

A project report submitted in partial fulfillment of requirement for the course On

**Smart System Design**

**By**

D.CHAITANYA (2203A51043)

D.HASINI (2203A51044)

G.AKSHITHA (2205A42007)

K.VISHWARAM (2205A42009)

A.SANTHOSH (2203A51764)

Under the guidance of

**Mr. Rajeshwar Rao Arabelli**

Asst. Prof. & Director, Centre for Embedded Systems and IoT, Department of ECE

&

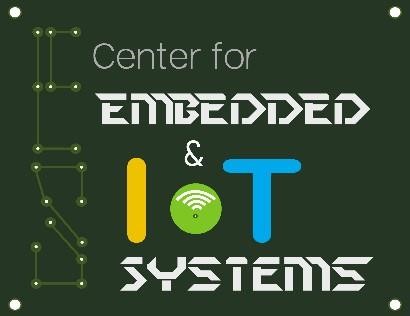
**Dr. Ch.Rajendra Prasad**

Asst. Prof., Department of ECE



**Center for Embedded Systems and Internet of things**

**SR UNIVERSITY**



**CERTIFICATE**

**This is to certify that the course project entitled “ROBO FIGHTING” is the bonafide work carried out by D.CHAITANYA(2203A51043),D.HASINI(2203A51044), G.AKSHITHA (2205A42007), K.VISHWARAM (2205A42009),A.SANTHOSH**

**(2203A51764) in the partial fulfillment of the requirement for the award of course Smart System Design** during the academic year 2022-2023 under our guidance and Supervision.

**Mr. Rajeshwar Rao Arabelli** Asst. Prof. & Director, Centre for Embedded Systems and IoT,

Department of ECE

# CHAPTER 1 INTRODUCTION

## 1.1 INTRODUCTION

Robot combat is a mode of robot competition in which custom-built machines fight using various methods to incapacitate each other. The machines have generally been remotecontrolled vehicles rather than autonomous robots. Robot combat competitions have been made into television series, including Robot Wars in the UK and Battlebots in the US. These shows were originally broadcast in the late 1990s to early 2000s and experienced revivals in the mid2010s. As well as televised competitions, smaller robot combat events are staged for live audiences such as those organized by the Robot Fighting League.

Robot builders are generally hobbyists and the complexity and cost of their machines can vary substantially. Robot combat uses weight classes, with the heaviest robots able to exert more power and destructive capabilities. The rules of competitions are designed for the safety of the builders, operators, and spectators while also providing for an entertaining spectacle. Robot combat arenas are generally surrounded by a bulletproof screen. Competitor robots come in a variety of designs, with different strategies for winning fights. Robot designs typically incorporate weapons for attacking opponents, such as axes, hammers, flippers, and spinning devices. Rules almost always prohibit gun-like weapons as well as other strategies not conducive to the safety and enjoyment of participants and spectators.

## 1.2 OBJECTIVES

➢ The objective of this project is to implement a low cost , reliable, and scalable home automation system that can be used to remotely switch on or off any house hold appliance

**CHAPTER 2**

**PROJECT DESCRIPTION**

## 2.1 BLOCK DIAGRAM OF THE PROJECT

As shown in the above schematic diagram it mainly consists of an Arduino Uno, four Dc motors*,* HC-05 Bluetooth module, Motor Driver ,9V battery, four 12V motors. Bluetooth module is connected to RX connected to TX. TX is connected to RX. Vcc is connected to 5V

