บทที่ 6

ระบบควบคุมหุ่นยนต์โดยไมโครคอนโทรลเลอร์ Arduino ขับเคลื่อนด้วยเซอร์โวมอเตอร์

หัวข้อเรื่อง

1. เซอร์โวมอเตอร์
2. ฟังก์ชั่น Arduino ในการควบคุมเซอร์โวมอเตอร์
3. การเขียนโปรแกรมควบคุมหุ่นยนต์ด้วยเซอร์โวมอเตอร์

สาระสำคัญ

สมรรถนะประจำหน่วยการเรียนรู้

ออกแบบการควบคุมหุ่นยนต์โดยไมโครคอนโทรลเลอร์ Arduino ขับเคลื่อนด้วยเซอร์โวมอเตอร์

จุดประสงค์การเรียนรู้

จุดประสงค์ทั่วไป

1. เพื่อให้มีความรู้ความเข้าใจเกี่ยวกับสเต็ปปิ้งมอเตอร์
2. เพื่อให้มีความรู้ความเข้าใจในการเขียนโปรแกรมควบคุมสเต็ปปิ้งมอเตอร์
3. เพื่อให้สามารถนำความรู้ไปประยุกต์ออกแบบหุ่นยนต์อุตสาหกรรม
4. เพื่อให้ตระหนักถึงความสำคัญของโครงสร้างและส่วนประกอบของหุ่นยนต์

5.1 การควบคุม Rc Servo Motor ด้วย Arduino (thaieasyelec.com)

Arduino มีไลบรารี่สำหรับสั่งงาน RC Servo Motor มาให้ใช้งานอยู่แล้วเป็นฟังกชั่นสำเร็จรูปและใช้งานได้ง่าย สามารถสั่งงาน Rc Servo Motor ได้ทั้งแบบหมุนไป-กลับ ได้ 180 องศา และแบบต่อเนื่องที่หมุนครบรอบ ได้เรียกว่าเป็น Continuous Rotation Servo ได้ถึง 12 ตัวกับบอร์ด Arduino UNO และรองรับสูงสุดถึง 48 ตัวหากใช้บอร์ด Arduino Mega

ฟังก์ชั่นภายใน Servo Library

-attach()

- write()

- writeMicroseconds()

- read()

- attached()

-detach()

➢ Attach()

คำอธิบาย

คือฟังก์ชั่นที่ใช้ในการกำหนดขาสัญญาณที่ Servo Motor ต่อกับ Arduino และกำหนดความกว้างของพัลส์ที่ 0 องศา และ 180 องศา

รูปแบบ

Servo.attach(pin)

Servo.attach(pin,min,max)

ค่าพารามิเตอร์

Pin: คือขาสัญญาณของ Arduino ที่ใช้เชื่อมต่อกับ Servo Motor

Min :คือความกว้างของพัลส์ที่ 0 องศา ของ Servo ตัวที่ใช้ในหน่วยไมโครวินาที (us) โดยปกติแล้วหากไม่มีการตั้งค่าโปรแกรมจะกำหนดไว้ที่ 544 us

Max: คือความกว้างของพัลส์ที่ 180 องศาของServo ตัวที่ใช้ในหน่วยไมโครวินาที(us)โดยปกติหากไม่มีการตั้งค่าโปรแกรมจะกำหนดค่าไว้ที่ 2400 us

- Write()

คำอธิบาย

คือฟังก์ชั่นที่ใช้ความตำแหน่งที่ต้องการให้ Servo Motor หมุนไปยังองศาที่กำหนดสามารถกำหนดเป็นค่าองศาได้เลย คือ 0-180 องศา แต่ใน Servo Motor ที่เป็น Full Rotation คำสั่ง write จะเป็นการกำหนดความเร็วในการหมุนโดย

