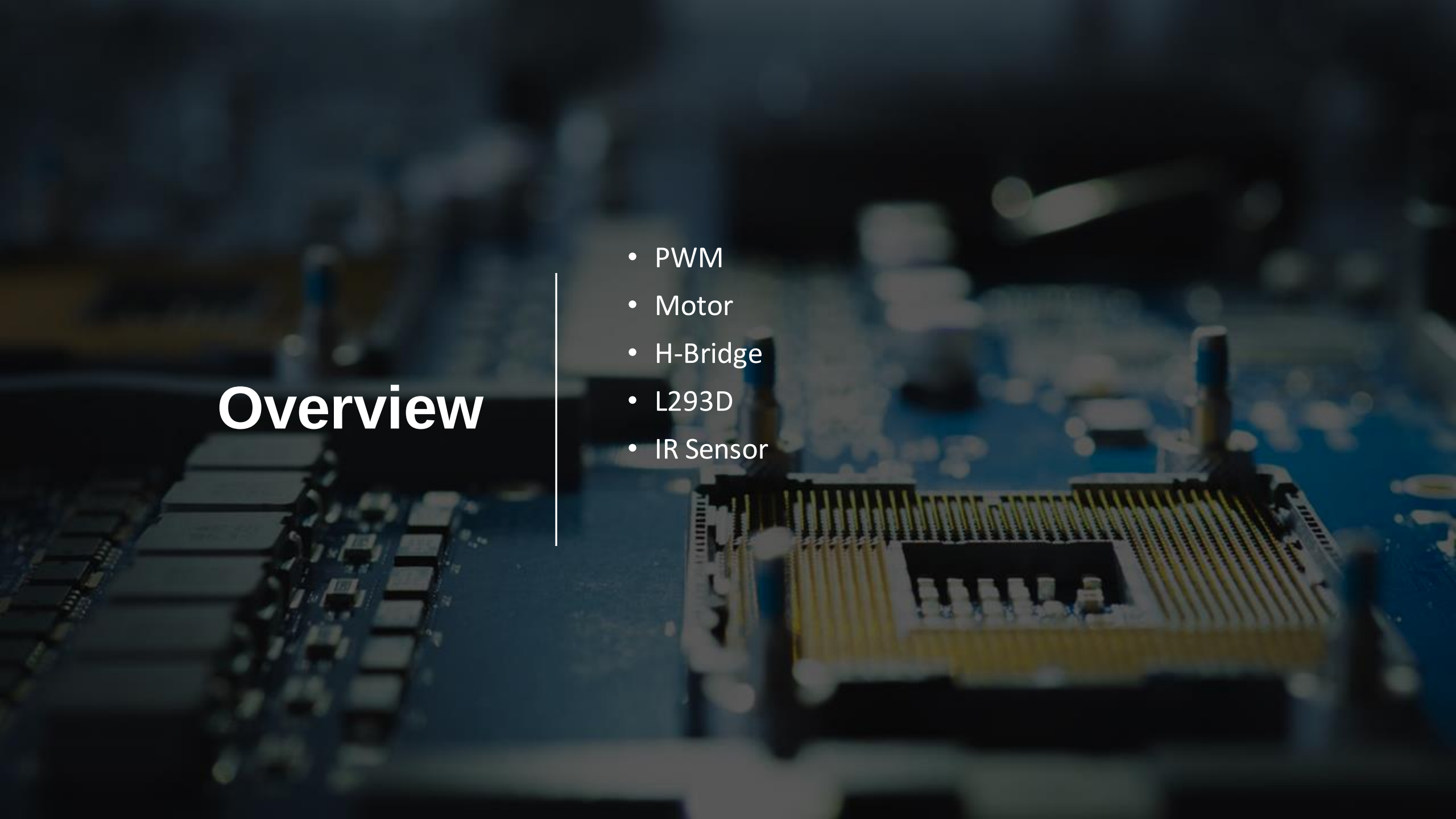
ROBOTICS CLUB
TRIBHUVAN UNIVERSITY
PULCHOWK CAMPUS

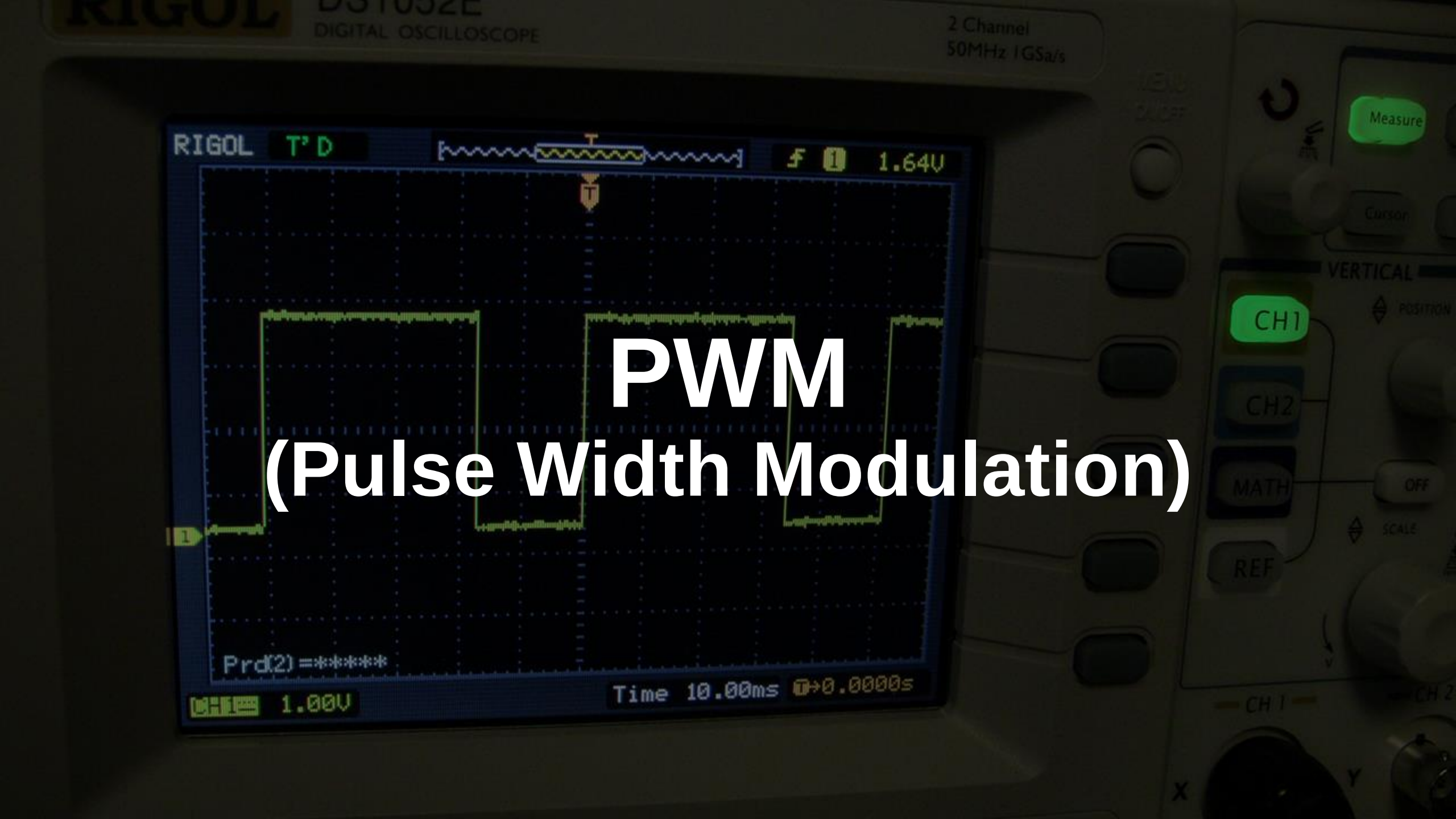
LOCUS 2021
18th National Technological Festival

