Virtual Reality Haptic Interactions (虛擬實境與觸覺回饋互動)

Lecturer: Ray

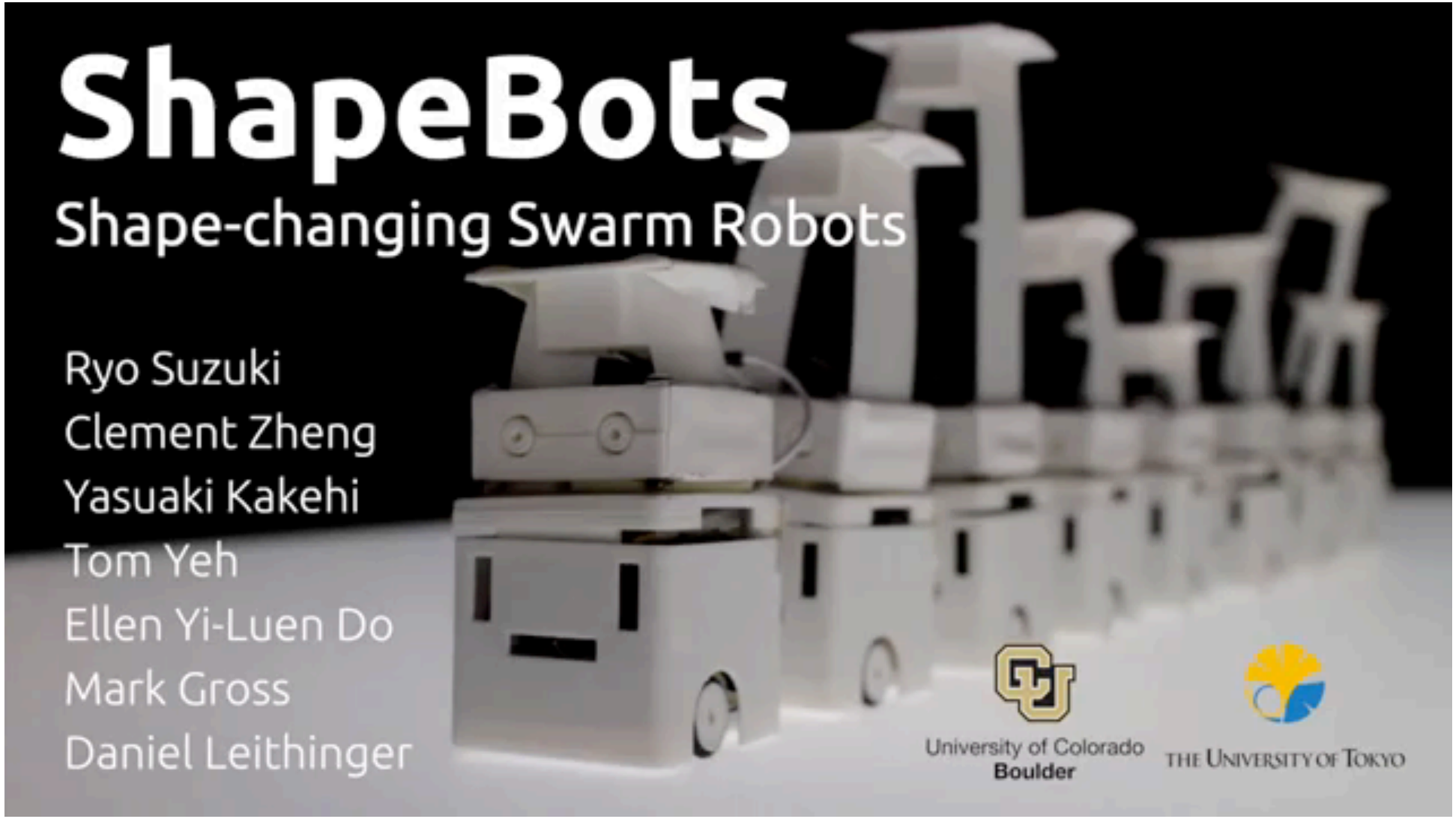
Week 7 (10/23)

Syllabus

	1	Introduction	9	Unity	
	2	Brain storming & paper reading instruction	10	Unity	
Team up	3	Introducing Arduino	11	Unity, communication and XR	**Check point**
	4	Arduino, coding	12	AI and LLM	
Proposal check point	5	Motors	13	Project discussion	
	6	Motor and encoder	14	3D printing	**Check point**
Proposal check point	7	Paper discussion	15	Project discussion and checkpoint	
	8	Proposal (with lo-fi prototype)	16	Final project	

Syllabus

	1	Introduction	9	Proposal (with lo-fi prototype)	
	2	Brain storming & paper reading instruction	10	Unity	
Team up	3	Introducing Arduino	11	Unity, communication and XR	**Check point**
	4	Arduino, coding	12	AI and LLM	
Proposal check point	5	Motors	13	Project discussion	
	6	Motor and encoder	14	3D printing	**Check point**
	7	Motor and encoder	15	Project discussion and checkpoint	
Proposal check point	8	Paper discussion	16	Final project	



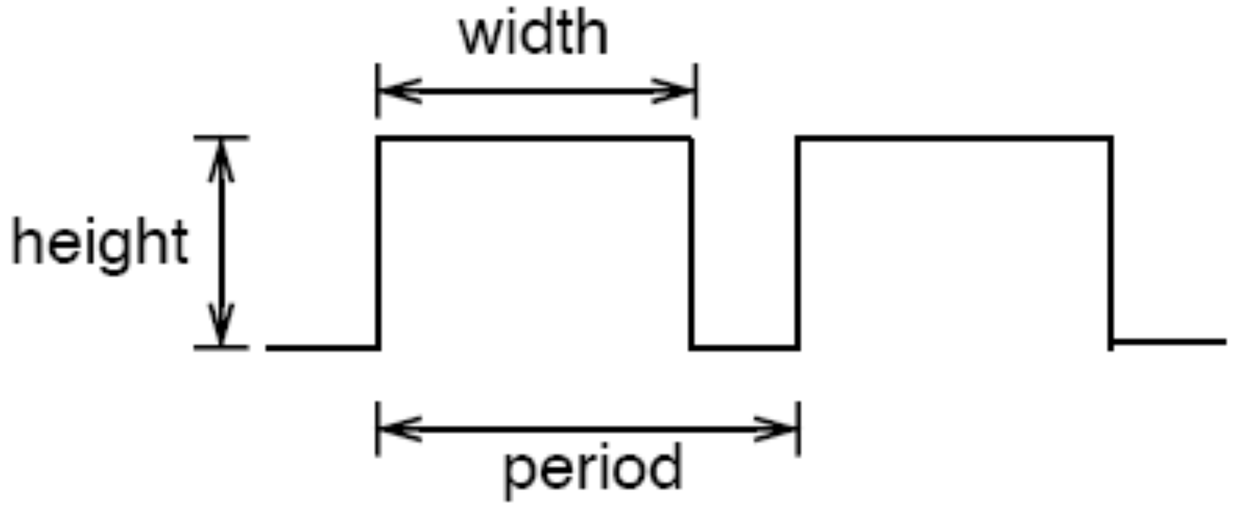
ShapeBots, UIST'19

Rovables, UIST'16

FacePush, UIST’18

Transcalibur, CHI'19

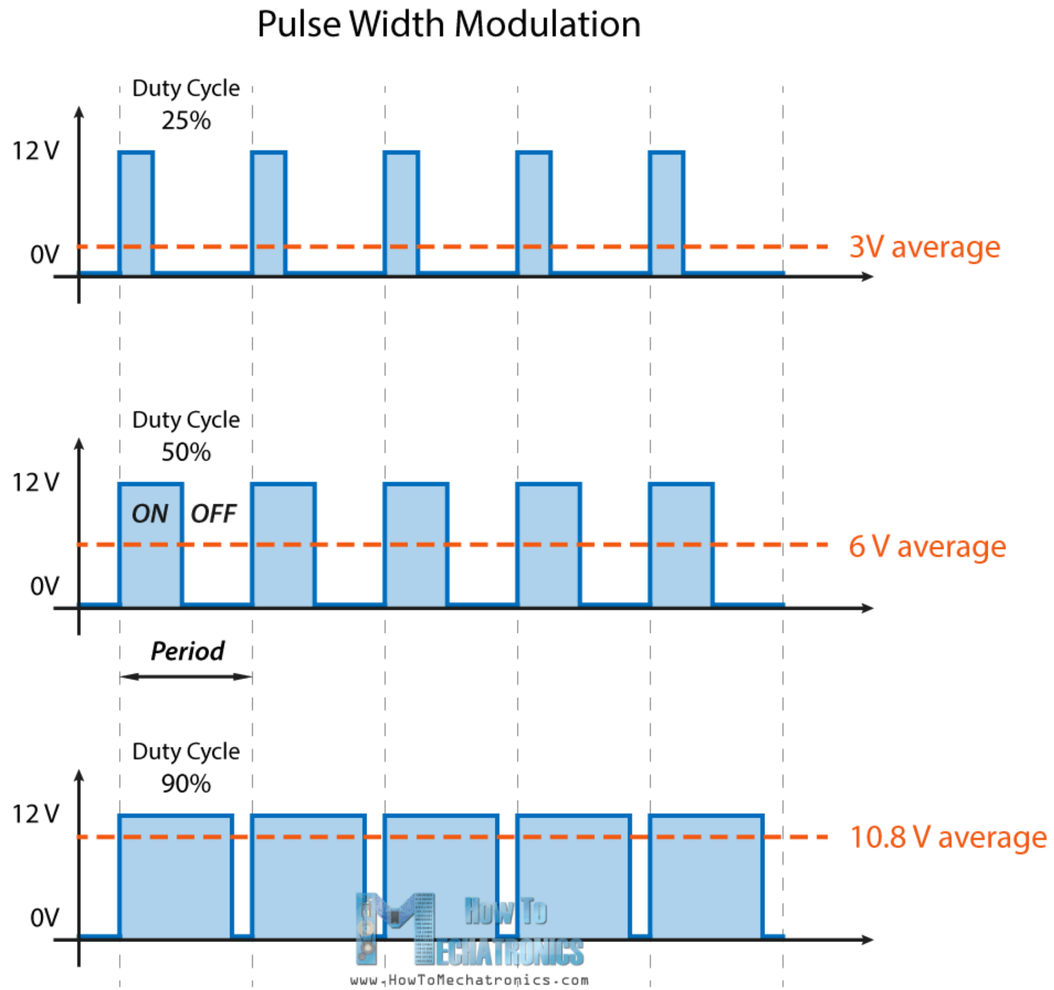
Pulse Width Modulation (PWM)



Three characteristics of PWM signals:

- **Pulse width range (min/max)**
- **Pulse period (=1/pulse per second)**
- **voltage levels (0-5V, for instance)**

Pulse Width Modulation (PWM)

