User's Guide

AFE20408EVM Python 用户指南

摘要

本文档介绍了 AFE20408EVM 软件包含的 Python 代码，并展示了使用示例。

内容

表格清单

商标

1 简介

AFE20408EVM 附带一个安装在 EVM 上的 FT4222 模块。FT4222 是一款 USB 转 I2C/SPI 桥接器，用于在 EVM GUI 和 AFE20408 之间通信。可以利用 FT4222 创建自定义命令和代码脚本。

EVM 中包含基本 Python 代码，可作为用户操作的起点。本文档将说明每个文件的作用以及如何利用初始代码创建您自己的代码。

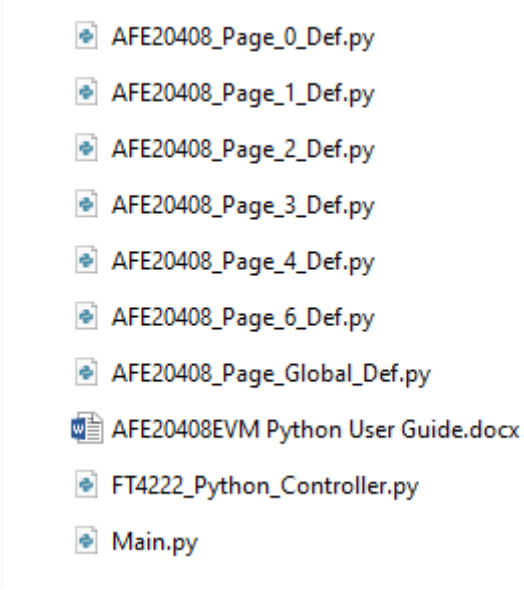

图 1-1. 包含的 Python 文件

Python 文件夹包含九个 Python 文件。后续章节会详细说明这些文件。

2 FT4222_Python_Controller.py

此文件包含与 FT4222 控制器相关的函数。

FtdiController()

初始化 FTDI。

getDeviceInfo()

获取 FTDI 器件信息。

closeHandle()

程序完成后，关闭 FTDI 句柄。最佳实践是在重新初始化 FTDI 之前先关闭 FTDI 句柄。

2.1 I2C 命令

initializeI2C（速度）

在 I2C 模式下初始化 FTDI。

表 2-1. InitializeI2C 参数

速度	可以在 60k 到 3400k 之间。请注意，AFE20408 只能处理最高 400kbps 的速度

i2cWrite (从机器件地址，寄存器地址，寄存器数据)

向 EVM 发送 I2C 写入命令。

表 2-2. I2CWrite 参数

从器件地址	8 位从器件地址。此地址可使用 EVM 跳线进行配置。
寄存器地址	8 位寄存器地址
寄存器数据	16 位寄存器数据

i2cRead (从机器件地址，寄存器地址)

向 EVM 发送 I2C 读取命令。返回一个 16 位值。

表 2-3. I2CRead 参数

从器件地址	8 位从器件地址。此地址可使用 EVM 跳线进行配置。
寄存器地址	8 位寄存器地址

示例：

```
ftdiObject = FtdiController()   # initialize the FTDI
ftdiObject.getFtdiDeviceInfo()  # Get FTDI info
ftdiObject.initializeI2C(400)   # initialize I2C with 400kbps

# Write to slave address 0x40, register address 0x08, with data 0x0001
ftdiObject.i2cWrite(0x40, 0x08, 0x0001)

# Read back from slave address 0x40, register address 0x08, and print the data.
print(ftdiObject.i2cRead(0x40, 0x08))

ftdiObject.closeHandle()    # close the FTDI when done.
```

2.2 SPI 命令

initializeSPI (模式，时钟，CPOL，CPHA，从器件选择)

在 SPI 模式下初始化 FTDI

表 2-4. InitializeSPI 参数

名称	参数	说明
模式	Ft4222.SPIMaster.Mode.NONE Ft4222.SPIMaster.Mode.SINGLE Ft4222.SPIMaster.Mode.DUAL Ft4222.SPIMaster.Mode.QUAD	AFE20408EVM 使用 *SINGLE* 模式。请勿使用其他模式。
时钟	ft4222.SPIMaster.Clock.NONE ft4222.SPIMaster.Clock.DIV_2 ft4222.SPIMaster.Clock.DIV_4 ft4222.SPIMaster.Clock.DIV_8 ft4222.SPIMaster.Clock.DIV_16 ft4222.SPIMaster.Clock.DIV_32 ft4222.SPIMaster.Clock.DIV_64 ft4222.SPIMaster.Clock.DIV_128 ft4222.SPIMaster.Clock.DIV_256 ft4222.SPIMaster.Clock.DIV_512	默认时钟值为 60MHz。AFE20408 最大时钟为 20MHz。
CPOL	ft4222.SPI.Cpol.IDLE_LOW ft4222.SPI.Cpol.IDLE_HIGH	AFE20408 使用 IDLE_LOW
CPHA	ft4222.SPI.Cpha.CLK_LEADING ft4222.SPI.Cpha.CLK_TRAILING	AFE20408 使用 CLK_TRAILING

