第10章模数转换器 (ADC)

10.1 F2802X的ADC内核介绍和寄存器列表
10.2 SOC的工作原理
10.3 ADC的其他要点
10.4 ADC重要寄存器
10.5 ADC应用举例

第10章 模数转换器（ADC)

常用ADC的分类

双积分
逐次比较
Σ-Δ

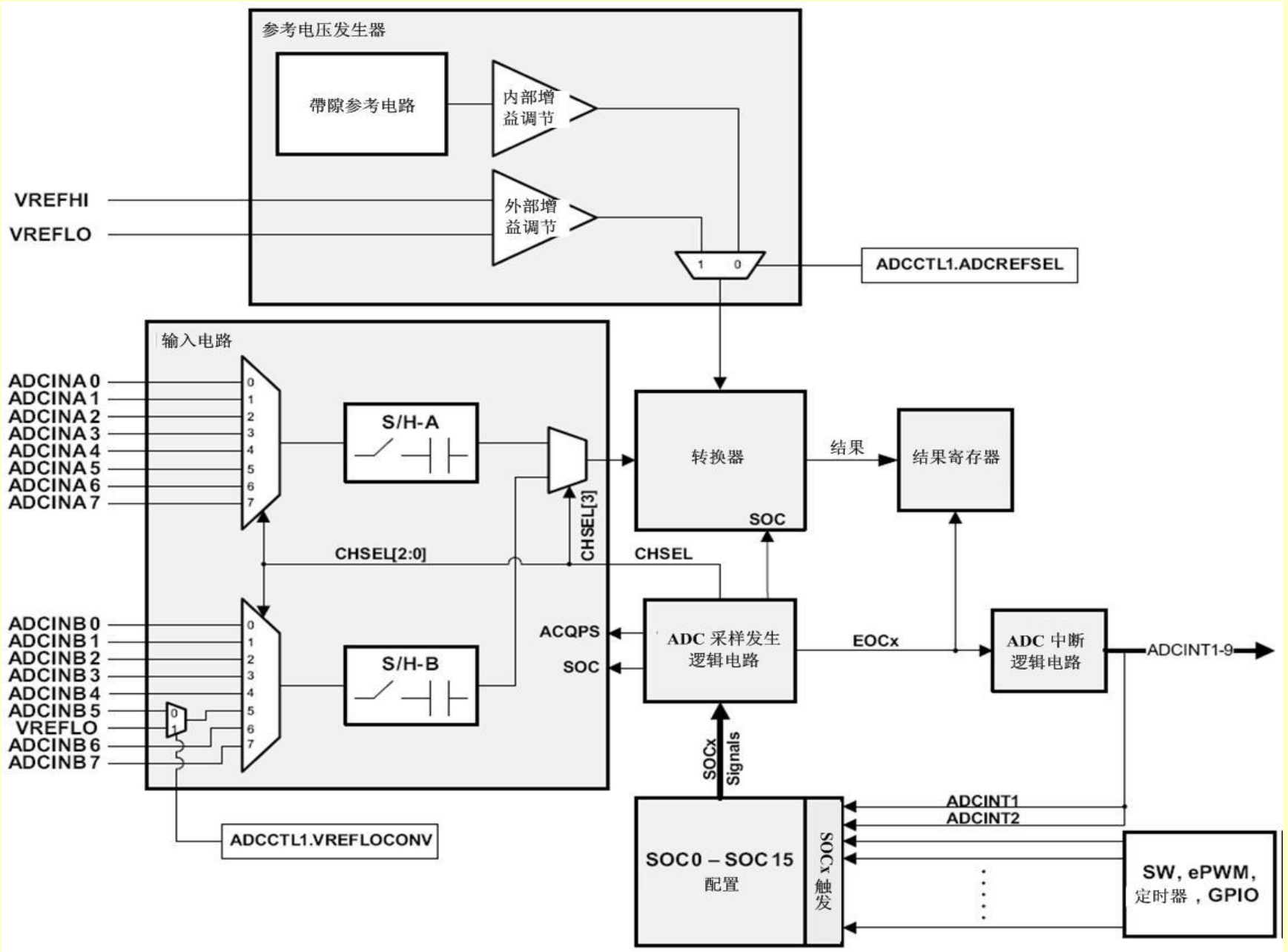
ADC的选择

精度(分辨率，转换误差...)
转换速度(采样保持时间，转换时间)
接口电路

ADC使用的一般要点

1. 配置引脚和工作（参考）电源
2. 配置转换顺序和采样方式
3. 配置转换速度(采样保持时间，转换时间)
4. 配置触发方式
5. 配置优先级
6. 配置结果读取方式
7. 配置中断方式
8. 其他特殊功能

10.1 F2802X的ADC内核介绍和寄存器列表



- ✓ 内置两个采样/保持(S/H)电路的12位ADC内核
- ✓ 模拟输入的满量程：0V～3.3V(固定的)，或者VREFHI～VREFLO(比例模式)
- ✓ 以全系统时钟运行，无需预分频
- ✓ 多达16个多路复用输入通道
- ✓ 16个**SOC**，其触发源、采样窗口和通道均可配置
- ✓ 16个结果寄存器(可单独寻址)，用于存储转换值

与之前那些ADC相反，现在这种ADC不基于序列发生器。用户可以很容易地从一个触发器创建一串转换结果。但是，操作的基本原则主要围绕着每个单独转换的配置，我们称作“SOC”，即“开始转换（Start-Of-Conversion）

- ✓ 多个触发源
  - ➢ S/W —软件立即启动
  - ➢ ePWM1 ~ ePWM7
  - ➢ GPIO XINT2
  - ➢ CPU定时器0/1/2
  - ➢ ADCINT1/2
- ✓ 9个灵活的PIE中断，在任意转换之后可以配置中断请求

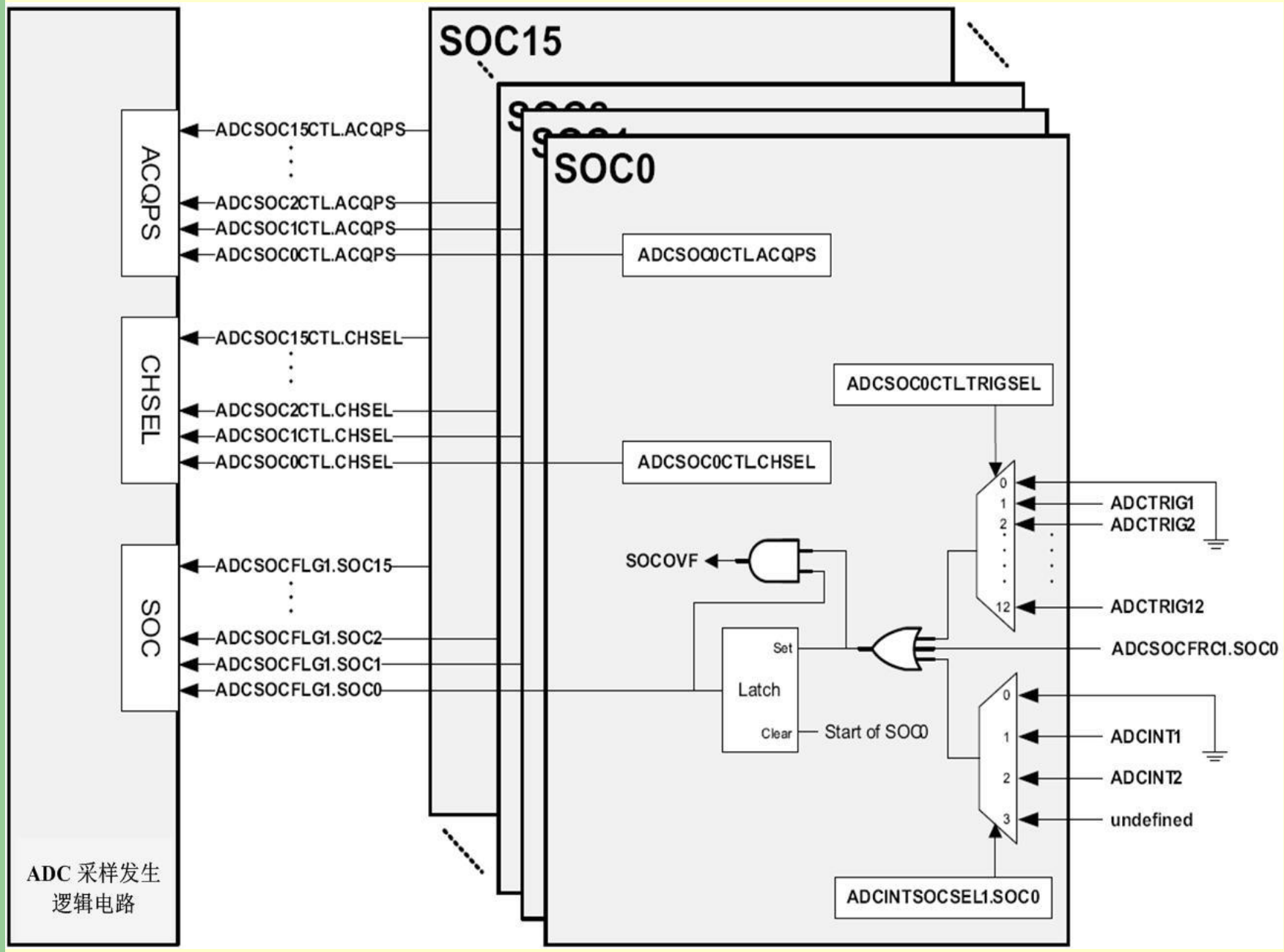
表10.1 ADC相关寄存器（AdcRegs和AdcResult)

寄存器名称	地址偏移量	大小（x16）	描述
ADCCTL1	0x00	1	控制1寄存器[1]
ADCINTFLG	0x04	1	中断标志寄存器
ADCINTFLGCLR	0x05	1	中断标志清除寄存器
ADCINTOVF	0x06	1	中断溢出寄存器
ADCINTOVFCLR	0x07	1	中断溢出清除寄存器
ADCINTSEL1AND2	0x08	1	中断1和2选择寄存器[1]
ADCINTSEL3AND4	0x09	1	中断3和4选择寄存器[1]
ADCINTSEL5AND6	0x0A	1	中断5和6选择寄存器[1]
ADCINTSEL7AND8	0x0B	1	中断7和8选择寄存器[1]
ADCINTSEL9AND10	0x0C	1	中断9选择寄存器[1]
SOCPRICTL	0x10	1	SOC优先级控制寄存器[1]
ADCSAMPLEMODE	0x12	1	采样模式寄存器[1]
ADCINTSOCSEL1	0x14	1	中断SOC选择1寄存器（适用于8通道）[1]
ADCINTSOCSEL2	0x15	1	中断SOC选择2寄存器（适用于8通道）[1]
ADCSOCFLG1	0x18	1	SOC标志1寄存器（适用于16通道）
ADCSOCFRC1	0x1A	1	SOC强制1寄存器（适用于16通道）
ADCSOCOVF1	0x1C	1	SOC溢出1寄存器（适用于16通道）
ADCSOCOVFCLR1	0x1E	1	SOC溢出清除1寄存器（适用于16通道）
ADCSOC0CTL–ADCSOC15CTL	0x20 – 0x2F	1	SOC0控制寄存器～SOC15控制寄存器[1]
ADCREFTRIM	0x40	1	基准调节寄存器[1]
ADCOFFTRIM	0x41	1	偏置量调节寄存器[1]
ADCREV – reserved	0x4F	1	保留的修订寄存器
ADCRESULT0 – ADCRESULT15	0x00 – 0x0F	1	ADC结果0寄存器～ADC结果15寄存器，其基地址与其他ADC寄存器的基地址不同

