
12 位高速多 SAR A/D 转换器（ADC）

目录

本章包括下列主题：

1.0	简介	2
2.0	寄存器	5
3.0	转换序列	36
4.0	ADC 操作	36
5.0	应用示例	57
6.0	节能模式期间的操作	70
7.0	复位的影响	70
8.0	寄存器映射	71
9.0	相关应用笔记	75
10.0	版本历史	76

注： 本系列参考手册章节旨在用作对器件数据手册的补充。本手册章节可能并不适用于所有dsPIC33/PIC24器件，具体取决于器件型号。请参见具体器件数据手册中相应章节开头部分的注，以检查本文档是否支持您所使用的器件。

器件数据手册和系列参考手册章节可从Microchip网站<http://www.microchip.com>下载。

1.0 简介

dsPIC33/PIC24 12位高速多SAR模数转换器（Analog-to-Digital Converter，ADC）包含以下特性：

- 多个ADC内核：
 - 多个单通道专用ADC内核（取决于具体器件实现）
 - 一个共用ADC内核
- 每个ADC内核可配置为6、8、10或12位分辨率
- 12位分辨率时，每通道的转换速率最高为3.25 Msps
- 最多32个模拟输入源（取决于具体器件实现）
- 可单独为所有通道中的每个通道选择单端或伪差分输入
- 可单独为所有通道中的每个通道的转换结果选择无符号或有符号数据格式
- 针对每个模拟输入的独立16位转换结果寄存器
- 提前中断产生，可实现转换数据的快速处理
- 集成多个数字比较器（取决于具体器件实现）：
 - 多个比较选项
 - 可分配给特定模拟输入
- 多个过采样滤波器（取决于具体器件实现）：
 - 提高分辨率
 - 可分配给特定模拟输入
- 可在CPU休眠和空闲模式下工作
- 通过硬件电容分压器（Capacitive Voltage Divider，CVD）测量连接到输入的电容值

图1-1、图1-2和图1-3给出了12位多SAR ADC的简化框图。

该模块由几个独立的SAR ADC内核组成。模拟输入（通道）通过多路开关和开关连接到每个ADC内核的采样保持（Sample-and-Hold，S/H）电路。内核使用通道信息（输出格式、测量模式和输入编号）来处理模拟采样。转换完成时，结果会存储在特定模拟输入的结果缓冲区内，并且会在数字滤波器和数字比较器配置为使用该特定通道的数据时传送给二者。

12位高速多SAR A/D转换器（ADC）

图 1-1: 12位高速多SAR ADC框图

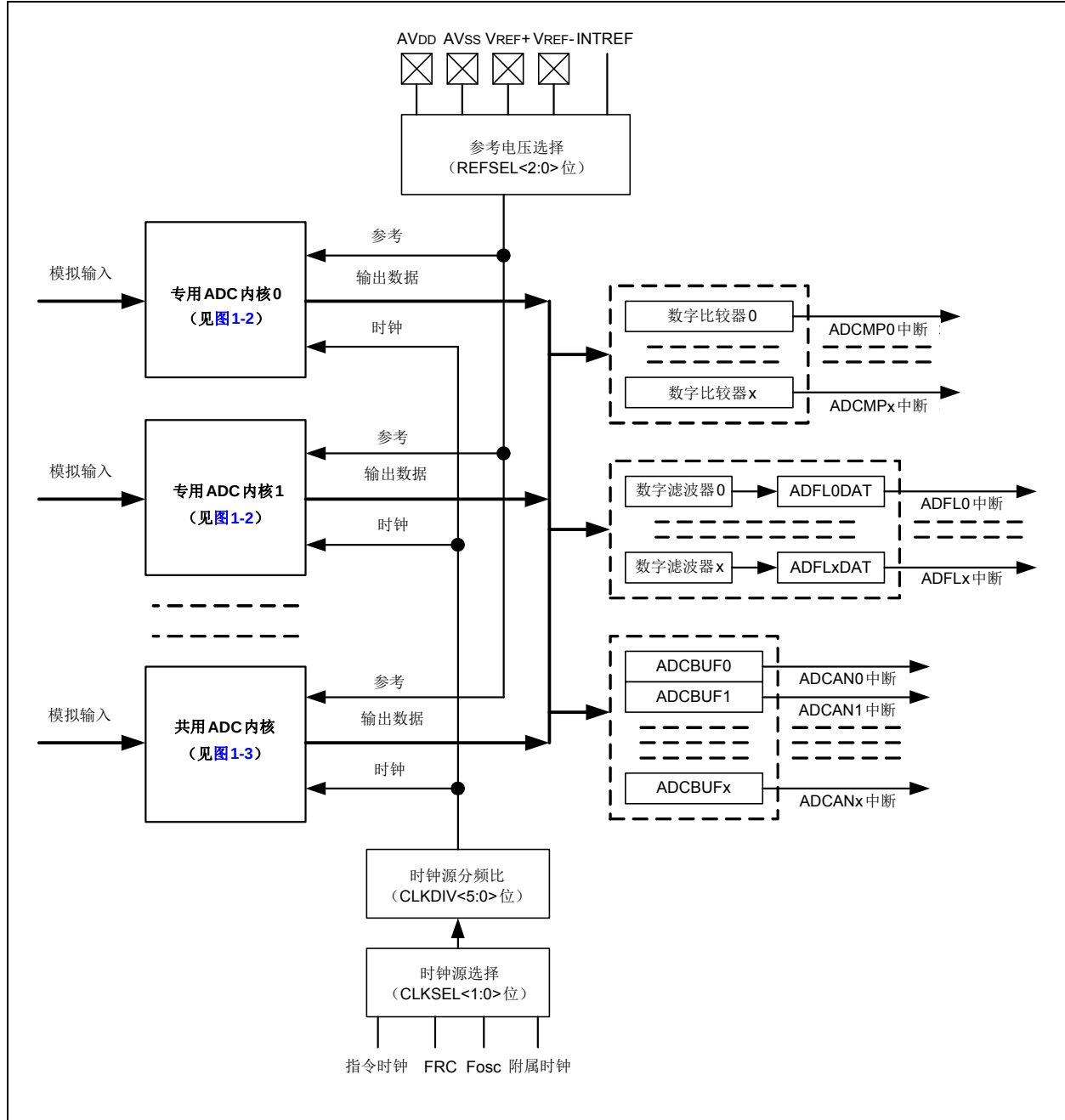


图 1-2: 专用 ADC 内核

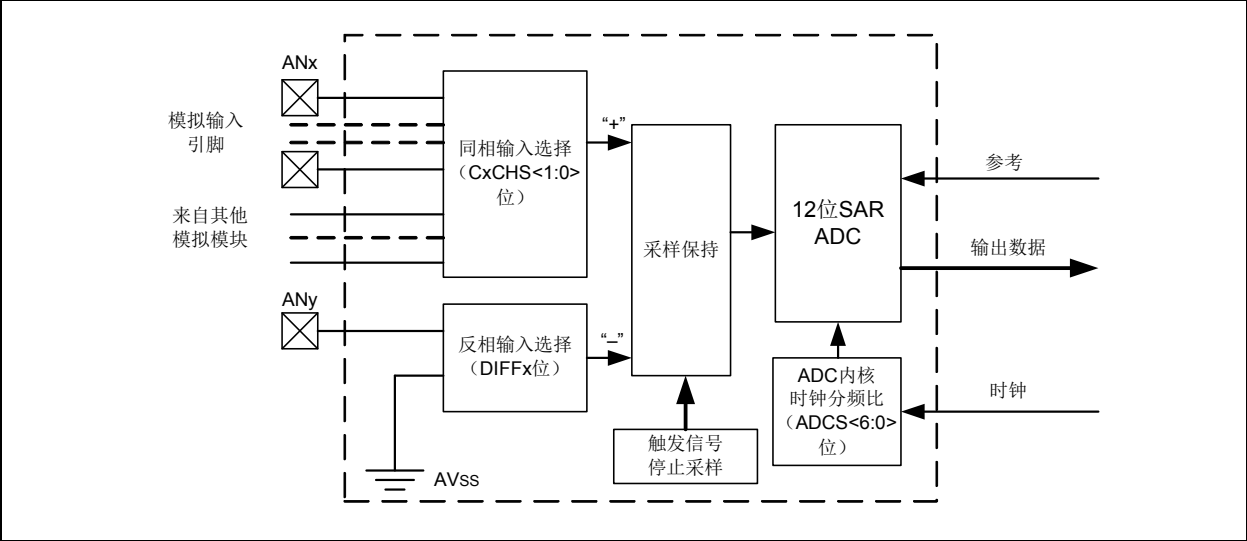
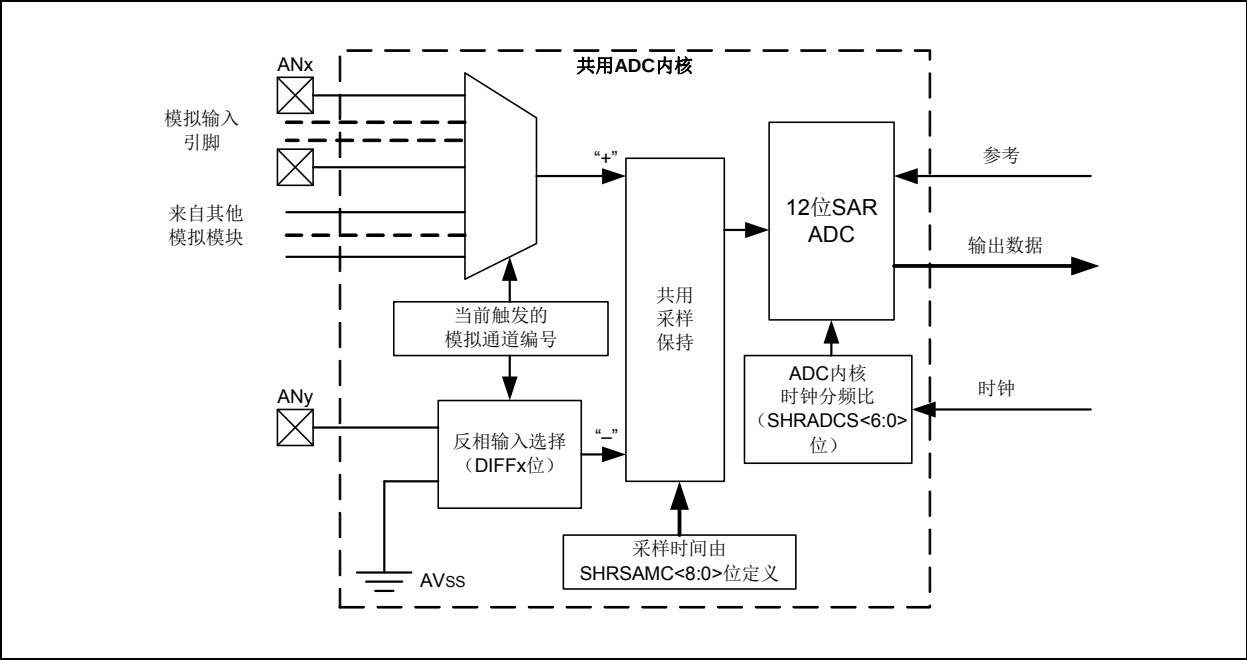


图 1-3: 共用 ADC 内核



2.0 寄存器

12 位高速多 SAR ADC 模块的特殊功能寄存器 (Special Function Register, SFR) 分为两组: 控制寄存器和数据寄存器。表 8-1 提供了 ADC 实现的所有 SFR 的完整列表。

2.1 控制寄存器

ADCON1L 寄存器 (寄存器 2-1) 包含用于使能模块、定义空闲模式下模块行为以及使能 CVD 功能的位。

ADCON1H 寄存器 (寄存器 2-2) 用于控制输出数据格式和共用 ADC 内核分辨率。

ADCON2L 寄存器 (寄存器 2-3) 用于控制共用 ADC 内核的时钟分频比和提前中断时序选择。它具有多个用于为与参考电压相关的事件允许共用中断的位和一个用于为各个输入通道允许提前中断功能的位。

ADCON2H 寄存器 (寄存器 2-4) 用于控制共用 ADC 内核的采样时间。它还提供状态位来指示模块参考电压是否已准备好开始工作。该寄存器允许为 CVD 功能调整内部电容值。

ADCON3L 寄存器 (寄存器 2-5) 用于为所有 ADC 内核选择参考电压并控制共用触发、电平触发和软件单触发。它还具有用于暂停对模块的所有触发的控制位。

ADCON3H 寄存器 (寄存器 2-6) 具有用于使能所有 ADC 内核并为模块选择时钟源的位。该寄存器还可控制模块时钟源分频比。

ADCON4L 寄存器 (寄存器 2-7) 能够在专用 ADC 内核的触发和转换操作之间插入延时, 并且能够触发同步。

ADCON4H 寄存器 (寄存器 2-8) 用于为专用 ADC 内核选择通道。

ADCON5L 寄存器 (寄存器 2-9) 用于控制所有 ADC 内核的电源。

ADCON5H 寄存器 (寄存器 2-10) 具有用于在每个 ADC 内核已上电并准备好开始工作时为其允许共用中断的位。在该寄存器中, 还可为所有 ADC 内核指定上电延时。

ADCOREnL (其中 “n” 是专用 ADC 内核编号) 寄存器 (寄存器 2-11) 用于为每个专用 ADC 内核定义触发和转换操作之间的延时。

ADCOREnH (其中 “n” 是专用 ADC 内核编号) 寄存器 (寄存器 2-12) 用于为每个专用 ADC 内核定义分辨率、提前中断时间选择和 ADC 内核时钟分频比。

ADLVLTRGL 和 ADLVLTRGH 寄存器 (寄存器 2-13 和寄存器 2-14) 具有用于为每个输入通道选择电平敏感触发或边沿敏感触发的位。

ADEIEL 和 ADEIEH 寄存器 (寄存器 2-15 和寄存器 2-16) 具有用于为每个输入通道允许提前中断产生的位。

ADEISTATL 和 ADEISTATH 寄存器 (寄存器 2-17 和寄存器 2-18) 包含每个输入通道的提前中断状态标志。

ADMOD0L、ADMOD0H、ADMOD1L 和 ADMOD1H 寄存器 (寄存器 2-19 至寄存器 2-22) 具有用于为每个输入通道使能伪差分模式和有符号输出数据格式的位。

ADIEL 和 ADIEH 寄存器 (寄存器 2-23 和寄存器 2-24) 具有用于为每个输入通道允许独立中断和共用中断的位。

ADSTATL 和 ADSTATH 寄存器 (寄存器 2-25 和寄存器 2-26) 包含每个输入通道的数据就绪标志。

ADTRIGNL 和 ADTRIGH 寄存器 (寄存器 2-27) 用于为每个输入通道定义触发源。

ADCAL0L、ADCAL0H、ADCAL1L 和 ADCAL1H 寄存器 (寄存器 2-28 至寄存器 2-31) 用于控制对每个 ADC 内核的校准。某些器件不需要校准功能。请参见具体器件的数据手册, 了解器件是否实现了这些寄存器。

ADCMpNCON 寄存器（[寄存器 2-32](#)）用于控制数字比较器的操作，包括中断产生和要使用的比较条件。这些寄存器还可在比较器事件发生时提供状态指示。每个数字比较器都分配有一个寄存器。

ADCMpNENL 和 ADCMpNENH 寄存器（[寄存器 2-33](#) 和 [寄存器 2-34](#)）用于选择哪些模拟输入转换结果要通过数字比较器处理。每个数字比较器都分配有一对（L 和 H）寄存器。

ADFLnCON 寄存器（[寄存器 2-35](#)）用于控制过采样滤波器的操作并提供滤波器操作状态位。每个过采样滤波器都分配有一个寄存器。ADCSSL 和 ADCSSH 寄存器（[寄存器 2-36](#) 和 [寄存器 2-37](#)）用于选择 CVD 将扫描/处理哪些模拟输入。

2.2 数据寄存器

ADCBUFx 寄存器用于存储模数转换的输出数据。通常，每个实现的模拟通道都分配有一个寄存器；每个通道将具有相应编号的 ADCBUFx 寄存器。尽管这些寄存器为 16 位宽，但如何使用这些寄存器存储 12 位转换结果取决于所选的数据输出格式。更多信息，请参见 [第 4.10 节“转换结果”](#)。

ADCMpNLO 和 ADCMpNHI 寄存器用于存储 16 位上限和下限数字比较值，以供数字比较器使用。每个 ADC 比较器都分配有一对（HI 和 LO）寄存器。

ADFLnDAT 寄存器包含来自过采样滤波器的 16 位输出数据。每个过采样滤波器都分配有一个寄存器。

ADCVDDAT 寄存器包含电容分压器（CVD）的 16 位输出数据。某些器件可能未实现该寄存器。请参见具体器件的数据手册，了解器件是否实现了 CVD 功能。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-1: **ADCON1L: ADC 控制寄存器 1 低位字**

R/W-0	U-0	R/W-0	U-0	R/W-0	U-0	U-0	U-0
ADON ⁽¹⁾	—	ADSIDL	—	CVDEN ⁽²⁾	—	—	—
bit 15				bit 8			

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
NRE ⁽³⁾	—	—	—	—	—	—	—
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15 **ADON: ADC 使能位⁽¹⁾**

1 = 使能 ADC 模块

0 = 关闭 ADC 模块

bit 14 **未实现: 读为 0**

bit 13 **ADSIDL: ADC 空闲模式停止位**

1 = 当器件进入空闲模式时, 模块停止工作

0 = 在空闲模式下模块继续工作

bit 12 **未实现: 读为 0**

bit 11 **CVDEN: CVD 使能位⁽²⁾**

1 = 使能 CVD

0 = CVD 关闭

bit 10-8 **未实现: 读为 0**

bit 7 **NRE: 降低噪声使能位⁽³⁾**

1 = 当另一个内核完成转换时暂停转换过程 1 TADCORE, 以降低内核之间的噪声干扰

0 = 禁止降低噪声功能

bit 6-0 **未实现: 读为 0**

注 1: 只有在 ADC 模块配置完成后, 才能将 ADON 位置 1。如果在 ADON = 1 时更改 ADC 配置位, 将导致不可预测的行为。

2: 并非所有器件都具有 CVD 功能, CVDEN 位可能未实现。请参见具体器件的数据手册了解更多信息。

3: 并非所有器件都具有降低噪声功能, NRE 位可能未实现。请参见具体器件的数据手册了解更多信息。

寄存器 2-2: ADCON1H: ADC 控制寄存器 1 高位字

r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0
—	—	—	—	—	—	—	—
bit 15							bit 8
R/W-0	R/W-1	R/W-1	r-0	r-0	r-0	r-0	r-0
FORM	SHRRES1	SHRRES0	—	—	—	—	—
bit 7							bit 0

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

- bit 15-8 保留: 必须写为 0
- bit 7 **FORM:** 小数数据输出格式位
 1 = 小数
 0 = 整数
- bit 6-5 **SHRRES<1:0>:** 共用 ADC 内核分辨率选择位
 11 = 12 位分辨率
 10 = 10 位分辨率
 01 = 8 位分辨率
 00 = 6 位分辨率
- bit 4-0 保留: 必须写为 0

12位高速多SAR A/D转换器（ADC）

寄存器2-3: **ADCON2L: ADC控制寄存器2低位字**

R/W-0	R/W-0	r-0	R/W-0	r-0	R/W-0	R/W-0	R/W-0
REFCIE	REFERCIE ⁽²⁾	—	EIEN	—	SHREISEL2 ⁽¹⁾	SHREISEL1 ⁽¹⁾	SHREISEL0 ⁽¹⁾
bit 15						bit 8	

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	SHRADCS6	SHRADCS5	SHRADCS4	SHRADCS3	SHRADCS2	SHRADCS1	SHRADCS0
bit 7						bit 0	

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

bit 15 **REFCIE:** 带隙和参考电压就绪共用中断允许位
1 = 当带隙和参考电压就绪时，将产生共用中断
0 = 禁止在发生带隙和参考电压就绪事件时产生共用中断

bit 14 **REFERCIE:** 带隙和参考电压错误共用中断允许位⁽²⁾
1 = 当检测到带隙和参考电压错误时，将产生共用中断
0 = 禁止在发生带隙和参考电压错误事件时产生共用中断

bit 13 **保留:** 必须写为0

bit 12 **EIEN:** 提前中断允许位
1 = 为输入通道中断允许提前中断功能（EISTATx标志置1时）
0 = 转换完成时产生各个中断（ANxRDY标志置1时）

bit 11 **保留:** 必须写为0

bit 10-8 **SHREISEL<2:0>:** 共用内核提前中断时间选择位⁽¹⁾
111 = 在数据就绪的8个TADCORE时钟前产生提前中断
110 = 在数据就绪的7个TADCORE时钟前产生提前中断
101 = 在数据就绪的6个TADCORE时钟前产生提前中断
100 = 在数据就绪的5个TADCORE时钟前产生提前中断
011 = 在数据就绪的4个TADCORE时钟前产生提前中断
010 = 在数据就绪的3个TADCORE时钟前产生提前中断
001 = 在数据就绪的2个TADCORE时钟前产生提前中断
000 = 在数据就绪的1个TADCORE时钟前产生提前中断

bit 7 **未实现:** 读为0

bit 6-0 **SHRADCS<6:0>:** 共用ADC内核输入时钟分频比位
这些位用于确定一个共用TADCORE（内核时钟周期）内的TCORESRC（源时钟周期）数
1111111 = 254个源时钟周期
•
•
•
0000011 = 6个源时钟周期
0000010 = 4个源时钟周期
0000001 = 2个源时钟周期
0000000 = 2个源时钟周期

- 注 1:** 对于6位共用ADC内核分辨率（SHRRES<1:0> = 00），SHREISEL<2:0>设置100至111无效，不应使用。对于8位共用ADC内核分辨率（SHRRES<1:0> = 01），SHREISEL<2:0>设置110和111无效，不应使用。
- 2:** 为了避免错误的中断，REFERCIE位必须仅在模块使能（ADON = 1）后置1。

dsPIC33/PIC24 系列参考手册

寄存器 2-4: **ADCON2H: ADC 控制寄存器 2 高位字**

R/HS/HC-0	R/HS/HC-0	r-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
REFRDY	REFERR	—	CVDCAP2 ⁽¹⁾	CVDCAP1 ⁽¹⁾	CVDCAP0 ⁽¹⁾	SHRSAMC9	SHRSAMC8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SHRSAMC7	SHRSAMC6	SHRSAMC5	SHRSAMC4	SHRSAMC3	SHRSAMC2	SHRSAMC1	SHRSAMC0
bit 7							bit 0

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

bit 15 **REFRDY:** 带隙和参考电压就绪标志位

1 = 带隙和参考电压就绪
0 = 带隙和参考电压未就绪

bit 14 **REFERR:** 带隙或参考电压错误标志位

1 = 带隙或参考电压在 ADC 模块使能 (ADON = 1) 后中断
0 = 未检测到带隙或参考电压错误

bit 13 **保留:** 必须写为 0

bit 12-10 **CVDCAP<2:0>:** CVD 附加电容值选择位⁽¹⁾

当使能 CVD 时, 该电容值加到共用内核采样保持电容 (CHOLD)。

111 = 7 * 2.5 pF = 17.5 pF
110 = 6 * 2.5 pF = 15 pF
101 = 5 * 2.5 pF = 12.5 pF
100 = 4 * 2.5 pF = 10 pF
011 = 3 * 2.5 pF = 7.5 pF
010 = 2 * 2.5 pF = 5 pF
001 = 1 * 2.5 pF = 2.5 pF
000 = 0 * 2.5 pF = 0 pF

bit 9-0 **SHRSAMC<9:0>:** 共用 ADC 内核采样时间选择位

这些位用于指定共用 ADC 内核采样时间内的共用内核时钟周期 (TADCORE) 数。

111111111 = 1025 个 TADCORE

•
•
•

000000001 = 3 个 TADCORE

000000000 = 2 个 TADCORE

注 1: 并非所有器件都具有 CVD 功能, CVDCAP<2:0> 位可能未实现。请参见具体器件的数据手册了解更多信息。

12位高速多SAR A/D转换器（ADC）

寄存器2-5: **ADCON3L: ADC控制寄存器3低位字**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/HS/HC-0	R/W-0	R/W/HC-0
REFSEL2 ⁽¹⁾	REFSEL1 ⁽¹⁾	REFSEL0 ⁽¹⁾	SUSPEND	SUSPCIE	SUSPRDY	SHRSAMP	CNVRTCH
bit 15							bit 8

