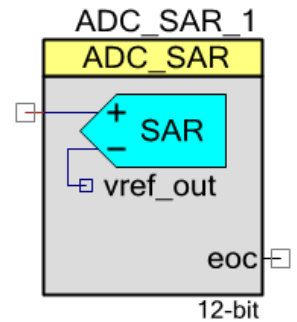


ADC 逐次逼近寄存器 (ADC_SAR)

2.10

特性

- 支持 PSoC 5LP 器件系列
- 12 位分辨率，最大采样率 1 Msps
- 支持四种功耗模式
- 可配置的分辨率和采样率
- 单端或差分输入



概述

ADC 逐次逼近寄存器 (ADC_SAR) 组件提供最大 1 Msps 采样率，最高 12 位分辨率的模数转换功能。

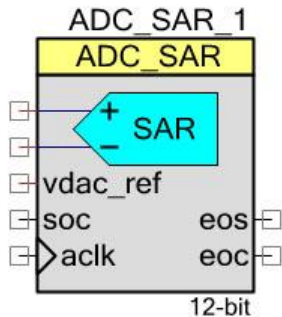
何时使用 ADC_SAR

ADC_SAR 组件的典型应用包括：

- LED 照明控制
- 电机控制
- 磁卡读卡器
- 高速数据收集
- 功率计
- 脉搏血氧计

输入/输出接口

本节介绍 ADC_SAR 的输入和输出接口。I/O 列表中的星号 (*) 表示：在 I/O 说明部分中所列出的情况下，该 I/O 可能不可见。



+输入 — 模拟

此输入是 ADC_SAR 的正向模拟信号输入。转换结果是一个+输入信号减去电压参考的函数。电压参考是-输入信号或 V_{SSA}。

-输入 — 模拟*

显示时，此可选输入是 ADC_SAR 的负向模拟信号（或参考）输入。转换结果取决于+输入减去-输入。在将 **Input Range**（输入范围）参数设置为差分模式之一时可看到此引脚。

vdac_ref — 输入*

VDAC 参考 (vdac_ref) 是可选引脚。如果选择了 **Vssa to VDAC*2 (Single Ended)** (Vssa 至 VDAC*2 (单端)) 或 **0.0 +/- VDAC (Differential)** (0.0 +/- VDAC (差分)) 输入范围，则可看到该引脚；否则，此 I/O 会隐藏。只能将此引脚连接到 VDAC 组件输出。请勿将其连接到任何其他信号。

soc — 输入*

开始转换 (soc) 是可选引脚。如果您选择**硬件触发**采样模式，该输入可见。该输入的上升沿会触发 ADC 转换。调用 SAR_Start()函数时，如果该输入为高电平，转换将立即启动。在第一个转换完成后，输入上的上升沿将启动一个 ADC 转换。此信号必须与 ADC_SAR 时钟同步。如果将 **Sample Mode**（采样模式）参数设置为 **Free Running**（自由运行），则此 I/O 会隐藏。

aclk — 输入*

如果将 **Clock Source** (时钟源) 参数设置为 **External** (外部), 则可看到此可选引脚; 否则, 此引脚会隐藏。在转换模式和分辨率选定的情况下, 该时钟确定了 ADC 的转换速率。

eos — 输出*

采样结束 (eos) 输出上的上升沿表示采样窗口已完成。该信号可用于控制输入通道复用器。采样完成后, 即使转换还没有结束, 输入复用器选择开关仍将切换到下一个通道。采样结束 (eos) 信号允许 SAR ADC 以最大的速度运行。在选择了 **Enable EOS output** (使能 EOS 输出) 参数时, 该输出可见。

eoc — 输出

结束转换 (eoc) 输出上的上升沿表示转换已经完成。可以将 DMA 请求连接到此引脚以将转换输出传输到系统 RAM、DFB 或其他组件。还可将内部中断连接到此信号, 或者可以连接自己的中断。

组件参数

将一个 ADC_SAR 组件拖放到您的设计上, 并双击以打开 **Configure** (配置) 对话框。

Configure 'ADC_SAR'

Name:

Configure Built-in

Modes

Resolution (bits):

Conversion rate (SPS):

Actual conv. rate (SPS): 631579

Clock frequency (kHz):

Sample mode

☒ Free running

☐ Software trigger

☐ Hardware trigger

Clock source

☒ Internal

☐ External

Input

Input range:

Reference:

Voltage reference: Volts

☐ Enable EOS output

ADC_SAR 具有以下参数。粗体表示默认选项。

Modes (模式)

Resolution (分辨率)

设置 ADC 的分辨率。

ADC_Resolution	值	说明
12	12	将分辨率设置为12位。
10	10	将分辨率设置为10位。
8	8	将分辨率设置为8位。

Conversion Rate (转换速率)

此参数设置 ADC 转换。转换时间与转换速率成反比。以每秒采样数为单位输入转换速率。转换一个采样需要 18 个时钟周期。根据可用的时钟速度和分频器范围的情况不一样，实际的转换速率会有所不同。

Clock Frequency (时钟频率)

该文本框为只读（始终为灰色）区域，用于显示所选工作条件（分辨率和转换速率）所需的时钟速率。当这两个条件中的任一条件更改或两个条件都更改时，会更新该文本框。时钟频率范围介于 1 MHz 和 18 MHz 之间。占空比应为 50%。最小脉冲宽度应大于 25.5 ns。如果时钟不在这些限制范围内，则 PSoC Creator 会在编译过程中生成错误。在这种情况下，在设计范围资源时钟编辑器中更改主设备时钟。

在高转换速率下，ADC 可生成大量数据。在这些情况下，需要使用 DMA 或 CPU 来收集数据。如果使用 CPU，该 CPU 时钟必须为高时钟频率，并要求具有最小的中断服务子程序。例如，对于每秒钟 700,000 个样本的转换速率和 66 MHz 的 CPU 时钟速率，每个样本仅有 $66 \text{ MHz} / 700,000 \text{ sps} = 94$ 个 CPU 时钟周期来处理。有关 ISR 的优化指南，请参见《中断服务子程序》章节。

