Project Report

on

Arduino Bluetooth Control Car

## PRACTICAL FILE

submitted to the Faculty of Engineering of the Department of Electronics and Communication Engineering, Punjabi University, Patiala

**BACHELOR OF TECHNOLOGY IN**

**ELECTRONICS AND COMMUNICATION**

****

## Department of Electronics and Communication Engineering, Punjabi University, Patiala-147001, Punjab (India)

(May 2025)

**Submitted by**

### 

Name: Sujal Gaba

Roll No: 12202042

**Submitted to**

**Dr.Amandeep kaur dhaliwal**  Professor Department of Electronics and Communication Engineering Punjabi University, Patiala

**Dr. ………………………….**

Professor Department of Electronics and Communication Engineering Punjabi University, Patiala.

Index

1. Introduction………………………………………………………………………………………..
   1. Basics about project……………………………………………………………………….
2. Basic components………………………………………………………………………………..
   1. HC 05 blue­­-tooth module……………………………………………………………………..
   2. Relay module………………………………………………………………………………………..
   3. Arduino…………………………………………………………………………………………………..
   4. AC bulb and holder………………………………………………………………………..
3. Block diagram………………………………………………………………………………………….
4. Connection diagram………………………………………………………………………………..
5. Coding to operate the system………………………………………………………………
   * 1. Library………………………………………………………………………………..
     2. Commands………………………………………………………………………..
     3. Functions……………………………………………………………………………
     4. Variables…………………………………………………………………………………..
6. Applications……………………………………………………………………………………………
7. Sources…………………………………………………………………………………………….

**ACKNOWLEDGMENT**

**“***Success is a sweet fruit to which everyone strives to taste*.**”**

Any job in this world, however trivial or tough cannot be accomplished without the assistance of others. I would hereby take the opportunity to express my indebtedness to people who have helped me to accomplish this task. I feel a deep sense of gratitude in thanking all those who helped me to carry this project to its eventual fruition.

This project was quite a learning experience for me at each and every step. At the same time, it has given me the confidence to work in the professional set-up. I feel the experience gained during the project development would lead me to bright prospect in future.

I take this opportunity with much pleasure to thank all the people who have helped me through the course and producing this synopsis. I sincerely thank my teacher, Dr.Amandeep kaur for his guidance, help and motivation. Apart from the subject of my course, I learnt a lot from him, which I am sure, will be useful in different stages of my life.

I am grateful to HOD, Electronics and Communication, Punjabi University in pursuing this training in a smooth and organized manner.

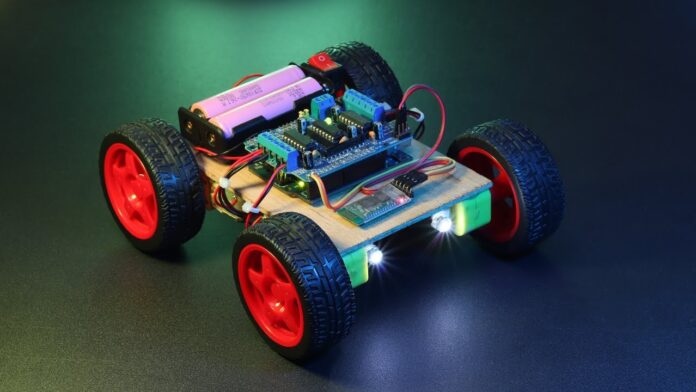
I would also like to express my gratitude to the all other members of my report advisory committee

Last but not the least, I would like to thank my family and the All Mighty God.

Thanks

1. Introduction

**Basics of project :- Arduino Bluetooth controlled car with Front & Back Lights using Arduino UNO, L293D Motor Driver, HC-05**



## What is an Arduino Bluetooth controlled car with Front & Back Lights?

Bluetooth car is mainly a simple RC Car that is mainly controlled over Bluetooth Protocol. An Arduino Bluetooth Module (HC-05 or HC-06) is necessary for controlling the car over Bluetooth signal through Android Device.

## How do you control a Bluetooth car?

A Bluetooth-controlled car is a simple car that runs on a wireless Bluetooth signal. In Arduino, we have mainly 2 Bluetooth Options HC-05 and HC-06. In this article, we are mainly focusing on HC-05 Bluetooth Module. The HC-05 has many features compared to HC-06 Like we can use HC-05 as a Master and Slave device. And we can Send, Receive and Display Data with this Bluetooth module.

## How do I control my RC car with Bluetooth?

There are many ways for controlling the Bluetooth car. We can use Gamepad, Custom Made controller with HC-05, and much more. But for making things simple I am using an Android Phone couple with an android app for controlling the car.

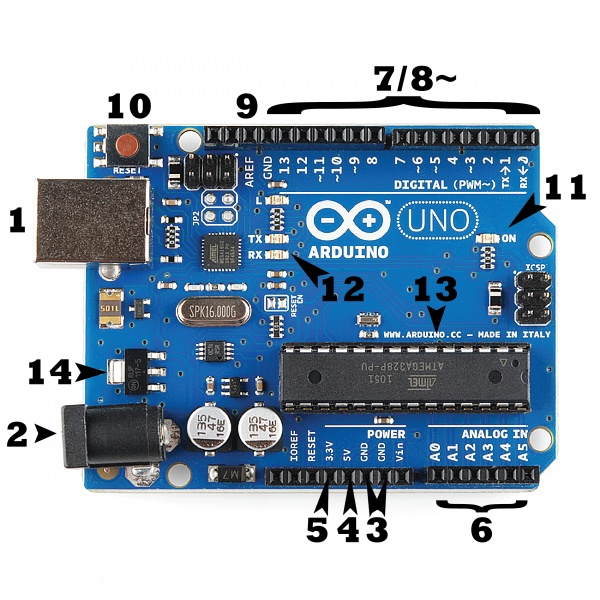
Basic components

### **Arduino­:-**

[Arduino](http://arduino.cc/) is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a [microcontroller](http://en.wikipedia.org/wiki/Microcontroller)) and a piece of [software](http://arduino.cc/en/Main/Software), or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

There are many varieties of Arduino boards that can be used for different purposes. Some boards look a bit different from the one below, but most Arduinos have the majority of these components in common:

[](https://cdn.sparkfun.com/assets/b/f/e/9/c/513824face395f6d3d000000.png)

### Power (USB / Barrel Jack)

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply that is terminated in a barrel jack. In the picture above the USB connection  and the barrel jack.

The USB connection is also how you will load code onto your Arduino board.

 Do NOT use a power supply greater than 20 Volts as you will overpower (and thereby destroy) your Arduino. The recommended voltage for most Arduino models is between 6 and 12 Volts.

### Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)

The pins on your Arduino are the places where you connect wires to construct a circuit (probably in conjuction with a [breadboard](https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard/) and some [wire](https://learn.sparkfun.com/tutorials/working-with-wire). They usually have black plastic ‘headers’ that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

* **GND (3)**: Short for ‘Ground’. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
* **5V (4) & 3.3V (5)**: As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
* **Analog (6)**: The area of pins under the ‘Analog In’ label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a [temperature sensor](https://www.sparkfun.com/products/10988)) and convert it into a digital value that we can read.
* **Digital (7)**: Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
* **PWM (8)**: You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have [a tutorial on PWM](https://learn.sparkfun.com/tutorials/pulse-width-modulation), but for now, think of these pins as being able to simulate analog output (like fading an LED in and out).
* **AREF (9)**: Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

### **Reset Button**

Just like the original Nintendo, the Arduino has a reset button **(10)**. Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn’t repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

### **Power LED Indicator**

Just beneath and to the right of the word “UNO” on your circuit board, there’s a tiny LED next to the word ‘ON’ **(11)**. This LED should light up whenever you plug your Arduino into a power source. If this light doesn’t turn on, there’s a good chance something is wrong. Time to re-check your circuit!