R/W-0	R/W/HC-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SWLCTRG	SWCTRG	CNVCHSEL5	CNVCHSEL4	CNVCHSEL3	CNVCHSEL2	CNVCHSEL1	CNVCHSEL0
bit 7							bit 0

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

- bit 15-13 **REFSEL<2:0>**: ADC参考电压选择位⁽¹⁾
- bit 12 **SUSPEND**: 所有ADC内核触发禁止位
1 = 禁止全部ADC内核的所有新触发事件
0 = 可触发全部ADC内核
- bit 11 **SUSPCIE**: 暂停所有ADC内核共用中断允许位
1 = 当ADC内核触发暂停（SUSPEND位 = 1）且之前的所有转换均完成（SUSPRDY位置1）时，将产生共用中断
0 = 发生暂停ADC内核事件时，不产生共用中断
- bit 10 **SUSPRDY**: 所有ADC内核暂停标志位
1 = 所有ADC内核均暂停（SUSPEND位 = 1）且未在进行转换
0 = ADC内核正在进行之前的转换
- bit 9 **SHRSAMP**: 共用ADC内核采样直接控制位
该位应与由CNVRTCH位控制的独立通道转换触发配合使用。它将由CNVCHSEL<5:0>位指定的模拟输入连接到共用ADC内核，并且可延长采样时间。该位不是由硬件控制，必须在转换开始（CNVRTCH置1）前清零。
1 = 共用ADC内核采样由CNVCHSEL<5:0>位指定的模拟输入
0 = 采样由共用ADC内核硬件控制
- bit 8 **CNVRTCH**: 软件独立通道转换触发位
1 = 为由CNVCHSEL<5:0>位指定的模拟输入生成单个触发信号；该位置1后，将在下一个指令周期由硬件自动清零
0 = 可生成下一个独立通道转换触发信号
- bit 7 **SWLCTRG**: 软件电平敏感共用触发位
1 = 当在ADTRIGNL和ADTRIGNH寄存器中将软件电平敏感共用触发信号选作触发源时，将为所有通道连续生成触发信号
0 = 不生成软件电平敏感共用触发信号
- bit 6 **SWCTRG**: 软件共用触发位
1 = 当在ADTRIGNL和ADTRIGNH寄存器中将软件共用触发信号选作触发源时，为所有通道生成单个触发信号；该位置1后，将在下一个指令周期由硬件自动清零
0 = 准备好生成下一个软件共用触发信号
- bit 5-0 **CNVCHSEL<5:0>**: 软件独立通道转换触发的通道编号选择位
这些位用于定义CNVRTCH位置1时要转换的通道。

注 1: 关于可用的参考电压源选项，请参见具体器件的数据手册。

dsPIC33/PIC24 系列参考手册

寄存器 2-6: **ADCON3H: ADC 控制寄存器 3 高位字**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CLKSEL1 ⁽¹⁾	CLKSEL0 ⁽¹⁾	CLKDIV5	CLKDIV4	CLKDIV3	CLKDIV2	CLKDIV1	CLKDIV0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SHREN	C6EN ⁽²⁾	C5EN ⁽²⁾	C4EN ⁽²⁾	C3EN ⁽²⁾	C2EN ⁽²⁾	C1EN ⁽²⁾	C0EN ⁽²⁾
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-14 **CLKSEL<1:0>: ADC 模块时钟源选择位⁽¹⁾**

bit 13-8 **CLKDIV<5:0>: ADC 模块时钟源分频比位**

CLKSEL<1:0> 位选择的 Tsrc ADC 模块时钟源通过该分频比形成所有 ADC 内核（共用和专用）使用的 Tcoresrc 时钟。之后，每个 ADC 内核会使用 ADCOREnH 寄存器中的 ADCS<6:0> 位或 ADCON2L 寄存器中的 SHRADCS<6:0> 位单独对 Tcoresrc 时钟进行分频以得到特定于内核的 TadcCore 时钟。

111111 = 64 个源时钟周期

•

•

•

000011 = 4 个源时钟周期

000010 = 3 个源时钟周期

000001 = 2 个源时钟周期

000000 = 1 个源时钟周期

bit 7 **SHREN: 共用 ADC 内核使能位**

该位不会禁止内核时钟和模拟偏置电路。

1 = 使能共用 ADC 内核

0 = 禁止共用 ADC 内核

bit 6-0 **C6EN:C0EN: 专用 ADC 内核 x 使能位⁽²⁾**

该位不会禁止内核时钟和模拟偏置电路。

1 = 使能专用 ADC 内核 x

0 = 禁止专用 ADC 内核 x

注 1: 关于可用的 ADC 模块时钟源选项，请参见具体器件的数据手册。

2: 可用的专用 ADC 内核的数量特定于器件，某些 CxEN 位可能未实现。更多信息，请参见器件的数据手册。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-7: **ADCON4L: ADC 控制寄存器 4 低位字**

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

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	SAMC6EN ⁽¹⁾	SAMC5EN ⁽¹⁾	SAMC4EN ⁽¹⁾	SAMC3EN ⁽¹⁾	SAMC2EN ⁽¹⁾	SAMC1EN ⁽¹⁾	SAMC0EN ⁽¹⁾
bit 7							bit 0

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

- bit 15 **未实现:** 读为 0
- bit 14-8 **保留:** 必须写为 0
- bit 7 **未实现:** 读为 0
- bit 6-0 **SAMC6EN:SAMC0EN:** 专用 ADC 内核 x 转换延时使能位 ⁽¹⁾
- 1 = 触发后, 转换将被延迟, ADC 内核将在由 ADCOREnL 寄存器中的 SAMC<9:0> 位指定的时间内继续采样
- 0 = 触发后, 采样将立即停止, 转换将在下一个内核时钟周期开始

注 1: 可用的专用 ADC 内核的数量特定于器件, 某些 SAMCxEN 位可能未实现。更多信息, 请参见器件的数据手册。

dsPIC33/PIC24 系列参考手册

寄存器 2-8: **ADCON4H: ADC 控制寄存器 4 高位字**

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	C6CHS1 ⁽¹⁾	C6CHS0 ⁽¹⁾	C5CHS1 ⁽¹⁾	C5CHS0 ⁽¹⁾	C4CHS1 ⁽¹⁾	C4CHS0 ⁽¹⁾
bit 15							bit 8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C3CHS1 ⁽¹⁾	C3CHS0 ⁽¹⁾	C2CHS1 ⁽¹⁾	C2CHS0 ⁽¹⁾	C1CHS1 ⁽¹⁾	C1CHS0 ⁽¹⁾	C0CHS1 ⁽¹⁾	C0CHS0 ⁽¹⁾
bit 7							bit 0

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

- bit 15-14 **未实现:** 读为 0
- bit 13-12 **C6CHS<1:0>:** 专用 ADC 内核 6 输入通道选择位 ⁽¹⁾
- bit 11-10 **C5CHS<1:0>:** 专用 ADC 内核 5 输入通道选择位 ⁽¹⁾
- bit 9-8 **C4CHS<1:0>:** 专用 ADC 内核 4 输入通道选择位 ⁽¹⁾
- bit 7-6 **C3CHS<1:0>:** 专用 ADC 内核 3 输入通道选择位 ⁽¹⁾
- bit 5-4 **C2CHS<1:0>:** 专用 ADC 内核 2 输入通道选择位 ⁽¹⁾
- bit 3-2 **C1CHS<1:0>:** 专用 ADC 内核 1 输入通道选择位 ⁽¹⁾
- bit 1-0 **C0CHS<1:0>:** 专用 ADC 内核 0 输入通道选择位 ⁽¹⁾

注 1: 可用的专用 ADC 内核的数量以及每个 ADC 内核的输入通道选项特定于器件。某些 CxCHS<1:0> 位可能未实现。有关可用 ADC 内核及其输入通道选项的信息, 请参见器件的数据手册。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-9: **ADCON5L: ADC 控制寄存器 5 低位字**

R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0
SHRRDY	C6RDY ⁽¹⁾	C5RDY ⁽¹⁾	C4RDY ⁽¹⁾	C3RDY ⁽¹⁾	C2RDY ⁽¹⁾	C1RDY ⁽¹⁾	C0RDY ⁽¹⁾
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SHRPWR	C6PWR ⁽¹⁾	C5PWR ⁽¹⁾	C4PWR ⁽¹⁾	C3PWR ⁽¹⁾	C2PWR ⁽¹⁾	C1PWR ⁽¹⁾	C0PWR ⁽¹⁾
bit 7							bit 0

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

- bit 15 **SHRRDY:** 共用 ADC 内核就绪标志位
1 = ADC 内核已上电并准备好开始工作
0 = ADC 内核未准备好开始工作
- bit 14-8 **C6RDY:C0RDY:** 专用 ADC 内核 x 就绪标志位⁽¹⁾
1 = ADC 内核已上电并准备好开始工作
0 = ADC 内核未准备好开始工作
- bit 7 **SHRPWR:** 共用 ADC 内核电源使能位
1 = ADC 内核上电
0 = ADC 内核关闭
- bit 6-0 **C6PWR:C0PWR:** 专用 ADC 内核 x 电源使能位⁽¹⁾
1 = ADC 内核上电
0 = ADC 内核关闭

注 1: 可用的专用 ADC 内核的数量特定于器件。某些 CxRDY 和 CxPWR 位可能未实现。关于可用 ADC 内核的信息, 请参见器件的数据手册。

dsPIC33/PIC24 系列参考手册

寄存器 2-10: **ADCON5H: ADC 控制寄存器 5 高位字**

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	WARMTIME3	WARMTIME2	WARMTIME1	WARMTIME0
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SHRCIE	C6CIE ⁽¹⁾	C5CIE ⁽¹⁾	C4CIE ⁽¹⁾	C3CIE ⁽¹⁾	C2CIE ⁽¹⁾	C1CIE ⁽¹⁾	C0CIE ⁽¹⁾
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-12 **未实现:** 读为 0

bit 11-8 **WARMTIME<3:0>:** ADC 内核上电延时位

这些位用于为所有 ADC 内核确定以内核时钟源周期 (TCORESRC) 数表示的上电延时。

1111 = 32768 个源时钟周期

1110 = 16384 个源时钟周期

1101 = 8192 个源时钟周期

1100 = 4096 个源时钟周期

1011 = 2048 个源时钟周期

1010 = 1024 个源时钟周期

1001 = 512 个源时钟周期

1000 = 256 个源时钟周期

0111 = 128 个源时钟周期

0110 = 64 个源时钟周期

0101 = 32 个源时钟周期

0000-0100 = 16 个源时钟周期

bit 7 **SHRCIE:** 共用 ADC 内核就绪共用中断允许位

1 = 当 ADC 内核已上电且准备好开始工作时, 产生共用中断

0 = 禁止在发生 ADC 内核就绪事件时产生共用中断

bit 6-0 **C6CIE:C0CIE:** 专用 ADC 内核 x 就绪共用中断允许位 ⁽¹⁾

1 = 当 ADC 内核已上电且准备好开始工作时, 产生共用中断

0 = 禁止在发生 ADC 内核就绪事件时产生共用中断

注 1: 可用的专用 ADC 内核的数量特定于器件。某些 CxCIE 位可能未实现。关于可用 ADC 内核的信息, 请参见器件的数据手册。

12 位高速多 SAR A/D 转换器（ADC）

寄存器 2-11: **ADCOREnL**: 专用 ADC 内核 n 控制寄存器低位字⁽¹⁾

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	SAMC<9:8>	
bit 15						bit 8	

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SAMC<7:0>							
bit 7						bit 0	

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-10 **未实现**: 读为 0

bit 9-0 **SAMC<9:0>**: 专用 ADC 内核 n 转换延时选择位

这些位用于确定触发事件与转换开始的时间间隔, 这段间隔用内核时钟周期 (TADCORE) 数表示。在此期间, ADC 内核继续采样。该功能通过 ADCON4L 寄存器中的 SAMCxEN 位使能。

1111111111 = 1025 个 TADCORE

•

•

•

0000000001 = 3 个 TADCORE

0000000000 = 2 个 TADCORE

注 1: 可用的专用 ADC 内核的数量特定于器件。关于可用 ADC 内核的信息, 请参见器件的数据手册。

dsPIC33/PIC24 系列参考手册

寄存器 2-12: **ADCOREnH: 专用 ADC 内核 n 控制寄存器高位字⁽¹⁾**

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
—	—	—	EISEL2 ⁽²⁾	EISEL1 ⁽²⁾	EISEL0 ⁽²⁾	RES1	RES0
bit 15						bit 8	

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-13 **未实现: 读为 0**

bit 12-10 **EISEL<2:0>: ADC 内核 n 提前中断时间选择位⁽²⁾**

111 = 在数据就绪的 8 个 TADCORE 时钟前产生提前中断
110 = 在数据就绪的 7 个 TADCORE 时钟前产生提前中断
101 = 在数据就绪的 6 个 TADCORE 时钟前产生提前中断
100 = 在数据就绪的 5 个 TADCORE 时钟前产生提前中断
011 = 在数据就绪的 4 个 TADCORE 时钟前产生提前中断
010 = 在数据就绪的 3 个 TADCORE 时钟前产生提前中断
001 = 在数据就绪的 2 个 TADCORE 时钟前产生提前中断
000 = 在数据就绪的 1 个 TADCORE 时钟前产生提前中断

bit 9-8 **RES<1:0>: ADC 内核 n 分辨率选择位**

11 = 12 位分辨率
10 = 10 位分辨率
01 = 8 位分辨率
00 = 6 位分辨率

bit 7 **未实现: 读为 0**

bit 6-0 **ADCS<6:0>: ADC 内核 x 输入时钟频比位**

这些位用于确定一个内核时钟周期 (TADCORE) 内的内核时钟源周期 (TCORESRC) 数。
1111111 = 254 个源时钟周期
.
.
.
0000011 = 6 个源时钟周期
0000010 = 4 个源时钟周期
0000001 = 2 个源时钟周期
0000000 = 2 个源时钟周期

注 1: 可用的专用 ADC 内核的数量特定于器件。关于可用 ADC 内核的信息, 请参见器件的数据手册。

2: 对于 6 位 ADC 内核分辨率 (RES<1:0> = 00), EISEL<2:0> 设置 100 至 111 无效, 不应使用。对于 8 位 ADC 内核分辨率 (RES<1:0> = 01), EISEL<2:0> 设置 110 和 111 无效, 不应使用。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-13: ADLVLTRGL: ADC 电平敏感触发控制寄存器低位字

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
LVLEN<15:8> ⁽¹⁾							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LVLEN<7:0> ⁽¹⁾							
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-0 **LVLEN<15:0>**: 电平触发使能位⁽¹⁾

1 = 输入通道触发为电平敏感

0 = 输入通道触发为边沿敏感

注 1: 可用的 ADC 通道的数量特定于器件。某些 LVLENx 位可能未实现。有关可用 ADC 内核及其输入通道选项的信息, 请参见器件的数据手册。

寄存器 2-14: ADLVLTRGH: ADC 电平敏感触发控制寄存器高位字

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1
LVLEN<31:24> ⁽¹⁾							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LVLEN<23:16> ⁽¹⁾							
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-0 **LVLEN<31:16>**: 电平触发使能位⁽¹⁾

1 = 输入通道触发为电平敏感

0 = 输入通道触发为边沿敏感

注 1: 可用的 ADC 通道的数量特定于器件。某些 LVLENx 位可能未实现。有关可用 ADC 内核及其输入通道选项的信息, 请参见器件的数据手册。

寄存器 2-15: **ADEIEL: ADC 提前中断允许寄存器低位字**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EIE<15:8> ⁽¹⁾							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EIE<7:0> ⁽¹⁾							
bit 7				bit 0			

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

bit 15-0 **EIE<15:0>**: 相应模拟输入的提前中断允许位⁽¹⁾
1 = 为通道允许提前中断
0 = 为通道禁止提前中断

注 1: 可用通道特定于器件。某些EIE_x位可能未实现。关于可用通道的信息, 请参见器件的数据手册。

寄存器 2-16: **ADEIEH: ADC 提前中断允许寄存器高位字**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EIE<31:24> ⁽¹⁾							
bit 15				bit 8			

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EIE<23:16> ⁽¹⁾							
bit 7				bit 0			

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

bit 15-0 **EIE<31:16>**: 相应模拟输入的提前中断允许位⁽¹⁾
1 = 为通道允许提前中断
0 = 为通道禁止提前中断

注 1: 可用通道特定于器件。某些EIE_x位可能未实现。关于可用通道的信息, 请参见器件的数据手册。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-17: ADEISTATL: ADC 提前中断状态寄存器低位字

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EISTAT<15:8> ⁽¹⁾							
bit 15							
bit 8							

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EISTAT<7:0> ⁽¹⁾							
bit 7							
bit 0							

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-0 EISTAT<15:0>: 相应模拟输入的提前中断状态位⁽¹⁾

1 = 已产生提前中断

0 = 自上一次 ADCBUFx 读取后未产生提前中断

注 1: 可用通道特定于器件。某些 EISTATx 位可能未实现。关于可用通道的信息, 请参见器件的数据手册。

寄存器 2-18: ADEISTATH: ADC 提前中断状态寄存器高位字

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EISTAT<31:24> ⁽¹⁾							
bit 15							
bit 8							

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
EISTAT<23:16> ⁽¹⁾							
bit 7							
bit 0							

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit 15-0 EISTAT<31:16>: 相应模拟输入的提前中断状态位⁽¹⁾

1 = 已产生提前中断

0 = 自上一次 ADCBUFx 读取后未产生提前中断

注 1: 可用通道特定于器件。某些 EISTATx 位可能未实现。关于可用通道的信息, 请参见器件的数据手册。

dsPIC33/PIC24 系列参考手册

寄存器 2-19: **ADMOD0L: ADC 输入模式控制寄存器 0 低位字**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DIFF7 ⁽¹⁾	SIGN7 ⁽¹⁾	DIFF6 ⁽¹⁾	SIGN6 ⁽¹⁾	DIFF5 ⁽¹⁾	SIGN5 ⁽¹⁾	DIFF4 ⁽¹⁾	SIGN4 ⁽¹⁾
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DIFF3 ⁽¹⁾	SIGN3 ⁽¹⁾	DIFF2 ⁽¹⁾	SIGN2 ⁽¹⁾	DIFF1 ⁽¹⁾	SIGN1 ⁽¹⁾	DIFF0 ⁽¹⁾	SIGN0 ⁽¹⁾
bit 7							bit 0

图注:

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

- bit (奇数) **DIFF<7:0>:** 相应模拟输入的差分模式位⁽¹⁾
 1 = 通道为差分模式
 0 = 通道为单端模式
- bit (偶数) **SIGN<7:0>:** 相应模拟输入的输出数据符号位⁽¹⁾
 1 = 通道输出数据为有符号数
 0 = 通道输出数据为无符号数

注 1: 可用的输入通道特定于器件。某些通道可能未实现。另外, 并非所有通道都支持差分模式。关于可用 SIGNx 和 DIFFx 位的信息, 请参见器件的数据手册。

寄存器 2-20: **ADMOD0H: ADC 输入模式控制寄存器 0 高位字**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DIFF15 ⁽¹⁾	SIGN15 ⁽¹⁾	DIFF14 ⁽¹⁾	SIGN14 ⁽¹⁾	DIFF13 ⁽¹⁾	SIGN13 ⁽¹⁾	DIFF12 ⁽¹⁾	SIGN12 ⁽¹⁾
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DIFF11 ⁽¹⁾	SIGN11 ⁽¹⁾	DIFF10 ⁽¹⁾	SIGN10 ⁽¹⁾	DIFF9 ⁽¹⁾	SIGN9 ⁽¹⁾	DIFF8 ⁽¹⁾	SIGN8 ⁽¹⁾
bit 7							bit 0

图注:

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

- bit (奇数) **DIFF<15:8>:** 相应模拟输入的差分模式位⁽¹⁾
 1 = 通道为差分模式
 0 = 通道为单端模式
- bit (偶数) **SIGN<15:8>:** 相应模拟输入的输出数据符号位⁽¹⁾
 1 = 通道输出数据为有符号数
 0 = 通道输出数据为无符号数

注 1: 可用的输入通道特定于器件。某些通道可能未实现。另外, 并非所有通道都支持差分模式。关于可用 SIGNx 和 DIFFx 位的信息, 请参见器件的数据手册。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-21: ADMOD1L: ADC 输入模式控制寄存器 1 低位字

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DIFF23 ⁽¹⁾	SIGN23 ⁽¹⁾	DIFF22 ⁽¹⁾	SIGN22 ⁽¹⁾	DIFF21 ⁽¹⁾	SIGN21 ⁽¹⁾	DIFF20 ⁽¹⁾	SIGN20 ⁽¹⁾
bit 15							bit 8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DIFF19 ⁽¹⁾	SIGN19 ⁽¹⁾	DIFF18 ⁽¹⁾	SIGN18 ⁽¹⁾	DIFF17 ⁽¹⁾	SIGN17 ⁽¹⁾	DIFF16 ⁽¹⁾	SIGN16 ⁽¹⁾
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit (奇数) **DIFF<23:16>:** 相应模拟输入的差分模式位⁽¹⁾

1 = 通道为差分模式

0 = 通道为单端模式

bit (偶数) **SIGN<23:16>:** 相应模拟输入的输出数据符号位⁽¹⁾

1 = 通道输出数据为有符号数

0 = 通道输出数据为无符号数

注 1: 可用的输入通道特定于器件。某些通道可能未实现。另外, 并非所有通道都支持差分模式。关于可用 SIGNx 和 DIFFx 位的信息, 请参见器件的数据手册。

寄存器 2-22: ADMOD1H: ADC 输入模式控制寄存器 1 高位字

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DIFF31 ⁽¹⁾	SIGN31 ⁽¹⁾	DIFF30 ⁽¹⁾	SIGN30 ⁽¹⁾	DIFF29 ⁽¹⁾	SIGN29 ⁽¹⁾	DIFF28 ⁽¹⁾	SIGN28 ⁽¹⁾
bit 15							bit 8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DIFF27 ⁽¹⁾	SIGN27 ⁽¹⁾	DIFF26 ⁽¹⁾	SIGN26 ⁽¹⁾	DIFF25 ⁽¹⁾	SIGN25 ⁽¹⁾	DIFF24 ⁽¹⁾	SIGN24 ⁽¹⁾
bit 7							bit 0

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为 0

-n = POR 时的值

1 = 置 1

0 = 清零

x = 未知

bit (奇数) **DIFF<31:24>:** 相应模拟输入的差分模式位⁽¹⁾

1 = 通道为差分模式

0 = 通道为单端模式

bit (偶数) **SIGN<31:24>:** 相应模拟输入的输出数据符号位⁽¹⁾

1 = 通道输出数据为有符号数

0 = 通道输出数据为无符号数

注 1: 可用的输入通道特定于器件。某些通道可能未实现。另外, 并非所有通道都支持差分模式。关于可用 SIGNx 和 DIFFx 位的信息, 请参见器件的数据手册。

dsPIC33/PIC24 系列参考手册

寄存器 2-23: ADIEL: ADC 中断允许寄存器低位字

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IE<15:8> ⁽¹⁾							
bit 15				bit 8			
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IE<7:0> ⁽¹⁾							
bit 7				bit 0			

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

bit 15-0 **IE<15:0>**: 中断允许位⁽¹⁾
1 = 允许相应通道的独立中断和共用中断
0 = 禁止相应通道的独立中断和共用中断

注 1: 可用通道特定于器件。某些 IEx 位可能未实现。关于可用通道的信息, 请参见器件的数据手册。

寄存器 2-24: ADIEH: ADC 中断允许寄存器高位字

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IE<31:24> ⁽¹⁾							
bit 15				bit 8			
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IE<23:16> ⁽¹⁾							
bit 7				bit 0			

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

bit 15-0 **IE<31:16>**: 中断允许位⁽¹⁾
1 = 允许相应通道的独立中断和共用中断
0 = 禁止相应通道的独立中断和共用中断

注 1: 可用通道特定于器件。某些 IEx 位可能未实现。关于可用通道的信息, 请参见器件的数据手册。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-25: ADSTATL: ADC 数据就绪状态寄存器低位字

R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0
AN15RDY ⁽¹⁾	AN14RDY ⁽¹⁾	AN13RDY ⁽¹⁾	AN12RDY ⁽¹⁾	AN11RDY ⁽¹⁾	AN10RDY ⁽¹⁾	AN9RDY ⁽¹⁾	AN8RDY ⁽¹⁾
bit 15							bit 8