ค่าเท่ากับ 90 คือคำสั่งให้ Servo Motor หยุดหมุน

ค่าเท่ากับ 0 คือการหมุนด้วยความเร็วสูงสุดในทิศทางหนึ่ง

ค่าเท่ากับ 180 คือการหมุนด้วยความเร็วสูงสุดในทิศทางตรงกันข้าม

รูปแบบ

Servo.write(angle)

พารามิเตอร์

Angle: คือมุมที่ต้องการให้ RC Servo Motor แบบ 0-180 องศาหมุนไป แต่หากเป็น RC Servo Motor แบบ Full Rotation ค่า Angle คือ การกำหนดความเร็วและทิศทางในการหมุน

- writeMicroseconds()

คำอธิบาย

คือฟังก์ชั่นที่ใช้ควบคุมตำแหน่งที่ให้ Servo Motor หมุนไปยังตำแหน่งองศาที่กำหนดโดยกำหนดเป็นค่าความกว้างของพัลส์ในหน่วย us ซึ่งปกติแล้ว RC Servo Motor จะใช้ความกว้างของพัลส์อยู่ที่ 1000-2000 us

การใช้ฟังก์ชั่น writeMicrseconds สามารถกำหนดค่าได้อิสระ ตรงนี้ “ต้องระวังในการใช้งาน” หากสั่งงาน Servo Motor (แบบ 0-180 องศา) จะหมุนไปเกนจุดสิ้นสุดคือเกินฝั่ง 0 หรือ 180 องศา จะทำให้เกิดเสียงคราดังจากการหมุนไปต่อไม่ได้และมอเตอร์จะกินกระแสสูงขึ้นด้วยในเวลาเดียวกัน ซึ่งอาจทำให้ RC Servo Motor เกิดความเสียหายได้

รูปแบบ

Servo.writeMicroseconds(us)

พารามิเตอร์

uS: คือค่าความกว้างของพัลส์ที่ต้องการกำหนดในหน่วยไมโครวินาที (โดยตัวแปร int)

- read()

  คำอธิบาย

  คือฟังก์ชั่นอ่านค่าองศาที่สั่งเข้าไปด้วยฟังก์ชั่น write() เพื่อให้รู้ว่าตำแหน่งองศาสุดท้ายที่สั่งเข้าไปนั่นมีค่าเท่ากับเท่าไหร่ซึ่งค่าที่อ่านมานั้นจะมีค่าอยู่ในช่วง 0-180

  รูปแบบ

  Servo.read()

  พารามิเตอร์

  ไม่มี: จะ Return ค่า 0-180

- attached()

  คำอธิบาย

  คือฟังก์ชั่นตรวจสอบว่า Servo ที่เราต้องการใช้กำลังต่ออยู่ขาสัญญาณของ Arduino หรือไม่

  รูปแบบ

  Servo.attached()

  พารามิเตอร์

  ไม่มี :จะ Return ค่า true ออกมา หาก Servo Motor เชื่อมต่ออยู่กับ Arduino แต่ถ้าหาก Return ออกมาเป็นค่าอื่นถือว่าไม่มีการเชื่อมต่อ

- detach()

  คำอธิบาย

  คือฟังก์ชั่นคืนสถานะของขาที่เรากำหนดให้เป็นขาควบคุม Servo Motor ด้วยคำสั่ง attached() ให้กลับคืนสู่การใช้งานปกติ

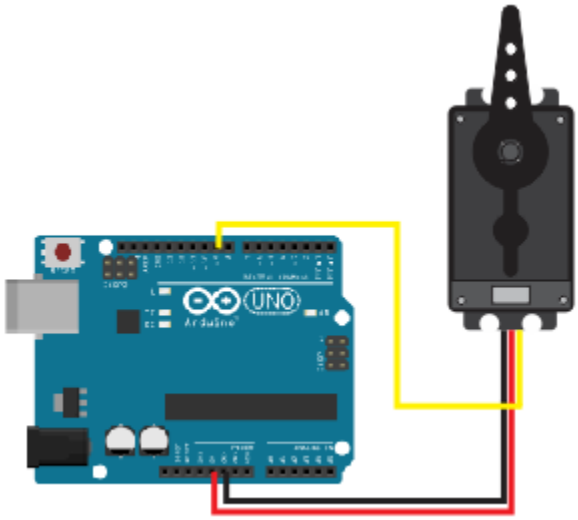
  รูปแบบ

  Servo.detach()

