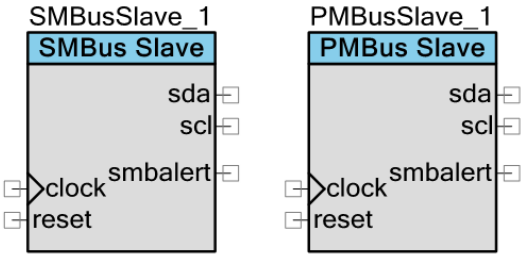
SM 总线和 PM 总线从器件

2.20



性能

- SMBus 从器件模式
- PMBus 从器件模式
- SMBALERT#引脚支持
- 25 ms 超时
- 固定功能（FF）和 UDB 实现
- 可配置 SM/PM 总线指令

概述

系统管理总线（SMBus）和电源管理总线（PMBus）从器件组件提供了一种简单的方式来使用其上运行的 SMBus 或 PMBus 协议将 I^2C 物理层接口添加至 PSoC 3 或 PSoC 5LP 设计中。

SMBus 是一个双线接口，包括可与系统主机进行通信的各种系统管理芯片。它将 I^2C 当作为物理层使用。SMBus 从器件组件实现了大多数的 SMBus 从器件规范，同时提供用于配置从器件参数的选项。此从器件可使用提供的 API 与 SMBus 主控进行通信。

PMBus 协议是更为通用的 SMBus 协议的特定实现。使用 PMBus，此组件显示所有可能的 PMBus 指令，并允许选择哪些指令与您的应用程序相关联。

何时使用 SM 总线和 PM 总线从器件

此组件可用于需要 SMBus 或 PMBus 从器件通信接口的设计中。此组件处理硬件中的大量物理层请求。此固件处理协议和存储器缓冲区管理；它还管理 I^2C 中的数据传送。

输入/输出连接

本节介绍 SMBus 从器件的各种输入和输出连接。I/O 列表中的星号（*）表示 I/O 可能在某种 SAR ADC 配置下隐藏。

时钟 — 输入

时钟输入应用作为I²C SCL/SDA低电平停滞超时计时器的时钟源。当Implementation（实现）参数设置为UDB时，它需要时钟提供16倍过采样。

当 I²C**Implementation**（实现）参数设置为 **UDB** 时，时钟输入可用。

数据速率	时钟
10 kbps	160 kHz
50 kbps	800 kHz
100 kbps	1.6 MHz
400 kbps	6.4 MHz
1000 kbps	16 MHz

复位 — 输入

UDB I²C 的硬件复位。如果高电平复位引脚保持在逻辑高电平状态，则 I²C 模块将处于复位状态，并且通过 I²C 进行的通信会停止。SDA 和 SCL 应强制为高电平。这仅适用于硬件复位。而软件必须使用 Stop()和 Start() API 进行单独复位。

sda（SMBDAT）— 输入/输出

串行数据（SDA）是 I²C 数据信号。这种双向数据信号用于传输或接收所有总线数据。应该将连接至 sda 的引脚配置为开漏驱动低电平（Open-Drain-Drives-Low）。如果您在自定义程序中选择了“External I/O Option”，单个 sda 双向信号将由单独的输入和输出（sda_o 和 sda_i）替换。在某些应用程序需要使能多个 I²C 总线的多路复用时，这是必要的。

scl （SMBCLK） — 输入/输出

串行时钟（SCL）是主控生成的 I²C 时钟。虽然从器件从不会生成时钟信号，但它可使时钟保持在低电平状态，并使总线停顿，直至它准备发送数据或确认/否认最新数据或地址为止。应该将连接至 scl 的引脚配置为开漏驱动低电平（Open-Drain-Drives-Low）。如果您在自定义程序中选择了“External I/O Option”，单个 scl 双向信号将由单独的输入和输出（sdl_o 和 sdl_i）替换。在某些应用程序需要使能多个 I²C 总线的多路复用时，这是必要的。

smbalert （SMBALERT#） — 输出

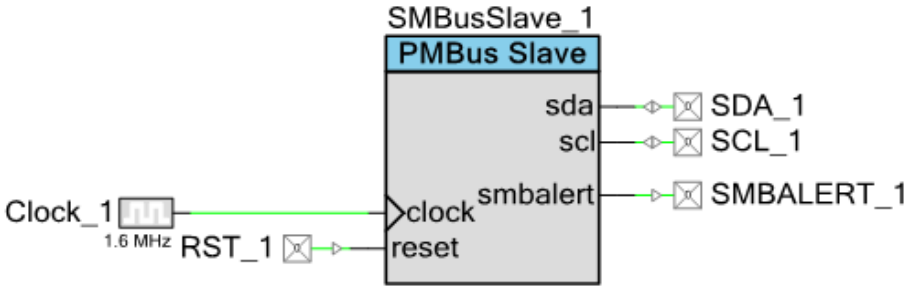
警报是可选 SMBus 定义的 SMBALERT#引脚。可以选择使能此信号。警报引脚可通过后文介绍的 API 进行设置/解除。应该将连接至 smbalert 的引脚配置为开漏驱动低电平（Open-Drain-Drives-Low）。

原理图宏信息

默认情况下，PSoC Creator 组件目录包括 SM/PM 总线从器件组件的原理图宏实现。这些宏包括已经连接和配置的引脚以及定义数据速率的内部配置的时钟值。原理图宏使用配置为使用固定功能 I²C 模块和硬件地址解址的 I²C 从器件组件。

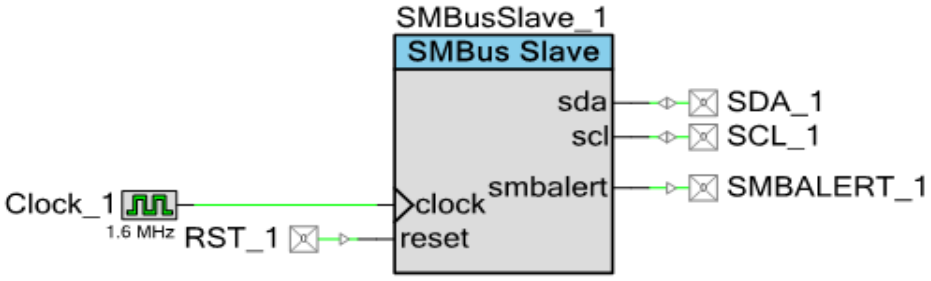
SM 总线从器件宏

此宏提供了 SM/PM 总线从器件组件在 SM 总线模式下的正确配置。连接至终端 SCL 和 SDA 的引脚被配置为双向，且 Drive（驱动）模式被设置为开漏驱动低电平（Open-Drain-Drives-Low）。此组件定义数据速率为 100 kHz。



PM 总线从器件宏

此宏提供了 SM/PM 总线从器件组件在 PM 总线模式下的正确配置。连接至终端 SCL 和 SDA 的引脚被配置为双向，且 Drive（驱动）模式被设置为开漏驱动低电平（Open-Drain-Drives-Low）。此组件将数据速率定义为 400 kHz。

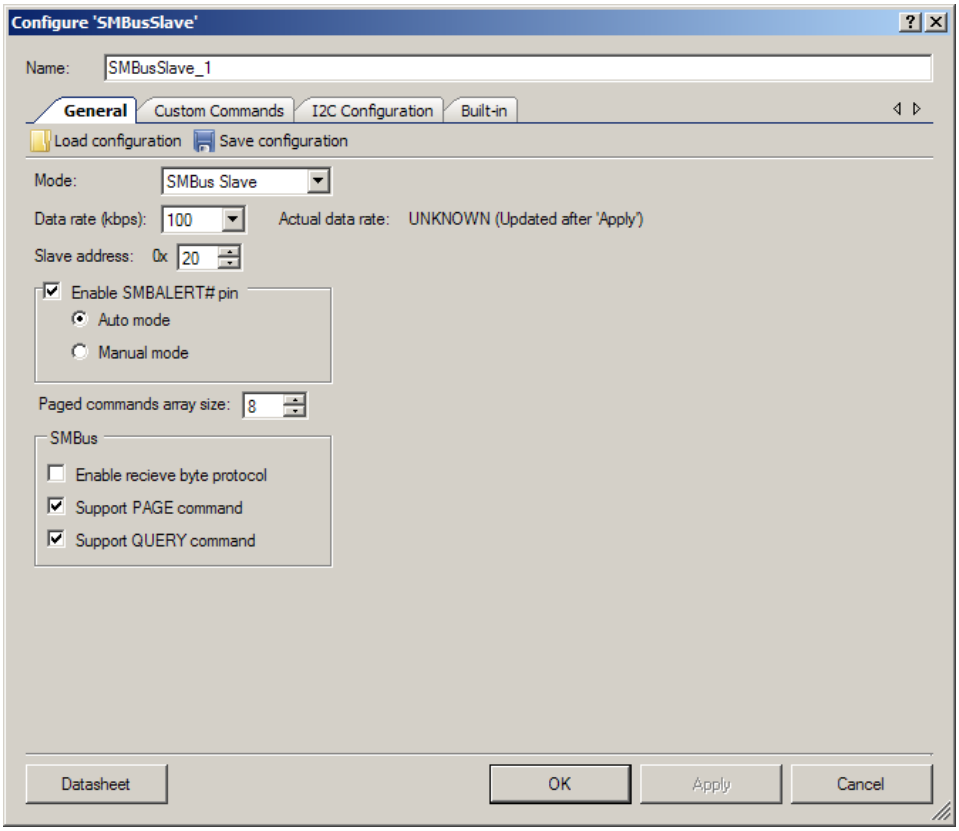


组件参数

将一个 SM/PM 总线从器件组件拖放到您的设计上，并双击以打开 **Configure** 对话框。默认情况下，**Configure** 对话框最初显示“**General**”选项卡。

