DOCUMENTATION

Stage 1

DC Motors and Motor Drivers

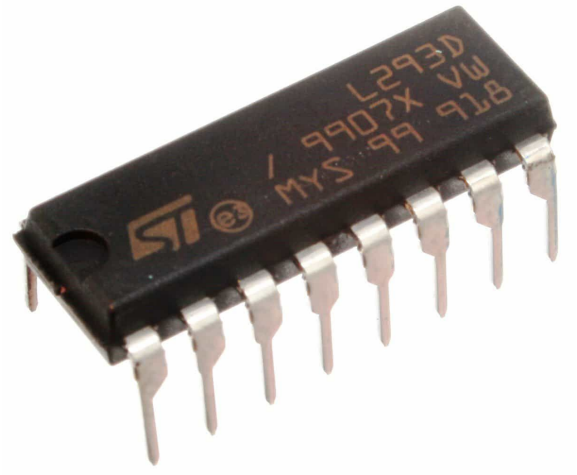
1. Introduction to Components

a. Arduino Uno (Microcontroler)

The Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital Input/Output pins, and 6 analog inputs. For this experiment we are going to use 6 Digital Output pins. 2 of which are going to be used for controlling motor speed (PWM) and the other 4 for motor direction.

b. L293D Motor Driver

The L293D is a quadruple Half-H Bridge IC used to drive motors, in this case brushed DC motors. It is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5V to 36V.

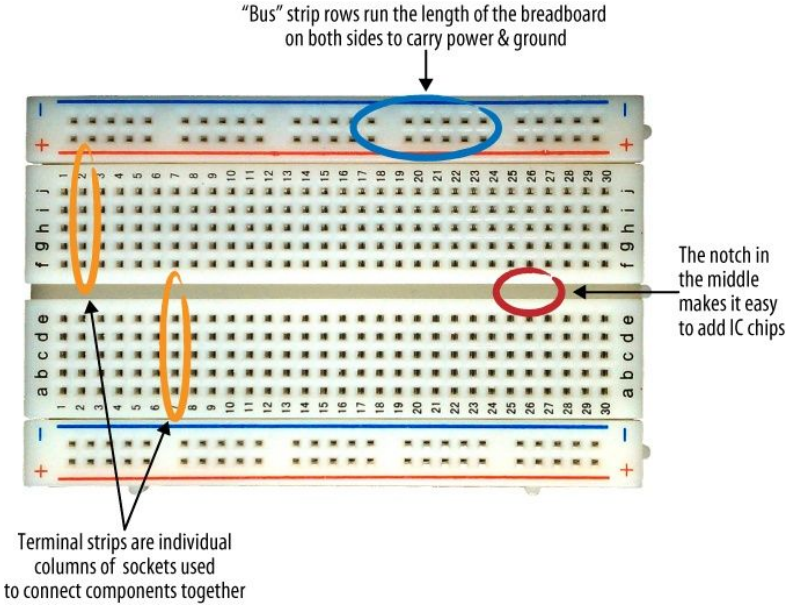


c. Brushed DC Motors

Brushed DC motors, as its name suggests, is a type of motor that is powered by a DC source. It is a useful component that provides mechanical energy and has plenty of applications especially in robotics. Its speed can be controlled by varying the operating voltage, and depending on how you connect the motors, its rotational direction (CW or CCW) can be adjusted.

d. Breadboard

The breadboard is an important and useful tool in electronics. It is used for prototyping circuits where they test and debug those circuits before soldering. It is usually made up of two types of strips - terminal strips and bus strips.



