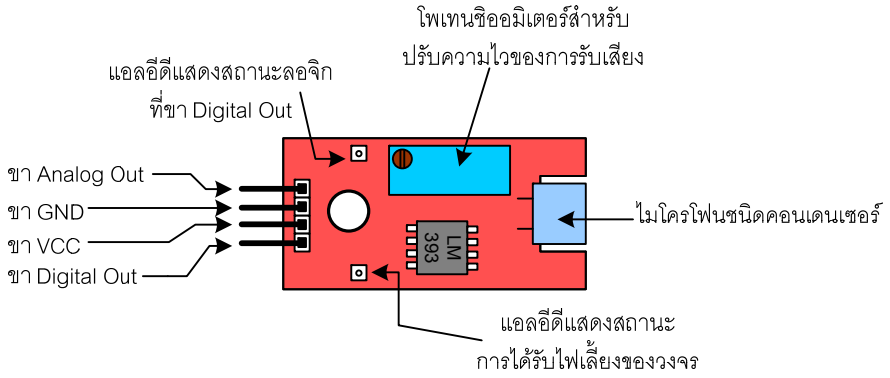
LAB 12

มอดูลรับเสียง และมอดูล DHT-11

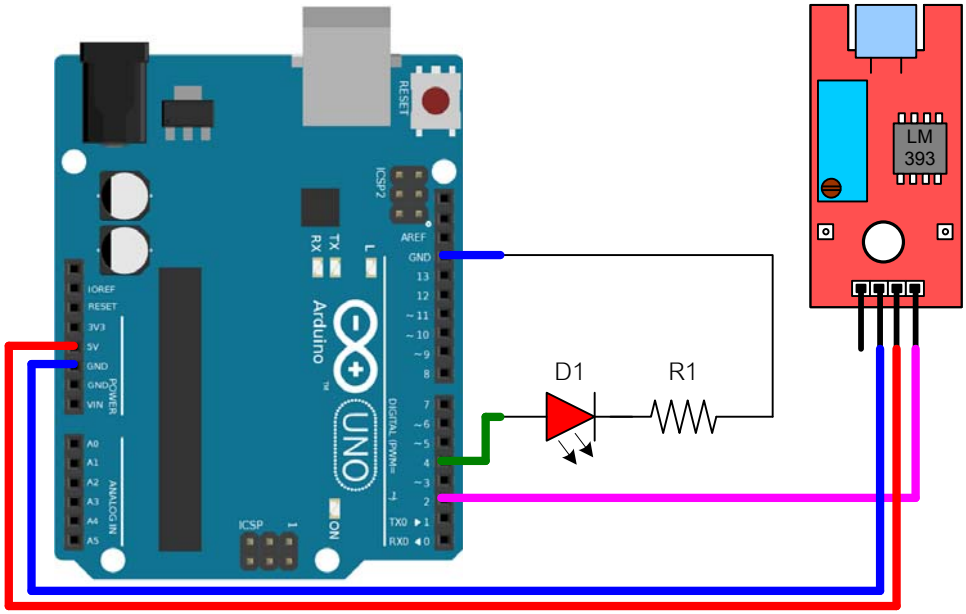
12.1 มอดูลรับเสียง

ในการรับเสียงเข้ามาประมวลผลด้วยบอร์ด Arduino นั้น จะต้องมีมอดูลรับเสียงซึ่งมีไมโครโฟนทำหน้าที่แปลงเสียงให้กลายเป็นสัญญาณไฟฟ้า หากเสียงมีระดับความดังเปลี่ยนไปก็จะทำให้ระดับค่าแรงดันไฟฟ้าที่ได้มีการเปลี่ยนแปลงตามไปด้วย โดยบนแผงวงจรของมอดูลรับเสียงจะมีวงจรเปรียบเทียบแรงดันอยู่เพื่อตรวจสอบระดับแรงดันว่ามีค่าสูงถึงค่าที่กำหนดหรือไม่ หากค่าระดับแรงดันสูงถึงค่า ๆ หนึ่ง ซึ่งเรียกว่าค่าระดับแรงดัน Threshold จะทำให้วงจรเปรียบเทียบแรงดันส่งค่าลอจิก LOW ออกมาที่ขา digital Out ของบอร์ด แต่หากค่าระดับแรงดันต่ำกว่าค่า Threshold จะทำให้ได้ค่าลอจิก HIGH ออกมาที่ขา digital Out ของบอร์ด

รูปที่ 12.1 แสดงมอดูลรับเสียงที่ใช้ในการทดลอง โดยจะมีไมโครโฟนชนิดคอนเดนเซอร์ทำหน้าที่รับเสียง และมีไอซี LM393 ทำหน้าที่เป็นวงจรเปรียบเทียบแรงดัน มอดูลรับเสียงตัวนี้สามารถให้สัญญาณขาออกได้ทั้งแบบแอนะล็อกและแบบดิจิทัล แต่ในใบงานนี้จะใช้เฉพาะสัญญาณดิจิทัลเท่านั้นในการติดต่อกับบอร์ด Arduino ผู้ใช้สามารถปรับความไวของการรับเสียงของไมโครโฟนได้ด้วยการใช้ไขควงหมุนสกรูที่อยู่บนตัวโพเทนชิออมิเตอร์ ซึ่งจะเป็นการปรับค่าระดับแรงดัน Threshold ของวงจรผ่านการเปลี่ยนค่าความต้านทานของโพเทนชิออมิเตอร์



รูปที่ 12.1 มอดูลรับเสียงที่ใช้ในการทดลอง



รูปที่ 12.2 การเชื่อมต่อมอดูลรับเสียงกับบอร์ด Arduino

```
1  int soundPin = 2;
2  int ledPin = 4;
3  unsigned long previous_time=0;
4
5  bool check_clap()
6  {
7    static unsigned long current_time;
8    char sw;
9    sw = digitalRead(soundPin);
10   if (sw)
11   {
12     return false;
13   }
14   else
15   {
16     current_time = millis();
17     if ((current_time-previous_time)>25)
18     {
19       previous_time = millis();
20       return true;
21     }
22     else
23     {
24       previous_time = millis();
25       return false;
26     }
27   }
28 }
29
30 int val;
31 int state;
32 int num_clap;
33
34 void setup()
35 {
36   pinMode(ledPin, OUTPUT);
37   pinMode(soundPin, INPUT);
38   Serial.begin (9600);
39   state =0;
40   num_clap = 0;
41   Serial.println("num_clap = 0");
42 }
```

