Practical Robotics Projects with Arduino

(CSE 4571)

Lab Assignment No – 06

Servo Motor Control

Submission Date: _____

Branch: CSE		**Section: 2241026**
Name	**Registration No.**	**Signature**
Dinanath Dash	2241004161	

Department of Computer Science and Engineering
Institute of Technical Education and Research (Faculty of Engineering)
Siksha 'O' Anusandhan (Deemed to be University)
Bhubaneswar, Odisha-751030.

Aim:

Exploration and Implementation of Servo Motor Control Systems with Arduino Platform.

Objectives:

1) **Interface and control servomotors using Arduino, including using a potentiometer to adjust the position of the servo.**

   1.1) **Demonstrate how to interface and control a servo motor with Arduino and configure the servo library.**

   - ✓ Introduce the concept of servo motors and their applications in embedded systems.
   - ✓ Explain how to connect a servo motor to an Arduino board using jumper wires.
   - ✓ Demonstrate how to install and configure the servo library in the Arduino IDE.
   - ✓ Write an Arduino sketch to Control a Servo Motor to Move Sequentially to Different Positions with Arduino and Serial Monitor.

   1.2) **Control a servo motor to move back and forth between 0 and 180 degrees using the Arduino Servo library and for loop.**

   - ✓ Understanding the use of for loop to control the servo motor
   - ✓ Configuring the servo motor to move from 0 to 180 degrees using for loop
   - ✓ Configuring the servo motor to move from 180 to 0 degrees using for loop
   - ✓ Testing the servo motor and verifying its motion using serial monitor

   1.3) **To control the position of a servo motor using values received from the serial monitor and verify its movement by observing the motor's motion through the use of serial communication.**

   - ✓ Understanding how to use the serial monitor to read input from the user.
   - ✓ Understanding how to parse the user input to obtain the desired servo position.
   - ✓ Understanding how to use the parsed input to control the servo motor.
   - ✓ Testing the program and verifying that the servo motor moves to the desired position in response to user input.

   1.4) **Write an Arduino sketch to control the position of a servo motor using a potentiometer.**

   - ✓ Demonstrate how to connect a potentiometer to an Arduino board and read its analog input using the "analogRead" function.
   - ✓ Explain how to map the analog input value to the corresponding servo position value.
   - ✓ Write an Arduino sketch that reads the analog input from the potentiometer and generates PWM signals to control the position of the servo motor accordingly.

2) **Interface and control analog input devices such as an LDR and joystick with Arduino to collect data for use in servo motor control.**

   2.1) **Interface an LDR (Light Dependent Resistor) with Arduino and use the collected data on environmental light conditions to control the position of a servo motor.**

   - ✓ Configure the LDR with Arduino.
   - ✓ Read the analog data from the LDR using the "analogRead" command.
   - ✓ Map the analog LDR data to the corresponding servo position using the "map" command.
   - ✓ Control the position of a servo motor based on the mapped LDR data.
   - ✓ Display the LDR value and corresponding servo position using the Serial monitor.

2.2) Interface a Joystick with Arduino and Collect Data on its Position and Switch State

- ✓ Configure the pins for the joystick and switch as inputs.
- ✓ Read the values of X and Y axis of the joystick and the switch.
- ✓ Write an Arduino sketch to print the values of X and Y axis of the joystick and the switch state to the serial monitor.

2.3) Interface a joystick with Arduino to control the position of a servo motor:

- ✓ Connect the joystick to Arduino and calibrate the joystick inputs.
- ✓ Map the joystick inputs to control the position of the servo motor.
- ✓ Write an Arduino sketch to control the position of the servo motor using the joystick inputs.

2.4) Implement advanced control techniques, such as using two servos to direct a laser beam, to demonstrate the versatility of servo motors in embedded systems.

- ✓ **Explain the principles of laser beam control using two servo motors and how it can be used in practical applications.** In this section, you would explain how two servo motors can be used to control the position of a laser beam. One servo motor would control the horizontal movement of the laser, while the other would control the vertical movement. This allows for precise targeting of the laser beam, which can be useful in a variety of practical applications such as laser cutting, engraving, and alignment.
- ✓ **Demonstrate how to connect a laser diode to a servo motor and control its position using Arduino.** In this section, you would explain how to connect a laser diode to a servo motor and control its position using Arduino. This would likely involve wiring the servo motors to the appropriate pins on the Arduino, connecting the laser diode to one of the servo motors, and configuring the servo motors to move the laser beam in the desired manner.
- ✓ Write an Arduino sketch to control the position of a laser beam using two servo motors.

Pre-Lab Questionnaire:

1) What are motors and how do they work?
2) What are the components and mechanisms involved in the operation of a servomotor?
3) How do torque, speed, and precision relate to the unique features of servomotors?
4) What are some common applications for servomotors in embedded systems?
5) What is the role of feedback in motor control, and how can it be implemented with Arduino?
6) What is the difference between open-loop and closed-loop motor control, and how can Arduino be used to implement closed-loop control?
7) What is the purpose of the servo library in Arduino?
8) What are the applications of servo motors in embedded systems?
9) What is the role of pulse width modulation (PWM) in servo control?
10) How is the position of a servo motor related to the duty cycle of the PWM signal?
11) What is the purpose of mapping the analog input value to the corresponding servo position value?
12) What is the difference between digital and analog signals, and how are they used in servo control?

Answers to Pre-Lab Questions

Components/Equipment Required:

Sl. No.	Name of the Component / Equipment	Specification	Quantity
1)	Arduino UNO R3	16MHz	1
2)	Arduino UNO cable	USB Type A to B	1
3)	Trimmer Potentiometer	10k, Preset	1
4)	LDR/Photoresistor	5mm	1
5)	Dual axis Joystick module		1
6)	Servo Motor	SG90	2
7)	Dual H-Bridge Motor Driver IC	L293D, DIP-16 Package	1
8)	Toy fan blade	----------	1
9)	Resistors (carbon type)	¼ watt (330Ω)	1
		¼ watt (4.7kΩ)	1
10)	LED	Any two different colour of your choice	2
11)	Buzzer	5V, small	1
12)	Breadboard	840 Tie points	1
13)	Digital Multimeter	----------	1
14)	Jumper Wire	----------	As per requirement

Objective 1

Interface and control servomotors using Arduino, including using a potentiometer to adjust the position of the servo.

