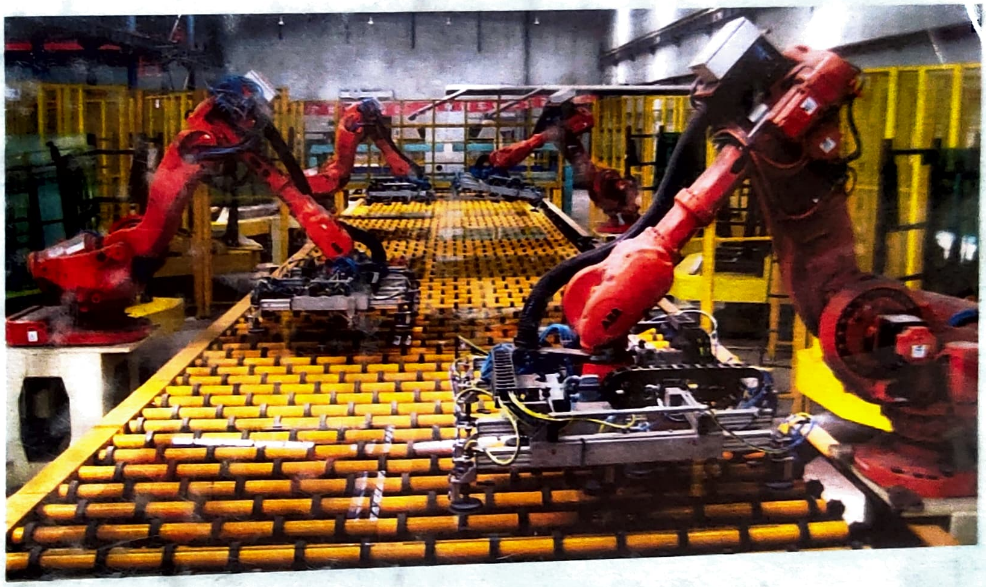
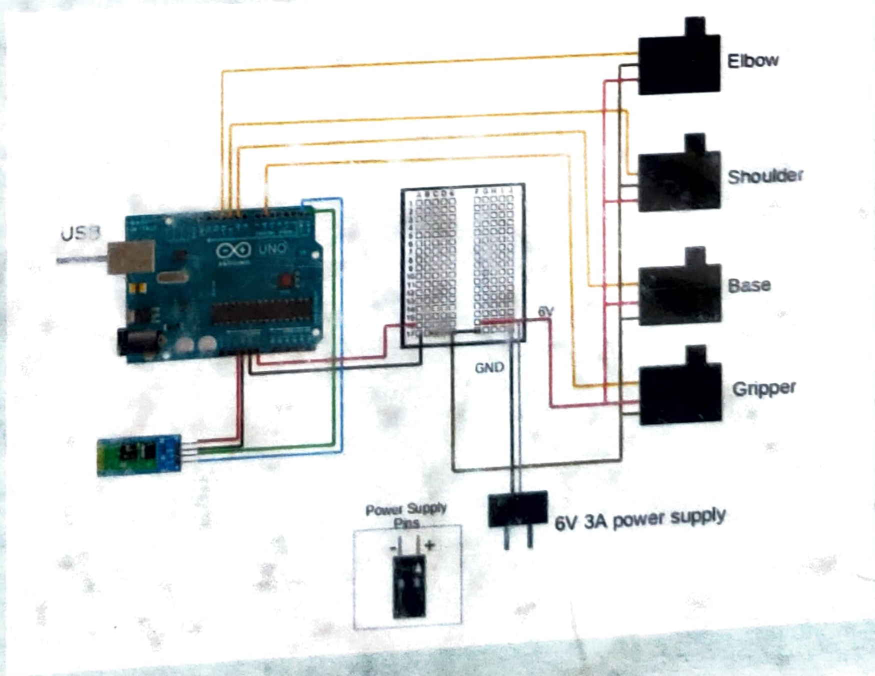
2021

# MATERIAL HANDLING ROBOTIC ARM USING AURDINO AND SERVO MOTORS

PROJECT REPORT

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@






****************************************

ELECTRONICS & COMMUNICATION

1/1/2021

# ACKNOWLEDGEMENT

*Any achievement big or small should have a catalyst and a constant encouragement and advice of valuable noble minds for out effort to bring out this report work. The satisfaction that accompanies the successful culmination of any task, would be incomplete without mentioning those who made it possible. Success is the epitome of hard work, determination, concentration and dedication. Those list of thanks is heartfelt only begins acknowledge those people to whom one way or other this owes its existence.*

*We are expressing our sincere thanks to our beloved project guide **and** lecturer of **Electronics & Communication** Department for his valuable guidance and extended co-Operation in preparing this project report.*

*We will also thankful to our beloved **H.O.D. Mrs. MUMTAZ BEGUM.***

*Who inspired us to take up this report by giving valuable guidance and encouragement.*

*We wish to thank the **principal** of our college for providing academic climate, which made this project Report successful.*

*We are thankful to our friends who have helped us as in preparation of the project Report*


S.J.M.
VIDYAPEETHA(R.)
S.J.M. POLYTECHNIC(Aided)
SJMP
CHITRADURGA
WORK IS WORSHIP

# DEDICATED

THE CO-OPERATION INVOLMENT, ACHIEVEMENT AND WHOLE EFFORT AND THE PROJECT IS HEARTLY DEDICATED TO OUR LECURERS OF ELECTRONICS AND COMMUNICATION DEPARTMENT, OUR BELOVED PARENTS AND OUR FRIENDS.

GOVERNMENT OF KARNATAKA

DEPARTMENT OF TECHNICAL EDUCATION

A PROJECT REPORT ON

# MATERIAL HANDLING ROBOTIC ARM USING AURDINO AND SERVO MOTORS

**2020-2021**

***A PROJECT REPORT SUBMITTED IN PRACTICAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF***

**DIPLOMA IN ELECTRONICS & COMMUNICATION ENGINEERING**

*SUBMITTED BY:*


MOHAMMED SHARUK. M.S
MOHAMMED MUSSAVIR
KRISHNA MURTHY. M
GIRI PRASAD.N
DHANUSH KUMAR. K
FAYAZ. H
KARIBASAPPA. C
SACHIN. J
BASAVARAJ.C

**Under the guidance of:**
**Mrs. PRATHIMA. J**
B.E (E&C), MTech

S.J.M. VIDHYAPEETHA (REGD.)

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGG.

S.J.M. POLYTECHNIC (AIDED)

CHITRADURGA – 577 502

KARNATAKA

# GOVERNMENT OF KARNATAKA
# DEPARTMENT OF TECHNICAL EDUCATION

**BOARD OF TECHNICAL EXAMINATIONS,**
**PALACE ROAD,**
**BANGALORE-01**

## S.J.M. POLYTECHNIC (AIDED)
## CHITRADURGA-577502

**DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING**

**2020-2021**

## *CERTIFICATE*

***This is to certify that, Mr. MOHAMMED SHARUK. M.S has successfully completed the Project work entitled, MATERIAL HANDLING ROBOTIC ARM USING AURDINO AND SERVO MOTORS of Sixth semester Diploma in E&C Engineering for the practical fulfillment of the requirement for the award of Diploma in E&C Engineering, Board of Technical Education Bangalore, Karnataka, for the academic year 2020-2021***

**Under the guidance of:**

Prathima J

**Mrs. PRATHIMA. J**
**B.E (E&C),**

Mumtaz B
..........................................
**Mrs. MUMTAZ BEGUM**
**B.E (E&C)**
**H.O.D. Dept. of E&C Engineering.**

S.J.M. POLYTECHNIC CHITRADURGA
WORK IS WORSHIP

..........................................
**Mr. DV MURUGESH**
**B.E (CIVIL),LLB**
**PRINCIPAL**

Date: 17/11/21

Name/Signature of Examiners

1)..........................................

(..........................................)

2)..........................................

(..........................................)

Reg. no: 321EC18020

Examination Centre: S.J.M POLYTECHNIC (AIDED) CHITRADURGA.

Robotic ARM using AURDINO & SERVO MOTORS.

## ABSTRACT

In a place where science fiction of yesterday is rapidly becoming reality of today. The world needs less physical interaction and more productivity. An android device controlled robotic arm via Bluetooth wireless technology is presented here. Such device is very much required where human reach is not possible or must be avoided. Also, the cheap and robust design of this project makes it a very useful tool in multiple industries.

## Table of Contents

## INTRODUCTION

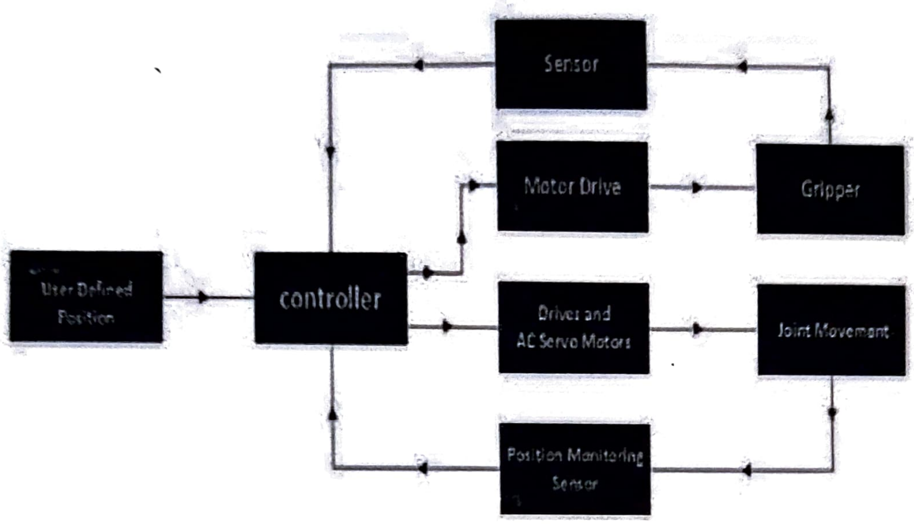
**Android based robotic arm is designed to provide access to places where human presence must be avoided such as places with very low or very high atmospheric pressure, war zones and even biohazardous places. It can also be used for repetitive and cumbersome works like automobile painting, assembly work in manufacturing processes.**

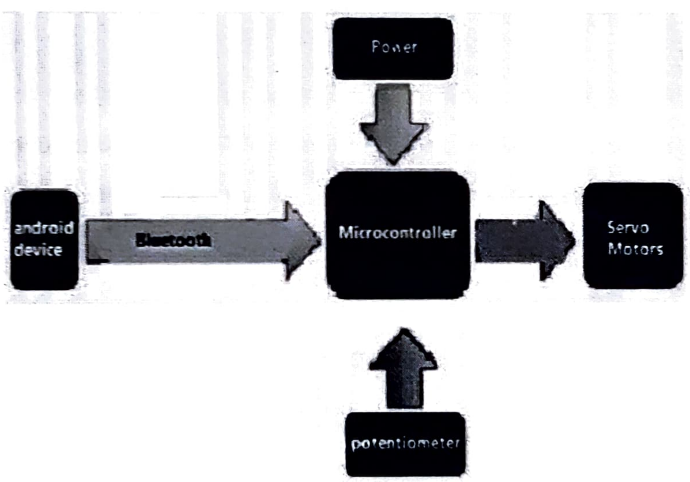
**The system uses an Arduino UNO board that controls the movement robotic arm through Servo motors. The Arduino uno board is connected with a Bluetooth module for wireless connection of android device. An android app is used to provide commands to the robotic arm. This gives the user to work on particular task from a safe distance.**

## LITRATURE REVIEW

**Many automated and advanced robotic arm system has been developed and found applications in industries for welding, material handling, painting, laser engraving, palletizing and assembly work. But these arms are either not controlled remotely or have fixed set of commands. To overcome this limitation an HC-05 Bluetooth module is used was useful in understanding the functioning and control of robotic arm. A pick-and-place robot.**

## BLOCK DIAGRAM



**ardware**

# COMPONENTS USED

**Table -1: System Components**

| S. No. | Component name | Quantity |
|---|---|---|
| 1 | Arduino UNO | 1 |
| 2 | MG90S Metal Gear Servo Motor | 4 |
| 3 | HC-05 Bluetooth Module | 1 |
| 4 | Robotic arm | 1 |
| 5 | Connecting wires | - |
| 6 | 12volt battery | 1 |
| 7 | Android device with custom application installed | |

- Software requirements:
- 1) AURDINO SOFTWARE

## Programming part

### Code

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>

#include <Servo.h>

double angle_rad = PI/180.0;
double angle_deg = 180.0/PI;
Servo servo_6;
Servo servo_9;
Servo servo_10;
Servo servo_11;

void setup(){
    Serial.begin(115200);
    Serial.begin(9600);
    servo_6.attach(6); // init pin
    servo_9.attach(9); // init pin
    servo_10.attach(10); // init pin
    servo_11.attach(11); // init pin
    pinMode(12,OUTPUT);
    pinMode(12,INPUT);
    servo_6.write(90); // write to servo
    servo_9.write(90); // write to servo
    servo_10.write(90); // write to servo
    servo_11.write(90); // write to servo
    Serial.println("SJM  POLYTECHNIC");
    _delay(1);
    Serial.println("E&C DEPARTMENT ");
    _delay(1);
    Serial.println("ROBOTIC ARM PROJECT");
    _delay(1);
    Serial.println("GUIDED BY PRATHIMA MADAM");
    _delay(1);
    Serial.println(        );
    digitalWrite(12,1);
```

