School of Engineering, Discipline of Mechanics and Mechatronics, pp. 1–10

Lab 1: Sensors
Digital I/O, ADC and Serial Communication

MECH2110

Semester 1 2019

Nicholas O'Dell, Johannes Hendriks, Craig Wheeler.

---

1 Introduction

In this lab, you will use the Arduino to acquire data from several types of sensors that may be useful in your design project. These sensors are:

Figure 1: Potentiometer

Figure 2: Ultrasonic Transducer

Figure 3: Momentary Pushbutton

The lab is worth 2% of your course grade and is graded from 0-2 marks.

2 Plugging in

The Arduino is programmed through the USB port of the computer using the blue cable included.

1) Plug the Arduino into the computer using the white USB cable.
2) If this is the first time someone has used an Arduino on that computer this semester you will have to wait until the driver has installed then unplug the Arduino. (This should take about a minute and is necessary because of the method used to run the Arduino software on the university computers.)
3) Open **Arduino** by selecting **Start** → **All Apps** → **Arduino**
4) Select **Tools** → **Port** and note the number of each port that appears.
5) Plug in the Arduino, go back to **Tools** → **Port** and connect to the new port which should appear when the Arduino is plugged in the second time. It may take up to 20 seconds or so to appear after plugging in the Arduino.
6) Select **Tools** → **Port** the default is probably `COM1`, select the `COM` port that just appeared. This selects the port that the computer uses to communicate to the Arduino.

   Hint
   The USB ports on the left of the lab PCs seem to consistently be `COM3` and `COM4`

7) It is important to select the correct board for the compiler, this provides information such as available program memory, pin configuration etc. To do this select **Tools** → **Board** → **Arduino/Genuino Uno**.

3 Creating a new project

1) Open **Arduino** and create a new project by selecting **File** → **New** (or press **Ctrl** + **N**). Save this project into a directory to which you have write permission to (e.g., home folder, U: drive, USB drive etc.).
2) It is important to select the correct board for the compiler; this provides information such as available program memory, pin configuration etc. To do this, select **Tools** → **Board** → **Arduino/Genduino Uno**.
3) Connect to the COM port that you found previously. Select **Tools** → **Port** → **COMxx (Arduino/Genduino Uno)**.

Recommended: Become familiar with the **Arduino Language Reference** page outlining the use of the Arduino specific functions. https://www.Arduino.cc/en/Reference/HomePage (Think of this like `>>help` in Matlab.)

4 Analog to Digital Converter (ADC), Potentiometer (1 mark)

Task: Obtain a voltage reading from an analog pin, and change the state of an LED accordingly.

An analog sensor varies a voltage depending on its input. For example a potentiometer is a resistor that varies the voltage at the center pin between the reference voltages connected to each of the other pins (voltage divider). Other examples include microphones, moisture sensors, IR sensors, and Light Dependant Resistors (LDR).

The Arduino has a number of analog inputs, The Arduino Uno has 6 (**A0-A5**), that convert an analog voltage to a number (10 bit unsigned integer) ranging between 0 and 1023. A result of 0 indicates a voltage of 0 volts and a result of 1023 indicates a voltage of 5 volts when the function `analogRead(pin)` is called (https://www.Arduino.cc/en/Reference/AnalogRead).

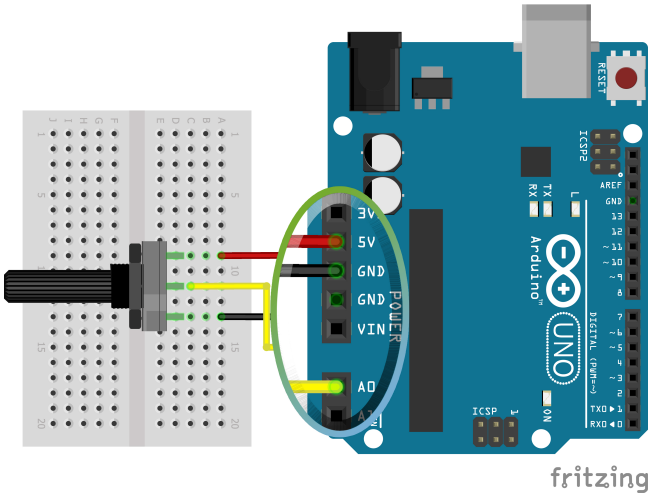

Figure 4: Potentiometer circuit

Figure 5 shows how the rails of a bread board are connected. The power running down the edges

of the board are connected along the whole length and all other pins are connected in groups of 5, running across the board.

