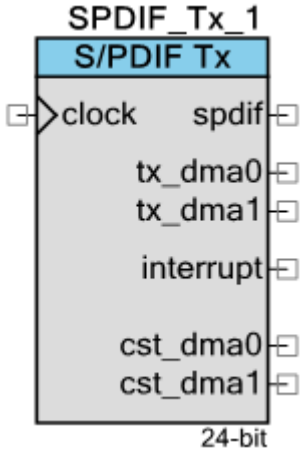
S/PDIF 发射器 (SPDIF_Tx)

1.0

特性



- 符合线性 PCM 音频传输的 IEC-60958、AES/EBU、AES3 标准
- 采样率为（时钟除以 128）（最高频率为 192 kHz）
- 可配置音频采样长度 (8/16/24)
- 消费应用的通道状态位发生器
- DMA 支持
- 独立的左声道和右声道 FIFO 或交叉立体声 FIFO

概述

SPDIF_Tx 组件可以简化在任何设计上添加数字音频输出的方法。它格式化输入的音频数据和元数据，用以创建适用于光纤或同轴数字音频的 S/PDIF 位流。该组件支持交叉及独立的音频模式。

SPDIF_Tx 组件接收来自 DMA 的音频数据及通道状态信息。在大部分时间内，通道状态 DMA 均由组件来管理；然而，可以选择单独指定此数据，以便更好地控制系统。

何时使用 SPDIF_Tx

无论何时需要 S/PDIF 发射器，SPDIF_Tx 组件均提供一个快速的解决方案，其中包括某些应用，例如：

- 数字音频播放器
- 计算机音频接口
- 音频主设备

组件的使用情况可能是：

- 编程时，PSoC 3 枚举为 USB 音频 HID。PSoC 3 是计算机声卡，它通过数字音频连接来播放。
- 该组件可以结合 I2S 使用，用外部 ADC 进行模数转换。

输入/输出连接

本节介绍 SPDIF_Tx 组件的各种输入和输出连接。I/O 列表中的星号 (*) 表示，当满足 I/O 说明中列出的条件，该 I/O 可能不可见。

时钟 — 输入

时钟速率必须是 Spdif 输出所需数据速率的 2 倍。例如，要生成 48-kHz 的音频，时钟频率将是：

2 × 48 kHz × 64 = 6.144 MHz

spdif — 输出

串行数据输出。

sck — 输出

输出串行时钟。

中断 — 输出

中断输出。

tx_DMA0 — 输出

DMA 请求音频 FIFO 0（通道 0 或交叉式）。

tx_DMA1 — 输出

DMA 请求音频 FIFO 1（通道 1）。显示是否在 **Audio Mode**（音频模式）参数下选中 **Separated**（独立）。

cst_DMA0 — 输出 *

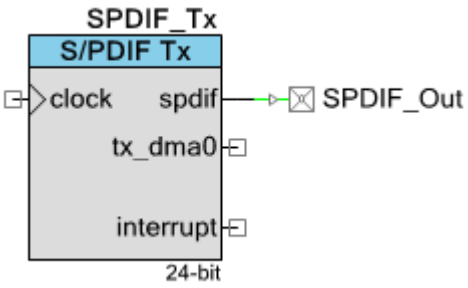
DMA 请求通道状态 FIFO 0（通道 0）。显示是否在 **Managed DMA**（管理 DMA）参数下取消选择该复选框。

cst_DMA1 — 输出 *

DMA 请求通道状态 FIFO 1（通道 1）。显示是否在 **Managed DMA**（管理 DMA）参数下取消选择该复选框。

原理图宏信息

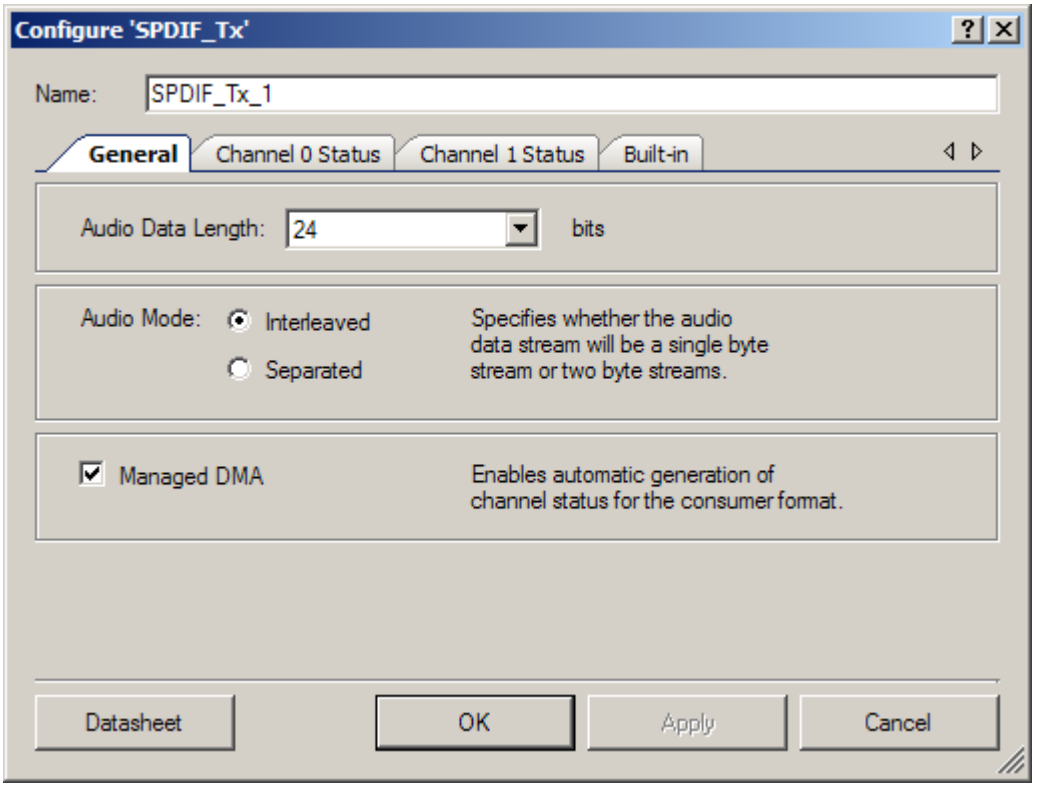
在组件目录中，默认设置的 SPDIF 发射器是使用 SPDIF 组件的原理图宏。它与数字输出引脚组件连接。针对引脚的 API 生成程序被关闭。



