ใบงานที่ 16

เซ็นเซอร์และการใช้งาน

จุดประสงค์การเรียนรู้

1.ศึกษาการแสดงอุณหภูมิและความชื้นออกทางโมดูลจอ LCD

2.ศึกษาการเปิด/ปิดหลอดไฟ LED ด้วยเซ็นเซอร์ LDR

3.ศึกษาการวัดระยะทางและแจ้งเตือนด้วย Ultrasonic Sensor

เครื่องมือและอุปกรณ์การทดลอง

1.เครื่องไมโครคอมพิวเตอร์

2.บอร์ดไมโครคอนโทรลเลอร์ ESP32

3.โปรแกรการทดลอง

4.อุปกรณ์อิเล็กทอรนิกส์สำหรับทดลอง

การทดลองที่ 1 การแสดงอุณหภูมิและความชื้นออกทางโมดูลจอ LCD

อุปกรณ์ในการทดลอง

1.บอรด์ NodeMCU ESP32

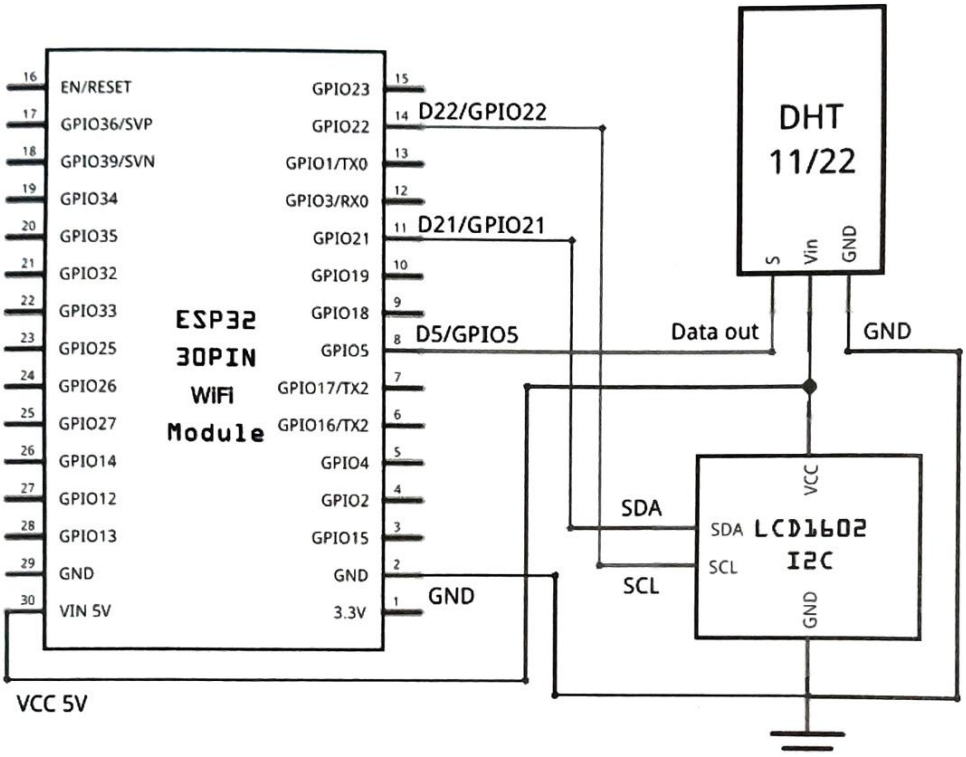
2.โมดูล DHT11 หรือ DHT22

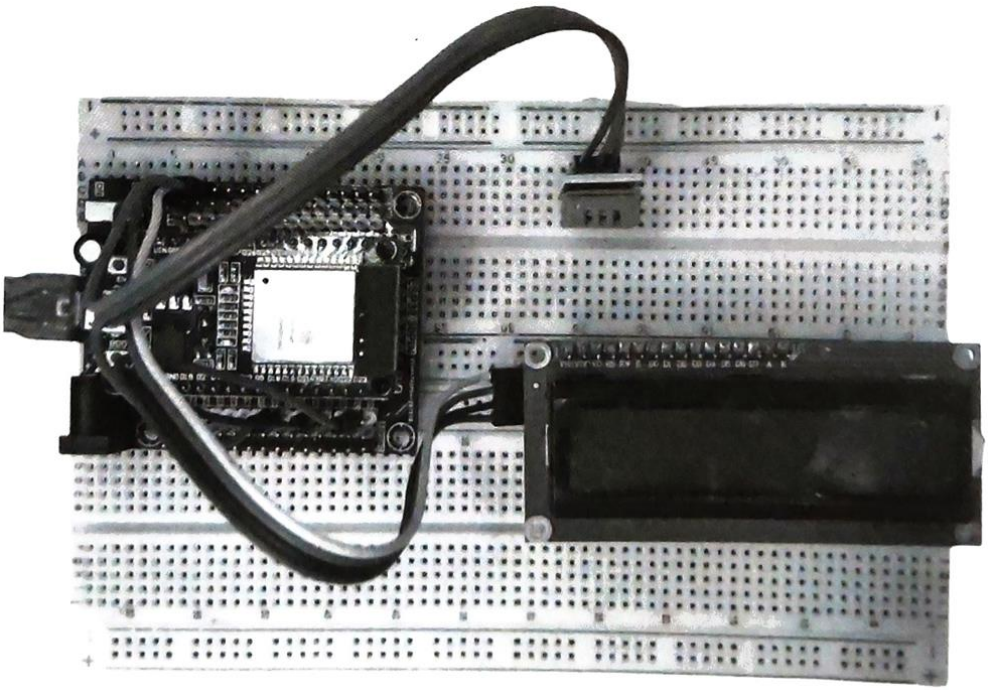
3.โมดูล LCD 16x2

4.แผงต่อวงจร

5.สายไฟต่อวงจร

ประกอบวงจรตามรูป





การติดตั้งไลบรารี่ LiquidCrystal_I2C ของโมดูลจอ LCD

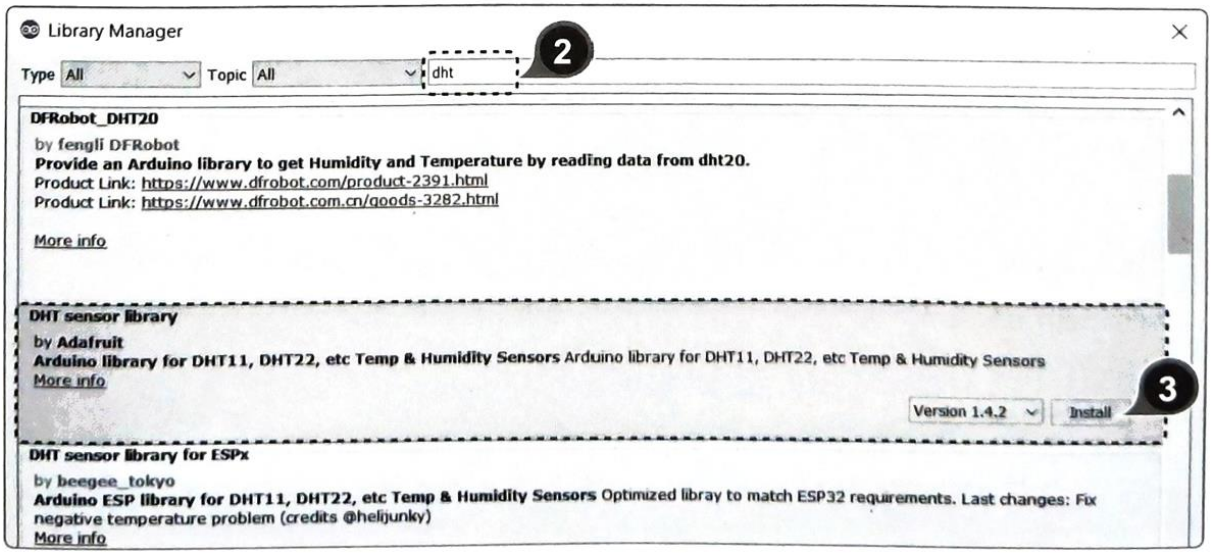
