



EGR 597: Internet of Things: Standards, Protocols, and Applications
Lab 3: Data Converters

Central Michigan University
2018-2019

Objective:

This experiment will use an ADC data converter on the Raspberry Pi to construct a digital light meter. The measured light intensity data will be displayed numerically as a fixed-point number on the terminal, using an integer format.

Materials Needed:

- 1) Raspberry Pi 3
- 2) Breadboard
- 3) ADS1115 ADC
- 4) MCP3008 ADC
- 5) Jumper wires
- 6) A photocell resistor
- 7) 1k Ω , 10k Ω resistor

Section 1: ADS1115 Breakout Board

A photocell or a light dependent resistor (LDR) changes its resistance depending on the amount of light irradiated on it. The light level and the resistance of the LDR are inversely related so the resistance increases when there is less light and vice versa.

Step 1: Insert the photocell resistor into the breadboard with both pins in the same column. Now insert the 10k Ω resistor into the breadboard with one pin in the same row as the bottom pin of the photocell resistor with one space between the photocell resistor and the 10k Ω resistor. The wiring schematic is shown in Fig. 1

Your setup should look like this:

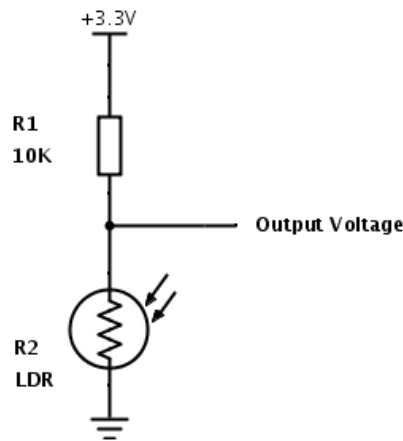


Fig. 1: Wiring Schematic for LDR

Step 2: Now it is time to wire the breadboard to the Raspberry Pi. Insert a male jumper wire into the hole between the photocell resistor and the 10k Ω resistor and connect it to the A0 pin on the ADS1115 sensor. Next, insert a male jumper wire into the same row as the top pin of the photocell and connect it to a pin on the Raspberry Pi with 5V of power. Finally, insert a male jumper wire into the same row as the bottom pin of the 10k Ω resistor and connect it to a ground pin on the Raspberry Pi as in Fig. 2.

Step 3: Wire the ADS1115 sensor to the Raspberry Pi. Use the four female jumper wires to do so, do not wire your ADS1115 sensor to the breadboard! The pinout is as follows:

- VDD to Raspberry Pi 3.3V
- GND to Raspberry Pi GND
- SCL to Raspberry Pi SCL
- SDA to Raspberry Pi SDA

