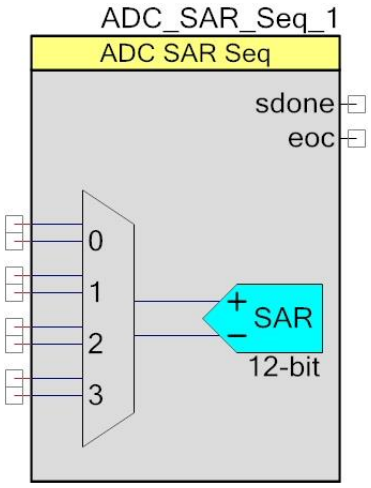
PSoC 4 序列逐次逼近 ADC

1.0

特性



- 可选 8、10 和 12 位分辨率
- 采用 12 位分辨率时，采样率最高 1 Msps
- 支持 Single Ended（单端）和 Differential（差分）输入
- 多输入范围和多参考电压选项
- 自动扫描最多 8 个通道，支持单通道输入
- 支持软件使能“插入” 通道
- 支持硬件求平均

概述

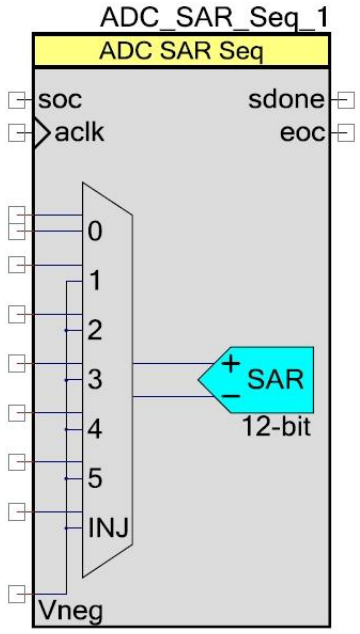
序列 SAR ADC 使您能够在 PSoC 4 上配置和使用不同操作模式的 SAR ADC。您拥有原理图和固件级别的支持，在 PSoC Creator 中可以方便的使用序列 SAR ADC。您还可以配置最多 8 个模拟通道，它们可自动扫描，每个通道的扫描结果被存储到独立的结果寄存器。可选“插入”通道可以通过软件使能，偶尔扫描某个不需要与其他通道同速率扫描的信号。

何时使用 SAR ADC

序列 SAR ADC 是用于使用 PSoC 4 中 ADC 功能的组件。序列控制器和多路复用选择开关是序列 SAR ADC 的硬件组成部分。该组件可在无 CPU 参与的情况下以高速的采样率自动的实现多通道数据采样。它同样可以用于低速率或单通道数据采样。

输入/输出连接

本节介绍序列 SAR ADC 的各种输入和输出接口。I/O 列表中的星号 (*) 表示 I/O 可能在某种 SAR ADC 配置下被隐藏。



+输入 – 模拟

此输入是序列 SAR ADC 的正向模拟输入。SAR ADC 的转换结果与 +输入信号和参考电压的差相关。参考电压可以是 –输入信号、Vneg（外部参考电压）、Vref （内部参考电压）或 Vss。

组件显示的 +输入模拟端口的数目始终与组件中使能的模拟通道的数目相同，包括插入通道。

–输入 – 模拟 *

组件显示的 – 输入模拟端口的数量不仅取决于组件中使能的模拟通道的数目，还是能通道有多少被配置成单端输入有关。当通道被配置为单端输入后，所有 –输入在芯片内部连接在一起，合并成一个单一网络。

Vneg – 输入 *

这是一个通用参考电压输入端。此端口仅在一个或多个**模拟通道被设置为单端输入且单端负向输入**参数被设置为 **External**（外部）时出现。

soc – 输入 *

ADC 转换的硬件触发端口。如果您选择了**硬件触发**采样模式，此输入可见。此输入的上升沿触发 ADC 转换。SAR ADC 组件启动后，参考电压和电荷泵电压的稳定需要至少 10uS 的时间，第一个 soc 触发应该在参考电压和电压泵电压稳定之后。所以首个 **soc** 触发应该在组件启动 10 uS 以后发生，以保证转换精度。。如果将 **Sample Mode**（采样模式 ）设置为 **Free Running**（持续运行），此端口被隐藏。

aclk – 输入 *

ADC 模拟时钟输入端口。如果将 **Clock Source**（时钟资源）设置为 **External**（外部），此端口可见，否则会被隐藏。此时钟影响 SAR ADC 的转换速率。

sdone（sample done，采样结束） – 输出

当 SAR ADC 已完成对当前通道的采样操作时，该端口输出一个高电平脉冲，宽度为一个 SAR Clock 时钟周期。此时多路模拟选择开关切换到下一通道。多路模拟选择开关在当前通道采样完成时切换到下一通道，此时该通道的 ADC 转换过程可能还没有结束。

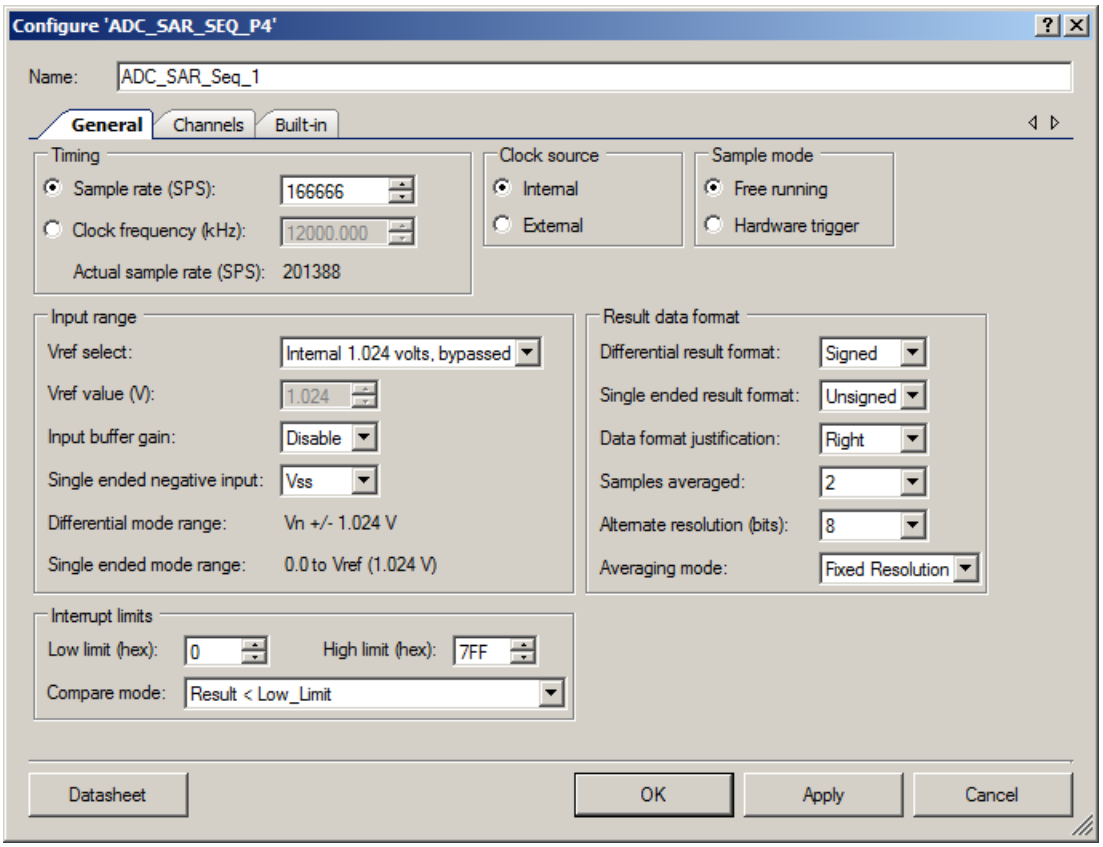
eoc – 输出

转换结束 (eoc) 端口输出上升沿表示一个转换周期已经完成。此时，所有被使能通道的转换结果准备就绪，可从寄存器读取。该事件可触发组件内部中断。用户也可以将中断信号引出，触发自定义中断或控制外部逻辑电路。

