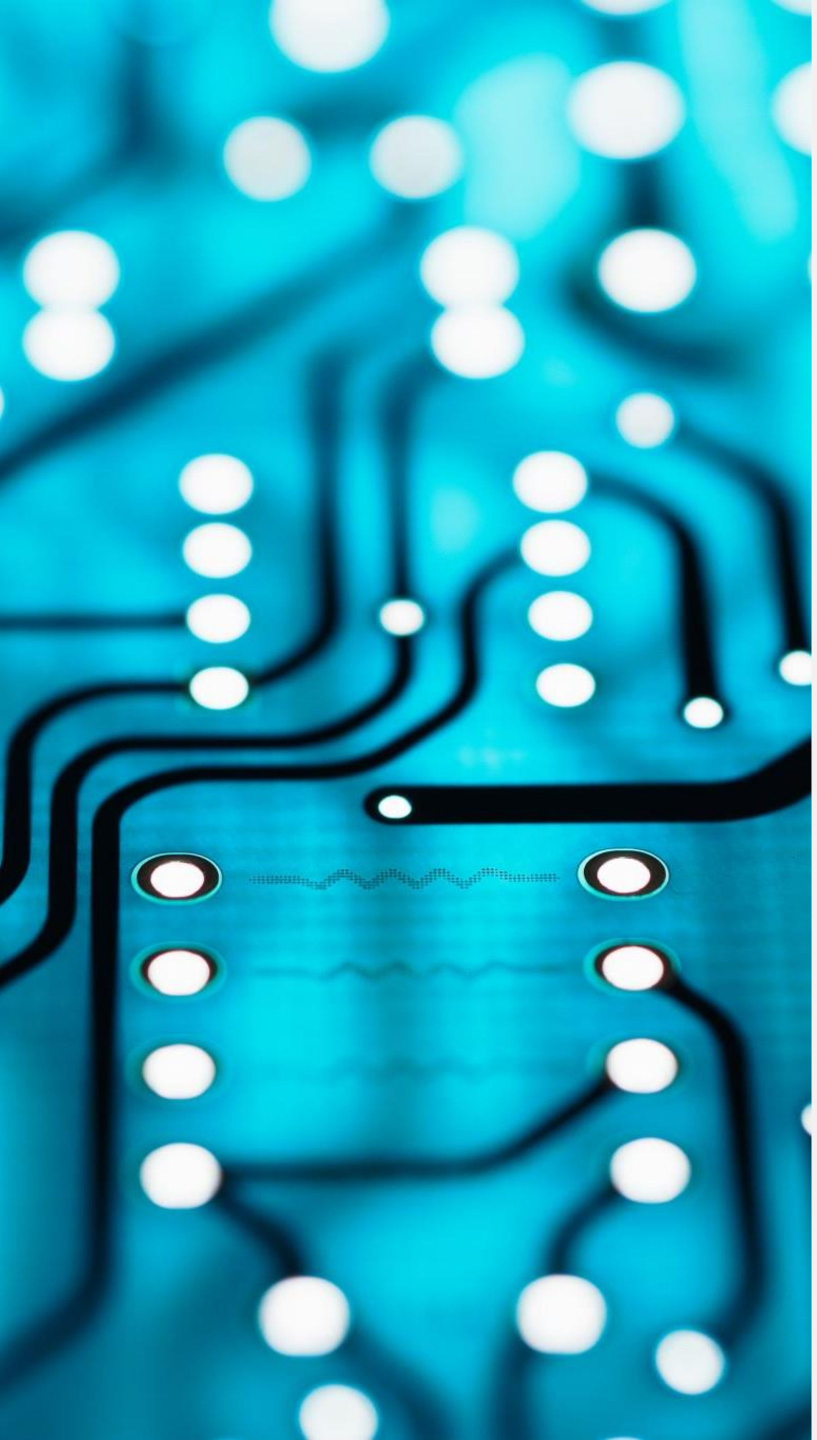


Embedded Systems (EPM)

Lecture (7) Summary

<u>1 - Relay:</u>

it’s an electrically operated switch consists of a set of input terminals for a single or multiple control signals, and a set of operating contact terminals.

The switch may have any number of contacts in multiple contact forms, such as make contacts, break contacts, or combinations thereof.

Relays used to control a circuit by an independent low-power signal.

Traditional form uses an electromagnet to close or open the contacts, but other operating principles have been invented such as:

