

Lab 07A: PCB Design

Lab Presentation: 03/11/2025 (Tuesday), 03/13/2025 (Thursday)

Prototype Demo: 03/25/2025 (Tuesday), 03/27/2025 (Thursday)

1. Introduction
2. Analog Sensors with Raspberry Pis and Other Embedded Systems
3. The Microchip MCP3008 A/D Converter
4. Testing with Grove Sensors
5. Step 1: Prototype the Circuit with a Breadboard
6. Step 2: Testing the Hardware with Software
7. Test Suite

1. Introduction

The objective of this lab is to design a simple circuit board to act as a custom Raspberry Pi shield to interface with multiple analog sensors. From this experience, you will get an overview of the PCB design process and learn how to solder. By having this custom shield on hand, we hope you will buy your own Raspberry Pi and analog output sensors (instead of the expensive and limiting GrovePi board) to tinker at home.

The main idea of this lab is to cover the whole process, so we do not intend to explore PCB design deeply, which is a very large field that requires a dedicated course to master. In addition, the PCB design we will work with is more at the hobbyist level, which will not fully resemble PCB design in the field.

2. Analog Sensors with Raspberry Pis and Other Embedded Systems

As we learned in class, a large number of sensors used in real applications provide analog signals. Real-world analog signals need to be converted to digital signals in order to be processed. We might have an embedded ADC on our controller platform, but sometimes ADCs are not included, or they don't meet the application requirements.

The Raspberry Pi, for instance, is an embedded system that does not have an ADC. It is based on the Broadcom BCM2837 SoC, which has a quad-core 1.2 GHz ARM Cortex A53 processor, a VideoCore IV @ 400 MHz GPU, and many other digital peripherals such as SPI, I2C, UART, etc. Many other computing platforms that can run Linux, such as the Intel Edison and general PCs, also do not have an ADC chip available. While we have access to the GrovePi boards, these are education boards designed to work particularly with Grove sensors, and there are many higher quality, specialized sensors in the market.

If we want to connect an analog sensor directly to Raspberry Pi, we need to use an external ADC chip, which is very cheap nowadays and can be interfaced via digital protocols such as I2C and SPI. We are going to use the MCP3008 chip for this task.

3. The Microchip MCP3008 A/D Converter

The MCP3008 is a 10-bit Analog-to-Digital Converter (ADC) that is ideal for embedded control applications. It features 8 analog input channels and supports SPI interface. The sampling frequency is 200 kilo samples per second (or ksp/s) when powered at 5V and 75 ksp/s when powered at 2.7-3.3V. You should spend a couple of minutes going over the MCP3008 [datasheet](#) and studying its specs.

We are using an MCP3008 with a 16-pin PDIP socket. Its pinout diagram is shown in Figure 1. Pins 1 to 8 are input channels, each one can be connected to one analog sensor. Pin 16 (VDD) is the power source and has to be between 2.7 and 5.5V. VREF is the digital voltage reference. AGND and DGND are, respectively, the analog and digital ground. Finally, CLK, DOUT, DIN, and CS are the SPI signals that can be connected to a digital processor.

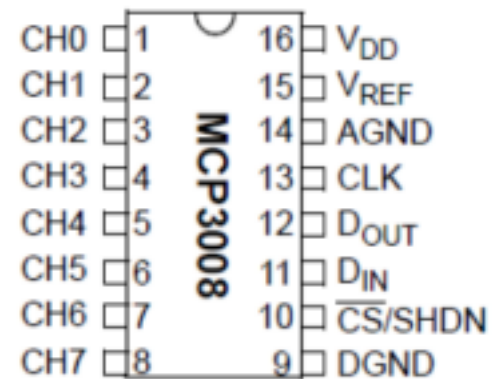


Figure 1 - MCP3008 pinout

To simplify this task, we will use SPI libraries for the Raspberry Pi to interface with the MCP3008. Figure 2 shows the Raspberry Pi 40-pin header. We can see that pins 19, 21, 23, 24, and 26 can be used to connect an external device to the SPI bus.

In Table 1, we have labeled the connections that need to be made between the MCP3008 and Raspberry Pi header. **Most sensors on the market can operate on both 3.3V or 5V, but there are many sensors that only operate on 3.3V. Thus, we will hardwire a 3V3 line from the Raspberry Pi to all the sensor slots. The Raspberry Pi has multiple 3V3 and GND pins, and any of them can be used.**

| MCP3008 | Raspberry Pi |
|-------------------------|------------------------|
| Pin 16 - VDD | 3V3 - Pin 1 |
| Pin 15 - VREF | 3V3 - Pin 1 |
| Pin 14 - AGND | GND - Pin 9 |
| Pin 13 - CLK | (SCLK) - Pin 23 |
| Pin 12 - DOUT | (MISO) - Pin 21 |
| Pin 11 - DIN | (MOSI) - Pin 19 |
| Pin 10 - CS/SHDN | (CE0) - Pin 24 |
| Pin 9 - DGND | GND - Pin 9 |

