



### Project Objectives:

By the end of the project, students should be able to:

- (a) Gather the components and parts needed, and construct a schematic of circuit (Design)
- (b) Assemble a robotic gripper, capable of performing simple actions (Build)
- (c) Adjust the sketch to perform desired outcome (Code)
- (d) Fine-tune the robotic gripper capability (Troubleshoot)

School: **Admiralty Secondary School**

Name: ( )

Class:

## Project Assignment: Robotic Gripper

In the subsequent lessons, we will learn, build and code a simple robotic gripper, capable of performing a few movements, using readily available materials in the school.

### A. Introduction

Just because it does not move, it does not mean it is not a robot. Some of the world's most sophisticated bots are nailed to the floor. They're robotic arms, used for such jobs as building cars, playing a mean game of chess, and even delicate surgery. The robot arm is a science unique to itself, with its own special set of challenges, techniques and solutions.

### B. Robot Arm Movement

Robotic arm comes in many movements. They are defined by the *coordinate* system. For example, the human arm is said to have *revolute* coordinates. The figure on the right (*Figure 1*) describes the four (4) basic robotic arm motions.

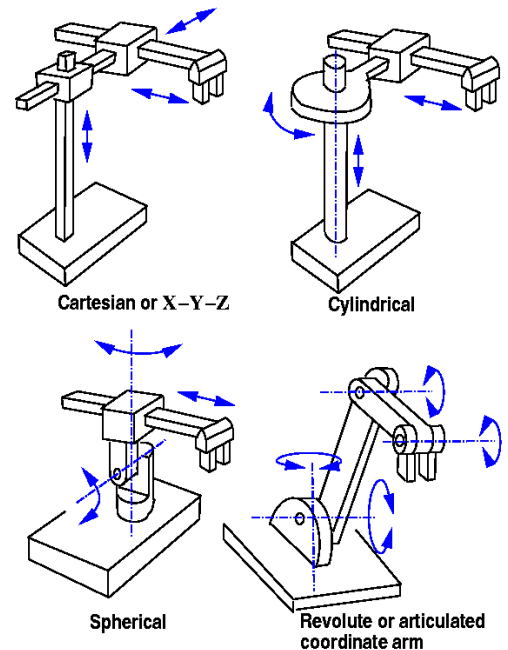


Figure 1: Four Basic Motion of a Robot Arm

### C. Design. Build. Code. Troubleshoot.

In the construction of any mechanical system, it is an excellent practise to start with an idea. This idea will then seeded to grow and develop, and take form into a more concrete vision of the end product. The initial process will be the most crucial – **Design**.

Design takes into consideration of many spheres of options: defining, research and ideation. Each has it unique process of consideration. As a person who builds, you need to ask yourself a few questions such as: what do you want to create, who is the target audience, and how will it work.

Information such as materials selection, mechanisms, dimensions, circuitries (if needed), and how each part assembles together is important. Once a concrete idea is formed, the next crucial part, is the to put together a prototype – **Build**.

Often, most mechanical system has circuitries integrated into them. Rarely, can you find a mechanical system, which exists on its own. These circuitries will require a microcontroller to be the 'brain' of the system. In this project, the Arduino board will be used. Sketch lines will be written so that the robot act upon how it is designed for - **Code**.

Once the integration of an eletro-mechanical system is complete, the program will be tested. Do not worry if the program does not work the first time. Even experienced builders and programmers face the exact same situation. Calmly, read through the program step by step, and correct errors along the way – **Troubleshoot**.

**C1. Design**

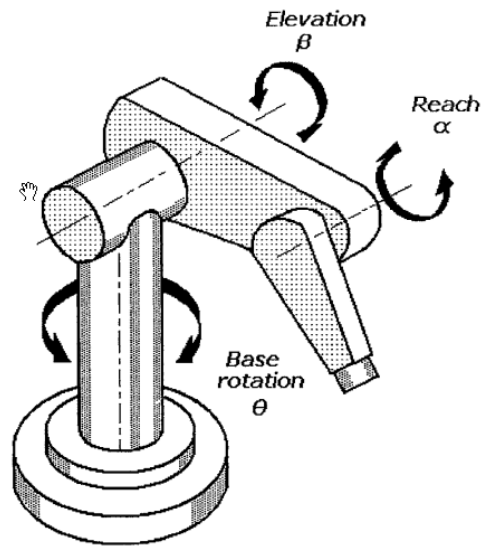
Mentioned in Section B, the robotic gripper uses a semi-cylindrical, semi-revolute coordinate. Figure 2 is a sketch as well as the working motion of the arm.