组件参数

将一个序列 SAR ADC 拖放到您的设计上，并双击以打开“ Configure”（配置）对话框。**Error! eference source not found.** 显示如下 General（常规）对话框。

常规选项卡



序列 SAR ADC 有以下参数：

采样率

采样率选定后，组件会根据被使能通道的数量和通道配置来自动计算/设置组件的时钟频率，以符合采样率的要求。此字段下的只读字段显示的是根据时钟编辑器（Clock Editor）实际提供时钟频率计算出的实际采样率。实际采样率与设置采样率可能不同，这是因为时钟分频器不能分频出任意频率的时钟。

时钟频率

选择该选项后，可以输入 SARADC 的模拟时钟频率。此参数仅在 “Clock Source(时钟源)" 选择 "Internal (内部)" 选项时可选。时钟频率可设置为 1 MHz 到 18 MHz（CY8C4100 系列 中为 14.508 MHz）之间的任何频率。如果设置的时钟频率不在此范围内， PSoC Creator 工程无法通过编译。实际的时钟频率与设置的时钟频率可能不同，这是因为时钟分频器不能分频出任意的频率的时钟。此字段下的只读字段显示的是根据时钟编辑器实际提供的时钟频率计算出的实际采样率。

在高采样速率下，ADC 可生成大量数据。此时需要将 CPU 时钟设置的足够快， 来保证 CPU 可以处理这些数据，同时需要最大程度减少中断服务的开销。例如，对于每秒钟 700,000 个采样样本的转换速率和 48 MHz 的 CPU 时钟速率， 每个样本 CPU 仅有 48 MHz/700,000 sps = 68 个时钟周期来处理。有关 ISR 的优化（使用）指南，请参见“中断服务例程”章节。

时钟源

此参数允许您选择一个组件内部的时钟或组件外部的时钟源。

采样模式

采样模式设置 SAR ADC 是否每次扫描结束后必须由 SOC 端口触发启动后持续运行，直至 ADC_StopConvert() API 被调用。

Sample Mode （采样模式）	说明
持续运行	序列 ADC SAR 持续运行。
硬件触发	SOC 引脚上的一个上升沿脉冲触发一次 ADC 转换。 ADC_StartConvert() 也可以触发一次 ADC 转换。

Vref 选择

Vref Select 参数用于选择 SAR ADC 的参考电压。

参考选项	说明
VDDA/2VDDA 内部 1.024V	使用无旁路电容的内部参考。对于 VDDA/2 和内部 1.024V， SAR ADC 允许的最高工作频率是 3 Mhz。如果想使用更高的工作频率，必须外接旁路电容。
内部 1.024V，旁路 VDDA/2，旁路	使用带旁路电容的内部参考。必须在引脚 P1[7] 上连接一个旁路电容*。
外部 Vref	使用引脚 P1[7] 上的外部参考电压。
内部 Vref 内部 Vref，旁路	当前 PSoC 4 芯片不支持这些选项。

* 如果内部数字开关引发的噪音超过应用对模拟性能的要求，建议使用外部旁路电容。如需使用此选项，推荐使用连接一个介于 0.01 µF 和 10 µF 之间的电容到 P1[7]。

内部 1.024 V Vref 的启动时间因旁路电容值的大小而不同。此表列出了两个最常见旁路电容及对应的内部 1.024V Vref 的启动时间。

内部 Vref 启动时间	最大时间规定
使用 1uF 外部旁路电容时内部参考 Vref 的启动时间	2 ms
使用 100nF 外部旁路电容器内部参考 Vref 的启动时间	200 µs

Vref 值

此参数显示用于 SAR ADC 的参考电压值。如果选择内部参考 **Vref select** 参数，则此值变为固定值。如果已选定一个内部参考，例如 VDDA 或 VDDA/2，则该值派生自 DWR 窗口中的系统参数 Vdda 的设置值。在 Vref 未知的情况下，例如使用外部参考（PSoC 或组件外部），值可以由用户输入。

输入缓冲器增益

此参数确定将输入信号放大并缓冲到 SAR ADC。当前的 PSoC 4 芯片不支持此选项。

单端负向输入

如果任何通道被配置为单端输入， 此参数选择 SAR ADC 负向输入端连接的电平。此选择影响输入电压范围和有效分辨率。无论输入范围如何设置，连接至 IC 的模拟信号的电压都必须在 Vssa 和 Vdda 之间。

负向输入	说明
Vss	如果输入范围是 0.0 到 Vref，那么有效分辨率将比设定分辨率少一位。
Vref	输入范围为 0.0 至 Vref*2。使用内部参考 (1.024 V) 时，ADC 的可用输入电压为 0.0 至 2.048 V。
External（外部）	此模式针对差分输入配置。将通用单端负向输入连接至 Vneg 端口。当使用外部参考 (1.024 V) 时，输入范围为 Vneg ± 1.024 V。 例如，如果将 Vneg 连接至 2.048 V，可用的输入范围为 2.048 ± 1.024 V 。对于单端和差分信号均扫描的系统，当扫描一个单端输入通道时，连接 Vneg 至 Vssa。 您可以使用外部参考源提供更宽的操作范围。您可以使用公式 Vneg ± Vref 计算所有通道的输入范围。

差分模式输入范围

这是非可编辑文本框，显示差分模式输入范围。它基于 **Vref 选择**和 **Vref 值**参数。此文本框的范例为（Vn +/- 1.024 V、Vn +/- Vdda/2、 Vn +/- Vdda 等）

单端模式输入范围

这是非可编辑文本框，显示单端模式输入范围。它基于 **Vref 选择**、**Vref 值**和**单端负向输入**参数。此文本框的范例为（0.0 至 Vref (1.024V)、0.0 至 Vref (2.048 V)、0.0 至 Vref (5 V) 等）

差分测量数据格式

此参数确定来自差分测量的结果是**有符号**还是无符号。这是针对所有差分通道的全局设置。

单端测量数据格式

此参数确定来自单端测量的结果是**有符号**还是无符号。这是针对所有单端通道的全局设置。

下方表格显示这些参数如何影响输入电压到 12 位数字采样值的转换。

单端/差分	有符号/无符号	单端负向输入	-Input（-输入）	+Input（+输入）	结果寄存器
单端	无	Vss	Vss	Vref Vss 负噪声	0x07FF 0x0000 0xFFxx
单端	无	External（外部）	Vneg	Vneg+Vref Vneg Vneg-Vref	0x07FF 0x0000 0xF800
单端	无符号	Vref	Vref	2*Vref Vref Vss	0x0FFF 0x0800 0x0000
单端	有符号	Vref	Vref	2*Vref Vref Vss	0x07FF 0x0000 0xF800
差分	无符号	无	Vx	Vx+Vref Vx Vx-Vref	0x0FFF 0x0800 0x0000
差分	有符号	无	Vx	Vx+Vref Vx Vx-Vref	0x07FF 0x0000 0xF800