Table 1 - Pin Connection Table (connections between MCP3008 and Raspberry Pi)





















| WiringPi | BCM(Name) | Physical | Physical | BCM(Name) | WiringPi |
|----------|----------------|----------|---|-----------|------------------|
| | 3v3 Power | 1 |  | 2 | 5v Power |
| 8 | BCM 2 (SDA) | 3 |  | 4 | 5v Power |
| 9 | BCM 3 (SCL) | 5 |  | 6 | Ground |
| 7 | BCM 4 (GCLK0) | 7 |  | 8 | BCM 14 (TXD) 15 |
| | Ground | 9 |  | 10 | BCM 15 (RXD) 16 |
| 0 | BCM 17 | 11 |  | 12 | BCM 18 (PCM_C) 1 |
| 2 | BCM 27 (PCM_D) | 13 |  | 14 | Ground |
| 3 | BCM 22 | 15 |  | 16 | BCM 23 4 |
| | 3v3 Power | 17 |  | 18 | BCM 24 5 |
| 12 | BCM 10 (MOSI) | 19 |  | 20 | Ground |
| 13 | BCM 9 (MISO) | 21 |  | 22 | BCM 25 6 |
| 14 | BCM 11 (SCLK) | 23 |  | 24 | BCM 8 (CE0) 10 |
| | Ground | 25 |  | 26 | BCM 7 (CE1) 11 |
| | BCM 0 (ID_SD) | 27 |  | 28 | BCM 1 (ID_SC) |
| 21 | BCM 5 | 29 |  | 30 | Ground |
| 22 | BCM 6 | 31 |  | 32 | BCM 12 26 |
| 23 | BCM 13 | 33 |  | 34 | Ground |
| 24 | BCM 19 (MISO) | 35 |  | 36 | BCM 16 27 |
| 25 | BCM 26 | 37 |  | 38 | BCM 20 (MOSI) 28 |
| | Ground | 39 |  | 40 | BCM 21 (SCLK) 29 |

Figure 2 - Raspberry Pi pinout

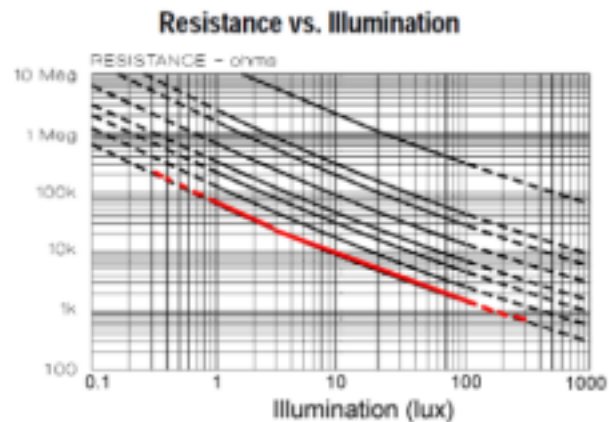
4. Testing with Grove Sensors

To test our PCB design, we will use analog-output Grove sensors without the GrovePi board. In particular, we will use the Grove light and sound sensors.

The light sensor consists of a photoresistor which has a resistance that decreases with the

intensity of light it is subjected to. In dark environments, its resistance is very large. Under direct light, its resistance becomes small. There is a curve that shows the relation between resistance and illuminance, which is measured in lux, as shown in Figure 3. In most applications, we do not need to have a quantitative value for the illuminance, but only a qualitative measurement. That is, we just want to know if an environment is dark or not instead of the actual value of lux. In this case, using a simple pre-calculated threshold from experimentation to create a binary output (light or dark) is sufficient.

The Grove sound sensor is a simple microphone, which is not accurate enough to measure the characteristics of sound but is good enough to detect if the environment is noisy or not. Through experimenting with the analog output, you will determine a threshold to detect if someone is tapping on the sensor.



Source: <https://learn.adafruit.com/photocells/measuring-light>

Figure 3 - Resistance vs. illuminance

5. Step 1: Prototype the Circuit with a Breadboard

Before we design and manufacture our PCB, we have to prototype the target circuit using a breadboard. In Figure 4, we have drawn out the circuit you should breadboard. Because the Grove sensors have a special connector, see Figure 5 for how to breadboard with them.



Figure 4 - Circuit to be built.

