



Vincent Rodriguez, Errol Leon, Kevin Allen

## 摘要

该应用报告提供了有关如何将 C2000 器件与 AFE031 连接，以及如何使用移频键控通过有线耦合接口发送和接收数据的信息。文中讨论了 FSK 信息传输演示，其中使用的音调频率范围为 130-145kHz。

软件的目标处理器是 TMS320F28379D，但该软件可以移植到其他器件中。可以在最新的 [C2000Ware](#) 版本中找到本文档中讨论的工程配套资料和示例代码，其目录为：

C:\ti\c2000\C2000Ware\_x\_xx\_xx\_xx\device\_support\f28379d\examples\cpu1

提供的示例工程包括：

- boostxl\_afe031\_f28379d\_pwmmode
- boostxl\_afe031\_f28379d\_dacmode
- boostxl\_afe031\_f28379d\_rx

可通过以下 URL 找到 BOOSTXL-AFE031 的参考设计：<http://www.ti.com.cn/tool/cn/TIDA-060001>。

## 内容

<b>1 FSK 概述</b> .....	3
<b>2 硬件预览</b> .....	5
2.1 方框图.....	6
2.2 硬件设置.....	7
<b>3 连接 AFE03x</b> .....	10
3.1 配置 AFE031.....	10
<b>4 发送路径</b> .....	12
4.1 FSK 示例规格.....	12
4.2 PWM 模式.....	13
4.3 DAC 模式.....	17
4.4 将 TX 移植到 LAUNCHXL-F280049C.....	20
<b>5 接收路径</b> .....	24
5.1 接收路径概述.....	24
5.2 接收器软件实现.....	26
5.3 调优和校准.....	30
5.4 将 RX 移植到 LAUNCHXL-F280049C.....	32
<b>6 连接电源线</b> .....	33
6.1 线路耦合.....	33
6.2 耦合到交流线路.....	33
6.3 耦合到直流线路.....	35
6.4 保护电路.....	36
6.5 确定 PA 电源要求.....	41
<b>7 总结</b> .....	42
<b>8 参考文献</b> .....	43
<b>9 原理图</b> .....	44
9.1 原理图 ( PWM 模式 ) .....	44
9.2 原理图 ( DAC 模式 ) .....	48
<b>10 修订历史记录</b> .....	52

## 插图清单

图 1-1. 时域中的二进制 FSK.....	3
图 1-2. 发送器示例.....	4
图 1-3. 接收器示例.....	4
图 2-1. TMDS28379D LaunchPad.....	5
图 2-2. BoosterPack 测试板.....	5
图 2-3. C2000 和 AFE031 方框图.....	6
图 2-4. BOOSTXL-AFE031 电源引脚.....	7
图 2-5. BOOSTXL-AFE031 接线盒接头.....	7
图 2-6. F28379D 跳线配置.....	8
图 2-7. F28379D JP6 位置.....	8
图 2-8. TX-RX 解决方案系统.....	9
图 4-1. PWM 累加.....	13
图 4-2. PWM 发送路径.....	13
图 4-3. TX 滤波器增益与频率之间的关系.....	14
图 4-4. PWM 模式标记频率, 131.25kHz.....	15
图 4-5. PWM 模式空间频率, 143.75kHz.....	15
图 4-6. 完整的数据包传输波形.....	15
图 4-7. DAC 发送路径.....	17
图 4-8. DAC 正弦波斜坡.....	17
图 4-9. DAC 模式标记频谱分析器.....	19
图 4-10. DAC 模式空间频谱分析器.....	19
图 4-11. 完整的数据包传输波形.....	19
图 4-12. 60Hz 正弦波.....	20
图 4-13. 100Hz 正弦波.....	20
图 4-14. 60Hz + 100Hz 正弦波.....	20
图 4-15. 组合信号的 FFT.....	20
图 5-1. 与 C2000 ADC 连接的 AFE031 接收路径.....	24
图 5-2. AFE031 接收路径的外部滤波.....	25
图 5-3. RX 滤波器增益与频率之间的关系.....	25
图 5-4. 接收的 FSK 信号.....	26
图 5-5. 包含接收到的数据位的数据包.....	29
图 5-6. 打包的数据.....	29
图 6-1. 交流电源的简化线路耦合电路.....	33
图 6-2. 直流线路的耦合电路.....	35
图 6-3. 多个 PLC 模块耦合到直流线路的示例.....	35
图 6-4. 建议的瞬态保护.....	36
图 6-5. 金属氧化物压敏电阻 (MOV).....	36
图 6-6. 典型 MOV I/V 特性.....	37
图 6-7. 满足 MOV 和 TVS 性能规格的典型 8/20 $\mu$ s 脉冲.....	38
图 6-8. 双向 TVS 二极管.....	38
图 6-9. 典型的双向 TVS I/V 特性.....	39
图 6-10. TVS 上的典型信号.....	40
图 6-11. 导流二极管.....	40
图 6-12. 典型的线路耦合电路.....	41
图 9-1. 原理图 (PWM 模式) - 第 1 页.....	44
图 9-2. 原理图 (PWM 模式) - 第 2 页.....	45
图 9-3. 原理图 (PWM 模式) - 第 3 页.....	46
图 9-4. 原理图 (PWM 模式) - 第 4 页.....	47
图 9-5. 原理图 (DAC 模式) - 第 1 页.....	48
图 9-6. 原理图 (DAC 模式) - 第 2 页.....	49
图 9-7. 原理图 (DAC 模式) - 第 3 页.....	50
图 9-8. 原理图 (DAC 模式) - 第 4 页.....	51

## 表格清单

表 4-1. FSK 规格.....	12
表 4-2. 用于增加 PWM 应用中滤波器响应的外部 R 和 C 值.....	14

表 4-3. PWM 和 HRPWM 的分辨率.....	16
表 5-1. AFE031 RX PGA 增益设置.....	24
表 5-2. 建议的 Rx 滤波器外部电容器值.....	25
表 5-3. 四阶无源带通滤波器的建议元件值 (0dB 通带衰减) .....	25
表 5-4. 软件频率参数.....	27
表 5-5. 利用的 C2000 资源.....	30
表 5-6. 库函数 CPU 周期数.....	30
表 5-7. 软件 ISR/函数使用情况.....	30

**商标**

C2000™ and Code Composer Studio™ are trademarks of Texas Instruments.  
 所有商标均为其各自所有者的财产。

**1 FSK 概述**

移频键控 (FSK) 是一种利用频率的离散变化来发送和接收数字数据的调制方案。调制方案中有一个被称为二进制移频键控 (BFSK) 的最简单的调制方式，也在本演示中使用。

在该方案中，系统在两个离散频率之间进行切换：标记频率 (“1”) 和空间频率 (“0”)。这两种频率与传输数据的位值直接相关。

图 1-1 在时域中显示了对应的数据。

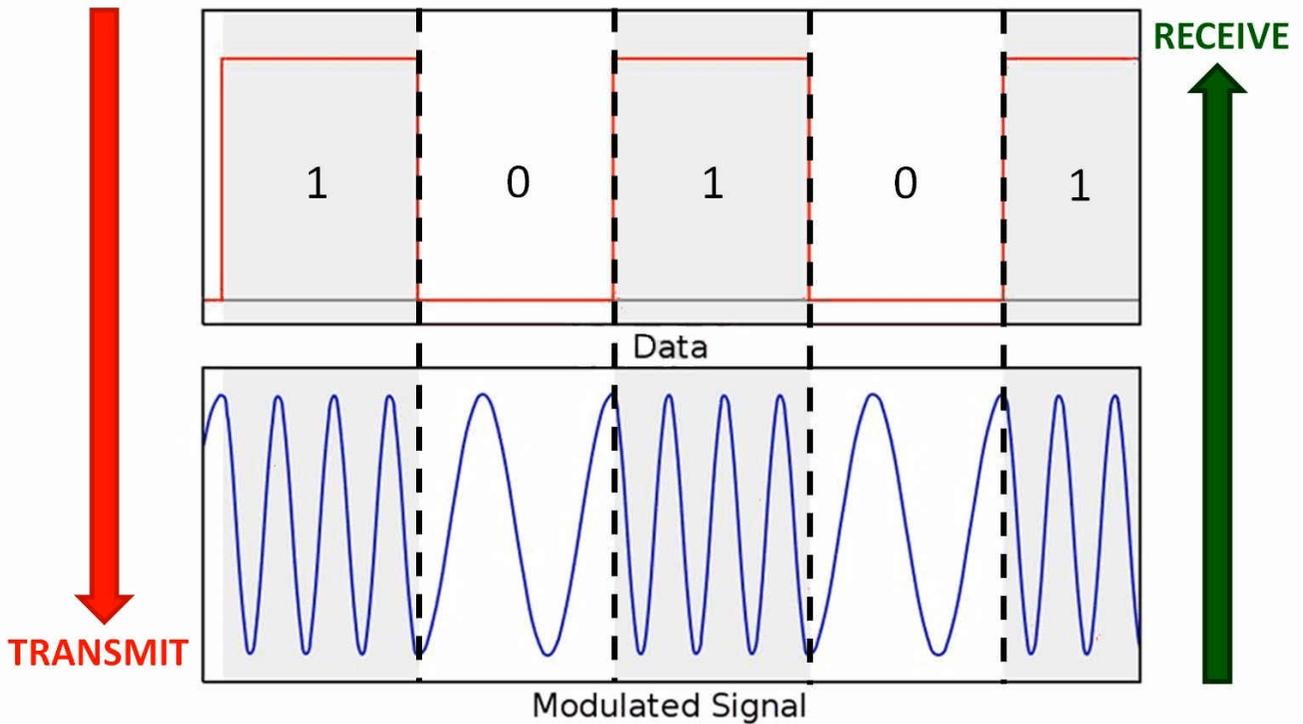


图 1-1. 时域中的二进制 FSK

图 1-2 显示了一个简化的 FSK 发送器示例，其中该模块包含两个振荡器（具有一个内部时钟）和一个用于控制开关位置的输入二进制序列。

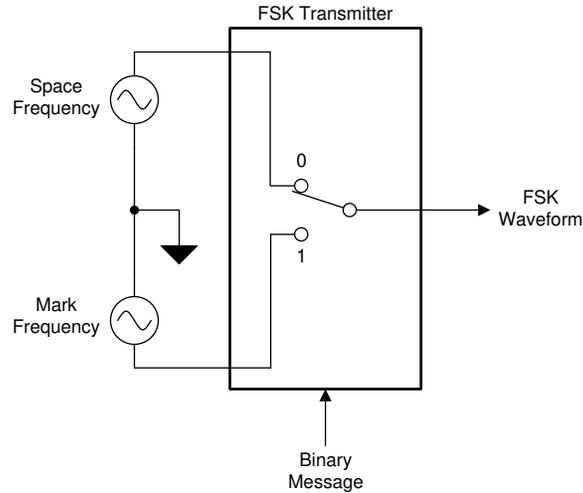


图 1-2. 发送器示例

两个振荡器产生一个较高（空间）频率信号和一个较低（标记）频率信号，与一个内部时钟一同连接到一个开关。两个振荡器在内部共用一个时钟，以避免在消息传输期间输出波形的相位不连续。然后根据二进制输入，通过二进制输入序列来选择频率。在本例中，二进制“0”对应于空间频率的输出，二进制“1”对应于标记频率的输出。

图 1-3 显示了一个简化的 FSK 接收器示例，该接收器用于将接收到的信号转换回所需的数字信息。

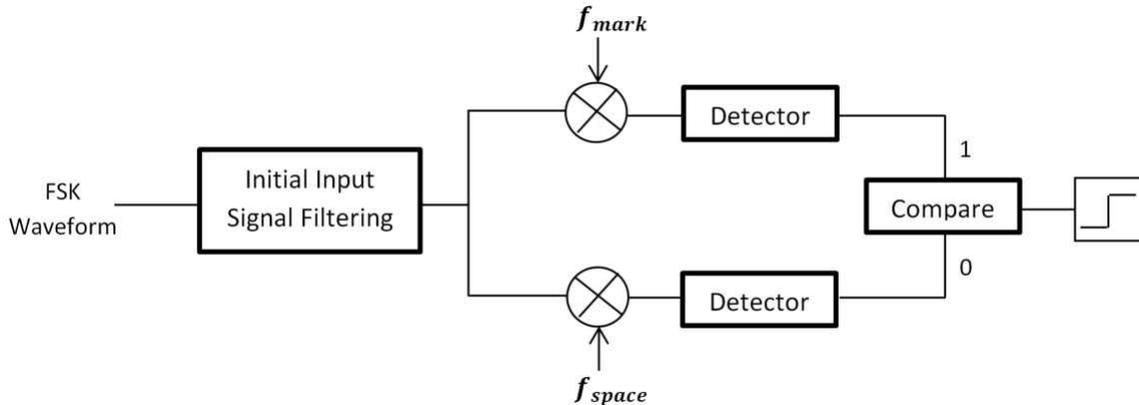


图 1-3. 接收器示例

FSK 波形经过初始滤波，然后与所需的标记频率 ( $f_{mark}$ ) 和空间频率 ( $f_{space}$ ) 信号进行混合。使用检测器算法对输出进行处理，然后对结果进行比较，以辨别接收到的信号是与标记频率相关（二进制“1”）还是与空间频率相关（二进制“0”）。其他功能还包括，根据接收到的标记信号或空间信号的持续时间来辨别接收到的位，并处理连续位之间的边界。

以上是关于 FSK 工作原理的简单概述，以下各节讨论了如何在 C2000 器件上实现 FSK。

## 2 硬件预览

为 FSK 发送器创建的系统包含以下电路板：



图 2-1. TMS28379D LaunchPad

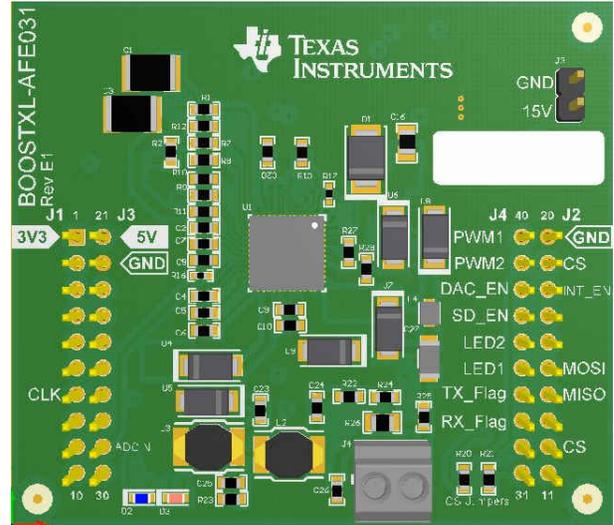


图 2-2. BoosterPack 测试板

C2000Ware 中提供了 BoosterPack 测试板设计文件。节 9 也包含相关的原理图。

## 2.1 方框图

图 2-3 显示了系统的方框图。在该示例中，并未使用所有连接，但 BoosterPack 具有所有连接，以为将来的开发做准备。

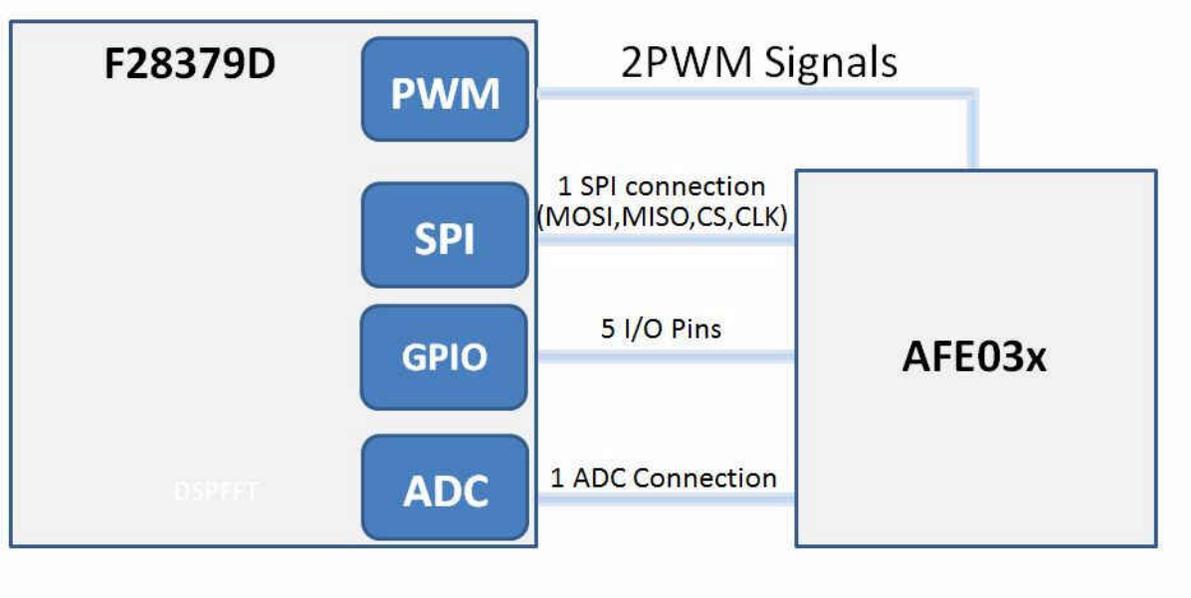


图 2-3. C2000 和 AFE031 方框图

AFE 具有多个内部寄存器，用于配置 AFE 芯片的内部元件，包括滤波器选择、增益选择和模式选择。可以使用 SPI 外设来访问这些寄存器。

AFE 还具有各种 GPIO，允许 MCU 将 AFE 设置为特定的模式并接收中断，从而在 AFE 上实现关键事件。ADC 连接允许 MCU 接收输入信号或对其进行采样。PWM 信号提供了一种为 AFE 创建输出的方法。目前 AFE031 支持两种数据发送模式，即 PWM 模式和 DAC 模式。节 3 提供了这两种模式的说明和实现方式。

## 2.2 硬件设置

需要执行一些操作才能正确调试硬件。通过 15V 跳线为 BOOSTXL-AFE031 测试板供电。BoosterPack 带有一个稳压器，用于为连接的 LaunchPad 供电。图 2-4 突出显示了 15V 接头。

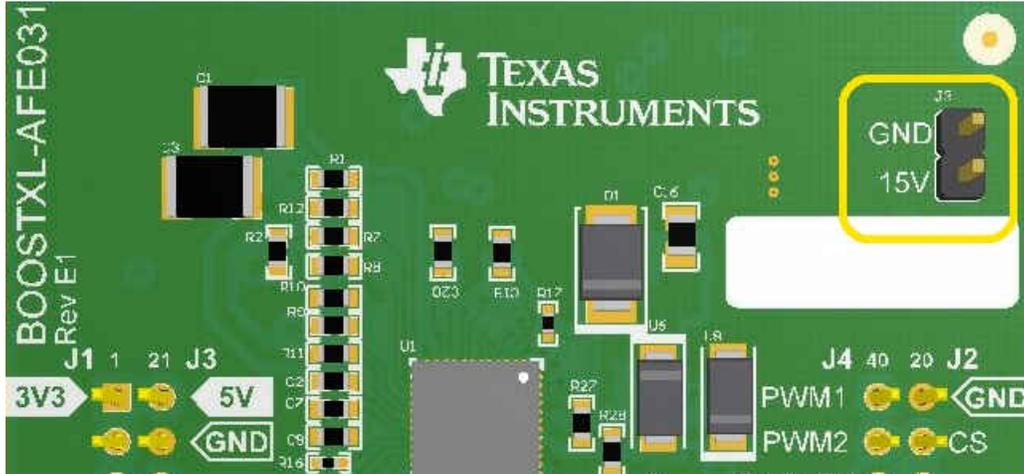


图 2-4. BOOSTXL-AFE031 电源引脚

连接到图 2-5 所示接线盒右侧输入的导线将是发送的输出 FSK 信号或接收的输入 FSK 信号所在的线路，具体取决于系统被设置为发送器还是接收器。连接到接线盒左侧输入的导线是接地线。然后可以将这些线路连接到一些耦合电路或直接连接到另一个 BOOSTXL-AFE031 接线盒，以进行对照测试 ( TX/RX 连接到 TX/RX ， GND 连接到 GND ) 。

