模拟开关电容 PSoC 模块数据表 SCBLOCK V 2.4

资源	PSoC® 模块			API 内存 （字节）		引脚 （每个外部 I/O）
	数字	模拟 CT	模拟 SC	闪存	RAM	
CY8C29/27/26/25/24/23/22xxx、CY8CLED04/08/16、CY8CLED0xD、CY8CLED0xG、CY8CTST120、CY8CTMG120、CY8CTMA120、CY8C28x45、CY8CPLC20、CY8CLED16P01、CY8C28x43、 CY8C28x52						
	0	0	1	20	0	

如需一个或多个使用此用户模块的完全配置的功能性示例工程，请转到 www.cypress.com/psocexampleprojects.

特性与概述

- 对自定义开发完全参数化
- 原型的自定义模块
- 可选功耗设置

SCBLOCK 用户模块是完全参数化的模拟开关电容 (SC) PSoC 模块。 它可以创建自定义开关电容函数。 SCBLOCK 电源管理包括应用程序编程接口 (API) 。

对 CY8C26/25xxx 系列的器件，SCBLOCK 可能由 “A” 型或 “B” 型 SC PSoC 模块组成。 对于其他 PSoC 器件，SCBLOCK 可能由 “C” 型或 “D” 型 SC PSoC 模块组成。