表 2-4. InitializeSPI 参数（续）

名称	参数	说明
从器件选择	ft4222.SPIMaster.SlaveSelect.SS0 ft4222.SPIMaster.SlaveSelect.SS1 ft4222.SPIMaster.SlaveSelect.SS2 ft4222.SPIMaster.SlaveSelect.SS3	AFE20408EVM 使用 SS0

spiWrite（寄存器地址，寄存器数据）

向 EVM 发送 SPI 写入命令。

表 2-5. SPIWrite 参数

寄存器地址	8 位寄存器地址
寄存器数据	16 位寄存器数据

spiRead（寄存器地址）

向 EVM 发送 SPI 读取命令。返回一个 16 位值。

表 2-6. SPIRead 参数

寄存器地址	8 位寄存器地址

示例：

```
    ftdiObject = FtdiController()   # initialize the FTDI
    ftdiObject.getFtdiDeviceInfo()  # Get FTDI info
    # Initialize SPI. Note the only changeable variable is the clock division. Everything else is
static for the EVM
    ftdiObject.initializeSPI(ft4222.SPIMaster.Mode.SINGLE, ft4222.SPIMaster.Clock.DIV_4,
ft4222.SPI.Cpol.IDLE_LOW, ft4222.SPI.Cpha.CLK_TRAILING, ft4222.SPIMaster.SlaveSelect.SS0)

    # Write to register address 0x08, with data 0x0001
    ftdiObject.spiWrite(0x08, 0x0001)

    # Read back from register address 0x08 and print the data.
    print(ftdiObject.spiRead(0x08))

    ftdiObject.closeHandle()    # close the FTDI when done.
```

3 AFE20408_Page_Global_Def.py

AFE20408 的 Global 页面始终可供访问 – 无需更改 PAGE 寄存器即可访问任何全局寄存器。此文件包含变量格式的全局寄存器地址。此文件还包含特定数据组合的相关变量。

此页面包含以下寄存器：

表 3-1. 全局页面寄存器

寄存器	说明
NOP	软件复位 = 0x00AD
PAGE	PAGE 0 = GEN 配置 PAGE 1 = ADC 配置 PAGE 2 = ADC CCS 配置 PAGE 3 = DAC 配置 PAGE 4 = DAC 缓冲器 PAGE 6 = DAC 有效
GEN_STATUS	只读
ALARM_STATUS_0	只读
ALARM_STATUS_1	只读
PWR_STATUS_0	只读

表 3-1. 全局页面寄存器（续）

寄存器	说明
PWR_STATUS_1	只读
PWR_EN	DAC 使能配置
触发	ADC 和 DAC 触发
GPIO	
DRVEN_SW_EN	启用软件 DRVEN
DRVEN	软件 DRVEN 配置
DAC_BCAST	DAC 广播数据
GLOBAL_CFG	
ADC_SENSE0	SENSE0 回读
ADC_SENSE1	SENSE1 回读
ADC_ADC0	ADC0 回读
ADC_ADC1	ADC1 回读
ADC_TMP	局部温度回读

示例：

```
# Write to the DAC Power Enable address, enable all of the DACs

   # Example 1: Use the address variable, use raw data.
   ftdiObject.spiWrite(PWR_EN_ADDRESS, 0x01FF)

   # Example 2: Use the variables to set the data.
   ftdiObject.spiWrite(PWR_EN_ADDRESS, PWR_EN_PAON_En + PWR_EN_PDACB3_En + PWR_EN_PDACB2_En +
PWR_EN_PDACB1_En + PWR_EN_PDACB0_En + PWR_EN_PDACA3_En + PWR_EN_PDACA2_En + PWR_EN_PDACA1_En +
PWR_EN_PDACA0_En)
```

4 AFE20408_Page_0_Def.py

AFE20408 第 0 页包含基本配置寄存器。此文件包含寄存器和特定数据组合的相关变量。在对此页面上的任何寄存器进行写入或读取之前，您需要将 0x0000 写入 PAGE 寄存器。

此页面包含以下寄存器：

表 4-1. 页 0 寄存器

寄存器	说明
CHIP_ID	回读值为 0x2480
CHIP_VER	
SDO_EN	需要启用 SDO 才能进行 SPI 回读。
GEN_CFG_0	PAON 和 FLEXIO 配置
GEN_CFG_1	VSS 范围和 FLEXIO 选择
ALARMOUT_SRC_0	
ALARMOUT_SRC_1	
ALARM_STATUS_0_BYP	
ALARM_STATUS_1_BYP	
PAON_SRC_0	
PAON_SRC_1	
RESET_FLAGS	

示例：

```
#Example: Read chip ID
ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_GEN_CONFIG)  #Set page
ftdiObject.spiWrite(SDO_EN_ADDRESS, SDO_EN_FSDO_Dis + SDO_EN_SDOenabled)  #Enable SDO
print(ftdiObject.spiRead(CHIP_ID_ADDRESS))          #Read Chip ID
```

