**CHAPTER 1**

**INTRODUCTION**

**CHAPTER 1**

**INTRODUCTION**

* 1. **Robotic Arm**

A robotic arm is a type of mechanical arm, usually programmable, with similar functions to a human arm; the arm may be the sum total of the mechanism or may be part of a more complex robot.

Robotic Arm makes use of Actuators, Hydraulics, Servos or other related mechanisms to control the movements. These movements can be either pre-programmed or user-controlled depending on the needs put forth by the environment. Robotic arm is immensely used for industrial purposes as it allows precision-controlled movements.

The term “robotic arm” broadly describes a group of robotic mechanisms. These different robot types tend to have similar applications. However, each type has distinctions that typically lead to it being optimized for certain tasks over other robotic arms. Types of robotic arms include:

**1.1.1 Articulated arm**

The Articulated arm robots are general-purpose robotic arms with 4 or more joints or degrees of freedom. Fig 1.1 represents an articulated robotic arm. The articulated arm is an umbrella term for many other robot types. For instance, a six-axis robot is an articulated arm with six degrees of freedom. Articulated arms cover the broadest range of robot types used in industry and include six-axis and collaborative robots.



Fig 1.1 Articulated robotic arm

**1.1.2 Six-axis**

The Six-axis robots are the most common articulated arm. This also makes them the most common robotic arm used in industry today. Fig 1.2 shows a collaborative six axis robotic arm along with their various axes of rotation. Thanks to their flexibility, they are a great general-purpose robotic arm. This gives the six-axis an impressive list of uses. The six-axis robot is the most easily identified industrial robot.



Fig. 1.2 Collaborative six-axis robotic arm

**1.1.3 Collaborative robot**

Collaborative robot is a robotic arm purpose-built for hybrid work. This means it is designed to work near humans. Certain safety features allow for significant risk reduction in hybrid work environments. This is a relatively new robot type and its uses are still being explored. Collaborative robots are becoming more prominent in the industry as more manufacturers are being exposed to their benefits. The future is exciting for collaborative.

**1.1.4 SCARA**

SCARA robots are Selectively Compliant Robot Arms. This means they don’t have the same flexibility afforded to articulated arms. This limits them in some respects but gives them certain advantages over articulated arm types. Fig 1.3 shows a SCARA robot that can allow movements along x-y direction but cannot allow movement in z-direction.



Fig 1.3 Selectively Compliant robotic arm

**1.1.5 Cartesian** **Robotic Arm**

Cartesian robots are rigid systems that move around in a 3D coordinate plane. These robots are typically constructed of 3 linear actuators. One actuator moves left and right in the x-axis. An additional actuator is attached to the x-axis actuator. This actuator moves up and down in the y-axis plane. A final actuator is attached to the y-axis member and moves back and forth in the z-axis plane. Cartesian robots are positioned for small applications. It can be observed from Fig 1.4 that the gripper is attached to a y-axis linear actuator that can move up and down. This linear actuator is connected to another linear actuator which can allow movements in left and right direction along z-axis.

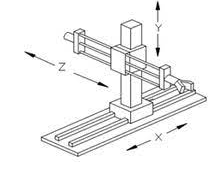
****

Fig. 1.4 Cartesian robotic arm

**1.1.6 Cylindrical Robotic Arm**

Cylindrical robotic arms are designed around a single arm that moves up and down a vertical member. This vertical member rotates the arm horizontally. The arm can extend and retract to perform its task. These robots are very compact and are deployed for small and simple tasks. Fig 1.5 represents a cylindrical robotic arm.

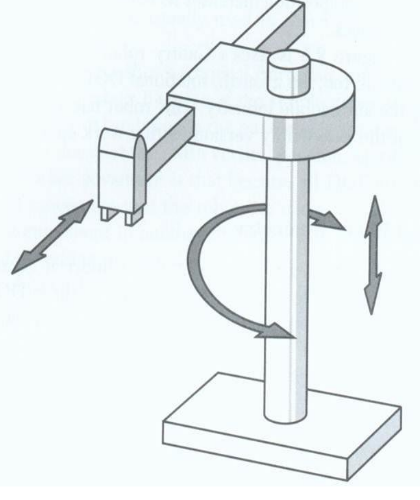
****

Fig. 1.5 Cylindrical robotic arm

**1.1.7 Spherical/Polar Robotic Arm**

The first modern industrial robot was a spherical (polar) robot. This robot type has a simple design that isn’t as common today as it once was. Spherical robots are similar to cylindrical robots except they swap the vertical linear axis with an additional rotary axis. This axis allows it to rotate vertically. It was designed for simple tasks that don’t require high speed or complex motion. The robotic arm as shown in Fig1.6 has a larger working envelope when compared to cylindrical robotic arm



Fig. 1.6 Spherical robotic arm

**1.1.8 Parallel/Delta** **Robotic Arm**

Parallel/Delta robots are high-speed options for robotic automation. These robots’ unique design allows them to reach incredible rates of speed. The delta robot is a great choice for high-speed and lightweight tasks. Parallel/Delta robotic Arm as shown in Fig 1.7 is generally used on assembly lines where products run in a random order. Example: Removing inconsumable potatoes from line in a fast food facility.



Fig. 1.7 Parallel/Delta robotic arm

**1.2 Arduino**

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low-cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step by step instructions of a kit, or sharing ideas online with other members of the Arduino community.

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems.

**1.3 Advantages of Arduino**

1. **Inexpensive** - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50
2. **Cross-platform** - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
3. **Simple, clear programming environment** - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.
4. **Open source and extensible software** - The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
5. **Open source and extensible hardware** - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the [breadboard version of the module](https://www.arduino.cc/en/Main/Standalone) in order to understand how it works and save money.

**CHAPTER 2**

**LITERATURE SURVEY**

## CHAPTER 2

**LITERATURE SURVEY**

1. 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. HC-05 Bluetooth module was used by Adarsh V Patil, Sreevarsh Prakash, Akshay S., Mahadevaswamy, Chandanb Patgar and Sharath Kumar A.J for “Android based smart door locking system” [1], the Bluetooth module was used for a smart door lock system using an android platform which made it cost effective and user friendly.
2. A detailed survey of robotic arm by Virendra Patidar and Ritu Tiwari “Survey of Robotic Arm and Parameters” [2] was useful in understanding the functioning and control of robotic arm. We were able to comprehend what factors affect the performance of robotic arm and how multiple axes is used to change the mass of an arm, Degrees of Freedom(DOF) increased by simple adding joints, working envelope and space to decide according to the situation, kinematics improved movement of the robot. Also the presence of diagrams helped us in understanding of robotic arm. The discussed gaps in research and issues was used as guideline for implementation of our project.
3. A pick-and-place robotic arm vehicle was developed by Kumar Aaditya, Divesh Kumar Pande and Preksha Moondra “Android controlled pick and place robotic arm vehicle” [3] which provided the necessary information for controlling the robotic arm using android app. We concluded that Android has a growing selection of third-party applications, which can be acquired by users by downloading and installing the application's APK (Android application package) file, or by downloading them using an application store program that allows users to install, update, and remove applications from their devices. Google Play Store is the primary application store installed on Android devices that comply with Google's compatibility requirements and license the Google Mobile Services software. Usage of high torque motor in the circuits helps the arm to lift heavier loads.
4. A robotic arm was implemented using Arduino by Ramkrishna Ghogre, Harshad Mehta, Ankit Khobragade, Ashutosh Tripathi, “Arduino Based Android Controlled Robotic Arm”, 2020, International Research Journal of Engineering and Technology (IRJET) [4] was extremely helpful and comprehensive and enabled us to understand better on pre-requisites to design the project. We reduced that the main problem to overcome in this project was to interface the Arduino Uno board with the android device via Bluetooth module. We extensively referred this paper during the initial stages of implementation of our project.
5. Arduino.cc provides us extensive library and documentation support. We can import these libraries and make the code simpler and efficient [5]. Arduino.cc provides board manuals which included details regarding voltage specifications, pin-out diagrams, analog and digital port configuration for Pulse Width Modulation (PWM) and processor architecture. We were also provided with information regarding communication interfaces between Arduino Board and host computer (Computer running Arduino IDE). We henceforth selected USB Type A/B connectors as they were easy to operate and easily available.
6. Detailed characteristics of HC-05 modules was provided by Components101 hosted on components101.com [6]. We were provided with specifications of the module including pin configuration, operating modes, voltage and current ratings. We noted that it supports serial communication at baud rates of 9600, 19200, 38400, 57600, 115200, 230400 and 460800 which is ideal for USB interface supported by the host computer.

**CHAPTER 3**

**ROBOTIC ARM USING ARDUINO**

## CHAPTER 3

**ROBOTIC ARM USING ARDUINO**

**3.1 TABLE OF COMPONENTS FOR IMPLEMENTATION OF ROROBOTIC ARM USING ARDUINO**

|  |  |  |
| --- | --- | --- |
| Sl.no | Components Used | Quantity |
| 1. | Arduino Uno | 1 |
| 2. | SG90 Servo Motor | 4 |
| 3. | HC-05 Bluetooth Module | 1 |
| 4. | Robotic Arm | 1 |
| 5. | Connecting Wires | - |
| 6. | 12-volt Battery | 1 |
| 7. | Android Device with custom application installed | 1 |

