ME381 ROBOTICS

Experiment 1: Basics of Microcontroller

Aim: Learn about the layout and basic input/output capabilities of Arduino Uno microcontroller.

This module is aimed at getting a brief introduction about the Arduino Uno microcontroller and learning basic input and output operations. Starting off from the basic board layout, we will learn about various pins, buttons and components available on an Arduino Uno microcontroller. This is successively followed by a number of small experiments to be performed involving functioning of various pins, reading digital and analog values from the pins, and writing digital and analog values to the pins. The following table shows in brief the tasks to be performed in this lab. The details of these tasks and reporting for evaluation is given in the 'Tasks' section.

Sr. No.	Experiment name	Lab work to be performed in brief
1	Basics of microcontrollers	1. Know the functioning of different types of pins 2. Digital read and write 3. Analog read and PWM (analog write)

Tasks:

- Each student in the group should perform all the tasks listed in the table below. The reported outputs obtained after performing each of these tasks should be mentioned in their report.
- Each student should get at least one task individually assigned to him/her by the TA. He/she is supposed to extensively present all the results for that task in his/her report such as taking snapshot of the circuit and snapshot of the serial monitor or any other output device.

Task no.	Topic	Task	Report
1	Introduction	Get to know these pins: IOREF, RESET, GND, 5V, 3.3V, Vin, A0-A5, Aref, Rx0-Tx1, PWM pins, and Built-in LED pin.	None
2	Digital Read	Use the provided program for Task 2 to digital-read the pin 8 and print it on the serial monitor. Connect the following pins to pin 8 and print the result on the serial monitor: none (open), GND, RESET, IOREF, AREF, 3.3V, 5V.	2(a) Report the values observed at serial monitor for each connection to pin 8. 2(b) What happens to pin 8 as observed on serial monitor when it is (i) Connected to GND once and then left unconnected (ii) Connected to 5V once and then left unconnected
3	Digital Read	Use the provided program for Task 3 to turn on the built-in LED only if pin 8 is LOW (connected to GND) and turn off otherwise.	Is the LED turning on as soon as a logic LOW is applied to pin 8? Is it turning off if the logic LOW is removed from

		(in the code, the command `pinMode(8, INPUT_PULLUP)` maintains the pin 8 at HIGH when left unconnected)	pin 8 and it remains unconnected?
4	Digital Read	Introduce a five-second delay in the loop of the last task.	Does the LED turn ON immediately when pin 8 is given a LOW signal? If not, why?
5	Digital Write	Write a code to alternatively switch pin13 (LED) to HIGH and LOW with a delay of 1 second in between.	Read the voltage of pin 13 using a multimeter and report the same when (i) LED is on (i) LED is off
6	Analog Write (PWM)	Write a code to apply a PWM signal on pin 9. Read the voltage at pin 9 using a multimeter. Tweak the argument inside analogWrite() function in the range 0-255 and re-upload the code. Read the voltage again on the multimeter.	Report 2 such argument values used by you inside analogWrite() and the voltage read on the multimeter for each of these arguments.
7	Analog Read	Measure the voltage of the Arduino 5V pin using a multimeter. Let it be V1. Use the provided code to Analog-read pin A1. Connect pin A1 one by one to: nothing (open), GND, 3.3V, 5V, Vin, AREF, and IOREF pins. For each such connection, map the analog read value (0-1023) to 0 to V1 and display it on the serial monitor in volts.	7(a) Report V1. 7(b) Report the readings (in volts) obtained by connecting pin A1 to: nothing (open), GND, 3.3V, 5V, Vin, AREF, and IOREF pins. 7(c) Report the reading (in volts) when pin A1 is left unconnected. Is it constant everytime?
8	Analog Read	Using the provided code for Task 8, connect pin A1 to PWM output pin 9. Analog value corresponding to 30% duty cycle is already written on pin 9. Change the duty cycle to other values such as 60%, 85% in the code. Observe the changes on Serial Plotter based on the values read at pin A1.	Describe the pattern observed on the Serial Plotter for the PWM duty cycle of 60% and 85%.

Task manual:

1. **Introduction:** Get to know these pins: IOREF, RESET, GND, 5V, 3,3V, Vin, A0-A5, Aref, Rx0-Tx1, PWM pins, and Built-in LED pin.
   Vin: This is the input voltage pin of the Arduino board used to provide input supply from an external power source.

   5V: This pin of the Arduino board is used as a regulated power supply voltage and it is used to give supply to the board as well as onboard components and other external low current devices.

   3.3V: This pin of the board is used to provide a supply of 3.3V which is generated from a voltage regulator on the board

GND: This pin of the board is used to ground the Arduino board.

Reset: This pin of the board is used to reset the microcontroller. An external device or shield can bring this line LOW to reset the microcontroller.

Analog Pins: The pins A0 to A5 are used as analog inputs and they operate in the input range of 0-5V by default.

Digital Pins: The pins 0 to 13 are used as a digital input or output for the Arduino board.

Serial Pins: These pins are also known as a UART pins. It is used for communication between the Arduino board and a computer or any other device. The transmitter pin number 1 and receiver pin number 0 are used to transmit and receive the data.

PWM Pins: These pins of the board are used to convert the digital signal into an equivalent analog value by varying the duty cycle of Pulse Width Modulated (PWM) wave. The pin numbers 3,5,6,9,10 and 11 are used as a PWM pins.