Sample Mode (采样模式)

该参数确定了 ADC 的工作方式。

Start_of_Conversion	说明
Free Running (连续运行)	ADC连续运行。
Software trigger (软件触发器)	ADC_StartConvert()函数可以触发一个转换。
Hardware trigger (硬件触发)	SOC引脚上的上升沿脉冲会触发一个转换。

Clock Source (时钟源)

通过此参数可以选择 ADC_SAR 模块内部的时钟或外部时钟。

ADC_Clock	说明
Internal (内部)	使用ADC_SAR的内部时钟。
External (外部)	使用外部时钟。时钟源可以是模拟、数字或由其他组件生成。

Input (输入)

Input Range (输入范围)

该参数按给定的输入范围配置 ADC。无论输入范围的设置情况如何，连接至 PSoC 的模拟信号都必须在 V_{SSA} 与 V_{DDA} 的范围内。

输入范围	说明
0.0至2.048 V (单端) 0至Vref*2	当使用内部参考 (1.024 V) 时，可用输入范围为0.0至2.048 V。ADC配置为在单端输入模式下运行，同时-输入在内部连接到Vrefhi_out。如果使用外部参考电压，可用的输入范围为0.0至Vref*2。
Vssa to Vdda (Single Ended) (Vssa 至 Vdda (单端))	此模式使用 $V_{DDA}/2$ 参考；可用输入范围完整涵盖模拟电源电压。ADC置于单端输入模式，同时-输入在内部连接到Vrefhi_out。如果使用外部参考电压，可用的输入范围为0.0至Vref*2。
Vssa to VDAC*2 (Single Ended) (Vssa 至VDAC*2 (单端))	此模式使用VDAC参考，该参考应连接到vdac_ref引脚。可用的输入范围为Vssa到VDAC*2伏。ADC配置为在单端输入模式下运行，同时-输入在内部连接到Vrefhi_out。
0.0±1.024V (差分) 负输入± Vref	此模式针对差分输入配置。当使用内部参考 (1.024 V) 时，输入范围为-输入±1.024 V。 例如，如果-输入连接到2.048 V，则可用输入范围为2.048±1.024 V或1.024至3.072 V。对于既需要扫描单端又需要扫描差分信号的系统，在扫描单端输入时将-输入连接到Vssa。 您可以使用外部参考源以提供更广的操作范围。可以使用相等等式“-输入±Vref”来计算可用的输入范围。
0.0±Vdda (差分) 负输入±Vdda	此模式配置用于差分输入，与输入电压成比例关系。输入范围为-输入±Vdda。对于既需要扫描单端又需要扫描差分信号的系统，在扫描单端输入时，将-输入连接至Vssa。如果使用外部参考电压，则可用输入范围为-输入±Vref。
0.0±Vdda/2 (差分) 负输入±Vdda/2	此模式配置用于差分输入，与输入电压成比例关系。输入范围为-输入±Vdda/2。对于需要扫描单端和差分信号的系统，在扫描单端输入时，将输入连接至Vssa。如果使用外部参考电压，则可用的输入范围为-输入±Vref。

输入范围	说明
0.0±VDAC（差分） 负输入±VDAC	此模式配置用于差分输入，并使用VDAC参考。该参考应连接到vdac_ref引脚。输入范围为–输入 ±VDAC。对于需要扫描单端和差分信号的系统，在扫描单端输入时，将–输入连接至Vssa。

Reference（参考）

此参数用于选择 ADC_SAR 参考配置的开关。

ADC_Reference	说明
Internal Vref（内部Vref）	使用内部参考。该选项允许的最大采样率为100,000 sps，“0.0 ± Vdda” 除外。对于较高的采样率，请使用Internal Vref, bypassed选项。
Internal Vref, bypassed（内部Vref，旁路）	使用内部参考；必须将一个旁路电容器置于SAR1的引脚P0[2]*上或SAR0的引脚P0[4]*上。该模式不适用于“0.0 ± Vdda” 输入范围。
外部Vref	在SAR1的引脚P0[2]*上或SAR0的引脚P0[4]上使用外部参考。

* 如果内部数字开关引发的噪音超过某个应用对模拟性能的要求，建议使用外部旁路电容。若要使用该选项，请将端口引脚 P0[2]或 P0[4]配置为模拟高阻引脚，并连接到介于 0.01 μF 至 10 μF 范围内的外部电容。

注意：ADC_SAR 和 ADC_DelSig 组件使用相同的内部参考。如果两类 ADC 同时需要使用内部参考，请使用 Internal Vref, bypassed 选项以获得最佳的性能。

注意：使用外部参考或外部旁路内部参考时，请使用 ADC_SAR:ExtVref 或 ADC_SAR:Bypass 引脚上设计范围资源（DWR）的引脚选项卡中的锁定特性。这样会将 SAR 组件锁定到指定的 SAR 硬件模块内。

Voltage Reference（电压参考）

电压参考用于计算《应用编程接口》一节中讨论的电压转换函数的 ADC。当使用内部参考时，该参数为只读。当使用外部参考时，可以编辑该值以匹配外部参考电压。

- 选择输入范围 Vssa至 Vdda、-输入 +/- Vdda 或 -输入 +/- Vdda/2时，其值由“设计范围资源”（DWR）窗口中系统选项卡内的 VDDA 设置演变而来。
- 当选择输入范围 Vssa至 VDAC*2 或–输入 +/- VDAC 时，请输入 VDAC 电源电压值。