General（常规）选项卡

General 选项卡提供用于配置 SM/PM 总线的一般设置的选项。有以下参数可用。



Export/Import Configuration（导出/导入配置）

Export/Import Configuration（导出/导入配置）允许您将自定义程序设置保存和恢复到外部文件中。它允许快速加载预设的配置文件和保留自定义设置。

Mode（模式）

Mode 参数将此组件的模式选择为 SMBus 从器件模式或 PMBus 从器件模式。如果选择了 SMBus 从器件模式，则将禁用 PMBus Commands 选项卡。**Mode** 还将确定可用的数据速率。在 PMBus 从器件模式中，只有两个选项：100 和 400 kHz。在 SMBus 从器件模式中，还有额外的 10 和 50 kHz 选项。此参数的默认设置为 SMBus 从器件模式。

Data Rate（数据速率）

Data Rate 参数用于选择 I²C 数据速率。可用选项取决于 SM/PM Bus **Mode** 选择。PMBus 从器件模式允许 **Data Rate** 值为 100 kHz 和 400 kHz。SMBus 从器件模式提供 10 kHz、50 kHz、100 kHz 和 400 kHz 选项。**Data Rate** 参数的默认值为 100 kHz。

Slave address（从器件地址）

Slave address 参数确定此器件的 I²C 地址（7 位格式）。输入的值是十进制或十六进（如果最前面是“0x”）。自定义程序验证此地址以确保它不会与 SMBus 从器件保留列表上任何地址相冲突。**Slave address** 的默认值为默认值：0x20。

SMBALERT

通过 **SMBALERT** 框，您可以为主机通知配置可选 SMBALERT#输出引脚。如果选择 **Enable SMBALERT# pin**，该引脚将当作为组件符号上的输出显示。主机 Auto/Manual 按钮确定在主机查询警报响应地址（SetSmbAlertMode()API 的 GUI 表示）处的器件后，**SMBALERT# pin** 是否会自动解除。此参数的默认值设置为 Enabled Auto。

Paged Commands（分页指令）

此参数框配置 PMBus PAGE 指令。**Maximum page** 参数确定分页指令的阵列大小。所有分页指令共享此阵列大小。默认值设置为 8 页。**有效范围为 1 到 32。**

SMBus 框

SMBus 框配置可选 SMBus 功能。它仅在 SMBus 从器件模式下显示。

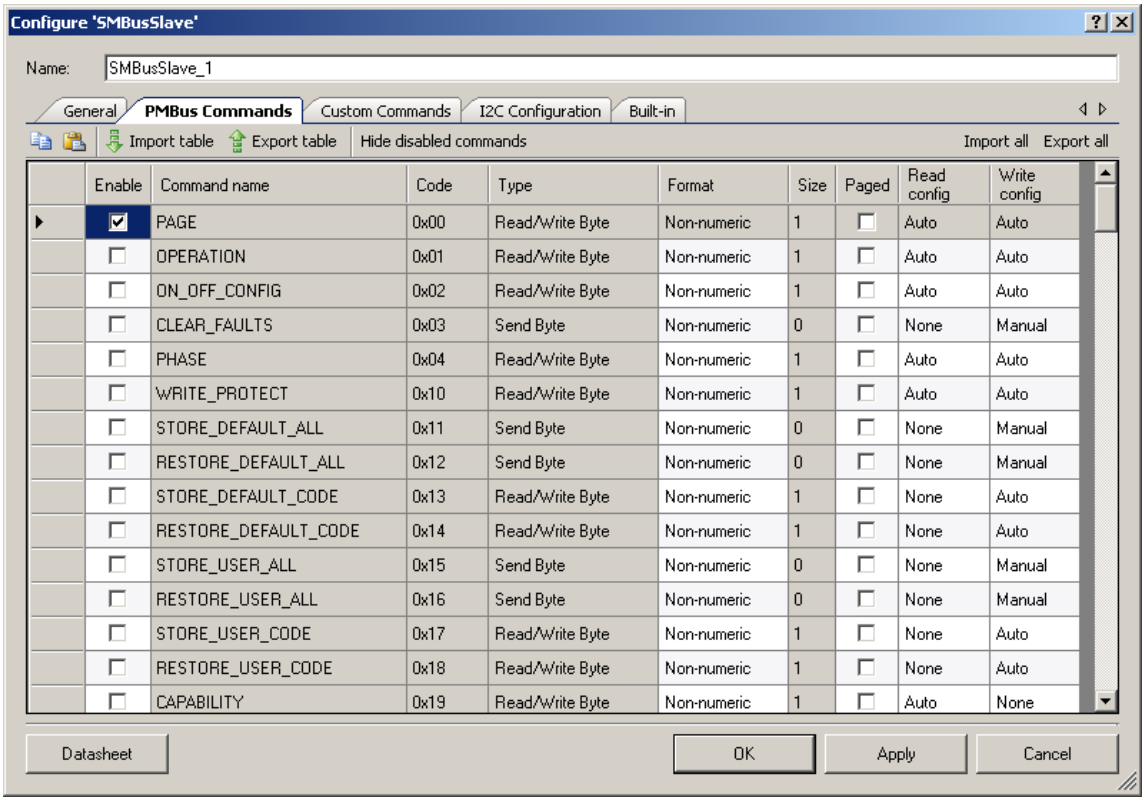
- **Enable receives the byte protocol** 复选框使能/禁用对 SMBus 接收字节协议的支持。如果未选中，任何接收字节数据操作都将被视为总线错误。如果已选中，此组件将调用 SMBus_GetReceiveByteResponse() API 确定接收字节请求的响应字节。
- **Support PAGE Command** 复选框允许您在 SMBusSlave 模式下访问 PAGE 指令。
- **Support QUERY Command** 复选框允许您在 SMBusSlave 模式下访问 QUERY 指令。

如果使能了 PAGE 或 QUERY 指令，则这些指令将被添加至 **Custom Commands** 选项卡上的指令列表中。这些指令的属性是基于 PMBus 规范的，但也可以对指令代码进行完全的自定义。

此框的默认值为：**Enable receives byte protocol** 未选中、**Support PAGE Command** 已启用、**Command Code**=0x00、**Support QUERY command** 已启用、**Command Code**=0x1A。

PM Bus Commands（PM 总线指令）选项卡

PM Bus Commands 选项卡在此组件的 **Mode**（模式）设置为 PMBus Slave 时可用。此选项卡显示 PMBus 规范中的定义指令的完整列表。预先填充的信息包括指令名称、其数字指令代码以及指令的类型（即 SMBus 协议）。您可以启用/禁用您希望此组件的实例处理的指令。名称、代码和类型字段都是只读字段。此选项卡中的可用参数概述如下。



Enable	Command name	Code	Type	Format	Size	Paged	Read config	Write config
☑	PAGE	0x00	Read/Write Byte	Non-numeric	1	☐	Auto	Auto
☐	OPERATION	0x01	Read/Write Byte	Non-numeric	1	☐	Auto	Auto
☐	ON_OFF_CONFIG	0x02	Read/Write Byte	Non-numeric	1	☐	Auto	Auto
☐	CLEAR_FAULTS	0x03	Send Byte	Non-numeric	0	☐	None	Manual
☐	PHASE	0x04	Read/Write Byte	Non-numeric	1	☐	Auto	Auto
☐	WRITE_PROTECT	0x10	Read/Write Byte	Non-numeric	1	☐	Auto	Auto
☐	STORE_DEFAULT_ALL	0x11	Send Byte	Non-numeric	0	☐	None	Manual
☐	RESTORE_DEFAULT_ALL	0x12	Send Byte	Non-numeric	0	☐	None	Manual
☐	STORE_DEFAULT_CODE	0x13	Read/Write Byte	Non-numeric	1	☐	None	Auto
☐	RESTORE_DEFAULT_CODE	0x14	Read/Write Byte	Non-numeric	1	☐	None	Auto
☐	STORE_USER_ALL	0x15	Send Byte	Non-numeric	0	☐	None	Manual
☐	RESTORE_USER_ALL	0x16	Send Byte	Non-numeric	0	☐	None	Manual
☐	STORE_USER_CODE	0x17	Read/Write Byte	Non-numeric	1	☐	None	Auto
☐	RESTORE_USER_CODE	0x18	Read/Write Byte	Non-numeric	1	☐	None	Auto
☐	CAPABILITY	0x19	Read/Write Byte	Non-numeric	1	☐	Auto	None

Format（格式）

Format 参数指定此指令的数字格式。此组件在指定对 QUERY 指令的响应时使用此格式。QUERY 指令中的可选格式值有 Linear（线性）、Signed（有符号）、Direct（直接号）、Unsigned（无符号）和 VID Mode（VID 模式）。此字段仅用于 QUERY 指令目的，因为计算机并不执行任何实际的数字转换。

Size（大小）

对于模块和进程调用类型的指令，您可以编辑 **Size** 字段以指定数据元素的大小。此大小不包括 SMBus 协议附加到模块传输的起始位置的大小/计数字节。仅可针对模块或进程调用指令编辑此字段。对于所有其他类型，**Size** 字段来源于 PMBus 规范。默认值为 16。

Read/Write Config（读/写配置）

针对每条指令，选择此指令是否通过 **Read Config** 和 **Write Config** 参数可读和/或可写。为每条指令选择 None（无）、Auto（自动）或 Manual（手动）。

