01076001
วิศวกรรมคอมพิวเตอร์เบื้องต้น
Introduction to Computer Engineering

Arduino #7

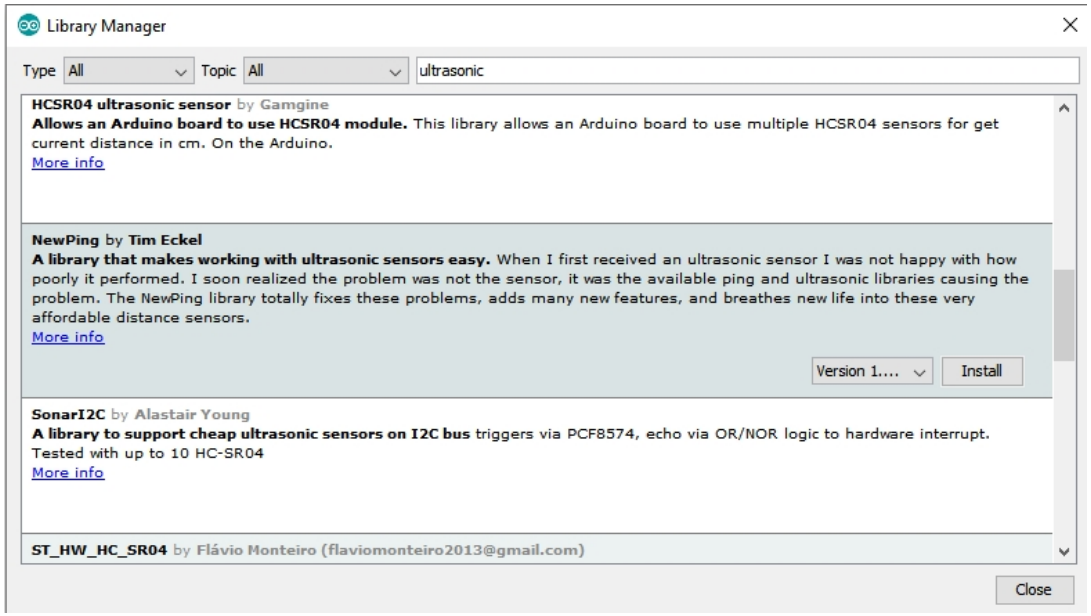
Servo, Ultrasonic

Ultrasonic

- เป็นเซ็นเซอร์วัดระยะทาง หรือ ตรวจจับสิ่งกีดขวาง โดยใช้หลักการคำนวณระยะทางจากเวลาที่ใช้ในการสะท้อนกลับของคลื่นเสียงในอากาศ
- โมดูล Ultrasonic มีมากมายหลายแบบมาก
- สำหรับ US-015 การใช้งานจะส่งสัญญาณออกไป โดยสั่งที่ขา Trig
- จากนั้นสัญญาณสะท้อนกลับจะรับมาทางขา Echo

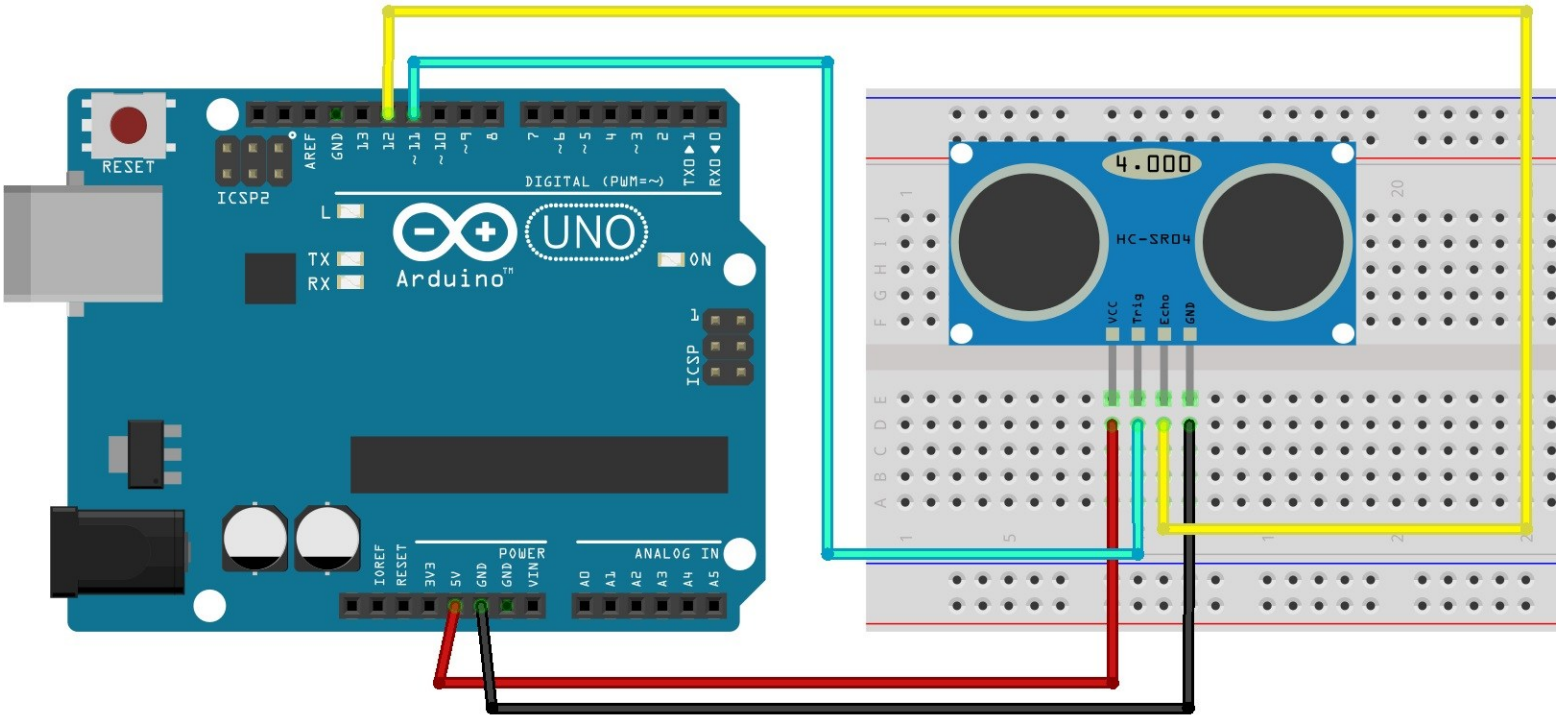
Ultrasonic

- เราสามารถนำเวลามาคำนวณเพื่อหาระยะทางได้
- แต่เพื่อความสะดวก มีผู้ทำ Library ไว้ให้ใช้งานแล้ว
- ไปที่ Library Manger แล้วพิมพ์คำว่า ultrasonic จะเห็นว่ามี library หลายตัว แต่เราจะเลือกตัวที่ชื่อ NewPing ตามรูป (ให้กดติดตั้ง)



Ultrasonic

- ต่อเซ็นเซอร์ตามรูป (Trig -> 11, Echo -> 12)



