Experiment A7
Microcontrollers
Procedure

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

Recommended Reading: Chapters 2, 9, 16, and 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 an analog pressure sensor, calculate the pressure and airspeed, and display the values to an LCD screen. A digital push button will be implemented to toggle the units of pressure and airspeed.

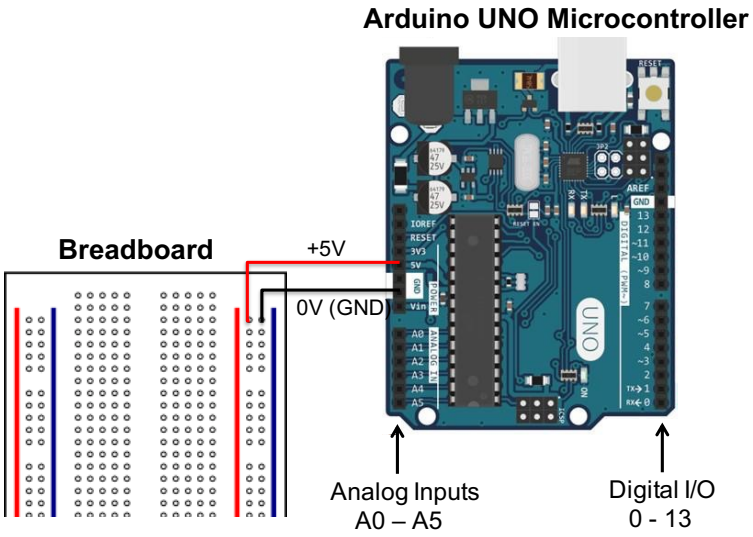

Figure 1 – The Arduino UNO Microcontroller works in tandem with breadboard.

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. an analog pressure sensor or a digital push button).

Here are a few of the more salient features of the Arduino 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 (I/O)** - The UNO has 14 different digital input/outputs (I/O). These pins always output a voltage of 0V (LOW) or +5V (HIGH). 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.)

Part I: Digital Flip/Flop

Background

In this part of the lab, a push button will be used to toggle two separate LEDs between ON and OFF. The digitalRead() function will be used to determine when the button is pushed, and the digitalWrite() function will be used to light up either a yellow or a green LED.

digitalRead() – This function reads a voltage input, and returns a single bit as either HIGH or LOW. The bit is HIGH if the voltage is greater than 1.5V, and LOW if the voltage is less than 1.5V.

digitalWrite() – This function outputs a voltage that is either HIGH (5V) or LOW (0V).

Shown in Figure 2, the digital input pin 2 of the Arduino reads the voltage from the push button and stores it in memory as a single bit that is either HIGH or LOW.