# Hardware Fellowship

# Overview

- PWM
- Motor
- H-Bridge
- L293D
- IR Sensor






# PWM
# (Pulse Width Modulation)

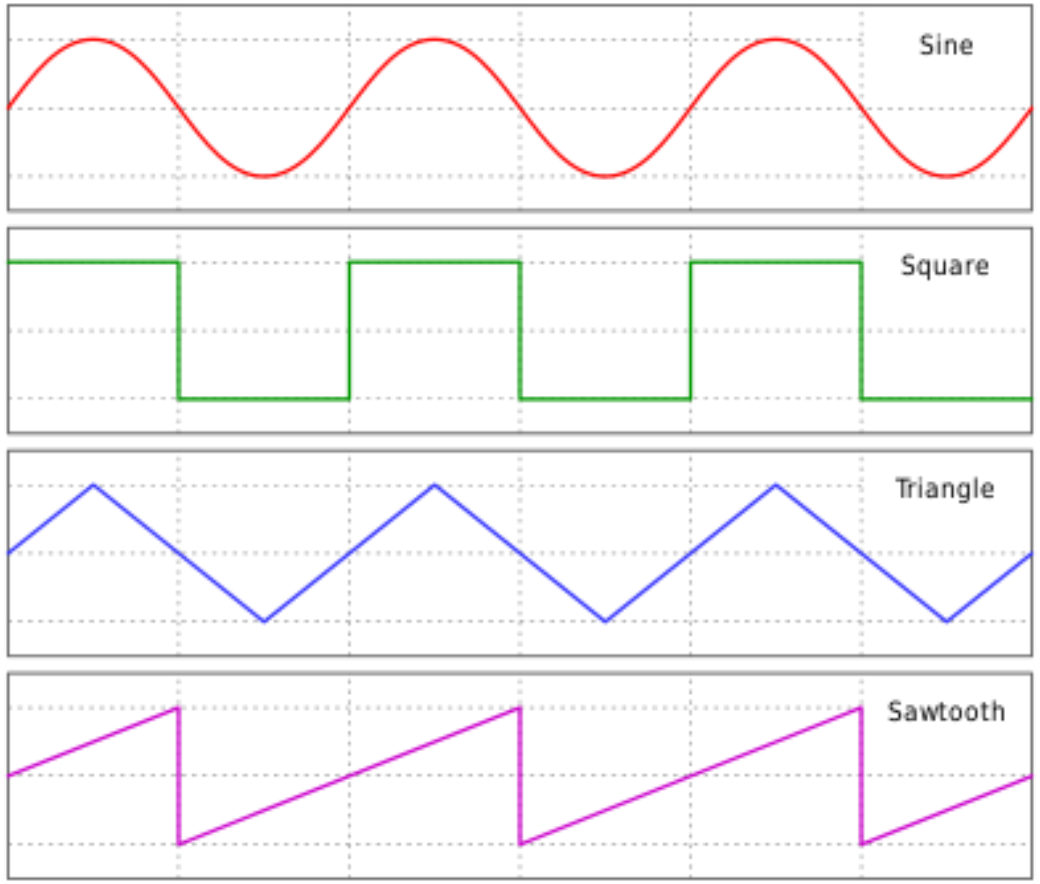
# What basically do you understand by Pulse?

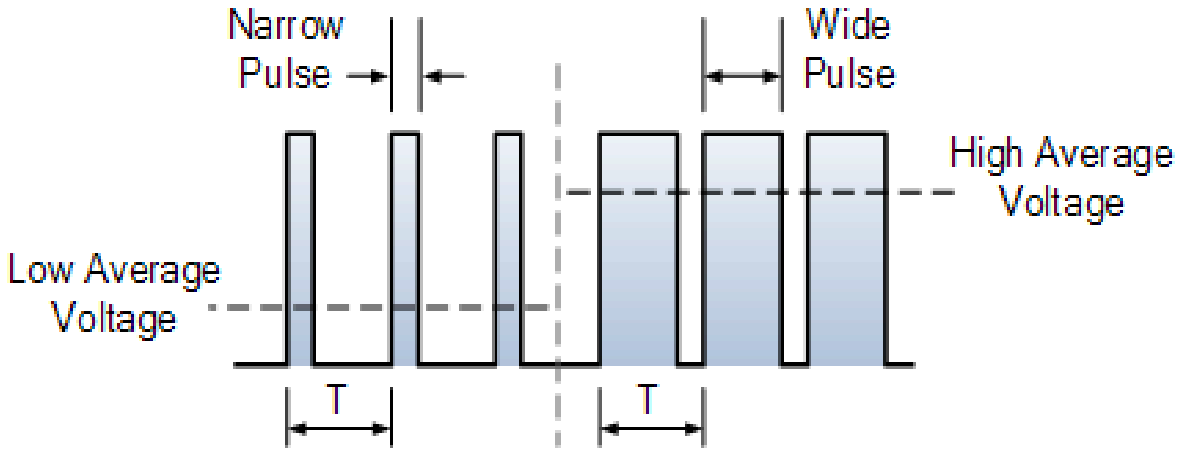
- Electronic Pulse is nothing but a graphical representation of the nature of the flow of current in an electrical or electronic circuit.

- Simply, it is a pulse is a burst of current, voltage or electromagnetic-field energy.

- Digital pulses usually have well-defined shapes (like square and triangular) which can be easily observed in an oscilloscope.

# Different Types of Pulses



# What do you mean by PWM?

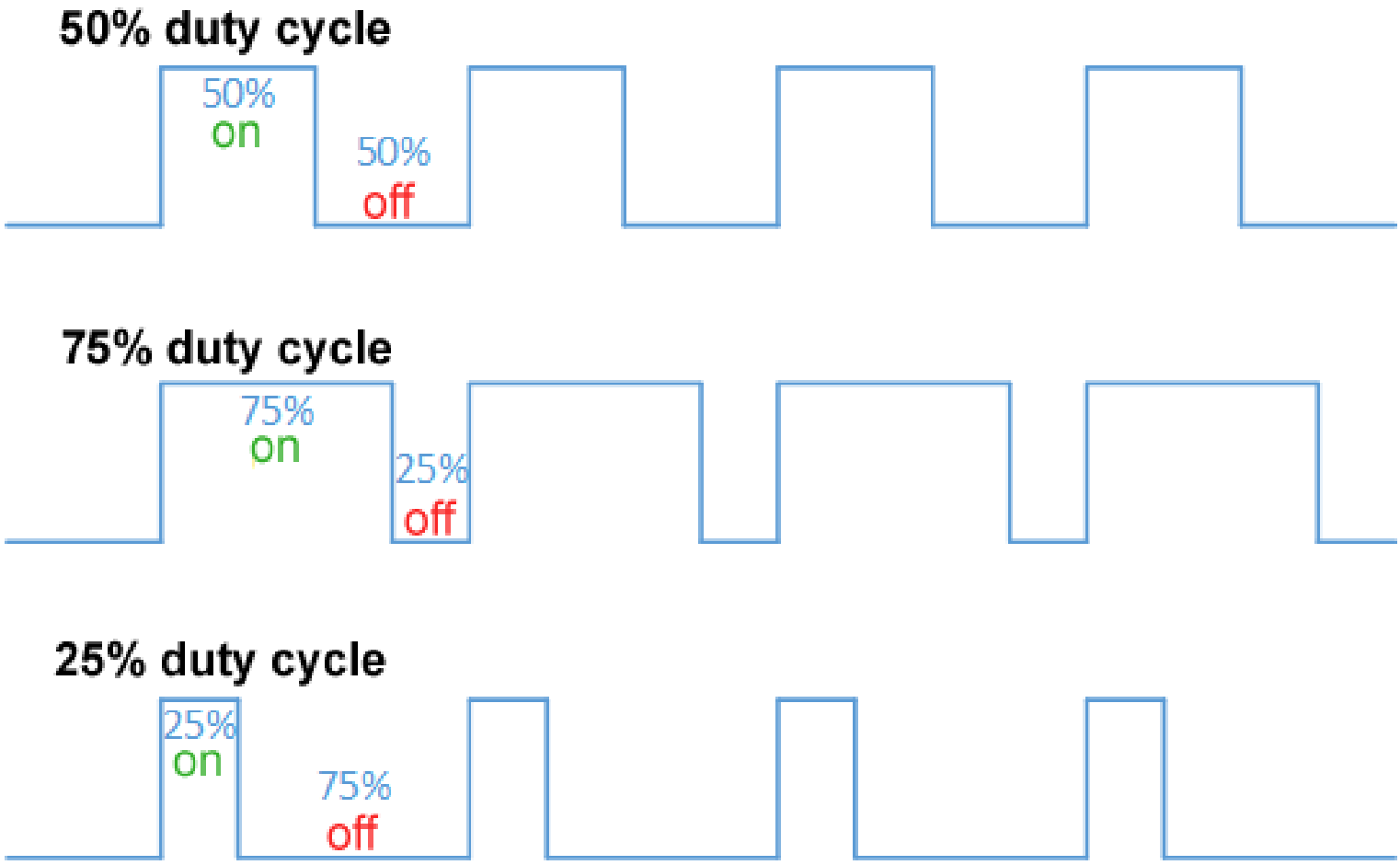
- Pulse width modulation (PWM) provides a way of controlling certain analog quantities, by varying the pulse width of a fixed frequency having waveform.
- Pulse Width Modulation (PWM) uses digital signals varying is usage from normal communication to the control power applications.
- So, what are the factors that impact on the PWM's behavior :

    1. Duty cycle.
    2. Frequency of a PWM signal.

# Duty Cycle in PWM

$$Duty\ Cycle\ (\%) = \frac{TON}{TON + TOFF}\ X\ 100$$

- It is commonly expressed as a percentage or a ratio.
- Duty cycle of 50% means pulse is high about half of its total time period.
- So, what does 100% duty cycle means?



# Frequency of PWM signal

- A period equal to the time this signal completes a one-on-off cycle.

$$f = \frac{1}{T}$$

f = frequency

# Applications of PWM:

1. It is used to control the speed of the motors in Industries.
2. It is used for dimming of the light.
3. It is used in communication system(encoding of messages).
4. It is used in audio system like making buzzer with different loudness.
5. Using the PWM technique and the switching mechanism we can control analog signals using the digital signal

# PWM in Arduino

PWM can be implemented in various ways on Arduino.

PWM output can be generated by specific pins in the uno R3 board **(i.e. pin 3, 5, 6, 9, 10, 11 )** .

With **analogWrite()** function,calling the AnalogWrite() function allows a stable square wave with a specified duty cycle to be generated on the PWM pins.