注意：输入范围和参考电压受 VDDA 电压的限制。

Enable EOS output（使能 EOS 输出）

此参数使能“EOS”输出。



应用编程接口

通过应用编程接口 (API)，您可以使用软件对组件进行配置。下表列出并说明了每个函数的接口。后面部分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_1”分配给指定设计中组件的第一个实例。您可以将该实例重新命名为符合标识符语法规则的任意唯一值。实例名称会成为每个全局函数名称、变量和符号常量的前缀。为增加可读性，下表中使用了实例名称“ADC”。

函数	说明
ADC_Start()	对ADC加电并复位所有状态
ADC_Stop()	停止ADC转换并将功耗减少到最小值
ADC_SetPower()	设置功耗模式
ADC_SetResolution()	设置ADC的分辨率
ADC_StartConvert()	开始转换
ADC_StopConvert()	停止转换
ADC_IRQ_Enable()	将内部IRQ连接到eoc。该API使能了内部ISR。
ADC_IRQ_Disable()	将内部IRQ连接到eoc。该API禁用了内部ISR。
ADC_IsEndConversion()	如果转换完成，将返回非零值
ADC_GetResult8()	返回有符号8位转换结果
ADC_GetResult16()	返回有符号16位转换结果
ADC_SetOffset()	设置ADC的偏移
ADC_SetScaledGain()	设置ADC增益，其单位为每10伏计数。
ADC_CountsTo_Volts()	将ADC计数转换为单位为伏的浮点电压值
ADC_CountsTo_mVolts()	将ADC计数转换为单位为毫伏的电压值
ADC_CountsTo_uVolts()	将ADC计数转换为单位为微伏的电压值
ADC_Sleep()	停止ADC操作，并保存用户配置
ADC_Wakeup()	恢复并使能用户配置
ADC_Init()	初始化自定义程序所提供的默认配置
ADC_Enable()	使能ADC的时钟，并使其通电
ADC_SaveConfig()	保存当前用户配置
ADC_RestoreConfig()	恢复用户配置

全局变量

变量	说明
ADC_initVar	该变量指出ADC是否已初始化。该变量初始化为0，并在第一次调用ADC_Start()时设置为1。这样，第一次调用ADC_Start()子程序后，组件不用重新初始化即可重启。 如果需要重新初始化组件，可在调用ADC_Start()或ADC_Enable()函数前调用TIA_Init()函数。
ADC_offset	此变量校准偏移。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，该变量可影响ADC_CountsTo_Volts()、ADC_CountsTo_mVolts()和ADC_CountsTo_uVolts()函数。
ADC_countsPer10Volt	此变量用于校准增益。它在首次调用ADC_Start()时以及每次调用ADC_SetResolution()时进行计算。其值取决于分辨率、输入范围和电压参考。可以使用ADC_SetScaledGain()更改它。 通过在ADC计数和施加的输入电压之间进行正确转换，此变量可影响ADC_CountsTo_Volts、ADC_CountsTo_mVolts和ADC_CountsTo_uVolts函数。
ADC_shift	在差分输入模式中，SAR ADC以二进制偏移方案输出经过数字转换的数据。此变量用于将ADC计数转换为2的补码形式。 此变量在首次调用ADC_Start()时以及每次调用ADC_SetResolution()时进行计算。计算的 值取决于分辨率和输入模式。 通过减去正确移位值，此变量可影响ADC_GetResult8()和ADC_GetResult16()函数。

void ADC_Start(void)

- 说明：

这是开始执行组件操作的首选方法。ADC_Start()函数设置initVar变量，调用ADC_Init()函数，然后调用ADC_Enable()函数。
- 参数：

无
- 返回值：

无
- 其他影响：

如果已设置initVar变量，则该函数仅调用ADC_Enable()函数。

void ADC_Stop(void)

- 说明：

停止ADC转换并将功耗减少到最小值。
- 参数：

无
- 返回值：

无
- 其他影响：

无

void ADC_SetPower(uint8 power)

说明: 设置ADC的运行功耗。应将较高功耗设置用于较快时钟速度。

参数: uint8 power: 功耗设置

参数名称	值	说明	时钟频率
ADC__HIGHPOWER	0	Normal (正常) 功耗	18 MHz
ADC__MEDPOWER	1	1/2功耗	4.5 MHz
ADC__LOWPOWER	2	1.25倍功耗	保留
ADC__MINPOWER	3	1/4功耗	2.25 MHz

返回值: 无

其他影响: 功耗设置可能会影响转换精度。

void ADC_SetResolution(uint8 resolution)

说明: 设置GetResult16()和GetResult8() API的分辨率。

参数: uint8 resolution: 设置分辨率

参数名称	值	说明
ADC__BITS_12	12	将分辨率设置为12位。
ADC__BITS_10	10	将分辨率设置为10位。
ADC__BITS_8	8	将分辨率设置为8位。

返回值: 无

其他影响: 在转换周期过程中不能更改ADC分辨率。建议的最佳实践是使用ADC_StopConvert()停止转换，更改分辨率，然后使用ADC_StartConvert()重新启动转换。

如果决定在调用此API之前不停止转换，请使用ADC_IsEndConversion()等到转换完成，然后再更改分辨率。

如果在转换过程中调用ADC_SetResolution()，则在当前转换完成之前，分辨率不会更改。对于另外6 + “新分辨率（以位为单位）”个时钟周期，数据在新分辨率中不可用。在调用ADC_SetResolution()之后可能需要添加此时钟周期数的延迟，数据才能再次有效。

通过在ADC计数和施加的输入电压之间计算正确转换，可影响ADC_CountsTo_Volts()、ADC_CountsTo_mVolts()和ADC_CountsTo_uVolts()。计算取决于分辨率、输入范围和电压参考。

void ADC_StartConvert(void)