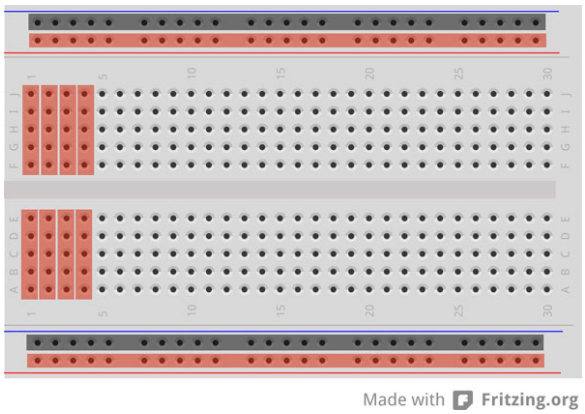

Figure 5: Breadboard connections

1) Build the circuit shown in Figure 4.
2) Complete the **TO DO** statements in the code shown in Listing 1. This code template and all others within this lab are available on Blackboard in `lab1_files.zip`.

Listing 1: Potentiometer Code

```
/*
Lab 1 MECH2110
Analog to Digital Conversion, Potentiometer
*/
#define pot A0

void setup() {
        //start serial connection
        Serial.begin(9600);
        //TODO: configure A0 as an input (NOT INPUT_PULLUP!)
        //TODO: Initialise pin 13 (LED) as an output.
}

//global variables to store reading
int reading =0;
float voltage = 0;

void loop() {
        delay(500);
        //TODO: Use analogRead to read the potentiometer
        //      Print the conversion to the serial monitor
        Serial.print("reading␣=");
        Serial.println(reading);
        //TODO: Convert Reading to voltage
        //      Print the voltage to the serial monitor

        //TODO: If the voltage is less than 1 Volt turn on the LED,
        //      otherwise turn it off.
}
```

i) Initialise pin 13 (LED) as an output (https://www.arduino.cc/reference/en/language/functions/digital-io/pinmode/).

ii) Read the conversion result from the potentiometer pin, store it in `reading` and print it to the serial monitor (https://www.Arduino.cc/en/Reference/analogRead).

iii) Open the serial monitor (**Ctrl + Shift + M**) and confirm the result is being printed every 100 milliseconds.

iv) Convert the result (0-1023) to a voltage between 0 and 5 volts, store it in `voltage` and print it to the serial monitor.

v) At this point run your program to make sure that your voltage conversion is correct.

> **Hint**
>
> Ensure floating point maths is used, append the arguments of your division with '.0', e.g. $1/2 \Rightarrow 1.0/2.0$.

vi) Use an `if` statement to turn the LED on if the voltage read is less that 1 volt and demonstrate this to your tutor.

- `digitalWrite`: https://www.Arduino.cc/en/Reference/digitalWrite
- `if`: https://www.Arduino.cc/en/reference/if

5 Ultrasonic distance sensor, Advanced I/O (1 mark)

Task: Obtain a distance measurement using the ultrasonic transducer.

A useful sensor for navigating a course such as Warman is a "distance sensor" for non-contact measurement. There are two basic sensors available for this application; IR distance sensors and ultrasonic distance sensor. An ultrasonic transducer sends a pulse and then it records the response time between sending the pulse and receiving the echo. The distance can then be calculated given the speed of sound. A timing diagram of the HC-SR04 which is the sensor being used, is shown below in Figure 6.

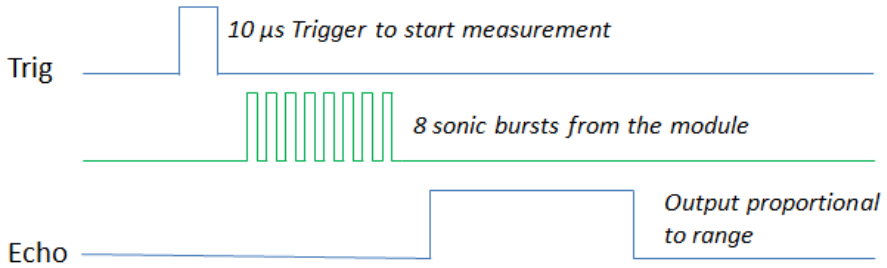

Figure 6: HC-SR04 timing diagram obtained from data sheet (http://www.micropik.com/PDF/HCSR04.pdf).