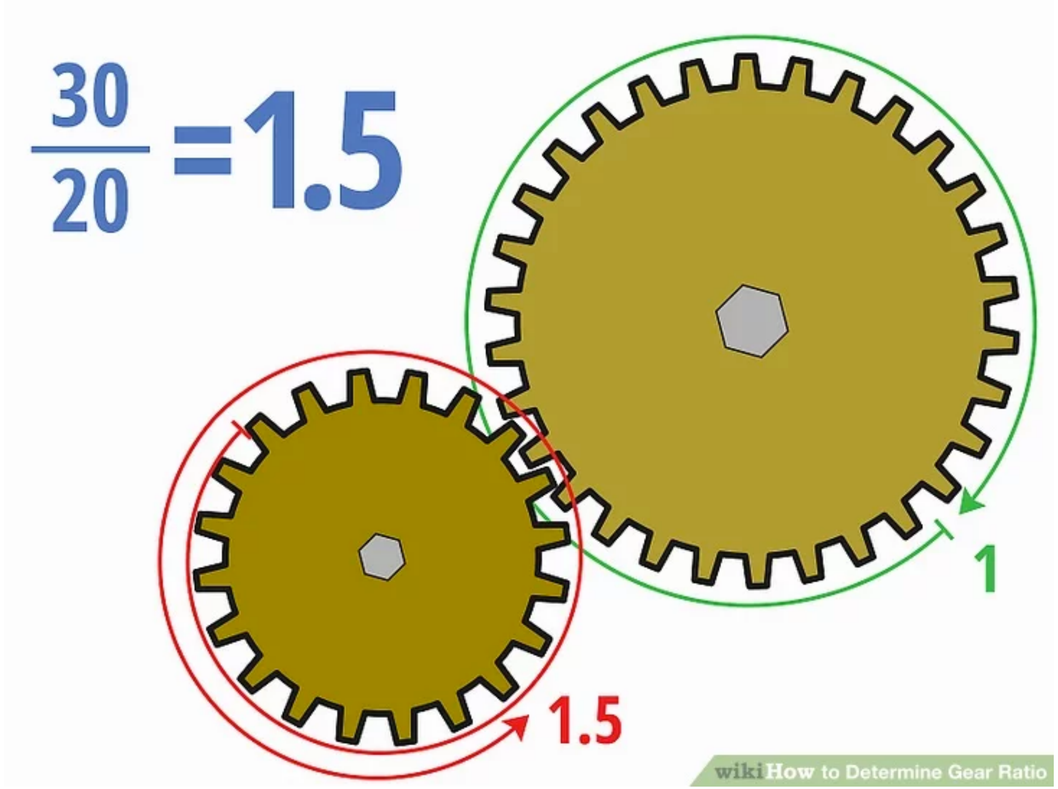


https://goo.gl/hcSV1W

https://goo.gl/vfDVW6

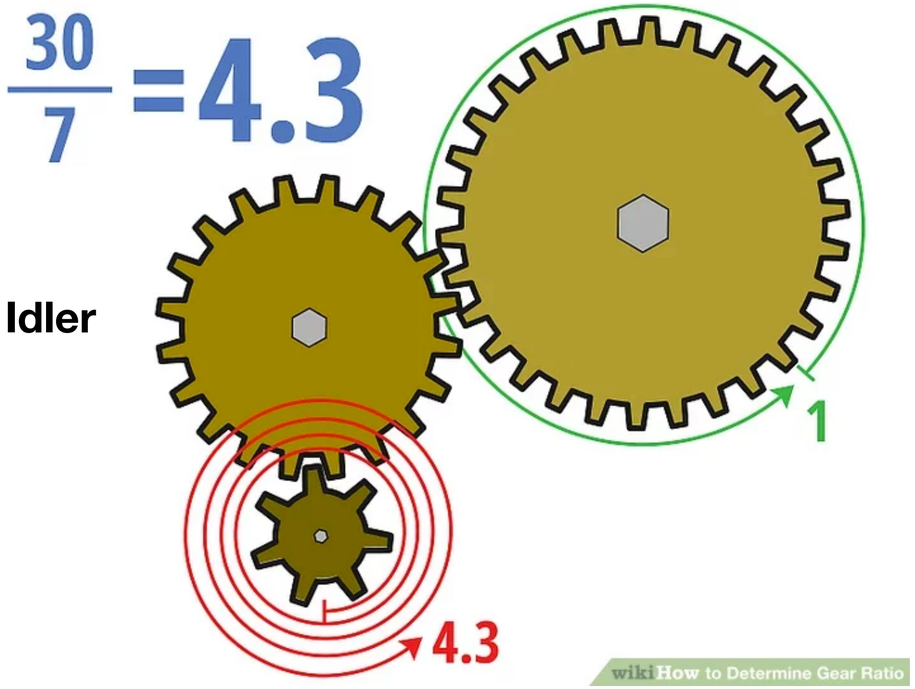
Gear Ratio

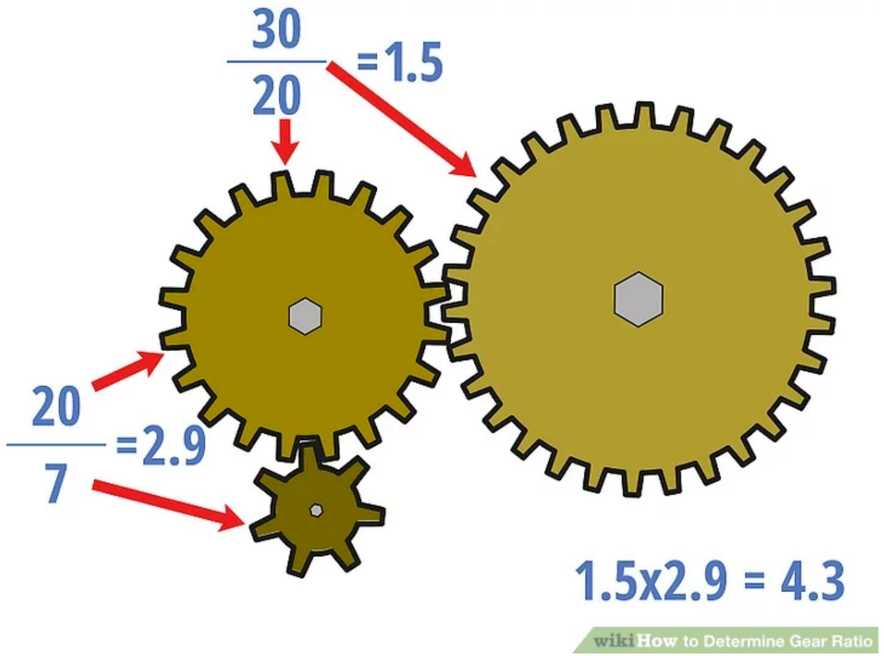
$$\text{Gear ratio} = \frac{\text{Number of teeth on driven gear}}{\text{Number of teeth on driver gear}}$$



Gear Ratio

$$\text{Gear ratio} = \frac{\text{Number of teeth on driven gear}}{\text{Number of teeth on driver gear}}$$

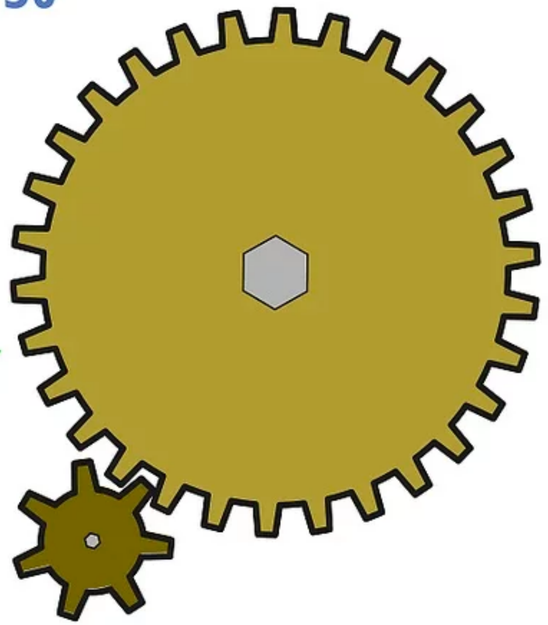


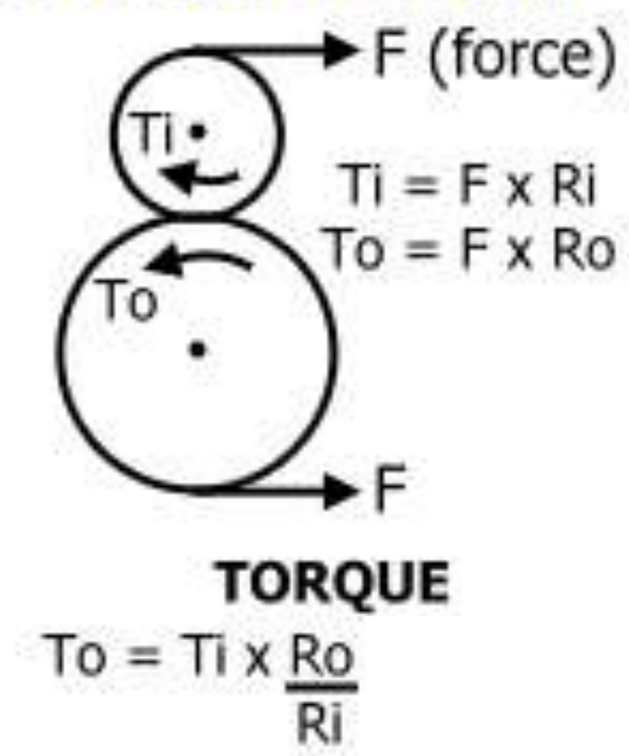


Gear Ratio

130 rpms × 7 = S2 × 30
910 = S2 × 30
910/30 = S2
30.33 rpms = S2

30.33
rpm

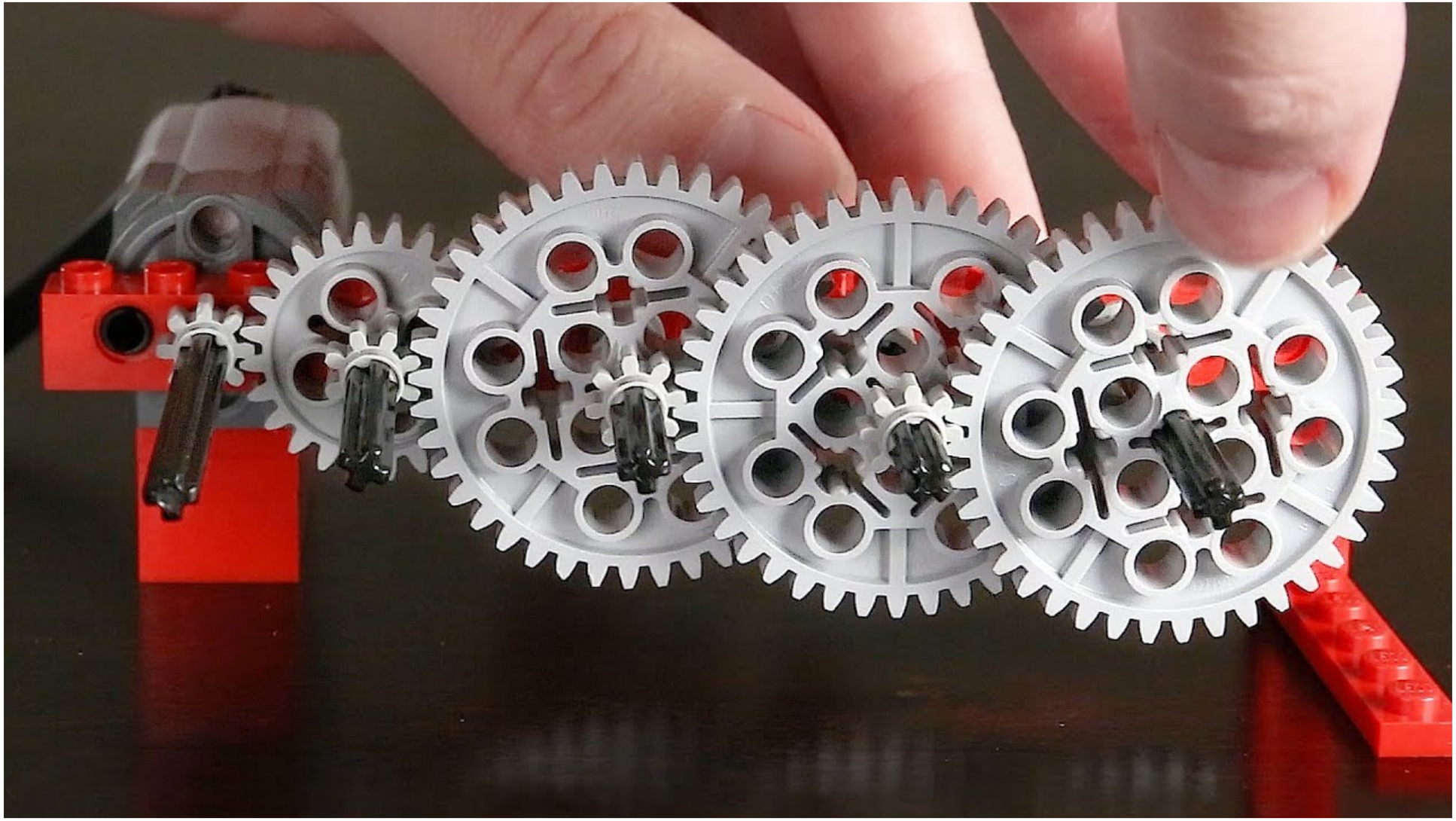




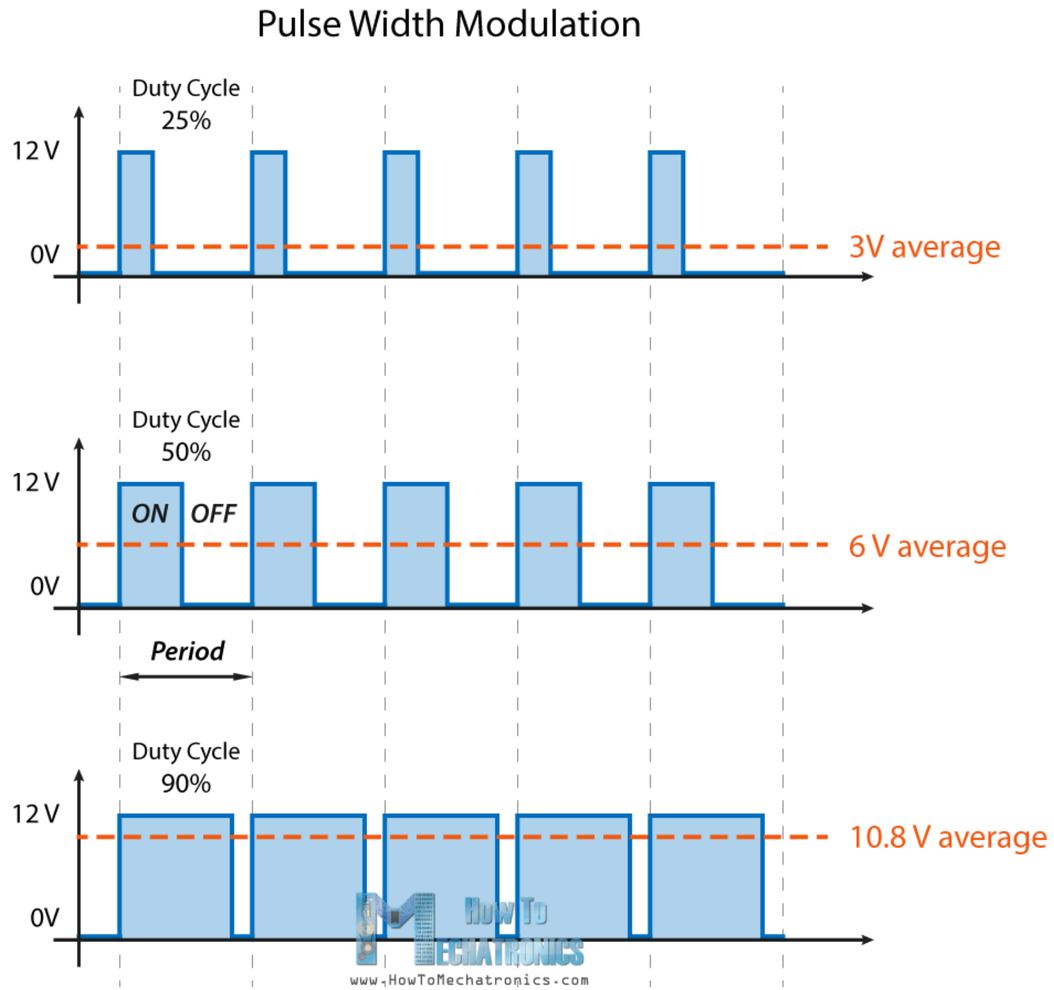
S1 × T1 = S2 × T2

https://goo.gl/1YwnLY

DC Motor Control



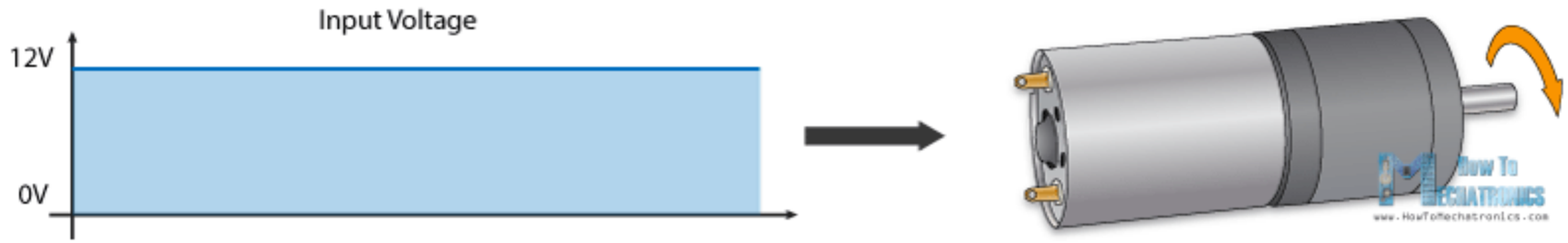
Pulse Width Modulation (PWM)