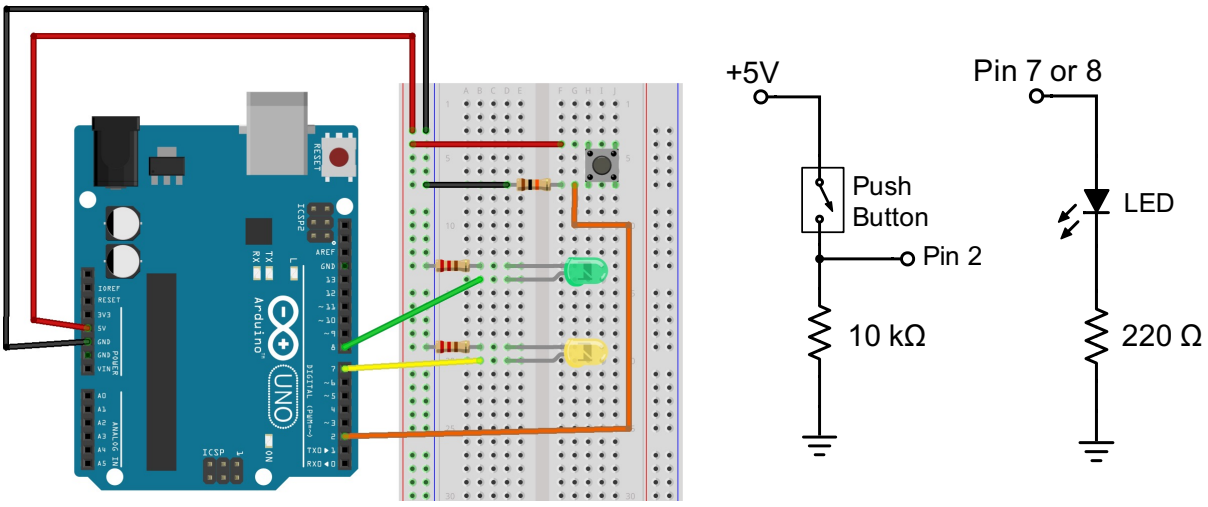

Figure 2 – A signal from a pushbutton is read into the Arduino via pin 2. Pushing the button toggles an internal state variable in the Arduino memory. The Arduino then illuminates either a yellow or green LED depending on the digital state.

Procedure

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. 2. Use the orange handheld DMM to verify that it is providing +5V of power.
3. Insert the push button into the breadboard such that its pins are in separate rows. Use a 10kΩ "pull-down" resistor to connect one pin to ground, and connect the other pin to 5V, as shown in Fig. 2.
4. Measure the voltage across the resistor. When you push the button, the voltage should go up to 5V (HIGH). When you release the button, the voltage should be pulled down to 0V (LOW).
5. Connect the output of the push button to digital I/O pin 2 on the Arduino.
6. Connect the Green LED in series with a 220 Ω resistor to ground. Connect the other end of the LED to digital I/O pin 8, as shown in Fig. 2. Make sure that the shorter "cathode" wire of the LED is on the ground (negative) side of the circuit.
7. Connect the Yellow LED in series with a 220 Ω resistor to ground. Connect the other end of the LED to digital I/O pin 7, as shown in Fig. 2.
8. Download the A7 "Part I Template" code template from the lab webpage, read the comments, and modify it.
    a. Right-click the link and "Save as…" with an intelligent file name (i.e. "A7_FlipFlop_yourName.ino").
    b. Open the file in the Arduino IDE software.
    c. Replace the *** in the beginning with the correct digital I/O pin numbers for the button and LEDs.
    d. Replace the *** in the if-then structure at the end with HIGH or LOW, so that it does what the comments say it should do.
9. Use the Arduino IDE software to compile the code and send it to the microcontroller.
    a. Go to "Tools" > "Board" and make sure either "Arduino/Genuino" or "Arduino UNO" is selected.
    b. Go to "Tools" > "Port" and select the COM port that says "(Arduino/Genuino Uno)" next to it.
    c. Click the check mark at the top of the Arduino program to check the code for errors.
    d. Press the arrow button to compile the program and send it to the Arduino.
    e. 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. (Make sure the Baud rate is set to 9600.)

10. Press the button on the breadboard. You should see the Green LED and the Yellow LED alternate between ON and OFF. Additionally, the serial monitor on the computer screen should update the status of the system each time the button is pressed.
11. **Demonstrate the working system to the TA, so you can be awarded points on your score sheet.**
12. Remove the LEDs and 220 Ω resistors from the breadboard. Leave the push button and 10 kΩ resistor.

Part II: Measuring Pressure

You will now connect an analog pressure sensor to the Arduino. The analog pressure transducer outputs a voltage V_{out} that is linearly related to the differential pressure p, such that $p = aV_{out} + b$, where a and b are calibration constants. The Arduino will read the analog voltage V_{out} from the sensor and use it to calculate pressure via the linear calibration formula.

analogRead() – This function reads a voltage input and stores it in the Arduino memory as an 8-bit integer. Voltages between 0V and 5V are mapped to integer values between 0 and 1023, respectively.

