ME381 ROBOTICS

Experiment 5: Servo with microcontroller

Aim: Learn About working of servo motor and its libraries

This module aims to facilitate the understanding and practical implementation of servo motors, both with and without the use of libraries. A servo motor is a closed-loop actuator that utilizes Pulse-Width Modulation (PWM) signals to control its angular position precisely. Refer to Figure 1 for a visual representation of the basic structure of a servo motor.

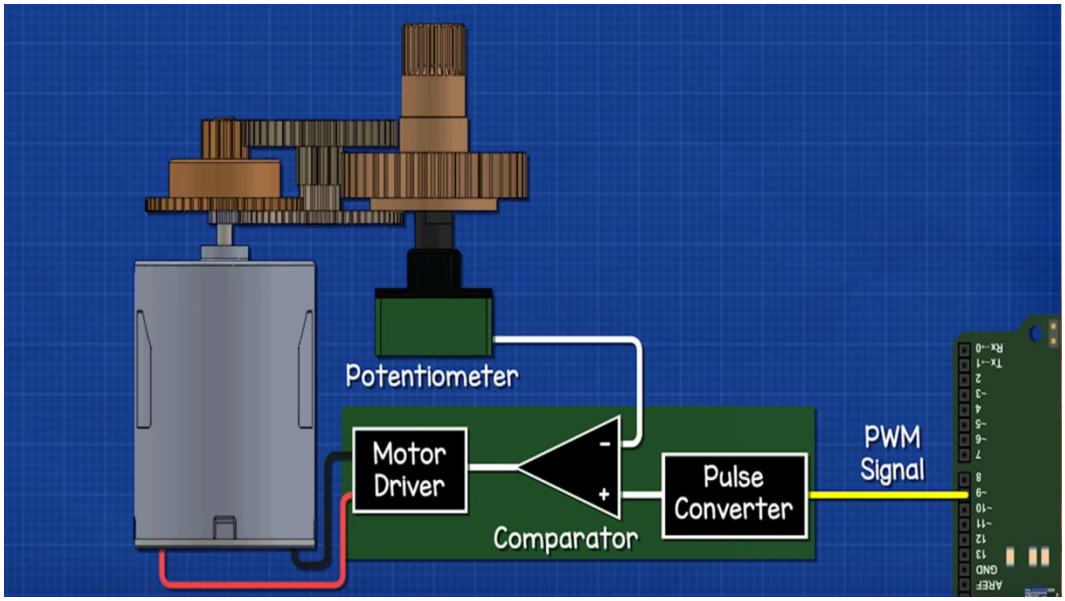
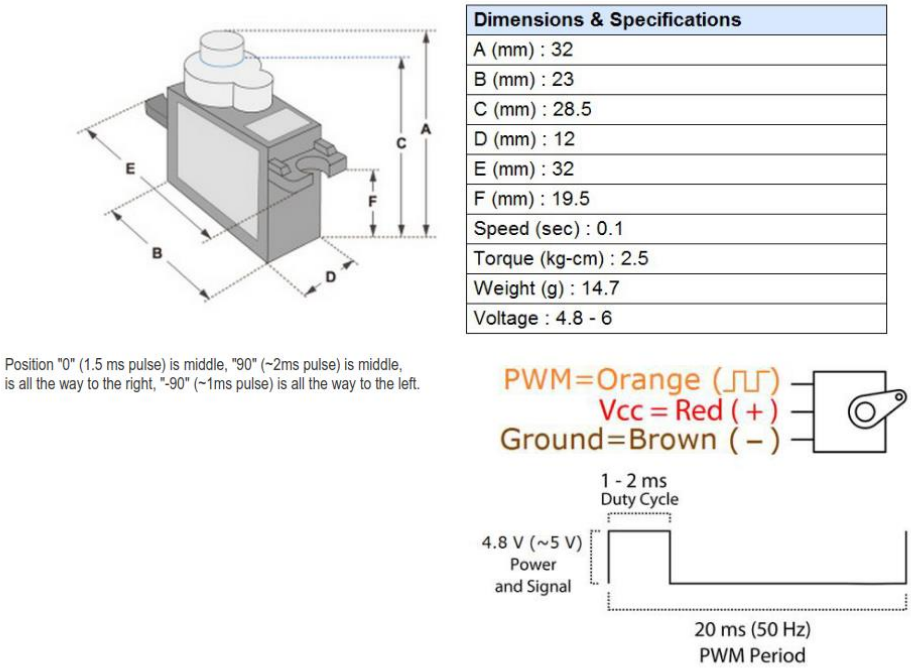


Figure 1 Basic structure of Servo motor.

The servo motor utilized in our experiment is the SG90 micro servo. Unlike DC motors where PWM signals primarily control speed by varying the effective voltage, servo motors interpret PWM signals differently. In the case of a servo motor, the PWM signal's duty cycle determines the angular position of the servo horn (output shaft). Refer to Figure 2, which displays the datasheet for the SG90 servo motor. As indicated, to achieve positions of 0°, 90°, and 180°, duty cycles of 1 ms, 1.5 ms, and 2 ms are required, respectively. It's important to note that the PWM period (time between pulses) remains constant at 20 ms (50 hz), whereas Arduino's PWM frequency is about 2ms(500hz)



Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

Figure 1 Datasheet of SG90.

In our experiment, we will explore two distinct methods for controlling the servo motor:

- Custom-Built PWM Signal: This approach involves generating the PWM signal directly using Arduino's digital output pins. It provides a deeper understanding of the underlying principles of servo control and highlights the distinction between manually crafting a PWM signal and the PWM signal generated by the **analogWrite()** function.
- Servo Library: Utilizing a pre-built servo library simplifies the control process by abstracting the low-level PWM generation.