- None 说明此操作被禁用（即将只读指令的 Write Config 设置为 None）。
- “Auto”（自动）模式命令完全由此器件处理。它们在寄存器存储区与 I^2C 传输缓冲区之间传输，同时不会出现用户固件干扰或通知。
- “Manual”（手动）命令被添加至数据操作队列，必须由用户固件进行处理。这些参数的默认值为“Manual”。

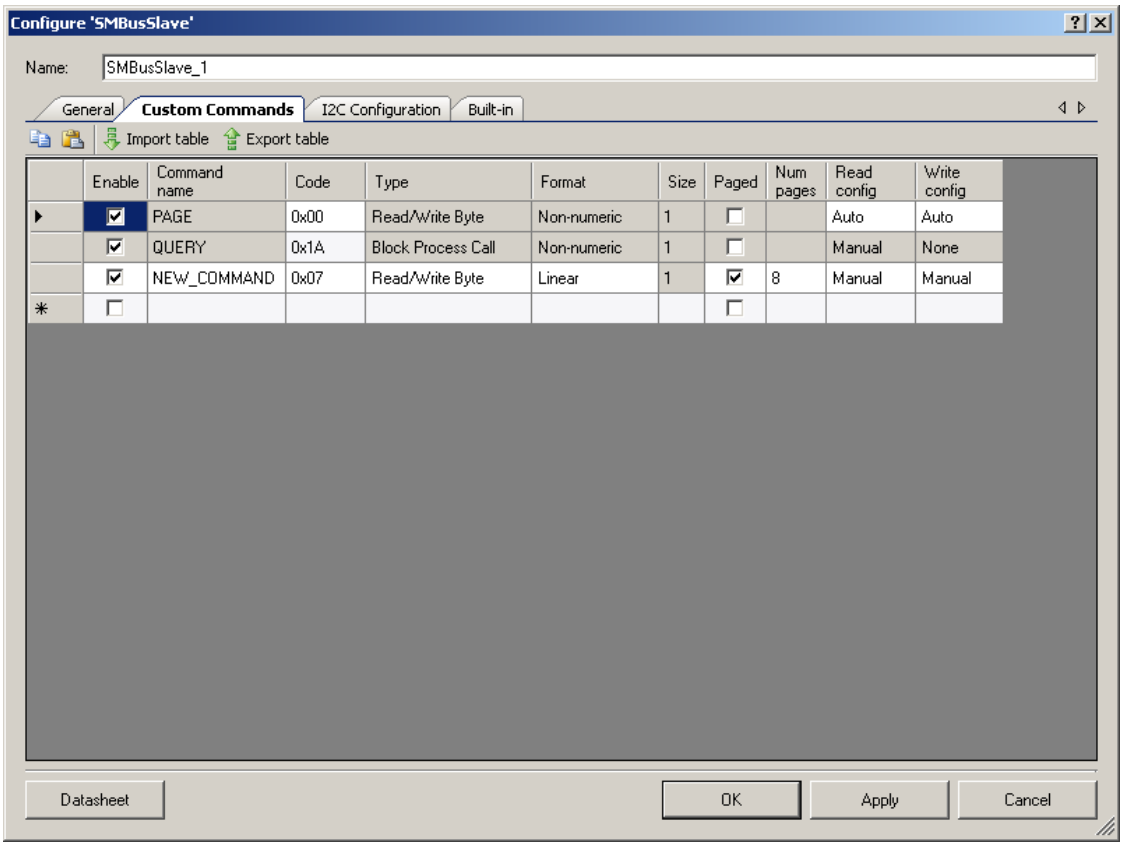
注意：由于写与读之间可能的不对称以及进程调用协议的复杂性，针对使用此协议的指令不得选择自动模式。

Paged（分页）

Paged 复选框说明此指令是否分页（即索引）。针对被指定为分页的指令，此组件将自动在寄存器存储区中为此指令生成阵列。此阵列的大小取决于 **Maximum Page** 参数。针对 Auto（自动）读写，此组件将基于当前的 PMBus 页面自动索引至分页参数的正确阵列编号（由上一个 PAGE 指令选定）。此参数的默认设置为未选中。

Custom Commands（自定义指令）选项卡

此选项卡允许您修改并自定义 Command Name（指令名称）、Code（代码）和 Type（类型）字段。



Command Name（指令名称）

这是用户为此指令指定的名称。允许的字符为 A-Z（全部大写）、0-9 和下划线“_”。最大长度为 24 个字节。第一个字符不得为数字。不允许指令名称重复（包括重复标准的 PMBus 指令名称的自定义指令名称）。**Command Name** 默认为空白。

Command Code（指令代码）

这是此指令的数字代码。它是十六进制值，限于两个字符（0-9、ssA-F）。不允许重复的指令代码。这包括与使能的 PMBus 指令相冲突的代码。**Command Code** 默认为空白。

Type（类型）

Type 指定针对此指令使用的 SMBus 传输协议/数据大小。可选值为 Send Byte（发送字节）、Read/Write Byte（读/写字节）、Read/Write Word（读/写字）、Read/Write Block（读/写模

块）、Process Call（进程调用）和 Block Process Call（模块进程调用）。它默认设置为 Read/Write Byte。

Format（格式）

Format 参数指定此指令的数字格式。此组件在指定对 QUERY 指令的响应时使用此格式。QUERY 指令中的可选格式值有 Linear（线性）、Signed（有符号）、Direct（直接号）、Unsigned（无符号）和 VID Mode（VID 模式）。此字段仅用于 QUERY 指令目的，因为计算机并不执行任何实际的数字转换。

Size（大小）

对于模块和进程调用类型的指令，您可以编辑 **Size** 字段以指定数据元素的大小。此大小不包括 SMBus 协议附加到模块传输的起始位置的大小/计数字节。仅可针对模块或进程调用指令编辑此字段。对于所有其他类型，**Size** 字段来源于 PMBus 规范。默认值为 16。

Paged（分页）

Paged 复选框说明此指令是否分页（即索引）。针对被指定为分页的指令，此组件将自动在寄存器存储区中为此指令生成阵列。此阵列的大小取决于 **Pages** 参数。针对 Auto（自动）读写，此组件将基于当前的 PMBus 页面自动索引至分页参数的正确阵列编号（由上一个 PAGE 指令选定）。此参数的默认设置为未选中。

分页参数的数量

默认情况下，分页参数的数量是指定在 General 选项卡上的分页总数。将该参数应用于特殊指令时，用户可以选择减小它。该参数的最小值为 1。最大值为分页的总数。未选中 Paged 复选框时，该参数将以灰色显示。

Read/Write Config（读/写配置）

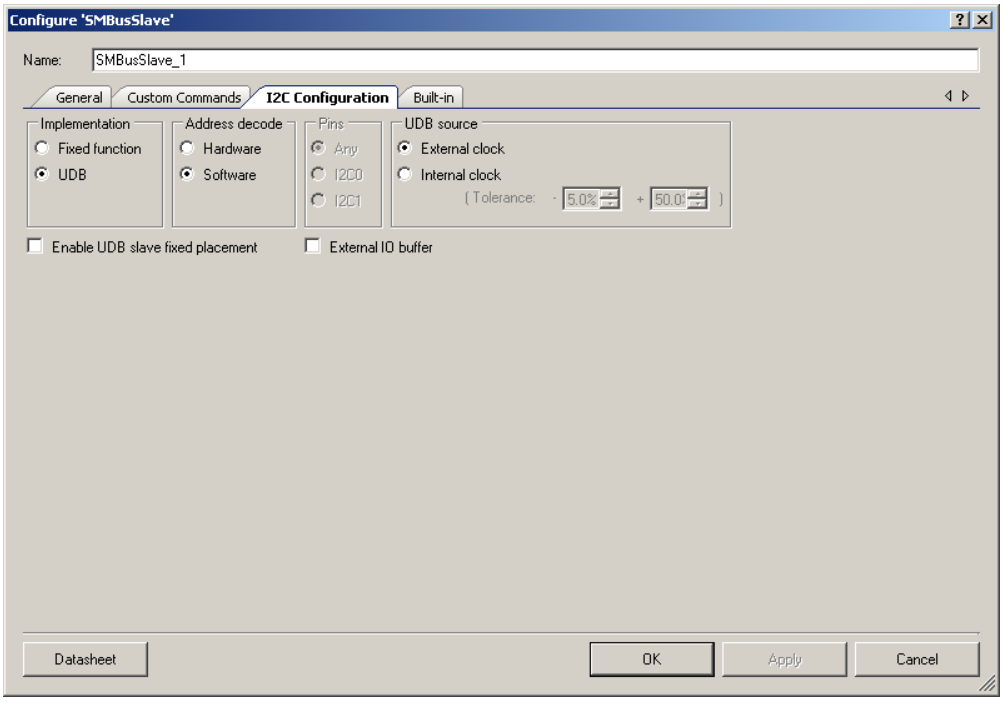
针对每条指令，选择此指令是否通过 **Read Config** 和 **Write Config** 参数可读和/或可写。为每条指令选择 None（无）、Auto（自动）或 Manual（手动）。

- None 说明此操作被禁用（即将只读指令的 Write Config 设置为 None）。
- “Auto”（自动）模式命令完全由此器件处理。它们在寄存器存储区与 I^2C 传输缓冲区之间传输，同时不会出现用户固件干扰或通知。
- “Manual”（手动）命令被添加至数据操作队列，必须由用户固件进行处理。这些参数的默认值为“Manual”。