R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0
AN7RDY ⁽¹⁾	AN6RDY ⁽¹⁾	AN5RDY ⁽¹⁾	AN4RDY ⁽¹⁾	AN3RDY ⁽¹⁾	AN2RDY ⁽¹⁾	AN1RDY ⁽¹⁾	AN0RDY ⁽¹⁾
bit 7							bit 0

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

bit 15-0 **AN15RDY:AN0RDY:** 相应模拟输入的数据就绪状态位⁽¹⁾
1 = 相应 ADCBUFx 寄存器中通道转换结果已就绪
0 = 通道转换结果未就绪

注 1: 可用通道特定于器件。某些 ANxRDY 可能未实现。关于可用通道的信息, 请参见器件的数据手册。

寄存器 2-26: ADSTATH: ADC 数据就绪状态寄存器高位字

R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0
AN31RDY ⁽¹⁾	AN30RDY ⁽¹⁾	AN29RDY ⁽¹⁾	AN28RDY ⁽¹⁾	AN27RDY ⁽¹⁾	AN26RDY ⁽¹⁾	AN25RDY ⁽¹⁾	AN24RDY ⁽¹⁾
bit 15							bit 8

R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0
AN23RDY ⁽¹⁾	AN22RDY ⁽¹⁾	AN21RDY ⁽¹⁾	AN20RDY ⁽¹⁾	AN19RDY ⁽¹⁾	AN18RDY ⁽¹⁾	AN17RDY ⁽¹⁾	AN16RDY ⁽¹⁾
bit 7							bit 0

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

bit 15-0 **AN31RDY:AN16RDY:** 相应模拟输入的数据就绪状态位⁽¹⁾
1 = 相应 ADCBUFx 寄存器中通道转换结果已就绪
0 = 通道转换结果未就绪

注 1: 可用通道特定于器件。某些 ANxRDY 可能未实现。关于可用通道的信息, 请参见器件的数据手册。

dsPIC33/PIC24 系列参考手册

寄存器 2-27: ADTRIGnL 和 ADTRIGnH: ADC 通道触发 n 选择寄存器低位字和高位字
 (其中 n 为寄存器编号 0 至 7)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	TRGSRC(x+1)<4:0> ⁽¹⁾				
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	TRGSRCx<4:0> ⁽¹⁾				
bit 7							bit 0

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

- bit 15-13 未实现: 读为 0
- bit 12-8 **TRGSRC(x+1)<4:0>**: 相应模拟输入 (x+1) 的触发源选择位⁽¹⁾
 11111 = 外部触发引脚
 00100-11110 = 器件特定的其他触发选项; 更多信息, 请参见器件的数据手册
 00011 = 保留
 00010 = 通用电平敏感软件触发
 00001 = 通用软件触发
 00000 = 未使能触发
- bit 7-5 未实现: 读为 0
- bit 4-0 **TRGSRCx<4:0>**: 相应模拟输入 x 的触发源选择位⁽¹⁾
 11111 = 外部触发引脚
 00100-11110 = 器件特定的其他触发选项; 更多信息, 请参见器件的数据手册
 00011 = 保留
 00010 = 通用电平敏感软件触发
 00001 = 通用软件触发
 00000 = 未使能触发

注 1: 可用通道特定于器件。某些 TRGSRCx<4:0> 位可能未实现。关于可用通道的信息, 请参见器件的数据手册。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-28: ADCAL0L: ADC 校准寄存器 0 的低位字⁽²⁾

R/HC/HS-0	U-0	U-0	U-0	r-0	R/W-0	R/W-0	R/W-0
CAL1RDY ⁽¹⁾	—	—	—	—	CAL1DIFF ⁽¹⁾	CAL1EN ⁽¹⁾	CAL1RUN ⁽¹⁾
bit 15						bit 8	

R/HC/HS-0	U-0	U-0	U-0	r-0	R/W-0	R/W-0	R/W-0
CAL0RDY ⁽¹⁾	—	—	—	—	CAL0DIFF ⁽¹⁾	CAL0EN ⁽¹⁾	CAL0RUN ⁽¹⁾
bit 7						bit 0	

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

- bit 15 **CAL1RDY:** 专用 ADC 内核 1 校准状态标志位⁽¹⁾
1 = 专用 ADC 内核校准完成
0 = 专用 ADC 内核校准正在进行
- bit 14-12 **未实现:** 读为 0
- bit 11 **保留:** 必须写为 0
- bit 10 **CAL1DIFF:** 专用 ADC 内核 1 差分模式校准位⁽¹⁾
1 = 专用 ADC 内核将在差分输入模式下校准
0 = 专用 ADC 内核将在单端输入模式下校准
- bit 9 **CAL1EN:** 专用 ADC 内核 1 校准使能位⁽¹⁾
1 = 专用 ADC 内核校准位 (CAL1RDY、CAL1DIFF 和 CAL1RUN) 可通过软件访问
0 = 禁止专用 ADC 内核校准位
- bit 8 **CAL1RUN:** 专用 ADC 内核 1 校准启动位⁽¹⁾
1 = 如果该位由软件置 1, 则将启动专用 ADC 内核校准周期; 该位由硬件自动清零
0 = 软件可启动下一个校准周期
- bit 7 **CAL0RDY:** 专用 ADC 内核 0 校准状态标志位⁽¹⁾
1 = 专用 ADC 内核校准完成
0 = 专用 ADC 内核校准正在进行
- bit 6-4 **未实现:** 读为 0
- bit 3 **保留:** 必须写为 0
- bit 2 **CAL0DIFF:** 专用 ADC 内核 0 差分模式校准位⁽¹⁾
1 = 专用 ADC 内核将在差分输入模式下校准
0 = 专用 ADC 内核将在单端输入模式下校准
- bit 1 **CAL0EN:** 专用 ADC 内核 0 校准使能位⁽¹⁾
1 = 专用 ADC 内核校准位 (CAL0RDY、CAL0DIFF 和 CAL0RUN) 可通过软件访问
0 = 禁止专用 ADC 内核校准位
- bit 0 **CAL0RUN:** 专用 ADC 内核 0 校准启动位⁽¹⁾
1 = 如果该位由软件置 1, 则将启动专用 ADC 内核校准周期; 该位由硬件自动清零
0 = 软件可启动下一个校准周期

- 注 1: 可用的专用 ADC 内核的数量特定于器件。某些 CALxRDY、CALxDIFF、CALxEN 和 CALxRUN 位可能未实现。关于可用 ADC 内核的信息, 请参见器件的数据手册。
- 2: 某些器件不需要校准功能。请参见具体器件的数据手册, 了解器件是否实现了该寄存器。

dsPIC33/PIC24 系列参考手册

寄存器 2-29: ADCAL0H: ADC 校准寄存器 0 的高位字⁽²⁾

R/HC/HS-0	U-0	U-0	U-0	r-0	R/W-0	R/W-0	R/W-0
CAL3RDY ⁽¹⁾	—	—	—	—	CAL3DIFF ⁽¹⁾	CAL3EN ⁽¹⁾	CAL3RUN ⁽¹⁾
bit 15							bit 8

R/HC/HS-0	U-0	U-0	U-0	r-0	R/W-0	R/W-0	R/W-0
CAL2RDY ⁽¹⁾	—	—	—	—	CAL2DIFF ⁽¹⁾	CAL2EN ⁽¹⁾	CAL2RUN ⁽¹⁾
bit 7							bit 0

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

- bit 15 **CAL3RDY:** 专用 ADC 内核 3 校准状态标志位⁽¹⁾
1 = 专用 ADC 内核校准完成
0 = 专用 ADC 内核校准正在进行
- bit 14-12 **未实现:** 读为 0
- bit 11 **保留:** 必须写为 0
- bit 10 **CAL3DIFF:** 专用 ADC 内核 3 差分模式校准位⁽¹⁾
1 = 专用 ADC 内核将在差分输入模式下校准
0 = 专用 ADC 内核将在单端输入模式下校准
- bit 9 **CAL3EN:** 专用 ADC 内核 3 校准使能位⁽¹⁾
1 = 专用 ADC 内核校准位 (CAL3RDY、CAL3DIFF 和 CAL3RUN) 可通过软件访问
0 = 禁止专用 ADC 内核校准位
- bit 8 **CAL3RUN:** 专用 ADC 内核 3 校准启动位⁽¹⁾
1 = 如果该位由软件置 1, 则将启动专用 ADC 内核校准周期; 该位由硬件自动清零
0 = 软件可启动下一个校准周期
- bit 7 **CAL2RDY:** 专用 ADC 内核 2 校准状态标志位⁽¹⁾
1 = 专用 ADC 内核校准完成
0 = 专用 ADC 内核校准正在进行
- bit 6-4 **未实现:** 读为 0
- bit 3 **保留:** 必须写为 0
- bit 2 **CAL2DIFF:** 专用 ADC 内核 2 差分模式校准位⁽¹⁾
1 = 专用 ADC 内核将在差分输入模式下校准
0 = 专用 ADC 内核将在单端输入模式下校准
- bit 1 **CAL2EN:** 专用 ADC 内核 2 校准使能位⁽¹⁾
1 = 专用 ADC 内核校准位 (CAL2RDY、CAL2DIFF 和 CAL2RUN) 可通过软件访问
0 = 禁止专用 ADC 内核校准位
- bit 0 **CAL2RUN:** 专用 ADC 内核 2 校准启动位⁽¹⁾
1 = 如果该位由软件置 1, 则将启动专用 ADC 内核校准周期; 该位由硬件自动清零
0 = 软件可启动下一个校准周期

- 注 1: 可用的专用 ADC 内核的数量特定于器件。某些 CALxRDY、CALxDIFF、CALxEN 和 CALxRUN 位可能未实现。关于可用 ADC 内核的信息, 请参见器件的数据手册。
- 2: 某些器件不需要校准功能。请参见具体器件的数据手册, 了解器件是否实现了该寄存器。

12位高速多SAR A/D转换器（ADC）

寄存器 2-30: ADCAL1L: ADC校准寄存器1的低位字⁽²⁾

R/HC/HS-0	U-0	U-0	U-0	r-0	R/W-0	R/W-0	R/W-0
CAL5RDY ⁽¹⁾	—	—	—	—	CAL5DIFF ⁽¹⁾	CAL5EN ⁽¹⁾	CAL5RUN ⁽¹⁾
bit 15						bit 8	

R/HC/HS-0	U-0	U-0	U-0	r-0	R/W-0	R/W-0	R/W-0
CAL4RDY ⁽¹⁾	—	—	—	—	CAL4DIFF ⁽¹⁾	CAL4EN ⁽¹⁾	CAL4RUN ⁽¹⁾
bit 7						bit 0	

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

- bit 15 **CAL5RDY:** 专用ADC内核5校准状态标志位⁽¹⁾
1 = 专用ADC内核校准完成
0 = 专用ADC内核校准正在进行
- bit 14-12 **未实现:** 读为0
- bit 11 **保留:** 必须写为0
- bit 10 **CAL5DIFF:** 专用ADC内核5差分模式校准位⁽¹⁾
1 = 专用ADC内核将在差分输入模式下校准
0 = 专用ADC内核将在单端输入模式下校准
- bit 9 **CAL5EN:** 专用ADC内核5校准使能位⁽¹⁾
1 = 专用ADC内核校准位（CAL5RDY、CAL5DIFF和CAL5RUN）可通过软件访问
0 = 禁止专用ADC内核校准位
- bit 8 **CAL5RUN:** 专用ADC内核5校准启动位⁽¹⁾
1 = 如果该位由软件置1, 则将启动专用ADC内核校准周期; 该位由硬件自动清零
0 = 软件可启动下一个校准周期
- bit 7 **CAL4RDY:** 专用ADC内核4校准状态标志位⁽¹⁾
1 = 专用ADC内核校准完成
0 = 专用ADC内核校准正在进行
- bit 6-4 **未实现:** 读为0
- bit 3 **保留:** 必须写为0
- bit 2 **CAL4DIFF:** 专用ADC内核4差分模式校准位⁽¹⁾
1 = 专用ADC内核将在差分输入模式下校准
0 = 专用ADC内核将在单端输入模式下校准
- bit 1 **CAL4EN:** 专用ADC内核4校准使能位⁽¹⁾
1 = 专用ADC内核校准位（CAL4RDY、CAL4DIFF和CAL4RUN）可通过软件访问
0 = 禁止专用ADC内核校准位
- bit 0 **CAL4RUN:** 专用ADC内核4校准启动位⁽¹⁾
1 = 如果该位由软件置1, 则将启动专用ADC内核校准周期。该位由硬件自动清零
0 = 软件可启动下一个校准周期

- 注 1: 可用的专用ADC内核的数量特定于器件。某些 CALxRDY、CALxDIFF、CALxEN 和 CALxRUN 位可能未实现。关于可用ADC内核的信息, 请参见器件的数据手册。
- 2: 某些器件不需要校准功能。请参见具体器件的数据手册, 了解器件是否实现了该寄存器。

dsPIC33/PIC24 系列参考手册

寄存器 2-31: ADCAL1H: ADC 校准寄存器 1 的高位字⁽²⁾

R/HC/HS-0	U-0	U-0	U-0	r-0	R/W-0	R/W-0	R/W-0
CSHRRDY	—	—	—	—	CSHRDIFF	CSHREN	CSHRRUN
bit 15							bit 8

R/HC/HS-0	U-0	U-0	U-0	r-0	R/W-0	R/W-0	R/W-0
CAL6RDY ⁽¹⁾	—	—	—	—	CAL6DIFF ⁽¹⁾	CAL6EN ⁽¹⁾	CAL6RUN ⁽¹⁾
bit 7							bit 0

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

- bit 15 **CSHRRDY:** 共用 ADC 内核校准状态标志位
1 = 共用 ADC 内核校准完成
0 = 共用 ADC 内核校准正在进行
- bit 14-12 **未实现:** 读为 0
- bit 11 **保留:** 必须写为 0
- bit 10 **CSHRDIFF:** 共用 ADC 内核差分模式校准位
1 = 共用 ADC 内核将在差分输入模式下校准
0 = 共用 ADC 内核将在单端输入模式下校准
- bit 9 **CSHREN:** 共用 ADC 内核校准使能位
1 = 共用 ADC 内核校准位 (CSHRRDY、CSHRDIFF 和 CSHRRUN) 可通过软件访问
0 = 禁止共用 ADC 内核校准位
- bit 8 **CSHRRUN:** 共用 ADC 内核校准启动位
1 = 如果该位由软件置 1, 则将启动共用 ADC 内核校准周期; 该位由硬件自动清零
0 = 软件可启动下一个校准周期
- bit 7 **CAL6RDY:** 专用 ADC 内核 6 校准状态标志位⁽¹⁾
1 = 专用 ADC 内核校准完成
0 = 专用 ADC 内核校准正在进行
- bit 6-4 **未实现:** 读为 0
- bit 3 **保留:** 必须写为 0
- bit 2 **CAL6DIFF:** 专用 ADC 内核 6 差分模式校准位⁽¹⁾
1 = 专用 ADC 内核将在差分输入模式下校准
0 = 专用 ADC 内核将在单端输入模式下校准
- bit 1 **CAL6EN:** 专用 ADC 内核 6 校准使能位⁽¹⁾
1 = 专用 ADC 内核校准位 (CAL6RDY、CAL6DIFF 和 CAL6RUN) 可通过软件访问
0 = 禁止专用 ADC 内核校准位
- bit 0 **CAL6RUN:** 专用 ADC 内核 6 校准启动位⁽¹⁾
1 = 如果该位由软件置 1, 则将启动专用 ADC 内核校准周期; 该位由硬件自动清零
0 = 软件可启动下一个校准周期

- 注 1:** 可用的专用 ADC 内核的数量特定于器件。某些 CALxRDY、CALxDIFF、CALxEN 和 CALxRUN 位可能未实现。关于可用 ADC 内核的信息, 请参见器件的数据手册。
- 2:** 某些器件不需要校准功能。请参见具体器件的数据手册, 了解器件是否实现了该寄存器。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-32: ADCMPnCON: ADC 数字比较器 n 控制寄存器⁽¹⁾

U-0	U-0	U-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0	R/HC/HS-0
—	—	—	CHNL4	CHNL3	CHNL2	CHNL1	CHNL0
bit 15							bit 8

R/W-0	R/W-0	R/HC/HS-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMPEN	IE	STAT	BTWN	HIHI	HILO	LOHI	LOLO
bit 7							bit 0

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

bit 15-13 未实现: 读为 0

bit 12-8 **CHNL<4:0>**: 输入通道编号位
 这些位用于标识引起比较器事件的模拟输入。
 11111 = AN31
 •
 •
 •
 00001 = AN1
 00000 = AN0

bit 7 **CMPEN**: 数字比较器使能位
 1 = 使能比较器
 0 = 禁止比较器并清零 STAT 状态位

bit 6 **IE**: 比较器中断允许位
 1 = 当比较器检测到比较事件时, 将产生独立中断和共用中断
 0 = 不会为比较器产生独立中断和共用中断

bit 5 **STAT**: 比较器事件状态位
 当从 CHNL<4:0> 位读取通道编号时, 该位由硬件清零。
 1 = 自上一次读取 CHNL<4:0> 位后, 已检测到比较事件
 0 = 自上一次读取 CHNL<4:0> 位后, 未检测到比较事件

bit 4 **BTWN**: 介于下限/上限之间的比较器事件位
 1 = 当 $ADCMPnLO \leq ADCBUFx < ADCMPnHI$ 时, 产生数字比较器事件
 0 = 当 $ADCMPnLO \leq ADCBUFx < ADCMPnHI$ 时, 不产生数字比较器事件

bit 3 **HIHI**: 上限/上限比较器事件位
 1 = 当 $ADCBUFx \geq ADCMPnHI$ 时, 产生数字比较器事件
 0 = 当 $ADCBUFx \geq ADCMPnHI$ 时, 不产生数字比较器事件

bit 2 **HILO**: 上限/下限比较器事件位
 1 = 当 $ADCBUFx < ADCMPnHI$ 时, 产生数字比较器事件
 0 = 当 $ADCBUFx < ADCMPnHI$ 时, 不产生数字比较器事件

bit 1 **LOHI**: 下限/上限比较器事件位
 1 = 当 $ADCBUFx \geq ADCMPnLO$ 时, 产生数字比较器事件
 0 = 当 $ADCBUFx \geq ADCMPnLO$ 时, 不产生数字比较器事件

bit 0 **LOLO**: 下限/下限比较器事件位
 1 = 当 $ADCBUFx < ADCMPnLO$ 时, 产生数字比较器事件
 0 = 当 $ADCBUFx < ADCMPnLO$ 时, 不产生数字比较器事件

注 1: 可用的数字比较器的数量特定于器件。关于可用数字比较器的信息, 请参见器件的数据手册。

寄存器2-33: ADCMPnENL: ADC数字比较器n通道使能寄存器的低位字⁽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
CMPEN<15:8> ⁽²⁾							
bit 15 bit 8							

R/W/0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMPEN<7:0> ⁽²⁾							
bit 7 bit 0							

图注:

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

bit 15-0 **CMPEN<15:0>**: 相应输入通道的比较器使能位⁽²⁾

1 = 比较器使用相应通道的转换结果

0 = 比较器不使用相应通道的转换结果

注 1: 可用的数字比较器的数量特定于器件。关于可用数字比较器的信息，请参见器件的数据手册。

2: 可用通道特定于器件。某些CMPENx位可能未实现。关于可用通道的信息，请参见器件的数据手册。

寄存器2-34: ADCMPnENH: ADC数字比较器n通道使能寄存器的高位字⁽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
CMPEN<31:24> ⁽²⁾							
bit 15							
bit 8							

R/W/0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMPEN<23:16> ⁽²⁾							
bit 7 bit 0							

图注:

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

bit 15-0 **COMPEN<31:16>**: 相应输入通道的比较器使能位⁽²⁾

1 = 比较器使用相应通道的转换结果

0 = 比较器不使用相应通道的转换结果

注 1: 可用的数字比较器的数量特定于器件。关于可用数字比较器的信息，请参见器件的数据手册。

2: 可用通道特定于器件。某些CMPENx位可能未实现。关于可用通道的信息，请参见器件的数据手册。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-35: ADFLnCON: ADC 数字滤波器 n 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/HC/HS-0
FLEN	MODE1	MODE0	OVRSAM2	OVRSAM1	OVRSAM0	IE	RDY
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	FLCHSEL4	FLCHSEL3	FLCHSEL2	FLCHSEL1	FLCHSEL0
bit 7							bit 0

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

- bit 15 **FLEN:** 滤波器使能位
1 = 使能滤波器
0 = 禁止滤波器并清零 RDY 位
- bit 14-13 **MODE<1:0>:** 滤波器模式位
11 = 平均模式
10 = 保留
01 = 保留
00 = 过采样模式
- bit 12-10 **OVRSAM<2:0>:** 滤波器平均/过采样率位
如果 MODE<1:0> = 00:
111 = 128x (ADFLnDAT 寄存器中的 16 位结果)
110 = 32x (ADFLnDAT 寄存器中的 15 位结果)
101 = 8x (ADFLnDAT 寄存器中的 14 位结果)
100 = 2x (ADFLnDAT 寄存器中的 13 位结果)
011 = 256x (ADFLnDAT 寄存器中的 16 位结果)
010 = 64x (ADFLnDAT 寄存器中的 15 位结果)
001 = 16x (ADFLnDAT 寄存器中的 14 位结果)
000 = 4x (ADFLnDAT 寄存器中的 13 位结果)
如果 MODE<1:0> = 11 (ADFLnDAT 寄存器中的 12 位结果):
111 = 256x
110 = 128x
101 = 64x
100 = 32x
011 = 16x
010 = 8x
001 = 4x
000 = 2x
- bit 9 **IE:** 滤波器中断允许位
1 = 当滤波器结果就绪时, 将产生独立中断和共用中断
0 = 不会为滤波器产生独立中断和共用中断
- bit 8 **RDY:** 过采样滤波器数据就绪标志位
当从 ADFLnDAT 寄存器读取结果后, 该位由硬件清零。
1 = ADFLnDAT 寄存器中的数据就绪
0 = ADFLnDAT 寄存器已被读取, ADFLnDAT 寄存器中的新数据未就绪
- bit 7-5 **未实现:** 读为 0

注 1: 可用的过采样滤波器的数量特定于器件。关于可用过采样滤波器的信息, 请参见器件的数据手册。

寄存器 2-35: **ADFLnCON: ADC 数字滤波器 n 控制寄存器⁽¹⁾** (续)

bit 4-0 **FLCHSEL<4:0>**: 过采样滤波器输入通道选择位
11111 = AN31
•
•
•
00001 = AN1
00000 = AN0

注 1: 可用的过采样滤波器的数量特定于器件。关于可用过采样滤波器的信息，请参见器件的数据手册。

12 位高速多 SAR A/D 转换器 (ADC)

寄存器 2-36: ADCSSL: CVD 扫描选择寄存器低位字⁽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
CSS<15:8> ⁽²⁾							
bit 15				bit 8			

R/W/0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS<7:0> ⁽²⁾							
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为0

-n = POR时的值

1 = 置1

0 = 清零

x = 未知

bit 15-0 **CSS<15:0>**: 相应模拟输入的CVD扫描使能位⁽²⁾

1 = CVD扫描包含共用内核模拟输入

0 = CVD不扫描共用内核模拟输入

- 注 1: 如果器件未实现 CVD 功能, 则该寄存器不可用。请参见具体器件的数据手册, 了解器件是否具有 CVD 功能。
- 2: ADC 共用内核的可用通道取决于器件。某些 CSSx 位可能未实现。请参见具体器件的数据手册, 了解可用通道信息。

寄存器 2-37: ADCSSH: CVD 扫描选择寄存器高位字⁽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
CSS<31:24> ⁽²⁾							
bit 15				bit 8			

R/W/0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS<23:16> ⁽²⁾							
bit 7				bit 0			

图注:

R = 可读位

W = 可写位

U = 未实现位, 读为0

-n = POR时的值

1 = 置1

0 = 清零

x = 未知

bit 15-0 **CSS<31:16>**: 相应模拟输入的CVD扫描使能位⁽²⁾

1 = CVD扫描包含共用内核模拟输入

0 = CVD不扫描共用内核模拟输入

- 注 1: 如果器件未实现 CVD 功能, 则该寄存器不可用。请参见具体器件的数据手册, 了解器件是否具有 CVD 功能。
- 2: ADC 共用内核的可用通道取决于器件。某些 CSSx 位可能未实现。请参见具体器件的数据手册, 了解可用通道信息。