Circuit / Schematic Diagram

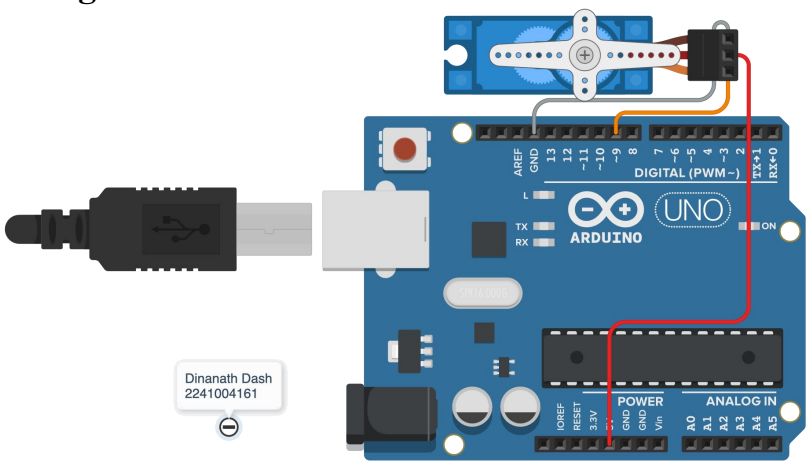


Figure 1.1: Schematic of interface and control a servo motor with Arduino and configure the servo library

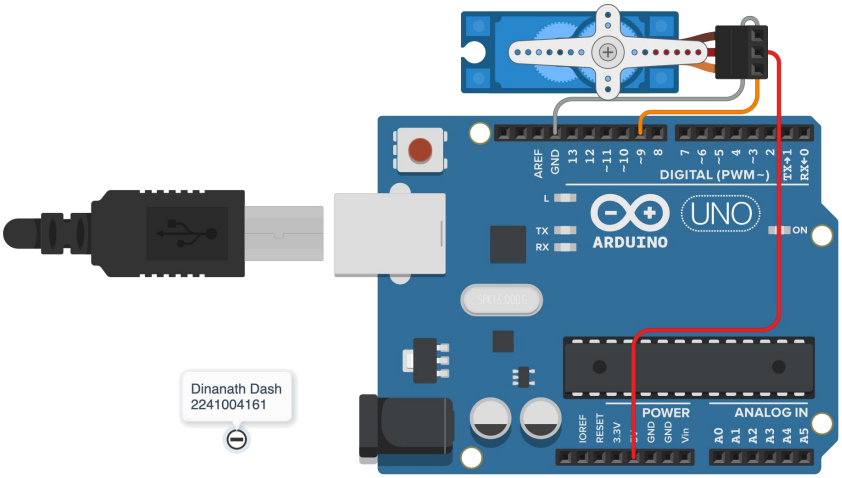


Figure 1.2: Schematic of control a servo motor to move back and forth between 0 and 180 degrees using the Arduino Servo library and for loop.

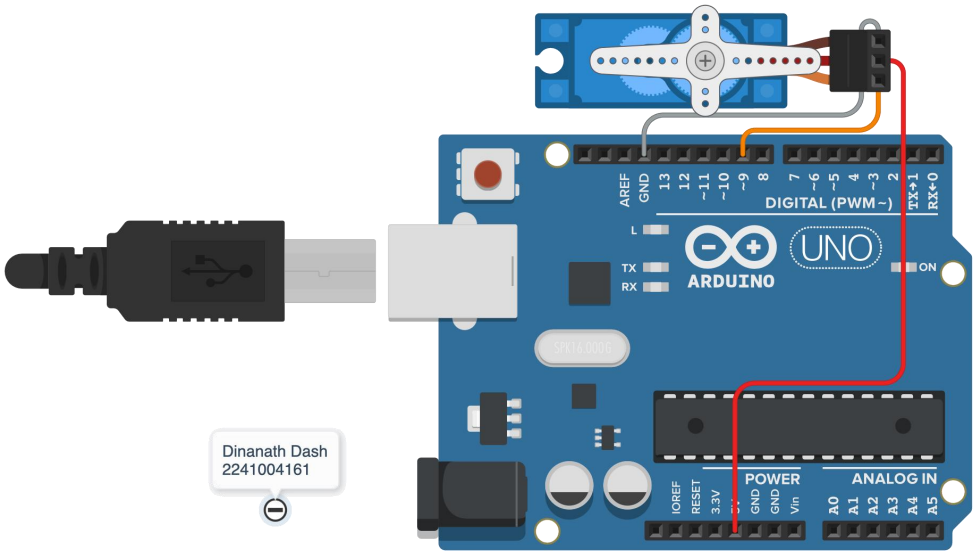


Figure 1.3: Schematic of controlling the position of a servo motor using values received from the serial monitor

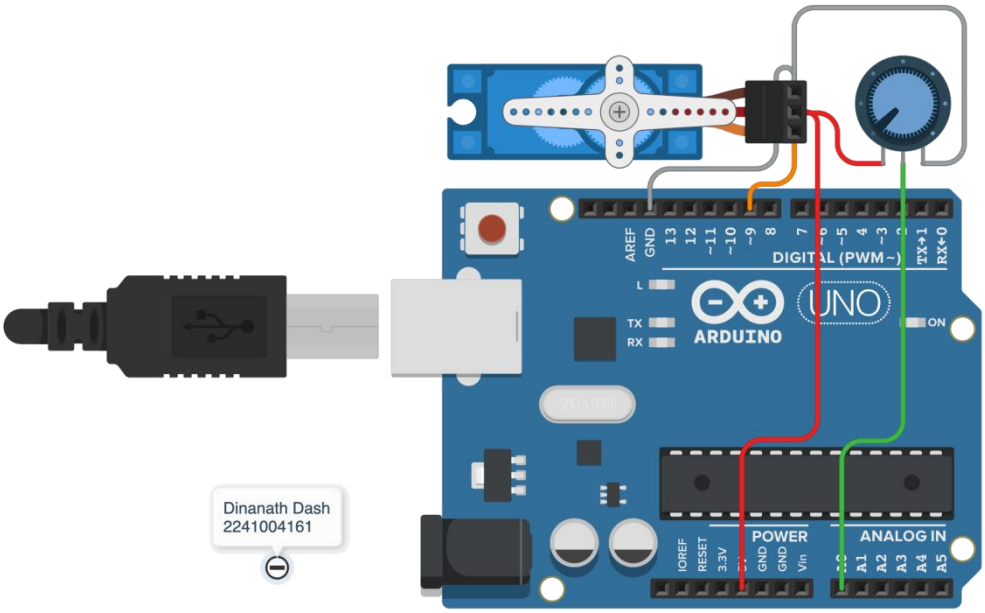


Figure 1.4: Schematic of controlling the position of a servo motor using a potentiometer.

