Lab 3: INTERFACING ANALOG TO DIGITAL CONVERTER (ADC) WITH A MICROCONTROLLER

---

1 Learning Objective

In this lab, you are going to

1. Learn about interfacing analog sensors with microcontrollers using A/D converters.
2. Use an ADC chip to analyze analog signals.
3. Examine the characteristics of a signal using an oscilloscope.

2 Components Needed

Raspberry Pi 3, resistors, temperature sensor, ADC chip MCP3008, and light sensor

3 Lab Activities

3.1 Introduction to Raspberry Pi's SPI port

SPI, or Serial Peripheral interface is a communication bus that is used to interface one or more slave IC (integrated circuits) to a single master device (Raspberry Pi is the master in this lab). In lab 3, you used the I2C port as output. However, the SPI port could have been used instead. Please note that SPI is a faster bus than I2C. On the other hand, I2C can connect many devices with only a 2 wire bus, while each slave device for SPI requires an additional wire bus. There are three SPI wires shared by all slave devices on the bus:

a. Master in slave out (MISO), data from slave to master on this port

b. Slave in master out (MOSI), data from master to slave on this port

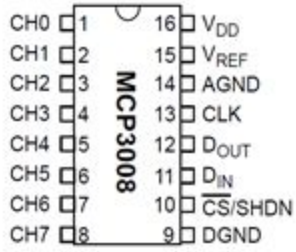
c. Device clock (CLK), clock for synchronizing communications

Each slave device will have one additional port connected to the master device. This port is for selecting a device to communicate with. For Raspberry Pi, there are two available ports for you to connect with your slave devices. Once you connect devices, data transmission can happen. During each clock cycle, the master sends data on the MOSI line and the slave reads it. At the same time, the slave sends data on the MISO line, and the master reads it. This behavior is maintained even if only one line has data to transmit.

To be able to use SPI, you must enable it on the Raspberry Pi. To do this, follow the instructions from Lab 3 for enabling I2C, except instead of selecting I2C select SPI.

3.2 Introduction to MCP3008 ADC

Your Raspberry Pi has no built-in analog inputs. This means it cannot read any data from analog sensors. The MCP3008 is a 10-bit, 8-channel, SPI-based ADC (analog to digital converter). It communicates with the Raspberry Pi using the SPI bus on the Raspberry Pi's GPIO header.



The following table shows how you can connect your Raspberry Pi to the chip:

- MCP3008 VDD -> 3.3V
- MCP3008 VREF -> 3.3V
- MCP3008 AGND -> GND
- MCP3008 CLK -> SCLK
- MCP3008 DOUT -> MISO
- MCP3008 DIN -> MOSI
- MCP3008 CS -> #22
- MCP3008 DGND -> GND

Follow this tutorial for more details.

When connected to the Raspberry Pi, the ADC chip will take analog input from CH0-CH7 and provide 1/0 digital signal to the Dout pin. The following communication happens when the Raspberry Pi tries to read from the ADC chip:

1. The Raspberry Pi will first send a byte containing the digital value 1. The MCP3008 will send back its first byte, which is not important, as a response.

2. Then the Raspberry Pi will send a second byte to indicate which channel on the MCP3008 chip should receive the analog signal.

3. As the result will be a 10 bit data, which cannot be held by a single byte, MCP3008 will send back the second byte, which contains two bits of the conversion result (which is the 8th and 9th bit).

4. The Raspberry Pi then sends a response byte to indicate that the previous byte was received, and then MCP3008 sends back the last byte containing the rest of the bits (bits 0 to 7) of the converted digital value of the analog signal.

5. Finally, the Raspberry Pi merges bits 8 & 9 with bits 0 through 7 to create the 10 bit digital value from the conversion.

3.3 Reading Data from Sensors (optional)

We will use a photocell (CdS photoresistor) as an example to demonstrate how to read data from sensors through MCP3008. Under normal light conditions, the resistance of the photocell is about 5-10KΩ, but in the dark it goes up to 200KΩ. We can use it just like we use a normal resistor.

We can connect one side of a 10k resistor to the power of 3.3V. (To protect the chip, make sure you connect to the 3.3V pin). Then connect the other side of the 10K resistor to two wires, one connected to the photoresistor and then to the ground, the other connected to the input pin of MCP3008. With bright light in the room, the resistance will drop to a low value. This will cause a higher current to flow through the photoresistor, resulting in a voltage drop across it. The input voltage will drop to almost 0V. In the dark, the voltage will go up near 3.3V due to the high resistance in the photocell relative to the resistance in the resistor.

To read the data, follow this tutorial.

Use your Raspberry Pi to read data from a light sensor and a temperature sensor **(Two sensors should connect to different channels and then use software to read data one by one from the correct channel).** Display the current readings in the command window.

For the temperature sensor, the temperature range is approximately -50°F-280°F corresponding to 0-3.3V. For the lighting sensor, we will use a scale of 0 to 100 which corresponds to 0-3.3V. You can find the temperature sensor datasheet here.

3.4 Building an Oscilloscope with Raspberry Pi and the MCP3008 Chip

In the previous activity, you accomplished reading data from the MCP3008 and processing that input data with the Raspberry Pi. In this activity, you are going to build a more complex system-- a simple oscilloscope using your Raspberry Pi with the MCP3008. The simple oscilloscope will have two functions:

1. **Recognize a wave**: This is a very basic feature that all oscilloscopes should have. In this lab assignment, you should be able to display the name of the input wave on a command window: You should implement an algorithm to distinguish square, sine, and triangle waves based on the data sampled. Every time your algorithm detects a shape change, print out the name of the shape to the command window. Be aware that there will be noise when sampling the data
2. **Characterizing a wave:** The mini-oscilloscope should be able to find out the frequency for the input waveform and display it to a command window or with your visualized wave.

Build a simple oscilloscope that will meet the requirements as above. You can use Lab 3's output as input in this activity. Two teams can work with each other (one team generates waveform and the other recognizes it). It's not the best practice to use the same Raspberry Pi generating and reading the wave. You can also use a function generator for the waves. Each team should provide their own solution by the end of the deadline and demonstrate it to the TA. Make sure to note the sampling rate of your oscilloscope.

4 Question to Explore

1. Summarize the difference between SPI and I2C ports. Explain in what situation using the SPI ports is better than the I2C ports, and vice versa.
2. What are the various types of ADCs in use? Which type of ADC is MCP3008 and what are its advantages/disadvantages?
3. What is the sampling rate for your oscilloscope?
4. If you use the same Raspberry Pi to do waveform generation and waveform recognition at the same time, you might generate a waveform that the frequency keeps changing and get random readings from the MCP3008. Explain why this is the case.
5. It is highly likely that your sampled data contains lots of noise (in amplitude and frequency). How can you filter the noise? Explain your method.

5 What to Turn In

- Answer to Questions (4)
- Code (1)
- Demo: recognizing 3 waveforms (3), recognizing frequency (2)