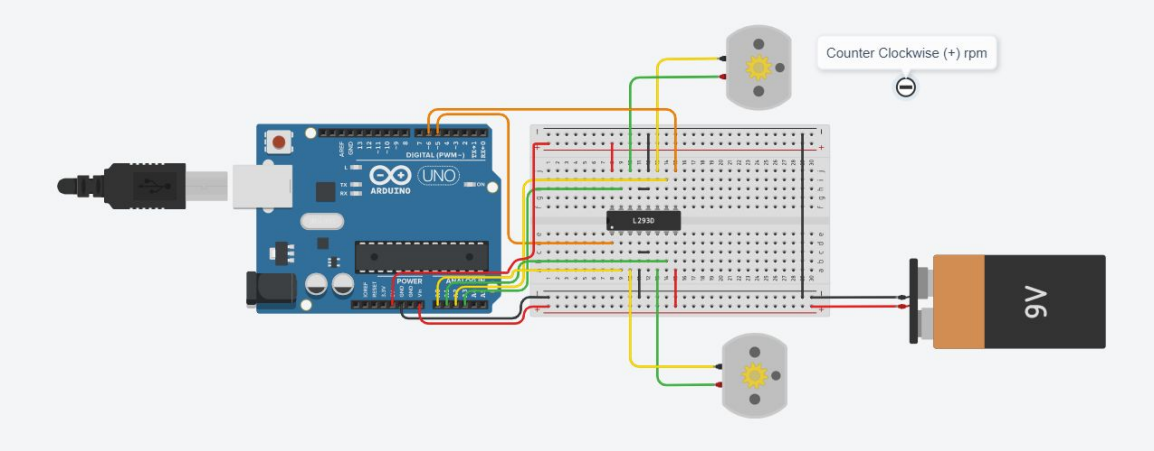
The terminal strips are the main area where most electrical components are placed. One terminal strip is composed of one row of interconnected electrical terminals or holes. The bus strips are usually used to provide power to the electrical components.

One bus strip is composed of one column of interconnected electrical terminals or holes. In the case where you want to connect two or more strips, you can use jumper wires to do so.

2. Connecting the Components

a. Overview

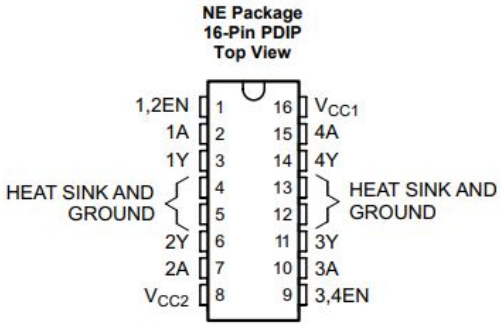
In this experiment we will learn how to control both speed and direction of two Brushed DC motors using the Arduino Uno and the L293D. In general the setup should look as below.



b. Arduino Nano and L293D pin connections

First thing we want to connect are the control pins. Enable pins are the pins that will control the motor's speed and the Input pins will control Motor A and B's direction (Clockwise or counter clockwise).

5 Pin Configuration and Functions



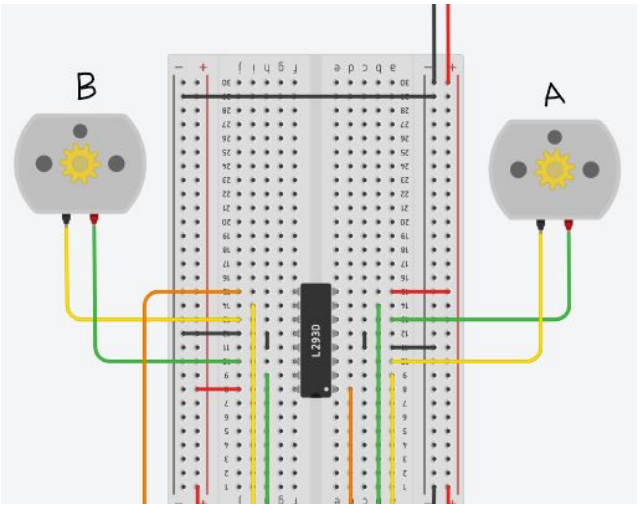
Pin Functions

PIN		TYPE	DESCRIPTION
NAME	NO.		
1,2EN	1	I	Enable driver channels 1 and 2 (active high input)
<1:4>A	2, 7, 10, 15	I	Driver inputs, noninverting
<1:4>Y	3, 6, 11, 14	O	Driver outputs
3,4EN	9	I	Enable driver channels 3 and 4 (active high input)
GROUND	4, 5, 12, 13	—	Device ground and heat sink pin. Connect to printed-circuit-board ground plane with multiple solid vias
V_{CC1}	16	—	5-V supply for internal logic translation
V_{CC2}	8	—	Power VCC for drivers 4.5 V to 36 V

Arduino Pin	**L298N Pin (and pin function)**
D5	1,2EN (Enable for Motor A)
D6	3,4EN (Enable for Motor B)
A0 (D14)	1A (Direction pin for Motor A)
A1 (D15)	2A (Direction pin for Motor A)
A2 (D16)	3A (Direction pin for Motor B)
A3 (D17)	4A (Direction pin for Motor B)

c. L293D and Motor Connections

The next step is to connect motors to the L293D driver. To do this, connect Motor A to the driver's 1Y and 2Y pins, and connect Motor B to the driver's 3Y and 4Y pins.



d. Supply Voltage and Ground Connections

A 9V battery will serve as the main supply voltage of the system with its negative terminal as the ground. For this section, it is very important to check the datasheet of the components to make sure that the components work properly and to prevent them from breaking down. Shown below are the recommended operating conditions for both the L293D and Arduino Uno.