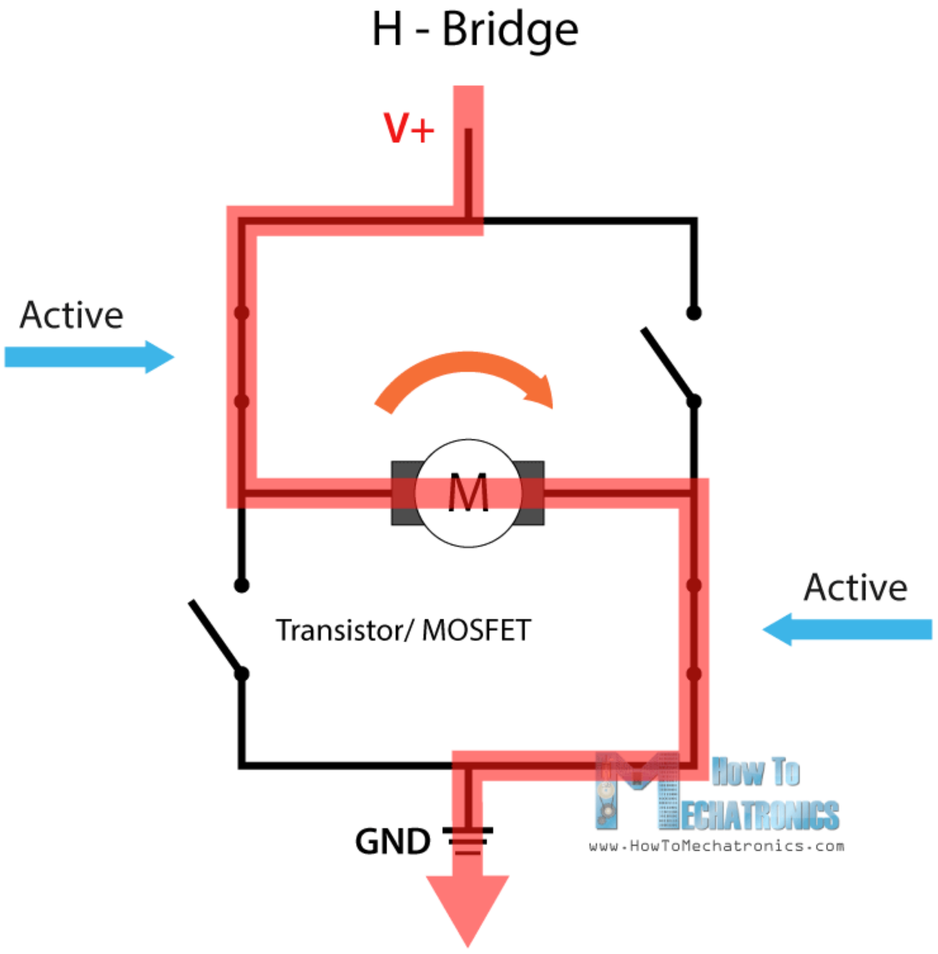
https://goo.gl/hcSV1W

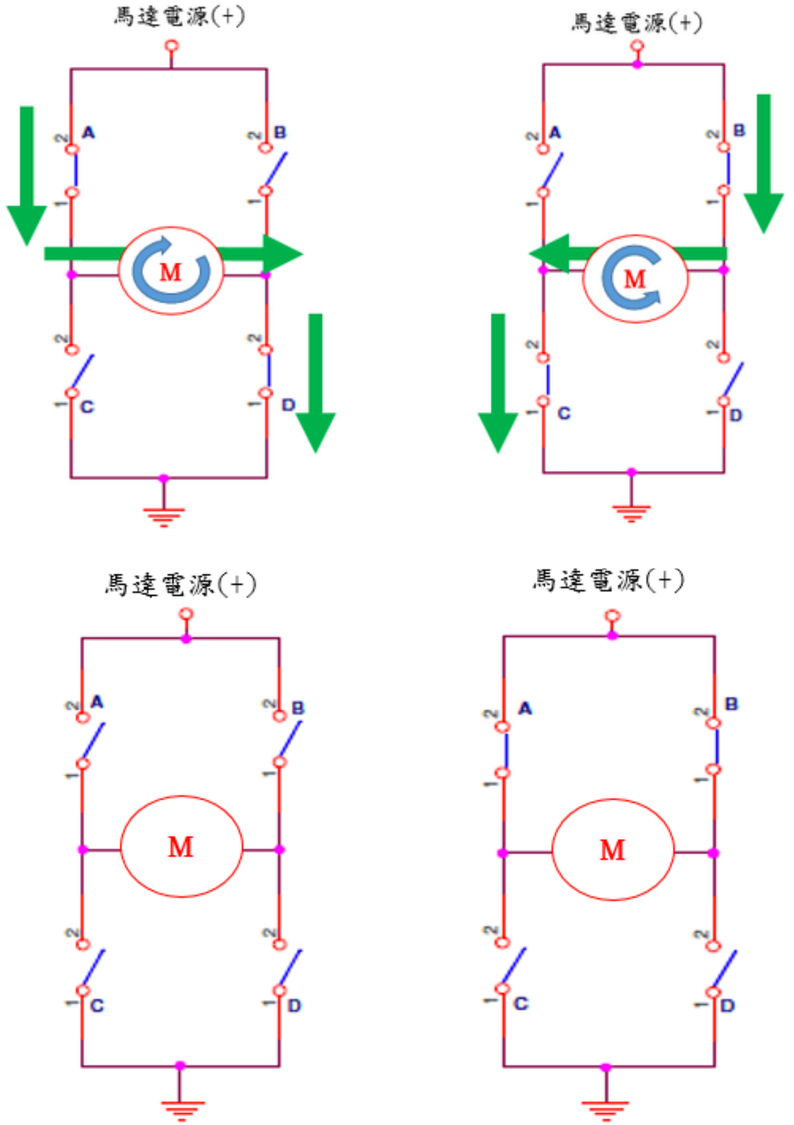
https://goo.gl/vfDVW6

DC Motor Control



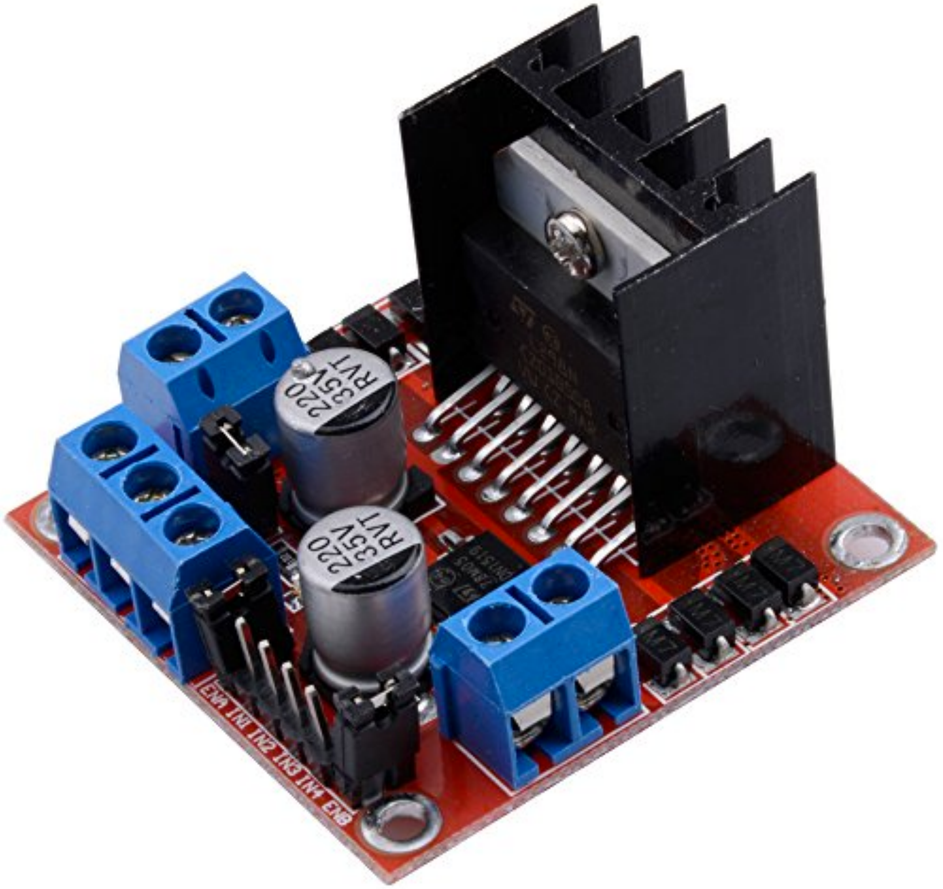
DC Motor Control H-Bridge



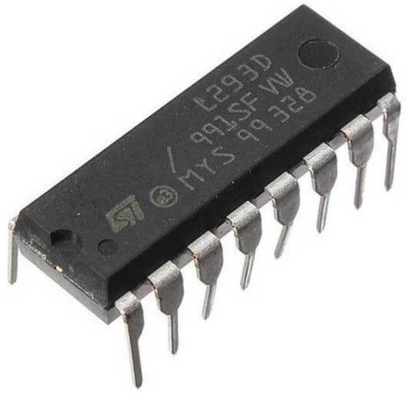


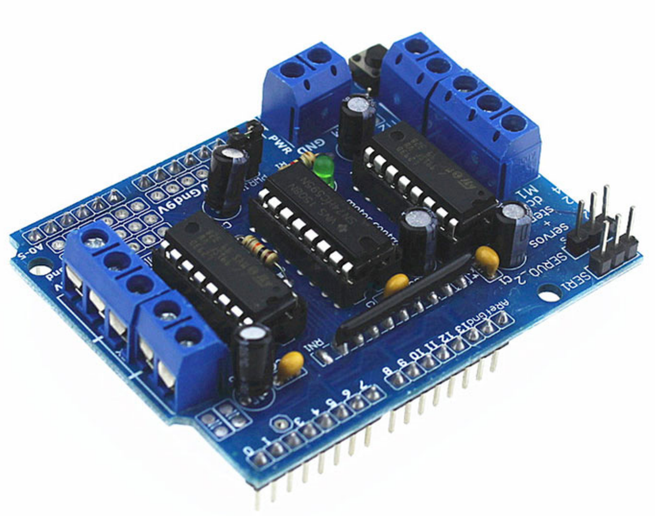
switching elements: transistors or MOSFETs