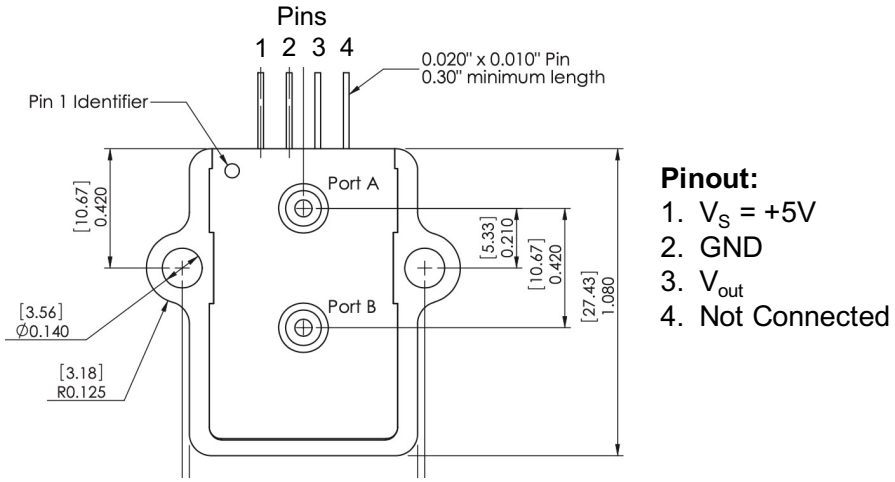

Figure 3 – The pinout for the capacitive pressure transducer shows where to connect the pins. Note that the pin numbers go left to right when looking at the side with the pressure ports with the "Pin 1 Identifier" in the upper left corner.

Procedure

1. Sketch the pin-out for the analog pressure sensor show in Fig. 3 in your lab notebook.
2. The calibration constants for the pressure sensor are a = 124.5 Pa/V and b = -273.9 Pa. Write these calibration constants down
3. Close the serial monitor window, and unplug the USB cable from the computer to power down the Arduino.

4. Carefully make the following connections between the analog pressure sensor, the breadboard, and Arduino:
    a. Purple wire → +5V on breadboard
    b. Blue wire → GND on breadboard
    c. Green wire → A0 analog input on the Arduino

Caution: Make sure these connections are correct. Connecting the sensor incorrectly will break it!

5. When you are sure the sensor is properly connect, power up the Arduino by plugging the USB cable back into the computer.
6. Download the A7 "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*. (Pressure and airspeed should increase with airflow. If not, switch the tubes on the pressure sensor.)
7. Save the code with an intelligent file name (i.e. "A7_pressure_sensor_yourName.ino").
8. In the Arduino IDE software, go to "Tools" > "Port" and select the COM port that says "(Arduino/Genuino Uno)" next to it.
9. Click the check mark at the top of the Arduino program to check the code for errors.
10. Press the arrow button to compile the program and send it to the Arduino.
11. Make sure the tubes are properly connected to the pitot probe, pressure sensor, flowmeter and nozzle. Check that the nozzle is aligned with the pitot probe. The probe should aimed and aligned with the nozzle centered, within 1 cm of the nozzle.
12. 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 pressure and airspeed printed.
13. Adjust the flow rate using the needle valve on the flowmeter. Do the printed pressure and airspeed values seem reasonable?
14. **Demonstrate the working system to the TA, so you can be awarded points on your score sheet.**
15. Leave the pressure sensor cables and tubing connected. You will need it for Parts III, IV, and V.

Part III: LCD Display – Hello World!

You will now implement an LCD display and test it by printing a simple "Hello World!" message.