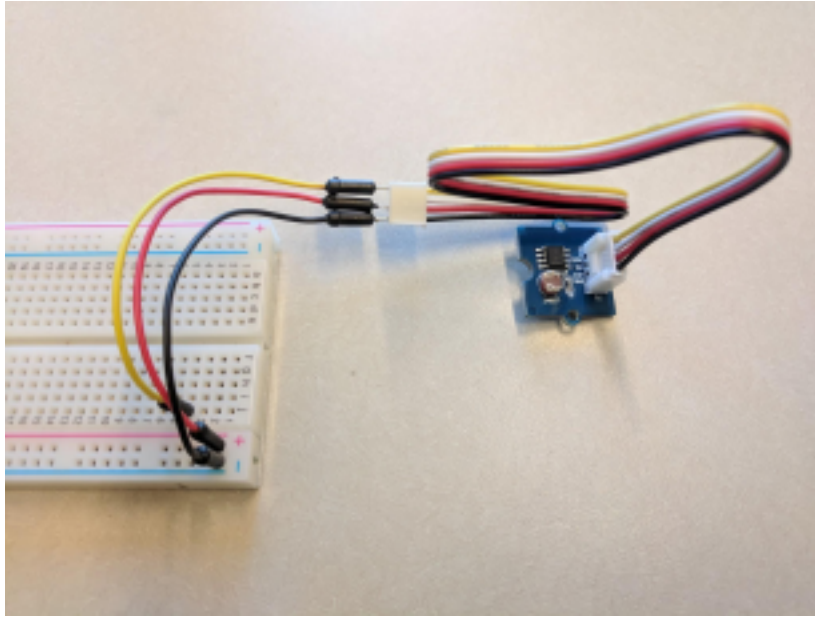


Figure 5: How to connect a Grove sensor to a breadboard.

Most student errors with breadboarding are rooted in not having a clean and neat network of wires. Try to practice creating a cleaner breadboard during this exercise. On most embedded boards, LEDs are included to help debug software designs. Think of these as an alternative to print statements. This is why we will be including an LED on our PCBs. Do not forget to add an adequate series resistor to limit the current through the LED.

After building the circuit, do a quick visual test. Use the multimeter to make sure you have not short-circuited any components.

6. Step 2: Testing the Hardware with Software

Many companies have dedicated test and verification teams responsible for designing complex systems that test and verify if a PCB or Integrated Circuit (IC) design is correct. While we are not a company, we still need to be ready to test our PCB once we have received them to check if the manufacturer made any errors.

You will need to write software to read the sensors and actuate on the LED based on the sensing data (i.e. you detect a loud noise). The first thing we have to do is to enable the hardware SPI interface on Raspberry Pi. Login to your Raspberry Pi via SSH and execute the following command:

```
sudo raspi-config
```

Then, select Interfacing Options, select SPI, select Yes, and finally OK. This [website](#) has screenshots of the whole process. Next, install the Python libraries for the MCP3008 via pip:

```
sudo pip3 install adafruit-mcp3008
```

You can find the source code and examples of the MCP3008 library on this [website](#). **The best way to learn how to use a library is to read the source code, examples, and (if they exist) tests. Run the test code to make sure your circuit works.** Try to create functions that make your code modular and allow you to reuse it in the future when you buy your own analog sensors. Code to turn on/off the LED can be based on the default GPIO library that is included on Raspberry Pi (found [here](#)) or Adafruit's portable python GPIO library (found [here](#)). Examples of how to set a GPIO pin as an output and make it HIGH or LOW can be found on those sites too.

7. Test Suite

A test suite is a collection of test cases that are intended to be used to show that a system satisfies a specified set of behaviors. Your task is to write a testing routine that continuously performs the sequence of actions listed below in an infinite loop.

1. Blink the LED 5 times with on/off intervals of 500ms.
2. For about 5 seconds, read the output of the Grove light sensor with intervals of 100 ms and print the raw value along with the text "bright" or "dark" (you will have to determine a threshold for this through experimentation).
3. Blink the LED 4 times with on/off intervals of 200ms.
4. For about 5 seconds, read the output of the Grove sound sensor with intervals of 100 ms and print the raw value. If the sound sensor is tapped (i.e. the sound magnitude goes above the threshold you decide from experimentation), the LED should turn on for 100 ms.

| <u>Points</u> | <u>Description</u> |
|----------------------|--|
| <u>DEMO</u> | |
| 2 | LED blinks 5 times with 500ms on/off times at the beginning of the test. |
| 4 | The light sensor is read at 100 ms intervals for 5 seconds. |
| 2 | LED blinks 4 times with 200ms on/off times between sensor tests.. |
| 6 | The sound sensor is read at 100 ms intervals for 5 seconds. The LED turns on for 100 ms when the sound magnitude goes above a threshold while not affecting the 5-second sequence time of sampling the sound sensor. |
| <u>CODE</u> | Students must submit test_suite.py. |
| 2 | The test suite follows the desired structure. |
| 2 | Code correctness (no Python syntax errors, sufficiently bug-free for the assignment, etc.) |
| | Total Possible: 18 |