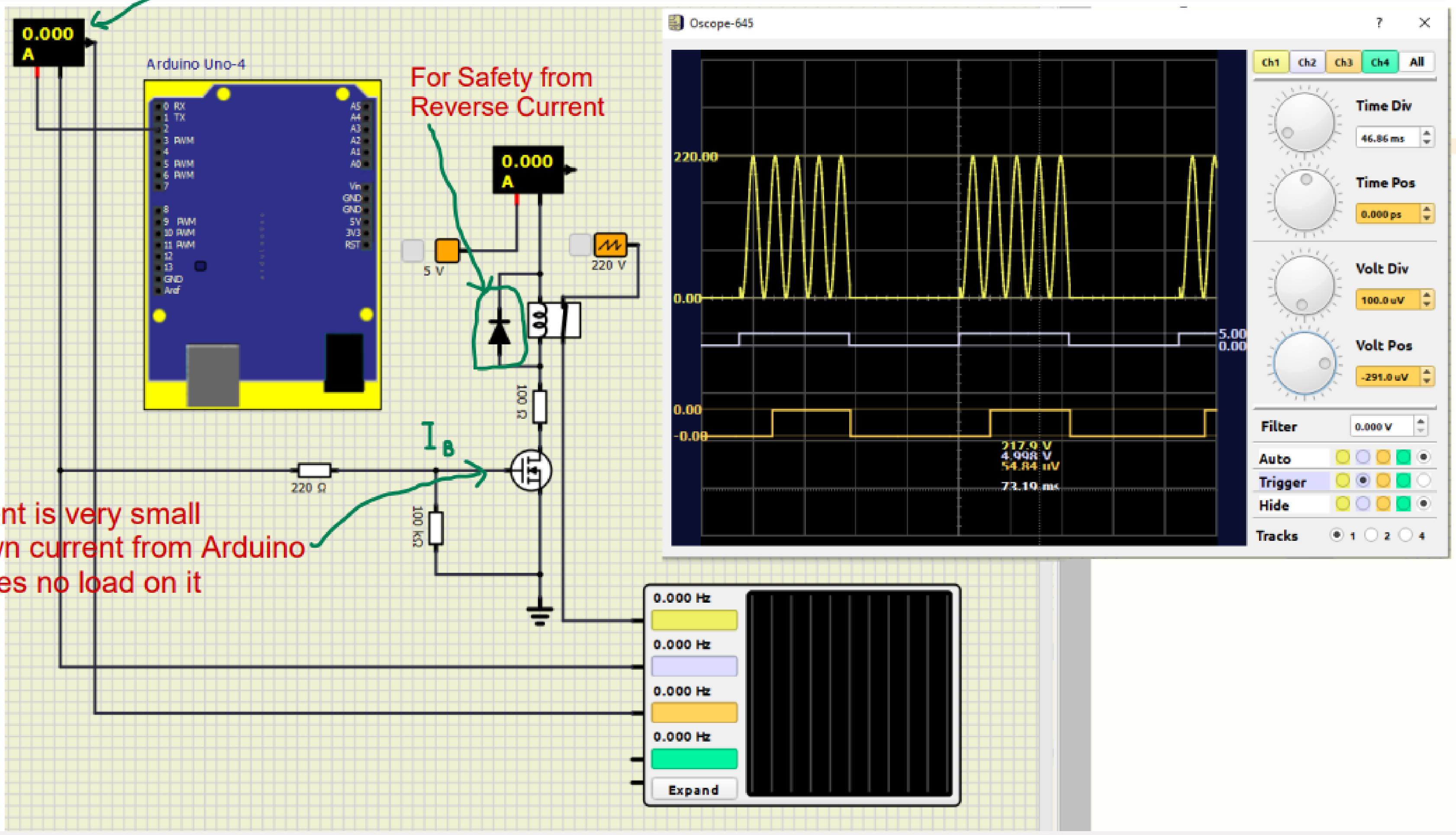
in solid-state relays which use semiconductor properties for control.

Relay(Ex.)

```
#define  OUT_PIN 2
void setup() {
  pinMode(OUT_PIN,OUTPUT);
  digitalWrite(OUT_PIN,HIGH);
}
void loop() {
  digitalWrite(OUT_PIN,LOW);
  delay(1000);
  digitalWrite(OUT_PIN,HIGH);
  delay(1000);
}
```

Ammeter

For Safety from Reverse Current



Time Div 46.86 ms
Time Pos 0.000 ps
Volt Div 100.0 uV
Volt Pos -291.0 uV
Filter 0.000 V
Auto
Trigger
Hide
Tracks [✓ 1] 2 4

Transistor Base Current is very small which makes the drawn current from Arduino is very small that makes no load on it

Note: Optocoupler is better than Diode in safety

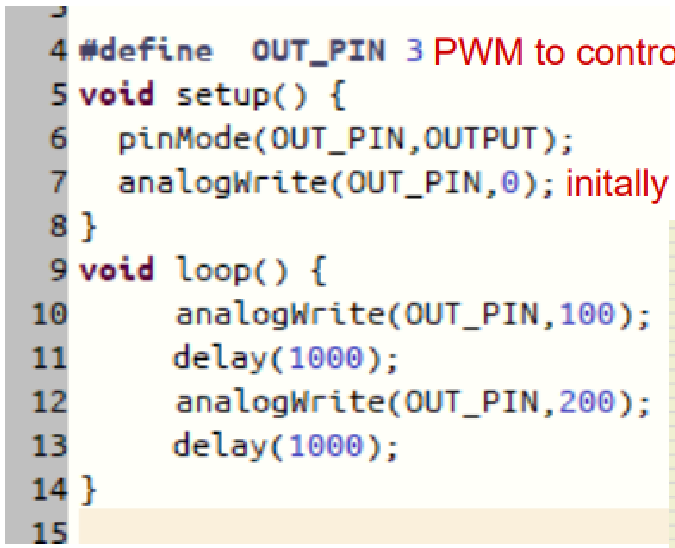
2- DC Motors:

Rotary electrical motors that converts (DC) electrical energy into mechanical energy, most types rely on the forces produced by magnetic fields.

