Experiment A8
Microcontrollers
Procedure

Deliverables: checked lab notebook, demonstration of working device to Lab TA

Recommended Reading: Section 6.4; Chapter 18 of the textbook

You do NOT have to write a tech memo for this lab. Just make sure the TA fills out the score sheet as you complete each part of the lab. **Each item on the score sheet must be completed before the end of lab for you to receive credit for it.**

Overview

A microcontroller is a rudimentary computer packed into a small IC that can be programmed to automate various tasks. In this lab, you will use an Arduino UNO microcontroller to read the voltage output from the thermistor voltage divider circuit used in A4, calculate the temperature, and light up different colored LEDs depending on the temperature.

There is no tech memo for this lab. Rather, you will construct and program two different electronic systems and demonstrate them to the TA, who will award you points.

Part I: Controlling the Brightness of an LED

Background

The Arduino UNO is an inexpensive microcontroller commonly used by hobbyists and engineering students. Shown in Figure 1, the UNO is connected to a breadboard where various sensors and circuits can be implemented (i.e. a thermistor in a voltage divider circuit). Here are a few of the more salient features of the UNO.

- **Analog inputs** - The UNO has a 10-bit analog-to-digital converter (A/D) that can read up to six different analog voltages into memory. Voltages between 0V and 5V are mapped to integer values between 0 and 1023, respectively.
- **Digital input/outputs** - The UNO has 14 different digital input/outputs (I/O). These pins always output a voltage of 0V (LOW) or +5V (HIGH). However, the pins marked with a '~' can output a 500Hz *pulse width modulation* (PWM) square wave. This square wave oscillates between 0V and +5V. Varying the *duty cycle* (% of time the +5V is ON) creates an average voltage that can be treated as an analog output.
- **+5V DC Power** – This pin outputs a constant +5V, which can be connected to a breadboard to power various sensors and peripherals. (It is only capable of producing a very small amount of current, so beware of voltage droop!)
- **USB Connection** – Code and commands are sent to the UNO and data can be received from the UNO via a USB cable. The USB cable also provides the +5V power to the UNO. (In the absence of a USB connection, the UNO can be powered via the "Vin" pin or the 7 – 12V DC 2.1mm barrel connector.)

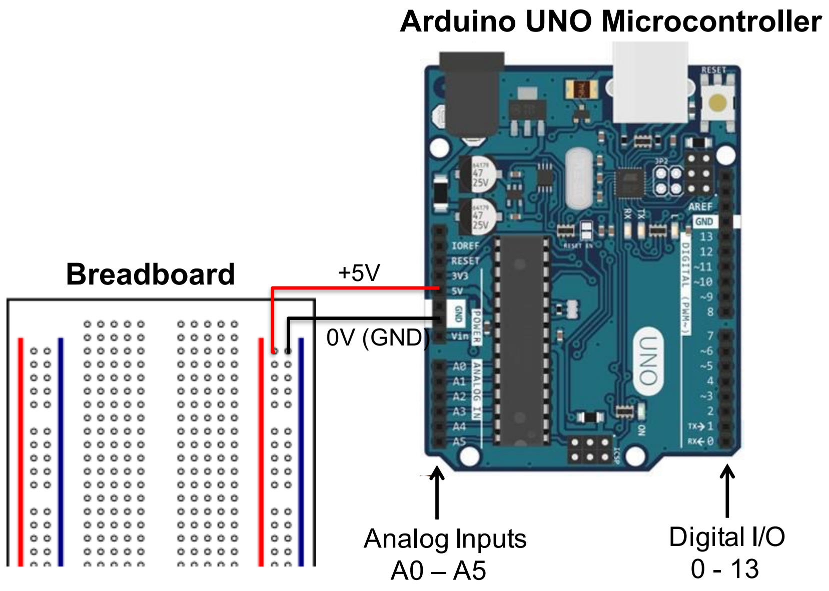

Figure 1 – The Arduino UNO Microcontroller works in tandem with breadboard.

