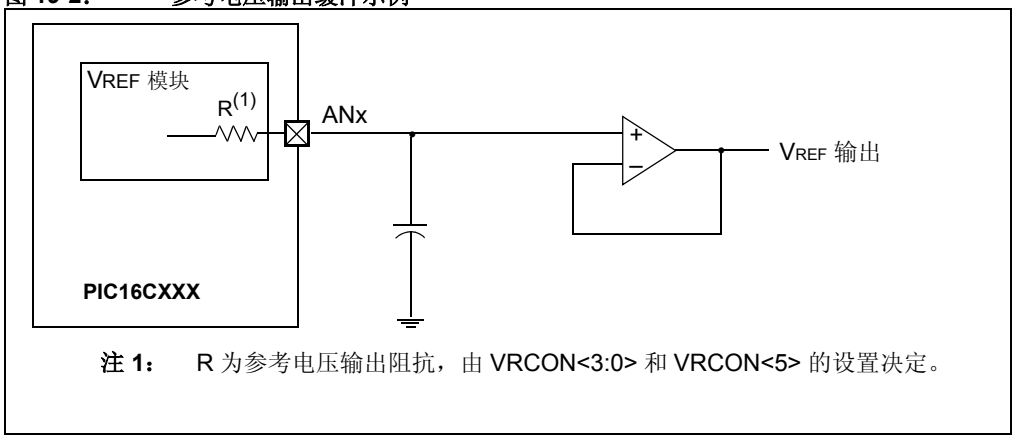
单片机复位使 VREN 位 （VRCON<7>）清零，从而关闭参考电压模块。同时使 VROE 位（VRCON<6>）清零将参考电压与 VREF 引脚断开，使 VRR 位 （VRCON<5>）清零来选择高值量程。 VREF 值选择位 VRCON<3:0> 也被清零。

19.7 连接注意事项

参考电压模块的操作是独立于比较器模块的。如果 TRIS 和 VROE（VRCON<6>）位被置“1”，参考电压发生器的输出将从 VREF 引脚输出。参考电压输出连在 VREF 引脚时，如果有输入信号也加在该引脚上，将增大电流消耗。同样，使能 VREF 输出时，如果 VREF 被用作数字输出，也将增大电流消耗。

VREF 引脚可以用作简单的 D/A 输出，但是其驱动能力有限。要提高驱动能力，必须在参考电压输出端外接一个缓冲器。图 19-2 举例说明了这一缓冲技术。

图 19-2： 参考电压输出缓冲示例



注 1： R 为参考电压输出阻抗，由 VRCON<3:0> 和 VRCON<5> 的设置决定。

19.8 初始化

例 19-1 介绍了配置参考电压模块的步骤。

例 19-1： 配置参考电压模块

```
    MOVLW   0x02        ; 4 Inputs Muxed to 2 comparators
    MOVWF   CMCON       ;
    BSF     STATUS,RP0  ; go to Bank1
    MOVLW   0x07        ; RA3:RA0 are outputs
    MOVWF   TRISA       ; outputs
    MOVLW   0xA6        ; enable VREF
    MOVWF   VRCON       ; low range set VR3:VR0 = 6
    BCF     STATUS,RP0  ; go to Bank0
    CALL    DELAY10     ; 10 µs delay
```

19.9 设计技巧

问 1： ***我的 VREF 与预期值不同。***

答 1：

器件 VDD 的任何变化都会立即影响 VREF 引脚。此外计算的分压比要正确。

问 2： ***我将 VREF 接到一个低阻抗电路上， VREF 没有达到预期的值。***

答 2：

参考电压模块不能驱动较大的负载。 PICmicro® 单片机的 VREF 引脚和负载间必须接缓冲器。

19.10 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 （即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制，即可使用。目前与参考电压模块相关的应用笔记有：

标题	应用笔记 #
Resistance and Capacitance Meter using a PIC16C622	AN611

19.11 版本历史

版本 A

这是描述参考电压模块的初始发行版。

第 20 章 比较器

目录

本章包括下面一些主要内容：

20.1 简介 20-2
20.2 控制寄存器 20-3
20.3 设置比较器模式 20-4
20.4 比较器工作原理 20-6
20.5 比较器参考源 20-6
20.6 比较器的响应时间 20-8
20.7 比较器输出 20-8
20.8 比较器中断 20-9
20.9 休眠状态下比较器的操作 20-9
20.10 复位的影响 20-9
20.11 模拟输入连接方式注意事项 20-10
20.12 初始化 20-11
20.13 设计技巧 20-12
20.14 相关应用笔记 20-13
20.15 版本历史 20-14

20.1 简介

比较器模块包含两个模拟比较器。比较器的输入端与 I/O 引脚复用。片内参考电压（参见**“参考电压模块”一章**）也能作为比较器的一个输入。

CMCON 寄存器（如图 20-1 所示）用于控制比较器的输入输出复用。图 20-1 是比较器的结构框图。

20.2 控制寄存器

寄存器 20-1： CMCON 寄存器

R-0	R-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	—	—	CIS	CM2	CM1	CM0
bit 7							bit 0

bit 7 **C2OUT**：比较器 2 输出指示位
1= 表示 C2 VIN+ > C2 VIN–
0= 表示 C2 VIN+ < C2 VIN–

bit 6 **C1OUT**：比较器 1 输出指示位
1= 表示 C1 VIN+ > C1 VIN–
0= 表示 C1 VIN+ < C1 VIN–

bit 5:4 **未用**：读为 “0”

bit 3 **CIS**：比较器输入选择位

当 CM2:CM0=001 时：

1= 表示 C1 VIN– 和 AN3 相连
0= 表示 C1 VIN– 和 AN0 相连

当 CM2:CM0=010 时：

1= 表示 C1 VIN– 和 AN3 相连
C2 VIN– 和 AN2 相连
0= 表示 C1 VIN– 和 AN0 相连
C2 VIN– 和 AN1 相连

bit 2:0 **CM2:CM0**：比较器模式选择位
见图 20-1。

图注：		
R = 可读位	W = 可写位	
U = 未用位，读为 “0”		-n= 上电复位时的值

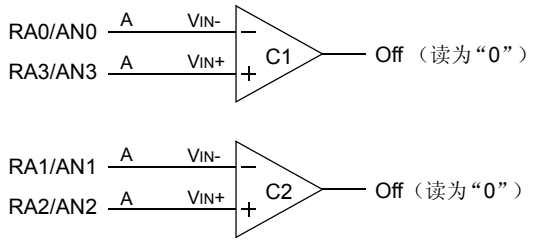
20.3 设置比较器模式

比较器共有 8 种工作模式，如图 20-1 所示。 CMCON 寄存器用于设置比较器模式，各种模式下比较器 I/O 引脚的输入输出方向由 TRIS 寄存器控制。如果改变比较器模式，比较器的输出电平可能会在器件电气规范中规定的时间内对新模式无效。

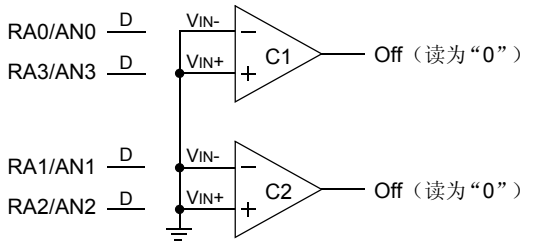
注： 改变比较器工作模式时，应禁止比较器发生中断，以免产生错误中断。

图 20-1：　　比较器 I/O 工作模式

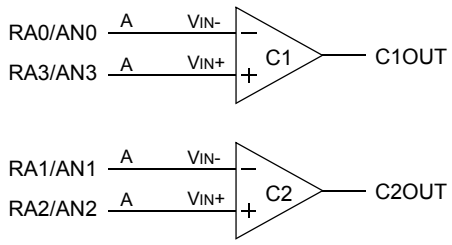
CM2:CM0 = 000
比较器复位（上电复位的缺省值）



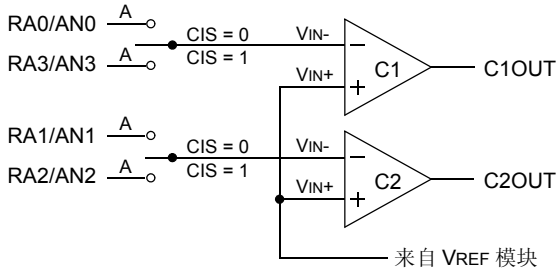
CM2:CM0 = 111
比较器关闭



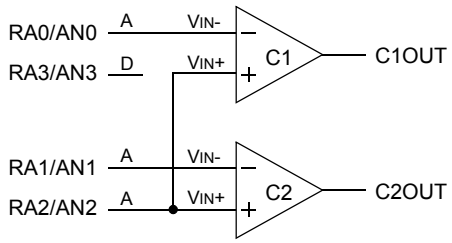
CM2:CM0 = 100
两个独立的比较器



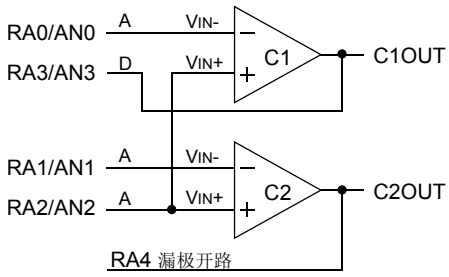
CM2:CM0 = 010
四输入的双比较器



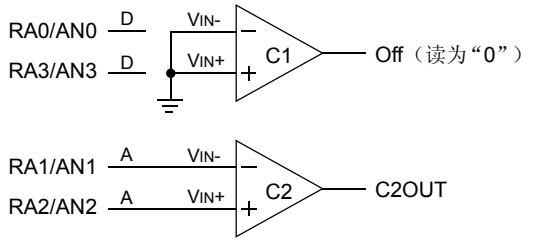
CM2:CM0 = 011
具有公共参考端的双比较器



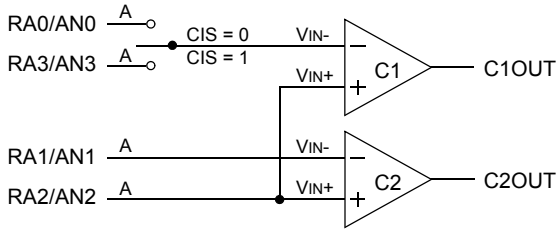
CM2:CM0 = 110
具有公共参考端的双比较器，比较结果可由 I/O 端口输出



CM2:CM0 = 101
单比较器



CM2:CM0 = 001
三输入的双比较器



A = 模拟输入，端口始终为 0。
D = 数字输入。
CIS (CMCON<3>) 是比较器输入选择开关。

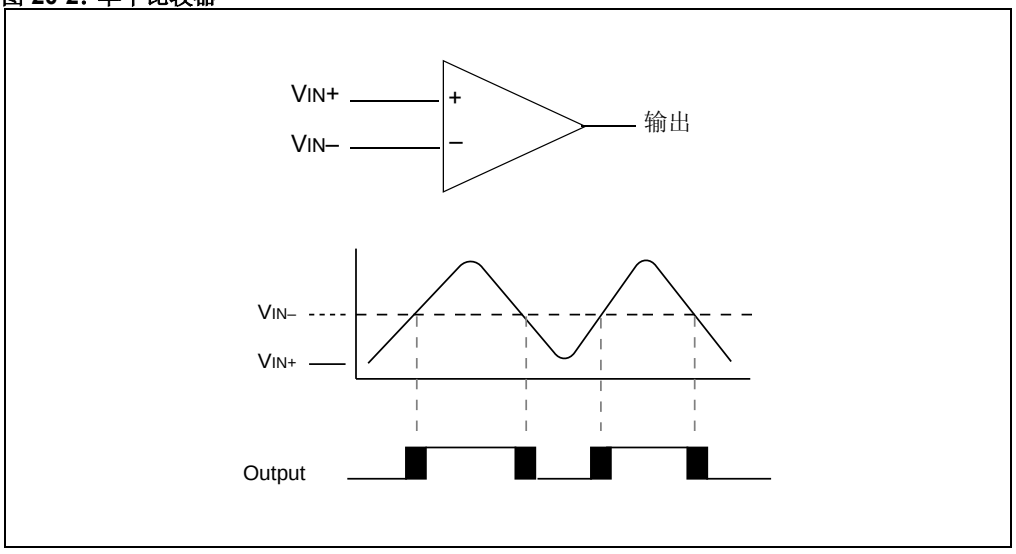
20.4 比较器工作原理

图 20-2 所示，单比较器的模拟输入电压与数字输出之间的关系。当模拟输入端 VIN+ 电压低于 VIN– 电压时，比较器输出数字低电平。当模拟输入端 VIN+ 电压高于 VIN– 时，比较器输出数字高电平。图 20-2 中比较器输出部分的阴影区是由于输入偏移和响应时间所引起的不确定区。

20.5 比较器参考源

根据不同的工作模式，比较器可以使用外部或内部的参考源。加在 VIN– 的模拟信号和 VIN+ 的信号相比较，比较器的数字输出也随之改变（图 20-2）。

图 20-2：单个比较器



20.5.1 外部参考信号

当使用外部参考电压时，比较器模块可以设置为两个比较器使用同一个参考源，也可以使用不同的参考源。参考信号应在 VSS 和 VDD 之间，且可加到比较器的任一引脚上。

20.5.2 内部参考信号

比较器模块也可以选择使用内部产生的参考电压。**“参考电压模块”一章**对产生这一内部参考信号的参考电压模块进行了详细描述。在 CM2:CM0 = 010 模式（如图 20-1）下，比较器使用内部参考信号。该模式下，内部参考电压加到两个比较器的 VIN+ 输入引脚上。

比较器在任何一种模式下，都可使用内部参考电压。比较器以这种方式工作时，I/O/VREF 引脚可用作 I/O 引脚。此时参考电压连接在比较器的 VREF 引脚 （即 VIN+ 引脚）上。

20.6 比较器的响应时间

响应时间是指比较器从选定一个新的参考电压或输入源到输出达到一个稳定有效电平时的最短时间。如果内部参考电压发生了改变，在使用比较器输出时必须考虑到内部参考电压的最大建立时间的影响。否则应应用比较器的最大响应时间。

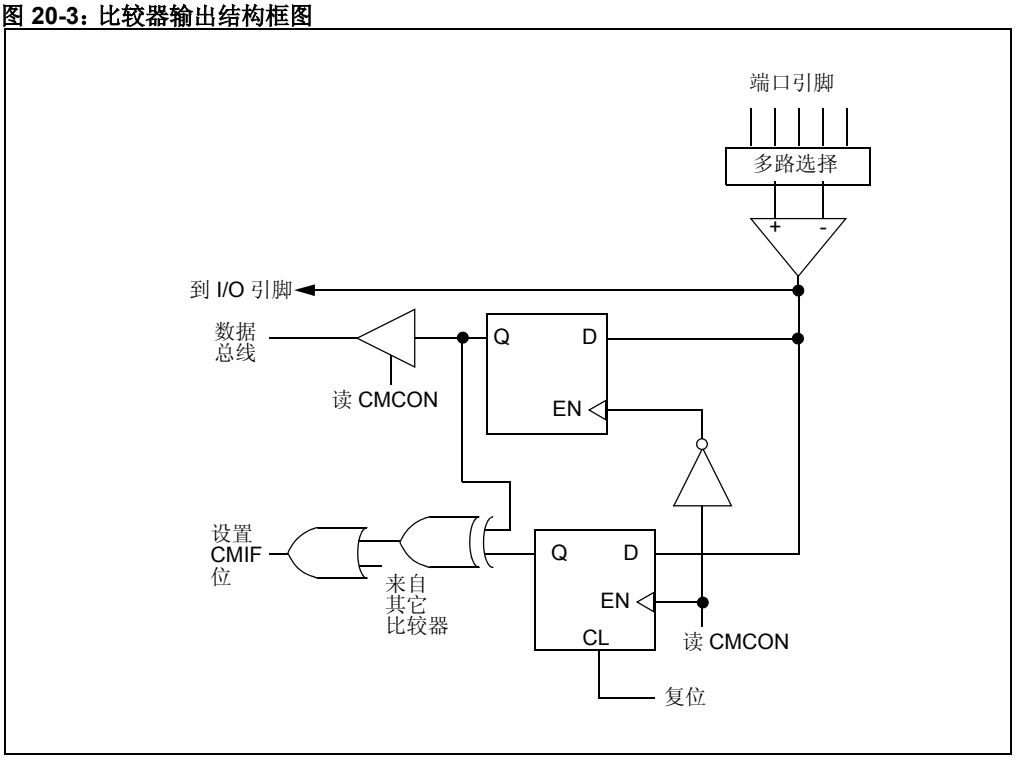
20.7 比较器输出

通过 CMCON 寄存器的相应位，可以得到比较器的输出，这些位都是只读的。比较器的输出也可以直接输出到 I/O 引脚。当在 CM2:CM0 = 110 模式时，可以切换多路转换器来选择输出路径的 I/O 引脚，而各 I/O 引脚并不会同时输出比较结果。每个比较器的不确定区的大小与规格说明里给的输入偏置电压和响应时间相关的。图 20-3 是比较器输出的结构框图。

在该模式下，通过设置 TRIS 寄存器的相应位，可以允许或禁止 I/O 引脚输出比较结果。

注 1：	如果端口寄存器的引脚被定义为模拟输入，读这些引脚的结果都为“0”。而引脚定义为数字输入时，将根据施密特触发器的门槛电平输入规范对模拟输入进行转换。
注 2：	模拟电平加在数字输入引脚将增加输入缓冲器的消耗电流。

图 20-3：比较器输出结构框图



20.8 比较器中断

一旦比较结果与保存在 CMxOUT 位中的上次值不同时，就会将比较器中断标志置为 1。软件需要读 CMCON<7:6> 的值与前次结果做比较，以判断值是否不同，因此应妥善保存比较器各输出位的结果。 CMIF 位是比较器中断标志位，该位必须由软件清零，有时也可以将其置 1 来实现模拟中断。

要使能比较器中断， CMIE 位和 PEIE 位 (INTCON<6>) 必须置 1，同时 GIE 位也必须置 1。只要其中的任何一位被清零，即使有中断条件产生而将 CMIF 位置 1，仍不会发生中断响应。

在中断服务程序中，用户可以通过下面的步骤清除中断：

a) 对 CMCON 寄存器进行读写操作。这将会把 CMxOUT 位的新值装入 CMCON 寄存器。

b) CMIF 标志位清零。

如果不结束中断条件， CMIF 标志位会继续置为 1 ；对 CMCON 寄存器的读操作将结束中断条件，此时才可能将 CMIF 标志位清零。

20.9 休眠状态下比较器的操作

当比较器处于运行状态而器件处于休眠状态时，比较器仍保持工作。此时如果中断使能，则中断仍将工作。在中断使能时，中断会把器件从休眠状态下唤醒。当比较器处于通电状态时，每个比较器工作时都会消耗额外的电流，具体参见比较器的规格说明。若要减小休眠模式下的功耗，可在进入休眠状态前将 CM2:CM0 = 111，以关闭比较器模块。器件从休眠状态中唤醒时，CMCON 寄存器的内容不保持不变。

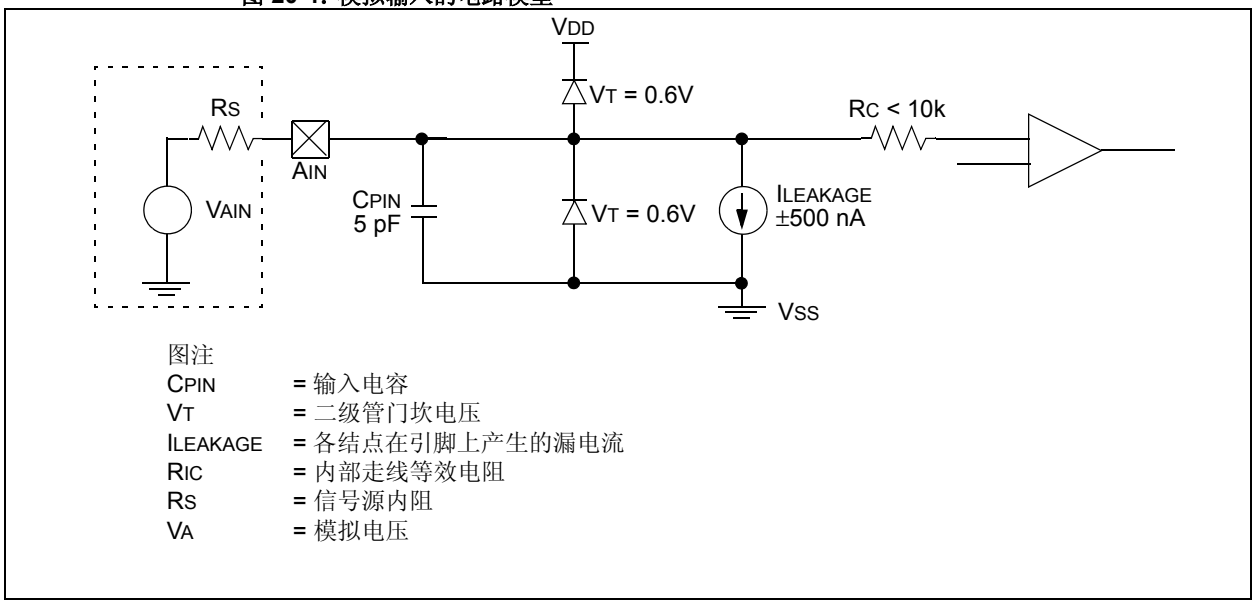
20.10 复位的影响

复位器件，会使 CMCON 寄存器恢复复位值，使比较器模块处于比较器复位模式（CM2:CM0 = 000）。这时所有相关的输入引脚均为模拟输入，会减小器件的工作电流。在复位期间，比较器将掉电。

20.11 模拟输入连接方式注意事项

图 20-4 是一个简化的模拟输入电路。由于模拟引脚和数字输出端相连，因而它们与 VDD 及 VSS 之间加有反向偏置二极管，模拟输入电压被限制在 VDD 和 VSS 之间。一旦输入电压在任一方向上超出极限 0.6V，就会有一个二极管正偏使输入电压被钳位。模拟输入信号源的最大阻抗值为 10 kΩ （推荐值）。

图 20-4：模拟输入的电路模型



图注
CPIN = 输入电容
VT = 二级管门坎电压
ILEAKAGE = 各结点在引脚上产生的漏电流
RIC = 内部走线等效电阻
RS = 信号源内阻
VA = 模拟电压

表 20-1： 和比较器模块相关的寄存器

寄存器名	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	上电复位，欠压复位后的值	其它复位后的值
CMCON	C2OUT	C1OUT	—	—	CIS	CM2	CM1	CM0	00-- 0000	00-- 0000
VRCON	VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	000- 0000	000- 0000
INTCON	GIE	PEIE	T0IE	INTE	RBIE(2)	T0IF	INTF	RBIF(2)	0000 000x	0000 000x
PIR	CMIF (1)								0	0
PIE	CMIE (1)								0	0

图注： x = 未知， - = 未用，读为 “0”。
阴影部分不用于比较器模块。
注 1：该位的位置和器件的具体型号有关。
2：这些位也可命名为 GPIE 和 GPIF。

20.12 初始化

例 20-1 中的代码为设置 PIC16C62X 器件的比较器模块的示例步骤。RA3 和 RA4 设置为数字输出。 RA0 和 RA1 设置为两个比较器的 V- 输入， RA2 为 V+ 输入。

例 20-1： 初始化比较器模块 (PIC16C62X)

```
FLAG_REG   EQU    0X20
;
   CLRF    FLAG_REG      ; Init flag register
   CLRF    PORTA         ; Init PORTA
   ANDLW   0xC0          ; Mask comparator bits
   IORWF   FLAG_REG,F    ; Store bits in flag register
   MOVLW   0x03          ; Init comparator mode
   MOVWF   CMCON         ; CM<2:0> = 011
   BSF     STATUS,RP0    ; Select Bank1
   MOVLW   0x07          ; Initialize data direction
   MOVWF   TRISA         ; Set RA<2:0> as inputs, RA<4:3> as outputs,
                         ;    TRISA<7:5> always read '0'
   BCF     STATUS,RP0    ; Select Bank0
   CALL    DELAY 10      ; 10µs delay
   MOVF    CMCON,F       ; Read    CMCON to end change condition
   BCF     PIR1,CMIF     ; Clear pending interrupts
   BSF     STATUS,RP0    ; Select Bank1
   BSF     PIE1,CMIE     ; Enable comparator interrupts
   BCF     STATUS,RP0    ; Select Bank0
   BSF     INTCON,PEIE   ; Enable peripheral interrupts
   BSF     INTCON,GIE    ; Global interrupt enable
```

20.13 设计技巧

问 1： *我的程序好象锁死了。*

答 1：

可能是因为你没有按照正确的顺序来清除CMIF标志位，而陷入比较器中断服务程序的无限循环。正确顺序应该是先读取 CMCON 寄存器的内容，然后再清除 CMIF 标志位。

20.14 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 （即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制，即可使用。目前与比较器模块相关的应用笔记有：

标题	应用笔记 #
Resistance and Capacitance Meter using a PIC16C622	AN611

20.15 版本历史

版本 A

这是描述比较器模块的初始发行版。

第 21 章 8 位 A/D 转换器

目录

本章包括以下一些主要内容：

21.1 简介 21-2
21.2 控制寄存器 21-3
21.3 操作 21-5
21.4 A/D 采集时间要求 21-6
21.5 A/D 转换时钟的选择 21-8
21.6 配置模拟输入端口 21-9
21.7 A/D 转换 21-10
21.8 休眠期间的 A/D 转换 21-12
21.9 A/D 精度 / 误差 21-13
21.10 复位对 A/D 转换的影响 21-13
21.11 CCP 触发器的使用 21-14
21.12 连接注意事项 21-14
21.13 传递函数 21-14
21.14 初始化 21-15
21.15 设计技巧 21-16
21.16 相关应用笔记 21-17
21.17 版本历史 21-18

注：	器件是否具有该模块，请参考附录 C.3 或器件数据手册。

21.1 简介

此模数转换器 (A/D) 模块有多达 8 个模拟输入通道。

A/D 转换器能将一个模拟输入信号转换成相应的 8 位数字信号。采样保持输出是转换器的输入，A/D 转换器采用逐次逼近法产生转换结果。通过软件设置，模拟参考电压可以选择为器件的正向电源电压 (VDD) 或 VREF 引脚上的电平。 A/D 转换器具备可在休眠状态下工作的独特特性。

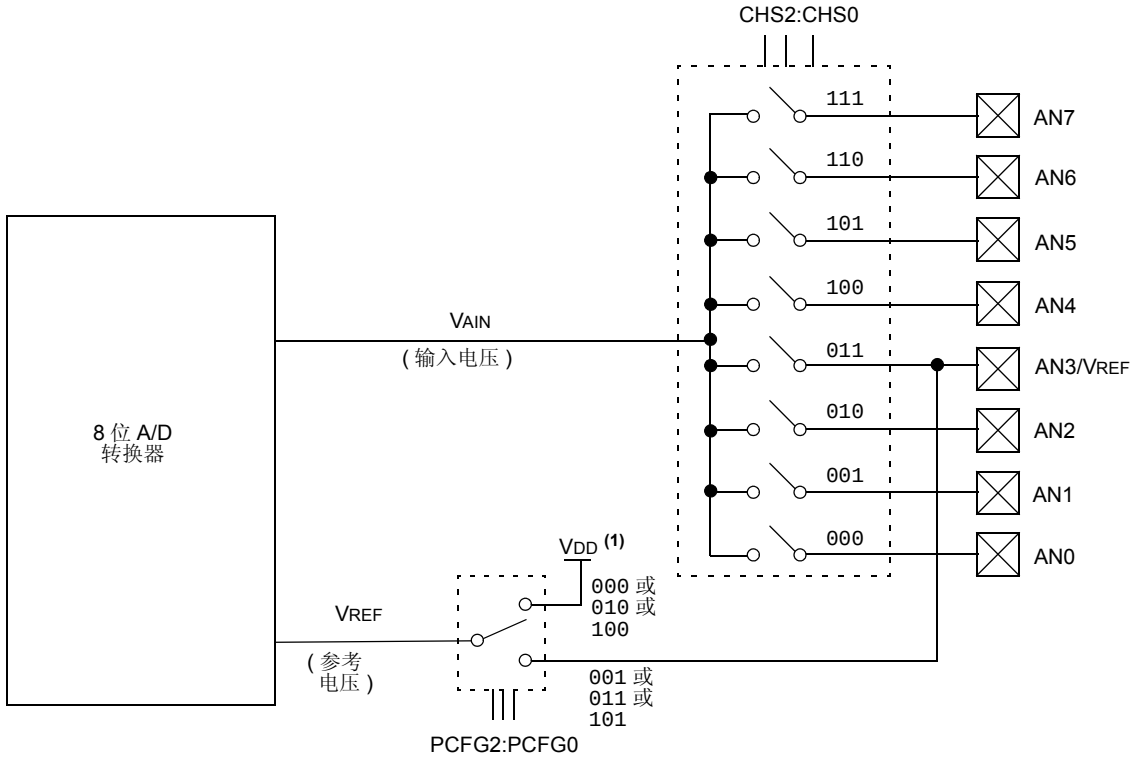
A/D 转换器有 3 个寄存器，它们是：

- A/D 结果寄存器 (ADRES)
- A/D 控制寄存器 0 (ADCON0)
- A/D 控制寄存器 1 (ADCON1)

ADCON0 寄存器，如图 21-1 所示，控制 A/D 模块的操作。ADCON1 寄存器，如图 21-2 所示，可对端口的引脚功能进行配置。这些 I/O 引脚可被配置成模拟输入 (其中一个 I/O 也可作为模拟参考电压) 或数字 I/O 口。

A/D 模块结构框图如图 21-1 所示。

图 21-1：8 位 A/D 转换器结构图



注 (1)：在某些器件上，该引脚是一个被称为 AVDD 的独立引脚。这样可允许 A/D 转换器的 VDD 连接到一个精密电压源上。

21.2 控制寄存器

寄存器 21-1: ADCON0 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
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	Resv	ADON
bit 7							bit 0

bit 7:6 **ADCS1:ADCS0**：A/D 转换时钟选择位
00 = FOSC/2
01 = FOSC/8
10 = FOSC/32
11 = FRC (来自内部 A/D 的 RC 振荡器的时钟)

bit 5:3 **CHS2:CHS0**：模拟通道选择位

000 = 通道 0 (AN0)
001 = 通道 1 (AN1)
010 = 通道 2 (AN2)
011 = 通道 3 (AN3)
100 = 通道 4 (AN4)
101 = 通道 5 (AN5)
110 = 通道 6 (AN6)
111 = 通道 7 (AN7)

注： 对未用满 8 个 A/D 通道的器件，未使用的选项被保留。不要选择未使用的通道。

bit 2 **GO/$\overline{\text{DONE}}$**：A/D 转换状态位

当 ADON = 1 时

1 = A/D 转换正在进行
(该位置 1 启动 A/D 转换。 A/D 转换结束后该位由硬件自动清零)
0 = 未进行 A/D 转换

bit 1 **保留**：总是保持该位为 0。

bit 0 **ADON**：A/D 模块开启位

1 = A/D 转换器模块工作
0 = A/D 转换器关闭，不消耗工作电流

图注		
R = 可读位	W = 可写位	
U = 未用，读为 0		- n = 上电复位时的值

寄存器 21-2: ADCON1 寄存器

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7:3 **未用：**读为 0

bit 2:0 **PCFG2:PCFG0**: A/D 端口配置控制位

PCFG2:PCFG0	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
000	A	A	A	A	A	A	A	A
001	A	A	A	A	VREF	A	A	A
010	D	D	D	A	A	A	A	A
011	D	D	A	A	VREF	A	A	A
100	D	D	D	D	A	D	A	A
101	D	D	D	D	VREF	D	A	A
11x	D	D	D	D	D	D	D	D

A = 模拟输入　　D = 数字 I/O

注： 当 AN3 被选作 VREF 时，A/D 的参考电压为 AN3 引脚的电压。当 AN3 被选作模拟输入 (A) 时， A/D 的参考电压为器件的 VDD。

图注

R = 可读位　　W = 可写位

U = 未用，读为 0　　- n = 上电复位时的值

注 1: 在器件的复位时，复用为模拟功能 (ANx) 的端口引脚均被强制置为模拟输入。

21.3 操作

当 A/D 转换完成之后，转换结果被载入 ADRES 寄存器，GO/$\overline{\text{DONE}}$ (ADCON0<2>) 位被清零，且 A/D 中断标志位 ADIF 置 1。

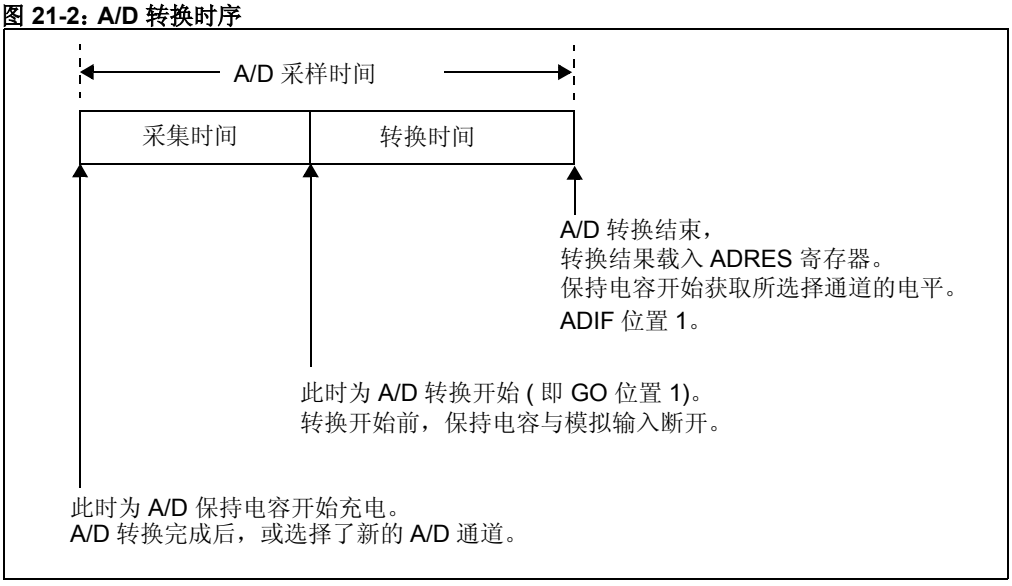
当配置好 A/D 模块后，在启动转换前必须先选择 A/D 转换的通道。模拟输入通道的相应 TRIS 位必须设置为输入。采集时间 （acquisition time）的确定参见 **21.4 “A/D 采集时间要求”小节**。在这一采集时间过去之后， A/D 转换即可开始。按照以下步骤进行 A/D 转换：

1. 配置 A/D 模块
   - 对模拟引脚 / 参考电压 / 数字 I/O (ADCON1) 进行配置
   - 选择 A/D 输入通道 (ADCON0)
   - 选择 A/D 转换时钟 (ADCON0)
   - 打开 A/D 转换模块 (ADCON0)
2. 需要时，设置 A/D 中断
   - 将 ADIF 位清零
   - 将 ADIE 位置 1
   - 将 GIE 位置 1
3. 等待所需的采集时间
4. 启动 A/D 转换
   - 将 GO/$\overline{\text{DONE}}$ 置 1 (ADCON0)
5. 等待 A/D 转换完成，通过以下两种方法之一可判断转换是否完成：
   - 查询 GO/$\overline{\text{DONE}}$ 位是否被清零；

   或
   - 等待 A/D 转换的中断。
6. 读取 A/D 结果寄存器 (ADRES)，需要时将 ADIF 位清零。
7. 要再次进行 A/D 转换，根据要求转入步骤 1 或步骤 2。每一位的 A/D 转换时间定义为 T_{AD}。在下一次采集开始前至少需要等待 $2T_{AD}$。

图 21-2 为 A/D 转换顺序及所使用的术语。采集时间是 A/D 模块的保持电容连接到外部电平的时间。随后是 10 T_{AD} 的转换时间，开始于 GO 位被置 1。这两段时间的总和即采样时间（sampling time）。为确保保持电容充电至适当电平以使 A/D 转换达到所需精度，应保证一个最小采集时间。

图 21-2：A/D 转换时序



21.4 A/D 采集时间要求

为了使 A/D 转换达到规定精度，必须让充电保持电容 (***CHOLD***) 充满至输入通道的电平。图 21-3 显示了模拟输入模型。模拟信号的源阻抗 (***RS***) 和内部采样开关阻抗 (***RSS***) 直接影响电容器 CHOLD 所需的充电时间。采样开关 (***RSS***) 电阻随器件电压 (VDD) 变化，见图 21-3。**模拟信号源的最大推荐阻抗为 10 kΩ**。选择（改变）模拟输入通道后，在转换开始前必须先完成模拟信号的采集。

要计算最小采集时间，可使用公式 21-1。该公式假设误差为 1/2 LSb(即 A/D 的 512 步)。1/2 LSb 误差是 A/D 模块达到规定分辨率的最大允许误差。

公式 21-1： 采集时间

TACQ	=	放大器的建立时间 + 保持电容充电时间 + 温度系数
	=	TAMP + TC + TCOFF

公式 21-2： A/D 最小充电时间

$$V_{HOLD} = (V_{REF} - (V_{REF}/512)) \cdot (1 - e^{(-Tc/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

或

$$Tc = -(51.2\ pF)(1\ k\Omega + R_{SS} + R_S)\ \ln(1/511)$$

例 21-1 显示了所需最小采集时间 TACQ 的计算过程。该计算过程基于以下假设：

Rs	=	10 kΩ	
转换误差	≤	1/2 LSb	
VDD	=	5V → Rss = 7 kΩ	(参见例 21-3)
温度	=	50°C （系统最大值）	
VHOLD	=	0V （时间 = 0 时）	

例 21-1: 计算所需最小采集时间

TACQ = TAMP + TC + TCOFF

TACQ = 5 µs + Tc + [(Temp - 25°C)(0.05 µs/°C)]

TC = -CHOLD (RIC + RSS + RS) ln(1/512)
-51.2 pF (1 kΩ + 7 kΩ + 10 kΩ) ln(0.0020)
-51.2 pF (18 kΩ) ln(0.0020)
-0.921 µs (-6.2146)
5.724 µs

TACQ = 5 µs + 5.724 µs + [(50°C - 25°C)(0.05 µs/°C)]
10.724 µs + 1.25 µs
11.974 µs

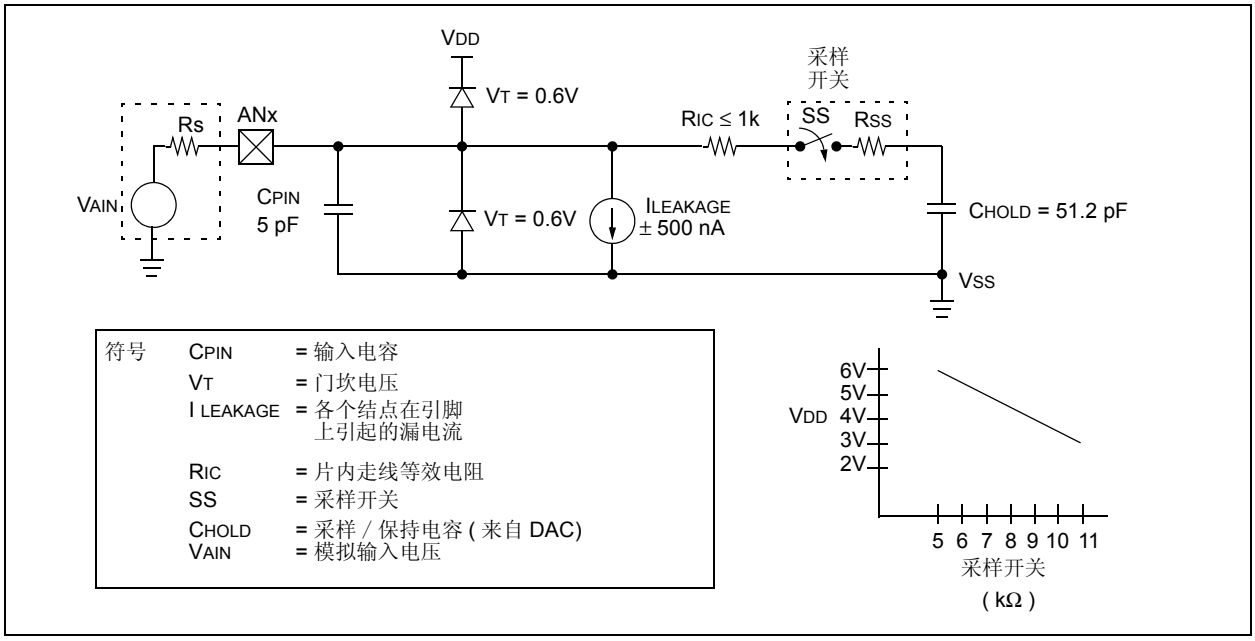
注 1: 参考电压 (VREF) 对该公式不产生影响，因为它在计算中已将自身消去。

注 2: 在每次转换后，充电保持电容 (CHOLD) 并不放电。

注 3: 模拟信号源的最大建议阻抗为 10 kΩ，这是为了满足引脚漏电流的要求。

注 4: 在转换完成之后，下一次采集重新开始前应等待 2.0 TAD。在此期间，保持电容并不与所选 A/D 输入通道相连接。

图 21-3：模拟输入模型



21.5 A/D 转换时钟的选择

每一位的 A/D 转换时间被定义为 TAD。每完成一次 8 位 A/D 转换需要 9.5 TAD。 A/D 转换的时钟源可用软件进行选择。 TAD 的 4 种选项为：

- 2TOSC
- 8TOSC
- 32TOSC
- 内部 RC 振荡器

为了确保 A/D 转换正确，所有器件的 A/D 转换时钟 (TAD) 的选择必须满足最小 1.6 µs 的 TAD 时间，参见器件电气规范的参数 130。

表 21-1 和 表 21-2 显示了器件在不同工作频率下以及所选的不同 A/D 时钟源下得到的 TAD 结果。

表 21-1: TAD 与器件工作频率关系表 (对于标准型 C 器件)

AD 时钟源 (TAD)		器件工作频率			
工作于	ADCS1:ADCS0	20 MHz	5 MHz	1.25 MHz	333.33 kHz
2TOSC	00	100 ns(2)	400 ns(2)	1.6 µs	6 µs
8TOSC	01	400 ns(2)	1.6 µs	6.4 µs	24 µs(3)
32TOSC	10	1.6 µs	6.4 µs	25.6 µs(3)	96 µs(3)
RC	11	2 - 6 µs(1,4)	2 - 6 µs(1,4)	2 - 6 µs(1,4)	2 - 6 µs(1)

图注： 阴影部分不在推荐工作范围内。
注 1： RC 时钟源的典型 TAD 为 4 µs。
2： 这些值违反了所需最小 TAD 时间。
3： 要加快转换时间，建议选择另一时钟源。
4： 器件工作频率高于 1 MHz 时，整个转换过程应在休眠模式下进行，否则 A/D 转换精度可能超出允许范围。

表 21-2: TAD 与器件工作频率关系表 (对于扩展型 LC 器件)

AD 时钟源 (TAD)		器件工作频率			
工作于	ADCS1:ADCS0	4 MHz	2 MHz	1.25 MHz	333.33 kHz
2TOSC	00	500 ns(2)	1.0 µs(2)	1.6 µs(2)	6 µs
8TOSC	01	2.0 µs(2)	4.0 µs	6.4 µs	24 µs(3)
32TOSC	10	8.0 µs	16.0 µs	25.6 µs(3)	96 µs(3)
RC	11	3 - 9 µs(1,4)	3 - 9 µs(1,4)	3 - 9 µs(1,4)	3 - 9 µs(1)

图注： 阴影部分不在推荐工作范围内。
注 1： RC 时钟源的典型 TAD 为 6 µs。
2： 这些值低于要求的最小 TAD 时间。
3： 要加快转换时间，建议选择另一时钟源。
4： 器件工作频率高于 1 MHz 时，整个转换过程应在休眠模式下进行，否则 A/D 转换精度可能超出允许范围。

21.6 配置模拟输入端口

ADCON1 和相应的 TRIS 寄存器用来控制 A/D 端口引脚的运行。若希望端口引脚为模拟输入，则必须将其相应的 TRIS 位置 1(输入)；如果 TRIS 位被清零（输出），则数字输出电平 （VOH 或 VOL) 将被转换。

A/D 转换与 CHS2:CHS0 位及 TRIS 位的状态无关。

注 1:	读取端口寄存器时，所有配置为模拟输入通道的引脚均读为 0(低电平)。配置为数字输入的引脚将转换模拟输入信号。配置为数字输入的引脚上的模拟电平将不影响转换精度。
注 2:	定义为数字输入的引脚上的模拟电平 (包括 AN7:AN0 引脚)，可能导致输入缓冲器消耗超出器件规范的电流。

21.7 A/D 转换

例 21-2 显示了如何进行 A/D 转换。I/O 引脚被配置成模拟输入。模拟参考电压 (VREF) 为器件电压 VDD。使能 A/D 中断， A/D 转换时钟设为 FRC。该转换在 AN0 通道上进行。

注： 由于所需采集时间的要求，不应在打开 A/D 模块的同一指令中将 GO/$\overline{\text{DONE}}$ 位置 1。

在转换期间将 GO/$\overline{\text{DONE}}$ 位清零将中止当前 A/D 转换。ADRES 寄存器中的内容不会被部分完成的 A/D 转换样本所更新，即， ADRES 寄存器仍然保持上一次转换完成后的结果 (或上一次写入 ADRES 寄存器中的值)。A/D 转换被中止后，在下一次采集开始前，需要等待 2TAD 的时间。等待 2TAD 之后，采集将在所选通道上自动开始。

例 21-2: 进行一次 A/D 转换

```
    BSF     STATUS, RP0       ; Select Bank1
    CLRF    ADCON1            ; Configure A/D inputs
    BSF     PIE1,   ADIE      ; Enable A/D interrupts
    BCF     STATUS, RP0       ; Select Bank0
    MOVLW   0xC1              ; RC Clock, A/D is on, Channel 0 is selected
    MOVWF   ADCON0            ;
    BCF     PIR1,   ADIF      ; Clear A/D interrupt flag bit
    BSF     INTCON, PEIE      ; Enable peripheral interrupts
    BSF     INTCON, GIE       ; Enable all interrupts
;
;  Ensure that the required sampling time for the selected input
;  channel has elapsed. Then the conversion may be started.
;
    BSF     ADCON0, GO        ; Start A/D Conversion
      :                       ; The ADIF bit will be set and the GO/DONE
      :                       ; bit is cleared upon completion of the
      :                       ;     A/D Conversion.
```

图 21-4：A/D 转换中的 TAD 周期

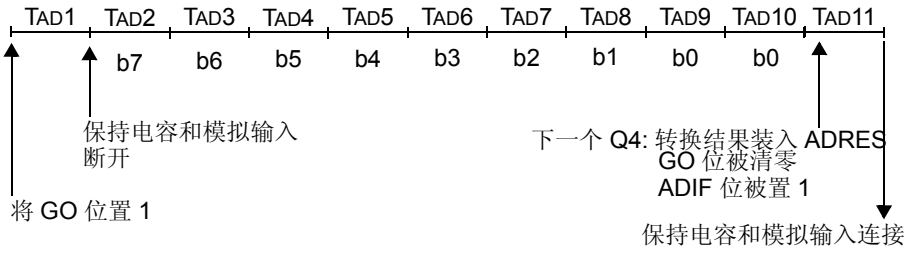
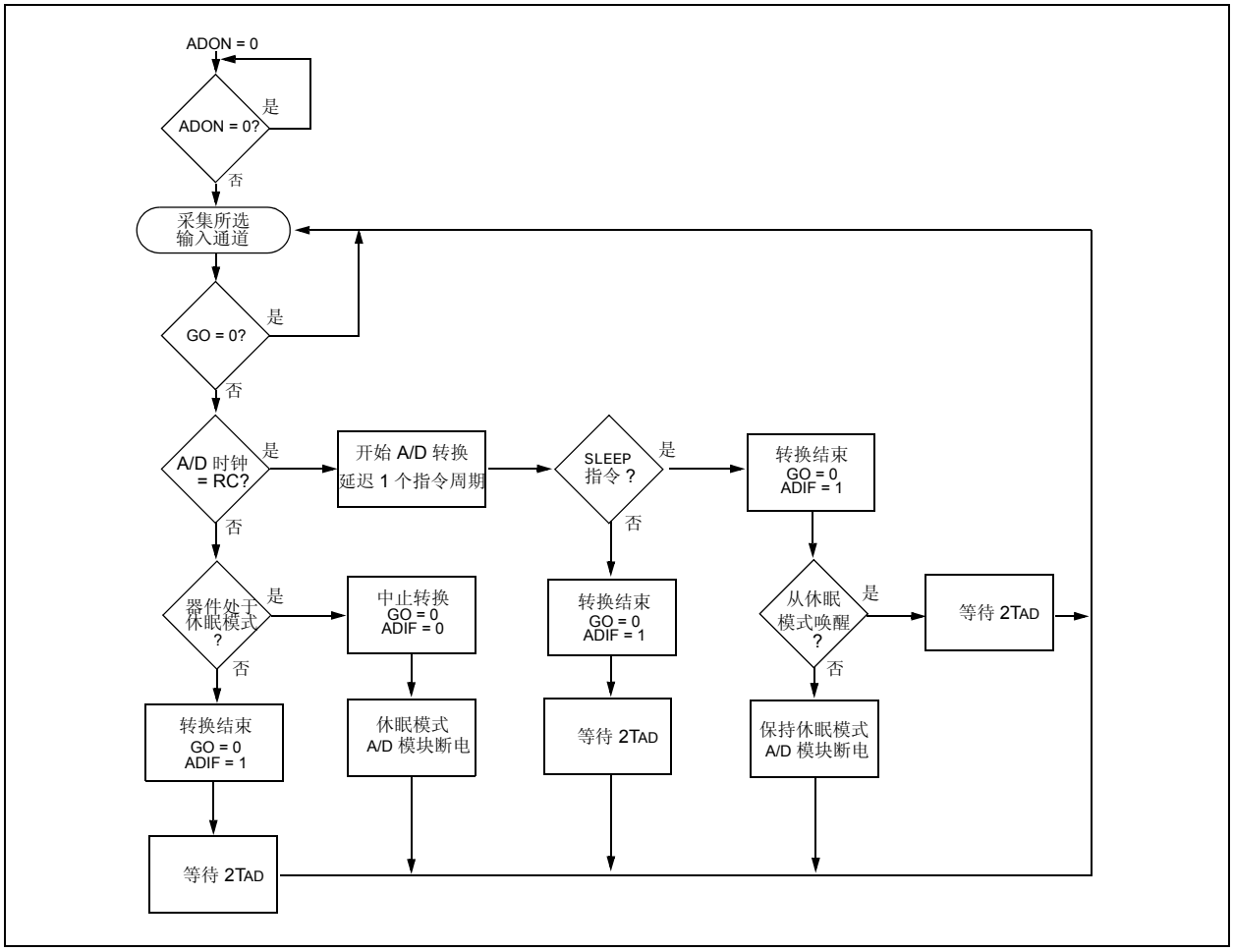


图 21-5：A/D 工作流程图



21.7.1 加快转换速度与降低转换分辨率的权衡

并非所有的应用都需要 8 位分辨率的转换结果，相反，它们可能需要更快的转换时间。A/D 模块允许用户降低转换分辨率以换取转换速度。无论所需的分辨率如何，采集时间都是相同的。为了加快转换速度，可切换 A/D 模块的时钟源，以使 TAD 违反规定的最小时间 (参见电气规范的有关说明)。一旦 TAD 违反了规定最小时间，所有接下来的 A/D 转换结果位将不再有效 (参见电气规范章节中 有关 A/D 转换时间的说明)。时钟源只能在三种振荡器间切换 (不能在振荡器模式和 RC 模式间相互切换)。用以下公式确定须经过多长时间才可切换振荡器：

转换时间 = TAD + N • TAD + (10 - N)(2TOSC)
其中 N = 所需分辨率的位数

由于 TAD 基于器件振荡器，用户必须使用一些方法 (如定时器，软件循环等) 以决定何时切换 A/D 振荡器。例 21-3 显示了 4 位分辨率与 8 位分辨率转换时间的对照。该例中器件的工作频率为 20 MHz (A/D 转换时钟设为 32TOSC)，并假定 A/D 时钟在紧随 5TAD 之后被设为 2TOSC。

由于后 4 位将无法正确转换，因此 2TOSC 违反了最小 TAD 时间。

例 21-3: 4 位和 8 位转换时间

	频率 (MHz)(1)	分辨率	
		4 位	8 位
TAD	20	1.6 µs	1.6 µs
TOSC	20	50 ns	50 ns
TAD + N • TAD + (10 - N)(2TOSC)	20	8.6 µs	17.6 µs

注 1： 要求最小 TAD 时间为 1.6 µs
2： 若需全部 8 位转换结果，则不能切换 A/D 时钟源。

21.8 休眠期间的 A/D 转换

A/D 模块可在休眠期间运行，这需要把 A/D 的时钟源设置成 RC 方式 (ADCS1:ADCS0 = 11)。选择了 RC 时钟源后，A/D 模块等待一个指令周期后才开始转换。这样使 SLEEP 指令得以执行，从而消除了转换时所有内部的数字开关噪声。A/D 转换完成后，GO/DONE 位清零，转换结果送入 ADRES 寄存器。如果 A/D 中断被使能，器件将从 SLEEP 唤醒。如果 A/D 中断被禁止， A/D 模块将被关闭 (以节省功耗)，尽管此时 ADON 位保持置 1 状态。

如果 A/D 时钟源为另一时钟选项 (非 RC)，执行一条 SLEEP 指令将中止当前 A/D 转换并关闭 A/D 模块，尽管此时 ADON 位保持置 1 状态。

将 A/D 关闭将 A/D 模块置于电流消耗最小的状态。

注： 要使 A/D 模块在 SLEEP 模式下运行， A/D 时钟源必须被设置成 RC 模式 (ADCS1:ADCS0 = 11)。要在 SLEEP 下进行 A/D 转换，必须将 GO/DONE 位置 1，然后执行 SLEEP 指令。

21.9 A/D 精度 / 误差

在器件频率较低的系统中，最好使用 A/D 模块的 RC 时钟；在中高频时，TAD 应来源于器件的振荡器。

A/D 转换器的绝对精度参数包括量化误差、积分误差、微分误差、满量程误差、偏移误差和单一性等所有误差的总和。它定义为任意数码的实际转换值和理想转换值之间的最大偏差。 当 VDD = VREF 时，A/D 转换器（在器件的规定工作范围内）的绝对误差为 < ±1 LSb ； 然而，当 VDD 偏离 VREF 时， A/D 转换器的精度将下降。

在一个给定的模拟输入范围内， A/D 的输出数字数码是相同的，这是因为模拟输入被量化到数码了。典型量化误差为 ± 1/2 LSb，并存在于整个模拟数字转换过程中。减小量化误差的唯一方法是提高 A/D 转换器的分辨率。

偏移误差是首个实际数码转换电平与首个理想数码转换电平的差值。偏移误差使整个传递函数发生平移。通过模拟输入端的总漏电流和源阻抗的相互作用，偏移误差可在系统外校准或引入系统。

增益误差是指经过偏移误差调整后，末次实际转换电平与末次理想转换电平之间的最大偏差。增益误差显示为传递函数的斜率变化。增益误差和满量程误差的区别在于满量程误差不考虑偏移误差。增益误差可通过软件校正以从系统中消除。

线性误差是指数码一致性的变化。线性误差不能从系统中校准。积分非线性误差是指经过增益误差调整后，各个输出数码的实际转换电平和理想转换电平之间的差值。

微分非线性误差是指最大实际数码宽度和理想数码宽度之间的差值，该误差无法校正。

引脚的最大漏电流在器件数据手册的电气规范参数 D060 中作了规定。

在器件频率较低的系统中，最好使用 A/D 模块的 RC 时钟。在中高频率时， TAD 应来源于器件的振荡器。 TAD 不得违反最小时间，且应最大限度地降低以减小由噪声和采样保持电容器放电造成的误差。

在 A/D 转换开始后器件就进入休眠模式的系统中，必须选择 RC 时钟源。在这种模式下，消除了模块的数字噪声。这种方法能提供较好的转换精度。

21.10 复位对 A/D 转换的影响

器件复位强制所有寄存器为复位状态，同时强制 A/D 模块关闭并中止任何正在进行的转换。上电复位时， ADRES 寄存器中的值保持不变。上电复位后 ADRES 寄存器中的值不确定。

21.11 CCP 触发器的使用

CCP 模块的“特殊事件触发器”可以启动 A/D 转换。要求将 CCPxM3:CCPxM0 位（CCPxCON<3:0>）设置为 1011，且使能 A/D 模块 (ADON 位置 1)。触发时， GO/DONE 位被置 1，启动 A/D 转换，Timer1 计数值被复位为 0。复位 Timer1 可实现 A/D 采集周期的自动重复，最大限度地降低了软件开销 (将 ADRES 移到所需位置)。在“特殊事件触发器”将 GO/DONE 位置 1(启动转换) 前，必须正确选择模拟输入通道，并要保证最小采集时间。

如果 A/D 模块未被使能 (ADON 清零)，则“特殊事件触发器”将被 A/D 模块忽略，但它仍会将 Timer1 计数器复位。

21.12 连接注意事项

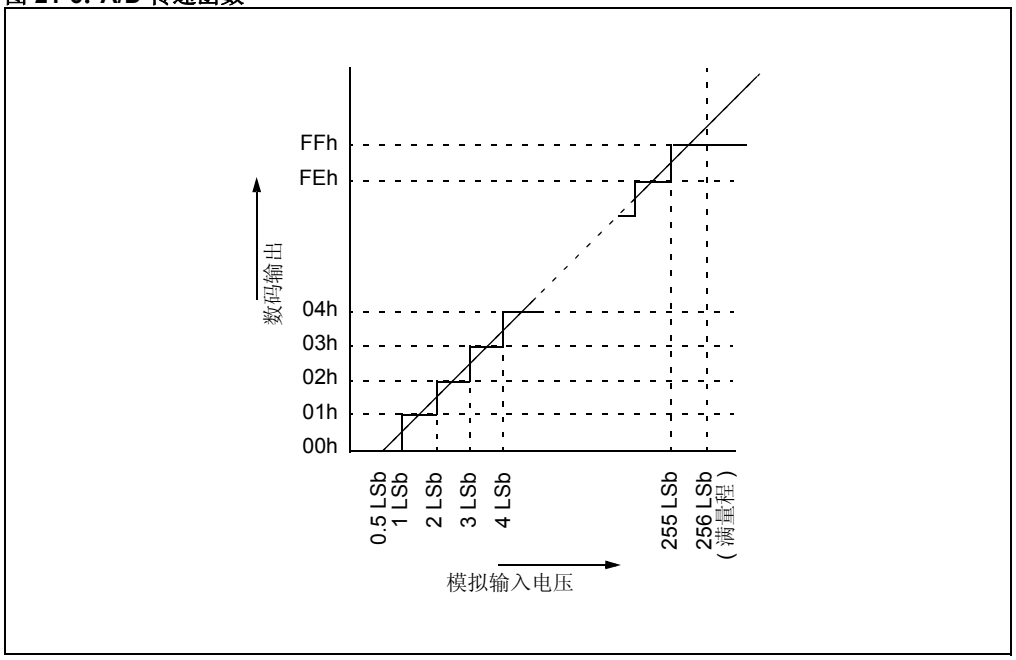
如果输入电压超出满幅电压值 (VSS 或 VDD)0.3V，转换精度将超出规定范围。

有时可增加一个输入信号抗混叠外部 RC 滤波器。选择的 R 元件应确保总源阻抗小于建议值 10 kΩ。任何通过高阻抗连接到模拟输入引脚上的外部元件（如电容器、齐纳二极管等)，应使其在引脚上的漏电流很小。

21.13 传递函数

以下是 A/D 转换器的理想传递函数：第一次转换发生在模拟输入电压 (VAIN) 为 1 LSb （或模拟 VREF / 256）时 (见图 21-6)。

图 21-6：A/D 传递函数



21.14 初始化

例 21-4 显示了 PIC16C74A 的 A/D 模块的初始化。

例 21-4: A/D 初始化 (PIC16C74A)

```
    BSF     STATUS, RP0      ; Select Bank1
    CLRF    ADCON1           ; Configure A/D inputs
    BSF     PIE1, ADIE       ; Enable A/D interrupts
    BCF     STATUS, RP0      ; Select Bank0
    MOVLW   0xC1             ; RC Clock, A/D is on, Channel 0 is selected
    MOVWF   ADCON0           ;
    BCF     PIR1, ADIF       ; Clear A/D interrupt flag bit
    BSF     INTCON, PEIE     ; Enable peripheral interrupts
    BSF     INTCON, GIE      ; Enable all interrupts
;
;  Ensure that the required sampling time for the selected input
;  channel has elapsed. Then the conversion may be started.
;
    BSF     ADCON0, GO       ; Start A/D Conversion
      :                      ; The ADIF bit will be set and the GO/DONE
      :                      ; bit is cleared upon completion of the
      :                      ;     A/D Conversion.
```

21.15 设计技巧

问 1: *在使用 PIC16C7X 时，我发现模拟数字转换结果并不总是准确的。如何提高转换精度？*

答 1:

1. 请确保您满足了所有时序规范要求。如果你关闭 A/D 模块后再打开，应等待一个最小延迟时间后再开始采样；如果改变了输入通道，同样需要等待一个最小延迟时间后；最后是 TAD，它是所选择的每一位的转换时间。TAD 在 ADCON0 中选择，其值应该处于 2 到 6μs 之间。如果 TAD 太短，转换终止时尚未对数据进行完全转换，而如果 TAD 过长，采样电容上的电压会在转换结束前下降过大。数据手册提供了这些计时参数的表格或公式，您应根据具体器件与具体情况进行查找。
2. 模拟输入信号的源阻抗经常很高 (大于 1kΩ)，因此为采样电容充电的信号源的输出电流会影响精度。如果输入信号并不快速变化，可以尝试在模拟输入端连接一个 0.1 μF 的电容。该电容可充电到所采样的模拟电压，并为内部 51.2 pf 的保持电容提供所需的瞬时充电电流。
3. 最后，直接参见数据手册的内容：“在器件工作频率较低的系统中，最好使用来自器件振荡器的 A/D 时钟……这将在很大程度上降低数字开关噪声的影响”，以及“在 A/D 转换开始后器件就进入休眠模式的系统中，必须选择 RC 时钟源。这种方法可得到最高的转换精度。”

问 2: *在 A/D 转换开始后是否可以改变输出通道 (以便进行下一次转换)？*

答 2:

在保持电容从输入通道断开后， GO 位置 1 后经过一个 TAD 时间，就可以改变输入通道。

问 3: *请问有没有关于 A/D 的较好的参考书？*

答 3:

“Analog-Digital Conversion Handbook”是便于理解 A/D 转换的较好的参考书，该书由 Prentice Hall 出版 (ISBN 0-13-03-2848-0)。

21.16 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，经过适当修改并受到一定限制后即可使用。目前与 8 位 A/D 模块相关的应用笔记有：

标题	应用笔记 #
Using the Analog to Digital Converter	AN546
Four Channel Digital Voltmeter with Display and Keyboard	AN557

21.17 版本历史

版本 A

这是描述 8 位 A/D 转换器模块的初始发行版。

第 22 章 基本型 8 位 A/D 转换器

目录

本章包括以下一些主要内容：

22.1 简介 22-2
22.2 控制寄存器 22-3
22.3 A/D 采集时间要求 22-6
22.4 A/D 转换时钟的选择 22-8
22.5 配置模拟输入端口 22-10
22.6 A/D 转换 22-11
22.7 休眠期间的 A/D 转换 22-14
22.8 A/D 转换精度 / 误差 22-15
22.9 复位对 A/D 转换的影响 22-16
22.10 连接时的考虑事项 22-16
22.11 传递函数 22-16
22.12 初始化 22-17
22.13 设计技巧 22-18
22.14 相关应用笔记 22-19
22.15 版本历史 22-20

注： 器件是否具有该模块，请参考附录 C.2 或者器件数据手册。

22.1 简介

这一模数转换器 (A/D) 模块总共有 4 个模拟输入通道。

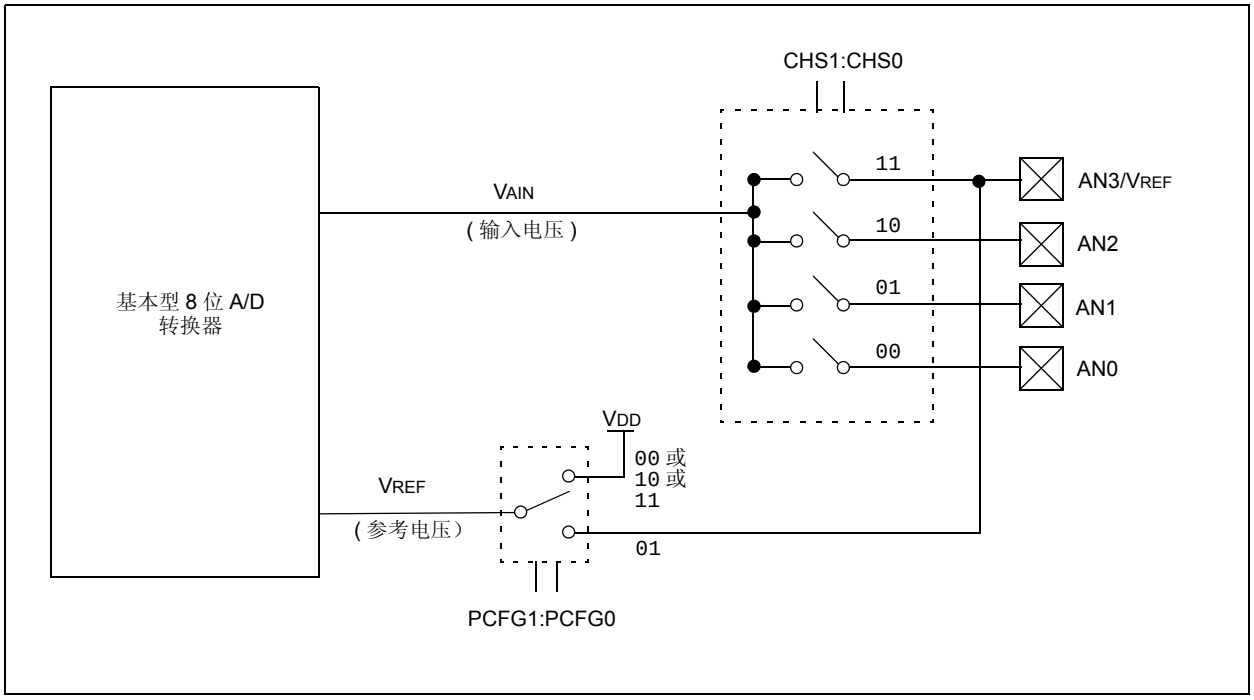
A/D 转换器能将一个模拟输入信号转换成相应的 8 位数字信号。采样保持输出是转换器的输入，A/D 转换器采用逐次逼近法产生转换结果。通过软件设置，模拟参考电压可以选择为器件的正向电源电压 (V_{DD}) 或 AN3/V_{REF} 引脚上的电平。 A/D 转换器具备可在休眠状态下工作的独特特性。

A/D 转换器有 3 个寄存器，它们是：

- A/D 结果寄存器 (ADRES)
- A/D 控制寄存器 0 (ADCON0)
- A/D 控制寄存器 1 (ADCON1)

ADCON0 寄存器，如图 22-1 所示，控制 A/D 模块的操作。ADCON1 寄存器，如图 22-2 所示，可对端口的引脚功能进行配置。这些端口可被配置成模拟输入 (或作为参考电压) 或数字 I/O 口。

图 22-1：基本型 8 位 A/D 转换器结构图



22.2 控制寄存器

寄存器 22-1: ADCON0 寄存器

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS1	ADCS0	— (1)	CHS1	CHS0	GO/$\overline{\text{DONE}}$	ADIF / — (2)	ADON
bit 7							bit 0

bit 7:6 **ADCS1:ADCS0**：A/D 转换时钟选择位
00 = FOSC/2
01 = FOSC/8
10 = FOSC/32
11 = FRC (来自 A/D 模块内部 RC 振荡器的时钟)

bit 5 **未用**：读为 0。

bit 4:3 **CHS1:CHS0**：模拟通道选择位
00 = 通道 0 (AN0)
01 = 通道 1 (AN1)
10 = 通道 2 (AN2)
11 = 通道 3 (AN3)

bit 2 **GO/$\overline{\text{DONE}}$:** A/D 转换状态位
如果 ADON = 1
1 = A/D 转换正在进行 (该位置 1 启动 A/D 转换)
0 = 未进行 A/D 转换 (在 A/D 转换完成后，该位自动被硬件清零)

bit 1 **ADIF (2):** A/D 转换完成中断标志位
1 = 转换完成 (必须用软件清零)
0 = 转换没有完成

bit 0 **ADON:** A/D 模块开启位
1 = A/D 转换器模块工作
0 = A/D 转换器关闭，不消耗工作电流

图注	
R = 可读位	W = 可写位
U = 未使用，读为 0	- n = 上电复位值

注 1: 对于 PIC16C71, ADCON0 的 bit5 是通用读写位。对于 PIC16C710/711/715，该位未使用，读为 0。

注 2: 对于 PIC12CXXX 器件，该位被保留。ADIF 位包含在 PIR 寄存器中。不建议将该位作为通用读写位。应将该位始终清零。

寄存器 22-2: ADCON1 寄存器

U-0	U-0	U-0	U-0	U-0	U-0 / R/W-0	R/W-0	R/W-0
—	—	—	—	—	— / PCFG2 (1)	PCFG1	PCFG0
bit 7							bit 0

bit 7:2 **未用：**读为 0

注： 某些器件使用 bit2 作为 PCFG2 位。

bit 1:0 **PCFG1:PCFG0**: A/D 端口配置控制位

PCFG1:PCFG0	AN3	AN2	AN1	AN0
00	A	A	A	A
01	VREF+	A	A	A
10	D	D	A	A
11	D	D	D	D

A = 模拟输入

D = 数字 I/O

注： 当 AN1 被选作 VREF+ 时， A/D 的参考电压为 AN3 引脚上的电压。当 AN3 被选作模拟输入 (A) 时， A/D 的参考电压为器件的 VDD。

bit 2:0 **PCFG2:PCFG0**: A/D 端口配置控制位 (1)

PCFG2:PCFG0	AN3	AN2	AN1	AN0
000	A	A	A	A
001	A	A	VREF+	A
010	D	A	A	A
011	D	A	VREF+	A
100	D	D	A	A
101	D	D	VREF+	A
110	D	D	D	A
111	D	D	D	D

A = 模拟输入

D = 数字输入 / 输出

注： 当 AN1 被选作 VREF+ 时， A/D 的参考电压为 AN1 引脚的电压。当 AN1 被选作模拟输入 (A) 时， A/D 的参考电压为器件的 VDD。

图注

R = 可读位 W = 可写位

U = 未用，读为 0 - n = 上电复位值

注 1: 某些器件增加了一个额外端口配置位 (PCFG2)，允许将最小模拟通道数设置为 1。这对于带有 A/D 转换器的 8 引脚器件最为有用的，因为在 8 引脚器件中，I/O 资源是非常宝贵的。在其它器件中，该位未使用，读为 0。

注 2: 在器件的复位时，复用为模拟功能 (ANx) 的端口引脚均被强制置为模拟输入。

ADRES 寄存器中保存了 A/D 转换的结果。当 A/D 转换完成之后，转换结果被载入 ADRES 寄存器，GO/DONE (ADCON0<2>) 位被清零，且 A/D 中断标志位 ADIF 置 1。A/D 模块的结构框图见图 22-1。

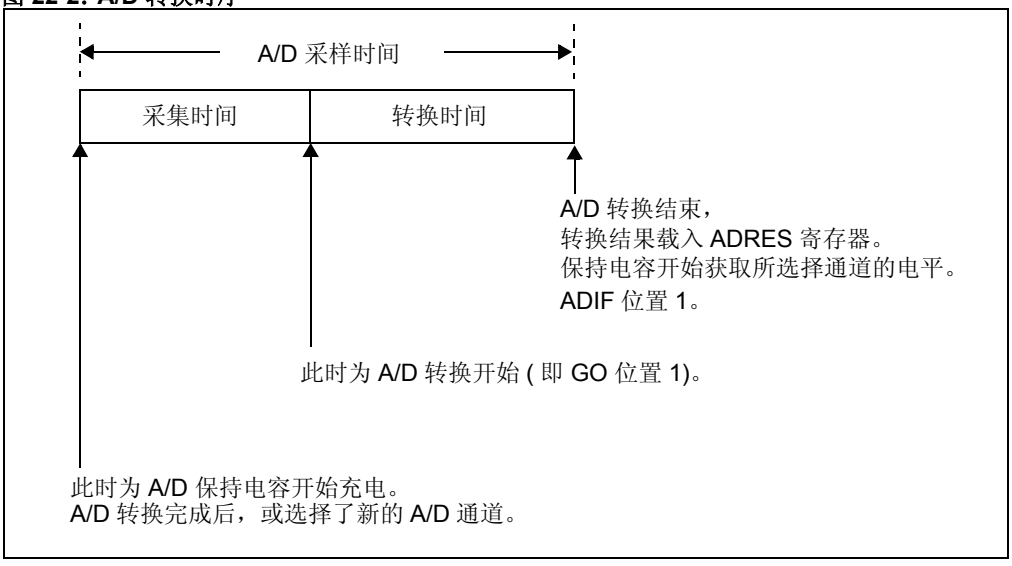
当配置好 A/D 模块后，在启动转换前必须先选择 A/D 转换的通道。模拟输入通道的相应 TRIS 位必须设置为输入。采集时间（acquisition time）的确定参见 **22.3 “A/D 采集时间要求”小节**。在这一采集时间过去之后， A/D 转换即可开始。按照以下步骤进行 A/D 转换：

1. 配置 A/D 模块
   - 对模拟引脚 / 参考电压 / 数字 I/O (ADCON1) 进行配置
   - 选择 A/D 输入通道 (ADCON0)
   - 选择 A/D 转换时钟 (ADCON0)
   - 打开 A/D 转换模块 (ADCON0)
2. 需要时，设置 A/D 中断
   - 将 ADIF 位清零
   - 将 ADIE 位置 1
   - 将 GIE 位置 1
3. 等待所需的采集时间
4. 启动 A/D 转换
   - 将 GO/DONE 置 1 (ADCON0)
5. 等待 A/D 转换完成，通过以下两种方法之一可判断转换是否完成：
   - 查询 GO/DONE 位是否被清零；

   或
   - 等待 A/D 转换的中断。
6. 读取 A/D 结果寄存器 (ADRES)，需要时将 ADIF 位清零。
7. 要再次进行 A/D 转换，根据要求转入步骤 1 或步骤 2。每一位的 A/D 转换时间定义为 TAD。在下一次采集开始前至少需要等待 2TAD。

图 22-2 为 A/D 转换顺序及所使用的术语。采集时间是 A/D 模块的保持电容连接到外部电平的时间。随后是 10 T_{AD} 的转换时间，开始于 GO 位被置 1。这两段时间的总和即采样时间（sampling time）。为确保保持电容充电至适当电平以使 A/D 转换达到所需精度，应保证一个最小采集时间。

图 22-2：A/D 转换时序



22.3 A/D 采集时间要求

为了使 A/D 转换达到规定精度，必须让充电保持电容 (CHOLD) 充满至输入通道的电平。图 22-3 显示了模拟输入模型。模拟信号的源阻抗 (RS) 和内部采样开关阻抗 (***Rss***) 直接影响电容器 CHOLD 所需的充电时间。采样开关 (RSS) 电阻随器件电压 (VDD) 变化，见图 22-3。**模拟信号源的最大建议阻抗为 10 kΩ**。选择（改变）模拟输入通道后，在转换开始前必须先完成模拟信号的采集。

要计算最小采集时间，可使用公式 22-1。该公式假设误差为 1/2 LSb(即 A/D 的 512 步)。1/2 LSb 误差是 A/D 模块达到规定分辨率的最大允许误差。

公式 22-1： 采集时间

TACQ	=	放大器的建立时间 + 保持电容充电时间 + 温度系数
	=	TAMP + TC + TCOFF

公式 22-2： A/D 转换最小充电时间

$$V_{HOLD} = (V_{REF} - (V_{REF}/512)) \cdot (1 - e^{(-Tc/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$$