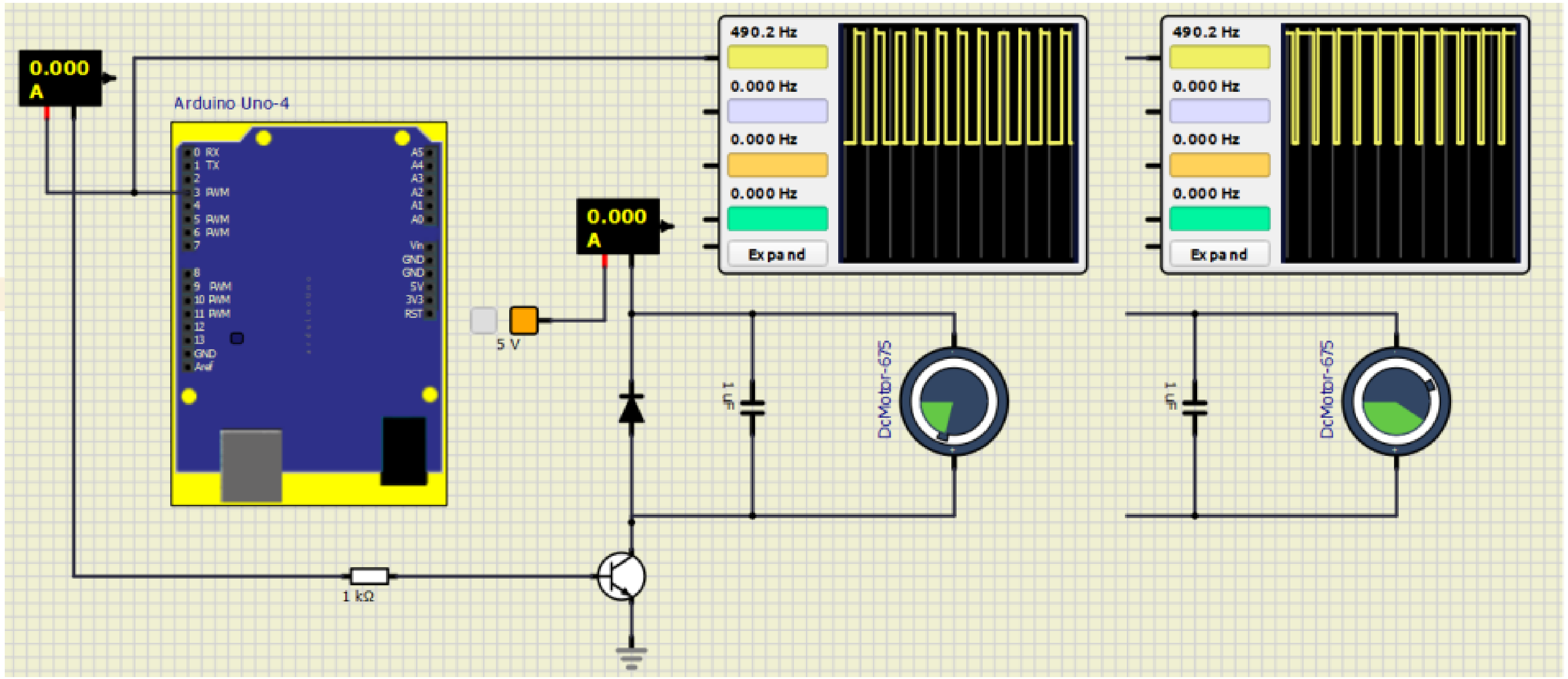
Rotate directly when DC voltage applied, As the input current increase rotation speed increase, no way to make a precise rotation angle by applying the power for a certain time

Brushed	Brushless
Simple to operate.	High efficiency.
Reliable.	Longer life.
Available in many sizes and ratings.	Less maintenance.
Easy controls.	Better suited to continuous or long-running duty cycles.
Good on lower duty cycles	Precise speed control.
Less efficiency.	Requires electronic controller.
Shorter life span.	More expensive.
Requires more maintenance.	More complex .

DC-Motor (Speed-Control) Example



```
#define  OUT_PIN 3 PWM to control the Speed of Motor
void setup() {
  pinMode(OUT_PIN,OUTPUT);
  analogWrite(OUT_PIN,0); initally Zero Value
}
void loop() {
     analogWrite(OUT_PIN,100);
     delay(1000);
     analogWrite(OUT_PIN,200);
     delay(1000);
}
```



3- H-Bridge:

it switches the polarity of a voltage applied to a load.

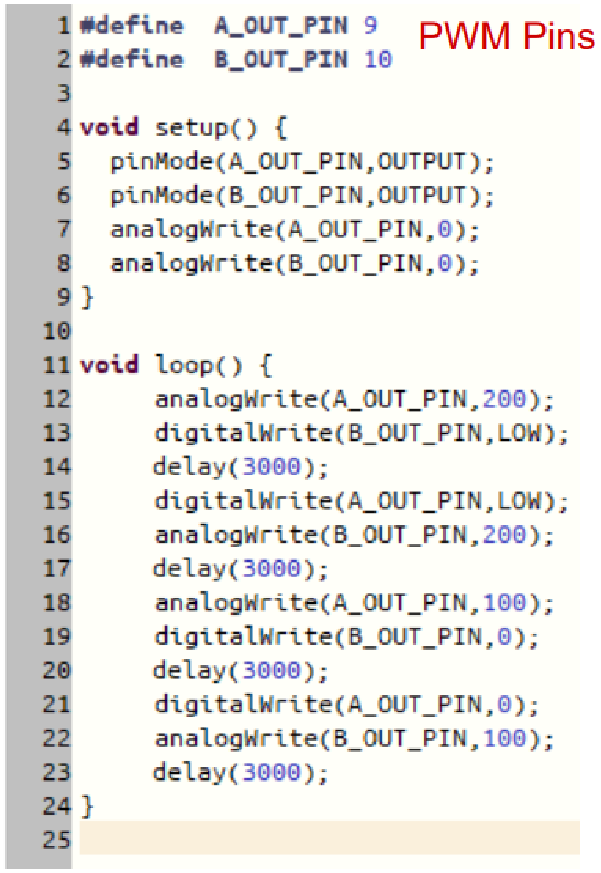
used in robotics and other applications to allow DC motors to run forwards or backwards.