1. ไปที่เว็บไซต์ https://github.com/lucasmaziero/LiquidCrystal_I2C จากนั้นไฟล์ที่ ดาวน์โหลดมาจะเป็นไฟล์ zip เช่น LiquidCrystal_I2C-master.zip เป็นต้น

2. ติดตั้งไลบรารี่ (Library) ให้กับ Arduino IDE โดยใช้ตัว Manager ของ Arduino IDE ด้วยการคลิกที่เมนู Sketch > Include Library » Add .ZIP Library... หรือจะใช้วิธีแบบ Manual โดย แตกซิป แล้วเปลี่ยนชื่อโฟลเดอร์เป็น LiquidCrystal_I2C จากนั้นค่อยก็อปปี้โฟลเดอร์นี้ไปวาง ไว้ที่ Documents\Arduino\libraries ก็ได้

3. สุดท้ายหลังจากติดตั้งไลบรารี่ (Library) เสร็จ ก็ให้เข้าไปตรวจสอบที่ Arduino IDE ดูว่ามีชื่อ ไลบรารี่ที่เราเพิ่งจะติดตั้งถูกเพิ่มเข้าไปรายชื่อไลบรารี่ทั้งหมดแล้วหรือยัง โดยคลิกไปที่เมนู Sketch - Include Library หรือจะคลิกดูที่เมนู File > Examples ก็ได้ ซึ่งในกรณีนี้จะต้อง ปรากฏชื่อไลบรารี่ว่า LiquidCrystal_I2C ถ้าพบแล้วแสดงว่าการติดตั้งเสร็จเรียบร้อย พร้อม ใช้งานแล้ว