```
void loop(){
  digitalWrite(12,1);
  if(((digitalRead(12))==(0))){
    servo_9.write(90); // write to servo
    Serial.println("PICKING PART");
    _delay(1);
    servo_6.write(180); // write to servo
    _delay(3);
    servo_11.write(135); // write to servo
    _delay(3);
    servo_10.write(135); // write to servo
    _delay(3);
    servo_11.write(90); // write to servo
    _delay(3);
    Serial.println("PART PICKED");
    servo_10.write(90); // write to servo
    _delay(3);
    servo_6.write(135); // write to servo
    _delay(3);
    servo_10.write(135); // write to servo
    Serial.println("WELDING IN PROGRESS MACHINE 1");
    _delay(10);
    servo_10.write(90); // write to servo
    Serial.println("WELDING COMPLETE IN MACHINE 1");
    _delay(3);
    servo_6.write(90); // write to servo
    _delay(3);
    servo_10.write(135); // write to servo
    Serial.println("WELDING IN  PROGRESS MACHINE 2");
    _delay(10);
    servo_10.write(90); // write to servo
    Serial.println("WELDING COMPLETE IN MACHINE 2");
    _delay(3);
    servo_6.write(0); // write to servo
    _delay(3);
    servo_10.write(135); // write to servo
    _delay(3);
    servo_11.write(135); // write to servo
```

```
    Serial.println("PART LOADED IN TRUCK ");
    _delay(3);
    servo_10.write(45); // write to servo
    _delay(3);
    servo_10.write(90); // write to servo
    _delay(3);
    servo_11.write(90); // write to servo
    _delay(3);
  }else{
    Serial.println("WAITING FOR PART");
    _delay(10);
  }
  _loop();
}

void _delay(float seconds){
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime)_loop();
}