元件参数

将 SPDIF_Tx 组件拖入设计中，双击该组件，打开 **Configure**（配置）对话框。该对话框有 3 个选项卡，可引导您完成 SPDIF_Tx 组件的设置。

通用选项卡



音频数据长度

确定各个采样配置的数据位数（编译的硬件）。该值可设置为：8、16 或 24。默认设置为 **24**。

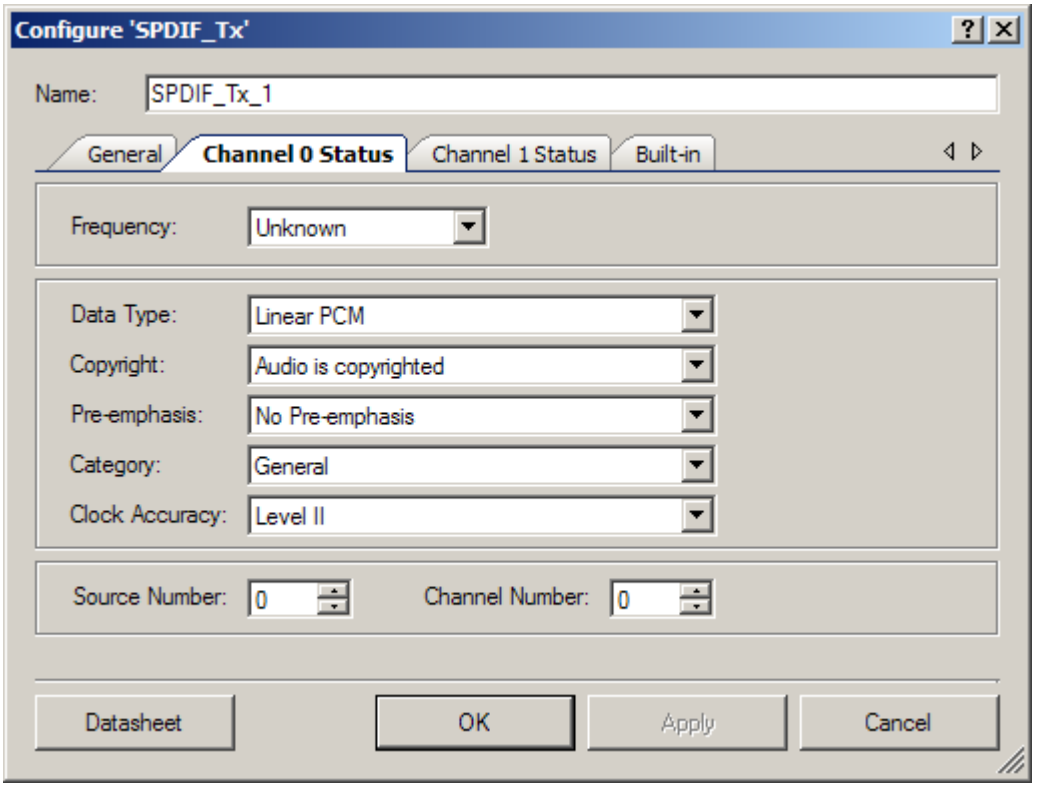
音频模式

通过此参数，您可以选择音频数据是 **Interleaved**（交叉数据）（默认设置）还是 **Separated**（独立数据）（硬件编译）。

管理的 DMA

通过此参数，您可以选择该组件是否将管理通道状态 DMA（编译的硬件）。如果选择 **Managed DMA**（管理的 DMA），则启用 **Channel 0 Status**（通道 0 状态）和 **Channel 1 Status**（通道 1 状态）选项卡。默认情况下，系统启用该选项。

Channel 0 Status（通道 0 状态）选项卡



频率

通过此字段，您可以为指定频率选择通道状态值。此值应用于双通道。源频率可能是：22 kHz、24 kHz、32 kHz、44 kHz、48 kHz、64 kHz、88 kHz、96 kHz、192 kHz 或未知。默认值为 **Unknown**（未知）。

数据类型

为通道状态 0 指定数据类型值。此值可以设置为 **Linear PCM**（线性 PCM）（默认值）或其他数据。

版权所有

通过此字段，您可以选择 **Audio is copyrighted**（音频受版本保护）（默认值）或 **Audio is not copyrighted**（音频不受版权保护）。

预加重

确定通道状态 0 的 PCM 预加重值。该值可设置为：**未加重**或 **50/15 µs**。默认设置为 **No Pre-emphasis**（未预加重）。

类别

指定通道状态 0 的类别类型。此值可设置为 **General**（常规）（默认值）或 **Digital to Digital**（数字到数字）。

时钟精度

通过此字段，您可以选择通道状态 0 的时钟精度。该值可设置为：**Level I**（电平 I）、**Level II**（电平 II）或 **Level III**（电平 III）。默认设置为 **Level II**（电平 II）。