รูปที่ 12.3 ตัวอย่างการเขียนโปรแกรมเพื่ออ่านค่าจากมอดูลรับเสียง

```
43  void loop()
44  {
45    val = check_clap();
46    if (val==true)
47    {
48      Serial.print("Clap deteected!!! #");
49      Serial.print(num_clap);
50      num_clap++;
51      if (state ==1)
52      {
53        state =0;
54        Serial.println("  LED is turned off");
55      }
56      else
57      {
58        state =1;
59        Serial.println("  LED is turned on");
60      }
61    }
62    if (state ==1)
63      digitalWrite(ledPin, HIGH);
64    else
65      digitalWrite(ledPin, LOW);
66  }
```

รูปที่ 12.3 ตัวอย่างการเขียนโปรแกรมเพื่ออ่านค่าจากมอดูลรับเสียง (ต่อ)

12.2 มอดูลรีเลย์

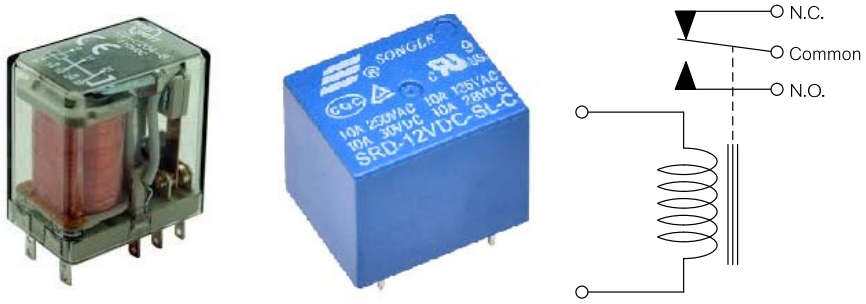
รีเลย์ คือ สวิตช์ไฟฟ้าชนิดหนึ่งซึ่งใช้สัญญาณควบคุมซึ่งมีกำลังต่ำมาใช้ในการตัดต่อวงจรไฟฟ้าซึ่งมีกำลัง สูงกว่าสัญญาณควบคุมจะถูกป้อนให้กับขดลวด ส่งผลให้เกิดสนามแม่เหล็กไปดูดเหล็กอ่อนซึ่งมักถูกยึดติดกับสวิตช์ ส่งผลให้เกิดการแยกจากกันหรือการเชื่อมติดกันของหน้าสัมผัสของสวิตช์ ทำให้เราสามารถควบคุมการเชื่อมต่อของหน้าสัมผัสของสวิตช์ซึ่งทนกำลังสูงได้ด้วยการควบคุมสัญญาณที่ป้อนให้กับขดลวด รูปที่ 12.4 แสดงรีเลย์และสัญลักษณ์ทางไฟฟ้าที่ใช้ในวงจร

ที่หน้าสัมผัสของสวิตช์ในรีเลย์ทั่วไปจะมีขาเชื่อมต่อมาใช้งานอย่างน้อย 3 ขา ได้แก่

- ขา N.O. หรือ Normally Open หรือ ขาปรกติเปิด
- ขา N.C. หรือ Normally Closed หรือขาปรกติปิด
- ขา Common หรือขาร่วม

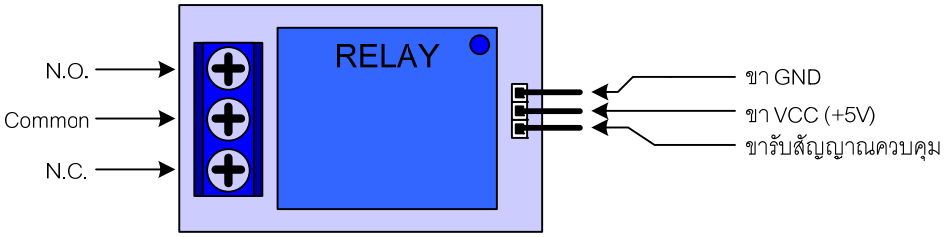
เมื่อมีสัญญาณควบคุมป้อนเข้ามาที่ขดลวด จะส่งผลให้เกิดสนามแม่เหล็กดูดเหล็กอ่อนซึ่งเชื่อมกับหน้าสัมผัส ส่งผลให้หน้าสัมผัสของขา Common ถูกดึงมาให้เชื่อมต่อกับขา N.O. ส่งผลให้กระแสไฟฟ้าไหลจากขา Common ไปยังขา N.O. ได้ แต่หากสัญญาณควบคุมถูกตัดขาดลงจะส่งผลให้หน้าสัมผัสของขา Common เลื่อนไปต่อกับขา N.C. ตามเดิม

นอกจากรีเลย์ชนิดกลไกแล้ว ยังมีรีเลย์อีกชนิดหนึ่ง คือ โซลิดสเตทรีเลย์ (Solid-state Relay) ซึ่งทำงานด้วยวงจรอิเล็กทรอนิกส์ล้วนโดยไม่มีการใช้ขดลวดเพื่อสร้างสนามแม่เหล็กบังคับการเคลื่อนไหวของหน้าสัมผัสแต่อย่างใด อย่างไรก็ดี ในการทดลองนี้จะเน้นเพียงการใช้งานรีเลย์ประเภทกลไกเท่านั้น