Code

1.1) Interface and control a servo motor with Arduino and configure the servo library.

```
#include <Servo.h>
Servo myservo;
void setup() {
  myservo.attach(9);
  Serial.begin(9600);
}
void loop() {
  for (int pos = 0; pos <= 180; pos += 45) {
    myservo.write(pos);
    Serial.print("Position: ");
    Serial.println(pos);
    delay(1000);
  }
}
```

1.2) Control a servo motor to move back and forth between 0 and 180 degrees using the Arduino Servo library and for loop.

```
#include <Servo.h>
Servo myservo;
void setup() {
  myservo.attach(9);
  Serial.begin(9600);
}
void loop() {
  for (int pos = 0; pos <= 180; pos++) {
    myservo.write(pos);
    delay(15);
  }
  for (int pos = 180; pos >= 0; pos--) {
```

```
  myservo.write(pos);
  delay(15);
 }
}
```

1.3) Control the position of a servo motor using values received from the serial monitor and verify its movement by observing the motor's motion through the use of serial communication.

```
#include <Servo.h>
Servo myservo;
void setup() {
 Serial.begin(9600);
 myservo.attach(9);
 Serial.println("Enter angle (0-180):");
}
void loop() {
 if (Serial.available()) {
  int pos = Serial.parseInt();
  if (pos >= 0 && pos <= 180) {
   myservo.write(pos);
   Serial.print("Moved to: ");
   Serial.println(pos);
  } else {
   Serial.println("Invalid input");
  }
 }
}
```

1.4) Control the position of a servo motor using a potentiometer.

```
#include <Servo.h>
Servo myservo;
int potPin = A0;
void setup() {
 myservo.attach(9);
}
void loop() {
 int val = analogRead(potPin);
 int angle = map(val, 0, 1023, 0, 180);
 myservo.write(angle);
 delay(15);
}
```

Observation

Figure 1.1.1: Simulation based interface and control a servo motor with Arduino and configure the servo library.

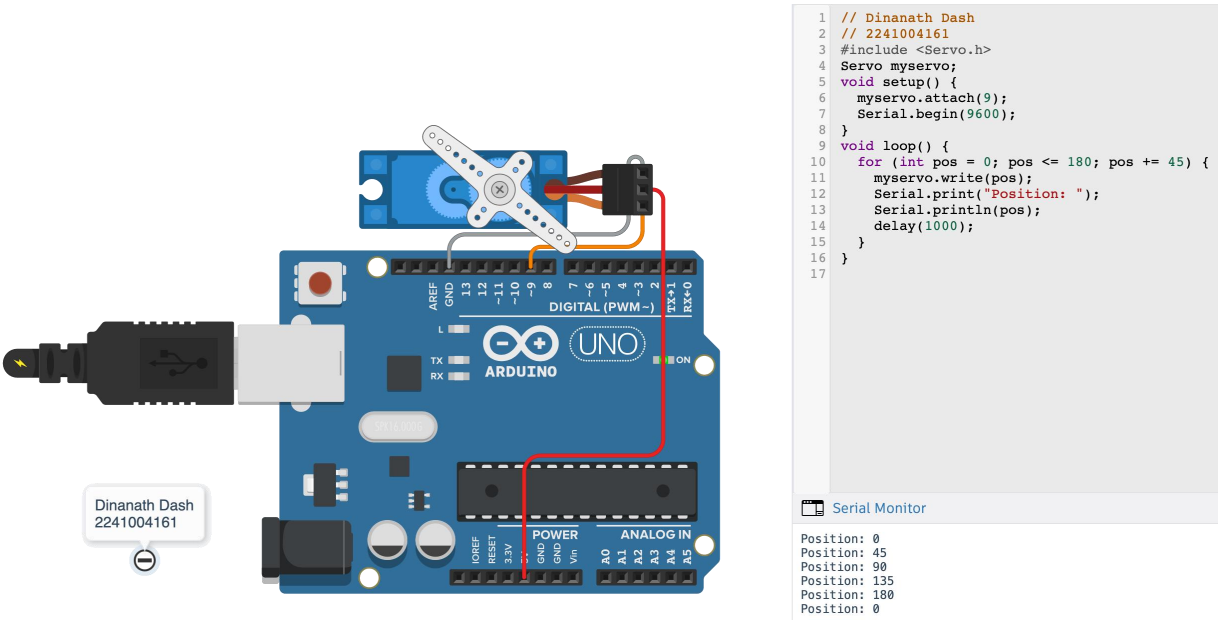
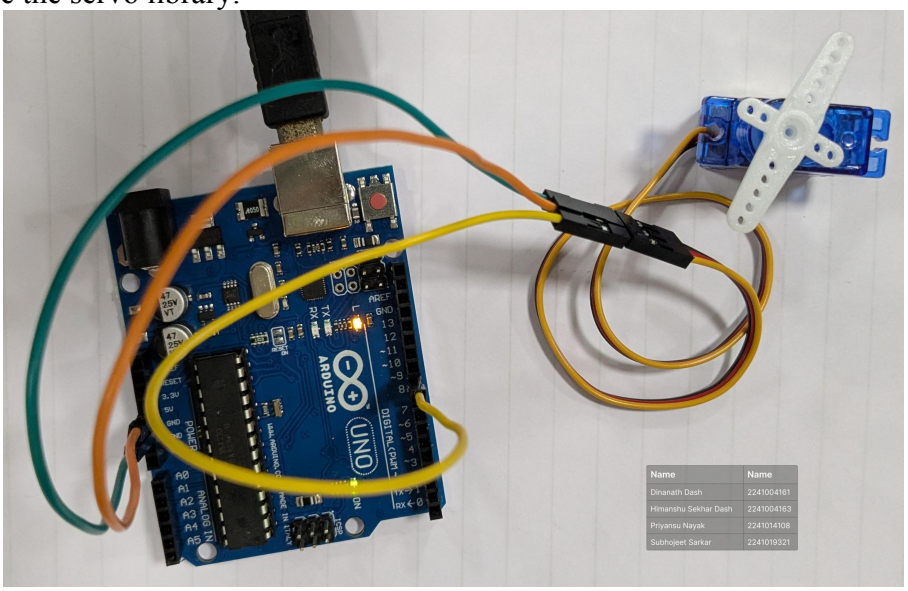


Figure 1.1.2: Hardware Implementation based interface and controlling a servo motor with Arduino and configure the servo library.




```
Position: 0
Position: 45
Position: 90
Position: 135
Position: 180
Position: 0
Position: 45
```

Figure 1.2.1: Simulation based controlling a servo motor to move back and forth between 0 and 180 degrees using the Arduino Servo library and for loop.

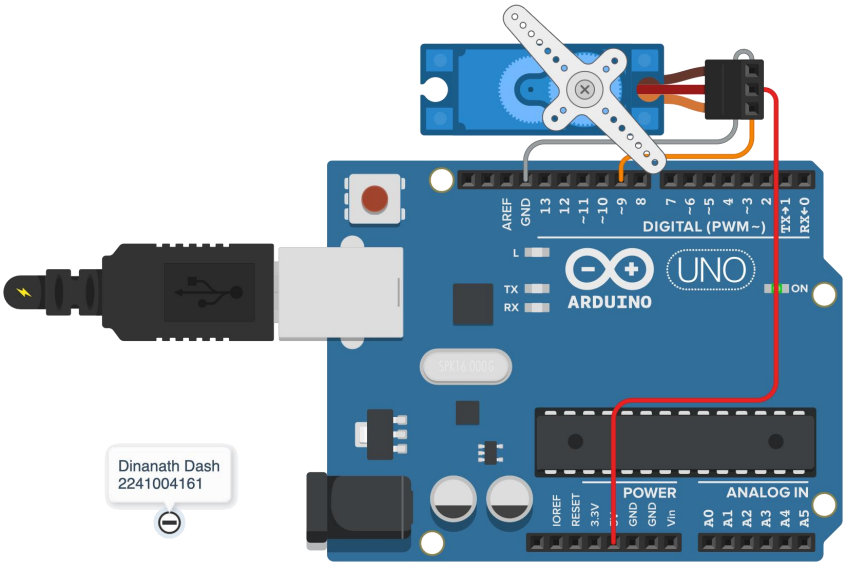


Figure 1.2.2: Hardware Implementation based controlling a servo motor to move back and forth between 0 and 180 degrees using the Arduino Servo library and for loop.