Table 1.1. Components Required

**3.2 BLOCK DIAGRAM**

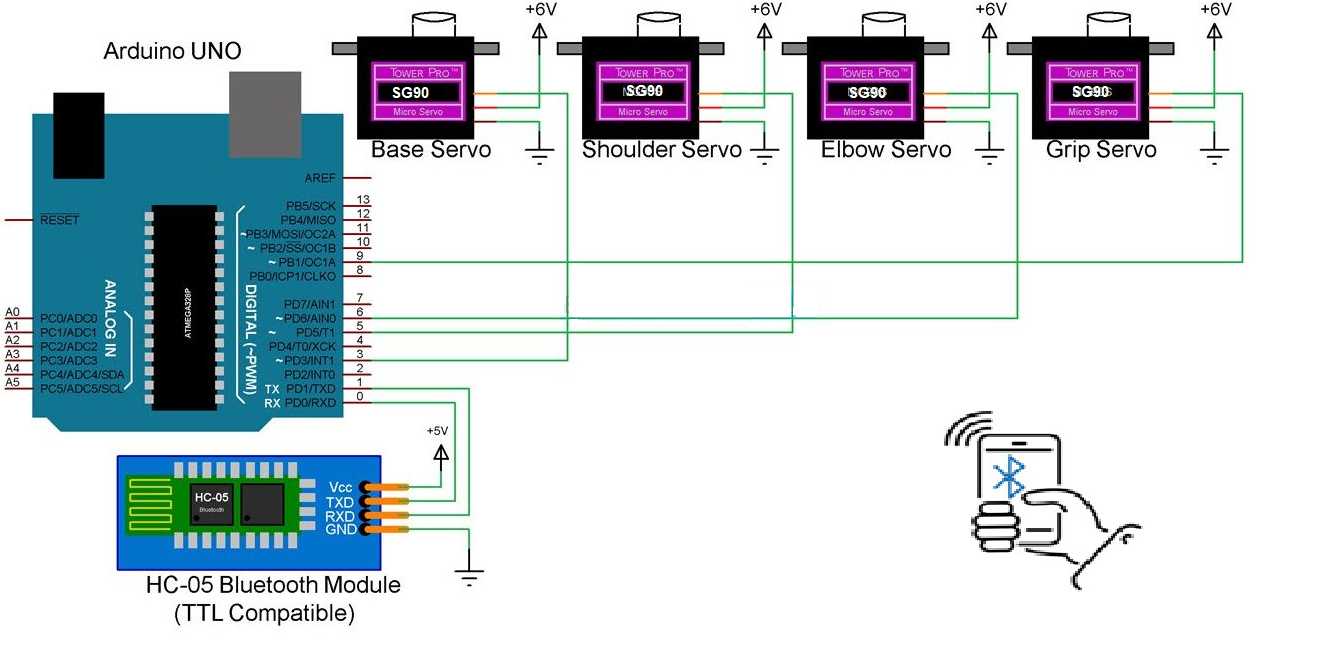
****

Fig. 1.8. Block Diagram for Robotic Arm using Arduino and Smartphone

**3.3 ARDUINO UNO**

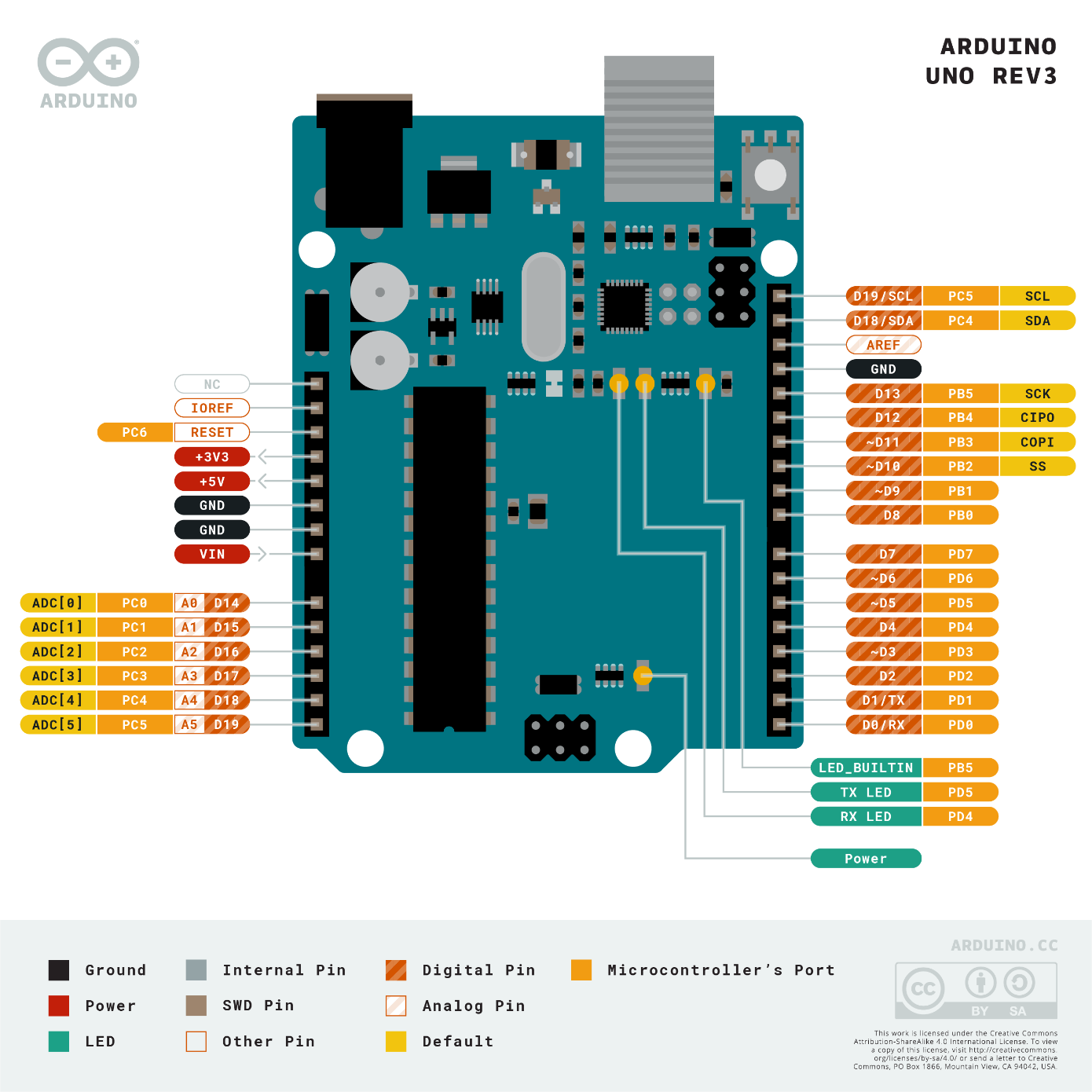


Fig. 1.9. Arduino Uno PCB Architecture With Various Ports.

The **Arduino Uno** is an [open-source](https://en.wikipedia.org/wiki/Open-source) [microcontroller board](https://en.wikipedia.org/wiki/Microcontroller_board) based on the [Microchip](https://en.wikipedia.org/wiki/Microchip_Technology) [ATmega328P](https://en.wikipedia.org/wiki/ATmega328P) microcontroller and developed by [Arduino.cc](https://en.wikipedia.org/wiki/Arduino). As shown in Fig1.9, the board is equipped with sets of digital and analog [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O) pins that may be interfaced to various [expansion boards](https://en.wikipedia.org/wiki/Expansion_board) (shields) and other circuits. The board has 14 digital I/O pins (six capable of [PWM](https://en.wikipedia.org/wiki/Pulse-width_modulation) output), 6 analog I/O pins, and is programmable with the [Arduino IDE](https://en.wikipedia.org/wiki/Arduino#Software) (Integrated Development Environment), via a type B [USB cable](https://en.wikipedia.org/wiki/USB_cable). It can be powered by the USB cable or by an external [9-volt battery](https://en.wikipedia.org/wiki/9-volt_battery), though it accepts voltages between 7 and 20 volts. [7]

Arduino Pin-Out Configuration

|  |  |  |
| --- | --- | --- |
| Pin Category | Pin Name | Details |
| Power | Vin, 3.3V, 5V, GND | Vin: Input voltage to Arduino when using an external power source.  5V: Regulated power supply used to power microcontroller and other components on the board.  3.3V: 3.3V supply generated by on-board voltage regulator. Maximum current draw is 50mA.  GND: ground pins. |
| Reset | Reset | Resets the microcontroller. |
| Analog Pins | A0 – A5 | Used to provide analog input in the range of 0-5V |
| Input/Output Pins | Digital Pins 0 – 13 | Can be used as input or output pins. |
| Serial | 0(Rx), 1(Tx) | Used to receive and transmit TTL serial data. |
| External Interrupts | 2, 3 | To trigger an interrupt. |
| PWM | 3, 5, 6, 9, 11 | Provides 8-bit PWM output. |
| SPI | 10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK) | Used for SPI communication. |
| Inbuilt LED | 13 | To turn on the inbuilt LED. |
| TWI | A4 (SDA), A5 (SCA) | Used for TWI communication. |
| AREF | AREF | To provide reference voltage for input voltage. |

Table 1.2 Pinout Configuration for Arduino Uno

**3.4 MICRO SERVO SG90**

Micro Servo Motor SG90 is a tiny and lightweight server motor with high output power. Servo can rotate approximately 180 degrees (90 in each direction)

* 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)

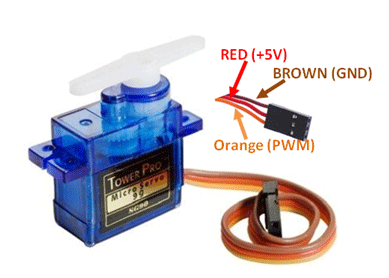


Fig. 1.3. Micro Servo SG90

Fig. 1.10. Micro Servo SG90

The Servo Motor Micro SG90 as shown in Fig 1.10 work well for basic servo experimentation and can be used in applications where small size is a virtue and that don’t require a huge amount of torque, but they are still pretty strong. Gears are nylon which is the case with lower cost Servos.

Servo motors can be commanded to go to a specific position and so are the usual go-to motor when accurate positioning is needed, such as for turning the front wheels on an RC model for steering or pivoting a sensor to look around on a robotic vehicle.

Servo motors are comprised of a DC motor, gears, a potentiometer to determine its position and a small electronic control board. Fig 1.11 shows the internal construction of a servo motor.

Standard servos have a specified limited range.  This is usually specified as 180 degrees.  Frequently the actual range is less than the full 180 degrees and is limited by the mechanical gears and potentiometer used for position sensing that is contained in the device.  If the motor is run all the way to 0 or 180, it may start making unhappy sounds and start vibrating as it tries to drive to a position that it cannot get to.  This causes a high stall current condition and has the potential of stripping gears and damaging the motor, so it is best to either drive it to a safely reduced range such as 20-160 or experiment a bit to determine the actual usable range if you want to maximize the range. Servos expect to see a pulse on their PWM pin every 20 milliseconds.  The pulse is active HIGH and the width of the pulse determines the position (angle) of the servos shaft.  The pulse can vary between 1mSec and 2mSec.  A 1mSec pulse positions the shaft at 0 degrees.  A 1.5mSec pulse positions the shaft at 90 degrees (centred in its range).  A 2 milliseconds pulse positions the shaft at 180 degrees.  Pulses with values between these can be used to position the shaft arbitrarily. [8]

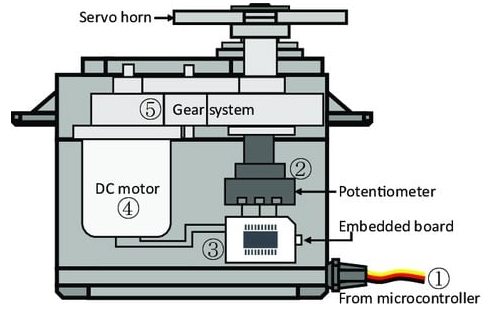


Fig. 1.11. Internal Diagram of SG 90 Servo

**3.5 HC-05 BLUETOOTH MODULE**

HC-05 Bluetooth module works in 2.4GHz range with 3Mbps speed which helps in controlling the wireless robotic arm though a Bluetooth compatible android device.