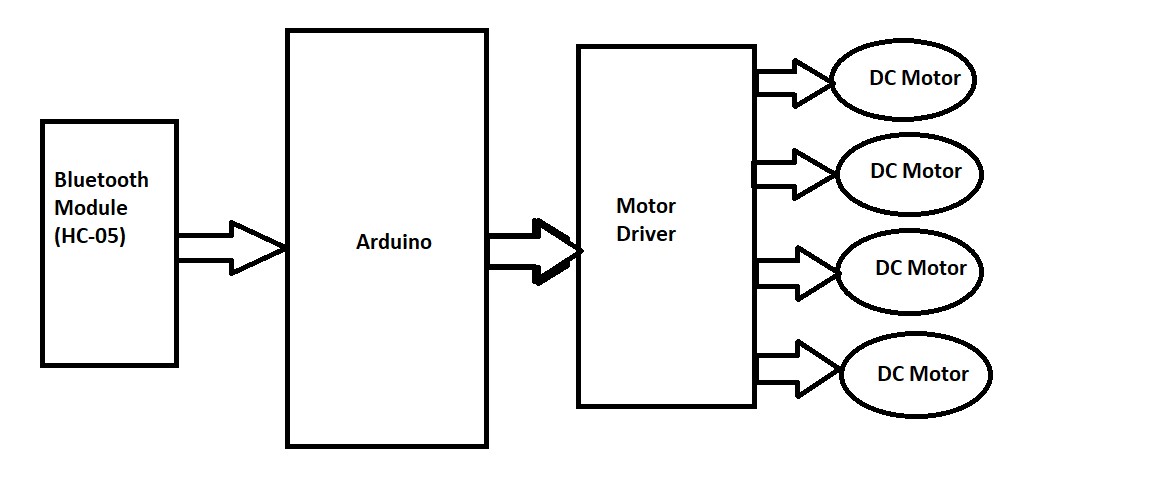


Fig.2.1 Block Diagram

### 2.2 HARDWARE DESCRIPTION

#### 2.2.1 Arduino UNO

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, 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. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a

USB-to-serial converter. Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line

to ground, making it easier to put into DFU mode. Revision 3 of the board has the following new features: 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes. Stronger RESET circuit. ATmega 16U2 replace the 8U2. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Microcontroller ATmega328

Operating Voltage 5V

Input Voltage (recommended) 7-12V

Input Voltage (limits) 6-20V

Digital I/O Pins 14 (of which 6 provide PWM output)

Analog Input Pins 6

DC Current per I/O Pin 40 mA

DC Current for 3.3V Pin 50 mA

Flash Memory 32 KB of which 0.5 KB used by bootloader

SRAM 2 KB (ATmega328)

EEPROM 1 KB (ATmega328)

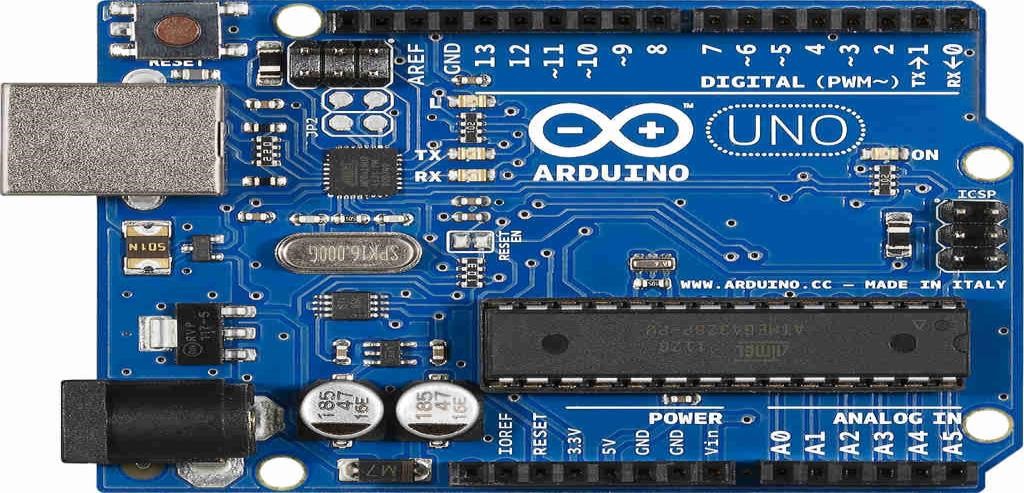


Fig. 2.2 Arduino Uno

**Applications:**

* Xoscillo, an open-source [oscilloscope](https://en.wikipedia.org/wiki/Oscilloscope)
* [Arduinome,](https://en.wikipedia.org/wiki/Arduinome) a [MIDI controller](https://en.wikipedia.org/wiki/MIDI_controller) device thn t mimics the [Monome](https://en.wikipedia.org/wiki/Monome)
* [OBDuino,](https://en.wikipedia.org/wiki/OBDuino) a [trip computer](https://en.wikipedia.org/wiki/Trip_computer) that uses the [on-board diagnostics](https://en.wikipedia.org/wiki/On-board_diagnostics) interface found in most modern cars
* Gameduino, an Arduino shield to create retro 2D video games
* ArduinoPhone, a do-it-yourself cellphone
* Water quality testing platform
* Automatic titration system based on Arduino and stepper motor
* Low cost data glove for virtual reality applications
* Impedance sensor system to detect bovine milk adulteration
* Homemade CNC using Arduino and DC motors with close loop control by Homofaciens
* DC motor control using Arduino and H-Bridge

