C++ Coding

"if()" Statement

```
while(1) {
    switchstate = pushbutton.read();
    if(switchstate) {
        switchcount++;  // increment the switchcount value
        if(ledindex==1)
            led1 = 1;      // turn ON LED #1 if indexcount is 1
        else if(ledindex==2)
            led2 = 1;      // turn ON LED #2 if indexcount is 2
        ...
    }
    ...
}
```

"switch()" Statement

```
while(1) {
    switchstate = pushbutton.read();
    if(switchstate) {
        switchcount++;     // increment the switchcount value if the button is
pressed
        switch(ledindex) {  // Integer variable or equation to be evaluated
            case 1:
                led1 = 1;      // turn ON LED #1 if indexcount is 1
                break;
            case 2:
                led2 = 1;      // turn ON LED #2 if indexcount is 2
                break;
            default:
                led3 = 1;      // turn ON LED #3 if indexcount is anything else
                break;
        }
    }
    ,,,
}
```

Digital In & Out

Clarifications

```
DigitalOut variable_name(pin_label);
DigitalIn variable_name(pin_label);
//For the pin lable, use p5, p10, p24, LED2, etc.
```

Digital In

```
DigitalOut var(p5);

var = 1;        // set pin p5 HI
var = 0;        // set pin p5 LO
var.write(1);  // set pin p5 HI
var.write(0);  // set pin p5 LO
```

Digital Out

```
DigitalIn var(p5);
bool result;

result = var;         // assign value of var to result
result = var.read();  // assign value of var to result
```

Wait

```
wait(0.5);
```

Analog In & Out

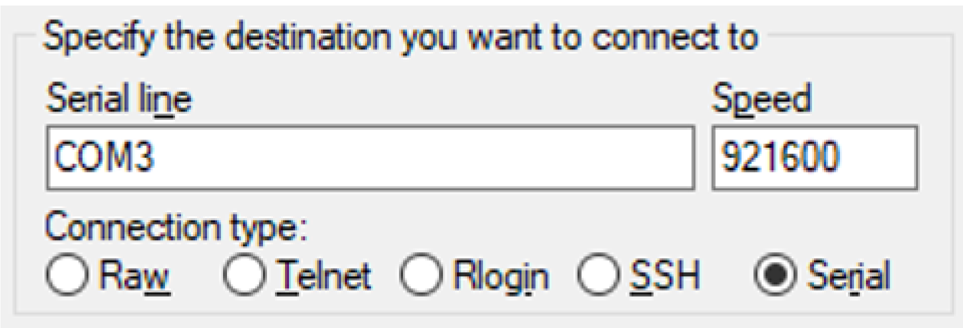
Clarifications

```
AnalogIn variable_name(pin_label);
AnalogIn var(p17);
float result_f;
int   result_i;

result_f = var.read();      // read floating point input
result_i = var.read_u16(); // read 16-bit integer input
```

PUTTY

Settings



Codes

```
#include "mbed.h"
...
Serial pc(USBTX, USBRX);  // USB serial interface

int main() {
  ...
  while(1) {
    pc.baud(921600);//Speed
    pc.printf(...);//Print
    ...
  }
}
```

Pulse Width Modulation (PWM)

Pulse Width Modulation (PWM) is a method to create an *analogue-like* (but NOT truly) signal by using full-ON and full-OFF digital signals.

PWM period

PWM period is the total time of the pulse, which equals to the **ON time + OFF time**.

Duty Cycle is the percentage of ON time. **Duty cycle = ON time / period**.

Coding

Mbed use functions to handle ON time and OFF time internally, and no calculation is needed.

```
PwmOut var(p21);   // declare PWM output pin

var.period(0.020); // set period = 20ms

var.write(0.00);   // set duty cycle =   0%
var.write(0.50);   // set duty cycle =  50%
var.write(0.90);   // set duty cycle =  90%
var.write(1.00);   // set duty cycle = 100%
```

Note that for a fixed period, the `.period()` function need only be called once.