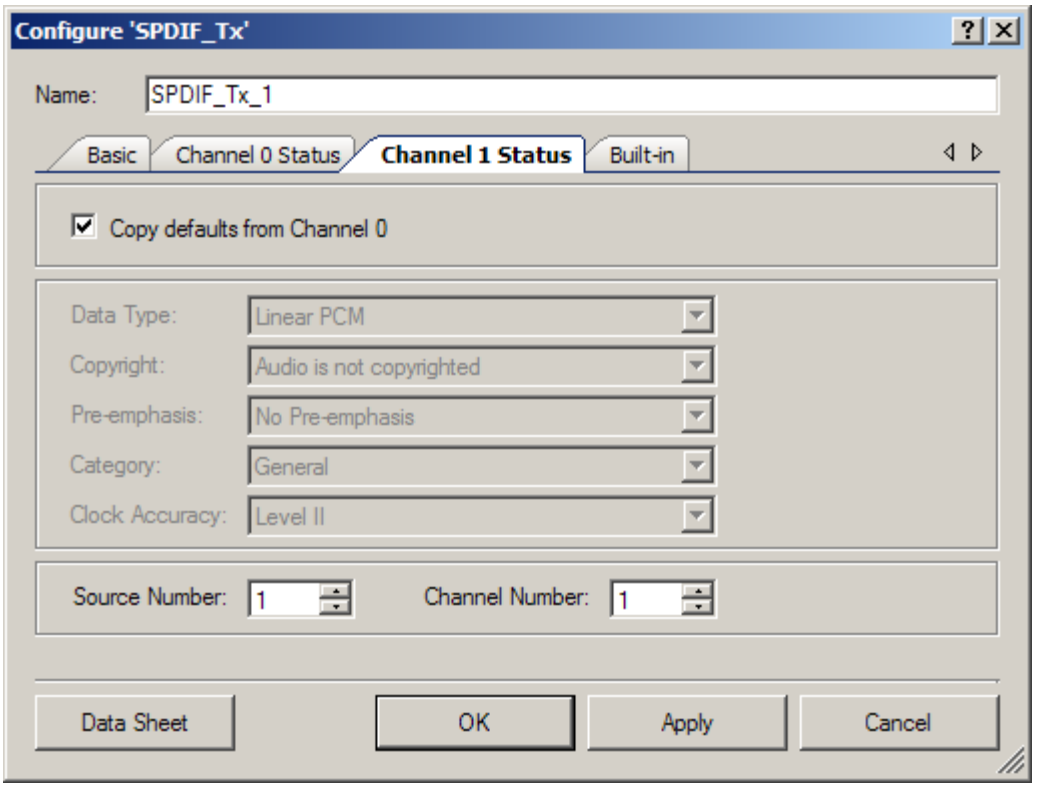
源编号

确定通道状态 0 的源编号。此值可以设置为 0-15 之间。默认设置为 **0**。

通道编号

确定通道状态 0 的通道编号。此值可以设置为 0-15 之间。默认设置为 **0**。

Channel 1 Status（通道 1 状态）选项卡



Copy defaults from Channel 0（从通道 0 复制默认值）

通过此复选框，可以选择通道 1 的通道状态是否与通道 0 相同。如果选中该复选框，则在 **Channel 1 Status**（通道 1 状态）选项卡上禁用所有下拉框。默认情况下，选中该设置。

Channel 1 Status（通道 1 状态）选项卡的剩余参数与 **Channel 0 Status**（通道 0 状态）选项卡相同。

时钟选择

此组件中没有内部时钟。您必须附加时钟源。时钟速率必须是 Spdif 输出所需数据速率的 2 倍。

放置

SPDIF_Tx 组件放置于整个 UDB 阵列中，并且所有放置信息通过 *cyfitter.h* 文件提供给 API。

资源

资源	资源类型						API Memory（API 存储器）（字节）		Pins（引脚）（每个外部 I/O）
	数据路径单元	PLDs	状态单元	Control/Count7 单元	Drqs	Irqs	Flash（闪存）	RAM	
管理的 DMA	4	9	1	2	2	2	1420	106	0
未管理的 DMA	4	9	1	2	0	0	290	2	0

应用程序编程接口

应用程序编程接口 (API) 子程序允许您使用软件配置组件。下表列出了每个函数的接口，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 将给设计中第一个组件命名为 SPDIF_Tx_1” 您可以在符合标识符语法规则的前提下，给实体重新命名 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“SPDIF”。

函数	说明
SPDIF_Start()	启动 S/PDIF 接口。
SPDIF_Stop()	禁用 S/PDIF 接口。
SPDIF_Sleep()	保存配置和禁用 S/PDIF 接口。
SPDIF_Wakeup()	恢复 S/PDIF 接口的配置。
SPDIF_EnableTx()	启用 S/PDIF 位流中的音频数据输出。
SPDIF_DisableTx()	禁用 S/PDIF 位流中的音频输出。
SPDIF_WriteTxByte()	向音频 FIFO 写入一个字节。
SPDIF_WriteCstByte()	向通道状态 FIFO 写入一个字节。
SPDIF_SetInterruptMode()	设置 S/PDIF 中断的中断源。
SPDIF_ReadStatus()	返回 S/PDIF 状态寄存器中的状态。
SPDIF_ClearTxFIFO()	清除音频 FIFO。
SPDIF_ClearCstFIFO()	清除通道状态 FIFO。
SPDIF_SetChannelStatus()	在运行时设置通道状态的值。
SPDIF_SetFrequency()	设置指定频率通道状态的值。
SPDIF_Init()	初始化或恢复 S/PDIF 默认配置。

