BLUETOOTH CONTROLLEED CAR USING ARDUINO NANO

Rajdeep Saha 25,

**

*Department of Electronics and Communication Engineering,*

*Supreme Knowledge Foundation Group of Institutions*

**Technical Report On**

**Bluetooth Controlled Car Using Arduino NANO**

**Under the Guidance of**

Mr. Anirban Neogi

Assistant Professor

ECE Dept.

**Submitted by**

Rajdeep Saha

University Roll Number

25300322025

**DATE OF SUBMISSION**

31st July 2023

**CERTIFICATE**

This is to certify that this project report entitled” **Bluetooth Controlled Car Using Arduino NANO”** submitted by “**Rajdeep Saha”**  on “ **31st July 2023”**

SIGNATURE

ANIRBAN NEOGI

ASSISTANT PROFESSOR

DEPT.OF ELECTRONICS

AND TELECOMMUNICATION

ENGINEERING

**Bluetooth Controlled Car Using Arduino NANO**

**1. Introduction:**

The Bluetooth Controlled Car is a robotic vehicle designed to be remotely controlled using a smartphone or any other Bluetooth-enabled device. This project combines the functionalities of Arduino, motor drivers, and Bluetooth communication to create an interactive and versatile remote-controlled car. The car can perform various movements, such as moving forward, backward, turning left, turning right, and diagonal movements. Additionally, it includes features like a horn and front/backlights that can be controlled through the Bluetooth interface.

**2. Objective:**

The primary objective of this project is to develop a Bluetooth controlled car that provides an engaging and user-friendly experience. The project aims to demonstrate the integration of Arduino with motor drivers and Bluetooth communication to achieve smooth and precise control over the car's movements and features.

**3. Components Used:**

- Arduino NANO: The main microcontroller responsible for processing Bluetooth commands and controlling the motors and other peripherals.

- L298N Motor Driver Module: Used to control the DC motors for the car's movements.

- HC-05 Bluetooth Module: Enables wireless communication between the car and a mobile device.

- DC Motors: Four motors used to drive the wheels and provide mobility to the car.

- Chassis: The physical structure that holds all the components together, forming the car's body.

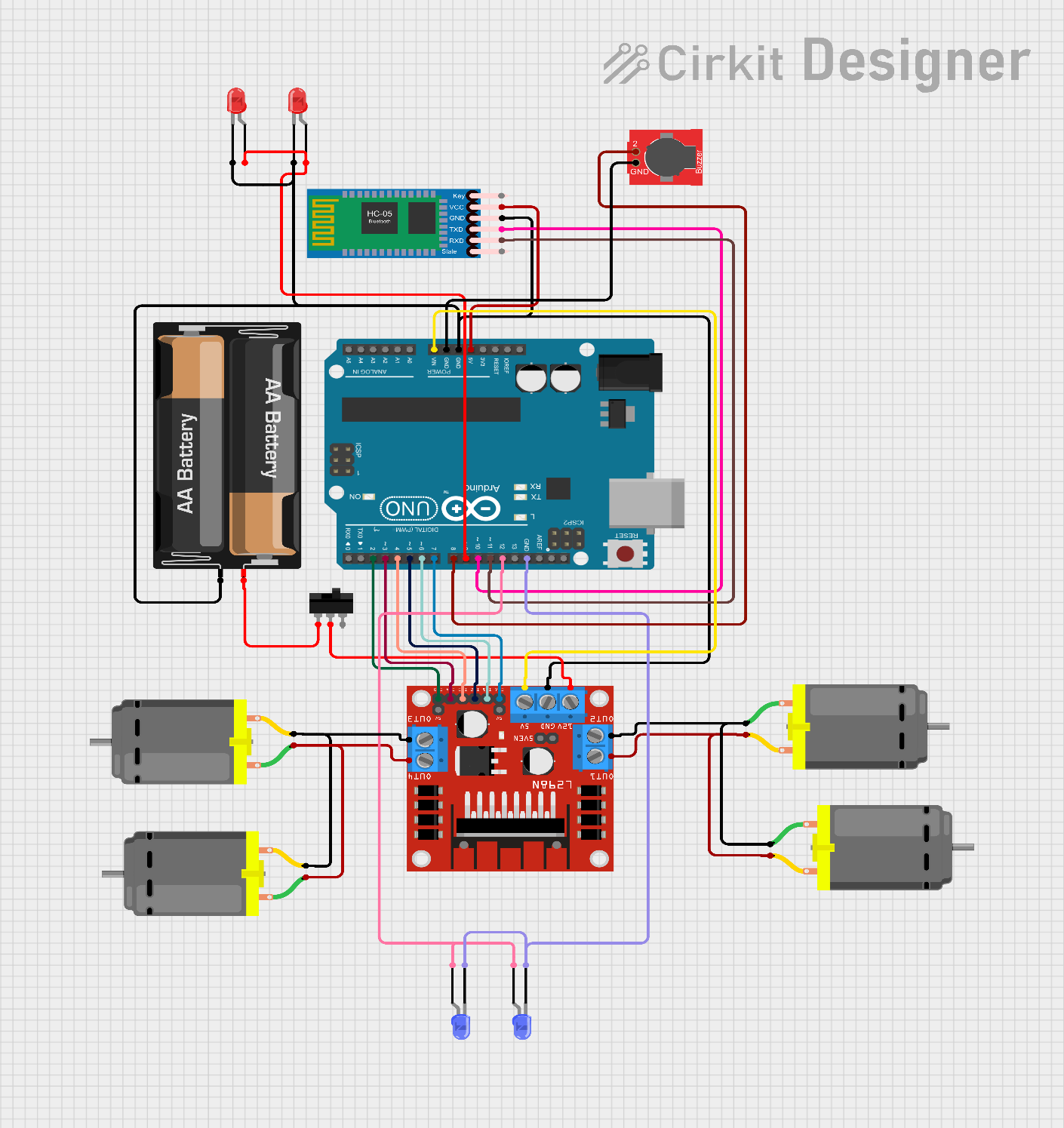
- Wheels: Four wheels that are attached to the DC motors for movement.

