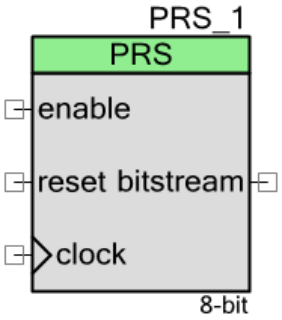
伪随机序列（PRS）

2.40



特性

- 2 至 64 位 PRS 序列长度
- 时分复用模式
- 串行输出比特流
- 连续或单步运行模式
- 标准或自定义多项式
- 标准或自定义种子值
- 使能输入提供了与其他组件的同步操作
- 可以从线性反馈移位寄存器（LFSR）直接读取计算后得出的伪随机数字

概述

伪随机序列（PRS）组件使用 LFSR 生成伪随机序列，由此输出一个伪随机比特流。LFSR 是 Galois 格式（有时也称为模块格式），并使用所提供的最大代码长度或周期。使能输入保持在高电平时，PRS 组件便可以在启动后连续运行。使用除 0 以外的任意有效种子值，可以启动 PRS 数字发生器。

何时使用 PRS

LFSR 可以在硬件中实现。这使得 LFSR 在要求非常快速地生成伪随机序列的应用中非常有用，例如，直接序列扩频无线电。

全球定位系统使用 LFSR 快速传输相对时间偏移的高精度序列。此外，某些视频游戏控制器也将 LFSR 用作音响系统的一部分。

用作计数器

可以接受非二进制序列时，通过 LFSR 状态的重复序列，将其用作分频器或计数器。与自然二进制计数器或格雷码计数器相比，LFSR 计数器具有简单的反馈逻辑，因此可以在较高的时间频率下