its name is derived from its common schematic diagram representation, with four switching elements configured as the branches of a letter "H" and the load connected as the cross-bar.

to build an H-bridge is to use an array of relays from a relay board

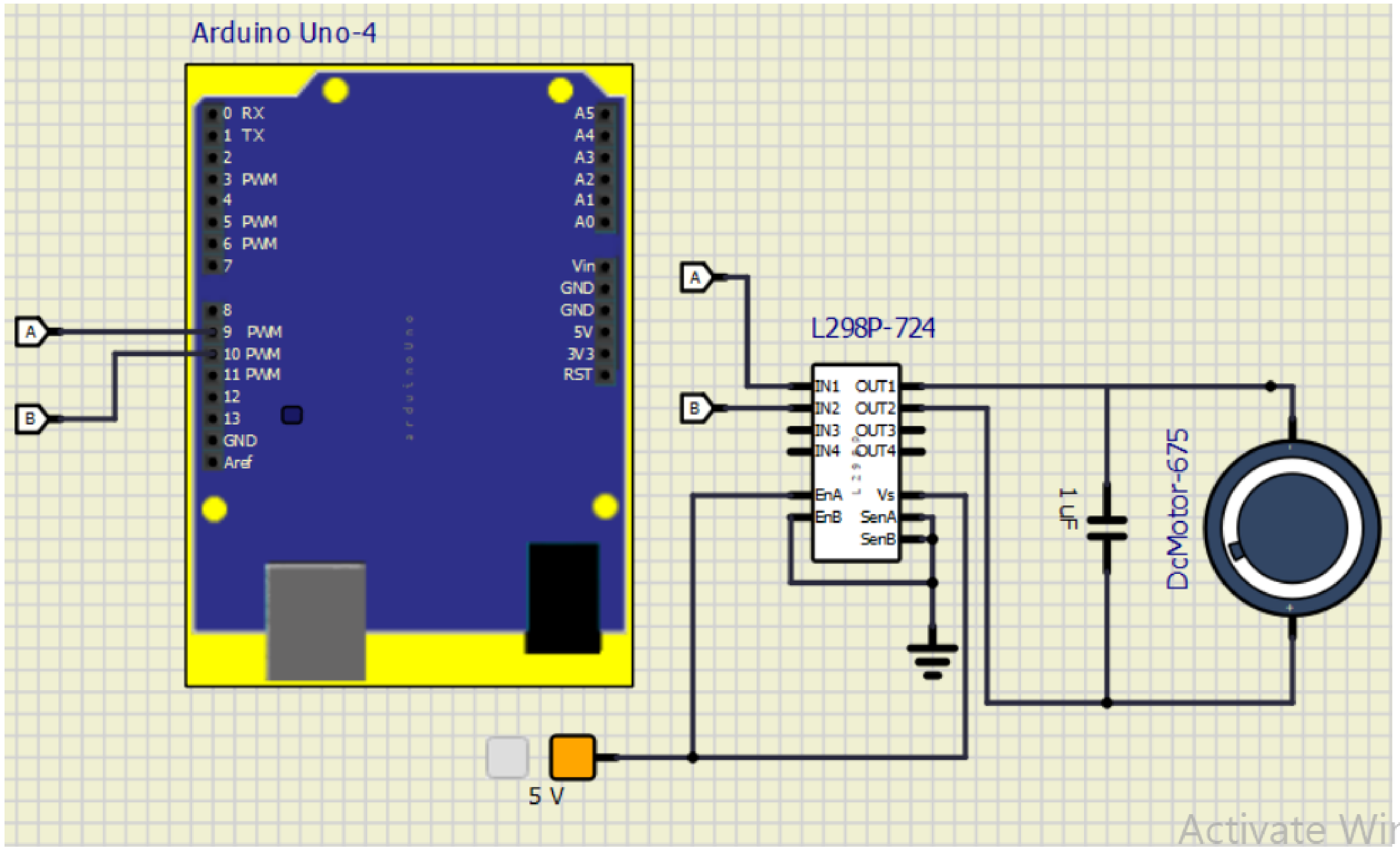
A "double pole double throw" (DPDT) relay can achieve same electrical functionality as an H-bridge (considering the usual function of the device).
a semiconductor-based H-bridge would be preferable to the relay where a smaller physical size, high speed switching, or low driving voltage (or low driving power) is needed, or where the wearing out of mechanical parts is undesirable

DC-Motor (Speed-Control/Direction) Example



```
#define  A_OUT_PIN 9
#define  B_OUT_PIN 10

void setup() {
  pinMode(A_OUT_PIN,OUTPUT);
  pinMode(B_OUT_PIN,OUTPUT);
  analogWrite(A_OUT_PIN,0);
  analogWrite(B_OUT_PIN,0);
}

void loop() {
     analogWrite(A_OUT_PIN,200);
     digitalWrite(B_OUT_PIN,LOW);
     delay(3000);
     digitalWrite(A_OUT_PIN,LOW);
     analogWrite(B_OUT_PIN,200);
     delay(3000);
     analogWrite(A_OUT_PIN,100);
     digitalWrite(B_OUT_PIN,0);
     delay(3000);
     digitalWrite(A_OUT_PIN,0);
     analogWrite(B_OUT_PIN,100);
     delay(3000);
}
```