DC Motor + Motor Driver



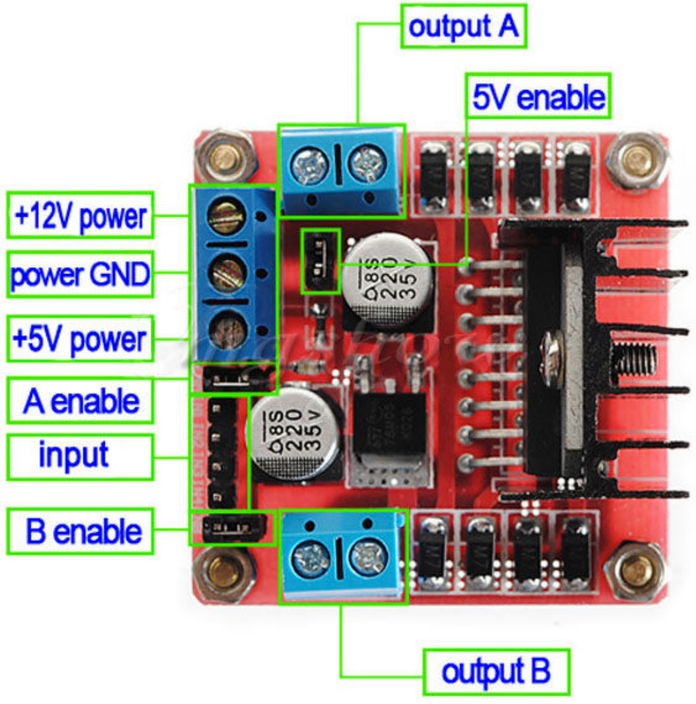
L298N

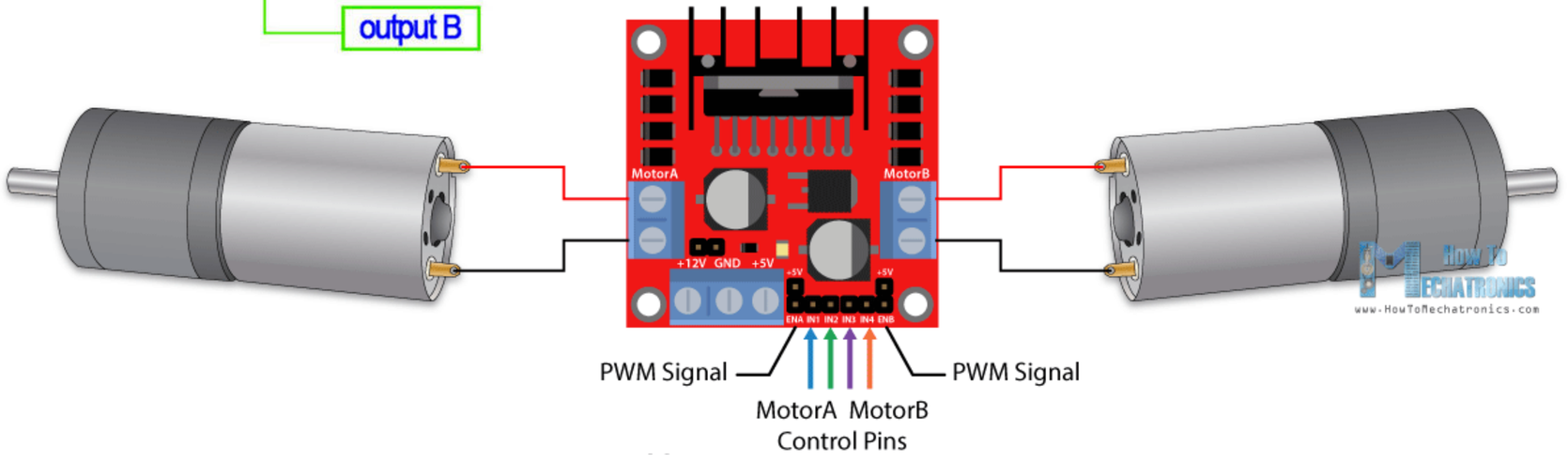




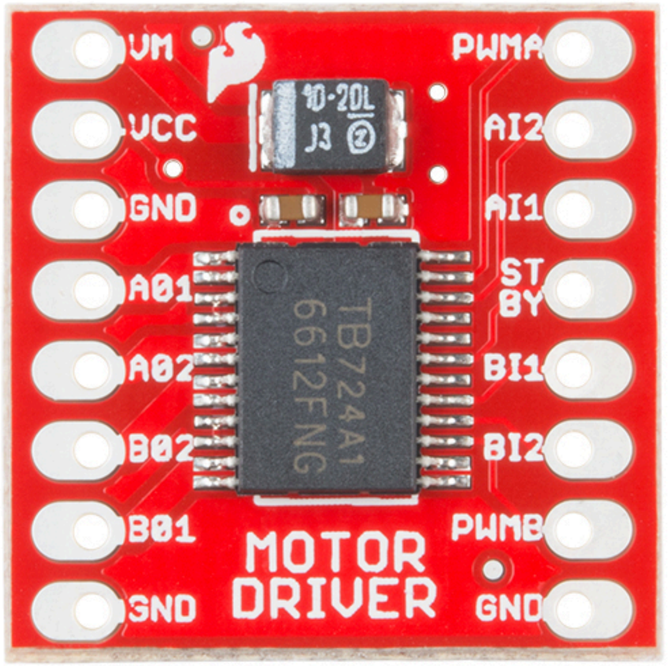
L293D

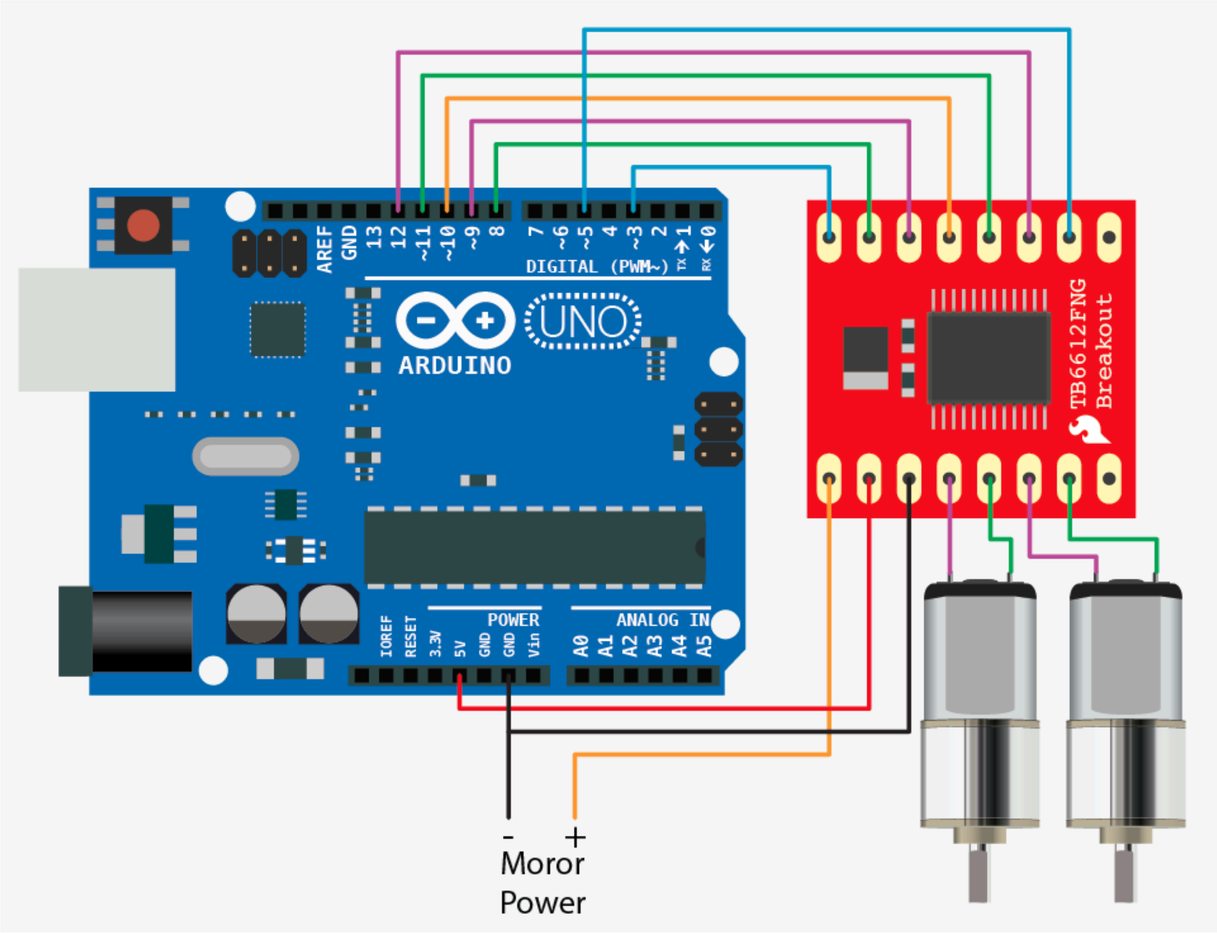
Motor Driver - L298N



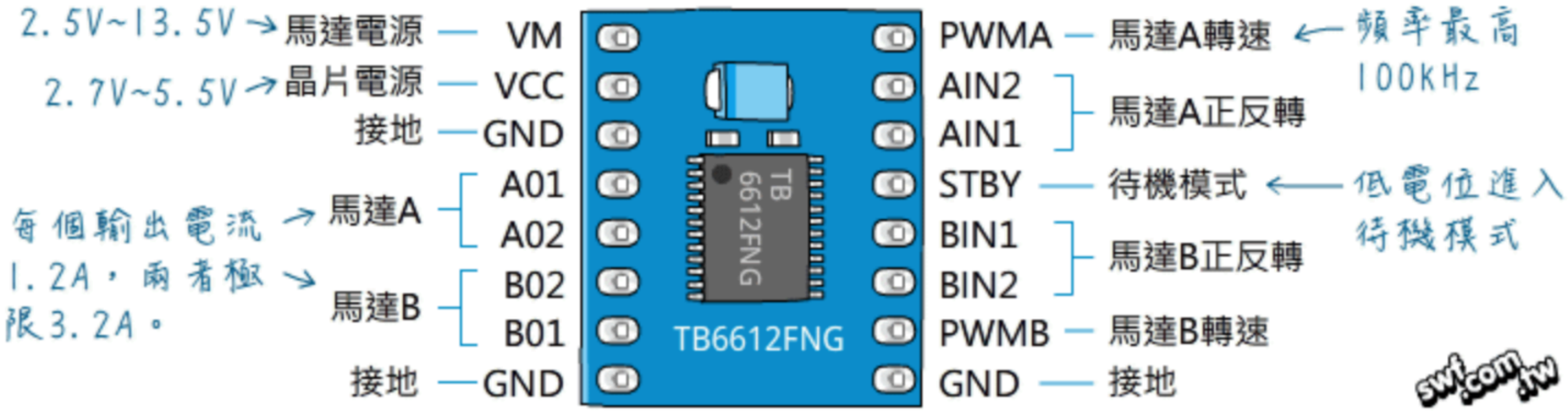


Motor Driver - TB6612FNG





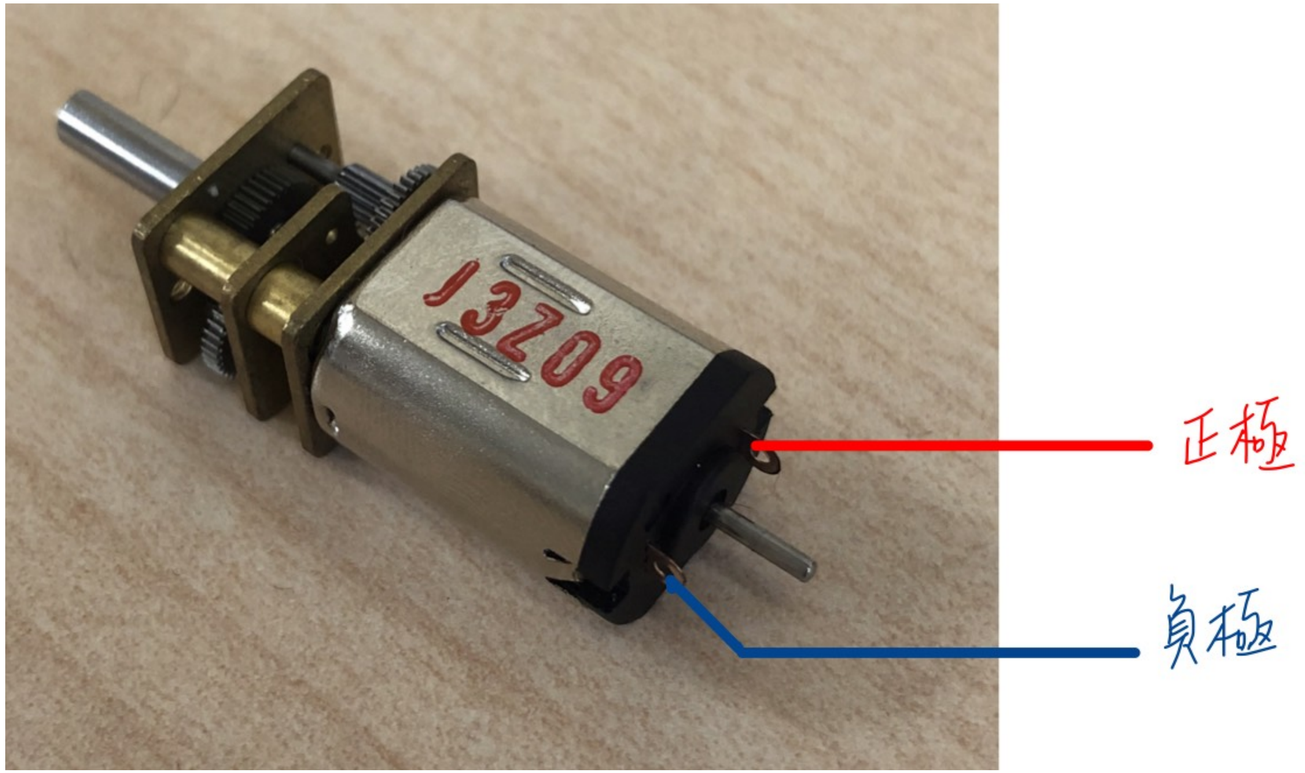
Motor Driver - TB6612FNG



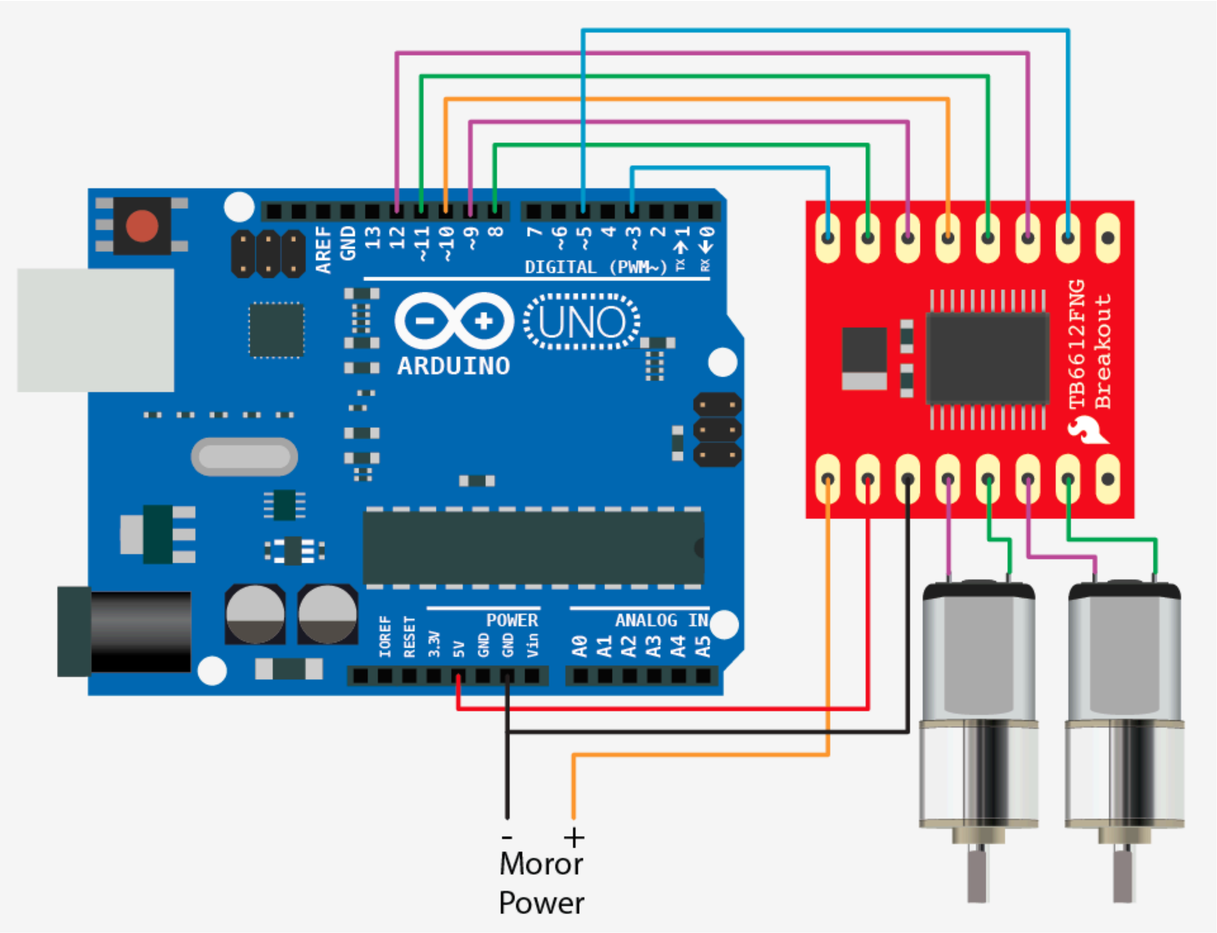
輸入			輸出		模式說明
AIN1	AIN2	STBY	A01	A02	
1	1	1	0	0	煞車（short brake）
0	1	1	0	1	逆時針方向旋轉
1	0	1	1	0	順時針方向旋轉
0	0	1	0	0	停止（stop）
0	0	0	0	0	待機（standby）