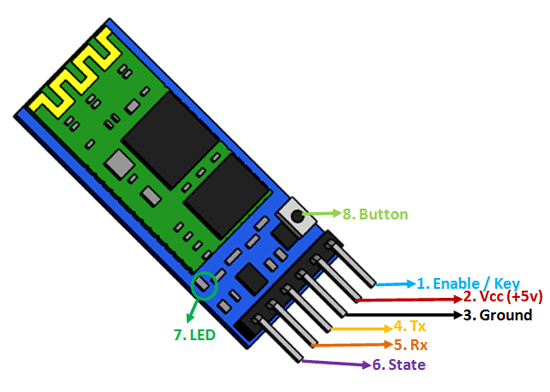


Fig. 1.12. HC-05 Bluetooth Module

HC-05 module as shown in Fig. 1.12 has a total of 6-pins with their stated below:

1. **Key/EN:** It is used to bring Bluetooth module in AT commands mode. If Key/EN pin is set to high, then this module will work in command mode. Otherwise by default it is in data mode. The default baud rate of HC-05 in command mode is 38400bps and 9600 in data mode.

HC-05 module has two modes,

          1.  **Data mode:**Exchange of data between devices.

          2.  **Command mode:**It uses AT commands which are used to change setting of HC-05. To send these commands to module serial (USART) port is used.

2.  **VCC:**Connect 5 V or 3.3 V to this Pin.

3.  **GND:**Ground Pin of module.

4.  **TXD:**Transmit Serial data (wirelessly received data by Bluetooth module transmitted out serially on TXD pin)

5.  **RXD:** Receive data serially (received data will be transmitted wirelessly by Bluetooth module).

6.  **State:**It tells whether module is connected or not.

The **HC-05** has two operating modes, one is the Data mode in which it can send and receive data from other Bluetooth devices and the other is the AT Command mode where the default device settings can be changed. We can operate the device in either of these two modes by using the key pin as explained in the pin description.

It is very easy to pair the HC-05 module with microcontrollers because it operates using the Serial Port Protocol (SPP). Simply power the module with +5V and connect the Rx pin of the module to the Tx of MCU and Tx pin of module to Rx of MCU.

During power up the key pin can be grounded to enter into Command mode, if left free it will by default enter into the data mode. As soon as the module is powered you should be able to discover the Bluetooth device as “HC-05” then connect with it using the default password 1234 and start communicating with it. The name password and other default parameters can be changed by entering into the application. [6]

**3.6 ARDUINO IDE**

The Arduino [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) application (for [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), [macOS](https://en.wikipedia.org/wiki/MacOS), and [Linux](https://en.wikipedia.org/wiki/Linux)) that is written in the [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) programming language. It originated from the IDE for the languages [Processing](https://en.wikipedia.org/wiki/Processing_(programming_language)) and [Wiring](https://en.wikipedia.org/wiki/Wiring_(development_platform)). It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, [brace matching](https://en.wikipedia.org/wiki/Brace_matching), and [syntax highlighting](https://en.wikipedia.org/wiki/Syntax_highlighting), and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus.

The Arduino IDE as shown in Fig 1.13 supports the languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) using special rules of code structuring. The Arduino IDE supplies a [software library](https://en.wikipedia.org/wiki/Software_library) from the [Wiring](https://en.wikipedia.org/wiki/Wiring_(development_platform)) project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable [cyclic executive](https://en.wikipedia.org/wiki/Cyclic_executive).

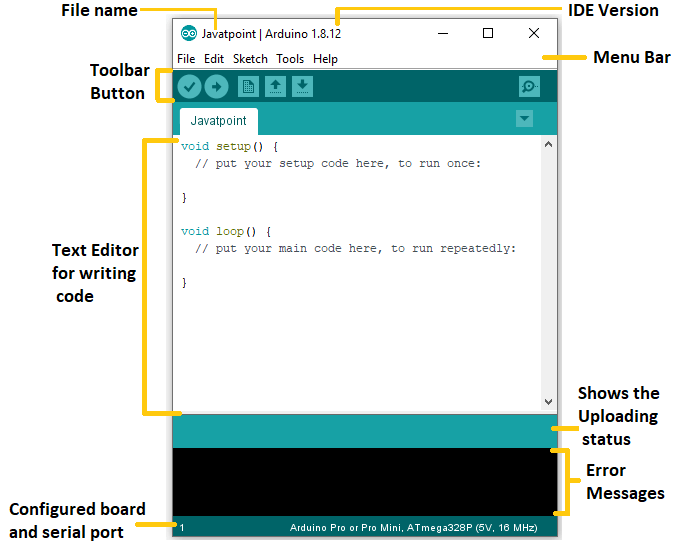


Fig. 1.13. Arduino IDE Interface

Programs written using Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.[9]

**3.7 ARDUINO RC**

It is an Android App which used for interface between smartphone and Arduino controller. It makes use of Bluetooth provision provided by smartphone to connect with HC-05 module.

This android application can establish a connection with any Arduino/microcontroller project that involves a Bluetooth module! It allows the user to set a UUID of his own Bluetooth module in order to connect the android application with The projects. The default UUID that comes with this application is for the HC-06 Wireless Serial 4 Pin Bluetooth RF Transceiver Module RS232.

The application is comprised of 4 Command Modes:  
  
1. Vehicle Mode : By pressing the "joystick" button the user can remote control a vehicle with specific gestures. For the interpretation of gestures to commands, the application uses the built-in accelerometer sensor of the smart device. There are 8 different gestures available (FRONT, BACK, LEFT, RIGHT, FRONT\_LEFT, FRONT\_RIGHT, BACK\_LEFT, and BACK\_RIGHT). You can set these gestures with your own custom commands through the menu button "Set Commands".  
Additionally, there is a STOP and an ANDROID button that can be assigned with custom commands as well.

2. Fader Mode: This mode provides the user with a fader-seek bar that can send commands from 0-9 and allow the user to experiment with servos, LEDs, motors and many more.  
  
3. Controller Mode: This mode provides a standard controller layout with 10 buttons in total that can be assigned with custom commands. The arrow buttons are specifically designed to send commands continuously while pressed, in order to simulate an actual controller.  
  
4. Terminal Mode: This mode provides a terminal-like interface in which the user can type and send an individual or multiple commands which will execute sequentially. Since Arduino serial port receives one byte at a time, if the user types a string in the input text field, the application divides that string into characters and sends them one by one to Arduino. It should be noted that spaces or empty strings are skipped. [10] Fig 1.14 shows the Android RC Application Interface as available on Android 2.3.3 or higher.



Fig. 1.14. Arduino RC Android Application Interface

**3.8 IMPLEMENTATION & DESIGN OF ROBOTIC ARM USING ARDUINO**

**3.8.1 Flow Chart for Understanding Operations of Robotic Arm**

Flow Chart is a pictorial representation flow of control and data in a program. The flow chart for the implementation of Robotic Arm Using Arduino is unique as it incorporates seamless wireless data transfer between varied levels of software and hardware encapsulation.

1. We start by initializing ports for transmission and reception followed by ports for servo motor connections.
2. The program then follows an infinite loop that is in execution until the power supply is terminated.
3. Input is passed by the user. A connector checker checks if the BT module is ready for communication with the Arduino Board. The results of this check can be viewed on the Serial Monitor.
4. If the connection checker is successful, then the input flows through a series of IF conditions. In case a particular condition is found to be true, then the statement corresponding to that condition is executed and a signal is sent to the respective servo to perform the operations as specified in 5).
5. If button pressed is left or right arrows on remote control, then base is rotated in clockwise or counter clockwise direction respectively. If button pressed is up or down then shoulder arm is moved. On the other hand buttons to control Elbow movements are pressed, elbow moves up and down. Similarly Grippers are controlled by pressing square and circle buttons

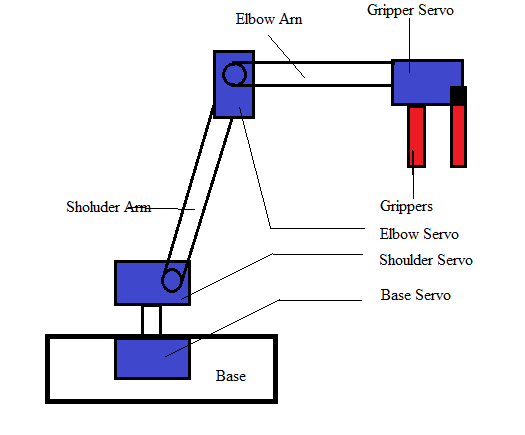
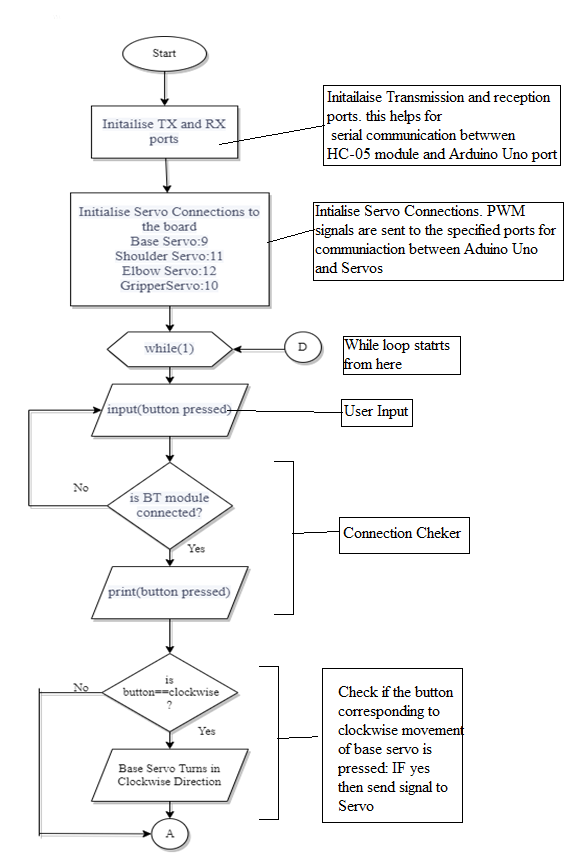
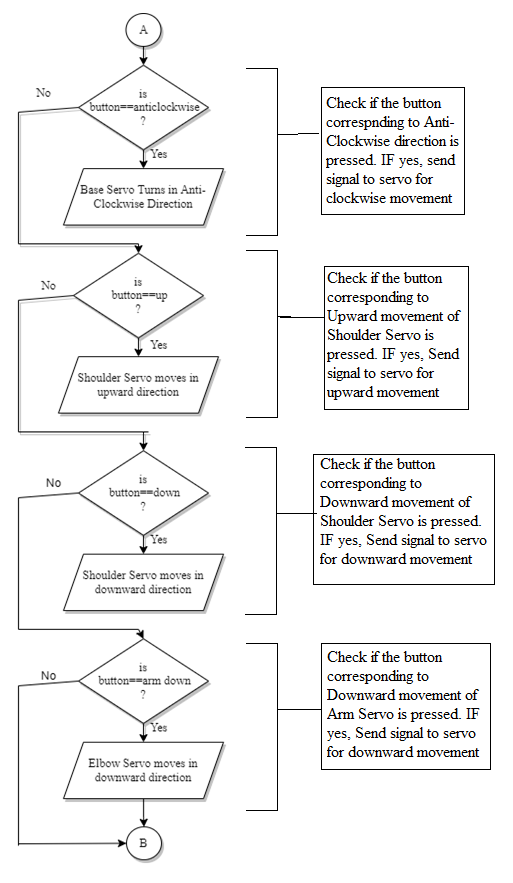
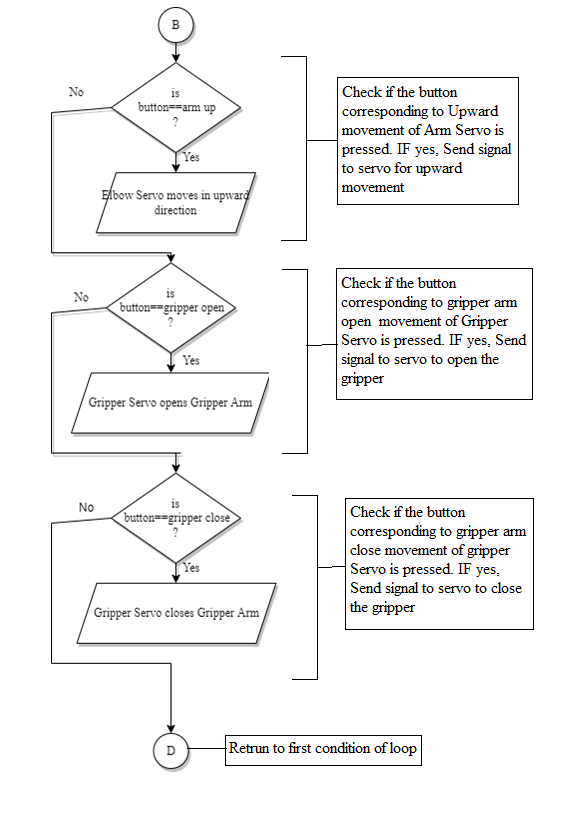