#### 2.2.2 Bluetooth

It is used for many applications like wireless headset, game controllers, wireless mouse, wireless keyboard, and many more consumer applications.

* It has range up to <100m which depends upon transmitter and receiver, atmosphere, geographic & urban conditions.
* It is IEEE 802.15.1 standardized protocol, through which one can build wireless Personal Area Network (PAN). It uses frequency-hopping spread spectrum (FHSS) radio technology to send data over air.
* It uses serial communication to communicate with devices. It communicates with microcontroller using serial port (USART).

**HC-05 Bluetooth Module:** HC-05 is a Bluetooth module which is designed for wireless communication. This module can be used in a master or slave configuration. The HC-05 Bluetooth module is shown in fig.2.3.

Bluetooth serial modules allow all serial enabled devices to communicate with each other using Bluetooth. It has 6 pins,

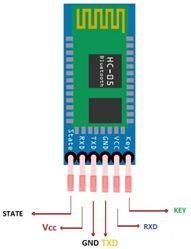
1. **Key/EN**: It is used to bring Bluetooth module in AT commands mode. If Key/EN pin is set to high, then this module will work in command mode. Otherwise by default it is in data mode.

The default baud rate of HC-05 in command mode is 38400bps and 9600 in data mode.

**HC-05 module has two modes**:

* + - * **Data mode**: Exchange of data between devices.
      * **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.

 Fig. 2.3 Bluetooth module pinout

1. **VCC**: Connect 5 V or 3.3 V to this Pin.
2. **GND:** Ground Pin of module.
3. **TXD:** Transmit Serial data (wirelessly received data by Bluetooth module transmitted out serially on TXD pin)
4. **RXD:** Receive data serially (received data will be transmitted wirelessly by Bluetooth module).
5. **State:** It tells whether module is connected or not.

**HC-05 module Information:**

* + - * HC-05 has red LED which indicates connection status, whether the Bluetooth is connected or not. Before connecting to HC-05 module this red LED blinks continuously in a periodic manner. When it gets connected to any other Bluetooth device, its blinking slows down to two seconds.
      * This module works on 3.3V. We can connect 5V supply voltage as well since the module has on board 5 to 3.3 V regulator.
      * As HC-05 Bluetooth module has 3.3V level for RX/TX and microcontroller can detect 3.3 V level, so, no need to shift transmit level of HC-05 module. But we need to shift the transmit voltage level from microcontroller to RX of HC-05 module.
      * The data transfer rate of HC-05 module can vary up to 1Mbps is in the range of 10 meters.

**Bluetooth communication between Devices:**

E.g. Send data from Smartphone terminal to HC-05 Bluetooth module and see this data on PC serial terminal and vice versa.

To communicate smartphone with HC-05 Bluetooth module, smartphone requires Bluetooth terminal application for transmitting and receiving data. You can find Bluetooth terminal applications for android and windows in respective app. Store.

So, when we want to communicate through smartphone with HC-05 Bluetooth module, connect this HC-05 module to the PC via serial to USB converter.

Before establishing communication between two Bluetooth devices, 1st we need to pair HC05 module to smartphone for communication. **Pair HC-05 and smartphone:**

* + - * + Search for new Bluetooth device from your phone. You will find Bluetooth device with “HC-05” name.
        + Click on connect/pair device option; default pin for HC-05 is 1234 or 0000.

After pairing two Bluetooth devices, open terminal software (e.g. Teraterm, Realterm etc.) in PC, and select the port where we have connected USB to serial module. Also select default baud rate of 9600 bps.

