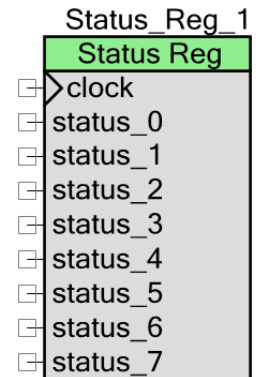


状态寄存器

1.80

特性

- 最高达 8 位的状态寄存器
- 中断支持



概述

状态寄存器允许固件读取数字信号。

何时使用状态寄存器

当固件需要查询内部数字信号的状态时，将使用状态寄存器。

输入/输出连接

本节描述了状态寄存器的输入连接。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。

时钟 — 输入

状态寄存器时钟。当位配置为透明时，时钟输入信号被忽略。

status_0 - status_7 — 输入*

状态寄存器输入。通过读取状态寄存器，固件可查询输入信号。输入的数量取决于 **Inputs**（输入）参数。这些输入可能被保留为悬空，没有与外部连接。如果这些线路上没有任何连接，则组件将分配一个常量逻辑 0。

status[N:0] — 输入*

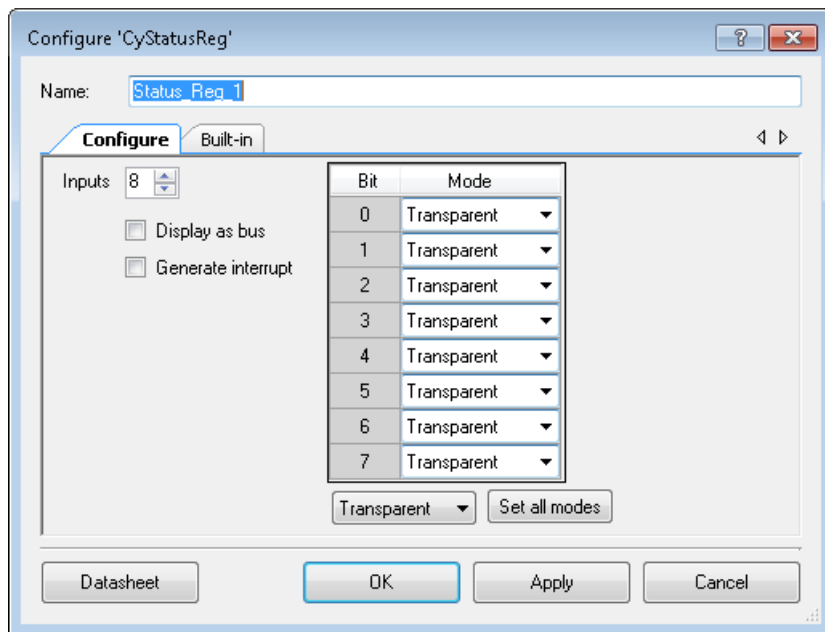
该可选输入可将各个单独的输入端并入一个单总线端。使能 **Display as bus**（显示为总线）参数后，该引脚可见。N 的值等于输入数量减去 1。该输入可悬空，而不与任何外部连接。如果该线路没有任何连接，组件将分配一个常量逻辑 0。