### **TX RX LEDs**

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for [serial communication](https://learn.sparkfun.com/tutorials/serial-communication). In our case, there are two places on the Arduino UNO where TX and RX appear -- once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs **(12)**. These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we’re loading a new program onto the board).

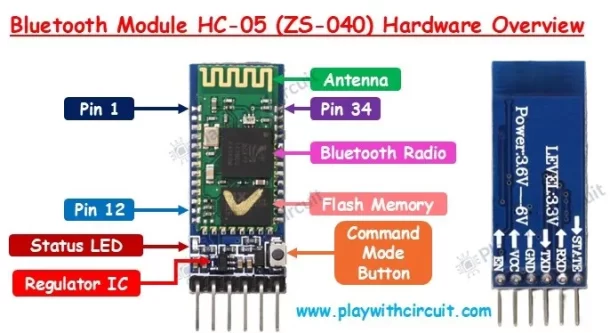
### **Main IC**

The black thing with all the metal legs is an IC, or Integrated Circuit **(13)**. Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC’s from the ATMEL company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC's, reading the datasheets is often a good idea.

### **Voltage Regulator**

The voltage regulator **(14)** is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it’s for. The voltage regulator does exactly what it says -- it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don’t hook up your Arduino to anything greater than 20 volts.

1. **Bluetooth module (HC-05):**

****

The HC-05 is a class 2 Bluetooth module designed for transparent wireless serial communication. It is pre-configured as a slave Bluetooth device. Once it is paired to a master Bluetooth device such as PC, smart phones and tablet, its operation becomes transparent to the user. All data received through the serial input is immediately transmitted over the air. When the module receives wireless data, it is sent out through the serial interface exactly at it is received. No user code specific to the Bluetooth module is needed at all in the user microcontroller program.

The HC-05 supports two work modes: Command and Data mode. The work mode of the HC-05 can be switched by the onboard push button. The HC-05 is put in Command mode if the push button is activated. In Command mode, user can change the system parameters (e.g. pin code, baud rate, etc) using host controller itself of a PC running terminal software using a serial to TTL converter. Any changes made to system parameters will be retained even after power is removed. Power cycle the HC-05 will set it back to Data Mode. Transparent UART data transfer with a connected remote device occurs only while in Data Mode.

The HC-05 can be re-configured by the user to work as a master Bluetooth device using a set of AT commands. Once configured as master, it can automatically pair with a HC-05 in its default slave configuaration or a HC-06 module, allowing an point to point serial communications.

The HC-05 will work with supply voltage of 3.6VDC to 6VDC, however, the logic level of RXD pin is 3.3V and is not 5V tolerant. A [Logic Level Converter](https://www.sgbotic.com/index.php?dispatch=products.view&product_id=3054) is recommended to protect the sensor if connect it to a 5V device (e.g Arduino Uno and Mega). The power to the HC-05 will cut off if the "EN" pin is pulled to logic 0.

**Features**:

* Bluetooth v2.0+EDR
* 2.4GHz ISM band frequency
* Supported baud rate: 9600 (default), 19200,38400, 57600, 115200, 230400, 460800.
* Power supply: 3.6V to 6V DC
* Passkey: 1234

### **(C) What is a TT Motor?**

A **TT motor** is a compact, low-cost **DC gear motor** commonly used in **robotics and DIY electronics projects**, especially with platforms like **Arduino** or **Raspberry Pi**. It typically comes with a **plastic gearbox** to reduce the motor's speed and increase torque.

Here are some **typical specs** and **features** of a TT motor gear set:

* **Voltage:** 3V–6V DC (often 3.7V or 5V is used)
* **No-load speed (at 6V):** ~200–250 RPM (depends on gear ratio)
* **Gear Ratio:** Often around 1:48 or 1:90
* **Torque:** Low to medium (suitable for light loads)
* **Output Shaft:** Often D-shaped for better grip with wheels
* **Mounting holes:** Easy to attach to chassis or frames
* **Direction Control:** Reversible by changing polarity
* **Typical Current:** ~100–250mA (higher under load)

### **What Comes in a Gear Set?**

A **TT Motor gear set** might include:

* 1 or 4 TT gear motors
* Plastic gears (inside or visible depending on type)
* Wheels (rubber or plastic)
* Motor brackets or mounts
* Screws and axles
* Optionally: encoder disks or wires

### **Applications**

* Mini **robots** (line followers, obstacle avoiders)
* Motorized **cars** and **buggies**
* **Conveyor belts** or small **mechanical arms**
* Educational STEM kits
* **Automated doors**, **pan/tilt systems**, etc.

### **Tips for Using in a Project**

* Use a **motor driver** like **L298N** or **L293D** to control speed/direction.
* Pair with a **wheel encoder** if you need feedback for speed or position.
* Use **battery packs** (3.7V Li-ion or 4xAA) for portable power.
* Use **plastic chassis kits** for easy mounting.



## **Overview**

In this project overview, we introduce the Arduino Bluetooth car, a mobile-controlled robotic vehicle created with Arduino technology. The Android app serves as the primary interface for controlling the car through Bluetooth communication. Resembling an RC car, it offers functionalities such as horn activation, front and back lights, and speed adjustment. The wireless control extends up to a maximum range of 15 m -20 m, providing flexibility and convenience in maneuvering the vehicle.

## **What you need for this project**

1. Arduino Uno or compatible board
2. 4-wheel chassis kit with DC motors
3. L298N motor driver module
4. HC-05 or HC-06 Bluetooth Module
5. Jumper wires
6. Two 18650 Rechargeable Battery 3.7V
7. 18650 Li-ion Battery Holder 2-Way
8. Toggle Switch
9. USB cable for programming the Arduino
10. Screwdriver and assorted screws
11. Glue Gun
12. Front and back lights (LEDs)

**Steps to assemble bluetooth car**

Firstly, we will need Motors. for making. the car. At a minimum, you can use 2 motors for the left and right sides, But in this Project, we are using 4 DC Gear motors. More specifically, Here we are using 300RPM 9v TT Gear Motor for making Arduino Bluetooth car **l293d.**

I  am using a **Small Plywood Board for making the chassis**. For attaching the motor with the chassis we are using a Hot Glue gun. In this same process we. have to connect all the 4 Motors. at the 4 Sides

After connecting all 4 Motors the Plywood Chassis will look like this. Try to maintain the Orientation and equal distance as I have shown in the Picture. I will also provide you **Bluetooth controlled car using Arduino project report**.