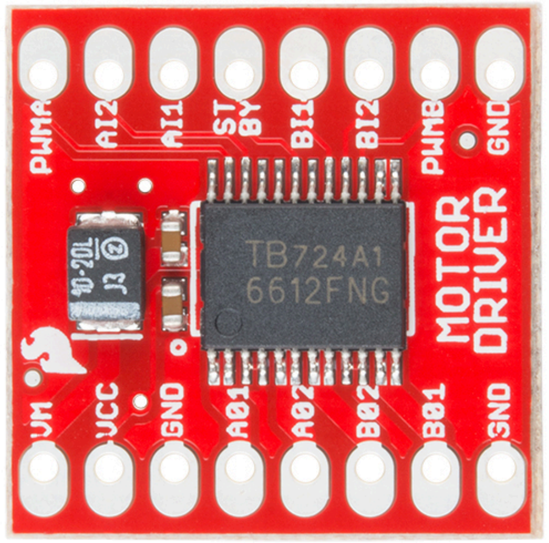
source:https://swf.com.tw/?p=1066

DC motor



Motor Driver Control

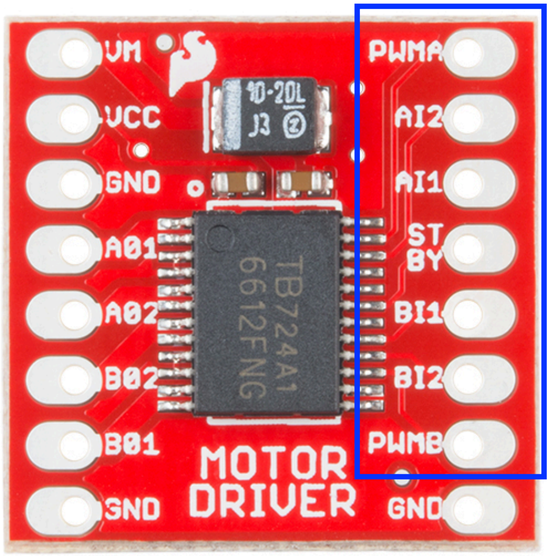




https://bit.ly/3IwTolY

Motor Driver Control

```
//Define the Pins
//Motor 1
int pinAIN1 = 9; //Direction
int pinAIN2 = 8; //Direction
int pinPWMA = 3; //Speed

//Motor 2
int pinBIN1 = 11; //Direction
int pinBIN2 = 12; //Direction
int pinPWMB = 5; //Speed

//Standby
int pinSTBY = 10;

//Constants to help remember the parameters
static boolean turnCW = 0;  //for motorDrive function
static boolean turnCCW = 1; //for motorDrive function
static boolean motor1 = 0;  //for motorDrive, motorStop, motorBrake functions
static boolean motor2 = 1;  //for motorDrive, motorStop, motorBrake functions
```



https://bit.ly/3IwTolY

Motor Driver Control

```
void motorDrive(boolean motorNumber, boolean motorDirection, int motorSpeed)
{
  boolean pinIn1;  //Relates to AIN1 or BIN1 (depending on the motor number specified)

  //Specify the Direction to turn the motor
  //Clockwise: AIN1/BIN1 = HIGH and AIN2/BIN2 = LOW
  //Counter-Clockwise: AIN1/BIN1 = LOW and AIN2/BIN2 = HIGH
  if (motorDirection == turnCW){
    pinIn1 = HIGH;
  }
  else{
    pinIn1 = LOW;
  }

  //Select the motor to turn, and set the direction and the speed
  if(motorNumber == motor1)
  {
    digitalWrite(pinAIN1, pinIn1);
    digitalWrite(pinAIN2, !pinIn1);  //This is the opposite of the AIN1
    analogWrite(pinPWMA, motorSpeed);
  }
  else
  {
    digitalWrite(pinBIN1, !pinIn1);
    digitalWrite(pinBIN2, pinIn1);  //This is the opposite of the BIN1
    analogWrite(pinPWMB, motorSpeed);
  }

  //Finally , make sure STBY is disabled - pull it HIGH
  digitalWrite(pinSTBY, HIGH);
}
```

Direction

https://bit.ly/3IwTolY

Motor Driver Control

```
void motorBrake(boolean motorNumber)
{
/*
This "Short Brake"s the specified motor, by setting speed to zero
*/

  if (motorNumber == motor1){
    analogWrite(pinPWMA, 0);
  }
  else{
    analogWrite(pinPWMB, 0);
  }
}
```

https://bit.ly/3IwTolY

Motor Driver Control

```
void motorStop(boolean motorNumber)
{
  /*
  This stops the specified motor by setting both IN pins to LOW
  */
  if (motorNumber == motor1) {
    digitalWrite(pinAIN1, LOW);
    digitalWrite(pinAIN2, LOW);
  }
  else
  {
    digitalWrite(pinBIN1, LOW);
    digitalWrite(pinBIN2, LOW);
  }
}

 void motorsStandby()
 {
   /*
   This puts the motors into Standby Mode
   */
   digitalWrite(pinSTBY, LOW);
 }
```

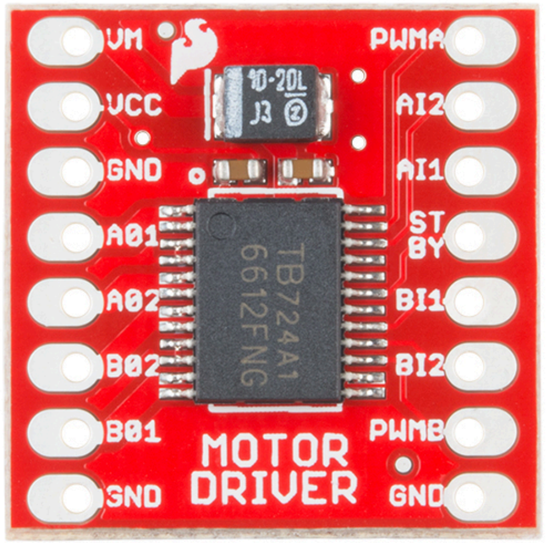
https://bit.ly/3IwTolY

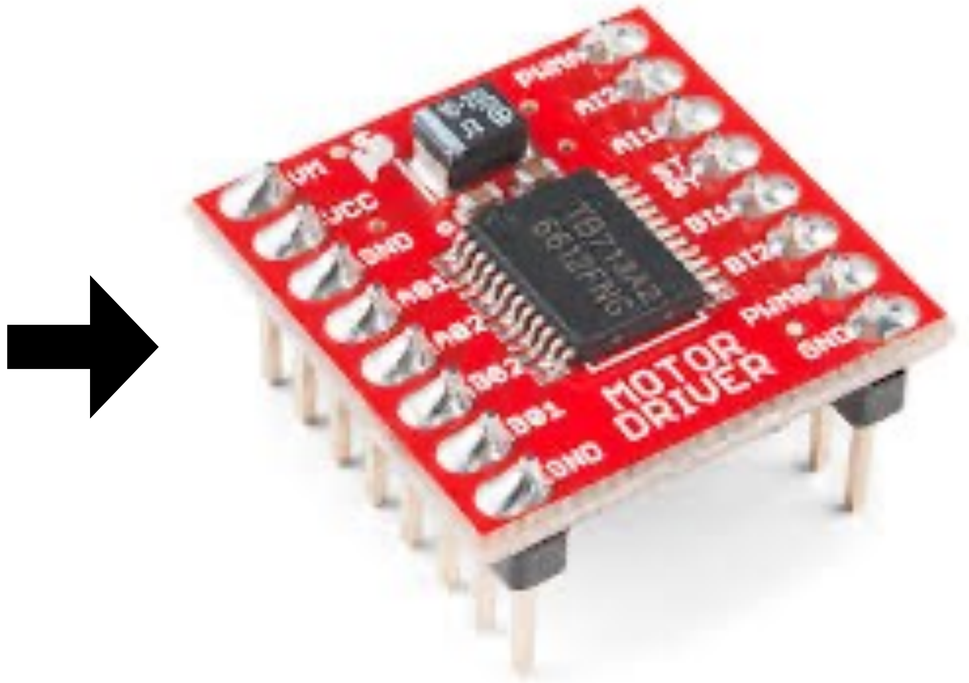
Motor Driver Control

```
void setup()
{
  Serial.begin(9600);

  //Set the PIN Modes
  pinMode(pinPWMA, OUTPUT);
  pinMode(pinAIN1, OUTPUT);
  pinMode(pinAIN2, OUTPUT);

  pinMode(pinPWMB, OUTPUT);
  pinMode(pinBIN1, OUTPUT);
  pinMode(pinBIN2, OUTPUT);

  pinMode(pinSTBY, OUTPUT);

}

void loop()
{
  motorDrive(motor1, turnCW, 255);
  delay(1000);

  motorDrive(motor1, turnCCW, 255);
  delay(1000);
}
```

https://bit.ly/3IwTolY

Motor Driver





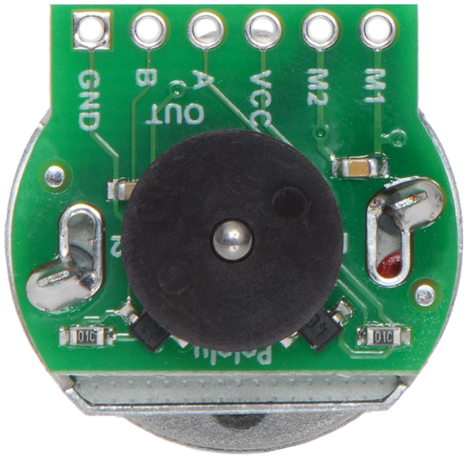
Soldering

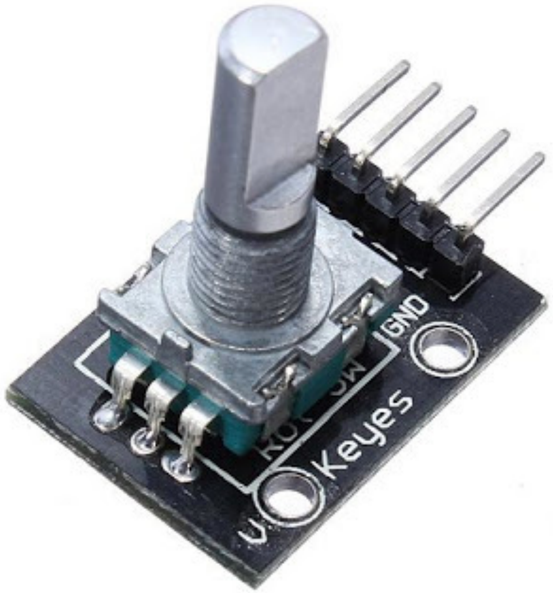
Soldering

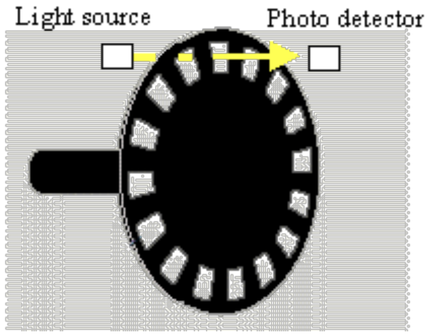
注意事項：

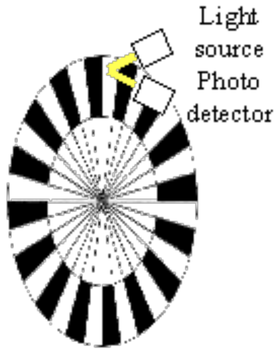
- 加熱時不能碰到前端金屬
- 加熱時，焊槍不能放置於架子之外的地方
- 海綿保持濕潤
- 氣味有害，不要靠太近