请注意，对于**单端转换，若单端负向输入**设置为 **Vss**， 则转换有效位数仅为 11 位，因为低于 Vss 的电压在 PSoC 4 引脚上为无效输入。对这种设置，全局配置位**单端测量数据格式**无效。如果 **+Input**（+输入） 引脚上的噪音电平稍微低于 Vss，会产生负向结果。

请注意，在单端模式下如果负向输入端连接到某个外部参考，这和差分模式下将负向输入端连接到此外部参考是等效的。如果 +input 的电压值无法降低到该参考值一下，那么 ADC 的有效分辨率也是 11 位。数据存储格式

此参数设置转换结果在 16 位字符中存储的对齐方式为 **Left**（左）还是 **Right**（右）。对于有符号的值，在右对齐模式下，结果扩展为有符号。这是针对所有差分通道的全局设置。此表显示了所有详情。

调整	有符号/无符号	分辨率	结果寄存器															
			15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
右	无符号	12 10 8	- - -	- - -	- - -	- - -	11 - -	10 - -	9 9 -	8 8 -	7 7 7	6 6 6	5 5 5	4 4 4	3 3 3	2 2 2	1 1 1	0 0 0
右	有符号	12 10 8	11 9 7	11 9 7	11 9 7	11 9 7	11 9 7	10 9 7	9 9 7	8 8 7	7 7 7	6 6 6	5 5 5	4 4 4	3 3 3	2 2 2	1 1 1	0 0 0
左	无	12 10 8	11 9 7	10 8 6	9 7 5	8 6 4	7 5 3	6 4 2	5 3 1	4 2 0	3 1 -	2 0 -	1 - -	0 - -	- - -	- - -	- - -	- - -

采样值求平均

此参数设置 “平均”功能的平均数该参数应用于使能 “平均” 功能的所有通道。

次分辨率

此参数设置 ADC 次分辨率为 8 或 10 位。每个输入转换可选择主分辨率（12 位）或次分辨率。

平均模式

此参数设置平均值的计算方法。如果选择累加选项，每个 ADC 结果累加求和，直至该和试图超过一个 16 位值，在该点截断。如果选择**固定分辨率**模式，LSb 会被截断，这样值就无法增加超出给定分辨率的最大值。

Compare Mode（阀值比较模式）

序列 SAR ADC 支持转换结果范围检测，允许在 CPU 不参与的情况下， 将转换结果与两个可编程阈值自动对比。检测范围由两个全局阈值和一个条件决定。

此参数设置阀值比较条件，如果转换结果不符合条件则可触发一个可屏蔽检测中断。

阀值比较条件	说明
结果 < 下限	在范围以下
下限 <= 结果 < 上限	在范围内
上限 <= 结果	在范围之上
（结果 < 下限）或（上限 <= 结果）	超出范围

下限

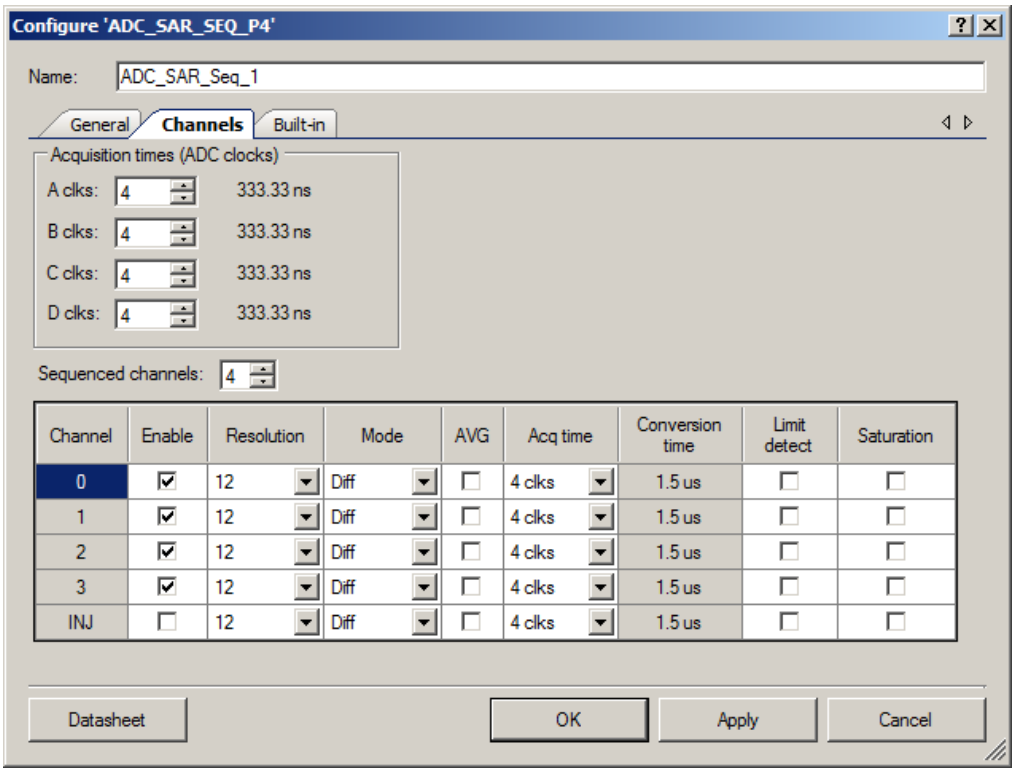
此参数设置阈值下限。

上限

此参数设置阈值上限。

阀值检测在取平均值、调整和符号扩展后进行（如使能）。换言之，阈值的数据格式必须与 ADC 转换结果的格式相同。

通道选项卡



采样时间

每个通道可从四种采样时间中任意选其一。采样时间以 SAR ADC 的时钟周期为单位设置/显示。每个选项右侧的显示字段显示实际需要的采样时间。如果 SAR ADC 的工作时钟被更改，显示的时间也更改。

顺序通道

此参数选择使能多少输入通道，不计插入通道。通道最大数量为 4 或 8，具体取决于输入通道的模式配置（差分或单端）。最小通道数始终为 1。

每个通道有几个可配置参数。实际通道数量取决于**顺序通道**参数。插入通道 **INJ** 参数始终存在。如果插入通道未启用，则不在符号上显示。符号显示**顺序通道**参数选择的所有通道如果该通道被使能，插入通道除外。

使能

对于通道 0 至 7，它在运行期间会使能通道扫描。对于插入通道，它确定输入是都显示在组件上。

分辨率

此参数选择通道分辨率，可以选择 12 位主分辨率或 10 位/8 位次 (ALT)分辨率， 这具体取决于 **General**（常规）选项卡中的**次分辨率的设置值。**

模式

此参数选择该通道 ADC 的转换模式为**差分**还是单端模式。

AVG

此选项选择是否对通道转换结果求平均。使能后，SAR 序列控制器会对该通道连续采样 N 次，并将结果相加求平均。采样数量由 **General**（常规）选项卡中的设定的平均值 N 决定。注意：求平均时总是使用 12 位分辨率和右对齐。

