LP87561 Exercise Write-up

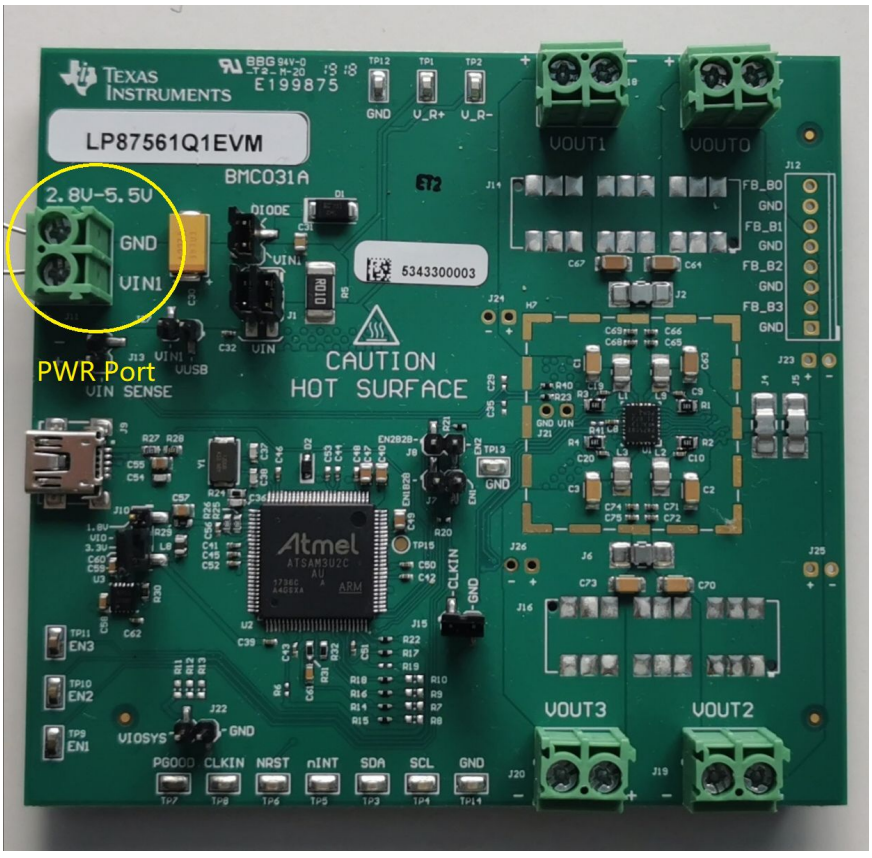
Goal:

The purpose of this exercise is to use a Raspberry Pi to control the output of an LP87561Q1EVM using I2C at the same time detect and capture the output with an oscilloscope (I personally used a Rigol DS1102E) also controlled by the same Raspberry Pi through USBTMC.

Procedure:

