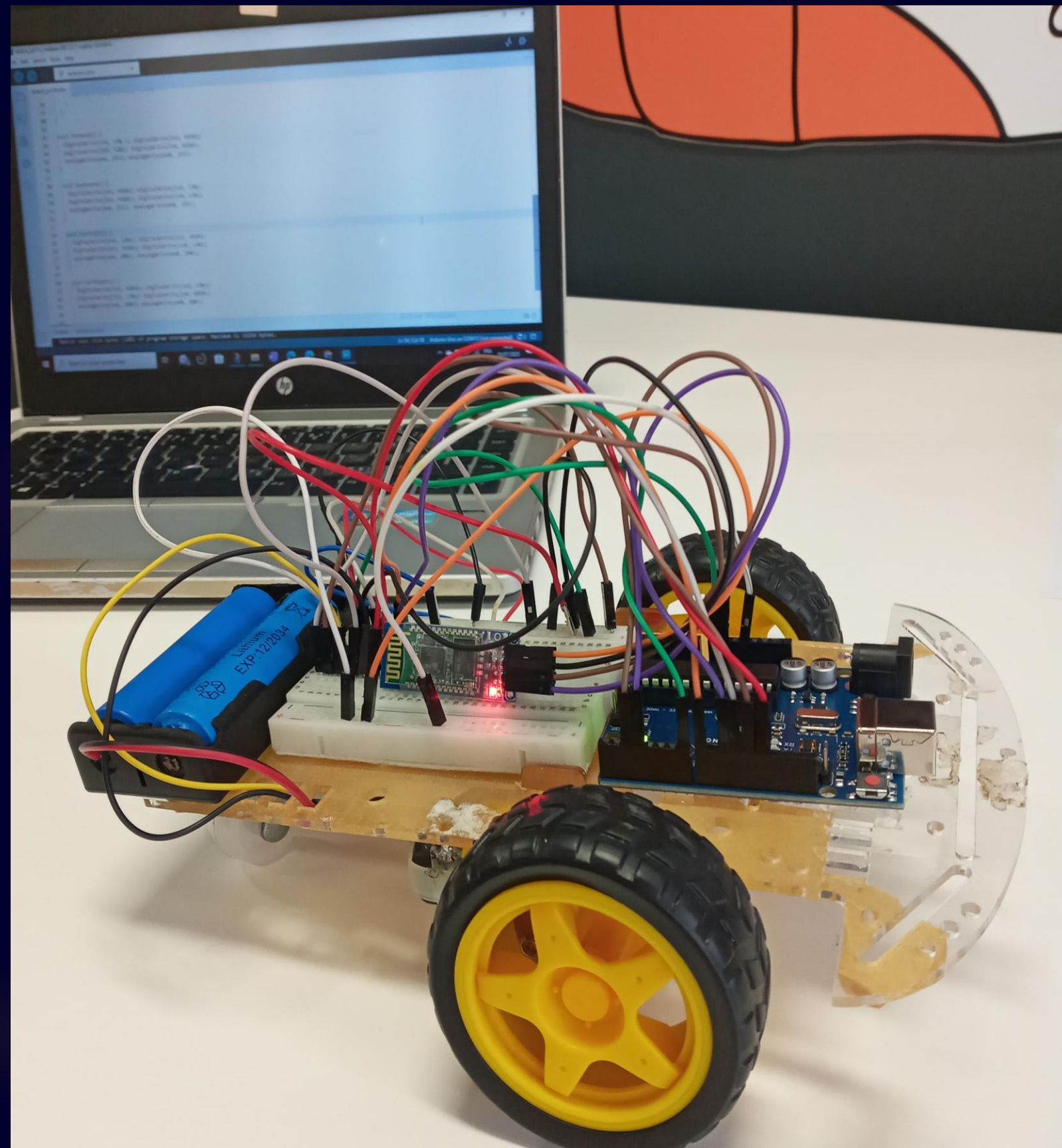


Real-Time Robot Control Using Arduino and Bluetooth (HC-06)

This presentation explores a mini-project focused on wireless motor control using Arduino and the L293D motor driver. We'll delve into the fascinating world of robotics, from basic components to practical implementation and debugging.






Kaoutar Rachdi



Project Objective: Wireless Robot Control

Our primary goal is to wirelessly control a two-wheeled robot via a smartphone using Bluetooth commands. This enables precise movement for:

-  **Forward and Backward Motion**
Navigate the robot in linear directions.
-  **Left and Right Turns**
Execute directional changes with accuracy.
-  **Emergency Stop Command**
Halt all movement instantly for safety or precision.



Essential Components

To bring our robot to life, we utilize a specific set of **hardware** and **software components**, carefully selected for their functionality and compatibility.

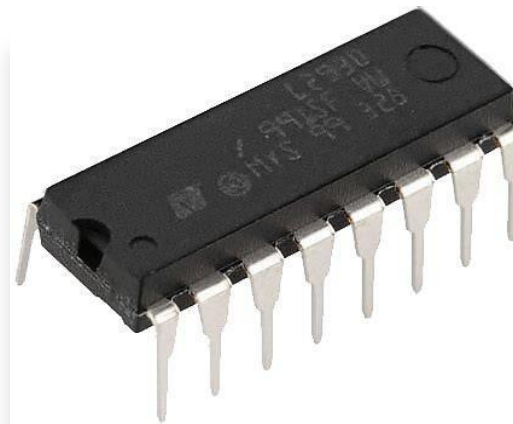
- Arduino Uno: The brain of our robot, processing commands.
- HC-06 Bluetooth Module: For wireless communication.
- L293D Motor Driver: To control the DC motors.
- 2 DC Motors: Provide mobility to the robot.
- 7.4V Battery (2 × 3.7V Li-ion): Powers the entire system.
- Jumper Wires: For electrical connections.
- Breadboard: For prototyping and circuit assembly.
- 2WD mobile robot platform
- Proteus Software: For circuit simulation and verification.
- Arduino IDE: For writing and uploading code.



