SENSOR & ACTUATOR

Contents

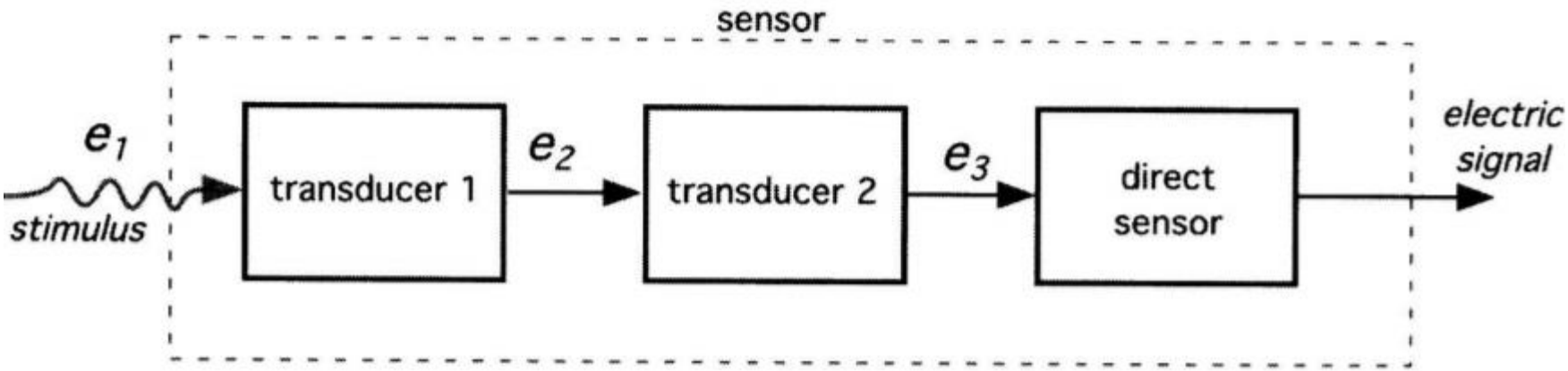
- Sensor
  - Distance measurement with Ultrasonic sensor
- Actuator
  - DC motor Control
  - Servo motor Control

Sensor (Cont.)

- Sensor is a device, module, or subsystem whose purpose is to detect events or changes in its environment and send the information to other electronics, mainly computer processor
- Sensors are devices converting any kind of physical attributes (temperature, luminance, force, acceleration etc.) to a understandable form for humans or machines.

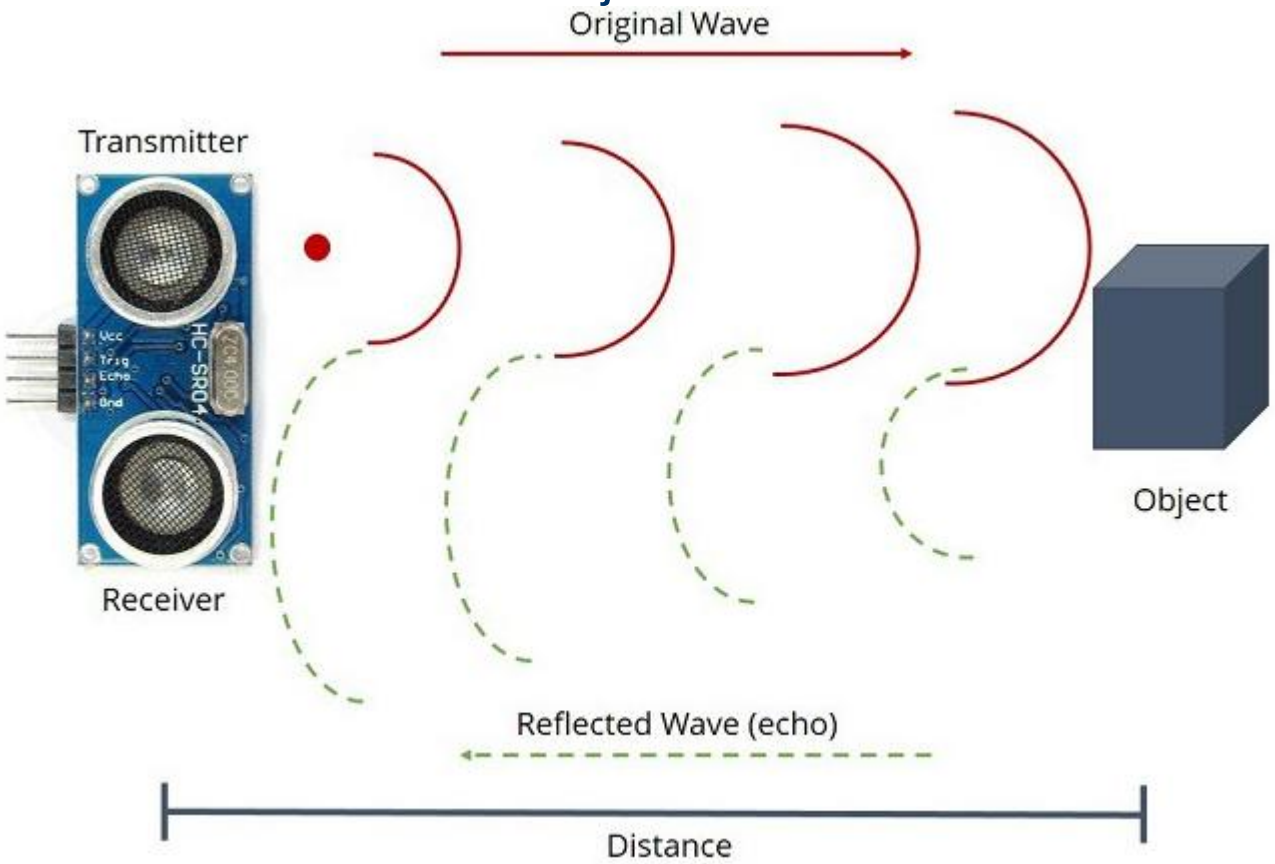
Sensor

- A sensor acquires a physical quantity and converts it into a signal suitable for processing
- Common sensors convert measurement of physical phenomena into an electrical signal
- Sensor is called a transducer which converts one form of energy to another
  - When input is a physical quantity and output electrical → Sensor
  - When input is electrical and output a physical quantity → Actuator



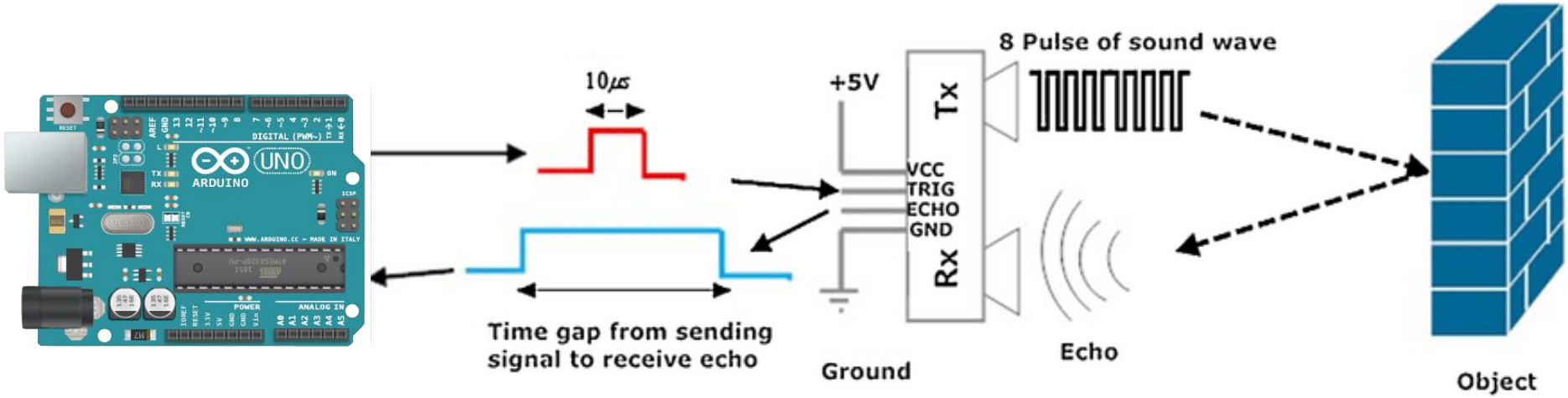
Ultrasonic Sensor (HC-SR06)