注意：由于写与读之间可能的不对称以及进程调用协议的复杂性，针对使用此协议的指令不得选择自动模式。

I²C Configuration（I2C 配置）选项卡

此选项卡允许配置 I²C 硬件。



Implementation（实现）

此参数确定 I²C 硬件是使用 **Fixed Function** 还是 **UDB** 实现。默认模式设置为 UDB。

Address decode（地址解码）

通过此参数，您可以选择软件地址解码或硬件地址解码。对于提供了足够的 API 并且只需一个从器件地址的大部分应用程序来说，将首选硬件地址解码。而对于希望修改源代码以检测多个从器件地址或 10 位地址的应用程序而言，则必须使用软件地址检测。Hardware（硬件）是此参数的默认设置。如果使能硬件地址解码，该模块会自动不响应不属于自己的地址，而无需 CPU 干预。它会在收到正确地址后自动中断 CPU 的运行，并将 SCL 线保持为低电平，直至 CPU 干预为止。

Pins（引脚）

此参数可确定用于 SDA 和 SCL 信号连接的引脚类型。此参数包含三个可选值：Any、I2C0 和 I2C1。默认值为“Any”。“Any”表示通用 I/O（GPIO 或 SIO）。

UDB clock source（UDB 时钟源）

该参数允许您针对数据速率生成在内部配置的时钟和外部配置的时钟之间进行选择。考虑到 16 倍过采样，当设置为 Internal Clock（内部时钟）时，PSoC Creator 将基于 **Data Rate** 参数计算和

配置所需时钟频率。在 External Clock（外部时钟）模式下，组件不控制数据速率而是基于用户连接的时钟源显示实际速率。如果将此参数设置为 Internal Clock（内部时钟），则符号上将不显示时钟输入。您可以为内部时钟输入所需容差值。时钟容差可指定为百分比。默认范围为-5%到+50%。

Enable UDB slave fixed placement（使能 UDB 从器件固定放置）

Enable UDB slave fixed placement 参数允许您选择一个固定组件放置，以提供高于无限制放置下的组件性能。当设置此参数后，全部组件资源都将固定在器件的右上角。此参数可控制连接至组件的引脚分配。引脚分配的选择并非组件性能的决定因素。此选项仅在 **Implementation** 设置为 UDB 时才有效。默认情况下，禁用此选项。此组件的固定放置状态消除了路由可变性。此外，还可以在一个完全空白的设计中按照与非固定放置设计相同的方式继续运行固定放置。

External IO Buffer（外部 IO 缓冲区）

此参数允许内部 I²C 总线复用。内部 OE 缓冲区已被移除，双向 scl 和 sda 终端已被单独的输入（sda_i 和 scl_i）和输出（sda_o 和 scl_o）替换。应用程序编程接口（API）路由允许您使用软件配置此组件。下表列出每个函数的接口，并进行了说明。以下各节将更详细地介绍每个函数。

应用程序编程接口

应用程序编程接口（API）函数允许用户在运行时使用软件配置组件。下表列出每个函数的接口，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“SMBusSlave_1”分配给提供的设计中的第一个组件实例。您可以将其重新命名为任何一个符合标识符语法规则的值。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“SMBusSlave”。

注意：一些组件 API 功能被用于组件中断服务子程序中，因此，在使用 Keil 构建时，编译器可能会引起编译器警告提示。为了避免这种情况，将这些函数纳入到了“.cyre”文件中。

函数	说明
SMBusSlave_Start()	初始化并使能SMBus组件。使能I²C中断后，该组件可响应SMBus通信。
SMBusSlave_Stop()	停止对SMBus通信的响应。同时禁用中断。
SMBusSlave_EnableInt()	使能I²C中断。
SMBusSlave_DisableInt()	禁用I²C中断。
SMBusSlave_Init()	使用自定义程序提供的初始值来初始化I²C寄存器。
SMBusSlave_Enable()	激活I²C硬件。不使能正常操作需要的I²C中断。
SMBusSlave_SetAddress()	设置I²C从器件地址。

函数	说明
SMBusSlave_SetAlertResponseAddress()	设置I²C从器件地址，在此地址中当处于警报响应地址模式时，此器件将进行响应。
SMBusSlave_GetNextTransaction()	将一个指针返回至数据操作队列中的下一个数据操作记录。如果此队列为空，此函数将返回NULL。
SMBusSlave_GetTransactionCount()	返回数据操作队列中的数据操作记录的数量。
SMBusSlave_CompleteTransaction()	促使此组件完成此队列最前面的当前待处理数据操作。
SMBusSlave_SetSmbAlert()	声明或解除SMBALERT# smbalert引脚。
SMBusSlave_SetSmbAlertMode()	确定此组件如何响应SMBus主控针对警报响应地址的读取。
SMBusSlave_HandleSmbAlertResponse()	当主机响应警报响应地址且SMBALERT模式设置为FIRMWARE_MODE时，将被此组件调用。
SMBusSlave_GetReceiveByteResponse()	当I²C中断服务子程序检测到“接收字节”协议请求时，将被其调用以确定响应字节。
SMBusSlave_HandleBusError()	当出现总线协议错误时，将被此组件调用。
SMBusSlave_StoreUserAll()	将RAM寄存器存储区保存至闪存中的用户寄存器存储区。
SMBusSlave_RestoreUserAll()	验证用户寄存器存储区的CRC字段，并将用户寄存器存储区的内容复制到RAM寄存器存储区中。
SMBusSlave_EraseUserAll()	通过调用该函数，用户可以擦除闪存中的用户存储区。
SMBusSlave_RestoreDefaultAll()	验证默认寄存器存储区的签名字段，并将默认寄存器存储区的内容复制至RAM寄存器存储区中。
SMBusSlave_StoreComponentAll()	调用此函数以使用当前PMBus设置更新系统中的其他组件的参数。
SMBusSlave_RestoreComponentAll()	调用此函数以使用系统中的其他组件的当前配置参数更新PMBus工作寄存器存储区。
SMBusSlave_Lin11ToFloat()	将参数“linear11”转换为浮点并将其返回。
SMBusSlave_FloatToLin11()	使用参数“floatvar”（浮点编号）并将其转换为16位LINEAR11值（11位尾数+5位指数），同时将其返回。
SMBusSlave_Lin16ToFloat()	将参数“linear16”转换为浮点并将其返回。
SMBusSlave_FloatToLin16()	使用参数“floatvar”（浮点编号）并将其转换为16位LINEAR16值（16位尾数），同时将其返回。

全局变量

函数	说明
SMBusSlave_initVar (static)	initVar变量用于说明此组件的初始配置。此变量前面加有组件名称，此处为SMBusSlave。SMBusSlave_initVar变量被初始化为0，并在第一次调用SMBusSlaveStart()时被设置为1。这可以实现组件初始化，同时无需重新初始化SMBusSlave Start()路由中的所有后续调用。 当此器件经过睡眠周期时，需要重新初始化此组件。因此，在执行睡眠SMBusSlave Sleep()时，此变量设置为0；并在SMBusSlave Wakeup()中执行的重新初始化期间进行设置。

void SMBusSlave_Start(void)

说明： 这是开始执行组件操作的首选方法。SMBusSlave_Start()调用SMBusSlave_Init()函数，然后调用SMBusSlave_Enable()函数。您必须在执行I²C总线操作之前调用SMBusSlave_Start()。此API可使能I²C中断。

参数： 无

返回值： 无

副作用： 无

void SMBusSlave_Stop(void)

说明： 此函数可禁用I²C硬件和中断。如果I²C总线被器件锁定且被设为空闲状态，则它会将其释放。

参数： 无

返回值： 无

副作用： 无

void SMBusSlave_EnableInt(void)

说明： 此函数可使能I²C中断。

参数： 无

返回值： 无

副作用： 无

void SMBusSlave_DisableInt(void)

说明： 此函数可使能I²C中断。通常情况下，在I2C_Stop()函数禁用中断后就不需要此函数了。

参数： 无

返回值： 无

副作用： 如果在运行I²C的同时禁用I²C中断，则可能造成I²C总线锁定。

void SMBusSlave_Init(void)

说明： 此函数根据配置窗口“Configure”对话框设置来初始化或恢复组件。您无需调用SMBusSlave_Init()，因为SMBusSlave_Start() API会调用此函数，该函数是开始执行组件操作的首选方法。

参数： 无

返回值： 无

副作用： 根据自定义程序“Configure”对话框中的内容，对所有寄存器进行设置。

void SMBusSlave_Enable(void)

说明： 此函数激活硬件并开始执行组件操作。您无需调用SMBusSlave_Enable()，因为SMBusSlave_Start() API会调用此函数，该函数是开始执行组件操作的首选方法。如果要调用此API，则必须首先调用SMBusSlave _Start()或SMBusSlave _Init()。

参数： 无

返回值： 无

副作用： 无

void SMBusSlave_SetAddress(uint8 address)

说明： 此函数可设置I²C从器件地址。

参数： uint8 address：主设备的I²C从器件地址。此值可以为0至127 （0x00 - 0x7F）之间的任意地址。此地址为右对齐的7位从器件地址，它不包括读/写位。

返回值： 无

副作用： 无

void SMBusSlave_SetAlertResponseAddress(uint8 address)

说明： 此函数设置I²C从器件地址，在此地址中当处于警报响应地址模式时，此器件将进行响应。

参数： uint8 address：针对警报响应模式的I²C从器件地址。此值可以为0至127 （0x00 - 0x7F）之间的任意地址。此地址为右对齐的7位从器件地址，它不包括读/写位。

返回值： 无

副作用： 无

TRANSACTION_STRUCT* SMBusSlave_GetNextTransaction(void)