运行。然而，必须确保 LFSR 从未进入全零状态，例如，在启动时，通过将其预设为序列中其他任何状态。

输入/输出接口

本节介绍 PRS 组件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 配置中被隐藏。

时钟 — 输入*

时钟输入定义要计算 PRS 的信号。当选择 API 单步运行模式时，此输入不可用。

复位 — 输入*

复位输入用来定义要同步复位 PRS 的信号。此输入在选择时钟模式时不可用。只有使能输入保持在高电平时，才能复位 PRS。

使能 — 输入

使能输入保持在高电平时，PRS 组件便可以在启动后运行。此输入提供与其他组件的同步操作。

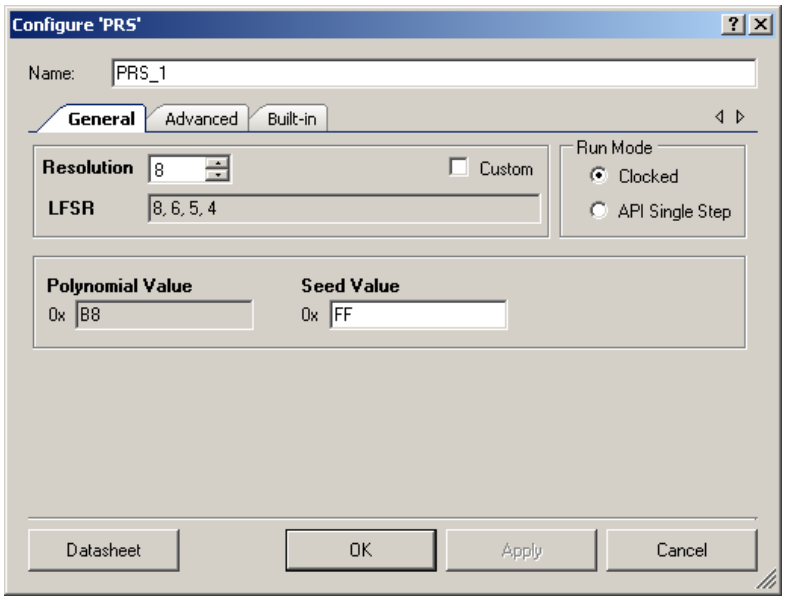
bitstream — 输出

LFSR 的输出。

组件参数

将 PRS 组件拖入设计窗口中，双击该组件，打开 **Configure**（配置）对话框。该对话框有若干选项卡，可引导您完成 PRS 组件的设置过程。

General 选项卡



Resolution（分辨率）

用来定义 PRS 序列的长度。该值设置介于 2-64 之间。默认值为 **8**。

默认情况下，**Resolution** 用来定义 **LFSR** 系数和 **Polynomial Value**（多项式值）。这些系数取自下表。此外，此参数还用来定义最大代码长度或周期，如下表所示。

分辨率	LFSR	周期（$2^{Resolution} – 1$）
2	2、1	3
3	3、2	7
4	4、3	15
5	5、4、3、2	31
6	6、5、3、2	63

分辨率	LFSR	周期（$2^{Resolution} – 1$）
34	34、31、30、26	17179869183
35	35、34、28、27	34359738367
36	36、35、29、28	68719476735
37	37、36、33、31	137438953471
38	38、37、33、32	274877906943

分辨率	LFSR	周期（$2^{Resolution} – 1$）
7	7、6、5、4	127
8	8、6、5、4	255
9	9、8、6、5	511
10	10、9、7、6	1023
11	11、10、9、7	2047
12	12、11、8、6	4095
13	13、12、10、9	8191
14	14、13、11、9	16383
15	15、14、13、11	32767
16	16、14、13、11	65535
17	17、16、15、14	131071
18	18、17、16、13	262143
19	19、18、17、14	524187
20	20、19、16、14	1048575
21	21、20、19、16	2097151
22	22、19、18、17	4194303
23	23、22、20、18	8388607
24	24、23、21、20	16777215

分辨率	LFSR	周期（$2^{Resolution} – 1$）
39	39、38、35、32	549755813887
40	40、37、36、35	1099511627775
41	41、40、39、38	2199023255551
42	42、40、37、35	4398046511103
43	43、42、38、37	8796093022207
44	44、42、39、38	17592186044415
45	45、44、42、41	35184372088831
46	46、40、39、38	70368744177663
47	47、46、43、42	140737488355327
48	48、44、41、39	281474976710655
49	49、45、44、43	562949953421311
50	50、48、47、46	1125899906842623
51	51、50、48、45	2251799813685247
52	52、51、49、46	4503599627370495
53	53、52、51、47	9007199254740991
54	54、51、48、46	18014398509481983
55	55、54、53、49	36028797018963967
56	56、54、52、49	72057594037927935

分辨率	LFSR	周期（$2^{Resolution} – 1$）
25	25、24、23、22	33554431
26	26、25、24、20	67108863
27	27、26、25、22	134217727
28	28、27、24、22	268435455
29	29、28、27、25	536870911
30	30、29、26、24	1073741823
31	31、30、29,、28	2147483647
32	32、30、26、25	4294967295
33	33、32、29、27	8589934591

分辨率	LFSR	周期（$2^{Resolution} – 1$）
57	57、55、54、52	144115188075855871
58	58、57、53、52	288230376151711743
59	59、57、55、52	576460752303423487
60	60、58、56、55	1152921504606846975
61	61、60、59、56	2305843009213693951
62	62、59、57、56	4611686018427387903
63	63、62、59、58	9223372036854775807
64	64、63、61、60	18446744073709551615

要手动设置 LFSR 系数：

1. 定义 **Resolution**。
2. 选中 **Custom**（自定义）选框。
3. 在 **LFSR** 文本框中输入系数，用逗号分隔，然后按下**[Enter]**。系统将自动重新计算多项式值。