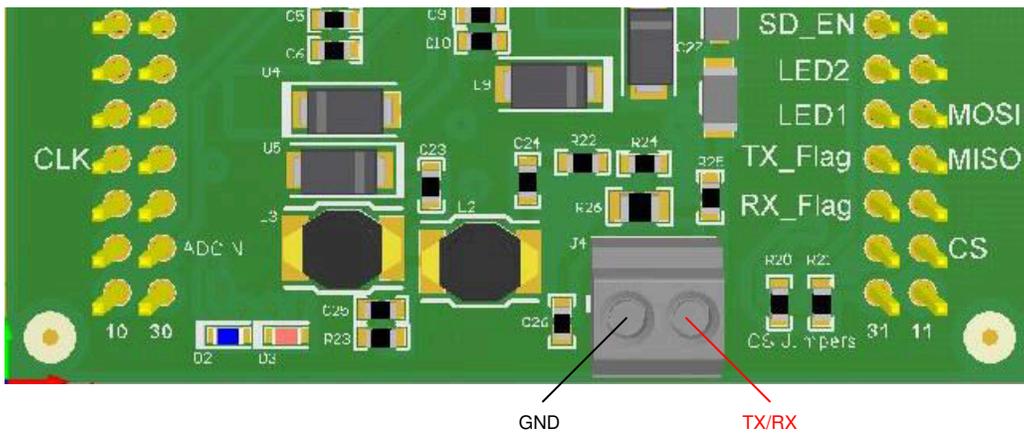


图 2-5. BOOSTXL-AFE031 接线盒接头

BoosterPack 将为 LaunchPad 供电，因此需要隔离 PC USB 接口。需要移除 LaunchPad 上的接头跳线 JP1、JP2 和 JP3 以实现电气隔离。图 2-6 突出显示了接头位置。



图 2-6. F28379D 跳线配置

在将系统用作 FSK 接收器时，LaunchPad 需要使用 5V 的电压为 C2000 ADC 供电。在 LaunchPad 与 PC USB 接口相隔离的情况下，需要通过提供的 3.3V 进行升压来生成 5V 电源轨。需要在 LaunchPad 上添加接头 JP6 以启用升压稳压器。图 2-7 突出显示了接头位置。

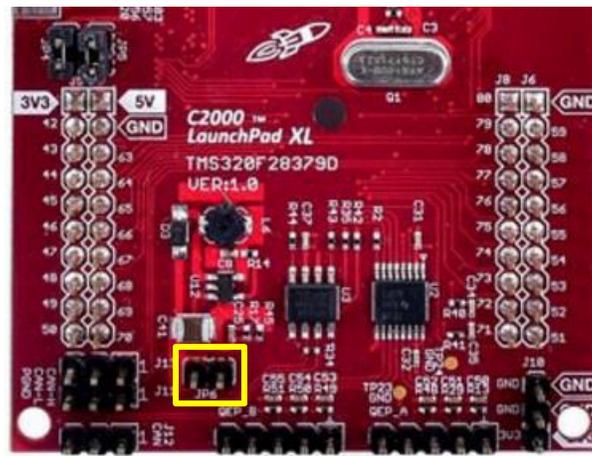


图 2-7. F28379D JP6 位置

在将系统用作接收器时，确保 F28379D LaunchPad 为版本 2.0 或更高版本。对于早期版本（版本 1.1 和 1.2），连接到 BOOSTXL-AFE031 的 ADCIN 引脚存在问题。更多信息，请参阅 [LAUNCHXL-F28379D 概述](#) 的修订部分。

在将发送器解决方案与接收器解决方案结合使用时，完整的系统连接应与图 2-8 所示的连接类似。

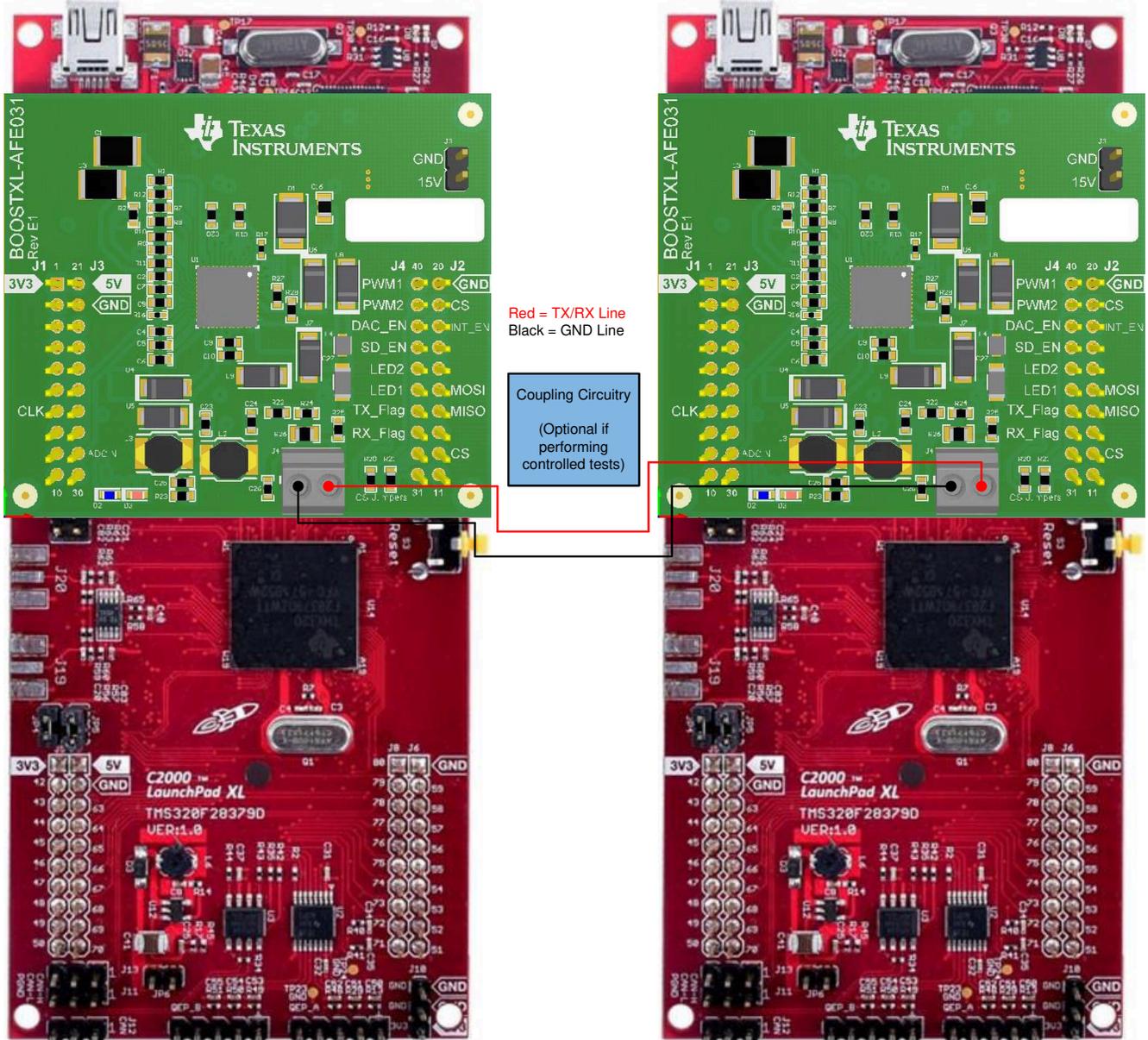


图 2-8. TX-RX 解决方案系统

### 3 连接 AFE03x

将 AFE03x 作为发送器进行连接时，首先确定 AFE03x 采用哪种工作模式 ( DAC 模式或 PWM 模式 )。

在 DAC 模式下，C2000 发送 SPI 数据以更新 AFE031 中的内部 DAC；对输出的 DAC 值进行滤波和放大。在该模式下，会循环遍历一个正弦表，通过 SPI 将每个点发送到 AFE031。

在 PWM 模式下，SPI 连接仅用于配置 AFE 的寄存器。会生成两个 PWM 信号，并以 FSK 传输所需的频率提供给 AFE031。AFE 会将这些信号加在一起，产生比单个 PWM 信号的谐波更少的波形。

请注意，这两种模式需要不同的硬件配置，因此请务必选择其中一种。如果使用 BOOSTXL-AFE031 测试板作为参考，则该硬件仅通过添加或移除一些元件即可使用两种模式。

在将 AFE03x 作为接收器进行连接时，SPI 连接仅用于设置 AFE 的寄存器，这与 PWM 发送模式非常类似。AFE 的接收路径已启用，从而允许输入信号到达 C2000 的 ADC 输入端。在该路径上放置了一些滤波器和两个可编程增益放大器 (PGA)，用于在输入信号进入 C2000 ADC 之前对其进行升压或降压。

#### 3.1 配置 AFE031

配置 AFE 器件需要执行两个主要步骤。首先，确保正确配置 GPIO。在 AFE 器件上，设置了两个主要 GPIO：DAC Mode Select ( DAC 模式选择 ) 和 System Shutdown ( 系统关闭 )。这两个引脚都被拉至低电平。也可将连接的所有其他 GPIO 拉至低电平。

下一步是通过 SPI 配置器件。在软件示例中，`HAL_afe031Init()` 函数配置 AFE031，该函数在 `AFE03x_Config.c` 中进行定义。

以下步骤显示了为正确配置器件，须遵循的函数顺序：

1. 配置 GPIO。
  - a. 设置 AFE 器件上的 SD 和 DAC 引脚。需要将这两个引脚拉至低电平。
  - b. 软件示例函数：`HAL_afe031_cfgGpio()`;
2. 配置 SPI。
  - a. 针对 16 位字符配置 SPI 模块。更多有关 SPI 要求的信息，请参阅 [AFE031 电力线通信模拟前端数据表](#)。
  - b. 软件示例函数：`HAL_spi_cfg()`;
3. 在 AFE 器件上执行软复位。
  - a. 将 0x14 写入复位寄存器。
  - b. 软件示例函数：`HAL_afe031_softReset()`;
4. 启用偏置。
  - a. 将 0x03 写入 enable2 寄存器。
  - b. 软件示例函数：`HAL_afe031_biasEnable()`;
5. 选择频带。
  - a. 将 1 或 0 写入 Control1 寄存器中的 CA\_CBCD 位。如果写入 1，则 TX 和 RX 滤波器的频率响应将配置为 CENELEC B、C、D。如果写入 0，则将配置 CENELEC A。
  - b. 软件示例函数：`HAL_afe031_bandSelect(1)`;
6. 清除所有中断。
  - a. 将 0x00 写入 control2 寄存器。
  - b. 软件示例函数：`HAL_afe031_clrAllInt()`;
7. 配置所有中断。
  - a. 可以通过对 control2 寄存器进行写入来配置中断。软件示例函数目前仅启用指示热过载的 T\_flag。
  - b. 软件示例函数：`HAL_afe031_cfgInt()`;
8. 启用 zc。
  - a. 将 1 写入 enable2 寄存器中的 ZC 位。
  - b. 软件示例函数：`HAL_afe031_zcEnable()`;
9. 写入 TX 增益 - 如果要发送
  - a. 将 0、1、2 或 3 写入增益选择寄存器中的 TXG 位。增益如下所示：
    - i. 0 = 0.25V/V

- ii. 1 = 0.5V/V
  - iii. 2 = 0.707V/V
  - iv. 3 = 1V/V
  - b. 软件示例函数：`HAL_afe031_writeTxGain(UINT16 gain)`;
    - i. 请注意，软件示例函数使用数组来写入 0-3。因此，在使用该函数时，输入参数对应于所需的 `HAL_afe031_txGainLut` 数组元素的索引。例如，将 0 传递给函数对应于 0.25V/V 的增益。
10. 写入 RX 增益 - 如果要接收。
- a. 将 0-15 写入增益选择寄存器中的 RXG 位。增益范围为 0.25V/V (将 RXG 设置为 0x0 时) 至 128V/V (将 RXG 设置为 0xF 时)。
    - i. 如需了解具体的增益，请参阅表 5-1。
  - b. 软件示例函数：`HAL_afe031_writeRxGain(UINT16 gain)`;
    - i. 请注意，软件示例函数使用数组来写入 0-15。因此，在使用该函数时，输入参数对应于所需的 `HAL_afe031_rxGainLut` 数组元素的索引。包含的数组仅具有用于部分 RX 增益配置的值。

完成以上步骤之后，根据该器件是用作发送器还是接收器，开始执行所需的最后几项配置。

## 4 发送路径

现在，AFE 已正确初始化，可使系统根据所选传输方法进行传输：PWM 模式和 DAC 模式。

### 4.1 FSK 示例规格

表 4-1 介绍了在软件示例中生成的 FSK 信号的相关参数。

最终目标是发送完整的数据包，等待规定的等待周期，然后重复。

**表 4-1. FSK 规格**

符号	TX 规格	最小值	标称值	最大值	单位	注释
W1	逻辑 1 代码字	{-1,-1,-1,+1,+1,+1,-1,+1,+1,-1,+1}}				+1 = 标记， -1 = 空间
W0	逻辑 0 代码字	{+1,+1,+1,-1,-1,-1,+1,-1,-1,+1,-1}}				+1 = 标记， -1 = 空间
packet_1	完整的数据包	ABC = [W1,W1,W1]				
packet_0	完整的数据包	ABC = [W0,W0,W0]				
Fm	标记频率	131.236875	131.25	131.263125	kHz	
Fs	空间频率	143.735625	143.75	143.764375	kHz	
Ts	位周期	5.119488	5.12	5.120512	ms	
Tt	Tx 周期	168.943104	168.96	168.976896	ms	3 个字
Tq	静默周期	901.029888	901.12	901.210112	ms	16 个字
Tc	循环周期	1069.972992	1070.08	1070.187008	ms	19 个字

## 4.2 PWM 模式

在 PWM 模式下，C2000 F28379D 生成两个直接进入 AFE 器件的对称 PWM 信号。这两个对称的 PWM 信号分别具有 66% 和 33% 的占空比。这些信号在 AFE 器件内累加，形成噪声极小的波形。图 4-1 显示了这种累加如何发挥作用。

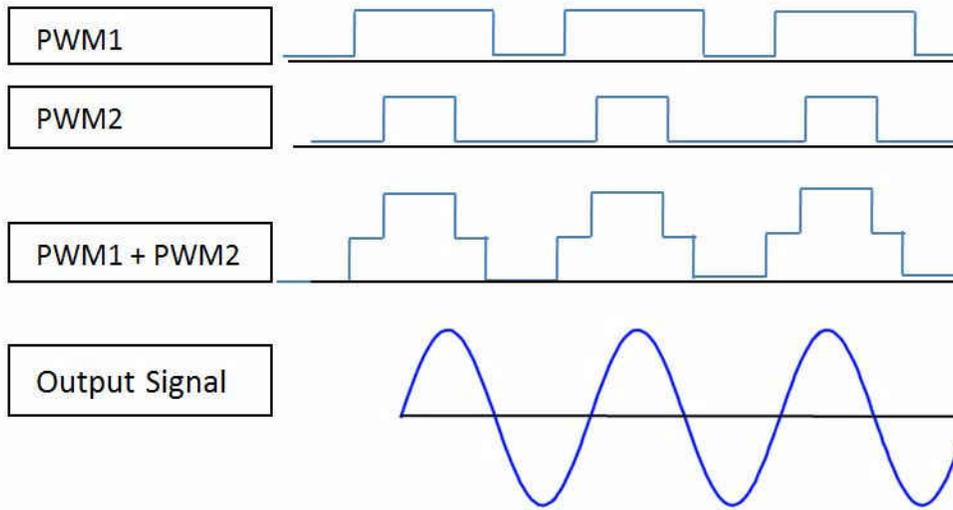


图 4-1. PWM 累加

图 4-2 显示了 PWM 信号的路径。PWM 信号进入 AFE030/1 器件内部的低通滤波器，并累加在一起，形成上述的 PWM1+PWM2 波形。

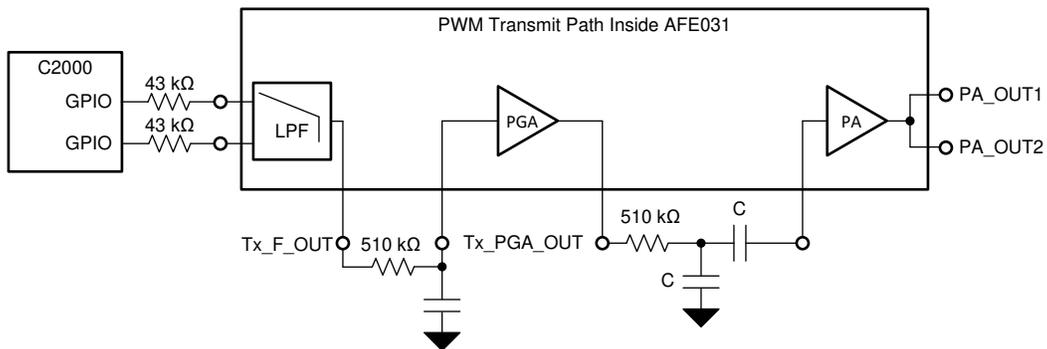


图 4-2. PWM 发送路径

图 4-3 显示了内部 TX 低通滤波器输出端在特定频率下的增益。

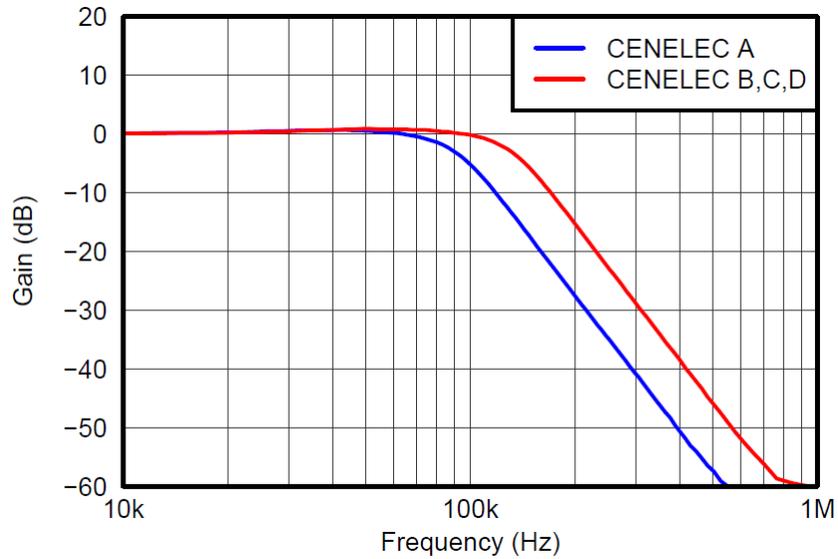


图 4-3. TX 滤波器增益与频率之间的关系

接下来，信号依次通过一个低通滤波器、PGA 和另一个低通滤波器，最后离开 PA。可调整外部低通滤波器，以过滤表 4-2 中所示的所需频率。

表 4-2. 用于增加 PWM 应用中滤波器响应的外部 R 和 C 值

频段	R ( $\Omega$ )	C (nF)
SFSK : 63kHz、74kHz	510	2.7
CENELEC A	510	1.5
CENELEC B、C、D	510	1