说明： 此函数将一个指针返回至数据操作队列中的下一个数据操作记录中。如果此队列为空，此函数将返回NULL。此函数将仅返回手动读取和写入，因为此组件将处理队列中的任何自动数据操作。如果是写入，服务于数据操作队列的用户固件负责将“有效负载”复制到寄存器存储区。如果是读取，用户固件负责更新寄存器存储区中此指令的变量的内容。如果是这两者，调用SMBusSlave_CompleteTransaction()以释放数据操作记录。

注意，对于读取数据操作，长度字段和有效负载字段不适于大多数的数据操作类型。例外情况是进程调用，其中写入相中的字将被存储在有效负载字段中。

参数： 无

返回值： 指向下一个数据操作记录的指针

副作用： 无

uint8 SMBusSlave_GetTransactionCount(void)

说明： 返回数据操作队列中的数据操作记录的数量。

参数： 无

返回值： uint8：数据操作队列中的记录数量

副作用： 无

void SMBusSlave_CompleteTransaction(void)

说明： 促使此组件完成此队列最前面的当前待处理数据操作。用户固件数据操作处理程序在处理数据操作之后调用此函数。这将提醒组件代码将与寄存器存储区中的待处理读取数据操作相关联的寄存器变量复制到I²C传输存储区中，以便于传输可以完成。它还可加快队列。必须针对读取和写入进行调用。

参数： 无

返回值： 无

副作用： 无

void SMBusSlave_SetSmbAlert(uint8 assert)

说明： 声明或解除SMBALERT# smbalert引脚。只要声明了SMBALERT#，此组件就将响应主控针对警报响应地址的读取。此响应将是此器件的主要从器件地址。根据模式设置，此组件将自动解除SMBALERT#，调用SMBusSlave_HandleSmbAlertResponse() API，或不执行任何操作。

参数： uint8：声明

函数值	说明
SMBusSlave_SMBALERT_DEASSERT	解除smbalert引脚
SMBusSlave_SMBALERT_ASSERT	声明smbalert引脚

返回值： 无

副作用： 无

void SMBusSlave_SetSmbAlertMode(uint8 alertMode)

说明： 此函数确定此组件如何响应SMBus主控针对警报响应地址的读取。当设置了SMBALERT#时，SMBus主控可将读取传播至全局警报响应地址以确定共享总线上的哪个SMBus设备已设置SMBALERT#。

在自动模式下，一旦总线主控成功地READ警报响应地址，SMBALERT#将自动解除。

在手动模式下，此组件将调用API SMBusSlave_HandleSmbAlertResponse()，其中用户代码（在合并部分）负责解除SMBALERT#。

在DO_NOTHING模式中，此组件将不执行任何操作。

参数： uint8：alertMode，定义SMBALERT引脚模式的字节

函数值	说明
SMBusSlave_DO_NOTHING	不对SMBALERT#引脚执行任何操作
SMBusSlave_AUTO_MODE	自动解除SMBALERT#引脚
SMBusSlave_FIRMWARE_MODE	用户负责解除SMBALERT#引脚

返回值： 无

副作用： 无

void SMBusSlave_HandleSmbAlertResponse(void)

说明： 当主机响应警报响应地址且SMBALERT模式设置为FIRMWARE_MODE时，此API将被此组件调用。此函数包含合并代码部分，在此部分中当主控响应后，用户插入要运行的代码。例如，用户可能会更新状态寄存器并解除SMBALERT#引脚。

参数： 无

返回值： 无

副作用： 无

uint8 SMBusSlave_GetReceiveByteResponse(void)

说明： 当I²C中断服务子程序检测到“接收字节”协议请求时，此函数将被其调用以确定响应字节。此函数包括合并代码部分，在此部分中用户可插入其代码以覆盖此函数的返回值 - 即0xFF。此函数将在中断服务子程序上下文中进行调用。因此，用户合并代码必须快速、无阻塞，且仅可调用重入函数。

参数： 无

返回值： uint8：用户指定的状态字节

函数值	说明
SMBusSlave_RET_UNDEFINED	默认返回状态

副作用： 无

void SMBusSlave_HandleBusError(uint8 errorCode)

说明： 当出现总线协议错误时，此API将被此组件调用。总线错误的示例有：无效的指令、数据下溢和时钟延展冲突。此函数仅负责错误的后果，因为此组件将以确定的方式处理错误。此函数主要用于通知用户固件已出现错误。例如，在PMBus器件中，这会向用户固件提供在STATUS_CML寄存器中设置相关错误的机会。

参数： uint8 errorCode：

函数值	说明
SMBusSlave_ERR_READ_FLAG	读取标志设置错误
SMBusSlave_ERR_RD_TO_MANY_BYTES	主机尝试读取大量字节
SMBusSlave_ERR_WR_TO_MANY_BYTES	主机尝试写入大量字节
SMBusSlave_ERR_UNSUPORTED_CMD	不支持接收的指令
SMBusSlave_ERR_INVALID_DATA	收到的数据无效
SMBusSlave_ERR_TIMEOUT	出现了总线复位超时
SMBusSlave_ERR_WR_TO_FEW_BYTES	主机尝试写入少量字节

返回值： 无

副作用： 无

uint8 SMBusSlave_StoreUserAll(char * flashRegs)

说明： 此函数将RAM寄存器存储区保存至闪存中的用户寄存器存储区。寄存器存储区数据结构中的CRC字段将在保存之前进行重新计算和更新。默认情况下，此函数不会向闪存执行任何存储。相反，它包含合并区域，在此区域中用户可以执行将工作内存存储至闪存的算法。

参数： flashRegs：

指向闪存中应存储工作内存（RAM）所在位置的指针。

返回值： uint8：以下标准返回状态之一

函数值	说明
CYRET_SUCCESS	成功完成操作
CYRET_MEMORY	出现内存问题

副作用： 无

uint8 SMBusSlave_RestoreUserAll(char * flashRegs)

说明： 此函数验证用户寄存器存储区的CRC字段，并将用户寄存器存储区的内容复制到RAM寄存器存储区中。默认情况下，此函数不会执行从闪存恢复寄存器存储区。相反，它包含合并区域，在此区域中用户可以执行将数据恢复至工作内存（RAM）的算法。

参数： flashRegs：

指向闪存中存储工作内存（RAM）所在位置的指针

返回值： uint8：以下标准返回状态之一。

函数值	说明
CYRET_SUCCESS	CRC匹配，因此工作内存更新用户寄存器存储器（Flash）中的内容
CYRET_BAD_DATA	数据已损坏。CRC不匹配

副作用： 无

uint8 SMBusSlave_EraseUserAll(void)

说明： 通过调用该函数，用户可以擦除闪存中的用户存储区。该函数含有合并区域，因此用户必须执行它自己的机制来擦除闪存中用户储存区的内容。

参数： 无

返回值： uint8：以下标准返回状态之一。

函数值	说明
CYRET_SUCCESS	成功完成操作

或用户定义的其他非成功状态。

副作用： 无

uint8 SMBusSlave_RestoreDefaultAll(void)

说明： 此函数验证默认寄存器存储区的签名字段，并将默认寄存器存储区的内容复制至RAM寄存器存储区中。

参数： 无

返回值： uint8：以下标准返回状态之一。

函数值	说明
CYRET_SUCCESS	成功完成操作
CYRET_BAD_DATA	数据已损坏。CRC不匹配

副作用： 无

uint8 SMBusSlave_StoreComponentAll(void)

说明： 用户调用此函数以使用当前PMBus设置更新系统中的其他组件的参数。因为此参数是非常特定于应用程序的，所以此函数几乎完全由合并部分组成。组件提供的唯一一个固件是返回值变量（retval），它将被初始化为CYRET_SUCCESS且在此函数结束时被恢复。此函数的其余部分必须由用户提供。

参数： 无

返回值： uint8：以下标准返回状态之一。

函数值	说明
CYRET_SUCCESS	成功完成操作

或用户定义的其他非成功状态。

副作用： 无

uint8 SMBusSlave_RestoreComponentAll(void)

说明： 用户调用此函数以使用系统中的其他组件的当前配置参数更新PMBus工作寄存器存储区。因为此参数是非常特定于应用程序的，所以此函数几乎完全由合并部分组成。组件提供的唯一一个固件是返回值变量（retval），它将被初始化为CYRET_SUCCESS且在此函数结束时被恢复。此函数的其余部分必须由用户提供。

参数： 无

返回值： uint8：以下标准返回状态之一。

函数值	说明
CYRET_SUCCESS	成功完成操作

或用户定义的其他非成功状态。

副作用： 无

float SMBusSlave_Lin11ToFloat (uint16 linear11)

说明： 此函数将参数“linear11”转换为浮点并将其返回。

参数： uint16 linear11：使用LINEAR11格式的数字。

返回值： float：被转换为浮点的linear11参数

副作用： 无

uint16 SMBusSlave_FloatToLin11 (float floatvar)

说明： 此函数使用参数“floatvar”（浮点编号）并将其转换为16位LINEAR11值（11位尾数+5位指数），同时将其返回。

参数： float floatvar：浮点编号

返回值： uint16：被转换为LINEAR11的floatvar

副作用： 无

float SMBusSlave_Lin16ToFloat(uint16 linear16, int8 inExponent)

说明： 此函数将参数“linear16”转换为浮点并将其返回。参数Linear16包含尾数。参数inExponent是转换中要使用的5位二进制补码指数。

参数： uint16 linear16：LINEAR16数字的16位尾数。

int8inExponent：LINEAR16数字的5位指数。在低5位中。2的补码。

返回值： float：被转换为浮点的参数

副作用： 无

uint16 SMBus_FloatToLin16(float floatvar, int8 outExponent)