Circuit

LDR

Light Dependent Resistor (LDR), which resistance changes with the brightness of its environment.

- Darker -> Higher resistance
- Brighter -> Lower resistance

Potentiometers

- Fixed resistance value between the two end ports

• Variable resistance between each end and the middle 'wiper' connection

$R_{pot} = R_{port1->wiper} + R_{wiper->port2}$

Light Bulbs

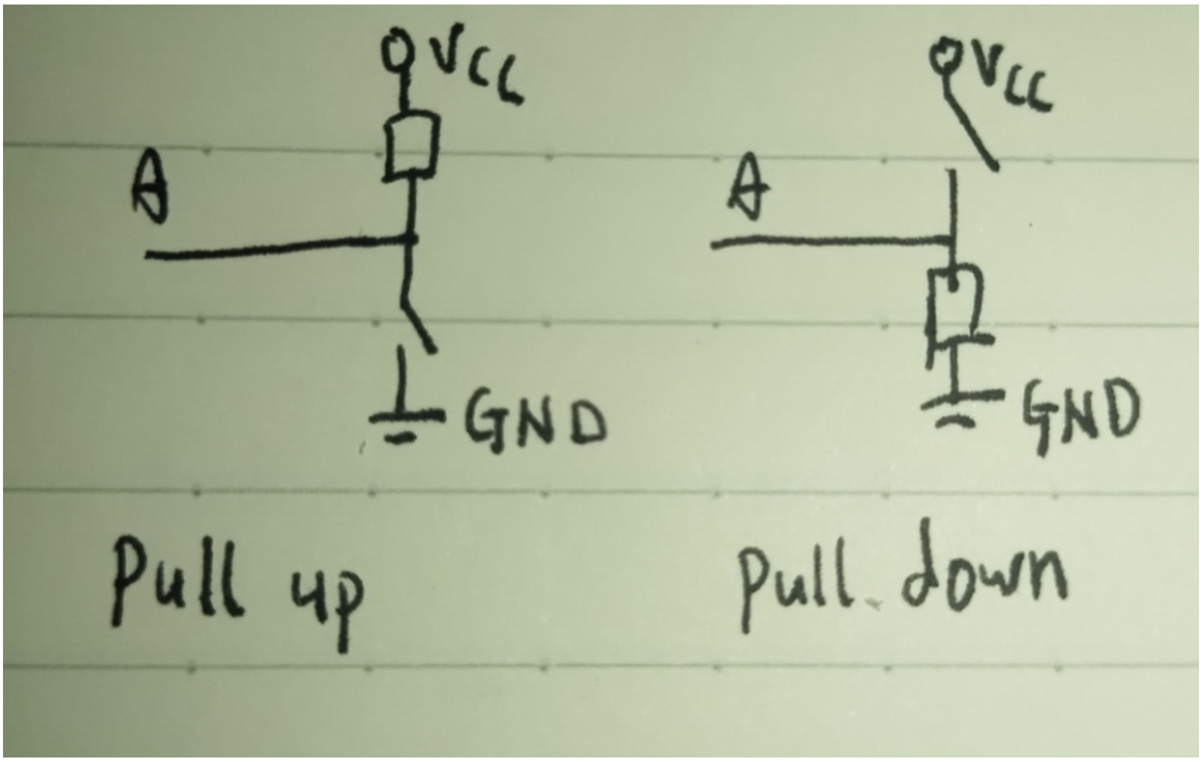
Incandescent lamps (aka light bulbs) convert current into illumination and heat.