10.2 SOC的工作原理

- ✓ 与之前的那些ADC不同，这个ADC不基于序列发生器(sequencer)，而是基于SOC
- ✓ 术语**SOC**是一种配置设置，它定义的是单通道单转换
- ✓ 该设置包含3个配置：启动转换的触发源、转换通道和采集(采样)窗口尺寸
- ✓ 每个SOC都是单独配置的，并且可以是触发源、通道和/或采集窗口的任意组合
- ✓ 这一特性提供了一种非常灵活的转换配置方式，**包括从“使用不同触发器、不同通道的单独采样”到“使用单个触发器、相同通道的过采样”**

- ✓ SOCx的触发源由ADCSOCxCTL寄存器中的TRIGSEL字段以及ADCINTSOCSEL1或ADCINTSOCSEL2寄存器中的相应位联合配置
- ✓ 软件通过ADCSOCFRC1寄存器也可以强制产生一个SOC事件
- ✓ SOCx的通道和采样窗尺寸都通过ADCSOCxCTL寄存器的CHSEL和ACQPS字段来配置


SOC15
SOC0
ACQPS
CHSEL
SOC
ADCSOC15CTL.ACQPS
ADCSOC2CTL.ACQPS
ADCSOC1CTL.ACQPS
ADCSOC0CTL.ACQPS
ADCSOC15CTL.CHSEL
ADCSOC2CTL.CHSEL
ADCSOC1CTL.CHSEL
ADCSOC0CTL.CHSEL
ADCSOCFLG1.SOC15
ADCSOCFLG1.SOC2
ADCSOCFLG1.SOC1
ADCSOCFLG1.SOC0
ADCSOC0CTL.ACQPS
ADCSOC0CTL.CHSEL
ADCSOC0CTL.TRIGSEL
SOCOVF
Set
Latch
Clear
Start of SOC0
0
1
2
12
ADCTRIG1
ADCTRIG2
ADCTRIG12
ADCSOCFRC1.SOC0
0
1
2
3
ADCINT1
ADCINT2
undefined
ADCINTSOCSEL1.SOC0
ADC 采样发生
逻辑电路

10.2.1 ADC采集(采样和保持)窗口

- ✓ 各外部驱动器驱动模拟信号的能力各不相同，有些电路需要更长的时间才能正确地将电荷(charge)传送到ADC的采样电容器(sampling capacitor)
- ✓ 为解决这个问题，**可对每个SOC采样窗口的长度进行控制，每个ADCSOCxCTL寄存器都包含一个6位字段ACQPS，该字段决定采样和保持(S/H)窗的长度**
- ✓ 对于SOC来说，写入该字段的值要比采样窗的期望值(单位为周期数)小1，如果该字段为15,则表示采样时间其实为16个时钟周期,采样周期的最小值为7，(ACQPS=6)
- ✓ **总采样时间等于采样窗长度加上ADC转换时间(共13个ADC时钟)**

ADC时钟	ACQPS	采样窗口	转换时间(13个周期)	处理模拟电压的总时间
60MHz	6	116.67ns	216.67ns	333.34ns
60MHz	8	150.00ns	216.67ns	366.67ns
60MHz	10	183.33ns	216.67ns	400.00ns
60MHz	14	250.00ns	216.67ns	466.67ns
60MHz	25	433.33ns	216.67ns	650.00ns
40MHz	6	175ns	325ns	500.00ns
40MHz	25	650ns	325ns	975.00ns

10.2.2 触发操作

- ✓ **每个SOC都可以配置成由许多输入触发器中的其中一个启动。如果需要，几个SOC可以配置成使用同一个通道**
- ✓ 一些可用的输入触发器:
  - ➢ 软件
  - ➢ CPU定时器0/1/2
  - ➢ XINT2 SOC
  - ➢ ePWM1～7 SOCA和SOCB
- ✓ **ADCINT1和ADCINT2可以反馈回来触发另外一次转换，这种配置由ADCINTSOCSEL1/2寄存器控制，如果需要连续转换，这种模式将非常有用**

10.2.3 通道选择

- ✓ **每个SOC都可以配置成转换任意一个可用的ADCIN输入通道**
- ✓ 当SOC被配置成顺序采样模式时，由ADCSOCxCTL寄存器的4位CHSEL字段决定要转换的通道
- ✓ 当SOC被配置成同步采样模式时，CHSEL字段的最高有效位(第4位)将被丢弃且由低3位决定转换哪一对通道
- ✓ ADCINA0与VREFHI复用，因此在使用外部电压基准时，它不能用作一个可变的输入源

ADC SOC0～SOC15控制寄存器（ADCSOCxCTL）

15	11	10	9	6	5	0
TRIGSEL		Reserved	CHSEL		ACQPS	
R/W-0		R-0	R/W-0		R/W-0	

位	名称	值	描述
15-11	TRIGSEL		SOCx触发源选择位。 配置由哪个触发源在ADCSOCFLG1寄存器中设置SOCx标志并在SOCx拥有优先权时启动转换。ADCINTSOCSEL1或ADCINTSOCSEL2寄存器中相应的SOCx字段的值优先于TRIGSEL字段。
		00h	ADCTRIG0——仅由软件触发
		01h	ADCTRIG1——CPU定时器0.TINT0n
		02h	ADCTRIG2——CPU定时器1.TINT1n
		03h	ADCTRIG3——CPU定时器2.TINT2n
		04h	ADCTRIG4——XINT2.XINT2SOC
		05h	ADCTRIG5——ePWM1.ADCSOCA
		06h	ADCTRIG6——ePWM1.ADCSOCB
		07h	ADCTRIG7——ePWM2.ADCSOCA
		08h	ADCTRIG8——ePWM2.ADCSOCB
		09h	ADCTRIG9——ePWM3.ADCSOCA
		0Ah	ADCTRIG10——ePWM3.ADCSOCB
		0Bh	ADCTRIG11——ePWM4.ADCSOCA
		0Ch	ADCTRIG12——ePWM4.ADCSOCB
		0Dh	ADCTRIG13——ePWM5.ADCSOCA
		0Eh	ADCTRIG14——ePWM5.ADCSOCB
		0Fh	ADCTRIG15——ePWM6.ADCSOCA
		10h	ADCTRIG16——ePWM6.ADCSOCB
		11h	ADCTRIG17——ePWM7.ADCSOCA
		12h	ADCTRIG18——ePWM7.ADCSOCB
		其它值	无效选择
10	Reserved		读返回0；对该位执行写操作无影响

位	名称	值	描述
9-6	CHSEL		SOCx通道选择位。用于在ADC接收到SOCx时选择将被转换的通道。
			顺序采样模式（SIMULENx = 0）：
		0h	ADCINA0
		1h	ADCINA1
		2h	ADCINA2
		3h	ADCINA3
		4h	ADCINA4
		5h	ADCINA5
		6h	ADCINA6
		7h	ADCINA7
		8h	ADCINB0
		9h	ADCINB1
		Ah	ADCINB2
		Bh	ADCINB3
		Ch	ADCINB4
		Dh	ADCINB5
		Eh	ADCINB6
		Fh	ADCINB7
			同步采样模式（SIMULENx = 1）：
		0h	ADCINA0/ADCINB0对
		1h	ADCINA1/ADCINB1对
		2h	ADCINA2/ADCINB2对
		3h	ADCINA3/ADCINB3对
		4h	ADCINA4/ADCINB4对
		5h	ADCINA5/ADCINB5对
		6h	ADCINA6/ADCINB6对
		7h	ADCINA7/ADCINB7对
		8h	无效选择
		9h	无效选择
		Ah	无效选择
		Bh	无效选择
		Ch	无效选择
		Dh	无效选择
		Eh	无效选择
		Fh	无效选择

位	名称	值	描述
5-0	ACQPS		SOCx采集预分频控制位。该位用于控制SOCx其采样/保持窗口的长度，最小值可以为6。
		00h	无效选择
		01h	无效选择
		02h	无效选择
		03h	无效选择
		04h	无效选择
		05h	无效选择
		06h	采样窗口7个周期长（6+1个时钟周期）
		07h	采样窗口8个周期长（7+1个时钟周期）
		08h	采样窗口9个周期长（8+1个时钟周期）
		09h	采样窗口10个周期长（9+1个时钟周期）
		…	…….
		3Fh	采样窗口64个周期长（63+1个时钟周期）

10.2.4 SOC配置其他注意事项

每个SOC都是单独配置的，
并且可以是触发源、通道和/或采集窗口的任意组合。

这一特性提供了一种非常灵活的转换配置方式：
包括从“使用不同触发器、不同通道的单独采样”
到“使用单个触发器、相同通道的过采样”。

在ADC的应用实例中还必须通过PCLKCR0寄存器将时钟使能，且必须让ADC上电并正常工作。

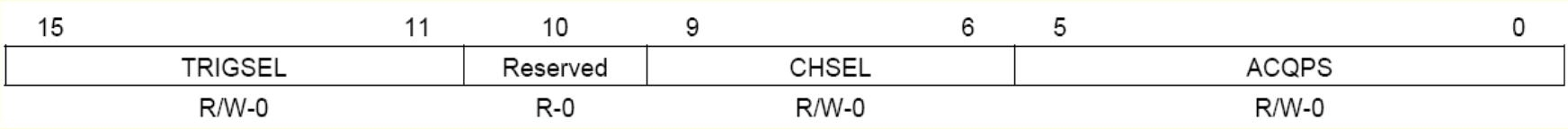
举例

- ✓ 如果要将通道ADCINA1上的单一转换设置成在ePWM3定时器到达周期值时发生
- ✓ 首先必须将ePWM3设置成在到达周期值时输出一个SOCA或SOCB信号。此时，我们选择SOCA。
- ✓ 然后将其中一个SOC设置成使用其ADCSOCxCTL寄存器。由于选择哪个SOC信号都没有什么区别，所以我们使用SOC0。
- ✓ ADC允许采样窗最短为7个周期。如果要让采样窗的时间最短，转换通道为ADCINA1，SOC0触发器为ePWM3，我们必须分别将ACQPS字段设为6，CHSEL字段设为1，TRIGSEL字段设为9。这样，写入寄存器的结果值将为：

15 11	10	9 6	5 0
TRIGSEL	Reserved	CHSEL	ACQPS
R/W-0 01001	R-0 0	R/W-0 0001	R/W-0 000110

ADCSOC0CTL = 4846h;