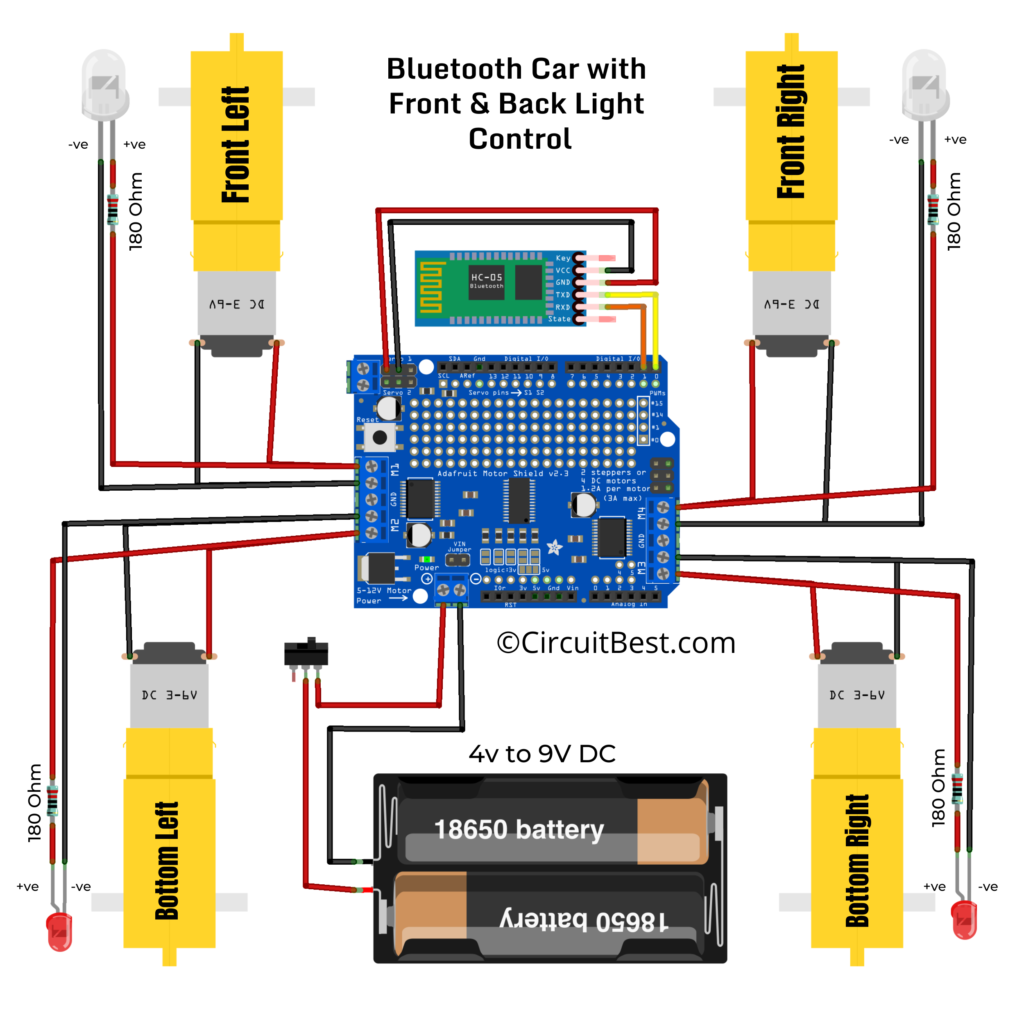
#### ****Note:****

* In this project, I am using TT Gear Motor. These Motors are cheap and also good for small projects. But If you have some different motors with the same voltage (6v to 12v) & current (200mA to 1000mA) range, Then you can also use that motors as well.
* For the Chassis, I have used a small piece of Plywood. That works great for my work. You can also use Acrylic Sheet, Fiber sheet, and any other things for the **Bluetooth controlled car using Arduino and l293d**.
* If you don’t have Glue Gun then you can also use any other strong Glue. But I would suggest you use a glue gun with a stick. This will work great for **Arduino Bluetooth car L293D motor shield**.

### **Step 2:**

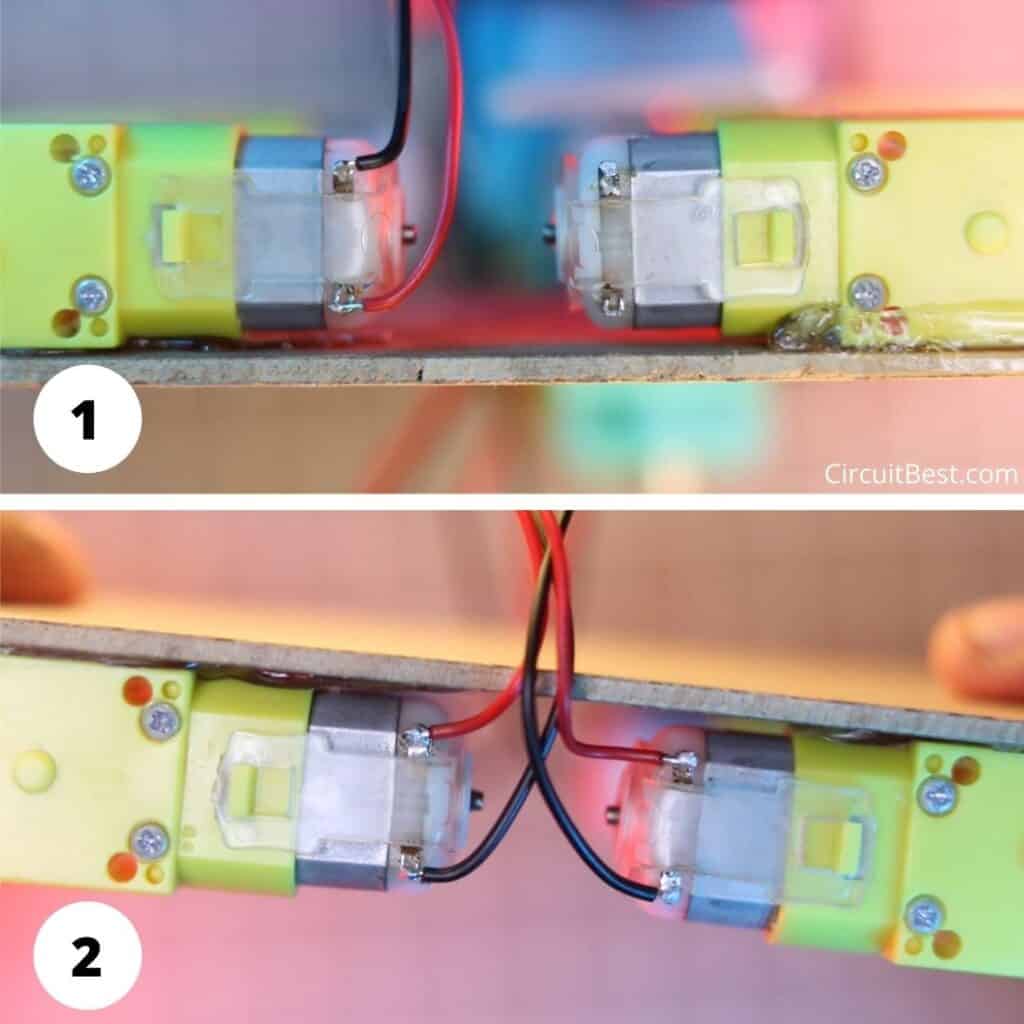
## **Wiring the components in a circuit diagram**

The Arduino Bluetooth RC Car Circuit Diagram is not complicated. First, you download the circuit diagram from the download button. It’s more obvious. Small jumpers should be connected to ENA and ENB on the L298N motor driver you are using. It is connected by default when you buy it. Use two or three 3.7V 18659 li-ion batteries to provide power.**It cannot use a**9V battery. If a 9V battery is used, only the front light, rear light, and horn will work. The gear motor is not working. The reason for that is that the 9V battery does not have enough amperage. **Front light, rear light, and horn are optional things of Arduino Bluetooth RC Car**. Arduino Bluetooth RC car can be made without front light, back light and horn. No need to modify Arduino code.