In the first part of this lab, you will use a potentiometer as a knob to control the brightness of an LED. A potentiometer is a variable resistor whose value can be changed by turning a knob. Shown in Figure 2 below, the relative resistance between A and B and between B and C changes as the knob is turned. The potentiometer is essentially a variable voltage divider. Connecting +5V to terminal A and 0V (ground) to terminal C creates a variables voltage source on terminal B.

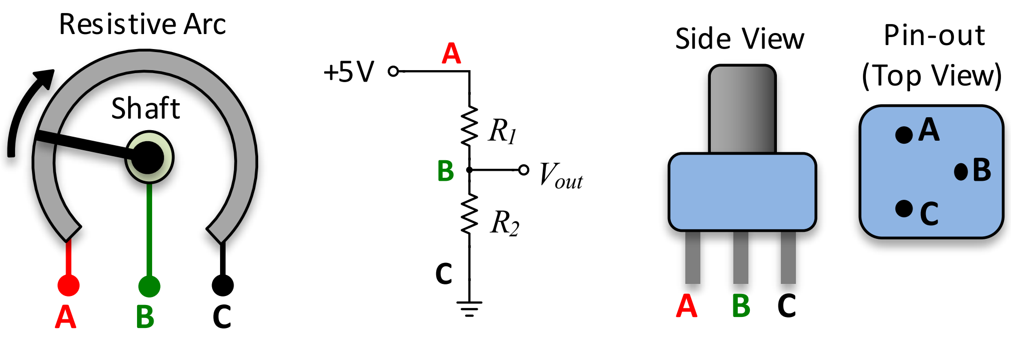

Figure 2 – (Left) A schematic representing the operation of a potentiometer. (Center) The potentiometer used as a voltage divider. (Right) The potentiometer has three pins A, B, and C. Pin B is sometimes called the "wiper".

The potentiometer can be used as a variable voltage source. However, it is not capable of producing large currents, because the resistances of the potentiometer are fairly large, typically over 1000 Ohms. Thus the potentiometer cannot be used to directly control the brightness of the LED. To overcome this, we will use the Arduino microcontroller as a digital amplifier.

Shown in Figure 3, the analog input pin A0 of the Arduino reads the voltage from the potentiometer and stores it in memory as a 10-bit integer between 0 and 1023. This 10-bit integer is then linearly mapped to an 8-bit integer between 0 and 255, which controls the duty cycle of the PWM signal from pin 9. (In the code, the "analogWrite" function maps the integer 0 to 0% duty cycle, and 255 to 100% duty cycle.) The LED is connected to this digital output, and its brightness can be logically traced back to the angle of the potentiometer knob.