5 AFE20408_Page_1_Def.py

AFE20408 第 1 页包含 ADC 配置寄存器。此文件包含寄存器和特定数据组合的相关变量。在对此页面上的任何寄存器进行写入或读取之前，您需要将 0x0001 写入 PAGE 寄存器。

此页面包含以下寄存器：

表 5-1. 页 1 寄存器

寄存器	说明
ADC_GEN_CFG	包含误报配置、ADC 自动/直接模式和分流范围
ADC_CONV_CFG_0	包含转换率配置
ADC_CONV_CFG_1	包含样本大小配置
ADC_HYST_0	
ADC_HYST_1	
SENSE0_UP_THRESH	
SENSE0_LO_THRESH	
SENSE1_UP_THRESH	
SENSE1_LO_THRESH	
ADC0_UP_THRESH	
ADC0_LO_THRESH	
ADC1_UP_THRESH	
ADC1_LO_THRESH	
TMP_UP_THRESH	

示例：

```
    #Example: Configure ADC
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_ADC_CONFIG)  #Set ADC page

    #Set the following parameters:
        #ADC False Alarm = 16
        #Sense False Alarm = 32
        #Temp False Alarm = 4
        #ADC in automode
        #Shunt range 163.84mV
    ftdiObject.spiWrite(ADC_GEN_CFG_ADDRESS, ADC_GEN_CFG_FALR_ADC_16 + ADC_GEN_CFG_FALR_SENSE_32 +
ADC_GEN_CFG_FALR_TMP_4 + ADC_GEN_CFG_AUTOMODE_Automode + ADC_GEN_CFG_SHUNT_RANGE_163p84mVrange)
    #Set the following parameters in the ADC_CONV_CFG_0 register:
        #ADC Conversion rate = 152us
        #SENSE Conversion rate = 282us
        #Temperature conversion rate = 4122us
    ftdiObject.spiWrite(ADC_CONV_CFG_0_ADDRESS, ADC_CONV_CFG_0_CONV_RATE_ADC_152us +
ADC_CONV_CFG_0_CONV_RATE_SENSE_282us + ADC_CONV_CFG_0_CONV_RATE_TMP_4122us)
    #Set the following parameters in the ADC_CONV_CFG_1 register:
        #ADC Samples = 16
        #SENSE Samples = 4
        #Temperature Samples = 1
    ftdiObject.spiWrite(ADC_CONV_CFG_1_ADDRESS, ADC_CONV_CFG_1_AVG_ADC_16samples +
ADC_CONV_CFG_1_AVG_SENSE_4samples + ADC_CONV_CFG_1_AVG_TMP_1sample)
```

6 AFE20408_Page_2_Def.py

AFE20408 第 2 页包含 ADC 自定义通道时序控制寄存器。此文件包含寄存器和特定数据组合的相关变量。在对此页面上的任何寄存器进行写入或读取之前，您需要将 0x0002 写入 PAGE 寄存器。

此页面包含以下寄存器：

表 6-1. 页 2 寄存器

寄存器	说明
ADC_CCS_IDS_n	总共 63 个 CCS 寄存器。 每个寄存器包含 2 个索引，总共 126 个独立通道。 每个通道的选项包括： GND SENSE0 SENSE1 ADC0 ADC1 TMP
ADC_CCS_CFG_0	包含序列发生器的启动和停止索引

示例：

```
    #Example: Configure CCS. This will have the ADC iterate through only SENSE0 and ADC1.
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_ADC_CCS_CONFIG)  #Set ADC CCS page

    #Set the following parameters:
       #Index 0 = SENSE0
       #Index 1 = ADC1
    ftdiObject.spiWrite(ADC_CCS_IDS_0_ADDRESS, ADC_CCS_IDS_0_CCS_ID_0_SENSE0 +
ADC_CCS_IDS_0_CCS_ID_1_ADC1)
    ftdiObject.spiWrite(ADC_CCS_CFG_0_ADDRESS, 0x0001) #Start at index 0, end at index 1
```

7 AFE20408_Page_3_Def.py

AFE20408 第 3 页包含 DAC 配置寄存器。此文件包含寄存器和特定数据组合的相关变量。在对此页面上的任何寄存器进行写入或读取之前，您需要将 0x0003 写入 PAGE 寄存器。

此页面包含以下寄存器：

表 7-1. 页 3 寄存器

寄存器	说明
DAC_CURRENT	启动电流 = 15mA 低电流 = 30mA 正常电流 = 90mA 高电流 = 120mA
DAC_SYNC_CFG	同步和广播配置
DAC_CFG	输出引脚钳位配置
DAC_APD_EN	警报断电使能
DACA_APD_SRC_0	
DACA_APD_SRC_1	
OUTA_APD_SRC_0	
OUTA_APD_SRC_1	
DACB_APD_SRC_0	
DACB_APD_SRC_1	
OUTB_APD_SRC_0	
OUTB_APD_SRC_1	
DAC_CODE_LIMIT_0	