- Ultrasonic sensors generate high frequency sound waves and evaluate the echo which is reflected back to the sensor. A timing chip measures the time interval between transmitting the signal and receiving the echo to calculate the distance to the object

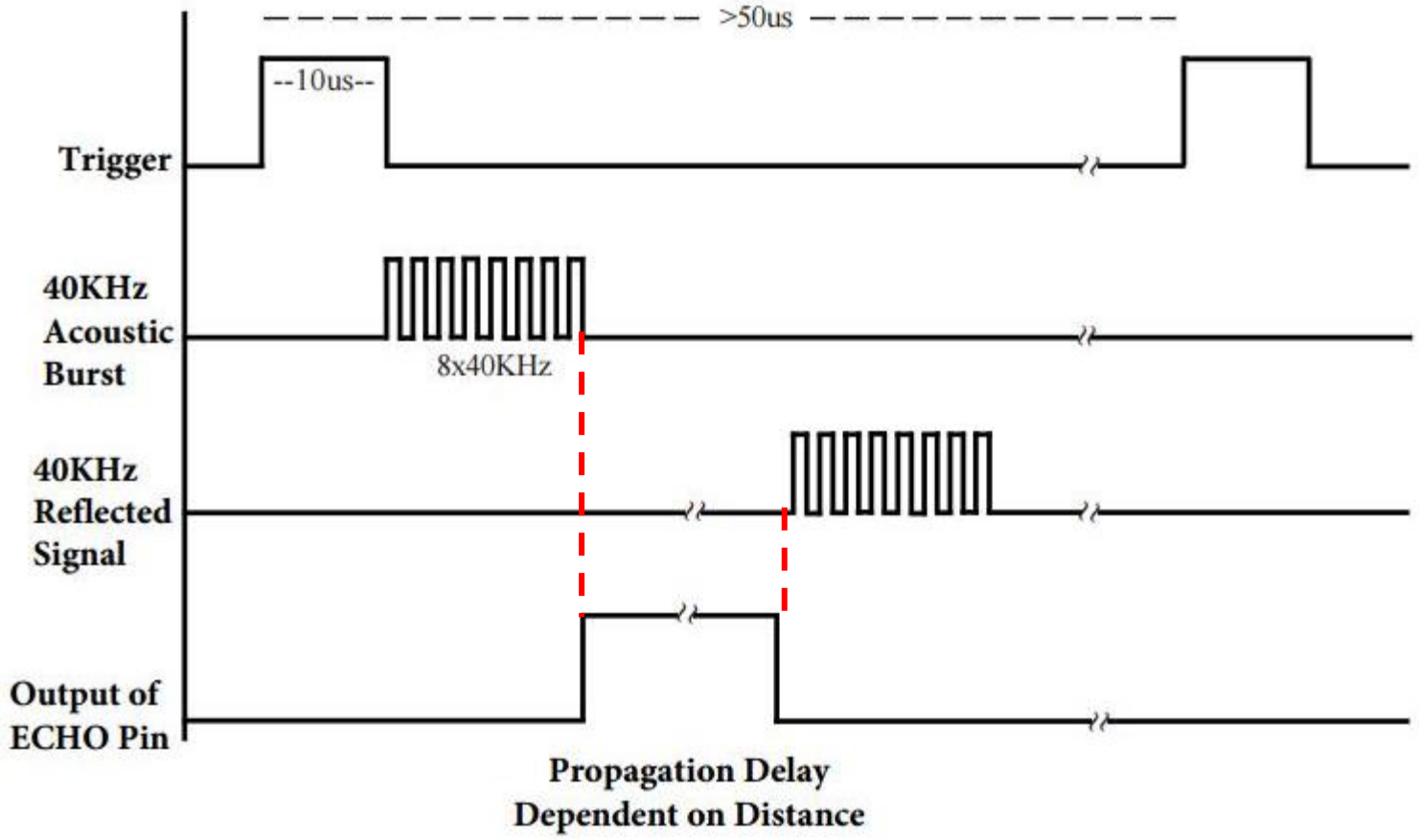


Principle of Operation

- The basic principle of operation:
  1. Send a trigger pulse for at least 10us high to the ultrasonic module
  2. The HC-SR04 module automatically sends eight 40 kHz and detect whether there is a pulse signal back
  3. If the signal back, ECHO output of the sensor will be in HIGH state(5V) for a duration of time taken for sending and receiving ultrasonic burst.



Timing Diagram



- Once ultrasonic burst is transmitted ECHO pin goes high and stays high until the wave returns back.
- To obtain the distance, measure the width of ECHO pin

Measuring the width of Pin

- pulseIn()
  - Reads a pulse (either HIGH or LOW) on a pin
  - Syntax
    - pulseIn(pin, value)
      - pin: the number of the pin on which you want to read the pulse. (int)
      - value: type of pulse to read: either HIGH or LOW. (int)
  - Returns (unsigned long)
    - the length of the pulse (in microseconds)
    - 0 if no pulse started before the timeout

If value is HIGH, pulseIn() waits for the pin to go from LOW to HIGH starts timing then waits for the pin to go LOW and stops timing.

Returns the length of the pulse in microseconds or

gives up and returns 0 if no complete pulse was received within the timeout

Distance Calculation

- To find the distance to the object

  : Distance = speed of sound(344m/s) **x** time taken **/** 2

  Distance in centimeters = Time/2 * 0.0344 (or Time / 58)

  Time = Width of Echo pulse, in us (micro second)