3.0 转换序列

采用 12 位高速多 SAR ADC 的模数转换过程涉及以下三个步骤：

1. 采样输入信号。
2. 捕捉输入信号（保持）并将其传送到转换器。
3. 将模拟信号转换为对应的数字形式。

输入信号采样涉及为采样保持（S/H）电路中的电容充电。采样时间必须足够长，才能使电容充电到与输入电压相等的值。在适当的时间，输入将与电容断开，接着模拟电压将传送到转换器。随后，转换器会对模拟信号进行数字化并提供相应结果。

转换器需要时钟源和参考电压。时钟和参考电压源以及时钟预分频比均是可选的。

4.0 ADC 操作

4.1 SAR ADC 内核

可用的专用 SAR ADC 内核的数量特定于器件。更多信息，请参见具体器件的数据手册。

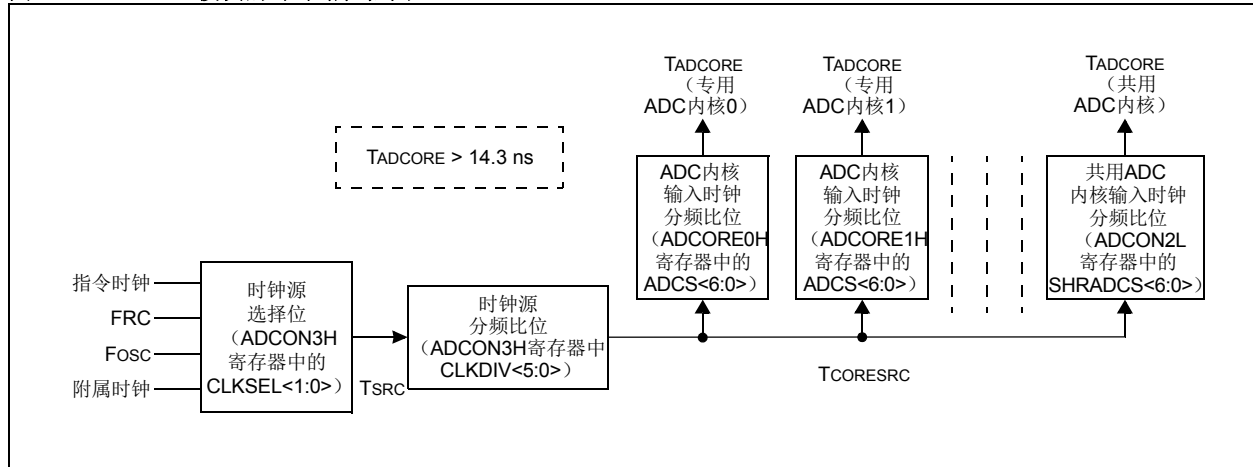
该模块可实现最多 8 个独立的 SAR ADC 内核。它允许同时从多个模拟输入采样信号。前 7 个 SAR ADC 内核（0 至 6）称为专用内核，因为每个内核都有一个专用的模拟通道。每个通道都处于连接状态，ADC 内核连续采样（跟随）输入信号电压。只有当转换启动时，专用 ADC 内核的通道才会断开。多个专用的独立 ADC 内核允许应用同时采样与专用 ADC 内核相关的模拟通道并将采样的信号转换为一个“快照”。

最后一个 SAR ADC 内核称为共用内核，因为它被一些与专用 ADC 内核不相关的模拟输入共用。对于此内核，要采样的模拟通道和采样过程均由 ADC 模块控制。当未进行转换时，所有输入均与共用 ADC 内核断开。ADC 将根据触发事件连接触发信号中定义的模拟输入并在指定的时间段内采样输入信号。采样完成后，模拟输入将再次断开，然后执行转换。

4.2 ADC 时钟

ADC 模块具有不同的时钟源选项。可通过 ADCON3H 寄存器中的 CLKSEL<1:0> 位选择时钟源。所选时钟源的周期为 T_{SR}C，可通过 ADCON3H 寄存器中的 CLKDIV<5:0> 位指定的分频比进行分频。经过该分频比分频后，所产生的周期为 T_{CORE}SR_C 的时钟将进入每个 SAR ADC 内核。每个 ADC 内核均有自己的时钟分频比，专用内核通过相应 ADCOREnH 寄存器的 ADCS<6:0> 位配置；共用 ADC 内核通过 ADCON2L 寄存器中的 SHRADCS<6:0> 位配置。经过分频后，各个 ADC 内核的时钟周期 T_{ADC}CORE 可能会有所不同。每个 SAR ADC 内核的最大工作时钟频率限制为 70 MHz。因此，所选的时钟设置必须能够提供大于 14.3 ns 的内核时钟周期 T_{ADC}CORE。图 4-1 给出了模块的时钟路径图。

图 4-1: ADC 模块的时钟路径框图



4.3 ADC 分辨率

每个 SAR ADC 内核的分辨率可通过 ADCOREnH 寄存器中的 RES<1:0> 位（对于专用内核）以及 ADCON1H 寄存器中的 SHRRES<1:0> 位（对于共用内核）单独设置。

根据设置的不同，ADC 的分辨率可以是 12 位、10 位、8 位或 6 位。复位后，所有内核均默认配置为 12 位分辨率。

4.4 采样和转换时序

所有ADC内核的转换时间均取决于RES<1:0>或SHRRES<1:0>位选择的分辨率。转换所需的时间由公式4-1定义。

公式4-1: 转换时间

$$\text{转换时间} = 8 \cdot T_{\text{CORESRC}} + (\text{位分辨率} + 2.5) \cdot T_{\text{ADCORE}}$$

例如，如果ADC内核配置为12位分辨率，则该内核的转换时间将为：

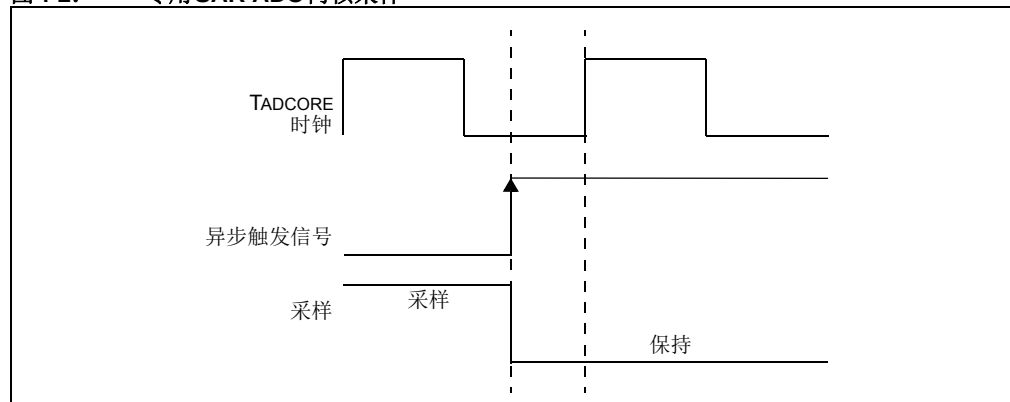
$$8 \cdot T_{\text{CORESRC}} + 14.5 \cdot T_{\text{ADCORE}}。$$

如果几个ADC内核同时得出转换结果，则低优先级的内核将额外等待一个TCORESRC周期，以供各个高优先级的内核存储各自的结果。

此外，如果使能降低噪声功能，转换时间可进一步延长。如果ADCON1L寄存器中的NRE位置1，将对转换结束时间进行调整，以降低ADC内核之间的噪声干扰。根据正在进行转换的内核数量以及输入的优先级，可额外插入几个TAD，这使转换时间略微不确定。并非所有器件都具有降低噪声功能，NRE位可能未实现。请参见具体器件的数据手册了解更多信息。

使用多个专用SAR ADC内核时，可同时捕捉多个模拟信号。每个专用内核在采样模式下连续跟踪输入信号，直至发生异步触发事件。触发事件将导致专用内核立即停止采样，进入保持状态。请注意，在默认情况下，用于结束采样的触发事件以与ADC内核时钟异步的方式发生。当S/H电路立即进入保持状态后，异步触发信号必须与ADC时钟同步，在将转换请求发给SAR之前最多经过一个ADC时钟边沿（图4-2）。

图4-2: 专用SAR ADC内核采样

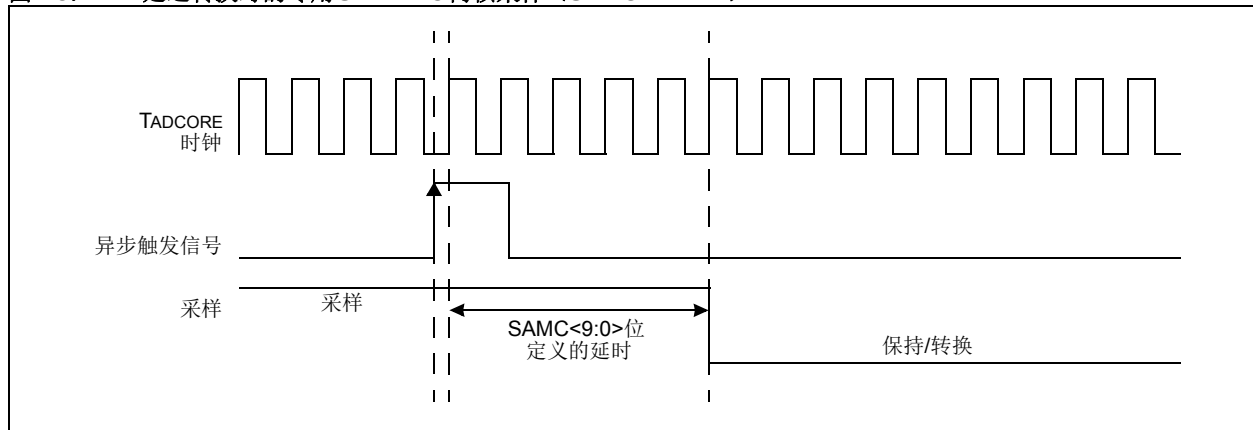


如果将周期性触发源与专用内核配合使用，则总采样时间取决于触发速率。触发速率不得违反必需采样时间的要求。更多信息，请参见第4.4.1节“采样时间要求”。

12 位高速多 SAR A/D 转换器（ADC）

另一个专用 SAR ADC 内核采样选项是延迟转换。该功能通过 ADCON4L 寄存器中的 SAMCxEN 位控制。当 SAMCxEN = 1 时，将在触发事件与转换启动之间插入延时（图 4-3）。在此期间，内核继续采样输入信号。延时由相应专用 ADC 内核 n 控制寄存器的低位字 ADCOREnL 中的 SAMC<9:0> 位定义。该延时应与数字滤波器或电平敏感触发信号配合使用来确保最短采样时间。无论 SAMCxEN 位为何值，ADCOREnL 寄存器中的 SAMC<9:0> 位限制触发之间的时间。如果触发之间的时间比 SAMC<9:0> 位指定的时间短，触发将被延时。

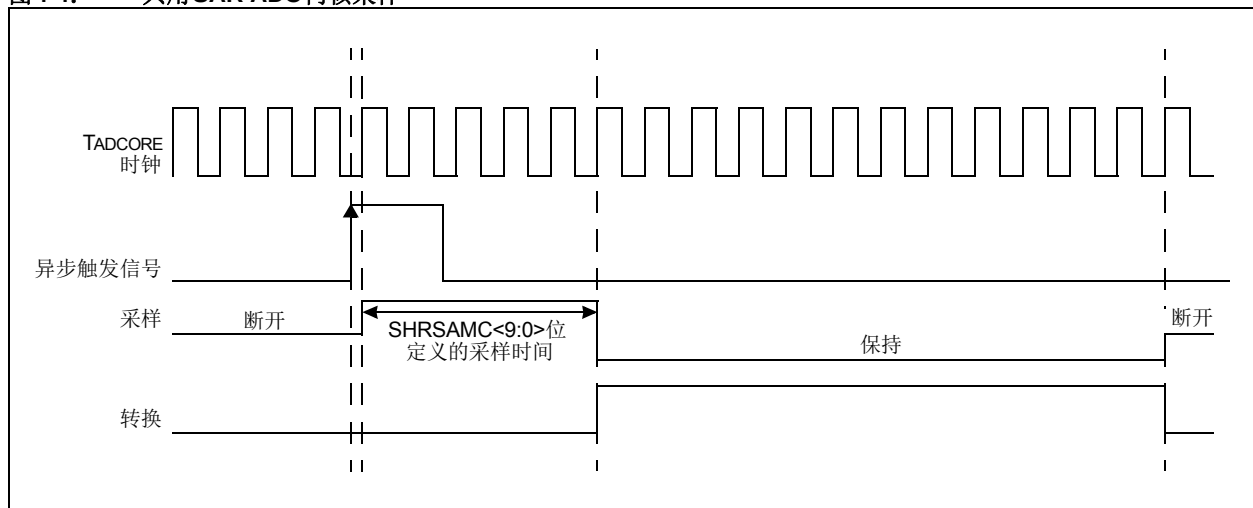
图 4-3: 延迟转换时的专用 SAR ADC 内核采样 (SAMCxEN = 1)



与专用 ADC 内核不同，共用 ADC 内核的触发事件使用 ADCON2H 寄存器中的 SHRSAMC<9:0> 位指定的采样时间来启动采样过程。信号被采样指定的 ADC 内核时钟（TADCORE）数后，S/H 立即进入保持状态，并且会发出转换请求（如图 4-4 所示）。

当定期触发共用 ADC 内核的单输入时，触发速率不得超过采样时间与转换时间之和。无法保证共用 ADC 内核的转换请求立即得到处理。共用内核的转换请求按优先级顺序进行处理。

图 4-4: 共用 SAR ADC 内核采样

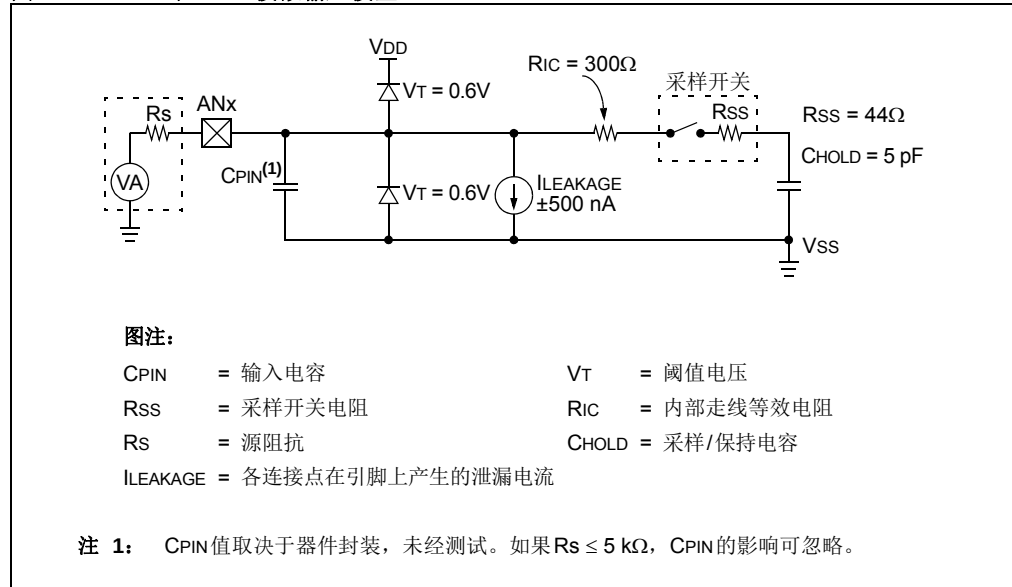


4.4.1 采样时间要求

图4-5给出了12位高速多SAR ADC的模拟输入模型。模数转换的总采集时间是保持电容（CHOLD）充电时间的函数。

为了使ADC模块达到规定的精度，必须让保持电容（CHOLD）充分充电至模拟输入引脚上的电压。模拟输出信号源阻抗（ R_s ）、内部走线等效电阻（ R_{ic} ）和内部采样开关阻抗（ R_{ss} ）共同直接影响着电容CHOLD充电所需的时间。因此合并的模拟源阻抗必须足够小，以便在选择的采样时间内对保持电容充分充电。在每次采样操作前，内部保持电容将处于放电状态。

图4-5: 12位ADC模拟输入模型



4.5 参考电压

参考电压选项因器件而异。关于特定选项的信息，请参见器件的数据手册。

ADCON3L寄存器中的ADC参考电压选择位REFSEL<2:0>用于为所有SAR ADC内核选择参考电压。此外，ADC模块还取决于内部带隙电路的电压。当参考电压和带隙均准备好进行工作时，ADCON2H寄存器中的REFRDY位置1。当模块使能（ADON（ADCON1L<15>）= 1）时，参考电压源无法改变。如果ADC参考或AVDD电源发生改变或中断，则ADCON2H寄存器中的REFERR位置1。检测到参考电压故障事件后，必须重新校准ADC模块。参考电压就绪和参考电压故障事件分别由REFRDY和REFERR位指示，当ADCON2L寄存器中相应的REFCIE和REFERCIE位置1时，会产生共用中断。为了避免错误的故障中断，REFERCIE位必须仅在模块使能（ADON = 1）后置1。

4.6 模拟输入通道

可用的输入通道的数量特定于器件。更多信息，请参见具体器件的数据手册。

4.6.1 配置模拟端口引脚

I/O 端口的 ANSELx 寄存器与模拟输入关联，用于将相应引脚配置为模拟引脚。当 ANSELx 位 = 1 时，相应的引脚将被配置为模拟输入。当 ANSELx 位 = 0 时，引脚将被设置为数字控制。当引脚被配置为模拟输入时，相关的端口 I/O 数字输入缓冲器被禁止，因此不消耗电流。ANSELx 寄存器将在器件退出复位状态时置 1，这将使 ADC 输入引脚被配置为模拟输入（默认）。

TRISx 寄存器用于控制端口引脚的数字功能。对于需配置为模拟输入的端口引脚，必须将特定 TRISx 寄存器中的相应位置 1，才能将该引脚配置为输入。如果通过清零 TRISx 位将与 ADC 输入关联的 I/O 引脚配置为输出，则端口的数字输出电平将被转换。在器件复位后，所有 TRISx 位均置 1。有关端口引脚配置的更多信息，请参见具体器件数据手册中的“**I/O 端口**”一章。

注： 如果 PORT 寄存器位的相应引脚配置为模拟输入，则该位将读为 0。

4.6.2 单端和伪差分输入选项及输出结果格式

A/D 转换器同时包含单端和伪差分通道。任何模拟引脚上的输入电压不得低于模拟地电平 AVSS 且不得超过模拟电源电压 AVDD（对于单端模式或伪差分模式）。伪差分通道具有反相输入和同相输入。在伪差分模式下，用作反相输入的模拟引脚对于每个 ADC 内核是固定的。有关特定 ADC 内核的反相输入编号，请参见特定器件的数据手册。为了确保在伪差分模式下正确操作，一个输入（反相或同相）的电压范围为 VR- 至 VR+，另一个输入的电压限制为 $(VR+ + VR-)/2 \pm 150 \text{ mV}$ 左右（其中 VR+ 和 VR- 分别为正参考电压和负参考电压）。

每个 ADC 输入通道的单端模式或伪差分模式由 ADMODnL 或 ADMODnH 寄存器中的 DIFFx 位定义。如果相应的 DIFFx 位置 1，则通道为差分模式。如果该 DIFFx 位清零，则通道为单端模式。此外，每个通道的数据输出格式可通过 ADMODnL 或 ADMODnH 寄存器中的 SIGNx 位单独设置。如果 SIGNx 位置 1，则输入的转换结果将以有符号值的形式写入相应的结果缓冲区 ADCBUFx。如果 SIGNx 位清零，则转换结果为无符号值。表 4-1 给出了输入配置选项的详细信息。

表 4-1: 输入配置

输入		输入电压		输出代码（FORM 位 （ADCON1H<7>）= 0 时）
伪差分模式位 （DIFFx）	有符号结果位 （SIGNx）	VINP = 同相输入上的电压，VINN = 反相输入上的电压， VR+ = 正参考电压，VR- = 负参考电压		
1	1	最小值	$VINP \leq VR-; \quad VINN = (VR+ + VR-)/2$	-1024
		最大值	$VINP \geq VR+; \quad VINN = (VR+ + VR-)/2$	+1023
1	0	最小值	$VINP \leq VR-; \quad VINN = (VR+ + VR-)/2$	+1024
		最大值	$VINP \geq VR+; \quad VINN = (VR+ + VR-)/2$	+3071
0	1	最小值	$VINP \leq VR-$	-2048
		最大值	$VINP \geq VR+$	+2047
0	0	最小值	$VINP \leq VR-$	0
		最大值	$VINP \geq VR+$	+4095

4.6.3 为专用ADC内核选择模拟输入

为提高专用ADC内核的灵活性，每个内核都配有多个输入。其中可能包括模拟引脚以及放大器等其他模拟模块的输出。输入选项通过ADCON4H寄存器中的CxCHS<1:0>位选择。

输入选项特定于器件。更多信息，请参见器件的数据手册。CxCHS<1:0>位将不同的模拟输入与专用ADC内核进行物理连接，但它们不会更改触发源。专用ADC内核仅接受来自CxCHS<1:0> = 00选项所对应的通道的触发信号，而与写入CxCHS<1:0>位的值无关。

注： CxCHS<1:0>位不会更改专用内核的触发源通道。触发源由分配给内核的默认通道定义。通常情况下，对于专用内核0，触发信号始终由TRGSR0<4:0>位选择；而对于专用内核1，触发信号始终由TRGSCR1<4:0>位选择，依此类推。关于触发选项的详细信息，请参见器件的数据手册。此外，无论CxCHS<1:0>位为何值，专用内核的转换结果：对于内核0，存储在ADCBUF0中；对于内核1，存储在ADCBUF1中。

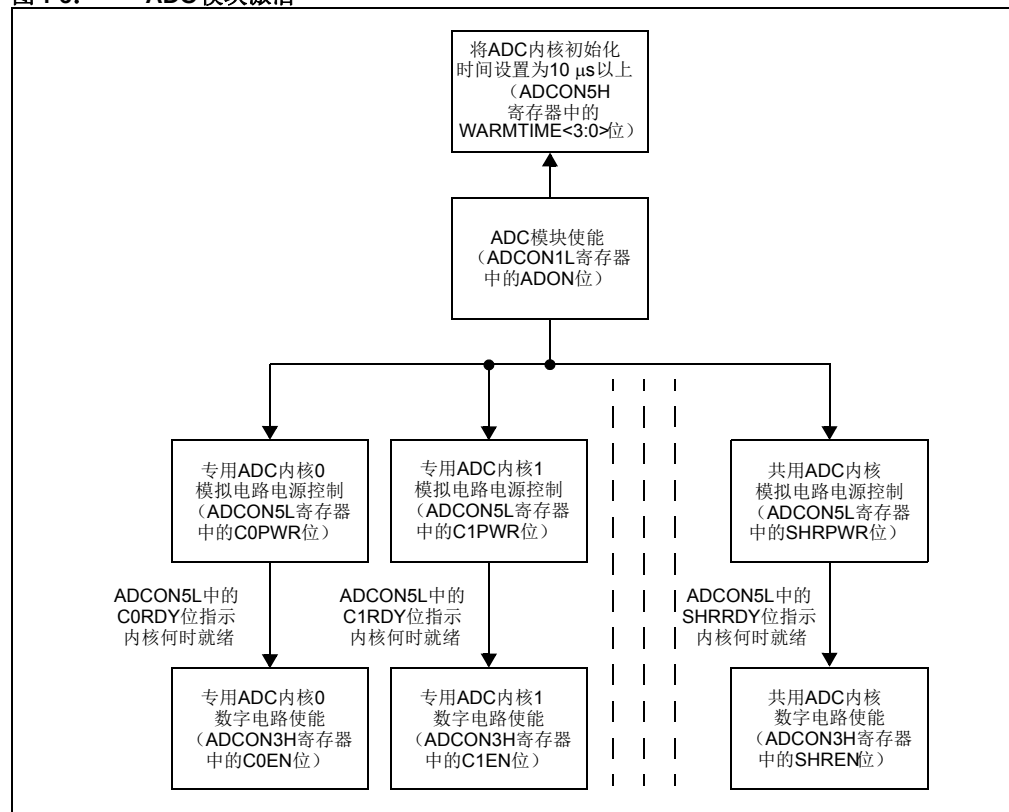
4.6.4 输入优先级

为解决不同通道同时请求输入转换的问题，共用ADC内核采用了自然顺序优先级方案。该优先级方案是固定的且由输入通道编号定义，即编号最小的通道优先级最高。也就是说，按模拟通道编号的升序顺序进行输入转换，编号最小的通道最先转换。