表 7-1. 页 3 寄存器（续）

寄存器	说明
DAC_CODE_LIMIT_1	
DAC_CODE_LIMIT_2	
DAC_CODE_LIMIT_3	
DRVEN0_EN	为特定 DAC 启用 DRVEN0
DRVEN1_EN	为特定 DAC 启用 DRVEN1
FLEXIO_EN	

示例：

```
    #Example: Configure DACs
     ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_DAC_CONFIG)  #Set DAC Config page

     #Set all of the DAC currents to High mode
     ftdiObject.spiWrite(DAC_CURRENT_ADDRESS, DAC_CURRENT_DACA0_CURRENT_Normal +
DAC_CURRENT_DACA1_CURRENT_Normal + DAC_CURRENT_DACA2_CURRENT_Normal +
DAC_CURRENT_DACA3_CURRENT_Normal + DAC_CURRENT_DACB0_CURRENT_Normal +
DAC_CURRENT_DACB1_CURRENT_Normal + DAC_CURRENT_DACB2_CURRENT_High + DAC_CURRENT_DACB3_CURRENT_High)
     #Set OUTA0 CLAMP to DACA1, OUTA2 CLAMP to VSSA, OUTB0 CLAMP to DACB1, and OUTB2 CLAMP to VSSB
     ftdiObject.spiWrite(DAC_CFG_ADDRESS, DAC_CFG_CLAMP_SEL_OUTA0_OUTA0clampvoltageisDACA1 +
DAC_CFG_CLAMP_SEL_OUTA2_OUTA2clampvoltageisVSSA + DAC_CFG_CLAMP_SEL_OUTB0_OUTB0clampvoltageisDACB1
+ DAC_CFG_CLAMP_SEL_OUTB2_OUTB2clampvoltageisVSSB)
     #Set OUTA0, DACA3, OUTB0, and DACB3 to toggle with DRVEN0
     ftdiObject.spiWrite(DRVEN0_EN_ADDRESS, DRVEN0_EN_DRVEN0_EN_DACA0_En +
DRVEN0_EN_DRVEN0_EN_DACA3_En + DRVEN0_EN_DRVEN0_EN_DACB0_En + DRVEN0_EN_DRVEN0_EN_DACB3_En)
     #Set OUTA2 and OUTB2 to toggle with DRVEN1
     ftdiObject.spiWrite(DRVEN1_EN_ADDRESS, DRVEN1_EN_DRVEN1_EN_DACA2_En +
DRVEN1_EN_DRVEN1_EN_DACB2_En)
```

8 AFE20408_Page_4_Def.py

AFE20408 第 4 页包含 DAC 缓冲器寄存器。此文件包含寄存器的相关变量。在对此页面上的任何寄存器进行写入或读取之前，您需要将 0x0004 写入 PAGE 寄存器。

此页面包含以下寄存器：

表 8-1. 页 4 寄存器

寄存器
DACA0
DACA1
DACA2
DACA3
DACB0
DACB1
DACB2
DACB3

示例：

```
    #Example: Set DAC Data
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_DAC_BUFFER)  #Set DAC Buffer page

    #Set DACA0 to midscale code 0x1000
    ftdiObject.spiWrite(DACA0_ADDRESS, 0x1000)
```

9 AFE20408_Page_6_Def.py

AFE20408 第 6 页包含有源 DAC 寄存器。此文件包含寄存器的相关变量。在从此页面上的任何寄存器读取之前，您需要将 0x0006 写入 PAGE 寄存器。这些寄存器是只读的。它们包含在 DAC 输出端看到的实际 DAC 值。

此页面包含以下寄存器：

表 9-1. 页 6 寄存器

寄存器
DACA0_ACTIVE
DACA1_ACTIVE
DACA2_ACTIVE
DACA3_ACTIVE
DACB0_ACTIVE
DACB1_ACTIVE
DACB2_ACTIVE
DACB3_ACTIVE

示例：

```
#Example: Read DAC Data
              ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_DAC_ACTIVE) #Set DAC Buffer page #Read the
              DACA0 data print(ftdiObject.spiRead(DACA0_ACTIVE_ADDRESS))
```