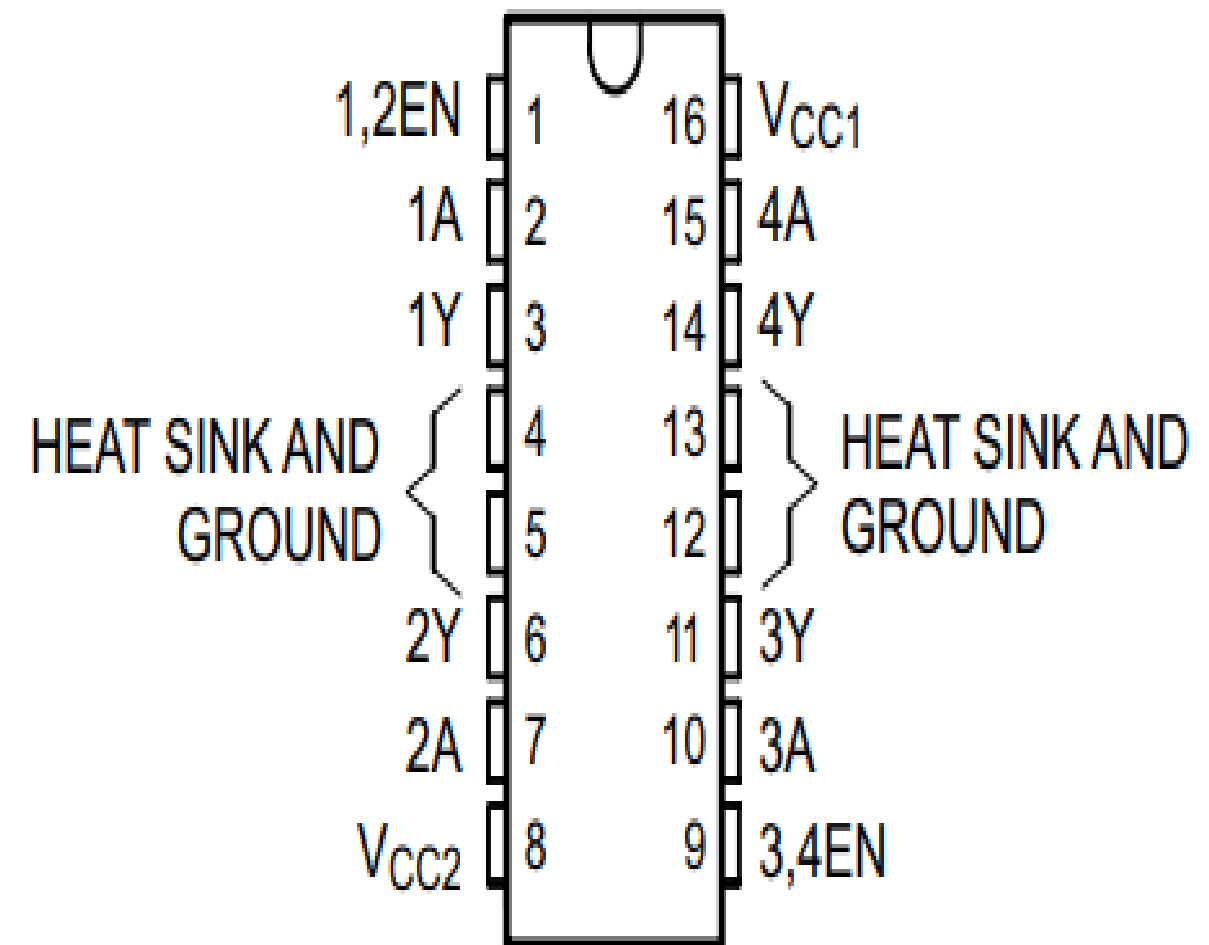
The L293D Motor Driver

- The **L293D** is dual H-Bridge motor driver ICs used to control two DC motors independently. Each IC includes two full **H-Bridge drivers**, and can be controlled using logic signals from microcontrollers like the Arduino. Key pins include:
- **IN1/IN2 & IN3/IN4**: Control motor directions.
- **EN1/EN2**: Enable/disable motor channels.
- **VCC1 (Logic Power)**: Typically connected to 5V.
- **VCC2 (Motor Power)**: Connected to external motor supply (e.g., 7.4V).
- **GND**: Ground.

This pinout is essential for correct wiring and integration into a larger circuit.

