ใบงานที่ 7

วิชา ไมโครคอนโทรลเลอร์เบื้องต้น **รหัสวิชา** 20104-2112

ชื่อหน่วย คำสั่งในการเขียนโปรแกรมควบคุมอุปกรณ์ภายนอกงานเชื่อมต่อบอร์ดคอนโทรลเลอร์กับอุปกรณ์อินพุต เอาต์พุต (ภาษา C++คืออะไร -การคอมไพเลอร์)

จุดประสงค์การเรียนรู้

1. จุดประสงค์ทั่วไป /บูรณาการเศรษฐกิจพอเพียง

1.1 อธิบายการใช้คำสั่งต่างๆในภาษา C++ได้ (ด้านความรู้)

1.2 สามารถเขียนโปรแกรมเพื่อควบคุมอุปกรณ์ภายนอกได้ (ด้านทักษะ)

1.3 ใช้วัสดุและอุปกรณ์อย่างเหมาะสมคุ้มค่ามากที่สุดดูแลรักษาเครื่องมือยืดอายุการใช้งาน (ด้านคุณธรรมจริยธรรม)

2. จุดประสงค์เชิงพฤติกรรม/บูรณาการเศรษฐกิจพอเพียง

2.1. ใช้ความรู้นำไปประยุคใช้งานด้านอิเล็กทรอนิกส์ได้อย่างถูกต้องและคุ้มค่า(ด้านความรู้)

2.2. ปฏิบัติงานได้ถูกต้องและสำเร็จภายในเวลาที่กำหนดอย่างมีเหตุผล (ด้านทักษะ)

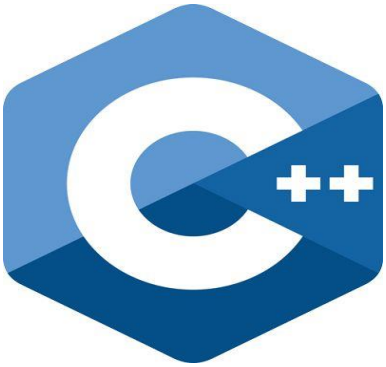
2.3. เตรียมความพร้อมด้านวัสดุอุปกรณ์สอดคล้องกับงานและใช้วัสดุอุปกรณ์อย่างคุ้มค่าประหยัด ตามหลักปรัชญาของเศรษฐกิจพอเพียง (ด้านคุณธรรมจริยธรรม)

เครื่องมือ/อุปกรณ์

1. บอร์ด Arduino UNO R3 สาย Upload
2. LED จำนวน 4 ดวง
3. ความต้านทาน 560Ω 4 ตัว
4. สายไฟ จัมเปอร์
5. คอมพิวเตอร์ PC หรือ Note Book 1 เครื่อง

รายการสอน

ภาษา C++ เป็นภาษาคอมพิวเตอร์เพื่อวัตถุประสงค์ทั่วไป ซึ่งสามารถเขียนโปรแกรมได้ทั้งแบบออบเจ็ค และการเขียนแบบปกติทั่วไป และยังมีเครื่องมืออำนวยความสะดวกในการจัดการและเข้าถึงระดับหน่วยความจำ นอกจากนี้มันยังถูกนำไปใช้ในการเขียนโปรแกรมแบบต่างๆ มากมาย เช่น โปรแกรมคอมพิวเตอร์ ระบบฝังตัว (Embedded) เว็บเซิร์ฟเวอร์ การพัฒนาเกม และแอพพลิเคชันที่ต้องการประสิทธิภาพอย่างสูง