PWM Pins



4 - Servo Motors:

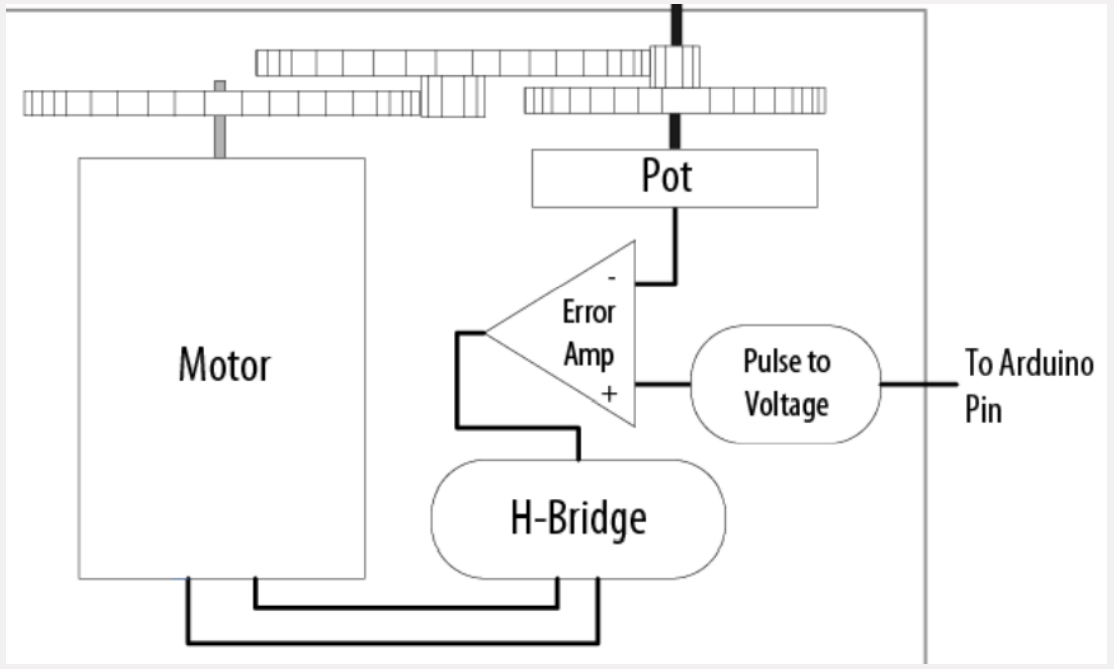
uses internal feedback system to go directly to the required angle.

controlled by a PWM input.

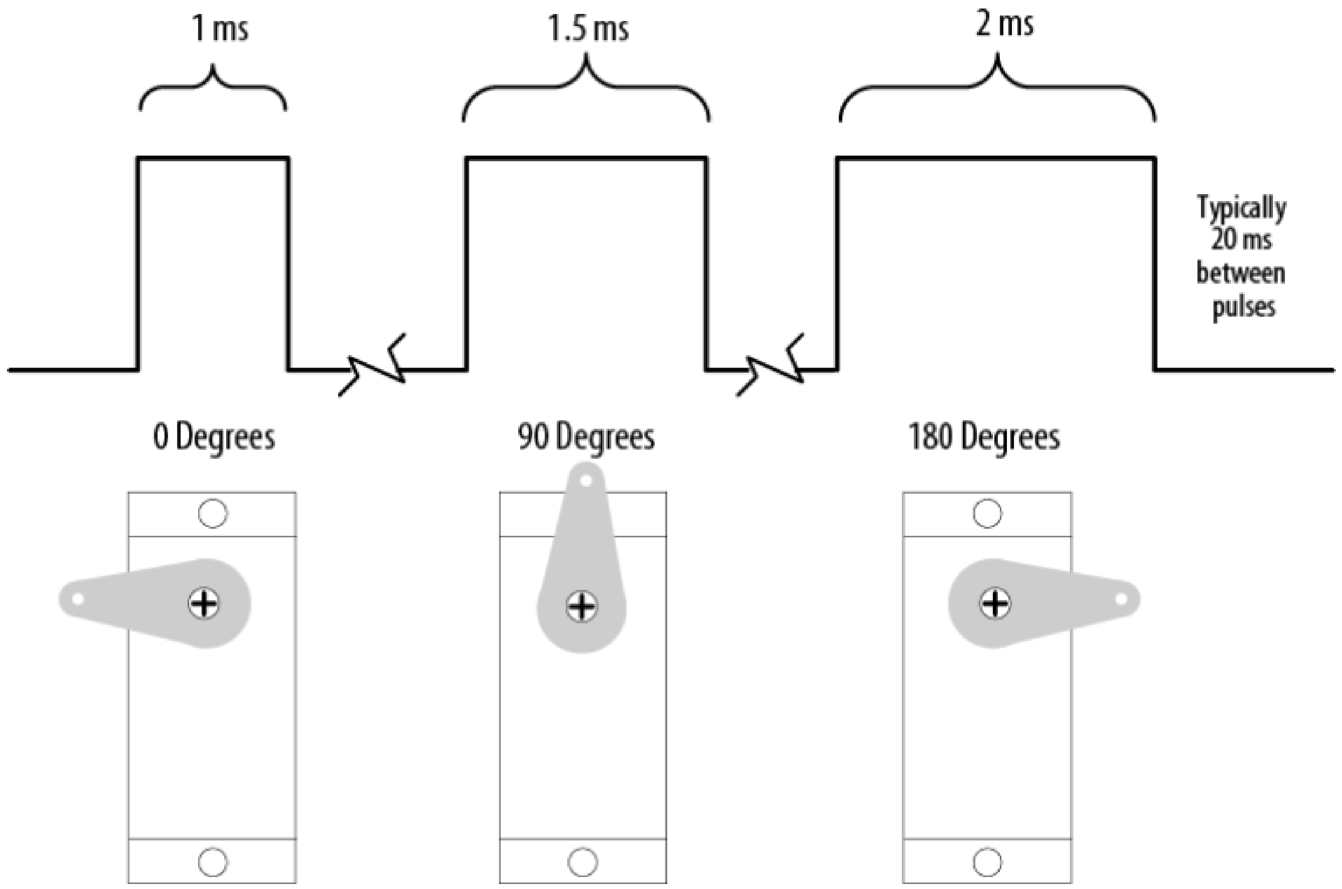
pulse width vary between minimum and maximum pulse width to produce minimum and maximum deviation angle.