void _loop(){
}
```

## *APPLICATIONS*

- Material handling in industries
- Robot welding
- Risk job
- Risk location
- High current application
- Pick and place application
- Accurate/ precision job

## ADVANTAGES

- **Grasping and holding objects and then move them to a new location, or mixing with other fluids. (used in laboratories that trust such arms to work within a toxic environment and so do not endanger the researcher. Building cars.**
- **Retrieving suspicious objects without endangering humans.**
- **Dig trenches.**
- **A source of entertainment and education.**
- **An appendage of an anthropocentric robot.**
- **Used in surgery.**
- **Used in farming.**

## Disadvantages and Limitations:-

- This project is a small scale production it can pick up only small and lighter objects.
- On large scale this project may become costly and its circuit complexity increases.
- On large scale may become hazardous due to uncontrollable robotic arm it can harm physically

APPLICATIONS:

The characteristics of a robotic arm are: - its extension: how far from its base it can operate - its positioning: can it control its wrist position, orientation, with what precision, what speed - the tools and objects it can carry

Therefore they can be used as: Painting (cars) - soldering (cars) - access unevenly placed parts (for scanning, selecting...) - pick and place (most industries, a lot for food industry). - act in a human-designed environment: send the arm on a mobile base to a damaged/radioactive building and use the arm to open the door and manipulate the tools (by itself or remote controlled)

Conclusion:

This proposed work is an overview of how we can make use of servo motor to make joints of a robotic arm and control it using potentiometer and arduino UNO. Also used for high loaded industrial application work joints; there are many joints . The design process is clearly explained in the next section with detailed information regarding the components which are used.

# CHAPTER 2: DESIGNING PORTION

## 2.1 Design of Robotic Arm

The Robotic Arm is designed using the Microcontroller i.e. ATMEGA328p Micro-controller using Arduino programming. This process works on the principle of interfacing servos and potentiometers. This task is achieved by using Arduino board. Potentiometers play an important role The remote is fitted with potentiometers and the servos are attached to the body of the robotic arm. The potentiometer converts the mechanical motion into electrical motion. Hence, on the motion of the remote the potentiometers produce the electrical pulses, which are in route for the Arduino board. The board then processes the signals received from the potentiometers and finally, converts them into requisite digital pulses that are then sent to the servomotors. This servo will respond with regards to the pulses which results in the moment of the arm.

**Figure 2.1 shows the image of a servo motor.** It consists of a motor which is coupled to a sensor, used for position feedback, through a reduction gearbox. It also accompanies a relatively sophisticated controller, usually a dedicated module designed specifically for use with servo motors

In short, the micro controller interfaces all these components specified below. A short list of components include

1. Servo motors
2. Potentiometers
3. Atmega 328p.
4. Arduino Deumilanove "IDE"
5. Bloutooth module.

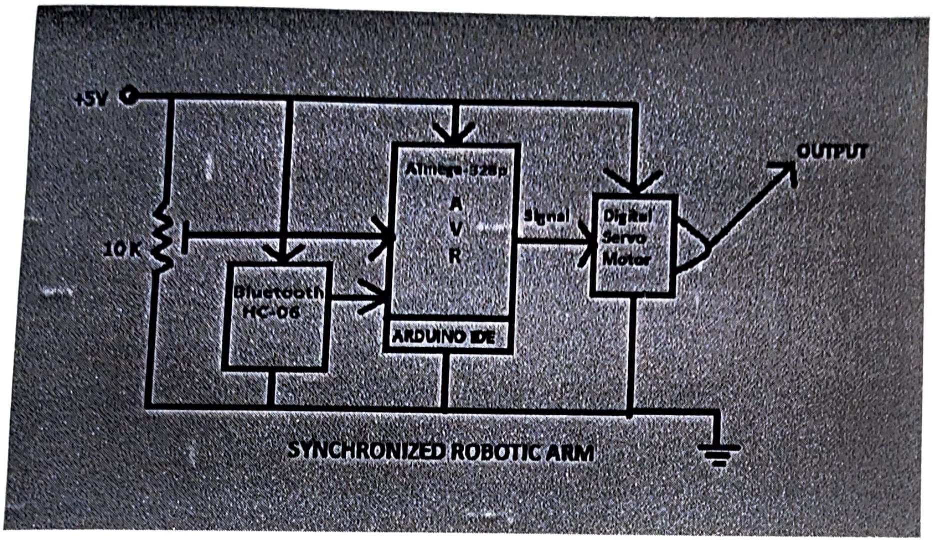



**Figure2.1 shows the image of a servo motor**

## 2.2 Five degrees of freedom

Serial and parallel manipulator systems are generally designed to position an end-effector with five degrees of freedom, consisting of three in translation and tow in orientation. This provides a direct relationship between actuator positions and the configuration of the manipulator

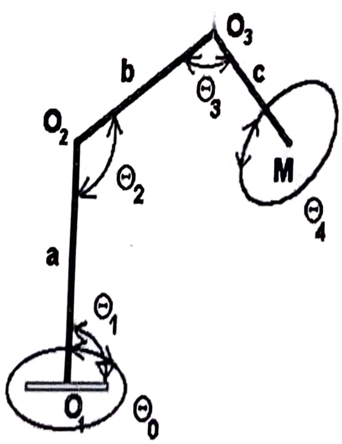


**Figure2.2 Five degrees of freedom**

Robot arms are described by their degrees of freedom. This number typically refers to the number of single-axis rotational joints in the arm, where higher number indicates an increased flexibility in positioning a tool. This is a practical metric, in contrast to the abstract definition of degrees of freedom which measures the aggregate positioning capability of a system.

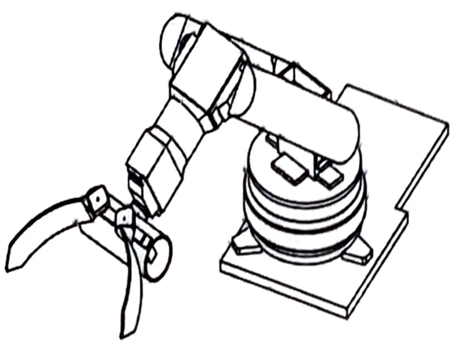


**Figure2.3 Robot Arm**

## 2.3 Positive and negative on robotic arm

### 2.3.1 The Positive:

- Increase productivity
- Use equipment effectively
- Reduce working costs
- Flexibility at work
- Get the job done in the shortest time
- Provide good returns on investment
- Better accuracy in performance
- Ability to work in risky ways and make it more safe

### 2.3.2 The negative:

- Cause unemployment for manual workers
- High initial cost
- designed Arm to perform specific tasks and not comparable to the human hand
- Difficulty programmed to perform Accurate tasks
- Needed a large number of sensors and high accuracy to perform the Complex tasks
- And other technical problems, "especially in the fields of artificial intelligence and Machine vision" .
- When the Robotic arm break down the production line will go off in the factories.

# CHAPTER 3: THEORETICAL PORTION

## 3.1 Arduino

Arduino Uno is a microcontroller board based on the ATmega328P . It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

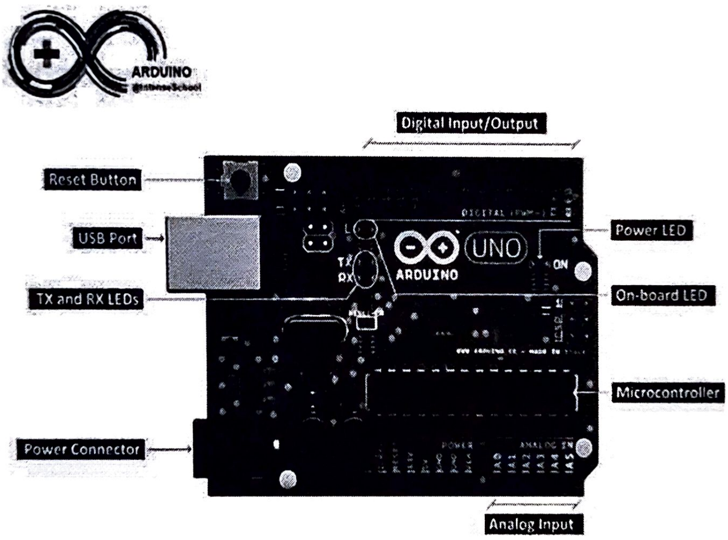



**Figure3.1 Arduino Uno microcontroller board (interface)**

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

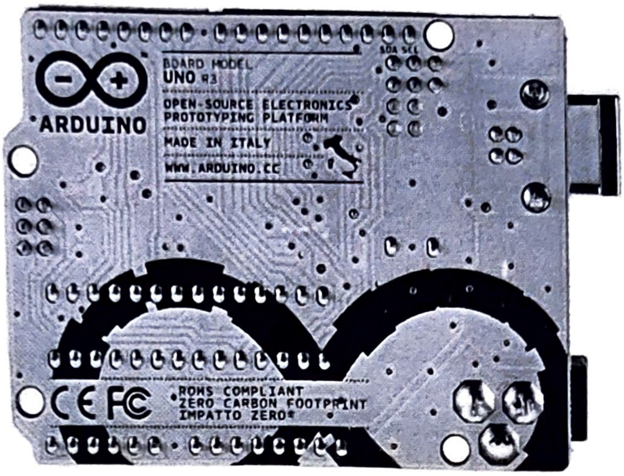



**Figure 3.2 Arduino Uno microcontroller board (back view)**

## 3.2 Specifications

| | |
|---|---|
| Microcontroller | ATmega328P |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limit) | 6-20V |
| Digital I/O Pins | 14 (of which 6 provide PWM output) |
| PWM Digital I/O Pins | 6 |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 20 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (ATmega328P) |

| | |
|---|---|
| | of which 0.5 KB used by bootloade*r |
| SRAM | 2 KB (ATmega328P) |
| EEPROM | 1 KB (ATmega328P) |
| Clock Speed | 16 MHz |
| Length | 68.6 mm |
| Width | 53.4 mm |
| Weight | 25 g |

## 3.3 Programming

The Arduino Uno can be programmed with the (Arduino Software (IDE)).The ATmega328 on the Arduino Uno comes preprogrammed with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol.

## 3.4 Power

The Arduino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

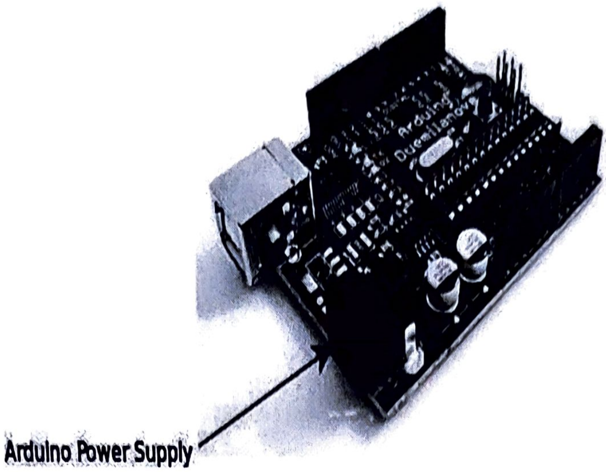



**Figure 3.3 Arduino Power Supply**

## 3.5 Memory

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM