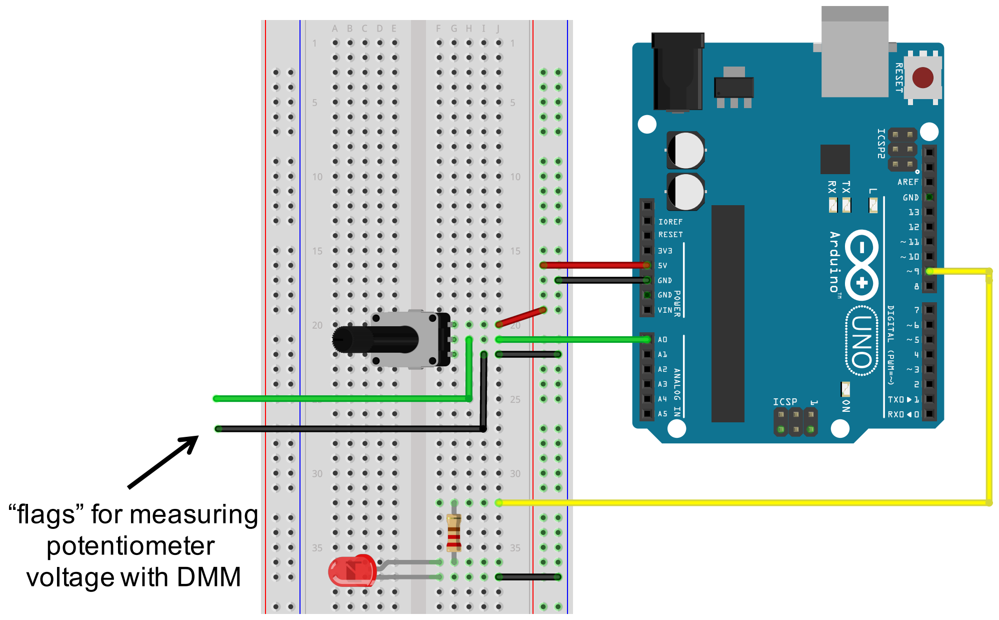

Figure 3 – The potentiometer and LED are mounted in the breadboard and connected to the Arduino.

Procedure

Important: You will use the Arduino to power the circuit on the breadboard. Do NOT turn on the breadboard.

1. Connect the Arduino UNO to the lab computer via the USB cable. You should see a green LED light up on the Arduino.
2. Use red and black jumper wires to connect the +5V and GND pins on the Arduino to the vertical bus lines on the breadboard, as shown in Fig. 1. Use the orange handheld DMM to verify that it is providing +5V of power.
3. Before you plug the potentiometer into the breadboard, connect the DMM to pins A and B, and measure the resistance. Turn the knob, and you should see the resistance change.
4. In your lab notebook, write down the resistance when the knob is turned completely clockwise, and the resistance when the knob is turned completely counter-clockwise. Repeat this with pins B and C.
5. Measure the resistance between pins A and C. Does the resistance change for pins A and C when you turn the knob? Base on the illustration on the left side of Fig. 2, does this make sense?

6. Insert the potentiometer into the breadboard near the top, as shown in Fig. 3. Make the following connections:
    a. Pin A on potentiometer → +5V
    b. Pin B on potentiometer → Analog input A0 on the Arduino
    c. Pin C on potentiometer → 0V (GND)
7. Use the DMM to measure the output voltage from the potentiometer between pin B and ground. Turn the knob, and the voltage should change accordingly.

Pro-tip: Insert a jumper wires into the rows of the breadboard where you wish to connect the DMM mini-grabbers. (These wires are sometimes called "flags".) DO NOT try to grab onto the pins of the potentiometer or the base of any wire already in the breadboard.

8. Connect the LED and a 220Ω in series between digital output pin 9 and ground, as shown in Fig. 3. Make sure that the shorter "cathode" wire of the LED is on the ground (negative) side of the circuit.
9. Download the A8 "Part I Template" code template from the lab webpage. Read the comments and replace the *** in the beginning with the correct pin numbers for the analog input from the potentiometer and digital output driving the LED (i.e., "A0" and "9").
10. Save the code with an intelligent file name (i.e. "A8_potentiometer_yourName.ino").
11. In the Arduino IDE software, go to "Tools" > "Board" and make sure either "Arduino/Genuino" or "Arduino UNO" is selected.
12. In the Arduino IDE software, go to "Tools" > "Port" and select the COM port that says "(Arduino/Genuino Uno)" next to it.
13. Click the check mark at the top of the Arduino program to check the code for errors.
14. Press the arrow button to compile the program and send it to the Arduino.
15. Go to "Tools" > "Serial Monitor" (or press "Ctrl + Shift + M") to view the output from the "Serial.print()" commands at the bottom of the screen. You should see an integer between 0 and 1023 corresponding to the potentiometer angle (or voltage) continuously printed. (Make sure the Baud rate is set to 9600.)
16. Turn the knob. The printed integer values should change, as well as the brightness of the LED. Set the knob, so that the integer is about 500 and the voltage on the DMM is around 2.5V.
17. Use the oscilloscope to measure the PWM signal that is driving the LED. Connect the mini-grabbers to the appropriate places in the LED circuit, and press the "Autoset" button at the top of the oscilloscope. You should see a square wave.
18. Turn the knob, and observe how the duty cycle of the PWM signal changes on the oscilloscope. Turn the knob so the average voltage on the DMM is around 1 V, and sketch the PWM signal in your lab notebook. Repeat this for a potentiometer voltage of 4 V.
19. **Demonstrate the working system to the TA, so you can be awarded points on your score sheet.**
20. Leave the potentiometer plugged into the breadboard for Part IV.