- ✓ 如果配置成这样的话，ePWM3 SOCA事件将启动一次ADCINA1转换，并将结果值存放在ADCRESULT0寄存器中



15 11	10	9 6	5 0
TRIGSEL	Reserved	CHSEL	ACQPS
R/W-0	R-0	R/W-0	R/W-0

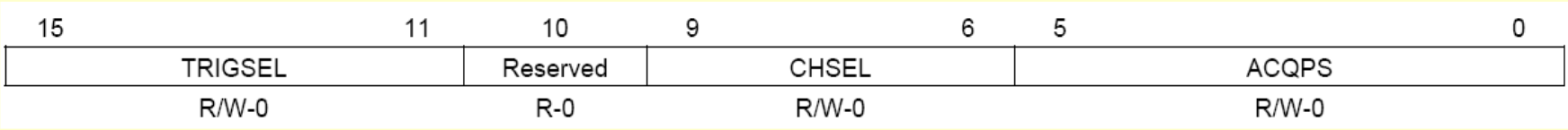
- ✓ 如果ADCINA1需要被3次**过采样**，那么可以将SOC1、SOC2和SOC3配置成和SOC0一样

```
ADCSOC1CTL = 4846h;
ADCSOC2CTL = 4846h;
ADCSOC3CTL = 4846h;
```

- ✓ 如果配置成这样的话，ePWM3 SOCA事件将连续启动4次ADCINA1转换，并将结果值分别存放在ADCRESULT0～ADCRESULT3寄存器中

过采样

- ➢ **可以提高ADC的精度**
- ➢ **合理编排，可以起到其他方面很好的效果，比如积分、滤波等等**
- ➢ **在DSP控制中广泛采样**



15 11	10	9 6	5 0
TRIGSEL	Reserved	CHSEL	ACQPS
R/W-0	R-0	R/W-0	R/W-0

- ✓ 可能需要从同一触发器采样3个不同的信号。这可以通过改变SOC0 ~ SOC2的CHSEL字段并将TRIGSEL字段保持不变来实现

```
ADCSOC0CTL = 4846h; // (ACQPS = 6, CHSEL = 1, TRIGSEL = 9)
ADCSOC1CTL = 4886h; // (ACQPS = 6, CHSEL = 2, TRIGSEL = 9)
ADCSOC2CTL = 48C6h; // (ACQPS = 6, CHSEL = 3, TRIGSEL = 9)
```

- ✓ 如果配置成这样的话，ePWM3 SOCA事件将连续启动3次转换。通道ADCINA1上转换的结果将在ADCRESULT0中显示。ADCINA2通道上转换的结果将在ADCRESULT1中显示。ADCINA3通道上转换的结果将在ADCRESULT2中显示。
- ✓ **转换通道和触发器这两者与转换结果在何处显示没有任何关系。RESULT寄存器与SOC有关**

10.3 ADC的其他要点

（1）ADC转换优先级

- ✓ 当几个SOC标志同时置位时，有两种优先级方式可用来决定它们转换的顺序
- ✓ 默认的优先级方式为**轮询机制**(round robin)
- ✓ ADCSOCPRIORITYCTL寄存器中的SOCPRIORITY字段可以为单个SOC或所有SOC分配高优先级

轮询机制

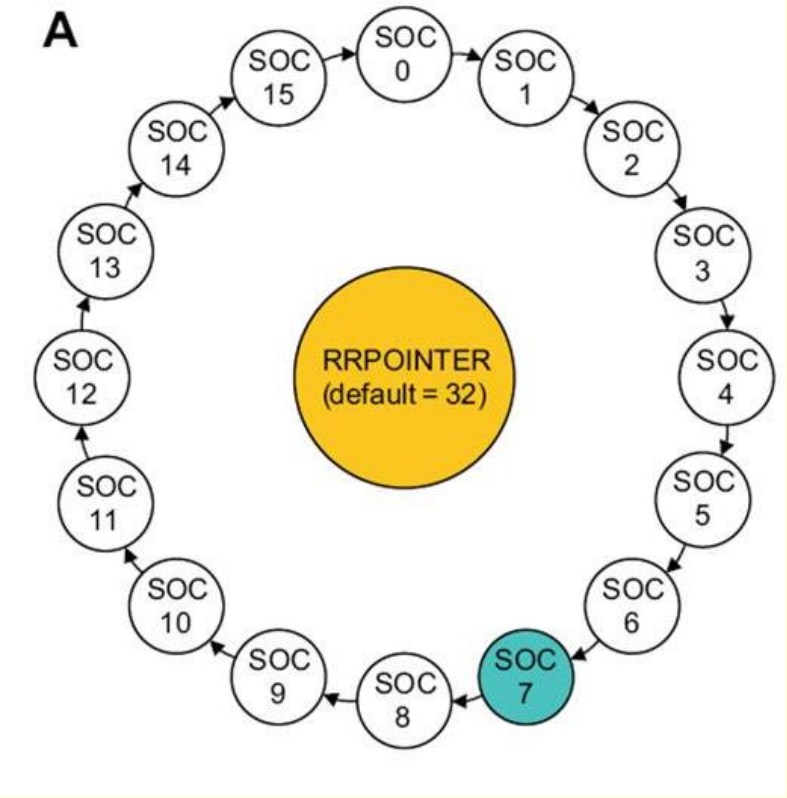
在这种机制中，各SOC之间的优先级都一样。

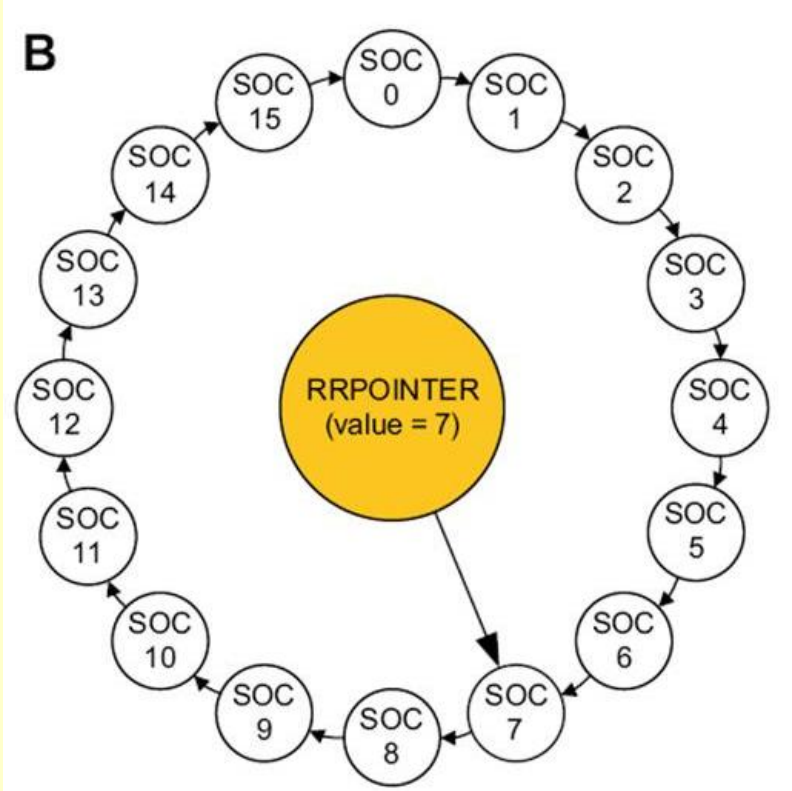
优先级仅取决于**轮询指针**(RRPOINTER)。

RRPOINTER的值在ADCSOCPRIORITYCTL寄存器中反映，它指向上一个被转换的SOC。下一个比RRPOINTER大的值将被设为优先级最高的SOC，并在SOC15之后又绕回SOC0。

因为0表示已经发生了转换，所以复位时RRPOINTER的值为32。当RRPOINTER等于32时，SOC0将被设为最高优先级。

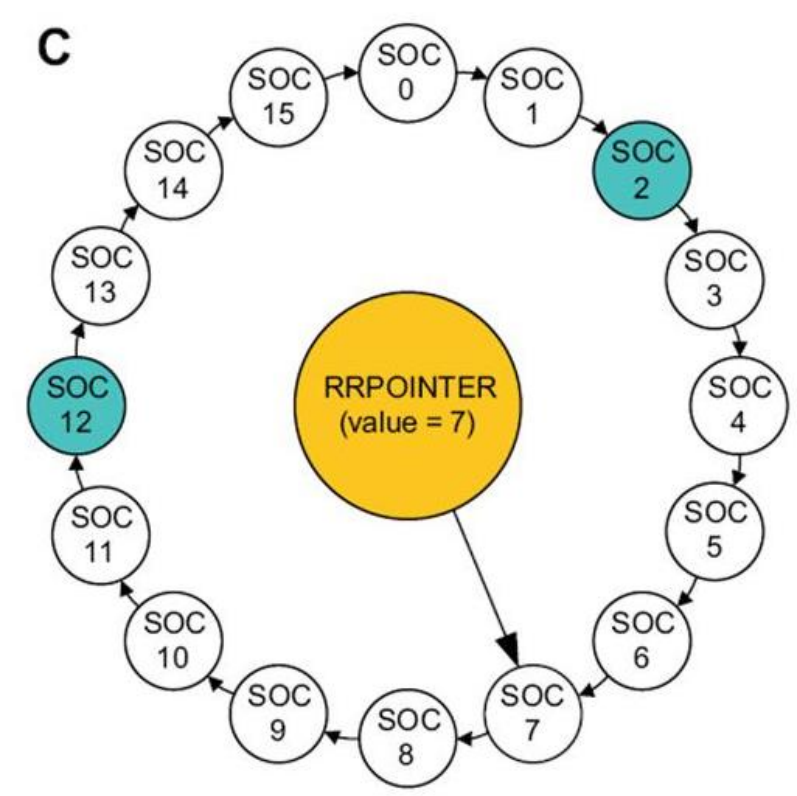
当ADCCTL1.RESET位置1或SOCPRICTL寄存器被写时，器件复位且RRPOINTER也被复位