- Battery: A suitable power source, such as a 6V LiPo battery, to power the car and its components.

- Jumper wires and Breadboard: Used for connecting and prototyping the electronic circuits.

- LEDs: 6 LEDs for the front and backlights.

- Buzzer: Used to create the horn sound.

**4. Circuit Diagram**: 

The circuit consists of the following connections:

- The L298N Motor Driver Module is connected to the Arduino to control the DC motors.

- The HC-05 Bluetooth Module is connected to the Arduino to enable wireless communication with the mobile device.

- The front and backlights are connected to digital pins of the Arduino.

- The horn (buzzer) is connected to a digital pin of the Arduino.

**5. Working Principle:**

The Bluetooth Controlled Car operates by establishing a Bluetooth connection between the mobile device and the HC-05 Bluetooth Module. Commands are sent from the mobile device to the car, and the Arduino processes these commands to control the movements and features of the car. Depending on the received commands, the Arduino activates specific functions that drive the motors to perform forward, backward, left, right, or diagonal movements. The car's horn and lights can also be controlled using the corresponding commands.

**6. Software Implementation:**

The Arduino code uploaded to the Arduino NANO enables the car to process incoming data from the Bluetooth module. The code reads the characters sent from the mobile device and triggers the appropriate functions based on the received commands. For instance, if the mobile device sends 'F', the Arduino executes the forward() function to make the car move forward. Similarly, other commands activate the relevant functions for different movements and features.

Now, let's go through the code step by step:

1. Include the necessary libraries:

```

#include <SoftwareSerial.h>

```

2. Define the software serial object to communicate with the Bluetooth module:

```

SoftwareSerial BTserial(10, 11); // RX | TX

```

3. Define the pin connections for the motor driver, motors, horn, front light, and back light:

```

int ena = 7;

int motorPin1 = 6;

int motorPin2 = 5;

int motorPin3 = 4;

int motorPin4 = 3;

int enb = 2;

int motorSpeed1 = 0;

int motorSpeed2 = 0;

int horn = 8;

int frontlight = 9;

int backlight = 12;

```

4. Initialize the setup function:

```

void setup() {

Serial.begin(9600);

BTserial.begin(9600);

// Set pin modes for motors, motor driver, horn, front light, and back light.

pinMode(motorPin1, OUTPUT);

pinMode(motorPin2, OUTPUT);

pinMode(ena, OUTPUT);

pinMode(enb, OUTPUT);

pinMode(motorPin3, OUTPUT);

pinMode(motorPin4, OUTPUT);

pinMode(horn, OUTPUT);

pinMode(frontlight, OUTPUT);

pinMode(backlight, OUTPUT);

}