- The frequency of PWM signal on pins 5 and 6 will be about 980Hz and on other pins will be 490Hz.
- The output voltage from Arduino pins are 5V, and different duty cycles output different voltage levels as seen in the table.

| Duty Cycle | Average Voltage Levels |
|---|---|
| 0% | 0V |
| 25% | 1.25V |
| 50% | 2.5V |
| 75% | 3.75V |
| 100% | 5V |

# The PWM pins are labeled with ~ (tide) sign.


# analogWrite()

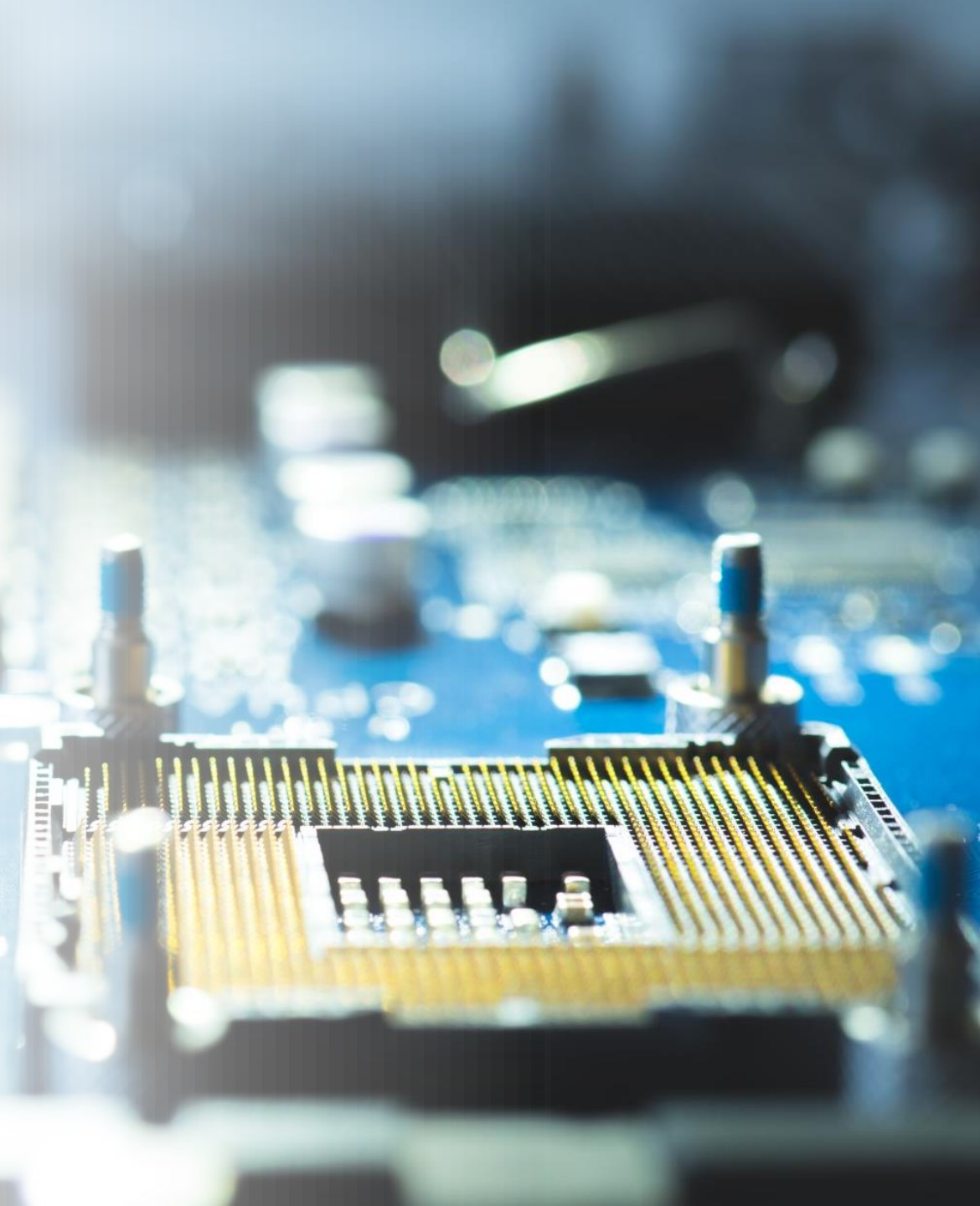
- Syntax:

analogWrite(pin, value)

Here, pin represents the pwm pin number in the microcontroller.

Value is the value ranging from 0 to 255.

- It writes an analog value (PWM wave) to a pin.