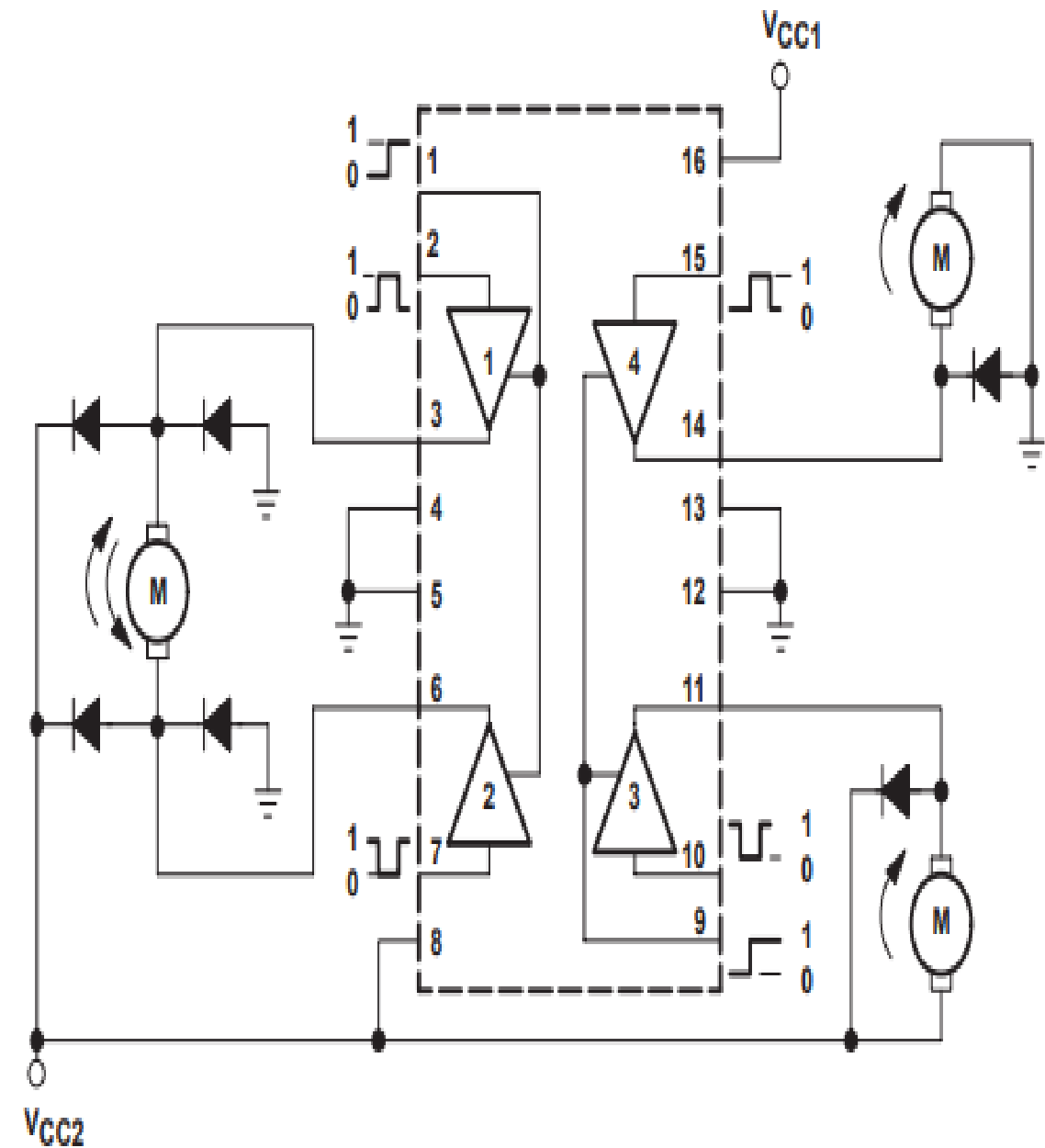


NE Package
16-Pin PDIP
Top View



Understanding the H-Bridge Concept

- The **H-Bridge** is a key electronic circuit used to control the direction of DC motors. It consists of four switching elements (typically transistors or **MOSFETs**) arranged in an 'H' shape. By selectively activating two transistors at opposite corners, current can flow in either direction through the motor, enabling both forward and reverse motion. This mechanism is fundamental in robotics and motor control, allowing for bidirectional movement using simple digital signals from a microcontroller like Arduino.



Internal Functional Diagram of the L293D

- Internally, the L293D features two independent H-Bridge driver circuits. Each H-Bridge consists of transistors arranged to handle high current and control the direction of current flow through the connected motor. The internal diagram helps understand how inputs (IN1–IN4) control the motor driver outputs (OUT1–OUT4), and how enable pins (EN1/EN2) activate each channel. This insight is crucial for debugging, simulation, and optimizing control strategies.