函数	说明
SPDIF_Enable()	启用 S/PDIF 接口。
SPDIF_SaveConfig()	保存 S/PDIF 接口配置。
SPDIF_RestoreConfig()	恢复 S/PDIF 接口配置。

全局变量

变量	说明
SPDIF_initVar	SPDIF_initVar 表示 S/PDIF 组件初始化是否完成。变量将初始化为 0，并在第一次调用 SPDIF_Start() 时设置为 1。这样，第一次调用 SPDIF_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该组件，则须首先调用 SPDIF_Init() 函数，然后再调用 SPDIF_Start() 或 SPDIF_Enable() 函数。
SPDIF_wrkCstStream0[]、SPDIF_wrkCstStream1[]	通道 0 和通道 1 各自的通道状态阵列。对这些阵列进行有条件编译，只在组件管理通道状态 DMA 时，才提供给该组件。 将通道状态流存储在两个不同缓冲区，这两个缓冲区要足够大以用来存储整个 SPDIF 模块流（2x24 字节）。 注意：设置通道状态的值，建议您使用 SPDIF_SetChannelStatus() 函数及通道状态宏。 例如，要设置通道 0 状态的类别字段，通常使用下列函数调用： SPDIF_SetChannelStatus(SPDIF_CHANNEL_0, SPDIF_CAT_GEN);

void SPDIF_Start(void)

说明： 启动 S/PDIF 接口。如果配置组件以处理通道状态 DMA，则启动通道状态 DMA。根据需要，启用活动模式电源模板位或时钟关断。通过通道状态启动 S/PDIF 输出的生成，而将音频数据全部设置为 0。这样，S/PDIF 接收器便可以锁定到组件时钟。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_Stop(void)

说明： 禁用 S/PDIF 接口。根据需要，禁用活动模式电源模板位或时钟关断。S/PDIF 输出设置为 0。清除音频数据和通道数据 FIFO。SPDIF_Stop() 函数调用 SPDIF_DisableTx()，并停止管理通道状态 DMA。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_Sleep(void)

说明： 这是组件准备睡眠时的首选子程序。SPDIF_Sleep() 子程序保存当前组件状态。然后调用 SPDIF_Stop() 函数和 SPDIF_SaveConfig() 以保存硬件配置。根据需要，禁用活动模式电源模板位或时钟关断。Spdif 输出设置为 0。

调用 SPDIF_Sleep() 函数，然后再调用 CyPmSleep() 或 CyPmHibernate() 函数。有关电源管理函数的更多信息，请参考 PSoC Creator *System Reference Guide*（《系统参考指南》）。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_Wakeup(void)

说明： 恢复 SPDIF 配置和非保留寄存器值。进入睡眠前，该组件停止，不考虑其状态如何。必须明确调用 SPDIF_Start() 函数，以便再次启动组件。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： 如果在调用 SPDIF_Wakeup() 函数前未调用 SPDIF_Sleep() 或 SPDIF_SaveConfig() 函数，可能产生意外行为。

void SPDIF_EnableTx(void)

说明： 启用 S/PDIF 位流中的音频数据输出。传输将在下一个 X 或 Z 帧开始。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_DisableTx(void)

说明： 禁用 S/PDIF 位流中的音频输出。数据传输将在时钟下一个上升沿停止，并将传输常量 0 值。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_WriteTxByte(uint8 wrData, uint8 channelSelect)

说明： 向音频数据 FIFO 写入一个字节。在此调用前，应选中组件状态，以便确认音频数据 FIFO 未满状态。

参数： uint8 wrData：包含要传输的音频数据的字节。

uint8 channelSelect：包含要写入的通道常量的字节。请参见以下状态宏。在交叉模式下，此参数被忽略

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_WriteCstByte(uint8 wrData, uint8 channelSelect)

说明： 向指定通道状态 FIFO 写入一个字节。在此调用前，应选中组件状态，以便确认通道状态 FIFO 未满状态。

参数： uint8 wrData：包含要传输的状态数据的字节。

uint8 channelSelect：包含要写入的通道常量的字节。请参见以下通道状态宏。

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_SetInterruptMode(uint8 interruptSource)

说明： 设置 S/PDIF 中断的中断源。多个源可能均是“或”运算。

参数： 包含所选中断源常量的 uint8 字节。

SPDIF Tx 中断源	值
AUDIO_FIFO_UNDERFLOW	0x01
AUDIO_0_FIFO_NOT_FULL	0x02
AUDIO_1_FIFO_NOT_FULL	0x04
CHST_FIFO_UNDERFLOW	0x08
CHST_0_FIFO_NOT_FULL	0x10
CHST_1_FIFO_NOT_FULL	0x20

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

uint8 SPDIF_ReadStatus(void)

说明： 返回 SPDIF 状态寄存器中的状态。

参数： None（无）

Return Value（返回值）： SPDIF 状态寄存器的 uint8 状态。

SPDIF 状态掩码	值	类型
AUDIO_FIFO_UNDERFLOW	0x01	读取时清除
AUDIO_0_FIFO_NOT_FULL	0x02	透明
AUDIO_1_FIFO_NOT_FULL	0x04	透明
CHST_FIFO_UNDERFLOW	0x08	读取时清除
CHST_0_FIFO_NOT_FULL	0x10	透明
CHST_1_FIFO_NOT_FULL	0x20	透明

Side Effects（副作用）： 清除“读取时清除”类型的 SPDIF 状态寄存器位。

void SPDIF_ClearTxFIFO(void)

说明： 清除音频数据 FIFO。FIFO 中提供的所有数据均将丢失。在独立音频模式下，两个音频 FIFO 将被清除。仅在禁用传输时，才调用此函数。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_ClearCstFIFO(void)