Typical values:

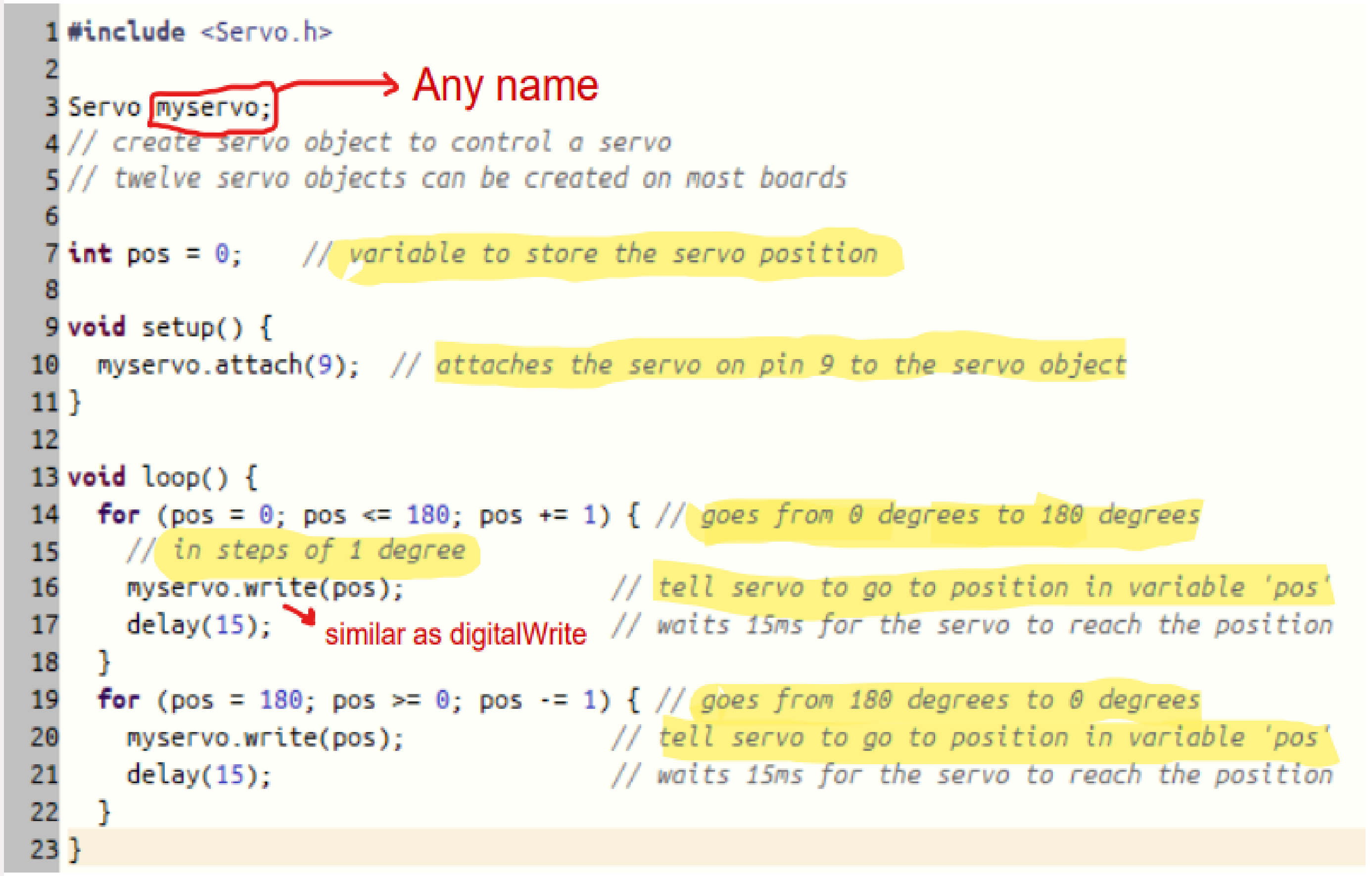
Min Pulse width: 1ms → 0 degree
Max Pulse width: 2ms → 180 degree
PWM duty cycle: 20ms