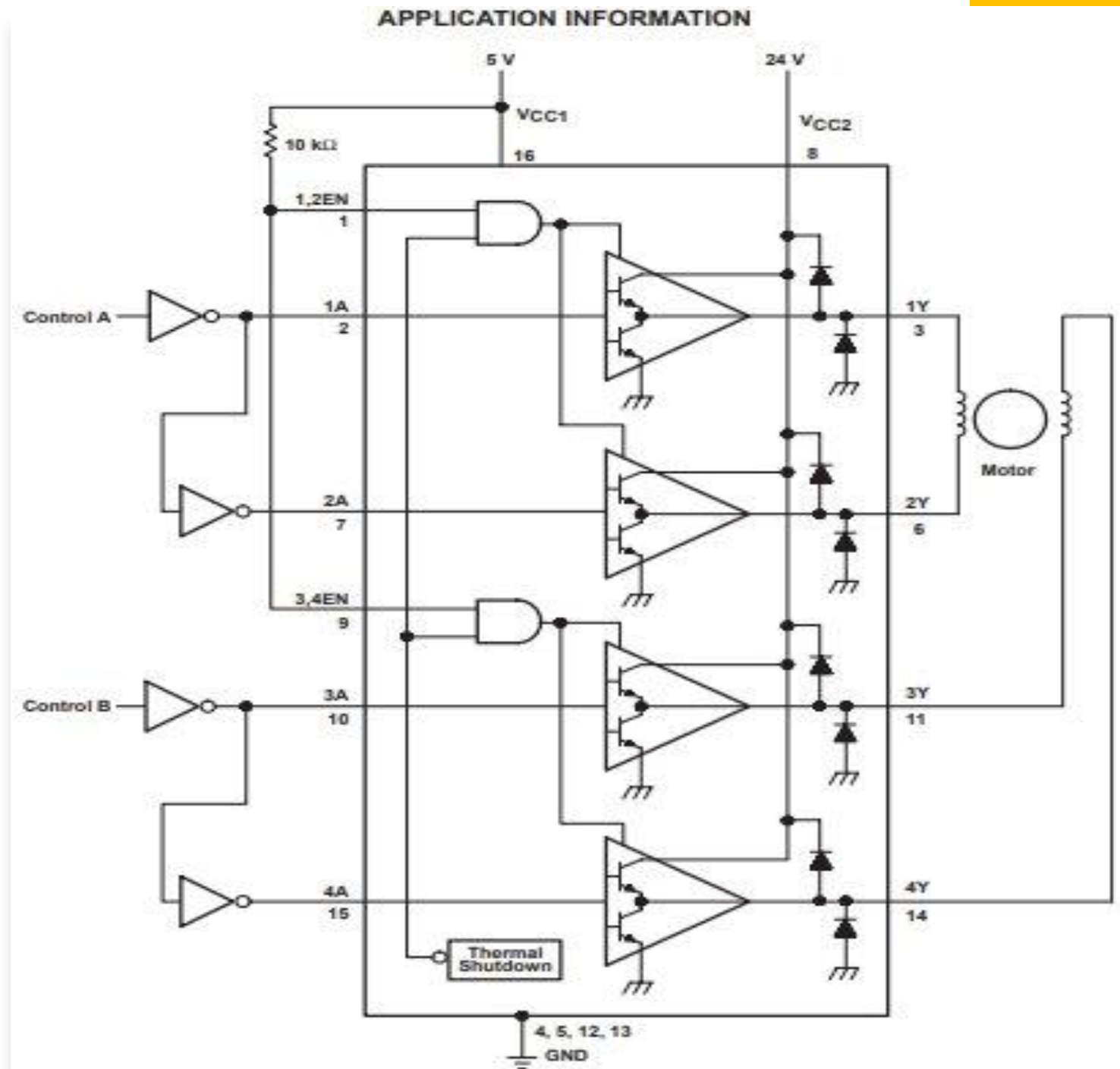
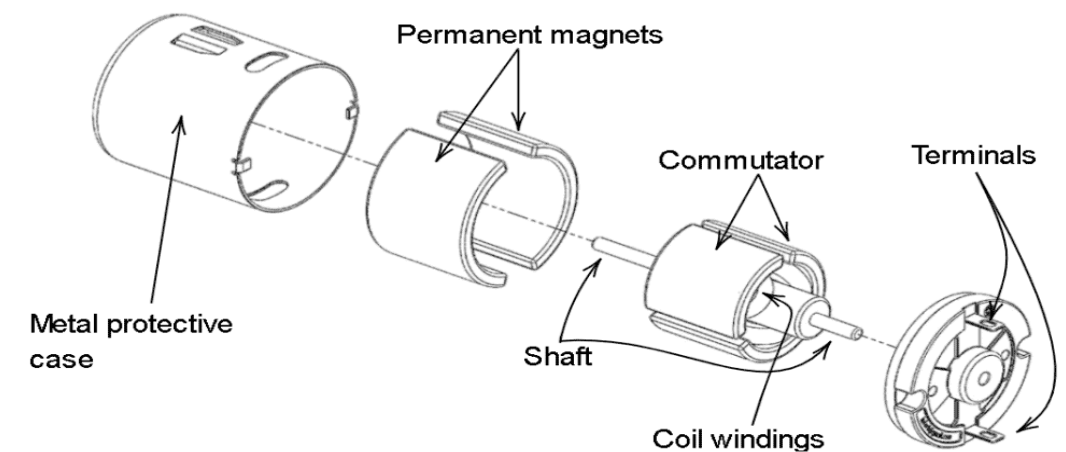
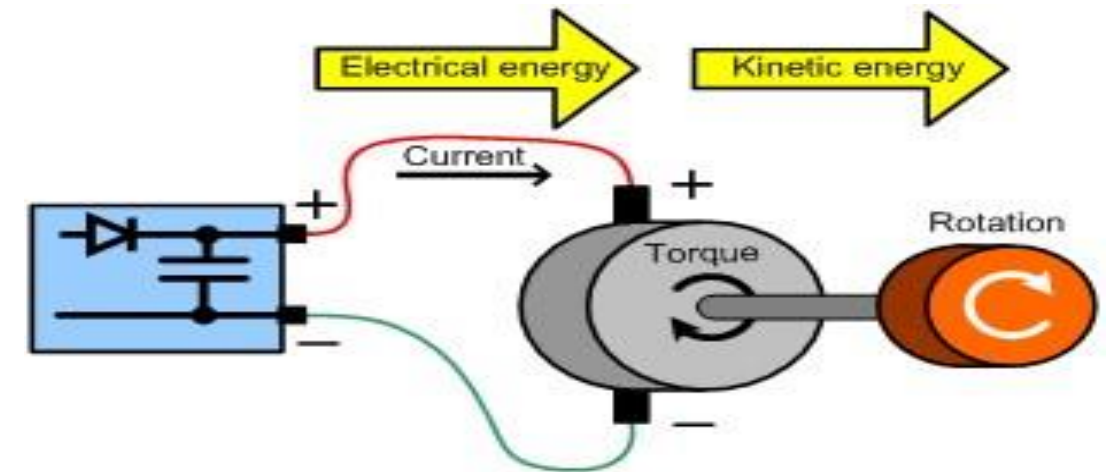
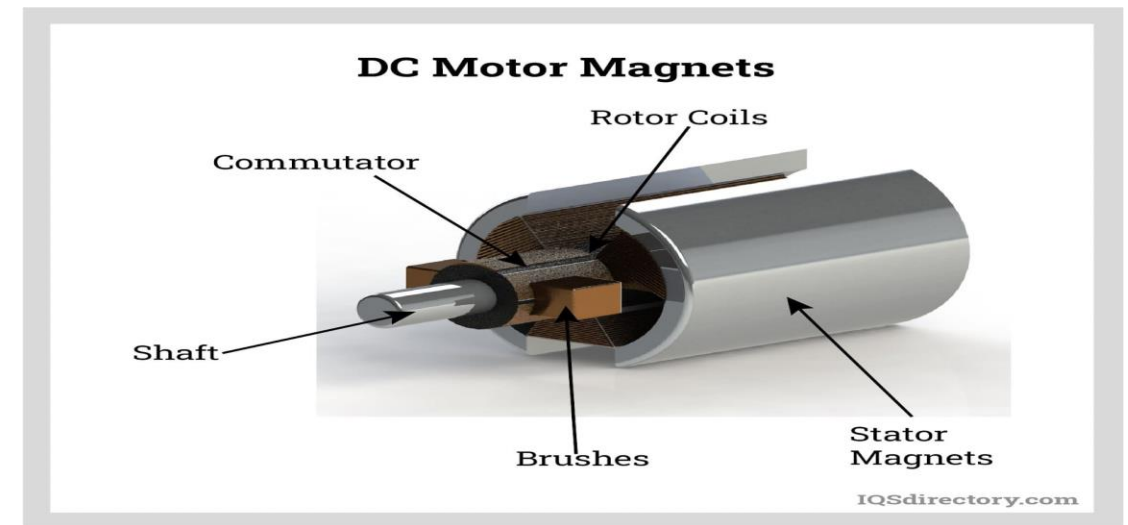