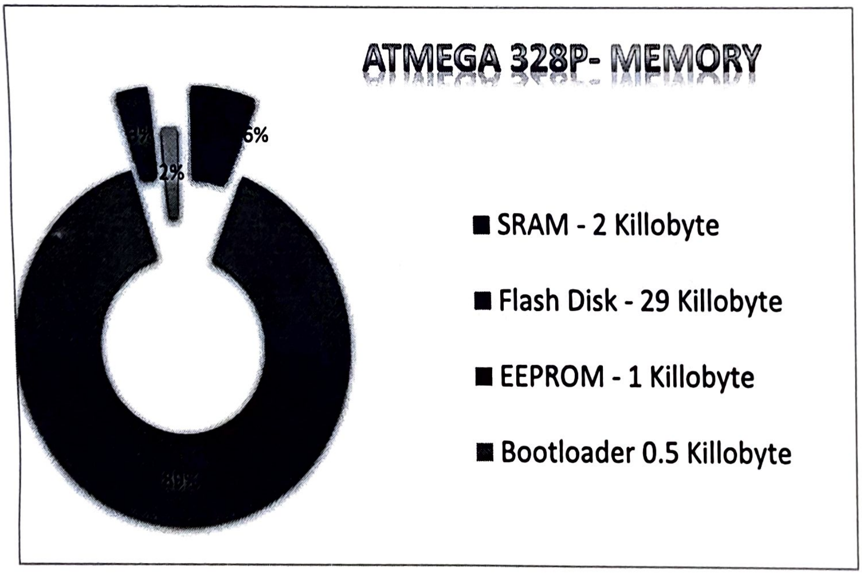



**Figure 3.4 ATmega328P – Memory**

## 3.6 Arduino development "IDE"

The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and is derived from the IDE for the Processing programming language and the Wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. There is typically no need to edit make files or run programs on a command-line interface

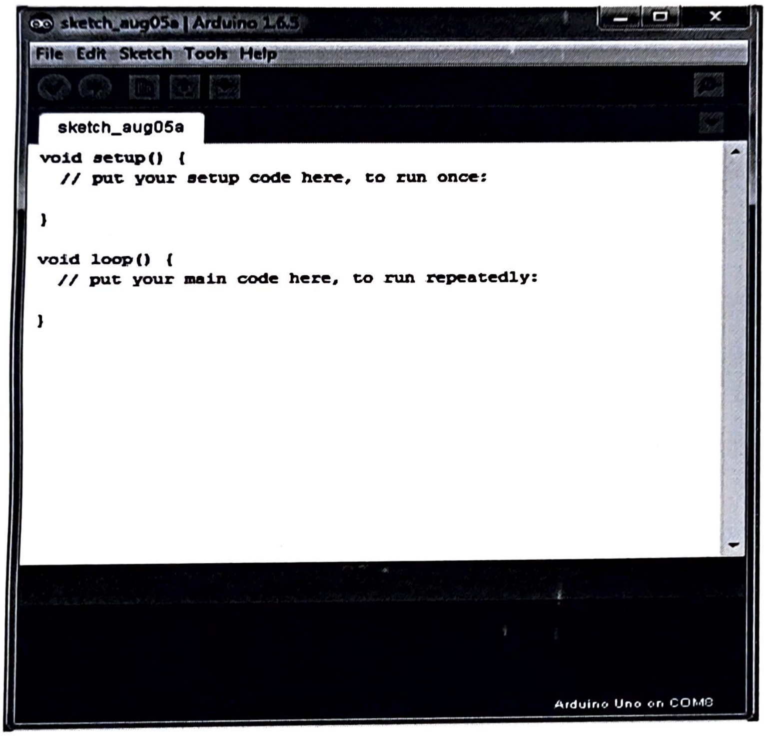



Figure 3.5 Interface Arduino Uno Program

| | |
|---|---|
| Developer(s) | Arduino Software |
| Stable release | 1.0.3 / December 10, 2012; 3months ago |
| Written in | Java, C and C++ |
| Operating system | Cross-platform |
| Type | Integrated development environment |
| Website | arduino.cc |

Arduino programs are written in C or C++ The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output.

Operations much easier. Users only need define two functions.

To make a runnable cyclic executive program:

- **Setup ():** a function run once at the start of a program that can initialize settings.
- **Loop ():** a function called repeatedly until the board powers off.

# CHAPTER 4: PRACTICAL PORTION

## 4.1 Servo Motors

Servo refers to an error sensing feedback control which is used to correct the performance of a system. Servo or RC Servo Motors are DC motors equipped with a servo mechanism for precise control of angular position.

The RC servo motors usually have a rotation limit from 90° to 180°. But servos do not rotate continually. Their rotation is restricted in between the fixed angles.

The Servos are used for precision positioning. They are used in robotic arms and legs, sensor scanners and in RC toys like RC helicopter, airplanes and cars.

The specifications for big Servomotor used are as follows:

- Weight- 55g
- Dimension- 40.7*19.7*42.9mm
- Stall torque- 10kg/cm
- Operating speed-0.20sec/60degree(4.8v)
- Operating voltage 4.8-7.2V.
- Temperature range 0-55 degrees.

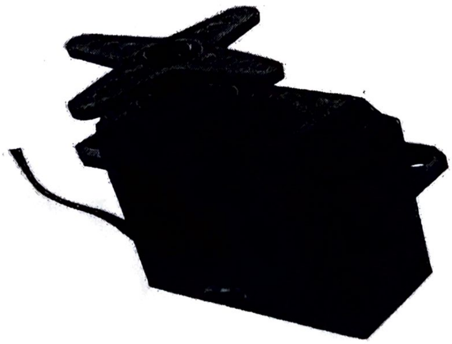


**Figure 4.1 Big Servo Motor**

The specifications for small Servomotor G9 used are as follows:

- Weight: 9 g
- Dimension: 22.2 x 11.8 x 31 mm approx.
- Stall torque: 1.8 kgf·cm • Operating speed: 0.1 s/60 degree
- Operating voltage: 4.8 V (~5V)
- Dead band width: 10 µs
- Temperature range: 0 °C – 55 °C

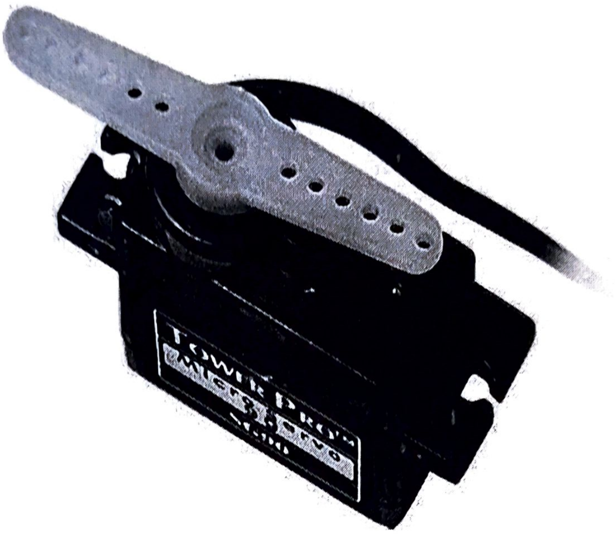


**Figure 4.2 G9 Servo Motor**

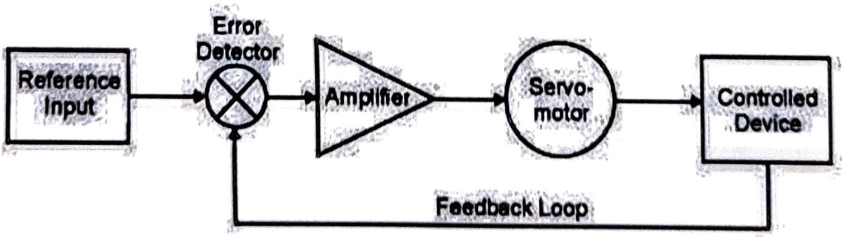



**Figure 4.3 Block diagram of a Servo motor**

## 4.2 Theory of DC Servo Motor

As we know that any electrical motor can be utilized as servo motor if it is controlled by servomechanism. Likewise, if we control a DC motor by means of servomechanism, it would be referred as **DC servo motor**. There are different types of DC motor, such shunt wound DC motor, series DC motor, Separately excited DC motor, permanent magnet DC motor, Brushless DC motor etc. Among all mainly separately excited DC motor, permanent magnet DC motor and brush less DC motor are used as servo.

## 4.3 Separately Excited DC Servo Motor

Figure (4.4) shows the block diagram of the separately excited Dc servo motor

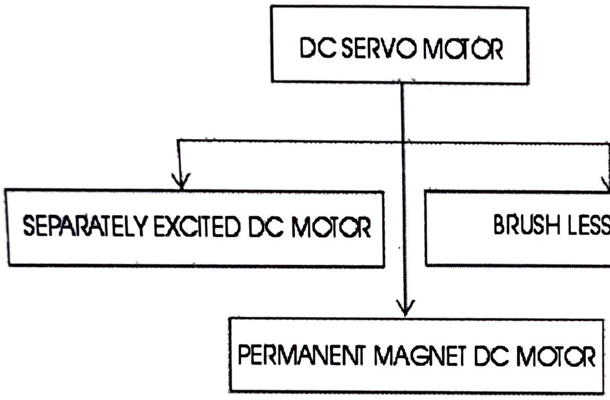



**Figure 4.4 Separately Excited DC Servo Motor**

### 4.3.1 DC Servo Motor Theory:

The motors which are utilized as **DC servo motors**, generally have separate DC source for field winding and armature winding. The control can be archived either by controlling the field current or armature current. Field control has some specific advantages over armature control and on the other hand armature control has also some specific advantages over field control. Which type of control should be applied to the **DC**

**servo motor**, is being decided depending upon its specific applications. Let's discus **DC servo motor working principle** for field control and armature control one by one.

**Field Controlled DC Servo Motor Theory**

The Figure below illustrates the schematic diagram for a field controlled DC servo motor. In this arrangement the field of DC motor is excited be the amplified error signal and armature winding is energized by a constant current source

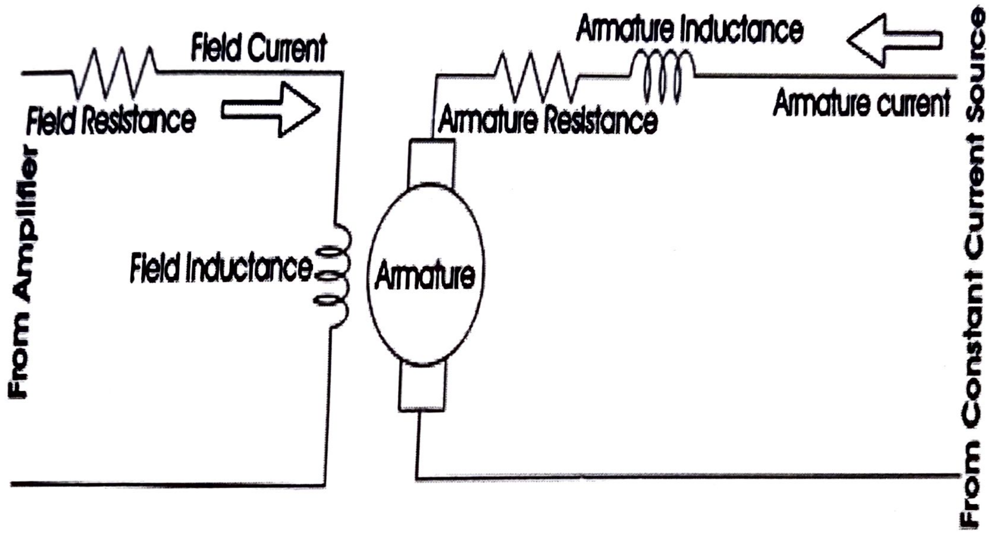



**Figure 4.5 Field controlled DC servo motor**

The field is controlled below the knee point of magnetizing saturation curve. At that portion of the curve the MMF linearly varies with excitation current. That means torque developed in the DC motor is directly proportional to the field current below the knee point of magnetizing saturation curve.

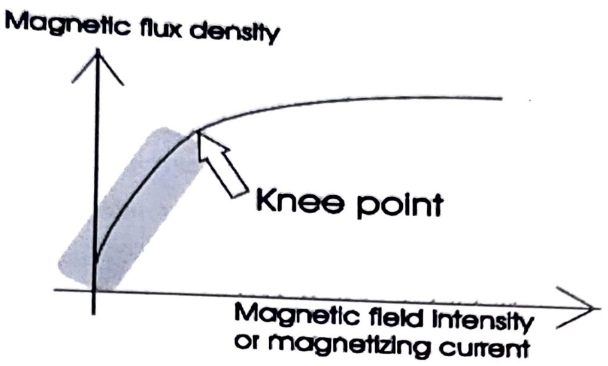



**Figure 4.6 knee point of magnetizing saturation curve**

From general torque equation of DC motor it is found that, torque $T \propto \varphi I_a$. Where, $\varphi$ is field flux and $I_a$ is armature current. But in field controlled DC servo motor, the armature is excited by constant current source , hence $I_a$ is constant here. Hence, $T \propto \varphi$

As field of this DC servo motor is excited by amplified error signal, the torque of the motor i.e. rotation of the motor can be controlled by amplified error signal. If the constant armature current is large enough then, every little change in field current causes corresponding change in torque on the motor shaft.

The direction of rotation can be changed by changing polarity of the field.

The direction of rotation can also be altered by using split field DC motor, where the field winding is divided into two parts, one half of the winding is wound in clockwise direction and other half in wound in anticlockwise direction. The amplified error signal is fed to the junction point of these two halves of the field as shown below. The magnetic field of both halves of the field winding opposes each other. During operation of the motor, magnetic field strength of one half dominates other depending upon the value of amplified error signal fed between these halves. Due to this, the DC servo motor rotates in a particular direction according to the amplified error signal voltage.

The main disadvantage of field control **DC servo motors**, is that the dynamic response to the error is slower because of longer time constant of inductive field circuit. The field is an electromagnet so it is basically a highly inductive circuit hence due to sudden change in error signal voltage, the current through the field will reach to its steady state value after certain period depending upon the time constant of the field circuit. That is why field control DC servo motor arrangement is mainly used in small servo motor applications.