或

$$Tc = -(51.2\ pF)(1\ k\Omega + R_{SS} + R_S)\ \ln(1/511)$$

例 22-1 显示了所需最小采样时间 TACQ 的计算过程。该计算过程基于以下的假设：

Rs	=	10 kΩ	
转换误差	≤	1/2 LSb	
VDD	=	5V → Rss = 7 kΩ	(参见图 22-3)
温度	=	50°C (系统最大值)	
VHOLD	=	0V （时间 = 0 时）	

例 22-1: 计算所需最小采集时间

TACQ =	TAMP + TC + TCOFF
TACQ =	5 µs + Tc + [(Temp - 25°C)(0.05 µs/°C)]
TC =	-CHOLD (RIC + RSS + RS) ln(1/512) -51.2 pF (1 kΩ + 7 kΩ + 10 kΩ) ln(0.0020) -51.2 pF (18 kΩ) ln(0.0020) -0.921 µs (-6.2146) 5.724 µs
TACQ =	5 µs + 5.724 µs + [(50°C - 25°C)(0.05 µs/°C)] 10.724 µs + 1.25 µs 11.974 µs

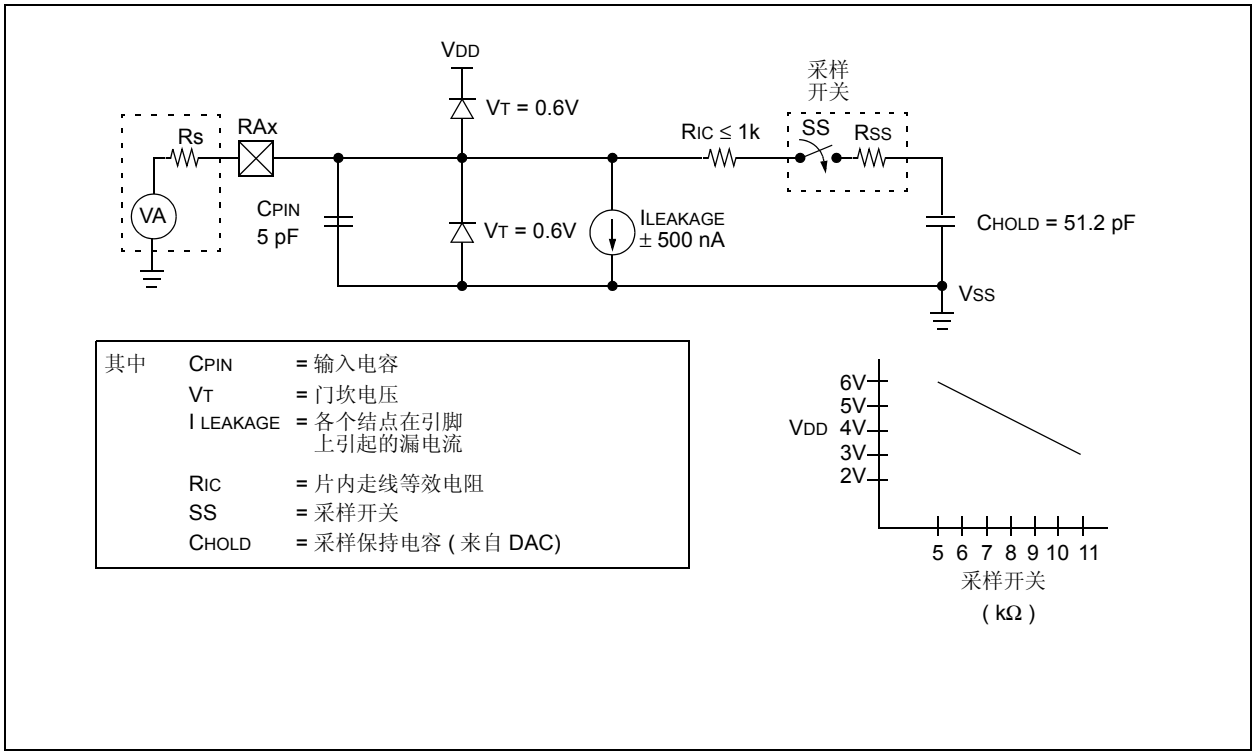
注 1: 参考电压 (VREF) 对该公式不产生影响，因为它在计算中已将自身消去。

注 2: 在每次转换后，充电保持电容 (CHOLD) 并不放电。

注 3: 模拟信号源的最大建议阻抗为 10 kΩ，这是为了满足引脚漏电流的要求。

注 4: 在转换完成之后，下一次采集重新开始前应等待 2.0 TAD。在此期间，保持电容并不与所选 A/D 输入通道相连接。

图 22-3：模拟输入模型



22.4 A/D 转换时钟的选择

每一位的 A/D 转换时间被定义为 TAD。每完成一次 8 位 A/D 转换需要 9.5 TAD。 A/D 转换的时钟源可用软件进行选择。 TAD 的 4 种选项为：

- 2TOSC
- 8TOSC
- 32TOSC
- 内部 RC 振荡器

为了确保 A/D 转换正确， A/D 转换时钟 (TAD) 必须满足最小 TAD 要求：

对于 PIC16C71 最小 TAD 为 2.0 µs，见器件电气规范的参数 130。

对于其他器件最小 TAD 为 1.6 µs，见器件电气规范的参数 130。

表 22-1 到表 22-4 显示了器件在不同工作频率下以及所选的不同A/D时钟源下得到的TAD结果。

表 22-1: TAD 与器件工作频率关系表 (除 PIC16C71 以外的所有 C 型器件)

AD 时钟源 (TAD)		器件工作频率			
操作	ADCS1:ADCS0	20 MHz	5 MHz	1.25 MHz	333.33 kHz
2TOSC	00	100 ns(2)	400 ns(2)	1.6 µs	6 µs
8TOSC	01	400 ns(2)	1.6 µs	6.4 µs	24 µs(3)
32TOSC	10	1.6 µs	6.4 µs	25.6 µs(3)	96 µs(3)
RC(5)	11	2 - 6 µs(1,4)	2 - 6 µs(1,4)	2 - 6 µs(1,4)	2 - 6 µs(1)

注 1： RC 时钟源的典型 TAD 为 4 µs。
2： 这些值违反了最小 TAD 时间要求。
3： 要加快转换时间，建议选择另一时钟源。
4： 器件工作频率高于 1 MHz 时，整个转换过程应在休眠模式下进行，否则 A/D 转换精度可能超出允许范围。

表 22-2: TAD 与器件工作频率关系表 (除 PIC16LC71 以外的所有 LC 型器件)

AD 时钟源 (TAD)		器件工作频率			
操作	ADCS1:ADCS0	4 MHz	2 MHz	1.25 MHz	333.33 kHz
2TOSC	00	500 ns(2)	1.0 µs(2)	1.6 µs(2)	6 µs
8TOSC	01	2.0 µs(2)	4.0 µs	6.4 µs	24 µs(3)
32TOSC	10	8.0 µs	16.0 µs	25.6 µs(3)	96 µs(3)
RC(5)	11	3 - 9 µs(1,4)	3 - 9 µs(1,4)	3 - 9 µs(1,4)	3 - 9 µs(1)

注 1： RC 时钟源的典型 TAD 为 6 µs。
2： 这些值违反了最小 TAD 时间要求。
3： 要加快转换时间，建议选择另一时钟源。
4： 器件工作频率高于 1 MHz 时，整个转换过程应在休眠模式下进行，否则 A/D 转换精度可能超出允许范围。

表 22-3: TAD 与器件 PIC16C71 工作频率关系表 (C 型器件)

AD 时钟源 (TAD)		器件工作频率				
操作	ADCS1:ADCS0	20 MHz	16 MHz	4 MHz	1 MHz	333.33 kHz
2TOSC	00	100 ns(2)	125 ns(2)	500 ns(2)	2.0 μs	6 μs
8TOSC	01	400 ns(2)	500 ns(2)	2.0 μs	8.0 μs	24 μs(3)
32TOSC	10	1.6 μs(2)	2.0 μs	8.0 μs	32.0 μs(3)	96 μs(3)
RC	11	2 - 6 μs(1,4)	2 - 6 μs(1,4)	2 - 6 μs(1,4)	2 - 6 μs(1)	2 - 6 μs(1)

图注： 阴影部分不在推荐工作范围内。

注 1： RC 时钟源的典型 TAD 为 4 μs。

2： 这些值违反了最小 TAD 时间要求。

3： 要加快转换时间，建议选择另一时钟源。

4： 器件工作频率高于 1 MHz 时，整个转换过程应在休眠模式下进行，否则 A/D 转换精度可能超出允许范围。

表 22-4: TAD 与器件 PIC16LC71 工作频率关系表 (LC 型器件)

AD 时钟源 (TAD)		器件工作频率			
操作	ADCS1:ADCS0	4 MHz	2 MHz	1.25 MHz	333.33 kHz
2TOSC	00	500 ns(2)	1.0 μs(2)	1.6 μs(2)	6 μs
8TOSC	01	2.0 μs(2)	4.0 μs	6.4 μs	24 μs(3)
32TOSC	10	8.0 μs	16.0 μs	25.6 μs(3)	96 μs(3)
RC	11	3 - 9 μs(1,4)	3 - 9 μs(1,4)	3 - 9 μs(1,4)	3 - 9 μs(1)

图注： 阴影部分不在推荐工作范围内。

注 1： RC 时钟源的典型 TAD 为 6 μs。

2： 这些值违反了最小 TAD 时间要求。

3： 要加快转换时间，建议选择另一时钟源。

4： 器件工作频率高于 1 MHz 时，整个转换过程应在休眠模式下进行，否则 A/D 转换精度可能超出允许范围。

22.5 配置模拟输入端口

ADCON1 和 TRISA 寄存器用来控制 A/D 端口引脚的运行。若希望端口引脚为模拟输入，则必须将其相应的 TRIS 位置 1(输入) ；如果 TRIS 位被清零 (输出)，则数字输出电平 （VOH 或 VOL) 将被转换。

A/D 转换与 CHS2:CHS0 位及 TRIS 位的状态无关。

注 1:	读取端口寄存器时，所有配置为模拟输入通道的引脚均读为 0 (低电平)。配置为数字输入的引脚将转换模拟输入信号。配置为数字输入的引脚上的模拟电平将不影响转换精度。
注 2:	定义为数字输入的引脚上的模拟电平 (包括 AN3:AN0 引脚)，可能导致输入缓冲器消耗超出器件规范的电流。

22.6 A/D 转换

例 22-2 显示了如何进行 A/D 转换。I/O 引脚被配置成模拟输入。模拟参考电压 (VREF) 为器件电压 VDD。使能 A/D 中断， A/D 转换时钟设为 FRC。该转换在 AN0 通道上进行。

注： 由于所需采集时间的要求，不应在打开 A/D 模块的同一指令中将 GO/$\overline{\text{DONE}}$ 位置 1。

在转换期间将 GO/$\overline{\text{DONE}}$ 位清零将中止当前 A/D 转换。ADRES 寄存器中的内容不会被部分完成的 A/D 转换样本所更新，即， ADRES 寄存器仍然保持上一次转换完成后的结果 (或上一次写入 ADRES 寄存器中的值)。A/D 转换被中止后，在下一次采集开始前，需要等待 2TAD 的时间。等待 2TAD 之后，采集将在所选通道上自动开始。

例 22-2: A/D 转换

```
     BSF     STATUS, RP0    ; Select Bank1
     CLRF    ADCON1         ; Configure A/D inputs
     BCF     STATUS, RP0    ; Select Bank0
     MOVLW   0xC1           ; RC Clock, A/D is on, Channel 0 selected
     MOVWF   ADCON0         ;
     BSF     INTCON, ADIE   ; Enable A/D Interrupt
     BSF     INTCON, GIE    ; Enable all interrupts
;
;  Ensure that the required sampling time for the selected input
;  channel has elapsed. Then the conversion may be started.
;
     BSF     ADCON0, GO     ; Start A/D Conversion
       :                    ; The ADIF bit will be set and the GO/DONE bit
       :                    ;    is cleared upon completion of the
                            ;    A/D Conversion.
```

图 22-4：A/D 转换过程中的 TAD 周期

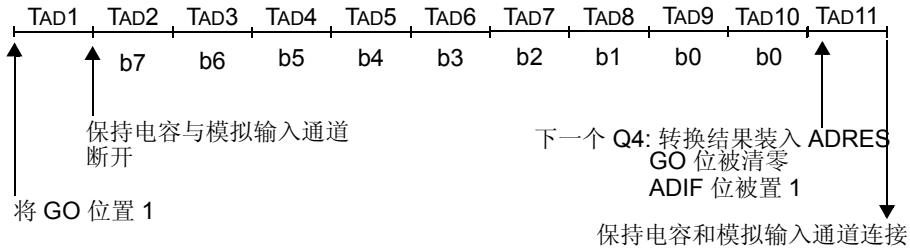
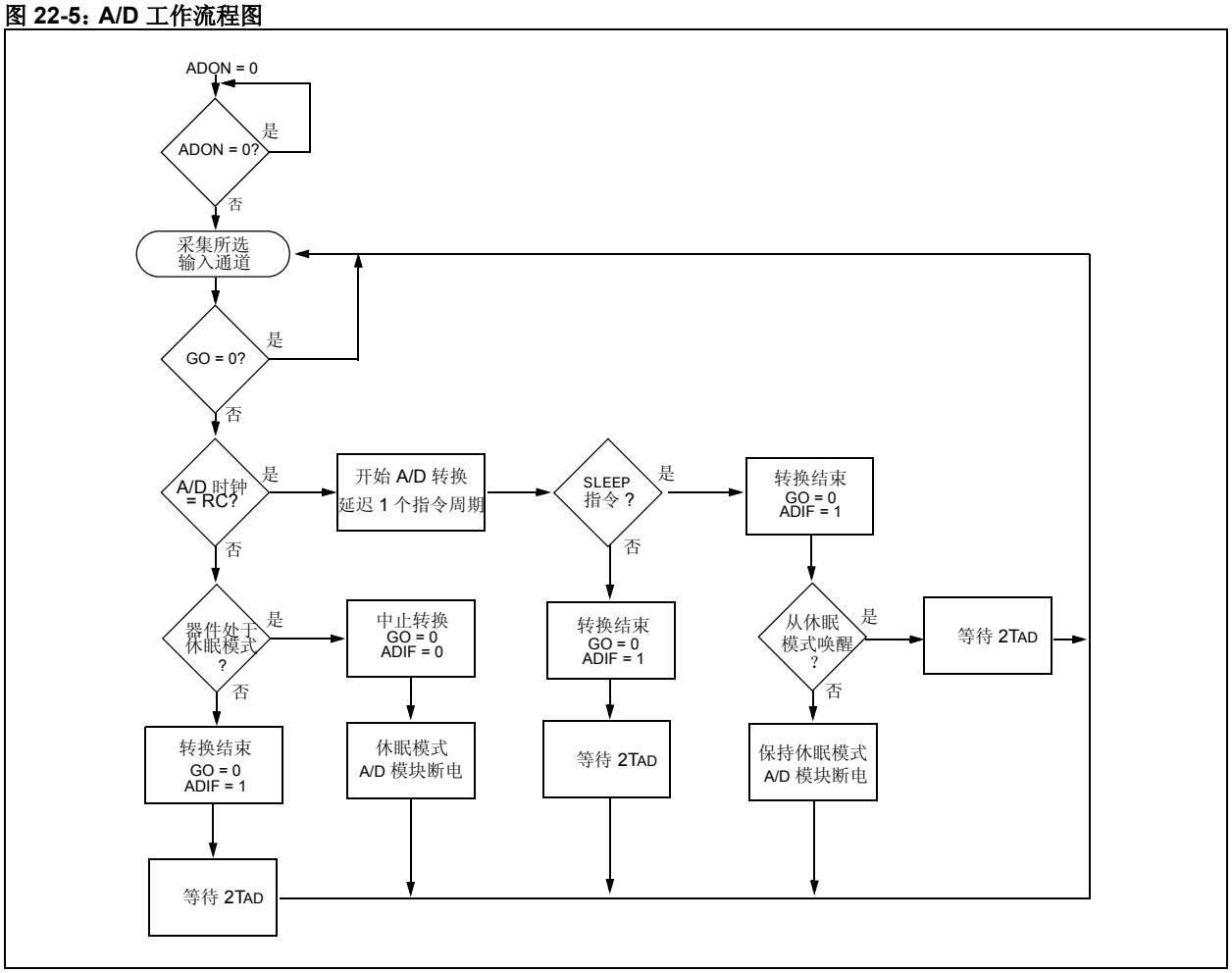


图 22-5：A/D 工作流程图



22.6.1 加快转换速度与降低转换精度的权衡

并非所有的应用都需要 8 位分辨率的转换结果，相反，它们可能需要更快的转换时间。A/D 模块允许用户降低转换分辨率以换取转换速度。无论所需的分辨率如何，采集时间都是相同的。为了加快转换速度，可切换 A/D 模块的时钟源，以使 TAD 违反规定的最小时间 (参见电气规范的有关说明)。一旦 TAD 违反了规定最小时间，所有接下来的 A/D 转换结果位将不再有效 (参见电气规范章节中 有关 A/D 转换时间的说明)。时钟源只能在三种振荡器间切换 (不能在振荡器模式和 RC 模式间相互切换)。用以下公式确定须经过多长时间才可切换振荡器：

转换时间 = TAD + N • TAD + (10 - N)(2TOSC)
其中 N = 所需分辨率的位数

由于 TAD 基于器件振荡器，用户必须使用一些方法 (如定时器，软件循环等) 以决定何时切换 A/D 振荡器。例 22-3 显示了 4 位分辨率与 8 位分辨率转换时间的对照。该例中器件的工作频率为 20 MHz 和 16 MHz (A/D 转换时钟设为 32TOSC)，并假定 A/D 时钟在紧随 5TAD 之后被设为 2TOSC。

由于后 4 位将无法正确转换，因此 2TOSC 违反了最小 TAD 时间规则。

例 22-3: 4 位和 8 位转换时间

	频率 (MHz)(1)	分辨率	
		4 位	8 位
TAD	20	1.6 µs	1.6 µs
	16	2.0 µs	2.0 µs
TOSC	20	50 ns	50 ns
	16	62.5 ns	62.5 ns
TAD + N • TAD + (10 - N)(2TOSC)	20	8.6 µs	17.6 µs
	16	10.75 µs	22 µs

注 1：PIC16C71 的最小 TAD 时间是 2.0 µs。
其它器件的最小 TAD 时间是 1.6 µs。
2：若需全部 8 位转换结果，则不能切换 A/D 时钟源。

22.7 休眠期间的 A/D 转换

A/D 模块可在休眠期间运行，这需要把 A/D 的时钟源设置成 RC 方式 (ADCS1:ADCS0 = 11)。选择了 RC 时钟源后，A/D 模块等待一个指令周期后才开始转换。这样使 SLEEP 指令得以执行，从而消除了转换时所有内部的数字开关噪声。A/D 转换完成后，GO/DONE 位清零，转换结果送入 ADRES 寄存器。如果 A/D 中断被使能，器件将从 SLEEP 唤醒。如果 A/D 中断被禁止，A/D 模块将被关闭，尽管此时 ADON 位保持置 1 状态。

如果 A/D 时钟源为另一时钟选项 (非 RC)，执行一条 SLEEP 指令将中止当前 A/D 转换并关闭 A/D 模块，尽管此时 ADON 位保持置 1 状态。

将 A/D 关闭将 A/D 模块置于电流消耗最小的状态。

注：	要使 A/D 模块在 SLEEP 模式下运行，A/D 时钟源必须被设置成 RC 模式 (ADCS1:ADCS0 = 11)。要在 SLEEP 下进行 A/D 转换，必须将 GO/DONE 位置 1，然后执行 SLEEP 指令。

22.8 A/D 转换精度 / 误差

在器件频率较低的系统中，最好使用 A/D 模块的 RC 时钟；在中高频时，TAD 应来源于器件的振荡器。

A/D 转换器的绝对精度参数包括量化误差、积分误差、微分误差、满量程误差、偏移误差和单一性等所有误差的总和。它定义为任意数码的实际转换值和理想转换值之间的最大偏差。 当 VDD = VREF 时，A/D 转换器（在器件的规定工作范围内）的绝对误差为 < ±1 LSb ； 然而，当 VDD 偏离 VREF 时， A/D 转换器的精度将下降。

在一个给定的模拟输入范围内， A/D 的输出数码是相同的，这是因为模拟输入被量化到数码了。典型量化误差为 ± 1/2 LSb，并存在于整个模拟数字转换过程中。减小量化误差的唯一方法是提高 A/D 转换器的分辨率。

偏移误差是首个实际数码转换电平与首个理想数码转换电平的差值。偏移误差使整个传递函数发生平移。通过模拟输入端的总漏电流和源阻抗的相互作用，偏移误差可在系统外校准或引入系统。

增益误差是指经过偏移误差调整后，末次实际转换电平与的末次理想转换电平之间的最大偏差。增益误差显示为传递函数的斜率变化。增益误差和满量程误差的区别在于满量程误差不考虑偏移误差。增益误差可通过软件校正以从系统中消除。

线性误差是指数码一致性的变化。线性误差不能从系统中校准。积分非线性误差是指经过增益误差调整后，各个输出数码的实际转换电平和理想转换电平之间的差值。

微分非线性误差是指最大实际数码宽度和理想数码宽度之间的差值，该误差无法校正。

引脚的最大漏电流在器件数据手册的电气规范参数 D060 中作了规定。

在器件频率较低的系统中，最好使用 A/D 模块的 RC 时钟。在中高频率时， TAD 应来源于器件的振荡器。 TAD 不得违反最小时间，且应最大限度地降低以减小由噪声和采样保持电容器放电造成的误差。

在 A/D 转换开始后器件就进入休眠模式的系统中，必须选择 RC 时钟源。在这种模式下，消除了来自休眠模块的数字噪声。这种方法能提供较好的转换精度。

22.9 复位对 A/D 转换的影响

器件复位强制所有寄存器为复位状态，同时强制 A/D 模块关闭并中止任何正在进行的转换。上电复位时， ADRES 寄存器中的值保持不变。上电复位后 ADRES 寄存器中的值不确定。

22.10 连接时的考虑事项

如果输入电压超出满幅电压值 (V_{SS} 或 V_{DD})0.3V，转换精度将超出规定范围。

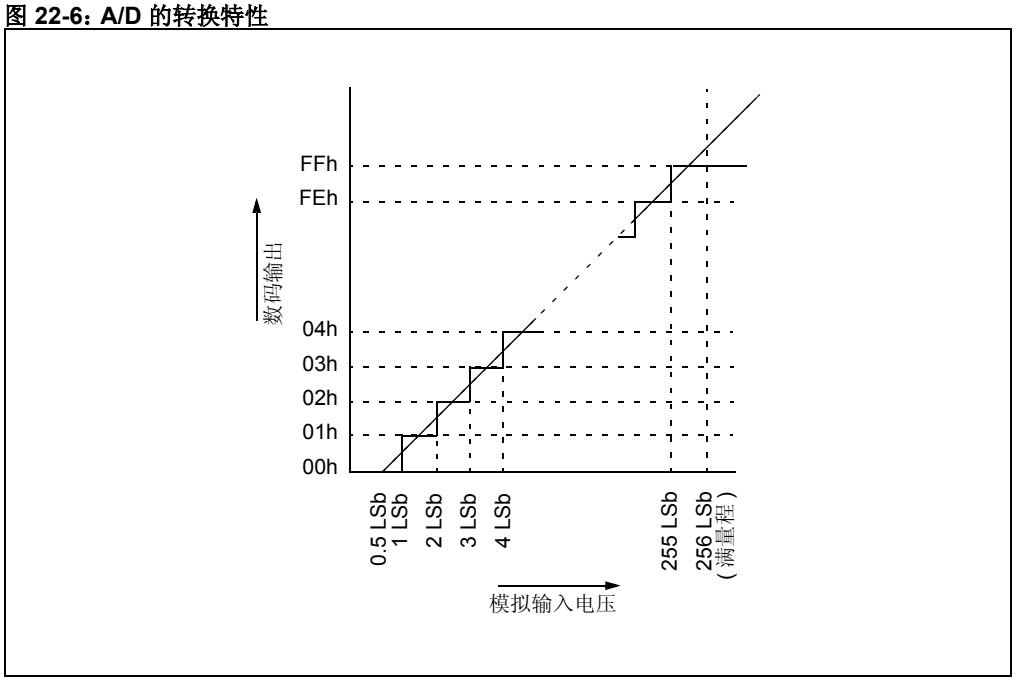
注： 由于 RA0 引脚比较靠近 OSC1 引脚，因此在进行 A/D 转换时必须谨慎使用该引脚。

有时可增加一个输入信号抗混叠外部 RC 滤波器。选择的 R 元件应确保总源阻抗小于建议值 10 kΩ。任何通过高阻抗连接到模拟输入引脚上的外部元件（如电容器、齐纳二极管等），应使其在引脚上的漏电流很小。

22.11 传递函数

以下是 A/D 转换器的理想传递函数：第一次转换发生在模拟输入电压 (V_{AIN}) 为 1 LSb(或 V_{REF} / 256) 时 (图 22-6)。

图 22-6：A/D 的转换特性



22.12 初始化

例 22-4 显示了 PIC16C711 的 A/D 模块的初始化。

例 22-4: A/D 的初始化 (PIC16C711)

```
     BSF     STATUS, RP0    ; Select Bank1
     CLRF    ADCON1         ; Configure A/D inputs
     BCF     STATUS, RP0    ; Select Bank0
     MOVLW   0xC1           ; RC Clock, A/D is on, Channel 0 selected
     MOVWF   ADCON0         ;
     BSF     INTCON, ADIE   ; Enable A/D Interrupt
     BSF     INTCON, GIE    ; Enable all interrupts
;
;  Ensure that the required sampling time for the selected input
;  channel has elapsed. Then the conversion may be started.
;
     BSF     ADCON0, GO     ; Start A/D Conversion
       :                    ; The ADIF bit will be set and the GO/DONE bit
       :                    ;    is cleared upon completion of the
                            ;    A/D Conversion.
```

22.13 设计技巧

问 1: *在使用 PIC16C7X 时，我发现模拟数字转换结果并不总是准确的。如何提高转换精度？*

答 1:

1. 请确保您满足了所有时序规范要求。如果你关闭 A/D 模块后再打开，应等待一个最小延迟时间后再开始采样；如果改变了输入通道，同样需要等待一个最小延迟时间后；最后是 TAD，它是所选择的每一位的转换时间。TAD 在 ADCON0 中选择，其值应该处于 2 到 6μs 之间。如果 TAD 太短，转换终止时尚未对数据进行完全转换，而如果 TAD 过长，采样电容上的电压会在转换结束前下降过大。数据手册提供了这些计时参数的表格或公式，您应根据具体器件与具体情况进行查找。
2. 模拟输入信号的源阻抗经常很高 (大于 1kΩ，因此为采样电容充电的信号源的输出电流会影响精度。如果输入信号并不快速变化，可以尝试在模拟输入端连接一个 0.1 μF 的电容。该电容可充电到所采样的模拟电压，并为内部 51.2 pf 的保持电容提供所需的瞬时充电电流。
3. 在 PIC16C71 上，有一个模拟输入引脚排列在振荡器引脚的旁边。当走线彼此相邻时，振荡器的数字噪声会耦合到模拟电路上。时钟源是外部振荡器时尤为如此，因为与晶体电路提供的上升较慢的正弦波不同，外部振荡器带有高频元件，其输出是边沿陡峭的方波信号。对这个模拟输入引脚进行去耦有助于改善此问题，或者，如果您不使用它，可将该引脚设置为输出并拉低电平。这将大大消除模拟输入的交叉耦合。
4. 最后，直接参见数据手册的内容：“在器件工作频率较低的系统中，最好使用来自器件振荡器的 A/D 时钟 ... 这将在很大程度上降低数字开关噪声的影响”，以及 “在 A/D 转换开始后器件就进入休眠模式的系统中，必须选择 RC 时钟源。这种方法可得到最高的转换精度。”

问 2: *在 A/D 转换开始后是否可以改变输出通道 (以便进行下一次转换)？*

答 2:

在保持电容从输入通道断开后， GO 位置 1 后经过一个 TAD 时间，就可以改变输入通道。

问 3: *请问有没有关于 A/D 的较好的参考书？*

答 3:

“Analog-Digital Conversion Handbook”是便于理解 A/D 转换的较好的参考书，该书由 Prentice Hall 出版 (ISBN 0-13-03-2848-0)。

22.14 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，经过一定的修改并受到一定的限制即可使用。目前与基本型 8 位 A/D 模块相关的应用笔记有：

标题	应用笔记 #
Using the Analog to Digital Converter	AN546
Four Channel Digital Voltmeter with Display and Keyboard	AN557

22.15 版本历史

版本 A

这是描述基本型 8 位 A/D 转换模块的初始发行版。

第 23 章 10 位 A/D 转换器

目录

本章包括以下一些主要内容：

23.1 简介 23-2
23.2 控制寄存器 23-3
23.3 操作 23-5
23.4 A/D 采集时间要求 23-6
23.5 A/D 转换时钟的选择 23-8
23.6 模拟输入引脚的设置 23-9
23.7 A/D 转换的编程举例 23-10
23.8 休眠期间的 A/D 转换 23-14
23.9 复位对 A/D 转换的影响 23-14
23.10 A/D 转换精度与误差 23-15
23.11 连接注意事项 23-16
23.12 传递函数 23-16
23.13 初始化 23-17
23.14 设计技巧 23-18
23.15 相关应用笔记 23-19
23.16 版本历史 23-20

注 1: 目前发布的中档系列单片机没有包含此模块。已计划推出包含此模块的单片机，但目前没有供货时间表。请登陆 Microchip 网站或 BBS 查阅已发布的包含器件详细特性的产品简介。

如果您现在的设计需要一个 10 位 A/D，请参阅 PIC17C756，它带有一个 12 通道的 10 位 A/D，该 A/D 的特性和本章描述的模块相同。

23.1 简介

该模拟数字转换器 (A/D) 模块有多达 8 个模拟输入通道。

模拟输入对一个采样保持电容器充电，采样保持电容的输出是 A/D 转换器的输入。A/D 转换器采用逐次逼近法将这一模拟电平产生数字转换结果，其转换结果为 10 位数字。

模拟参考电压（正电源电压和负电源电压）可通过软件选择为器件的电源电压 (AVDD、AVss) 或 AN3/VREF+ 和 AN2/VREF- 引脚上的电平。

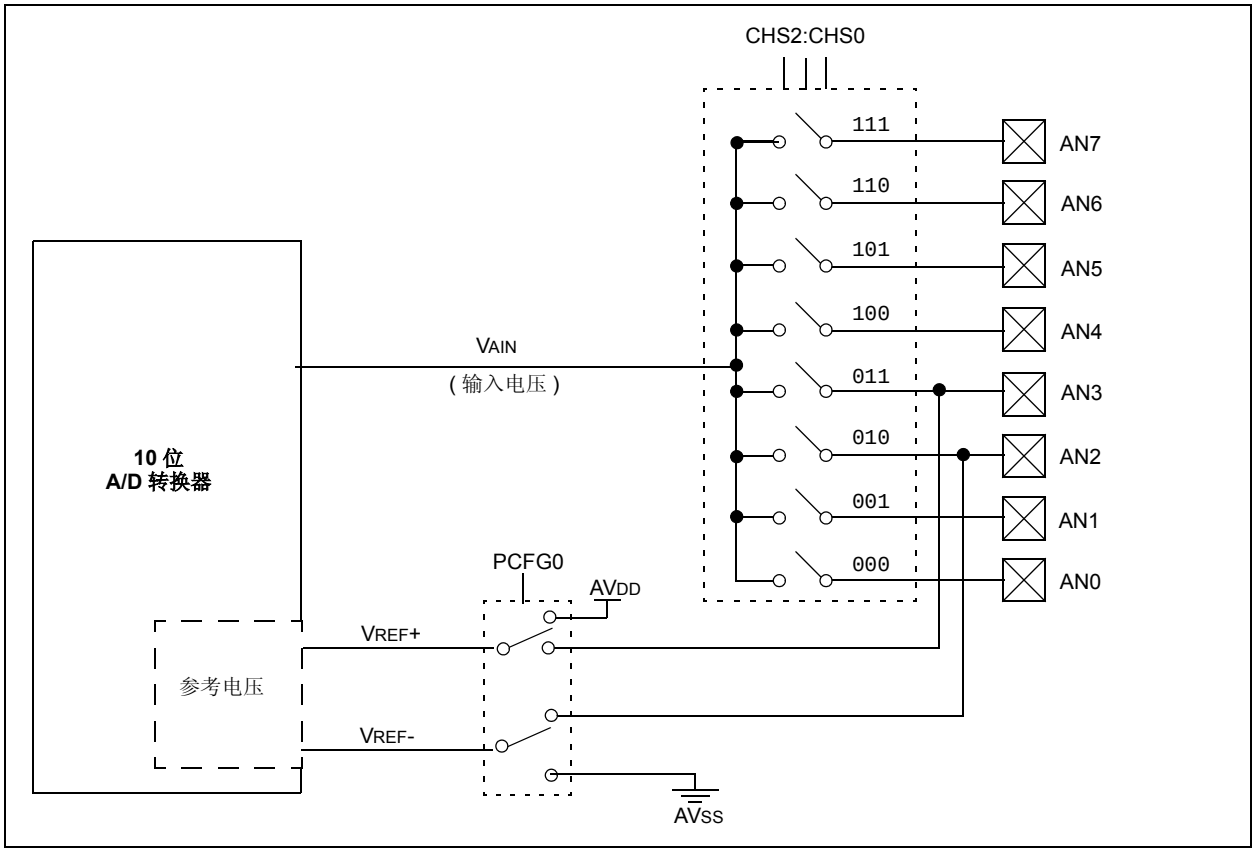
A/D 转换器具备可在休眠状态下工作的独特特性。

A/D 模块有四个寄存器，它们是：

- A/D 结果高位寄存器 (ADRESH)
- A/D 结果低位寄存器 (ADRESL)
- A/D 控制寄存器 0 (ADCON0)
- A/D 控制寄存器 1 (ADCON1)

ADCON0 寄存器，如图 23-1 所示，控制 A/D 模块的操作。ADCON1 寄存器，如图 23-2，可对端口的引脚功能进行配置。这些端口引脚可被设置成模拟输入 (其中 AN3 和 AN2 也可作为参考电压输入) 或数字 I/O 引脚。

图 23-1：10 位 A/D 转换器结构框图



23.2 控制寄存器

寄存器 23-1: ADCON0 寄存器

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7:6 **ADCS1:ADCS0**: A/D 转换时钟选择位

00 = FOSC/2
01 = FOSC/8
10 = FOSC/32
11 = FRC (A/D 模块内部专用的 RC 振荡器)

bit 5:3 **CHS2:CHS0**: 模拟通道选择位

000 = 通道 0 (AN0)
001 = 通道 1 (AN1)
010 = 通道 2 (AN2)
011 = 通道 3 (AN3)
100 = 通道 4 (AN4)
101 = 通道 5 (AN5)
110 = 通道 6 (AN6)
111 = 通道 7 (AN7)

注 : 对未用满 8 个 A/D 通道的器件，未使用的选项被保留。不要选择未使用的通道。

bit 2 **GO/$\overline{\text{DONE}}$:** A/D 转换状态位

当 ADON = 1 时

1 = A/D 转换正在进行 (该位置 1 启动 A/D 转换。 A/D 转换结束后该位由硬件自动清零)
0 = 未进行 A/D 转换

bit 1 **保留 :** 总是保持该位为 0。

bit 0 **ADON**: A/D 模块开启位

1 = A/D 转换器模块工作
0 = A/D 转换器关闭，不消耗工作电流

图注		
R = 可读位	W = 可写位	
U = 未用，读为 0		- n = 上电复位值

寄存器 23-2: ADCON1 寄存器

U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	ADFM	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7:6 **未用**：读为 0

bit 5 **ADFM**：A/D 结果格式选择位 (另见图 23-6)。

1 = 右对齐， ADRESH 寄存器的高 6 位读为 0
0 = 左对齐， ADRESL 寄存器的低 6 位读为 0

bit 4 **未用**：读为 0

bit 3:0 **PCFG3:PCFG0**：A/D 端口配置控制位

PCFG	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C / R
0000	A	A	A	A	A	A	A	A	AVDD	AVSS	8 / 0
0001	A	A	A	A	VREF+	A	A	A	AN3	AVSS	7 / 1
0010	D	D	D	A	A	A	A	A	AVDD	AVSS	5 / 0
0011	D	D	D	A	VREF+	A	A	A	AN3	AVSS	4 / 1
0100	D	D	D	D	A	D	A	A	AVDD	AVSS	3 / 0
0101	D	D	D	D	VREF+	D	A	A	AN3	AVSS	2 / 1
011x	D	D	D	D	D	D	D	D	—	—	0 / 0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6 / 2
1001	D	D	A	A	A	A	A	A	AVDD	AVSS	6 / 0
1010	D	D	A	A	VREF+	A	A	A	AN3	AVSS	5 / 1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4 / 2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3 / 2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2 / 2
1110	D	D	D	D	D	D	D	A	AVDD	AVSS	1 / 0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1 / 2

A = 模拟输入 D = 数字 I/O
C/R = 模拟输入通道数 / A/D 参考电压数

图注
R = 可读位 W = 可写位
U = 未用，读为 0 - n = 上电复位时的值

注 1: 在器件的复位时，复用为模拟功能 （ANx）的端口引脚均被强制置为模拟输入。

23.3 操作

ADRESH:ADRESL 寄存器中保存了 A/D 转换的 10 位结果。当 A/D 转换完成之后，转换结果被载入这一 A/D 结果寄存器对中， GO/DONE (ADCON0<2>) 位被清零，且 A/D 中断标志位 ADIF 置 1。 A/D 模块的结构框图见图 23-1。

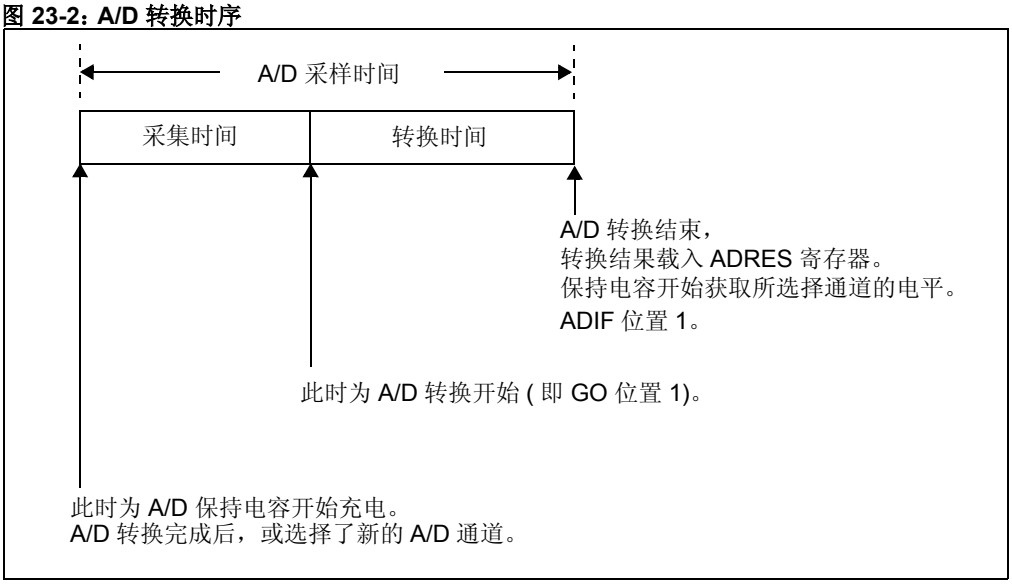
当配置好 A/D 模块后，在启动转换前必须先选择 A/D 转换的通道。模拟输入通道的相应 TRIS 位必须设置为输入。采集时间（acquisition time）的确定参见 **23.4 “A/D 采集时间要求”小节**。在这一采集时间过去之后， A/D 转换即可开始。按照以下步骤进行 A/D 转换：

1. 配置 A/D 模块
   - 对模拟引脚 / 参考电压 / 数字 I/O (ADCON1) 进行配置
   - 选择 A/D 输入通道 (ADCON0)
   - 选择 A/D 转换时钟 (ADCON0)
   - 打开 A/D 转换模块 (ADCON0)
2. 需要时，设置 A/D 中断
   - 将 ADIF 位清零
   - 将 ADIE 位置 1
   - 将 GIE 位置 1
3. 等待所需的采集时间
4. 启动 A/D 转换
   - 将 GO/DONE 置 1 (ADCON0)
5. 等待 A/D 转换完成，通过以下两种方法之一可判断转换是否完成：
   - 查询 GO/DONE 位是否被清零或 ADIF 位被置 1；

   或
   - 等待 A/D 转换的中断。
6. 读取 A/D 结果寄存器对 (ADRESH:ADRESL)，需要时将 ADIF 位清零。
7. 要再次进行 A/D 转换，根据要求转入步骤 1 或步骤 2。

图 23-2 为 A/D 转换顺序及所使用的术语。采集时间是 A/D 模块的保持电容连接到外部电平的时间。随后是 12 T_{AD} 的转换时间，开始于 GO 位被置 1。这两段时间的总和即采样时间（sampling time）。为确保保持电容充电至适当电平以使 A/D 转换达到所需精度，应保证一个最小采集时间。

图 23-2：A/D 转换时序



23.4 A/D 采集时间要求

为了使 A/D 转换达到规定精度，必须让充电保持电容 (CHOLD) 充满至输入通道的电平。图 23-3 显示了模拟输入模型。模拟信号的源阻抗 (RS) 和内部采样开关阻抗 (RSS) 直接影响电容器 CHOLD 所需的充电时间。采样开关 (RSS) 电阻随器件电压 (VDD) 变化，见图 23-3。**模拟信号源的最大建议阻抗为 10 kΩ**。采集时间随阻抗的降低而缩短。选择（改变）模拟输入通道后，在转换开始前必须先完成模拟信号的采集。

要计算最小采集时间，可使用公式 23-1。该公式假设误差为 1/2 LSb(即 A/D 的 1024 步)。 1/2 LSb 误差是 A/D 模块达到规定分辨率的最大允许误差。

公式 23-1： 采集时间

TACQ	=	放大器的建立时间 + 保持电容器充电时间 + 温度系数
	=	TAMP + TC + TCOFF

公式 23-2： A/D 最小充电时间

VHOLD	=	$(V_{REF} - (V_{REF}/2048)) \bullet (1 - e^{(-Tc/C_{HOLD}(R_{IC} + R_{SS} + R_S))})$
或		
Tc	=	-(120 pF)(1 kΩ + RSS + RS) ln(1/2047)

例 23-1 显示了所需最小采集时间 TACQ 的计算过程。该计算过程基于以下的假定：

CHOLD	=	120 pF
RS	=	10 kΩ
转换误差	≤	1/2 LSb
VDD	=	5V → Rss = 7 kΩ (参见图 23-3)
温度	=	50°C (系统最大值)
VHOLD	=	0V （时间 = 0 时）

例 23-1: 计算所需最小采集时间 (情况 1)

TACQ =	TAMP + TC + TCOFF
只有在温度 > 25°C 时才需要温度系数	
TACQ =	2 µs + Tc + [(Temp - 25°C)(0.05 µs/°C)]
TC =	-CHOLD (RIC + RSS + RS) ln(1/2047) -120 pF (1 kΩ + 7 kΩ + 10 kΩ) ln(0.0004885) -120 pF (18 kΩ) ln(0.0004885) -2.16 µs (-7.6241) 16.47 µs
TACQ =	2 µs + 16.47 µs + [(50°C - 25°C)(0.05 µs/°C)] 18.47 µs + 1.25 µs 19.72 µs

现在来了解一下当源阻抗为最小值时 (*RS* = 50 Ω) 时采集时间有什么变化。例 23-2 和例 23-1 的条件基本相同，唯一的不同在于源阻抗为最小值 (*RS* = 50 Ω)。

例 23-2: 计算所需的最小采集时间 (情况 2)

TACQ =	TAMP + TC + TCOFF
只有在温度 > 25°C 时，才需要温度系数。	
TACQ =	2 µs + Tc + [(Temp - 25°C)(0.05 µs/°C)]
TC =	-CHOLD (RIC + RSS + RS) ln(1/2047) -120 pF (1 kΩ + 7 kΩ + 50 Ω) ln(0.0004885) -120 pF (8050 Ω) ln(0.0004885) -0.966 µs (-7.6241) 7.36 µs
TACQ =	2 µs + 16.47 µs + [(50°C - 25°C)(0.05 µs/°C)] 9.36 µs + 1.25 µs 10.61 µs

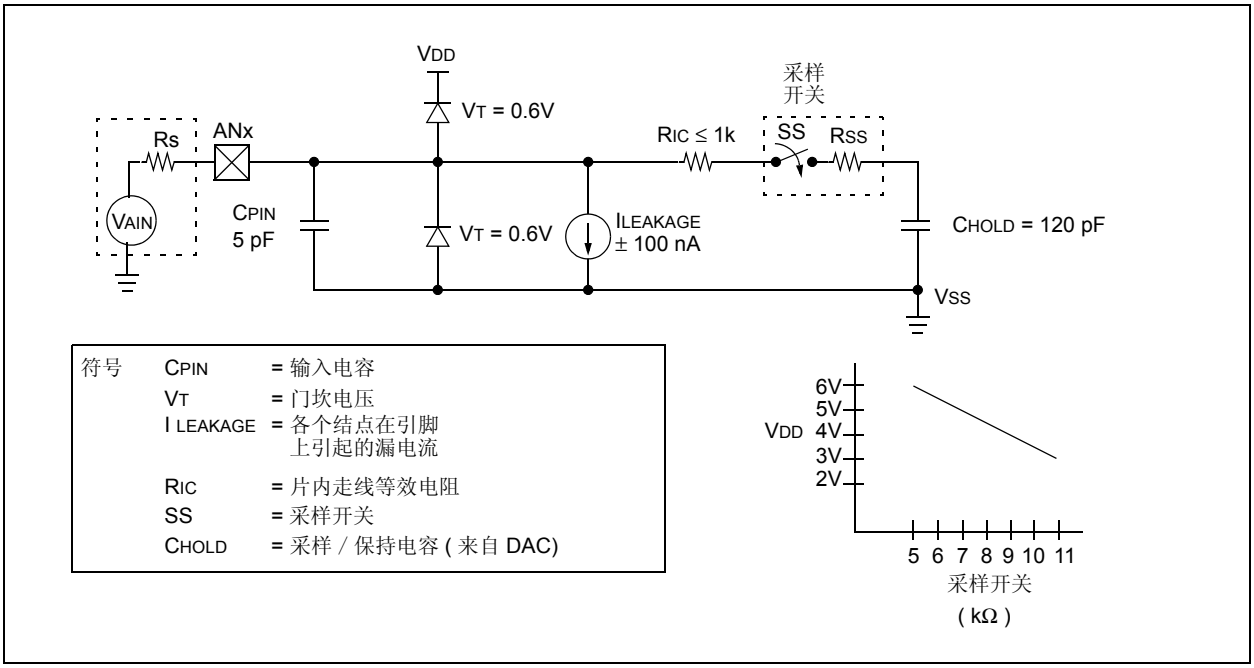
注 1: 参考电压 (VREF) 对该公式不产生影响，因为它在计算中已将自身消去。

注 2: 在每次转换后，充电保持电容 (CHOLD) 并不放电。

注 3: 模拟信号源的最大建议阻抗为 10 kΩ，这是为了满足引脚漏电流的要求。

注 4: 在转换完成之后，下一次采集重新开始前应等待 2.0 TAD。在此期间，保持电容并不与所选 A/D 输入通道相连接。

图 23-3：模拟输入模型



23.5 A/D 转换时钟的选择

每一位的 A/D 转换时间被定义为 T_{AD}。每完成一次 10 位 A/D 转换需要 11.5 个 T_{AD}。A/D 转换的时钟可用软件进行选择，对于 T_{AD} 可以有以下 4 种选择：

- 2Tosc
- 8Tosc
- 32Tosc
- A/D 模块内部 RC 振荡器

为了确保 A/D 转换正确，A/D 转换时钟 (TAD) 的选择必须满足最小 1.6 µs 的 TAD 时间，参见**“电气规范”一章**中的参数 130。

表 23-1 和表 23-2 显示了器件在不同工作频率下以及所选的不同 A/D 时钟源下得到的 TAD 结果。这些时间适用于标准电压范围的器件。

表 23-1: TAD 与器件工作频率关系表 (对于标准的 C 型器件)

AD 时钟源 (TAD)		器件工作频率			
状态	ADCS1:ADCS0	20 MHz	5 MHz	1.25 MHz	333.33 kHz
2TOSC	00	100 ns(2)	400 ns(2)	1.6 µs	6 µs
8TOSC	01	400 ns(2)	1.6 µs	6.4 µs	24 µs(3)
32TOSC	10	1.6 µs	6.4 µs	25.6 µs(3)	96 µs(3)
RC	11	2 - 6 µs(1,4)	2 - 6 µs(1,4)	2 - 6 µs(1,4)	2 - 6 µs(1)

图注： 阴影部分不在推荐工作范围内。
注 1： RC 时钟源的典型 TAD 为 4 µs。
2： 这些值违反了所需最小 TAD 时间规则。
3： 要加快转换时间，建议选择另一时钟源。
4： 器件工作频率高于 1 MHz 时，整个转换过程应在休眠模式下进行，否则 A/D 转换精度可能超出允许范围。

表 23-2: TAD 与器件工作频率关系表 (对于扩展的 LC 型器件)

AD 时钟源 (TAD)		器件频率			
工作状态	ADCS1:ADCS0	4 MHz	2 MHz	1.25 MHz	333.33 kHz
2TOSC	00	500 ns(2)	1.0 µs(2)	1.6 µs(2)	6 µs
8TOSC	01	2.0 µs(2)	4.0 µs	6.4 µs	24 µs(3)
32TOSC	10	8.0 µs	16.0 µs	25.6 µs(3)	96 µs(3)
RC	11	3 - 9 µs(1,4)	3 - 9 µs(1,4)	3 - 9 µs(1,4)	3 - 9 µs(1,4)

图注： 阴影部分不在推荐工作范围内。
注 1： RC 时钟源的典型 TAD= 6 µs。
2： 这些值违反了所需最小 TAD 时间。
3： 要加快转换时间，建议选择另一时钟源。
4： 器件工作频率高于 1 MHz 时，整个转换过程应在休眠模式下进行，否则 A/D 转换精度可能超出允许范围。

23.6 模拟输入引脚的设置

ADCON1 和 TRISA 寄存器用来控制 A/D 端口引脚的运行。若希望端口引脚为模拟输入，则必须将其相应的 TRIS 位置 1(输入)；如果 TRIS 位被清零（输出），则数字输出电平 （VOH 或 VOL) 将被转换。

A/D 转换与 CHS2:CHS0 位及 TRIS 位的状态无关。

注 1:	读取端口寄存器时，所有配置为模拟输入通道的引脚均读为 0(低电平)。配置为数字输入的引脚将转换模拟输入信号。配置为数字输入的引脚上的模拟电平将不影响转换精度。
注 2:	定义为数字输入的引脚上的模拟电平 (包括 AN7:AN0 引脚)，可能导致输入缓冲器消耗超出器件规范的电流。

23.7 A/D 转换的编程举例

例 23-3 显示了如何在 PIC17C756 上进行 A/D 转换。PORTF 端口和 PORTG 端口的低四位被配置成模拟输入。模拟参考电压 (V_{REF+} 和 V_{REF-}) 为器件的 AV_{DD} 和 AV_{SS}。使能 A/D 中断， A/D 转换时钟设为 F_{RC}。该转换在 AN0 引脚 （通道 0）上进行。

注： 由于所需采集时间的要求，不应在打开 A/D 模块的同一指令中将 GO/$\overline{\text{DONE}}$ 位置 1。

在转换期间将 GO/$\overline{\text{DONE}}$ 位清零将中止当前 A/D 转换。A/D 结果寄存器对中的内容不会被部分完成的 A/D 转换样本所更新，即， ADRESH:ADRESL 寄存器对仍然保持上一次转换完成后的结果 (或上一次写入 ADRESH:ADRESL 寄存器中的值)。 A/D 转换被中止后，在下一次采集开始前，需要等待 $2T_{AD}$ 的时间。等待 $2T_{AD}$ 之后，采集将在所选通道上自动开始。

例 23-3: A/D 转换

```
    BSF     STATUS, RP0       ; Select Bank1
    CLRF    ADCON1            ; Configure A/D inputs,
                              ;    result is left justified
    BSF     PIE1, ADIE        ; Enable A/D interrupts
    BCF     STATUS, RP0       ; Select Bank0
    MOVLW   0xC1              ; RC Clock, A/D is on, Channel 0 is selected
    MOVWF   ADCON0            ;
    BCF     PIR1, ADIF        ; Clear A/D interrupt flag bit
    BSF     INTCON, PEIE      ; Enable peripheral interrupts
    BSF     INTCON, GIE       ; Enable all interrupts
;
;  Ensure that the required sampling time for the selected input
;  channel has elapsed. Then the conversion may be started.
;
    BSF     ADCON0, GO        ; Start A/D Conversion
      :                       ; The ADIF bit will be set and the GO/DONE
      :                       ; bit is cleared upon completion of the
      :                       ;     A/D Conversion.
```

图 23-4：A/D 转换中的 T_{AD} 周期

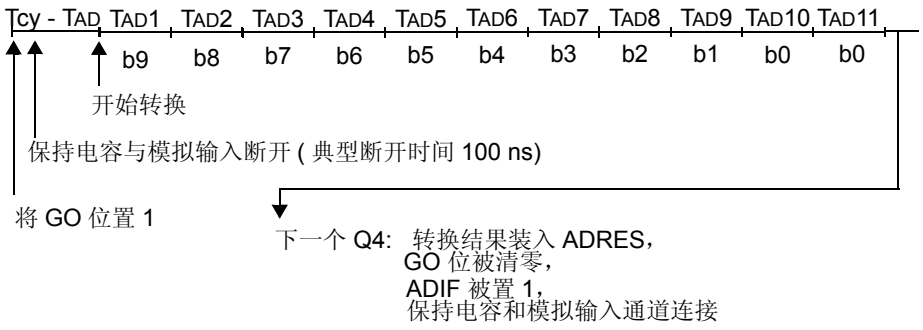
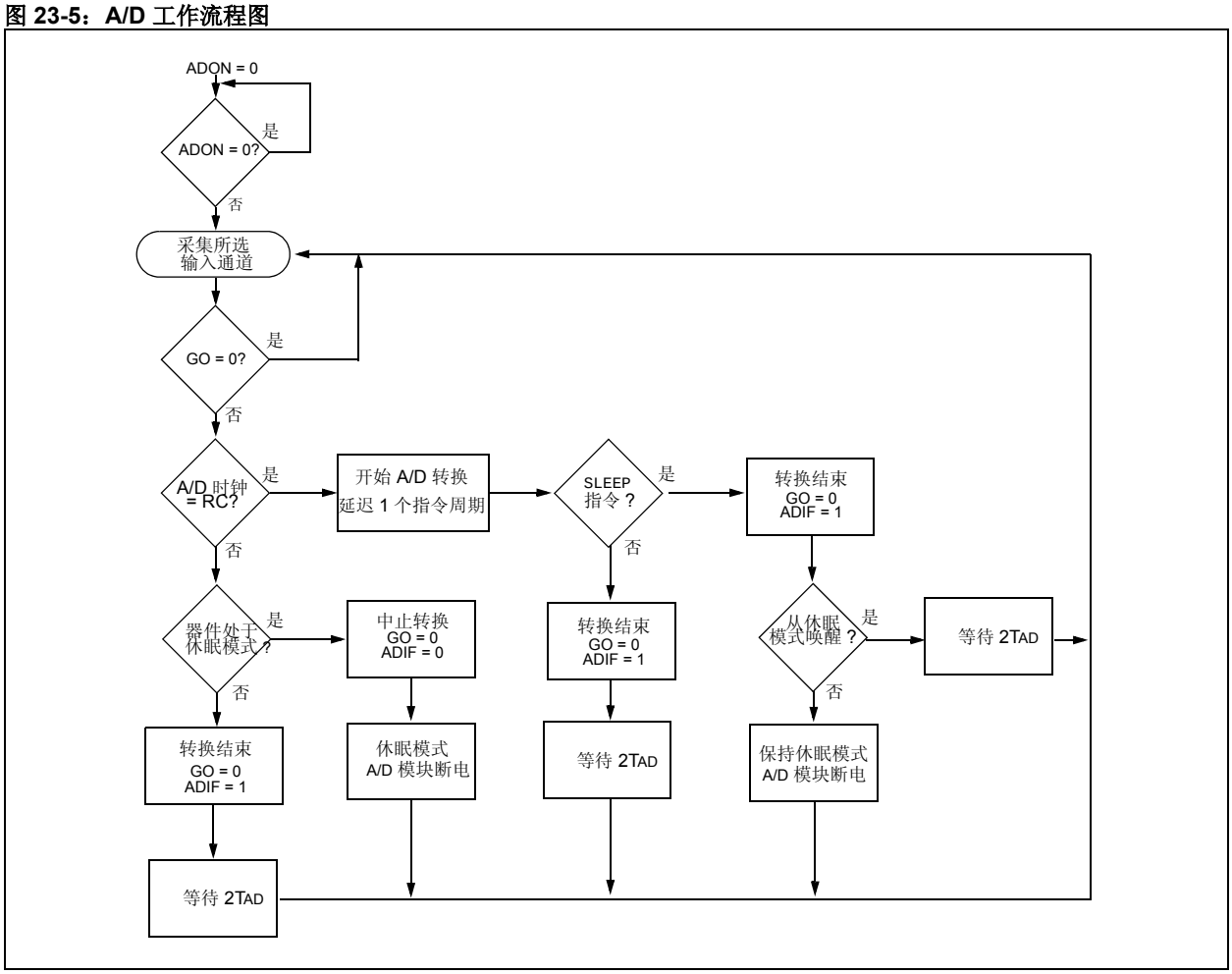


图 23-5：A/D 工作流程图



23.7.1 加快转换速度与降低转换精度的权衡

并非所有的应用都需要 10 位分辨率的转换结果，相反，它们可能需要更快的转换时间。 A/D 模块允许用户降低转换分辨率以换取转换速度。无论所需的分辨率如何，采集时间都是相同的。为了加快转换速度，可切换 A/D 模块的时钟源，以使 TAD 违反规定的最小时间 (参见电气规范的有关说明)。一旦 TAD 违反了规定最小时间，所有接下来的 A/D 转换结果位将不再有效 (参见电气规范章节中 有关 A/D 转换时间的说明)。时钟源只能在三种振荡器间切换 (不能在振荡器模式和 RC 模式间相互切换)。用以下公式确定须经过多长时间才可切换振荡器：

转换时间 = TAD + N • TAD + (11 - N)(2TOSC)
其中 N = 所需分辨率的位数

由于 TAD 基于器件振荡器，用户必须使用一些方法 (如定时器，软件循环等) 以决定何时切换 A/D 振荡器。例 23-4 显示了 4 位分辨率与 10 位分辨率转换时间的对照。该例中器件的工作频率为 20 MHz (A/D 转换时钟设为 32TOSC)，并假定 A/D 时钟在紧随 6TAD 之后被设为 2TOSC。

由于后 4 位将无法正确转换，因此 2TOSC 违反了最小 TAD 时间规则。

例 23-4: 4 位和 10 位转换时间

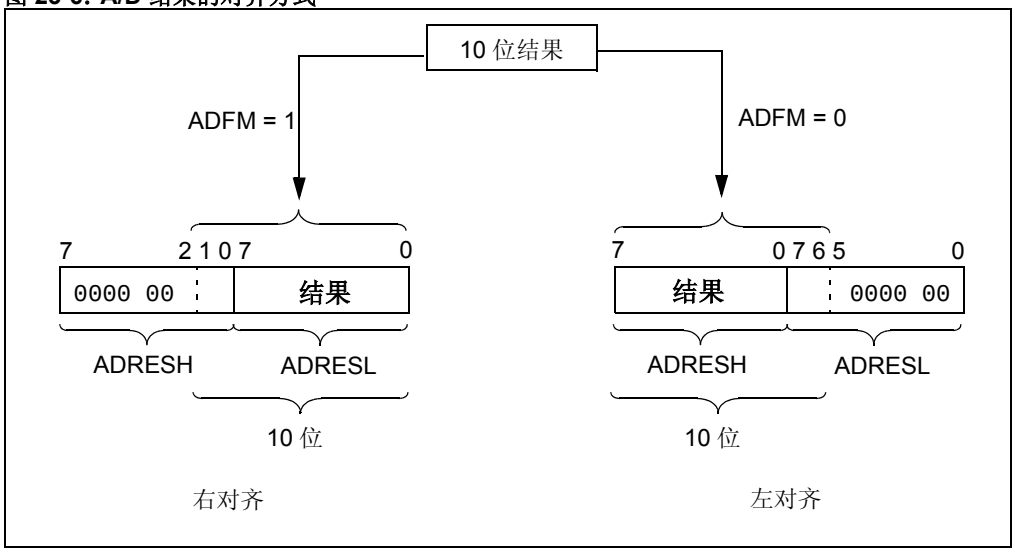
	频率 (MHz)(1)	分辨率	
		5 位	*10 位*
TAD	20	1.6 µs	1.6 µs
TOSC	20	50 ns	50 ns
2TAD + N • TAD + (11 - N)(2TOSC)	20	8.7 µs	17.6 µs

注 1： 要求最小 TAD 时间为 1.6 µs。
2： 若需全部 10 位转换结果，则不能切换 A/D 时钟源。

23.7.2 A/D 结果寄存器

在 A/D 转换结束后，其 10 位结果会存放于 ADRESH:ADRESL 寄存器对中。这对寄存器是 16 位宽。这一 A/D 模块提供了将 10 位结果以左对齐或右对齐的方式存放在 16 位结果寄存器中的灵活性。图 23-6 显示了 A/D 结果对齐的操作。多余位填入 0。当 A/D 结果不改写这些位置时 (A/D 被禁止)，这些寄存器可用作两个通用的 8 位寄存器。

图 23-6：A/D 结果的对齐方式



23.8 休眠期间的 A/D 转换

A/D 模块可在休眠期间运行，这需要把 A/D 的时钟源设置成 RC 方式 (ADCS1:ADCS0 = 11)。选择了 RC 时钟源后，A/D 模块等待一个指令周期后才开始转换。这样使 SLEEP 指令得以执行，从而消除了转换时所有内部的数字开关噪声。A/D 转换完成后， GO/DONE 位清零，转换结果送入 ADRES 寄存器。如果 A/D 中断被使能，器件将从 SLEEP 唤醒。如果 A/D 中断被禁止， A/D 模块将被关闭，尽管此时 ADON 位保持置 1 状态。

如果 A/D 时钟源为另一时钟选项 (非 RC)，执行一条 SLEEP 指令将中止当前 A/D 转换并关闭 A/D 模块 (以节省功耗)，尽管此时 ADON 位保持置 1 状态。

将 A/D 关闭将 A/D 模块置于电流消耗最小的状态。

注 :	要使 A/D 模块在 SLEEP 模式下运行， A/D 时钟源必须被设置成 RC 模式 (ADCS1:ADCS0 = 11)。要在 SLEEP 下进行 A/D 转换，必须将 GO/DONE 位置 1，然后执行 SLEEP 指令。

23.9 复位对 A/D 转换的影响

器件复位强制所有寄存器为复位状态，同时强制 A/D 模块关闭并中止任何正在进行的转换。

上电复位时， ADRESH:ADRESL 寄存器中的值保持不变。上电复位后 ADRESH:ADRESL 寄存器中的值不确定。

23.10 A/D 转换精度与误差

在器件频率较低的系统中，最好使用 A/D 模块的 RC 时钟；在中高频时， TAD 应来源于器件的振荡器。

A/D 转换器的绝对精度参数包括量化误差、积分误差，微分误差、满量程误差、偏移误差和单一性等所有误差的总和。它被定义为任意数码的实际转换值和理想转换值之间的最大偏差。 当 VDD = VREF 时，A/D 转换器（在器件的规定工作范围内）的绝对误差为 < ±1 LSb ；然而，当 VDD 偏离 VREF 时， A/D 转换器的精度将下降。

在一个给定的模拟输入范围内， A/D 的输出数码是相同的，这是因为模拟输入被量化到数码了。典型量化误差为 ± 1/2 LSb，并存在于整个模拟数字转换过程中。减小量化误差的唯一方法是提高 A/D 转换器的分辨率。

偏移误差是首个实际数码转换电平与首个理想数码转换电平的差值。偏移误差使整个传递函数发生平移。通过模拟输入端的总漏电流和源阻抗的相互作用，偏移误差可在系统外校准或引入系统。

增益误差是指经过偏移误差调整后，末次实际转换电平与的末次理想转换电平之间的最大偏差。增益误差显示为传递函数的斜率变化。增益误差和满量程误差的区别在于满量程误差不考虑偏移误差。增益误差可通过软件校正以从系统中消除。

线性误差是指数码一致性的变化。线性误差不能从系统中校准。积分非线性误差是指经过增益误差调整后，各个输出数码的实际转换电平和理想转换电平之间的差值。

微分非线性误差是指最大实际数码宽度和理想数码宽度之间的差值，该误差无法校正。

引脚的最大漏电流在器件数据手册的电气规范参数 D060 中作了规定。

在器件频率较低的系统中，最好使用 A/D 模块的 RC 时钟。在中高频率时， TAD 应来源于器件的振荡器。 TAD 不得违反最小时间，且应最大限度地降低以减小由噪声和采样保持电容器放电造成的误差。

在 A/D 转换开始后器件就进入休眠模式的系统中，必须选择 RC 时钟源。在这种模式下，消除了模块的数字噪声。这种方法能提供较好的转换精度。

23.11 连接注意事项

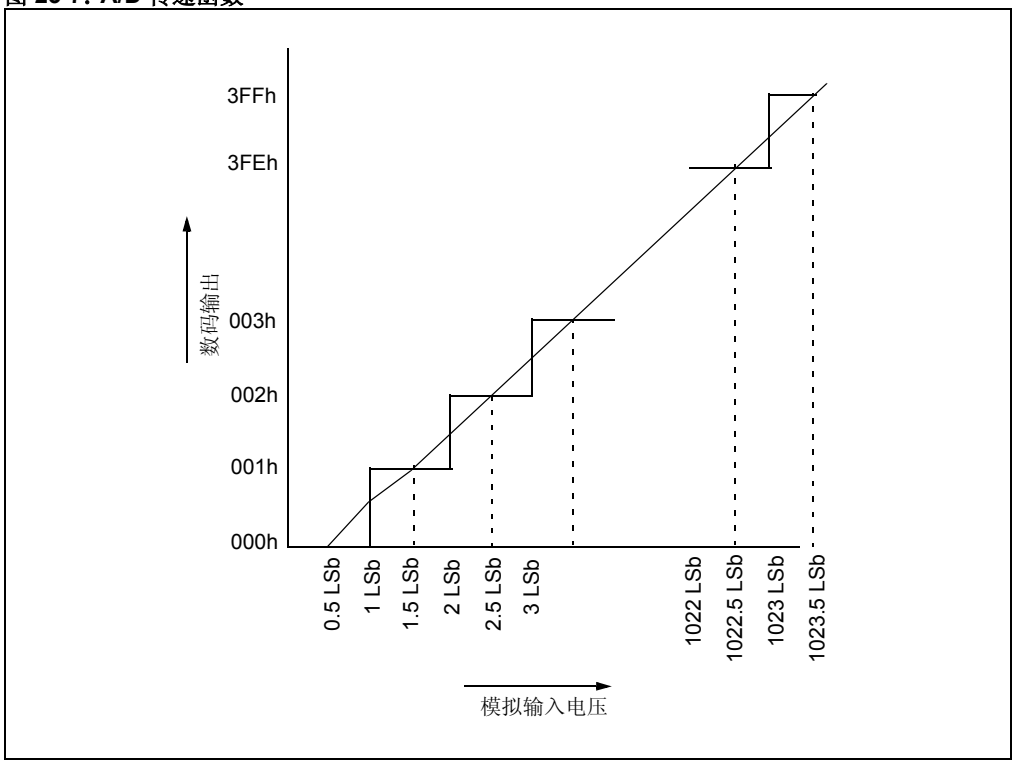
如果输入电压超出满幅电压值 (VSS 或 VDD)0.3V，转换精度将超出规定范围。

有时可增加一个输入信号抗混叠外部 RC 滤波器。选择的 R 元件应确保总源阻抗小于建议值 10 kΩ。任何通过高阻抗连接到模拟输入引脚上的外部元件（如电容器、齐纳二极管等)，应使其在引脚上的漏电流很小。

23.12 传递函数

以下是 A/D 转换器的理想传递函数：第一次转换发生在模拟输入电压 (VAIN) 为 1 LSb(或模拟 VREF / 1024) 时 （图 23-7）。

图 23-7：A/D 传递函数



23.13 初始化

例 23-5 给出了 A/D 模块的初始化。

例 23-5: A/D 初始化

```
    BSF     STATUS, RP0      ; Select Bank1
    CLRF    ADCON1           ; Configure A/D inputs
    BSF     PIE1, ADIE       ; Enable A/D interrupts
    BCF     STATUS, RP0      ; Select Bank0
    MOVLW   0xC1             ; RC Clock, A/D is on, Channel 0 is selected
    MOVWF   ADCON0           ;
    BCF     PIR1, ADIF       ; Clear A/D interrupt flag bit
    BSF     INTCON, PEIE     ; Enable peripheral interrupts
    BSF     INTCON, GIE      ; Enable all interrupts
;
;  Ensure that the required sampling time for the selected input
;  channel has elapsed. Then the conversion may be started.
;
    BSF     ADCON0, GO       ; Start A/D Conversion
      :                      ; The ADIF bit will be set and the GO/DONE
      :                      ; bit is cleared upon completion of the
      :                      ;     A/D Conversion.
```

23.14 设计技巧

问 1: *我发现模拟数字转换结果并不总是准确的。如何提高转换精度？*

答 1:

1. 请确保您满足了所有时序规范要求。如果你关闭 A/D 模块后再打开，应等待一个最小延迟时间后再开始采样；如果改变了输入通道，同样需要等待一个最小延迟时间后；最后是 TAD，它是所选择的每一位的转换时间。TAD 在 ADCON0 中选择，其值应该处于 1.6 到 6μs 之间。如果 TAD 太短，转换终止时尚未对数据进行完全转换，而如果 TAD 过长，采样电容上的电压会在转换结束前下降过大。**“电气规范”一章**中提供了这些计时参数。器件的具体信息请参见器件数据手册。
2. 模拟输入信号的源阻抗经常很高 (大于 1kΩ)，因此为采样电容充电的信号源的输出电流会影响精度。如果输入信号并不快速变化，可以尝试在模拟输入端连接一个 0.1 μF 的电容。该电容可充电到所采样的模拟电压，并为内部 120 pf 的保持电容提供所需的瞬时充电电流。
3. 最后，直接参见数据手册的内容：“在器件工作频率较低的系统中，最好使用来自器件振荡器的 A/D 时钟 ... 这将在很大程度上降低数字开关噪声的影响”，以及 “在 A/D 转换开始后器件就进入休眠模式的系统中，必须选择 RC 时钟源。这种方法可得到最高的转换精度。”

问 2: *在 A/D 转换开始后是否可以改变输出通道 (以便进行下一次转换)？*

答 2:

在保持电容从输入通道断开后， GO 位置 1 后 100 ns （典型值），就可以改变输入通道。

问 3: *请问有没有关于 A/D 的较好的参考书？*

答 3:

“Analog-Digital Conversion Handbook”是便于理解 A/D 转换的较好的参考书，该书由 Prentice Hall 出版 (ISBN 0-13-03-2848-0)。

23.15 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与 10 位 A/D 模块相关的应用笔记有：

标题	应用笔记 #
Using the Analog to Digital Converter	AN546
Four Channel Digital Voltmeter with Display and Keyboard	AN557

23.16 版本历史

版本 A

这是描述 10 位 A/D 模块的初始发行版。

第 24 章 积分型 A/D 转换器

目录

本章包括以下一些主要内容：

24.1 简介 24-2
24.2 控制寄存器 24-3
24.3 转换过程 24-6
24.4 其它模拟模块 24-12
24.5 校准参数 24-13
24.6 设计技巧 24-14
24.7 相关应用笔记 24-15
24.8 版本历史 24-16

24.1 简介

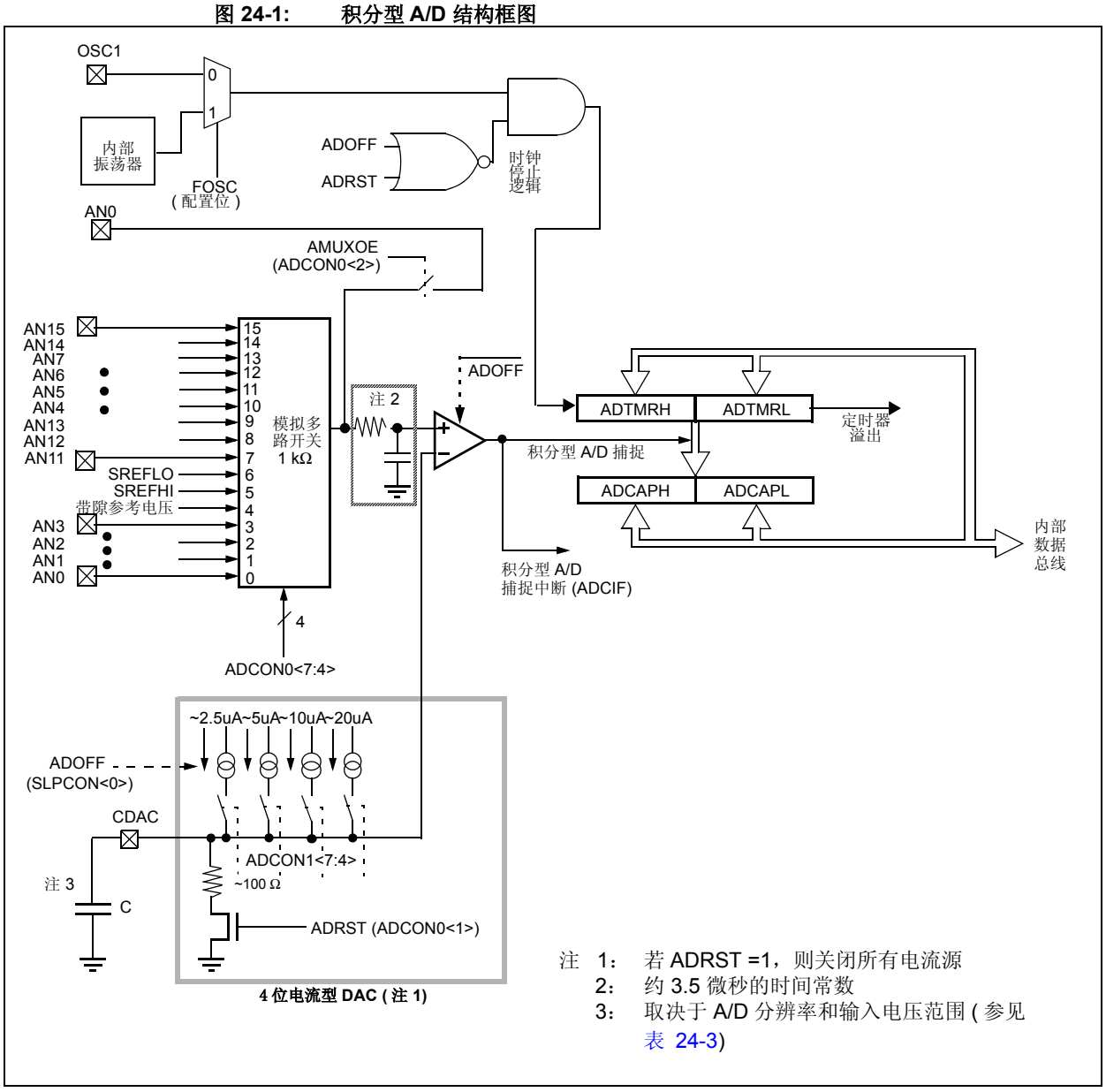
设计一个积分型 A/D 转换器需要以下元件：

- 高精度比较器
- 4 位可编程电流源
- 16 通道模拟多路开关 (MUX)
- 带捕捉寄存器的 16 位定时器

本章将讨论如何使用它们构成积分型 A/D。

每个模拟输入通道均通过一个多路器连接到一个模拟输入源，并通过斜率转换法（用一个高精度比较器）转换成数字信号。可编程电流源流入一个外部电容以产生用于转换的斜坡电压。