Fig 1.15 Robotic Arm with position of various Servos and their names

The diagram in Fig 1.15 helps us to understand various position of servos and their movements while referring to the flow chart

****

****

****

**3.8.2 Procedure for implementation of Robotic Arm**

1. Install Arduino.Ide application from Arduino CC official website.

Once the application is installed, click on the icon  and open the application



Fig. 1.15. Desktop Window with Arduino Icon

1. Once the application opens, the following screen appears. Go to File and select New to ensure that new file is opened and previous files are closed.

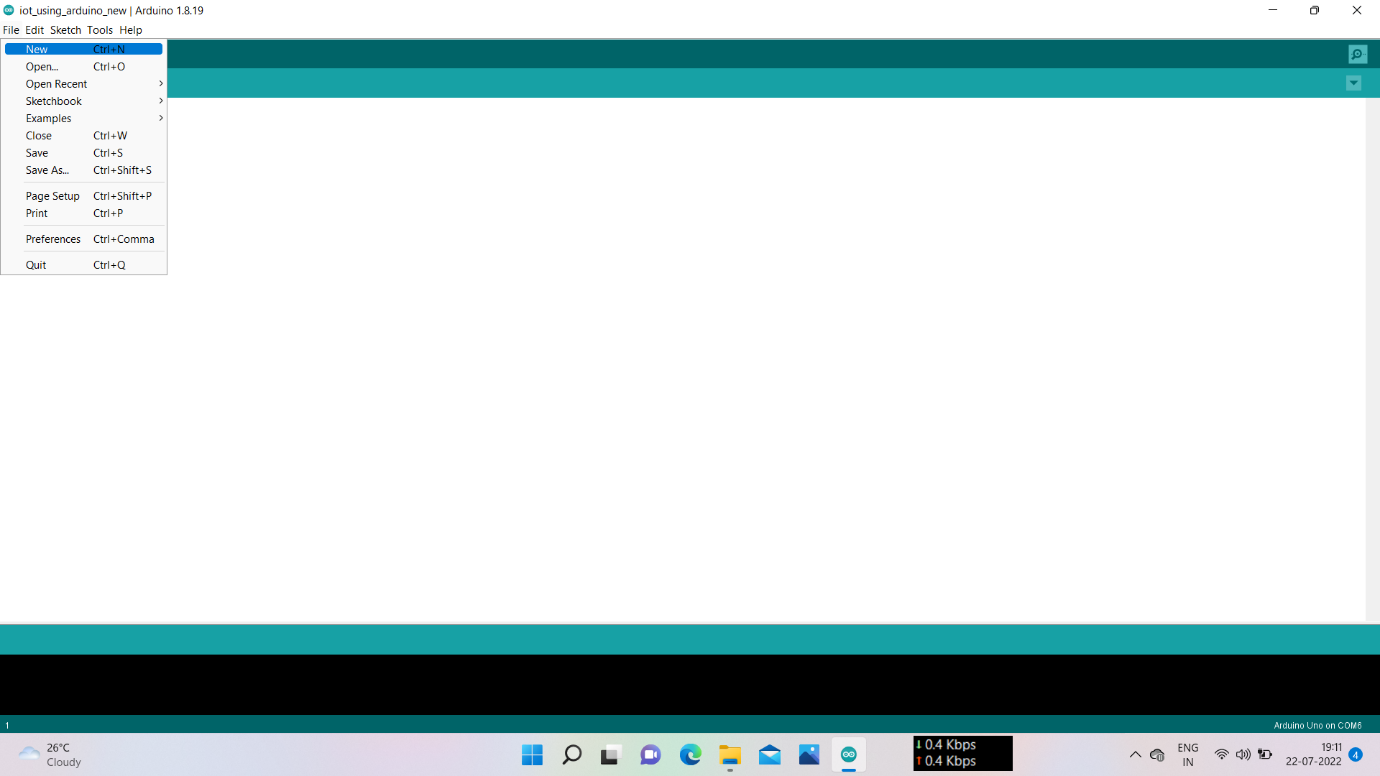


Fig. 1.16 Arduino IDE Text Editor Window with File Menu

1. Type the program code on the text editor. Use proper comments while typing for future upgradation if required. Also check for angle settings for Servo motors.

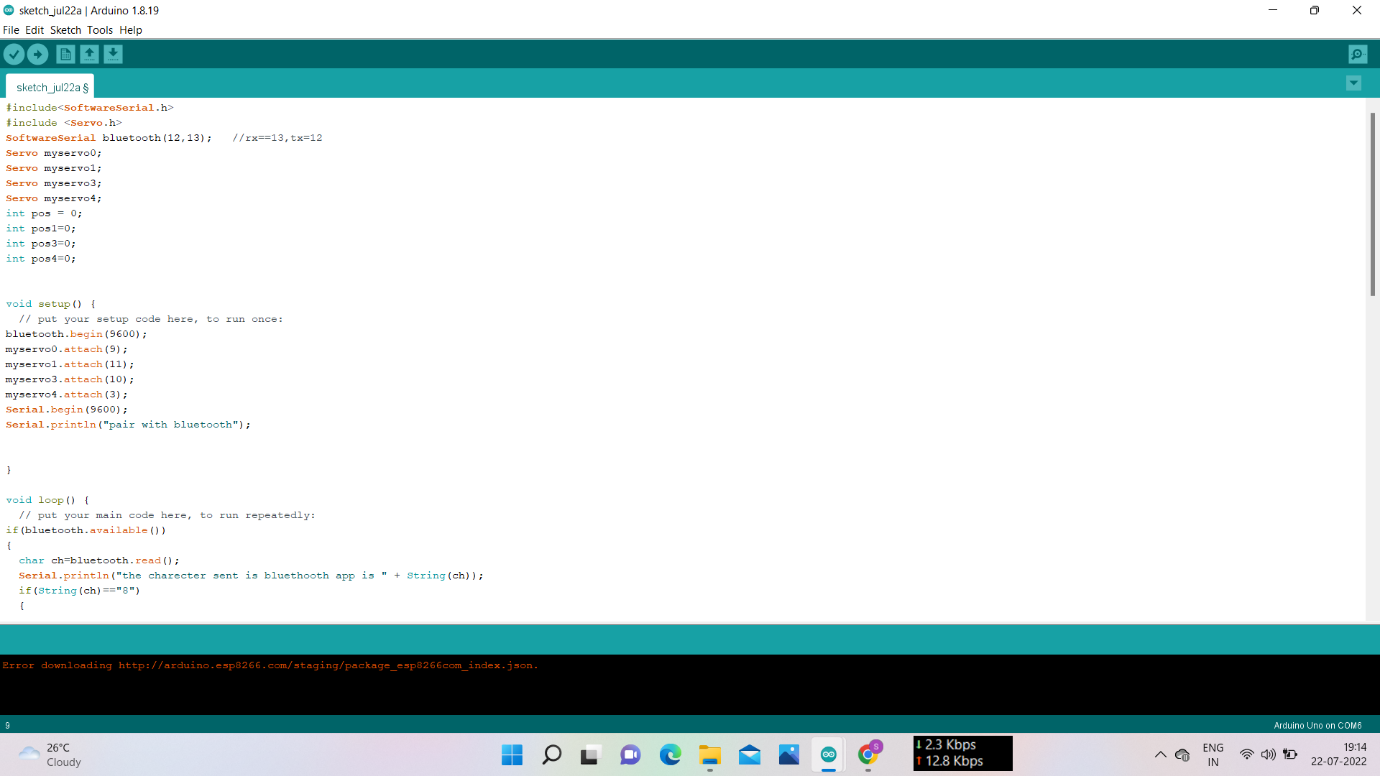


Fig. 1.17 Arduino IDE Text Editor Window with code typed

1. Save the program. Choose the folder so that all the related file information and documents can be stored at one location. Create folder named ARM\_Project in Local Disk D and store the code file there.