When current is provided, the internal element heats and then glows. The brightness is related with current flow

Pull-up and Pull-down Circuits

Pull-up: $V_{ref} -> resistor -> signal -> switch -> ground$

Pull-down: $V_{ref} -> switch -> signal -> resistor -> ground$

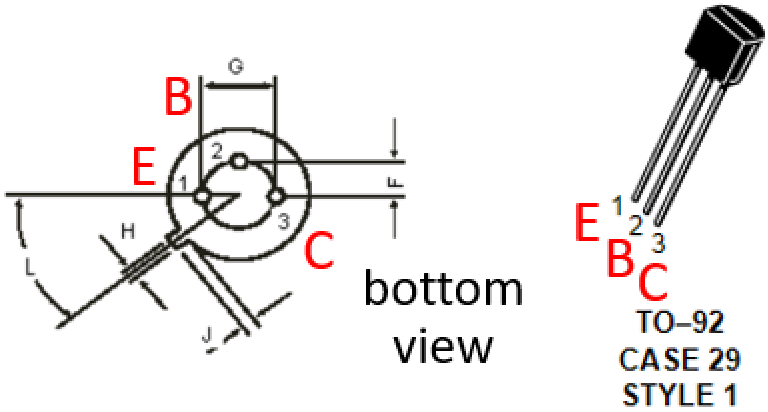


Transistors

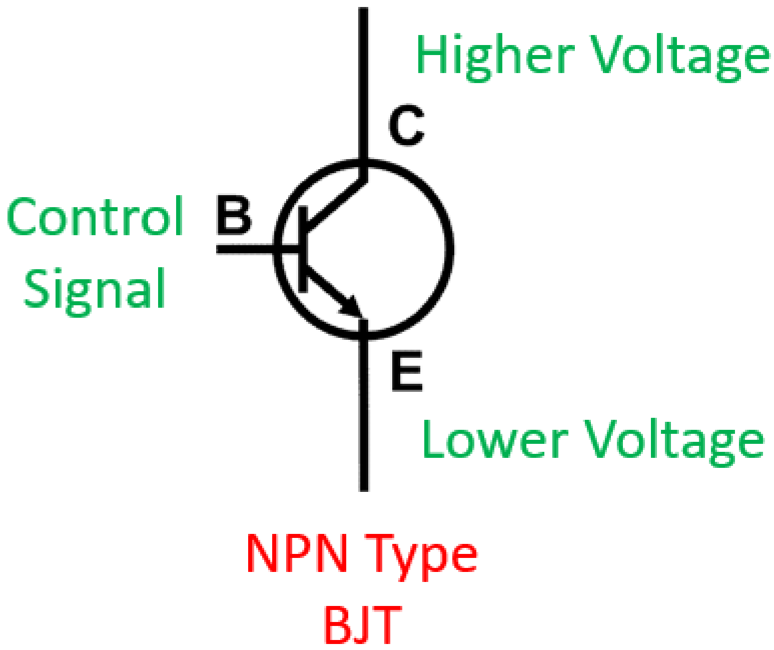
Transistors are semiconductor devices that regulate circuit current flow based on a control signal。

A common transistor style is the Bi-Junction Transistor (BJT)

The three pins of a transistor are denoted by: **Emitter** (E), **Collector** (C) and **Base** (B).



NPN Type



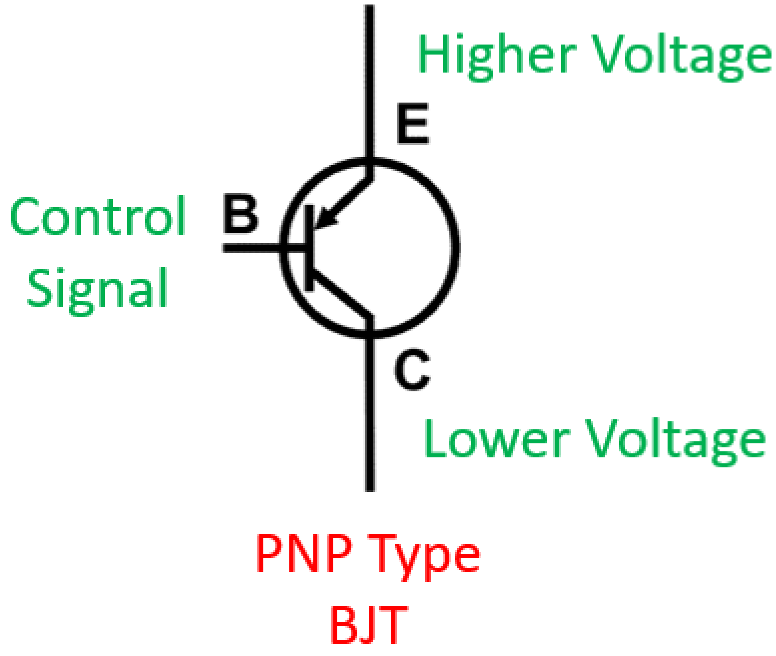
When $V_B > V_E + V_f(0)$, the transistor is **ON**, and current is **allowed** to flow from **C** to **E**.