图 24-1： 积分型 A/D 结构框图



注 1： 若 ADRST =1，则关闭所有电流源
2： 约 3.5 微秒的时间常数
3： 取决于 A/D 分辨率和输入电压范围 (参见表 24-3)

24.2 控制寄存器

利用两个 A/D 控制寄存器用来控制转换过程，它们是 ADCON0 和 ADCON1。这两个寄存器都是可读写的。

寄存器 24-1: ADCON0 寄存器

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-1	R/W-0
ADCS3	ADCS2	ADCS1	ADCS0	—	AMUXOE	ADRST	ADZERO
bit 7							bit 0

bit 7-4: **ADCS3:ADCS0**：模拟通道选择位

0000 = AN0 为输入
0001 = AN1 为输入
0010 = AN2 为输入
0011 = AN3 为输入
0100 = 带隙参考电压输入
0101 = 斜率参考 SREFHI 输入
0110 = 斜率参考 SREFLO 输入
0111 = AN11 为输入
1000 = AN12 为输入
1001 = AN13 为输入
1010 = AN4 为输入
1011 = AN5 为输入
1100 = AN6 为输入
1101 = AN7 为输入
1110 = AN14 为输入
1111 = AN15 为输入

注： 对于未使用全部 16 个 A/D 输入通道的器件，未使用选择位被保留。请不要选择任何未用通道。

bit 3: **未用：**读作 0

bit 2: **AMUXOE:** 模拟 MUX 输出使能位

1 = AMUX 输出连接到 AN0 引脚 (改写 TRIS 设置)
0 = 正常 AN0 引脚

bit 1: **ADRST:** A/D 复位控制位

1 = 停止 A/D 定时器， CDAC 电容放电
0 = 正常运行 (A/D 转换)

bit 0: **ADZERO**：A/D 零点选择控制

1 = 使能 AN1 和 AN5 上的调零操作
0 = 正常运行，在 AN1 和 AN5 引脚上采样

图注		
R = 可读位	W = 可写位	
U = 未用位，读为 0		- n = 上电复位 (POR) 时的值

寄存器 24-2: ADCON1 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
ADDAC3	ADDAC2	ADDAC1	ADDAC0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7-4: **ADDAC3:ADDAC0**：可编程电流源选择位：

0000： OFF - 关闭所有电流源
0001： 2.25 µA
0010： 4.5 µA
0011： 6.75 µA
0100： 9 µA
0101： 11.25 µA
0110： 13.5 µA
0111： 15.75 µA
1000： 18 µA
1001： 20.25 µA
1010： 22.5 µA
1011： 24.75 µA
1100： 27 µA
1101： 29.25 µA
1110： 31.5 µA
1111： 33.75 µA

bit 3-0: **PCFG3:PCFG0**：端口配置选择位

PCFG3:PCFG2	AN4	AN5	AN6	AN7
00	A	A	A	A
01	A	A	A	D
10	A	A	D	D
11	D	D	D	D

PCFG1:PCFG0	AN0	AN1	AN2	AN3
00	A	A	A	A
01	A	A	A	D
10	A	A	D	D
11	D	D	D	D

A = 模拟输入 D = 数字 I/O

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

注： 当器件复位时，所有与模拟功能复用的引脚（ANx 引脚）都被强制为模拟输入。

寄存器 24-3: SLPCON 寄存器

R/W-0	U-0	R/W-1	R/W-1	U-1	R/W-1	R/W-1	R/W-1
Resv	—	REFOFF	Resv	OSCOFF	Resv	Resv	ADOFF
bit 7							bit 0

bit 7: **保留位 :** 始终保持为 0

bit 6: **未用 :** 读为 0

bit 5: **REFOFF:** 积分型 A/D 的参考电压源通电控制位

1 = 参考电压源关闭 (不消耗电流)
0 = 参考电压源通电 (消耗电流)

bit 4: **保留位 :** 始终保持为 0

bit 3: **OSCOFF:** 积分型 A/D 的振荡器休眠控制位

1 = 休眠状态时，关闭积分型 A/D 振荡器 (不消耗电流)
0 = 休眠状态时，使能积分型 A/D 振荡器 (消耗电流)

bit 2: **保留位 :** 始终保持为 0

bit 1: **保留位 :** 始终保持为 0

bit 0: **ADOFF:** 积分型 A/D 通电控制位

1 = 积分型 A/D 被关闭 (不消耗电流)
0 = 积分型 A/D 通电 (消耗电流)

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

24.3 转换过程

进行 A/D 转换有两种方法。为了确定转换是否结束，第一种方法使用 ADTMR 溢出中断（OVFIF 位）。第二种方法使用 A/D 捕捉中断（ADCIF 位）。转换结束时，用这两位来判断是否出现了溢出条件。

方法 1 使用固定的转换时间，即电容电压总是上升到满幅电压。建议在 ADTMR 溢出后，立即将 ADRST 位置 1 来对外部电容放电。这将确保外部电容上的残留电压被电介质吸收，而不影响输入电压或先前的转换。

方法 2 使用可变的转换时间，这在输入电压较低时可转换得更快。

方法 1 的步骤 （“固定转换时间”）：

1. 初始化积分型 A/D 模块
   a) REFOFF 位清零 (SLPCON<5>)
   b) ADOFF 位清零 (SLPCON<0>)
   c) 初始化 ADCON1<7:4> 以选择可编程电流源。
2. ADRST 位置 1 (ADCON0<1>)，直到斜坡电容达到地电平。这由电容大小决定，建议最小值为 200 µs。
3. 选择输入通道。
4. OVFIF 和 ADCIF 位清零。
5. 初始化积分型A/D定时器(ADTMR)。ADTMR 的值取决于所要求的分辨率（参见表 24-2）。
6. 将 ADRST 位清零以启动转换，这将使斜坡电容开始充电且 ADTMR 开始递增计数。
7. 当积分型 A/D 定时器 (ADTMR) 从 FFFFh 到 0000h 发生溢出时，转换结束。这将使 OVFIF 位置 1。
8. 检测 ADCIF 位是否被置 1。如果该位被置 1，则捕捉寄存器 ADCAP 上的值有效。这种方法根据 ADTMR 溢出后的最小反应时间来检验捕捉中断标志位。如果 ADCIF 位被清零，则输入电压超出了 A/D 输入范围。
9. 将 ADRST 位置 1(ADCON0<1>)，停止 ADTMR，并对外部电容放电。
10. 做转换计算。
11. 转到步骤 2。

方法 2 的步骤 （“可变转换时间”）：

1. 初始化积分型 A/D 模块：
   a) REFOFF(SLPCON<5>) 位清零
   b) ADOFF (SLPCON<0>) 位清零
   c) 初始化 ADCON1<7:4> 位选择可编程电流源
2. ADRST 位置 1 (ADCON0<1>)，直到斜坡电容达到地电平。这由电容大小决定，建议最小值为 200 µs。
3. 选择输入通道。
4. 清除 OVFIF 和 ADCIF 位。
5. 初始化积分型A/D 定时器 (ADTMR)。ADTMR的值取决于所要求的分辨率 (参见表 24-2)。
6. 将 ADRST 位清零以启动转换，这将使斜坡电容开始充电，且 ADTMR 开始递增计数。
7. 当斜坡电压超出模拟输入电压时，转换结束，比较器输出从高电平变为低电平。这将使 ADCIF 位被置 1。
8. 检测 ADTMR 是否在达到最大分辨率所允许的值后继续递增计数。如果继续计数，则输入电压超出了 A/D 输入范围。
9. 将 ADRST 位置 1，停止 ADTMR，并对外部电容放电。
10. 做转换计算。
11. 转到步骤 2。

注： 在发生捕捉事件后，积分型 A/D 定时器将继续运行。

积分型 A/D 定时器的最大计数值为 65,536。它可由片上或外部振荡器提供时钟。在 4MHz 的振荡频率下，满计数的最大转换时间是 16.38 ms。一次典型的转换应在满计数前完成。一旦定时器计满回零 （从 FFFFh 到 0000h），定时器的溢出标志位就被置 1，并当中断使能时产生中断。

使用片上 EPROM 可简化或取消最终用户的校准。内部元件值在出厂前最终测试时被测定并储存在存储器中，以备应用程序固件使用。

应定期进行带隙和斜坡参考电压 (参见 **24.4 “其它模拟模块” 小节**) 的转换，以便对积分型 A/D 的元件参数漂移进行补偿。在校准期间，参考电压的值被赋予存储在 EPROM 中的电压值。由于所有测量都与该参考值相关，比较器的固有偏置电压被最大限度地降低。积分型 A/D 的时钟源并不需要有精确的频率，而仅需一个稳定的频率即可。

欲进一步了解积分型 A/D 运行的详细内容，请参见 AN624，“PIC14000 积分型 A/D 理论及其实现方法”。

24.3.1 积分型 A/D 定时器 (ADTMR)

积分型 A/D 定时器 (ADTMR) 是由一个 16 位定时器 (ADTMRH:ADTMRL) 构成的，在每个振荡周期进行递增计数。上电复位时， ADTMR 寄存器被清零，否则，必须在每次转换完成后用软件对它进行初始化。当一个积分型 A/D 捕捉事件发生时，使用一个 16 位捕捉寄存器 (ADCAPH:ADCAPL) 捕捉 ADTMR 的计数值 (见以下内容)。积分型 A/D 定时器和捕捉寄存器都是可读写的。 16 位定时器是一个可读写寄存器，并在器件复位时清零。

注 1:	在积分型 A/D 转换进行期间，对 ADTMR 寄存器的读写会产生无法预测的结果，建议不要这么做。
注 2:	写 ADTMR 寄存器时，正确的顺序是先高字节后低字节。颠倒这一顺序将阻止积分型 A/D 定时器运行。

在转换期间，可能会分别或同时发生下列事件：

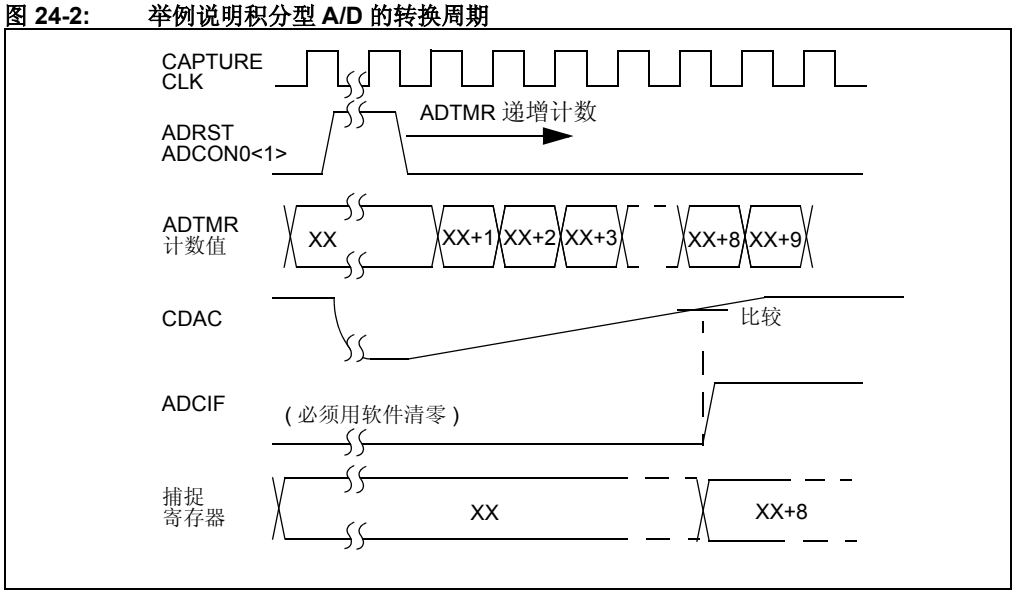
- 捕捉事件
- 定时器溢出

在一次捕捉事件中，当 CDAC 的输出超出所选择的积分型 A/D 通道的输入电压时，比较器的输出将由高电平跳变为低电平。这将使当前计数器的值被转移到捕捉寄存器，并将 ADCIF 标志位置 1。

如果 ADCIE 位被置 1 （中断使能）将产生中断。此外必须将 GIE 和 PIE 位置 1。在下次转换开始前，必须用软件将 ADCIF 标志位清零。每个转换周期内，只会发生一次该中断。

在溢出条件下，定时器从 FFFFh 计到 0000h，捕捉溢出标志 (OVFIE) 位被置 1。在溢出发生后，定时器继续递增计数。 OVFIE 位被置 1 时可产生一个中断 （中断使能时）。此外必须将 GIE 和 PIE 位也置 1。在下次转换开始前，必须用软件将 OVFIF 标志位清零。

图 24-2: 举例说明积分型 A/D 的转换周期



24.3.2 休眠状态下的操作

当器件处于休眠状态时，积分型 A/D 可以继续工作。要在休眠状态时进行转换，积分型 A/D 模块必须有一个时钟。要使时钟工作， OSCOFF 位必须在器件进入休眠状态前被清零。REFOFF 和 ADOFF 位也必须被清零，以确保转换结果正确反映输入通道的电压。在休眠模式下进行 A/D 转换，转换结果会因为系统噪声的减少而更精确。

当器件时钟被关闭时，积分型 A/D 定时器 (ADTMRH:ADTMRL) 停止递增计数。即使积分型 A/D 模块本身未被关闭，积分型 A/D 仍不能唤醒器件。这是因为用于把器件从休眠状态唤醒的控制位之一 ADCIF 位，不可能被置 1。当器件唤醒时，如果比较器的值已经跳变，将产生捕捉和中断。ADCAP 寄存器中的值是无意义的。

为了最大限度地节省能量，当没有进行转化时，应关闭所有积分型 A/D 模块的模拟部件。

24.3.3 复位的影响

在任何一次复位后，积分型 A/D 模块将被关闭 (最小电流状态)，且器件的 I/O 引脚均被配置为模拟通道。

24.3.4 积分型 A/D 的比较器

该模块包括一个用于积分型 A/D 转换的高增益比较器。比较器的同相输入端通过一个 RC 低通滤波器连接到模拟多路开关（MUX）的输出端。其反相输入端与外部斜坡电容相连接。

比较器的输出用于产生捕捉事件，这会使 ADTMR 寄存器中的值被载入 ADCAP 寄存器。该输出也会使 ADCIF 位置 1。

24.3.5 模拟多路开关 (MUX)

共有 16 个通道通过多路开关内部连接到单片积分型 A/D 比较器的正输入端。4 个配置位 (ADCON0<7:4>) 用于选择要进行转换的通道。

24.3.6 可编程电流源

4 个配置位 (ADCON1<7:4>) 用来控制可编程电流源，以产生一个输入到积分型 A/D 比较器的斜坡电压。选择不同的电流源可对满幅输入电压、时钟频率和外部电容的容差进行补偿。

将 ADRST 置 1 将断开 CDAC 引脚的电流源。当 ADRST 位被清零时，将接通电流源。

可编程电流源的输出被连接到 CDAC 引脚。该电流源用于对外部电容充电，该电容用于产生输入到 A/D 比较器的斜坡电压 (图 24-1)。

24.3.7 积分型 A/D 的分辨率、转化速度、电压范围和电容的选择

积分型 A/D 模块可允许对许多参数进行取舍。在进行转换时，用户需要进行以下取舍：

- 转换结果的分辨率
- 转换速度
- 模拟输入电压范围
- 外部电容

分辨率是 ADTMR 用来表示表征测得的输入电压的位数。分辨率会影响完成转换所需要的时间。表 24-2 给出了转换分辨率和最大转换时间之间的取舍。

积分型 A/D 转换器的转换时间可以用下面的公式计算：

转换时间 = (1/Fosc) x 2^N

其中， Fosc 是振荡频率， N 是所需分辨率的位数。

因此，频率为 4 MHz 时，16 位分辨率的转换时间为 16.384 ms。而 10 位的转换时间为 256 µs。

表 24-1: ADTMR 的初始值和转换时间

分辨率位数	装入 ADTMR 中的值	最大转换时间		
		周期	20 MHz	4 MHz
16	0000h	65536 TOSC	3.28 ms	16.38 ms
15	8000h	32768 TOSC	1.64 ms	8.2 ms
14	C000h	16384 TOSC	820 µs	4.1 ms
13	E000h	8192 TOSC	410 µs	2.05 ms
12	F000h	4096 TOSC	204.8 µs	1.03 ms
11	F800h	2048 TOSC	102.4 µs	500 µs
10	FC00h	1024 TOSC	51.2 µs	250 µs

外部电容的选择由所希望得到的应用特性决定，包括：

- 输入电压范围 (所有输入通道的最大范围)
- 转换时间
- 可编程电流源的输出值

上述参数的选择应能最大限度地缩短比较器跳变 (ADCIF 被置 1) 和 ADTMR 溢出 (OVFIF 被置 1) 的时间，这将确保整个 ADTMR 计数范围都被用于 A/D 转换过程。

用于选择斜坡电容值的公式是：

电容 = (转换时间（单位：秒）) X
(电流源输出（单位：安培）) / (满量程电压（单位：伏特）)

表 24-2 给出了电容值与所需的积分型 A/D 的分辨率、转换时间和满量程电压之间的关系。

为了得到最佳转换结果，CDAC 引脚上的电容应具有较低电压系数，如聚四氟乙烯、聚丙烯或聚苯乙烯电容等。在每次转换周期开始时，必须通过对 ADRST 位置 1 将外部电容放电。ADRST 位被置 1 的时间取决于外部电容的完全放电特性。

表 24-2: 外部电容的选择 （@ 4 MHz）

积分型 A/D 分辨率（位）	转换时间（ms）	满量程电压 （V）	积分型 A/D 电流源选择		CDAC 电容的计算值	最接近标准值的 CDAC 电容
			ADDAC3:ADDC0	典型输出 （μA）		
16	16.384	3.5	1100	27	0.126 μF	0.12 μF
		2.0	1010	22.5	0.184 μF	0.18 μF
		1.5	1011	24.75	0.270 μF	0.27 μF
14	4.096	3.5	1101	29.25	34 nF	33 nF
		2.0	1011	24.75	50.7 nF	47 nF
		1.5	1100	27	73.7 nF	68 nF
12	1.024	3.5	1101	29.25	8.56 nF	8.2 nF
		2.0	1001	20.25	10.4 nF	10 nF
		1.5	1010	22.5	15.4 nF	15 nF
10	0.256	3.5	1011	24.75	1.81 nF	1.8 nF
		2.0	1010	22.5	2.88 nF	2.7 nF
		1.5	1011	24.75	4.22 nF	3.9 nF

24.4 其它模拟模块

对于混合信号的应用场合，需要额外的模拟模块，包括：

- 带隙参考电压源
- 斜率参考电压分压器

24.4.1 带隙参考电压源

带隙参考电压电路用于为积分型 A/D、低压检测器和斜率参考电压分压器产生一个 1.2V 标称稳定的参考电压。模拟 MUX 上具备带隙参考电压源。要使能带隙参考电压，REFOFF (SLPCON<5>) 位必须清零。任何积分型 A/D 转换均必须使能带隙参考电压。

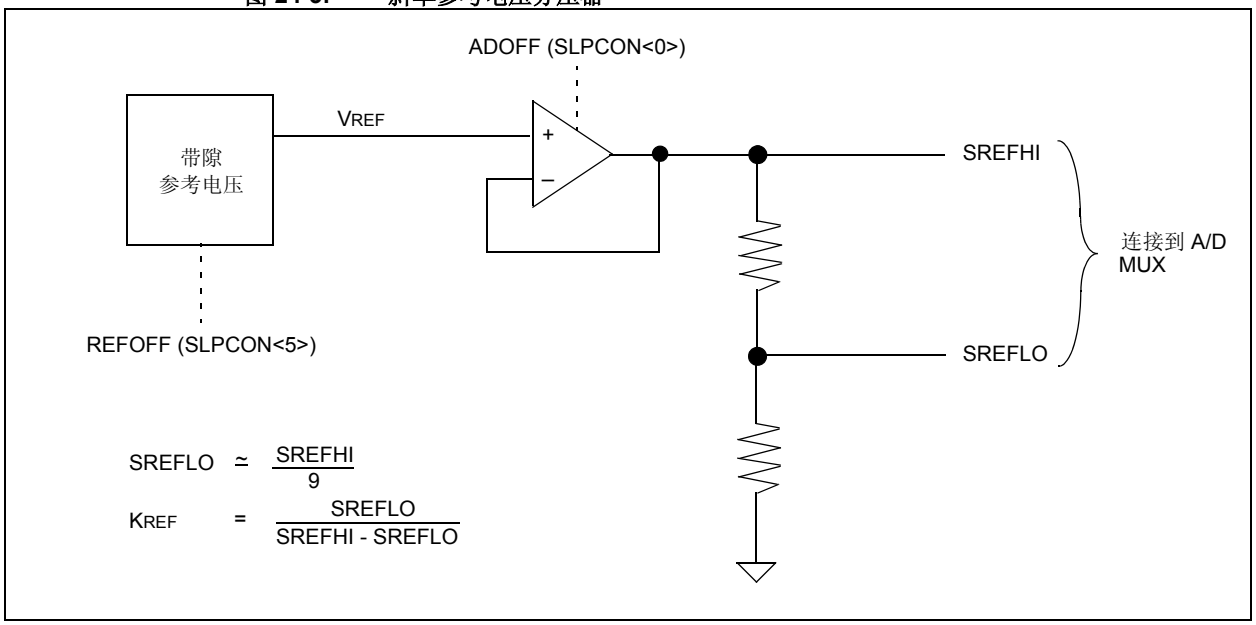
带隙参考电压的校正因数存储在 EPROM 校准空间中。

24.4.2 斜率参考电压分压器

由一个缓冲放大器和电阻分压器构成的斜率参考电压分压器电路，连接到内部带隙参考电压上，以产生另外两个参考电压，称为 SREFHI 和 SREFLO (见图 24-3)。SREFHI 的标称值和带隙电压相同（1.2V），SREFLO 的标称值为 0.13V。这两个参考电压可在两个模拟通道上得到。积分型 A/D 模块和固件可测量 SREFHI 和 SREFLO 电压，并结合 KREF 和 KBG 校准参数，对 ADC 的偏置和斜率误差进行校正。

详细的内容见 AN624 文档。

图 24-3: 斜率参考电压分压器



24.5 校准参数

积分型 A/D 模块有几个模拟组件。象所有的 CMOS 电路一样，其参数值随着制造工艺、温度、电压和时间的变化而变化。器件在设计时已尽量减小这些影响。此外，器件及其所带的积分型A/D模块，在工厂测试时都通过测量关键参数进行校准，并将其存储到 EPROM 的指定地址。用户的应用程序可访问这些校准数据，并利用它们对器件的参数差异进行数学补偿。

总地来说，这些数据值被称作校准常数。表 24-3 中列出了这些校准常数。32 位浮点格式中有一个字节表示指数，另外 3 个字节表示尾数。有关浮点算法的信息，请参见 AN575 文档。

表 24-3: 校准常数

参数	符号	字节数	值的表示
A/D 斜率参考率	KREF	4	32 位浮点
带隙参考电压	KBG	4	32 位浮点

有关校准参数的更多信息，请参见 AN624。

24.5.1 使用校准数据

为了获得最好的转换精度，应用的固件应使用这些校准常数。 KREF 和 KBG 被用于 A/D 转换。

24.5.2 参数偏差

表 24-4 列出了当没有使用校准数据时的 “最大参数偏差”以及 “校准后期望的参数偏差”。

如果未经校准的精确度已足够满足任务要求，则无需再对模块进行校准。如果要求更高的精确度，则必须使用校准参数。

表 24-4: 参数偏差

符号	参数	未经校准的最大偏差	经校准后可达到的偏差
KREF	A/D 斜率参考率	+/- 2.2%	+/- 0.13%
KBG	带隙参考电压	+/- 4.2%	+/- 0.058%

24.5.3 器件编程

24.5.3.1 无窗型器件

无窗型器件可以象任何 PIC16CXXX 处理器一样编程。在厂家校准时，已将校准区域写保护。

24.5.3.2 窗口型器件

注：
不得将窗口型器件写保护。如果器件被紫外线擦除，校准参数将会丢失，若器件被写保护，就无法将标准参数再烧写进去。

在擦除一个窗口型器件前，应将校准数据读出并保存起来。校准数据一旦丢失就无法重新创建，因而无法再对器件进行校准。

24.6 设计技巧

问 1: *你们推荐哪些类型的电容?*

答 1:

从价格、供货和性能等方面综合考虑，聚丙烯薄膜电容器是不错的选择。

问 2: *能否推荐几个电容器供应商?*

答 2:

推荐的供应商如下：

Southern Electronics Company
电话 : 1-203-876-7488

问 3: *我使用了推荐的电容器和表 24-2 列出的可编程电流源，但 A/D 输入范围却不匹配。*

答 3:

该表仅供最初使用时参考，但它不包括由于器件的工作频率偏离 4 MHz 而导致的偏差，或由于制造工艺和环境温度所引起的外部电容的容差和可编程电流源的偏差。

带隙参考电压的转换结果可用来判断应如何调整可编程电流源的输出，以确保正确的 A/D 满量程转换。 PICDEM™-14A 演示板上有代码示例 （见 P14_RV10.ASM 程序中的 ad_optimize 子程序）显示如何进行调整。

问 4: *我使用的 PIC14C000 片内有温度传感器，传感器的结果好象有点偏高。*

答 4:

这可能是由 DIE 的自热所致。引起 DIE 自热的原因有几种，包括：

- I/O 端口的拉 / 灌电流太大
- 器件运行时的功耗
  (记住 PIC14C000 能够在休眠模式下运行)
- 封装类型，这是因为封装的结温到周围温度的温度系数可导致自热

为了得到最佳结果，应使器件保持较低的功耗。器件的校准是在其低功耗状态下进行的。

问 5: *我的 A/D 转换结果似乎会受到电路板上高电流组件工作的影响。如何将这种影响降到最低限度?*

答 5:

板上的高电流组件可能产生接地走线或接地面的电势差。为了最大限度地减小影响，您应在应用电路板上使用两个接地系统。第一个是模拟接地，供参考模拟信号 （积分型 A/D 外部电容的接地，电阻分压器的接地等）使用。高电流或任何数字电源回路不得通过该模拟接地。

第二个是数字接地，供系统中所有其它数字逻辑使用。应用中的数字逻辑电路产生的噪声会注入这个接地端。应使用适当的接地技术来减小该噪声。

这两个接地端连接到 PICmicro® 单片机的接地引脚上。理想状况下，应使用两个独立的的接地面实现这两个接地。但多数情况下，依然可通过双层电路板来实现。其中一层用于实现这两个接地系统，并用一条缝将两个接地面隔开。另一层作为走线层。

24.7 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，通过适当修改并受到一定限制即可使用。目前与积分型 A/D 相关的应用笔记是：

标题	应用笔记 #
PIC14C000 Calibration Parameters	AN621
PIC14C000 A/D Theory and Implementation	AN624
Lead Acid Battery Charger Implementation using the PIC14C000	AN626

24.8 版本历史

版本 A

这是描述积分型 A/D 模块的初始发行版。

第 25 章 LCD

目录

本章主要包括以下一些内容：

25.1 简介 25-2
25.2 控制寄存器 25-3
25.3 LCD 定时 25-6
25.4 LCD 中断 25-12
25.5 像素控制 25-13
25.6 电压发生器 25-15
25.7 休眠模式下的操作 25-16
25.8 复位的影响 25-17
25.9 LCD 模块的设置 25-17
25.10 判别比 25-18
25.11 LCD 电压发生器 25-20
25.12 对比度 25-22
25.13 LCD 玻璃基板 25-22
25.14 初始化 25-23
25.15 设计技巧 25-24
25.16 相关应用笔记 25-25
25.17 版本历史 25-26

25.1 简介

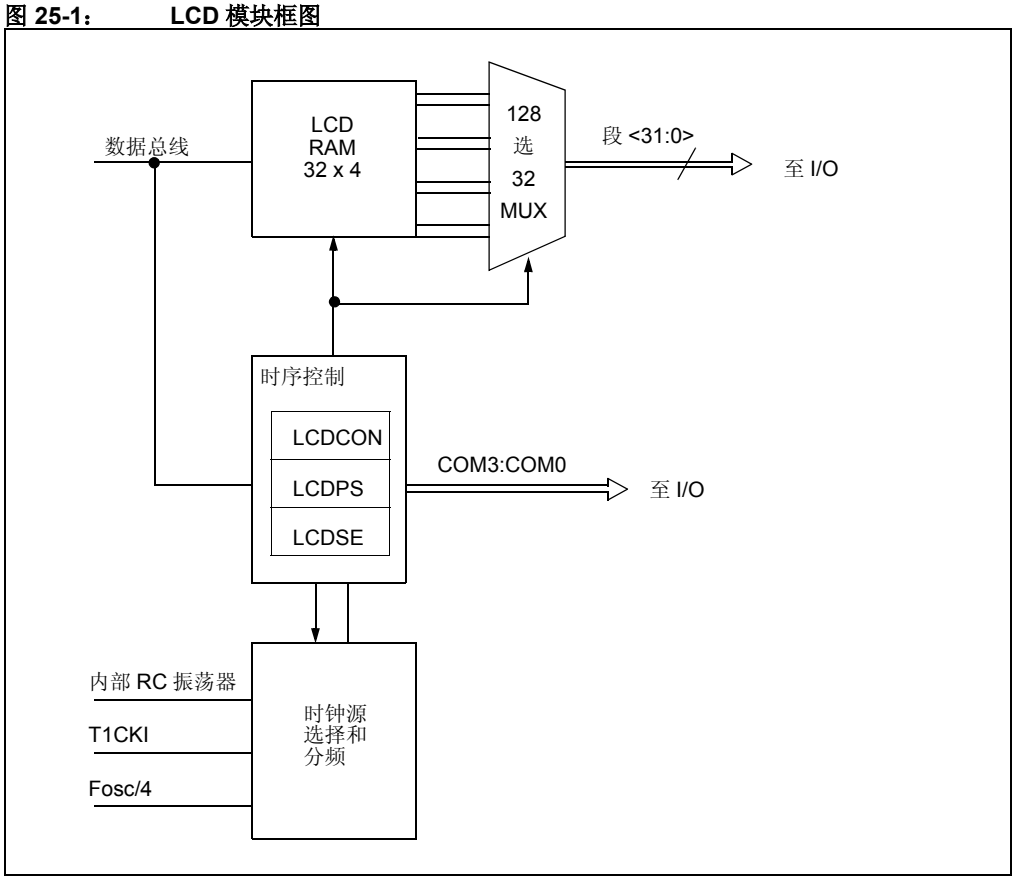
LCD 模块产生时序控制以驱动一个静态的或复用的 LCD 显示，最多可支持 32 段和 4 位公共端。它也提供 LCD 像素数据的控制。

模块的接口由 LCDCON、LCDSE 和 LCDPS 三个控制寄存器和 16 个 LCD 数据寄存器（LCD00-LCD15）组成。控制寄存器定义对 LCD 面板的时序要求，数据寄存器存储像素数据的数组。为运行正常，需配置控制寄存器以满足 LCD 面板的显示。初始化主要包括：选择 LCD 面板所需的公共端和段数量，然后指定供面板使用的 LCD 帧的时钟速率。

对 LCD 模块初始化后，通过将 LCD 数据寄存器的某一位清零 / 置 “1”，可以控制一个像素的显示。

模块设置完成后， LCDEN 位 （LCDCON<7>）可使能或禁止 LCD 模块。当清零 SLPEN 位（LCDCON<6>）时， LCD 面板在休眠状态下也能工作。

图 25-1： LCD 模块框图



25.2 控制寄存器

寄存器 25-1： LCDCON 寄存器

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDEN	SLPEN	—	VGEN	CS1	CS0	LMUX1	LMUX0
bit 7							bit 0

bit 7 **LCDEN**：模块驱动使能位
1 = 使能 LCD 驱动
0 = 禁止 LCD 驱动

bit 6 **SLPEN**：LCD 显示休眠使能位
1 = LCD 模块在休眠状态下停止操作
0 = LCD 模块在休眠状态下将继续显示

bit 5 **未用位**：读为 “0”

bit 4 **VGEN**：电压发生器使能位
1 = 使能内部 LCD 电压发生器 （上电）
0 = 关闭内部 LCD 电压发生器，电压由外部提供

bit 3:2 **CS1:CS0**：时钟源选择位
00 = Fosc/256
01 = T1CKI （Timer1）
1x = 内部 RC 振荡器

bit 1:0 **LMUX1:LMUX0**：公共端选择位
指定公共端数量和偏置方法

LMUX1:LMUX0	复用模式		偏置	最大段数
00	静态	(COM0)	静态	32
01	1/2	(COM0, 1)	1/3	31
10	1/3	(COM0, 1, 2)	1/3	30
11	1/4	(COM0, 1, 2, 3)	1/3	29

图注		
R = 可读位	W = 可写位	
U = 未用位，读为 “0”		- n = 上电复位时的值

寄存器 25-2： LCDPS 寄存器

U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	—	LP3	LP2	LP1	LP0
bit 7							bit 0

bit 7:4 **未用位，**读为 “0”

bit 3:0 **LP3:LP0**：帧时钟预分频比选择位

LMUX1:LMUX0	复用模式	帧频率
00	静态	时钟源 / (128 * (LP3:LP0 + 1))
01	1/2	时钟源 / (128 * (LP3:LP0 + 1))
10	1/3	时钟源 / (96 * (LP3:LP0 + 1))
11	1/4	时钟源 / (128 * (LP3:LP0 + 1))

图注		
R = 可读位	W = 可写位	
U = 未用位，读为 “0”		- n = 上电复位时的值

寄存器 25-3： 通用 LCDD （像素数据）寄存器的布局

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc	SEGs COMc
bit 7							bit 0

bit 7:0 **SEGsCOMc**：段 s 和公共端 c 的像素数据位

1 = 像素开 （变黑而显示）
0 = 像素关 （不显示）

图注		
R = 可读位	W = 可写位	
U = 未用位，读为 “0”		- n = 上电复位时的值

寄存器 25-4： LCDSE 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
SE29	SE27	SE20	SE16	SE12	SE9	SE5	SE0
bit 7							bit 0

bit 7 **SE29**：COM1/SEG31 - COM3/SEG29 的引脚功能选择
1 = 引脚有 LCD 段驱动功能
0 = 引脚为数字输入

注： LMUX1:LMUX0 的设置优先于 SE29 位的设置，这使得引脚成为公共驱动。

bit 6 **SE27**：SEG28 和 SEG27 的引脚功能选择位
1 = 引脚有 LCD 段驱动功能
0 = 引脚为数字输入

bit 5 **SE20**：SEG26 - SEG20 的引脚功能选择位
1 = 引脚有 LCD 段驱动功能
0 = 引脚为数字输入

bit 4 **SE16**：SEG19 - SEG16 的引脚功能选择位
1 = 引脚有 LCD 段驱动功能
0 = 引脚为数字输入

bit 3 **SE12**：SEG15 - SEG12 的引脚功能选择位
1 = 引脚有 LCD 段驱动功能
0 = 引脚为数字输入

bit 2 **SE9**：SEG11 - SEG09 的引脚功能选择位
1 = 引脚有 LCD 段驱动功能
0 = 引脚为数字输入

bit 1 **SE5**：SEG08 - SEG05 的引脚功能选择位
1 = 引脚有 LCD 段驱动功能
0 = 引脚为数字输入

bit 0 **SE0**：SEG04 - SEG00 的引脚功能选择位
1 = 引脚有 LCD 段驱动功能
0 = 引脚为数字 I/O

图注		
R = 可读位	W = 可写位	
U = 未用位，读为 “0”		- n = 上电复位时的值

注： 上电复位时， LCD 引脚被自动初始化为 LCD 驱动。

25.3 LCD 定时

LCD 模块有三种可能的时钟源输入，并支持静态、 1/2、 1/3 和 1/4 复用。

25.3.1 定时时钟源的选择

LCD 定时发生器的时钟源有：

- 内部 RC 振荡器　　适用于低频率的器件或在休眠模式下使用
- Timer1 振荡器　　适用于低频率的器件或在休眠模式下使用
- 系统时钟 /256

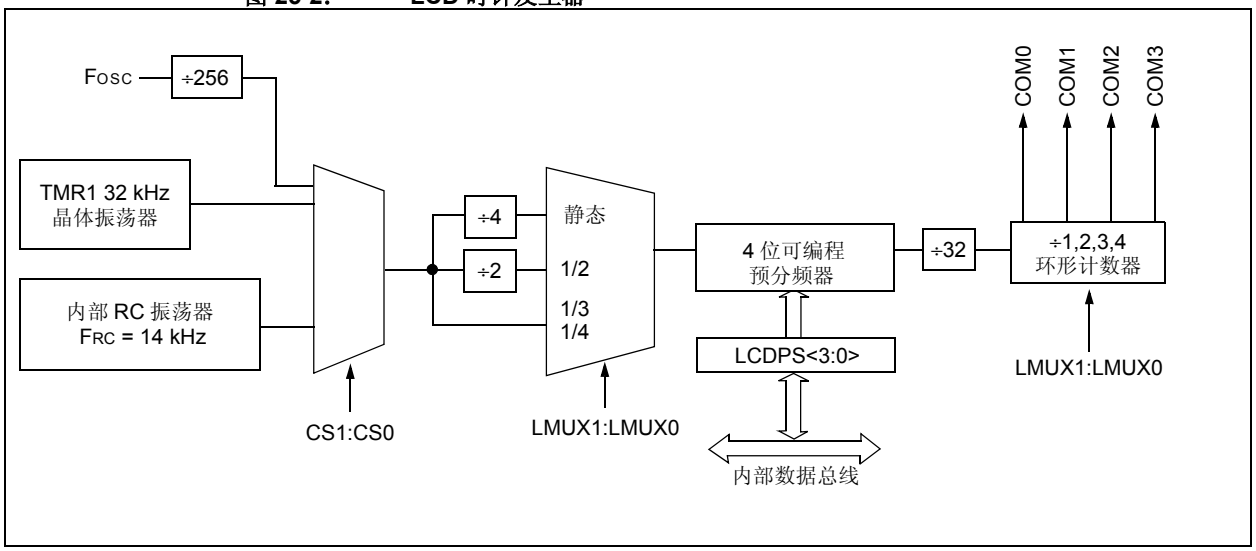
第一个定时钟源是内部 RC 振荡器，频率为 14kHz。这个振荡器提供了一个低速时钟，此时钟可以在休眠模式时继续运行 LCD。当没有选择 RC 振荡器或者 LCD 模块未使能时， RC 振荡器将断电停止工作。

第二个时钟源是 Timer1 外部振荡器。这个振荡器提供一个低速时钟，此时钟可以在休眠模式时使 LCD 继续运行，频率为 32 kHz。为使 Timer1 振荡器作为 LCD 模块时钟源，必须把 T1OSCEN（T1CON<3>）位置 “1”。

第三个时钟源是系统时钟的 256 分频。当外部振荡器是 8 MHz 时，将提供 32 kHz 的输出。分频器是不可编程的。实际上，是用 LCDPS 寄存器来设置 LCD 帧时钟的分频。

时钟源可以用位 CS1:CS0 （LCDCON<3:2>）来选择。请参见寄存器 25-1，了解该寄存器编程的详细信息。

图 25-2： LCD 时钟发生器



25.3.2 复用的定时发生器

对于各种显示模式，定时发生电路将产生 1 到 4 个公共时钟，具体模式由 LMUX1:LMUX0 位（LCDCON<1:0>）指定。表 25-1 显示了计算帧频率的公式。

表 25-1： 帧频率公式

复用模式	帧频率
静态	时钟源 / (128 * (LP3:LP0 + 1))
1/2	时钟源 / (128 * (LP3:LP0 + 1))
1/3	时钟源 / (96 * (LP3:LP0 + 1))
1/4	时钟源 / (128 * (LP3:LP0 + 1))

表 25-2： 使用 Timer1 （频率为 32.768kHz）或者 Fosc （频率为 8 MHz）的近似帧频率（Hz）

LP3:LP0	静态	1/2	1/3	1/4
2	85	85	114	85
3	64	64	85	64
4	51	51	68	51
5	43	43	57	43
6	37	37	49	37
7	32	32	43	32

表 25-3： 使用内部 RC 振荡器 （频率为 14 kHz）的近似帧频率 （Hz）

LP3:LP0	静态	1/2	1/3	1/4
0	109	109	146	109
1	55	55	73	55
2	36	36	49	36
3	27	27	36	27

图 25-3： 静态驱动波形

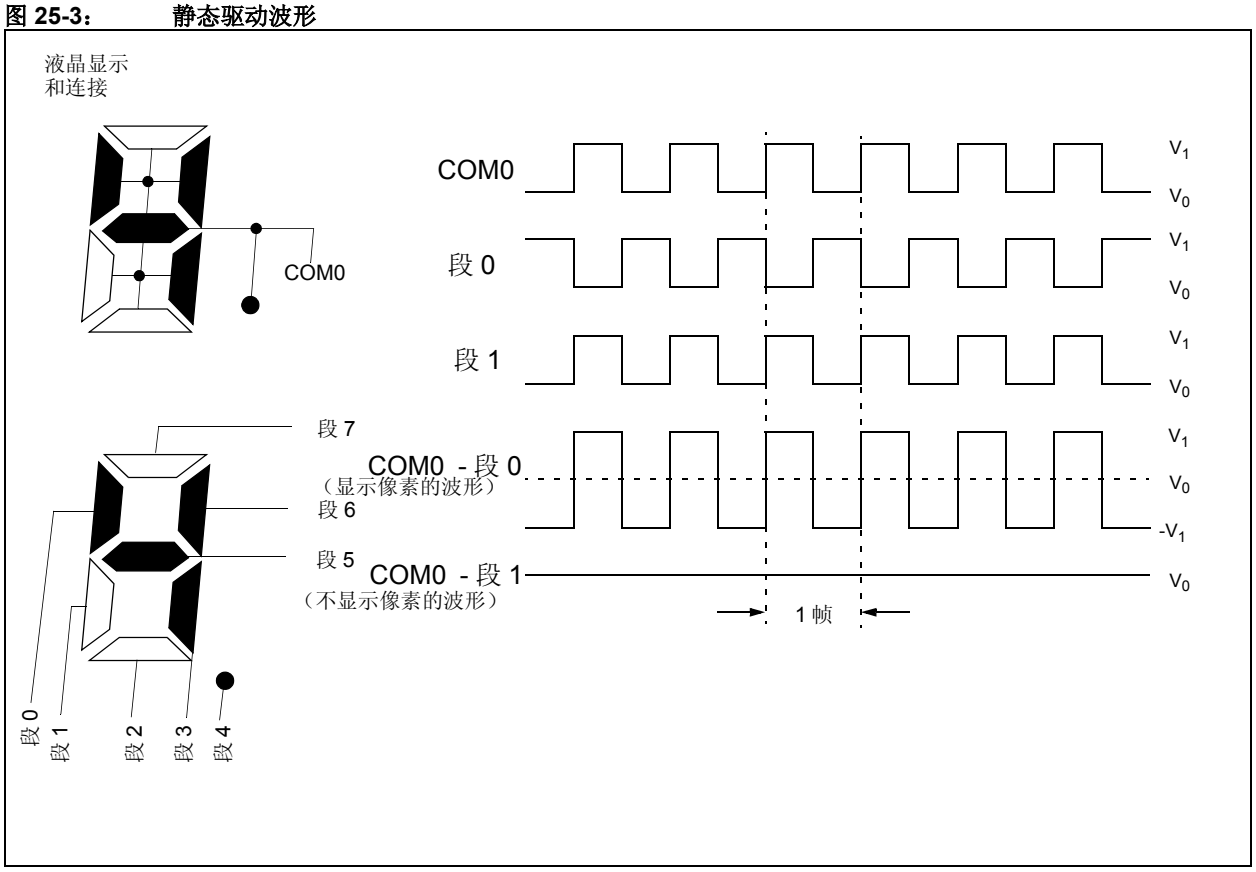


图 25-4： 1/2 MUX， 1/3 偏置波形

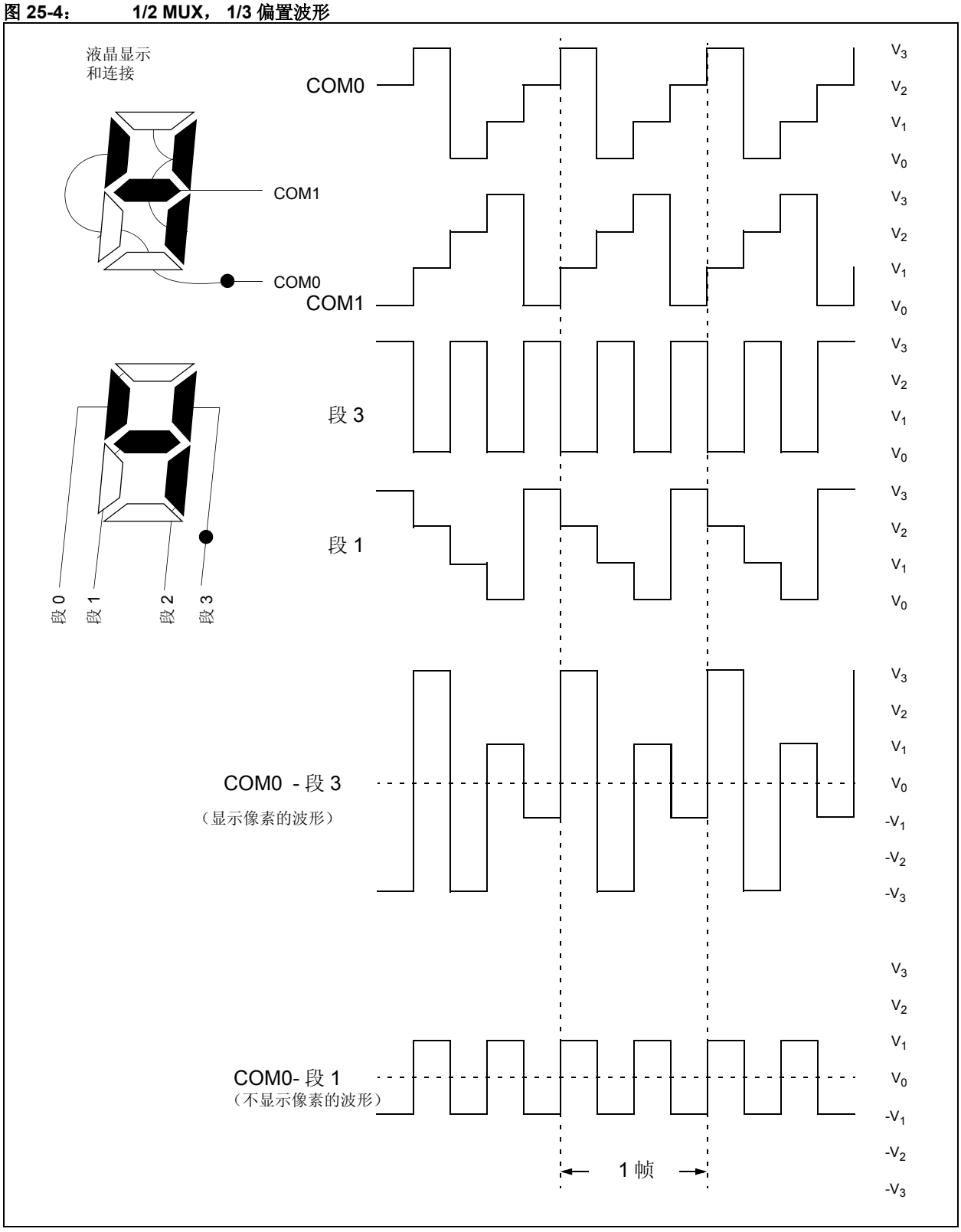


图 25-5： 1/3 MUX， 1/3 偏置波形

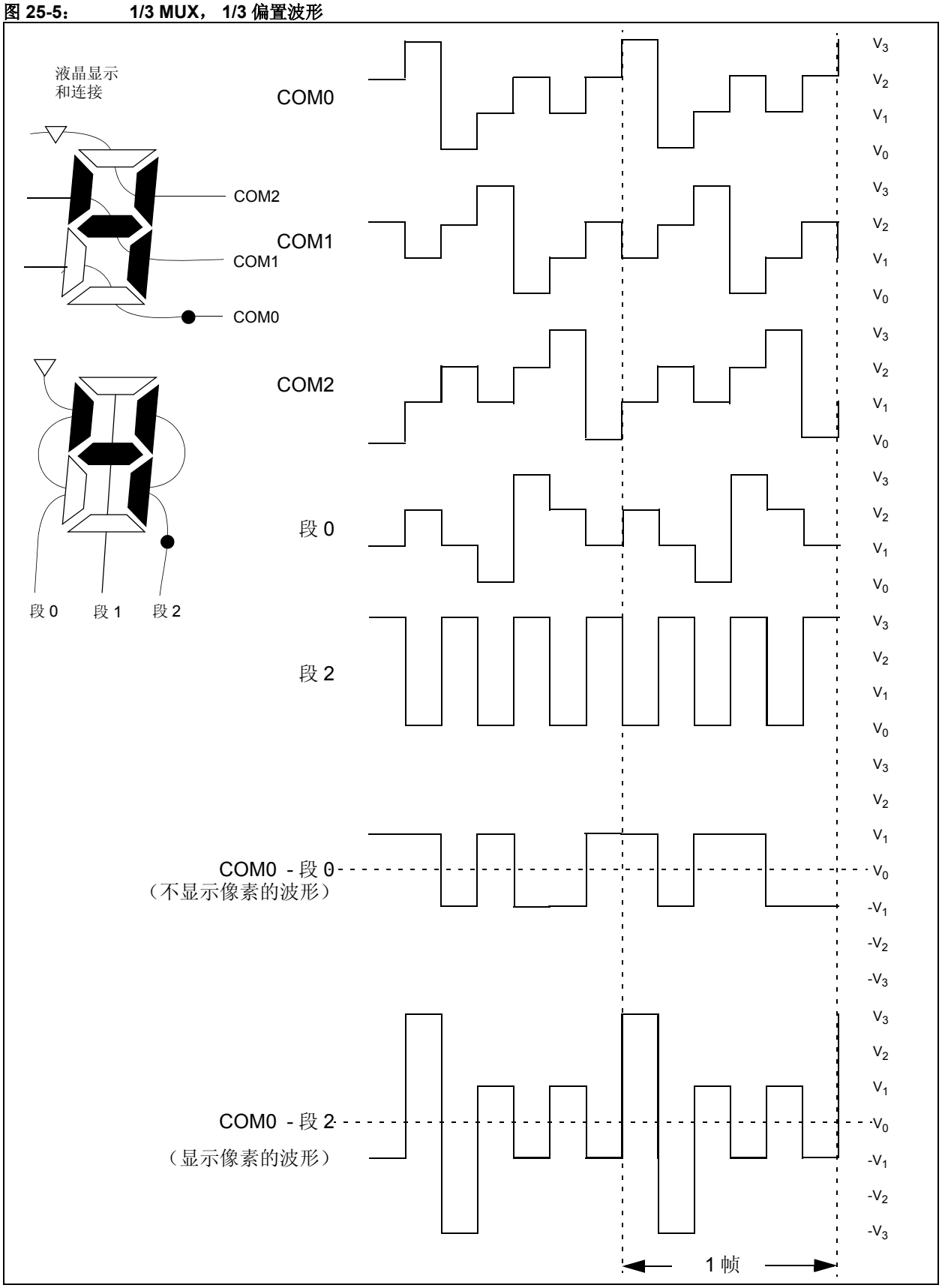
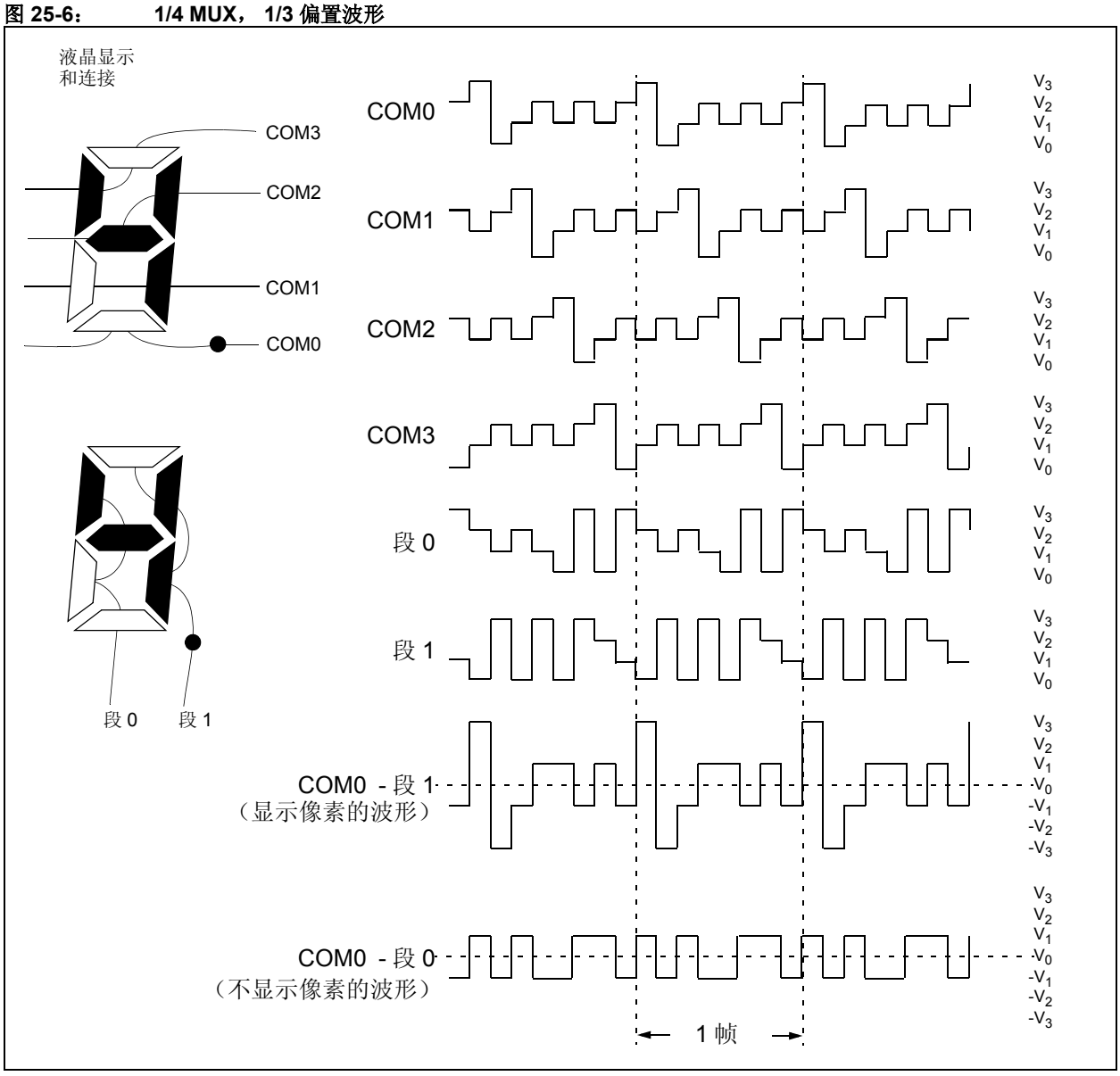


图 25-6： 1/4 MUX， 1/3 偏置波形

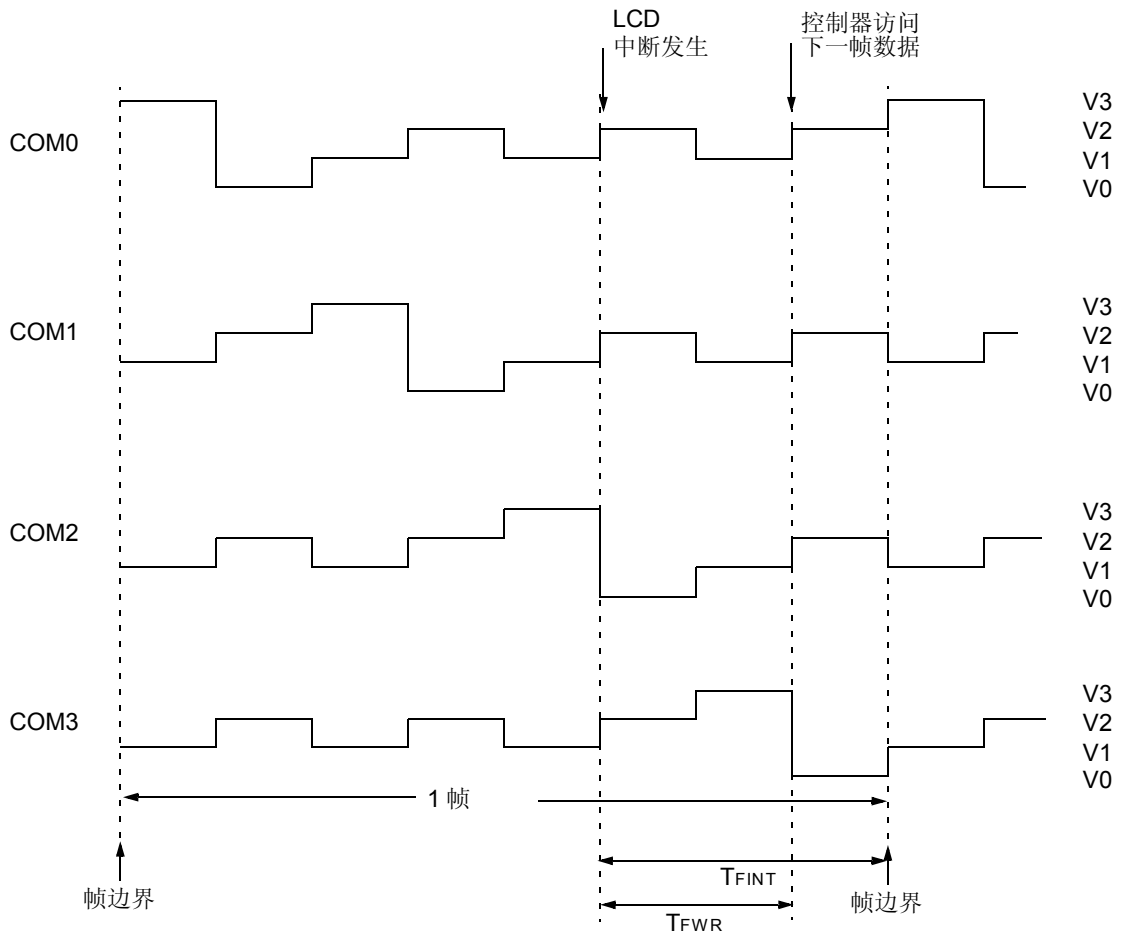


25.4 LCD 中断

LCD 定时发生器提供一个中断，该中断用于定义 LCD 的帧时序。它可在一个新帧开始时写入像素数据。在帧边界处写入像素数据可使图像过渡更清晰。这个中断也用于 LCD 和外部事件的同步。例如：与外部段驱动器的接口 （如 Microchip 的 AY0438），使更新的段数据与 LCD 帧同步。

一个新帧开始于 COM0 公共信号的前沿。在 LCD 控制器完成对某一帧所需的所有像素数据的访问后，将立刻产生中断。该中断发生在帧边界前的某个固定时间，如图 25-7 所示。在中断发生的 TFWR 时间后， LCD 控制器开始访问下一帧的数据。

图 25-7： 1/4 MUX 驱动的波形



TFWR = TFRAME/(LMUX1:LMUX0 + 1)

TFINT = (TFWR /2 - (2TCY + 40 ns)) → 最小
(TFWR /2 - (1TCY + 40 ns)) → 最大

25.5 像素控制

25.5.1 LCDD （像素数据）寄存器

像素寄存器决定每个像素的状态，每位都定义一个单独的像素。

表 25-4 显示了 LCDD 寄存器各位和段信号之间的关系。

任何没有用于显示的 LCD 像素位置都可以用作通用 RAM。

表 25-4： LCDD 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	其它复位值
LCDD00	SEG07 COM0	SEG06 COM0	SEG05 COM0	SEG04 COM0	SEG03 COM0	SEG02 COM0	SEG01 COM0	SEG00 COM0	xxxx xxxx	xxxx xxxx
LCDD01	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG09 COM0	SEG08 COM0	xxxx xxxx	xxxx xxxx
LCDD02	SEG23 COM0	SEG22 COM0	SEG21 COM0	SEG20 COM0	SEG19 COM0	SEG18 COM0	SEG17 COM0	SEG16 COM0	xxxx xxxx	xxxx xxxx
LCDD03	SEG31 COM0	SEG30 COM0	SEG29 COM0	SEG28 COM0	SEG27 COM0	SEG26 COM0	SEG25 COM0	SEG24 COM0	xxxx xxxx	xxxx xxxx
LCDD04	SEG07 COM1	SEG06 COM1	SEG05 COM1	SEG04 COM1	SEG03 COM1	SEG02 COM1	SEG01 COM1	SEG00 COM1	xxxx xxxx	xxxx xxxx
LCDD05	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG09 COM1	SEG08 COM1	xxxx xxxx	xxxx xxxx
LCDD06	SEG23 COM1	SEG22 COM1	SEG21 COM1	SEG20 COM1	SEG19 COM1	SEG18 COM1	SEG17 COM1	SEG16 COM1	xxxx xxxx	xxxx xxxx
LCDD07	SEG31 COM1 (1)	SEG30 COM1	SEG29 COM1	SEG28 COM1	SEG27 COM1	SEG26 COM1	SEG25 COM1	SEG24 COM1	xxxx xxxx	xxxx xxxx
LCDD08	SEG07 COM2	SEG06 COM2	SEG05 COM2	SEG04 COM2	SEG03 COM2	SEG02 COM2	SEG01 COM2	SEG00 COM2	xxxx xxxx	xxxx xxxx
LCDD09	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG09 COM2	SEG08 COM2	xxxx xxxx	xxxx xxxx
LCDD10	SEG23 COM2	SEG22 COM2	SEG21 COM2	SEG20 COM2	SEG19 COM2	SEG18 COM2	SEG17 COM2	SEG16 COM2	xxxx xxxx	xxxx xxxx
LCDD11	SEG31 COM2 (1)	SEG30 COM2 (1)	SEG29 COM2	SEG28 COM2	SEG27 COM2	SEG26 COM2	SEG25 COM2	SEG24 COM2	xxxx xxxx	xxxx xxxx
LCDD12	SEG07 COM3	SEG06 COM3	SEG05 COM3	SEG04 COM3	SEG03 COM3	SEG02 COM3	SEG01 COM3	SEG00 COM3	xxxx xxxx	xxxx xxxx
LCDD13	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG09 COM3	SEG08 COM3	xxxx xxxx	xxxx xxxx
LCDD14	SEG23 COM3	SEG22 COM3	SEG21 COM3	SEG20 COM3	SEG19 COM3	SEG18 COM3	SEG17 COM3	SEG16 COM3	xxxx xxxx	xxxx xxxx
LCDD15	SEG31 COM3 (1)	SEG30 COM3 (1)	SEG29 COM3 (1)	SEG28 COM30	SEG27 COM3	SEG26 COM30	SEG25 COM3	SEG24 COM3	xxxx xxxx	xxxx xxxx

注 1： 这些像素位不显示，但可作为通用 RAM 使用。

25.5.2 段驱动使能

LCDSE 寄存器用来选择成组引脚的功能，它可以选择每个引脚组是作为 LCD 驱动器还是作为数字引脚。要将引脚设置为数字端口，必须将 LCDSE 寄存器的相应位清零。

如果引脚作为数字输入，则相应的 TRIS 位控制数据方向。 LCDSE 寄存器中的任何位置为 “1” 将覆盖相应 TRIS 寄存器中对应位的设置。

注 1： 当上电复位时， LCD 引脚自动初始化为 LCD 驱动功能。
注 2： 对RD7、RD6和RD5引脚来说，LMUX1:LMUX0位的设置优先于LCDSE位的设置。

例 25-1： 32 段的静态驱动

```
    BCF   STATUS,RP0  ; Select Bank2
    BSF   STATUS,RP1   ;
    BCF   LCDCON,LMUX1 ; Select Static MUX
    BCF   LCDCON,LMUX0 ;
    MOVLW 0xFF         ; Make PortD,E,F,G LCD pins
    MOVWF LCDSE        ; configure rest of LCD
```

例 25-2： 13 段的 1/3 MUX 驱动

```
    BCF   STATUS,RP0   ; Select Bank2
    BSF   STATUS,RP1   ;
    BSF   LCDCON,LMUX1 ; Select 1/3 MUX
    BCF   LCDCON,LMUX0 ;
    MOVLW 0x87         ; Make PORTD<7:0> & PORTE<6:0> LCD pins
    MOVWF LCDSE        ; configure rest of LCD
```

25.6 电压发生器

LCD 的电压产生有两种方法，内部电荷泵或者外部梯形电阻网络。

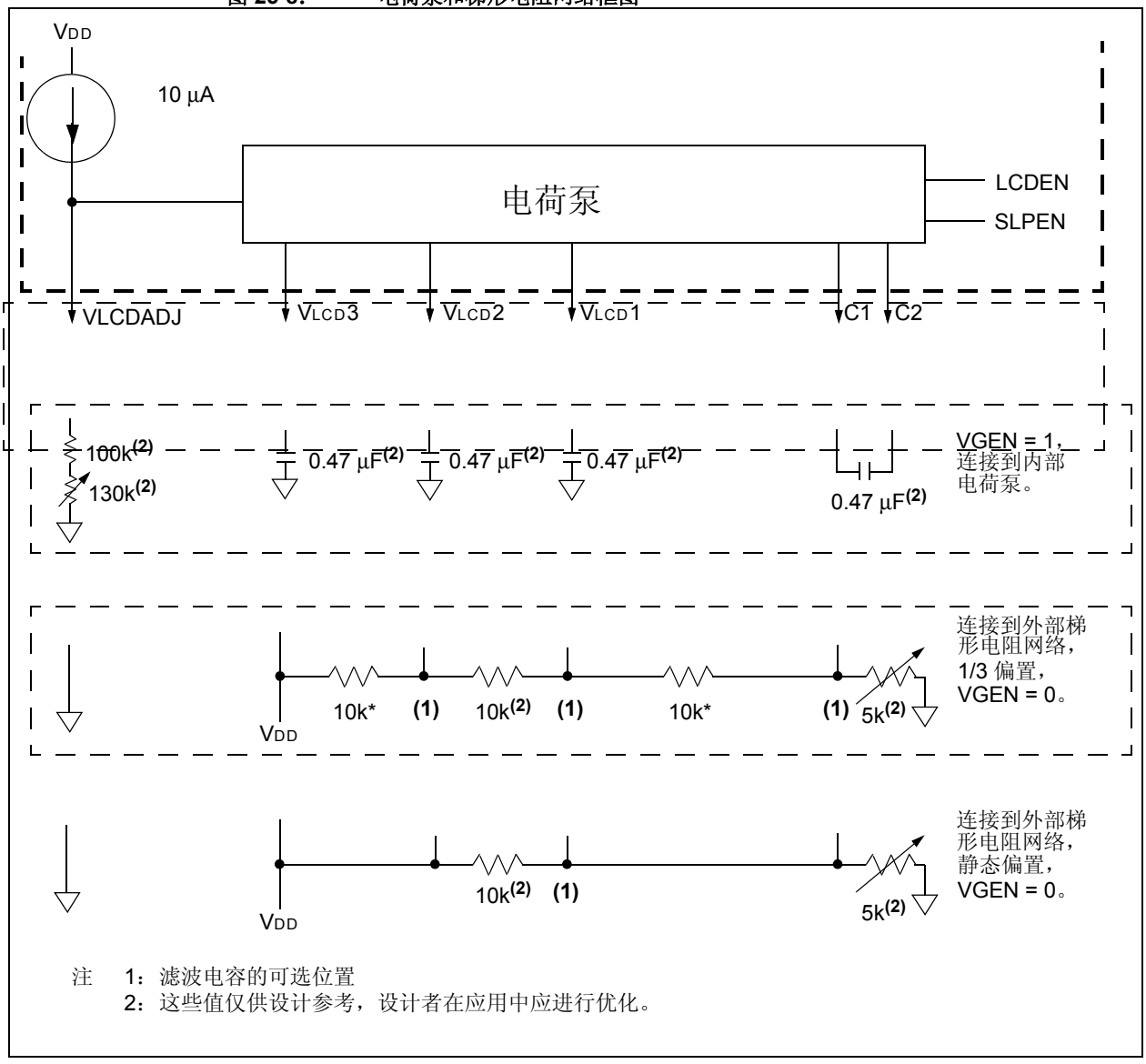
25.6.1 电荷泵

LCD 电荷泵如图 25-8 所示。1.0V 至 2.3V 的调节器将从变化的电池电压建立一个稳定的参考电压。调节器通过连接在 VLCDADJ 到地的外接可变电阻来调节。电位器为 LCD 提供对比度调节。这个参考电压连接到电荷泵的$V_{LCD}1$，电荷泵放大$V_{LCD}1$至$V_{LCD}2 = 2 * V_{LCD}1$和 $V_{LCD}3 = 3 * V_{LCD}1$。当电荷泵不工作时，$V_{LCD}3$在内部连向V_{DD}。关于电荷泵的电容和电位器数值可参考电荷泵的电气规范说明。

25.6.2 外部梯形电阻网络

VGEN (LCDCON<4>) 位清零时，LCD 模块用外部梯形电阻网络来产生 LCD 电压。图 25-8 显示如何在静态和 1/3 偏置情况下连接外部电阻网络。

图 25-8： 电荷泵和梯形电阻网络框图



注 1：滤波电容的可选位置
2：这些值仅供设计参考，设计者在应用中应进行优化。

25.7 休眠模式下的操作

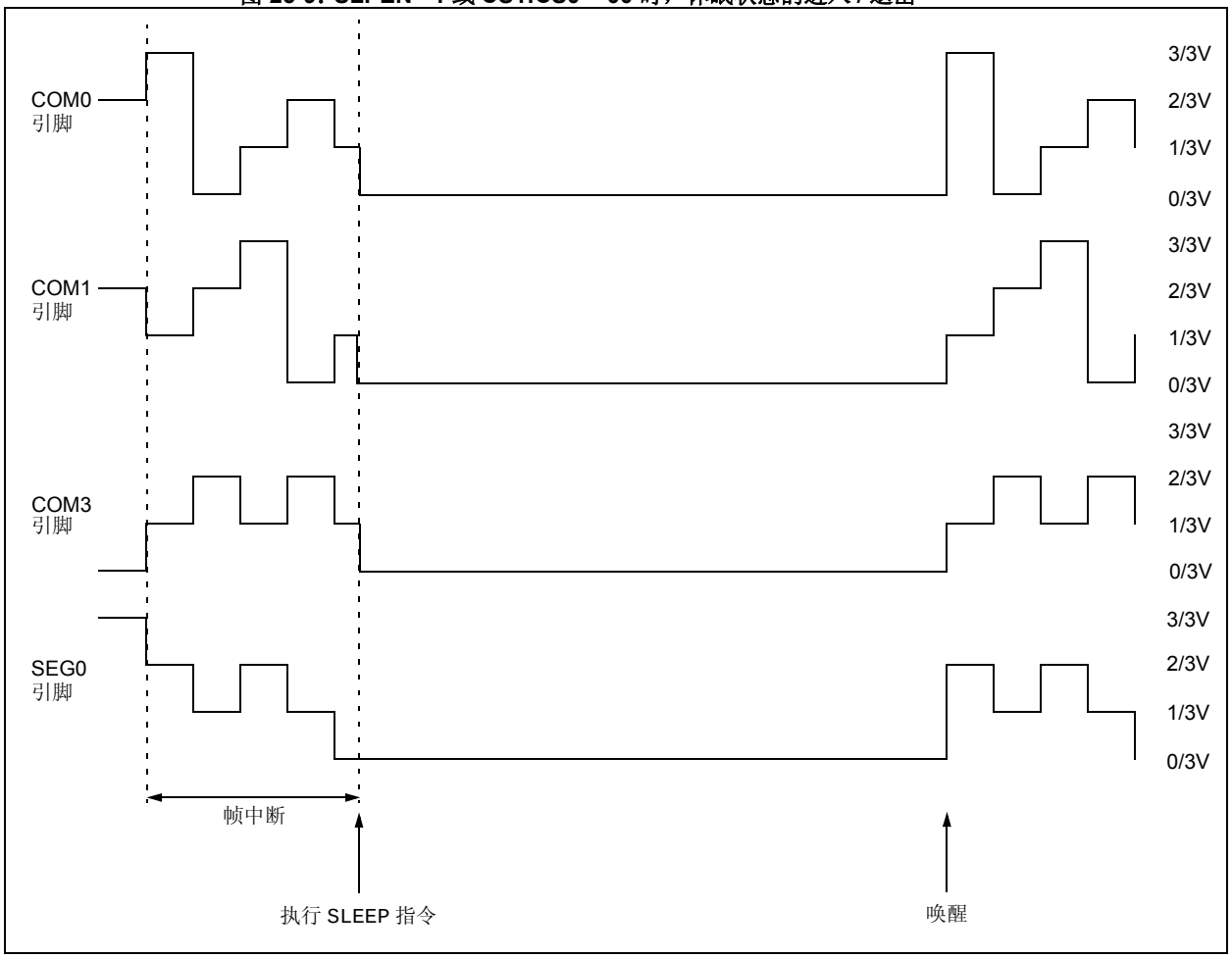
LCD 模块在休眠模式下也能工作。 SLPEN 位 （LCDCON<6>）对此进行控制。将 SLPEN 位置“1”允许 LCD 模块进入休眠模式，而 SLPEN 位清零允许 LCD 模块在休眠模式下继续工作。

如果执行 SLEEP 指令时，SLPEN = '1'，LCD 模块将停止所有的功能，进入一个极低的功耗模式。该模块将立即停止工作，并在段和公共引脚上输出最低的 LCD 电压，如图 25-9 所示。为确保 LCD 完成一帧的扫描，SLEEP 指令应该紧接着 LCD 帧边界后执行。LCD 中断能够确定帧边界。计算延迟的公式请参考 **25.4 节**。

如果执行 SLEEP 指令时， SLPEN = '0'，模块将继续显示 LCDD 寄存器当前的内容。为使模块在休眠模式下继续工作，时钟源必须是内部 RC 振荡器或 Timer1 外部振荡器。在休眠时， LCD 数据不能被改变。在这种模式下， LCD 模块的功耗并未降低，然而由于内核和其它外设功能的关闭，器件的总功耗将降低。

注： 当休眠时，必须使用内部 RC 振荡器或外部 Timer1 振荡器来操作 LCD 模块。

图 25-9：SLPEN =1 或 CS1:CS0 = 00 时，休眠状态的进入 / 退出



25.8 复位的影响

复位时，LCD 模块被关闭，但 LCD 引脚被设置为 LCD 驱动。这确保单片机不会因突然有 DC 电压加在 LCD 显示段上而损坏 LCD 玻璃基板。

25.9 LCD 模块的设置

下面是设置 LCD 模块的步骤：

1. 用 LP3:LP0 位 （LCDPS<3:0>）选择帧时钟预分频比。
2. 用 LCDSE 寄存器将适当的引脚设置为段驱动器。
3. 用 LCDCON 寄存器如下设置 LCD 模块。
   - LMUX1:LMUX0 位设置复用模式和偏置值
   - CS1:CS0 位设置时钟源
   - VGEN 位使能电压发生器
   - SLPEN 位使能休眠模式
4. 将初始数值写到像素数据寄存器 LCDD00 到 LCDD15。
5. 将 LCD 中断标志位 LCDIF 清零。如果需要，可以通过 LCDIE 置 “1” 使能该中断。
6. 通过 LCDEN 位 （LCDCON<7>）置 “1”，使能 LCD 模块。

25.10 判别比

判别比是一种计算 LCD 对比度的方法。第一个例子是关于图 25-3 中的静态驱动波形。电压 V_1 和 V_0 分别赋予 1 和 0 值。首先，列出像素点亮和未点亮时的 DC 和 RMS 电压公式，然后计算 DC、 RMS 和判别比。

例 25-3： 静态 MUX 的判别比计算

$$COMx - SEGx\ [ON] = 1 - 1, \quad V_{DC} = 0$$

$$COMx - SEGx\ [OFF] = 0 + 0, \quad V_{DC} = 0$$

$$V_{RMS}\ [ON] = \Delta V \sqrt{\frac{(1)^2 + (-1)^2}{2}} = 1\Delta V$$

$$V_{RMS}\ [OFF] = \Delta V \sqrt{\frac{(0)^2 + (0)^2}{2}} = 0\Delta V$$

$$D = \frac{V_{RMS}\ [ON]}{V_{RMS}\ [OFF]} = \frac{1\Delta V}{0\Delta V} = \infty$$