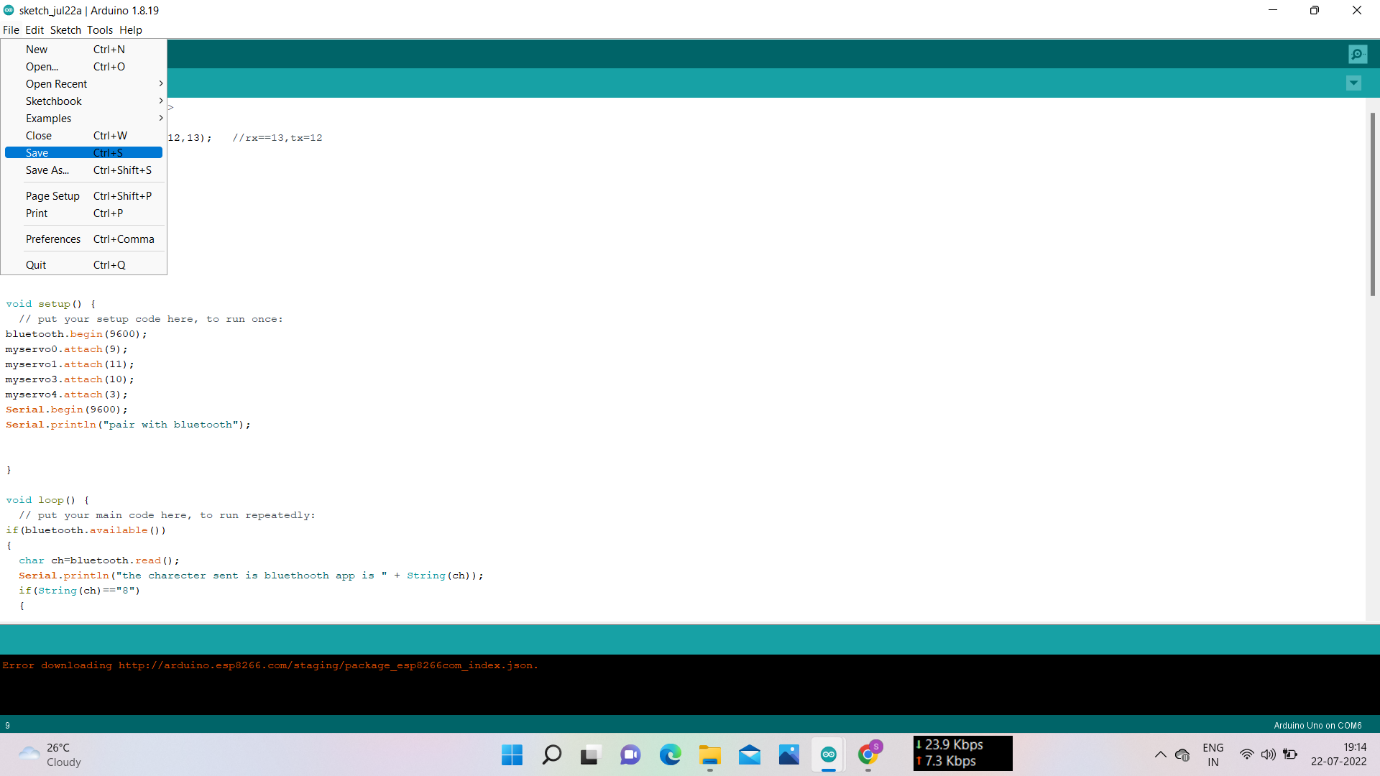


Fig. 1.18 Arduino IDE Text Editor Window with code being saved

1. Once the file is stored, click Tools, select Board: click on Arduino AVR Boards and select Arduino Uno.

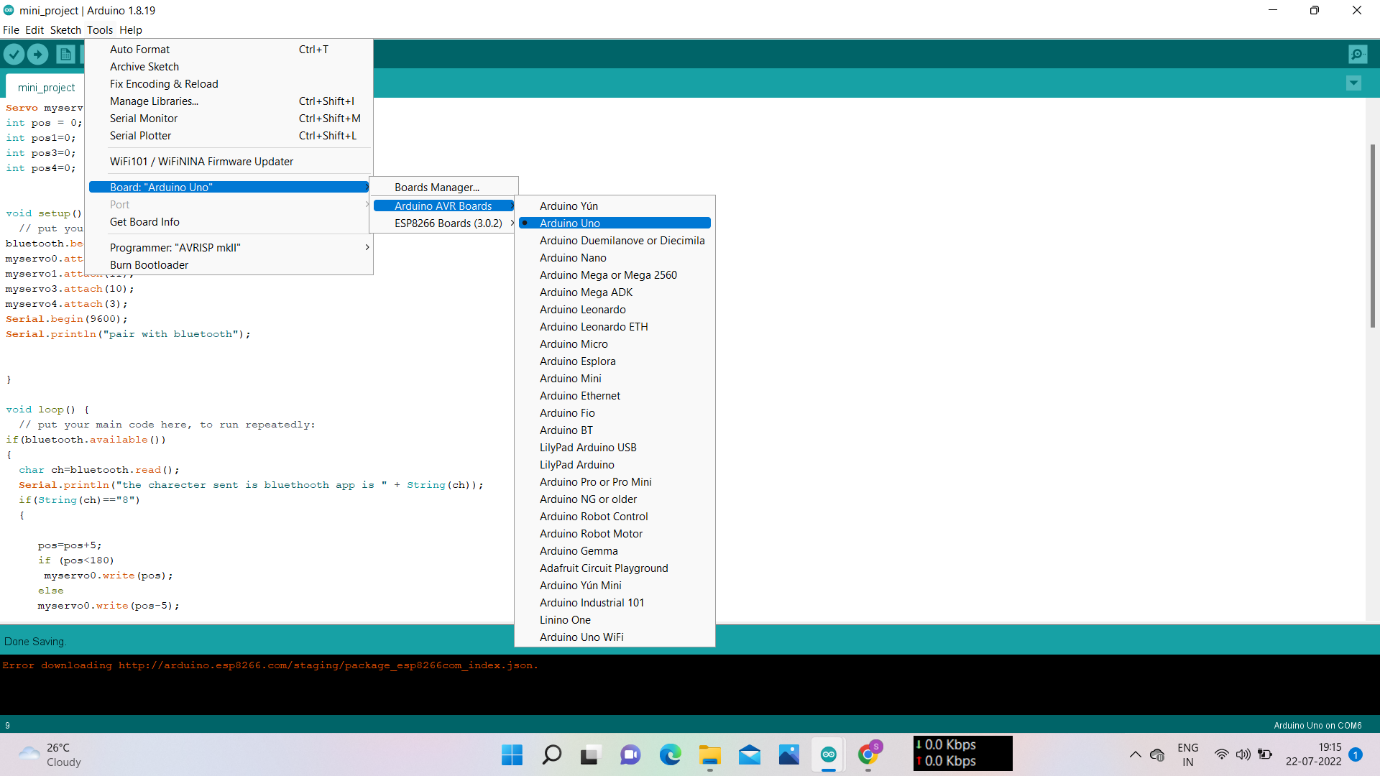


Fig. 1.19 Arduino IDE Text Editor Window with Board Settings Option

1. Click on Tools, select Port and click Select COM3(Arduino Uno)

Note: COM port settings may vary depending on USB port being used. In that case COM port has to checked by opening Control Panel select Hardware And Sound, click Devices and Printers, select Check the COM port on Unspecified Devices List.

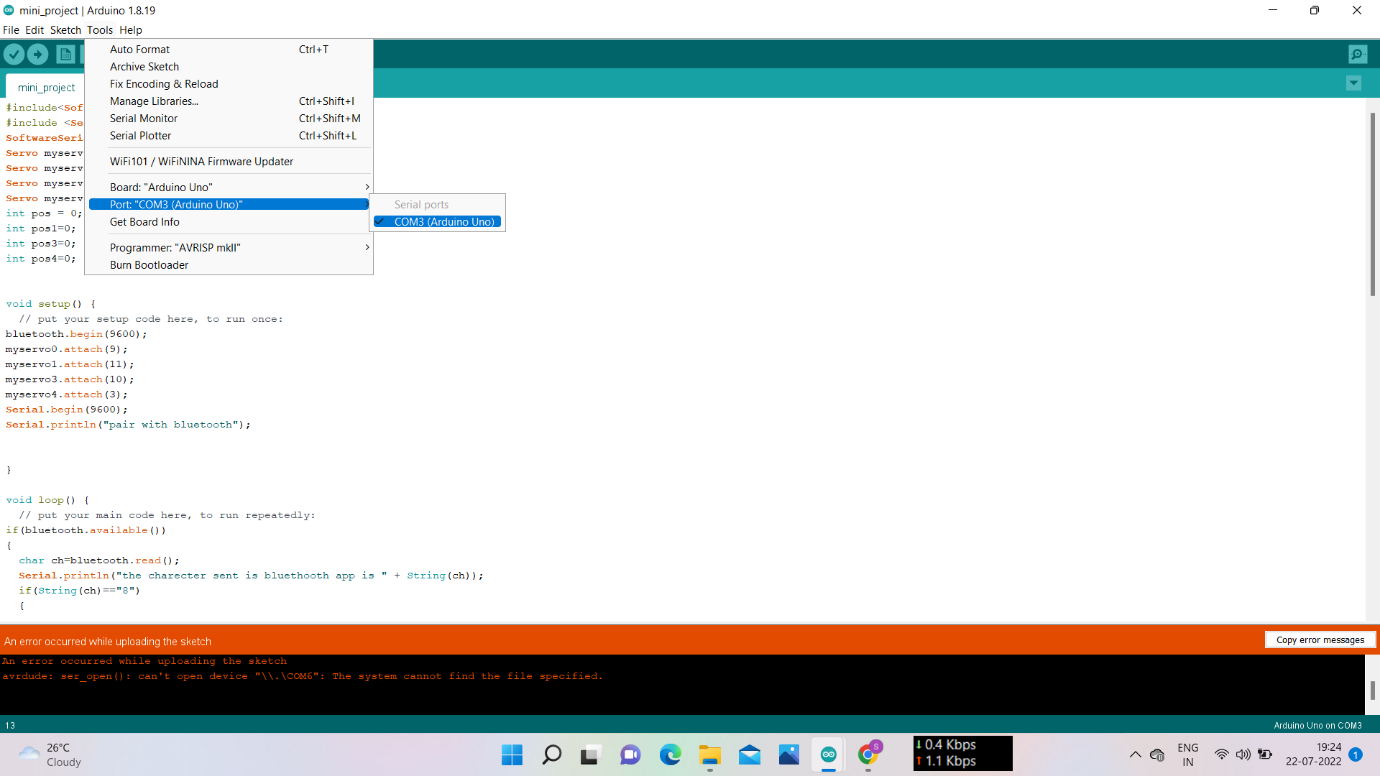


Fig. 1.20 Arduino IDE Text Editor Window with COM Port Settings Option

1. Click on Sketch, select Verify/Compile. Check for Syntax Errors and Runtime Errors. Make necessary change.
2. Connect the Circuit as shown below:

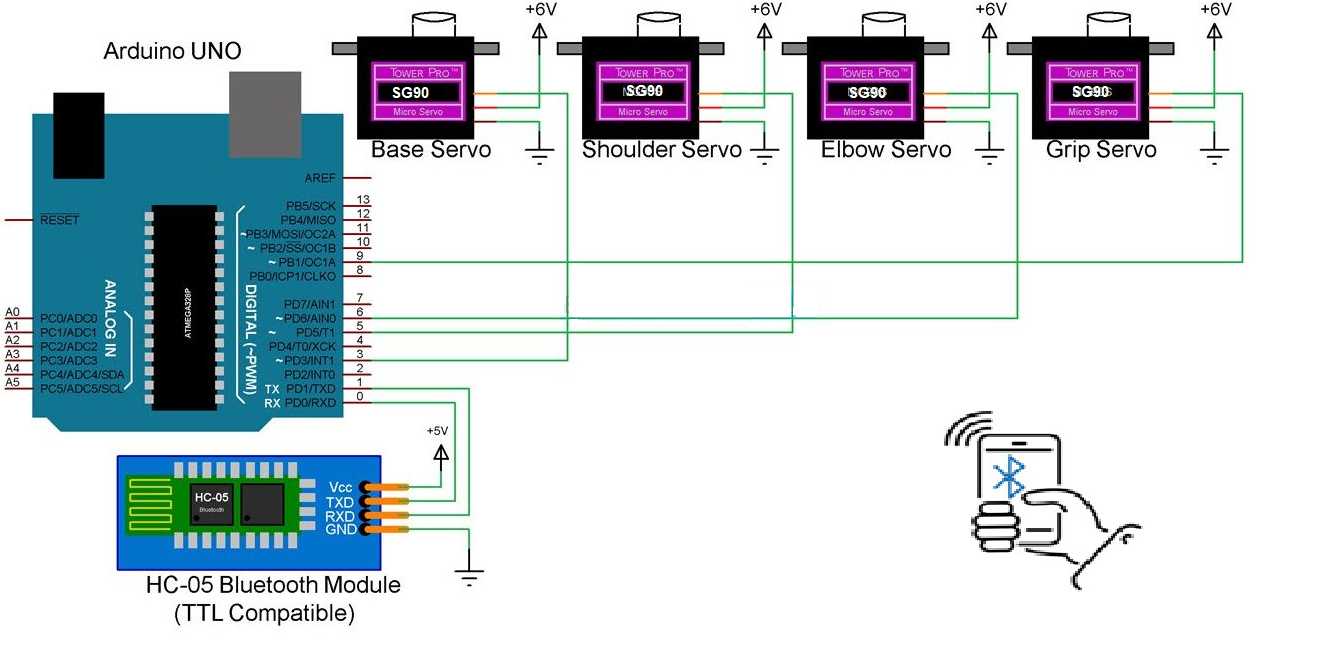
****

Fig. 1.21.Circuit Diagram for Implementation of Robotic Arm using Arduino

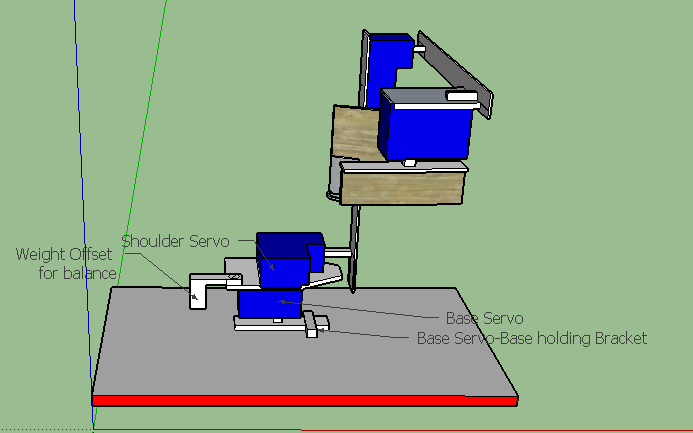
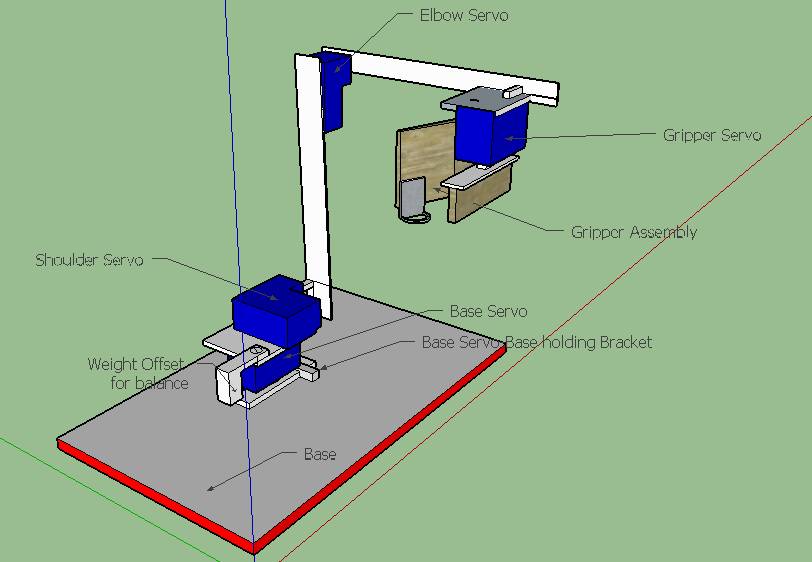
 

Fig. 1.22 .Robotic Arm Assembly showing various servos and their positions

1. Click Sketch and select Upload. The former part of Step 8 can be ignored and we can directly proceed with the latter part of Step 8.



Fig. 1.23 Arduino IDE Text Editor Window with Terminal Window at bottom.

1. If there are no errors in the code, we get “Done Uploading” message on Terminal Window present on the bottom side the Editor window.

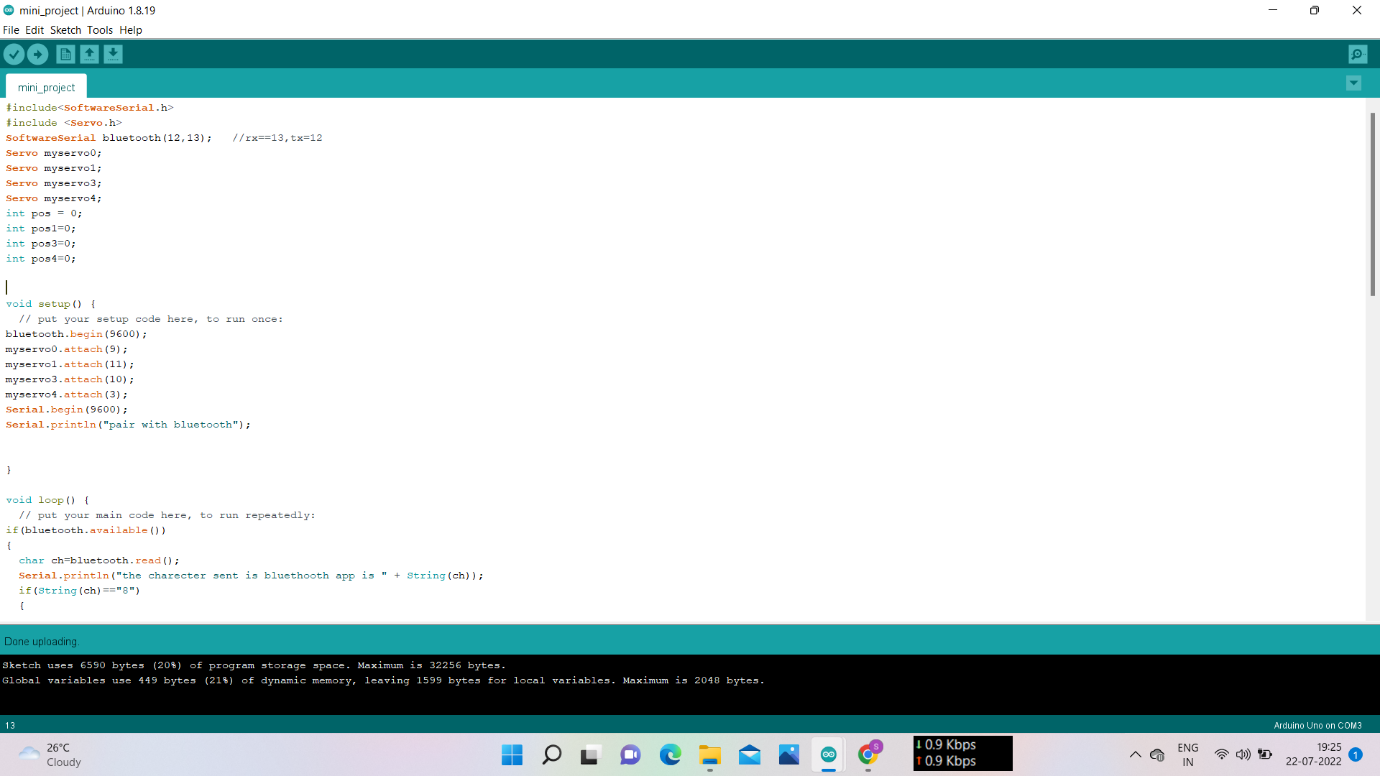


Fig. 1.24 Arduino IDE Text Editor Window with Terminal Window at bottom after uploading the code into board

1. On Smartphone download “Arduino Bluetooth Controller” via Playstore/Appstore. Pair the smartphone with HC-05 module using Bluetooth. After the application has been installed click on  icon to open the application. On the pop-up window in the application, select the device (HC-05 module).



Fig. 1.25 Arduino Bluetooth Controller Application Window on Smartphone.

1. The Mode window opens as shown below. Click on the “Controller Mode”



Fig. 1.26 Arduino Bluetooth Controller Application Window on Smartphone with Various Modes.

1. In the Controller Mode click on Three-dots Icons - Free SVG & PNG Three-dots Images - Noun Project symbol. Select “Set Commands” option.