- 说明:** 强制ADC启动一个转换。在自由运行模式中，ADC连续运行。在软件触发模式中，该函数还充当软件版本的SOC，并且必须通过ADC_StartConvert()触发每个转换。选择**硬件触发**采样模式时，该函数则不可用。
- 参数:** 无
- 返回值:** 无
- 其他影响:** 调用ADC_StartConvert()会禁用外部SOC引脚。

void ADC_StopConvert(void)

- 说明:** 强制ADC停止转换。如果当前正在执行转换，将会完成该转换，并且没有启动其他转换。选择**硬件触发**采样模式时，该函数则不可用。
- 参数:** 无
- 返回值:** 无
- 其他影响:** 在**软件触发**采样模式中，此函数将软件版本的SOC设置为低级别并将SOC源切换为硬件SOC输入。

void ADC_IRQ_Enable(void)

- 说明:** 转换结束后，使能中断。必须使能全局中断，以实现ADC中断。若要使能全局中断，在使能任何中断之前，请调用main.c文件内的使能全局中断宏“CYGlobalIntEnable;”。
- 参数:** 无
- 返回值:** 无
- 其他影响:** 使能中断。读取结果将清除中断。

void ADC_IRQ_Disable(void)

- 说明:** 转换结束后禁用中断。
- 参数:** 无
- 返回值:** 无
- 其他影响:** 无

uint8 ADC_IsEndConversion(uint8 retMode)

说明: 根据retMode参数的设置情况，会立即返回转换状态或不会返回转换状态直到转换完成（锁住）为止。

参数: uint8 retMode: 检查转换返回模式。有关各选项，请参见下表。

选项	说明
ADC_RETURN_STATUS	立即返回状态。如果返回值为零，则表示转换未完成，这时应重试该函数，直至返回非零值为止。
ADC_WAIT_FOR_RESULT	在ADC转换完成之前不返回结果。

返回值: uint8: 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。

其他影响: 该函数可读取转换结束的状态，并在读取后会清除。

int8 ADC_GetResult8(void)

说明: 返回8位转换的结果。如果设置的分辨率大于8位，则函数返回结果的最低有效位。应调用ADC_IsEndConversion()以验证数据采样是否就绪。

参数: 无

返回值: int8: 最后一次ADC转换的最低有效位。

其他影响: 将ADC计数转换为2的补码形式。

int16 ADC_GetResult16(void)

说明: 返回转换的16位结果，以及分辨率为8至12位的结果。应调用ADC_IsEndConversion()以验证数据采样是否就绪。

参数: 无

返回值: int16: 最后一次ADC转换的16位结果

其他影响: 将ADC计数转换为2的补码形式。

void ADC_SetOffset(int16 offset)

- 说明:** 设置ADC_CountsTo_Volts()、ADC_CountsTo_mVolts()和ADC_CountsTo_uVolts()所使用的ADC偏移,以便在计算电压转换前从给定读数中减去该偏移量。
- 参数:** int16 offset: 如果该输入短路或连接到相同输入电压,则该值为测量值。
- 返回值:** 无
- 其他影响:** 通过减去给定偏移,可影响ADC_CountsTo_Volts()、ADC_CountsTo_mVolts()和ADC_CountsTo_uVolts()。

void ADC_SetScaledGain(int16 adcGain)

- 说明:** 为以下电压转换函数设定ADC增益(单位为每10 V电压的计数)。默认情况下,该值由参考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ADC使用外部参考的情况下使用。
- 参数:** int16 adcGain: ADC增益(单位为每10 V电压的计数)。要校准该增益,向ADC输入提供参考电压并通过使用万用表测量此电压。使用以下公式来计算增益系数:

$$adcGain = \frac{counts \times 10}{V_{measured}}$$

其中, **counts** (计数)是从ADC_GetResult16()值返回的值, $V_{measured}$ — 使用万用表进行测量,它的单位为伏特。

- 返回值:** 无
- 其他影响:** 通过在ADC计数和施加的输入电压之间进行正确转换,影响ADC_CountsTo_Volts()、ADC_CountsTo_mVolts()和ADC_CountsTo_uVolts()函数。

float ADC_CountsTo_Volts(int16 adcCounts)

- 说明:** 将ADC输出转换为单位为伏的浮点电压值。例如,如果测得的ADC输出为0.534 V,则返回值为0.534。电压计算取决于电压参考值。当选定的 V_{ref} 是 V_{DDA} 的派生电压时,显示的输入范围将依赖于用户在 .cydwr-->system 选项卡中 V_{DDA} 的设置值。
- 参数:** int16 adcCounts: ADC转换的结果
- 返回值:** Float: 返回单位为伏的电压值。
- 其他影响:** 无

int16 ADC_CountsTo_mVolts(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。电压计算取决于电压参考值。当选定的Vref是VDDA的派生电压时，显示的输入范围将依赖于用户在.cydw-->system选项卡中VDDA的设置值。
- 参数:** int16 adcCounts: ADC转换的结果
- 返回值:** int16: 以mV为单位的电压值
- 其他影响:** 无

int32 ADC_CountsTo_uVolts(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。电压计算取决于电压参考值。当选定的Vref是VDDA的派生电压时，显示的输入范围将依赖于用户在.cydw-->system选项卡中VDDA的设置值。
- 参数:** int16 adcCounts: ADC转换的结果
- 返回值:** int32: 以μV为单位的电压值
- 其他影响:** 无

void ADC_Sleep(void)

- 说明:** 这是组件准备进入睡眠模式时的首选函数。ADC_Sleep()函数保存当前组件的状态，然后调用ADC_Stop()函数。
- 在调用CyPmSleep()或CyPmHibernate()函数之前调用ADC_Sleep()函数。有关功耗管理函数的详细信息，请参考PSoC Creator 《系统参考指南》中介绍的内容。
- 参数:** 无
- 返回值:** 无
- 其他影响:** 无

void ADC_Wakeup(void)