intr — 输出*

使能 **Generate interrupt**（生成中断）参数后，符号上会显示该可选引脚。该选项仅在所选的输入数量少于 8 时才有效。

组件参数

将状态寄存器拖入到你的设计中，双击它以打开 **Configure** 对话框。



Inputs（输入）

输入端的数量（1 至 8）。默认值为 8。

Display as bus（显示为总线）

该参数将输入显示为总线，而不是各个单独端。该选项默认为未选中。

Generate interrupt（产生中断）

该参数会在符号上显示中断输入。该选项默认为未选中。仅在输入数量少于 8 时，中断才有效。



Set all modes（设置所有模式）

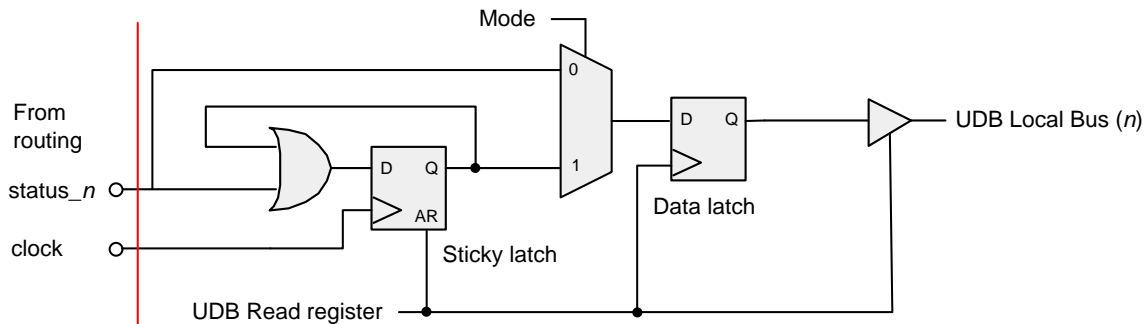
该按键会将所有位设置为 **Transparent**（透明）或 **Sticky**（粘滞）模式，具体情况取决于该按键左侧组合框中所选的模式。

Mode（模式）

这些参数用于设置状态寄存器的特定位，以在寄存后保持高电平，直到执行读取操作时为止。该读取操作会清除所有寄存值。这些设置如下所述：

- **Transparent**（透明）— 默认情况下，CPU 读写寄存器时将透明地读取相关布线网络的状态，它与模块时钟是异步的。该模式可用于 UDB 内部计算和寄存的瞬变状态。
- **Sticky (Clear on Read)**（粘滞（读取时清除））— 在该模式下，在状态和控制时钟的每个周期内对布线网络进行相关采样。如果信号在指定的采样中为高电平，则在状态位中被捕获，并保持高电平，而不考虑相关布线的后续状态。CPU 固件读取状态寄存器时，会清除该位。状态寄存器的清除不依赖于任何模式，即使在模块时钟被禁用时也同样发生；它基于总线时钟，并作为读取操作的一部分而发生。

图 1. 透明与粘滞模式的行为特性



Interrupt mask（中断掩码）

该选项仅在选中 **Generate interrupt**（产生中断）时才会显示在 **Configure** 对话框中。使用这些参数，可为状态寄存器中的每一位设置中断掩码值。中断掩码的默认值为 0。

低功耗模式状态

在各种低功耗模式（睡眠、深睡眠和休眠模式）下，将不保留状态寄存器的数据。当器件从低功耗模式中唤醒时，状态寄存器组件的每一位的初始值为 ‘0’。

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。

默认情况下，PSoC Creator 将实例名称“Status_Reg_1”分配给指定设计中状态寄存器的第一个实例。您可以将组件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。为了便于阅读，下列函数中所使用的实例名称为“StatusReg”。

函数	说明
StatusReg_Read()	读取状态寄存器的当前值
StatusReg_InterruptEnable()	使能状态寄存器中断
StatusReg_InterruptDisable()	禁用状态寄存器中断
StatusReg_WriteMask()	对分配给掩码寄存器的值进行写操作
StatusReg_ReadMask()	从掩码寄存器返回当前中断掩码值

uint8 StatusReg_Read (void)

- 说明：** 读取状态寄存器的值。
- 参数：** 无
- 返回值：** 返回状态寄存器的当前值。
- 其他影响：** 无

void StatusReg_InterruptEnable (void)

- 说明：** 使能状态寄存器中断。默认状态被禁用。该函数仅在状态寄存器生成中断时才有效。
- 参数：** 无。
- 返回值：** 无。
- 其他影响：** 无。

void StatusReg_InterruptDisable (void)

- 说明：** 禁用状态寄存器中断。该函数仅在状态寄存器生成中断时才有效。
- 参数：** 无。
- 返回值：** 无。
- 其他影响：** 无。



void StatusReg_WriteMask (uint8 mask)

- 说明:** 对分配给状态寄存器的当前掩码值进行写操作。该函数仅在状态寄存器生成中断时才有效。
- 参数:** **mask:** 要写入掩码寄存器中的值。
- 返回值:** 无。
- 其他影响:** 无。

uint8 StatusReg_ReadMask (void)

- 说明:** 读取分配给状态寄存器的当前中断掩码值。该函数仅在状态寄存器生成中断时才有效。
- 参数:** 无。
- 返回值:** 返回中断掩码的当前值。
- 其他影响:** 无。

DMA

可以使用 DMA 组件直接读取状态寄存器的数据。可以使用 DMA 向导配置 DMA 操作，具体如下：

DMA向导中 DMA源/目标的名称	方向	DMA请求 信号	DMA请求 类型	说明
StatusReg_Status_PTR	源	不可用	不可用	存储状态寄存器的值。