Ultrasonic

- ไปที่ File -> Examples -> NewPing -> NewPingExample

NewPingExample §

```
// ---------------------------------------------------------------------------
// Example NewPing library sketch that does a ping about 20 times per second.
// ---------------------------------------------------------------------------

#include <NewPing.h>

#define TRIGGER_PIN  11  // Arduino pin tied to trigger pin on the ultrasonic sensor.
#define ECHO_PIN     12  // Arduino pin tied to echo pin on the ultrasonic sensor.
#define MAX_DISTANCE 200 // Maximum distance we want to ping for (in centimeters). Maximum sensor distance is rated at 400-500cm.

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); // NewPing setup of pins and maximum distance.

void setup() {
  Serial.begin(115200); // Open serial monitor at 115200 baud to see ping results.
}

void loop() {
  delay(50);                     // Wait 50ms between pings (about 20 pings/sec). 29ms should be the shortest delay between pings.
  Serial.print("Ping: ");
  Serial.print(sonar.ping_cm()); // Send ping, get distance in cm and print result (0 = outside set distance range)
  Serial.println("cm");
}
```

Exercise

- ให้ทดสอบการทำงานของ Ultrasonic
- เมื่อมีวัตถุเข้าใกล้กว่า 20 cm ให้ไฟกระพริบ

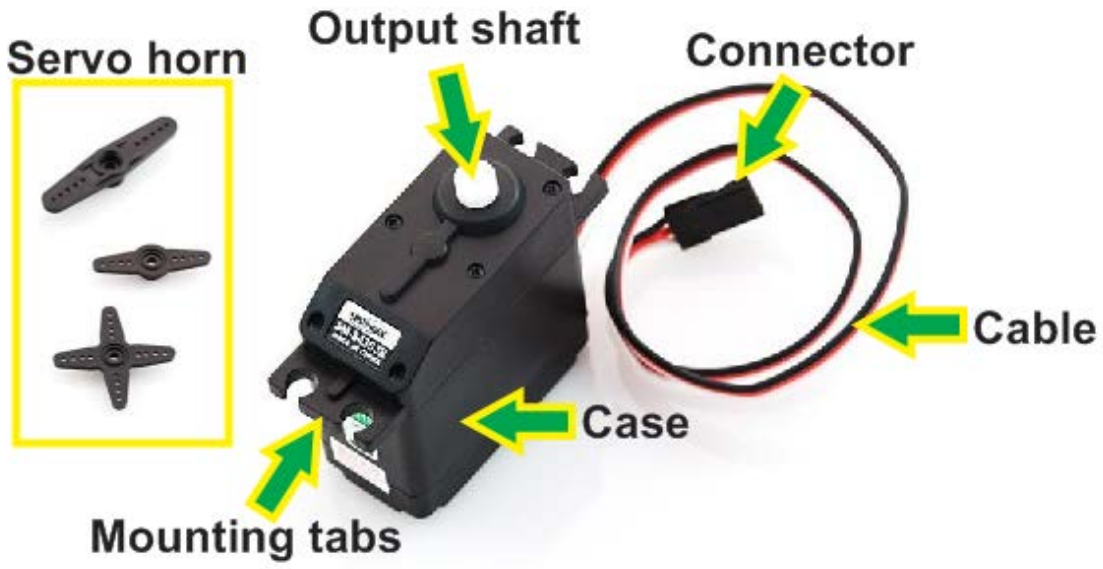
- Option : ถ้าเอา Ultrasonic 2 ตัวหันใส่กัน จะยังบอกระยะตรงหรือไม่

Servo Motor

- เซอร์โวมอเตอร์ เป็นมอเตอร์ชนิดหนึ่งที่มีการใช้กันมาก ลักษณะพิเศษของเซอร์โวมอเตอร์ คือ การสามารถบังคับหรือสั่งงาน ให้มอเตอร์หมุนไปยังองศาหรือตำแหน่งที่เราต้องการได้
- ที่เซอร์โวมอเตอร์ สามารถทำงานเช่นนี้ได้ เนื่องจากเซอร์โวมอเตอร์ ได้มีการเพิ่มวงจรที่ทำหน้าที่ควบคุมการป้อนกลับ (Feedback Control)
- เซอร์โวมอเตอร์ มีการนำไปใช้งานกันอย่างกว้างขวาง เช่น รถ/เรือบังคับวิทยุ เฮลิคอปเตอร์บังคับวิทยุ ในส่วนของการควบคุมการเลี้ยว หรือใช้ในระบบแขนกล

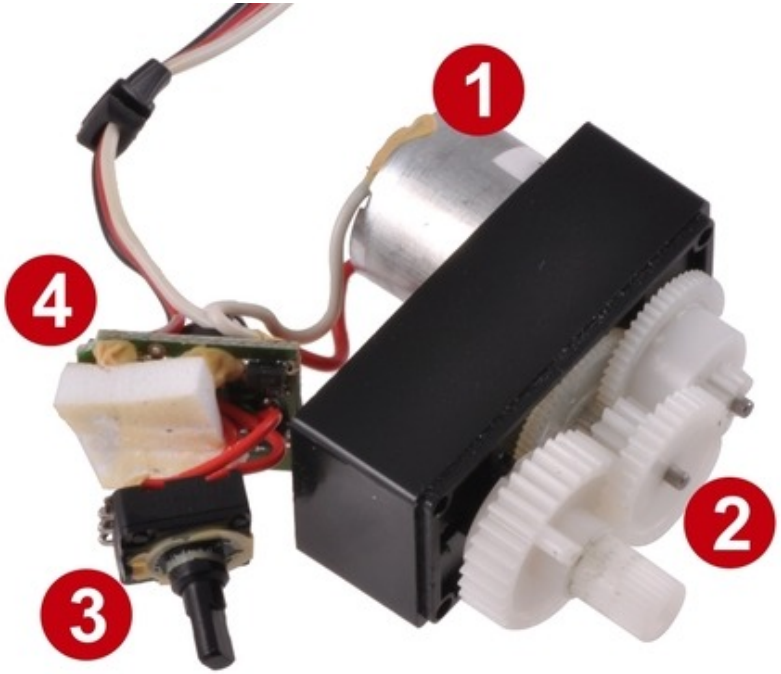
Servo Motor

- ส่วนประกอบของเซอร์โวมอเตอร์
  - Case ตัวถัง หรือ กรอบของตัว Servo Motor
  - Mounting Tab ส่วนจับยึดตัว Servo กับชิ้นงาน
  - Output Shaft เพลาส่งกำลัง
  - Servo Horns ส่วนเชื่อมต่อกับ Output shaft เพื่อสร้างกลไก



Servo Motor

- ส่วนประกอบภายในเซอร์โวมอเตอร์
  - Motor เป็นส่วนของตัวมอเตอร์
  - Gear Train หรือ Gearbox เป็นชุดเกียร์ทดแรง
  - Position Sensor เป็นเซ็นเซอร์ตรวจจับตำแหน่งเพื่อหาค่าองศาในการหมุน
  - Electronic Control System เป็นส่วนที่ควบคุมและประมวลผล

