อินพุท - เอ้าท์พุท (Input - Output)

การเขียนโปรแกรมกับ M5Stack คล้ายกับการเขียนโปรแกรมบนบอร์ด Arduino เป็นสิ่งที่ไม่เหมือนกับการเขียนโปรแกรมกับคอมพิวเตอร์ เราใช้เขียนการทำงานกับบอร์ด นี้ เพียงเพื่อควบคุม การรับ-ส่งข้อมูลของเซ็นเซอร์ต่างๆ เพื่อประมวลผลในพฤติกรรมต่างๆ ในความจำกัดของทรัพยากร ที่มีมากกว่า คอมพิวเตอร์ทั่วไป แต่ข้อจำกัดนี้ ก็มีข้อดี คือ เล็ก ใช้พลังงานไฟฟ้าน้อย ราคาย่อยเยา แต่ข้อเสียคือ เราไม่อาจจะทำให้ อาร์ดูโน ประมวลผลรูปภาพได้ เราทำได้แค่งานเล็กๆ น้อย ในการรับ-ส่งข้อมูลพื้นฐานเท่านั้น การทำงานที่หนักกว่านั้น จำเป็นต่อเชื่อมกับอุปกรณ์อื่น ๆ แต่การประมวลผลในระดับนี้ ก็จำเป็นสำหรับงานพื้นฐาน เช่น การควบคุม การปิด-เปิด อุปกรณ์อิเล็กทรอนิกส์ ซึ่งนำไปใช้ทำงานได้หลากหลายในชีวิตประจำวัน ไม่ว่าจะเป็นการ ให้เปิดกล่องวงจรปิด เริ่มทำงาน เมื่อมีผู้บุกรุกเข้าพื้นที่ตรวจสอบ แล้วส่งข้อมูลไปยังอินเตอร์เน็ต การตรวจสอบความชื้น แสง เพื่อการเปิด-ปิดน้ำ ในการดูแลต้นไม้ และอื่นๆ ที่จะประยุกต์ใช้ได้

เริ่มต้น Hello

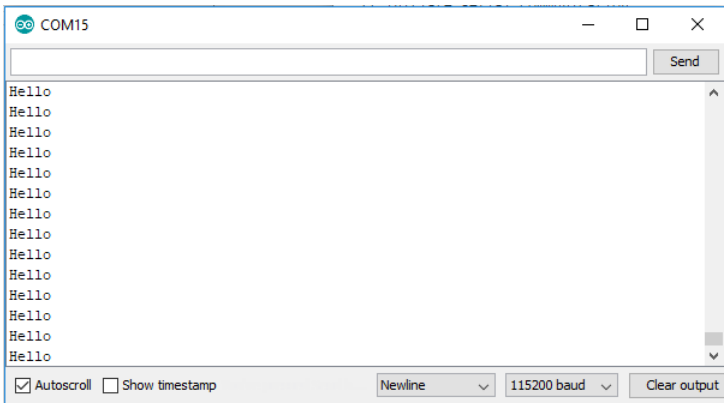
เพื่อเป็นกำลังใจในการใช้งานครั้งแรก ก็เหมือนกับการเขียนโปรแกรมภาษาคอมพิวเตอร์อื่น ๆ ที่มักเริ่มต้นเขียนโปรแกรมแรกที่เรียกว่า Hello world ใน M5Stack ก็เช่นเดียวกัน ให้ เขียนโปรแกรมทักทายกัน เป็นการเขียนผ่าน Arduino IDE แล้วลงโปรแกรมที่เขียนนี้ไปยังบอร์ด M5Stack โดยยังไม่ต้องต่ออุปกรณ์อิเล็กทรอนิกส์อื่น ด้วยการทำตามตัวอย่างต่อไปนี้

ตัวอย่าง 1 สวัสดี M5Stack

```
void setup() {
  // Initiate serial communication
  Serial.begin(115200);//baud rate :115200 bit per second
}

void loop() {
  Serial.println("Hello M5Stack");
}
```

ต่อไปก็เปิดเมนู Tool > Serail Monitor และกำหนด อัตราส่งข้อมูล 115200 ดังรูปต่อไปนี้