Figure 3. Two-Phase Motor Driver (L293D)

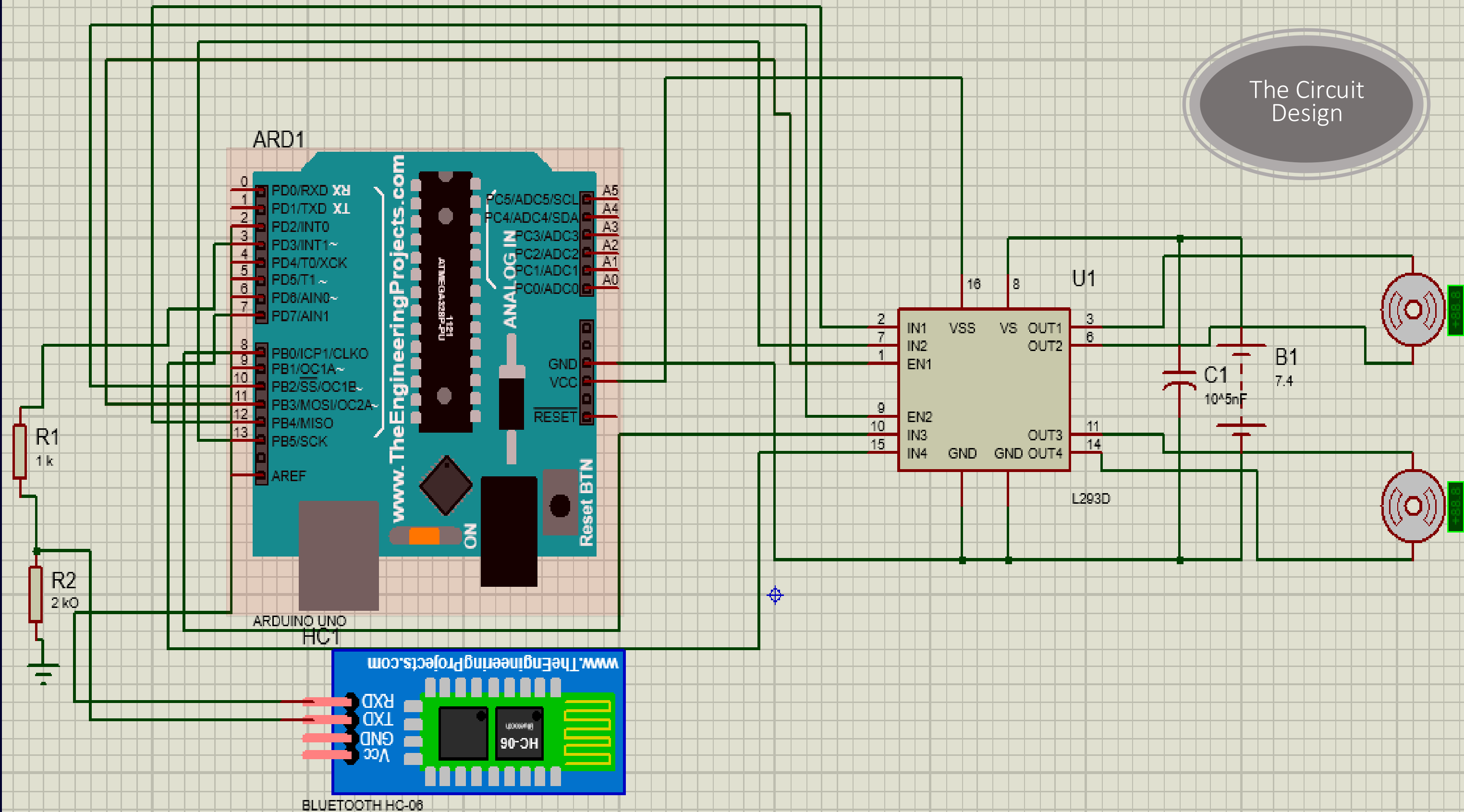


Understanding the DC Motor

- A **DC motor** is an electromechanical device that converts direct current (DC) electrical energy into mechanical rotational motion. It works based on the principle that when an electric current flows through a coil placed in a magnetic field, it experiences a force (**Lorentz force**) that causes rotation. The main components include a **stator** (providing the magnetic field), a **rotor** or **armature** (which rotates), brushes (delivering current), and a **commutator** (reversing current direction to maintain rotation). The motor's rotation direction depends on the polarity of the applied voltage, and its speed can be adjusted by varying the voltage level.