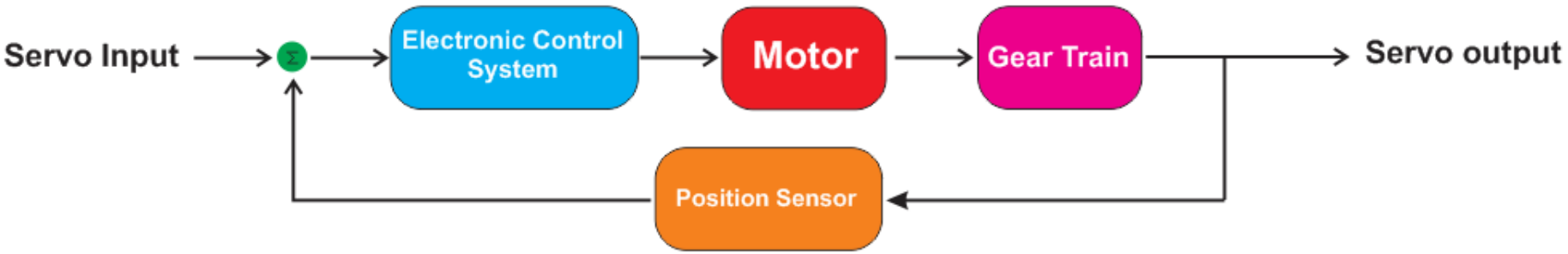


Servo Motor

- Cable สายเชื่อมต่อเพื่อ จ่ายไฟฟ้า และ ควบคุมเซอร์โวมอเตอร์ จะประกอบด้วยสายไฟ 3 เส้น และ ในเซอร์โวมอเตอร์ จะมีสีของสายแตกต่างกันไปดังนี้ (ในรุ่นอื่นอาจมีสีแตกต่างออกไป)
  - สายสีแดง คือ ไฟเลี้ยง (4.8-6V)
  - สายสีดำ หรือ น้ำตาล คือ กราวด์
  - สายสีเหลือง (ส้ม ขาว หรือฟ้า) คือ สายส่งสัญญาณพัลซ์ควบคุม (3-5V)

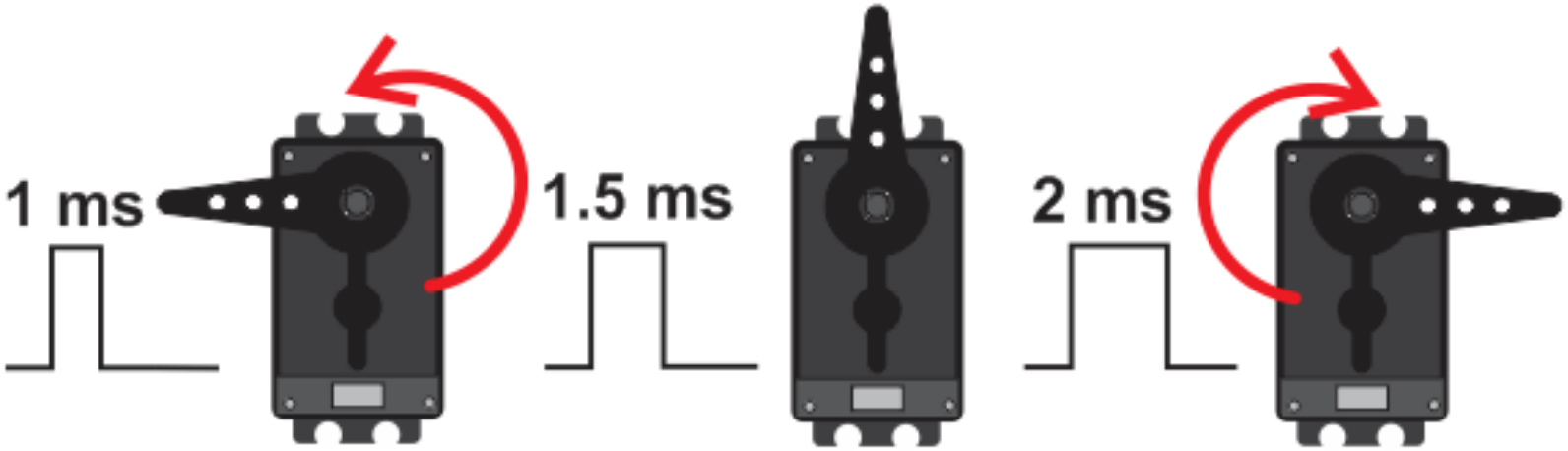
Servo Motor

- หลักการทำงานของเซอร์โวมอเตอร์
  - เมื่อจ่ายสัญญาณพัลซ์เข้ามายังเซอร์โวมอเตอร์ ส่วนวงจรควบคุม (Electronic Control System) ภายในเซอร์โวจะอ่านค่าความกว้างของสัญญาณพัลซ์ที่เข้ามาและประมวลผลเพื่อแปลงค่าเป็นตำแหน่งองศาที่ต้องการให้มอเตอร์หมุนไปยังตำแหน่งนั้น แล้วส่งคำสั่งไปควบคุมมอเตอร์ ให้หมุนไปยังตำแหน่งที่ต้องการ โดยมี Position Sensor คอยวัดค่ามุมที่ Motor กำลังหมุน เป็น Feedback กลับมาให้วงจรควบคุมเปรียบเทียบกับค่าอินพุตเพื่อควบคุมให้ได้ตำแหน่งที่ต้องการอย่างถูกต้องแม่นยำ



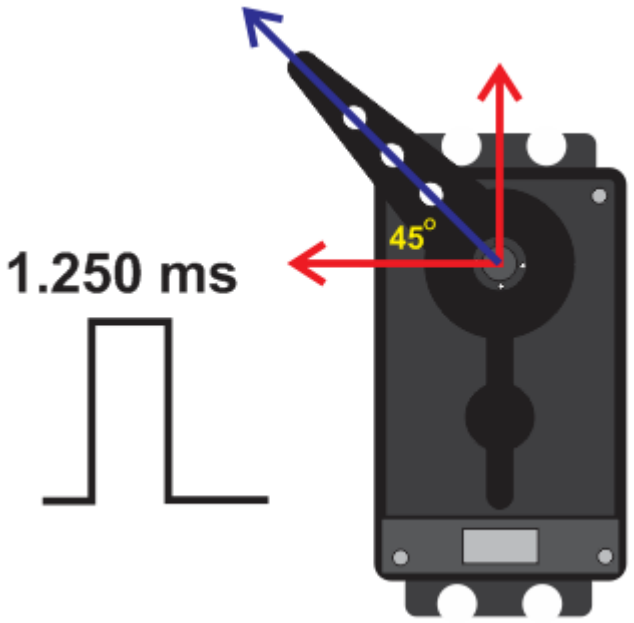
Servo Motor

- มุมหรือองศาจะขึ้นอยู่กับความกว้างของสัญญาณพัลซ์ เช่น หากกำหนดความกว้างของสัญญาณพัลซ์ไว้ที่ 1 ms เซอร์โวมอเตอร์จะหมุนไปทางด้ายซ้ายจนสุด ในทางกลับกันหากกำหนดความกว้างของสัญญาณพัลซ์ไว้ที่ 2 ms ตัว Servo Motor จะหมุนไปยังตำแหน่งขวาสุด แต่หากกำหนดความกว้างของสัญญาณพัลซ์ไว้ที่ 1.5 ms ตัว Servo Motor ก็จะหมุนมาอยู่ที่ตำแหน่งตรงกลางพอดี