Speed of sound

: The speed of sound in air changes with temperature and humidity

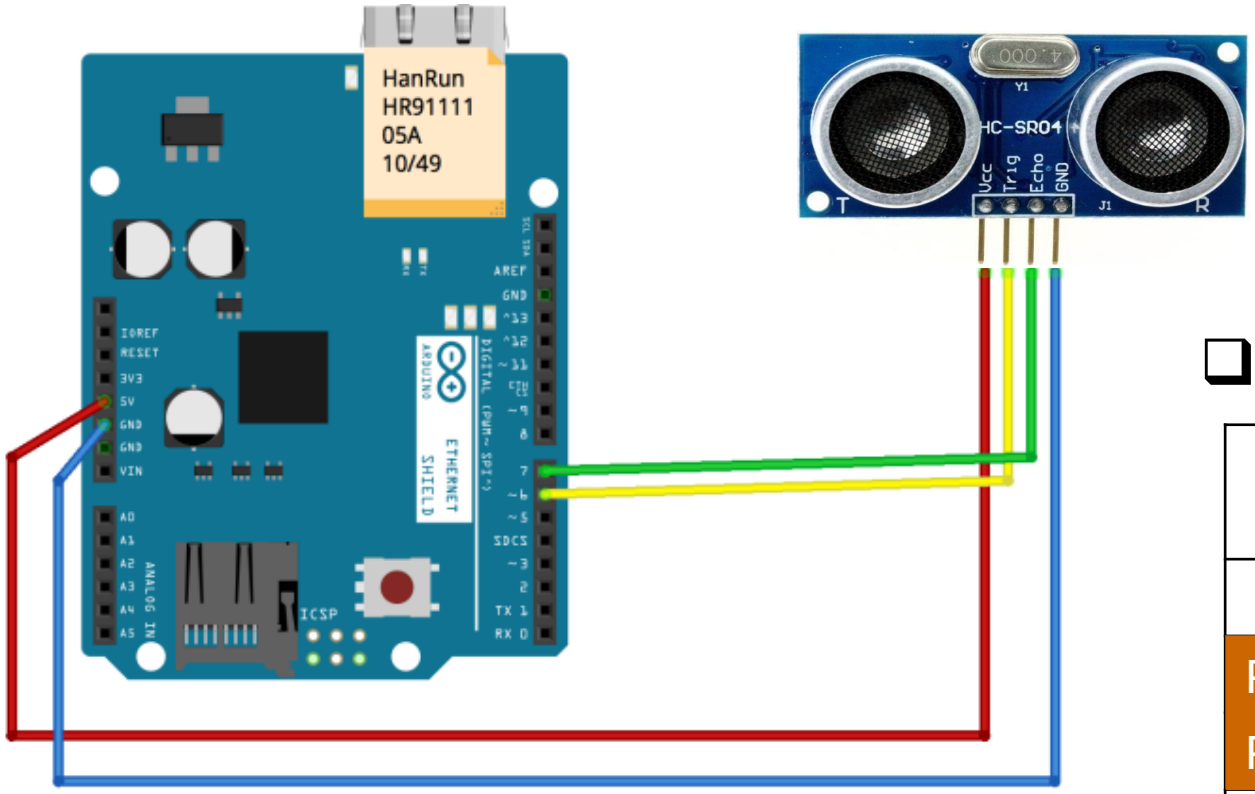
c(m/s) = 331.4 + (0.606 x T) + (0.0124 x H)

331.4: Speed of sound(in m/s) at 0℃ and 0 % humidity

T: Temperature in ℃

H: % huminity

Hardware Wiring

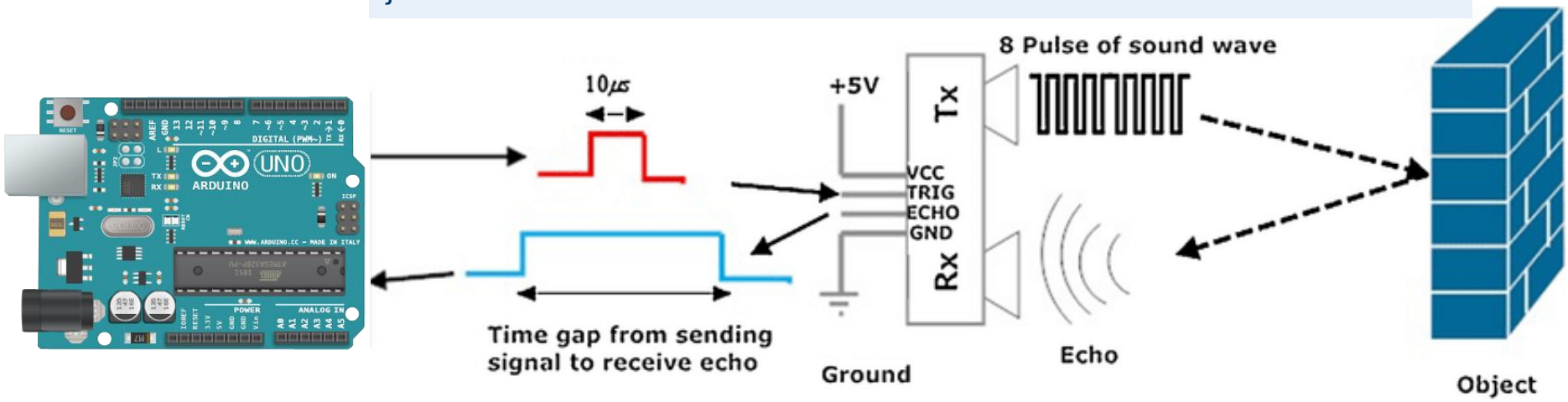


❑ Pin Connection

Arduino	Ultrasonic sensor
5V	VCC
Pin A4(Output)	Trig
Pin A5(Input)	Echo
GND	GND

Example Code


```
void setup() {
   pinMode(trigPin, OUTPUT);                    // sets the trigPin as output
   pinMode(echoPin, INPUT);                     // sets the echoPin as input
}
void loop() {
    float duration, distance;

    digitalWrite(trigPin, HIGH);                // the trigPin is at 5 volts
    delayMicroseconds(10);                      // 10us delay
    digitalWrite(trigPin, LOW);                 // the trigPin is at ground

   duration = pulseIn(echoPin, HIGH);           // read the width of Echo pin
   distance = (duration) / 58;                  // calculate the distance in cm
}
```



Example Code

```
#define trigPin   A4
#define echoPin A5

void setup() {
    Serial.begin(9600);                    // opens serial port, sets data rate to 9600 bps
    pinMode(trigPin, OUTPUT);              // sets the trigPin as output
    pinMode(echoPin, INPUT);               // sets the echoPin as input
}
void loop() {
     float duration, distance;

     digitalWrite(trigPin, HIGH);          // the trigPin is at 5 volts
     delayMicroseconds(10);                // 10us delay
     digitalWrite(trigPin, LOW);           // the trigPin is at ground

    duration = pulseIn(echoPin, HIGH);     // read the width of Echo pin
    distance = (duration) / 58;            // calculate the distance in cm

    Serial.println(distance);              // display distance
    Serial.println(" cm");                 // followed by unit(cm)
    delay(500);                            // 500ms delay
}
```

#1 задача: Measure Distance

❑ Task(Задание)

: Measure the distance to an object and show corresponding messages and control LED as below table.

Distance Range	Serial message	LED (Pin 13)
distance ≤ 2	Out of range	OFF
2 < distance ≤ 10	Obstacle	ON
10 < distance < 200	No obstacle	Blinking
distance ≥ 200	Out of range	OFF

❑ Use Tip(Использование Совет)

- Add your code to example code
- Print messages via serial port with Serial.print() and Serial.println()
- Write HIGH or LOW value to 13 pin with pinMode() and digitalWrite()

#1-a задача: Measure Distance Function

- Task(Задание)

  : Write a function to measure distance using the 1st task code

  The result is same as the 1st task code.