- 说明:** 该函数是将组件恢复到调用ADC_Sleep()时状态的首选函数。如果组件在调用ADC_Sleep()函数前已使能，则ADC_Wakeup()函数将重新使能组件。
- 参数:** 无
- 返回值:** 无
- 其他影响:** 调用ADC_Wakeup()函数前未调用ADC_Sleep()或ADC_SaveConfig()函数可能会产生意外行为。

void ADC_Init(void)

- 说明:** 根据自定义程序“Configure”对话框设置初始化或恢复组件。无需调用ADC_Init(), 因为ADC_Start()子程序会调用该函数并是开始组件操作的首选方法。
- 参数:** 无
- 返回值:** 无
- 其他影响:** 根据自定义程序“Configure”对话框中的内容, 设置所有寄存器。

void ADC_Enable(void)

- 说明:** 激活硬件, 并开始执行组件操作。会根据时钟速度自动设置较高功耗。ADC_SetPower() API 说明包含功耗与时钟速率的关系。无需调用ADC_Enable(), 因为ADC_Start()子程序会调用该函数, 这是开始组件操作的首选方法。
- 参数:** 无
- 返回值:** 无
- 其他影响:** 无

void ADC_SaveConfig(void)

- 说明:** 该函数会保存组件配置和非保留寄存器。它还保存Configure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
- 其他影响:** 保留所有ADC配置寄存器。该函数不会实现, 以备将来使用。在此提供该函数, 以便使API在整个组件中保持一致。

void ADC_RestoreConfig(void)

- 说明:** 该函数会恢复组件配置和非保留寄存器。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
- 其他影响:** 调用此函数前未调用ADC_Sleep()或ADC_SaveConfig()函数可能会产生意外行为。该函数不会实现，以备将来使用。在此提供该函数，以便使API在整个组件中保持一致。

MISRA 合规性

本节介绍了MISRA-C:2004合规性和本器件的偏差情况。定义了下面两种类型的偏差：

- 项目偏差 — 适用于所有 PSoC Creator 组件的偏差
- 特定偏差 — 仅适用于该组件的偏差

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”章节中介绍了项目偏差以及有关 MISRA 合规性验证环境的信息。

ADC_SAR 组件没有任何特定偏差。

该组件使用了下面的嵌入式组件：中断和时钟组件。欲了解 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。

DMA

可以使用 DMA 组件将转换的结果从 ADC_SAR 寄存器传输到 RAM。应将 DMA 数据请求信号 (DRQ) 从 ADC 连接至 EOC 引脚。用户可以使用 DMA 向导按如下所示配置 DMA 操作：

DMA源的名称	长度	方向	DMA 请求信号	DMA 请求类型	说明
ADC_SAR_WRK0_PTR	2	源	EOF	上升沿	接收转换的2字节结果，以及分辨率始终为12位的结果。 请注意，此寄存器未进行符号扩展；结果始终是无符号的。0-V差分输入返回半量程代码。完整负向输入返回0代码，完整正向输入返回满量程代码。

示例固件源代码

PSoC Creator 在“Find Example Project”（查找示例项目）对话框中提供了多种包括原理图和代码示例的示例项目。要查看特定组件示例，请打开“Component Catalog”中的对话框或原理图中的组件实例。要查看通用示例，请打开“Start Page”或“File”菜单中的对话框。根据要求，可以通过使用对话框中的 **Filter Options** 选项来限定可选的项目列表。

更多有关信息，请参考《PSoC Creator 帮助》部分中主题为“查找示例项目”中的内容。

中断服务子程序

ADC_SAR 在 ADC_SAR_1_INT.c 文件中（其中“ADC_SAR_1”为实例名称）包含空白中断服务子程序。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。下面显示了空白中断服务子程序的副本。将自定义代码置于“/* `#START MAIN\_ADC\_ISR` */”和“/* `#END` */”注释之间。这可确保在重新生成项目时保留代码。

```
CY_ISR( ADC_SAR_1_ISR )
{
    /* Place user ADC ISR code here. This can be a good place */
    /* to place code that is used to switch the input to the */
    /* ADC. It may be good practice to first stop the ADC */
    /* before switching the input then restart the ADC. */

    /* `#START MAIN_ADC_ISR` */
    /* Place user code here. */
    /* `#END` */
}
```



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` */
```

下面是使用中断捕获数据的代码示例。

```
#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_SAR_1_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_SAR_1_ISR )
{
    /******
    /* Place user ADC ISR code here.          */
    /* This can be a good place to place code */
    /* that is used to switch the input to the */
    /* ADC. It may be good practice to first   */
    /* stop the ADC before switching the input */
    /* then restart the ADC.                   */
    /******
    /* `#START MAIN_ADC_ISR` */
    result = ADC_SAR_1_GetResult16();
```



```

        dataReady = 1;
    /* `#END` */
}

```

正确设置 **Conversion Rate**（转换速率）和 **Master Clock**（主设备时钟）参数十分重要。

例如，对于最大转换速率（12 位时为 700 ksp/s），在设计范围资源时钟编辑器中将 **Master Clock** 设置为 53 MHz，并优化 **ISR** 子程序。否则，处理器将无法以足够快的速度处理 **ISR**。如果选择较低 **Master Clock**，则 **ISR** 的运行时间会长于 **ADC_SAR** 转换时间。

可以通过直接读取采样寄存器优化 **ISR**：

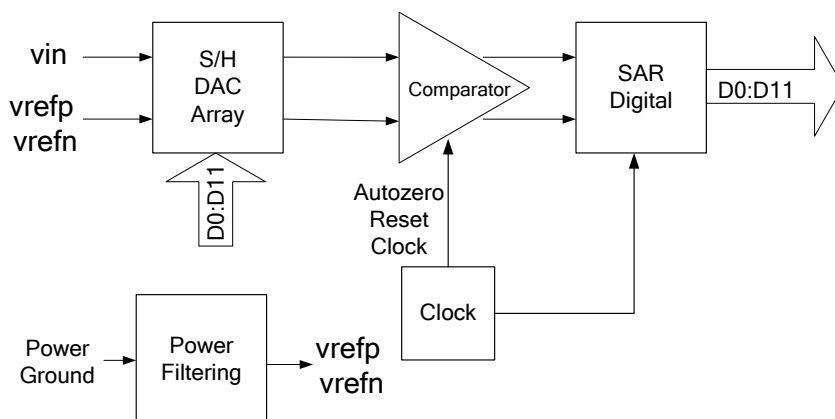
```