Servo Motor

- ดังนั้นสามารถกำหนดองศาการหมุนของเซอร์โวมอเตอร์ได้ โดยการเทียบค่า เช่น เซอร์โวมอเตอร์สามารถหมุนได้ 180 องศา โดยที่ 0 องศาใช้ความกว้างพัลซ์ 1000 us ที่ 180 องศาความกว้างพัลซ์ 2000 us เพราะฉะนั้นค่าที่เปลี่ยนไป 1 องศาจะใช้ความกว้างพัลซ์ต่างกัน (2000-1000)/180 เท่ากับ 5.55 us
- จากการหาค่าความกว้างพัลซ์ข้างต้น หากจะให้เซอร์โวมอเตอร์หมุนไปที่มุม 45 องศา ค่าพัลซ์ที่ต้องการ คือ 5.55 x 45 เท่ากับ 249.75 us แต่ที่มุม 0 องศาเริ่มที่ความกว้างพัลซ์ 1ms หรือ 1000 us เพราะฉะนั้นความกว้างพัลซ์ที่ใช้กำหนดให้เซอร์โวมอเตอร์หมุนไปที่ 45 องศา คือ 1000 + 249.75 หรือประมาณ 1250 us



Servo Motor

- การควบคุมด้วย Arduino
  - เราสามารถสั่งงาน เซอร์โวมอเตอร์ โดยใช้ PWM ได้
  - แต่เพื่อให้ง่ายกว่านั้น Arduino ได้ Build In ไลบรารีของเซอร์โวมาให้ด้วย โดยมีฟังก์ชันดังนี้
    - attach()
    - write()
    - writeMicroseconds()
    - read()
    - attached()
    - detach()

Servo Motor

- ฟังก์ชัน attach()

Description

เป็นฟังก์ชันที่ใช้ในการกำหนดขาสัญญาณเซอร์โวมอเตอร์ต่อกับ Arduino และกำหนดความกว้างของพัลซ์ที่ 0 องศาและ 180 องศา

Syntax

```
Servo.attach(pin)
Servo.attach(pin,min,max)
```

Parameters

Pin: คือ ขาสัญญาณของ Arduino ที่ใช้เชื่อมต่อกับ Servo Motor

Min: คือ ความกว้างของพัลซ์ที่ 0 องศาของ Servo ตัวที่ใช้ในหน่วยไมโครวินาที (us) โดยปกติแล้วหากไม่มีการตั้งค่าโปรแกรมจะกำหนดค่าไว้ที่ 544 us

Max: คือ ความกว้างของพัลซ์ที่ 180 องศาของ Servo ตัวที่ใช้ในหน่วยไมโครวินาที (us) โดยปกติแล้วหากไม่มีการตั้งค่าโปรแกรมจะกำหนดค่าไว้ที่ 2400 us

Servo Motor

- ฟังก์ชัน Write()

  Description

  ฟังก์ชันที่ใช้ควบคุมตำแหน่งที่ต้องการให้เซอร์โวมอเตอร์หมุนไปยังองศาที่กำหนด สามารถกำหนดเป็นค่าองศาได้เลย คือ 0-180 องศา แต่ในเซอร์โวมอเตอร์ที่เป็น Full Rotation (คือ หมุนได้รอบ) คำสั่ง write จะเป็นการกำหนดความเร็วในการหมุน โดย

  ค่าเท่ากับ 90 คือคำสั่งให้ Servo Motor หยุดหมุน

  ค่าเท่ากับ 0 คือการหมุนด้วยความเร็วสูงสุดในทิศทางหนึ่ง

  ค่าเท่ากับ 180 คือการหมุนด้วยความเร็วสูงสุดในทิศทางตรงกันข้าม

- Syntax

  servo.write(angle)

- Parameters

  Angle: คือมุมที่ต้องการให้เซอร์โวมอเตอร์แบบ 0-180 องศาหมุนไป แต่หากเป็น เซอร์โวมอเตอร์แบบ Full Rotation ค่า Angle คือ การกำหนดความเร็วและทิศทางในการหมุน

Servo Motor

- ฟังก์ชัน writeMicroseconds()

  Description

  ฟังก์ชันที่ใช้ควบคุมตำแหน่งของเซอร์โวมอเตอร์ โดยกำหนดเป็นค่าความกว้างของพัลซ์ในหน่วย us ซึ่งปกติแล้วเซอร์โวมอเตอร์ จะใช้ความกว้างของพัลซ์อยู่ที่ 1000-2000 us แต่เซอร์โวมอเตอร์บางรุ่นหรือบางยี่ห้อไม่ได้ใช้ช่วงความกว้างของพัลซ์ตามที่ได้กล่าวเอาไว้ เช่น อาจใช้ช่วง 700-2300 us ก็สามารถใช้ฟังก์ชัน writeMicroseconds เพื่อกำหนดความกว้างพัลซ์ได้เอง

  การใช้ฟังก์ชั่น writeMicroseconds สามารถกำหนดค่าได้อิสระ ดังนั้นต้องระวังในการใช้งาน หากสั่งงานเซอร์โวมอเตอร์ (แบบ 0 - 180 องศา) จนหมุนไปเกินจุดสิ้นสุดคือเกินทั้งฝั่ง 0 หรือ 180 องศา จะทำให้เกิดเสียงครางดังจากการหมุนไปต่อไม่ได้และมอเตอร์จะกินกระแสสูงขึ้นด้วย ซึ่งอาจทำให้เซอร์โวมอเตอร์เกิดความเสียหายได้

- Syntax

  servo.writeMicroseconds(uS)

- Parameters

  uS: คือค่าความกว้างของพัลซ์ที่ต้องการกำหนดในหน่วยไมโครวินาที (ตัวแปร int)

Servo Motor

- ฟังก์ชัน read()

  Description

  คือฟังก์ชันอ่านค่าองศาที่สั่งด้วยฟังก์ชั่น write() เพื่อให้รู้ว่าตำแหน่งองศาสุดท้ายที่สั่งเข้าไปนั้นมีค่าเท่าไหร่ซึ่งค่าที่อ่านออกมานั้นจะมีค่าอยู่ในช่วง 0 - 180

- Syntax

  servo.read()

- Parameters

  ไม่มี: จะ Return ค่า 0-180

Servo Motor

- ฟังก์ชัน attached()

  Description

  ฟังก์ชันตรวจสอบว่า เซอร์โวมอเตอร์ ที่ต้องการใช้ต่ออยู่กับขาสัญญาณของ Arduino หรือไม่

- Syntax

  servo.attached()

- Parameters

  ไม่มี: จะ Return ค่า True ออกมา หากเซอร์โวมอเตอร์ เชื่อมต่ออยู่กับ Arduino แต่ถ้าหาก Return ออกมาเป็นค่าอื่นถือว่าไม่เชื่อมต่อ

- ฟังก์ชัน detach()

  Description

  ฟังก์ชันคืนสถานะของขาที่เรากำหนดให้เป็นขาควบคุมเซอร์โวมอเตอร์ ด้วยคำสั่ง attached() ให้กลับคือสู่การใช่งานปกติ (ไม่งั้นจะเอาไปทำอย่างอื่นไม่ได้)

- Syntax

  servo.detach()