รูป 1 Serial Monitor

การสื่อสารผ่าน UART

จากตัวอย่างที่ผ่านมา ถือว่าเป็นการสื่อสารผ่าน UART0 ใน ESP32 มี UART1, UART2 สำหรับ UART0 ใช้ช่องทางสื่อสารของ USB ที่ใช้ในการเขียนโปรแกรมกับ ESP ส่วย UART ที่เหลือใช้ต่อกับอุปกรณ์

UART จัดเป็นการสื่อสารอนุกรมชนิดหนึ่ง และเป็นการสื่อสาร แบบ อะซิงโคนัส (Asynchronous) และยังการสื่อสารแบบอนุกรมแบบอื่น คือ I2C, I2S, SPI ที่จัดเป็นแบบ ซิงโครนัส

การสื่อสารของ UART ใช้การรับการส่งสัญญาณ เพียง 2 เส้นทาง คือ Tx และ Rx ซึ่งใช้เพื่อส่ง และเพื่อรับ ตามลำดับ การต่ออุปกรณ์จะต้องไขว้กัน เช่น Tx ต่อกับ Rx และ Rx ต่อกับ Tx

ฟังก์ชันสำคัญของการสื่อสาร UART
void Serial.being(long speed)
void Serial.print(val) หรือ print(val, format) หรือ println(val)
byte Serial.available( ) เป็นการตรวจสอบว่ามีข้อมูลหรือไม่
int Serial.read( ) เป็นอ่าน char หรือ ไบท์ ซึ่งเป็นการอ่านได้ที่ละหนึ่งอักษร
String Serial.readString( ) เป็นอ่านข้อมูล String ซึ่งมีหลายอักษร
void Serial.setTimeout(long timeout) เป็นตัดสินใจว่าจะหมดเวลาสิ้นสุดข้อมูลที่เวลาที่ระบุ

ตัวอย่างต่อไปใช้ ใช้งานใช้ทำงานผ่าน UART0 หรือ USB ซึ่งดูผลที่หน้าต่าง Serial Monitor

ตัวอย่าง 2 LED เปิด - ปิด

```
void setup() {
  Serial.setTimeout(100);
  Serail.begin(115200);
}

void loop() {
  if(Serial.avarable( ) > 0){
    String str = Serial.readString( );
    Serial.print(“Echo:”);
    Serial.println(str);
  }
}
```

สำหรับ UART1, UART2 ใช้กับ Serial1 และ Serial2 ตามลำดับ มีขาเป็น T1, R1, T1, และ R2 มีฟังก์ชันการใช้งานเหมือนกับ UART0 ทุกอย่าง

ดิจิตอล digitalWrite( )

หลอดไฟ ในที่นี้คือ LED (Light-Emitting Diode) เป็นอุปกรณ์อิเล็กทรอนิกส์ที่เปล่งแสงขนาดเล็ก ใช้พลังงานไฟฟ้าน้อย ในตัวอย่างโปรแกรมต่อไป เราใช้ LED ต่อเชื่อมกับ MStack ให้ทำงานแบบกระพริบทุกๆ หนึ่ง วินาที ลองพิจารณาโปรแกรมต่อไป และการต่อเชื่อม LED กับบอร์ด อาร์ดูโน ตามรูป 2

ตัวอย่าง 3 LED เปิด - ปิด

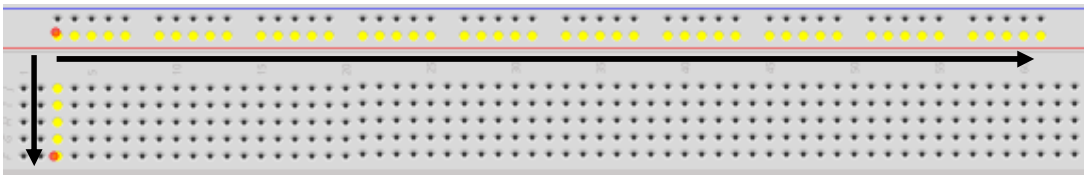
```
void setup() {
  pinMode(2, OUTPUT);

}

void loop() {
  digitalWrite(2,1);
  delay(1000);
  digitalWrite(2,0);
  delay(1000);
}
```