4.7 使能ADC

ADC模块有多个激活层级（见图4-6）。

图4-6: ADC模块激活



ADCON1L 寄存器中的 ADON 位将 ADC 模块作为一个整体来使能。之后，每个 SAR ADC 内核应单独开关。ADCON5L 寄存器中的 CxPWR 和 SHRPWR 位分别控制专用和共用 ADC 内核的模拟电路。当 CxPWR 或 SHRPWR 位置 1 时，将插入上电延时以稳定模拟电路。延时值由 ADCON5H 寄存器中的 WARMTIME<3:0> 位定义。经过延时后，ADCON5L 寄存器中相应的 CxRDY 或 SHRRDY 位将置 1 以指示 ADC 内核准备好开始工作。此时，ADC 内核不接收触发信号。要使能 ADC 内核数字电路，必须相应地为专用和共用 ADC 内核将 ADCON3H 寄存器中的 CxEN 或 SHREN 位置 1。当 CxEN 或 SHREN 位置 1 时，相应的 SAR ADC 内核将完全正常工作。SAR ADC 内核需要 10 μ s 的初始化时间。必须对 WARMTIME<3:0> 位进行相应配置以提供足够的初始化时间。

要使能 ADC 模块，应执行以下步骤：

1. 将 ADCON5H 寄存器中的 WARMTIME<3:0> 位置 1 以提供至少 10 μ s 的 ADC 内核初始化时间。
2. 将 ADCON1L 寄存器的 ADON 位置 1。
3. 为所选的 ADC 内核将 ADCON5L 寄存器中的 CxPWR 和/或 SHRPWR 位置 1。
4. 为所选的 ADC 内核轮询 ADCON5L 寄存器中的 CxRDY 和/或 SHRRDY 位，直到它们置 1。
5. 为所选的 ADC 内核将 ADCON3H 寄存器中的 CxEN 和/或 SHREN 位置 1。

模拟电路就绪事件由 CxRDY 和 SHRRDY 位指示，可用于产生共用 ADC 中断。为此，ADCON5H 寄存器中相应的 CxCIE 和/或 SHRCIE 位必须置 1。

4.8 校准

在某些器件上，只要模块被使能（见第 4.7 节“使能 ADC”），就必须对所有 SAR ADC 内核执行校准。请参见具体器件的数据手册，了解是否需要校准以及是否实现了校准寄存器 ADCALnL 和 ADCALnH。ADCALnL 和 ADCALnH 寄存器用于控制所有内核的校准过程。每个内核必须执行两个校准程序。一个程序用于单端输入模式（专用内核的 CALxDIFF 位和/或共用内核的 CSHRDIF 位清零时），另一个程序用于差分模式（CALxDIFF 和/或 CSHRDIF 位置 1 时）。如果内核仅使用单端模式，可跳过差分校准。同样地，如果内核仅使用差分输入，则无需单端校准。要进入校准模式，应将 CALxEN 位和/或 CSHREN 位（分别用于专用内核和共用内核）置 1。之后，应将 CALxRUN 和/或 CSHRRUN 位置 1 以执行校准。这些位由硬件清零后可允许下一个校准周期。此外，当 CALxRUN 和/或 CSHRRUN 位置 1 时，相应的 CALxRDY 和/或 CSHRRDY 位将清零。应用程序应轮询 CALxRDY 和 CSHRRDY 位来检测校准过程是否结束。校准后，CALxEN 和/或 CSHREN 位必须清零才能使所有 SAR ADC 内核恢复正常工作。

要执行 SAR ADC 内核校准，应执行以下步骤：

1. 将 CALxEN 和/或 CSHREN 位置 1。
2. 将 CALxDIFF 和/或 CSHRDIF 位清零。
3. 将 CALxRUN 和/或 CSHRRUN 位置 1。
4. 轮询 CALxRDY 和/或 CSHRRDY 位，直到它们置 1。
5. 将 CALxDIFF 和/或 CSHRDIF 位置 1。
6. 将 CALxRUN 和/或 CSHRRUN 位置 1。
7. 轮询 CALxRDY 和 CSHRRDY 位，直到它们置 1。
8. 将 CALxEN 和/或 CSHREN 位清零。

ADC 模块使能后，当以下 ADC 选项发生更改时需要重新校准：

- 参考电压 (REFSEL<2:0> 位)
- 时钟 (CLKSEL<1:0>、CLKDIV<5:0>、SHRADCS<6:0> 和 ADCS<6:0> 位)

4.9 触发

有四种方法可发起转换请求：

- 独立输入触发
- 共用软件触发
- 共用电平敏感软件触发
- 独立输入软件触发（单触发）

4.9.1 独立输入触发

应用程序可通过ADTRIGNL或ADTRIGNH寄存器中的TRGSRcn<4:0>位单独为每个模拟输入指定独立转换触发源。典型触发源可能包括通用定时器、输出比较模块、PWM发生器、比较器、外部引脚、共用软件触发信号和共用电平敏感软件触发信号。每个触发源由TRGSRcn<4:0>位选择，并且可通过ADLVLTRGL或ADLVLTRGH寄存器中的LVLENx位为相应通道设置为边沿敏感或电平敏感。如果LVLENx位置1，则只要触发信号置为有效，便会连续触发ADC内核。如果LVLENx位清零，则将仅在触发信号跳变时触发一次ADC内核。当专用SAR ADC内核使用电平敏感触发时，应将ADCON4L寄存器中相应的SAMCxEN位置1以在两次转换间预留足够的采样时间。采样时间由ADCOREnL寄存器中相应的SAMC<9:0>位定义。

4.9.2 通道扫描

如果需要扫描多个模拟通道，则可通过用ADTRIGNL或ADTRIGNH寄存器中的TRGSRcn<4:0>位为每个通道选择同一触发源来实现。

4.9.3 共用软件触发

任何模拟输入的转换均可通过软件共用触发位SWCTRG（ADCON3L<6>）来触发。当ADTRIGNL或ADTRIGNH寄存器中相应的通道位TRGSRcn<4:0>设置为00001时，将选择该选项。当SWCTRG位由软件置1时，将触发相应的模拟输入。该位由硬件自动清零，从而允许软件根据需要触发另一次转换。

4.9.4 共用电平敏感软件触发

任何模拟输入的转换均可通过软件电平敏感共用触发位SWLCTRG（ADCON3L<7>）来触发。要选择该选项，必须为相应输入通道将ADTRIGNL或ADTRIGNH寄存器中的TRGSRcn<4:0>位设置为00010。ADLVLTRGL或ADLVLTRGH寄存器中的LVLENx位也必须置1。当SWLCTRG位置1时，将连续触发相应的模拟输入，直到SWLCTRG位由软件清零。当专用SAR ADC内核使用电平敏感触发时，应将ADCON4L寄存器中相应的SAMCxEN位置1以在两次转换间预留足够的采样时间。采样时间由ADCOREnL寄存器中相应的SAMC<9:0>位定义。

4.9.5 独立通道软件触发

应用程序可在程序执行期间的任意时刻明确请求对任何选定的模拟输入进行单次转换，而无需更改ADC的触发源配置。要转换的输入应由ADCON3L寄存器中的CNVCHSEL<5:0>位指定。ADCON3L寄存器中的CNVRTCH位用于触发转换。该位由硬件自动清零，从而允许应用程序根据需要触发另一次转换。

4.9.6 触发暂停

ADC模块的ADCON3L寄存器中有一个SUSPEND位，专用于暂停对全部ADC内核的所有触发。当SUSPEND位置1时，将禁止对全部SAR ADC内核的所有未来触发。但是，该位不会取消“按优先级延迟”的触发以及正在进行的触发。触发暂停后，软件应轮询ADCON3L寄存器中的SUSPRDY位以确保所有待处理触发得到处理。当SUSPEND和SUSPRDY位置1时，将禁止对全部ADC内核的触发。如果ADCON3L寄存器中的SUSPCIE位置1，则SUSPRDY位事件可产生共用ADC中断。

4.10 转换结果

模块包含有名为ADCBUFx（其中，“x”是模拟通道的编号）的数据输出寄存器，供各个模拟输入存储A/D结果。这些缓冲区是只读的。当数据写入数据寄存器时，ADSTATL或ADSTATH寄存器中相应的ANxRDY位将置1。当ANxRDY位置1时，将产生中断请求。当读取特定的数据寄存器时，相应的ANxRDY位立即清零。如果缓冲单元尚未被软件读取而ADC需要用新的转换结果覆盖该单元，则前一个数据将丢失。

4.10.1 ADC结果的格式

将A/D转换器的转换结果数据写入ADC结果寄存器前会经过数据格式器。最终数据会被格式化为16位字。

ADC结果寄存器中的数据可以所支持的四种数据格式之一进行读取。用户可从无符号整数、有符号整数、无符号小数或有符号小数这四种格式中选择。整数数据采用右对齐方式，小数数据采用左对齐方式。

- 可通过ADCON1H寄存器中的FORM位为所有ADC输入全局指定整数/小数数据格式的选择。
- 可通过ADMODnL或ADMODnH寄存器中的DIFFx和SIGNx位单独为每个输入通道指定有符号/无符号数据格式的选择，如第4.6.2节“单端和伪差分输入选项及输出结果格式”所述。
- 输出数据格式取决于ADC内核的分辨率，分辨率通过ADCOREnH寄存器中的RES<1:0>位（对于专用内核）以及ADCON1H寄存器中的SHRRES<1:0>位（对于共用内核）指定。

图4-7、图4-8、图4-9和图4-10说明了结果的格式。

图4-7: ADC输出数据格式（12位分辨率）

模数转换结果

d11	d10	d09	d08	d07	d06	d05	d04	d03	d02	d01	d00
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

无符号整数 (**SIGN_x = 0**, **FORM = 0**)

0	0	0	0	d11	d10	d09	d08	d07	d06	d05	d04	d03	d02	d01	d00
---	---	---	---	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

有符号整数 (**SIGN_x = 1**, **FORM = 0**)

S	S	S	S	S	d10	d09	d08	d07	d06	d05	d04	d03	d02	d01	d00
---	---	---	---	---	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

小数 (**SIGN_x = 0**, **FORM = 1**)

d11	d10	d09	d08	d07	d06	d05	d04	d03	d02	d01	d00	0	0	0	0
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	---	---	---	---

有符号小数 (**SIGN_x = 1**, **FORM = 1**)

S	d10	d09	d08	d07	d06	d05	d04	d03	d02	d01	d00	0	0	0	0
---	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	---	---	---	---

图 4-8: ADC 输出数据格式 (10 位分辨率)

模数转换结果

d09	d08	d07	d06	d05	d04	d03	d02	d01	d00
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

无符号整数 (SIGNx = 0, FORM = 0)

0	0	0	0	d09	d08	d07	d06	d05	d04	d03	d02	d01	d00	1	0
---	---	---	---	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	---	---

有符号整数 (SIGNx = 1, FORM = 0)

S	S	S	S	S	d08	d07	d06	d05	d04	d03	d02	d01	d00	1	0
---	---	---	---	---	-----	-----	-----	-----	-----	-----	-----	-----	-----	---	---

小数 (SIGNx = 0, FORM = 1)

d09	d08	d07	d06	d05	d04	d03	d02	d01	d00	1	0	0	0	0	0
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	---	---	---	---	---	---

有符号小数 (SIGNx = 1, FORM = 1)

S	d08	d07	d06	d05	d04	d03	d02	d01	d00	1	0	0	0	0	0
---	-----	-----	-----	-----	-----	-----	-----	-----	-----	---	---	---	---	---	---

图 4-9: ADC 输出数据格式 (8 位分辨率)

模数转换结果

d07	d06	d05	d04	d03	d02	d01	d00
-----	-----	-----	-----	-----	-----	-----	-----

无符号整数 (**SIGNx = 0, FORM = 0**)

0	0	0	0	d07	d06	d05	d04	d03	d02	d01	d00	1	0	0	0
---	---	---	---	-----	-----	-----	-----	-----	-----	-----	-----	---	---	---	---

有符号整数 (**SIGNx = 1, FORM = 0**)

S	S	S	S	S	d06	d05	d04	d03	d02	d01	d00	1	0	0	0
---	---	---	---	---	-----	-----	-----	-----	-----	-----	-----	---	---	---	---

小数 (**SIGNx = 0, FORM = 1**)

d07	d06	d05	d04	d03	d02	d01	d00	1	0	0	0	0	0	0	0
-----	-----	-----	-----	-----	-----	-----	-----	---	---	---	---	---	---	---	---

有符号小数 (**SIGNx = 1, FORM = 1**)

S	d06	d05	d04	d03	d02	d01	d00	1	0	0	0	0	0	0	0
---	-----	-----	-----	-----	-----	-----	-----	---	---	---	---	---	---	---	---

图 4-10: ADC 输出数据格式 (6 位分辨率)

模数转换结果

d05	d04	d03	d02	d01	d00
-----	-----	-----	-----	-----	-----

无符号整数 (SIGNx = 0, FORM = 0)

0	0	0	0	d05	d04	d03	d02	d01	d00	1	0	0	0	0	0
---	---	---	---	-----	-----	-----	-----	-----	-----	---	---	---	---	---	---

有符号整数 (SIGNx = 1, FORM = 0)

S	S	S	S	S	d04	d03	d02	d01	d00	1	0	0	0	0	0
---	---	---	---	---	-----	-----	-----	-----	-----	---	---	---	---	---	---

小数 (SIGNx = 0, FORM = 1)

d05	d04	d03	d02	d01	d00	1	0	0	0	0	0	0	0	0	0
-----	-----	-----	-----	-----	-----	---	---	---	---	---	---	---	---	---	---

有符号小数 (SIGNx = 1, FORM = 1)

S	d04	d03	d02	d01	d00	1	0	0	0	0	0	0	0	0	0
---	-----	-----	-----	-----	-----	---	---	---	---	---	---	---	---	---	---

4.11 数字比较器

ADC 模块具有多个数字比较器，可用于监视所选模拟输入的转换结果并在转换结果处于/未处于用户指定的限值范围内时产生中断。比较操作在转换完成后自动执行。数字比较器通过将 ADCMPnCON 寄存器中的数字比较器使能位 CMPEN 置 1 来使能。

模数转换结果处于上下限值之间、高于上限值或低于下限值时会产生中断（上、下限值由 ADCMPnLO 和 ADCMPnHI 寄存器指定）。ADCMPnENL/H 寄存器中的 CMPENx 位用于指定受数字比较器监视的模拟输入。写入 ADCMPnLO 和 ADCMPnHI 的限值必须与由 ADCON1H 寄存器中的 FORM 位以及 ADMODnL/H 寄存器中的 DIFFx 和 SIGNx 位选择的数据格式匹配。

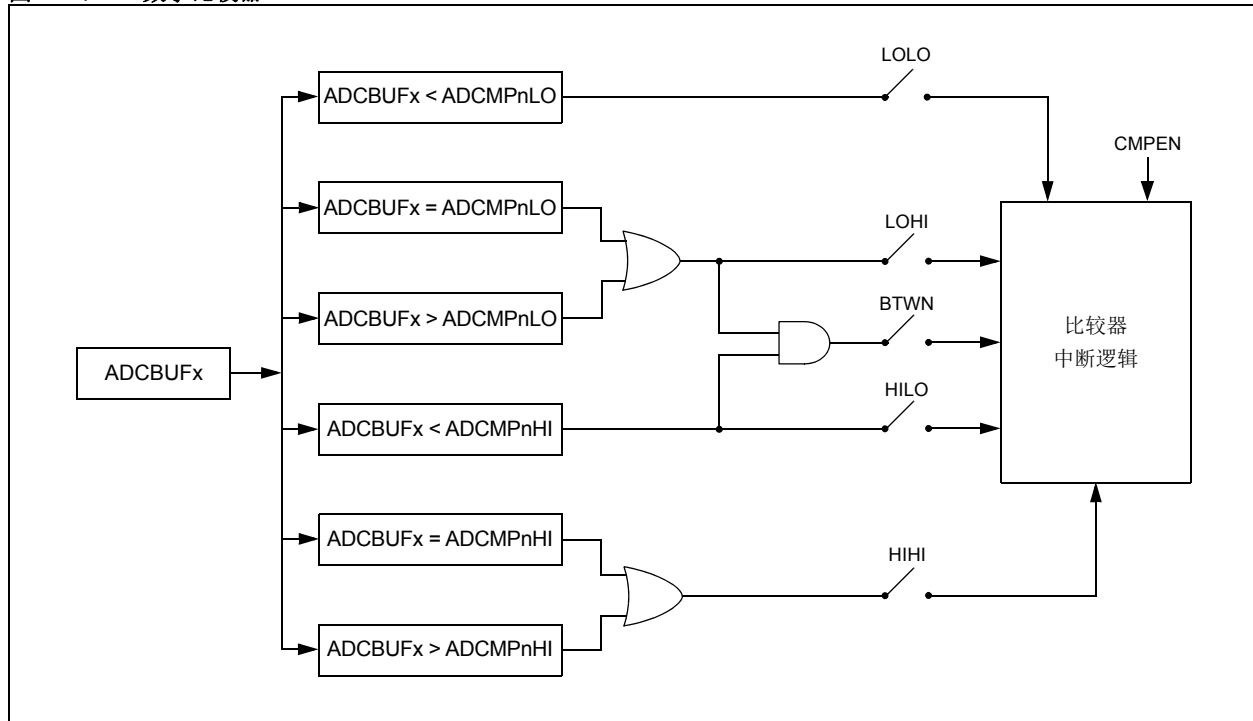
ADCMPnCON 寄存器指定将产生中断的比较条件：

- 如果 BTWN = 1，则 $\text{ADCMPnLO} \leq \text{ADCBUFx} < \text{ADCMPnHI}$ 时产生事件
- 如果 HIHI = 1，则 $\text{ADCBUFx} \geq \text{ADCMPnHI}$ 时产生事件
- 如果 HILO = 1，则 $\text{ADCBUFx} < \text{ADCMPnHI}$ 时产生事件
- 如果 LOHI = 1，则 $\text{ADCBUFx} \geq \text{ADCMPnLO}$ 时产生事件
- 如果 LOLO = 1，则 $\text{ADCBUFx} < \text{ADCMPnLO}$ 时产生事件

比较器事件的产生如图 4-11 所示。当 ADC 产生转换结果时，数字比较器会将所选通道的 ADC 结果与上、下限值（取决于 ADCMPnCON 寄存器中选择的比较条件）进行比较。如果发生比较器事件，则 ADCMPnCON 寄存器中的比较器事件状态位 STAT 置 1 且输入通道编号位 CHNL<4:0> 自动更新，以便应用程序获悉哪个模拟输入产生了事件。读取 CHNL<4:0> 位将清零 STAT 标志。如果 ADCMPnCON 寄存器中的 IE 位置 1，则比较器会产生独立中断和共用中断。

注： 应用程序必须对 ADCMPnLO 和 ADCMPnHI 寄存器中包含的值进行格式化，以便与转换后的数据格式（有符号或无符号，小数或整数）匹配。

图 4-11： 数字比较器



4.12 过采样数字滤波器

ADC 模块可支持多个过采样数字滤波器。滤波器由累加器和抽取器（降低采样频率采样器）组成，二者一起构成低通滤波器。以高于所需采样率的速率对模拟输入进行采样并通过滤波器处理数据，可提高 ADC 模块的有效分辨率，但是会降低转换吞吐量。例如，使用 4x 过采样可使分辨率增加 1 位，使用 16x 过采样可使分辨率增加 2 位，使用 64x 过采样可使分辨率增加 3 位，使用 256x 过采样可使分辨率增加 4 位。

要使用过采样数字滤波器执行转换，请按照以下步骤操作：

1. 使用 ADFLnCON 寄存器中的 OVRSAM<2:0> 位选择过采样率。
2. 为后续转换设置采样时间：
 - 对于专用 ADC 内核输入，通过将 ADCON4L 寄存器中相应的 SAMCxEN 位置 1 来使能触发和转换开始之间的延时，并使用 ADCOREnL 寄存器中的 SAMC<9:0> 位选择重复转换的采样时间
 - 对于共用 ADC 内核输入，使用 ADCON2H 寄存器中的 SHRSAMC<9:0> 位选择采样时间
3. 通过配置 ADFLnCON 寄存器中的 FLCHSEL<4:0> 位来选择将进行过采样的特定模拟输入。
4. 使用 ADFLnCON 寄存器中的 MODE<1:0> 位选择平均模式或过采样模式。
5. 通过将 FLEN 位置 1 使能过采样滤波器。

过采样数字滤波器经过配置后，会等待输入通道触发信号启动过采样过程。该触发信号会导致累加器清零并启动第一次转换。从输入通道获得该初始触发信号后，后续的所有触发信号都将由滤波器本身自动生成。处理完每个转换请求后，会根据 SAMC<9:0> 或 SHRSAMC<9:0> 位的值分别为专用 ADC 内核或共用 ADC 内核启动采样。该过程会一直持续到转换完所需次数（4、8、16、32、64、128 或 256）的采样为止。已转换的采样完成求和后，输出会传送至 ADFLnDAT 寄存器且 ADFLnCON 寄存器中的 RDY 位会置 1。读取 ADFLnDAT 寄存器会清零 RDY 标志。如果 ADFLnCON 寄存器中的 IE 位置 1，则滤波器会产生独立中断和共用中断。滤波器不支持小数数据格式；如果使用滤波器，ADCON1H 寄存器中的 FORM 位必须为 0。

12 位高速多 SAR A/D 转换器 (ADC)

图4-12给出了对专用ADC内核的一个输入进行4x过采样的图示。触发之前，ADC内核会跟踪输入信号。过采样过程由触发信号启动。转换过程将在经过由SAMC<9:0>位定义的采样延时后启动。之后，将发生一个新的采样/转换序列。每个采样经过转换后会加到累加器中。该序列会重复执行，直到由OVSAM<2:0>位域指定的采样数完成累加为止。最后一个采样完成转换后，其值会被加到累加器中。结果采用MODE<1:0>位定义的格式，并存储在ADFLnDAT寄存器中。读取ADFLnDAT寄存器会清零RDY标志。

图4-12: 专用ADC内核模拟输入的4x过采样

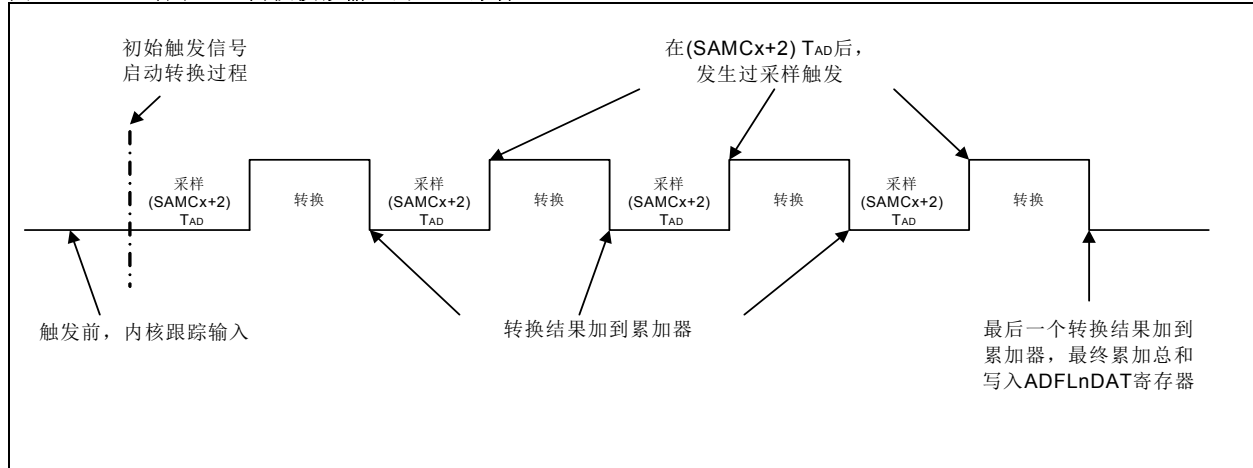
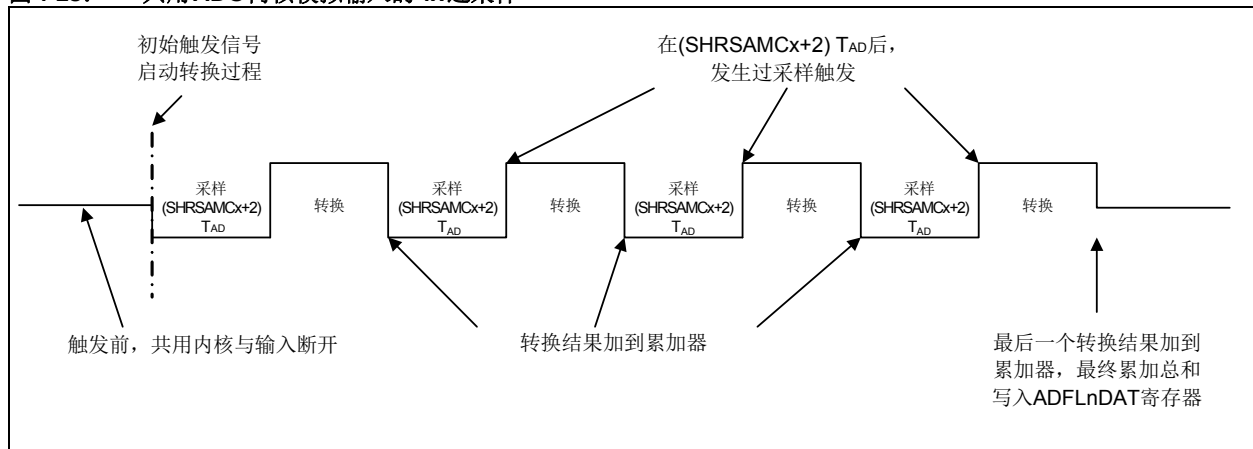