### 4.2.1 软件实现

引用的示例程序：`boostxl_afe031_f28379d_pwmmode`

若要在软件中启用 PWM 模式，需要完成以下流程：

- 启用 PWM 发送模式
  - 若要启用 PWM 发送模式，请将使能寄存器中的 TX 和 PA 位设置为 1，并将 DAC 位设置为 0。
  - 软件示例函数：`HAL_afe031_txPWMEnable()`;
- 禁用 DAC 发送模式

使用两个 PWM 源来创建两个 PWM 信号：一个 PWM 源用于设置两个输出信号的频率，另一个用于控制发送数据的比特率。在软件示例中，PWM2 用于控制比特率并产生中断以确定需要输出的频率。

在软件示例中，PWM2 中断处理所有 FSK 协议要求。实现的协议是可重复的模式，它允许软件基于周期数。一个周期数是一位的时长。在该实现中，发送了 33 位（每个字 11 位，三个字）。在每个周期中，它会检查下一位的值，并且 PWM 频率会更改为标记频率或空间频率。33 个周期之后，系统停止发送 PWM 信号并进入静默模式。209 个周期之后，会重置周期数，软件开始再次发送数据包。在由 PWM2 中断来处理 FSK 传输的情况下，CPU 的主要功能被释放，可用于其他应用。默认情况下，软件示例将发送表 4-1 中引用的 `packet_1`，但可以通过将 `packet_to_send` 变量设置为零来将其更改为 `packet_0`。

### 4.2.2 测试结果

图 4-4 和图 4-5 显示系统同时创建了标记频率和空间频率。此外，图 4-6 显示了正从系统发出的完整数据包。

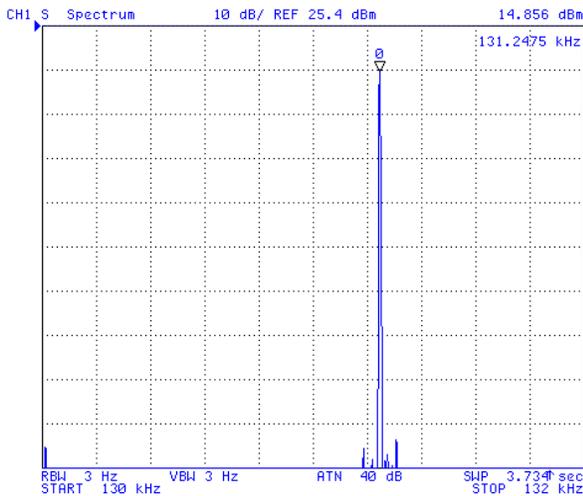


图 4-4. PWM 模式标记频率，131.25kHz

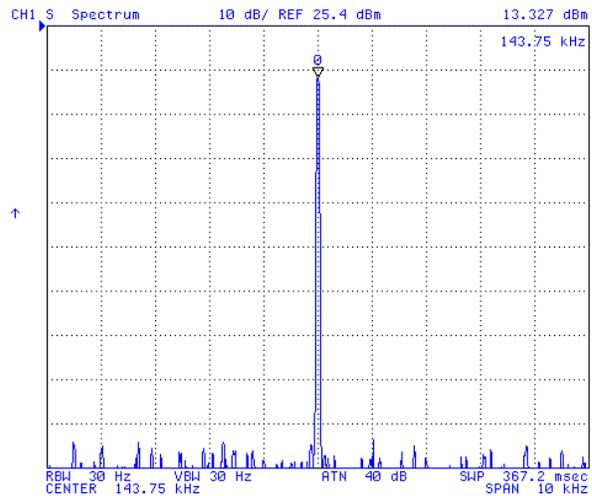


图 4-5. PWM 模式空间频率，143.75kHz

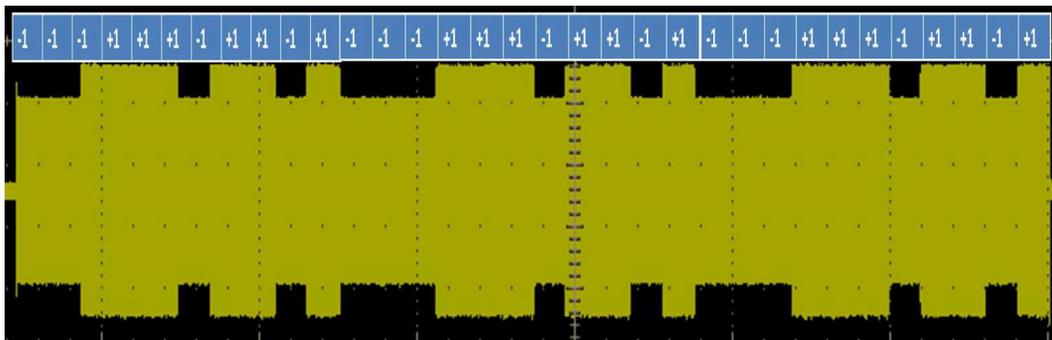


图 4-6. 完整的数据包传输波形

### 4.2.3 HRPWM 与 EPWM

C2000 器件的独特之处在于能够使用高分辨率 PWM (HRPWM)。HRPWM 可以提高 PWM 信号的占空比和周期的分辨率。在本例中，HRPWM 用于生成标记频率和空间频率。

HRPWM 基于微边沿定位 (MEP) 技术。MEP 逻辑能够通过细分传统 PWM 发生器的一个粗略系统时钟来非常精细地定位边沿。时间阶跃精度约为 150ps。

表 4-3 显示了使用和不使用 HRPWM 时可能实现的分辨率。

表 4-3. PWM 和 HRPWM 的分辨率

PWM 频率 (kHz)	常规分辨率 (PWM)		高分辨率 (HRPWM)	
	100MHz EPWMCLK			
	位	%	位	%
20	12.3	0.02	18.1	0.000
50	11	0.05	16.8	0.001
100	10	0.1	15.8	0.002
150	9.4	0.15	15.2	0.003
200	9	0.2	14.8	0.004
250	8.6	0.25	14.4	0.005
500	7.6	0.5	13.4	0.009
1000	6.6	1	12.4	0.018
1500	6.1	1.5	11.9	0.027
2000	5.6	2	11.4	0.036

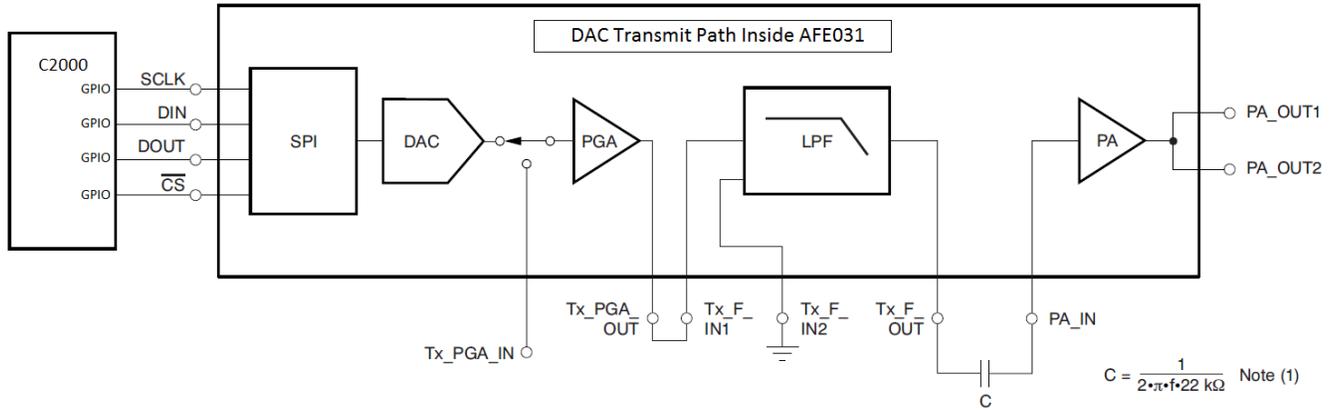
例如，标记频率是在 131.25kHz 信号处生成的。在使用 PWM 时，只能生成 131.2kHz 或 131.3kHz 信号。该限制是 PWM 模块可用的分辨率导致的。如果添加 HRPWM，则可以有效生成 131.250kHz 信号。

如果不需要该精度或使用正常的 PWM 即可达到所需的频率，则不需要 HRPWM。HRPWM 是对 PWM 的扩展。若要在软件示例中禁用该功能，请在 main.c 文件中删除与其相关联的代码。

### 4.3 DAC 模式

本节介绍了一种创建 DAC 模式 FSK 发送器的方法。DAC 模式与 PWM 模式非常相似，两者都使用 PWM 中断来完成 FSK 传输。

图 4-7 显示了采用 DAC 模式时使用的发送路径。



(1) For capacitor value  $C$ ,  $f$  is the desired lower cutoff frequency and  $22\text{ k}\Omega$  is the PA input resistance.

图 4-7. DAC 发送路径

在 DAC 模式下，软件通过 SPI 将数据发送到内部 DAC 以设置输出值。若要完成单频或正弦波发送，应发送给给定频率下的 DAC 梯度值。使用每个周期发生的一个 PWM 中断，以由 PWM 周期确定的频率发送更新的 DAC 值。正弦表中的点数和所需的频率将决定 PWM 信号的频率。例如，如果给定一个包含十个点的  $100\text{kHz}$  正弦波，则需要以  $100\text{kHz}$  的频率发送全部十个点。这意味着 PWM 必须生成遵循以下公式的中断：

PWM 频率 = 正弦表的点数 \* 所需的频率

图 4-8 在时域中显示了对应的数据。

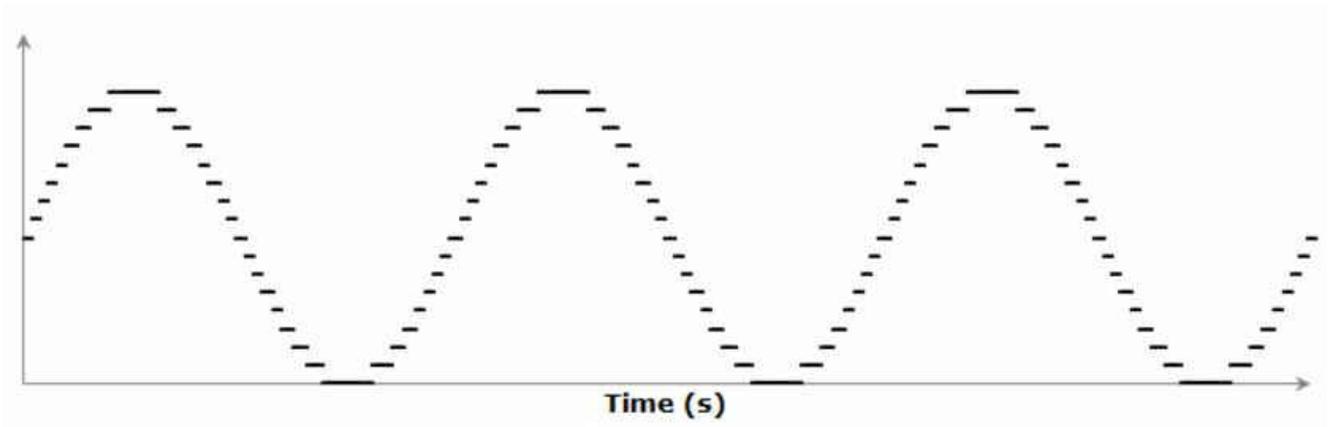


图 4-8. DAC 正弦波斜坡

### 4.3.1 软件实现

引用的示例程序：`boostxl_afe031_f28379d_dacmode`

若要在软件中启用 DAC 模式，需要完成以下流程：

- 启用 AFE 的内部 DAC 发送模式。
  - 若要启用 PWM 发送模式，请将使能寄存器中的 TX 和 PA 位设置为 1，并将 DAC 位设置为 0。
  - 软件示例函数：`HAL_afe031_txDACEnable()`;
- 通过 GPIO 切换启用 DAC 发送模式并配置 SPI 的字长。
  - 将连接 DAC 引脚的 GPIO 设置为 1。
  - 将 SPI 通信的字长设置为 10 字节。
  - 软件示例函数：`HAL_afe031_dacEnable()`;

在 DAC 模式下发送信息的方式与 PWM 模式的工作方式非常相似。一个 PWM 源用于将 DAC 模式值设置为发出的正弦斜坡的正确值。第二个 PWM 用于比特率以产生中断并确定需要输出的频率。

如果尝试使用该实现来达到非常精确的频率，则必定会出现问题。可以通过[方程式 1](#) 来查看该问题。

$$\text{正弦表中的阶跃数} = (\text{中断频率}) / (\text{所需信号的频率}) \quad (1)$$

如果尝试生成 131.25kHz，那么只能更改两个变量。一种方法是在正弦表中设置阶跃数。例如，如果正弦表中有 10 个阶跃，则中断频率为：

$$\text{中断频率} = (\text{正弦表中的阶跃数}) * (\text{所需信号的频率})$$

$$\text{中断频率} = 1.3125\text{MHz}$$

在使用 200MHz 时钟时，即使在 152 或 153 个 CPU 周期中发生中断，中断频率也将分别为 1.31579MHz 和 1.30719MHz。这些频率不处于[表 4-1](#) 中频率容差的规格范围之内。这意味着设置阶跃大小无法正确实现精确的频率生成。

另一种思路是设置中断频率。让我们将中断设置为 1MHz，这是可以生成的频率。正弦波中的阶跃数将是：

$$\text{正弦表中的阶跃数} = (1\text{MHz}) / (131.25\text{kHz})$$

$$\text{正弦表中的阶跃数} = 7.61905$$

利用 F28379D 的浮点功能，处理器可以跟踪该余数，现在精度取决于正弦表，而不是 1MHz 时钟。可以使用以下公式来确定阶跃大小：

$$\text{阶跃大小} = (\text{正弦表中的点数}) / (\text{正弦表中的阶跃数})$$

利用 4096 正弦表并继续执行前一个示例，这样可以将阶跃大小确定为 537.6。这意味着在每次中断中，正弦表都会再产生 537.6 的阶跃。程序将对数组进行排序，因此该数字将去尾取整为 537，但添加该数字后，程序在正弦表中移动时，下一个阶跃将忽略阶跃大小的小数部分。下面显示了一个中断例程示例：

- `//Transmit next data point in sine wave.Also convert float SinePosition to unsigned int`
  - `Uint16 temp = sinePosition;`
  - `HAL_spi_xmt((sineTable[temp]));`
- `//Calculate next step`
  - `sinePosition += sineStep; }`
- `//Check for overflow`
  - `if(sinePosition > 4095)`
  - `{ sinePosition -= 4095; }`

需要注意的一点是，在该实现中创建了一个 1MHz 中断，它仅将数据从一个存储器地址移动到另一个存储器地址。一种降低 CPU 密集操作的方法是利用 C2000 的 DMA (直接存储器访问)。DMA 外设根据触发事件将数据从一个存储器地址移动到另一个存储器地址。使用被选为通过 SPI 发送的正确数据来预填充两个缓冲区，并使用 DMA 在两者之间进行切换。当从一个缓冲区切换到另一个缓冲区时，旧缓冲区中将重新填充这些值。

在 DAC 模式软件中，使用一个 1MHz PWM 信号来生成一个 DMA 事件，将数据从这些缓冲区之一移动到 SPI TX 缓冲区中。每次使用后都需要重新填充旧缓冲区，因此 DMA 将在每次完成整个缓冲区读取后触发一个事件。在该中断期间，会对缓冲区进行切换，旧缓冲区被重新填充。通过该实现，可以调整 CPU 利用率与存储器之间的权衡点，并且可以相应地调整缓冲区的大小。如果有额外的存储器，则可以通过使用更大的缓冲区来减少 CPU 开销。如果没有大量的可用存储器，则较小的缓冲区大小会增加 CPU 开销。

PWM2 中断以与 PWM 模式软件实现完全一样的方式处理 FSK 协议的所有需求。要实现的协议是可重复的模式，它允许软件基于周期数。一个周期数是一位的时间。在该实现中，将发送 33 位（每个字 11 位，三个字）。在每个周期中，通过检查下一位的值和阶跃大小变化，能够以标记频率或空间频率发送正弦波。33 个周期之后，系统停止发送 DAC 值并进入静默模式。209 个周期之后，周期数将被重置，软件开始再次发送数据包。在由 PWM2 中断来处理 FSK 传输的情况下，CPU 的主要功能被释放，可用于其他应用。默认情况下，软件示例将发送表 4-1 中引用的 packet\_1，但可以通过将 packet\_to\_send 变量设置为零来将其更改为 packet\_0。

