Part 2

lesson 8

Servo

Overview

Servo is a type of geared motor that can only rotate 180 degrees. It is controlled by sending electrical pulses from your UNO R3 board. These pulses tell the servo what position it should move to. The Servo has three wires, of which the brown one is the ground wire and should be connected to the GND port of UNO, the red one is the power wire and should be connected to the 5v port, and the orange one is the signal wire and should be connected to the Dig #9 port.

Component Required:

(1) x Elegoo ESP32

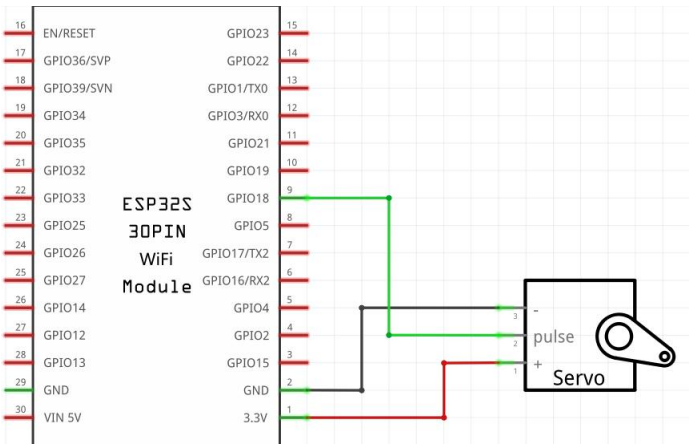
(1) x Servo (SG90)

(3) x M-M wires (Male to Male jumper wires)

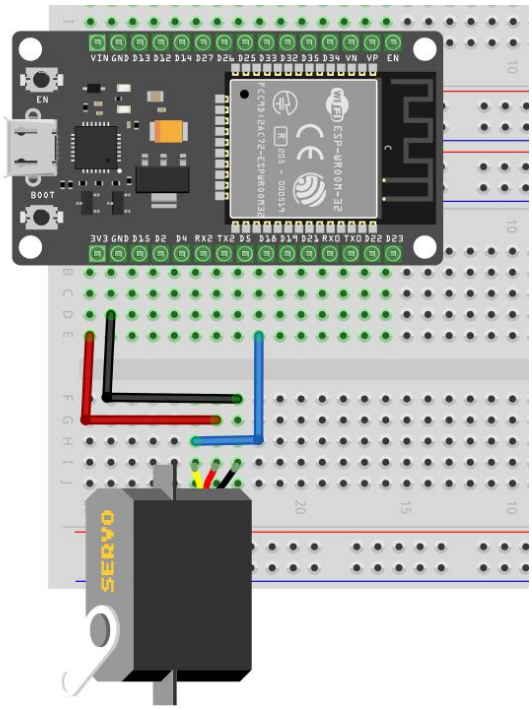
Component Introduction

SG90

- Universal for JR and FP connector
- Cable length : 25cm
- No load; Operating speed: 0.12 sec / 60 degree (4.8V), 0.10 sec / 60 degree (6.0V)
- Stall torque (4.8V): 1.6kg/cm
- Temperature : -30~60'C
- Dead band width: 5us
- Working voltage: 3.5~6V
- Dimension : 1.26 in x 1.18 in x 0.47 in (3.2 cm x 3 cm x 1.2 cm)
- Weight : 4.73 oz (134 g)



Connection Schematic



Wiring diagram

Code

After wiring, please open the program in the Folder **Servo** where the course is located and click UPLOAD to upload the program. See Lesson 5 in part 1 for details about program uploading if there are any errors.

Before you can run this, make sure that you have installed the **<Servo>** library or re-install it, if necessary. Otherwise, your code won't work.

For details about loading the library file, see Lesson 5 in part 1 too.

Part 1: Library Inclusion & Object Declaration

```
#include <Servo.h>  // Include the Servo library
Servo myservo;      // Create a Servo object named "myservo"
```

Library Inclusion (#include <Servo.h>): The Servo library is an official Arduino library designed to simplify servo motor control. It provides pre-defined functions (e.g., attach(), write()) to handle the PWM (Pulse-Width Modulation) signals required for servo operation, eliminating the need to manually generate complex timing signals.

Object Declaration (Servo myservo): In object-oriented programming, Servo is a "class" (a template for creating objects) that encapsulates the functions and properties needed to control a servo. By declaring Servo myservo, we create an instance (object) named myservo that can use all the built-in functions of the Servo class to control a specific servo motor.

Part 2: setup() Function (Initialization)

```
void setup(){
  myservo.attach(18);  // Connect the servo's control pin to Arduino pin 18
  myservo.write(90);   // Set the servo to the center position (90 degrees)
}
```

setup() Function Role: The setup() function runs once when the Arduino board is powered on or reset. It is used for initialization tasks such as pin configuration and setting initial states.

myservo.attach(18): This function associates the myservo object with Arduino pin 18.

myservo.write(90): The write() function sets the servo's rotation angle. Most standard servos have a rotation range of 0° (minimum) to 180° (maximum), with 90° being the center position. This line initializes the servo to the neutral position when the board starts.

Part 3: loop() Function (Continuous Execution)

loop() Function Role: The loop() function runs repeatedly after the setup() function completes. It contains the main logic for the program's continuous operation.

```
void loop(){
  myservo.write(90);   // Rotate to center (90°)
  delay(1000);         // Wait for 1 second (1000 milliseconds)
  myservo.write(60);   // Rotate to 60° (left of center)
  delay(1000);         // Wait for 1 second
  myservo.write(90);   // Rotate back to center (90°)
  delay(1000);         // Wait for 1 second
  myservo.write(150);  // Rotate to 150° (right of center)
  delay(1000);         // Wait for 1 second
}
```

The delay() function pauses the program execution for the specified time (in milliseconds). This ensures the servo has enough time to reach the target angle before moving to the next position.