类型如下所示：

Figure 1. SCBLOCK A 型 (ASA) 框图

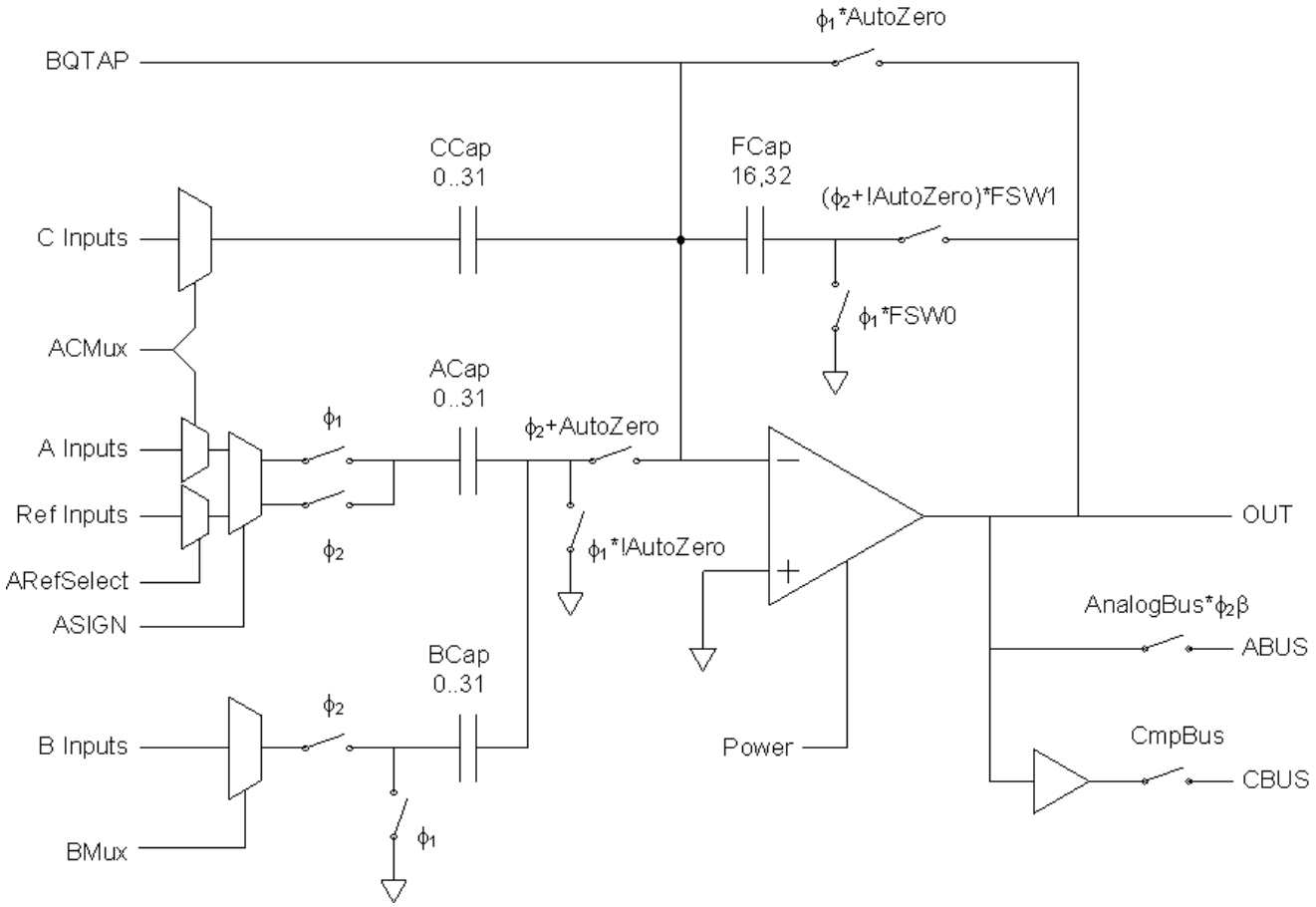