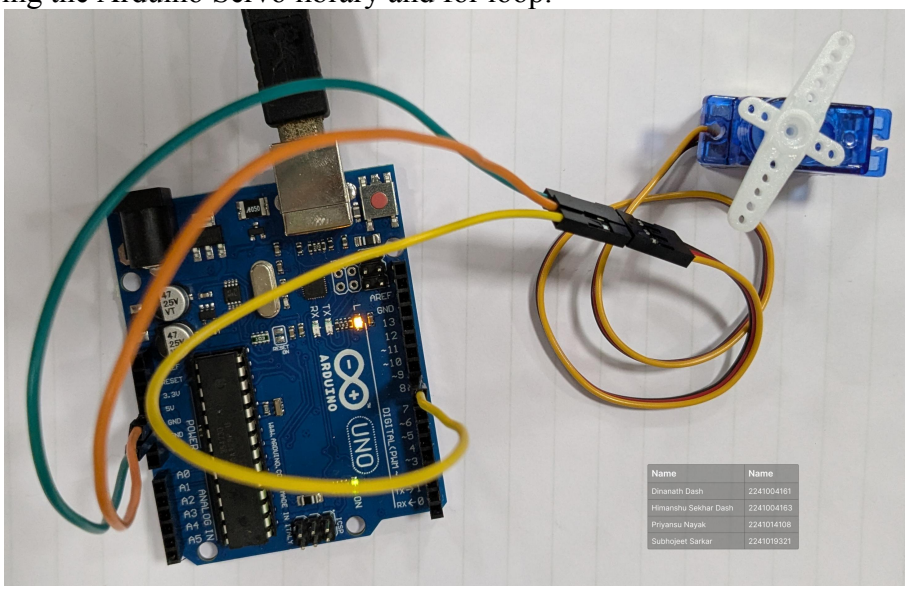


Figure 1.3.1: Simulation based controlling the position of a servo motor using values received from the serial monitor and verify its movement by observing the motor's motion through the use of serial communication.

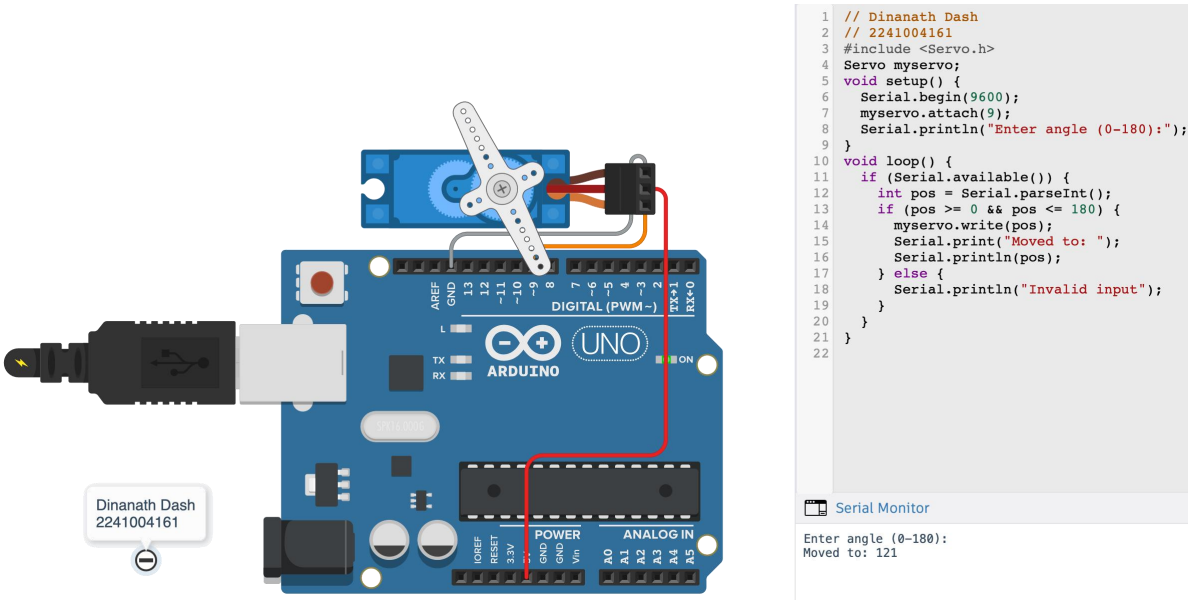


Figure 1.3.2: Hardware Implementation based controlling the position of a servo motor using values received from the serial monitor and verify its movement by observing the motor's motion through the use of serial communication.

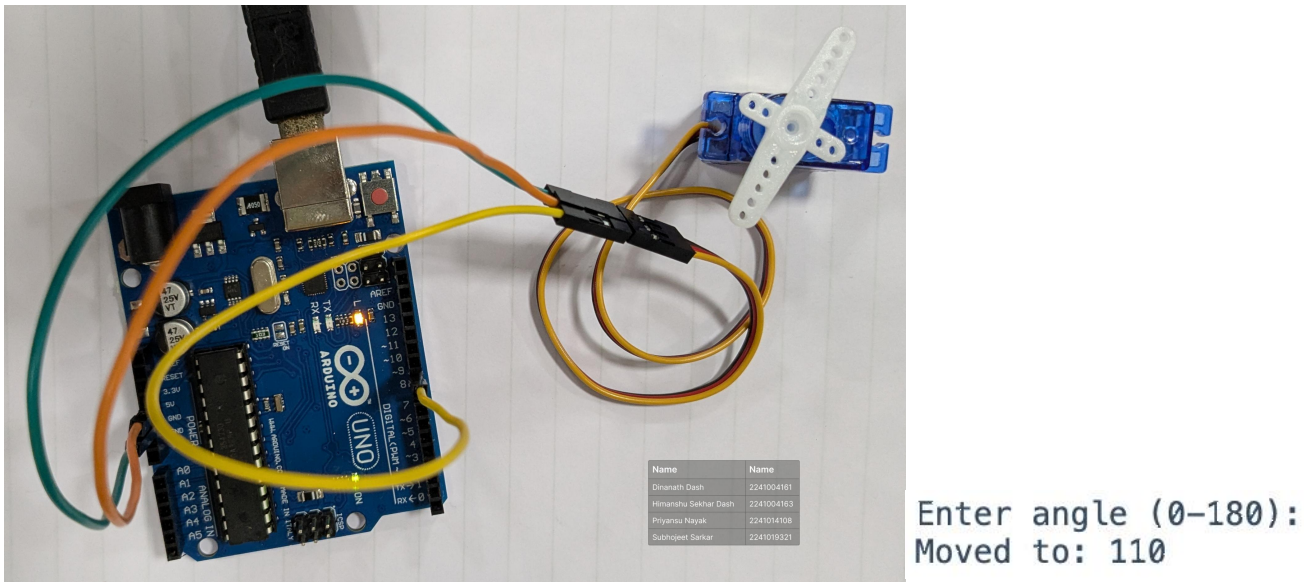


Figure 1.4.1: Simulation based controlling the position of a servo motor using a potentiometer.