LED Pin: The board has an inbuilt LED using digital pin-13. The LED glows only when the digital pin becomes high.

AREF Pin: The AREF (Analog Reference) pin can be used to provide an external reference voltage for the analog-to-digital conversion of inputs to the analog pins. The reference voltage essentially specifies the value for the top of the input range, and consequently each discrete step in the converted output. The function *analogReference(EXTERNAL)* makes the voltage applied to the AREF pin (0 V to 5 V only) to be used as the reference.

IOREF Pin: This is a voltage corresponding to the input/output logic levels of that board, for example, an 'Arduino Uno' would supply 5 V to this pin, but an 'Arduino Due' would supply 3.3 V. Sending a signal to this pin does nothing. A properly configured external shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5 V or 3.3 V.

External Interrupt Pins: These pins of the Arduino board can be used to produce external interrupt and it is done by pin numbers 2 and 3. Interrupt is a process by which arduino stops its regular task and go to interrupt function and executes it. After initialization of external interrupt if there is any change in signal in the interrupt pin, then that will create external interrupt.

SPI Pins: These are the Serial Peripheral Interface pins which are used to implement the SPI communication protocol with other devices or peripherals. The protocol is generally implemented using the <SPI.h> library or sometimes the library provided by the manufacturer of the external device to be communicated with. SPI pins include:

1. SS: Pin number 10 is used as a Slave Select
2. MOSI: Pin number 11 is used as a Master Out Slave In
3. MISO: Pin number 12 is used as a Master In Slave Out
4. SCK: Pin number 13 is used as a Serial Clock

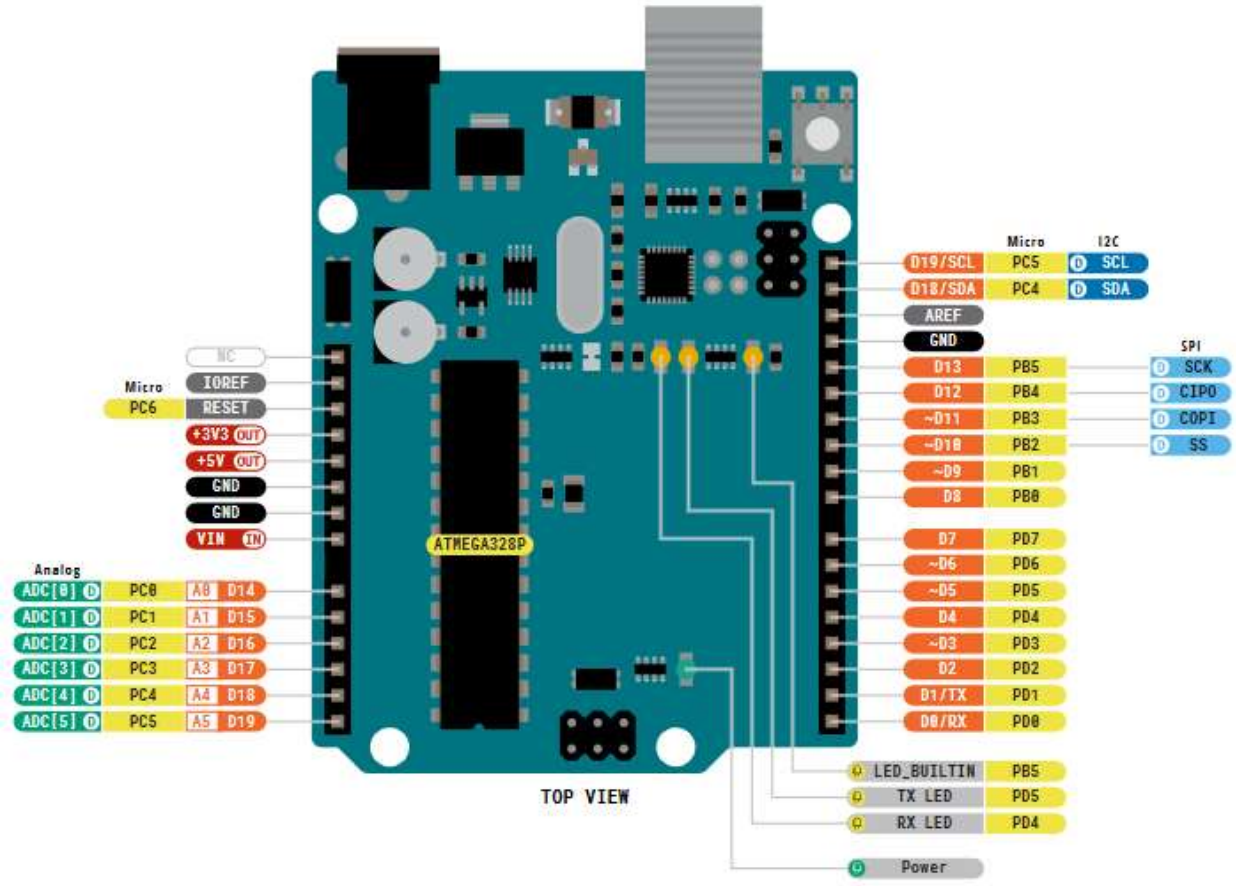

Figure 1 Arduino Pinout (Image Source: https://docs.arduino.cc/resources/pinouts/A000066-full-pinout.pdf)

2. **Digital Read**: Open the Arduino IDE (version 2.3.2). Connect the Arduino to the laptop using the cable shown below.