Figure 2. SCBLOCK B 型 (ASB) 框图

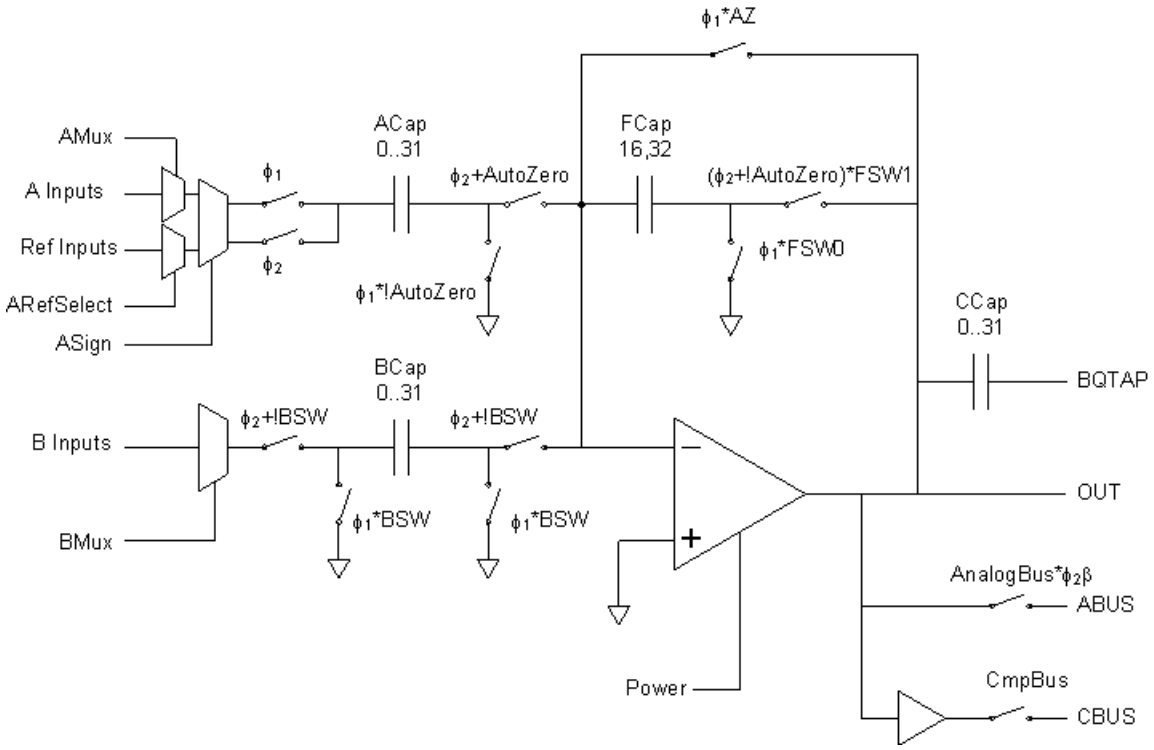


Figure 3. SCBLOCK C 型 (ASC) 框图