In smart phone, open Bluetooth terminal application and connect to paired device HC-05. It is simple to communicate, we just have to type in the Bluetooth terminal application of smartphone. Characters will get sent wirelessly to Bluetooth module HC-05. HC-05 will automatically transmit it serially to the PC, which will appear on terminal. Same way we can send data from PC to smartphone.

**Command Mode:**

* + - * + When we want to change settings of HC-05 Bluetooth module like change password for connection, baud rate, Bluetooth device’s name etc.
        + To do this, HC-05 has AT commands.
        + To use HC-05 Bluetooth module in AT command mode, connect “Key” pin to High (VCC).
        + Default Baud rate of HC-05 in command mode is 38400bps.
        + Table 1 shows some AT command generally used to change setting of Bluetooth module.
        + To send these commands, we have to connect HC-05 Bluetooth module to the PC via serial to USB converter and transmit these command through serial terminal of PC.

Table 1 Bluetooth module AT commands

|  |  |  |
| --- | --- | --- |
| **Command** | **Description** | **Response** |
| AT | Checking communication | OK |
| AT+PSWD=XXXX | Set Password  e.g :AT+PSWD=4567 | OK |
| AT+NAME=XXXX | Set Bluetooth Device Name e.g. T+NAME=MyHC-05 | OK |
| AT+UART=Baud rate,  Stop bit, parity bit | Change Baud rate  e.g. AT+UART=9600,1,0 | OK |
| AT+VERSION? | Respond version no. of  Bluetooth module | +Version: XX OK  e.g.+Version:2.0 2013107 OK |
| AT+ORGL | Send detail of setting done by manufacturer | Parameters: device type, module mode, serial parameter, passkey, etc. |

***2.2.3* Arduino’s L293D motor driver shield**

The Arduino L293D motor driver shield guide is a robobtics project that involves driving various types of motors. The most common types used for robotic applications include DC, servo, and stepper motors.

However, these motors typically cannot be driven directly by Arduino or another microcontroller. This is because of their higher current and power ratings, so motor shields or driver ICs are used instead. These shields or ICs isolate a motor’s power supply and use control logic from the microcontroller circuitry.

One of the most popular motor driver shields used with Arduino is the L293D. The fullfeatured L293D motor driver shield can control up to four bi-directional DC motors with 8bit speed selection, two stepper motors, and two servo motors.



**Specification of L293D Motor Driver :**

* Operating Voltage :5V to 12V.
* Motor controller: L293D, Drives 2 DC motors or 1 stepper motor.
* Max current: 600mA per channel.
* Peak Output Current :1.2 Amp.

**Applications of L293D Motor Driver :**

* It is used by Arduino users.
* Multiple DIY Projects.

#### 2.2.4 DC Motor

A **DC motor** is any of a class of rotary [electrical motors](https://en.wikipedia.org/wiki/Electrical_motor) that converts direct current (DC) electrical energy into mechanical energy. The most common types rely on the forces produced by induced magnetic fields due to flowing current in the coil. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current in part of the motor.

DC motors were the first form of motors widely used, as they could be powered from existing direct-current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances. The [universal motor,](https://en.wikipedia.org/wiki/Universal_motor) a lightweight [brushed](https://en.wikipedia.org/wiki/Brush_(electric)) motor used for portable power tools and appliances can operate on direct current and alternating current. Larger DC motors are currently used in propulsion of electric vehicles, elevator and hoists, and in drives for steel rolling mills. The advent of [power electronics](https://en.wikipedia.org/wiki/Power_electronics) has made replacement of DC motors with [AC motors](https://en.wikipedia.org/wiki/AC_motors) possible in many applications.



### 2.3 SOFTWARE DISCRIPTION

The software used here is ARDUINO SOFTWARE:

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

**Writing Sketches:**

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.

**NB:**

Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the ino extension on save.

 ***Verify***

Checks your code for errors compiling it.

 ***Upload***

Compiles your code and uploads it to the configured board. See [uploading](https://www.arduino.cc/en/Guide/Environment#uploading) below for details.