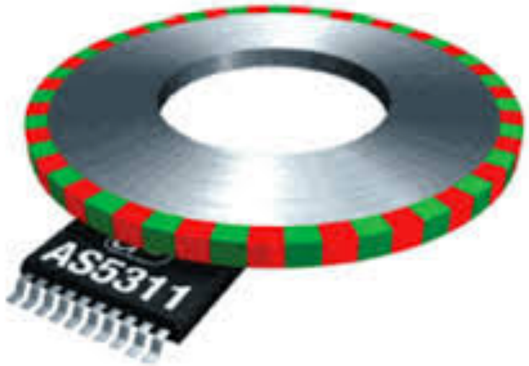
Rotary Encoder

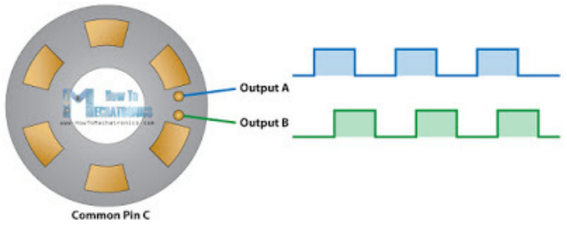




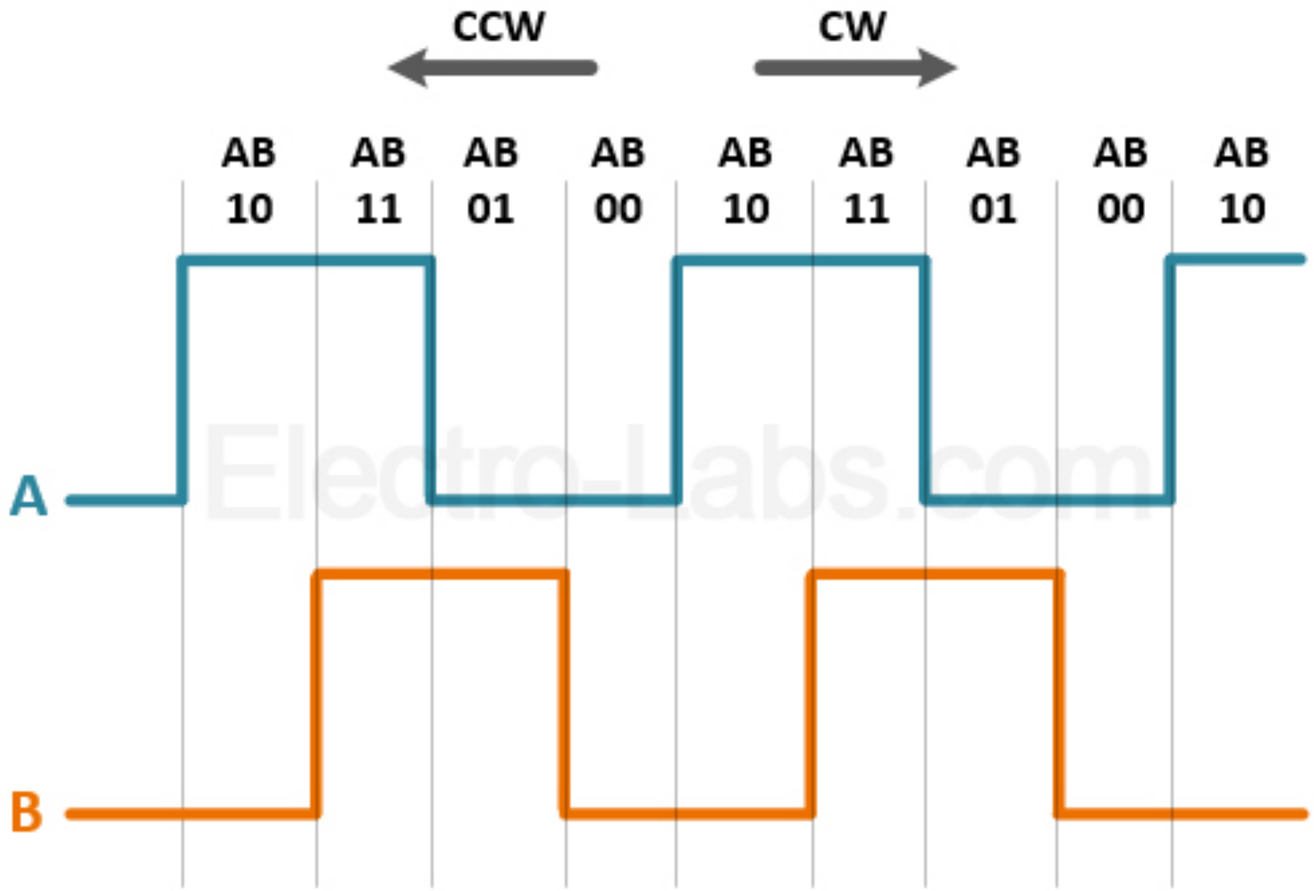








Rotary Encoder

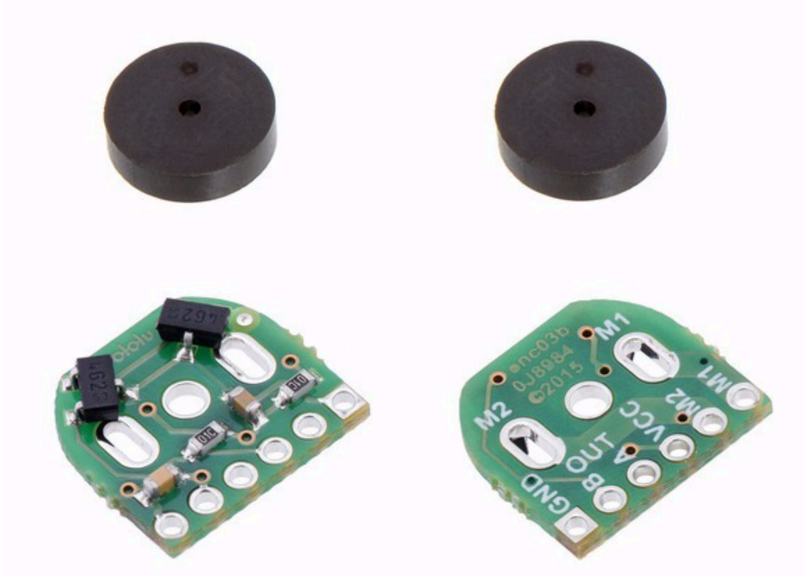


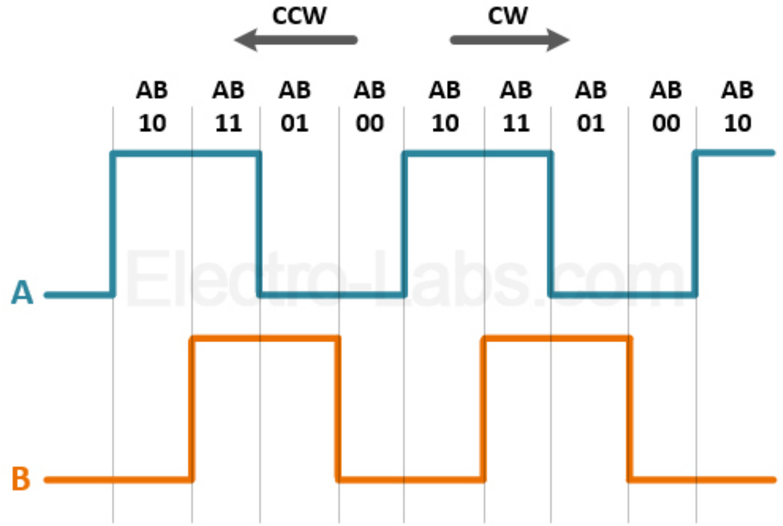
https://bit.ly/368Axkg

Rotary Encoder

new pin2	new pin1	old pin2	old pin1	Result
0	0	0	0	no movement
0	0	0	1	+1
0	0	1	0	-1
0	0	1	1	+2 (assume pin1 edges only)
0	1	0	0	-1
0	1	0	1	no movement
0	1	1	0	-2 (assume pin1 edges only)
0	1	1	1	+1
1	0	0	0	+1
1	0	0	1	-2 (assume pin1 edges only)
1	0	1	0	no movement
1	0	1	1	-1
1	1	0	0	+2 (assume pin1 edges only)
1	1	0	1	-1
1	1	1	0	+1
1	1	1	1	no movement

Rotary Encoder

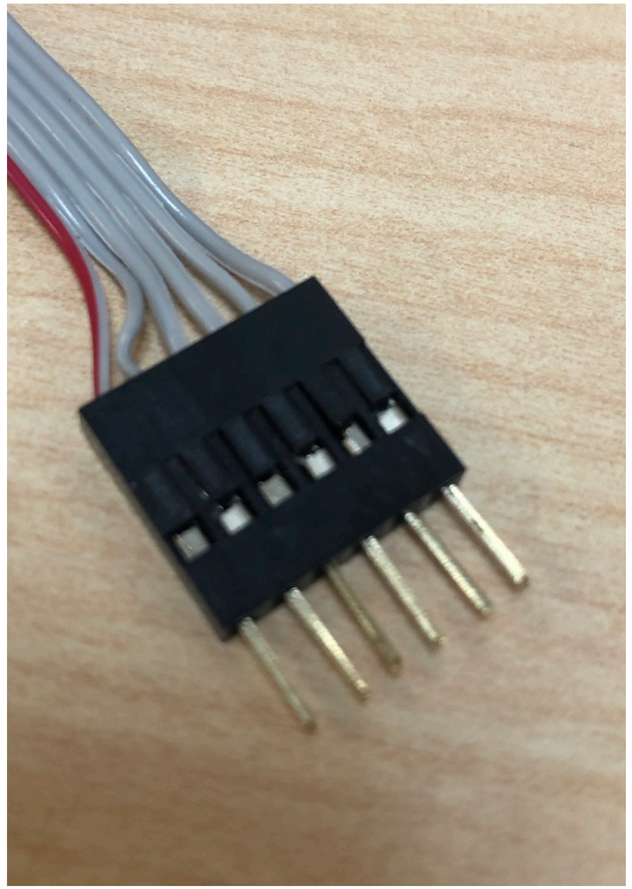


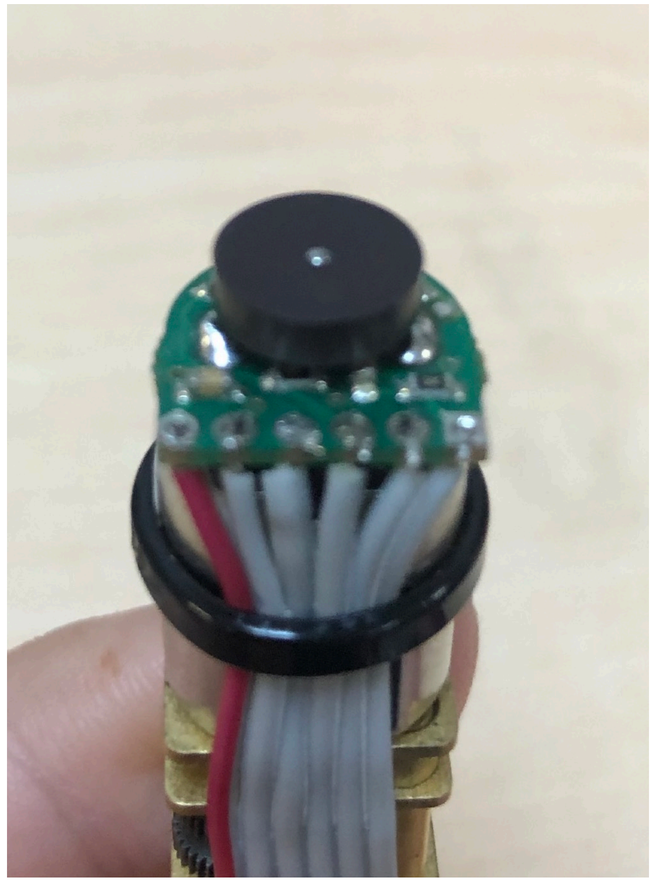


12 counts per revolution (CPR)

source : https://www.pololu.com/product/3081/pictures

Rotary Encoder





Rotary Encoder

```
int PinA = 8, PinB = 9;   // pushbutton connected to digital pin 7
int A = 0, B = 0;     // variable to store the read value
void setup()
{
  Serial.begin(9600);
pinMode(PinA, INPUT);      // sets the digital pin A as input
pinMode(PinB, INPUT);      // sets the digital pin B as input
}
void loop()
{
  A = digitalRead(PinA);   // read the input pin
  Serial.print("A: ");
  Serial.print(A);
  Serial.print(", ");
  B = digitalRead(PinB);   // read the input pin
  Serial.print("B: ");
  Serial.println(B);
}
```

https://bit.ly/3qnxbRa

Rotary Encoder

```
void loop()
{
  reA = digitalRead(PinA); // Reads the "current" state of the outputA
  reB = digitalRead(PinB);


  // If the previous and the current state of the outputA are different, that means a Pulse has occured
  if (reA != rePA || reB != rePB) {
    // If the outputB state is different to the outputA state, that means the encoder is rotating clockwise
    if(reA != rePA && reB != rePB){ // rotate too fast, miss 1 change
      recounter += 2;
    }
    else if (reA == rePB) {
      recounter ++;
    } else {
      recounter --;
    }
    Serial.print("A: ");
    Serial.print(reA);
    Serial.print(", ");
    Serial.print("B: ");
    Serial.println(reB);
    Serial.print("Position: ");
    Serial.println(recounter);
  }

  rePA = reA; // Updates the previous state of the outputA with the current state
  rePB = reB;
}
```

https://bit.ly/3ui3BgY

Rotary Encoder - Interrupt

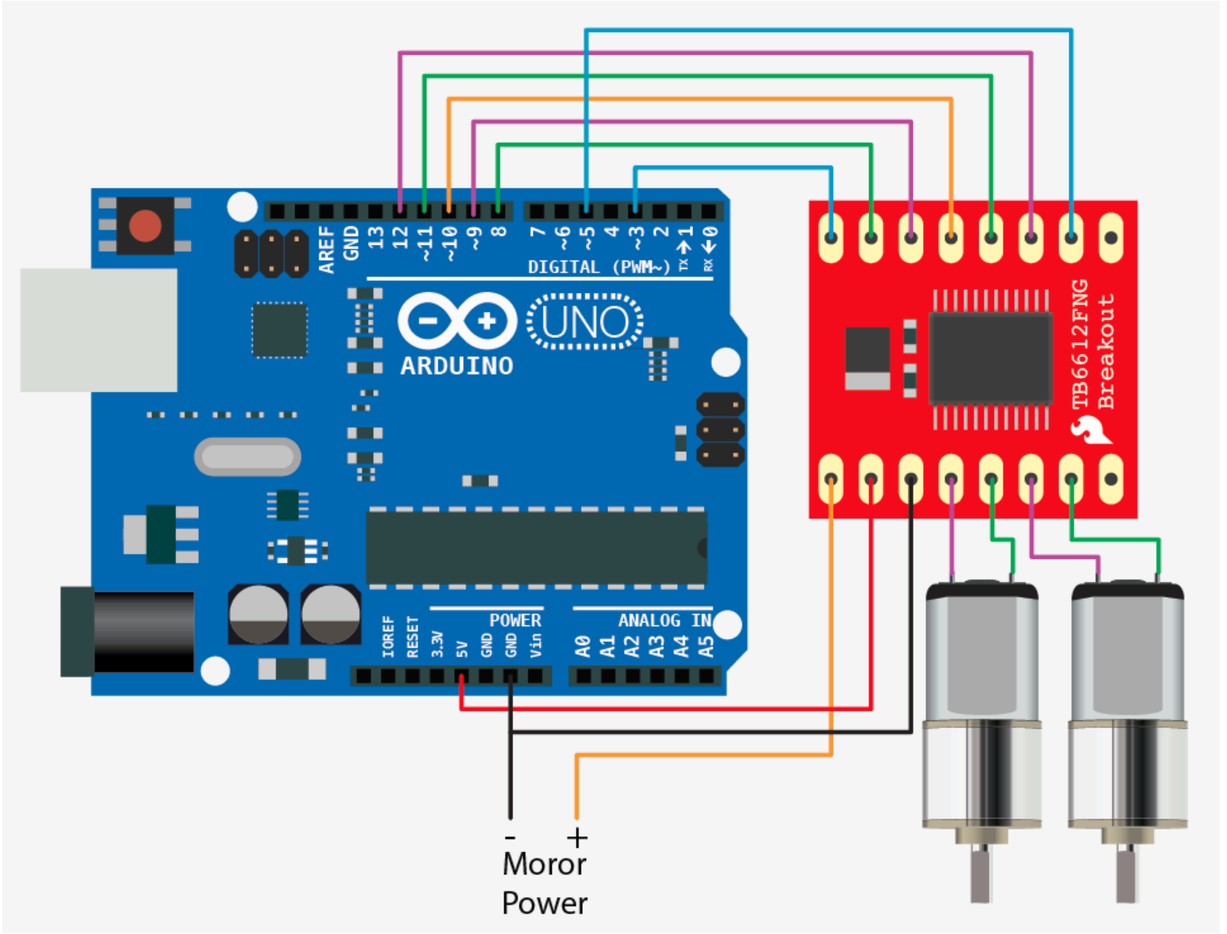
BOARD	DIGITAL PINS USABLE FOR INTERRUPTS
Uno, Nano, Mini, other 328-based	2, 3
Uno WiFi Rev.2	all digital pins
Mega, Mega2560, MegaADK	2, 3, 18, 19, 20, 21
Micro, Leonardo, other 32u4-based	0, 1, 2, 3, 7
Zero	all digital pins, except 4
MKR Family boards	0, 1, 4, 5, 6, 7, 8, 9, A1, A2
Due	all digital pins
101	all digital pins (Only pins 2, 5, 7, 8, 10, 11, 12, 13 work with CHANGE)

Rotary Encoder - Interrupt

```
#include <Encoder.h>

Encoder Enc1(2,3);

long Encodervalue;
int PastEncodervalue = -999;
void setup()
{
  Serial.begin(9600);
}

void loop()
{
  Encodervalue = Enc1.read();
  if(Encodervalue != PastEncodervalue){
    Serial.println(Encodervalue);
    PastEncodervalue = Encodervalue;
  }
}
```