The main advantage of using field control scheme is that, as the motor is controlled by field. The controlling power requirement is much lower than rated power of the motor.

### 4.3.2 Armature Controlled DC Servo Motor Theory:

Figure (4.5) shows the schematic diagram for an armature controlled DC servo motor. Here the armature is energized by amplified error signal and field is excited by a constant current source .

The field is operated at well beyond the knee point of magnetizing saturation curve. In this portion of the curve, for huge change in magnetizing current, there is very small change in mmf in the motor field. This makes the servo motor is less sensitive to change in field current. Actually for armature controlled DC servo motor, we do not want that, the motor should response to any change of field current.

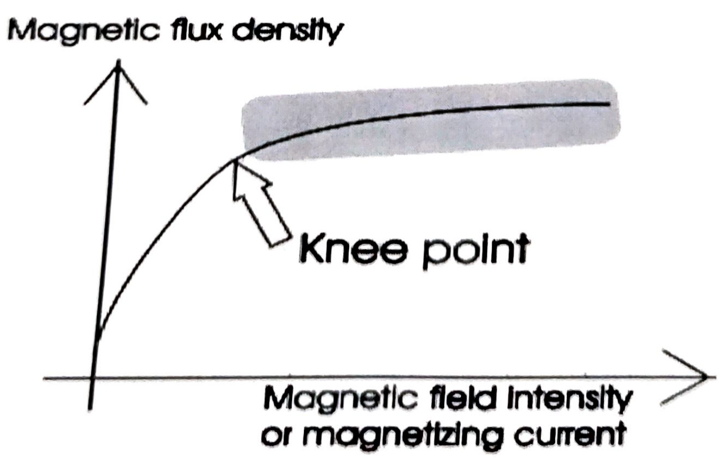


**Figure 4.7 knee point of magnetizing saturation**

Again, at saturation the field flux is maximum. As we said earlier, the general torque equation of DC motor is, torque $T \propto \varphi I_a$. Now if $\varphi$ is large enough, for every little change in armature current $I_a$ there will be a prominent changer in motor torque. That means servo motor becomes much sensitive to the armature current.

As the armature of DC motor is less inductive and more resistive, time constant of armature winding is small enough. This causes quick change of armature current due to sudden change in armature voltage.

That is why dynamic response of armature controlled DC servo motor is much faster than that of field controlled DC servo motor.

The direction of rotation of the motor can easily be changed by reversing the polarity of the error signal.

### 4.3.3 Permanent Magnet DC Servo Motor:

Field control is not possible in the case of permanent magnet DC motor as the field is a permanent magnet here. DC servo motor working principle in that case is similar to that of armature controlled motor.

## 4.4 Deriving State Equations for a DC Servo Motor

### 4.4.1System Model:

A useful component in many real control systems is a permanent magnet DC servo motor. The input signal to the motor is the armature voltage Va(t), and the output signal is the angular position θ(t). A schematic diagram for the motor is shown in Figure. (4.4.1.1). The terms Ra and La are the resistance and inductance of the armature winding in the motor, respectively. The voltage Vb is the back EMF generated internally in the motor by the angular rotation. J is the inertia of the motor and load (assumed lumped together), and B is the damping in the motor and load relative to the fixed chassis.

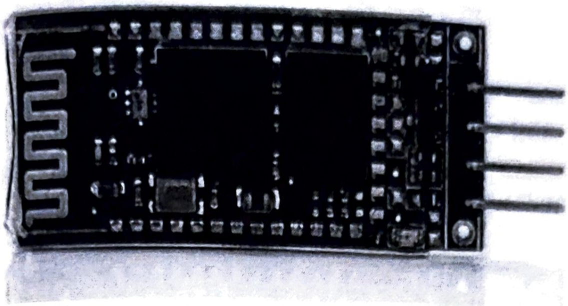


Figure 4.12 Bluetooth HC.06 module (Front View)

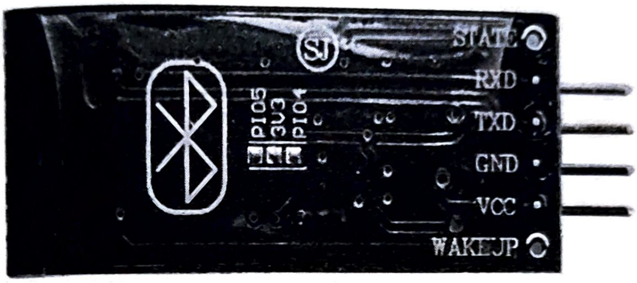



Figure 4.13 Bluetooth HC.06 module(Back View)

## 4.7 Specifications

### 4.7.1 Hardware features

- Typical .80dBm sensitivity
- Up to +4dBm RF transmit power
- Low Power 1.8V Operation ,1.8 to 3.6V I/O
- PIO control
- UART interface with programmable baud rate
- With integrated antenna
- With edge connector

### 4.7.2 Software features

- Default Baud rate: 38400, Data bits: 8, Stop bit: 1, Parity: No parity, Data

Control: has supported baud rate:

9600,19200,38400,57600,115200,230400,460800.

- Given a rising pulse in PIO0, device will be disconnected.
- Status instruction port PIO1: low disconnected, high connected;
- PIO10 and PIO11 can be connected to red and blue led separately. When master and slave are paired, red and blue led blinks 1time/2s in interval, while disconnected only blue led blinks 2times/s.
- Auto connect to the last device on power as default.
- Permit pairing device to connect as default.
- Auto pairing PINCODE:"0000" as default
- Auto reconnect in 30 min when disconnected as a result of beyond the range of connection.

# CHAPTER 5: MECHANICAL PORTION

## 5.1Mechanical Design

The mechanical design of the robot arm is based on a robot manipulator with similar functions to a human arm . The links of such a manipulator are connected by joints allowing rotational motion and the links of the manipulator is considered to form a kinematic chain. The business end of the kinematic chain of the manipulator is called the end effector or end of arm tooling and it is analogous to the human hand. Figure 5.1 shows the Free Body Diagram for mechanical design of the robotic arm.

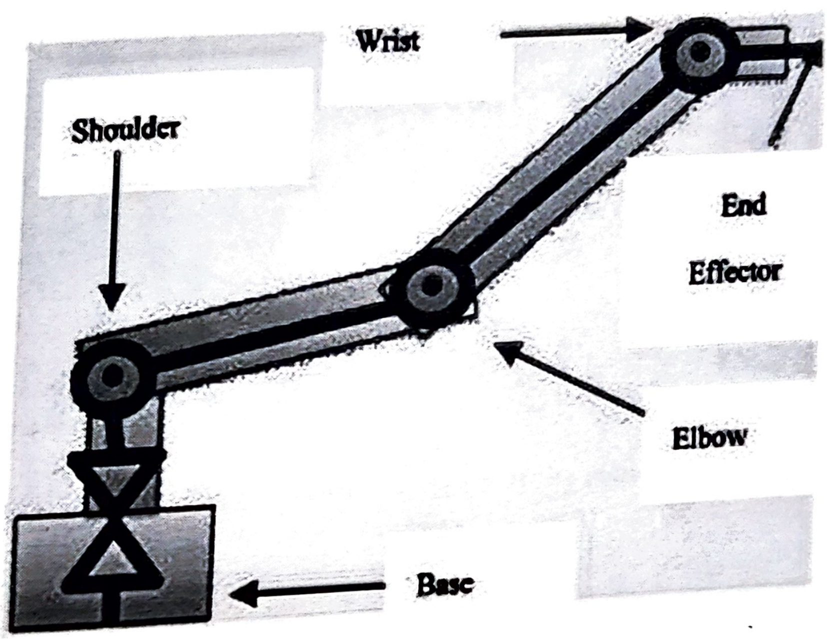
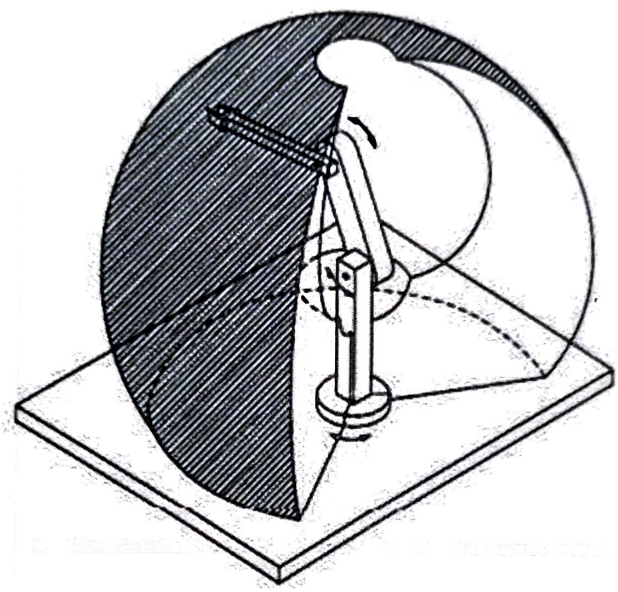


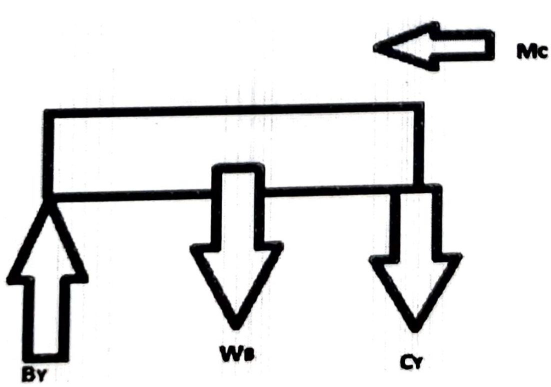

**Figure 5.1 Free body diagram of the robot arm**

As shown, the end effector is not included in the design because a commercially available gripper is used.This is because that the end effector is one of the most complex parts of the system and, in turn, it is much easier and economical to use a commercial one than build it. Figure 5.17 shows the work region of the robotic arm. This is the typical workspace of a robot arm with four degree of freedom (6 DOF). . The mechanical design was limited to 6 DOF mainly because that such a design allows most of the necessary movements and keeps the costs and the complexity of the robot competitively Accordingly, rotational motion of the joints is restricted where rotation is done around two axis in the shoulder and around only one in the elbow and the wrist, see Figure 5.18. The robot arm joints are typically actuated by electrical motors. The servo motors were chosen, since they include encoders which automatically provide feedback to the motors



and adjust the position accordingly.

**Figure 5.2 Work region of the robotic arm**



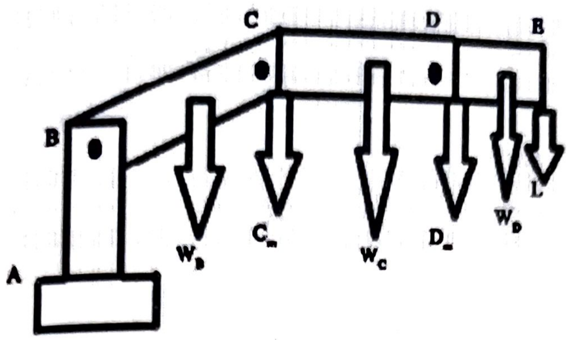



**Figure 5.3 Force diagram of robot arm**

**Figure 5.4 Force diagram of link CB**

However, the disadvantage of these motors is that rotation range is less than 180° span, which greatly decreases the region reached by the arm and the possible positions [9]. The qualifications of servo motors were selected based on the maximum torque required by the structure and possible loads. In the current study, the material used for the structure was acrylic.

Figure 2.16 shows the force diagram used for load calculations. The calculations were carried out only for the joints that have the largest loads, since the other joints would have the same motor, i.e. the motor can move the links without problems. The calculations considered the weight of the motors, about 50 grams, except for the weight of motor at joint B, since it is carried out by link BA. Figure 2.17 shows the force diagram on link CB, which contains the joints (B and C) with the highest load (carry the links DC and ED) and the calculations are carried out as follows.