请参考图 25-3 静态驱动波形。

下一个例子针对图 25-6，一个 1/4 MUX 和 1/3 偏置时的波形。在这个例子中，将值 3、2、1 和 0 分别赋给 V_3、V_2、V_1 和 V_0。DC 电压，RMS 电压和判别比的计算如例 25-4 所示。

例 25-4：　1/4 MUX 时判别比的计算

COM0 - SEGx [ON] = 3 - 3 + 1 - 1 + 1 - 1 + 1 - 1　　V_{DC} = 0
COM0 - SEGx [OFF] = 1 - 1 - 1 + 1 - 1 + 1 - 1 + 1　　V_{DC} = 0

$$V_{RMS}\,[ON] = \Delta V\sqrt{\frac{(3)^2 + (-3)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2}{8}} = \sqrt{3}\,\Delta V$$

$$V_{RMS}\,[OFF] = \Delta V\sqrt{\frac{(1)^2 + (-1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2 + (-1)^2 + (1)^2}{8}} = \Delta V$$

$$D = \frac{V_{RMS}\,[ON]}{V_{RMS}\,[OFF]} = \frac{\sqrt{3}\,\Delta V}{1\,\Delta V} = 1.732$$

注：　请参考图 25-6

从这些例子可以看出，静态显示有很好的对比度。LCD 的复用比越高，判别比就越低，因此，显示的对比度就越小。

表 25-5 显示了各种 MUX 和偏置组合下的 V_{OFF}，V_{ON} 和判别比。

随着 LCD 的复用比增加，判别比将减小，显示的对比度也将减小，因此为了有较高的对比度，可以加大 V_{OFF} 和 V_{ON} 两种状态下的 LCD 驱动电压的差异。

表 25-5：　判别比与复用比和偏置的关系

	1/3 偏置		
	V_{OFF}	V_{ON}	D
静态	0	1	∞
1/2 MUX	0.333	0.745	2.236
1/3 MUX	0.333	0.638	1.915
1/4 MUX	0.333	0.577	1.732

25.11 LCD 电压发生器

在产生 LCD 电压的众多方法之中，有两种方法最常用：

- 梯型电阻网络
- 电荷泵。

梯型电阻网络方法，如图 25-10 所示，是最常用的产生较高 Vcc 电压的方法。这种方法用较便宜的电阻产生多级 LCD 电压。不管要驱动的像素数目有多少，它总能使激励电流保持恒定不变。V3 点的电压一般与 Vcc 相连（外部连接或内部连接皆可）。

电阻值的选择主要取决两个特性：显示性能和功耗。显示性能是 LCD 驱动波形的函数。因为 LCD 是一个容性负载，在充电和放电时波形将失真。这种失真能通过减小电阻值而减小，然而减小电阻，会加大流经电阻的电流，进而增加功耗。随着 LCD 尺寸的增加，必须减小电阻值以保持显示的图像质量不失真。

有时候在电阻两端并联电容可以减小因充放电引起的失真。该电容可充当电荷存储器，在驱动波形变化时提供额外的电流。一般， R 为 1 kΩ 到 50 kΩ，电位器为 5 kΩ 到 200 kΩ。

图 25-10： 梯型电阻网络

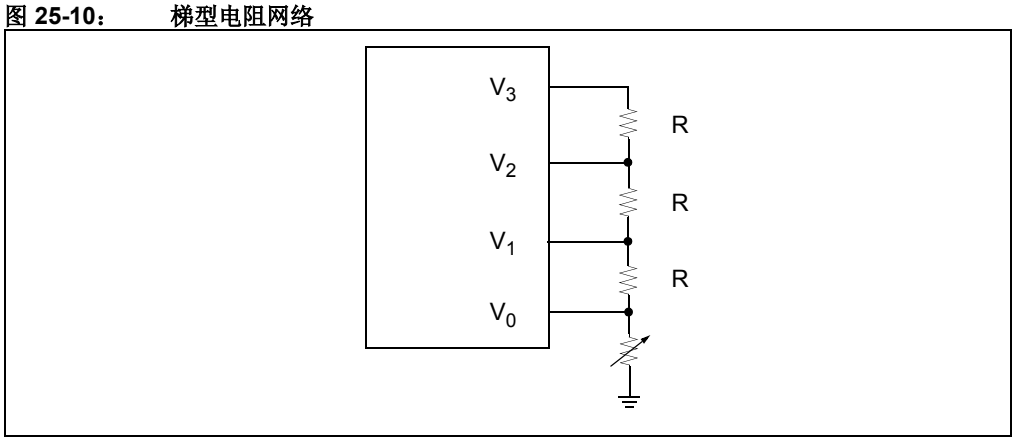
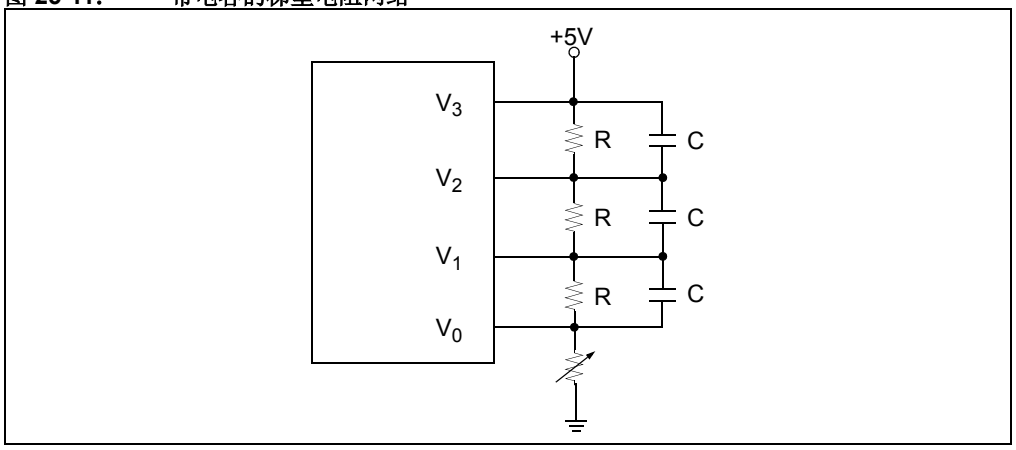
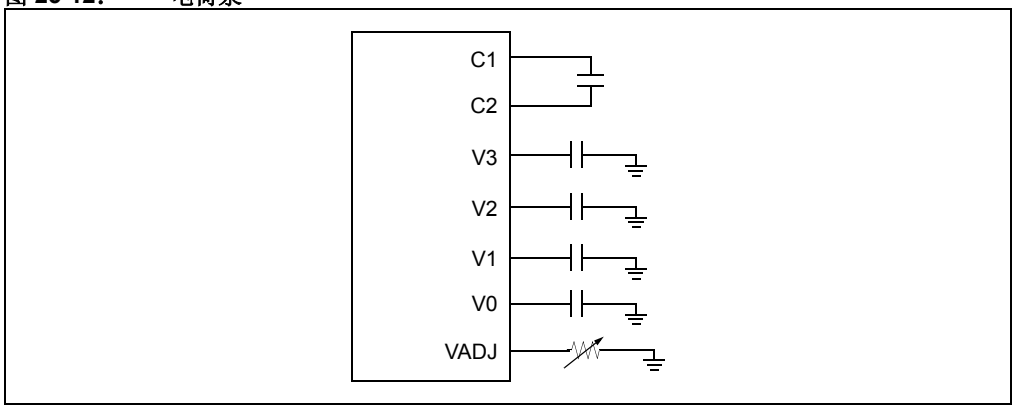


图 25-11： 带电容的梯型电阻网络



利用电荷泵可以将 VDD 电压提升到显示 LCD 所需的驱动电压，所以电荷泵很适合于低压电池的应用场合。如图 25-12 所示，对每组 LCD 电压，电荷泵都需要外接一个充电电容和滤波电容。这些电容通常是由低泄漏材料组成，例如聚酯、聚丙烯或聚苯乙烯。此外，电流的消耗与要驱动的像素数目成比例，这也是电荷泵适用于电池应用场合的另一因素。

图 25-12： 电荷泵



25.12 对比度

尽管对比度主要由光源和复用模式决定，但它也随 LCD 驱动电压的变化而变化。如前所述，电位器可以控制 LCD 显示的对比度。电位器可以调节每组 LCD 驱动电压之间差异，差异越大，对比度就越明显。

25.13 LCD 玻璃基板

LCD玻璃基板的特性取决于所采用的材料。附录 B列出了一些LCD制造商。若想了解所用基板材料的特性，请与他们联系。

25.14 初始化

例 25-5 所示的代码将初始化 LCD 模块，使所有段清零。

例 25-5： LCD 初始化

```
    BCF    PIR1,LCDIF      ; Clear LCD interrupt flag
    BCF    STATUS,RP0      ; Go to Bank2
    BSF    STATUS,RP1
    MOVLW  0x06            ; Set frame freq to ~37Hz
    MOVWF  LCDPS
    MOVLW  0xff            ; Make all pin functions LCD drivers
    MOVWF  LCDSE
    MOVLW  0x17            ; Drive during SLEEP, Charge pump enabled
    MOVWF  LCDCON          ; Timer1 clock source, 1/4 MUX
    CLRF   LCDD00          ; Clear all data registers to turn
    CLRF   LCDD01          ;    all pixels off
    CLRF   LCDD02
    CLRF   LCDD03
    CLRF   LCDD04
    CLRF   LCDD05
    CLRF   LCDD06
    CLRF   LCDD07
    CLRF   LCDD08
    CLRF   LCDD09
    CLRF   LCDD10
    CLRF   LCDD11
    CLRF   LCDD12
    CLRF   LCDD13
    CLRF   LCDD14
    CLRF   LCDD15
    BSF    PIE1,LCDIE       ; Enable LCD interrupts
    BSF    LCDCON,LCDEN     ; Enable LCD Module
    BCF    STATUS,RP1       ; Go to Bank0
```

25.15 设计技巧

问 1： ***我想使用一些 LCD 引脚作为输入。***

答 1：

要正确设置 LCDSE 寄存器中的控制位，因为这些位会覆盖相应 TRIS 位的设置。

问 2： ***LCD 显示总在闪烁。***

答 2：

您的帧频率可能太低。帧频率可以在 LCDPS 寄存器中更改。

问 3： ***LCD 的显示段看不太清楚。***

答 3：

这可能是由于 LCD 驱动电压不正确，解决方法有：

1. 如果您使用梯型电阻网络，尝试改变 R 的数值或梯型电阻网络中的电位器大小。此外 VLCDADJ 引脚应该接地。
2. 如果您使用电荷泵，可调节 VLCDADJ 引脚上的电阻值。

25.16 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制，即可使用。当前与 LCD 驱动器相关的应用笔记有：

标题	应用笔记 #
Yet Another Clock Using the PIC16C92X	AN649
LCD Fundamentals Using PIC16C92x Microcontrollers	AN658
PICDEM3 Demo Board User's Guide	DS51079

25.17 版本历史

版本 A

这是描述 LCD 模块的初始发行版。

第 26 章 看门狗定时器与休眠模式

目录

本章包括下面一些主要内容：

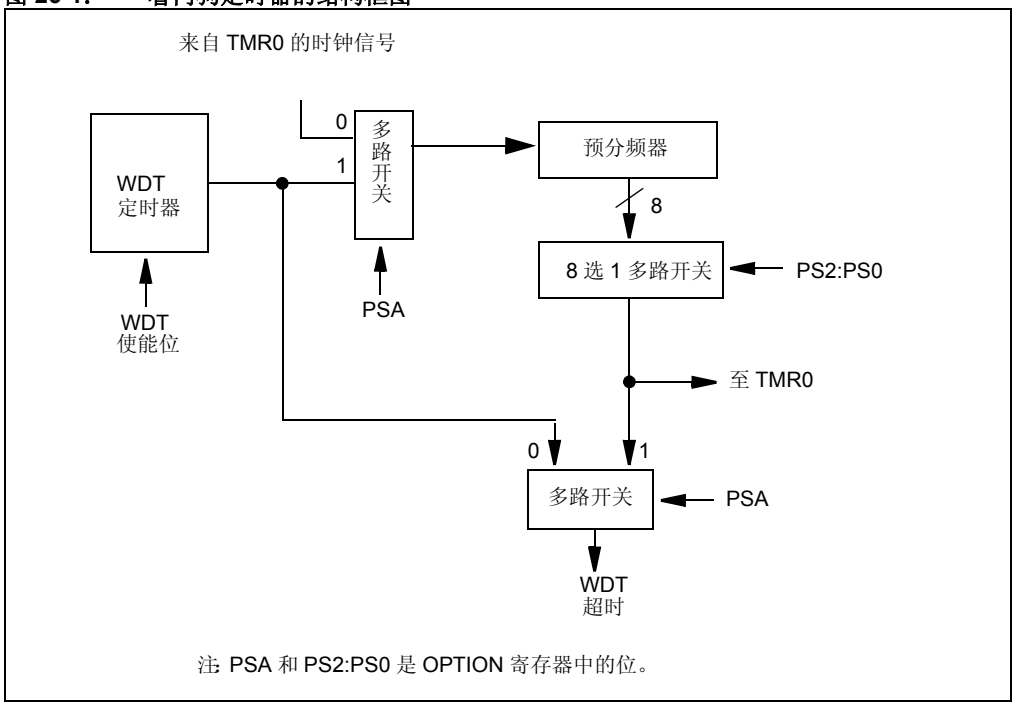
26.1 简介 26-2
26.2 控制寄存器 26-3
26.3 看门狗定时器 （WDT）的操作 26-4
26.4 休眠省电模式 26-7
26.5 初始化 26-9
26.6 设计技巧 26-10
26.7 相关应用笔记 26-11
26.8 版本历史 26-12

26.1 简介

看门狗定时器（WDT）是一个运行在片内的 RC 振荡器，它不需要任何的外接元件。图 26-1 为看门狗定时器的结构框图。该 RC 振荡器独立于 OSC1/CLKIN 引脚上的 RC 振荡器。这样，即使器件的 OSC1 和 OSC2 引脚上的时钟停振（例如执行了 SLEEP 指令），WDT 仍将正常工作。

有一个器件配置位是控制看门狗定时器（WDT）的使能 / 关闭。如果 WDT 被使能，就不能通过软件关闭此功能。

图 26-1： 看门狗定时器的结构框图



注：PSA 和 PS2:PS0 是 OPTION 寄存器中的位。

26.2 控制寄存器

OPTION_REG 寄存器是一个可读写寄存器，它含有各种控制位，用来设置 TMR0 预分频器 /WDT 后分频器、外部 INT 中断、 TMR0 和 PORTB 弱上拉等。

注：	将预分频器分配给看门狗定时器时， TMR0 寄存器的预分频比为 1:1。

寄存器 26-1： OPTION_REG 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
$\overline{\text{RBPU}}$ (1)	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$ (1)**：弱上拉使能位
1 = 禁止弱上拉
0 = 弱上拉由各端口锁存器值使能

bit 6 **INTEDG**：中断触发边沿选择位
1 = INT 引脚的上升沿触发中断
0 = INT 引脚的下降沿触发中断

bit 5 **T0CS**：TMR0 时钟源选择位
1 = T0CKI 引脚上的外部时钟
0 = 内部指令周期时钟 （CLKOUT）

bit 4 **T0SE**：TMR0 计数边沿选择位
1 = T0CKI 引脚上的下降沿递增
0 = T0CKI 引脚上的上升沿递增

bit 3 **PSA**：预分频器分配位
1 = 预分频器分配给 WDT
0 = 预分频器分配给 Timer0 模块

bit 2:0 **PS2:PS0**：TMR0 预分频比 /WDT 后分频比选择位

位值	TMR0 分频比	WDT 分频比
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

图注	
R = 可读位	W = 可写位
U = 未用位，读为 “0”	- n = 上电复位时的值

注 1：	一些器件也称此位为 $\overline{\text{GPPU}}$。有 $\overline{\text{RBPU}}$ 位的器件在 PORTB 上有弱上拉，有 $\overline{\text{GPPU}}$ 位的器件在 GP 端口有弱上拉。

26.3 看门狗定时器 （WDT）的操作

在正常操作期间，一次 WDT 超时溢出将产生一次器件复位。如果器件处于休眠状态，一次 WDT 超时溢出将唤醒器件，使其继续正常操作（即称作 WDT 唤醒）。对 WDTE 设置位清零可以永久性地关闭 WDT。

后分频器分配完全是由软件控制，即它可在程序执行期间随时更改。

注：	为避免发生不可预测的器件复位，当从 Timer0 预分频器的分配改为 WDT 后分频器的分配时，必须执行下列指令序列（如例 26-1 所示）。即使 WDT 被禁止，也要执行这个指令序列。

在例 26-1 中，如果需要的预分频比不是 1:1，就不需要对 OPTION_REG 寄存器做初始修改。如果需要的预分频比是 1:1，那么先向 OPTION_REG 设置一个非 1:1 的临时预分频比，在完成其它操作后，在最后修改 OPTION_REG 时再设置 1:1 的预分频比。这样操作，主要是因为无法知道 TMR0 预分频器的当前计数值，而且分频器更改后，该值将变为 WDT 后分频器的当前计数值，所以必须遵循示例中的代码顺序。如果没有按照示例中的代码顺序改变 OPTION_REG 寄存器，那么无法准确得知 WDT 复位前的时间。

例 26-1：　　预分频器的更改（Timer0→WDT）

```
BSF     STATUS, RP0    ; Bank1
MOVLW   B’xx0x0xxx’    ; Select clock source and postscale value
MOVWF   OPTION_REG     ;   other than 1:1
BCF     STATUS, RP0    ; Bank0
CLRF    TMR0           ; Clear TMR0 & Prescaler
BSF     STATUS, RP0    ; Bank1
MOVLW   B’xxxx1xxx’    ; Select WDT, do not change prescale value
MOVWF   OPTION_REG     ;
CLRWDT                 ; Clears WDT
MOVLW   b'xxxx1xxx'    ; Select new prescale value and WDT
MOVWF   OPTION_REG     ;
BCF     STATUS, RP0    ; Bank0
```

将预分频器从 WDT 改成 Timer0 模块，可以使用例 26-2 所示的指令序列。

例 26-2：　　预分频器的更改（WDT→Timer0）

```
CLRWDT                ; Clear WDT and postscaler
BSF     STATUS, RP0    ; Bank1
MOVLW   b'xxxx0xxx'    ; Select TMR0, new prescale
MOVWF   OPTION_REG     ; value and clock source
BCF     STATUS, RP0    ; Bank0
```

26.3.1 WDT 周期

WDT 的超时溢出周期在不使用后分频器时的典型值为 18ms。这个周期随着温度、VDD 和制造工艺偏差而不尽相同（见DC规范）。如果需要更长的超时溢出周期，可以用软件设置OPTION_REG寄存器，把后分频器分配给 WDT ；这时最大分频比可达 1:128，可以实现 2.3s 左右的超时溢出周期。

CLRWDT 和 SLEEP 指令将对 WDT 和后分频器（如果分配给 WDT）清零，防止其超时而引起器件复位。

看门狗定时器超时溢出使 WDT 复位 （正常工作状态下）或 WDT 唤醒 （休眠状态下），同时状态寄存器中的 $\overline{TO}$ 位将被清零。

26.3.2 WDT 编程注意事项

在最恶劣的情况下 （VDD 最小、温度最高、 WDT 后分频比最大），要过几秒钟 WDT 才会发生超时溢出，因此在编写程序时要考虑到这一点。

注：	当后分频器分配给 WDT 时，在改变后分频值前应务必先执行一条 CLRWDT 指令，否则可能会发生 WDT 复位。

表 26-1： 和看门狗定时器有关的寄存器

寄存器名称	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
配置位	MPEEN	BODEN	CP1	CP0	$\overline{PWRTE}$	WDTE	FOSC1	FOSC0
OPTION_REG	$\overline{RBPU}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0

图注： 阴影部分没有被看门狗定时器使用。

26.4 休眠省电模式

执行一条 SLEEP 指令，器件便进入休眠模式。在这种模式下，电流消耗降到最低。而且，器件的振荡器停振，因此没有系统时钟。

在执行 SLEEP 指令时，如果看门狗定时器是使能的，将把看门狗定时器的当前计数值清零，使其由零重新计数运行，同时状态寄存器 STATUS 的 $\overline{PD}$ 位将被清零， $\overline{TO}$ 位将被置 “1”，振荡器停振。 I/O 端口保持执行 SLEEP 指令之前的状态 （高电平、低电平或高阻状态）。

在休眠模式下，为了使电流消耗降至最低，所有 I/O 引脚应保持为 VDD 或 VSS 电平，不要有拉电流输出，同时应关闭休眠模式下可能消耗电流的功能模块。为了避免悬浮输入引起的开关电流，应拉高或拉低高阻输入的 I/O 引脚。将 T0CKI 的输入设置为 VDD 或 VSS 电平也可降低电流消耗。还要考虑到 PORTB 内部上拉的影响。

$\overline{MCLR}$ 引脚必须处于逻辑高电平 （VIHMC）。

利用器件配置位使能器件的某些功能部件 （如看门狗定时器 （WDT）和欠压复位 （BOR）电路模块）时，休眠模式下可能会多消耗一定量的电流，而在配置位中关闭该部件时也就关闭了该类电流的消耗。

26.4.1 唤醒

下列事件之一可唤醒器件：

1. 器件复位。
2. 看门狗定时器唤醒 （如果 WDT 被使能）。
3. 可以在休眠模式下设置中断标志的外设模块，如：
   - 外部 INT 引脚
   - 端口引脚上的电平变化
   - 比较器
   - A/D
   - Timer1
   - LCD
   - SSP
   - 捕捉

第一种事件会在唤醒器件使器件复位，而后两种事件仅是唤醒器件，然后让程序继续正常运行。状态寄存器 STATUS 中的 $\overline{TO}$ 和 $\overline{PD}$ 位可用来确定器件复位的原因。 $\overline{PD}$ 位在上电时被置 “1”，当执行 SLEEP 指令时被清零。如果发生 WDT 唤醒，则 $\overline{TO}$ 位被清零。

在执行 SLEEP 指令时，下一个指令 （PC + 1）被预先取出。当通过中断事件唤醒器件时，相应的中断使能位必须置 “1”（使能）。唤醒与 GIE 位的状态无关。如果 GIE 位清零 （禁止中断响应），器件将继续执行 SLEEP 后面的指令。如果 GIE 位被置 “1”（使能），则器件在执行 SLEEP 之后的一条指令后，将跳转到中断地址 （0004h）。如果不希望执行 SLEEP 指令之后的指令，应该在 SLEEP 指令之后加一条 NOP 指令。

26.4.2 中断唤醒

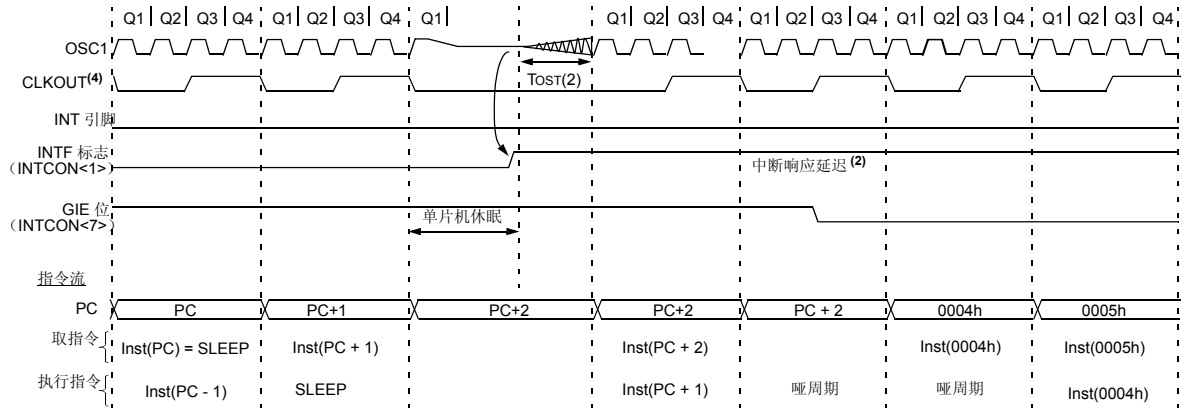
即使全体中断都被禁止（GIE 位清零），对于任何中断源，只要它的中断使能位和中断标志位都置“1”，就会发生下列事件之一：

- 在执行 SLEEP 指令前，发生了中断，SLEEP 指令将作为一个 NOP 指令来执行。因此，WDT 和 WDT 后分频器将不会被清零，$\overline{TO}$ 位不会被置“1”，$\overline{PD}$ 位也不会被清零。
- 在 SLEEP 指令执行时或执行后，发生了中断，器件将立即被唤醒。SLEEP 指令将在唤醒之前完成执行。因此，WDT 和 WDT 后分频器将被清零，$\overline{TO}$ 位将被置“1”，$\overline{PD}$ 位将被清零。

即使执行 SLEEP 指令之前，检验了标志位，它也可能在 SLEEP 指令完成之前被置“1”。要确定是否执行了 SLEEP 指令，可以检测 $\overline{PD}$ 位。如果 $\overline{PD}$ 位为 1，说明 SLEEP 指令被作为一个 NOP 指令来执行而并没有真正执行。

在 SLEEP 指令之前，先执行一条 CLRWDT 指令，可以确保在进入休眠模式时 WDT 已被清零。

图 26-2： 中断唤醒的时序图



注 1：假设工作在 XT、HS 或 LP 振荡器模式。
2：TOST = 1024TOSC（图中未按比例画出）；若为 RC 振荡模式，该延时将不在此时段。
3：假设 GIE = '1'。在这种情况下，单片机被唤醒后，将跳至中断程序。如果 GIE = '0'，将在这一行继续执行。
4：在这些振荡器模式下，CLKOUT 信号无效，在这里仅作为时间基准显示出来。

26.5 初始化

当前没有初始化代码。

26.6 设计技巧

问 1： *器件的工作电压先是下降，然后又回到正常电压范围内。此时器件不再正常工作，而 WDT 并没有复位单片机而使其正常工作。为什么会这样？*

答 1：

WDT 的设计并没有使其在欠压后具备恢复功能。在正常工作范围内，WDT 只能从软件操作错误中恢复。如果是欠压问题，应使能片内欠压电路或运用外部欠压电路。

问 2： *为什么在程序循环体内执行了 CLRWDT 指令，器件仍然会复位？*

答 2：

带 CLRWDT 指令的循环体的执行时间应小于 WDT 的最小超时溢出时间，而不是典型超时溢出时间。

问 3： *器件不能从复位状态中跳出。*

答 3：

在上电期间，必须考虑到振荡器的上电延迟时间 （Tost）。因为上电延迟时间可能有差异，最好将 CLRWDT 指令放在程序循环体的开始处。

26.7 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制，即可使用。目前与看门狗定时器和休眠模式相关的应用笔记有：

标题	应用笔记 #
Power-up Trouble Shooting	AN607

26.8 版本历史

版本 A

这是描述看门狗定时器和休眠模式的初始发行版。

第 27 章 器件配置位

目录

本章包括以下一些主要内容：

27.1 简介 27-2
27.2 配置字位 27-4
27.3 编程校验 / 代码保护 27-8
27.4 识别码 ID 的位置 27-9
27.5 设计技巧 27-10
27.6 相关应用笔记 27-11
27.7 版本历史 27-12

27.1 简介

器件的配置位允许用户根据应用需要配置器件。当器件上电时，这些位的状态决定了器件的工作模式。**27.2“配置字位”小节**对这些配置位及其所能配置的模式进行了讨论。这些配置位的程序存储器映射位置为 2007h。器件正常运行时，无法对该位置进行存取 (只能在编程模式下存取)。

可以通过对配置位编程 (读为 '0') 或不编程 (读为 '1') 来选择不同的器件配置。对配置位编程后，是否能够更改其设置取决于器件的存储工艺和封装形式。

对于只读存储器 (ROM) 器件，这些配置位在 ROM 代码提交时即被确定，且一旦器件掩膜完成，即无法更改 (若要更改，则需新的掩膜代码)。

对一次可编程 (OTP) 器件，一旦这些位被编程 (为 '0')，就无法更改了。

对窗口型 EPROM 器件，一旦这些位被编程 (为 '0') 后，必须通过紫外线擦除，使器件的配置字返回到已擦除状态。紫外线擦除器件时，也擦除了程序存储器的内容。

对闪存器件，这些位可擦除并重新编程。

注： Microchip 建议不要将窗口型器件进行代码保护。

因文档格式的需要，将第 **27.2 节**置于下页。

27.2 配置字位

这些配置位规定了器件的部分工作模式，可通过器件编程器或中档器件的在线串行编程(ICSP™)功能对其编程。器件无法读取这些位的值，而且当用户在器件编程器中选好器件时，配置位的地址就自动确定了。欲获得更多信息，请参见器件的编程规范说明。

注 1:	请始终确保器件编程器上的所选器件就是您编程的器件
注 2:	Microchip 建议将所需配置位的状态嵌入应用的源代码中。通过 MPASM 汇编器中的 CONFIG 指令，可以轻松做到这一点。请参看 **27.2.1 “MPASMTM 的 CONFIG 指令”小节**。

CP1:CP0: 代码保护位

11 = 代码保护关
10 = 参见器件数据手册
01 = 参见器件数据手册
00 = 所有存储器均受代码保护

注 : 有些器件使用较多或较少的位数来配置代码保护。目前就一些只使用一位 (CP0) 的器件，保护位的说明如下：
1 = 代码保护关
0 = 代码保护开

DP: 数据 EEPROM 存储器的代码保护位

1 = 代码保护关
0 = 数据 EEPROM 存储器受代码保护

注 : 对于具有数据 EEPROM 存储器的 ROM 程序存储器器件，使用该位。

BODEN: 欠压复位 (BOR) 使能位

1 = BOR 使能
0 = BOR 禁止

注 : 无论 $\overline{\text{PWRTE}}$ 位的值为何，使能欠压复位即自动使能了上电定时器 (PWRT)。请确保使能欠压复位时，也使能了上电定时器。

$\overline{\text{PWRTE}}$: 上电定时器（PWRT）使能位

1 = PWRT 禁止
0 = PWRT 使能

注 1: 无论 $\overline{\text{PWRTE}}$ 位的值为何，使能欠压复位即自动使能了上电定时器 (PWRT)。请确保使能欠压复位时，也使能了上电定时器。

注 2: 在一些早期的 PICmicro® 单片机中，该位的极性被保留。

注 3:

MCLRE: $\overline{\text{MCLR}}$ 引脚功能选择位

1 = 引脚功能为 $\overline{\text{MCLR}}$

0 = 引脚功能为数字 I/O。$\overline{\text{MCLR}}$ 在内部连接到 VDD 上。

WDTE: 看门狗定时器 (WDT) 使能位

1 = WDT 使能
0 = WDT 禁止

FOSC1:FOSC0: 振荡器选择位

11 = RC 振荡器
10 = HS 振荡器
01 = XT 振荡器
00 = LP 振荡器

FOSC2:FOSC0: 振荡器选择位

111 = EXTRC 振荡器，带 CLKOUT 时钟输出
110 = EXTRC 振荡器
101 = INTRC 振荡器，带 CLKOUT 时钟输出
100 = INTRC 振荡器
011 = 保留
010 = HS 振荡器
001 = XT 振荡器
000 = LP 振荡器

未用 : 读为 '1'

图注		
R = 可读位	P = 可编程位	U = 未用位 , 读为 '0'
- n = POR 复位后的值		u = 编程后状态不变

注 : 配置位的位置视器件的不同而不同。欲了解配置位的位置，请参看器件编程规范说明。使用 Microchip 的器件编程器时，不要求您了解配置位的具体位置。

27.2.1 MPASM™ 的 CONFIG 指令

Microchip 的汇编器 MPASM™ 中有一个很实用的功能，即允许您在源代码文件中规定本程序中配置位所选择的状态。这样可保证在对器件进行编程时，所需的配置位也同时被编程。这最大限度地减小了将错误的配置烧录进器件的风险，使您无需为器件为何不能正常工作而苦恼。

例 27-1 给出了一个使用 CONFIG 指令的模板程序。

例 27-1: 使用 CONFIG 指令的模板程序

```
     LIST   p = p16C77      ; List Directive,
;    Revision History
;
     #INCLUDE    <P16C77.INC>    ; Microchip Device Header File
;
     #INCLUDE    <MY_STD.MAC>    ; File which includes my standard macros
     #INCLUDE    <APP.MAC>       ; File which includes macros specific
                                 ;   to this application
;
; Specify Device Configuration Bits
;
     __CONFIG    _XT_OSC & _PWRTE_ON & _BODEN_OFF & _CP_OFF & _WDT_ON
;
     org   0x00            ; Start of Program Memory
RESET_ADDR   :             ; First instruction to execute after a reset


     end
```

目前在 Microchip 头文件中直接使用 CONFIG 指令的符号显示在表 27-1 中。欲了解您所使用的器件具备哪些可用符号，请参见 Microchip 所提供的该器件的头文件。

注： 只要 (在 LIST 和 INCLUDE 文件指令中) 正确指定了器件，即可保证所有位的极性。

表 27-1: __CONFIG 指令符 (来自 Microchip 的头文件)

特性	符号
振荡器	_RC_OSC
	_EXTRC_OSC
	_EXTRC_OSC_CLKOUT
	_EXTRC_OSC_NOCLKOUT
	_INTRC_OSC
	_INTRC_OSC_CLKOUT
	_INTRC_OSC_NOCLKOUT
	_LP_OSC
	_XT_OSC
	_HS_OSC
看门狗定时器	_WDT_ON
	_WDT_OFF
上电定时器	_PWRTE_ON
	_PWRTE_OFF
欠压复位	_BODEN_ON
	_BODEN_OFF
总清零使能	_MCLRE_ON
	_MCLRE_OFF
代码保护	_CP_ALL
	_CP_ON
	_CP_75
	_CP_50
	_CP_OFF
代码保护数据 EEPROM	_DP_ON
	_DP_OFF
代码保护校准空间	_CPC_ON
	_CPC_OFF

注 1: 任一器件不一定具备所有上述配置位符号。欲了解某一器件具备哪些符号，请参见 Microchip 所提供的该器件的头文件。

27.3 编程校验 / 代码保护

若代码保护位未被烧录，则可读出片上程序存储器的内容以便校验。

注： Microchip 建议不要将窗口型器件进行代码保护。

27.3.1 ROM 器件

当一个 ROM 器件也具备数据 EEPROM 存储器时，则可额外实现一个代码保护配置。程序存储器的配置位是作为部分 ROM 代码提交的。数据 EEPROM 存储器的代码保护配置位将成为一个 EEPROM 位。在 ROM 器件测试完毕后，EEPROM 数据存储器的代码保护位将被烧录成与程序存储器代码保护位相同的状态，即当程序存储器代码保护被关闭时，数据 EEPROM 代码保护也被关闭，所有其它选择均将打开数据 EEPROM 的代码保护功能。

有些应用中，如果器件已受代码保护，而在应用交付前尚需对 EEPROM 进行编程，则需先将整个数据 EEPROM 擦除。器件的编程规范中详细说明了擦除步骤。Microchip 的器件编程器贯彻了指定的步骤。完成上述步骤后， EEPROM 存储器的代码保护即被关闭。这样即可将所需数据烧录到 EEPROM 中。对数据 EEPROM 阵列的编程完成后，应按需要将数据 EEPROM 的代码保护配置位编程。

27.4 识别码 ID 的位置

识别码 ID 指定了四个位置 (2000h - 2003h)，供用户存储校验和或其它识别代码。器件正常运行时，这些位置是不可存取的，但在编程 / 校验时可以被读写。建议只使用识别码 ID 位置的最低 4 位。

27.5 设计技巧

问 1: *我有一个 JW 型器件但无法再对它编程（读出的数据是乱码或者全是 '0'）。这是什么问题？*

答 1:

没有问题。可能是您对器件进行了代码保护。如果是这样，这个器件就不能再用了。详细内容请参见 **27.3“编程校验 / 代码保护”小节**。

问 2: *从 PIC16C74 转换到 PIC16C74A 后，我的应用就无法正常工作了。*

答 2:

1. 您是否在INCLUDE文件和LIST指令中指定了PIC16C74A并重新编译了源程序？我们强烈建议使用 CONFIG 指令。
2. 在器件编程器中您是否指定了 PIC16C74A？配置位是否全都符合要求？

问 3: *对器件进行擦除时，程序存储器已擦空，但配置字却没有被擦除。*

答 3:

这是设计原因。另请记住 Microchip 建议不要将窗口型器件代码保护。

27.6 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，经过适当修改并受一定限制即可使用。目前与配置字相关的应用笔记有：

标题	应用笔记 #
目前没有相关的应用笔记	

27.7 版本历史

版本 A

这是描述配置字的初始发行版。

第 28 章 在线串行编程

目录

本章包括下面一些主要内容：

28.1 简介 28-2
28.2 进入在线串行编程模式 28-3
28.3 应用电路 28-4
28.4 编程器 28-6
28.5 编程环境 28-6
28.6 其它优点 28-7
28.7 PICmicro® OTP 型单片机的现场编程 28-8
28.8 FLASH 型 PICmicro® 单片机的现场编程 28-10
28.9 设计技巧 28-12
28.10 相关应用笔记 28-13
28.11 版本历史 28-14

28.1 简介

所有中档系列单片机都可以在最终应用电路中在线串行编程（In-Circuit Serial Programming™，ICSP™）。ICSP 只需使用五根线，其中时钟和数据线各 1 根，另外三根线作为电源、地和编程电压线。

在线串行编程有利于降低库存开销和加快产品上市。在产品中嵌入一片空白的 Microchip单片机，作为一个备用的设计。当接到订单后，便可以在很短的时间内用最新版本的固件对这些产品进行编程、测试，然后交货。这种方法也减少了旧固件版本引起的报废库存。这种生产方法有利于快速根据用户要求定制产品。

大部分人认为只能在装配线上通过 ICSP 对 PICmicro® OTP 单片机进行一次编程，这是不正确的。实际上，可以采用一定的方法对 OTP 器件进行多次编程，这与固件大小有关。下文将对这个方法进行介绍，该方法提供了一种类似于 EEPROM 或闪存存储器的现场升级固件的手段。

28.2 进入在线串行编程模式

通过保持 RB6 和 RB7 引脚为低电平，V_{DD} 为编程电压，并将 $\overline{MCLR}$ (V_{PP}) 引脚电压从 V_{IL} 增加到 V_{IHH}(见编程规范)，器件便进入编程 / 校验模式。此时， RB6 为编程时钟线，RB7 为编程数据线。在该模式下，RB6 和 RB7 都是施密特触发器输入，当 RB7 驱动数据时，它是 CMOS 输出驱动。

复位后，为使器件进入编程 / 校验模式，程序计数器 (PC) 指向 00h 地址。然后可向器件发送一个 6 位的命令，根据这一命令是装入还是读出， 14 位编程数据将被提供给器件或是从器件中读出。有关串行编程的完整细节，请参考器件的编程规范。

在线串行编程模式下，看门狗定时器电路不能产生器件复位。

28.3 应用电路

应用电路必须设计为使所有编程信号直接连接到PICmicro单片机的相关引脚上。图 28-1 给出了一个典型的 ICSP 连接电路。应用电路必须解决以下问题：

- MCLR/VPP 引脚与电路其它部分相隔离
- RB6 和 RB7 的负载
- VDD、MCLR/VPP、 RB6 和 RB7 引脚的电容问题
- VDD 的最小和最大工作电压
- PICmicro 单片机的振荡器
- 与编程器的接口

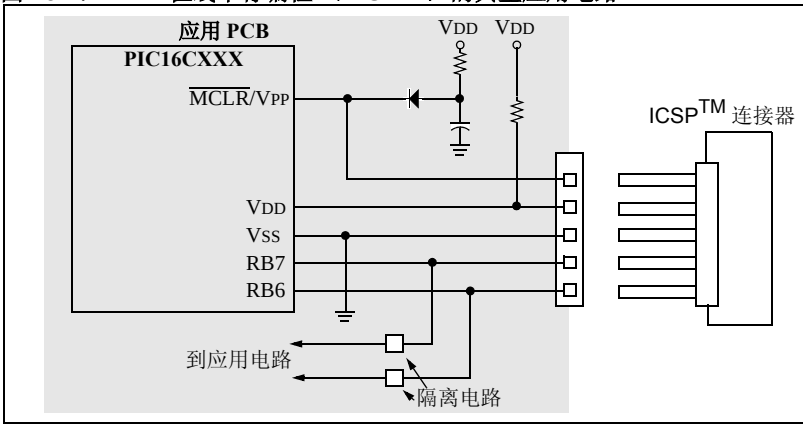
MCLR/VPP 引脚通常与 RC 电路相连，上拉电阻接 VDD，电容接地。VPP 电压必须与电路的其它部分隔离 (大多数情况下电阻不起隔离电路的作用)，根据电容器的大小，RC 电路可能影响 ICSP 的操作。因此当 RC 电路与 MCLR/VPP 相连时，推荐采用图 28-1 所示的电路，其中使用了肖特基型二极管。MCLR/VPP 的另一个问题是当对 PICmicro 单片机编程时，该引脚将被同时驱动至大约 13V 和地，因此应用电路必须与编程器提供的编程电压隔离。

RB6 和 RB7 引脚用于 PICmicro 单片机的串行编程。RB6 是时钟线，RB7 是数据线。RB6 由编程器驱动， RB7 是双向引脚，编程时由编程器驱动，校验时由 PICmicro 单片机驱动。这两个引脚必须与应用电路的其它部分隔离，从而在编程时不会对信号产生影响。将 RB6 和 RB7 与电路其它部分隔离时必须考虑编程器的输出阻抗。隔离电路必须使 RB6 能够作为 PICmicro 单片机的输入，而 RB7 能够作为双向引脚 (PICmicro 单片机和编程器都能驱动它)。例如， PRO MATE® II 的输出阻抗为 1kΩ。如果设计允许，应用电路不要使用这些引脚。对于大多数应用，如果需要使用这些引脚，设计人员可以考虑信号是否需要缓冲。设计人员必须考虑 RB6 和 RB7 连接哪种类型的电路，然后决定如何隔离这些引脚。图 28-1 没有给出应用电路中隔离 RB6 和 RB7 的电路，因为隔离电路的设计与具体的应用有很大关系。

为简化接口设计，下面列出在应用中使用这些 I/O 引脚的最佳方法 （按最佳顺序排列）：

1. 将 RB6/RB7 专用于 ICSP™。
2. 这些端口作为输出时，具有极轻的负载。
3. 采用隔离电路，使信号满足 ICSP 规范。

图 28-1： 在线串行编程 （ICSP™）的典型应用电路



编程引脚的总电容将影响编程器输出信号的上升速率。典型电路中，一般在 VDD 和地之间接有几百微法的滤波电容以抑制噪声和电源电压波动。但是这种电容需要编程器必须具有相当强的驱动能力，才能满足 VDD 上升速率的要求。大多数编程器只能对 PICmicro 单片机进行编程，而不能驱动整个应用电路。一种解决方案是在编程器和应用电路之间加一块驱动电路板。该驱动电路板有独立的电源，可以满足 VPP 和 VDD 引脚电压上升速率的要求，并可为整个应用电路供电。RB6 和 RB7是否需要缓冲取决于具体的应用。作为示例，图 28-2 给出了一个驱动电路板原理图。

注：	必须在用户的应用电路中对该驱动电路板进行测试，以确定编程信号时序对应用电路的影响。如果应用电路的 VDD、VPP、RB6 或 RB7 引脚负载过重，电路可能需要做出一些更改。

Microchip 编程规范规定器件应在 5V 电压下编程。如果应用电路只能在 3V 电压下工作，那么需要一些特殊的措施。例如在编程时将 PICmicro 单片机与其它应用电路完全隔离。另一个问题是，必须在应用电路的最小和最大工作电压下对器件进行校验。例如，在一个使用三个 1.5V 电池供电的系统中，其工作电压范围是 2.7V 到 4.5V。而编程器必须在 5V 电压下对器件进行编程，并且必须在 2.7V 和 4.5V 电压下对程序存储器进行校验，以确保编程正确。这样可以保证 PICmicro 单片机在整个工作电压范围内都能正常工作。

最后一个问题是关于应用电路中的振荡器。在单片机执行任何代码前，$\overline{\text{MCLR}}$/VPP 的电压必须升高到特定值，以便进入编程模式。 PICmicro 单片机的晶体模式不用考虑这一问题，因为在代码执行前振荡器上电延迟定时器要等候 1024 个振荡周期。 RC 振荡器不需要上电延迟时间，因此不使用上电延迟定时器。编程器必须在 RC 振荡器振荡 4 次之前，令 $\overline{\text{MCLR}}$/VPP 达到进入编程模式所需的电压。如果 RC 振荡器振荡了 4 次或 4 次以上，程序计数器将会增加到一个不确定的值 X。如果这时器件进入编程模式，程序计数器不为零，编程器将从偏移量 X 开始烧写代码。有一些方法可以弥补 $\overline{\text{MCLR}}$/VPP 的低上升速率问题。第一种方法是先不接 RC 振荡器的电阻，对器件编程后再接入 R 电阻。另一种方法是在编程时用编程接口将 PICmicro 的 OSC1 引脚短接到地，这样在编程期间便不会产生振荡。

剩下的就是如何将应用电路与编程器相连。这在很大程度上取决于编程环境，将在相应章节中加以讨论。

28.4 编程器

另一个需要考虑的是编程器。PIC16CXXX 系列单片机只支持串行编程，因此所有支持该系列器件的编程器都支持 ICSP。编程器的一个问题是它的驱动能力。前面已经讨论过，编程器必须提供 ICSP 信号所需的上升速率和驱动应用电路所需的电流。图 28-2 给出了一个驱动板的例子，该原理图中没有包含 RB6 和 RB7 的缓冲电路。建议您针对具体应用，考虑是否需要缓冲。编程器的另一个问题是：应该在哪些 VDD 电压值下校验 PICmicro 单片机程序存储器中的内容。例如，PRO MATE® II 可在指定器件的最小和最大 VDD 工作电压下校验程序存储器内容，因而被认为是品质优良的生产编程器。而 PICSTART® Plus 只能在 5V 电压下进行校验，因此只能用于样机编程。 Microchip 编程规范要求：必须在应用电路的最小和最大 VDD 工作电压下对程序存储器的内容进行校验，这样可确保单片机能适应 VDD 电压的变化。

另外还可以使用几家第三方的编程器。使用时必须根据这些编程器的特点及其是否适用于编程环境来加以选择。在 *Microchip Development Systems Ordering Guide* (DS30177) 里，提供了关于 Microchip 所有开发工具的详细信息。*Microchip Third Party Guide* (DS00104) 提供了所有第三方工具开发工具提供商的信息。选择编程器时，可以查阅这两份参考资料。包括并行或串行 PC 主机连接、独立编程以及单个或量产编程器等许多选择。第三方开发工具提供商包括 Advanced Transdata Corporation、BP Microsystems、Data I/O、Emulation Technology 和 Logical Devices 等。

28.5 编程环境

编程环境决定了选择哪种类型的编程器、编程器电缆长度及其应用电路接口。有些编程器适用于手工装配线，而另一些编程器可能适用于自动装配线。有时可能需要量产编程器，实现对多个系统同时编程。

编程器和应用电路之间的物理距离会影响每个编程信号的负载电容，这会直接影响到对信号上升速率和电流的驱动能力。编程电缆应尽可能地短，并且要正确地进行端接和屏蔽，否则信号反射和噪声会破坏编程信号。

最后，编程器的应用电路接口取决于装配线和应用电路本身的尺寸。用一个简单的插座就可以将应用电路和编程器连接起来，这种方法适用于手工装配线，技术员可通过插座将编程器电缆插在电路板上。另一种方法是使用弹簧测试针 （通常称为 pogo pin）。应用电路的每个编程信号都在板上有相应的焊盘测试点。此外还需有一个带有弹簧针的夹具，夹具的探针与应用电路板上焊盘测试点的位置互相吻合。将夹具和应用电路板相互对齐时，弹簧针即与焊盘点相连。这种方法适合于自动装配线。

在考虑了应用电路、编程器和编程环境的各种问题之后，就可以建立一条基于 ICSP 的高质量和可靠生产线了。

28.6 其它优点

ICSP 还具有其它一些优点，例如标定和产品序列化。如果程序存储器容量足够，可以将标定数据放在程序存储器中而不是外部串行 EEPROM 中，这可以降低成本和提高可靠性。例如，如果应用系统使用了热敏电阻，对于不同的系统来说，热敏电阻的值不同。可以将标定数据以表格的形式存储在程序存储器中，以便单片机对外部元件的容差作补偿（用软件实现）。采用软件标定技术，可以在不影响系统性能的情况下降低系统的成本。但是这与 ICSP 有什么关系呢？假设完成标定功能的固件已经烧写到 PICmicro 单片机中。标定数据通过标定夹具传送。当传送完所有的标定数据时，夹具将单片机置于编程模式，并将标定数据编程到 PICmicro 单片机中。应用笔记 *AN656，In-Circuit Serial Programming™ of Calibration Parameters Using a PICmicro® Microcontroller* 介绍了如何实现这种标定数据编程。

ICSP 的另一个优点是可以实现产品序列化。每个应用产品都可以烧写进一个唯一的或随机的序列号。具有唯一序列号的这种应用适用于安全系统。一般的系统使用 DIP 开关来设定序列号。而现在通过 ICSP 可以将序列号烧写到程序存储器中，这样可以降低整个系统的成本并且减小篡改序列号的可能性。

28.7 PICmicro® OTP 型单片机的现场编程

OTP型器件通常不能再编程，但是PICmicro 单片机架构具有这样的灵活性，只要单片机的程序存储器容量至少是固件的两倍，而且单片机没有代码保护。如果目标单片机没有足够的程序存储器空间，可以换用程序存储器容量更大的单片机， Microchip 提供了具有相同外设功能而程序存储器容量不同（从 0.5K 到 8K）的一系列单片机。

PIC16CXXX 单片机有两个矢量，即复位矢量和中断矢量，其地址分别为 0x0000 和 0x0004。当 PICmicro 单片机复位时，将从 0x0000 开始执行代码；而中断响应时，将从 0x0004 开始执行代码。例 28-2 的第一列是第一次写入 PICmicro 单片机的代码。例 28-2 的第二列是第二次写入 PICmicro 单片机的代码。

例 28-2 表明，如果对 PICmicro 单片机进行第二次编程，存储单元 0x0000 的内容由 `goto Main` (0x2808) 变成了全 0，这正好是一个空操作。该存储单元不能被再次写入新操作码 (0x2860)，因为对 OTP 型器件编程时，只能将 1 变为 0，而不能将 0 变成 1。下一个存储单元 0x0001 由原来的空白 (即全 1) 变成 `goto Main` (0x2860)。当发生复位时，PICmicro 单片机执行地址 0x0000 中的空操作，然后执行 `goto Main` 指令跳转到主程序。在 该例中， 0x005A 之后的所有存储单元在初始编程时都是空白的，因而对 PICmicro 单片机 进行第二次编程时，可以将代码写入这些单元。对中断向量 （0x0004）的第二次编程与上述类似。

对于程序存储器超过 2K 字的 PICmicro 单片机，这种方法要略做改动。由于对超过 2K 字时，存在程序存储器分页的问题。 `goto Main` 和 `goto ISR` 指令要按照例 28-1 所示作改动。

例 28-1： 程序存储器分页

```
   movlw <page>            movlw <page>
   movwf PCLATH            movwf PCLATH
   goto  Main              goto  ISR
```

这样一次可编程（OTP）型 PICmicro 单片机便具有了类似 EEPROM 或闪存程序存储器的特性。

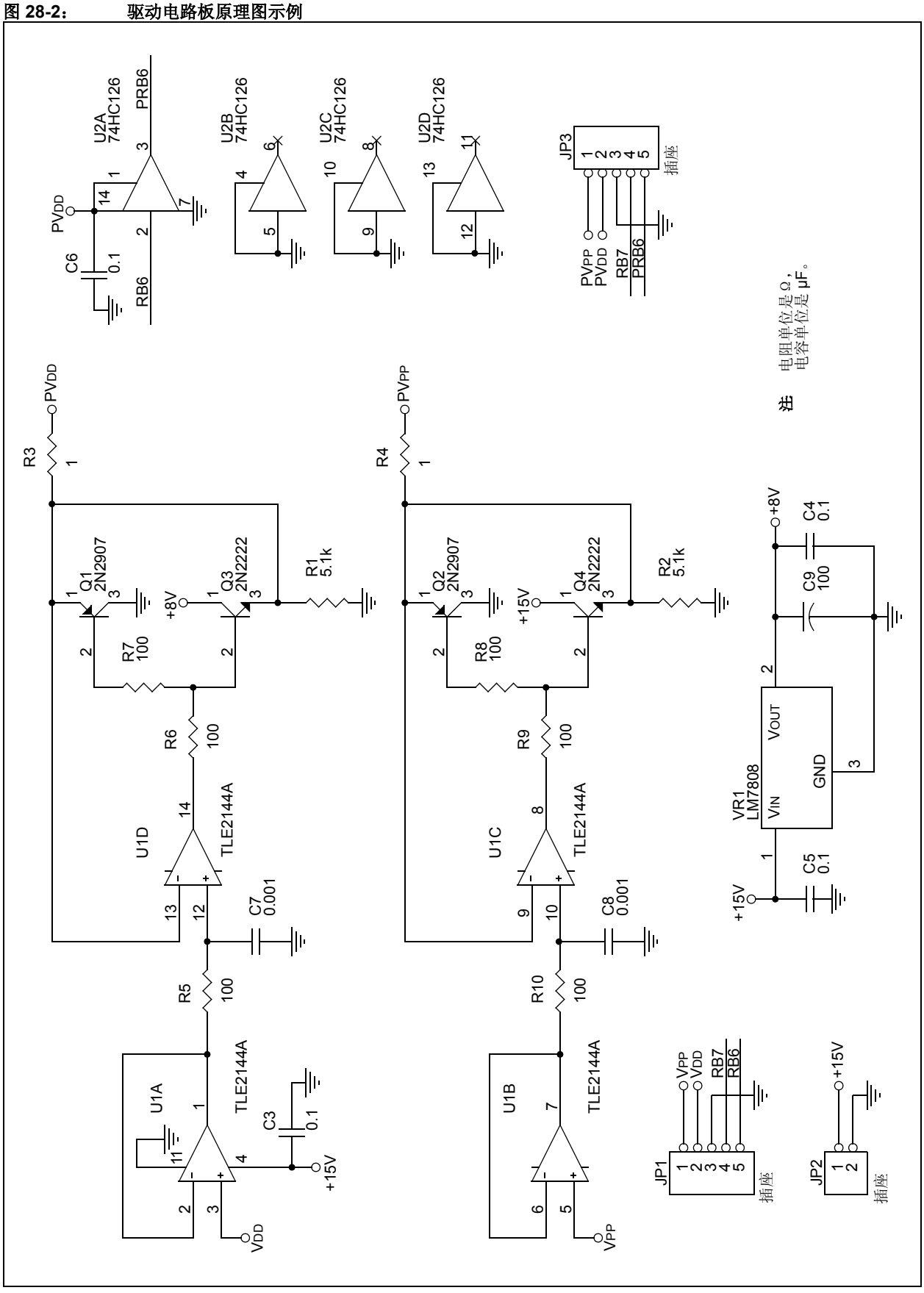
例 28-2：　　两次编程的列表文件

```
    First Program CycleSecond Program Cycle
_____________________________________________________________________________________________
ProgOpcodeAssembly            |ProgOpcodeAssembly
Mem     Instruction            |MemInstruction
---------------------------------------------------------------------------------------------
00002808goto  Main;Main loop  |00000000nop
00013FFF<blank>; at 0x0008 |00012860goto  Main; Main now
00023FFF<blank>            |00023FFF<blank>;   at 0x0060
00033FFF<blank>            |00033FFF<blank>
00042848goto  ISR; ISR at    |00040000nop
00053FFF<blank>;   0x0048  |000528A8goto  ISR; ISR now at
00063FFF<blank>            |00063FFF<blank>;   0x00A8
00073FFF<blank>            |00073FFF<blank>
00081683bsf   STATUS,RP0           |00081683bsf   STATUS,RP0
00093007movlw 0x07            |00093007movlw 0x07
000A009Fmovwf ADCON1            |000A009Fmovwf ADCON1
 .                   | .
 .                   | .
 .                   | .
00481C0Cbtfss PIR1,RBIF            |00481C0Cbtfss PIR1,RBIF
0049284Egoto  EndISR            |0049284Egoto  EndISR
004A1806btfsc PORTB,0            |004A1806btfsc PORTB,0
 .                   | .
 .                   | .
 .                   | .
00603FFF<blank>            |00601683bsf   STATUS,RP0
00613FFF<blank>            |00613005movlw 0x05
00623FFF<blank>            |0062009Fmovwf ADCON1
 .                   | .
 .                   | .
 .                   | .
00A83FFF<blank>            |00A81C0Cbtfss PIR1,RBIF
00A93FFF<blank>            |00A928AEgoto  EndISR
00AA3FFF<blank>            |00AA1806btfsc PORTB,0
 .                   | .
 .                   | .
 .                   | .
---------------------------------------------------------------------------------------------
```

28.8 FLASH 型 PICmicro® 单片机的现场编程

因为 FLASH 型 PICmicro 单片机内置了 ICSP 接口电路，所以对其很容易实现现场再编程。即使有代码保护，也可以对 FLASH 单片机进行再编程。由便携式电脑和编程器就可构成一个 ICSP 编程平台。技术人员将 ICSP 接口电缆插到应用电路上，通过运行编程软件就可将新固件下载到 FLASH 型 PICmicro 单片机中。应用系统的另一个问题是如何升级和保证无故障（bug）运行。当希望给应用系统增加新功能时，通过 ICSP，可以将系统现有的程序现场升级到最新固件版本。

图 28-2：　　驱动电路板原理图示例



28.9 设计技巧

问 1： *在线串行编程时，为什么整个程序在程序存储器中的位置发生了偏移？*

答 1：

如果 $\overline{\text{MCLR}}$ 引脚的电压上升速率不够快，器件电压在正常工作电压范围内，则程序计数器 (PC) 的值就会增加（即 PC 值不再是零）。此时 PC 指向的具体位置，由在进入编程模式前已运行的时钟数决定。

问 2： *通过自己设计的插座将 PRO MATE II 编程信号与应用电路板相连时，为什么有时进行 ICSP™ 编程时，会出现编程错误？*

答 2：

编程电压或时序可能不对，这可能是由于：

- 应用板电路
- 从编程器到目标电路板的电缆长度
- V_{DD} 上的大电容影响了电压或时序

28.10 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与在线串行编程相关的应用笔记有：

标题	应用笔记 #
In-Circuit Serial Programming™ of Calibration Parameters using a PICmicro®	AN656
In-Circuit Serial Programming™ Guide	DS30277

28.11 版本历史

版本 A

这是描述在线串行编程的初始发行版。

第 29 章 指令集

目录

本章包括下面一些主要内容：

29.1 简介 29-2
29.2 指令格式 29-4
29.3 作为源 / 目标寄存器的特殊功能寄存器 29-6
29.4 Q 周期操作 29-7
29.5 指令描述 29-8
29.6 设计技巧 29-45
29.7 相关应用笔记 29-47
29.8 版本历史 29-48

29.1 简介

中档系列单片机的每个指令都是 14 位字，由指明指令类型的操作码和进一步说明指令具体操作的一个或多个操作数组成。表 29-1 是对中档单片机指令集的概括，表中列出了 MPASM 汇编器识别的指令。整个指令系统具有高度正交性，分为三种基本类型：

- **字节**操作类指令
- **位**操作类指令
- **立即数与控制**操作类指令

表 29-2 给出了对操作码各个字段的描述。

对于**字节**操作类指令，'f' 代表文件寄存器标识符，'d' 代表目标标识符。文件寄存器标识符指定了指令将会使用哪一个文件寄存器。

目标标识符指定了操作结果的存放位置。如果 'd' 为 0，操作结果存入 W 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。

对于**位**操作类指令，'b' 代表位域标识符，选择操作影响到的位，而 'f' 代表位在哪个文件寄存器中。

对于**立即数和控制**操作类指令，'k' 代表一个 8 位或 11 位的立即数或常数。

除条件测试为真的程序跳转或指令结果改变了 PC 值的指令以外，其它所有的指令都是单周期指令。对于前面的指令，执行指令需要两个指令周期，第二个周期中执行的是一条 NOP 指令。每个指令周期包括 4 个振荡周期，因此，如果振荡器频率为 4 MHz，通常指令的执行时间为 1 µs。对于条件测试为真的程序跳转或指令结果改变了 PC 值的指令，则其执行时间为 2 µs。

表 29-1： 中档单片机指令集

助记符、操作数		描述	周期	14 位指令字 MSb			LSb	受影响的状态	注
针对字节的文件寄存器操作指令									
ADDWF	f, d	将 W 和 f 寄存器内容相加	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	f, d	W 和 f “与”运算	1	00	0101	dfff	ffff	Z	1,2
CLRF	f	f 清零	1	00	0001	lfff	ffff	Z	2
CLRW	-	W 清零	1	00	0001	0xxx	xxxx	Z	
COMF	f, d	f 取反	1	00	1001	dfff	ffff	Z	1,2
DECF	f, d	f 减 1	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	f, d	f 减 1，为 0 则跳过	1(2)	00	1011	dfff	ffff		1,2,3
INCF	f, d	f 加 1	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	f, d	f 加 1，为 0 则跳过	1(2)	00	1111	dfff	ffff		1,2,3
IORWF	f, d	f 的内容和 W 的内容相或	1	00	0100	dfff	ffff	Z	1,2
MOVF	f, d	f 内容送入 f 或 W	1	00	1000	dfff	ffff	Z	1,2
MOVWF	f	将 W 送至 f	1	00	0000	lfff	ffff		
NOP	-	空操作	1	00	0000	0xx0	0000		
RLF	f, d	f 寄存器带进位位循环左移	1	00	1101	dfff	ffff	C	1,2
RRF	f, d	f 寄存器带进位位循环右移	1	00	1100	dfff	ffff	C	1,2
SUBWF	f, d	f 减 W	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	f, d	f 半字节交换	1	00	1110	dfff	ffff		1,2
XORWF	f, d	W 与 f 异或运算	1	00	0110	dfff	ffff	Z	1,2
针对位的文件寄存器操作指令									
BCF	f, b	f 的 bit b 清零	1	01	00bb	bfff	ffff		1,2
BSF	f, b	f 的 bit b 置 1	1	01	01bb	bfff	ffff		1,2
BTFSC	f, b	检测 f 的 bit b，为 0 则跳过	1 (2)	01	10bb	bfff	ffff		3
BTFSS	f, b	检测 f 的 bit b，为 1 则跳过	1 (2)	01	11bb	bfff	ffff		3
立即数和控制操作指令									
ADDLW	k	W 加立即数	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	k	W 与立即数相与	1	11	1001	kkkk	kkkk	Z	
CALL	k	调用子程序	2	10	0kkk	kkkk	kkkk		
CLRWDT	-	看门狗定时器清零	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	跳转	2	10	1kkk	kkkk	kkkk		
IORLW	k	立即数或 W	1	11	1000	kkkk	kkkk	Z	
MOVLW	k	立即数送 W	1	11	00xx	kkkk	kkkk		
RETFIE	-	中断返回	2	00	0000	0000	1001		
RETLW	k	立即数送W，子程序返回	2	11	01xx	kkkk	kkkk		
RETURN	-	子程序返回	2	00	0000	0000	1000		
SLEEP	-	进入休眠模式	1	00	0000	0110	0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	立即数减 W	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	k	立即数与 W 异或	1	11	1010	kkkk	kkkk	Z	

注 1：当 I/O 口寄存器用自身内容修改自身时 (例如：MOVF PORTB,1)，使用的值将是出现在引脚上的值，而非锁存器中的值。例如，如果一引脚配置为输入，其数据锁存器中的值为 ‘1’，但此时外部器件将该引脚拉为低电平，则被写回数据锁存器的数据值将是 ‘0’。

2： 如果预分频器被分配给 Timer0 模块，那么对 TMR0 寄存器执行这条指令 （适当时， d = 1）将清零预分频器。

3：如果程序计数器 (PC) 被修改或者执行一个条件检测为真的跳转，则指令需要两个指令周期。第二个指令周期执行一条空操作 NOP 指令。

29.2 指令格式

图 29-1 给出了指令的三种通用格式。从指令的通用格式可以看出，指令字的操作码部分可以有 3 位到 6 位的信息变化。这就是中档系列单片机指令可以有 35 条之多的原因。

注 1：	任何未使用的操作码都是保留的，使用任何保留的操作码将会导致异常操作。
注 2：	为保持对将来中档系列产品的向上兼容性，不要使用 OPTION 和 TRIS 指令。

所有的指令例子均使用下面的格式来表示十六进制数：

0xhh

其中 h 代表一个 16 进制位。

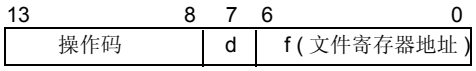
表示二进制数：

00000100b

b 为二进制数的标志。

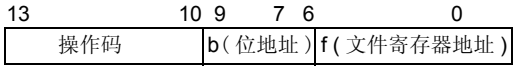
图 29-1： 指令的通用格式

针对字节的文件寄存器操作指令



d = 0，目标寄存器为 W
d = 1，目标寄存器为 f
f = 7 位文件寄存器地址

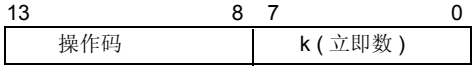
针对位的文件寄存器操作指令



b = 3 位位地址
f = 7 位文件寄存器地址

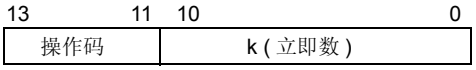
立即数和控制操作指令

一般指令



k = 8 位立即数

仅 CALL 和 GOTO 指令



k = 11 位立即数

表 29-2： 指令描述约定

字段	描述
f	文件寄存器地址 (0x00 到 0x7F)
W	工作寄存器 (累加器)
b	某 8 位文件寄存器内的位地址 (0 到 7)
k	立即数、常数或标号 （可以是 8 位或 11 位值）
x	与取值无关的位 (0 或 1) 汇编器将产生 x = 0 的代码，为了与所有的 Microchip 软件工具兼容，建议使用这种格式。
d	目标寄存器选择： d = 0：结果保存至 W d = 1：结果保存至文件寄存器 f
dest	目标寄存器， W 寄存器或指定的文件寄存器地址
label	标号名
TOS	栈顶
PC	程序计数器
PCLATH	程序计数器高字节锁存器
GIE	全局中断允许位
WDT	看门狗定时器
$\overline{TO}$	超时标志位
$\overline{PD}$	掉电标志位
[ ]	可选的
( )	内容
→	赋值给
< >	寄存器位域
∈	表示属于某个集合
italics	用户定义项（字体为 courier）

29.3 作为源 / 目标寄存器的特殊功能寄存器

在本章中介绍的指令系统高度正交，可以实现对所有寄存器，包括特殊功能寄存器的读写。一些需要注意的特殊情况介绍如下：

29.3.1 STATUS 状态寄存器作为目标寄存器

如果一条指令写 STATUS 状态寄存器，Z、C、DC 和 OV 位会被指令运行结果置 1 或清零而覆盖掉写入的原始数据位。例如，执行 CLRF STATUS 会将 STATUS 寄存器清零，并将 Z 位置 1，寄存器的内容变为 0000 0100b。

29.3.2 PCL 寄存器作为源寄存器或目标寄存器

对 PCL 进行读、写或读－修改－写可能会导致以下结果：

读 PC：	PCL →dest ； PCLATH 不变；
写 PCL：	PCLATH → PCH ； 8 位目标值 → PCL
读－修改－写：	PCL→ ALU 操作数 PCLATH → PCH ； 8 位结果 → PCL

其中 PCH 为程序计数器的高字节 (不可寻址寄存器)，PCLATH 为程序计数器高字节锁存器，dest 为目标寄存器 （W 寄存器或文件寄存器 f）。

29.3.3 位操作

所有位操作指令将首先读整个寄存器的内容，然后在选择的位上进行操作，最后将结果写回 (读－修改－写 (R-M-W)) 指定寄存器。当对一些特殊功能寄存器操作时 （如端口寄存器），应特别注意这一点。

注：	在 Q1 周期中，器件操作的状态位 (包括中断标志位) 被置 1 或清零，所以对包含这些位的寄存器执行 R-M-W 指令无效。

29.4 Q 周期操作

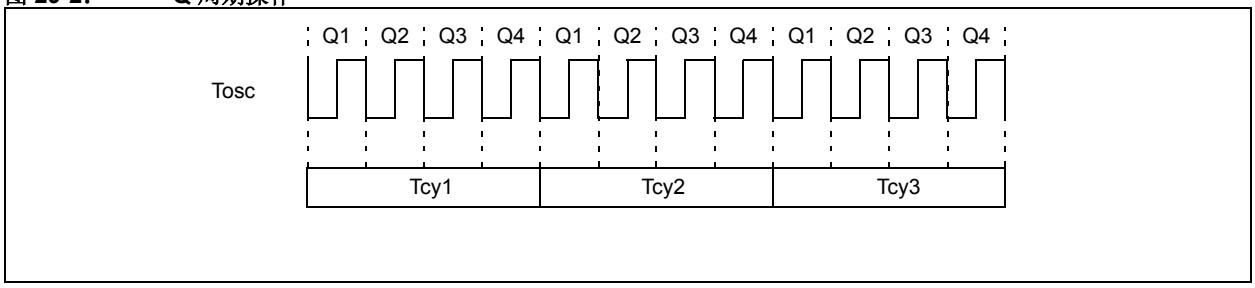
每一个指令周期 (Tcy) 由 4 个 Q 周期 (Q1-Q4) 组成。 Q 周期和器件振荡器周期 (Tosc) 相同。在每个指令周期，Q 周期提供译码、处理数据和读写操作的时序。下图给出了 Q 周期与指令周期的关系。

4 个 Q 周期组成一个指令周期 (Tcy)，它们是：

Q1：指令译码周期或无操作
Q2：指令读周期或无操作
Q3：处理数据
Q4：指令写周期或无操作

每条指令都有详细的 Q 周期操作。

图 29-2： Q 周期操作



29.5 指令描述

ADDLW 立即数与 W 寄存器内容相加

语法： [*标号*] ADDLW k

操作数： 0 ≤ k ≤ 255

操作： (W) + k → W

影响的状态位： C、 DC 和 Z

机器码：

11	111x	kkkk	kkkk

描述 8 位立即数 'k' 与 W 寄存器的内容相加，结果存入 W 寄存器。

指令字数： 1

指令周期： 1

Q 期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 'k'	处理 数据	写 W 寄存器

例 1

```
ADDLW   0x15
```

执行指令前：
W=0x10
执行后
W = 0x25

例 2

```
ADDLW   MYREG
```

执行指令前
W = 0x10
MYREG† 地址 = 0x37
† MYREG 是表示数据存储单元的符号
执行后
W = 0x47

例 3

```
ADDLW   HIGH (LU_TABLE)
```

执行指令前：
W = 0x10
LU_TABLE† 地址 = 0x9375
† LU_TABLE 为程序存储器中的地址标号
执行后
W = 0xA3

例 4

```
ADDLW   MYREG
```

执行指令前
W = 0x10
PCL† 地址 = 0x02
† PCL 是表示程序计数器低字节单元的符号
执行后
W = 0x12

ADDWF W 与 f 相加

语法： [*标号*] ADDWF f,d

操作数： $0 \le f \le 127$
$d \in [0,1]$

操作： (W) + (f) → 目标寄存器

影响的状态位： C、 DC 和 Z

机器码：

00	0111	dfff	ffff

描述： W 寄存器与 'f' 寄存器的内容相加。如果 'd' 为 0，结果存储在 W 寄存器中。如果 'd' 为 1，结果存储在 'f' 寄存器中。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 'f'	处理 数据	写 目标存储器

例 1
```
ADDWF   FSR, 0
```
执行指令前：
W = 0x17
FSR = 0xC2
执行后
W = 0xD9
FSR = 0xC2

例 2
```
ADDWF   INDF, 1
```
执行指令前
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x20
执行后
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x37

例 3
```
ADDWF   PCL
```
情况 1:
执行指令前
W = 0x10
PCL = 0x37
C = x
执行后
PCL = 0x47
C = 0

情况 2:
执行指令前
W = 0x10
PCL = 0xF7
PCH = 0x08
C = x
执行后
PCL = 0x07
PCH = 0x08
C = 1

ANDLW 立即数与 W 相与

语法： [*标号*] ANDLW k

操作数： 0 ≤ k ≤ 255

操作： (W).AND. (k) → W

影响的状态位： Z

机器码：

11	1001	kkkk	kkkk

描述： W 寄存器的内容与 8 位立即数 'k' 相与，结果存入 W 寄存器中。

指令字数： 1

指令周期： 1

Q 周期操作

Q1	Q2	Q3	Q4
译码	读立即数 'k'	处理数据	写 W 寄存器

例 1

```
ANDLW   0x5F

执行指令前：                    ; 0101 1111  (0x5F)
        W   =  0xA3             ; 1010 0011  (0xA3)
执行后                          ;----------  ------
        W   =  0x03             ; 0000 0011  (0x03)
```

例 2

```
ANDLW   MYREG

执行指令前
        W   =  0xA3
        MYREG† 地址 = 0x37
        † MYREG 是数据存储单元的标号
执行后
        W   =  0x23
```

例 3

```
ANDLW   HIGH (LU_TABLE)

执行指令前
        W   =  0xA3
        LU_TABLE† 地址 = 0x9375
        † LU_TABLE 为程序存储单元的标号
执行后
        W   =  0x83
```

ANDWF　　W 与 f 相与

语法：　[*标号*] ANDWF　f,d

操作数：　0 ≤ f ≤ 127
d ∈ [0,1]

操作：　(W).AND. (f) → 目标寄存器

影响的状态位：　Z

机器码：

00	0101	dfff	ffff

描述：　W 寄存器与 'f' 寄存器相与。如果 'd' 为 0，结果存入 W 寄存器。如果 'd' 为 1，结果存入 'f' 寄存器。

指令字数：　1

指令周期：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 'f'	处理 数据	写目标 寄存器

例 1

```
ANDWF   FSR, 1

执行指令前                          ; 0001 0111  (0x17)
       W  =  0x17                   ; 1100 0010  (0xC2)
      FSR =  0xC2                   ;----------  ------
执行后                              ; 0000 0010  (0x02)
      W   =  0x17
      FSR =  0x02
```

例 2

```
ANDWF   FSR, 0

执行指令前                          ; 0001 0111  (0x17)
       W  =  0x17                   ; 1100 0010  (0xC2)
      FSR =  0xC2                   ;----------  ------
执行后                              ; 0000 0010  (0x02)
      W   =  0x02
      FSR =  0xC2
```

例 3

```
ANDWF   INDF, 1

执行指令前
      W   =  0x17
      FSR =  0xC2
      地址 (FSR) 的内容 = 0x55
执行后
      W   =  0x17
      FSR =  0xC2
      地址 (FSR) 的内容 = 0x15
```

BCF 位清零

语法： [*标号*] BCF f,b

操作数： 0 ≤ f ≤ 127
0 ≤ b ≤ 7

操作： 0 → f<b>

影响的状态位： 无

机器码：

01	00bb	bfff	ffff

描述： 将寄存器 'f' 的位 'b' 清零。

指令字数： 1

指令周期： 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'

例 1

```
BCF     FLAG_REG, 7
```

执行指令前

FLAG_REG = 0xC7 ; **1**100 0111

执行后

FLAG_REG = 0x47 ; **0**100 0111

例 2

```
BCF     INDF, 3
```

执行指令前

W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x2F

执行后

W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x27

BSF　　位置 1

语法：　[*标号*] BSF　f,b

操作数：　0 ≤ f ≤ 127
0 ≤ b ≤ 7

操作：　1 → f<b>

影响的状态位：　无

机器码：

01	01bb	bfff	ffff

描述：　将寄存器 'f' 的位 'b' 置 1。

指令字数：　1

指令周期：　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'