10 Main.py

Main.py 是一个示例文件，展示如何初始化 FTDI 并将基本命令写入 AFE20408。为方便起见，将 Main.py 中的代码粘贴到了此处。

```
#Import all helper files
from FT4222_Python_Controller import *
from AFE20408_Page_0_Def import *
from AFE20408_Page_1_Def import *
from AFE20408_Page_2_Def import *
from AFE20408_Page_3_Def import *
from AFE20408_Page_4_Def import *
from AFE20408_Page_6_Def import *
from AFE20408_Page_Global_Def import *

if __name__ == '__main__':
    ftdiObject = FtdiController()   #initialize FTDI
    ftdiObject.getFtdiDeviceInfo()  #Get FTDI Info

    #SPI Communication Initialisation
    ftdiObject.initializeSPI(ft4222.SPIMaster.Mode.SINGLE, ft4222.SPIMaster.Clock.DIV_2,
ft4222.SPI.Cpol.IDLE_LOW, ft4222.SPI.Cpha.CLK_TRAILING, ft4222.SPIMaster.SlaveSelect.SS0)

    """
    TO WRITE TO AFE20408:
    ftdiObject.spiWrite([REGISTER ADDRESS], [DATA])

    TO READ FROM AFE20408:
    ftdiObject.spiRead([REGISTER ADDRESS])
    """

    """Page 0 Setup - GENERAL CONFIGURATION"""
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_GEN_CONFIG) #Set page to Gen Config, page 0
    ftdiObject.spiWrite(SDO_EN_ADDRESS, SDO_EN_SDOenabled) #enable SDO for reading
    print(ftdiObject.spiRead(CHIP_ID_ADDRESS))   #confirm AFE20408 is communicating

    #Setup PAON pin = Push-pull, Active High.
    ftdiObject.spiWrite(GEN_CFG_0_ADDRESS, GEN_CFG_0_FLEXIO_OUT_ODE_PushPull +
GEN_CFG_0_PAON_POLARITY_ActiveHigh)
    #Setup VSS range. This sets the Mid Range. Sets the FLEXIO function as RESET.
    ftdiObject.spiWrite(GEN_CFG_1_ADDRESS, GEN_CFG_1_VSSA_RANGE_MidRange +
GEN_CFG_1_VSSB_RANGE_MidRange + GEN_CFG_1_FLEXIO_FUNC_RESET)
```

```
    #Set up the ALARMOUT alarms. This only works if the FLEXIO pin is set to ALARMOUT.
    ftdiObject.spiWrite(ALARMOUT_SRC_0_ADDRESS, 0x0000) #Set to default
    ftdiObject.spiWrite(ALARMOUT_SRC_1_ADDRESS, 0x1833) #Set to default
    #Set up the PAON alarms. This does not matter if you do not use the PAON pin.
    ftdiObject.spiWrite(PAON_SRC_0_ADDRESS, 0x0133) #Set to all sources.
    ftdiObject.spiWrite(PAON_SRC_1_ADDRESS, 0x1833) #Set to default

    """Page 1 Setup - ADC CONFIGURATION"""
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_ADC_CONFIG)  #Set the page to ADC CONFIG, page 1
    #Set the False Alarm threshold, ADC Auto/Direct mode, and Shunt range.
    #This will set the False Alarm thresholds to 4 samples, the ADC to Automode, and the Shunt
range to 163.84mV
    ftdiObject.spiWrite(ADC_GEN_CFG_ADDRESS, ADC_GEN_CFG_FALR_SENSE_4 + ADC_GEN_CFG_FALR_ADC_4 +
ADC_GEN_CFG_FALR_TMP_4 + ADC_GEN_CFG_AUTOMODE_Automode + ADC_GEN_CFG_SHUNT_RANGE_163p84mVrange)
    #Set the conversion rate. This will set all channels to 152us.
    ftdiObject.spiWrite(ADC_CONV_CFG_0_ADDRESS, ADC_CONV_CFG_0_CONV_RATE_ADC_152us +
ADC_CONV_CFG_0_CONV_RATE_SENSE_152us + ADC_CONV_CFG_0_CONV_RATE_TMP_152us)
    #Set the sampling rate. This will set all channels to 1 sample.
    ftdiObject.spiWrite(ADC_CONV_CFG_1_ADDRESS, ADC_CONV_CFG_1_AVG_ADC_1sample +
ADC_CONV_CFG_1_AVG_SENSE_1sample + ADC_CONV_CFG_1_AVG_TMP_1sample)
    #Set the ADC Hysteresis. This sets to default.
    ftdiObject.spiWrite(ADC_HYST_0_ADDRESS, ADC_HYST_0_DEFAULT)
    ftdiObject.spiWrite(ADC_HYST_1_ADDRESS, ADC_HYST_1_DEFAULT)
    #Set the ADC channel thresholds for the Alarm configurations. These set to default.
    ftdiObject.spiWrite(ADC0_LO_THRESH_ADDRESS, ADC0_LO_THRESH_DEFAULT)
    ftdiObject.spiWrite(ADC1_LO_THRESH_ADDRESS, ADC1_LO_THRESH_DEFAULT)
    ftdiObject.spiWrite(ADC0_UP_THRESH_ADDRESS, ADC0_UP_THRESH_DEFAULT)
    ftdiObject.spiWrite(ADC1_UP_THRESH_ADDRESS, ADC1_UP_THRESH_DEFAULT)
    ftdiObject.spiWrite(SENSE0_LO_THRESH_ADDRESS, SENSE0_LO_THRESH_DEFAULT)
    ftdiObject.spiWrite(SENSE0_UP_THRESH_ADDRESS, SENSE0_UP_THRESH_DEFAULT)
    ftdiObject.spiWrite(SENSE1_LO_THRESH_ADDRESS, SENSE1_LO_THRESH_DEFAULT)
    ftdiObject.spiWrite(SENSE1_UP_THRESH_ADDRESS, SENSE1_UP_THRESH_DEFAULT)