### 4.3.2 测试结果

图 4-9 和图 4-10 显示系统同时创建了标记频率和空间频率。此外，图 4-11 显示了正从系统发出的完整数据包。

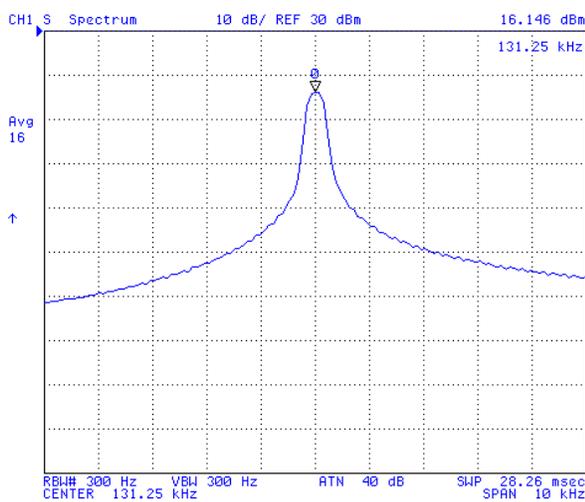


图 4-9. DAC 模式标记频谱分析器

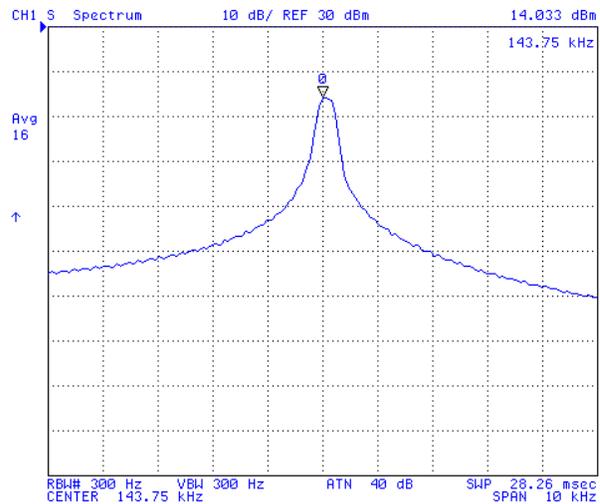


图 4-10. DAC 模式空间频谱分析器

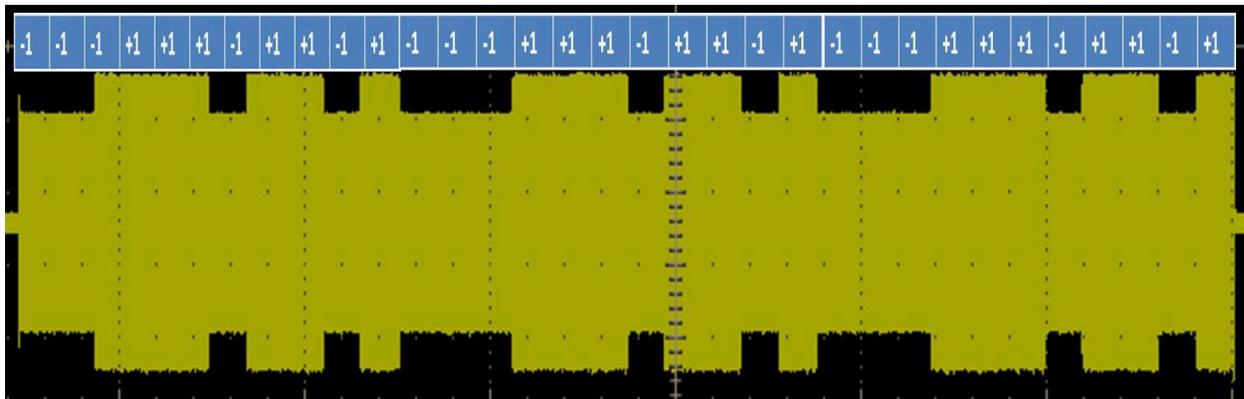


图 4-11. 完整的数据包传输波形

### 4.3.3 OFDM 功能

虽然未实现，但使用 AFE 器件的 DAC 模式可以实现 OFDM 频率调制功能。利用能够在每个点选择准确的 DAC 值这一功能，可以在 MCU 侧组合频率分量，并将 DAC 设置为所需的值。

图 4-12 至图 4-15 显示了组合两个频率的示例，其中展示了如何实现该目的。这两个信号是 50Hz 和 100Hz 正弦波。为了独立发送这两个信号，将以正确的时间间隔离散发送每个点，以创建正弦波。

若要同时发送这两个频率，首先需要组合这两个频率。该结果如图 4-14 所示。为了发送该波形中的每个点，系统将同时发送两个频率。对数据执行 FFT 以查看两个离散频率。

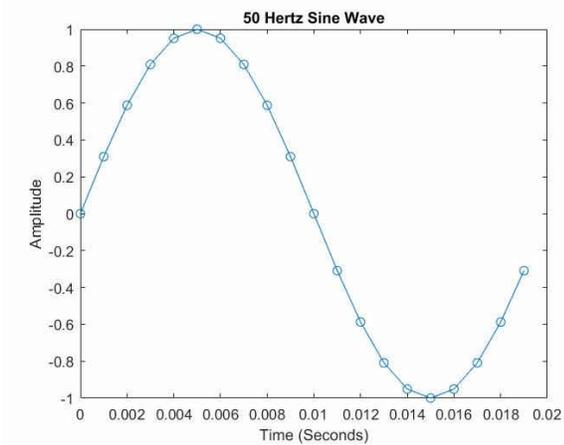


图 4-12. 60Hz 正弦波

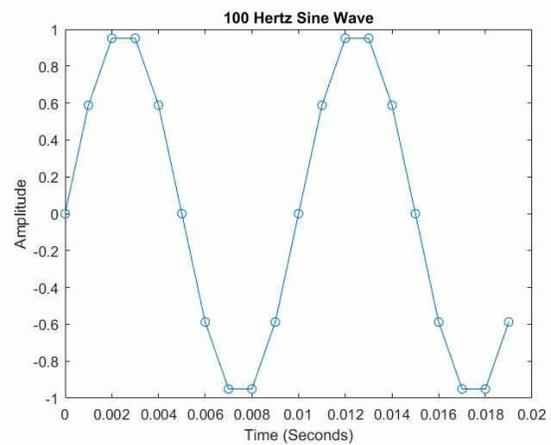


图 4-13. 100Hz 正弦波

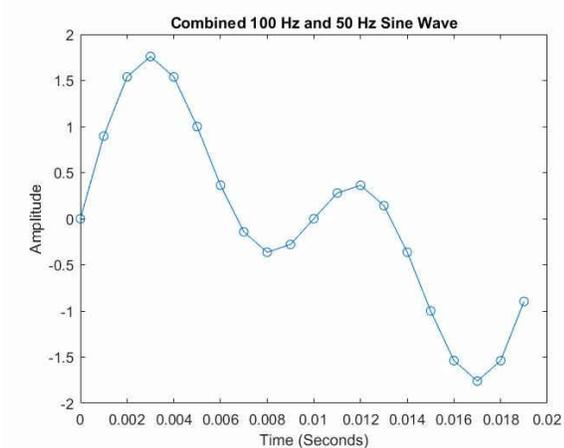


图 4-14. 60Hz + 100Hz 正弦波

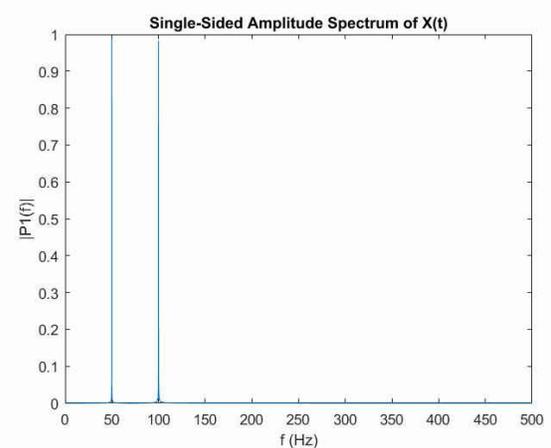


图 4-15. 组合信号的 FFT

#### 4.4 将 TX 移植到 LAUNCHXL-F280049C

在将 boostxl\_afe031\_f28379d\_pwmmode 和 boostxl\_afe031\_f28379d\_dacmode 导入 Code Composer Studio™ 之后，若要将这些工程从 LAUNCHXL-F28379D 移植到 LAUNCHXL-F280049C，需要执行以下步骤。

1. 将 C2000Ware 包含搜索路径从 f2837xd 更改为 f28004x。
  - a. 右键点击工程 --> 选择“Properties” --> 展开“Resource” --> 选择“Linked Resources”。
  - b. 选择“F28379D\_DEVICE\_SUPPORT\_ROOT”并点击“Edit”。
  - c. 将名称从 F28379D\_DEVICE\_SUPPORT\_ROOT 更改为 F28004X\_DEVICE\_SUPPORT\_ROOT。
  - d. 将目录位置更改为 \device\_support\f28004x (而不是 \device\_support\f2837xd)。
  - e. 在“Properties”中，展开“CCS Build” --> 展开“C2000 Compiler” --> 选择“Include Options”。
  - f. 将包含路径从 F28379D\_DEVICE\_SUPPORT\_ROOT 的更改为 F28004X\_DEVICE\_SUPPORT\_ROOT。
  - g. 选择“Apply and Close”。
2. 将工程中的所有 F2837xD 源文件替换为 f28004x 源文件。
  - a. 可以在 C2000Ware 目录 C:\ti\c2000\C2000Ware\_<version>\device\_support\f28004x\common\source 中找到 f28004x 源文件
3. 用 28004x 链接器命令文件替换 2837x 链接器命令文件。
  - a. 可以在 C2000Ware 目录 C:\ti\c2000\C2000Ware\_<version>\device\_support\f28004x\common\cmd 中找到通用闪存和 RAM 28004x 链接器命令文件。

4. 更改 boostxl\_afe031\_f28379d\_main.c 中 LED 控制的 GPIO 配置。
  - a. 更改在 AFE\_InitGpio 函数中配置的 GPIO。

```
//GPIO-23 - LaunchPad RED LED
GPIO_SetupPinMux(23, GPIO_MUX_CPU1, 0);
GPIO_SetupPinOptions(23, GPIO_OUTPUT, GPIO_PUSHPULL);
// GPIO-34 - LaunchPad GREEN LED
GPIO_SetupPinMux(34, GPIO_MUX_CPU1, 0);
GPIO_SetupPinOptions(34, GPIO_OUTPUT, GPIO_PUSHPULL);
```

5. 更改在 `afe031_config.c` 中为 AFE031 控制配置的 GPIO。a. 更改在 `HAL_afe031_cfgGpio` 函数中配置的 GPIO：

```
//SD PIN
GPIO_SetupPinMux(9, GPIO_MUX_CPU1, 0);
GPIO_SetupPinOptions(9, GPIO_OUTPUT, GPIO_PUSHPULL);
// SD=0
GPIO_WritePin(9, 0);
//DAC Pin Enable
GPIO_SetupPinMux(8, GPIO_MUX_CPU1, 0);
GPIO_SetupPinOptions(8, GPIO_OUTPUT, GPIO_PUSHPULL);
// DAC=0
GPIO_WritePin(8, 0);
```

b. 在 `HAL_afe031_cfgGpio` 函数中删除或注释掉 INT 引脚的 GPIO 配置。

```
// INT Pin - Not available on LAUNCHXL-F280049C
// GPIO_SetupPinMux(123, GPIO_MUX_CPU1, 0);
// GPIO_SetupPinOptions(123, GPIO_OUTPUT, GPIO_PUSHPULL);
```

6. 更改在 `hal_spi.c` 中为 SPI 通信配置的 GPIO：a. 更改在 `InitAFESpiGpio` 函数中配置的 GPIO。

```
EALLOW;
// SPI_MOSI
GPIO_SetupPinOptions(16, GPIO_INPUT, GPIO_ASYNC | GPIO_PULLUP);
// SPI_MISO
GPIO_SetupPinOptions(17, GPIO_INPUT, GPIO_ASYNC | GPIO_PULLUP);
// SPI_CS
GPIO_SetupPinOptions(57, GPIO_INPUT, GPIO_ASYNC | GPIO_PULLUP);
// SPI_CLK
GPIO_SetupPinOptions(56, GPIO_INPUT, GPIO_ASYNC | GPIO_PULLUP);
GPIO_SetupPinMux(16, GPIO_MUX_CPU1, 1);
GPIO_SetupPinMux(17, GPIO_MUX_CPU1, 1);
GPIO_SetupPinMux(57, GPIO_MUX_CPU1, 1);
GPIO_SetupPinMux(56, GPIO_MUX_CPU1, 1);
EDIS;
```

7. 更改在 `hal_afe031.h` 中为 DAC 启用而写入的 GPIO。a. 更改写入 `HAL_afe031_dacEnable` 宏的 GPIO：

```
GPIO_WritePin(8, 1);}
```

b. 更改写入 `HAL_afe031_dacDisable` 宏的 GPIO：

```
GPIO_WritePin(8, 0); \
```

8. 删除 `boostxl_afe031_f28379d_main.c` 中的 `InitEPwm1Gpio()` 函数。

#### 4.4.1 特定于 PWM 模式的移植

以下步骤专门用于移植 boostxl\_afe031\_f28379d\_pwmmode。

1. 更改 HRPWM 库的搜索路径。
  - a. 右键点击工程 --> 选择“Properties” --> 展开“Build” --> 展开“C2000 Linker” --> 选择“File Search Path”。
  - b. 将 HRPWM 库的搜索路径从 f2837x 目录更改为 f28004x 目录。

```
${F28004X_DEVICE_SUPPORT_ROOT}/../../../../libraries/calibration/hrpwm/f28004x/lib
```

2. 将易失性结构 \*ePWM[PWM\_CH] 更改为 \*ePWM[9]。
3. 使用 ePWM6 代替 ePWM1。
  - a. 将所有 EPwm1Regs 实例替换为 EPwm6Regs。(提示：使用 CCS 中的全部替换功能。)
  - b. 将 EPWM1 函数和 #define 重命名为 EPWM6 不是必需的，但有助于提高代码的可读性。

#### 4.4.2 特定于 DAC 模式的移植

以下步骤专门用于移植 boostxl\_afe031\_f28379d\_dacmode。

1. 删除以下代码行：

```
CpuSysRegs.SECMSEL.bit.PF2SEL = 1;
```

2. 在正弦表的链接器命令文件中分配 RAM。
  - a. 请参阅默认的 2837x\_afe031 链接器命令文件，以了解如何针对 RAM 和闪存配置完成该操作。
  - b. 建议为默认的正弦表组合并分配至少两个 RAM 块。
3. 进行以下更改以正确配置 F28004x SPI 时钟频率。
  - a. 更改 hal\_spi.h 中的 HAL\_SPI\_LSPCLK 定义：

```
#define HAL_SPI_LSPCLK 0 //LSPCLK=SYSCLK, make 0 for f28004x
```

4. 将 pingBuf 和 pongBuf 缓冲区映射到全局共享 RAM (如果尚未执行该操作)。
  - a. 将以下代码行添加到 boostxl\_afe031\_f28379d\_main.c 的顶部：

```
#pragma DATA_SECTION(pingBuf, "ramgs0"); // map pingBuf to memory
#pragma DATA_SECTION(pongBuf, "ramgs1"); // map pongBuf to memory
```

将示例程序移植到其他器件和 LaunchPad 的说明与前面所述的说明类似。

## 5 接收路径

C2000 AFE031 接口也可用作 FSK 接收器，以将传输的频移信号转换为数字数据。本节介绍了一种使用 C2000 AFE031 接口创建 FSK 接收器的方法。

### 5.1 接收路径概述

图 5-1 显示了 C2000 AFE031 系统接收路径。输入信号必须沿着从右侧变压器到左侧 C2000 ADC 输入端的路径经过大量滤波环节。在 AFE 侧，AFE031 对这些信号具有强大的滤波能力。

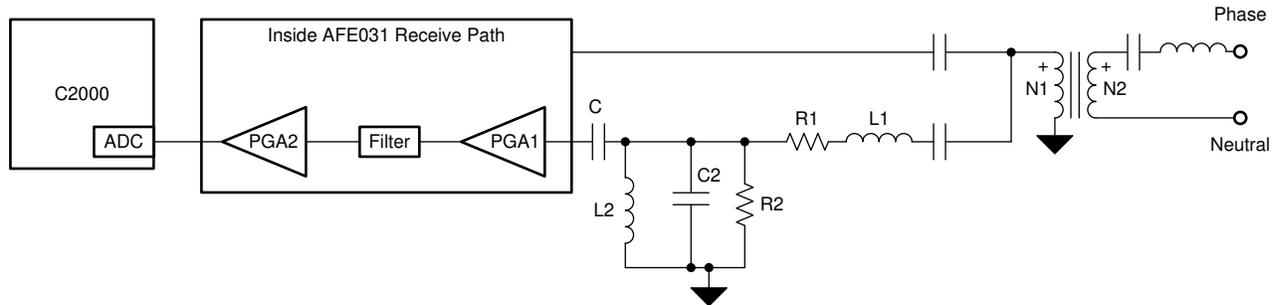


图 5-1. 与 C2000 ADC 连接的 AFE031 接收路径

AFE031 RX 路径由 Rx PGA1、Rx 低通滤波器和 Rx PGA2 组成。Rx PGA1 和 Rx PGA2 都是可通过 SPI 进行配置的高性能可编程增益放大器。Rx PGA1 可以作为衰减器运行（提供损耗），也可以作为放大器运行（提供增益）。Rx PGA1 的增益阶跃为 0.25V/V、0.5V/V、1V/V 和 2V/V。Rx PGA2 的增益阶跃为 1V/V、4V/V、16V/V 和 64V/V。有关特定的 RX PGA 增益选择寄存器值，请参阅表 5-1。将 Rx PGA1 配置为衰减器（增益小于 1V/V）对于信号频带内存在大干扰信号的应用非常有用。通过对严重的干扰进行衰减，这些信号可以通过模拟 Rx 信号链而不会导致过载；然后可以根据需要在微控制器内处理和消除干扰信号。

表 5-1. AFE031 RX PGA 增益设置

位名字	位置 (0 = LSB)	默认	读/写	功能
RX1G-0、RX1G-1	0, 1	0, 1	读/写	该位用于设置 RX PGA1 的增益。 00 = 0.25V/V 01 = 0.5V/V 10 = 1V/V 11 = 2V/V
RX2G-0、RX2G-1	2, 3	0,0	读/写	该位用于设置 RX PGA2 的增益。 00 = 1V/V 01 = 4V/V 10 = 16V/V 11 = 64V/V

Rx 滤波器是噪声非常低、具有单位增益的四阶低通滤波器。可以在 CENELEC A 或 CENELEC B、C、D 模式之间选择 Rx 滤波器截止频率（在控制寄存器中设置）。Rx 滤波器是噪声很低的模拟滤波器，因此需要使用两个外部电容器（如图 5-2 所示）来正确配置 Rx 滤波器。表 5-2 显示了 CENELEC A 和 B、C、D 频带的正确电容值。

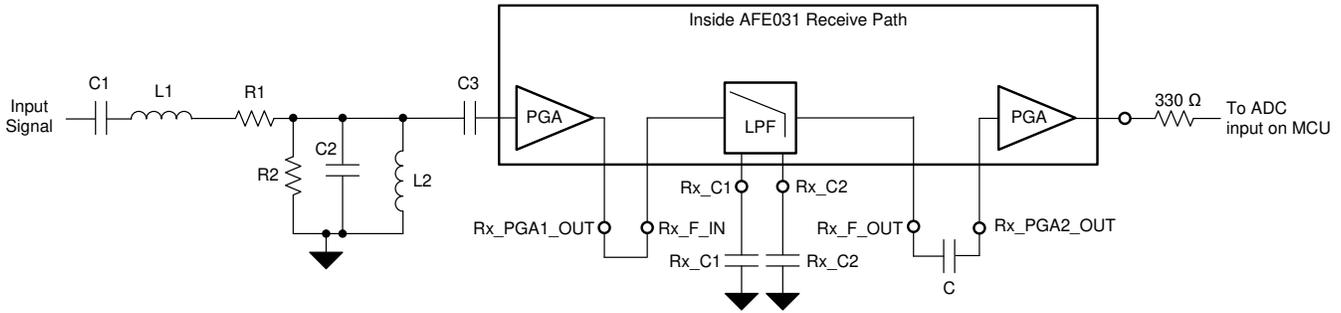


图 5-2. AFE031 接收路径的外部滤波

表 5-2. 建议的 Rx 滤波器外部电容器值

频段	Rx C1 (引脚 24)	Rx C2 (引脚 23)	截止频率 (kHz)
CENELEC A	680pF	680pF	90
CENELEC B、C、D	270 pF	560 pF	145