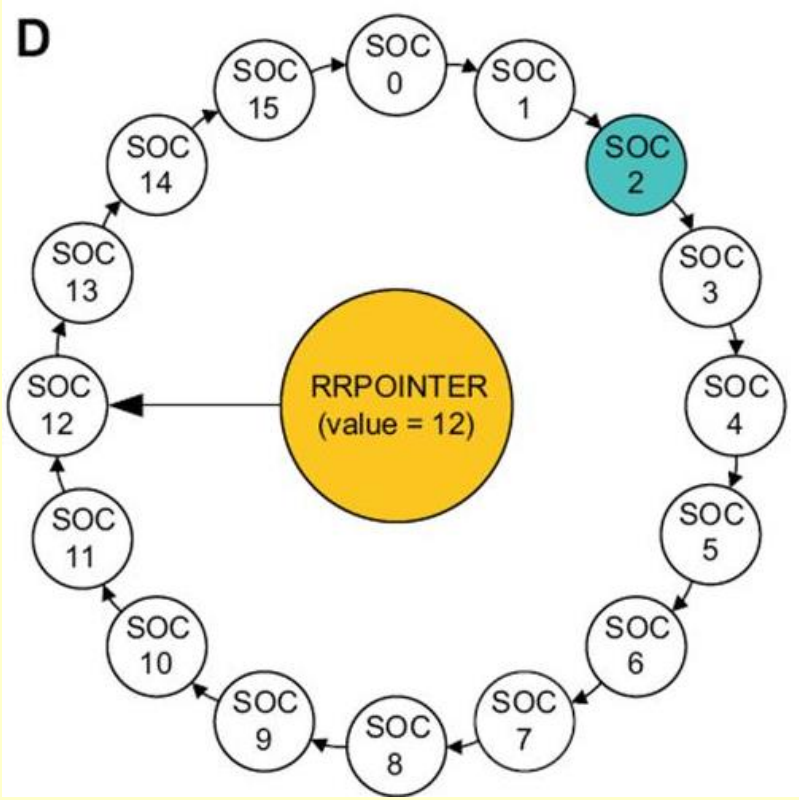




复位后，SOC0是优先级最高的SOC；SOC7收到触发事件，SOC7配置通道被立即转换

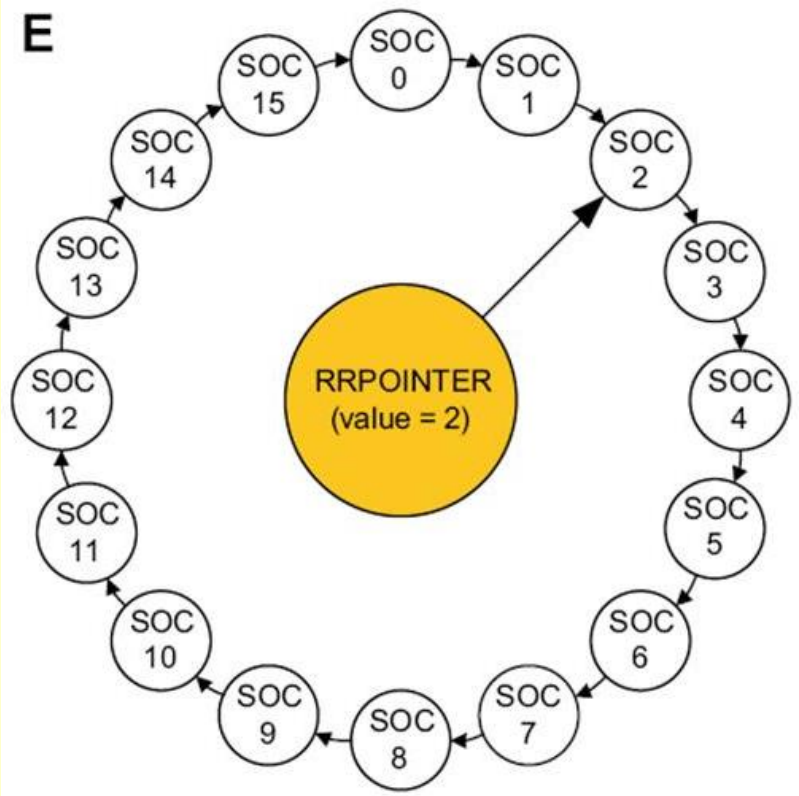
REPOINTER转而指向SOC7，SOC8现在是优先级最高的SOC





SOC2&SOC12同时收到触发事件，SOC12首先出现在round robin wheel上，SOC12配置通道被立即转换，同时SOC2挂起

REPOINTER转而指向SOC12，SOC2配置通道在被转换



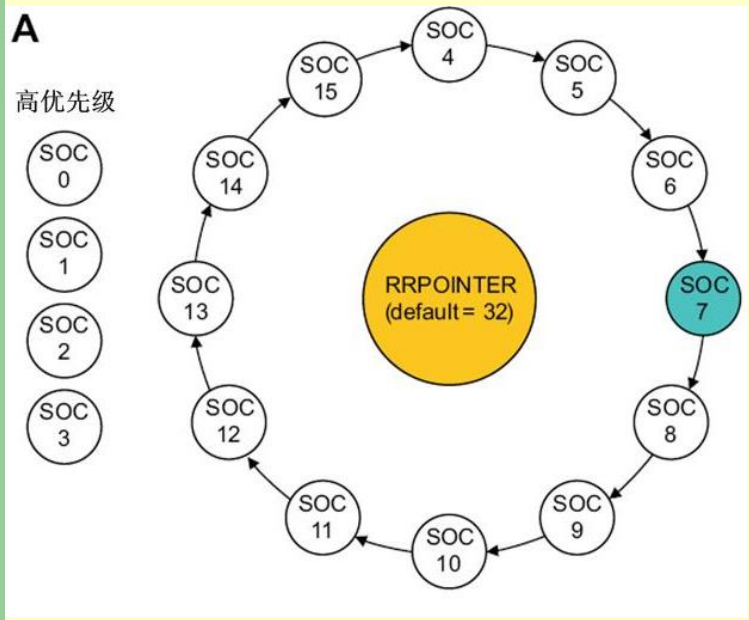
REPOINTER转而指向SOC2，SOC3现在是优先级最高的SOC

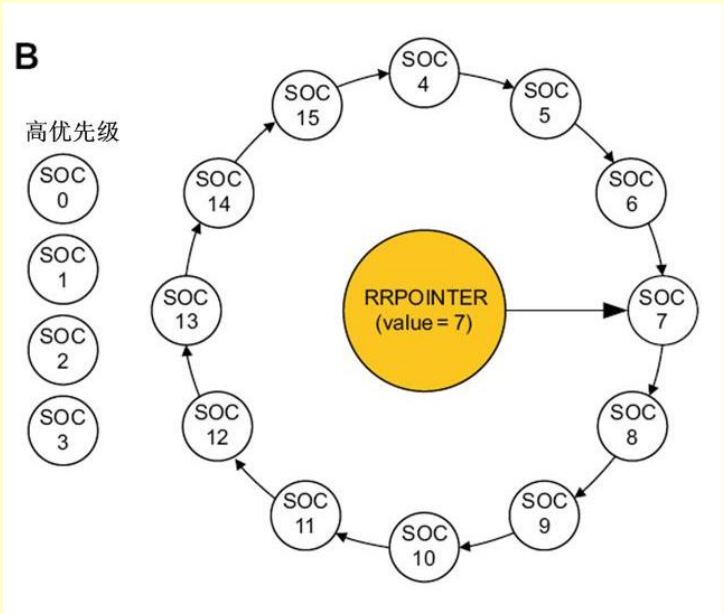
分配优先级

ADCSOCPRIORITYCTL寄存器中的SOCPRIORITY字段可以为单个SOC或所有SOC分配高优先级。当某个SOC被配置为高优先级时，在当前转换结束后它会中断当前这个**轮询环**(round robin wheel)并插入进去作为下一个转换的SOC。当它转换结束后，轮询环会在它中断的地方继续工作。如果两个高优先级的SOC同时被触发，那么编号较小的那个SOC将先转换

高优先级首先被分配给SOC0，然后按照数字递增的顺序分配。写入SOCPRIORITY字段的值定义第一个不是高优先级的SOC。换句话说，如果写入SOCPRIORITY的值为4，那么SOC0、SOC1、SOC2和SOC3将被定义为高优先级，其中SOC0的优先级最高

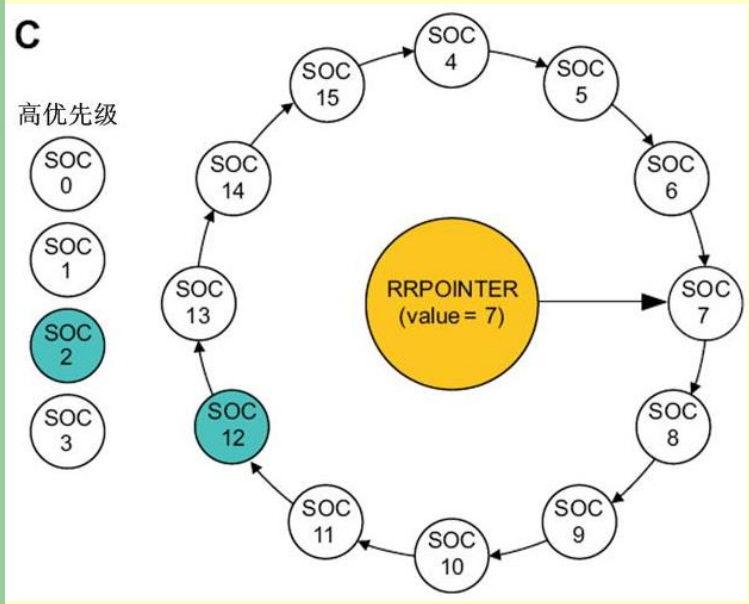
举例：当SOCPRIORITY=4时

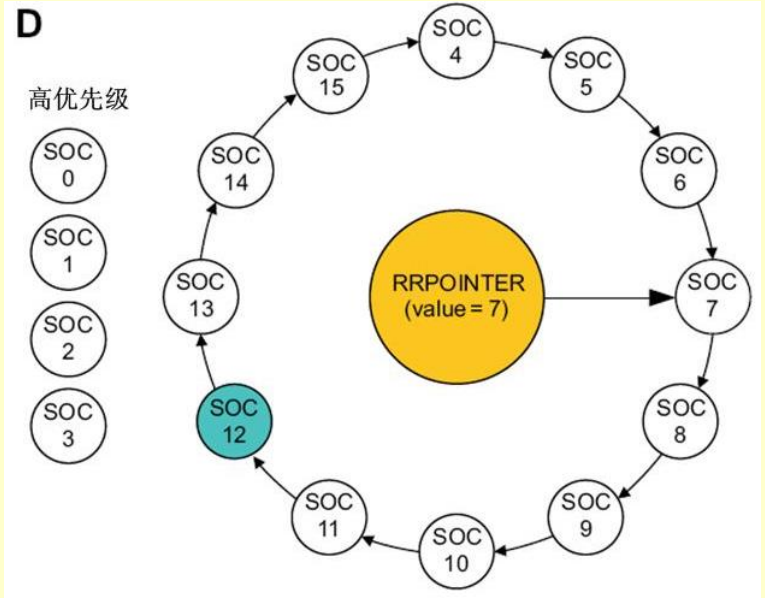




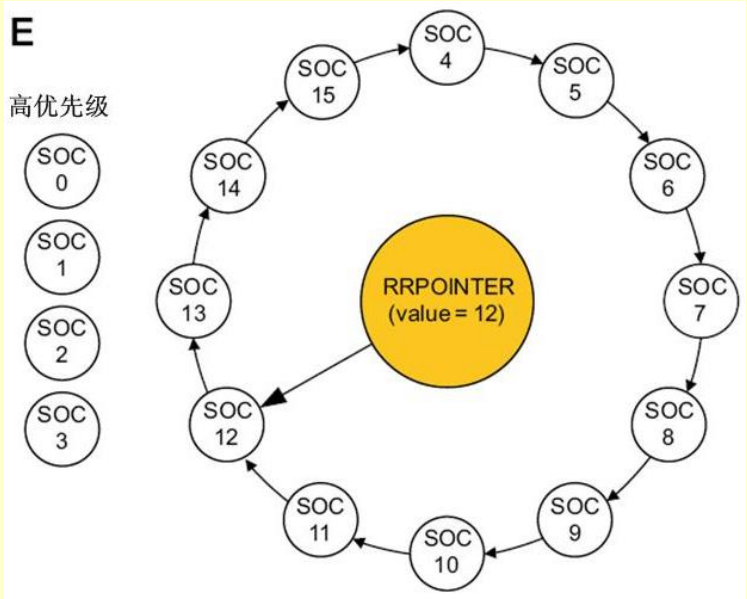
复位后，SOC4是round robin wheel上的No 1，SOC7收到触发事件，SOC7配置通道被立即转换