多项式值显示为十六进制格式。

注意：LFSR 系数的值不能大于 **Resolution** 值。

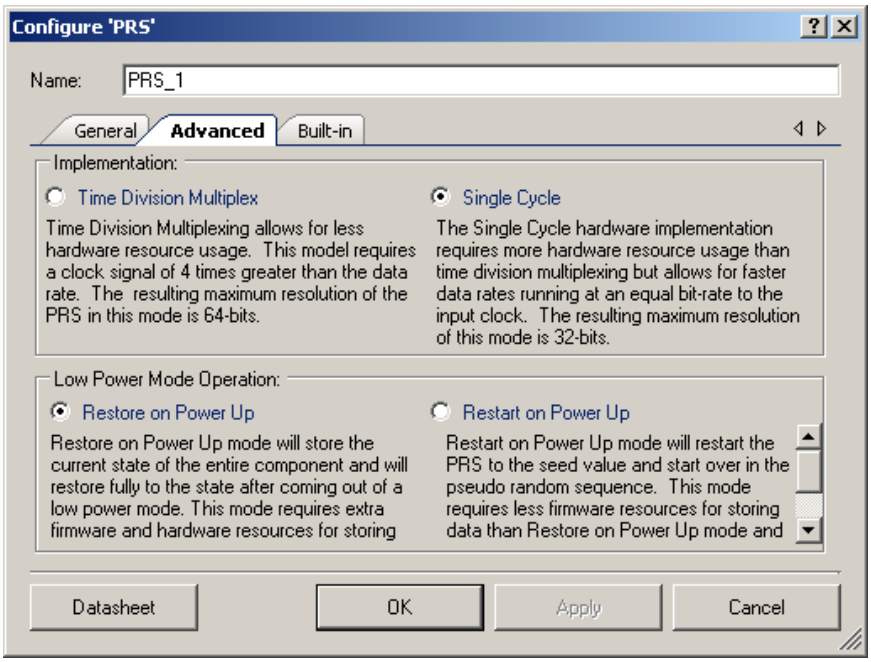
默认情况下，“Seed value”（种子值）设置为最大可能值（$2^{Resolution} – 1$）。此值可修改为除 0 以外的任何值。种子值显示为十六进制格式。

注意：更改 **Resolution** 值会使 **Seed Value** 复位到默认值。

Run Mode（运行模式）

此参数用于将组件运行模式定义为连续运行或单步运行模式。您可以选择 **Clocked**（时钟）（默认值）或 **API Single Step**（API 单步执行）项。如果连续读取 PRS 的值，或需要读取一个值，则必须停止时钟或在时钟模式下将使能设置为低电平。

Advanced 选项卡



PRS **Advanced** 选项卡包含以下设置：

Implementation（实现）

用来定义 PRS 组件的实现：包含或不包含时间复用（**Single Cycle**（单周期））。默认值为 **Single Cycle**。

Low Power Mode Operation（低功耗模式运行）

用来定义低功耗模式下的 PRS 行为。默认值为 **Restore on Power Up**（上电时存储）。

本地参数（供 API 使用）

以下参数用于 API，并不在 GUI 中显示：

- **PolyValueLower(uint32)** — 包含多项式值的下半部分（十六进制格式）。默认值为 0xB8h（**LFSR**= [8,6,5,4]），因为默认分辨率为 8。
- **PolyValueUpper(uint32)** — 包含多项式值的上半部分（十六进制格式）。默认值为 0x00h，因为默认分辨率为 8。
- **SeedValueLower (uint32)** — 包含种子值的下半部分（十六进制格式）。默认值为 0xFFh，因为默认分辨率为 8。

- **SeedValueUpper (uint32) —** 包含种子值的上半部分（十六进制格式）。默认值为 0，因为默认分辨率为 8。

时钟选择

如果选择 **Run Mode**（运行模式）参数的 **Clocked**（时钟）选项，则必须连接时钟源。

注意：如果为 **Implementation**（实现）参数选择 **Time Division Multiplex**（时间分频复用），想要生成分辨率大于 8 的正确 PRS 序列，则要求时钟信号比数据速率大 4 倍。

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。下面的表格列出并介绍了每个函数的接口。以下各节将对每个函数加以说明。