หลังจากลงมือเชื่อมต่อสายสัญญาณเรียบร้อยแล้ว ก่อนที่จะลงมือเขียนโค้ดและอัพโหลดโปรแกรม เราจะ ต้องมาติดตั้งไลบรารี่ (Library) ที่จำเป็น ซึ่งในที่นี้ก็คือไลบรารี่ DHT.h ให้กับ Arduino IDE เสียก่อนโดย

1. ที่ Arduino IDE คลิกเมนู Tools » Manage Libraries... หรือกดคีย์ Ctrl + Shift + I
2. ในช่อง Search พิมพ์คำว่า dht แล้วกดคีย์ Enter
3. ที่ไลบรารี่ DHT sensor library by Adafruit ดังรูป คลิกปุ่ม Install เพื่อติดตั้ง



4. จะปรากฏหน้าต่างแจ้งว่าไลบรารี่นี้ จำเป็นต้องใช้งานไลบรารี่ Adafruit Unified Sensor ร่วมด้วย ซึ่งยังไม่ได้ถูกติดตั้ง ในที่นี้ให้เราเลือกว่าจะ ติดตั้งทั้งหมด คลิกปุ่ม Install all

```
#include <Wire.h>                  //เรียกใช้ไลบรารี่ Wire.h
#include <LiquidCrystal_I2C.h>     //เรียกใช้ไลบรารี่ LiquidCrystal_I2C.h
#include <DHT.h>                   //เรียกใช้ไลบรารี่ DHT.h

#define DHTPIN 5                   //กำหนดให้ขา GPIO5/D5 เป็นขาอินพุตที่รับข้อมูลมา จากขา Data Out
                                   ของโมดูล DHT11 มาเก็บไว้ในตัวแปร DHTPIN
#define DHTTYPE DHT11              //กำหนดให้ตัวแปร DHTTYPE ใช้เก็บชนิดของตัวเซ็นเซอร์

LiquidCrystal_I2C lcd(0x27, 16, 2);  //สร้างออบเจ็กต์จากคลาส LiquidCrystal_I2C แล้วนำไป
                                   เก็บไว้ในตัวแปร lcd โดยจะต้องผ่านค่าแอดเดรส 0x27 และขนาด
                                   ของหน้าจอ LCD (แสดงผล 2 แถว แถวละ 16 ตัวอักษร)
DHT dht(DHTPIN, DHTTYPE);          //สร้างออบเจ็กต์จากคลาส DHT แล้วนำไปเก็บไว้ในตัวแปร dht
                                   โดยใช้ค่าที่เก็บไว้ในตัวแปร DHTPIN และ DHTTYPE

void setup() {
  dht.begin();                     //เริ่มต้นใช้งานเซ็นเซอร์ DHT
  lcd.begin();                     //เริ่มต้นการสื่อสารกับโมดูลจอ LCD
  lcd.backlight();
  lcd.print("DHT11 Test!");        //แสดงข้อความใน " " ออกทางหน้าจอ LCD
}

void loop() {
  delay(2000);                     //หน่วงรอเป็นเวลา 2 วินาที
  float t = dht.readTemperature();   //ตัวแปร t ชนิด float ใช้เก็บค่าอุณภูมิที่อ่านได้จาก
                                     ตัวเซ็นเซอร์
  float h = dht.readHumidity();  //ตัวแปร h ชนิด float ใช้เก็บค่าความชื้นที่อ่านได้จากตัวเซ็นเซอร์
```

```
 if (isnan(t) || isnan(h)) {     //ถ้าไม่สามารถอ่านค่าอุณภูมิและความชื้นที่เก็บไว้ในตัวแปร
                                   t และ h ได้ แสดงว่าเกิดข้อผิดพลาด
   lcd.clear();                  //เคลียร์หน้าจอ
   lcd.print("DHT Failed!");    //แสดงข้อความใน " " ออกทางหน้าจอ LCD
 } else {                        //แต่ถ้าไม่
   lcd.setCursor(0, 0);          //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 0 แถว 0
   lcd.print("Temp : ");         //แสดงข้อความใน " " ออกทางหน้าจอ LCD
   lcd.setCursor(7, 0);          //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 7 แถว 0
   lcd.print(t);                 //แสดงค่าตัวเลขของอุณหภูมิที่เก็บไว้ในตัวแปร t
   lcd.setCursor(13, 0);         //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 13 แถว 0
   lcd.print((char)223);         //แสดงสัญลักษณ์องศา (°)
   lcd.setCursor(14, 0);         //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 14 แถว 0
   lcd.print("C");               //แสดงข้อความใน " " ออกทางหน้าจอ LCD
   lcd.setCursor(0, 1);          //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 0 แถว 1
   lcd.print("Humid : ");        //แสดงข้อความใน " " ออกทางหน้าจอ LCD
   lcd.setCursor(8, 1);          //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 8 แถว 1
   lcd.print(h);                 //แสดงค่าตัวเลขของความชื้นที่เก็บไว้ในตัวแปร h
   lcd.setCursor(14, 1);         //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 14 แถว 1
   lcd.print("%");               //แสดงข้อความใน " " ออกทางหน้าจอ LCD
  }
  delay(1000);
}
```