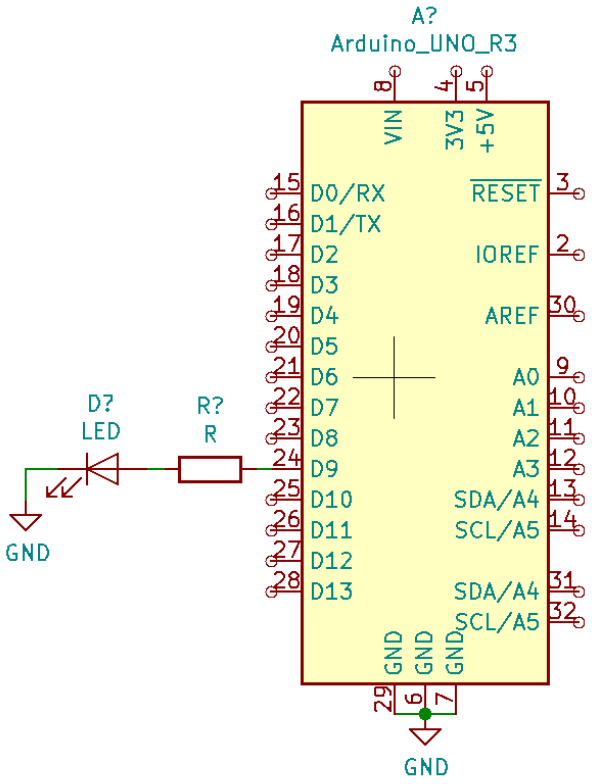


`analogWrite(pin,0-255)`

VS

`digitalWrite(pin,LOW-HIGH)`

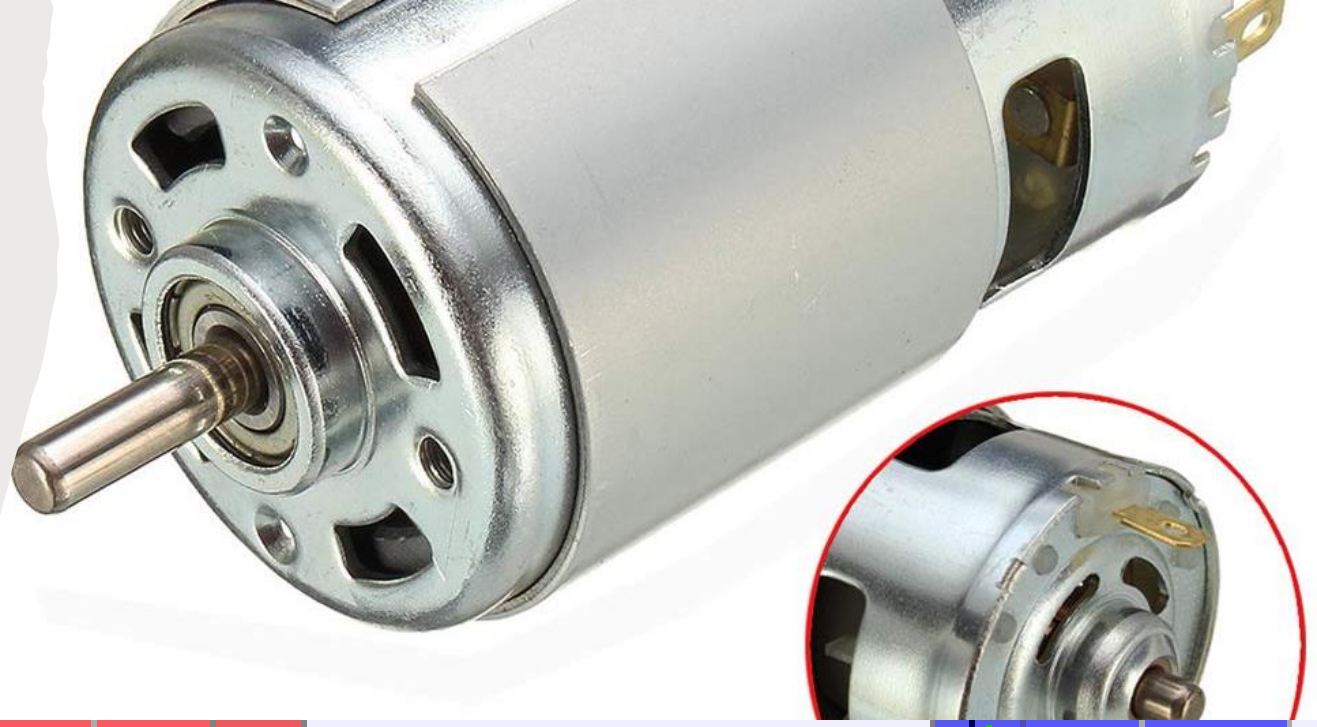
# Light Dimmer using PWM

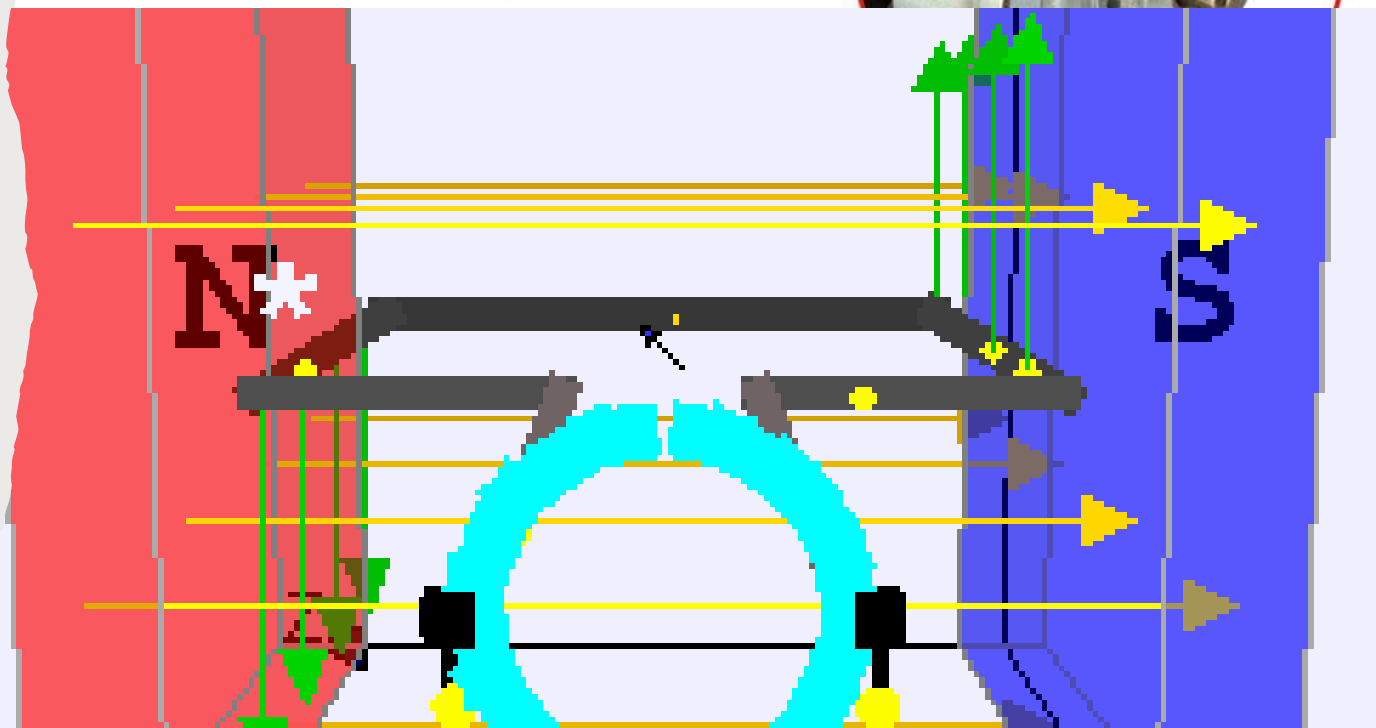


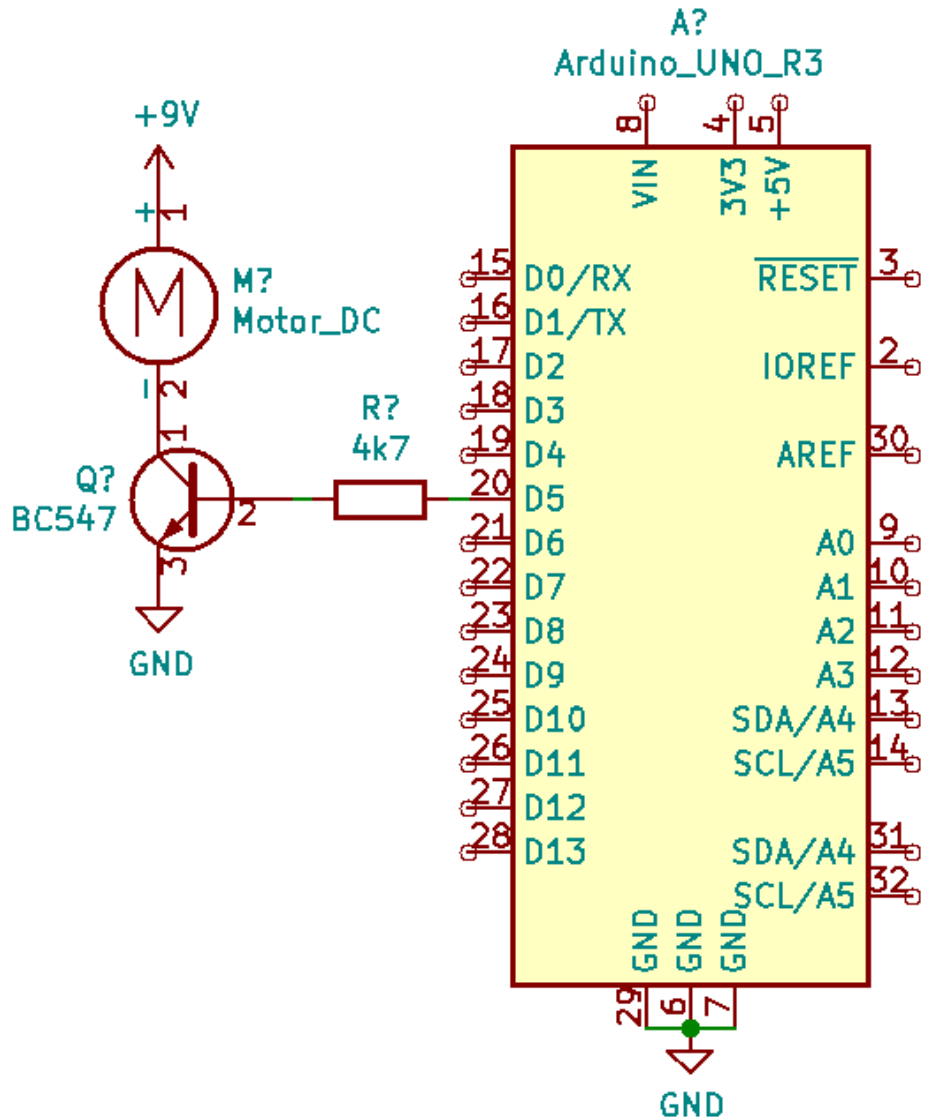

```
int led = 9;           // the pin that the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup()  {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop()  {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

# DC MOTOR

A rotatory electrical motor that converts direct current(DC) electrical energy into mechanical energy.

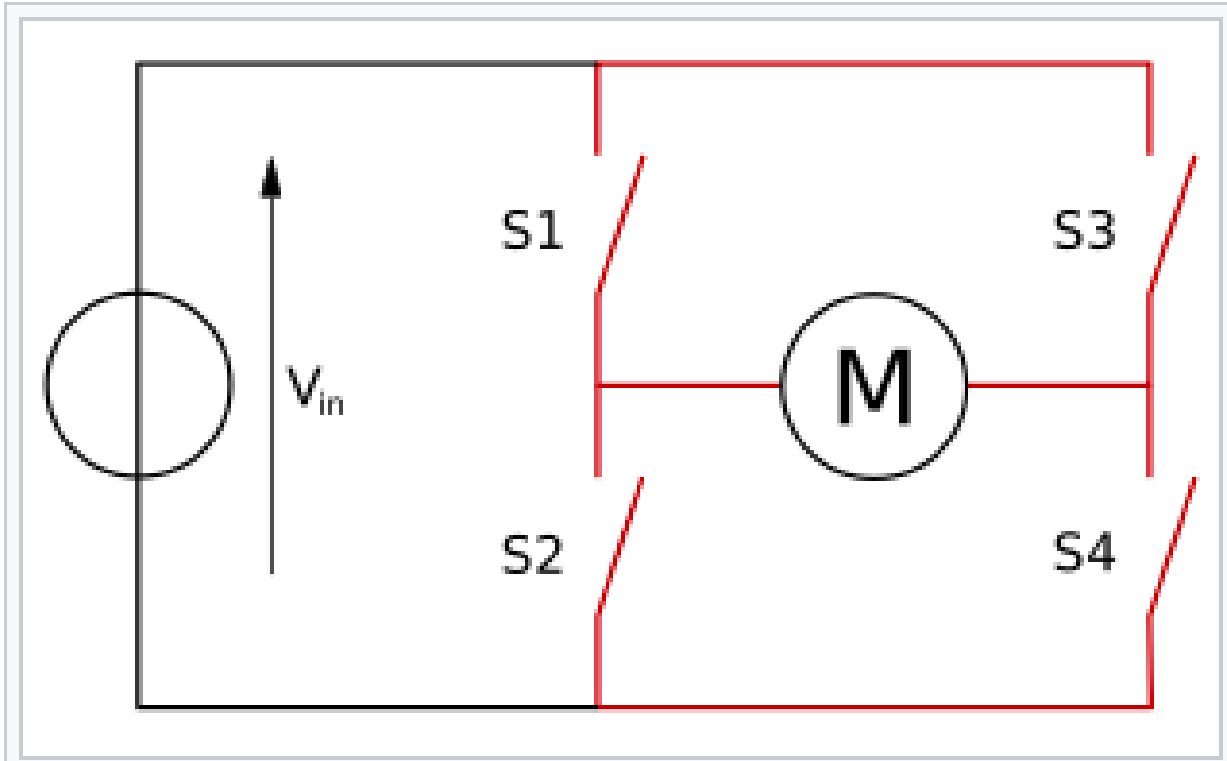





A?
Arduino_UNO_R3
+9V
M?
Motor_DC
R?
4k7
Q?
BC547
GND
VIN
3V3
+5V
D0/RX
D1/TX
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
RESET
IOREF
AREF
A0
A1
A2
A3
SDA/A4
SCL/A5
SDA/A4
SCL/A5
GND
GND
GND
GND

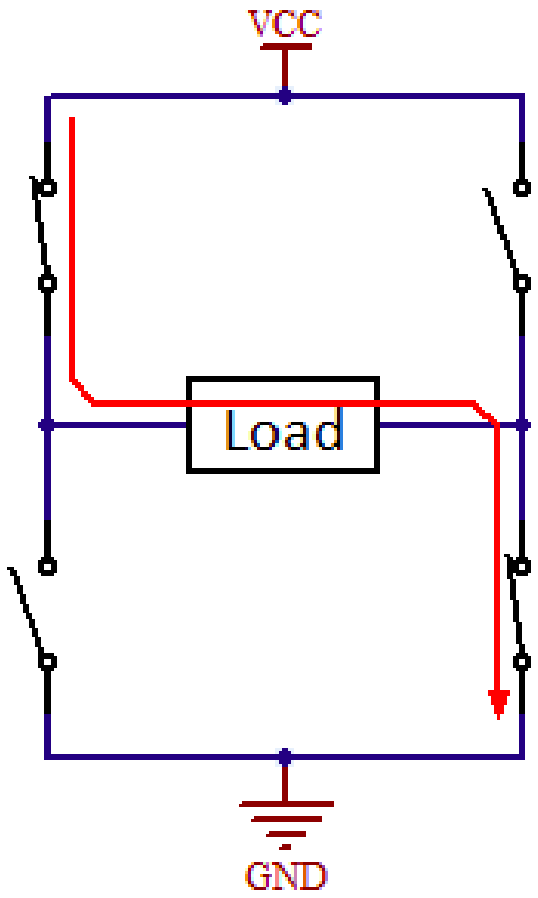