例 1

```
BSF     FLAG_REG, 7
```

执行指令前

```
FLAG_REG =0x0A          ; 0000 1010
```

执行后

```
FLAG_REG =0x8A          ; 1000 1010
```

例 2

```
BSF     INDF, 3
```

执行指令前

```
W   =  0x17
FSR =  0xC2
```

地址 (FSR) 的内容 = 0x20

执行后

```
W   =  0x17
FSR =  0xC2
```

地址 (FSR) 的内容 = 0x28

BTFSC 位测试，为 0 跳过

语法： [*标号*] BTFSC f,b

操作数： 0 ≤ f ≤ 127
0 ≤ b ≤ 7

操作： skip if (f<b>) = 0

影响的状态位： 无

机器码：

01	10bb	bfff	ffff

描述： 如果寄存器 'f' 的位 'b' 为 '0'，则跳过下一条指令。
如果位 'b' 为 '0'，那么下一条指令 (在当前指令执行期间取指) 被丢弃而执行一条 NOP 指令，使该指令变成 2 周期指令。

指令字数： 1

指令周期： 1(2)

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 'f'	处理 数据	无 操作

如果跳过 (第二个周期):

Q1	Q2	Q3	Q4
无 操作	无 操作	无 操作	无 操作

例 1

```
HERE    BTFSC   FLAG, 4
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
        •
```

情况 1:
执行指令前
PC = 地址 HERE
FLAG= xxx0 xxxx
执行后
由于 FLAG<4>= 0,
PC = 地址 TRUE

情况 2:
执行指令前
PC = 地址 HERE
FLAG= xxx1 xxxx
执行后
由于 FLAG<4>=1,
PC = 地址 FALSE

BTFSS 位测试，为 1 跳过

语法： [*标号*] BTFSS f,b

操作数： 0 ≤ f ≤ 127
0 ≤ b < 7

操作： 跳过，如果 (f<b>) = 1

影响的状态位： 无

机器码：

01	11bb	bfff	ffff

描述： 如果寄存器 'f' 的位 'b' 为 '1'，则跳过下一条指令。
如果位 'b' 为 '1'，那么下一条指令 (在当前指令执行期间取指) 被丢弃而执行一条 NOP 指令，使该指令变成 2 周期指令。

指令字数： 1

指令周期： 1(2)

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 'f'	处理数据	无操作

如果跳过 (第二个周期):

Q1	Q2	Q3	Q4
无操作	无操作	无操作	无操作

例 1

```
HERE    BTFSS   FLAG, 4
FALSE   GOTO    PROCESS_CODE
TRUE    •
        •
        •
```

情况 1:
执行指令前
PC = 地址 HERE
FLAG= xxx0 xxxx
执行后
由于 FLAG<4>= 0,
PC = 地址 FALSE

情况 2:
执行指令前
PC = 地址 HERE
FLAG= xxx1 xxxx
执行后
由于 FLAG<4>=1,
PC = 地址 TRUE

CALL 调用子程序

语法： [*标号*] CALL k

操作数： $0 \le k \le 2047$

操作： (PC)+ 1→ TOS,
k → PC<10:0>,
(PCLATH<4:3>) → PC<12:11>

影响的状态位： 无

机器码：

10	0kkk	kkkk	kkkk

描述： 调用子程序。首先， 13 位返回地址 (PC+1) 被压入堆栈。 11 位立即数地址被装入 PC 位 <10:0>。PC 高位从 PCLATH<4:3> 装入。CALL 为一条两周期指令。

指令字数： 1

指令周期： 2

Q 周期操作：

第一个周期：

Q1	Q2	Q3	Q4
译码	读立即数 'k'	处理数据	无操作

第二个周期：

Q1	Q2	Q3	Q4
无操作	无操作	无操作	无操作

例 1

```
HERE    CALL    THERE
```

执行指令前
PC = 地址HERE
执行后
TOS = 地址 HERE+1
PC = 地址 THERE

CLRF 寄存器 f 清零

语法： [*标号*] CLRF f

操作数： 0 ≤ f ≤ 127

操作： 00h → f
1 → Z

影响的状态位： Z

机器码：

00	0001	1fff	ffff

描述： 寄存器 'f' 被清零， Z 位置 1。

指令字数： 1

指令周期： 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'

例 1

```
CLRF    FLAG_REG
```

执行指令前：
FLAG_REG=0x5A
执行后
FLAG_REG=0x00
Z = 1

例 2

```
CLRF    INDF
```

执行指令前
FSR = 0xC2
地址 (FSR)的内容 =0xAA
执行后
FSR = 0xC2
地址 (FSR) 的内容=0x00
Z = 1

CLRW　　W 寄存器清零

语法：　[*标号*]　CLRW

操作数：　无

操作：　00h → W
1 → Z

影响的状态位：　Z

机器码：

00	0001	0xxx	xxxx

描述：　W 寄存器被清零。 Z 位置 1。

指令字数：　1

指令周期：　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 'W'

例 1

```
CLRW
```

执行指令前
W = 0x5A
执行后
W = 0x00
Z = 1

CLRWDT 看门狗定时器 WDT 清零

语法： [*标号*] CLRWDT

操作数： 无

操作： 00h → WDT
0 → WDT 预分频器计数，
1 → $\overline{TO}$
1 → $\overline{PD}$

影响的状态位： $\overline{TO}$, $\overline{PD}$

机器码：

00	0000	0110	0100

描述： CLRWDT 指令清零看门狗定时器，同时将 WDT 的预分频器计数值清零。状态位 $\overline{TO}$ 和 $\overline{PD}$ 被置 1。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	无操作	处理数据	清零 WDT 计数器

例 1 CLRWDT

执行指令前
WDT 计数器= x
WDT 预分频器分频比 =1:128
执行后
WDT 计数器=0x00
WDT 预分频器计数=0
$\overline{TO}$ = 1
$\overline{PD}$ = 1
WDT 预分频器分频比 =1:128

注： CLRWDT 指令并不影响 WDT 预分频器的分配。

COMF　　f 寄存器内容取反

语法：　[*标号*]　COMF　f,d

操作数：　$0 \le f \le 127$
$d \in [0,1]$

操作：　$(\bar{f}) \rightarrow$ 目标寄存器

影响的状态位：　Z

机器码：

00	1001	dfff	ffff

描述：　寄存器 'f' 的内容取反。如果 'd' 为 0，结果存入 W；如果 'd' 为 1，结果存入寄存器 'f'。

指令字数：　1

指令周期：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 'f'	处理 数据	写 目标寄存器

例 1

```
COMF    REG1, 0
```

执行指令前
REG1=　0x13
执行后
REG1=　0x13
W　=　0xEC

例 2

```
COMF    INDF, 1
```

执行指令前
FSR =　0xC2
地址 (FSR)的内容 =0xAA
执行后
FSR =　0xC2
地址 (FSR)的内容=0x55

例 3

```
COMF    REG1, 1
```

执行指令前
REG1=　0xFF
执行后
REG1=　0x00
Z　=　1

DECF　　f 寄存器内容减 1

语法：　[*标号*]　DECF f,d

操作数：　0 ≤ f ≤ 127
d ∈ [0,1]

操作：　(f) - 1 → 目标寄存器

影响的状态位：　Z

机器码：

00	0011	dfff	ffff

描述：　寄存器 'f' 内容减 1。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1，结果存回 'f' 寄存器。

指令字数：　1

指令周期：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 'f'	处理 数据	写 目标寄存器

例 1

```
DECF    CNT, 1
```

执行指令前
CNT = 0x01
Z = 0
执行后
CNT = 0x00
Z = 1

例 2

```
DECF    INDF, 1
```

执行指令前
FSR = 0xC2
地址 (FSR) 的内容 = 0x01
Z = 0
执行后
FSR = 0xC2
地址 (FSR) 的内容 = 0x00
Z = 1

例 3

```
DECF    CNT, 0
```

执行指令前
CNT = 0x10
W = x
Z = 0
执行后
CNT = 0x10
W = 0x0F
Z = 0

DECFSZ　　f 寄存器内容减 1，为 0 跳过

语法：　[*标号*]　DECFSZ　f,d

操作数：　0 ≤ f ≤ 127
d ∈ [0,1]

操作：　(f) - 1 → 目标寄存器；结果为 0 跳过

影响的状态位：　无

机器码：

00	1011	dfff	ffff

描述：　寄存器 'f' 的内容减 1。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1，结果存入寄存器 'f'。
如果结果为 0，那么下一条指令 (在当前指令执行期间取指) 被丢弃而执行一条 NOP 指令，使该指令变成 2 周期指令。

指令字数：　1

指令周期：　1(2)

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 'f'	处理 数据	写 目标寄存器

如果跳过 (第二个周期):

Q1	Q2	Q3	Q4
无操作	无操作	无操作	无操作

例

```
HERE     DECFSZ   CNT, 1
         GOTO     LOOP
CONTINUE •
         •
         •
```

情况 1:
执行指令前
PC = 地址 HERE
CNT = 0x01
执行后
CNT = 0x00
PC = 地址 CONTINUE

情况 2:
执行指令前
PC = 地址 HERE
CNT = 0x02
执行后
CNT = 0x01
PC = 地址 HERE + 1

GOTO 无条件转移

语法： [*标号*] GOTO k

操作数： $0 \le k \le 2047$

操作： $k \rightarrow PC<10:0>$
$PCLATH<4:3> \rightarrow PC<12:11>$

影响的状态位： 无

机器码：

10	1kkk	kkkk	kkkk

描述： GOTO 是无条件转移指令。11 位立即数被装入 PC 位 <10:0>，PC 的高位从 PCLATH<4:3> 装入。GOTO 为 2 周期指令。

指令字数： 1

指令周期： 2

Q 周期操作：

第一个周期：

Q1	Q2	Q3	Q4
译码	读数 'k'<7:0>	处理数据	无操作

第二个周期：

Q1	Q2	Q3	Q4
无操作	无操作	无操作	无操作

例

```
GOTO THERE
```

执行后

PC = 地址 THERE

INCF　　f 寄存器内容加 1

语法：　[*标号*]　INCF　f,d

操作数：　$0 \le f \le 127$
$d \in [0,1]$

操作：　(f) + 1 → 目标寄存器

影响的状态位：　Z

机器码：

00	1010	dfff	ffff

描述：　寄存器 'f' 的内容加 1。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1 结果存回 'f' 寄存器。

指令字数：　1

指令周期：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 'f'	处理 数据	写 目标寄存器

例 1

```
INCF    CNT, 1
```

执行指令前
CNT = 0xFF
Z = 0
执行后
CNT = 0x00
Z = 1

例 2

```
INCF    INDF, 1
```

执行指令前
FSR = 0xC2
地址 (FSR) 的内容 = 0xFF
Z = 0
执行后
FSR = 0xC2
地址 (FSR) 的内容 = 0x00
Z = 1

例 3

```
INCF    CNT, 0
```

执行指令前
CNT = 0x10
W = x
Z = 0
执行后
CNT = 0x10
W = 0x11
Z = 0

INCFSZ　　f 寄存器内容加 1, 为 0 跳过

语法：　[*l 标号*]　INCFSZ　f,d

操作数：　0 ≤ f ≤ 127
d ∈ [0,1]

操作：　(f) + 1 → 目标寄存器，结果为 0 跳过

影响的状态位：　无

机器码：

00	1111	dfff	ffff

描述：　寄存器 'f' 的内容加 1。如果 'd' 为 0 结果存入 W 寄存器；如果 'd' 为 1 结果存回 'f' 寄存器。
如果结果为 0，那么下一条指令 (在当前指令执行期间取指) 被丢弃而执行一条 NOP 指令，使该指令变成 2 周期指令。

指令字数：　1

指令周期：　1(2)

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 寄存器 'f'	处理 数据	写 目标寄存器

如果跳过 (第二个周期):

Q1	Q2	Q3	Q4
无操作	无操作	无操作	无操作

例

```
HERE     INCFSZ   CNT, 1
         GOTO     LOOP
CONTINUE •
         •
         •
```

情况 1:　执行指令前
PC = 地址 HERE
CNT = 0xFF
执行后
CNT = 0x00
PC = 地址 CONTINUE

情况 2:　执行指令前
PC = 地址 HERE
CNT = 0x00
执行后
CNT = 0x01
PC = 地址 HERE + 1

IORLW 8 位立即数和 W 寄存器内容相或

语法： [*标号*] IORLW k

操作数： 0 ≤ k ≤ 255

操作： (W).OR. k → W

影响的状态位： Z

机器码：

11	1000	kkkk	kkkk

描述： W 寄存器的内容与 8 位立即数 'k' 相或，结果存入 W 寄存器。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读 立即数 'k'	数据处理	写 W 寄存器

例 1

```
IORLW   0x35
```

执行指令前
W = 0x9A
执行后
W = 0xBF
Z = 0

例 2

```
IORLW   MYREG
```

执行指令前
W = 0x9A
MYREG† 地址 = 0x37
† MYREG 是数据存储单元的符号
执行后
W = 0x9F
Z = 0

例 3

```
IORLW   HIGH (LU_TABLE)
```

执行指令前
W = 0x9A
LU_TABLE† 地址 = 0x9375
† LU_TABLE 是程序存储单元的标号
执行后
W = 0x9B
Z = 0

例 4

```
IORLW   0x00
```

执行指令前
W = 0x00
执行后
W = 0x00
Z = 1

IORWF　　W 寄存器内容与 f 寄存器内容相或

语法：　[*l 标号*]　IORWF　f,d

操作数：　0 ≤ f ≤ 127
d ∈ [0,1]

操作：　(W).OR. (f) → 目标寄存器

影响的状态位：　$\overline{Z}$

机器码：

00	0100	dfff	ffff

描述：　W 寄存器内容与 f 寄存器内容相或。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1，结果存入 'f' 寄存器。

指令字数：　1

指令周期：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 'f'	处理数据	写目标寄存器

例 1

```
IORWF   RESULT, 0
```

执行指令前
RESULT=0x13
W = 0x91
执行后
RESULT=0x13
W = 0x93
Z = 0

例 2

```
IORWF   INDF, 1
```

执行指令前
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x30
执行后
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x37
Z = 0

例 3

```
IORWF   RESULT, 1
```

情况 1:　执行指令前
RESULT=0x13
W = 0x91
执行指令前
RESULT=0x93
W = 0x91
Z = 0

情况 2:　执行指令前
RESULT=0x00
W = 0x00
执行后
RESULT=0x00
W = 0x00
Z = 1

MOVLW 立即数送入 W 寄存器

语法： [*标号*] MOVLW k

操作数： $0 \le k \le 255$

操作： $k \rightarrow W$

影响的状态位： 无

机器码：

11	00xx	kkkk	kkkk

描述： 8 位立即数 'k' 送入 W 寄存器。无关的位置为 0。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 'k'	处理数据	写 W 寄存器

例 1

```
MOVLW   0x5A
```

执行后

W = 0x5A

例 2

```
MOVLW   MYREG
```

执行指令前

W = 0x10

MYREG† 地址 = 0x37

† MYREG 是数据存储单元的符号

执行后

W = 0x37

例 3

```
MOVLW   HIGH (LU_TABLE)
```

执行指令前

W = 0x10

of LU_TABLE 地址 = 0x9375

† LU_TABLE 程序存储单元的标号

执行后

W = 0x93

MOVF 寄存器 f 传送

语法： [*标号*] MOVF f,d

操作数： 0 ≤ f ≤ 127
d ∈ [0,1]

操作： (f) → 目标寄存器

影响的状态位 Z

机器码：

00	1000	dfff	ffff

描述： 将寄存器 'f' 的内容送入目标寄存器。如果 'd' = 0，目标寄存器为 W 寄存器；如果 'd' = 1，目标寄存器为 'f' 寄存器本身。由于状态标志位 Z 受到指令结果的影响，'d' = 1 可用于检测文件寄存器内容是否为 0。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 'f'	处理数据	写目标寄存器

例 1

```
MOVF    FSR, 0
```

执行指令前
W = 0x00
FSR = 0xC2
执行后
W = 0xC2
Z = 0

例 2

```
MOVF    INDF, 0
```

执行指令前
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x00
执行后
W = 0x00
FSR = 0xC2
地址 (FSR) 的内容 = 0x00
Z = 1

例 3

```
MOVF    FSR, 1
```

情况 1: 执行指令前
FSR = 0x43
执行后
FSR = 0x43
Z = 0

情况 2: 执行指令前
FSR = 0x00
执行后
FSR = 0x00
Z = 1

MOVWF　　W 内容传送至 f

语法：　[*标号*]　MOVWF　f

操作数：　$0 \le f \le 127$

操作：　$(W) \rightarrow f$

影响的状态位：　无

机器码：

00	0000	1fff	ffff

描述：　将数据从 W 寄存器送入 'f' 寄存器。

指令字数：　1

指令周期：　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'

例 1

```
MOVWF   OPTION_REG
```

执行指令前
OPTION_REG=0xFF
W = 0x4F
执行后
OPTION_REG=0x4F
W = 0x4F

例 2

```
MOVWF   INDF
```

执行指令前
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x00
执行后
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x17

NOP 空操作

语法： [*标号*] NOP

操作数： 无

操作： 空操作

影响的状态位： 无

机器码：

00	0000	0xx0	0000

描述： 无操作

指令字数： 1

指令周期： 1

Q 周期操作

Q1	Q2	Q3	Q4
译码	空操作	空操作	空操作

例：

```
HERE    NOP
```

执行指令前

PC = HERE 地址

执行后

PC = HERE 地址 + 1

OPTION　　Option 寄存器赋值

语法：	[*标号*]　OPTION
操作数：	无
操作：：	(W) → OPTION
影响的状态位：：	无
机器码：	00 | 0000 | 0110 | 0010
描述：：	此指令将 W 寄存器的内容送入 OPTION 寄存器。支持这条指令是为了与 PIC16C5X 产品代码兼容。 OPTION 为一个可读 / 写寄存器，用户可对其直接寻址。
指令字数：	1
指令周期：：	1

为保持与未来 PIC16CXX 产品的向上兼容，不要使用该指令。

RETFIE 中断返回

语法： [*标号*] RETFIE

操作数： 无

操作： TOS → PC，
1 → GIE

影响的状态位 无

机器码：

00	0000	0000	1001

描述： 从中断返回。栈顶（TOS）的 13 位地址装入 PC。全局中断允许位 GIE (INTCON<7>)，自动置 1，允许中断。这是一条 2 周期指令。

指令字数： 1

指令周期： 2

Q 周期操作：

第一个周期：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	空操作

第二个周期：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

例

```
RETFIE
```

执行后

```
PC  =  TOS
GIE =  1
```

RETLW 带立即数子程序返回

语法： [*标号*] RETLW k

操作数： 0 ≤ k ≤ 255

操作： k → W ；
TOS → PC

影响的状态位： 无

机器码：

11	01xx	kkkk	kkkk

描述： 8 位立即数 'k' 装载至 W 寄存器，栈顶的 13 位地址 (返回地址) 送入程序计数器。这是一条 2 周期指令。

指令字数： 1

指令周期： 2

Q 周期操作：

第一个周期：

Q1	Q2	Q3	Q4
译码	读立即数 'k'	处理数据	写 W 寄存器

第二个周期：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

Example

```
HERE    CALL TABLE   ; W contains table
                     ; offset value
        •            ; W now has table value
        •
        •
TABLE   ADDWF PC     ;W = offset
        RETLW k1     ;Begin table
        RETLW k2     ;
        •
        •
        •
        RETLW kn     ; End of table
```

执行指令前

W = 0x07

执行后

W = k8 的值

PC = TOS = Here 地址 + 1

RETURN 子程序返回

语法： [*标号*] RETURN

操作数： 无

操作： TOS → PC

影响的状态位： 无

机器码：

00	0000	0000	1000

描述： 从子程序返回。堆栈中的数据被弹出，栈顶 (TOS) （即返回地址）被装入程序计数器。这是一条 2 周期指令。

指令字数： 1

指令周期： 2

Q 周期操作：

第一个指令周期：

Q1	Q2	Q3	Q4
译码	空操作	处理数据	空操作

第二个指令周期：

Q1	Q2	Q3	Q4
空操作	空操作	空操作	空操作

例

```
HERE    RETURN
```

执行后

PC = TOS

RLF 带进位位循环左移

语法： [*标号*] RLF f,d

操作数： $0 \le f \le 127$
$d \in [0,1]$

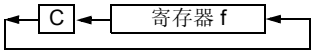
操作： 参见下面的描述

影响的状态位： C

机器码：

00	1101	dfff	ffff

描述： 寄存器 'f' 的内容带进位位循环左移。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1，结果存入 'f' 寄存器。



指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 'f'	处理数据	写目标寄存器

例 1

```
RLF     REG1,0
```

执行指令前
REG1= 1110 0110
C = 0
执行后
REG1=1110 0110
W =1100 1100
C =1

例 2

```
RLF     INDF, 1
```

情况 1: 执行指令前
W = xxxx xxxx
FSR = 0xC2
地址 (FSR) 的内容 = 0011 1010
C = 1
执行后
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0111 0101
C = 0

情况 2: 执行指令前
W = xxxx xxxx
FSR = 0xC2
地址 (FSR) 的内容 = 1011 1001
C = 0
执行后
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0111 0010
C = 1

RRF 带进位位循环右移

语法： [*标号*] RRF f,d

操作数： 0 ≤ f ≤ 127
d ∈ [0,1]

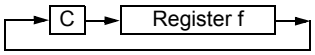
操作： 参见下面的描述

影响的状态位： C

机器码：

00	1100	dfff	ffff

描述： 寄存器 'f' 的内容带进位循环右移。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1，结果存入 'f' 寄存器。



指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 'f'	处理数据	写目标寄存器

例 1

```
RRF          REG1,0
```

执行指令前
```
REG1= 1110 0110
W   =  xxxx xxxx
C   =  0
```
执行后
```
REG1= 1110 0110
W   =  0111 0011
C   =  0
```

例 2

```
RRF     INDF, 1
```

情况 1: 执行指令前
```
W   =  xxxx xxxx
FSR =  0xC2
地址 (FSR) 的内容 = 0011 1010
C   =  1
```
执行后
```
W   =  0x17
FSR =  0xC2
地址 (FSR) 的内容  = 1001 1101
C   =  0
```

情况 2: 执行指令前
```
W   =  xxxx xxxx
FSR =  0xC2
地址 (FSR) 的内容 = 0011 1001
C   =  0
```
执行后
```
W   =  0x17
FSR =  0xC2
地址 (FSR) 的内容 = 0001 1100
C   =  1
```

SLEEP

语法： [*标号*] SLEEP

操作数： 无

操作： 00h → WDT，
0 → WDT 预分频器计数，
1 → $\overline{TO}$,
0 → $\overline{PD}$

影响的状态位 $\overline{TO}$, $\overline{PD}$

机器码：

00	0000	0110	0011

描述： 掉电状态位 $\overline{PD}$ 清零。超时状态位 $\overline{TO}$ 位置 1。看门狗定时器及其预分频器计数清零。
振荡器停止工作，单片机进入休眠模式。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	空操作	空操作	进入休眠模式

例： SLEEP

注： SLEEP 指令并不影响 WDT 预分频器的分配。

SUBLW 立即数与 W 寄存器的内容相减

语法： [*标号*] SUBLW k

操作数： $0 \le k \le 255$

操作： $k - (W) \rightarrow W$

影响的状态位： C、 DC 和 Z

机器码：

11	110x	kkkk	kkkk

描述： 8 位立即数减去 W 寄存器内容 (通过二进制补码方法进行减法运算)，结果存入 W 寄存器。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 'k'	处理数据	写 W 寄存器

例 1:
```
SUBLW   0x02
```
情况 1: 执行指令前
```
W   =  0x01
C   =  x
Z   =  x
```
执行后
```
W   =  0x01
C   =  1        ; 结果为正
Z   =  0
```
情况 2: 执行指令前
```
W   =  0x02
C   =  x
Z   =  x
```
执行后
```
W   =  0x00
C   =  1        ; 结果为 0
Z   =  1
```
情况 3: 执行指令前
```
W   =  0x03
C   =  x
Z   =  x
```
执行后
```
W   =  0xFF
C   =  0        ; 结果为负
Z   =  0
```
例 2
```
SUBLW   MYREG
```
执行指令前
```
W   =  0x10
```
MYREG† 地址 = 0x37

† MYREG 是数据存储单元的符号

执行后
```
W   =  0x27
C   =  1    ; 结果为正
```

SUBWF　　f 寄存器内容减 W 寄存器内容

语法：　[*标号*]　SUBWF　f,d

操作数：　0 ≤ f ≤ 127
d ∈ [0,1]

操作：　(f) - (W) → 目标寄存器

影响的状态位　C、 DC 和 Z

机器码：

00	0010	dfff	ffff

描述：　f 寄存器内容减去 W 寄存器内容。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1，结果存入 'f' 寄存器。

指令字数：　1

指令周期：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 'f'	处理数据	写 目标寄存器

例 1:　SUBWF　REG1,1

情况 1:　执行指令前

```
REG1= 3
W   = 2
C   = x
Z   = x
```

执行后

```
REG1= 1
W   = 2
C   = 1          ; 结果为正
Z   = 0
```

情况 2:　执行指令前

```
REG1= 2
W   = 2
C   = x
Z   = x
```

执行后

```
REG1= 0
W   = 2
C   = 1          ; 结果为 0
Z   = 1
```

情况 3:　执行指令前

```
REG1= 1
W   = 2
C   = x
Z   = x
```

执行后

```
REG1= 0xFF
W   = 2
C   = 0          ; 结果为负
Z   = 0
```

SWAPF 寄存器半字节交换

语法： [*标号*] SWAPF f,d

操作数： 0 ≤ f ≤ 127
d ∈ [0,1]

操作： (f<3:0>) → 目标寄存器 <7:4>,
(f<7:4>) → 目标寄存器 <3:0>

影响的状态位： 无

机器码：

00	1110	dfff	ffff

描述： 寄存器 'f' 的高半字节和低半字节交换。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1，结果存入 'f' 寄存器。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 'f'	处理数据	写目标寄存器

例 1

```
SWAPF   REG, 0
```

执行指令前
REG1= 0xA5

执行后
REG1= 0x5A
W = 0x5A

例 2

```
SWAPF   INDF, 1
```

执行指令前
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x20

执行后
W = 0x17
FSR = 0xC2
地址 (FSR) 的内容 = 0x02

例 3

```
SWAPF   REG, 1
```

执行指令前
REG1= 0xA5

执行后
REG1= 0x5A

TRIS 装载 TRIS 寄存器

语法：	[*标号*] TRIS f
操作数：	5 ≤ f ≤ 7
操作：	(W) → TRIS 寄存器 f；
影响的状态位：	无
机器码：	00 | 0000 | 0110 | 0fff
描述：	支持该指令是为了与 PIC16C5X 产品代码兼容。 TRIS 寄存器可读写，用户可直接对它们寻址。
指令字数：	1
指令周期：	1

例

为保持与未来 PIC16CXX 产品的向上兼容，不要使用该指令。

XORLW 立即数异或 W 寄存器

语法： [*标号*] XORLW k

操作数： $0 \le k \le 255$

操作： (W).XOR. k → W

影响的状态位： Z

机器码：

11	1010	kkkk	kkkk

描述： 寄存器 W 的内容与 8 位立即数 'k' 进行异或运算。结果存入 W 寄存器。

指令字数： 1

指令周期： 1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读立即数 'k'	处理数据	写 W 寄存器

例 1

```
XORLW   0xAF                ; 1010 1111   (0xAF)
执行指令前                  ; 1011 0101   (0xB5)
        W   =  0xB5         ; ---------   ------
执行后                      ; 0001 1010   (0x1A)
        W   =  0x1A
        Z   =  0
```

例 2

```
XORLW   MYREG
执行指令前
        W   =  0xAF
        MYREG† 地址 = 0xB7
        † MYREG 是数据存储单元的符号
执行后
        W   =  0x18
        Z   =  0
```

例 3

```
XORLW   HIGH (LU_TABLE)
执行指令前
        W   =  0xAF
        LU_TABLE† 地址 = 0x9375
        † LU_TABLE 是程序存储单元的标号
        执行后
        W   =  0x3C
        Z   =  0
```

XORWF　　W 寄存器内容异或 f 寄存器内容

语法：　[*标号*]　XORWF　f,d

操作数：　$0 \le f \le 127$
$d \in [0,1]$

操作：　(W).XOR. (f) → 目标寄存器

影响的状态位：　Z

机器码：

00	0110	dfff	ffff

描述：　W 寄存器内容与 f 寄存器内容进行异或运算。如果 'd' 为 0，结果存入 W 寄存器；如果 'd' 为 1，结果存入寄存器 "f"。

指令字数：　1

指令周期：　1

Q 周期操作：

Q1	Q2	Q3	Q4
译码	读寄存器 'f'	处理数据	写目标寄存器

例 1

```
XORWF    REG, 1                ; 1010 1111   (0xAF)
```
执行指令前
```
                               ; 1011 0101   (0xB5)
         REG=  0xAF            ; ---------   ------
         W   =  0xB5           ; 0001 1010   (0x1A)
```
执行后
```
         REG=  0x1A
         W   =  0xB5
```

例 2

```
XORWF    REG, 0                ; 1010 1111   (0xAF)
```
执行指令前
```
                               ; 1011 0101   (0xB5)
         REG=  0xAF            ; ---------   ------
         W   =  0xB5           ; 0001 1010   (0x1A)
```
执行后
```
         REG=  0xAF
         W   =  0x1A
```

例 3

```
  XORWF   INDF, 1
```
执行指令前
```
         W   =  0xB5
         FSR =  0xC2
         地址 (FSR) 的内容 = 0xAF
```
执行后
```
         W   =  0xB5
         FSR =  0xC2
         地址 (FSR) 的内容 = 0x1A
```

29.6 设计技巧

问 1： ***如何直接修改 W 寄存器的值？我想将 W 寄存器内容递减。***

答 1：

有多种方法，现给出两种方法：

1. 对于中档单片机来说，有几条对立即数和 W 寄存器进行操作的指令。例如，如果希望 W 寄存器减 1，可以用 ADDLW 0xFF （0x 向汇编器指明这是十六进制数）。
2. 注意所有指令都可以直接修改文件寄存器中的值。这意味着，可以直接将文件寄存器中的值减 1，而不需要先将文件寄存器中的值传送至 W 寄存器。如果想将文件寄存器的值减 1 后，再移至其它寄存器，可以先使用以 W 寄存器为目标寄存器的减 1 指令 (DECF 寄存器，W)，然后将 W 的值放到指定寄存器。这样做和直接将寄存器中的值移到另一个寄存器所用的指令数一样，但这样做的同时还完成了减 1 操作。

问 2： ***对于 PIC16CXXX 系列单片机，在使用 TRIS 指令时有危险吗？数据手册中建议不要使用该指令。***

答 2：

从代码兼容性和产品升级来说，不推荐使用 TRIS 指令。应该注意 TRIS 指令只能用于端口 A、B 和 C。将来的器件可能不支持 TRIS 指令。

问 3： ***在使用 TRIS 指令之前，一定要切换到数据存储器的 Bank1 吗（指存储器中包含 TRIS 寄存器的器件）？***

答 3：

不需要。TRIS 指令与存储区无关。不推荐使用 TRIS 指令。

问 4： ***读了数据手册上关于读－修改－写指令的描述后，我不太明白，能解释一下吗？为什么需要了解这些呢？***

答 4：

读－修改－写 (R-M-W) 指令的一个简单例子是位清零指令 BCF。你或许会认为单片机对端口输出引脚执行位清零指令，会直接将引脚清零。而实际上，单片机首先读整个端口 （寄存器），然后进行位清零操作，最后将修改的新值写入端口 （寄存器）。实际上，任何依赖于寄存器中当前值的指令都是读－修改－写指令，这包括 ADDWF、 SUBWF、 BCF、 BSF、 INCF 和 XORWF 等等。不依赖于寄存器当前值的指令，像 MOVWF、 CLRF 等，不是读－修改－写指令。

对于经常在输入和输出之间切换的端口，需要考虑读－修改－写指令的影响。例如，将所有 TRISB 位置 0，将所有 PORTB 引脚配置为输出，然后向 PORTB 寄存器写 0XFF，那么所有 PORTB 引脚都变为高电平。现在，假设将 RB3 引脚设置为输入，而其输入为低电平，然后执行 BCF PORTB,6 指令，将 RB6 引脚驱动为低电平。这时如果将 RB3 重新设置为输出，尽管最后写到这一位的是 1，但其输出为低电平。这是由于对 RB6 引脚进行 BCF 操作时，先读入整个 B 端口，包括 RB3 引脚为输入时的 ‘0’。然后 bit 6 清零，结果被写回 B 端口的锁存器，这时由于 RB3 读为 ‘0’ ，这个 ‘0’ 值被写回数据锁存器，将原来的值 ‘1’ 覆盖掉。因此当引脚变为输出时，RB3 将输出新值 （‘0’）。

问 5： ***当执行 BCF 指令时，端口的其它引脚被清零了，为什么？***

答 5：

这有几种可能性，其中两种可能性为：

1. 另举一个读－修改－写指令意外改变其它引脚值的例子：假设将 PORTC 所有引脚配置为输出，并将引脚驱动为低电平。在每个端口引脚上连接一个另一端接地的 LED，这样引脚输出高电平时将点亮 LED。每个 LED 旁并联了一个 100 µF 的电容。同样假设单片机的运行速度非常快，比方说 20 MHz。现在如果依次置 1 各个引脚：BSF PORTC,0，然后 BSF PORTC,1，然后 BSF PORTC,2 等等，你可能会发现此时只有最后一个引脚被置 1，只有最后一个 LED 被点亮。这是因为电容充电需要时间。当置 1 每个引脚时，其前一个引脚并没有充电到高电平，所以前一个引脚读为零，这个零值被写回端口锁存器 (记住：执行的是读－修改－写指令)，从而使该位清零。当单片机运行速度比较高，进行连续端口操作时，通常需要考虑这个问题。
2. 如果使用的是 PIC16C7X 系列单片机，你可能没有在 ADCON1 寄存器中正确地配置 I/O 引脚。如果一个引脚配置为模拟输入，那么无论引脚上的电压为多少，每次读该引脚时，其结果均为零。总是读引脚状态是一个特例。你可以通过 TRIS 寄存器将模拟引脚设置成输出，然后写该引脚使其输出高电平或低电平，但是读该引脚时结果将始终为零。因此执行读－修改－写指令 (参见前面的问题) 时，所有模拟引脚均读为零，那些未被指令直接修改的读入值可以将端口锁存器重新写为零。对于设置成模拟的引脚，其引脚值可能既不是逻辑高电平也不是逻辑低电平，也不是浮动的。数字引脚应禁止输入端悬空，因为可能导致输入缓冲器电流消耗过大而使输入缓冲器工作失效。

29.7 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定的限制即可使用。目前与指令集相关的应用笔记有：

目前没有相关的应用笔记

29.8 版本历史

版本 A

这是描述指令集的初始发行版。

第 30 章 电气规范

目录

30.1 简介 30-2
30.2 绝对最大值 30-3
30.3 器件选型表 30-4
30.4 器件电压规范 30-5
30.5 器件电流特性 30-6
30.6 输入阈值电平 30-9
30.7 I/O 电流特性 30-10
30.8 输出驱动电压 30-11
30.9 I/O 引脚的容性负载 30-12
30.10 数据 EEPROM / 闪存 30-13
30.11 LCD 30-14
30.12 比较器和参考电压 30-15
30.13 时序参数符号 30-16
30.14 外部时钟时序波形图和时序要求示例 30-17
30.15 上电和复位时序波形图及要求示例 30-19
30.16 定时器 Timer0 和 Timer1 时序波形图及要求示例 30-20
30.17 CCP 的时序图及要求 30-21
30.18 并行从动端口 (PSP) 时序图及要求 30-22
30.19 SSP 和 MSSP SPI 模式时序波形图及要求示例 30-23
30.20 SSP I^2C 模式时序波形图及要求示例 30-27
30.21 MSSP I^2C 模式时序波形图及要求示例 30-30
30.22 USART/SCI 时序波形图及要求示例 30-32
30.23 8 位 A/D 时序波形图及要求示例 30-34
30.24 10 位 A/D 时序波形图及要求示例 30-36
30.25 积分型 A/D 时序波形图及要求示例 30-38
30.26 LCD 时序波形图及要求示例 30-40
30.27 相关应用笔记 30-41
30.28 版本历史 30-42

30.1 简介

本章旨在介绍器件数据手册中所规定的电气规范及其含义，其目的并**非**给出具体参数值。应将本章出现的所有值视为例子，要查找器件的具体值，**必须**参考具体器件的数据手册。

在前面各章节中关于器件及功能模块的描述中出现有电气规范参数的引用。在本手册的电子版中，这些引用以超级链接的形式出现，以便用户使用。

注： 在开始设计之前， Microchip 公司强烈建议您先取得最新的器件数据手册，并查看其电气规范的说明以确保其满足需求。

本章中使用的术语如表 30-1 所示。

表 30-1: 常用术语

术语	说明
PIC16**C**XXX	通过标准电压范围测试的器件
PIC16**LC**XXX	通过扩展电压范围测试的器件
PIC16**F**XXX	通过标准电压范围测试的器件
PIC16**LF**XXX	通过扩展电压范围测试的器件
PIC16**CR**XXX	通过标准电压范围测试的器件
PIC16**LCR**XXX	通过扩展电压范围测试的器件
PIC16XXXX-**04**	通过 4MHz 工作频率测试的器件
PIC16XXXX-**08**	通过 8MHz 工作频率测试的器件
PIC16XXXX-**10**	通过 10MHz 工作频率测试的器件
PIC16XXXX-**20**	通过 20 MHz 工作频率测试的器件
LP osc	器件配置为 LP 振荡器模式
XT osc	器件配置为 XT 振荡器模式
HS osc	器件配置为 HS 振荡器模式
RC osc	器件配置为 RC 振荡器模式
商业级	器件工作温度范围为商业级 (0°C ≤ TA ≤ +70°C)
工业级	器件工作温度范围为工业级 (-40°C ≤ TA ≤ +85°C)
扩展级	器件温度范围为扩展级 (-40°C ≤ TA ≤ +125°C)

30.2 绝对最大值

绝对最大参数值定义为器件的最恶劣工作条件。这些值并不代表器件的工作参数，如果运行条件超过所列值，可对器件造成损坏。这些参数并不总是独立存在的，即，参数值还需满足其它条件。

例如 “任一 I/O 引脚的最大拉 / 灌电流”。在某一时刻可以拉 / 灌电流的 I/O 引脚数取决于端口的最大拉 / 灌电流（或多个端口电流的总和），以及流入 VDD 引脚或流出 VSS 引脚的最大电流。出现本例这种情况的原因是电源 / 地与 I/O 端口和内部逻辑电路的总线宽度。如果超出该参数，在电源总线和接地总线上就可能发生电迁移。长时间的电迁移会导致这些总线开路（即断开与引脚的连接），从而使连接在这些总线上的逻辑电路停止工作。因此，超过该绝对参数值会导致器件的可靠性问题。

输入箝位电流定义为当引脚的电压超过规定值时，从二极管流向 VSS/VDD 的电流。

绝对最大参数值示例

偏置电压下的环境温度 ..... -40 至 +125°C

储存温度 ..... -40°C 至 +150°C

任一引脚 (除 VDD、$\overline{MCLR}$ 和 RA4 外) 相对于 VSS 的电压 ..... -0.3V 至 (VDD + 0.3V)

VDD 相对于 VSS 的电压 ..... -0.3 至 +7.5V

$\overline{MCLR}$ 引脚相对于 VSS **(2)** 的电压 ..... 0 至 +14V

RA4 引脚相对于 VSS 的电压 ..... 0 至 +14V

总功耗 **(1)** 1.0W

VSS 引脚的最大输出电流 ..... 300 mA

VDD 引脚的最大输入电流 ..... 250 mA

输入箝位电流，IIK (VI < 0 或 VI > VDD) ..... ± 20 mA

输出箝位电流，IOK (VO < 0 或 VO > VDD) ..... ± 20 mA

任一 I/O 引脚的最大灌电流 ..... 25 mA

任一 I/O 引脚的最大驱动电流 ..... 25 mA

PORTA、PORTB 和 PORTE 的最大灌电流总和 ..... 200 mA

PORTA、PORTB 和 PORTE 的最大驱动电流总和 ..... 200 mA

PORTC 和 PORTD 最大灌电流总和 ..... 200 mA

PORTC 和 PORTD 最大驱动电流总和 ..... 200 mA

PORTC 和 PORTD 最大灌电流总和 ..... 200 mA

PORTF 和 PORTG 最大驱动电流总和 ..... 100 mA

PORTF 和 PORTG 最大灌电流总和 ..... 100 mA

注 1: 功耗按如下公式计算：
$Pdis = V_{DD} \times \{I_{DD} - \sum I_{OH}\} + \sum \{(V_{DD} - V_{OH}) \times I_{OH}\} + \sum (V_{OL} \times I_{OL})$

注 2: 如果 $\overline{MCLR}$ 引脚的尖峰电压低于 VSS，感应出的电流超过 80 mA，可引起器件锁死。因此，如果要在 $\overline{MCLR}$ 引脚施加 “低电平”，应串联一个 50-100Ω 的电阻，而不是将该引脚直接拉至 VSS。

† 注：如果器件运行条件超过了上述 “绝对最大参数值”，即可能对器件造成永久性损坏。上述值为运行条件极大值，我们不建议器件运行在该规范范围以外。器件长时间在绝对最大参数条件下工作，其稳定性会受到影响。

30.3 器件选型表

数据手册中的这张表旨在帮助用户确定某种器件的适用振荡器，并提供了经过测试的参数值。用户在编程时可选择任何振荡器，但只有所列出的振荡器经过 Microchip 公司测试。

由于 RC 和 XT 振荡器的频率只达 4 MHz，因此它们只在 -04 (4 MHz) 器件上通过了测试。

频率在 10 MHz 或 20 MHz 的 PICmicro® 单片机只通过了 HS 模式的测试。在表 30-2 中，由于在 HS 振荡模式的电压范围中没有 IPD 测试点，因此该模式下的 IPD 是灰色的。表中给出的均为特性测试中的典型值。

电池应用通常要求器件工作在扩展电压范围。标有 **LC** 的器件具备扩展电压范围，其 RC、XT 和 LP 振荡器通过了测试。

窗口型器件是各种器件的超集，其振荡器在 -04，-20 和 LC 器件的所有规定范围通过了测试。器件通过测试的温度范围应视为商业级温度范围，尽管今后可对其工业级或扩展级温度范围进行测试。

表 30-2: 器件的振荡器配置规范与工作频率 (商业级器件) 对照示例

OSC	PIC16CXXX-04	PIC16CXXX-10	PIC16CXXX-20	PIC16LCXXX-04	带窗口器件
RC	VDD: 4.0V 至 6.0V IDD: 5.5V 时最大值 5 mA IPD: 4V 时最大值 6 µA Freq: 最大值 4 MHz	VDD: 4.5V 至 5.5V IDD: 5.5V 时典型值 2.7 mA IPD: 4V 时典型值 1.5 µA Freq: 最大值 4 MHz	VDD: 4.5V 至 5.5V IDD: 5.5V 时典型值 2.7 mA IPD: 4V 时典型值 1.5 µA Freq: 最大值 4 MHz	VDD: 2.5V 至 6.0V IDD: 3.0V 时最大值 3.8 mA IPD: 3V 时最大值 5 µA Freq: 最大值 4 MHz	VDD: 2.5V 至 6.0V IDD: 5.5V时最大值3.8mA IPD: 4V 时最大值 16 µA Freq: 最大值 4 MHz
XT	VDD: 4.0V 至 6.0V IDD: 5.5V 时最大值 5 mA IPD: 4V 时最大值 16 µA Freq: 最大值 4 MHz	VDD: 4.5V 至 5.5V IDD: 5.5V 时典型值 2.7 mA IPD: 4V 时典型值 1.5 µA Freq: 最大值 4 MHz	VDD: 4.5V 至 5.5V IDD: 5.5V时典型值2.7mA IPD: 4V 时典型值 1.5 µA Freq: 最大值 4 MHz	VDD: 2.5V 至 6.0V IDD: 3.0V 时最大值 3.8 mA IPD: 3V 时最大值 5 µA Freq: 最大值 4 MHz	VDD: 2.5V 至 6.0V IDD: 5.5V 时最大值 3.8 mA IPD: 4V 时最大值 16 µA Freq: 最大值 4 MHz
HS	VDD: 4.5V 至 5.5V IDD: 5.5V 时典型值 13.5 mA IPD: 4.5V 时典型值 1.5 µA Freq: 最大值 4 MHz	VDD: 4.5V 至 5.5V IDD: 5.5V时典型值 10 mA IPD: 1.5 µA typ. at 4.5V Freq: 最大值 10 MHz	VDD: 4.5V 至 5.5V IDD: 5.5V时最大值20 mA IPD: 4.5V时典型值1.5 µA Freq: 最大值 20 MHz	在 HS 模式下建议不使用	VDD: 4.5V 至 5.5V IDD: 5.5V时最大值20 mA IPD: 4.5V时典型值1.5 µA Freq: 最大值 20 MHz
LP	VDD: 4.0V 至 6.0V IDD: 32 kHz, 4.0V 时典型值 52.5 µA IPD: 4.0V时典型值0.9 µA Freq: 最大值 200 kHz	在 LP 模式下建议不使用	在 LP 模式下建议不使用	VDD: 2.5V 至 6.0V IDD: 32 kHz, 3.0V 时最大值 48 µA IPD: 3.0V 时最大值 5.0 µA Freq: 最大值 200 kHz	VDD: 2.5V 至 6.0V IDD: 32 kHz, 3.0V 时最大值 48 µA IPD: 3.0V时最大值5.0 µA Freq: 最大值 200 kHz

阴影部分表明所选振荡器只作了功能性测试，而未作 MIN/MAX 规范测试。
建议用户使用能满足器件规范要求的器件。

注： 标有工程样片 (ENG SMP) 的器件是针对当前工程测试项目的测试，不保证这些器件符合器件数据手册上的任一或全部技术要求。

30.4 器件电压规范

这些参数与器件的 VDD 电压、器件上电和器件的功能有关。

电源电压为器件正常运行所必须在器件上施加的电平。

Ram 数据保存电压为器件不丢失数据的电平。

VDD 启动电压是确保 POR 电路正常工作的 VDD 起始电平，用于确保内部上电复位信号的产生。

VDD 上升率是是使 POR 电路跳变所必需的最小 VDD 上升斜率，用于确保内部上电复位信号的产生。

欠压复位电压是欠压复位电路跳变的电压范围。当 BOR 电路跳变时，器件或者进入欠压复位，或刚刚从欠压复位中恢复。

表 30-3: 直流特性参数示例

直流特性			标准运行条件 (除非另外声明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级				
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
	VDD	**电源电压**					
D001		PIC16**C**XXX	4.0	—	6.0	V	XT, RC 和 LP 振荡模式
		PIC16**LC**XXX	2.5	—	6.0	V	
D001A		PIC16**C**XXX	4.5	—	5.5	V	HS 振荡模式
D002	VDR	**RAM 数据保存电压 (1)**	1.5	—	—	V	
D003	VPOR	保证内部上电复位信号的 **VDD 起始电压**	—	VSS	—	V	详见有关上电复位的章节
D004	SVDD	**保证内部上电复位信号的 VDD 上升率**	0.05	—	—	V/ms	详见有关上电复位的章节
	VBOR	**欠压复位电压**					
D005			3.7	4.0	4.3	V	配置字中 BODEN 位使能
D005A			3.7	4.0	4.4	V	仅限于扩展温度范围的器件

† 除非另外声明，否则，“典型值”一栏中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 这是在 SLEEP 模式下不丢失 RAM 数据的 VDD 电压下限。

30.5 器件电流特性

IDD 是器件工作时消耗的电流 (I)。测量时所有 I/O 引脚均作为输入，无论是高电平或低电平。即，没有悬空的输入引脚，也没有引脚驱动输出 （带负载）。

IPD 是器件在休眠模式下（ 断电时 ）消耗的电流 (I)，称为掉电电流。测量时所有 I/O 引脚均作为输入，无论是高电平或低电平。即，没有悬空的输入引脚，也没有引脚驱动输出 （带负载），弱上拉也被关闭。

器件在休眠模式时，一些模块也能正常工作。这些模块如下：

- 看门狗定时器 (WDT)
- 欠压复位 (BOR) 电路
- 定时器 Timer1
- 模数转换器
- LCD 模块
- 比较器
- 参考电压模块

禁止所有这些功能时，器件将消耗最小电流 (漏电流)。器件休眠时，任何模块的运行将使器件消耗更高的电流。最低功耗模式 （所有模块被禁止）与使能一个模块 （如 WDT）的电流消耗之差称为**模块差别电流** （**Module Differential Current**）。如果不止一个功能被使能，按照下面的方法我们可轻易计算出预期的电流：基本电流（休眠模式下禁止所有功能）加上所有模块的差别电流 (电流差)。例 30-1 给出了 5V 时使能 WDT 和定时器 Timer1 振荡器计算典型电流的例子。

例 30-1: IPD 的计算 （5V 时使能 WDT 和 Timer1 振荡器)

基本电流	14 nA	; 器件的漏电流
WDT 电流差	14 µA	; 14 µA - 14 nA = 14 µA
Timer1 电流差	22 µA	; 22 µA - 14 nA = 22 µA
休眠时的总电流	36 µA	;

表 30-4: 直流特性示例

直流特性			标准运行条件 (除非另外声明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级				
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
	IDD	**电源电流 (2,4,5)**					XT, RC 振荡器配置 (PIC16**C**XXX-**04**)
D010			—	2.7	5	mA	FOSC = 4 MHz, VDD = 5.5V
			—	2.0	3.8	mA	FOSC = 4 MHz, VDD = 3.0V
D010A			—	22.5	48	µA	LP 振荡器配置 FOSC = 32 kHz, VDD = 3.0V, WDT 禁止
D010C			—	7.7	5	mA	INTRC 振荡器配置 , Fosc = 4 MHz, VDD = 5.5V
D013			—	13.5	30	mA	HS 振荡器配置 (PIC16**C**XXX-**20**) Fosc = 20 MHz, VDD = 5.5V
	IPD	**掉电电流 (3,5)**					
D020			—	10.5	42	µA	VDD = 4.0V, WDT 使能 , -40°C 至 +85°C
			—	7.5	30	µA	VDD = 3.0V, WDT 使能 , -40°C 至 +85°C
			—	1.5	21	µA	VDD = 4.0V, WDT 禁止 , -0°C 至 +70°C
D021			—	0.9	13.5	µA	VDD = 3.0V, WDT 禁止 0°C 至 +70°C
			—	1.5	24	µA	VDD = 4.0V, WDT 禁止 , -40°C 至 +85°C
D021A			—	0.9	18	µA	VDD = 3.0V, WDT 禁止 , -40°C 至 +85°C
D021B			—	1.5	—	µA	VDD = 4.0V, WDT 禁止 , -40°C 至 +125°C

* 这些参数仅为特征值，未经测试。

† 除非另外声明，否则，“典型值”一栏中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 不适用

2: 电源电流主要受运行电压和频率的影响。其它因素，如 I/O 引脚的负载和开关频率、振荡器类型、内部代码执行模式和温度等也会影响电流消耗。
在有源运行模式下所有 IDD 测量值的测试的条件为：
OSC1 = 外部方波，满幅；所有 I/O 引脚均为三态，拉至 VDD 电平。
MCLR = VDD; WDT 按规定使能 / 禁止。

3: 休眠模式下的掉电电流并不取决于振荡器类型。掉电电流是在器件休眠时，所有 I/O 引脚处于高阻态并连接到 VDD 和 VSS 时测得的。

4: 器件振荡器配置为 RC 模式时，该电流不包括流经 Rext 的电流。流经该电阻的电流可根据以下公式估算：Ir = VDD/2Rext (mA)，其中 Rext 的单位为 kΩ。

5: 定时器 Timer1 振荡器 (使能时) 在参数上增加了约 20 µA 电流。该电流值仅为特征值，仅供设计参考，未经测试。

表 30-5: 直流特性示例

直流特性			标准运行条件 (除非另外声明) 运行温度				0°C ≤ TA ≤ +70°C 商业级 , -40°C≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级
参数编号	**符号**	**特性**	**最大值**	**典型值 †**	**最大值**	**单位**	**条件**
		模块差别电流 (5)					
D022	ΔIWDT	看门狗定时器	—	6.0	20	μA	VDD = 4.0V
			—	—	25	μA	-40°C 至 +125°C
D022A	ΔIBOR	欠压复位	—	350	425	μA	BODEN 位清零 , VDD = 5.0V
D023	ΔICOMP	比较器 (每个比较器)	—	85	100	μA	VDD = 4.0V
D023A	ΔIVREF	参考电压	—	94	300	μA	VDD = 4.0V
D024	ΔILCDRC	LCD 内部 RC 振荡器使能	—	6.0	20	μA	VDD = 3.0V
D024A	ΔILCDVG	LCD 电压发生	—	TBD	TBD	μA	VDD = 3.0V
D025	ΔIT1OSC	Timer1 振荡器	—	3.1	6.5	μA	VDD = 3.0V
D026	ΔIAD	A/D 转换器	—	1.0	—	μA	A/D 开启，没有进行转换
D027	ΔISAD	积分型 A/D (总电流)	—	165 *	250 *	μA	REFOFF = 0
D027A	ΔISADVR	积分型 A/D 带隙参考电压	—	20 *	30 *	μA	REFOFF = 0
D027B	ΔISADCDAC	积分型 A/D 可编程电流源	—	50 *	70 *	μA	ADCON1<7:4> = 1111b
D027C	ΔISADSREF	积分型 A/D 参考电压分压器	—	55 *	85 *	μA	ADOFF = 0
D027D	ΔISADCMP	积分型 A/D 比较器	—	40 *	65 *	μA	ADOFF = 0

† 除非另外声明，否则，“典型值”一栏中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

30.6 输入阈值电平

输入低电压 (VIL) 是被读作逻辑 '0' 的电平。高于输入低电压的输入电平可能不会被读作 '0'。由于器件与器件（或引脚间）的不同会导致该电平的差异，因此所有设计均应以具体规范为准。

输入高电平 (VIH) 是被读作逻辑 '1' 的电平。低于输入高电压的输入电平可能会被读作 '1'。由于器件与器件（或引脚间）的不同会导致该电平的差异，因此所有设计均应以具体规范为准。

带 TTL 逻辑的 I/O 引脚电平有两种规范。一种是工业标准 TTL 规范，其规定电压范围为 4.5V 到 5.5V。另一种是器件运行于整个电压范围内的规范。设计时可选用两者中要求较高的参数。

表 30-6: 直流特性示例

直流特性			标准运行条件 (除非另外声明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C f 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中表 30-3 中的说明。				
参数编号	**符号**	**特性**	**最大值**	**典型值 †**	**最大值**	**单位**	**条件**
	VIL	**输入低电压**					
		I/O 端口：					
D030		带 TTL 缓冲器	VSS	—	0.15VDD	V	整个 VDD 范围 (4)
D030A			—	—	0.8	V	4.5V ≤ VDD ≤ 5.5V (4)
D031		带施密特触发缓冲器	VSS	—	0.2VDD	V	整个 VDD 范围
D032		$\overline{MCLR}$, OSC1 (RC 模式)	VSS	—	0.2VDD	V	
D033		OSC1 (XT, HS 和 LP 模式)(1)	VSS	—	0.3VDD	V	
	VIH	**输入高电压**					
		I/O 端口					
D040		带 TTL 缓冲器	0.25VDD + 0.8V	—	VDD	V	整个 VDD 范围 (4)
D040A			2.0	—	VDD	V	4.5V ≤ VDD ≤ 5.5V (4)
D041		带施密特触发缓冲器	0.8VDD	—	VDD	V	整个 VDD 范围
D042		$\overline{MCLR}$	0.8VDD	—	VDD	V	
D042A		OSC1 (XT, HS 和 LP 模式)(1)	0.7VDD	—	VDD	V	
D043		OSC1 (RC 模式)	0.9VDD	—	VDD	V	
D050	VHYS	施密特触发器输入的滞后	TBD	—	—	V	

† 除非另外声明，否则，“典型值”一栏中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 在 RC 振荡器配置中，OSC1/CLKIN 引脚是一个施密特触发器输入。我们不建议在 RC 模式下采用外部时钟驱动 PICmicro 单片机。

2: 不适用。

3: 不适用。

4: 可选用两者中要求较高的参数。对于 VIL 应选用较高的电压，而对于 VIH 应选用较低的电压。

30.7 I/O 电流特性

PORT/GIO 的**弱上拉电流**是当弱上拉使能时器件的额外电流。

漏电流是器件在实际制造过程中因无法达到理想特征而消耗的电流。理想状况下输入端应没有电流存在，但在现实中总是存在寄生导电通路要消耗小到可忽略不计的电流。

表 30-7: 直流特性示例

直流特性			标准运行条件 (除非另外声明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中表 30-3 中的说明。				
参数编号	**符号**	**特性**	**最小值**	**典型值 †**	**最大值**	**单位**	**条件**
	IIL	**输入漏电流 (2,3)**					
D060		I/O 端口	—	—	±1	µA	VSS ≤ VPIN ≤ VDD, 高阻状态的引脚
D060A		CDAC	—	—	±1	µA	VSS ≤ VPIN ≤ VDD, 高阻状态的引脚
D061		$\overline{\text{MCLR}}$	—	—	±5	µA	VSS ≤ VPIN ≤ VDD
D063			—	—	±5	µA	VSS ≤ VPIN ≤ VDD, XT, HS 和 LP osc 模式
	IPU	**弱上拉电流**					
D070	IPURB	PORTB 弱上拉电流	50	250	400	µA	VDD = 5V, VPIN = VSS
D070A	IPUGIO	GIO 弱上拉电流	50	250	400	µA	VDD = 5V, VPIN = VSS
		可编程电流源 (积分型 A/D 器件)					CDAC 引脚 = 0V
D160		输出电流	18.75	33.75	48.75	µA	ADCON1<7:4> = 1111b (满量程)
D160A			1.25	2.25	3.25	µA	ADCON1<7:4> = 0001b (1 LSB)
D160B			-0.5	0	0.5	µA	ADCON1<7:4> = 0000b (零刻度)

† 除非另外声明，否则，“典型值”一栏中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 在 RC 振荡器配置中， OSC1/CLKIN 引脚是一个施密特触发器输入。我们不建议在 RC 模式下采用外部时钟驱动 PICmicro 单片机。

2: $\overline{\text{MCLR}}$ 引脚上的漏电流在很大程度上取决于施加在该引脚上的电平。规定电平为正常运行条件下的电平。在不同的输入电压下可测得更大的漏电流。

3: 负电流定义为引脚的拉电流。

30.8 输出驱动电压

I/O 引脚的**输出低电压** (VOL) 取决于该 I/O 引脚的外部连接。如果 I/O 引脚短接到 VDD，无论 I/O 引脚的驱动能力如何，都将不能输出低电平 (并且器件会消耗过量驱动电流)。VOL 是 I/O 引脚驱动的输出电压，前提是 I/O 的灌电流低于电器规范中条件部分规定的 IOL 电流 (在规定的器件电压下)。

I/O 引脚的**输出高电压** (VOH) 取决于 该 I/O 引脚的外部连接。如果 I/O 引脚短接到 VSS，无论 I/O 引脚的驱动能力如何，都将不能输出高电平 (并且器件会消耗过量驱动电流)。VOH 是 I/O 引脚驱动的输出电压，前提是 I/O 口的拉电流低于电器规范中条件部分规定的 IOH 电流 (在额定的器件电压下)。

表 30-8: 直流特性示例

直流特性			标准运行条件 (除非另外声明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中表 30-3 中的说明。				
参数编号	**符号**	**特性**	**最小值**	**典型值 †**	**最大值**	**单位**	**条件**
	VOL	**输出低电压**					
D080		I/O 端口	—	—	0.6	V	IOL = 8.5 mA, VDD = 4.5V, -40°C 至 +85°C
D080A			—	—	0.6	V	IOL = 7.0 mA, VDD = 4.5V, -40°C 至 +125°C
D083		OSC2/CLKOUT (RC 模式)	—	—	0.6	V	IOL = 1.6 mA, VDD = 4.5V, -40°C 至 +85°C
D083A			—	—	0.6	V	IOL = 1.2 mA, VDD = 4.5V, -40°C 至 +125°C
	VOH	**输出高电压 (3)**					
D090		I/O 端口	VDD - 0.7	—	—	V	IOH = -3.0 mA, VDD = 4.5V, -40°C 至 +85°C
D090A			VDD - 0.7	—	—	V	IOH = -2.5 mA, VDD = 4.5V, -40°C 至 +125°C
D092		OSC2/CLKOUT (RC 模式)	VDD - 0.7	—	—	V	IOH = -1.3 mA, VDD = 4.5V, -40°C 至 +85°C
D092A			VDD - 0.7	—	—	V	IOH = -1.0 mA, VDD = 4.5V, -40°C 至 +125°C
D150	VOD	**开漏高电压**	—	—	12	V	RA4 引脚
		可编程电流源					
D170	VPCS	输出电压范围	Vss	—	VDD – 1.4	V	CDAC 引脚
D171	SNPCS	输出电压灵敏度	– 0.1	–0.01	—	%/V	Vss ≤ VCDAC ≤ VDD – 1.4
		带隙参考					
D180	VBGR	输出电压范围	1.14	1.19	1.24	V	在 AN0 引脚上 AMUXOE =1 且 ADCS3:ADSC0 = 0100b 时

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 在 RC 振荡器配置中， OSC1/CLKIN 引脚是一个施密特触发器输入。我们不建议在 RC 模式下采用外部时钟驱动 PICmicro 单片机。

2: $\overline{\text{MCLR}}$ 引脚上的漏电流在很大程度上取决于施加在该引脚上的电平。规定电平为正常运行条件下的电平。在不同的输入电压下可测得更大的漏电流。

3: 负电流定义为引脚的拉电流。

30.9 I/O 引脚的容性负载

这些特性说明了 I/O 引脚在器件测试平台上测试时需要满足的条件。这些负载影响了器件电气规范的时间特性。如果具体应用中的负载不同，就要确定它们是如何影响器件特性的。低于特性说明中的值的电容将不会对系统产生影响。

表 30-9: 直流特性

直流特性			标准运行条件 (除非另外声明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中表 30-3 中的说明。				
参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
		输出引脚上的容性负载规范					
D100	COSC2	OSC2 引脚	—	—	15	pF	当使用外部时钟驱动 OSC1 时的 XT、 HS 和 LP 模式下。
D101	CIO	所有 I/O 引脚和 OSC2 (RC 模式)	—	—	50	pF	满足器件的时序特性
D102	CB	SCL, SDA	—	—	400	pF	在 I2C 模式下

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 在 RC 振荡器配置中， OSC1/CLKIN 引脚是一个施密特触发器输入。我们不建议在 RC 模式下采用外部时钟驱动 PICmicro 单片机。

2: MCLR 引脚上的漏电流在很大程度上取决于施加在该引脚上的电平。规定电平为正常运行条件下的电平。在不同的输入电压下可测得更大的漏电流。

3: 负电流定义为引脚的拉电流。

30.10 数据 EEPROM / 闪存

表 30-10: 数据 EEPROM / 闪存特性示例

直流特性			标准运行条件 (除非另外说明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中表 30-3 中的说明。				
参数编号	**符号**	**特性**	**最小值**	**典型值 †**	**最大值**	**单位**	**条件**
		数据 EEPROM 存储器					
D120	ED	耐久度	1M	10M	—	E/W	25°C at 5V
D121	VDRW	用于读写的 VDD	VMIN	—	6.0	V	VMIN = 最小运行电压
D122	TDEW	擦 / 写周期时间	—	—	10	ms	
		闪存程序存储器					
D130	EP	耐久度	100	1000	—	E/W	
D131	VPR	用于读入的 VDD	VMIN	—	6.0	V	VMIN = 最小运行电压
D132	VPEW	用于擦 / 写的 VDD	4.5	—	5.5	V	
D133	TPEW	擦 / 写周期时间	—	—	10	ms	

† 除非另外声明，否则，“典型值”一列中的数据为 5V、25°C 条件下的值。这些参数仅供设计参考，未经测试。

30.11 LCD

表 30-11: LCD 模块电气特性示例

直流特性			标准运行条件 (除非另外说明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中表 30-3 中的说明。				
参数编号	**符号**	**特性**	**最小值**	**典型值 †**	**最大值**	**单位**	**条件**
D200	VLCD3	VLCD3 引脚上的 LCD 电压	VDD - 0.3	—	Vss + 7.0	V	
D201	VLCD2	VLCD2引脚上的LCD电压	—	—	VLCD3	V	
D202	VLCD1	VLCD1引脚上的LCD电压	—	—	VDD	V	
D210	RCOM	Com 输出源阻抗	—	—	1k	Ω	COM 输出
D211	RSEG	Seg 输出源阻抗	—	—	10k	Ω	SEG 输出
D220	VOH	输出高电压	Max (VLCDN) - 0.1	—	Max (VLCDN)	V	COM 输出 IOH = 25 µA SEG 输出 IOH = 3 µA
D221	VOL	输出低电压	Min (VLCDN)	—	Min (VLCDN) + 0.1	V	COM 输出 IOL = 25 µA SEG 输出 IOL = 3 µA

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: VLCD 引脚上的源阻抗为 0Ω。

表 30-12: VLCD 电荷泵电气特性示例

直流特性			标准运行条件 (除非另外说明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中的表 30-3 中的说明。				
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**条件**
D250	IVADJ	VLCDADJ 稳定电流输出	—	10	—	µA	
D251	Ivr	VLCDADJ 电流消耗	—	—	20	µA	
D252	Δ IVADJ / Δ VDD	VLCDADJ 电流 VDD 抑制	—	—	0.1/1	µA/V	
D253	Δ IVADJ / Δ T	VLCDADJ 温度变化的电流	—	—	0.1/70	µA/°C	
D260(1)	RVADJ	VLCDADJ 外接电阻	100	—	230	kΩ	
D265	VVADJ	VLCDADJ 电压极限	1.0	—	2.3	V	
D271(1)	CECPC	外部电荷泵电容	—	0.5	—	µF	

注 1: 仅供设计参考。

30.12 比较器和参考电压

表 30-13: 比较器特性示例

直流特性				标准运行条件 (除非另外说明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中表 30-3 中的说明。				
参数编号	**符号**	**特性**		**最小值**	**典型值**	**最大值**	**单位**	**备注**
D300	VIOFF	输入偏置电压		—	± 5.0	± 10	mV	
D301	VICM	共模输入电压		0	—	VDD - 1.5	V	
D302	CMRR	共模抑制比		35	70	—	db	
300	TRESP	响应时间 (1)	PIC16**C**XXX	—	150	400	ns	
300A			PIC16**LC**XXX	—	210	600	ns	
301	TMC2OV	比较器模式变化到输出有效模式		—	—	10	µs	

注 1: 当一个比较器输入电压为 (VDD - 1.5)/2，而另一个比较器输入从 VSS 跳变到 VDD 时所测得的响应时间。

表 30-14: 参考电压特性

直流特性			标准运行条件 (除非另外说明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中表 30-3 中的说明。				
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**备注**
D310	VRES	分辨率	VDD/32	—	VDD/24	V	
D311	VRAA	绝对精度	— —	— —	1/4 1/2	LSb LSb	低精度 (VRR = 1) 高精度 (VRR = 0)
D312	VRUR	单位电阻 (R)	—	2k	—	Ω	
310	TSET	稳定时间 (1)	—	—	10	µs	

注 1: 当 VRR = 1， VR3:VR0 从 0000 跳变到 1111 时所测得的稳定时间。

30.13 时序参数符号

时序参数符号是根据以下格式之一创建的：

1. TppS2ppS
2. TppS
3. TCC:ST (仅适用于 I²C 规范)
4. Ts (仅适用于 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	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	数据输入	t0	T0CKI
io	I/O 端口	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

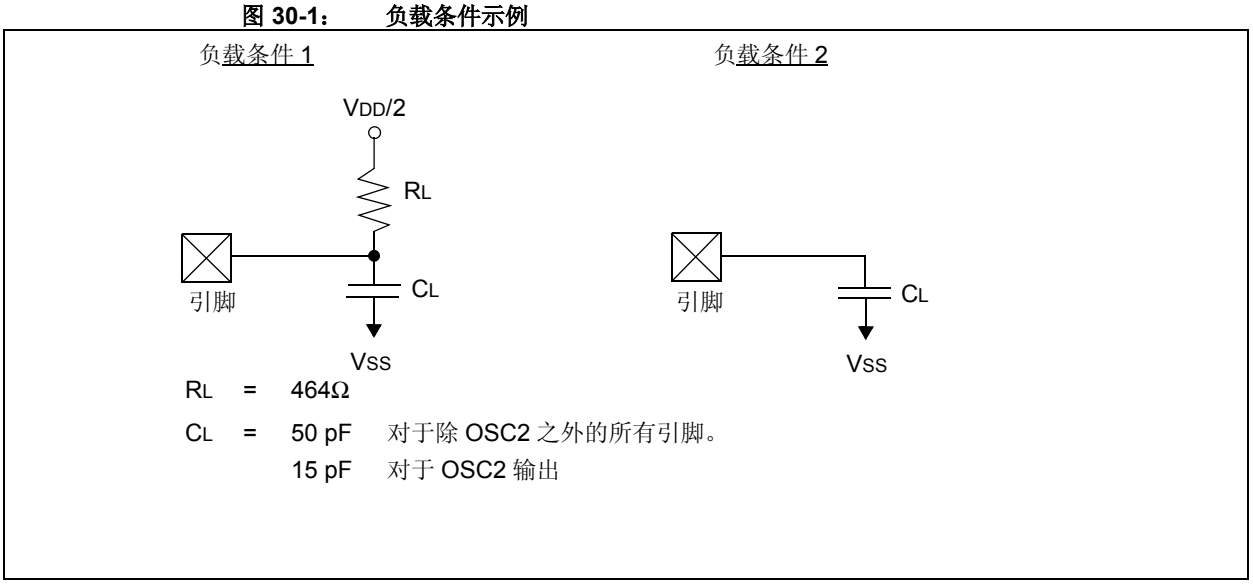
大写字母及其含义：

S			
F	下降	P	周期
H	高	R	上升
I	无效 (高阻态)	V	有效
L	低	Z	高阻态
仅 I²C			
AA	输出通道	High	高
BUF	总线空闲	Low	低

TCC:ST (仅适用于 I²C 规范)

CC			
HD	保持	SU	启动
ST			
DAT	DATA 输入保持	STO	STOP 条件
STA	START 条件		

图 30-1： 负载条件示例



RL = 464Ω

CL = 50 pF 对于除 OSC2 之外的所有引脚。

15 pF 对于 OSC2 输出

30.14 外部时钟时序波形图和时序要求示例

图 30-2： 外部时钟时序波形图示例

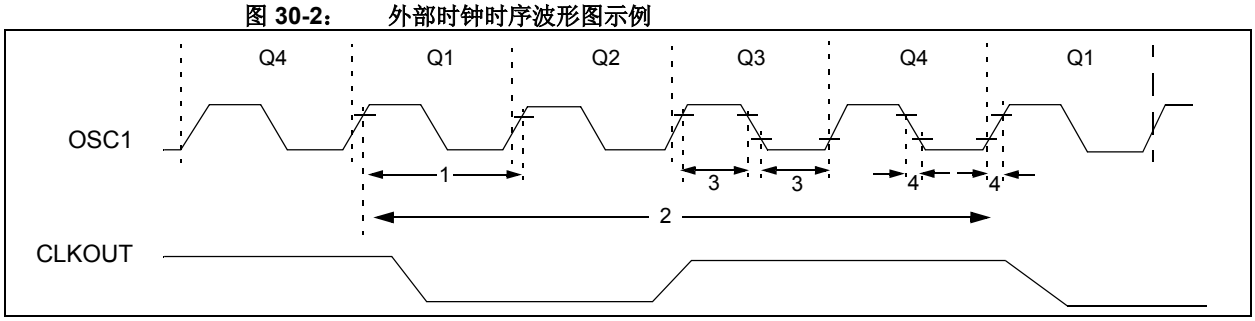


表 30-15: 外部时钟时序要求示例

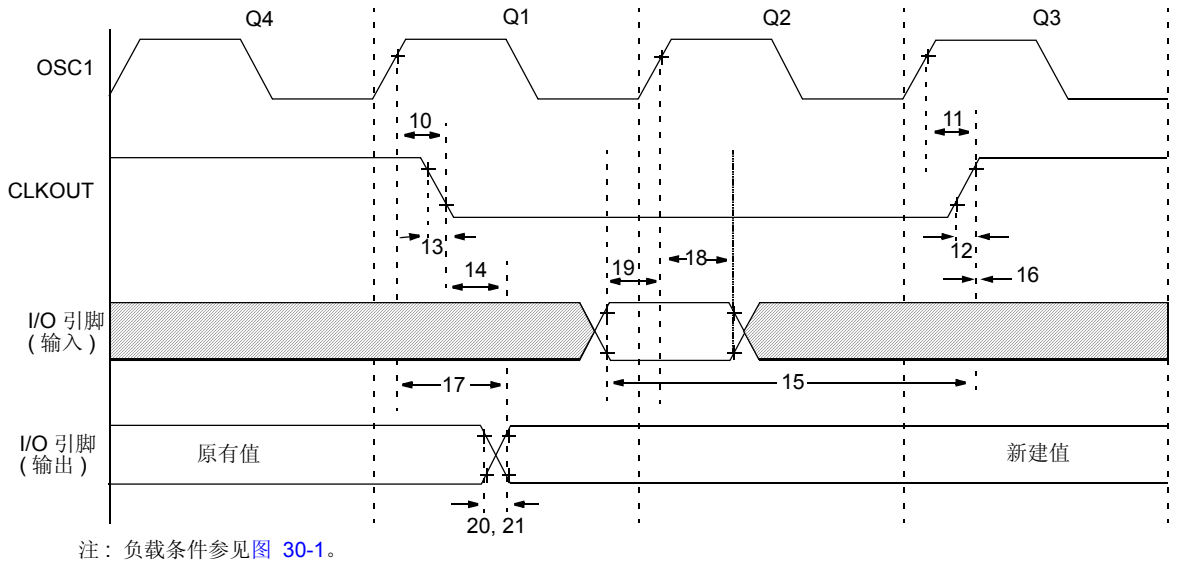
参数编号	符号	特性	最小值	典型值†	最大值	单位	条件	
1A	Fosc	外部时钟输入频率 (1)	DC	—	4	MHz	XT 和 RC 振荡模式	
								PIC16**C**XXX-**04**
								PIC16**LC**XXX-**04**
			DC	—	10	MHz	HS 振荡模式	PIC16**C**XXX-**10**
			DC	—	20	MHz		PIC16**C**XXX-**20**
			DC	—	200	kHz	LP 振荡模式	PIC16**LC**XXX-**04**
		振荡器频率 (1)	DC	—	4	MHz	RC 振荡模式	PIC16**C**XXX-**04**
								PIC16**LC**XXX-**04**
			0.1	—	4	MHz	XT 振荡模式	PIC16**C**XXX-**04**
								PIC16**LC**XXX-**04**
			4	—	10	MHz	HS 振荡模式	PIC16**C**XXX-**10**
			4	—	20	MHz		PIC16**C**XXX-**20**
			5	—	200	kHz	LP 振荡模式	PIC16**LC**XXX-**04**
1	Tosc	外部时钟输入周期 (1)	250	—	—	ns	XT 和 RC 振荡模式	
								PIC16**C**XXX-**04**
								PIC16**LC**XXX-**04**
			100	—	—	ns	HS 振荡模式	PIC16**C**XXX-**10**
			50	—	—	ns		PIC16**C**XXX-**20**
			5	—	—	µs	LP 振荡模式	PIC16**LC**XXX-**04**
		振荡器周期 (1)	250	—	—	ns	RC 振荡模式	PIC16**C**XXX-**04**
								PIC16**LC**XXX-**04**
			250	—	10,000	ns	XT 振荡模式	PIC16**C**XXX-**04**
								PIC16**LC**XXX-**04**
			100	—	250	ns	HS 振荡模式	PIC16**C**XXX-**10**
			50	—	250	ns		PIC16**C**XXX-**20**
			5	—	—	µs	LP 振荡模式	PIC16**LC**XXX-**04**
2	TCY	指令周期时间 (1)	200	—	DC	ns	TCY = 4/FOSC	
3	TosL, TosH	外部时钟输入 (OSC1) 的高电平或低电平时间	50	—	—	ns	XT 振荡模式	PIC16**C**XXX-**04**
			60	—	—	ns	XT 振荡模式	PIC16**LC**XXX-**04**
			2.5	—	—	µs	LP 振荡模式	PIC16**LC**XXX-**04**
			15	—	—	ns	HS 振荡模式	PIC16**C**XXX-**20**
4	TosR, TosF	外部时钟 (OSC1) 上升时间	—	—	25	ns	XT 振荡模式	PIC16**C**XXX-**04**
		外部时钟 (OSC1) 下降时间	—	—	50	ns	LP 振荡模式	PIC16**LC**XXX-**04**
			—	—	15	ns	HS 振荡模式	PIC16**C**XXX-**20**