**Note:** If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"

 ***New***

Creates a new sketch.  ***Open***

Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

**Note:** due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead.  ***Save***

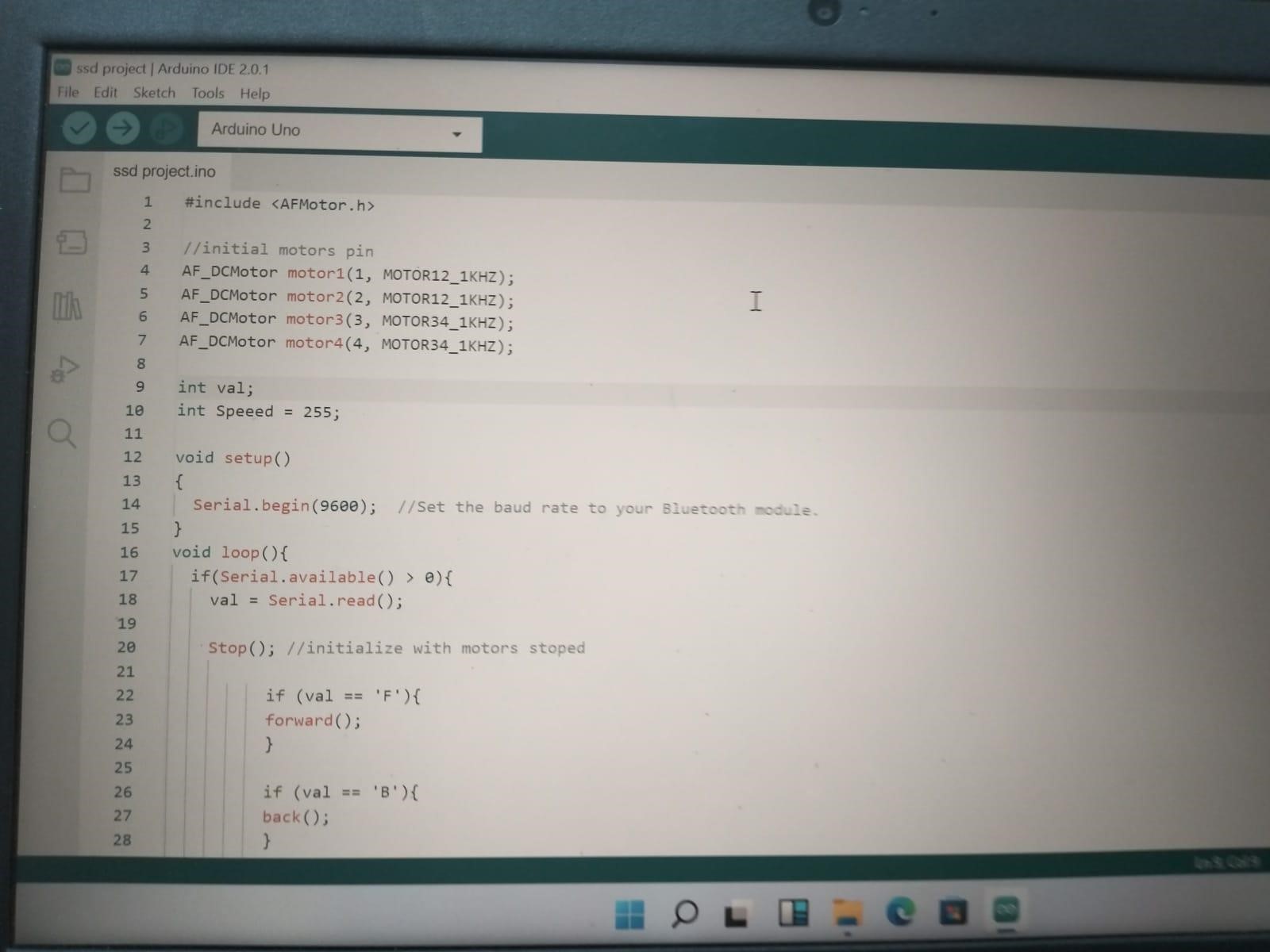
Saves your sketch.

 ***Serial Monitor***

Opens the [serial monitor.](https://www.arduino.cc/en/Guide/Environment#serialmonitor)

Additional commands are found within the five menus: File, Edit, Sketch, Tools,and help.

## Programming on arduinouno



In order for the Arduino-Uno board to be able to interact with the application used in this project certain program (code) needs to be uploaded to the Arduino-Uno.