The following table shows in brief the tasks to be performed in this lab. The details of these tasks and reporting for evaluation are given in the 'Tasks' section. The 'Task Manual' describes the detailed procedure to be followed for performing each task.

Sr. No.	Experiment name	Lab work to be performed in brief
1	Study the control of servo motors	Interface these sensors with the Arduino Uno microcontroller: 1. Control servo with custom build PWM signal 2. Control servo with prebuild libraries. 3. Control servo with potentiometer.

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.

Sr. no.	Topic	Task	Report
1	Control servo with custom build PWM signal	• Make the connections as illustrated in the 'Task Manual' and upload the program for Task 1. Observe the servo motor's rotation. • Within the code, locate the section responsible for commanding the servo to move to zero degrees. Comment out this portion of the code to disable its execution. • Similarly, remove the comment from the section that sets the duty cycle to 2.3 for 180-degree rotation. • Change the signal pin from its current pin to pin 8 and update the code accordingly.	1(a). For what duty cycle does the servo head turn 45°? *Please provide the modified code that demonstrates this 45° rotation. 1(b). Does changing the signal pin to pin 8 (a non-PWM pin) affect the servo's rotation? If not, explain why. 1(c). Does adding a delay impact the servo's rotation? If yes, provide a reason for this behaviour.

		• Introduce a 200ms delay at the end of the loop() function and observe any changes in servo behaviour	
2	Control servo with prebuild libraries.	• Upload the program for Task2 • Modify the code to change the angle of the servo • Add delay of 200ms at the end of the loop function and observe the change. • Add two different angle in one loop	2(a) Clearly document the specific code modifications you made to alter the servo's angle. 2(b). Does adding a delay impact the servo's rotation? If not, provide a reason for this behaviour. 2(c) Capture and report the position of the servo head for different position.
3	Control servo with potentiometer.	• Make the connections as shown in the 'Task Manual' and upload the program for Task 3. • Modify the code so that when the potentiometer is rotated clockwise, the servo rotates counterclockwise.	3(a) Report the observation 3(b) Clearly document the specific code modifications you made

Task manual:

1. **Control servo with custom build PWM signal**

   Make the connections of Arduino with the servo as follows. Upload the Task1 code and perform the tasks given in the 'Tasks' table.

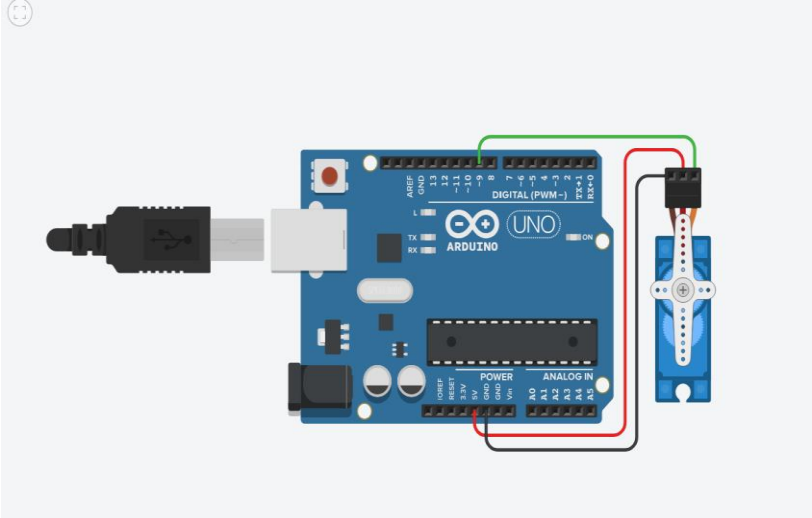


Figure 3 Servo connected to Arduino

2. Control servo with prebuild libraries.

Make the connections same as Task1. Upload the Task 2 code and perform the tasks given in the 'Tasks' table.

3. Control servo with potentiometer

Make the connections of Arduino with servo and pot as follows. Upload the Task 3 code and perform the tasks given in the 'Tasks' table.

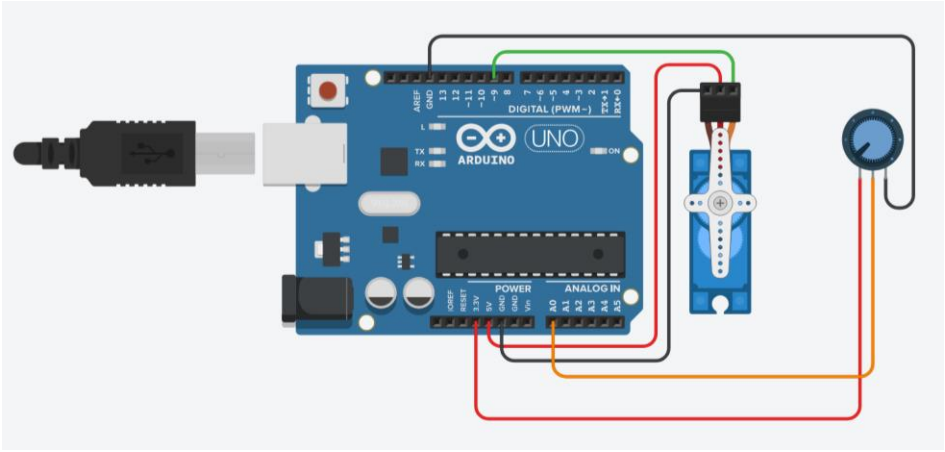


Figure 4 Servo and Potentiometer connected to Arduino

REPORT FORMAT

(Submit a single pdf file of your report to Hello IITK after renaming the file in the format: ME381_E3_Name_RollNo_GroupNumber)

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			

Results for individually assigned task:

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