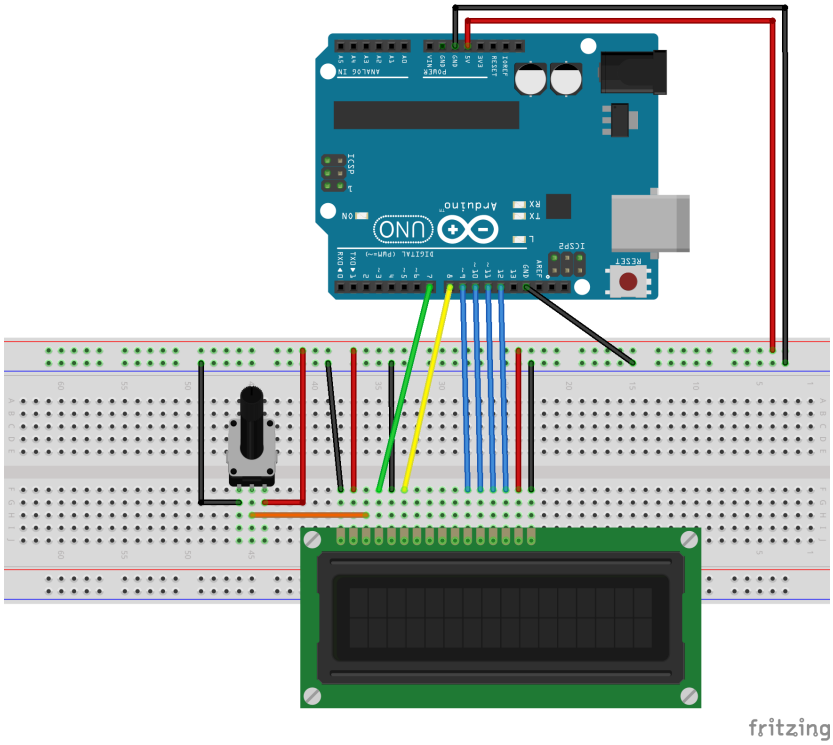

Figure 4 – A backlit LCD display is plugged into the breadboard and connected to the Arduino. The potentiometer knob on the left side of the circuit is used adjust the brightness of the screen.

Procedure

1. Close the serial monitor window, and unplug the USB cable from the computer to power down the Arduino.
2. Carefully plug the LCD display and potentiometer into the breadboard, as shown in Fig. 4.
3. Make the following connections with the LCD display:
   a. VSS → GND on breadboard
   b. VDD → +5V on breadboard
   c. V0 → Middle pin on potentiometer
   d. RS → Digital pin 7 on Arduino
   e. RW → GND on breadboard
   f. E → Digital pin 8 on Arduino
   g. D0, D1, D2, and D3 are NOT connected to anything
   h. D4 → Digital pin 9 on Arduino
   i. D5 → Digital pin 10 on Arduino

j. D6 → Digital pin 11 on Arduino
k. D7 → Digital pin 12 on Arduino
l. A → +5V on breadboard
m. K → GND on breadboard

4. Connect the left pin of the potentiometer to GND and the right pin to +5V, as shown in Fig. 4.
5. When you are confident that the electrical connections are all correct, plug the USB cable back into the computer to power up the circuit.
6. Download the A7 "Part III – Hello World!" code template from the lab webpage. Click the check mark at the top of the Arduino program to check the code for errors.
7. In the Arduino IDE software, go to "Tools" > "Port" and select the COM port that says "(Arduino/Genuino Uno)" next to it.
8. Press the arrow button to compile the program and send it to the Arduino.
9. Turn the potentiometer knob to adjust the screen brightness. You should see a message displayed on the screen. (If the screen is blank, try wiggling the potentiometer so it gets a better connection to the breadboard.)
10. **Demonstrate the working system to the TA, so you can be awarded points on your score sheet.**
11. Leave the circuit intact for the next part of the lab.

Part IV: Avionics Interface

You will now combine Parts II and III so that the LCD screen displays the measured pressure and airspeed.