  - Function Name: Calc_dist
  - Function Parameter: None
  - Function Return: distance (integer type)

- Use Tip(Использование Совет)

```
int Calc_dist()
{
   ….
   return (int) distance;
}
```

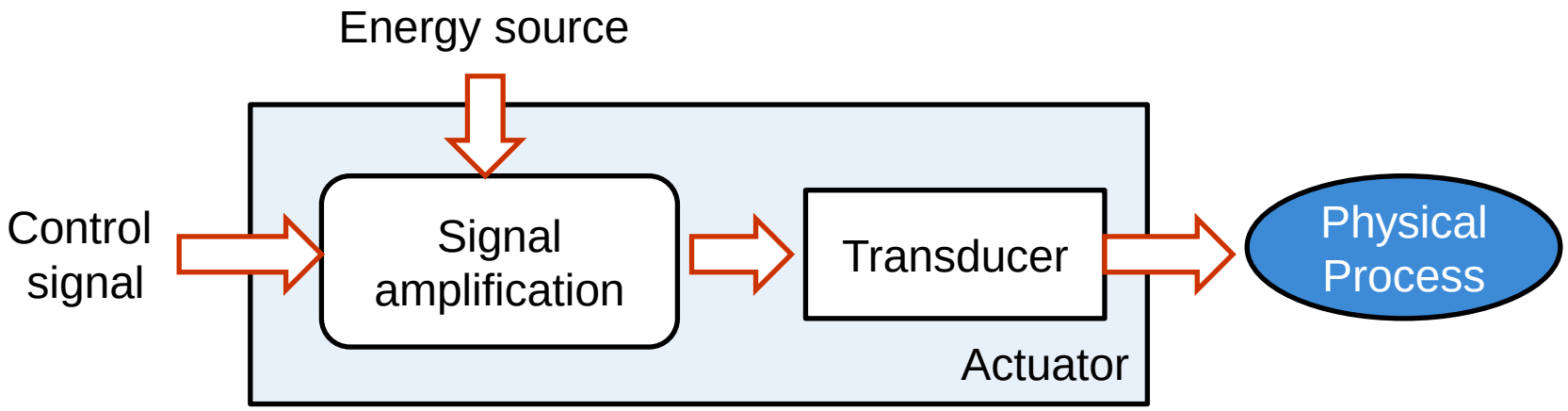
Contents

- Sensor
  - Distance measurement with Ultrasonic sensor

- Actuator
  - DC motor Control
  - Servo motor Control

Actuator

- An **actuator** is a component of a machine that is responsible for moving and controlling a mechanism or system
- An actuator requires a control signal and a source of energy (mainly electrical signal, air, fluids)
- When it receives a control signal, an actuator responds by converting the signal's energy into mechanical motion



Types of Actuators

- Mechanical actuators

- Pneumatic actuators
  - Use compressed air as the driving force

- Hydraulic actuators
  - Use hydraulic fluid to amplify the controller command signal

- Electrical actuators
  - Electric motors
    - AC motors
    - DC motors
    - Stepper motors
  - Solenoids

Electrical actuators

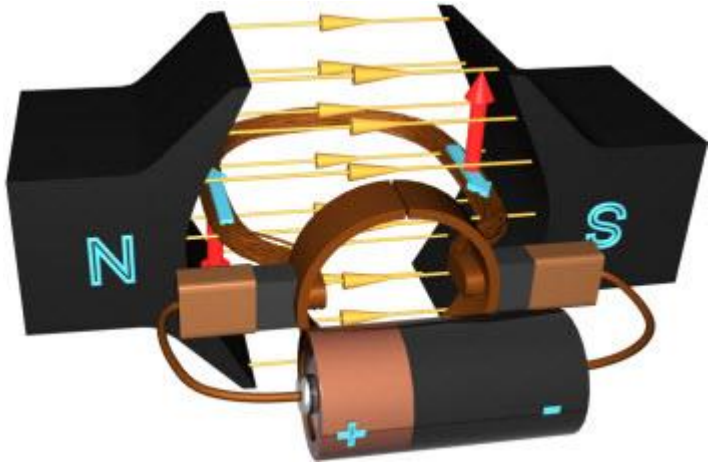
- An actuator that uses electricity to create mechanical motion
  - Actuated by motor that converts electrical energy into mechanical torque
  - More precise and accurate
  - Can be programmed for complicated paths of motion



- Applications
  - General purpose applications
  - Widely used in industrial processes

DC Motors

- Electromechanical devices which use the interaction of magnetic fields and conductors to convert the electrical energy into rotary mechanical energy
- Consist of two parts, the stationary body of the motor called the **Stator** and the inner part which rotates producing the movement called the **Rotor** or **“Armature”**



BRUSHLESS

- Longer Life
- More Power
- Quieter

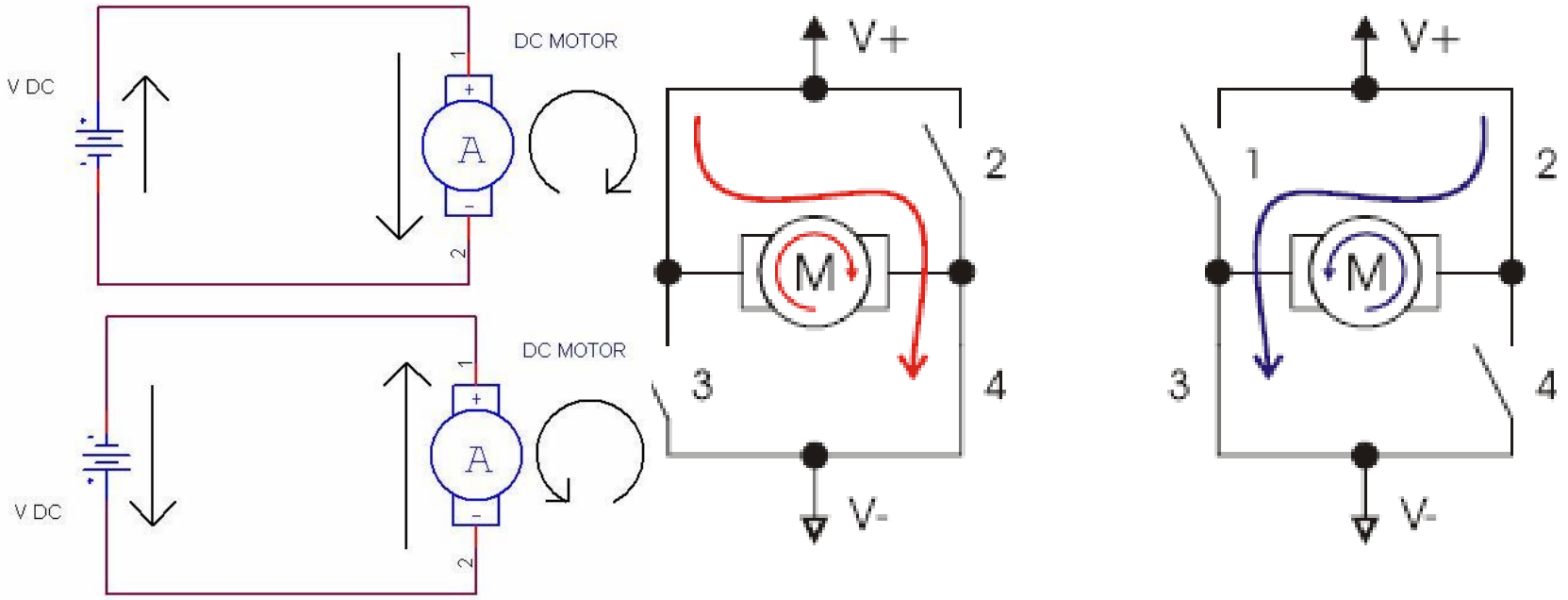
BRUSHED

- Shorter Life
- Less Power
- Noisier

DC Motor Control (Direction)

❑ DC Motor direction control

: When we apply DC voltage with proper current to a motor, it rotates in a particular direction but when we reverse the connection of voltage between two terminals, motor rotates in another direction.