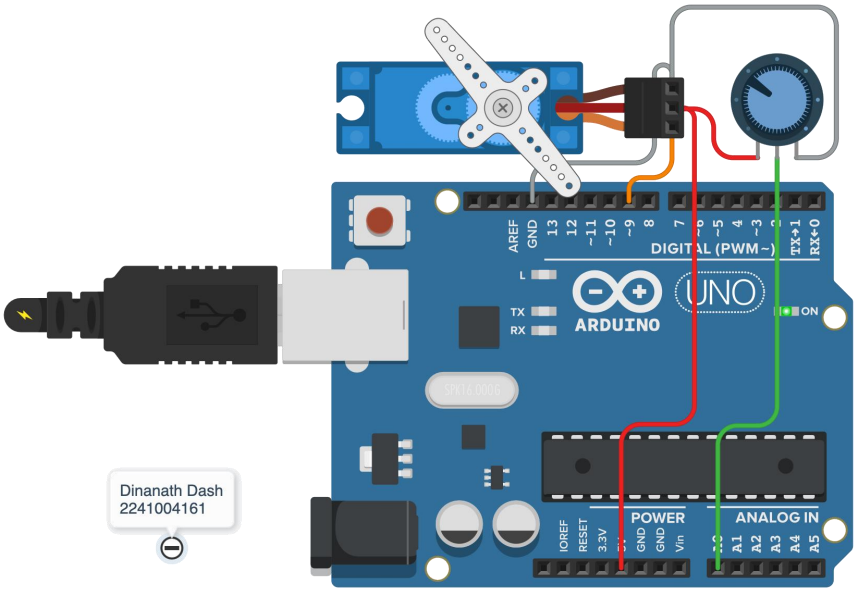
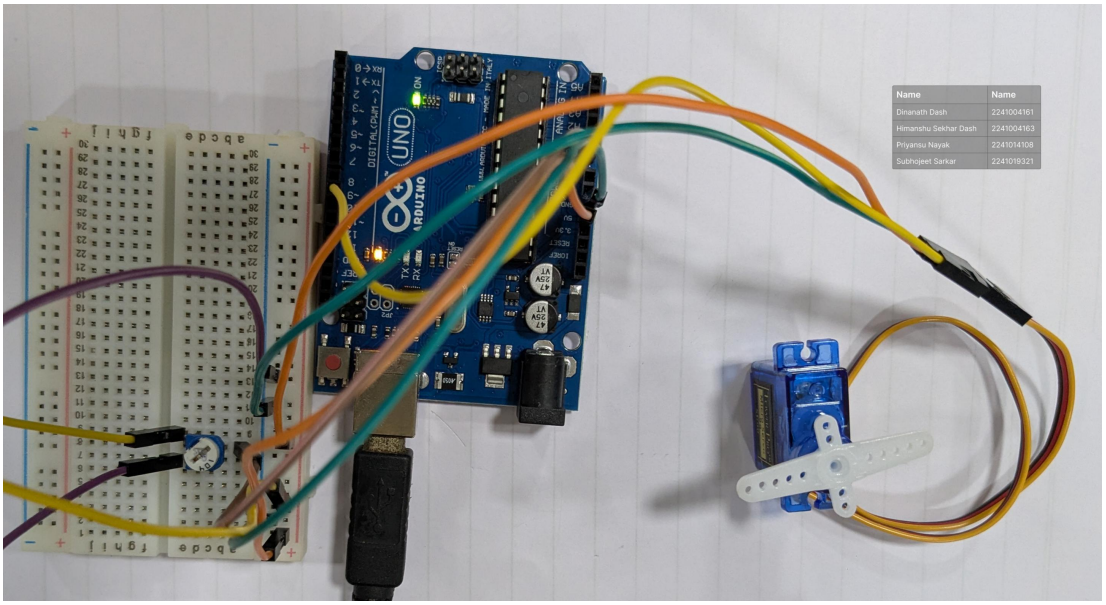


Figure 1.4.2: Hardware Implementation based controlling the position of a servo motor using a potentiometer.



Objective 2

Interface and control analog input devices such as an LDR and joystick with Arduino to collect data for use in servo motor control.

Circuit / Schematic Diagram

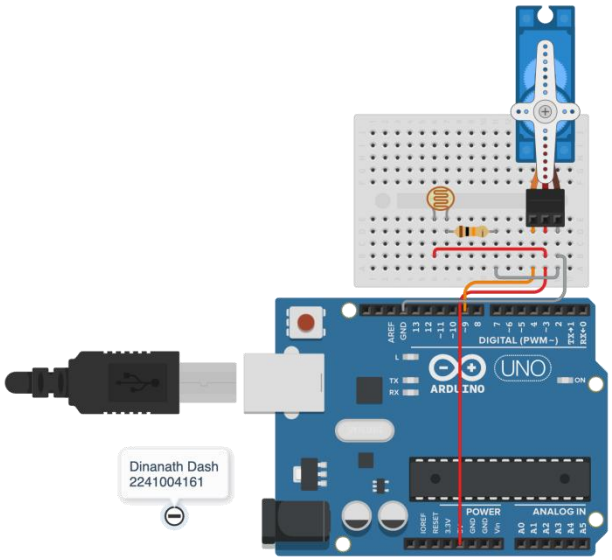


Figure 2.1: Schematic of interfacing an LDR (Light Dependent Resistor) with Arduino and use the collected data on environmental light conditions to control the position of a servo motor.

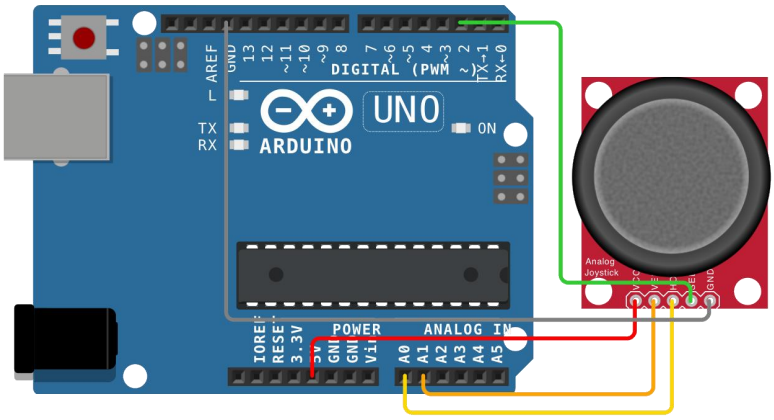


Figure 2.2: Schematic of interfacing a Joystick with Arduino and Collect Data on its Position and Switch State.