Figure 2 Arduino Uno Cable

Select the Arduino Board in the IDE using the Menu bar by clicking on:

Tools→Board→Arduino AVR Boards→Arduino Uno

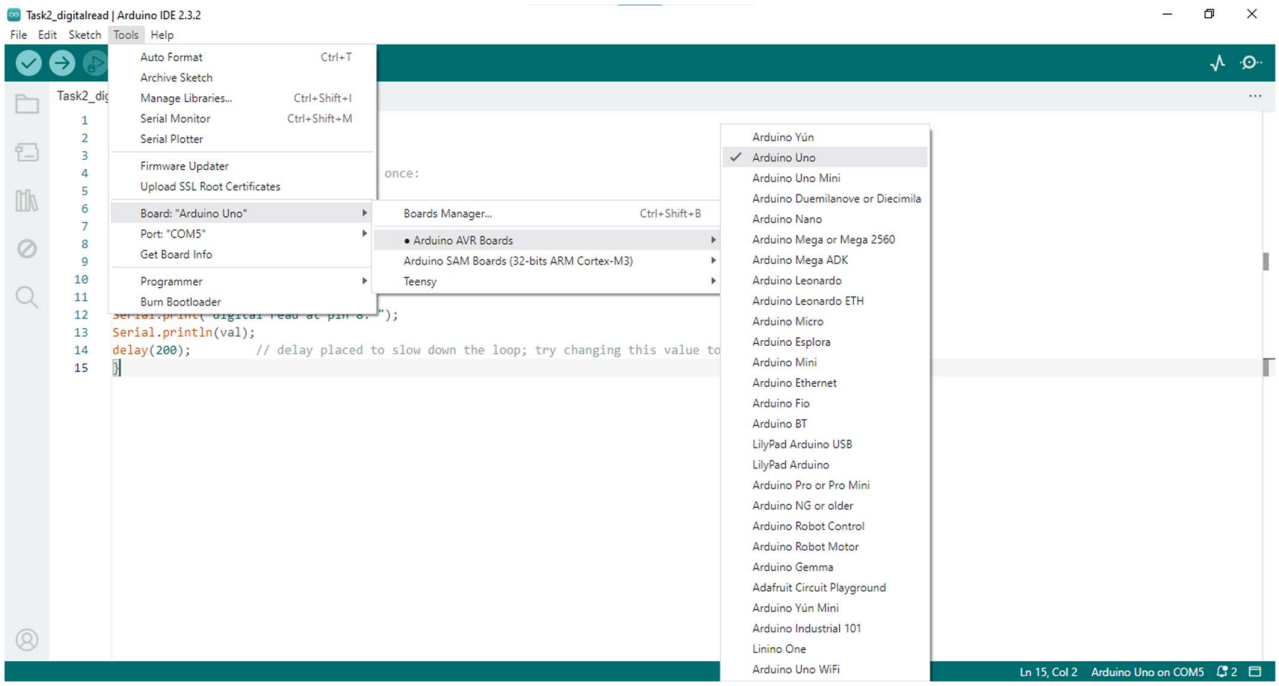

Figure 3 Selecting Board "Arduino Uno" in Arduino IDE

Select the COM port using the Menu bar by clicking on:

Tools→Port→COM port showing Arduino Uno

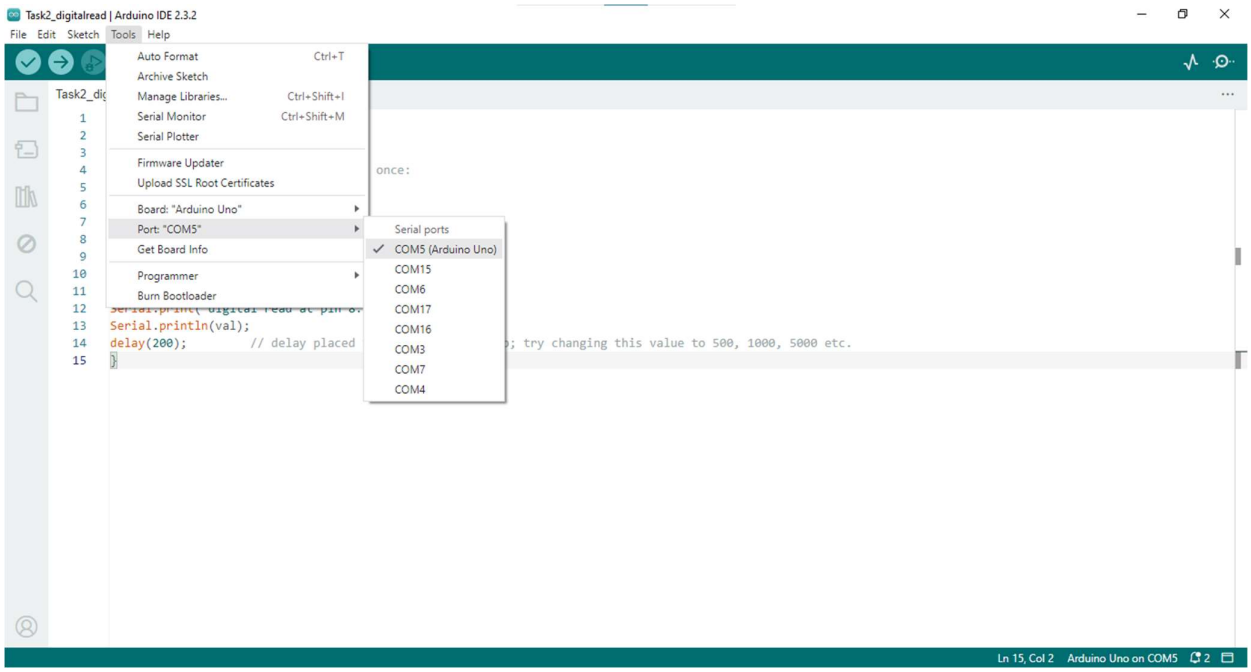

Figure 4 Selecting COM port in Arduino IDE

Upload the corresponding Arduino code for Task 2 (codes provided to you have Task number as their initial to help you figure out which code to be uploaded for a given task). Make the following circuits one by one keeping the same code already uploaded for Task 2. Observe the outputs at the serial monitor (serial monitor can be opened by clicking on Tools→Serial Monitor).

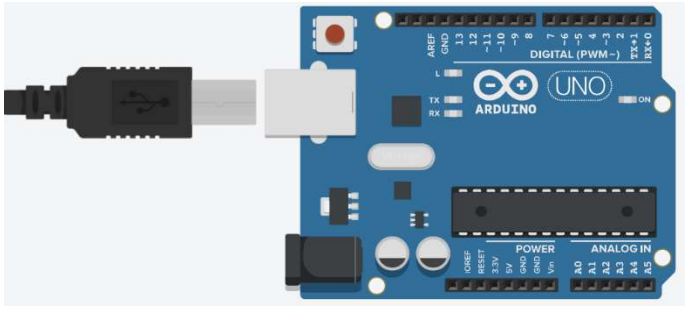

Figure 5 Pin 8 not connected (open)