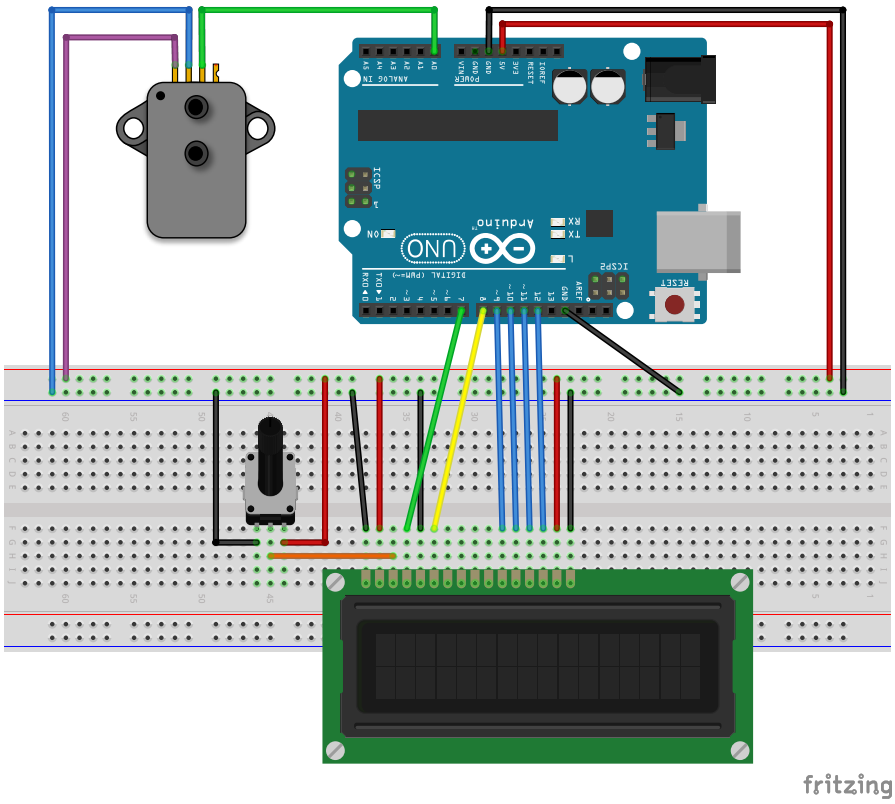

Figure 5 – The Arduino uses pin A0 to read the voltage from the pressure sensor. It then computes the pressure and airspeed, then prints them to the LCD display.

Procedure

1. Check that the pressure sensor is still properly connected to the a Arduino as it was in Part II. (Your circuit should look like Fig. 5.)
2. Download the A7 "Part IV Template" code template from the lab webpage. Read the comments and fill in the missing values for the calibration constants *a* and *b*.
3. Save the code with an intelligent file name (i.e. "A7_pressure_sensor_yourName.ino").
4. In the Arduino IDE software, go to "Tools" > "Port" and select the COM port that says "(Arduino/Genuino Uno)" next to it.
5. Click the check mark at the top of the Arduino program to check the code for errors.
6. Press the arrow button to compile the program and send it to the Arduino.
7. Make sure the tubes are properly connected to the pitot probe, pressure sensor, flowmeter and nozzle. Check that the nozzle is aligned with the pitot probe as it was in A4.
8. You should see the pressure and airspeed printed to the LCD display.

9. Adjust the flow rate using the needle valve on the flowmeter. Do the printed pressure and airspeed values seem reasonable? How close are they to the values you reported in you A4 tech memo?
10. **Demonstrate the working system to the TA, so you can be awarded points on your score sheet.**
11. Leave everything connected for Part V (if you have time).

Part V: Design Challenge

Combine Parts I and IV to create a system where the user can push the button to toggle the units on the display from Pa and m/s to in.H2O and mph (miles per hour).

Please consider the following:

- Look up the conversion factors to go from Pa to in.H2O and from m/s to mph. In your code, create new variables for these parameters and compute them from the original variables using the appropriate conversion formulas.
- The push-button should function similar to Part I. However, instead of lighting up different LEDs, it will switch between different lcd.print() commands that print the pressure and airspeed in different units.
- The user should hold down the push button for half a second before the display updates.

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 3 out of 20 points, and you will still get a B if you completed everything else.)

Clean-up

To receive full credit, you must return the lab bench to its initial state:

- Disassemble your circuit. Place the wires, push button, LEDs, breadboard, and LCD display back in the plastic bag.
- Disconnect the USB cable from the computer.
- Leave the tubes connected to the pressure sensor.
- Leave the pitot probe and nozzle mounted in the beaker clamps.

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

- Arduino UNO Microcontroller
- 6ft USB cable
- Bag containing:
  - Small breadboard
  - Jumper wires with breadboard pins
  - Push button
  - Yellow and Green LEDs (from Elegoo kit)
  - Small Potentiometer w/ breadboard pins, 10k, blue
  - LCD Display (from Elegoo kit)
- Analog pressure sensor - All Sensors 1 INCH-D-4V (Digikey Part #: 442-1012-ND)
- Pitot probe with 2 tubes (1 ft long each)
- Rotameter (0 – 24,000 L/min)
  - Long tube with quick-connect going to drop-down compressed air
  - Short tube with brass nozzle
- Lab stands with two beaker clamps