ภาษา C++ เป็นภาษาที่ถูกออกแบบมาในการเขียนโปรแกรมระบบ ซึ่งมีประสิทธิภาพและความยืดหยุ่นในการออกแบบโปรแกรมสูง C++ เป็นภาษาที่ต้องคอมไพล์ก่อนที่จะนำไปใช้งาน ซึ่งสามารถพัฒนาได้ในหลายๆ แพลตฟอร์ม ซึ่งได้รับการสนับสนุนโดยองค์กรต่างๆ ที่ประกอบไปด้วย Free Software Foundation (FSF's GCC) LLVM Microsoft Intel และ IBM C++ นั้นถูกกำหนดให้เป็นภาษาที่เป็นมาตรฐานโดย International Organization for Standardization (ISO) ซึ่งเวอร์ชันล่าสุดนั้นเผยแพร่ในธันวาคม 2014 คือ ISO/IEC 14882:2014 หรือที่รู้จักกันในชื่อของ C++14 โดยที่ภาษา C++ ได้เริ่มกำหนดมาตราฐานครั้งแรกในปี 1998 คือ ISO/IEC 14882:1998 ภาษา C++ ถูกพัฒนาโดย Bjarne Stroustrup ที่ Bell Labs ตั้งแต่ปี 1979 ซึ่งในตอนแรกเป็นส่วนขยายของภาษา C โดยที่เขาต้องการที่จะพัฒนาภาษาที่มีประสิทธิภาพและยืดหยุ่นเหมือนกับภาษา C และยังมีคุณสมบัติใหม่ที่สูงกว่าสำหรับพัฒนาโปรแกรม Bjarne Stroustrup นักวิทยาศาสตร์คอมพิวเตอร์ชาวเดนมาร์ก ได้สร้างภาษา C++ ขึ้นในปี 1979 โดยเขาเริ่มจาก "C with Classes" ซึ่งเป็นภาษาก่อนหน้าของภาษา C++ แรงจูงใจสำหรับการสร้างภาษาใหม่นั้นมีต้นกำเนิดมาจากประสบการณ์ในการเขียนโปรแกรมสำหรับงานวิจัยในการศึกษาระดับปริญญาเอกของเขา ในขณะที่ Stroustrup เริ่มต้นการทำงานที่ AT&T Bell Labs เขามีปัญหาในการวิเคราะห์ UNIX kernel ซึ่งเกี่ยวกับ distributed computing จากการจดจำในประสบการณ์ปริญญาเอกของเขา Stroustrup ตั้งใจว่าจะเพิ่มความสามารถให้ภาษา C กับคุณสมบัติที่เหมือนภาษา Simula เขาเลือกภาษา C เพราะว่ามันเป็นภาษาเขียนโปรแกรมเพื่อวัตถุประสงค์ทั่วไป ที่ทำงานเร็ว สะดวกใช้งานง่ายและใช้กันอย่างแพร่หลาย จนกระทั่งในปี 2011 มาตฐานของ C++11 ได้ถูกเผยแพร่ โดยการเพิ่มคุณสมบัติใหม่เข้ามามากมาย รวมทั้งการเพิ่มเติมขนาดของไลบรารี่มาตรฐาน และให้ความสะดวกแก่โปรแกรมเมอร์ภาษา C++ เป็นอย่างมาก หลังจากบทเรียนนี้เสร็จสิ้นคุณจะเข้าใจและสามารถสร้างแอพพลิเคชันของคุณเอง เพราะว่าภาษา C++ เป็นพื้นฐานที่นำไปสู่การกำเนิดภาษาอื่นๆ และเป็นภาษาที่พัฒนามาจากภาษา C การเริ่มต้นเรียนรู้กับภาษา C++ ยังช่วยให้คุณเข้าใจและเรียนภาษาอื่นได้ง่ายขึ้น เช่น ภาษา C# ภาษา Java หรือ ภาษา PHP เป็นต้น

คอมไพเลอร์

คอมไพเลอร์คือโปรแกรมคอมพิวเตอร์หรือกลุ่มของโปรแกรมที่แปลงซอสโค้ดที่เขียนขึ้นในภาษา C++ ไปเป็นภาษาคอมพิวเตอร์ (Target language) หลังจากที่ทำการแปลงแล้วจะได้ข้อมูลในรูปแบบของฐานสอง (Binary) ที่เรียกกันว่า Object code เหตุผลที่ต้องแปลงโปรแกรมจากภาษาเขียนโปรแกรมไปเป็นภาษาเครื่องโดยคอมไพเลอร์ก็เพื่อสร้างโปรแกรมที่สามารทำงานได้ (Executable program)