When $V_B < V_E + V_f(0)$, the transistor is **OFF**, and current is **not allowed** to flow from **C** to **E**.

PNP Type

When $V_E > V_B + V_f(0)$, the transistor is **ON**, and current is **allowed** to flow from **E** to **C**.

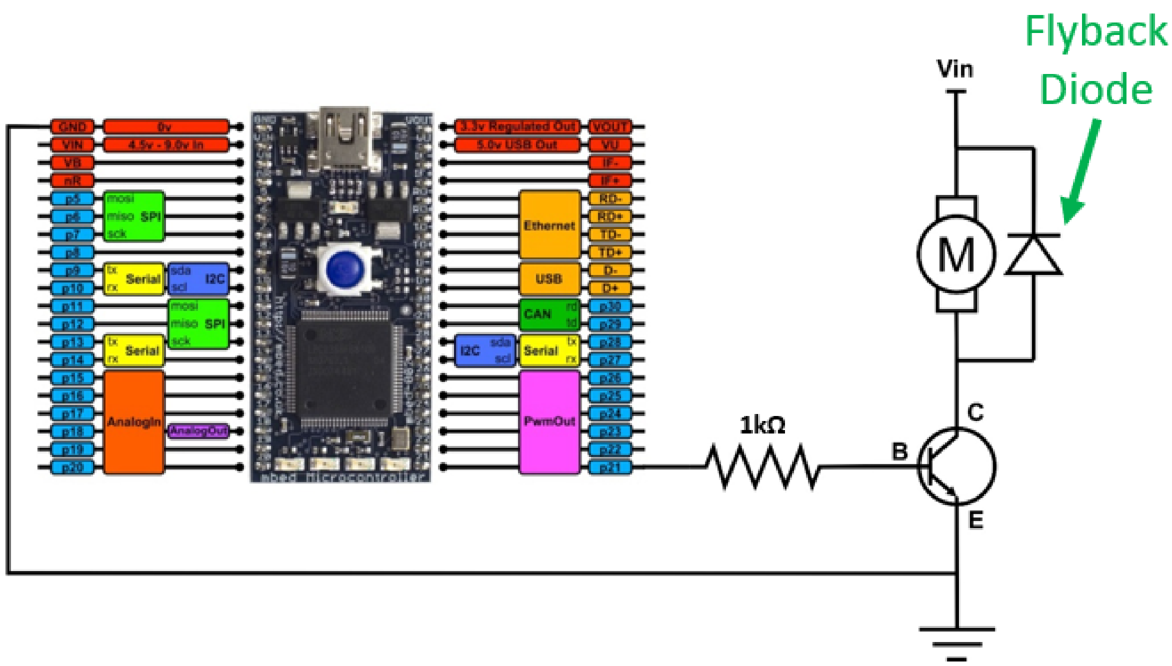
When $V_E < V_B + V_f(0)$, the transistor is **OFF**, and current is **not allowed** to flow from **E** to **C**.



Circuit Example (NPN Type)

Transistor connections:

- Collector – DC Motor
- Base – Microcontroller via resistor
- Emitter – Ground



Note that the resistor on **B** is necessary.