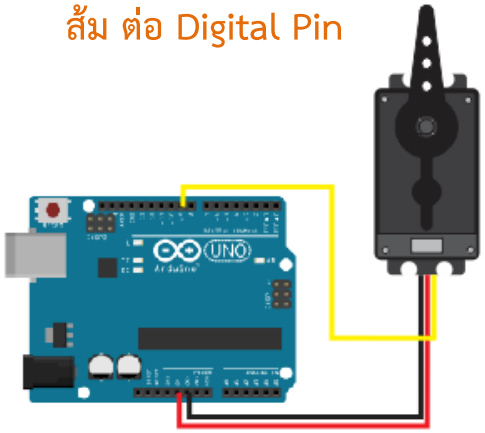
- Parameters

  ไม่มี

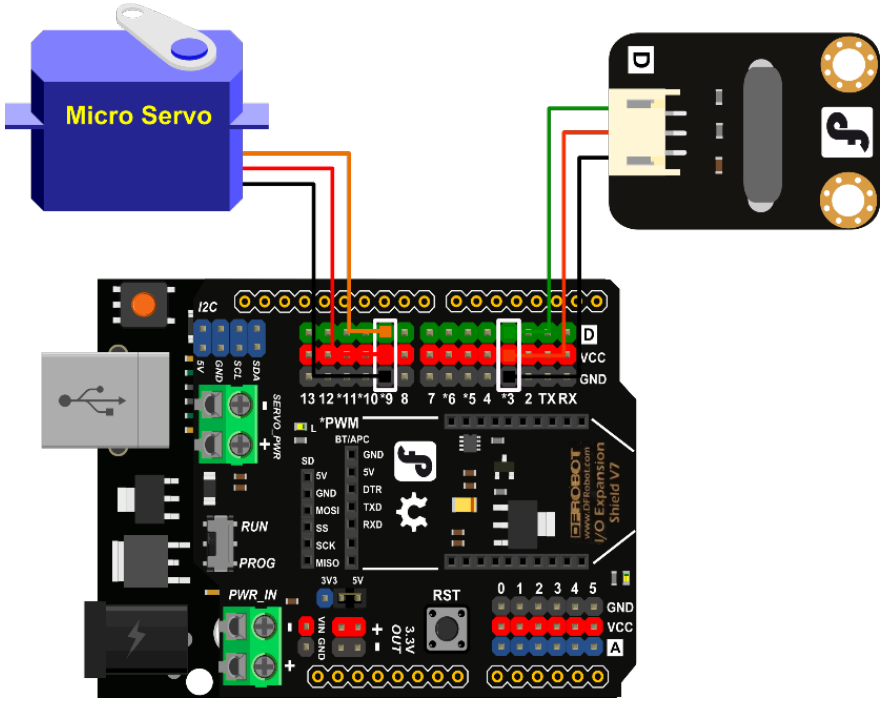
Servo Motor

- การเชื่อมต่อ เซอร์โวมอเตอร์ เข้ากับบอร์ด Arduino
- กรณีต่อกับ Arduino โดยตรง

แดง ต่อ 5V
น้ำตาล GND
ส้ม ต่อ Digital Pin



- กรณีต่อผ่าน I/O Expansion Shield ให้ต่อกับขาฝั่ง Digital Pin (ตามรูป) ต่อขา 9



Servo Motor : Exercise

- โหลดโปรแกรมจาก Examples -> Servo -> Sweep แล้วลองทำงาน

```
#include <Servo.h>

Servo myservo; // เรียกใช้งานคำสั่งจาก Library

int pos = 0; // ประกาศตัวแปรสำหรับเก็บค่าองศา Servo กำหนดให้เริ่มที่ 0 องศา

void setup() {
myservo.attach(9); // ต่อ Servo กับ Digital Pin 9
}

void loop() {
for (pos = 0; pos <= 180; pos += 1) // ค่อยๆเพิ่มองศาการหมุน จาก 0 ถึง 180 เพิ่มขึ้นที่ละ 1 องศาในเวลา 0.015 วินาที
{
myservo.write(pos); // คำสั่งองศาการหมุน
delay(5);
}
for (pos = 180; pos >= 0; pos -= 1) // ถ้าองศาถึง 180 แล้ว ให้ลดทีละ 1 องศาจนถึง 0 แล้ววนไปเรื่อยๆ
{
myservo.write(pos);
delay(5);
}
}
```

Servo Motor

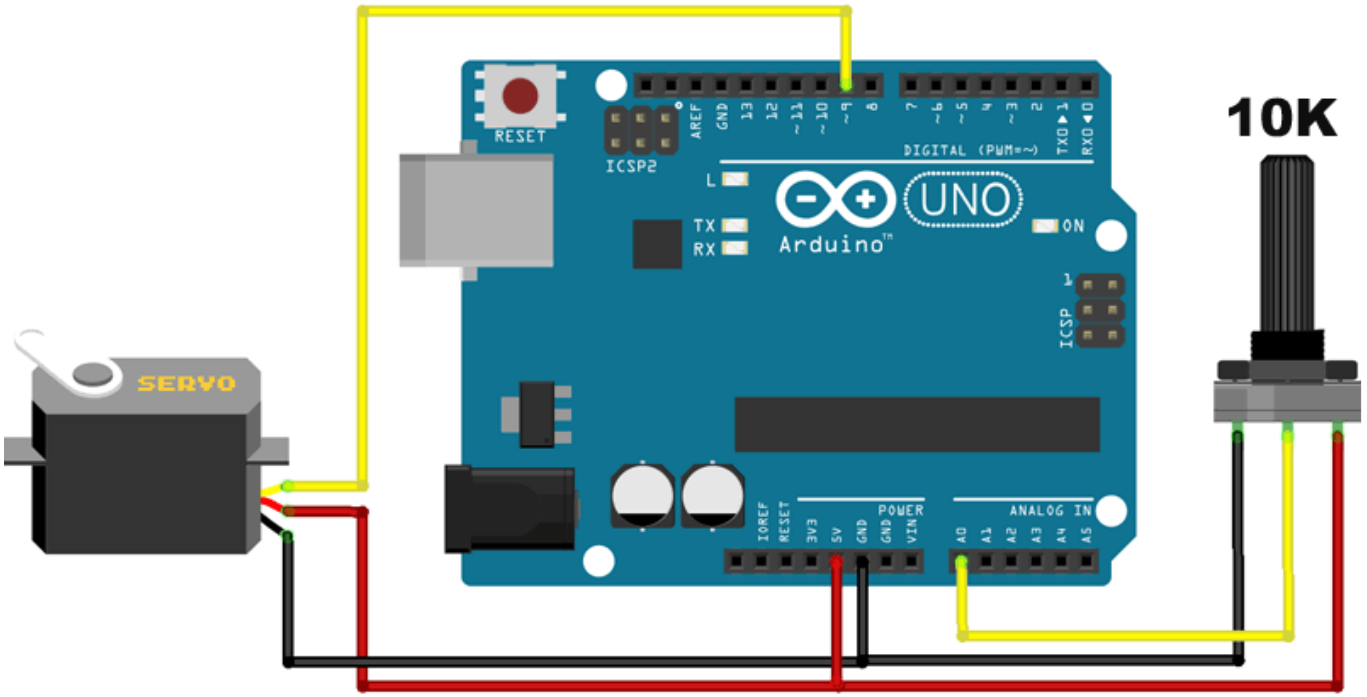
- การควบคุม เซอร์โวมอเตอร์ จาก Input
  - เป็นการบังคับให้เซอร์โวมอเตอร์หมุนตามค่าที่ Input เข้าไป เช่น ใช้ potentiometer รับค่าแล้วสั่งให้เซอร์โวมอเตอร์หมุนตาม
  - เนื่องจากค่าของ potentiometer กับค่าที่เซอร์โวมอเตอร์ต้องใช้ มีค่าคนละช่วงกัน เพื่อให้ง่ายจึงได้มีการจัดทำคำสั่ง map โดยมีรูปแบบการใช้ดังนี้

```
map(ตัวแปร, ต่ำสุดเดิม, สูงสุดเดิม, ต่ำสุดใหม่, สูงสุดใหม่)
```