REPOINTER转而指向SOC7，SOC8现在是round robin wheel上的No 1





SOC2&SOC12同时收到触发事件，SOC2中断round robin wheel，且SOC2配置通道被转换，同时SOC12挂起REPOINTER转而指向7，SOC12配置通道现在被转换



REPOINTER转而指向SOC12，SOC13现在是round robin wheel上的No 1

(2) 同步采样模式

在某些应用中需要保证两个信号之间的采样延迟最短。ADC包含两个采样/保持电路，允许同时对两个不同的通道进行采样。

同步采样模式使用ADCSAMPLEMODE寄存器对一对SOCx进行配置。

偶数编号的SOCx和它之后的奇数编号的SOCx(即SOC0和SOC1)配成一对，一起连接一个使能位(此时为SIMULEN0)

配对规则如下：

- ✓ 任意一个SOCx触发源都可以启动一对转换
- ✓ 那对转换通道将由A通道和B通道组成，这两个通道由被触发SOCx的CHSEL值决定。这种模式下，有效值为0～7
- ✓ 同时采样两个通道
- ✓ 一直是首先转换A通道
- ✓ A通道转换后将会产生偶数编号的EOCx脉冲，B通道转换后则产生奇数编号的EOCx脉冲
- ✓ A通道的转换结果存放在偶数编号的ADCRESULTx寄存器中，B通道的转换结果则存放在奇数编号的ADCRESULTx寄存器中

举例

如果ADCSAMPLEMODE.SIMULEN0=1且SOC0配置为

CHSEL = 2(ADCINA2/ADCINB2通道);

TRIGSEL = 5(ADCTRIG5=ePWM1.ADCSOCA);

当ePWM1发出一个ADCSOCA触发事件时，ADCINA2和ADCINB2将会同时被采样。

之后ADCINA2通道将会立即被转换，且转换结果存放在ADCRESULT0寄存器中。

根据ADCCTL.INTPULSEPOS的设置，在ADCINA2转换开始或结束时将出现EOC0脉冲。接着ADCINB2通道被转换，且转换结果存放在ADCRESULT1寄存器中。

根据ADCCTL1.INTPULSEPOS的设置，在ADCINB2转换开始或结束时将出现EOC1脉冲

在典型的应用中，通常希望只使用SOCx对中的偶数SOCx。不过，也可以使用SOCx对中的奇数SOCx，甚至偶数SOCx和奇数SOCx都使用。

当偶数SOCx和奇数SOCx都使用时，两个SOCx触发源都会启动转换。但需要注意的是，因为两个SOCx都将它们的转换结果存放在同一个ADCRESULTx寄存器中，因此两者可能会相互覆盖。

SOCx的优先级规则与顺序采样模式相同

(3) EOC和中断操作

就像有16组独立的SOCx配置一样，同样也有16个EOCx脉冲。

在顺序采样模式下，EOCx与SOCx是直接相关的。

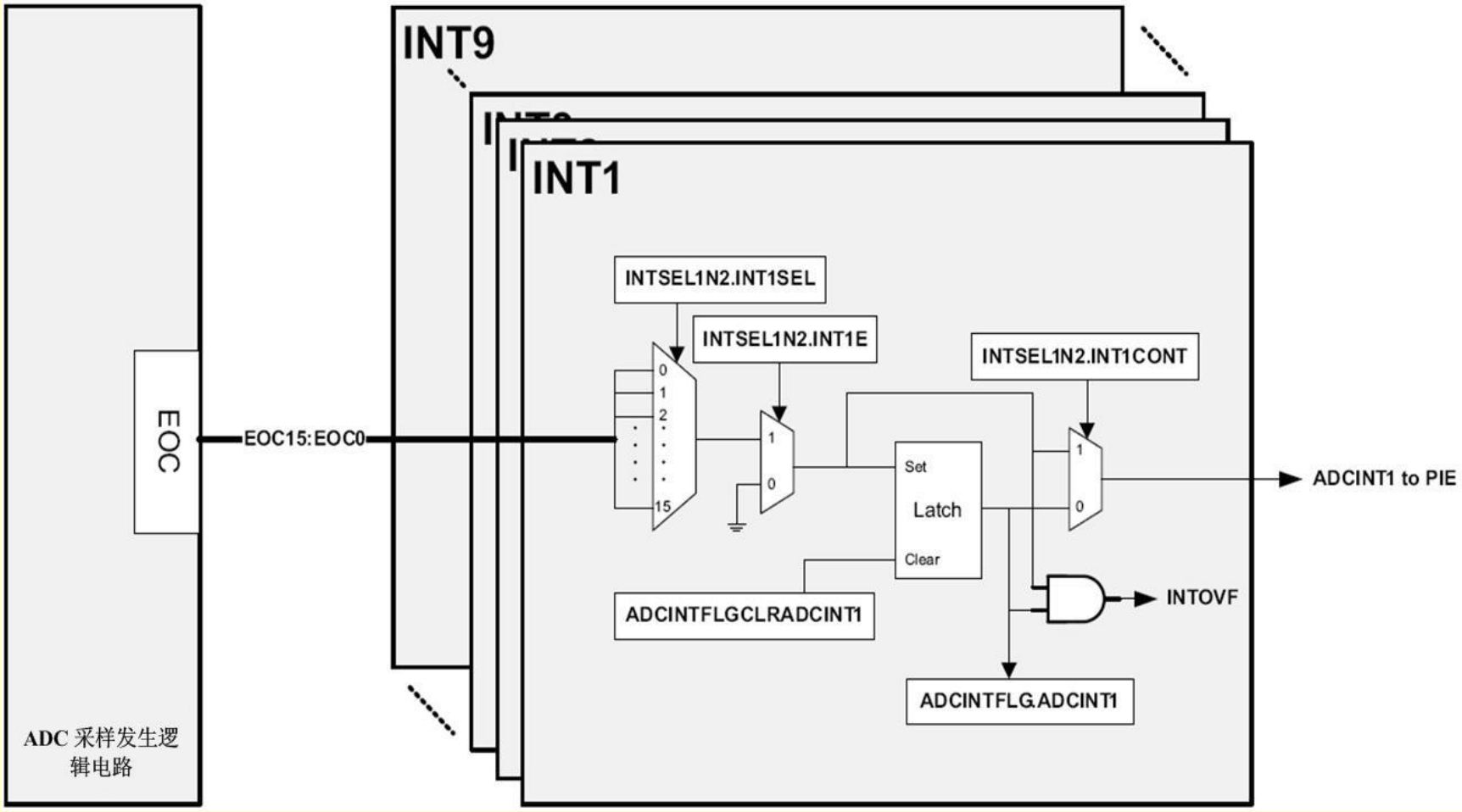
在同步采样模式下，“偶数EOCx和其后的奇数EOCx”与“偶数SOCx和其后的奇数SOCx”是相关的，如上节所述。根据ADCCTL1.INTPULSEPOS的设置，EOCx脉冲将在转换开始或转换结束时出现

ADC包含9个中断，这些中断可以被标记(flag)和/或传送到PIE。

其中每一个中断都可以配置成接收任何一个EOCx信号作为其触发源。

至于哪个EOCx是触发源，则由INTSELxNy寄存器决定。

此外，ADCINT1和ADCINT2信号可以配置成产生一个SOCx触发事件。这对连续转换来说非常有用


INT9
INT1
INTSEL1N2.INT1SEL
INTSEL1N2.INT1E
INTSEL1N2.INT1CONT
EOC
EOC15:EOC0
0
1
2
15
1
0
Set
Latch
Clear
1
0
ADCINT1 to PIE
INTOVF
ADCINTFLGCLR.ADCINT1
ADCINTFLG.ADCINT1
ADC 采样发生逻辑电路

ADC中断的结构框图

（4）上电顺序

ADC复位到ADC关闭状态。

在对任意一个ADC寄存器执行写操作之前，PCLKCR0寄存器中的ADCENCLK位必须置1。

ADC的上电顺序如下：

- 第一步：若需要外部基准，可通过ADCCTL1寄存器的位3（ADCREFSEL）使能；
- 第二步：通过置位ADCCTL1寄存器中的7-5位（ADCPWDN、ADCBGPWD、ADCREFPWD）将基准、带隙和模拟电路一起上电；
- 第三步：通过置位ADCCTL1寄存器的位14（ADCENABLE）使能ADC。

在执行第一次转换之前，在第二步后面需要1毫秒的延迟，第一步到第三步也可以同时执行。在ADC掉电时，第二步中的那3个位可以同时清零。

（5） ADC校准(calibration)

零偏置误差(offset error)和**满量程增益误差**(gain error)是转换器的固有特性。

厂家在25℃时对ADC的偏置误差和增益误差进行了校准，与此同时也允许用户根据特定应用环境的需要(例如环境温度)修改校正。

除了在特定的仿真条件下，或者需要修改厂家的设置，否则用户无需执行任何动作。ADC在器件启动过程期间将被正确校准

偏置误差：理想的ADC是当类比电压输入为0V，对应的数字编码也为0；但实际上是输入的电压为0，但对应的数字编码不为0，其间的误差就称为偏置误差

增益误差：经过偏置误差调整为0后，若理想ADC的斜率与实际ADC的斜率不同，两者之间的斜率差就称为增益误差

厂家设置和校准功能

在校准和测试过程中，TI对几个ADC设置和一对内部振荡器的设置进行校准。

这些设置被嵌入到TI保留的OTP内存中，成为Device_call( )的一部分。当在启动Boot ROM程序的过程中调用这个函数时，厂家设置会被写入相应的active寄存器中。

在这之前，ADC和内部振荡器一直坚持使用指定参数。

如果在仿真期间省略了启动过程，那么用户必须保证将调节设置写入它们相应的寄存器中，这样才能保证ADC和内部振荡器的设置满足数据手册中的技术规范。这项工作可通过手动或在应用程序中调用该函数来完成，也可以通过CCS直接执行写操作来完成

ADC零偏置校准

零偏置误差被定义为在输入电压等于VREFLO时转换得到的数字值。这个基本误差影响着所有ADC转换，并且和满量程增益和线性规范一起决定着转换器的DC精度。

零偏置误差可以是正的，表示在输入电压为VREFLO时输出一个正的数字值；零偏置误差也可以是负的，表示一个比VREFLO高一个步长(step)的电压仍被读作一个数字零

为了校正这种误差，将这两个误差的补数写入ADCOFFTRIM寄存器。这个寄存器包含的值在转换结果存放在ADC结果寄存器之前应用。这一操作完全在ADC内核内进行，因此结果的时序将不会受影响，并且ADC的全动态范围将会为所有调节值保留

虽然调用Device_cal()函数可以向ADCOFFTRIM寄存器写入经过工厂校准的偏置误差校正值，但用户也可以修改ADCOFFTRIM寄存器，以补偿由应用环境引起的其它偏置误差。这项工作无需通过ADC通道就可完成，方法很简单，只需使用ADCCTRL1寄存器中的VREFLOCONV位即可

ADC满量程增益校准

增益误差是一个随着输入电压递增而递增的误差。满量程增益误差在输入电压最大时出现。

像偏置误差一样，增益误差也有正负。正的满量程增益误差表示在输入最大电压之前就已到达满量程数字结果了。负的满量程增益误差表示从未到达过满量程数字结果。

校准函数Device_cal()向ADCREFTRIM寄存器写入一个厂家调节值，以此来校正ADC满量程增益误差。这个寄存器在调用Device_cal()函数之后不能修改