Part II: Measuring Temperature

Procedure

You will now construct the thermistor voltage divider circuit from the A4 calibration lab. The Arduino will read the analog voltage V_{out} from the transducer and use it to calculate the temperature via the voltage divider and Steinhart equations.

Important: You will use the Arduino to power the circuit on the breadboard. Do NOT turn on the breadboard.

1. Connect the Arduino UNO to the lab computer via the USB cable. You should see a green LED light up on the Arduino.
2. Use red and black jumper wires to connect the +5V and GND pins on the Arduino to the vertical bus lines on the breadboard, as shown in Fig. 1. Use the orange handheld DMM to verify that it is providing +5V of power.
3. Use the handheld DMM to verify that the thermistor works. It should have a resistance around 10kΩ at room temperature. The resistance should *decrease* when it is warmed up in your hand, because it has a *negative temperature coefficient* (NTC).
4. Take a 4.7k resistor out of its bin. Measure its resistance with the orange handheld DMM. Record the measured value for R_2 in your lab notebook.

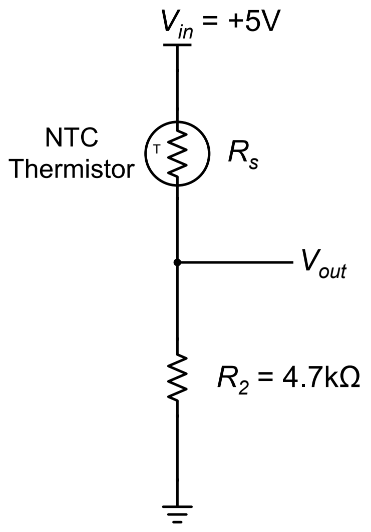

Figure 4 – A thermistor is wired up in a voltage divider circuit.

5. Construct the thermistor voltage divider circuit shown in Fig. 4 somewhere near the center of the breadboard. Test the circuit by measuring V_{out} relative to ground. The voltage V_{out} should increase when you warm up the thermistor in your hand.
6. Use a long jumper wire to connect V_{out} to the A0 analog input on the Arduino. The A0 analog input will read the voltage into the Arduino's memory as a 10-bit integer.

7. Open the Arduino IDE software on the lab computer.
8. Download the A8 "Part II Template" code template from the lab webpage. Read the comments and fill in the missing values for the calibration constants *A* and *B* and the measured resistance R_2. (Use the calibration constants you determined from the 2-point calibration in A4.)
9. Save the code with an intelligent file name (i.e. "A8_thermistor_yourName.ino").
10. In the Arduino IDE software, go to "Tools" > "Port" and select the COM port that says "(Arduino/Genuino Uno)" next to it.
11. Click the check mark at the top of the Arduino program to check the code for errors.
12. Press the arrow button to compile the program and send it to the Arduino.
13. Go to "Tools" > "Serial Monitor" (or press "Ctrl + Shift + M") to view the output from the "Serial.print()" commands at the bottom of the screen. You should see the measured temperature printed. Hold the thermistor tip in your hand. Does the printed temperature seem reasonable? How close is it to the temperature you reported in you A4 tech memo?
14. **Demonstrate the working system to the TA, so you can be awarded points on your score sheet.**
15. Leave the Thermistor circuit intact. You will need it for Parts III and IV.

Part III: LED Indicators

You will now use red and green LEDs to indicate whether the probe tip is hot or cold. Specifically, the micro controller will light up the Red LED if $T > 302$K, and light up the Green LED if $T < 302$K.