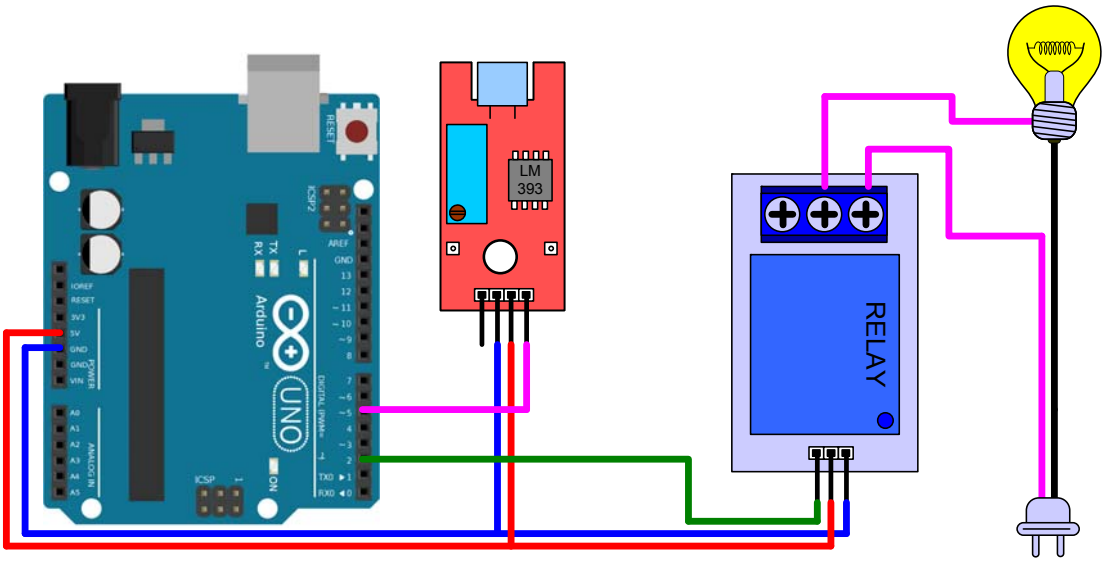
รูปที่ 12.4 รีเลย์ชนิดกลไก

รีเลย์แต่ละตัวจะมีค่าแรงดันใช้งานของขดลวดอยู่ค่าหนึ่ง ซึ่งถูกกำหนดโดยผู้ผลิต รีเลย์ที่ใช้ในงานอิเล็กทรอนิกส์ส่วนใหญ่มักใช้แรงดันไฟฟ้ากระแสตรงสำหรับป้อนขดลวด รูปที่ 12.5 แสดงมอดูลรีเลย์ที่ใช้ในการทดลองซึ่งต้องการค่าแรงดัน 5 โวลต์ในการทำงาน โดยมีขาสัญญาณควบคุมซึ่งรับค่าระดับลอจิก HIGH เพื่อสั่งให้เกิดกระแสไฟฟ้าไหลผ่านขดลวด และรับค่าลอจิก LOW เพื่อสั่งให้รีเลย์หยุดทำงาน รีเลย์ในมอดูลนี้มีหน้าสัมผัสของสวิตช์แค่ชุดเดียว โดยมีขาสำหรับเชื่อมต่อกับโหลด 3 ขา ได้แก่ ขา N.O. ขา N.C. และขา Common



รูปที่ 12.5 มอดูลรีเลย์ที่ใช้ในการทดลอง

สำหรับมอดูลรีเลย์ในรูปที่ 12.5 นั้น เราสามารถควบคุมได้โดยการเชื่อมต่อขาสัญญาณควบคุมของมอดูลเข้ากับขาสัญญาณดิจิทัลใด ๆ ก็ได้ของบอร์ด Arduino (D2-D13) จากรูปที่ 12.5b เราสามารถควบคุมการติดดับของหลอดไฟ 220 โวลต์ได้ด้วยการต่อกับรีเลย์ และสามารถใช้เสียงควบคุมการเปิดปิดหลอดไฟได้



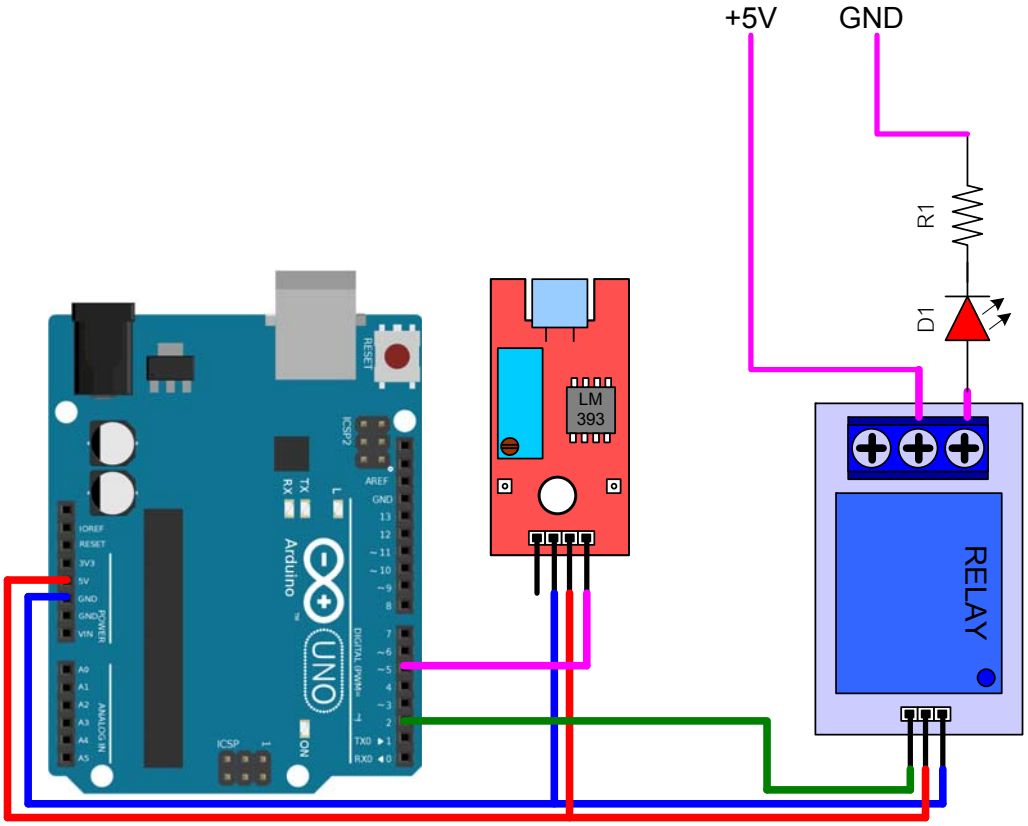
รูปที่ 12.5b ตัวอย่างวงจรควบคุมการเปิดปิดหลอดไฟ 220 โวลต์ผ่านรีเลย์ด้วยเสียง

การทดลองที่ 12.1

จงต่อมอดูลรับเสียงและหลอดแอลอีดีเข้ากับบอร์ด Arduino ดังรูปที่ 12.2 จากนั้นนำโค้ดโปรแกรมในรูปที่ 12.3 มาเขียนบน Arduino IDE จากนั้นคอมไพล์และอัพโหลดโปรแกรมลงสู่บอร์ด Arduino และเปิดโปรแกรม Serial Monitor ให้ใช้วิธีปรบมือเพื่อให้เกิดเสียงเป็นจังหวะ จากนั้นสังเกตการทำงานของโปรแกรมผ่านหน้าจอโปรแกรม Serial Monitor และการแสดงผลที่หลอดแอลอีดี