# H-Bridge

- An **H-bridge** is an electronic circuit that switches the polarity of a voltage applied to a load. These circuits are often used in robotics and other applications to allow DC motors to run forwards or backwards

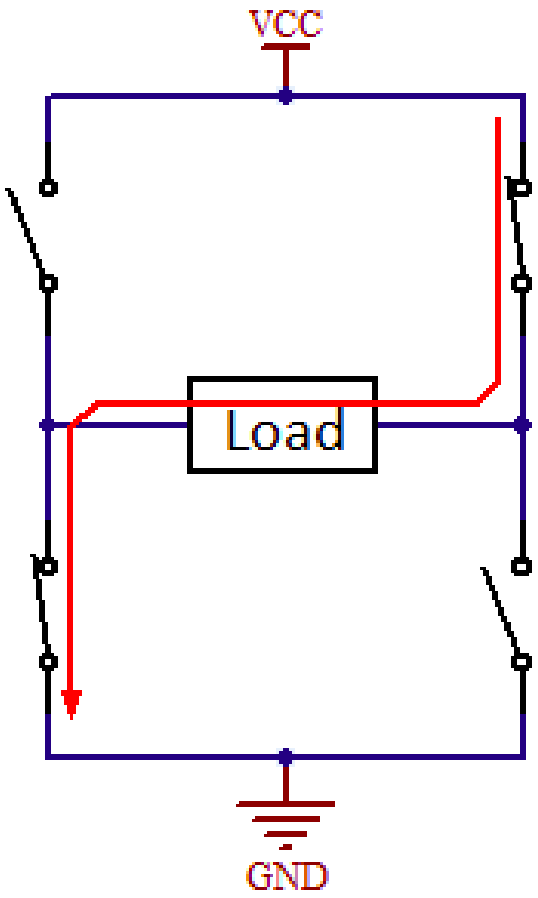


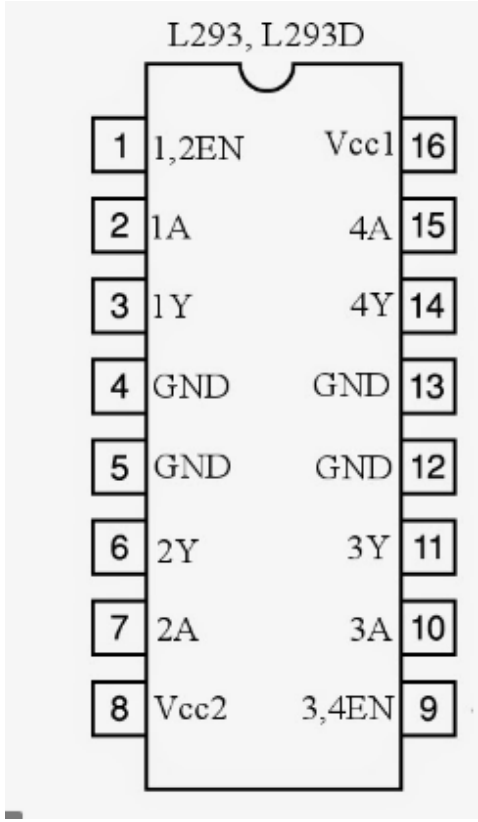

Structure of an H-bridge (highlighted in red)

Connecting the load
in one direction

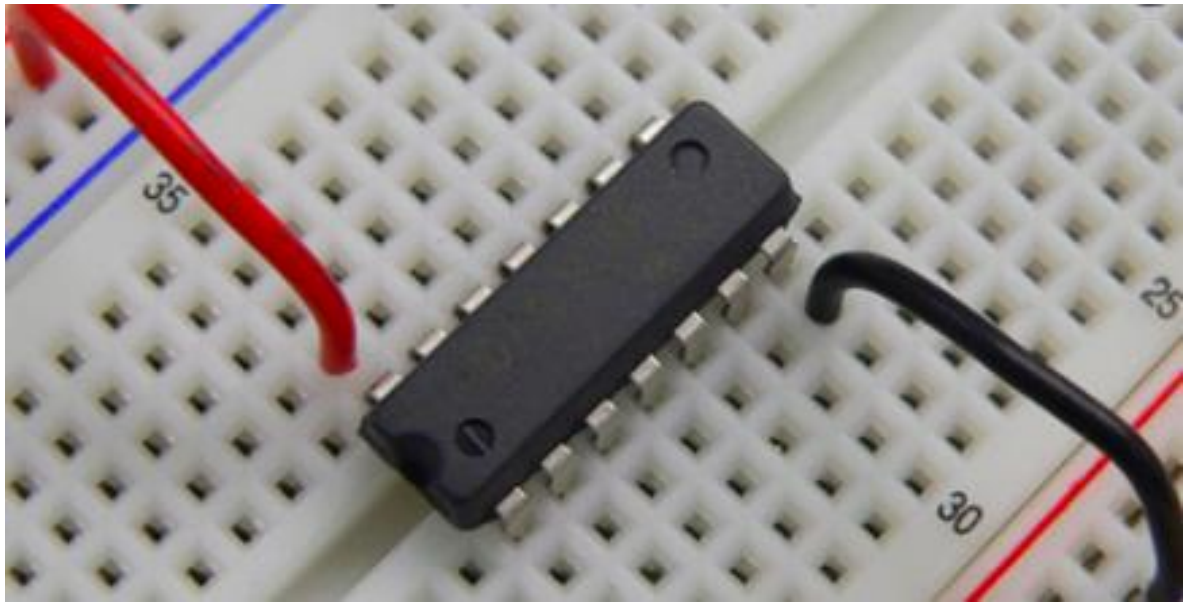



Connecting the load
in the other direction

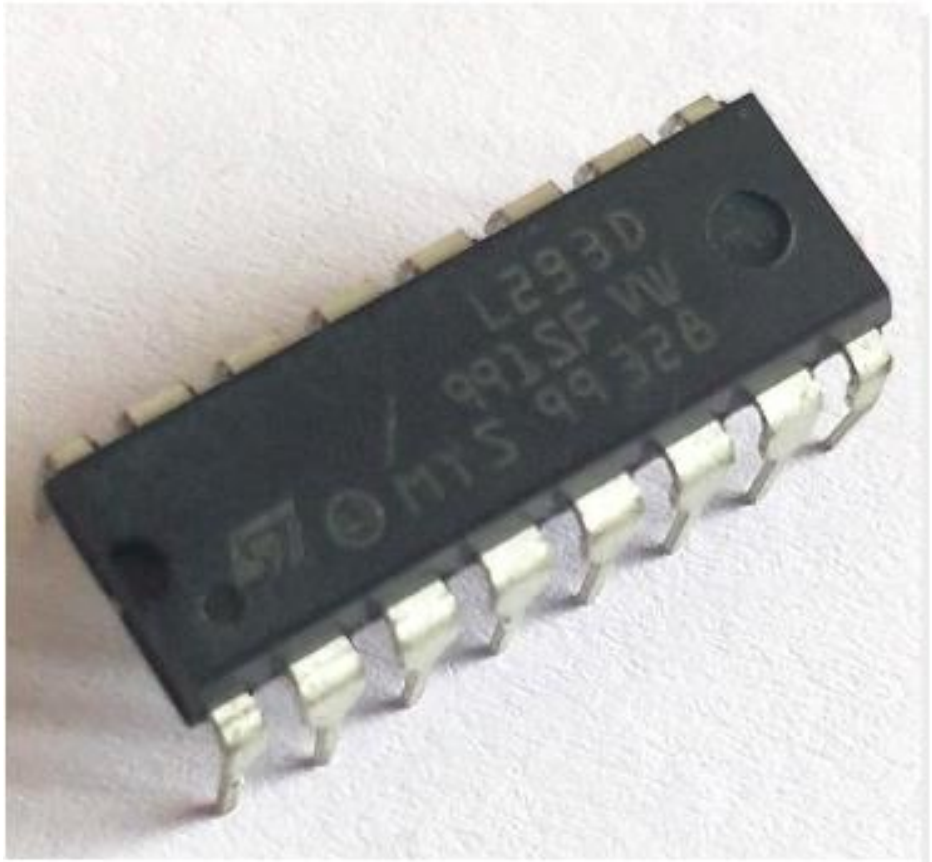




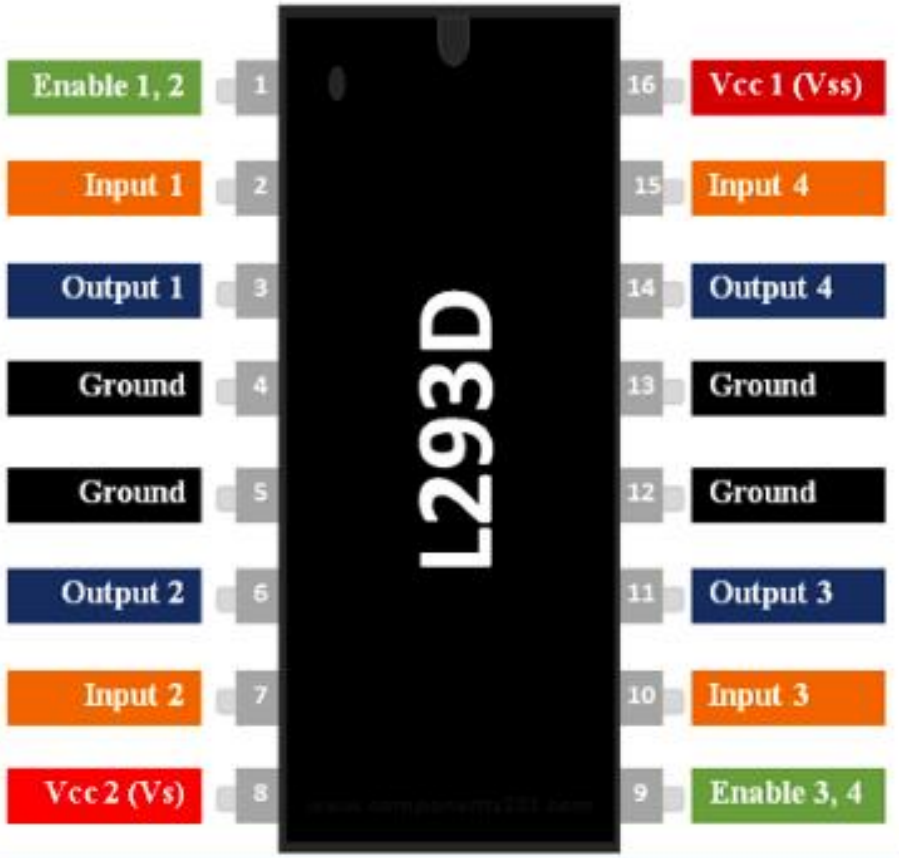

Reading IC pin out




# Using IC in bread board



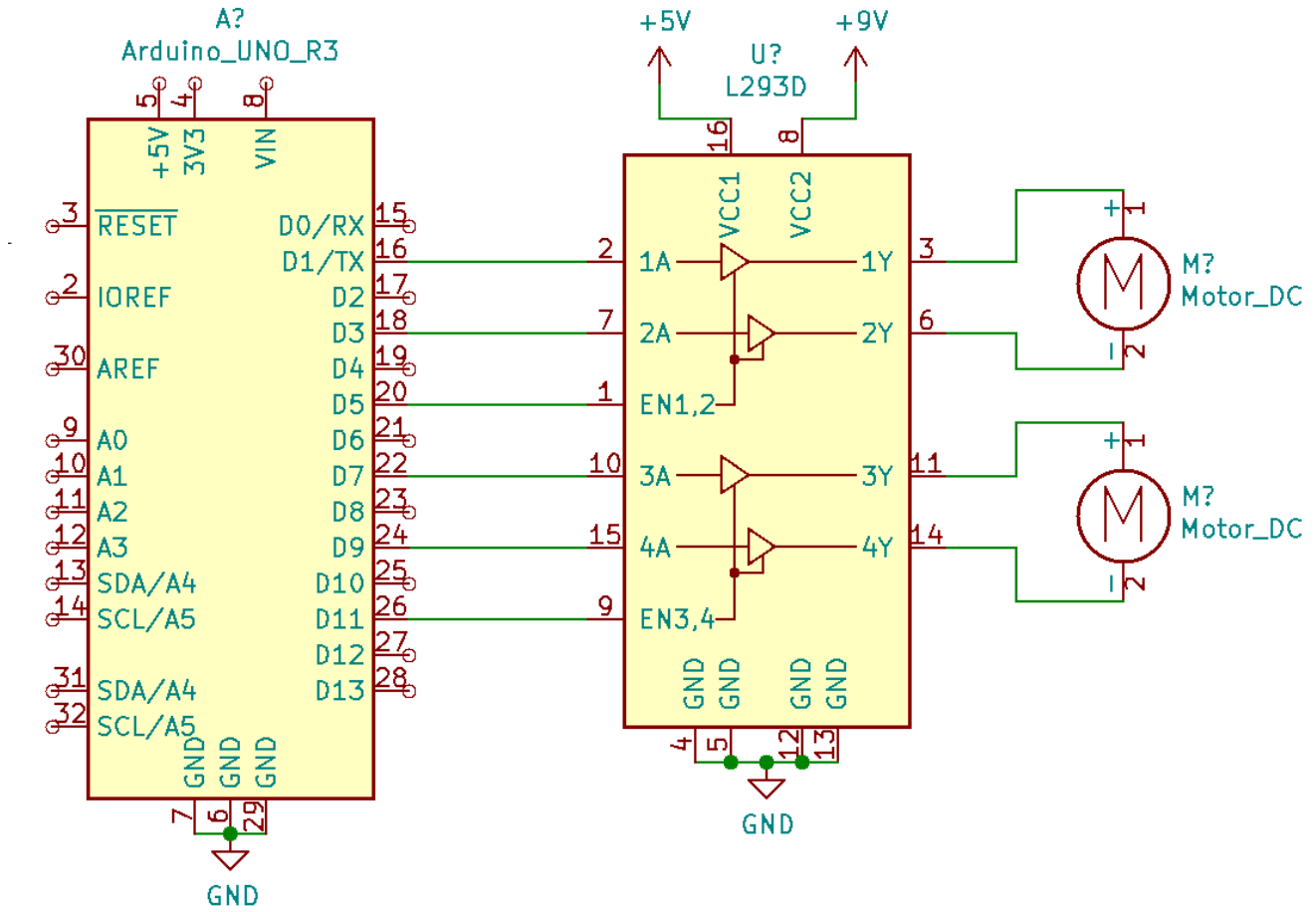

L293D Motor Driver IC




Motor Driver IC L293D Pinout

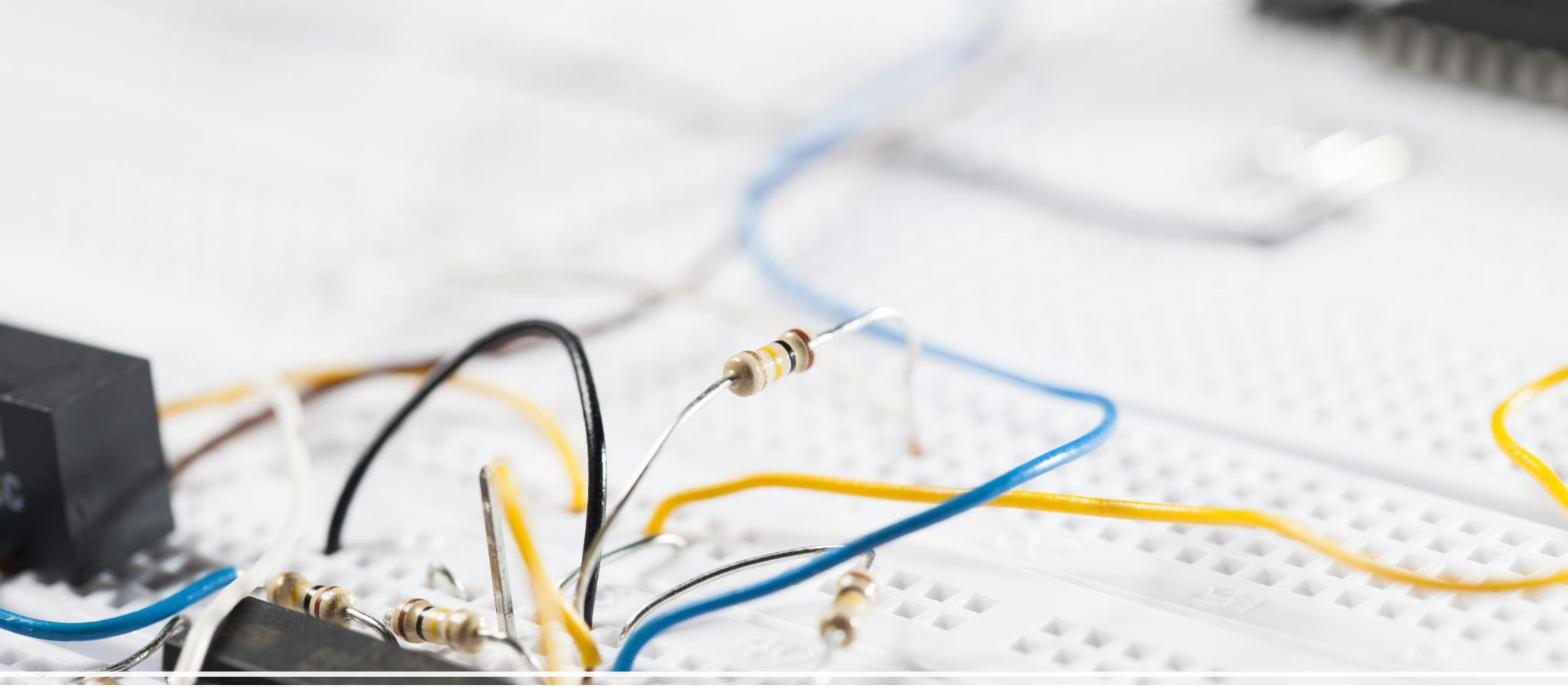
## L293D Pin Configuration

| Pin Number | Pin Name | Description |
|---|---|---|
| 1 | Enable 1,2 | This pin enables the input pin Input 1(2) and Input 2(7) |
| 2 | Input 1 | Directly controls the Output 1 pin. Controlled by digital circuits |
| 3 | Output 1 | Connected to one end of Motor 1 |
| 4 | Ground | Ground pins are connected to ground of circuit (0V) |
| 5 | Ground | Ground pins are connected to ground of circuit (0V) |
| 6 | Output 2 | Connected to another end of Motor 1 |
| 7 | Input 2 | Directly controls the Output 2 pin. Controlled by digital circuits |
| 8 | Vcc2 (Vs) | Connected to Voltage pin for running motors (4.5V to 36V) |
| 9 | Enable 3,4 | This pin enables the input pin Input 3(10) and Input 4(15) |
| 10 | Input 3 | Directly controls the Output 3 pin. Controlled by digital circuits |
| 11 | Output 3 | Connected to one end of Motor 2 |
| 12 | Ground | Ground pins are connected to ground of circuit (0V) |
| 13 | Ground | Ground pins are connected to ground of circuit (0V) |
| 14 | Output 4 | Connected to another end of Motor 2 |
| 15 | Input 4 | Directly controls the Output 4 pin. Controlled by digital circuits |
| 16 | Vcc2 (Vss) | Connected to +5V to enable IC function |




Note: EN1,2 and EN3,4 pins can be used as PWM pins to control the speed of motors