Ideation

The arm that we will be building uses a semi-revolute coordinate. It will operate on four (4) servomotors, with 4 Degree-of-Freedom (DoF):

- The base of the motor swivels left or right
- The elevation adjust the height of the gripper
- The reach controls how far it can reach protrude
- The gripper grabs the object.

It has 2 joints (ie. elevation and reach joints), 1 base, and a gripper. It is also a good practise to visualise the movement of the arm.



**Figure 2:** Sketch and Motion of arm

At this stage, it is also wise to pre-determine the materials for the arm, actuation and mounting method.

Actuation

Type : Servomotor  
Specification : Able to lift 10~50g of weight

Material

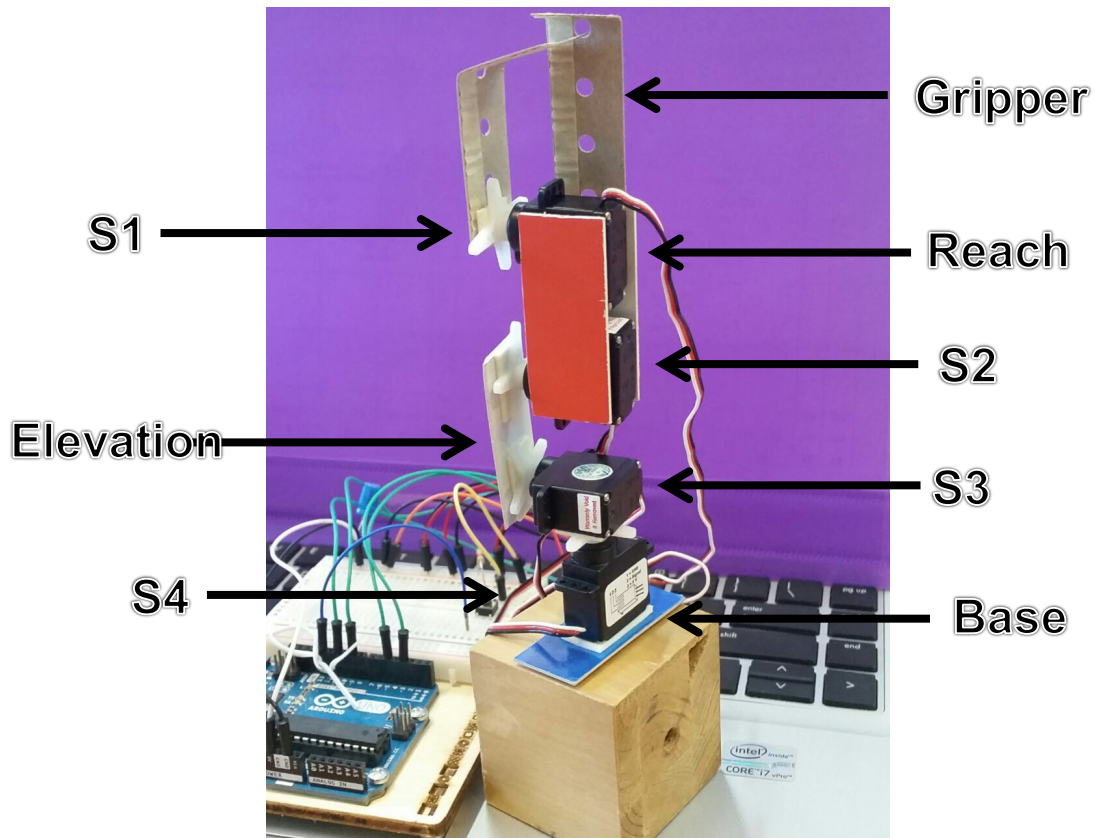
| Part      | Material  | Dimension | Quantity | Remarks       |
|-----------|-----------|-----------|----------|---------------|
| Gripper   | Cardboard | 190x17mm  | 1 pc     | Cut into half |
| Reach     | PVC       | 50x20mm   | 2 pc     |               |
| Elevation | PVC       | 50x20mm   | 1 pc     |               |
| Base      | Acrylic   | 50x50mm   | 1 pc     | Square base   |

Mounting

Type : Adhesive tapes  
Specification : 3M double-sided (1mm thickness) tape




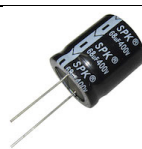


**C2i. Build (Mechanical)**

Follow the following steps to assemble the robotic arm.



**C2ii. Build (Electronic)**

What you'll need

|              |   |                      |   |
|--------------|---|----------------------|---|
| Breadboard   |  | Servo motor (x4)     |  |
| Jumper wires |  | Capacitor (470µF) X1 |  |
| Push Button  |  | Resistor (10 kΩ)     |  |

**Note:**

A capacitor has polarities. The longer lead is the (+) anode and should be connected to the +5V.