The values used for the torque calculations:

Wd = 0.020 kg (weight of link DE)

Wc = 0.030 kg (weight of link CD)

Wb = 0.030 kg (weight of link CB)

L = 1 kg (load)

Cm = Dm = 0.050 kg (weight of motor)

LBC = 0.14 m (length of link BC)

LCD = 0.14 m (length of link CD)

LDE = 0.05 m (length of link DE)

Performing the sum of forces in the Y axis, using the loads as shown in Figure 5.20, and solving for CY and CB, see Equations (1).(4). Similarly, performing the sum of moments around point C, Equation (5), and point B, Equation (6), to obtain the torque e in C and B, Equations (7) and (8), respectively.

$$\sum F_y = (L + W_d + D_m + W_c + C_m)g - C_Y = 0 \quad (1)$$

$$C_Y = (1.141\,\mathrm{kg})\,9.8\,\mathrm{m/s^2} = 11.18\,\mathrm{N} \quad (2)$$

$$\sum F_y = (L + W_d + D_m + W_c + C_m + W_B)g - C_B = 0 \quad (3)$$

$$C_B = (1.171\,\mathrm{kg})\,9.8\,\mathrm{m/s^2} = 11.4758\,\mathrm{N} \quad (4)$$

$$\sum M_c = -\left(\frac{W_c L_{CD}}{2}\right) - W_D\left(L_{CD} + \frac{L_{DE}}{2}\right) - L(L_{CD} + L_{DE}) - D_m(L_{CD}) + M_c = 0 \quad (5)$$

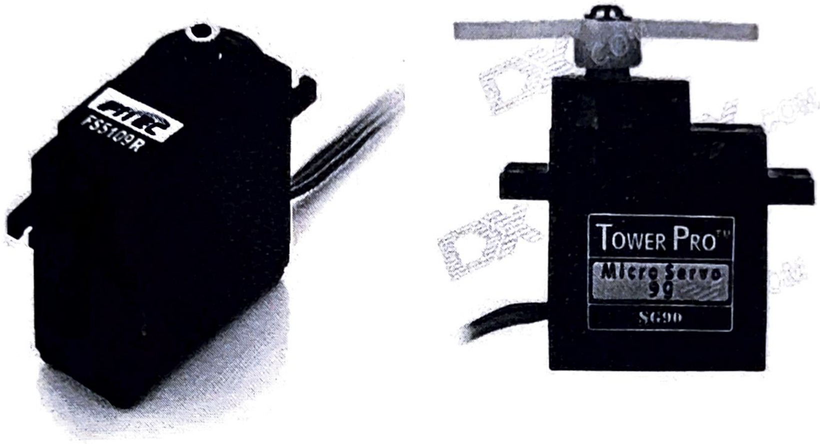
$$\sum M_B = -L(L_{BC} + L_{CD} + L_{DE}) - W_D\left(L_{BC} + L_{CD} + \frac{L_{DE}}{2}\right) - D_m(L_{BC} + L_{CD}) - W_c\left(L_{BC} + \frac{L_{CD}}{2}\right) - C_m(L_{BC}) - W_B\left(\frac{L_{BC}}{2}\right) + M_B = 0 \quad (6)$$

$$M_c = 1.968\,\mathrm{Nm} = 278.6\,\mathrm{oz/in} \quad (7)$$

$$M_B = 3.554\,\mathrm{Nm} = 503.38\,\mathrm{oz/in} \quad (8)$$

The servo motor that was selected, based on the calculations, is 7 small servo with torque: 1.8 kg/cm and one big servo with torque 10kg/cm

This motors was recommended because it is much cheaper than any other motor with same specifications. Since we need more torque at joint B, see Equation (8), we used two motors at point B to comply with the torque requirements; however, one motor is enough for the other joints. Using two motors at joint B is much cheaper than using one big motor with 10kg/cm. Other relevant characteristics of the motors, which can be shown in Figureure 5, are that they can turn 60 degrees in 130 milliseconds and they have a weight of 47.9 grams each. Once the initial dimensions for the robot arm and the motor




were defined, the design were carried out using the SolidWorks platform; design should

**Figure 5.5 Two Type of Servo Motor**

carefully take into account the thickness of the acrylic sheet and the way that the pieces would be attached to each other. The acrylic sheet used to make the robot is 1/8 thickness

And that thin sheet was chosen because it easier for machining and less weight with a good resistance. During design, we faced some difficulties due to the way of joining thin acrylic parts strongly. It was needed tools to burn and join the acrylic parts and that weren't available and the team considered that a mechanical junction based on screws and nuts would be much strong than other alternatives, such as glue for example. In order to accomplish this, a small feature was designed which allowed to fasten the bolts with the nuts without having to screw in the thin acrylic layer. The result of this process was the tridimensional design shown in Figure 6.21 By end of design, each part was printed in full scale in cardboard paper and then we verified all the dimensions and the interfaces of the assembly. In turn, we built the first prototype of the robot arm. Next, parts of the robot arm were machined from the acrylic sheet using a circular saw and Dermal tools. The detailing on the parts was done in a professional workshop since the parts of robot arm were too small and it is not an easy for accomplishing such small and accurate cuts. During assembling the robot parts with the motors, few problems pop up. There were critical points that did not resist the fastening and, in turn, may break down; hence, reinforcements in these points were considered. The final result of the robot arm is shown in Figure 5.21

## 5.2 Driving engines using complementary angle

We use this Operation to control tow servo motors In opposite direction for each other's. The small servo G9 we use have a rotation limit 180 degree so we command the first servo motor to move to angle X and command the opposite servo motor to move to angle 180.X then the tow motors well move in Synchronous movement that meets our needs.

## 5.3 Robot Arm Control

The robot arms can be autonomous or controlled manually. In manual mode, a trained operator (programmer) typically uses a portable control device (a teach pendant) to teach a robot to do its task manually. Robot speeds during these programming sessions are slow. In the current work we enclosed the both modes. The control for the presented robot arm consists basically of three levels: a microcontroller, a driver, and a computer based user interface. This system has unique characteristics that allow flexibility in programming and controlling method, which was implemented using inverse kinematics; besides it could also be implemented in a full manual mode. The electronic design of control is shown in Figure 11. The microcontroller used is an Atmega 368p which comes with a development/programming board named "Arduino", as shown in Figure 12. The programming language is very similar to C but includes several libraries that help in the control of the I/O ports, timers, and serial communication. This microcontroller was chosen because it has a low price, it is very easy to reprogram, the programming language is simple, and interrupts are available for this particular chip. The driver used is a eight .channel for servo controller board. It supports tow control methods: Bluetooth for direct connection to an android device or direct control using variable resistors. This controller, as shown in Figure 5.6.

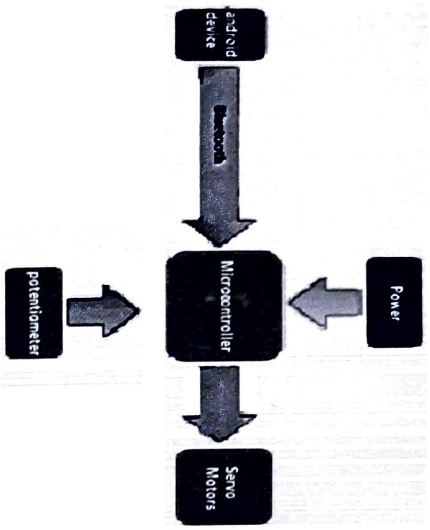
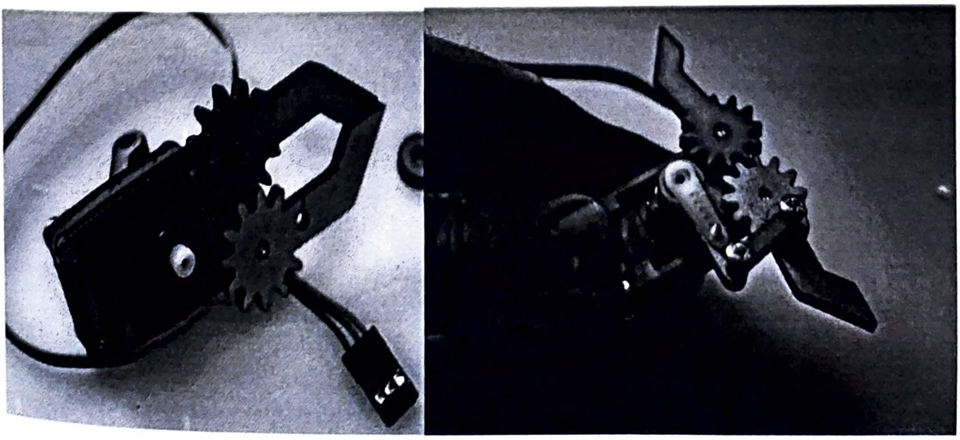


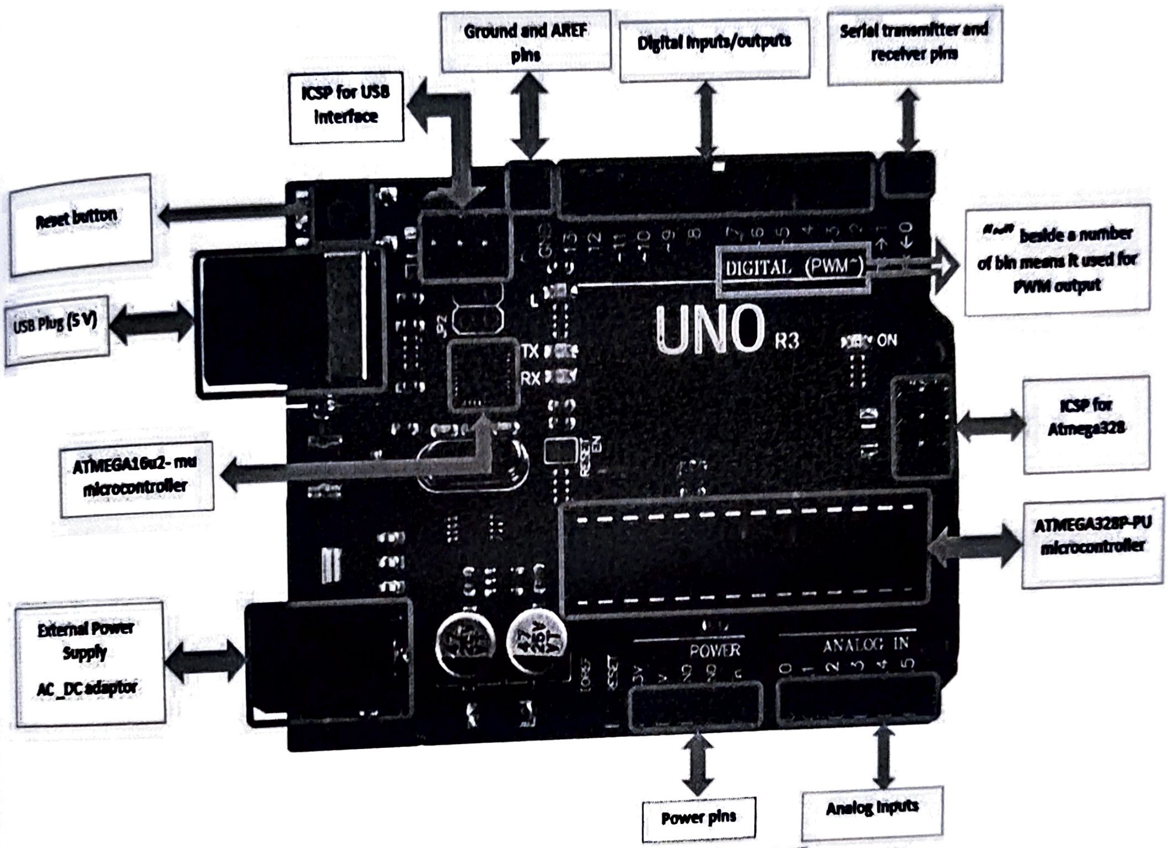

**Figure 5.6 Electronic scheme of control**

## 5.4 End-Effector Selection

The end effector is probably one of the most important and most complex parts of the system. The end effector varies mainly according to the application and the task that



# Arduino Uno R3




## INTRODUCTION

Arduino is used for building different types of electronic circuits easily using of both a physical programmable circuit board usually microcontroller and piece of code running on computer with USB connection between the computer and Arduino.

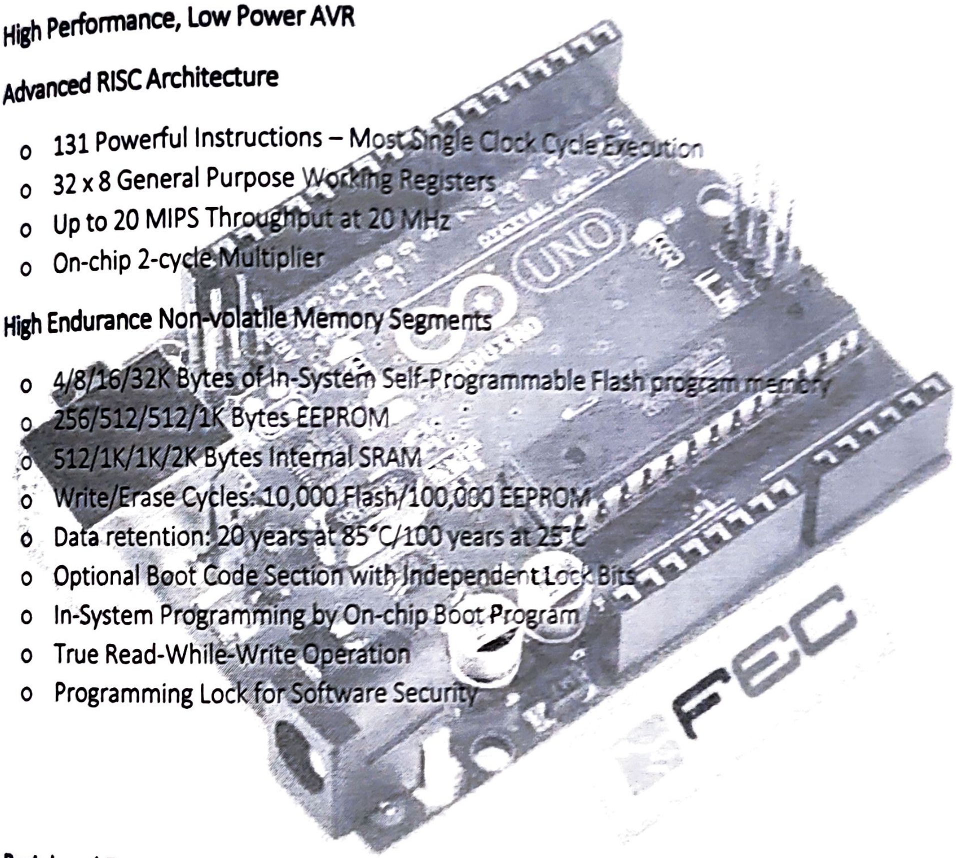