采样时间

此参数选择采样时间 （采样加保持的时间），在此时间内 SAR 输入保持。时间基于 SAR ADC 的时钟周期。默认 为 **4** 个时钟周期，但可选择另外四种采样时间。这些**采样时间** 参数被标为 A、B、C 和 D，可在 4 至 1027 个时钟周期内调整。

阀值检测

此选项允许触发阀值检测中断， 前提是通道 0 至 7 或插入通道触发阀值**下限**或阀值**上限**以及 **General**（常规）选项卡中**阀值比较模式**参数设置的比较条件。

饱和检测

如果某个通道的采样值等于相应分辨率下转化结果的下限值或上限值时，使能该选项可触发饱和中断。例如 12 位分辨率下得到的转换结果为 0x000 或者 0xFFF。

应用程 序编程接口

应用程序编程接口 (API) 子程序允许您使用软件配置组件。此表列出了每个函数的接口，并进行了说明。以下各节将更详细地介绍每个函数。

默认情况下，PSoC Creator 将实例名称 “ADC_SAR_Seq_1” 分配给指定设计中第一个实例组件。您可以将其重命名为遵循标识符语法规则的任何唯一值。实例名称会成为每个全局函数名称、变量和常量符号的前缀。为增加可读性，下表中使用了实例名称“ADC”。

函数

函数	说明
ADC_Start()	执行此组件需要的所有初始化并开始为模块供电。
ADC_Stop()	此函数停止 ADC 转换并将 ADC 置于最低功耗模式。
ADC_StartConvert()	对于持续运行模式，此 API 开启转换过程并持续运行。在单次触发模式中，调用此 API 触发一次转换。
ADC_StopConvert()	强制 ADC 停止转换。如果有转换正在执行，则该转换完成后停止，进一步的转换将不会发生。
ADC_IRQ_Enable()	当前转换结束后，使能中断。还必须使能全局中断，以实现 ADC 中断。
ADC_IRQ_Disable()	当前转换结束后，禁用中断。
ADC_IsEndConversion()	立即返回当前转换状态或在转换完成后返回（阻塞），具体取决于 retMode 参数。
ADC_GetResult16()	在 SAR 结果寄存器中获得转换结果。
ADC_SetChanMask()	设置通道使能标志位。设置要扫描的通道。
ADC_EnableInjection()	仅针对下一次扫描使能插入通道。
ADC_SetLowLimit()	此参数设置对比阀值的下限值。
ADC_SetHighLimit()	此参数设置对比阀值的上限值。
ADC_SetLimitMask()	设置哪些通道可以触发阀值检测中断。
ADC_SetSatMask()	设置哪些通道可以触发饱和检测中断。
ADC_SetOffset()	设置 ADC 通道的偏移量。
ADC_SetGain()	设置 ADC 通道每 10V 电压对应的增益计数。

函数	说明
ADC_CountsTo_Volts()	将 ADC 输出转换为单位为 V 的浮点电压值。
ADC_CountsTo_mVolts()	将 ADC 输出转换为单位为毫伏的电压值。
ADC_CountsTo_uVolts()	将 ADC 输出转换为单位为微伏的电压值。
ADC_Sleep()	停止 ADC 运行，并保存配置寄存器和组件的使能状态。
ADC_Wakeup()	恢复组件使能状态和配置寄存器。
ADC_SaveConfig()	保存 ADC 非保持寄存器的当前状态。
ADC_RestoreConfig()	恢复 ADC 非保持寄存器当前的配置。

全局变量

函数	说明
ADC_initVar	initVar 变量用于标记此组件的初始配置情况。该变量初始化为 0，并在第一次调用 ADC_Start() 时设置为 1。这允许进行组件初始化以后，无需在再次调用ADC_Start() 时再次进行模块初始化。 如需重新初始化组件，可在 ADC_Start() 或 ADC_Enable() 函数前调用 ADC_Init() 函数。
ADC_offset[]	此数组校准每个通道的偏移量。ADC_Start() 首次调用时设置为 0，并可使用 ADC_SetOffset() 修改。通过减去给定偏移量，此数组可影响 ADC_CountsTo_Volts()、ADC_CountsTo_mVolts() 和 ADC_CountsTo_uVolts() 函数的返回值。
ADC_countsPer10Volt[]	此数组用于校准每个通道的增益。它在首次调用 ADC_Start() 时计算。值取决于通道分辨率和电压参考。它可以使用 ADC_SetGain() 更改。 通过在 ADC 计数和输入电压之间设定正确的转换值，此数组影响 ADC_CountsTo_Volts、ADC_CountsTo_mVolts 和 ADC_CountsTo_uVolts 函数的返回值。

可用常量

函数	说明
ADC_SEQUENCED_CHANNELS_NUM	此常量代表可用于扫描的输入通道数量。
ADC_TOTAL_CHANNELS_NUM	此常量代表包括插入通道在内的输入通道总数。

void ADC_ Start(void)

说明： 执行此组件需要的所有初始化并开始为模块供电。功耗将基于时钟频率设置为合适功耗。

参数： 无

返回值： 无

副作用： 无

void ADC_Stop(void)

说明： 此函数停止 ADC 转换并将 ADC 置于最低功耗模式。

参数： 无

返回值： 无

副作用： 无

void ADC_StartConvert(void)

说明： 对于持续运行模式，此 API 开始转换过程并持续运行。

在**硬件触发**模式，该函数也作为 SOC 的软件版本，每次转换均需调用此函数。

参数： 无

返回值： 无

副作用： 无

void ADC_StopConvert(void)

说明： 强制 ADC 停止转换。如果有转换正在执行，则该转换完成后停止，进一步的转换将不会发生。。

参数： 无

返回值： 无

副作用： 无

void ADC_IRQ_Enable(void)

说明： 当前转换结束后，使能中断。还必须使能全局中断，以实现 ADC 中断。。

参数： 无

返回值： 无

副作用： 无

void ADC_IRQ_Disable(void)

说明： 当前转换结束后，禁用中断。。

参数： 无

返回值： 无

副作用： 无

uint32 ADC_IsEndConversion(uint32 retMode)

说明： 立即返回当前转换状态或在转换完成后返回（阻塞），具体取决于 retMode 参数。。

参数： retMode：检查转换返回模式。有关选项，请参见下表。

选项	说明
ADC_RETURN_STATUS	立即返回连续通道转换状态。如果返回值为零，则转换未完成，应重试此函数，直至返回非零值。
ADC_WAIT_FOR_RESULT	直到所有连续通道的 ADC 转换完成后才返回值。
ADC_RETURN_STATUS_INJ	立即返回插入通道转换状态. 如果返回值为零，则转换未完成，应重试此函数，直至返回非零值。
ADC_WAIT_FOR_RESULT_INJ	ADC 完成插入通道转换后才会返回结果。

返回值： uint 8：如果返回非零值，则最后一次转换已完成。如果返回值为零，则 ADC 仍在计算最后的结果。

副作用： 此函数读取并在随后清除结束转换状态。

int16 ADC_GetResult16(uint32 chan)

说明： 获取通道结果寄存器中可用的数据。

参数： chan：ADC 通道编号。第一个通道为 0，插入通道如启用，是有效通道的数量。

返回值： 返回转换数据为有符号的16 位整数值

副作用： 无

void ADC_SetChanMask(uint32 mask)

说明： 设置通道使能标记。

参数： mask：设置要扫描的通道。标记不存在的通道数没有副作用例如，如果仅启用 6 个通道，设置一个 0x0103 标记仅会启用最后两个通道（0 和 1）。

返回值： 无

副作用： 无

void ADC_EnableInjection(void)