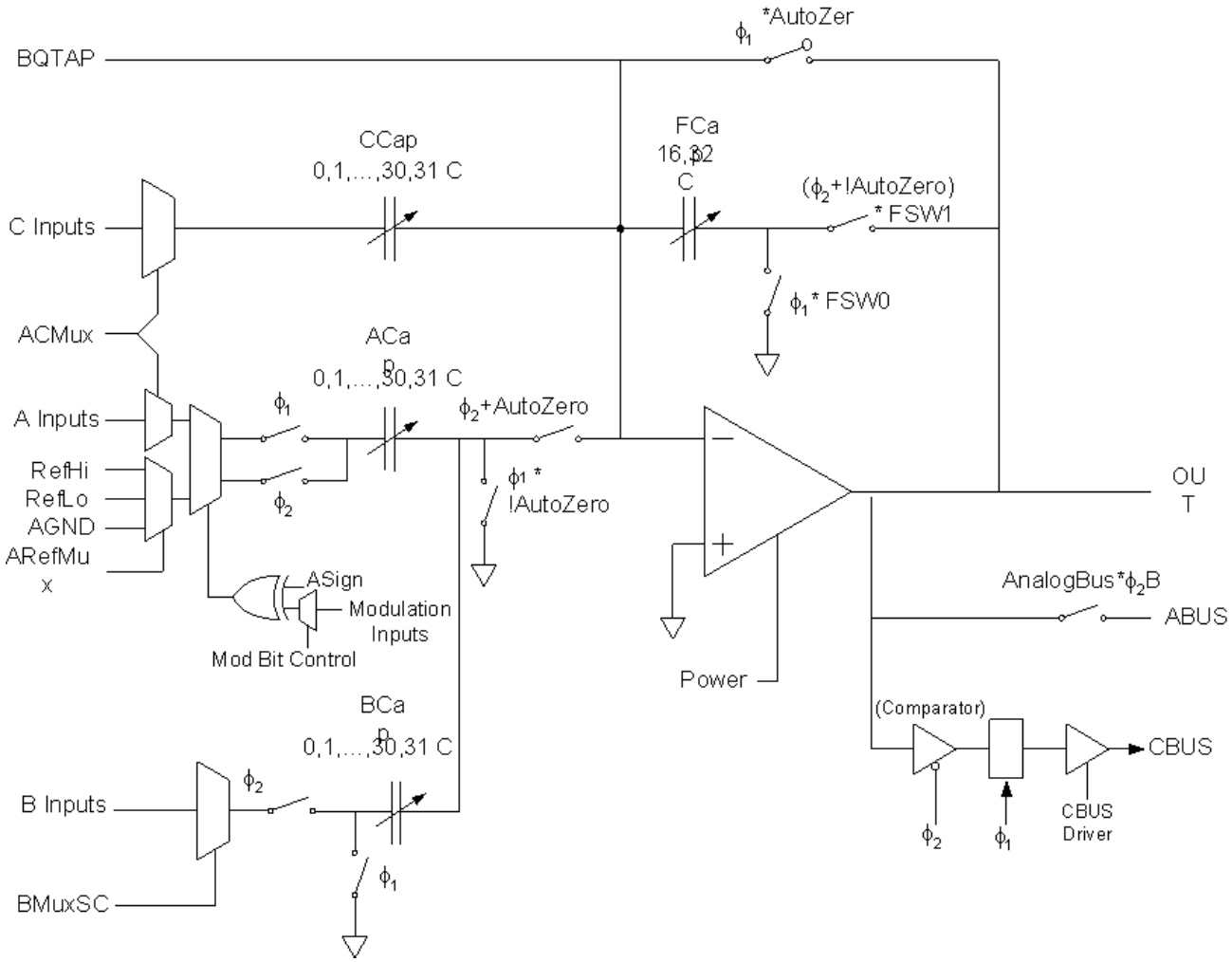
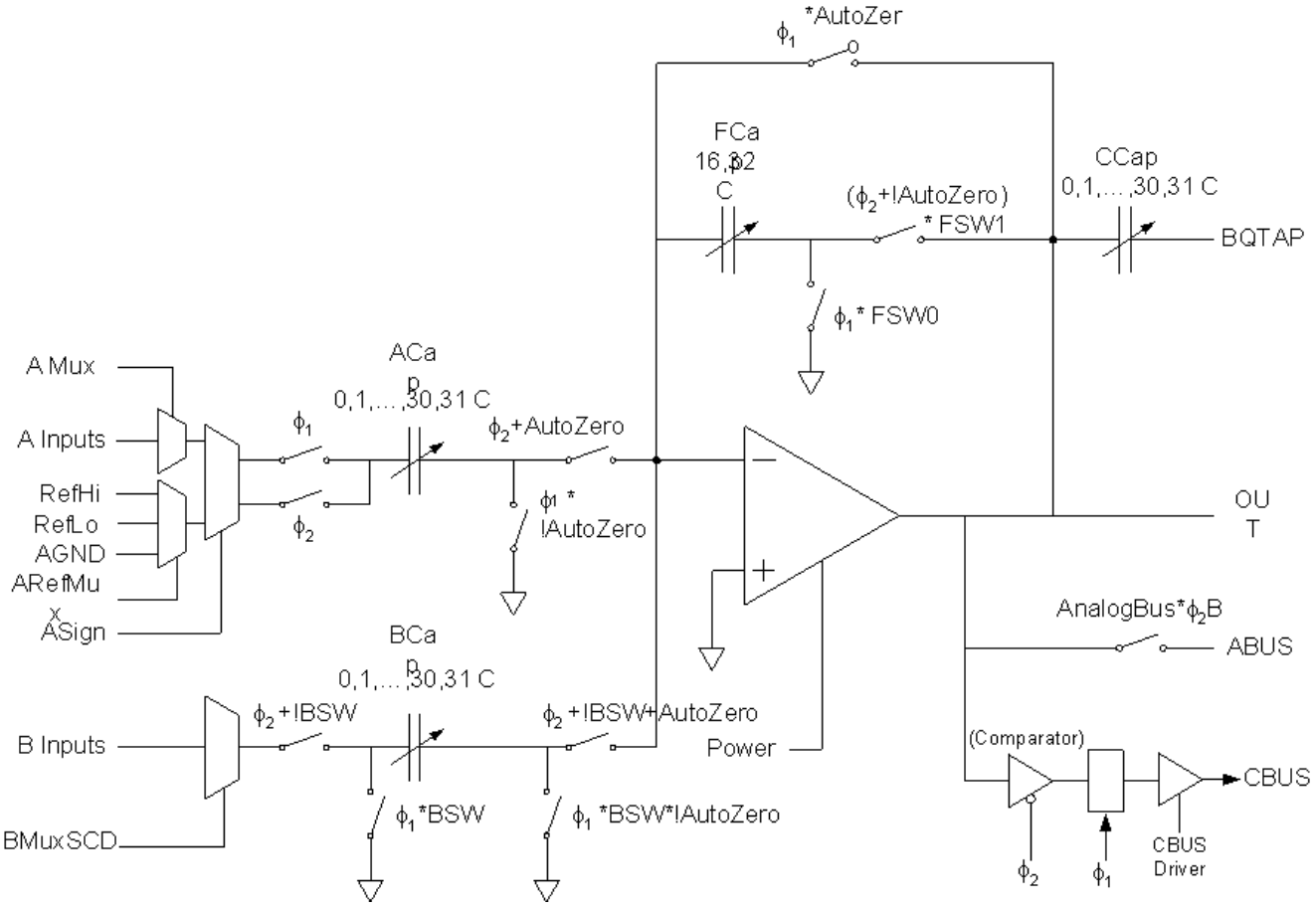