บันทึกผลการทดลอง

.....

.....

.....

.....

การทดลองที่ 2 การเปิด/ปิดหลอดไฟ LED ด้วยเซ็นเซอร์ LDR

อุปกรณ์ในการทดลอง

1.บอรด์ NodeMCU ESP32

2.โมดูล LDR

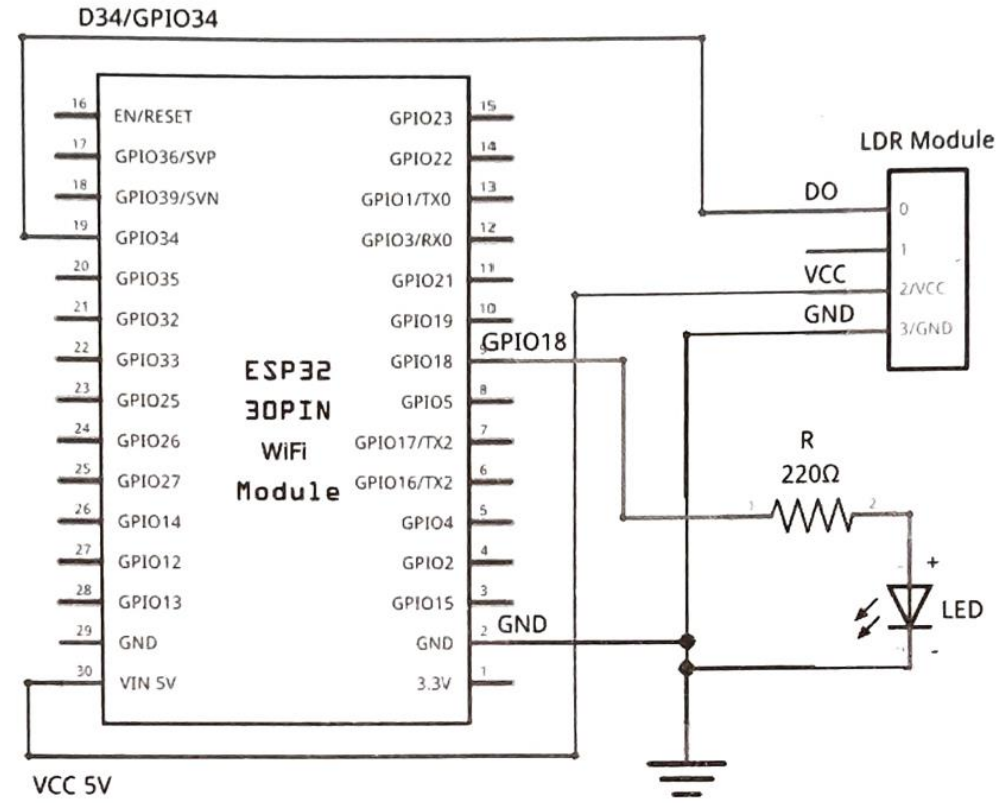
3.หลอดไฟ LED

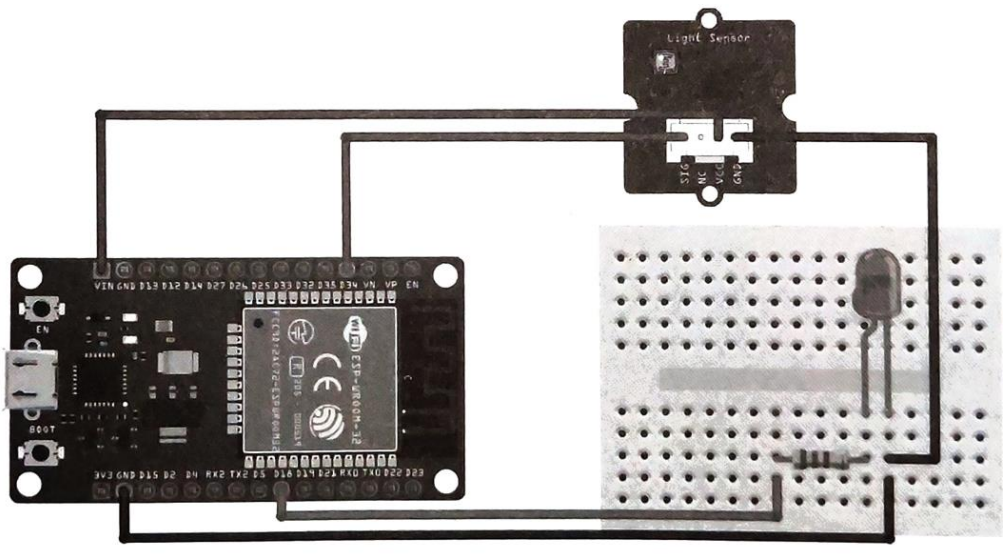
4.ตัวต้านทาน

5.แผงต่อวงจร

6.สายไฟต่อวงจร

ประกอบวงจรตามรูป





ลงมือต่อวงจรตามรูป ให้เชื่อมต่อขา DO ของโมดูลเซ็นเซอร์ LDR เข้ากับขา D34/GPIO34/ADC1_6 ของ บอร์ด ESP32 ส่วนขา Vcc ให้เชื่อมต่อกับขา VIN ของบอร์ดเพื่อจ่ายไฟเลี้ยงขนาด 5V และเชื่อมต่อ GND เข้า ที่ขา GND ของบอร์ด ต่อมาที่ขา Anode (+) หรือขาบวกของหลอดไฟ LED ให้นำตัวต้านทาน 2209 มาต่อไว้ เพื่อป้องกันไม่ให้หลอดไฟ LED เสียหาย และเชื่อมต่อไปยังขาดิจิตอลใดๆ ของบอร์ด (ในที่นี้ใช้ขา D18/GPIO18) เพื่อรับค่าดิจิตอลเอาต์พุตมาแสดงผลที่หลอดไฟ LED ส่วนขา Cathode (-) หรือขาลบของหลอดไฟ LED ให้ เชื่อมต่อเข้ากับขา GND ของบอร์ด

```
const int ledPin = 18;            //ประกาศตัวแปรขาที่ต่อกับหลอดไฟ LED