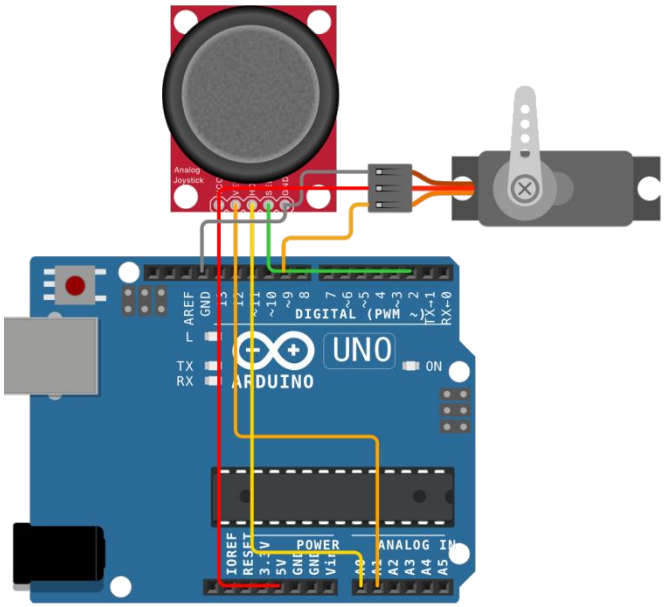


Figure 2.3: Schematic of interfacing a joystick with Arduino to control the position of a servo motor.

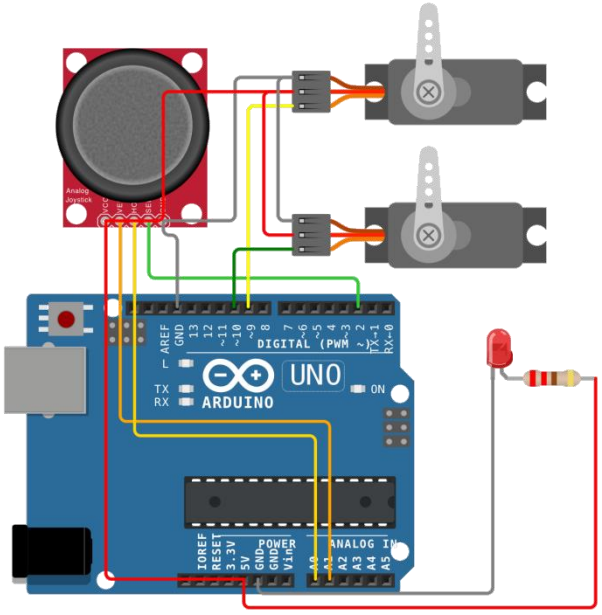


Figure 2.4: Schematic of implementing advanced control techniques, such as using two servos to direct a laser beam, to demonstrating the versatility of servo motors in embedded systems.

Code

2.1) Interfacing an LDR (Light Dependent Resistor) with Arduino and use the collected data on environmental light conditions to control the position of a servo motor

```
#include <Servo.h>
Servo myservo;
int ldrPin = A0;
void setup() {
  myservo.attach(9);
  Serial.begin(9600);
}
void loop() {
  int ldrValue = analogRead(ldrPin);
  int angle = map(ldrValue, 0, 1023, 0, 180);
  myservo.write(angle);
  Serial.print("LDR: ");
  Serial.print(ldrValue);
  Serial.print(" | Servo: ");
  Serial.println(angle);
  delay(200);
}
```

2.2) Interfacing a Joystick with Arduino and Collect Data on its Position and Switch State.

```
int xPin = A0, yPin = A1, swPin = 2;
void setup() {
  Serial.begin(9600);
  pinMode(swPin, INPUT_PULLUP);
}
void loop() {
```

```
  int xVal = analogRead(xPin);
  int yVal = analogRead(yPin);
  int swVal = digitalRead(swPin);
  Serial.print("X: "); Serial.print(xVal);
  Serial.print(" | Y: "); Serial.print(yVal);
  Serial.print(" | Switch: "); Serial.println(swVal == 0 ? "Pressed" : "Released");
  delay(200);
 }
```

2.3) Interfacing a joystick with Arduino to control the position of a servo motor.

```
#include <Servo.h>
Servo myservo;
int xPin = A0;
void setup() {
  myservo.attach(9);
  Serial.begin(9600);
}
void loop() {
  int xVal = analogRead(xPin);
  int angle = map(xVal, 0, 1023, 0, 180);
  myservo.write(angle);
  Serial.print("Joystick X: ");
  Serial.print(xVal);
  Serial.print(" | Servo: ");
  Serial.println(angle);
  delay(100);
 }
```

2.4) Implementing advanced control techniques, such as using two servos to direct a laser beam, to demonstrating the versatility of servo motors in embedded systems.

```
#include <Servo.h>
Servo servoX, servoY;
void setup() {
  servoX.attach(9);
  servoY.attach(10);
}
void loop() {
  for (int x = 0; x <= 180; x += 10) {
   for (int y = 0; y <= 180; y += 10) {
    servoX.write(x);
    servoY.write(y);
    delay(100);
   }
  }
 }
```

Observation

Figure 2.1.1: Simulation based interfacing an LDR (Light Dependent Resistor) with Arduino and use the collected data on environmental light conditions to control the position of a servo motor