คอมพิวเตอร์สามารถเข้าใจแค่ภาษาเครื่อง ภาษาที่ประกอบไปด้วยตัวเลข 1 และ 0 เราจำเป็นต้องใช้คอมไพเลอร์เพื่อแปลงโปรแกรมที่เราเขียนไปเป็นภาษาเครื่องที่ให้คอมพิวเตอร์สามารถทำงานได้

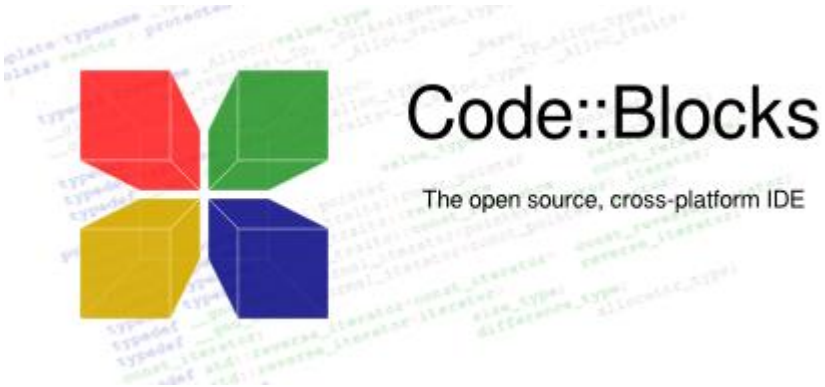
คอมไพเลอร์ช่วยให้โปรแกรมเมอร์พัฒนาโปรแกรมของพวกเขาได้อย่างง่ายดายในการเขียนโปรแกรมด้วยภาษาระดับสูง อย่างเช่น ภาษา C++

เครื่องมือพัฒนาโปรแกรม

ทางที่ง่ายที่สุดที่คุณจะสามารถเขียนโปรแกรมในภาษา C++ ได้นั้นคือการใช้ IDE ซึ่ง IDE เป็นการรวบรวมชุดโปรแกรมที่จำเป็นสำหรับในการพัฒนาโปรแกรม มันเป็นโปรแกรมที่อำนวยความสะดวกและให้เครื่องมือที่จำเป็นสำหรับในการพัฒนาโปรแกรม โดยปกติแล้ว IDE จะประกอบไปด้วยโปรแกรมสำหรับเขียนและแก้ไขโค้ดที่มาพร้อมกับเครื่องมือโดยอัตโนมัติ คอมไพเลอร์ และตัวดีบักโปรแกรม

สำหรับในบทเรียนนี้ โปรแกรมที่เป็นที่นิยมที่สุดที่เราจะแนะนำคือ Code blocks มันสามารถใช้ได้บนแพลตฟอร์มต่างๆ เช่น Windows Linux และ MacOS ซึ่งมากับคอมไพเลอร์ GCC (MingW / GNU GCC)

MSVC++ clang Digital Mars Borland C++ 5.5 Open Watcom และอื่นๆ Code blocks นั้นสนับสนุนการเขียนทั้งภาษา C++ และภาษา C



Arduino ภาษา C/C++

Arduino ภาษา C/C++

ภาษาของการเขียนโปรแกรมใช้งานArduino Board

ภาษา C/C++ โดยประกอบด้วย Structure,values (variables and constants) และ Functions

ฟังก์ชั่นหลัก(Structure)

เป็นฟังก์ชั่นหลักในการเขียนโปรแกรม จำเป็นต้องมีในทุกโปรแกรม

Setup()

คือ ฟังก์ชั่นใช้ในการประกาศค่าเริ่มต้น ตำแหน่งพอร์ตที่ใช้งาน รวมถึงฟังก์ชั่นที่อยู่ไลบารี่ที่ใช้งาน เป็นฟังก์ชั่นที่ทำงานเพียงครั้งเดียว จะทำงานทุกครั้ง ที่มีการรีเซต หรือรีบูตเครื่องใหม่ เท่านั้น

เช่น

```
intbuttonPin = 3;// การตั้งค่าตัวแปร buttonPin เท่ากับ 3
void setup()
{
    Serial.begin(9600); //ประกาศการใช้งานการสื่อสารรับส่งข้อมูลผ่าน พอร์ตRS232
     pinMode(buttonPin, INPUT);// การตั้งค่าโหมด ของตัวแปรแบบคงที่ buttonPin เป็นโหมด อินพุต
}
void loop()
{
  // ...
}
```