const int ldrPin = 34;            //ประกาศตัวแปรขาอินพุต Digital

void setup() {

Serial.begin(115200);
pinMode(ledPin, OUTPUT);          //กำหนดให้ตัวแปรหรือขา D18/GPIO18 เป็น Output
pinMode(ldrPin, INPUT);           //กำหนดให้ตัวแปรหรือขา D34/GPIO34 เป็น Input
}
void loop() {
int ldrStatus = digitalRead(ldrPin);  //อ่านค่าอินพุต Digital จาก ldrPin ไปเก็บไว้ที่ตัวแปร
if (ldrStatus == HIGH) {     //ตรวจสอบว่าถ้าค่าในตัวแปรสูงกว่าค่าที่ถูกกำหนดจากตัวต้านทานปรับค่าได้
digitalWrite(ledPin, HIGH);       //ให้ Output ที่ขา GPIO18 เป็น HIGH หลอดไฟ "ติด"
Serial.println("LDR --> DARK, LED --> ON");  //แสดงข้อความใน " " ออกทาง Serial Monitor
} else {                          //แต่ถ้าไม่
digitalWrite(ledPin, LOW);        //ให้ Output ที่ขา GPIO18 เป็น LOW หลอดไฟ "ดับ"
Serial.println("LDR --> Bright, LED --> OFF");  //แสดงข้อความใน " " ออกทาง Serial Monitor
}
delay(1000);
}
```

บันทึกผลการทดลอง

.....

.....

.....

.....