MOSFET

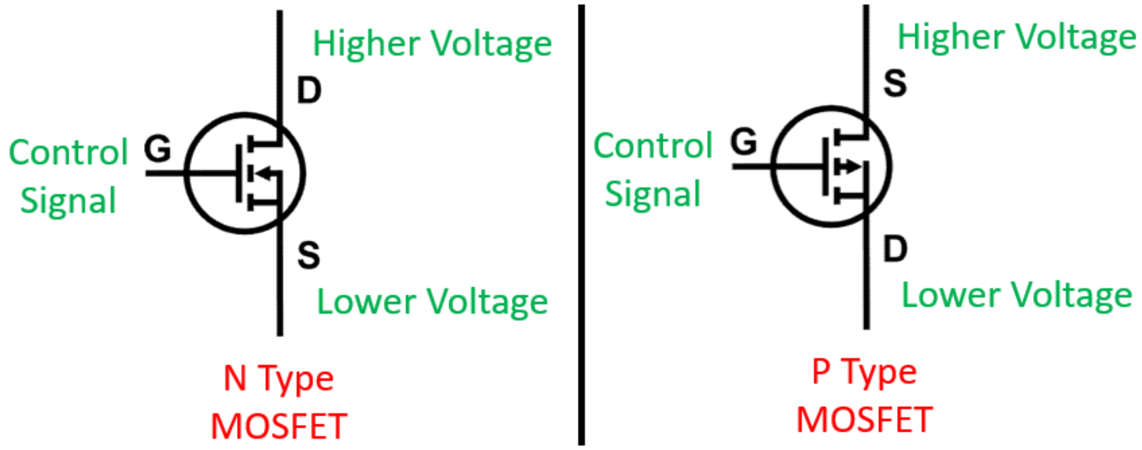
MOSFETs are:

- Constructed slightly differently from BJTs
- Capable of higher voltages and currents