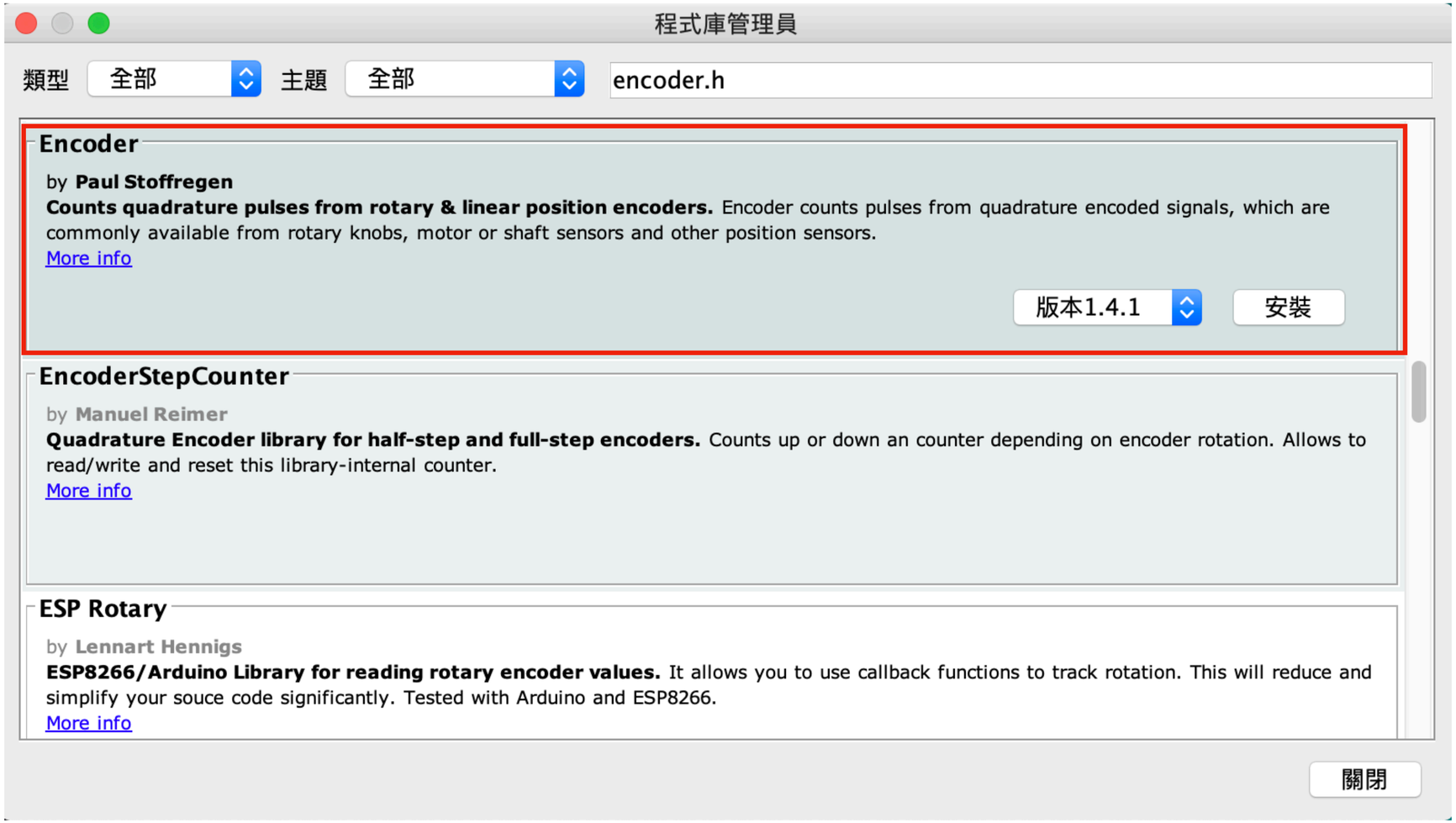
https://bit.ly/368AWDi

DC Motor Control



https://bit.ly/3qnAisi

Encoder.h


程式庫管理員
類型
全部
主題
全部
encoder.h
Encoder
by Paul Stoffregen
Counts quadrature pulses from rotary & linear position encoders. Encoder counts pulses from quadrature encoded signals, which are commonly available from rotary knobs, motor or shaft sensors and other position sensors.
More info
版本1.4.1
安裝
EncoderStepCounter
by Manuel Reimer
Quadrature Encoder library for half-step and full-step encoders. Counts up or down an counter depending on encoder rotation. Allows to read/write and reset this library-internal counter.
More info
ESP Rotary
by Lennart Hennigs
ESP8266/Arduino Library for reading rotary encoder values. It allows you to use callback functions to track rotation. This will reduce and simplify your souce code significantly. Tested with Arduino and ESP8266.
More info
關閉

DC Motor Control

```
#include <Encoder.h>

// pin2, pin3 are interrupt pin
Encoder Enc1(2,4);
Encoder Enc2(3,7);

long newPosition1;
long oldPosition1  = -999;

//Define the Pins
//Motor 1
int pinAIN1 = 9; //Direction
int pinAIN2 = 8; //Direction
int pinPWMA = 5; //Speed

//Motor 2
int pinBIN1 = 11; //Direction
int pinBIN2 = 12; //Direction
int pinPWMB = 6; //Speed

//Standby
int pinSTBY = 10;

// Set motor variable
int cpr = 12;
int gearratio = 211;
```

https://bit.ly/3qnAisi

DC Motor Control

```
void motor_rotate(float loopnumber) {
  //int rotationspeed;//, loopnumber = 3;
  long current = oldPosition1;
  long target = current + (loopnumber * cpr * gearratio);
  if (loopnumber >= 0){
    motorDrive(motor1, turnCCW, 255);
    while (oldPosition1 < target) {
      read_encoder();
    }
    motorBrake(motor1);
  }
  else{
    motorDrive(motor1, turnCW, 255);
    while (oldPosition1 > target) {
      read_encoder();
    }
    motorBrake(motor1);
  }
}
```

https://bit.ly/3qnAisi

How about parallel control of two DC motors?

How about parallel control of two DC motors?

https://bit.ly/37Xd0Ui

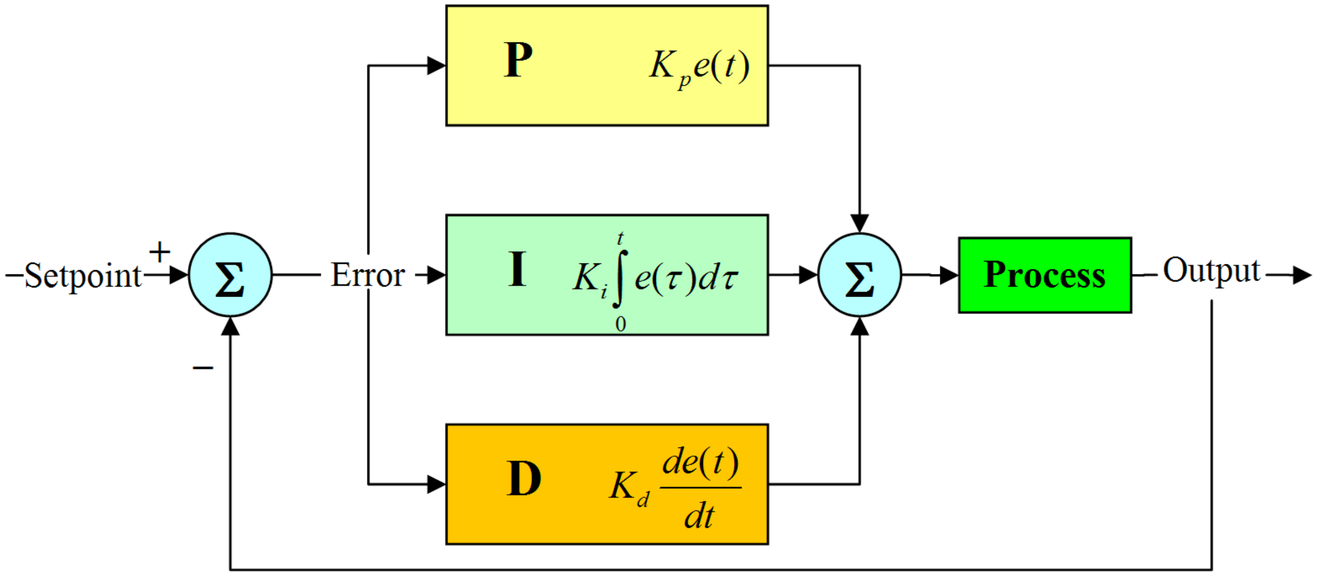
PID Controller

$$u(t) = K_\mathrm{p} e(t) + K_\mathrm{i} \int_0^t e(t')\,dt' + K_\mathrm{d} \frac{de(t)}{dt},$$

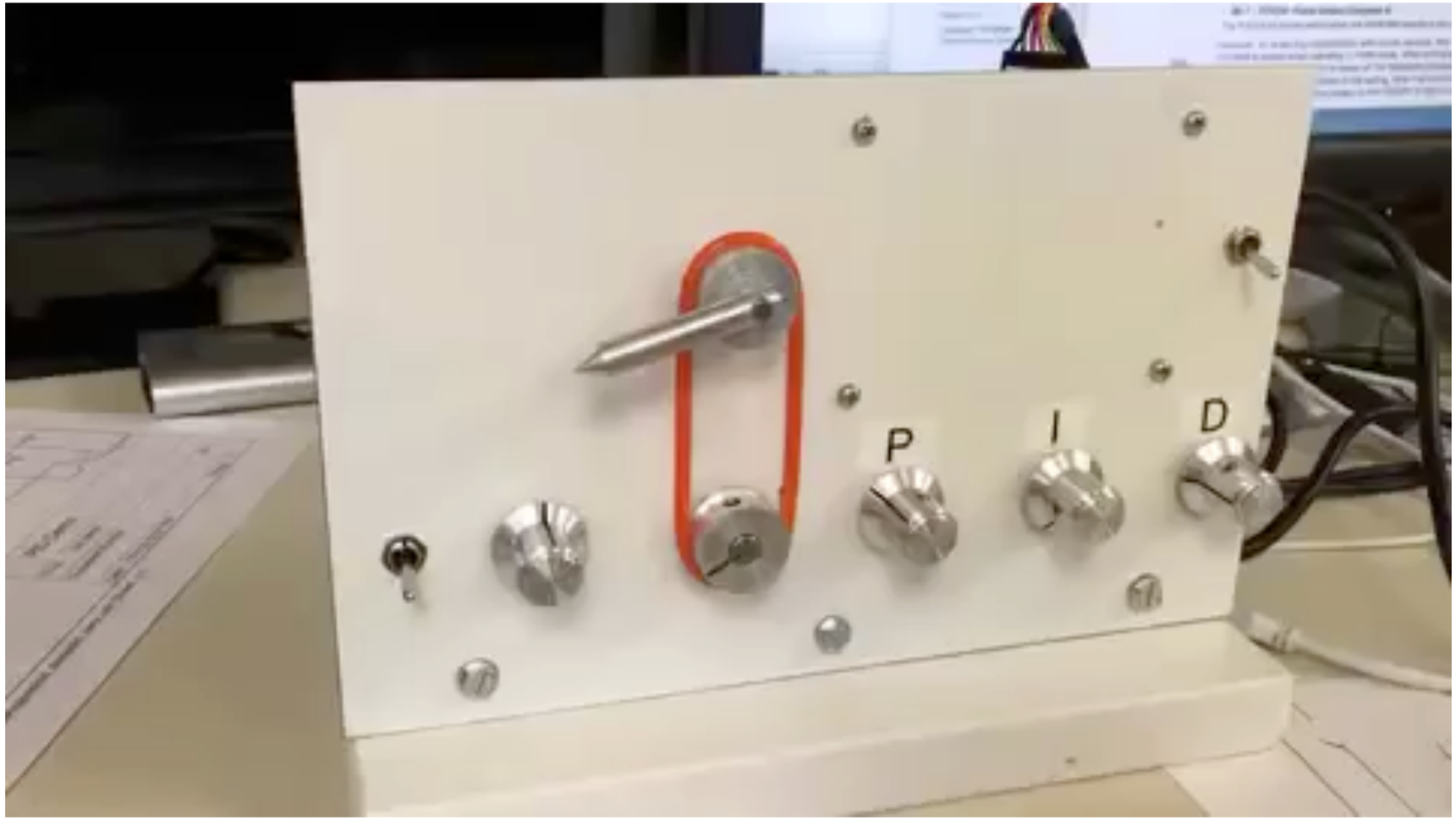
K_p is the proportional gain, a tuning parameter,

K_i is the integral gain, a tuning parameter,

K_d is the derivative gain, a tuning parameter,



PID Controller



http://grauonline.de/alexwww/ardumower/pid/pid.html

PID Controller

```
void motor_rotate(float bal)
{
  //Serial.println(bal);
  targetCount = bal * encoderValue;
  Serial.print("targetCount");
  Serial.println(targetCount);
  integral = 0; //reset integral in PID control
  motorActive = true;
}
```

https://bit.ly/3JFTPvu

PID Controller

```
void getVoltageFeedFromPID()
{
  int error;
  float I, D;

  error = targetCount - newPosition1;
  // Integral
  if (abs(error) < KiActivationErrorRange) {
    integral += error * usPassed;
    if (abs(error) < KiRange2) {
      I = Ki * KiRange2Factor * integral;
    } else {
      I = Ki * integral;
    }
  } else {
    integral = 0;
    I = 0;
  }
  //Derivative
  if (abs(error) < KdActivationErrorRange) {
    D = Kd * encoderSpeed;
  } else {
    D = 0;
  }
```

$$u(t) = K_\mathrm{p} e(t) + K_\mathrm{i} \int_0^t e(t')\,dt' + K_\mathrm{d} \frac{de(t)}{dt}$$

https://bit.ly/3JFTPvu

PID Controller

```
//Derive driving voltage
if (abs(targetCount - newPosition1) > activationCountError) {
  motorActive = true;
}
if (abs(targetCount - newPosition1) > acceptableCountError && motorActive) {
  voltageFeed = Kp * (error) + I + D;

  if (voltageFeed > 0) {
    if (voltageFeed > maximalVoltage) {
      voltageFeed = maximalVoltage;
    }
    //motorDrive(motor1, turnDirection, voltageFeed);
  } else {
    if (voltageFeed < -maximalVoltage) {
      voltageFeed = -maximalVoltage;
    }
    //motorDrive(motor1, !turnDirection, -voltageFeed);
  }
} else {
  integral = 0;
  voltageFeed = 0;
}
```

$$u(t) = K_\mathrm{p} e(t) + K_\mathrm{i} \int_0^t e(t')\,dt' + K_\mathrm{d} \frac{de(t)}{dt}$$

https://bit.ly/3JFTPvu

PID Controller

$$u(t) = K_\mathrm{p} e(t) + K_\mathrm{i} \int_0^t e(t')\,dt' + K_\mathrm{d} \frac{de(t)}{dt},$$

```
const float Kp =
const float Ki =
const float Kd =
```

?