Checkpoint 12.1

จงต่อมอดูลรับเสียงและรีเลย์และหลอดไฟเข้ากับบอร์ด Arduino ดังรูปที่ 12.6 จากนั้นให้เขียนโปรแกรมควบคุมการติดดับของหลอดไฟโดยใช้การปรบมือ โดยเริ่มต้นให้หลอดไฟดับ หากมีการปรบมือก็จะส่งผลให้หลอดไฟติด หลังจากนั้นหากมีการปรบมืออีกครั้งก็จะทำให้หลอดไฟดับ และเป็นเช่นนี้สลับกันไปเรื่อย ๆ เมื่อวงจรทำงานได้ตามที่กำหนดแล้ว ให้ยกมือเรียกเจ้าหน้าที่หรือ TA เพื่อตรวจ Checkpoint



รูปที่ 12.6 วงจรสำหรับ Checkpoint 12.2

Checkpoint 12.2

จงต่อมอดูลรับเสียงและรีเลย์และหลอดไฟเข้ากับบอร์ด Arduino ดังรูปที่ 12.6 จากนั้นให้เขียนโปรแกรมควบคุมการติดดับของหลอดไฟโดยใช้การปรบมือ โดยเริ่มต้นให้หลอดไฟดับ หากมีการปรบมือ 2 ครั้งก็จะส่งผลให้หลอดไฟติด หลังจากนั้นหากมีการปรบมือ 2 ครั้งก็จะทำให้หลอดไฟดับ และเป็นเช่นนี้สลับกันไปเรื่อย ๆ เมื่อวงจรทำงานได้ตามที่กำหนดแล้ว ให้ยกมือเรียกเจ้าหน้าที่หรือ TA เพื่อตรวจ Checkpoint

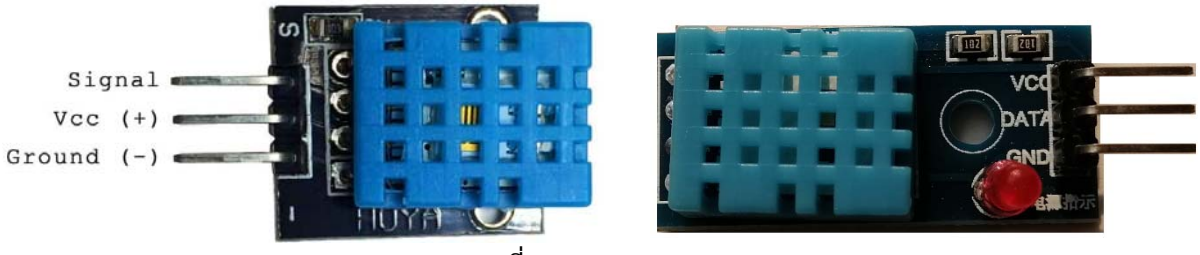
Checkpoint 12.2B (Bonus)

จงนำวงจรในรูปที่ 12.6 มาดัดแปลงใหม่ โดยเปลี่ยนวิธีการติดต่อกับมอดูลเสียงใหม่ โดยให้มอดูลเสียงติดต่อกับซีพียูด้วยกลไกอินเตอร์รัพต์ และเขียนโปรแกรมควบคุมการติดดับของหลอดไฟโดยใช้การปรบมือ โดยเริ่มต้นให้หลอดไฟดับ หากมี

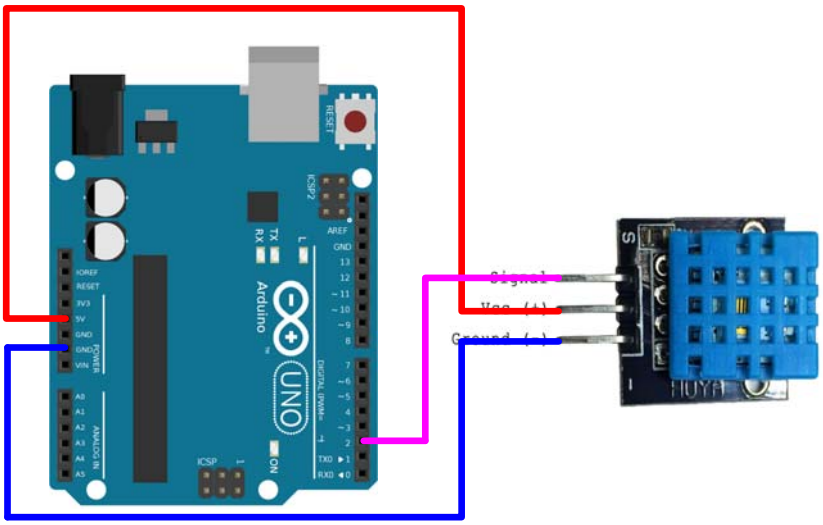
การปรบมือ 2 ครั้งก็จะส่งผลให้หลอดไฟติด หลังจากนั้นหากมีการปรบมือ 2 ครั้งก็จะทำให้หลอดไฟดับ และเป็นเช่นนี้สลับกันไปเรื่อย ๆ สำหรับ Checkpoint 12.2B นี้ นักศึกษาคนใดจะไม่ทำก็ได้ แต่หากทำจะมีคะแนนพิเศษ (Bonus) ให้

12.3 เซนเซอร์ DHT11

DHT11 เป็น มอดูลเซนเซอร์ขนาดเล็ก สามารถวัดค่าความชื้นและอุณหภูมิได้ค่อนข้างแม่นยำพอสมควร เนื่องจากมีราคาถูกและใช้งานง่าย จึงทำให้มอดูลนี้เป็นที่นิยมใช้ในหมู่นักพัฒนา คุณสมบัติของมอดูลนี้ คือ จะให้ค่าเอาต์พุตออกมาเป็นค่าความชื้นสัมพัทธ์ (relative humidity, RH) และค่าอุณหภูมิออกมาในหน่วยองศาเซลเซียสหรือฟาเรนไฮต์ โดยสามารถวัดค่าความชื้นสัมพัทธ์ได้ในย่าน 20-90 เปอร์เซนต์ และค่าย่านอุณหภูมิที่วัดได้อยู่ในช่วง 0-60 องศาเซลเซียส มีขาใช้งานแสดงดังรูปที่ 12.7 การส่งข้อมูลระหว่างบอร์ด Arduino และมอดูล DHT11 ทำได้โดยใช้สายสัญญาณเพียงหนึ่งเส้น อย่างไรก็ดีเนื่องจากมอดูล DHT11 จะต้องได้รับไฟเลี้ยงวงจรภายในด้วย ดังนั้นในการต่อมอดูล DHT11 กับบอร์ด Arduino จะต้องมีการเชื่อมต่อระหว่างขา +5V และ GND ณ ฝั่งบอร์ด Arduinoเข้ามายังขา VCC และ Ground ของมูดูล DHT11 ด้วย ดังรูปที่ 12.8 ส่วนการเชื่อมต่อขาสัญญาณ (ขา signal) ของมอดูลนี้สามารถนำมาเชื่อมต่อกับขาสัญญาณดิจิทัล D2-D13 ขาใดก็ได้ของบอร์ด Arduino ดังนั้นผู้ใช้งานจึงต้องกำหนดขาใช้งานในตัวโปรแกรมให้ตรงกับที่ต่อใช้งานจริงกับบอร์ด Arduino ด้วย