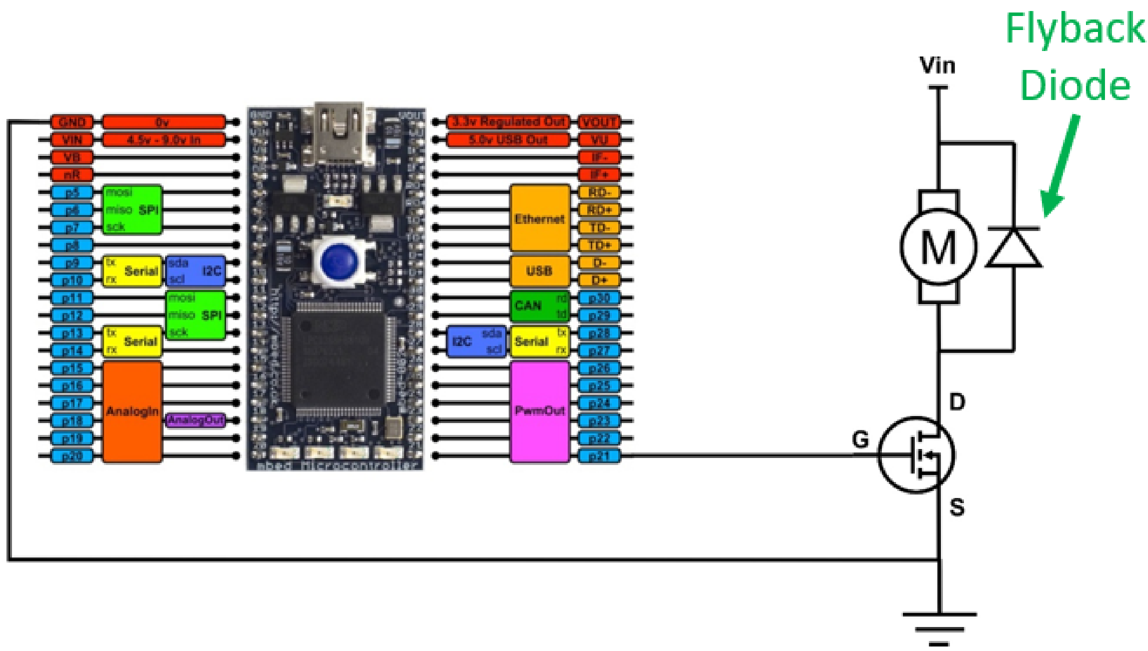
- Controlled in the same manner

The three pins of a transistor are denoted by: **Source** (S), **Gate** (G) and **Drain** (D).

Like transistors, MOSFETs have two types: N-Channel type and P-Channel type.

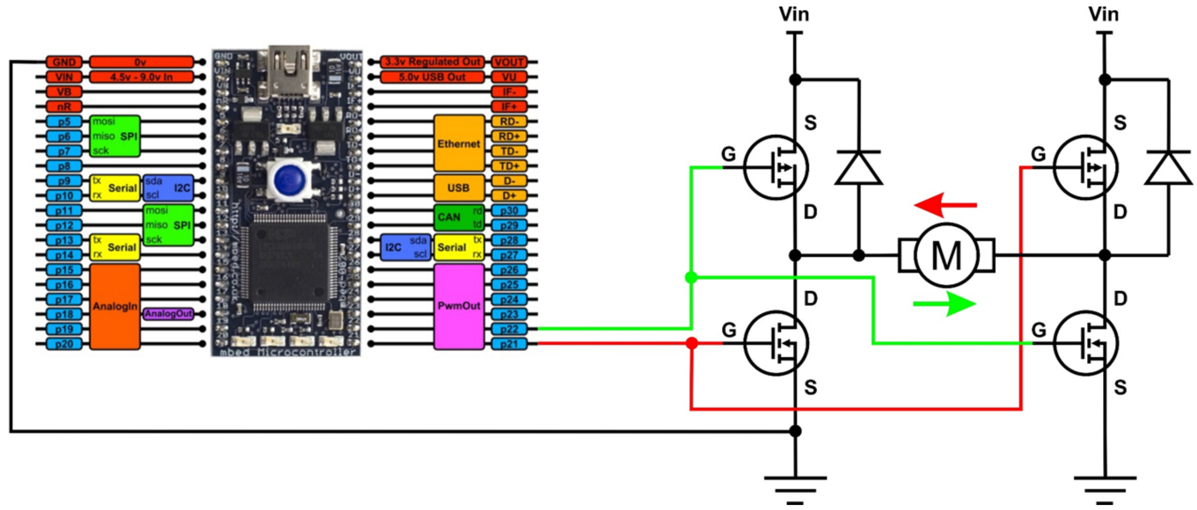


Circuit Example (N-Type MOSFET)