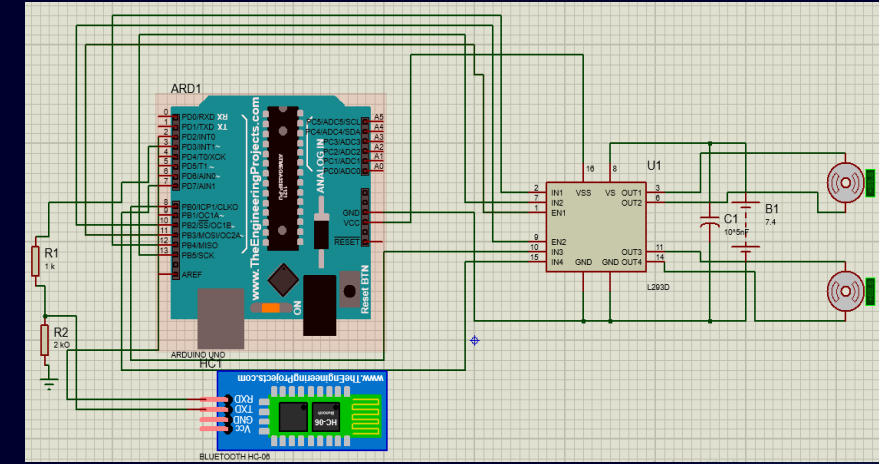
The Circuit Design

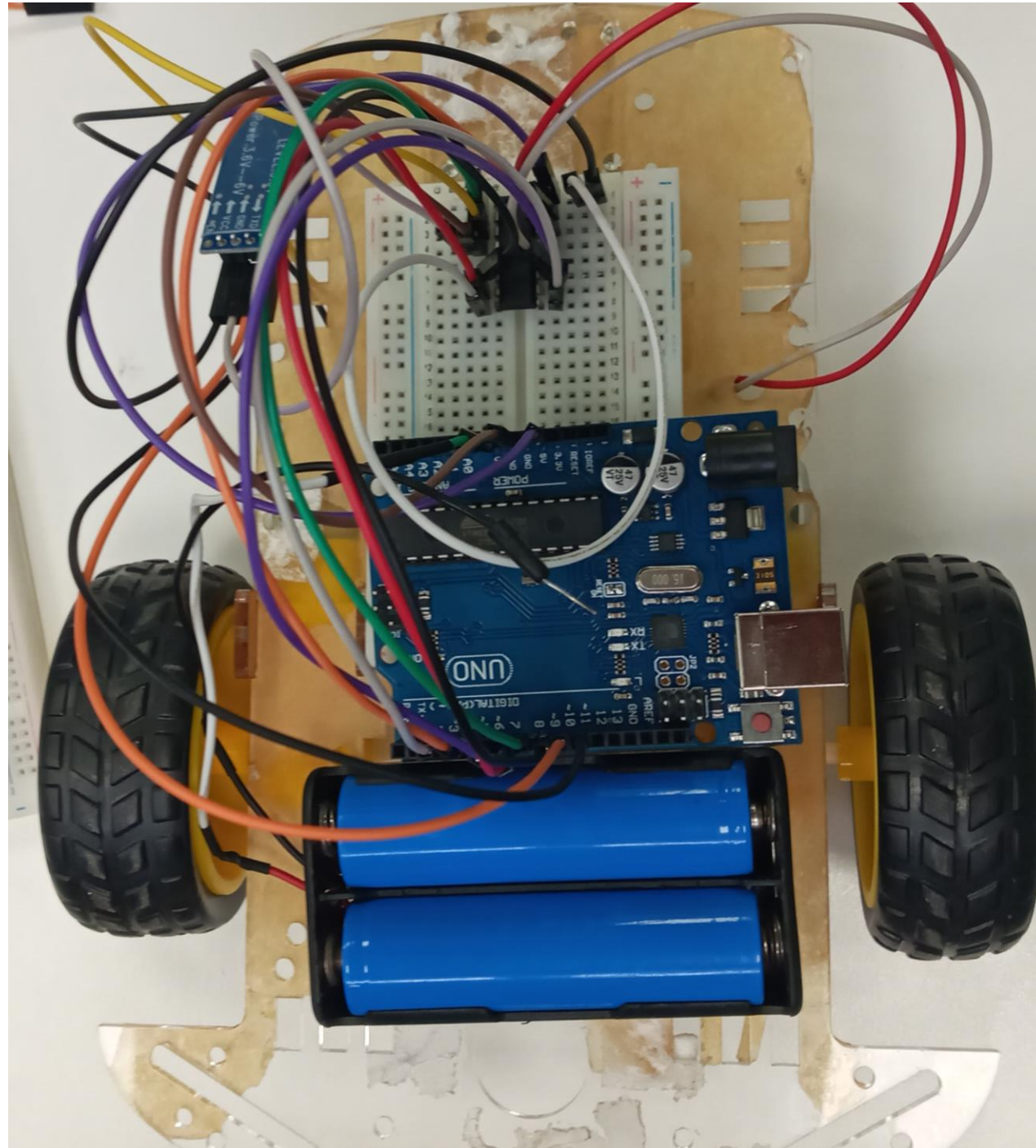


The accurate connection of each component is crucial for the robot's functionality. The design prioritizes stable communication and efficient motor control.