CY_ISR(ADC_SAR_1_ISR )
{
    /******
    /* Place user ADC ISR code here.
    /* This can be a good place to place code
    /* that is used to switch the input to the
    /* ADC. It may be good practice to first
    /* stop the ADC before switching the input
    /* then restart the ADC.
    /******
    /* `#START MAIN_ADC_ISR` */
    result = CY_GET_REG16(ADC_SAR_1_SAR_WRK0_PTR);
    dataReady = 1;
    /* `#END` */
}

```

功能说明

下图显示的是一个框图。对输入模拟信号进行采样，并使用二进制搜索算法将其与 **DAC** 的输出进行比较以确定从 **MSB** 到 **LSB** 的输出位。



寄存器

采样寄存器

ADC 结果可能在 8 至 12 位分辨率之间。输出分为两个 8 位寄存器。CPU 或 DMA 可访问这些寄存器，读取 ADC 结果。

ADC_SAR_WRK0_REG (SAR 工作寄存器 0)

位	7	6	5	4	3	2	1	0
数值	Data[7:0]							

ADC_SAR_WRK1_REG (SAR 工作寄存器 1)

位	7	6	5	4	3	2	1	0
数值	overrun_det	N/A			Data[11:8]			
	NA							

- Data[11:0]: ADC 结果
- overrun_det: 数据过速检测标志。默认情况下，禁用该功能。

资源

ADC_SAR 使用芯片中的固定模块 SAR。

API 存储器的使用情况

根据不同编译器、器件、所使用的 API 数量以及组件的配置情况，组件所用的存储空间大小也不一样。下表提供默认组件配置中可用的 API 的存储器大小。

数据是在将编译器设置为 Release 模式并将优化等级设置为 Size 的情况下测得的。对于特定的设计，分析完编译器生成的映射文件后可以确定存储器的使用情况。

配置	PSoC 3 (Keil_PK51)		PSoC 5LP (GCC)	
	闪存 字节	SRAM 字节	闪存 字节	SRAM 字节
默认值	N/A	N/A	920	14

直流和交流电气特性

下列各值表示 PSoC 5LP 的特性。

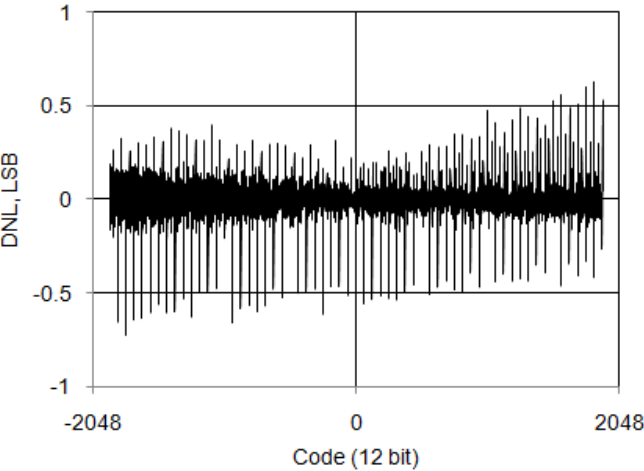
直流规范

参数	说明	条件	最小值	典型值	最大值	单位
	分辨率		8	—	12	位
	通道数量 — 单端		—	—	GPIO 数量	—
	通道数量 — 差分	差分对由一对GPIO组成。	—	—	GPIO 数量/2	—
	单调性 ¹		是	—	—	
Ge	增益误差	外部参考电压	—	—	±0.1	%
Vos	输入偏移电压		—	—	±2	mV
I _{DD}	电流消耗		—	—	1	mA
	输入电压范围 — 单端 ¹		V _{SSA}	—	V _{DDA}	V
	输入电压范围 — 差分 ¹		V _{SSA}	—	V _{DDA}	V
PSRR	电源抑制比 ¹		70	—	—	dB
CMRR	共模抑制比		70	—	—	dB
INL	积分非线性 ¹	VDDA电压范围为1.71至5.5 V，采样率为1 Msps，VREF电压范围为1至5.5 V，在ExtRef引脚旁路。	—	—	+2 /-1.5	LSB
		VDDA电压范围为2.0至3.6 V，采样率为1 Msps，VREF电压范围为2至VDDA，在ExtRef引脚旁路	—	—	±1.2	LSB
		VDDA电压范围为1.71至5.5 V，采样率为500 ksp/s，VREF电压范围为1至5.5 V，在ExtRef引脚旁路	—	—	±1.3	LSB