图4-13给出了对共用ADC内核的一个输入进行4x过采样的图示。输入通道触发信号启动采样，采样时长由SHRSAMC<9:0>位定义。

图4-13: 共用ADC内核模拟输入的4x过采样



4.13 中断

ADC 模块可为多种源产生独立中断，并且可为所有 ADC 事件调用共用中断。该模块还具有提前中断功能，用于补偿中断服务延时。当产生某个已允许中断后，CPU 将跳转到为该中断分配的向量。之后，CPU 会开始执行该向量地址处的代码。该向量地址处的应用程序应执行所需操作，例如处理数据结果和清零中断标志，然后退出。中断控制器标志在 ADC 模块中相应的事件标志清零或相应的中断禁止后才能清零。

4.13.1 独立中断

ADC 产生的许多事件都有自己惟一的的中断向量。这样可以使每个中断服务程序（Interrupt Service Routine, ISR）集中地高效处理特定事件，从而显著优化对多个 ADC 事件的处理。

独立中断由以下事件产生：

- 独立输入数据就绪事件：完成对模拟输入源（ANx）的转换后，ADSTATL/H 寄存器中与该输入相关的 ANxRDY 位将置 1。每个 ANxRDY 位均与其在器件级的 ADCANxIF 中断标志关联。要清零 ADC 中断控制器标志 ADCANxIF，必须先通过读取 ADCBUFx 寄存器将 ANxRDY 位清零。要允许独立输入通道中断，必须将 ADIEL 或 ADIEH 寄存器中相应的 IEx 位置 1。
- 数字比较器事件：当满足已使能数字比较器的转换比较条件时，ADCMPnCON 寄存器中的 STAT 位将置 1。每个数字比较器都能够在相应的 STAT 位置 1 时产生自己的器件级中断（由 DCMPxIF 标志控制）。要清零 DCMPxIF 中断控制器标志，必须先通过读取 ADCMPnCON 寄存器中的 CHNL<4:0> 位将 STAT 位清零。要允许独立比较器中断，必须将 ADCMPnCON 寄存器中相应的 IE 位置 1。
- 过采样滤波器数据就绪事件：当过采样滤波器完成累加/抽取过程并存储结果后，ADFLnCON 寄存器中的 RDY 位将置 1。每个过采样滤波器都能够在相应的 RDY 位置 1 时产生自己的器件级中断（由 ADFLTRxIF 标志控制）。要清零 ADFLTRxIF 中断控制器标志，必须先通过读取 ADFLnDAT 寄存器将 RDY 位清零。要允许独立滤波器中断，必须将 ADFLnCON 寄存器中相应的 IE 位置 1。

与其他中断一样，必须将相应中断控制器允许位（ADCANxIE、DCMPxIE 或 ADFLTRxIE）置 1 才能使应用程序在对应的中断标志置 1 时跳转到 ISR 向量。

4.13.2 共用中断

所有ADC事件使用一个共用中断。ADC模块中允许的所有中断事件均会将中断控制器中的ADCIF标志置1。该共用ADC中断标志(ADCIF)将保持置1状态,直到ADC模块中使能的所有状态标志清零或ADC模块中相应的中断禁止后才能清零。发生以下事件时将产生共用中断:

- ADIEL或ADIEH寄存器中相应的IE_x位置1时的每个独立输入数据就绪中断事件。
- ADCMPnCON寄存器中相应的IE位置1时的每个数字比较器中断事件。
- ADFLnCON寄存器中相应的IE位置1时的每个过采样滤波器中断事件。
- ADCON5H寄存器中相应的Cx_CIE或SHRCIE位置1时的每个ADC内核电源就绪事件。当使用的ADC内核处于工作状态时,电源就绪事件标志(ADCON5L寄存器中的Cx_RRDY和SHRRDY)将保持置1。因此,检测到事件后,必须禁止相应的中断(Cx_CIE = 0且SHRCIE = 0)以清零ADCIF标志。
- 暂停ADC触发信号时ADCON3L寄存器中的SUSPCIE位置1。在通过清零ADCON3L寄存器中的SUSPEND位恢复ADC触发信号之前,所有ADC内核暂停标志(ADCON3L寄存器中的SUSPRDY)均将保持置1。因此,检测到事件后,必须禁止相应的中断(SUSPCIE = 0)以清零ADCIF标志。
- 当ADCON2L寄存器中的REFCIE和REFERCIE位分别为参考电压就绪事件和参考电压故障事件置1时。在通过ADCON1L寄存器中的ADON位禁止ADC之前,事件标志(ADCON2H寄存器中的REFRDY和REFERR)会保持置1。因此,检测到事件后,必须禁止相应的中断(REFCIE = 0且REFERCIE = 0)以清零ADCIF标志。

与其他中断一样,必须将相应的ADC中断允许位ADCIE置1才能使应用程序跳转到公共ISR向量。

4.13.3 提前中断

提前中断可通过将ADC转换的完成与同中断关联的处理器开销重叠来提高系统吞吐量。要使用该功能，应将ADCON2L寄存器中的EIEN位置1。如果该位置1，则将在转换完成之前执行所有内核（输入通道）的共用中断和独立中断。当产生输入通道提前中断时，ADEISTATL或ADEISTATH寄存器中相应的EISTATx位将置1。读取相应的ADCBUFx寄存器时，EISTATx标志将清零。可使用ADEIEL或ADEIEH寄存器中的EIEx位单独为每个输入通道允许中断。

尽管输入仍处于转换过程中，但应用软件可使用“抢先启动”（head start）开始进入ISR。

ADCOREnH寄存器中的EISEL<2:0>位（用于专用ADC内核）和ADCON2L寄存器中的SHREISEL<2:0>位（用于共用ADC内核输入）中存储的值决定在转换完成之前提前几个TADCORE时钟周期执行ADC内核提前中断。

提前中断可缩短从触发模拟输入到应用软件可以使用数据之间的延时。收到转换请求时，可使用提前中断选项立即（零延时）处理相应的模拟输入中断。

- 注 1:** 所有内核（输入通道）的提前中断同时被允许（EIEN位 = 1）。无法单独为每个通道选择提前中断功能，但可为每个SAR ADC内核单独设置提前中断延时（时序）（EISEL<2:0>位用于专用内核，SHREISEL<2:0>位用于共用内核）。
- 2:** 允许提前中断后，ADIEL和ADIEH寄存器中的设置不起任何作用。要允许中断，应转为使用ADEIEL和ADEIEH寄存器。
- 3:** 对于6位ADC内核分辨率（RES<1:0>或SHRRES<1:0> = 00），EISEL<2:0>或SHREISEL<2:0>设置从100（5个TADCORE）到111（8个TADCORE）无效，不应使用。对于8位ADC内核分辨率（RES<1:0>或SHRRES<1:0> = 01），EISEL<2:0>或SHREISEL<2:0>设置110（7个TADCORE）和111（8个TADCORE）无效，不应使用。

4.14 电容分压器 (CVD)

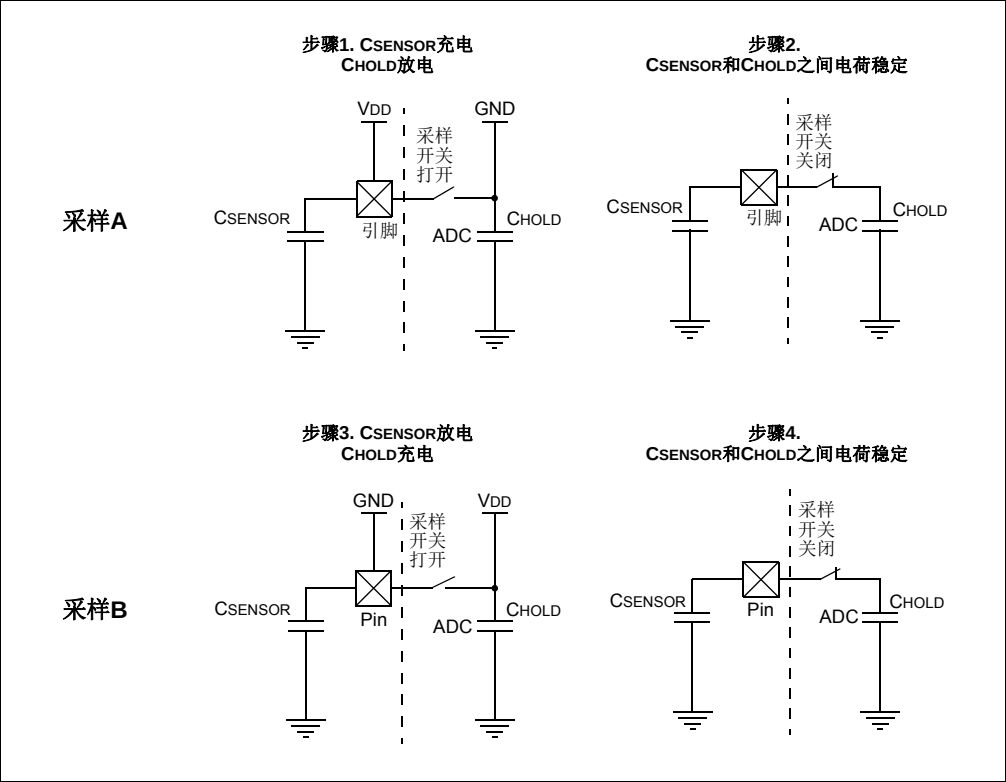
ADC 的共用内核包含支持电容分压器 (CVD) 算法的硬件。并非所有器件都具有此功能, 请参见具体器件的数据手册, 了解器件是否实现了 CVD 功能。CVD 允许测量连接到 ADC 输入的电容值, 可用于在触摸传感器应用中检测触摸事件。

要将模拟输入添加到 CVD 扫描列表, 必须将 ADCSSL 或 ADCSSH 寄存器中的相应位置 1。必须仅在使能和校准 ADC 后, 通过 ADCON1L 寄存器中的 CVDEN 位使能 CVD 功能。当 CVDEN 位置 1 时, CVD 硬件自动对 ADCSSL 和 ADCSSH 寄存器中所选的所有输入运行 CVD 序列。

CVD 测量算法包含两个阶段, 采样 A 和采样 B:

- **采样 A:** 电容传感器 CSENSOR 连接到 I/O 电源 (VDD, 充电)。内部 ADC 采样保持电容 CHOLD 连接到地 (GND, 放电), 如图 4-14 中的步骤 1 所示。接着, 两个电容在一半的采样时间内并联连接, 采样时间由 ADCCON2H 寄存器中的 SHRSAMC<9:0> 位设置, 如图 4-14 中的步骤 2 所示。采样后, 两个电容之间的电荷稳定下来。产生的电压与 CSENSOR 和 CSENSOR + CHOLD 容值的比值成正比。然后, ADC 共用内核对此电压进行转换, ADSTATL 或 ADSTATH 寄存器中相应通道的 TADxRDY 标志置 1, 结果存储在 ADCBUFx 寄存器中, 并产生单独的通道中断。
- **采样 B:** 电容传感器 CSENSOR 连接到地 (GND, 放电)。内部 ADC 采样保持电容 CHOLD 连接到 I/O 电源 (VDD, 充电), 如图 4-14 中的步骤 3 所示。接着, 两个电容在一半的采样时间内并联连接, 采样时间由 ADCCON2H 寄存器中的 SHRSAMC<9:0> 位设置, 如图 4-14 中的步骤 4 所示。采样后, 两个电容之间的电荷稳定下来。产生的电压与 CHOLD 和 CSENSOR + CHOLD 容值的比值成正比。然后, ADC 共用内核对此电压进行转换, ADSTATL 或 ADSTATH 寄存器中相应通道的 TADxRDY 标志置 1, 结果存储在 ADCBUFx 寄存器中, 并产生单独的通道中断。

图 4-14: 采样 A 和采样 B 的 CVD 连接图



公式4-2给出了CVD结果与CSENSOR和CHOLD电容值的关系：

公式 4-2: 12 位分辨率的 CVD 结果计算

$$\text{采样A} - \text{采样B} = 4095 \times \frac{VDD}{Vr+ - Vr-} \times \frac{CSENSOR - CHOLD}{CSENSOR + CHOLD}$$

其中：
Vr+ = 正参考电压
Vr- = 负参考电压

为了确保达到最大灵敏度，CSENSOR应该与CHOLD的电容值相近，这可以通过ADCCON2H寄存器中的CVDCAP<2:0>位向CHOLD额外增加电容值来实现。

当使能CVD时（CVDEN位置1），数字比较器0（如果器件中已实现）将自动连接到CVD硬件，可用于产生比较器事件和中断。数字比较器0的操作如第4.11节“数字比较器”中所述，只有一点不同：在CVD模式下，ADCMPOENL和ADCMPOENH寄存器中的通道选择被忽略；数字比较器0只监视用于当前CVD通道的采样A–采样B值。导致事件产生的模拟输入存储在ADCMPOCON寄存器的CHNL<4:0>位中。此外，如果检测到事件，采样A–采样B值将以有符号格式存储在 ADCVDDAT 寄存器中。

5.0 应用示例

要使用 12 位高速多 SAR A/D 转换器，请按照以下步骤操作：

1. 通过将 ANSELx 和 TRISx 寄存器中相应的位置 1 将 I/O 引脚配置为模拟输入。
2. 使用 ADCON3H 寄存器中的 CLKSEL<1:0> 和 CLKDIV<5:0> 位选择共用 ADC 时钟源并配置预分频比。
3. 分别使用 ADCOREnH 寄存器中的 ADCS<6:0> 位和 ADCON2L 寄存器中的 SHRADCS<6:0> 位为每个专用和共用 ADC 内核选择时钟周期。
4. 使用 ADCON3L 寄存器中的 REFSEL<2:0> 位配置 ADC 参考源。
5. 分别使用 ADCOREnH 寄存器中的 RES<1:0> 位和 ADCON1H 寄存器中的 SHRRES<1:0> 位为每个专用和共用 ADC 内核选择结果分辨率。
6. 使用 ADCON1H 寄存器中的 FORM 位将数据输出格式配置为整数或小数。
7. 使用 ADMODnL 或 ADMODnH 寄存器中的 DIFFx 和 SIGNx 位为每个输入通道选择单端或差分输入配置以及输出格式。
8. 如果使用共用 ADC 内核，则配置 ADCON2H 寄存器中的共用 ADC 内核采样时间选择位 SHRSAMC<8:0>。
9. 配置并允许 ADC 中断。
10. 将 ADCON1L 寄存器中的 ADON 位置 1 以使能模块，并设置 ADCON5H 寄存器中的 WARMTIME<3:0> 位以提供至少 10 μ s 的初始化时间。
11. 接通模块电源：
 - a) 将 ADCON5L 寄存器中的 CxPWR 和 SHRPWR 位置 1。
 - b) 轮询 ADCON5L 寄存器中的 CxRDY 或 SHRRDY 位，直到它们置 1。
 - c) 将 ADCON3H 寄存器中的 CxEN 或 SHREN 位置 1。
12. 使用 ADCALnL 和 ADCALnH 寄存器校准所有 ADC 内核：
 - a) 将 CALxEN 和 CSHREN 位置 1。
 - b) 将 CALxDIFF 和 CSHRDIFF 位清零。
 - c) 将 CALxRUN 和 CSHRRUN 位置 1。
 - d) 轮询 CALxRDY 和 CSHRRDY 位，直到它们置 1。
 - e) 将 CALxDIFF 和 CSHRDIFF 位置 1。
 - f) 将 CALxRUN 和 CSHRRUN 位置 1。
 - g) 轮询 CALxRDY 和 CSHRRDY 位，直到它们置 1。
 - h) 将 CALxEN 和 CSHREN 位清零。
13. 在相应的 ADTRIGnL 或 ADTRIGnH 寄存器中为每个模拟输入设置触发源。

以下部分提供了一些使用 ADC 各种功能的典型示例。

5.1 开启并校准ADC内核

例5-1给出了使能并校准2个专用ADC内核和1个共用ADC内核的过程。某些器件可能不需要校准过程，请参见具体器件的数据手册，了解器件是否需要校准以及是否实现了校准寄存器ADCALnL和ADCALnH。

例5-1: ADC开启和校准过程

```
void EnableAndCalibrate()
{
    // Set initialization time to maximum
    ADCON5Hbits.WARMTIME = 15;

    // Turn on ADC module
    ADCON1Lbits.ADON = 1;

    // Turn on analog power for dedicated core 0
    ADCON5Lbits.C0PWR = 1;
    // Wait when the core 0 is ready for operation
    while(ADCON5Lbits.C0RDY == 0);
    // Turn on digital power to enable triggers to the core 0
    ADCON3Hbits.C0EN = 1;

    // Turn on analog power for dedicated core 1
    ADCON5Lbits.C1PWR = 1;
    // Wait when the core 1 is ready for operation
    while(ADCON5Lbits.C1RDY == 0);
    // Turn on digital power to enable triggers to the core 1
    ADCON3Hbits.C1EN = 1;

    // Turn on analog power for shared core
    ADCON5Lbits.SHRPWR = 1;
    // Wait when the shared core is ready for operation
    while(ADCON5Lbits.SHRRDY == 0);
    // Turn on digital power to enable triggers to the shared core
    ADCON3Hbits.SHREN = 1;

    // Enable calibration for the dedicated core 0
    ADCAL0Lbits.CAL0EN = 1;
    // Single-ended input calibration
    ADCAL0Lbits.CAL0DIFF = 0;
    // Start calibration
    ADCAL0Lbits.CAL0RUN = 1;
    // Poll for the calibration end
    while(ADCAL0Lbits.CAL0RDY == 0);
    // Differential input calibration
    ADCAL0Lbits.CAL0DIFF = 1;
    // Start calibration
    ADCAL0Lbits.CAL0RUN = 1;
    // Poll for the calibration end
    while(ADCAL0Lbits.CAL0RDY == 0);
    // End the core 0 calibration
    ADCAL0Lbits.CAL0EN = 0;
```

例 5-1: ADC 开启和校准过程 (续)

```
// Enable calibration for the dedicated core 1
ADCAL0Lbits.CAL1EN = 1;
// Single-ended input calibration
ADCAL0Lbits.CAL1DIFF = 0;
// Start calibration
ADCAL0Lbits.CAL1RUN = 1;
// Poll for the calibration end
while(ADCAL0Lbits.CAL1RDY == 0);
// Differential input calibration
ADCAL0Lbits.CAL1DIFF = 1;
// Start calibration
ADCAL0Lbits.CAL1RUN = 1;
// Poll for the calibration end
while(ADCAL0Lbits.CAL1RDY == 0);
// End the core 1 calibration
ADCAL0Lbits.CAL1EN = 0;

// Enable calibration for the shared core
ADCAL1Hbits.CSHREN = 1;
// Single-ended input calibration
ADCAL1Hbits.CSHRDIFF = 0;
// Start calibration
ADCAL1Hbits.CSHRRUN = 1;
// Poll for the calibration end
while(ADCAL1Hbits.CSHRRDY == 0);
// Differential input calibration
ADCAL1Hbits.CSHRDIFF = 1;
// Start calibration
ADCAL1Hbits.CSHRRUN = 1;
// Poll for the calibration end
while(ADCAL1Hbits.CSHRRDY == 0);
// End the shared core calibration
ADCAL1Hbits.CSHREN = 0;
}
```

5.2 基本转换序列

例 5-2 给出了基本的初始化和转换序列。本示例演示了通过两个专用 ADC 内核同时进行采样和转换的情况。所有输入由一个源 (Timer2) 触发。各输入 ISR 用于存储转换结果。

例 5-2: 通过专用 ADC 内核同时进行采样和转换

```
// These variables will keep the conversion result.
volatile unsigned short dataAN0;
volatile unsigned short dataAN1;

int main()
{
    // ADC INITIALIZATION
    // Configure the I/O pins to be used as analog inputs.
    ANSELAbits.ANSA0 = 1; TRISAbits.TRISA0 = 1; // AN0/RA0 connected the dedicated core 0
    ANSELAbits.ANSA1 = 1; TRISAbits.TRISA1 = 1; // AN1/RA1 connected the dedicated core 1

    // Configure the common ADC clock.
    ADCON3Hbits.CLKSEL = 2; // clock from FRC oscillator
    ADCON3Hbits.CLKDIV = 0; // no clock divider (1:1)
    // Configure the cores' ADC clock.
    ADCORE0Hbits.ADCS = 0; // clock divider (1:2)
    ADCORE1Hbits.ADCS = 0; // clock divider (1:2)

    // Configure the ADC reference sources.
    ADCON3Lbits.REFSEL = 0; // AVdd as voltage reference
    // Configure the integer of fractional output format.
    ADCON1Hbits.FORM = 0; // integer format

    // Select single-ended input configuration and unsigned output format.
    ADMOD0Lbits.SIGN0 = 0; // AN0/RA0
    ADMOD0Lbits.DIFF0 = 0; // AN0/RA0
    ADMOD0Lbits.SIGN1 = 0; // AN1/RA1
    ADMOD0Lbits.DIFF1 = 0; // AN1/RA1

    // Enable and calibrate the module.
    EnableAndCalibrate(); // See Example 5-1

    // Configure and enable ADC interrupts.
    ADIELbits.IE0 = 1; // enable interrupt for AN0
    ADIELbits.IE1 = 1; // enable interrupt for AN1
    _ADCAN0IF = 0; // clear interrupt flag for AN0
    _ADCAN0IE = 1; // enable interrupt for AN0
    _ADCAN1IF = 0; // clear interrupt flag for AN1
    _ADCAN1IE = 1; // enable interrupt for AN1
```

例5-2: 通过专用ADC内核同时进行采样和转换（续）

```
// Set same trigger source for all inputs to sample signals simultaneously.
ADTRIG0Lbits.TRGSR0 = 13;           // timer 2 for AN0
ADTRIG0Lbits.TRGSR1 = 13;           // timer 2 for AN1

// TIMER 2 INITIALIZATION (TIMER IS USED AS A TRIGGER SOURCE FOR ALL CHANNELS).
T2CONbits.TCS = 0;                   // clock from peripheral clock
T2CONbits.TCKPS = 0;                 // 1:1 prescale
PR2 = 0x8000;                       // rollover every 0x8000 clocks
T2CONbits.TON = 1;                   // start timer to generate ADC triggers
while(1);
return 1;
}

// ADC AN0 ISR
void __attribute__((interrupt, no_auto_psv)) _ADCAN0Interrupt(void)
{
    dataAN0 = ADCBUF0;                // read conversion result
    _ADCAN0IF = 0;                    // clear interrupt flag
}

// ADC AN1 ISR
void __attribute__((interrupt, no_auto_psv)) _ADCAN1Interrupt(void)
{
    dataAN1 = ADCBUF1;                // read conversion result
    _ADCAN1IF = 0;                    // clear interrupt flag
}
```