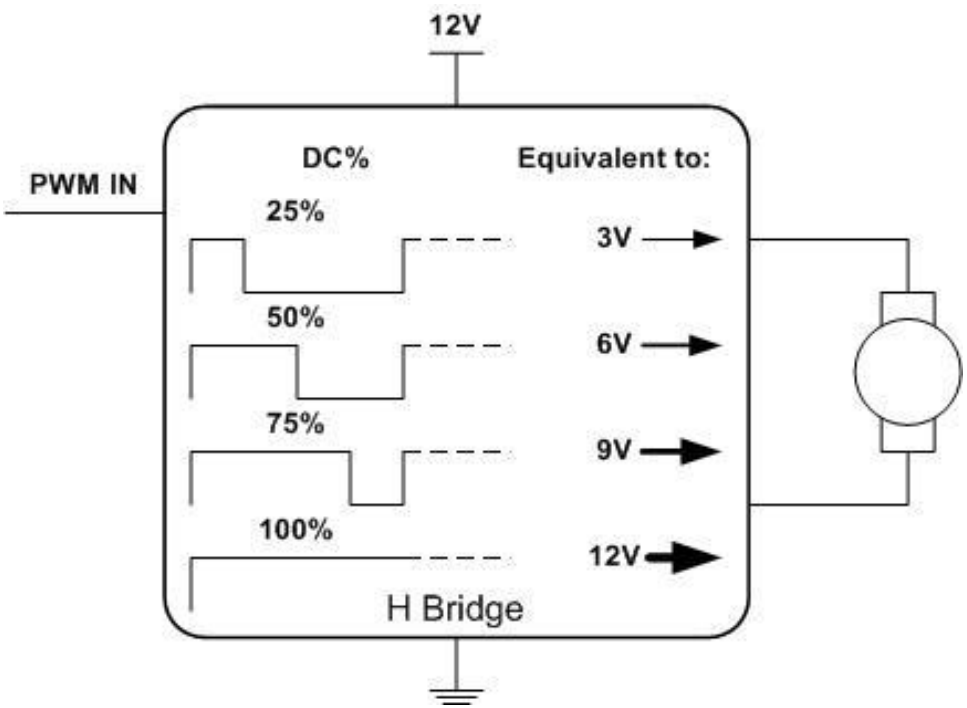


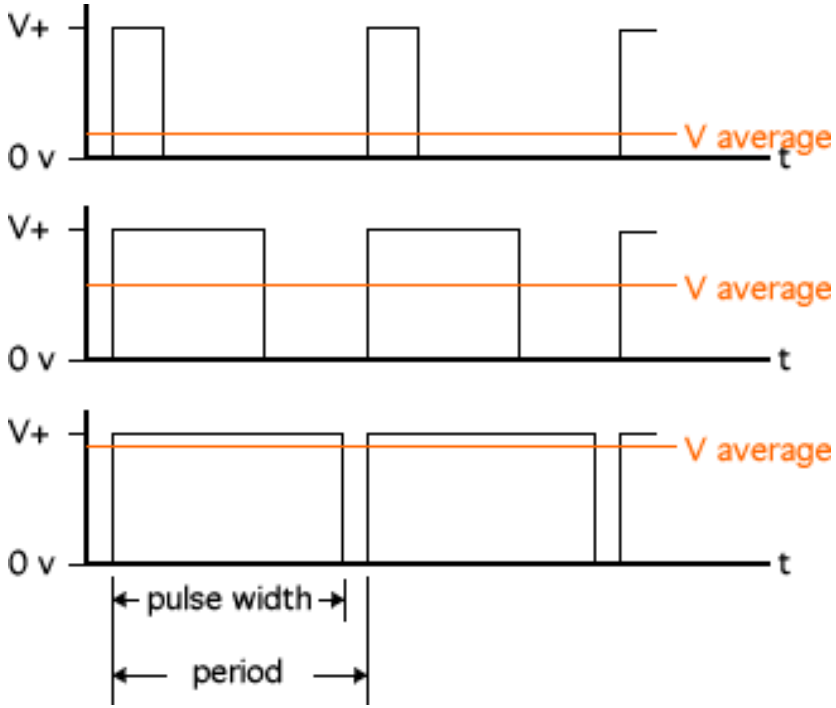
❖ H bridge is a special circuit which allows motor rotation in both directions

DC Motor Control (Speed)

❑ DC Motor speed control

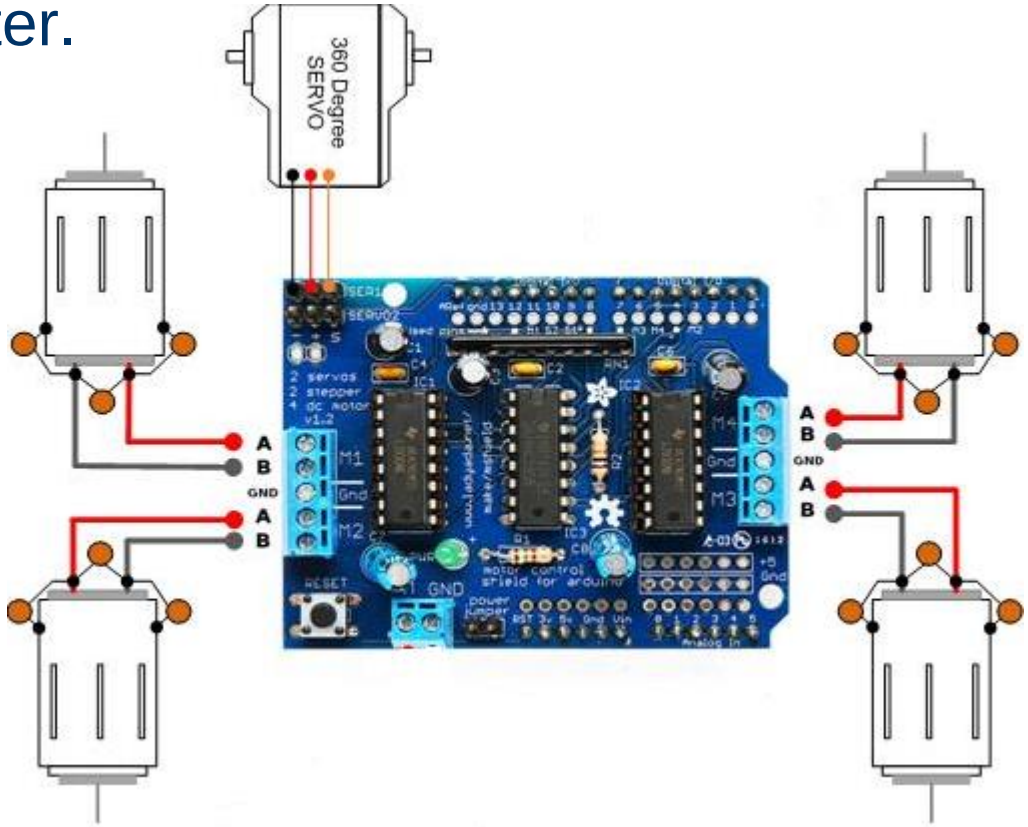
: The Duty Cycle will be directly proportional to the resulting voltage applied into the load. And on a DC Motor, voltage applied is directly proportional to motor speed.