Loop ()

คือ ฟังก์ชั่นใช้ในการเขียนโค้ดโปรแกรมการทำงานของArduinoเป็นฟังก์ชั่นการวนลูปไปเรื่อยๆ

เช่น

```
constintbuttonPin = 3; // การตั้งค่าตัวแปร buttonPin เท่ากับ 3
void setup()
{
```

```
    Serial.begin(9600);//ประกาศการใช้งานการสื่อสารรับส่งข้อมูลผ่าน พอร์ตRS232
    pinMode(buttonPin, INPUT);// การตั้งค่าโหมด ของตัวแปรแบบคงที่  buttonPin เป็นโหมด อินพุต
}

void loop ()
{
    if (digitalRead(buttonPin) == HIGH)// ตรวจสอบค่าอินพุตที่รับมา เป็น HIGH ใช่หรือไม่
        Serial.write('H');// ใช่   ส่งค่าอักษร H ผ่าน พอร์ตRS232
    else
        Serial.write('L');// ไม่  ส่งค่าอักษร L  ผ่าน พอร์ตRS232
    delay(1000); //หน่วงเวลา 1วินาที
}
```

ชุดคำสั่งในการควบคุม (Control Structures)
เป็นชุดคำสั่งในการใช้ในการตัดสินใจหาทางออก เพื่อใช้ในการทำงาน
If
คือ คำสั่งในการตัดสินใจ แบบตัวเลือกเดียว โดยใช้งานร่วมกับ And, Or Not, ==, !=, <, >เพื่อใช้ในการตัดสินใจในการหาคำตอบ เช่น

```
if (ตัวแปร> 50)
{
  // .....
}

if (ตัวแปร> 120)digitalWrite(LEDpin, HIGH);

if (ตัวแปร> 120)
 digitalWrite(LEDpin, HIGH);

if (ตัวแปร> 120) { digitalWrite(LEDpin, HIGH); }

if (ตัวแปร> 120) {
 digitalWrite(LEDpin1, HIGH);
 digitalWrite(LEDpin2, HIGH);
}
```

If...else

คือ คำสั่งในการตัดสินใจ แบบหลายตัวเลือก โดยใช้งานร่วมกับ And, Or Not, ==, !=, <, >เพื่อใช้ในการตัดสินใจในการหาคำตอบ เช่น

```
If (pinFiveInput< 500)
{
  // action A
}
Else
{
  // action B
}
```

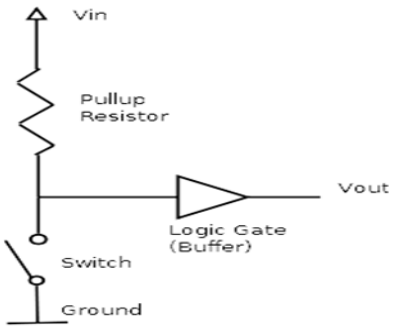
การต่อสวิทช์แบบ pull-up และ pull- down

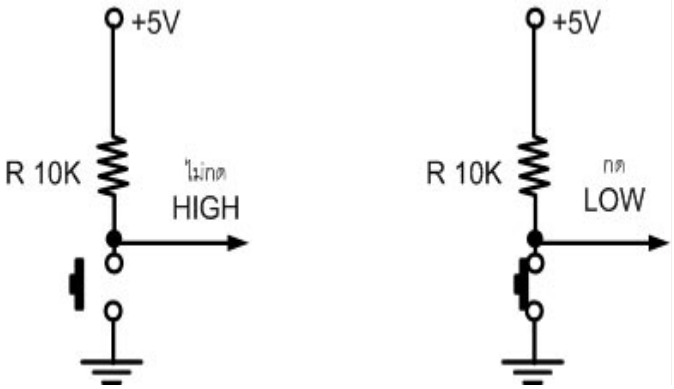
Pull up, Pull down คืออะไร คือการต่อ ตัวต้านทานที่ขา input ของไมโครคอนโทลเลอร์