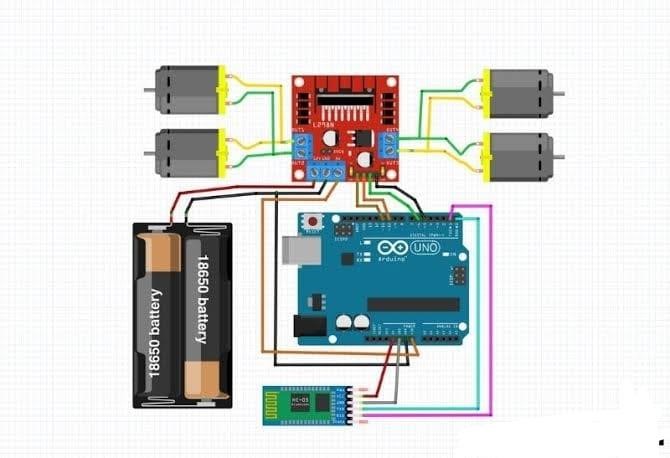
Arduino Company provides user friendly software which allows writing any code for any function wanted to be performed by the Arduino-Uno and upload it to the board.Refer to appendix A for the full source code of the Arduino-Uno board.

**CHAPTER 3**

**CIRCUIT DIAGRAM AND DESCRIPTION**

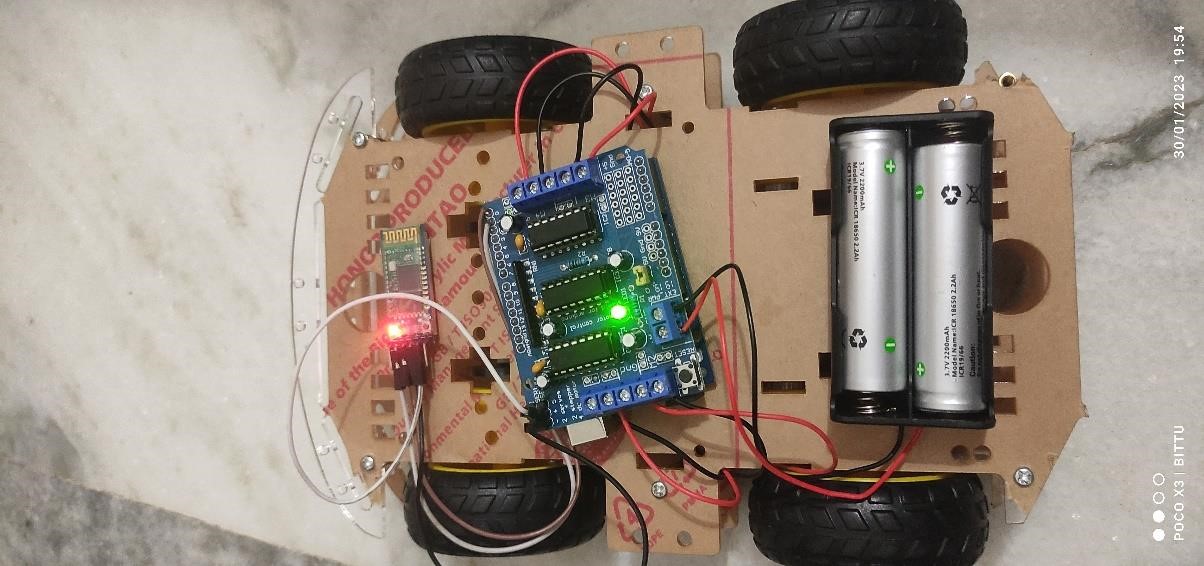
### 3.1 Working

We have used L293D Driver module and DC motors. DC motors move in clock wise and the anti-clock wise directions. The DC motors help the robot to move and the Driver module controls motor motion and power according to the Arduino Uno instructions. Arduino Uno follows instructions of HC-05 bluetooth module. It controls the robot according to instructions given by the instructor the Schematic diagram of working model.



### 3.2 RESULTS

The experimental result is as shown in below



Managed to successfully complete the project using Arduino Uno and HC-05 Bluetooth module, it was user friendly and cost effective. User friendly as in anyone can use just a click of a button on an android screen and everything works. And it is cost effective as in it will cost exactly as the project requires (optimum price).