Components :

- **L293D Integration:** Motor driver connected directly to Arduino digital pins for command reception.
- **Motor Connections:** DC motors wired to the L293D's output pins, ensuring proper power delivery and direction control.
- **Bluetooth HC-06 Interface:**
 - HC-06 TX (Transmit) → Voltage Divider → Arduino RX (Pin 2)
 - Arduino TX (Pin 3) → HC-06 RX (Receive)
- **Voltage Stability:** A 100μF capacitor positioned between power rails to smooth out voltage fluctuations, particularly critical in the Proteus simulation environment.
- **Power Source:** The 7.4V battery supplies power to both the L293D motor driver and the Arduino Uno, ensuring consistent operation.
- **Voltage divider:** protects HC-06 RX (since it works at 3.3V logic).

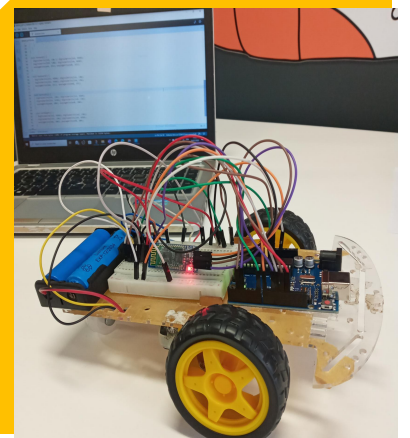




Final Wiring

Wiring the Physical Prototype

- To bring the project to life, I carefully assembled the hardware components on a breadboard, replicating the simulated circuit in a real-world environment. The Arduino Uno acts as the central controller, connected to the HC-06 Bluetooth module for wireless communication, the L293D motor driver to control two DC motors, and a 7.4V battery pack for power.



The Code

```
✓ → Arduino Uno Verify

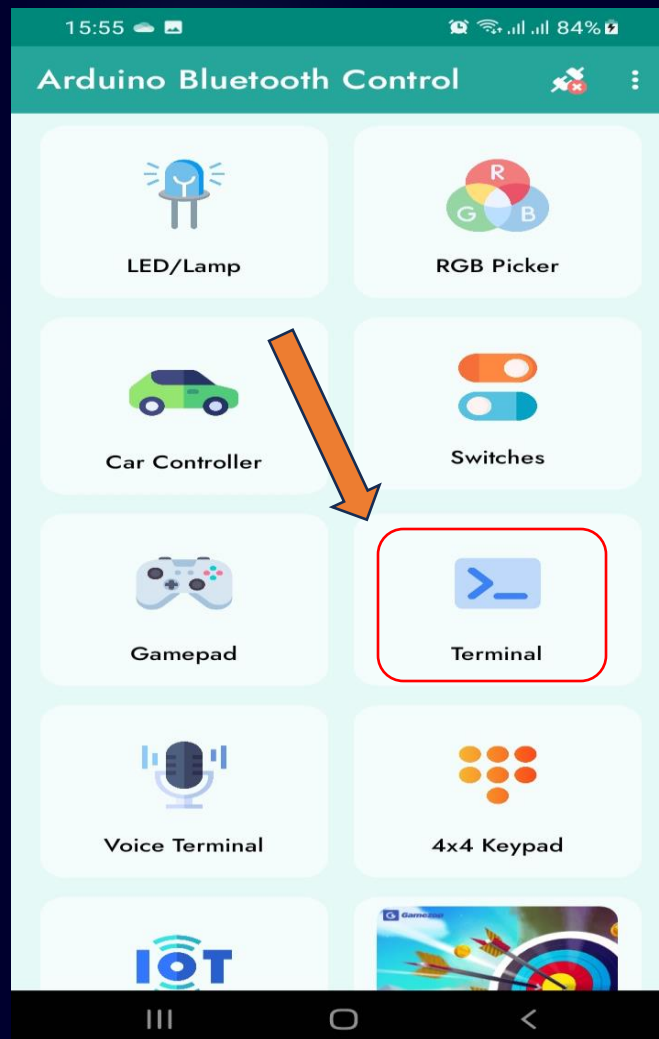
MultipleUltrasonicSensors.ino
1  #include <SoftwareSerial.h>
2
3  // Définir les broches moteurs
4  int enA = 11;
5  int in1 = 12;
6  int in2 = 13;
7
8  int enB = 10;
9  int in3 = 8;
10 int in4 = 7;
11
12 // Créer une liaison série logicielle pour Bluetooth
13 SoftwareSerial bluetooth(2, 3); // RX, TX
14
15 void setup() {
16     // Initialiser les moteurs
17     pinMode(enA, OUTPUT); pinMode(in1, OUTPUT); pinMode(in2, OUTPUT);
18     pinMode(enB, OUTPUT); pinMode(in3, OUTPUT); pinMode(in4, OUTPUT);
19
20     // Initialiser la communication série
21     bluetooth.begin(9600);
22     Serial.begin(9600); // Facultatif, pour debug via USB
23
24     stopMotors();
25 }
26
27 void loop() {
```

```
MultipleUltrasonicSensors.ino
27 void loop() {
28     if (bluetooth.available()) {
29         char cmd = bluetooth.read();
30         Serial.println(cmd); // Debug facultatif
31
32         switch (cmd) {
33             case 'f': forward(); break;
34             case 'b': backward(); break;
35             case 'l': turnLeft(); break;
36             case 'r': turnRight(); break;
37             case 's': stopMotors(); break;
38         }
39     }
40 }
41
42 void forward() {
43     digitalWrite(in1, LOW ); digitalWrite(in2, HIGH);
44     digitalWrite(in3, LOW); digitalWrite(in4, HIGH);
45     analogWrite(enA, 255); analogWrite(enB, 255);
46 }
47
48 void backward() {
49     digitalWrite(in1, HIGH); digitalWrite(in2, LOW);
50     digitalWrite(in3, HIGH); digitalWrite(in4, LOW);
51     analogWrite(enA, 255); analogWrite(enB, 255);
52 }
53
```

```
MultipleUltrasonicSensors.ino
43     digitalWrite(in1, LOW ); digitalWrite(in2, HIGH);
44     digitalWrite(in3, LOW); digitalWrite(in4, HIGH);
45     analogWrite(enA, 255); analogWrite(enB, 255);
46 }
47
48 void backward() {
49     digitalWrite(in1, HIGH); digitalWrite(in2, LOW);
50     digitalWrite(in3, HIGH); digitalWrite(in4, LOW);
51     analogWrite(enA, 255); analogWrite(enB, 255);
52 }
53
54 void turnLeft() {
55     digitalWrite(in1, LOW); digitalWrite(in2, HIGH);
56     digitalWrite(in3, HIGH); digitalWrite(in4, LOW);
57     analogWrite(enA, 200); analogWrite(enB, 200);
58 }
59
60 void turnRight() {
61     digitalWrite(in1, HIGH); digitalWrite(in2, LOW);
62     digitalWrite(in3, LOW); digitalWrite(in4, HIGH);
63     analogWrite(enA, 200); analogWrite(enB, 200);
64 }
65
66 void stopMotors() {
67     digitalWrite(in1, LOW); digitalWrite(in2, LOW);
68     digitalWrite(in3, LOW); digitalWrite(in4, LOW);
69 }
```