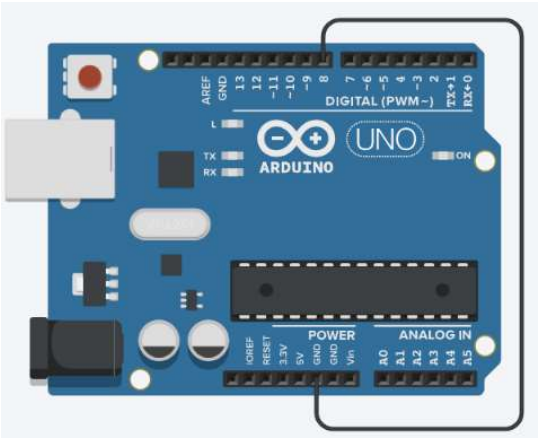

Figure 6 Pin 8 connected to GND

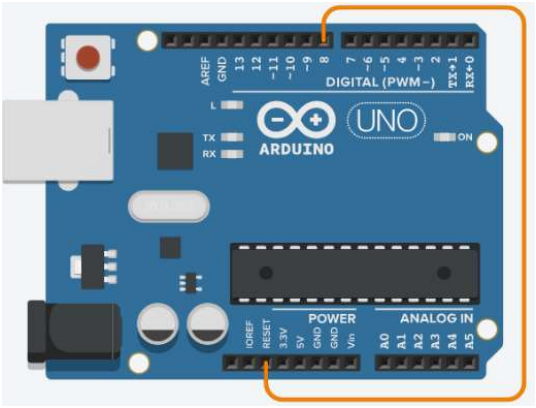

Figure 7 Pin 8 connected to RESET

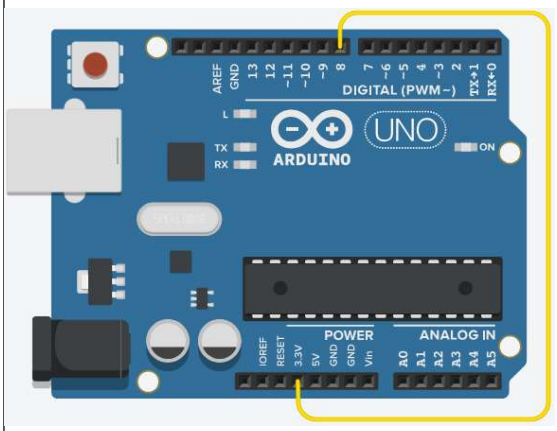

Figure 8 Pin 8 connected to 3.3V

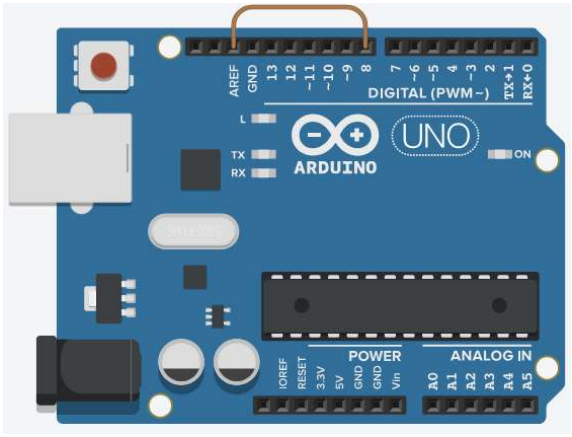

Figure 9 Pin 8 connected to AREF

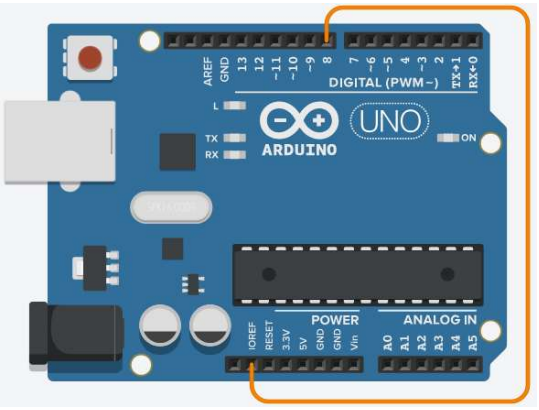

Figure 10 Pin 8 connected to IOREF

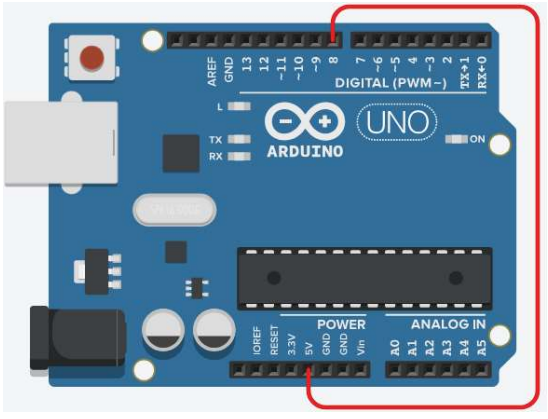

Figure 11Pin 8 connected to 5V

3. **Digital Read**: Upload the corresponding Arduino code for Task 3. Inside the code, the command `pinMode(8, INPUT_PULLUP)` maintains pin 8 to HIGH when it is left unconnected. Observe the LED by connecting and disconnecting pin 8 to GND pin repetitively.

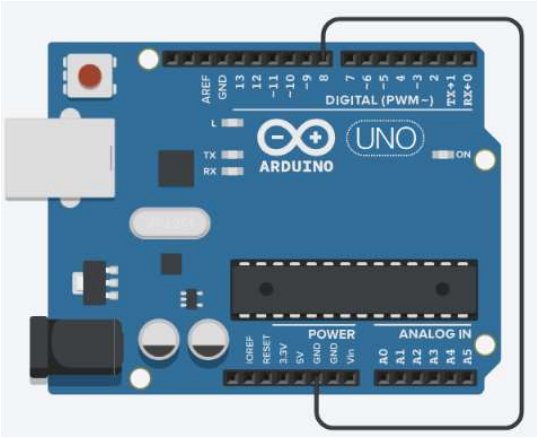

Figure 12 Pin 8 connected to GND

4. **Digital Read**: Upload the corresponding Arduino code for Task 4 and observe the LED by connecting and disconnecting pin 8 to GND pin repetitively with a delay of roughly more than 5 seconds between connection to disconnection.

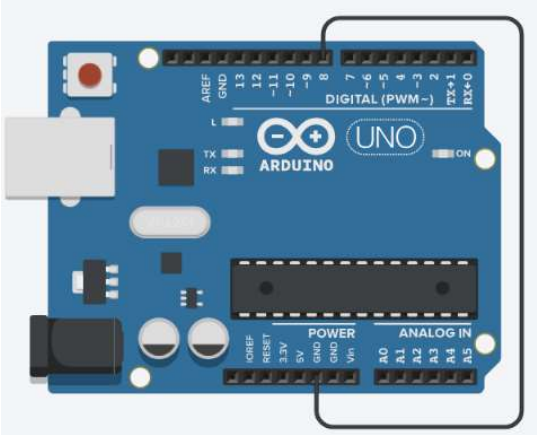

Figure 13 Pin 8 connected to GND

5. **Digital Write:** Upload the corresponding Arduino code and connect the multimeter to pin 8 as shown in the circuit below. The multimeter knob should be set to read voltage (20V range). Read the voltage at pin 8 on the multimeter and report the change in voltage with time.

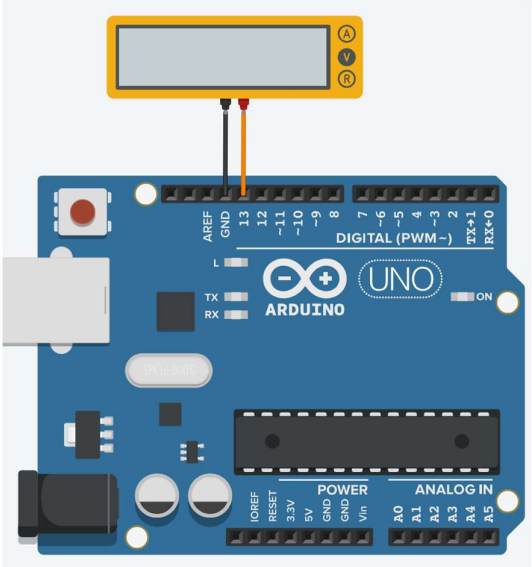

Figure 14 Multimeter connected to pin 13

6. **Analog Write (PWM):** We can use analogWrite() function to write an analog value to a pin in the form of a Pulse Width Modulated (PWM) wave. There are certain pins which have a '~' symbol as their initial. Only these pins can be used for analogWrite(). A call to analogWrite() is on a scale of 0 - 255, such that analogWrite(255) corresponds to 100% duty cycle (always on), and analogWrite(127) is a 50% duty cycle (on half the time) for example.