  - กรณีนี้ ต้านทานปรับค่าได้ เราสามารถเขียน map ได้ดังนี้ map(val, 0, 1023, 0, 180) หมายถึง ค่า Input ที่รับจาก potentiometer จะมีค่าได้ตั้งแต่ 0 - 1023 จะปรับให้ต่ำสุดเป็น 0 และสูงสุดไม่เกิน 180 ตามองศาสูงสุดที่หมุนได้นั่นเอง

Servo Motor

- ให้ต่อวงจรตามนี้ (สามารถใช้ I/O Expansion Shield ได้)



Servo Motor : Exercise

- โหลดโปรแกรมจาก Examples -> Servo -> Knob แล้วลองทำงาน

```
#include <Servo.h>

Servo myservo; // เรียกใช้งานคำสั่งจาก Library

int potpin = 0; // ต่อตัวต้านทานปรับค่าได้กับ analog pin 0
int val; // ประกาศค่าที่อ่านได้จากตัวต้านทาน

void setup() {
myservo.attach(9); // ต่อ Servo กับ Digital Pin 9
}

void loop() {
val = analogRead(potpin); // อ่านค่าตัวแปร
val = map(val, 0, 1023, 0, 180); // ทำการ map จากค่าที่อ่านได้ ไปเป็นองศาของ Servo
myservo.write(val); // กำหนดองศาให้ Servo
delay(1);
}
```

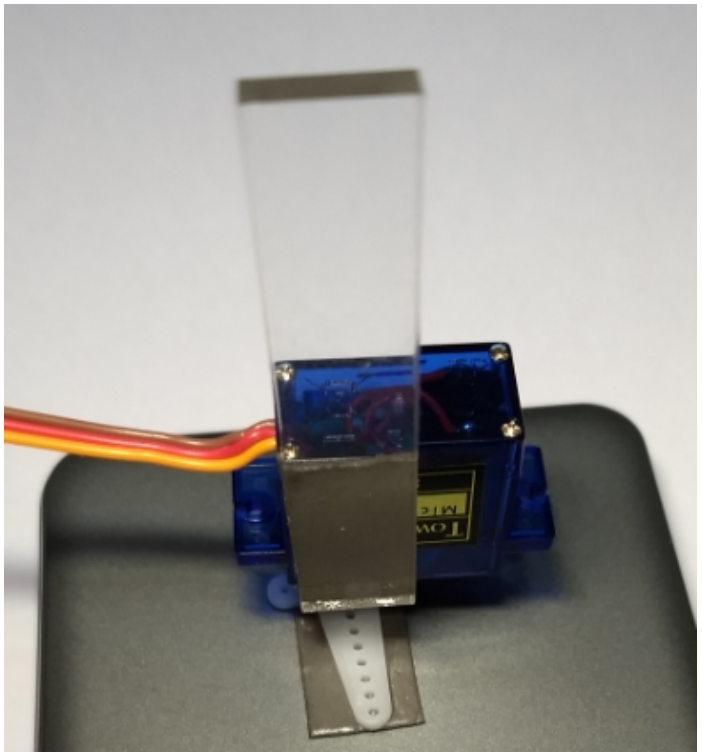
Servo Motor : Exercise

- Sonar
- ให้นำ Ultrasonic ต่อกับ
  เซอร์โวมอเตอร์ ตามรูป
- ใช้เทป 2 หน้าช่วยยึด
- แล้วเขียนโปรแกรมสแกนหาวัตถุ
  ที่อยู่ในระยะที่กำหนด
  เมื่อพบแล้ว ให้หยุดหมุน
  หากหลุดให้กลับไปสแกนใหม่
- Option : เราสามารถเอา Ultrasonic
  อีกตัวมารบกวนการทำงานของ Sonar ได้หรือไม่

Servo Motor : mini robot arm

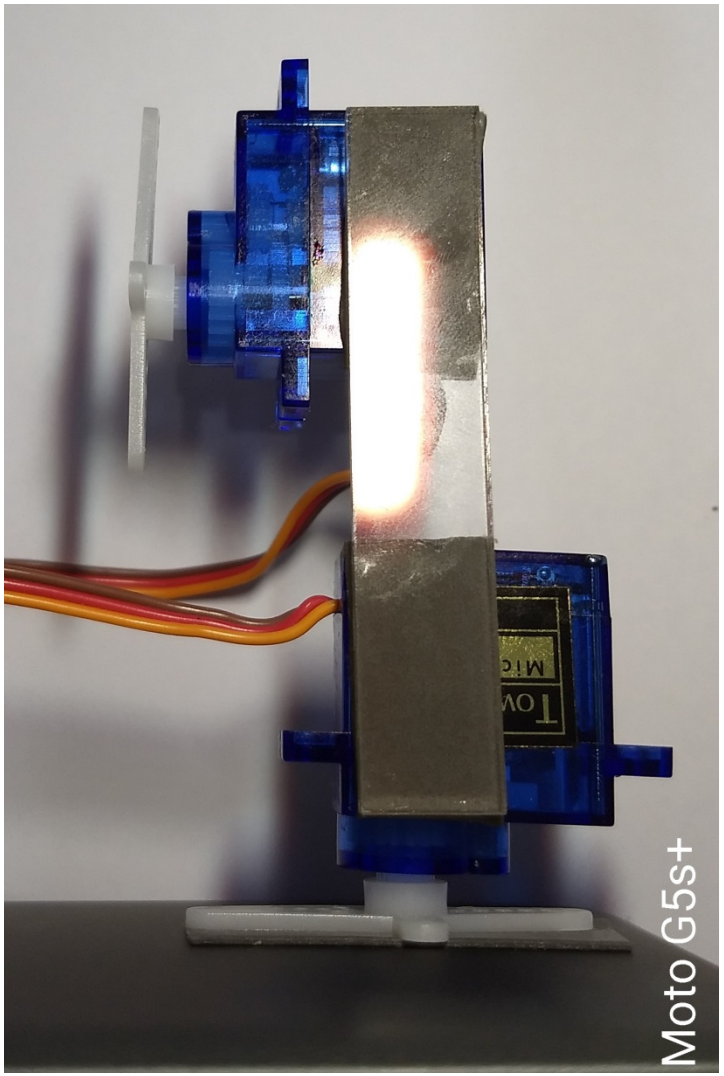
- ให้ประกอบเซอร์โวมอเตอร์ตามรูปตามลำดับ
  - นำก้านพลาสติกไขติดกับเซอร์โวมอเตอร์ตามรูป (ให้ใช้ตัวที่ไม่มีปีก)

  - จากนั้นนำไปติดกับฐานตามรูป (ใช้อะไรก็ได้ หรือยังไม่ติดก็ได้) จากนั้นนำแผ่นพลาสติกติดตามรูป