    ftdiObject.spiWrite(TMP_UP_THRESH_ADDRESS, TMP_UP_THRESH_DEFAULT)

    """Page 2 Setup - ADC CCS CONFIGURATION"""
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_ADC_CCS_CONFIG) #Set the page to ADC CCS CONFIG, Page 2
    #Example: Setup to read from SENSE0 after every other channel.
    ftdiObject.spiWrite(ADC_CCS_IDS_0_ADDRESS, ADC_CCS_IDS_0_CCS_ID_0_ADC0 +
ADC_CCS_IDS_0_CCS_ID_1_SENSE0)
    ftdiObject.spiWrite(ADC_CCS_IDS_1_ADDRESS, ADC_CCS_IDS_1_CCS_ID_2_ADC1 +
ADC_CCS_IDS_1_CCS_ID_3_SENSE0)
    ftdiObject.spiWrite(ADC_CCS_IDS_2_ADDRESS, ADC_CCS_IDS_2_CCS_ID_4_SENSE1 +
ADC_CCS_IDS_2_CCS_ID_5_SENSE0)
    ftdiObject.spiWrite(ADC_CCS_IDS_3_ADDRESS, ADC_CCS_IDS_3_CCS_ID_6_TMP +
ADC_CCS_IDS_3_CCS_ID_7_SENSE0)
    #Set the Start and Stop indices to 0 and 7
    ftdiObject.spiWrite(ADC_CCS_CFG_0_ADDRESS, 0x0007)

    """Page 3 Setup - DAC CONFIGURATION"""
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_DAC_CONFIG) #Set the page to DAC CONFIG, Page 3
    #Set the DAC current limits. This sets all the DACs to max current limit, 120mA
    ftdiObject.spiWrite(DAC_CURRENT_ADDRESS, 0xFFFF)
    #Set the OUT CLAMPs. This sets OUTA0 CLAMP = DACA1, OUTA2 CLAMP = DACA3, OUTB0 CLAMP = DACB1,
OUTB2 CLAMP = DACB3
    ftdiObject.spiWrite(DAC_CFG_ADDRESS, DAC_CFG_CLAMP_SEL_OUTA0_OUTA0clampvoltageisDACA1 +
DAC_CFG_CLAMP_SEL_OUTA2_OUTA2clampvoltageisDACA3 + DAC_CFG_CLAMP_SEL_OUTB0_OUTB0clampvoltageisDACB1
+ DAC_CFG_CLAMP_SEL_OUTB2_OUTB2clampvoltageisDACB3)
    #Disables DAC broadcast and Syncronous modes.
    ftdiObject.spiWrite(DAC_SYNC_CFG_ADDRESS, 0x0000)
    #Set the DAC Alarm Powerdown settings. This sets the register to default - All DACs power down
to VSS during an Alarm.
    ftdiObject.spiWrite(DAC_APD_EN_ADDRESS, DAC_APD_EN_DEFAULT)
    #Configure what Alarms trigger each DACs. This enables all alarms)
    ftdiObject.spiWrite(DACA_APD_SRC_0_ADDRESS, 0x0133)
    ftdiObject.spiWrite(DACA_APD_SRC_1_ADDRESS, DACA_APD_SRC_1_DEFAULT)
    ftdiObject.spiWrite(DACB_APD_SRC_0_ADDRESS, 0x0133)
    ftdiObject.spiWrite(DACB_APD_SRC_1_ADDRESS, DACB_APD_SRC_1_DEFAULT)
    ftdiObject.spiWrite(OUTA_APD_SRC_0_ADDRESS, 0x0133)
    ftdiObject.spiWrite(OUTA_APD_SRC_1_ADDRESS, OUTA_APD_SRC_1_DEFAULT)
    ftdiObject.spiWrite(OUTB_APD_SRC_0_ADDRESS, 0x0133)
    ftdiObject.spiWrite(OUTB_APD_SRC_1_ADDRESS, OUTB_APD_SRC_1_DEFAULT)
    #Set the DRVEN pin configuration. This sets the OUT pins to toggle with DRVEN0, and disables
DRVEN1.
    ftdiObject.spiWrite(DRVEN0_EN_ADDRESS, DRVEN0_EN_DRVEN0_EN_DACA0_En +
DRVEN0_EN_DRVEN0_EN_DACA2_En + DRVEN0_EN_DRVEN0_EN_DACB0_En + DRVEN0_EN_DRVEN0_EN_DACB2_En)
    ftdiObject.spiWrite(DRVEN1_EN_ADDRESS, 0x0000)

    """Setup Global Page"""
```

```
    #Disable Software DRVEN for the OUT pins
    ftdiObject.spiWrite(DRVEN_SW_EN_ADDRESS, 0x00AA)
    #Enable Software DRVEN on the odd DACs. REQUIRED for enabling DACA1/3, DACB1/3
    ftdiObject.spiWrite(DRVEN_ADDRESS, 0x00AA)
    #Enable all DACs
    ftdiObject.spiWrite(PWR_EN_ADDRESS, 0x00FF)

    """Set DAC voltage"""
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_DAC_BUFFER) #Set DAC Buffer page, Page 4
    #DAC voltage (Positive) = (DAC_CODE * 10)/2^13
    #DAC Voltage (Negative) = (DAC_CODE * 10)/2^13 - 10
    ftdiObject.spiWrite(DACA0_ADDRESS, 0x1570) # -3.3V or 6.7V depending on output range.
    ftdiObject.spiWrite(DACB0_ADDRESS, 0x0A8F) # 3.3V or -6.7V depending on output range.

    #Start ADC
    ftdiObject.spiWrite(TRIGGER_ADDRESS, TRIGGER_ADC_TRIG_EnableADC)

    print("SENSE0: " + ftdiObject.spiRead(0x18)) # ADC_SENSE0 channel readback
    print("SENSE1: " + ftdiObject.spiRead(0x19)) # ADC_SENSE1 channel readback
    print("ADC0: " + ftdiObject.spiRead(0x1A)) # ADC_ADC0 channel readback
    print("ADC1: " + ftdiObject.spiRead(0x1B)) # ADC_ADC1 channel readback
    print("TEMP: " + ftdiObject.spiRead(0x1C)) # ADC_TEMP channel readback

    #close the FTDI when done.
    ftdiObject.closeHandle()
```