In this section of the lab, you will activate an ultrasonic sensor every 100 ms and convert the timed pulse into a distance.

1) Build the circuit shown in Figure 7.

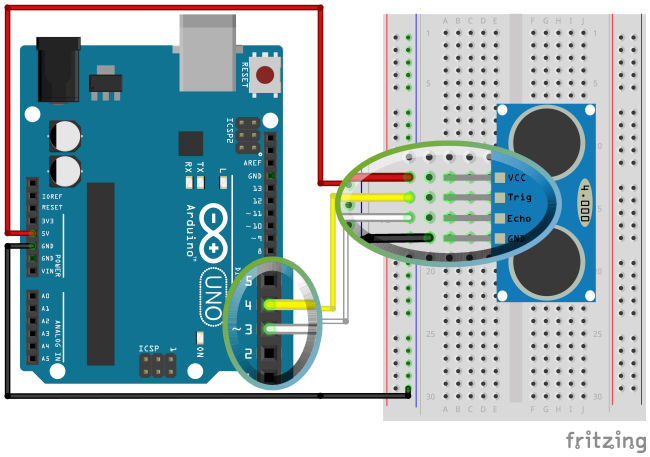

Figure 7: Distance sensor connection

2) Begin with the code shown in Listing 2 and complete the **TO DO** statements.

Listing 2: Ultrasonic distance sensor code

```
/*
Lab 1 MECH2110
Ultrasonic Sensor Task
*/
#define TRIG 4
#define ECHO 3


#define SPEED_OF_SOUND 340.29 // m/s

void setup() {
        // initialize serial communication:
        Serial.begin(9600);
        //TODO: initialize sonar sensor pins
}

// establish variables for duration of the ping,
// and the distance in mm
long duration;
float mm;

void loop() {
        // The PING))) is triggered by a HIGH pulse of 10 microseconds,
        // for the HC-SR04 sensor

        //pulse the trigger pin high for 10 microseconds
        digitalWrite(TRIG, HIGH);
        delayMicroseconds(10);
        digitalWrite(TRIG, LOW);

        //TODO: Read the pulse duration into the variable duration
```

```
        //TODO: Convert the time into a distance in mm and store in mm,

        //print result
        Serial.print(mm);
        Serial.println("mm");

        delay(100);
}
```

i) Initialise the `TRIG` and `ECHO` as `OUTPUT` and `INPUT` respectively (https://www.arduino.cc/reference/en/language/functions/digital-io/pinmode/).

ii) Immediately after the pulse is sent on the trigger pin, wait for the response pulse on the echo pin and store its width.

> **Hint**
>
> Arduino has a built in function for measuring pulse width called `pulseIn(pin,HIGH,LOW);` , e.g., `pulseIn(7,HIGH);` waits for a high pulse on pin 7 and returns its width in microseconds (see https://www.Arduino.cc/en/Reference/PulseIn).

iii) Convert this duration, which is in microseconds to a distance in millimetres, given that the pulse travels at the speed of sound and print it to the serial monitor. Demonstrate this to your tutor.

> **Hint**
>
> The pulse width is the time taken to travel the distance from the sensor **and** back again.

Which one of the following conversions from pulse width in microseconds to distance in mm is correct in the '**C**', and why?

- `mm = duration * 1e-6 * 1e3 * SPEED_OF_SOUND;`
- `mm = duration * 10^(-3) * SPEED_OF_SOUND;`
- `mm = 0.5 * duration * 1e-6 * 1e3 * SPEED_OF_SOUND;`
- `mm = 1/2 * duration/1000000 * 1000 * SPEED_OF_SOUND;`

Note for Warman: The ultrasonic sensor used in this lab is not minimalist. Its on board electronics are designed to output a clean signal on the echo pin; without this more complex software is needed to operate the sensor.

6 Digital input, momentary button (non-assessed)

Task: To change the state of an LED based on digital input from a button.

Digital input is used to read an on-off state from a digital sensor such as a button, switch, encoder or any other sensor that inputs a high or low signal to the micro-controller. In this section of the lab you will use a momentary push button to change the state of the built-in LED.

1) Build the circuit shown in Figure 8 by connecting one pin to **GND** and the other to pin **2** of the Arduino.