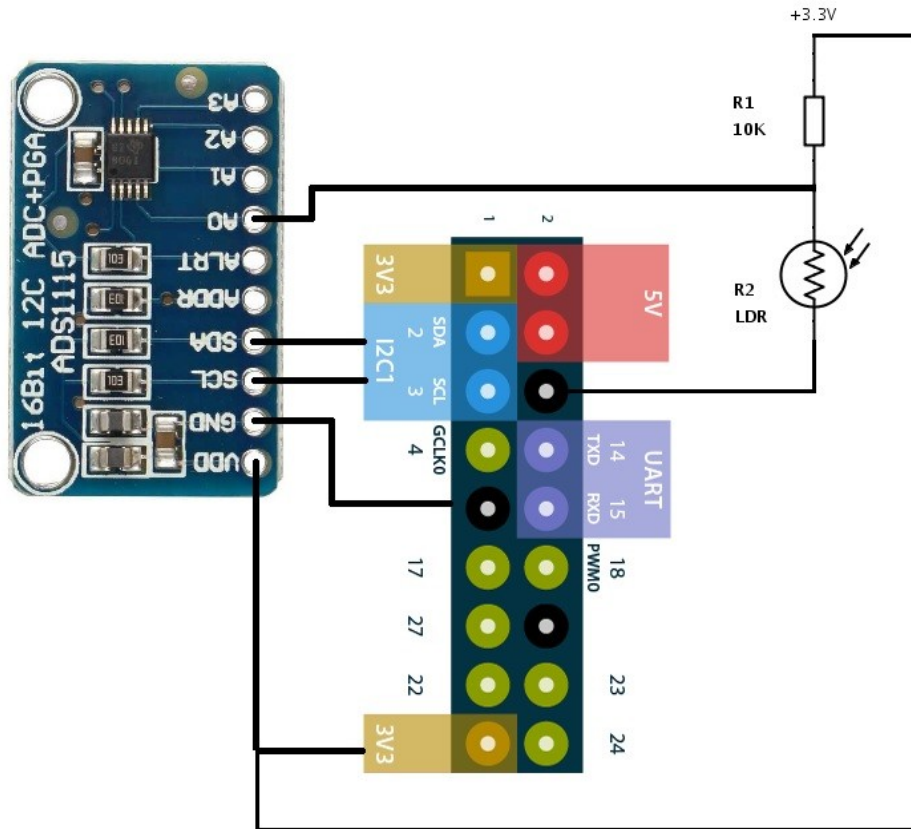


Fig.2. Wiring Layout for ADC, LDR, and Raspberry Pi

The Code

Step 1: On your Raspberry Pi, you will need to open a terminal window. Click on the terminal icon in the top left corner of your screen.

Step 2: The code that we want to use for this ADS1115 sensor is stored in a library in Github, so we are now going to install it. To do so, run the following commands by typing them exactly as they appear here and press 'return' after each one.

```
sudo apt-get install build-essential python-dev python-smbus git
```

Note: after pressing 'return' and letting this command run, you may receive a message in your terminal saying, "Do you want to continue? [Y/n]" To continue, type 'Y' and press 'return' and continue on to the next command.**

```
cd ~
```

```
git clone https://github.com/adafruit/Adafruit_Python_ADS1x15.git
```

```
cd Adafruit_Python_ADS1x15
```

```
sudo python setup.py install
```

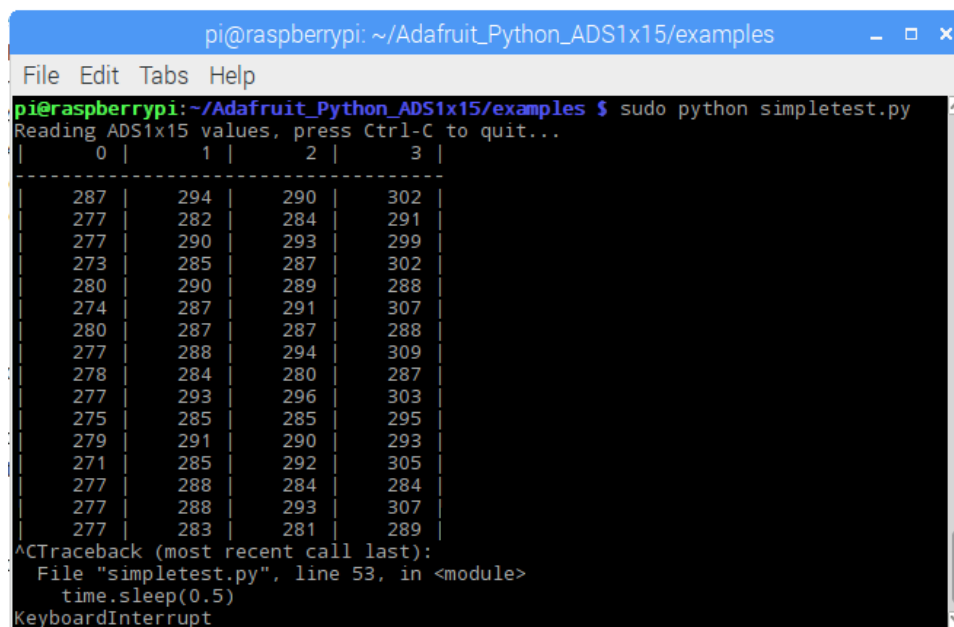
Step 3: Now let's run an example. To do this, run the following commands in a terminal window to get to the examples directory of the library.

```
cd ~/Adafruit_Python_ADS1x15/examples
```

Step 4: Now it is time to test your sensor! To do this simply run the following command in your terminal window:

```
sudo python simpletest.py
```