默认情况下，PSoC Creator 将实例名称“PRS_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、变量和常量符号的前缀。出于可读性考虑，下表中使用的实例名称为“PRS”。

函数	说明
PRS_Start()	初始化由自定义程序提供的种子和多项式寄存器。在输入时钟上升沿启动PRS计算功能。
PRS_Stop()	停止PRS计算功能。
PRS_Sleep()	停止PRS计算功能并保存PRS配置。
PRS_Wakeup()	恢复PRS配置，并在输入时钟上升沿启动PRS计算功能。
PRS_Init()	用初始值初始化种子和多项式寄存器。
PRS_Enable()	在输入时钟上升沿启动PRS计算功能。
PRS_SaveConfig()	保存种子和多项式寄存器。
PRS_RestoreConfig()	恢复种子和多项式寄存器。
PRS_Step()	使用API单步执行模式时，PRS增加 1。
PRS_WriteSeed()	写入种子值。
PRS_WriteSeedUpper()	写入种子值的上半部分。仅为33-64位PRS而生成。
PRS_WriteSeedLower()	写入种子值的下半部分。仅为33-64位PRS而生成。
PRS_Read()	读取PRS值。
PRS_ReadUpper()	读取上半部分PRS值。仅为33-64位PRS而生成。

函数	说明
PRS_ReadLower()	读取下半部分PRS值。仅为33-64位PRS而生成。
PRS_WritePolynomial()	写入PRS多项式值。
PRS_WritePolynomialUpper()	写入上半部分PRS多项式值。仅为33-64位PRS而生成。
PRS_WritePolynomialLower()	写入下半部分PRS多项式值。仅为33-64位PRS而生成。
PRS_ReadPolynomial()	读取PRS多项式值。
PRS_ReadPolynomialUpper()	读取上半部分PRS多项式值。仅为33-64位PRS而生成。
PRS_ReadPolynomialLower()	读取下半部分PRS多项式值。仅为33-64位PRS而生成。

全局变量

变量	说明
PRS_initVar	表示是否完成初始化PRS。变量将初始化为0，并在第一次调用PRS_Start()时设置为1。这样，第一次调用PRS_Start()子程序后，组件不用重新初始化即可重启。 如需重新初始化组件，可在PRS_Start()或PRS_Enable()函数前调用PRS_Init()函数。

void PRS_Start(void)

说明： 初始化种子和多项式寄存器。在输入时钟上升沿启动PRS计算功能。

参数： 无

返回值： 无

其他影响： 无

void PRS_Stop(void)

说明： 停止PRS计算功能。

参数： 无

返回值： 无

其他影响： 无

void PRS_Sleep(void)

说明： 停止PRS计算功能，然后保存PRS配置。

参数： 无

返回值： 无

其他影响： 无

void PRS_Wakeup(void)

说明： 恢复PRS配置，并在输入时钟上升沿启动PRS计算功能。

参数： 无

返回值： 无

其他影响： 无

void PRS_Init(void)

说明： 用初始值初始化种子和多项式寄存器。

参数： 无

返回值： 无

其他影响： 无

void PRS_Enable(void)

说明： 在输入时钟上升沿启动PRS计算功能。

参数： 无

返回值： 无

其他影响： 无

void PRS_SaveConfig(void)

说明： 保存种子和多项式寄存器。

参数： 无

返回值： 无

其他影响： 无

void PRS_RestoreConfig(void)

说明： 恢复种子和多项式寄存器。

参数： 无

返回值： 无

其他影响： 无

void PRS_Step(void)

说明： 使用API单步执行模式时，PRS增加1。

参数： 无

返回值： 无

其他影响： 无

void PRS_WriteSeed(uint8/16/32 seed)