说明： 此函数使用参数“floatvar”（浮点编号）并将其转换为16位LINEAR16值（16位尾数），同时将其返回。参数outExponent是转换中要使用的5位二进制补码指数。

参数： floatfloatvar：要转换为LINEAR16的浮点编号。

int8outExponent：转换中要使用的用户提供的5位指数。

返回值： uint16：被转换为LINEAR16的参数。

副作用： 无

Bootloader 支持

SMBus 和 PMBus 从器件组件可作为 Bootloader 的通信组件使用。有关 Bootloader 的更多信息，请参考*《系统参考指南》*中的“Bootloader 系统”一节。

SMBus 和 PMBus 从器件组件为使用 Bootloader 提供了一组 API 函数。

函数	说明
SMBusSlave_CyBtldrCommStart	启动SMBus和PMBus从器件组件，并使能其中断。
SMBusSlave_CyBtldrCommStop	禁用SMBus和PMBus从器件组件并禁用其中断。

函数	说明
SMBusSlave_CyBtldrCommReset	将读取和写入I²C缓冲区设置为其初始状态，然后重置从设备状态。
SMBusSlave_CyBtldrCommWrite	允许调用程序将数据写入bootloader主机。该函数将处理轮询以便让数据块完全发送至主机器件。
SMBusSlave_CyBtldrCommRead	允许调用程序读取bootloader主机中的数据。该函数将处理轮询以便从主机器件完全接收到数据块。

void SMBusSlave_CyBtldrCommStart(void)

说明： 启动通信组件并使能中断。读取缓冲区初始状态为已满，读取始终为0xFFu。写入缓冲区已清理，可接收指令。

参数： 无

返回值： 无

副作用： 此函数使能组件中断。如果已使能I²C而未使能中断，它将锁定总线。

void SMBusSlave_CyBtldrCommStop(void)

说明： 禁用通信组件并禁用中断。

参数： 无

返回值： 无

副作用： 无

void SMBusSlave_CyBtldrCommReset(void)

说明： 将缓冲区设置为初始状态，并使从状态复位。读取缓冲区初始状态为已满，读取始终为0xFFu。写入缓冲区已清理，可接收指令。

参数： 无

返回值： 无

副作用： 无

cystatus SMBusSlave_CyBtldrCommRead(uint8 * Data、uint16 size、uint16 * count、uint8 timeOut)

说明： 接收来自主机的指令。所有字节由I²C中断服务子程序接收并存储在内部I²C缓冲区中。此函数使用超时检查状态以确定传输结束，然后将数据复制到bootloader缓冲区。退出此函数后，I²C中断服务子程序能够接收更多的数据。

参数： uint8 *Data：指向要从bootloader主机读取的数据块的存储的指针

uint16 size：要读取的字节数

uint16 *count：指向用于写实际读取字节数的变量的指针

uint8 timeOut：等待的单位数（时间为10毫秒），之后会因超时而返回

返回值： cystatus：如果未遇到任何问题则返回CYRET_SUCCESS，或是返回对该问题描述最准确的值。有关更多信息，请参考*《系统参考指南》*中的“返回代码”一节。

副作用： 无

cystatus SMBusSlave_CyBtldrCommWrite(uint8 * Data、uint16 size、uint16 * count、uint8 timeOut)

说明： 将已执行指令的状态传送至主机。此函数使用响应更新I²C读取缓冲区并将其释放给主机。所有读取都返回0xFF，直至缓冲区被释放。所有字节由I²C中断服务子程序传送。此函数超时等待，直至已读取所有字节。退出此参数后，所有读取返回0xFF。

参数： uint8 *Data：指向要写入bootloader主机的数据块的指针

uint16 size：要写入的字节数

uint16 *count：指向实际写入字节数的变量指针

uint8 timeOut：等待的单位数（时间为10毫秒），之后会因超时而返回

返回值： cystatus：如果未遇到任何问题则返回CYRET_SUCCESS，或是返回对该问题描述最准确的值。有关更多信息，请参考*《系统参考指南》*中的“返回代码”一节。

副作用： 在“Bootloader Write”数据操作期间调用启动时钟延展时，临时使能组件中断，但在返回之前通过此函数禁用此中断（以符合手动指令处理行为）。当不使用“Bootloader Write”进行调用时 - 始终保留组件中断已使能。

宏

- SMBusSlave_FL_ADDR_TO_ROW(addr) — 从指定地址提取闪存行数。
- SMBusSlave_FL_ADDR_TO_ARRAYID(addr) — 从指定地址提取闪存阵列 ID。
- SMBusSlave_SIZE_TO_ROW(size) — 计算并返回存储 ***size*** 定义的数据量所需的闪存行数。
- SMBusSlave_MAX_PAGES — 指定分页指令所使用的最大页数。

- SMBusSlave_NUM_COMMANDS — 定义分页指令的页数。

MISRA 合规性

本节介绍了 MISRA-C:2004 合规性和本组件的偏差情况。有两种差异的类型，如下定义：

- 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”章节中介绍了项目偏差以及有关 MISRA 合规性验证环境的信息。

还未证明 SM 总线和 PM 总线从器件组件源代码符合 MISRA-C:2004 编码准则。

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** 项来限定可选的项目列表。

更多有关信息，请参考 PSoC Creator 帮助中的“Find Example Project”（查找示例项目）主题。

中断服务子程序

SMBus 和 PMBus 从器件组件使用 2 个中断服务子程序进行操作。一个中断服务子程序处理通过 SM/PM 总线的指令解码和数据操作。另一个中断服务子程序是 25 ms 超时中断，设计用于在出现低电平停滞情况时复位总线。

功能说明

此组件的工作原理与 I²C 从器件组件的工作原理非常相似。所有 SMBus 规范均参考版本 2.0 的 SMBus 规范。本节将重点介绍此组件的关键功能。

I²C 物理层

SM/PM 总线从器件组件的物理层是基于 I²C 协议的。影响此组件的三个主要区别是：

SMBus 规范强制规定，如果检测到 SCL 信号低电平停滞 25 ms，此组件必须复位并释放 SCL 和 SDA 线。该内容在《交流和直流电气性能要求》一节中有更加详细的说明。此组件还监控 SDA 线，并且如果其仍保持低电平停滞状态 25 ms，作为额外防范措施，还将对其进行复位。

SMBus 规范强制规定此组件不得在任何给定的传输中延展时钟超过 25 ms（累积时间）。如果任何数据操作中的累积延展时间未超过 25 ms，当此从器件忙碌于将 SCL 拉至低电平（时钟延展）时，允许此从器件延迟传输

增加 SMBALERT#引脚以通知主机此器件需要引起关注

SMBus/PMBus 寻址

每个 SMBus/PMBus 从器件都有一个 I^2C 地址。以下地址留作特定的 SMBus 用途，且不得作为 SMBus/PMBus 从器件的通用从器件地址使用。

从器件地址（位7:1）	R/W#位（位0）	注释
0000 000	0	通用调用地址
0000 000	1	START字节
0000 001	X	CBUS地址
0000 010	X	留作用于其他总线格式
0000 011	X	留作日后使用
0000 1XX	X	留作日后使用
0101 000	X	留作用于ACCESS总线主机
0110 111	X	留作用于ACCESS总线默认地址
1111 0XX	X	留作用于10位从器件寻址
1111 1XX	X	留作日后使用
0001 000	X	留作用于SMBus主机
0001 100	X	SMBus警报响应地址
1100 001	X	SMBus组件默认地址

如果用户使能了 SMBALERT#引脚选项，此组件将响应其主要地址以及警报响应地址。

SMBus/PMBus 协议

SMBus 规范中定义了 9 个不同的协议。这些协议的概要及其支持状态如下所示。

协议	支持状态	注释
快速指令	不支持	仅SMBus
发送字节	支持	两者
接收字节	支持	仅SMBus
写入字节/字	支持	两者
读取字节/字	支持	两者
进程调用	支持	两者
模块写入/读取	支持	两者
模块写入/模块读取进程调用	支持	两者
SMBus主机通知协议	不支持	两者

体现 SM/PM 总线组件特性的关键概念如下。

SMBus 和 PMBus 都不使用子地址或 I²C 寄存器映射的概念。所有传输都是基于指令的。I²C 从器件地址后面紧跟的是指令代码。指令可以是只写、只读或读/写指令。在 SMBus 中，指令的定义以及读/写限制几乎完全取决于用户。在 PMBus 中，指令很大程度上是预先定义的，但提供了一系列的指令代码，这些指令代码未进行分配，可用于“特定于制造商的”实现。

对于读取数据操作，主机通过发出 REPEATED START 总线条件在编写读取指令代码与读取请求数据之间来回切换。一旦整个数据操作完成，将仅生成 STOP 位。如下例所示：

Read Byte Protocol

1	7	1	1	8	1	1	7	1	1	8	1	1
S	Slave Address	Wr	A	Command Code	A	Sr	Slave Address	Rd	A	Data Byte	A	P