การทดลองที่ 3 การวัดระยะทางและแจ้งเตือนด้วย Ultrasonic Sensor

อุปกรณ์ในการทดลอง

1.บอรด์ NodeMCU ESP32

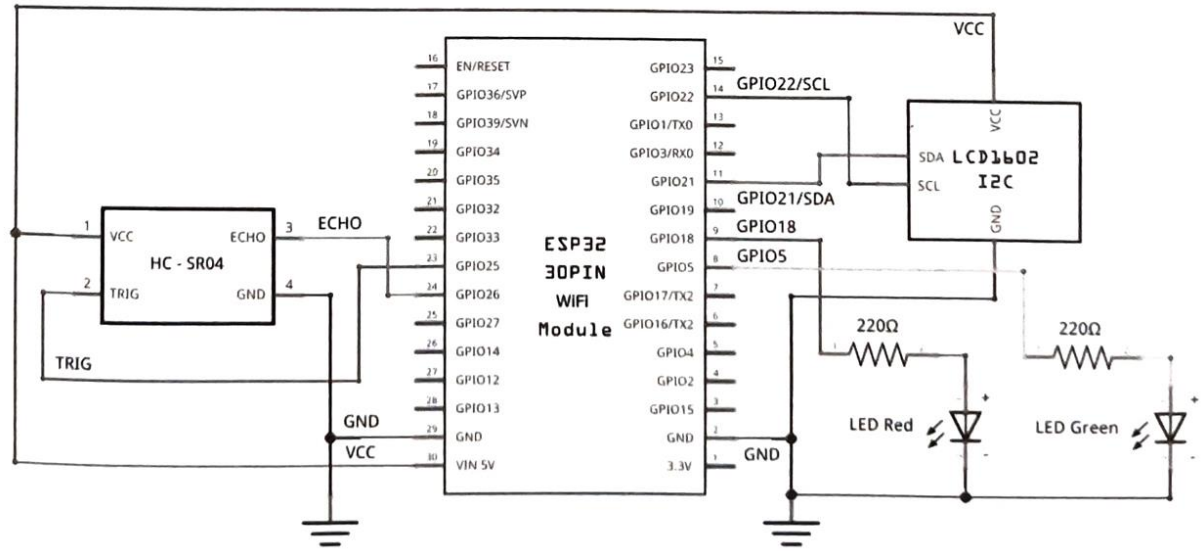
2.โมดูลจอ LCD 16x2

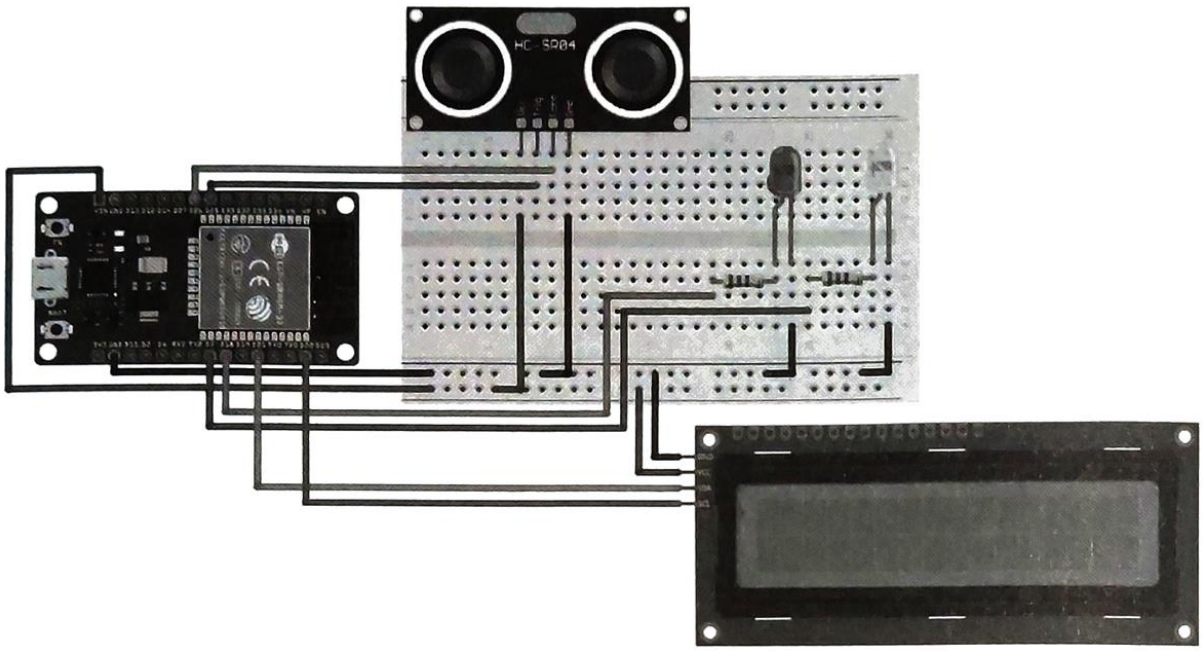
3.โมดูล Ultrasonic Sensor

4.แผงต่อวงจร

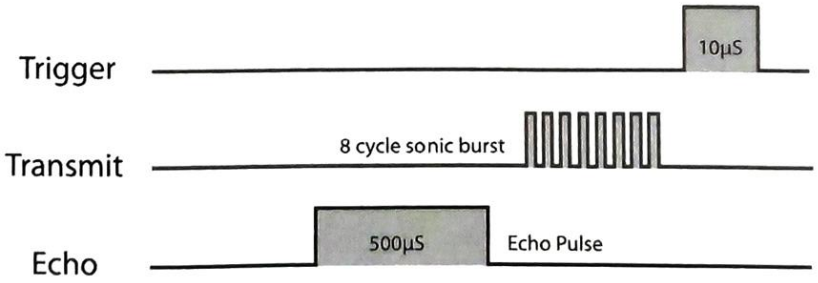
5.สายไฟต่อวงจร

ประกอบวงจรตามรูป





สำหรับโมดูลจอ LCD + I2C Controller ให้ต่อขา SDA และ SCL เข้ากับ D21/GPIO21 และ D22/GPIO22 ที่ซึ่งเป็นขา Default ของบอร์ด แล้วต่อขา VCC 5V จากขา VIN ของบอร์ด กับ GND ให้เรียบร้อย สุดท้ายหลอดไฟ LED สีแดงและเขียวที่เอาไว้แจ้งเตือนเมื่อเข้าใกล้ระยะที่กำหนด ทั้งสองดวงให้ต่อขา Anode (+) เข้ากับ ตัวต้านทาน 2200 แล้วเชื่อมต่อไปยังขา GPIO ใดๆ (ในที่นี้ใช้ขา D18/GPIO18 สำหรับสีแดง และขา D5/GPIOS สำหรับสีเขียว) ส่วนขา Cathode (-) ของทั้งสองดวงก็ต่อกับ GND ให้เรียบร้อย