#### Motor Wire Connection Notes:

* Make sure you are using Good Soldering Iron for Soldering.
* Use High-Quality Solder Wire for Good solder joints.
* For any loose wiring (Dry Solder) **Arduino Bluetooth remote control car** may not work properly.



### **Step 3:**

For making  bluetooth car with speed control we are using Arduino UNO. This board is the main Heart of the Project

#### Which Arduino You Should use?

* Here I am using the original **Arduino UNO**. So, it will not need any type of modification.
* Some Arduino boards come with SMD Version of Atmega328P IC. Those Clone boards will also work just fine for Arduino Bluetooth controlled car with Front & Back Lights.
* You can also Use any Chinese clone of Arduino UNO. Some of that boards have CH340 IC as the bootloader. For any windows computer, the driver of CH340 is not pre-installed (MAC users don’t need any modification)on your device. So you will have to install the necessary drivers to upload code to Arduino.



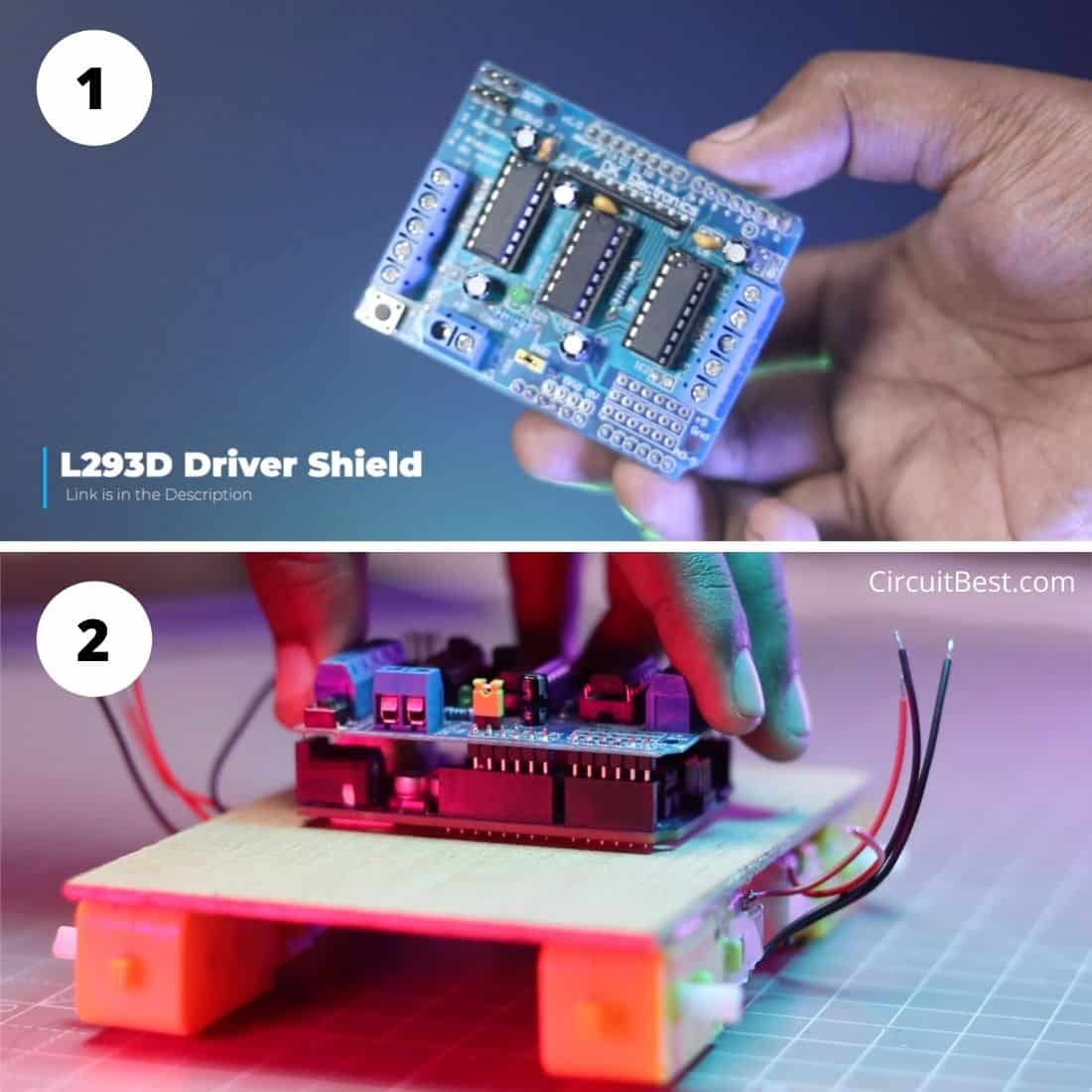
Now we have to attach the Arduino UNO to the Chassis. So, I am using Double-sided tape for this. You can also Drill holes on the Chassis and attach the Arduino Board with M3 Screws.

Now I have connected the Arduino with the chassis and this will look as shown in the picture and make the Arduino Bluetooth controlled car with Front & Back Lights.

### **Step 4:**

For the bluetooth controlled car using Arduino project report we will need a motor driver to run the 4 motors. So, we are using L293D motor Driver.The main purpose of using the L293D Motor driver is that it is really easy to use and implement in the code. You will just need the Adafruit Motor Shield Library and a few lines of code for controlling the car.

Now Attach the L293D driver on the Arduino UNO for **Arduino Bluetooth car with speed control**.



#### Motor Driver Connection Notes:

* As this is a Shield. So, It fits on the Arduino Uno or Arduino Mega.
* No, Need to connect extra wires to power the Arduino. If you give power to the Motor shield then that power from the shield goes directly to the Arduino UNO.
* This L293D driver has 4 Outputs so you can control 4 motors individually.
* This L293D Motor Driver’s max current per channel is 1A.

### **Step 5:**

For the Back Lights, I am using 10MM Red LEDs. This is really great for this type of DIY Project. Now we have to connect the LED with the car.

**But we can’t Directly attach the LED to the car. Because LED needs a certain voltage to run. Now If you provide and higher voltage than its tolerance then the LED will be fried.**

#### ****Resistor Selection:****

The Higher the Resistor value the Lower LED Brightness. My recommendation will be the highlighted options for making an Arduino Bluetooth controlled car with Front & Back Lights.