Identifying the Pins

A. Identify the Pulse Wave Modulation (PWM) pins on your breadboard. Shade the pins numbers in the boxes below:

|   |                 |   |   |   |   |   |   |   |   |   |    |    |    |    |
|---|-----------------|---|---|---|---|---|---|---|---|---|----|----|----|----|
|   |                 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| A | Shade PWM       |   |   |   |   |   |   |   |   |   |    |    |    |    |
| B | Tick pins       |   |   |   |   |   |   |   |   |   |    |    |    |    |
| C | Servo Indicator |   |   |   |   |   |   |   |   |   |    |    |    |    |
| D | Push Button     |   |   |   |   |   |   |   |   |   |    |    |    |    |

- B. Identify four (4) PWM pin to be used for the robotic arm with a tick (✓).
- C. Label each pin with the name of the Servo (e.g.S1, S2, S3, S4). It is a good practise to label the servos in order, in the table provided.
- D. Identify a non-PWM pin for your push button and label accordingly on the table provided.

Connect it up!

Draw the wiring of the servos, push button and the Arduino on the breadboard. You may refer to Lessons 3 & 5 or refer to the Arduino blog. Ensure that your Arduino is powered off. Get the Educator to check your drawing before connecting.

**Remember...**  
 Yellow : Signal  
 Red : +5V  
 Black : GND

**Note 1:**

| Servomotor wires | Diagram Wires | Pin                     |
|------------------|---------------|-------------------------|
| Orange           | Yellow (1st)  | Signal (connect to pin) |
| Red              | Red (2nd)     | +5V                     |
| Brown            | Black (3rd)   | GND                     |