# CHAPTER 4

**CONCLUSION**

## 4.1CONCLUSIO

Based on the instructions, information given by our SSD faculty we have prepared a robot module based on the theme robot war. This is an ideal system for real time application automation with Bluetooth access. An ideal system should be available from all over the world to a user and in military can be interfaced. The advantages we enhance from this project are less human efforts, decreasing the human death and it can go everywhere. The disadvantages we enhance from this project are reduction in the employment in the military section, environmental crisis.

# BIBLIOGRAPHY

1. <https://en.wikipedia.org/wiki/Arduino_Uno>
2. <https://circuitbest.com/category/arduino-projects/>
3. https://circuitbest.com/arduino-bluetooth-controlled-car-with-front-back-lights-using-arduinouno-l293d-motor-driver-hc-05/

# APPENDIX

1. /\*
2. Code Name: Arduino Bluetooth Car with Front and Back Light Control
3. Code URI:<https://circuitbest.com/category/arduino-projects/>
4. Before uploading the code you have to install the "Adafruit Motor Shield" library
5. Open Arduino IDE >> Go to sketch >> Include Libray >> Manage Librays... >> Search "Adafruit 6 AFMotor Library:<https://learn.adafruit.com/adafruit-motor-shield/library-install>7 Author: Make DIY
6. Author URI:<https://circuitbest.com/author/admin/>
7. Description: This program is used to control a robot using an app that communicates with Ard 10 Module.

11 App URI:<https://bit.ly/3mn6LuZ>12 Version: 1.0

13 License: Remixing or Changing this Thing is allowed. Commercial use is not allowed. 14 \*/

15

16 #include <AFMotor.h>

17

1. //initial motors pin
2. AF\_DCMotor motor1(1, MOTOR12\_1KHZ);
3. AF\_DCMotor motor2(2, MOTOR12\_1KHZ);
4. AF\_DCMotor motor3(3, MOTOR34\_1KHZ);
5. AF\_DCMotor motor4(4, MOTOR34\_1KHZ);

23

1. int val;
2. int Speeed = 255;

26

27 void setup() 28 {

1. Serial.begin(9600); //Set the baud rate to your Bluetooth module.
2. }
3. void loop(){
4. if(Serial.available() > 0){
5. val = Serial.read();

34

35 Stop(); //initialize with motors stoped

36

1. if (val == 'F'){
2. forward();
3. }

40

1. if (val == 'B'){
2. back();
3. }

44

1. if (val == 'L'){
2. left();
3. }

48

1. if (val == 'R'){
2. right();
3. }
4. if (val == 'I'){
5. topright();
6. }

55

1. if (val == 'J'){
2. topleft();
3. }

59

1. if (val == 'K'){
2. bottomright();
3. }

63

1. if (val == 'M'){
2. bottomleft();
3. }
4. if (val == 'T'){
5. Stop();
6. }
7. }
8. }

72

73

74

75

76

77

1. void forward()
2. {
3. motor1.setSpeed(Speeed); //Define maximum velocity
4. motor1.run(FORWARD); //rotate the motor clockwise
5. motor2.setSpeed(Speeed); //Define maximum velocity
6. motor2.run(FORWARD); //rotate the motor clockwise
7. motor3.setSpeed(Speeed);//Define maximum velocity
8. motor3.run(FORWARD); //rotate the motor clockwise
9. motor4.setSpeed(Speeed);//Define maximum velocity
10. motor4.run(FORWARD); //rotate the motor clockwise
11. }

89

1. void back()
2. {
3. motor1.setSpeed(Speeed); //Define maximum velocity 93 motor1.run(BACKWARD); //rotate the motor anti-clockwise
4. motor2.setSpeed(Speeed); //Define maximum velocity
5. motor2.run(BACKWARD); //rotate the motor anti-clockwise
6. motor3.setSpeed(Speeed); //Define maximum velocity
7. motor3.run(BACKWARD); //rotate the motor anti-clockwise
8. motor4.setSpeed(Speeed); //Define maximum velocity
9. motor4.run(BACKWARD); //rotate the motor anti-clockwise
10. }