(6) 参考电压的选择

内部参考电压

默认情况下选择内部带隙(bandgap)来产生ADC参考电压。这样一来，将根据固定的0 ~ 3.3V量程范围来转换电压。控制这种转换模式的等式如下

Digital Value = 0 输入 ≤ 0v时

Digital Value = 4096*[(Input – VREFLO)/3.3v] 0v < 输入 < 3.3v时

Digital Value = 4095, 输入 ≥3.3v时

注：

所有小数部分都被截掉
在该模式下，VREFLO必须与地连接

外部参考电压

若要将一个输入电压按比例转换，则应该选择外部VREFH/VREFLO管脚作为参考电压的引脚。与内部带隙模式的0 ~ 3.3v固定输入范围不同，比例模式的输入范围是VREFLO到VREFHI。转换值将被调节到该范围内。例如，如果VREFLO被设置成0.5v且VREFHI被设置成3.0v，那么1.75v电压将被转换成数字2048。VREFLO和VREFHI的允许范围请参阅器件具体的数据手册。在一些器件中，VREFLO与地在内部连接，因此被限制为0V。控制这种转换模式的等式如下：

Digital Value = 0 输入≤ VREFLO时

Digital Value = 4096*[(Input – VREFLO)/(VREFHI – VREFLO)]
VREFLO < 输入 < VREFHI时

Digital Value = 4095输入 ≥VREFHI时

10.4 ADC重要寄存器

寄存器名称	地址偏移量	大小（x16）	描述
ADCCTL1	0x00	1	控制 1 寄存器[1]
ADCINTFLG	0x04	1	中断标志寄存器
ADCINTFLGCLR	0x05	1	中断标志清除寄存器
ADCINTOVF	0x06	1	中断溢出寄存器
ADCINTOVFCLR	0x07	1	中断溢出清除寄存器
ADCINTSEL1AND2	0x08	1	中断 1 和 2 选择寄存器[1]
ADCINTSEL3AND4	0x09	1	中断 3 和 4 选择寄存器[1]
ADCINTSEL5AND6	0x0A	1	中断 5 和 6 选择寄存器[1]
ADCINTSEL7AND8	0x0B	1	中断 7 和 8 选择寄存器[1]
ADCINTSEL9AND10	0x0C	1	中断 9 选择寄存器（保留中断 10 选择寄存器）
SOCPRICTL	0x10	1	SOC 优先级控制寄存器[1]
ADCSAMPLEMODE	0x12	1	采样模式寄存器[1]
ADCINTSOCSEL1	0x14	1	中断 SOC 选择 1 寄存器（适用于 8 通道）[1]
ADCINTSOCSEL2	0x15	1	中断 SOC 选择 2 寄存器（适用于 8 通道）[1]
ADCSOCFLG1	0x18	1	SOC 标志 1 寄存器（适用于 16 通道）
ADCSOCFRC1	0x1A	1	SOC 强制 1 寄存器（适用于 16 通道）
ADCSOCOVF1	0x1C	1	SOC 溢出 1 寄存器（适用于 16 通道）
ADCSOCOVFCLR1	0x1E	1	SOC 溢出清除 1 寄存器（适用于 16 通道）
ADCSOC0CTL–ADCSOC15CTL	0x20 – 0x2F	1	SOC0 控制寄存器～SOC15 控制寄存器[1]
ADCREFTRIM	0x40	1	基准调节寄存器[1]
ADCOFFTRIM	0x41	1	偏置量调节寄存器[1]
ADCREV – reserved	0x4F	1	修订寄存器
ADCRESULT0 – ADCRESULT15	0x00 – 0x0F	1	ADC 结果 0 寄存器～ADC 结果 15 寄存器

(1) ADC控制寄存器1(ADCCTL1)

15	14	13	12	8
RESET	ADCENABLE	ADCBSY	ADCBSYCHN	
R-0/W-1	R/W-0	R-0	R-0	

7	6	5	4	3	2	1	0
ADCPWN	ADCBGPWD	ADCREFPWD	Reserved	ADCREFSEL	INTPULSEPOS	VREFLO CONV	TEMPCONV
R/W-0	R/W-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0

图注：**R/W** = 读/写；**R** = 只读；**R-0/W-1** = 读取结果总为**0**，写**1**置位；**-n** = 复位后的值

位	名称	值	描述
15	RESET		ADC模块的软件复位位。这个位可使整个ADC模块复位。当器件的复位管脚被拉低（或在上电复位后）时，寄存器的所有位和状态机都复位到初始状态。这是一个一次有效（one-time-effect）的位，意思是它在置1后会立即自清零。读取这个位总是返回0。此外，ADC复位还有2个时钟周期的延迟（即，在复位ADC指令之后的两个时钟周期时，才可以修改ADC控制寄存器的其它位）
		0	无影响

		1	复位整个ADC模块（接着这个位被ADC逻辑电路置为0）
			注：ADC模块在系统复位期间也复位。如果其它任意时间需要ADC模块复位的话，可以通过写1到这个位来实现。两个时钟周期之后，你可以向ADCCTL1寄存器位写入合适的值。汇编代码如下：
			MOV ADCCTL1, #1xxxxxxxxxxxxxxxb; 复位ADC（RESET = 1）
			RPT #1 ‖ NOP; 延迟两个周期
			MOV ADCCTL1, #0xxxxxxxxxxxxxxxb; 设置成用户期望值
			注：如果默认配置就能满足需要，那么无需使用第二条MOV指令。

14	ADCENABLE		ADC 使能位
		0	禁用 ADC（不会将 ADC 掉电）
		1	使能 ADC。必须在 ADC 转换之前置位
13	ADCBSY		ADC 忙状态位
			在产生 ADC SOC 时置 1。被 ADC 状态机用来确定 ADC 是否适合采样。
			顺序模式：在 S/H 脉冲之后的 4 个 ADC 时钟时清零。
			同步模式：在 S/H 脉冲之后的 14 个 ADC 时钟时清零。
		0	ADC 忙且不能采样另一个通道
		1	ADC 可以采样下一个通道
12-8	ADCBSYCHN		在当前通道产生 ADC SOC 时置 1
			当 ADCBSY = 0 时：保持上一个转换通道值 n
		n	当 ADCBSY = 1 时，反映当前正被处理的通道值 n
			n=1xh 时无效值
7	ADCPWDN		ADC 掉电控制位（低电平有效），控制模拟内核内除带隙和基准电路之外
			的所有模拟电路的上电和掉电。
		0	模拟内核内除带隙和基准电路之外的所有模拟电路都掉电
		1	内核内的模拟电路上电
6	ADCBGPWD		带隙电路掉电控制位（低电平有效）
		0	带隙电路掉电
		1	内核内的带隙缓冲器电路上电
5	ADCREFPWD		参考缓冲器电路的掉电控制位（低电平有效）
		0	参考缓冲器电路掉电
		1	内核内的参考缓冲器电路上电

4	Reserved	0	读返回 0；对该位执行写操作无影响
3	ADCREFSEL		内部/外部参考电压选择位
		0	使用内部带隙产生参考电压
		1	使用外部 VREFHI/VREFLO 管脚产生参考电压。在某些器件上，VREFHI 管脚与 ADCINA0 管脚复用。此时，ADCINA0 在该模式下不能用于转换。而在另一些器件上，VREFLO 管脚与 VSSA 复用。此时 VREFLO 电压不能改变。
2	INTPULSEPOS		INT 脉冲产生控制位
		0	在 ADC 开始转换时产生 INT 脉冲（采样脉冲或采样信号的负边沿）
		1	在 ADC 结果锁存到结果寄存器之前的 1 个周期时产生 INT 脉冲
1	VREFLOCONV		VREFLO 转换控制位。使能时 VREFLO 在内部与 ADC 通道 B5 连接并将 ADCINB5 管脚从 ADC 断开。ADCINB5 管脚上的所有外部电路都不受该模式影响。
		0	VREFLO 不与 ADC 连接
		1	VREFLO 在内部与 ADC 连接，用于采样

0	TEMPCONV		Temperature sensor convert. When enabled internally connects the internal temperature sensor to ADC channel A5 and disconnects the ADCINA5 pin from the ADC. Whether the pin ADCINA5 exists on the device does not affect this function. Any external circuitry on the ADCINA5 pin is uneffected by this mode
		0	ADCINA5 is passed to the ADC module as normal, internal temperature sensor connection to ADCINA5 is disabled.
		1	Temperature sensor is internally connected to the ADC for sampling

(2) ADC中断寄存器

A：（包括4个寄存器）

中断标志寄存器（ADCINTFLG）、

中断标志清除寄存器（ADCINTFLGCLR）、

中断溢出寄存器（ADCINTOVF）

中断溢出清除寄存器（ADCINTOVFCLR），

这四个寄存器中第9到第15位保留，第0到第8位为分别对应ADCINT1到ADCINT9。

(2) ADC中断寄存器

中断标志寄存器（ADCINTFLG）：只读

位为0时表示没有产生ADC中断脉冲，

位为1时表示产生了ADC中断脉冲，如果ADC中断被置于连续模式（INTSELxNy寄存器），那么即使该标志位被设置，每当发生所选的EOC事件时也还会产生中断脉冲。如果没有使能连续模式，在用户使用ADCINTFLGCLR寄存器清除该标志位之前都将不会产生中断脉冲。

复位时ADCINTFLG默认值为0。

中断标志清除寄存器（ADCINTFLGCLR）：可读可写

位清0时无动作，位置1时清除ADCINTFLG寄存器中相应的标志位。

复位时ADCINTFLGCLR默认值为0。

(2) ADC中断寄存器

中断溢出寄存器（ADCINTOVF）：可读寄存器

ADC中断溢出状态位，指示在产生ADCINT脉冲时是否发生了溢出，如果相关的ADCINTFLG位被设置且额外又产生了EOC触发事件，那么就表示出现了溢出条件。

如果某位为0表示对应的ADCINT没有检测到ADC中断溢出事件，
某位为1表示对应的ADCINT检测到中断溢出位。
复位时ADCINTOVF默认值为0。

中断溢出寄存器清除（ADCINTOVFCLR）：可读可写

ADC中断溢出清除位，
清0时无动作，
置1时清除ADCINTOVF寄存器中的相应溢出位。
复位时ADCINTOVFCLR默认值为0。

B:中断选择寄存器(INTSELxNy)

有5个INTSELxNy寄存器（其中y=x+1，x=1,3,5,7,9），分别指明ADCINTn（n=1-9）的选择，这5个寄存器受EALLOW保护。
以INTSEL1N2寄存器为例说明如下：

15	14	13	12				8
Reserved	INT2CONT	INT2E	IINT2SEL				
R-0	R/W-0	R/W-0	R/W-0				
7	6	5	4				0
Reserved	INT1CONT	INT1E	IINT1SEL				
R-0	R/W-0	R/W-0	R/W-0				

图 10.6 中断 1、2 选择寄存器（INTSEL1N2）

INTxCONT：

清0表示只有在用户清除ADCINTx标志之后才能再产生ADCINTx脉冲。非连续

置1表示每当产生EOC脉冲时便产生ADCINTx脉冲，与ADCINTx是否被清除无关。连续

INTxE：

清0表示禁止ADCINTx，置1表示使能ADCINTx。

INTxSEL:

ADCINTx的EOC触发源选择位，当数值为0~15时表示选择EOC0~EOC15为ADCINTx的触发源，其他值无效。

(3) ADC SOC寄存器

A：ADC SOC0～SOC15控制寄存器（ADCSOCxCTL）
ADCSOCxCTL受EALLOW保护

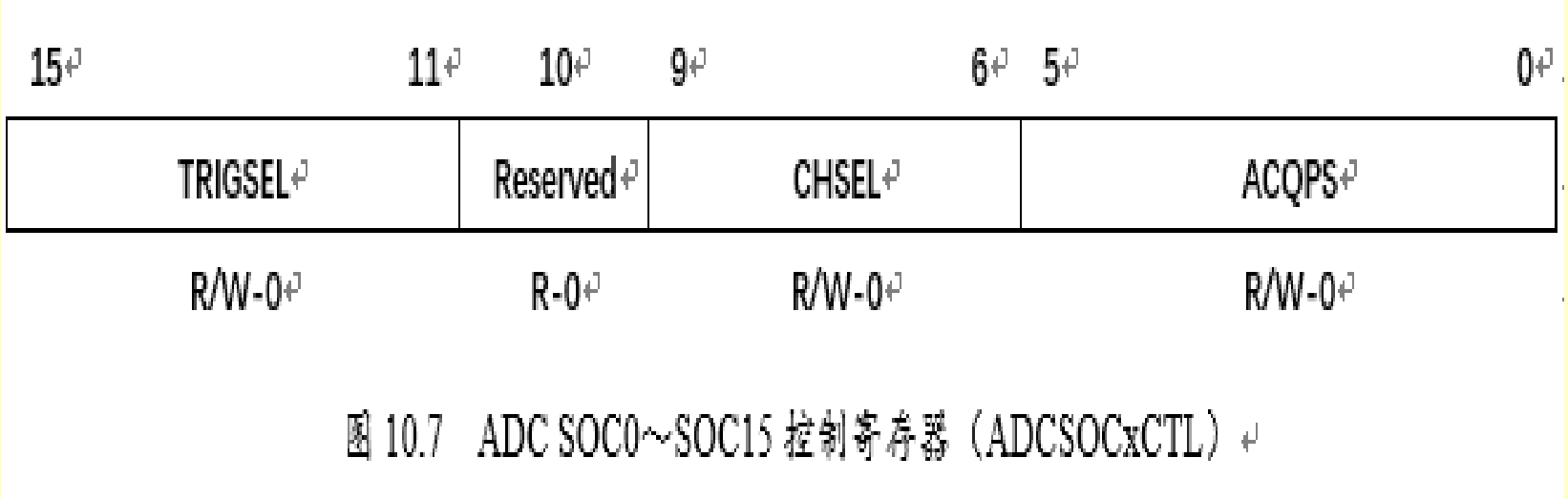


图 10.7 ADC SOC0~SOC15 控制寄存器（ADCSOCxCTL）

表 10.4 ADC SOC0～SOC15 控制寄存器（ADCSOCxCTL）字段描述

位	名称	值	描述
15-1	TRIGSEL		SOCx 触发源选择位。
1			配置由哪个触发源在 ADCSOCFLG1 寄存器中设置 SOCx 标志并在 SOCx 拥有优先权时启动转换。ADCINTSOCSEL1 或 ADCINTSOCSEL2 寄存器中相应的 SOCx 字段的值优先于 TRIGSEL 字段。
		00h	ADCTRIG0——仅由软件触发
		01h	ADCTRIG1——CPU 定时器 0.TINT0n
		02h	ADCTRIG2——CPU 定时器 1.TINT1n
		03h	ADCTRIG3——CPU 定时器 2.TINT2n
		04h	ADCTRIG4——XINT2.XINT2SOC
		05h	ADCTRIG5——ePWM1.ADCSOCA
		06h	ADCTRIG6——ePWM1.ADCSOCB
		07h	ADCTRIG7——ePWM2.ADCSOCA
		08h	ADCTRIG8——ePWM2.ADCSOCB
		09h	ADCTRIG9——ePWM3.ADCSOCA
		0Ah	ADCTRIG10——ePWM3.ADCSOCB
		0Bh	ADCTRIG11——ePWM4.ADCSOCA
		0Ch	ADCTRIG12——ePWM4.ADCSOCB
		0Dh	ADCTRIG13——ePWM5.ADCSOCA
		0Eh	ADCTRIG14——ePWM5.ADCSOCB
		0Fh	ADCTRIG15——ePWM6.ADCSOCA
		10h	ADCTRIG16——ePWM6.ADCSOCB
		11h	ADCTRIG17——ePWM7.ADCSOCA
		12h	ADCTRIG18——ePWM7.ADCSOCB
		其它值	无效选择

9-6	CHSEL		SOCx 通道选择位。用于在 ADC 接收到 SOCx 时选择将被转换的通道。
			顺序采样模式（SIMULENx = 0）：
		0h~7h	ADCINA0 ~ ADCINA7
		8h ~fh	ADCINB0 ~ ADCINB7
			同步采样模式（SIMULENx = 1）：
		0h~7h	ADCINA0/ADCINB0 对 ~ ADCINA7/ADCINB7 对
		8h~Fh	无效选择
5-0	ACQPS		SOCx 采集预分频控制位。该位用于控制 SOCx 其采样/保持窗口的长度，最小值可以为 6。
		00h~05h	无效选择
		06h~3Fh	采样窗口 7 个周期长（6+1）~ 64 个周期长（63+1）

(3) ADC SOC寄存器

B：ADC采样模式寄存器（ADCSAMPLEMODE）
受EALLOW保护

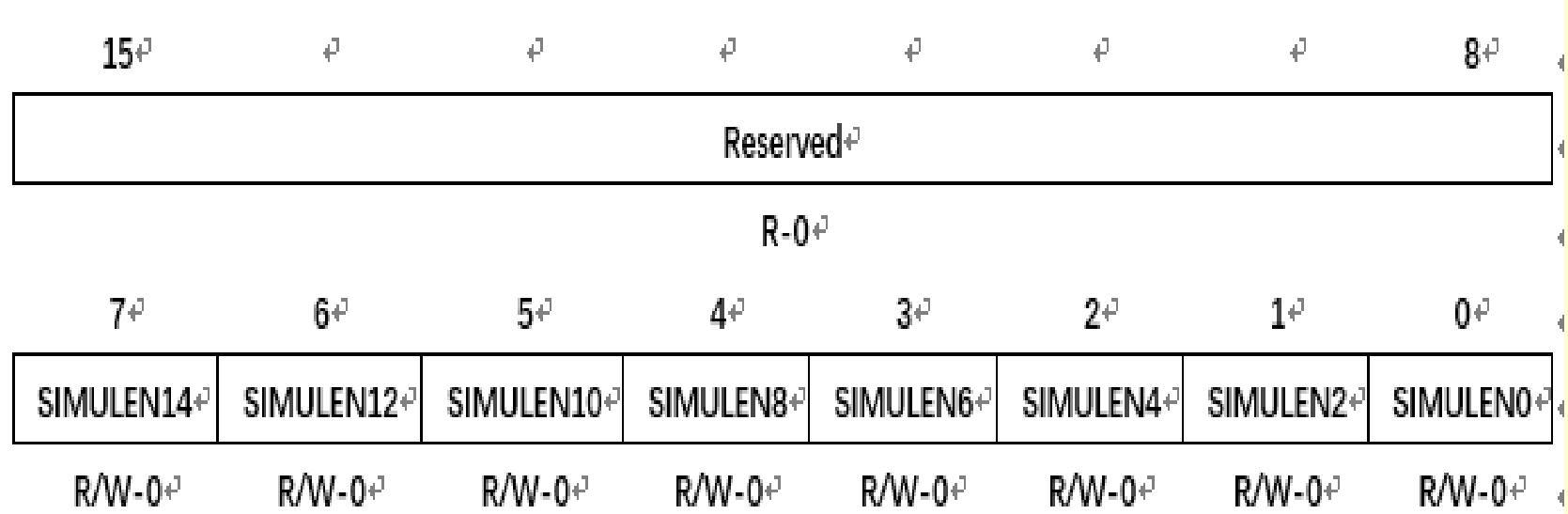


图 10.8 ADC 采样模式寄存器（ADCSAMPLEMODE）

B：ADC采样模式寄存器（ADCSAMPLEMODE）

SIMULENx位定义为：SOCx/SOCx+1的同步采样使能位。

同步采样模式下SOCx和SOCx+1连在一起，这个位不能在ADC转换SOCx或SOCx+1的时候置1。

该位清0表示将SOCx和SOCx+1设置成单采样模式，CHSEL字段的所有位决定要被转换的通道。

EOCx与SOCx对应，EOCx+1与SOCx+1对应，SOCx的结果存放在ADCRESULTx寄存器中，SOCx+1的结果存放在ADCRESULTx+1中。

该位置1表示将SOCx和SOCx+1设置成同步采样模式。CHSEL字段的低三位决定要被转换的一对通道。EOCx和EOCx+1这一对与SOCx和SOCx+1这一对对应，SOCx和SOCx+1的结果将分别存放在ADCRESULTx和ADCRESULTx+1寄存器中。

(3) ADC SOC寄存器

C：ADC中断SOC选择寄存器1和2（ADCINTSOCSEL1和ADCINTSOCSEL2），寄存器受EALLOW保护

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

图 10.9 ADC 中断触发 SOC 选择 1 寄存器（ADCINTSOCSEL1）（地址偏移量 14h）

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

图 10.10 ADC 中断触发 SOC 选择 2 寄存器（ADCINTSOCSEL2）

（3）ADC SOC寄存器

C：ADC中断SOC选择寄存器1和2（ADCINTSOCSEL1和ADCINTSOCSEL2），寄存器受EALLOW保护

字段描述：

SOCx 为ADC中断触发器选择位，选择由哪个ADCINT触发SOCx，这个字段优先于ADCSOCxCTL寄存器中的TRIGSEL字段。

SOCx=0，将没有ADCINT触发SOCx，由TRIGSEL字段决定SOCx的触发源。

SOCx=1，ADCINT1将触发SOCx，TRIGSEL字段被忽略。

SOCx=2，ADCINT2将触发SOCx，TRIGSEL字段被忽略。

SOCx=3，无效选择。

(3) ADC SOC寄存器

D: ADC SOC标志1寄存器（ADCSOCFLG1）、
SOC强制1寄存器（ADCSOCFRC1）、
SOC溢出1寄存器（ADCSOCOVF1）
SOC溢出清零1寄存器（ADCSOCOVFCLR1）

这四个寄存器的第0位~第15位分别为SOC0~SOC15，对应于SOC0~SOC15

(3) ADC SOC寄存器

ADCSOCFLG1：只读寄存器，

SOCx开始转换标志位，用于指示单个SOC转换状态。“0”表示SOCx采样没挂起。“1”表示已经收到触发事件且SOCx采样被挂起；当对应的SOCx转换开始后，这个位会自动清除。