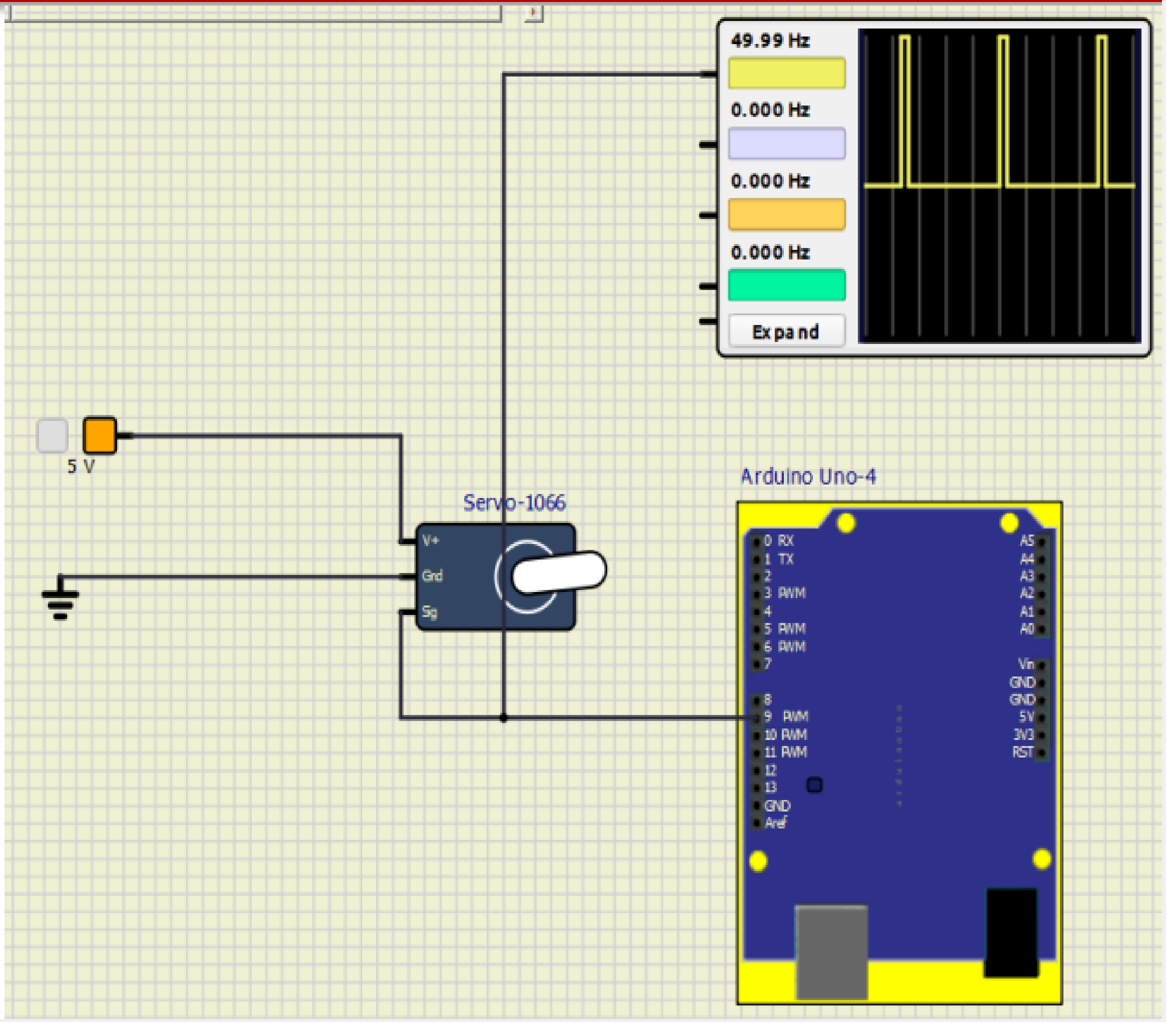
PWM Angle Control



Servo Motors (Example)



```
#include <Servo.h>

Servo myservo;
// create servo object to control a servo
// twelve servo objects can be created on most boards

int pos = 0;    // variable to store the servo position

void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
}
```



5 - Stepper Motors:

Provide precise angular steps with fixed value.

It has 2 types : Uni-Polar Model & Bi-Polar Model

Bi-polar is used to obtain the highest possible torque from the motor

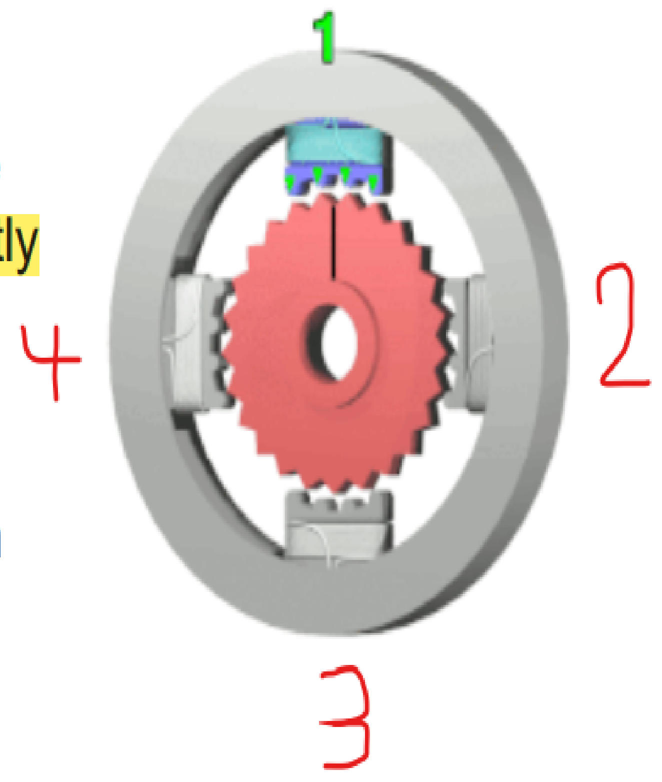
Animation of a simplified stepper motor (unipolar)