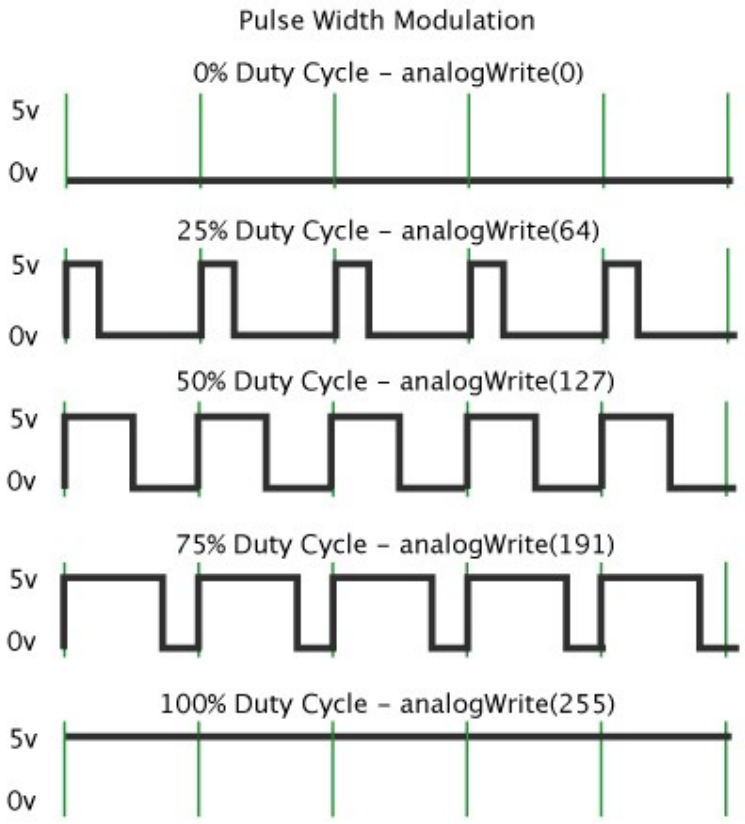

Figure 15 Pulse Width Modulation (PWM) (Image Source: https://docs.arduino.cc/learn/microcontrollers/analog-output/)

Upload the corresponding Arduino code for Task 6. Set the multimeter knob to 20 V voltage range. Connect the multimeter to pin 9 as shown in the circuit below. Change the values inside analogWrite (in 0-255 range) and read the multimeter. Report two such values used by you in analogWrite() along with the voltage on multimeter corresponding to each value.

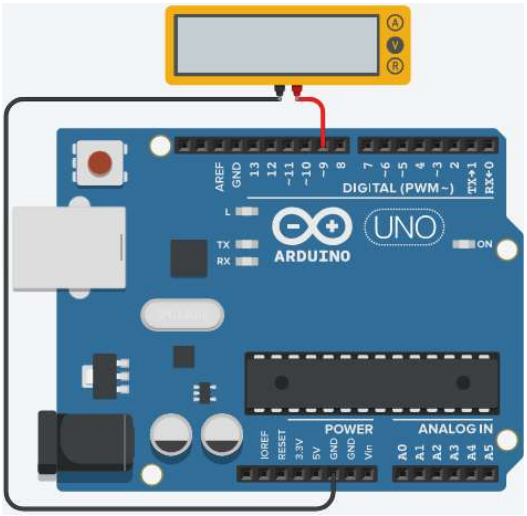

Figure 16 Multimeter connected to pin 9

7. **Analog Read:** Upload the corresponding Arduino code for Task 7. Set the multimeter knob to read voltage (20 V range) and connect the multimeter to 5V pin as shown in the circuit below. Report the reading V1 obtained on multimeter for the 5V pin. Make changes in the Arduino code based on the voltage obtained on the multimeter for 5V pin such that the ADC range of 0-1023 is mapped to the range 0 of V1. Connect A1 pin to different pins (nothing, GND, 3.3V, 5V, Vin, AREF, and IOREF) and report the readings in volts as obtained from serial monitor for each of these connections. Also report the reading (in volts) when pin A1 is left unconnected.