电容器 Rx C1 连接在引脚 24 和接地端之间，Rx C2 连接在引脚 23 和接地端之间。对于所示的电容器，建议这些元件的额定值能够承受最大的 AVDD 电源电压。

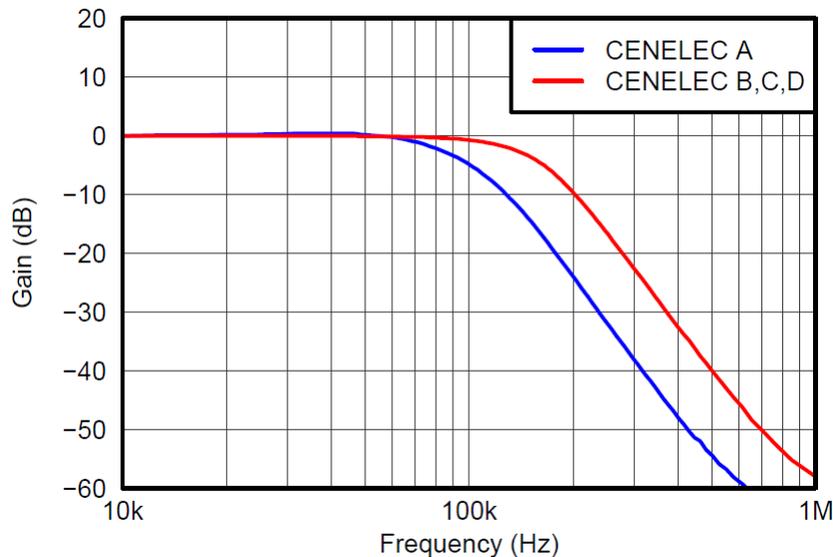


图 5-3. RX 滤波器增益与频率之间的关系

图 5-3 显示了在特定频率下在 RX 低通滤波器输出端看到的增益。显示的这一响应是正常的 AFE03x 工作条件下的响应。衰减将从略早于表 5-2 中实际截止频率的频率开始。

外部四阶无源带通滤波器是可选的，但建议将其用于需要高性能的应用。外部无源带通滤波器从信号路径中消除任何不需要的带外信号，防止其到达 AFE031 内的有源内部滤波器。表 5-3 显示了具有 0dB 通带的 CENELEC A 或 CENELEC B、C、D 所需的值。BoosterPack 上使用的元件值适用于具有 0dB 通带的 CENELEC B、C、D。

表 5-3. 四阶无源带通滤波器的建议元件值 (0dB 通带衰减)

频段	频率范围 (kHz)	特征阻抗 (Ω)	R1 (Ω)	R2 (Ω)	C1 (nF)	C2 (nF)	L1 (μH)	L2 (μH)
CENELEC A	35 至 95	1k	1k	10k	4.7	1.5	1500	4700

表 5-3. 四阶无源带通滤波器的建议元件值 (0dB 通带衰减) (continued)

频段	频率范围 (kHz)	特征阻抗 ( $\Omega$ )	R1 ( $\Omega$ )	R2 ( $\Omega$ )	C1 (nF)	C2 (nF)	L1 ( $\mu$ H)	L2 ( $\mu$ H)
CENELEC B、C、D	95 至 150	1k	1k	10k	1.7	1	1200	1500
SFSK	63 至 74	1k	1	10k	2.7	2.2	2200	2200

有关具有不同通带衰减的其他带通滤波器元件值，请参阅 [AFE031 电力线通信模拟前端数据表](#)。

## 5.2 接收器软件实现

引用的示例程序：`boostxl_afe031_f28379d_rx`

该软件示例用于执行以下操作：

- 在接收到的 FSK 信号通过 AFE031 接收路径之后对其进行连续采样
- 对采样的值执行基于相关性的算法，以检测是否正在接收标记频率或空间频率
- 根据检测到的频率及其持续时间辨别标记位和空间位
- 存储接收到的位并将其打包成所需的可用格式

为了使系统能够在软件中进行接收，需要完成以下操作：

- 必须将使能寄存器中的 RX 位设置为 1 才能打开 AFE031 接收路径。
  - 软件示例函数：`HAL_afe031_rxEnable()`;
- 在工程中包含 `fsk_corr_detect` 库和相关头文件：
  - `fsk_corr_detect.lib`
  - `fsk_corr_detector.h`
  - `fsk_packetization.h`

### 5.2.1 初始设置和参数

接收到的 FSK 信号遵循一组必须在软件内设计的通信参数。

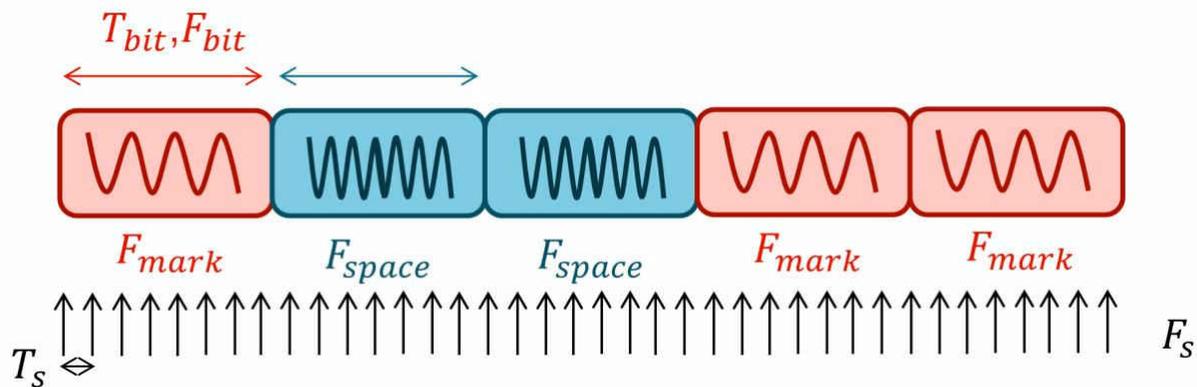


图 5-4. 接收的 FSK 信号

图 5-4 说明了相关的通信参数。标记的参数具有以下含义：

- $T_{bit}$ ：位周期
- $F_{bit}$ ：位频率
- $F_{mark}$ ：标记频率
- $F_{space}$ ：空间频率
- $T_S$ ：采样周期
- $F_S$ ：采样频率

对于示例程序，默认遵循的通信参数如表 4-1 所示。创建一个 `FSK_CORR_DETECTOR` 结构（在 `fsk_corr_detector.h` 中声明），以保存实现准确接收所需的参数。

```
volatile FSK_CORR_DETECTOR FSK_struct1; // FSK structure
```

示例软件和 `fsk_corr_detect` 库用于检测一组用户指定的频率、一个标记频率和一个空间频率。这些频率必须处于 AFE031 的 CENELEC A 或 CENELEC B、C、D 配置的频带范围之内。示例程序使用 131.25kHz 的标记频率和 143.75kHz 的空间频率，旨在与 CENELEC B、C、D 配置一同使用。使用这些频率来设置 `FSK_CORR_DETECTOR` 结构的 `mark_freq` 和 `space_freq` 成员。

```
FSK_struct1.mark_freq = 131250; // Mark Frequency Detected
FSK_struct1.space_freq = 143750; // Space Frequency Detected
```

C2000 的 ADC 用于对 FSK 输入信号进行采样。采样频率  $F_S$  必须遵循奈奎斯特定理，即输入信号的采样率必须至少是试图检测到的最高信号频率的 2 倍。也就是说，如果要检测的最高信号频率为 100kHz，则  $F_S$  必须至少为 200kHz。在示例程序中，要检测的最高频率是 143.75kHz 的空间频率，采样率设置为 300kHz，这高于所需的频率。将 `FSK_CORR_DETECTOR` 结构的 `isr_freq` 成员设置为可接受的  $F_S$ 。

```
FSK_struct1.isr_freq = 300000; // ADC Sampling frequency
```

位判定算法的运行频率应该是位频率的三倍。例如，如果每个位周期为 1ms，则位频率为 1kHz，从而使所需的位判定频率为 3kHz。示例程序以 5.12ms 的周期检测位，从而使位频率为 195.3125Hz，所需的位判定频率为 585.9375Hz。软件中的位判定频率应尽可能接近所需的频率，以防止出现位边界问题。使用该位判定频率来设置 `FSK_CORR_DETECTOR` 结构的 `bit_freq` 成员。

```
FSK_struct1.bit_freq = 586; // Bit decision frequency, 3x bit frequency
```

总之，为示例程序设置的频率参数如表 5-4 所示。

表 5-4. 软件频率参数

参数	频率
检测到的标记频率	131.25 kHz
检测到的空间频率	143.75 kHz
输入信号采样频率	300kHz
位判定算法频率	586Hz ( 经过向上舍入 )

设置 `FSK_CORR_DETECTOR` 结构的 `detection_threshold` 成员。该值对于调整位检测灵敏度而言有一定的作用。

```
#define FSK_BIT_DETECTION_THRESHOLD 0.1 // Bit detection threshold value
FSK_struct1.detection_threshold = FSK_BIT_DETECTION_THRESHOLD; // Set threshold
```

根据通过调用相应的 `init` 函数输入的的成员值完成 `fsk_corr_detect` 库的初始化。

```
FSK_CORR_DETECTOR_INIT(&FSK_struct1); // Initialize FSK structure
```

此外，通过在软件中设置以下参数来考虑所接收信息的格式。

- 构成一个字的位数（`fsk_packetization.h` 中的 `#define`）：
  - `#define NUMBER_OF_BITS_PER_WORD 11`
- 构成一个数据包的字数（`fsk_packetization.h` 中的 `#define`）：
  - `#define NUMBER_OF_WORDS 3`
- 一个数据表中的总位数（`fsk_corr_detector.h` 中的 `#define`）：
  - `#define RX_MESSAGE_SIZE 33`

### 5.2.2 中断服务例程

以这些预定频率运行的 ISR 执行接收器解决方案的主要功能。示例程序利用 C2000 的 EPwm 和 CPU 计时器来触发必要的中断。

由 EPwm1 配置的 ADC 采样 ISR 被设置为以 300kHz 输入信号采样频率触发。ISR 函数对 ADC 进行采样，对采样值进行调节，并将经调节的值传递给库函数以进行信号处理。

由 EPwm2 配置的位判定 ISR 被设置为以 585.92Hz 的频率进行触发。该频率非常接近于该 EPwm 模块可以达到的所需频率，即 585.9375Hz。ISR 函数接受先前 ADC 采样 ISR 的信号处理工作，并检查是否检测到标记位或空间位。如果检测到标记位或空间位，则将检测到的位放入接收消息缓冲区。一旦消息缓冲区已满，将设置一个标志以表示已收到完整的数据包。

由 CPU 计时器 2 配置的消息超时 ISR 被设置为在接收时达到用户指定的时间限制时触发。默认情况下，该时间限制由 `RX_MESSAGE_TIMEOUT #define` 设置为 3 秒。计时器恰好在系统开始接收之前启动，并在收到数据包时重置。如果在指定的时间限制内没有接收到数据包，则 ISR 将触发并导致系统停止接收。

### 5.2.3 运行时工作

在运行时期间，接收器以下列方式工作：

1. 系统将启动 EPwm 和 CPU 计时器来触发中断，以开始连续接收传入的数据。
  - a. 示例程序函数：`Start_Receiving()`;
2. ISR 将一直运行，直到接收到完整的数据包或达到规定的超时时间。
  - a. ISR 在运行时将使用接收到的位来填充 `rxMessage[]` 缓冲区
3. 在接收到数据包或达到规定的超时时间后，系统将停止接收数据，以降低静默周期期间 CPU 的使用率。
  - a. 示例程序函数：`Stop_Receiving()`;
  - b. 该函数使 EPwm 停止并使 CPU 计时器停止并重置
4. 然后将接收到的数据打包成所需的格式。
  - a. 示例程序函数：`Packetize(int message[], int packet[])`;
  - b. 该函数获取包含 33 个接收位的 `rxMessage[]` 缓冲区，并将每个字的接收位相加，以用三个 11 位字填充 `packet[]` 缓冲区。然后 `rxMessage[]` 缓冲区中的内容被设置为零。
    - i. 根据表 4-1 中的规格，+1 等于 W1
    - ii. 根据表 4-1 中的规格，-1 等于 W0
  - c. 该函数对 `packet[]` 的值求和，并将总和保存到 `packet_sum` 变量中
    - i. 根据表 4-1 中的规格，+3 等于 `packet_1`
    - ii. 根据表 4-1 中的规格，-3 等于 `packet_0`

5. 在接收过程重新启动之前使用打包的数据。
  - a. 示例程序函数：*Visual\_Indication()*;
  - b. 该函数使 BOOSTXL-AFE031 上相应的 LED 闪烁
    - i. 如果接收到 `packet_1`，则蓝色 LED 会闪烁
    - ii. 如果接收到 `packet_0`，则红色 LED 会闪烁
    - iii. 如果既没有接收到 `packet_1`，也没有接收到 `packet_0`，或者发生超时，则红色和蓝色 LED 会闪烁

#### 5.2.4 测试结果

使用 PWM 或 DAC 模式发送器解决方案来提供输入 FSK 信号，以在受控的实验室环境中对 FSK 接收器解决方案的性能进行了测试。建立的设置类似于图 2-8 所示的设置，未采用任何耦合电路。测试的目的是确认接收器解决方案是否可以有效地将接收到的 FSK 信号转换为原始数字信息。

当正确地按照节 5.2 中讨论的指南进行操作时，C2000 MCU 可以接收 FSK 输入信号并准确地解码其中包含的每个数据位。图 5-5 显示了相关内容，其中成功捕获了一个包含三十三位的数据包。

Index	Value
[0]	-1
[1]	-1
[2]	-1
[3]	1
[4]	1
[5]	1
[6]	-1
[7]	1
[8]	1
[9]	-1

图 5-5. 包含接收到的数据位的数据包

该捕获的数据采用 11 位代码字的形式连续传输三次。然后按照图 5-6 所示的格式对每个和为一的代码字进行打包。

Index	Value
[0]	1
[1]	1
[2]	1

图 5-6. 打包的数据

### 5.2.5 系统利用率

表 5-5 列出了 C2000 MCU 用作 FSK 接收器时消耗的资源。

表 5-5. 利用的 C2000 资源

资源名称	类型	用途	使用/限制
ADCINA	模块/IO	用于对输入信号进行采样的 ADC 输入	限于可访问的 ADC 输入
EPwm1	模块	触发信号采样例程的中断	可以通过多种方式进行配置以满足所需的 ISR 频率要求
EPwm2	模块	触发运行位判定例程的中断	可以通过多种方式进行配置以满足所需的 ISR 频率要求
SPI	模块/IO	用于在初始化期间访问 AFE03x 的寄存器	仅用于 RX 解决方案的初始化
CPU 计时器 2	模块	可用于在一定时间内未收到任何信息时产生超时	ISR 的优先级可能低于其他设置的 ISR

表 5-6 列出了每个 fsk\_corr\_detect 库函数消耗的 CPU 周期数。

表 5-6. 库函数 CPU 周期数

函数名称	说明	CPU 周期数	类型
FSK_CORR_DETECTOR_INIT	根据用户在 fsk_corr_detector 结构中设置的频率参数初始化 FSK 库使用的变量	60	初始化
FSK_CORR_DETECTOR_RUN	对采样的 ADC 值执行必要的计算以解调输入信号	59	运行时
FSK_CORR_DETECTOR_OverSamp_RUN	执行逻辑以辨别是否已接收到一个位	134	运行时
Packetize	获取接收到的消息数据缓冲区并构建可用的代码字和数据包	1381	运行时

在默认状态下使用并接收表 4-1 中指定的信息时，可以使用表 5-7 中的信息来计算软件解决方案的运行时 CPU 利用率。

表 5-7. 软件 ISR/函数使用情况

ISR/函数	平均周期数	执行频率
ADC 采样 ISR	74	300kHz
位判定 ISR	175	约 586Hz
打包函数	1381	约 1Hz

**CPU 利用率公式：**

$$\text{CPU Utilization} = ((74 * F_S + 175 * 3 * F_{\text{bit}} + 1381 * F_{\text{Packetization}}) / F_{\text{CPU}}) * 100\%$$

**F28379D 在 200MHz 时钟频率下的 CPU 利用率：**

$$\text{CPU 利用率} = ((74 * 300\text{kHz} + 175 * 586\text{Hz} + 1381 * 1\text{Hz}) / 200\text{MHz}) * 100\% = 11.15\%$$

### 5.2.6 器件相关性和移植

尽管 FSK 接收器解决方案是为 F2837xD 构建并在该系列器件上进行测试的，但该解决方案应该能够直接移植到支持浮点单元 (FPU) 和三角函数加速器 (TMU) 的器件中。fsk\_corr\_detect 库的功能取决于提供的 FPU 和 TMU。同时支持 FPU 和 TMU 的器件包括：F2837xD、F2837xS、F2807x 和 F28004x。

## 5.3 调优和校准

可以在软件和硬件内针对接收对 C2000 AFE031 系统进行调优和校准。以下各小节将对这些概念进行讨论。

### 5.3.1 设置 AFE03X 的 PGA

AFE031 沿 RX 路径的 PGA 用于放大或衰减输入信号，使其在进入 C2000 的 ADC 之前处于所需的电压范围内。ADC 引脚上的信号最好处于 ADC 基准电压定义的范围内。例如，F28379D LaunchPad 的基准电压为 3V；因此，信号应处于 0V 至 3V 范围内以防止削波。

为获得最佳结果，该信号应接近此电压范围，不得超过或低于此电压范围，以利用 ADC 的全分辨率。使用图 5-1 中的增益值设置，使用 AFE031 的 PGA 来调整信号以满足该标准。有关如何在软件中设置 RX PGA 值，请参阅节 3.1 的步骤 10。

### 5.3.2 自动增益控制 (AGC)

如果要在具有不一致振幅的输入信号的应用中使用接收器系统，则可能需要实施某种形式的自动增益控制 (AGC) 以在信号到达 C2000 的 ADC 输入端之前对其进行管理。AGC 是一个闭环反馈系统，用于测量输入信号的振幅或相关响应，并根据观察到的情况放大或衰减信号。

可以利用 AFE031 的内部 PGA 动态放大或衰减信号，使其处于所需的振幅范围之内。有关沿 RX 路径的两个 PGA 的增益选择时间，请参阅 AFE031 数据表。

另一种方法是在信号进入 C2000/AFE031 系统之前在外部执行 AGC。不过，这需要添加涉及可变增益放大器 (VGA) 的附加电路。

### 5.3.3 设置位检测阈值

节 5.2.1 所述的 FSK\_CORR\_DETECTOR 结构的 *detection\_threshold* 成员与位检测的准确性直接相关。需要将阈值设置为介于 0 和 1 之间的值。如果设置的值太高，则根本不会检测到位，但如果设置的值太低，则可能会混淆标记位或空间位的内容并导致位错误。可能需要通过试错来找到能够一致地检测位而不会产生错误的值。