  พารามิเตอร์

  ไม่มี

ตัวอย่างการเชื่อมต่อ RC Servo Motor เข้ากับบอร์ด Arduino



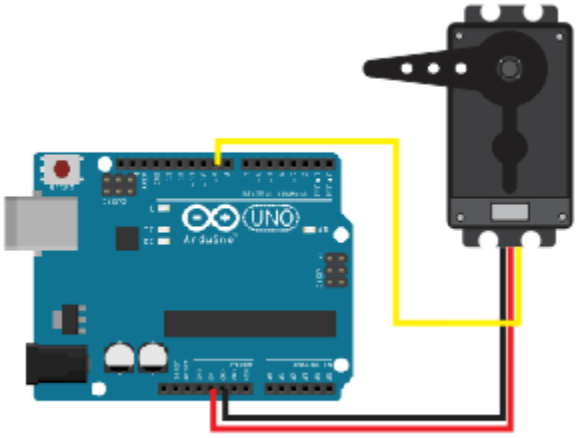
รูปที่ การเชื่อมต่อ Servo Motor กับบอร์ด Arduino

ที่มา: https://thaieasyelec.com/article-wiki/review-product-article/

Code

```
#include<Servo.h>
Servo myservo;
void setup() {
myservo.attach(9);
 }
void loop() {
myservo.write(0);
delay(1000);
myservo.write(90);
delay(1000);
myservo.write(180);
delay(1000);
}
```

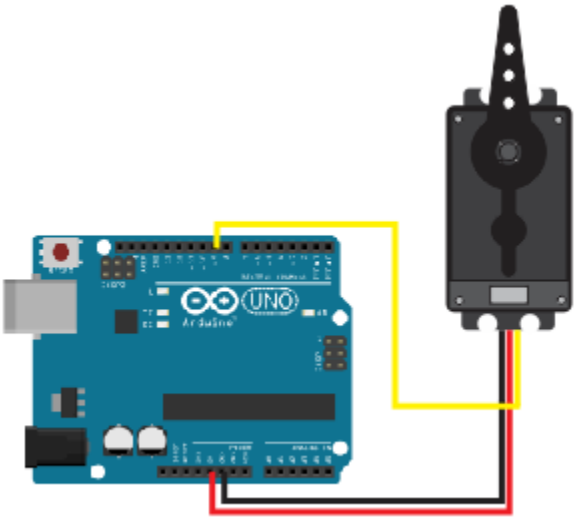
ผลการทำงานของ Code



myservo.write(0);

delay(1000);

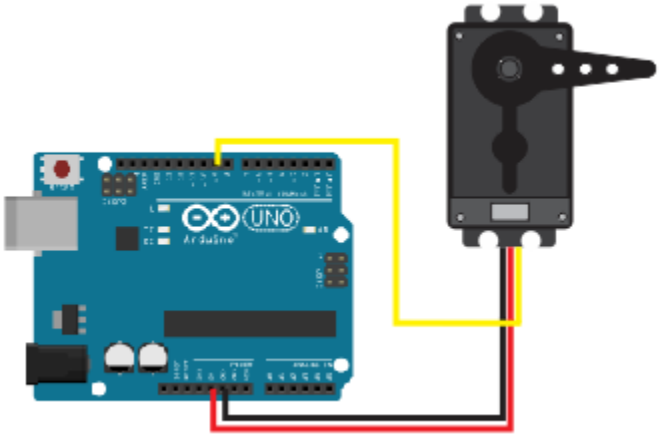
Servo Motor จะหมุนไปที่ตำแหน่ง 0 องศา และหยุดเป็นเวลา 1 วินาที



myservo.write(90);

delay(1000);