```
const int IN1 = 6, IN2 = 5;
const int enable12 = 3;
void setup()
{
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(enable12, OUTPUT);
  digitalWrite(enable12, HIGH);
}
void loop()
{
  clockwise();
  delay(5000);
  anticlockwise();
  delay(5000);
  brake();
  delay(5000);
}
void clockwise()
{
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
}
void anticlockwise()
{
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
}
void brake()
{
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
}
```

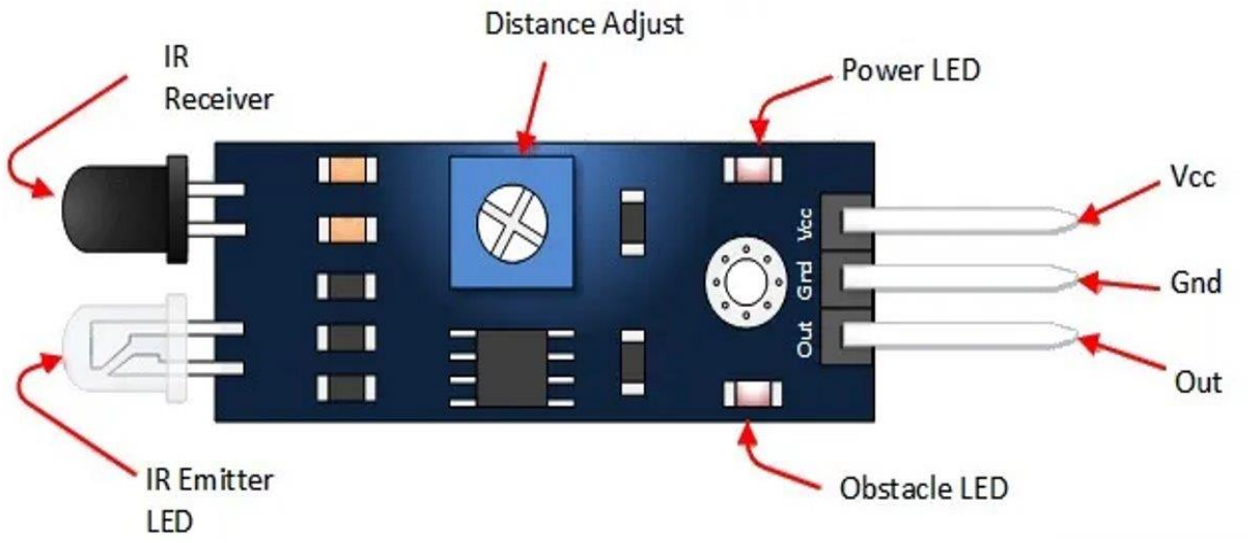


# IR sensors

# IR Sensor

- an electronic device, that emits in order to sense some aspects of the surroundings.
- It can measure the heat of an object as well as detects the motion.
- The emitter is simply an IR LED and the detector is simply an IR photodiode.[PhotoCouple]
- When IR light falls on the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received.

# IR SENSOR

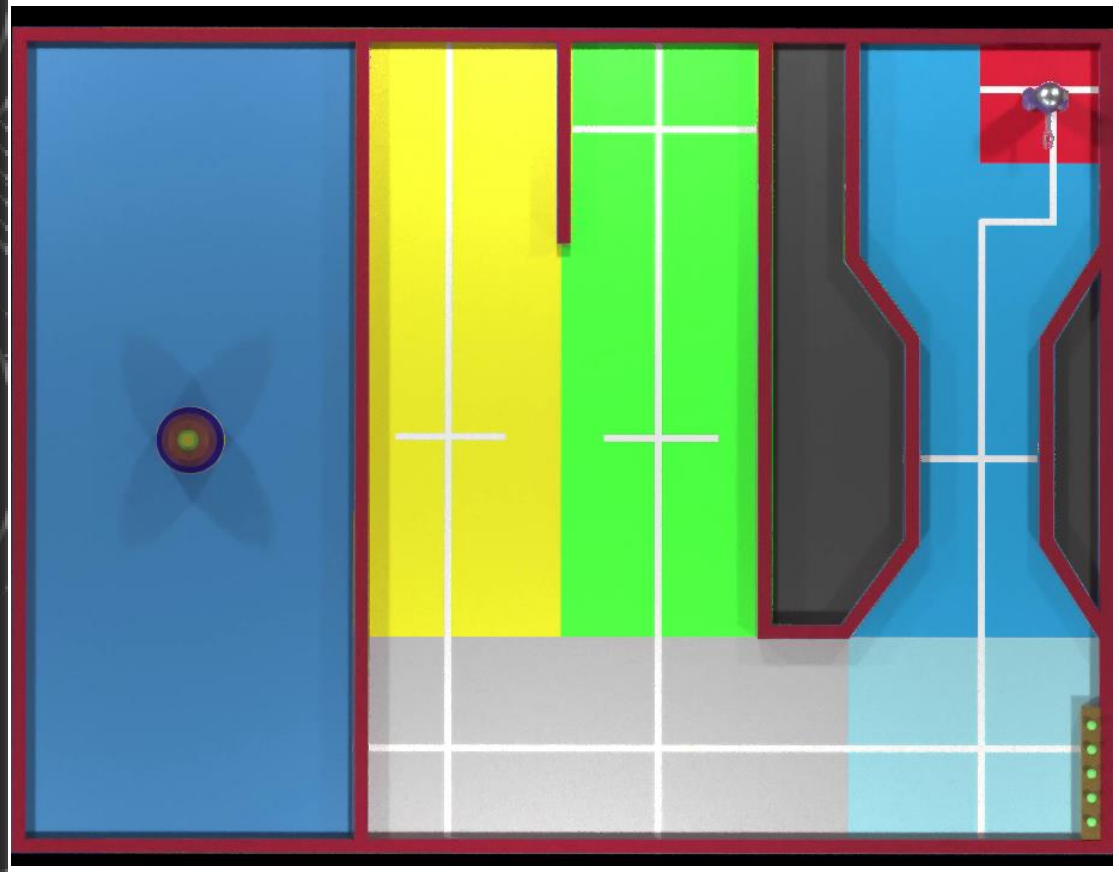



- VCC: 3.3V-5V power input pin
- GND: 0V power pin
- OUT: Digital Output Pin

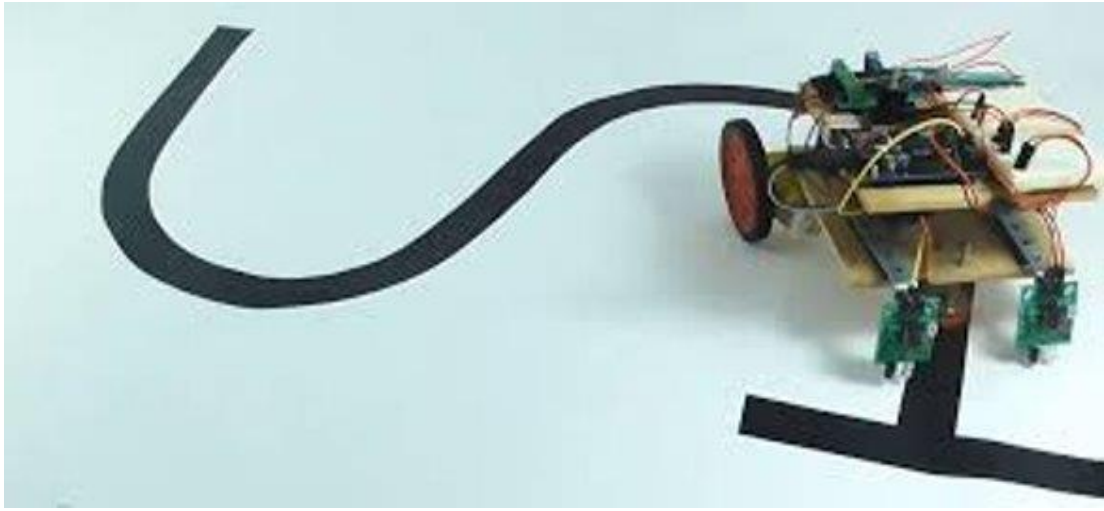
- Output of IR sensor is 'LOW' when there is obstacle closer than the threshold distance, else 'HIGH'
- Threshold distance can be adjusted by Distance Adjust

