# DIG5118 Lab Session 3

## Aim

To understand and programmatically control the analogue and digital inputs of the Arduino board.

## Equipment

You will need the following items to complete this lab session

* Arduino Uno Rev3 (Classic Family)
* USB A to USB B cable
* Breadboard
* Resistors 220 or 240 Ohm
* Resistors for pulldown 10K Ohm
* Rotary Potentiometers
* Push button switches
* LEDs
* Jumper wires

## Controlling the inputs

From the last lab session, we know how to control the outputs of our Arduino using **digitalWrite**. However, we really want the Arduino to respond to external inputs. This allows us to interact and control the functionality. All our Arduino digital pins can be configured as either outputs or **inputs.** We determine this using the **pinMode** function in setup.

**pinMode(pinNumber, direction):** Set the pin specified in argument 1 to either INPUT or OUTPUT

With a pin set as the INPUT, we can read what is happening by reading from the pin. Our digital pins are binary, so they return a TRUE/FALSE or HIGH/LOW representation. The Arduino function to read a digital pin is **digitalRead**

**digitalRead(pinNumber):** Return the current binary state of the pin as ON/OFF, HIGH/LOW or TRUE/FALSE (all the same).

Diagram, schematic

Description automatically generated

Figure 1 circuit to connect switch with pull down resistor

<https://docs.arduino.cc/built-in-examples/digital/Button>

Figure *1* shows the configuration of a switch (button) connection to the Arduino board. We can expand this circuit with a LED to use button to control the LED. Below is the reference circuit and construct the circuit and finish the following task in the section 1.3.1.

A close-up of a circuit board

Description automatically generated with low confidence

### Task 1: Controlling the internal LED

1. Make the LED turn on when the button is pressed and held. Releasing the button turns off the LED (code golf version too!)
2. Using our blink function, make the LED blink when our button is pressed.
3. Have the LED turn on when the button is pressed. Pressing the button again turns off the LED

## Using the serial console

Our Arduino’s can also talk with the computer, over a serial connection. On the Arduino Uno boards, this is on Pins 0 and 1, and the USB interface.

Graphical user interface, text, application

Description automatically generated

Figure 2 Open Serial interface tools

By using this interface, we can allow our Arduino to communicate with us in real time.

Firstly, we need to tell the Arduino to use the Serial interface using the Serial library begin function

**Serial.begin(speed):** Open the Serial port with the given baud rate. For example, serial.begin(9600);

serial.begin(38400);

Serial.begin(115200);

When we have our Serial rate, we can print information to the console using Serial.print

**Serial.print(val):** Send a string to the port

**Serial.println(val):** Send a string to the port and terminate with a new-line character

### Task 2:

1. Have the Arduino print you a message when it starts up!
2. When the button is pressed, let it send you a message
3. Have the Arduino tell you the button state every few seconds
4. Have the Arduino tell you if the button was pressed in the previous 5 seconds

## Serial Plotter

Next to the Serial Monitor in the Arduino IDE is the serial plotter. This works much the same way as the Serial Monitor but is designed to plot a value on a time-value graph.

We can use this to help debug aspects of our program which are signal, or time-value based.

We still use print here but need to send numbers only!

(You may find it easier to use a fresh script)

### Task 3:

1. Have the Arduino send a 1 when the button is pressed, and a 0 when the button is released
2. Build a clicker! Each time the button is pressed, add one to the sent value!
3. Send a random number each time the button is pressed
4. Send a sinewave!

## Potentiometer and analogue inputs (optional work this week)

You can combine the potentiometer with PWM output to using the rotary knob to control the brightness of the LED. Try to add a potentiometer to your circuit using the below reference circuit to make it work. Hint, you need to think about how to read the input from A0 pin and use that value to control the output of PWM pin D6.

Diagram, schematic

Description automatically generated

Figure 3 Connect external LED to digital PWM pin

1. You can build the code on base of Lab2 task2.2 to start with.
2. You can use Serial.print to inspect the values.
3. <https://www.arduino.cc/reference/en/language/functions/math/map/>
4. <https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/>