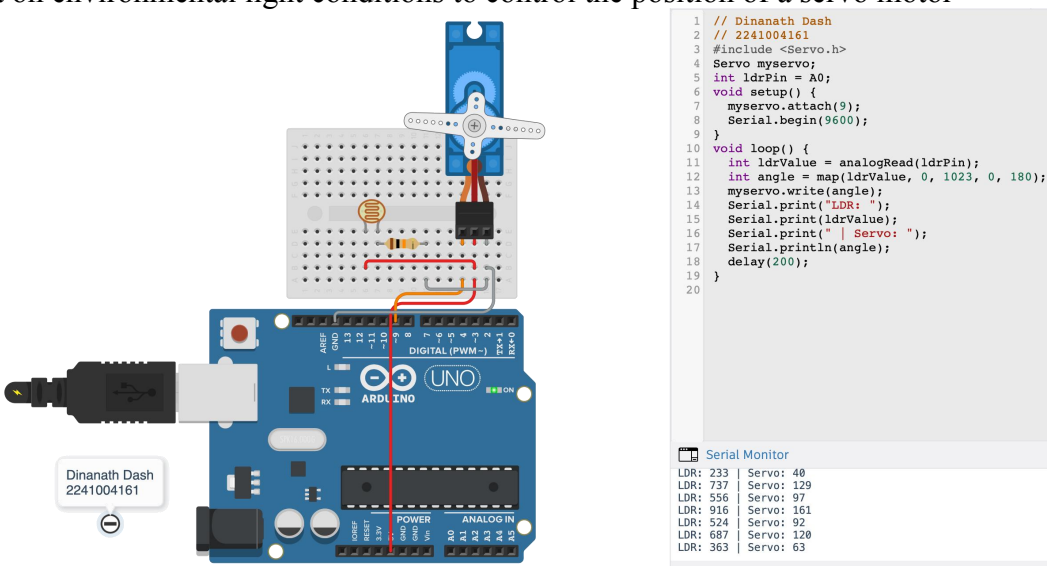
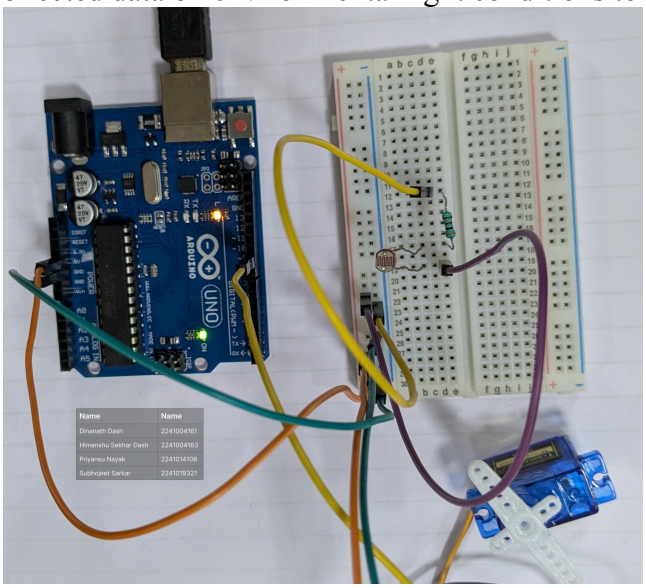
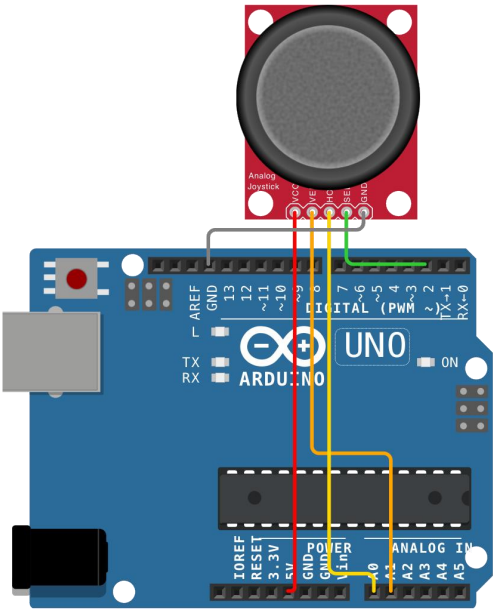


Figure 2.1.2: Hardware Implementation based interfacing an LDR (Light Dependent Resistor) with Arduino and use the collected data on environmental light conditions to control the position of a servo motor.



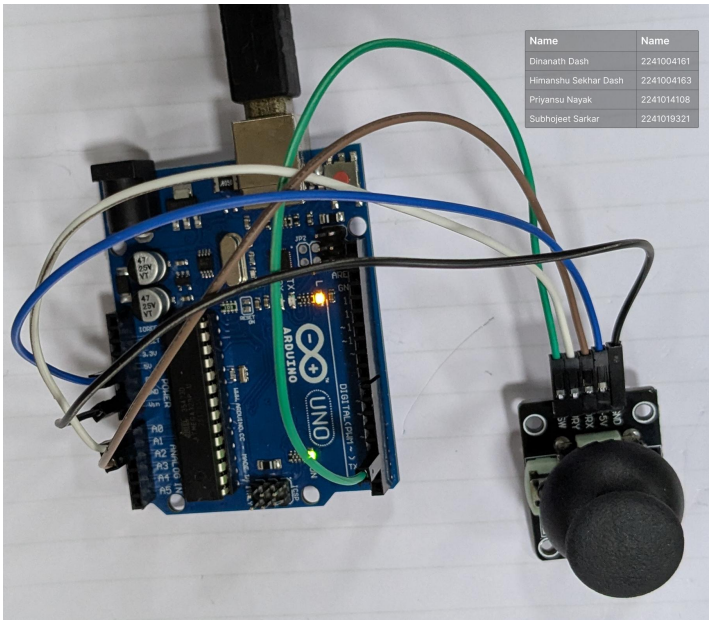
```
LDR: 940 | Servo: 165
LDR: 791 | Servo: 139
LDR: 345 | Servo: 60
LDR: 908 | Servo: 159
LDR: 460 | Servo: 80
LDR: 897 | Servo: 157
LDR: 358 | Servo: 62
```

Figure 2.2.1: Simulation based interfacing a Joystick with Arduino and Collect Data on its Position and Switch State.



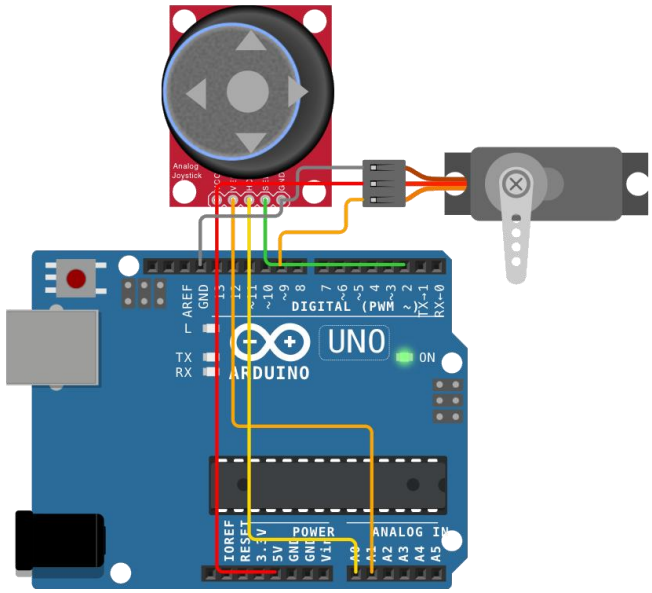
```
X: 512 | Y: 512 | Switch: Released
X: 512 | Y: 512 | Switch: Pressed
X: 1023 | Y: 512 | Switch: Released
X: 512 | Y: 512 | Switch: Released
X: 512 | Y: 1023 | Switch: Released
X: 512 | Y: 512 | Switch: Released
X: 0 | Y: 512 | Switch: Released
```

Figure 3.2.2: Hardware Implementation based interfacing a Joystick with Arduino and Collect Data on its Position and Switch State.