เหตุผลที่ต้องต่อ ถ้าเราต่อสวิทช์ หรือ เซนเซอร์ต่างๆ เข้ากับไมโครคอนโทรเลอร์ตรงๆ อย่างเดียว อาจจะทำให้เกิดสัญญาณรบกวนได้ในกรณีที่ input ถูกลอยขาไว้ ไม่ได้จ่าย logic high หรือ low เช่น การต่อสวิทช์ ถ้าเรากดสวิทช์ จะทำให้มี logic high จ่ายให้กับ input ของไมโครคอนโทลเลอร์ แต่ถ้าเราปล่อยสวิทช์ทำให้ ขา input ถูกลอยไว้ ไม่ได้ต่อลงกราวหรือ logic low ดังนั้นจึงต้องต่อ Pull up, Pull down เพื่อให้แน่ใจว่าเป็น logic high หรือ low เสมอ ถึงแม้ว่าจะไม่มี input ป้อนเข้ามาไม่ได้ใช้เฉพาะเจาะจงใน Arduino อย่างเดียว ทุกอุปกรณ์อิเล็กทรอนิกส์เลยที่บอกว่าขาอินพุทเป็น High Impedanceตามปกติ ตัวต้านทานที่ใช้ในวงจร Pull-up หรือ Pull-down จะใช้ประมาณ 5k -20kΩ

Pull-up Resistor

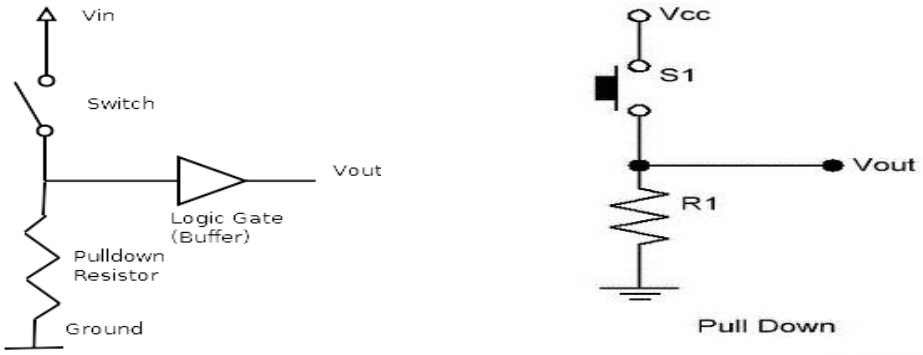
คือการนำตัวต้านทานต่อเข้ากับ Vcc (+5V) เพื่อให้แรงดันอยู่คงที่ ทำให้อยู่ในสถานะ “HIGH” หรือ “1” ตลอดเวลา และเมื่อกดสวิตช์ กระแสไฟฟ้าจะไหลลง Ground ทันที ซึ่งทำให้สถานะเป็นลอจิก “LOW” หรือ “0” และ การทำงานลักษณะนี้ จะเรียกว่า Active Low เพราะว่าจะเขียนโปรแกรมที่ทำงาน เมื่อลอจิกเป็น “LOW” ส่วนใหญ่ เราจะเห็นต่อสวิตช์ นิยมใช้แบบ Pull-up มากกว่า





Pull-down Resistor

คือการต่อ ตัวต้านทาน จาก input ของไมโครคอนโทลเลอร์ เข้ากับ กราว โดยใน Pull-down จะมีลักษณะคล้ายกับ Pull-up Resistor แตกต่าง ตรงที่ สภาวะปกติของ Pull-down จะเป็นลอจิก “LOW” หรือ “0” เมื่อมีการกดปุ่ม กระแสไฟจะไหลเข้าขาอินพุท ทำให้ ลอจิกเป็น “HIGH” หรือ “1” ได้ การทำงานในลักษณะนี้ จะเรียกว่า Active High



คำสั่ง if

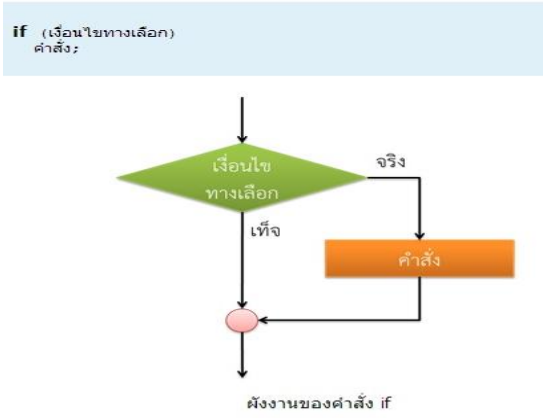
if(เงื่อนไข)