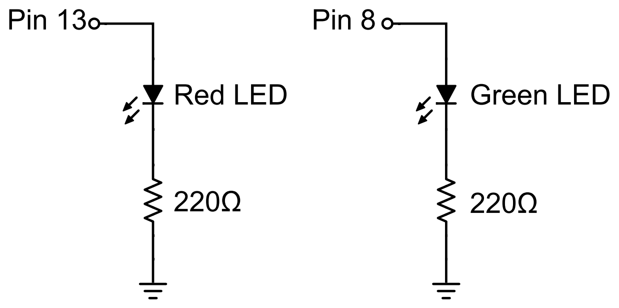

Figure 5 – Digital outputs from the Arduino are used to drive two different LEDs. The 220Ω resistors are used to limit the current and prevent voltage droop.

Procedure

1. Using the breadboard, connect the Red LED in series with a 220Ω resistor. Use a long jumper wire to connect digital output pin 13 to the LED, as shown in Fig. 5. The resistor should be connected to GND via the vertical bus line as shown in Figure 1.
2. Construct the Green LED circuit in a similar fashion. Use another long jumper wire to

connect digital output pin 8 to the LED, as shown in Fig. 5.

3. Download the A8 "Part III Template" code template from the lab webpage. Save it as "A8_ Part3_yourName.ino".
4. Look through the code. It is similar to the previous code, except there is an if-else conditional statement that lights up different LEDs depending on the temperature.
5. Replace the "***" in the code with appropriate information.
    a. In the variable declaration, enter the pin numbers that the Red and Green LEDs are connected to.
    b. Fill in the missing values for the calibration constants A and B and the measured resistance R_2. (Use the calibration constants you determined from the 2-point calibration in A4.)
    c. In the if-else conditional, enter the correct temperature threshold.
    d. In the if-else conditional, enter appropriate values of "HIGH" or "LOW" that will light up the Red LED if $T > 302$K, and light up the Green LED if $T < 302$K.
6. Test the program. Note that the LEDs must be connected in forward bias. If you connect one backwards, it will not light up!
7. **Demonstrate the working system to the TA, so you can be awarded points on your score sheet.** Only the green LED must be lit at ambient temperature, and only the red LED should light up when warmed up by your hand.
8. Return the lab bench to its initial state. Disconnect all of the wires, sensors, and LEDs. Close all your code windows and exit the Arduino IDE software.
9. Leave the Thermistor circuit intact. You will need it for Parts IV.

Part IV: Design Challenge

Combine Parts I and III to create a system where the threshold temperature can be adjusted by turning the potentiometer. That is, the threshold temperature for switching the Red and Green LEDs is no longer fixed at 302K, rather it can be adjusted by turning the knob. The system must have the following features:

- Use analogRead() to read in the potentiometer voltage. Map the 10-bit potentiometer reading to a the threshold temperature, such that turning the knob will adjust the threshold temperature between 295 to 310K.
- Use serial.print() and serial.println() to print the threshold temperature and measured temperature. Both should be printed side-by-side on the same new line for each iteration, separated by a comma with appropriate units.

When you have this working, demonstrate it to the TA. You must finish before the end of your lab section. (It is not the end of the world if you do not finish in time. This part of the lab is only 2 out of 16 points, and you will still get a B if you finish everything else.)

Data Analysis and Deliverables

You do NOT have to write a tech memo for this lab. Just make sure the TA fills out the score sheet as you complete each part of the lab.

Appendix A

Equipment

- Red and Green LEDs
- Small Potentiometer w/ breadboard pins
- 10k Vishay NTC thermistor NTCLE413E213F102L (Digikey part # BC2647-ND, black heat shrink on pins)
  OR
- 4.7k Vishay NTC thermistor NTCLE400E3472H (Digikey part # BC2466-ND, white heat shrink on pins)
- Arduino UNO Microcontroller
- 12" jumper wires with male pins
- 6ft USB cable
- Breadboard
- Extech Handheld Digital Multimeter
  - One red banana-to-banana cable
  - One black banana-to-grabber cable