该寄存器复位默认值为0。

ADCSOCFRC1：可读可写寄存器，

SOCx强制开始转换标志位。

向该位写1将会强制向ADCSOCFLG1寄存器中的SOCx标志位写1，这样便可以进行一次由软件启动的转换。

写0无影响。该寄存器复位默认值为0。

(3) ADC SOC寄存器

ADCSOCOVF1：只读寄存器，

SOCx开始转换溢出标志位，用于指示在目前SOCx事件挂起期间是否产生了SOCx事件。

“0”表示无SOCx事件溢出，

“1”表示SOCx事件溢出，溢出条件并不会阻止处理SOCx事件，它只是指示某个触发事件被错过了。

该寄存器复位默认值为0。

ADCSOCOVFCLR1：可读可写寄存器，

SOCx开始转换溢出标志清除位。

向该位写1会清除ADCSOCOVF1寄存器中相应的SOCx溢出标志。

向该位写0无影响。

该寄存器复位默认值为0。

(4) ADC其他控制寄存器

A：ADC优先级寄存器

ADC SOC优先级控制寄存器（SOCPRICTL）受EALLOW保护。RRPOINTER指明轮询状态，SOCPRIORITY设置高优先级。

15	11	10	5	4	0
Reserved		RRPOINTER		SOCPRIORITY	
R-0		R-20h		R/W-0	

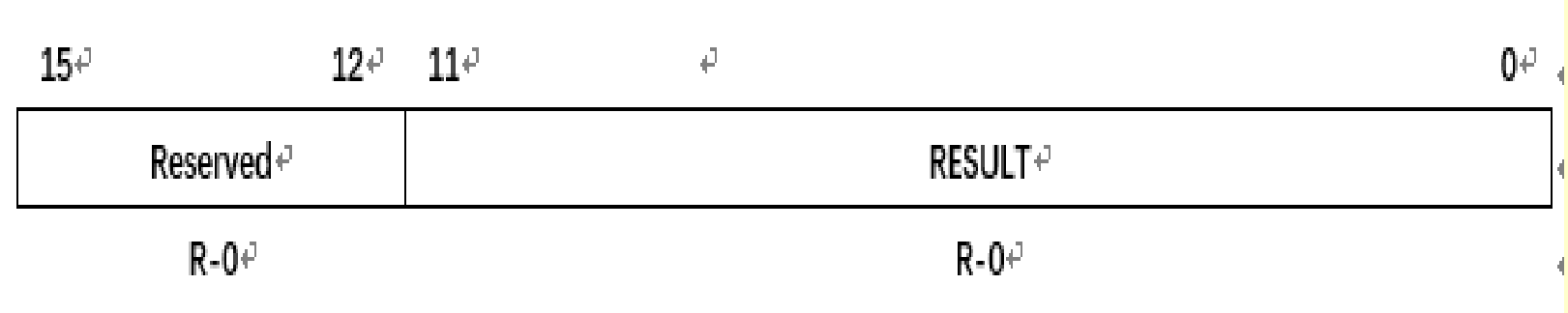
图 10.11 ADC SOC 优先级控制寄存器（SOCPRICTL）

B：ADC校准寄存器

C：ADC校正寄存器（ADCREV）

(5) ADC结果寄存器

ADC结果寄存器位于外设帧0（PF0）



(6) ADC头文件中寄存器结构体说明

头文件中包括2个寄存器结构体：
控制寄存器结构体ADC_REGS和结果寄存器结构体ADC_RESULT_REGS。
寄存器结构体中的位定义与前面介绍的寄存器位定义相同。

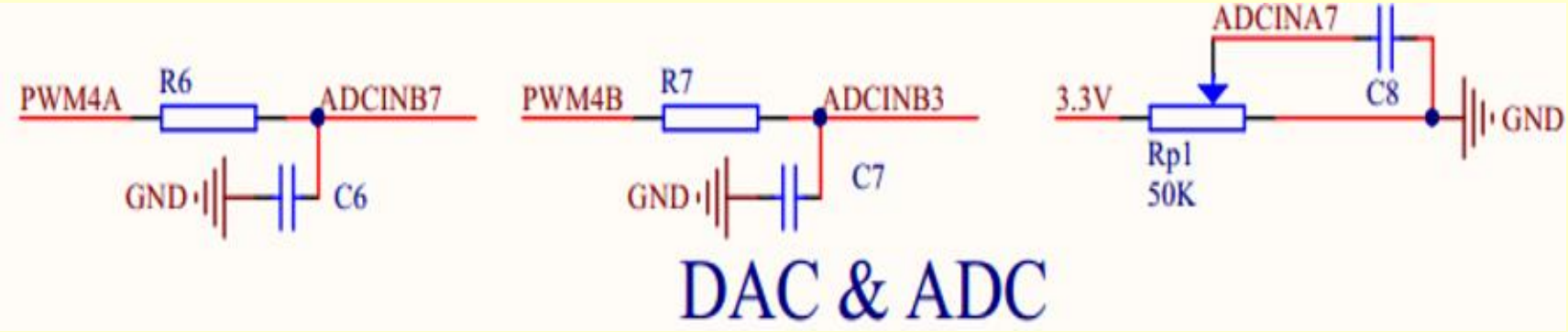
10.5 ADC应用举例

对图4.7中的几路模拟量进行采样

要求进行同步采样和过采样，

采样由PWM1A的下溢（过零点）触发，

中断读取采样结果



1）ADC初始化程序

```
void InitADC()
{
  int DCSampT;
  DCSampT=15;
  EALLOW;
  SysCtrlRegs.PCLKCR0.bit.ADCENCLK = 1;
  (*Device_cal)();
  EDIS;
  DELAY_US(ADC_usDELAY);
  EALLOW;
  AdcRegs.ADCCTL1.bit.ADCBGPWD  = 1;// 内核内的带隙缓冲器电路上电
  AdcRegs.ADCCTL1.bit.ADCREFPWD = 1;//内核内的参考缓冲器电路上电
  AdcRegs.ADCCTL1.bit.ADCPWDN   = 1; // 内核内的模拟电路上电
  AdcRegs.ADCCTL1.bit.ADCENABLE = 1;   // ADC使能
  AdcRegs.ADCCTL1.bit.ADCREFSEL = 1;
                                          // 使用外部VREFHI/VREFLO参考电压
  //AdcRegs.ADCCTL1.bit.ADCREFSEL = 0;// 使用内部带隙产生参考电压
  EDIS;
```

```
DELAY_US(ADC_usDELAY);  // 转换ADC通道前延迟
  AdcOffsetSelfCal();
  DELAY_US(ADC_usDELAY);  // 转换ADC通道前延迟
  EALLOW;
  AdcRegs.ADCCTL1.bit.INTPULSEPOS= 1;
                                        //转换完成前一个ADC时钟周期产生EOC
  AdcRegs.INTSEL1N2.bit.INT1E     = 1;     // ADCINT1使能
  AdcRegs.INTSEL1N2.bit.INT1CONT  = 0;  //禁用ADCINT 1连续模式
  AdcRegs.INTSEL1N2.bit.INT1SEL= 0x0f;
                                          //设置EOC 15以触发ADCINT 1启动
  AdcRegs.ADCSAMPLEMODE.all = 0xff;  //SOCAx和SOCBx同步采样
  //对ADCINA7、ADCINB7过采样
  AdcRegs.ADCSOC0CTL.bit.CHSEL = 0x07;  //A7,Result0;  B7,Result1
  AdcRegs.ADCSOC2CTL.bit.CHSEL = 0x07;  //A7,Result2;  B7,Result3
  AdcRegs.ADCSOC4CTL.bit.CHSEL = 0x07;  //A7,Result4;  B7,Result5
  AdcRegs.ADCSOC6CTL.bit.CHSEL = 0x07;  //A7,Result6;  B7,Result7
  //对ADCINA3、ADCINB3过采样
```

```
AdcRegs.ADCSOC8CTL.bit.CHSEL = 0x03;  //A3,Result8;  B3,Result9
AdcRegs.ADCSOC10CTL.bit.CHSEL = 0x03;  //A3,Result10; B3Result11
AdcRegs.ADCSOC12CTL.bit.CHSEL = 0x03;  //A3,Result12; B3,Result13
AdcRegs.ADCSOC14CTL.bit.CHSEL = 0x03;  //A3,Result14; B3,Result15
AdcRegs.ADCSOC0CTL.bit.TRIGSEL= 5;
                  //ADCTRIG5 – ePWM1, ADCSOCA
AdcRegs.ADCSOC2CTL.bit.TRIGSEL= 5;
AdcRegs.ADCSOC4CTL.bit.TRIGSEL= 5;
AdcRegs.ADCSOC6CTL.bit.TRIGSEL= 5;
AdcRegs.ADCSOC8CTL.bit.TRIGSEL= 5;
AdcRegs.ADCSOC10CTL.bit.TRIGSEL= 5;
AdcRegs.ADCSOC12CTL.bit.TRIGSEL= 5;
AdcRegs.ADCSOC14CTL.bit.TRIGSEL= 5;
```

```
  AdcRegs.ADCSOC0CTL.bit.ACQPS= ADCSampT;
                                        //设置SOC0~1采样窗口的长度
  AdcRegs.ADCSOC2CTL.bit.ACQPS= ADCSampT;
  AdcRegs.ADCSOC4CTL.bit.ACQPS= ADCSampT;
  AdcRegs.ADCSOC6CTL.bit.ACQPS= ADCSampT;
  AdcRegs.ADCSOC8CTL.bit.ACQPS= ADCSampT;
  AdcRegs.ADCSOC10CTL.bit.ACQPS = ADCSampT;
  AdcRegs.ADCSOC12CTL.bit.ACQPS = ADCSampT;
  AdcRegs.ADCSOC14CTL.bit.ACQPS = ADCSampT;
  EDIS;
  EPwm1Regs.ETSEL.bit.SOCAEN= 1;// 使能EPWM1SOCA
  EPwm1Regs.ETSEL.bit.SOCASEL= 1;//在CTR = ZERO时使能事件
  EPwm1Regs.ETPS.bit.SOCAPRD = 1;// 在第一个事件产生脉冲
  EPwm1Regs.ETCLR.bit.SOCA = 1;      // 清除SOCA标志
}
```

2）ADC中断服务程序

```
interrupt void ADC1Int_isr(void)
{
    ADCA7 = AdcRegs.ADCRESULT0+AdcRegs.ADCRESULT2;
    ADCA7 += AdcRegs.ADCRESULT4+AdcRegs.ADCRESULT6;
    ADCA3 = AdcRegs.ADCRESULT6+AdcRegs.ADCRESULT10;
    ADCA3 += AdcRegs.ADCRESULT12+AdcRegs.ADCRESULT14;
    ADCB7 = AdcRegs.ADCRESULT1+AdcRegs.ADCRESULT3;
    ADCB7 += AdcRegs.ADCRESULT5+AdcRegs.ADCRESULT7;
    ADCB3  = AdcRegs.ADCRESULT9+AdcRegs.ADCRESULT11;
    ADCB3 += AdcRegs.ADCRESULT13+AdcRegs.ADCRESULT15;

    AdcRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;
    PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;
    EINT;
    }
```

程序中，ADCA7、ADCA3、ADCB7和ADCB3分别对应于通道ADCINA7、ADCINA3、ADCINB7和ADCINB3转换的结果。