5.3 数字比较器

数字比较器可用于监视所选的模拟输入转换结果。对于数字比较器模块，建议的初始化序列如下：

1. 按照为ADCBUFx寄存器中的输出数据选择的格式在ADCMPnLO和ADCMPnHI寄存器中设置上、下限阈值。
2. 通过将ADCMPnENL/H寄存器中相应的位置1来选择进行比较的模拟输入。
3. 使用ADCMPnCON寄存器中的BTWN、HIHI、HILO、LOHI和LOLO位设置比较器事件规则。
4. 配置并允许ADC数字比较器中断。
5. 将ADCMPnCON寄存器中的CMPEN位置1以使能比较器。

例5-3演示了使用数字比较器为两个超出ADCMPnLO和ADCMPnHI所定义阈值范围的模拟输入检测电压的情况。

例5-3: 使用数字比较器

```
// This variable will contain the analog input number
// which has generated an event for the digital comparator.
volatile unsigned short comparatorChannelNumber;

// These variables will keep the conversion result.
volatile unsigned short dataAN0;
volatile unsigned short dataAN1;

int main()
{
    // ADC INITIALIZATION
    // Configure the I/O pins to be used as analog inputs.
    ANSELAbits.ANSA0 = 1; TRISAbits.TRISA0 = 1; // AN0/RA0 connected the dedicated core 0
    ANSELAbits.ANSA1 = 1; TRISAbits.TRISA1 = 1; // AN1/RA1 connected the dedicated core 1

    // Configure the common ADC clock.
    ADCON3Hbits.CLKSEL = 2; // clock from FRC oscillator
    ADCON3Hbits.CLKDIV = 0; // no clock divider (1:1)
    // Configure the cores' ADC clock.
    ADCORE0Hbits.ADCS = 0; // clock divider (1:2)
    ADCORE1Hbits.ADCS = 0; // clock divider (1:2)

    // Configure the ADC reference sources.
    ADCON3Lbits.REFSEL = 0; // AVdd as voltage reference
    // Configure the integer of fractional output format.
    ADCON1Hbits.FORM = 0; // integer format

    // Select single-ended input configuration and unsigned output format.
    ADMOD0Lbits.SIGN0 = 0; // AN0/RA0
    ADMOD0Lbits.DIFF0 = 0; // AN0/RA0
    ADMOD0Lbits.SIGN1 = 0; // AN1/RA1
    ADMOD0Lbits.DIFF1 = 0; // AN1/RA1

    // Enable and calibrate the module.
    EnableAndCalibrate(); // See Example 5-1
    // Configure and enable ADC interrupts.
    ADIELbits.IE0 = 1; // enable interrupt for AN0
    ADIELbits.IE1 = 1; // enable interrupt for AN1
    _ADCAN0IF = 0; // clear interrupt flag for AN0
    _ADCAN0IE = 1; // enable interrupt for AN0
    _ADCAN1IF = 0; // clear interrupt flag for AN1
    _ADCAN1IE = 1; // enable interrupt for AN1
```

例 5-3: 使用数字比较器 (续)

```
// Set same trigger source for all inputs to sample signals simultaneously.
ADTRIG0Lbits.TRGSRC0 = 13;           // timer 2 for AN0
ADTRIG0Lbits.TRGSRC1 = 13;           // timer 2 for AN1

// TIMER 2 INITIALIZATION (TIMER IS USED AS A TRIGGER SOURCE FOR ALL CHANNELS).
T2CONbits.TCS = 0;                    // clock from peripheral clock
T2CONbits.TCKPS = 0;                  // 1:1 prescale
PR2 = 0x8000;                         // rollover every 0x8000 clocks
T2CONbits.TON = 1;                    // start timer to generate ADC triggers

// DIGITAL COMPARATOR INITIALIZATION.
// Set high and low thresholds.
ADCMP0LO = 1024;
ADCMP0HI = 4096-1024;
// Select analog inputs for the comparison.
ADCMP0ENLbits.CMPEN0 = 1;            // AN0
ADCMP0ENLbits.CMPEN1 = 1;            // AN1
// Set the comparator event rule.
ADCMP0CONbits.LOLO = 1;               // Generate interrupt if input level is outside window
ADCMP0CONbits.HIHI = 1;               // specified by ADCMP0LO and ADCMP0HI.
// Enable the ADC Digital Comparator interrupt.
ADCMP0CONbits.IE = 1;
_ADCMP0IF = 0;
_ADCMP0IE = 1;
// Enable the comparator.
ADCMP0CONbits.CMPEN = 1;

while(1);
return 1;
}

// ADC COMPARATOR ISR
// If the conversion result for AN0 or AN1 is less than ADCMP0LO or more than ADCMP0HI,
// the interrupt is generated
void __attribute__((interrupt, no_auto_psv)) _ADCMP0Interrupt(void)
{
    comparatorChannelNumber = ADCMP0CONbits.CHNL; // read the channel number
                                                    // that generated interrupt
    _ADCMP0IF = 0;                                // clear interrupt flag
}

// ADC AN0 ISR
void __attribute__((interrupt, no_auto_psv)) _ADCAN0Interrupt(void)
{
    dataAN0 = ADCBUF0;                            // read conversion result
    _ADCAN0IF = 0;                                // clear interrupt flag
}

// ADC AN1 ISR
void __attribute__((interrupt, no_auto_psv)) _ADCAN1Interrupt(void)
{
    dataAN1 = ADCBUF1;                            // read conversion result
    _ADCAN1IF = 0;                                // clear interrupt flag
}
```

5.4 过采样滤波器

过采样滤波器可帮助提高分辨率或滤除噪声。过采样滤波器的常规初始化步骤如下：

1. 使用ADFLnCON寄存器中的FLCHSEL<4:0>位来选择将进行滤波的模拟输入。
2. 使用ADFLnCON寄存器中的MODE<1:0>位设置滤波器模式。
3. 使用ADFLnCON寄存器中的OVSAM<2:0>位设置所需的过采样系数。
4. 使用ADCOREnL寄存器中的SAMC<9:0>位（用于专用ADC内核）和ADCON2H寄存器中的SHRSAMC<9:0>位（用于共用内核）设置正确的采样时间。
5. 配置并允许ADC过采样滤波器中断。
6. 将ADFLnCON寄存器中的FLEN位置1以使能滤波器。

例5-4演示了针对与专用ADC内核相连的模拟输入的滤波过程。

例5-4： 使用过采样滤波器

```
// This variable will contain the output data from the oversampling filter.
volatile unsigned short filterData;

int main()
{
    // ADC INITIALIZATION
    // Configure the I/O pins to be used as analog inputs.
    ANSELAbits.ANSA0 = 1; TRISAbits.TRISA0 = 1; // AN0/RA0 connected the dedicated core 0

    // Configure the common ADC clock.
    ADCON3Hbits.CLKSEL = 2; // clock from FRC oscillator
    ADCON3Hbits.CLKDIV = 0; // no clock divider (1:1)
    // Configure the cores' ADC clock.
    ADCORE0Hbits.ADCS = 0; // clock divider (1:2)

    // Configure the ADC reference sources.
    ADCON3Lbits.REFSEL = 0; // AVdd as voltage reference
    // Configure the integer of fractional output format.
    ADCON1Hbits.FORM = 0; // integer format

    // Select single-ended input configuration and unsigned output format.
    ADMOD0Lbits.SIGN0 = 0; // AN0/RA0
    ADMOD0Lbits.DIFF0 = 0; // AN0/RA1

    // Enable and calibrate the module.
    EnableAndCalibrate(); // See Example 5-1

    // Set software common trigger as AN0 input trigger source.
    ADTRIG0Lbits.TRGSR0 = 1;

    // OVERSAMPLING FILTER INITIALIZATION.
    ADFL0CONbits.FLCHSEL = 0; // Select the AN0 input for the filter.
    ADFL0CONbits.MODE = 3; // Averaging, 12-bit result.
    ADFL0CONbits.OVSAM = 6; // 128X
```


例 5-4: 使用过采样滤波器 (续)

```
// Enable delay between trigger and the conversion start (SAMC bits).
ADCON4Lbits.SAMC0EN = 1;
// Set sampling time (10x Tad)
ADCORE0Lbits.SAMC = 10;

// Enable the filter.
ADFL0CONbits.FLEN = 1;

while(1)
{
    // Generate Software Common Trigger
    ADCON3Lbits.SWCTRG = 1;

    // Wait for the filter result is ready.
    while(ADFL0CONbits.RDY == 0);

    // Read result (it will clear RDY bit)
    filterData = ADFL0DAT;
}

return 1;
}
```

5.5 提前中断

提前中断可通过将ADC转换的完成与同中断关联的处理器开销重叠来提高系统吞吐量。[例5-5](#)中的代码给出了提前中断的用法。在本示例中，专用内核0和共用内核的中断在转换完成之前产生（专用内核提前1个TAD周期，共用内核提前4个TAD周期）。

例5-5: 使用提前中断

```
// These variables will keep the conversion result.
volatile unsigned short dataAN0;           // dedicated core
volatile unsigned short dataAN2;           // shared core

int main()
{
    // ADC INITIALIZATION
    // Configure the I/O pins to be used as analog inputs.
    ANSELAbits.ANSA0 = 1; TRISAbits.TRISA0 = 1; // AN0/RA0 connected the dedicated core 0
    ANSELAbits.ANSA2 = 1; TRISAbits.TRISA2 = 1; // AN2/RA2 connected the shared core
    // Configure the common ADC clock.
    ADCON3Hbits.CLKSEL = 2;                  // clock from FRC oscillator
    ADCON3Hbits.CLKDIV = 0;                  // no clock divider (1:1)
    // Configure the cores' ADC clock.
    ADCORE0Hbits.ADCS = 0;                   // dedicated core clock divider (1:2)
    ADCON2Lbits.SHRADCS = 0;                 // shared core clock divider (1:2)
    // Configure sample time for shared core.
    ADCON2Hbits.SHRSAMC = 10;                // 12 TAD sample time
    // Configure the ADC reference sources.
    ADCON3Lbits.REFSEL = 0;                  // AVdd as voltage reference
    // Configure the integer of fractional output format.
    ADCON1Hbits.FORM = 0;                    // integer format
    // Select single-ended input configuration and unsigned output format.
    ADMOD0Lbits.SIGN0 = 0;                   // AN0/RA0
    ADMOD0Lbits.DIFF0 = 0;                   // AN0/RA0
    ADMOD0Lbits.SIGN2 = 0;                   // AN2/RA2
    // Enable and calibrate the module.
    EnableAndCalibrate();                    // See Example 5-1
    // Configure and enable early ADC interrupts.
    ADCORE0Hbits.EISEL = 0;                  // early interrupt is generated 1 TADCORE clock prior
                                           // to when the data is ready
    ADCORE1Hbits.EISEL = 3;                  // early interrupt is generated 4 TADCORE clocks prior
                                           // to when the data is ready
    ADCON2Lbits.EIEN = 1;                   // enable early interrupts for ALL inputs
    ADEIELbits.EIEN0 = 1;                   // enable interrupt for AN0
    ADEIELbits.EIEN2 = 1;                   // enable interrupt for AN2
    _ADCAN0IF = 0;                           // clear interrupt flag for AN0
    _ADCAN0IE = 1;                           // enable interrupt for AN0
    _ADCAN2IF = 0;                           // clear interrupt flag for AN2
    _ADCAN2IE = 1;                           // enable interrupt for AN2
}
```

例 5-5: 使用提前中断 (续)

```
// Set same trigger source for all inputs to sample signals simultaneously.
ADTRIG0Lbits.TRGSR0 = 13;           // timer 2 for AN0
ADTRIG0Hbits.TRGSR2 = 13;           // timer 2 for AN2
// TIMER 2 INITIALIZATION (TIMER IS USED AS A TRIGGER SOURCE FOR ALL CHANNELS).
T2CONbits.TCS = 0;                   // clock from peripheral clock
T2CONbits.TCKPS = 0;                 // 1:1 prescale
PR2 = 0x8000;                       // rollover every 0x8000 clocks
T2CONbits.TON = 1;                   // start timer to generate ADC triggers

while(1);

return 1;
}

// ADC AN0 ISR (DEDICATED CORE)
void __attribute__((interrupt, no_auto_psv)) _ADCAN0Interrupt(void)
{
    dataAN0 = ADCBUF0;                // read conversion result
    _ADCAN0IF = 0;                    // clear interrupt flag
}

// ADC AN2 ISR (SHARED CORE)
void __attribute__((interrupt, no_auto_psv)) _ADCAN2Interrupt(void)
{
    dataAN2 = ADCBUF2;                // read conversion result
    _ADCAN2IF = 0;                    // clear interrupt flag
}
```

5.6 电容分压器 (CVD)

CVD允许测量连接到ADC输入的电容值。它可用于检测触摸传感器应用中的触摸事件。当触摸传感器时，其电容值将增大。例5-6中的代码展示了CVD功能的使用。在此例中，CVD扫描2个输入，当这两个输入上的电容值发生变化时（电容触摸传感器被按下），数字比较器0可以检测到这一情况。

例 5-6: 使用 CVD

```
// These variables will keep the CVD result.
volatile short dataAN5CVDSampleA;
volatile short dataAN5CVDSampleB;

volatile short dataAN6CVDSampleA;
volatile short dataAN6CVDSampleB;
volatile short dataCVDResult;

// These variable will contain the CVD input number
// for the touched sensor.
volatile short comparatorChannelNumber;

int main()
{
    // ADC INITIALIZATION
    // Configure the I/O pins to be used as analog inputs.
    // Only shared core inputs support CVD.
    ANSELBbits.ANSB10 = 1; TRISBbits.TRISB10 = 1; // AN5/RB10 connected the shared dedicated core 0
    ANSELBbits.ANSB1 = 1; TRISBbits.TRISB1 = 1; // AN6/RB1 connected the dedicated core 1

    // Configure the common ADC clock.
    ADCON3Hbits.CLKSEL = 2; // clock from FRC oscillator (7.3728 MHz)
    ADCON3Hbits.CLKDIV = 0; // no clock divider (1:1)
    // Configure the cores' ADC clock.
    ADCORE0Hbits.ADCS = 0; // clock divider (1:2)
    ADCORE1Hbits.ADCS = 0; // clock divider (1:2)
    ADCON2Lbits.SHRADCS = 0; // clock divider (1:2) for shared core
    // Configure the ADC reference sources.
    ADCON3Lbits.REFSEL = 0; // AVdd as voltage reference
    // Configure the integer of fractional output format.
    ADCON1Hbits.FORM = 0; // integer format

    // Select single-ended input configuration and unsigned output format.
    ADMOD0Lbits.SIGN5 = 0; // AN5/RB10
    ADMOD0Lbits.SIGN6 = 0; // AN6/RB1

    // Add AN5 and AN6 to CVD scan list
    ADCSSLbits.CSS5 = 1; // AN5
    ADCSSLbits.CSS6 = 1; // AN6

    // Disable all triggers for CVD inputs.
    // Conversions for these channels will be triggered by CVD.
    ADTRIG1Lbits.TRGSRC5 = 0; // AN5
    ADTRIG1Hbits.TRGSRC6 = 0; // AN6

    // Select CVD settle time Tsettle = 128*Tad/2 = 128*(2/FRC)/2 = 17uS
    ADCON2Hbits.SHRSAMC = 128-2;

    // Enable and calibrate the module.
    EnableAndCalibrate(); // See Example 5-1

    // Configure and enable ADC interrupts.
    ADIELbits.IE5 = 1; // enable interrupt for AN5
    ADIELbits.IE6 = 1; // enable interrupt for AN6
```

例 5-6: 使用 CVD（续）

```
_ADCAN5IF = 0; // clear interrupt flag for AN5
_ADCAN5IE = 1; // enable interrupt for AN5
_ADCAN6IF = 0; // clear interrupt flag for AN6
_ADCAN6IE = 1; // enable interrupt for AN6

// DIGITAL COMPARATOR INITIALIZATION.
// Set high threshold.
ADCMP0HI = 2048;
// Set the comparator event rule.
ADCMP0CONbits.HIHI = 1; // generate event when ADCVDDAT > ADCMP0HI
// Enable the ADC Digital Comparator interrupt.
ADCMP0CONbits.IE = 1;
_ADCMP0IF = 0;
_ADCMP0IE = 1;
// Enable the comparator.
ADCMP0CONbits.CMPEN = 1;

// Scan CVD inputs.
ADCON1Lbits.CVDEN = 1;

while(1);
return 1;
}

// ADC AN5 ISR
void __attribute__((interrupt, no_auto_psv)) _ADCAN5Interrupt(void)
{
    static short sampleA = 0;

    sampleA ^= 1; // toggle SampleA flag
    if(sampleA){
        dataAN5CVDSampleA = ADCBUF5; // read CVD SampleA result for AN5
    }else{
        dataAN5CVDSampleB = ADCBUF5; // read CVD SampleB result for AN5
    }

    _ADCAN5IF = 0; // clear interrupt flag
}

// ADC AN6 ISR
void __attribute__((interrupt, no_auto_psv)) _ADCAN6Interrupt(void)
{
    static short sampleA = 0;

    sampleA ^= 1; // toggle SampleA flag
    if(sampleA){
        dataAN6CVDSampleA = ADCBUF6; // read CVD SampleA result for AN6
    }else{
        dataAN6CVDSampleB = ADCBUF6; // read CVD SampleB result for AN6
    }

    _ADCAN6IF = 0; // clear interrupt flag
}

// ADC COMPARATOR ISR
// If the conversion result for AN5 or AN6 is greater than ADCMP0HI,
// the interrupt is generated
void __attribute__((interrupt, no_auto_psv)) _ADCMP0Interrupt(void)
{
    comparatorChannelNumber = ADCMP0CONbits.CHNL; // read the channel number
    // that generated interrupt
    dataCVDResult = ADCVDDAT; // read SampleA - SampleB result
    _ADCMP0IF = 0; // clear interrupt flag
}
```

6.0 节能模式期间的操作

节能模式（休眠和空闲）可最大限度地减少CPU、总线和其他外设的数字活动，这对于降低转换噪声非常有用。

6.1 休眠模式

当器件进入休眠模式时，系统振荡器（Fosc）及依赖它工作的所有组件均被暂停，选择Fosc作为ADC时钟源时包括ADC。如果在使用Fosc作为时钟源的转换过程中调用休眠模式，转换将中止。在从休眠模式退出时，转换器不会继续进行部分完成的转换。器件进入或退出休眠模式不会影响ADC寄存器的内容。

如果ADC时钟源在休眠模式期间处于激活状态，则ADC模块可继续工作。FRC振荡器是一种休眠模式下操作的逻辑选择。休眠模式下的ADC操作可降低转换期间单片机其余部分产生的数字开关噪声。

如果已允许任何ADC中断，则器件将在发生该ADC中断时从休眠模式唤醒。如果ADC中断的优先级高于当前的CPU优先级，程序执行将跳转到ADC ISR。否则，程序将从使器件处于休眠模式的PWRSAV指令后的指令继续执行。

对于休眠模式期间的操作，应用程序必须使用相应的转换触发源来确保将在休眠模式下进行A/D转换。例如，可以在器件处于休眠模式时使用外部触发引脚选项（TRGSRcn<4:0> = 11111）执行采样和转换。

6.2 空闲模式期间的操作

对于ADC，ADCON1L寄存器中的空闲模式停止位ADSIDL指定ADC模块在空闲模式下停止工作还是继续工作。如果ADSIDL = 0，当器件进入空闲模式时，ADC模块将继续正常工作。如果已允许任何ADC中断，则器件将在发生该ADC中断时从空闲模式唤醒。如果ADC中断的优先级高于当前的CPU优先级，程序执行将跳转到ADC ISR。否则，程序将从使器件处于空闲模式的PWRSAV指令后的指令继续执行。

如果ADSIDL = 1，在空闲模式下，ADC模块将停止工作。如果器件在转换过程中进入空闲模式，转换将中止。在从空闲模式退出时，转换器不会继续进行部分完成的转换。

7.0 复位的影响

发生任何复位事件后，所有的ADC控制和状态寄存器将复位为默认值，控制位处于非活动状态。这将禁止ADC模块并将模拟输入引脚设置为模拟输入模式。任何正在进行的转换均将终止，转换结果不会写入结果缓冲区。器件复位期间，ADCBUFx寄存器的值初始化为0000h。

8.0 寄存器映射

表 8-1: 12 位高速多 SAR A/D 转换器寄存器映射⁽¹⁾

寄存器名称	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	所有复位时的值
ADCON1L	ADON	—	ADSIDL	—	CVDEN	—	—	—	NRE	—	—	—	—	—	—	—	0000
ADCON1H	r	r	r	r	r	r	r	r	FORM	SHRRES1	SHRRES0	r	r	r	r	r	0060
ADCON2L	REFCIE	REFERCIE	r	EIEN	r	SHREISEL2	SHREISEL1	SHREISEL0	—	SHRADCS6	SHRADCS5	SHRADCS4	SHRADCS3	SHRADCS2	SHRADCS1	SHRADCS0	0000
ADCON2H	REFRDY	REFERR	r	CVDCAP2	CVDCAP1	CVDCAP0	SHRSAMC9	SHRSAMC8	SHRSAMC7	SHRSAMC6	SHRSAMC5	SHRSAMC4	SHRSAMC3	SHRSAMC2	SHRSAMC1	SHRSAMC0	0000
ADCON3L	RFSEL2	RFSEL1	RFSEL0	SUSPEND	SUSPCIE	SUSPRDY	SHRSAMP	CNVRTCH	SWLCTRG	SWCTRG	CNVCHSEL5	CNVCHSEL4	CNVCHSEL3	CNVCHSEL2	CNVCHSEL1	CNVCHSEL0	0000
ADCON3H	CLKSEL1	CLKSEL0	CLKDIV5	CLKDIV4	CLKDIV3	CLKDIV2	CLKDIV1	CLKDIV0	SHREN	C6EN	C5EN	C4EN	C3EN	C2EN	C1EN	C0EN	0000
ADCON4L	—	r	r	r	r	r	r	r	—	SAMC6EN	SAMC5EN	SAMC4EN	SAMC3EN	SAMC2EN	SAMC1EN	SAMC0EN	0000
ADCON4H	—	—	C6CHS1	C6CHS0	C5CHS1	C5CHS0	C4CHS1	C4CHS0	C3CHS1	C3CHS0	C2CHS1	C2CHS0	C1CHS1	C1CHS0	C0CHS1	C0CHS0	0000
ADMOD0L	DIFF7	SIGN7	DIFF6	SIGN6	DIFF5	SIGN5	DIFF4	SIGN4	DIFF3	SIGN3	DIFF2	SIGN2	DIFF1	SIGN1	DIFF0	SIGN0	0000
ADMOD0H	DIFF15	SIGN15	DIFF14	SIGN14	DIFF13	SIGN13	DIFF12	SIGN12	DIFF11	SIGN11	DIFF10	SIGN10	DIFF9	SIGN9	DIFF8	SIGN8	0000
ADMOD1L	DIFF23	SIGN23	DIFF22	SIGN22	DIFF21	SIGN21	DIFF20	SIGN20	DIFF19	SIGN19	DIFF18	SIGN18	DIFF17	SIGN17	DIFF16	SIGN16	0000
ADMOD1H	DIFF31	SIGN31	DIFF30	SIGN30	DIFF29	SIGN29	DIFF28	SIGN28	DIFF27	SIGN27	DIFF26	SIGN26	DIFF25	SIGN25	DIFF24	SIGN24	0000
ADIEL	IE<15:0>																0000
ADIEH	IE<31:16>																0000
ADCSSL	CSS<15:0>																0000
ADCSSH	CSS<31:16>																0000
ADSTATL	AN15RDY	AN14RDY	AN13RDY	AN12RDY	AN11RDY	AN10RDY	AN9RDY	AN8RDY	AN7RDY	AN6RDY	AN5RDY	AN4RDY	AN3RDY	AN2RDY	AN1RDY	AN0RDY	0000
ADSTATH	AN31RDY	AN30RDY	AN29RDY	AN28RDY	AN27RDY	AN26RDY	AN25RDY	AN24RDY	AN23RDY	AN22RDY	AN21RDY	AN20RDY	AN19RDY	AN18RDY	AN17RDY	AN16RDY	0000
ADCMP0ENL	CMPEN<15:0>																0000
ADCMP0ENH	CMPEN<31:16>																0000
ADCMP0LO	ADC CMPLO 寄存器																0000
ADCMP0HI	ADC CMPHI 寄存器																0000
ADCMP1ENL	CMPEN<15:0>																0000
ADCMP1ENH	CMPEN<31:16>																0000
ADCMP1LO	ADC CMPLO 寄存器																0000
ADCMP1HI	ADC CMPHI 寄存器																0000
ADCMP2ENL	CMPEN<15:0>																0000
ADCMP2ENH	CMPEN<31:16>																0000
ADCMP2LO	ADC CMPLO 寄存器																0000
ADCMP2HI	ADC CMPHI 寄存器																0000
ADCMP3ENL	CMPEN<15:0>																0000
ADCMP3ENH	CMPEN<31:16>																0000
ADCMP3LO	ADC CMPLO 寄存器																0000
ADCMP3HI	ADC CMPHI 寄存器																0000

