Low-Cost Mecanum Wheel RC Car Using Arduino

---

1. Overview

This document outlines the design and construction of a minimal, low-cost Mecanum wheel RC car using Arduino. The project focuses on remote control capability, holonomic motion (movement in any direction), and affordability. It is an ideal side project to enhance a technical resume.

---

2. Components List

- **Arduino Nano or Uno**
- **L298N Motor Driver Module (or 2x)**
- **HC-05 Bluetooth Module**
- **4 x DC Gear Motors (preferably with encoders)**
- **4 x Mecanum Wheels (2 Left, 2 Right)**
- **12V Battery Pack (or 2S/3S Li-ion)**
- **Chassis (Acrylic, 3D printed, or wood)**
- **Jumper wires, screws, zip ties**
- **Smartphone (for Bluetooth control app)**

---

3. Schematic Diagram (Textual Description)

- **Motor Connections (via L298N):**

- Motor A: IN1, IN2 → Left Front Motor

- Motor B: IN3, IN4 → Right Rear Motor

- Second L298N (if needed):

    - IN1, IN2 → Right Front Motor
    - IN3, IN4 → Left Rear Motor

- **Bluetooth (HC-05):**

- VCC → 5V

- GND → GND
- TX → RX (via voltage divider)

- RX → TX

- **Power:**

- Battery powers motors through VIN on L298N
- Arduino powered via 5V regulator or onboard 5V pin

---

4. Frame Design

- Rectangular or square acrylic chassis (20x20 cm or so)
- Mecanum wheels mounted at all four corners
- Motors securely screwed to base
- Motor drivers and Arduino mounted in center
- Bluetooth module accessible on top

---

5. Arduino Code for Basic Mecanum Control

```
char command;
int LF1 = 2, LF2 = 3;  // Left Front
int RF1 = 4, RF2 = 5;  // Right Front
int LR1 = 6, LR2 = 7;  // Left Rear
int RR1 = 8, RR2 = 9;  // Right Rear

void setup() {
  Serial.begin(9600);
  pinMode(LF1, OUTPUT); pinMode(LF2, OUTPUT);
  pinMode(RF1, OUTPUT); pinMode(RF2, OUTPUT);
  pinMode(LR1, OUTPUT); pinMode(LR2, OUTPUT);
  pinMode(RR1, OUTPUT); pinMode(RR2, OUTPUT);
}

void loop() {
  if (Serial.available()) {
    command = Serial.read();
    stopCar();
    switch(command) {
      case 'F': forward(); break;
      case 'B': backward(); break;
      case 'L': strafeLeft(); break;
      case 'R': strafeRight(); break;
      case 'X': rotateCW(); break;
      case 'Z': rotateCCW(); break;
    }
  }
}

void forward() {
```

```
  digitalWrite(LF1,HIGH); digitalWrite(LF2,LOW);
  digitalWrite(RF1,HIGH); digitalWrite(RF2,LOW);
  digitalWrite(LR1,HIGH); digitalWrite(LR2,LOW);
  digitalWrite(RR1,HIGH); digitalWrite(RR2,LOW);
}

void backward() {
  digitalWrite(LF1,LOW); digitalWrite(LF2,HIGH);
  digitalWrite(RF1,LOW); digitalWrite(RF2,HIGH);
  digitalWrite(LR1,LOW); digitalWrite(LR2,HIGH);
  digitalWrite(RR1,LOW); digitalWrite(RR2,HIGH);
}

void strafeLeft() {
  digitalWrite(LF1,LOW); digitalWrite(LF2,HIGH);
  digitalWrite(RF1,HIGH); digitalWrite(RF2,LOW);
  digitalWrite(LR1,HIGH); digitalWrite(LR2,LOW);
  digitalWrite(RR1,LOW); digitalWrite(RR2,HIGH);
}

void strafeRight() {
  digitalWrite(LF1,HIGH); digitalWrite(LF2,LOW);
  digitalWrite(RF1,LOW); digitalWrite(RF2,HIGH);
  digitalWrite(LR1,LOW); digitalWrite(LR2,HIGH);
  digitalWrite(RR1,HIGH); digitalWrite(RR2,LOW);
}

void rotateCW() {
  digitalWrite(LF1,HIGH); digitalWrite(LF2,LOW);
  digitalWrite(RF1,LOW); digitalWrite(RF2,HIGH);
  digitalWrite(LR1,HIGH); digitalWrite(LR2,LOW);
  digitalWrite(RR1,LOW); digitalWrite(RR2,HIGH);
}

void rotateCCW() {
  digitalWrite(LF1,LOW); digitalWrite(LF2,HIGH);
  digitalWrite(RF1,HIGH); digitalWrite(RF2,LOW);
  digitalWrite(LR1,LOW); digitalWrite(LR2,HIGH);
  digitalWrite(RR1,HIGH); digitalWrite(RR2,LOW);
}

void stopCar() {
  digitalWrite(LF1,LOW); digitalWrite(LF2,LOW);
  digitalWrite(RF1,LOW); digitalWrite(RF2,LOW);
  digitalWrite(LR1,LOW); digitalWrite(LR2,LOW);
  digitalWrite(RR1,LOW); digitalWrite(RR2,LOW);
}
```

6. Resume Value

This project showcases:

- Embedded systems and motor control
- Holonomic drive understanding
- Remote communication via Bluetooth
- Arduino development experience
- Mechanical assembly and layout

---

7. Next Steps (Optional)

- Add encoder feedback for better control
- Implement PID speed control
- Add obstacle detection with ultrasonic sensors
- Implement autonomous navigation via line following or waypoints

---

Note: Control can be done using a Bluetooth serial controller app (like "Arduino Bluetooth Controller" on Android).