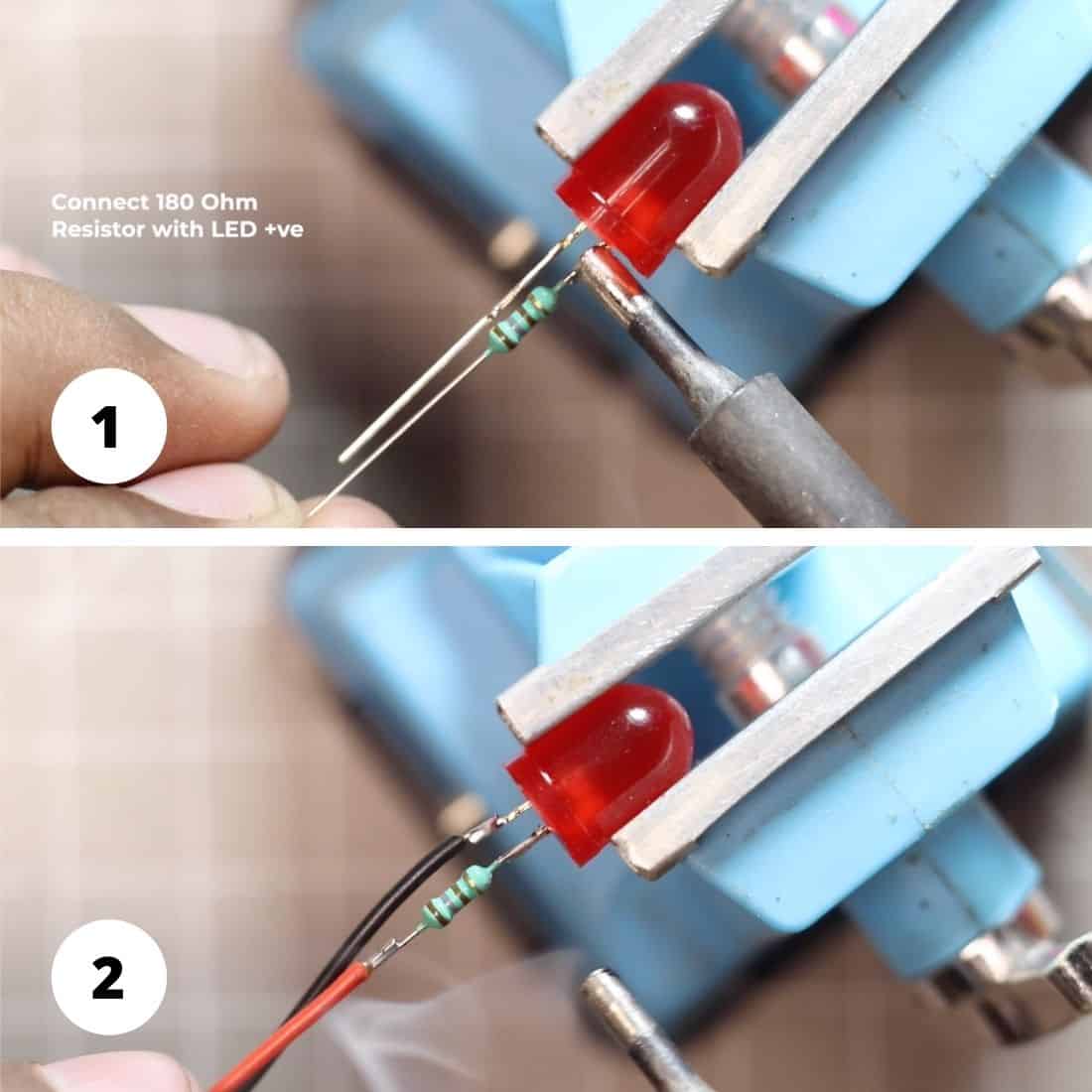
The 8.20V lies between 9V and 5V So, here I chosen the **180 Ohm**Resistor for the **Current Limitation**. This will save the Light frying from over voltage and give an optimal brightness.

Here I have connected the LED +ve with the 180 Ohm Resistor and the LED -ve is connected with the Black Supply wire. and the Red wire is connected with the 180 Ohm resistor Out. So, we will give power to the Red and the Black wire.

In the same process, I would prepare the front light also. Here I also connected the 180 Ohm resistor with the LED +ve. And the Red and Black wire is connected with the Resistor out and the LED -ve respectively.

#### LED Selection Note:

* Here I used 8MM White LED and 10MM Red LED. You can also use general 5MM, 3MM LEDs.
* The resistor values will remain the same for 5MM, 3MM LEDs.
* Here you can use a 1/4W Resistor for the given values.
* Make sure you have connected the Red and Black wire correctly otherwise the LEDs will not run properly.