Servo Motor จะหมุนไปที่ตำแหน่ง 90 องศา และหยุดเป็นเวลา 1 วินาที



Myservo.write(90);

Delay(1000);

Servo Motor จะหมุนไปที่ตำแหน่ง 180 องศาและหยุดเป็นเวลา 1 วินาที จากนั้นจะหมุนกลับไปที่ตำแหน่ง 0 องศา และวนรอบไปเช่นนี้เรื่อยๆ

Code ตัวอย่างการควบคุมตำแหน่ง RC Servo Motor แบบ Sweep

```
#include<Servo.h>
Servo myservo;
int pos=0;
void setup() {
myservo.attach(9);
}
void loop() {
for(pos=0;pos<180;pos+=1)
 {
myservo.write(pos);
delay(15);
}
for(pos=180;pos>=1;pos-=1)
```

```
    {
  myservo.write(pos);
  delay(15);
    }
   }
```

ผลของการทำงานของ Code

```
    for(pos=0;pos<180;pos+=1){
    myservo.write(pos);
    delay(15);
    }
```

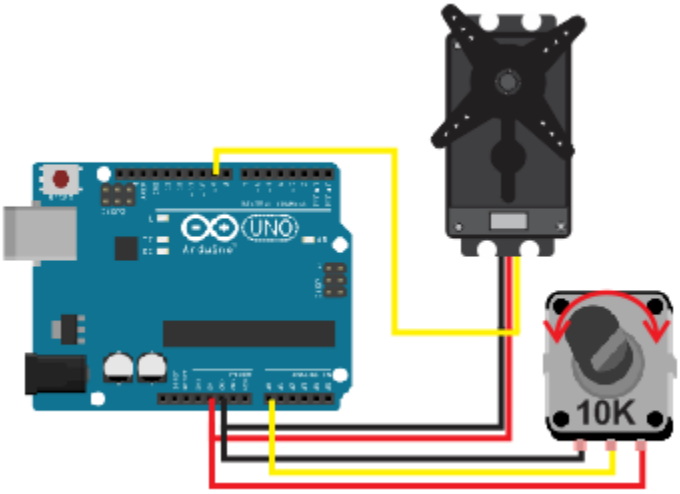
ลูป for กำหนดให้ค่า pos มีค่าเท่ากับ 0 และทุกๆการทำงานคำสั่งภายใน for loop ค่า pos จะเพิ่มขึ้น 1 ค่า จนถึงค่า 180 ก็จะหลุดออกจาก loop

ภายใน loop for คำสั่ง myservo.write(pos); ก็คือการกำหนดให้ Servo Motor หมุนไปยังตำแหน่ง มุมตามค่าในตัวแปร pos และหน่วงเวลา 15 ms ด้วยคำสั่ง delay(15); ดังนั้น Servo Motor จะค่อยๆหมุนอย่างช้าๆจากตำแหน่ง 0 องศาไปที่ 180 องศา

```
    for(pos=180;pos>180;pos-=1){
    myservo.write(pos);
   delay(15);
   }
```

ใน loop[for ที่สองนี้จะทำงานเช่นเดียวกับใน loop for แรกเพียงแต่เปลี่ยนค่าเริ่มต้นจาก 180 เป็น 0 และลดค่าลง 1 ค่า ทุกๆการทำงาน 1 รอบ ส่งผลให้ Servo Motor จะหมุนจากตำแหน่งมุม 180 องศา ไปยังมุม 0 องศาอย่างช้าๆ

Code ตัวอย่างการควบคุมตำแหน่ง RC Servo Motor โดยใช้ Potentiometer




```
#include<Servo.h>
 Servo myservo;
 int potpin=0;
 int val;
 void setup() {
 myservo.attach(9);
 }
void loop() {
val=analogRead(potpin);
val=map(val,0,1023,0,179);
myservo.write(val);
delay(15);
}
```