说明： 仅针对下一次扫描使能插入通道。

参数： 无

返回值： 无

副作用： 无

void ADC_SetLowLimit(uint32 lowLimit)

说明： 设置阀值的下限值。

参数： lowLimit：阀值的下限。

返回值： 无

副作用： 无

void ADC_SetHighLimit(uint32 highLimit)

说明： 设置阀值的上限值

参数： highLimit: 阀值上限

返回值： 无

副作用： 无

void ADC_SetLimitMask(uint32 mask)

说明： 设置哪些通道可以触发阀值检测中断。。

参数： mask：设置可以引发阀值监测中断的通道。设置不存在的通道数没有副作用。例如，如果仅启用 6 个通道，设置一个 0x0103标记仅会启用最后两个通道（0 和 1）。

返回值： 无

副作用： 无

void ADC_SetSatMask(uint32 mask)

说明： 设置哪些通道可以触发饱和检测中断。。

参数： mask：设置哪些通道可以引发饱和检测中断。设置不存在的通道数没有副作用。例如，如果仅启用 8 个通道，设置一个 0x01C0 标记仅会启用最后两个通道（6 和 7）。

返回值： 无

副作用： 无

void ADC_SetOffset(uint32 chan, int16 offset)

说明： 设置函数 ADC_CountsTo_uVolts、ADC_CountsTo_mVolts 和 ADC_CountsTo_Volts 所用的 ADC 偏移量，以便在计算电压转换前，从给定读数中减去该偏移量。

参数： chan：ADC 通道数量

偏移量 如果该输入短路或连接到相同输入电压，ADC 的测量值。

返回值： 无

副作用： 无

void ADC_SetGain(uint32 chan, int32 adcGain)

说明： 为以下电压转换函数设定每 10V电压的 ADC 增益计数。默认情况下，该值由参考电压和输入范围设置设定。该值仅可用于进一步校准具有已知输入的 ADC，或仅在使用外部参考的情况下使用。通过在 ADC 计数和电压之间进行正确转换，影响 ADC_CountsTo_uVolts、ADC_CountsTo_mVolts 和 ADC_CountsTo_Volts 函数的返回值。

参数： chan：ADC 通道数量。

adcGain：每 10V电压的 ADC 增益计数。

返回值： 无

副作用： 无

float32 ADC_CountsTo_Volts(uint32 chan, int16 adcCounts)

说明： 将 ADC 输出转换为单位为 V 的浮点电压值。例如，如果测得的 ADC 输出为 0.534 V，则返回值为 0.534。电压计算取决于参考电压值。当 Vref 基于 Vdda 的值，则 Vdda 的实际值需要在 DWR 的 System（系统）选项卡中正确设置。

参数： chan：ADC 通道数量。

adcCounts: ADC 转换的结果。

返回值： 结果（单位为 V)

副作用： 无

int16 ADC_CountsTo_mVolts(uint32 chan, int16 adcCounts)

说明： 将 ADC 输出转换为单位为 mV 的 16 位整数电压值。例如，如果测得的 ADC 输出为 0.534 V，则返回值为 534。电压计算取决于参考电压值。当 Vref 基于 Vdda 的值，则 Vdda 的实际值需要在 DWR 的 System（系统）选项卡中正确设置

参数： chan：ADC 通道数量。

adcCounts：ADC 转换的结果。

返回值： 结果（单位为 mV.）

副作用： 无

int32 ADC_CountsTo_uVolts(uint32 chan, int16 adcCounts)

说明： 将 ADC 输出转换为单位为 mV 的 32 位整数电压值。例如，如果测得的 ADC 输出为 0.534 V，则返回值为 534000。电压计算取决于参考电压值。当 Vref 基于 Vdda 的值，则 Vdda 的实际值需要在 DWR 的 System（系统）选项卡中正确设置

参数： chan：ADC 通道数量。

adcCounts: ADC 转换的结果

返回值： 结果（单位 µV）

副作用： 无

void ADC_Sleep(void)

说明： 这是组件准备睡眠需要调用的子程序。ADC_Sleep() 例程保存当前组件状态。然后该程序调用 ADC_Stop() 函数，并调用 ADC_SaveConfig() 保存硬件配置。

在调用 CySysPmDeepSleep() 或 CySysPmHibernate() 函数前调用 ADC_Sleep() 函数。有关功耗管理函数的详细信息，请参考 PSoC Creator *System Reference Guide*（《系统参考指南》）。

参数： 无

返回值： 无

副作用： 无

void ADC_Wakeup(void)

说明： 该函数是将组件恢复到调用 ADC_Sleep() 之前的状态的子程序。ADC_Wakeup() 函数调用 ADC_RestoreConfig() 函数以恢复配置。如果组件在调用 ADC_Sleep() 函数前已启用，则 ADC_Wakeup() 函数将重新启用组件。

参数： 无

返回值： 无

副作用： 在未事先调用 ADC_Sleep() 的情况下调用此函数可能导致不可预知的结果。

void ADC_SaveConfig(void)

说明： 此函数会保存组件配置和非保持寄存器。它还保存 Configure（配置）对话框中定义的或通过相应 API 修改的当前组件参数值。该函数由 ADC_Sleep() 函数调用。

参数： 无

返回值： 无

副作用： 保存所有 ADC 配置寄存器。此函数现在还没有编写内容，但未来会完成。在这里列出是为了兼容整个 API 系统。

void ADC_RestoreConfig(void)

说明： 此函数会恢复组件配置和非保持寄存器。它还将组件参数值恢复为在调用 ADC_Sleep() 函数之前的值。

参数： 无

返回值： 无

副作用： 在未事先调用 ADC_SaveConfig() 或 ADC_Sleep() 的情况下调用此函数可能产生不可预料的行为。此函数现在还没有编写内容，但未来会完成。在这里列出是为了兼容整个 API 系统

MISRA 合规性

本节介绍了本组件与 MISRA-C:2004 的合规和偏差情况。定义了两种类型的偏差：

- 项目偏差 - 适用于所有 PSoC Creator 组件的偏差
- 特定偏差 - 仅适用于此组件的偏差

本节提供了有关组件特定偏差的信息。*系统参考指南*的“MISRA 合规性”章节中介绍项目偏差以及有关 MISRA 合规性验证环境的信息。

此序列 SAR ADC 组件没有任何特定偏差。

该组件配有以下嵌入式构件：中断、时钟 MISRA 合规性与特定偏差的相关信息请参见相应组件数据手册。

固件源代码示例

PSoC Creator 在“查找示例项目”对话框中提供了大量包括原理图和代码的例子项目。要获取组件特定的示例，请打开组件目录中的对话框或原理图中的组件实例。要获取通用的示例，请打开 Start Page（开始页）或 **File**（文件）菜单中的对话框。根据需要，使用对话框中的 **Filter Options**（筛选选项）可缩小可选项目的列表。