Figure 4. SCBLOCK D 型 (ASD) 框图



功能说明

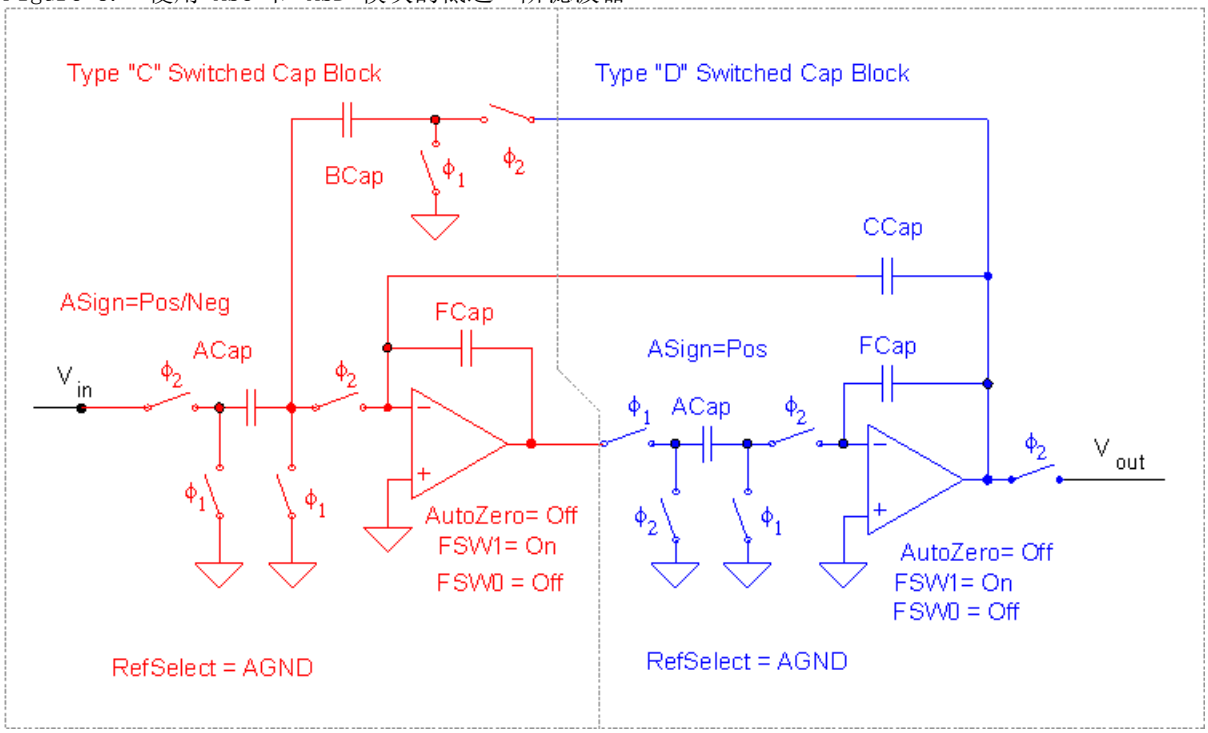
有关 PSoC 开关电容模块的完整描述信息，请参见*应用笔记 AN2041*。 该用户模块的目的是通过使用开关电容模拟 PSoC 模块来完成自定义模拟函数的设计。

ASC 和 ASD 类型的 SCBLOCK 用户模块之间有几个区别：

- CCap 分支在 ASC 类型 SCBLOCK 中作为输入使用，而在 ASD 类型 SCBLOCK 中作为输出使用。
- ASC 类型 SCBLOCK 允许 BCap 的正常自动归零操作。
- ASC 类型 SCBLOCK 有三个可选输入，而 ASD 类型 SCBLOCK 只有两个可选输入。

设计 ASC 和 ASD SCBLOCK 旨在结合二者创建一个二阶滤波器。 ASC 模块用于二阶滤波器的输入段，而 ASD 模块用于输出段。 下图展示了这两个用户模块，ASC 和 ASD SCBLOCK 用户模块，如何连接构成一个低通二阶滤波器。 ASC 和 ASD 类型 SCBLOCK 是 ASA 和 ASB 类型 SCBLOCK 的升级版本。 ASC 和 ASD 类型 SCBLOCK 有更多的输入选择选项，并且运算放大器具有改进的电气特性。

Figure 5. 使用 ASC 和 ASD 模块的低通二阶滤波器



直流和交流电气特性

除非下表中另外指定，否则所有的限制保证 T_A = -40° C 至 +85° C, V_{dd} = 5.0V +/- 5%。

Table 1. SCBLOCK 直流和交流电气特性

参数	条件和注释	典型值	限值	单位
电容值		70		fF
电容匹配		0.1		%
F_{max}	SCBLOCK[1] 中的时钟	2[1]		MHz
供电电流				
偏压 = 低		125		uA
偏压 = 中		280		uA
偏压 = 高		760		uA

电气特性注释

1. 时钟信号通过 1:4 分频器在模拟时钟复用器和 SCBLOCK 之间传递。 要给 SCBLOCK 提供一个 2 MHz 时钟，必须将 8 MHz 时钟连接到模拟时钟复用器。