รูป 2 ต่อ LED กับ M5Stack

ก่อนอื่น เราต้องรู้จัก บอร์ดขนมปัง หรือ เรียกว่าอย่างเป็นทางการว่า Solderless breadboard แต่บางครั้งอาจได้ยินบางคนเรียกว่า Solderless หรือบางคนก็เรียกว่า Breadboard ตัวบอร์ดนี้เราจะใช้เชื่อวงจรไว้ด้วยกัน แนวเส้นสีเหลือง (ดู**รูปที่ 3**) จะเป็นจุดที่ใช้เชื่อมต่อกัน สังเกตว่า แนวนอน เป็นแนวยาว จะเชื่อมต่อกันหมด หมายความว่า ถ้าเราใช้สายไฟ ต่อในจุดใดก็ได้ของแนวนอน จะถือว่าเชื่อมต่อกันอยู่ เช่น เดียวกับแนวตั้ง จะต่อจุดใดของแนวตั้งก็ถือว่าเชื่อมต่อกันหมด



รูป 3 แนวการต่อเชื่อม ของบอร์ดขนมปัง (Breadboard)

กฎของโอห์ม

เพื่อทำความเข้าใจเล็กน้อย เกี่ยวการไฟฟ้า ที่ไหลในวงจรทำงานอย่างไรกัน เพราะจะส่วนสำคัญต่อการใช้งานอุปกรณ์อิเล็กทรอนิกส์ อย่างน้อยก็เพื่อเลือกใช้งานไม่เกิดความเสียหายได้

ก่อนอื่นให้นึกถึง ระบบน้ำประปาที่บ้าน โดยเปรียบเทียบว่า กระแสน้ำ ที่ไหลตามท่อ ปริมาณน้ำที่ไหลนั้นเปรียบเสมือนกระแสไฟฟ้า ซึ่งมีหน่วยวัดเป็น แอมแปร์ (Amperes) หรือเรียกสั้นว่า แอมป์ เทียบแรงดันน้ำเป็นแรงดันไฟฟ้า มี

หน่วยเป็น โวลต์ (Volt) และเทียบ ความต้านทาน เป็นเหมือนท่อน้ำ มีหน่วยเป็น โอห์ม (Ohm) ใช้สัญลักษณ์ Ω แทนหน่วยนี้ได้ หลักการคำนวณ ค่าเหล่านี้ มีสูตรสั้นๆ ว่า

$$V = I \times R \quad (1)$$

โดยที่ V แทนโวลต์ I แทนกระแส และ R แทน ความต้านทาน เราจำเพียงสูตร (1) ก็พอ การหาค่า I หรือ R ก็มาจากกลับไป-มาของสูตรนี้ ลองพิจารณาสูตรต่อไปนี้ ที่มาจากการแปลงจากสูตรแรก

$$I = V / R \quad (2)$$

$$R = V / I \quad (3)$$

จากสูตร (1) นี้ ถ้าเพิ่ม กระแส (I) หรือ ความต้านทาน (R) ก็จะทำให้ แรงดัน (V) สูงขึ้น เพราะเป็นมาจากผลคูณของตัวนี้ สำหรับสูตร (2) ถ้า ต้องการให้ กระแส มีมาก ก็ต้องให้ตัวหาร (R) มีค่าน้อยๆ หรือตัวถูกหาร (V) มีค่ามากๆ สำหรับสูตร (3) เราก็วิเคราะห์ในทำนองเดียวกัน

ในกรณี การหา **R** ที่เหมาะสม เมื่อไฟฟ้ามีแรงดัน 5 **V** และ **LED** ทั่วไปใช้แรงดันฟ้าไม่เกิน 2.3 **V** โดยใช้กระแสประมาณ 20 **mA** ตัวต้านทาน ควร มีขนาด 135 โอห์ม คำนวณได้จากสูตร

$R = (V_{cc} - V_f)/ I_f = (5-2.3)/0.02 = 135$ โอห์ม

ปุ่ม A, B, C ของ M5Stack

ปุ่ม 3 ปุ่ม มีชื่อ BtnA, BtnB, และ BtnC ตามลำดับ ใช้สำหรับอ่านอินพุท ผลการอ่าน จะได้ค่า 1 ซึ่งใช้ฟังก์ชันของ M5 มีให้เลือกใช้ดังนี้

`read()` ใช้หรับอ่าน ค่า เช่น M5.BtnA.read( ) ซึ่งหากกดจะคืนค่า 1

`isPressed()`
ใช้สำหรับตรวจสอบการกด หากมีการกดปุ่ม จะค้นค่า 1 เมื่อปล่อยจะคืนค่า 0