说明： 清除通道状态 FIFO。在任意一个 FIFO 中提供的所有数据均将丢失。仅在组件停止时，才调用此函数。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_SetChannelStatus(uint8 channel, uint8 byte, uint8 mask, uint8 value)

说明： 在运行时设置通道状态的值。仅在该组件管理 DMA 时，此 API 才有效。

参数： uint8 通道：包含用来指定要修改通道的常量的字节。请参见以下状态宏。
uint8 字节：要修改的字节 [0..23]。请参见以下状态宏。
uint8 掩码：字节上的掩码。请参见以下状态宏。
uint8 值：要设置的值。请参见以下状态宏。

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

uint8 SPDIF_SetFrequency(uint8 frequency)

说明： 设置指定频率通道状态的值，并返回值 1。此函数仅在组件停止时起作用。如果在启动组件时调用此函数，则返回 0，并且不将修改任何值。仅在该组件管理 DMA 时，此 API 才有效。

参数： uint8：包含指定频率常量的字节。

Name（名称）	说明
SPDIF_SPS_22KHZ	为 22-kHz 音频设置的时钟频率
SPDIF_SPS_44KHZ	为 44-kHz 音频设置的时钟频率
SPDIF_SPS_88KHZ	为 88-kHz 音频设置的时钟频率
SPDIF_SPS_24KHZ	为 24-kHz 音频设置的时钟频率
SPDIF_SPS_48KHZ	为 48-kHz 音频设置的时钟频率
SPDIF_SPS_96KHZ	为 96-kHz 音频设置的时钟频率
SPDIF_SPS_32KHZ	为 32-kHz 音频设置的时钟频率
SPDIF_SPS_64KHZ	为 64-kHz 音频设置的时钟频率
SPDIF_SPS_192KHZ	为 192-kHz 音频设置的时钟频率
SPDIF_SPS_UNKNOWN	未指定时钟频率

Return Value（返回值）： uint8： 成功时为 1
失败时为 0

Side Effects（副作用）： None（无）

void SPDIF_Init(void)

说明： 初始化或恢复自定义程序提供的默认 S/PDIF 配置，因此在配置组件以用来处理通道状态 DMA 时定义组件的中断源和通道状态。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： 配置组件以处理通道状态 DMA 时，仅恢复中断生成的掩码寄存器和通道状态。它不会从 FIFO 中清除数据，也不会复位组件硬件状态机。

void SPDIF_Enable(void)

说明： 激活硬件并开始执行组件操作。无需调用 SPDIF_Enable()，因为 SPDIF_Start() 子程序会调用此函数，这是开始组件操作的首选方法。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_SaveConfig(void)

说明： 此函数保存组件配置。它将保存非保存寄存器。此函数还将保存当前“配置”对话框中定义的或通过相应 API 修改的组件参数值。通过 SPDIF_Sleep() 函数调用此函数。

参数： None（无）

Return Value（返回值）： None（无）

Side Effects（副作用）： None（无）

void SPDIF_RestoreConfig(void)

说明：	此函数恢复组件配置。这将恢复非保留寄存器。此外，该函数还将组件参数值恢复到调用 SPDIF_Sleep() 函数之前的值。此子程序由 SPDIF_Wakeup() 来调用，从而恢复退出睡眠状态时的组件。
参数：	None（无）
Return Value（返回值）：	None（无）
Side Effects（副作用）：	必须在调用 SPDIF_SaveConfig() 子程序之后调用。否则，该组件配置将被其原始设置所覆盖。

通道状态宏

通道状态宏用来封装指定通道状态设置和掩码和值。这些宏及通道宏用于在运行时通过 SPDIF_SetChannelStatus() API 设置通道状态值。

建议最好在相同时间下设置这两个通道值。

如下例所示：

SPDIF_SetChannelStatus(SPDIF_CHANNEL_0, SPDIF_DATA_TYPE_LINEAR_PCM);

通道名称常量

Name（名称）	说明
SPDIF_CHANNEL_0	通道 0
SPDIF_CHANNEL_1	通道 1

通道状态常量

Name（名称）	字节	掩码	值
SPDIF_DATA_TYPE_LINEAR_PCM	0	0x02	0x00
SPDIF_DATA_TYPE_OTHERDATA	0	0x02	0x02
SPDIF_COPY_HAS_CP_RIGHT	0	0x04	0x00
SPDIF_COPY_NO_CP_RIGHT	0	0x04	0x04
SPDIF_PREEMP_NO_PREEMP	0	0x38	0x00
SPDIF_PREEMP_PREEMP50	0	0x38	0x08
SPDIF_CAT_GEN	1	0xFF	0x00
SPDIF_CAT_D2D	1	0xFF	0x02

Name（名称）	字节	掩码	值
SPDIF_SRC_NUM00	2	0x0F	0x00
SPDIF_SRC_NUM01	2	0x0F	0x01
SPDIF_SRC_NUM02	2	0x0F	0x02
SPDIF_SRC_NUM03	2	0x0F	0x03
SPDIF_SRC_NUM04	2	0x0F	0x04
SPDIF_SRC_NUM05	2	0x0F	0x05
SPDIF_SRC_NUM06	2	0x0F	0x06
SPDIF_SRC_NUM07	2	0x0F	0x07
SPDIF_SRC_NUM08	2	0x0F	0x08
SPDIF_SRC_NUM09	2	0x0F	0x09
SPDIF_SRC_NUM10	2	0x0F	0x0A
SPDIF_SRC_NUM11	2	0x0F	0x0B
SPDIF_SRC_NUM12	2	0x0F	0x0C
SPDIF_SRC_NUM13	2	0x0F	0x0D
SPDIF_SRC_NUM14	2	0x0F	0x0E
SPDIF_SRC_NUM15	2	0x0F	0x0F
SPDIF_CH_NUM00	2	0xF0	0x00
SPDIF_CH_NUM01	2	0xF0	0x10
SPDIF_CH_NUM02	2	0xF0	0x20
SPDIF_CH_NUM03	2	0xF0	0x30
SPDIF_CH_NUM04	2	0xF0	0x40
SPDIF_CH_NUM05	2	0xF0	0x50
SPDIF_CH_NUM06	2	0xF0	0x60
SPDIF_CH_NUM07	2	0xF0	0x70
SPDIF_CH_NUM08	2	0xF0	0x80
SPDIF_CH_NUM09	2	0xF0	0x90
SPDIF_CH_NUM10	2	0xF0	0xA0
SPDIF_CH_NUM11	2	0xF0	0xB0
SPDIF_CH_NUM12	2	0xF0	0xC0
SPDIF_CH_NUM13	2	0xF0	0xD0
SPDIF_CH_NUM14	2	0xF0	0xE0