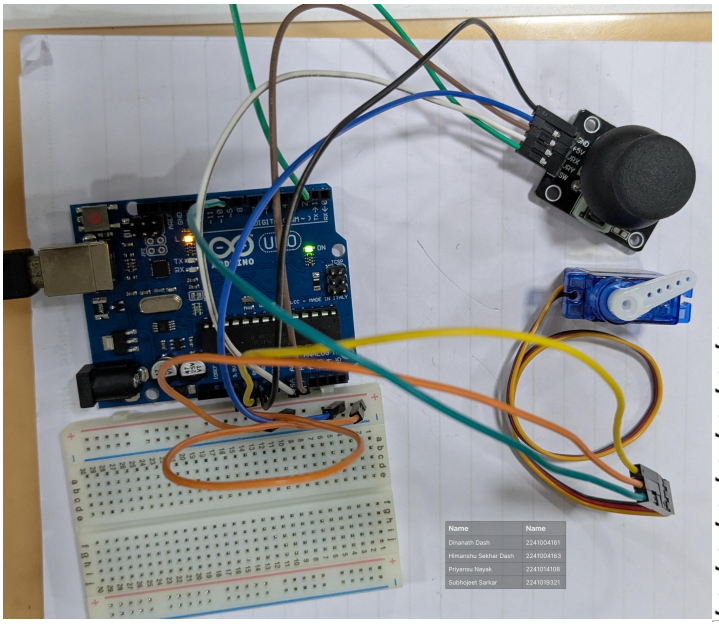
```
X: 512 | Y: 512 | Switch: Released
X: 512 | Y: 512 | Switch: Pressed
X: 1023 | Y: 512 | Switch: Released
X: 512 | Y: 512 | Switch: Released
X: 512 | Y: 1023 | Switch: Released
X: 512 | Y: 512 | Switch: Released
X: 0 | Y: 512 | Switch: Released
```

Figure 2.3.1: Simulation based interfacing a joystick with Arduino to control the position of a servo motor.



```
Joystick X: 512 | Servo: 90
Joystick X: 1023 | Servo: 180
Joystick X: 512 | Servo: 90
Joystick X: 512 | Servo: 90
Joystick X: 0 | Servo: 0
Joystick X: 512 | Servo: 90
Joystick X: 512 | Servo: 90
```

Figure 2.3.2: Hardware Implementation based interfacing a joystick with Arduino to control the position of a servo motor.



```
Joystick X: 512 | Servo: 90
Joystick X: 1023 | Servo: 180
Joystick X: 512 | Servo: 90
Joystick X: 512 | Servo: 90
Joystick X: 0 | Servo: 0
Joystick X: 512 | Servo: 90
Joystick X: 512 | Servo: 90
```

Figure 2.4.1: Simulation based implementing advanced control techniques, such as using two servos to direct a laser beam, to demonstrating the versatility of servo motors in embedded systems.

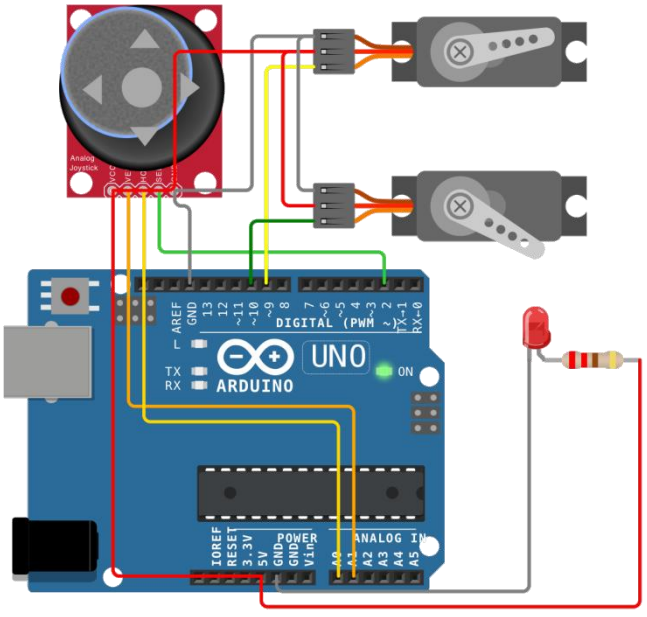
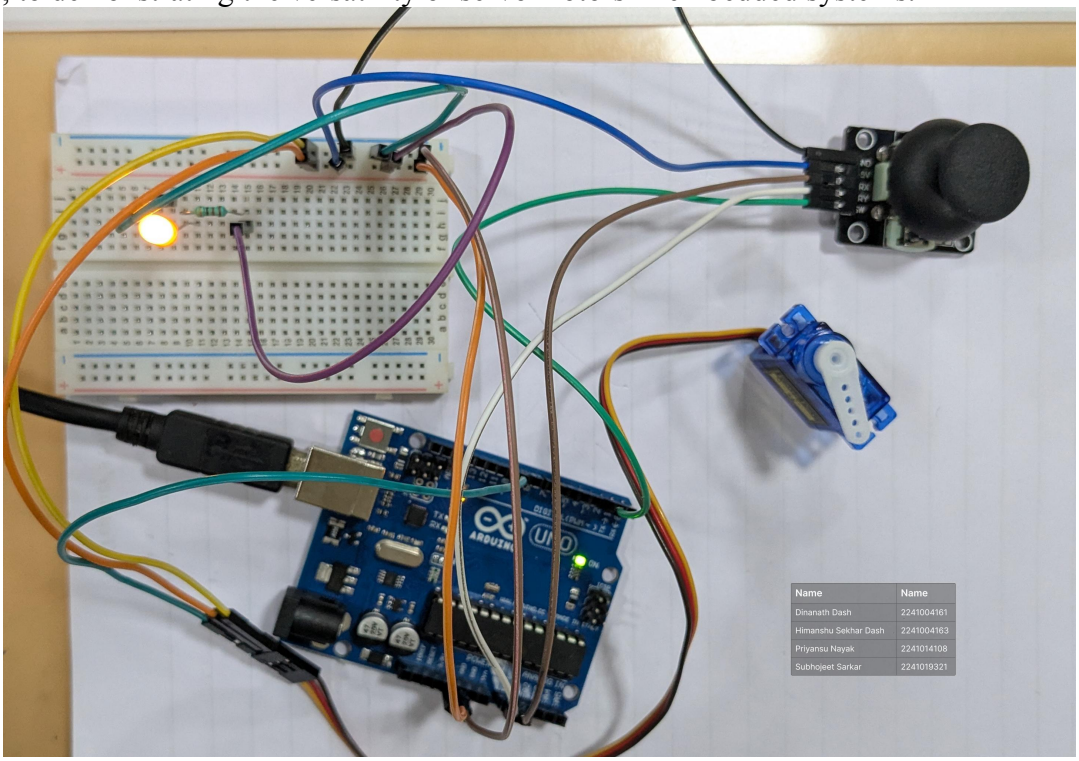


Figure 2.4.2: Hardware based implementing advanced control techniques, such as using two servos to direct a laser beam, to demonstrating the versatility of servo motors in embedded systems.



Conclusion

Precautions

Post Experiment Questionnaire:

1) What is laser beam control using two servo motors, and what are its practical applications?
2) How can advanced control techniques be used to demonstrate the versatility of servo motors in embedded systems?
3) What are the principles of using two servo motors to direct a laser beam, and how is this useful in practical applications?

Answers to Post-Lab Questions

(Signature of the Faculty)

Date: _____

(Signature of the Student)

Name: _____

Registration No.: _____

Branch: _____

Section _____