H-bridge Circuit

Traditional Circuit



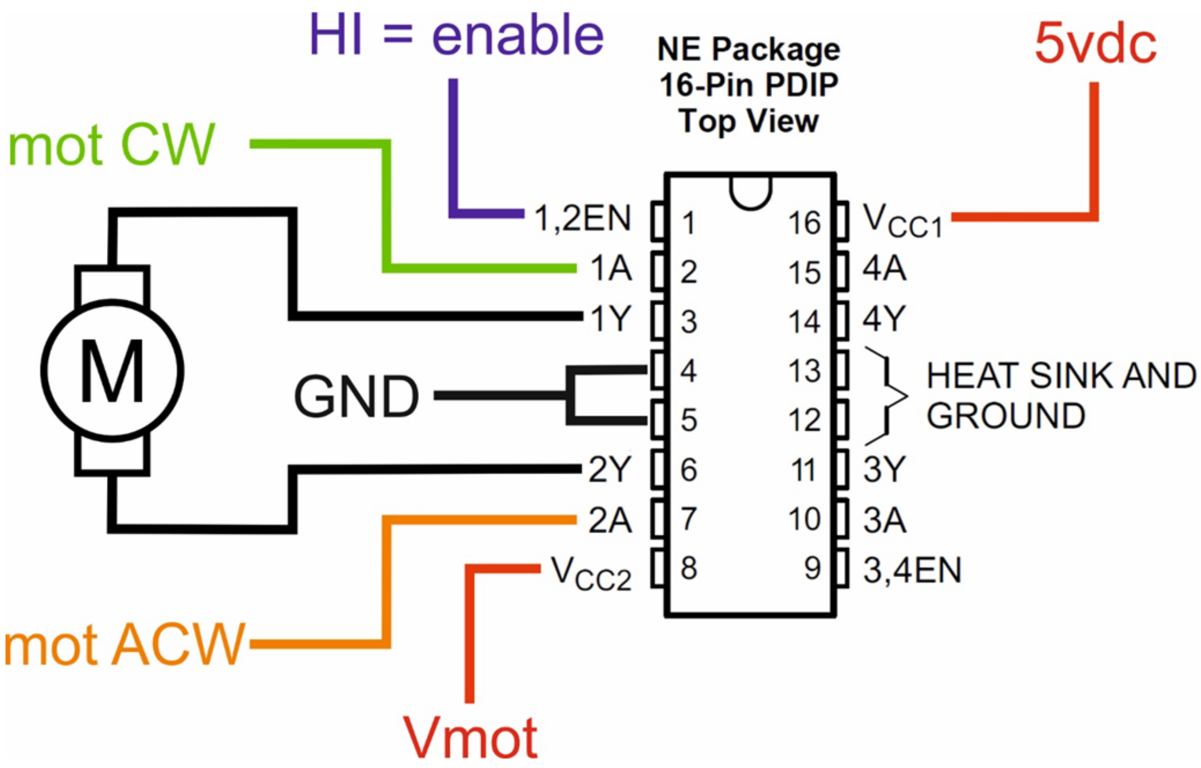
The H-bridge circuit is used to control the speed and direction of a DC motor.

When the signal output of `p22` (which is the green line) is "HI" and the signal output of `p21` (which is the red line) is "LO", the current will flow from left to right, and the motor will rotate clockwise (just for example ;)).

When the signal output of `p22` is "LO" and the signal output of `p21` is "HI", the current will flow from right to left, and the motor will rotate anti-clockwise(just for example).

Note that the signal output of `p21` and `p22` **CAN NEVER BE** "HI" at the same time, or the power unit of the H-bridge will be damaged.

L293D Motor Controller



L293D can be connected as follows:

- VCC1 – Logic power (+5vdc)
- VCC2 – Motor power
- GND – Common circuit Ground
- 1,2EN – HI level enable signal (the key of the circuit)

- 1A – CW(clockwise) motor control signal
- 2A – ACW(anti-clockwise) motor control signal
- 1Y – Motor terminal #1
- 2Y – Motor terminal #2

There are three methods to control the speed of the motor:

Method 1 (with fewest connections)

- 1,2EN – connected to **+5vdc**
- 1A – Microcontroller **PWM** pin
- 2A – Microcontroller **PWM** pin

Pro:

- Fewest microcontroller connection

Cons:

- Motor is always 'on', more power
- Costly to expand with an extra motor

Method 2 (easily controllable)

- 1,2EN – Microcontroller **Digital** pin
- 1A – Microcontroller **PWM** pin
- 2A – Microcontroller **PWM** pin

Pro:

- Enable/disable motor, saves power

Cons:

- More control code needed
- Costly to expand with an extra motor

Method 3 (best method)

- 1,2EN – Microcontroller **PWM** pin
- 1A – Microcontroller **Digital** pin
- 2A – Microcontroller **Digital** pin

Pro:

- Enable/disable motor, saves power
- Only 1 PWM pin, easy to expand

Cons:

- More control code needed