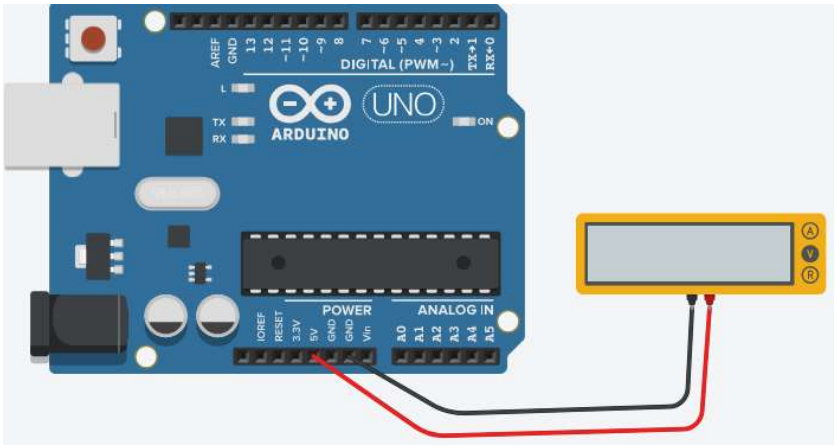

Figure 17 Multimeter connected to 5V pin

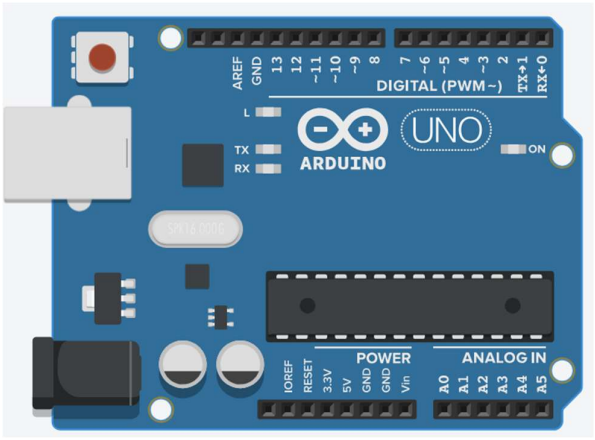

Figure 18 Pin A1 not connected (open)