Programming language used in Arduino is just a simplified version of C++ that can easily replace thousands of wires with words.

## ARDUINO UNO-R3 PHYSICAL COMPONENTS

### ATMEGA328P-PU microcontroller

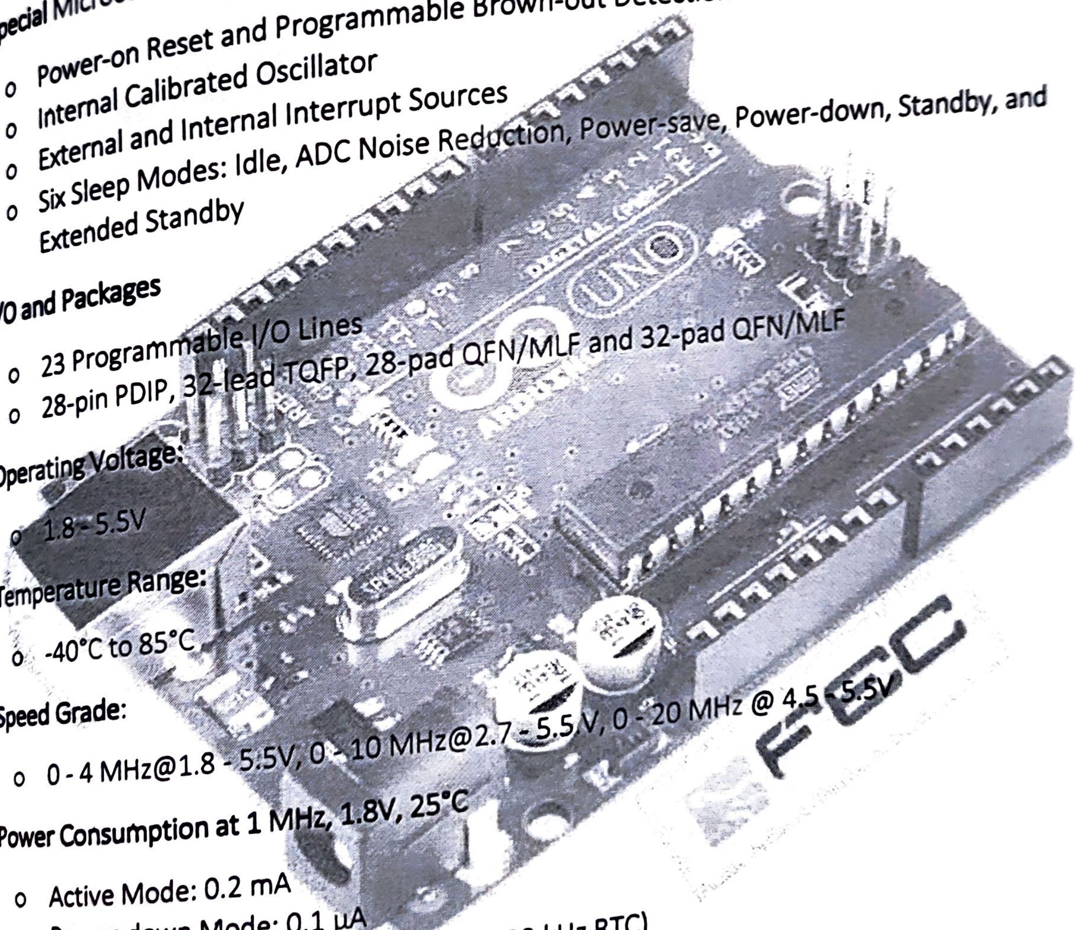
The most important element in Arduino Uno R3 is ATMEGA328P-PU is an 8-bit Microcontroller with flash memory reach to 32k bytes. It's features as follow:

- High Performance, Low Power AVR
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash program memory
  - 256/512/512/1K Bytes EEPROM
  - 512/1K/1K/2K Bytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C
  - Optional Boot Code Section with Independent Lock Bits
  - In-System Programming by On-chip Boot Program
  - True Read-While-Write Operation
  - Programming Lock for Software Security




- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
  - Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
  - Temperature Measurement
  - Programmable Serial USART

- Master/Slave SPI Serial Interface
- Byte-oriented 2-wire Serial Interface (Philips I2 C compatible)
- Programmable Watchdog Timer with Separate On-chip Oscillator
- On-chip Analog Comparator
- Interrupt and Wake-up on Pin Change




**• Special Microcontroller Features**

- Power-on Reset and Programmable Brown-out Detection
- Internal Calibrated Oscillator
- External and Internal Interrupt Sources
- Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby

**• I/O and Packages**

- 23 Programmable I/O Lines
- 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF

**• Operating Voltage:**

- 1.8 - 5.5V

**• Temperature Range:**

- -40°C to 85°C

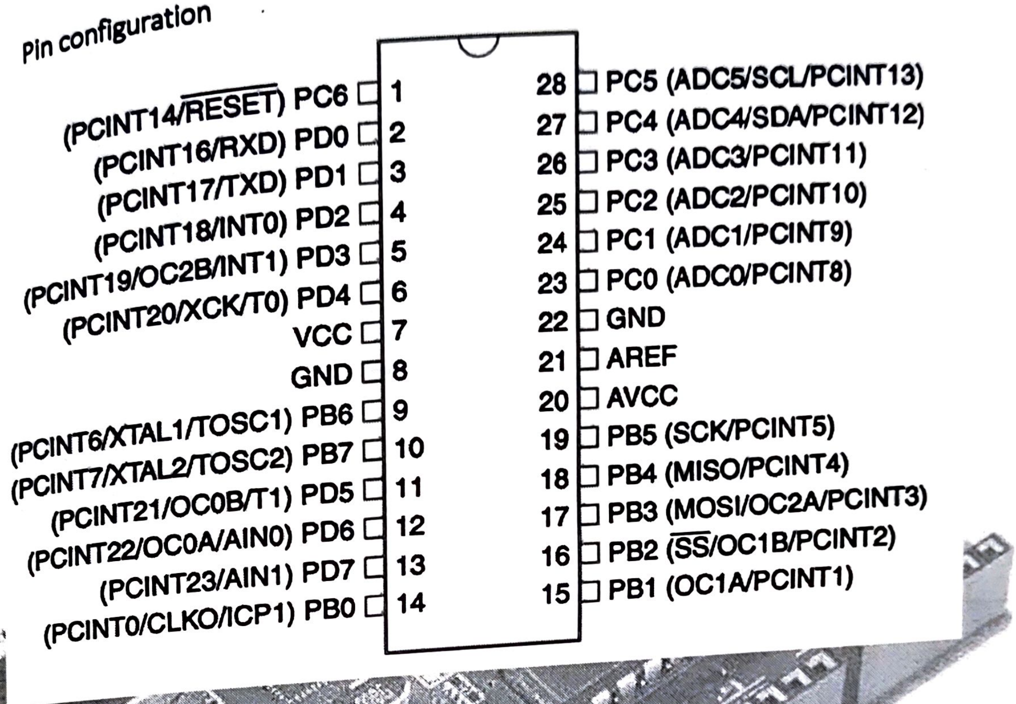
**• Speed Grade:**

- 0 - 4 MHz@1.8 - 5.5V, 0 - 10 MHz@2.7 - 5.5 V, 0 - 20 MHz @ 4.5 - 5.5V

**• Power Consumption at 1 MHz, 1.8V, 25°C**

- Active Mode: 0.2 mA
- Power-down Mode: 0.1 µA
- Power-save Mode: 0.75 µA (Including 32 kHz RTC)

- Pin configuration




ATMEGA16u2- mu microcontroller

Is a 8-bit microcontroller used as USB driver in Arduino uno R3 it's features as follow:

- High Performance, Low Power AVR

• Advanced RISC Architecture

  - 125 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz

• Non-volatile Program and Data Memories

  - 8K/16K/32K Bytes of In-System Self-Programmable Flash
  - 512/512/1024 EEPROM
  - 512/512/1024 Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/ 100,000 EEPROM
  - Data retention: 20 years at 85°C/ 100 years at 25°C

- Optional Boot Code Section with Independent Lock Bits
- In-System Programming by on-chip Boot Program hardware-activated after reset
- Programming Lock for Software Security

• **USB 2.0 Full-speed Device Module with Interrupt on Transfer Completion**

- Complies fully with Universal Serial Bus Specification REV 2.0
- 48 MHz PLL for Full-speed Bus Operation: data transfer rates at 12 Mbit/s
- Fully independent 176 bytes USB DPRAM for endpoint memory allocation
- Endpoint 0 for Control Transfers: from 8 up to 64-bytes
- 4 Programmable Endpoints:
  - IN or Out Directions
  - Bulk, Interrupt and Isochronous Transfers
  - Programmable maximum packet size from 8 to 64 bytes
  - Programmable single or double buffer
- Suspend/Resume Interrupts
- Microcontroller reset on USB Bus Reset without detach
- USB Bus Disconnection on Microcontroller Request

• **Peripheral Features**

- One 8-bit Timer/Counters with Separate Prescaler and Compare Mode (two 8-bit PWM channels)
- One 16-bit Timer/Counter with Separate Prescaler, Compare and Capture Mode(three 8-bit PWM channels)
- USART with SPI master only mode and hardware flow control (RTS/CTS)
- Master/Slave SPI Serial Interface
- Programmable Watchdog Timer with Separate On-chip Oscillator
- On-chip Analog Comparator
- Interrupt and Wake-up on Pin Change

• **On Chip Debug Interface (debug WIRE)**

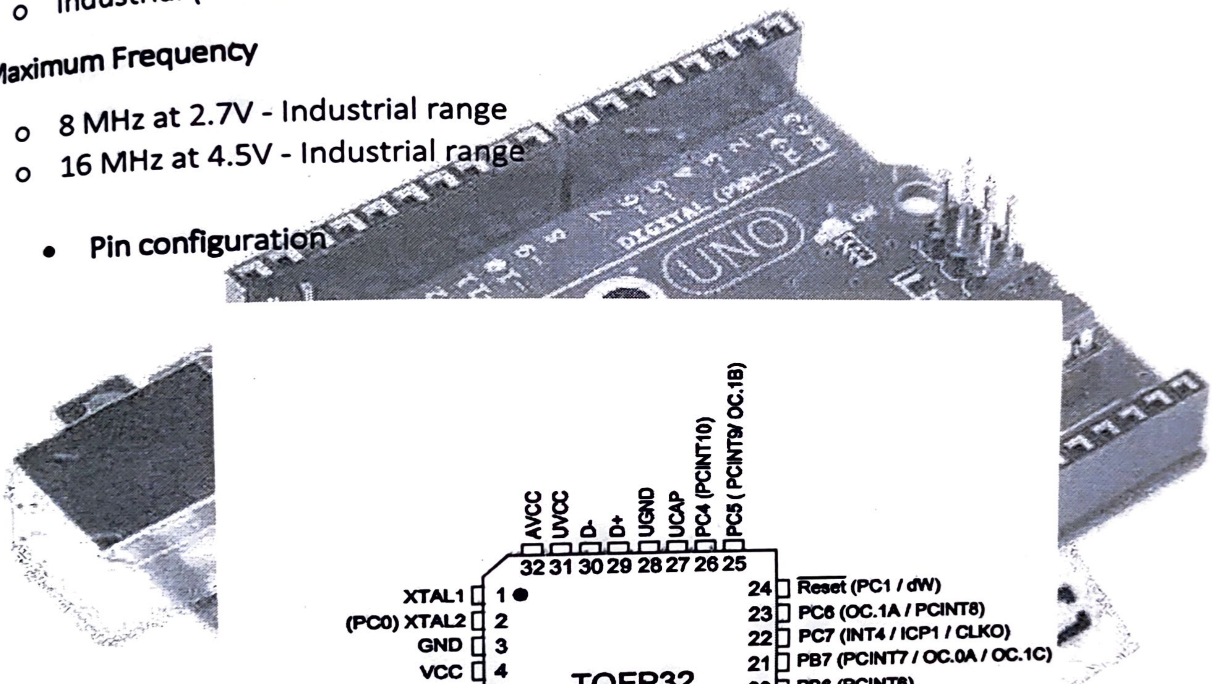
• **Special Microcontroller Features**

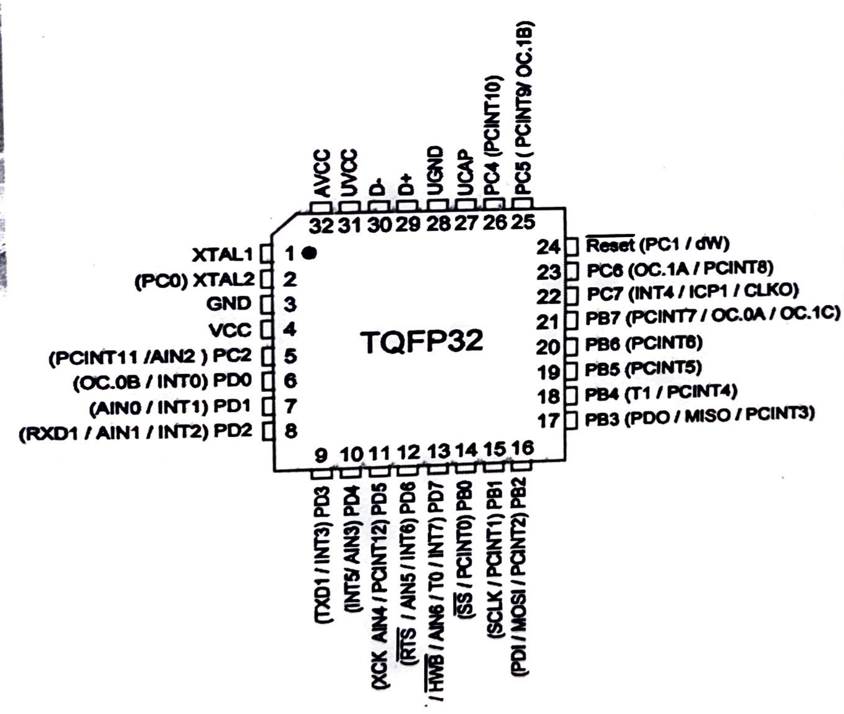
- Power-On Reset and Programmable Brown-out Detection
- Internal Calibrated Oscillator
- External and Internal Interrupt Sources
- Five Sleep Modes: Idle, Power-save, Power-down, Standby, and Extended Standby

• **I/O and Packages**

- 22 Programmable I/O Lines
- QFN32 (5x5mm) / TQFP32 packages

- Operating Voltages
  - 2.7 - 5.5V
- Operating temperature
  - Industrial (-40°C to +85°C)
- Maximum Frequency
  - 8 MHz at 2.7V - Industrial range
  - 16 MHz at 4.5V - Industrial range
- Pin configuration



## OTHER ARDUINO UNO R3 PARTS

### Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode (), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 k Ohms. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

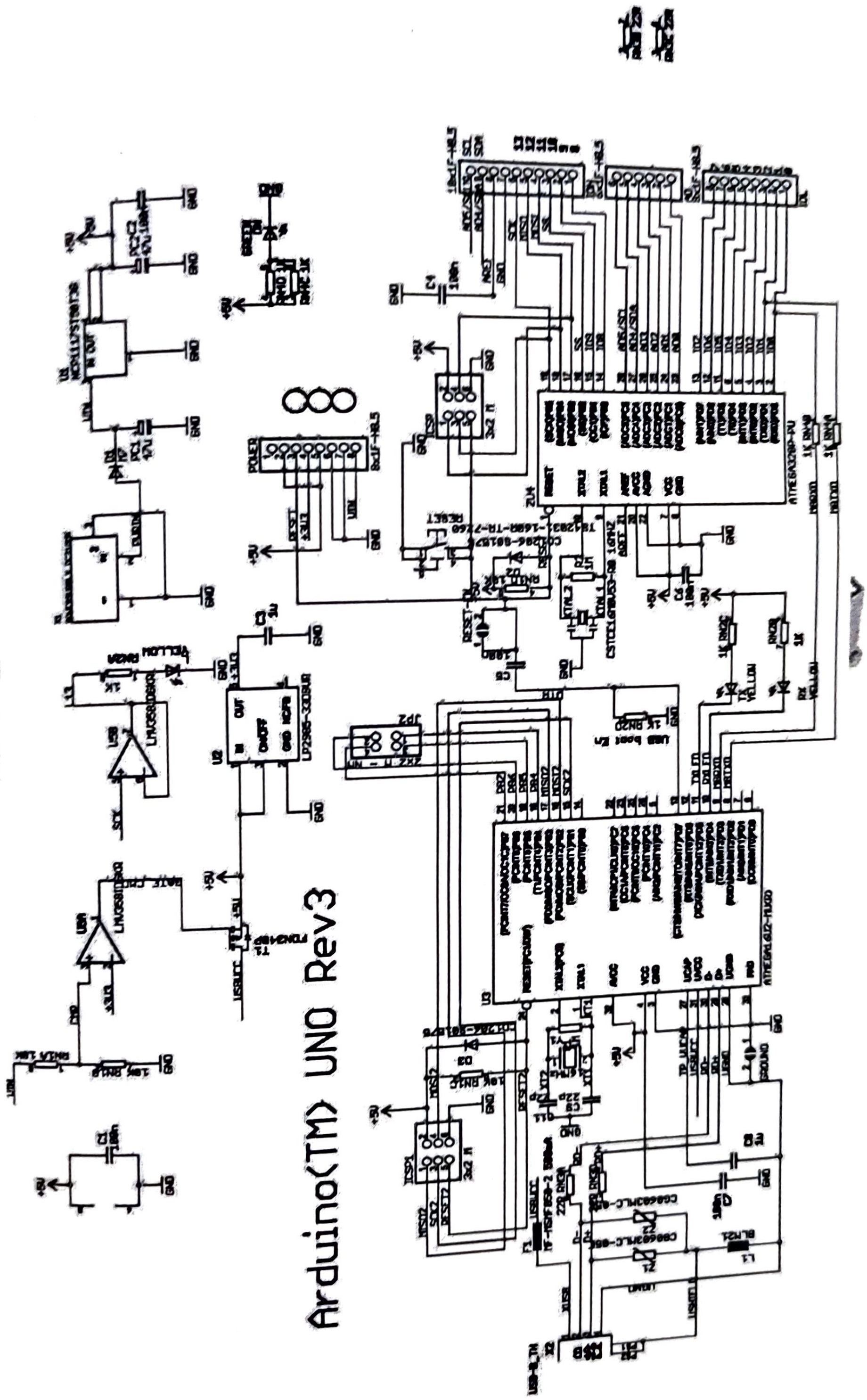
The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e.1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.
Additionally, some pins have specialized functionality:

- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

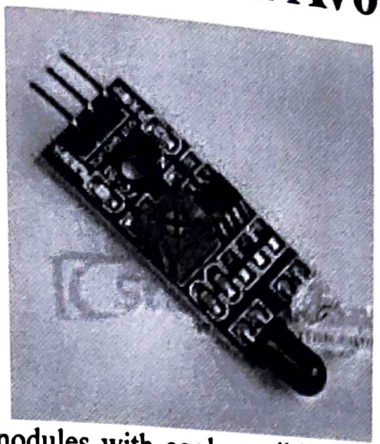
There are a couple of other pins on the board:

- AREF: Reference voltage for the analog inputs. Used with analogReference().
- Reset: Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

# ARDUINO UNO R3 SCHEMATIC DIAGRAM



# Arduino Infrared Collision Avoidance

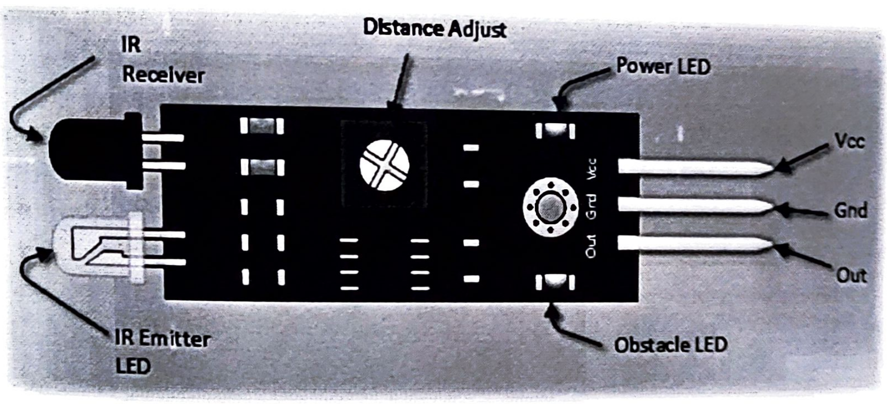


This is yet another one of those modules with cool possibilities. You could for example, sound an alarm when something got too close or you could change the direction of a robot or vehicle.

The device consists of an Infrared Transmitter, an Infrared Detector, and support circuitry. It only requires three connections. When it detects an obstacle within range it will send an output low.

# IR Obstacle Detection Module Pin Outs

The drawing and table below identify the function of module pin outs, controls and indicators.

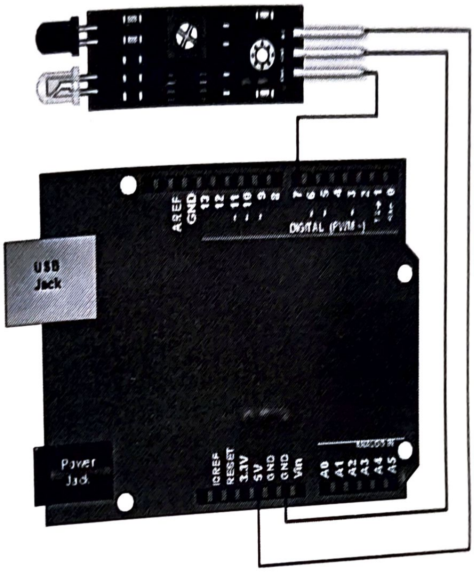



| Pin, Control Indicator | Description |
|---|---|
| Vcc | 3.3 to 5 Vdc Supply Input |
| Gnd | Ground Input |
| Out | Output that goes low when obstacle is in range |
| Power LED | Illuminates when power is applied |
| Obstacle LED | Illuminates when obstacle is detected |
| Distance Adjust | Adjust detection distance. CCW decreases distance. CW increases distance. |
| IR Emitter | Infrared emitter LED |
| IR Receiver | Infrared receiver that receives signal transmitted by Infrared emitter. |

# Arduino IR Obstacle Collision Module Tutorial

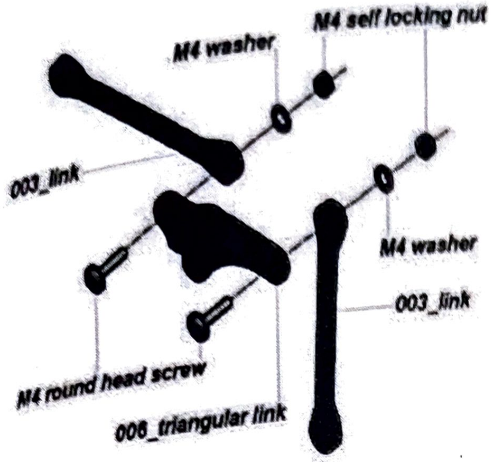
## Connect the Arduino to the Detection Module

Use the picture below. It only requires three wires.



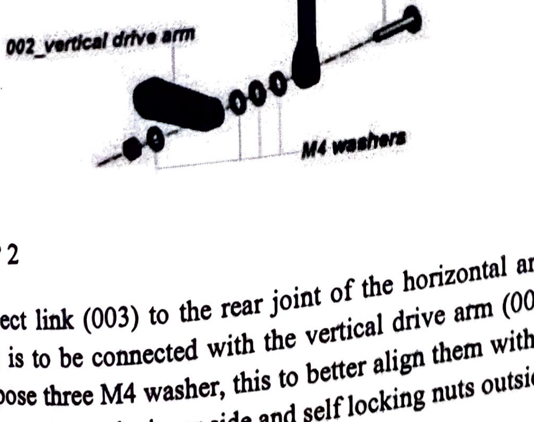

## Copy, Paste and Upload the Sample Sketch