ผลการทำงานของ Code

Val=analogRead(potpin);

อ่านค่า Analog จาก Potentiometer ที่ต่ออยู่ที่ขา A0 เก็บไว้ในตัวแปร val

Val=map(val,0,1023,0,179);

เนื่องจาก ADC ภายใน Arduino เป็น ADC ขนาด 10-bit จึงอ่านค่า Analog ได้ตั้งแต่ 0-1023 แต่ RC Servo Motor สามารถหมุนได้เพียงแค่ 1-180 องศา จึงต้องใช้ Funtion map เพื่อทำการสเกลค่าลงจาก 0-1023 เป็น 0-179 แล้วนำไปเก็บไว้ในตัวแปร val

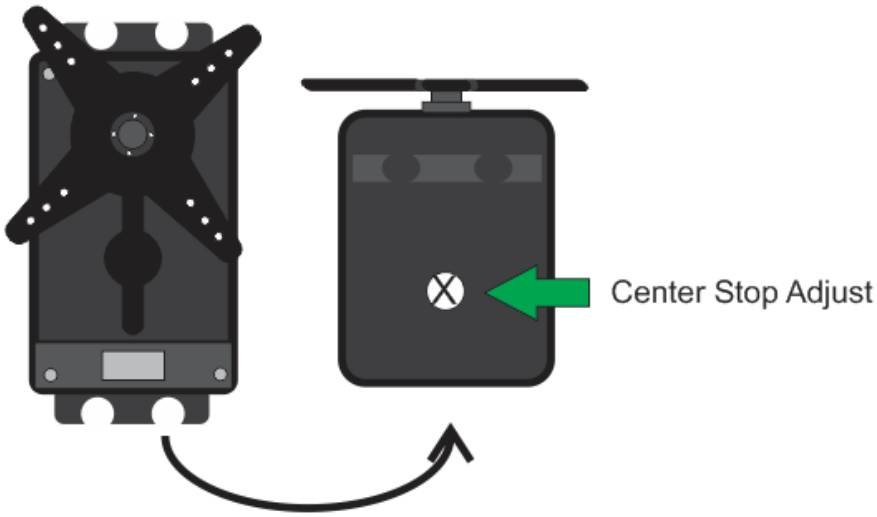
myservo.write(val);

เมื่อสเกลค่าจาก 0-1023 ลงเหลือ 0-179 แล้วก็นำมาสั่งให้ Servo Motor หมุนไปยังตำแหน่งในค่าตัวแปร val

Delay(15);

หน่วงเวลา 15 ms

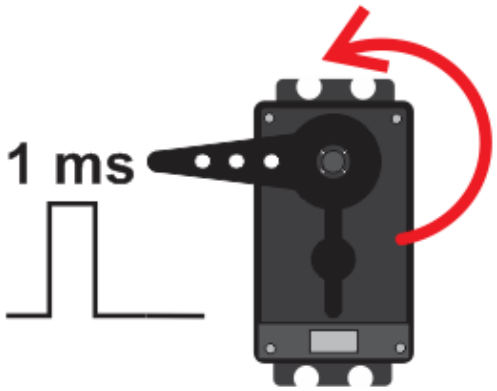
ผลของการทำงานทำให้สามารถปรับตำแหน่งองศาของ Servo Motor ได้โดยการหมุนปรับค่า Potentiometer



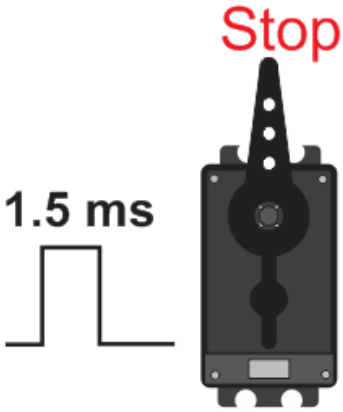
Continuous Rotation Servo คือ RC Servo Motor แบบที่สามารถหมุนได้ 360 องศา ส่วนประกอบภายนอกนั้นจะมีหน้าตาคล้ายกับ RC Servo Motor แบบที่หมุนได้ 180 องศา เพียงแต่จะมี Potentiometer เพื่อใช้สำหรับปรับตำแหน่ง Center Stop Adjust ของตัว Servo