DNL	微分非线性度 ¹	VDDA的电压范围为1.71至5.5 V，采样率为1 Msps，VREF的电压范围为1至5.5 V，在ExtRef引脚旁路。	—	—	+2 /-1	LSB

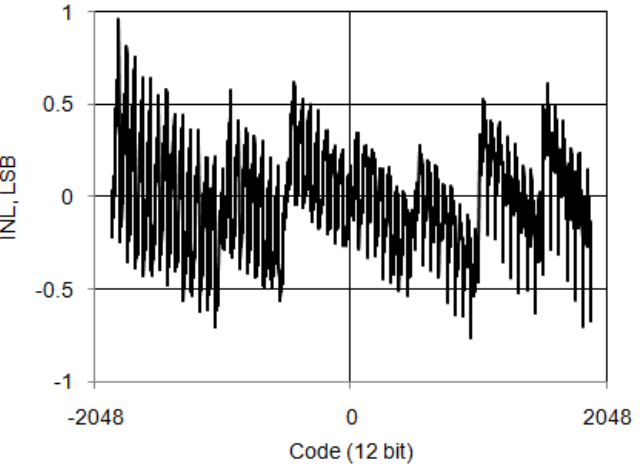
¹ 基于器件特性表征（未经过生产测试）。

参数	说明	条件	最小值	典型值	最大值	单位
		VDDA电压范围为2.0至3.6 V，采样率为1 Msps，VREF的电压范围为2至VDDA，在ExtRef引脚旁路。 无遗漏代码	—	—	1.7/−0.99	LSB
		VDDA的电压范围为1.71至5.5 V，采样率为500 ksp/s，VREF的电压范围为1至5.5 V，在ExtRef引脚旁路。 无遗漏代码	—	—	+2/−0.99	LSB
R _{IN}	输入电阻		—	180	—	kΩ

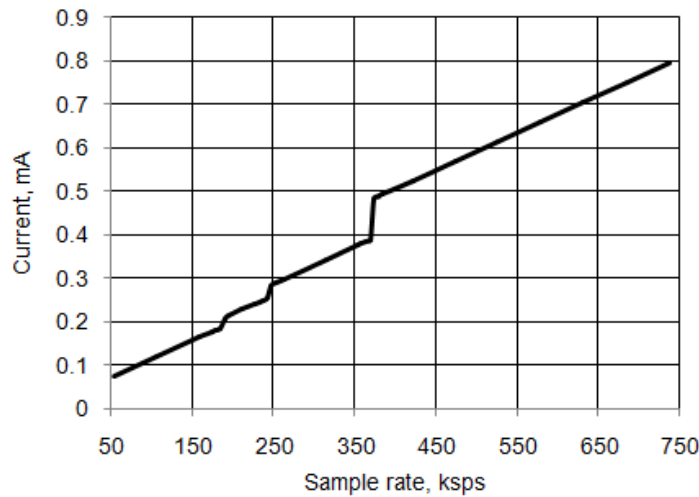
DNL和输出代码，
旁路内部参考模式



DNL和输出代码，
旁路内部参考模式



IDD和sps, VDDA = 5 V, 连续采样模式, 外部参考模式

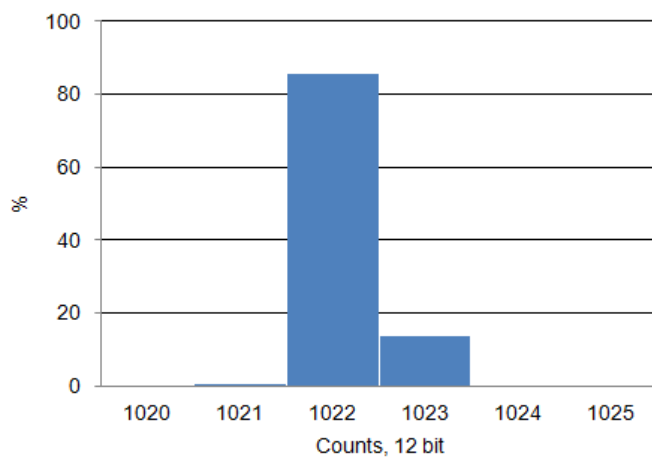


交流规范

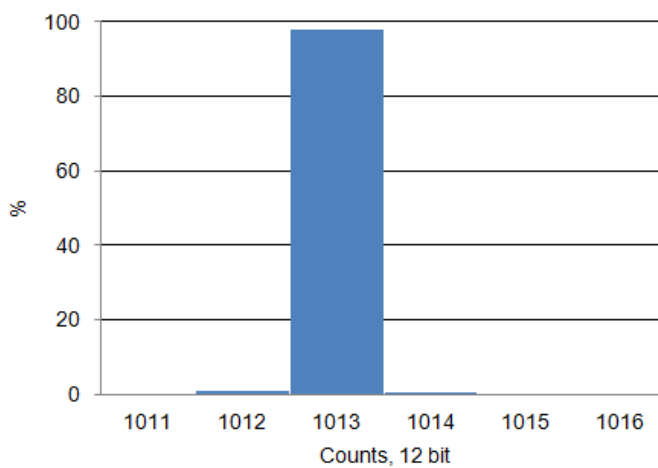
参数	说明	条件	最小值	典型值	最大值	单位
Fclk	SAR时钟频率 ²		1	—	18	Mhz
Tc	转换时间	一次转换需要18个SAR时钟周期。 最大采样率为1 Msps	1	—	18	μs
	启动时间 ²		—	—	10	μs
SINAD	信噪比 ²		68	—	—	dB
THD	总谐波失真 ²		—	—	0.02	%

² 基于器件特性表征（未经过生产测试）。

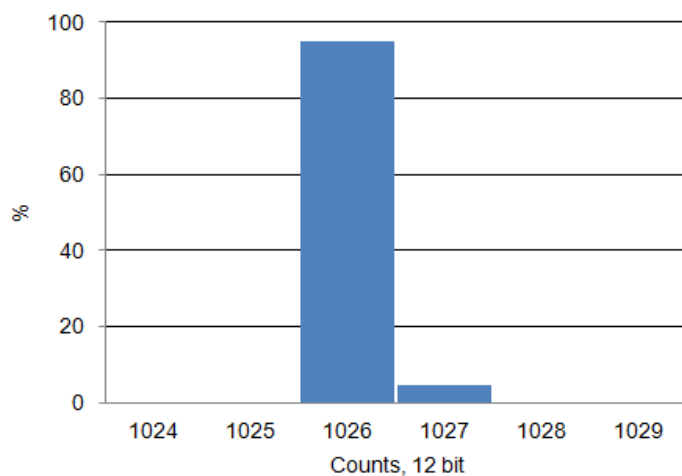
噪声柱状图, 1000个采样, 700 ksps,
无旁路内部参考, $V_{IN} = V_{REF}/2$



噪声柱状图, 1000个采样, 700 ksps,
旁路内部参考, $V_{IN} = V_{REF}/2$



噪声柱状图, 1000个采样, 700 ksps,
外部参考, $V_{IN} = V_{REF}/2$



组件勘误表

本节列出了组件的已知问题。

赛普拉斯ID	组件版本	问题	解决方案
191257	v2.10	对该组件进行修改，但没有更改PSoC Creator 3.0 SP1中的版本号。更多信息，请参考‘基础知识文章 KBA94159’（ www.cypress.com/go/kba94159 ）。	不需要解决方案。不会对设计产生任何影响。

组件更改

本节列出了该组件各版本中的主要更改内容。

版本	更改说明	更改原因/影响
2.10.a	编辑数据手册，以添加组件勘误部分。	文档的组件被更改，但设计不受任何影响。
2.10	修正了ADC_CountsTo_uVolts() API中可能溢出的定点算术。	ADC_SetOffset() API设置带减号的偏移时会发生该问题。
	更新了增益校准，以得到更高的分辨率。 添加了新的全局变量ADC_countsPer10Volt和ADC_SetScaledGain() API函数。这些函数将增益系数设置为12位以上的分辨率。	在Vssa到Vdda输入范围内，该增益系数（ADC_countsPerVolt）为10位分辨率。该问题影响到ADC_CountsTo_Volts()、ADC_CountsTo_mVolts()和ADC_CountsTo_uVolts() API。
2.0	添加了MISRA合规性章节。	该组件没有任何特定偏差。
	将 触发 采样模式重新命名为 软件触发 模式。添加了 硬件触发 采样模式。在 软件触发 采样模式中显示的SOC输入可提供向后兼容。	ADC_StartConvert() API禁用触发模式中的外部SOC输入。为了解决该问题，已添加了 硬件触发 模式。其中ADC_StartConvert()和ADC_StopConvert() API没有实现。