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 指令周期 (TCY) 等于输入振荡器时基周期的 4 倍。所有值均基于标准运行条件下，器件执行代码时对应特定振荡器类型的特征数据。超过规定值可导致振荡器运行不稳定，并 / 或使电流消耗超过预期。所有器件在测试 “最小”值时，均在 OSC1/CLKIN 引脚接入了外部时钟。

当使用外部时钟输入时，所有器件的 “最大”周期时间极限为 “DC”(无时钟)。

图 30-3： 时钟输出 （CLKOUT）和 I/O 时序波形图示例



注： 负载条件参见图 30-1。

表 30-16: 时钟输出 CLKOUT 和 I/O 时序要求示例

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↓		—	75	200	ns	(1)
11	TosH2ckH	OSC1 到 CLKOUT↑		—	75	200	ns	(1)
12	TckR	CLKOUT 上升时间		—	35	100	ns	(1)
13	TckF	CLKOUT 下降时间		—	35	100	ns	(1)
14	TckL2ioV	CLKOUT ↓ 到端口输出有效		—	—	0.5TCY + 20	ns	(1)
15	TioV2ckH	在 CLKOUT ↑ 前端口输入有效		0.25TCY + 25	—	—	ns	(1)
16	TckH2ioI	在 CLKOUT ↑ 后端口输入保持		0	—	—	ns	(1)
17	TosH2ioV	OSC1↑ (Q1 周期) 到端口输出有效		—	50	150	ns	
18	TosH2ioI	OSC1↑ (Q2 周期) 到端口输入无效 (I/O 输入保持时间)	PIC16**C**XXX	100	—	—	ns	
18A			PIC16**LC**XXX	200	—	—	ns	
19	TioV2osH	端口输入有效到 OSC1↑ (I/O 输入启动时间)		0	—	—	ns	
20	TioR	端口输出上升时间	PIC16**C**XXX	—	10	25	ns	
20A			PIC16**LC**XXX	—	—	60	ns	
21	TioF	端口输出下降时间	PIC16**C**XXX	—	10	25	ns	
21A			PIC16**LC**XXX	—	—	60	ns	
22††	Tinp	INT 引脚高或低电平时间		TCY	—	—	ns	
23††	Trbp	RB7:RB4 改变 INT 高或低电平时间		TCY	—	—	ns	
24††	Trcp	RC7:RC4 改变 INT 高或低电平时间		20			ns	

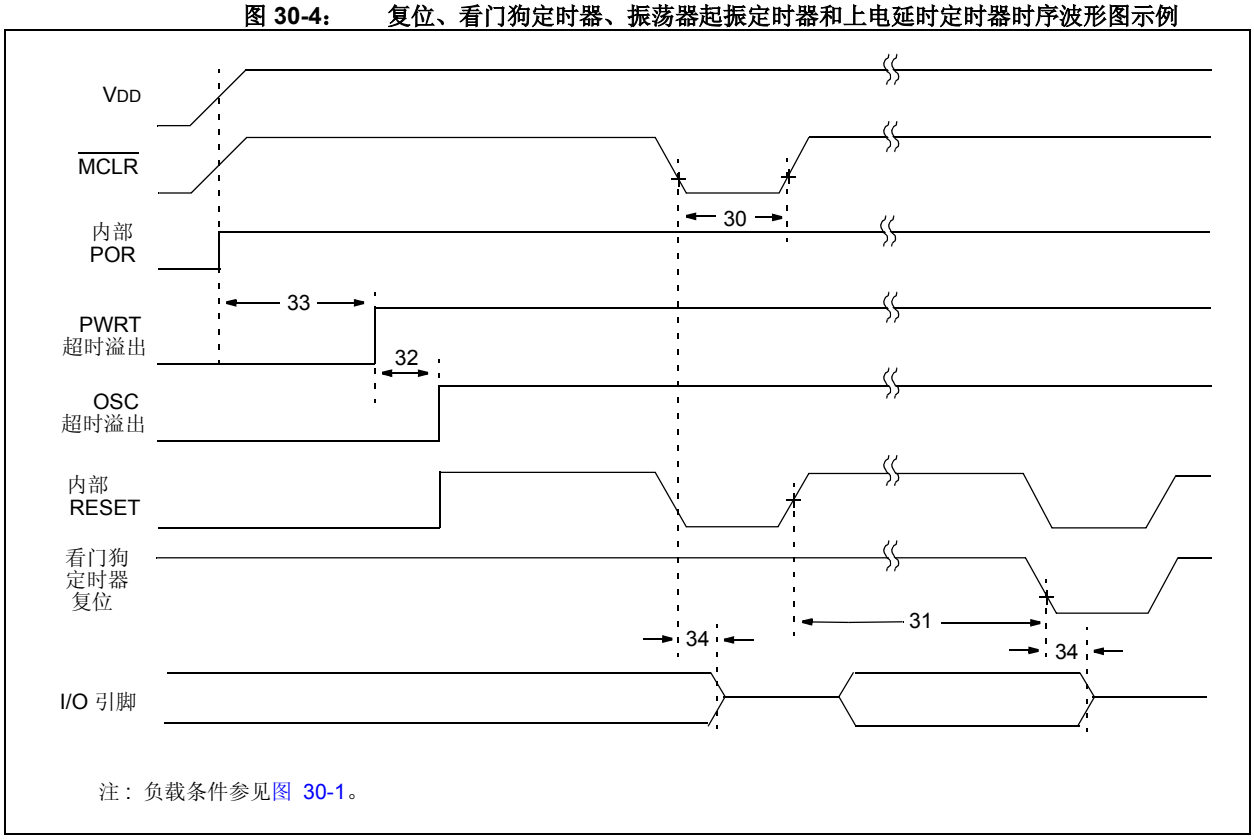
† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

†† 这些参数为异步事件，与任何内部时钟边沿无关。

注 1: 测量是在 RC 模式下进行的，此时， CLKOUT 输出为 4 x TOSC。

30.15 上电和复位时序波形图及要求示例

图 30-4： 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波形图示例



注： 负载条件参见图 30-1。

图 30-5： 欠压复位时序图

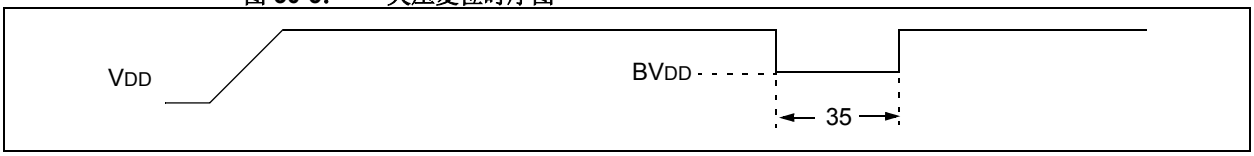


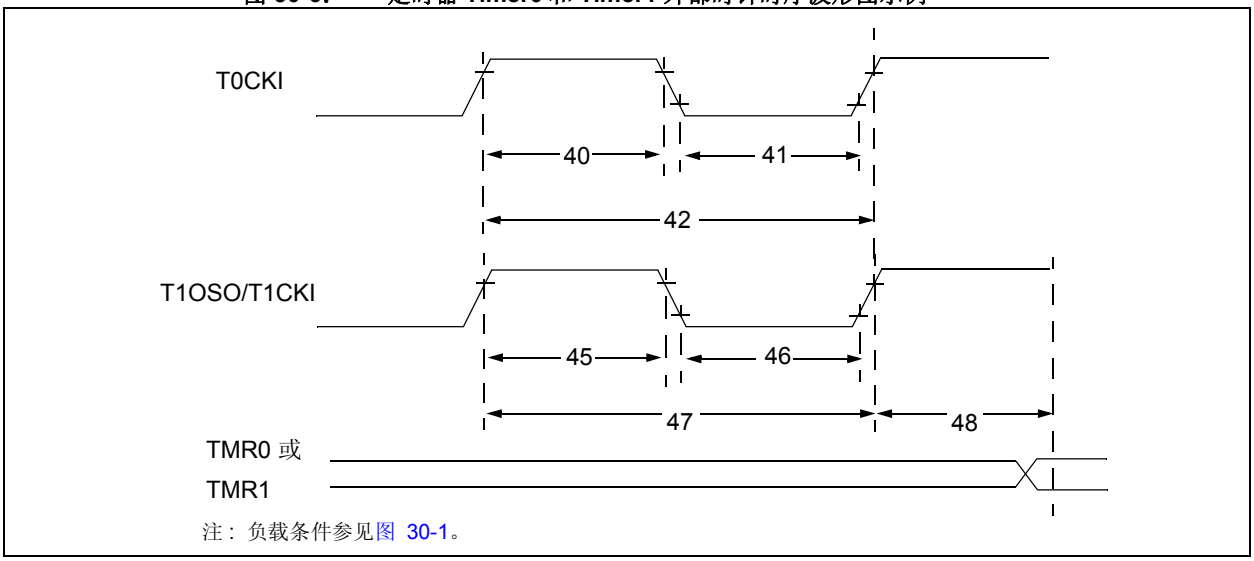
表 30-17: 复位、看门狗定时器、振荡器起振定时器、上电延时定时器和欠压复位要求示例

参数编号	符号	特性	最小值	典型值†	最大值	单位	条件
30	TmcL	MCLR 脉冲宽度 (低)	2	—	—	µs	VDD = 5V, -40°C 至 +125°C
31	Twdt	看门狗定时器超时溢出周期 (无预分频器)	7	18	33	ms	VDD = 5V, -40°C 至 +125°C
32	Tost	振荡器起振定时器周期	—	1024TOSC	—	—	TOSC = OSC1 周期
33	Tpwrt	上电延时定时器周期	28	72	132	ms	VDD = 5V, -40°C 至 +125°C
34	TIOZ	自 MCLR 或看门狗定时器复位起 I/O 处于高阻态的时间	—	—	2.1	µs	
35	TBOR	欠压复位脉冲宽度	100	—	—	µs	VDD ≤ BVDD (见 D005)

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

30.16 定时器 Timer0 和 Timer1 时序波形图及要求示例

图 30-6： 定时器 Timer0 和 Timer1 外部时钟时序波形图示例



注： 负载条件参见图 30-1。

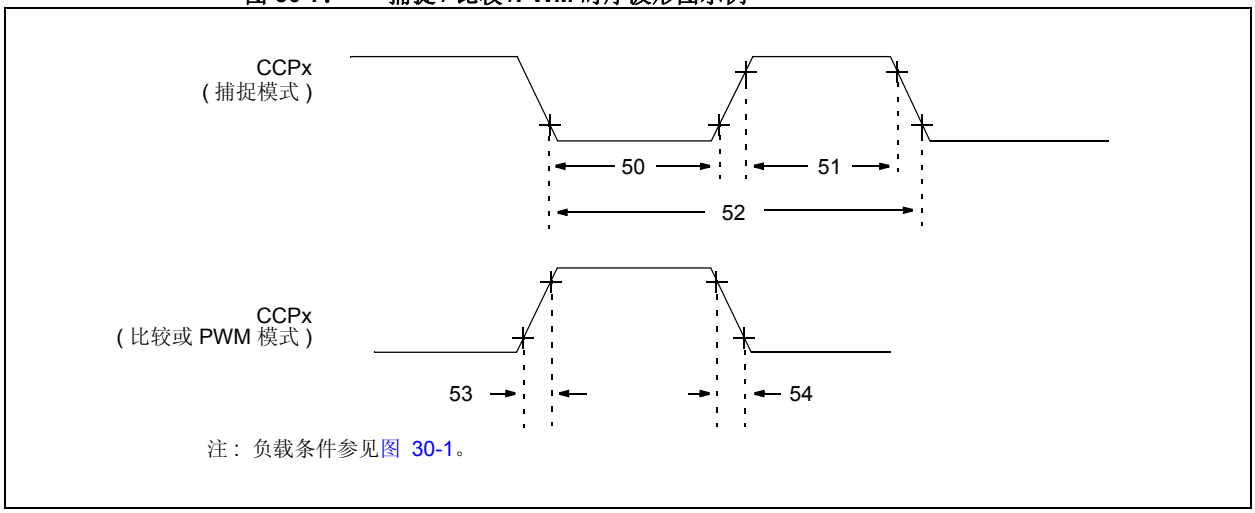
表 30-18： 定时器 Timer0 和 Timer1 外部时钟要求示例

参数编号	符号	特性			最小值	典型值†	最大值	单位	条件
40	Tt0H	T0CKI 高电平脉冲宽度		无预分频器	0.5TCY + 20	—	—	ns	
				有预分频器	10	—	—	ns	
41	Tt0L	T0CKI 低电平脉冲宽度		无预分频器	0.5TCY + 20	—	—	ns	
				有预分频器	10	—	—	ns	
42	Tt0P	T0CKI 周期			取最大值： 20 µS 或 TCY + 40 N	—	—	ns	N = 预分频比 (1, 2, 4,..., 256)
45	Tt1H	T1CKI 高电平时间	同步，无预分频器		0.5TCY + 20	—	—	ns	
			同步，有预分频器	PIC16**C**XXX	15	—	—	ns	
				PIC16**LC**XXX	25	—	—	ns	
			异步	PIC16**C**XXX	30	—	—	ns	
				PIC16**LC**XXX	50	—	—	ns	
46	Tt1L	T1CKI 低电平时间	同步，无预分频器		0.5TCY + 20	—	—	ns	
			同步，有预分频器	PIC16**C**XXX	15	—	—	ns	
				PIC16**LC**XXX	25	—	—	ns	
			异步	PIC16**C**XXX	2TCY	—	—	ns	
				PIC16**LC**XXX					
47	Tt1P	T1CKI 输入周期	同步		取最大值： 20 µS 或 TCY + 40 N	—	—	ns	N = 预分频比 (1, 2, 4, 8)
			异步		取最大值： 20µS 或 4TCY	—	—	ns	
	Ft1	定时器 Timer1 振荡器输入频率范围 (T1OSCEN 位置 1 使能振荡器)			DC	—	200	kHz	
48	Tcke2tmrl	从外部时钟边沿到定时器增 1 的延时			2Tosc	—	7Tosc	—	

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

30.17 CCP 的时序图及要求

图 30-7： 捕捉 / 比较 /PWM 时序波形图示例



注： 负载条件参见图 30-1。

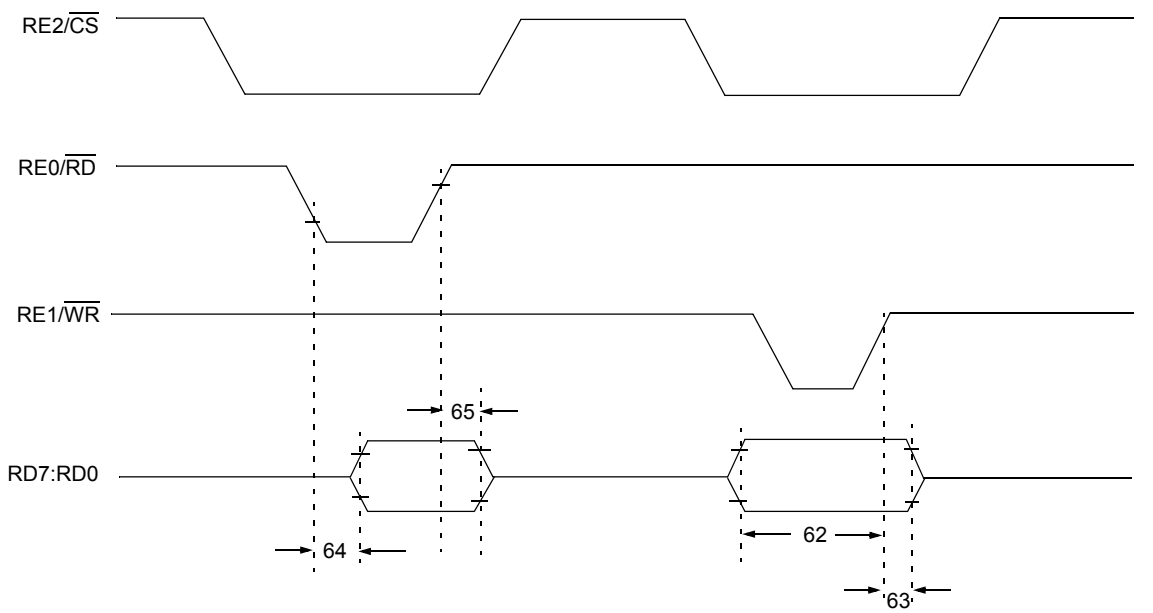
表 30-19： 捕捉 / 比较 /PWM 要求示例

参数编号	符号	特性			最小值	典型值 †	最大值	单位	条件
50	TccL	CCPx 输入低电平时间	无预分频器		0.5TCY + 20	—	—	ns	
			有预分频器	PIC16**C**XXX	10	—	—	ns	
				PIC16**LC**XXX	20	—	—	ns	
51	TccH	CCPx 输入高电平	无预分频器		0.5TCY + 20	—	—	ns	
			有预分频器	PIC16**C**XXX	10	—	—	ns	
				PIC16**LC**XXX	20	—	—	ns	
52	TccP	CCPx 输入周期			3TCY + 40 N	—	—	ns	N = 预分频比 (1,4 或 16)
53	TccR	CCPx 输出上升时间		PIC16**C**XXX	—	10	25	ns	
				PIC16**LC**XXX	—	25	45	ns	
54	TccF	CCPx 输出下降时间		PIC16**C**XXX	—	10	25	ns	
				PIC16**LC**XXX	—	25	45	ns	

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

30.18 并行从动端口 (PSP) 时序图及要求

图 30-8： 并行从动端口时序波形图示例



注： 负载条件参见图 30-1。

表 30-20: 并行从动端口要求

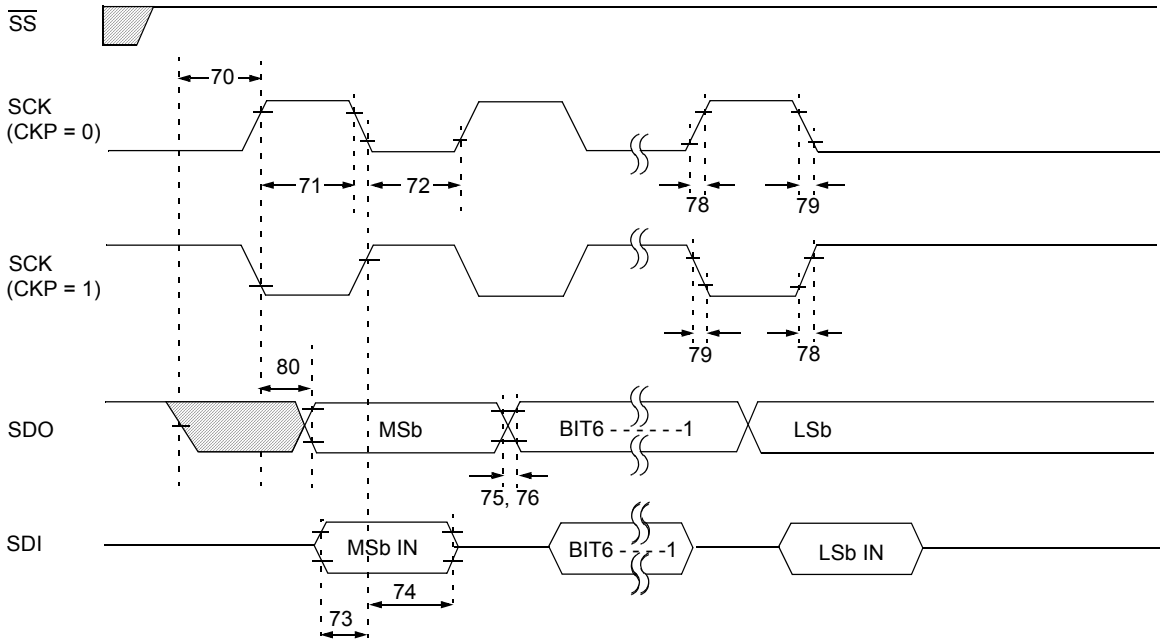
参数编号	符号	特性		最小值	典型值†	最大值	单位	条件
62	TdtV2wrH	在 WR↑ 或 CS↑ 前数据输入有效时间 (建立时间)		20	—	—	ns	
63	TwrH2dtI	WR↑ 或 CS↑ 到数据输入无效时间 (保持时间)	PIC16**C**XXX	20	—	—	ns	
			PIC16**LC**XXX	35	—	—	ns	
64	TrdL2dtV	RD↓ 和 CS↓ 到数据输出有效		—	—	80	ns	
65	TrdH2dtI	RD↑ 或 CS↓ 到数据输出无效		10	—	30	ns	
66	TibfINH	禁止 IBF 标志位被 WR↑ 或 CS↑ 清零		—	—	3Tcy§		

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

§ 该参数须经设计验证。

30.19 SSP 和 MSSP SPI 模式时序波形图及要求示例

图 30-9： SPI 主控模式时序图示例 (CKE = 0)



注：负载条件参见图 30-1。

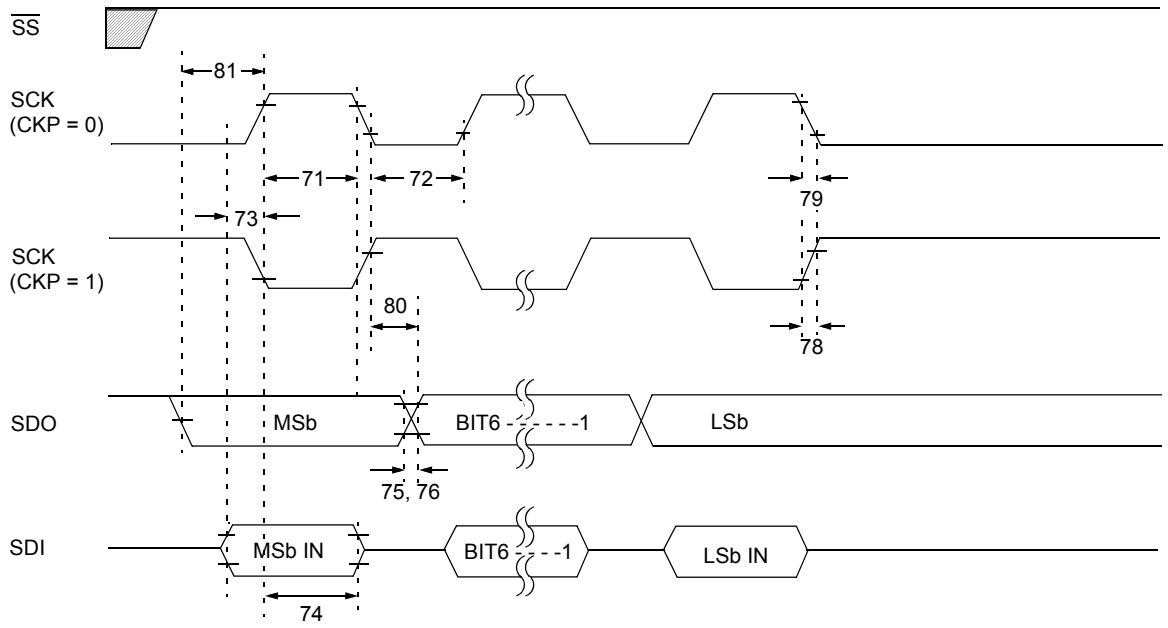
表 30-21： SPI 主控模式要求示例 (CKE = 0)

参数编号	符号	特性		最小值	典型值	最大值	单位	条件
70	TssL2scH, TssL2scL	SS↓ 到 SCK↓ 或 SCK↑ 输入		TCY	—	—	ns	
71	TscH	SCK 输入高电平时间	连续	1.25TCY + 30	—	—	ns	
71A		(从动模式)	单字节	40	—	—	ns	注 1
72	TscL	SCK 输入低电平时间	连续	1.25TCY + 30	—	—	ns	
72A		(从动模式)	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	Byte1 的末个时钟边沿到 Byte2 的首个时钟边沿		1.5TCY + 40	—	—	ns	注 1
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	PIC16**C**XXX	—	10	25	ns	
			PIC16**LC**XXX	—	20	45	ns	
76	TdoF	SDO 数据输出下降时间		—	10	25	ns	
78	TscR	SCK 输出上升时间	PIC16**C**XXX	—	10	25	ns	
		(主控模式)	PIC16**LC**XXX	—	20	45	ns	
79	TscF	SCK 输出下降时间 (主控模式)		—	10	25	ns	
80	TscH2doV, TscL2doV	SCK 边沿后 SDO 数据输出有效	PIC16**C**XXX	—	—	50	ns	
			PIC16**LC**XXX	—	—	100	ns	

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 只有在使用参数 71A 和 72A 时，才需要参数 73A。

图 30-10： SPI 主控模式时序图示例 (CKE = 1)



注：负载条件参见图 30-1。

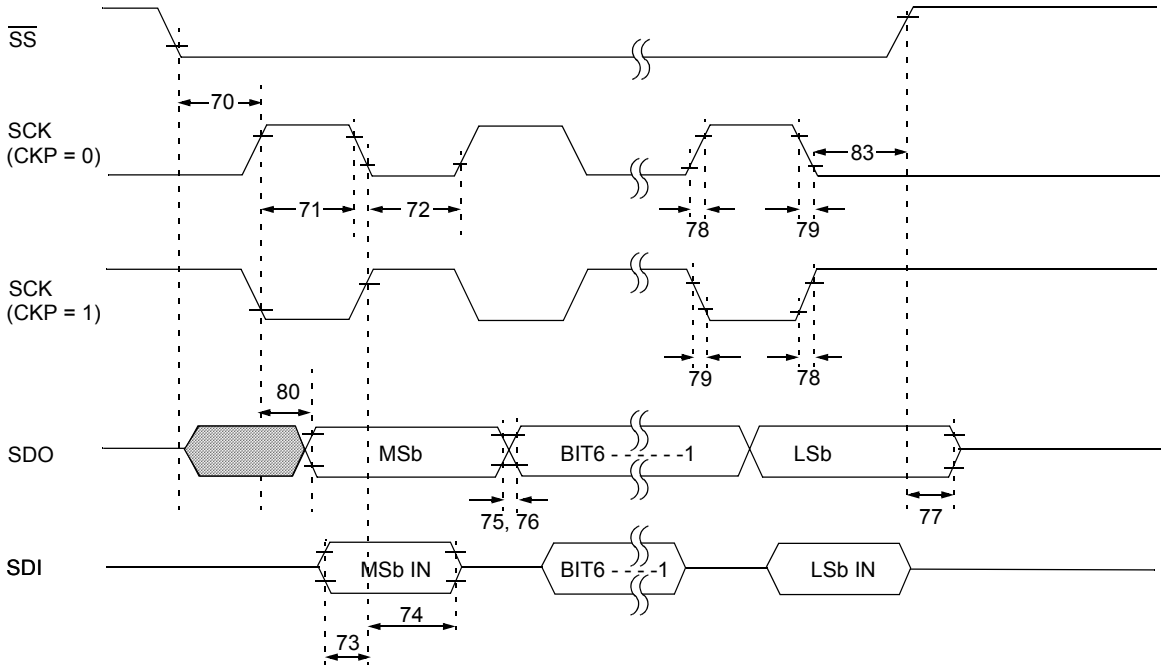
表 30-22: SPI™ 主控模式要求示例 (CKE = 1)

参数编号	符号	特性		最小值	典型值†	最大值	单位	条件
71	TscH	SCK 输入高电平时间	连续	1.25TCY + 30	—	—	ns	
71A		(从动模式)	单字节	40	—	—	ns	注 1
72	TscL	SCK 输入低电平时间	Continuous	1.25TCY + 30	—	—	ns	
72A		(从动模式)	Single Byte	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	Byte1的末个时钟边沿到Byte2的首个时钟边沿		1.5TCY + 40	—	—	ns	注 1
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	PIC16**C**XXX	—	10	25	ns	
			PIC16**LC**XXX		20	45	ns	
76	TdoF	SDO 数据输出下降时间		—	10	25	ns	
78	TscR	SCK 输出上升时间 (主控模式)	PIC16**C**XXX	—	10	25	ns	
			PIC16**LC**XXX		20	45	ns	
79	TscF	SCK 输出下降时间 (主控模式)		—	10	25	ns	
80	TscH2doV, TscL2doV	SCK 边沿后 SDO 数据输出有效	PIC16**C**XXX	—	—	50	ns	
			PIC16**LC**XXX		—	100	ns	
81	TdoV2scH, TdoV2scL	SDO 数据输出建立到 SCK 边沿		TCY	—	—	ns	

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 只有在使用参数 71A 和 72A 时，才需要参数 73A。

图 30-11： SPI™ 从动模式时序图示例 (CKE = 0)



注： 负载条件参见图 30-1。

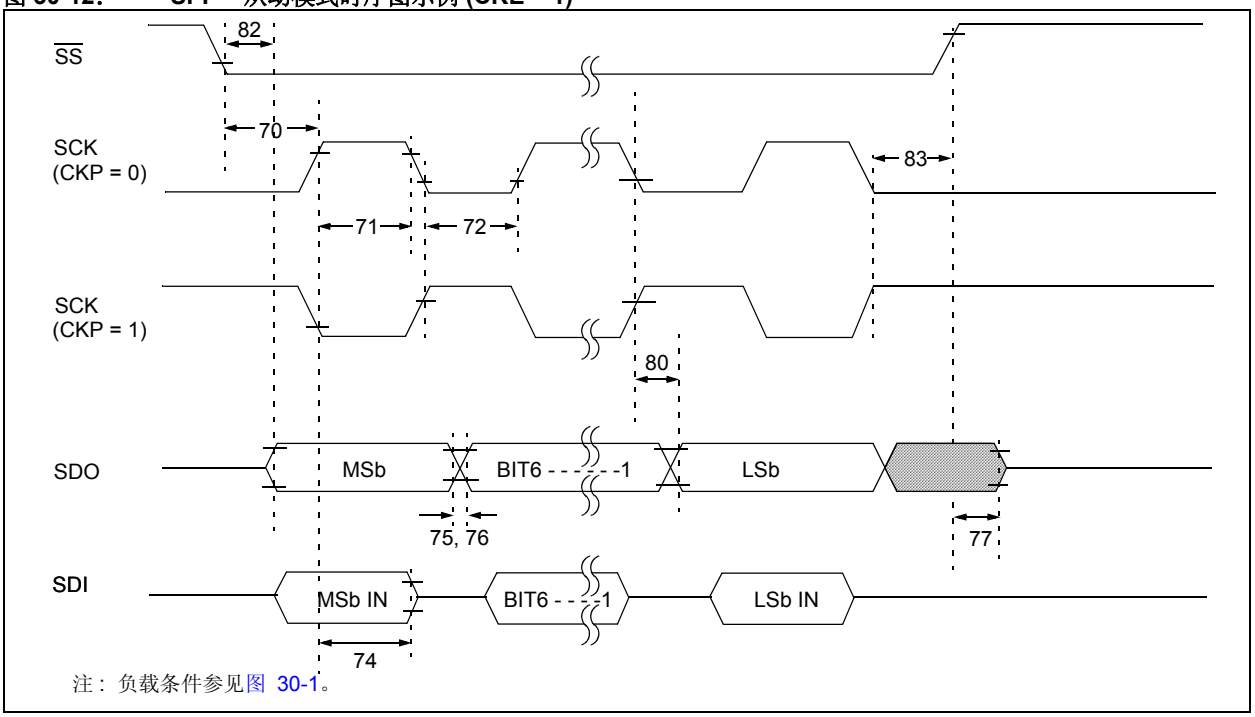
表 30-23: SPI™ 从动模式要求示例 (CKE = 0)

参数编号	符号	特性		最小值	典型值†	最大值	单位	条件
70	TssL2scH, TssL2scL	SS↓ 到 SCK↓ 或 SCK↑ 输入		TCY	—	—	ns	
71	TscH	SCK 输入高电平时间 (从动模式)	连续	1.25TCY + 30	—	—	ns	
71A			单字节	40	—	—	ns	注 1
72	TscL	SCK 输入低电平时间 (从动模式)	连续	1.25TCY + 30	—	—	ns	
72A			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	Byte1 的末个时钟边沿到 Byte2 的首个时钟边沿间的时间		1.5TCY + 40	—	—	ns	注 1
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	PIC16**C**XXX	—	10	25	ns	
			PIC16**LC**XXX		20	45	ns	
76	TdoF	SDO 数据输出下降时间		—	10	25	ns	
77	TssH2doZ	SS↑ 到 SDO 输出高阻态		10	—	50	ns	
78	TscR	SCK 输出上升时间 (主控模式)	PIC16**C**XXX	—	10	25	ns	
			PIC16**LC**XXX		20	45	ns	
79	TscF	SCK 输出下降时间 (主控模式)		—	10	25	ns	
80	TscH2doV, TscL2doV	SCK边沿后SDO数据输出有效	PIC16**C**XXX	—	—	50	ns	
			PIC16**LC**XXX		—	100	ns	
83	TscH2ssH, TscL2ssH	在 SCK 边沿到后 SS ↑		1.5TCY + 40	—	—	ns	

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 只有在使用参数 71A 和 72A 时，才需要参数 73A。

图 30-12： SPI™ 从动模式时序图示例 (CKE = 1)



注：负载条件参见图 30-1。

表 30-24: SPI™ 从动模式要求示例 (CKE = 1)

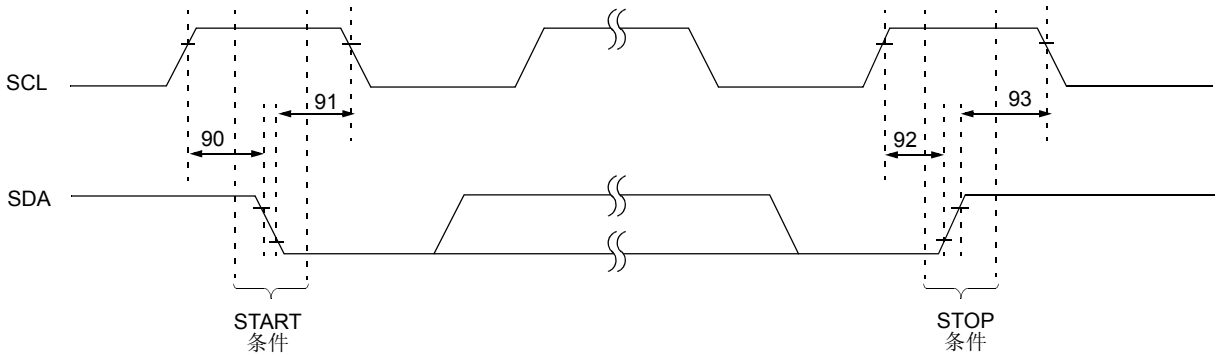
参数编号	符号	特性		最小值	典型值†	最大值	单位	条件
70	TssL2scH, TssL2scL	$\overline{SS}$↓ 到 SCK 或 SCK↑ 输入		TCY	—	—	ns	
71	TscH	SCK 输入高电平时间	连续	1.25TCY + 30	—	—	ns	
71A		(从动模式)	单字节	40	—	—	ns	注 1
72	TscL	SCK 输入低电平时间	连续	1.25TCY + 30	—	—	ns	
72A		(从动模式)	单字节	40	—	—	ns	注 1
73A	TB2B	Byte1 的末个时钟边沿到 Byte2 的首个时钟边沿间的时间		1.5TCY + 40	—	—	ns	注 1
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	PIC16**C**XXX	—	10	25	ns	
			PIC16**LC**XXX		20	45	ns	
76	TdoF	SDO 数据输出下降时间		—	10	25	ns	
77	TssH2doZ	$\overline{SS}$↑ 到 SDO 输出高阻态		10	—	50	ns	
78	TscR	SCK 输出上升时间	PIC16**C**XXX	—	10	25	ns	
		(主控模式)	PIC16**LC**XXX	—	20	45	ns	
79	TscF	SCK 输出下降时间 (主控模式)		—	10	25	ns	
80	TscH2doV, TscL2doV	SCK 边沿到后 SDO 数据输出有效	PIC16**C**XXX	—	—	50	ns	
			PIC16**LC**XXX	—	—	100	ns	
82	TssL2doV	在 $\overline{SS}$↓ 沿到后 SDO 数据输出有效	PIC16**C**XXX	—	—	50	ns	
			PIC16**LC**XXX	—	—	100	ns	
83	TscH2ssH, TscL2ssH	SCK 边沿后 $\overline{SS}$ ↑		1.5TCY + 40	—	—	ns	

†除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: 只有在使用参数 71A 和 72A 时，才需要参数 73A。

30.20 SSP I²C 模式时序波形图及要求示例

图 30-13： SSP I²C™ 总线启动 / 停止位时序图示例

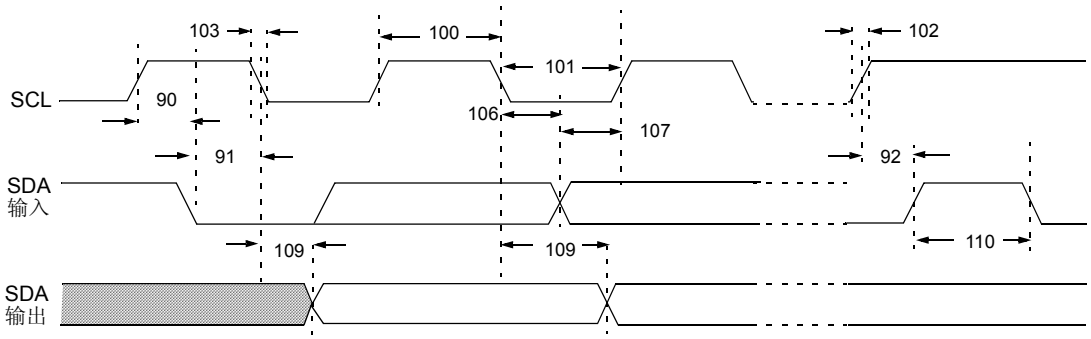


注：负载条件参见图 30-1。

表 30-25： SSP I²C 总线启动 / 停止位要求

参数编号	符号	特性		最小值	典型值	最大值	单位	条件
90	TSU:STA	START 条件建立时间	100 kHz 模式	4700	—	—	ns	仅与 START 条件重复有关
			400 kHz 模式	600	—	—		
91	THD:STA	START 条件保持时间	100 kHz 模式	4000	—	—	ns	该周期之后，产生第一个时钟脉冲
			400 kHz 模式	600	—	—		
92	TSU:STO	STOP 条件建立时间	100 kHz 模式	4700	—	—	ns	
			400 kHz 模式	600	—	—		
93	THD:STO	STOP 条件保持时间	100 kHz 模式	4000	—	—	ns	
			400 kHz 模式	600	—	—		

图 30-14： SSP I^2C^{TM} 总线数据时序图示例



注：负载条件参见图 30-1。

表 30-26: SSP I^2C^{TM} 总线数据要求示例

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	PIC16CXXX 的最小工作频率为 1.5 MHz
			400 kHz 模式	0.6	—	µs	PIC16CXXX 的最小工作频率为 10 MHz
			SSP 模块	1.5TCY	—		
101	TLOW	时钟低电平时间	100 kHz 模式	4.7	—	µs	PIC16CXXX 的最小工作频率为 1.5 MHz
			400 kHz 模式	1.3	—	µs	PIC16CXXX 的最小工作频率为 10 MHz
			SSP Module	1.5TCY	—		
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20 + 0.1Cb	300	ns	指定 Cb 从 10 到 400 pF
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20 + 0.1Cb	300	ns	指定 Cb 从 10 到 400 pF
90	TSU:STA	START条件建立时间	100 kHz 模式	4.7	—	µs	仅与 START 条件重复有关
			400 kHz 模式	0.6	—	µs	
91	THD:STA	START 条件保持时间	100 kHz 模式	4.0	—	µs	该周期之后，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	STOP 条件建立时间	100 kHz 模式	4.7	—	µs	
			400 kHz 模式	0.6	—	µs	
109	TAA	时钟输出有效	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	在开始新的传送之前总线必须空闲的时间
			400 kHz 模式	1.3	—	µs	
D102	Cb	总线容性负载		—	400	pF	

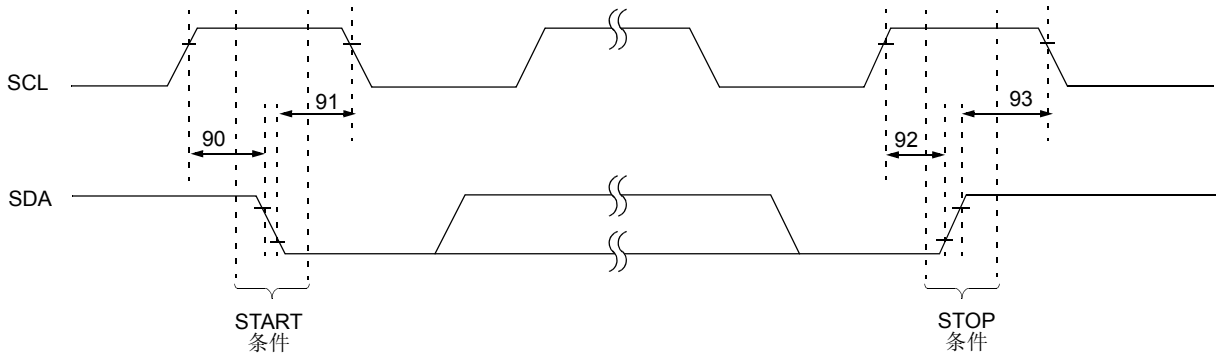
注 1: 作为发送器，为了避免意外产生 START 或 STOP 条件，器件必须提供该内部最小延迟时间，以连通 SCL 下降沿的非定义区域 (最小值 300 ns)。

2: 可将快速模式 I^2C 总线器件用于标准模式 I^2C 总线系统中，但必须满足 TSU:DAT ≥ 250 ns 的要求。如果器件不延长 SCL 信号的 LOW 周期，上述要求将自动满足。如果该器件延长了 SCL 信号的 LOW 周期，其下一个数据位必须输出到 SDA 线路。

SCL 线路释放前 TR max. + TSU:DAT = 1000 + 250 = 1250 ns (根据标准方式 I^2C 总线规范)。

30.21 MSSP I²C 模式时序波形图及要求示例

图 30-15： MSSP I²C 总线启动 / 停止位时序波形图示例



注： 负载条件参见图 30-1。

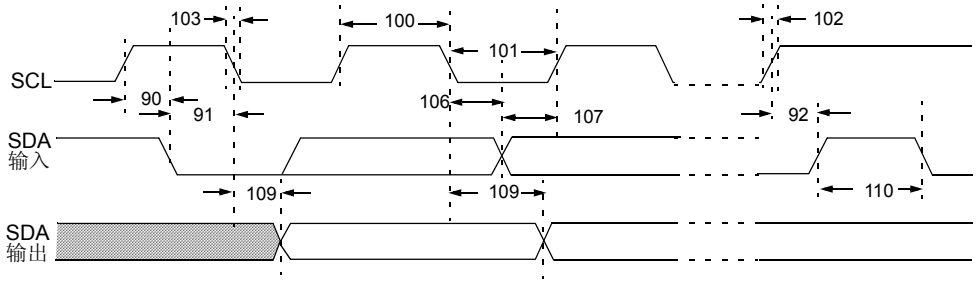
表 30-27： MSSP I²C 总线启动 / 停止位要求

参数编号	符号	特性		最小值	典型值	最大值	单位	条件
90	TSU:STA	START 条件建立时间	100 kHz 模式	2(Tosc)(BRG + 1) §	—	—	ns	仅与重复的 START 条件有关
			400 kHz 模式	2(Tosc)(BRG + 1) §	—	—		
			1 MHz 模式 (1)	2(Tosc)(BRG + 1) §	—	—		
91	THD:STA	START 条件保持时间	100 kHz 模式	2(Tosc)(BRG + 1) §	—	—	ns	在这个周期之后，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	STOP 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	STOP 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) §	—	—		

§ 该参数须由设计验证。 I²C 规范所需的值参见 **“附录”** 中的图 A-11。

所有 I²C 引脚的最大引脚电容为 10 pF。

图 30-16： MSSP I²C™ 总线数据时序图示例



注：负载条件参见图 30-1。

表 30-28： MSSP I²C™ 总线数据要求示例

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	指定 Cb 为 10 到 400 pF
			400 kHz 模式	20 + 0.1Cb	300	ns	
			1 MHz 模式 (1)	—	300	ns	
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	指定 Cb 为 10 到 400 pF
			400 kHz 模式	20 + 0.1Cb	300	ns	
			1 MHz 模式 (1)	—	100	ns	
90	TSU:STA	START 条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1) §	—	ms	仅与 START 条件重复有关
			400 kHz 模式	2(TOSC)(BRG + 1) §	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1) §	—	ms	
91	THD:STA	START 条件保持时间	100 kHz 模式	2(TOSC)(BRG + 1) §	—	ms	该周期之后，产生第一个时钟脉冲
			400 kHz 模式	2(TOSC)(BRG + 1) §	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1) §	—	ms	
106	THD:DAT	数据输入保持时间	100 kHz 模式	0	—	ns	
			400 kHz 模式	0	0.9	ms	
			1 MHz 模式 (1)	TBD	—	ns	
107	TSU:DAT	数据输入建立时间	100 kHz 模式	250	—	ns	注 2
			400 kHz 模式	100	—	ns	
			1 MHz 模式 (1)	TBD	—	ns	
92	TSU:STO	STOP 条件建立时间	100 kHz 模式	2(TOSC)(BRG + 1) §	—	ms	
			400 kHz 模式	2(TOSC)(BRG + 1) §	—	ms	
			1 MHz 模式 (1)	2(TOSC)(BRG + 1) §	—	ms	
109	TAA	时钟输出有效	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	开始新的传送之前总线必须空闲的时间
			400 kHz 模式	1.3 ‡	—	ms	
			1 MHz 模式 (1)	TBD	—	ms	
D102 ‡	Cb	总线容性负载		—	400	pF	

§ 该参数须由设计验证。I²C 规范所需的值参见 **“附录”** 中的图 A-11。

‡ 这些参数仅供设计参考，未经测试，也未经特性测试。

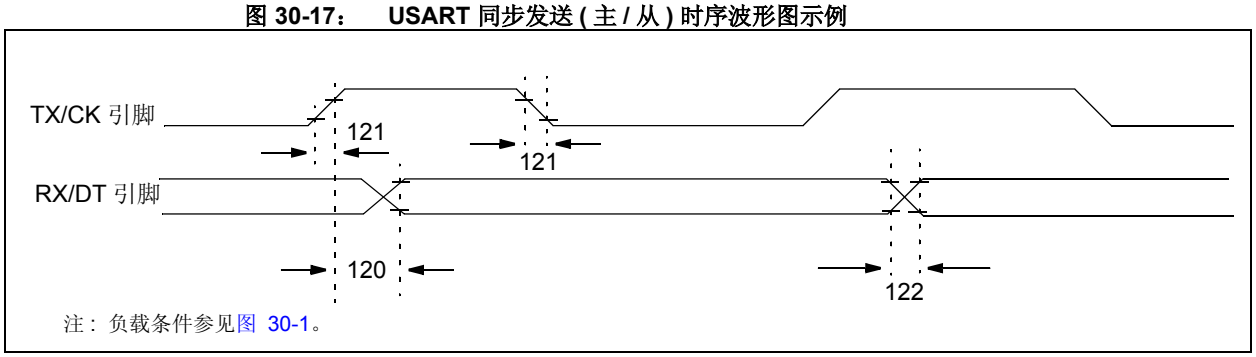
注 1: 所有 I²C 引脚的最大引脚电容为 10 pF。

2: 快速模式 I²C 总线器件可在标准模式 I²C 总线系统上应用，但必须满足 TSU:DAT ≥ 250 ns 的要求。如果器件不延长 SCL 信号的 LOW 周期，上述要求将自动满足。如果该器件延长了 SCL 信号的 LOW 周期，其下一个数据位必须输出到 SDA 线路。

SCL 线路释放前，参数 102 + 参数 107 = 1000 + 250 = 1250 ns (100 kHz 模式)。

30.22 USART/SCI 时序波形图及要求示例

图 30-17： USART 同步发送 (主 / 从) 时序波形图示例



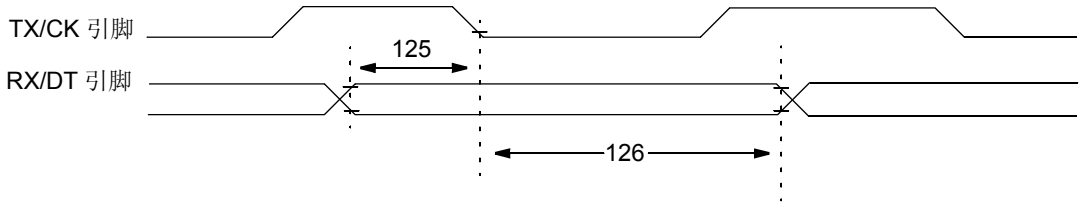
注： 负载条件参见图 30-1。

表 30-29: USART 同步发送 (主 / 从) 要求示例

参数编号	符号	特性		最小值	典型值 †	最大值	单位	条件
120	TckH2dtV	SYNC XMIT (MASTER & SLAVE) 时钟高电平到数据输出有效	PIC16**C**XXX	—	—	80	ns	
			PIC16**LC**XXX	—	—	100	ns	
121	Tckrf	时钟输出上升和下降时间 (主控模式)	PIC16**C**XXX	—	—	45	ns	
			PIC16**LC**XXX	—	—	50	ns	
122	Tdtrf	数据输出上升和下降时间	PIC16**C**XXX	—	—	45	ns	
			PIC16**LC**XXX	—	—	50	ns	

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

图 30-18： USART 同步接收 (主控 / 从动) 时序波形图示例



注 : 负载条件参见图 30-1。

表 30-30: USART 同步接收 (主控 / 从动) 要求示例

参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
125	TdtV2ckl	SYNC RCV (MASTER & SLAVE) 在 CK ↓ 前数据设置 (DT 保持时间)	15	—	—	ns	
126	TckL2dtl	在 CK ↓ 后数据保持 (DT 保持时间)	15	—	—	ns	

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

30.23 8 位 A/D 时序波形图及要求示例

表 30-31: 8 位 A/D 转换器特性示例

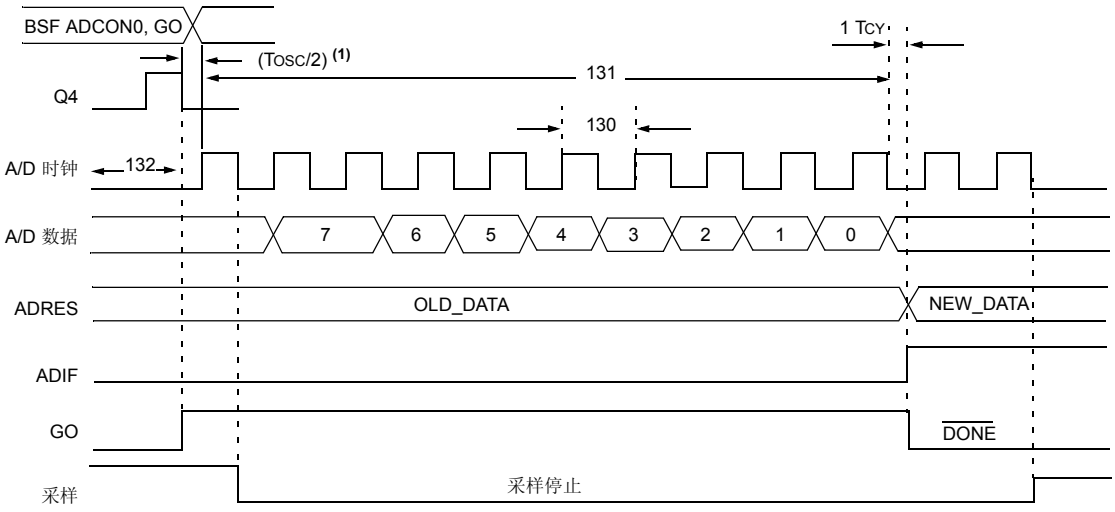
参数编号	符号	特性		最小值	典型值	最大值	单位	条件
A01	NR	分辨率		—	—	8 位	位	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A02	EABS	总绝对误差		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A03	EIL	积分线性误差		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A04	EDL	微分线性误差		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A05	EFS	满量程误差		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A06	EOFF	偏移误差		—	—	< ± 1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A10	—	单调性		—	保证	—	—	VSS ≤ VAIN ≤ VREF
A20	VREF	参考电压		3.0V	—	VDD + 0.3	V	
A25	VAIN	模拟输入电压		VSS - 0.3	—	VREF + 0.3	V	
A30	ZAIN	模拟电压源的推荐阻抗		—	—	10.0	kΩ	
A40	IAD	A/D 转换电流 (VDD)	PIC16**C**XXX	—	180	—	μA	当 A/D 开启时的平均电流消耗 **(注 1)**
			PIC16**LC**XXX	—	90	—	μA	
A50	IREF	VREF 输入电流 (注 2)		10	—	1000	μA	处于 VAIN 采集期间。基于 VHOLD 到 VAIN 的微分。对 CHOLD 的充电见 **“8 位 A/D 转换器”一章**。
				—	—	10	μA	在 A/D 转换周期中

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: A/D 转换器关闭时，除了消耗极小的漏电流外，不消耗任何其它电流。掉电电流参数包括任何来自 A/D 模块的漏电流。

2: VREF 电流来自选择作为参考输入的 RA3 引脚或 VDD 引脚。

图 30-19： 8 位 A/D 转换时序波形图示例



注 1: 如果选择 RC 为 A/D 的时钟源，在 A/D 时钟开始之前会增加一个 TCY 时间。这种模式允许 SLEEP 指令执行。

表 30-32: 8 位 A/D 转换要求

参数编号	符号	特性		最小值	典型值†	最大值	单位	条件
130	TAD	A/D 时钟周期	PIC16**C**XXX	1.6	—	—	µs	基于 TOSC VREF ≥ 3.0V
			PIC16**LC**XXX	2.0	—	—	µs	基于 TOSC VREF 满量程
			PIC16**C**XXX	2.0	4.0	6.0	µs	A/D RC 模式
			PIC16**LC**XXX	3.0	6.0	9.0	µs	A/D RC 模式
131	TCNV	转换时间 (不包括 S/H 时间) (注 1)		11	—	11	TAD	
132	TACQ	数据采集时间		注 2 5	20 —	— —	µs µs	最小时间值是放大器的稳定时间。如果 “新的” 输入电压相对于上次采样电压（如 CHOLD 上的电压）的变化不超过 1 LSb (即 5.12V 时 20 mV)，则可采用此时间。
134	TGO	Q4 到 A/D 时钟开始		—	2TOSC §	—	—	如果选择 RC 为 A/D 时钟源，在 A/D 时钟开始之前会增加一个 TCY 时间。这样即可使 SLEEP 指令得以执行。
136	TAMP	放大器稳定时间 (注 2)		1	—	—	µs	如果 “新的” 输入电压相对于上次采样电压（如 CHOLD 上的电压）的变化不超过 1 LSb (即 5.12V 时 20 mV)，则可采用此时间。
135	TSWC	转换 → 采样的切换时间		1 §	—	1 §	TAD	

† 除非另外声明，否则，“典型值” 一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

§ 这些参数须经设计验证。

注 1: ADRES 寄存器可在紧随其后的 TCY 周期中读出。
最小要求见 **“8 位 A/D 转换器” 一章**。

30.24 10 位 A/D 时序波形图及要求示例

表 30-33: 10 位 A/D 转换器特性示例

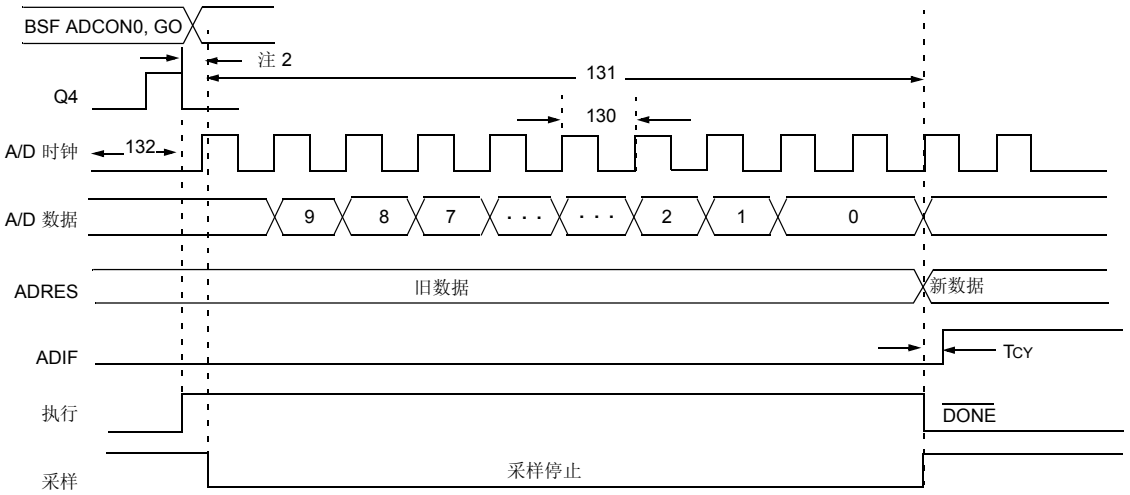
参数编号	符号	特性		最小值	典型值 †	最大值	单位	条件
A01	NR	分辨率		—	—	10	bit	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A02	EABS	绝对误差		—	—	<±1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A03	EIL	积分线性误差		—	—	<±1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A04	EDL	微分线性误差		—	—	<±1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A05	EFS	满量程误差		—	—	<±1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A06	EOFF	偏移误差		—	—	<±1	LSb	VREF = VDD = 5.12V, VSS ≤ VAIN ≤ VREF
A10	—	单调性		—	保证	—	—	VSS ≤ VAIN ≤ VREF
A20	VREF	参考电压 (VREFH - VREFL)		0V	—	—	V	对于无闭锁时
A20A				3V	—	—	V	对于 10 位分辨率时
A21	VREFH	参考高电压		AVSS	—	AVDD + 0.3V	V	
A22	VREFL	参考低电压		AVSS - 0.3V	—	AVDD	V	
A25	VAIN	模拟输入电压		AVSS - 0.3V	—	VREF + 0.3V	V	
A30	ZAIN	模拟电压源的建议阻抗		—	—	10.0	kΩ	
A40	IAD	A/D 转换电流 (VDD)	PIC16**C**XXX	—	180	—	μA	当 A/D 开启时的平均电流消耗 (注 1)
			PIC16**LC**XXX	—	90	—	μA	
A50	IREF	VREF 输入电流 (注 2)		10	—	1000	μA	处于 VAIN 采集期间。基于 VHOLD 到 VAIN 的差。对 CHOLD 的充电见 **“10 位 A/D 转换器”一章**。
				—	—	10	μA	在 A/D 转换周期期间

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: A/D 转换器关闭时，除了消耗极小的漏电流外，不消耗任何其它电流。掉电电流参数包括任何来自 A/D 模块的漏电流。

2: VREF 电流来自被选择作为参考输入的 RG0 和 RG1 引脚或 AVDD 和 AVSS 引脚。

图 30-20： 10 位 A/D 转换时序波形图示例



注 1: 如果选择 RC 为 A/D 的时钟源，在 A/D 时钟开始前会增加一个 TCY 时间。这种模式允许 SLEEP 指令执行。

2: 这是最小 RC 延迟 (典型值为 100 nS)，在这段时间里保持电容也从模拟输入端断开。

表 30-34: 10 位 A/D 转换要求

参数编号	符号	特性		最小值	典型值 †	最大值	单位	条件
130	TAD	A/D 时钟周期	PIC16**C**XXX	1.6	—	—	µs	基于 TOSC VREF ≥ 3.0V
			PIC16**LC**XXX	3.0	—	—	µs	基于 TOSC VREF 满量程
			PIC16**C**XXX	2.0	4.0	6.0	µs	A/D RC 模式
			PIC16**LC**XXX	3.0	6.0	9.0	µs	A/D RC 模式
131	TCNV	转换时间 (不包括采集时间) (注 1)		11 §	—	12 §	TAD	
132	TACQ	数据采集时间 (注 3)		15 10	— —	— —	µs µs	-40°C ≤ Temp ≤ 125°C 0°C ≤ Temp ≤ 125°C
136	TAMP	放大器稳定时间 (注 2)		1	—	—	µs	如果 " 新的 " 输入电压相对于上次采样电压 (如 CHOLD 上的电压) 的变化不超过 1 LSb (即 5.12V 时 5 mV)，则可采用此时间。
135	TSWC	转换 → 采集的变换时间		—	—	注 4		

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

§ 该参数须经设计验证。

注 1: ADRES 寄存器可在随后的 TCY 周期中读出。

2: 当输入电压的变化超过 1 LSb 时，最小条件参见 **“10 位 A/D 转换器”一章**。

3: 转换完成后当电压满量程变化时 (AVDD 至 AVSS, 或 AVSS 至 AVDD)，采样保持电容获取一个 " 新的 " 输入电压所需的时间。输入通道的源阻抗 (*RS*) 为 50 Ω。

4: 在器件时钟的下一个 Q4 周期。

30.25 积分型 A/D 时序波形图及要求示例

图 30-21： 积分型 A/D 转换周期示例

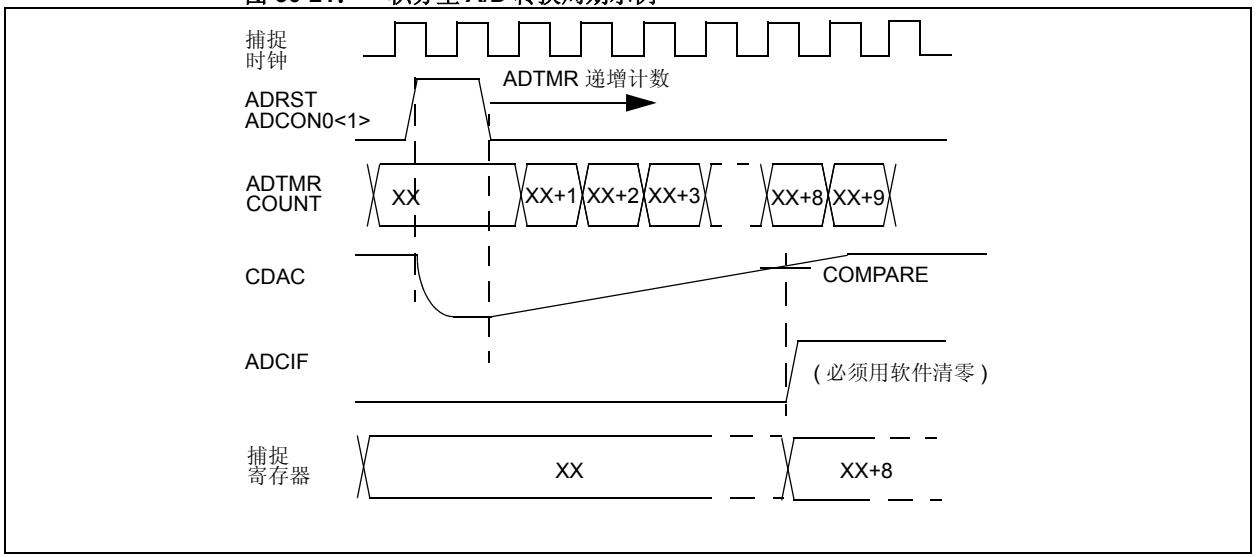


表 30-35: 积分型 A/D 组件特性

直流特性			标准运行条件 (除非另外声明) 运行温度 0°C ≤ TA ≤ +70°C 商业级 -40°C ≤ TA ≤ +85°C 工业级 -40°C ≤ TA ≤ +125°C 扩展级 运行电压 VDD 的范围见 DC 规范中的表 30-3 中的说明。				
参数编号	**符号**	**特性**	**最小值**	**典型值**	**最大值**	**单位**	**条件**
		积分型 A/D 的比较器					
A100	VAIN	模拟输入电压范围	VSS	—	VDD –1.4	V	
A101		输入偏移电压	–10	2	10	mV	在通用模式范围以外量得
A102	GDV	微分电压增益 (注 1)	—	100	—	dB	
A103	CMRR	共模抑制比 (注 1)	—	80	—	dB	VDD = 5V, TA = 25°C, 超出通用模式范围
A104	RRadc	电源抑制比 (注 1)	—	70	—	dB	TA = 25°C, VDD 最小值 ≤ VDD ≤ VDD 最大值
	TSET	**开启稳定时间**					
140		带隙参考电压 (至 < 0.1% 注 1)	—	1	10	ms	SLPCON寄存器上的REFOFF 位 1 → 0
141		可编程电流源 (至 < 0.1%)	—	1	10	ms	偏压源 (参考电压源) 开启时间 (REFOFF 1 → 0) (参考电压源开启）(注 1)
141A			—	1	10	µs	REFOFF = 0 (恒定值), ADCON1<7:4> 0000b → 1111b (参考电压源打开并处于稳定状态下) (注 3)
	TC	**温度系数 (注 1)**					
A110	TCBGR	带隙参考	— —	+50 –50	— —	ppm/°C	–40°C ≤ TA ≤ +25°C 25°C ≤ TA ≤ +85°C
A110A			— —	+20 –20	— —	ppm/°C	0°C ≤ TA ≤ +25°C 25°C ≤ TA ≤ +70°C
A111	TCPCS	可编程电流源	— —	+0.1 –0.1	— —	%/°C	–40°C ≤ TA ≤ +25°C 25°C ≤ TA ≤ +85°C
A112	TCkref	斜率参考电压分压器	—	20	—	ppm/°C	–40°C ≤ TA ≤ +85°C
	CA	**校准精度 (注 3, 5)**					所有的参数在 VDD = 5V 和 TA = +25°C 下校准
A120	CABGR	带隙参考电压	—	0.01	—	%	
A121	CASRV	斜率参考电压分压器	—	0.02	—	%	
	SN	**电源灵敏度 (注 1)**					
A130	SNBGR	带隙参考电压	—	0.04	—	%/V	从 VDD 最小值到 VDD 最大值
A131	SNPCS	可编程电流源	—	0.2	—	%/V	从 VDD 最小值到 VDD 最大值
A132	SNkref	斜率参考电压分压器	—		—	%/V	从 VDD 最小值到 VDD 最大值
		可编程电流源					
A140	IRES	分辨率	1.25	2.25	3.25	µA	1 LSb
A141	EIL	相对精度 (线性误差)	–1/2		+1/2	LSb	CDAC = 0V

30.26 LCD 时序波形图及要求示例

图 30-22： LCD 电压波形图示例

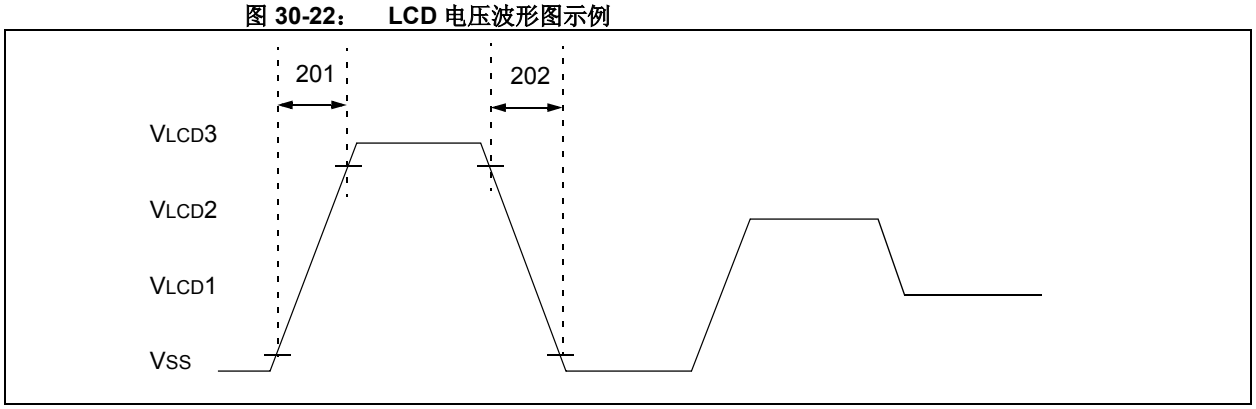


表 30-36: LCD 模块时序要求

参数编号	符号	特性	最小值	典型值 †	最大值	单位	条件
200	FLCDRC	LCDRC 振荡器频率	—	14	22	kHz	VDD = 5V, -40°C 至 +85°C
201	TrLCD	输出上升时间	—	—	200	µs	COM 输出 Cload = 5,000 pF SEG 输出 Cload = 500 pF VDD = 5.0V, T = 25°C
202	TfLCD	输出下降时间 (注 1)	TrLCD - 0.05TrLCD	—	TrLCD + 0.05TrLCD	µs	COM 输出 Cload = 5,000 pF SEG 输出 Cload = 500 pF VDD = 5.0V, T = 25°C

† 除非另外声明，否则，“典型值”一列中的数据为 5V, 25°C 条件下的值。这些参数仅供设计参考，未经测试。

注 1: VLCD 上的电源阻抗是 0Ω。

30.27 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的 (即有些针对低档系列，有些针对高档系列)，但其概念是相近的，通过适当修改并受一定的限制即可使用。目前与电气规范相关的应用笔记有：

标题	应用笔记 #
没有相关应用笔记	

30.28 版本历史

版本 A

这是描述电气规范的初始发行版。

第 31 章 器件特性

目录

本章包括以下一些主要内容：

31.1 简介 31-2
31.2 特性和电气规范 31-2
31.3 DC 和 AC 特性图表 31-2
31.4 版本历史 31-22

31.1 简介

Microchip 为其制造的器件提供特性信息。这些信息是在器件进行了完整的特性测试和数据分析后得到的。数据取自于器件的测试设备和基准模型。这些特性数据有助于设计人员更好地理解器件特性，以更好地判断器件是否满足应用要求。

31.2 特性和电气规范

特性和电气规范之间的差异可以类比为用户期望器件能做什么，和 Microchip 对器件测试后认定它能做什么。所提供的特性图表仅作为设计指南，未经测试，不作保证。

特性参数的极限值可能与电气规范章节所给出的测试数据有所不同。这是由于生产性测试设备的能力不同，并在必要时增加了一定的安全余量。

31.3 DC 和 AC 特性图表

每张表都给出了有助于设计的具体信息。这些数据是在固定条件下获得的。如果您的应用条件有别于测试条件，则测量的数据也将不同。

在一些图表中所给出的数据超过了规定的工作范围 (即超过了规定的 V_{DD} 范围)。这些数据仅供参考，器件在规定范围内将正常工作。

注 1: 在数据手册的特性章节中所给的数据是在一段时间内从分批取样的器件上得到的统计结果。“典型值”代表在 25°C 下的平均值，而“max”代表（平均值 +3σ)，“min”代表 (平均值 -3σ)，其中， σ 代表标准偏差。

31.3.1 IPD 和 VDD

IPD 是指器件处于休眠状态 (电源关闭) 时，器件消耗的电流，称为关断电流。这些测试是在所有 I/O 作为输入时进行的，无论这些 I/O 被拉高或拉低。即，测试时没有悬空的输入端，也没有引脚驱动输出负载。

在看门狗定时器 (WDT) 关闭和使能的两种情况下，分别给出特性图。这是因为 WDT 所需的 RC 振荡器会消耗额外电流，因此需分别给出。

在休眠状态下，器件的某些功能部件和模块仍可工作，其中一些模块如：

- 看门狗定时器 (WDT)
- 欠压复位 (BOR) 电路
- 定时器 Timer1
- 模数转换器
- LCD 模块
- 比较器
- 参考电压模块

如果上述功能在休眠状态下工作，器件将消耗更高的电流。当关闭所有功能部件时，器件的消耗电流最低 (即漏电流)。如果使能了多个功能部件，则可很容易地计算出消耗电流，即等于基本电流 (休眠模式下关闭所有模块) 加上功能部件的新增电流。例 31-1 给出了 5V 电源下，使能 WDT 和定时器 Timer1 振荡器时，计算器件典型工作电流的例子。

例 31-1: 使能 WDT 和定时器 TIMER1 振荡器时， IPD 的计算值 (@ 5V)

基本电流	14 nA	; 器件的漏电流
WDT 的新增电流	14 µA	; 14 µA - 14 nA = 14 µA
Timer1 的新增电流	22 µA	; 22 µA - 14 nA = 22 µA
休眠状态时的总电流	36 µA	;

图 31-1：典型 IPD － VDD 关系曲线示例 (WDT 被禁止， RC 振荡模式)

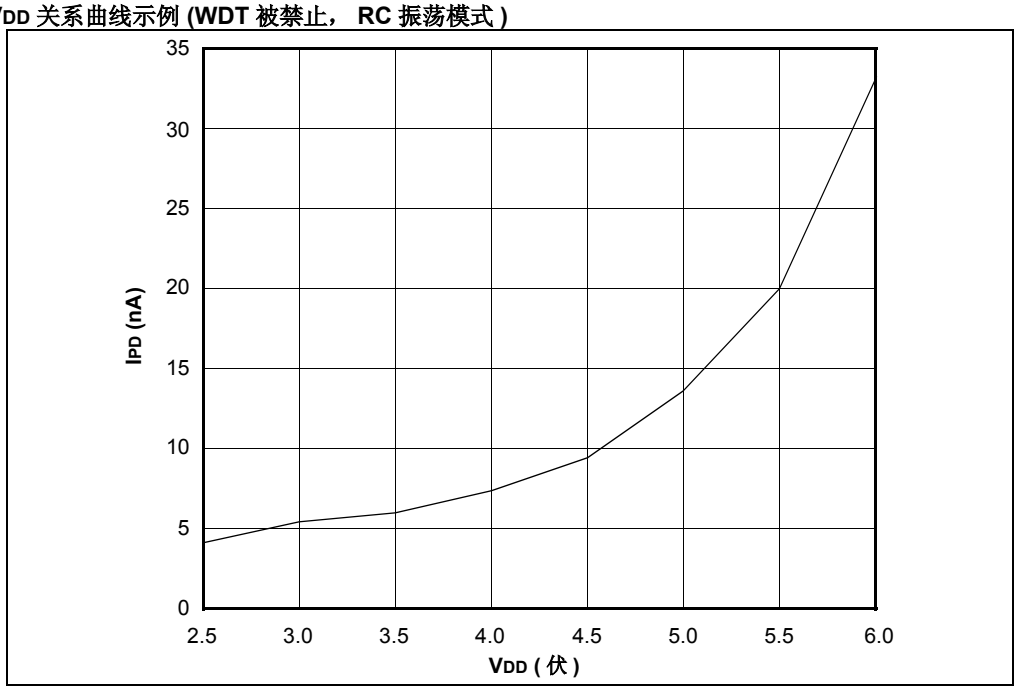


图 31-2：最大 IPD － VDD 关系曲线示例 (WDT 被禁止， RC 振荡模式)

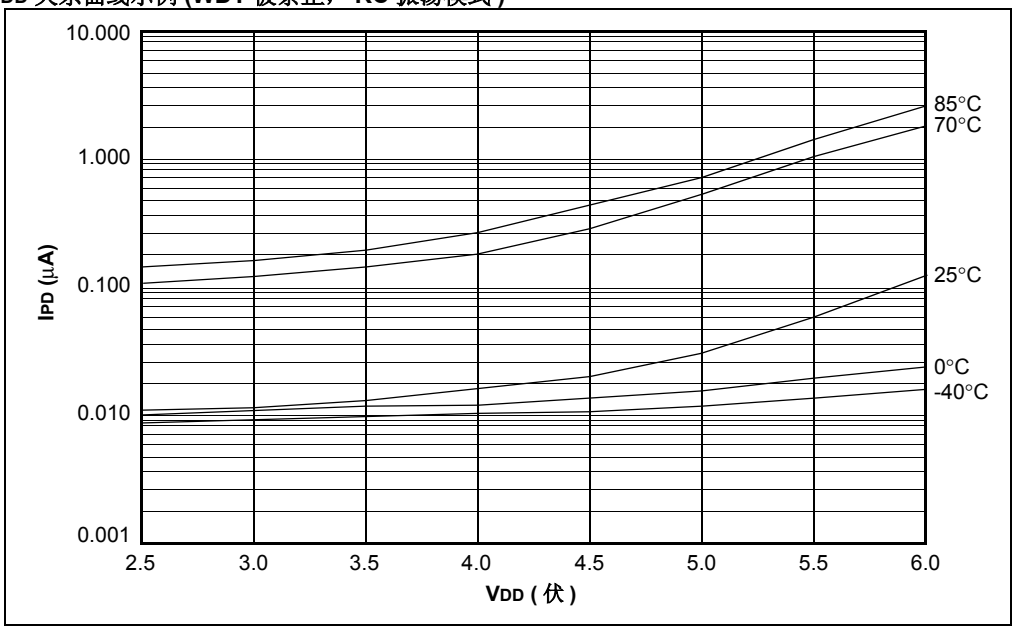


图 31-3：25°C 时，典型 IPD － VDD 关系曲线示例 (WDT 被使能， RC 振荡模式)