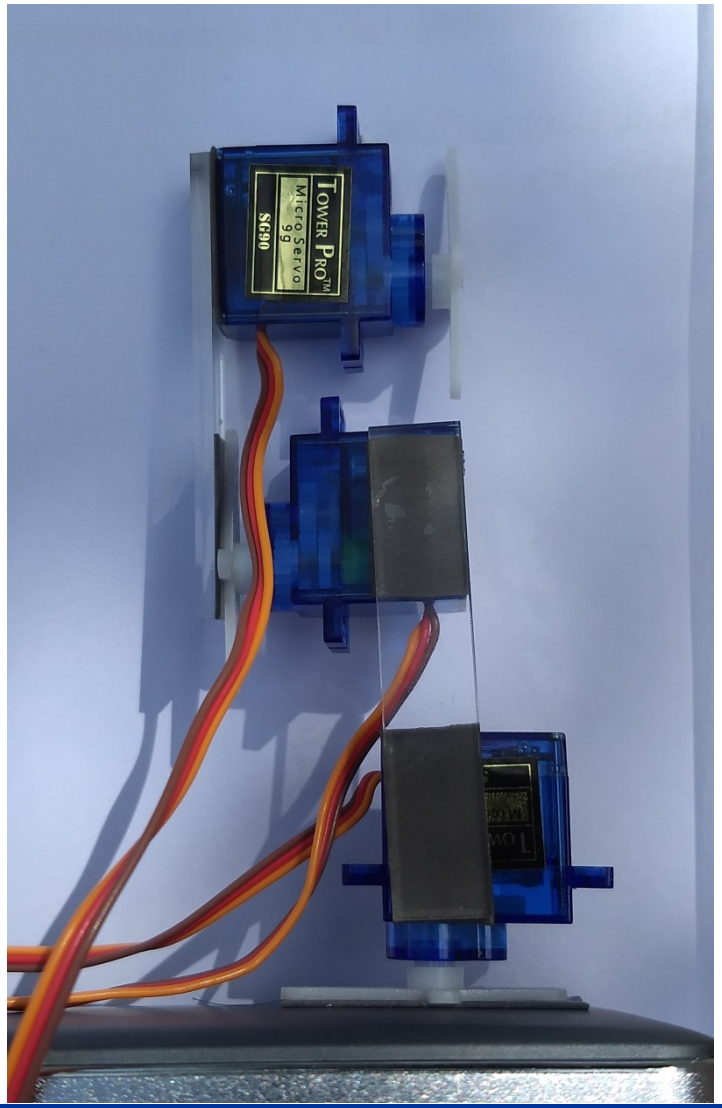
Servo Motor : mini robot arm

- นำเซอร์โวมอเตอร์อีกตัวมาติดก้าน 4 แฉก แล้วนำไปติดกับแผ่นพลาสติก ตามรูป (การเสียบต้องให้เซอร์โวมอเตอร์อยู่ที่ 90 องศา) (เพื่อให้ติดแน่น ให้แกะสติ๊กเกอร์ของเซอร์โวออกก่อน) (ตรวจสอบว่าพื้นผิวสะอาด ไม่มีน้ำมันก่อนติด เพราะจะทำให้ติดไม่แน่น)



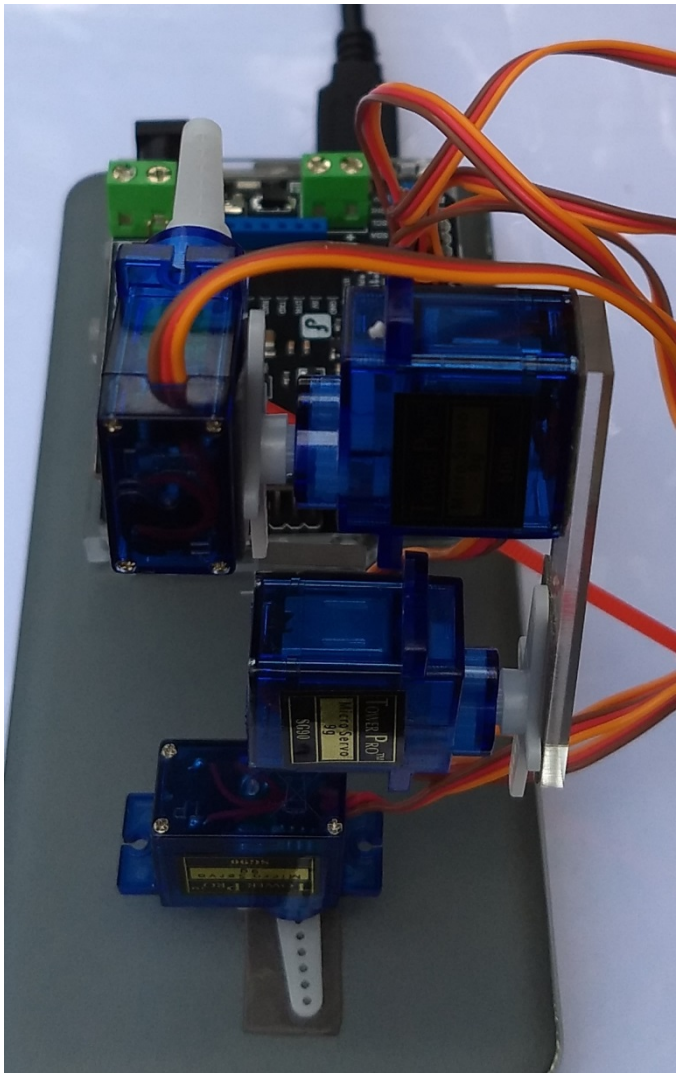
Servo Motor : mini robot arm

- นำเซอร์โวมอเตอร์ตัวที่ 3 มาติดก้าน 4 แฉก แล้วใช้แผ่นพลาสติกอีกแผ่นมาเชื่อมกับเซอร์โว 2 ตัวก่อนหน้านี้



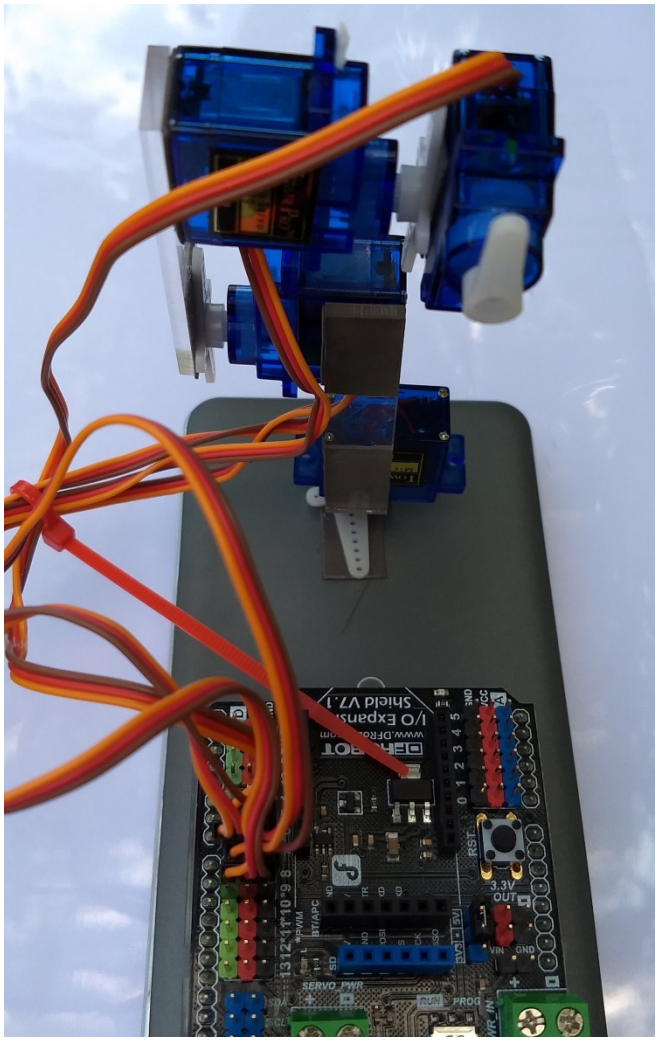
Servo Motor : mini robot arm

- นำเซอร์โวมอเตอร์ตัวที่ 4 มาติดก้านแขนข้างเดียว แล้วนำไปติดตั้งกับเซอร์โวมอเตอร์ตัวที่ 3 ตามรูป