有关更多信息，请参考 PSoC Creator 帮助中的“查找示例项目”主题。

中断服务子程序

此序列 SAR ADC 在 *ADC_INT.c* 中包含一个空白中断服务例程。您可以将自定义代码放入指定的区域，以在转换结束后执行所需的任何功能。一个空白中断服务例程副本显示如下。将自定义代码置于“/* `#START MAIN_ADC_ISR` */”和“/* `#END` */”注释之间。此操作可确保项目生成时，代码会被保存。

```
CY_ISR( ADC_ISR )
{
    uint32 intr_status;

    /* Rear interrupt status register */
    intr_status = ADC_SAR_INTR_REG;

    /***************************************************************************
    *  Custom Code
    *  - add user ISR code between the following #START and #END tags
    ***************************************************************************/
    /* `#START MAIN_ADC_ISR`  */

    /* `#END`  */

    /* Clear handled interrupt */
    ADC_SAR_INTR_REG = intr_status;
}
```

第二个指定区域用于放置变量定义和常量定义。

```
/* System variables */

/* `#START ADC_SYS_VAR`  */
    /* Place user code here. */
/* `#END`  */
```

接下来是使用中断捕获数据的代码实例。

```
#include <device.h>

int16 result = 0;
uint8 dataReady = 0;
void main()
{
```

```
    int16 newReading = 0;
    CYGlobalIntEnable;           /* Enable Global interrupts */
    ADC_SAR_1_Start();           /* Initialize ADC */
    ADC_SAR_1_IRQ_Enable();      /* Enable ADC interrupts */
    ADC_SAR_1_StartConvert();    /* Start ADC conversions */
    for(;;)
    {
        if (dataReady != 0)
        {
           dataReady = 0;
           newReading = result;
           /* More user code */
        }
    }
}
```

文件 *ADC_INT.c* 中的中断代码段。

```
/*********************************
*      System variables
*********************************/
/* `#START ADC_SYS_VAR`  */
extern int16 result;
extern uint8 dataReady;
/* `#END`  */

CY_ISR( ADC_ISR )
{
     uint32 intr_status;

     /* Read interrupt status register */
     intr_status = ADC_SAR_INTR_REG;

     /***************************************************************************
     *  Custom Code
     *  - add user ISR code between the following #START and #END tags
     ***************************************************************************/
     /* `#START MAIN_ADC_ISR`  */
     result = ADC_GetResult16(0);
     dataReady = 1;
     /* `#END`  */

     /* Clear handled interrupt */
     ADC_SAR_INTR_REG = intr_status;
}
```

正确设置采样率和主时钟参数很重要。

您可直接读取结果寄存器来优化 ISR：

```
CY_ISR( ADC_ISR )
{
     uint32 intr_status;
```

```
        /* Rear interrupt status register */
        intr_status = ADC_SAR_INTR_REG;

        /*******************************************************************
        *  Custom Code
        *  - add user ISR code between the following #START and #END tags
        *******************************************************************/
        /* `#START MAIN_ADC_ISR`  */
        result = (int16)(ADC_SAR_CHAN0_RESULT_REG & ADC_RESULT_MASK);
        dataReady = 1;
        /* `#END`  */

        /* Clear handled interrupt */
        ADC_SAR_INTR_REG = intr_status;
    }
```

请注意，您可使用您的 main.c 文件中的备选中断服务子程式。在此案例中使用以下模板：

在 main.c 中实现中断服务例程：

```
CY_ISR( ADC_SAR_SEQ_ISR_LOC )
{
         /* Place your code here */
}
```

启用 ADC 中断并设置中断句柄到本地例程：

```
ADC_SAR_SEQ_IRQ_StartEx(ADC_SAR_SEQ_ISR_LOC);
```

功能描述

序列 SAR ADC 包含以下模块：

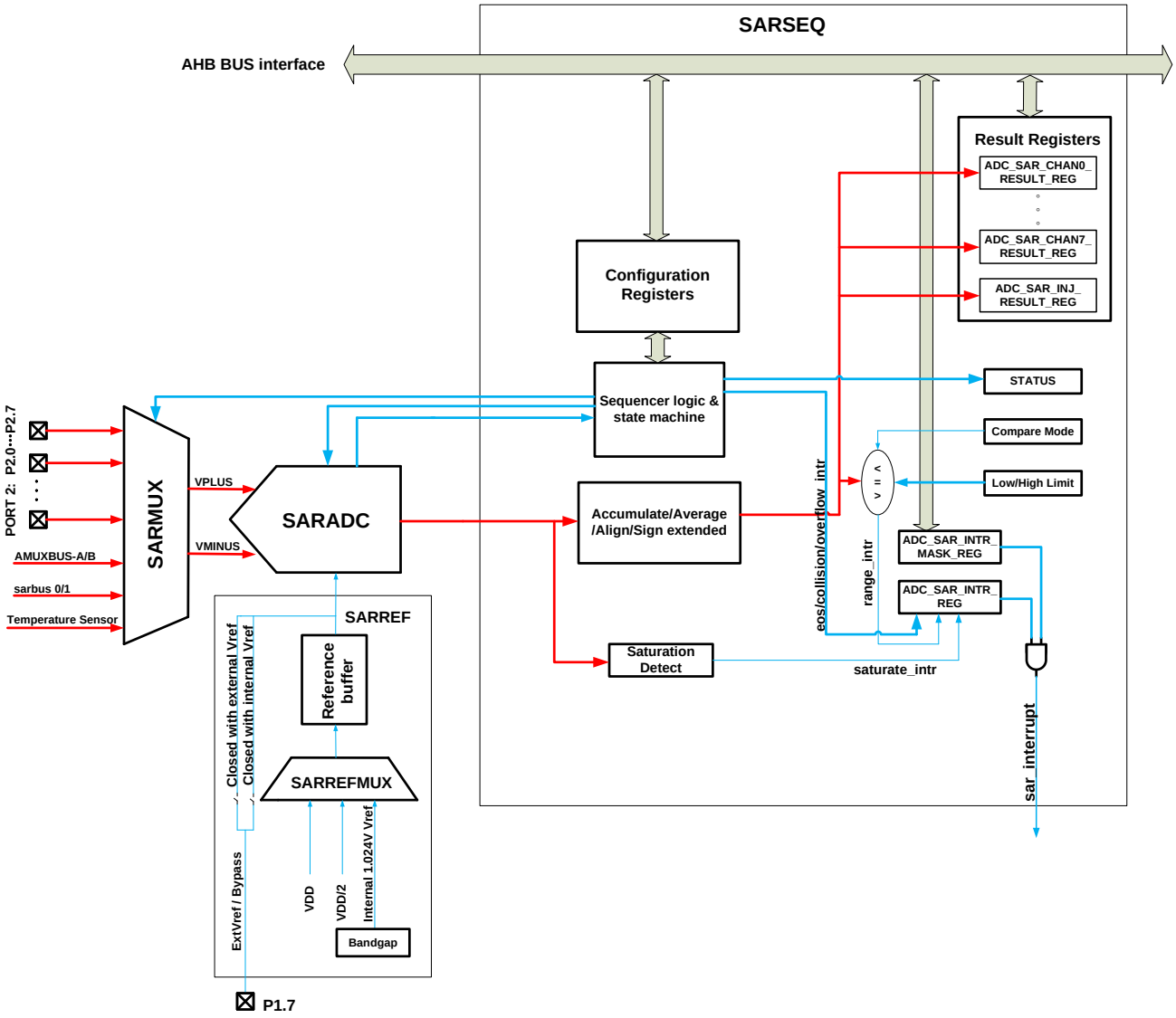
- SARMUX
- SARADC 内核
- SARREF
- SARSEQ

SARADC 内核是一个 SAR 架构的快速逐次逼近型 12 位 ADC。SARADC 前级是 SARMUX，SARMUX 可将外部引脚、温度传感器 (DieTemp) 或运算放大器 (Opamp) 等内部信号路由到 SARADC 的八个内部通道。SARREF 用于选择多个电压参考。SARSEQ 序列控制器模块控制着 SARMUX 和 SARADC，自动扫描所有启用的通道和后处理，如求输出数据的平均值。