Frame 1: The top electromagnet (1) is turned on, attracting the nearest teeth of the gear-shaped iron rotor. With the teeth aligned to electromagnet 1, they will be slightly offset from right electromagnet (2).

Frame 2: The top electromagnet (1) is turned off, and the right electromagnet (2) is energized, pulling the teeth into alignment with it. This results in a rotation of 3.6° in this example.

Frame 3: The bottom electromagnet (3) is energized; another 3.6° rotation occurs.

Frame 4: The left electromagnet (4) is energized, rotating again by 3.6°. When the top electromagnet (1) is again enabled, the rotor will have rotated by one tooth position; since there are 25 teeth, it will take 100 steps to make a full rotation in this example.



Stepper Motor (unipolar/bipolar Example) (non-practical diagram)

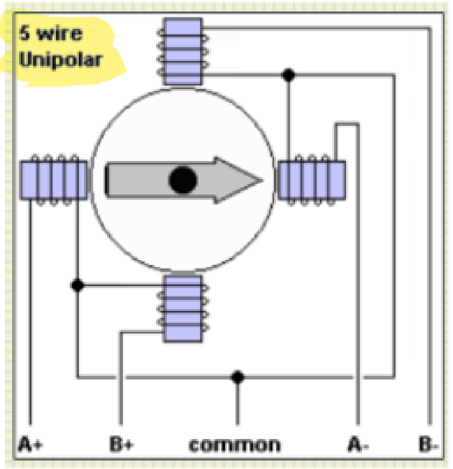
```
#include <stdio.h>
#include <stdlib.h>

#define  INP_PIN 2

#define  Bn_OUT_PIN 3
#define  An_OUT_PIN 4
#define  Bp_OUT_PIN 5
#define  Ap_OUT_PIN 6

#define  A_OUT_PIN 3
#define  B_OUT_PIN 4
#define  C_OUT_PIN 5
#define  D_OUT_PIN 6

#define  DELAY 1000

#define V1 HIGH
#define V2 LOW

//#define V1 LOW
//#define V2 HIGH

void setup() {
  pinMode(INP_PIN,INPUT_PULLUP);
  pinMode(A_OUT_PIN,OUTPUT);
  pinMode(B_OUT_PIN,OUTPUT);
  pinMode(C_OUT_PIN,OUTPUT);
  pinMode(D_OUT_PIN,OUTPUT);

  digitalWrite(A_OUT_PIN,V1);
  digitalWrite(B_OUT_PIN,V1);
  digitalWrite(C_OUT_PIN,V2);
  digitalWrite(D_OUT_PIN,V2);
}

```

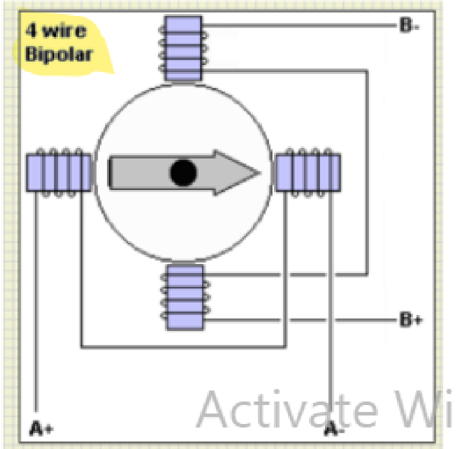
Unipolar Function

```

void unipolar_driver()
{
     digitalWrite(A_OUT_PIN,V1);
     delay(DELAY);

     digitalWrite(A_OUT_PIN,V2);
     digitalWrite(B_OUT_PIN,V1);

     delay(DELAY);

     digitalWrite(B_OUT_PIN,V2);
     digitalWrite(C_OUT_PIN,V1);

     delay(DELAY);

     digitalWrite(C_OUT_PIN,V2);
     digitalWrite(D_OUT_PIN,V1);

     delay(DELAY);

     digitalWrite(D_OUT_PIN,V2);
}

```

Bipolar Function

```

void bipolar_driver()
{

     digitalWrite(A_OUT_PIN,V1);
     digitalWrite(B_OUT_PIN,V1);

     delay(DELAY);

     digitalWrite(A_OUT_PIN,V2);
     digitalWrite(C_OUT_PIN,V1);

     delay(DELAY);

     digitalWrite(B_OUT_PIN,V2);
     digitalWrite(D_OUT_PIN,V1);

     delay(DELAY);

     digitalWrite(C_OUT_PIN,V2);
     digitalWrite(A_OUT_PIN,V1);

     delay(DELAY);
     digitalWrite(D_OUT_PIN,V2);

}

void loop() {

unipolar_driver();
//bipolar_driver();

}
```


5 wire Unipolar
A+
B+
common
A-
B-


4 wire Bipolar
B-
B+
A+
A-