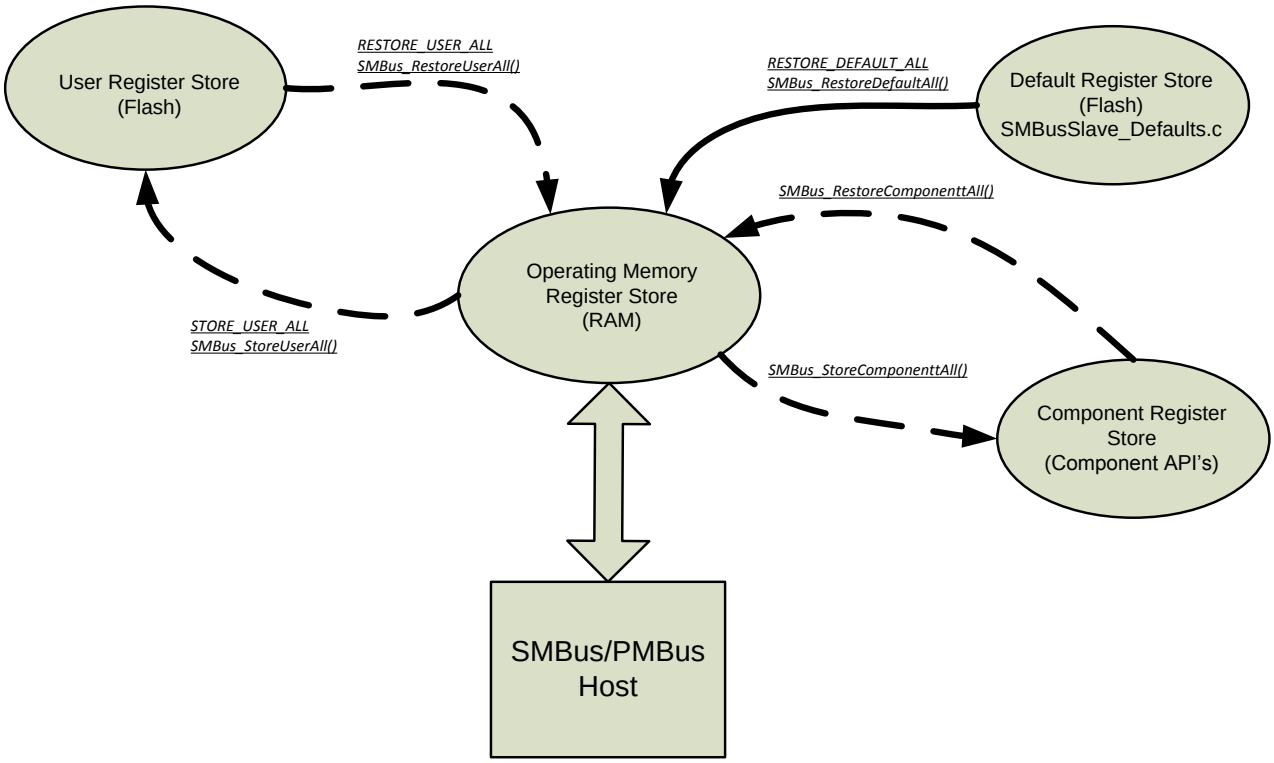
SMBus 和 PMBus 是低位优先协议（即在总线上将首先传送多字节数据类型的最低有效字节）。从 PSoC 角度需要所有寄存器以 PSoC 固有的字节顺序格式存储。此组件负责在 PSoC 与 I²C 缓冲区之间正确地转换字节顺序。这仅适用于 16 位字传输。

SMBus 规范不定义任何指令代码；它们由用户进行定义。但是，PMBus 规范第 2 部分附录 1 定义了所有 255 个可选指令代码；其中 46 个由用户进行定义（被称为特定于制造商的指令）。

寄存器存储区概念

根据 PMBus 规范 II 5.4.2 — “可写入的每个参数必须可读”。通常，接受写入的值的任何指令在读取时也必须返回此值。为此，使用了寄存器存储区的概念。

图 1. 寄存器存储区概念



工作内存寄存器存储区

这是 RAM 版本的寄存器存储区。运行时，SMBus/PMBus 指令可修改此版本的寄存器存储区。由于此寄存器存储区位于 RAM 中，因此在复位时其内容被假定为无效。

默认寄存器存储区

闪存版本的寄存器存储区，包含所有 SMBus/PMBus 参数的默认值。启动时，默认寄存器存储区可被复制至工作内存寄存器存储区以初始化工作寄存器存储区。默认寄存器存储区中的参数值在编译/链接期间是固定的。用户负责为寄存器存储区中的参数提供默认值。要打开 *SMBusSlave_Defaults.c*，从注释模块中复制参数并将其粘贴到下面的合并区域内。如果您这样做，它们将存储在 SMBusSlave_regsDefault 中，且每次启动时，工作内存将使用这些参数进行初始化。另一种使用默认值初始化工作内存的方式是调用 SMBus_RestoreDefaultAll()。此函数可在用户处理程序用于 RESTORE_DEFAULT_ALL (0x12) PM 总线指令。

用户寄存器存储区

这是另一个闪存版本的寄存器存储区。不同于默认寄存器存储区，它基本上是只读的，而用户寄存器存储区则可以基于工作寄存器存储区的当前内容进行更新。由于将数据存储至闪存中的操作是特定于实现的，因此此组件不会执行任何存储至闪存中的操作；它提供了两个包括合并区域的 API 功能，此区域可用于此目的。SMBusSlave_StoreUserAll()执行检查恢复时的数据有效性所需的 CRC 的计算。

要设计您自己的将数据存储至闪存的数据的方法，请参考*《系统参考指南》*。第 10 章提供了基本信息以及用于处理闪存的指令说明。此组件中还提供了一组宏，在处理闪存时可使用这些宏。

组件寄存器存储区

组件寄存器存储的主要动机是允许 PMBus 从标准的 PSoC Creator 组件中提取参数并配置这些组件（因此名称为组件寄存器存储区）。Creator 组件都有其自己的配置参数，这些参数通常是使用组件自定义程序设置的。在运行时可通过特定于组件的 SMBusSlave_StoreComponentAll()/SMBusSlave_RestoreComponentAll() API 来访问这些参数。可通过 API 访问的组件参数包括组件寄存器存储区。启动时，用户 PMBus 固件可能需要：

- 基于用户或默认寄存器存储区中存储的 PMBus 参数更新其他组件设置，或
- 基于组件设置更新 PMBus 寄存器存储区。

特例指令

PAGE 指令

PMBus PAGE（代码 0x00）指令（PMBus 第 2 部分 — 第 11.10 节）允许访问同一个 PMBus I^2C 地址下的多个逻辑 PMBus 器件。例如，控制多个电源轨的 PMBus 功耗监控器可提供对每个轨在其自己的页面上的指令/参数的访问。可将该页面视为到指令/寄存器的阵列的索引。一旦通过 PAGE 指令设置了此页面，页面设置将是持久性的，直至再次被另一个 PAGE 命名设置。

在 PMBus 模式中，此组件具有对 PAGE 指令的内置支持。在 SMBus 模式中，用户可以选择使能 PAGE 指令并为 PAGE 指定要使用的指令代码。

如果使能了 PAGE 指令，则 PAGE 指令值的有效范围为 1 到 32 之间，且必须在用户确定的 **Max Page**（最大页面）设置范围内。异常情况是“All Pages”（所有页面）通配符设置，为 0xFF。“All Pages”（所有页面）通配符仅针对写入数据操作有效，且必须在“Manual”（手动）模式中进行处理。如果 PAGE 被设置为 0xFF，以下数据操作将被视为错误：

- 通过分页指令尝试读取
- 通过配置为 Manual（手动）的分页指令尝试写入

甚至在包括多个页面的 PMBus 器件中，一些指令不会独立于当前页面，且始终按相同的方式进行处理，与 PAGE 设置情况无关。例如，用于返回此组件支持的 PMBus 版本的 PMBUS_REVISION 指令对于所有页面是通用的。因此，存在页面指令和常见指令的概念。对于每个 SMBus/PMBus 指令，自定义程序允许用户指定指令是 Paged（分页）指令还是 Common（常见）指令。当指令被标记为 Paged（分页）时，此组件将在寄存器存储区中为此指令定义一个阵列。

QUERY 指令

PMBus QUERY（代码 0x1A）指令用于询问 PMBus 器件是否支持所给出的指令，以及如果支持的话它针对此指令支持哪些数据格式。在 PMBus 模式中，此组件具有对 QUERY 指令的内置支持。在 SMBus 模式中，用户可以选择使能 QUERY 指令并为 QUERY 指定要使用的指令代码。

此指令使用 SMBus 规范中介绍的模块写入模块读取进程调用。针对进程调用的写入部分，一个数据字节是未分配的二进制整数，此整数等于正在被操作的指令的指令代码。针对进程调度的读取部分，一个数据字节是未分配的二进制整数，其包括以下值：

位	数值	含义
7	1	支持指令
	0	不支持指令
6	1	支持指令写入
	0	不支持指令写入
5	1	支持指令读取
	0	不支持指令读取
4:2	000	使用了线性数据格式
	001	16位有符号数字
	010	保留
	011	使用了直接模式格式
	100	8位无符号数字
	101	使用了VID模式格式
	110	使用了特定制造商的格式
	111	指令不返回数字数据。这还用于返回各个数据区块的指令。
1:0	XX	留作日后使用

可以基于自定义程序中的用户选择生成上表中的所有信息。如果不支持使用 QUERY 研究的指令代码，在这种情况下，将返回“0x00”。

数据操作队列

对于任何还没指定为 AUTO 的 SMBus/PMBus 指令，则必须在主程序上下文中进行及时处理。要实现这一操作，该组件将保持一个数据操作队列，从而可通过您的代码在 I^2C ISR 上下文之外对总线数据操作进行处理。

只要检测到“Manual”（手动）类型的指令，则将它录入数据操作队列，然后它应由用户代码进行处理。数据操作队列的记录具有以下结构：

```
typedef struct
{
    uint8 read;         /* r/w flag - 1=read 0=write */
    uint8 commandCode; /* SMBus/PMBus command code */
    uint8 page;         /* SMBus/PMBus page */
    uint8 length;       /* bytes transferred */
    uint8 payload[65]; /* payload for the transaction */
} TRANSACTION_STRUCT
```