第九个通道为插入通道，固件将其用于偶然和附带的引脚和信号（如温度传感器）采样。

每个通道有多个 16 位转换结果存储寄存器。扫描结束后，可屏蔽中断被置位。序列控制器还标记阀值溢出及配置的饱和检测错误，以置位中断。

图 1. 框图


SARSEQ
AHB BUS interface
Result Registers
ADC_SAR_CHAN0_ RESULT_REG
ADC_SAR_CHAN7_ RESULT_REG
ADC_SAR_INJ_ RESULT_REG
Configuration Registers
Sequencer logic & state machine
STATUS
Compare Mode
Low/High Limit
Accumulate/Average /Align/Sign extended
ADC_SAR_INTR_ MASK_REG
ADC_SAR_INTR_ REG
eos/collision/overflow_intr
range_intr
saturate_intr
Saturation Detect
sar_interrupt
PORT 2: P2.0···P2.7
SARMUX
AMUXBUS-A/B
sarbus 0/1
Temperature Sensor
VPLUS
VMINUS
SARADC
SARREF
Reference buffer
SARREFMUX
Closed with external Vref
Closed with internal Vref
ExtVref / Bypass
VDD
VDD/2
Internal 1.024V Vref
Bandgap
P1.7

寄存器

通道结果数据寄存器

此 32 位寄存器包含通道 0 的 16 位 ADC 结果以及描述结果正确性的 3 个状态位。

ADC_SAR_CHAN_RESULT_REG

位	Name（名称）	说明
15:0	数据	第一个通道的 SAR 转换结果。对此扫描中的所有启用通道取样后，此数据从工作字段复制到此处。
29	ADC_SATURATE_INTR_MIR	ADC_SAR_SATURATE_INTR_REG 寄存器中对应位的镜像位
30	ADC_RANGE_INTR_MIR	ADC_SAR_RANGE_INTR_REG 寄存器中对应位的镜像位
31	ADC_CHAN_RESULT_VALID_MIR	ADC_SAR_CHAN_RESULT_VALID_REG 寄存器中对应位的镜像位

剩余通道的结果寄存器依次位于内存中。每个通道的直接定义如下：
ADC_SAR_CHAN*X*_RESULT_REG，*X* 是从 0 至 7 的通道数量。

ADC_SAR_INJ_RESULT_REG

位	Name（名称）	说明
15:0	数据	插入通道的 SAR 转换结果。
28	ADC_INJ_COLLISION_INTR_MIR	ADC_SAR_INTR_REG 寄存器中对应位的镜像位
29	ADC_INJ_SATURATE_INTR_MIR	ADC_SAR_INTR_REG 寄存器中对应位的镜像位
30	ADC_INJ_RANGE_INTR_MIR	ADC_SAR_INTR_REG 寄存器中对应位的镜像位
31	ADC_INJ_EOC_INTR_MIR	ADC_SAR_INTR_REG 寄存器中对应位的镜像位

中断请求寄存器

本节所描述的每个中断都在 ADC_SAR_INTR_MASK_REG 寄存器中有中断屏蔽。向相应的中断屏蔽位写 0，相应的中断源将被忽略。如果 ADC_SAR_INTR_REG 寄存器中相应的中断标志位与 ADC_SAR_INTR_MASK_REG 寄存器中的相应的中断屏蔽位的逻辑与不为 0 时 说明相应位对应的中断被触发。

响应中断时，中断服务子程式 (ISR) 在采集相关数据后，在中断位上写入“1”来清除中断源。

为了方便软件操作，还可在 SADC_SAR_INTR_MASKED_REG 寄存器中查询中断标志和中断屏蔽的逻辑与的结果。

ADC_SAR_INTR_REG

位	Name（名称）	说明
0	ADC_EOS_MASK*	扫描结束中断：固件在完成所有启用的通道的扫描后设置这一中断。在从 ADC_SAR_CHAN_RESULT_REG 寄存器获取数据后，使用 “1” 写入以清除位。
1	ADC_OVERFLOW_MASK	溢出中断：当硬件设置新的 ADC_EOS_MASK 而该位尚未被固件清除时，硬件会设置这一中断。使用 “1” 写入以清除位。
2	ADC_FW_COLLISION_MASK	冲突中断：当 SAR 正在转换时，如果**硬件触发 SAR ADC 转换 或** ADC_StartConvert() API 被调用时，硬件设置该中断。中断产生延迟至 当前扫描完成时。此中断设置后，意味着对通道的取样晚于预期（时序抖动）。使用 “1” 写入以清除位。
3	ADC_DSI_COLLISION_MASK	DSI 冲突中断：当硬件 SOC 触发信号被置位而 SAR 繁忙时，硬件设置此中断。生成中断延迟至 ADC_StartConvert() API 导致的扫描完成时，即不是与触发冲突的先前扫描完成时。此中断设置后，意味着对通道的取样晚于预期（时序抖动）。使用 “1” 写入以清除位。
4	ADC_INJ_EOC_MASK*	插入结束转换中断： 硬件在完成插入通道转换后设置此中断。请注意：ADC_EOS_MASK 与开始插入通道转换同时生成。此插入通道不被视为扫描的一部分。从 ADC_SAR_INJ_RESULT_REG 寄存器挑选数据后，使用 “1” 写入以清除位
5	ADC_INJ_SATURATE_MASK	插入饱和中断：硬件为设置此中断，前提是插入转换结果（取平均值前）为 0x000 或 0xFFF（对于 12 位清晰度）, 这显示 ADC 可能饱和。使用 “1” 写入以清除位。
6	ADC_INJ_RANGE_MASK	插入范围检测中断：硬件为设置此中断，前提是插入转换结果（取平均值前）符合**Compare Mode**（阀值比较模式）参数。使用 “1” 写入以清除位。
7	ADC_INJ_COLLISION_MASK	插入冲突中断。此功能默认禁用。

这两位在 ADC_SAR_INTR_MASK_REG 寄存器中默认被组件启用，并生成一个中断。

ADC_SAR_SATURATE_INTR_REG

位	Name（名称）	说明
15:0	SATURATE_INTR	饱和中断请求寄存器。 硬件为每一通道设置饱和中断，前提是该通道的转换结果（取平均值前）为 0x000 或 0xFFF（对于 12 位清晰度）, 这显示 ADC 可能饱和。当通道选择 10 位或 8 位分辨率后，则忽略高位。使用 “1” 写入以清除位。

ADC_SAR_SATURATE_INTR_MASK_REG

位	Name（名称）	说明
15:0	SATURATE_MASK	饱和中断屏蔽寄存器。 根据选择的**饱和**参数默认设置。使用 ADC_SetSatMask() API 更改此屏蔽寄存器。

ADC_SAR_SATURATE_INTR_MASKED_REG

位	Name（名称）	说明
15:0	SATURATE_MASKED	饱和中断屏蔽结果寄存器。 如果值非零，则 SAR 中断生成。读取时，此寄存器反映饱和中断请求和屏蔽寄存器之间的位 AND。