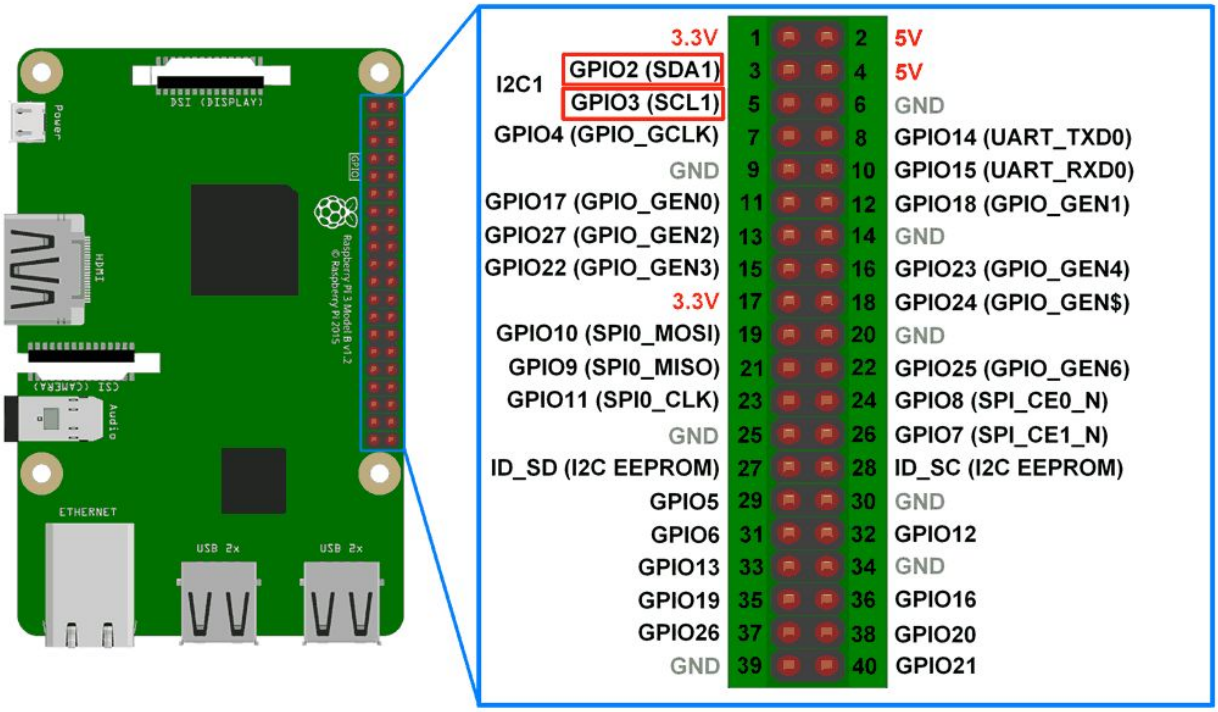
Wiring:

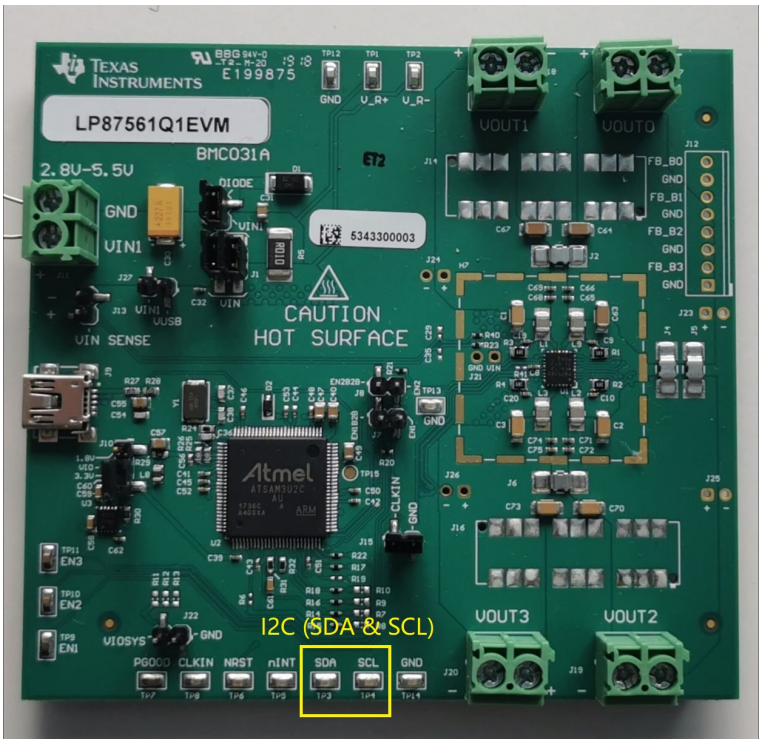
- Powering up the devices
  For the EVM, you need a 2.8V-5.5V power source. It is recommended to be powered by a 5V source. You then just have to connect to positive and negative terminals of the supply to the respective spot on the power port of the EVM