11 通用序列

11.1 复位序列

软件将器件中的所有寄存器复位到其默认状态。

```
    #RESET SEQUENCE
    ftdiObject.spiWrite(NOP_ADDRESS, NOP_SW_RST_SoftwareReset)
```

11.2 ADC 配置

可通过许多方式配置 ADC。示例如下：

```
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_ADC_CONFIG)  #Set the page to ADC CONFIG, page 1
    #Set the False Alarm threshold, ADC Auto/Direct mode, and Shunt range.
    #This will set the False Alarm thresholds to 4 samples, the ADC to Automode, and the Shunt
range to 163.84mV
    ftdiObject.spiWrite(ADC_GEN_CFG_ADDRESS, ADC_GEN_CFG_FALR_SENSE_4 + ADC_GEN_CFG_FALR_ADC_4 +
ADC_GEN_CFG_FALR_TMP_4 + ADC_GEN_CFG_AUTOMODE_Automode + ADC_GEN_CFG_SHUNT_RANGE_163p84mVrange)
    #Set the conversion rate. This will set all channels to 152us.
    ftdiObject.spiWrite(ADC_CONV_CFG_0_ADDRESS, ADC_CONV_CFG_0_CONV_RATE_ADC_152us +
ADC_CONV_CFG_0_CONV_RATE_SENSE_152us + ADC_CONV_CFG_0_CONV_RATE_TMP_152us)
    #Set the sampling rate. This will set all channels to 1 sample.
    ftdiObject.spiWrite(ADC_CONV_CFG_1_ADDRESS, ADC_CONV_CFG_1_AVG_ADC_1sample +
ADC_CONV_CFG_1_AVG_SENSE_1sample + ADC_CONV_CFG_1_AVG_TMP_1sample)
    #Set the ADC Hysteresis. This sets to default.
    ftdiObject.spiWrite(ADC_HYST_0_ADDRESS, ADC_HYST_0_DEFAULT)
    ftdiObject.spiWrite(ADC_HYST_1_ADDRESS, ADC_HYST_1_DEFAULT)
    #Set the ADC channel thresholds for the Alarm configurations. These set to default.
    ftdiObject.spiWrite(ADC0_LO_THRESH_ADDRESS, ADC0_LO_THRESH_DEFAULT)
    ftdiObject.spiWrite(ADC1_LO_THRESH_ADDRESS, ADC1_LO_THRESH_DEFAULT)
    ftdiObject.spiWrite(ADC0_UP_THRESH_ADDRESS, ADC0_UP_THRESH_DEFAULT)
    ftdiObject.spiWrite(ADC1_UP_THRESH_ADDRESS, ADC1_UP_THRESH_DEFAULT)
    ftdiObject.spiWrite(SENSE0_LO_THRESH_ADDRESS, SENSE0_LO_THRESH_DEFAULT)
    ftdiObject.spiWrite(SENSE0_UP_THRESH_ADDRESS, SENSE0_UP_THRESH_DEFAULT)
    ftdiObject.spiWrite(SENSE1_LO_THRESH_ADDRESS, SENSE1_LO_THRESH_DEFAULT)
    ftdiObject.spiWrite(SENSE1_UP_THRESH_ADDRESS, SENSE1_UP_THRESH_DEFAULT)
ftdiObject.spiWrite(TMP_UP_THRESH_ADDRESS, TMP_UP_THRESH_DEFAULT)

    #Start ADC. TRIGGER then read the ADC outputs.
    ftdiObject.spiWrite(TRIGGER_ADDRESS, TRIGGER_ADC_TRIG_EnableADC)

    print("SENSE0: " + ftdiObject.spiRead(0x18)) # ADC_SENSE0 channel readback
    print("SENSE1: " + ftdiObject.spiRead(0x19)) # ADC_SENSE1 channel readback
    print("ADC0: " + ftdiObject.spiRead(0x1A)) # ADC_ADC0 channel readback
    print("ADC1: " + ftdiObject.spiRead(0x1B)) # ADC_ADC1 channel readback
    print("TEMP: " + ftdiObject.spiRead(0x1C)) # ADC_TEMP channel readback
```