การทำงาน เมื่อขา Trig ของตัวเซ็นเซอร์ HC-SR04 ได้รับสัญญาณพัลส์ (Trigger) ที่มีความกว้างอย่างน้อยๆ 10us ซึ่งเป็นข้อกำหนดในการทำงานของโมดูล HC-SR04 เพื่อที่มันจะได้ส่งสัญญาณพัลส์ (Sonic Burst) ที่ เป็นคลื่นเสียงความถี่สูงหรือคลื่นอัลตร้าซาวนด์ความถี่ 40 KHz จำนวน 8 ลูก (Cycle) ออกไปได้ ในขณะที่ คลื่นเสียงถูกส่งออกไปนั้น สัญญาณพัลส์ที่ขา Echo ของตัวเซ็นเซอร์จะเปลี่ยนสถานะจาก LOW เป็น HIGH และจะคงสถานะนี้ไว้จนกว่าคลื่นเสียงจะสะท้อนกลับมาที่ตัวเซ็นเซอร์ จึงจะเปลี่ยนสถานะจาก HIGH เป็น LOW นั่นเท่ากับว่าเราก็จะได้สัญญาณพัลส์ที่มีสถานะเป็น HIGH ที่ขา Echo เป็นตัวบอกระยะเวลาที่คลื่นเสียงใช้ใน การเดินทางไปและกลับ เพื่อนำไปใช้คำนวณหาระยะทางหรือระยะห่างระหว่างตัวเซ็นเซอร์กับวัตถุต่อไป

```
#include <Wire.h>                  //เรียกใช้ไลบรารี่ Wire.h
#include <LiquidCrystal_I2C.h>     //เรียกใช้ไลบรารี่ LiquidCrystal_I2C.h
LiquidCrystal_I2C lcd(0x27, 16, 2);  //สร้างออบเจ็กต์จากคลาส LiquidCrystal_I2C แล้วนำไป
                                     เก็บไว้ในตัวแปร lcd โดยจะต้องผ่านค่าแอดเดรสของโมดูล
                                     LCD (ในที่นี้แอดเดรสคือ 0x27) และขนาดของหน้าจอ LCD
                                     (แสดงผล 2 แถว แถวละ 16 ตัวอักษร)
#define TRIG 25                    //กำหนดให้ขา GPIO25/D25 เป็นขา TRIG
#define ECHO 26                    //กำหนดให้ขา GPIO26/D26 เป็นขา ECHO
#define RED_PIN 18                 //กำหนดให้ขา GPIO18/D18 เป็นขา RED_PIN
#define GREEN_PIN 5                //กำหนดให้ขา GPIO5/D5 เป็นขา GREEN_PIN
long duration, distance;           //ประกาศตัวแปรชนิด long ใช้เก็บระยะเวลาที่คลื่นเสียง
                                     เดินทางไปและกลับ (duration) และระยะทางหรือ
                                     ระยะห่างระหว่างตัวเซ็นเซอร์กับวัตถุ (distance)
```

```
void setup() {
  pinMode(TRIG, OUTPUT);          //กำหนดให้ขา TRIG เป็น Output
  pinMode(ECHO, INPUT);           //กำหนดให้ขา ECHO เป็น Input
  pinMode(RED_PIN, OUTPUT);       //กำหนดให้ขา RED_PIN เป็น Output
  pinMode(GREEN_PIN, OUTPUT);     //กำหนดให้ขา GREEN_PIN เป็น Output
  lcd.begin();                    //เริ่มต้นการสื่อสารกับโมดูลจอ LCD
  lcd.backlight();
  lcd.home();                     //เลื่อนเคอร์เซอร์ไปยังจุดเริ่มต้นที่ตำแหน่งคอลัมน์ 0 (ซ้ายสุด)
                                  แถว 0 (บรรทัดบน)
  lcd.print("Check Distance");    //แสดงข้อความใน " " ออกทางหน้าจอ LCD
  lcd.setCursor(0, 1);            //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 0 แถว 1 (บรรทัดล่าง)
  lcd.print("Ready!");            //แสดงข้อความใน " " ออกทางหน้าจอ LCD
  delay(2000);                    //หน่วงรอเป็นเวลา 2 วินาที
}
void loop() {