### 5.3.4 FSK 相关性检测器库

如对软件示例中使用的 FSK 相关性检测器库有疑问，请向以下地址发送电子邮件：[C2000-fsk\\_rx\\_source\\_access@list.ti.com](mailto:C2000-fsk_rx_source_access@list.ti.com)。

若要索取访问库的源代码（取决于德州仪器 (TI) 公司的最终应用审查和出口管制批准），请按照以下 URL 中的说明进行操作：[https://www.ti.com/licreg/docs/swlicexportcontrol.tsp?form\\_id=268791&prod\\_no=C2000-AFE031\\_FSK\\_RX\\_SOURCE&ref\\_url=c2000](https://www.ti.com/licreg/docs/swlicexportcontrol.tsp?form_id=268791&prod_no=C2000-AFE031_FSK_RX_SOURCE&ref_url=c2000)。

## 5.4 将 RX 移植到 LAUNCHXL-F280049C

将 boostxl\_afe031\_f28379d\_rx 导入 Code Composer Studio 后，若要将该工程从 LAUNCHXL-F28379D 移植到 LAUNCHXL-F280049C，需要执行以下步骤。

1. 执行节 4.4 中的步骤 1 至 7。
2. 对 boost\_afe031\_f28379d\_rx\_main.c 中的 ConfigureADC(void) 函数进行以下更改。
  - a. 删除以下代码行：

```
AdcSetMode(ADC_ADCA, ADC_RESOLUTION_12BIT, ADC_SIGNALMODE_SINGLE); // 12 bit res
```

- b. 在 EALLOW 之前添加以下代码行：

```
//  
// Setup VREF as internal  
//  
SetVREF(ADC_ADCA, ADC_INTERNAL, ADC_VREF3P3);
```

3. 在 boost\_afe031\_f28379d\_rx\_main.c 中为通道 9 (而不是通道 1) 上的 ePWM 触发转换设置 ADC。
  - a. 将 SetupADCEpwm() 函数调用替换为以下代码行：

```
SetupADCEpwm(9);
```

4. 将 CPU 计时器配置为 100MHz 时钟 (而不是 200MHz 时钟)。
  - a. 将 ConfigCpuTimer() 函数调用替换为以下代码行：

```
ConfigCpuTimer(&CpuTimer2, 100, RX_MESSAGE_TIMEOUT);
```

5. 在 fsk\_corr\_lib\_data 的链接器命令文件中分配全局共享 RAM。
  - a. 请参阅默认的 2837x\_afe031 链接器命令文件，以了解如何为 RAM 和闪存配置完成该操作。

将示例程序移植到其他器件和 LaunchPad 的说明与上述说明类似。

## 6 连接电源线

德州仪器 (TI) C2000 ( 与 AFE031 器件搭配使用 ) 使用电力线通信 (PLC) 并通过现有电源线发送数据。这意味着只需通过与这些器件相连的电源线即可同时为数据传输供电和控制/检索数据。这可以最大限度地降低总成本；如果使用额外的电缆创建通信路径，则所需的总成本会增加。

### 6.1 线路耦合

线路耦合是 PLC 系统中最关键的部分之一，具有两个主要功能。第一个功能是将来自 AFE031 的信号与交流市电/直流总线进行耦合。第二个功能是防止市电的 50/60Hz 低频高电压对 PLC 电路造成损坏。

### 6.2 耦合到交流线路

为了耦合到交流电源，需要使用以下元件：低压电容器、变压器、高压电容器和电感器。需要注意的是，本节没有展示必要的保护电路；节 6.4 讨论了相关信息。

图 6-1 显示了简化的示意图。

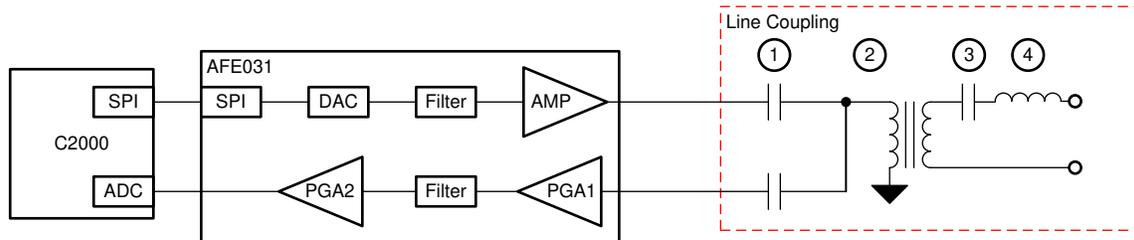


图 6-1. 交流电源的简化线路耦合电路

本节按照从左（连接到 AFE031 器件的 TX 和 RX 的元件）到右（连接到电源线的元件）的顺序介绍了这些元件。在硬件设计过程中务必考虑以下要点。

#### 6.2.1 低压电容器

低压电容器将 PA 输出信号的时变分量耦合到线路耦合变压器中。低压电容器应具有足够大的电容，以在整个所需的信号频带中显示为低阻抗。值 10μF 是 35kHz 至 150kHz 信号的常用值。BoosterPack 低压电容器值为 10μF，因为空间频率和标记频率（131.25kHz 和 141.75kHz）处于该范围内。低压电容器的额定电压应足以承受在浪涌条件下工作的 TVS 二极管（在保护部分中讨论）的钳位电压。通常，该限值应等于或略高于 PA 电源电压。

#### 6.2.2 变压器的匝数比

大多数电力线通信变压器都很紧凑，匝数比介于 1:1 和 4:1 之间，漏电感很低，绕组电感约为 1mH。这种与高压电容器串联的电感使分压器在模块输出端将交流电源电压衰减到可忽略不计的水平。

若要确定变压器的最佳匝数比，必须以 PA 的最大输出摆幅和最大输出电流能力为根据，从而向负载传输最大功率。

$$\frac{N_1}{N_2} = \sqrt{\frac{PA, V_{OUT\_Peak}}{PA, I_{OUT\_Peak} \times R_{Load}}} \quad (2)$$

在三种情况下，有一种限制相对于其他限制占据主导地位。

- **案例 1：**

如果变压器的匝数比大于理想的计算值，则 AFE031 的 TX 输出端受到 PA 电压摆幅的限制。

- **案例 2：**

如果变压器的匝数比小于理想的计算值，则 AFE031 的 TX 输出端受到 PA 最大输出电流的限制。

- **案例 3：**

如果变压器的匝数比等于理想的计算值，则 TX 最大输出发生在放大器接近其最大输出电压和最大输出电流时，从而能够向负载传输最大功率。

还需要注意的是，在 2MHz 频率下，变压器会影响 EN50065-1 的耦合发射性能。为了进行补偿，TI 建议使用 Würth Elektronik 的产品。

### 6.2.3 高压电容器

高压 (HV) 电容器通过与线路耦合变压器的绕组电感形成分压器来阻断低频市电电压。在使用高压 CBB 电容器时，最大电压范围必须超过电网交流电压的振幅。在大约 80% 的交流额定电压下运行电容器，可确保元件具有较长的使用寿命。下一项重要的要求是设置最大无功功率 (VA 限制) 的标准。例如，将器件连接到电网的欧洲产品标准必须具有小于 10VAR 的无功功率，从而要求电容器值小于 0.55μF。下面的公式显示了如何使用值 0.55μF 来确定高压电容器值。

$$HVCap = \frac{VA_{LIMIT}}{VAC^2 (2\pi \times f)} \quad (3)$$

对于具有 10VA 限制的 240VAC、50Hz 应用，

$$HVCap \leq \frac{10}{240^2 (2\pi \times 50)} \cong 550 \text{ nF} \quad (4)$$

需要注意的是，建议使用金属化聚丙烯电磁干扰和射频干扰 (EMI/RFI) 抑制电容器，因为与电介质相关的损耗因数较低，可以最大限度地减少内部自发热。

### 6.2.4 高压侧电感器

在驱动低线路阻抗时，需要使用一个与高压电容器串联的电感器，并且将该高压电容器的电容限制在 470nF 左右，原因如前所述。对于在 CENELEC A 频带中运行的应用，40kHz 下 470nF 电容的阻抗约为 8.5Ω。例如，如果应用需要具备驱动 2Ω 负载的能力，则该串联阻抗是有限制的。添加串联电感器可以减轻这种影响。若要正确选择电感值，必须知道系统的工作频率范围。一个常见的例子是 PRIME 频带，其范围大约为 40kHz 至 90kHz。建议选择高压电容器和电感器以使谐振频率处于频带的中心，这就要求串联电感器值为 12.8μH，高压电容器值为 470nF。

$$L = \frac{1}{HVCap \times (2\pi \times f)^2} \quad (5)$$

在确定电感器的大小时，应该使电感器能够在不达到饱和的情况下承受最大负载电流。

### 6.3 耦合到直流线路

耦合到直流线路需要类似的保护元件，但缺少高压电感器、MOV 和变压器等元件。图 6-2 显示了耦合电路和瞬态保护所需的元件。

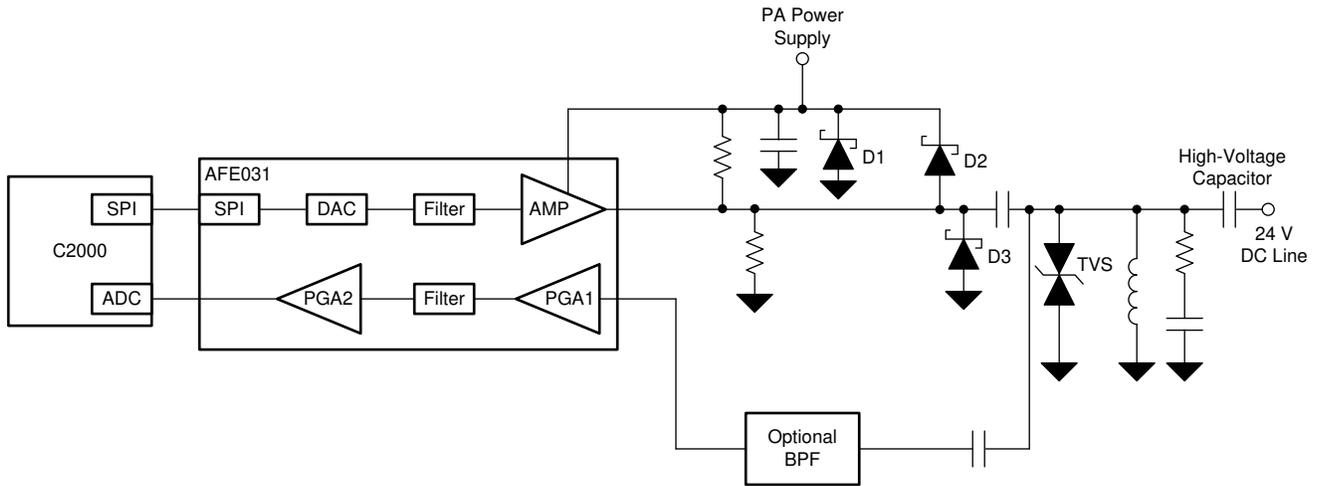


图 6-2. 直流线路的耦合电路

使用了一个  $10\mu\text{F}$  电容器来耦合到直流线路。在耦合到直流线路时，务必记住线路通常具有低阻抗，并且可能会影响 AFE031 中 PA 的输出摆幅。通过将电感器与线路以串联方式相连接，可以为电力线通信信号提供足够的阻抗，从而使电源（可能具有非常低的阻抗）不会干扰 PLC 信号调制。图 6-3 显示了耦合到低阻抗直流线路的 PLC 模块。

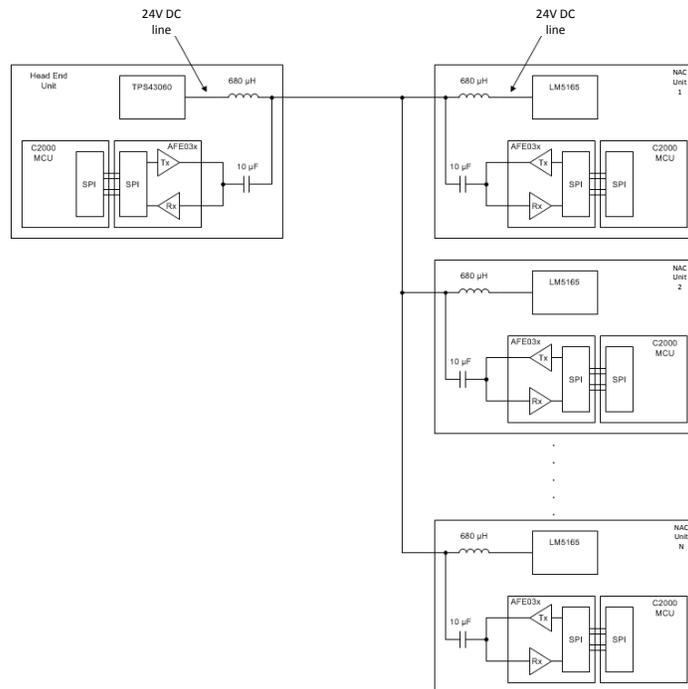


图 6-3. 多个 PLC 模块耦合到直流线路的示例

有关该电路的详细测试数据，请参阅[直流电力线通信参考设计](#)。

## 6.4 保护电路

电力线通信通常用于对连接到交流线路的电气元件而言非常恶劣的工作环境。如果没有得到适当的保护，电气异常（如闪电、电容器组切换、电感应开关或其他电网故障情况）产生的噪声或浪涌会损坏高性能集成电路。如果遵循以下几项建议，AFE031 甚至可以承受极恶劣的条件：金属氧化物压敏电阻 (MOV)、瞬态电压抑制二极管 (TVS)、肖特基二极管和齐纳二极管。

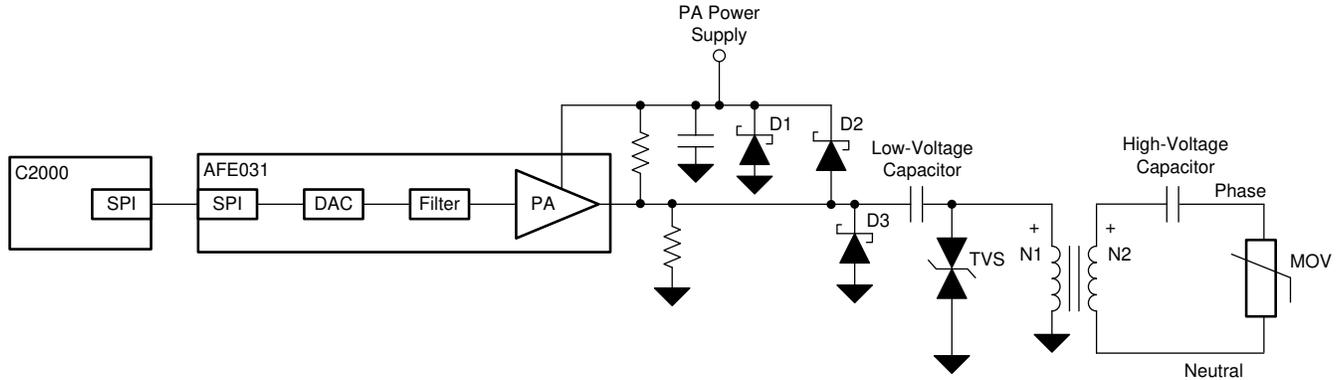


图 6-4. 建议的瞬态保护

### 6.4.1 金属氧化物压敏电阻

选择 MOV 时需要考虑几个因素：工作电压、需要被 MOV 吸收的瞬态能量、峰值瞬态电流和功率耗散。

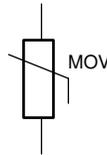


图 6-5. 金属氧化物压敏电阻 (MOV)

MOV 是一种在超过其触发电压之前具有高电阻的器件。一旦超过该电压电平，MOV 就会降低其电阻并吸收脉冲能量。图 6-6 显示了典型 MOV 的 I/V 特性。

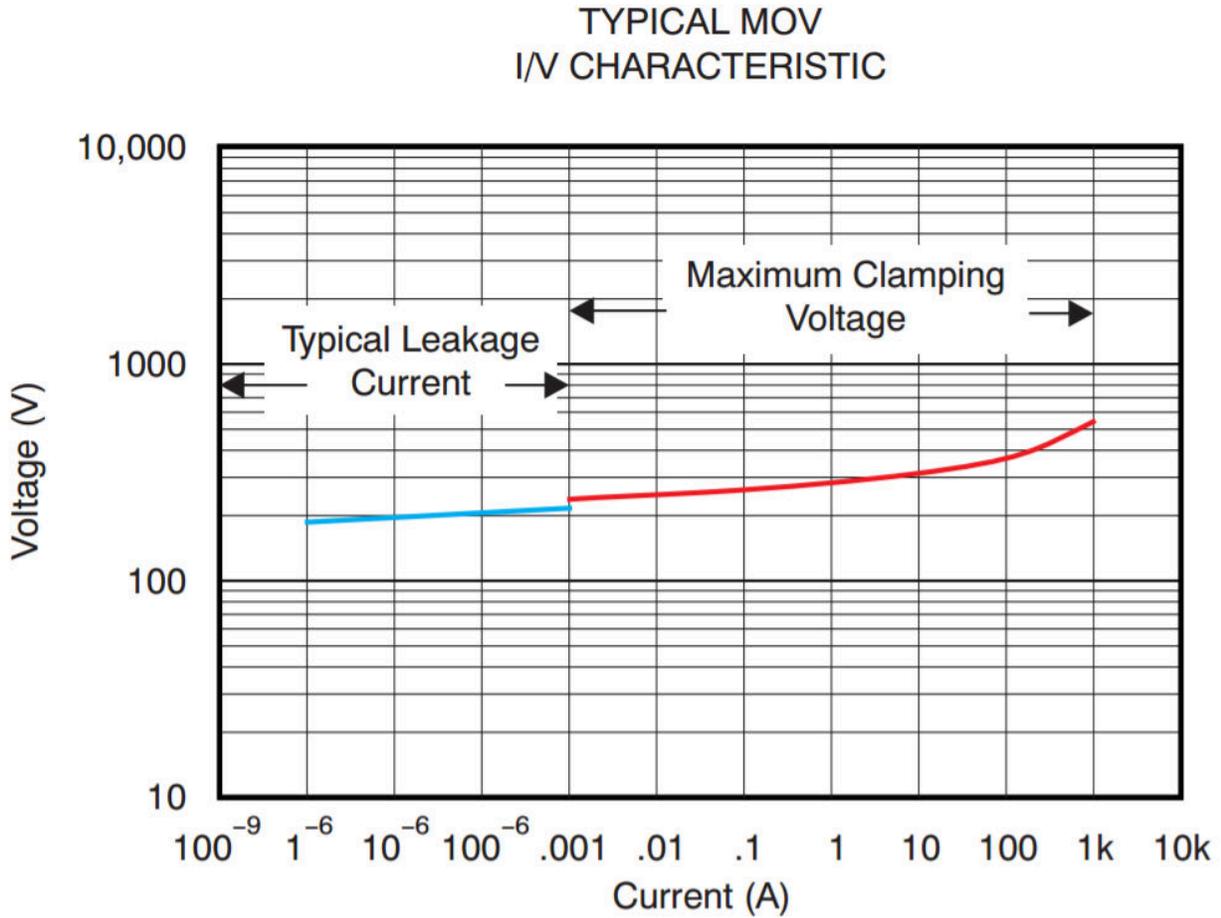


图 6-6. 典型 MOV I/V 特性

由于这些元件的构造中使用的材料和技术的性质，MOV 可以迅速响应快速瞬态脉冲，具有高瞬时额定功率，非常适合在交流线路中提供保护。最大钳位电压通常是响应类似于图 6-7 所示的高速瞬态而指定的。通常将 8/20 $\mu$ s 波形作为表示雷击频谱成分的波形进行关联。