`wasPressed()`
ใช้สำหรับตรวจสอบการกดปุ่ม เพียงครั้งเดียวที่เคยกดแล้วจะคืนค่า 1 แต่ถ้าไม่เคยกดจะคืนค่า 0

`pressedFor()`
ใช้สำหรับ ตรวจสอบการกดตามเวลาที่ระบุ เช่น ได้กำหนดให้นับการกดแช่นานเกิน 3 วินาที จะค้นค่า 1 ถ้าไม่ถึงจะคืนค่า 0

M5.update()
ใช้สำหรับปรุงข้อมูลใหม่ ของปุ่ม ซึ่งจะต้องเขียนก่อน การใช้งานปุ่ม

ตัวอย่าง 4 ทดสอบการกดแช่นาน 2 วินาที

```
#include <M5Stack.h>

void setup() {
```

```
  M5.begin();
}

void loop() {
  M5.update();
  M5.Lcd.clear();
  M5.Lcd.setCursor(0, 0);
  if (M5.BtnA.pressedFor(2000)) {
    M5.Lcd.printf("Button A was pressed for more than 2 seconds.");
    delay(1000);
  }
}
```

สร้างสวิสปิด-เปิด LED

ที่ได้สร้างโปรแกรมปิด-เปิด ที่ผ่านมา จะเห็นว่า โปรแกรมจะทำงานปิด และปิด ไปเลื่อย แต่ส่วนใหญ่การใช้งานจริง เราควรมีอะไรที่ใช้เปิด และปิด เหมือน หลอดไฟ ที่บ้านของเรา มากกว่า ต่อไปนี้ เราลองสร้างสิ่งที่ว่านี้กัน นอกจากนี้เราควรใส่ตัวต้านทานลงไปลดกระแสไฟฟ้าที่จะผ่าน LED ด้วย

ตัวอย่างต่อไปนี้ ใช้ BtnA ทำหน้าที่กด ผ่านการตรวจสอบของฟังก์ชัน isPressed( ) ถ้าตรวจพบการกดก็จะสว่าง ไป 1 วินาที (หลังจากกดแล้ว ปล่อย) การตรวจสอบทำได้โดยใช้ฟังก์ชัน if ... else นอกจากนี้ควรต่อ LED กับ ตัวตานทาน ขนาด ไม่มาก ก่อนต่อลง กราวด์ (G)

ตัวอย่าง 5 ปุ่ม A เปิดไฟ LED

```
#include <M5Stack.h>

void setup() {
  M5.begin();
  pinMode(2, OUTPUT);
}

void loop() {
  M5.update();
  if (M5.BtnA.isPressed()) {
    digitalWrite(2,1);
    delay(1000);
  }
  else{
    digitalWrite(2,0);
  }
}
```

อนาลอก dacWrite( )

จากรูปก่อนหน้าที่ถ้าเราเพิ่ม ปุ่มกดปิด-เปิด ไฟ และทำ LED กระพริบมาแล้ว ในรูปแบบ digitalWrite( ) ที่ให้ผลเป็น 0 กับ 1 แต่สำหรับ dacWrite( ) ให้ผลเป็นค่าช่วงได้ (ไม่มีเพียงแค่ 0 กับ 1) นั้นหมายความว่า เราทำให้ LED เป็นไฟหรี่ได้ ตามค่าช่วง

จากตัวอย่างต่อไปนี้ สร้างเป็นไฟหรี่ ให้ค่อยๆ สว่าง และค่อยๆ ดับ ตามค่าช่วงที่ใส่เข้าไป สำหรับ pinMode( ) ไม่จำเป็นสำหรับค่าอนาลอก นอกจากนี้ขาที่ต่อกับ LED ที่เป็นอนาลอก มีเพียง G25 กับ G26 เท่านั้น ในตัวอย่างกำหนดให้ ขาที่ G25 ต่อกับ LED

กลไกการทำไฟหรี่ ใช้ การวนซ้ำของ for ที่ค่อยเพิ่มค่าเป็นให้ LED ค่อยๆ สว่าง และใช้อีก for ที่ค่อยๆ ลง ค่าลงเพื่อให้ค่อยๆ ดับ ทำซ้ำวนกันไปมาของ ทั้ง for