Bluetooth Communication Protocol

The HC-06 module serves as the bridge for wireless control, translating smartphone commands into actions. This section details the critical aspects of its setup and operation.

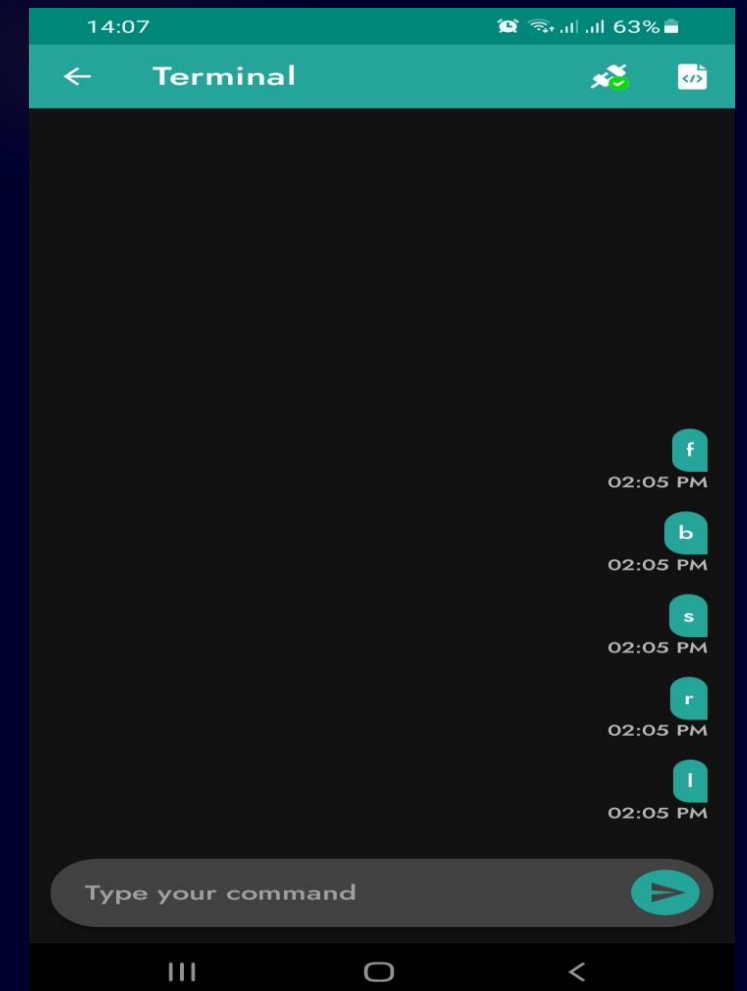


Listening for Commands

The HC-06 module listens wirelessly for incoming Bluetooth commands from the mobile device, and transmits them via serial UART to the Arduino for processing.

Baud Rate Standardization

Communication is established at a standard baud rate of 9600 bps, ensuring reliable data transfer between the module and the Arduino.



Arduino Sketch Summary

The Arduino code efficiently manages the robot's movements based on received Bluetooth commands, ensuring precise and real-time control.

Setup Phase

- **Serial Communication:** Initializes the serial port for communication with the HC-06 module.
- **Pin Mode Configuration:** Sets the direction and control pins for the L293D motor driver as outputs.

Loop Function

- **Bluetooth Input Reading:** Continuously monitors the serial input from the HC-06 for incoming commands.
- **Motor Direction Control:** Based on the received character, it activates the appropriate motor pins to control movement.

Command Conditions:

- 'f' → Forward
- 'b' → Backward
- 'l' → Turn Left
- 's' → Stop
- 'r' → Turn Right

Key Learning Outcomes

This project provided me with invaluable insights into various aspects of robotics and electronics.

Motor Control Mastery

Developed a deep understanding of H-bridge motor control principles using the L293D, crucial for precise robot movement.

Secure Serial Communication

Implemented voltage level shifting for safe and reliable serial communication, protecting sensitive components.

Real-Time Debugging Skills

Gained practical experience in debugging real-time systems using SoftwareSerial, identifying and resolving issues efficiently.



Kaoutar Rachdi