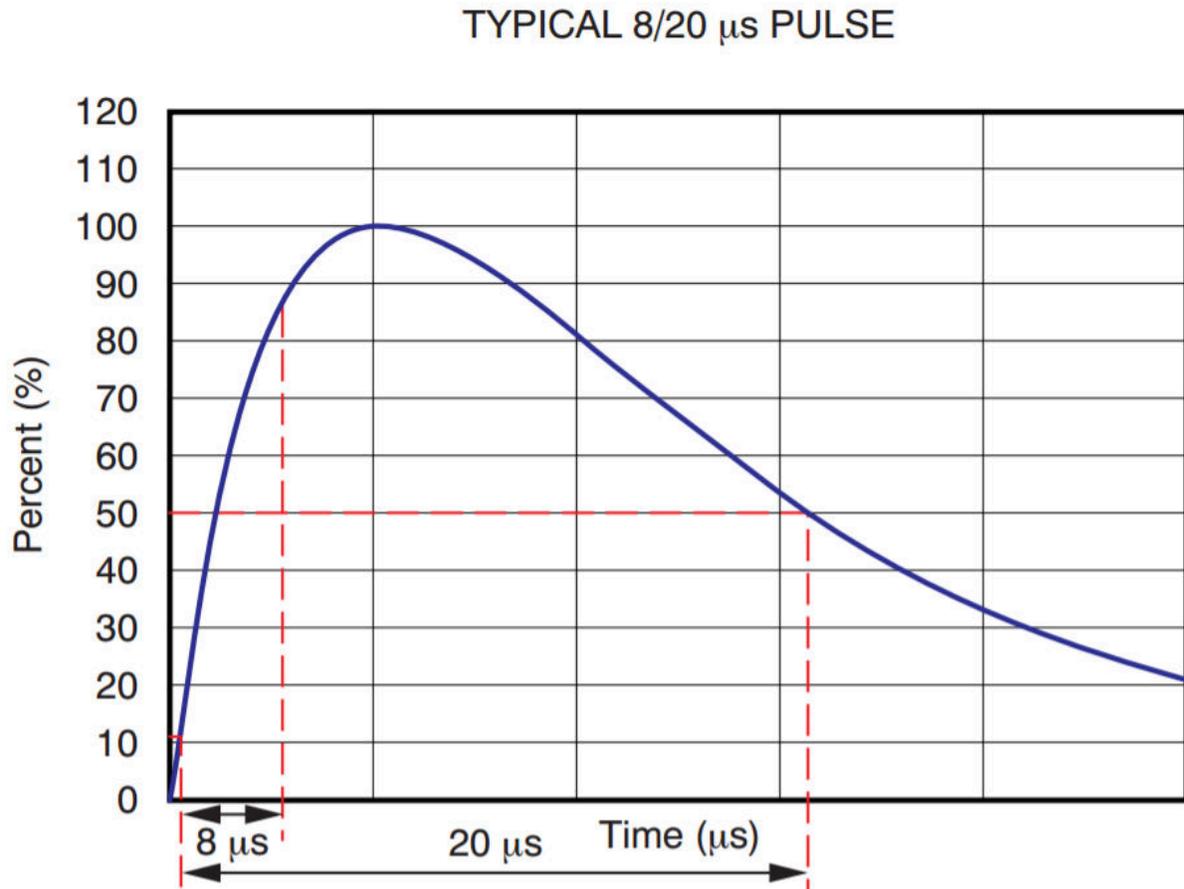


图 6-7. 满足 MOV 和 TVS 性能规格的典型 8/20 $\mu$ s 脉冲

#### 6.4.2 瞬态电压抑制器

TVS 是一种快速作用的钳位器件，在过压情况下导通，将浪涌电流分流到接地端。TVS 的主要性能指标为电源处理能力和钳位电压。TVS 提供单向或双向配置。

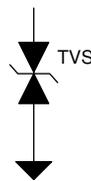


图 6-8. 双向 TVS 二极管

对于电力线通信应用，建议使用双向 TVS；该元件放置在线路耦合变压器旁边。图 6-9 显示了典型双向 TVS 的 I/V 特性。

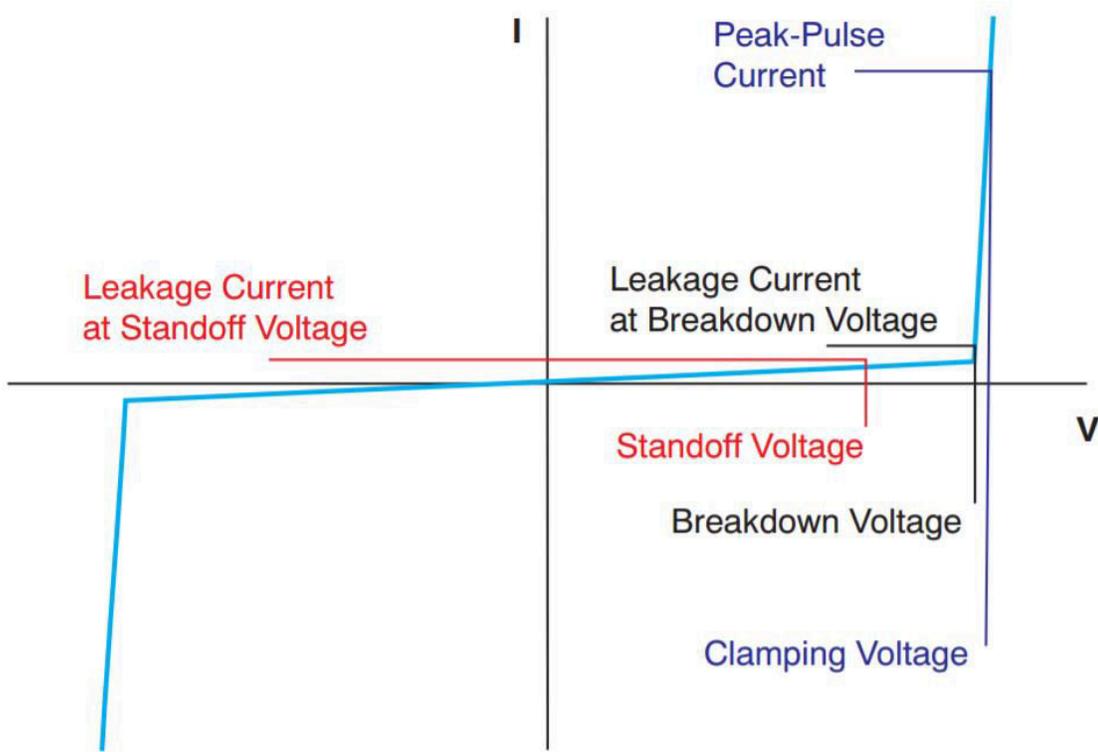


图 6-9. 典型的双向 TVS I/V 特性

当交流线路上出现浪涌或脉冲时，TVS 上的电压会上升。如果电压上升到高于 TVS 击穿电压，则 TVS 导通并迅速从高阻抗变为低阻抗，将电流分流到接地端。请注意，PA 输出和 TVS 之间的低压电容器会阻止 TVS 上的任何直流电压。因此，来自 PA 的正常 FSK 或 OFDM 信号似乎以 TVS 的接地端为中心。这种情况要求 TVS 是双向的。

图 6-10 阐明了这一概念。

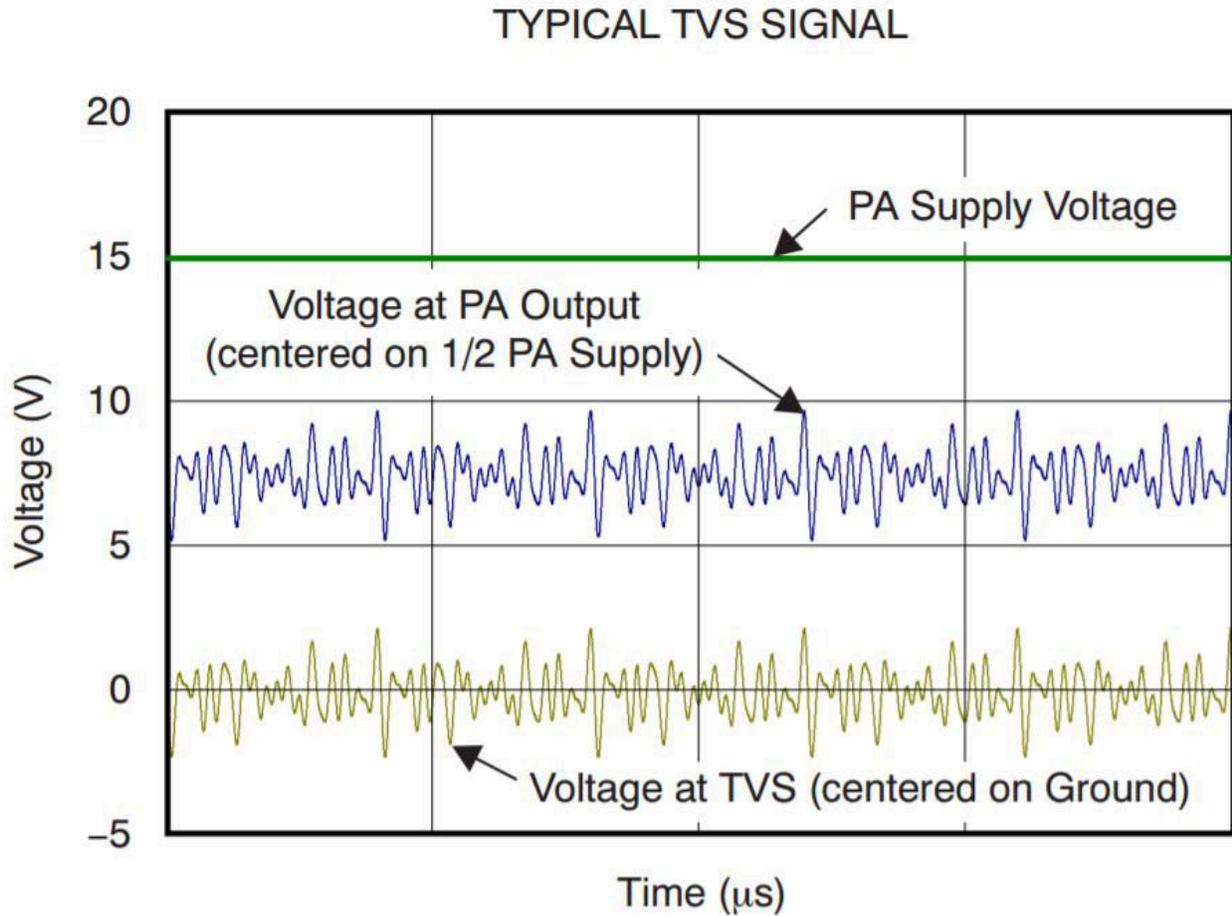


图 6-10. TVS 上的典型信号

信号在 TVS 处以接地端为中心对称，因此 TVS 击穿电压应约等于 PA 电源电压的二分之一。TVS 应在正常操作期间保持关闭，以避免削波和在输出信号中引入失真，这一点很重要。同样重要的是，TVS 应在非正常操作期间导通并钳制到尽可能低的电压，以提供最大保护。

该 BoosterPack 设计有浪涌保护器件。具有连接到变压器 LV 侧的分流接头的双向瞬态电压抑制 (TVS) 二极管用于钳制浪涌电压。TVS 的稳定电压必须恰好是 AFE 的 PA 电源电压的 1/2，这意味着 15V AFE 必须使用 7.5V TVS。该 1/2 比率基于以下事实：

- AFE 的 PA 为 AB 型，使用单个电源轨。因此，用 12V 电压供电的 PA TX 输出处于 6V 偏置，振幅为  $\pm 6V$ ，这意味着信号具有  $\pm 6V$  的范围。在传输过程中，TVS 不得使信号饱和，因此稳定电压必须一直高于 PA 电源轨的  $\frac{1}{2}$ 。
- 如果 12V PA 使用 7.5V TVS，则在浪涌脉冲到达期间，LV 侧偏置锁定在 7.5V，但如果浪涌恰好在 TX 达到最大振幅（即 6V）时发生，则 TX 路由上的信号为  $7.5V + 6V = 13.5V$ ，高于电源轨电压，从而对 AFE 的 PA 造成损坏。因此，TVS 电压不得高于 PA 电源轨的  $\frac{1}{2}$ 。

### 6.4.3 导流二极管

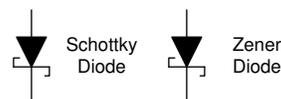


图 6-11. 导流二极管

尽管 MOV 和 TVS 元件可以钳制脉冲并将大部分能量消散或重新引导至接地端，但还是建议在 AFE031 的 PA 部分的输出端放置导电（肖特基）二极管。如果瞬态浪涌使 PA 输出引脚的电压超出其电源轨电压（不太可能发生），则低压降肖特基二极管可以将 AFE031 周围的电流安全地引导至接地端。建议在肖特基二极管上保持较低（小于 0.8V）的正向压降，以最大限度地提供保护。如果将 PA 的输出端连接到电源轨的肖特基二极管导通并变为正向偏置，则务必在不显著干扰 PA 电源电压的情况下将电流引导至接地端。通过在 PA 电源引脚上放置一个接地的齐纳二极管，可以为试图将电源电压提高到超过 AFE031 的绝对最大额定电压的浪涌提供低阻抗路径。

## 6.5 确定 PA 电源要求

在计算 PA 的最低电源要求时，必须知晓所需的负载电压、负载阻抗以及可用的电源电压或所需的变压器匝数这些参数。对于该 FSK 电力线通信示例，与 PRIME 类似，目标是将 1VRMS 信号驱动到一个 2Ω 负载中。所需的最小电源电压通过将峰-峰值负载电压、高压电容器和电感器上的压降 V<sub>2</sub>、低压电容器上的压降 V<sub>1</sub> 和相对于 PA 电源轨限值的两倍输出摆幅 V<sub>SWING</sub> 相加得出。对于 FSK 和 SFSK 系统，峰均比为 √2，而对于 OFDM 系统，该比率约为 3:1。

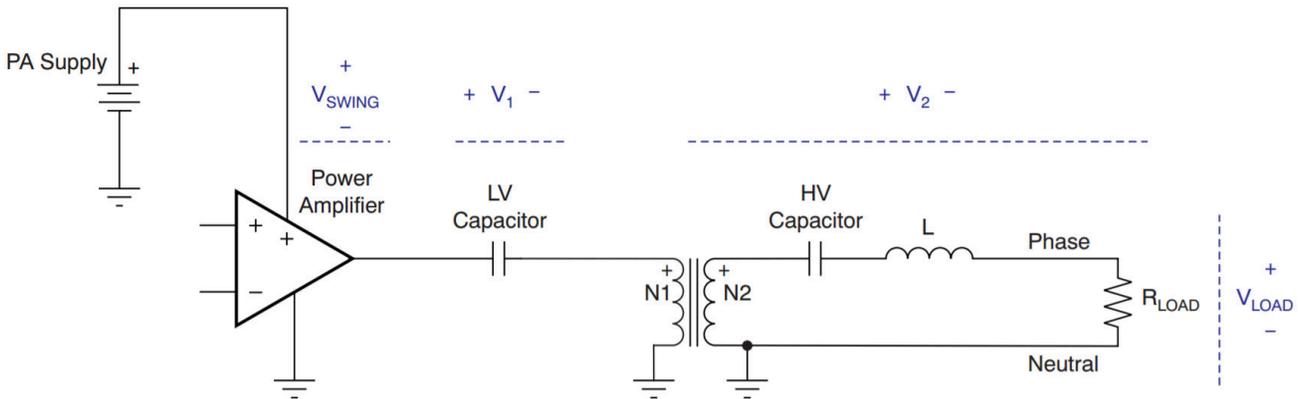


图 6-12. 典型的线路耦合电路

在分析期间执行与 RMS 电压和峰值电压相关的计算时，必须考虑这些比率。为低压电容器选择较大的值会导致压降 (V<sub>1</sub>)，在大多数情况下变得可以忽略不计。如果使用具有低 DCR 的合适变压器，即使在高负载电流下，变压器中的损耗也可以忽略不计。对于 FSK 和 SFSK 系统，高压电容器和电感器上的压降 V<sub>2</sub> 通常也可以忽略不计；在 OFDM 系统中，由于工作带宽更宽，压降 V<sub>2</sub> 可以忽略不计，并且可以用 1.5 乘以负载电压作为近似值来进行计算。

对于 AFE031 BoosterPack，具有 2Ω 负载和 1VRMS 负载电压的 FSK 信号：

$$PA_{Supply} = V_{Load} \times Turns\ Ratio \times (2 \times V_{Swing}) \quad (6)$$

$$PA_{Supply} = 2.878\ V \times 1.5 + (2 \times 2\ V) = 8.25\ V \quad (7)$$

对于 OFDM，可以使用以下公式：

$$PA_{Supply} = V_{Load} \times OFDM \times Multiplier \times Turns\ Ratio \times (2 \times V_{Swing}) \quad (8)$$

更多有关 AFE031 功耗的信息，请参阅 [采用 AFE031 且适用于窄带电力线通信调制解调器的模拟前端设计](#)。

## 7 总结

FSK 发送器为使用 AFE030/1 器件和 C2000 F28379D 进行通信开发提供了垫脚石，可以进行更多的开发来演示作为 FSK 接收器的系统。通过展示该示例，希望对 PLC 通信感兴趣的人能够掌握该示例并开始学习。通过创建足够灵活并可在多种模式下使用的硬件，可以在同一硬件上实现各种通信协议。最后提醒一下，可以在 [C2000Ware](#) 中找到所有软件和硬件。

## 8 参考文献

- SunSpec 协议：
  - [SunSpec 主页](#)
- AFE031 信息和设计：
  - [采用 AFE031 且适用于窄带电力线通信调制解调器的模拟前端设计](#)
  - [AFE031 电力线通信模拟前端数据表](#)
  - [直流电力线通信参考设计](#)
  - [LAUNCHXL-F28379D 概述](#)