图注: — = 未实现, 读为 0; r = 保留, 必须写为 0。复位值以十六进制显示。

注 1: 并非所有器件上都实现了所有寄存器。有关器件特定的寄存器映射和位实现的信息, 请参见器件的数据手册。

表 8-1: 12 位高速多 SAR A/D 转换器寄存器映射⁽¹⁾ (续)

寄存器名称	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	所有复位时的值
ADCMP4ENL	CMPEN<15:0>																0000
ADCMP4ENH	CMPEN<31:16>																0000
ADCMP4LO	ADC CMPLO 寄存器																0000
ADCMP4HI	ADC CMPHI 寄存器																0000
ADCMP5ENL	CMPEN<15:0>																0000
ADCMP5ENH	CMPEN<31:16>																0000
ADCMP5LO	ADC CMPLO 寄存器																0000
ADCMP5HI	ADC CMPHI 寄存器																0000
ADFL0DAT	ADC FLDATA 寄存器																0000
ADFL0CON	FLEN	MODE1	MODE0	OVRSAM2	OVRSAM1	OVRSAM0	IE	RDY	—	—	—	FLCHSEL4	FLCHSEL3	FLCHSEL2	FLCHSEL1	FLCHSEL0	0000
ADFL1DAT	ADC FLDATA 寄存器																0000
ADFL1CON	FLEN	MODE1	MODE0	OVRSAM2	OVRSAM1	OVRSAM0	IE	RDY	—	—	—	FLCHSEL4	FLCHSEL3	FLCHSEL2	FLCHSEL1	FLCHSEL0	0000
ADFL2DAT	ADC FLDATA 寄存器																0000
ADFL2CON	FLEN	MODE1	MODE0	OVRSAM2	OVRSAM1	OVRSAM0	IE	RDY	—	—	—	FLCHSEL4	FLCHSEL3	FLCHSEL2	FLCHSEL1	FLCHSEL0	0000
ADFL3DAT	ADC FLDATA 寄存器																0000
ADFL3CON	FLEN	MODE1	MODE0	OVRSAM2	OVRSAM1	OVRSAM0	IE	RDY	—	—	—	FLCHSEL4	FLCHSEL3	FLCHSEL2	FLCHSEL1	FLCHSEL0	0000
ADFL4DAT	ADC FLDATA 寄存器																0000
ADFL4CON	FLEN	MODE1	MODE0	OVRSAM2	OVRSAM1	OVRSAM0	IE	RDY	—	—	—	FLCHSEL4	FLCHSEL3	FLCHSEL2	FLCHSEL1	FLCHSEL0	0000
ADFL5DAT	ADC FLDATA 寄存器																0000
ADFL5CON	FLEN	MODE1	MODE0	OVRSAM2	OVRSAM1	OVRSAM0	IE	RDY	—	—	—	FLCHSEL4	FLCHSEL3	FLCHSEL2	FLCHSEL1	FLCHSEL0	0000
ADTRIG0L	—	—	—	TRGSRC14	TRGSRC13	TRGSRC12	TRGSRC11	TRGSRC10	—	—	—	TRGSRC04	TRGSRC03	TRGSRC02	TRGSRC01	TRGSRC00	0000
ADTRIG0H	—	—	—	TRGSRC34	TRGSRC33	TRGSRC32	TRGSRC31	TRGSRC30	—	—	—	TRGSRC24	TRGSRC23	TRGSRC22	TRGSRC21	TRGSRC20	0000
ADTRIG1L	—	—	—	TRGSRC54	TRGSRC53	TRGSRC52	TRGSRC51	TRGSRC50	—	—	—	TRGSRC44	TRGSRC43	TRGSRC42	TRGSRC41	TRGSRC40	0000
ADTRIG1H	—	—	—	TRGSRC74	TRGSRC73	TRGSRC72	TRGSRC71	TRGSRC70	—	—	—	TRGSRC64	TRGSRC63	TRGSRC62	TRGSRC61	TRGSRC60	0000
ADTRIG2L	—	—	—	TRGSRC94	TRGSRC93	TRGSRC92	TRGSRC91	TRGSRC90	—	—	—	TRGSRC84	TRGSRC83	TRGSRC82	TRGSRC81	TRGSRC80	0000
ADTRIG2H	—	—	—	TRGSRC114	TRGSRC113	TRGSRC112	TRGSRC111	TRGSRC110	—	—	—	TRGSRC104	TRGSRC103	TRGSRC102	TRGSRC101	TRGSRC100	0000
ADTRIG3L	—	—	—	TRGSRC134	TRGSRC133	TRGSRC132	TRGSRC131	TRGSRC130	—	—	—	TRGSRC124	TRGSRC123	TRGSRC122	TRGSRC121	TRGSRC120	0000
ADTRIG3H	—	—	—	TRGSRC154	TRGSRC153	TRGSRC152	TRGSRC151	TRGSRC150	—	—	—	TRGSRC144	TRGSRC143	TRGSRC142	TRGSRC141	TRGSRC140	0000
ADTRIG4L	—	—	—	TRGSRC174	TRGSRC173	TRGSRC172	TRGSRC171	TRGSRC170	—	—	—	TRGSRC164	TRGSRC163	TRGSRC162	TRGSRC161	TRGSRC160	0000
ADTRIG4H	—	—	—	TRGSRC194	TRGSRC193	TRGSRC192	TRGSRC191	TRGSRC190	—	—	—	TRGSRC184	TRGSRC183	TRGSRC182	TRGSRC181	TRGSRC180	0000
ADTRIG5L	—	—	—	TRGSRC214	TRGSRC213	TRGSRC212	TRGSRC211	TRGSRC210	—	—	—	TRGSRC204	TRGSRC203	TRGSRC202	TRGSRC201	TRGSRC200	0000
ADTRIG5H	—	—	—	TRGSRC234	TRGSRC233	TRGSRC232	TRGSRC231	TRGSRC230	—	—	—	TRGSRC224	TRGSRC223	TRGSRC222	TRGSRC221	TRGSRC220	0000
ADTRIG6L	—	—	—	TRGSRC254	TRGSRC253	TRGSRC252	TRGSRC251	TRGSRC250	—	—	—	TRGSRC244	TRGSRC243	TRGSRC242	TRGSRC241	TRGSRC240	0000
ADTRIG6H	—	—	—	TRGSRC274	TRGSRC273	TRGSRC272	TRGSRC271	TRGSRC270	—	—	—	TRGSRC264	TRGSRC263	TRGSRC262	TRGSRC261	TRGSRC260	0000

图注: — = 未实现, 读为 0; r = 保留, 必须写为 0。复位值以十六进制显示。

注 1: 并非所有器件上都实现了所有寄存器。有关器件特定的寄存器映射和位实现的信息, 请参见器件的数据手册。

表 8-1: 12位高速多 SAR A/D 转换器寄存器映射⁽¹⁾ (续)

寄存器名称	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	所有复位时的值
ADTRIG7L	—	—	—	TRGSRC294	TRGSRC293	TRGSRC292	TRGSRC291	TRGSRC290	—	—	—	TRGSRC284	TRGSRC283	TRGSRC282	TRGSRC281	TRGSRC280	0000
ADTRIG7H	—	—	—	TRGSRC314	TRGSRC313	TRGSRC312	TRGSRC311	TRGSRC310	—	—	—	TRGSRC304	TRGSRC303	TRGSRC302	TRGSRC301	TRGSRC300	0000
ADCMPOCON	—	—	—	CHNL4	CHNL3	CHNL2	CHNL1	CHNL0	CMPEN	IE	STAT	BTWN	HIHI	HILO	LOHI	LOLO	0000
ADCVDDAT	ADCVDDAT<15:0>																0000
ADCMPICON	—	—	—	CHNL4	CHNL3	CHNL2	CHNL1	CHNL0	CMPEN	IE	STAT	BTWN	HIHI	HILO	LOHI	LOLO	0000
ADCMPI2CON	—	—	—	CHNL4	CHNL3	CHNL2	CHNL1	CHNL0	CMPEN	IE	STAT	BTWN	HIHI	HILO	LOHI	LOLO	0000
ADCMPI3CON	—	—	—	CHNL4	CHNL3	CHNL2	CHNL1	CHNL0	CMPEN	IE	STAT	BTWN	HIHI	HILO	LOHI	LOLO	0000
ADCMPI4CON	—	—	—	CHNL4	CHNL3	CHNL2	CHNL1	CHNL0	CMPEN	IE	STAT	BTWN	HIHI	HILO	LOHI	LOLO	0000
ADCMPI5CON	—	—	—	CHNL4	CHNL3	CHNL2	CHNL1	CHNL0	CMPEN	IE	STAT	BTWN	HIHI	HILO	LOHI	LOLO	0000
ADLVLTRGL	LVLEN<15:0>																0000
ADLVLTRGH	LVLEN<31:16>																0000
ADCORE0L	—	—	—	—	—	—	—	SAMC<9:0>									0000
ADCORE0H	—	—	—	—	—	—	RES1	RES0	—	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0300
ADCORE1L	—	—	—	—	—	—	—	SAMC<9:0>									0000
ADCORE1H	—	—	—	EISEL2	EISEL1	EISEL0	RES1	RES0	—	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0300
ADCORE2L	—	—	—	—	—	—	—	SAMC<9:0>									0000
ADCORE2H	—	—	—	EISEL2	EISEL1	EISEL0	RES1	RES0	—	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0300
ADCORE3L	—	—	—	—	—	—	—	SAMC<9:0>									0000
ADCORE3H	—	—	—	EISEL2	EISEL1	EISEL0	RES1	RES0	—	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0300
ADCORE4L	—	—	—	—	—	—	—	SAMC<9:0>									0000
ADCORE4H	—	—	—	EISEL2	EISEL1	EISEL0	RES1	RES0	—	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0300
ADCORE5L	—	—	—	—	—	—	—	SAMC<9:0>									0000
ADCORE5H	—	—	—	EISEL2	EISEL1	EISEL0	RES1	RES0	—	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0300
ADCORE6L	—	—	—	—	—	—	—	SAMC<9:0>									0000
ADCORE6H	—	—	—	EISEL2	EISEL1	EISEL0	RES1	RES0	—	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0	0300
ADEIEL	EIE<15:0>																0000
ADEIEH	EIE<31:16>																0000
ADEISTATL	EISTAT<15:0>																0000
ADEISTATH	EISTAT<31:16>																0000
ADCON5L	SHRRDY	C6RDY	C5RDY	C4RDY	C3RDY	C2RDY	C1RDY	CORDY	SHRPWR	C6PWR	C5PWR	C4PWR	C3PWR	C2PWR	C1PWR	C0PWR	0000
ADCON5H	—	—	—	—	WARMTIME3	WARMTIME2	WARMTIME1	WARMTIME0	SHRCIE	C6CIE	C5CIE	C4CIE	C3CIE	C2CIE	C1CIE	C0CIE	0000
ADCAL0L	CAL1RDY	—	—	—	r	CAL1DIFF	CAL1EN	CAL1RUN	CAL0RDY	—	—	—	r	CAL0DIFF	CAL0EN	CAL0RUN	0000
ADCAL0H	CAL3RDY	—	—	—	r	CAL3DIFF	CAL3EN	CAL3RUN	CAL2RDY	—	—	—	r	CAL2DIFF	CAL2EN	CAL2RUN	0000
ADCAL1L	CAL5RDY	—	—	—	r	CAL5DIFF	CAL5EN	CAL5RUN	CAL4RDY	—	—	—	r	CAL4DIFF	CAL4EN	CAL4RUN	0000
ADCAL1H	CSHRRDY	—	—	—	r	CSHRDIFF	CSHREN	CSHRRUN	CAL6RDY	—	—	—	r	CAL6DIFF	CAL6EN	CAL6RUN	0000

图注： — = 未实现，读为0；r = 保留，必须写为0。复位值以十六进制显示。

注 1：并非所有器件上都实现了所有寄存器。有关器件特定的寄存器映射和位实现的信息，请参见器件的数据手册。

表 8-1: 12 位高速多 SAR A/D 转换器寄存器映射⁽¹⁾ (续)

寄存器名称	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	所有复位时的值
ADCBUF0	DATA0<15:0>																0000
ADCBUF1	DATA1<15:0>																0000
ADCBUF2	DATA2<15:0>																0000
ADCBUF3	DATA3<15:0>																0000
ADCBUF4	DATA4<15:0>																0000
ADCBUF5	DATA5<15:0>																0000
ADCBUF6	DATA6<15:0>																0000
ADCBUF7	DATA7<15:0>																0000
ADCBUF8	DATA8<15:0>																0000
ADCBUF9	DATA9<15:0>																0000
ADCBUF10	DATA10<15:0>																0000
ADCBUF11	DATA11<15:0>																0000
ADCBUF12	DATA12<15:0>																0000
ADCBUF13	DATA13<15:0>																0000
ADCBUF14	DATA14<15:0>																0000
ADCBUF15	DATA15<15:0>																0000
ADCBUF16	DATA16<15:0>																0000
ADCBUF17	DATA17<15:0>																0000
ADCBUF18	DATA18<15:0>																0000
ADCBUF19	DATA19<15:0>																0000
ADCBUF20	DATA20<15:0>																0000
ADCBUF21	DATA21<15:0>																0000
ADCBUF22	DATA22<15:0>																0000
ADCBUF23	DATA23<15:0>																0000
ADCBUF24	DATA24<15:0>																0000
ADCBUF25	DATA25<15:0>																0000
ADCBUF26	DATA26<15:0>																0000
ADCBUF27	DATA27<15:0>																0000
ADCBUF28	DATA28<15:0>																0000
ADCBUF29	DATA29<15:0>																0000
ADCBUF30	DATA30<15:0>																0000
ADCBUF31	DATA31<15:0>																0000

图注: — = 未实现, 读为 0; r = 保留, 必须写为 0。复位值以十六进制显示。
注 1: 并非所有器件上都实现了所有寄存器。有关器件特定的寄存器映射和位实现的信息, 请参见器件的数据手册。

9.0 相关应用笔记

本节列出了与手册本章内容相关的应用笔记。这些应用笔记可能并不是专为 dsPIC33/PIC24 器件系列而编写的，但其概念是相近的，通过适当修改并做出一定限制即可使用。当前与 12 位高速多 SAR A/D 转换器（ADC）模块相关的应用笔记有：

标题

应用笔记编号

目前没有相关的应用笔记。

注： 如需获取更多 dsPIC33/PIC24 系列器件的应用笔记和代码示例，请访问 Microchip 网站（ www.microchip.com ）。

10.0 版本历史

版本 A（2014 年 11 月）

这是本文档的初始版本。

版本 B（2015 年 5 月）

更新了第 4.4 节“采样和转换时序”并增加了公式 4-1。

版本 C（2015 年 9 月）

将出现的所有差分输入更改为伪差分输入。删除了所有对 CALxSKIP 位的引用。

更新了图 1-2、图 4-5、图 4-6 和图 4-10。

更新了寄存器 2-1、寄存器 2-2、寄存器 2-3、寄存器 2-4、寄存器 2-7、寄存器 2-27、寄存器 2-28、寄存器 2-29、寄存器 2-30 和寄存器 2-31。

更新了表 4-1 和表 8-1。

此版本还更正了整篇文档中的大量语法错误。

版本 D（2016 年 1 月）

更新了图 4-12 和图 4-13。

更新了第 4.6.3 节“为专用 ADC 内核选择模拟输入”和第 4.13.3 节“提前中断”，并增加了第 5.5 节“提前中断”。

版本 E（2017 年 1 月）

更新了寄存器 2-5、寄存器 2-7、寄存器 2-27、寄存器 2-31 和寄存器 2-35。

删除了图 4-2 标题中的“异步”。

删除了原来的图 4-3，并在图 4-4 前新增了一段。

在第 4.6.3 节“为专用 ADC 内核选择模拟输入”中的“注”中额外增加了内容。

在第 4.7 节“使能 ADC”中，更新了使能 ADC 模块的步骤。

更新了表 8-1。

版本 F（2017 年 6 月）

更新了第 1.0 节“简介”、第 2.1 节“控制寄存器”、第 2.2 节“数据寄存器”和第 4.4 节“采样和转换时序”。增加了第 4.14 节“电容分压器（CVD）”和第 5.6 节“电容分压器（CVD）”。

更新了寄存器 2-1 和寄存器 2-4。增加了寄存器 2-36 和寄存器 2-37。

增加了图 4-14。

增加了公式 4-2。

增加了例 5-6。

更新了表 8-1。

请注意以下有关 Microchip 器件代码保护功能的要点：

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

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

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

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

Microchip 位于美国亚利桑那州 Chandler 和 Tempe 与位于俄勒冈州 Gresham 的全球总部、设计和晶圆生产厂及位于美国加利福尼亚州和印度的设计中心均通过了 ISO/TS-16949:2009 认证。Microchip 的 PIC® MCU 与 dsPIC® DSC、KEELOQ® 跳码器件、串行 EEPROM、单片机外设、非易失性存储器 and 模拟产品严格遵守公司的质量体系流程。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。

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
== ISO/TS 16949 ==

商标

Microchip 的名称和徽标组合、Microchip 徽标、AnyRate、AVR、AVR 徽标、AVR Freaks、BeaconThings、BitCloud、CryptoMemory、CryptoRF、dsPIC、FlashFlex、flexPWR、Heldo、JukeBlox、KEELOQ、KEELOQ 徽标、Kleer、LANCheck、LINK MD、maXStylus、maXTouch、MediaLB、megaAVR、MOST、MOST 徽标、MPLAB、OptoLyzer、PIC、picoPower、PICSTART、PIC32 徽标、Prochip Designer、QTouch、RightTouch、SAM-BA、SpyNIC、SST、SST 徽标、SuperFlash、tinyAVR、UNI/O 及 XMEGA 均为 Microchip Technology Inc. 在美国和其他国家或地区的注册商标。

ClockWorks、The Embedded Control Solutions Company、EtherSynch、Hyper Speed Control、HyperLight Load、IntelliMOS、mTouch、Precision Edge 和 Quiet-Wire 均为 Microchip Technology Inc. 在美国的注册商标。

Adjacent Key Suppression、AKS、Analog-for-the-Digital Age、Any Capacitor、AnyIn、AnyOut、BodyCom、chipKIT、chipKIT 徽标、CodeGuard、CryptoAuthentication、CryptoCompanion、CryptoController、dsPICDEM、dsPICDEM.net、Dynamic Average Matching、DAM、ECAN、EtherGREEN、In-Circuit Serial Programming、ICSP、Inter-Chip Connectivity、JitterBlocker、KleerNet、KleerNet 徽标、Mindi、MiWi、motorBench、MPASM、MPF、MPLAB Certified 徽标、MPLIB、MPLINK、MultiTRAK、NetDetach、Omniscient Code Generation、PICDEM、PICDEM.net、PICKit、PICtail、PureSilicon、QMatrix、RightTouch 徽标、REAL ICE、Ripple Blocker、SAM-ICE、Serial Quad I/O、SMART-I.S.、SQI、SuperSwitcher、SuperSwitcher II、Total Endurance、TSHARC、USBCheck、VariSense、ViewSpan、WiperLock、Wireless DNA 和 ZENA 均为 Microchip Technology Inc. 在美国和其他国家或地区的商标。

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

Silicon Storage Technology 为 Microchip Technology Inc. 在除美国外的国家或地区的注册商标。

GestIC 为 Microchip Technology Inc. 的子公司 Microchip Technology Germany II GmbH & Co. KG 在除美国外的国家或地区的注册商标。

在此提及的所有其他商标均为各持有公司所有。

© 2016-2018, Microchip Technology Inc. 版权所有。

ISBN: 978-1-5224-2624-0

全球销售及服务中心

DS70005213F_CNDS

70005213F_CN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://www.microchip.com/support>

网址: www.microchip.com

亚特兰大 Atlanta Duluth, GA

Tel: 1-678-957-9614
Fax: 1-678-957-1455

奥斯汀 Austin, TX Tel: 1-512-257-3370

波士顿 Boston
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Novi, MI
Tel: 1-248-848-4000

休斯敦 Houston, TX
Tel: 1-281-894-5983

印第安纳波利斯
Indianapolis
Noblesville, IN
Tel: 1-317-773-8323
Fax: 1-317-773-5453
Tel: 1-317-536-2380

洛杉矶 Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608
Tel: 1-951-273-7800

罗利 Raleigh, NC
Tel: 1-919-844-7510

纽约 New York, NY
Tel: 1-631-435-6000

圣何塞 San Jose, CA
Tel: 1-408-735-9110
Tel: 1-408-436-4270

加拿大多伦多 Toronto
Tel: 1-905-695-1980
Fax: 1-905-695-2078

亚太地区

中国 - 北京
Tel: 86-10-8569-7000

中国 - 成都
Tel: 86-28-8665-5511

中国 - 重庆
Tel: 86-23-8980-9588

中国 - 东莞
Tel: 86-769-8702-9880

中国 - 广州
Tel: 86-20-8755-8029

中国 - 杭州
Tel: 86-571-8792-8115

中国 - 南京
Tel: 86-25-8473-2460

中国 - 青岛
Tel: 86-532-8502-7355

中国 - 上海
Tel: 86-21-3326-8000

中国 - 沈阳
Tel: 86-24-2334-2829

中国 - 深圳
Tel: 86-755-8864-2200

中国 - 苏州
Tel: 86-186-6233-1526

中国 - 武汉
Tel: 86-27-5980-5300

中国 - 西安
Tel: 86-29-8833-7252

中国 - 厦门
Tel: 86-592-238-8138

中国 - 香港特别行政区
Tel: 852-2943-5100

中国 - 珠海
Tel: 86-756-321-0040

台湾地区 - 高雄
Tel: 886-7-213-7830

台湾地区 - 台北
Tel: 886-2-2508-8600

台湾地区 - 新竹
Tel: 886-3-577-8366

亚太地区

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733

印度 India - Bangalore
Tel: 91-80-3090-4444

印度 India - New Delhi
Tel: 91-11-4160-8631

印度 India - Pune
Tel: 91-20-4121-0141

日本 Japan - Osaka
Tel: 81-6-6152-7160

日本 Japan - Tokyo
Tel: 81-3-6880-3770

韩国 Korea - Daegu
Tel: 82-53-744-4301

韩国 Korea - Seoul
Tel: 82-2-554-7200

马来西亚
Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

马来西亚 Malaysia - Penang
Tel: 60-4-227-8870

菲律宾 Philippines - Manila
Tel: 63-2-634-9065

新加坡 Singapore
Tel: 65-6334-8870

泰国 Thailand - Bangkok
Tel: 66-2-694-1351

越南 Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

欧洲

奥地利 Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

丹麦
Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

芬兰 Finland - Espoo
Tel: 358-9-4520-820

法国 France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 Germany - Garching
Tel: 49-8931-9700

德国 Germany - Haan
Tel: 49-2129-3766400

德国 Germany - Heilbronn
Tel: 49-7131-67-3636

德国 Germany - Karlsruhe
Tel: 49-721-625370

德国 Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

德国 Germany - Rosenheim
Tel: 49-8031-354-560

以色列 Israel - Ra'anana
Tel: 972-9-744-7705

意大利 Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

意大利 Italy - Padova
Tel: 39-049-7625286

荷兰 Netherlands - Druenen
Tel: 31-416-690399
Fax: 31-416-690340

挪威 Norway - Trondheim
Tel: 47-7289-7561

波兰 Poland - Warsaw
Tel: 48-22-3325737

罗马尼亚
Romania - Bucharest
Tel: 40-21-407-87-50

西班牙 Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

瑞典 Sweden - Gothenberg
Tel: 46-31-704-60-40

瑞典 Sweden - Stockholm
Tel: 46-8-5090-4654

英国 UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820