รูปที่ 12.7 มอดูล DHT11



รูปที่ 12.8 การเชื่อมต่อมอดูล DHT11 กับบอร์ด Arduino

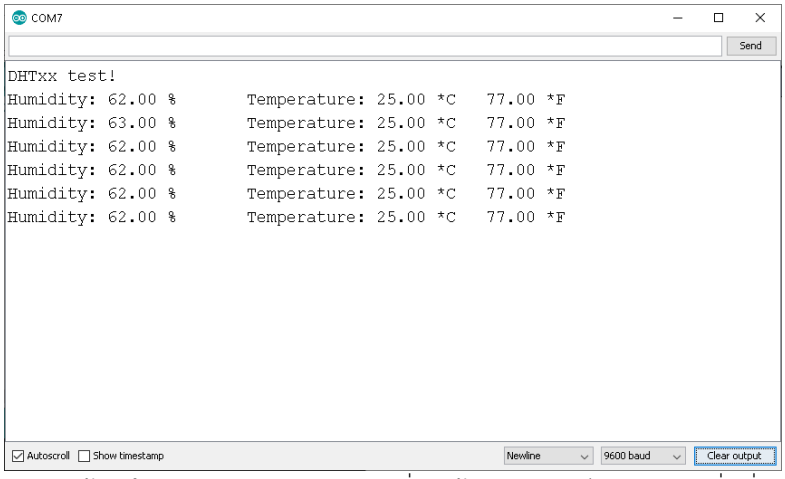
ในการเขียนโปรแกรมเพื่ออ่านค่าจากมอดูล DHT11 มาใช้งานจะต้องมีการเรียกใช้คลังโปรแกรม DHT.h เสียก่อน ดังแสดงในโค้ดโปรแกรมในรูปที่ 12.9 ซึ่งจะเห็นว่ามีการเรียกใช้คลังโปรแกรมในบรรทัดที่ 1 และมีการสั่งให้มีการสร้างออบเจ็กต์สำหรับใช้งานมอดูล DHT11 กับขา 2 ของบอร์ด Arduino ดังโค้ดโปรแกรมในบรรทัดที่ 7 นอกจากนี้ยังต้องมีการสั่งให้ DHT11 เริ่มทำงานในส่วนของฟังก์ชัน setup ดังโค้ดโปรแกรมในบรรทัดที่ 12

ในการอ่านค่าความชื้นจากมอดูล DHT11 จะทำโดยการเรียกใช้ฟังก์ชัน readHumidity ส่วนการอ่านค่าอุณหภูมิจะใช้ฟังก์ชัน readTemperature จากโค้ดตัวอย่างจะมีการนำค่าอุณหภูมิและความชื้นส่งมาแสดงผลยังโปรแกรม Serial Monitor ซึ่งทำงานอยู่บนเครื่องพีซี ดังรูปที่ 12.10

ในการแสดงผลออกยังโปรแกรม Serial Monitor นั้น จะต้องมีการสั่งให้มีการติดต่อผ่านอนุกรมของบอร์ด Arduino ด้วยการสั่ง **Serial.begin**(*n*) ดังโค้ดในบรรทัดที่ 10 ของรูปที่ 12.9 โดยค่า *n* นี้ เป็นค่าความเร็วในการรับส่งข้อมูลของพอร์ตอนุกรมซึ่งมีหน่วยเป็นบิตต่อวินาที ซึ่งจากโค้ดตัวอย่างได้ตั้งค่าความเร็วไว้ที่ 9600 บิตต่อวินาที ส่วนการส่งข้อมูลจะใช้เมธอด Serial.println

```
1  #include "DHT.h"          //เรียกใช้คลังโปรแกรม DHT.h
2  #define DHTPIN 2          //กรณีนี้มีการต่อขาสัญญาณ (signal) ของเซนเซอร์ DHT11 เข้ากับขา 2 ดังรูป 4.2
3  #define DHTTYPE DHT11     // DHT 11
4  //#define DHTTYPE DHT22    // DHT 22  (AM2302)
5  //#define DHTTYPE DHT21    // DHT 21 (AM2301)
6
7  DHT dht(DHTPIN, DHTTYPE) ;  //สั่งให้ใช้มอดูล DHT11 กับขา 2 ของบอร์ด Arduino
8
9  void setup() {
10   Serial.begin(9600);          //ตั้งค่าการสื่อสารอนุกรมกับเครื่องคอมพิวเตอร์ให้มีความเร็ว 9600 bps
11   Serial.println("DHTxx test!");
12   dht.begin();                 //สั่งให้ DHT11 ทำงาน
13 }
14
15 void loop() {
16   delay(2000);
17   float h = dht.readHumidity();          //อ่านค่าความชื้นสัมพัทธ์ใส่ตัวแปร h
18   float t = dht.readTemperature();       //อ่านค่าอุณหภูมิในหน่วยองศาเซลเซียสใส่ตัวแปร t
19   float f = dht.readTemperature(true);   //อ่านค่าอุณหภูมิในหน่วยองศาฟาเรนไฮต์ใส่ตัวแปร f
20
21   //หากค่าใน h, t, f ไม่ใช่ค่าตัวเลขที่ถูกต้องให้แสดงผลให้แสดง Error message ออกทาง Serial Monitor
22   if (isnan(h) || isnan(t) || isnan(f)) {
23     Serial.println("Failed to read from DHT sensor!");
24     return;
25   }
26   Serial.print("Humidity: ");
27   Serial.print(h);
28   Serial.print(" %\t");
29   Serial.print("Temperature: ");
30   Serial.print(t);
31   Serial.print(" *C ");
32   Serial.print(f);
33   Serial.println(" *F\t");
34 }
```