```
//Code for Collision detection
int LEDpin = 13;
int obstaclePin = 11;
int hasObstacle = LOW;  // LOW MEANS OBSTACLE

void setup() {
  pinMode(LEDpin, OUTPUT);
  pinMode(obstaclePin, INPUT);
  Serial.begin(9600);
}

void loop() {
  hasObstacle = digitalRead(obstaclePin);

  if (hasObstacle == LOW) {
    Serial.println("Stop something is ahead!!");
    digitalWrite(LEDpin, HIGH);
  }
  else {
    Serial.println("Path is clear");
    digitalWrite(LEDpin, LOW);
  }
  delay(200);
}
```

# Line Following Robot

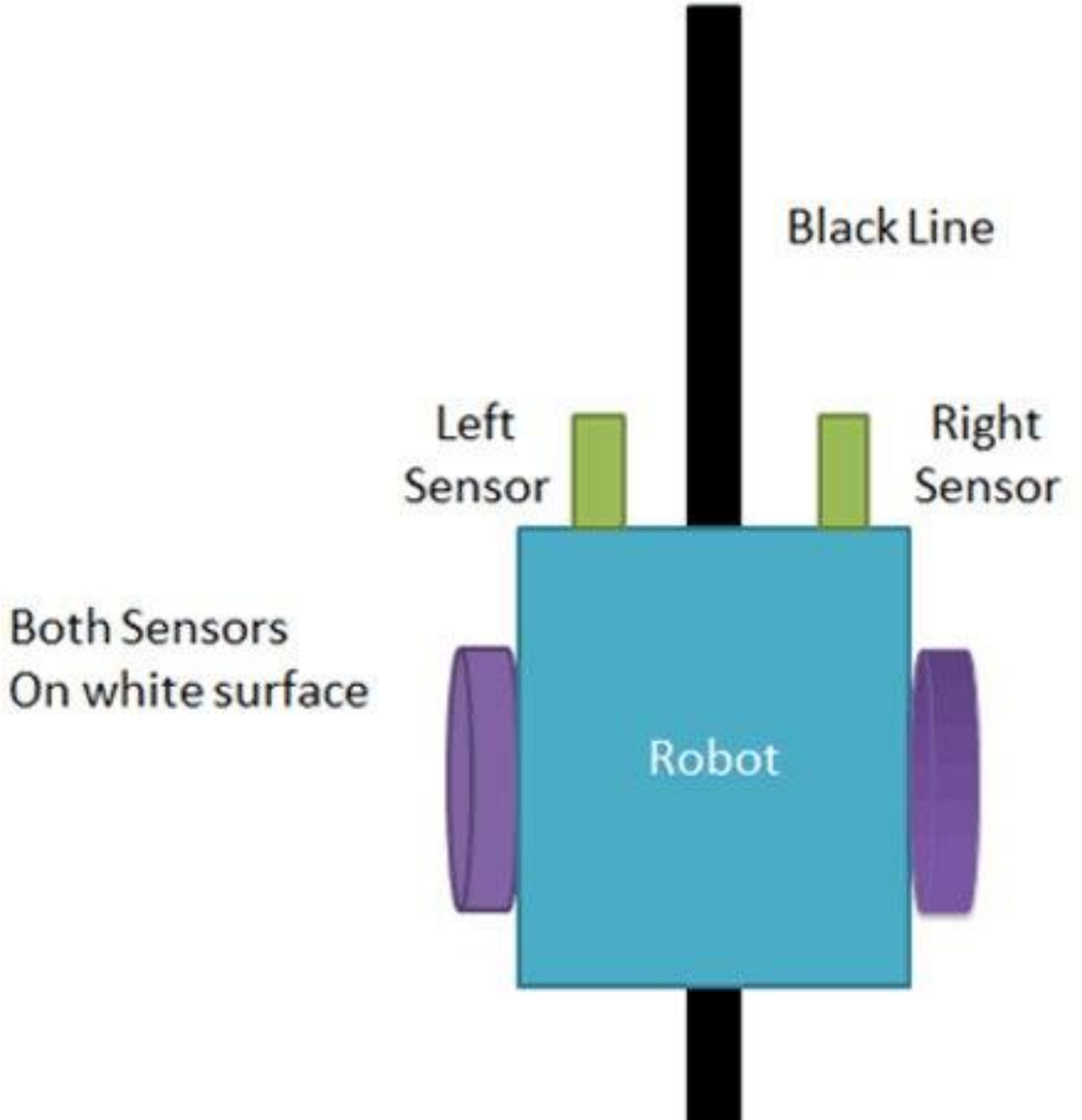


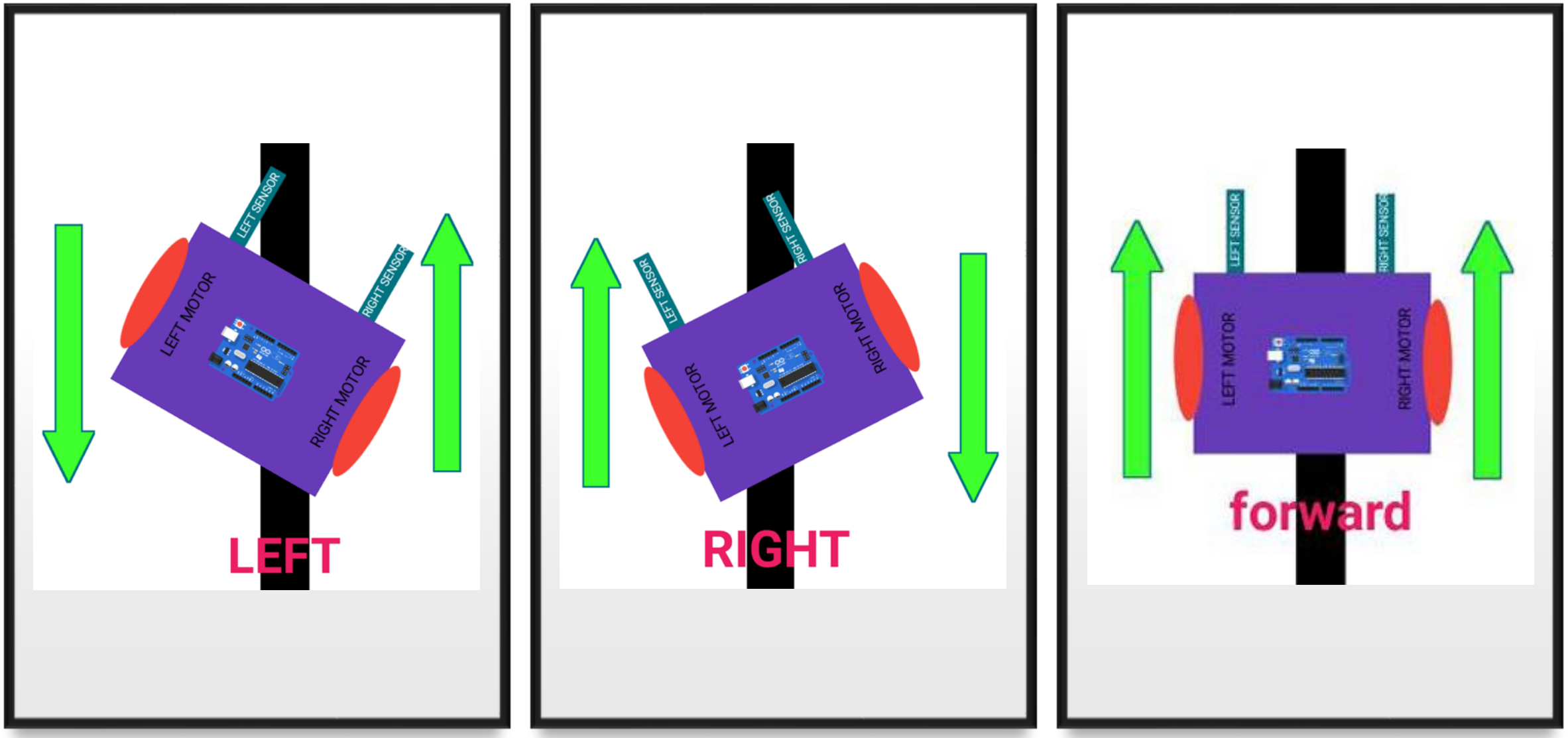
White line on a dark surface



Black line on a light surface

# Robot Configuration




LEFT SENSOR
RIGHT SENSOR
LEFT MOTOR
RIGHT MOTOR
LEFT
LEFT SENSOR
RIGHT SENSOR
LEFT MOTOR
RIGHT MOTOR
RIGHT
LEFT SENSOR
RIGHT SENSOR
LEFT MOTOR
RIGHT MOTOR
forward