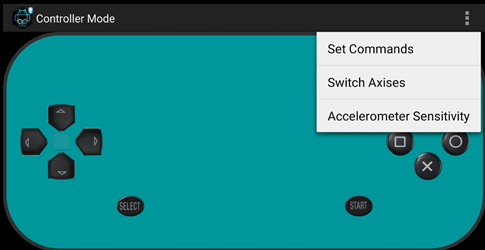


Fig. 1.27 Arduino Bluetooth Controller Application Window in Controller Mode

1. Change the commands as shown below:

UP ARROW:=6

DOWN ARROW:=4

LEFT ARROW:= 8

RIGHT ARROW:=2

TRIANGLE BUTTON:=Y

CROSS BUTTON:=X

SQUARE BUTTON:=C

CIRCLE BUTTON:=D

START BUTTON:=E

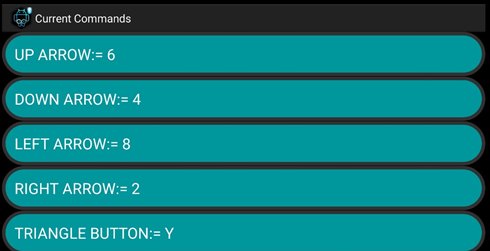


Fig. 1.28 Arduino Bluetooth Controller Application Window in Button Configuration during Updation.

1. On Computer: Click on Tools and select Serial Monitor or press Ctrl+Shift+M. This opens Serial Monitor. We can check the interface between HC-05 and host device.

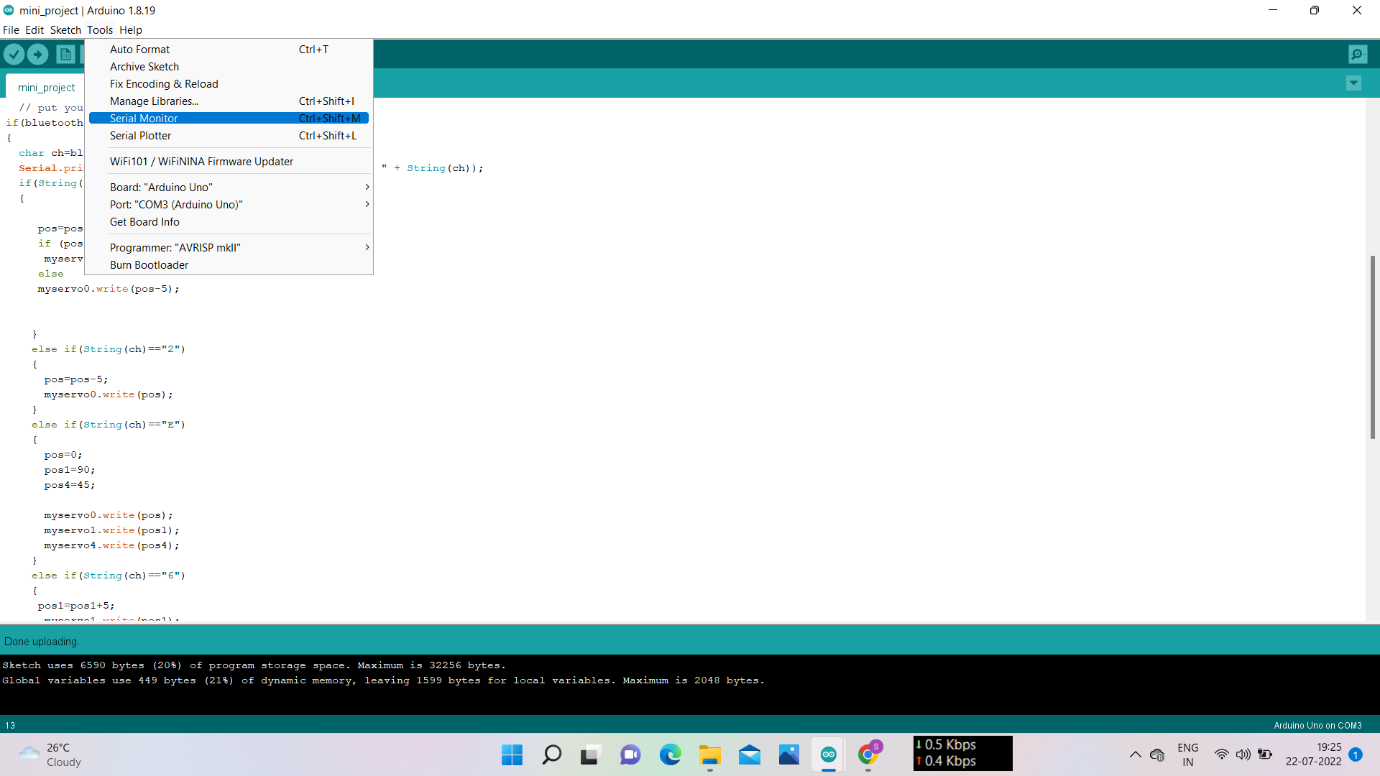


Fig. 1.29 Serial Monitor in Tools Menu on Arduino IDE

1. Press the control buttons on smartphone and check the Command window for output. This verifies the communication between smartphone and HC-05 device.

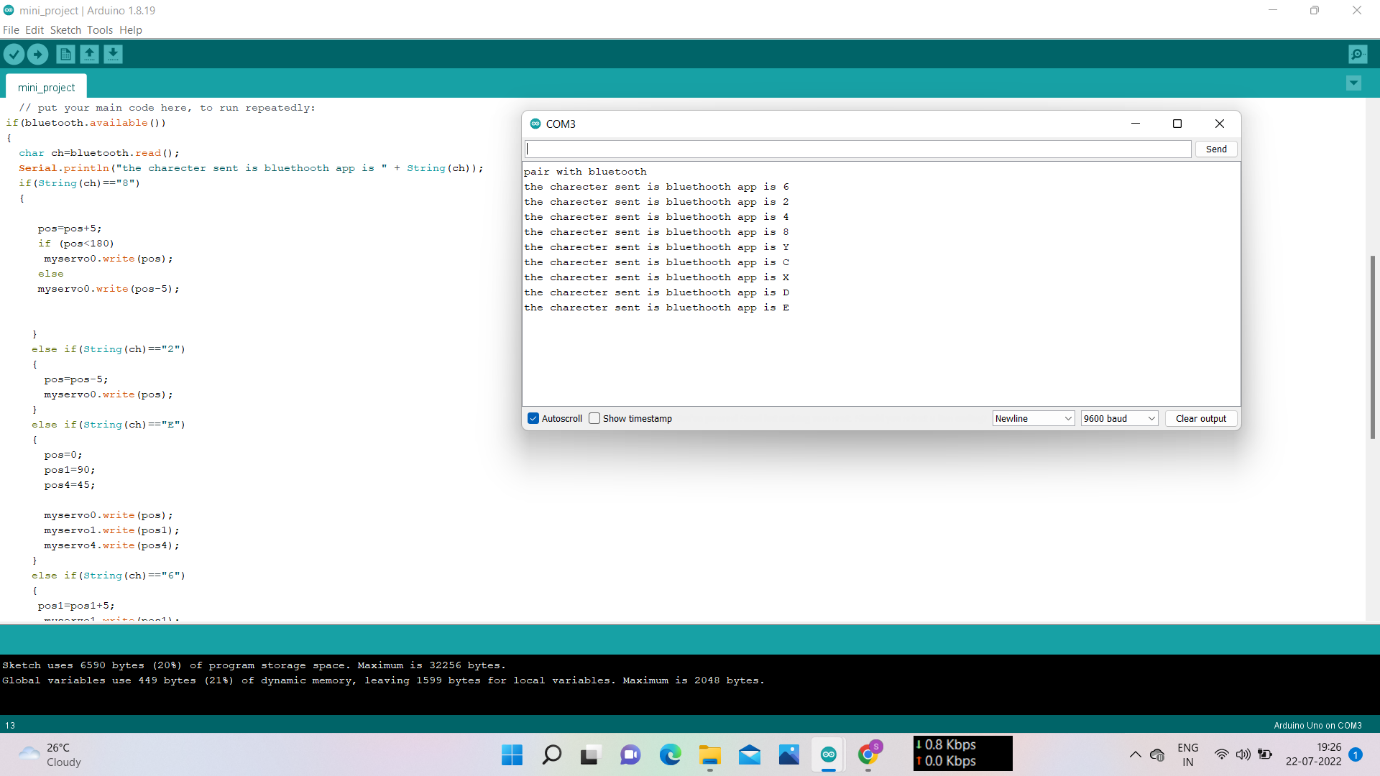


Fig. 1.30 Serial Monitor on Arduino IDE

1. Once the communication interface has been established, start controlling the arm using buttons smartphone application.

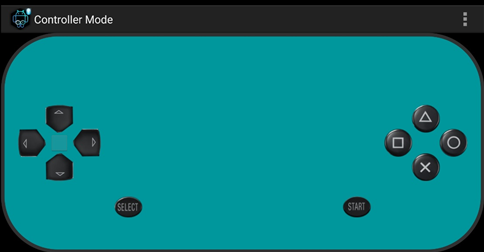


Fig. 1.31 Controller Mode on Arduino Bluetooth Controller Application

**3.9 RESULTS FOR IMPLEMENTATION OF ROBOTIC ARM**

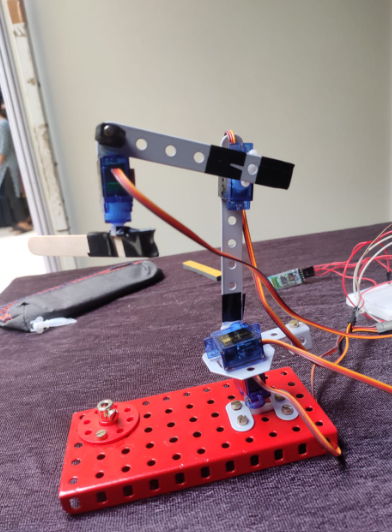
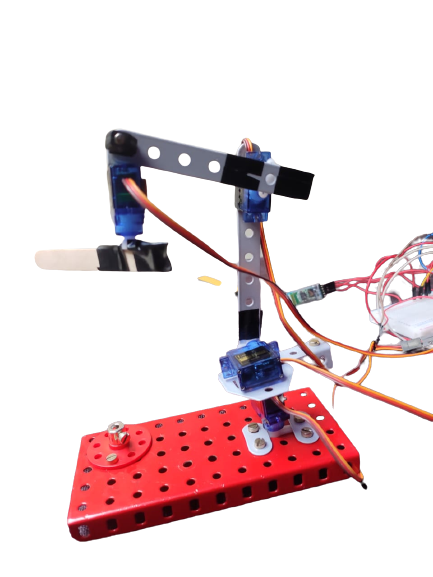
**** 

Fig. 1.32 Robotic Arm (Side View) Fig. 1.33 Robotic Arm (Enhanced)

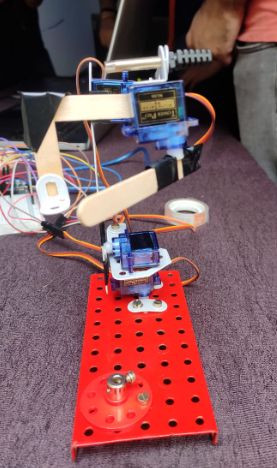
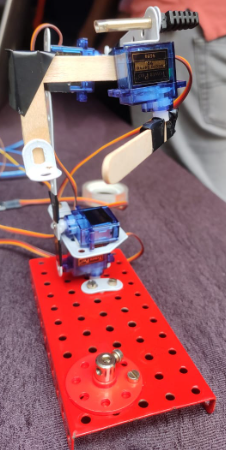
 ****

Fig.1.34 Robotic Arm (Front View) Fig. 1.35

Fig 1.32 to Fig 1.35 represents the robotic arm assembly as viewed by observers from various angles.