说明： 写入种子值。

参数： uint8/16/32 seed：种子值

返回值： 无

其他影响： 根据 mask（掩码）= $2^{Resolution} – 1$进行截取种子值。例如，如果PRS分辨率为14位，则掩码值为：掩码 = $2^{14} – 1$ = 0x3FFFu。截取种子值 = 0xFFFFu：种子 AND 掩码 = 0xFFFFu AND 0x3FFFu = 0x3FFFu。

void PRS_WriteSeedUpper(uint32 seed)

说明： 写入种子值的上半部分。仅为33-64位PRS而生成。

参数： uint32 seed：种子值的上半部分

返回值： 无

其他影响： 根据掩码 = $2^{(Resolution – 32)} – 1$截取上半部分种子值。

例如，如果PRS分辨率为35位，则掩码值为：

$2^{(35 – 32)} – 1 = 2^{\wedge 3} – 1$ = 0x0000 0007u。

截取上半部分种子值 = 0x0000 00FFu：

上半部分种子AND掩码 = 0x0000 00FFu AND 0x0000 0007u = 0x0000 0007u。

void PRS_WriteSeedLower(uint32 seed)

说明： 写入种子值的下半部分。仅为33-64位PRS而生成。

参数： uint32 seed：种子值的下半部分

返回值： 无

其他影响： 无

uint8/16/32 PRS_Read(void)

说明： 读取PRS值。

参数： 无

返回值： uint8/16/32：返回PRS值。

其他影响： 无

uint32 PRS_ReadUpper(void)

说明： 读取上半部分PRS值。仅为33-64位PRS而生成。

参数： 无

返回值： uint32：返回上半部分PRS值。

其他影响： 无

uint32 PRS_ReadLower(void)

说明： 读取下半部分PRS值。仅为33-64位PRS生成

参数： 无

返回值： uint32：返回下半部分PRS值。

其他影响： 无

void PRS_WritePolynomial(uint8/16/32 polynomial)

说明： 写入PRS多项式值。

参数： uint8/16/32 polynomial：PRS多项式值。

返回值： 无

其他影响： 根据掩码 = $2^{Resolution} – 1$截取多项式值。
例如，如果PRS分辨率为14位，则掩码值为：掩码 = $2^{14} – 1$ = 0x3FFFu。
截取多项式值 = 0xFFFFu：
多项式 AND 掩码 = 0xFFFFu AND 0x3FFFu = 0x3FFFu。

void PRS_WritePolynomialUpper(uint32 polynomial)

说明： 写入上半部分PRS多项式值。仅为33-64位PRS而生成。

参数： uint32 polynomial：上半部分PRS多项式值。

返回值： 无

其他影响： 根据掩码 = $2^{(Resolution – 32)} – 1$截取上半部分多项式值。
例如，如果PRS分辨率为35位，则掩码值为：
$2^{(35 – 32)} – 1 = 2^3 – 1$ = 0x0000 0007u。
多项式值的上半部分 = 0x0000 00FFu 被截取：
上半部分多项式AND掩码 = 0x0000 00Ffu AND 0x0000 0007u = 0x0000 0007u。

void PRS_WritePolynomialLower(uint32 polynomial)

说明： 写入下半部分PRS多项式值。仅为33-64位PRS而生成。

参数： uint32 polynomial：下半部分PRS多项式值。

返回值： 无

其他影响： 无

uint8/16/32 PRS_ReadPolynomial(void)

说明： 读取PRS多项式值。

参数： 无

返回值： uint8/16/32：返回PRS多项式值。

其他影响： 无

uint32 PRS_ReadPolynomialUpper(void)

说明： 读取上半部分PRS多项式值。仅为33-64位PRS而生成。

参数： 无

返回值： uint32：返回上半部分PRS多项式值。

其他影响： 无

uint32 PRS_ReadPolynomialLower(void)

说明： 读取下半部分PRS多项式值。仅为33-64位PRS而生成。

参数： 无

返回值： uint32：返回下半部分PRS多项式值。

其他影响： 无

MISRA 合规性

本节介绍了MISRA-C:2004合规性和本组件的偏差情况。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 — 仅适用于该组件的偏差