```
// IR Obstacle Collision Detection Module
// Henry's Bench

int LED = 13; // Use the onboard Uno LED
int isObstaclePin = 7;  // This is our input pin
int isObstacle = HIGH;  // HIGH MEANS NO OBSTACLE

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(isObstaclePin, INPUT);
  Serial.begin(9600);

}

void loop() {
  isObstacle = digitalRead(isObstaclePin);
  if (isObstacle == LOW)
  {
    Serial.println("OBSTACLE!!, OBSTACLE!!");
    digitalWrite(LED, HIGH);
  }
  else
  {
    Serial.println("clear");
    digitalWrite(LED, LOW);
  }
  delay(200);
}
```



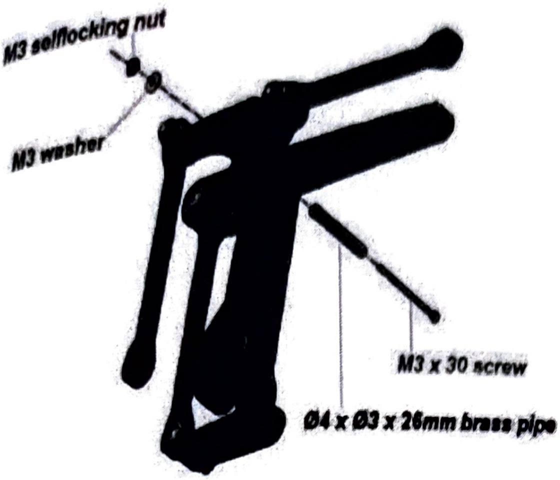

STEP 1

Connect two link arms (003) to the Triangular link (006). Keep the M4 round heads screws to the inner side like shown on image and selflocking nuts to the outer side.



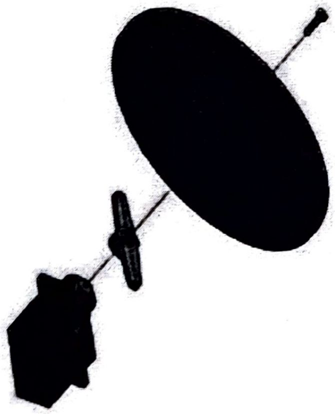

STEP 2

Connect link (003) to the rear joint of the horizontal arm (005). The lower part of the link (003) is to be connected with the vertical drive arm (002) as shown. Between the two links interpose three M4 washer, this to better align them with the vertical arm Keep the M4 round heads screws to the inner side and self locking nuts outside.



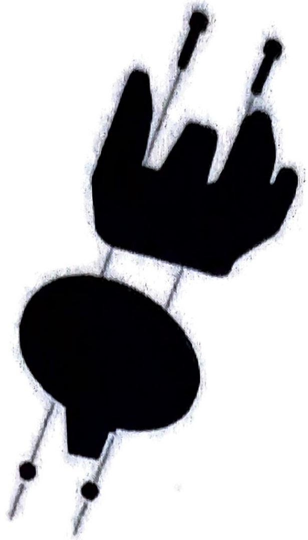

STEP 3

Connect now the two preassembled links to the forward drive arm (004). Punt in position horizontal arm (005) and triangular link (006) aligned with the upper connection of the forward drive arm (004). Insert the dia 4 mm brass pipe crossing all the parts and fix it with the M3x30 screw, locked by the nut on the other side. Verify the freedom of movement and If everything is ok, proceed to the next step. .



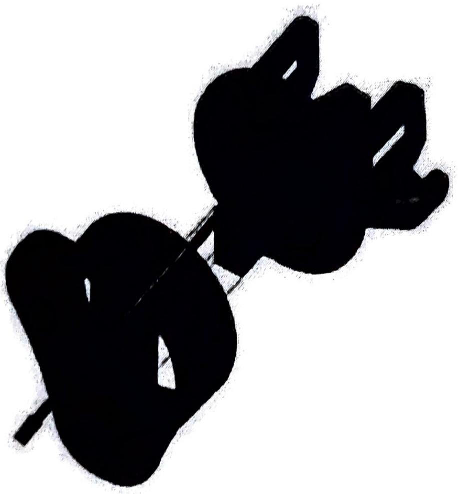
STEP 5

Be sure that the servo is in the neutral position than install the double arm horn on the splined shaft keeping the arms parallel to the servo body Insert the horn inside the housing below the round plate and fix the servo to the plate using one of the two long screw supplied with the servo (the small one in too short due to the thickness of round plate) .



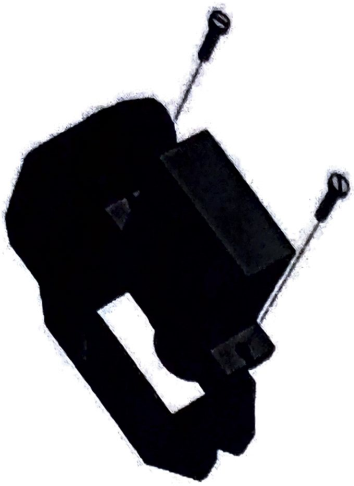
STEP 6

Put in position the base between the two shoulder on the plate and attach together using the two M3 screws and nuts. There two hexagonal housing below, so nuts will be kept in position during tightening



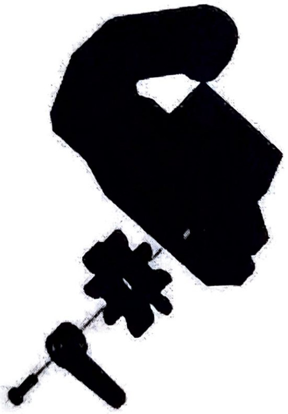
STEP 7

Align the servo and introduce the wiring in the central part of the basement. Gently pull the wire to make it straight while continue to push in it housing the servo The wire is then kept in position making it pass through a frontal hole .



STEP 9

Attach the servo to the claw support using the two self tapping screws supplied with the servo. Keep the output shaft forward.



STEP 10

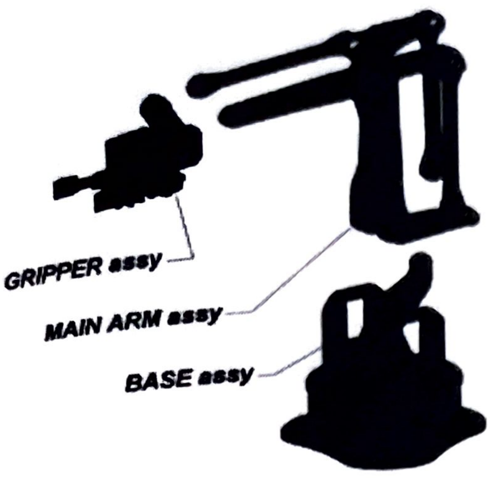
Insert the horn in the driving gear, then attach the horn at the splined servo shaft using the supplied screw The horn has to be aligned forward with the servo in neutral position. Cut the exceeding part of the horn from gear using a cutter.

STEP 11

Insert an M3 screw in the central hole connect it to the claw support then tight the self locking nut checking the freedom of movement.

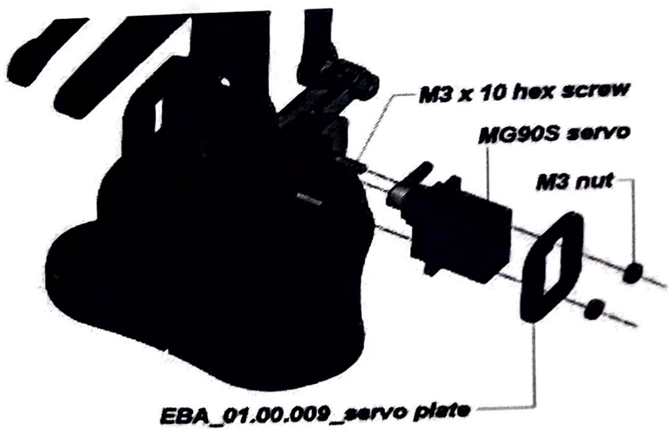
STEP 12

Insert the two pin of the driven gear into the dedicated holes on the left finger The driven gear has also a shoulder that has to be aligned with the lateral side of the finger. If you find difficulties coupling them, reduce interference using a file. Once coupled insert an M3 screw in the central hole and attach the finger to the claw support Now the gripper is ready to be installed on the horizontal arm of the EEzybot Verify freedom of movement of the gripper manually or using a servo tester Video on YouTube. .



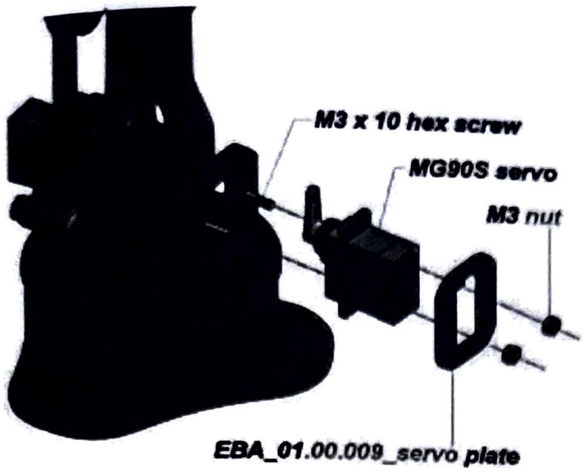

STEP 14

Now we have the three main sub assembly ready to be connected each other. Next step we will join the base with the main arms
To join the base with the main arms align the axis of the parts and insert from one side the brass pipe 24mm long. Also the short arm of the servo that drive the vertical movement has to be supported by the brass pipe as shown on the pictures. Check the freedom of movement .



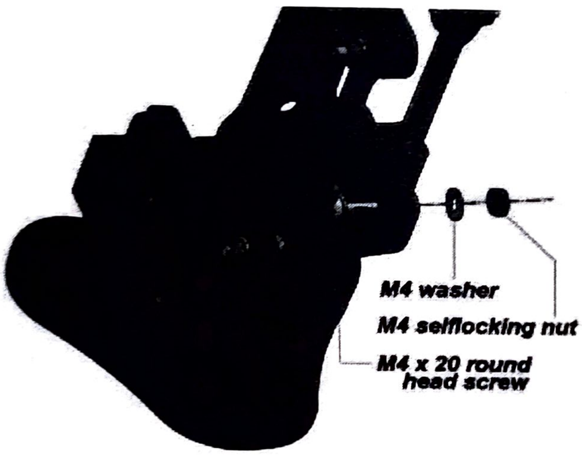

STEP 15

Is time now to install the servo that drive the vertical movement of the arm. Put in the dedicate receptacles two M3x10 hex screw. The servo has to be in the neutral position with the horn at 90 degrees on the right side with the press plate (009) installed (Make the wiring pass through the dedicated enlargment). Introduce the servo angled in the square seat on the base plate and slide the horn in the shaped housing of the arm that drives the vertical movement. Fixt the press plate against the servo using two M3 nuts .



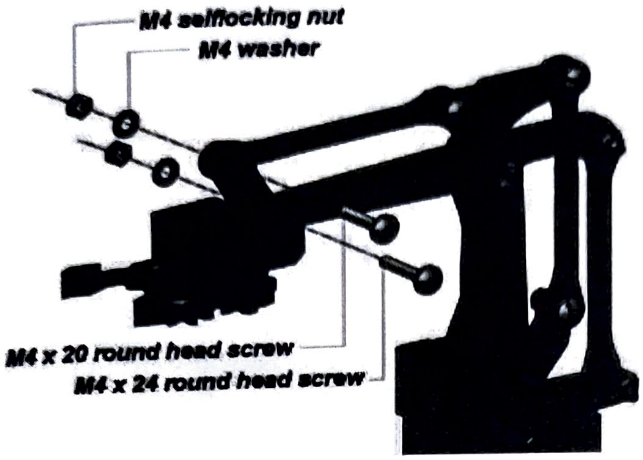

STEP 16

Sequence for the forward&backward driving servo is similar to the previous. In this case the servo horn has to be installed with the servo in neutral condition aligned vertically..



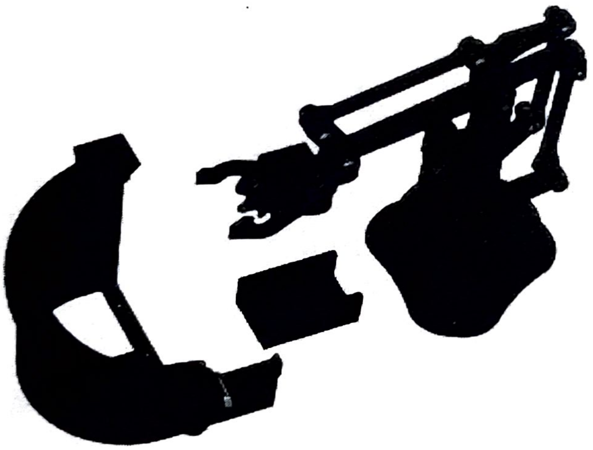

STEP 17

attach the latest link to the fixed arm on the rear side of the base using a M4x20 a washer and a selflocking nut




STEP 18

The last assembly step is to join the gripper to the horizontal arm as shown on the picture..



STEP 19

At the end of the last step the ARM is ready to work. As an optional, In the 3D model downodable from Thinghiverse, I add a round ramp that allow to easy obtain a loop test with a ball (3D printed, obviously!). Video on YouTube. To make this tool to work you have to attach another servo (cheap SG90) to the end of the ramp. I keep th ramp center at a distance of about 180mm from the base vertical axis. There is also a 3D model of a support dedicated to the Pololu USB servo.

# THANK YOU……..