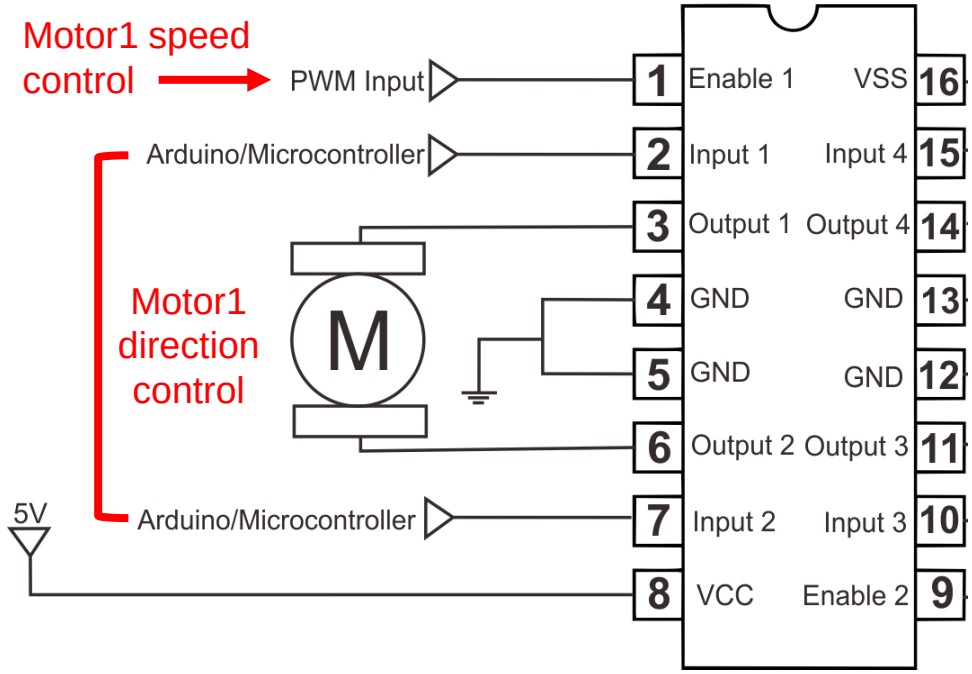


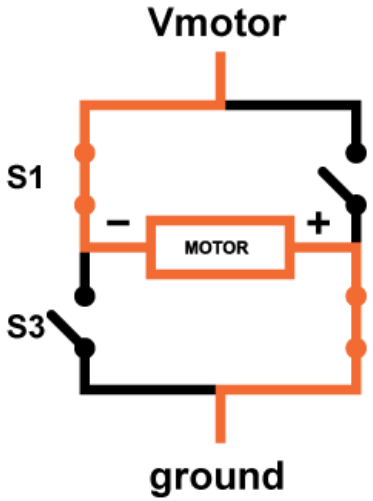
Adafruit Motor Shield

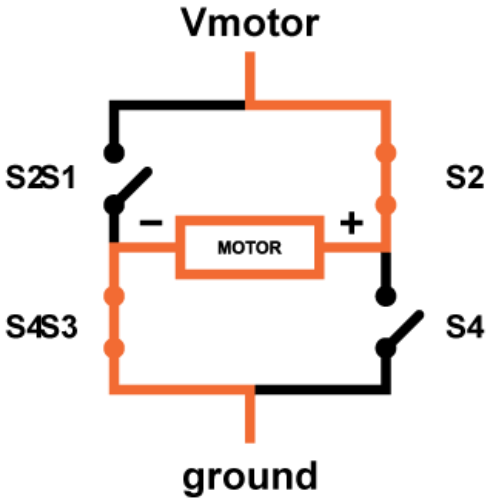
- The Adafruit Motor Shield is a great and quick way to control DC motors, servos or even stepper motors.
- The shield contains two L293D motor drivers and one 74HC595 shift register.
  - Up to 4 DC motors
  - Up to 2 stepper motors
  - Up to 2 Servo motors



DC Motor Control with L293D







Enable1	Input1	Input2	Motor Output
HIGH	LOW	HIGH	Turn in Clockwise direction
HIGH	HIGH	LOW	Turn in Anti-clockwise direction
HIGH	LOW	LOW	Stop
HIGH	HIGH	HIGH	Stop
LOW	X	X	Stop