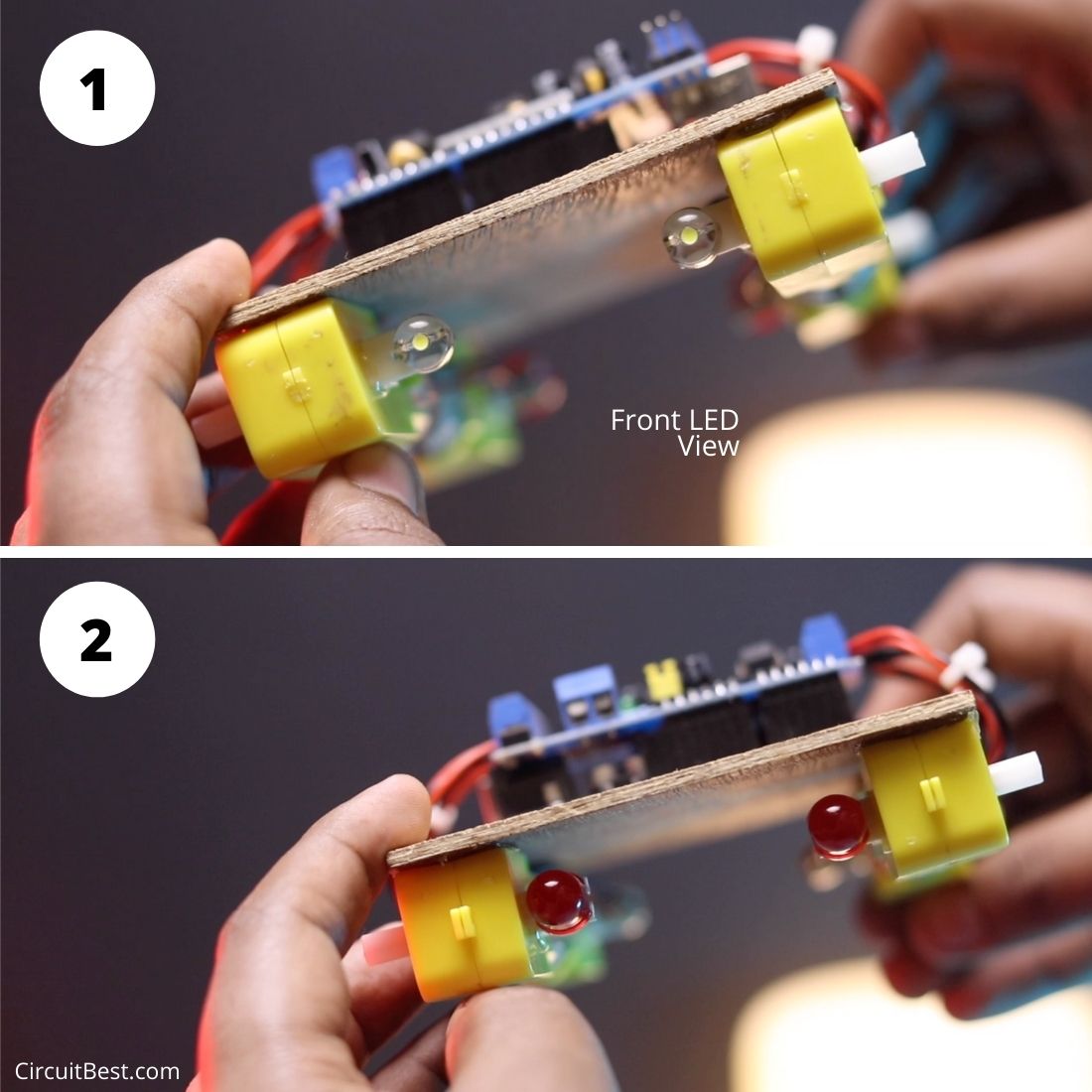
### **Step 6:**

### https://circuitbest.com/wp-content/uploads/2021/06/3D-Printed-LED-Bracket-Install.jpg

#### Now We have to connect the LEDs with the changes and I will attach this on the Chassis with a glue gun so that It can act as a Ligh

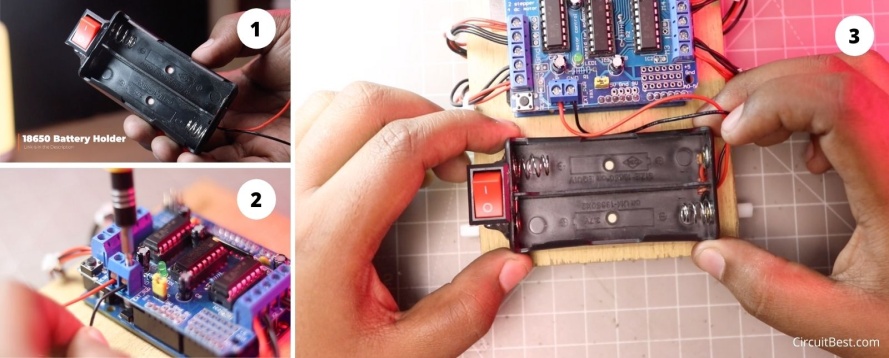
#### Extras:

* Make sure you are using a very small amount of glue for attaching the Holder with the **Bluetooth-controlled Car**.
* If you are using Instant Glue then attach it carefully, Otherwise, the Car shaft may be blocked with the glue.
* Here I used Anet ET4 Pro 3D Printer for Printing the Holder.
* First, connect the LED wires with the L293D Motor Driver shield for making a remote control car with Arduino. Just follow the all necessary connections as shown here.
* Now I have placed all the LEDs in their respective place. 8MM White LEDs at the front of the car chassis and I also Placed 10MM Red LEDs at the back of the car chassis. You can follow the same process as shown below for creating an Arduino Bluetooth control car with Front & Back Lights.



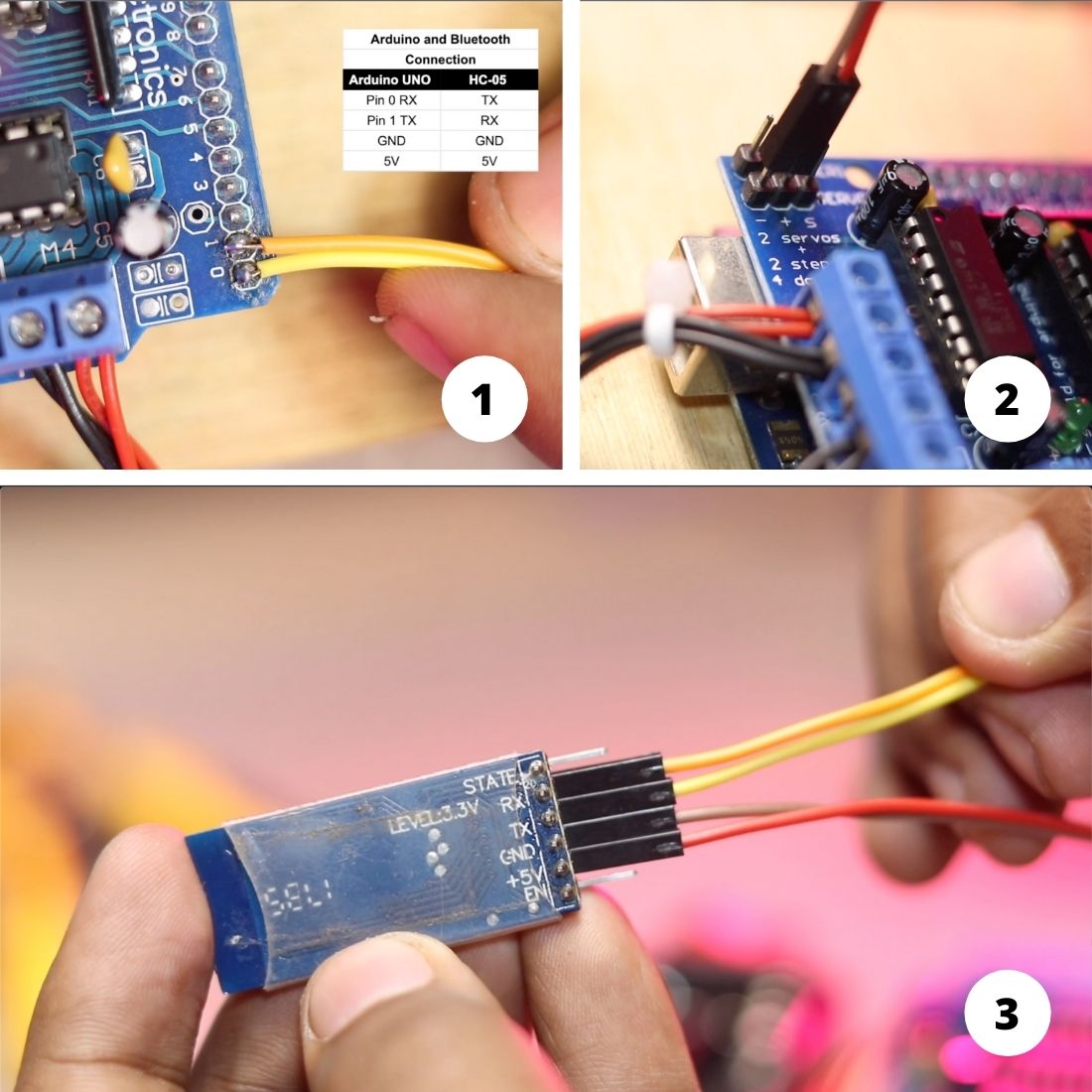
### **Step 8:**

The next part is the battery connection part. Here I am using 2S Li-ion Battery for running the car. the L293D supports 12V as well. So, if you want higher speed then you can also use 3s Li-ion Battery as well.

Here I have also added a switch for turning On/ Off the car. The connection of the switch is really simple. We have to connect the switch in series with the battery +ve. After that, the Switch Out (Battery +ve) and the Black Wire (Battery -ve) will be connected with the Motor Shield Power Input. 