Name（名称）	字节	掩码	值
SPDIF_CH_NUM15	2	0xF0	0xF0
SPDIF_CLKLVL_1	3	0x30	0x10
SPDIF_CLKLVL_2	3	0x30	0x00
SPDIF_CLKLVL_3	3	0x30	0x20
SPDIF_STDLEN	4	0x0F	0x00
SPDIF_24BLEN	4	0x0F	0x0B

固件源代码示例

PSoC Creator 在“查找示例项目”对话框中提供了大量包括原理图和代码示例的示例项目。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。要获取通用的示例，请打开 Start Page（开始页）或 **File**（文件）菜单中的对话框。根据需要，使用对话框中的 **Filter Options**（滤波器选项）可缩小可选项目的列表。

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

功能描述

此组件用来格式化输入的音频数据和元数据以创建 S/PDIF 位流。该组件接收来自 DMA 的音频数据及通道状态信息。在大部分时间内，通道状态 DMA 均由组件来管理；然而，您可以选择单独指定此数据，以便更好地控制系统。

数据流格式

音频和通道状态数据是独立的字节流。字节流从最低有效字节和有效位开始被压缩。每个采样所使用的字节数是容纳采样的最少字节数。任何未使用的位从最左侧的位开始使用 0 填塞。

音频数据流即可以是一个字节流，也可以是两个字节流。在一个字节流的情况下，左通道和右通道与采样互相交错，左通道后面紧接右通道。在双流情况下，左右通道字节流均使用各自的 FIFO。该状态字节流始终为双字节流。

DMA

S/PDIF 接口是要求未中断数据流的连接接口。对大多数应用而言，这需要使用 DMA 传输，以便防止音频数据或通道状态 FIFO 下溢。通常情况下，在该组件中使用两个通道状态阵列完整地执行通道状态 DMA；然而，您可以使用自定义 DMA 或 CPU 来提供数据，从而提高灵活性。

S/PDIF 可以驱动高达 4 个 DMA 组件，这取决于组件配置。DMA 向导用于配置如下 DMA 操作：

在 DMA 向导中 DMA 源/目录的名称	方向	DMA 请求信号	DMA 请求类型	说明
SPDIF_TX_FIFO_0_PTR	目标	tx_dma0	电平	传输通道 0 的 FIFO，或交错音频数据
SPDIF_TX_FIFO_1_PTR	目标	tx_dma1	电平	传输通道 1 音频数据的 FIFO
SPDIF_CST_FIFO_0_PTR	目标	cst_dma0	电平	传输通道 0 通道状态数据的 FIFO
SPDIF_CST_FIFO_1_PTR	目标	cst_dma1	电平	传输通道 1 通道状态数据的 FIFO

在所有情况下，DMA 请求信号上的高电平信号表示可以传输一个附加字节。

错误处理

如果音频为空，并出现后续读取事件（传输下溢）或通道状态 FIFO 为空，而发生后续读取事件（状态下溢），则该组件可能发生两个错误条件。

如果出现传输下溢，则组件强制传输音频数据常量，并继续修正所有帧和状态数据的生成。再次启动传输前，必须禁用传输，并不应清除 FIFO，必须缓冲传输数据，然后重新启用传输。使用组件状态位 AUDIO_FIFO_UNDERFLOW 通过 CPU 来监控此下溢条件。此外，还可以针对此错误条件配置中断。