  digitalWrite(TRIG, LOW);        //กำหนดให้ Output ที่ขา TRIG มีสถานะเป็น LOW
  delayMicroseconds(5);           //หน่วงรอเป็นเวลา 5 ไมโครวินาที (µs) เพื่อเตรียมพร้อม
                                  หรือเพื่อให้แน่ใจว่าจะยังไม่มีคลื่นวิทยุถูกส่งออกไปในตอนนี้
  digitalWrite(TRIG, HIGH);       //กำหนดให้ Output ที่ขา TRIG มีสถานะเป็น HIGH
  delayMicroseconds(10);          //หน่วงรอเป็นเวลา 10 ไมโครวินาที (µs) เพื่อให้เกิด
                                  สัญญาณพัลส์ที่เป็น HIGH กว้างอย่างน้อย 10 µs
  digitalWrite(TRIG, LOW);        //เมื่อครบ 10 µs จึงกำหนดให้ Output ที่ขา TRIG มีสถานะเป็น LOW
                                  จากนั้นเซ็นเซอร์จะส่งคลื่นเสียงความถี่สูงจำนวน 8 Cycle ออกไป
  duration = pulseIn(ECHO, HIGH);  //ด้วยคำสั่งนี้ บอร์ดจะเริ่มนับเวลาตั้งแต่คลื่นเสียงเริ่มเดิน
                                  ทางไป หรือก็คือเมื่อขา ECHO เริ่มเปลี่ยนสถานะจาก LOW และจะนับ
                                  ไปเรื่อยๆ (คงสถานะ HIGH ไว้) จนกว่าเป็น HIGH คลื่นเสียงจะสะท้อน
                                  และเดินทางกลับมา ซึ่งจะทำให้ขา ECHO เปลี่ยนสถานะจาก HIGH เป็น
                                  LOW ค่าเวลาที่ได้ในช่วง HIGH จะถูกเก็บไว้ในตัวแปร duration
  distance = duration * 0.034 / 2;  //นำค่าในตัวแปร duration มาคำนวณหาค่าระยะทางหรือ
                                  ระยะห่างระหว่างตัวเซ็นเซอร์กับวัตถุ (distance)
  lcd.setCursor(0, 0);            //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 0 แถว 0 (บรรทัดบน)
  lcd.print("Distance: ");        //แสดงข้อความใน " " ออกทางหน้าจอ LCD
  lcd.print(distance);            //แสดงค่าตัวเลขที่เก็บไว้ในตัวแปร distance
  lcd.print(" cm. ");             //แสดงข้อความใน " " ออกทางหน้าจอ LCD
```

```
  if (distance >= 15) {          //ถ้าระยะทางที่คำนวณได้มากกว่าหรือเท่ากับ 15 เซนติเมตร
    digitalWrite(GREEN_PIN, HIGH);    //กำหนดให้ขา GREEN_PIN เป็น HIGH หลอดไฟสีเขียวติด
    digitalWrite(RED_PIN, LOW); //กำหนดให้ขา RED_PIN เป็น LOW หลอดไฟสีแดงดับ
    lcd.setCursor(0, 1);         //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 0 แถว 1 (บรรทัดล่าง)
    lcd.print("Safe!");          //แสดงข้อความใน " " ออกทางหน้าจอ LCD
  } else {                       //แต่ถ้าไม่
    digitalWrite(GREEN_PIN, LOW);     //กำหนดให้ขา GREEN_PIN เป็น LOW หลอดไฟสีเขียวดับ
    digitalWrite(RED_PIN, HIGH);      //กำหนดให้ขา RED_PIN เป็น HIGH หลอดไฟสีแดงติด
    lcd.setCursor(0, 1);              //เลื่อนเคอร์เซอร์ไปที่ตำแหน่งคอลัมน์ 0 แถว 1 (บรรทัดล่าง)
    lcd.print("STOP! ");              //แสดงข้อความใน " " ออกทางหน้าจอ LCD
  }
  delay(500);
}
```

บันทึกผลการทดลอง

.....

.....

.....

.....

สรุปผลการทดลอง

.....

.....

.....

.....