https://bit.ly/3JFTPvu

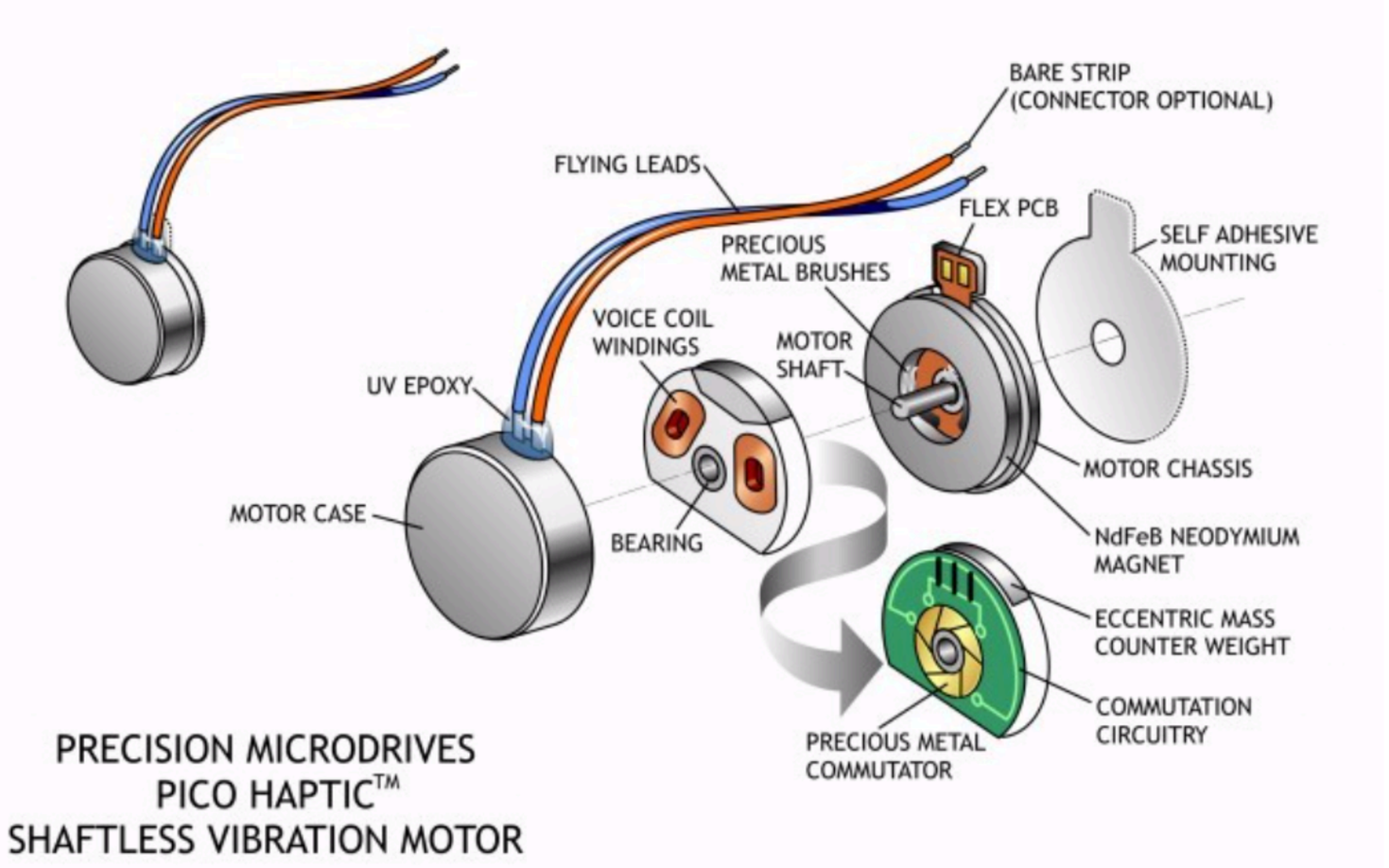
PID Controller

https://youtu.be/AN3yxIBAxTA

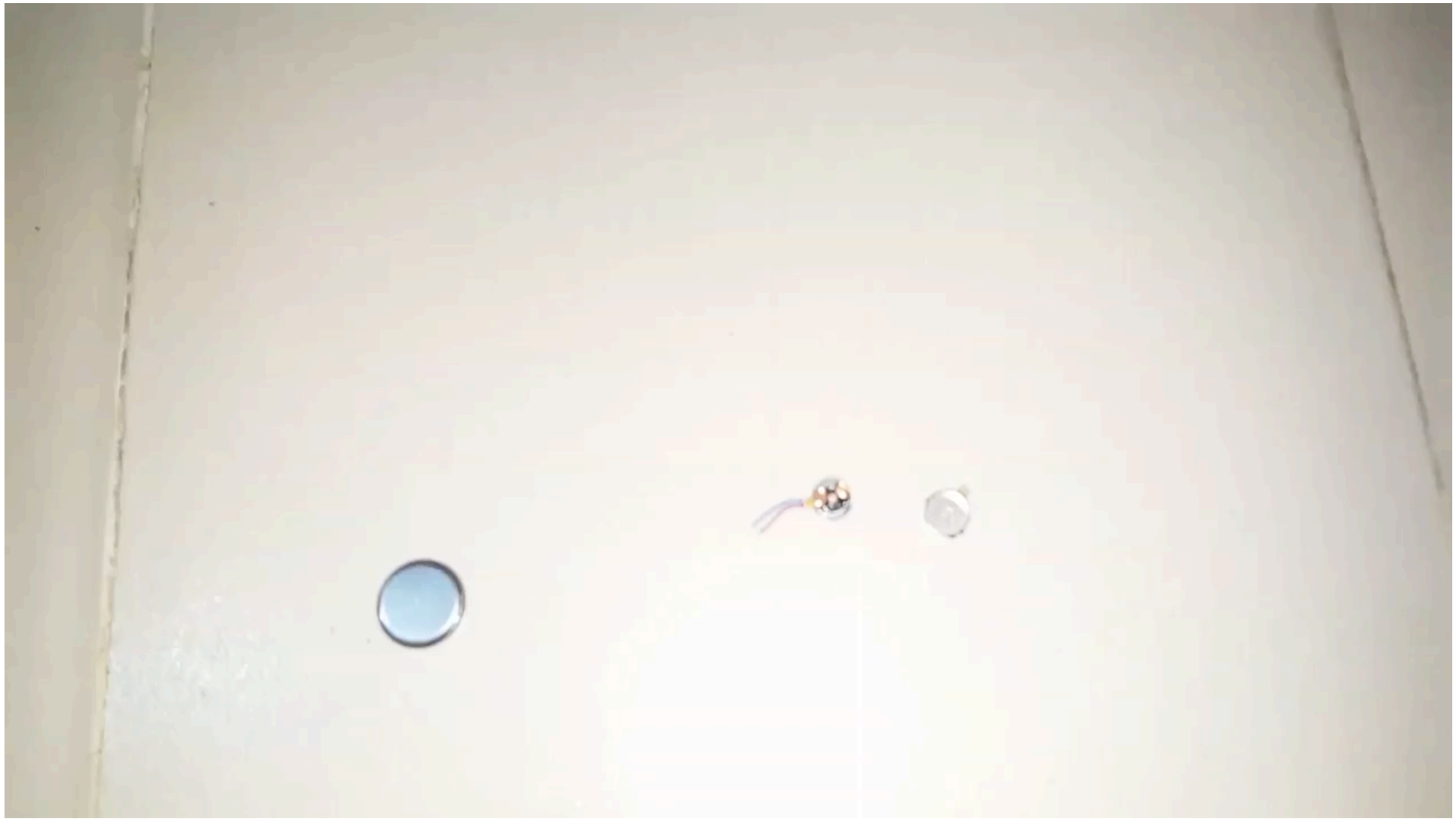
Vibration Motor

source: https://www.precisionmicrodrives.com

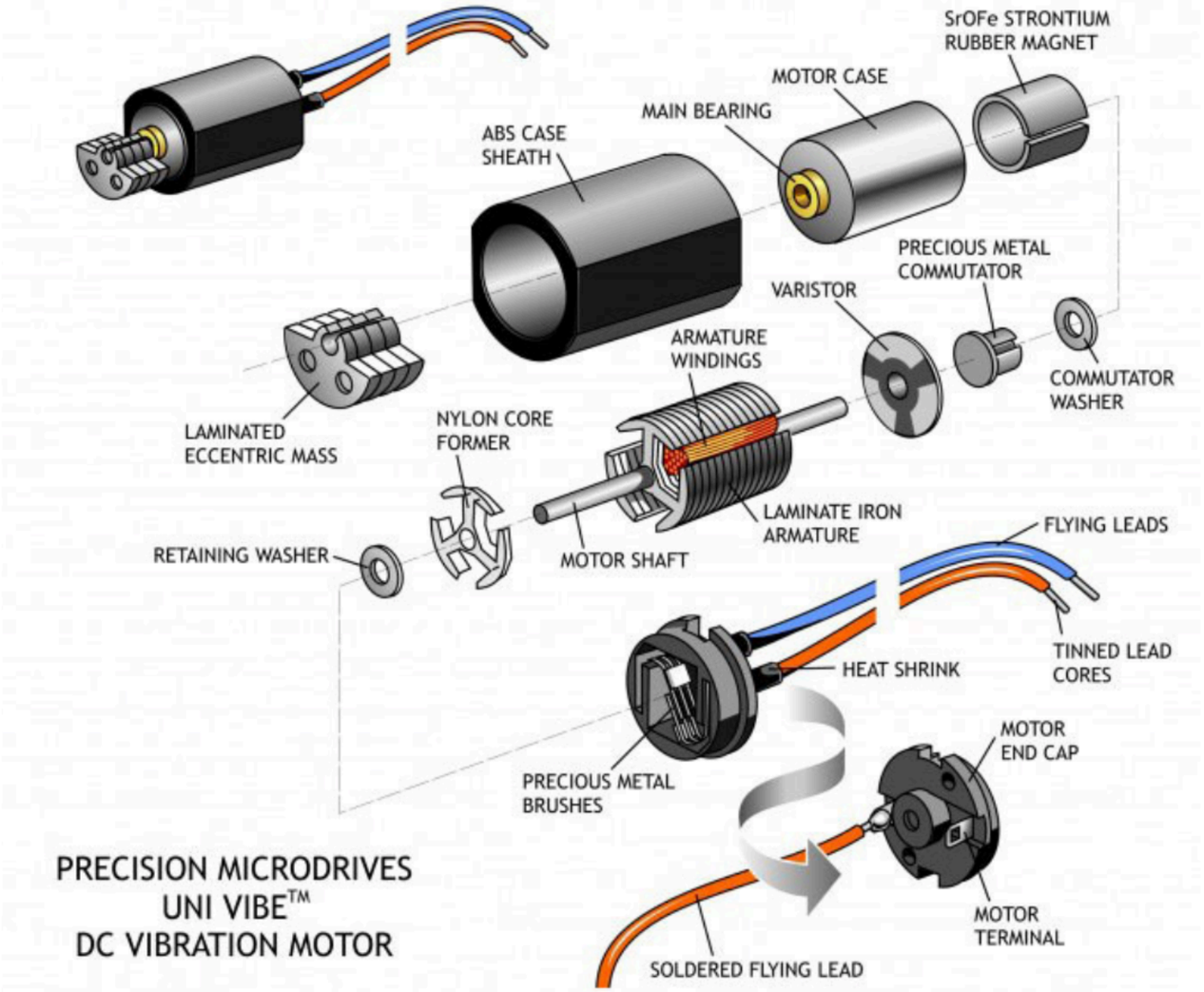
Vibration Motor



Vibration Motor



Vibration Motor



Vibration Motor

https://youtu.be/WWgN20Xx-5A

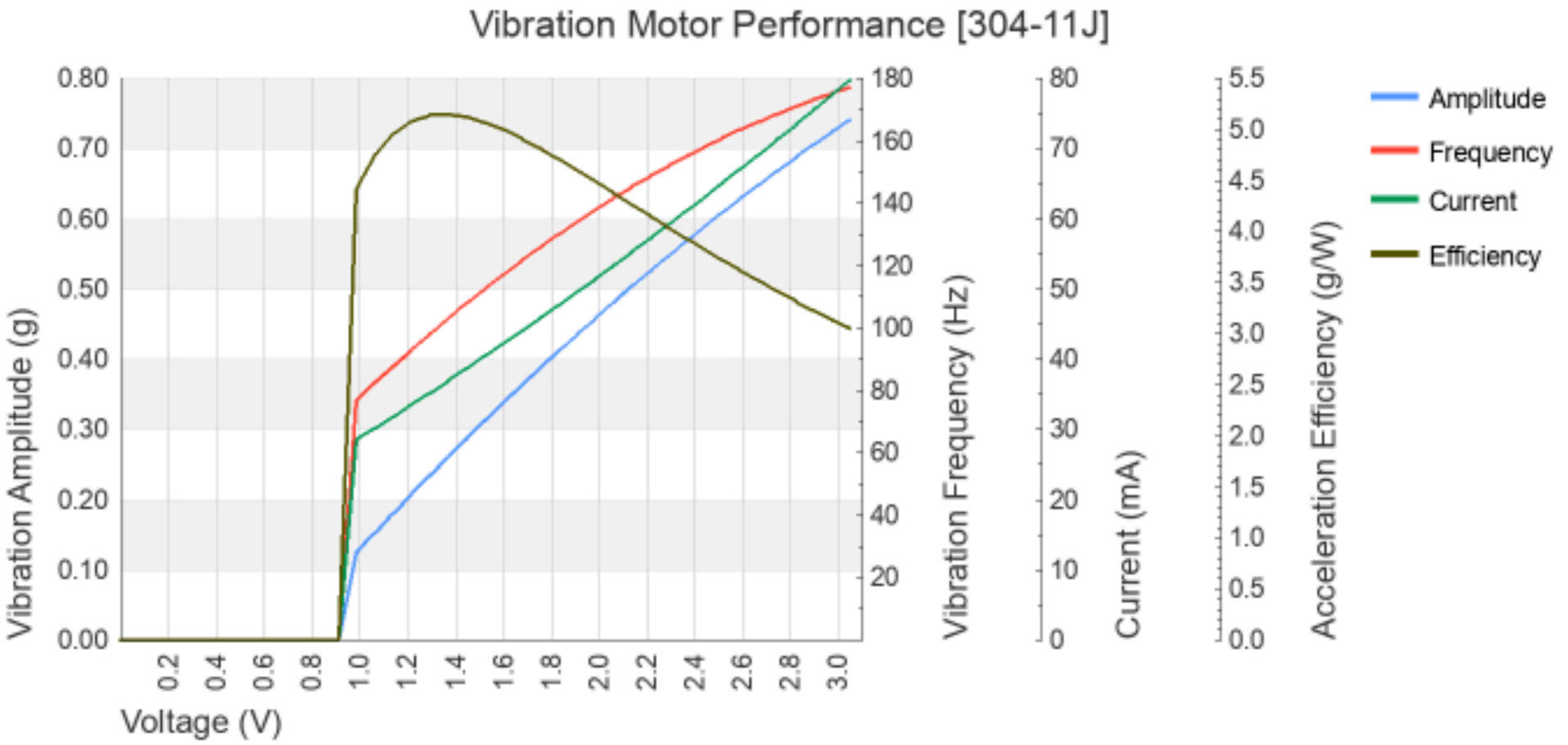
Vibration Motor

Key Features

Body Diameter	4.5 mm [+/- 0.2]
Body Length	6.2 mm [+/- 0.2]
Ecc. Weight Radius	2.1 mm [+/- 0.1]
Ecc. Weight Length	3 mm [+/- 0.1]
Rated Operating Voltage	3 V
Rated Vibration Speed	10,400 rpm [+/- 2,100]
Typical Rated Operating Current	78 mA
Typical Norm. Amplitude	0.72 G

source: https://www.precisionmicrodrives.com

Vibration Motor



source: https://www.precisionmicrodrives.com

Vibration Motor

```
void loop() {
  digitalWrite(pinSTBY, HIGH);

  if (Serial.available() > 0) {
    char inst;
    inst = Serial.read();

    if (inst == '1'){
      digitalWrite(pinAIN1, HIGH);
      digitalWrite(pinAIN2, LOW);
      analogWrite(pinPWMA, 255);
    }
    else if (inst == '2'){
      digitalWrite(pinBIN1, HIGH);
      digitalWrite(pinBIN2, LOW);
      analogWrite(pinPWMB, 255);
    }
  }
  digitalWrite(pinSTBY, LOW);
}
```

https://bit.ly/3qpkIfU