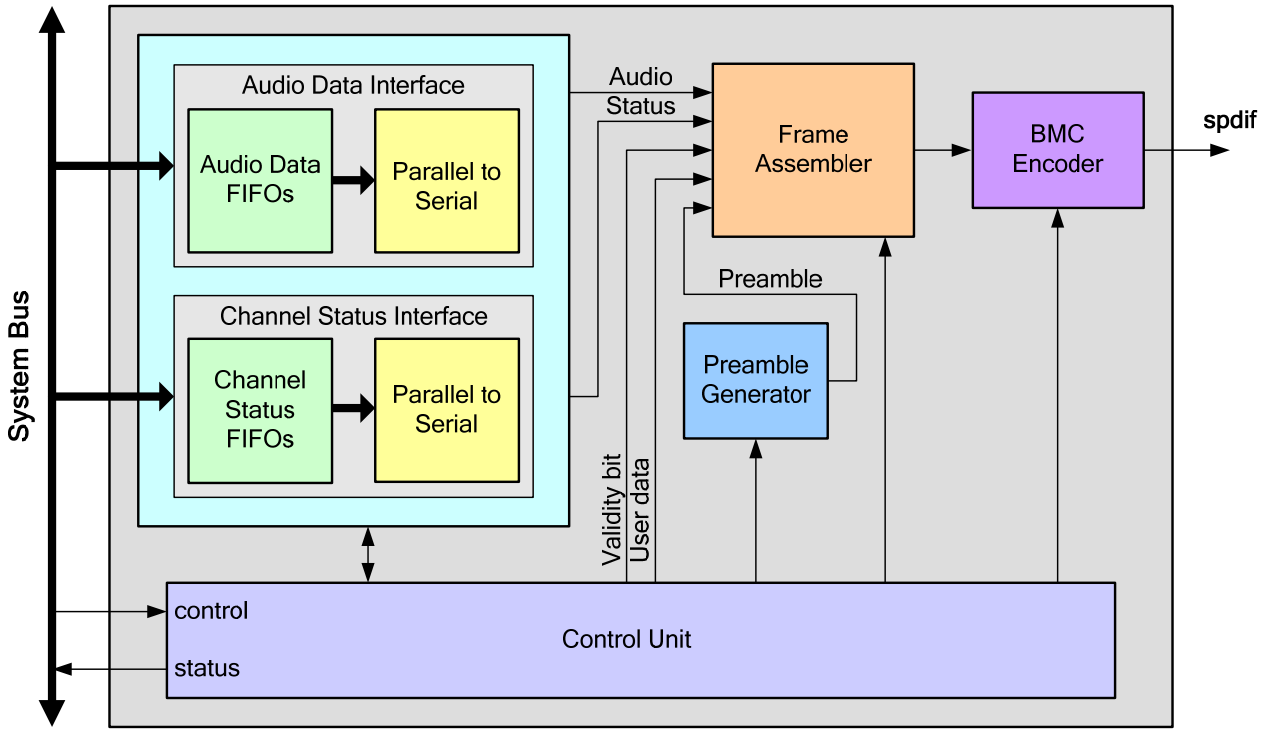
启动组件时，如果发生状态下溢，则该组件将通过正确生成 X、Y、Z 帧和正确的奇偶校验位来为通道状态发送所有 0 值。音频数据是连续的，而不受影响。要正确传输通道状态，必须停止组件，然后再启组件。使用状态位 CHST_FIFO_UNDERFLOW 通过 CPU 来监控此下溢条件。此外，还可以针对此错误条件配置中断。如果组件未管理 DMA，则必须在重启组件前缓冲状态数据。

启用系统

音频数据传输具有专用的启用系统。启动组件而非启用组件时，在通道状态下生成 S/PDIF 输出，而将音频数据全部设置为 0。这样，S/PDIF 接收器便可以锁定到组件时钟上。在 X 或 Z 帧发生时，切换为启用状态。

框图和配置

SPDIF_Tx 可以实现为一组 UDB 配置。在以下框图中显示该实现。



通过系统总线接口接收输入的音频数据，并通过 CPU 或 DMA 提供该音频数据。该数据是从 LSB 开始的字节宽，存储在音频缓冲区（一个或两个 FIFO，这取决于组件配置）中。

通道状态流具有自己的专用接口。同样，音频数据也有两个通道状态 FIFO。此外，通道状态为字节宽数据，从最低有效字节开始。从这些 FIFO 中，每 8 个采样使用一个字节。音频数据和状态数据均从并行格式转变成串行格式。

在 S/PDIF 标准中未定义用户数据，某些接收器可能忽略该用户数据，因此这些数据将作为常量 0 发送。

当有效位为低位时，表示音频采样适合于模拟转换。此位作为常量 0 发送。

在前导码发生器模块中生成前导码模式，然后以串行格式传输。

这是构成 SPDIF 子帧结构的所有必要数据，但奇偶校验位除外。将所有输入汇集到子帧结构时，在帧汇编程序模块计算奇偶校验位。

帧汇编程序模块输出转到 BMC 编码器，在此将数据编译到 Spdif 输出中。

控制单元模块从系统总线接口获得控制数据，并将组件操作状态返回到总线。传输过程中，它控制其他所有模块。

寄存器

SPDIF_Tx_CONTROL_REG

位	7	6	5	4	3	2	1	0
值	保留						使能	txenable

- 启用：启用/禁用 SPDIF_Tx 组件。未启用时，组件处于复位状态。
- txenable：启用/禁用 S/PDIF 位流中的音频数据输出。

SPDIF_Tx_STATUS_REG

位	7	6	5	4	3	2	1	0
值	保留		chst1_fifo_not_full	chst0_fifo_not_full	chst_fifo_underflow	audio1_fifo_not_full	audio0_fifo_not_full	audio_fifo_underflow

- chst1_fifo_not_full：如果通道状态 FIFO 1 未满
- chst1_fifo_not_full：如果设置通道状态时，FIFO 0 未满
- chst_fifo_underflow：如果设置通道状态时，发生 FIFO 下溢事件
- audio1_fifo_not_full：如果设置音频数据时，FIFO 1 未满
- audio0_fifo_not_full：如果设置音频数据时，FIFO 0 未满
- audio_fifo_underflow：如果设置音频数据时，发生 FIFO 下溢事件

通过 SPDIF_Tx_ReadStatus() API 函数读取寄存器值。

注意：在粘连模式下配置状态寄存器的位 3 和位 0，其类型为“读取时清除”。在此模式下，在状态寄存器时钟的每个周期对输入状态进行采样，当输入达到高电平时，设置寄存器位，并保持设置，而不考虑输入的后续状态。在后续读取时，通过 CPU 清除寄存器位。

特性

下面的值表示了预计性能，它们基于初始特性数据。

时序特性“额定路由的最大值”