101

1. void left()
2. {
3. motor1.setSpeed(Speeed); //Define maximum velocity
4. motor1.run(BACKWARD); //rotate the motor anti-clockwise
5. motor2.setSpeed(Speeed); //Define maximum velocity
6. motor2.run(BACKWARD); //rotate the motor anti-clockwise
7. motor3.setSpeed(Speeed); //Define maximum velocity
8. motor3.run(FORWARD); //rotate the motor clockwise
9. motor4.setSpeed(Speeed); //Define maximum velocity
10. motor4.run(FORWARD); //rotate the motor clockwise
11. }

113

1. void right()
2. {
3. motor1.setSpeed(Speeed); //Define maximum velocity
4. motor1.run(FORWARD); //rotate the motor clockwise
5. motor2.setSpeed(Speeed); //Define maximum velocity 119 motor2.run(FORWARD); //rotate the motor clockwise
6. motor3.setSpeed(Speeed); //Define maximum velocity
7. motor3.run(BACKWARD); //rotate the motor anti-clockwise
8. motor4.setSpeed(Speeed); //Define maximum velocity
9. motor4.run(BACKWARD); //rotate the motor anti-clockwise
10. }

125

1. void topleft(){
2. motor1.setSpeed(Speeed); //Define maximum velocity 128 motor1.run(FORWARD); //rotate the motor clockwise
3. motor2.setSpeed(Speeed); //Define maximum velocity
4. motor2.run(FORWARD); //rotate the motor clockwise
5. motor3.setSpeed(Speeed/3.1);//Define maximum velocity
6. motor3.run(FORWARD); //rotate the motor clockwise
7. motor4.setSpeed(Speeed/3.1);//Define maximum velocity
8. motor4.run(FORWARD); //rotate the motor clockwise
9. }

136

1. void topright()
2. {
3. motor1.setSpeed(Speeed/3.1); //Define maximum velocity
4. motor1.run(FORWARD); //rotate the motor clockwise
5. motor2.setSpeed(Speeed/3.1); //Define maximum velocity 142 motor2.run(FORWARD); //rotate the motor clockwise
6. motor3.setSpeed(Speeed);//Define maximum velocity
7. motor3.run(FORWARD); //rotate the motor clockwise
8. motor4.setSpeed(Speeed);//Define maximum velocity
9. motor4.run(FORWARD); //rotate the motor clockwise
10. }

148

1. void bottomleft()
2. {
3. motor1.setSpeed(Speeed); //Define maximum velocity
4. motor1.run(BACKWARD); //rotate the motor anti-clockwise 153 motor2.setSpeed(Speeed); //Define maximum velocity
5. motor2.run(BACKWARD); //rotate the motor anti-clockwise
6. motor3.setSpeed(Speeed/3.1); //Define maximum velocity
7. motor3.run(BACKWARD); //rotate the motor anti-clockwise
8. motor4.setSpeed(Speeed/3.1); //Define maximum velocity
9. motor4.run(BACKWARD); //rotate the motor anti-clockwise 159 }

160

1. void bottomright()
2. {
3. motor1.setSpeed(Speeed/3.1); //Define maximum velocity
4. motor1.run(BACKWARD); //rotate the motor anti-clockwise
5. motor2.setSpeed(Speeed/3.1); //Define maximum velocity
6. motor2.run(BACKWARD); //rotate the motor anti-clockwise
7. motor3.setSpeed(Speeed); //Define maximum velocity
8. motor3.run(BACKWARD); //rotate the motor anti-clockwise
9. motor4.setSpeed(Speeed); //Define maximum velocity
10. motor4.run(BACKWARD); //rotate the motor anti-clockwise 171 }

172

173

1. void Stop()
2. {
3. motor1.setSpeed(0); //Define minimum velocity
4. motor1.run(RELEASE); //stop the motor when release the button
5. motor2.setSpeed(0); //Define minimum velocity
6. motor2.run(RELEASE); //rotate the motor clockwise
7. motor3.setSpeed(0); //Define minimum velocity
8. motor3.run(RELEASE); //stop the motor when release the button 182 motor4.setSpeed(0); //Define minimum velocity

183 motor4.run(RELEASE); //stop the motor when release the button }