放置

SCBLOCK 模块可以放在开关电容 PSoC 模块的任一位置。 ASC 和 ASD 类型模块之间的差异取决于放置。

参数和资源

本节讨论 SCBLOCK 的参数。 请注意，一些参数为 ASC 类型 SCBLOCK 用户模块或者 ASD 类型 SCBLOCK 用户模块所特有。

FCap

- **16** - Cap 设置为 16 单位
- **32** - Cap 设置为 32 单位

ClockPhase

- **标准** - 标准相位
- **交换** - 交换了 Φ1 和 Φ2 时钟

ASign

- **Pos** - 在 Φ1 期间 Cap 连接到 A 输入，在 Φ2 期间 Cap 连接到参考输入
- **Neg** - 在 Φ1 期间 Cap 连接到参考输入，在 Φ2 期间 Cap 连接到 A 输入

ACap

在 0 到 31 单位之间设置 ACap 的值。

ACMux (ASC)

选择连接到 ACap 的 A 输入和连接到 CCap 的 C 输入。 当 SCBLOCK 放置在特定模块中时，PSoC Designer 显示可用的连接选项。

Note 只有当 SCBLOCK 组件放置在 ASC 模块中时，才适用该参数。

AMux (ASD)

选择连接到 ACap 的 A 输入。 当 SCBLOCK 放置在特定模块中时，PSoC Designer 显示可用的连接选项。

Note 只有当 SCBLOCK 组件放置在 ASD 模块中时，才适用该参数。

BCap

在 0 到 31 单位之间设置 BCap 的值。

AnalogBus

- **禁用** - 模拟输出未连接到模拟输出总线，释放了其他用户模块的输出总线。
- **启用** - 模拟输出连接到模拟输出总线作为其列，并可能会路由到模拟缓冲区。

Note 每列只有一个模拟模块输出可以连接到模拟总线。

CompBus

- **禁用** - 电压比较器输出未连接到电压比较器输出总线，释放了电压比较器总线以供相同列中其他用户模块使用。
- **启用** - 电压比较器输出连接到电压比较器输出总线作为其列，并可能会路由到数字资源。

Note 每列只有一个模拟模块电压比较器输出可以连接到电压比较器总线。

AutoZero

- **关** - 禁用 Autozero 函数。

- **开** - 启用 Autozero 函数。 在 Φ1 期间输出连接到负向输入，以测量运算放大器的偏移阻抗。 在 Φ2 期间，实际信号抵消了偏移。

CCap

在 0 到 31 单位之间设置 CCap 的值。

ARefMux

- **AGND** - 在 Φ2 期间 ACap 连接到 AGND
- **REFHI** - 在 Φ2 期间 ACap 连接到 REFHI
- **REFLO** - 在 Φ2 期间 ACap 连接到 REFLO
- **ComparatorBus_x** - 在 Φ2 期间，电压比较器处于高电平时，ACap 连接到 REFHI；电压比较器处于低电平时，ACap 连接到 REFLO。

FSW1

- **关** - FCap 断开与反馈环的连接
- **开** - FCap 连接到反馈环

FSW0

- **关** - 在 Φ1 期间 FCap 没有放电
- **开** - 在 Φ1 期间 FCap 放电

BSW (ASD)

- **关** - BCap 没有作为电容器连接
- **开** - BCap 作为电容器连接

BMux

选择连接到 BCap 的 B 输入。

电源

- **Off** - SCBLOCK 关闭电源
- **Low** - SCBLOCK 设置为最低功耗
- **Med** - SCBLOCK 设置为中功耗
- **High** - SCBLOCK 设置为满功耗

应用程序编程接口

应用程序编程接口 (API) 子程序作为用户模块的一部分提供，使设计人员能够在较高的层级处理模块。本部分具体说明了每个函数对应的接口以及 “include” 文件所提供的相关常量。

每次布置用户模块时，都会为其分配一个实例名称。 默认情况下，PSoC Designer 将会为给定项目中此用户模块的第一个实例分配 SCBLOCK_1。 可将该值更改为符合标识符语法规则的任意唯一值。 分配的实例名称成为每个全局函数名称、变量和常量符号的前缀。 为简便起见，在以下说明中将实例名称缩写为 SCBLOCK。

Note 在这里，如同所有用户模块 API 中的一样，A 和 X 寄存器的值可能因调用 API 函数而更改。 发起调用 API 的函数有责任在调用之前保留 A 和 X 的值 （如果调用后需要再次用到它们）。 选择这种 “ 寄存器易变 ” 策略是为了提高效率，并且自从 PsoC Designer 的 1.0 版本起使用。 C 编