6.3 Recommended Operating Conditions

over operating free-air temperature range (unless otherwise noted)

			MIN	NOM	MAX	UNIT
	Supply voltage	V_{CC1}	4.5		7	V
		V_{CC2}	V_{CC1}		36	
V_{IH}	High-level input voltage	$V_{CC1} \leq 7$ V	2.3		V_{CC1}	V
		$V_{CC1} \geq 7$ V	2.3		7	V
V_{IL}	Low-level output voltage		–0.3(1)		1.5	V
T_A	Operating free-air temperature		0		70	°C

(1) The algebraic convention, in which the least positive (most negative) designated minimum, is used in this data sheet for logic voltage levels.

Recommended Operating Conditions for the L293D Driver

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V

Recommended Operating Conditions for the Arduino Uno

Based on the L293D's datasheet, V_{CC1} must be between 4.5V and 7V. With that,it must be connected to the Arduino's 5V pin. For V_{CC2}, it must be between V_{CC1} and 36V. With that, we can connect it directly to the 9V source. We then connect the ground pins of the L293D to the system's ground.

The 9V battery will be used to power the Arduino Uno since its recommended operating supply voltage is 7V-12V. To do so, we connect the positive terminal of the battery to the Arduino Uno's VIN pin, and the battery's negative terminal to the ground pin of the Arduino.

3. **Writing Code**

   a. **Setting up pins operating modes (using pinMode(pin, output/input))**
   To make our code more readable we choose to define aliases for output pins. We then set whether each pin is an output or input using pinMode(). We do this setup in the setup() function.

```
#define L_speed_pin 5 //D5
#define R_speed_pin 6 //D6
#define L_forward 14  //A0
#define L_backward 15 //A1
#define R_forward 16  //A2
#define R_backward 17 //A3
#define constant_speed 100

void setup(){
  //Setting up pinmodes

  pinMode(L_speed_pin, OUTPUT);
  pinMode(R_speed_pin, OUTPUT);
  pinMode(L_forward, OUTPUT);
  pinMode(L_backward, OUTPUT);
  pinMode(R_forward, OUTPUT);
  pinMode(R_backward, OUTPUT);

}
```

   b. **Operating the control pins and running the motors**
      i. Inside the loop() function we code most functions and routines to control our pins. Our Arduino will run this code over and over while it's powered.
      ii. First we set the direction of our motors by Writing High and Low to the appropriate pins. Then we write a PWM signal to our speed pins, to do this we use analogWrite(pin, 0-255). Note that this only works for PWM capable pins

```
void loop(){

  digitalWrite(L_forward, HIGH);
  digitalWrite(L_backward, LOW);
  digitalWrite(R_forward, LOW);
  digitalWrite(R_backward, HIGH);

  analogWrite(L_speed_pin, constant_speed);
  analogWrite(R_speed_pin, constant_speed);
}
```

iii. Note that in the code, the motors were programmed to have opposite rotational direction (i.e. one motor will move CCW while the other moves CW). This is done to ensure that the robot will move forward or backward in the actual implementation. If both motors have the same rotational direction after programming, the robot will fail to move forward or backward. To solve this, you change the motor connections or you can change the High and Low settings in the code.

4. Conducting the Exercise

a. To find the relationship between output voltage and our PWM input signals, fill out this table. You can use the multimeter to probe the L293D output pins (<1:4>Y).

analogWrite (speed_pin,value)	**Voltage output**	**Speed ranking (1st , 2nd, 3rd etc.)**
analogWrite(0)		
analogWrite(64)		
analogWrite(127)		
analogWrite(191)		
analogWrite(255)		

b. To help you, here is a figure of how the Value in ***analogWrite(pin, Value)*** affects your pin's output