### 9 原理图

#### 9.1 原理图 ( PWM 模式 )

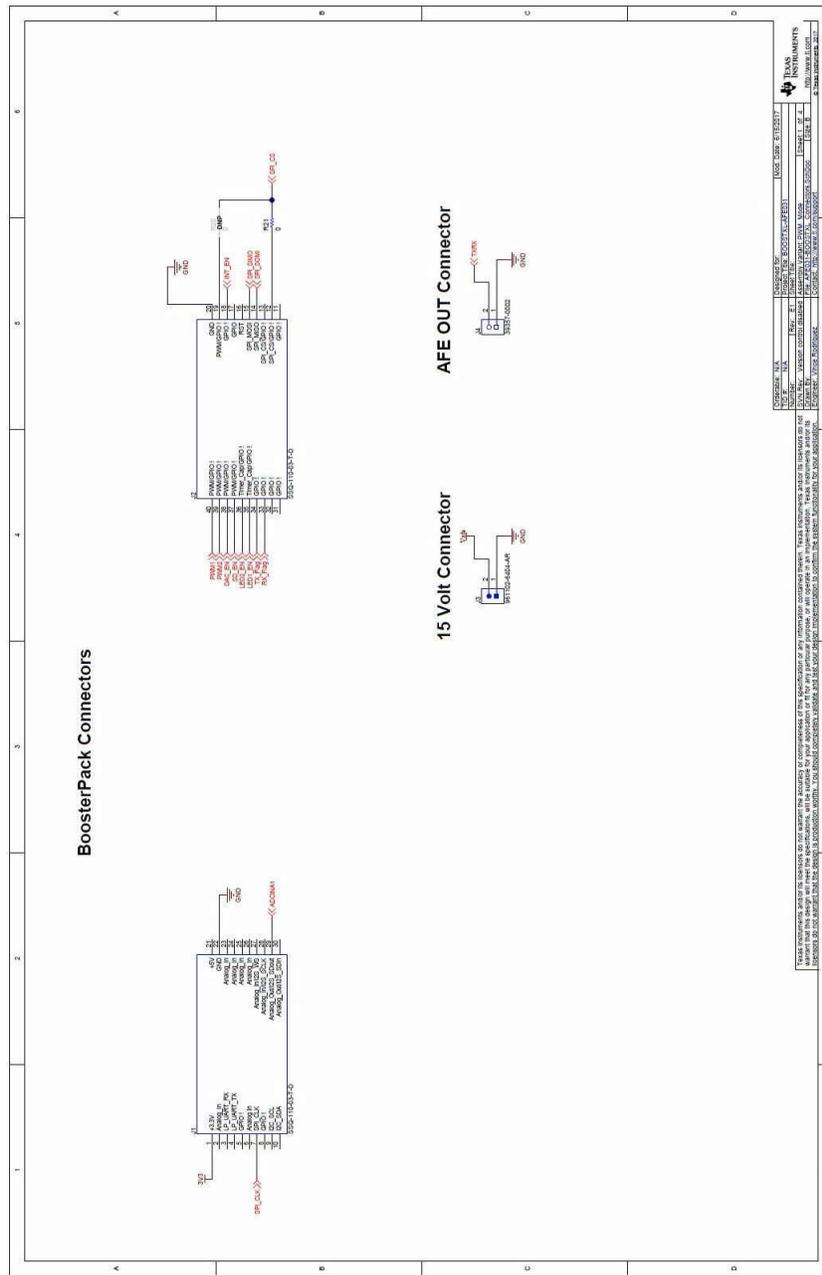


图 9-1. 原理图 ( PWM 模式 ) - 第 1 页





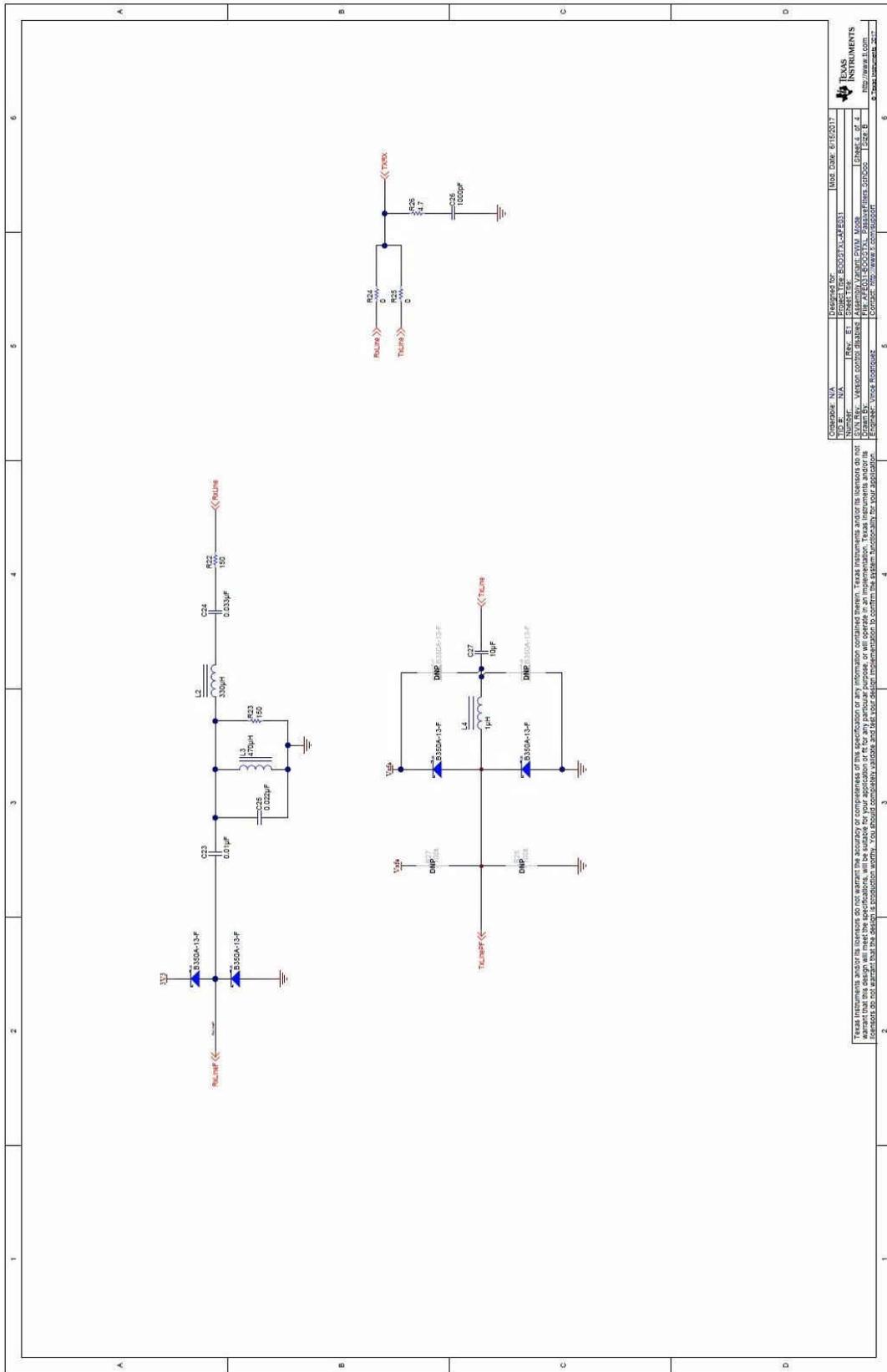


图 9-4. 原理图 ( PWM 模式 ) - 第 4 页

## 9.2 原理图 ( DAC 模式 )

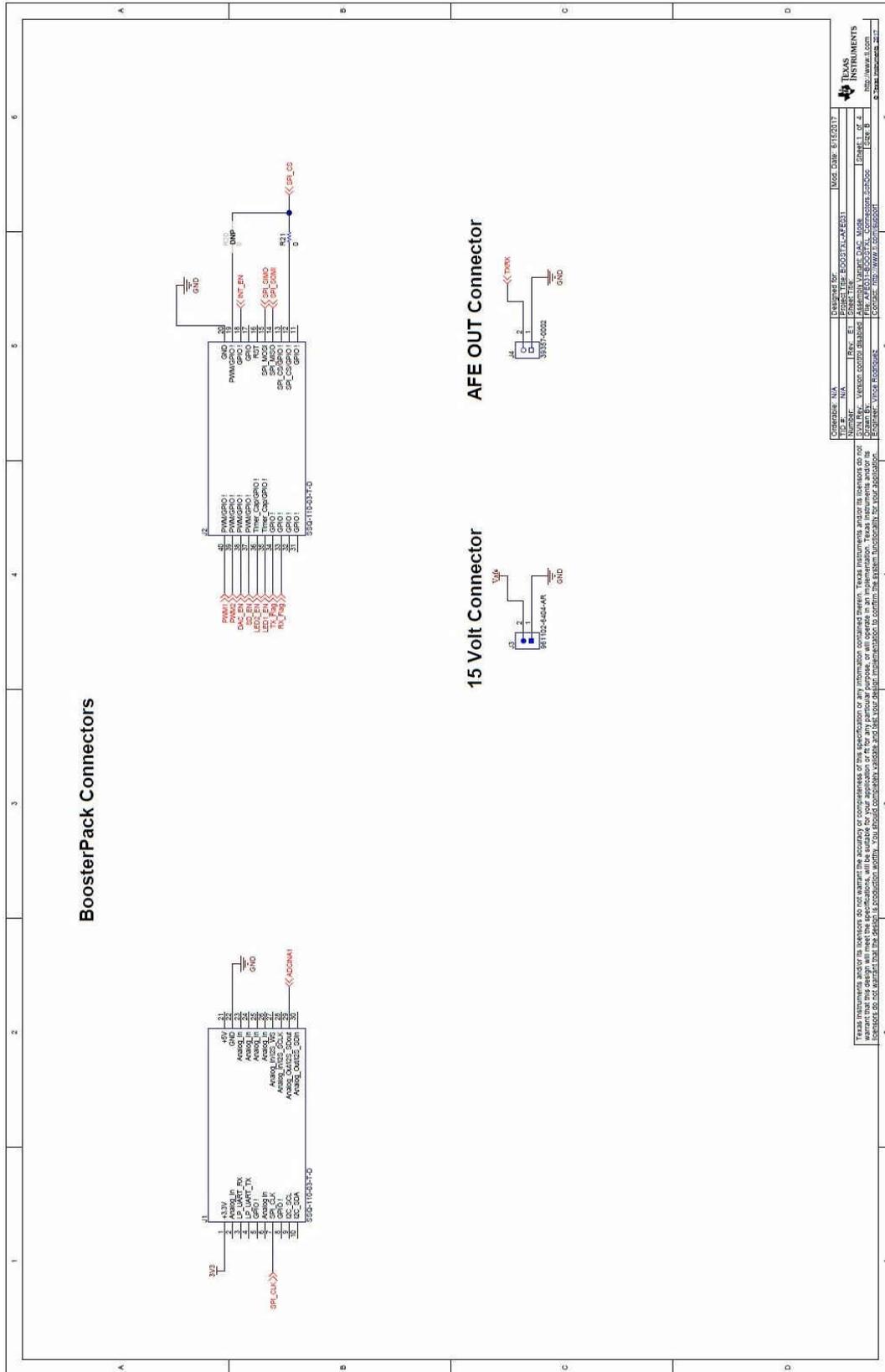


图 9-5. 原理图 ( DAC 模式 ) - 第 1 页

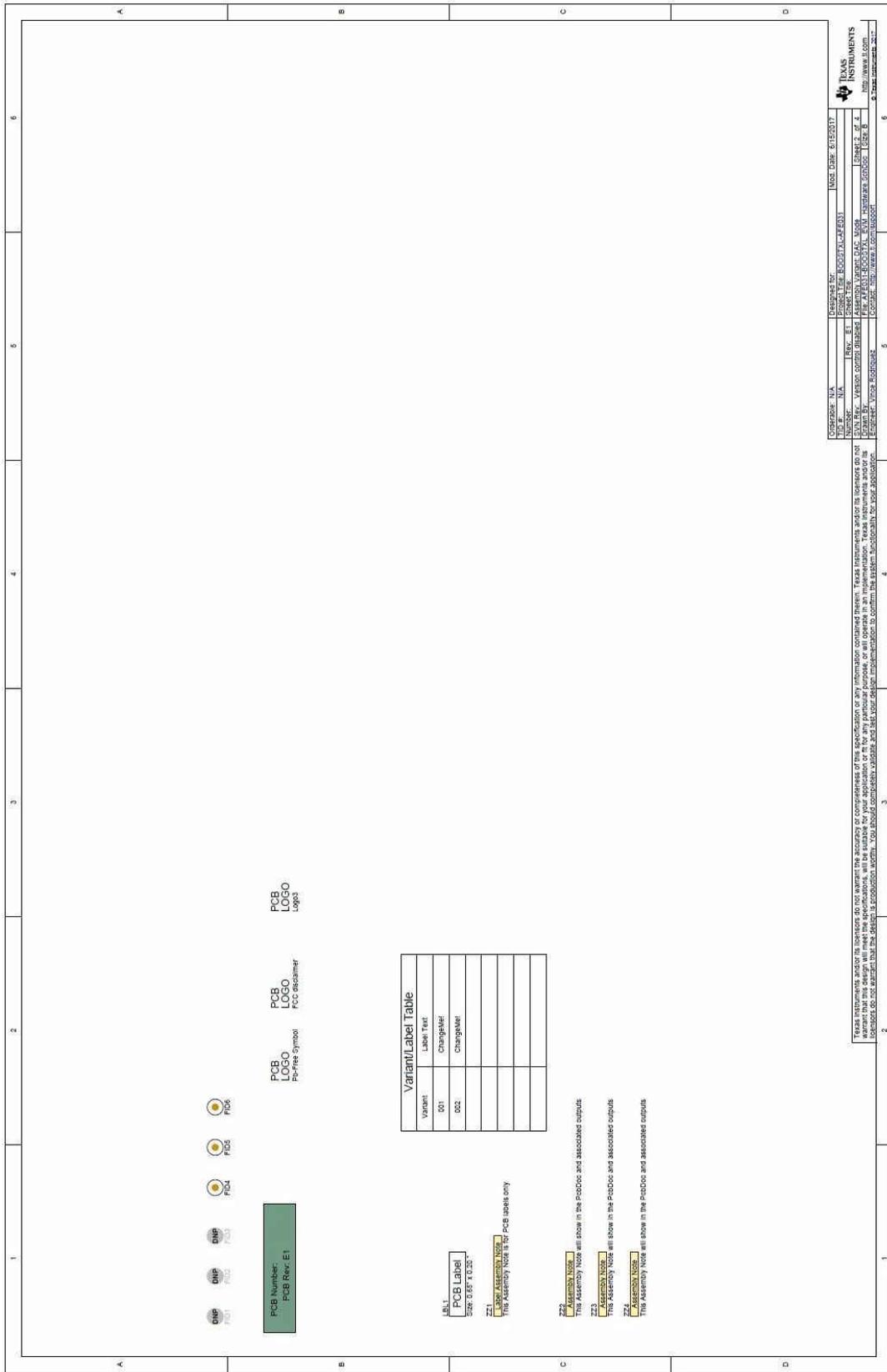


图 9-6. 原理图 ( DAC 模式 ) - 第 2 页



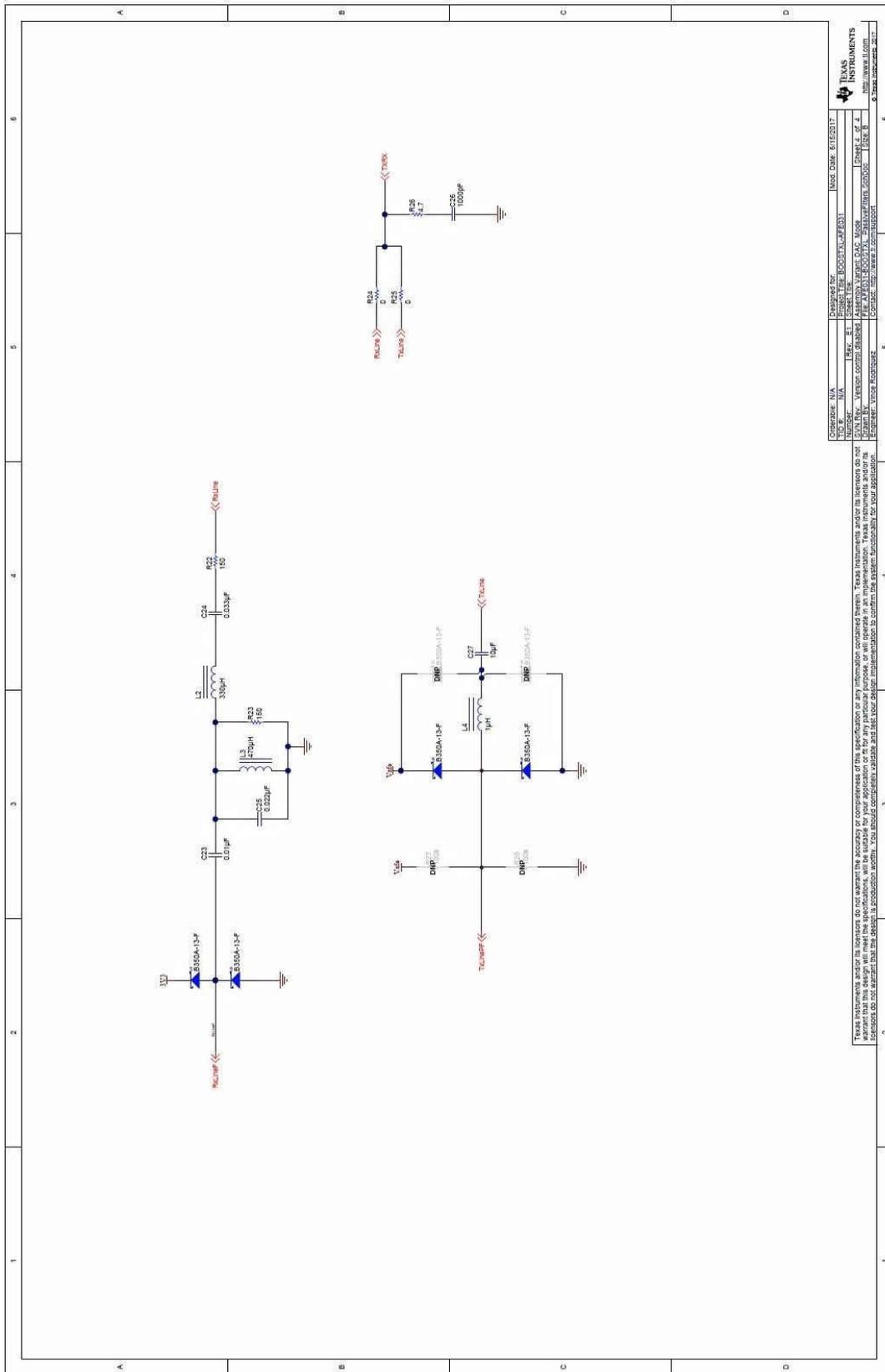


图 9-8. 原理图 ( DAC 模式 ) - 第 4 页

## 10 修订历史记录

注：以前版本的页码可能与当前版本的页码不同

<b>Changes from Revision C (September 2018) to Revision D (March 2022)</b>	<b>Page</b>
• 更新了整个出版物中的表格、图和交叉参考的编号格式.....	<b>1</b>
<b>Changes from Revision B (January 2018) to Revision C (September 2018)</b>	<b>Page</b>
• 对节 4.4 进行了更新.....	<b>20</b>
• 对节 5.4 进行了更新.....	<b>32</b>

## 重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 [TI 的销售条款](#) 或 [ti.com](#) 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

Copyright © 2022，德州仪器 (TI) 公司