### **Step 9:**

Now to make an Arduino Bluetooth car control app we will need a wireless receiver. For the wireless signal transmission to the car here we are using the HC-05 Bluetooth Module. This is a commonly used module for making different types of wireless signal transmission projects.



HC-05 has 6 Pins But we would need RX, TX, GND, +5V pins. Here the GND and the +5v pin is for power. and the RX and the TX pin is for signal Transfer. Here I used Female Female Jumper for giving power.

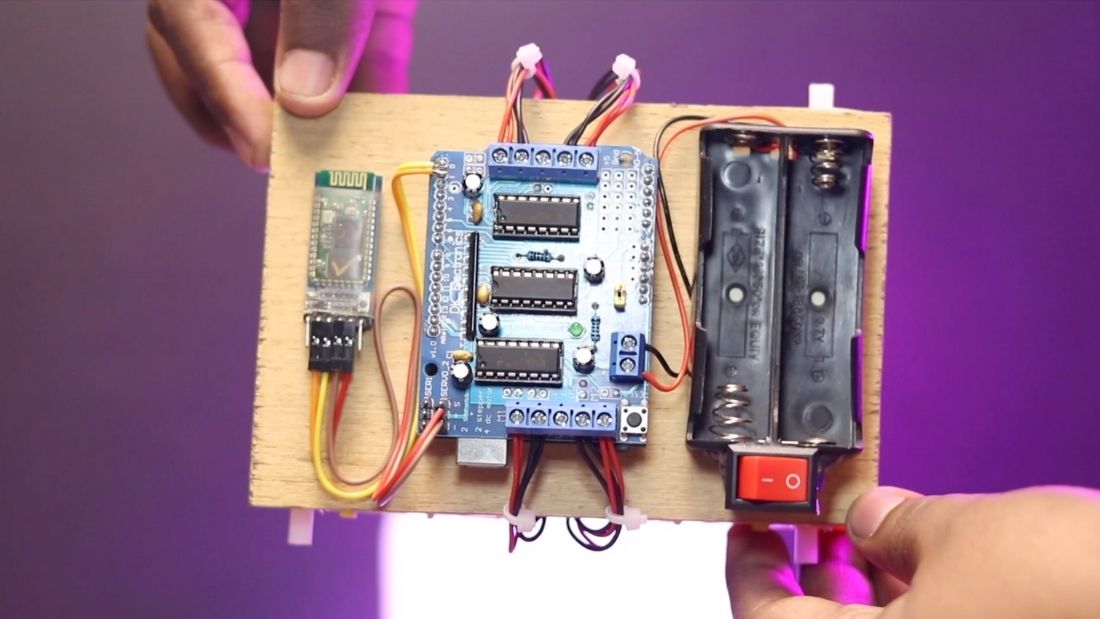
For the data Transfer I have soldered the RX and the Tx wires with the 1 & 2 Pin respectively.

.

#### Arduino and HC-05 Bluetooth Module Connection:

Here is the simple pinout connections for connecting with Arduino Uno with the HC-05 Bluetooth Module. Connect it accordingly.

|  |  |
| --- | --- |
| **Arduino Uno** | **HC-05 Module** |
| Pin 0 Rx | Tx |
| Pin 1 Tx | Rx |
| GND | GND |
| +5V | +5V |



#### Some Notes about HC-05 Bluetooth Module:

* **Some people suggest giving 3.3V power to the HC-05 Bluetooth Mode** for making the Arduino Bluetooth control car with L293D. Their main logic is that the HC-05 module runs with +3.3V. If you connect the HC-05 Bluetooth module with 5V it may get damaged. For these people, I would tell that the HC-05 Module has an inbuilt voltage regulator for 5V to 3.3V. So, **You don’t need to worry if you are powering the HC-05 Module with 5V Input from Arduino**.
* While soldering the RX and the TX jumpers with the L293D make sure you don’t short the pins of the shield. Otherwise, Arduino may be damaged. So, try to do a clean job in the soldering task.

### **Step 10:**

Now We have to upload the code to the Arduino UNO. So, we will connect the Arduino UNO’s USB poet with the PC COM Port. First. Now If you are using Original Arduino that is labeled as “Made in Italy” then you will not need any modification. **If you are using a Chinese clone with SMD Chip mounted on the Arduino then most likely you have to install the CH340 driver**. Here we are using the Arafruit Motor shield library for running the Bluetooth-controlled Car. So you need to go to the following and then search for Adafruit Motor Shield Library.

**Sketch > Include Library > Manage Libraries Option > Search “adafruit motor shield” > Install it**.

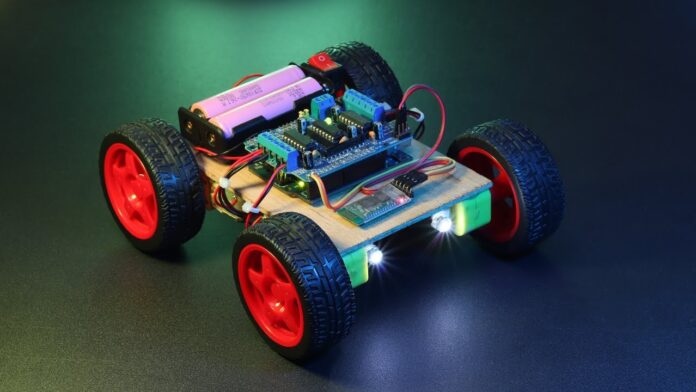
#### Some Key Factors while Uploading the code to Arduino:

* Make Sure you are installing Exactly the same Library as shown in the project. If you install any other library with this code then the chances are your car will not work.
* Before Uploading the code **you must disconnect Bluetooth module HC05 power**. Otherwise, your code will not be uploaded to Arduino.
* While uploading the code make sure that you are selecting the right COM port and the Board as Arduino UNO.
* While uploading the code it is recommended not to use any external power. Let me tell you why… If you have connected the Arduino in the wrong way Suppose you have connected the Battery in the wrong polarity. Then not only your Arduino Uno will be damaged also your PC COM port may be damaged.