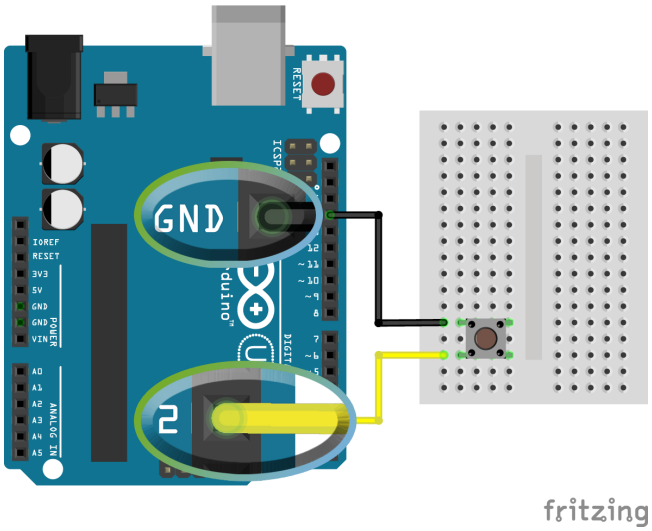

Figure 8: Button circuit

Listing 3: Digital Input

```
/*
Lab 1 MECH2110
Digital Input, Push Button
*/

void setup() {
  //start serial connection
  Serial.begin(9600);
  //configure I/O
  //TODO: Initialise pin  2 (button) as input_pullup
  //TODO: Initialise pin 13 (LED) as an output.
}

void loop() {

  delay(200);
  //read the pushbutton value into a variable
  int sensorVal = digitalRead(2); //returns either 0 or 1
  //print out the value of the pushbutton
  Serial.println(sensorVal);
  //TODO: Add some code here to write the value read from pin 2 to the LED.
}
```

2) Copy the above code into your project and complete the **TO DO** statements above.

   i) Using the function `pinMode` (https://www.Arduino.cc/en/Reference/PinMode), initialise the LED pin **(13)** as `OUTPUT` and the pin **2** as `INPUT_PULLUP`.

Note: The following figure demonstrates how a pull-up resistor, whether internal or external, works.

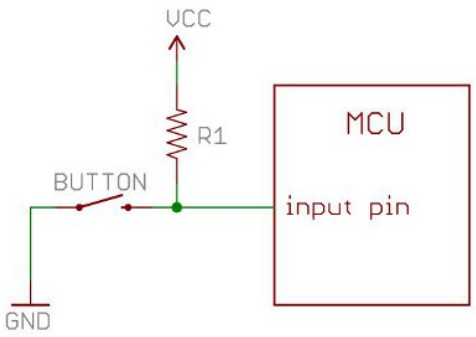

Figure 9: Pull-up resistor, for more information see:
https://learn.sparkfun.com/tutorials/pull-up-resistors

   ii) Open the serial monitor (**Ctrl + Shift + M**) and check that it's displaying 0 when the button is depressed, and 1 otherwise.

   iii) Use `digitalWrite()` to write the value read from the button to the LED.

7 Recommendations

- **Important:** It is highly recommended that you attempt next week's lab exercises, Lab 2, before the lab. To do this at home or on any computer, consider using an online Arduino simulator such as Autodesk Circuits, https://circuits.io/, which provides a graphical environment for you to test your code on a virtual Arduino and includes features such as a virtual oscilloscope.
- Think about the sensors used here in the lab and how you could use them in your Warman design.
- Another distance/proximity sensor to use is an infrared proximity sensor, e.g., Sharp GP2Y0A21YK, which is harder to calculate a distance with but is more accurate and does not emit such a wide beam as the ultrasonic sensor.
- Consider other available hobby sensors and how you might use them. Try googling Arduino sensor kit and look at the range of sensors available, and how they can be used with your mechanical design.
- One sensor that can be used to detect colour is called the *pixy*, and an example of its use can be found here: https://create.Arduino.cc/projecthub/niftyjoeman/pixy-cmucam-arduino-3ac141,

https://www.youtube.com/watch?v=DV4YK_Kk5IY. This advanced sensor returns the location of specific coloured objects.

- Another nifty sensor incorporating a proximity sensor, gesture detection and colour sensing is the APDS-9960 (https://www.sparkfun.com/products/12787).
- Experiment with interrupts as opposed to using `pulseIn(pin,HIGH,LOW)` for an ultrasonic distance sensor (http://www.instructables.com/id/Non-blocking-Ultrasonic-Sensor-for-Arduino/).