รูปที่ 12.9 ตัวอย่างการเขียนโปรแกรมเพื่ออ่านค่าจากมอดูล DHT11



รูปที่ 12.10 ตัวอย่างหน้าจอโปรแกรม Serial Monitor ซึ่งรับข้อมูลจากบอร์ด Arduino ซึ่งเชื่อมต่อกับ DHT11

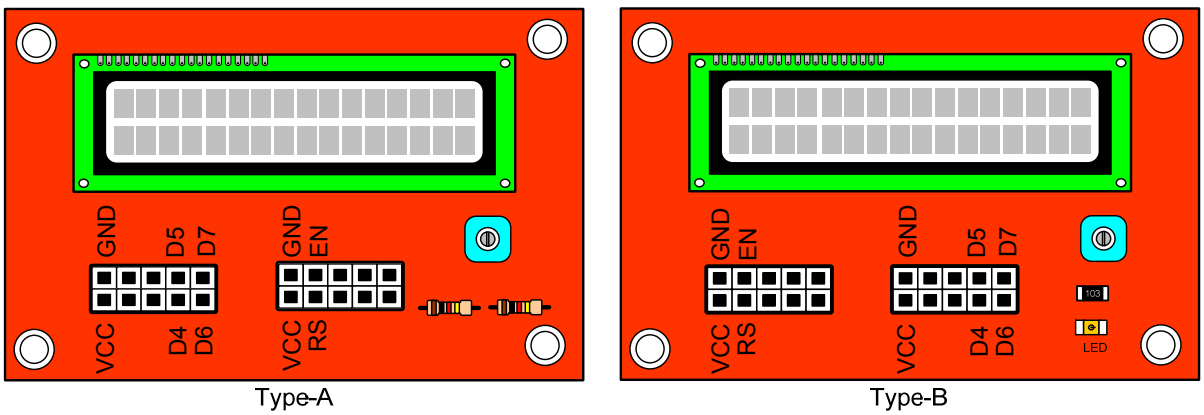
การทดลองที่ 12.2

จงต่อมอดูล DHT11 กับบอร์ด Arduino ดังรูปที่ 12.8 จากนั้นนำโค้ดโปรแกรมในรูปที่ 12.9 มาเขียนบน Arduino IDE จากนั้นคอมไพล์และอัพโหลดโปรแกรมลงสู่บอร์ด Arduino และเปิดโปรแกรม Serial Monitor สังเกตการทำงานของโปรแกรมผ่านหน้าจอโปรแกรม Serial Monitor

12.4 มอดูลแอลซีดี

จอภาพผลึกเหลว (Liquid Crystal Display: LCD) เป็นจอแสดงผลที่มีความบางและกินพลังงานต่ำ ทำงานโดยอาศัยหลักการใช้ผลึกเหลวภายในซึ่งถูกควบคุมโดยใช้ขั้วไฟฟ้าเพื่อควบคุมการส่งผ่านของแสงออกทางด้านหน้าของจอภาพ โดยมีหลอดกำเนิดแสงอยู่ด้านหลังของจอ จอชนิดนี้จะต้องมีหน่วยประมวลผลขนาดเล็ก (Controller) สำหรับควบคุมการทำงานตลอดจนการรับข้อมูลเข้าไปเก็บไว้ชั่วคราวในหน่วยความจำของตัวจอภาพเอง ในการใช้งาน ผู้เขียนโปรแกรมจะต้องส่งคำสั่งและข้อมูลจากบอร์ด Arduino ให้กับตัวควบคุมแอลซีดีซึ่งติดตั้งอยู่ภายในจอแอลซีดีเพื่อให้ได้การแสดงผลบนหน้าจอตามที่ต้องการ

รูปที่ 12.11 แสดงมอดูลแอลซีดีที่ใช้ในการทดลอง ซึ่งเป็นมอดูลแอลซีดีชนิดตัวอักษาแบบ 16x2 ซึ่งหมายความว่าจอแอลซีดีชนิดนี้สามารถแสดงผลได้ 2 บรรทัด บรรทัดละ 16 ตัวอักษร จะเห็นว่าบนแผงวงจรมีขาเชื่อมต่อที่จำเป็นต้องใช้จำนวน 8 เส้น ซึ่งได้แก่ ขา D4-D7 (รวม 4 เส้น) ใช้ในการรับข้อมูลจากตัวประมวลผล ขา VCC และ GND ใช้ในการรับค่าแรงดันไฟฟ้ากระแสตรงจากแหล่งจ่าย ขา RS ใช้ในการสั่งมอดูลแอลซีดีว่าจะให้รับคำสั่งหรือข้อมูลจากตัวประมวลผล ขา Enable ใช้ในการเปิดทางให้มอดูลแอลซีดีทำงาน



รูปที่ 12.11 มอดูลแอลซีดีที่ใช้ในการทดลอง

ตารางที่ 4.1 แสดงฟังก์ชันที่นิยมใช้งานของคลังโปรแกรม LiquidCrystal.h ซึ่ง

ชื่อฟังก์ชัน	คำอธิบาย
LiquidCrystal()	สร้างตัวแปรชนิด LiquidCrystal ขึ้นมาในระบบ
begin()	เริ่มต้นการทำงานของหน้าจอแอลซีดีพร้อมระบุขนาดความกว้างและความสูงของหน้าจอ
clear()	ลบล้างหน้าจอแอลซีพีและเลื่อนเคอร์เซอร์ไปยังมุมบนซ้ายของหน้าจอ
home()	เลื่อนเคอร์เซอร์ไปยังมุมบนซ้ายของหน้าจอ
print()	พิมพ์ข้อความออกสู่จอแอลซีดี ณ จุดที่เคอร์เซอร์กำลังชี้อยู่
setCursor()	เลื่อนเคอร์เซอร์ของหน้าจอให้ตรงกับตำแหน่งที่ระบุ
noDisplay()	สั่งให้ลบตัวอักษรทั้งหมดที่แสดงบนหน้าจอเป็นการชั่วคราว
Display()	สั่งให้หน้าจอกลับมาแสดงผลตัวอักษรเดิมใหม่อีกครั้งหลังจากที่สั่ง noDisplay ไปก่อนหน้านี้