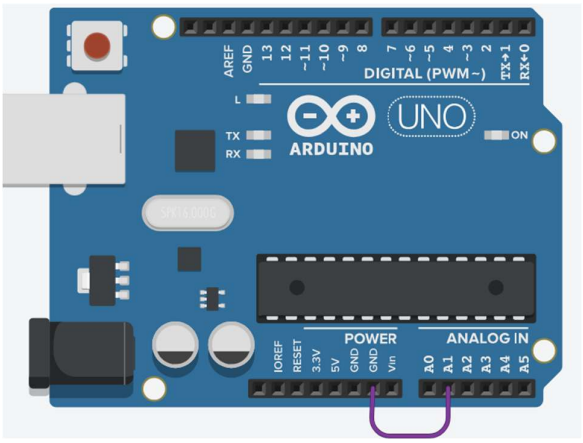

Figure 19 Pin A1 connected to GND

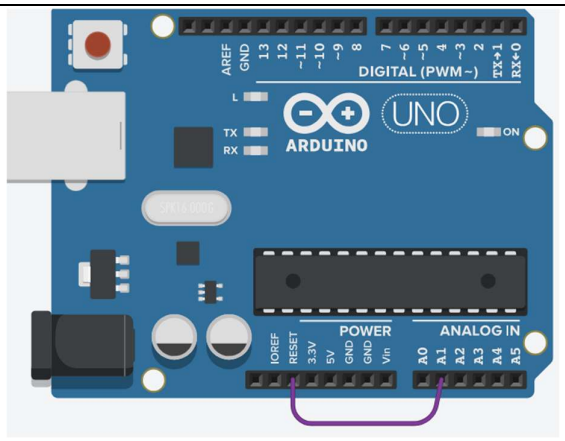

Figure 20 Pin A1 connected to RESET

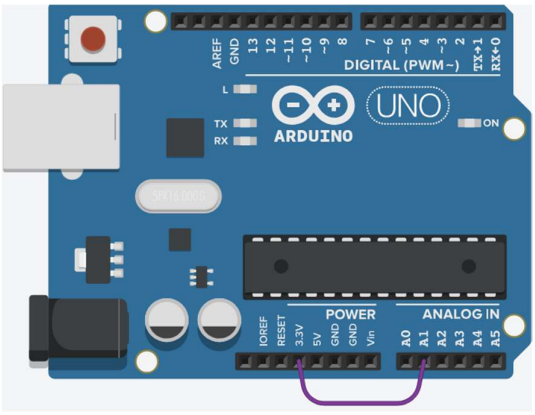

Figure 21 Pin A1 connected to 3.3V

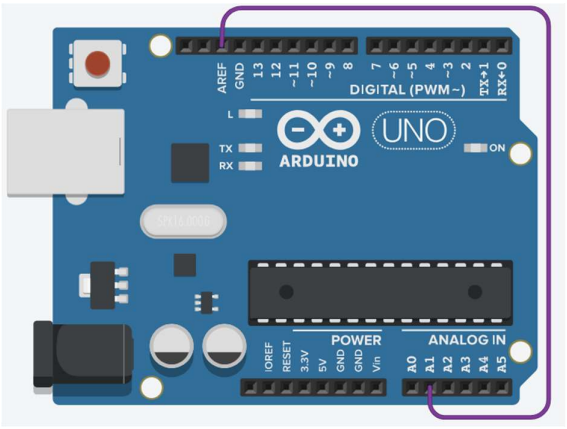

Figure 22 Pin A1 connected to AREF

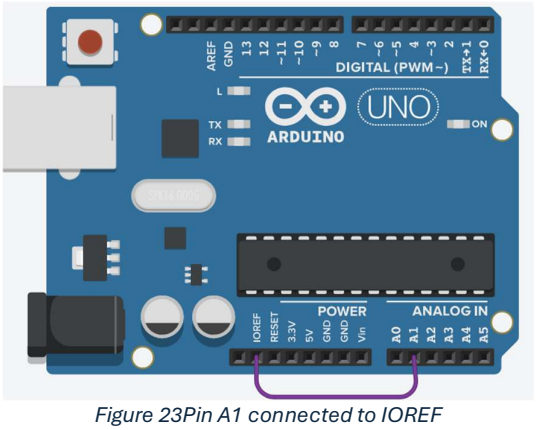

Figure 23Pin A1 connected to IOREF

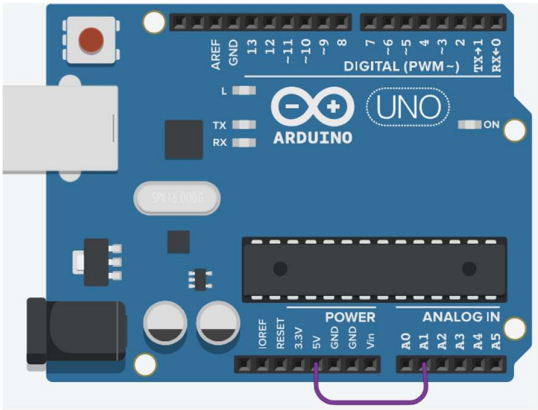

Figure 24 Pin A1 connected to 5V

8. **Analog Read:** Upload the corresponding Arduino code for Task 8. Connect pin A1 to PWM output pin 9. Make changes in the code to set a different duty cycle (60%, 85%) for the PWM. Observe the pattern on the Serial Plotter (serial plotter can be opened by clicking on Tools→Serial Plotter) and report the same.

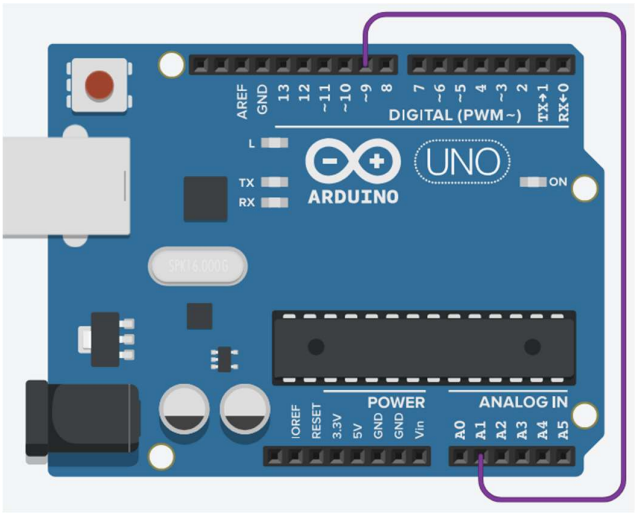

Figure 25 PWM output at pin 9 connected to analog input pin A1

Below is an example of Serial Plotter observed for 30% duty cycle.

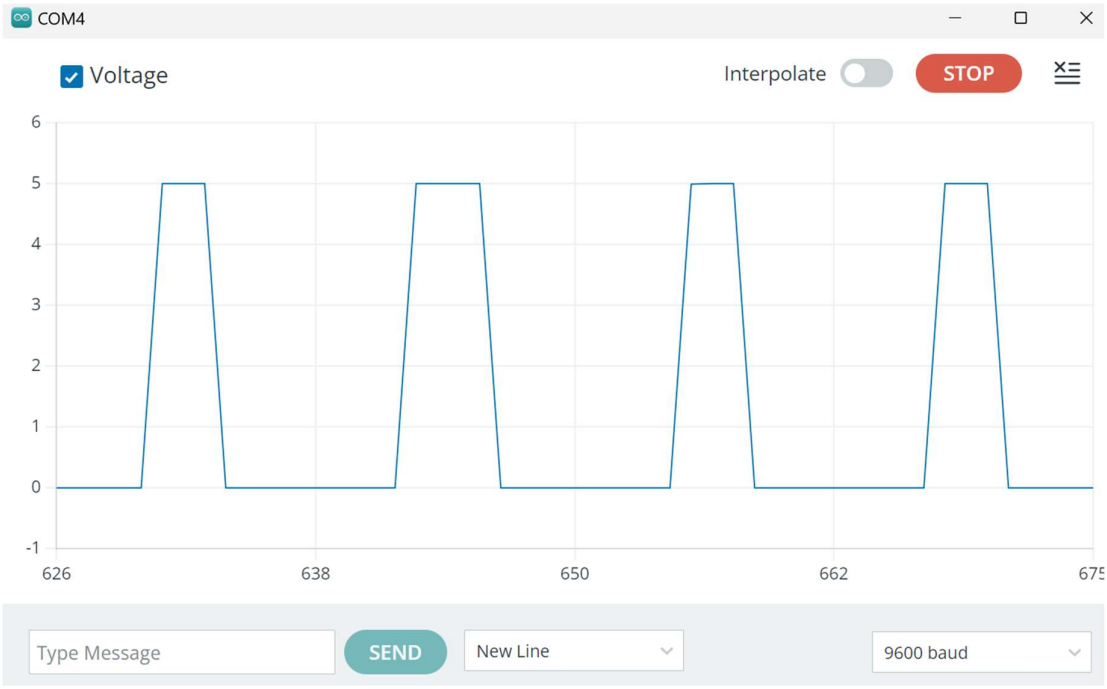

Figure 26 Serial Plotter observed for 30% duty cycle of PWM

REPORT FORMAT

Experiment No.

Experiment Title:

Group No.:

Name:

Roll no.

Results for the group task:

Task no.	Topic	Task	Report
1			
2			
3			
4			
5			
6			
7			
8			

Results for individually assigned task:

- Task description:
- Pictures of the circuit along with multimeter reading (if any):
- Pictures of the serial monitor (if any):