#### Arduino Bluetooth Control Car Code:

Here is the simple Bluetooth control car code that you can use for your school or college project. For the lighting, you don’t need to change any parameters in the code. You Just have to connect all the LEDs in the right way as shown in the video and you are good to go for using Arduino Bluetooth controlled car with Front & Back Lights.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183 | /\*  Code Name: Arduino Bluetooth Car with Front and Back Light Control  Code URI: <https://circuitbest.com/category/arduino-projects/>  Before uploading the code you have to install the "Adafruit Motor Shield" library  Open Arduino IDE >> Go to sketch >> Include Libray >> Manage Librays...  >> Search "Adafruit Motor Shield" >> Install the Library  AFMotor Library: <https://learn.adafruit.com/adafruit-motor-shield/library-install>  Description: This program is used to control a robot using an app that communicates with Arduino through an HC-05 Bluetooth Module.  App URI: <https://bit.ly/3mn6LuZ>  Version: 1.0  License: Remixing or Changing this Thing is allowed. Commercial use is not allowed.  \*/    #include <AFMotor.h>    //initial motors pin  AF\_DCMotor motor1(1, MOTOR12\_1KHZ);  AF\_DCMotor motor2(2, MOTOR12\_1KHZ);  AF\_DCMotor motor3(3, MOTOR34\_1KHZ);  AF\_DCMotor motor4(4, MOTOR34\_1KHZ);    int val;  int Speeed = 255;    void setup()  {    Serial.begin(9600);  //Set the baud rate to your Bluetooth module.  }  void loop(){    if(Serial.available() > 0){      val = Serial.read();        Stop(); //initialize with motors stoped              if (val == 'F'){            forward();            }              if (val == 'B'){            back();            }              if (val == 'L'){            left();            }              if (val == 'R'){            right();            }            if (val == 'I'){            topright();            }              if (val == 'J'){            topleft();            }              if (val == 'K'){            bottomright();            }              if (val == 'M'){            bottomleft();            }            if (val == 'T'){            Stop();            }    }  }              void forward()  {    motor1.setSpeed(Speeed); //Define maximum velocity    motor1.run(FORWARD); //rotate the motor clockwise    motor2.setSpeed(Speeed); //Define maximum velocity    motor2.run(FORWARD); //rotate the motor clockwise    motor3.setSpeed(Speeed);//Define maximum velocity    motor3.run(FORWARD); //rotate the motor clockwise    motor4.setSpeed(Speeed);//Define maximum velocity    motor4.run(FORWARD); //rotate the motor clockwise  }    void back()  {    motor1.setSpeed(Speeed); //Define maximum velocity    motor1.run(BACKWARD); //rotate the motor anti-clockwise    motor2.setSpeed(Speeed); //Define maximum velocity    motor2.run(BACKWARD); //rotate the motor anti-clockwise    motor3.setSpeed(Speeed); //Define maximum velocity    motor3.run(BACKWARD); //rotate the motor anti-clockwise    motor4.setSpeed(Speeed); //Define maximum velocity    motor4.run(BACKWARD); //rotate the motor anti-clockwise  }    void left()  {    motor1.setSpeed(Speeed); //Define maximum velocity    motor1.run(BACKWARD); //rotate the motor anti-clockwise    motor2.setSpeed(Speeed); //Define maximum velocity    motor2.run(BACKWARD); //rotate the motor anti-clockwise    motor3.setSpeed(Speeed); //Define maximum velocity    motor3.run(FORWARD);  //rotate the motor clockwise    motor4.setSpeed(Speeed); //Define maximum velocity    motor4.run(FORWARD);  //rotate the motor clockwise  }    void right()  {    motor1.setSpeed(Speeed); //Define maximum velocity    motor1.run(FORWARD); //rotate the motor clockwise    motor2.setSpeed(Speeed); //Define maximum velocity    motor2.run(FORWARD); //rotate the motor clockwise    motor3.setSpeed(Speeed); //Define maximum velocity    motor3.run(BACKWARD); //rotate the motor anti-clockwise    motor4.setSpeed(Speeed); //Define maximum velocity    motor4.run(BACKWARD); //rotate the motor anti-clockwise  }    void topleft(){    motor1.setSpeed(Speeed); //Define maximum velocity    motor1.run(FORWARD); //rotate the motor clockwise    motor2.setSpeed(Speeed); //Define maximum velocity    motor2.run(FORWARD); //rotate the motor clockwise    motor3.setSpeed(Speeed/3.1);//Define maximum velocity    motor3.run(FORWARD); //rotate the motor clockwise    motor4.setSpeed(Speeed/3.1);//Define maximum velocity    motor4.run(FORWARD); //rotate the motor clockwise  }    void topright()  {    motor1.setSpeed(Speeed/3.1); //Define maximum velocity    motor1.run(FORWARD); //rotate the motor clockwise    motor2.setSpeed(Speeed/3.1); //Define maximum velocity    motor2.run(FORWARD); //rotate the motor clockwise    motor3.setSpeed(Speeed);//Define maximum velocity    motor3.run(FORWARD); //rotate the motor clockwise    motor4.setSpeed(Speeed);//Define maximum velocity    motor4.run(FORWARD); //rotate the motor clockwise  }    void bottomleft()  {    motor1.setSpeed(Speeed); //Define maximum velocity    motor1.run(BACKWARD); //rotate the motor anti-clockwise    motor2.setSpeed(Speeed); //Define maximum velocity    motor2.run(BACKWARD); //rotate the motor anti-clockwise    motor3.setSpeed(Speeed/3.1); //Define maximum velocity    motor3.run(BACKWARD); //rotate the motor anti-clockwise    motor4.setSpeed(Speeed/3.1); //Define maximum velocity    motor4.run(BACKWARD); //rotate the motor anti-clockwise  }    void bottomright()  {    motor1.setSpeed(Speeed/3.1); //Define maximum velocity    motor1.run(BACKWARD); //rotate the motor anti-clockwise    motor2.setSpeed(Speeed/3.1); //Define maximum velocity    motor2.run(BACKWARD); //rotate the motor anti-clockwise    motor3.setSpeed(Speeed); //Define maximum velocity    motor3.run(BACKWARD); //rotate the motor anti-clockwise    motor4.setSpeed(Speeed); //Define maximum velocity    motor4.run(BACKWARD); //rotate the motor anti-clockwise  }      void Stop()  {    motor1.setSpeed(0); //Define minimum velocity    motor1.run(RELEASE); //stop the motor when release the button    motor2.setSpeed(0); //Define minimum velocity    motor2.run(RELEASE); //rotate the motor clockwise    motor3.setSpeed(0); //Define minimum velocity    motor3.run(RELEASE); //stop the motor when release the button    motor4.setSpeed(0); //Define minimum velocity    motor4.run(RELEASE); //stop the motor when release the button  } **Step 10:** For the Battery Here I am using 2, Samsung 2600mAh Li-ion Batteries.  I attached the batteries on the holder and then turn the switch on for testing the car.  Battery connection with the Bluetooth car\  **Steps to connect the Bluetooth controlled car with Phone:**   * First open the “Make DIY” App. * Currently this app is only available for android users. * Now Turn on your Phone’d Bluetooth. Then Connect with the HC-05 Bluetooth Module. * If you have a Fresh New HC-05 then you have to do one more step before all this steps.   1. First go to your bluetooth settings.   2. Then search for Bluetooth Devices.   3. After that you will Find a Bluetooth Device named HC-05.   4. Then first pair the bluetooth device with your phone.   5. For pairing, you have to add a default Password.   6. Generally the Password is 1234 or 0000.   7. Just give the password and you are now your   8. Bluetooth Controlled car is connected with your Android Device.   Here you can see the Motors are rotating with a phone button press.  And also the Lights are glowing perfectly.  Here you can see when you press the top arrow then only Front LEDs Glows  and similarly when you press the Bottom arrow then the Backside Red LEDs glows  . So the Lighting Part is working perfectly.  Here is the final testing with the Tire connected to the car. In this project,  I used general 65mm TT Motor Tires. The car runs fluently.  No Data Transmission loss is there. The car moves just instantly.  Now, Our Arduino Bluetooth controlled car with L293D is working perfectly with Front and Back Lights. Here I can announce this project is a Great Success.  . |



## Conclussion:

From this fun project, we understand how basically a Bluetooth-controlled car with Front and backlights has been made in real life with Arduino. Here i get vast electronics knowledge from this project. If you are an electronics enthusiastic person then you should definitely try this project in your home