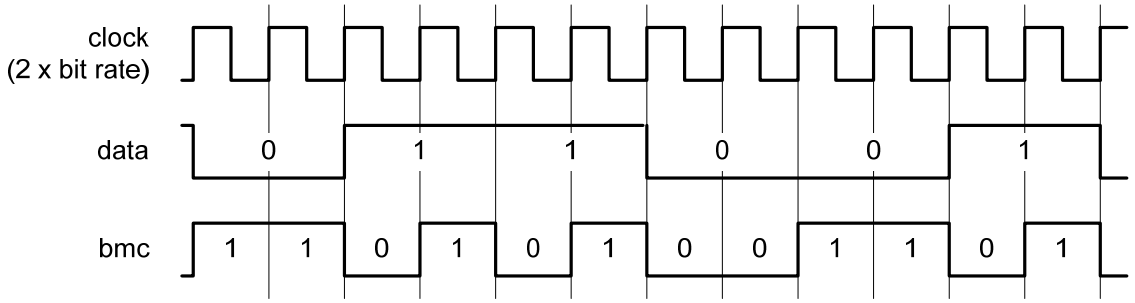
参数	说明	最小值	典型值	最大值	单位
f_S	采样频率	–	–	192	kHz
f_{CLOCK}	组件时钟频率	–	128 × f_S	35	MHz

时序特性“所有路由的最大值”

参数	说明	最小值	典型值	最大值[1]	单位
f_S	采样频率	–	–	96	kHz
f_{CLOCK}	组件时钟频率	–	128 × f_S	17	MHz

S/PDIF 通道编码

S/PDIF 是单线串行接口。位时钟嵌入 S/PDIF 数据流中。使用双相符号编码 (BMC) 对数字信号进行编码，这是一种相位调制。时钟频率是位速率的两倍。原始数据的每一个位表示两个逻辑状态，这两个状态一起形成一个单元。在开始位的逻辑电平始终转换成上一个结束位的电平。要使用此格式传输‘1’，在数据位边界中间有一次跃变。如果在此中间未发生跃变，则该数据被视为‘0’。



S/PDIF 协议分级

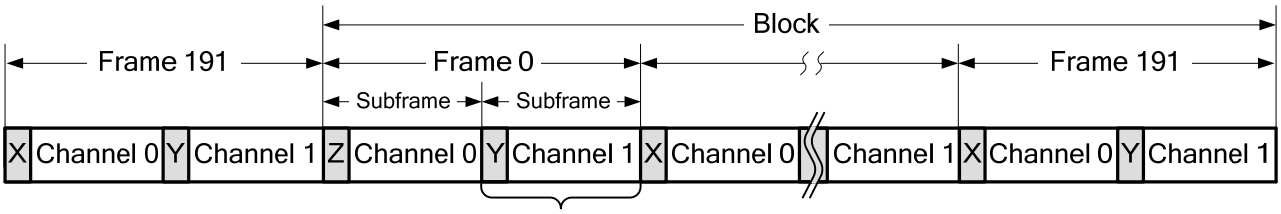
下图说明了 SPDIF 信号格式。音频数据以连续模块的格式进行传输。模块包含 192 帧。每一个帧由两个子帧组成，这两个子帧是组织数字音频数据的基本组件。

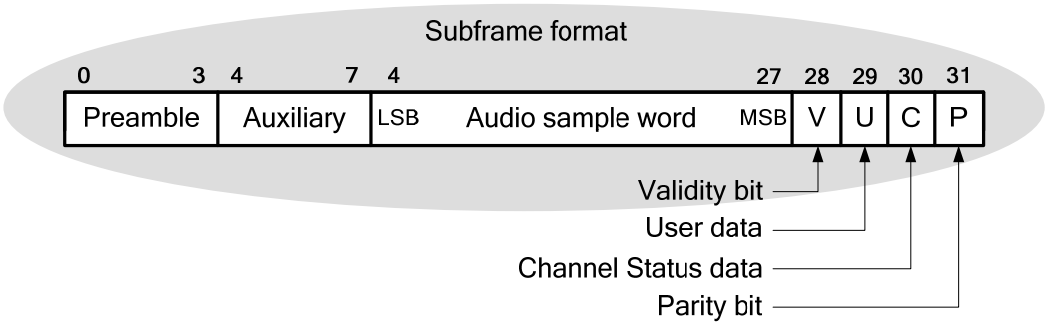
子帧包含：

- 前导码模式
- 高达 24 位宽的单一音频采样
- 表示采样是否有效的有效位
- 包含用户数据的位
- 包含通道状态的位
- 该子帧的偶校验位

[1]“所有路由”的最大值的计算方法是：<额定值>/2，然后取整到最近的整数。此值提供了一个基础，因此用户不必担心在以该频率或低于组件频率运行该组件时遇到时序问题。

前导码有 3 种类型：X、Y 和 Z。前导码 Z 表示模块和子帧通道 0 的起始位。前导码 X 表示通道 0 子帧的起始位，而未在模块起始位。前导码 Y 始终表示通道 1 子帧的起始位。





如何将 **STA** 结果用于特性数据

f_{CLOCK} **最大组件时钟频率显示在命名外部时钟的时钟汇总中的时序结果中。内部时钟限制示例如下所示：**

+Clock Summary

Clock	Actual Freq	Max Freq	Violation
BUS_CLK	24.000 MHz	52.845 MHz	
CLK	6.000 MHz	38.962 MHz	

组件更改

版本 1.0 是 SPDIF_Tx 组件的首次发行版本。

© 赛普拉斯半导体公司，2012。此处所包含的信息可能会随时更改，恕不另行通知。除赛普拉斯产品的内嵌电路之外，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。除上述指定的用途之外，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（但不仅限于）针对特定用途的适销性和适用性的暗示保证。赛普拉斯保留在不做出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。

产品使用可能受适用的赛普拉斯软件许可协议限制。