การเขียนโปรแกรมควบคุมจอแอลซีดีในสมัยก่อนมีความยุ่งยาก เนื่องจากผู้ใช้ต้องทราบการทำงานของจอแอลซีดีในระดับล่างอย่างละเอียด แต่ปัจจุบันมีการพัฒนาคลังโปรแกรมสำหรับควบคุมจอแอลซีดีขึ้นมาเพื่ออำนวยความสะดวกให้กับผู้พัฒนาโปรแกรม คลังโปรแกรมที่ว่านี้ คือ LiquidCrystal.h ซึ่งมีฟังก์ชันเตรียมให้ผู้พัฒนาได้เลือกใช้หลายตัวด้วยกัน ดังแสดงในตารางที่ 4.1

ฟังก์ชัน LiquidCrystal() ใช้ในการสร้างตัวแปรชนิด LiquidCrystal ขึ้นมาในระ บบ โดยจะต้องมีการป้อนค่าพารามิเตอร์ เข้ามาในรูปแบบ LiquidCrystal(rs, en, d4, d5, d6, d7) มีรายละเอียดดังนี้

- rs คือ หมายเลขขาเชื่อมต่อของบอร์ด Arduino ที่ผู้ใช้ทำการเชื่อมต่อกับขา RS ของมอดูลแอลซีดี
- en คือ หมายเลขขาเชื่อมต่อของบอร์ด Arduino ที่ผู้ใช้เชื่อมต่อกับขา enable ของมอดูลแอลซีดี
- d4-d7 หมายเลขขาเชื่อมต่อของบอร์ด Arduino จำนวน 4 ขา ที่ผู้ใช้เชื่อมต่อกับขา d4-d7 ของมอดูลแอลซีดี เพื่อรับข้อมูลจากตัวประมวลผลแบบขนานขนาด 4 บิต

```
1  #include <LiquidCrystal.h>
2  LiquidCrystal lcd1(8, 9, 10, 11, 12, 13);     // RS, E, D4, D5, D6, D7
3
4  void setup() {
5    lcd1.begin(16, 2);                          // จอกว้าง 16 ตัวอักษร 2 บรรทัด
6    lcd1.clear();                               // ล้างหน้าจอ
7  }
8
9  void loop() {
10   lcd1.setCursor(0, 0);
11   lcd1.print("    Hello TEST!!!!");
12   lcd1.setCursor(0, 1);
13   lcd1.print("Department of Computer Engineering");
14   delay(2000);
15   for (int i=0;i<18;i++)
16   {
17     lcd1.scrollDisplayLeft();
18     delay(200);
19   }
```

รูปที่ 12.12 ตัวอย่างการเขียนโปรแกรมเพื่อแสดงผลสู่จอแอลซีดี

```
20    delay(2000);
21    for (int i=0;i<18;i++)
22    {
23      lcd1.scrollDisplayRight();
24      delay(200);
25    }
26    delay(2000);
27    for (int i=0;i<3;i++)
28    {
29      lcd1.noDisplay();
30      delay(500);
31      lcd1.display();
32      delay(500);
33    }
34    delay(3000);                    // หน่วงเวลา 3 วินาที
35    lcd1.clear();                   // ล้างหน้าจอ
36  }
```

รูปที่ 12.12 ตัวอย่างการเขียนโปรแกรมเพื่อแสดงผลสู่จอแอลซีดี (ต่อ)

จากโค้ดโปรแกรมในรูปที่ 12.12 จะเห็นว่าก่อนการใช้งานจะต้องมีการสร้างตัวแปรชนิด LiquidCrystal ขึ้นมาในระบบเสียก่อน ดังโค้ดบรรทัดที่ 2 ซึ่งจะเห็นว่า การสั่ง LiquidCrystal **lcd1**(8, 9, 10, 11, 12, 13) เป็นการสร้างตัวแปรชื่อ lcd1 ทำการแมปขา rs, en, d4, d5, d6, d7 ของมอดูลแอลซีดีเข้ากับขา 8, 9, 10, 11, 12, 13 ตามลำดับ หลังจากนั้นจะต้องมีการสั่งให้เริ่มต้นการทำงานของ lcd1 ด้วยการสั่ง lcd1.begin(16, 2) ซึ่งหมายถึงการเริ่มใช้งานมอดูลแอลซีดีในแบบหน้าจอ 2 บรรทัด แต่ละบรรทัดมีความยาว 16 ตัวอักษร ดังโค้ดโปรแกรมบรรทัดที่ 5 ส่วนโค้ดโปรแกรมบรรทัดที่ 6 มีการสั่งให้ลบหน้าจอทั้งหมดด้วยฟังก์ชัน clear

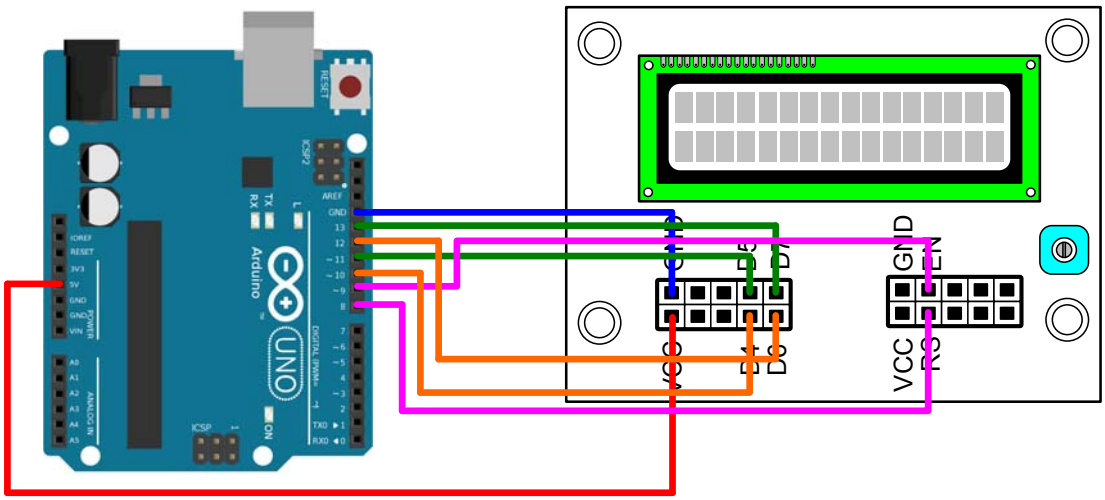
ในการส่งข้อความให้แสดงผลบนจอแอลซีดี สามารถทำได้โดยการเรียกใช้ฟังก์ชัน print ดังโค้ดโปรแกรมบรรทัดที่ 11 โดยฟังก์ชัน print ของคลังโปรแกรม LiquidCrystal.h จะทำการส่งตัวข้อความมาเก็บในบัฟเฟอร์ของมอดูลแอลซีดี และจอแอลซีดีจะนำออกแสดงผลเริ่มต้นตั้งแต่ตำแหน่งเคอร์เซอร์ปัจจุบันที่หน้าจอชี้อยู่ ดังนั้นก่อนที่จะใช้งานบางครั้งอาจจำต้องมีการตั้งค่าตำแหน่งเคอร์เซอร์ใหม่เสียก่อน ดังตัวอย่างโค้ดบรรทัดที่ 11 ซึ่งจะเห็นว่ามีการตั้งเคอร์เซอร์ให้อยู่ตำแหน่งซ้ายมือสุดของของในบรรทัดล่างของหน้าจอด้วยการสั่ง setCursor(0, 1) ตัวเลข 0 หมายถึงค่าตำแหน่งที่ 0 ซึ่งก็คือตำแหน่งแรกด้านซ้ายมือสุดภายในบรรทัดที่ระบุ ส่วนค่าถัดมา คือ เลข 1 แสดงถึงบรรทัดล่างของหน้าจอ (บรรทัดบนจะมีหมายเลขบรรทัดเท่ากับศูนย์)