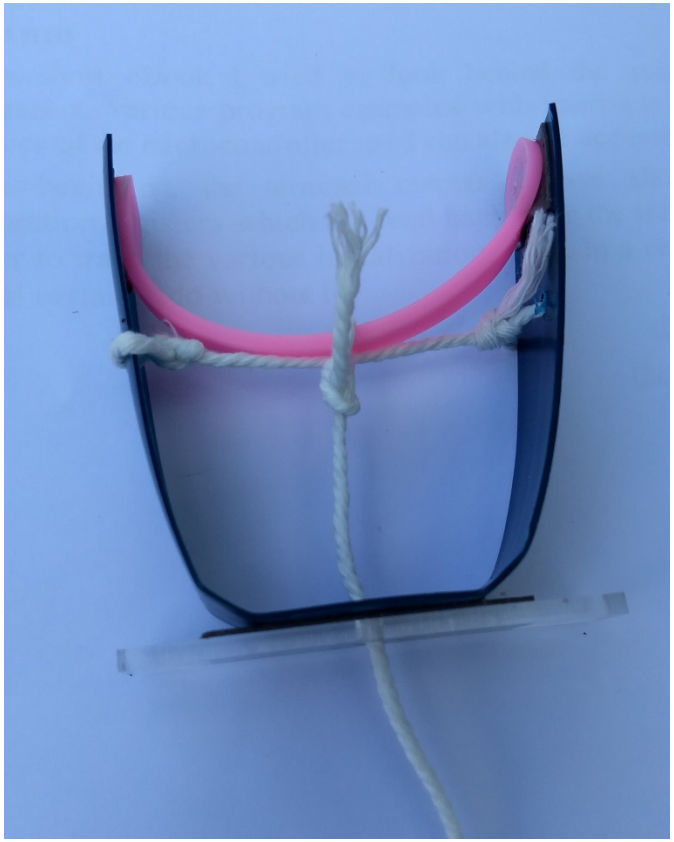
Servo Motor : mini robot arm

- นำสายสัญญาณไปเสียบบนบอร์ด Shield แล้วเขียนโปรแกรมทดสอบการหมุนของแต่ละแกน

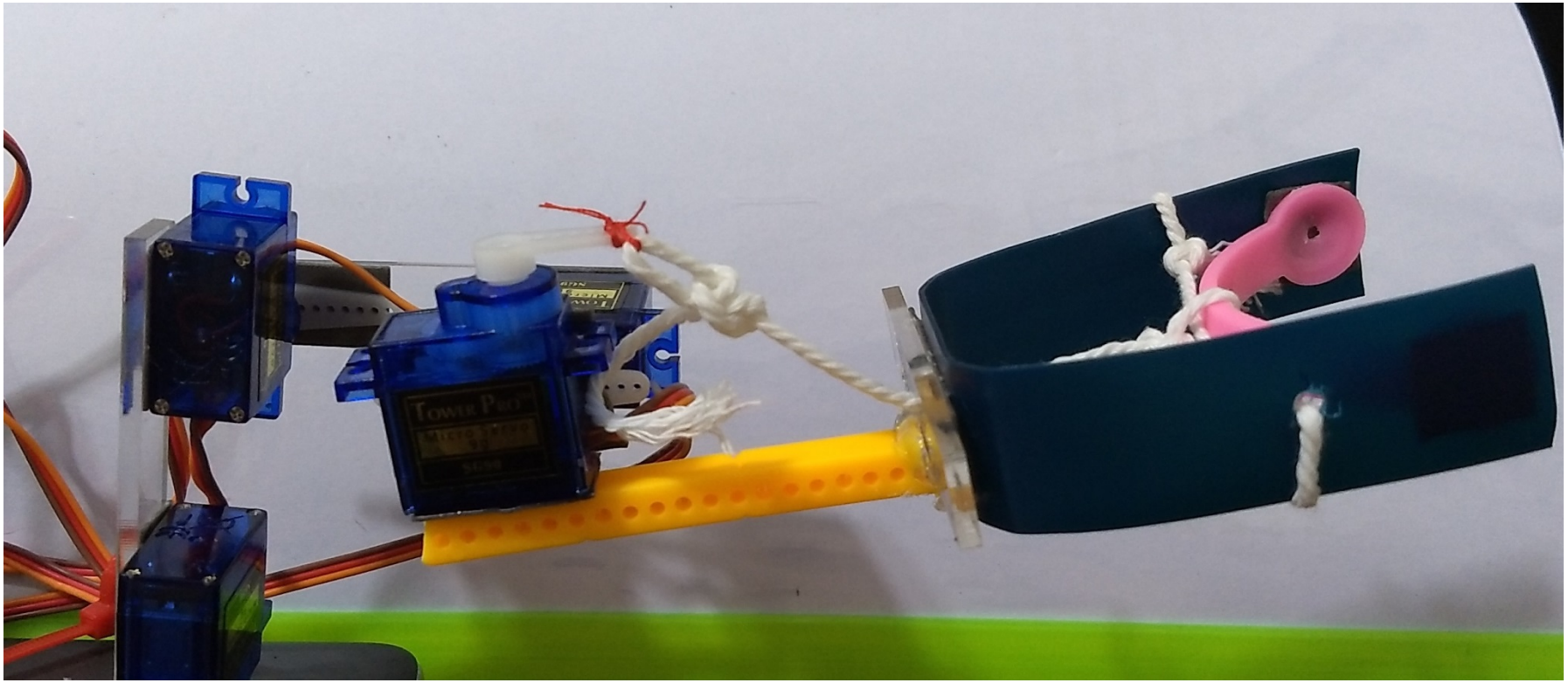


Servo Motor : mini robot arm

- ทำส่วนมือจับ โดยหาวัสดุเหลือใช้ เช่น ขวดพลาสติก ตัดเป็นแนวยาว (ควรยาวมากกว่า 16 ซม.) เจาะรู ผูกเชือก โดยเมื่อดึงเชือกแล้ว ต้องจับฝาขวดน้ำได้
- สามารถสร้างสรรค์ได้ตามต้องการ



Servo Motor : mini robot arm



Assignment #9 : Robot Arm

- การส่งงาน (4 คะแนน)
  - สร้าง Robot Arm
  - เขียนโปรแกรมควบคุมแขนกล โดยต้องสามารถหยิบฝาขวดน้ำที่อยู่บนพื้น แล้วนำไปหย่อนในถ้วย (ไม่ควรใช้ถ้วยสูงมากนัก) โดยตำแหน่งของฝาขวดน้ำ และตำแหน่งของถ้วย ให้แต่ละกลุ่มกำหนดได้ตามความเหมาะสม (ถ้าไม่มีถ้วยมาด้วย จะใช้ถ้วยเจ้าจุก)
  - ให้ demo กับอาจารย์หรือ staff
  - ส่งงานประกอบด้วย 1) รายงาน วิธีคิด การทำงานของโปรแกรมโดยย่อ และรูปถ่าย 2) Link VDO บน Youtube 3) โปรแกรม
  - ส่งงานใน mycourseville

For your attention