	添加了显示实际转换速率的标签。	根据可用时钟速度和分频器范围的情况，实际转换速率与所需要的速率有所不同。
1.90	添加了可选的EOS输出。	当SAR ADC与输入通道复用器一起使用时，通过该信号可以快速切换复用器。
	在选择“0.0 ± Vdda”作为输入范围时，会在参考下拉列表中隐藏“Internal Vref, bypassed”（内部Vref，旁路）项目。该范围可能有高转换速率，但没有旁路电容。	Vdda参考不需要旁路电容。
1.80	增加对PSoC 5LP芯片的支持。	
	通过编辑数据手册来说明时钟频率。	
1.71	修正了ADC_GetResult8()和ADC_GetResult16() API以执行一个16位读取操作而非两个8位读取。	如果在读取了一个字节之后SAR ADC更新输出采样寄存器，则生成的数据可能会损坏。
	修正了ADC_IsEndConversion() API以等待释放EOF状态位。	此函数可能会在快速连续调用之后返回意外转换完成状态。
1.70	更正了SampleRate错误提供程序消息中的最小值。	
	在选择“VDAC”作为Input Range（输入范围）时，会在Reference（参考）下拉列表中隐藏“External Vref（外部 Vref）”项目。	当选择VDAC范围时，不可使用外部参考。

版本	更改说明	更改原因/影响
	当选择了 External Vref (外部Vref) 选项时, 将外部引脚重命名为 “ExtVref”。当选择了 Internal reference with Bypass (带绕过的内部参考) 选项时, 保留名称 “Bypass”。	用于将引脚名称与功能匹配。
	数据手册纠正	
1.60	从自定义程序中清除了 “Power” (功耗) 参数。	根据时钟速度自动设置较高功耗。 ADC_SetPower() API说明包含功耗与时钟速率的关系。
	SAR 在12位模式下运行。8位和10位选项会保留, 但是仅影响 ADC_GetResult16() API。	在8位和10位模式下, SAR ADC 仅将 ODD 计数显示为输出。
	将默认 SAR 转换速率从1 Msps更改为631579 sps (12 MHz时钟)。	SAR 应能够用默认设置进行放置和构建。
	ADC_Stop() API不会关闭 ADC 的电源, 但是会将功耗减少到最小值。	PSoC 5 芯片有一个缺陷, 导致与某些模拟资源的连接在未通电时不可靠。
	将转换时间从18个周期更改为19个周期。	用于提高 SAR 性能。
1.50.a	添加了时钟频率验证。	此更改提供一种方式避免使用带有不规范时钟的 SAR ADC 。 如果从版本1.10的 SAR ADC 组件更新并使用工作范围之外的时钟, 请选择正确时钟频率。
	向组件中添加了消息, 以说明它与芯片修订版的兼容性。	如果组件在不兼容的芯片上使用, 该工具将报告错误/警告。如果发生此情况, 请更新所支持您目标器件的修订版。
	对数据手册进行了少量编辑及更新	
1.50	添加了睡眠/唤醒和初始化/使能API。	用于支持低功耗模式并提供常用接口, 以单独控制大多数组件的初始化和使能。
	添加了 ADC_CountsTo_Volts 和 ADC_CountsTo_uVolts API。	扩展功能。此API以伏和毫伏为单位返回转换结果。
	向组件中添加了DMA功能文件。	此文件允许 ADC_SAR 在 PSoC Creator 中受DMA向导工具的支持。
	在 ADC_GetResult8 和 ADC_GetResult16 API中实现了 ADC 计数到2的补码形式的转换。 从 ADC_CountsTo_mVolts 函数中清除了相同内容。	进行此更改是为了与 ADC DelSig 一致。

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，WICED，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。