โค้ดตัวอย่างในบรรทัดที่ 15-19 เป็นการยกตัวอย่างการใช้ลูป for ในการเลื่อนข้อความทั้งหมดที่แสดงบนหน้าจอไปทางซ้ายเป็นจำนวน 18 ครั้ง ส่วนโค้ดบรรทัดที่ 21-25 เป็นการยกตัวอย่างการใช้ฟังก์ชัน scrollDisplayRight ในการเลื่อนข้อความทั้งหมดในหน้าจอไปทางขวาเป็นจำนวน 18 ครั้งเช่นกัน

โค้ดบรรทัดที่ 27-33 เป็นการยกตัวอย่างการสั่งให้ข้อความบนหน้าจอกระพริบติดดับสลับกันครั้งละครึ่งวินาที ด้วยการเรียกฟังก์ชัน noDisplay และ display สลับกัน ก็จะทำให้เกิดการติดและดับของข้อความบนหน้าจอสลับกันไปตามเวลาที่กำหนด ซึ่งทำให้ผู้ใช้ที่มองดูหน้าจออยู่สังเกตเห็นว่าตัวอักษรที่แสดงมีการกระพริบตามจังหวะที่ผู้เขียนโปรแกรมตั้งค่าเอาไว้

การทดลองที่ 12.3

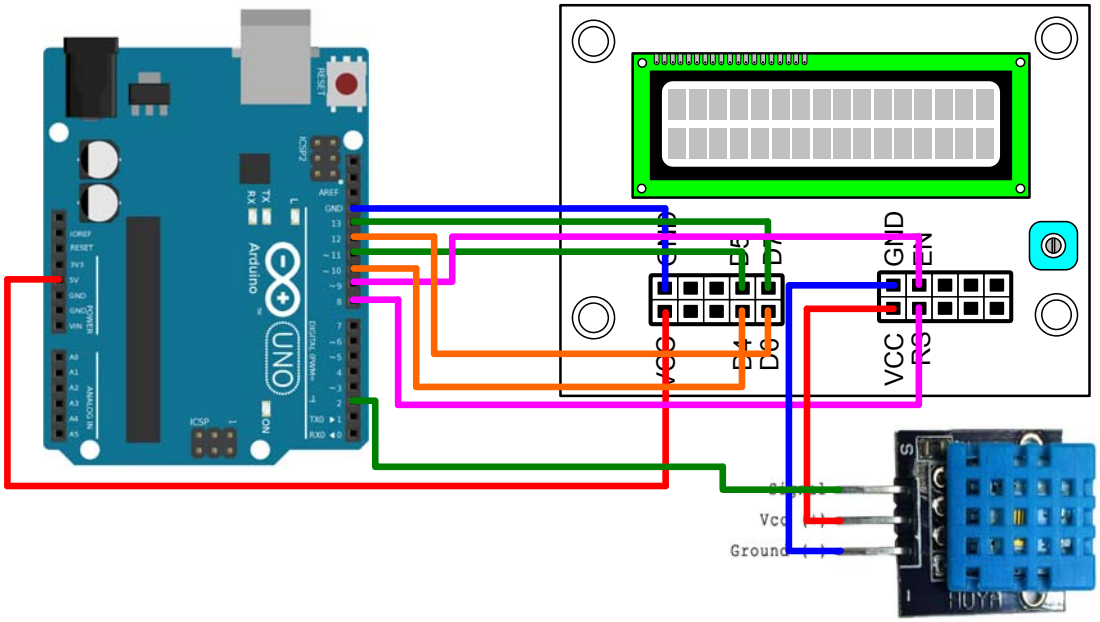
จงต่อมอดูล DHT11 กับบอร์ด Arduino ดังรูปที่ 12.13 จากนั้นนำโค้ดโปรแกรมในรูปที่ 12.12 มาเขียนบน Arduino IDE จากนั้นคอมไพล์และอัพโหลดโปรแกรมลงสู่บอร์ด Arduino และเปิดโปรแกรม Serial Monitor สังเกตการทำงานของโปรแกรมผ่านหน้าจอโปรแกรม Serial Monitor



รูปที่ 12.13 การต่อมอดูลแอลซีดีเข้ากับบอร์ด Arduino ใน การทดลองที่ 12.3

Checkpoint 12.3

จงต่อมอดูล DHT11 และมอดูลแอลซีดีกับบอร์ด Arduino ดังรูปที่ 12.14 จากนั้นจึงเขียนโปรแกรมเพื่ออ่านค่าจากมอดูล DHT11 จากนั้นนำค่าความชื้นและค่าอุณหภูมิออกแสดงผลยังจอแอลซีดี เมื่อวงจรทำงานได้ตามที่กำหนดแล้ว ให้ยกมือเรียกเจ้าหน้าที่หรือ TA เพื่อตรวจ Checkpoint



รูปที่ 12.14 การต่อมอดูลแอลซีดีและมอดูล DHT11 เข้ากับบอร์ด Arduino ในการทดลองที่ 12.3

คำถามหลังการทดลอง

จาก Checkpoint 12.2B ให้เขียนบล็อกไดอะแกรมของวงจรที่ใช้งาน พร้อมโค้ดโปรแกรม และอธิบายการทำงานของโค้ดและแนวคิดการออกแบบโปรแกรมอย่างละเอียด