- Connect I2C

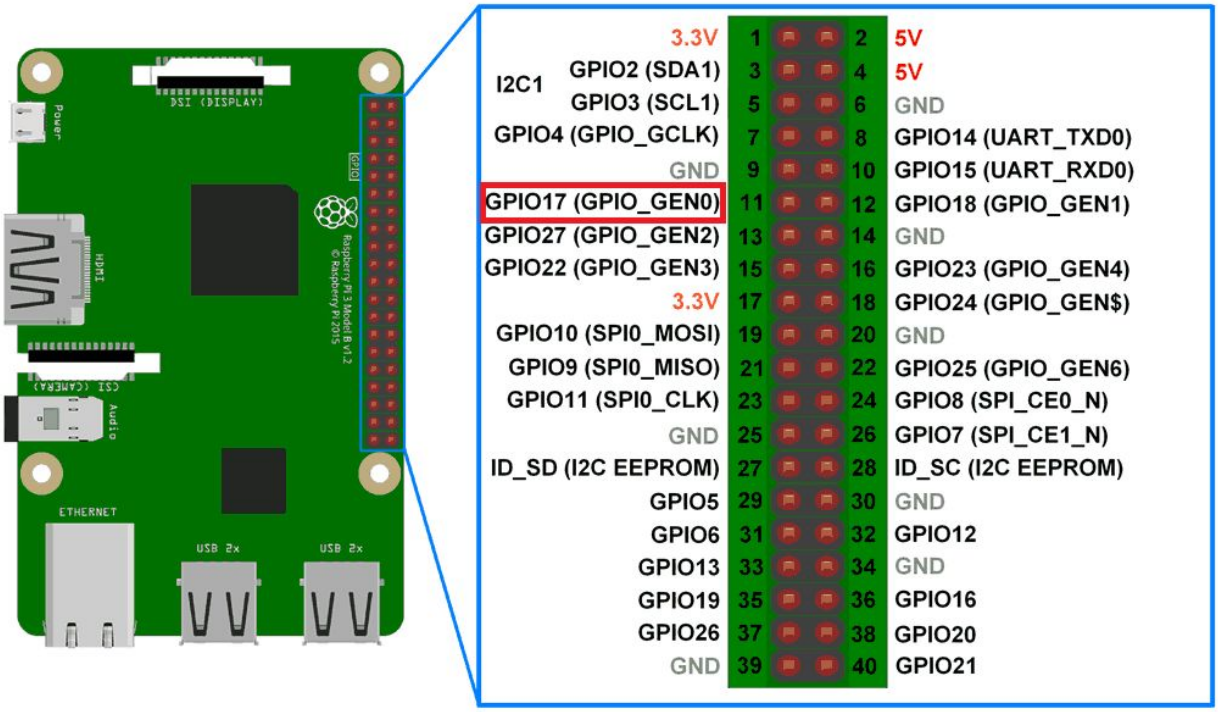
  To connect I2C, you have to locate the SCL and SDA pin on both the EVM and Raspberry Pi. Then connect SDA with SDA, SCL with SCL and connect the GND of EVM with the GND of Raspberry Pi.