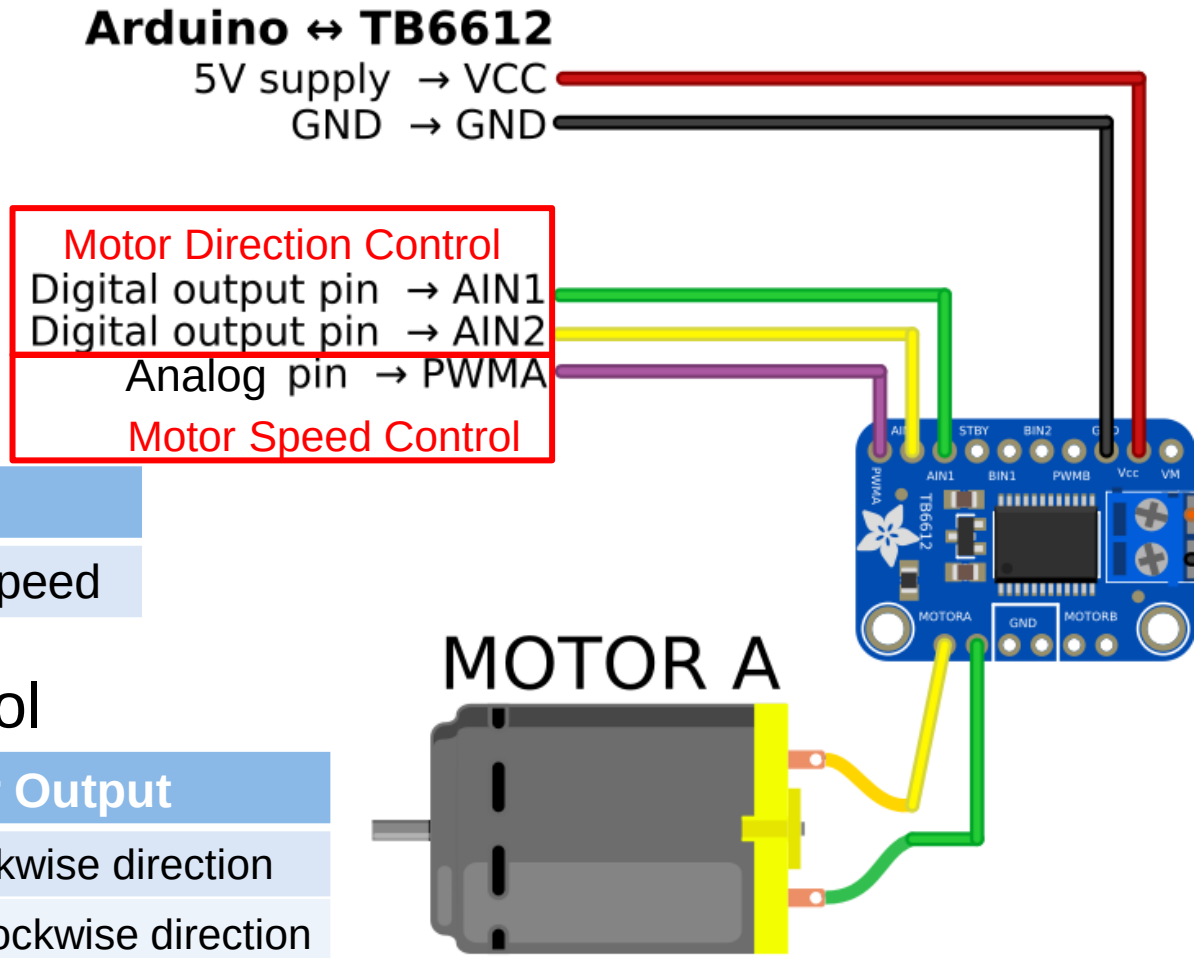
DC Motor Control using AFMotor

- 4 steps in your sketch:
  1. Include a library in your code: **#include <AFMotor.h>**
  2. Create the AF_DCMotor object with **AF_DCMotor(*motor_number*)** class
  3. Set the speed of the motor using **setSpeed(*speed*)** where the ***speed*** ranges from 0 (stopped) to 255 (full speed)
  4. To run the motor, call **run(*direction*)** where ***direction*** is
     - **FORWARD** or
     - **BACKWARD** or
     - **RELEASE**

Example Code using L293D

```
#include <AFMotor.h>                    // include motor driver library

AF_DCMotor  motor1(1);                  // create motor #1 object
AF_DCMotor  motor2(2);                  // create motor #2  object
AF_DCMotor  motor3(3);                  // create motor #3  object
AF_DCMotor  motor4(4);                  // create motor #4  object

void loop() {
   motor1.setSpeed(200);                // set the speed to 200/255
   motor2.setSpeed(200);                // set the speed to 200/255
   motor3.setSpeed(200);                // set the speed to 200/255
   motor4.setSpeed(200);                // set the speed to 200/255

   motor1.run(FORWARD);                  // turn it on going forward
   motor2.run(FORWARD);                  // turn it on going forward
   motor3.run(FORWARD);                  // turn it on going forward
   motor4.run(FORWARD);                  // turn it on going forward
   delay(1000);
}
```

DC Motor Control with TB6612



❑ Motor Speed Control

PWMA	Motor Output
0~255	0 is off, 255 is full speed

❑ Motor Direction Control

AIN1	AIN2	Motor Output
LOW	HIGH	Turn in Clockwise direction
HIGH	LOW	Turn in Anti-clockwise direction
LOW	LOW	Stop
HIGH	HIGH	Stop

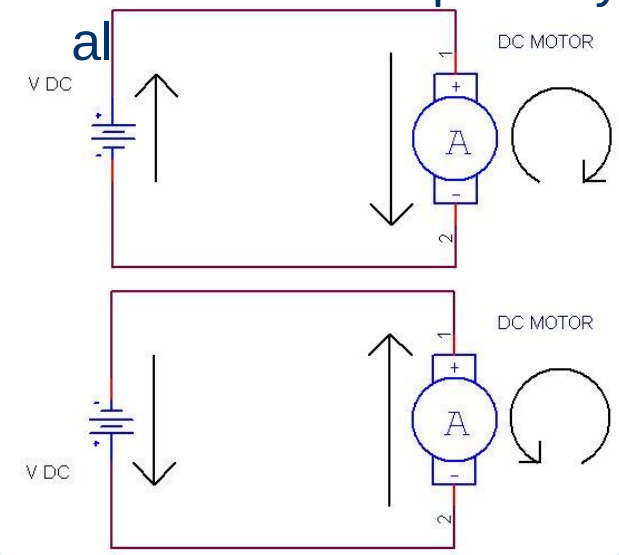
Example Code using TB6612

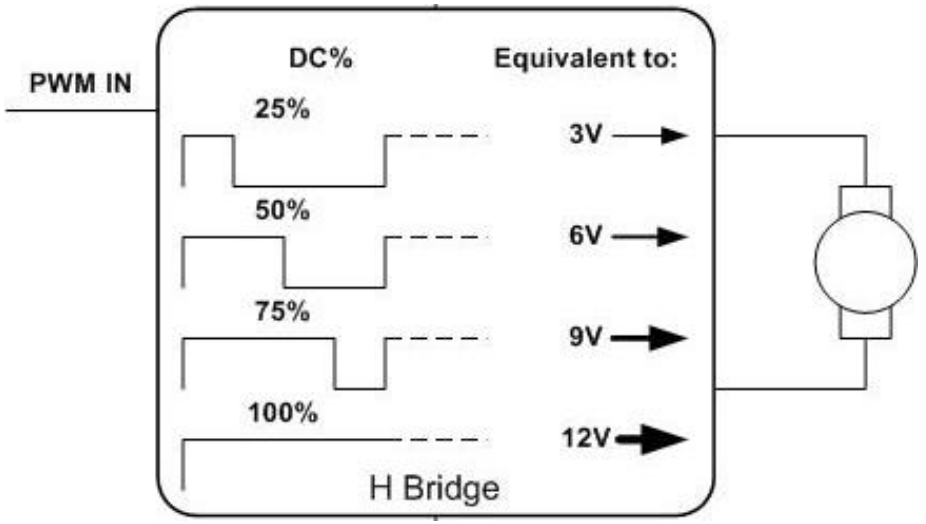
```
#define PWMA   6                  //Left Motor Speed pin (ENA)
#define AIN1   A1                 //Motor-L backward (IN1)
#define AIN2   A0                 //Motor-L forward (IN2).

#define PWMB   5                  //Right Motor Speed pin (ENB)
#define BIN1   A2                 //Motor-R forward (IN3)
#define BIN2   A3                 //Motor-R backward (IN4)

#define SPEED 100                 // motor Speed

analogWrite(PWMA,SPEED)           // Set the speed on MotorA
digitalWrite(AIN1, LOW)           // Move MotorA forward
digitalWrite(AIN2, HIGH)          // Move MotorA forward

analogWrite(PWMB,SPEED)           // Set the speed on MotorB
digitalWrite(BIN1, LOW)           // Move MotorB forward
digitalWrite(BIN2, HIGH)          // Move MotorB forward
```

#2 задача: Motor Control

- Task(Задание)
  1. Write a code to move the vehicle(or robot) backward
  2. Write a code that slowly rotates the vehicle(or robot) left or right

- Use Tip(Использование Совет)
  - Apply DC voltage correctly to the two digital pins
  - Set the motor speed by applying appropriate voltage with PWM sign al