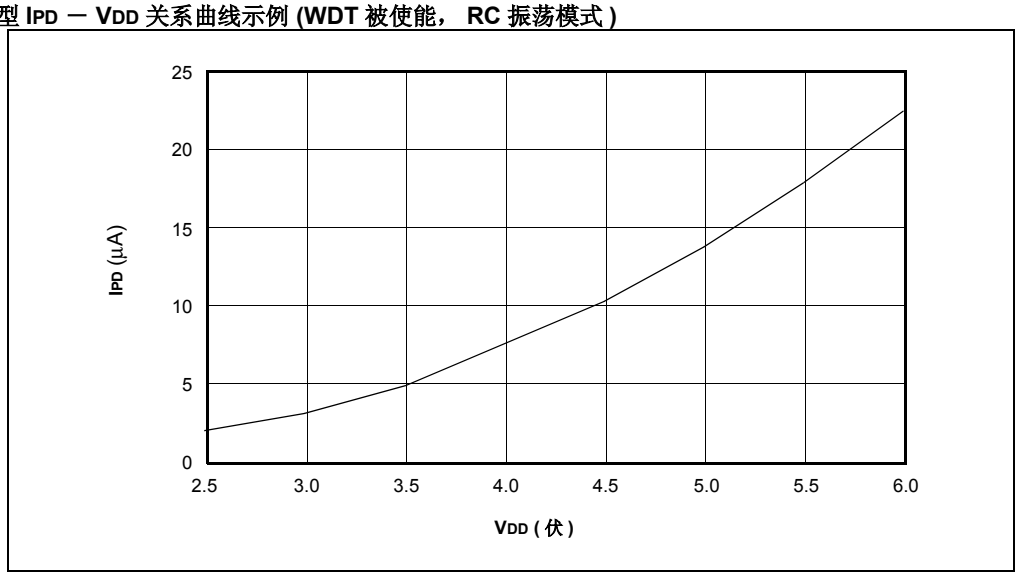


图 31-4：最大 IPD － VDD 关系曲线示例 (WDT 被使能， RC 振荡模式)

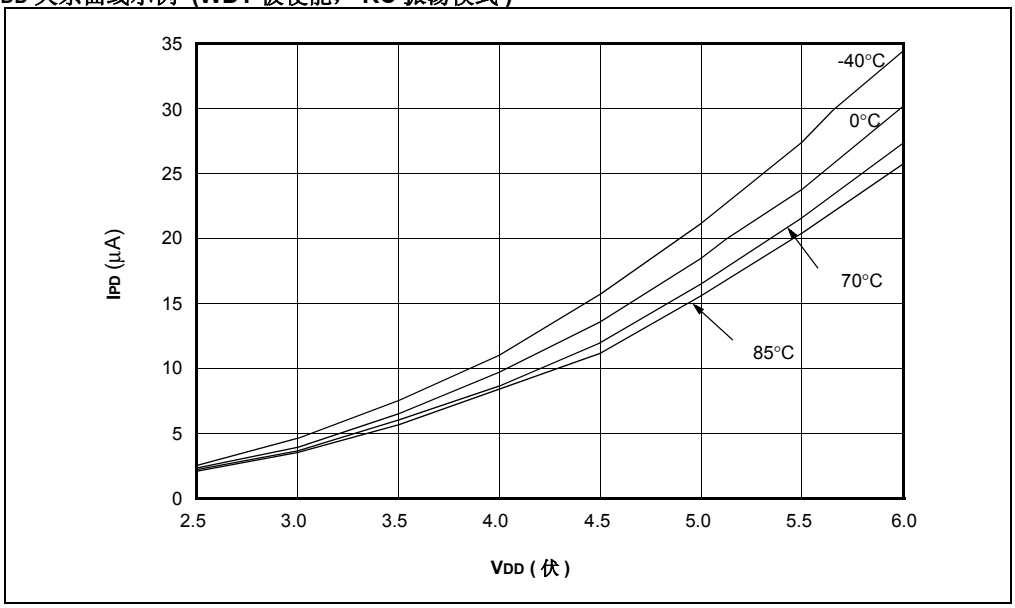
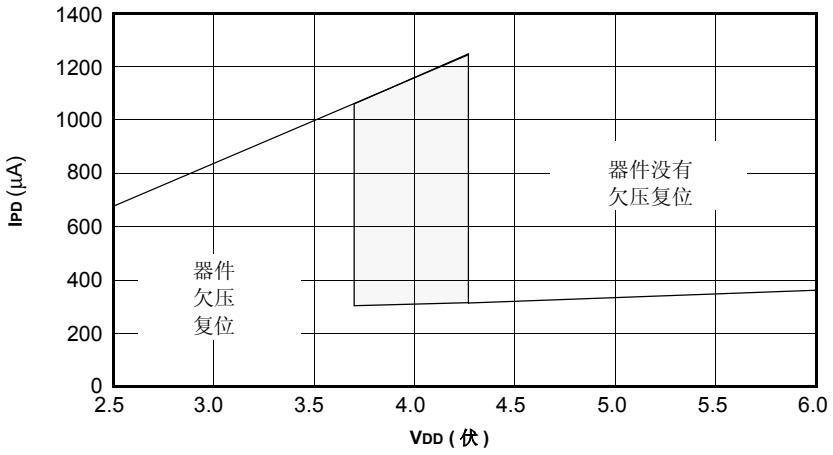
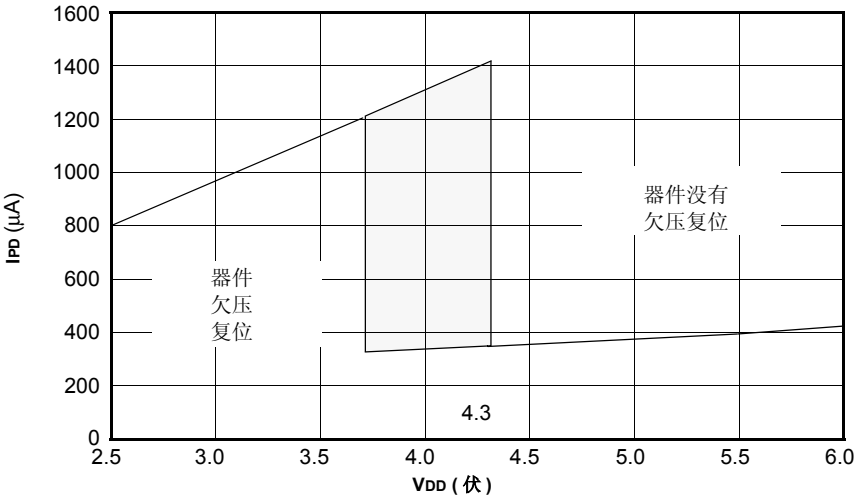


图 31-5：欠压检测使能时，典型 IPD — VDD 关系曲线示例 (RC 振荡模式)



阴影区代表欠压复位电路的内置滞后区。

图 31-6：欠压检测使能时，最大 IPD — VDD 关系曲线示例 (85°C 至 -40°C， RC 振荡模式)



阴影区代表欠压复位电路的内置滞后区。

图 31-7：Timer1 使能时，典型 IPD － VDD 的关系曲线示例 (32 kHz, RC0/RC1 = 33 pF/33 pF, RC 振荡模式)

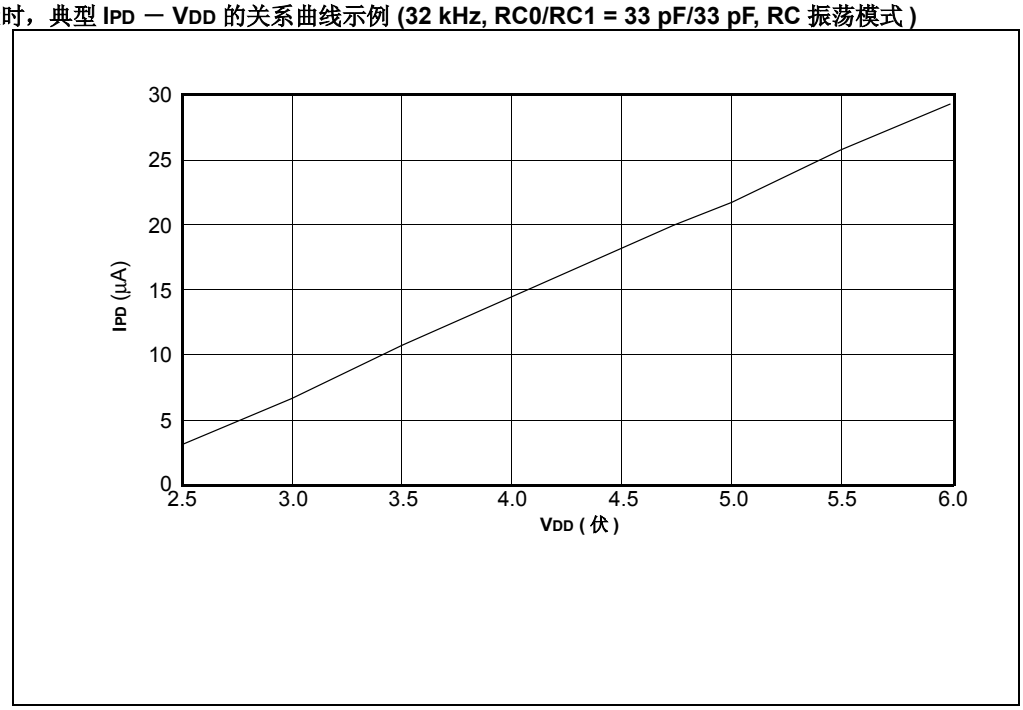
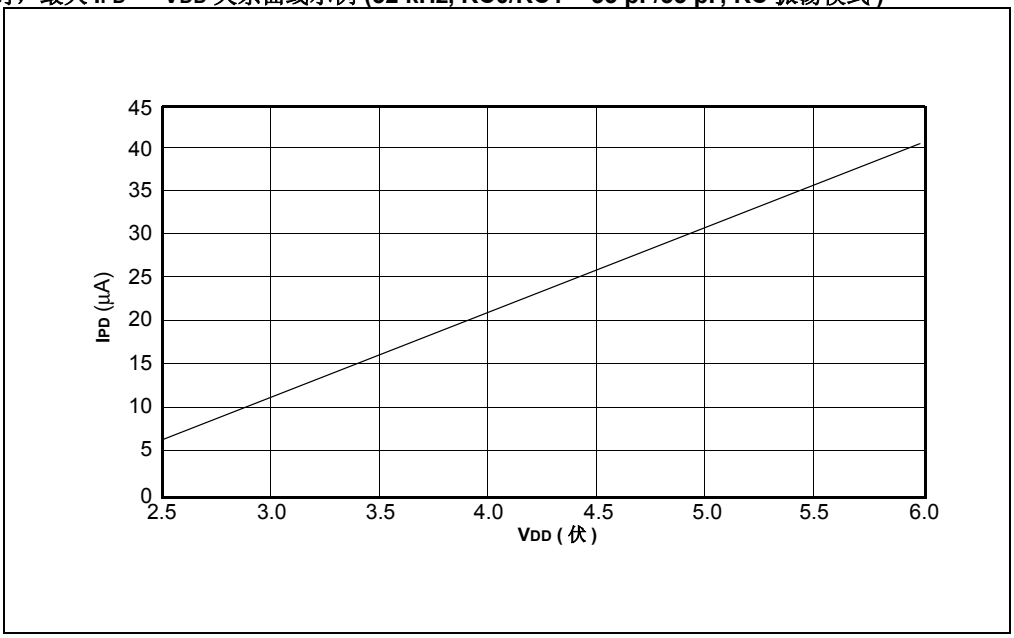


图 31-8：Timer1 使能时，最大 IPD － VDD 关系曲线示例 (32 kHz, RC0/RC1 = 33 pF/33 pF, RC 振荡模式)



31.3.2 IDD 与频率

IDD 是指器件处于运行模式时所消耗的电流 (I)。该测试是在所有 I/O 作为输入时进行的，无论这些 I/O 被拉高或拉低。即，测试时没有悬空的输入端，也没有引脚驱动输出负载。

IDD 与频率的关系图是从 Microchip 的自动化基准模型中测量所得，该模型被称为 DCS(数据收集系统)。 DCS 准确地反映出器件和特定部件的值，即，它不包含增加杂散电容或电流。

31.3.2.1 RC 测量

在测量 RC 时，DCS 先选定一个电阻和电容值，然后在规定范围内改变电压值。器件的运行频率将随着电压的变化而变化。对于一个固定的 RC，频率随着 VDD 的增加而增加。在该 RC 下完成测量后，再反复改变 RC 的值重新测量。图上各点分别对应器件的电压、电阻值 (R) 和电容值 (C)。

图 31-9：典型 IDD 一 频率关系曲线示例 (RC 振荡模式， @ 22 pF, 25°C)

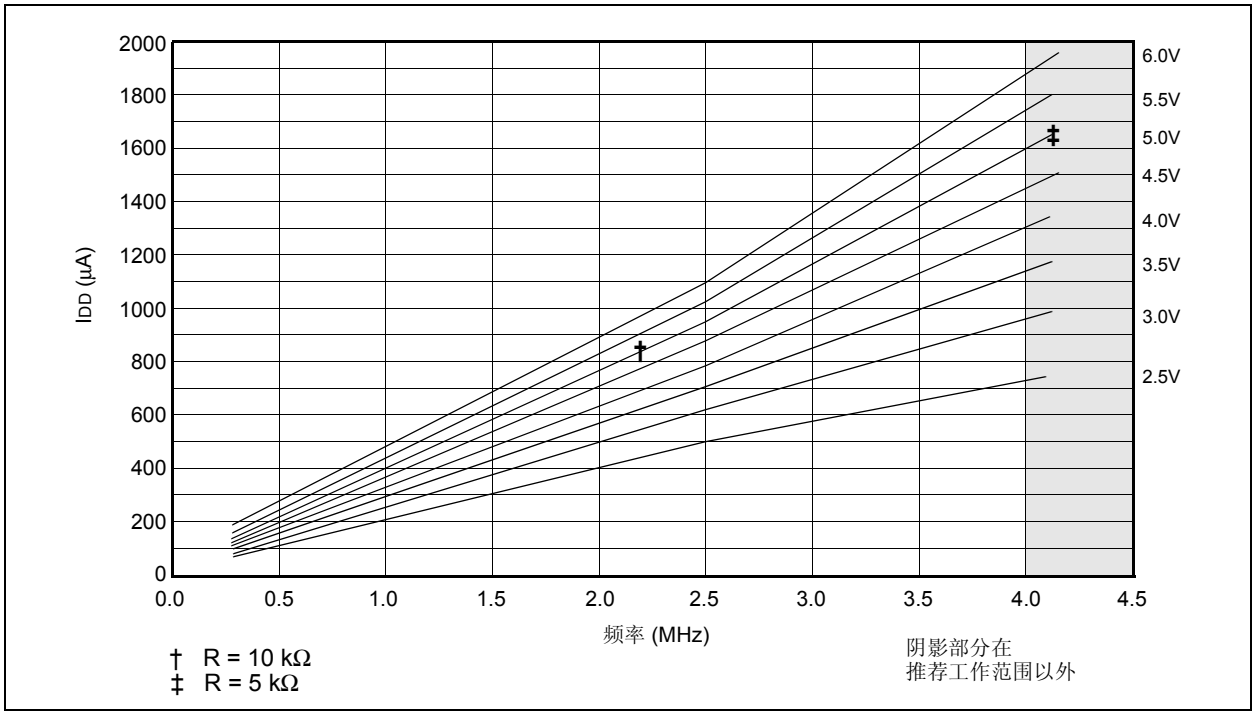


图 31-10：最大 IDD 一 频率关系曲线示例 (RC 振荡模式， @ 22 pF, -40°C 到 85°C)

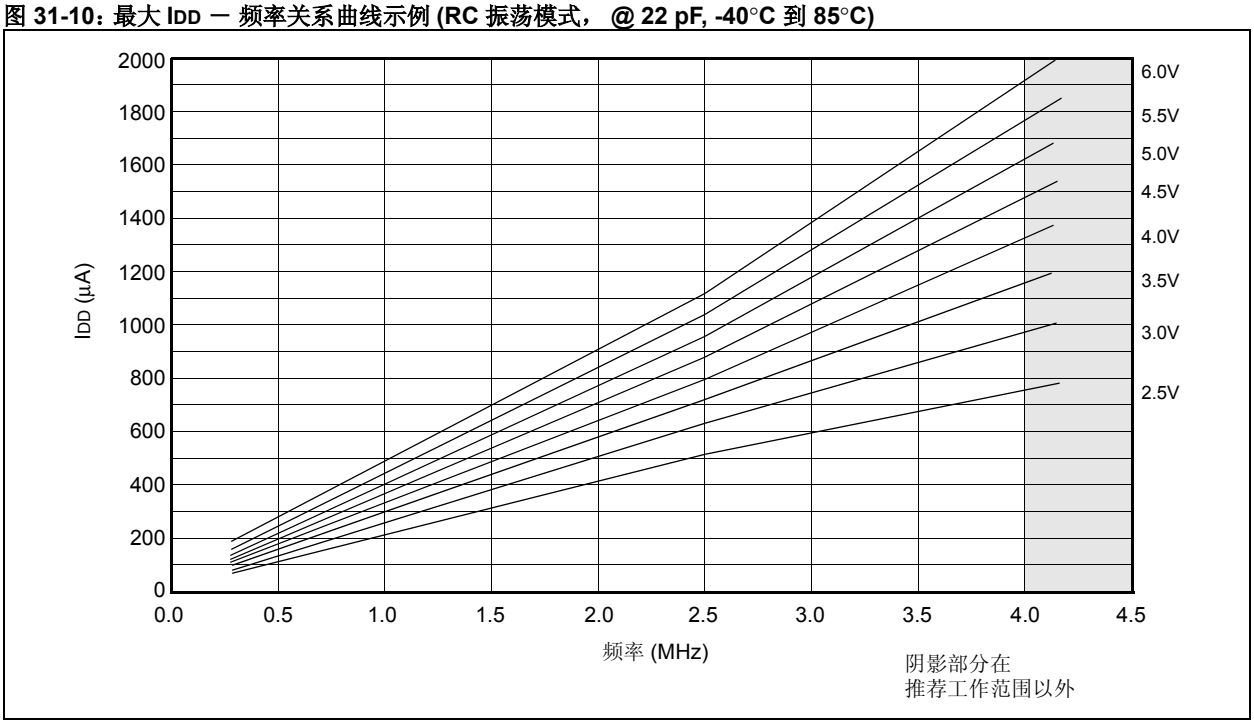


图 31-11：典型 IDD 一 频率关系曲线示例 (RC 振荡模式， @ 100 pF, 25°C)

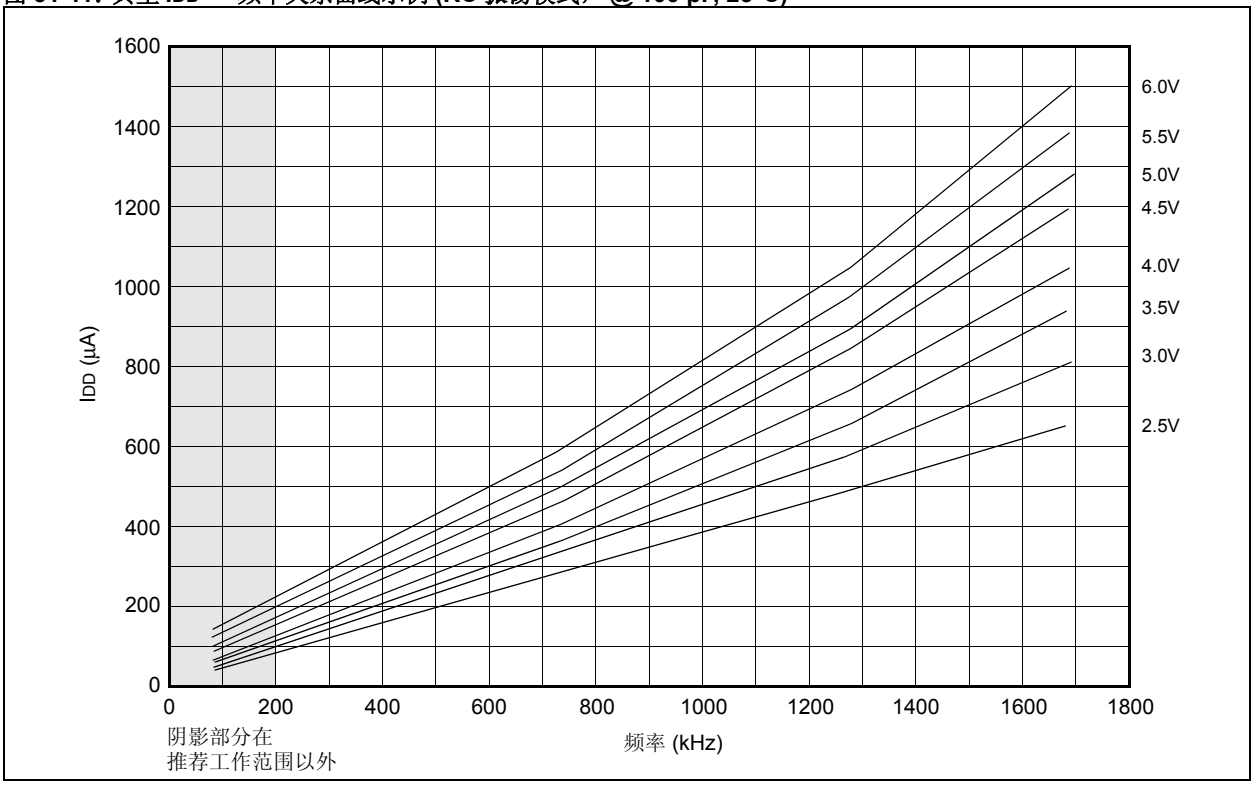


图 31-12：最大 IDD 一 频率关系曲线示例 (RC 振荡模式， @ 100 pF, -40°C 到 85°C)

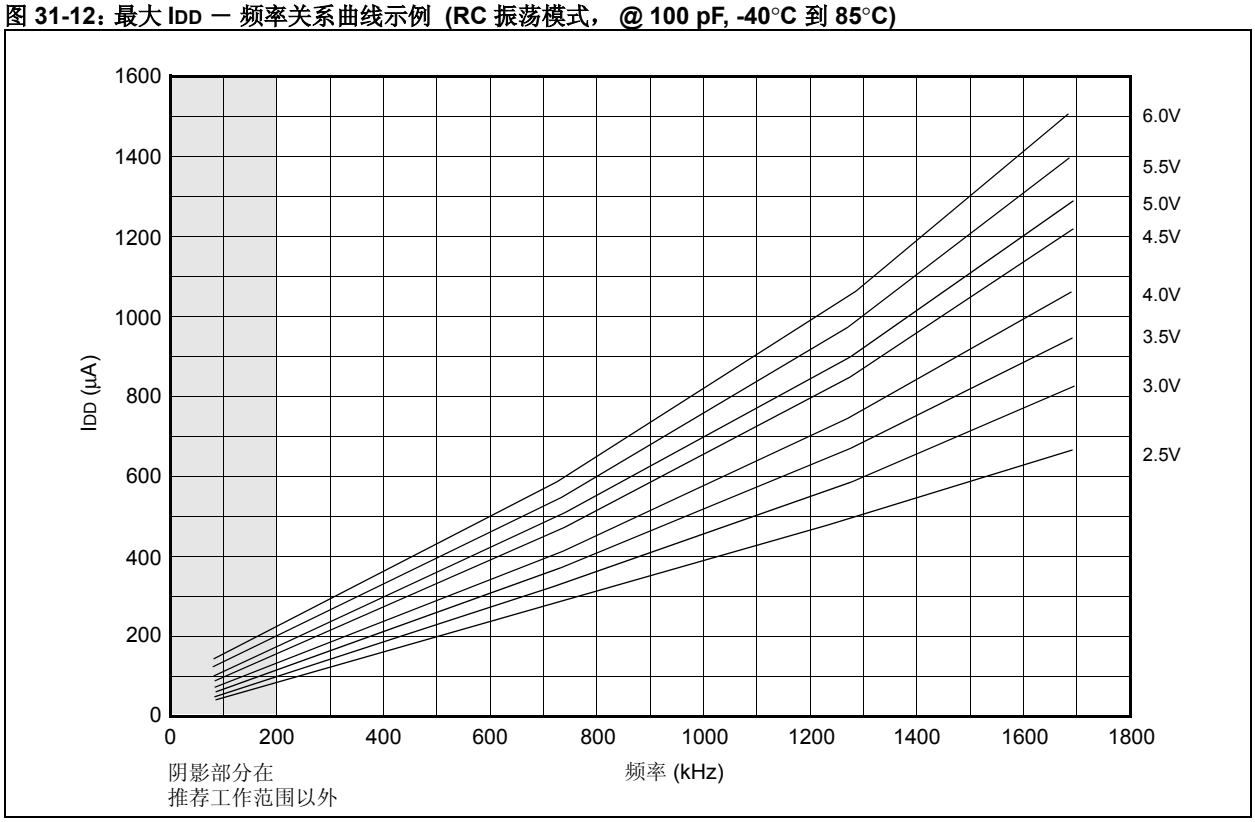


图 31-13：典型 IDD 一 频率关系曲线示例 (RC 振荡模式， @ 300 pF, 25°C)

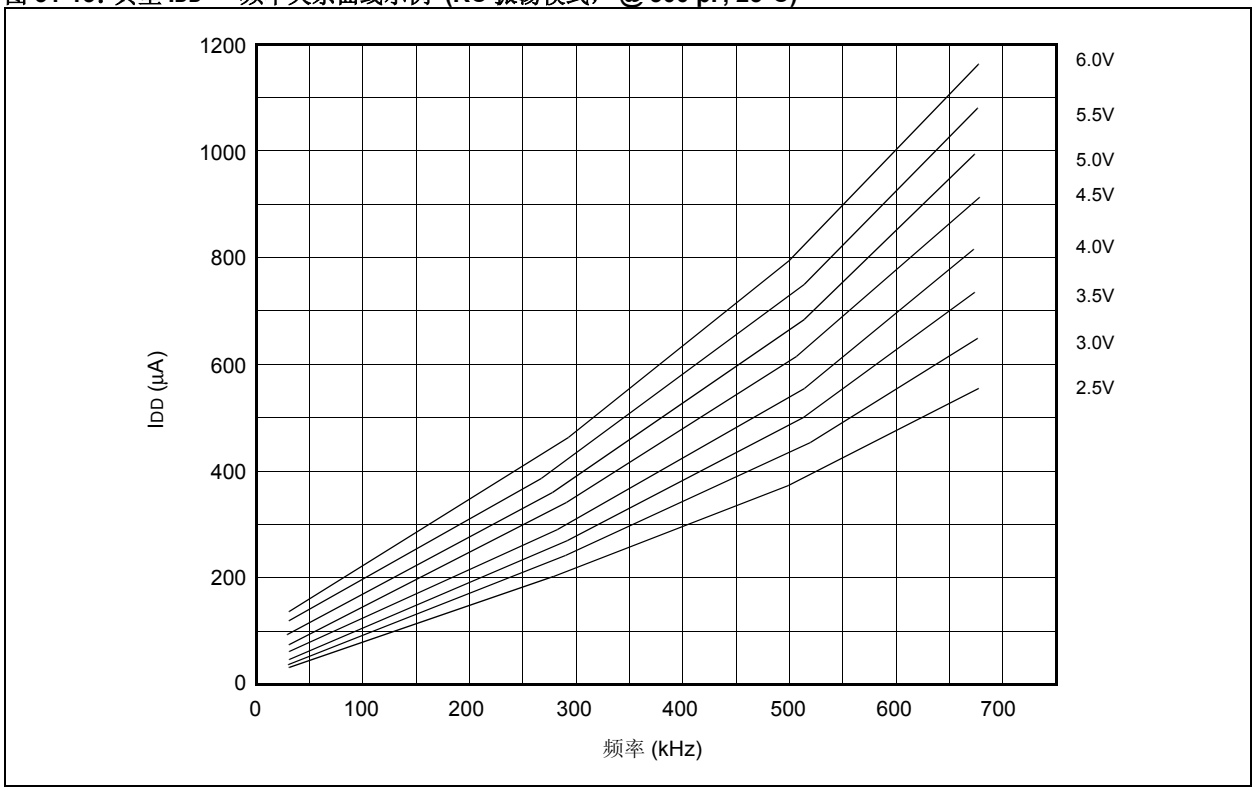


图 31-14：最大 IDD 一 频率关系曲线示例 (RC 振荡模式， @ 300 pF, -40°C 到 85°C)

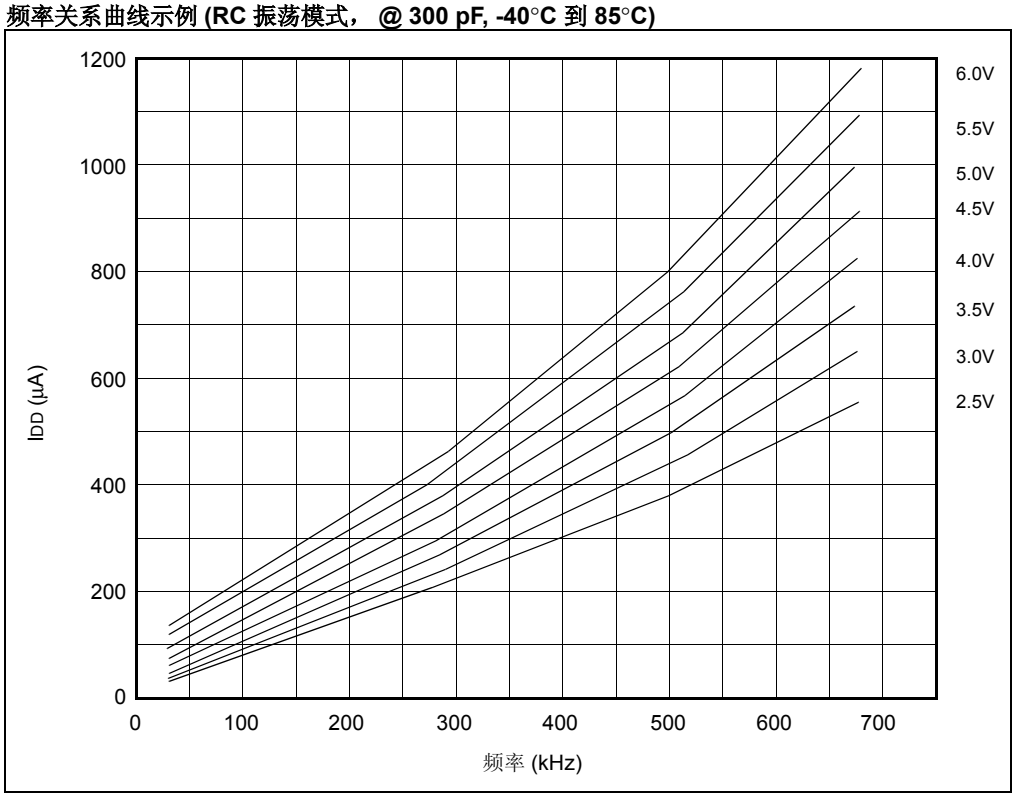
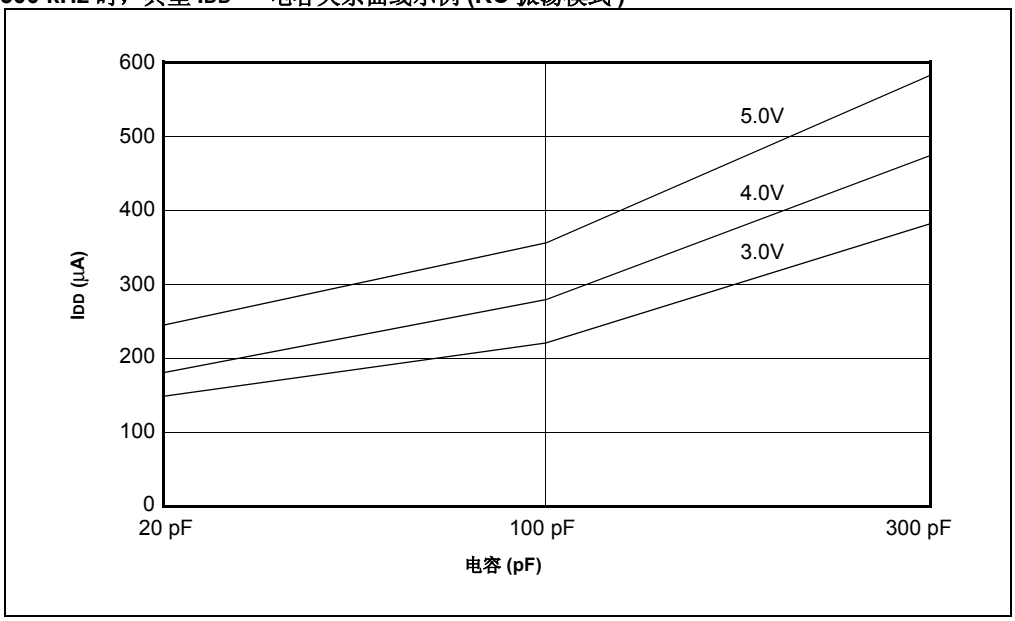


图 31-15：振荡频率为 500 kHz 时，典型 IDD 一 电容关系曲线示例 (RC 振荡模式)



31.3.2.2 晶体振荡器的测量

在数据收集系统中，使用不只一个晶体振荡器。在这项测试中，将一块晶体复用到器件电路中，而晶体的电容值可以改变。调整外接电容和电压以确定最佳电路特性 (如电流、振荡波形以及振荡器的起振)，再测量随电压变化的电流。然后更换另一个晶体振荡器，重复上述过程。

图 31-16：典型 IDD 一 频率关系曲线示例 (LP 模式 , 25°C)

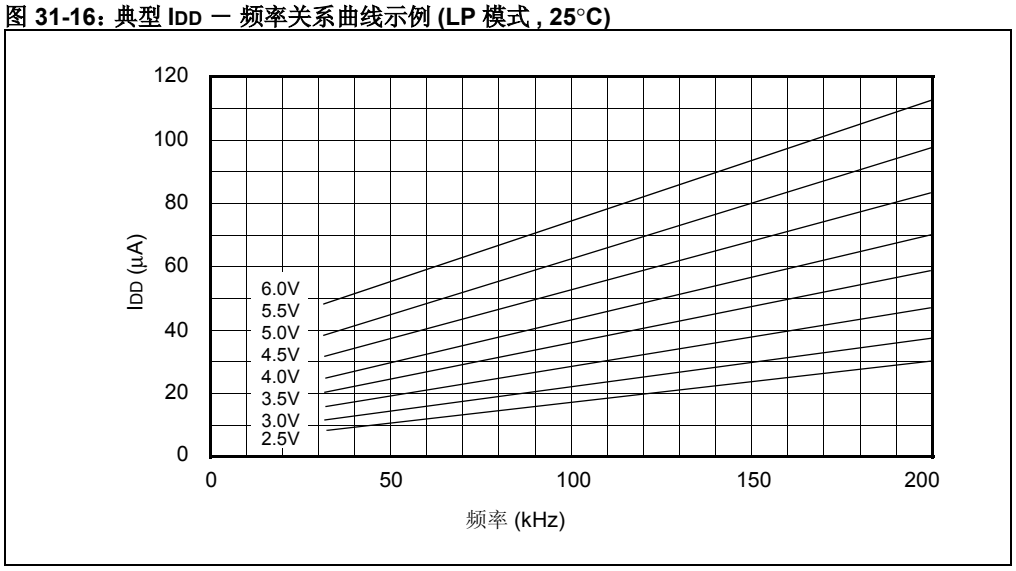


图 31-17：最大 IDD 一 频率关系曲线示例 (LP 模式 , 85°C 到 -40°C)

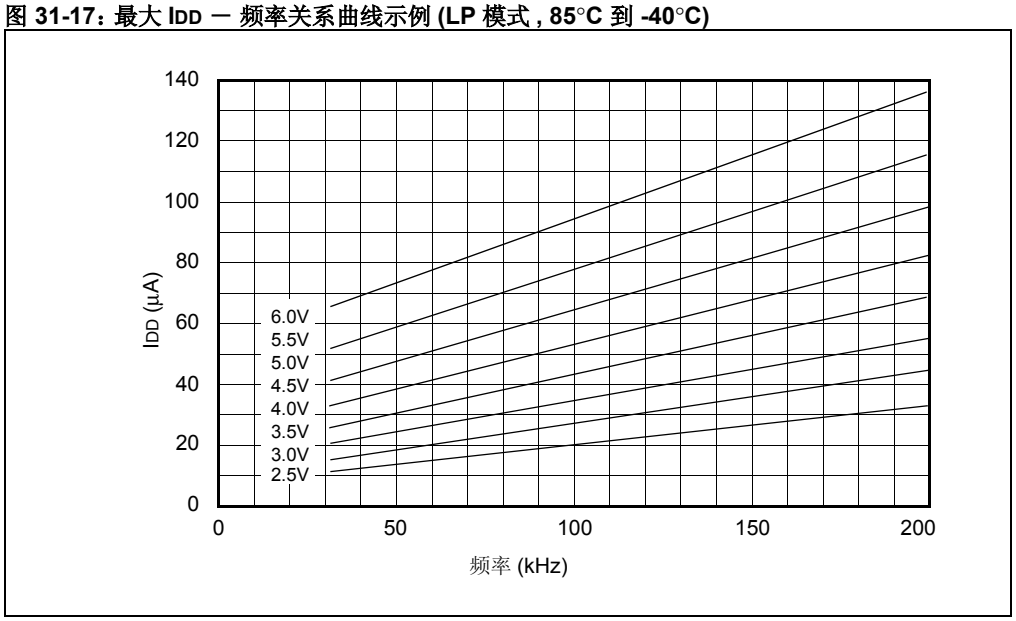


图 31-18：典型 IDD 一 频率关系曲线示例 (XT 模式 , 25°C)

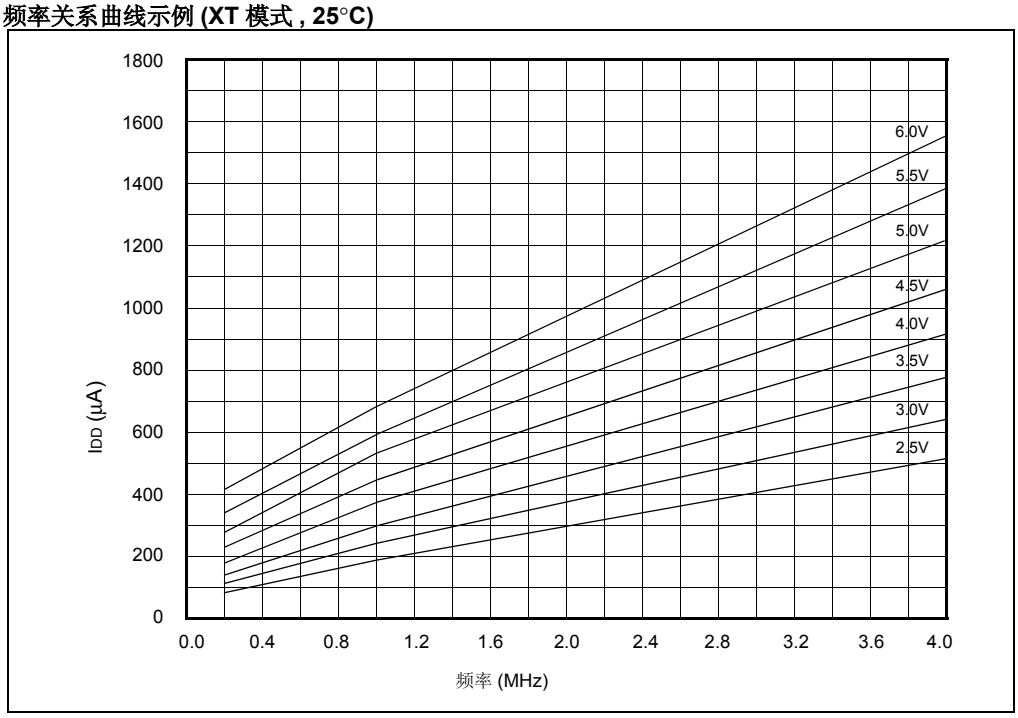


图 31-19：最大 IDD 一 频率关系曲线示例 (XT 模式 , -40°C 到 85°C)

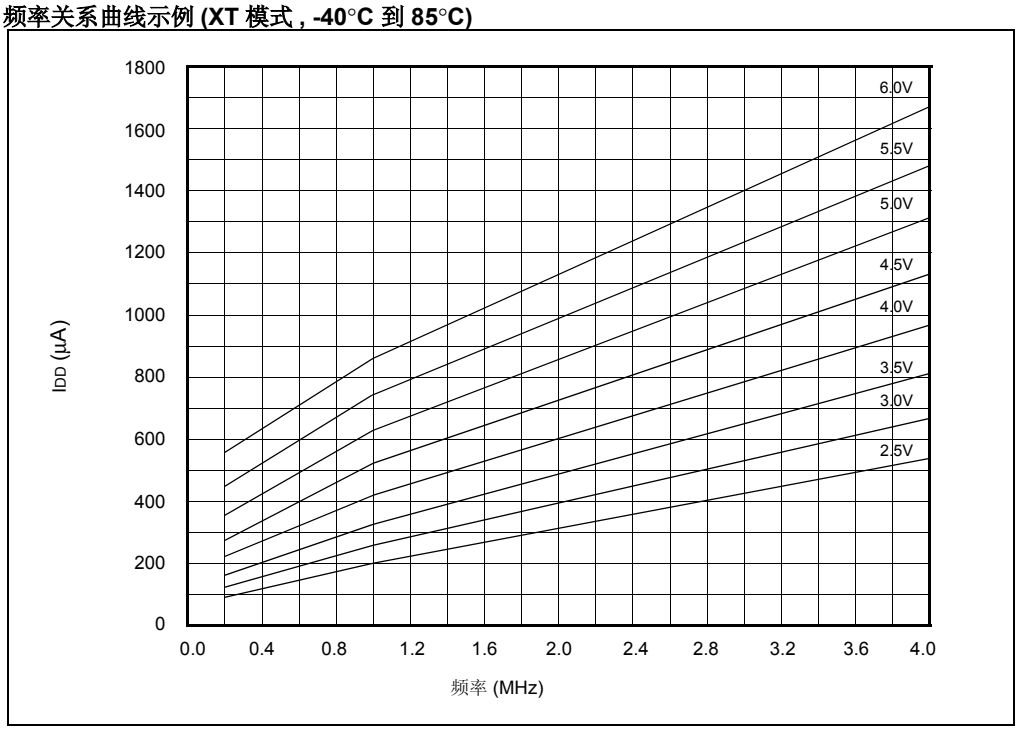


图 31-20：典型 IDD 一 频率关系曲线示例 (HS 模式 , 25°C)

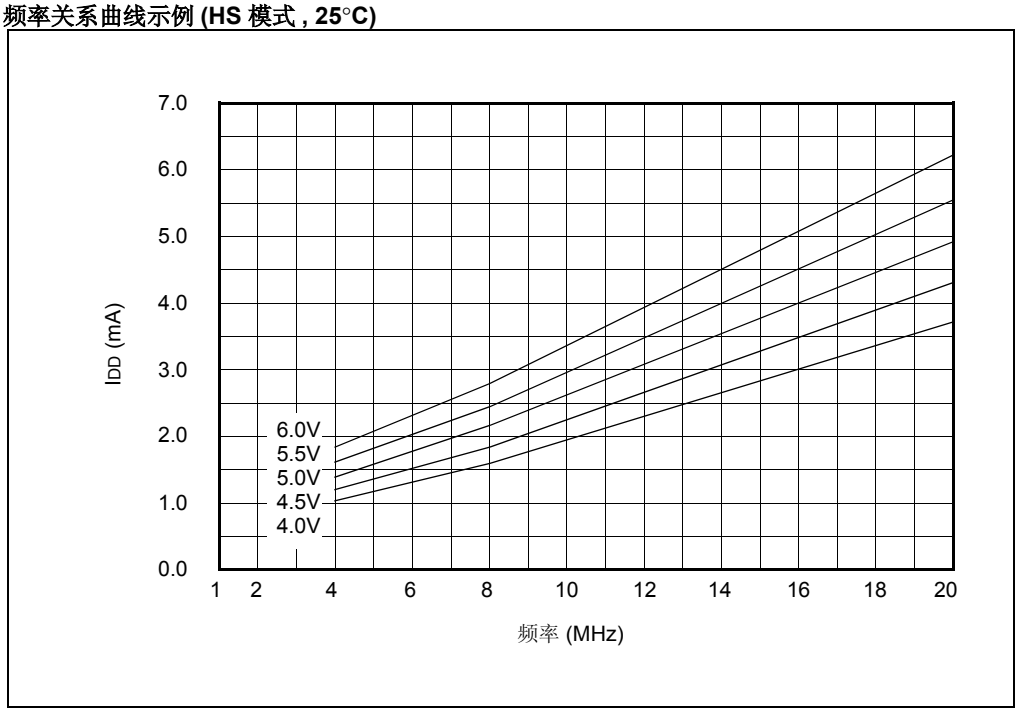
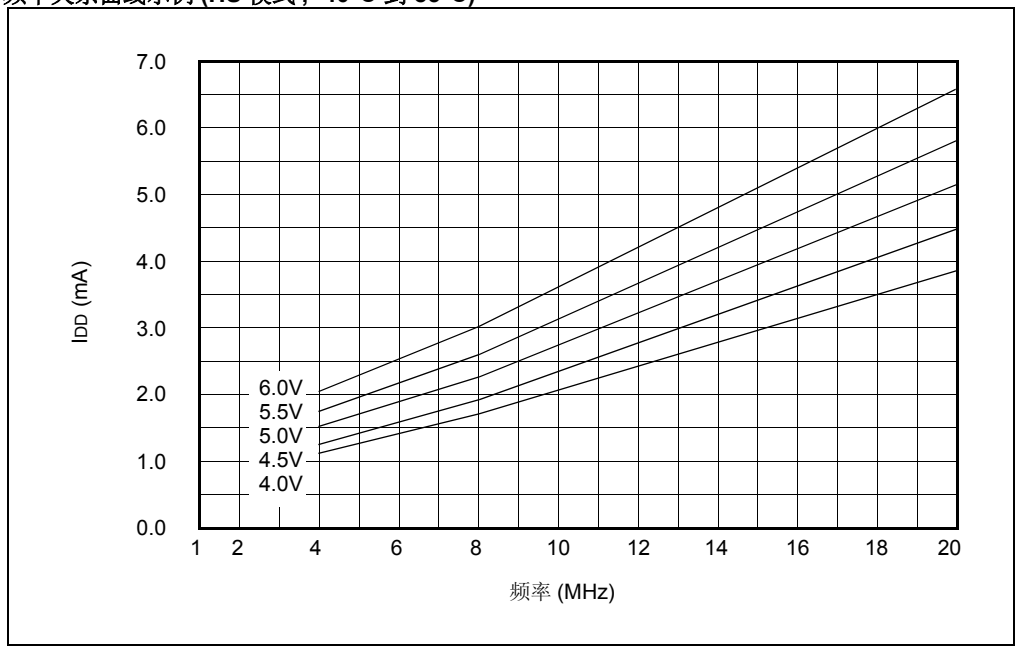


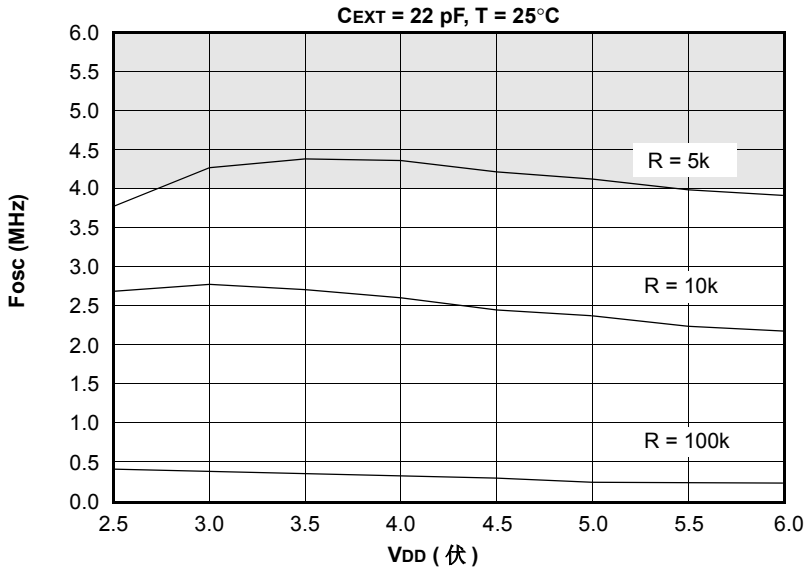
图 31-21：最大 IDD 一 频率关系曲线示例 (HS 模式 , -40°C 到 85°C)



31.3.3 RC 振荡器的频率

以下各图显示了器件电压的变化对 RC 振荡器频率产生的影响。测量时，先选定电容和电阻值，然后改变器件工作电压，测量 RC 振荡频率。下表显示了 5V 时，在给定 R 和 C 下的典型频率，以及由于器件制造工艺的差异所造成的频率变化。

图 31-22：典型 RC 振荡器频率 — VDD 关系曲线示例



阴影区在推荐的工作范围外。

图 31-23：典型 RC 振荡器频率 — VDD 关系曲线示例

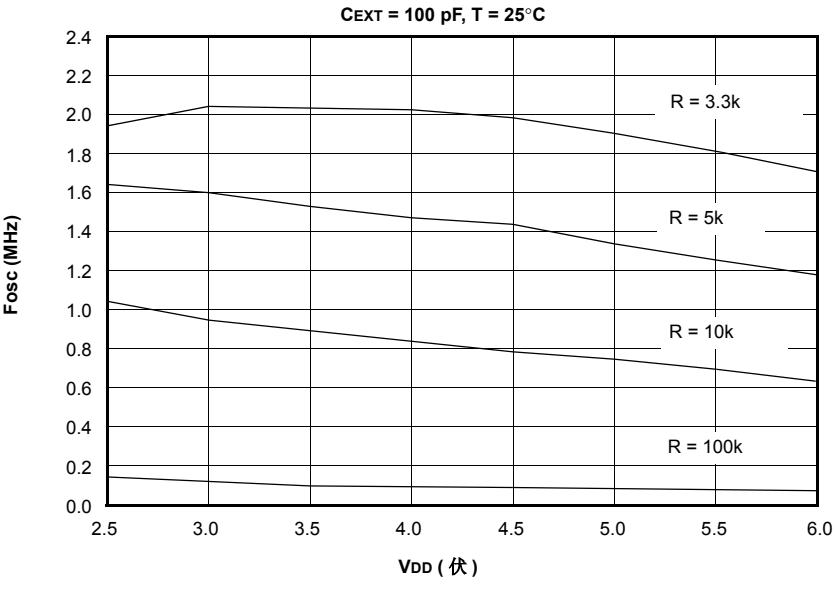


图 31-24：典型 RC 振荡器频率 – VDD 关系曲线示例

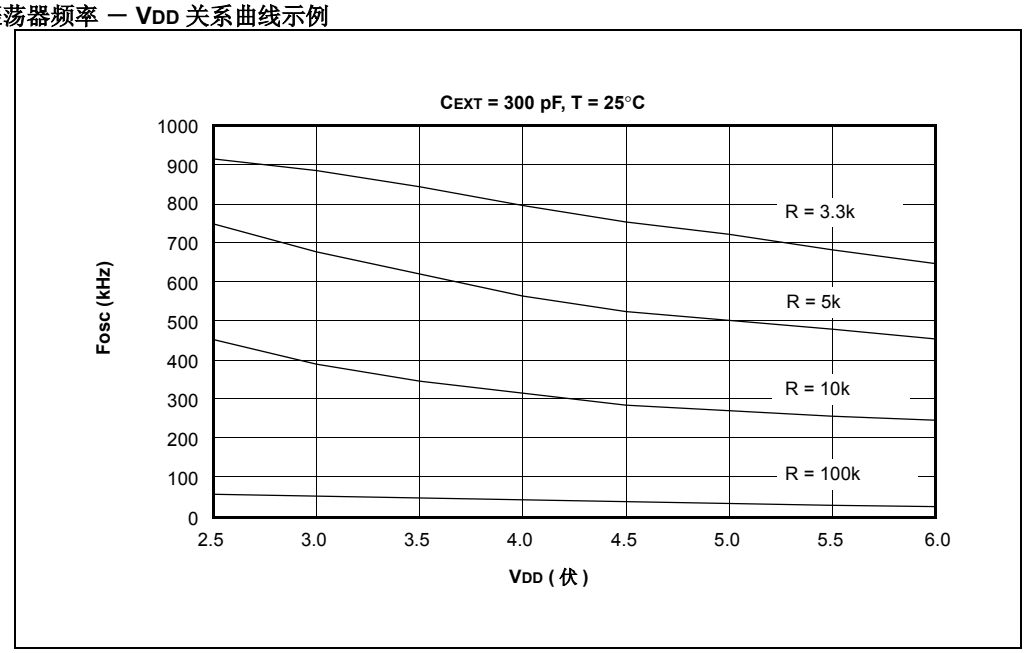


表 31-1： RC 振荡器频率示例

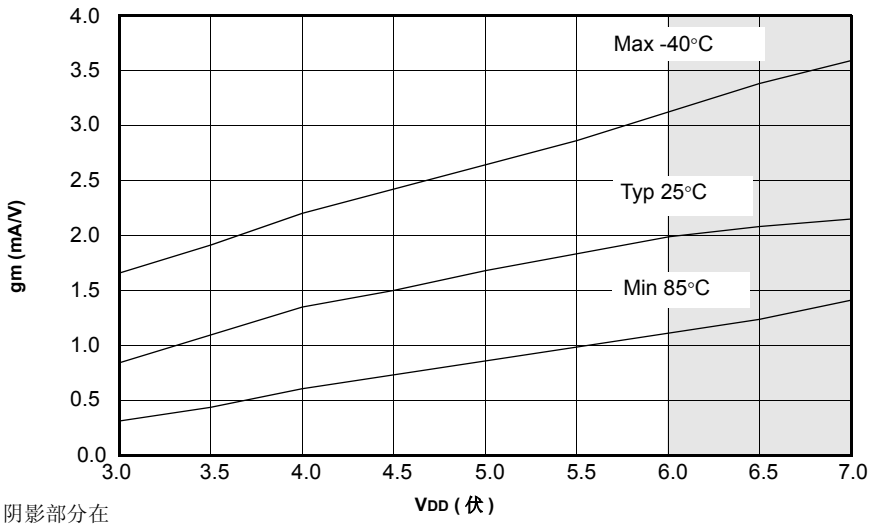
CEXT	REXT	平均值	
		Fosc @ 5V, 25°C	
22 pF	5k	4.12 MHz	± 1.4%
	10k	2.35 MHz	± 1.4%
	100k	268 kHz	± 1.1%
100 pF	3.3k	1.80 MHz	± 1.0%
	5k	1.27 MHz	± 1.0%
	10k	688 kHz	± 1.2%
	100k	77.2 kHz	± 1.0%
300 pF	3.3k	707 kHz	± 1.4%
	5k	501 kHz	± 1.2%
	10k	269 kHz	± 1.6%
	100k	28.3 kHz	± 1.1%

本表所示的差异的百分数是由正常制造偏差所导致的器件与器件间的差异。所示差异为在 VDD = 5V 时，平均值的 ±3 标准偏差。

31.3.4 振荡器的跨导

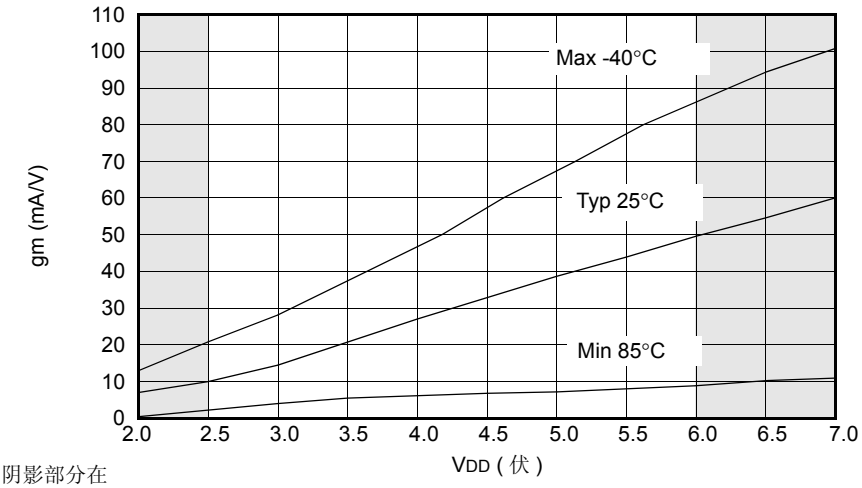
振荡器的跨导是指振荡器的增益。当振荡器跨导增加时，振荡器的增益也增大，导致振荡器电路的电流消耗增加。同时，随着跨导的增加，振荡器电路所能支持的最大工作频率增加，或振荡器的起振时间减小。

图 31-25：HS 振荡器的跨导 (gm) ─ VDD 关系曲线示例



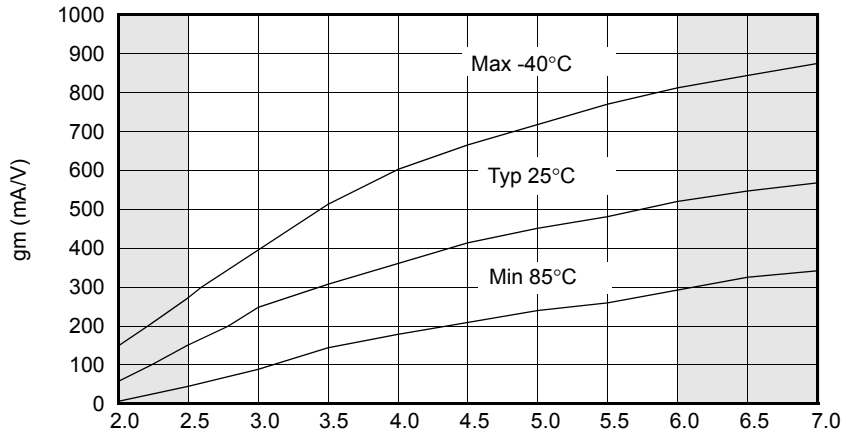
阴影部分在
推荐工作范围以外

图 31-26： LP 振荡器的跨导 (gm) ─ VDD 关系曲线示例



阴影部分在
推荐工作范围以外

图 31-27：XT 振荡器的跨导 (gm) － VDD 关系曲线示例



阴影部分在
推荐工作范围以外

31.3.5 晶体的起振时间

以下各图所示为，在给定晶体 / 电容下，规定工作电压与晶体的起振时间的关系。

图 31-28：典型 XTAL 起振时间 — VDD 关系曲线示例 (LP 振荡模式 , 25°C)

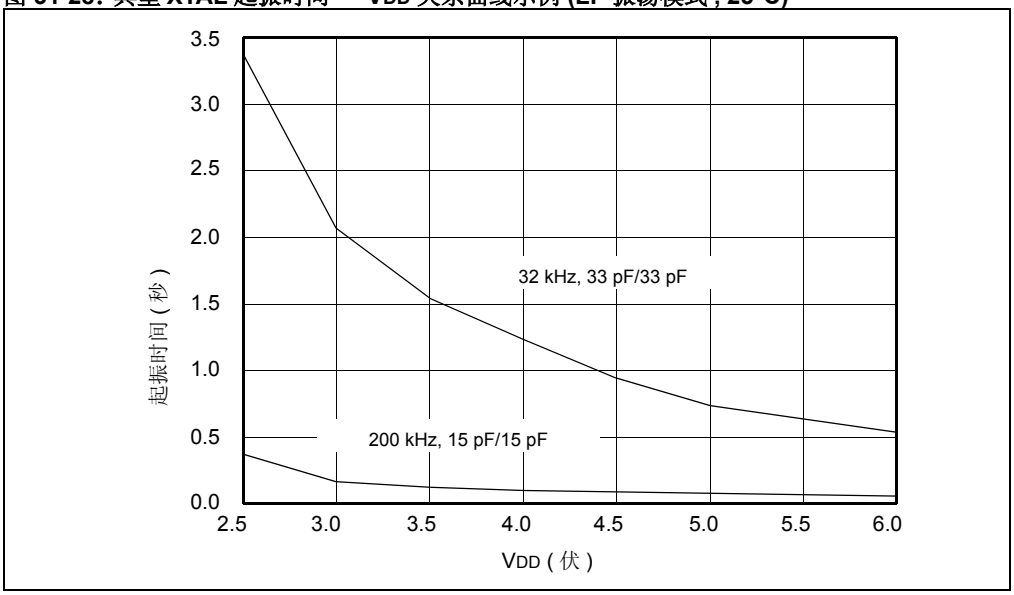


图 31-29：典型 XTAL 起振时间 — VDD 关系曲线示例 (HS 振荡模式 , 25°C)

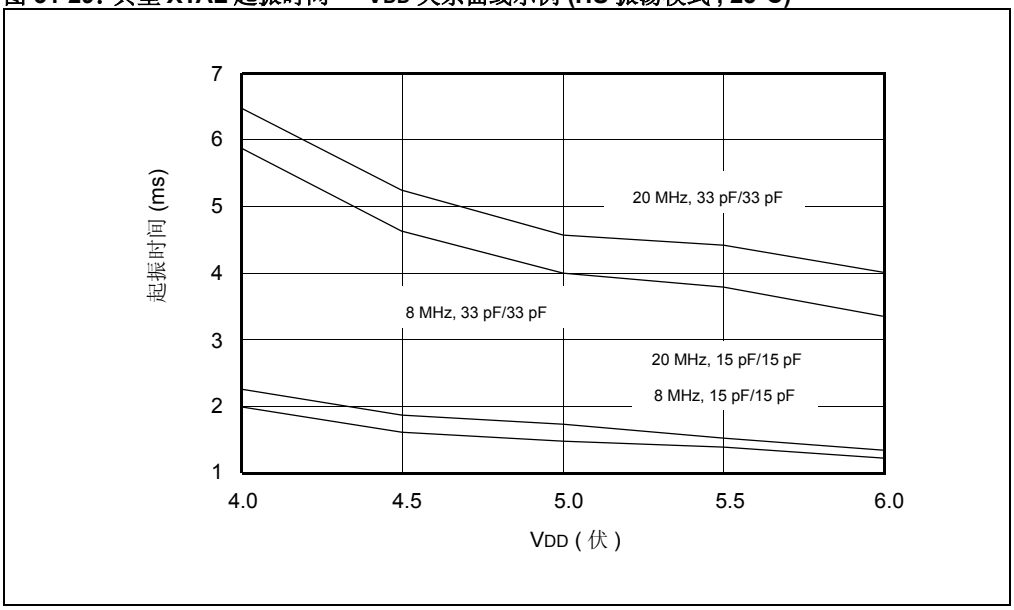
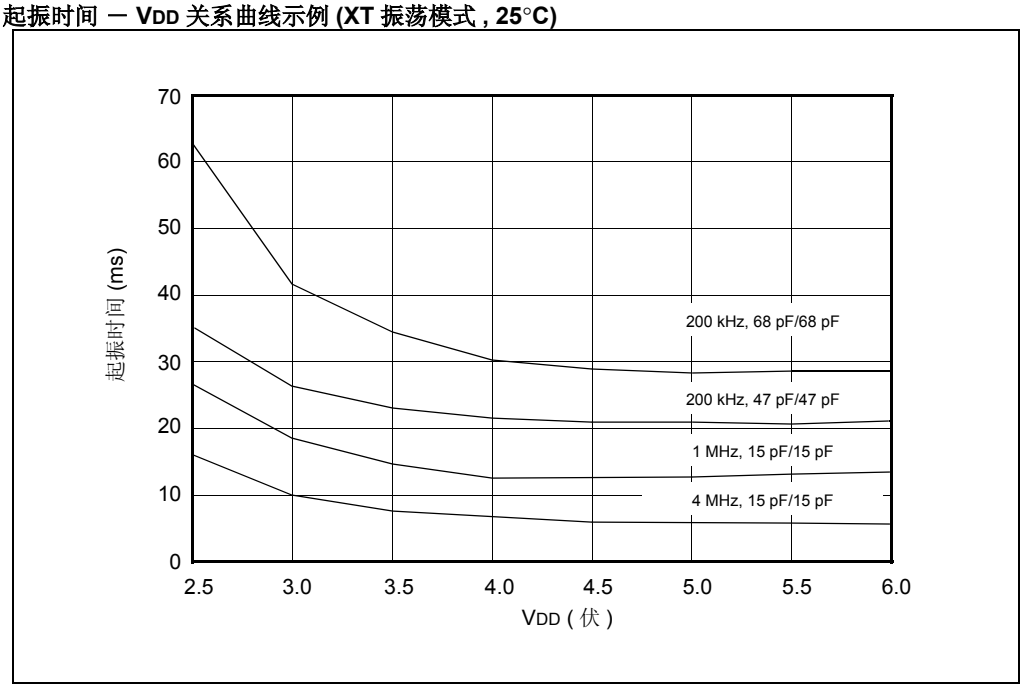


图 31-30：典型 XTAL 起振时间 – VDD 关系曲线示例 (XT 振荡模式 , 25°C)



31.3.6 经测试的晶体及其电容值

下表给出了在本章测试中所使用的晶体频率及其制造商，以及得到最佳特性的电容容量和范围。

表 31-2: 晶体振荡器的电容选型示例

振荡类型	晶体频率	电容器 C1 的范围	电容器 C2 的范围
LP	32 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	200 kHz	47-68 pF	47-68 pF
	1 MHz	15 pF	15 pF
	4 MHz	15 pF	15 pF
HS	4 MHz	15 pF	15 pF
	8 MHz	15-33 pF	15-33 pF
	20 MHz	15-33 pF	15-33 pF
注： 增大电容可以加大振荡器的稳定性，但同时也延长了起振时间。上述数值仅供设计参考。在HS 和 XT 模式下，可能要用到 ***Rs***，以防止对要求低驱动的晶体产生过驱动。由于每个晶体都有其自身的特性，用户应向晶体制造商咨询以获得适当的外接元件参数，或自行校验振荡器的性能。			
使用的晶体：			
32 kHz	Epson C-001R32.768K-A	± 20 PPM	
200 kHz	STD XTL 200.000KHz	± 20 PPM	
1 MHz	ECS ECS-10-13-1	± 50 PPM	
4 MHz	ECS ECS-40-20-1	± 50 PPM	
8 MHz	EPSON CA-301 8.000M-C	± 30 PPM	
20 MHz	EPSON CA-301 20.000M-C	± 30 PPM	

31.3.7 EPROM 存储器的擦除时间示例

一个 EPROM 单元的紫外线擦除时间是由 EPROM 单元的几何大小和制造工艺决定的。表 31-3 给出了不同器件的擦除时间的例子。

表 31-3: 典型 EPROM 擦除时间推荐值示例

器件	波长 (埃)	强度 ($\mu W/cm^2$)	到紫外灯的距离 (英寸)	典型时间 (1) (分钟)
1	2537	12,000	1	15 - 20
2	2537	12,000	1	20
3	2537	12,000	1	40
4	2537	12,000	1	60

注 1: 如果未达到上述标准，擦除时间将有所不同。

表 31-4: 欲了解某一器件的典型擦除时间，请参见器件数据手册。

31.4 版本历史

版本 A

这是描述器件特性的初始发行版。

第 32 章 开发工具

重点

本章包括下面一些主要内容：

32.1 简介 32-2
32.2 集成开发环境（IDE） 32-3
32.3 MPLAB 软件语言支持 32-6
32.4 MPLAB SIM 软件模拟器 32-8
32.5 MPLAB 硬件仿真器支持 32-9
32.6 MPLAB 编程器支持 32-10
32.7 辅助工具 32-11
32.8 开发板 32-12
32.9 针对其它 Microchip 产品的开发工具 32-14
32.10 相关应用笔记 32-15
32.11版本历史 32-16

32.1 简介

Microchip 提供了一系列高度集成的开发工具，可以简化应用程序的开发过程。这些工具分为核心开发工具和附属工具。

基本开发工具主要有：

- MPLAB® 集成开发环境，包括功能齐全的编辑器。
- 语言产品
  - MPASM™ 汇编器
  - MPLAB-C C 编译器
- MPLAB-SIM 软件模拟器
- 实时在线仿真器
  - PICMASTER®/PICMASTER® CE 仿真器，具有全功能的跟踪检查和断点调试能力
  - ICEPIC™ 低成本仿真器，带有断点调试功能
- 器件编程器
  - PRO MATE® II 通用编程器
  - PICSTART® Plus 初级开发编程器

附属工具：

- 其它软件编程工具
  - fuzzyTECH®–MP 模糊逻辑开发系统
  - MP-Driveway 应用代码生成器
- 开发板
  - PICDEM™-1 低成本型演示板
  - PICDEM™-2 低成本型演示板
  - PICDEM™-3 低成本型演示板
  - PICDEM™-14A 低成本型演示板

MPLAB® 的最小配置包括集成开发环境（IDE）、汇编器（MPASM™）和软件模拟器（MPLAB-SIM）。其它工具可以在安装时添加到 MPLAB 中。这为设计工作提供了一个从源代码的编写、汇编 / 编译、模拟 / 仿真，到器件编程的通用平台。

注：可以从 Microchip 的网站或 BBS 上免费下载到 MPLAB® 的最新版本。

除了 Microchip，还有许多第三方的厂商提供开发工具。Microchip 的第三方厂商手册中，给出了第三方厂商和及其工具的介绍。

32.2 集成开发环境 （IDE）

核心开发工具都运行在 MPLAB® 集成开发环境下，这使得这些开发工具看起来是一个整体，减少了学习新工具界面的麻烦。MPLAB IDE 集成了下述的开发功能：

- 源代码编辑
- 项目管理
- 从汇编或 C 语言生成机器代码
- 器件模拟
- 器件仿真
- 器件编程

MPLAB® 是基于 PC 的 Windows® 3.x 应用软件。在 Windows 95 系统下进行过严格的测试，推荐在 Windows 3.x 或 Windows 95 操作系统下运行。

使用这个综合的开发工具包，就可以在 MPLAB 环境中完成一个项目的开发全过程。

Windows is a registered trademark of Microsoft Corporation.