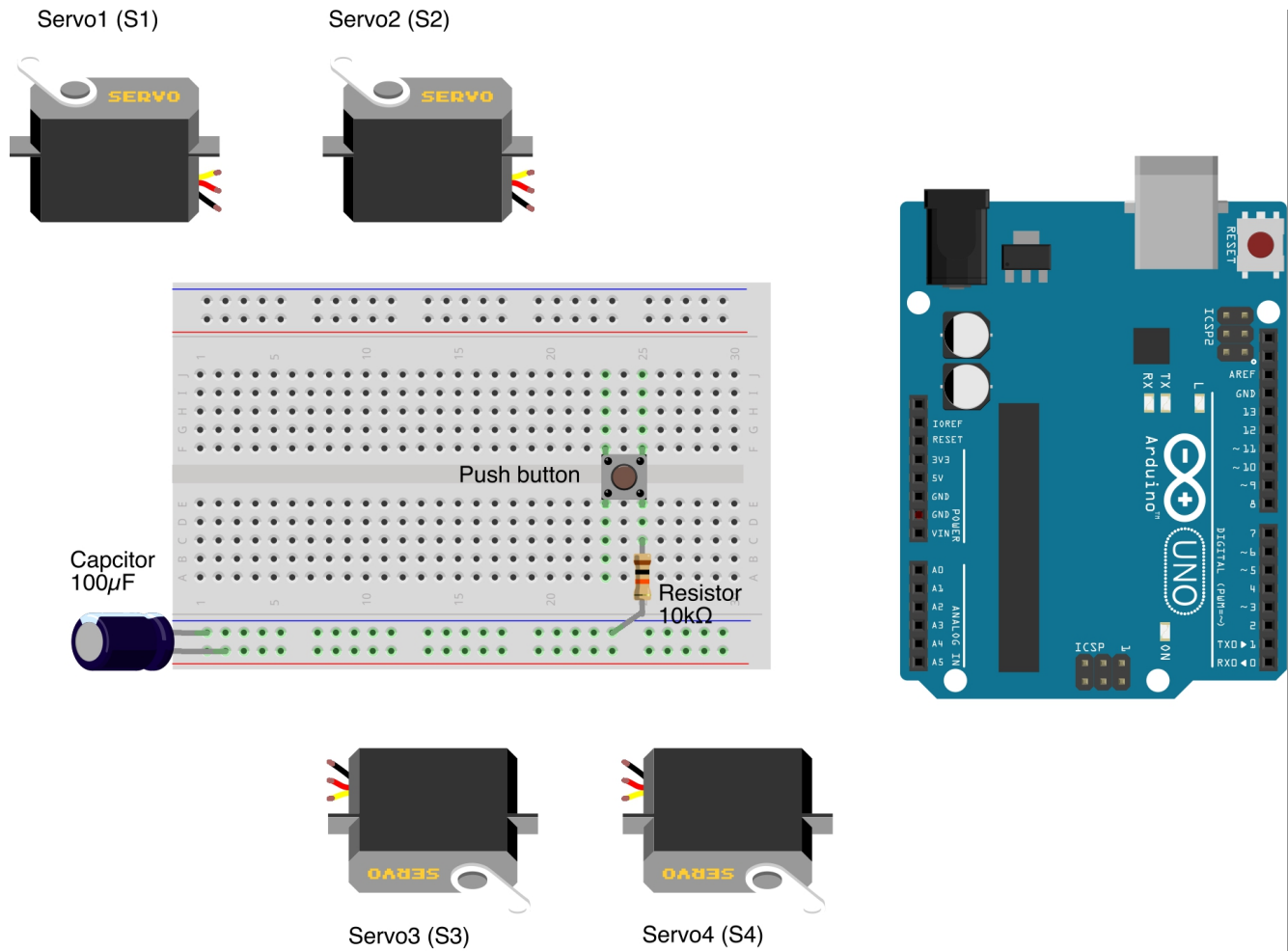
**Note 2:**

The robotic arm needs to be **vertically straight** (see diagram on page 4) upon sketch initialisation. If your arm is not vertically straight, remount the **REACH** and **ELEVATION** arms using adhesive tapes.

**Note 3:**

Watch the 'video 1' footage on [arduinoandme.weebly.com](http://arduinoandme.weebly.com) website.

Observe and plan your arm movement before keying the code onto you sketch. Use the comments on each line to aid your sketch lines.



**Figure 3: Circuit map of the robotic arm**

### **C3. Code**

A few simple sketch lines will integrate the mechanical and electronic systems in harmony. Coding is as important as Building and Designing.

#### Allowable Angles & Commands

| Part      | Servo | Minimum Angle | Initial Angle | Maximum Angle |
|-----------|-------|---------------|---------------|---------------|
| Gripper   | S1    | 0             | 90            | 90            |
| Reach     | S2    | 30            | 90            | 150           |
| Elevation | S3    | 30            | 90            | 150           |
| Base      | S4    | 0             | 0             | 180           |

Lets get Coding!

```

#include <Servo.h>

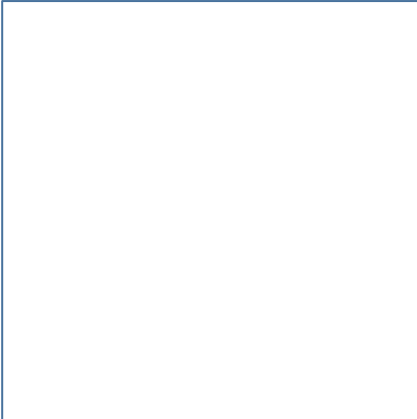
Servo S1;           // create servo object to control a servo
Servo S2;
Servo S3;
Servo S4;

const int button = 13; // button to pin 13
int currentState = LOW; // button is not pressed

void setup()
{
  S1.attach(3);      // assign S1 to Pin 3
  S2.attach(5);      // assign S2 to Pin 5
  S3.attach(9);      // assign S3 to Pin 9
  S4.attach(11);     // assign S4 to Pin 11

  S1.write(90);      // set servo at 90 deg position
  S2.write(90);      // set servo at 90 deg position
  S3.write(90);      // set servo at 90 deg position
  S4.write(0);       // set servo at 0 deg position

  pinMode(button, INPUT); // set the push button as INPUT
  delay(3000);          // pause for 3 secs
}

void loop()
{
  currentState = digitalRead(button); // read the state of button
  if (currentState == HIGH)          // if the button is pressed
  {

    // 30 < angle < 150
    // pause for 1 sec
    // 30 < angle < 150
    // pause for 1 sec
    // 0 < angle < 90
    // pause for 1 sec
    // 0 < angle < 180
    // pause for 1 sec
    // 0 < angle < 90
    // pause for 3 sec
  }
}

```



```

else // if the button is depressed (reset)
{
  // 0 < angle < 120
  // pause for 0.5 sec
  // 30 < angle < 180
  // pause for 0.5 sec
  // 0 < angle < 180
  // pause for 0.5 sec
  // 0 < angle < 180
  // pause for 0.5 sec
}
}

```

#### **4. Troubleshoot**

You are just moments away to witness your very own robotic arm at work.

**Step 1:** Click the **Verify** button (to check for errors)

**Step 2:** Click the **Upload** button.

#### **Lets Think!**

1. Were there errors upon verifying your program? How do you correct it?

---



---

2. Is your arm able to move upon uploading your sketch? What was your greatest challenge?

---



---

3. Could you identify what part(s) went wrong? Discuss with your Educator.

#### **Challenge Yourself (Optional)**

1. Add another switch to your circuit.  
Program another set of instructions to perform another movement shown in video 2.

**GOOD LUCK!!**

Thank you.

📌 Please upload 'File > Examples > 01.Basics > Blink' at the end of every class