#3 задача: Motor Control by Range

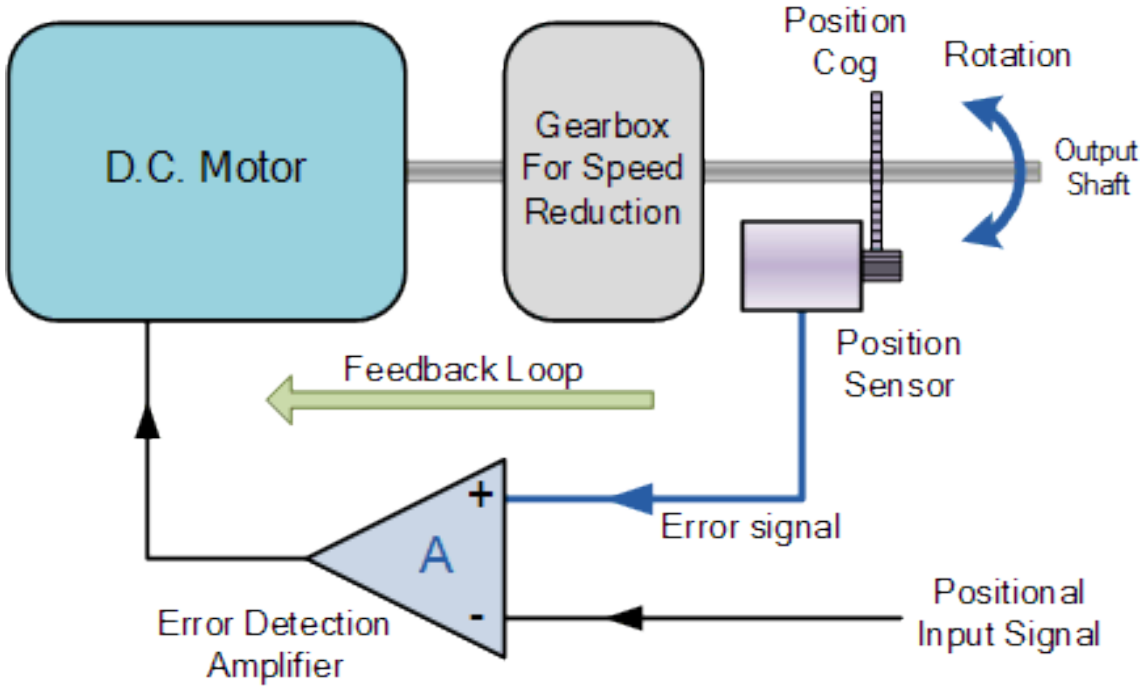
- Task(Задание)

  : According to the range, control Motor and show corresponding messages as below table.

Distance Range	Motor	Print message
distance ≤ 5	Stop	Stop
5 < distance ≤ 20	Go Forward slowly	Obstacle
distance ≥ 20	Go forward fast	No obstacle

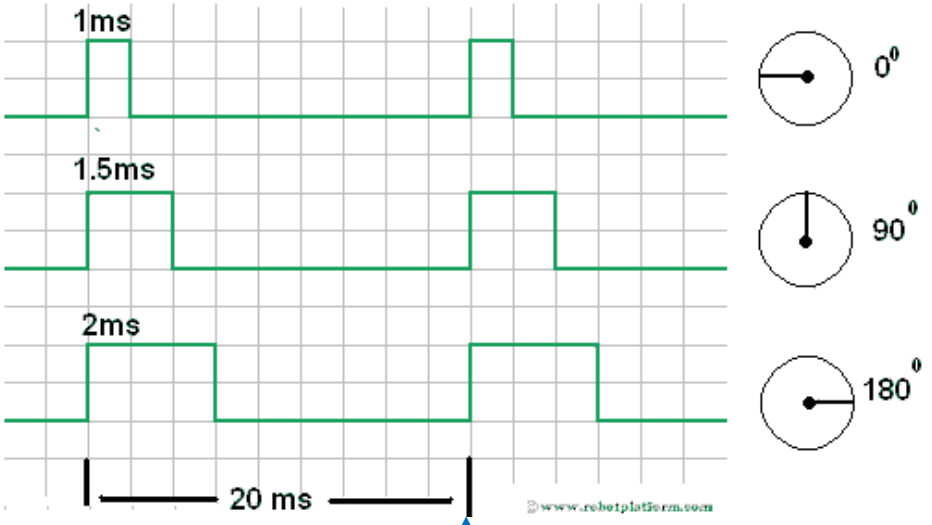
DC Servo Motor

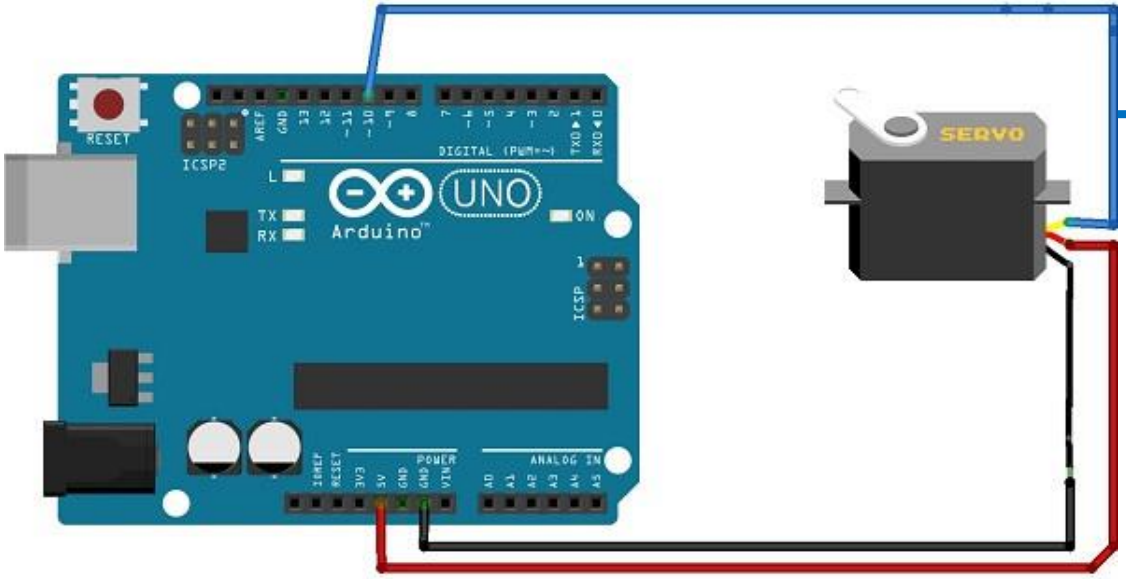
- A servo motor consist of several devices in one package, DC motor, gearbox, feedback device and error correction.
- Easily controlled using just three wires, Power, Ground an d Signal Control. The angle of rotation is controlled by the duration of applied pulse to Signal PIN.



Servo Motor Control with Arduino

- The position data to the control should be sent in the form of PWM signal through the Signal pin of servo motor





Signal pin

- Servo checks the pulse in every 20ms
- PWM Pulse on Control Pin
  - 1ms width can rotate servo to 0°
  - 1.5ms width can rotate to 90°
  - 2ms width can rotate to 180°

Servo Motor Control Software

- 4 steps in your sketch:
  1. Include a library in your code: **#include <Servo.h>**
  2. Create the an object with **Servo** class
  3. Assign a pin on the Arduino to a servo motor using **attach(*pin No.*)**
  4. Write the position of the servo motor using **write(*angle*)** where the ***angle*** ranges from **0 to 180 degree.**

Example Code

```
#include <Servo.h>

Servo myservo;                          // create a servo object

void setup() {
    myservo.attach(10);                 // assign pin10 on the board to servo motor
}

void loop() {
    myservo.write(90);                  // set the position of servo motor to the center
    delay(20);                          // delay time for moving of servo motor
}
```

#4 задача: Servo Motor control

- Task(Задание)
  1. Set the position of servo motor from 0 to 180 degree
     - Increase 1 degree every task
     - Apply 20ms delay time for moving of servo motor after sending command to increase 1 degree
  2. After moving the position to 180 degree, move the position to the center