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 — 仅适用于本组件的偏差

本节介绍了有关组件特定偏差的信息。在《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。

状态寄存器组件没有任何特定偏差。



固件源代码实例

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

资源

状态寄存器组件使用 UDB 阵列的一个状态单元。

API 存储器的使用情况

根据不同的编译器、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。下表提供了某一组件配置中所有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)	RAM (字节)	闪存 (字节)	RAM (字节)	闪存 (字节)	RAM (字节)
默认值	46	0	92	0	92	0

组件更改

本节列出了各版本中主要组件的更改内容。

版本	更改内容	更改原因/影响
1.80.b	更新了数据手册。	更新了图1，以便能够更好地说明透明模式与年制模式。 已添加了“低功耗模式状态”一节。
	更新了Configure对话框中的表。	修正了120 dpi分辨率中存在的问题。
1.80.a	更新了PSoC 4数据手册中有关存储器大小的内容。	

1.80	已添加了MISRA合规性章节。	该组件没有任何特定偏差。
1.70	添加了PSoC 5LP支持。	
	实现了StatusReg_InterruptEnable()、StatusReg_InterruptDisable()、StatusReg_WriteMask()和StatusReg_ReadMask()等API。	以支持中断功能。
	更新了Configure对话框。	增加了Display as bus（显示为总线）、Generate interrupt（生成中断）、Set all modes（设置所有模式）以及Interrupt mask（中断掩码）等参数，并对设计进行了少量更改。
	向输入端增加了中断引脚和显示为总线选项。还实现了DMA功能以及调试窗口支持。	目的在于将输入端作为总线使用，并支持中断生成。
1.60	更新了Configure对话框。	更改了位显示，并解决了Configure对话框中的小问题。
1.50.b	编辑了数据手册。	
1.50.a	编辑了数据手册。	
1.50	更新了Configure对话框。	创建了自定义接口。增加了“Set All”（设置所有项）按键，并更改了输入数量字段以允许键盘输入。更新了对话框，使其符合公司的标准。

©赛普拉斯半导体公司，2013。另外，这里所包含的信息可能会随时更改，恕不另行通知。除赛普拉斯产品内嵌的电路以外，赛普拉斯半导体公司不对任何其他电路的使用承担任何责任。也不根据专利或其他权利以明示或暗示的方式授予任何许可。除非与赛普拉斯签订明确的书面协议，否则赛普拉斯产品不保证能够用于或适用于医疗、生命支持、救生、关键控制或安全应用领域。此外，对于可能发生运转异常和故障并对用户造成严重伤害的生命支持系统，赛普拉斯不授权将其产品用作此类系统的关键组件。若将赛普拉斯产品用于生命支持系统中，则表示制造商将承担因此类使用而招致的所有风险，并确保赛普拉斯免于因此而受到任何指控。

PSoC®是赛普拉斯半导体公司的注册商标，PSoC Creator™和 Programmable System-on-Chip™是赛普拉斯半导体公司的商标。此处引用的所有其他商标或注册商标归其各自所有者所有。

所有源代码（软件和/或固件）均归赛普拉斯半导体公司（赛普拉斯）所有，并受全球专利法规（美国和美国以外的专利法规）、美国版权法以及国际条约规定的保护和约束。赛普拉斯据此向获许可者授予适用于个人的、非独占性、不可转让的许可，用以复制、使用、修改、创建赛普拉斯源代码的派生作品、编译赛普拉斯源代码和派生作品，并且其目的只能是创建自定义软件和/或固件，以支持获许可者仅将其获得的产品依照适用协议规定的方式与赛普拉斯集成电路配合使用。除上述指定用途外，未经赛普拉斯的明确书面许可，不得对此类源代码进行任何复制、修改、转换、编译或演示。

免责声明：赛普拉斯不针对此材料提供任何类型的明示或暗示保证，包括（但不限于）针对特定用途的适销性和适用性的暗示保证。赛普拉斯保留在不另行通知的情况下对此处所述材料进行更改的权利。赛普拉斯不对此处所述之任何产品或电路的应用或使用承担任何责任。对于合理预计可能发生运转异常和故障，并对用户造成严重伤害的生命支持系统，赛普拉斯不授权将其产品用作此类系统的关键组件。若将赛普拉斯产品用于生命支持系统，则表示制造商将承担因此类使用而招致的所有风险，并确保赛普拉斯免于因此而受到任何指控。

产品使用可能受适用的赛普拉斯软件许可协议限制。