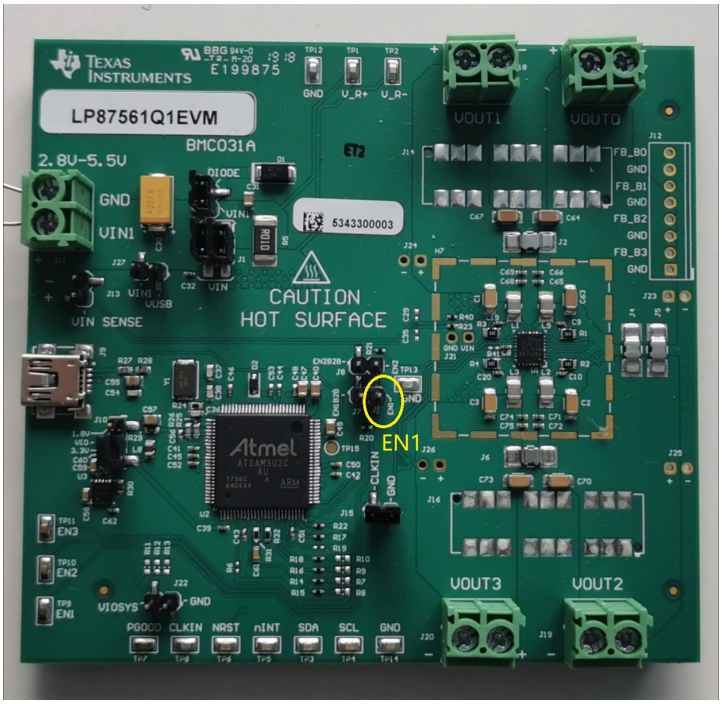




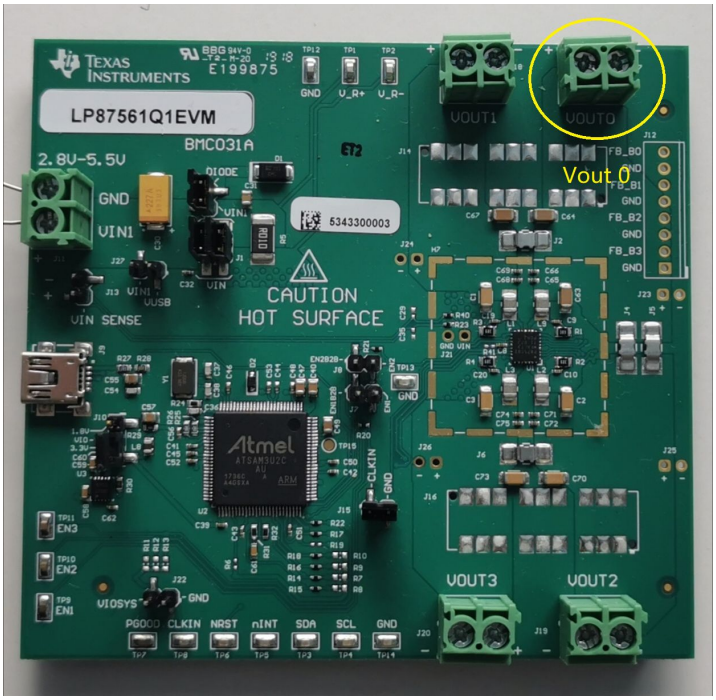
- Set up ENx pin

  Connect GPIO 17 of Raspberry Pi to EN1 pin on EVM so that you can control the output of the EVM with Raspberry Pi.





- Connect oscilloscope
  Connect one of the channels to VOUT0 (BUCK0) and another channel to EN1.



Programming Setup

1. Enabling I2C on Raspberry Pi
   Based on :
   https://learn.sparkfun.com/tutorials/raspberry-pi-spi-and-i2c-tutorial/all#i2c-on-pi
   1. Power on both the EVM and Raspberry Pi
   2. Click on **Pi Start Menu** > **Preferences** > **Raspberry Pi Configuration** > **Interfaces** > **Enable** (for I2C) > **OK**
   3. Reboot Raspberry Pi
   4. Install the necessary libraries.
      Open terminal and type in these commands to get *python-smbus (For use of i2c in script)* and *i2ctools (For detecting connected i2c devices).*

      ```
      sudo apt-get install -y python-smbus
      sudo apt-get install -y i2c-tools
      ```
   5. Find the address of the EVM
      In the terminal type:

      ```
      sudo i2cdetect -y 1
      ```

You should see a grid similar to this:

```
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- 62 -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

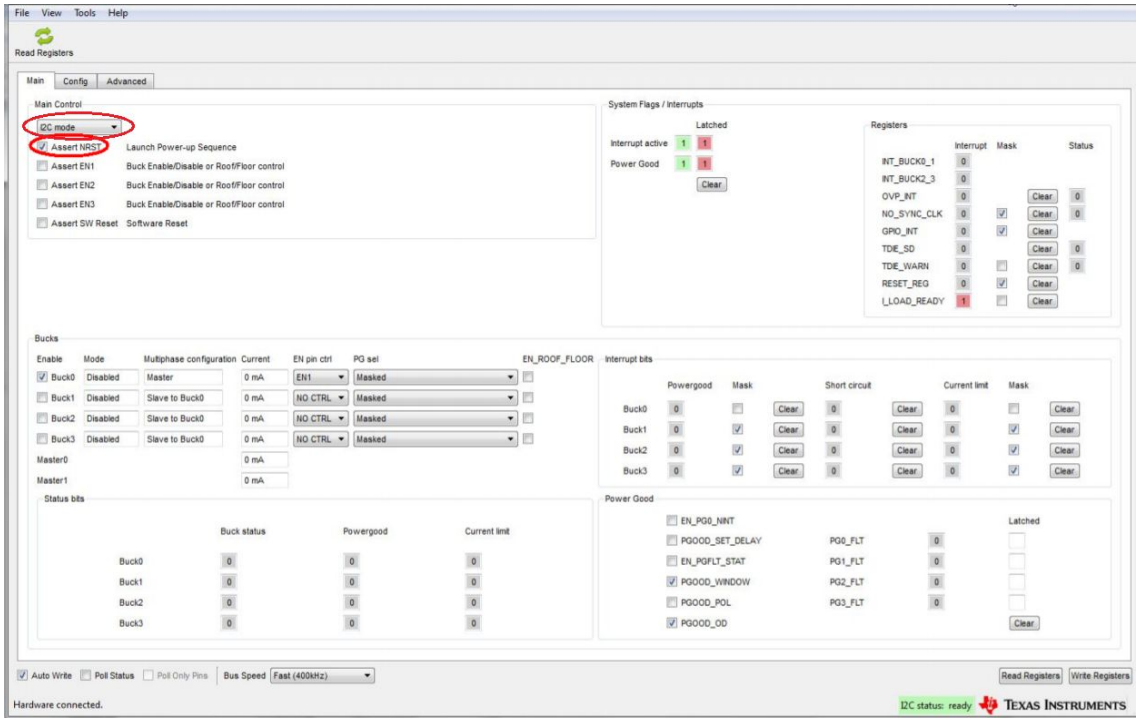
This grid represents all the connected I2C devices and their addresses. The addresses are in hexadecimal. The column represents the LSD (least significant digit) of the address. The row represents the 2nd LSD of the address. So in the case above, the address of the connected device is **0x62**

Remember to take note of the address as it is needed when writing the script.

2. Setup EVM

Based on: TI's official "The LP8756xQ1EVM (SV601325) Evaluation Module User's Guide, subsection: 'Quick Setup Guide'"

1. Download the official GUI software from TI
2. Connect the USB port on the EVM to your computer
3. Open the official GUI "LP8756 EVM", change into "i2c mode" then press "Assert NRST"



Note the content within the two ovals

3. Setup Raspberry Pi to control the oscilloscope

Based on:
https://scruss.com/blog/2013/12/15/my-raspberry-pi-talks-to-my-oscilloscope/

1. Get necessary packages and set up USBTMC

   Open terminal and type these commands

```
sudo groupadd usbtmc
sudo usermod -a -G usbtmc pi
sudo pip install pyusb
sudo pip install python-usbtmc
```

   Gather the vendor id and product id of your scope, type in this in terminal:

```
lsusb
```

   You should see something similar to this

```
Bus 001 Device 004: ID 1ab1:0588 Rigol Technologies DS1000
SERIES
```

   Remember the line “ID 1ab1:0588 Rigol Technologies DS1000 SERIES”
   In this case “1ab1” is the vendor id, and “0588” is the product id.

Create a file /etc/udev/rules.d/usbtmc.rules to put in your device's ID values

```
sudo nano /etc/udev/rules.d/usbtmc.rules
```

Then type in these texts:

```
# USBTMC instruments
# {your device ID here, for me it is Rigol DS1102E – ID
1ab1:0588 Rigol Technologies DS1000 SERIES}
SUBSYSTEMS=="usb", ACTION=="add", ATTRS{idVendor}=="{your
vendor id, for me it is 1ab1}", ATTRS{idProduct}=="{your
product id, for me it is 0588", GROUP="usbtmc", MODE="0660"
```

Reboot your Raspberry Pi

4. Setup graphing utilities on Raspberry Pi and upload script

1. Get "numpy" and "matplotlib"
   In terminal type:

```
sudo pip install numpy
sudo pip install matplotlib
```

2. Clone or download this repository on your Raspberry Pi (If not yet done)

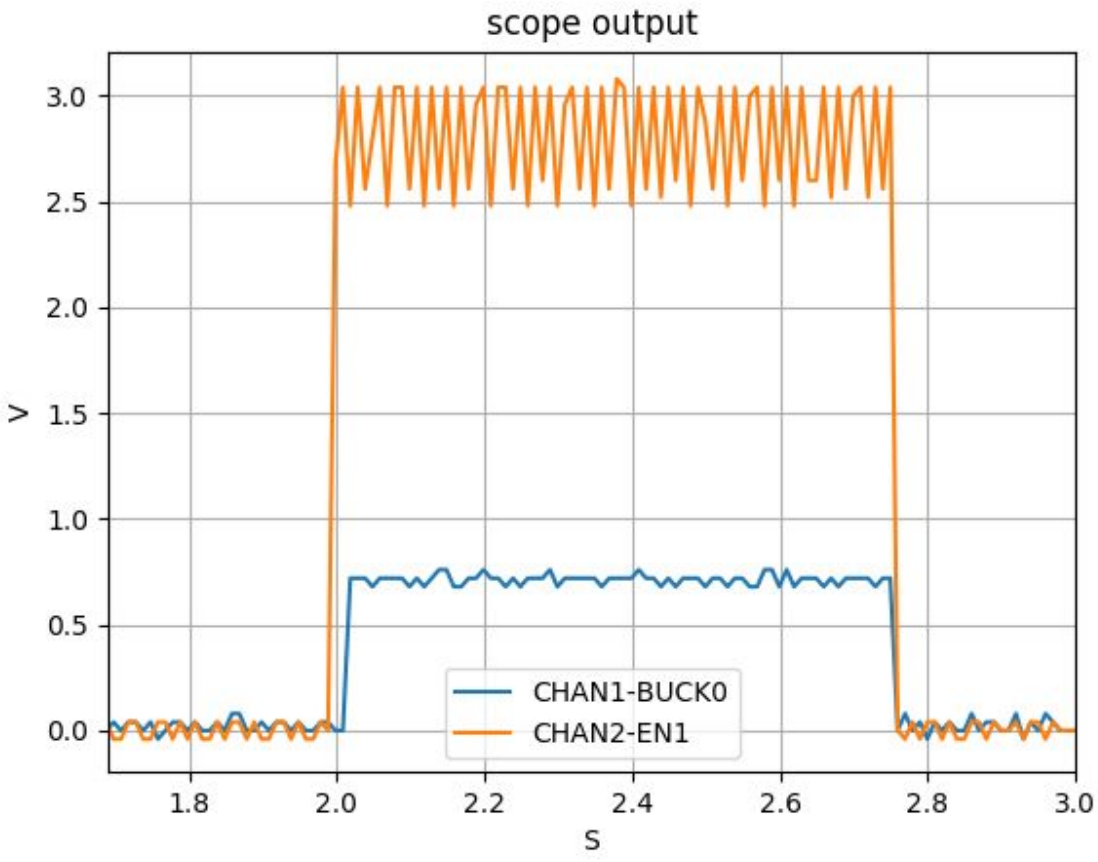
3. In python script "lp87561Testing.py" change

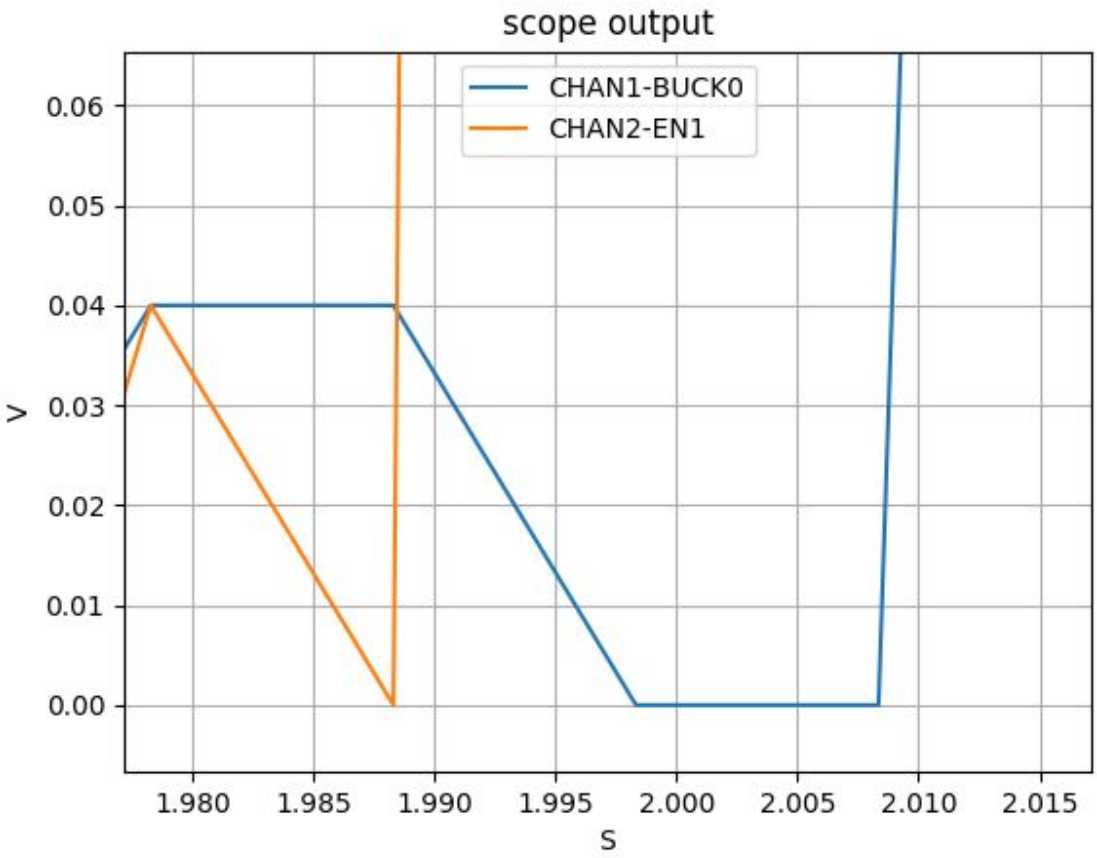
```
scope = usbtmc.Instrument(0x{your vendor id}, 0x{your product id})

address = {your EVM address}

# Consult the official TI data sheet for LP8756x data sheet for
register mappings
bus.write_i2c_block_data(address, reg_write["{characteristic you
want to change}"], [{value of that register}])

# To add a characteristic that you want to change
reg_write = {"enable" : 0x02, "Vout" : 0x0A, "slew rate" : 0x03,
"delay" : 0x12, "{characteristic you want to add}" : {Register of
that characteristic} }
```

4. Run the script on the Raspberry Pi

Result:

My results for setting the EVM at:
Slew rate : 0.47mV/us
Startup Delay : 15ms
Vout : 0.73V





Here is a zoomed-in view of the captured output during startup. You can kind of see the startup delay. CHAN2 is the output from the GPIO pin of the Raspberry Pi, CHAN1 is the output from the EVM.

References:

Technical:

- https://learn.sparkfun.com/tutorials/raspberry-pi-spi-and-i2c-tutorial/all#i2c-on-pi
- https://scruss.com/blog/2013/12/15/my-raspberry-pi-talks-to-my-oscilloscope/
- TI's official "The LP8756xQ1EVM (SV601325) Evaluation Module User's Guide"

Image source:

- https://www.electronicwings.com/raspberry-pi/raspberry-pi-i2c