```

5. Implement the main loop:

```

void loop() {

// Check if data is available from the Bluetooth module

if (BTserial.available() > 0) {

char data = BTserial.read();

Serial.write(data); // Send received data to Serial Monitor

// Check the received data and execute corresponding actions

if (data == 'F') {

Serial.println("Moving Forward...");

forward();

}

else if (data == 'B') {

Serial.println("Moving Backward...");

backward();

}

else if (data == 'R') {

Serial.println("Turning Left...");

left();

}

else if (data == 'L') {

Serial.println("Turning Right...");

right();

}

// Other commands for diagonal movements (G, I, H, J), horn (V, v),

// front light (W, w), and back light (U, u) are handled similarly.

else if (data == 'S') {

Serial.println("Stopping...");

stop();

}

}

}

```

6. Implement the movement functions:

```

void forward() {

motorSpeed1 = 255;

motorSpeed2 = 255;

digitalWrite(motorPin1, LOW);

digitalWrite(motorPin2, HIGH);

digitalWrite(motorPin3, HIGH);

digitalWrite(motorPin4, LOW);

analogWrite(ena, motorSpeed1);

analogWrite(enb, motorSpeed1);

}

void backward() {

// Similar to forward, but with different motor control to move backward.

}

// Implement the forwardleft(), forwardright(), backleft(), and backright() functions for diagonal movements.

void left() {

motorSpeed1 = 200;

motorSpeed2 = 200;

digitalWrite(motorPin1, HIGH);

digitalWrite(motorPin2, LOW);

digitalWrite(motorPin3, HIGH);

digitalWrite(motorPin4, LOW);

analogWrite(ena, motorSpeed1);

analogWrite(enb, motorSpeed2);

}

void right() {

// Similar to left, but with different motor control to turn right.

}

void stop() {

digitalWrite(motorPin1, LOW);

digitalWrite(motorPin2, LOW);

digitalWrite(motorPin3, LOW);

digitalWrite(motorPin4, LOW);

analogWrite(ena, 0);

analogWrite(enb, 0);

}

```

7. Implement the other functions for controlling the horn, front light, and back light, similar to the movement functions.

**7. Features and Functions:**

The Bluetooth Controlled Car includes the following features and functions:

- Moving Forward: The car moves forward when the character 'F' is received.

- Moving Backward: The car moves backward when the character 'B' is received.

- Turning Left: The car turns left when the character 'L' is received.

- Turning Right: The car turns right when the character 'R' is received.

- Diagonal Movements: The car can move diagonally in four directions (front left, front right, back left, back right) when specific characters (G, I, H, J) are received.

- Horn: The horn can be activated by sending the character 'V', and it can be turned off by sending 'v'.

- Front Light: The front light can be turned on with the character 'W' and off with 'w'.

- Back Light: The back light can be turned on with the character 'U' and off with 'u'.

**8. Testing and Demonstration:**

The Bluetooth Controlled Car is tested by establishing a Bluetooth connection between the mobile device and the car's HC-05 Bluetooth Module. A Bluetooth terminal app is used to send commands to the car. The car should respond accurately and perform the desired movements and functions as per the received commands.

**9. Conclusion:**

The Bluetooth Controlled Car project successfully demonstrates the integration of Arduino, Bluetooth communication, and motor drivers to build a remote-controlled car. It provides an interactive and entertaining experience for users to control the car wirelessly through a mobile device. The project showcases the versatility of Arduino-based projects and encourages further exploration and enhancements.

**10. Future Enhancements:**

Several potential enhancements can be made to further improve the Bluetooth Controlled Car:

- Obstacle Avoidance: Integrate ultrasonic sensors or infrared sensors to enable obstacle avoidance and autonomous navigation.

- Speed Control: Implement speed control functionality to adjust the car's speed based on user input.

- Gesture Control: Incorporate gesture recognition to control the car using hand movements or gestures detected by the mobile device's sensors.

- Mobile App Interface: Design a user-friendly mobile app with a graphical interface to control the car's movements and features more intuitively.

- Video Streaming: Add a camera to the car and stream live video to the mobile device for real-time monitoring and remote exploration.

Overall, the Bluetooth Controlled Car project serves as an exciting and educational platform for learning about Arduino-based robotics and wireless communication. It provides an engaging experience for users and offers ample opportunities for further customization and development.