ตัวอย่าง 6 ไฟหรี่ LED

```
#include <M5Stack.h>

const int LED = 25;
void setup() {
  M5.begin();
}

void loop() {
  M5.update();
  for(int i = 0; i<255; i++){
    dacWrite(LED,i);
    delay(100);
  }
  for(int i = 255; i>0; i--){
    dacWrite(LED,i);
    delay(10);
  }
}
```

ดิจิตอล digitalRead()

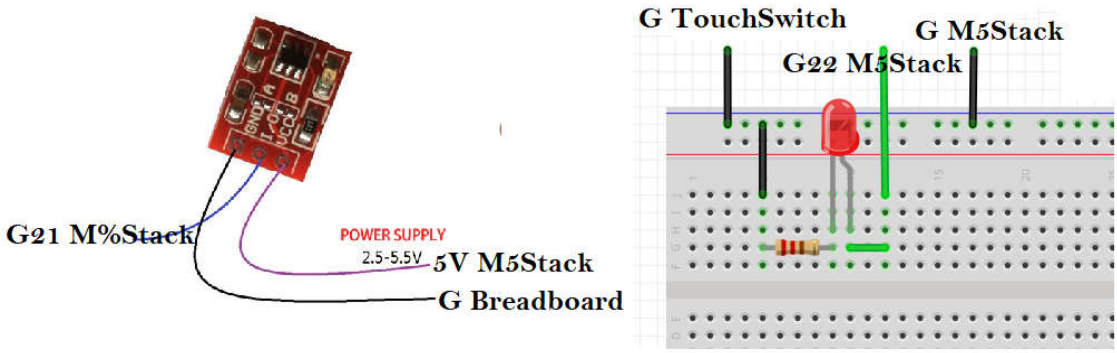
การอ่านข้อมูลดิจิตอล ด้วย digitalRead(pin) ให้ผลลัพธ์ เป็น 0 หรือ 1 แต่ก่อนใช้งาน ต้องกำหนด pinMode(pin) ก่อน เช่น ต้องการ อ่านข้อมูล จาก ปุ่มสัมผัส (TouchSwitch) จากตัวอย่างต่อไปนี้ ที่ต้องการให้ปุ่มสัมผัส มีการเตะ จะส่งสัญญา 1 แต่หากไม่มีการแตะจะส่งค่า 0 ในตัวอย่างใช้ LED แสดงผลการสว่างเมื่อมีการแตะ

ตัวอย่าง 7 ปิด-เปิดไฟ LED จาก ปุ่มสัมผัส

```
int touchSwitch = 21;
int output = 22;
void setup() {

  pinMode(touchSwitch, INPUT);
  pinMode(output, OUTPUT);
}
void loop() {
  if(digitalRead(touchSwitch)){
    digitalWrite(output, 1);
  }  else{
    digitalWrite(output, 0);
  }
```

```
}
```



รูป 4 การต่อ LED, Touch Switch, และ M5Stack

อนาลอก analogRead( )

การอ่านข้อมูลเป็นอนาลอก ใช้ฟังก์ชัน `analogRead(pin)` ใช้งานผ่านขา G35 และ G36 คำสังการอ่านนี้ จะต้องกำหนดค่าสูงสุด ผ่านฟังก์ชัน `analogSetWidth( )`

ในตัวอย่างต่อไปนี้ ใช้การอ่านผ่าน ตัวต้านทานปรับค่าได้ (Potentiometer) โดยดูผลลัพธ์กับ หน้าต่าง Serial Monitor

ตัวอย่าง 8 การอ่านค่าอนาลอกผ่าน ตัวต้านทานปรับค่าได้

```
const int POT = 35;   //The pin for Potentiometer

int val = 0;
void setup() {
  Serial.begin(115200);
}
void loop() {
  val = analogRead(POT);
  Serial.print("val: ");
  Serial.println(val);
  delay(1000);
}
```

รูป 5 การต่อ ตัวแทนทานปรับค่าได้ กับ M5Stack

ทดลองด้วยตนเอง

1. ให้สร้าง สวิทช์ ปิด-เปิด LED ผ่านช่องทางการสื่อสาร UART
2. ให้สร้าง สวิทช์ ปิด-เปิด LED เมื่อ แตะที่ ปุ่มสัมผัส ครั้งแรกให้ LED สว่าง และเมื่อแตะอีกครั้งให้ LED ดับ
3. ให้อ่านค่าอนาลอกจาก ตัวทานปรับค่าได้ เพื่อปรับความสว่างของ LED