ADC_SAR_RANGE_INTR_REG

位	Name（名称）	说明
15:0	RANGE_INTR	阀值检测中断请求寄存器。 如果转换结果（求平均值后）符合**Compare Mode**（对比模式）参数规定的条件，硬件设定每一通道的检测中断。使用 “1” 写入以清除位。

ADC_SAR_RANGE_INTR_MASK_REG

位	Name（名称）	说明
15:0	RANGE_MASK	阀值检测中断屏蔽寄存器。 按照选择的**Limit 检测**参数默认设置。使用 ADC_SetLimitMask() API 更改此屏蔽寄存器。

ADC_SAR_RANGE_INTR_MASKED_REG

位	Name（名称）	说明
15:0	RANGE_MASKED	范围中断屏蔽结果寄存器。 如果此值非零，则生成 SAR 中断。读取时，此寄存器反映检测中断请求和屏蔽寄存器之间的位 AND。

资源

序列 SAR ADC 作为一个固定功能模块执行。该组件也使用一个中断。

API 存储器使用

根据不同的编译器、设备、所用 API 的数量以及组件配置，组件存储器的使用有着显著的不同。此表说明了在默认组件配置中可用的所有 API 的存储器使用情况。

已利用 release 模式中配置的相关编译器进行了测量，大小采用了优化设定。对于特定设计，可以分析编译器生成的映射文件以确定存储器使用情况。

配置	PSoC 4 (GCC)	
	闪存字节	SRAM 字节
默认值	956	29

直流和交流电气特性

除非另有说明，否则这些规范的适用条件是 -40 °C ≤ T_A ≤ 85 °C 且 T_J ≤ 100 °C。除非另有说明，否则这些规范的适用范围为 1.71 V 到 5.5 V。

直流电规范

参数	说明	最小值	典型值	最大值	单位	条件
A_RES	分辨率	–	–	12	位	
A_CHNIS_S	通道数量 - 单端	–	–	8		8 全速
A-CHNKS_D	通道数量 - 差分	–	–	4		差分输入使用相邻 I/O
A-MONO	单调性	–	–	–		是
A_GAINERR	增益误差	–	–	±0.1	%	使用外部参考
A_OFFSET	输入失调电压	–	–	2	mV	使用 1-V V_{REF} 测量
A_ISAR	电流消耗	–	–	1	mA	
A_VINS	输入电压范围 - 单端	V_{SS}	–	V_{DDA}	V	
A_VIND	输入电压范围 - 差分	V_{SS}	–	V_{DDA}	V	
A_INRES	输入电阻	–	–	2.2	KΩ	

参数	说明	最小值	典型值	最大值	单位	条件
A_INCAP	输入电容	-	-	10	pF	

交流电规范

参数	说明	最小值	典型值	最大值	单位	条件
A_PSRR	电源抑制比	70	–	–	dB	
A_CMRR	共模抑制比	66	–	–	dB	在 1 V 电压下测量
A_SAMP	采样率	–	–	1	Msps	
A_SNDR	信噪比和失真比 (SINAD)	65	–	–	dB	F_{IN} = 10 kHz
A_INL	积分非线性	–1.7	–	+2	LSB	V_{DD} = 1.71 至 5.5，1 Msps， Vref = 1 至 5.5
A_INL	积分非线性	–1.5	–	+1.7	LSB	V_{DDD} = 1.71 至 3.6，1 Msps，Vref = 1.71 至 V_{DDD}
A_INL	积分非线性	–1.5	–	+1.7	LSB	V_{DDD} = 1.71 to 5.5, 500 Ksps, Vref = 1 to 5.5
A_DNL	微分非线性	–1	–	+2.2	LSB	V_{DDD} = 1.71 to 5.5, 1 Msps, Vref = 1 to 5.5
A_DNL	微分非线性	–1	–	+2	LSB	V_{DDD} = 1.71 至 3.6，1 Msps, Vref = 1.71 to V_{DDD}
A_DNL	微分非线性	–1	–	+2.2	LSB	V_{DDD} = 1.71 至 5.5，500 Ksps，Vref = 1 至 5.5
A_THD	总谐波失真	–	–	–65	dB	F_{IN} = 10 kHz。

组件更改

本节介绍组件与以前版本相比的主要更改。

版本	更改说明	更改/影响原因
1.0	首次组件发行	

赛普拉斯半导体公司，2013-2016 年。本文件是赛普拉斯半导体公司及其子公司，包括 Spansion LLC（“赛普拉斯”）的财产。本文件，包括其包含或引用的任何软件或固件（“软件”），根据全球范围内的知识产权法律以及美国与其他国家签署条约由赛普拉斯所有。除非在本款中另有明确规定，赛普拉斯保留在该等法律和条约下的所有权利，且未就其专利、版权、商标或其他知识产权授予任何许可。如果软件并不附随有一份许可协议且贵方未以其他方式与赛普拉斯签署关于使用软件的书面协议，赛普拉斯特此授予贵方属人性质的、非独家且不可转让的如下许可（无再许可权）（1）在赛普拉斯特软件著作权项下的下列许可权（一）对以源代码形式提供的软件，仅出于在赛普拉斯硬件产品上使用之目的且仅在贵方集团内部修改和复制软件，和（二）仅限于在有关赛普拉斯硬件产品上使用之目的将软件以二进制代码形式的向外部最终用户提供（无论直接提供或通过经销商和分销商间接提供），和（2）在被软件（由赛普拉斯公司提供，且未经修改）侵犯的赛普拉斯专利的权利主张项下，仅出于在赛普拉斯硬件产品上使用之目的制造、使用、提供和进口软件的许可。禁止对软件的任何其他使用、复制、修改、翻译或汇编。

在适用法律允许的限度内，赛普拉斯未对本文件或任何软件作出任何明示或暗示的担保，包括但不限于关于适销性和特定用途的默示保证。赛普拉斯保留更改本文件的权利，届时将不另行通知。在适用法律允许的限度内，赛普拉斯不对因应用或使用本文件所述任何产品或电路引起的任何后果负责。本文件，包括任何样本设计信息或程序代码信息，仅为供参考之目的提供。文件使用人应负责正确设计、计划和测试信息应用和由此生产的任何产品的功能和安全性。赛普拉斯产品不应被设计为、设定为或授权用作武器操作、武器系统、核设施、生命支持设备或系统、其他医疗设备或系统（包括急救设备和手术植入物）、污染控制或有害物质管理系统中的关键部件，或产品植入之设备或系统故障可能导致人身伤害、死亡或财产损失其他用途（“非预期用途”）。关键部件指，若该部件发生故障，经合理预期会导致设备或系统故障或会影响设备或系统安全性和有效性的部件。针对由赛普拉斯产品非预期用途产生或相关的任何主张、费用、损失和其他责任，赛普拉斯不承担全部或部分责任且贵方不应追究赛普拉斯之责任。贵方应赔偿赛普拉斯因赛普拉斯产品任何非预期用途产生或相关的所有索赔、费用、损失和其他责任，包括因人身伤害或死亡引起的主张，并使之免受损失。

赛普拉斯、赛普拉斯徽标、Spansion、Spansion徽标，及上述项目的组合，及PSoC、CapSense、EZ-USB、F-RAM和Traveo 应视为赛普拉斯在美国和其他国家的商标或注册商标。请访问 cypress.com 获取赛普拉斯商标的完整列表。其他名称和品牌可能由其各自所有者主张为该方财产。