译器自动遵循此要求。 汇编语言程序员也必须确保其代码遵守这一策略。 虽然一些用户模块 API 函数可以保留 A 和 X 不变，但是无法保证它们将来也会如此。

SCBLOCK_Start

说明：

设置此用户模块的功耗水平，并执行所有必需的初始化。

C 原型：

```
void  SCBLOCK_Start (BYTE bPowerSetting)
```

汇编：

```
mov   A, bPowerSetting
lcall  SCBLOCK_Start
```

参数：

bPowerSetting： 用于指定功率电平的一个字节。 在复位和配置之后，关闭分配的模拟 PSoC 模块的电源。 下表包含了在 C 语言和汇编语言中提供的符号名称及其相关值。

符号名	值
SCBLOCK_OFF	0
SCBLOCK_LOWPOWER	1
SCBLOCK_MEDPOWER	2
SCBLOCK_HIGHPOWER	3

功耗水平会影响模拟性能。 必须为每个应用程序确定正确的功耗设置。

返回值：

无

副作用：

此函数可能更改 A 和 X 寄存器。

SCBLOCK_SetPower

说明：

设置 SCBLOCK 用户模块的功耗水平。

C 原型：

```
void  SCBLOCK_SetPower (BYTE bPowerSetting)
```

汇编：

```
mov   A, bPowerSetting
lcall  SCBLOCK_SetPower
```

参数：

bPowerSetting： 与用于 “ 启动 ” 进入点的 bPowerSetting 参数相同。 允许用户在运行模块时更改功耗水平。

返回值：

无

副作用：

此函数可能更改 A 和 X 寄存器。

SCBLOCK_Stop

说明：

将功耗水平设置为 OFF。

C 原型：

```
void  SCBLOCK_Stop()
```

汇编：

```
lcall  SCBLOCK_Stop
```

参数：

无

返回值：

无

副作用：

此函数可能更改 A 和 X 寄存器。

固件源代码示例

以下是一个使用 C 语言编写的示例程序，该程序使用了 SCBLOCK：

```
#include "SCBLOCK.h"
void main(void)
{
    SCBLOCK_Start(SCBLOCK_HIGHPOWER);    // Turn on SCBlock power
    // User code
    SCBLOCK_Stop();                      // Turn off SCBlock power
}
```

此示例代码首先启动 SCBLOCK 用户模块，然后等到计数达 255 次后停止 SCBLOCK：

```
;;; Sample Code for the SCBLOCK

include "SCBLOCK.inc"
include "m8c.inc"
export _main

_main:
   mov  A,SCBLOCK_HIGHPOWER      ; Set Power
   call SCBLOCK_Start            ; Turn on SCBLOCK

   mov  A,ffh                    ; Set loop counter
loop1:
   dec  A
   jnz  loop1
   call SCBLOCK_Stop             ; Turn off SCBlock
```

```
ret
```

配置寄存器

Table 2. 模块 SCBLOCK：寄存器 CR0

位	7	6	5	4	3	2	1	0
值	FCap	ClockPhase	ASign	ACap				

Table 3. 模块 SCBLOCK：寄存器 CR1

位	7	6	5	4	3	2	1	0
ASC	ACMux			BCap				
ASD	AMux			BCap				

ACMux 在模块置于 ASC 模块中时使用。 AMux 在模块置于 ASD 模块中时使用。 这两个字段值均取决于用户连接输入的方式。

Table 4. 模块 SCBLOCK：寄存器 CR2

位	7	6	5	4	3	2	1	0
值	AnalogBus	CompBus	AutoZero	CCap				

Table 5. 模块 SCBLOCK：寄存器 CR3

位	7	6	5	4	3	2	1	0
ASC	ARefMux		FSW1	FSW0	BMuxASC		功耗	
ASD	ARefMux		FSW1	FSW0	BSW	BMuxASD	功耗	

BMuxASC 在模块置于 ASC 模块中时使用，并且取决于用户连接输入的方式。 BMuxASD 在模块置于 ASD 模块中时使用，并且取决于用户连接输入的方式。 BSW 在模块置于 ASD 模块中时使用。 该值决定了 BCap 开关是否处于活动状态。