下面说明了各字段的内容：

- “read”是一个标志，表示接收到 Read（读取）或 Write（写入）的指令类型。
- “commandCode”是目前接收到的指令的 1 字节指令代码
- “page”是目前接收到的指令的页码

  它仅适用于分页指令。对于常见指令，该字段将是零。
- ″length″
  - 对于 Write（写入）指令，“length”指定了目前接收指令的数据长度（单位为字节）。
  - 对于 Read（读取）指令，“length”指定需要发送的字节数。
  - 对于模块指令，“length”包括了“字节数”字节。
- ″payload″
  - 对于 Write（写入）指令，“payload”包含目前指令的接收数据。用户代码用于更新寄存器存储区，然后对 SMBusSlave_CompleteTransaction()函数进行调用。
  - 对于 Read（读取）指令，“payload”即不可用。用户代码用于更新寄存器存储区中该指令的变量，然后调用 SMBusSlave_CompleteTransaction()函数。

每次接收手动指令，该组件将开始伸展时钟，直至处理待定数据操作，以及 SMBusSlave_CompleteTransaction()被调用。对于 Read（读取）指令，重复开始后，时钟将开始伸展，并等待用户代码为外部主机提供响应。对于 Write（写入）指令，接收目前指令的数据

后，时钟将开始伸展。当时钟开始伸展时，内部定时器也开始计数 25 ms 超时。如果在发生超时前，未调用 SMBusSlave_CompleteTransaction()函数，则该组件将复位总线。因此，这将使数据操作队列中剩余的记录无效。

Bootloader 指令

当此组件被安置在“Bootloader”项目中，Configure 对话中的 **Custom Commands**（自定义指令）选项卡上将有两个额外的指令可用。这两个指令是 BOOTLOAD_READ 和 BOOTLOAD_WRITE，它们的默认指令代码分别为 0xFD 和 0xFC。它们向此组件增加了一项作为 Bootloader 组件的通信组件的功能。

这两个指令可与设计原理图上安置的 Bootloader 组件进行交互。在安装 Bootloader 后，在 Bootloader 组件的 Configure 对话框中选择“Custom interface”（自定义界面）。

资源

下面列出了固定功能实现的 SM/PM 总线从器件组件的资源图表。这些数据是数据速率为 400 kbps 时收集的。

PSoC 3

配置	资源类型					
	数据路径单元	宏单元	状态单元	控制单元	I²C固定模块	中断
SM总线/PM总线（UDB）	3	32	2	6	__	2
SM总线/PM总线（固定功能）	2	9	1	4	1	2

PSoC 5LP

配置	资源类型					
	数据路径单元	宏单元	状态单元	控制单元	I²C固定模块	中断
SM总线/PM总线（UDB）	3	32	2	6	__	2
SM总线/PM总线（固定功能）	1	5	2	2	1	2

API 存储器大小

组件存储器使用的大小显著不同，它取决于编译器、设备、所使用的 API 数量以及组件配置。下表提供的是已给组件配置中的所有 API 存储器大小。

此次测量使用的编译器配置是针对代码大小优化的 Release（发布）模式有关特定的设计，可分析编译器生成的映射文件以确定内存使用情况。

配置	PSoC 3 （Keil_PK51）		PSoC 5LP（GCC）	
	闪存 字节	SRAM 字节	闪存 字节	SRAM 字节
SM总线从器件（固定功能）	5554	203	3844	142
SM总线从器件（UDB）	5628	199	3956	132
PM总线从器件（固定功能）	7539	1795	5085	3326
PM总线从器件（UDB）	7613	1791	5189	3316

直流和交流电的电气特性

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。

SMBus 和 PMBus 从器件直流电规范

参数	说明		最小值	典型值[1]	最大值	单位
I_{DD}	组件电流消耗					
	SMBus (UDB)	数据速率 = 50kbps	—	380	—	µA
	SMBus（固定功能）		—	670	—	µA
	PMBus (UDB)	数据速率 = 100kbps	—	440	—	µA
	PMBus（固定功能）		—	620	—	µA
	SMBus (UDB)	数据速率 = 400kbps	—	720	—	µA
	SMBus（固定功能）		—	860	—	µA

[1] 未包括设备 IO 和时钟分配的电流。这些值是在温度是 25 °C 时的值。数据是在 BUS_CLK 被设置为 24 MHz 时测得的。

参数	说明		最小值	典型值[1]	最大值	单位
	PMBus (UDB)	数据速率 = 400kbps	—	750	—	µA
	PMBus（固定功能）		—	980	—	µA

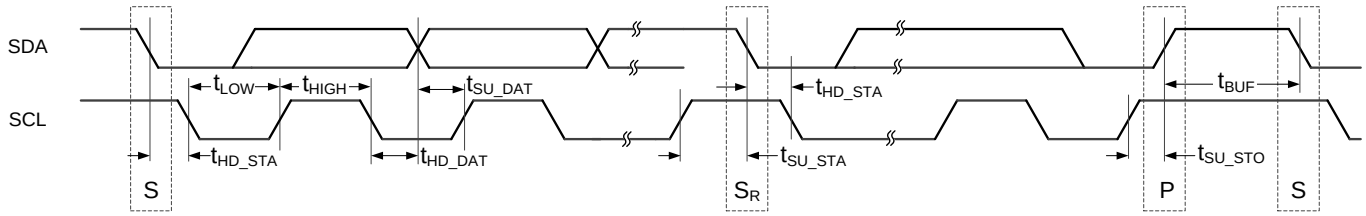
SMBus 和 PMBus 从器件交流电规范

参数	说明	最小值	典型值	最大值	单位
f_{SCL}	SCL时钟频率	— — —	— — —	100 400 1000	kHz
f_{CLOCK}	组件输入时钟频率	—	16 × f_{SCL}	—	kHz
t_{RESET}	复位信号脉冲宽度	—	2	—	t_{CY_clock}[2]
t_{LOW}	SCL时钟的低电平周期	4.7 1.3 0.5	— — —	— — —	µs
t_{HIGH}	SCL时钟的高电平周期	4.0 0.6 0.26	— — —	— — —	µs
t_{HD_STA}	（重复）启动条件后的保留时间	4.0 0.6 0.26	— — —	— — —	µs
t_{SU_STA}	重复启动条件的建立时间	4.7 0.6 0.26	— — —	— — —	µs
t_{HD_DAT}	数据保留时间	5.0 — —	— — —	— — —	µs
t_{SU_DAT}	数据建立时间	250 100 50	— — —	— — —	ns
t_{SU_STO}	停止条件的建立时间	4.0 0.6 0.26	— — —	— — —	µs

[2] $t_{CY_clock} = 1/f_{CLOCK}$。这是一个时钟周期的时间长度

参数	说明	最小值	典型值	最大值	单位
t_{BUF}	停止和启动之间的总线空闲时间	4.7 1.3 0.5	— — —	— — —	µs
t_{FLASH_WRITE}		—	40.6[3]	—	ms

图 2. 数据转换时序图



组件更改

本节列出了各版本的主要组件更改内容。

版本	更改内容	更改原因/影响
2.20	修复了有关在尝试读取“仅写入”指令时组件传送“垃圾”数据（而不是FFs）的问题。	该问题是由ISR组件中处理指令读取部分的错误条件引起的。
	从组件删除了对PSoC 5系列器件的支持。	
	使用新值更新了API存储器占用表。	
	添加了数据操作队列的说明。	需要数据操作队列的信息，以处理手动类型的指令。
	修复了将页码错误地设置为最大页码值的问题。	因为页面索引是从零开始的，所以最大页码应等于最大页码值减去1。
	解决了有关时钟输入的错误存在问题。	当在组件定制器中选中了固定功能实现时，组件不能在符号上公开时钟输入，因为在该模式下，组件只能使用内部时钟。

[3] 通过以下方式获取的值：使用所有 PM 总线指令存储为 PM 总线模式配置的寄存器存储区，同时这些指令是使用 BUS_CLK 为 24 MHz 时的默认配置使能的。

版本	更改内容	更改原因/影响
	将定制器中“General”选项卡下的“Actual data rate”（实际数据速率）改为“Attainable data rate”（可达到的数据速率）。	
2.10	纠正了无法将PAGE设置为0xFF（“所有页面”通配符）的相关问题。	
2.0	必须始终使用分页指令陈列来声明PMBus寄存器存储区，即使用户在设计中只能选择一个页面。	导致代码重组的错误，
	将“Pages”列添加到自定义指令表格内。	这是为特殊定义指令定义页数的参数。
	纠正了有关未完成写入数据操作的问题。	通过修改该代码来验证“Byte count”（字节计数）字段所指定的字节数是否等于接收数据的字节数。修改前的代码验证了数据数量是否等于特殊数据块指令中的“Size”参数，用户通过组件自定义程序访问这些指令。
	清除SMBALERT引脚保持未连接状态时所导致的编译错误。	
	页面设置为“所有页面”通配符情况下，当处理页面索引和手动指令时，纠正了有关出错生成总线错误的问题。	
	限制从器件地址将地址留作特殊的SMBus用途。	这是一个错误，因为组件没有验证该地址，
	修复了一些小问题。	
	添加了新函数 — SMBusSlave_EraseUserAll()。	
1.10	已添加MISRA合规性章节。	此组件未进行MISRA合规性验证。
	将I2C和控制寄存器组件的最新版本更新到SM总线和PM总线从器件。	
1.0	第一版	

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。