{

//คำสั่ง

}

แปลว่า ถ้าเมื่อไหร่ที่นั้นเงื่อนไขถูกต้องหรือเป็นจริงก็ให้เริ่มทำคำสั่งนั้นได้



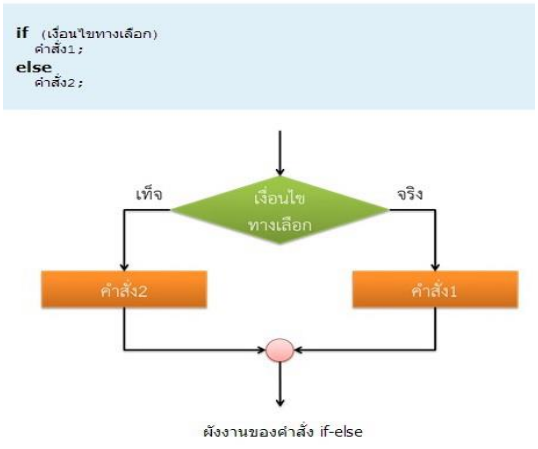
ตัวอย่างที่1

```
void setup()
{
  pinMode(4,INPUT);
  pinMode(3,OUTPUT);
}
void loop()
{
  if(digitalRead(4)==HIGH)  //ถ้ามีการกดสวิตซ์ที่ขา4
          {
                  digitalWrite(3,HIGH);  //LED ที่ขา3จะแดง 5วินาทีแล้วจึงดับ
                  delay(5000);
                  digitalWrite(3,LOW);
  }
  delay(100);   //ในการตรวจสอบว่ามีการกดสวิตซ์หรือไม่ ในแต่ละครั้งชิปจะรอดูอยู่0.1วินาที
                //ถ้าไม่มีการกดชิปก็จะตรวจสอบใหม่อีกรอบไปเรื่อยๆจนกว่าจะพบว่ามีการกด
}
```

คำสั่ง if...else

```
if(เงื่อนไข)
{
        //คำสั่ง 1
}else
{
        //คำสั่ง 2
}
```

แปลว่า ถ้าเมื่อไหร่ที่เงื่อนไขถูกต้องหรือเป็นจริงก็ให้เริ่มทำคำสั่งที่ 1 ได้ แต่ถ้ายังไม่เป็นจริงก็ให้ทำคำสั่งที่ 2 รอไปก่อน



ตัวอย่างที่ 2

```
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin
// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status
void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}
void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);
  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else
 {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

คำสั่ง if...else if...else

คำสั่งนี้สามารถต่อไปได้เรื่อยๆจนกว่าจะพอใจ

```
if(เงื่อนไข 1)
{
        //คำสั่ง 1
}else if(เงื่อนไข 2)
{
        //คำสั่ง 2
}else
{
        //คำสั่ง 3
```

} ไปเรื่อยๆๆ

แปลว่า ถ้าเมื่อไหร่ที่นั้นเงื่อนไข 1ถูกต้องหรือเป็นจริงก็ให้เริ่มทำคำสั่งที่ 1 ได้ แต่ถ้ายังไม่เป็นจริงก็ไปตรวจสอบที่เงื่อนไขที่ 2 ถ้าเงื่อนไขที่ 2 เป็นจริงก็ทำคำสั่งที่ 2 แต่ถ้าไม่เป็นจริงทั้งสองคำสั่งก็จะทำคำสั่งที่ 3 รอไปก่อน

ตัวอย่างที่ 3