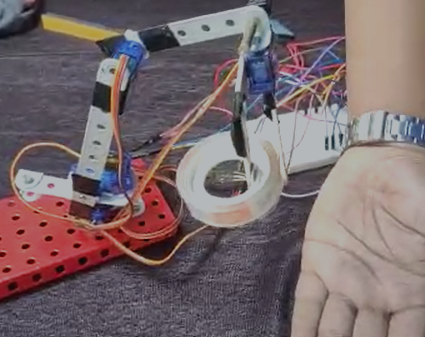


Fig. 1.36 Robotic Arm Lifting a Cellophane Tape

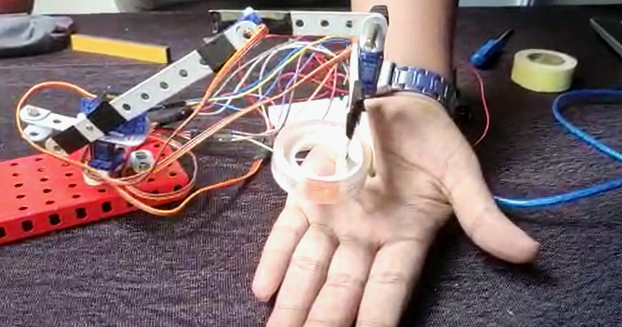


Fig. 1.37 Robotic Arm Dropping a Cellophane Tape on hand of demonstrator

Fig 1.36 and Fig 1.37 shows Robotic Arm lifting a load of cellophane tape from a spot on table and dropping it on the palm of the observer.

**CHAPTER 4**

**CONCLUSION**

**CHAPTER 4**

**CONCLUSION**

**4.1 CONCLUSION**

1. The hardware project has performed the basic operations of robotic arm as expected.
2. The main problem to overcome in this project was to interface the Arduino UNO board with the android device via Bluetooth module.
3. Android based robotic arm provides remote access to hard-to-reach places and with android based interface it comes under the category of user-friendly projects.

**4.2 APPLICATIONS OF ROBOTIC ARM**

**Palletizing**

Robotic arms can be used to automate the process of placing goods or products onto pallets. By automating the process, palletizing becomes more accurate, cost-effective, and predictable. The use of robotic arms also frees human workers from performing tasks that present a risk of bodily injury. Fig 1.38 shows a palletizing Arm in operation.



Fig. 1.38. Palletizing Robotic Arm

**Material Handling**

Material-handling robotic arms can help create a safe and efficient warehouse by ensuring goods and materials are properly stored, easy to find, or transported correctly. Automating these processes can help accelerate the delivery of goods to customers, prevent workplace accidents, and improve the efficiency of a facility. Fig 1.39 shows a Material Handling Robot lifting a heavy load.



Fig. 1.39. Material Handling Robotic Arm

**Welding**

Welding is a task that can be performed by robots in advanced industrial settings such as automotive manufacturing. Given its critical impact on product quality, welding is an excellent candidate for advanced robotics with vision and AI augmentation for inline quality inspection.

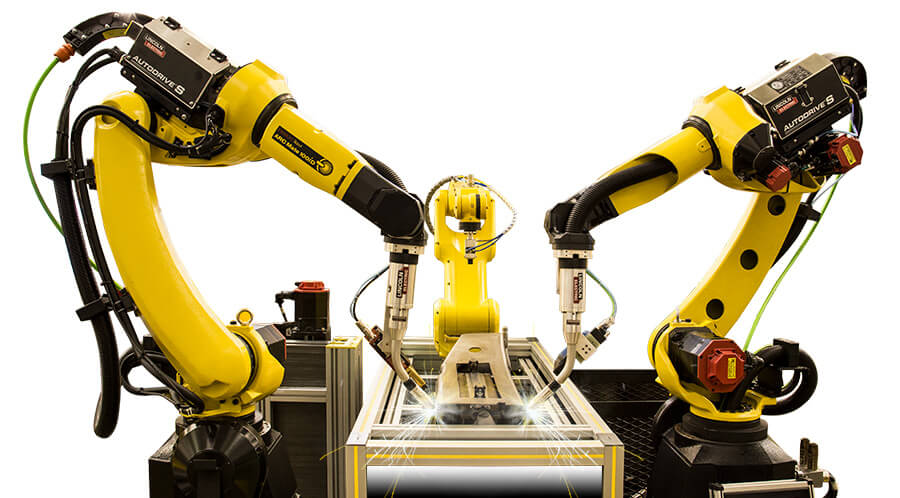


Fig. 1.40. Welding Robotic Arm

**Inspection**

Performing quality inspection is typically completed at the end of a production line, which delays the detection of production quality issues. By enhancing robots with vision and AI systems, businesses can benefit from real-time inspection, helping to reduce waste and downtime. Fig 1.41 is a Robotic Arm used for inspection of undercarriage of vehicles.



Fig. 1.41. Robotic Arm for Inspection

**Pick and Place**

Pick-and-place robots are typically used in modern manufacturing and logistics. They are equipped with advanced machine vision systems to identify an object, grasp it, and move it from one location to another —quickly and efficiently— to increase speed of production and distribution of goods. Fig 1.42 is a Pick and Place Robot

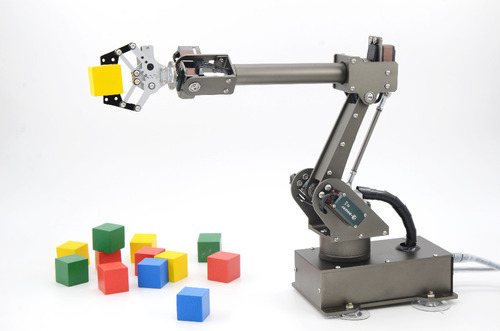


Fig. 1.42. Pick and Place Robotic Arm

**4.3 FUTURE WORK**

1. This project can be taken another notch upwards by replacing Bluetooth module with GSM module to increase the range.
2. A camera can be implemented along with Digital Image Processing Techniques to increase the applicability of the system.
3. A pedestal with wheels/axle can be used in synchronous with the existing arm system to demonstrate applications such as bomb defusal robot, farm-produce picking robot, etc.

## 4.4 REFERENCES

1. Adarsh V Patil, Sreevarsha Prakash, Akshay S, Mahadevaswamy, Chandanb Patgar and SharathKumar A. J, 2018, Android Based Smart Door Locking System, International Research Journal of Engineering and Technology (IRJET) NCESC – 2018 (Volume 6 – Issue 13)
2. V. Patidar and R. Tiwari, "Survey of robotic arm and parameters," 2016 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, 2016, pp. 1-6.
3. Kumar Aaditya, Divesh Kumar Pande and Preksha Moondra, “Android controlled pick and place robotic arm vehicle”, 2015, International Research Journal of Engineering and Technology (IRJET), pp. 604-608.
4. Ramkrishna Ghogre, Harshad Mehta, Ankit Khobragade, Ashutosh Tripathi, “Arduino Based Android Controlled Robotic Arm”, 2020, International Research Journal of Engineering and Technology (IRJET).
5. Arduino, “Servo Motor Basics with Arduino”. Retrieved from <https://docs.arduino.cc/learn/electronics/servo-motors>.
6. Components 101, “HC-05 – Bluetooth Module”, 2021. Retrieved from <https://components101.com/wireless/hc-05-bluetooth-module>.
7. Wikipedia, “Arduino Uno”, 2022. Retrieved from Wikipedia archives available at <https://en.wikipedia.org/wiki/Arduino_Uno>.
8. Proto Supplies, “Servo Motor Micro SG”, 2022. Retrieved from <https://protosupplies.com/product/servo-motor-micro-sg90/>.
9. Wikipedia, “Arduino”, 2022. Retrieved from Wikipedia archives available at <https://en.wikipedia.org/wiki/Arduino#IDE>.
10. [Ioannis Tzanellis](https://play.google.com/store/apps/developer?id=Ioannis+Tzanellis), “Arduino Bluetooth Controller”, 2017, Google Play Services, Google LLC.

**CHAPTER 5**

**APPENDIX**

**CHAPTER 5**

**APPENDIX**

**5.1 Code for the program**

#include<SoftwareSerial.h>

#include <Servo.h>

SoftwareSerial bluetooth(12,13); //rx==13,tx=12

Servo myservo0;

Servo myservo1;

Servo myservo3;

Servo myservo4;

int pos = 0;

int pos1=0;

int pos3=0;

int pos4=0;

void setup() {

// put your setup code here, to run once:

bluetooth.begin(9600);

myservo0.attach(9);

myservo1.attach(11);

myservo3.attach(10);

myservo4.attach(3);

Serial.begin(9600);

Serial.println("pair with bluetooth");

}

void loop() {

// put your main code here, to run repeatedly:

if(bluetooth.available())

{

char ch=bluetooth.read();

Serial.println("the charecter sent is bluethooth app is " + String(ch));

if(String(ch)=="8")

{

pos=pos+5;

if (pos<180)

myservo0.write(pos);

else

myservo0.write(pos-5);

}

else if(String(ch)=="2")

{

pos=pos-5;

myservo0.write(pos);

}

else if(String(ch)=="E")

{

pos=0;

pos1=90;

pos4=45;

myservo0.write(pos);

myservo1.write(pos1);

myservo4.write(pos4);

}

else if(String(ch)=="6")

{

pos1=pos1+5;

myservo1.write(pos1);

}

else if(String(ch)=="4")

{

pos1=pos1-5;

myservo1.write(pos1);

}

else if(String(ch)=="C")

{

pos3=45;

myservo3.write(pos3);

}

else if(String(ch)=="D")

{

pos3=0;

myservo3.write(pos3);

}

else if(String(ch)=="X")

{

pos4=pos4+10;

myservo4.write(pos4);

}

else if(String(ch)=="Y")

{

pos4=pos4-10;

myservo4.write(pos4);

}

}

}