Appendix B

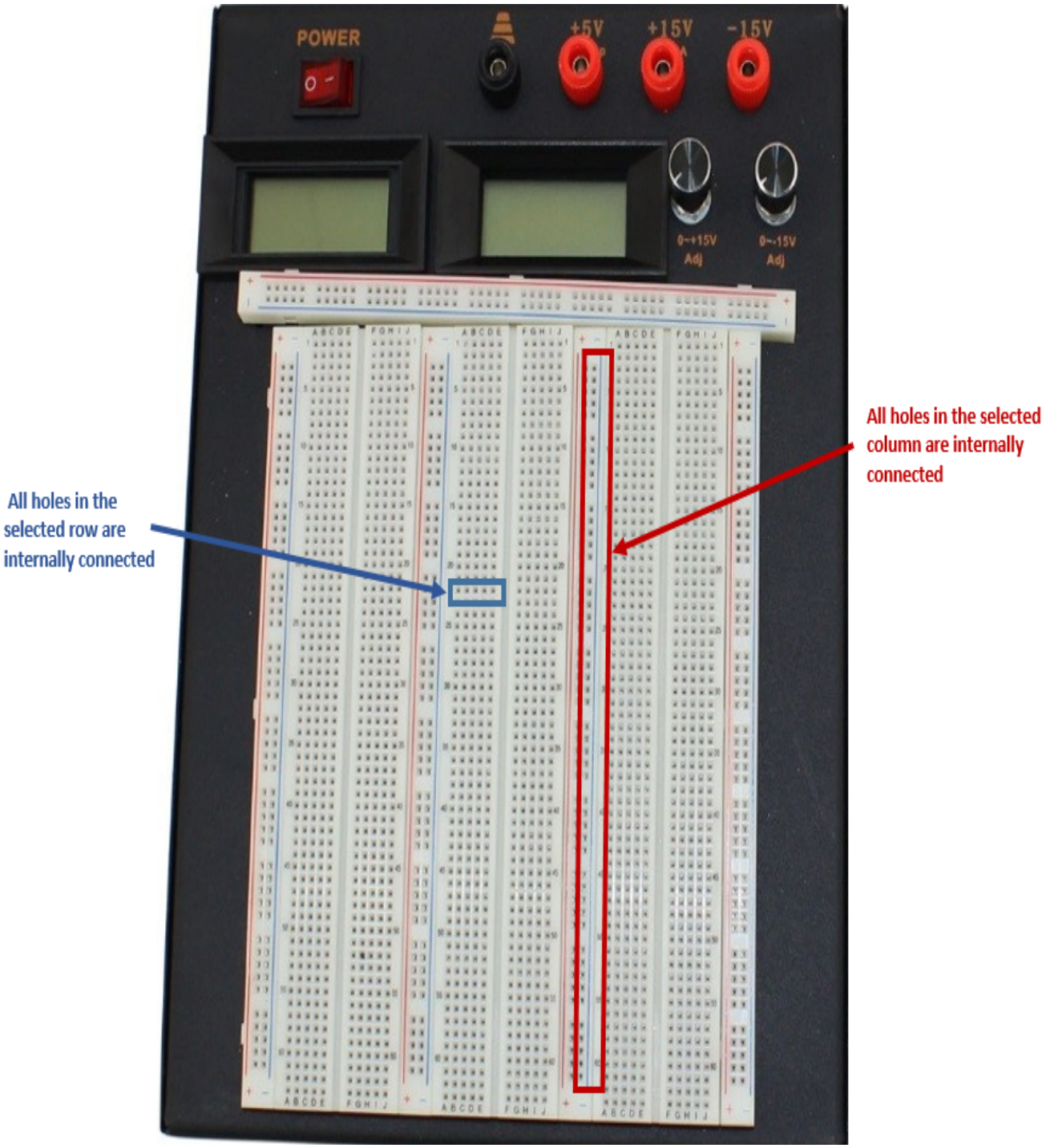

Figure 6 – Shown in blue, any 5 holes in a horizontal row are electrically connected, but they are NOT connected to the adjacent row of 5. Shown in red, all 50 holes in any vertical column or "bus bar" are electrically connected.

Appendix C