Appendix B

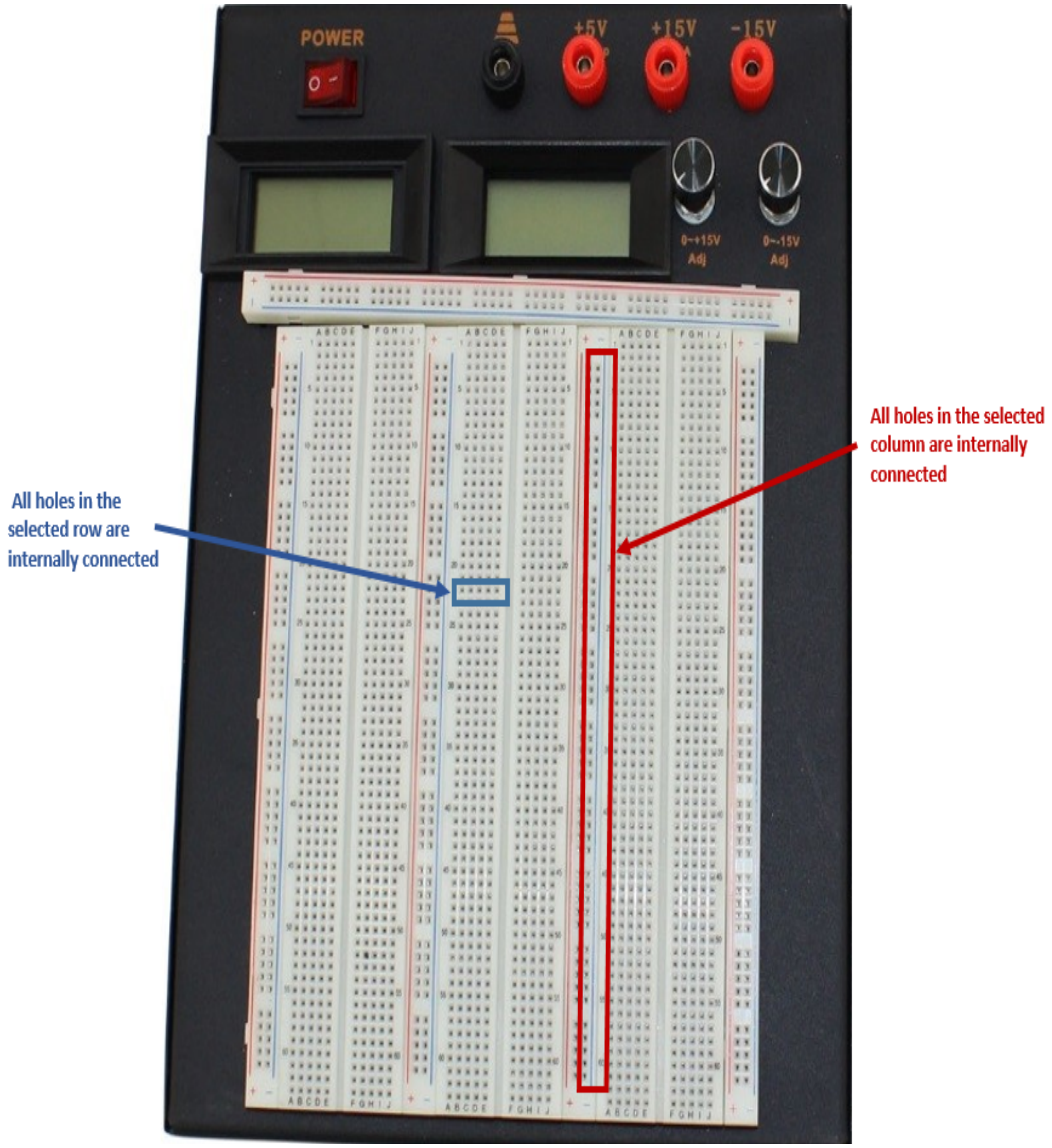
POWER
+5V
+15V
-15V
0~+15V Adj
0~-15V Adj
All holes in the selected column are internally connected
All holes in the selected row are internally connected

Figure 6 – Shown in blue, any 5 holes in a horizontal row are electrically connected, but they are NOT connected to the adjacent row of 5. Shown in red, all 50 holes in any vertical column or "bus bar" are electrically connected.