ลักษณะการใช้งาน RC Servo Motor ชนิดนี้จะแตกต่างจากการใช้งาน RC Servo Motor แบบ 180 องศา ตรงที่ Servo ชนิดนี้จะมีความกว้างของสัญญาณพัลส์ในการกำหนดความเร็วและทิศทางในการหมุน ไม่ได้ใช้เพื่อกำหนดมุมจึงไม่สามารถกำหนดให้Motor หมุนไปยังตำแหน่งมุมต่างๆตามความต้องการได้ สัญญาณความกว้างของพัลส์ที่ใช้ควบคุมจะอยู่ในช่วง 1000-2000 us แต่จะมีความแตกต่างในความหมายของแต่ละความกว้างของพัลส์ดังนี้

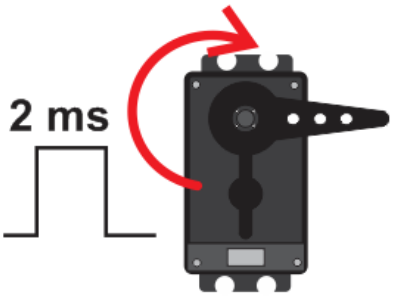
ความกว้าง1000 us หมายถึงการหมุนไปทางซ้ายด้วยความเร็วสูงสุดที่ Servo Motor จะหมุนได้



ความกว้าง 1500 us หมายถึงการสั่งให้ Servo Motor หยุดหมุน



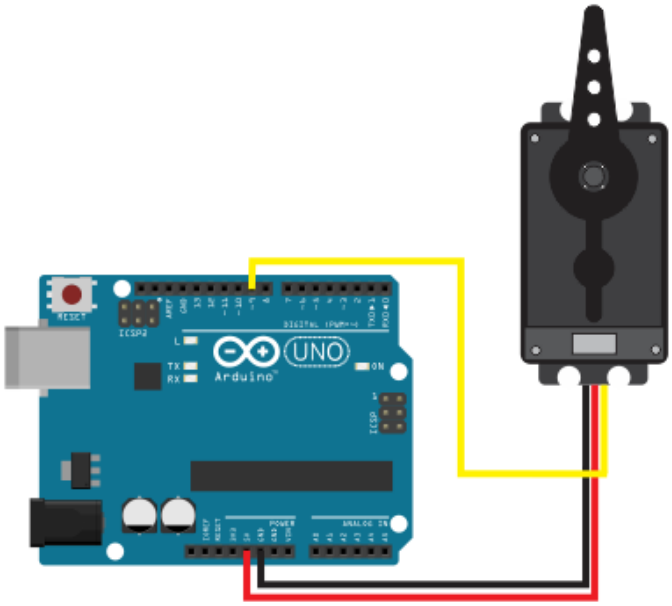
ความกว้าง 2000 us หมายถึงการหมุนไปทางขวาด้วยความเร็วสูงสุดที่ Servo Motor จะหมุนได้



การ Calibrate Center Stop

ในการใช้งาน Continuous Rotation Servo เมื่อใช้งานไประยะหนึ่งจุด Center Stop อาจมีการคลาดเคลื่อนได้ ซึ่งแม้ว่าจะสั่งให้สัญญาณพัลส์มีความกว้างเท่ากับ 1500 us ไป Continuous rotation servo ก็จะไม่หยุดหมุน ดังนั้นจะต้องปรับตั้งค่า Center Stop ดังนี้