You should now be getting data in your terminal window that looks like this:



```
pi@raspberrypi: ~/Adafruit_Python_ADS1x15/examples
File Edit Tabs Help
pi@raspberrypi:~/Adafruit_Python_ADS1x15/examples $ sudo python simpletest.py
Reading ADS1x15 values, press Ctrl-C to quit...
 0 | 1 | 2 | 3 |
---|---|---|---|
287 | 294 | 290 | 302 |
277 | 282 | 284 | 291 |
277 | 290 | 293 | 299 |
273 | 285 | 287 | 302 |
280 | 290 | 289 | 288 |
274 | 287 | 291 | 307 |
280 | 287 | 287 | 288 |
277 | 288 | 294 | 309 |
278 | 284 | 280 | 287 |
277 | 293 | 296 | 303 |
275 | 285 | 285 | 295 |
279 | 291 | 290 | 293 |
271 | 285 | 292 | 305 |
277 | 288 | 284 | 284 |
277 | 288 | 293 | 307 |
277 | 283 | 281 | 289 |
^CTraceback (most recent call last):
  File "simpletest.py", line 53, in <module>
    time.sleep(0.5)
KeyboardInterrupt
```

Fig. 3. Terminal window reading light intensity values from ADS1115

Since we connected the output from the LDR to pin A0 of the ADS1115, our results will be shown in the first column. If you cover the photocell resistor with your hand the data displaying in the first column should increase, if this happens then everything is set up correctly.

To end the code, press Ctrl+C.

Section 2: MCP3008

In this section, you will utilize an ADC available as a discrete IC.

Section 1: Insert the MCP3008 into the breadboard as in Fig. 4.

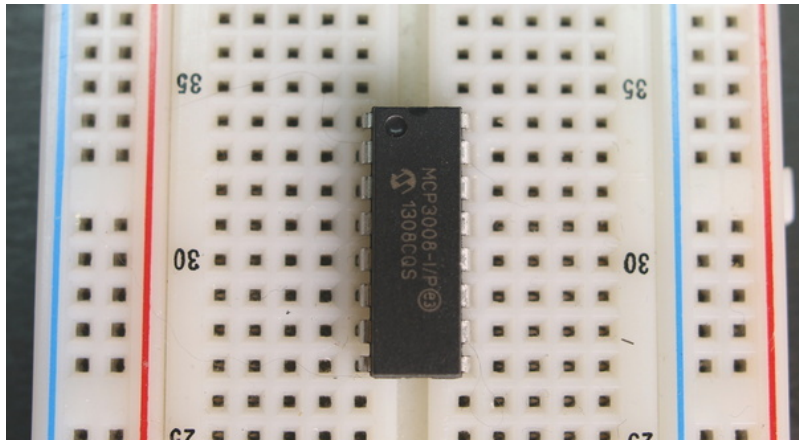


Fig. 4. IC Placement on a Breadboard

Section 2: Once the MCP3008 is inserted, wire the ADC and Raspberry as shown in Fig. 5.