本节介绍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。

PRS 组件具有以下特定偏差：

MISRA-C:2004 规则	规则类别（必须/建议）	规则说明	偏差说明
19.7	A	函数应该优先于类似于函数的宏。	宏PRS_IS_PRS_ENABLE导致偏差。如果删除该宏，会导致直接使用该宏的设计失败。

样例固件源代码

在“Find Example Project”对话框中，PSoC Creator 提供了大量的示例项目，包括原理图和代码的。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》中主题为“查找示例项目”中的内容。

功能说明

PRS 运行模式：时钟

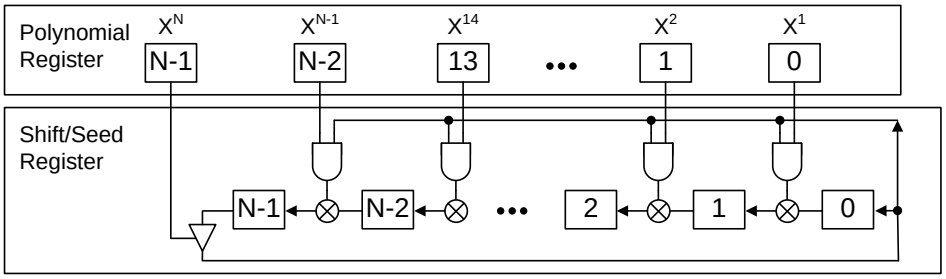
在此模式下，只要使能输入保持在高电平时，PRS 组件便可以在启动后连续运行。

PRS 运行模式：API 单步执行

在此模式下，通过 API 调用增加 PRS。

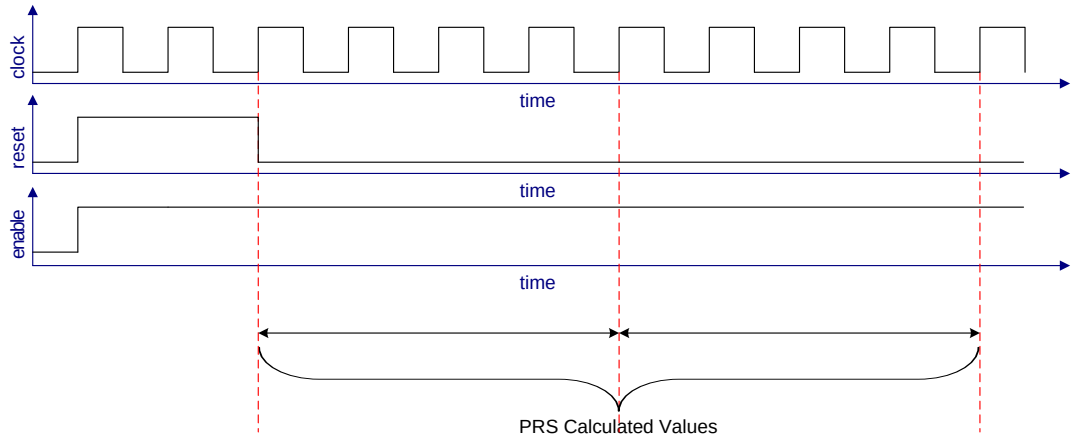
框图和配置

PRS 作为一组 UDB 配置得以实现。在以下框图中显示该实现。

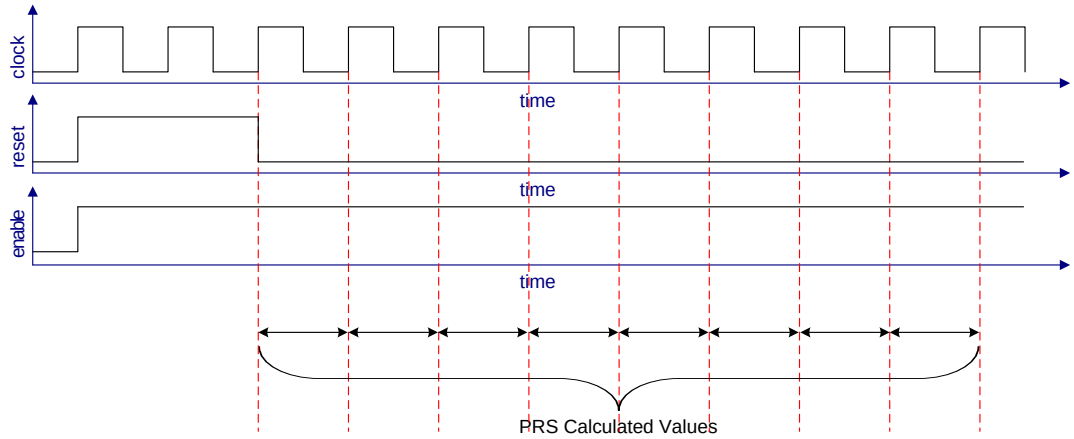


时序图

时分复用实现模式



单周期实现模式



寄存器

多项式寄存器（根据分辨率，从 2 位到 64 位）

多项式寄存器包含多项式值。通过 PRS_WritePolynomial()、PRS_WritePolynomialUpper()或 PRS_WritePolynomialLower()函数，可以对该值进行更改。此外，还可以使用 PRS_ReadPolynomial()、PRS_ReadPolynomialUpper()或 PRS_ReadPolynomialLower()读取当前多项式值。

移位/种子寄存器（根据寄存器，从 2 位到 64 位）

移位/种子寄存器包含种子值。可以使用 PRS_WriteSeed()、PRS_WriteSeedUpper()或 PRS_WriteSeedLower()函数，可以对该值进行更改。此外，还可以使用 PRS_ReadSeed()、PRS_ReadSeedUpper()或 PRS_ReadSeedLower()读取当前种子值。