32.2.1 MPLAB

MPLAB IDE 软件给软件开发带来的便利是以往 8 位单片机开发环境所未曾有的，MPLAB 是基于 Windows 平台的应用软件，包括：

- 功能齐全的编辑器
- 三种工作模式
  - 编辑器
  - 硬件仿真器
  - 软件模拟器
- 项目管理器
- 用户可定制的工具栏和快捷键
- 显示项目信息的状态栏
- 丰富的在线帮助

MPLAB 使你可以：

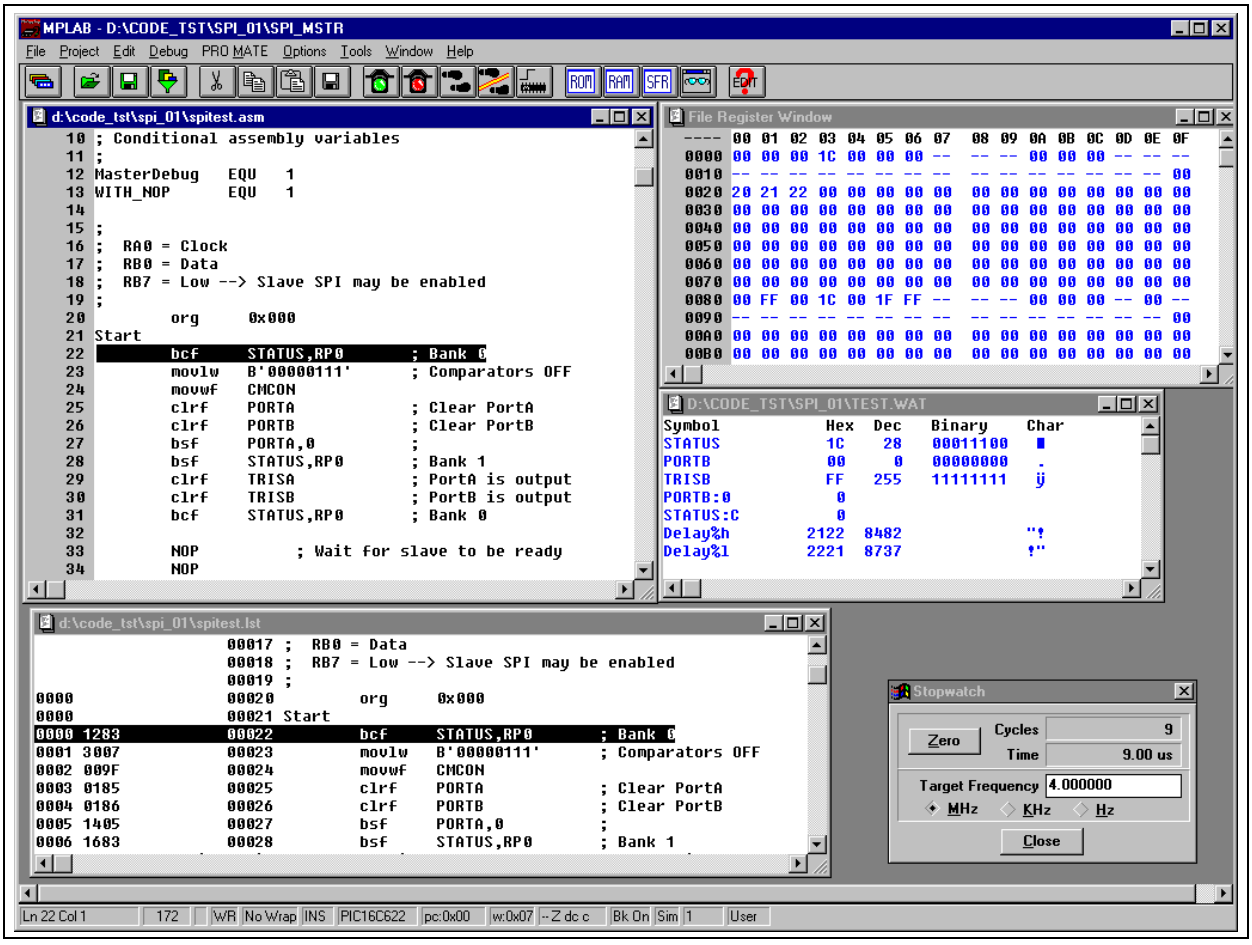
- 编辑源文件，包括：
  - MPASM™ 汇编语言源文件
  - MPLAB-C C 语言源文件
- 点击一次就可以完成汇编 （或编译）并将代码下载到 PIC16/17 系列器件中（自动更新所有项目信息）
- 可使用如下各项进行调试：
  - 源文件
  - 绝对列表文件
  - 程序存储器中机器码
- 同一 台 PC 上可运行 4 个仿真器
- 运行或单步执行
  - 程序存储器中机器码
  - 源文件
  - 绝对列表

Microchip 的软件模拟器 MPLAB-SIM 和 PICMASTER® 仿真器运行在同一个平台上。用户只需要学会一种工具的使用即可，因为软件模拟器和全功能仿真器的功能相同。

图 32-1 是一个项目的典型 MPLAB 界面。有下面一些特点：

- 有多种可选择的工具栏，还可以用户自定义
- 状态、模式和帮助信息显示在底部状态栏中
- 多个窗口，如：
  - 源代码窗口
  - 源代码列表 （对 C 程序很有用）
  - 数据寄存器窗口 （RAM）
  - Watch （观察）窗口 （查看特定寄存器的内容)
  - 进行时间 / 周期计算的 Stop watch 窗口
- 对编程器的支持 （图中所示为 PRO MATE® 编程器的下拉菜单)

图 32-1： MPLAB® 项目窗口



32.3 MPLAB 软件语言支持

为使单片机在应用中按照期望运行，需要为单片机编写软件程序。需要用两种编程语言之一来编写这个程序。目前 MPLAB 支持两种 Microchip 的语言产品。

- Microchip 汇编器 （MPASM™)
- Microchip C 编译器 (MPLAB-C)
- 其它支持公共对象描述 (COD) 的语言工具也可以在 MPLAB 中使用

32.3.1 汇编器 （MPASM™）

MPASM 通用宏汇编器是一种基于 PC 平台的符号汇编器，它支持 Microchip 所有系列的单片机。

MPASM 提供完整的宏功能、条件汇编，以及几种源文件和列表文件格式。它产生不同的目标代码格式，以支持 Microchip 的多种开发工具以及第三方的编程器。

MPASM 支持在 Microchip 通用仿真系统 (PICMASTER) 上进行符号调试。

MPASM 的如下特征有助于为具体应用开发软件：

- 为 Microchip 的所有单片机提供汇编源代码到目标代码的转换。
- 宏汇编功能。
- 产生使用 Microchip 仿真系统进行符号调试所需要的所有文件 （目标文件、列表文件、符号文件和专用文件）。
- 支持十六进制 (默认)、十进制和八进制的源文件和列表文件格式。

MPASM™ 提供丰富的指示语言来支持 PICmicro® 单片机的编程，这使汇编源代码的开发更快、更具可维护性。

32.3.2 C 编译器 (MPLAB-C)

MPLAB-C 是针对 Microchip PICmicro 系列单片机的 C 编译器。它具有强大的集成功能，且比其它编译器使用起来更方便。

为便于源代码调试，编译器提供可与 MPLAB® IDE 的存储显示窗口、观察窗口和文件寄存器窗口显示相兼容的符号信息。

32.3.3 MPLINK™ 链接器

MPLINK是针对 Microchip的C编译器 MPLAB-C和可重定位汇编器MPASM™的链接器。MPLINK 是随 MPLAB-C v2.00 引入的，只能用于该版本或更高版本。

MPLINK 使得可以利用 MPLAB-C 和 MPASM™ 产生模块化和可重用的代码。对链接过程的控制是通过一个链接器 “脚本”文件和命令行选项完成的。

MPLINK 把 MPLAB-C 或 MPASM 产生的多个输入目标模块合并成一个可执行文件。数据的实际地址和函数的位置在执行 MPLINK 时分配。所以，可以指定 MPLINK 把代码和数据放到指定存储区的某个位置，而不必事先明确地指定代码和数据的具体物理地址。

MPLINK 链接器根据目标器件里可用的 ROM 和 RAM 区，分析所有的输入文件，并设法将应用程序放到 ROM 中，数据变量放到可用的 RAM 中。如果代码或变量太多而放不下， MPLINK 将给出错误消息。

MPLINK 可以灵活地指定哪些数据存储块是可重用的，这样不同的函数就能够共享有限的 RAM 空间（这些函数不能互相调用，并且执行一个函数时不能使用执行另外一个函数后存储在此 RAM 块中的变量值）。

32.3.4 MPLIB™ 库管理器

MPLIB™ 管理着由 MPASM v2.0、 MPASMWIN v2.0、MPLAB-C v2 或更高版本创建的 COFF 目标模块。

MPLIB™ 对库文件的创建和修改进行管理。一个库文件是指保存在一个文件中的目标模块的集合。创建库文件的原因如下：

- 库文件更易于链接。因为库文件能够包含许多目标文件，在链接时就可以只使用库文件名，而不是各种不同的目标文件名。
- 库有助于缩减代码大小。因为链接器仅仅使用一个库中所需要的目标文件，而不是库中所有的目标文件。
- 库更易于项目的维护，如果在项目中包含一个库文件，那么是否调用该库不会改变链接过程。
- 库有助于表示不同目标模块的不同用途。因为库能够 把几个相关的目标模块集合在一起，库文件的用途通常比各个目标模块的用途更容易理解。例如名字为 “math.lib”的库文件，其用途就比 “power.o”、 “ceiling.o”和 “floor.o”的用途更明了。

32.4 MPLAB SIM 软件模拟器

软件模拟器是一种用来评估 Microchip 产品和设计的免费工具。它对软件调试（特别是算法）非常有帮助。根据设计项目复杂性的不同，对比硬件仿真器，它在开发时间 / 成本方面有优势。

在开发过程有许多工程师参与的项目中，模拟器和仿真器结合使用可以降低成本，并能快速地对复杂困难的问题进行调试。

MPLAB SIM 软件模拟器可在指令级对 PICmicro 系列单片机进行模拟调试。对于任一条指令，用户都可以进行数据区的检查或修改，也可向任何引脚提供外部激励。输入 / 输出的进制可以由用户设定，程序运行方式有连续运行、单步运行、运行到断点或跟踪模式。

MPLAB SIM 支持使用 MPLAB-C 和 MPASM™ 进行符号调试。软件模拟器提供了一种低成本和灵活的开发和调试方法，且不需实验室环境，是一种优秀的多项目软件开发工具。

32.5 MPLAB 硬件仿真器支持

Microchip 提供两种硬件仿真器，一种高端版本 (PICMASTER) 和一种低端版本 (ICEPIC™)。两个版本都有很好的性价比，要依据你所需求的功能来选择仿真器。对于用 Microchip 单片机做几个项目的用户（或使用高档单片机的用户），选用 PICMASTER 虽然价格略高，但 PICMASTER 有高级的断点和跟踪功能，会节省开发时间。

32.5.1 PICMASTER：高性能通用在线仿真器

PICMASTER 通用在线仿真器为产品开发工程师提供了完善的单片机设计工具，适用于各种系列的单片机（包括低档、中档和高档系列的单片机）。PICMASTER 在 MPLAB 集成开发环境（IDE）下运行，可在同一个环境下进行编辑、编译、下载以及源代码调试。

通过更换不同的仿真头，可以很容易地重新配置仿真器，以仿真不同的处理器。这种通用结构，允许 PICMASTER 通过扩展来支持所有新型 Microchip 单片机。

PICMASTER 仿真器系统设计为一种实时仿真系统，并具有更昂贵的开发工具才有的高级功能。

符合 CE 标准的 PICMASTER 版本适合欧盟国家使用。

32.5.2 ICEPIC™：低成本 PIC16CXXX 在线仿真器

ICEPIC™ 是针对 Microchip 8 位一次性编程 （OTP）低档和中档系列单片机的一种廉价在线仿真器。

ICEPIC™ 的用户软件可以在运行 Windows 3.x 的 286-AT® 至 Pentium™ 的 PC 兼容机上使用。ICEPIC™ 的特点是实时仿真，并可在 MPLAB® 环境下工作。

ICEPIC™ 由 Neosoft 公司设计，授权 RF Solutions 生产。从 RF Solutions 也可直接获得其它仿真器的解决方案。

32.6 MPLAB 编程器支持

Microchip 提供两种级别的器件编程器。对于多数工作室，使用 PICSTART 就足够了。当对系统的合格性要求严格时，可以使用 PRO MATE II，因为 PRO MATE II 可以在 VDD 的最大值和最小值下校验程序存储器，以获得最大的可靠性。

32.6.1 PRO MATE® II：通用器件编程器

PRO MATE II 通用编程器是一种功能全面的编程器，能够在单机模式和 PC 主机模式下运行。PRO MATE II 可以在 MPLAB 下运行或在 DOS 命令行下运行。

PRO MATE II 拥有 VDD 和 VPP 两个可编程电源，可以在 VDD 为最小值和最大值时校验程序存储器，以获得最大的可靠性。它有一个 LCD 显示器用来显示错误信息，有用来输入命令的按键，以及一个可以支持各种封装类型的可分离插座。在单机模式下，PRO MATE II 能对低、中、高档器件进行读、校验和编程，还能设置配置位和代码保护位。PRO MATE II 编程器也支持 Microchip 的串行 EEPROM 和 KEELOQ 安全器件。

独立的在线串行编程 (ICSP) 模块适用于生产环境下的批量编程。关于具体应用要求，请参阅编程模块的文档。

32.6.2 PICSTART® Plus 低成本开发工具包

PICSTART Plus 编程器是一种易操作、低价位的开发编程器。它通过 COM (RS-232) 端口和 PC 相连。MPLAB IDE 软件使得该编程器的使用简便、高效。建议不要用 PICSTART Plus 作为生产编程器，因为它不在最小工作电压 VDDMIN 和最大工作电压 VDDMAX 下，对程序存储器进行校验。

PICSTART Plus 支持所有低、中、高档单片机的编程。对于多于 40 引脚的器件，需要一个转接插座。DIP 封装可以直接支持，其它封装类型需要有转接插座。

32.7 辅助工具

Microchip 致力于为客户提供广泛的解决方案。有些产品可能超出了传统开发工具的范畴，包括更高级的功能，例如高级语言，模糊逻辑或可视化编程等。这些工具视为辅助工具，并能直接从 Microchip 或其它厂商获得。第三方厂商指南中给出了这些工具的供应商列表。

32.7.1 *fuzzy*TECH-MP 模糊逻辑开发系统

*fuzzy*TECH-MP 模糊逻辑开发工具共有两种版本，一种是低成本的入门版 MP Explorer，它主要是帮助设计者能全面了解模糊逻辑系统的设计知识。另一种是全功能版 *fuzzy*TECH-MP，它用于设计更复杂的系统。

两个版本都带有 Microchip 的 *fuzzy*LAB™ 演示板，用于做一些模糊逻辑系统实现的实验。

32.7.2 MP-DriveWay™ — 应用代码生成器

MP-DriveWay 是一种简单易用的、基于 Windows 的应用代码生成器。利用 MP-DriveWay，你能对 PIC16/17 器件的所有外设进行可视化配置，并且通过点击鼠标就可以产生 C 语言形式的初始化及功能代码模块。产生的代码与 Microchip 的 MPLAB-C C 编译器完全兼容，而且是高度模块化的，易于与自己编写的代码集成在一起。

32.7.3 第三方指南

还需要其它的产品吗？ Microchip 大力支持和鼓励第三方提供产品。由 Microchip 出版的 “第三方指南”，提供了 100 多家公司和 200 多个产品的丰富信息：

- 公司
- 产品
- 联系信息
- 咨询

这些产品包括仿真器、器件编程器、批量编程器、语言产品以及其它工具解决方案。

32.8 开发板

开发板对特定器件的功能进行演示。根据对器件功能和操作的不同评估需求，也可对演示程序进行修改。

32.8.1 PICDEM™-1 低成本的 PIC16/17 演示板

PICDEM-1 是一种可以对几个型号的 Microchip 单片机进行演示的简易演示板，它所支持的单片机包括：PIC16C5X（PIC16C54 到 PIC16C58A）、PIC16C61、PIC16C62X、PIC16C71、PIC16C710、PIC16C711、PIC16C8X、PIC17C42A、PIC17C43 和 PIC17C44。它包含运行基本演示程序所必需的软硬件。用户通过 PRO MATE II 或PICSTART-Plus 编程器，可以对PICDEM-1 板上的单片机样片进行编程，便于固件测试。用户也可以把PICDEM-1 板和PICMASTER® 仿真器相连，把固件下载到仿真器中进行测试。在板子的实验布线区，可以添加一些其它硬件。该板的资源包括：一个 RS-232 接口、一个用于仿真模拟量输入的电位器、按钮式开关和与 PORTB 口相连的 8 个发光二极管。

32.8.2 PICDEM™-2 低成本的 PIC16CXXX 演示板

PICDEM-2 是一种支持 PIC16C62、PIC16C63、PIC16C64、PIC16C65、PIC16C72、PIC16C73 和 PIC16C74 单片机的简易演示板，它包括了必备的软件和硬件，以运行基本演示程序。通过 PRO MATE II 或PICSTART-Plus 编程器，用户可以对PICDEM-2 板上的单片机样片进行编程，并对固件进行测试。可以用 PICMASTER® 仿真器来测试 PICDEM-2 板上的固件。在板子的实验布线区，可以添加一些其它硬件该板的资源包括：一个 RS-232 接口、一个用于仿真模拟量输入的电位器、按钮式开关、一个用来演示 I^2C 总线的串行 EEPROM 和分别用于连接 LCD 模块和键盘的插口。

32.8.3 PICDEM™-3 低成本的 PIC16CXXX 演示板

PICDEM-3 是一种简单演示板，支持 PLCC 封装的 PIC16C923 和 PIC16C924，它也支持未来带有 LCD 模块的 44 引脚 PLCC 封装单片机。它包括了必需的软件和硬件，以运行基本演示程序。通过 PRO MATE II 编程器或带有转接插座的PICSTART Plus，用户可对 PICDEM-3 板上的单片机进行编程，并对固件进行测试。 可以用 PICMASTER® 仿真器来测试 PICDEM-3 板上的固件。在板子的实验布线区，可以添加一些其它硬件。该板的资源包括：一个 RS-232 接口、一个用于仿真模拟量输入的电位器、按钮式开关、一个热敏电阻，一个外部 LCD 模块接口，一个键盘接口。PICDEM-3 板也提供了一个 LCD 面板，它具有 4 个公共端和 12 段，可以用来显示时间、温度和星期。PICDEM-3 还提供另外一个 RS-232 接口和一套 Windows 3.1 应用软件，用以在 PC 机上显示解复用的LCD信号。一个简单的串行接口允许用户为LCD信号建立一个硬件解复用器。

32.8.4 PICDEM™-14A 低成本的 PIC14C000 演示板

PICDEM-14A 演示板是一个用来评估 PIC14C000 混合信号单片机的通用平台。该板使用 PIC14C000 测量电位器和片内温度传感器上的电压。这些电压值由片内带隙参考电压进行校准。最后，电压和温度数据发送到 RS-232 端口。该数据可以通过一个终端仿真程序来显示，例如 Windows 终端。用户通过演示板上的一些外设模块，可以在一个 LCD 面板上显示数据，可以对串行 EEPROM 进行读写操作，或自己设计制作与单片机接口的电路。

32.9 针对其它 Microchip 产品的开发工具

32.9.1 SEEVAL® 评估和编程系统

SEEVAL® 串行 EEPROM 设计工具包支持 Microchip 所有的 2 线和 3 线串行 EEPROM。该工具包包含了用于读、写、擦除或对 Microchip 的 SEEPROM 产品（包括 Smart Serials™ EEPROM 和安全串行 EEPROM）的特殊功能进行编程的所有必备工具。Total Endurance™ 软件用来进行折衷分析和可靠性计算。 使用 Total Endurance 工具包能够显著缩短产品开发周期，并能设计出一个更为优化的系统。

32.9.2 KEELOQ® 评估和编程工具

KEELOQ 评估和编程工具支持 Microchip 的 HCS 安全数据产品。HCS 评估工具包包括一个 LCD 显示器，用来显示代码的变化； 一个解码器，用于对传送信号解码；以及一个编程接口，用来对测试发送器编程。

32.10 相关应用笔记

本部分列出了与本章内容相关的应用笔记。这些应用笔记并非都是专门针对中档单片机系列而写的（即有些针对低档系列，有些针对高档系列），但是其概念是相近的，通过适当修改并受到一定限制即可使用。目前与 Microchip 开发工具相关的应用笔记有：

标题	应用笔记 #
Air Flow using Fuzzy Logic	AN600

32.11 版本历史

版本 A

这是描述 Microchip 开发工具的初始发行版。

第 33 章 代码开发

主要内容

目前没有这方面的资料。请注意查看 Microchip 的网站，关注中档单片机系列参考手册代码开发章节的 B 版本。

33.1 版本历史

版本 A

这是描述 PICmicro® 单片机代码开发的初始发行版。

第 34 章 附录

目录

本章包括下面一些主要内容：

附录 A：I²C™ 概述 34-2
附录 B：LCD 玻璃基板生产商 34-11
附录 C：改进的器件特性 34-13
附录 D：版本历史 34-19

附录 A： I^2C™ 概述

本章对内部互联（Inter-Integrated Circuit， I^2C) 总线进行了概述，附录 A.2 讨论了 SSP 模块工作在 I^2C 模式时的运行情况。

I^2C 总线是双线制的串行接口。原始规范 （或标准模式）的数据传输速率为 100 Kbps，增强规范支持快速模式，数据传输速率为 400 Kbps。如果总线以较慢器件的速度运行，标准模式和快速模式的器件可在同一总线上工作。

I^2C 接口用完善的协议来确保发送和接收数据的可靠性。当发送数据时，一个器件作为主机，控制总线传输和产生时钟信号，而其它器件作为从机。除了通用呼叫支持外，所有从机协议部分都由 SSP 模块的硬件实现，而主机协议部分需要由 PIC16CXX 的软件实现。 MSSP 模块可以实现所有的 I^2C 主机协议、通用呼叫地址和高达 1 Mbps 的数据传输速率。 Microchip 的某些串行 EEPROM 支持 1 Mbps 的数据传输速率。表 A-1 对 I^2C 总线中的术语进行了说明。

在 I^2C 接口协议中，每个器件都有一个地址。当主机开始一次数据传输时，它首先发送要 “通话”的器件的地址。所有器件都 “接听”该地址，看是否与自己的地址匹配。在该地址中，有一位指定主机是读还是写从机。在数据传输期间，主机和从机总是处在相反的工作模式 （发送器 / 接收器），也就是说主机和从机可以是下面两种关系之一：

- 主机 － 发送器，从机 － 接收器
- 从机 － 发送器，主机 － 接收器

在这两种情况下，都是由主机产生时钟信号。

为了实现总线的 “线与”功能，时钟线 （SCL）和数据线 （SDA）的输出级都必须采用漏极开路或集电极开路。所以需要外接上拉电阻，以保证总线在没有器件把其拉低时为高电平。总线上连接的器件数目仅受到最大总线负载要求 （400 pF）和寻址能力的限制。

I^2C is a trademark of Philips Corporation.

A.1 启动和停止数据传输

在总线不传输数据时 (空闲时)，时钟线 (SCL) 和数据线 (SDA) 都通过外部的上拉电阻拉为高电平。 由启动和停止条件决定数据传输的启动和停止。启动条件定义为 SCL 保持高电平而 SDA 由高变低；停止条件定义为 SCL 保持电平而 SDA 由低变高。图 A-1 给出了启动和停止条件的波形图。主机产生这些条件来控制数据传输的启动和停止。根据启动和停止条件的定义，在数据传输过程中， SDA 线上数据的改变只能在 SCL 为低电平时完成。

图 A-1： 启动和停止条件

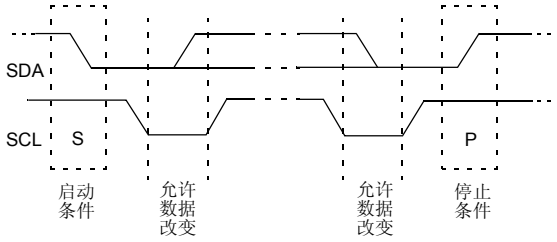


表 A-1： I^2C^{TM} 总线术语

术语	描述
发送器	发送数据到总线的器件。
接收器	从总线上接收数据的器件。
主机	启动传输，产生时钟和结束传输的器件。
从机	由主机寻址的器件。
多主机	系统中有一个以上的主机。这些主机，可同时尝试对总线进行控制而不会破坏消息。
仲裁	保证只有一个主机控制总线的过程，这可确保传输的数据不会被破坏。
同步	使两个以上器件的时钟信号同步的过程。

A.2 对 I²C™ 器件的寻址

有两种地址格式。最简单的一种是 7 位地址码加一位读 / 写控制位（图 A-2)。另一种复杂些，是 10 位地址码加一位读 / 写控制位 (图 A-3)。对于 10 位地址格式，要传输两个字节。第一个字节的前五位用来指定 10 位地址格式，此外该字节还包括地址的高两位和读 / 写控制位。第二个字节是其余的低 8 位地址。

图 A-2： I²C™ 7 位地址格式

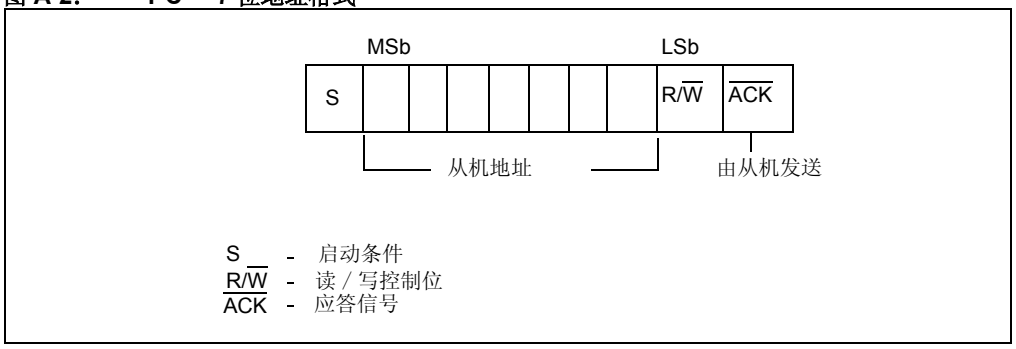
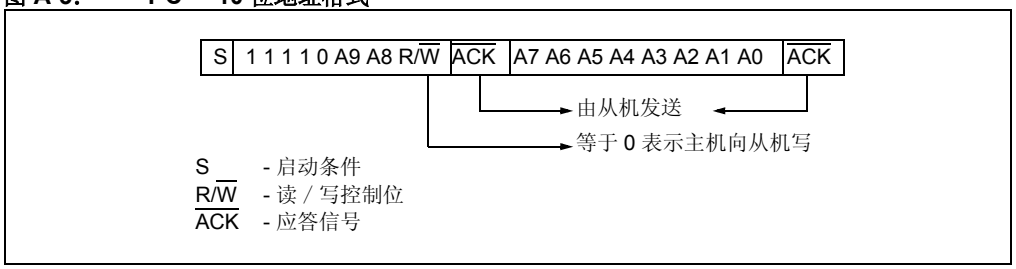


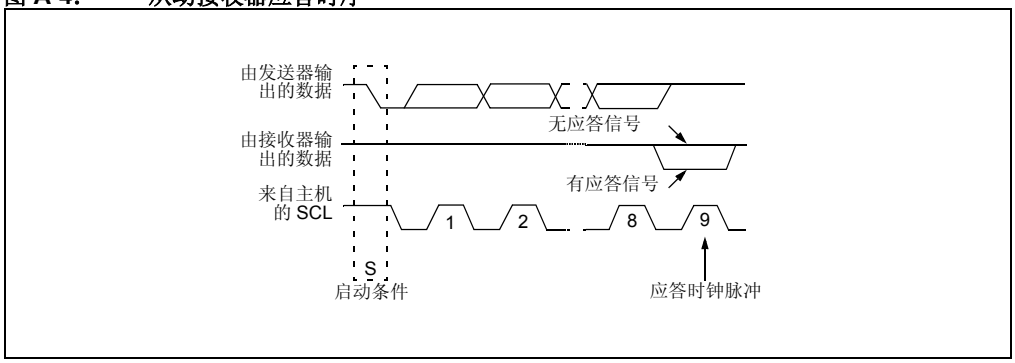
图 A-3： I²C™ 10 位地址格式



A.3 传输应答

所有数据必须以字节为单位进行发送，而每次发送的字节数没有限制。从动接收器每收到一字节数据后，就发出一个应答位 (ACK) (图 A-4)。如果从动接收器没有对从机地址或接收到的数据发应答信号，则主机必须停止传输。从机必须释放 SDA 线，使其为高电平，以使主机能产生停止条件 （图 A-4）。

图 A-4： 从动接收器应答时序



如果主机接收数据 （主接收器），则除了最后一个传送字节，主机在收到每一个数据字节后都产生一个应答信号。如果主机没有发出应答信号，则通知从动发送器数据传输结束。此时从机释放 SDA 线以使主机能够产生停止条件。主机也可以在应答脉冲期间产生停止条件以终止数据传输。

如果从机需要延迟下一个字节的传输，可以通过保持 SCL 线为低电平来迫使主机进入等待状态。当从机释放 SCL 线时，数据传输又可继续。这就允许从机在时钟启动前先移入接收到的数据或取出需要发送的数据。这种 “等待状态” 技术也可以用在对 “位” 的处理上，如图 A-5。

图 A-5： 数据传输的等待状态

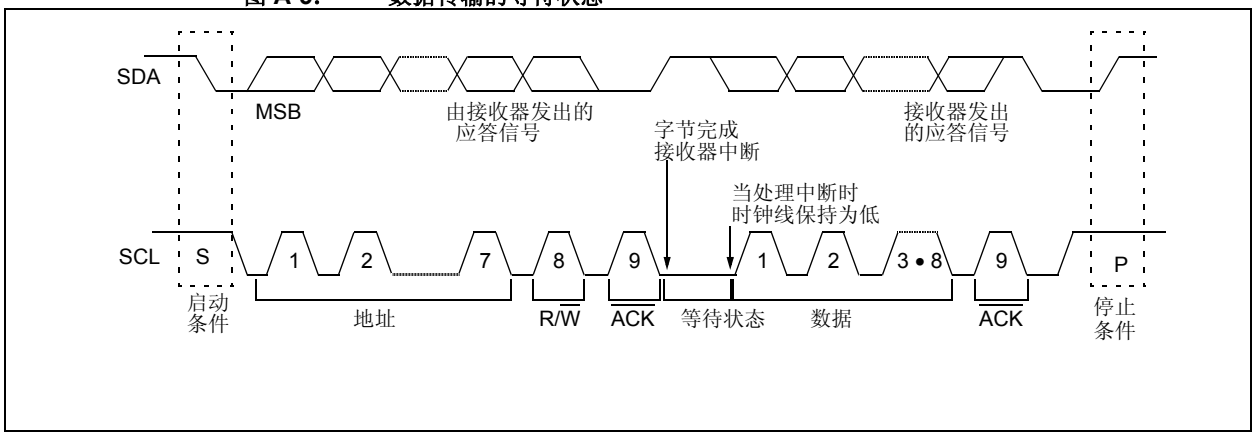


图 A-6 和图 A-7 给出了主发送器和主接收器的数据传输序列。

图 A-6： 主发送器序列

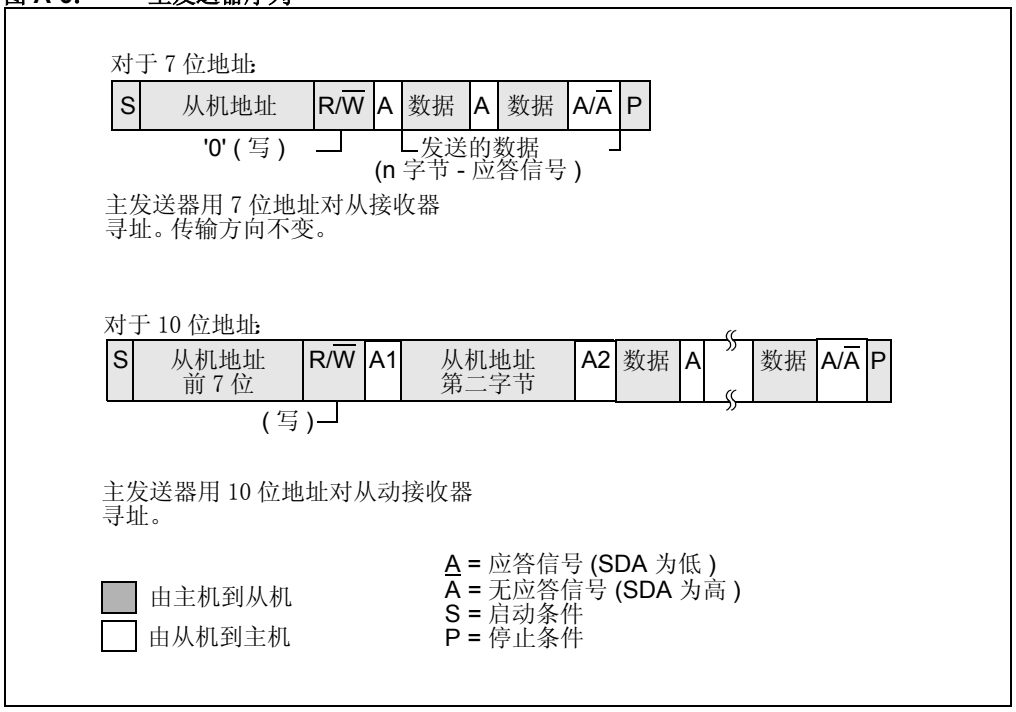
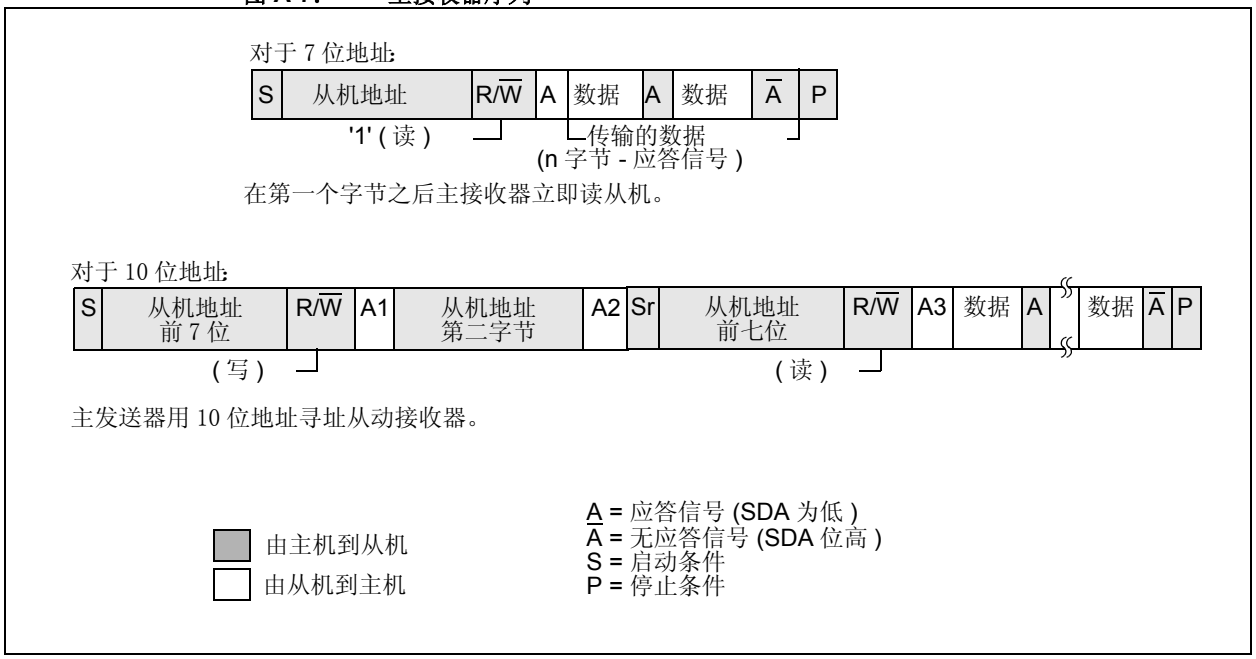
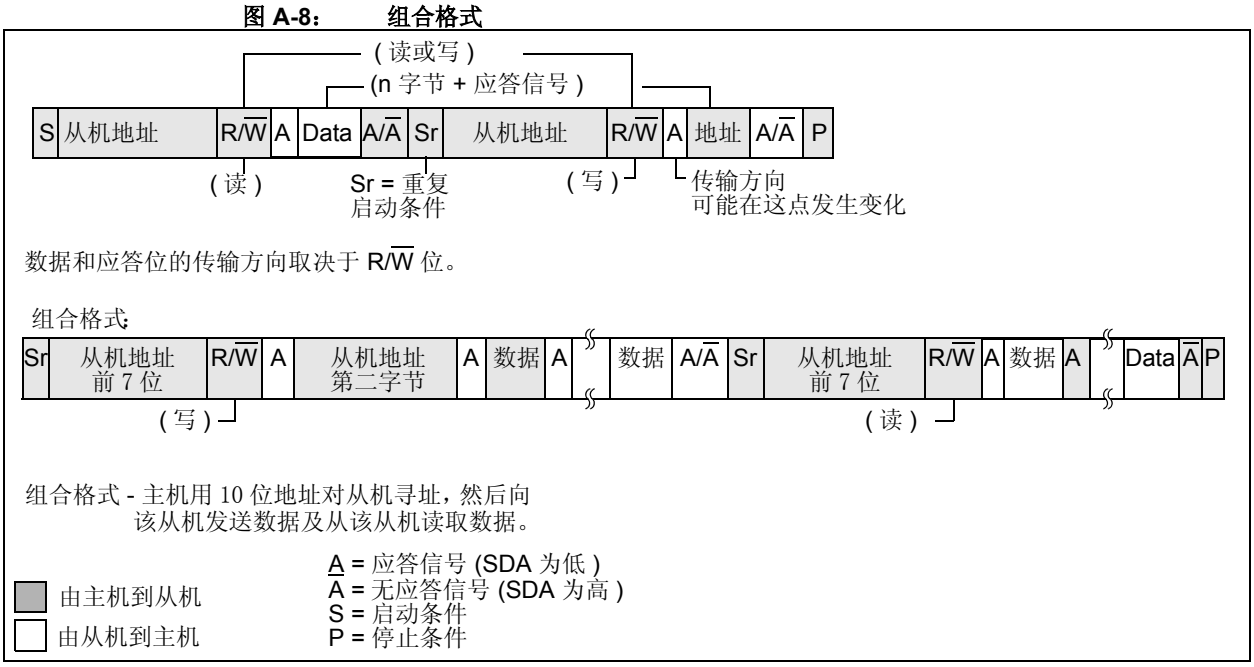


图 A-7： 主接收器序列



如果主机不想放弃总线控制权（发出“停止”条件），必须产生一个“重复启动”条件。这个条件和启动条件是一样的（在 SCL 保持为高时， SDA 由高变低），只是发生在数据传输应答脉冲产生之后（而不是在总线空闲状态时）。这样主机就可以把“命令”发送给从机，然后接收所需要的信息或对另外一个从机进行寻址。该序列图如图 A-8 所示。

图 A-8： 组合格式



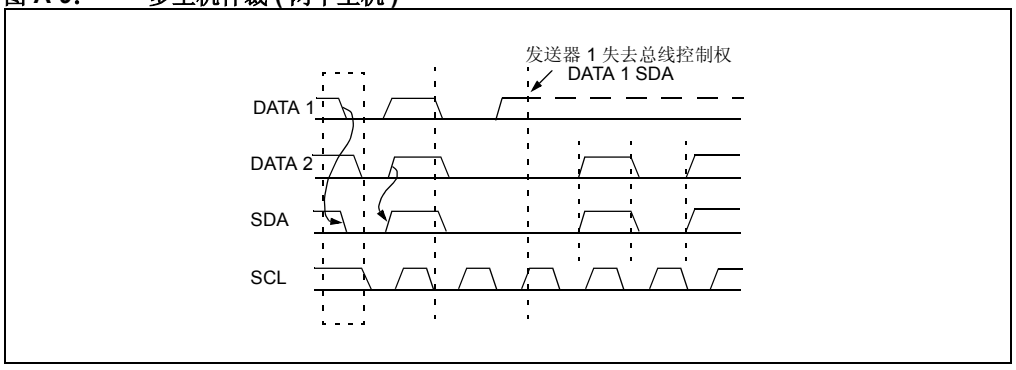
A.4 多主机操作方式

I^2C 协议允许在一个系统中有多个主机存在，这称为多主机操作方式。当两个或两个以上的主机试图同时传输数据时，就需要进行总线仲裁和时钟同步。

A.4.1 总线仲裁

SCL 线为高电平时，在 SDA 线上进行总线仲裁。当主机发送 1 到 SDA，另一个主机发送 0 到 SDA 时，则前者失去总线控制权并关闭其数据输出级 (图 A-9)。失去总线控制权的主机将继续发送时钟脉冲，直到当前数据字节传送结束。如果多个主机对同一器件寻址，则会继续进行总线仲裁。

图 A-9： 多主机仲裁 (两个主机)



兼有从机功能且已经失去总线控制权的主机，必须立即切换到从接收器模式，因为已控制总线的主机可能对它进行寻址。

以下几种情况不允许进行仲裁：

- 出现重复启动条件时
- 停止条件和数据位同时出现时
- 重启动条件和停止条件同时出现时

编程时要对这几种状态特别考虑，以确保这几种状态不会发生。

A.4.2 时钟同步

时钟同步在总线仲裁开始后发生。这由 SCL 线上的 “线与” 连接实现。当 SCL 线的电平由高变低时，相关的器件就开始对其时钟低电平时间进行计算。一旦某个器件的 SCL 引脚变为低电平，那么在其 SCL 引脚变为高电平之前，SCL 线将一直保持为低电平。如果另一个器件的 SCL 引脚处于低电平状态，那么器件时钟由低电平变为高电平不会改变 SCL 线的状态。 SCL 线上低电平的持续时间由低电平时间最长的器件决定，而低电平持续时间较短的器件将首先进入高电平等待状态，直到 SCL 线变为高电平。此时，所有器件开始计算时钟高电平的时间，第一个变为低电平的器件又会把 SCL 拉低。SCL 线上高电平的持续时间由高电平时间最短的器件决定，如图 A-10 所示。

图 A-10： 时钟同步

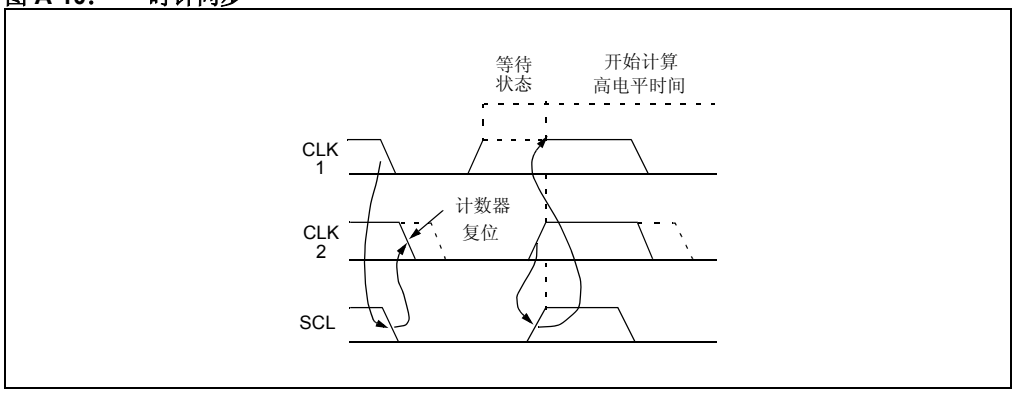


图 A-11： I^2C^{TM} 总线启动 / 停止位时间特性

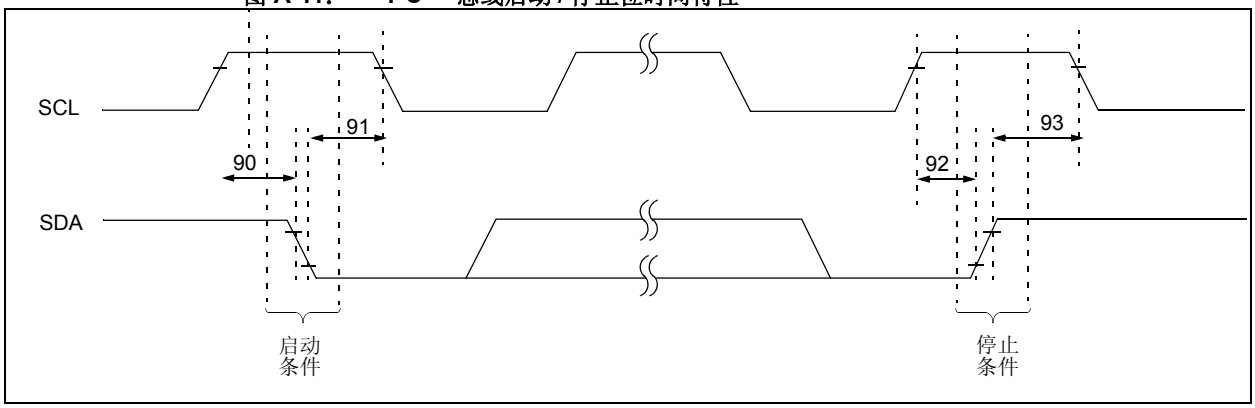


表 A-2： I^2C^{TM} 总线启动 / 停止位时间特性

Microchip 参数编号	符号	特性		最小值	典型值	最大值	单位	条件
90	TSU:STA	启动条件建立时间	100 kHz 模式	4700	—	—	ns	仅和重复启动条件有关
			400 kHz 模式	600	—	—		
91	THD:STA	启动条件保持时间	100 kHz 模式	4000	—	—	ns	在这段时间后，发出第一个时钟脉冲
			400 kHz 模式	600	—	—		
92	TSU:STO	停止条件建立时间	100 kHz 模式	4700	—	—	ns	
			400 kHz 模式	600	—	—		
93	THD:STO	停止条件保持时间	100 kHz 模式	4000	—	—	ns	
			400 kHz 模式	600	—	—		

图 A-12： I^2C^{TM} 总线数据时序特性

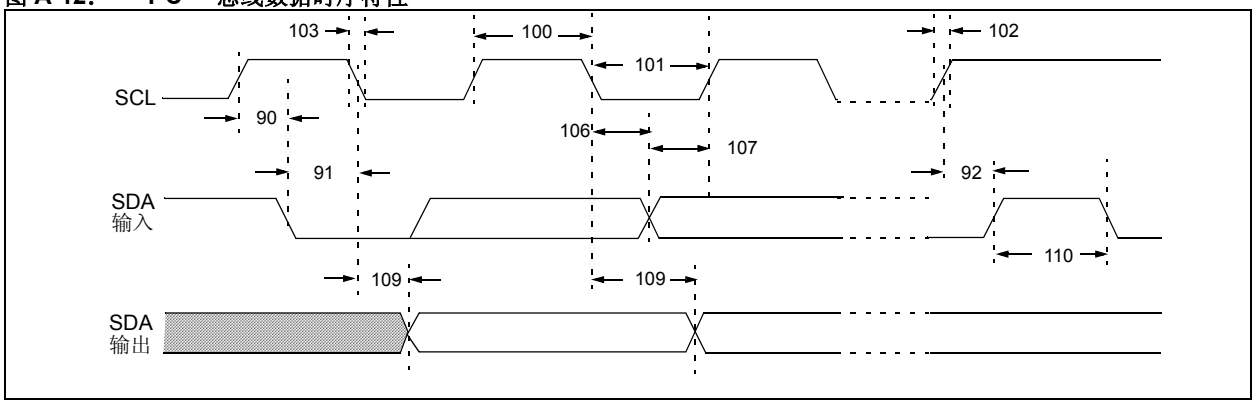


表 A-3： I^2C^{TM} 总线数据时间特性

Microchip 参数编号	符号	特性		最小值	最大值	单位	条件
100	THIGH	时钟高电平时间	100 kHz 模式	4.0	—	µs	
			400 kHz 模式	0.6	—	µs	
101	TLOW	时钟低电平时间	100 kHz 模式	4.7	—	µs	
			400 kHz 模式	1.3	—	µs	
102	TR	SDA 和 SCL 上升时间	100 kHz 模式	—	1000	ns	
			400 kHz 模式	20 + 0.1Cb	300	ns	Cb 规定为 10 到 400 pF
103	TF	SDA 和 SCL 下降时间	100 kHz 模式	—	300	ns	
			400 kHz 模式	20 + 0.1Cb	300	ns	Cb 规定为 10 到 400 pF
90	TSU:STA	启动条件建立时间	100 kHz 模式	4.7	—	µs	仅和重复启动条件有关
			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	在这段时间后，发出第一个时钟脉冲
			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	输出有效时间	100 kHz 模式	—	3500	ns	注 1
			400 kHz 模式	—	1000	ns	
110	TBUF	总线空闲时间	100 kHz 模式	4.7	—	µs	在新的传输开始前总线必须空闲
			400 kHz 模式	1.3	—	µs	
D102	Cb	总线的容性负载		—	400	pF	

注 1：作为发送器，为避免意外地产生启动或停止条件，器件必须提供此内部最小延迟时间，以补偿 SCL 线的下降沿时间 (最小 300ns)。

2：快速模式的 I^2C 总线器件可以用在标准模式的 I^2C 总线系统中，但必须满足数据输入建立时间条件 $T_{SU:DAT} \geq 250$ ns。如果快速模式的器件没有延长 SCL 信号的低电平时间，该数据输入建立时间条件必然满足；如果延长了 SCL 信号的低电平时间，则可以直接向 SDA 线发送下一位数据。根据标准的 I^2C 总线规格，从发送下一位数据到拉高 SCL 线的时间为 T_R max.+$T_{SU:DAT}$ = 1000 + 250 = 1250 ns。

附录 B： LCD 玻璃基板生产商

AEG-MIS
3340 Peachtree Rd. NE Suite 500
Atlanta, GA 30326
TEL: 1-404-239-0277
fax: 1-404-239-0383

All Shore INDS Inc.
1 Edgewater Plaza
Staten Island, NY 10305
TEL: 1-718-720-0018
fax: 1-718-720-0225

Crystaloid
5282 Hudson Drive
Hudson, OH 44236-3769
TEL: 1-216-655-2429
fax: 1-216-655-2176

DCI Inc.
14812 W. 117th St.
Olathe, KS 66062-9304
TEL:1-913-782-5672
fax: 1-913-782-5766

Excel Technology International Corporation
Unit 5, Bldg. 4, Stryker Lane
Belle Mead, NJ 08502
TEL: 1-908-874-4747
fax: 1-908-874-3278

F-P Electronics/Mark IV Industries
6030 Ambler Drive
Mississauga, ON Canada L4W 2PI
TEL: 1-905-624-3020
fax: 1-905-238-3141

Hunter Components
24800 Chagrin Blvd, Suite 101
Cleveland, OH 44122
TEL: 1-216-831-1464
fax: 1-216-831-1463

Interstate Electronics Corp.
1001 E. Bull Rd.
Anaheim, CA 92805
TEL: 1-800-854-6979
fax: 1-714-758-4111

Kent Display Systems
343 Portage Blvd.
Kent, OH 44240
TEL: 1-330-673-8784

LCD Planar Optics Corporation
2100-2 Artic Ave.
Bohemia, NY 11716
TEL: 1-516-567-4100
fax: 1-516-567-8516

LXD Inc.
7650 First Place
Oakwood Village, OH 44146
TEL: 1-216-786-8700
fax: 1-216-786-8711

Nippon Sheet Glass
Tomen America Inc.
1285 Avenue of the Americas
New York, NY 10019
TEL: 1-212-397-4600
fax: 1-212-397-3351

OPTREX America
44160 Plymouth Oaks Blvd.
Plymouth, MI 48170
TEL: 1-313-416-8500
fax: 1-313-416-8520

Phillips Components
LCD Business Unit
1273 Lyons Road, Bldg G
Dayton, OH 45459
TEL: 1-573-436-9500
fax: 1-573-436-2230

Satori Electric
23717 Hawthorne Blvd. 3rd Floor
Torrance, CA 90505
TEL: 1-310-214-1791
fax: 1-310-214-1721

Seiko Instruments USA Inc.
Electronic Components Division
2990 West Lomita Blvd.
Torrance, CA 90505
TEL: 1-213-517-7770
1-213-517-8113
FAX: 1-213-517-7792

Standish International
European Technical Center
Am Baümstuck II
65520 Bad Camberg/Erbach
Germany
TEL: 011 49 6434 3324
fax: 011 49 6434 377238

Standish LCD
W7514 Highway V
Lake Mills, WI 53551
TEL: 1-414-648-1000
fax: 1-414-648-1001

Truly Semiconductors Ltd. (USA)
2620 Concord Ave.
Suite 106
Alhambra, CA 91803
TEL: 1-818-284-3033
fax: 1-818-284-6026

Truly Semiconductor Ltd.
2/F, Chung Shun Knitting Center
1-3 Wing Yip Street,
Kwai Chung, N.T., Hong Kong
TEL: 852 2487 9803
fax: 852 2480 0126

Varitronix Limited Inc.
3250 Wilshire Blvd. Suite 1901
Los Angeles, CA 90010
TEL: 1-213-738-8700
fax: 1-213-738-5340

Varitronix Limited Inc.
4/F, Liven House
61-63 King Yip Street
Kwun Tong, Kowloon
Hong Kong
TEL: 852 2389 4317
fax: 852 2343 9555

Varitronix (France) S.A.R.L.
13/15 Chemin De Chilly
91160 Champlain
France
TEL:(33) 1 69 09 7070
FAX:(33) 1 69 09 0535

Varitronix Italia, S.R.L.
Via Bruno Buozzi 90
20099 Sesto San Giovanni
Milano, Italy
TEL:(39) 2 2622 2744
FAX:(39) 2 2622 2745

Varitronix (UK) Limited
Display House, 3 Milbanke Court
Milbanke Way, Bracknell
Berkshire RG12 1BR
United Kingdom
TEL:(44) 1344 30377
FAX(44) 1344 300099

Varitronix (Canada) Limited
18 Crown Steel Drive, Suite 101
Markham, Ontario
Canada L3R 9X8
TEL:(905) 415-0023
FAX:(905) 415-0094

Vikay America Inc.
195 W. Main St.
Avon, CT 06001-3685
TEL: 1-860-678-7600
fax: 1-860-678-7625

附录 C： 改进的器件特性

随着中档系列单片机架构的成熟，某些模块和功能被加强了，例如：

1. 数据存储器映射
2. SSP 模块
3. A/D 模块
4. 在内核中加入欠压复位功能
5. MCLR 滤波器
6. USART
7. 器件振荡器

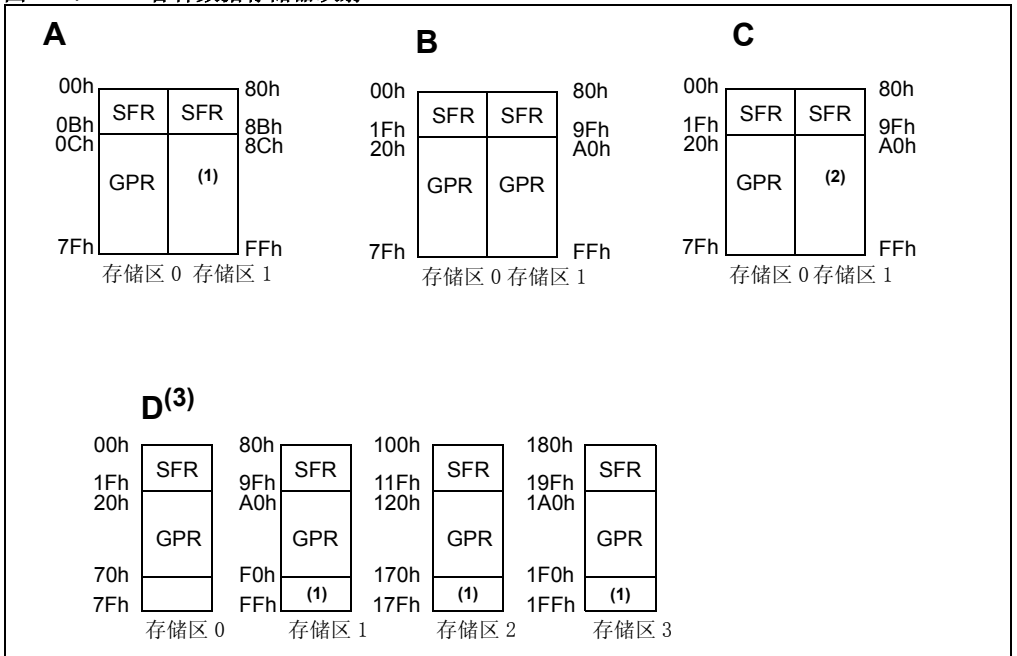
下面的章节将对这些增强功能进行论述。

C.1 数据存储器映射

数据存储器映射图给出了特殊功能寄存器 (SFR) 和通用寄存器 (GPR) 的位置。特殊功能寄存器提供对器件运行的控制，并提供器件的工作状态。而通用寄存器就是通用 RAM。

图 C-1 给出了中档系列单片机的各种存储器映射。存储器映射 **A** 用于最早的中档器件，这些器件的引脚数为 18/20，外设功能有限。当器件需要有更多的 I/O 和更丰富的外设时，使用存储器映射 **B**。存储器映射 **C** 实际上是存储器映射 **B** 的一个子集，但在中断保护现场时需要编写额外的软件，这是因为在存储区 1 内没有通用寄存器。为了减小用于现场保护的软件的规模，定义了存储器映射 **D**。以后生产的器件都将使用标准 RAM 存储器映射。请参看 **“存储器构成”一章**中关于中档系列 PICmicro® 单片机存储器的使用和实现。

图 C-1： 各种数据存储器映射



注 1：映射在存储区 0 中。
2：未用，读为 '0'。
3：有些器件的某些 GPR 与 SFR 存储空间重合。

C.2 SSP （同步串行口）模块

SSP 模块有 2 种工作方式：

- SPI™ (串行外设接口)
- I²C™ (内部互联)

Microchip 的设计库中有 3 种不同的 SSP 模块。第一种 SSP 模块 (现在叫基本 SSP) 实现 4 种 SPI 模式中的两种和 I²C 从动模式。第二种 SSP 模块 (称作 SSP) 实现全部 4 种 SPI 模式和 I²C 从动模式。第三种 SSP 模块 (称作 MSSP) 实现所有 4 种 SPI 模式和 I²C 的主控和从动模式。表 C-1 列出了具有 SSP 模块的器件及具体的实现版本。随着新器件的推出，将实现 SSP 模块或 MSSP 模块 (即基本 SSP 模块正逐渐被淘汰)。由于不同 SSP 模块对器件体积和成本的影响不同，仅会推出一部分带有主 SSP 模块的器件。如果在应用中需要使用 I²C 主控模式，可考虑使用 Microchip 的高档系列单片机 PIC17CXXX 和 PIC18F/CXXX。

表 C-1： 具有 SSP 模块的器件

器件	同步串行口的版本		
	SSP	基本 SSP	主 SSP (1)
16C62	—	是	—
16C62A	—	是	—
16CR62	—	是	—
16C63	—	是	—
16CR63	—	是	—
16C64	—	是	—
16C64A	—	是	—
16CR64	—	是	—
16C65	—	是	—
16C65A	—	是	—
16CR65	—	是	—
16C66	是	—	—
16C67	是	—	—
16C72	—	是	—
16CR72	是	—	—
16C73	—	是	—
16C73A	—	是	—
16C74	—	是	—
16C74A	—	是	—
16C76	有	—	—
16C77	有	—	—
16C923	有	—	—
16C924	有	—	—
未来将推出的带 SSP 模块的器件	请参阅器件数据手册	—	请参阅器件数据手册

注 1： 目前还没有中档系列的器件带有 MSSP 模块。请登陆 Microchip 的网页或 BBS 获得产品介绍。你会得到新器件功能的详细内容。

Microchip 的高档系列器件 (PIC17CXXX 和 PIC18F/CXXX) 具有 MSSP 模块。请通过 Microchip 的网页、BBS、各地销售办事处或代理商进行咨询。

C.3 A/D （模数转换）模块

Microchip 的设计库中存在几种不同版本的 A/D 模块。第一个 A/D 模块 (现在称作基本型 8 位 A/D) 是具有 4 个输入通道的 8 位 A/D。第二个 A/D 模块 (称作 8 位 A/D) 是具有 8 个输入通道的 8 位 A/D。第三个 A/D 模块 (称作 10 位 A/D) 是具有 16 个输入通道的 10 位 A/D。表 C-2 列出了哪些器件具有 A/D 模块以及模块的版本。新的器件将更多的采用 8 位 A/D 模块或 10 位 A/D 模块（即基本型 8 位 A/D 正逐渐被淘汰）。如果在应用中需要采用 10 位 A/D，可考虑使用 Microchip 的高档系列单片机 (PIC17CXXX 和 PIC18F/CXXX)，该系列中的某些器件具有此模块。

表 C-2： 具有 A/D 模块的器件

器件	8 位 A/D	基本 8 位 A/D	10 位 A/D (1)	积分型 A/D
16C710	—	有	—	
16C71	—	有	—	
16C711	—	有	—	
16C715	—	有	—	
16C72	有	—	—	
16CR72	有	—	—	—
16C73	有	—	—	
16C73A	有	—	—	
16C74	有	—	—	
16C74A	有	—	—	
16C76	有	—	—	
16C77	有	—	—	
16C924	有	—	—	
14C000	—	—	—	有
将来推出的带有 A/D 模块的器件	请参阅器件数据手册	请参阅器件数据手册	请参阅器件数据手册	请参阅器件数据手册

注 1： 目前中档系列的器件都不带有 10 位 A/D 模块。请访问 Microchip 的网站或 BBS 获得产品简介。你会得到关于未来新器件功能的详细介绍。

Microchip 的高档系列器件 (PIC17CXXX 和 PIC18F/CXXX) 具有 10 位 A/D 模块。请通过 Microchip 的网站、 BBS、各地销售办事处或代理商进行咨询。

C.4 欠压复位

内部欠压复位 (BOR) 电路作为一种特殊的功能添加到了器件中。该电路将添加到大多数新器件中。例外情况是如果器件的目标市场需要器件在欠压复位跳变点下仍能正常工作（如手持电池供电应用），这时不需要欠压复位电路。表 C-3 列出了升级后带有 BOR 电路的器件。

表 C-3： 修改后包含欠压复位功能的器件

没有欠压复位功能的基本器件	带欠压复位功能的改进型器件
16C62	16C62A
16C64	16C64A
16C65	16C65A
16C71	16C711
16C73	16C73A
16C74	16C74A

C.5 比较器

在执行一个读操作时（从 Q2 周期的开始），如果改变 CMCON 寄存器 (C1OUT 或 C2OUT)，那么 CMIF 中断标志位可能不会被置 1。

C.6 $\overline{\text{MCLR}}$ 滤波器

主复位 ($\overline{\text{MCLR}}$) 逻辑电路增加了一个滤波器，该滤波器滤除主复位引脚上的低电平窄脉冲（毛刺），从表 C-4 可以看出器件是否带有主复位滤波器。

表 C-4： 带有主复位滤波器的器件

器件	主复位	
	无滤波器 (快速复位)	有滤波器
16C61	是	—
16C62	是	—
16C62A	—	是
16CR62	—	是
16C63	—	是
16CR63	—	是
16C64	是	—
16C64A	—	是
16CR64	—	是
16C65	是	—
16C65A	—	是
16CR65	—	是
16C66	—	是
16C67	—	是
16C620	—	是
16C621	—	是
16C622	—	是
16C710	—	是
16C71	是	—
16C711	—	是
16C715	—	是
16C72	—	是
16CR72	—	是
16C73	是	—
16C73A	—	是
16C74	是	—
16C74A	—	是
16C76	—	是
16C77	—	是
16C83	是	—
16C84	是	—
16F83	是	—
16F84	是	—
16C923	—	是
16C924	—	是
所有新器件	—	是

C.7 USART

最初在中档器件中提供的 USART/SCI 模块具有 “高速”模式 （当 BRGH 控制位置 1 时）。由于采样电路设计的原因，“高速”模式的工作并没有期望的那么可靠。现在已经改进了采样电路，“高速”模式的工作能满足 Microchip 的设计要求。关于采样的差异在 **“USART”一章**中详细介绍，表 C-5 列出了器件使用新、旧采样逻辑的情况。

表 C-5： USART/SCI 采样逻辑

器件	采样逻辑	
	旧	新
16C63	是	—
16CR63	是	—
16C65	是	—
16C65A	是	—
16CR65	是	—
16C66	—	是
16C67	—	是
16C73	是	—
16C73A	是	—
16C74	是	—
16C74A	是	—
16C76	—	是
16C77	—	是
带有 USART/SCI 模块的新器件	—	是

C.8 器件振荡器

在器件振荡器中增加了一种新模式，该模式允许器件工作时使用内部 RC 振荡器。用户可以在编程器件时，通过设置配置字选择内部 RC 振荡模式。在许多未来的器件中都将包含这个新模式。请参阅具体器件数据手册中的配置字，以了解器件是否支持此模式。

C.9 并行从动端口

控制引脚已从电平触发变为边沿触发。

表 C-6： 并行从动端口控制引脚的电平触发方式

器件	触发方式	
	电平触发	边沿触发
16C64	是	—
16C64A	—	是
16C65	是	—
16C65A	—	是
16C67	—	是
16C74	是	—
16C74A	—	是
16C77	—	是
带有并行从动口的新器件	—	是

附录 D： 版本历史

版本 A

这是本参考手册附录部分的初始发行版。

第 35 章 术语表

A

A/D

参见模数转换 （Analog to Digital）。

ALU

算数逻辑单元。 CPU 内负责执行数学运算 （例如加、减等运算）、逻辑运算 （例如与、或等运算）和移位操作的逻辑单元。

B

BCD

参见二进制编码的十进制 （Binary Coded Decimal）。

BOR

参见欠压复位 （Brown-out Reset）。

比较 （Compare）

这是 CCP 模块的一种工作方式。在该方式下，当定时器寄存器的值与比较寄存器中的值匹配时，器件将执行操作。

比较寄存器 （Compare Register）

这是一个 16 位寄存器，它的值与 16 位 TMR1 寄存器进行比较。当计数器的值与比较寄存器的内容匹配时触发比较功能。

并行从动端口 （Parallel Slave Port，PSP）

用来与单片机的 8 位数据总线接口的并行通信端口。

波特率 （Baud）

通常用来描述串行端口的通信速率。即每秒传输的位数 （BPS）。

捕捉 （Capture ）

这是 CCP 模块的一种工作方式。在该方式下，当预先定义的事件发生时，定时器 / 计数器的值即被 “捕捉”到 CCP 寄存器中去。

捕捉寄存器 （Capture Register）

一个 16 位的寄存器。当捕捉事件发生时， 16 位 TMR1 寄存器的值将装入该寄存器。

C

CCP

捕捉、比较、脉宽调制 (PWM)。这个模块可配置为工作在输入捕捉、定时器比较或 PWM 输出方式下。

CPU

中央处理单元。其主要功能是将指令译码、确定需要的操作数和要执行的操作。算数运算、逻辑运算和移位操作将在 ALU 中执行。

采集时间 （Acquisition Time，TACQ）

这是与模数转换器有关的一个术语。采集时间就是 A/D 转换器的采样保持电容充电到与之相连的模拟输入电压值所需要的时间。当 GO 位被置 1 后，模拟输入通道就与采样保持电容断开，启动 A/D 转换。

采样保持电容 （Holding Capacitor）

这是模数转换 （A/D）模块中用到的一个电容，它在模数转换开始后用来 “保持”模拟输入电平。在采样期间，采样保持电容的充电 / 放电是由模拟输入引脚的电平大小决定的。一旦转换开始，采样保持电容就与模拟输入断开连接，开始为 A/D 转换 “保持”电压。

采样时间 （Sampling Time）

采样时间是获得一个 A/D 结果所需的全部时间。它包括采集时间和转换时间。

参考电压 （Voltage Reference， VREF）

一个电压值，可以作为 A/D 转换的参考点 （AVDD 和 AVSS）或比较器的翻转点。

操作码 （Opcode）

14 位指令字的一部分，指定需要进行何种操作。操作码的长度可变，由要执行的指令决定，一般在 4 位以上。其余的指令字包含程序或数据存储器信息。

长字指令 （Long Word Instruction）

包含所有需要信息 （操作码和数据）的单个指令字。这确保了对每条指令的访问和执行都在一个指令周期内完成。

程序存储器 （Program Memory）

程序存储器总线上的任何存储器。静态变量位于程序存储器中 （如表）。

程序计数器 （Program Counter）

指定下一条将要执行的指令在程序存储器中的地址的寄存器。

程序总线 （Program Bus）

将指令字从程序存储器传送到 CPU 的总线。

出栈 （POP）

表示从堆栈 （软件和 / 或硬件）中恢复信息的操作的术语。参见压栈 （PUSH）。

串行外设接口 （Serial Peripheral Interface， SPI）

这是 SSP 模块的一种模式。一般是一个 3 线接口，包括一个数据输出线，一个数据输入线和一个时钟线。因为存在时钟，所以是一个同步接口。

存储区 （Bank）

这是一个寻址数据存储器的方法。因为中档单片机有 7 位用于直接寻址，指令可以寻址 128 字节 （包括特殊功能寄存器）。为寻址更大的数据存储空间，数据存储器被分成多个相邻的存储区，每个存储区 128 字节。为选择存储区，需要准确设置存储区选择位 （RP1 和 RP0）。由于有 2 个存储区选择位，所以可选实现 4 个存储区。

D

D/A

参见数模转换 （Digital to analog）。

DAC

数模转换器。

单周期指令 （Single cycle instruction）

在单个机器周期 （TCY）中完成的指令。

欠压 （Brown-out）

器件的电源电压临时降到规定的最小工作电压以下的一种情况。当负载被接入时会发生这种情况，并引起系统 / 器件的电压下降。

欠压复位 （Brown-out Reset， BOR）

当器件的电压下降到规定的工作电压以下时，使器件复位的电路。某些器件带有内部 BOR 电路，而另一些器件则需要使用外部 BOR 电路。

读－修改－写 （Read-Modify-Write）

读寄存器，修改寄存器，然后将修改后的值写回原来的寄存器。在一个或多个指令周期内完成。

堆栈 （Stack）

CPU 的一部分，保存程序执行的返回地址。当执行一条 CALL 指令或发生中断时，将程序计数器的值压入堆栈。

E

EEPROM

电可擦除的可编程只读存储器。该存储器能够在线编程和擦除。

EPROM

可编程只读存储器。该存储器能够在线编程。可用紫外线擦除。

EXTRC

外部 RC 振荡电路。某些器件允许外部 RC 振荡电路提供器件主时钟，这类似于一些器件的 RC 模式。

二进制编码的十进制 （Binary Coded Decimal， BCD）

每 4 位即半个字节表示一个 0-9 的数字。通常两个这样的数字 （即一个字节）就可以组合表示 0 - 99 范围内的十进制数值。

F

Flash 存储器

这种存储器能够在线编程和擦除。其程序存储器技术在功能上几乎等同于 EEPROM。

Fosc

器件振荡器的频率。

非归零码 （Non-Return to Zero）

用来在通信介质中传输数据的双电平编码机制。值为 1 表示高电压信号，值为 0 表示低电压信号。数据线缺省为高电平。

冯 · 诺依曼架构 （von Neumann Architecture）

在这一架构中，程序存储器和数据存储器位于同一个存储区域。这意味着对程序存储器和数据存储器的访问必须依次进行，这影响了器件的性能。

G

GIO

一般输入 / 输出。

GPIO

通用输入 / 输出。

GPR

通用寄存器 （RAM）。数据存储器的一部分，可以用来存储程序的动态变量。

公用 RAM （Common RAM）

这是数据存储器 RAM 中的一个区域，这个区域对于所有存储区都具有相同的 RAM 地址。公用 RAM 可以实现为地址 70h -7Fh（包括 70h 和 7Fh）。这个区域主要用来在现场切换过程中（如在中断过程中）保存必需的变量。

工作寄存器 （Working Register， W）

可视为器件的累加器，也可以在执行与 ALU 有关的双操作数指令时作为其中的一个操作数。

H

HS

高速。器件振荡器的一种模式，在该模式下，振荡器电路支持高频操作，用于从 4 MHz 至 20 MHz 的操作。

哈佛架构 （Harvard Architecture）

在这种架构中，程序存储器和数据存储器的总线是分离的。这就使对数据存储器和程序存储器的存取能同时进行，提高了器件的性能。

后分频器 （Postscaler）

一种电路，通过对定时器 / 计数器分频来降低该定时器 / 计数器的中断发生（或 WDT 复位）率。

汇编语言 （Assembly Language）

用一种易读的形式描述二进制机器代码的符号语言。

I

I^2C

内部互联。这是一个双线的通信接口，是 SSP 模块的一种工作模式。

INTRC

内部电阻－电容 （RC）振荡器。某些器件可以选择振荡器，因此可以将内部 RC 作为时钟。

J

寄存器文件 （Register File）

数据存储器，包括 SFR 和 GPR。

机器周期 （Machine cycle）

这一概念表示对器件时钟分频形成的单位时间。对于 PICmicro® 单片机，这个单位时间是 4 个器件振荡器周期 （4TOSC），也称为 TCY。

间接寻址 （Indirect Addressing）

指令中不包含数据存储器地址的一种寻址方式。在这种寻址方式下，指令对 INDF 地址进行操作，此时数据寄存器地址是 FSR 寄存器中的值。即使用 INDF 寄存器进行间接寻址，实际上是访问 FSR 寄存器内容所指向的单元。

K

看门狗定时器 （Watchdog Timer， WDT）

用于从产品设计中不希望发生的软件走飞或其它相关系统问题中恢复，从而提高设计的稳定性。如果看门狗定时器没有在溢出清零，将导致看门狗定时器复位。PICmicro 单片机的时钟源是一个片内 RC 振荡器，可以增强系统的可靠性。

L

LCD

液晶显示器。用于显示系统状况。定制 LCD 基板玻璃需要一定的规格。

LED

发光二极管。用于显示系统状况。

LP

一种器件振荡器模式。用于低频操作，允许振荡器工作在低功耗模式。频率可达 200 kHz。

LSb

最低有效位。

LSB

最低有效字节。

立即数 （Literal）

指令字中的常数。

M

MSb

最高有效位。

MSB

最高有效字节。

脉冲宽度调制 （Pulse Width Modulation， PWM）

信息包含在固定频率信号的 （高）脉冲宽度中的信号序列。 CCP 模块提供具有相同占空比的 PWM 输出，不需要软件开销。

模数转换 （Analog to Digital， A/D）

将模拟输入电平按比例转换成数字等价值。

N

NRZ

参见非归零码 （Non-Return to Zero）。

O

OST

参见振荡器起振定时器 （Oscillator Start-up Timer）。

P

PSP

参见并行从动端口 （ Parallel Slave Port）。

PWM

参见脉冲宽度调制 （Pulse Width Modulation）。

配置字 （Configuration Word）

这是用来指定器件工作特性 （如振荡器模式、 WDT 使能、起振定时器使能）的寄存器单元。这些特性可在器件编程时指定。

Q

Q 周期 （Q-cycles）

它与器件振荡周期相同。每个指令周期有 4 个 Q 周期。

取指令 （Instruction Fetch）

由于采用了哈佛架构，当正在执行一条指令时，程序存储器中的下一条指令被 “取出”，一旦当前的指令执行完毕即对其进行译码。

R

RC

电阻－电容。器件振荡器的默认配置。它为器件时钟源提供了一种 “真正廉价的”的实现方法。这种时钟源不能提供准确的时基。支持 4 MHz 时钟频率。（参见 EXTRC）。

ROM

只读存储器。它是内容固定且不能修改的存储器。

S

SFR

特殊功能寄存器。这些寄存器包含器件的控制位和状态信息。

SPI

参见串行外设接口 （Serial Peripheral Interface）。

上电复位 （Power-on Reset， POR）

确定器件电压是否从欠压电平 （0V）上升的电路。如果器件电压从地电平开始上升，则器件复位并且 PWRT 开始工作。

上电延时定时器 （Power-up Timer， PWRT）

将内部复位信号保持低电平一段时间的定时器，以使器件电压从地上升到有效工作电压范围。一旦该定时器超时，则 OST 电路被使能 （对于所有的晶体 / 谐振振荡器模式）。

数据存储器 （Data Memory）

与数据总线相连的存储器。该存储器是易失性 （SRAM）的，由通用寄存器和特殊功能寄存器构成。

数据 EEPROM （Data EEPROM）

电可擦除可编程只读数据存储器。该存储器可被 CPU 编程和再编程，以确保在断电时关键值和变量保存在非易失性存储器中。

数据总线 （Data Bus）

将数据送入和送出数据存储器的总线。

数模转换 （Digital to Analog）

将数字值转换为等比例模拟电压。

T

TAD

在 A/D 转换器中，每一位所需的转换 （将模拟电压转换为数字值）时间。

TCY

指令周期。该周期等于 Fosc/4，并被分成四个 Q 周期。

Tosc

器件振荡器的周期。

特殊功能寄存器 （Special Function Register， SFR）

这些寄存器包含器件的控制位和状态信息。

U

USART

通用同步异步收发器。该模块既可作为全双工异步通信端口工作，也可作为半双工同步通信端口工作。当工作在异步模式下时，可以连接到 PC 的串口上。

W

WDT

看门狗定时器。

W 寄存器 （W Register）

参见工作寄存器 （Working Register）。

X

XT

一种器件振荡器模式，可工作在 100 kHz 至 4 MHz。

休眠模式

这是器件的低功率模式，振荡器不工作。这降低了器件的电流消耗。某些外设可能处于它们仍能继续工作的模式下。

Y

压栈 （PUSH）

表示将信息保存在堆栈 （软件和 / 或硬件）中操作的术语。参见出栈 （POP）。

页 （Pages）

程序存储器的寻址方法。中档器件提供的 CALL 和 GOTO 指令有 11 位寻址的能力，寻址范围可达 2K。为了使器件能够访问更大的程序存储空间，程序存储器被分为相邻的页，每页的大小都是 2K。通过适当地配置页面选择位 （PCLATCH<5:4>）可以选择需要的页。因为当前的器件只提供 2 位页面选择位，因此可以实现 4 个页面。

预分频器 （Prescaler）

降低计数器 / 定时器时钟源频率的电路。

Z

振荡器起振定时器 （Oscillator Start-up Timer， OST）

该定时器在释放内部复位信号前计数 1024 个晶体 / 谐振振荡器时钟。

直接寻址 （Direct Addressing）

当指令中含有数据存储器的地址时，寻址方式就称为直接寻址。这种指令直接访问指定地址中的数据。

指令周期 （Instruction cycle）

一条指令执行的事件。通常有以下四个事件：译码、读、执行和写。并不是所有的指令都要完成这四个事件。要了解指令周期内进行的操作，请参阅每条指令的描述。四个外部时钟周期（Tosc）构成一个指令周期 （TCY）。

指令总线 （Instruction Bus）

用来将程序存储器中的指令字传送到 CPU 的总线。

中断 （Interrupt）

发送到 CPU 的信号，使程序跳转至中断向量地址 （程序存储器中的 04h）。在程序跳转之前，将程序计数器 （PC）的内容保存到硬件堆栈中，以使处理完中断程序后返回到被中断点处。

转换时间 （Conversion Time， Tconv）

这是与模数 （A/D）转换器有关的一个时间量。表示 A/D 转换器将采样保持电容上的模拟电平转换成数字值所需的时间。

总线宽度 （Bus width）

总线所承载的信息的位数。数据存储器的总线宽度是 8 位。中档器件的程序存储器总线宽度是 14 位。

35.1 版本历史

版本 A

这是术语表的初始发行版。

注：

全球销售及服务网点

美洲

公司总部 Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://support.microchip.com
网址：www.microchip.com

亚特兰大 Atlanta
Alpharetta, GA
Tel: 1-770-640-0034
Fax: 1-770-640-0307

波士顿 Boston
Westford, MA
Tel: 1-978-692-3848
Fax: 1-978-692-3821

芝加哥 Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Farmington Hills, MI
Tel: 1-248-538-2250
Fax: 1-248-538-2260

科科莫 Kokomo
Kokomo, IN
Tel: 1-765-864-8360
Fax: 1-765-864-8387

洛杉矶 Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608

圣何塞 San Jose
Mountain View, CA
Tel: 1-650-215-1444
Fax: 1-650-961-0286

加拿大多伦多 Toronto
Mississauga, Ontario,
Canada
Tel: 1-905-673-0699
Fax: 1-905-673-6509

亚太地区

中国 – 北京
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

中国 – 成都
Tel: 86-28-8676-6200
Fax: 86-28-8676-6599

中国 – 福州
Tel: 86-591-8750-3506
Fax: 86-591-8750-3521

中国 – 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 – 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 – 深圳
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

中国 – 顺德
Tel: 86-757-2839-5507
Fax: 86-757-2839-5571

中国 – 青岛
Tel: 86-532-502-7355
Fax: 86-532-502-7205

台湾地区 – 高雄
Tel: 886-7-536-4818
Fax: 886-7-536-4803

台湾地区 – 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

台湾地区 – 新竹
Tel: 886-3-572-9526
Fax: 886-3-572-6459

亚太地区

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-2229-0061
Fax: 91-80-2229-0062

印度 India - New Delhi
Tel: 91-11-5160-8632
Fax: 91-11-5160-8632

日本 Japan - Kanagawa
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

韩国 Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

新加坡 Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

欧洲

奥地利 Austria - Weis
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

丹麦 Denmark - Ballerup
Tel: 45-4420-9895
Fax: 45-4420-9910

法国 France - Massy
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 Germany - Ismaning
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

英国 England - Berkshire
Tel: 44-118-921-5869
Fax: 44-118-921-5820

09/27/04