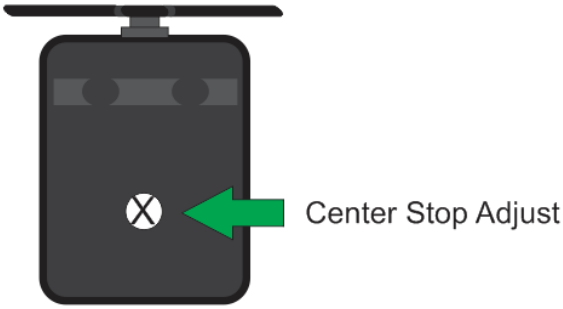
1) ต่อ Continuous rotation servo เข้ากับ Arduino



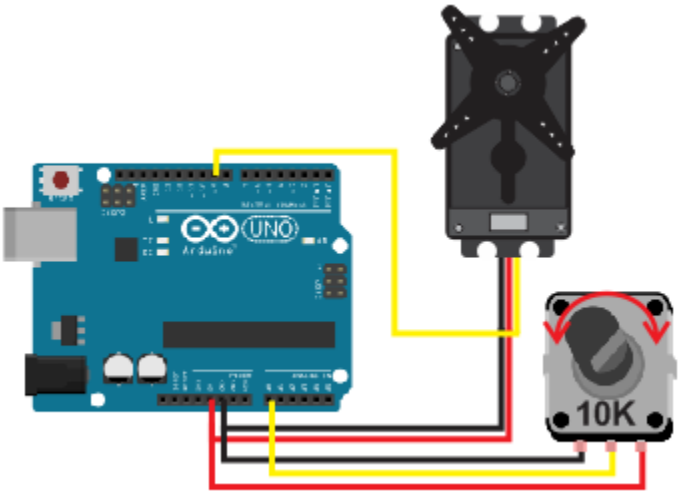
2) เขียนโปรแกรมจ่ายความกว้างพัลซ์ 15000us ให้กับ Servo Motor

```
#include<Servo.h>
 Servo myServo;
Void  setup(){
 myServo.attach(9);
 myServo.writeMicroseconds(1500);
 }
```

3) เมื่อรันโปรแกรมจ่ายความกว้างพัลซ์ 1500us แล้ว Servo Motor ไม่หยุดหมุน ให้ใช้ไขควงขนาดเล็กหมุนปรับ Center Stop Adjust จน Servo Motor หยุดหมุน



การควบคุม Continuous Rotation Servo โดยใช้ Potentinmeter ปรับความเร็วและทิศทางการหมุน



Code ตัวอย่างการควบคุม Continuous Rotation Servo โดยใช้ Potentiometer

```
#include<Servo.h>
 Servo myservo;
 int potpin=0;
 int val;
 void setup() {
 myservo.attach(9);
  }
  void loop() {
 val=analogRead(potpin);
 val=map(val,0,1023,0,179);
```

```
myservo.write(val);
delay(15);
 }
```

ผลการทำงานของ Code

Val=analogRead(potpin);

อ่านค่า Analog จาก Potentiometer ที่ต่ออยู่ที่ขา A0 เก็บไว้ในตัวแปร val

Val=map(val,0,1023,0,179);

เนื่องจาก ADC ภายใน Arduino เป็น ADC ขนาด 10-bit จึงอ่านค่า Analog ได้ตั้งแต่ 0-1023 แต่ RC Servo Motor สามารถหมุนได้เพียงแค่ 1-180 องศา จึงต้องใช้ Funtion map เพื่อทำการสเกลค่าลงจาก 0-1023 เป็น 0-179 แล้วนำไปเก็บไว้ในตัวแปร val

myservo.write(val);

เมื่อสเกลค่าจาก 0-1023 ลงเหลือ 0-179 แล้วก็นำมาสั่งให้ Servo Motor หมุนไปยังตำแหน่งในค่าตัวแปร val

Delay(15); หน่วงเวลา 15 ms