资源

PRS 组件放置在整个 UDB 阵列中。该组件利用以下资源。

配置	资源类型					
	Datapath 单元	宏单元[1]	状态单元	控制单元	DMA通道	中断
8位单周期	1	1	–	1	–	–
16位单周期	2	1	–	1	–	–
24位单周期	3	1	–	1	–	–
32位单周期	4	1	–	1	–	–
16位时分	1	9	1	1	–	–
24位时分	2	10	1	1	–	–
32位时分	2	9	1	1	–	–
40位时分	3	10	1	1	–	–
48位时分	3	9	1	1	–	–
56位时分	4	10	1	1	–	–
64位时分	4	9	1	1	–	–

1. 使用 API 单步运行模式为单周期实现使用其他宏。

API 存储器大小

根据编译器、器件、所使用的 API 数量以及组件的配置情况不同，组件的存储器大小也不一样。下表提供了组件配置中所有 API 占用的存储器大小。

通过使用“释放”模式下相应的编译器，可以完成测量操作。在该模式下，存储器的大小得到优化。对于特定的设计，分析编译器生成映射文件后可以确定存储器的使用大小。

配置	PSoC 3 （Keil_PK51）		PSoC 4（GCC）		PSoC 5LP（GCC）	
	闪存 字节	SRAM 字节	闪存 字节	SRAM 字节	闪存 字节	SRAM 字节
8位单周期	170	3	274	5	284	5
16位单周期	232	4	296	5	306	5
24位单周期	304	6	356	9	338	9
32位单周期	302	6	312	9	326	9
16位时分	306	6	424	9	434	9
24位时分	595	8	508	13	510	13
32位时分	671	8	536	13	546	13
40位时分	842	12	696	17	730	17
48位时分	977	12	748	17	804	17
56位时分	1083	12	824	17	880	17
64位时分	1175	12	850	17	864	17

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 和 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。