```
void setup()
{
  pinMode(4,INPUT);
  pinMode(6,INPUT);
  pinMode(3,OUTPUT);
  pinMode(5,OUTPUT);
}
void loop()
{
  if(digitalRead(4)==HIGH)  //ถ้ามีการกดสวิตซ์ที่ขา4
  {
    digitalWrite(3,HIGH);  //LED ที่ขา 3 จะแดง 5 วินาทีแล้วจึงดับและ LEDที่ขา 5 จะดับ 5 วินาที
    digitalWrite(5,LOW);
    delay(5000);
  }
else if(digitalRead(6)==HIGH) //ถ้ามีการกดสวิตซ์ที่ขา 6 LEDที่ขา 5 จะแดงแต่ที่ขา 3 จะดับ
  {
    digitalWrite(5,HIGH);
    digitalWrite(3,LOW);
    delay(5000);
  }else                    //ถ้าไม่มีการกดสวิตซ์ LED จะกระพริบทั้งคู่
  {
    digitalWrite(3,LOW);
    digitalWrite(5,LOW);
    delay(500);
    digitalWrite(3,HIGH);
    digitalWrite(5,HIGH);
    delay(500);
  }
  delay(100);  //ในการตรวจสอบว่ามีการกดสวิตซ์หรือไม่ ในแต่ละครั้งชิปจะรอดูอยู่0.1วินาที
               //ถ้าไม่มีการกดชิปก็จะตรวจสอบใหม่อีกรอบไปเรื่อยๆจนกว่าจะพบว่ามีการกด
}
```

ลำดับขั้นตอนการปฏิบัติงาน

1. เขียนโปรแกรมรับค่าจากสวิทช์โดยใช้คำสั่งเกี่ยวกับเงื่อนไข if ตาม **ตัวอย่างที่ 1** ในโปรแกรม Arduino IDE และทำการ compiler ให้เรียบร้อย แล้วนำ ไฟล์ที่มีนามสกุล .hex ไฟล์ มา run ในโปรแกรม proteus 8.1 หลังจากที่ได้ต่อวงจรไว้สมบูรณ์แล้วทำการ Simulate ดูผลของการทดลอง
2. ให้ทดลองต่อวงจรจริงที่ เบรดบอร์ด แล้วทำการ copy โปรแกรมลงบนบอร์ด Arduino UNO R3 เพื่อทดลอง สังเกตผลการทดลองบันทึกผล
3. เขียนโปรแกรมรับค่าจากสวิทช์โดยใช้คำสั่งเกี่ยวกับเงื่อนไข if...else ตาม **ตัวอย่างที่ 2** ในโปรแกรม Arduino IDE และทำการ compiler ให้เรียบร้อย แล้วนำ ไฟล์ที่มีนามสกุล .hex ไฟล์ มา run ในโปรแกรม proteus 8.1 หลังจากที่ได้ต่อวงจรไว้สมบูรณ์แล้วทำการ Simulate ดูผลของการทดลอง
4. ให้ทดลองต่อวงจรจริงที่ เบรดบอร์ด แล้วทำการ copy โปรแกรมลงบนบอร์ด Arduino UNO R3 เพื่อทดลอง สังเกตผลการทดลองบันทึกผล
5. เขียนโปรแกรมรับค่าจากสวิทช์โดยใช้คำสั่งเกี่ยวกับเงื่อนไข if ตาม **ตัวอย่างที่ 3** ในโปรแกรม Arduino IDE และทำการ compiler ให้เรียบร้อย แล้วนำ ไฟล์ที่มีนามสกุล .hex ไฟล์ มา run ในโปรแกรม proteus 8.1 หลังจากที่ได้ต่อวงจรไว้สมบูรณ์แล้วทำการ Simulate ดูผลของการทดลอง
6. ให้ทดลองต่อวงจรจริงที่ เบรดบอร์ด แล้วทำการ copy โปรแกรมลงบนบอร์ด Arduino UNO R3 เพื่อทดลอง สังเกตผลการทดลองบันทึกผล

ผลการทดลอง

.....

.....

.....

สรุปผลการทดลอง

.....

.....

.....

การประเมินผล.....

เอกสารอ้างอิง Credit :

รายละเอียดข้อมูลจาก
https://www.facebook.com/ElectronicsTh
http://marcuscode.com/lang/cpp/flow-control
- Pull up, Pull down คืออะไร?
http://www.thitiblog.com/blog/696

ชื่อ-สกุล.....ชั้น.....เลขที่.....