11.3 DAC 配置

可通过许多方式配置 DAC。示例如下：

```
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_DAC_CONFIG) #Set the page to DAC CONFIG, Page 3
    #Set the DAC current limits. This sets all the DACs to max current limit, 120mA
    ftdiObject.spiWrite(DAC_CURRENT_ADDRESS, 0xFFFF)
    #Set the OUT CLAMPs. This sets OUTA0 CLAMP = DACA1, OUTA2 CLAMP = DACA3, OUTB0 CLAMP = DACB1,
OUTB2 CLAMP = DACB3
    ftdiObject.spiWrite(DAC_CFG_ADDRESS, DAC_CFG_CLAMP_SEL_OUTA0_OUTA0clampvoltageisDACA1 +
DAC_CFG_CLAMP_SEL_OUTA2_OUTA2clampvoltageisDACA3 + DAC_CFG_CLAMP_SEL_OUTB0_OUTB0clampvoltageisDACB1
+ DAC_CFG_CLAMP_SEL_OUTB2_OUTB2clampvoltageisDACB3)
    #Disables DAC broadcast and Syncronous modes.
    ftdiObject.spiWrite(DAC_SYNC_CFG_ADDRESS, 0x0000)
    #Set the DAC Alarm Powerdown settings. This sets the register to default - All DACs power down
to VSS during an Alarm.
    ftdiObject.spiWrite(DAC_APD_EN_ADDRESS, DAC_APD_EN_DEFAULT)
    #Configure what Alarms trigger each DACs. This enables all alarms)
    ftdiObject.spiWrite(DACA_APD_SRC_0_ADDRESS, 0x0133)
    ftdiObject.spiWrite(DACA_APD_SRC_1_ADDRESS, DACA_APD_SRC_1_DEFAULT)
    ftdiObject.spiWrite(DACB_APD_SRC_0_ADDRESS, 0x0133)
    ftdiObject.spiWrite(DACB_APD_SRC_1_ADDRESS, DACB_APD_SRC_1_DEFAULT)
    ftdiObject.spiWrite(OUTA_APD_SRC_0_ADDRESS, 0x0133)
    ftdiObject.spiWrite(OUTA_APD_SRC_1_ADDRESS, OUTA_APD_SRC_1_DEFAULT)
    ftdiObject.spiWrite(OUTB_APD_SRC_0_ADDRESS, 0x0133)
    ftdiObject.spiWrite(OUTB_APD_SRC_1_ADDRESS, OUTB_APD_SRC_1_DEFAULT)
    #Set the DRVEN pin configuration. This sets the OUT pins to toggle with DRVEN0, and disables
DRVEN1.
    ftdiObject.spiWrite(DRVEN0_EN_ADDRESS, DRVEN0_EN_DRVEN0_EN_DACA0_En +
DRVEN0_EN_DRVEN0_EN_DACA2_En + DRVEN0_EN_DRVEN0_EN_DACB0_En + DRVEN0_EN_DRVEN0_EN_DACB2_En)
ftdiObject.spiWrite(DRVEN1_EN_ADDRESS, 0x0000)

    """Set DAC voltage"""
    ftdiObject.spiWrite(PAGE_ADDRESS, PAGE_DAC_BUFFER) #Set DAC Buffer page, Page 4
    #DAC voltage (Positive) = (DAC_CODE * 10)/2^13
    #DAC Voltage (Negative) = (DAC_CODE * 10)/2^13 - 10
    ftdiObject.spiWrite(DACA0_ADDRESS, 0x1570) # -3.3V or 6.7V depending on output range.
    ftdiObject.spiWrite(DACB0_ADDRESS, 0x0A8F) # 3.3V or -6.7V depending on output range.
```

12 常见问题

ERROR: "No module named 'ft4222'"

按照本指南将 FT4222 驱动程序导入您的 PC。在 VS Code 中导入 Python 库 | 作者：Gbemisola Adekoya | 中级

在终端上使用“pip install ft4222”。

13 总结

这些软件文件允许用户为 EVM 软件无法支持的更高级的应用程序编写自定义脚本。

重要通知和免责声明

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出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 产品的相关应用。 严禁以其他方式对这些资源进行复制或展示。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