直流电气特性

参数	说明	最小值	典型值[2]	最大值	单位
I_{DD}	组件电流消耗				
	8位单周期	–	11	–	µA/MHz
	16位单周期	–	17	–	µA/MHz
	24位单周期	–	23	–	µA/MHz
	32位单周期	–	31	–	µA/MHz
	16位时分	–	22	–	µA/MHz
	24位时分	–	30	–	µA/MHz
	32位时分	–	31	–	µA/MHz
	40位时分	–	40	–	µA/MHz
	48位时分	–	41	–	µA/MHz
	56位时分	–	51	–	µA/MHz
	64位时分	–	47	–	µA/MHz

交流电气特性

参数	说明	最小值	典型值	最大值[3]	单位
f_{CLOCK}	组件时钟频率				
	8位单周期	–	–	39	MHz
	16位单周期	–	–	33	MHz
	24位单周期	–	–	30	MHz
	32位单周期	–	–	29	MHz
	16位时分	–	–	29	MHz

2. 未包括器件的 IO 和时钟分配的电流。这些值是在温度为 25 °C 时的值。
3. 这些值提供了此组件的最大安全工作频率。可以在更高的时钟频率运行器件，在该频率将需要使用 STA 结果验证时序要求。

参数	说明	最小值	典型值	最大值[3]	单位
	24位时分	–	–	22	MHz
	32位时分	–	–	28	MHz
	40位时分	–	–	24	MHz
	48位时分	–	–	28	MHz
	56位时分	–	–	24	MHz
	64位时分	–	–	26	MHz

组件更改

本节列出了各版本的主要组件更改内容。

版本	更改说明	更改原因/影响
2.40	已添加了MISRA合规性章节。	此组件具有特定的描述偏差。
2.30	更新了数据手册中有关PSoC 4存储器大小的内容。 更新了对PSoC 4的[25 – 32]位单周期实现模式的定义。	
2.20	添加了“MISRA 合规性”章节	尚未根据MISRA合规性验证此组件
2.10	添加了PSoC 5LP支持	
	向.cyre文件中包括的所有API添加了CYREENTRANT关键词。	并非所有API都是真正可重入的函数。组件API源文件中的注释指出了适用的函数。 对于采用了安全方式并且是不可重入的函数，则需要通过标志或关键段防止同时调用的方式来消除编译器警告。
2.0.b	更新了数据表中的资源信息	
2.0.a	向数据手册中添加了特性数据	
	对数据手册进行了少量编辑和更新	

版本	更改说明	更改原因/影响
2.0	添加了对PSoC 3 Production的芯片的支持。更改包括： ■ 添加了时分复用实现的 4x 时钟 ■ 1x 时钟上现可用 1 到 32 位的单周期实现。 ■ 4x 时钟上现可用 9 到 64 位的时分复用实现。 ■ 补充了同步输入信号复位。 ■ 补充了同步输入信号使能。 ■ 在 Configure（配置）对话框新增“Implementation”与“Low Power Mode”参数的‘Advanced’（‘高级’）页面	新要求支持PSoC 3 Production器件，由此创建了PRS组件新的2.0版本
	补充了PRS_Sleep()/PRS_Wakeup()和PRS_Init()/PRS_Enable() API。	为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。
	更新了函数PRS_WriteSeed()和PRS_WriteSeedUpper()。	掩码参数用于在写入时截取种子值以定义分辨率。
	为下面的多项式写入函数添加了复位DFF触发器：PRS_WritePolynomial()、PRS_WritePolynomialUpper()以及PRS_WritePolynomialLower()。	开始计算前，必须在正确状态下设置DFF触发（多项式的最高有效位始终为1）。要满足此条件，任何写入种子或多项式寄存器中的值均可以复位DFF触发。
	更新了Configure（配置）对话框以用来支持某些参数的“表达式查看”。	“Expression View”（表达式视图）用于直接访问符号参数。此视图允许您用外部参数连接组件参数（如果需要）。
	已更新“Configure”（配置）对话框，以为各种参数添加错误图标。	如果在文本框中输入了错误的值，将显示错误图标以及问题说明的工具提示。这种方法比单独提供错误消息更方便。

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。