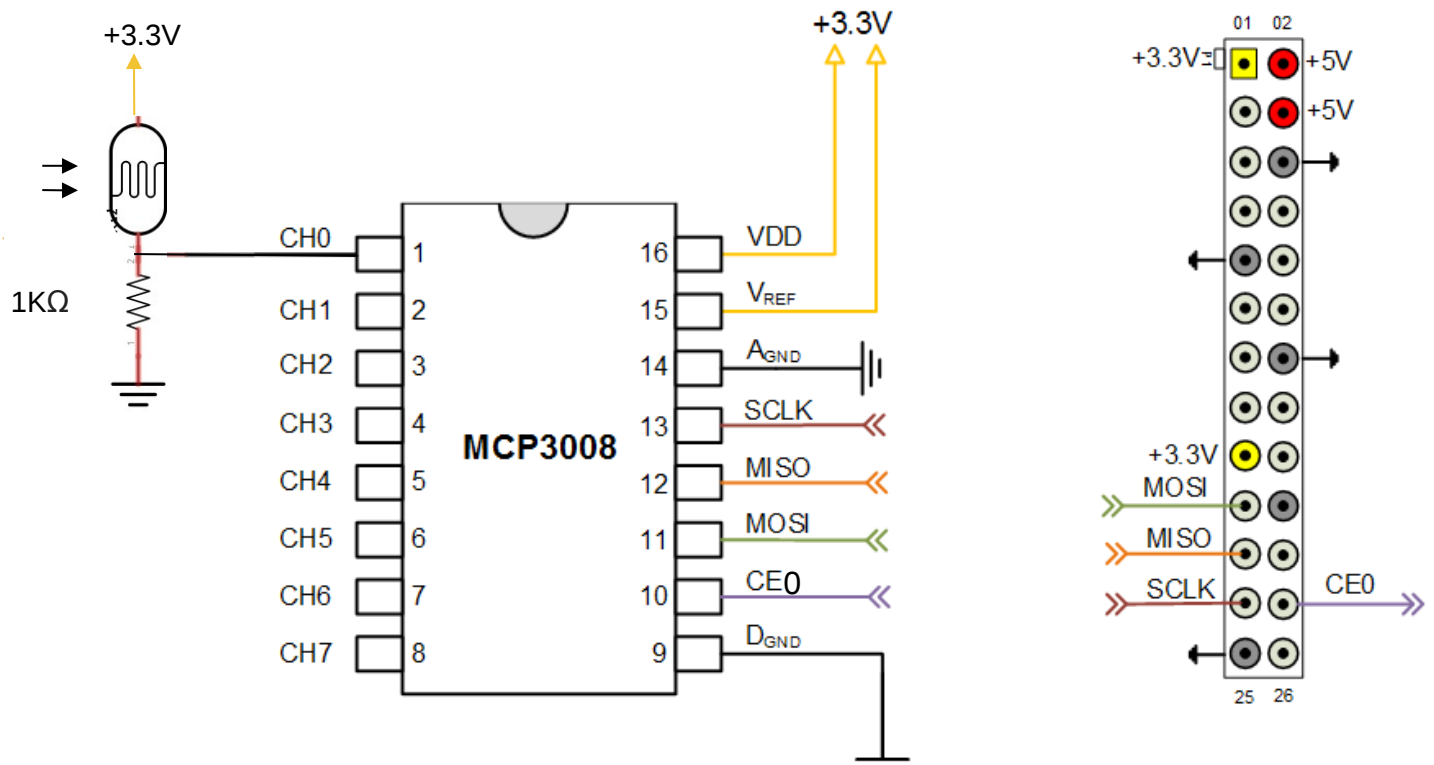


Fig. 5. Wiring Schematic for LDR, MCP3008, and Raspberry Pi

Note: Make sure you have placed the ADC in the breadboard in the right direction. To keep track of which way is which, note the location of the half circle on one of the sides of the ADC.

The Code

Step 1:

- Open the file manager tab at the top left side of the screen of your Raspberry Pi.
- Click “File” , “Create New”, “Empty File”, and name your file something along the lines of “MCP3008 ADC”, be sure to include your name in the title and take note of where your file is located on the Pi. Save the file as a Python file, it must have “.py” after the title.
- Enter the code located below into your file and be sure to save it.
- Once the code is saved you can run the program. Do this by first closing out of the text file, then right click on the file and select “Open With”, and then “Python 2 (IDLE)”. Now the program can run, do this by pressing F5 on your keyboard or by clicking “Run” and then “Run Module”.

You should now be getting data in your terminal window. If you cover the photocell resistor with your hand the data displaying in your terminal should change, if this happens then everything is set up correctly. .

To end the code, press Ctrl+C.

```
import spidev
import time
```

```
#Define Variables
delay = 2
ldr_channel = 0
```

```
#Create SPI
spi = spidev.SpiDev()
spi.open(0, 0)
```

```
def readadc(adcnum):
    # read SPI data from the MCP3008, 8 channels in total
    if adcnum > 7 or adcnum < 0:
        return -1
    r = spi.xfer2([1, 8 + adcnum << 4, 0])
    data = ((r[1] & 3) << 8) + r[2]
    return data
```

while True:

ldr_value = readadc(ldr_channel)

print("LDR Value: %d" % ldr_value)

time.sleep(delay)

Lab Report:

Include the following in your lab report

1. Schematics for both ADC and LDR hookups.
2. Updated Python codes.
3. Address the following questions
 - a. How does the voltage reading vary with high and low light?
 - b. How would you change the wiring and code to display light value in third column of the terminal for ADS1115?
 - c. Explain the reasons for seeing non zero values in columns 1-3 of the terminal for the code provided in ADS1115? How would you modify it for better presentation?
 - d. Why did we not utilize the same resistor in the voltage divider circuit of both circuits? Can we utilize the same resistor in both places? Discuss the same.
 - e. The resistor used as a pull up resistor in one circuit, and pull down resistor in the other circuit. Comment on the pros and cons.
4. Update the python code for MCP3008 ADC to display light intensity value at a rate of 5 Hz.
5. Implement a moving average filter (n=4) to display the light intensity value at a rate of 2 Hz.